Multi-target regression with spatially distributed data

Prediction of coordinates based on Finnish street addresses

Noora Backlund

Master's thesis
August 2024
Master of Engineering, Artificial Intelligence and Data Analytics

jamk

Description

Backlund, Noora

Multi-target regression with spatially distributed data: prediction of coordinates based on Finnish street addresses

Jyväskylä: Jamk University of Applied Sciences, August 2024, 113 pages

Degree Programme in Artificial Intelligence and Data Analytics. Master's thesis.

Permission for open access publication: Yes

Language of publication: English

Abstract

Automations are increasingly used in decision making processes, reducing the need for manual work and improving throughput times for business processes. As not all components of the data are always initially available for the processes, machine learning may be used to estimate some of the data to allow a broader scope of data to be entered into automation routines. Finnish building location data and road geometry data were used to investigate whether new, previously unknown addresses could be reliably geolocated by predicting the latitude and longitude coordinates based on lemmatized and vectorized postal address information. Additional assessments were made on whether the model would be capable of indicating the reliability of the prediction, and whether the predictions overall were accurate enough to be used as a part of business processes. The Finnish building data was spatially sampled to relatively evenly cover the Finnish geographic boundaries and was enriched with addresses generated from road network endpoints. The street names were split into prefix and suffix portions, and lemmatized and vectorized using pre-built models by TurkuNLP group. Random Forest Regressor, XGBoost with Random Forest Regression, and Support Vector Regression algorithms were explored for training models capable of reliably predicting the latitude and longitude coordinates of a given street address. Bootstrapping options, number of estimators and maximum tree depths were optimized for Random Forest Regressor, with the most optimal model coupled with forestci module to provide prediction accuracy estimations to further rule out inaccurate predictions. XGBoost model tuning explored estimator numbers, subsampling rates and maximum depths. Support Vector Regression turned out to be too resource-intensive with the complex dataset to be able to produce a trained model within 20 hours of runtime. The highest accuracy model was a 300-estimator Random Forest Regressor model with a maximum tree depth of 54 levels, coupled with forestci module to discard predictions which were in the top 5th percentile in variance of individual tree-level predictions. The model was capable of geolocating unknown addresses to 712 meters from their actual locations on average, with 80 % of addresses geolocated within 1000 meters of the true location. The model did still produce a small number of wildly inaccurate locations, furthest of which were over 300 kilometers away from the true location. Random Forest Regression can be used to automate address geolocation for processes which do not require high precision, though an additional layer of sanity checks should be built on top of the model to validate the results.

Keywords/tags (subjects)

Machine learning, artificial intelligence, spatial data, geographic information systems, natural language processing, lemmatization, vectorization, supervised learning, multi-target regression

Miscellaneous (Confidential information)

No confidentiality requirements.

Contents

Figures

1 Introduction

The collection and use of data has increased exponentially over the past decade. Taylor (2023) estimates that global data generation is expected to reach over 180 zettabytes by 2025, a significant increase from 15.5 zettabytes in 2015. The availability of drives the automation of processes across industries, including processes utilizing geographical information. As more and more processes rely on fully automated workflows, missing data limits the effectiveness of the automations, and in some cases creates manual processes for the cases that the automations cannot handle.

The Finnish telecommunications operator Elisa is no exception to the trend of prevalent automated processes. Multiple processes at Elisa utilize spatial information in automations, with some processes requiring a human touch to account for cases of missing spatial information. In many cases, predicting the missing spatial information based on other accompanying information is logically possible, and the results would be usable in automated processes even if the predicted results are inaccurate within a certain tolerance. This study aims to evaluate whether a machine learning model can:

1. Predict coordinates based on known street address information.
2. Provide consistently reliable, usable coordinate predictions.
3. Evaluate the likely accuracy of the prediction, indicating whether the resulting coordinates are likely to be useful.

If successful, the model would be able to fill in gaps of missing data in various automation processes across Elisa. The increased level of automation would in turn reduce the need for manual work, allowing for specialists to focus on more complex tasks, instead of routine data maintenance.

2 Research objectives

2.1 Research purpose

Artificial intelligence has established a solid foothold in different industries, creating predictions and categorizing data. Spam filters categorize whether content is valid or spam, recommendation lists attempt to predict similar content that a user of the service would be interested in, and search engines utilize artificial intelligence to improve search results. These use cases tend to utilize datasets that usually do not have a spatial (nor a temporal) dimension, and the models typically do not consider spatial relationships or distribution explicitly.

Machine learning methods have commonly been used in conjunction with spatially distributed data to create blanket predictions across geographical regions, such as soil type predictions projected on a map, based on individual soil sample measurement points across a geographic region. Based on previous research, spatial machine learning methods tend to produce descriptive values for a given spatial location, instead of predicting a spatial location given descriptive statistics.

The academic purpose of this research is to bridge the gap between the spatial and non-spatial approaches by utilizing techniques from spatial machine learning to enhance a more traditional predictive machine learning model. This research also aims to utilize natural language processing techniques to improve predictions where possible.

The business goal for this research is to create a model that can be reliably used to feed in data to business automations. The impact will vary across different use cases, but the ability to automatically generate accurate data for automations will significantly reduce manual work and time-delay in business processes.

2.2 Research objectives

The objective of the research is to create a machine learning model capable of reliably predicting coordinates for a given Finnish street address, such as “Brahenkatu 20, 20100 Turku”. To distinguish the research objective from a simple address registry lookup, the model should also be able to estimate the location of a completely new address not contained within the training dataset.

As Finland is a bilingual nation with Finnish as the major language and Swedish as the minor language, the dataset is bound to contain both Finnish and Swedish entries. Ignoring one or the other is not possible in this case, as some dataset entries are bound to be exclusively in Finnish or Swedish. The objective is to create a model that can predict coordinates across the entire Finnish geographic area, therefore both the Finnish and Swedish entries in the dataset must be included and the language difference accounted for.

The model training process should also consider the spatially distributed nature of the dataset and utilize lessons from spatial machine learning to improve the model prediction outcomes. Ignoring the spatial features of the dataset may introduce bias to the model and should therefore also be addressed.

As the prediction results are to be used in automations, the automation should also have an idea of how accurate the prediction is likely to be. Thus, the model needs to be able to evaluate the accuracy of its prediction and communicate the accuracy alongside the prediction. Alternatively, a separate sanity check could be utilized to evaluate the prediction.

2.3 Research questions

The research objectives can be condensed into three research questions, defined as follows:

1. What steps and methods are required to create a machine learning model capable of predicting latitude and longitude coordinates from a given Finnish street address?
2. How accurate are the predictions of the model?
3. How well can the model evaluate the accuracy of its prediction?

The answer to the first research question is expected to be objective, as the required steps and methods will vary based on individual circumstances and researcher’s abilities. The second and third research questions, however, can be evaluated mathematically to provide clear-cut answers.

3 Research method

3.1 Design Science Research Process

Design Science Research Process (DSRP) was introduced by Peffers et al. (2020) to formalize a structured methodology that can be applied to research on information systems in a serviceable and flexible manner. The authors drew from previous research into Design Science Research based methodologies commonly used in the information systems field, including system development research methodology by Nunamaker et al. (1991), information system design theory by Walls et al. (1992), as well as engineering process models by Eekels et al. (1991). Consistency with previous research was a priority for Peffers et al. (2020), and the methodologies heavily influenced the creation of DSRP.

Peffers et al. (2020) expected DSRP to benefit researchers, publishers and information system professionals alike. DSRP offers researchers a structured process to conduct research in a manner that ensures the research is both rigorous and applicable. Editors and reviewers in the publication community would have a model with which they can frame information science related research, and through which they can recognize the research goals, processes and results. Professionals would also benefit from the methodology, as information science related research is more applicable and accessible with DSRP.

3.2 Description of methodology

Peffers et al. (2020) outlined three objectives for DSR methodology: the methodology should a) be consistent with design science processes across disciplines, b) provide a process for how to conduct research, and c) describe a model for the desired design science research output. The authors compared and contrasted methodologies by previous researchers, creating an ensemble of actions that is both consistent with the prior research, and fulfills their objectives for the DSR methodology.

As a result, the authors proposed a methodology consisting of the following six activities:

1. problem identification and motivation,

2. objectives of a solution,
3. design and development,
4. demonstration,
5. evaluation, and
6. communication.

Figure 1 below shows the DSRP, interaction between the steps in the process, as well as the requirements for proceeding to the next steps.

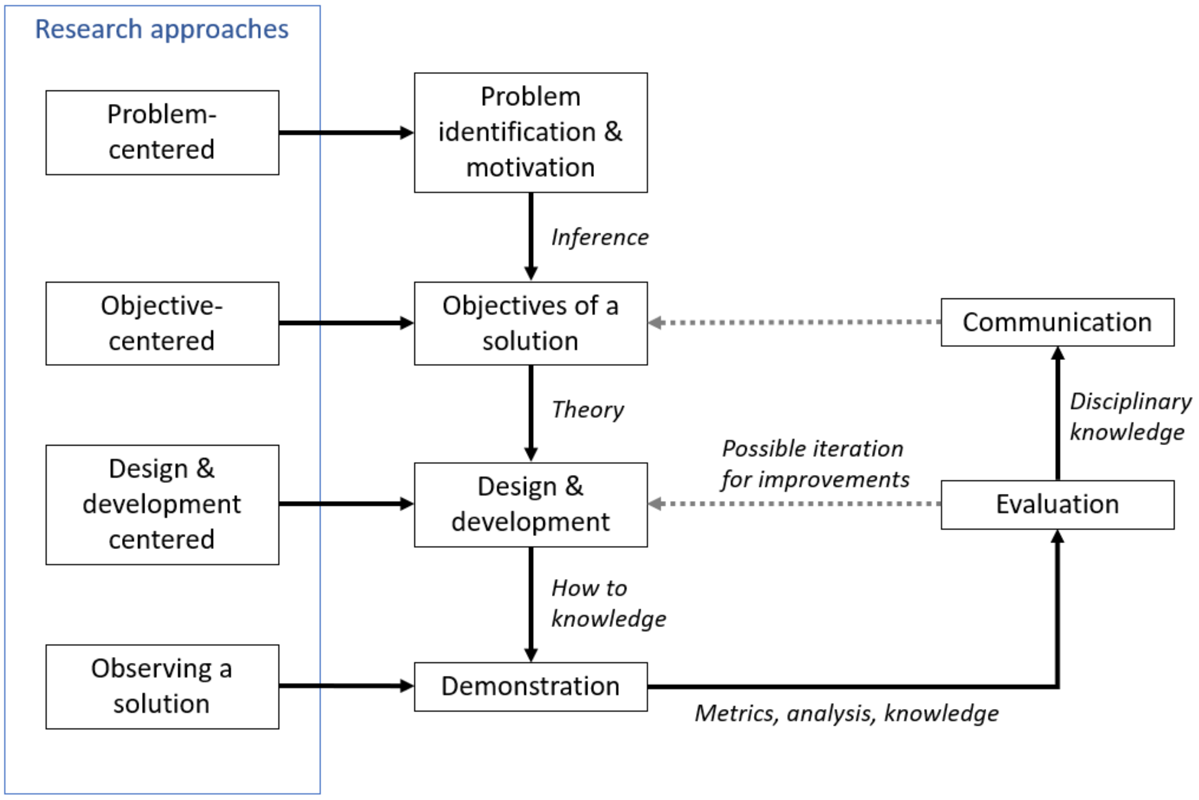


Figure 1. Design science research process model. Adapted from ”Design Science Research Process: A Model for Producing and Presenting Information Systems Research”, by K. Peffers, 2020, *Proceedings of the First International Conference on Design Science Research in Information Systems and Technology*, p. 93. Copyright 2020 by K. Peffers.

While objectives of a solution and communication were only proposed by two of the examined seven previous methodologies, the remaining steps were present in all or almost all methodologies. A summary of each of the six steps is presented below.

Problem identification and motivation

The first step is to describe out *what* needs to be solved and *why*. Describing the research problem (the “*what?*”) creates a specification that the solution is required to meet., while the justification (the “*why?*”) highlights the significance of the task at hand. The completion of this step requires a solid understanding of the nature of the problem and its implications.

Peffers et al. (2020) suggest breaking down the problem into individual components if required, so that the solution can account for the complex nature of the initial problem. Understanding the motivation behind solving the problem also pushes the researcher and audience towards finding and accepting a solution for the problem.

Objectives of a solution

The second step consists of defining what is needed to solve the problems outlined in the previous step. Peffers et al. state that these objectives can be either qualitative or quantitative and should clearly identify the conditions in which the solution can be considered successful. In an ideal situation, the objectives should be derived directly from the research problems from the previous step. The objectives of the solution can only be defined with sufficient understanding of the problems, their current solutions and efficacy.

Design and development

The creation of the solution happens at the design and development stage. Theoretical knowledge on the problem and its potential solutions are applied to practice: the solution’s desired functionality is specified, and its architecture planned. Finally, the solution itself is built according to these specifications. The solution may be, for example, a model, a method or a construct. (Peffers, et al., 2020.)

Demonstration

Demonstration step focuses on measuring the effectiveness of the solution. Peffers et al. (2020) suggest evaluating the solution though activities such as experimentations, simulations or case studies. The evaluation activities produce data, which can be further analyzed in the next step, especially in comparison to the solution definitions derived in the second step of the methodology.

Evaluation

Evaluation consists of comparing the measurements from the demonstration to the expected solutions that would solve the problems at the heart of the research. The specific evaluation techniques vary based on the nature of the research, and the researcher must have a good understanding of relevant analysis techniques and metrics. The evaluation should show whether the metrics produced by the solution meet the demands placed for the solution at the second step ("Objectives of a solution"). (Peffers et al., 2020.)

Depending on the setup of the research, the researcher may choose to return to the "Design and development" step to further improve the solution. If iteration is infeasible, or an effective solution has been obtained, the researcher can proceed to the final step: communication.

Communication

The final step, communication, commits the knowledge obtained through the research process into the common knowledge pool. Peffers et al. (2020) suggests using the structure of the DSRP to also structure a research paper, as the structure also follows the common structure for empirical research papers. The authors suggest that the communication should include the problem, motivation for solving the problem, the solution, the utility of the solution, the solution design, and effectiveness of the solution for researchers, practicing professionals and other audiences.

Research approaches

Peffers et al. (2020) identify four approaches towards applying DSRP, each targeting a different step as the point-of-entry to the process. A problem-centered approach starts at the first step of the process, with observations of a problem or suggestions of further research as the common initiators. On the other hand, objective-centered approaches may have the initial knowledge of the exact specifications for solving the problem at hand and start the application of DSRP at the second step. Objective-centered approaches may well be the result of consulting events, for example. Design and development centered approaches start with a potential solution already at hand, at the third step of the process, and work backwards in the DSRP to identify potential problems that could be mitigated with the solution at hand. The last approach, observing a solution, may happen in the case where a solution has already been produced and applied for a specific problem, and the DSRP is backtracked to formalize and structure the solution according to scientific standards. In this case, the DSRP starts at the fourth step ("demonstration"), and the previous steps are considered with the specific solution in mind.

3.3 Design Science Research guidelines

One of the previous studies that Peffers et al. (2020) examined was a paper by Hevner et al. (2004). Even though the essence of the proposed methodology has been incorporated into the DSRP, their guidelines also deserve special attention. In brief terms, the authors outline the following guidelines for design science research:

- **Design as an Artifact:** the solution must be a viable model, construct, method, or similar.
- **Problem Relevance:** the solution should be technology-based and should solve an important business problem.
- **Design Evaluation:** the effectiveness and quality of the solution must be shown effectively and rigorously.
- **Research Contributions:** the contribution of design science research to the implementation of the solution, its architecture or foundations must be clear.
- **Research Rigor:** rigorous methods must be used in the construction and evaluation of the solution.

- **Design as a Search Process:** available methods and paths should be investigated when searching for the best solution to the problem.
- **Communication of Research:** the communication related to the research should be approachable by both technology-oriented and management-oriented readers.

These guidelines are complementary, and partially included in, the DSRP proposed by Peffers et al. (2020). While the DSRP outlines the steps of the process, the guidelines provide further instructions on how to complete the steps in the DSRP.

3.4 Application of methodology

I utilized the DSRP throughout the implementation to guide and structure the research process. I chose DSRP to serve as the methodology for this research, because the methodology is specifically targeted for research activities in the information systems field. The methodology places importance on the solution of a business problem with technology and communication with both the technology and business sides of the audience, as highlighted by Hevner et al. (2004) in their guidelines. The initiator, the stakeholder and the end-user for the results of this study is the business organization, therefore a methodology that accounts for the importance of the business application of the results seems most suitable for the purpose.

The research approach in the context of DSRP is ambiguously somewhere between **objective-centered** and **design & development centered** approaches. The initial research problem has existed for quite a while, but the initial steps in this research process quickly led to both the specific objectives that a solution must meet as well as a potential solution itself. In the context of this methodology, the potential solution is a supervised machine learning model capable of producing coordinate predictions. Due to the ambiguity in the starting approach towards the research, I treated the process as an objective centered DSRP for which I already had a preferred solution. In order to adhere to the “Design as a Search Process” guideline by Hevner et al. (2004), I have conducted a thorough literature review to identify similar previous implementations, as well as investigated potential non-machine-learning solutions to the problem.

Otherwise, the research process fits squarely into the DSRP model. As building a machine learning model is by its nature an iterative process, the optional step from evaluation stage of the DSRP model back to the design & development stage is traversed quite a few times when seeking for a more fitting and more effective solution. The adaptation of DSRP in the context of my research topic is presented in Figure 2 below.

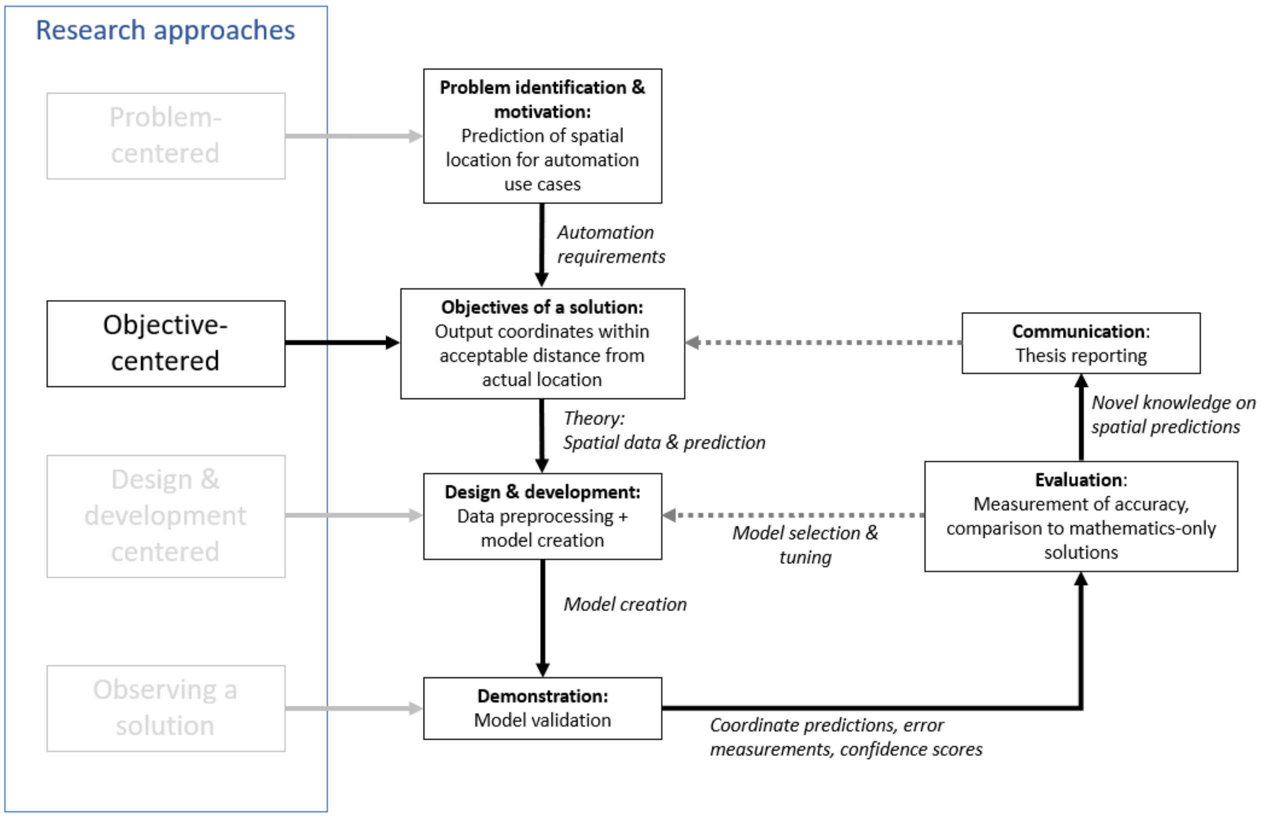

Research approaches
Problem-centered
Objective-centered
Design & development centered
Observing a solution
Problem identification & motivation: Prediction of spatial location for automation use cases
Automation requirements
Objectives of a solution: Output coordinates within acceptable distance from actual location
Theory: Spatial data & prediction
Design & development: Data preprocessing + model creation
Model creation
Demonstration: Model validation
Coordinate predictions, error measurements, confidence scores
Evaluation: Measurement of accuracy, comparison to mathematics-only solutions
Model selection & tuning
Novel knowledge on spatial predictions
Communication: Thesis reporting

Figure 2. Research topic in the context of design science research process.

The specific implementation of the DSRP in the context of this research is as follows:

- **Problem identification & motivation:** Introduction (chapter 1) identifies the research problem, while Research objectives section (chapter 2) expands on the research problem and the motivation.
- **Objectives of a solution**: Research objectives section (chapter 2) also discusses the objectives in further detail.
- **Design & development**: Theoretical background (chapter 4) provides the required theory and previous research on spatial data, machine learning and application implementations,

enabling the informed creation of a machine learning model in Implementation section (chapter 5).

- **Demonstration**: Implementation section (chapter 5) also contains the implementation of an evaluation round, aimed at producing data that can be evaluated to obtain model accuracy and confidence metrics. Notes on iterative rounds between the stages of the DSRP model are also discussed in the Implementation section.
- **Evaluation**: Results section (chapter 6) presents the evaluation and metrics of the final model. Results section also provides a comparison to pure mathematics solution to evaluate the choice of selecting machine learning methods over other possible methods.
- **Communication:** The resulting new knowledge is summarized in the Conclusions section (chapter 7), and the solution efficacy is compared to the objectives for a solution established in Research objectives section (chapter 2).

4 Theoretical background

4.1 Previous research on machine learning with spatial data

4.1.1 Requirements for previous research

Machine learning methods and artificial intelligence has been researched extensively over the past decades, resulting in a vast literary base on the subject matter. The fundamental underlying theories and concepts in machine learning methods also remain valid, meaning that most of the current knowledge base remains useful and usable even over decades. More recent research is likely to introduce novel concepts and improvements on the older methods and should thus be given more weight when applicable. As the topic of machine learning by itself is vast, this overview on previous research focuses on the spatial applications of machine learning methods and attempts to evaluate their usefulness to the research questions at hand.

4.1.2 Contemporary approaches to combining spatial data with machine learning

Perhaps the single most useful source for creating an overview of previous research was a recent article by Nikparvar and Thill (2021), where the authors provided an overview of recent findings in combining machine learning with spatial data. The authors performed exhaustive searches across high-quality geographic information system related journals (among other sources), categorized the articles and summarized the findings on how spatial data properties have been incorporated into machine learning methods. The authors identified two different approaches to accounting for spatial properties: Spatial Observation Matrix creation, and utilization of specific learning algorithms.

The Spatial Observation Matrix approach, in short, consists of finding a representation for spatial properties and including them in an observation matrix, after which typical machine learning methods can be applied. Nikparvar and Thill (2021) list the following spatial aspects as critical: spatial sampling, spatial features, dimensionality reduction and handling missing data. Each of these aspects needs to be reflected in the dataset for the machine learning model to be accurate. Further theoretical background related to spatial sampling, spatial features, dimensionality reduction and missing data handling is discussed in later chapters of this report.

The second approach Nikparvar and Thill (2021) found to combining spatial data and machine learning was to directly incorporate spatial properties with learning algorithms. Some studies incorporated spatial autocorrelation measures at decision tree node-level to account for spatial variability, some introduced geographically weighted versions of learning algorithms to account for spatial properties. Self-organizing maps have been enhanced to account for spatial dependency, radial basis functions combined with semivariogram models, and neural network based models adapted to account for spatial dependencies.

Nikparvar and Thilll (2021) concluded that machine learning models benefit from the inclusion of spatial properties, and the failure to properly include spatial properties will lead to worse training outcomes. The authors also noted that most of the research that covers machine learning with spatial data still fails to properly address the spatial properties, or only deals with them marginally.

4.1.3 Machine learning methods in spatial context

Kopczewska's (2022) recent paper provided another overview of machine learning methods in spatial context, complementing the earlier paper by Nikparvar and Thill (2021) but focusing on spatial clustering specifically. As was the case with the earlier paper, Kopczewska (2022) also identifies the same two ways in which machine learning methods have been combined with spatial data: the use of typical machine learning methods on spatial data, and development of new spatial data specific machine learning methods. The author explores different spatial clustering methods for various real-life applications, from determining catchment areas for services such as schools or post offices, to improving WLAN indoor positioning. Most common machine learning classifier models for data with spatial properties were naïve bayes, K-nearest neighbors, random forests, support vector machines, artificial neural networks, XGBoost and cubist. Recent years have also seen an increase in use of ensemble methods, where multiple machine learning models have been combined to produce a more optimal result. Of the aforementioned methods, random forest was often found to be the best performer.

4.1.4 Preparation of spatial data for machine learning purposes

Kanevski et. al. (2009) authored a book, Machine Learning for Spatial Environmental Data, which despite its age is still a treasure trove of textbook-style information on machine learning applications to spatial data. The book covers the basics on the types of machine learning algorithms generally utilized in spatial data based models, basics of machine learning concepts, methods of exploratory spatial data analysis, data preprocessing techniques, and finally practical geostatistics and artificial neural network applications. The chapters on spatial data exploration and data preprocessing provided valuable insights for the purposes of this research. While thorough and well written, the chapters on specific applications of machine learning methods on spatial data were better suited to different research problems than the one at hand.

In the chapter on exploratory data analysis, Kanevski et. al. (2009) discuss the importance of understanding the data at hand. The authors recommend using science-based models or deterministic models to understand data, though acknowledging that science-based models often contain limitations in incompleteness, complexity, uncertainty and measurement-related variables. The authors also emphasize that spatial data by its nature should be visualized in a geographic information system, as important information can be extracted from the data by just visualizing it. Further methods for spatial exploratory data analysis also include regional exploratory data analyses where individual geographical areas are examined in turn, calculation of moving window statistics, variogram analyses and nearest-neighbor modeling. Kanevski et. al. (2009) also note the importance of understanding the data collection and sampling methods, as data points are often not measured randomly, and may be affected by topography or other variables. The authors outline monitoring network analysis as a tool of analyzing spatial data sampling and identifying clustered data. Random declustering, Voronoi polygons, cell declustering and kriging weights were also introduced as tools to remove bias from data.

4.1.5 Summary of previous research

Even though plenty of studies have been conducted on machine learning with spatial data, none of the research encountered during the literature review was quite like topic of this research. A common application of machine learning to spatial data was the generation of raster maps from individual measurement points (such as prediction of seabed mud content by Li et. al. (2011)), with the most similar study by Chawla et. al. (2001) attempting to predict potential bird nest locations based on geographical features of an area. As a result, previous studies on machine learning applications on data with spatial properties were examined in a wider scope, and potentially relevant parts of the studies collated to form a base on which this research was built.

The search for previous research culminated to two major pieces of research: an article by Nikparvar and Thill (2021), and a book by Kanevski et. al. (2009). The article by Nikparvar and Thill (2021) provided usable formats for approaching the research problems, and specific steps related to spatial data handling to account for when conducting the research. The article was recent and very thorough in covering the research field, and other related publications in combination of spatial data and machine learning had already been covered and summarized by Nikparvar and Thill (2021) in their research. Kanevski et. al. (2009) provided solid theory on preparing spatial data for machine learning methods, complementing and adding to the information gained from the Nikparvar and Thill (2021) article. Kopczewska's (2022) article concentrated on clustering methods that are not directly applicable to the research questions of this paper, but the author's overall findings supported the assertions by Nikparvar and Thill (2021) and provided recommendations on specific machine learning methodologies for spatial data in classification context.

4.2 Spatial data and operations

4.2.1 Geographic information system

Geographic information system (GIS) combines descriptive information with location information. GIS enables mapping and analysis of geographic data, allowing for better understanding of data in a visual context. (What is GIS?, n.d.)

4.2.2 Spatial data

Spatial data is information that describes the location and shape of an object. (Spatial Data Definition, n.d.) Spatial data can be represented in either raster or vector form: raster is a grid of cells with individual values for each individual cell (or pixel), vector is a set of coordinates that list the vertices of the object. (Vector vs Raster in GIS, n.d.) Figure 3 shows the same spatial features represented as raster and vector.

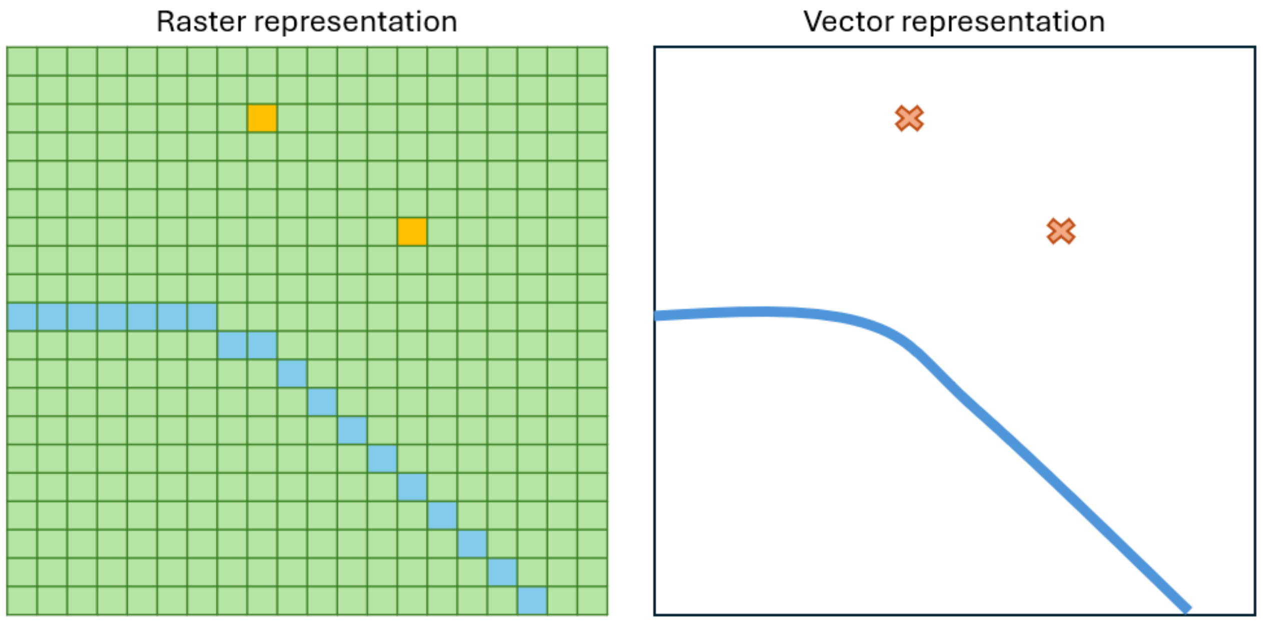


Figure 3: Raster and vector representations of same spatial data objects

Spatial data in vector format usually takes one of the three forms: points, lines or polygons (also known as areas). Point type geometries are represented as X and Y coordinates of the point. Lines consist of a list of X and Y coordinates of the individual vertices of the line. Polygons are structured in the same format as lines, except the line is closed: the first and the last coordinates are the same, forming a closed area within the individual vertices. (Vector vs Raster in GIS, n.d.)

