logistics

Article

Travel Time Prediction in a Multimodal Freight Transport Relation Using Machine Learning Algorithms

Nikolaos Servos [1,*], Xiaodi Liu [1], Michael Teucke [2] and Michael Freitag [2,3]

1 Bosch Connected Industry, Robert Bosch Manufacturing Solutions GmbH, Leitzstrasse 47, 70469 Stuttgart, Germany; taylorkkxiaodiliu@gmail.com
2 BIBA—Bremer Institut für Produktion und Logistik GmbH, University of Bremen, Hochschulring 20, 28359 Bremen, Germany; tck@biba.uni-bremen.de (M.T.); fre@biba.uni-bremen.de (M.F.)
3 Faculty of Production Engineering, University of Bremen, Badgasteiner Straße 1, 28359 Bremen, Germany
* Correspondence: nikolaos.servos@de.bosch.com

Received: 18 November 2019; Accepted: 23 December 2019; Published: 25 December 2019

Abstract: Accurate travel time prediction is of high value for freight transports, as it allows supply chain participants to increase their logistics quality and efficiency. It requires both sufficient input data, which can be generated, e.g., by mobile sensors, and adequate prediction methods. Machine Learning (ML) algorithms are well suited to solve non-linear and complex relationships in the collected tracking data. Despite that, only a minority of recent publications use ML for travel time prediction in multimodal transports. We apply the ML algorithms extremely randomized trees (ExtraTrees), adaptive boosting (AdaBoost), and support vector regression (SVR) to this problem because of their ability to deal with low data volumes and their low processing times. Using different combinations of features derived from the data, we have built several models for travel time prediction. Tracking data from a real-world multimodal container transport relation from Germany to the USA are used for evaluation of the established models. We show that SVR provides the best prediction accuracy, with a mean absolute error of 17 h for a transport time of up to 30 days. We also show that our model performs better than average-based approaches.

Keywords: logistics; supply chain management; multimodal freight transports; travel time prediction; machine learning

1. Introduction

An accurate travel time prediction is of high value for freight transports, as it allows supply chain participants to increase their logistics quality [1–3]. A material planner at the receiving plant can identify delayed deliveries in advance, and forecasts of material stocks can be optimized through this. Furthermore, a plant can adjust its capacities in time, e.g., staff or machinery, and increase its efficiency. Logistic service providers benefit in the same way. At warehouses, ports, or other hubs, capacities concerning staff, ramps, forklifts, etc. can be scheduled accordingly. Consequently, manufacturers and logistic service providers can enhance their efficiency, optimize their processes, and increase planning accuracy [1,3,4].

Generally speaking, a supply chain involves all collaborative operations by all involved companies to transform raw materials into the final product. This includes activities such as sourcing raw materials, manufacturing, assembly, and distribution to the end customer. To do so, logistics for the handling, transport, and storage of materials is required. While logistics deals with the operational level, supply chain management deals with the planning and management of supply chains. Logistics also includes the task to provide information between the various stages in the supply chain. Usually, a number of companies or organizations are responsible for the transport. Especially in multimodal transports with various transshipment points, these conditions make the supply chain significantly more complex. [5] This work focuses on estimating travel time accurately in multimodal transports. Accurate travel time estimates are important for efficient management of transport operations and logistics in supply chains.

For travel time estimation, a continuous monitoring of freight transports is required, e.g., by using mobile sensors attached to transported goods [6]. At the same time, achieving the required transparency is one of the most challenging tasks in logistics [7]. Travel time prediction is also difficult, as many factors, such as weather, traffic, vehicle, route, or transport relation influence it [8].

Currently, the majority of the published literature deals with passenger transportation, such as bus arrival times or highway travel times, rather than freight transports. Hereby, only rides with a duration of up to two hours are considered, which is not applicable to freight transports [9,10]. For travel time estimation, the authors use average-based approaches [11,12], Kalman filters [13–17], or machine learning (ML) algorithms (see Section 2). Furthermore, in their research, the route and stops are known in advance.

As per [9], only ML algorithms can adequately deal with complex and dynamic behavior during transports. ML has the ability to deal better with complex and non-linear relationships between predictors and can process complex and noisy data [10]. The literature also confirms this statement, as ML approaches usually perform better than average-based approaches [18–20]. Despite this situation, ML has only been applied in a minority of recent publications dealing with travel time prediction in freight transports [1,2,21,22].

In the forecasting of multimodal freight transports based on real-time tracking data, only very limited research has been conducted, which mainly uses average-based approaches [23]. Consequently, this paper focuses on the evaluation of ML for long term forecasting to predict travel times of multimodal freight transports. Tracking data is generated by sensors, which are attached to the transported goods. The sensor data is transmitted with a low frequency in longer periods of 30 min, due to a small battery coverage. Except for the origin and destination, no further information concerning the exact route and trans-loading points are available beforehand. Finally, we can show that machine learning algorithms can adequately predict travel time under the given constraints. In our use-case, support vector regression (SVR) provides the best prediction accuracy, with a mean absolute error of 17 h for a transport time of up to 30 days. We also show that our model performs better than average-based approaches.

We organize the rest of the paper as follows: In Section 2, we provide an overview of literature dealing with travel time estimation using machine learning to identify the research gap. Section 3 describes the selected learning algorithms and the framework of this work. Section 4 presents the use-cases and the collected tracking data, which will be processed and transformed in Section 5. Section 6 describes the process of creating the prediction model. The results are evaluated and compared to average-based models in Section 7. In Section 8, we conclude the paper and suggest further work in this field.

2. Literature Review

In this paper, we evaluate the capability of machine learning algorithms to predict the travel time in multimodal transports. Most research in this realm has been conducted on travel time prediction for segments of streets or highways and bus rides. Only a minor part of the research is related to freight transports or similar applications.

The authors of [24] use data from loop detectors and a global positioning system (GPS) to determine speed, occupancy, and volume of cars on highways, and include previous transport times to predict travel times on a four-mile stretch of highway. They use artificial neural networks (ANNs) as a learning algorithm. Similarly, the authors of [25] only include loop detectors and use support vector machines to estimate travel time on highway segments of up to 350 km. Ref. [26] identifies random forests (RF) as the best approach for predicting travel times on urban street segments using GPS data from 300 probe vehicles. In a similar use-case to ref. [24], the authors of [27] evaluate a gradient boosting method and RF for travel time estimation on highway segments and show that the gradient boosting method performs best. The authors of [8] also show that gradient boosting provides a good prediction. The authors also evaluate different features based on GPS data. They establish that including the speed between two transmissions increases prediction accuracy by 5%. The authors of [28] evaluate deep neuronal networks using 12 months of GPS data and the previous 12 travel times as features. Even for a 131 km stretch, their approach results in a mean absolute error (MAE) of 3.25 min. The above approaches do not consider any end-to-end transport relations as with freight transports. In addition, exact advance knowledge of the route and a high frequency of GPS measurements are required, or side-based approaches such as loop detectors are used.

Considering bus travel time prediction, nearly all authors, such as [18,29–33], use ANNs in their research. In all cases, the route is known in advance and is separated based on the bus stops. Most of the approaches only differ when considering the evaluated use-case and the used features. Usually, features such as the time of departure, public holidays, dwelling time at bus stops, current travel time, distance to destination, and average speed are used. Similarly, the authors of [20] also apply ANNs to tramways. The authors of [31] identify ANNs as a better approach than k-nearest neighbors (kNN), while the authors of [34] show that support vector regression (SVR) performs better. The authors of [35] use a multilinear regression for prediction. To consider all previous stops in their model, the authors add the cumulated number of stops and dwelling time. In contrast to the previous approaches, the bus ride is related to an origin and destination considering an end-to-end transport relation including stops, similar to freight logistics. However, the authors have not considered that many stops within supply chains are often unknown to supply chain participants and can differ between transports using the same transport relation. Furthermore, their approaches require a high number of rides and assume a high frequency of GPS measurements.

The authors of [2] apply SVR on milk run freight transports. They comment that SVR can better generalize information than neuronal networks. Hereby, the stops of the milk run are known in advance. Their prediction model achieves an accuracy of three minutes. However, it requires 1800 rides for training. The authors of [36] consider random forests (RF) as a reasonable algorithm to predict aircraft arrival times. Flight related data such as delays, origin, destination, position, weather data, and data from the air traffic control have been taken to consideration. The authors can prove that their estimation is better than the one provided by air traffic control. The authors of [1] apply neuronal networks and SVR to predict the time of arrival of container transports. The timestamp, distance to destination, and geo-coordinates have been chosen as features, as well as weather information. The authors show that SVR performs better than ANNs and that weather data does not have a significant influence on the transport time. Finally, the authors of [21] evaluate kNN, SVR, and RF for arrival time prediction of open-pit trucks. Hereby, a site-based approach is used. The position is only measured at a few discrete nodes of the route network, and no GPS data is used. RF provides the best prediction results.

The literature review shows that none of the identified research has yet been conducted on multimodal transports with the transported goods themselves being equipped with a sensor instead of the transport vehicle. Furthermore, the research has also not yet considered the scenario where transported goods are equipped with sensors with a small battery coverage that are consequently transmitting position data with a low frequency, in this case every 30 min, in comparable distances. The concluding algorithm has to be capable of deriving a good prediction with a lower amount of data. In addition, the case of the route being unknown has not been considered in the research.

3. Methodology

The following chapter provides a description of applied machine learning algorithms and the corresponding parameters of models based on the knowledge from the literature review. A framework of the proposed approach is also presented.

3.1. Choice of Learning Algorithms

This research will select extremely randomized trees (ExtraTrees), adaptive boosting (AdaBoost) and support vector regression (SVR) as representative algorithms for modelling, according to the results of the literature review. All three algorithms are capable of adapting to complex systems and are robust in dealing with complex and small data sets. They have shown superior performance in previous research with low processing time [1,2,25,37].

