Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Interfakultären Fachbereich für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

„Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien mittels lazy-learning.“

vorgelegt von

B. Sc. Laura Slimok
104547, UNIGIS MSc Jahrgang 2016

Betreuer/in:
Assoc. Prof. Dr. Gudrun Wallentin

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) – MSc(GIS)”

Köln, 15.09.2019

Danksagung

Ich bedanke mich herzlich bei Assoc. Prof. Dr. Gudrun Wallentin, dass Sie mich während meiner Abschlussarbeit unterstützt und geduldig meine langen Nachrichten beantwortet hat.

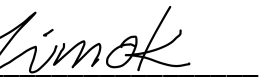
Ein Dankeschön richte ich auch an meinen ehemaligen Arbeitskollegen Matthias Daues und Heiko Kotynski für die fachliche und technische Unterstützung bei dem praktischen Teil meiner Master Thesis.

Von ganzen Herzen möchte ich mich bei meiner Familie und meinem Lebenspartner für die mentale Unterstützung und das Verständnis in der intensiven Zeit bedanken.

Eidesstattliche Erklärung

Ich erkläre hiermit eidesstattlich durch meine eigenhändige Unterschrift, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Stellen, die wörtlich oder inhaltlich den angegebenen Quellen entnommen wurden, sind als solche kenntlich gemacht.

Die vorliegende Arbeit wurde bisher in gleicher oder ähnlicher Form noch nicht als Bachelor-/ Master-/ Diplomarbeit/ Dissertation eingereicht.

Köln, 15.09.2019 

Ort, Datum, Unterschrift

Zusammenfassung

In Deutschland existiert aktuell keine vollständige, digitalisierte Dokumentation des bestehenden Telekommunikations-Kabelnetzes. Ein Grund dafür ist die regionale Aufteilung des Telekommunikationsmarktes auf unterschiedliche private Unternehmen im Rahmen der Privatisierung des Telekommunikations Marktes in den 90-er Jahren.

Die bestehende Netzinfrastruktur-Dokumentation dieser Unternehmen unterscheidet sich in Struktur und Inhalt und basierte meistens auf Papier als Dokumentationsmedium. Mit der Verbreitung von PCs und der digitalen Datenverarbeitung wurde der Bedarf nach der Digitalisierung dieser Dokumentation größer, um die Kabelnetze besser verstehen und diese mit modernen Geoinformationssystemen im Hinblick auf Kosten, Netzabdeckung und Potential analysieren zu können. Daher starteten die Unternehmen unter großen Aufwand die Digitalisierung der Dokumentation. Der Prozess der Digitalisierung war manuell, langsam und teuer, außerdem zeigte sich bei Qualitätsprüfungen, dass die digitalen Daten Fehler aufwiesen und unvollständig waren.

Diese Abschlussarbeit entwickelt einen auf maschinellem Lernen basierenden Ansatz, um die Datenqualität von bestehenden digitalen Daten eines Kabelnetzes zu evaluieren und fehlende Daten vorherzusagen.

Darüberhinaus ist dieser Ansatz so generisch, dass er für jeden Anwendungsfall genutzt werden kann, bei dem eine Metadateninformation eines geographischen Punktes auf eine streckentreue Polylinie übertragen werden soll.

Der Ansatz nutzt das grundlegende Konzept der räumlichen Verbindung und kombiniert es mit einem k-Nächste-Nachbarn Algorithmus aus dem Bereich des maschinellen Lernens.

Der Ansatz und das entstandene Modell werden in einem echten Anwendungsfall mit digitalisierten Daten eines deutschen Telekommunikationsanbietes im Testgebiet Köln getestet und evaluiert.

Die Implementierung nutzt die geographische Software ArcGis Pro für die Datenvorbereitung und Visualisierung, sowie die Programmiersprache Python für den k-Nächsten-Nachbarn Algorithmus.

Abstract (eng.)

In germany a complete digitalized documentation of the existing cable network of germany is missing. A reason for that is the regional division of the telecommunication market covered from different private telecommunication companies during the phase of privatization of the german cable network beginning in nineties. The existing network infrastructure documentation of these companies differ in structure and content and was commonly based on paper. The evolution of PCs and digital data created the need for digitalizing these documentation to get deeper insides and analysis with modern geoinformationsystems about costs, coverage and potential of the cable network. Therefore companies started to invest heavyly into digitalization of the documentation. The process of digitalization was very manual, slow and expensive and quality checks of digitalized data showed errors and incompleteness.

This master thesis developes an approach based on machine learning techniques to evaluate the quality of existing digitalized data and estimate incomplete data based on existing digitalized documentation. Furthermore this generic approach can be used in every use case which transfers a metadata information of a geographical object type point onto a geographical object type polyline. The approach uses the known concept of spatial joins in combination with a k-nearest-neighbor machine learning model.

The approach and evaluation of the machine learning model is tested in a real-world scenario with digitalized cable network point data from a german telecommunication company in the area of cologne.

The implementation of the approach is done in geographical Software ArcGis Pro for data preparation and visualization and programming language Python for the machine learning model.

Keywords:

spatial joins, linear ordering, distance metrics, basic machine learning concepts, lazy learning, k-nearest neighbour algorithm, data preparation, openstreet map

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Exkurs

Ausgangsituation zu der Verfassung der Abschlussarbeit

Die Abschlussarbeit lehnt an ein internes Projekt im Unternehmen Unitymedia NRW GmbH zur Bereitstellung der Information über den Kabelnetzverlauf mittels einer webbasierten Geoinformationssysteme (GIS)-Lösung an. Auf die Einzelheiten des Projektes wird in dieser Abschlussarbeit nicht eingegangen. Folgend werden die Besonderheiten der Entwicklung des Telekommunikationsmarktes in Deutschland dargestellt. Diese sollen ein besseres Verständnis der Problematik und der Notwendigkeit zur Nachbildung der physischen Kabelnetzverläufe schaffen. Eine entscheidende Rolle spielen unterschiedliche Faktoren, wie die Herkunft der Testdaten, die Geschichte des Kabelnetzbetreibers, sowie die Art, aktueller Stand und die Vollständigkeit einer Netzdokumentation. Diese Faktoren werden in der Beschreibung der Geschichte der Unitymedia NRW GmbH dargestellt. Anschließend wird eine aktuelle Entwicklung des Analytischen Kostenmodells für das Anschlussnetz erwähnt. Dieses Projekt beschäftigt sich mit einer Modellierung von Kabelnetzverläufen mit dem Ziel die Anschlusskosten je nach Anschlusstechnik und Verfügbarkeit eines Netzbetreibers nachträglich zu ermitteln. Das ist ein Beispiel für einen Anwendungsfall, wo eine vollständige Netzdokumentation der Telekommunikationskabelnetze das Berechnungsmodell optimieren würde. Abschließend werden die in der Abschlussarbeit verwendete Bezeichnungen der Netzelemente kurz dargestellt.

Mit der fortschreitenden Marktsättigung in der Telekommunikationsbranche im Einzugsgebiet der Unitymedia NRW GmbH wurde die Ermittlung der potentiellen Neukunden schwieriger als bei hohen Anfrage nach einem Internetanschluss auf dem Markt. Die bisherigen Methoden der Potential-Analysen mussten optimiert werden. Ein neuer Marketingeinsatz der gezielten Ansprache bedarf nicht nur einer Definition der Zielgruppen, sondern auch eine präzise Prüfung der Telefon-Anschlussmöglichkeit. Das

Wissen über den Verlauf von eigener Infrastruktur lässt nicht nur genaue Kosten-Nutzen-Rechnungen für eine Marketingmaßnahme aufstellen, sondern vermittelt den potentiellen Kunden einen seriösen Auftritt des Unternehmens und wirbt mit einem auf den Kunden genau zugeschnittenen Angebot, was den Vertragsabschluss wahrscheinlicher macht.

Das Kerngeschäft der Unitymedia NRW GmbH von seinen Anfängen an war der Vertrieb von Kabelanschlüssen in den Bereichen private Nutzer und Wohnungswirtschaft. Seit 2012 wurde das Angebot auf den Geschäftskunden Bereich ausgeweitet und die Abteilung Business to Business gegründet. Im Laufe der Zeit ist die Palette der Produkte gewachsen. Neben Kabelanschlüssen sind weitere Bausteine, wie Telefonie, Internet und Mobile Anschlüsse eingeführt worden. Speziell für die Geschäftskunden mit Gewerbe sind neue Lösungen, wie der schnelle Glassfasernetz-Ausbau hinzugekommen. Daraus hat sich ein neuer Zweig der Netzerweiterung entwickelt. Die Verantwortung über einen sinnvollen Neuausbau lag nun in der IT-Abteilung, wo die ersten GIS unterstützten Potential-Analysen durchgeführt worden sind. Im Rahmen der Methodenoptimierung für die schon angesprochene gezielte Marketinganspräche zur Neukundengewinnung ist die Notwendigkeit der Erhöhung der Datenqualität der Kabelnetzinfrastruktur festgestellt worden. Das Unitymedia Kabelnetz wurde ursprünglich von der Deutschen Post und später der Deutschen Telekom verlegt worden. Im Rahmen der Deregulierung des Telekommunikationsmarktes wurde das Telekom-Monopol aufgehoben (Kurth, 2003). Die Konsequenz des neuen Gesetzes war in der Phase der Liberalisierung des deutschen Telekommunikationsmarktes die Entstehung von neuen Telekommunikationsunternehmen als neue Anbieter von TV- und Telefonie-Produkten. *(Haucap and Coenen, 2010)*

Geschichte des Unternehmens Unitymedia NRW GmbH

Die bestehende Kabelnetzinfrastruktur wurde von verschiedenen Unternehmen von der Telekom aufgekauft und im eigenen Einzugsgebiet nach Bedarf mit eigenen Mitteln und Technik erweitert. Die Geschichte des Unternehmens fängt im Jahr 2001 als das Unternehmen ish bestehend aus den ehemaligen DTAG-Kabelnetze-NRW und -BW gegründet wird. Ein Jahr später wurde die Infrastruktur nach Bundesländern aufgeteilt, was zu einer Gründung von einem neuen Unternehmen Kabel BW für das Einzugsgebiet in Baden-Württemberg führte. Das Unternehmen ish blieb der Netzbetreiber für das Bundesland Nordrhein-Westfalen, dazu kommt das Bundesland Hessen. Im Jahr 2005 nach einer Übernahme von ish durch das Unternehmen easy entsteht Unitymedia mit der Zuständigkeit für die Bundesländer Nordrhein-Westfalen und Hessen. Im Laufe der Zeit kam es zu weiteren Übernahmen von regionalen Anbietern, sowie Gründungen von weiteren Tochtergesellschaften. Im Jahr 2015 entsteht die finale Unternehmensstruktur der Unitymedia als Tochtergesellschaft der Liberty Global Group und der Kabelnetzinfrastruktur von drei deutschen Bundesländern: Nordrhein-Westfalen, Hessen und Baden-Württemberg aufgrund einer Fusion der Unternehmen Unitymedia NRW GmbH und Kabel BW. Aus Sicht der Technik blieb die Kabelnetzinfrastruktur unverändert, jedoch wurde die Dokumentation der Kabelnetzverläufe in jedem Unternehmen anders strukturiert und gehandhabt.

Die Dokumentation der Baupläne des Kabelnetzes aus dem Aufkauf der ehemaligen DTAG-Kabelnetzen-NRW und -BW wurden in Papierform übermittelt, welche eingescannt und als PDF Dokumente archiviert wurden. Die erste Digitalisierung der Netzinfrastruktur erfolgte mit der Einführung von Netzinfrastruktursystem (NIS) für das Land Baden-Württemberg und Netzdokumentationssystem (NDOS) unterstützt durch die Software Smallworld für die Bundesländer Nordrhein-Westfalen und Hessen. Die Altbestände der DTAG-Kabelnetze wurden manuell nachgezeichnet und somit vektorisiert. Daher, dass die finanziellen Mittel für das Projekt frühzeitig ausgeschöpft wurden, sind die Kabelnetzverläufe nicht vollständig digitalisiert worden. Diese Abschlussarbeit ist aufgrund

der Überlegung: Wie kann man eine fehlende und unvollständige Netzdokumentation in kurzer Zeit und mit geringem Aufwand nachträglich digitalisieren? injiziert worden. Das Beantworten der praxisbezogenen Fragestellung erfolgt in dem praktischen Teil der Umsetzung von einem *lazy learning* Modells mit echten Daten der Unitymedia NRW GmbH.

Analytisches Kostenmodell für das Anschlussnetz

Ein Versorgungsnetz ist als ein Transportmedium zu sehen. In den Bereichen Energie, Telekommunikation und Schienenverkehr haben Netze eine Gemeinsamkeit, nämlich die Funktion des Transports. Das Telekommunikationsnetz hat die Aufgabe Nachrichten zu transportieren. Angefangen mit einem analogen Kupfernetz verlagerte sich die Aufgabe auf die virtuellen Netze. Der vereinfachte Zugang zu Informationen dank Internet führte zu einer Überlastung der bestehenden Telekommunikationsnetze. Durch das Zusammenschalten von mehreren Netzen können die Kapazitäten besser ausgelastet werden und die Produktivität aller Netze kann gesteigert werden. Insbesondere können Nachfragespitzen vermieden werden und die Lastkapazitäten kleiner gehalten werden. Das Phänomen der Netzüberlastung ist auf das wachsende Kundeneinzugsgebiet zurückzuführen und den steigenden wirtschaftlichen Nutzen von dem Angebot an Telekommunikationsdiensten. (Braun, 2003). Daraus folgt die Schlussfolgerung, dass die Kenntnis über die genaue Lage und Auslastung des eigenen Netzes die Zusammenschaltung von Netzen ermöglicht. Die politische Aufgabe der Regulierung des Telekommunikationsmarktes durch die Bundesnetzagentur hat dazu geführt, dass ein generischer Modellierungsansatz für die Kosten des Teilnehmeranschlussnetzes entwickelt worden ist. Im Rahmen einer Studie für die Bundesnetzagentur durchgeführt von WIK-Consult GmbH wurden die Modellierungsrahmen detailliert beschrieben und in dem Bericht: „Analytisches Kostenmodell für das Anschlussnetz AKM-AN Version 3.0“ veröffentlicht. (Kulenkampff, 2019) In einer Informationsveranstaltung am 15.05.2018 in

Bonn wurde das Analytischen Kostenmodell Anschlussnetz AKM- AN Version 3.0 Weiterentwicklung im Kontext der ND & KRM-Empfehlung präsentiert. (Kulenkampff, 2018)

Die Abbildung 1 fasst die Vorgehensweise zusammen und zeigt auf, dass das Modell auf einem k-*Nearest-Neighbour* Algorithmus unter Verwendung der Rechnung der minimalen Distanz basiert. Im ersten Schritt wird die Erstreckung und die Auslastung der Verzweigerbereiche festgelegt. Die gebildeten Cluster werden in einem Optimierungsschritt mittels einem modifizierten-Minimalen-Spannbaum Algorithmus unter Einfluss von spezifischen Strukturparametern der Trassen angepasst. Die dimensionierte Netzelemente werden durch spezifische Investitionsparameter zu einem Mengengerüst umgewandelt und anschließend unter Berücksichtigung von ökonomischen Parametern werden Kabelnetz spezifische Kosten bestimmt.

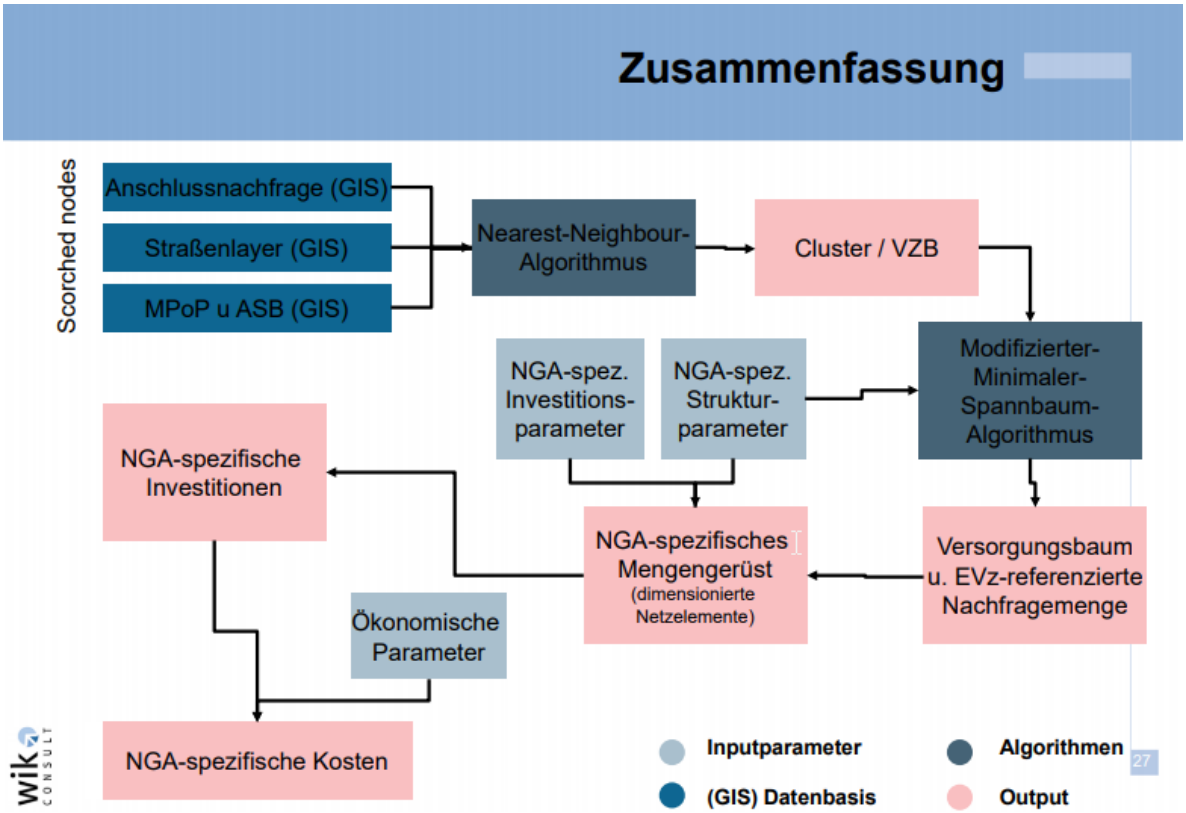


Abbildung 1: Zusammenfassung des Analytischen Kostenmodells Anschlussnetz AKM-AN Version 3.0, (Kulenkampff, 2018, p. 27)

Die Verzweigerbereichsbildung wird in der Praxis mit Vornoi-Polygonen gekennzeichnet. Für neue Kunden Anfragen kann man auf dieser Weise schnell sehen an welchem vorhandenen Kabel der Neukunde angeschlossen werden könnte und ob es noch freie Kapazität in der Leitung gibt. Das Abbilden der Vornoi-Polygone kann durch den Einsatz des *k-Nearest Neighbor* Algorithmus mit k=1 erreicht werden. Im Rahmen dieser Abschlussarbeit wird der *k-Nearest Neighbor* Algorithmus eingesetzt. Die getroffenen Annahmen zu der Bildung von Verzweigerbereichen wurden berücksichtigt und die Georeferenzierung des *OpenStreetMap* Straßenlayers angepasst.

Netzelemente

Die Netzinfrastruktur der Telefonanschlüsse mit der Kabelgebundenen Technologie kann in Segmente und Knoten in einer hierarchischen Ebene unterteilt werden (siehe dazu Abbildung 2). Die Rangordnungen sind von einem Anschluss-Objekt, hier z.B. ein Haus, bezeichnet als Endverzweiger über die Hauszuführung und den Verzweigerkabelsegment bis zu einem Verzweigerknoten verbunden mit einem Hauptkabelsegment bis zu einem städtischen Verteiler *Metropolitan Point of Presence* (MPoP) dargestellt. Bestandteil der praktischen Anwendung dieser Abschlussarbeit sind Erdkabellinien eines Kupferkabels, welche nach der beschriebenen Struktur den Verzweigerkabelsegmenten verbunden mit Hauptkabelsegmenten einzuordnen sind. Diese verfügen über eine technische Identifikationsnummer (TECH_ID), welche für jeden zusammenhängenden Abschnitt von einem Segment eindeutig ist.

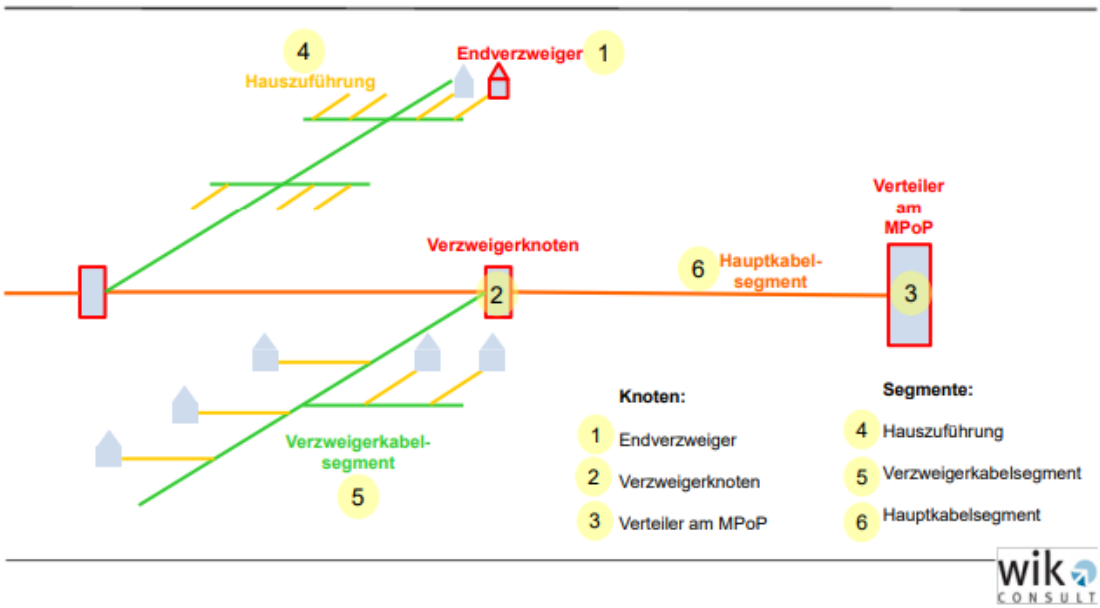


Abbildung 2: Investitionswertmodelierung und zu Grunde liegende Netzelemente, (Kulenkampff, 2019, p. 19)

Die D-Linien also Erdkabel von der Montagestelle bis zum Endverzweiger, sowie die Verkabelung innerhalb eines Gebäudes werden im Rahmen dieser Abschlussarbeit nicht berücksichtigt. (Kulenkampff, 2019)

1. Einleitung

1.1 Hintergründe der Abschlussarbeit und Motivation

Das räumliche Konzept der Distanz als Relation zwischen Objekten wird im Rahmen dieser Abschlussarbeit in einen neuen Kontext gesetzt.

Diese Abschlussarbeit beschäftigt sich mit einer möglichen Vorgehensweise zur nachträglichen Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien. Auf Basis eines *machine learning* Ansatzes wird eine Vorhersage von der technischen Identifikationsnummer der hausgenauen Punkte auf die streckentreue Polylinien getroffen. Die für dieses Vorhaben genutzte geoinformatische Operation baut auf der räumlichen Verbindung (sog. *spatial join*) zweier unterschiedlichen Datenmengen auf. Mit Hilfe der definierten Teilforschungsfragen (siehe dazu Absatz 1.4 Forschungsfrage) wird geprüft, ob ein *lazy learning* Modell, genauer ein *k-Nearest Neighbor Join* Algorithmus für einen solchen Anwendungsfall geeignet ist. Die Frage wird unter praktischer Anwendung des Modells mit realen Daten aus der Telekommunikationsbranche untersucht. Das Ergebnis soll einen Grundbaustein zu einer automatisierten Nachbildung von Versorgungsnetzen in unterschiedlichen Anwendungsbereichen legen und somit als ein generisches Vorgehensmodell zur Umsetzung eines solchen Anwendungsfalls dienen. Die Anwendung der Grundkonzepte der Geographie anhand des Fallbeispiels aus der Telekommunikationsbranche zeigt die Allgemeingültigkeit und Übertragbarkeit der Ansätze.

Heutige Telekommunikationsunternehmen stehen vor der Herausforderung den Bestand ihres gesamten Kabelnetzes zu digitalisieren, um wettbewerbsfähig zu bleiben.
Die Notwendigkeit der genauen Kenntnis des Kabelnetzes und des -verlaufes ergibt sich sowohl aus unternehmensinternen ökonomischen Überlegungen, als auch aufgrund von unternehmensexternen Marktanforderungen oder Wettbewerbsdruck. Die Unternehmen haben zur Gewinnmaximierung das Interesse möglichst kostengünstig und flächendeckend neue Haushalte an das Kabelnetz anzuschließen sowie das bestehende Kabelnetz zu

betreiben. Die genaue Kenntnis der Kabelnetze und –verläufe in digitaler Form ist für eine moderne Planung sowie die Durchführung und den Einsatz neuer Technologien oder Verfahren im Rahmen von Geoinformationssystemen essentiell.
Aber auch unternehmensexterne Anforderungen führen dazu, dass die Kabelnetze digital dokumentiert und ausgewertet werden müssen, so ist ein nachweisbarer Kabelnetzbestand im Rahmen der politischen Diskussion zur Netzabdeckung in Deutschland für den Dialog mit der Politik und Behörden wie der Bundesnetzagentur notwendig. Darüber hinaus lassen sich der Wettbewerb und die Marktchancen in unterschiedlichen regionalen Gebieten mithilfe der Kenntnis des eigenen Kabelnetzes bewerten.

Die bestehende Dokumentation von Versorgungsnetzen basiert historisch bedingt auf Papier als Dokumentationsmedium. Die Telekommunikationsunternehmen haben großen Aufwand betrieben diese Dokumentation zu digitalisieren, um diese maschinenlesbar, durchsuchbar und besser nutzbar zu machen.
Im Rahmen der manuellen Digitalisierung sind menschliche Übertragungsfehler gemacht worden, sowie aufgefallen, dass die ursprüngliche Dokumentation fehlerhaft bzw. unvollständig ist.
Das manuelle Auffinden solcher Fehler oder Dokumentationslücken ist vom Aufwand vergleichbar mit einer erneuten Digitalisierung der ursprünglichen Papierdokumentation, so dass die Notwendigkeit nach einer automatisierten Unterstützung zum Auffinden dieser Unregelmäßigkeiten besteht. Die Entwicklung und Anwendung eines solchen Verfahrens auf Echtdaten ist das Ziel dieser Abschlussarbeit.
Die Problemstellung liegt darin die in einer relationalen Datenbank bevorratete kundenbezogene Informationen, sowie die technischen Eigenschaften der Netzinfrastruktur mithilfe vektorisierter Kabelnetzstrecken miteinander zu verbinden und diese mit der Vorhersage über das Kabelnetz des entwickelten Modells abzugleichen. Somit ergibt sich eine bewertbare Plausibilität der bereits vorhandenen Daten, die einen gezielten Hinweis auf Datenfehler gibt und eine manuelle Prüfung rechtfertigen kann.

1.2 Stand der Forschung

Der erste Hauptsatz in der Geographie besagt, dass Alles mit Allem anderen in einer Beziehung zueinandersteht, aber je näher etwas zueinander ist, desto mehr steht es miteinander in Beziehung (Tobler, 1970). Miller (2004) sagt aus, dass Tobler den Grundsatz nicht allumfänglich hergeleitet hat, sondern einen empirischen Zusammenhang gemessen hat. Für Miller (2004) geht das Konstrukt der Beziehung oder Relation innerhalb eines geographischen Raumes über die reine Betrachtung der Nähe durch das Abmessen von Distanzen hinaus. Eine Relation kann auf unterschiedlichen Ebenen stattfinden und zeichnet sich durch eine positive oder negative Korrelation zwischen Objekten aus. Er betont, dass Korrelation keine Kausalität ausdrückt, und das sich eine Relation in sichtbaren, aber auch in nicht oder weniger sichtbaren Attributen und Eigenschaften ausdrückt. Dieser Rahmenbedingungen sollte man sich bei der Betrachtung von räumlichen Beziehungen immer bewusst sein. Diese Kernaussagen sind bis heute gültig und finden mit der Weiterentwicklung von Techniken zur räumlichen Analyse und vor allem im Bereich der Geoinformationssystem (GIS) gestützten Analysen eine große Zustimmung und Relevanz (vgl. (Goodchild, 1988)). Die Definition von Distanz als eine Art von Relation hat demzufolge zur Entwicklung von anspruchsvollen Technologien der geographischen Analysen unter Nutzung von alternativen räumlichen Metriken zur Beschreibung von Objekten im Raum und Zeit geführt. (Miller, 2004).

Obwohl eine Relation nicht nur durch die Nähe ausgedrückt werden kann, spielt sie doch eine wichtige Rolle um auch globale komplexe Interaktionen zwischen Entitäten zu beschreiben. „Near is Beautiful“ schreibt Miller (2004, p. 287) und argumentiert es mit *Complexity theory und Complex adaptive systems (CAS) theory*. Komplexität entwickelt sich dieser Theorie folgend auf Basis von lokalen Interaktionen oder Beziehungen zu einem großen Gesamtkonstrukt. Auch Flake (1998) und Watts (1999) begründen das aussagekräftige Ergebnisse bei der Betrachtung von Nähe und Distanz als Interaktion zur

Entstehung von umfangreichen Wechselwirkungen im räumlichen System entstehen können.
Im Rahmen dieser Abschlussarbeit wird Toblers Grundsatz gefolgt und auch die Nähe in den Vordergrund gestellt, um Relationen zwischen Objekten zu erzeugen und zu untersuchen. Diese Nähe wird durch ein oder mehrere Distanzmaße, welche in einem Raum definiert werden müssen, ausgedrückt.

Die verwendeten Daten sind topologische Objekte, welche in einer räumlichen Relation zueinanderstehen. Laut Gatrell (1983) ist die Definition von Raum jegliche Zusammensetzung von Objekten, welche in einer Relation zueinander stehen. Der Raum muss grundsätzlich nicht metrisch sein, aber die Definition einer metrischen Distanz innerhalb des Raumes macht diesen zum metrischen Raum. (Gatrell, 1983, pp. 42-43) Die Beziehungen zwischen zwei topologischen Objekten können eine nicht metrische Wechselwirkung haben, jedoch bei einer Beschreibung und einem Vergleich von Nähe von mehreren Objekten ist die metrische Distanz „ein sinnvolles Ähnlichkeitskriterium“ […]. „Das anschaulichste Maß für die Ähnlichkeit ist offensichtlich die euklidische Distanz“ (Bahrenberg, Giese et al., 2008a, p. 262). Dieses Maß ist in einem zweidimensionalen Raum als eine Erweiterung des Pythagoras Satzes definiert und liefert die „Luftlinienentfernung“ zwischen zwei Objekten. Weitere bereits in den ersten Geographischen Forschungsarbeiten diskutierte Maße sind die Manhattan- und Minkowski-Distanz. (Gatrell, 1983) Die sog. “City-Block”- Distanz (Manhattan) misst die Länge des Weges entlang der Koordinatenachsen der untersuchten Topologien, analog zur einem rechtwinkligen Straßensystem, wie es in der namensgebenden amerikanischen Stadt Manhattan üblich ist. Beide Metriken gehören zu den Minkowski Distanzen. Die Auswahl eines metrischen Maßes wirkt sich auf die Ergebnisse der Messung von Ähnlichkeit zweier oder mehrerer topologischer Objekte aus.
Daneben ist es durchaus üblich eigene, anwendungsfallspezifische mathematische Operationen zu definieren, welche zu einer Gewichtung von Distanzen führen, um das Ergebnis bewusst zu beeinflussen. (Bahrenberg, Giese et al., 2008a, p. 263)

Gatrell (1983) beschreibt auch nicht metrische Konzepte von Distanz und zählt dazu „*time*“, „*economic*“, „*cognitive*“ und „*social distance*“. In seiner späteren Arbeit beschreibt er diese aber als ungenau und unsicher für den Einsatz bei räumlichen Analysen. (Gatrell, 1991, p. 122) Diese werden daher in dieser Abschlussarbeit nicht näher berücksichtigt.
Die „euklidische“ und die „Manhattan“ Distanz werden in der Abschlussarbeit als metrische Distanzen eingesetzt und im methodischen Teil weiter definiert.
Da diese Teils zu sehr unterschiedlichen Ergebnissen führen können, wird der Einfluss der Distanzmetrik auf das Ergebnis in dieser Arbeit gesondert untersucht und findet sich als Teilfrage der Forschungsfrage wieder.

„*Armstrong (1988) has looked empirically at how different definitions of distance influence the results of such location (and location-allocation) models, using an example from the southeastern United States. Having defined a set of 26 demand nodes and 10 facilities to be allocated he then computes distances along the rad network, where this is digitized a different scales. He compares these estimates with Euclidean and Manhattan distances and demonstrates how the results change quite dramatically as different measures of spatial separation are adopted.“* (Gatrell, 1991, p. 122)

Mit der Digitalisierung und der systematischen Erfassung von unterschiedlichsten Informationen zu Objekten ist die schiere Menge der Informationen vom Menschen nicht mehr komplett zu erfassen und zu verarbeiten. Im Zuge dessen haben sich mit dem vermehrten Einsatz von Computer und Datenbanken auch die Ansätze und Weiterentwicklungen in der geographischen Analyse verändert. Ziel ist es (semi-) automatisiert Relationen zu erkennen, ohne dass der Mensch diese Relation genau definiert, erkannt hat oder in der Lage wäre diese zu erkennen.
Eine weitere Betrachtungsweise von räumlichen Beziehungen von zwei Entitäten kommt daher aus dem Bereich des Datenbankmanagements. Wie Ng and Han (1994) in seiner Arbeit definieren, ist Data Mining ein Werkzeug zur Suche nach verborgenen Mustern,

welche sich in großen Datenmenge erkennen lassen (Ng and Han, 1994). „*Spatial Data Mining*“ kristallisierte sich aus der Massen-Datenverarbeitung im Mehrdimensionalen Raum heraus und verzeichnet einen hohen Datenvolumen Zuwachs. „Dieses Wachstum geht weit über die menschlichen Fähigkeiten hinaus, um die Datenbanken zu analysieren, um implizite Regelmäßigkeiten, Regeln oder Cluster zu finden, die in den Daten verborgen sind.“ (Ester, Kriegel et al., 1997, p. 47) Daneben ermöglicht der Einsatz von sog. *machine learning* Methoden zum Beispiel die Analyse von weit entfernten Objekten mit großen Distanzen, wie andere Planeten in der Galaxis. Foroutan and Zimbelman entwickelten einen sog. „*Self-Organizing Maps (SOM)*“ Algorithmus zum automatisierten Feststellen von klimatischen Veränderungen aus hoch-aufgelösten Satellitenbildern aus dem Weltall (Foroutan and Zimbelman, 2017).