Some information systems also implement more refined versions of vectors than just the three basic forms, but even then, the same building blocks of lines and polygons tend to be used. For example, Oracle differentiates between interior and exterior polygon rings based on a compound element type identifier and the rotation direction of the coordinates, but the actual structure of the polygon still consists of coordinate pairs for which the first and the last coordinates are the same. (Spatial Data Types and Metadata, n.d.)

4.2.3 The EPSG Dataset / Coordinate systems

The EPSG Dataset is a collection of definitions for coordinate reference systems maintained by the Geodesy Subcommittee of the IOGP Geomatics Committee (History of the EPSG Dataset, n.d.). The abbreviation EPSG originally stood for European Petroleum Survey Group, whose first collaborative project was the creation of a common geodetic database to be shared across member organizations. The original European Petroleum Survey Group was disbanded in 2005 and reformed as Geomatics Committee, in charge of maintaining the open dataset of coordinate systems, currently containing over 7000 coordinate reference system definitions. The abbreviation EPSG remained as the brand name and individual coordinate reference systems still carry the abbreviation in their identification code, such as EPSG:3067 for coordinate reference system ETRS89-TM35FIN, or EPSG:4326 for WGS84 (World Geodetic System 1984). EPSG numbers can be used to unambiguously identify and differentiate between coordinate projections.

4.2.4 Map projections

Accurately depicting a round, three-dimensional Earth on two-dimensional media is difficult, and inevitably results in distortion of some parts of the Earths' surface. As Keith Ord (2010) aptly stated in his paper on spatial autocorrelation, "all maps are wrong, but some are useful" (p. 167). GISGeography (2024) describes a projection as a method of transforming the three-dimensional surface of the Earth into a two-dimensional map in a way that fits the purposes of the cartographer. Earth is projected onto developable surfaces – usually cones, cylinders or surfaces – which provide different benefits and limitations for different uses.

Intergovernmental Committee on Surveying and Mapping (n.d.) categorized map projections into four basic types:

- **Equal-area** projections maintain correct sizes for features.

- **Conformal** projections keep the correct shape of the features.
- **Equidistant** projections ensure correct distances between features.
- **True direction** projections maintain consistent directions across features.

A projection is always a compromise between the four types above, especially between equal-area and conformal projections, as these two types cannot both occur in the same projection. Mercator and Robinson projections visualized in Figure 4 demonstrate the dilemma well. The Mercator map created by Geradus Mercator in 1569 is a true direction projection that keeps the correct directions across all features but distorts shapes and sizes progressively closer to the poles. The Robinson is neither an equal-area, conformal, equidistant or a true direction projection, though the distortion of shapes and sizes is considerably less than with Mercator – Denis (n.d.) states that the main purpose of the Robinson projection was to simply create a visually pleasing map of the world.

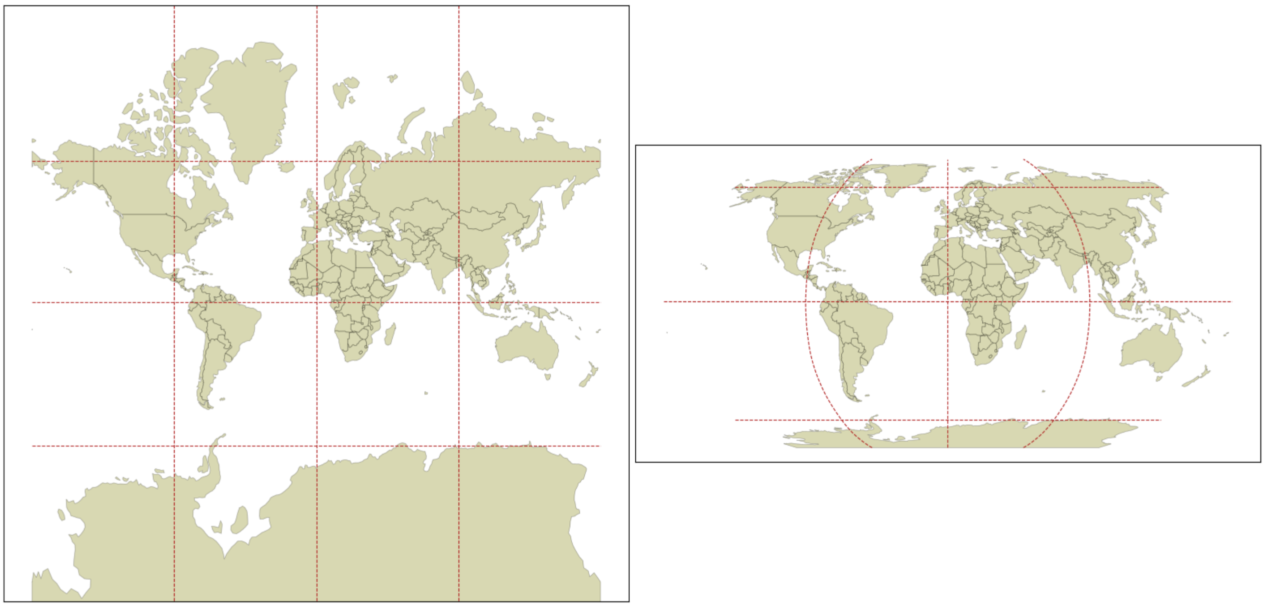


Figure 4. Comparison of Mercator (left) and Robinson (right) projections of the world.

GISGeography (2024) describes that a conic projection preserves the scale of countries, but is inaccurate at projecting the entire map, as the lower areas of the projection are heavily distorted.

Conic projection is also useful for projecting maps that show long east-to-west distances, as the distortion across the distance stays constant. Figure 5 below shows the principle behind conic projection.

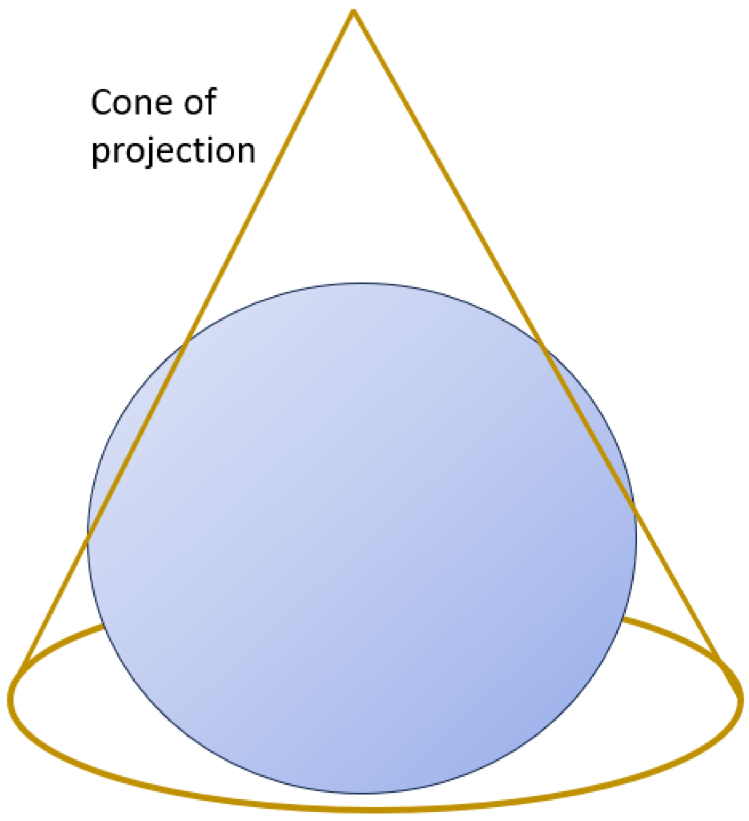


Figure 5. The principle of conic projection

According to a publication by the Finnish Geodetic Institute and the National Land Survey of Finland (n.d.), the ETRS89/TM35FIN projection (denoted as EPSG:3067 in the EPSG coordinate database) is recommended by the Nordic Geodetic Commission for use in the Nordic countries, including the target location of the study, Finland. Figure 6 shows an ETRS89/TM35FIN projection of the world map, with dashed lines indicating the locations of the equator, prime meridian, arctic circle and 90 degree longitudinal lines east and west of the prime meridian.

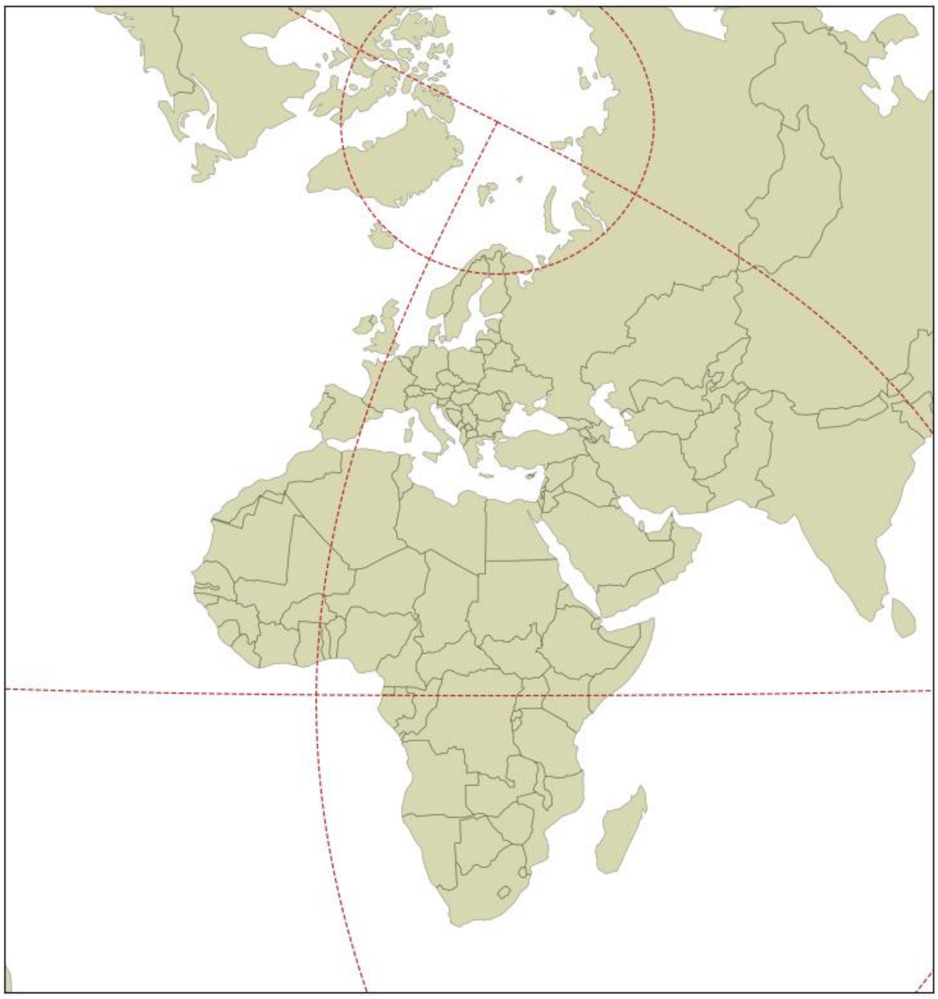


Figure 6. ETRS89 / TM35FIN (EPSG:3067) projection.

4.2.5 Representations of geometric objects

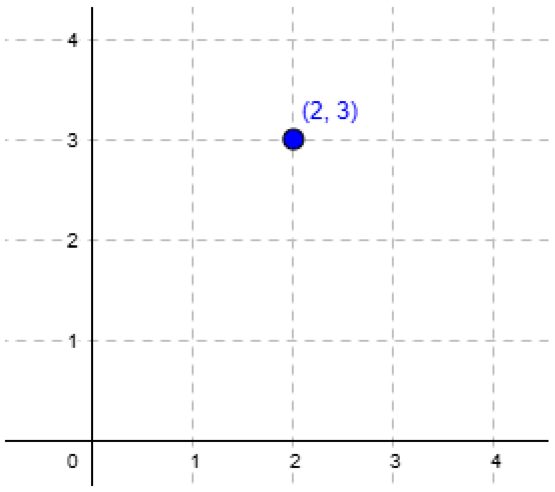
Analysis and calculations with spatial data require systems that are capable of representing spatial information in a structured way, with which mathematical calculations can be performed. Both raster and vector geometries can be used for calculations in different ways, and as this research utilizes vector geometries, the discussion ahead applies to vectors.

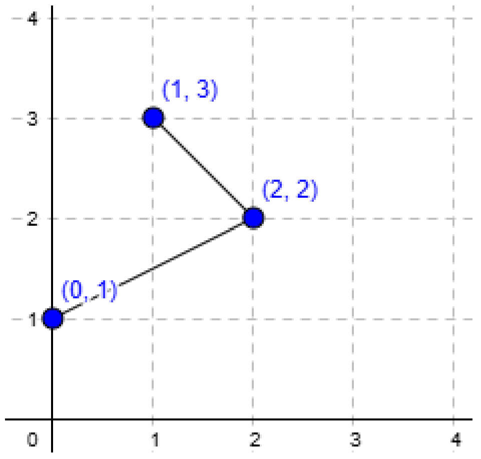
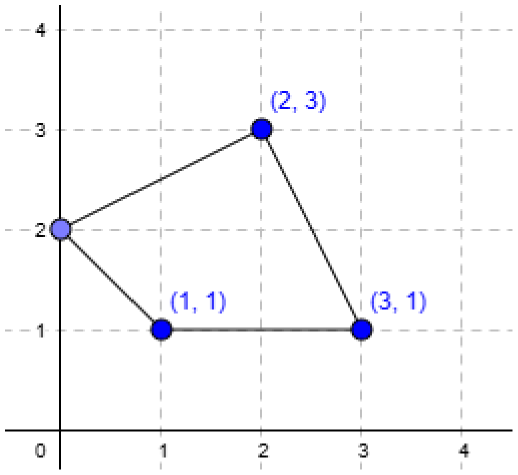
The exact implementations of vector-based geometric objects vary between different geographical systems, but fortunately the Open Geospatial Consortium (OGC) has created a specification, ISO 19125-1, that provides the basic definitions of globally accepted geometry types. The OGC guidelines ensure general compatibility of spatial objects between different geographic systems. (Open Geospatial Consortium, 2011; Rey et. al., 2020.)

The geometry object types relevant to this research are points, lines (or linestrings) and polygons. Rey et. al. (2020) define point geometry as a x and y coordinate pair. A line geometry contains multiple points, whereas a polygon is a surface that consists of one or more lines with the same coordinate as the start and end points. All of these can also be represented as multi-variants, where a multipoint contains more than one point geometry object, multiline more than one line object, and multipolygon more than one polygon. The Open Geospatial Consortium (2011) standard also allows for geometry collections, where a single geometry object can contain any mixture of points, lines and polygons. Table 1 shows different geometry objects as implemented in Oracle's spatial databases. (Spatial Data Types and Metadata, n.d.)

Table 1. Basic geometry types implemented and visualized

Geometry type	**Oracle's implementation**	**Geometry visualization**

Point	SDO_GEOMETRY(
	  2001,
	  NULL,
	  SDO_POINT_TYPE(2, 3, NULL),
	  NULL
	)
		

Line	SDO_GEOMETRY(2002, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 4, 2, 1, 2, 1, 3, 2, 1), SDO_ORDINATE_ARRAY(0, 1, 2, 2, 1, 3))	
Polygon	SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1005, 1, 1, 2, 1), SDO_ORDINATE_ARRAY(1, 1, 3, 1, 2, 3, 0, 2, 1, 1))	

Multi-versions of the basic geometry shapes are represented in Oracle's form by changing the geometry type identifier (first parameter in SDO_GEOMETRY) to suit the geometry contents in the object, appending the geometry types and coordinate point starting indexes to SDO_ELEM_INFO_ARRAY, and adding coordinate points to SDO_ORDINATE_ARRAY.

While individual implementations vary across different systems, the underlying representation of objects as lists of coordinates remains the same. Using representations such as Oracle's SDO_GEOMETRY format, spatial information becomes just another attribute that can be stored in a geographic database alongside non-geographic information. When stored in a database capable of performing spatial operations, the geographic data can be utilized seamlessly with other data attributes in analysis and calculations. (Rey et. al., 2020; Spatial Operators, n.d.)

4.2.6 Spatial queries and relationships

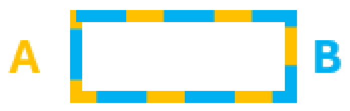
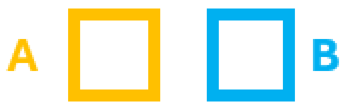
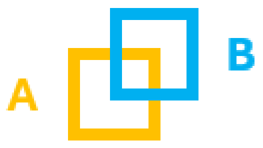
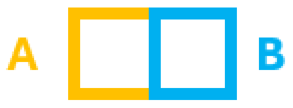
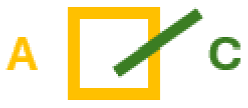
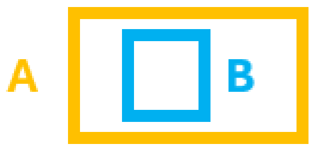
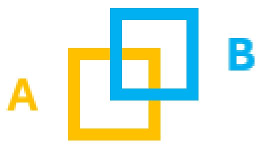
A spatial query searches for spatial objects in a dataset that have a specific spatial relationship with the search object, such as overlapping geography or distance under a given value. As with spatial data types, Open Geospatial Consortium has also standardized spatial queries, allowing different vendors to create their own implementations with specifications documented in ISO 19125-1. (Open Geospatial Consortium, 2011.)

Carniel (2023) divides spatial queries into five categories:

1. topology-based,
2. metric-based,
3. direction-based
4. hybrid, and
5. spatial joins.

Topology-based queries examine the relations between different spatial objects. Topology-based queries do not consider numeric metrics, as the relationships between objects remain unchanged even if the scale or rotation of the objects changes. Different relationships are valid for different combinations of geometry types, with some relationships undefined for certain combinations, such as “crosses” as a relationship between two polygons. The spatial relations defined by Open Geospatial Consortium are listed in Table 2. The relationships in the table are illustrated with two polygons, or where applicable, a line. (Carniel, 2023; Open Geospatial Consortium, 2011; Spatial Relations Defined, n.d.)

Table 2. Topological spatial relationships

Relation name	**Description**	**Visualization**
Equals	A and B have the same boundaries and the same interior.	
Disjoint	A and B are separate.	
Intersects	A and B are not disjoint.	
Touches	A and B share one or more common vertices or edges, but the interiors do not overlap.	
Crosses	C crosses A. Only defined for a line, crossing a polygon or another line.	
Contains Within	A contains B. B is within A.	
Overlaps	Interiors of A and B intersect, A and B are not equal nor contained within one or the other. Only defined for lines and polygons.	

According to Carniel (2023), metric-based queries search for spatial objects in relation to a measurable distance or proximity. The Open Geospatial Consortium (2011) specification defines some methods for spatial analysis (most importantly, "distance" and "buffer"), but does not provide specifications for any metric-based queries, leaving implementations up to different GIS vendors. A common example of a metric-based query is the nearest neighbor search, used to find the closest spatial objects to a given search object, for example the closest supermarket in relation to a person’s location. Figure 7 visualizes a nearest neighbor search performed with search object A, resulting in polygon t1.

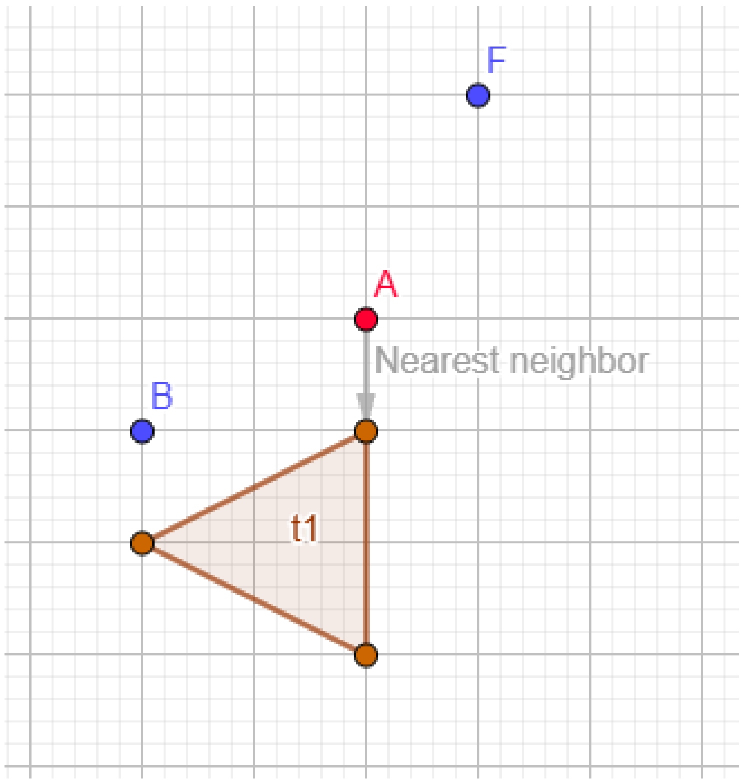


Figure 7. Visualization of nearest neighbor query

Metric-based spatial queries are also often used to search for all spatial objects within a given distance from a search point. Oracle has implemented this as “within distance” (SDO_WITHIN_DISTANCE), which buffers the search object and evaluates relationships between the buffered search object and target spatial objects. A “within distance” query is visualized below in Figure 8, where searching for all points that reside two distance units from point A would result in a dataset containing points B, E and G, with points C, D and F left outside the result dataset.

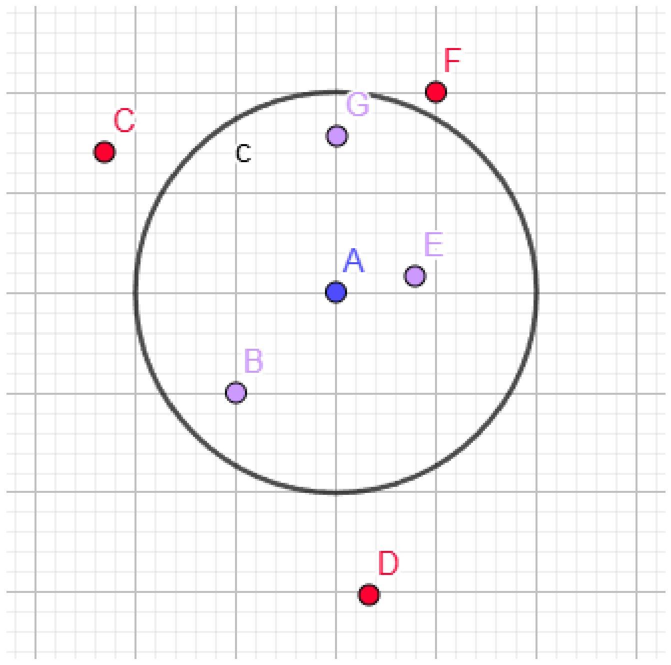


Figure 8. Query of points within two units from point A

Direction-based queries are described by Carniel (2023) as queries that fetch objects located towards a specific cardinal or relative direction from the search object. Direction-based searches seem to be less common than relationship-based and metric-based searches, and at least Oracle's spatial database implementation does not contain built-in functions for direction-based searches. If necessary, lack of directional query implementations can be circumvented by mathematical calculations on relationships between x and y coordinate values.

Carniel (2023) classifies hybrid queries as spatial queries that are formed as a combination of two or more simultaneous queries from the categories above. For example, a Finnish traveler arriving by land to Sweden might be interested in the closest gas station within the Swedish borders. This example query requires a relation-based spatial query, which identifies all gas stations contained within the Swedish borders and combines it with a distance-based nearest-neighbor query with the traveler's coordinates as the starting point.

Spatial join is a method of merging two datasets containing spatial data by matching the rows based on a spatial query. Where in previous categories a single spatial object is compared to a list of potential result objects, a spatial join performs the comparison with all geometries of one dataset to all geometries of a second dataset. The filter function can be separately specified and falls under one of the previous categories of spatial query types. A use case of spatial join would be, for example, to generate a list of municipalities that have lakes, by performing a spatial join on municipality geometry and lake geometry datasets. (Carniel, 2023; Indexing and Querying Spatial Data, n.d.)

4.2.7 Spatial indexing

As spatial comparisons are often more complicated than pure numeric or text-based comparisons, spatial indexes are used to optimize spatial queries. A spatial index is a method or a structure for efficiently retrieving related spatial objects, commonly an R-tree structure. R-tree structures function by grouping spatial objects by minimum bounding boxes – the smallest possible rectangle-shaped object that encompasses the entire spatial structure. Figure 9 displays a spatial object with its minimum bounding box. (Carniel, 2023.)

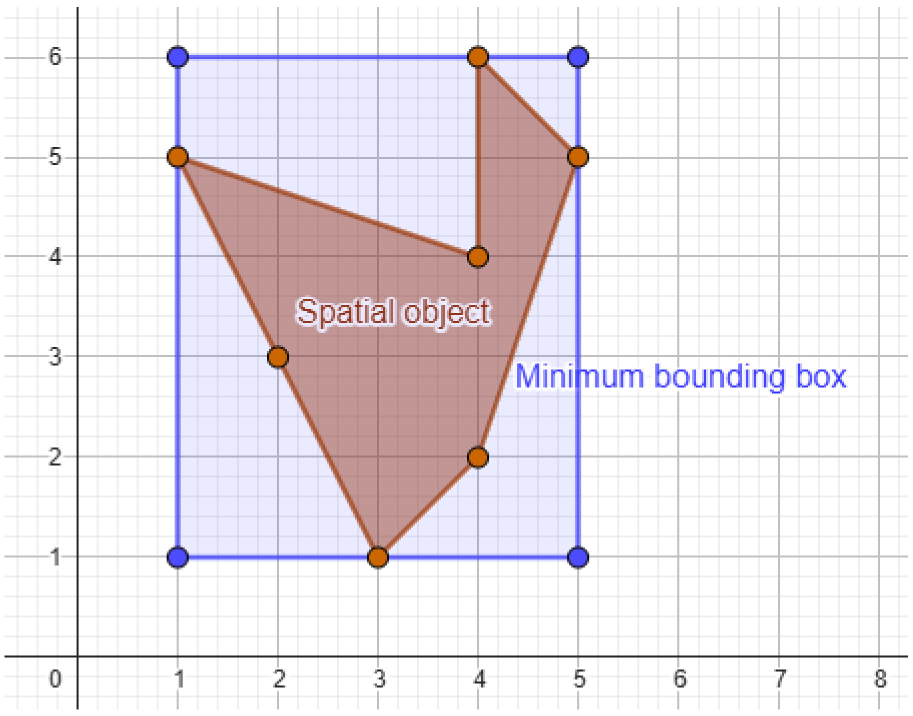


Figure 9. Spatial object and its minimum bounding box

The minimum bounding boxes are stored in an R-tree in a hierarchical manner, with each node of the tree including spatial objects that are guaranteed to be fully contained within the minimum bounding box of the node. R-tree structures are not limited to two or three dimensions, as the same minimum bounding box concept applies regardless of the number of dimensions. A simple visualization of a two-dimensional R-tree is depicted in Figure 10, where each R*n* depicts a minimum bounding box for spatial objects. (Carniel, 2023; Sypytkowski, 2022.)

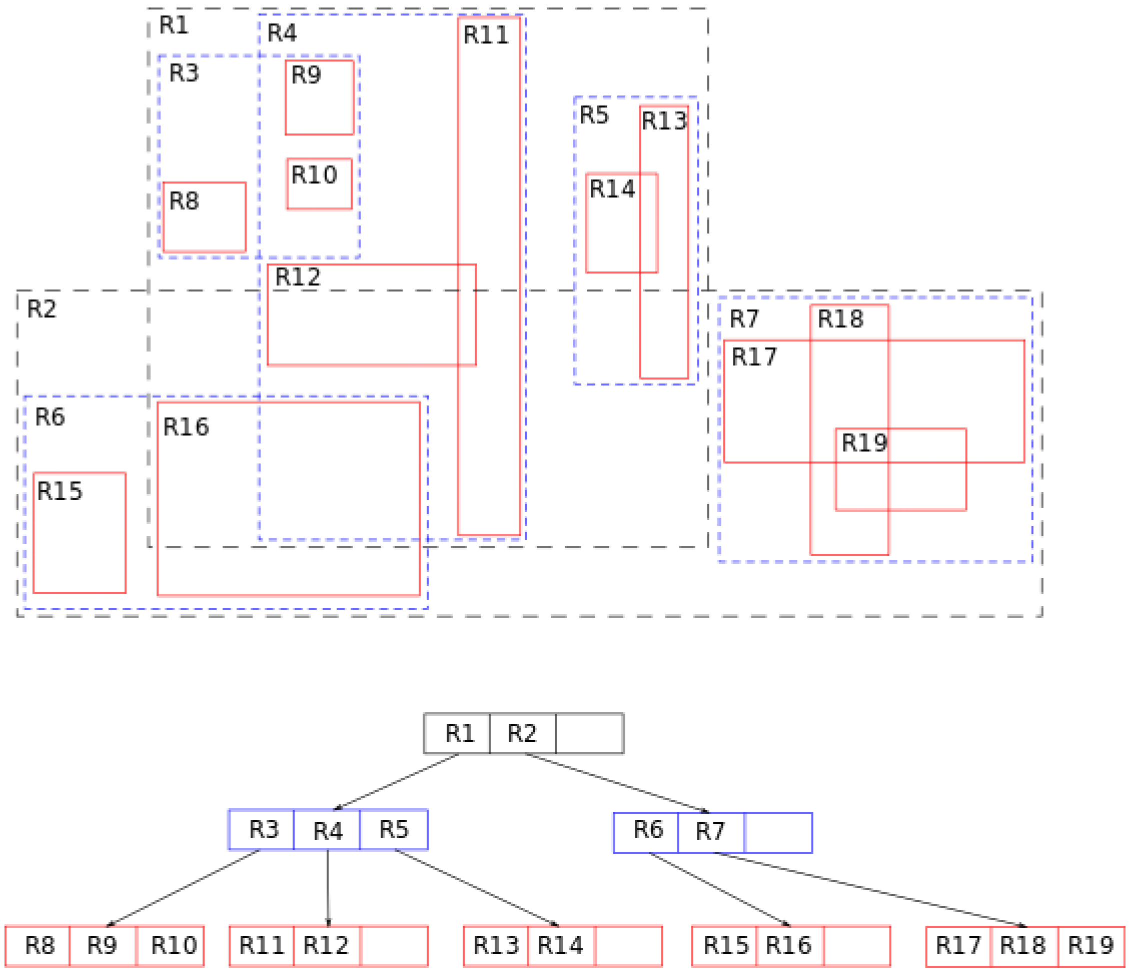


Figure 10. R-tree structure visualized

Carniel (2023) notes that R-tree indexing requires a two-tiered approach for efficient spatial queries. Comparison of the search object and minimum bounding boxes is computationally fast but does not produce accurate results, so a second step is required to refine the results to precise matches. As the search object may interact with the minimum bounding box of a spatial object but not necessarily with the spatial object itself, spatial indexes can efficiently determine which special objects *definitely do not* bear a relationship with the search object, and which spatial objects *may bear a relationship* with the search object. Instead of finding a definitive result as is often the case with regular indexes, a spatial index is used to first discard spatial objects that are with absolute certainty not matches to the search object. Once the first tier of filtering is completed by finding the potential matches, the matches are subjected to a more computationally heavy comparison to determine whether spatial object is an exact match to the spatial query.

