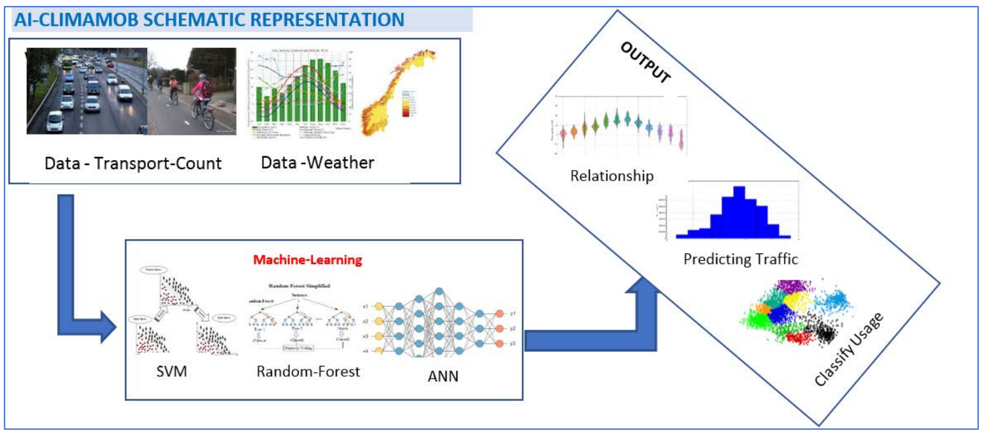
2017:00703 - Unrestricted

Report

Machine learning based investigation of influence of weather on transport mobility.

Author(s)
Mandar Tabib
Adil Rasheed



Influence of weather impact on transport mobility count

SINTEF Digital
Mathematics and Cybernetics
28th November 2017

SINTEF Digital
SINTEF Digital

Address:
P. O. box 4760 Sluppen
NO-7465 Trondheim
NORWAY

Telephone:	+47 73593000
Telefax:	+47 73594302

postmottak.digital@sintef.no
www.sintef.no
Enterprise / VAT No.:
NO 948 007 029 MVA

KEYWORDS:
Artificial Intelligence,
Machine Learning,
Climate, Weather,
Transport, Planning

Report

Machine learning based investigation of influence of weather on transport mobility.

VERSION
Final

DATE
28th November 2017

AUTHOR(S)
Mandar Tabib
Adil Rasheed

CLIENT(S)
TØI

CLIENT'S REFERENCE
Tanu Priya Uteng
Tanu.PriyaUteng@toi.no

PROJECT
102016466 AI-ClimaMOB

NUMBER OF PAGES AND ATTACHMENTS
34

ABSTRACT

Current work involves data-driven analysis of travel mobility. For this purpose, traffic counts of cars and bikes at various traffic routes along with the associated weather have been collected in Oslo, Norway. Amongst the 6 machine learning algorithms compared (linear regression, random forest, decision tree, gradient boost, support vector machine and artificial neural network), the random forest model is seen to perform the best with an accuracy score of around 0.96. The results indicate : a) bike traffic is more sensitive to weather than the car traffic, b) bike traffic counts showing a stronger bimodal distribution with the hour of the day for week days than the car traffic counts, thus suggesting a wider car-usage outside the bimodal peak-times. c) Monthly bike traffic counts is influenced by the weather, while the monthly car counts is influenced by the vacation periods.

PREPARED BY
Mandar Tabib,Adil Rasheed

SIGNATURE
[illegible]

CHECKED BY
Arne Morten Kvarving

SIGNATURE
[illegible]

APPROVED BY
Trond Runar Hagen

SIGNATURE
[illegible]

REPORT NUMBER
2017:00703

ISBN
978-82-14-06678-4

CLASSIFICATION
Unrestricted

CLASSIFICATION THIS PAGE
Unrestricted

SINTEF

Contents

List of Figures

1 Background and objectives

Daily travel activities are expected to be impacted by the climate change. In order to make strategic long-term transport and urban planning decisions, the climate change needs to be factored in. The proper way forward towards this direction is to start designing and testing newer tools and methodologies on the available credible data. In this work we devise some tools that are initially tested on available short-term weather data (which has more certainty of being accurate) as compared to long-term climate data (which generally has less certainty). This certainty in data helps in testing the proposed tools and methodologies. Later the tested tools, methodologies along with many of the results can be ported to climate change analysis. This makes sense as most of the variables defining atmospheric conditions in both the climate and the weather studies remains the same (such as temperature, humidity, wind-speeds, etc.), only their values and ranges may differ. The current work involves data-driven analysis of travel frequency based on weather conditions (such as temperature, precipitation, humidity and wind) using machine learning (which is an approach to artificial intelligence). For this purpose, the transport mobility data (of car and bike counts) along various traffic routes in Oslo, Norway for a period of one year have been considered. The weather data corresponding to these counts have also been collected simultaneously. The main objectives of this work are :