Ein notwendiges Grundkonzept für die automatisierte Suche und Arbeit mit digitalisierten räumlichen Daten, ist die räumliche Verbindung. Diese kann vereinfacht folgend zusammengefasst werden: In zwei unterschiedlichen Datensätzen mit Rechtecken werden alle Paare, die sich untereinander überschneiden innerhalb des Datensatzes gekennzeichnet. Das grundsätzliche Problem der räumlichen Verbindung ist auch als räumliche Überlagerungsverbindung bekannt. Die Möglichkeiten der Verbindungsarten sind unzählig und werden anhand von auffindbaren Beziehungen zwischen den Datensätzen beschrieben, die je nach Kombination von unterschiedlichen räumlichen Objektarten variieren können. Die meist vorkommenden Relationen zwischen Objekten sind die Überschneidung, die Nähe, die Einzäunung oder auch Richtungsbeziehungen. Die räumliche Verbindung kann innerhalb von einem Datensatz aber auch zwischen zwei oder mehr als zwei Datensätzen und Objektarten untersucht werden (Jacox and Samet, 2007, p. 2).

Die Operation der räumlichen Verbindung zwischen Objekten muss in Abhängigkeit der Anzahl der Objekte mehrfach ausgeführt werden, da jedes Objekt mit jedem anderen

Objekt verbunden werden kann oder zumindest geprüft werden muss, ob eine Verbindung besteht. Somit ergibt sich ein selbst für heutige Computer sehr Laufzeit- und rechenintensives Problem aufgrund der Maße der möglichen Verbindungen und Daten. Daher ist es sehr verständlich, dass sich ein Großteil der wissenschaftlichen Arbeiten mit Laufzeitoptimierungen von räumlichen Verbindungen und deren Anwendungen befasst. Diese zielen einerseits auf grundlegende Methoden zur Optimierung ab, als auch auf sehr technikspezifische Optimierungen, die nur für den Einsatz in einem Anwendungsfeld oder einer Infrastruktur geeignet sind.
Eine sehr grundsätzliche Optimierung war die Überlegung, dass versucht werden sollte, viele einzelne Verbindungsoperationen durch eine einzige Operation zur Verbindung aller Objekte zu ersetzen.

„*Recently, it has been recognized that many algorithms of similarity search (Agrawal et al. 1995) and data mining (Böhm et al. 2000) can be based on top of a single join query instead of many similarity queries. Thus, a high number of single similarity queries is replaced by a single run of a similarity join. The most well-known form of the similarity join is the distance range join. E.g. in (Böhm et al. 2000), the incremental distance join (Hjaltason, Samet 1998).*“ (Böhm and Krebs, 2004, p. 728)

Daneben ist es natürlich auch von Vorteil, wenn man versucht die zu betrachtende Objektmenge so früh wie möglich auf nur potentiell relevante Objekte einzuschränken, um Verbindungsoperationen wie z.B. die Distanzermittlung nicht auf Objekten durchführen zu müssen, bei denen schon vorher absehbar ist, dass diese nicht in dem gültigen Ergebnisraum enthalten sein werden (z.B. kürzeste Distanz). Bei diesen Ansätzen hat sich eine grundlegende Struktur entwickelt, die einem zweistufigen Prozess entspricht. Der erste Prozessschritt ist das Filtern, Sortieren oder andere Arten von Operationen auf der Gesamtobjektmenge zur Vorbereitung der Verbindungsoperation. Der zweite Prozessschritt ist dann erst die eigentliche Verbindungsoperation, welche nicht mehr auf

der Gesamtobjektmenge angewendet werden muss oder der aufgrund von Sortierung, Suchstrategie oder anderer Metainformationen wesentlich schneller möglich ist.
Eine grundlegende Arbeit dazu wurde von Orenstein (1989) mit den ersten Versuche zur Optimierung durch das Filtern der Treffer einer räumlichen Verbindung und Verfeinern der Ergebnisse getätigt. Die Ergebnisse der Versuche bestätigten die Notwendigkeit der Anwendung einer zweistufigen Strategie bei räumlichen Abfragen der Nachbarschaftsbeziehungen durch einen Filterungsschritt (*Filter Step*) und Verfeinerungsschritt (*Raffinement Step*). Die Verbesserung der Ergebnisse wird damit erreicht, dass die Paare einer Verbindung, welche die Bedingungen nicht erfüllen übersprungen werden (Orenstein, 1989). Das Filtern der Datensätze ist eng mit den Methoden der Sortierung der Daten verbunden. Dieser Prozessschritt der Verbindung überprüft die Relation untereinander und ordnet diese in entsprechender Abfolge. Auch Mishra and Eich (1992) suchten nach einem effizienten Algorithmus zum Kombinieren von in Beziehung stehenden eindimensionalen Datensätzen. Unterschiedliche Verbindungsarten in den relationalen Datenbanken wurden als eine algebraische Operation zur Zuordnung von verwandten Tupel anhand von verschiedenen Attributen der Daten beschrieben.
Diesen Überlegungen folgend wird auch in dieser Abschlussarbeit ein Schritt zur Sortierung und Filterung von der Gesamtobjektmenge eingefügt, um die notwendigen Rechenoperationen und die Laufzeit sinnvoll zu verkürzen.

Die Beziehungsarten von unterschiedlichen Objekten aus verschiedenen Datenquellen unterliegen dem Problem der Mehrdimensionalität. Wie (Jacox and Samet, 2007) diskutiert, ist die Mehrdimensionalität für die räumliche Verbindung Komplex und bedarf zusätzlicher geographischen Operationen zur Verfeinerung der Ergebnisse. Diese sind aufgrund der Bedeutung für den angewendeten k-*Nearest Neighbour* Algorithmus separat in Kapitel 2.1 Spatial join, die räumliche Verbindung näher beschrieben.

Der Einsatz von *machine learning* Verfahren (aus Daten selbstständig lernende Algorithmen) im Bereich von Geoinformationssystemen ergibt sich nicht zuletzt aus der oben beschriebenen Mehrdimensionalität und der daraus folgenden Komplexität, die für den Menschen nur noch schwer zu erfassen ist. Man unterscheidet grundsätzlich verschiedene Arten von *machine learning* Verfahren, dass *supervised* und *unsupervised learning*, sowie die Kombination dieser beiden bekannt als *semi-supervised learning* (siehe dazu (Torres-Moreno, Bougrain et al., 2009)). Diese grundlegendenden Konzepte werden in Kapitel 2.2 Das *lazy learning* Modell zur Ableitung von Kabelverläufen erläutert und um ein weiteres Unterscheidungsmerkmal von *machine learning* Verfahren dem *lazy learning* ergänzt. Da ein Großteil der Konzepte theoretisch und praktisch im Rahmen dieser Abschlussarbeit genutzt wird, wird an dieser Stelle auf die Definition und Einordnung verzichtet und diese an der jeweils relevanten Stelle bei der Nutzung in der Abschlussarbeit nachgeholt, um eine mehrfache Beschreibung zu vermeiden.

1.3 Zielsetzung und Relevanz

Zielsetzung dieser Arbeit ist die bekannte Klassifikationsmethode k-*Nearest Neighbor Classifier* in ein neues Studiengebiet, nämlich in die Telekommunikationsbranche zu übertragen. Die in dieser Abschlussarbeit beschriebene Vorgehensweise trägt dazu bei ein Infrastruktur-Dokumentationsproblem mit einer räumlichen Verbindung von zwei geographischen Objekten zu lösen. Die Herausforderung dieser Abschlussarbeit ist ein Vorhersageprozess zu entwerfen, welcher unter Anwendung von realen Daten der Telekommunikationsanschlüsse optimale Vorhersageergebnisse liefert. Damit sind sowohl die entsprechenden Datenvorverarbeitung (siehe dazu Kapitel 2.5) als auch die Abstimmung der Modellparameter zusammen mit Optimierungsoptionen (siehe dazu Kapitel 2.2) gemeint.

Im Rahmen der Abschlussarbeit wird geprüft, ob eine Ableitung von physischen Kabelverläufen mittels eines *lazy learning* Vorhersagemodells zufriedenstellende Ergebnisse liefert. Die Bewertung der Ergebnisse erfolgt anhand des im Absatz 2.2.4 Festlegen der Bewertungskriterien für das *lazy learning* Modell definierten Parameters der Modellgüte. Die Zielgröße liegt bei einer statistischen Modellgüte von mindestens 80%. Die fachliche Bewertung der Vorhersage der abgeleiteten Kabelverläufe wird im Rahmen dieser Abschlussarbeit nicht erfolgen, weil diese sehr stark mit der Datenqualität der gelieferten Anschlusspunkte zusammenhängt und nur mittels einer manuellen Überprüfung in der realen Welt validiert werden kann. Mit der Annahme, dass ein physischer Kabelverlauf eines Versorgungsnetzes sich mit der Strecke der Straßenkante deckt, ist die Nachbildung dessen Verlaufes mit einer hohen Wahrscheinlichkeit zu ermitteln (siehe dazu 2.5.2 Nachbilden der OpenStreetMaps Straßenzüge). Mit der Anwendung von einem *OpenStreetMap* Straßenlayer in Form von eindeutigen streckentreuen Polylinien für die Vorhersage ist diese Annahme erfüllt. Die denkbaren Schritte einer Überprüfung der Modellergebnisse in der realen Welt werden in Kapitel 4 Diskussion ausgeführt.

Die Ergebnisse dieser Arbeit sollen einen Beitrag für die Geoinformations-Community leisten und den Spezialisten ein neues Werkzeug für die räumlichen Analysen der Telekommunikationsbrache zur Verfügung zu stellen. Der gewählte Ansatz zur Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien soll auch zur Entwicklung weiterer Techniken der räumlichen Verbindung anregen. Die in dieser Abschlussarbeit beschriebene Vorgehensweise schlägt vor, ein Infrastruktur-Dokumentationsproblem in der Telekommunikationsbranche mit einer räumlichen Verbindung von zwei geographischen Objekten zu lösen. Wie von Reinhardt and Bockmuehl (2013) beschrieben, ist ein prozessorientiertes Qualitätsmanagement bei der Erfassung und auch der Aktualisierung von Geodaten in Netzinformationssystemen notwendig. In dem Fachbeitrag wird ein generischer Ansatz für den Aktualisierungsprozess von Netzinformationen vorgestellt. Dieser konzentriert sich auf die Vollständigkeit der

dokumentierten Prozessschritte bei Aktualisierung vorhandener Geodaten. Im Gegensatz dazu beruht der in dieser Abschlussarbeit beschriebene Vorschlag auf einer möglichen Korrektur, der Bestandsdaten des Unternehmens.

Der Mehrwert dieser Abschlussarbeit ist der Entwurf von einem Workflow zur Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten mit technischer Information und streckentreuen Polylinien. In diesem werden zwei Konzepte, nämlich die Nähe und die Distanz miteinander kombiniert und aufeinander abgestimmt. Basierend auf dem Komplexität Konzept in der Geographie wird die Annahme getroffen, dass auch einfache lokale Beziehungen zwischen Entitäten ausreichen, um eine Kopplung zwischen räumlichen Objekten zu beschreiben und vorherzusagen. (Miller, 2004, p. 284) Demzufolge wurden die zwei Objekttypen hausgenaue Punkte und streckentreue Polylinien, unter der Annahme, dass die Nähe eine Relation ist, in Beziehung gesetzt. Mit Hilfe von einem Distanzmaß wird eine Metadateninformation des Punktes auf die nächste Polylinie übertragen. Hierzu wird ein Vorhersagemodell des *lazy learning* k-*Nearest Neighbor Classifier* angewendet, um die Kopplung von zwei Entitäten zu bewerten und das Modell validieren zu können.

Die im Rahmen dieser Abschlussarbeit entworfene Vorgehensweise zur automatisierten Rekonstruktion und somit auch Digitalisierung und Vektorisierung einer Infrastrukturdokumentation eines Versorgungsnetzes hat eine hohe Relevanz für die Betreiber selbst. Im Bereich der Geoinformatik lässt die Methode eine anschließende räumliche Analyse mit den erzeugten Daten zu. Bisher sind die Netzinfrastrukturnetze in der Bundesrepublik Deutschland nicht flächendeckend als *Open Data* zur Verfügung gestellt. Für die Betreiber, welche über keine vollständige vektorisierte Netzdokumentation verfügen, kann die beschriebene Methode eine mögliche Lösung sein, bestehende Defizite zu beheben. Die im praktischen Teil der Abschlussarbeit abgeleitete Kabelnetzverläufe des Kupferkabels eines Telekommunikationsunternehmens zeigen, dass eine fachliche Korrektur der Ergebnisse notwendig ist, denn das Vorhersagemodell spiegelt die interne

Datenqualität wider. Das mit den Eingabe-Daten trainierte Modell zeigt besondere Konstellationen der Kabelverläufe auf und schlägt dabei eine genormte Zuordnung der technischen ID vor. Diese Vorgehensweise ist somit auch ein Vorschlag für Kabelnetzbetreiber zur Optimierung ihrer internen Datenqualitätssicherungsprozesse. Der Mehrwert ist eine gezielte Fachprüfung von erkannten Auffälligkeiten im Gegensatz zu stichprobenartigen Kontrollen.

1.4 Forschungsfrage

Im Rahmen dieser Abschlussarbeit wird untersucht, ob eine rechnergestützte, räumliche Verbindung von Punktdaten zu Polylinien, die mithilfe eines *lazy learning* Vorhersagemodells durchgeführt wird, zur Nachbildung eines physikalischen Kabelnetzes, basierend auf Metainformationen eines hausgenauen Punktes und einem gegebenen streckentreuem Polylinienetz geeignet ist.

Im Zuge der empirischen Forschungsarbeit werden folgende Teilfragen beantwortet:

- In welchem Maß wirkt sich die Sortierung der „gekennzeichneten Datensätze“ als Optimierungsmethode auf die Ergebnisse des Vorhersagemodells aus?
- Gibt es eine Korrelation zwischen der Größe des k-Distanz Wertes bei der Vorauswahl der relevanten streckentreuen Polylinien und der Güte des Modells?
- Welchen Einfluss auf die Güte des Modells hat der *k-Nearest Neighbor* Algorithmus Wert?
- Dominiert ein Distanzmaß, die euklidische Distanz oder die Manhattan Distanz?

1.5 Abgrenzung der Abschlussarbeit

Im Rahmen dieser Abschlussarbeit werden bekannte Geoinformatik Methoden in einem neuen Kontext angewendet. Der Fokus liegt auf dem Transfer des *k-Nearest Neighbor* Klassifizierungsmodells der scikit-learn Python Bibliothek als Geoinformationstechnik in einen neuen Anwendungsbereich der Telekommunikation. Diese Arbeit soll die Lösung des Problems der Vervollständigung einer Versorgungs-Netzdokumentation unter dem Aspekt von nachträglichem Vektorisieren physischer Kabelverläufe unterstützen.

Die Datengrundlage der praktischen Anwendung stammt aus der Telekommunikationsbranche und stellt die hausgenauen Anschlusspunkte des Koaxial-Kabel-Netzwerkes für die TV und Telefonanschlüsse dar. Für diese wissenschaftliche Arbeit wurden die Daten von dem Telekommunikationsanbieter Unitymedia NRW GmbH zur Verfügung gestellt.

Aufgrund der hohen Auslastung der Rechenkapazität des eingesetzten Computers während der Ausführung von *machine learning* Algorithmen aus der scikit-learn Python Bibliothek wurde die Datenmenge der Test- und Trainingsdatensätze für den praktischen Teil reduziert. Die Begrenzung der Datensätze erfolgte anhand von lagebezogener Auswahl in einem selbst festgelegten Untersuchungsgebiet, was einem Stadtteil der Stadt Köln gleich ist. Das Gebiet wurde anhand von folgenden Parametern festgelegt:

- Kompaktheit
- klare Abgrenzung durch geographische Grenzen
- Anzahl von hausgenauen Punktdaten innerhalb des Gebietes

Der Fokus liegt auf den theoretischen Ansatz der Entwicklung von einer Arbeitsmethode zur Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien. Nicht Bestandteil dieser Abschlussarbeit sind

Programmierarbeiten zur Anpassung des Algorithmus und dem Umgang mit Ausreißern der Klassifizierung für den gesamten Daten-Bestand der Unitymedia NRW GmbH.
Überlegungen und Umsetzungen möglicher Datenkomprimierungsmethoden zur schnelleren Verarbeitung des Gesamtdatenbestandes oder weiteren Performanz Optimierungen des Algorithmus werden hier nicht berücksichtigt.

Die Diskussion anderen Lösungswegen unter dem Einsatz von räumlichen Analysen ist im Kapitel 1.2 Stand der Forschung präsentiert und abschließend im Kapitel 5 Schlussfolgerungen und Ausblick abgeschlossen.

2 Methodik

Die grundlegenden Methoden der quantitativen Arbeitsweise in der Geographie bauen auf Fragestellungen mit einem Raumbezug auf, welche sich mit angewandten numerischen Techniken, Verfahren zur Datenerhebung, Analyse, Prognose und Modellbildung beantworten lassen. Die Methodik umfasst nicht nur die Anwendung von mathematisch-statistischen Methoden, sondern auch Entwicklung von Verfahren zur Verarbeitung quantitativen Informationen oder auch Simulationsmodellen. (de Lange and Nipper, 2018, p. 18)

Die in dieser Abschlussarbeit aufgestellte Forschungsfrage ist durch eine praktische Anwendung eines *machine learning* Modells und insbesondere der *lazy learning* Methode basierend auf der räumlichen Verbindung anhand der kürzesten Distanz und der Nachbarschaftsbeziehung zweier Datensätze getestet worden. Die einzelnen Schritte der praktischen Umsetzung sind im Kapitel 2.3 Prozessübersicht beschrieben. Die Vorgehensweise ist methodisch bei der Anwendung von der *nearest neighbour* Klassifikationsmethode zuzuordnen. Nach der a-posteriori-Definition von Klassen ist die Klassenzugehörigkeit einer Entität in der Datenstruktur vorgegeben. (de Lange and Nipper, 2018, p. 340) Die technische Identifikationsnummer der hausgenauen Punktdaten, welche in der praktischen Anwendung in dem Rahmen der Abschlussarbeit genutzt wurden, ist ein Ähnlichkeitsmaß der Objekte untereinander. Wie de Lange and Nipper (2018) a-posteriori-Definition von Klassen beschreiben, werden die Elemente einer Clusteranalyse so zerlegt, dass die Objektmenge in mehreren Teilgruppen innerhalb einer Gruppe einander möglichst ähnlich und die Gruppen untereinander möglichst unähnlich sind. Die Suche nach mehreren Gruppen ähnlicher Objekte in einem mehrdimensionalen Raum bedient sich an den Methoden der numerischen Klassifikation. Die Distanzgruppierung von Objekten nimmt an, dass die Ähnlichkeit solcher Objekte zueinander mit Nähe also mit dem Abstand voneinander gleichzusetzen ist. Der meist verwendete Operator der Ähnlichkeit ist der euklidische Abstand der Objekte im Raum mit dem Ansatz: „Je größer der Abstand, desto größer die Unähnlichkeit.“ (de Lange and Nipper, 2018, p. 354) Nach Bahrenberg, Giese et

al. (2008b) wird in der Clusteranalyse das Ähnlichkeits- bzw. Unähnlichkeitsmaß für eine vorgegebene Anzahl von Gruppen als ein nicht-hierarchisches Verfahren gesehen. Die Gruppierungsstrategie für die hausgenauen Punkte bedarf keiner Clusteranalyse, da die vorliegenden technischen Informationen des Kabelnetzwerkes die Untergruppen vorgeben und somit eine Clusterung darstellen. Diese technische Information wird mittels der räumlichen Verbindung, anhand der kürzesten Distanz zwischen hausgenauen Punkten und streckentreuen Polylinien, auf extrahierte Punkte der Polylinien übertragen.

Die für das Vorhaben verwendeten Werkzeuge sind die Geoverarbeitungssoftware: ArcGIS Pro der Firma Esri zu Gestaltung der Übersichtkarten und die *Open Source* Variante QGIS Desktop 2.18.3 für weitere Datenaufbereitungsmethoden, sowie eine Python Konsole Spyder the Scientific Python Development Environment zu dem Erstellen und Ausführen des Vorhersagemodells. Die benutzte Skriptsprache ist Python 3.7.1 mit folgenden *Open Source* Bibliotheken:

- pandas

  Diese Python-Bibliothek ist zur Durchführung von umfangreichen Daten Analysen und rechenaufwendigen Datenverarbeitungen geeignet.

  https://pandas.pydata.org
- GeoPandas

  Ist eine Erweiterung der pandas Bibliothek und erweitert diese um den Umgang mit räumlichen Daten, räumliche Analysen und räumliche Objekte. Mit dem Modul shapely sind geometrische Operationen mit räumlichen Objekttypen durchführbar. Zusätzlich ist das Modul fiona für die geographische Projektion und matplotlib für das Erzeugen von Diagrammen und Karten von Bedeutung.

  http://geopandas.org

- scikit-learn

  Ist eine führende frei verfügbare Python Bibliothek zu dem Ausführen von *machine learning* Modellen. Sie baut auf den Modulen NumPy, SciPy und matplotlib auf. Die verfügbaren Module und Tools sind performant in Verarbeitung von großen Datenmengen. Das Modul *sklearn.neighbors* hat den *k-Nearest Neighbors Algorithm* implementiert und lässt die Methoden des *unsupervised* und *supervised neighbors-based learning methods* nutzen.

  (https://scikit-learn.org/stable/modules/neighbors.html#neighbors)

- NumPy

  Ist ein Paket des Scipy.org Projektes für das wissenschaftlichen Rechnen mit Python. Die wichtigste Funktion dieses Modules ist die Arbeit mit N-dimensionalen array Objekten und hoch-performante Anwendung von mathematischen Funktionen.

  https://numpy.org

- logging

  Ist eine populäre Bibliothek zu der Protokollierung von Anwendungen und dem Erstellen von zusammengefassten Ergebnis-Berichten eigener Skripte. Der größte Vorteil ist die Kompatibilität mit allen Python Modulen, sowie die Ausgabe benutzerdefinierten Nachrichten.

  https://docs.python.org/3/library/logging.html

In den folgenden Kapiteln werden die Methoden der räumlichen Verbindung, aber auch das *lazy learning* Modell mit der Prozessübersicht und den einzelnen Datenverarbeitungsschritten beschrieben. Zuerst wird die Methode des Linear Ordering erklärt und in die *Nearest Neighbour spatial join* Methode eingeführt. Folgend wird der k-*Nearest Neighbor* Algorithmus beschrieben und die für die praktische Anwendung verwendete Parameter diskutiert. Anschließend wird das Workflow Modell skizziert und die Datengrundlage vorgestellt. Abschließend werden die wichtigsten Schritte der Testdatenvorverarbeitung und die dafür verwendete Methoden beschrieben.

2.1 Spatial join, die räumliche Verbindung

Wie in der Einleitung des Kapitels erklärt können die Nachbarschaftsbeziehung und auch die Ähnlichkeit bzw. Unähnlichkeit von Objekten in einem Merkmalsraum mit kartesischem Koordinatensystem als Grundlage festgelegt werden.
Die unterschiedlichen geographischen Objekttypen in einem Raum stehen in einer Beziehung zueinander. Die topologische Struktur der Daten beeinflusst die Fragestellungen der räumlichen Datenanalyse in der Art eines Bezugs oder im Aufbau möglicher Kombinationen eines Verbundpaares. Diese lassen sich in drei Kategorien unterteilen, die Erste geht von einer einfachen Grundeinheit aus, welche eine Grundlage für den Aufbau von komplexen Objekten ist. Die zweite Beziehung lässt sich aus den Koordinaten der einzelnen Objekte berechnen, hierzu gehört auch die Prüfung der Flächen auf Überschneidung oder auch Überlappung. Die letzte Kategorie bilden Kriterien, welche in der Attributtabelle eines Objektes als eine zusätzliche Information gespeichert sind und meistens nicht berechenbar oder ableitbar sind. (Kappas, 2012, pp. 66-67)

Die Beziehungen zwischen räumlichen Daten sind bedeutend für Datenanalyse. Die Beziehungen zwischen räumlichen Daten sog. *spatial data relationships* können mit der Methode räumliche Verbindung (*spatial join*) von zwei Datensätzen in einem Raum untersucht werden. (Kappas, 2012, p. 66) Jacox and Samet (2007) beschreiben diese Operation in einer vereinfachten Form. Die Definition lautet: gegeben sind zwei Mengen von Rechtecken, R und S, in diesen Mengen werden alle Paare von sich überschneidenden Rechtecken gesucht. Für jedes Rechteck r in der Menge R, werden die sich überschneidende Rechtecke s aus der Menge S ausgewählt. Diese standardisierte Vorgehensweise wird als *spatial overlay join* bezeichnet und kann auf unterschiedliche Weise ergänzt werden. Eine mögliche Erweiterung ist statt den beschriebenen Rechtecken topologische Objekte: Punkt, Linie oder Polygon in unterschiedliche Konstellationen miteinander zu verbinden. Diese Daten können auch mehr als zwei Dimensionen aufweisen. Die räumliche Beziehung der

Objekt-Paare untereinander können jede denkbare Relation von Überschneidung, Nähe, Einzäunung bis zur Richtung sein. Die räumliche Verbindung kann auch mit mehreren Datenmengen durchgeführt werden, was als ein *multiway spatial join* bezeichnet wird. Demgegenüber ist eine Suche nach Beziehungen innerhalb einer Datenmenge als sog. *self spatial join* möglich. (Jacox and Samet, 2007, pp. 2-3)
An dieser Stelle ist es wichtig zu sagen, dass eine relationale Verbindung (*relational join)* sich von der räumlichen Verbindung (*spatial join*) unterscheidet. Der Standard einer relationalen Verbindung von Daten lässt als Eingabedaten keine multi-dimensionale räumliche Attribute auf den zu verbinden Objekt zu. Diese Bedingung schließt das Anwenden von vielen weiter entwickelten relationalen Algorithmen zur Verbindung von Daten aus. (Jacox and Samet, 2007, p. 3)

Die unterschiedlichen Arten von möglichen relationalen Verbindungen *relational join operations* können bei Ullman (1988, pp. 59-63) oder auch bei Gunther (1993, pp. 3-5) nachgelesen werden. Ferner sind die verschiedenen Techniken der räumlichen Verbindung (*spatial join*) kompakt von Jacox and Samet (2007) in dem Artikel „Spatial join techniques“ vorgestellt. Dieser verweist auch auf eine Vielzahl von unterschiedlichen Abwandlungen der Algorithmen zur Unterstützung und Optimierung der computergestützten Rechenaufgabe bei einer räumlichen Verbindung *spatial join*.

Die räumliche Verbindung spielt eine entscheidende Rolle bei der Vorgehensweise zum Erreichen des Leitzieles (Kapitel 1.4 Forschungsfrage) dieser Abschlussarbeit. Die praktische Anwendung dieser Methode ist im Punkt 2.3 Prozessübersicht näher beschrieben.
Folgend wird *linear ordering* als eine Methode zur Verbesserung der Ergebnisse von einer räumlichen Verbindung (*spatial join*) erklärt. Anschließend wird der in dieser Abschlussarbeit verwendete *k-Nearest Neighbor join* von topologischen Objekten beschrieben.

2.1.1 Linear Ordering

Der Begriff *linear ordering* beschreibt die Abfolge von benachbarten Objekten, welche linear verlaufen und miteinander verbunden sind. Die folgenden Ausführungen basieren auf Jacox and Samet (2007) Spatial Join Techniques.

Die *linear order* erzeugt eine Gesamtreihenfolge über multi-dimensionale Objekte. Dies ermöglicht das Durchlaufen dieser Objekte in genau einer Reihenfolge.

Die Definition basiert darauf, dass alle Objekte genau einen Vorgänger und einen Nachfolger haben, außer den ersten und letzten Objekt, welches keinen Vorgänger bzw. keinen Nachfolger hat.

Die Richtung der Verbindungen untereinander ergibt sich aus der Sortierungsmethode der Daten, wie z.B. aufsteigende oder absteigende Reihenfolge.

Bei einer aufsteigenden Sortierung der Daten gilt, dass ein Objekt kleiner ist als sein Nachfolger und im Umkehrschluss ist das zweite Objekt größer als sein Vorgänger. Die Reihenfolge ist transitiv.

Im Gegensatz zu einer 1-dimensionalen linearen Sortierung, bei der die Nähe anhand einer Dimension ausgedrückt werden kann, existiert diese natürliche Sortierung bei multi-dimensionalen Objekten normalerweise nicht. Da die Nähe in einer Dimension nicht unbedingt die Nähe in einer anderen Dimension ausdrückt, ist es notwendig über die *linear order* eine eindeutige Nähe zu definieren, so dass gilt, dass die Nähe über die Anzahl der Nachfolger oder Vorgänger zwischen den Objekten ausgedrückt werden kann.

Abbildung 3 Unterschiedliche Reihenfolgen einer linearen Sortierung stellt zwei unterschiedliche Anordnungsarten von den Objekten dar.

Die in dieser Abbildung dargestellte Verbindungs-Pfeile zeigen die Rangfolge der Anordnung der Daten im Sinne von *linear ordering*. Die Verbindungen bilden eine durchgehende, eindeutige Reihenfolge. Je nach Art der Sortierung können sich die Nachbarschaftsbeziehungen ändern. So ist in der Abbildung a) das Objekt **b** mit **a** und **c**

benachbart. In der Abbildung b) gilt die Nachbarschaft-Beziehung nicht mehr, da das Objekt **b** an der letzten Stelle eingeordnet ist und mit dem Objekt **e** benachbart. Dieses Beispiel soll die Auswirkung einer Sortierung der Daten in Bezug auf die Nähe zueinander darstellen.

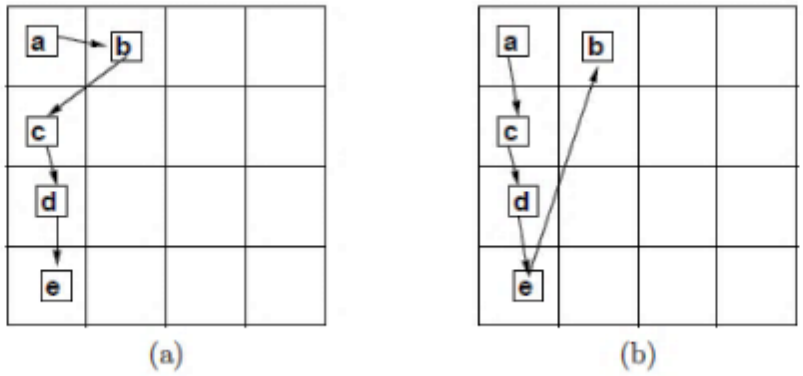


Abbildung 3: Unterschiedliche Reihenfolgen einer linearen Sortierung, (Jacox and Samet, 2007, p. 8)

Die lineare Sortierung spielt für die räumliche Verbindung (*spatial join)* bei der Verbindungsbeziehung kürzeste Distanz (*close to*) eine entscheidende Rolle, wenn die Distanzen von den zu verbindenden Objekten gleich und nicht eindeutig sind. In einem solchen Fall gibt der *spatial join* mehr als ein Treffer mit dem gleichen Distanz Wert zurück, so dass keine Entscheidung der kürzesten Distanz möglich ist. Bei der klassischen Vorgehensweise entscheidet dann die Reihenfolge (*linear order*) über die Auswahl der zu verbindenden Objekten. Daher ist es notwendig eine *linear order* zu definieren und sie beeinflusst das Ergebnis von *spatial join* Operationen.

Nach Jacox and Samet (2007, p. 8) hat die *linear ordering* Methode die größte Auswirkung bei 1- Dimensionalen Daten. Im Falle einer Multi-Dimensionalität steht eine Vielzahl von weiter entwickelten Sortierungsalgorithmen zur Verfügung. Ein Beispiel ist die von Xia, Lu et al. (2004) genutzte *G-ordering k-Nearest Neighbor join method*, welche auf einer Sortierung anhand eines Koordinatensystemgitters basiert. Hier wird auch sichtbar, dass die Methoden Sortieren (*ordering*) und Verbinden (*join*) kombiniert werden, um das Ergebnis

zu optimieren. Bei dem von Xia, Lu et al. (2004) beschriebenen Anwendungsfall handelt sich um die Verknüpfung von diesen Methoden mit anschließenden k-*Nearest Neighbor join.*

Der Algorithmus wird in zwei Phasen abgearbeitet: im ersten Schritt findet die Sortierung der Datensätze statt und danach folgt die räumliche Verbindung der Daten (siehe dazu Abbildung 4: Definition des Gorder_KNN).

Zur Optimierung der räumlichen Verbindung werden die Daten nicht nur in einer Reihenfolge sortiert, sondern auch gruppiert. Hier werden die benachbarten Punkte innerhalb eines Gitters in einer Gruppe zusammengefasst, um später in einer Schleife nach den aufgeteilten Blöcken verbunden zu werden. (Xia, Lu et al., 2004, p. 758)

Algorithm 1 Gorder_KNN(R, S)

Input:

R and S are two data sets.