4.3 Data preprocessing

4.3.1 Exploratory data analysis

Exploratory data analysis (EDA) is a process used to investigate and summarize datasets with the aim of detecting patterns and anomalies, testing hypotheses and verifying assumptions. The role of EDA is to provide an objective look at the data and help create an understanding of the data at hand, identify any obvious patterns, or even showcase obvious errors or quality issues within the dataset. EDA acts as a tool for data scientists to verify the validity and applicability of their results, as well as guide stakeholders in choosing the variables and relationships to analyze further. EDA is an important starting point in understanding data, especially in cases where numerous variables are at play, or samples hold spatial relationships. (Exploratory Data Analysis, n.d.; What is Exploratory Data Analysis?, n.d.)

The origins of EDA can be tracked to John Tukey's large repertoire of work in EDA, especially his book*, Exploratory Data Analysis*, published in 1977. The book presented tools and structured strategies for EDA with which a data scientist could approach a dataset in a comprehensive manner. As a pioneer of the field, Tukey also coined many of the terms still in use today, such as stem-and-leaf display, and box-and-whisker plot (Hoaglin, 2003.)

Unfortunately, Tukey's original material was not available at the local library or the accessible article databases during the implementation of this research. Instead, an article by Vigni et. al. (2013) uses Tukey's work as a basis for a three-part approach to EDA, covering descriptive statistics, projection techniques and clustering techniques.

Vigni et. al. (2013) outlines descriptive statistics as an explorative method to summarize the statistical properties of a small number of variables. The statistical indicators include measures related to the central position of a frequency distribution (such as mean, median and mode), measures of spread (such as range, quartiles and standard deviation), and measurements on the shape of the distribution (skewness and kurtosis). The statistics only provide a general feel to the data, and visual tools should be utilized to inspect the data in more detail. Histograms, box-whisker and scatter plots should be utilized to locate anomalies, outliers, variation changes and relationships between variables.

Vigni et. al. (2013) also mention projection techniques as useful tools for efficiently working with a dataset that contains a large number of variables. Projection techniques such as Principal Component Analysis (PCA) can be used to compress a large number of variables into a much smaller set of parameters while maintaining the underlying structure and information within the dataset, reducing the dimensionality of the dataset. Latent variables from PCA can then be visualized in scatter plots, providing a better understanding of a complex dataset.

Clustering techniques offer a different line of approach in comparison to projection tools. Cluster analysis attempts to group similar data points by calculating the distances (or magnitudes of differences) between different data points. Vigni et. al. (2013) also note that use of clustering methods requires extensive knowledge about the datasets in question. Different clustering methods are capable of detecting different cluster types, and multiple different methods may have to be applied to detect the type of clustering present in a specific dataset.

4.3.2 Exploratory spatial data analysis

Exploratory spatial data analysis (ESDA) is a subset of EDA used with spatially distributed data to reveal geographical patterns. As was the case with EDA, the statistical and graphical exploration of the dataset is equally important in ESDA. ESDA does not rule out the use of traditional EDA methods and should be instead considered an extension to the methods, to be used with geographical data. (Exploratory Data Analysis, n.d.; Kanevski et. al., 2009.)

Kanevski et. al. (2009) recommends the use of GIS throughout the ESDA process to visualize the data, starting with simply visualizing the spatial dataset. The geographical distribution of data points may reveal patterns in spatial distribution of data, just as a histogram would reveal patterns in value distribution of a single variable. When possible, the dataset should be plotted as points in geographical space, with different plots used when the initial visualization gives an indication that further plots may be beneficial, such as plotting density maps with many overlapping data points.

Further analysis techniques recommended by Kanevski et. al. (2009) include regionalized EDA and moving window statistics. Regionalized EDA involves extracting a specific geographic subset of the data and using traditional EDA methods on the regional dataset. Regional EDA may help in understanding spatial variability between different regions. In a similar manner, in moving window statistics, descriptive statistics are calculated for a spatial subset of data that is moved slightly at each calculation interval, providing a measure of how the geographic location affects the variables.

Kanevski et. al. (2009) also suggests examining the spatial relationships between variables by utilizing variogram analysis. Rose diagrams and other variograms capture the relationship between variables and distances between different data points, visualizing directions of anisotropy – directions of highest correlation – which provide numerical quantification to the spatial patterns within the dataset. *k*-Nearest Neighbor technique is also useful in visualizing the spatially distributed dataset and can also be used as a benchmark against machine learning algorithms.

4.3.3 Spatial clustering and sampling

Kanevski et. al. (2009) places high importance on spatial clustering of data points and measurement locations, as the sampling itself can be a source of bias in the analysis. Clustering and preferential sampling result in a nonhomogeneous collection of data, which complicates mean and covariance estimations and presents an obstacle to data-driven modelling. Figure 11 visualizes the difference between homogeneous sampling and clustered sampling, where the predictions are likely to overfit the area of clustered sampling.

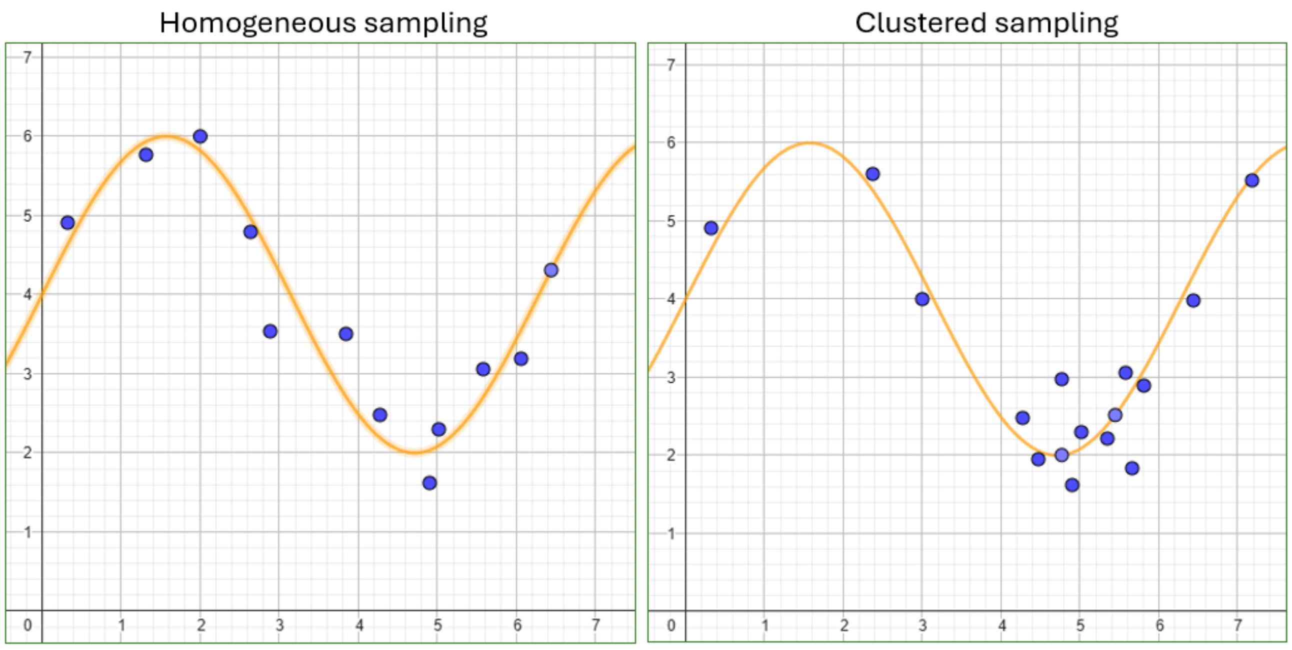


Figure 11. Homogeneous vs clustered sampling

In most spatial datasets, the measurements are clustered. Clustering commonly stems from practical reasons: the measurements may have to be conducted with specialized equipment only present in locations that are not geographically spread out, political and administrative boundaries may result in clear-cut differences as different administrations conduct measurements differently, or geographic features such as mountains or lakes may direct the measurements to more easily accessible locations. Even awareness and political reasons may play a role in preferential sampling: environmental and pollution data, for example, typically showcase preferential sampling, as more data points are collected in an area where pollution or environmental quality has been noticed and data measurement points set up as a result. (Kanevski et. al., 2009.)

Kanevski et. al. (2009) proposes using topological and distance-based measures as indicators to detect the level of clustering, all of which essentially produce visual or numerical representations of density measurements. One of the suggested methods is to utilize Voronoi diagrams and Voronoi Density Estimator to visualize the density of measurement points. Voronoi Density Estimator was originally proposed by Ord (1978) to estimate the number of trees in a forest and has been since adapted to work as a tool in a larger scale of problems and in a higher number of dimensions than Ord's original model.

Another method that Kanevski et. al. (2009) suggests determining the magnitude of clustering is box-counting. In box-counting, a grid of boxes is superimposed on top of the data points and calculating the number of boxes that contained any number of data points. The process is iterated over with grids consisting of ever smaller boxes, and the number of relevant boxes noted. The mathematical relationship between box size and number of boxes containing data points provides an indication of clustering of data points. Caballero et. al. (2022) further refined this process to produce a measure on spatial randomness and implemented the method to analyze the clustering extent of Colombia's COVID 19 infection cases.

Kanevski et. al. (2009) note that a variety of different methods exist for clustering detection, and the detection of clustering is more important than the specific method used in the process. Likewise, the handling of clustered data and dealing with clustering in model training must be accounted for. One of the simplest forms of declustering is to cover a geographical area with cells and pick one random data point from each cell, under-sampling the highest density data points. The random sampling will result in a more homogeneous and spatially representative dataset. However, Alencar (2017) also notes that purely random under-sampling of the majority portion of dataset may lead to loss of information. Clustering the spatially dense locations and performing under-sampling by targeting a cluster at a time is likely to be more efficient at preserving information. Oversampling, or duplicating random datapoints from the minority portion of dataset, may also improve training process, but is prone to overfitting.

4.4 Machine learning

4.4.1 Artificial intelligence

Google (What is Artificial Intelligence (AI)?: Google Cloud, n.d.) defines artificial intelligence (AI) as a field of science that builds machines capable of reasoning, learning and acting as if they had human intelligence, or capable of handling data which is beyond human analytic capabilities.

Google (What is Artificial Intelligence (AI)?: Google Cloud, n.d.) additionally divides AI into four categories: 1) reactive machines, 2) limited memory, 3) theory of mind, and 4) self-aware. Reactive machines are a form of AI that can react based on preprogrammed rules but are unable to transform new data into new knowledge. IBM's Deep Blue was a reactive machine, and in 1997 was able to best chess champion Garry Kasparov. Most contemporary AI implementations can improve their performance with new data and are as such considered limited memory type AI. Theory of mind type AI is capable of emulating human mind and can make decisions and react to social situations like a human. Theory of mind type AI is the target of ongoing research, but no functional implementation exists yet. Self-aware AI is intelligent and has human emotional capabilities in addition to being aware of its own existence, and like theory of mind type AI, does not currently exist.

IBM (What is Artificial Intelligence (AI)?: IBM, n.d.) combines Google's four categories into two: weak and strong AI. Weak AI (also: narrow AI) encompasses current AI implementations, where the AI implementation has been trained to perform a specific task. Strong AI is similar to Google's theory of mind and self-aware type AI: a theoretical self-aware form of AI capable of learning, planning and solving a wide variety of problems.

4.4.2 Machine learning

University of Berkeley (What is Machine Learning (ML)?, 2020) describes machine learning (ML) is a subset of AI involved in enabling a device to automatically learn over time from information provided to the device.

Salian (2018) categorizes learning algorithms into four categories: 1) supervised learning, 2) unsupervised learning, 3) semi-supervised learning, and 4) reinforcement learning. In supervised learning, the dataset used for training contains both the features from which AI will learn, as well as the target values that the AI should provide when presented with the data. Given the target values, the ML algorithm will iterate over multiple steps to improve the accuracy of the model. In unsupervised model, the dataset does not contain predicted labels or values, and the task is up to the AI to extract useful features and find structure in the given dataset. Unsupervised learning may be used to group individual data points, locate anomalies or find similar data points. Semi-supervised learning is a combination of the two, where some of the dataset has been labelled, assisting the model in making better decisions in unsupervised training schemes. In reinforcement learning, the

AI is rewarded when it takes an action towards the intended goal. Over time, the AI incorporates actions that have been rewarded into its strategy, while discarding the ones that have not resulted in a reward.

University of Berkeley (What is Machine Learning (ML)?, 2020) breaks typical supervised machine learning algorithms into three components: a decision process, an error function and an optimization process. The decision process performs calculations on the data and attempts to guess at the pattern that the algorithm attempts to find. The error function provides a measurement on how close the guess resulting from the decision process was. Finally, the optimization process adjusts the decision process to reduce the magnitude of the error, bringing the guesses closer to the actual values. As the three steps are iterated multiple times over, the magnitude of error of the ML algorithm lessens and model predictions improve.

4.4.3 Regression and classification

Supervised machine learning is classified into two main categories: classification tasks and regression tasks. In classification tasks, the goal of a machine learning algorithm is to decide which predetermined category a point of data falls into, based on training data that has been labelled into the same pre-determined categories. A machine learning implementation trained to recognize spam mail is an example application of a classification algorithm. Regression algorithms attempt to predict an output value from the input variables, which are known to have a linear relationship with the output value. An example of a regression-based machine learning implementation would be a machine learning model that predicts house values based on attributes of the house, such as number of rooms and age. (Kanade, 2022).

Multi-target regression aims to utilize a training dataset to simultaneously predict more than one continuous target variables. In practice, multi-target regression is often a single-target regression algorithm that can produce an accurate result to all the target variables separately. Multi-target regression is prone to complexity and overfitting, and special attention needs to be paid to target variable scaling and correlations. Multiple outputs also increase the dimensionality, which runs the risk of overfitting the model. Technical implementations of multi-target regression algorithms include the following:

- **Multi-Target Regressor:** a model developed by Scikit-Learn specifically for multi-class regression.
- **Multi-Target Linear Regression:** extension of simple linear regression but may fail to capture complex relationships.
- **Decision Trees and Random Forests:** native support for multiple outputs with the ability to learn complex relationships.
- **Neural Networks:** deep learning model capable of outputting multiple variables, but is complex, memory-intensive and computationally expensive.

Other models may also be manually adapted and combined to provide separate predictions for each target variable. (Developing Multi-Target Regression Models with Python, 2023.)

4.4.4 Machine learning algorithms

Machine learning algorithms are numerous, with different methods for a wide range of applications. The scope of this research limits the potential algorithms to models capable of multi-target output, namely: 1) Multi-Target Regressor, 2) Multi-Target Linear Regression, 3) Decision Trees and Random Forests, and 4) Neural Networks. The relationship between coordinates and address information is also complex, further limiting out Multi-Target Linear Regression as a potential algorithm, as simple linear regression is unlikely to be able to capture the relationship between the variables. While neural networks would likely be able to adequately learn the dependencies between the variables, the computation power and memory requirements cause concerns with the size of the dataset used for this research. This leaves Multi-Target Regressor and Random Forests as potential models for this research.

An implementation of Multi-Target Regressor is the MultiOutputRegressor by scikit-learn, which acts as a wrapper to fit one regressor algorithm per target output variable. The internal regressor algorithm can be any regular regression algorithm that implements “fit” and “predict” methods required for model training, such as Support Vector Regression (SVR). Image 12 shows the generic architecture of MultiOutputRegressor wrapper. (Multioutput Regression in Machine Learning, 2023; MultiOutputRegressor, n.d.)

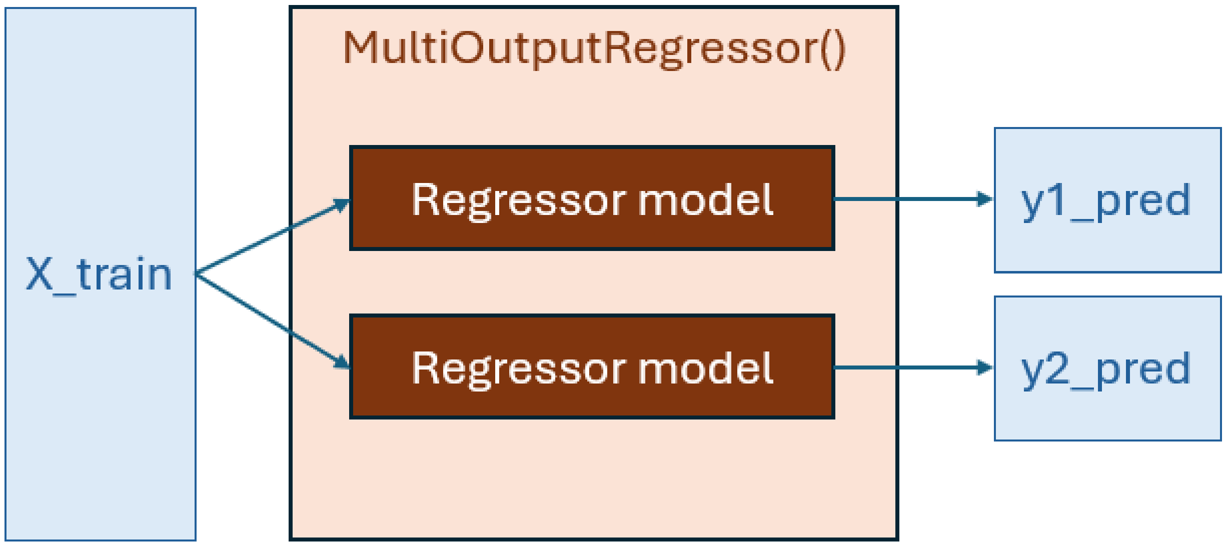


Figure 12. MultiOutputRegressor wrapper logic

A decision tree is a supervised learning method that aims to create a hierarchical set of if-else rules, with which the output variables can be predicted in a reliable manner. Decision trees by their nature produce singular values at their leaf nodes, and do not perform well in extrapolation tasks. Decision trees function well with minimal data preprocessing; normalization and non-numeric variable encoding are not mandatory, and some implementations are also able to handle missing values. Decision trees also natively support multi-output problems. On the other hand, decision trees are prone to overfitting, and optimization methods such as pruning and setting tree depth maximums may be necessary. (Decision trees, n.d.)

Random Forest is a supervised learning algorithm that combines multiple decision trees to form a model to predict continuous values. Random Forests splits the training data into subsets and distributes each subset of data to an individual decision tree. Each decision tree produces its own model with the given subset of data and generates its own predictions to the test data. The predictions from each decision tree within the random forest are averaged, resulting in a single prediction from the Random Forest model. Averaging the predictions of each decision tree helps improve accuracy and reduce variance. Image 13 shows a simplified Random Forest model architecture. (Chaya, 2020; Mwiti, 2023.)

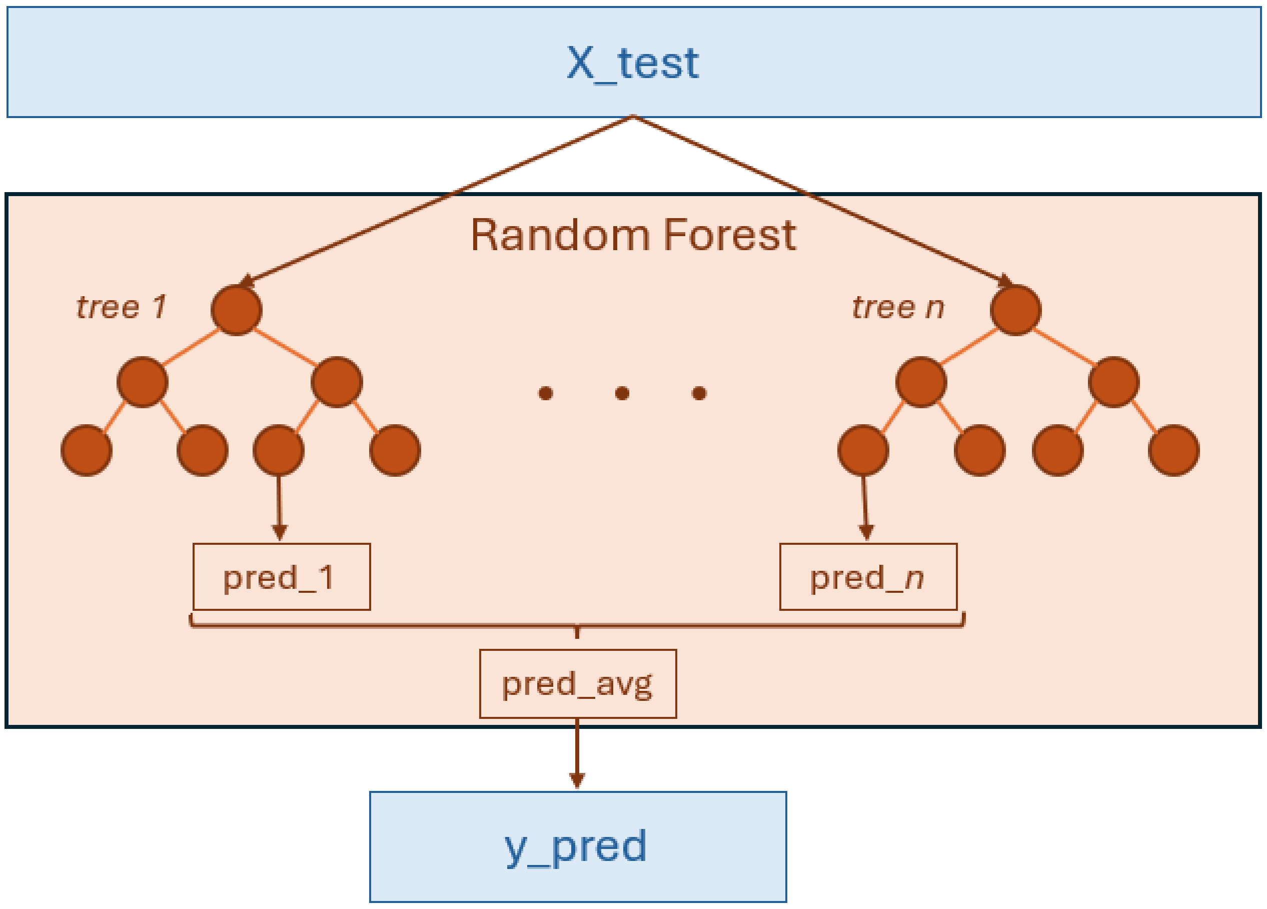


Figure 13. Random Forest model

The fact that Random Forest Regression creates its predictions by averaging also results in the limitation of not being able to extrapolate outside the data presented to the model. As the decision trees within the Random Forest model are split with if-else rules, the individual branches will always produce predictions within the same values that were provided within the dataset. If extrapolation capability is not important, then a Random Forest model may be a good fit while acknowledging the limitations regarding the predictions. If extrapolation abilities are required, a different model or an extended Random Forest model (such as Regression-Enhanced Random Forest) may be better suited to the task. (Mwiti, 2023.)

4.4.5 Random Forest Regression tuning

As outlined in previous chapter, Random Forest models utilize a group of decision trees to fetch potential solutions to a problem, which are then averaged or voted on to provide the final output. In the case of Random Forest Regression, each decision tree reaches a numeric label individually, which are then averaged to produce the final estimation of the entire forest. (Chaya, 2020; Mwiti, 2023.)

The behavior of individual decision trees can be altered in mainstream implementations of Random Forest Regression algorithms, potentially improving the performance of the algorithm. One major parameter in the behavior of the algorithm is whether bootstrapping is utilized in the training of the model. Bootstrapping is a resampling technique where the dataset used for model training is randomly sampled, so that each individual tree receives a different subset of the original dataset for training. The aim of bootstrapping is to reduce the variance of the predictions, resulting in more accurate predictions overall. (Bootstrap Aggregation, n.d.)

Random Forest Regression behavior can be further directed by setting the number of trees in the forests, as well as setting limits on the individual trees by limiting maximum depth, or imposing minimums on leaf and node sizes. Increasing the number of trees in the forest generally improves the accuracy of the model and reduces variance, as the forest averages out the predictions from each tree. Higher number of trees does bring additional computation costs, potentially for very limited performance gains. Maximum tree depth in turn allows for capturing of complex relationships, but may lead to overfitting of data, as individual quirks of the training data are learned in place of general patterns. (The Effects of the Depth and Number of Trees in a Random Forest, 2024.)

4.4.6 Extreme Gradient Boosting with Random Forest Regression

Where Random Forest Regression utilizes individual decision trees to calculate predictions in parallel, boosting algorithms work in a sequential manner to improve predictions after each iteration based on the prediction residuals. Extreme Gradient Boosting (XGBoost) is a specific adaptation of the boosting algorithm. In XGBoost Random Forest Regression, each tree is molded by the results from the previously constructed tree. The algorithm aims to counter overfitting, a common pitfall of otherwise highly accurate Random Forest algorithms, by building the new iteration of the tree only up to the point where it provides a significant improvement over the previous iteration. Ignoring the minute changes allows for better generalization, preventing overfitting. (Lev, 2022.)

In comparison to the Random Forest Regression default setup, XGBoost does not utilize bootstrapping. As the individual decision trees are constructed sequentially, the entire dataset is provided for the initial tree without random sampling. Likewise, the hyperparameters for XGBoost also only apply to the initial tree, as the algorithm takes over the fitting process by relying on loss metrics to adjust the model to the correct direction. (Lev, 2022.)

4.4.7 Machine learning best practices

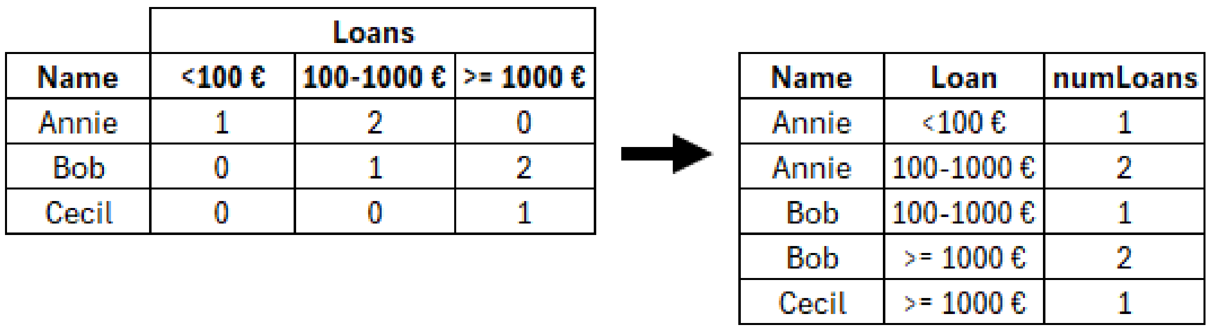
Machine learning outcomes are only as good as the data used in training the model. A dataset needs to contain valid, clean data that can be used to make predictions. The dataset should contain all necessary features or attributes needed to predict the outcome of a machine learning problem. Likewise, the number of good quality data points in the dataset needs to be high enough for the machine learning algorithm to be able to learn the relationships between the features and the target variables. Zhu et. al. (2013) considered an adequate sample-to-feature (SFR) to be at bare minimum 10 for classification, preferably over 100 to ensure good learning potential for more complex tasks. In cases where sample size is small or variable size is large, feature selection should be performed to reduce dataset dimensionality and increase SFR for better learning outcomes.

In addition to ensuring a sufficient quantity and quality of data, the data should be complete and representative of the entire domain at hand. Completeness of the dataset and potential loss of information needs to be considered when cleaning the dataset, as removal of observations may potentially introduce bias to the dataset. While replacing missing data will not lead to data loss, the process may affect covariance or correlation variables. Imputation of missing data is a better choice but can be time consuming and introduce more uncertainty. Data completeness and statistics should be reported even when no issues are identified. (Wujek et. al., 2016; Zhu et. al., 2013.)

The data used in creating a machine learning algorithm should be split into different sets for training and evaluation purposes. Zhu et. al. (2013) found that different splits were used by researchers, with a simple two-way train-test split being the most common form. Other split structures were train-validation-test split and train-test split with train data further split into cross-validation segments. The training portion should have enough samples to represent the entire dataset without bias and consist of between 10-40% of the entire dataset.

Special care needs to be taken when imputing and scaling the different segments of datasets to prevent data leakage. Imputing the test dataset with a median from the training dataset would bring in information from outside the test dataset, creating a data leakage. Values for imputations and other basic transformations should be evaluated from the data within the specific data segments, or created by a formula or a rule that can be applied across partitions. Inherent dependencies between variables also need to be considered as potential sources of data leakage, as for example a five-day time lag between variable and observation would render any data under five days old unreliable. (Wujek et. al., 2016; Zhu et. al., 2013.)

Dataset features should be well-structured and prepared before the training process. Wujek et. al. (2016) advises to use melting, string-splitting and casting to prepare the dataset structure. Melting is used to merge different columns used to denote the values of the same feature into a single column, demonstrated in Figure 14.



	Loans		
Name	<100 €	100-1000 €	>= 1000 €
Annie	1	2	0
Bob	0	1	2
Cecil	0	0	1

Name	Loan	numLoans
Annie	<100 €	1
Annie	100-1000 €	2
Bob	100-1000 €	1
Bob	>= 1000 €	2
Cecil	>= 1000 €	1

Figure 14. Melting features into smaller number of columns

String-splitting takes multiple values contained within a single column and splits the values into their own columns, decoupling variables from one another. For example, a “M/18” denoting the gender and age of a person would be split into columns “gender” and “age”, containing values “M” and “18”, respectively. Casting is the opposite of melting, where the values contained within a single column are turned into multiple columns. One-hot-encoding is a commonly used method of transforming labels into numeric data. (Wujek et. al., 2016.)

Features with values significantly different in range and magnitude can adversely effect machine learning algorithm performance, and in general should be accounted for by methods of standardization and normalization. Interval feature values should be standardized to the same scale, usually values with mean 0 and variance of 1. However, a sparse dataset with a high percentage of 0-values loses the sparseness with this manner of standardization and should instead be simplified along a scale based on the maximum absolute value of the feature. Extreme outliers will also cause problems with range standardization and should be dealt with before further processing. (Wujek et. al., 2016; Zhu et. al., 2013.)

Zhu et. al. (2013) recommend comparing at least two different machine learning algorithms in cases of supervised learning methods, as to avoid falling into the trap of only utilizing the most popular algorithm, when a different algorithm may produce a more suitable result. Likewise, use of more than one evaluation metric for model performance evaluation would be preferable. Random seed used in training should be standardized, however especially in cases with a small sample size, different random seeds should also be evaluated to understand the effect a different random state has on the resulting model. Model hyperparameters, the configuration parameters passed in code to the machine learning algorithm, should also be optimized to produce a better fitting model. Optimization can be carried out in a simple trial-or-error manner where different parameters are tested and the best combination kept for the final model, or performed in a more systematic manner with approaches such as grid search.