Support vector machine (SVM) is a classification technique based on the concept of supervised learning. It aims to find the decision boundary between data points and separate them by an optimally derived hyperplane maximizing margin [25] (pp. 277–278). SVM is applied to regression problems as SVR. Briefly speaking, SVR defines a weight for each variable of the model and learns the behavior of the data through a training and testing process. The kernel trick is applied to find a nonlinear decision boundary by creating a much higher dimension of new features transformed from existing features [38] (pp. 282–284). A common and well performing kernel function for travel time prediction is the radial basis function (RBF) [39] (p. 216). The other two hyper-parameters, ε and C, are important for structuring a decision boundary. While the parameter, ε, describes an insensitive loss function that penalizes prediction residuals larger than the value specified by it, C describes the tolerance towards prediction errors as a regularization parameter [40] (pp. 68–73).

SVR models have shown plausible performance in recent research due to their good ability to generalize data and guarantee global minima compared to other methods. The processing time can also be low, given the constraints of a small data size and high problem complexity [25,39]. For these reasons, SVR is chosen as one of the algorithms in our study.

Extremely randomized trees (ExtraTrees) is a tree-based ensemble method for both classification and regression problems. The basic principle behind ExtraTrees is to randomly create a number of different trees (*n_estimators*) with randomly chosen features. This process minimizes the variance of the prediction results [41] (pp. 3–4). In this study, regularization parameter, *random_state*, is also considered, which initiates the random number generator to randomize characteristics in trees [42] (p. 619).

Adaptive boosting (AdaBoost) is one of the most widely used boosting approaches, based on the concept of combining weak and inaccurate learners, mostly decision trees, to a strong and accurate learner. In regression problems, AdaBoost acts as a meta-estimator, which uses a weighted sum of prediction errors for a number of regressors to increase performance. These weights on the dataset are subsequently updated based on the prediction errors of the previous regressor. Important hyper-parameters of this method are the number of estimators (*n_estimators*) after which the algorithm will stop and the *learning_rate* that determines the degree of the weight adjustment [43].

According to [21,44], both ExtraTrees and AdaBoost are used for travel time prediction and have shown reasonable results with low errors. With the support of previous practical success, we also chose these two methods to build our prediction model.

3.2. Framework

In this research, the framework has been derived from the theory of knowledge discovery in databases (KDD) data mining process flow [45] (pp. 72–73) and is shown in Figure 1.

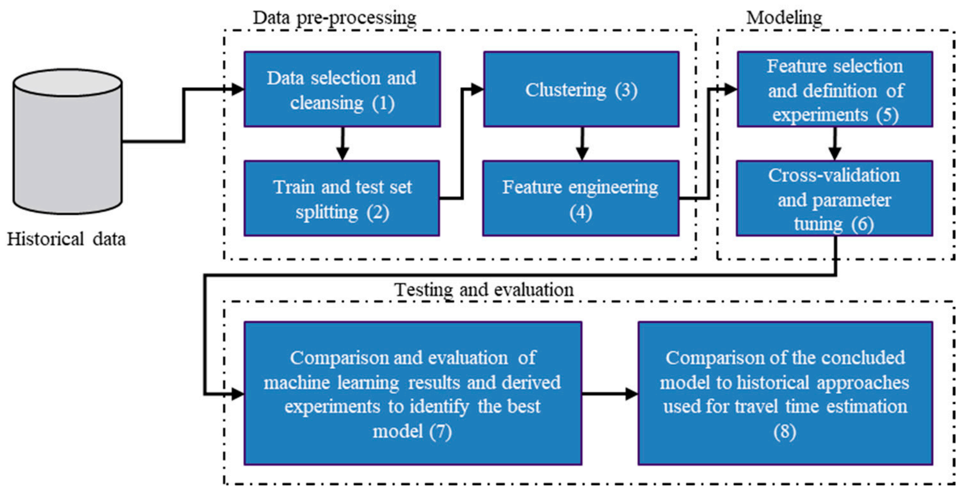
Data pre-processing
Modeling
Historical data
Data selection and cleansing (1)
Train and test set splitting (2)
Clustering (3)
Feature engineering (4)
Feature selection and definition of experiments (5)
Cross-validation and parameter tuning (6)
Testing and evaluation
Comparison and evaluation of machine learning results and derived experiments to identify the best model (7)
Comparison of the concluded model to historical approaches used for travel time estimation (8)

Figure 1. Framework used for travel time prediction.

The proposed approach, which is based on the historical tracking data of transports in the considered use-case described in Section 4.1, involves eight steps: (1) Relevant data for modelling is selected and cleansed to eliminate outliers. (2) Then, a train and test set splitting is performed. The test set represents unknown data that is used to evaluate the performance of the developed models, while the train set represents known data used for training the models. (3) To include information on the route, a clustering approach is used to identify stops and trans-loading points. Only the train set is used to identify the clusters. (4) Finally, features, which are considered as relevant, are derived from the tracking data and the clustering results based on domain knowledge and literature. (5) In the modelling phase, different methods have been used to select features and to derive different experiments as feature subsets. (6) Then, for each experiment and learning algorithm, grid search and cross validation with five train and validation sets have been used for parameter tuning. (7) The models are finally tested and evaluated, considering the experiments and learning algorithms. Based on the results, the best model is selected. (8) Finally, the best machine learning model is compared to commonly used average-based approaches.

The described approach has been applied using Python with Jupyter notebook. The used clustering and prediction algorithms have been implemented using the machine library scikit-learn.

4. Use-Case and Data Generation

In this chapter, our use-case and the system used to track the transported goods is briefly introduced. The generated data is also described.

4.1. Use-Case Description

In this use-case, real-life data derived from a multimodal supply chain extending from Bremen in Germany to Vance in the United States has been used. Trucks, trains, and ships carry out the included transports. The process is shown in Figure 2. First, the required goods are packed into a container at a container freight station of a logistics service provider in Bremen. Hereafter, it will be picked up by truck and transported to the port of Bremerhaven. After storing the container at the port's container yard for several days, the container is transported by vessel to the port of Charleston via the port of Norfolk. In Charleston, the container is loaded onto a train driving to a train yard in Bessemer, with several unknown intermediate stops. There, it is picked up by truck and transported to its final destination in Vance. Hereby, it should be noted that the actual stops during transport are not limited to the stops previously described. More stops can occur during the transport that have not been known by most of the supply chain participants. The total transport takes between 22 and 30 days.

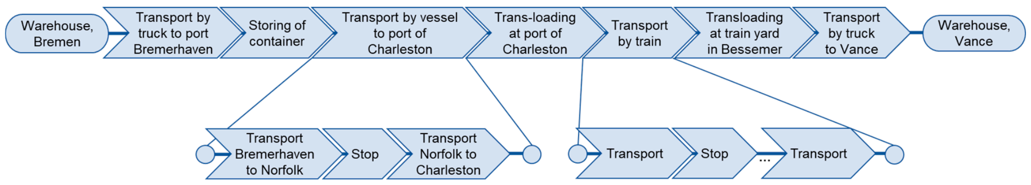
Warehouse, Bremen
Transport by truck to port Bremerhaven
Storing of container
Transport by vessel to port of Charleston
Trans-loading at port of Charleston
Transport by train
Transloading at train yard in Bessemer
Transport by truck to Vance
Warehouse, Vance
Transport Bremerhaven to Norfolk
Stop
Transport Norfolk to Charleston
Transport
Stop
...
Transport

Figure 2. Transport process of the considered use-case.

4.2. Data Generation and Description

Fourty-three pallets have been distributed among seven container shipments. Those pallets have been equipped with a hybrid sensor system, as shown in Figure 3, for a period of seven weeks for the purpose of tracking. Therefore, each pallet within the container has been equipped with a sensor. The sensor is associated with transport related information from a label attached to the palette, e.g., the serial number. This 'pairing process' is performed with a scan. The sensor is measuring quality related data, such as humidity and temperature, and transmits this data together with its ID via Bluetooth low energy (BLE). Then, the sensor data is transmitted to a device called a gateway, which has been attached on the container and is receiving the sensor data at a predefined interval. Due to the long transport and a restricted battery life, sensor measurements are received and transmitted only every 30 min. The gateway then adds geo-positioning data to the received sensor data using GPS and transmits the data to the Bosch IoT Cloud, a central data cloud system, through the global system for mobile communications (GSM) network. If no GSM connection is available, the sensor data is buffered on the gateway together with the GPS data until a connection for transmission is available again. All transmitted data is stored in the cloud for later retrieval and analysis. The gateways can also be used stationary with fixed geo-coordinates. During the end of each transport, the link to the transported unit load is removed automatically using BLE via this type of stationary gateway. This process is described as 'un-pairing'.

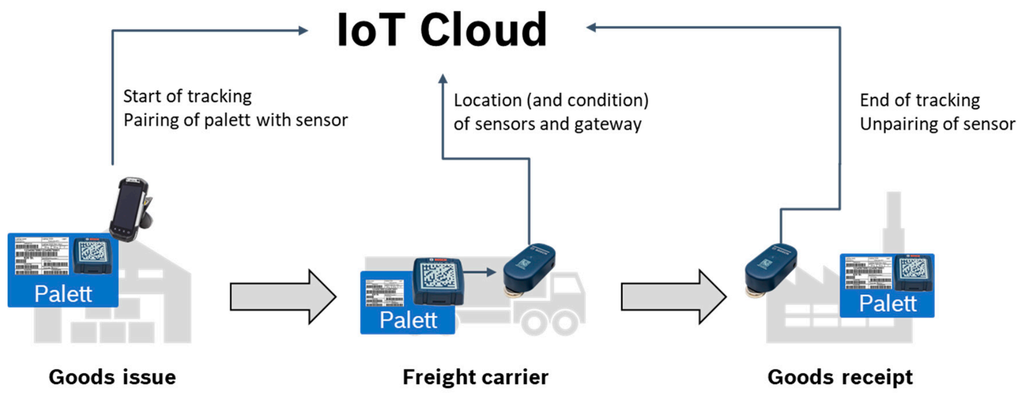

Figure 3. Architecture of the used tracking system.