Description:

1: G_Ordering R and S;
2: Join_Grid_Ordered_Data(R, S);

Abbildung 4: Definition des Gorder_KNN Algorithmus, (Xia, Lu et al., 2004, p. 758)

Wie in Abbildung 5: Schritte des G-ordering dargestellt werden die originalen Daten (Abbildung 5 a) in dem ersten Schritt nach der *principal component analysis* (*PCA,* nachzulesen bei (Jolliffe, 2011)) in den *Principal Component Space* (Abbildung 5 b) transformiert.

In der zweiten Stufe werden die Daten nach seiner Ausdehnung jeweiligen rechteckigen Zelle des Gitters des Koordinatensystems für jede Dimension separat zugeordnet (Abbildung 5 c) und für die räumliche Verbindung mit dem k-*Nearest Neighbor* Algorithmus in einer Schleife für jedes Block abgearbeitet. (Xia, Lu et al., 2004, pp. 758-759)

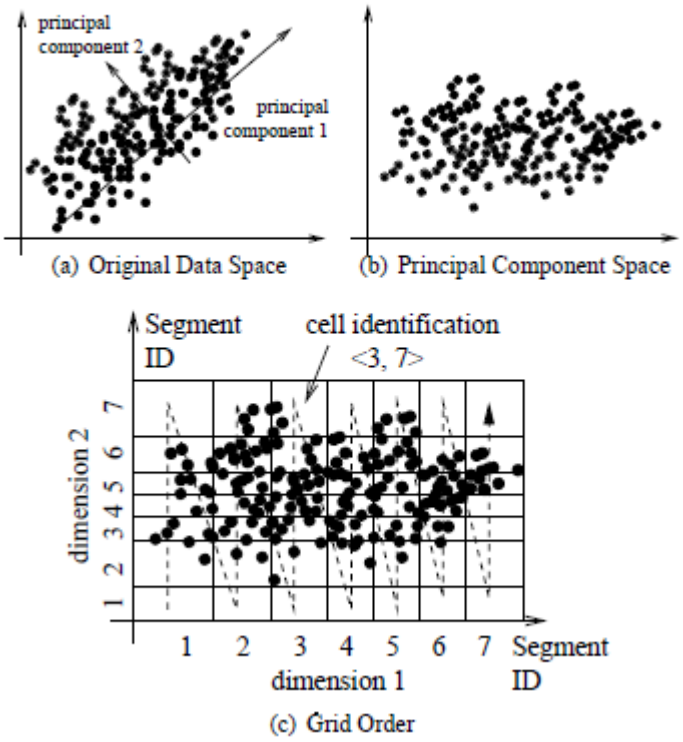


Abbildung 5: Schritte des G-ordering Konzeptes, (Xia, Lu et al., 2004, p. 758)

Daneben beschreiben Xia, Lu et al. (2004) noch weitere Algorithmen zur Optimierung von k-*Nearest Neighbor join* Operationen die hier nicht weiter beschrieben werden.

Bei der in Kapitel 1.4 Forschungsfrage gestellten Teilfrage wird dem Ansatz von Xia, Lu et al. (2004) in einfacher Weise gefolgt und eine Optimierung mittels Sortierung vor dem eigentlichen k-*Nearest Neighbor join* berücksichtigt.

In welchem Maß sich die Sortierung der „gekennzeichneten Datensätze“ auf die Ergebnisse des Vorhersagemodells auswirken wird in Kapitel 3.1.3.3 Einfluss der Vorsortierung auf das Modell beschrieben.

2.1.2 k-Nearest Neighbor Join

Der k-*Nearest Neighbor join* basiert auf der räumlichen Verbindung (*spatial join*). Die Beziehungen zwischen räumlichen Daten in einem Koordinatensystem können anhand eines Distanzmaßes berechnet werden und in ein Verhältnis gesetzt werden. Somit kann die Nachbarschaftsbeziehung von Objekten auf unterschiedliche Weise festgestellt werden, eine davon ist die Nähe, welche als eine metrische Distanz ausgedrückt werden kann. Andere Möglichkeiten sind die Beschreibung von topologischen Relationen durch Überschneiden (*intersect*), Treffen von Flächen (*meet*) oder Richtungsbeziehungen (*south-west*). (Koperski, Han et al., 1998, p. 46)

Die Feststellung der Ähnlichkeit in diesem Kontext die Nähe und das Verbinden von Datensätzen aus zwei unterschiedlichen Datenmengen wird als *similarity query* bezeichnet. Eine Menge von einzelnen *similarity queries* zur Bestimmung von ähnlichen oder benachbarten Objekten können in einem *similarity join* zusammengefasst bzw. durch einen einzelnen *similarity join* ersetzt werden.
Im Laufe der Zeit wurde die Operation *similarity join* weiterentwickelt. Die folgenden Ausführungen und Erklärungen sind aus (Böhm and Krebs, 2004, p. 728-729) entnommen.
Zu unterscheiden sind drei Arten, erstens der *distance range join*, der daraus entstandene *k-distance join* und schließlich die zurzeit meist genutzte Verbindungsart der k-*Nearest Neighbor join*. Anhand der Abbildung 6: Difference between similarity join operations lässt sich der Unterschied zwischen den einzelnen Verbindungsarten erklären.

Die vorgestellten Daten sind als vektorisierter Punkt-Layer aus zwei unterschiedlichen Mengen R und S definiert. Die angewandte Methode zur Bestimmung von Verbindungspaaren ist analog zu der räumlichen Verbindung (*spatial join)* (siehe dazu Absatz 2.1 Spatial join, die räumliche Verbindung).

Als Ergebnis wird eine Menge von S Datensätzen zu einem Punkt aus der Datenmenge R zugeordnet. Der Unterschied liegt in der Definition der räumlichen Beziehung von den Daten untereinander. Für den in der Abbildung 6 a) dargestellten *range distance join* lautet die Bedingung: verbinde die Punkte aus der Menge S zu einem Punkt aus der Menge R, welche in einem Abstand **a** nicht größer als folgend definiertes Parameter **ε:** sind. Daraus ergibt sich folgende Formel:

$$R \underset{\varepsilon}{\bowtie} S := \{(r_i, s_j) \in R \times S \mid \|p_i - q_j\| \leq \varepsilon\}$$

Diese Definition hat zufolge, dass alle Objekte aus S unabhängig von der Anzahl der Treffer in dem definiertem Umkreis **ε** dem Suchobjekt R zugeordnet bzw. mit ihm verbunden werden. Diese Operation ist in Standard GIS-Software als Buffer-Analyse bekannt. Nachteil dieses *similiarity joins* ist die unbekannte Anzahl an Rückgabewerten (ggf. auch die gesamte Datenbank), die normalerweise begrenzt werden soll. Daher wurde der k-*distance join* entwickelt, der die Ergebnismenge auf k Objekte begrenzt.

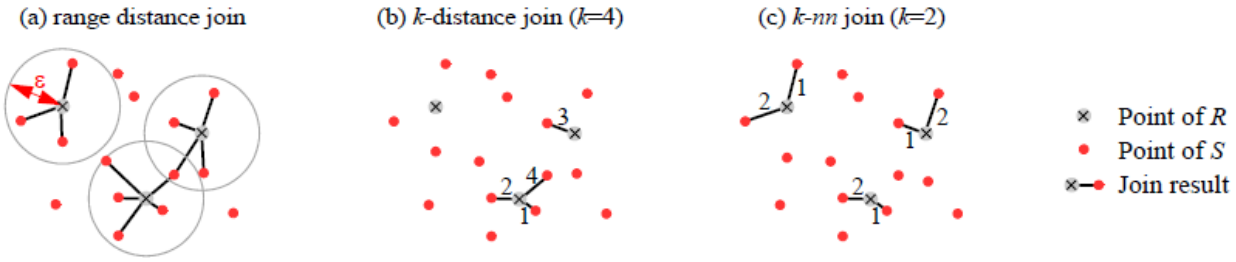


Abbildung 6: Difference between similarity join operations aus (Böhm and Krebs, 2004, p. 728)

Die Abbildung 6 b) zeigt die Verbindungsergebnisse der Operation k-*distance join* mit dem Parameter **k=4**. Die Suche geht von allen Punkten aus der Datenmenge **R** aus und gibt **k** Anzahl von den besten Paaren mit den besten Ergebnissen aus. Die Bedingung der Verbindung der Daten war die kürzeste Distanz, gemessen von Punkten aus der Datenmenge **S** zu den Punkten aus der Datenmenge **R**. In dieser Operation werden alle möglichen Kombinationen abgespeichert und aus der Grundgesamtheit der Berechnungen

nur die **k** besten Treffer mit der niedrigsten Distanz als Ergebnis rausgegeben. Diese Vorgehensweise ist die Grundlage für den nachfolgenden k-*Nearest Neighbor join*, wo die Auswahl der Treffer als Ergebnis angepasst wurde. Der Nachteil des k-*distance joins* ist das ggf. nicht alle Objekte aus **R** zu den besten **k** Verbindungen gehören, somit ist nicht gewährleistet, dass zu jedem Objekt aus **R** eine Verbindung zu **S** als Ergebnis geliefert wird. Daher wurde der Ansatz zum k-*Nearest Neighbor join* weiterentwickelt.

Die Abbildung 6 c) visualisiert die umgekehrte Situation der Auswahl der Endergebnisse von k-*Nearest Neighbor similarity join*. In diesem Fall werden nicht die **k** besten Paaren aus der Liste mit allen möglichen Kombinationen mit Distanzen der Entfernung von jeden Punkt aus der Datenmenge **R** zur den Punkten aus der Datenmenge **S** ausgewählt, sondern liefert die **k** besten Treffer mit der kürzesten Distanz aus der Datenmenge **S** für jeden Datensatz aus der Datenmenge **R**. (Böhm and Krebs, 2004, p. 728-729)

In dieser Abschussarbeit ist auch der KNN-Join die Grundlage für das *machine learning* Modell. Wie in Absatz 1.2 Stand der Forschung erwähnt gibt es eine Vielzahl an Forschungsarbeiten, welche sich mit dem Thema der Optimierung von Speicherauslastung und Nutzung von Rechnerkapazitäten für k-*Nearest Neighbor join* Operationen beschäftigen.

Der *k-Nearest Neighbor join* wird auch häufig zu der Laufzeit-Optimierung von bestehenden Klassifizierungsalgorithmen eingesetzt. Die neu entwickelten Vorgehensweisen unterscheiden sich in der technischen Umsetzung der bekannten Methode und auch in der Variation der Reihenfolge der zu ausführenden Schritten während der Suche nach dem nächsten Nachbar. Ein Beispiel für so eine Umsetzung ist der *multi-step algorithm* für den k-*Nearest Neighbor Search,* eingeführt von Seidl and Kriegel (1998). Es zeigt sich, dass Optimierung bei der Suche nach dem Nächsten Nachbarn und k-*Nearest Neighbor join* Operationen ein eigenes wissenschaftliches Themengebiet sind, auf welches aber, aufgrund

seiner Verknüpfung zur eingesetzten Hardware und dem damit einhergehenden technischen Fokus, nicht weiter eingegangen wird.

2.2 Das *lazy learning* Modell zur Ableitung von Kabelverläufen

Das *machine learning* auch als machinelles Lernen bekannt, kann hauptsächlich in drei Arten unterteilt werden. Hierzu gehört *supervised learning* (überwachtes Lernen), *unsupervised learning* (unüberwachtes Lernen) und *semisupervised learning* (halb-überwachtes Lernen). Das Forschungsfeld von *machine learning* beschäftigt sich mit der Anwendung von Algorithmen, welche statistische Zusammenhänge von Daten erkennen und aufgrund dessen ein Muster erlernen können. Anhand des erlernten Musters der Daten ist es möglich eine Vorhersage über zukünftige Ereignisse, Eigenschaften oder eine andere Zielgröße zu treffen.
Die Zielgröße kann sowohl die Vorhersage einer kategorialen Klasse (z.B. der TECH_ID) als auch einer stetigen Variablen (z.B. ein Score oder ein Euro-Betrag) sein.

Das Extrahieren des Wissens aus Daten durch einen selbstlernenden Algorithmus wird dem *machine learning* als Teildisziplin der *Artificial Intelligence* sog. künstlichen Intelligenz zugeordnet. In einem solchem Verfahren erübrigt sich die bisherige Vorgehensweise durch menschliches Eingreifen starre Regeln zu definieren nach denen ein Modell eine Vorhersage ausführen soll. (Raschka, 2017, pp. 23-24)

Beim überwachten Lernen wird ein *machine learning* Modell auf Daten trainiert, bei denen das Ergebnis bekannt ist (z.B. aufgrund von Historiendaten). Das Ziel ist es, ein Modell zu entwickeln, welches die Muster in den Trainingsdaten erkennt und somit ein generisches Modell für die Vorhersage bei unbekannten Daten festlegt. Die mögliche Ergebnismenge und das Ziel des Modells sind bekannt.

Beim unüberwachten Lernen wird ein Modell auf Daten trainiert, bei denen das Ergebnis unbekannt ist, d.h. das Modell soll selbstständig Muster erkennen und auch selbstständig diese in Ergebnismengen anordnen. Durch das eingesetzte Verfahren wird die unbekannte Struktur analysiert, um sinnvolle Informationen aus den Daten zu erhalten. Das gesamte Vorgehen hat keinen Richtwert in Form von einer vorgegebenen Zielvariable oder einer Bewertungsfunktion des Modells. Zu so einem Datenanalyseverfahren gehört das *clustering,* welches die Daten in sinnvolle Untergruppen aufteilt ohne eines vorgegebenen Gruppenzugehörigkeitskriteriums. (Raschka, 2017, p. 28). Die Vorteile, als auch die Herausforderung bei einem *clustering* sind, die nur aufgrund von statistischen Auffälligkeiten ermittelten Gruppen zu verstehen und für den jeweiligen Anwendungsfall relevante von nicht relevanten Gruppen zu unterscheiden.

In Abgrenzung zu anderen *machine learning* Algorithmen unterscheiden sich sog. *lazy learning* Algorithmen in drei wesentlichen Aspekten von anderen *machine learning* Algorithmen (Aha, 2013).

- Sie verschieben die Verarbeitung der Trainingsdaten auf den Zeitpunkt des Aufrufs des Modells zur Ergebnisgenerierung. Die Trainingsdaten werden bis dahin nur gespeichert, aber nicht genutzt.
- Sie kombinieren die gespeicherten Trainingsdaten um ein Ergebnis zu generieren.
- Sie verwerfen die generierten Ergebnisse und alle Zwischenergebnisse

Die *lazy learning* Methode wird als „träges Lernen“ bezeichnet. Der Grund dafür ist, dass in diesem Modell „das Lernen“ aus den Trainingsdaten anders als bei anderen *machine learning* Algorithmen nicht vorher, sondern beim Aufruf des Modells erfolgt. Die Trainingsdaten werden vorgemerkt und innerhalb des Modells gespeichert.
Im Gegensatz zu *lazy learning* kommt auch sogenanntes „eifriges Lernen“ (*eager learning*) vor. Die eifrige Form des Lernens zeichnet sich dadurch aus, dass die Trainingsdaten direkt verwendet werden und deren Muster und Strukturen für die Entwicklung und Beschreibung

des Modells (*training*) genutzt werden. Danach werden die Trainingsdaten nicht mehr benötigt und verworfen, da die relevanten Erkenntnisse in das Modell übergangen sind. Die Unterschiede zwischen den beiden Konzepten des Lernens bieten auch neue Einsatzmöglichkeiten der beiden Formen auf, indem diese miteinander kombiniert werden können. (Aha, 2013, pp. 7-10)

Der Ablauf von einem Vorhersagemodell kann sich je nach verwendeten Methoden und Algorithmen unterscheiden. Grundsätzlich lässt sich aber ein Vorgehen beim Einsatz von *machine learning* Algorithmen in verschiedenen Schritten abbilden. Die Abbildung 7: Ablauf in machine learning Vorhersagemodellen stellt eine mögliche Struktur eines Vorhersagemodells dar. Der Ablauf lässt sich in vier Abschnitte unterteilen: Vorverarbeitung, Lernen, Bewertung und Vorhersage. Unabhängig davon, welche Art von *machine learning* Algorithmen verwendet wird, sind in erster Linie *Input*-Daten notwendig. Die Rohdaten werden vorverarbeitet und mittels möglichen Datenverarbeitungsoperationen, wie Merkmalextrahierung, Maßstab Anpassung, Merkmalauswahl oder Dimensionsreduktion der Daten und Stichproben Prüfung an die gewünschte Struktur angepasst. Das Vorhersagemodell wird beim überwachten Lernen anhand einer dem Modell unbekannten Datenmenge bewertet, bei dem aber das Ergebnis bekannt ist. Dafür werden die Daten mit der zu vorhersagenden Bezeichnungen in Trainingsdaten- und Testdatenmenge aufgeteilt. Der Prozess des Lernens wird durch die Auswahl eines Lernalgorithmus bestimmt. Davon hängt auch ab, welches Vorhersagemodell angewendet wird. Hier ist es auch möglich ein eigenes Modell zu entwerfen, wenn die Annahmen über die Aufgabenstellung nicht in bestehende Muster zu erfassen sind. In der Praxis ist es notwendig verschiedene Algorithmen miteinander zu vergleichen um ein bestmögliches Modell zu entwickeln, dafür eignen sich Kreuzvalidierungsverfahren. Die meisten Standard-Vorhersagemodelle sind sehr allgemein gehalten. Um eine spezielle Aufgabenstellung zu beantworten ist die Definition der Hyperparameter-Optimierung ausschlaggebend. Die optimale Auswahl und Definition der Algorithmus Parameter führt

nicht nur zur Performance Steigerung des Modells, sondern auch zur höheren Genauigkeit der Vorhersage. Die definierten Parameter werden ausschließlich anhand der Trainingsdaten ermittelt und schließlich auf die Testdaten und neue Daten angewendet. Die Trainingsdaten sind dem Modell bekannt und somit für die Bewertung der Vorhersage ungeeignet, da damit nicht die Generalisierung des Modells auf andere Daten geprüft werden kann.

Die Testdaten sind für das trainierte Modell unbekannt, somit kann anhand dieser Ergebnisse das Modell bewertet werden, wie gut oder wie treffend die Vorhersage ist. Für die Bewertung des Vorhersagemodells kann je nach Anwendungsfall ein eigener Schwellenwert definiert werden, welcher erreicht werden muss, um ein Modell als gut oder ausreichend für den Anwendungsfall zu akzeptieren.

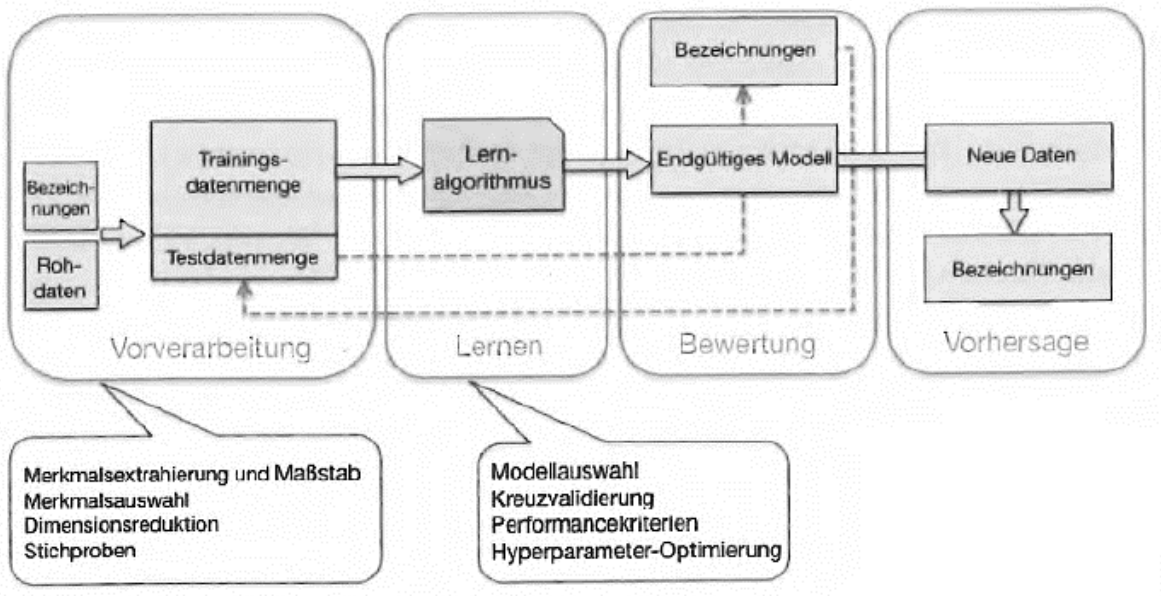


Abbildung 7: Ablauf in machine learning Vorhersagemodellen, (Raschka, 2017, p. 31)

Diese Abfolge wurde im Rahmen dieser Abschlussarbeit an die Anwendung des *lazy learning* k-*Nearest Neighbor Classifier* angepasst und im Absatz 2.2.1 Prozessübersicht ausführlich beschrieben.

2.2.1 Die Struktur des k- nächsten Nachbarn Algorithmus

Der *k-Nearest Neighbor* Klassifizierungs-Algorithmus gehört zu der Klasse der überwachten Lernalgorithmen und wird zur Vorhersage der Klassenbezeichnungen (hier TECH_ID) verwendet. Wie im Absatz 2.2 Das *lazy learning* Modell zur Ableitung von Kabelverläufen eingeführt, gehört dieser Algorithmus zu der *lazy learning* Klasse. Die Basis des Klassifizierers ist die *k-Nearest Neighbor* Methode (siehe dazu 2.1.2 k-Nearest Neighbor Join), welche die Aufgabe hat die **k** nächsten Nachbarn zu ermitteln, um eine Vorhersage zu treffen (siehe dazu auch 2.2.2 Definition der Parameter). Die Auswahl der zu vorhersagenden Objekte findet anhand der kürzesten Entfernung mit zugrunde liegenden Distanzmaß statt. (siehe dazu 2.2.3 Metrische Distanzen)

Kubat (2015) stellt den Aufbau eines simpelsten *k-Nearest Neighbor* Klassifizierer dar. Die grundsätzliche Aufgabenstellung ist es, einen Mechanismus zu finden mit dem die Ähnlichkeit von Attributen unterschiedlicher Vektorobjekten bewertet werden kann und somit eine Klassifizierung ermöglicht.

Als Beispiel wird hier **X** für einen Datensatz gesetzt, zu dem die Vorhersage der Zugehörigkeit zu einer Klasse getroffen werden soll.

Die benötigten Schritte sind vereinfacht die Folgenden:

1. Innerhalb der Trainingsdaten die **k** nächsten Nachbarn zu dem Datensatz **X** identifizieren, welche sich am meisten ähnlich sind (z.B. aufgrund von kürzester Distanz)
2. Ermitteln der unter den ausgewählten **k** nächsten Nachbarn dominierenden Klasse, als Beispiel wird hier die Klasse $\mathbf{c_i}$ genannt
3. Zuordnung der am häufigsten auftretenden Klassen zu dem Datensatz **X**, in diesem Fall bekommt **X** die Beschriftung $\mathbf{c_i}$

(Kubat, 2015, pp. 44-45)

Der zu prognoszierende Wert der Trainingsdatensätzen wird im Sinne des *k-Nearest Neighbor join*'s durch eine Mehrheitsentscheidung seiner **k** nächsten Nachbarn zugewiesen. Die Hauptaufgabe des Klassifikators ist bei dem **k** Abstimmungsnachbarn eine Vorhersage anhand der abgespeicherten Fälle aus den Trainingsdaten zu treffen. Wie in Abbildung 8 sichtbar beeinflusst der **k** Wert mit seiner Höhe die Entscheidung über die Zuordnung des neuen Wertes. Für den gesuchten Datensatz (dargestellt als Kreis mit Fragezeichen) wurden fünf (**k**=5) nächste Nachbarn gefunden. Die bekannten Klassen verteilen sich wie folgt: drei Nachbarn haben die Klasse Dreieck, einer hat die Klasse Minuszeichen und der Letzte hat die Klasse Pluszeichen. Da die Mehrheit der Nachbarn der Klasse Dreieck zugehört, wird diese Klasse dem gesuchten Datensatz zugeordnet.
Je höher der **k** Parameter und je unterschiedlicher die einzelnen Exemplare, desto niedriger ist die Wahrscheinlichkeit der Vorhersage der Klassenzugehörigkeit. (Raschka, 2017, pp. 104-105)

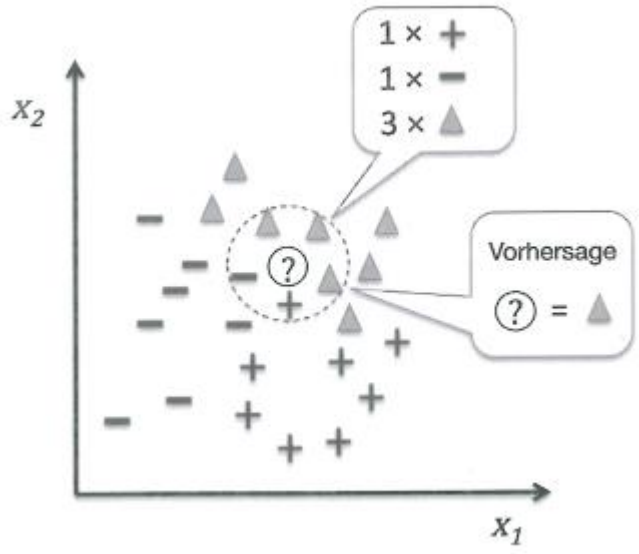


Abbildung 8: Beispiel Klassifizierer (Raschka, 2017, pp. 104-105)

Im Falle einer gleichmäßigen Verteilung der Klassen der Nachbarn besteht die Gefahr, dass die Mehrheitsentscheidung nicht ausgeführt werden kann, da keine Klasse in ihrer Anzahl überwiegt. Die endgültige Entscheidung kann dann je nach Implementierung

des dahinterstehenden Basis Algorithmus unterschiedlich ausfallen. Aus diesem Grund wird in der Praxis der **k** Wert auf ungerade Zahl gesetzt. (Kubat, 2015, p. 45) Aber auch diese Maßnahme verhindert nicht immer, dass es zu beschriebener Gleichverteilung kommen kann. In dieser Abschlussarbeit wurde der k-*Nearest Neighbor Classifier* der scikit-learn Python Bibliothek angewendet, welcher bei einem Gleichstand in erster Linie die Nachbarn mit der kürzesten Distanz bevorzugt. Wenn aber auch die Distanzen gleich sind wählt der Algorithmus die Klasse, welche in den gespeicherten Beispielen aus den Trainingsdaten zuerst vorkommt. (Raschka, 2017, p. 106) Somit ist in dieser Abschlussarbeit die Reihenfolge der Sortierung von Trainingsdatensätzen von Bedeutung.

2.2.2 Definition der Parameter

In diesem Absatz werden die benötigten Parameter des Vorhersagemodells beschrieben. Die Parameter können in zwei Gruppen unterteilt werden. Die erste Gruppe sind konstante Größen, welche im praktischen Teil der Anwendung des Modells nicht verändert werden. Demgegenüber können die restlichen Parameter in die Gruppe von variablen Parametern eingeteilt werden, weil diese je nach Durchlauf angepasst werden.
Tabelle 1: Unterteilung der Modellparameter in variable und konstante Werte stellt die Unterteilung der oben beschriebenen Gruppe der Modellparameter dar. Die Parameter k-Distanz und k-Wert sind sinngemäß die gleichen Parameter, welche aber an unterschiedlichen Stellen des Verarbeitungsprozesses eingesetzt werden und somit eine Unterscheidung notwendig ist.
Der Wert k beschreibt grundsätzlich die Anzahl der maximalen nächsten Nachbarn, welche in weiteren Schritten berücksichtigt werden sollen. Die möglichen Werte sind in beiden Fällen ungerade Zahlen im Intervall 1 bis 9 und entsprechen den Zahlen 1,3,5,7,9.

Mithilfe des k-Distanz Parameters wird die Auswahl der für den Algorithmus zu berücksichtigende streckentreuen Polylinien gesteuert. Dies dient der Performanz-

Optimierung, um schon vorab so viele nicht relevante Linien wie möglich auszuschließen und so die Menge der zu berechnenden Möglichkeiten gering zu halten. Es werden zu jedem Punkt die k nächsten streckentreuen Polylinien immer auf Basis der euklidischen Distanz ermittelt. Wurde eine streckentreuen Polylinien von keinem Punkt als k nächster Nachbar identifiziert, so wird diese entfernt und im weiteren Verlauf des Algorithmus nicht mehr berücksichtigt.

Der k-Wert Parameter legt die Anzahl der zu betrachtenden k nächsten Nachbarn für die Klassifikation im Rahmen des *k-Nearest Neighbor Classifier* fest. Dieser Parameter ist eng mit dem Parameter **p** verbunden, weil die Entscheidung über die Nähe der Objekte zueinander aufgrund der Berechnung von einer kürzesten Distanz erfolgt. In diesem Fall gibt es zwei unterschiedliche Distanzmaße, welche für die Berechnung eingesetzt werden können. Dazu gehört die euklidische Distanz, welche den Wert p=2 entspricht und die Manhattan Distanz mit dem Wert p=1. Die Distanzmaße sind im Absatz 2.2.3 Metrische Distanzen definiert.
Die Parameter **k-Wert**, **p** und **algorithm** sind Bestandteile des *k-Nearest Neighbor Classifier*. Der Parameter **algorithm** entscheidet wiederrum über die Auswahl des zugrundeliegenden Algorithmus, anhand dessen die Klassifikation der unbekannten Daten, der extrahierten Punkte aus den streckentreuen Polylinien und der Vorhersage der TECH_ID stattfindet. In dieser Abschlussarbeit spielt die Auswahl des Algorithmus keine Rolle für die Höhe der Modellgüte. Die möglichen Algorithmen sind für die Performanz-Steigerung der Rechenarbeit des Computers verantwortlich und werden im Rahmen dieser Abschlussarbeit nicht näher erklärt.

Parameter		Beschreibung		
Name	**Art**	**mögliche Werte**	**Bedeutung**	**Beschreibung**
k-Distanz	variabel	(1,3,5,7,9)	Anzahl der zu betrachtenden nächsten Nachbarn	**k**-Wert für die Auswahl der relevanten Linien zu dem hausgenauen Punkten
k-Wert	variabel	(1,3,5,7,9)	Anzahl der zu betrachtenden nächsten Nachbarn	**k**-Wert für die Auswahl der nächsten Nachbarn zur Vorhersage der TECH_ID
p	variabel	(1,2)	Distanzmaß, wo **p**=1 für die Manhattan Distanz und **p**=2 für die euklidische Distanz steht	Distanzmaß der Minkowski Familie; Parameter des k-Nearest Neighbor Classifier
algorithm	konstant	auto	Anhand der **fit** Methode wird automatisch unter 'ball_tree', 'kd_tree' oder 'brute' Algorithmus entschieden	Automatische Auswahl des grundlegenden Algorithmus, welcher für die Vorhersage eingesetzt wird. Reine Perfomanceunterschiede bei unterschiedlicher Datenlage

Tabelle 1: Unterteilung der Modellparameter in variable und konstante Werte, eigene Darstellung.

Die Ergebnisse der praktischen Anwendung des k-*Nearest Neighbour Classifier* zeigen eine starke Abhängigkeit von der Höhe des **k-Wertes** als Parameter zu der Modelgüte (siehe dazu Absatz 3.1.2 Einfluss des k-Wertes auf die Modellgüte). Demgegenüber hat sich rausgestellt, dass der **k-Distanz** Wert keinerlei Auswirkung auf die Modellgüte hat (siehe dazu 3.1.3.1 Einfluss des k-Distanz Wertes auf das Modell). Die aus der praktischen Arbeit gewonnene Erkenntnisse werden im Kapitel 4 Diskussion dargelegt. Folgend werden die Distanzmaßen des **p** Parameters aus der Minkowski Familie näher definiert.

2.2.3 Metrische Distanzen

In der Geographie wird Raum als ein Container definiert (Gatrell, 1983, p. 2). Anhand dieser Assoziation kann die Suche auf Basis einer metrischen Distanz nach einem nächsten Nachbarn besser geschildert werden. Das Auswahlverfahren des nächsten Nachbarn basiert auf der Feststellung der Ähnlichkeit der Objekte zueinander in der Ausdehnung des Containers als Raum. Die Definition der Ähnlichkeit erfolgt über die Nähe und ist vom jeweiligen Anwendungsfall abhängig.

In dieser Abschlussarbeit wird die kürzeste Distanz anhand zweier unterschiedlicher Distanzmetriken als Nähe für die Anwendung des k-*Nearest Neighbor Classifier* in einem kompakten Raum und mit einfachen Topologien definiert.

Aus der Minkowski Familie wird die euklidische- und die Manhattan-Distanz angewendet. Die formale Definition einer Metrik ist in Gatrell (1983) beschrieben und nachzulesen.

Die euklidische Distanz d_E im zweidimensionalen Raum zwischen den Objekten i und j ist wie folgt definiert:

$$d_E(i,j) = \sqrt{[(x_{i\,1} - x_{j\,1})^2 + (x_{i\,2} - x_{j\,2})^2]}$$

Unter Berücksichtigung der möglichen Mehr-Dimensionalität eines Raumes mit **m** Dimensionen wurden die Distanzmaße der Minkowski Familie mit einem zusätzlichen Parameter **p** generalisiert und als *Minkowski p-metrics* zusammengefasst. Dieser Parameter **p** ist eine konstante Zahl, welche von 1 bis unendlich gesetzt werden kann. Im Falle, wenn **p** kleiner als 1 wäre, würde die Dreiecksungleichung nicht mehr gelten und somit die Eigenschaft einer Metrik verletzt.

Die verallgemeinerte Form der *Minkowski p-metrics* sieht wie folgt aus:

$$d_p(i,j) = \left[\sum_{k=1}^{m} \left|x_{i\,k} - x_{j\,k}\right|^p\right]^{1/p}$$

$$p \geq 1$$

Die Manhattan Distanz ist definiert als die *Minkowski p-metrics* bei p=1*,* während die euklidische Distanz p=2 entspricht. Diese Parameterkombinationen finden sich dann auch bei der Anwendung des k-*Nearest Neighbor* Algorithmus wieder.
Während die euklidische Distanz in dieser Abschlussarbeit die „Luftlinienentfernung" zweier Punkte repräsentiert, entspricht die Manhattan Distanz eher einer „Straßenführung" und es ist daher interessant, welche Distanzmetrik geeigneter für den Anwendungsfall erscheint (siehe dazu auch Absatz 3.1.3.2 Einfluss der Distanzmetrik auf das Modell).