4.4.8 Natural language processing

Natural language processing (NLP) is a ML technology which allows computers to interpret, comprehend and manipulate human language. NLP is used to analyze text and speech to understand context, detect underlying emotions and sentiments (such as happiness, anger or sarcasm), and generate realistic written or spoken speech. NLP utilizes tools such as tokenization, stemming and lemmatization to change words into format that can be utilized in machine learning algorithms. Tokenization splits a sentence into individual components while stemming and lemmatization turn words into their base form (such as “resting” to “rest”). (Eppright, 2021; What is NLP?, n.d.)

Word vectorization, or word embedding, is an NLP methodology that turns text into numerical vectors, which in turn can be utilized in machine learning algorithms. Word vectorization has the benefit of grouping similar words closer to each other in vector space, so that characteristics of individual words can be represented in numeric terms. A higher-dimension vector form of a word may capture characteristics such as gender, verb tenses, or even more abstract concepts such as geographic associations between countries and cities. In a vectorized form, the patterns in word characteristics can be used in machine learning processes and other algorithms. (Bogacz, 2023; Raghav, 2019.)

4.5 Data understanding

4.5.1 Finnish building registry

Digital and Population Data Services Agency of Finland maintains a registry containing the addresses, postal codes, coordinates and other enriching information of all Finnish buildings. The dataset includes records for buildings that are completed or currently under construction. Municipal building supervision authorities of individual municipalities are responsible for supplying and updating building information to the national building registry. The 14.05.2024 dated dataset contains 3 837 772 buildings, which fulfill the conditions of having at minimum a set of coordinates and at least one postal code. (Addresses of Finnish Buildings, n.d.; Digital and Population Data Services Agency, n.d.-a; Digital and Population Data Services Agency, n.d.-b.)

A Finnish street address typically consists of up to six parts:

1. Street name,
2. Street number,
3. Door letter,
4. Apartment number,

5. Postal code, and
6. Postal location.

Detached houses that occupy a plot of land on their own typically only contain street name, street number, postal code and postal location in their address. The door information specifies the entrance of a building, where a building may have multiple separate entrances (such as apartment buildings or semi-detached houses). Door information is typically denoted with letters of the alphabet, starting from the letter A. Terraced houses are an exception, as instead of door numbers, the word “as” (short for “asunto”, “apartment” in English) is used instead. Apartment number further specifies the individual apartment within the building complex. Postal code is the regional identification number utilized by the Finnish postal services to group addresses into service areas, with the postal location providing the name of the geographical location for the address. Figure 15 shows an example address in a typical Finnish address structure, showcasing all parts of the address.

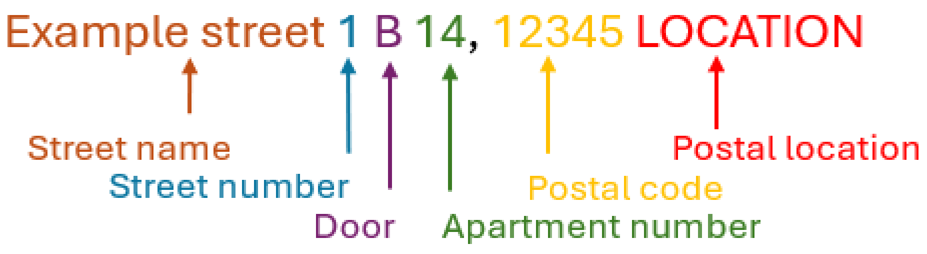


Figure 15. Structure of a Finnish address

While vast majority of the addresses conform to the rules above, as always, ambiguities and data quality issues exist. During their study on address changes related to the 2009 municipality merger of Hämeenlinna, Hauho, Kalvola, Lammi, Renko and Tuulos, Lehtiranta (2015) noted that the building registry addresses contained numerous deficiencies. The street name field often contained unnecessary information such as street numbers, addresses contained spelling mistakes and same addresses were written in a slightly different format (e.g. Tarkk’ampujankatu as Tarkkampujankatu).

Aside from data quality issues, some addresses have their own peculiarities. The street number indicates which physical locations the apartment covers and is not always in a pure single integer format. Uusikaupunki municipality, for example, has an apartment building at Harmaalokkikuja street at numbers 13 and 15. The resulting street address with street number for the building is Harmaalokkikuja 13-15, requiring some further logic if the street number is to be treated as a numeric value. In cases of semi-detached houses, the street number may also contain letters to denote the split between apartments in the building, such as Pensastie 3a and Pensastie 3b.

Sometimes naming conventions of islands also deviate from the generic form. Lehtiranta (2016) noted in their study that some municipality mergers resulted in an island receiving an additional name denoting the proximity to a bigger island or a village, such as Pahaluoto islands renamed to Pahaluoto (Merimaskun) and Pahaluoto (Rymättylän) – see Image 16 below.

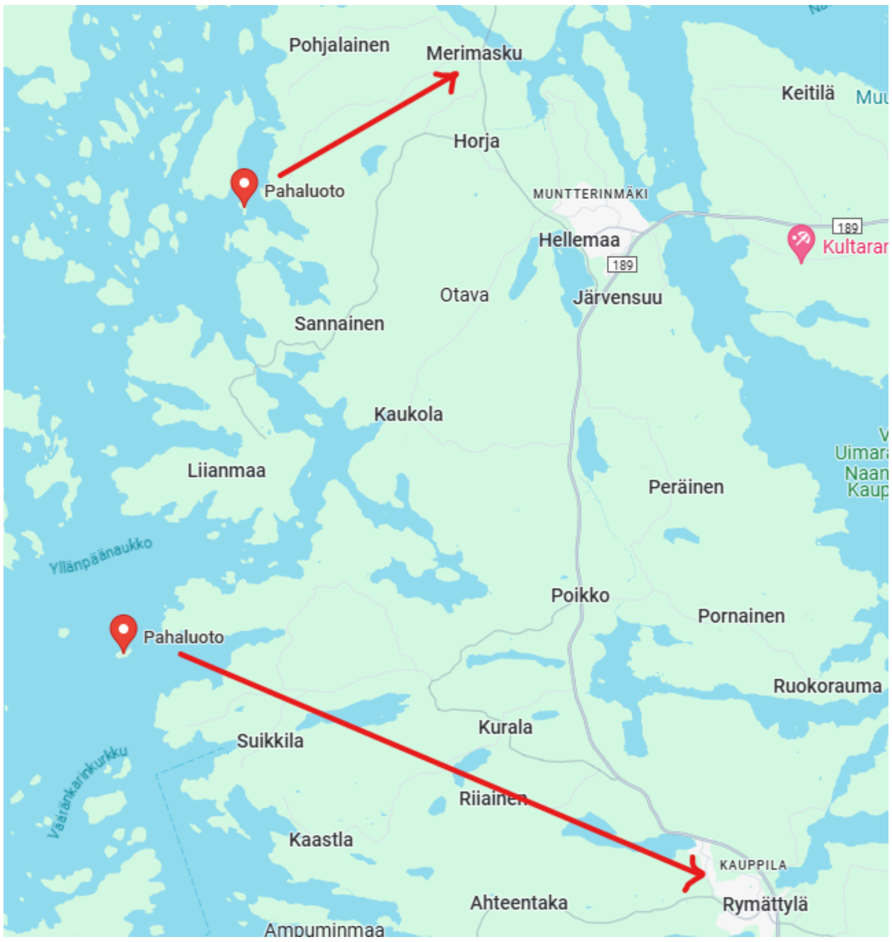


Figure 16. Pahaluoto (Merimaskun) and Pahaluoto (Rymättylän)

Elisa utilizes the Digital and Population Data Services Agency's building registry to maintain internal copies of building information for use with various systems within the company. The original building registry data is transformed and enriched by the Data Analytics department of Elisa, before distribution to various internal stakeholder groups. The building registry dataset used in this research is a 24.06.2024 dated version of the building registry dataset enriched, by the Data Analytics team at Elisa, stored in an Oracle spatial database. The complete dataset contains 2 374 609 rows of data on existing buildings and buildings under construction, with 50 different attributes related to each row. The exact causes for the discrepancy between the row count of the public building registry dataset (3 837 722 rows) and Elisa's internal dataset (2 374 609 rows) are not clearly documented by the Data and Analytics team but are likely to originate from missing data at street name and municipality name level. Most of the attributes of Elisa's internal building registry copy remain out of scope of this research, and only the attributes that were utilized in preprocessing and model training are documented in Table 3 below.

Table 3. Building registry dataset description

Attribute name	**Description**	**Example data**
KUNTANIMI	Name of municipality in which the address is located.	Espoo
KATUNIMI	Street name in Finnish, or street name in Swedish if Finnish name is not available.	Piispantilantie
KATUNRO	Street number. *Note*: this is not always purely numeric.	7
GATUNAMN	Street name in Swedish.	Biskopshemmansvägen
POSTINRO	Postal code of the address. Always five characters in length, padded with leading zeros if necessary.	02240

POSTITOIMIPAIKKA	Postal location in Finnish, or in Swedish if no Finnish equivalent is available. *Note:* this may differ from municipality name.	ESPOO
BUILDING_ID	Unique identifier for an individual building.	200167163
PKOORDINAATTI_TM35	Latitude coordinate in ETRS-TM35FIN (EPSG:3067) projection.	6671899
IKOORDINAATTI_TM35	Longitude coordinate in ETRS-TM35FIN (EPSG:3067) projection.	373922
POSTITOIMIPAIKKA_SWE	Postal location in Swedish. *Note*: this may differ from municipality name.	ESBO
POINTGEOMETRY	Oracle SDO_GEOMETRY geometry object formed from address coordinates.	<SDO_GEOMETRY object>

Some of the address format inconsistencies have already been handled by the Data and Analytics team, as the basic issues with data quality as described by Rahkola (2015) are not present in the Elisa-specific dataset. Ambiguities and deviations from the basic address structure like described by Lehtiranta (2016) are present in the address registry and should be carefully considered whether the identifiably ambiguous addresses can be dropped from the dataset or otherwise handled.

4.5.2 Road geometry data

The Finnish Transport Infrastructure Agency maintains and distributes Digiroad dataset, containing the properties and geometries of Finland’s road and street network. Digiroad data contents are maintained by municipalities, private road municipalities and ELY Centers. The dataset contains comprehensive information on all roads from walking and cycling lanes to highways and ferries, including street names and numbers, street types, restrictions, speed limits, and so on. The dataset is freely available through Finnish Transport Infrastructure Agency. (Finnish Transport Infrastructure Agency, 2014.)

As is the case with the building registry, Elisa also utilizes a local refined copy of the Digiroad dataset. However, as Elisa's version of the dataset has not been updated recently, this research utilizes the Digiroad dataset directly from the Finnish Transportation Infrastructure Agency, dated to be generated on 29.05.2024. As is evident from the Digiroad description in previous chapter, the dataset contains a wide range of variables, most of which do not benefit this research. The relevant portion of the dataset is described below in Table 4.

Table 4. Digiroad dataset description

Attribute name	**Description**	**Example data**
LINKKITYYP	Road link type. *Note*: values of <= 5 included, covering all road segments that can be navigated with a car.	5
LINK_ID	Unique identifier for a road link.	<guid>
KUNTAKOODI	Municipality code.	700
TIENIMI_SU	Road name in Finnish.	Susilammentie
TIENIMI_RU	Road name in Swedish.	Båthusvägen
ENS_TAL_O	First house number on the right side of the road.	1
ENS_TAL_V	First house number on the left side of the road.	2
VIIM_TAL_O	Last house number on the right side of the road.	89
VIIM_TAL_V	Last house number on the left side of the road.	90
GEOMETRY	Road link geometry object.	<geometry object>

The Digiroad dataset contains road links in sections, and a single road may consist of multiple segments not necessarily in geographical order. The first and last house numbers relate to the specific road link, rather than the entire road consisting of multiple road segments.

While the Digiroad dataset in general seems to be of better data quality than the building registry dataset, different policies across municipalities and other street related authorities mean that very few global rules can be derived on street numbering schemes in Finland. In general, the following rules apply:

1. Odd street numbers are on one side of the street, even street numbers on the other side.
2. Whether the left side of the street is odd or even numbered is consistent within a single municipality.
3. House numbers start from one and increase as the road continues.
4. Only one street with the same name can exist in one municipality.

However, municipality mergers have also brought about their own changes and ambiguities. Lehtiranta (2016) studied address related changes initiated by municipality mergers by gathering data from ten Finnish municipalities. The preparation projects took anywhere from four months to two years, and on average resulted in approximately 115 street name changes. Naming convention challenges often arose along the roads connecting different municipalities, especially in cases where the street name was the same in both municipalities, but both had started the street numbering on their own end of the road. The most common approach was to renumber the street so that the least number of buildings would be affected. Different municipalities also had different rules on which side of the road would have even numbers and which side would have odd numbers, and these had to be reconciliated by renumbering the streets of the other municipality. Some municipalities also operated on entirely different address number division systems, resulting in inconsistencies in address numbering schemes. Statistics Finland (n.d.-b) reports that no municipality mergers have taken place in 2023 or 2024, and therefore larger address renaming projects are unlikely to affect the quality of the dataset at the time of this study.

4.5.3 Municipality dataset

Statistics Finland (n.d.-a) maintains a list of municipality names, numbers and border geometries. The datasets are freely available through Statistics Finland website. The municipality dataset is a compact package of information of the 309 municipalities in Finland, including the municipality number, municipality name and its corresponding border geometry. The data is available through an API or as a direct download. Municipality number is a three-digit identifier with leading zeros where necessary (e.g. 010) and municipality name is the corresponding name of the municipality in Finnish (e.g. Alavus). A corresponding dataset is also maintained in Swedish, with municipality names in Swedish where one is available (e.g. Tavastehus instead of Hämeenlinna), with municipality identifiers shared across the languages. The municipality dataset from Statistics Finland was used in this research to enrich Digiroad road link data with municipality information.

4.5.4 Postal code areas

Statistics Finland (2021) maintains a dataset on Finnish postal areas, including postal codes, postal location names and geographic boundaries. The dataset is open access and can be retrieved through Statistics Finland WFS endpoint. The dataset is served in two different versions: one with land-area borders only, and another version which includes sea areas up to the Finnish borders. The postal area dataset is used to enrich Digiroad dataset with postal code and postal location by spatially matching road endpoint geometries to postal area geometries.

Postal codes are primarily used in Finland to route mail, but have also been adopted for other purposes, such as statistics and marketing. Finland has approximately 3100 different postal numbers, where each postal number denotes one geographically consistent postal area. The first two digits of the postal code denote the general area of the postal code, while the last three digits specify the postal location within the generic postal area. Image 17 shows the generic postal areas in Finland. (Posti, n.d.)

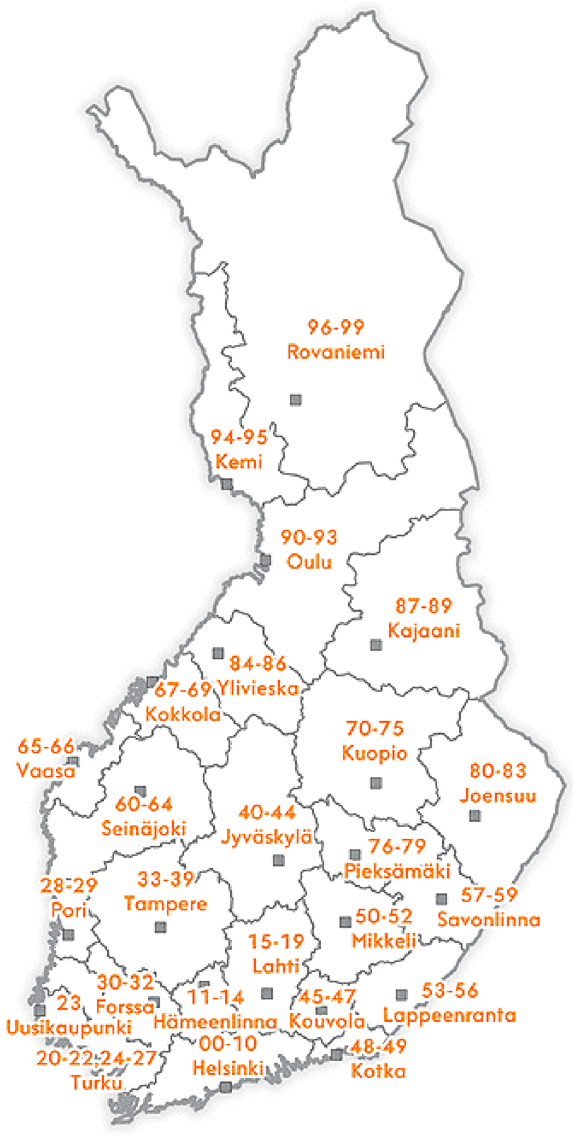


Figure 17. Postal regions in Finland. From *Postinumerot ja postinumeroalueet Suomessa*, by Posti, n.d., https://www.posti.fi/fi/postinumerohaku/postinumeroalueet

Municipality names and postal areas also do not always follow the same borders; one address may reside in a postal area with the neighboring municipality's name, while residing within the administrative region of a different municipality. Also, one municipality may have multiple different postal regions and postal codes, as seen in Figure 18 . Large corporations which receive large quantities of paper mail also present a special case, as they may have their own postal code which is not tied to any specific geographic region. (Posti, n.d.)

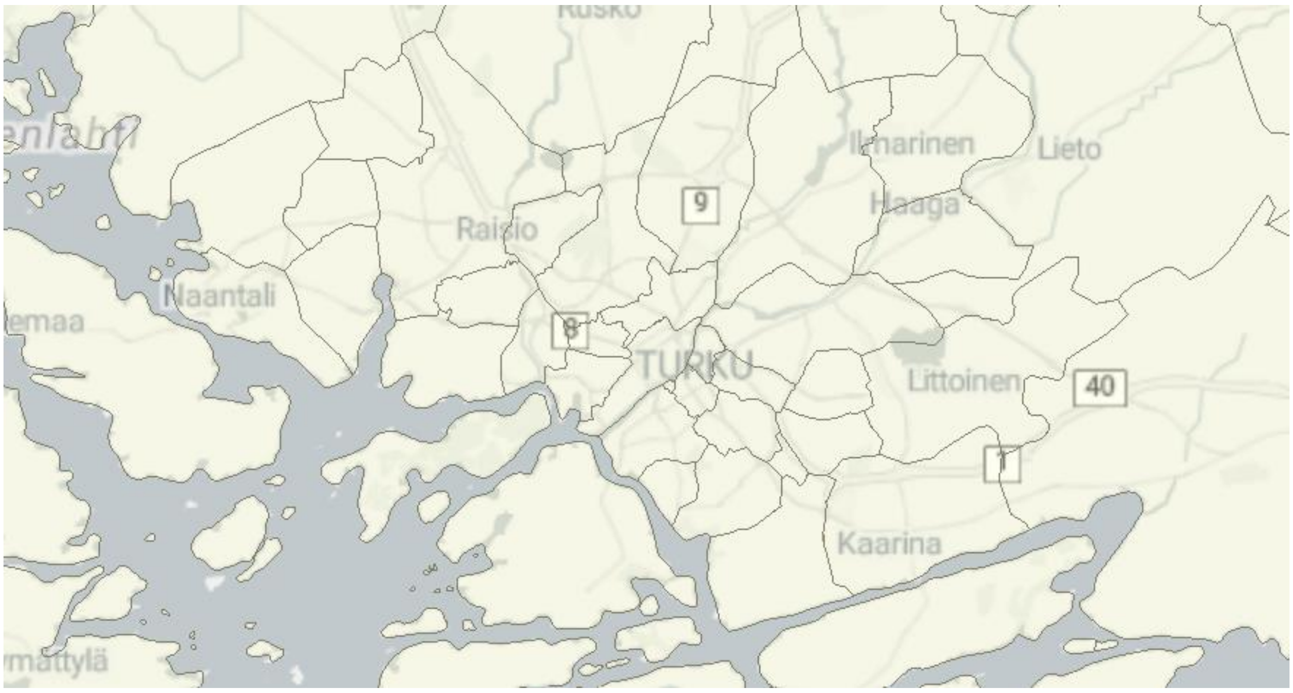


Figure 18. Postal code borders around city of Turku

Using municipalities and postal locations interchangeably in an address complicates many if-else based logical rules used in address handling. The postal code is not always obvious when looking at a map, and the postal code of the nearest major population center may often be used, which may not always be correct. This leads to identification challenges in typical algorithms, where two addresses in seemingly different geographical areas are in fact the same address.

4.5.5 Finnish language model

TurkuNLP Group, a group of researchers associated with University of Turku in Finland, maintain a set of NLP tools specifically developed for Finnish language. Their Finnish Internet Parsebank project analyzed vast quantities of Finnish text online, producing a mass-scale corpus including approximately 3,7 billion tokens. One of the resulting outcomes was a word2vec model for embedding of words in a binary form, which allows for the analysis of semantic similarities between words. (TurkuNLP, n.d.)

The word embeddings have been processed assuming that the language is Finnish, and as a result the model is capable of handling Finnish words well. TurkuNLP (Word embedding demo, n.d.) provides an online demo of the capabilities of the model, capable of producing nearest words, similarities of two words and word analogies. The models behind the online demo also contain words in Swedish and English, but do not produce accurate results for Swedish. The nearest words for Swedish include prepositions and conjunctions, indicating that the language model is not able to accurately find the true nearest equivalents for Swedish language words. Table 5 below displays the results for five nearest words, when searching for the equivalent of the word "road" in Finnish, Swedish and English.

Table 5. TurkuNLP produced nearest words for different languages of "road"

Language	**Word**	**Five nearest words**
Finnish	Tie	Polku Mutkainen Reitti Tieksi Hiekkatie
Swedish	Vägen	Från Och Som På Mig
English	Road	Highway Street Avenue Town Park

4.6 Legislative and ethical considerations

4.6.1 General Data Protection Regulation

The General Data Protection Regulation (GDPR) is a privacy and security law passed by the European Union in 2018 to regulate the collection and storage of personal data. In the context of GDPR, personal data is defined as “any information that relates to an individual who can be directly or indirectly identified” (What is GDPR, n.d., Scope, penalties and key definitions section). Name information is the most obvious personal data, but other secondary properties such as gender, ethnicity or location information can also constitute personal data. The key question is whether the information can be used alone or in conjunction with other datasets to uniquely identify individual persons. (What is GDPR, n.d.)

Applying GDPR, address information is considered personal data when the address can be used to identify an individual person. For example, a list of addresses of persons participating in a soccer camp at Turku in summer 2024 is considered personal data, as the addresses in question are directly related to individual persons. The addresses used in this research are *not* personal information, as the list of addresses bear no relation with any single individual or group of individuals, nor can the list of addresses used in this research be combined with another dataset to identify any group of individuals. As GDPR does not apply to the data in scope, further specifications of the regulation do not need to be considered.

Requirements from GDPR do have to be considered if any model created in this research is to be used for production purposes, as the address data used for production-use predictions may be bound by GDPR. If the model is used to, for example, generate approximate locations for upcoming detached housing construction projects, the addresses of the construction projects would constitute personal information and would need to be treated as such. The model itself would remain free of personal information, though.

4.6.2 Artificial Intelligence Act

The EU AI Act is the world’s first comprehensive regulation on the utilization of AI, aiming to ensure safe, transparent, traceable, non-discriminatory and environmentally friendly development and use of AI systems. The AI Act imposes limits and requirements on AI implementations based on their risk levels, going as far as to outlaw implementations which are deemed to pose an unacceptable risk to people. Unacceptable risk is considered to include systems which are seen as a threat to people, such as cognitive behavioral manipulation, social scoring, biometric identification and categorization, and real-time and remote biometric identification systems. Limited exceptions for biometric identification may be allowed for law enforcement in serious cases. High risk AI implementations that can negatively affect safety or fundamental rights must be assessed before market release and through their lifecycle afterwards, and designated national authorities must be established where people can file complaints regarding the AI systems. High-risk AI implementations are divided into two categories: 1) those falling under EU product safety regulations such as toys, aviation, cars, and 2) those falling into specific pre-determined categories such as critical infrastructure operation, law enforcement or employment, in which case the AI system needs to be registered in an EU database. The EU AI Act also introduces transparency requirements for Generative AI, requiring disclosure of AI generated content, prevention of creation of illegal content, and copyright compliance. (European Parliament, 2023.)

This research does not meet the criteria to be classified as an "unacceptable risk" or "high risk" level of AI implementation, as the resulting model will not have any capability of posing a threat to people, nor is it involved in any processes which relate to safety or fundamental rights of individuals. The implementation is also not a form of Generative AI, nor has any copyrighted material been used in the training of the model. As such, the EU AI Act does not set legal requirements that have to be accounted for.

5 Implementation

5.1 Implementation plan

The central purpose of this research is to create a model that takes a street address as an input and generates output latitude and longitude coordinates for that specific address. To achieve this end goal, a multi-target machine learning algorithm needs to be used, with Random Forests, XGBoost with Random Forests, and Multi-Target Regressor with Support Vector Regression as the potential model candidates.

Random Forests work in an averaging manner, necessitating that all the data to be predicted needs to be located within the extreme boundaries of the training data. As building registry data does not always cover the extreme ends, Kanevski's (2009) recommendation of utilizing modelled data is put into use, and Digiroad road links are utilized to generate the beginning and end points of road geometries, between where all the end results for that specific road are usually expected to land. The generated addresses also serve a dual purpose of ensuring that all potential street names are covered in the training data and providing more spatially evenly distributed training data.

As many municipalities have areas with theming street names, measuring the similarity of street names may increase predictive accuracy. To extract the theming portions of the addresses, the street names are split into a prefix and a suffix, when possible, where the prefix is often the theming part of the name, and the suffix a more general description of a road in some form. For example, "Hirvitie" (eng: Elk road) would be split into prefix "hirvi" (eng: elk) and suffix "tie" (eng: road). The prefix and suffix words are then turned into their base words (lemmatized), which are converted into 200-dimensional vectors. Pretrained models by TurkuNLP group are used for street name lemmatization and vectorization. The bilingual nature of Finnish landscape and rarity of Finnish language implementations in NLP tools poses challenges in vectorization, and best effort is made at preserving meanings of Finnish street names with no separate implementation for Swedish street names.

The enriched vectorized data is used to train multiple machine learning models using RandomForestRegression, XGBoost Random Forest Regression, and Multi-Target Regressor algorithms, fine-tuning and optimizing the models to reach the best outcome possible. The ability of the model to create accurate predictions is then evaluated to assess whether the model is usable in business automation cases.

5.2 Dataset compilation and refinement

5.2.1 Existing addresses in Finland

The building registry addresses were read in from Elisa's spatial database and visualized on a map to visually verify the validity of the data. As expected, the dataset contained a small number of individual locations which fell outside the Finnish land borders, as seen in Figure 19 below.

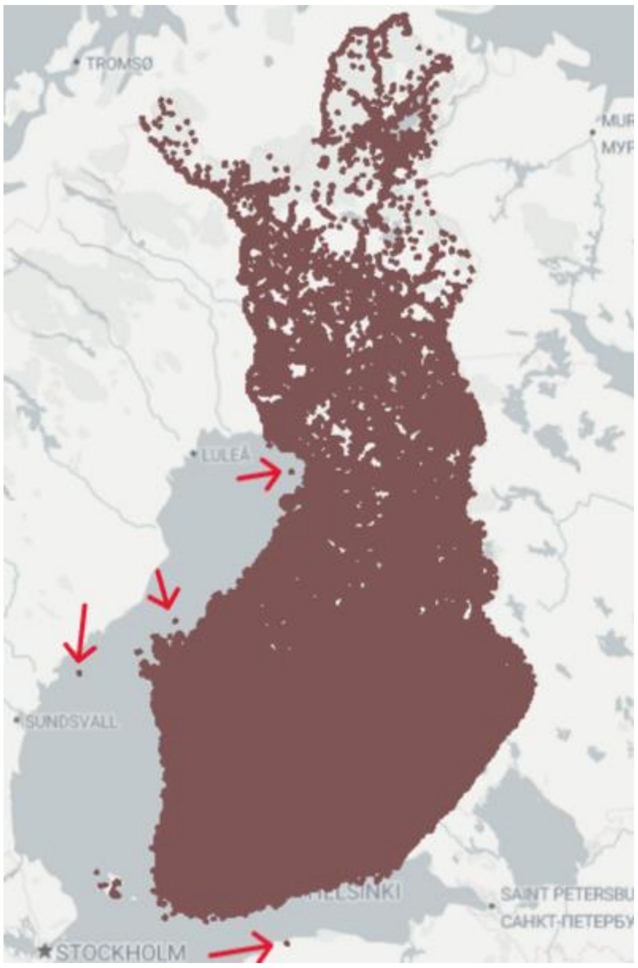


Figure 19. Addresses outside Finnish land borders

For example, the address "Luodontie 408, 21500 PIIKKIÖ" was erroneously documented, and the address appeared to be in the middle of the sea just north of Åland archipelago near Geta – see Figure 20.

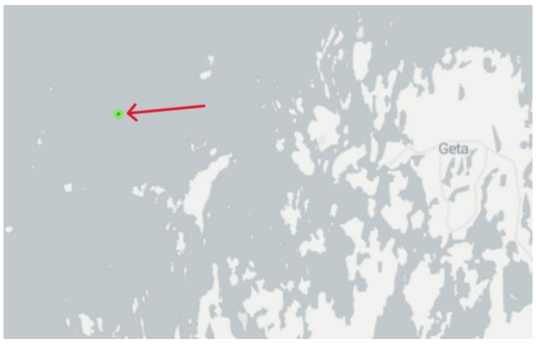


Figure 20. Geolocation for Luodontie 408, 21500 PIIKKIÖ

Municipality border geometries contain the geometric shapes for most of Finland’s land mass. The municipality border dataset consists of the geometries of individual municipalities, and the geometries along the southwestern archipelago attempt to follow the coast and the major islands, but many of the smaller islands are left out of the dataset. The municipality border dataset was read in, and the individual geometries were dissolved into a single geometry. Figure 21 shows the visualization of municipality dataset contents prior to dissolving of geometries.

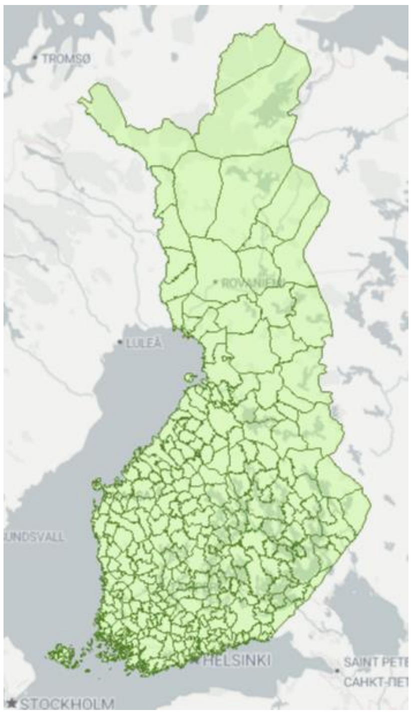


Figure 21. Finnish municipality geometries

Using the municipality geometries as a clipper against the building registry data resulted in a total of 62 982 addresses remaining outside what was considered the actual land areas of Finland. Almost all of the addresses reside along the coast of Finland, indicating that the clipper very likely removed more addresses than were intended. Figure 22 below shows the dissolved municipality geometry used as a clipper in blue, with green points indicating addresses which were left outside the municipality border areas.