Based on the previously described system, the collected data attributes are shown in Table 1. The system provides a table of tracking traces for each unit load between the pairing and un-pairing events.

Table 1. Collected tracking data from the tracking system.

Column	Description
Unit load ID	Unique identifier of paired transport unit load
Start/End of tracking	Pairing and un-pairing Unix time in seconds
Gateway ID	Unique identifier of transmitting gateway
Last Update	Unix time of transmission in milliseconds
Latitude, longitude	Geo-location of the last update
Temperature, humidity	Quality related data of sensor of last update

Apart from the sensor data, information considering the origin and destination location in the form of a geo-fence is provided. The geo-fence consists of a geo-coordinate and a radius.

5. Data Pre-Processing

In this chapter, two major tasks in the research will be presented: Data cleansing and feature engineering. Since GPS data can often be quite noisy, data pre-processing is required to increase the quality of the used data. Further, relevant features are derived from the cleansed data in order to build an accurate prediction model.

5.1. Data Cleansing

After a qualitative analysis of the sensor data, the data is cleansed based on the following rules:

- If the test pairings have no GPS transmissions falling within its destination geo-fence, they will be removed. This means that the goods have never reached the destination.
- If transmissions have a latitude and longitude set to 0°, the transmission will be removed and ignored as it implies the absence of a GPS signal.
- While storing the containers at the ports and at other intermediate stops, outliers have been detected. To identify outliers, the speed between two consecutive stops has been used, which exceeded 120 kmph in this case. Case 1 in Figure 4 shows one outlier where the speed exceeds more than 120 kmph to the previous and following transmission. At the same time, the speed between the transmissions before and after the outlier is significantly smaller, with a speed below 5 kmph. In Case 2 of Figure 4, two consecutive outliers have been detected, which are relatively close to each other. The speed between the outliers is below 5 kmph, while the speed to the previous and following transmission again exceeds 120 kmph. As before, the speed before and after the outliers is below 5 kmph. The transmissions identified as outliers based on the described rules have been removed, as shown in Figure 4.
- As only the transport itself should be considered, data points other than the last transmission in the origin geo-fence and the first transmission in the destination geo-fence are removed. This situation is shown in Figure 4, Case 3.

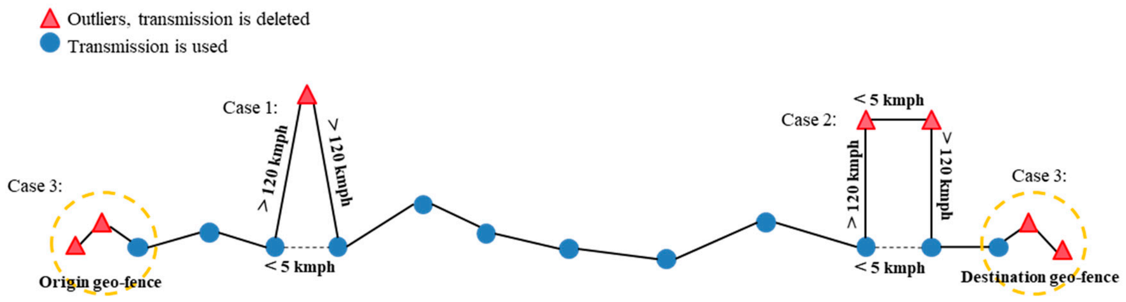

Figure 4. Data cleansing of the raw data.

In addition, one has to consider that we have received tracking data from a number of pallets per container. To avoid using the data from the same ride several times in the training and test set, only the tracking data of one pallet per transport is considered.

5.2. Clustering for Route Identification

As previously mentioned, our approach is only based on real-time tracking data from the material flow. This means that no information about the route and possible stops or trans-loading points is available. Thus, to include route information in the prediction model, those locations have to be identified based on the geo data provided. To do so, a clustering algorithm is required that can deal with geo data and noise without knowing the number of clusters. A suitable and proven algorithm for this purpose is density based spatial clustering of applications with noise (DBSCAN), [46] (p. 39) and [47].

Similar to [46] (p. 39), before applying the clustering algorithm on all tracking data, potential stop points are determined to decrease noise in the identified clusters. In a previously performed experiment, a gateway had been attached to a container, which was standing at the same position for two days. Hereby, noise of up to 300 m has been identified in the GPS data. Similarly, only tracking points with a distance smaller than 300 m either to the previous or next transmission are chosen for clustering. This rule applies to all points, P_1 to P_7. Further, to identify only significant stops, at least four consecutive points have to be identified, with a distance below 300 m between each other on a ride. This rule applies only to points P_1 to P_5. This circumstance is shown in Figure 5.

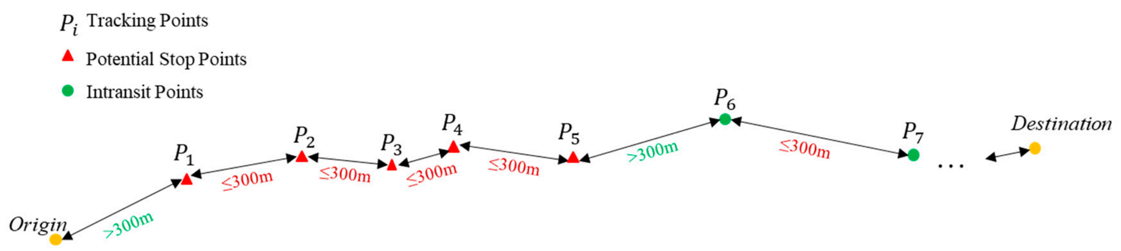

Figure 5. Identification of potential stop points.

Before applying the DBSCAN algorithm, a training and test set splitting is required, as the test set represents data unknown to the model and should not be considered to identify the clusters. The test set is also relevant for evaluation purposes, which are performed in Section 7. As some features depend on values of previous data points in a transport, the data is split by unit load and not by data transmission. Some previous experiments in this research have shown quite small prediction errors when having data points of the same unit load in the training and test data set. In this study, two transports out of seven in total have been randomly chosen as the testing dataset. Finally, only the training set is used for the identification of clusters.

For prediction, DBSCAN requires two parameters to check if data points are part of a cluster or outliers: *minPoints* and ε. ε is the maximum distance of a point to another to be considered as the same cluster and *minPoints* refers to the minimum required number of points below the distance of ε to be in the neighborhood. If this applies to a data point, this point is marked as a core point and the points in the neighborhood are marked as boarder points. Both types of points are part of the same cluster, but only from core points can additional points be added to a cluster. Thus, all identified boarder points are tested for core points as well. This process for a cluster is carried on until all boarder points have been tested. Then, the next unmarked data point is tested for a core point. If this point is a core point with the specified number of points below the distance of ε, it creates a new cluster with further boarder points to be tested. Otherwise, the point is marked as an outlier.

For considering significant clusters, *minPoints* has been selected as 10. To choose an adequate value for ε, a calculation based on a 10-NN-Distance graph, as shown in Figure 6, is proposed [47] (p. 230). The 10-NN-Distance refers to the distance of each point to its 10th nearest neighbor. By sorting the identified distances in ascending order, the graph in Figure 6 has been constructed. The distance value of the knee is chosen as ε equaling 416 m.

The algorithm results in 14 clusters, which is equivalent to the real number of stops. Figure 7 shows the identified clusters. One should note that a few clusters are quite close to each other in the figure. The number indicates the overlapping clusters. One can see that the ports in Charleston and Bremerhaven have been identified as clusters. One can also identify that in Norfolk stops happen in two terminals. To decrease complexity in the further modelling steps, only clusters that are visited more than once are considered, so that eight clusters remain. The potential stop points of the test set are then assigned to these eight clusters.

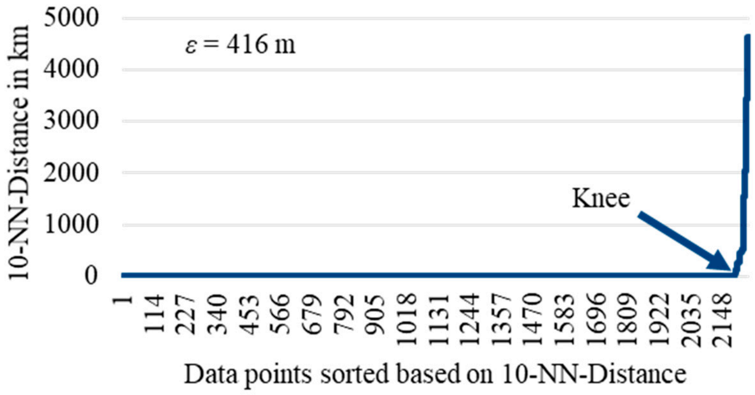

Figure 6. 10-NN-Distance graph to identify the value of the maximum distance of a point to another to be considered as the same cluster (ε) for the density based spatial clustering of applications with noise (DBSCAN) algorithm.

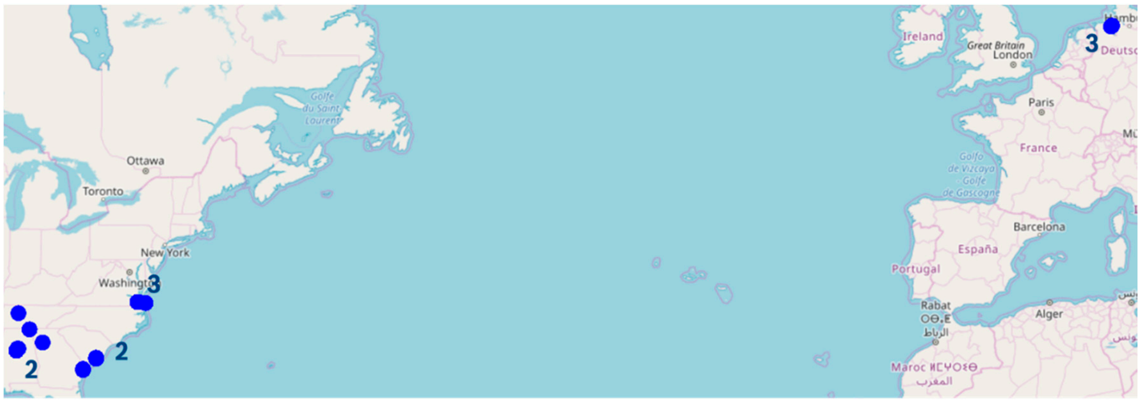

Figure 7. Resulted clusters of the clustering algorithm.