1. To compare and identify the best machine learning (ML) algorithms for the available data, and
2. use it to infer about user-behaviour and usage of different transport modes ($bikes, cars$) with regards to atmospheric conditions.

This usage and user-behaviour is analyzed at different locations (measured in terms of distance of an observed route from the city centre) and at different times of the day with regards to weather conditions. The research directions from climate change perspective that this work will try to address and the existing knowledge gaps that this work will try to fill are :

1. Need for development and demonstration of novel methodologies for potential use in transport planning with regards to climate change.
2. Inference about the user-behaviour (car-users and bike users) with regards to the existing atmospheric conditions for possible planning with regards to weather and climate.

The next section describes the methodology with regards to data collection, data preprocessing and model selection procedures. This is followed by the results and discussions.

2 Methodology

The methodology begins with collection of weather data and traffic count data for bikes and cars at different locations. This data is then preprocessed for machine learning analysis. Machine learning analysis will help us to identify the patterns in the data-sets. For the machine learning analysis, several well-known machine learning methods will be compared using standardized validation process for the pre-processed data. Then, the best performing machine learning data will be used to analyze the patterns. The details are described below in relevant sections:

2.1 Data collection

The data in its raw form were received in three different files: one for the location of the sensors in latitude and longitude, one for the data related to cars and another related to bikes. The location file had columns including the sensor id, sensor name, latitude, longitude, address of the location, format, sensor status, sensor type (counting cars or bikes) and measurement intervals. The car data included columns corresponding to sensor id, year, month, day, start hour, direction, measurement interval, total counts of the vehicles and average speed of the vehicles. The bike data consists of sensor id, time, lane, count of bikes and measurement intervals. The MET data consisted of the hourly recording of wind magnitude, wind direction, precipitation, humidity and temperature. The duration for which the data was available was from 01.02.2015 to 01.02.2016. The Appendix section provides some more information on data collection.

2.2 Data preprocessing

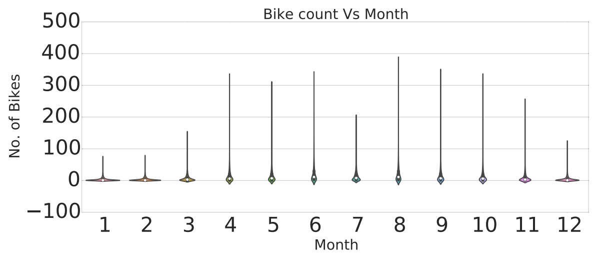
The collected traffic data needs to be processed. Data preprocessing methodology involves missing value detection and treatment, outlier detection and treatment, redundant and multi-colinear data removal and feature scaling. These are explained below.

2.2.1 Missing value

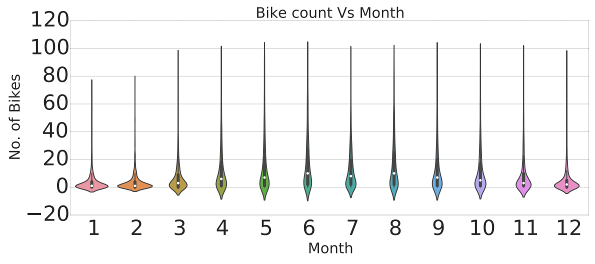
Missing values in the collected data refers to the data that are not-available for some reasons. They often get represented as blanks, NaNs or other placeholders in the data-set. Machine learning algorithms will not work with such representations as they need a value to be assigned to a variable in-order to unearth the pattern. In the current work, about 11% of the bike data was missing. For cars, most of the dataset was available. There are different strategies to deal with the missing data. One strategy is to avoid losing some information and to impute the missing values using either the mean, the median or the most frequent value of the row or column in which the missing values are located. This might create a bias in the data and alter results. We have used another basic strategy which is to discard entire rows and/or columns containing the missing values. However, this too comes at the price of losing the data which may be valuable (even though incomplete) but then it does not introduce a bias.