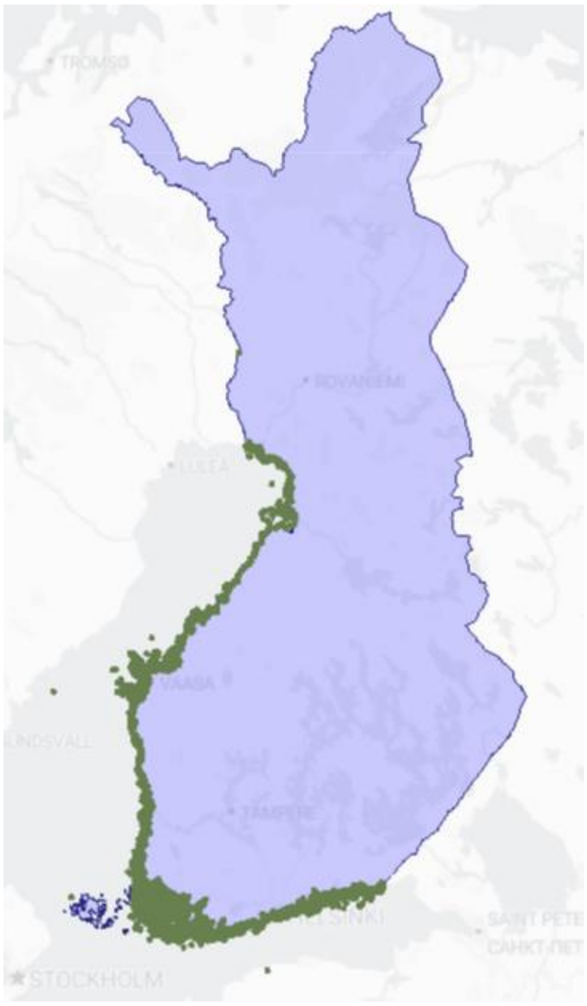


Figure 22. Municipality areas and outlier points

A more detailed visual examination of the data around Nagu (Finnish: Nauvo) island shows the issue at hand clearly. The Finnish archipelago consists of a vast number of islands both large and small, and not nearly all of them are documented in the municipality borders dataset. The exact rules of which islands are documented, and which are left out are not specified alongside the dataset. A quick visual check would suggest that large islands containing permanent residences and roads are likely to be documented, whereas small islands containing summer residences are more likely to be left out of the border geometries. Figure 23 showcases the municipality boundaries around Nagu in blue, with green dots representing addresses clipped out of the dataset.

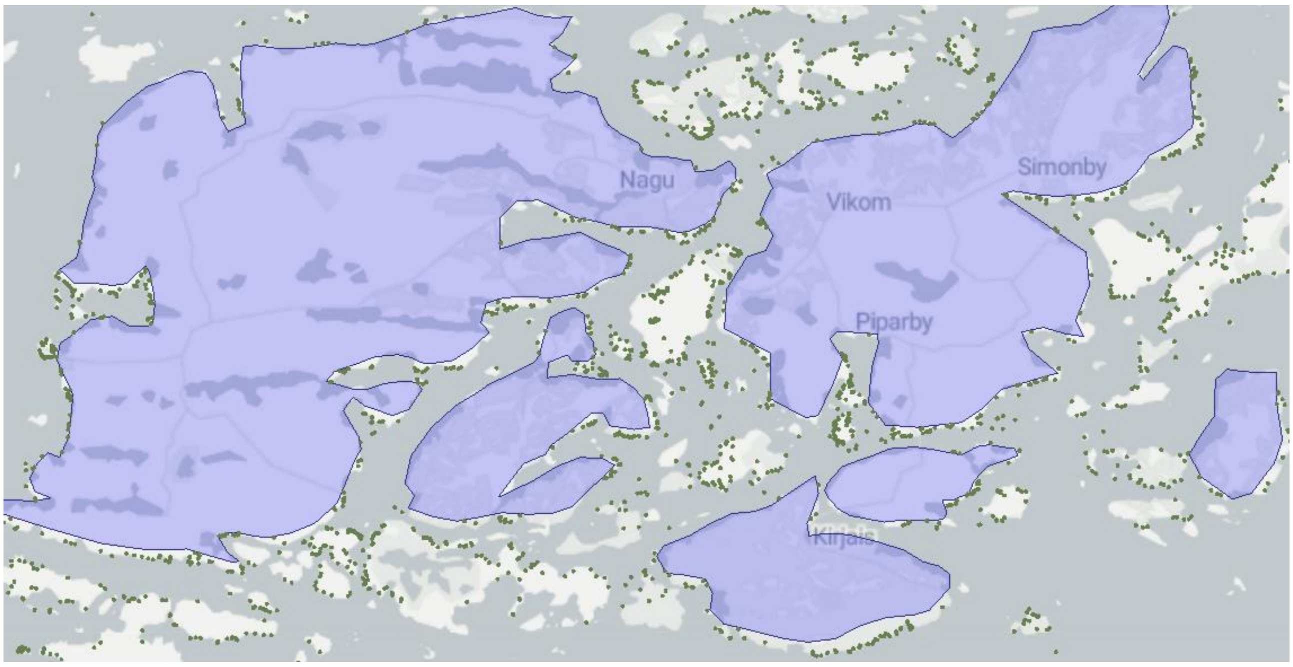


Figure 23. Municipality borders and outlier addresses near Nagu island

The visualization of Nagu island also shows that the municipality borders are not exact, and a significant number of addresses do reside within a documented island, but just outside the borders of the municipality areas. Adding a buffer to the municipality border extends the borders outwards, capturing more of the addresses just outside the current municipality borders, while leaving the true outlier data points and locations within smaller islands intact. Leaving out the data points at smaller islands is acceptable for the purposes of this research, as most of the smaller islands do not contain road, water or telecommunications infrastructure, and only some contain electricity infrastructure. Due to the lack of road infrastructure, the smaller islands do not contain complete address information that could be used for training the machine learning algorithms, nor are the smaller islands likely targets for fiber construction or other potential use cases of this study.

Municipality geometries were buffered at 50-meter intervals and used to re-examine the building dataset. The number of outlier points was recorded and a visual examination of the data around Nagu island was carried out at each interval. Table 6 shows the number of data points outside the municipality borders at different distances of buffers.

Table 6. Number of addresses outside borders with different buffer values

Buffer size (meters)	**Addresses outside borders (pc)**	**Addresses outside borders (%)**
0	62 982	2,652
50	54 582	2,299
100	48 229	2,031
150	43 416	1,828
200	39 936	1,682
250	37 471	1,578
300	35 561	1,498
350	34 040	1,433
400	32 640	1,375
450	31 377	1,321
500	30 188	1,271
550	29 061	1,223
600	28 091	1,183
650	27 103	1,141

The outlier numbers are additionally visualized in Figure 24, showing a steep initial decline in outlier numbers, which stabilizes to a near-constant decline after approximately 250 meters of buffering.

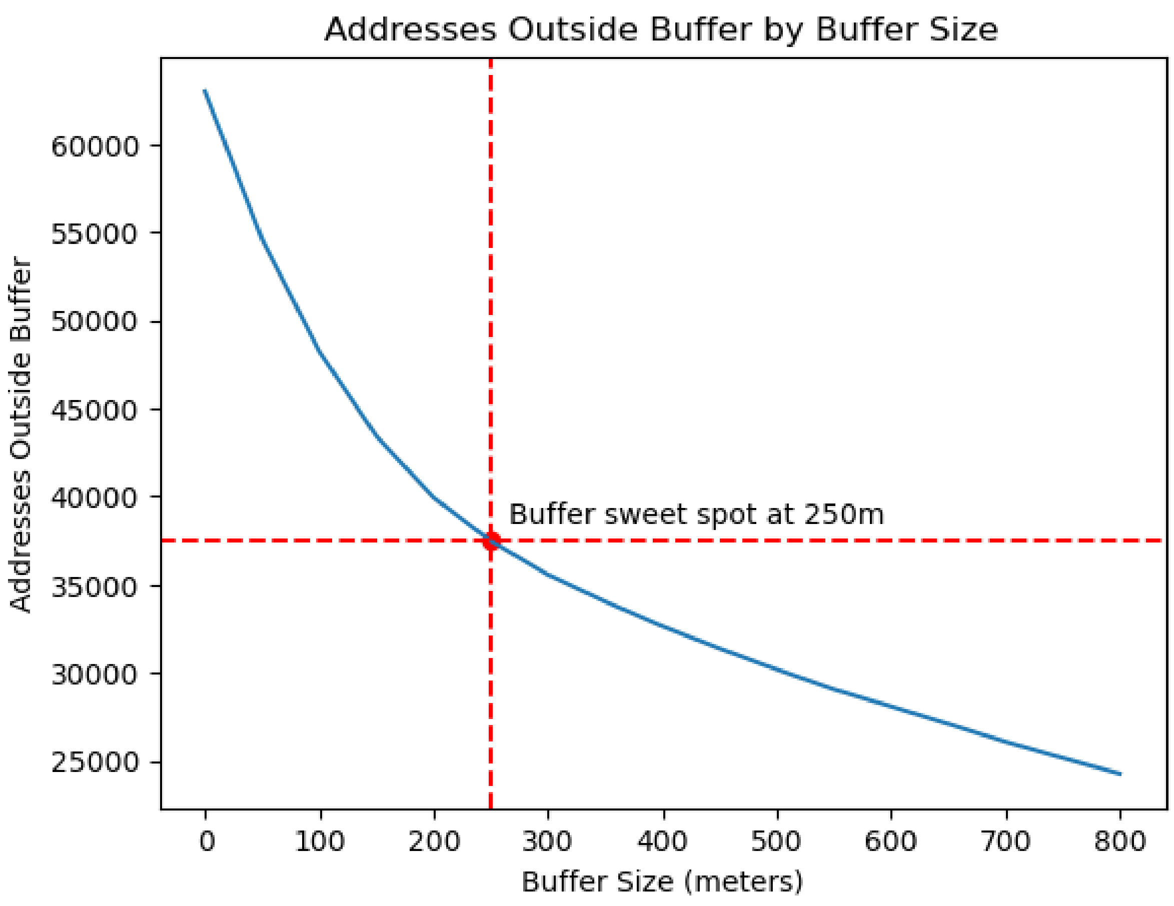


Figure 24. Outlier addresses at different buffer distances

Increasing the buffer distance initially captured many previous outlier points, as the buffered municipality borders started to catch up with and exceed the actual land borders of islands. The sweet spot for buffering was 250 meters, as after this distance the buffers started to mostly include addresses from neighboring islands. Figure 25 shows the Nagu island area with a 250-meter buffer and the resulting outlier points. While some individual addresses still remain outside the borders (such as the cluster at the southern end of Kirjais isle), most of the locations at the edges of the main islands have been captured in the dataset.

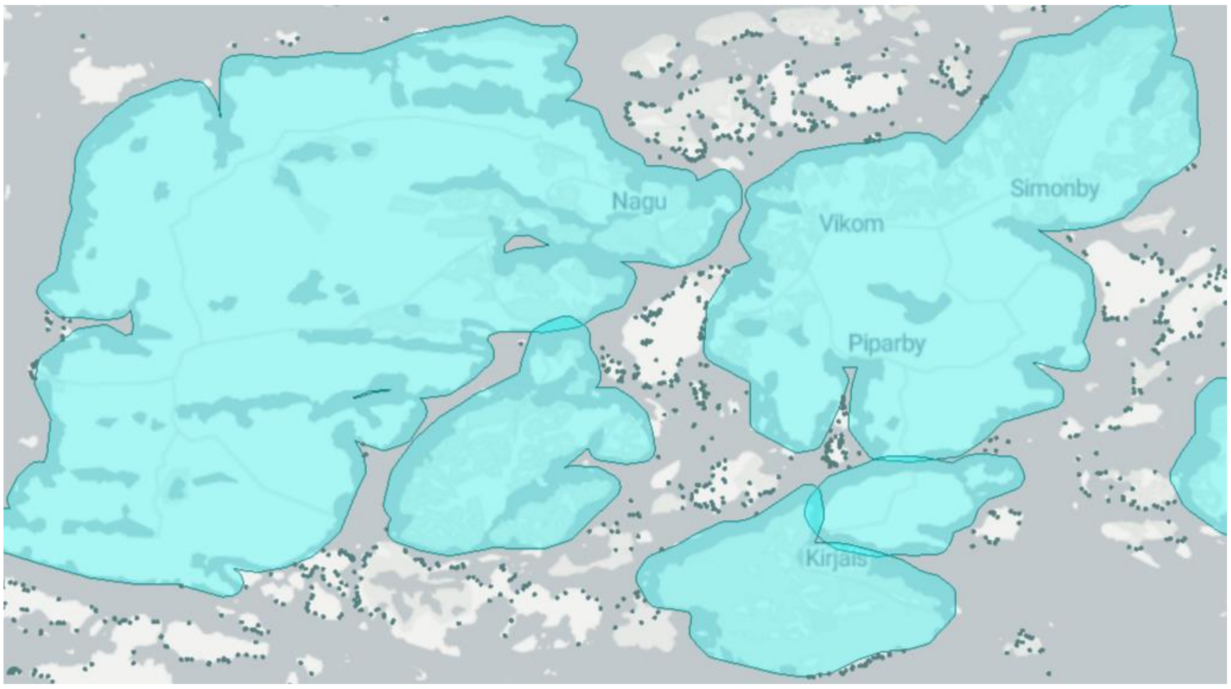


Figure 25. Nagu island address outliers with 250-meter buffer

The final iteration of the outlier removal included municipality borders buffered to 250 meters, which resulted in 2 325 091 rows of data (97,91 %) remaining within the municipality borders. These data rows were further checked for duplicate addresses, with a duplicate address considered one where the combination of the following attributes was not unique:

- **GATUNAMN**: street name in Swedish,
- **KATUNIMI:** street name in Finnish,
- **KATUNRO:** street number,
- **KUNTANIMI:** municipality name,
- **POSTINRO:** postal code,
- **POSTITOIMIPAIKKA:** postal location in Finnish, and
- **POSTITOIMIPAIKKA_SWE:** postal location in Swedish.

The complete dataset of 2 325 091 addresses, contained 1 853 381 unique addresses, and 471 710 (20,29 % of the dataset) were duplicates of another address. An examination of the data showed three main reasons for duplicate lines: community garden plots, missing address attributes, and plots with multiple buildings. The largest number of duplicate addresses was observed for Rajasillantie 11, 01730 Vantaa, with 238 copies of the same address in the dataset. Rajasillantie 11 seems to contain a tight conglomeration of small cottages and gardens, each with their own copy of the address. Other addresses with significantly high numbers of duplicates also had similar cottage communities. Figure 26 shows the map layout and satellite image of Rajasillantie 11 area.

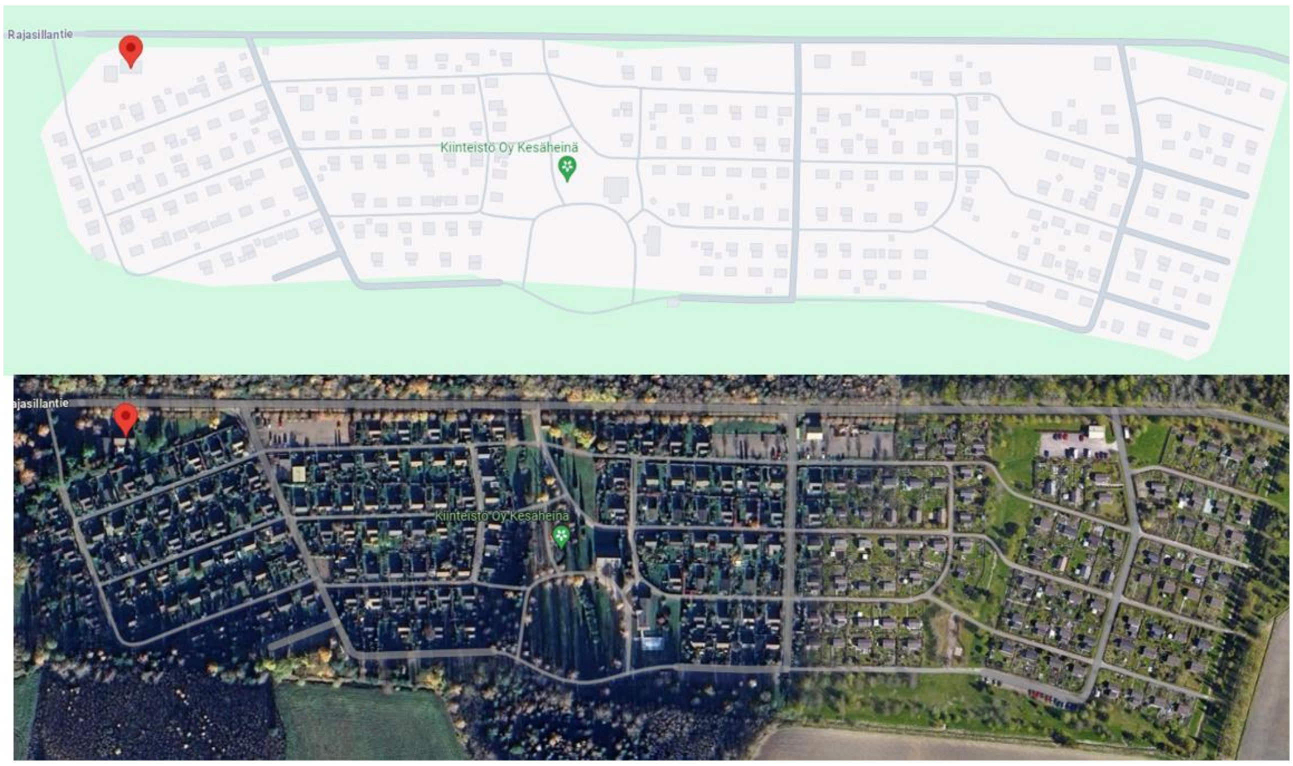


Figure 26. Map and satellite images of Rajasillantie 11

The second reason for a high number of duplicates was missing data, often the street number of the street name. As the model relies on street name and number information, any rows with missing street names and numbers will be removed from the dataset. The duplicate lines with missing data also geolocate to different places on the map, potentially large distances away from each other when the street name is missing, and therefore are not high enough quality data points to be usable.

In approximately 95% of the cases, an address contained at most four rows of duplicate data. As each building is separately documented in the building registry, each individual building on a plot generates a new row of data. Manual checks on data confirmed that duplicates in these cases consisted of a plot with a main building and one or more additional buildings, such as a garage or a shed. The buildings were in close proximity to each other, and the coordinates of any of the buildings could be used to train the model.

Duplicate rows of data were dropped from the dataset with no further examination on which data point or which specific building was retained. The community cottage areas had the most deviation in the geographic location but were so few in number that the overall effect on the dataset is negligible. Most cases of duplicates consisted of buildings so tightly grouped together that using any of the data points would provide an accurate estimate of the address location. Data points with missing data do not need to be considered in terms of duplicates, as the data points will be dropped at a later step.

The dataset resulting from the steps above contains 1 853 381 unique addresses from the Finnish building registry, located at most 250 meters away from a Finnish municipality’s land area. This dataset will be referred to as “actual addresses”.

5.2.2 Road geometry endpoints

The base data for road geometry endpoints is the 29.05.2024 dated Digiroad dataset for entire Finland. The data package is available as a compressed zip-package, containing individual shapefiles for different data types in a folder hierarchy split by Finnish provincial divisions. Files named DR_LINKKI.shp contain road geometries and related properties for their respective areas. Each DR_LINKKI.shp file was extracted from the package and combined to form a complete dataset of the entire Finnish street and road landscape. The unprocessed dataset of road links contained all streets and roads regardless of their method of traversal. Figure 27 shows the unfiltered road geometries of a single DR_LINKKI file, clearly showing commuter ferry routes in the archipelago, alongside regular roads and paths.

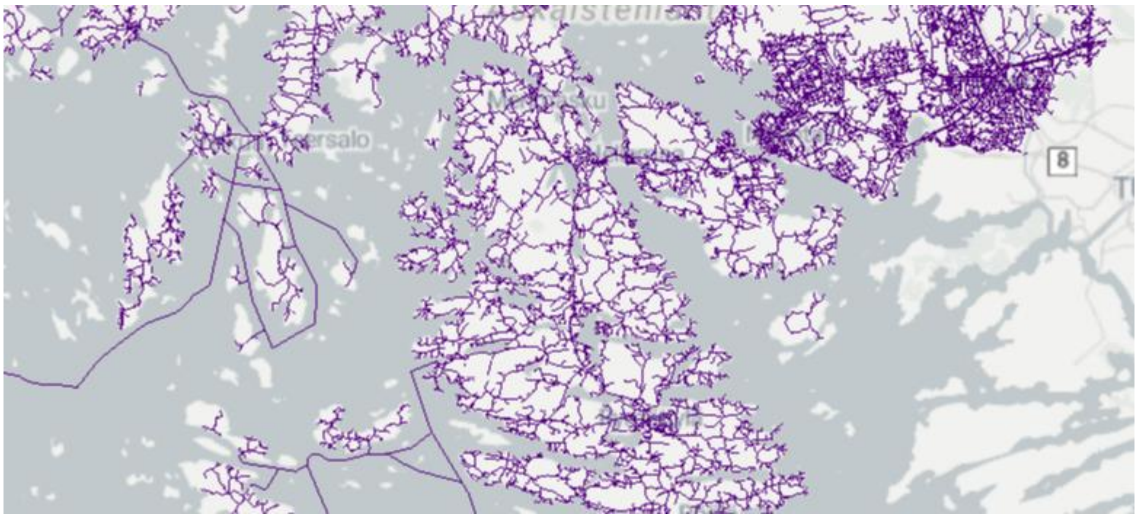


Figure 27. Unfiltered road link data, including ferry routes

Digiroad (2018) dataset specification provided crucial understanding on the dataset attributes. The dataset was filtered to only contain rows where LINKKITYYP <= 5, retaining all roads that are typically traversed with cars, except auxiliary parts such as highway ramps or resting areas at the sides of the roads. The assumption behind this filtering is that all street addresses considered in this research are traversable by car, and no buildings are built exclusively along cycling paths or forest paths. Figure 28 shows the resulting road geometry links in Nagu island area.

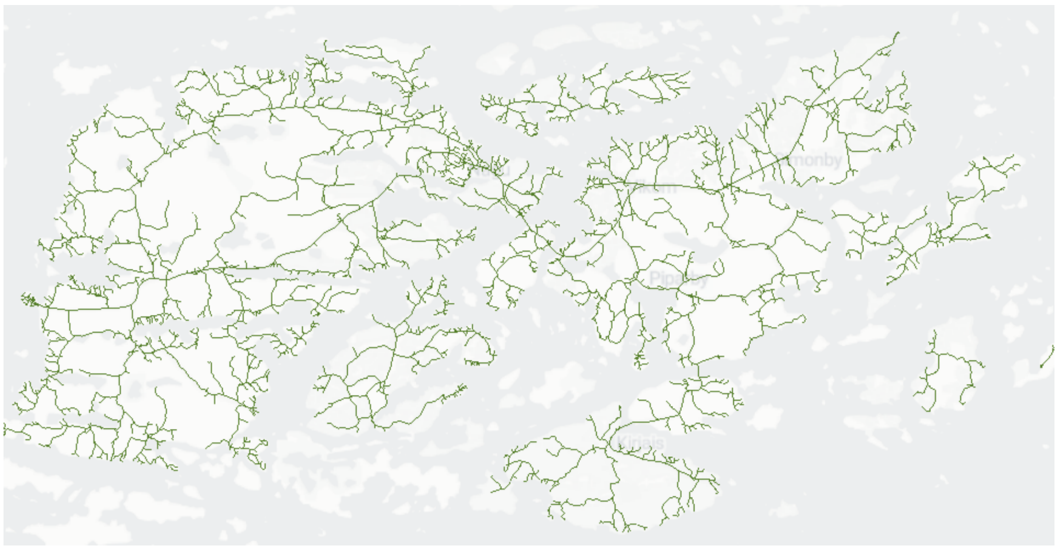


Figure 28. Road links around Nagu island

The remaining dataset contained a large number of unnamed road segments, spanning across the whole of Finland. A closer look at the road segments suggests that the roads are either short unnamed access roads, or internal roads of a restricted area, such as factory premises or harbors. Image 29 shows multiple examples of both types of roads. Unnamed road segments were discarded as unusable for dataset enrichment purposes, as address information is mandatory for model training purposes.

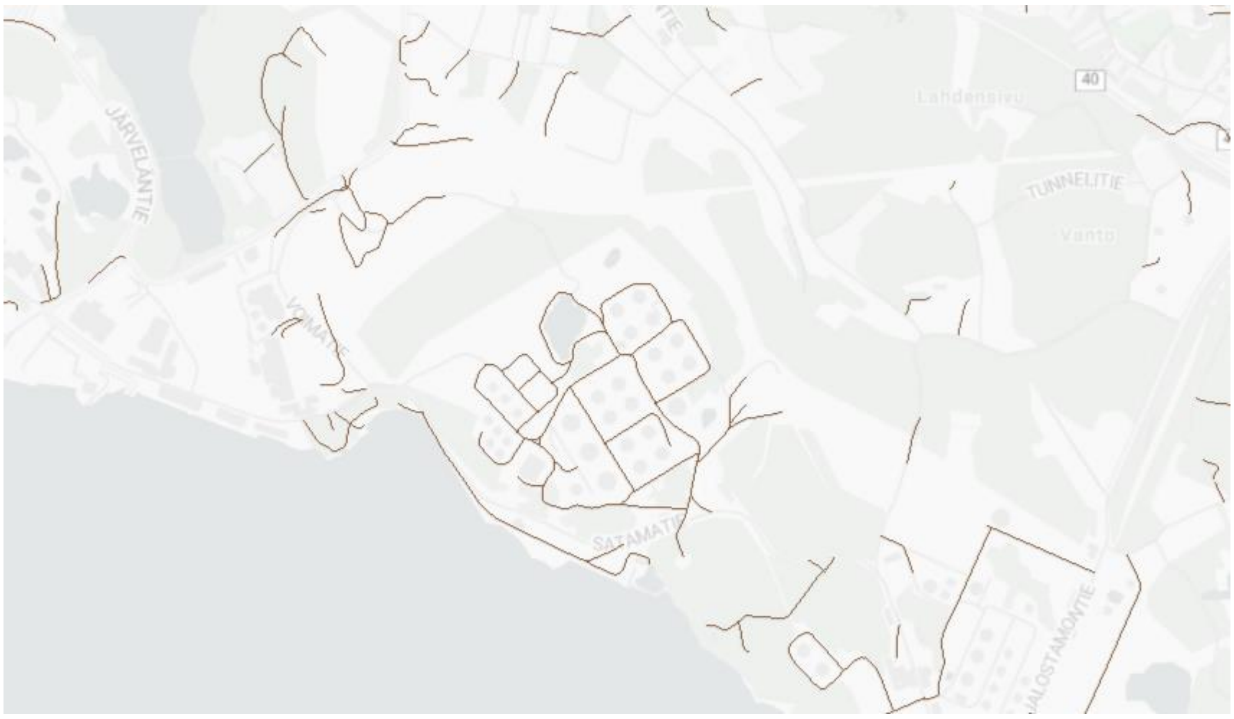


Figure 29. Unnamed road segments and internal roads

The initial plan was to examine the end points of a road geometry to determine the minimum and maximum extents of the road. This proved to be a challenge, as even after combining the geometries of individual roads, not all formed coherent geometric structures with a single starting and end point, preventing a geometric approach. Roundabouts and split roads also posed their challenges in mathematical approach. Figure 30 shows an example of a main road running from Pargas (Parainen) to Korpo (Korppoo), crossing two ferry routes along the way.

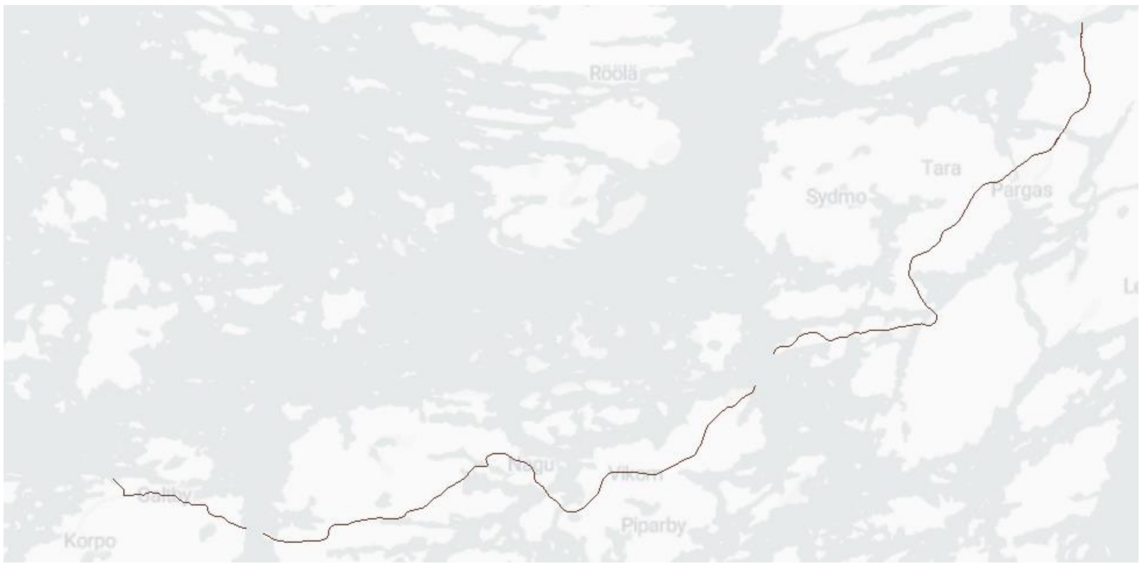


Figure 30. Fractured road geometry separated by two ferry segments

The implemented approach for determining coordinates for road endpoints involved examining the road number properties of each road segment separately. The road link data contained fields ENS_TAL_O and ENS_TAL_V to denote the first street number on the right and the left sides of the road, respectively, as well as the corresponding VIIM_TAL_O and VIIM_TAL_V fields for the last street numbers. The street numbers were in the scope of the individual road segment. The house numbers matched the direction of the road segment geometry object, as in, ENS_TALO_O/V always denoted the location at the first coordinate pair of the geometry object and VIIM_TAL_O/V matched with the last coordinate pair of the geometry object. In some cases, the number in ENS_TALO_O/V was larger than VIIM_TAL_O/V, indicating that the road geometry is documented in a reverse direction.

Individual road segments were grouped by KUNTAKOODI (municipal code), TIENIMI_SU (Finnish street name) and TIENIMI_RU (Swedish street name) properties to identify all road segments belonging to the same complete road structure. For each complete road structure, the smallest street number across all member road segments and its corresponding coordinate pair was extracted. The same was repeated for the largest street number. Figure 31 shows an area around Norrby, with road starting points (blue) and end points (red) that were extracted in this manner.