5.3. *Feature Engineering and Data Transformation*

According to the literature review and business context, the raw data has been transformed into features that are assumed to be relevant when modelling travel times. Our aim is to describe the behavior of travel time based on the tracking data without adding further data input. The determined features can be categorized as follows.

Features considering the departure time relate to the timestamp based on the last transmission in the origin geo-fence. This timestamp has been transformed into hour of the day, the time of day, departure month, and departure day of the week in numerical form, so that algorithms can process the data properly [2].

Further, features relate to the actual status of the ride. These features provide information considering the time, the current position, and the current number of transmissions [2].

Common features used by authors, such as [2,11,23], represent the relation to origin and destination considering distance, time, and average speed. The time to the destination is hereby the feature to predict, as it is only known after arrival at the destination.

To describe the characteristics of a ride without traffic information, features such as average speed, distance, and time to the previous transmission have been calculated. Based on those features, counters have been determined considering the speed between two transmissions to describe the driving behavior. Besides, the number of stops and the corresponding length of all stops at a given time have also been taken into account, using similar procedures as [35]. The calculation happens before applying the clustering, as described in Section 5.2.

Similarly, features considering the number of visited clusters and the total length of cluster stops have been determined. To consider the route in the prediction model, the previously visited cluster and the route path from the destination to the current point have been identified.

As we also assume that public holidays have a significant influence on the travel time, we have added features considering the day class of the departure day, the current day, and the next five days. The day of the week is classified into weekday, Saturday or bridge day, and Sunday or public holiday, [33] (p. 2) and [37] (p. 151).

Similar to [36], the created features can also be associated to different data sources for further evaluation. Table 2 shows the determined data sources.

Table 2. Description of determined data sources.

Data Source	Description
Tracking system (T&T)	Features derived from tracking data without complex transformations and further data sources
Country (Coun)	Country derived from a geo-coding application programming interface (API) based on geo coordinates
Day Class (DC)	Features considering the day class derived from the country and information considering public holidays
Clustering (Cl)	Features derived from the clustering

Finally, Table 3 gives an overview of the developed features, with the assigned data source in the column data. Furthermore, the features have been marked as categorical (Cat) or continuous. As categorical features are not ordinally scalable, they have to be encoded into numerical features. This means that each characteristic of each categorical feature is transformed into a binary numerical value. As many features are created by this procedure, which increases the model complexity, we have not applied the encoding procedure on the *PastRoutePath*. Instead, we have created one binary feature per cluster, which determines if a cluster has already been visited, or not.

Table 3. Developed features categorized into continuous and categorical and their domain.

Feature	Description	Cat.	Data
AbsoluteDistanceToPrevious	Absolute distance in kilometers from the previous position to the position.	No	T&T
AvgSpeedPrevious	Average speed from current transmission to previous transmission as quotient of AbsoluteDistanceToPrevious and DrivenTimePrevious.	No	T&T
CounterAvgSpeed0_1, CounterAvgSpeed1_30, CounterAvgSpeed30_60 and CounterAvgSpeed_above_60	Counter of the number of AvgSpeedPrevious from departure to the current transmission from 0 to 1 kmph, 1 to 30 kmph, 30 to 60 kmph, and over 60 kmph derived from the data. Those features are based on business knowledge to represent traffic conditions.	No	T&T
CounterStops	Number of detected stops. A stop is detected only if at least four data points with a distance below 300 m after each other are detected.	No	T&T
Current_DayOfWeek	Day of the current transmission from 1 to 7.	No	T&T
Current_DayClass	Day class of current transmission; 1: working day, 2: Saturday or bridge day, 3: Sunday or public holiday.	No	DC
Current_Lat and Current_Long	Latitude and longitude of the current transmission.	No	T&T
Current_MonthOfYear	Month from 1 to 12 of the current transmission.	No	T&T

Table 3. *Cont.*

Feature	Description	Cat.	Data
Current_TimeCategory	Period of the current transmission depending on the desired time interval. For example, a 2-h interval means that there are 12 intervals.	No	T&T
Current_TimeInHours	Hour represented as number between 0 to 23.99 of the current transmission.	No	T&T
CurrentCountry	Country based on the current position encoded according to ISO-3166 ALPHA-3 (3-letter encoding) using a geo-coder API.	Yes	Coun
CurrentGeofenceLocation	ID of the currently identified cluster, otherwise -1.	Yes	Cl
DayClassTomorrow	Tomorrow's day class based current timestamp.	No	DC
Departure_DayClass	Departure day class based on the last transmission in the origin; 1: working day, 2: Saturday or bridge day, 3: Sunday or public holiday.	No	DC
Departure_DayOfWeek	Departure day from 1 to 7 based on the last transmission in the geo-fence of the origin.	No	T&T
Departure_MonthOfYear	Month of departure from 1 to 12 based on the last transmission in the geo-fence of the origin.	No	T&T
Departure_TimeCategory	Time of departure based on the last transmission in the geo-fence of the origin and depending on the desired time interval. See also Current_TimeCategory.	No	T&T
Departure_TimeInHours	Hour of departure from 0 to 23 based on the last transmission in the geo-fence of the origin.	No	T&T
DestinationAbsoluteDistance	Absolute distance in kilometers from the position to the destination.	No	T&T
DestinationBearingToPosition	Direction from 0° to 360° from the point of the destination in which the unit load currently is located.	No	T&T
DestinationDrivenTime	**Time in hours from the current transmission to the destination. Variable to predict.**	**No**	**T&T**
DrivenTimePrevious	Time in hours from the previous transmission to the current transmission.	No	T&T
LastGeofenceLocation	ID of the last visited cluster.	Yes	Cl
LastRoutePart	Route considering the identified clusters until the current transmission.	Yes	Cl
PastActualDistanceToOrigin	Sum of absolute distances between the transmitted positions in kilometers.	No	T&T
PastAvgAbsoluteSpeed	Average speed from departure at the origin until the current transmission as a quotient of PastActualDistanceToOrigin and PastDrivenTimeToOrigin.	No	T&T
PastDrivenTimeToOrigin	Time in hours from departure at the origin until the current transmission.	No	T&T
QuantityDayClass2_nextday5 QuantityDayClass3_nextday5	Number of Saturdays or bridge days and Sundays or public holidays in the next five days.	No	DC
StandingTimeClusterInc	Standing time at the current cluster at a given time.	No	Cl
StatusOfRide	Status of current transmission. 'Origin' for being at the starting point, 'Driving' for being in transport, 'Stop' or 'Destination' for being at the destination.	Yes	T&T
TotalDurationOfStops	Sum of the length of all stops at a given time.	No	T&T
TotalLengthOfClusterStop	Sum of the length of all stops at a cluster at a given time.	No	Cl

6. Modelling

The following section deals mainly with the choice of relevant features in Section 6.1 and the hyper-parameters of the model in Section 6.2. Therefore, the cleansed and pre-processed data have already been divided into a training and a test set (see Section 5.2). The test set is relevant for evaluation purposes, which is performed in Section 7. For the modelling phase, only the training set is used.

6.1. Feature Selection

Based on a rough descriptive analysis of the processed data with the generated features, we are aware that not all features exert a positive and strong impact on travel time prediction. This makes feature selection highly important for this study before s modelling begins. According to [48], in this study we only consider filter methods, as these methods are generally less time consuming. Furthermore, easily interpretable metrics are used, and the feature selection can be performed independently from the training process. Finally, the aim of this chapter is to determine different experiments considering different combinations of features to apply to the machine learning algorithms. Therefore, five filter methods have been applied in our study:

- An easily interpretable method is one that uses a variance threshold. Features with a variation below a certain cut-off value have been removed. Therefore, for each feature, we identify the value with the highest frequency and divide it by the number of data points to calculate the percentage. In our study, features with at least 90% of their instances remaining constant are removed, and the threshold is set to 0.9 [49] (p. 509).
- F-regression runs an F test for each feature (regressor), relevant for travel time prediction apart from the target. Therefore, the variance of each feature and the target is calculated. The ratio provides the F value, which is then transformed into a p-value using the F probability distribution. Consequently, the p-value scores the individual effect of each regressor [50] (p. 629).
- The mutual information method considers non-linear dependencies as well as linear relationships by calculating the distance between the probability distributions of two features [51] (pp. 175–178). A feature selection procedure has been conducted by ranking the calculated mutual information of each feature.
- As per [48], the feature selection based on the ReliefF algorithm provided a superior performance in previous research. The applied algorithm considers feature interaction and is implemented in a way so that both continuous and categorical features can be processed. The algorithm calculates weights for each features, which are used for ranking them. Therefore, for a number of iterations, a random instance of all feature, which represents a row in the data, is chosen. For this chosen instance, the instances with the nearest hit and miss are identified based on the Manhattan distance. Finally, a diff-function is used to calculate the weight for each feature by subtracting it from the weight of the previous iteration. In the first iteration, the weight is set to zero. A higher weight means a better feature based on the ReliefF algorithm [48] (p. 487).
- Correlation based feature selection (CFS) is another well-known filter method, which calculates the correlation between predictor and target and then filters out features with a correlation below a given threshold [52] (pp. 361–362).

Figure 8 presents the overall structure of experiments created based on the selected filter methods. For F-regression, mutual information and ReliefF, features are ranked with certain scoring metrics and are subsequently selected for modelling as top 10, 15, and 20. For CFS, features have been filtered with three predefined thresholds for a moderate (0.5), high (0.7), and very high (0.9) correlation based on [53]. The use of all features has also been considered as an experiment. The application of those filter methods using different thresholds, as well as the use of all features for modelling, leads to 14 experiments.