2.2.2 Outliers

Outliers are data that are exceptionally far from the mainstream of data. They can either be significant potential events/anomaly or irrelevant corrupted measurements. There are several methods to detect an outlier such as - extreme value analysis using standard deviations, probabilistic approaches, principle component analysis approach, proximity based modelling approach involving cluster analysis. Our approach was based on simple extreme value analysis in which the outlier points as those points that deviates by more than three times the standard deviation from the mean value, so $outlierdata = [data > (Mean + 3 * SD) or data < (Mean - 3 * SD)]$. In case of bike data, about 2% of data comes under the category of outliers. However, the removal of even such small data-set has some influence on the bike count distribution (Figure 1). Figure 1 shows that without outlier removal, we observe peak bike traffic exceeds more than 200 counts (measured in a 15 minute duration) multiple times, but after the application of outlier removal, these extreme peak counts are not seen. Generally, the outliers can be safely removed if they have, a) low frequency of appearance, b) are corrupted measurements, c) do not correspond to potential events/failures. We currently do not know whether these outliers are corrupted measurements or potential events (like, ongoing biking competition event), but since these peaks are less frequent in appearance so their removal is not expected to harm the analysis.



(a) Bike count with outliers



(b) Bike count without outliers

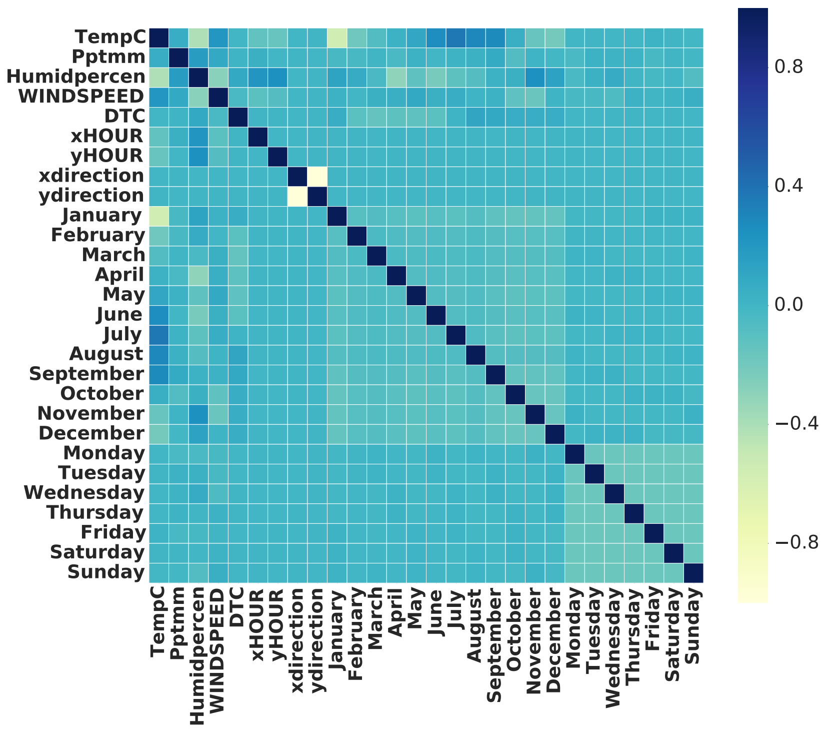
Figure 1: Influence of outliers on bike count