2.2.4 Festlegen der Bewertungskriterien für das *lazy learning* Modell

Um das Modell und deren unterschiedlichen Parameterkonstellationen bewerten und vergleichen zu können, wird ein Gütemaß definiert:
Die Modellgüte eines Modells **i** ist definiert als die Anzahl der Punkte $\mathbf{P_i}$ der Testdatenmenge $\mathbf{T_i}$ des Modells mit korrekter Vorhersage der TECH_ID geteilt durch die Gesamtanzahl aller Punkte $\mathbf{P_i}$ aus der Testdatenmenge $\mathbf{T_i}$.

$$Modellgüte_i = \frac{|(\mathrm{P_i}\,|mit\ korrekt\ vorhergesagter\ TECH\_ID)\,|}{|P_i|}$$

wobei:

$$P_i \in T_i$$

Die Modellgüte wird nur auf den Testdaten, also den Daten, die das Modell nicht zum Training benutzt hat, ermittelt. Somit simuliert man den Einsatz des Modells auf unbekannten Daten. Da man allerdings auch das Ergebnis der Testdaten kennt, kann man

die Aussage treffen, ob das Modell die richtige Vorhersage getroffen hat und somit das Gütemaß bestimmen.

Die Modellgüte liegt somit im Bereich von 0%-100%.

Je höher die Modellgüte, desto besser ist die Vorhersagekraft des Modells.

Die angenommene Zielgröße der Vorhersage bei der das Modell als korrekt bewertet wird, liegt bei einer Modellgüte von mindestens 80 %. Dieser Wert ergibt sich aus einem internen Projekt von Unitymedia NRW GmbH bei dem mithilfe anderer Methoden 80% des Kabelnetzbestandes nachträglich vektorisiert wurden, ohne die Möglichkeit die Qualität der Ableitung des Kabelnetzverlaufes überprüfen zu können. Da mithilfe des *lazy learning* Modells zu jedem Punkt eine Vorhersage getroffen werden kann, spielt hier die Vollständigkeit keine Rolle, sondern die Qualität der Ableitung, die durch die Modellgüte ausgedrückt wird. Die Ergebnisse werden in Kapitel 3 beschrieben.

2.3 Prozessübersicht

Im Rahmen dieser Abschlussarbeit wurde eine Vorgehensweise zur Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien entwickelt. Die einzelnen Schritte werden anhand Abbildung 9: Prozessübersicht zur Ableitung von physischen Kabelverläufen erklärt.

Der gesamte Prozess kann im Hinblick auf die ausführende Instanz in zwei Blöcke unterteilt werden. Der Kern des Prozesses ist eine Abfolge von Operationen gespeichert in einem Python Skript und der zweite Block beschreibt einzelne Operationen in einer Geoinformationssoftware. Die Schritte der Datenaufbereitung und der Modell Evaluation, welche außerhalb des Python Skriptes stattfinden, können je nach Datengrundlage variieren. In einem solchen Fall kann es zu der Situation kommen, dass bestimmte Schritte entfallen werden, wie z.B. das Duplizieren des OSM- Straßenlayers im Abstand von drei Metern. Denkbar ist auch eine Integration von der Model Evaluation in das Python Skript, wo es automatisiert abgearbeitet werden können.

Die in Absatz 2.4 Beschreibung der Datengrundlage verwendete Daten benötigen eine Vorverarbeitung, bevor diese im Python Skript verarbeitet werden können. Diese Notwendigkeit ergibt sich aus der Größe und der unterschiedlichen Typen oder Struktur der Daten. Die Anzahl der hausgenauen Punktdaten wird mit Hilfe der GIS-Software ArcGIS Pro der Firma Esri verkleinert, indem diese auf ein festgelegtes Untersuchungsgebiet abgegrenzt werden. Die Beschreibung des Prozesses der Verschneidung ist im Absatz 2.5.1 Abgrenzung des Testgebietes geschildert. Die streckentreue Polylinien werden in dem gesamten Prozess mehrfach genutzt, deswegen bedürfen diese einer mehrfachen Modifikation je nach ihrer Zielsetzung. Die Struktur und die Besonderheiten des *OpenStreetMap* Straßenlayers werden im Absatz 2.4.2 streckentreue Polylinien beschrieben. Der Schritt der Umwandlung des Straßenlayers in eine doppelseitige Straßenführung passt die Datengrundlage an die Struktur von realen Kabelnetzstrecken an. Die zweite Stufe dieser Vorgehensweise ist eine Auswahl von bestimmten Straßentypen, von denen der reale Verlauf des Kabelnetzes abgeleitet werden könnte. Die notwendige Operationen der Umwandlung und die einzelne Auswahlkriterien sind im Absatz 2.5.2 Nachbilden der OpenStreetMaps Straßenzüge beschrieben. Die angepasste Struktur des Straßenlayers als eine duplizierte Polylinie wird für die Anwendung des Vorhersagemodells modifiziert und als Punktelayer gespeichert. Diese Anpassung ist im Absatz 2.5.3 Extrahieren von Punktdaten näher erklärt.

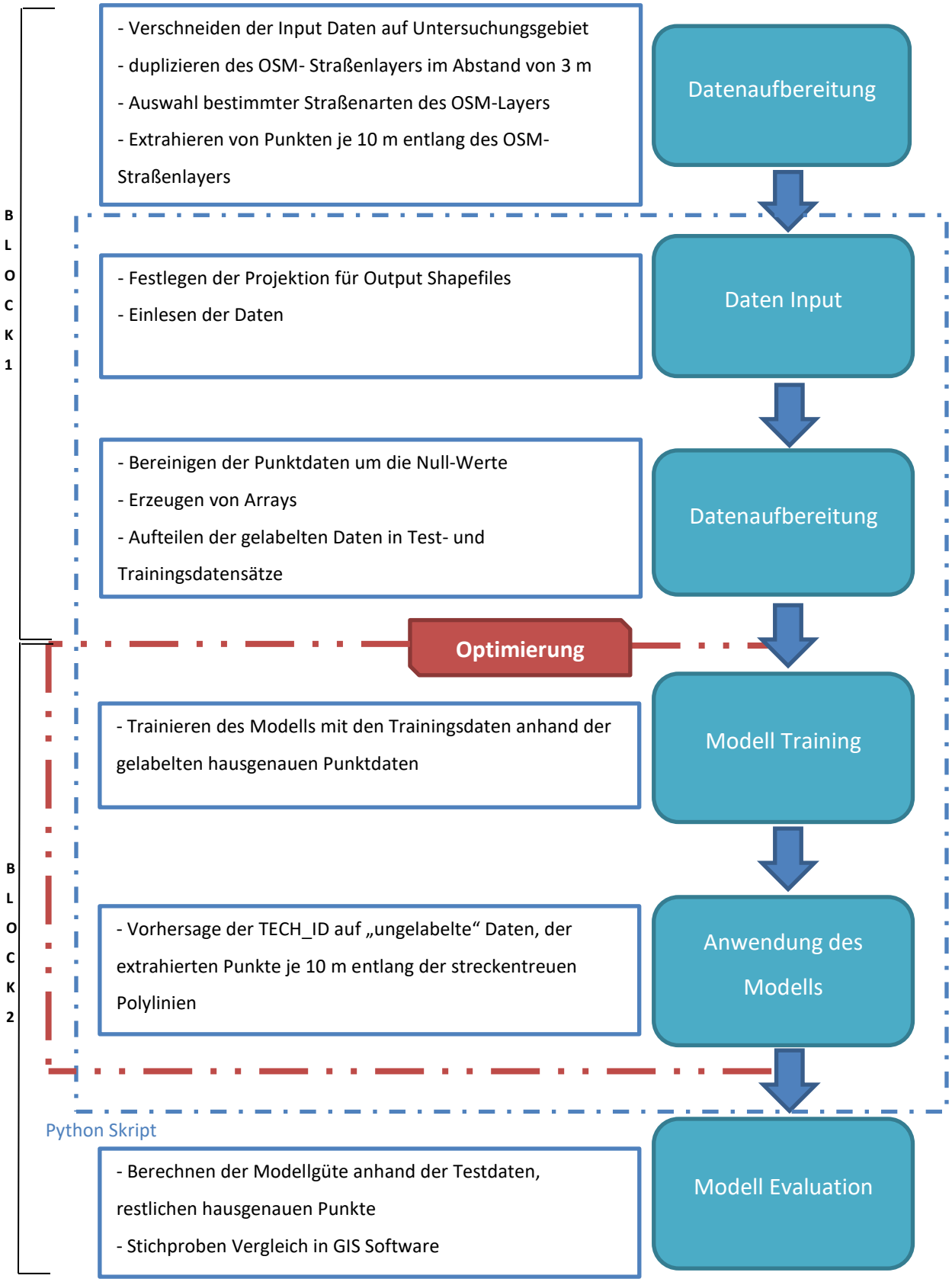


Abbildung 9: Prozessübersicht zur Ableitung von physischen Kabelverläufen, eigene Darstellung.

Die in diesem Kapitel beschriebene Methoden der räumlichen Verbindung (nachzulesen im Absatz 2.1 Spatial join, die räumliche Verbindung) und der Vorhersage mittels k-*Nearest Neighbor Classifier* (siehe dazu Absatz 2.2 Das *lazy learning* Modell zur Ableitung von Kabelverläufen) stehen zum Teil in Python Bibliotheken zur Verfügung, die verwendete Bibliotheken wurden in der Einleitung vom Kapitel 2 Methodik beschrieben.

Der zweite Block des Prozesses stellt die Schritte zur Erstellung des Vorhersagemodells dar, welche mit der Skriptsprache Python definiert sind. Die Abfolge der einzelnen Anweisungen ist in der Abbildung 10: Detaillierter Prozess der Modellvorhersage visualisiert. In der Darstellung des Modells sind die hausgenauen Punkte als „gelabelte“ Punkte, streckentreue Polylinien als Linien aus *OpenStreetMap* und die je zehn Meter extrahierte Punkte aus streckentreuen Polylinien als „ungelabelte“ Punkte mit OSM-ID bezeichnet. Das hier genannte „Label“ soll vorhersagt werden und bezieht sich auf die technische Identifikationsnummer der hausgenauen Punkte (TECH_ID). Diese Information wird für jeden Punkt der „ungelabelten“ Punkte mit OSM-ID vorausgesagt.
Die Eingabe Daten liegen in Shapefile Format der Firma Esri vor. Diese werden automatisiert eingelesen und in ein Dataframe-Format aus der Pandas Bibliothek umgewandelt. Das festgelegte geographische Koordinatensystem für die neu erzeugte Shapefile Dateien ist das geodätisches Referenzsystem World Geodetic System 1984 (WGS84). Die „gelabelte“ Punkte werden um alle Werte, die keinen Eintrag haben, bzw. als NULL Wert gespeichert sind bereinigt. Folgend werden die fachlichen Informationen über die technische Identifikationsnummer in ein Array Datenformat gespeichert, so dass eine weitere Datenverarbeitung optimal erfolgen kann.
Die Linien aus *OpenStreetMap* werden zur Ermittlung relevanter Linien eingesetzt. Diese Operation bedient sich an den k-*Nearest Neighbor* Algorithmus, der k Anzahl von den an den nächsten liegenden Linien aus *OpenStreetMap* zur einem „gelabelten“ Punkt ermittelt. Die Suche von dem nächsten Nachbarn basiert auf einer Abstandmessung, diese bedarf der Angabe von geographischer Lage im Koordinatensystem von den Objekten. Diese

Information über die geographische Länge und Breite ist in der Geometrie Spalte gespeichert. Die Werte werden als Paare in einen Array-Format gespeichert und wie im Absatz 2.2.3 Metrische Distanzen in die mathematische Formel zur Berechnung der euklidischen Distanz eingesetzt. Die **k** Anzahl Linien aus *OpenStreetMap* wird bereinigt um Mehrfach-Einträge aus der Auswahl. Die eindeutigen Linien werden in Shapefile-Format in eine externe Datei exportiert und zu der Ermittlung von relevanten „ungelabelten" Punkten weiterverwendet. Auf Basis der führenden Identifikationsnummer des *OpenStreetMap* Straßenlayers (OSM-ID) werden nur die „ungelabelte" Punkte, welche auf einer relevanten Linie liegen und die gleiche OSM-ID haben ausgewählt.

Im Rahmen der praktischen Anwendung dieser Abschlussarbeit wird die Sortierung als eine Teilfrage definiert (siehe Absatz 1.4 Forschungsfrage). Diese stellt eine potentielle Optimierungsmöglichkeit des Modells dar. Es wird erwartet, dass das Sortieren der „gelabelten" Punkten nach der technischen Identifikationsnummer erst im Schritt Modelltraining einen Einfluss auf das Endergebnis hat. Diese Daten werden vor dem Einsatz des Vorhersagemodells in Test- und Trainingsdaten aufgeteilt. Die Zuordnung der „gelabelten" Daten zu Test- und Trainingsdaten erfolgt zufällig aber reproduzierbar im Verhältnis 20% Testdaten zu 80% Trainingsdaten. Das Trainieren des Modells mit dem *lazy learning* k-*Nearest Neighbor Classifierer* lässt eine Eingabe von Parametern zu. Die in diesem Fall festgelegten Parameter sind im Einzelnen im Absatz 2.2.2 Definition der Parameter beschrieben. Der k-Wert und die Distanzmetrik als veränderbare Parameter werden im Rahmen dieser Abschlussarbeit diskutiert. Das Anwenden des trainierten Musters auf die „ungelabelte" relevante Punkte wird anhand einer selbst definierten Modellgüte (siehe dazu Absatz 3.1.2 Einfluss des k-Wertes auf die Modellgüte) bewertet. Die Modellevaluation basiert auf den Testdaten, auf welche das gleiche trainierte Muster angewendet wird. Die Vorhersage auf die „ungelabelte" relevante Punkte wird in eine externe Datei in Shapefile Format aber auch in einer tabellarischen Form exportiert.

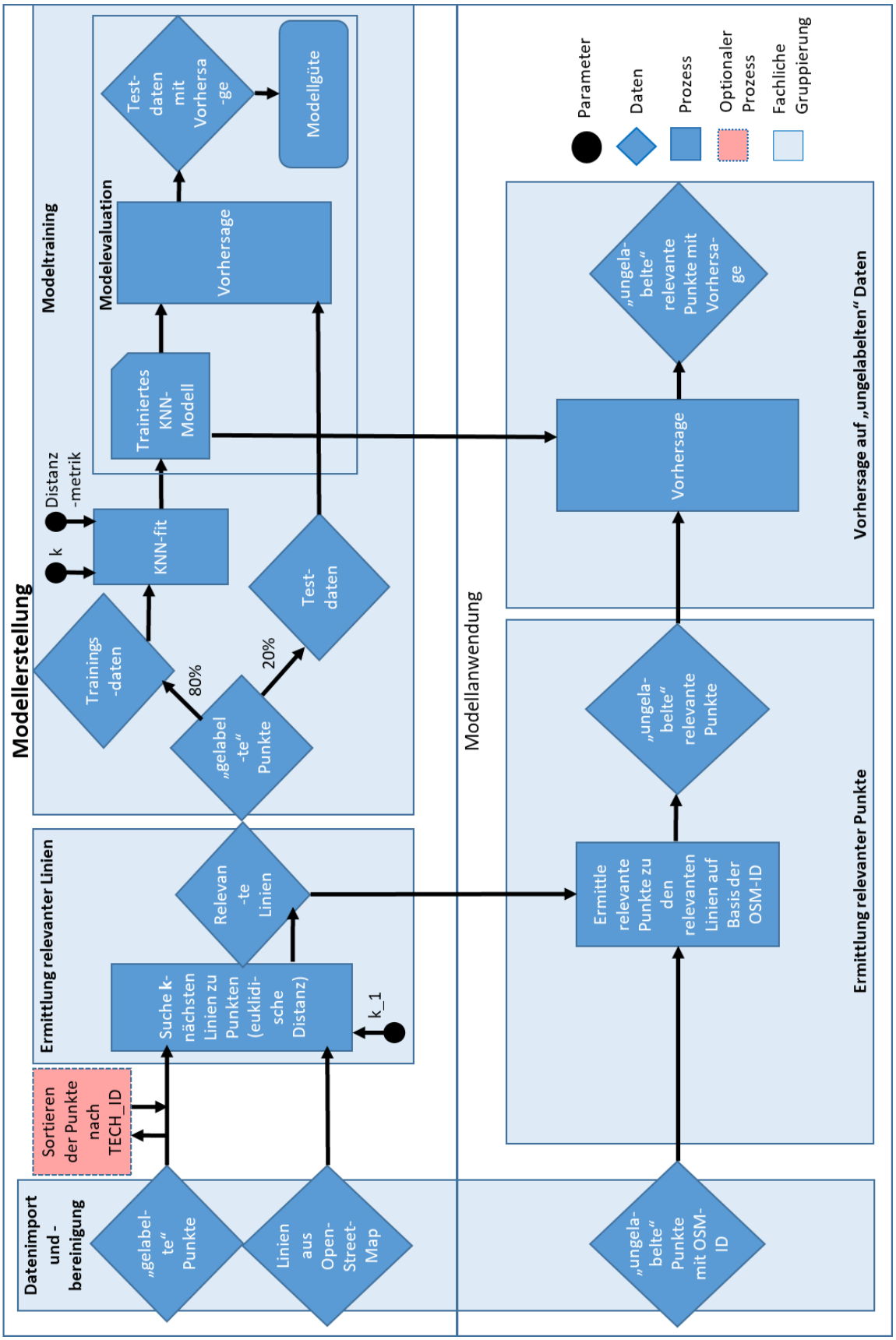


Abbildung 10: Detaillierter Prozess der Modellvorhersage, eigene Darstellung.

2.4 Beschreibung der Datengrundlage

Das Datenmodell in Geographischen Informationssystemen(GIS) besteht oft aus zwei unterschiedlichen Datentypen: Daten mit räumlichem Bezug (Geometriedaten) und Attributdaten (Sachdaten) ohne graphischen Bezug, welche sich aber durch weitere Merkmalen oder fachlichen Informationen auszeichnen. In der Praxis werden im Zuge einer elektronischen Datenverarbeitung die Geometriedaten mit den Attributdaten verbunden und nach Bedarf um weitere GIS charakteristische Merkmale angereichert.

In dieser Abschlussarbeit werden Daten mit räumlichem Bezug in zweidimensionaler Ausprägung verarbeitet, d.h. die Geometrie liegt als X,Y Koordinaten unter der Angabe von geographischer Länge und Breite vor. Diese sind auch als räumliche Objekte zu sehen, welche nach der Digital Cartographic Data Standard Task Force (DCDSTF) genauer durch die Ausdehnung klassifiziert werden können. (U.S.GeologicalSurvey, 2006) Zusammengefasst sind für diese Abschlussarbeit die räumlichen Objekttypen der 0, 1 und 2 topologischen Dimension von Bedeutung (siehe dazu Tabelle 2: räumliche Objekttypen, verändert nach (Kappas, 2012, p. 55))

Klassifikation räumlicher Objekttypen				**räumliches Datenmodell**
Name	**topologische Dimension**	**Objekttyp**	**genutzte Daten Abschlussarbeit**	**Vektordatenform**
Nullzellen	0	Punkt	hausgenaue Punktdaten	(x,y)
Einszellen	1	Polylinie	streckentreue Polylinien	(x_1,y_1), (x_2,y_2), ... (x_n,y_n)
Zweizellen	2	Polygon	Stadtgebietsgrenzen von Köln	(x_1,y_1), (x_2,y_2), ... (x_n,y_n) bei $[(x_1,y_1)=(x_n,y_n)]$

Tabelle 2: räumliche Objekttypen, verändert nach (Kappas, 2012, p. 55)

In der Geographie ist der Punkt als das einfachste Objekt und somit die kleinste topologische Einheit zu sehen. Es ist eine Entität, welche mit einem Koordinaten-Paar, der geographischen Länge und Breite beschrieben wird. Weitere Topologien basieren auf dem Punkt, somit setzt sich eine Linie aus einem Start und Endpunkt zusammen, welche verbunden sind und als ein Liniensegment bezeichnet werden. Eine Polylinie ist wiederum ein Zusammenschluss von mehreren Liniensegmenten, deren Position Start-, End- und Mittelpunkt in der Geometrie Spalte gespeichert sind. Die nächste Stufe ist eine Darstellung geschlossenen Fläche als ein Polygon, welcher ein Verbund von geschlossenen Linien ist. In einer 3- dimensionalen Ansicht wird zusätzlich zu den Topologien die Information über die Höhe des Objektes gespeichert. (Kappas, 2012, pp. 54-56).

2.4.1 hausgenaue Punktdaten

Die hausgenaue Punktdaten sind wie oben beschrieben Geometriedaten der 0-topologischen Dimension und liegen als Punkt mit den X, Y Koordinaten in vektorisierter Form als Shapefile vor. Dies ist ein Datentyp der Firma Esri, der sich zum Standard von Geographischen Informationssystemen entwickelt hat. Auf Anfrage bei Unitymedia NRW GmbH wurden für die wissenschaftliche Zwecke diese hausgenauen Daten zur Verfügung gestellt. Diese beinhalten Informationen aus einer relationalen Datenbank zur Beschreibung der Kabelnetzinfrastruktur der Unitymedia NRW GmbH.

Die geographische Lage des Punktes ist immer der Mittelpunkt eines Hausumringes nach dem Amtlichen Katasteramt der zuständigen Behörde. Die Gebäudegrundrisse sind potentiell seit 01.01.2017 nach den *Open Data* Prinzipien kostenfrei abrufbar. (Bezirksregierung_Köln, 2019) Diese verfügbaren Sachdaten wurden allerdings nicht für diese Abschlussarbeit genutzt, sondern ausschließlich auf den durch die Unitymedia NRW GmbH zur Verfügung gestellten, bereits verarbeiteten Punkte der Hausumringe, gearbeitet.

Wie in Kapitel 1.5 Abgrenzung der Abschlussarbeit beschrieben, wurde die Datengrundlage auf ein festgelegtes Untersuchungsgebiet verkleinert. Die detaillierte Vorgehensweise des Auswahlprozesses der Input Datensätze zur praktischen Anwendung zur Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien ist im nachfolgenden Unterkapitel 2.5.1 Abgrenzung des Testgebietes skizziert.

Der Ausgangswert der fachlichen Einträge ist die postalische Adresse mit Kombination von amtlicher Bezeichnung des Grundstückes und daran angehängten technischen Attributen, welche die Aussage über den Kabelanschluss und seinen Zustand treffen. Die Struktur der Infrastrukturdokumentation für ein Telekommunikations-Kabelnetz weißt eine besondere Hierarchie auf, welche bei der Vergabe der technischen Attribute wiederzufinden ist. Die hier verwendete Datensätze beschreiben ein Verzweigerkabelsegment mit eindeutigen systematisch vergebenen Identifikationsnummern. Diese und die einzelnen Attribute der hausgenauen Punkte sind in der nachfolgenden Tabelle 3: Attribute der hausgenauen Punktdaten aufgelistet. Zu den obligatorischen Angaben eines Shapefiles gehören Angaben, wie die laufende Nummer der Ausdehnung des Shapefiles (Shape ID), die Vektor Geometrie (GEOMETRY) und der Name des Objekttypes in Spalte Shape. Die Sachdaten, welche in dem Kabelnetzwerk beschrieben sind werden in den Spalten Verzweigerkabel, Kabelverzweiger und Hauptverteiler Identifikationsnummer gespeichert. Im praktischen Teil der Abschlussarbeit wird ein *lazy learning* Modell zur Ableitung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien getestet. Das Vorhersagemodell wird anhand des Attributes technische ID (TECH_ID) trainiert und den zu verbindenden streckentreuen Polylinien, welche als Punkte extrahiert wurden anhand der kürzesten Distanz untereinander zugeordnet. Die Vorgehensweise ist im Kapitel 2.5.3 Extrahieren von Punktdaten ausführlich beschrieben. Der Umgang mit Ausreißern der Eingabe-Daten ist im Kapitel 2.5.4 Umgang mit unvollständigen Datensätzen beschrieben.

Spaltenname	Beschreibung	Beispiel
OBJECTID_1	Shape_ID	1
SHAPE	Typ des Shapes (z.B. Linie oder Punkt)	Point
X	Längengrad	2572745,554
Y	Breitengrad	5650387,862
GRUNDST_ID	Grundstück ID	2932466
AGS27	Hausgenauer Schlüssel Infas	05315000010000000350305001
STR_HNR_Z	Straße Hausnummer und Hausnummerzusatz	Im Weidenkamp 1
PLZ	Postleitzahl	51061
ORT_ADRESS	Ort	Köln
ORTSZUSATZ	Ort Zusatz	
ORTSTEIL_A	Ortsteil	Höhenhaus
GEMEINDENA	Gemeinde Name	
VZK_ID	Verzweigerkabel Identifikationsnummer	1311E-20-013
KVZ_ID	Kabelverzweiger Identifikationsnummer	1311E-20
HVT_ID	Hauptverteiler Identifikationsnummer	1311E
TECH_ID	Technische Identifikationsnummer Koaxialkabels	1105645
GEOMETRY	Geometrie Vektor	POINT 2572745.6568 5650387.3894

Tabelle 3: Attribute der hausgenauen Punktdaten, eigene Darstellung

Die Vergabe einer technischen ID sagt aus, dass eine Hauszuführung und voraussichtlich ein Endverzweiger (siehe dazu Exkurs) an dieser Adresse vorhanden sind. Ein Bezug zu aktiven Kundenverträgen, kann nur intern beim Kabelnetzbetreiber festgestellt werden. Die Punkte, wo keine Information über das Vorhandensein vom Kabelnetz vorliegt, sind als NULL Werte gespeichert. In diesem Fall ist davon auszugehen, dass dieses Objekt nicht an das Kabelnetz angeschlossen ist. Abbildung 11: Beispiel der Verteilung der hausgenaue Punktdaten nach TECH_ID, eigene Darstellun zeigt solche Fälle auf und schildert anhand von einem Beispiel die räumliche Aufteilung der NULL Werte. Fehlende Werte können aber auch inkonsistente Daten sein, welche der mangelnden Datenqualität von erfassten Einträgen geschuldet sein können. Manchmal können keine Angaben falsch interpretiert werden, deswegen ist eine unternehmensinterne Datenqualitätssicherung der Fachdaten unabdingbar (Reinhardt and Bockmuehl, 2013). Im Kapitel 2.5.4 Umgang mit

unvollständigen Datensätzen wird die Bedeutung von unvollständigen Daten für das Vorhersagemodell näher erklärt.

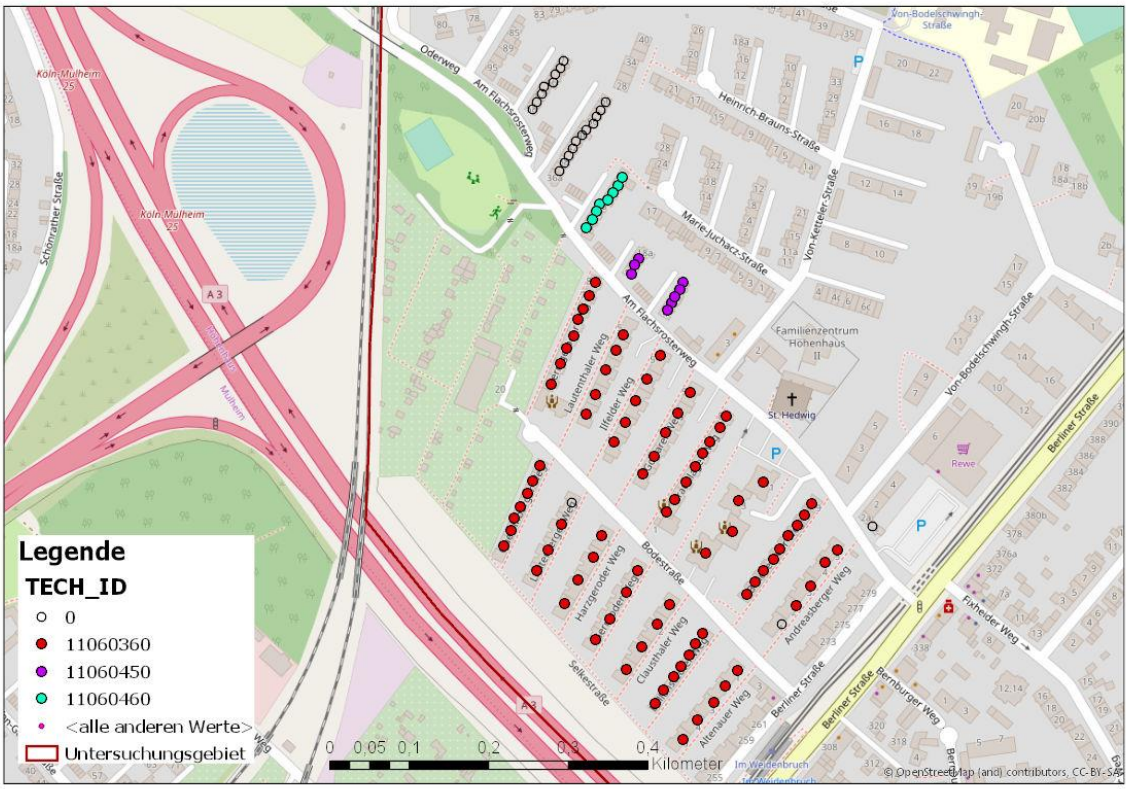


Abbildung 11: Beispiel der Verteilung der hausgenaue Punktdaten nach TECH_ID, eigene Darstellung

2.4.2 streckentreue Polylinien

Die zweite Datenquelle der Abschlussarbeit sind die streckentreuen Polylinien, auf welchen die Vorhersage der technischen Identifikationsnummer getroffen wird.
Genutzt wurde ein Straßenlayer der *OpenStreetMap Foundation* für die Region Bezirksregierung Köln im Format Shapefile. Die Daten wurden von der Internetseite der Geofabrik GmbH am 08.01.2019 bezogen. (OpenStreetMap, 2019a)

Allgemein ist das *OpenStreetMap-Projekt* im Rahmen einer Initiative zur Erstellung von frei zugänglichen geographischen Daten entstanden. Diese Organisation wirkt international und gehört zu den *not-for-profit* Unternehmen, welche aus freiwilligen Spenden finanziert werden. (OpenStreetMap, 2019b)

Das Prinzip des Aufbaus und der Aktualisierung des *OpenStreeMap* Layers beruht auf einer Gemeinschaftsarbeit von Nutzern des Produktes. Die erfassten Daten können von jedem angemeldeten Benutzer auf freiwilliger Basis ergänzt oder verändert werden. Die Nutzung der Daten setzt keine Lizenzgebühren voraus, jedoch verpflichtet diese nach der „Creative Commons Attribution-Share Alike 2.0“- Lizenz zu einer genauen Quellen Angabe. (OpenStreetMap, 2019c)

Die heruntergeladenen Daten beinhalten eine Reihe von einzelnen Layern mit unterschiedlichen Topologien und Informationsgehalt, welche gesondert kodiert sind. Eine Klassifikation der einzelnen Datentypen ist über die Spalte Code möglich. Die Zuordnung der Werte ist in der Katalog Übersicht in der Dokumentation des Paketes beschrieben (siehe dazu (Ramm, 2019)).

In dieser Abschlussarbeit wurde der Layer: „gis_osm_roads_free_1“ als Linien-Geometrie verwendet. Die Anzahl der Attribute der Klasse *roads* liegt bei 450.074 Liniensegmenten. Die Ausdehnung der Daten für die Region Köln erstreckt sich über die üblichen Stadtgrenzen und ist genau in der Abbildung 12: Ausdehnung der OpenStreetMap daten für die Region Regierungsbezirk Köln dargestellt.

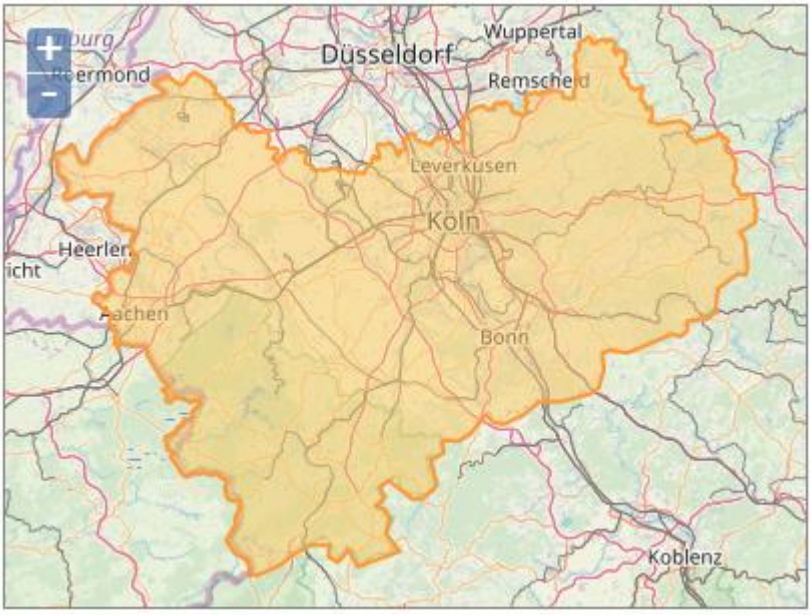


Abbildung 12: Ausdehnung der OpenStreetMap daten für die Region Regierungsbezirk Köln, (OpenStreetMap, 2019a)

Die geographische Projektion der Daten wurde beibehalten und ist gleich dem geodätischen Referenzsystem GCS WGS 1984 mit dem EPSG Code 4326. (OpenStreetMap, 2019a)

Die Struktur der Attributtabelle des Layers beinhaltet die Standardwerte der *OpenStreetMap* Eigenschaften (osm_id, code, fclass, name), sowie die zusätzlichen Bezeichnungen der Linien-Geometrie *roads* (ref, oneway, maxspeed, layer, bridge, tunnel). Die Spaltennamen und deren Bedeutung mit Beispielen sind in der Tabelle 4 dargestellt.