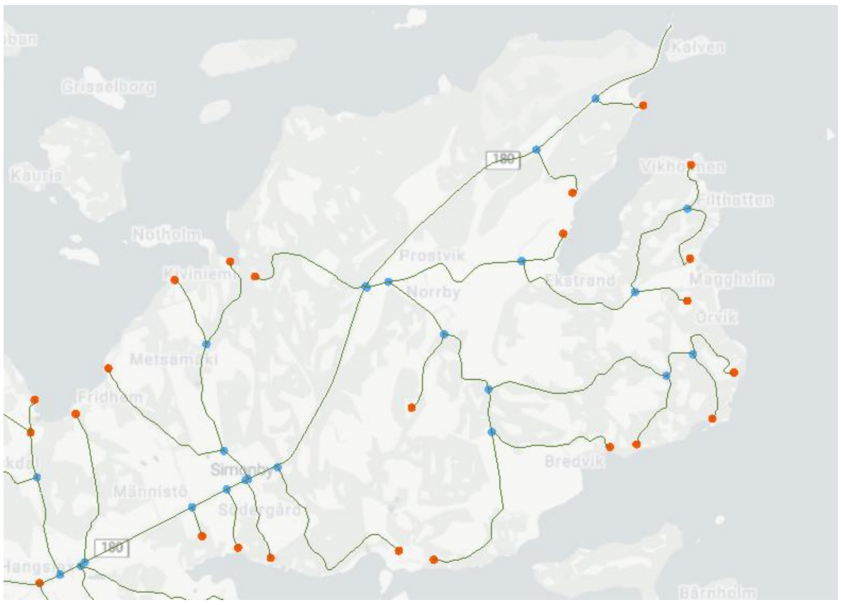


Figure 31. Road starting points (blue) and end points (red)

The starting point and end point coordinates for each complete road were paired with a street number. The starting point street number was set to 0, and the end point street number was set as one higher than the last street number across all road segments of that particular road. This ensures that all possible street numbers fall between the generated start and end point street numbers of the road, leaving the generated address points to represent the extreme extents of the street. Figure 32 shows the locations of address coordinates generated near Nagu in this manner.

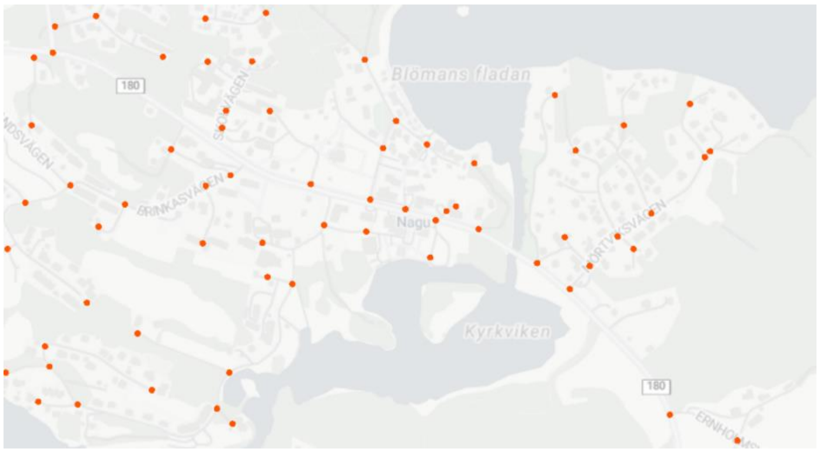


Figure 32. Generated address points near Nagu

As the Digiroad road segment data only contained street name, number and municipality code information, further enrichment of the data was necessary to create a dataset equivalent in structure to the actual addresses. The generated address locations were spatially joined to Finnish postal code regions, enriching the address data with postal code and postal location information. Municipality code was used as a key value to populate the municipality name from an equivalency list maintained by Statistics Finland.

The resulting dataset has the following features to be used for address prediction:

- **KUNTAKOODI**: municipality code,
- **TIENIMI_SU**: street name in Finnish,
- **TIENIMI_RU**: street name in Swedish,
- **KATUNRO**: street number,
- **POSTITOIMIPAIKKA_FI**: postal area in Finnish,
- **POSTITOIMIPAIKKA_SE**: postal area in Swedish, and
- **KUNTANIMI**: municipality name.

The target variables consist of the x- and y-components of a coordinate:

- **LATITUDE**: latitude coordinate, and
- **LONGITUDE**: longitude coordinate.

This complete dataset is later known as the “generated addresses” dataset, containing a total of 431 916 road segment endpoints.

5.3 Exploratory data analysis and preprocessing

5.3.1 Spatial distribution and sampling

While some exploratory data analysis and outlier handling was already carried out during the refinement of actual addresses and generated addresses, this step aims to look at the datasets from a pattern-detecting point of view, rather than data quality perspective that was used in earlier sections. The first step was to create a density heatmap to examine the spatial distribution of data points across Finland. A 5 x 5 kilometer grid was superimposed on top of the buffered municipality border geometries to create a Finland-shaped layer of square cells. Each actual address and generated address was spatially joined with a grid cell, allowing for the calculation of total number of data points per cell. The resulting distribution of datapoints for actual addresses and generated addresses is shown in Figure 33.

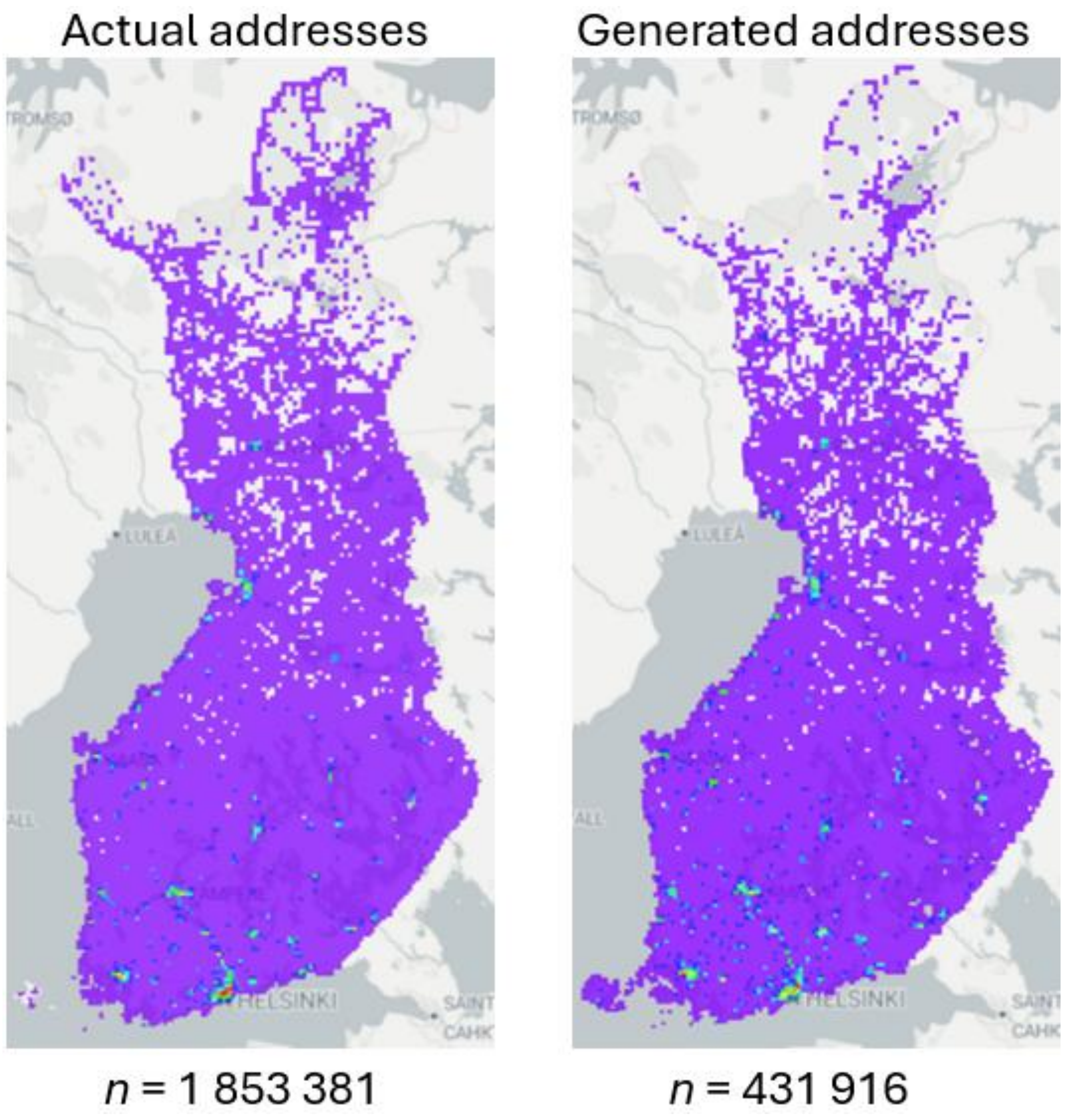


Figure 33. Spatial distribution of actual addresses and generated addresses

In both datasets, the spatial distribution of data points is extremely uneven, with large portions of Lapland containing no data points. Conversely, the largest population clusters (Helsinki metropolitan area, Turku and Tampere in the south, Oulu in the north) show high concentrations of data points in both datasets. The generated dataset shows slightly less variation in the countryside in central and southern Finland, with slightly more blue and green coverage than in the case of actual addresses. Åland islands in the southwest archipelago especially seems to have more roads and potential addresses, than actual documented addresses. Lapland on the other hand likely has such long distances that roads are sparse and long, and generated datapoints are few and far in between, while the actual addresses are speckled along the sparse and long main roads.

Following Kanevski's (2009) advise on spatial distribution of data, the actual address dataset was sampled so that a maximum of 130 addresses were picked from each 5 x 5 kilometer cell of the grid randomly. The resulting thinned dataset contained a total of 796 727 addresses.

Figure 34 shows the sampled data points from an area around Turku, Finland. Some areas near peninsulas or islands show a denser sampling of data points, likely resulting from a square cell falling mostly on water-based areas with no valid addresses elsewhere, leading to a clustered distribution of the addresses. Scaling the number of sampled addresses with the proportion of land area within a cell would mitigate the issue, but as the sampling only affected some areas along the coastline, proportional scaling was not implemented in this research.

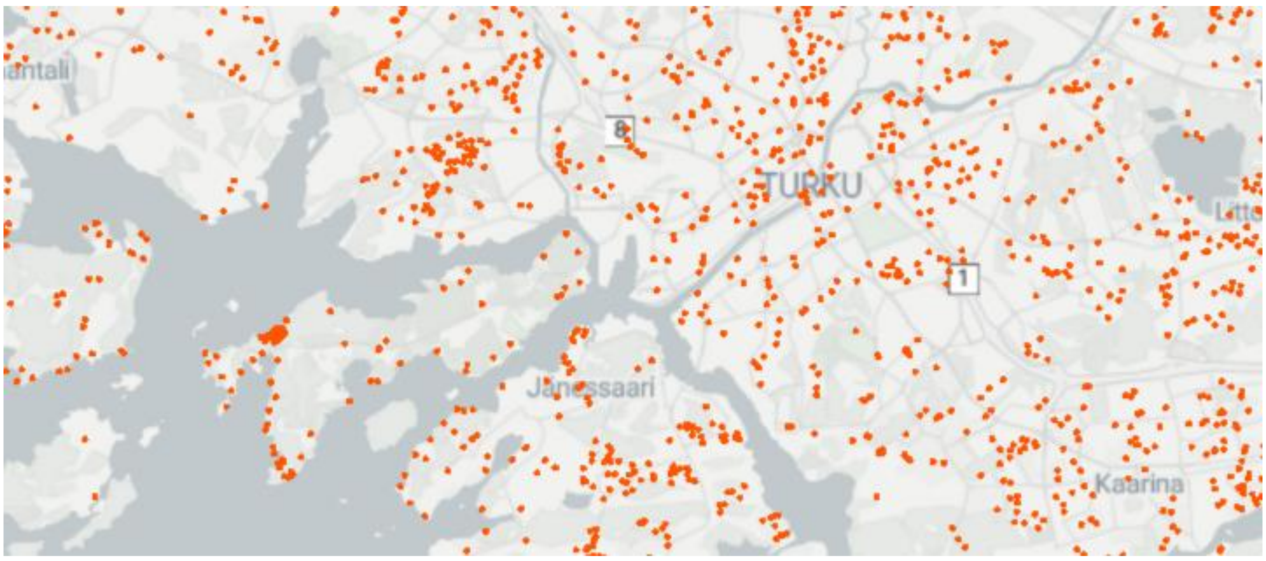


Figure 34. Sampled addresses near Turku city

Re-examining the density heatmap with the sampled addresses shows a much more even spatial distribution of data, as seen in Figure 35. Even though Lapland is still sparsely populated, and Helsinki metropolitan area still shines red, the spatial distribution of data points is much more even across the central and southern Finland.

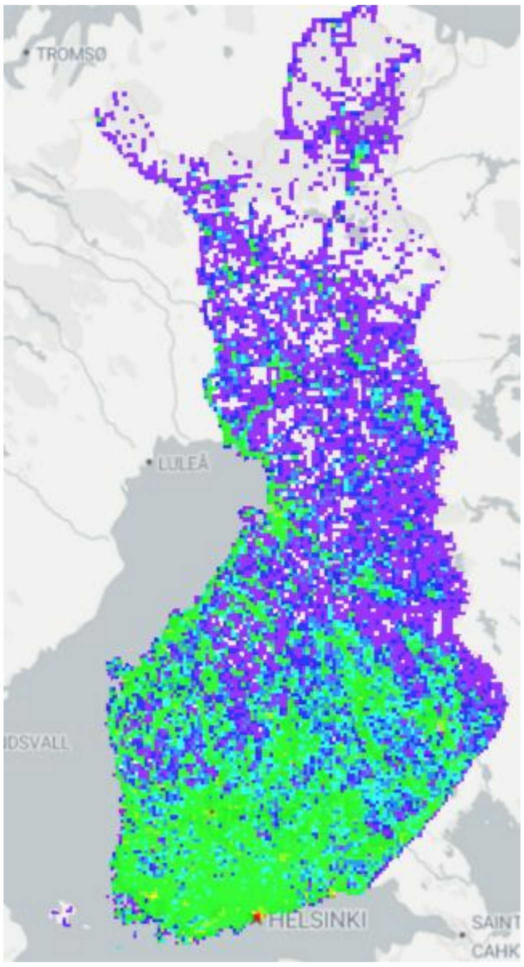


Figure 35. Spatial distribution of sampled addresses

The generated addresses were not subjected to the sampling, as each municipality-street name combination only has two rows of data in the generated dataset, and any sampling would result in a loss of information. The actual addresses dataset was a much larger dataset in terms of row counts, and combining a more evenly distributed actual address dataset with the unevenly distributed generated dataset dilutes the two, leading to a dataset that is less evenly distributed than the sampled set, but better distributed than the generated dataset. Another potential sampling method would have been to sample smaller amounts of actual addresses from grid squares with high numbers of generated addresses, leading to a more spatially evenly distributed dataset.

The spatial distribution of the combination of actual and generated address datasets is show in Figure 36. While major population centers still stand out as bright spots in the distribution, the overall spatial distribution of data is more even across the country.

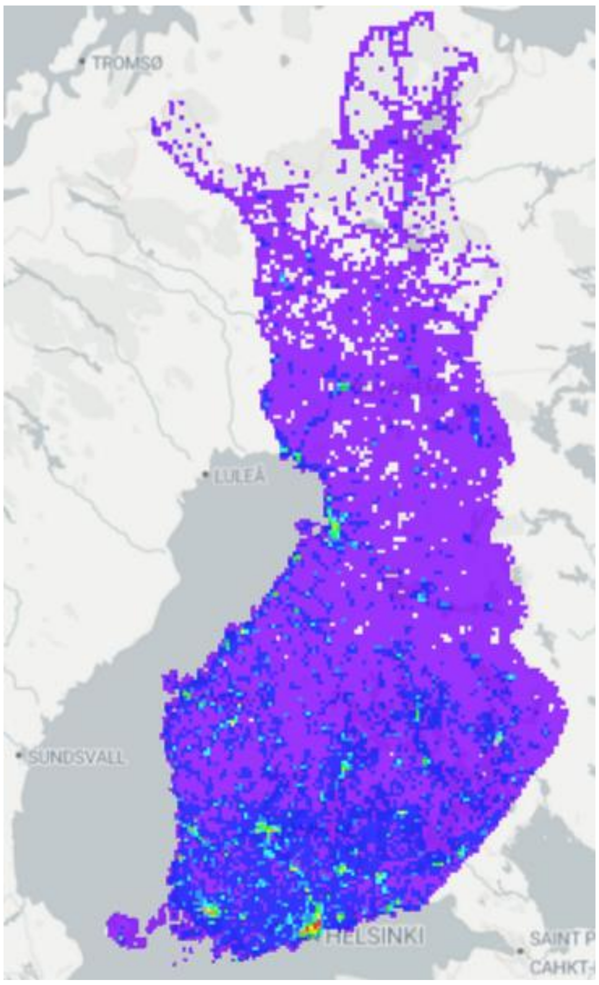


Figure 36. Spatial distribution of sampled actual addresses and generated addresses combined

The generated and sampled actual address datasets were combined into a single dataset to be used for machine learning training, though a separate property was added to each row to indicate whether the data was generated. Figure 37 shows a close-up of data points near Turku city that were included in the final dataset.

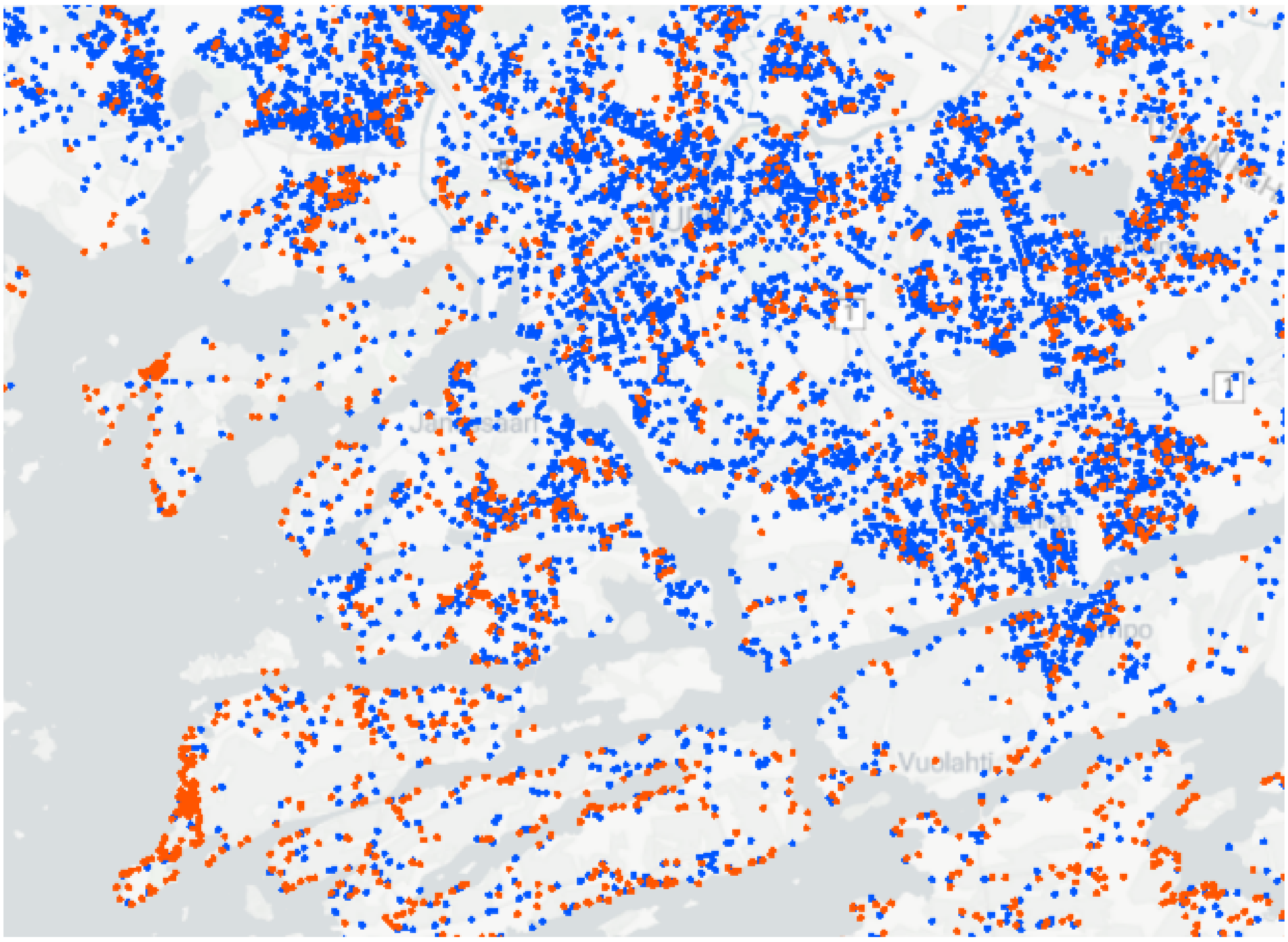
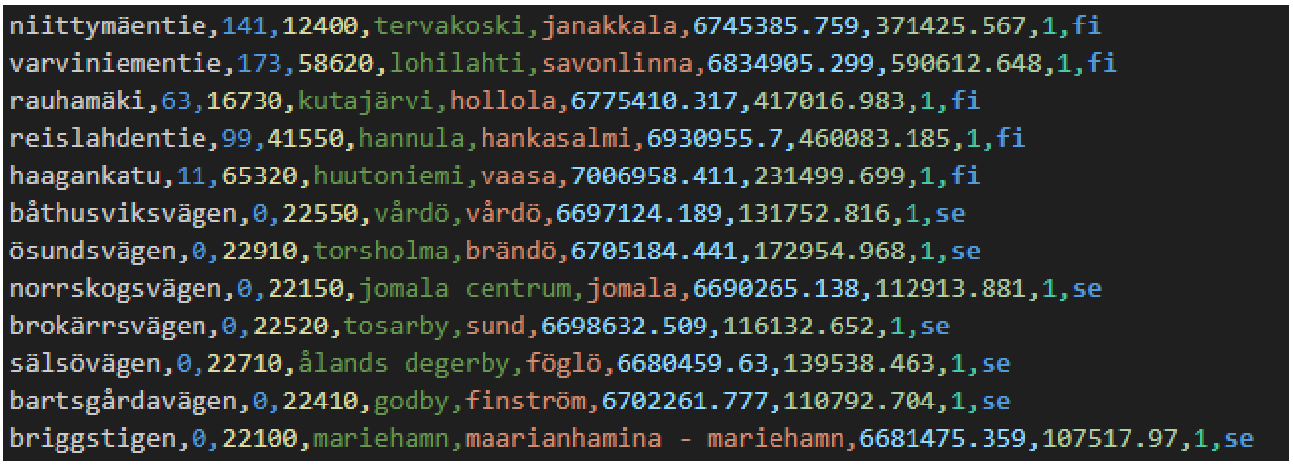


Figure 37. Sampled (red) and generated (blue) data points around Turku city

5.3.2 Address data preprocessing

Each address row within the dataset had a separate column for Finnish and Swedish versions of the street name and the postal region. Approximately 3,44% of the Finnish and 94,76% of the Swedish street name and postal region columns did not have a value. Some data rows have both Finnish and Swedish street names and postal regions. These values conform to expectations, as Finland is a bilingual country with Finnish being the only language for the population especially inland, and Swedish being a major or even the dominant language in most coastal regions. Because of this, many of the coastal regions have two versions of the same address: one in Finnish, one in Swedish. For example, Loviisa area has an address that is written as “Sahasalmi 1, 07930 PERNAJA” in Finnish, and “Sågsundet 1, 07930 PERNÅ” in Swedish. However, some of the addresses in Swedish majority area have been documented in the columns reserved for the Finnish versions of the name, even though the street names and postal region names are clearly in Swedish, limiting the ability to effectively vectorize the words without a separate layer of language detection later.

Each row of data was split into two separate rows, with one row receiving the Finnish street name and postal region and the other row receiving the corresponding Swedish data. All other attributes between the rows were the same. Subsequently, all rows containing nulls were dropped, resulting in a dataset of 1 250 752 rows, which contained all Finnish and Swedish language addresses for the dataset as their own rows. Figure 38 shows a snippet of rows from the dataset.



```
niittymäentie,141,12400,tervakoski,janakkala,6745385.759,371425.567,1,fi
varviniementie,173,58620,lohilahti,savonlinna,6834905.299,590612.648,1,fi
rauhamäki,63,16730,kutajärvi,hollola,6775410.317,417016.983,1,fi
reislahdentie,99,41550,hannula,hankasalmi,6930955.7,460083.185,1,fi
haagankatu,11,65320,huutoniemi,vaasa,7006958.411,231499.699,1,fi
båthusviksvägen,0,22550,vårdö,vårdö,6697124.189,131752.816,1,se
ösundsvägen,0,22910,torsholma,brändö,6705184.441,172954.968,1,se
norrskogsvägen,0,22150,jomala centrum,jomala,6690265.138,112913.881,1,se
brokärrsvägen,0,22520,tosarby,sund,6698632.509,116132.652,1,se
sälsövägen,0,22710,ålands degerby,föglö,6680459.63,139538.463,1,se
bartsgårdavägen,0,22410,godby,finström,6702261.777,110792.704,1,se
briggstigen,0,22100,mariehamn,maarianhamina - mariehamn,6681475.359,107517.97,1,se
```

Figure 38. Dataset sample

5.3.3 Address lemmatization and vectorization

Address lemmatization and vectorization aims to allow for pattern detection in regions with theming address names. “Niittymäentie”, for example, consists of the base words of “niitty”, “mäki” and “tie”. Attempting to vectorize the whole of “Niittymäentie” would likely produce no results, but separate vectorizations of the individual components are much more likely to succeed. The first step was to split addresses into a prefix and a suffix, with the suffix often representing a Finnish or Swedish equivalent of “road”, “street”, “bend”, “hill”, and so on. The address splitting was carried out by identifying common suffixes in street names, splitting the suffixes from the street names, inspecting unsplit street names for new potential suffixes and iterating the process over until all identifiable suffixes were covered. Approximately 98,71% of the 1 250 752 could be split into prefix and suffix in this manner. For the remaining addresses, the prefix and the suffix were both set to the original street name. Address splitting resulted in 82 044 unique address prefixes and 142 unique address suffixes for Finnish language, and 18 384 address prefixes and 158 address suffixes for Swedish language.

Trankit module for Python was then used to create lemmas for both the address prefix and the suffix columns, for both Finnish and Swedish languages. TurkuNLP’s open implementation of a closest word finder demo was used as a base to implement a code snippet, which fetched the vector forms of all lemmatized prefixes from TurkuNLP’s Finnish language word embeddings. Not all prefixes and suffixes could be vectorized, in which cases a 200-dimensional vector with zeros for all values was used instead.

The same code and word embedding file was used for both Finnish and Swedish languages, and as an earlier test showed that the Swedish word vectors do not function as reliably as the Finnish and English counterparts, the lack of dedicated Swedish NLP imposes a limitation in the ability to group Swedish language addresses by theme. An improved version of this process would aim to detect the actual language of the prefixes and suffixes and use NLP tools of the corresponding language. The implementation would have been significantly more complex and was left out of scope of this research.

5.3.4 Scaling and splitting

All non-numeric address information was dropped from the dataset, consisting of columns for street name, postal location, municipality, prefix lemma, suffix lemma and language indicator. The remaining columns consisted of street number, postal code, coordinate values, indicator of whether address is generated, and vectorized forms of the street name prefix and suffix. Standard scaler was used to scale postal codes and street numbers for model training, though for Random Forest algorithms this would not have been mandatory.

The dataset was split into train, test and validation datasets. All generated address rows were included in the training dataset, whereas the actual addresses were split three ways into training, testing and validation datasets using a static random seed. After the division, the training dataset consisted of a total of 875 627 rows, and test and validation datasets both had 191 478 rows. The respective percentage division between the datasets was 69,57% for training, and 15,21% for test and validation each. Latitude and longitude columns were split as target variables for each dataset, all remaining columns used for training. The training dataset consisted of 404 columns and 875 627 rows, resulting in a sample-feature-ratio of approximately 2 167, which by Zhu et. al. (2013) should allow for a great learning potential.

The training dataset was used to train each machine learning model. The test dataset was used to assess the accuracy of the model as the hyperparameters of the models were tuned at each point in the training process. The validation dataset was reserved to be used as the final measurement of the model's accuracy and was not used at any point during the training process. Figure 39 visualizes the interactions of each dataset portion with the model training process.

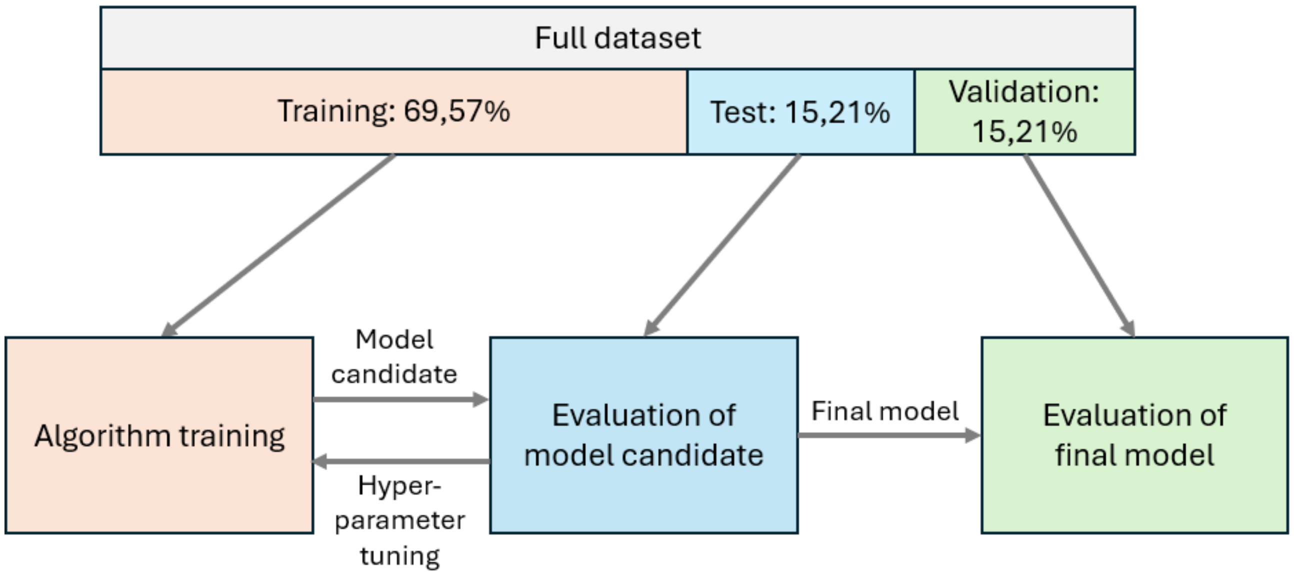


Figure 39. Dataset splits and utilization in training and evaluation processes

5.4 Model training and optimization

5.4.1 Random Forest Regression

The first machine learning model of choice was Random Forest Regression, as the algorithm is natively capable of producing multi-variable output and capable of capturing complex relationships.

Random Forest model training times also ranged from a few minutes to a few hours, allowing for multiple iterations as model parameters were tuned. Mean squared error was used as a metric for evaluating the accuracy of the model, with Euclidean distance between the actual and the predicted coordinates used as a second evaluation metric to allow for practical evaluation of how useful the predictions are.

Euclidean distance was implemented as a helper function, returning the result of the calculation of the following formula:

$$\sqrt[2]{(\text{predicted longitude} - \text{actual longitude})^2 + (\text{predicted latitude} - \text{actual latitude})^2}$$

The resulting metric is a straight point-to-point distance between the prediction and the actual coordinate in units of the coordinate system used, in this case, meters.

The main hyperparameter that was optimized during model training was the number of estimator trees within the Random Forest algorithm. Forest sizes of 5, 10, 15, 25, 50 and 150 trees were used to train models, the results of which are in Table 7 below.