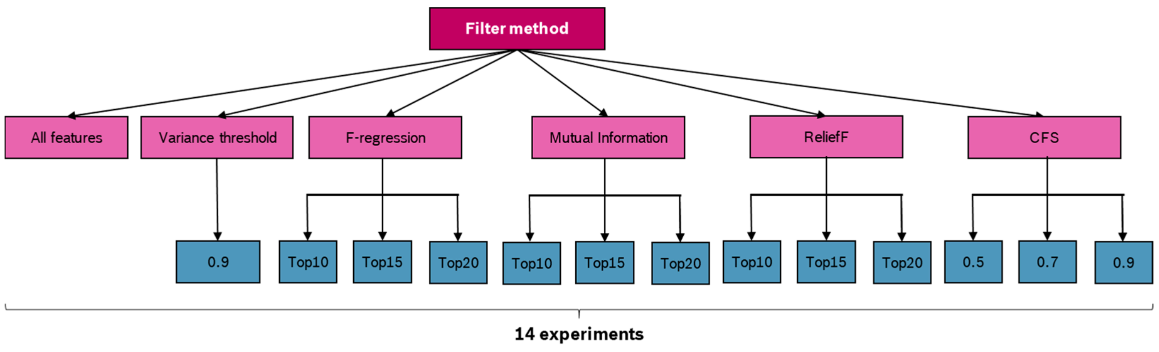

Figure 8. Resulting experiments based on applied filter methods (CFS: Correlation based feature selection).

To extend the variety of our research, the features of the data sources described in Table 2 have been assigned to three data domains based on the complexity required to compute them. The aim is to explore if the data sources influence the travel time prediction. The first domain refers only to the Tracking system (T&T) features, which can easily been derived from the data. The second domain further includes features derived from APIs, considering the country and public holidays, which include additional costs and integration effort. Finally, the last experiment further adds the clustering, which is based on more complex and time-consuming processing. To each data domain, the previously described 14 experiments are applied. This leads to 42 experiments in total (see Figure 9). The number of features in Figure 8 refers to the features shown in Table 3 after encoding. All those experiments will be finally applied to the three selected machine learning algorithms.

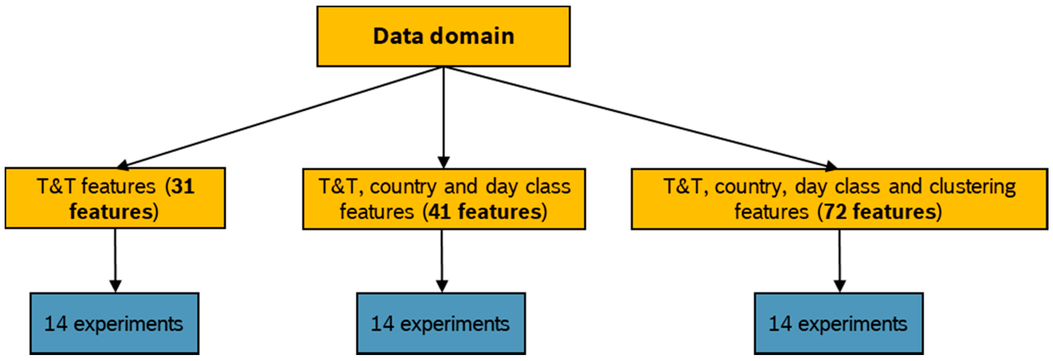
Data domain
T&T features (31 features)
T&T, country and day class features (41 features)
T&T, country, day class and clustering features (72 features)
14 experiments
14 experiments
14 experiments

Figure 9. All experiments considering determined data domains.

To get an idea of the relevance of the engineered features, for each data domain we have calculated the frequency of each feature in all experiments. This means that for each data domain we counted how often, based on the applied 14 experiments shown in Figure 8, a feature has been chosen. For example, if a feature is only chosen for modelling in the experiments using all features and features with a variance threshold above 0.9, the frequency is two. Figure 10 shows the results for the 20 features most often chosen, considering the data domain including the T&T, country, day class, and clustering features.

One can assume that *PastDrivenTimeToOrigin* is the most relevant feature, since it has been selected in 12 experiments. Further, basic features considering the distances to origin and destination as well as driven time to origin, which are often used in the literature, are identified as relevant. In addition, the self-developed counter of the average speed has been considered several times. Features considering the country or day class have been identified as less important for travel time prediction. However, some of them are still treated as important features for some feature selection methods.

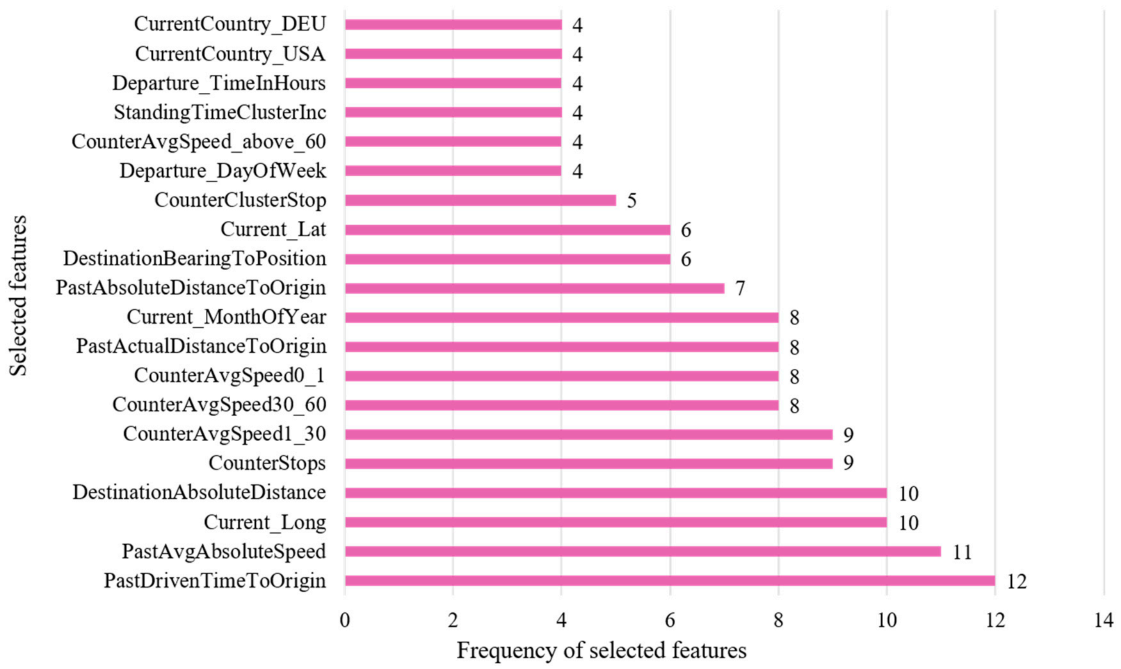

Figure 10. Top 20 selected features out of 14 experiments in the domain of the T&T, country, day class, and clustering features.

6.2. *Validation and Parameter Tuning*

The following phase deals with choosing the best parameters of the applied learning algorithms and experiments. Since we had a rather small number of unit loads, cross-validation is applied on the original training data set. The unit loads are again split into five random training and validation sets in order to better check the validity of the chosen parameters. Thereby, parameter tuning defines the process to choose the parameters with the best prediction accuracy for an applied learning algorithm using the validation dataset. Since each algorithm has many parameters, as described in Section 3.1, only the most relevant parameters are considered. The selected range of these features has been selected based on previously conducted experiments. The used parameters and corresponding values for each learning algorithm are shown in Table 4.

Table 4. Chosen parameters and defined range to apply grid search.

Learning Algorithm	Parameters

Support Vector Regression (SVR)	Kernel = 'rbf'
	C = [0.001, 5.001], step size 0.02
	ε = [0.001, 0.1], evenly split into 30 values

Extremely Randomized Trees (ExtraTrees)	n_estimators = [1, 10], step size 1
	random_state = [6, 18], step size 1

Adaptive Boosting (AdaBoost)	n_estimators = [1, 10], step size 1
	learning_rate = [0.0001, 1.0001], step size 0.025

To select the optimal parameter combination for each algorithm and experiment, a so-called grid search has been conducted. For each training set and for each possible combination of parameter values, the model is trained. The validation set is then used to calculate the prediction accuracy based on the mean absolute error (MAE, see Section 7.1). Then the average over all five MAEs of the five validation sets per parameter combination is calculated. The parameter combination, which returns the best average MAE, is then chosen for the corresponding model in accordance with [45] (pp. 247–248).

7. Evaluation

This chapter presents the evaluation results from all experiments by conducting meaningful comparisons and analysis through the results. Therefore, the test set is used to consider data that have not been seen by the model before. Section 7.1 evaluates the results based on the machine learning algorithms, while Section 7.2 compares the best machine learning model to average-based approaches.

7.1. Evaluation of Machine Learning Models

In this section, prediction results obtained from the testing phase based on data considering different feature combinations and various machine-learning algorithms have been compared and evaluated. Models created by SVR, ExtraTrees, and AdaBoost have been evaluated using mean absolute error (MAE) and root mean square error (RMSE), two of the most common evaluation metrics [1], to estimate the performance of each model.

The MAE calculates how close the predictions are to the actual outcome in terms of the Manhattan distance. It is simply the average of all prediction differences and is located on the same scale that the data is measured on [1] (p. 9). Equation (1) defines how the MAE is calculated. Parameter n is the number of observed data points in the test set.

$$\text{MAE} = \frac{1}{n}\sum\nolimits_{i=1}^{n}\left|\text{actual}_i - \text{prediction}_i\right|. \tag{1}$$

Since the MAE is based on the mean error, there is a certain chance that it understates the impact of large, but infrequent, errors. To adjust such large and rare errors, we also calculate the RMSE. The RMSE squares the differences before the mean is calculated and then takes the square root of the mean, as shown in Equation (2). Through this, a measure of the size of the error that gives more weight to the large but infrequent errors is achieved. The RMSE also provides information about the variance of errors of a model. The larger the difference between RMSE and MAE, the more inconsistent the error size.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\text{actual}_i - \text{prediction}_i\right)^2} \tag{2}$$

Based on the previously described metrics, Table 5 shows the results of the best conducted experiment per learning algorithm. The overall best result has been obtained by SVR using the ReliefF method for filtering, with a MAE of 16.91 h and RMSE of 22.7 h. The result is surprisingly good if we consider the small training set of five transports and a transport time of up to 30 days. The result of AdaBoost is still quite close to the SVR, while ExtraTrees has a significantly bad prediction accuracy, with 43.66 h.