Spaltenname	Beschreibung	Beispiel
OBJECTID	Shape_ID; eindeutiger Wert	9
osm_id	ID des Attributes; Strassen ID; eindeutiger Wert	9475852
code	4 -stelliges Schlüssel der Klasse; erste 2-Stellen definieren die Art des Layers	**51**14
fcclass	Beschreibung der Klasse des Attributtes	secondary
name	Eigenname des Attributtes; Strassenname	Dünnwalder Kommunalweg
ref	Referenznummer der Strasse	L 101
oneway	Ist das eine Einbahnstrasse?; Wert: **F** sagt ja, nur in Richtung der Linie; **T** sagt ja, nur in die entgegengesetzte Richtung; **B** ist Standardwert, Verkehr in beiden Richtungen möglich	B
maxspeed	Maximal zugelassene Geswindigkeit in **km/h**	50
layer	Relative Angabe der Schichtung der Straße	0
bridge	Liegt diese Straße auf einer Brücke?; **T** steht für ja; **F** steht für nein	F
tunnel	Verläuft diese Straße in einem Tunnel?; **T** steht für ja; **F** steht für nein	F
Shape_Length	Länge ders Features; in diesem Fall Angabe in **m**	126,671849

Tabelle 4: Spaltennamen der Attributtabelle OpenStreetMap roads Layer, verändert nach (Ramm, 2019, pp. 14-15)

Die Schritte der Datenaufbereitung und die einzelne Modifikationen des *OpenStreetMap* Straßenlayers sind im Kapitel 2.5 Schritte der Datenvorverarbeitung im Einzelnen beschrieben.

Die Auswahl dieser Datengrundlage ist mit der freien Verfügbarkeit der Daten für das Untersuchungsgebiet (siehe dazu Absatz 2.5.1 Abgrenzung des Testgebietes) im Shapefile Format zu begründen.

2.4.3 Stadtgebietsgrenzen von Köln

Der im Absatz 2.3 Prozessübersicht beschriebener Weg zur Ableitung von physischen Kabelverläufen bedarf einer entsprechenden Datengrundlage zur Abarbeitung von einzelnen Schritten des Prozesses.
Wie im vorherigen Unterpunkt beschrieben, erstreckt sich die Ausdehnung der streckentreuen Polylinien über die amtlichen Grenzen der Stadt Köln. Mit dem Ziel die Anzahl der Trainingsdaten und Testdaten des Vorhersagemodells klein zu halten, wurden die streckentreue Polylinien und die hausgenaue Punktdaten mit den Stadtgebietsgrenzen von Köln verschnitten. Die Operation im Detail wir in der Einleitung zu dem nachfolgenden Absatz 2.4 Beschreibung der Datengrundlage beschrieben.

Die Datengrundlage wurde aus dem Portal der Offenen Daten Köln im Shapefile-Format bezogen. Diese Seite wird von der Stadt Köln und DKAN zur Förderung der „Open Data Aktivitäten“ betrieben. Die zur Verfügung gestellten Daten können im Sinne der Creative Commons Namensnennung 3.0 Deutschland Lizenz mit dem Eintrag: „Datenquelle: Stadt Köln – offenedaten-koeln.de“ frei genutzt werden.
Die Basis des Portals sind öffentlich zugängliche Informationen, welche in unterschiedlichen Formaten thematisch geordnet zur Verfügung gestellt werden. Zu den gestellten Formaten gehören tabellarische Auflistungen in Excel-Format aber auch geographische Formate, wie JSON, Shapefile und KMZ Layer, sowie im XML-Format. (StadtKöln, 2019)

Die Stadtgebietsgrenzen wurden in Shapefile-Format in der Projektion ETRS 1989 / UTM Zone 32N heruntergeladen. Die Fläche ist als Polygon-Geometrie der kommunalen Gebietsgliederung der Stadtbezirke von Köln gespeichert. Die Datengrundlage ist aktuell und zuletzt am 07.03.2018 vom Verfasser aktualisiert worden.
Die folgende Abbildung 13: Stadtgebietsgrenzen von Köln zeigt die Richtigkeit der Abgrenzung des Stadtgebietes Köln.

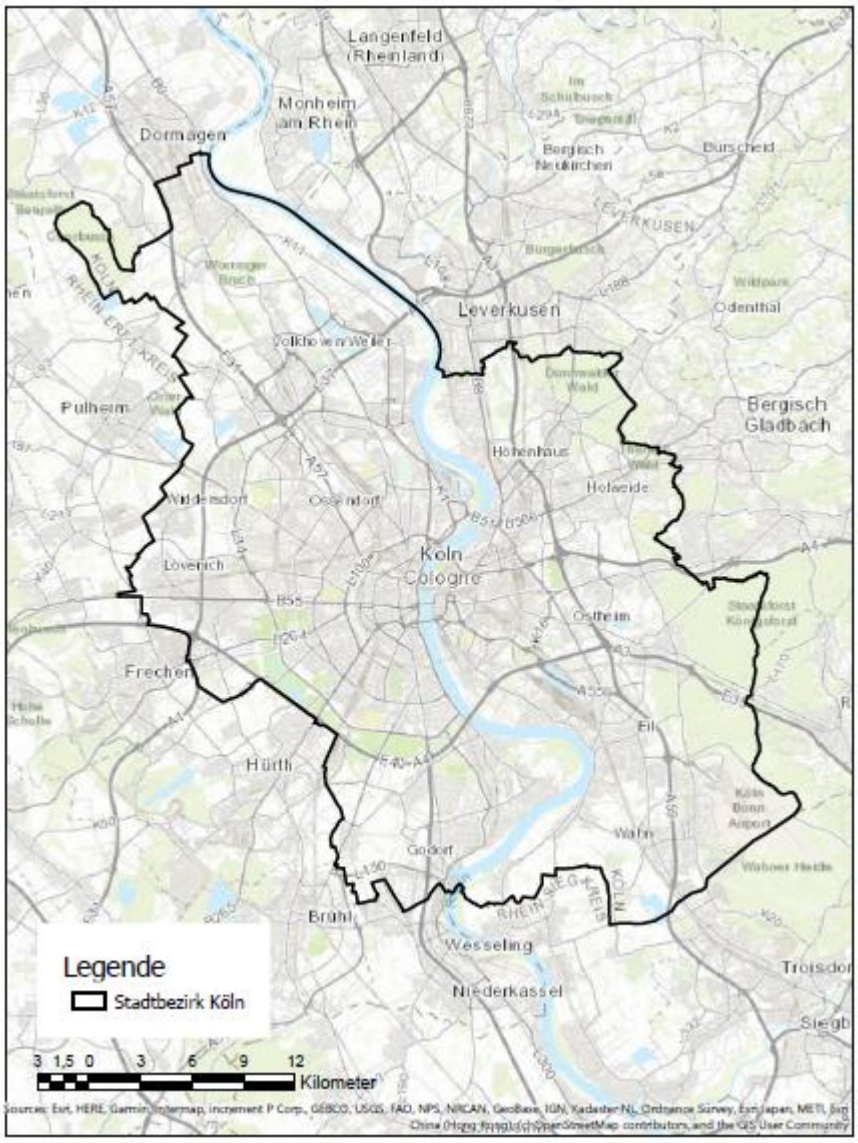


Abbildung 13: Stadtgebietsgrenzen von Köln, eigene Darstellung

Die vor der Weiterverwendung des Layers durchgeführten Operationen begrenzen sich auf die Anpassung der ursprünglichen geographischen Projektion an streckentreue Polylinien und hausgenaue Punktdaten, weil diese an die Stadtgebietsgrenzen Köln's angepasst werden.

2.5 Schritte der Datenvorverarbeitung

Im folgenden Absatz werden die Operationen der Datenvorverarbeitung für den Einsatz im Prozess zur Vorhersage der Kabelnetzverläufe beschrieben.
Die erste Anpassung der hausgenauen Punktdaten und der streckentreuen Polylinien bezieht sich auf die im vorherigen Absatz beschriebene Stadtbezirksgrenzen von Köln.
Die Operation der Verschneidung der Layer auf die Ausdehnung der Vorlage Fläche erfolgt standardgemäß in jeglichen GIS-Software mit dem Werkzeug *clip*. Der Vorgang bedarf einer Eingabe der Ausschneidung-Feature sog. *clip-Feature* anhand dieser die Eingabedaten genau oder auch mit einem angegebenen Abstand der Toleranz in eine neue Datei speichert.
Mit dem Ziel das Untersuchungsgebiet für den praktischen Teil der Abschlussarbeit festzulegen, war im ersten Schritt eine Abgrenzung der hausgenauen Punktdaten notwendig. Die räumliche Verteilung der technischen Identifikationsnummer ergab sich als ein kleinräumiges Klassifizierungskriterium, mit dem die Entscheidung über die Auswahl eines kompakten Untersuchungsgebietes nicht getroffen werden konnte. Infolgedessen wurden die über drei deutsche Bundesländer verteilte hausgenaue Punktdaten und die streckentreue Polylinie des *OpenStreetMaps* Layers auf einen gemeinsamen Nenner gebracht und somit an die Stadtgebietsgrenzen von Köln angepasst.
Die weiteren Operationen und Modifikationen der Trainings- und Testdaten werden folgend erläutert.

2.5.1 Abgrenzung des Testgebietes

Die praktische Anwendung des im Abstaz 2.3 Prozessübersicht dargestellten Prozesses bezieht sich auf die hausgenaue Punktdaten und streckentreue Polylinien innerhalb ausgewählten Testgebietes. Die Auswahl der Datensätze erfolgte in GIS-Software

ArcGISPro der Firma Esri. Aus dem Portfolio der räumlichen Analysen wurden die Werkzeuge: Verschneiden und Zeichnen verwendet.

Das gesuchte Testgebiet erfüllt die im Absatz 1.5 Abgrenzung der Abschlussarbeit beschriebene Kriterien: kompakt, gut abgrenzbar und ausreichend zu sein.
Aufgrund der örtlichen Kenntnis des Autors von der Stadt Köln wurden die Stadtgebietsgrenzen als Ausgangslage festgelegt.
Die erste Stufe der Abgrenzung der Daten war die in der Einleitung des Absatzes 2.5 Schritte der Datenvorverarbeitung beschriebene Verschneidung der hausgenauen Punkte und der streckentreuen Polylinien auf die Stadtgebietsgrenzen der Stadt Köln.
Da die technische Identifikationsnummer sich bei einer Visualisierung als ein kleinräumiges Klassifizierungsmerkmal erwiesen hat, wurden weitere Attribute bei der Vorauswahl berücksichtigt. Die Entscheidung über die Auswahl des Untersuchungsgebietes wurde anhand der Identifikationsnummer des Hauptanschlusskabels getroffen. Die gleichen Bezeichnungen wurden in Gruppen zusammengefasst und visualisiert, so dass jede Kategorie gesondert eingefärbt wurde. Zusätzlich in der Legende wurde die Anzahl der Adressen angegeben. Die Übersicht der Versorgungsgebiete von einem Hauptkabelanschluss stellt einen natürlichen Cluster dar, welche die Grundlage für die Auswahl des Untersuchungsgebietes diente.

Die im Absatz 1.5 Abgrenzung der Abschlussarbeit festgelegten Kriterien, welche das gesuchte Gebiet vorzuweisen hat, werden hier berücksichtigt:

- Kompaktheit
- klare Abgrenzung durch geographische Grenzen
- Anzahl von hausgenauen Punktdaten innerhalb des Gebietes

Die Kompaktheit wird anhand der Homogenität der visualisierten Klassen der Identifikationsnummer des Hauptanschlusskabels festgelegt. Die endgültige Auswahl des

Gebietes wurde durch die Verteilung der Klassen der technischen Identifikationsnummern validiert.

Das zweite Auswahlkriterium bezieht sich nicht auf die Datenstruktur, sondern auf die geographischen Gegebenheiten der Fläche. Gesucht wurde nach natürlich abgegrenzten Räumen durch einen Fluss, eine Grünanlage oder auch eine Eisenbahnlinie.

Daher, dass mehr als nur eine Fläche die ersten zwei Kriterien erfüllte, wurde zusätzlich die Anzahl der hausgenauen Punktdaten als ein Entscheidungskriterium festgelegt. Im Betracht dessen, dass die Vorhersage der technischen ID ausschließlich für die Daten innerhalb des Untersuchungsgebietes getroffen wird, hat die Anzahl der Punktdaten einen hohen Stellenwert bekommen.

Das Untersuchungsgebiet liegt in dem Kölner Stadtteil Buchforst und erfüllt alle diese Kriterien.

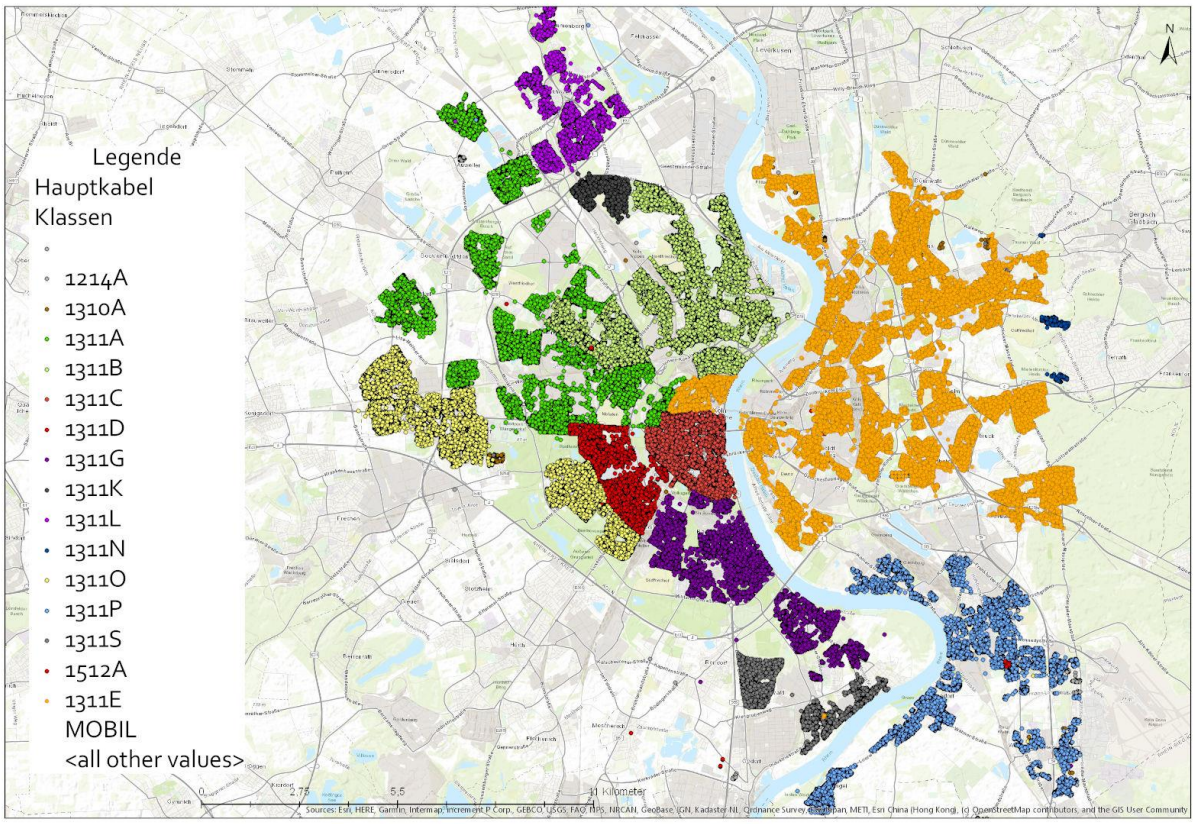


Abbildung 14: Aufteilung des Hauptanschlusskabels in Köln, eigene Darstellung

Die Abgrenzung des Gebietes wurde manuell in der GIS-Software gezeichnet und als Polygon gespeichert. Die Fläche des Testgebietes im Vergleich zu der Ausdehnung des Stadtbezirks Köln beansprucht ca. 1,33 % und ist in der Abbildung 15: Testgebiet im Stadtbezirk Köln visualisiert.

Die finale räumliche Anpassung der hausgenauen Punktdaten und der streckentreuen Polylinien wurde analog zu der bereits beschriebenen Verschneidung der Daten auf die Stadtgebietsgrenzen der Stadt Köln durchgeführt (siehe dazu 2.5 Schritte der Datenvorverarbeitung) durchgeführt.

Die nummerische Zuordnung der technischen ID für jede postalische Adresse erfolgt dort, wo ein Anschluss vorhanden ist. Die Vergabe der Identifikationsnummern erfolgt potentiell nicht fortlaufend, sondern wird von dem jeweiligen Verzweigerkabelsegment (siehe Exkurs) an dem das Objekt angeschlossen ist übernommen. Durch das Sortieren der hausgenauen Punktdaten nach der technischen ID bilden sich zusammenhängende Klassen. Die Cluster der Datensätze wurden für diese Abschlussarbeit übernommen und in der Abbildung 11: Beispiel der Verteilung der hausgenaue Punktdaten nach TECH ID, eigene Darstellung exemplarisch aufgezeigt. In der Darstellung ist erkennbar, dass die gebildeten Gruppen keinen linearen Zusammenhang haben und nicht immer klar abgrenzbar sind. Auffallend sind innerhalb einer Gruppe einzelne Werte, welche nicht über die gleiche technische ID verfügen. Solche Konstellationen werden als Ausreißer bezeichnet. Die Gründe hierfür können unterschiedlich sein, aus Erfahrungsberichten sind solche Aufteilungen entweder ein Datenfehler oder bilden eine besondere Konstellation einer nachträglichen Anbindung ab.

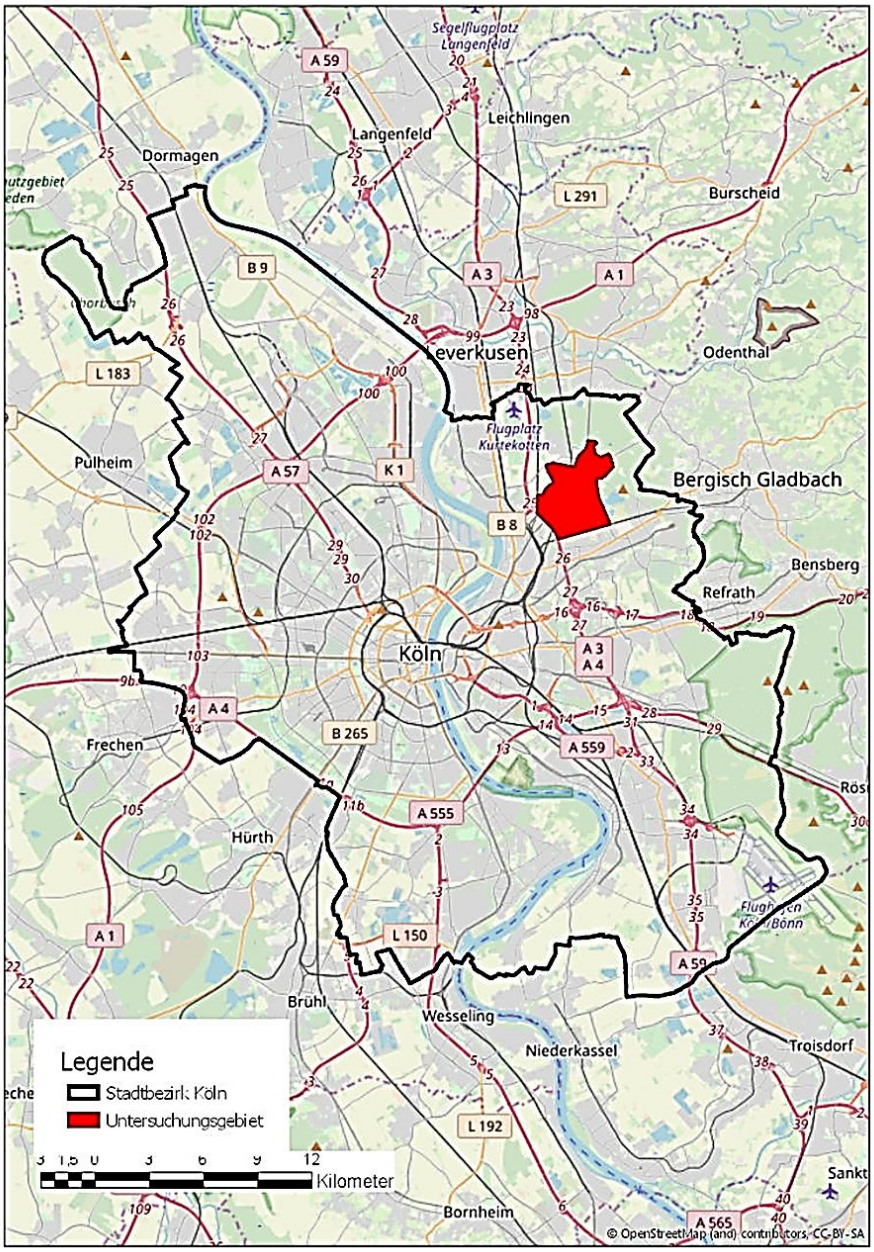


Abbildung 15: Testgebiet im Stadtbezirk Köln, eigene Darstellung

2.5.2 Nachbilden der OpenStreetMaps Straßenzüge

Die Annahme, dass das Kabelnetz entlang des Straßennetzes verläuft, wurde auch im Rahmen des generischen Modellierungsansatzes für die Kosten des Teilnehmeranschlussnetzes getroffen (siehe dazu Exkurs). Für die Anwendung des analytischen Kostenmodells wird ebenso der *OpenStreetMap* Straßenlayer eingesetzt, um eine realitätsnahe Berechnung durchzuführen. (Kulenkampff, 2019)
Da dieses Projekt von der Bundesnetzagentur beauftragt worden ist, wurden die Ergebnisse veröffentlicht und es wurde um Stellungnahmen der Teilnehmer gebeten. Zu den Teilnehmern gehörten deutsche Telekommunikationsunternehmen, wie die Telekom oder Vodafone. Aus der Rückmeldung der Teilnehmer ist ein wichtiger Aspekt anzumerken, welcher im Rahmen der praktischen Anwendung der Abschlussarbeit auch aufgenommen und angewendet wurde. Die Aussage lautet, dass die Kabelführung beidseitig der Straße verläuft, deswegen soll eine beidseitige Straßenführung betrachtet werden. Das analytische Kostenmodell wurde daher um eine doppelseitige Straßenführung in linke und rechte Seite ergänzt.

Analog dazu wurde der *OpenStreetMap* Straßenlayer, um nicht erwünschte Straßenelemente bereinigt und in streckentreue Polylinien umgewandelt. Die Annahme lautet, dass das Kabelnetzwerk entlang der Straßenkanten verläuft, daher sind beide Straßenseiten links und rechts der Fahrbahn zu berücksichtigen, um eine gesonderte Anschlussstrecke für jeweiligen Gebäuden, die an der der Straßenkante liegen abzubilden. (Kulenkampff, 2019) Zur Erzeugung der doppelseitigen Straßenführung wurde der *OpenStreetMap* Layer kopiert und rechts von der Polylinie um 3 m von der ursprünglichen Lage dupliziert. Die Größe des Versatzes wurde geschätzt und soll die durchschnittliche Breite einer Anliegerstraße entsprechen. Zur Unterscheidung der Straßenzüge in linke und rechte Straßenseite wurde die Vektor-Richtung des Original Datensatzes als Grundlage genommen. Diese Information wurde in einem neuen Feld als ein zusätzliches Attribut

gespeichert. Zur Berücksichtigung der Differenzierung der Straßenseite im Vorhersagemodell wurde die eindeutige osm_id zu einer neuen Identifikationsnummer (OSM_ID_Neu) zusammengefasst, indem die bestehende ID mit der Zahl 100 multipliziert wurde.

Die Straßenquerungen wurden aus dem Schritt der Daten-Bereinigung ausgenommen. Die, bei der Duplizierung der Polylinie entstandene Straßenquerungen bzw. die sich kreuzende Linien wurden nicht bereinigt, sondern weiterverwendet. Solche GIS-Operation hat als Ziel eine ansprechbare Visualisierung aufzubereiten, meistens in Form von einer Übersichtkarte. Die Qualität der Ergebnisse des Vorhersagemodells wird nicht durch das Entfernen von sich überlappenden Straßenquerungen verbessert.

Die Aufstellung aus Tabelle 5: Auswahl der OpenStreetMap Straßenarten, zeigt die Beschreibung und Begründung der Auswahl unterschiedlichen Klassen des *OpenStreetMap* Straßenlayers auf. Zusätzlich ist die Statistik des Vorkommens von den unterschiedlichen Straßenarten im Untersuchungsgebiet in die Tabelle integriert.

Anhand Abbildung 16: OpenStreetMap Straßenlayer vor und nach der Auswahl ist zu beobachten, dass sich bestimmte Straßenarten decken. Eine beispielhafte Situation kommt vor, wenn entlang einer lokalen Straßen in einem Ortsteil ein Fahrrad und/oder Fußgängerweg verläuft. Diese Wege sind dann einer eigenen Kategorie zugeordnet: livin_street, cycleway und footway und als eine gesonderte Linie dargestellt.

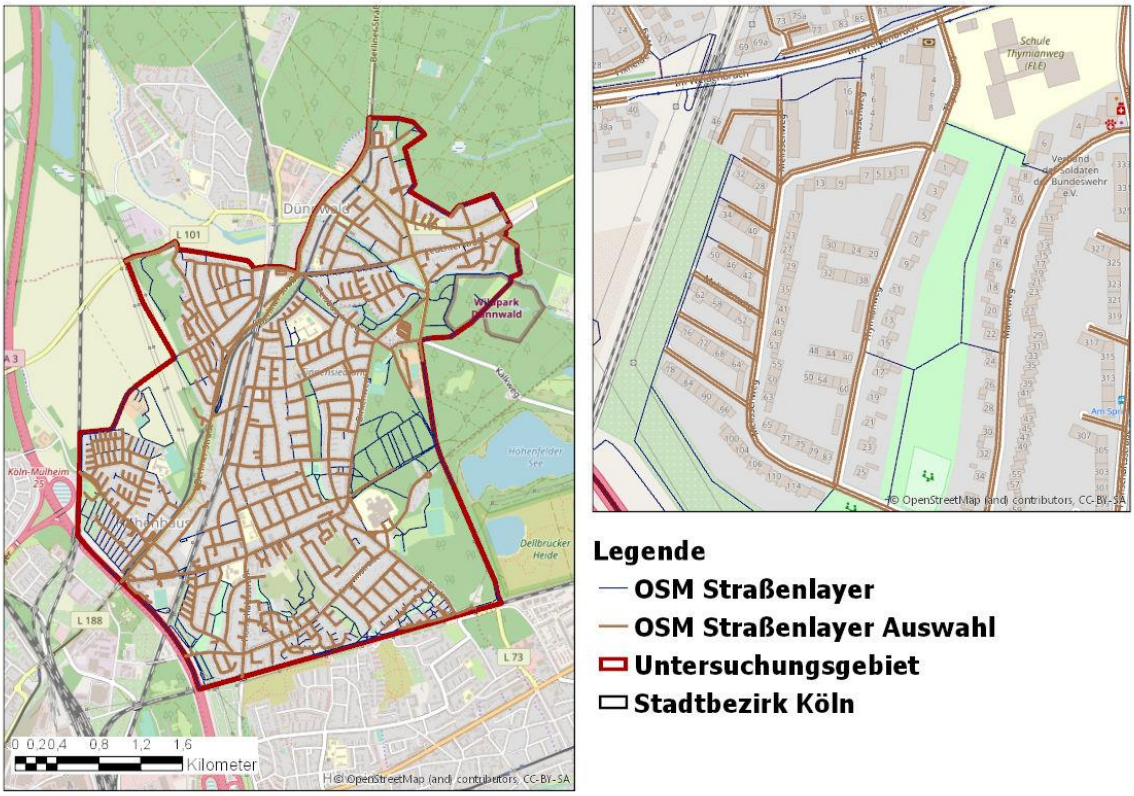


Abbildung 16: OpenStreetMap Straßenlayer vor und nach der Auswahl, eigene Darstellung

Wie beschrieben ist die Annahme, dass die Kabelnetzinfrastruktur entlang der primären Straßen verläuft, daher ist es notwendig bestimmte Straßenarten wie z.B. Fußweg, Feldwege, etc. aus dem Ursprungslayer zu entfernen.

Im ersten Schritt wurden die Brücken und Tunnels anhand des Attributes *bridge*= T und *tunnel*=T rausgefiltert. Die zweite Stufe der Bereinigung der Daten erfolgte manuell und bedarf einer Entscheidung aufgrund der Nähe zum hausgenauen Punkt bzw. Trainingsdatensatz. Die Entscheidungsgrundlage bildete eine Statistik der Anzahl der Straßen-Klassen im Untersuchungsgebiet (siehe Spalte Anzahl vor Auswahl in Tabelle 5: Auswahl der OpenStreetMap Straßenarten, eigene Darstellung).

Die führende Straßenarten aus der Klasse: *secondary, tertiary, residential, living_street, service* und *bridleway* wurden ohne Veränderungen beibehalten. Diese Straßen sind als Anschlusswege zu einzelnen Objekten zu sehen und werden als führende Straßen

bezeichnet. Im gesamten Untersuchungsgebiet kommt eine Ausnahme vor, wo der Anschluss eines abseits liegenden Objekts über einen zweitrangigen Weg erfolgt (Ausnahme: Hülsenweg). Wege, welche parallel zu den führenden Straßen verlaufen, wurden nicht berücksichtigt. Hierzu gehören folgenden Klassen: *track_grade1, track_grade3, track_grade4, track_grade5, cycleway*, welche Motorradwege oder auch Wege, die asphaltiert oder auch nur befestigt präsentieren. Weitere Wege, wie Landwirtschaftswege oder auch kleine lokale Wege der Klasse: *unclassified* und *track* wurden nur deswegen rausgenommen, weil diese nicht an hausgenauen Punkten angrenzen und somit für die Vorhersage keine Relevanz haben. Insgesamt sind nach der Bereinigung des *OpenStreetMap* Straßenlayers 60 % der Straßensegmente verblieben.

Beschreibung OpenStreetMap Straßenlayer				**Vorgenommene Änderungen am OpenStreetMap Straßenlayer**		
code	**layer**	**fclass**	**Description**	**Anzahl der Straßen im Unteruchungs-gebiet**		**Begründung**
511x	**roads**		**Major roads**	**vor der Auswahl**	**nach der Auswahl**	
5111	roads	motorway	Motorway/freeway	6	0	grenzt nicht an Anschlusspunkten an
5112	roads	trunk	Important roads, typically divided	0	0	kommt nicht vor
5113	roads	primary	Primary roads, typically national.	0	0	kommt nicht vor
5114	roads	secondary	Secondary roads, typically regional.	71	71	führende Straße, bleibt unverändert
5115	roads	tertiary	Tertiary roads, typically local.	76	76	führende Straße, bleibt unverändert
512x	**roads**	**Minor**	**Roads**			
5121	roads	unclassified	Smaller local roads	2	0	grenzt nicht an Anschlusspunkten an

5122	roads	residential	Roads in residential areas	416	416	führende Straße, bleibt unverändert
5123	roads	living_street	Streets where pedestrians have priority over cars	40	40	führende Straße, bleibt unverändert
5124	roads	pedestrian	Pedestrian only streets	0	0	kommt nicht vor
513x	**roads**		**Highway links (sliproads/ramps)**			
5131	roads	motorway_ link	Roads that connect from one road to another of the same of lower category.	0	0	kommt nicht vor
5132	roads	trunk_link		0	0	kommt nicht vor
5133	roads	primary_link		0	0	kommt nicht vor
5134	roads	secondary_ link		0	0	kommt nicht vor
514x	**roads**		**Very small roads**			
5141	roads	service	Service roads for access to buildings, parking lots, etc.	396	396	führende Straße, bleibt unverändert
5142	roads	track	For agricultural use, in forests, etc. Often gravel roads.	2	0	grenzt nicht an Anschlusspunkten an
5143	roads	track_grade 1	Tracks can be assigned a “tracktype” from 1 (asphalt or heavily compacted) to 5 (hardly visible). A detailed description is here: http://wiki.openstreetmap.org/wiki/Tracktype	6	0	verläuft parallel zu führenden Straße
5144	roads	track_grade 2		24	2	Ausnahme: Hülsenweg grenzt an Anschlusspunkt an
5145	roads	track_grade 3		21	0	verläuft parallel zu führenden Straße
5146	roads	track_grade 4		4	0	verläuft parallel zu

						führenden Straße
5147	roads	track_grade 5		6	0	verläuft parallel zu führenden Straße
515x	**roads**		**Paths unsuitable for cars**			
5151	roads	bridleway	Paths for horse riding	2	2	führende Straße, bleibt unverändert
5152	roads	cycleway	Paths for cycling	88	0	verläuft parallel zu führenden Straße
5153	roads	footway	Footpaths	422	0	
5154	roads	path	Unspecified paths	70	0	
5155	roads	steps	Flights of steps on footpaths	20	0	
516x	**roads**		**Ferries**			
5160	roads	ferry	Ferry routes	0	0	kommt nicht vor
			Unknown			
5199	roads	unknown	Unknown type of road or path	0	0	kommt nicht vor
			Summe:	1672	1003	

Tabelle 5: Auswahl der OpenStreetMap Straßenarten, eigene Darstellung

Ergänzend zu der tabellarischen Aufstellung der Auswahl der *OpenStreetMap* Straßenarten visualisiert die Abbildung 16: OpenStreetMap Straßenlayer vor und nach der Auswahl das Ergebnis.