Table 7. Model metrics at different number of estimators

Estimators	**Mean Squared Error**	**Mean Absolute Error**	**Max distance (m)**	**Min distance (m)**
5	5 119 375	1 157	317 372	0
10	4 672 625	1 098	317 372	0
15	4 530 880	1 075	317 380	0,25
25	4 503 622	1 052	317 539	0,53
50	4 374 721	1 029	317 528	0,49
150	4 337 439	1 012	317 465	0,31
250	4 326 807	1 008	317 419	0,43
300	4 325 980	1 006	317 426	0,58
350	4 328 951	1 007	317 422	0,34
450	4 334 252	1 005	317 406	0,73

The effect on estimators on model loss metrics is visualized in Figure 40.

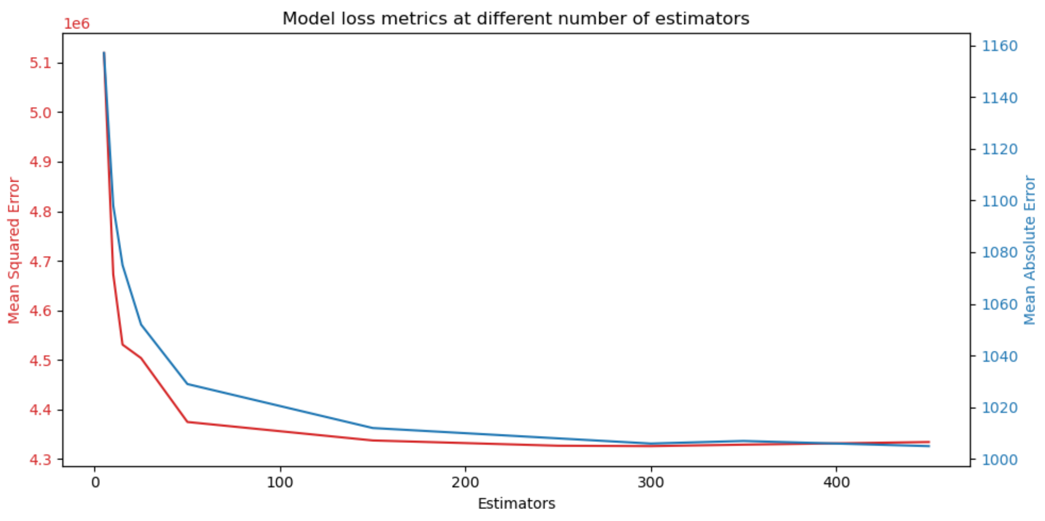


Figure 40. Model loss metrics at different estimator values

The evaluation results of the model show a rapid initial decrease in loss metrics when more estimators are added, though the scale of the change obscures the more detailed changes at the larger number of estimators. Figure 41 shows the loss metrics for 50 or more estimators, suggesting that 300 estimators would be most optimal for this model.

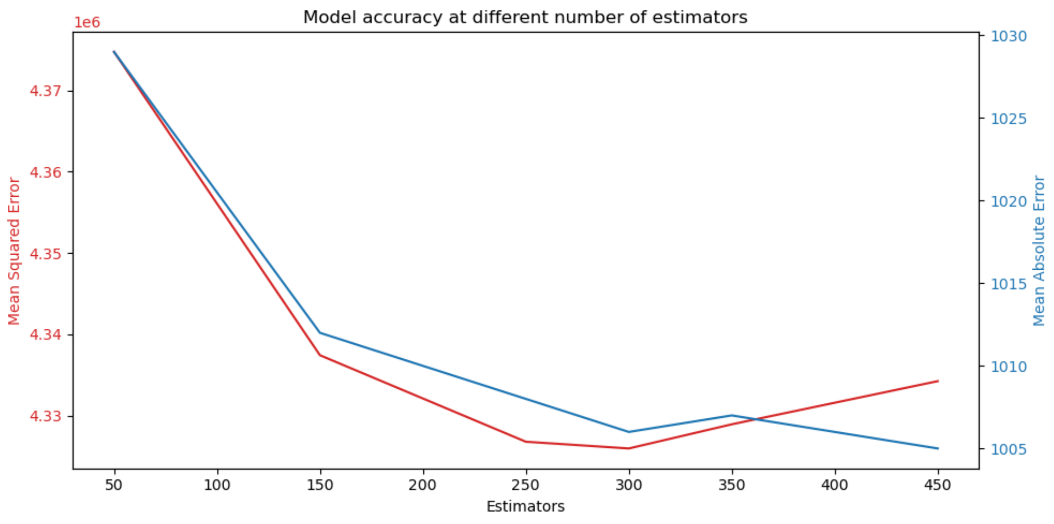


Figure 41. Model loss metrics for 50 or more estimators

The initial versions of the Random Forest Regression model did not limit the tree depth of the estimators. Tree depth can be limited to a static maximum to avoid overfitting and to retain generalizability of the model to new data. Maximum tree depths were limited as an experimentation on the effect of model accuracy, and as a potential optimization method to improve model performance overall. The outcomes of limiting the maximum tree depth of a 300-estimator model are listed in table 8 below.

Table 8. Loss metrics at different maximum tree depths for a 300-estimator model

Maximum depth	Mean Squared Error	Mean Absolute Error
3	4 682 024 081	85 365
6	681 613 775	31 861
9	117 899 249	12 495
12	28 357 397	5 603
15	14 604 668	3 554
18	10 380 279	2 583
21	7 920 597	1 943
24	6 406 338	1 538
27	5 498 201	1 298
30	4 971 367	1 162
33	4 668 817	1 088
36	4 500 548	1 048
39	4 405 094	1 027
42	4 360 834	1 016
45	4 338 341	1 010
48	4 328 715	1 008
51	4 325 531	1 007
54	4 322 721	1 007
57	4 323 230	1 007
60	4 323 336	1 007

The visualization of the loss metrics shows a rapid decrease when increasing tree depths, eventually evening out at deeper tree levels, as shown in Figure 42.

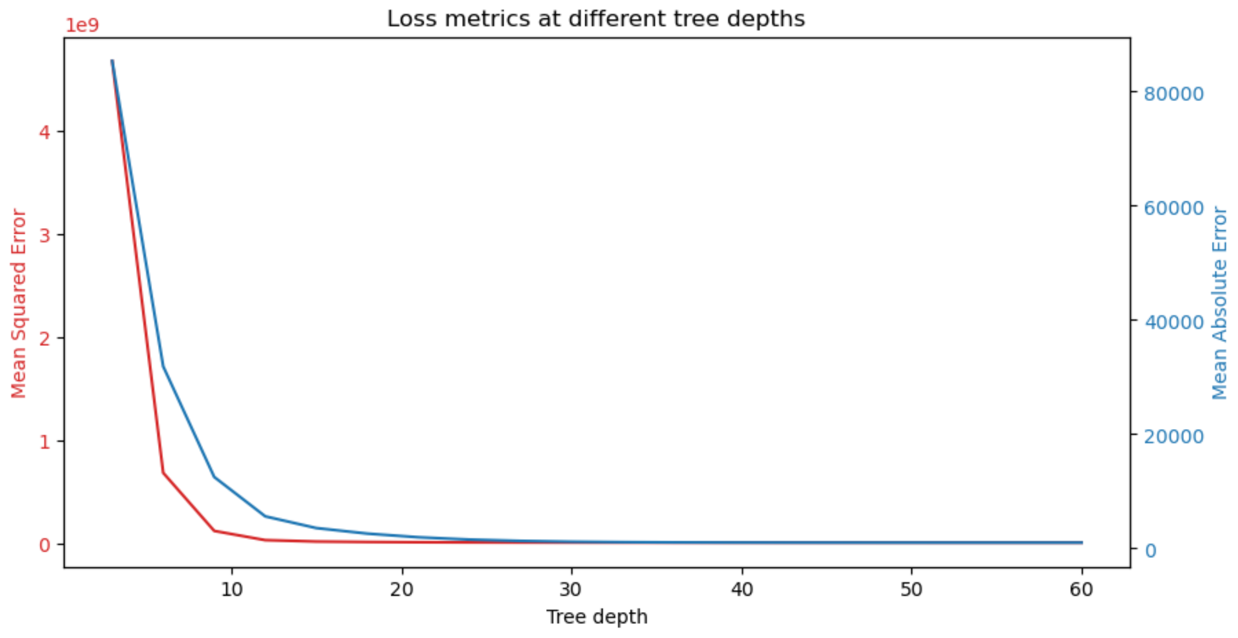


Figure 42. Mean squared error at different tree depths

Observing the tail end of the graph in Figure 43 shows that the loss metrics reach a minimum at tree depth of 54 levels, though the differences to the surrounding tree level depths are very minor.

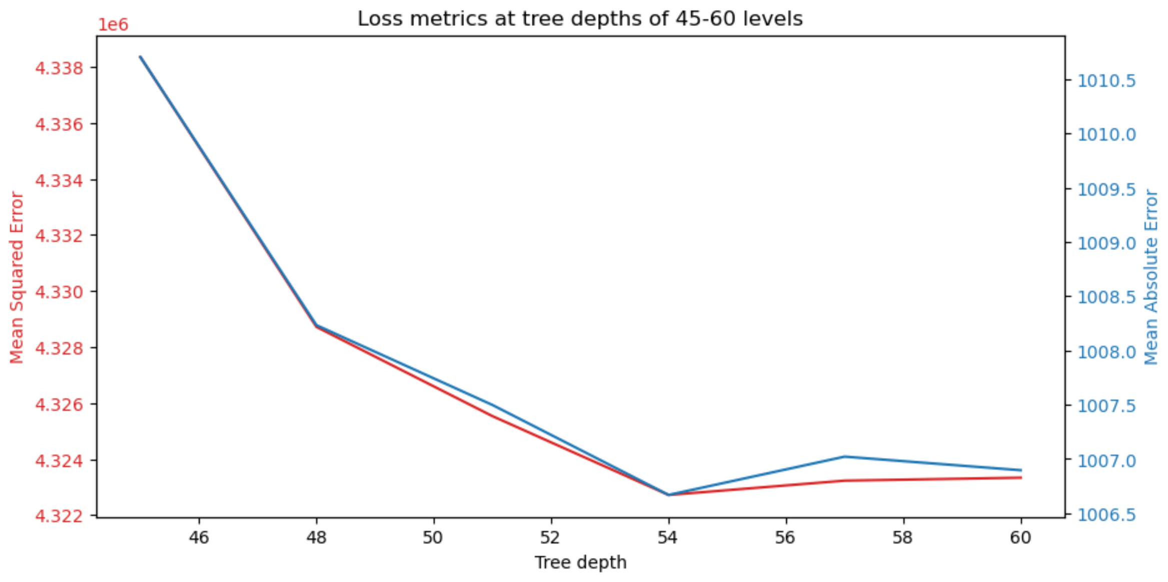


Figure 43. Loss metrics at higher maximum tree depths

Another optimization approach involved the disabling of bootstrapping. By default, Random Forest Regression uses bootstrapping to split the training dataset into smaller samples, from which each decision tree is then generated. This has the potential of leaving decision trees with no knowledge of the incoming data, leaving them unable to produce meaningful predictions. With bootstrapping disabled, each decision tree within the random forest is trained with the complete training dataset, which was intentionally designed to contain information on all street names in Finland.

The models trained with different numbers of estimators were re-trained with bootstrapping disabled. The training time increased significantly but was mitigated with heavy parallelization. Model sizes also grew correspondingly. Disabling bootstrapping increased root mean squared error values across all estimator counts, but simultaneously significantly decreased average distances between predictions and actual values. The differences in mean squared errors between bootstrapped and non-bootstrapped models are detailed in Table 9.

Table 9. Comparison of mean squared errors between bootstrapped and non-bootstrapped models

Estimators	**Mean Squared Error, bootstrapping**	**Mean Squared Error, no bootstrapping**	**Difference (absolute)**	**Difference (%)**
5	5 119 375	5 556 681	437 306	8,54
10	4 672 625	5 509 850	837 225	17,92
15	4 530 880	5 507 185	976 305	21,55
25	4 503 622	5 500 211	969 331	21,52
50	4 374 721	5 490 354	1 115 633	25,50
150	4 337 439	5 487 577	1 150 138	26,52
250	4 326 807	5 484 231	1 157 424	26,75
300	4 325 980	5 483 286	1 157 306	26,75
350	4 328 951	5 483 636	1 154 655	26,67
450	4 334 252	5 480 935	1 146 683	26,46

Disabling bootstrapping significantly increases the mean squared error metric, indicating that the model makes larger errors across all estimator numbers. This is expected, as the main purpose of bootstrapping is to decrease variance in predictions, resulting in more reliable predictions overall. At smaller number of estimators, the difference between bootstrapped and non-bootstrapped models is smaller than at larger number of estimators. Interestingly though, the mean absolute error metric significantly decreased in comparison to the bootstrapped model, as detailed in Table 10. With bootstrapping disabled, the model is able to reach a better performance when evaluated against mean absolute error, however, inaccurate predictions are also inaccurate by a significantly larger magnitude compared to bootstrapped models.

Table 10. Changes in distance differences between bootstrapped and non-bootstrapped models

Estimators	**Mean Absolute Error, bootstrap**	**Mean Absolute Error, no bootstrap**	**Difference (absolute)**	**Difference (%)**
5	1 157	863	294	25,41
10	1 098	861	237	21,58
15	1 075	860	215	20,00
25	1 052	860	192	18,25
50	1 029	859	170	16,52
150	1 012	859	153	15,12
250	1 008	859	149	14,78
300	1 006	859	147	14,61
350	1 007	859	148	14,70
450	1 005	859	146	14,53

As the aim of this research is to produce a model capable of accurately predicting coordinates for addresses, the implications are two-fold. A model that does not utilize bootstrapping produces on average much more reliable results, but at the same time the far-off predictions are significantly more far-off. Additionally, a meaningful improvement to the model would bring down the maximum distances between predictions and actual coordinates, which has so far remained unchanged. A maximum distance of 317 kilometers covers a significant distance across Finland, see Figure 44.

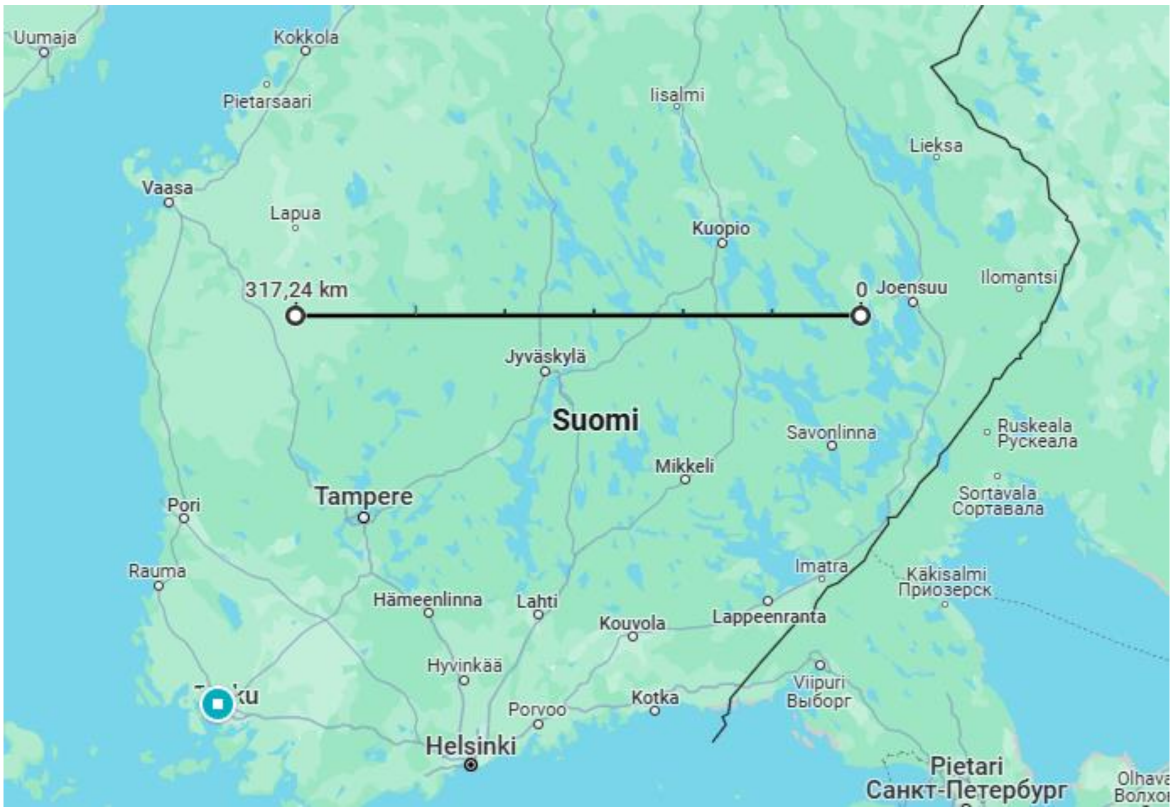


Figure 44. Scale of maximum distance between preditions and actual coordinates

Alternatively, the model would need to be able to assess the confidence of its predictions. Random Forest Regression does not include native calculation of prediction accuracy metrics, so an external implementation must be used to assess confidence. Forestci is a python module which calculates the unbiased variance for Random Forest Regression predictions, however, use of the module in combination with non-bootstrapped models requires the user to manually provide an in-bag matrix for the calculations. The implementation of manual in-bag matrix creations would have extended the scope of this research significantly, so the prediction confidence calculations were performed on the bootstrapped 300-estimator version of the model, rather than the non-bootstrapped model which would have been the preferred choice because of the smaller average distances between points.

Unbiased variances were calculated for predictions by applying forestci module's random_forest_error function to latitude and longitude predictions in turn. Variances were converted into standard deviations by taking a square root of the variance, as using standard deviations as measurements of variability in the data is more intuitive to apply to actual distances: a single unit of standard deviation denotes one meter in distance. Table 11 displays the summary statistics for the standard deviations of individual decision tree predictions.

Table 11. Summary statistics for standard deviations of predictions

Summary metric	Latitude	Longitude
Average	8 535	18 208
Minimum	8 199	17 640
Maximum	255 498	344 191

Further examination of the standard deviation statistics shows that most predictions share a standard deviation within the same numeric range, while a small number of predictions showcase a significantly higher standard deviation. Plotting the individual standard deviations by percentile range visualizes the phenomenon, shown in Figure 45.

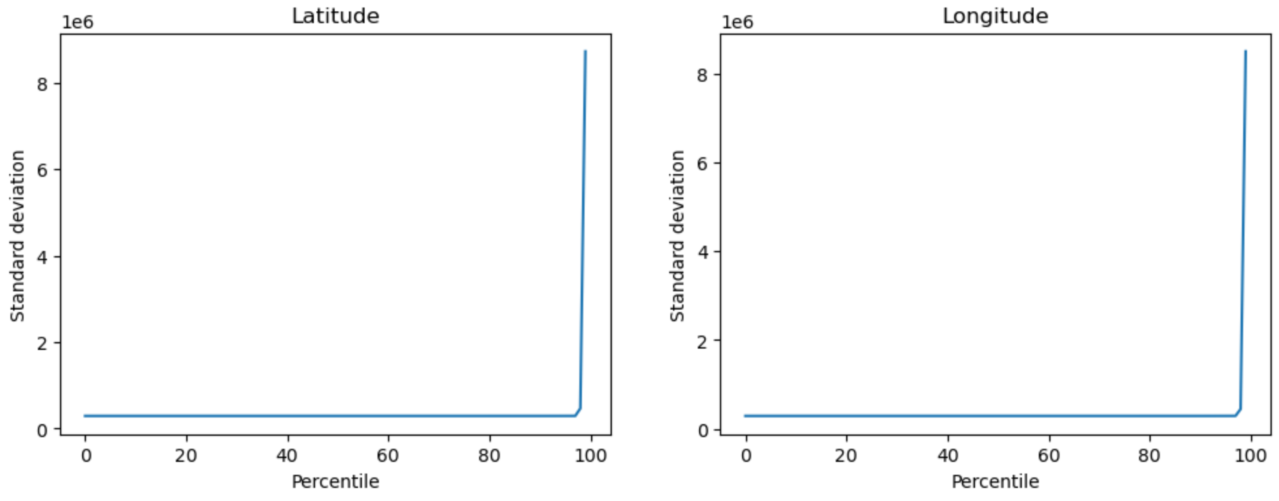


Figure 45. Standard deviation of latitude and longitude predictions by percentile

Predictions were classified into two categories based on the standard deviation values: potentially reliable predictions for those where both the latitude and longitude standard deviation values were less than the value at a given percentile, and unreliable predictions for those with standard deviation values over the value at a given percentile. New average, minimum and maximum distances between predictions and actual locations were calculated for both groups. Table 12 summarizes the results at different percentile calculations for potentially reliable predictions.

Table 12. Summary of distance metrics for reliable predictions

Percentile	Count	Avg. distance (m)	Min. distance (m)	Max. distance (m)
95	176 169	715	0,6	317 426
90	162 216	605	0,6	317 426
85	149 196	535	0,6	317 426
80	136 629	490	0,6	317 426

Similarly, Table 13 summarizes the actual distance differences for unreliable predictions.

Table 13. Summary of distance metrics for unreliable predictions

Percentile	Count	Avg. distance (m)	Min. distance (m)	Max. distance (m)
95	15 309	4 365	31,2	293 455
90	29 262	3 236	15,8	293 455
85	42 282	2 672	11,7	293 455
80	54 849	2 296	2,4	293 455

Using the forestci module to assess prediction confidence and divide predictions into reliable and unreliable offers significant improvement from the original accuracy of the model, as the mean absolute error without any prediction confidence assessment was 1007 meters with a model consisting of 300 estimators and a maximum tree depth of 54 levels. Setting a suitable percentile to serve as a categorization threshold is a tradeoff between the ability to predict coordinates and the accuracy of predictions. Figure 46 visualizes the data points considered unreliable when using the value at 95th percentile, showing how most of the filtered data points were unreliable, with a smaller number of accurate predictions categorized unreliable as false positives.

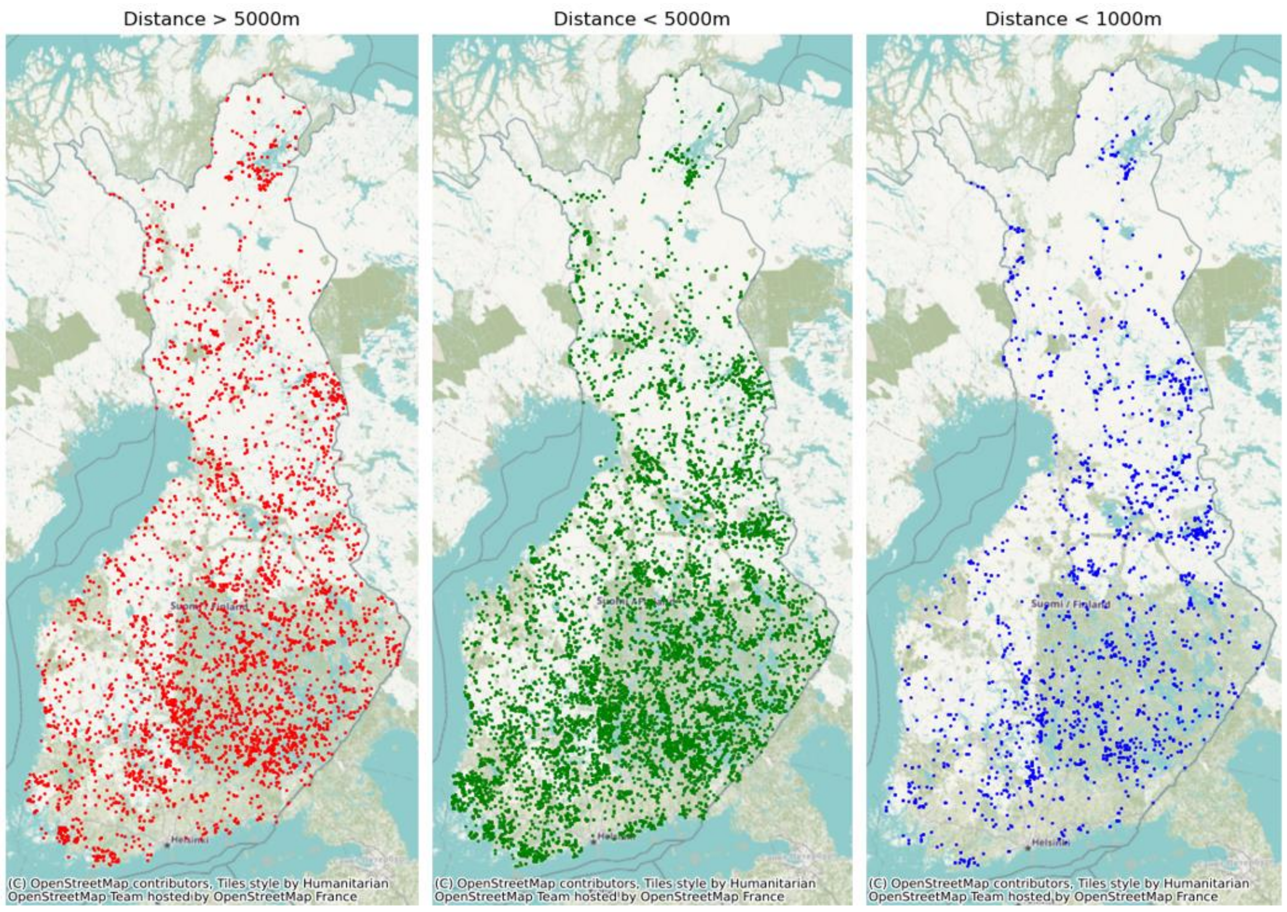

Distance > 5000m
Distance < 5000m
Distance < 1000m
Suomi / Finland
Helsinki
(C) OpenStreetMap contributors, Tiles style by Humanitarian OpenStreetMap Team hosted by OpenStreetMap France

Figure 46. Distance errors for predictions categorized as unreliable

Figure 47 shows the same visualizations, but for the predictions categorized as reliable. Notably, the number of predictions which are off by over five kilometers are significantly smaller in number than the case of unreliable predictions, while predictions where the error is under a kilometer are much more common.

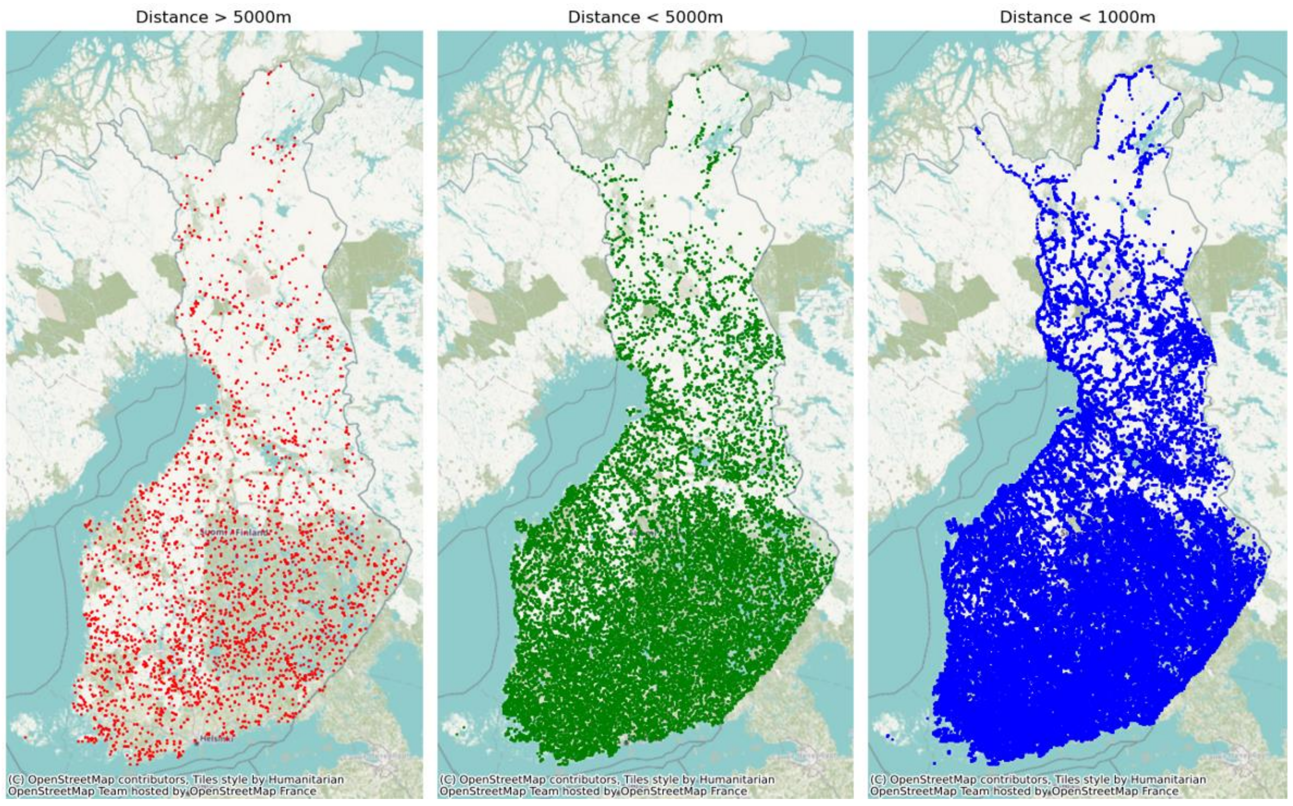

Distance > 5000m
Distance < 5000m
Distance < 1000m
(C) OpenStreetMap contributors, Tiles style by Humanitarian OpenStreetMap Team hosted by OpenStreetMap France

Figure 47. Distance errors and locations for predictions categorized as reliable

A more detailed visualization of the predictions with high error distances is presented in Figure 48, showing that vast majority of the predictions are generally around the correct area of the map, but the distance between the actual and the predicted point is large enough to be considered inaccurate. A handful of egregiously wrong predictions are also visible on the map, with lines stretching across the Finnish map, indicating that the model did not only match the address to the correct city location, but could not even match it to the correct province.