2.2.3 Multi-colinearity and redundancy

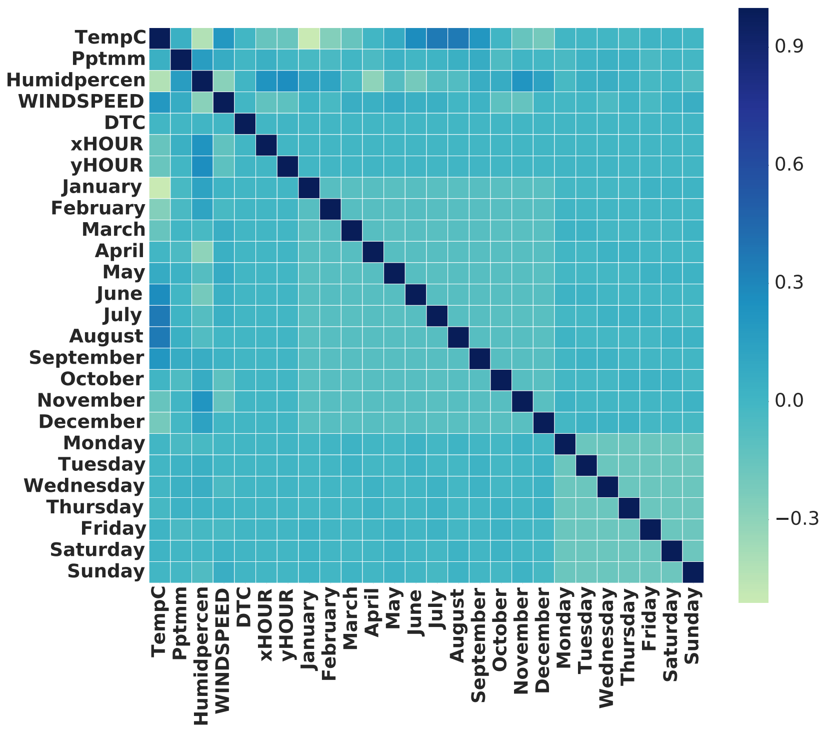
Multi-colinearity refers to existence of high correlations between two or more predictor (independent) variables. As a result, one independent predictor variable can be used to predict the other variable. This leads to redundant information in the dataset, which may skew the results in a regression model. Multi-colinearity is detected using the matrix of Pearson's bivariate correlation among all the independent variables (Figure2). Detecting multi-colinearity and removal of redundant variables can lead to :

1. A successful machine learning algorithm in terms of accurate numerical computation of matrix inversion,
2. A regression model that can better predict impact of an individual independent variable when other variables are fixed,
3. Avoidance of over-fitted models.

Figure 2 reveals mild correlation between months (January-December) and temperature for both the bike and car data, but this correlation value is not so high (<0.8) so as to deem the variables as redundant. Hence, all the variables are kept intact for further analysis.



(a) Car



(b) Bike

Figure 2: Correlation Analysis

2.2.4 Number of feature variables and data size

After removal of missing data, outliers and multi-colinearity check, the final data size for car and bike variables are : bike dataset has 77075 number of observations and car dataset has 101661 number of observations - which forms the rows on an input data matrix. The columns of this input data matrix comprises of a dependent variable (traffic counts) and several independent variables such as - station location (represented as distance from centre), wind speed, temperature, relative humidity, precipitation, lane direction, month, time, weekday.

2.2.5 Feature scaling and data standardization

Different variables used in the current work have different ranges for the variable values. If a feature has a variance that is orders of magnitude larger than others, then it might dominate the objective function and make the estimator unable to learn from other features correctly. The objective functions used in different machine learning estimators assume that all features are centered around zero and have variance in the same order. This assumption helps when the objective function in the majority of estimators need the Euclidean distance - the distance between two points (example - as needed by the radial basis function (RBF) kernel of Support Vector Machines, or the L1 and L2 regularizer of linear models). If the range of all features are normalized, then each feature contributes approximately proportionately to the final distance, and none of the feature variables dominate the objective function. In practice we often ignore the shape of the distribution and just transform the data - i.e. center it by removing the mean value of each feature, then scale it by dividing the non-constant features by their standard deviation. This transformation is termed as feature scaling, and this has now been done in this work.

2.2.6 Feature Engineering

Feature engineering is the process of using domain knowledge of the data to create new feature variables that make machine learning algorithms work. For applying feature engineering, it is necessary to know the type of feature variable. The feature variables in the current work can be categorized into categorical nominal variables (example - month, weekday), cyclic variables (example - hours and wind direction) and rest of the variables are continuous variables. The categorical variables will not be interpretable to machine learning codes and assigning numerical values to them will impart an ordinal property to them, which will not be correct. Hence, to make them interpretable, categorical variables are subjected to the ”one-hot encoding” procedure - where each categorical feature with m possible values is converted into m binary feature variables, and for every observation, only one of these m binary feature variable is active (with 1 as value) and rest are inactive (with 0 as value). Hence, care is taken to see that the training data is not missing any categorical features. The cyclic ordinal variables like ”hours” are also transformed into two newer variables in order to preserve relevant information (like ’23’ and ’0’ hour are close to each other and not far from each other). For a 24 hour clock, this is achieved through following transformation: $x = sin(2 * hour/24), y = cos(2pi * hour/24)$.