2.5.3 Extrahieren von Punktdaten

Die letzte Operation, welche außerhalb des Python-Skriptes in der Prozessübersicht im Absatz 2.3 beschrieben ist, ist das Extrahieren von Punkten je zehn Meter Abstand aus den streckentreuen Polylinien. Das Umwandeln der Polylinien in Punkte hatte als Ziel den

Einsatz des k-*Nearest Neighbor* Algorithmus in der Python Sprache aus der Bibliothek scikit-learn bekannt als *KNeighborsClassifier*, welcher nur mit Punktdaten umgehen kann. Dafür sind die Polylinien in mehrere Punkte zerlegt worden.
Dieser Vorgang wurde mit der frei Verfügbaren (*Open Source*) GIS-Software Q-GIS durchgeführt. Mit dem Werkzeug „Punkte entlang von Linien erzeugen“ wurden die streckentreue Polylinien innerhalb eines Datensatzes, d.h. einer bekannten Straße erzeugt. Die Start- und Endpunkte der Polylinien wurden beibehalten. Das Setzten von zusätzlichen Punkten auf der Polylinie fängt mit dem Start-Punkt an und endet am Endpunkt, so dass es möglich ist die ursprüngliche Länge der original Straßensegmente festzustellen. Auf der gesamten Länge zwischen den Start- und End-Punkt wurden je zehn Meter auf der Polylinie zusätzliche Punkte gesetzt. Die durchschnittliche Länge der streckentreuen Polylinien beträgt 132 m. Nach der Auswahl verblieben 1003 Polylinien aus welchen insgesamt 19521 Punkte extrahiert wurden. Die Auswahl des Abstands von 10 m ist aus der Überlegung entstanden, dass eine Gebäudekante unterschiedlich lang sein kann. Seltener beträgt die Länge eine vordere Hauswand zu Straße hin weniger als 10 m.

2.5.4 Umgang mit unvollständigen Datensätzen

Der Umgang mit unvollständigen Daten ist stark mit dem Thema Datenvorverarbeitung verbunden. Im Bereich von *machine learning* ist die Datenqualität der Trainingsdaten mitentscheidend für die Qualität der Vorhersage. Die Aussage der einzelnen Merkmale soll möglichst informativ und aussagekräftig sein, um die Prognose der realen Situation bestmöglich wiederzugeben.

Das Problem der Über- und Unteranpassung (*over- underfitting*) tritt auf, wenn die trainierte Vorhersage nicht generalisierbar ist und auf unbekannten Daten (Testdaten) ausgeführt nicht gelingt. Die Überanpassung ist meistens der zu hohen Anzahl an Parametern geschuldet. Demgegenüber kommt die Unteranpassung vor, wenn das Modell

nicht komplex genug ist und zu einer bestimmten Entscheidung tendiert. (Raschka, 2017, p. 81). Das Thema ist sehr wichtig bei einem Training von einem allgemeingültigen Vorhersagemodell mit dem Ziel eine Prognose auf völlig neuen und unbekannten Daten abzugeben.

Bei der in dieser Abschlussarbeit entwickelten Vorgehensweise zur Nachbildung von physischen Kabelnetzverläufen werden Daten bei der die TECH_ID nicht vorliegt (sog. NULL Werte) eleminiert, da diese keine Information für das Modell enthält und dieses potentiell eher schlechter macht. Die NULL Werte in den Eingabedaten stellen Objekte dar, wo kein Unitymedia Kabelanschluss vorhanden ist. Das bedeutet, dass diese Objekte keine eigene Kabelnetzwerk Klasse bilden, weil diese nicht systematisch eingeordnet sind. Aus geographischer Sicht ist die Lage der NULL Werte zwischen anderen Werten einer TECH_ID oder entlang eines nicht versorgten Straßenzuges. Wie in Abbildung 11: Beispiel der Verteilung der hausgenaue Punktdaten nach TECH_ID, eigene Darstellung sichtbar ist, sind NULL Werte direkte Nachbar von der TECH_ID = 11060360. Bei der Suche nach dem nächsten Nachbarn für den Wert TECH_ID = 11060360 würde der benachbarte NULL Wert miteinbezogen und je nach Höhe des k-Wertes die Vorhersage für den Wert NULL abgeben. Das Ziel der Abschlussarbeit ist die physischen Kabelnetzverläufe abzubilden, die Kabelstränge mit dem Wert NULL sind allerdings nicht existent und werden daher für die Vorhersage nicht miteinbezogen.

3 Ergebnisse

Das folgende Kapitel stellt die Ergebnisse der durchgeführten Durchläufe des Modells dar und gibt Antworten auf die in Kapitel 1.4 Forschungsfrage definierten Fragestellungen. Zusätzlich werden die Ergebnisse fachlich bewertet und auf ggf. relevante Erkenntnisse hingewiesen.

3.1 Bewertung des Modells

Der Aufruf des Modells wurde mit unterschiedlichen Parameterkombinationen durchgeführt, die folgenden Parameter wurden verändert:

- Der k-Distanz Wert für die Auswahl der relevanten Linien wurde auf 1, 3, 5, 7 und 9 festgelegt.
- Der k-Wert des k-*Nearest Neighbor* Algorithmus wurde auf 1, 3, 5, 7 und 9 festgelegt.
- Die Distanzmetrik wurde auf euklidische und Manhattan Distanz festgelegt.
- Es wurde eine Vorsortierung der zu betrachtenden Punkte nach TECH_ID vorgenommen oder nicht vorgenommen.

Somit ergeben sich 5x5x2x2 = 100 unterschiedliche Parameterkombinationen, die berechnet wurden und deren Ergebnisse in den folgenden Abschnitten anhand von vorzeige Beispielen untersucht werden.

3.1.1 Bewertung der Modellgüte als Bewertungskriterium

Die Modellgüte (Definition siehe 2.2.4) ist ein statistisches Maß für die Bewertung der Vorhersagekraft des Modells. Sie dient einerseits dazu das Modell alleinig zu bewerten, aber auch die unterschiedlichen Parameterkombinationen der Modelle miteinander zu vergleichen und deren Einfluss auf die Modelle auszudrücken. Trotzdem kann die

Modellgüte ggf. nicht alleinig als Bewertungskriterium für die fachliche Auswahl eines Modells herangezogen werden. Dies ist unter anderem auch von dem Ziel mit dem das Modell eingesetzt wird abhängig: Vorhersage von Kabelverläufen oder Erkennung von Anomalien/Datenqualitätsproblemen (siehe dazu auch Kapitel 5 Schlussfolgerungen und Ausblick).

3.1.2 Einfluss des k-Wertes auf die Modellgüte

Beim Vergleich der Modelle anhand ihrer Modellgüte stellt sich insbesondere die Frage, welchen Einfluss der k-Parameter hat. Da dieser Parameter die Anzahl der betrachteten k nächsten Nachbarn bestimmt und somit für die Zuordnung einer TECH_ID zu einem zu bewertenden Punkt entscheidend ist.
Der Parameterwert k=1 spielt in diesem Zusammenhang eine besondere Rolle, da dieser einer Zuordnung anhand des distanznächsten Punktes durchführt und damit einem klassischen Ansatz entspricht.
Dies hat den Nachteil, dass man keine Aussage über die nächsten Nachbarn des zu bewertenden Punktes bekommt und somit einen Vorteil des Verfahrens, nämlich die Plausibilisierung und die Entdeckung von Auffälligkeiten verliert.

Die folgende Abbildung zeigt die Modellgüte, des besten Modells je k-Wert über alle Parameterkombinationen an:

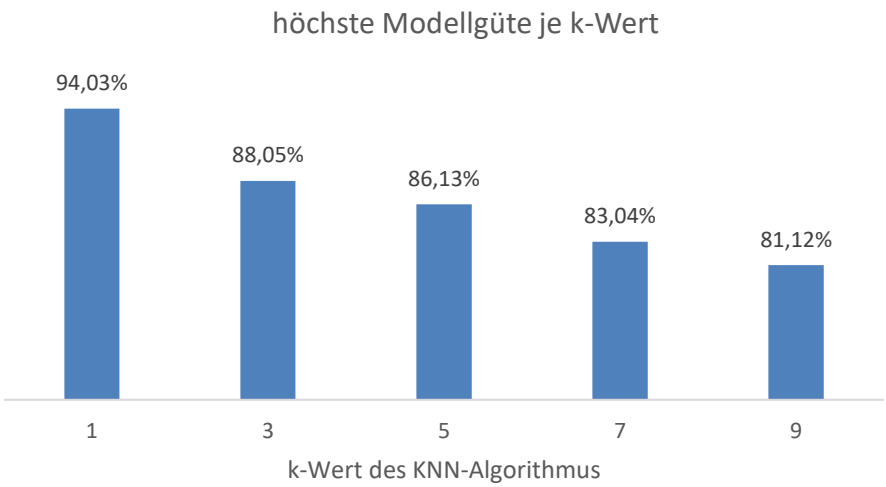


Abbildung 17: höchste Modellgüte je k-Wert des k-Nearest Neighbor Algorithmus, eigene Darstellung

Abbildung 17: höchste Modellgüte je k-Wert des k-Nearest Neighbor Algorithmus zeigt, dass das beste Modell eine Modellgüte von 94,03% bei einem k-Wert von 1 hat. Dies ist insofern nicht verwunderlich, da eine minimale räumliche Distanz zwischen hausgenauem Punkt und dem Kabelnetz optimal und sinnvoll erscheint.

Es zeigt sich damit also, dass für eine Vorhersage von einem Punkt zu einem Kabelnetz die minimale Distanz ein sehr sinnvoller Prädiktor ist.

Gleichzeitig zeigt es aber auch, dass die minimale Distanz nicht immer der richtige Prädiktor ist, da ansonsten eine Modellgüte von 100% erreicht werden würde. Auch dies erscheint sinnvoll, da es sich bei den Daten um Echtdaten handelt, kann es einerseits sein, dass ein Fehler in den Daten unterlaufen ist oder zu diesem Zeitpunkt des Ausbaus an dieser Stelle kein Objekt bzw. Haus vorhanden war und nur durch eine nachträgliche Baumaßnahme angeschlossen werden könnte. Die Herkunft der Daten spielt auch eine wichtige Rolle. Daher, dass in dieser Abschlussarbeit verwendete Daten nur einem Telekommunikationsanbieter repräsentieren, ist davon auszugehen, dass nicht alle Objekte flächendeckend von derselben Kabelnetzanbieter bedient werden. Somit ist die

Information kein Anschluss vorhanden mit Vorsicht zu genießen. (siehe dazu auch 2.5.4 Umgang mit unvollständigen Datensätzen).

Daher sind auch die Modelle mit k-Werten größer als 1 relevant, da diese einen Hinweis auf potentielle Datenfehler oder andere besondere Fallkonstellationen geben können (siehe dazu Absatz 3.2 Fachliche Einschätzung der räumlichen Verbindung). Im konkreten Anwendungsfall zur Ableitung der Kabelnetzverläufe zeigt sich, dass je größer der k-Wert, desto schlechter ist die Modellgüte. Für einen Anwendungsfall zur Datenqualitätskontrolle und zur Anomalien-Entdeckung würden sich die Modelle mit k=3 oder k=5 anbieten, da diese einen detaillierteren Blick auf die nächsten Nachbarn bei einer relativ hohen Modellgüte ermöglichen.

3.1.3 Bewertung der eingesetzten Parameter

Wie in Kapitel 3.1.2 Einfluss des k-Wertes auf die Modellgüte beschrieben nimmt der k-Wert des k-*Nearest Neighbor Classifier* eine besondere Rolle ein, aber auch die anderen Parameter können einen Einfluss auf die Modellgüte haben. Im folgenden Abschnitt sollen die Parameter k-Wert für die Auswahl der relevanten Linien, der Parameter der Distanzmetrik und die Vorsortierung genauer untersucht werden, ob und welchen Einfluss diese Parameter auf die Modellgüte haben.

3.1.3.1 Einfluss des k-Distanz Wertes auf das Modell

Der Einfluss des k-Distanz Wertes für die Auswahl der relevanten Linien auf die Modellgüte wird in Abbildung 18: Einfluss des k-Distanz Wertes auf die Modellgüte in Abhängigkeit des k-Wertes des KNN Algorithmus untersucht. Die Abbildung zeigt die maximale Modellgüte je k-Distanz Wert in Abhängigkeit des k-Wertes des k-*Nearest Neighbor* Algorithmus. Es zeigt sich, dass der k-Distanz Wert keinen Einfluss auf die

Modellgüte hat, da für jeden k-Distanz Wert die Güte für jeden k-Wert des k-*Nearest Neighbor* Algorithmus gleichbleibend ist.

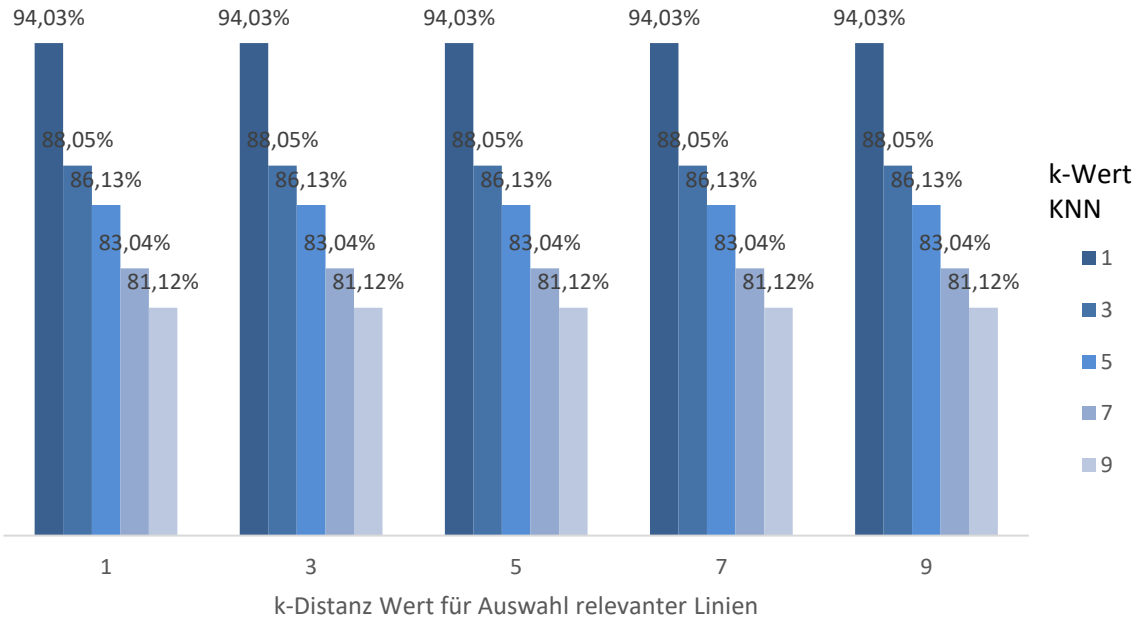


Abbildung 18: Einfluss des k-Distanz Wertes auf die Modellgüte in Abhängigkeit des k-Wertes des KNN Algorithmus, eigene Darstellung

Diese Erkenntnis ist auch valide, da der k-Distanz Wert nicht für das Training des Modells genutzt wird, sondern nur die Auswahl der relevanten Linien bei Nutzung des Modells auf unbekannten Daten beeinflusst, um die Anzahl der betrachteten Punkte möglichst gering zu halten und somit die Performanz des Modells zu erhöhen.

Betrachtet man nun inwieweit der k-Distanz Wert die Menge aller zu betrachtenden Linien einschränkt, so zeigt sich, dass je größer der k-Distanz Wert, je mehr relevanten Linien und somit je mehr extrahierten Punkte müssen berücksichtigt werden.

Diese Erkenntnis ist trivial, da der k-Distanz Wert gerade die Menge der zu betrachtenden Linien bestimmt. Relevant ist in diesem Zusammenhang, inwieweit schränkt der k-Distanz

Wert die Anzahl der relevanten Linien ein und wie groß ist der Anteil der relevanten Linien bei einem bestimmten k-Distanz Wert im Vergleich zu der Gesamtmenge aller Linien.

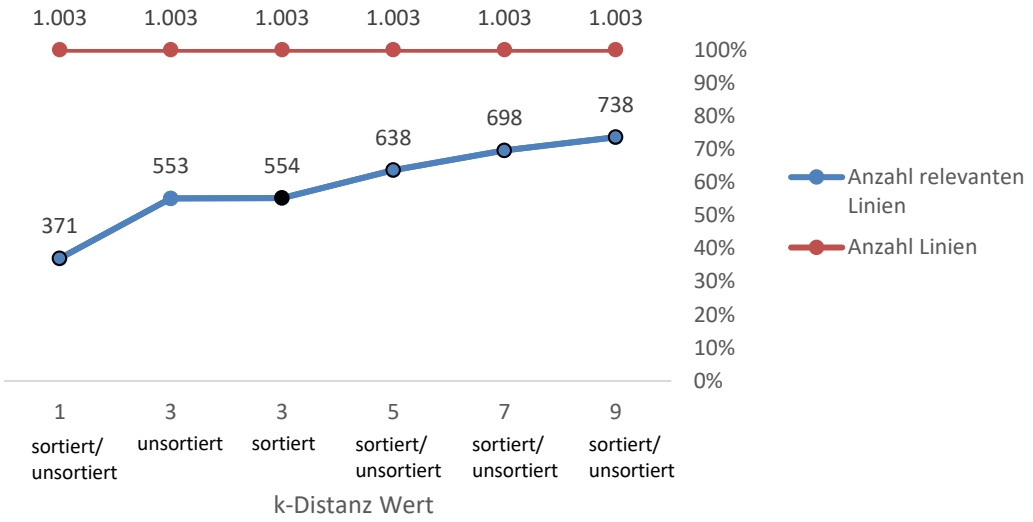


Abbildung 19: Einfluss des k-Distanz Wertes auf die Anzahl Linien, eigene Darstellung

Betrachtet man Abbildung 19: Einfluss des k-Distanz Wertes auf die Anzahl Linien so sieht man, dass die Anzahl der relevanten Linien bei einem k-Distanz Wert von 1 bei 371 liegt. Im Verhältnis zu der Gesamtanzahl von Linien in Höhe von 1.003 sind dies ca. 37%, d.h. man hat die Menge der zu betrachtenden Linien um ca. 63% gesenkt. Für den k-Distanz Wert drei spielt auch die Vorsortierung eine marginale Rolle. Daher ist für diesen k-Distanz Wert der Wert der unsortierten (blau) und sortierten Anzahl (schwarz) angegeben, bei den k-Distanz Werten 1, 5, 7 und 9 sind die Werte identisch (blau mit schwarzem Rand), daher wird dort keine Unterscheidung visualisiert.

Für die benötigte Rechenleistung des Modells ist aber nicht die Anzahl der Linien relevant, sondern die Anzahl der zu betrachtenden Punkte, zu denen die Entfernung berechnet werden muss, um die k-nächsten Nachbarn zu identifizieren. Daher ist in Abbildung 20: Einfluss des k-Distanz Wertes auf die Anzahl Punkte die Anzahl der relevanten Punkte, die

sich aus den relevanten Linien ergibt und die Anzahl aller extrahierter Punkte, die sich aus allen Linien ergeben in gleicher Weise dargestellt.

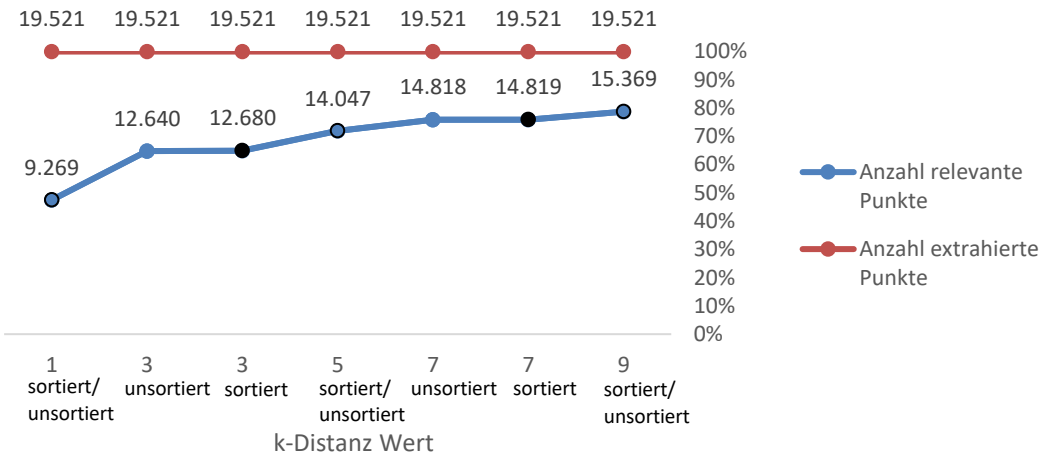


Abbildung 20: Einfluss des k-Distanz Wertes auf die Anzahl Punkte, eigene Darstellung

Es zeigt sich, dass die Auswirkungen auf Punktebene noch größer sind als auf Linienebene: Es müssen bei k-Distanz=1 nur 9.269 von 19.521 Punkten berücksichtigt werden, d.h. ca. 47% aller Punkte oder anders ausgedrückt: mehr als die Hälfte aller Punkte sind nicht mehr relevant.
Steigert man den k-Distanz Wert auf 9, so hat man immer noch eine Ersparnis von ca. 21% der Punkte, die man nicht berücksichtigen muss (1-(15.369/19.521)).
Die Suche nach relevanten Linien und damit relevanten Punkten beschleunigt also die Berechnungen sehr stark und stellen eine gute Performanz-Optimierung dar.

3.1.3.2 Einfluss der Distanzmetrik auf das Modell

Die Distanzmetrik ist für die Distanzberechnung der Punkte untereinander zuständig. Die in Abschnitt 2.2.3 Metrische Distanzen definierten Distanzmaße werden untersucht, ob sich eine davon als dominant in allen Parameterkombinationen erweist d.h.

man erreicht mit diesem Distanzmaß immer ein besseres Ergebnis bzgl. der Modellgüte als mit dem anderen Distanzmaß.

Abbildung 21: Einfluss der Distanzmetrik auf die Modellgüte zeigt die höchste Modellgüte je Distanzmaß und je k-Wert des k-*Nearest Neighbor* Algorithmus. Es zeigt sich, dass sowohl die euklidische, als auch die Manhattan Distanz je nach k-Wert ein besseres Ergebnis erzielen (k =1, 3 und 9). Außerdem gibt es auch Fälle indem die Distanzmaße gleichgute Ergebnisse erzielen (k = 5 und 7).

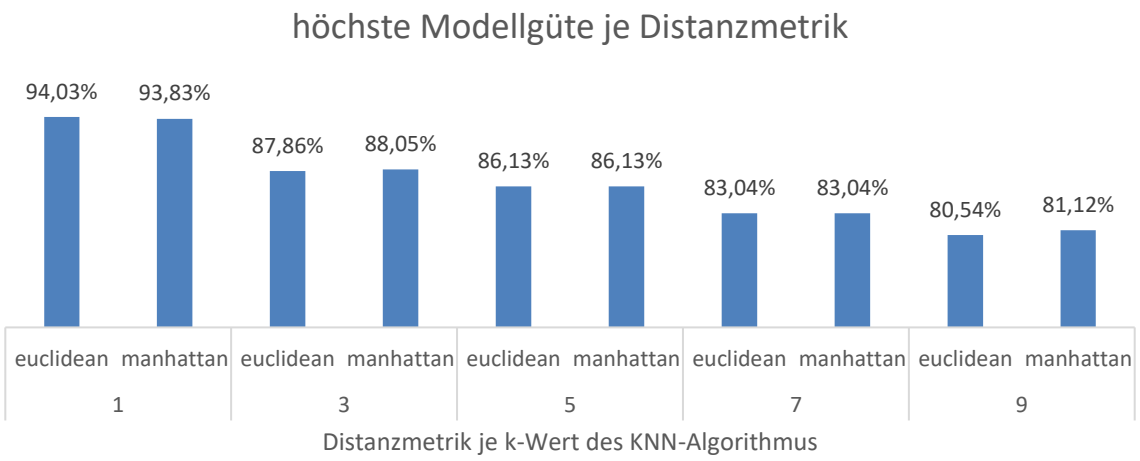


Abbildung 21: Einfluss der Distanzmetrik auf die Modellgüte, eigene Darstellung

Es gibt also kein dominierendes Distanzmaß, somit sollten bei dem Training des Modells immer beide Distanzmaße betrachtet werden, um ein möglichst gutes Modell zu erstellen.

3.1.3.3 Einfluss der Vorsortierung auf das Modell

Wie im Absatz 2.1.1 Linear Ordering beschrieben kann die Sortierung einen Einfluss auf die Ergebnisse eines *spatial joins* und somit auch auf die ausgewählten Objekte für den k-*Nearest Neighbor* Algorithmus haben. Hier soll untersucht werden, inwieweit die

Vorsortierung der gelabelten Trainings- und Testdaten nach der technischen Identifikationsnummer (TECH_ID) einen Einfluss auf die Modellgüte hat.

Wie aus der Abbildung 22: Höchste Modellgüte bei Sortierung ersichtlich führt eine Sortierung der Modelldaten bis auf k=1 des k-*Nearest Neighbor* Algorithmus immer zu besseren Ergebnissen als bei unsortierten Daten. Bei k=1 ist die Vermutung, dass die Eingabedaten, obwohl sie nicht nach der TECH_ID sortiert vorliegen, einer nicht direkt ersichtlichen fachlichen Sortierung unterliegen (z.B. nach Erfassungszeitpunkt der Daten) die einer gewissen Reihenfolge folgt. Somit würde sich auch die bessere Modellgüte bei unsortierten Daten und k=1 erklären, die sich bei Berücksichtigung nur eines Nachbarn stark auswirkt. Die genaue Untersuchung der Auffälligkeit wird hier aber nicht durchgeführt, da es sich um eine anwendungsfallspezifische Untersuchung handeln würde und der Einfluss einer Sortierung auf die Modellgüte in den Fällen k=3, 5, 7 und 9 hinreichend gezeigt werden konnte.

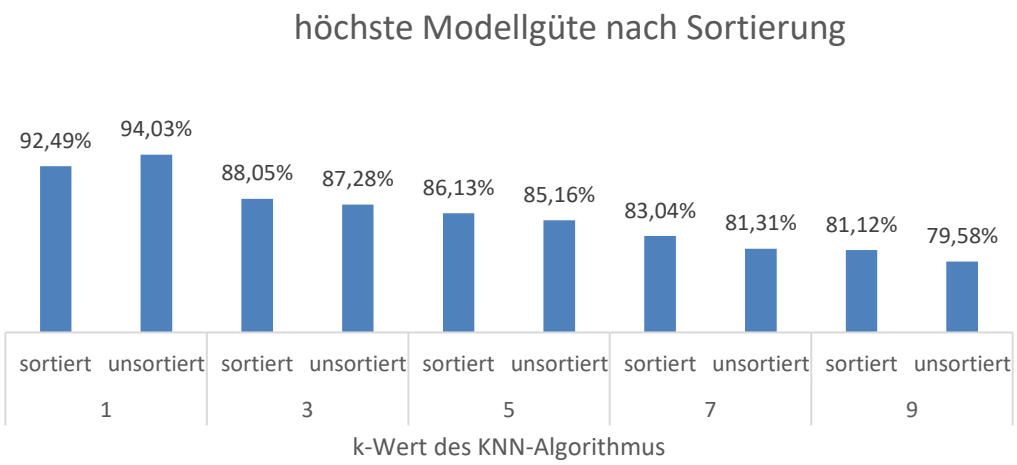


Abbildung 22: Höchste Modellgüte bei Sortierung, eigene Darstellung

3.2 Fachliche Einschätzung der räumlichen Verbindung

Neben der in Kapitel 3.1 Bewertung des Modells durchgeführte statistischen Bewertung des Modells ist ein Vorteil des Verfahrens, dass es die Möglichkeit bietet, die Ergebnisse der Vorhersage für die Plausibilisierung von gelabelten Daten zu verwenden. Dies ist insbesondere daher relevant, da davon auszugehen ist, dass ‚wie im Absatz 1.3 Zielsetzung und Relevanz beschrieben, bei der Digitalisierung der Papier-Dokumentation des Kabelnetzes zu einer fehlerhaften oder auch unvollständigen Erfassung kommen konnte.

Die Ergebnisse der Vorhersage werden automatisch in für jeden Modelldurchlauf einheitlichen Form in einem gesonderten Speicherordner als Excel-Datei abgelegt. Diese Form der Ausgabe soll eine weitere Verwendung der Daten unterstützen. Die Tabelle 6: Struktur der Tabelle mit Zwischenergebnissen der Vorhersage klärt über die Bedeutung der Spalten auf. Die fachlichen Informationen sind die OSM_ID_NEW, welche in dem Schritt der Erzeugung von streckentreuen Polylinien entstanden ist (siehe dazu 2.5.2 Nachbilden der OpenStreetMaps Straßenzüge) und die TECH_ID also die technische Identifikationsnummer, welche das vorherzusagenden Merkmal repräsentiert. Die weiteren Spalten sind das Ergebnis der mathematischen Aufzählung: count steht für die Anzahl der extrahierten Punkte je zehn Meter entlang eines Straßenabschnittes mit einer eindeutigen OSM_ID_NEW und die percentage beschreibt den prozentualen Anteil der vorhergesagten TECH_ID im Verhältnis zu der Gesamtanzahl aller Punkte mit der gleichen OSM_ID_NEW.

Name	Inhalt
OSM_ID_NEW	Die nachträglich erzeugte ID für einen duplizierten Straßenabschnitt mit einer OSM_ID. Repräsentiert zusätzlich eine Straßenseite.

TECH_ID_PREDICT	Die durch das Modell vorhergesagte(n) technischen Identifikationsnummern.
count	Anzahl der Punkte, die für die OSM_ID_NEW die TECH_ID vorhergesagt hat.
percentage	Anteil der Punkte, die diese TECH_ID vorhergesagt hat, bezogen auf die Gesamtanzahl der Punkte der OSM_ID_NEW in Prozent

Tabelle 6: Struktur der Tabelle mit Zwischenergebnissen der Vorhersage, eigene Darstellung

Die Tabelle 7: Vorhersagezwischenergebnisse der TECH_ID auf extrahierte Punkte mit OSM_ID stellt ein Ausschnitt der erhaltenen Zwischenergebnisse der Vorhersage der technischen Identifikationsnummer (TECH_ID) auf die extrahierten Punkte des *OpenStreetMaps* Straßenlayers mit der führenden OSM_ID_NEW als eindeutige Identifikationsnummer dar. Dieses Beispiel präsentiert den Modelldurchlauf mit Vorsortierung der gelabelten Daten, dem k-Distanz Wert gleich drei unter der Anwendung von dem Distanzmaß der euklidischen Distanz und dem k-Wert des k-*Nearest Neighbor* Algorithmus ebenso gleich drei.

Die Ergebnisse jedes Modelldurchlaufs werden standardisiert in Form von einer Exceltabelle gespeichert. Die aufgezeigten ersten acht Fälle des Ergebnisses sind wie folgend zu interpretieren:

Zeilennummer	OSM_ID_NEW	TECH_ID_PREDICT	count	percentage
1	**10106266**	**11060360**	13	100
2	**1010626600**	**11060360**	13	100
3	**13266345300**	**11062910**	15	100
4	**132663454**	**11064030**	12	100
5	**146505392**	**11060360**	56	51,37614679
6		**11060420**	4	3,669724771
7		**11060450**	13	11,9266055
8		**11060460**	36	33,02752294
9	**14650539200**	**11060360**	58	53,21100917
10		**11060420**	4	3,669724771
11		**11060450**	12	11,00917431
12		**11060460**	35	32,11009174

Tabelle 7: Vorhersagezwischenergebnisse der TECH_ID auf extrahierte Punkte mit OSM_ID, eigene Darstellung

Betrachtet man die erste Zeilennummer so sieht man, dass die OSM_ID_NEW gleich 10106266 durch das Modell eindeutig einer TECH_ID_PREDICT gleich 11060360 zugeordnet wurde. Außerdem erkennt man das die OSM_ID_NEW gleich 10106266 durch 13 Punkte (count) repräsentiert wurde, welche die TECH_ID_PREDICT von 11060360 vorhergesagt haben und somit eine 100 %-tigen Zuordnungswert (percentage=100) erhält. Dieses Ergebnis ist somit zu interpretieren, dass der Straßenabschnitt durch 13 Punkte repräsentiert wird, alle weisen die gleiche TECH_ID auf, es folgt daraus, dass es sehr wahrscheinlich ist, dass die Zuordnung der TECH_ID höchst wahrscheinlich korrekt ist und keine besondere Konstellation aufweist.