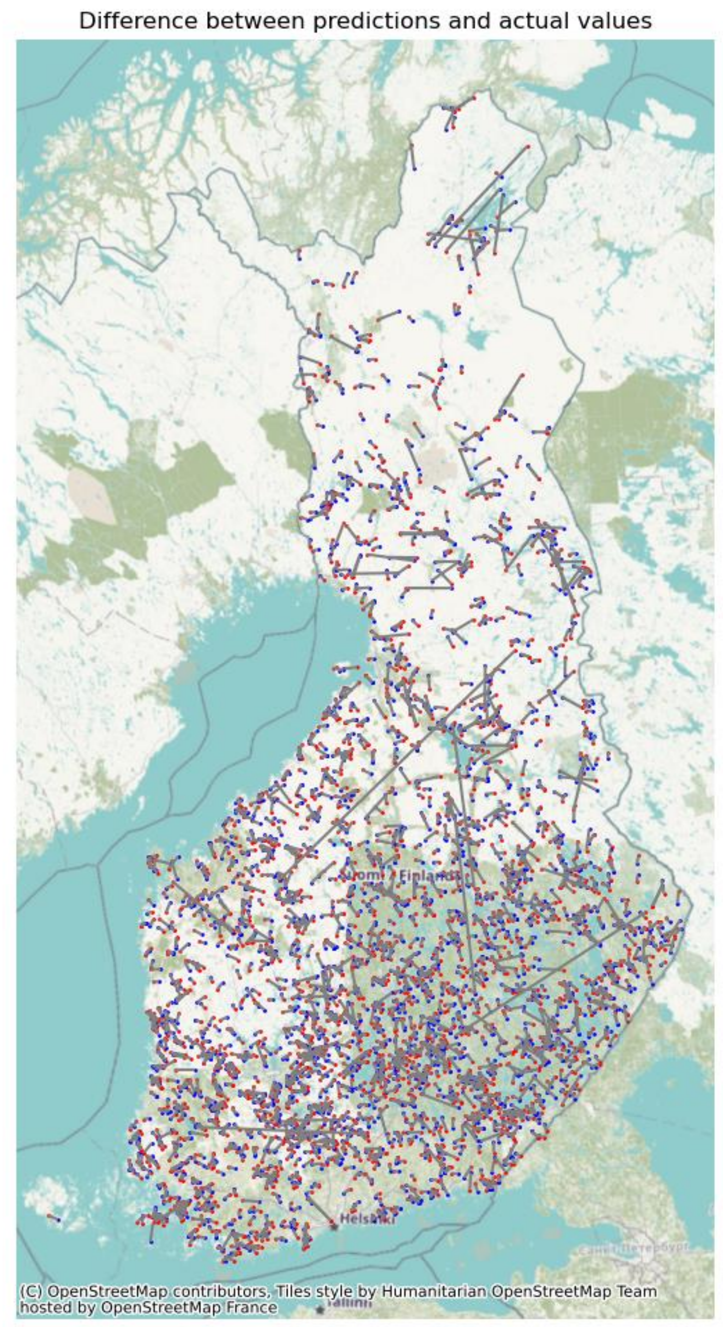

Difference between predictions and actual values
Suomi Finland
Helsinki
(C) OpenStreetMap contributors, Tiles style by Humanitarian OpenStreetMap Team hosted by OpenStreetMap France

Figure 48. Relationships between predicted and actual address locations

Even though the confidence value examination can be used to rule out the predictions that are most likely to be inaccurate, further tuning or separate sanity checks to the model would be needed to ensure that the results are truly accurate.

The final optimized model was a Random Forest Regression model with 300 estimators and a maximum tree depth of 54 levels, complemented by categorization of predictions into reliable and unreliable predictions based on whether the variance of the decision tree predictions was lower than the top 5% variance values within the training dataset. Bootstrapping was enabled in the final model due to limitations with the forestci module, but the optimal implementation would likely have not utilized bootstrapping in random forest model training and implemented a manual in-bag function for variance calculations instead.

5.4.2 XGBoost with Random Forest Regression

XGBoost with Random Forest Regression was examined as an alternative solution. The algorithm was trained with different numbers of estimators, of which the best candidates were picked for a grid-search on tuning node-level subsampling rates and learning rates. Table 14 lists the results for a simple XGBoost Random Forest Regressor model when trained with different numbers of estimators, and otherwise default parameters.

Table 14. Error metrics for XGBoost Random Forest Regressor with default parameters.

Estimators	Mean Squared Error	Mean Absolute Error	Max distance (m)	Min distance (m)
5	1 512 725 686	46 106	470 179	278
10	1 477 662 803	45 660	466 697	61
15	1 461 186 308	45 339	468 464	88
25	1 417 736 207	44 578	470 706	98
50	1 412 284 059	44 437	463 688	140
150	1 408 564 480	44 454	465 582	92
250	1 399 234 277	44 275	464 755	107
300	1 403 498 242	44 391	464 369	26
350	1 403 427 709	44 428	464 498	49
450	1 408 013 447	44 513	464 554	111
550	1 406 350 385	44 431	464 084	86
650	1 409 405 574	44 518	463 780	103
750	1 415 149 898	44 624	463 291	153

The error metrics for the default XGBoost Random Forest Regression indicate a significantly worse performance than was the case with the original Random Forest Regression model. A mean absolute error of 44-46 kilometers places the locations far away from the actual coordinates, and the mean squared errors are at a scale of close to 1000 times higher than with the original Random Forest Regression model. Nevertheless, Figure 49 visualizes the mean squared error for different estimator numbers, used for choosing the estimator counts to be subjected to further hyperparameter tuning.

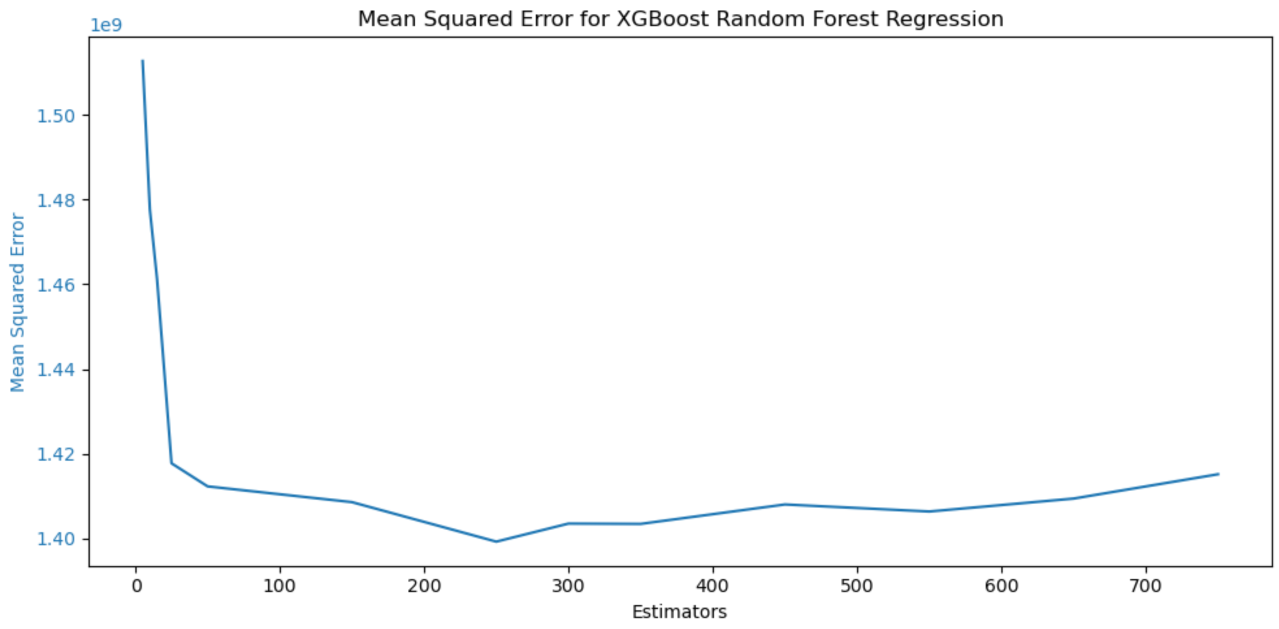


Figure 49. Mean Squared Error for XGBoost Random Forest Regression at different number of estimators

The lowest mean squared error occurred at 250 estimators, which was chosen as the middle point for the following grid search on nodel-level sampling rates in model training process. Different levels of sampling were tested on models with 50, 150, 250, 300 and 350 estimators, the results of which are outlined in Table 15.

Table 15. Mean squared error with different numbers of estimators and node sampling rates

	Estimators				
Sample rate	**50**	**150**	**250**	**300**	**350**
0,1	13 778 559 945	13 909 000 279	14 204 460 499	14 198 679 712	14 315 353 446
0,2	8 676 013 109	9 028 781 835	9 228 812 174	9 343 885 717	9 308 746 293
0,3	6 393 828 075	6 073 290 537	6 107 259 587	6 159 487 704	6 101 746 157
0,4	3 952 152 786	4 203 920 897	4 089 478 643	4 038 382 120	4 050 773 634
0,5	2 780 747 343	2 809 076 721	2 903 280 308	2 959 707 704	2 899 538 264
0,6	2 157 126 978	2 218 966 403	2 186 051 016	2 194 934 672	2 176 714 474
0,7	1 783 145 786	1 694 114 527	1 707 621 394	1 717 166 834	1 713 988 697
0,8	1 412 284 059	1 408 564 479	1 399 234 276	1 403 498 242	1 403 427 708
0,9	1 231 478 325	1 218 242 247	1 214 907 158	1 212 383 627	1 218 256 413
1,0	1 166 991 385	1 168 571 837	1 164 138 860	1 163 861 296	1 163 570 095

Higher sampling rates resulted in smaller error metrics across the board. The lowest mean squared error metric was observed with a 300-estimator model using a node-level sampling rate of 1,0, providing more accurate results than models trained with default values. Furthermore, a grid search was performed for different level sampling rates with a fixed node sampling rate of 1,0. The best metrics were observed when the sampling rates were left at the default values of 1,0, with resulting mean squared errors being the same as in previous search when node level sample rate was set to 1,0.

Another major parameter that was explored was tree depth. The default parameter for tree depth was 6, which is quite low compared to the tree depths explored in the original Random Forest Regression model. A grid search for estimators with different tree depths while keeping node level sample rate at a static 1,0 resulted in the metrics listed in Table 16.

Table 16. Mean squared error at different max depth values for 350 and 450 estimator XGBoost Random Forest Regression

	Estimators	
Max depth	**350**	**450**
10	370 929 262	371 017 449
15	229 115 178	229 098 005
20	158 726 435	158 783 329
25	124 853 213	124 809 170
30	108 002 359	107 967 263
35	101 084 651	101 068 275
40	98 309 897	98 268 136
45	97 356 829	97 281 187
50	97 243 013	97 142 905

The results indicate that sufficiently increasing the maximum depth value also significantly improves the model performance, with improvements becoming ever more marginal at maximum depths of 40-50 levels. Despite the improvements in metrics, the predictions by the XGBoost model still on average geolocate to approximately 44 kilometers away from the actual location, much too far to be reliably used as a part of any business process.

The most potential model candidate for XGBoost model featured 450 estimators with no subsampling to features, and with maximum depth limited to 50 levels. The mean squared error for XGBoost Random Forest Regression model was 97142905, a vast improvement over the initial runs with default parameters, but significantly worse than the mean squared error of the final Random Forest Regression model, 4322721. Slight improvements would likely have been made by further running the model with more estimators and deeper trees, but as the difference between the Random Forest Regression and XGBoost models was multiple magnitudes, further tweaks would likely not have been enough for the XGBoost model to take over as the more accurate model.

5.4.3 MultiOutputRegressor with Support Vector Regression

The third attempt at creating a machine learning model was conducted with the same dataset with the same test-train-validation splits using scikit-learn's MultiOutputRegressor as a wrapper to enable multi-target predictions with Support Vector Regression. The complexity and size of the dataset likely played a significant role in the model runtime, and the first training attempts did not complete even after over 20 hours of training runtime. Because of the long training times and relatively accurate initial results from the Random Forest Regressor model, multi-target Support Vector Regression was abandoned, and focus was instead turned back to tuning the two Random Forest based models.

6 Results

6.1 Predictive capabilities

Two models were trained and optimized: Random Forest Regressor, and XGBoost with Random Forest Regression. Of these models, Random Forest Regressor performed the best, and was chosen as the final model candidate. The optimized random forest model was trained using the training data and verified against the validation dataset, not previously used in model training and tuning. The validation dataset produced a mean squared error of 3 883 613, somewhat smaller than the training dataset used in model optimization.

The mean absolute error between the predicted location and the actual location was 1005 meters, with the closest predictions landing within a meter of the actual location, and furthest prediction placed at almost 370 kilometers away. Factoring in the confidence categorization by discarding the predictions with the highest variance, the predictions that are considered reliable are on average 712 meters away from the actual location of the address. The smallest distance to the actual location was 0,36 meters, while the least accurate prediction was placed 151 kilometers away from the actual location. Figure 50 shows the cumulative distributions of distances, up to 2000 meters, for both the unfiltered predictions, and the predictions deemed reliable.

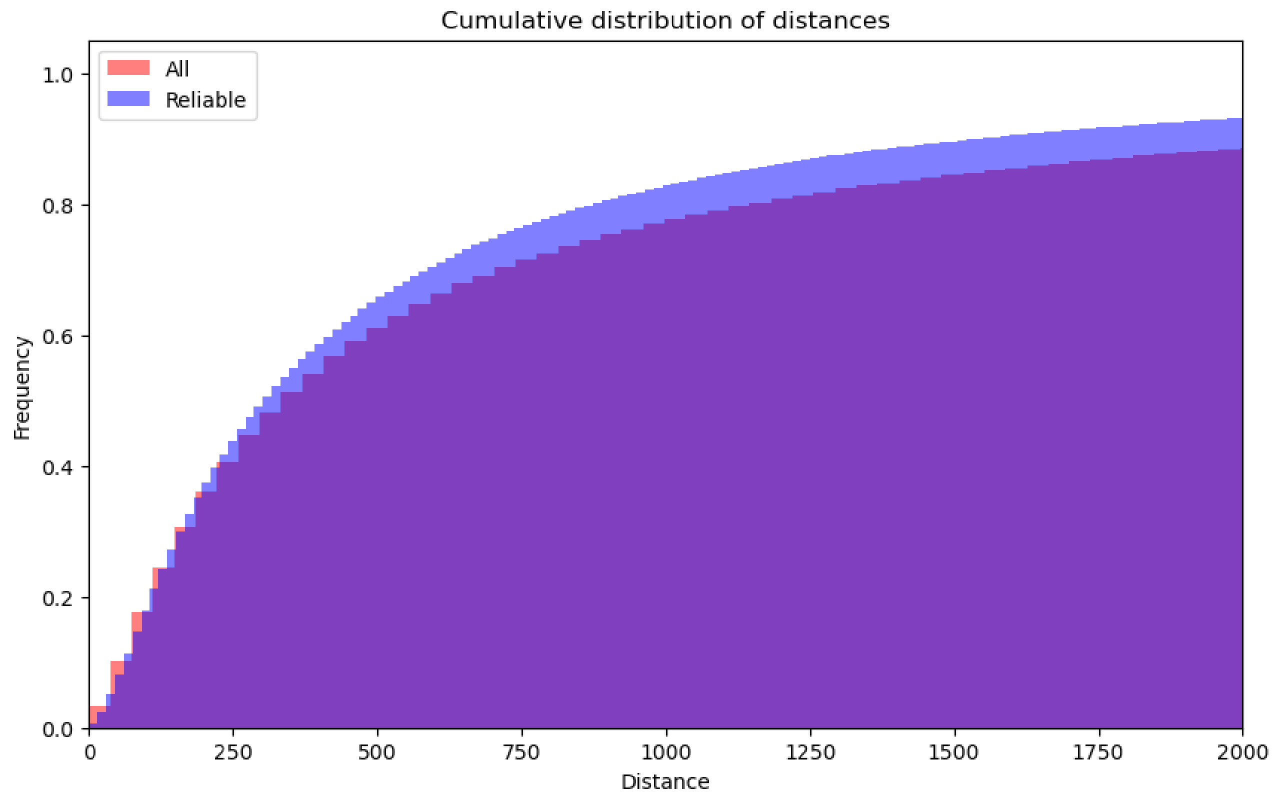


Figure 50. Cumulative distribution of unfiltered and reliable predictions

The categorization of predictions into reliable and unreliable predictions increased the accuracy of the model, with about 80% of the predictions landing within 1000 meters of the actual location when prediction confidence was assessed.

6.2 Usability as a part of a process

The accuracy of the model's predictions limits the usability of the model as a part of a decision-making process. For processes where highly accurate predictions are required, such as predictions of distances from closest roads to buildings, the model cannot provide the required accuracy. For use cases with high accuracy requirements, other types of solutions need to be considered.

The model may be accurate enough for purposes such as placing an address within the service area of a network component, where the tolerance for accuracy is calculated in hundreds of meters or even kilometers. The model would likely be able to provide a crude first step in the process, reducing the manual workload normally expended by human specialists. The results would still need to be verified by another layer of sanity checks or by human verification, but the model could reduce the amount of repetitive, routine workload currently used in some processes.

The model is definitely accurate enough to be used in a fully automated environment in processes which have large tolerances for distance differences and few business-critical connections. The model could be used to automatically route data based on municipality-level or province-level divisions, for instance routing sales queries to the respective sales representative areas. Even without additional verification layers, the model could automate routine tasks.

Overall, the model is usable for general coordinate predictions, but the stakeholders must be made aware of the limitations of the model's predictions. While the model may assist in automating some processes, it is not a catch-all solution for business- or production-critical automations.

6.3 Reflection to original research questions

The research questions were as follows:

1. What steps and methods are required to create a machine learning model capable of predicting latitude and longitude coordinates from a given Finnish street address?
2. How accurate are the predictions of the model?
3. How well can the model evaluate the accuracy of its prediction?

The outcome of this research suggests that Random Forest Regression can be used to create a machine learning model capable of predicting coordinates of a Finnish street address, while XGBoost Random Forest Regression falls short at the task, producing significantly less accurate predictions. The Random Forest Regression model produced reasonably accurate predictions even at lower estimator numbers, allowing for faster iteration rounds when optimizing the model with estimator numbers and bootstrapping options. Increasing the number of estimators provided additional accuracy to the model, at a cost of computational complexity and model size. Sufficient preparation of training data is important for increasing model accuracy, including assessment of spatial distribution, spatial sampling, and text vectorization.

The resulting model without any tuning is capable of landing the predicted location within 400 meters approximately 50% of the time, and within 750 meters around 75% of the time. The average distance from the actual location was approximately 1005 meters with a Random Forest Regression model composed of 300 estimators with maximum depth of 54 levels. Utilizing forestci module for variance calculations and categorizing the predictions based on variance value further improved the predictions of the model, reducing the average prediction distance error to 400-700 meters, depending on the threshold value used for prediction reliability categorization.

The model was not always aware of when its predictions were reliable. Most of the time the variance gave a decently accurate indication of the reliability, but both reliable and unreliable categories ended up with data points from the opposing category. Variance-based filtering improved the results and should be utilized to improve the model accuracy, but do not guarantee a foolproof method to extract universally reliable predictions from the model. Additional sanity checks on the model’s predictions is advised.

7 Conclusions

The creation of a model capable of predicting coordinates based on address data was successful, though the model is not a miracle solution for all processes. The model is usable in processes where highly accurate predictions are not required and general locations are sufficient.

Random Forest Regression is a useful algorithm for handling large datasets and can produce meaningful results even with highly complex patterns and spatially unevenly distributed data. Sufficient preparation and data preprocessing can improve the predictive capabilities of the algorithm, and special attention needs to be paid to the type of data and the patterns it contains. Random Forest Regression algorithm implementations contain sufficient tools, though some manual work may be needed to fully optimize the model.

Deeper understanding in linguistics and NLP would have helped in further optimizing the model, especially with handling the mixed Finnish and Swedish addresses. More advanced knowledge in lemmatization and vectorization would also likely have helped in decision-making during data pre-processing steps.

Further implementations of the model should also consider another layer of checks to the model predictions in addition to the confidence estimations. Potential sanity check types could include checking the location against raster maps of the area to verify whether the address is in a habitable place (and not in the middle of a lake, for example). Another check would be to compare the predicted address against the border of its corresponding postal code region, ruling out the most egregious false predictions. A custom in-bag function could also be implemented so that the model can take advantage of not bootstrapping the data, as not using bootstrapping initially produced better mean absolute error metrics than the original form of the algorithm. Finally, spatial sampling of data could be altered so that actual address is sampled at an inverse rate in relation to the number of generated addresses within the same area, reducing the spatial clustering of sampled data.

References

Addresses, postal codes and WGS84-coordinates of Finnish buildings. (n.d.) Dataset description at AvoinData website. https://www.avoindata.fi/data/en_GB/dataset/postcodes

Alencar, R. (2017, November 15). *Resampling strategies for imbalanced datasets.* Article on Kaggle website. https://www.kaggle.com/code/rafjaa/resampling-strategies-for-imbalanced-datasets

Bogacz, L. (2023, March 22). *Text Vectorization, an Introduction.* Article on The Information Lab website. https://theinformationlab.nl/2023/03/22/an-introduction-to-embeddings/

Bootstrap Aggregation, Random Forests and Boosted Trees. (n.d.) Article on QuantStart website. https://www.quantstart.com/articles/bootstrap-aggregation-random-forests-and-boosted-trees/

Caballero, Y., Giraldo, R., Mateu, J. (2022). A spatial randomness test based on the box-counting dimension. *Advances in Statistical Analysis, 106*, 499–524. https://doi.org/10.1007/s10182-021-00434-4

Carniel, A. C. (2023). Defining and designing spatial queries: The role of spatial relationships. Geo-Spatial Information Science, 1–25. https://doi.org/10.1080/10095020.2022.2163924

Chawla, S., Shekhar, S., Wu, W., & Ozesmi, U. (2001) Modeling Spatial Dependencies for Mining Geospatial Data. Proceedings of the 2001 SIAM International Conference on Data Mining, 1–17. https://doi.org/10.1137/1.9781611972719.27

Chaya. (2020, June 9). *Random Forest Regression*. Article on Medium. https://levelup.gitconnected.com/random-forest-regression-209c0f354c84

Decision Trees. (n.d.) Documentation on scikit-learn website. https://scikit-learn.org/stable/modules/tree.html

Denis, J. (n.d.). *The Robinson Projection*. https://geography.wisc.edu/maplibrary/the-robinson-projection/

Developing Multi-Target Regression Models with Python. (2023, December 30). Article on Medium website. https://medium.com/@tubelwj/developing-multi-class-regression-models-with-python-c8beca5dd482

Digiroad. (2018). *Digiroad: Tietolajien kuvaus* [Digiroad: data type specification]. https://vayla.fi/documents/25230764/0/Tietolajien+kuvaus_2_2018.pdf/3d0c7100-6c63-419e-9aad-e88f108545f6

Digital and Population Data Services Agency. (n.d.-a). *Maintaining the Population Information System.* https://dvv.fi/en/maintaining-the-population-information-system

Digital and Population Data Services Agency. (n.d.-b). *Real estate, building and spatial information*. https://dvv.fi/en/real-estate-building-and-spatial-information

Eekels, J., & Roozenburg, N. F. M. (1991). A methodological comparison of the structures of scientific research and engineering design: their similarities and differences. Design Studies, 12(4), 197–203. https://doi.org/10.1016/0142-694X(91)90031-Q

Eppright, C. (2021, March 25). *What Is Natural Language Processing (NLP)?* Article on Oracle website. https://www.oracle.com/artificial-intelligence/what-is-natural-language-processing/

Exploratory Data Analysis. (n.d.). United States Environmental Protection Agency. https://www.epa.gov/caddis/exploratory-data-analysis

Finnish Geodetic Institute, National Land Survey of Finland. (n.d.). *The Finnish coordinate reference systems.* https://www.maanmittauslaitos.fi/sites/maanmittauslaitos.fi/files/old/Finnish_Coordinate_Systems.pdf

Finnish Transport Infrastructure Agency. (2024, January 17). *About Digiroad*. Website on Finnish Transport Infrastructure Agency website. https://vayla.fi/en/about-digiroad

GISGeography. (2024, March 10). *How map projections work.* https://gisgeography.com/map-projections/

Hevner, March, Park, & Ram. (2004). Design Science in Information Systems Research. MIS Quarterly, 28(1), 75. https://doi.org/10.2307/25148625

History of the EPSG Dataset. (n.d.). EPSG website. https://epsg.org/history.html

Hoaglin, D. C. (2003). John W. Tukey and Data Analysis. *Statistical Science*, 18(3), 311–318. https://doi.org/10.1214/ss/1076102418

Indexing and Querying Spatial Data. (n.d.) Article on Oracle Database Developer's Guide. https://docs.oracle.com/en/database/oracle/oracle-database/19/spatl/indexing-querying-spatial-data.html

Intergovernmental Committee on Surveying and Mapping. (n.d.). *Projections*. https://www.icsm.gov.au/education/fundamentals-mapping/projections

Kanade, V. (2022, April 4). *What is Machine Learning? Definition, Types, Applications, and Trends.* Article on SpiceWorks website. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/

Kanevski, M., Timonin, V., Pozdnukhov, A. (2009). *Machine Learning for Spatial Environmental Data: Theory, Applications, and Software*. EPFL Press. https://doi.org/10.1201/9781439808085

Kopczewska, K. (2022). Spatial machine learning: new opportunities for regional science. *The Annals of Regional Science,* 68(3), 713–755. https://doi.org/10.1007/s00168-021-01101-x

Lehtiranta, K. (2016). *Osoitemuutokset kuntaliitoksissa* [Address changes in municipality mergers]. [Master's thesis, Lapin AMK]. Theseus. https://urn.fi/URN:NBN:fi:amk-2016053110913

Lev, A. (2022, December 19.) *XGBoost versus Random Forest*. Article on Qwak website. https://www.qwak.com/post/xgboost-versus-random-forest

Li, J., Heap, A. D., Potter, A., & Daniell, J. J. (2011). Application of machine learning methods to spatial interpolation of environmental variables. *Environmental Modelling & Software*, 26(12), 1647–1659. https://doi.org/10.1016/j.envsoft.2011.07.004

Multioutput Regression in Machine Learning. (2023, December 07). Article on Geeks for Geeks website. https://www.geeksforgeeks.org/multioutput-regression-in-machine-learning/

MultiOutputRegressor. (n.d.) Documentation on scikit-learn website. https://scikit-learn.org/stable/modules/generated/sklearn.multioutput.MultiOutputRegressor.html

Mwiti, Derrick. (2023, September 1). *Random Forest Regression: When Does It Fail and Why?* Blog post on NeptuneAI website. https://neptune.ai/blog/random-forest-regression-when-does-it-fail-and-why

Nikparvar, B. & Thill, J.-C. (2021). Machine Learning of Spatial Data. *ISPRS International Journal of Geo-Information*, 10(9), 600. https://doi.org/10.3390/ijgi10090600

Nunamaker, J. F., Chen, M., & Purdin, T. D. M. (1990). Systems Development in Information Systems Research. *Journal of Management Information Systems*, 7(3), 89–106. https://doi.org/10.1080/07421222.1990.11517898

Open Geospatial Consortium Inc. (2011). *OpenGIS Implementation Standard for Geographic information – Simple feature access.* (ISO 19125-1:2004). http://www.opengis.net/doc/is/sfa/1.2.1

Ord, J. K. (1978). How many trees in a forest. *Mathematical Scientist,* 3, 23–33.

Ord, J. K. (2010). Spatial autocorrelation: a statistician's reflections. In L. Anselin & S. J. Rey (Eds.), *Perspectives on Spatial Data Analysis* (pp. 165–180). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-01976-0_12

Peffers, K., Tuunanen, T., Gengler, C. E., Rossi, M., Hui, W., Virtanen, V., & Bragge, J. (2020). Design Science Research Process: A Model for Producing and Presenting Information Systems Research. https://doi.org/10.48550/ARXIV.2006.02763

Posti. (n.d.). *Postinumerot ja postinumeroalueet Suomessa* [Postal codes and postal areas in Finland]. Finnish postal office website. https://www.posti.fi/fi/postinumerohaku/postinumeroalueet

Raghav, P. (2019, November 11). *Understanding NLP Word Embeddings – Text Vectorization*. Article on Medium website. https://towardsdatascience.com/understanding-nlp-word-embeddings-text-vectorization-1a23744f7223

Rey, S. J., Arribas-Bel, D., Levi, J. W. (2020). *Geographic Data Science with Python.* CRC Press. https://doi.org/10.1201/9780429292507

Salian, I. (2018, August 2). *Difference Between Supervised, Unsupervised, & Reinforcement Learning.* NVIDIA Blog. https://blogs.nvidia.com/blog/supervised-unsupervised-learning/

Spatial Data Definition: GIS Dictionary. (n.d.) ESRI website. https://support.esri.com/en-us/gis-dictionary/spatial-data

Spatial Data Types and Metadata. (n.d.) Oracle Database Developer's Guide. https://docs.oracle.com/en/database/oracle/oracle-database/19/spatl/spatial-datatypes-metadata.html

Spatial Operators. (n.d.) Oracle Database Developer's Guide. https://docs.oracle.com/en/database/oracle/oracle-database/19/spatl/spatial-operators-reference.html

Spatial Relations Defined. (n.d.) Article on SAFE website. https://docs.safe.com/fme/html/FME-Form-Documentation/FME-Transformers/Transformers/spatialrelations.htm

Statistics Finland (n.d.-a). *Kartta-aineistot* [Maps]. Statistics Finland website. https://stat.fi/tup/karttaaineistot/index.html

Statistics Finland. (n.d.-b). *Luokitustiedotteet* [Classification news]. Statistics Finland website. Retrieved 23 July, 2024, from https://stat.fi/fi/luokitukset/luokitustiedotteet/

Statistics Finland. (2021). *Tilastolliset postinumeroalueet* [Statistical postal code areas]. Dataset description on Statistics Finland website. https://stat.fi/tup/karttaaineistot/postinumer-oalueet.html

Sypytkowski, B. (2022, April 19). *R-Tree: algorithm for efficient indexing of spatial data*. Blog post. https://www.bartoszsypytkowski.com/r-tree/

Taylor, P. (2023, November 16). *Data growth worldwide 2010-2025*. Statista. https://www.statista.com/statistics/871513/worldwide-data-created/

The Effects of the Depth and Number of Trees in a Random Forest. (2024, April 03.) Article on GeeksForGeeks website. https://www.geeksforgeeks.org/the-effects-of-the-depth-and-number-of-trees-in-a-random-forest/

TurkuNLP. (n.d.) Website on TurkuNLP pages. https://turkunlp.org/finnish_nlp.html

Vector vs Raster in GIS: What's the Difference? (n.d.) GIS Geography website. https://gisgeography.com/spatial-data-types-vector-raster/

Vigni, M. L., Durante, C., & Cocchi, M. (2013). Exploratory Data Analysis. In *Data Handling in Science and Technology* (Vol. 28, pp. 55–126). Elsevier. https://doi.org/10.1016/B978-0-444-59528-7.00003-X

Walls, J. G., Widmeyer, G. R., & El Sawy, O. A. (1992). Building an Information System Design Theory for Vigilant EIS. Information Systems Research, 3(1), 36–59. https://doi.org/10.1287/isre.3.1.36

What is Artificial Intelligence (AI)?: Google Cloud. (n.d.) Article on Google Cloud website. https://cloud.google.com/learn/what-is-artificial-intelligence

What is Artificial Intelligence (AI)?: IBM. (n.d.) Article on IBM website. https://www.ibm.com/topics/artificial-intelligence

What is Exploratory Data Analysis?: IBM. (n.d.) Article on IBM website. https://www.ibm.com/topics/exploratory-data-analysis

What is GDPR, the EU's new data protection law? (n.d.) Article on GDPR website. https://gdpr.eu/what-is-gdpr/

What is GIS?: Geographic Information System Mapping Technology. (n.d.) ESRI. https://www.esri.com/en-us/what-is-gis/overview

What is Machine Learning (ML)? (2020, June 26). University of Berkeley website. https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/

What is NLP?: Natural Language Processing Explained. (n.d.) Article on Amazon website. https://aws.amazon.com/what-is/nlp/

Word Embedding Demo. (n.d.) Demo application on University of Turku website. http://epsilon-it.utu.fi/wv_demo/

Wujek, B., Hall, P., Günes, F. (2016). Best practices for machine learning applications. *SAS Institute Inc,* 3.

Zhu, J., Yang, M., Ren, Z. (2023). Machine Learning in Environmental Research: Common Pitfalls and Best Practices. *Environmental Science & Technology, 57*(46), 17671–17689. https://doi.org/10.1021/acs.est.3c00026