Table 5. Best prediction results of each machine learning algorithm (MAE: Mean absolute error, RMSE: Root mean square error).

Model	Data Domain	Experiment	MAE	RMSE	Best Parameters
ExtraTrees	T&T + Day Class + Country + Clustering	Variance threshold 0.9	43.66 h	54.61 h	n_estimators: 12 random_state: 14
AdaBoost	T&T	Variance threshold 0.9	18.78 h	31.00 h	n_estimators: 14 learning_rate: 0.9251
SVR	T&T + Day Class + Country	ReliefF Top 20	**16.91 h**	**22.70 h**	C: 4.801 ε: 0.01

The same results can also be observed in Figure 11, where the MAE is shown in relation to the actual driven distance. One can see that the MAE of ExtraTrees is much larger than the other two. At the same time, AdaBoost and SVR provide a low prediction error. While AdaBoost provides a better prediction at the beginning, the prediction error of SVR decreases. At the end of the ride, one can also see that the error of SVR increases again to 12 h and AdaBoost to 24 h. Usually, it is easier to make an accurate prediction closer to the destination. The reason for this circumstance lies in the varying number of unknown stops between the port of Charleston and the train yard in Bessemer, as described in Section 4.1, which increases the complexity of the travel time prediction. Another reason could also be a quite long stop over at the train yard in Bessemer close the final destination.

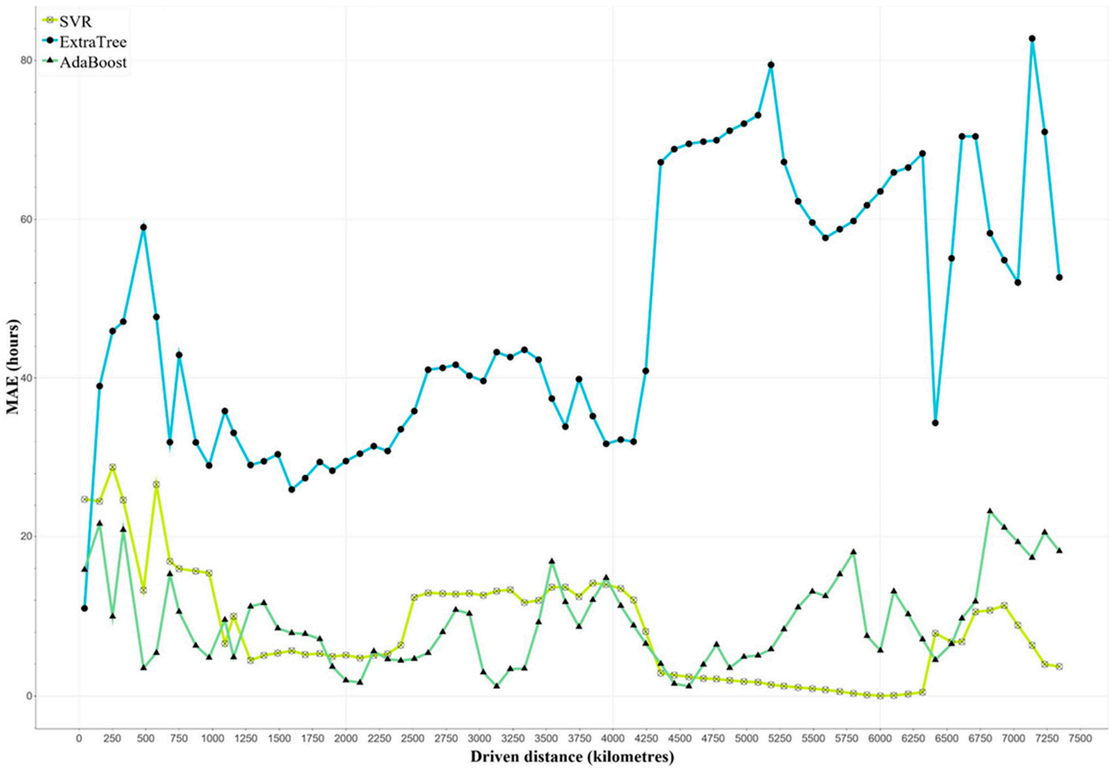

Figure 11. Distance based comparison of the best models per learning algorithm.

In Table 6, one can also see, that based on the average MAE and RMSE over all experiments SVR is independent of the experiments and data domains the best performing learning algorithm. The MAEs are quite stable, with MAEs in a range of 16.91 to 47.95 h, and SVR also provides, with 31.78 h, the lowest average MAE. Instead, the results of ExtraTrees and AdaBoost have a significant higher average MAE of above 100 h, despite the good prediction error of one of the experiments with AdaBoost. Furthermore, average MAEs are nearly the same when comparing the MAEs between the data domains, so that no significant improvement of the prediction accuracy can currently be expected when adding further data sources to the T&T data.

Table 6. Prediction Error comparison considering data domains.

Data Domain	Statistic	SVR		ExtraTrees		AdaBoost	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
T&T data	Average	31.00	38.88	108.39	119.94	103.67	113.83
	Minimum	21.87	27.49	50.27	58.91	**18.78**	**28.40**
	Maximum	47.95	56.74	125.43	134.81	119.05	131.40
	Standard Dev	7.59	9.09	18.07	18.50	24.70	27.21
T&T data, country and day class	Average	31.56	41.54	109.47	122.16	101.80	116.15
	Minimum	**16.91**	**22.70**	63.43	75.69	19.29	30.39
	Maximum	47.95	74.85	123.60	138.18	116.92	133.14
	Standard Dev	8.24	13.60	14.65	14.64	24.11	25.03
T&T data, country, day class and clustering	Average	32.78	42.31	107.63	119.56	101.55	115.65
	Minimum	18.37	25.47	**43.66**	**54.61**	19.29	32.39
	Maximum	47.95	64.44	128.39	138.67	114.97	130.34
	Standard Dev	8.65	11.48	20.00	20.18	23.75	24.12
Over all data domains	Average	31.78	40.91	108.50	120.55	102.34	115.21
	Minimum	16.91	22.70	43.66	54.61	18.78	28.40
	Maximum	47.95	74.85	128.39	138.67	119.05	133.14
	Standard Dev	8.20	11.63	17.73	17.96	24.21	25.51

7.2. Comparison to Average-Based Approaches

For further evaluation, a comparison to current approaches should be performed. As to the literature, usually a comparison to average-based approaches is performed, which is also applicable to the current research. Therefore, the following definitions are introduced:

- We are assuming that t transports have been completed. j determines one specific transport with $j = 1, \ldots, t$. $t+1$ is then the currently observed transport for prediction.
- We also assume n_j as the number of transmissions per transport j. Then i determines the number of transmissions of the current transport with $i = 1, \ldots, n_j$.
- ATT_j: Actual Travel Time of transport j.
- $ATT_{j,i}$: Actual Travel Time of transport j from transmission i to the destination.
- $ETT_{j,i}$: Estimated Travel Time of transport j and from transmission i.
- ATD_j: Actual Time of Departure of transport j.
- $ETA_{j,i}$: Estimated Time of Arrival of transport j and from transmission i.

The easiest approach is to use the moving average for travel time estimation, without updating it based on the current position of the transport. In this case, the average travel time of all completed transports, ATT_{avg}, is calculated and used for prediction. As one can see in Equation (3), ATT_{avg} is equivalent to the estimated travel time from the first transmission at the origin, $ETT_{t+1,1}$. The arrival time is then calculated as the sum of departure and average travel time, as shown in Equation (4). Thus, the estimated time of arrival is not updated during the transport.

$$ETT_{t+1,1} = ATT_{avg} = \frac{1}{t}\sum_{j}^{t} ATT_j \tag{3}$$

$$ETA_{t+1,i} = ATD_{t+1} + ATT_{avg} \tag{4}$$

The authors of [12] are also considering the moving average. However, in comparison to the previous approach they are considering the percentage of the completed distance of the current transport, $d_{p,t+1,i}$. As in our use-case, since the route is not known and the distance can vary from transport to transport, the currently completed distance, $d_{t+1,i}$, is divided by the average distance over all completed transports, d_{avg}. The percentage of the remaining distance to the destination, $1 - d_{p,t+1,i}$, is used in Equation (6) to predict the remaining travel time by multiplying it with the moving average, ATT_{avg} from Equation (3). Finally, the estimated travel time is again summed up with the time of departure to calculate the estimated time of arrival, $ETA_{t+1,i}$.

$$d_{p,t+1,i} = \frac{d_{t+1,i}}{d_{avg}} \tag{5}$$

$$ETT_{t+1,i} = \left(1 - d_{p,t+1,i}\right) * ATT_{avg} \tag{6}$$

$$ETA_{t+1,i} = ATD_{t+1} + ETT_{t+1,i} \tag{7}$$

In [23], the prediction is based on the current transmission and position of a transport. To estimate the travel time, all transmissions of previous transports within a sensitivity radius are identified, together with their actual travel times to the destination. The average of all travel times of the identified transmissions is then used to estimate travel time for the current transport based on its current position.