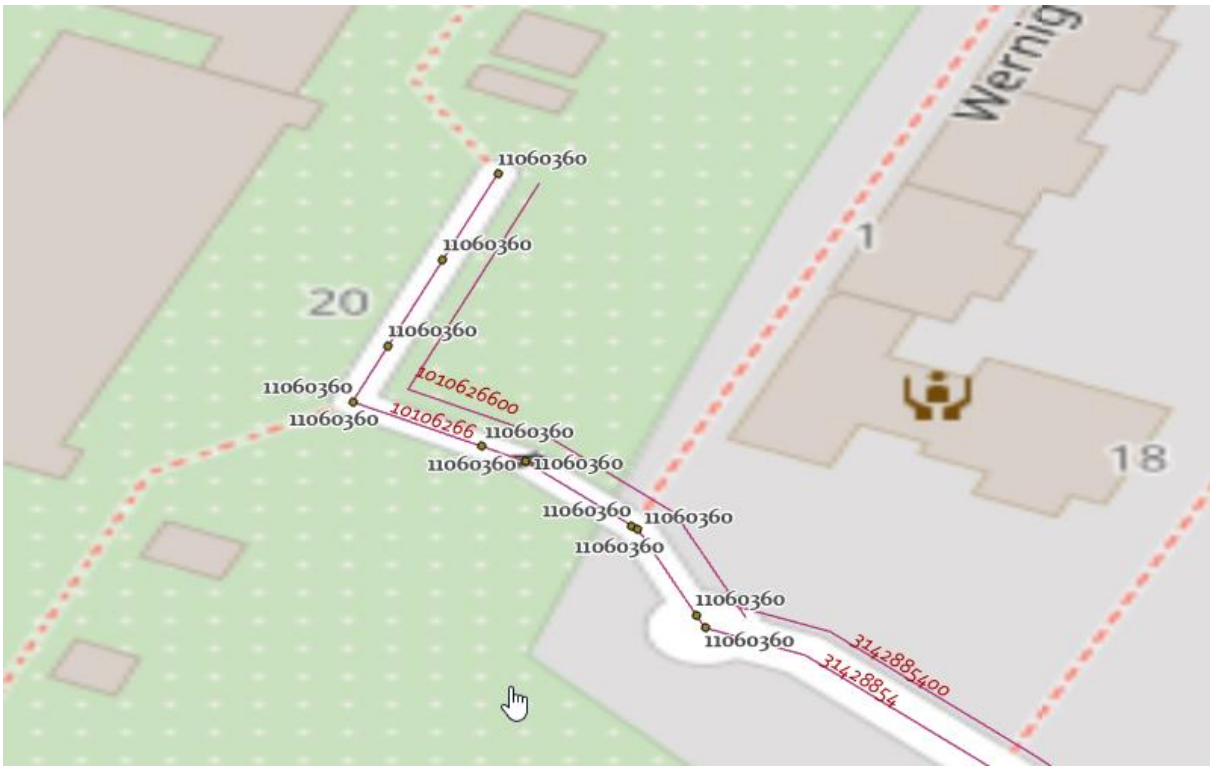


Abbildung 23: Fallbeispiel: Zuordnung der TECH_ID auf OSM_ID_NEW, eigene Darstellung

Im Gegensatz dazu kann man sich die Zeilen fünf bis acht anschauen, die für die OSM_ID_NEW gleich 146505392 vier unterschiedliche TECH_ID_PREDICTS vorhersagen. Der Straßenabschnitt besteht aus 56+4+13+36=109 Punkten denen zu 51,37%, 3,67%, 11,93% und 33,03% die TECH_ID_PREDICTs gleich: 11060360, 11060420, 11060450 und 11060460 vorhergesagt wurden.

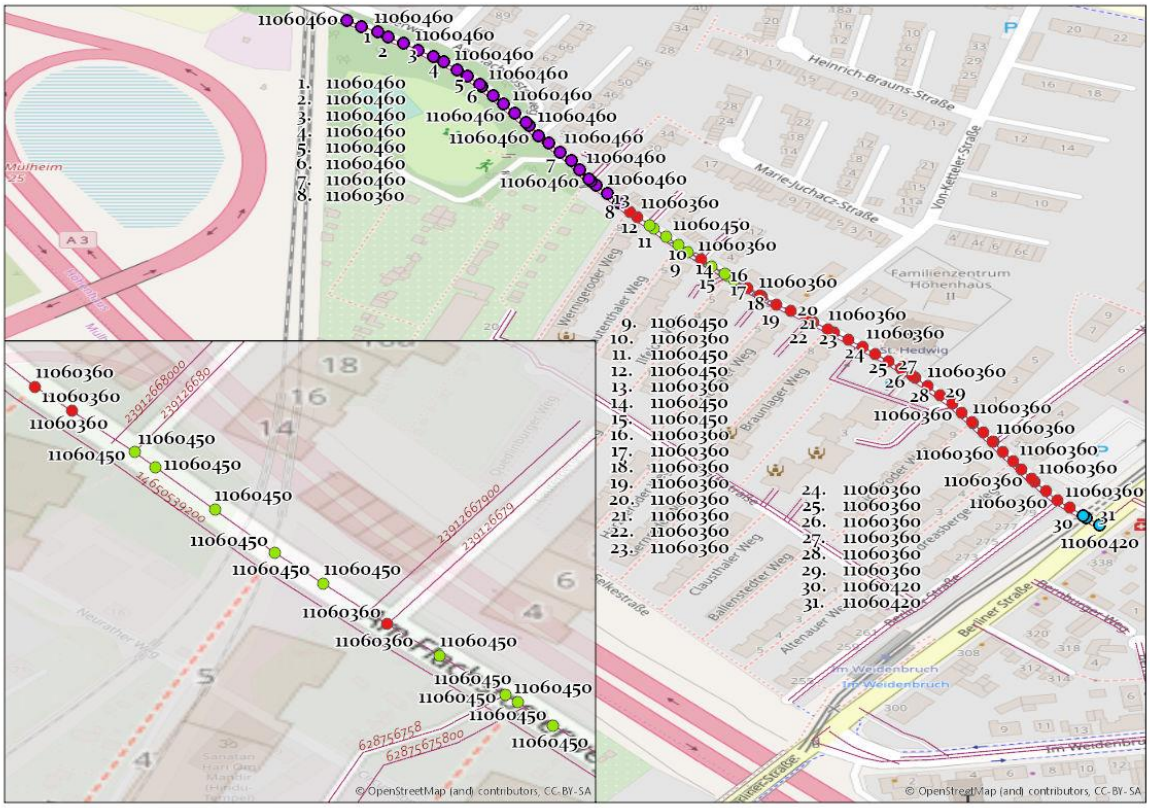


Abbildung 24:Fallbeispiel mehrhafte Zuordnung der TECH_ID zu einer OSM_ID_NEW, eigene Darstellung

Hier besteht die Möglichkeit unterschiedlicher Interpretationen:

- Der Straßenabschnitt ist tatsächlich an mehrere TECH_IDs angeschlossen.
- Die Daten für den Straßenabschnitt sind fehlerhaft.
- Es ist eine Kombination von den beiden vorherig genannten Möglichkeiten.

Welche Interpretation zutreffend ist, kann nur durch eine manuelle Analyse entschieden werden (z.B. fachliche Prüfung in einem Geo- oder Netzinformationssytem durch einen Sachbearbeiter).

Obwohl die vollständige Interpretation nicht automatisch erfolgen kann und der Mensch den Sachverhalt bewerten muss, bringt das Vorgehen einen großen Gewinn mit sich hin.

Das Verfahren hat folgende Vorteile:

- Man muss sich nicht mehr alle Zuordnungen und Straßenabschnitte anschauen, da diejenigen mit einem Anteil von 100% als korrekt angesehen werden können.
- Man kann potentielle Auffälligkeiten automatisiert und in Sekundenschnelle aufzeigen, im Gegensatz zu einer mühsamen manuellen Suche nach Auffälligkeiten.
- Man hat die Möglichkeit, Schwellenwerte zu definieren, ab wann eine fachliche Prüfung notwendig ist, z.B. nur wenn eine TECH_ID_PREDICT einen Anteil von weniger als 1% aufweist.
- Man bekommt mit den anderen vorhergesagten TECH_ID_PREDICT einen Vorschlag, welche Zuordnung potentiell richtig sein könnte.

Allgemein ist die Bewertung der räumlichen Verbindung von hausgenauen Punkten mit streckentreuen Polylinien als positiv zu bewerten. Die in der Tabelle 7: Vorhersagezwischenergebnisse der TECH_ID auf extrahierte Punkte mit OSM_ID dargestellten Ergebnisse sind in Abbildung 23 und Abbildung 24 visualisiert. Beide Formen bestätigen eine erfolgreiche räumliche Verbindung als geographische Operation. Die extrahierten Punkte sind mit den vorhergesagten Werten verbunden und in vektorisierter Form als Shapefile exportiert.
Die Evaluation der Ergebnisse aus fachlicher Sicht erfolgt in nachfolgendem Kapitel.

3.3 Validierung der Ergebnisse

Die Metrik zur Validierung der Ergebnisse ist der Anteil der Punkte pro *OpenStreetMap* Identifikationsnummer (OSM_ID_NEW), welche einer technischen Identifikationsnummer (TECH_ID) zugeordnet wurden umgerechnet in Prozent in Bezug auf Gesamtmenge der extrahierten Punkte aus einem Straßenzug.
Die Aussage, ob eine Zuordnung korrekt ist, kann bei eindeutigen Treffern mit den 100%-tigen Anteil des vorhergesagten Wertes angenommen werden. Für die restlichen

Zuordnungen der vorhergesagten technischen Identifikationsnummer kann folgende Annahme getroffen werden. Je höher der prozentuale Anteil, desto wahrscheinlicher ist es, dass die Zuordnung des Wertes und die Vorhersage treffend sind.

Die Gesamtzahl der Modell-Aufrufe mit unterschiedlichen Parameterkombinationen liegt bei 100 (siehe dazu Absatz 3.1 Bewertung des Modells). Um den Rahmen dieser Abschlussarbeit nicht zu sprengen wird die fachliche Validierung der Ergebnisse anhand von einzelnen Beispielen beschrieben. Hierzu wurde die Parameterkombination von k-Distanz Wert gleich drei, k-Wert gleich drei und der Distanzmetrik euklidische Distanz mit der Optimierung: Sortieren der TECH_ID verwendet.

Für diese Konstellation beträgt die Gesamtzahl der vorhergesagten *OpenStreetMap* Straßenabschnitte (OSM_ID_NEW) 554 Polylinien, was der Anzahl der relevanten Linien entspricht. Die Summe der ungelabelten relevanten Punkte, welche aus den relevanten Linien extrahiert wurden, ist gleich 12.680. (siehe dazu auch Absatz 2.3 Prozessübersicht)
Für diesen Aufruf wurde das Vorhersagemodell mit 2.076 gelabelten Punkten trainiert und gegen 519 gelabelten Punkten getestet. Die Modellgüte ergab 87,86% (siehe dazu 3.1.2 Einfluss des k-Wertes auf die Modellgüte).

Im Falle einer mehrfachen Zuordnung von vorhergesagten technischen Identifikationsnummer (TECH_ID_PREDICT) ergibt der prozentuale Anteil der Punkte (percentage) für einen Straßenabschnitt (OSM_ID_NEW) insgesamt 100% (siehe Tabelle 6: Struktur der Tabelle mit Zwischenergebnissen der Vorhersage).

Die tabellarische Auflistung der in Klassen zusammengefassten Ergebnisse in der Tabelle 8: Anteil der Punkte pro OpenStreetMap Identifikationsnummer zeigt die Unterteilung der prozentualen Werte für jede zusammengefasste Gruppe. Die Summe repräsentiert die

Anzahl der eindeutigen *OpenStreetMap* Straßenabschnitte innerhalb einer Klasse, diese ist die Grundlage für die Berechnung vom prozentualen Anteil auf die Gesamtmenge.

Klasse (percentage)	Summe eindeutige OSM_ID_NEW	% Anteil an Anzahl OSM_ID_NEW
0-10	60	11%
10-25	95	17%
25-50	97	18%
50-80	71	13%
80-99	45	8%
100	186	34%
Gesamtergebnis	**554**	**100%**

Tabelle 8: Anteil der Punkte pro OpenStreetMap Identifikationsnummer, eigene Darstellung

Ein Drittel der Daten gehört der Klasse 100 an, welche die 100%-tige und korrekt zugeordnete TECH_ID repräsentieren und bedürfen keiner weiteren Prüfung. Für die restlichen Fälle muss eine fachliche Entscheidung getroffen werden, ab welchen Schwellenwert eine manuelle Prüfung der vorhergesagten Werte erforderlich ist. Das Festlegen von einem Schwellenwert hängt sehr stark von dem Verwendungszweck der Vorhersage ab.

Die Ergebnisse aus der Klasse 80-99 machen acht Prozent der Zuordnungen aus, was gegenüber anderen Klassen deutlich weniger ist. Die TECH_ID Vorhersage mit dem percentage Wert zwischen 80% und 99% könnte auch ohne weitere Prüfung als korrekt bezeichnet werden. Die Beispiele aus der Tabelle 9: Fallbeispiel der Treffer aus der Klasse 80-99, eigenen Darstellung beinhalten mindestens ein Wert aus der Klasse 80-99. Anhand dieser Beispiele ist sichtbar, dass ein Straßenabschnitt mehreren TECH_ID Werten zugeordnet werden kann, sowie dass auch benachbarten Straßenabschnitten der gleiche TECH_ID Wert zugeordnet werden kann.

OSM_ID_NEW	TECH_ID_PREDICT	count	percentage
29098064	**11062390**	19	**95**
	11062430	1	**5**
2909806400	**11062380**	2	**10**
	11062390	17	**85**
	11062430	1	**5**
146505402	**11063210**	20	**80**
	11063220	1	**4**
	11063360	4	**16**
147059194	**11055630**	13	**92,86**
	11055670	1	**7,14**
14705919400	**11055630**	13	**86,67**
	11055670	2	**13,33**
166548734	**11055920**	1	**7,69**
	11063820	12	**92,31**

Tabelle 9: Fallbeispiel der Treffer aus der Klasse 80-99, eigenen Darstellung

Angenommen der Schwellenwert von korrekten Zuordnungen liegt bei mindestens 80%, dann würde dies bedeuten, dass die TECH_ID_PREDICT als korrekt eingestuft wären und eine manuelle Prüfung nicht erforderlich wäre. Die andere Werte, welche dem gleichen Straßenabschnitt zugeordnet wurden, haben geringere percentage Werte ergeben aber in Summe von den restlichen Prozenten Bereich 100%.

Die Tabelle 9: Fallbeispiel der Treffer aus der Klasse 80-99, eigenen Darstellung zeigt die möglichen Zuordnungen mit den Werten: 95% + 5%, 10% + 85% + 5%, 80% + 4% + 16%, 92,86% + 7,14%, 86,67% + 13,33% und 7,69% + 92,31%. Daher, dass die niedrigen Prozentwerte der gleichen OSM_ID_NEW zugeordnet sind, wird man bei Einzelfallprüfung natürlich auch die Werte aus der Klasse 80-99 sehen, obwohl diese bereits als korrekt angenommen werden.

Die Abbildung 25: Evaluationsbeispiel der vorhergesagten technischen Identifikationsnummern stellt den ersten und den zweiten Wertepaar dar.

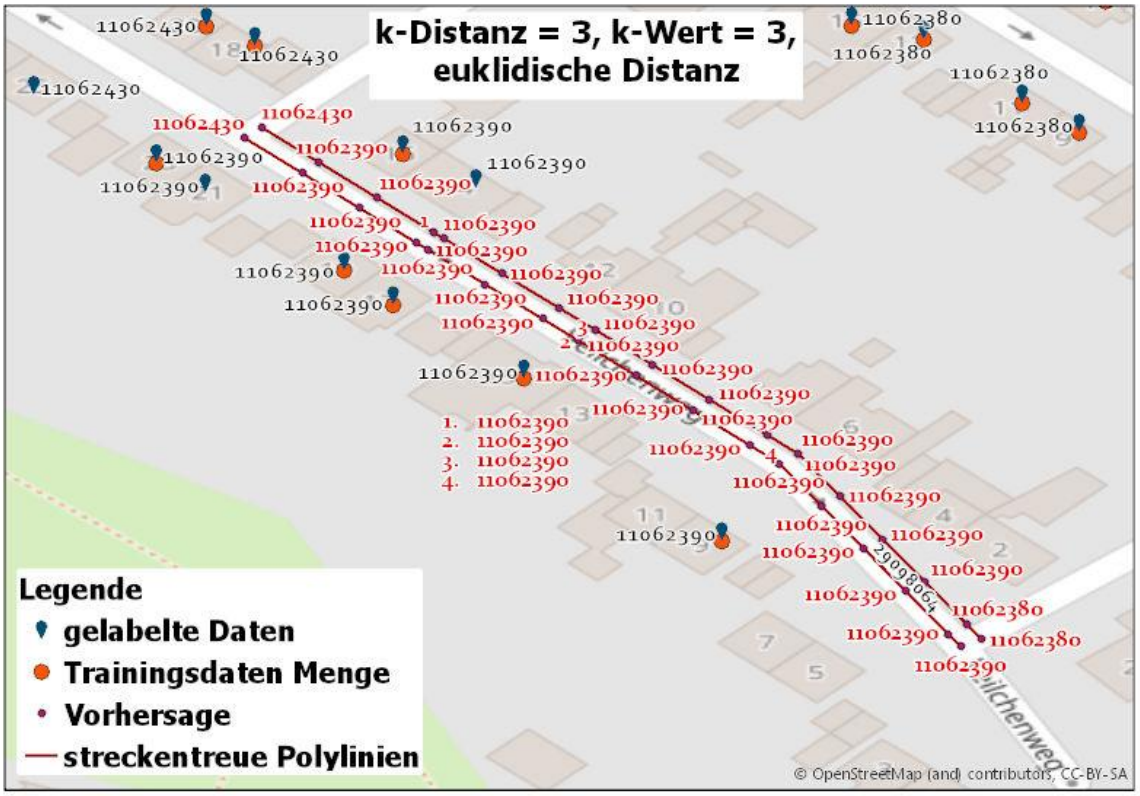


Abbildung 25: Evaluationsbeispiel der vorhergesagten technischen Identifikationsnummern, eigene Darstellung

Zusätzlich wurden die gelabelten Daten und die Trainingsdaten visualisiert, um die reale Situation der Aufteilung der technischen Identifikationsnummer zu zeigen. Die niedrigen prozentuale Werte (percentage) sind sog. Anschlusspunkte, wo sich die Kabelnetzabschnitte mit einer unterschiedlichen TECH_ID treffen. Die Länge des Straßenabschnittes trägt auch dazu bei, wie gut die Vorhersage getroffen werden kann, weil je länger ein Abschnitt, desto mehr unterschiedlichen Werte müssen einer OSM_ID_NEW zugeordnet werden (siehe Abbildung 23: Fallbeispiel: Zuordnung der TECH_ID auf OSM_ID_NEW). Auffällig ist, dass die Zuordnung der TECH_ID an den Endpunkten der Straßenabschnitte und auch in der Situation, wenn sich zwei Straßenabschnitte kreuzen, zum Wechsel der Werte tendiert. Diese Vorhersage scheint in diesem konkreten Fall als prüfungswürdig, denn aus der Abbildung ist ersichtlich, dass die drei nächsten Nachbarn

des Endpunktes aus den Trainingsdatensätzen die TECH_ID: 11062390, 11062430 und 11062430 haben. Nach dem Prinzip der Klassifikation Nächster Nachbar wird der Wert, welcher in der Mehrzahl vorkommt zugeschrieben.

Somit zeigt sich, dass mithilfe der Vorgehensweise und der Festlegung eines Schwellenwertes, auffällige Fallkonstellationen identifizieren lassen, die einer genaueren Prüfung bedürfen. Diese genaue fachliche Prüfung könnte nun im Rahmen einer Evaluation der Ergebnisse gegen die originale Netzbaupläne erfolgen.
Da diese aber im Rahmen dieser Abschlussarbeit nicht vorliegen, kann dies hier nicht erfolgen.

4 Diskussion

Die vorgeschlagene Vorgehensweise zur Nachbildung von physischen Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien hat sich im Rahmen der Bewertungskriterien als erfolgreich bestätigt. Im Rahmen dieser Abschlussarbeit wurde ein maschinelles Lernen Verfahren zur Vorhersage des zu verbindenden Attributes der technischen Identifikationsnummer erfolgreich angewendet.

Die empirische Forschungsarbeit im Rahmen der Abschlussarbeit bestätigt die gestellte Fragestellung, ob eine rechnergestützte, räumliche Verbindung von Punktdaten zu Polylinien, die mithilfe eines *lazy learning* Vorhersagemodells durchgeführt wird, zur Nachbildung eines physikalischen Kabelnetzes, basierend auf Metainformationen eines hausgenauen Punktes und einem gegebenen streckentreuem Polyliniennetz geeignet ist.

Die klassische Form der räumlichen Verbindung verbindet zwei Datensätze nur nach dem Erfüllen der Verbindungsbedingung. Bei der Verbindung anhand der kürzesten Distanz wird in diesem Fall genau ein eindeutiger Treffer ausgegeben, wo keine inhaltliche Bewertung des Verbindungsattributes mehr vorgenommen werden kann. (siehe dazu 2.1 Spatial join, die räumliche Verbindung)
Demgegenüber besteht bei der vorgestellten Vorgehensweise unter Anwendung von maschinellem Lernen die Möglichkeit die Vorhersage mithilfe der Zwischenergebnisse und der anteiligen Punktverteilung zu bewerten und Schwellenwerte für eine Prüfung zu definieren. Dieses Vorgehen bietet also einen Mehrwert gegenüber einer klassischen Form der räumlichen Verbindung und ermöglicht neue Anwendungebiete.

Die in dieser Abschlussarbeit verwendeten Punkt-Daten zur Abbildung des Kabelnetzverlaufes sind Echtdaten, d.h. um die durch das Modell gefundenen auffällige Werte abschließend beurteilen zu können, müssten diese nun gegen die Realität geprüft

werden, ob es sich dabei um echte Datenfehler handelte oder um den statistischen Fehler des Modells, da das Modell keine 100% Modellgüte aufweist. Diese Überprüfung ist nur durch eine manuelle Einzelfallanalyse und ggf. durch eine Vorort-Begehung möglich, da dafür die Kenntnis über die realen Kabelverläufe notwendig ist.
Das durch das Modell auf Basis des vektorisierten OSM-Layers vorhergesagte Kabelnetz konnte im Rahmen dieser Abschlussarbeit nicht mit dem bereits vektorisierten Kabelnetz der Unitymedia NRW GmbH verglichen werden, da diese nur die hausgenauen Punktdaten übergeben haben. Ein Abgleich zwischen dem vorhergesagten Kabelnetz und dem tatsächlich im Netzinfrastrukturinformationssystem (NIS) vorliegenden Kabelnetz würde die tatsächliche fachliche Modellgüte bestimmen.

Die Vorteile der entwickelten Vorgehensweise kommen genau dann optimal zum tragen, wenn:

- Ein niedriger prozentualer Richtwert für die Einzelfallprüfung ausgewählt wird.
- Die Ergebnisse in einer kurzen Zeit vorliegen sollen.
- Die Datenmenge sehr hoch ist.
- Eine Datenqualitätsprüfung vorgenommen werden soll.
- Die menschlichen Ressourcen zu knapp sind.
- Das Endergebnis unbekannt ist bzw. keine Kabelnetzdokumentation vorliegt.
- Die Vorgehensweise der Zuordnung der Werte nachvollziehbar, reproduzierbar und dokumentiert sein soll.

Die statistische Modellgüte ist von der Struktur und der Datenqualität der Eingabedaten unabhängig, d.h. auch wenn falsch aufbereitete Daten oder ein nicht sinnvolles Trainingsattributs ausgewählt wurde, kann die statistische Modellgüte hoch sein, obwohl das Modell für den eigentlich vorgesehenen realen Einsatzzweck nicht geeignet ist. Im Rahmen dieser Abschlussarbeit wurde die statistische Modellgüte als Erfolgskriterium für

das Vorhersagemodell definiert, um eine Objektivierung und Generalisierung der Vorgehensweise zu ermöglichen. (siehe dazu 2.2.4 Festlegen der Bewertungskriterien für das *lazy learning* Modell) Anhand dieser statistischen Modellgüte können die vorab definierten Teilfragen nun beantwortet werden (siehe dazu Absatz 1.4 Forschungsfrage).

- Welchen Einfluss auf die Güte des Modells hat der k-*Nearest Neighbor* Algorithmus Wert?
- Dominiert ein Distanzmaß, die euklidische Distanz oder die Manhattan Distanz?

Erwartet wurde, dass wie bei der Anwendung der Multidimensionalen Daten (Xia, Lu et al., 2004) eine Vorsortierung der Daten auch einen Einfluss auf die Genauigkeit der Vorhersage hat. Diese Erwartung wurde bestätigt für alle k-Werte größer als eins (siehe Absatz 3.1.3.3 *Einfluss der Vorsortierung auf das Modell*). Eine Vorsortierung der Daten erweist sich also als vorteilhaft für die Modellgüte.

Die Auswahl des k-Distanz Wertes hat keinen Einfluss auf die Modellgüte (siehe 3.1.3.1 *Einfluss des k-Distanz Wertes auf das Modell*). Trotzdem ist dieser Parameter für das Modell sinnvoll, da er die zu betrachtenden Polylinien einschränkt und zumindest zu einer Performance-Verbesserung des Modells beiträgt.

Wie zu erwarten wird die Modellgüte hauptsächlich durch die Auswahl des k-Wert Parameters beeinflusst. Die Modellgüte sinkt mit steigendem k-Wert. Obwohl ein k-Wert von eins die höchste Modellgüte aufweist, verliert man mit diesem Wert viele Vorteile des Verfahrens. Hier zeigt sich besonders deutlich das die Bewertung des Modells sich nicht rein anhand von statistischen Kennzahlen, sondern auch am Einsatzzweck orientieren muss (siehe 3.1.2 Einfluss des k-Wertes auf die Modellgüte).

Es hat sich kein Distanzmetrik als dominierend erwiesen, d.h. bei zukünftiger Anwendung des Modells sollte man immer sowohl die euklidische als auch die Manhattan Distanz nutzen und prüfen, welches zu einer besseren Modellgüte führt (siehe 3.1.3.2 *Einfluss der Distanzmetrik auf das Modell*). Eventuell liegt dies an den geringen Abständen zwischen den Punktdaten, die sich im zweistelligen Meterbereich befinden, so dass sich für Anwendungsfälle mit größeren Abständen ggf. doch eine Distanzmetrik dominiert.

Vorteile der *lazy learning* Vorhersage zur Ableitung von physische Kabelverläufen aus hausgenauen Punktdaten und streckentreuen Polylinien:

- Die Ausgabe der Vorhersage eines Attributes mit prozentualem Anteil der unterschiedlichen Möglichkeiten lässt mehr Freiraum und Entscheidungsfreiheit bei der Übernahme der Ergebnisse, der Plausibilisierung und der Entdeckung von Auffälligkeiten
- Ein generisches Modell der Vorgehensweise, welches leicht die Anwendung auf andere Punktdaten mit technischer Information über ein Versorgungsnetz zulässt.
- Gibt die Möglichkeit eigene spezifische Parameter für den k-*Nearest Neighbor Classifier* einzusetzen.
- Die Abgrenzung des *OpenStreetMap* Straßenlayers auf die relevanten Linien grenzt diese automatisch auf die Ausdehnung der Eingabedaten ab.
- Verbessert die Genauigkeit der räumlichen Verbindung der hausgenauen Punkte zur streckentreuen Polylinien durch Anwenden von extrahierten Punkten aus den Straßenabschnitten, welche gleich den vorab ausgewählten relevanten Linien sind.
- Das Ergebnis der Vorgehensweise ist reproduzierbar.
- Das Ausführend des Python-Skriptes bedarf keinerlei kommerzieller Software Installation und basiert ausschließlich auf frei zugänglichen Bibliotheken und freien Daten (*OpenStreetMap* Straßenlayer).

5 Schlussfolgerungen und Ausblick

Im Rahmen der definierten Bewertungskriterien ist die Vorhersage der physischen Kabelnetzverläufe aus hausgenauen Punktdaten und streckentreuen Polylinien mit dem *lazy learning* Verfahren als gelungen zu bewerten. Eine abschließende Bewertung kann allerdings erst mit einem realen Anwendungsfall vollständig erfolgen.

Die in dieser Abschlussarbeit vorgestellte Vorgehensweise ist so allgemein, dass sie je nach Ziel und Anwendungsfall spezifisch angepasst werden kann, die Grundstruktur als Rahmen aber aufrecht gehalten werden kann.

Die Vorgehensweise bietet noch unterschiedliche Optimierungsmöglichkeiten, von denen einige denkbare in Abbildung 26: Optimierungsvorschläge des Vorhersagemodells dargestellt sind.

Durch die Übertragung des Optimierungsschrittes zur Ermittlung der k-relevanten Linien auf die Punktdaten zur Ermittlung der k-relevanten Punkte könnte die Datenmenge weiter einschränken und die Performanz, als auch den Nutzen des Modells in einem realen Anwendungsfall weiter erhöhen. Dies wäre dann der Fall, wenn sich durch die Auswahl der relevanten Punkte die Fehlklassifikation durch Modellfehler weiter verringert, so die Aussage über die Anteile der Punkte einer Polylinie präziser wird und dadurch eine geringere Menge an notwendigen Einzelfallprüfungen ergibt.

Denkbar wäre auch die Einführung einer Gewichtung weiterer Parameter im Prozess der Auswahl der nächsten Nachbarn. Hierzu könnte zum Beispiel die gleiche Straßenseite des gelabelten und ungelabelten Punktes bei einer Kreuzung oder Überschneidung der Straßenabschnitte stärker gewichtet werden, um eine noch bessere Modellgüte zu erreichen.

Eine weitere Anpassung könnte der Einsatz von einem Entscheidungsbaum sein, um in Spezialfällen die Anzahl der korrekten Zuordnungen weiter zu erhöhen. Die bekannten besonderen Fallkonstellationen des Verlaufes innerhalb eines Kabelnetzes müssten dann in einem Katalog von Handlungsmöglichkeiten erfasst werden. Hiermit gewinnt das Modell an Präzision, würde aber an Allgemeingültigkeit verlieren, da es dann auf einen bestimmten Anwendungsfall hin optimiert würde.

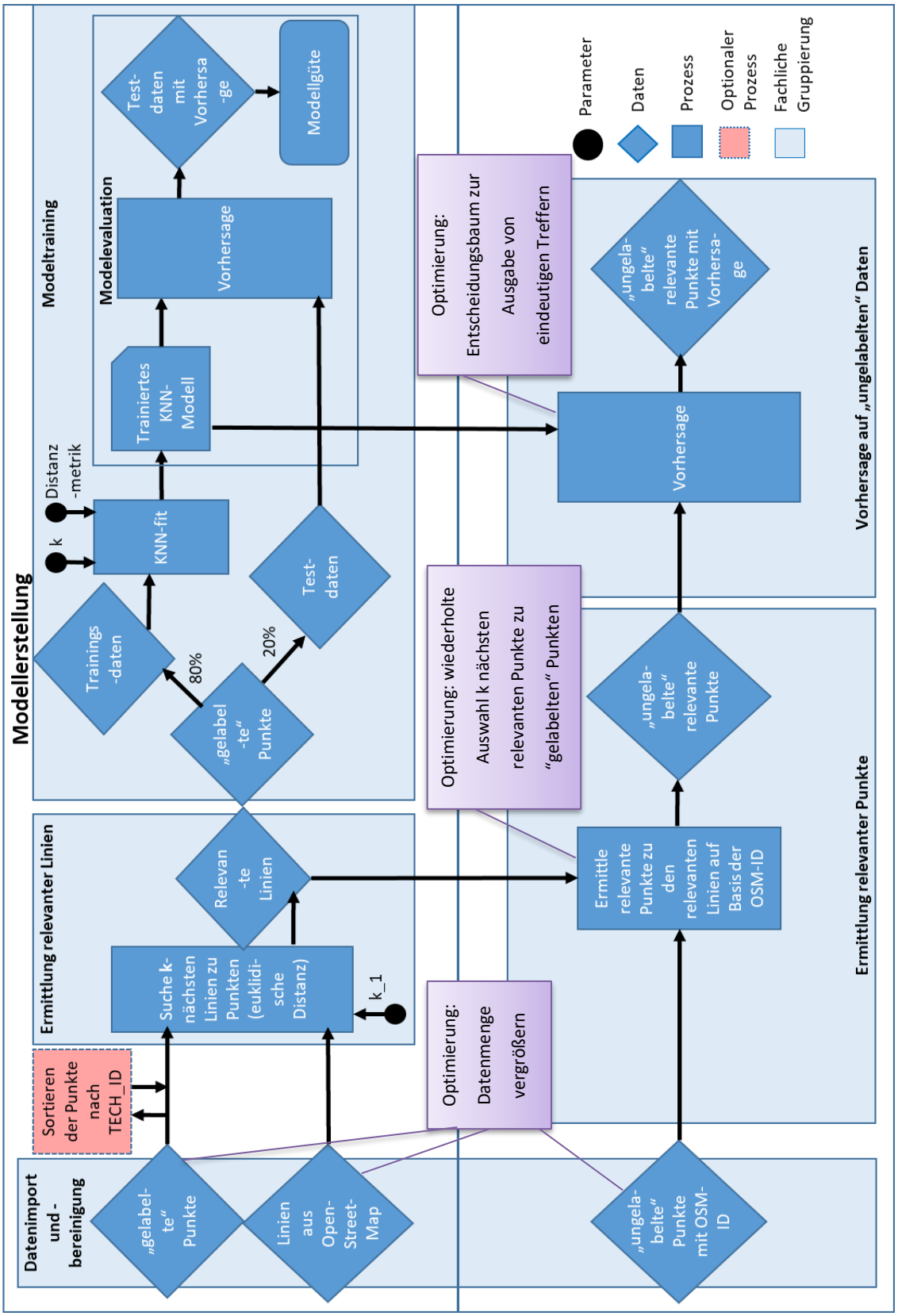


Abbildung 26: Optimierungsvorschläge des Vorhersagemodells, eigene Darstellung

Die in dieser Abschlussarbeit beschriebene Vorgehensweise kann in weitere Disziplinen übertragen werden, wenn als Ziel eine Erkennung von Anomalien oder auch Datenqualitätsproblemen gesetzt wird. So lassen sich bisher unbekannte Datenfehler in den Eingabedaten identifizieren. Somit könnte dieser Prozess ein Vorschlag zur Datenqualitätssicherung für Versorgungsunternehmen aus der Telekommunikationsbranche oder Grundversorgung sein.

Die Methode kann auch dazu genutzt werden, um den Datenbestand eines Unternehmens auf Datenqualitätsmängel zu überprüfen und diese automatisiert zu korrigieren.
Natürlich sind eigene Besonderheiten beim Verlauf eines Versorgungsnetzes, wie z. B. der Verlauf von Gasleitungen zu berücksichtigen, da diese mit den getroffenen Annahmen für ein Telekommunikationsnetz nicht zwingend übereinstimmen muss.
Die Vorgehensweise kann auch für die Weiterentwicklung des Analytischen Kostenmodels der Bundesnetzagentur von Bedeutung sein, um die Forschungsarbeit des WiK-Consult GmbH Instituts fortzusetzen. (Kulenkampff, 2019)

6 Literaturverzeichnis

Aha, D.W. 2013. *Lazy Learning*: Springer Netherlands.

Bahrenberg, G., Giese, E., Mevenkamp, N. and Nipper, J. 2008a. Statistische Methoden in der Geographie-Band 2: Multivariate Statistik.