In [11], the prediction of travel time is based on a predicted speed, $v_{pred,i}$, of the current transport at the current transmission, i, derived from the distances to the origin, $d_{O,i}$ and destination, $d_{D,i}$. The formula is shown in Equation (8). Hereby, v_{avg} stands for the average speed from previous rides at the transport relation, and $v_{P,i-1}$ stands for the speed to the previous transmission. In Equation (9), the predicted speed, v_{pred}, is then used to calculate the estimated travel time by multiplying it with the distance to the destination, $d_{D,i}$.

$$v_{pred,i} = \frac{v_{avg} * d_{O,i} + v_{P,i-1} * d_{D,i}}{d_{O,i} + d_{D,i}} \tag{8}$$

$$ETT_{t+1,i} = v_{pred,i} * d_{D,i} \tag{9}$$

Table 7 compares the results of the presented average-based approaches to the SVR as the best-derived model from the previous section. The results show that travel time prediction based on machine learning algorithms is considerably more accurate. In particular, the SVR algorithm has a prediction error that is significantly lower than the average-based approaches. Still, the average based approach is quite close to the prediction of the SVR, especially if considering the RMSE.

Table 7. Prediction results of the average-based approaches compared to the SVR.

Model	MAE	RMSE
Our model	16.91 h	22.70 h
Moving average	24.03 h	24.03 h
[12]	79.48 h	94.90 h
[23]	45.41 h	54.27 h
[11]	224.84 h	247.86 h

Figure 12 presents the MAE based on the distance to destination and proves that the SVR model performs best. Instead, [11,12] have a significantly high MAE. The moving average comes closest to the results of the SVR model and can be identified by the constant line, as predictions are not actualized during the ride. The approach of [23] also proves a good prediction accuracy. Just at the end of the ride, the error increases significantly. Here, a good prediction error is of high importance. While evaluating those results, one should also consider that the high errors of the average-based approaches occur because of the use of a sporadic transmission in a 30-min interval, a low amount of data, and the absence of information about the trans-loading points.

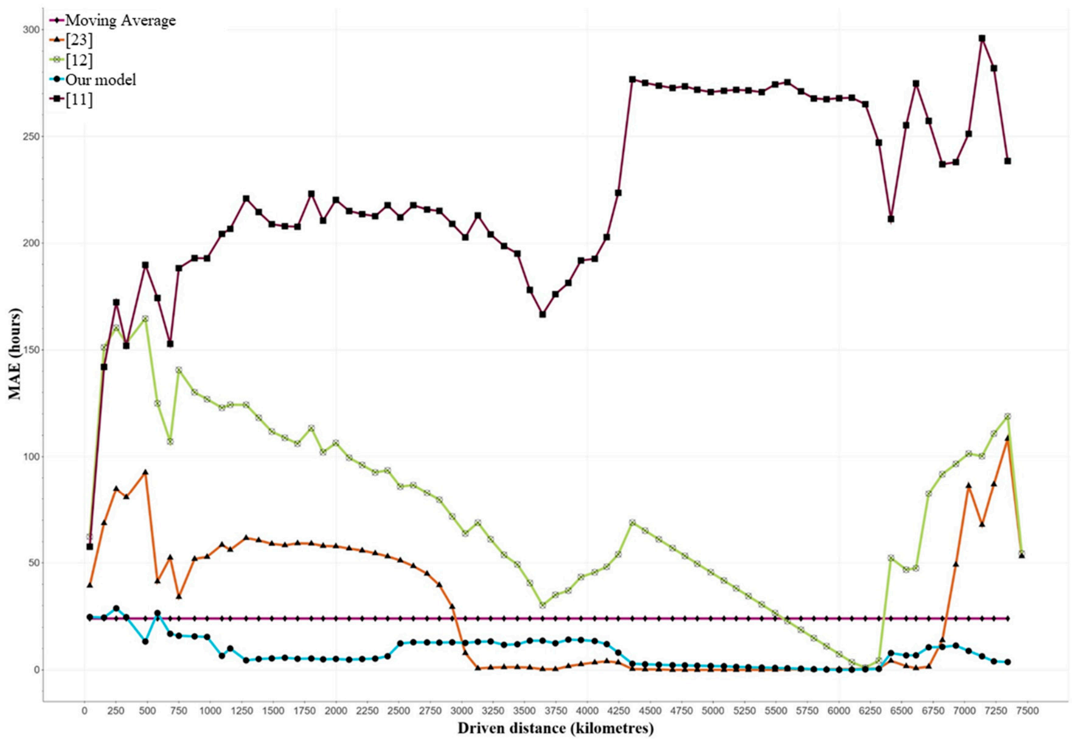

Figure 12. Comparison of SVR as the best model to the average-based approaches.

8. Conclusions

In this study, we have predicted travel times of multimodal transports using the machine learning algorithms ExtraTrees, AdaBoost, and SVR. Among the three applied machine learning algorithms, SVR has achieved the best performance in estimating the travel time. Depending on the experiment, the prediction error reaches up to 17 h over a transport time of up to 30 days. In addition, we could show that basic features directly derived from the tracking data, without adding further information considering the country or public holidays, already provide a good prediction. The other data domains do not improve the results significantly. Considering the low amount of transports and the complexity of the studied supply chain, we think that the estimation results are quite satisfactory. The SVR model also outperforms average-based approaches.

Despite the requirement to provide a prediction based on data available from only a small number of transports, the approach should be evaluated with more transports to further confirm the results and examine how availability of data influences the predictions. In addition, the approach should be applied to further transport relations, as the time behavior of transports can vary and other features might be relevant. Additional experiments could also be conducted with new combinations of features. For instance, AdaBoost and ExtraTrees provide an importance score to filter out unimportant features. Further data sources, such as weather, could also be evaluated, as transports by vessel highly depend on the weather conditions on the high sea. As the time prediction accuracy of multimodal transports highly depends on the route, a method could be implemented to predict the future route path. The prediction result could then be added as a feature for the travel time prediction.

Another highly relevant aspect is the construction of a productive digital travel time prediction service that implements the methods examined in this paper for real life usage. An appropriate architecture and infrastructure for sensor data generation, collection, and processing has to be defined. One big challenge results from the fact that data, which is generated locally and has to be transmitted over large distances, will be received in batches and will potentially not include all the transmissions of one unit load. This fact makes the pre-processing of the data more complex. Another challenge will be the re-training of the model based on new completed rides in an iterative process.

Author Contributions: Conceptualization, N.S., M.T., and M.F.; data curation, N.S. and X.L.; formal analysis, N.S. and X.L.; investigation, N.S. and X.L.; methodology, N.S.; resources, N.S., X.L., M.T., and M.F.; supervision, N.S.; validation, N.S., X.L., and M.T.; visualization, N.S. and X.L.; writing—original draft, N.S. and X.L.; writing—review and editing, M.T. and M.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the German Federal Ministry for Economic Affairs and Energy (BMWi), grant number 01MA16004.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Parolas, I.; Tavasszy, L.; Kourounioti, I.; van Duin, R. Predicition of Vessels' estimated time of arrival (ETA) using machine learning: A port of rotterdam case study. In Proceedings of the 96th Annual Meeting of the Transportation Research, Washington, DC, USA, 8–12 January 2017; pp. 8–12.
2. Masiero, L.P.; Casanova, M.A.; de Carvalho, M.T.M. Travel time prediction using machine learning. In Proceedings of the 4th ACM SIGSPATIAL International Workshop on Computational Transportation Science, Chicago, IL, USA, 1 November 2011; Thakuriah, P., Ed.; ACM: New York, NY, USA, 2011; pp. 34–38.
3. Teucke, M.; Broda, E.; Börold, A.; Freitag, M. Using Sensor-Based Quality Data in Automotive Supply Chains. *Machines* **2018**, *6*, 53. [CrossRef]
4. Lin, H.-E.; Taylor, M.A.P.; Zito, R. A review of travel-time prediction in transport and logistics. In Proceedings of the 6th Eastern Asia Society for Transportation Studies (EASTS) Conference, Bangkok, Thailand, 21–24 September 2005; pp. 1433–1448.
5. Zijm, H.; Klumpp, M.; Regattieri, A.; Heragu, S. Operations, Logistics and Supply Chain Management: Definitions and Objectives. In *Operations, Logistics and Supply Chain Management*; Zijm, H., Klumpp, M., Regattieri, A., Heragu, S., Eds.; Springer: Cham, Germany, 2019; pp. 27–42.
6. Sommerfeld, D.; Teucke, M.; Freitag, M. Identification of Sensor Requirements for a Quality Data-based Risk Management in Multimodal Supply Chains. *Procedia CIRP* **2018**, *72*, 563–568. [CrossRef]
7. Farahani, P.; Meier, C.; Wilke, J. Digital Supply Chain Management. 2020 Vision: Whitepaper. Available online: https://www.sap.com/documents/2017/04/88e5d12e-b57c-0010-82c7-eda71af511fa.html (accessed on 17 November 2019).
8. Li, X.; Bai, R. Freight Vehicle Travel Time Prediction Using Gradient Boosting Regression Tree. In Proceedings of the 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Anaheim, CA, USA, 18–20 December 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1010–1015.
9. Vlahogianni, E.I.; Karlaftis, M.G.; Golias, J.C. Short-term traffic forecasting: Where we are and where we're going. *Transp. Res. Part C Emerg. Technol.* **2014**, *43*, 3–19. [CrossRef]
10. Altinkaya, M.; Zontul, M. Urban bus arrival time prediction: A review of computational models. *Int. J. Rec. Technol. Eng.* **2013**, *2*, 164–169.
11. Singla, L.; Bhatia, P. GPS based bus tracking system. In Proceedings of the 2015 International Conference on Computer, Communication and Control (IC4), Indore, India, 10–12 September 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1–6.
12. Čelan, M.; Lep, M. Bus-arrival time prediction using bus network data model and time periods. *Future Gener. Comput. Syst.* **2018**. [CrossRef]
13. Kwon, J.; Petty, K. Travel time prediction algorithm scalable to freeway networks with many nodes with arbitrary travel routes. *Transp. Res. Rec.* **2005**, *1935*, 147–153. [CrossRef]
14. Shalaby, A.; Farhan, A. Prediction Model of Bus Arrival and Departure Times Using AVL and APC Data. *J. Public Transp.* **2004**, *7*, 41–61. [CrossRef]
15. Chen, M.; Liu, X.; Xia, J.; Chien, S.I. A Dynamic Bus-Arrival Time Prediction Model Based on APC Data. *Comput.-Aided Civ. Infrastruct. Eng.* **2004**, *19*, 364–376. [CrossRef]
16. Vanajakshi, L.; Subramanian, S.C.; Sivanandan, R. Travel Time Prediction under Heterogeneous Traffic Conditions Using Global Positioning System Data from Buses. *IET Intell. Transp. Syst.* **2009**, *3*, 1–9. [CrossRef]
17. Chien, S.I.-J.; Kuchipudi, C.M. Dynamic Travel Time Prediction with Real-Time and Historic Data. *J. Transp. Eng.* **2003**, *129*, 608–616. [CrossRef]
18. Fan, W.; Gurmu, Z. Dynamic Travel Time Prediction Models for Buses Using Only GPS Data. *Int. J. Transp. Sci. Technol.* **2015**, *4*, 353–366. [CrossRef]