Bahrenberg, G., Giese, E., Mevenkamp, N. and Nipper, J. 2008b. *Statistische Methoden in der Geographie.* 3., neubearbeitete Auflage ed. Stuttgart: Gebr. Borntraeger Verlagsbuchhandlung Berlin.

Bezirksregierung_Köln. 2019. *Hausumringe* [*online*]. Available at: https://www.bezreg-koeln.nrw.de/brk_internet/geobasis/liegenschaftskataster/hausumringe/index.html [Accessed 15.09.2019.

Böhm, C. and Krebs, F. 2004. *The k-Nearest Neighbour Join: Turbo Charging the KDD Process*.

Braun, S. 2003. *Der Zugang zu wirtschaftlicher Netzinfrastruktur: Telekommunikation, Schienenverkehr und Energiewirtschaft im Spannungsfeld staatlicher Interessen und deren Regulierung durch sektorspezifisches Recht auf einem Wettbewerbsmarkt.* Herbolzheim: Centaurus Verlag.

de Lange, N. and Nipper, J. 2018. *Quantitative Methodik in der Geographie.*: Verlag Ferdinand Schöningh.

Ester, M., Kriegel, H.-P. and Sander, J. 1997. Spatial data mining: A database approach. International Symposium on Spatial Databases, Springer.

Flake, G.W. 1998. *The computational beauty of nature [electronic resource]: computer explorations of fractals, chaos, complex systems, and adaptation*: Mit Press.

Foroutan, M. and Zimbelman, J.R. 2017. Semi-automatic mapping of linear-trending bedforms using 'Self-Organizing Maps' algorithm. *Geomorphology* 293 156-166.

Gatrell, A.C. 1983. *Distance and Space: A Geographical Perspective*: Clarendon Press.

Gatrell, A.C. 1991. Concepts of space and geographical data. *In* D. J. G. Maguire, M. F.; Rhind, D. W. ed. *Geographical information systems: principles and applications.* . Longman/New York, John Wiley & Sons Inc. . pp. 119-134.

Goodchild, M.F. 1988. Geographic information systems. *Progress in Human Geography* 12(4) 560-566.

Gunther, O. 1993. Efficient computation of spatial joins. Proceedings of IEEE 9th International Conference on Data Engineering.

Haucap, J. and Coenen, M. 2010. *Regulierung und Deregulierung in Telekommunikationsmärkten: Theorie und Praxis*: DICE ordnungspolitische Perspektiven.

Jacox, E.H. and Samet, H. 2007. Spatial join techniques. *ACM Trans. Database Syst.* 32(1) 7.

Jolliffe, I. 2011. Principal Component Analysis. *In* M. Lovric ed. *International Encyclopedia of Statistical Science*. Berlin, Heidelberg, Springer Berlin Heidelberg. pp. 1094-1096.

Kappas, M. 2012. *Geographische Informationssysteme*: Westermann.

Koperski, K., Han, J. and Stefanovic, N. 1998. An efficient two-step method for classification of spatial data. Proceedings of International Symposium on Spatial Data Handling (SDH'98).

Kubat, M. 2015. Similarities: Nearest-Neighbor Classifiers. *An Introduction to Machine Learning*. Cham, Springer International Publishing. pp. 43-64.

Kulenkampff, G.P., T.; Zoz, K. 2018. Analytischen Kostenmodell Anschlussnetz AKM-AN Version 3.0. Weiterentwicklung im Kontext der ND&KRM- Empfehlung. Available at: https://www.bundesnetzagentur.de/SharedDocs/Downloads/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Marktregulierung/Massstaebe_Methoden/Kostenmodelle/Anschlussnetz/20180515PraesentationWIKConsult.pdf?__blob=publicationFile&v=2 [Accessed 15.09.2019].

Kulenkampff, G.P., T.; Zoz, K. 2019. Analytisches Kostenmodell für das Anschlussnetz AKM-AN Version 3.0. Available at: https://www.bundesnetzagentur.de/SharedDocs/Downloads/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Marktregulierung/Massstaebe_Methoden/Kostenmodelle/Anschlussnetz/20190114_AKM_AN_RefDokpdf.pdf?__blob=publicationFile&v=2 [Accessed 15.09.2019].

Kurth, M. 2003. Privatisierung/Deregulierung/Marktverfassung: Die Sicht der Regulierungsbehörde. *Perspektiven der Wirtschaftspolitik* 4(3) 341-358.

Miller, H.J. 2004. Tobler's First Law and Spatial Analysis. *Annals of the Association of American Geographers* 94(2) 284-289.

Mishra, P. and Eich, M.H. 1992. Join processing in relational databases. *ACM Computing Surveys (CSUR)* 24(1) 63-113.

Ng, R.T. and Han, J. 1994. Efficient and Effective Clustering Methods for Spatial Data Mining. Proceedings of VLDB.

OpenStreetMap. 2019a. *Download OpenStreetMap data for this region: Regierungsbezirk Köln* [*online*]. Available at: http://download.geofabrik.de/europe/germany/nordrhein-westfalen/koeln-regbez.html [Accessed 15.09.2019.

OpenStreetMap. 2019b. *OpenStreetMap Blog* [*online*]. Available at: https://blog.openstreetmap.org/impressum/?lang=de [Accessed 15.09.2019.

OpenStreetMap. 2019c. *OpenStreetMap stellt Kartendaten für tausende von Webseiten, Apps und andere Geräte zur Verfügung* [*online*]. Available at: https://www.openstreetmap.org/about [Accessed 15.09.2019.

Orenstein, J.A. 1989. Redundancy in spatial databases. ACM SIGMOD Record, ACM.

Ramm, F. 2019. OpenStreetMap Data in Layered GIS Format. Available at: http://download.geofabrik.de/osm-data-in-gis-formats-free.pdf [Accessed 15.09.2019].

Raschka, S. 2017. *Machine Learning mit Python. Das Praxis-Handbuch für Data Science, Predictive Analytics und Deep Learning.* Frechen: mitp Verlags GmbH & Co. KG.

Reinhardt, W. and Bockmuehl, T. 2013. *Prozessorientiertes Qualitätsmanagement bei der Aktualisierung von GIS/NIS-Daten – Hintergrund und Ergebnisse einer Praxisstudie*.

Seidl, T. and Kriegel, H.-P. 1998. Optimal multi-step k-nearest neighbor search. ACM Sigmod Record, ACM.

StadtKöln. 2019. *Offene Daten Köln. Stadtbezirke* [*online*]. Available at: https://www.offenedaten-koeln.de/dataset/stadtbezirke [Accessed 15.09.2019.

Tobler, W.R. 1970. A Computer Movie Simulating Urban Growth in the Detroit Region. *Economic Geography* 46 234-240.

Torres-Moreno, J.-M., Bougrain, L. and Alexandre, F. 2009. Combining Supervised and Unsupervised Learning for GIS Classification.

U.S.GeologicalSurvey 2006. FGDC Digital Cartographic Standard for Geologic Map Symbolization (PostScript Implementation): U.S. Geological Survey Techniques and Methods Available at: http://pubs.usgs.gov/tm/2006/11A02/ [Accessed15.09.2019].

Ullman, J.D. 1988. *Principles of Database and Knowledge-base Systems*: Computer Science Press.

Watts, D.J. 1999. Networks, dynamics, and the small-world phenomenon. *American Journal of sociology* 105(2) 493-527.

Xia, C., Lu, H., Ooi, B.C. and Hu, J. 2004. Gorder: an efficient method for KNN join processing. Proceedings of the Thirtieth international conference on Very large data bases-Volume 30, VLDB Endowment.

7 Python Code

```
import data_loader as dl
import geopandas as gpd
import pandas as pd
from shapely.geometry import Point, LineString
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier
import numpy as np
import fiona
import fiona.crs
import os
import logging
import datetime


def setup_logger(name, log_file, level=logging.INFO):
    """Function setup as many loggers as you want"""

    handler = logging.FileHandler(log_file)

    logger = logging.getLogger(name)
    logger.setLevel(level)
    logger.addHandler(handler)

    return logger

basic_logger =
setup_logger("basic_logger",'E:\PythonProjects\Masterarbeit\data\SH_resul
t_neu\Console_Logfile.txt')
summary_logger =
setup_logger("summary_logger",'E:\PythonProjects\Masterarbeit\data\SH_res
ult_neu\Summary_Logfile.csv')

summary_logger.info('Index;Datum;Anzahl Linien OSM;Anzahl Punkte
TECH_ID;Punkte nach Bereinigung TECH_ID;k Distanz;Anzahl relevanten
Linien OSM;Anzahl Punkte OSM extrahiert 10m;Anzahl Punkte im
Training;Anzahl Punkte für Validierung;Summe gelabelte Daten;Anzahl
ungelabelte Daten;k NN Algorithm;KNN Algorithm;Distanz Metric;Parameter
Minkowski Metric p=;Distanz Name;Modell Genauigkeit %')

result_dict = {}


# Einfuegen der Projektion beim Export in Shapefile
# WGS84
crs = {'init': 'epsg: 4326'}

def get_min_distance_line_id(point_geometry, lines, k):

    distance_point_lines = lines.distance(point_geometry)
```

```
    lines.loc[:, 'distance'] = distance_point_lines

   # Sortieren der Datensätze nach kürzesten Distanz
    lines.sort_values(by='distance', ascending=True, inplace=True,
na_position='last')

    result = lines.head(k)
    return result

def extract_unique_lines_from_column(column_with_line_array):
    mylines = column_with_line_array
    df_relevant_lines = pd.DataFrame(mylines[0])
    for i in range(1, len(mylines)):
        df = pd.DataFrame(list(mylines)[i])
        df_relevant_lines = pd.concat([df_relevant_lines, df],
ignore_index=True, verify_integrity=True)

    # Bei der Auswahl von relevanten Linien kann es zur duplizieren der
Daten kommen
    # deswegen Bereinigung um die Duplikate
    df_relevant_lines.drop_duplicates(subset=['osm_id_new'],
inplace=True)
    df_unique_relevant_lines = gpd.GeoDataFrame(df_relevant_lines,
crs=crs, geometry='geometry')

    return df_unique_relevant_lines

def get_points_from_line(line):
    line = LineString(line)
    x,y = line.coords.xy

    points = [Point(my_x, my_y) for my_x, my_y in zip(x,y)]
    return points

def transform_data_from_point_line_shape2(gpd_points_from_shape):
    print('{0} Punkte
eingelesen.'.format(gpd_points_from_shape.shape[0]))
    basic_logger.info('{0} Punkte
eingelesen.'.format(gpd_points_from_shape.shape[0]))

    print('gpd_points_from_shape columns:
{0}'.format(gpd_points_from_shape.columns))

    array1 = gpd_points_from_shape.loc[:, ['Extrahie_1', 'Extrahie_4',
'geometry']].values

    # Extrahiere die Shapely Point Objekte
    object_ids, osm_values, points = zip(*array1)

    # Mach aus dem Point der Linien, Arrays mit x,y
    x_values = np.array([])
    y_values = np.array([])

    for my_point in points:
            x_values = np.append(x_values, my_point.coords.xy[0])
```

```
            y_values = np.append(y_values, my_point.coords.xy[1])

    xy_line_points_zip = list(zip(x_values, y_values))
    line_point_coords = np.array(list(xy_line_points_zip))
    osm = np.array(osm_values)
    gpd_points_from_lines = gpd.GeoDataFrame(
        {'X': line_point_coords[:, 0], 'Y': line_point_coords[:, 1],
'OSM_ID_NEW': osm, 'geometry': [Point(x, y) for x, y in zip(x_values,
y_values)]}, crs=crs)
    #
    print('{0} Punkte
verarbeitet.'.format(gpd_points_from_lines.shape[0]))
    basic_logger.info('{0} Punkte
verarbeitet.'.format(gpd_points_from_lines.shape[0]))
    #
    return xy_line_points_zip, x_values, y_values, osm_values,
gpd_points_from_lines

def get_XY_coords_from_filtered_data(gpd_points_from_lines_filtered):
    print('{0} Punkte
eingelesen.'.format(gpd_points_from_lines_filtered.shape[0]))
    basic_logger.info('{0} Punkte
eingelesen.'.format(gpd_points_from_lines_filtered.shape[0]))
    #
    array1 = gpd_points_from_lines_filtered.loc[:, ['X','Y']].values

    x, y = zip(*array1)

    x_values = np.array(x)
    y_values = np.array(y)
    xy_line_points_zip = list(zip(x_values, y_values))
    #
    print('{0} Punkte in XY Koordinaten numpy Array
umgewandelt.'.format(len(xy_line_points_zip)))
    basic_logger.info('{0} Punkte in XY Koordinaten numpy Array
umgewandelt.'.format(len(xy_line_points_zip)))
    #
    return xy_line_points_zip

def transform_training_data(gpd_points):
    #
    array1 = gpd_points.values

    # Extrahiere tech_IDs und die Shapely Point Objekte
    tech_ids, points = zip(*array1)

    # X sind die Points
    x = [coord.x for coord in points]
    y = [coord.y for coord in points]
    gpd_points['X'] = x
    gpd_points['Y'] = y
    xy_zip = list(zip(x, y))
    X = np.array(list(xy_zip))

    # Y sind die technischen IDs zur Vorhersage
```

```
    Y = np.array(tech_ids)
    gpd_all_points = gpd.GeoDataFrame({'X': X[:, 0], 'Y': X[:, 1],
'TECH_ID_TRUE': Y}, crs=crs)

    #
    return X, Y, gpd_all_points

def import_data():
    # 1. Import
    # Import data
    # importiere streckentreue Linien und hausgenaue Punkte aus
Shapefiles mit Geopands
    lines_path =
r'E:\PythonProjects\Masterarbeit\data\SH_neu\line_new\osm_str_choice_clip
.shp'
    raw_lines = dl.read_shape(lines_path)

        df_lines = raw_lines
    #
    print('Import lines')
    print('Rows: {} Columns {}'.format(df_lines.shape[0],
df_lines.shape[1]))
    print('Pfad lines: {0}'.format(lines_path))
    basic_logger.info('Import lines')
    basic_logger.info('Rows: {} Columns {}'.format(df_lines.shape[0],
df_lines.shape[1]))
    basic_logger.info('Pfad lines: {0}'.format(lines_path))
    result_dict['Anzahl Linien OSM'] = df_lines.shape[0]
    #
    points_path =
r'E:\PythonProjects\Masterarbeit\data\SH_neu\point\K_data_choice.shp'
    raw_points = dl.read_shape(points_path)

    df_points = raw_points
    #
    print('Import rows')
    print('Rows: {} Columns {}'.format(df_points.shape[0],
df_points.shape[1]))
    print('Pfad points: {0}'.format(points_path))
    basic_logger.info('Import rows')
    basic_logger.info('Rows: {} Columns {}'.format(df_points.shape[0],
df_points.shape[1]))
    basic_logger.info('Pfad points: {0}'.format(points_path))
    result_dict['Anzahl Punkte TECH_ID'] = df_points.shape[0]
    #
    return df_lines, df_points

def clean_import_data(df_points):
    # 2. Optimierung: (Optional) bereinige oder transfomiere Daten
    # hausgenaue Punkte
    df_points_cleaned = df_points.loc[df_points['TECH_ID'] != 0]
    df_points_cleaned = df_points_cleaned.dropna(subset=['TECH_ID'])
    #
    print('Bereinigung der hausgenauen Punkte')
    print('Rows: {} Columns {}'.format(df_points_cleaned.shape[0],
```

```
df_points_cleaned.shape[1]))
    basic_logger.info('Bereinigung der hausgenauen Punkte')
    basic_logger.info('Rows: {} Columns
{}'.format(df_points_cleaned.shape[0], df_points_cleaned.shape[1]))
    result_dict['Punkte nach Bereinigung TECH_ID'] =
df_points_cleaned.shape[0]

    # Optimierung: Sortieren der technischen ID
    # Teilfrage je nachdem ein- oder auskommentieren
    #print("Sortierung der Punkte nach TECH_ID...")
    #df_points_cleaned.sort_values(by='TECH_ID', ascending=True,
inplace=True, na_position='last')
    #print("Sortierung abgeschlossen.")
    #
    return df_points_cleaned

def run_find_relevant_lines(k_1, df_points_cleaned, df_lines,
saveFilepath):
    # 3. Bestimme k naechste streckentreue Linien zu hausgenauen Punkten
    # durch Berechnen der euklidischen Distanz
    # Der k Parameter entscheidet über die Menge der
    # k Distanz
    #
    print('Bestimme {} nächste Linien zu allen Punkten'.format(k_1))
    basic_logger.info('Bestimme {} nächste Linien zu allen
Punkten'.format(k_1))
    #
    df_points_cleaned.loc[:, 'nearest_k_lines'] =
df_points_cleaned['geometry'].apply(
        lambda x: get_min_distance_line_id(x, df_lines, k_1))

    # 3.1
    # alle_k_nearest_lines = alle unique k_nearest_lines von allen
punkten
    print('Finde relevante Linien...')
    basic_logger.info('Finde relevante Linien...')
    #
    df_relevant_lines =
extract_unique_lines_from_column(df_points_cleaned.loc[:,
'nearest_k_lines'])
    print('{} von {} Linien als relevant
gefunden.'.format(df_relevant_lines.shape[0], df_lines.shape[0]))
    basic_logger.info('{} von {} Linien als relevant
gefunden.'.format(df_relevant_lines.shape[0], df_lines.shape[0]))
    result_dict['Anzahl relevanten Linien OSM'] =
df_relevant_lines.shape[0]

    #
    filepath = os.path.join(saveFilepath,'relevant_lines.shp')
    df_relevant_lines.to_file(driver='ESRI Shapefile', filename=filepath)

    # 3.2
    # Wandle die relevanten Linien in Punktkoordinaten um
    gpd_lines = df_relevant_lines.loc[:, ['osm_id_new', 'geometry']]
    gpd_lines['points'] = gpd_lines.apply(lambda l:
```

```
get_points_from_line(l['geometry']), axis=1)
    #
    return gpd_lines

def import_unlabeled_point_data():
    # 3.2.1 Lade Punktdaten aus bearbeiteten Shapefile
    print('Lade Punktdaten aus bearbeiteten Shapefile...')
    basic_logger.info('Lade Punktdaten aus bearbeiteten Shapefile...')
    #
    filepath =
r'E:\PythonProjects\Masterarbeit\data\SH_neu\Extrahiert_Punkte_join_osm_c
hoice_clip.shp'
    print('Pfad: {0}'.format(filepath))
    basic_logger.info('Pfad: {0}'.format(filepath))
    #
    gpd_points_from_lines_loaded = dl.read_shape(filepath)
        print('{0} Punkte aus Shapefile
geladen.'.format(gpd_points_from_lines_loaded.shape[0]))
    basic_logger.info('{0} Punkte aus Shapefile
geladen.'.format(gpd_points_from_lines_loaded.shape[0]))
    result_dict['Anzahl Punkte OSM extrahiert 10m'] =
gpd_points_from_lines_loaded.shape[0]
    #
        # 3.3
    # Erzeuge numpy-Arrays für späteren Aufruf des Modells und
Geodataframe
    XY_coords_points_from_lines, X_usage, Y_usage,
OSM_ID_points_from_lines, gpd_points_from_lines =
transform_data_from_point_line_shape2(
        gpd_points_from_lines_loaded)
    #
    return XY_coords_points_from_lines, gpd_points_from_lines

def train_model(k, metric, p, algorithm, df_points_cleaned,
saveFilepath):
    # 4. Modell trainieren
    print('Starte Modelltraining...')
    basic_logger.info('Starte Modelltraining...')
    #
    # 4.1 k-Parameter für das Modell und Trainingsgröße setzen
    validation_size = 0.20
    print('Validation size = {0:.2%}'.format(validation_size))
    basic_logger.info('Validation size =
{0:.2%}'.format(validation_size))
    #
    # Zufaelige Aufteilung mit gleichen Daten
    # Soll das ziehen der Daten wiederholbar machen
    # k Algorithm
    seed = 7
    print('k={0}'.format(k))
    basic_logger.info('k={0}'.format(k))
    #
    # 4.2 Gelabelte Punktdaten für das Modelltraining in Numpy-Arrays
transformieren
    # X= Koordianten der Punkte, Y = C_LINE_ID
```

```
    gpd_points = df_points_cleaned.loc[:, ['TECH_ID', 'geometry']]
    X, Y, gpd_all_points = transform_training_data(gpd_points)
    print('Gesamtmenge aller gelabelten Daten:
{0}'.format(gpd_all_points.shape[0]))
    basic_logger.info('Gesamtmenge aller gelabelten Daten:
{0}'.format(gpd_all_points.shape[0]))
    result_dict['Summe gelabelte Daten'] = gpd_all_points.shape[0]
    #
    # 4.3 Train, Validation Split der Daten
    X_train, X_validation, Y_train, Y_validation = train_test_split(X, Y,
test_size=validation_size, random_state=seed)
    gdf_train = gpd.GeoDataFrame({'X': X_train[:, 0], 'Y': X_train[:, 1],
'TECH_ID_TRUE': Y_train}, crs=crs)
    gdf_validation = gpd.GeoDataFrame(
        {'X': X_validation[:, 0], 'Y': X_validation[:, 1],
'TECH_ID_TRUE': Y_validation}, crs=crs)
    print('Trainingsdatenset besteht aus {0}
Punkten.'.format(gdf_train.shape[0]))
    print('Validationdatenset besteht aus {0}
Punkten.'.format(gdf_validation.shape[0]))
    basic_logger.info('Trainingsdatenset besteht aus {0}
Punkten.'.format(gdf_train.shape[0]))
    basic_logger.info('Validationdatenset besteht aus {0}
Punkten.'.format(gdf_validation.shape[0]))
    result_dict['Anzahl Punkte im Training'] = gdf_train.shape[0]
    result_dict['Anzahl Punkte für Validierung'] =
gdf_validation.shape[0]

    #
    # 4.4 Modell Anwenden
    knn = KNeighborsClassifier(n_neighbors=k, algorithm=algorithm,
metric=metric, p=p, metric_params=None)
    knn.fit(X_train, Y_train)
    print('Start der TECH_ID Vorhersage auf OSM Points 10m. Mit
Parametern: k Algorithm={0}, algorithm={1}, metric={2}, p={3}'.format(
            k, algorithm, metric, p))
    basic_logger.info('Start der TECH_ID Vorhersage auf OSM Points 10m.
Mit Parametern: k Algorithm={0}, algorithm={1}, metric={2},
p={3}'.format(k, algorithm, metric, p))
    #
    # 5. Modell evaluieren
    print('Starte Modellevaluierung...')
    basic_logger.info('Starte Modellevaluierung...')
    #
    # 5.1 Modell auf Validation Data predictieren lassen
    predictions = knn.predict(X_validation)
    gdf_validation['TECH_ID_PREDICT'] = predictions
    # 5.2 Ergebnis der Validierung in Kennzahlen ausgeben
    print('Modellevalutionskennzahlen:')
    print('k={0}'.format(k))
    print('Modell accuracy: {0:.2%}'.format(accuracy_score(Y_validation,
predictions)))
    print('Confusion Matrix:')
    print(confusion_matrix(Y_validation, predictions))
    print('Classification Report:')
```

```
    print(classification_report(Y_validation, predictions))
    basic_logger.info('Modellevalutionskennzahlen:')
    basic_logger.info('k={0}'.format(k))
    basic_logger.info('Modell accuracy:
{0:.2%}'.format(accuracy_score(Y_validation, predictions)))
    basic_logger.info('Confusion Matrix:')
    basic_logger.info(confusion_matrix(Y_validation, predictions))
    basic_logger.info('Classification Report:')
    basic_logger.info(classification_report(Y_validation, predictions))
    #
    # 5.3 Einzeldaten exportieren
    print('Speichere Trainings und Validierungsergebnisse...')
    basic_logger.info('Speichere Trainings und
Validierungsergebnisse...')
    #
    # Alle Punkte speichern, bei den Validation Daten, die Prediction mit
ausgeben
    # Label in Spalte TECH_ID_TRUE, Modellvorhersage in Spalte
TECH_ID_PREDICT
    result = pd.merge(gpd_all_points, gdf_validation, how='left',
left_on=['X', 'Y'], right_on=['X', 'Y'],
                      suffixes=('_train', '_validate'))
    #
    excelFilename = 'result_{0}_{1}_{2}_{3}.xlsx'.format(k, metric, p,
algorithm)
    #
    filepath = os.path.join(saveFilepath, excelFilename)
    print('Pfad: {0}'.format(filepath))
    basic_logger.info('Pfad: {0}'.format(filepath))
    #
    result.to_excel(filepath)
    #
    return k, metric, p, algorithm, accuracy_score(Y_validation,
predictions), knn

def use_model(knn, XY_coords_points_from_lines, gpd_points_from_lines,
saveFilepath):
    # 6. Anwendung des Modells auf Punkte der relevanten Linien
    print('Anwendung des Modells auf nicht gelabelte Daten...')
    basic_logger.info('Anwendung des Modells auf nicht gelabelte
Daten...')
    #
    predicted_line_points = knn.predict(XY_coords_points_from_lines)
    gpd_points_from_lines['TECH_ID_PREDICT'] = predicted_line_points
    print('Modell auf {0} Punkte der Linien
angewendet.'.format(gpd_points_from_lines.shape[0]))
    basic_logger.info('Modell auf {0} Punkte der Linien
angewendet.'.format(gpd_points_from_lines.shape[0]))
    #
    # 6.1 Ergebnis der Einzeldaten speichern
    # Als Excel
    print('Speichere Ergebnis...')
    basic_logger.info('Speichere Ergebnis...')
    #
    filepath = os.path.join(saveFilepath,'Predicted_TECH_ID.xlsx')
```

```
    #filepath =
r'E:\PythonProjects\Masterarbeit\data\predicted_c_line_id.xlsx'
    print('Pfad: {0}'.format(filepath))
    basic_logger.info('Pfad: {0}'.format(filepath))
    #
    gpd_points_from_lines_2 = gpd_points_from_lines.drop('geometry',
axis=1)
    gpd_points_from_lines_2.to_excel(filepath)
    print('gespeichert')
    basic_logger.info('gespeichert')
    #
    # Auch als ShapeFile speichern
    print('Speichere Ergebnis als Shapefile...')
    basic_logger.info('Speichere Ergebnis als Shapefile...')
    #
    filepath =
os.path.join(saveFilepath,'points_from_lines_with_tech_id.shp')

    print('Pfad: {0}'.format(filepath))
    basic_logger.info('Pfad: {0}'.format(filepath))
    #
    gpd_points_from_lines.to_file(driver='ESRI Shapefile',
filename=filepath)
    return gpd_points_from_lines

def save_use_model_results(gpd_points_from_lines, saveFilepath):
    # 8. Erzeuge Wahrscheinlichkeit für die Klassifikation,
    # indem die Anzahl der unterschiedlichen vorhergesagten C-Line-IDs
ins Verhältnis der Gesamtanzahl der Punkte
    # gesetzt wird
    print('Berechne Wahrscheinlichkeiten der Vorhersage für jede
Linie...')
    basic_logger.info('Berechne Wahrscheinlichkeiten der Vorhersage für
jede Linie...')
    #
    probabilities = gpd_points_from_lines.groupby(['OSM_ID_NEW',
'TECH_ID_PREDICT'])['X'].count().rename("count")
    df_probabilities = pd.DataFrame(probabilities)
    df_probabilities['percentage'] =
probabilities.groupby(level=0).apply(lambda x: 100 * x / float(x.sum()))
    print('Speichere Ergebnis...')
    basic_logger.info('Speichere Ergebnis...')
    #
    filepath =
os.path.join(saveFilepath,'probability_osm_id_new_tech_id.xlsx')
    print('Pfad: {0}'.format(filepath))
    basic_logger.info('Pfad: {0}'.format(filepath))
    #
    df_probabilities.to_excel(filepath)

def create_result_directory(root_filepath, k_distanz, k_algorithm,
metric, p, algorithm,):

    fullpath = os.path.join(root_filepath,
'k_{0}_k_{1}/{2}_{3}/{4}'.format(k_distanz,k_algorithm, metric,p,
```

```
algorithm))
    # Create target directory & all intermediate directories if don't
exists
    if not os.path.exists(fullpath):
        os.makedirs(fullpath)
        print("Directory ", fullpath, " created. ")
        basic_logger.info("Directory ", fullpath, " created. ")
    else:
        print("Directory ", fullpath, " already exists.")
        basic_logger.info("Directory ", fullpath, " already exists.")
    return fullpath

def getRelevantPointsFromUnlabeledData(gpd_points_from_lines, gpd_lines):
    #Finde relevante Punkte aus den unlabeled Punkten zu den relevanten
Linien
    print('Entferne nicht relevante Punkte...')
    print('Anzahl Punkte vorher:
{0}'.format(gpd_points_from_lines.shape[0]))
    print('Columns: {0}'.format(gpd_points_from_lines.columns))
    print(gpd_points_from_lines.head())
    basic_logger.info('Entferne nicht relevante Punkte...')
    basic_logger.info('Anzahl Punkte vorher:
{0}'.format(gpd_points_from_lines.shape[0]))
    basic_logger.info('Columns:
{0}'.format(gpd_points_from_lines.columns))
    basic_logger.info(gpd_points_from_lines.head())
    #
    gpd_lines.rename(columns={'osm_id_new': 'OSM_ID_NEW'},
                     inplace=True)
    #
    df_result =
gpd_points_from_lines.loc[gpd_points_from_lines['OSM_ID_NEW'].isin(gpd_li
nes['OSM_ID_NEW']),:]
    print('Anzahl Punkte nachher: {0}'.format(df_result.shape[0]))
    print('Columns: {0}'.format(df_result.columns))
    print(df_result.head())
    basic_logger.info('Anzahl Punkte nachher:
{0}'.format(df_result.shape[0]))
    basic_logger.info('Columns: {0}'.format(df_result.columns))
    basic_logger.info(df_result.head())
    result_dict['Anzahl ungelabelte Daten'] = df_result.shape[0]
    return df_result


def run():

    df_lines, df_points = import_data()

    result_dict['Datum'] = datetime.datetime.today().strftime('%Y-%m-%d')

    df_points_cleaned = clean_import_data(df_points)

    k_1_array = [1,3,5,7,9]

    k_array = [1,3,5,7,9]
```

```
    p_array = [1,2]

    metric_array = ['minkowski']
    algorithm_array = ['auto']
    results = {}
    overall_index = 1
    for index_k_1, k_1_value in enumerate(k_1_array):
        for index, k_value in enumerate(k_array):
            for index_metric, metric_value in enumerate(metric_array):
                for index_algorithm, algorithm_value in
enumerate(algorithm_array):
                    for index_p, p_value in enumerate(p_array):
                        result_dict['Index'] = overall_index
                        result_dict['k Distanz'] = k_1_value


                        result_dict['k NN Algorithm'] = k_value
                        result_dict['KNN Algorithm'] = algorithm_value
                        result_dict['Distanz Metric'] = metric_value
                        result_dict['Parameter Minkowski Metric p='] =
p_value
                        result_dict['Distanz Name'] = "manhattan" if
p_value == 1 else "euclidean"



                        save_file_path =
create_result_directory('E:\PythonProjects\Masterarbeit\data\SH_result_ne
u', k_1_value, k_value, metric_value, p_value,

algorithm_value)

                        gpd_lines =
run_find_relevant_lines(k_1=k_1_value, df_lines=df_lines,
df_points_cleaned=df_points_cleaned, saveFilepath=save_file_path)


                        k, metric, p, algorithm, accuracy_score, knn =
train_model(k_value, metric_value, p_value, algorithm_value,
df_points_cleaned, save_file_path)
                        result_dict['Modell Genauigkeit %'] =
accuracy_score

                        XY_coords_points_from_lines,
gpd_points_from_lines = import_unlabeled_point_data()

                        gpd_points_from_lines_filtered =
getRelevantPointsFromUnlabeledData(gpd_points_from_lines, gpd_lines)

                        #hole nochmal den XY_coordinaten Vektor
                        # Erzeuge numpy-Arrays für späteren Aufruf des
Modells und Geodataframe
                        XY_coords_points_from_lines =
```

```
get_XY_coords_from_filtered_data(gpd_points_from_lines_filtered)

                        gpd_points_from_lines = use_model(knn,
XY_coords_points_from_lines,
gpd_points_from_lines_filtered,save_file_path)

save_use_model_results(gpd_points_from_lines,save_file_path)
                        results[overall_index] = [k, metric, p,
algorithm, accuracy_score]


                        log_header=''
                        log_row=''

                        for x in result_dict:
                            if overall_index == 1:
                                log_header = log_header + str(x) + ';'
                                log_row = str(result_dict['Index'])+';'+
str(result_dict['Datum']) + ';' + str(result_dict['Anzahl Linien
OSM'])+';'+str(result_dict['Anzahl Punkte TECH_ID'])+';'+
str(result_dict['Punkte nach Bereinigung TECH_ID'])+';'+
str(result_dict['k Distanz'])+';'+ str(result_dict['Anzahl relevanten
Linien OSM'])+';'+str(result_dict['Anzahl Punkte OSM extrahiert
10m'])+';'+ str(result_dict['Anzahl Punkte im
Training'])+';'+str(result_dict['Anzahl Punkte für
Validierung'])+';'+str(result_dict['Summe gelabelte
Daten'])+';'+str(result_dict['Anzahl ungelabelte
Daten'])+';'+str(result_dict['k NN Algorithm'])+';'+str(result_dict['KNN
Algorithm'])+';'+str(result_dict['Distanz
Metric'])+';'+str(result_dict['Parameter Minkowski Metric
p='])+';'+str(result_dict['Distanz Name'])+';'+str(result_dict['Modell
Genauigkeit %'])
                        summary_logger.info(log_row)

                        overall_index += 1

    print('index;k;metric;p;algorithm;accuracy_score')




    #
    for key in results:
        print('{0};{1};{2};{3};{4};{5}'.format(key,results[key][0],
results[key][1], results[key][2], results[key][3], results[key][4]))
        print('Berechnung abgeschlossen')
    basic_logger.info('Berechnung abgeschlossen')
run()

dataloader.py:
import geopandas as gpd

def read_shape(filename):
    return gpd.read_file(filename)
```