19. Yu, B.; Lam, W.H.K.; Tam, M.L. Bus arrival time prediction at bus stop with multiple routes. *Transp. Res. Part C Emerg. Technol.* **2011**, *19*, 1157–1170. [CrossRef]
20. Zychowski, A.; Junosza-Szaniawski, K.; Kosicki, A. Travel Time Prediction for Trams in Warsaw. In Proceedings of the 10th International Conference on Computer Recognition Systems (CORES), Polanica Zdroj, Poland, 22–24 May 2017; Kurzynski, M., Wozniak, M., Burduk, R., Eds.; Springer International Publishing: Cham, Germany, 2018; pp. 53–61.
21. Sun, X.; Zhang, H.; Tian, F.; Yang, L. The Use of a Machine Learning Method to Predict the Real-Time Link Travel Time of Open-Pit Trucks. *Math. Probl. Eng.* **2018**, *2018*, 4368045. [CrossRef]
22. Servos, N.; Teucke, M.; Freitag, M. Travel Time Prediction for Multimodal Freight Transports using Machine Learning. In Proceedings of the 23th International Conference on Material Handling, Constructions and Logistics (MHCL), Vienna, Austria, 18–20 September 2019; Kartnig, G., Zrnić, N., Bošnjak, S., Eds.; University of Belgrade Faculty of Mechanical Engineering: Belgrade, Serbia, 2019; pp. 223–228.
23. Heywood, C.; Connor, C.; Browning, D.; Smith, M.C.; Wang, J. GPS tracking of intermodal transportation: System integration with delivery order system. In Proceedings of the 2009 IEEE Systems and Information Engineering Design Symposium, Charlottesville, VA, USA, 24 April 2009; Louis, G.E., Crowther, K.G., Eds.; IEEE: Charlottesville, VA, USA, 2009; pp. 191–196.
24. Kisgyörgy, L.; Rilett, L.R. Travel Time prediction by Advanced Neural Network. *Periodica Polytech. Ser. Civ. Eng.* **2002**, *46*, 15–32.
25. Wu, C.-H.; Ho, J.-M.; Lee, D.T. Travel-Time Prediction with Support Vector Regression. *IEEE Trans. Intell. Transp. Syst.* **2004**, *5*, 276–281. [CrossRef]
26. Šemanjski, I. Potenial of Big Data in Forecasting Travel Times. *Promet-Traffic Transp.* **2015**, *27*, 515–528. [CrossRef]
27. Zhang, Y.; Haghani, A. A gradient boosting method to improve travel time prediction. *Transp. Res. Part C Emerg. Technol.* **2015**, *58*, 308–324. [CrossRef]
28. Siripanpornchana, C.; Panichpapiboon, S.; Chaovalit, P. Travel-time prediction with deep learning. In Proceedings of the 2016 IEEE Region 10 Conference (TENCON), Marina Bay Sands, Singapore, 22–25 November 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1859–1862.
29. Jeong, R.; Laurence, R.R. Bus arrival time prediction using artificial neural network model. In Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems, Washington, WA, USA, 3–7 October 2004; IEEE: Piscataway, NJ, USA, 2004; pp. 988–993.
30. Pan, J.; Dai, X.; Xu, X.; Li, Y. A Self-learning algorithm for predicting bus arrival time based on historical data model. In Proceedings of the 2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, Hangzhou, China, 30 October–1 November 2012; Li, D., Yang, F., Ren, F., Wang, W., Eds.; IEEE: Piscataway, NJ, USA, 2012; pp. 1112–1116.
31. Dong, J.; Zou, L.; Zhang, Y. Mixed model for prediction of bus arrival times. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation (CEC), Cancún, Mexico, 20–23 June 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 2918–2923.
32. Treethidtaphat, W.; Pattara-Atikom, W.; Khaimook, S. Bus Arrival Time Prediction at Any Distance of Bus Route Using Deep Neural Network Model. In Proceedings of the 20th International Conference on Intelligent Transportation Systems, Mielparque Yokohama in Yokohama, Kanagawa, Japan, 16–19 October 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 988–992.
33. Zhang, J.; Gu, J.; Guan, L.; Zhang, S. Method of predicting bus arrival time based on MapReduce combining clustering with neural network. In Proceedings of the IEEE 2nd International Conference on Big Data Analysis (ICBDA), Beijing, China, 10–12 March 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 296–302.
34. Yang, M.; Chen, C.; Wang, L.; Yan, X.; Zhou, L. Bus Arrival Time Prediction using Support Vector Machine with Genetic Algorithm. *Neural Netw. World* **2016**, *26*, 205–217. [CrossRef]
35. Patnaik, J.; Chien, S.; Bladikas, A. Estimation of Bus Arrival Times Using APC Data. *J. Public Transp.* **2004**, *7*, 1–20. [CrossRef]
36. Kern, C.S.; de Medeiros, I.P.; Yoneyama, T. Data-driven aircraft estimated time of arrival prediction. In Proceedings of the 9th Annual IEEE International Systems Conference (SysCon), Vancouver, BC, Canada, 13–16 April 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 727–733.

37. Kee, C.Y.; Wong, L.-P.; Khader, A.T.; Hassan, F.H. Multi-label classification of estimated time of arrival with ensemble neural networks in bus transportation network. In Proceedings of the 2nd IEEE International Conference on Intelligent Transportation Engineering (ICITE), Singapore, 1–3 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 150–154.
38. Cortes, C.; Vapnik, V. Support-Vector Networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
39. Barbour, W.; Martinez Mori, J.C.; Kuppa, S.; Work, D.B. Prediction of arrival times of freight traffic on US railroads using support vector regression. *Transp. Res. Part C Emerg. Technol.* **2018**, *93*, 211–227. [CrossRef]
40. Awad, M.; Khanna, R. Support Vector Regression. In *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*; Awad, M., Khanna, R., Eds.; Apress: Berkeley, CA, USA, 2015; pp. 67–80.
41. Geurts, P.; Ernst, D.; Wehenkel, L. Extremely randomized trees. *Mach. Learn.* **2006**, *63*, 3–42. [CrossRef]
42. Kadiyala, A.; Kumar, A. Applications of python to evaluate the performance of decision tree-based boosting algorithms. *Environ. Prog. Sustain. Energy* **2018**, *37*, 618–623. [CrossRef]
43. Schapire, R.E. Explaining AdaBoost. In *Empirical Inference*; Schölkopf, B., Luo, Z., Vovk, V., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 37–52.
44. Van der Spoel, S.; Amrit, C.; van Hillegersberg, J. Predictive analytics for truck arrival time estimation: A field study at a European distribution centre. *Int. J. Prod. Res.* **2017**, *55*, 5062–5078. [CrossRef]
45. Swamynathan, M. *Mastering Machine Learning with Python in Six Steps. A Practical Implementation Guide to Predictive Data Analytics Using Python*; Apress: New York, NY, USA, 2017.
46. Yang, J.; Xu, J.; Xu, M.; Zheng, N.; Chen, Y. Predicting Next Location Using a Variable Order Markov model. In Proceedings of the 5th ACM SIGSPATIAL International Workshop on GeoStreaming (IWGS), Dallas, TX, USA, 4 November 2014; Zhang, C., Basalamah, A., Hendawi, A., Eds.; ACM: New York, NY, USA, 2014; pp. 37–42.
47. Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD), Portland, OR, USA, 2–4 August 1996; Simoudis, E., Han, J., Fayyad, U., Eds.; AAAI Press: Palo Alto, CA, USA, 1996; pp. 226–231.
48. Bolón-Canedo, V.; Sánchez-Maroño, N.; Alonso-Betanzos, A. A review of feature selection methods on synthetic data. *Knowl. Inf. Syst.* **2013**, *34*, 483–519. [CrossRef]
49. He, X.; Cai, D.; Niyogi, P. Laplacian score for feature selection. In Proceedings of the 19th Advances in Neural Information Processing Systems (NIPS) Conference, Vancouver, BC, Canada, 5–10 December 2006; Weiss, Y., Schölkopf, B., Eds.; MIT Press: Cambridge, UK, 2006; pp. 507–514.
50. Omer Fadl Elssied, N.; Ibrahim, O.; Hamza Osman, A. A Novel Feature Selection Based on One-Way ANOVA F-Test for E-Mail Spam Classification. *Res. J. Appl. Sci. Eng. Technol.* **2014**, *7*, 625–638. [CrossRef]
51. Vergara, J.R.; Estévez, P.A. A review of feature selection methods based on mutual information. *Neural Comput. Appl.* **2014**, *24*, 175–186. [CrossRef]
52. Hall, M.A. Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning. In Proceedings of the 7th International Conference on Machine Learning (ICML), San Francisco, CA, USA, 29 June–2 July 2000; pp. 359–366.
53. Mukaka, M.M. A guide to appropriate use of Correlation coefficient in medical research. *Malawi Med. J.* **2012**, *24*, 69–71.