Another feature engineering process involves generating polynomial features for linear regression analysis. This helps to add complexity to the model by considering nonlinear features of the input data. For polynomial of order 2, say a feature variable X gets transformed from (X_1, X_2) to $(1, X_1, X_2, X_1^2, X_1X_2, X_2^2)$.

Next, this processed data is used for model selection as described below.

2.3 Model selection - Cross Validation and Learning curve

Cross-validation (CV) is a model validation technique for assessing the generalization ability of a machine learning algorithm to an independent data set. In our work, we split the original dataset into the training and the test dataset. The training dataset is subjected to a *k-fold cross-validation technique* which is used to build and estimate the performance of a machine learning algorithm and compare different machine learning algorithms. It helps to find the machine learning algorithm with best generalization ability for the given data. By building a machine learning algorithm, we mean finding suitable values of the parameters of a machine learning algorithm so that it performs well. Here, we have selected a $3 - foldCV$ procedure, where the training set is split into 3 different smaller sets. The model prediction is learned using 2 of these 3 folds at a time, and the 3rd fold left out is used for validation. Initially, a machine learning model is chosen along with a set of values for the hyperparameters. Here, we compare six machine learning models - linear regressor, random forest, decision tree, gradient boosting regressor, support vector machine and artificial neural network. For each machine learning algorithm and each parameter set, the following procedure is then applied over a "loop" for each of the 3 folds :

1. The chosen machine learning model with a selected parameter is trained using a different set of 2 folds each time ;

2. The resulting trained model is then validated on the remaining 3rd set (which is different in each loop).

The final performance measure reported by our 3-fold cross-validation is then the average of the values computed on the loop (over the 3 runs). This performance measure accuracy along with the average learning curve for each machine learning algorithm is used to select the best machine learning algorithm (and the best parameter set for it) for the given data. This best model is then applied on the initial test data set. This test data has so far remained hidden from the model during the model building and selection process. The matrix used for measuring performance is called as score (Y axis of all figures 3-7). The score is related to accuracy of the estimator. Best possible score is 1.0 suggesting a model with high accuracy and the score can be negative if the model gives the worse performance. A constant model that always predicts the same expected output disregarding the input features would get a score of 0.0. The learning curve shows the variation of average score with training data and validation data. Generally, training scores decreases with dataset (or the training error increases) and the validation scores increases with data-set (or validation error decreases). Learning curve for a given model estimator helps in its selection by showing the benefit from adding more training data to the estimator and showing whether a given estimator suffers more from a variance error or a bias error. Both these errors together serves to inform us about the expected generalization ability of a learning algorithm to unseen data. The errors are described below:

1. Bias Error - Bias of an estimator (machine learning algorithm) is the overall difference between this estimator's predicted value and the actual value of the parameter being estimated over the training data. If both the validation score (accuracy) and the training score (accuracy) converge to a value that is too low with increasing size of the training

set, then the estimator is under-fitting (has high bias error) and we will not benefit much from more training. We will then probably have to use a different estimator or a parametrization of the current estimator that can learn more complex concepts (i.e. has a lower bias) or add new features to the dataset.

2. Variance Error - Variance error refers to sensitivity of an estimator to data or new data and accounts for variability in the model predictions. With different datasets, a learning algorithm's predictive ability changes. Variance is measured in terms of deviations in the model's predictions of the expected parameter from its predicted mean over the training data. This error does not account for actual value of the parameter (as the bias error). High variance models can start modelling noise in the data as well and lead to overfitting. If the training score is much greater than the validation score for the maximum number of training samples (i.e. we have higher validation error than training error), then the estimator is over-fitting (has a higher variance error) and adding more training samples will be beneficial. With additional training data, the model will most likely increase generalization (i.e. reduce validation error). High variance errors can also be reduced by introducing a regularization term in the objective functions. The regularization term (also called penalty term) combats over-fitting by constraining the size of the weights. Increasing regularization may fix high variance (over-fitting) by encouraging smaller weights, resulting in a decision boundary plot that appears with lesser curvatures. Similarly, decreasing regularization may fix high bias (under-fitting) by encouraging larger weights, potentially resulting in a more complicated decision boundary.

In this section, we compare the model performances based on the results from learning curve and mean daily error on the test data. If both training and validation score are converging to a higher value, then we can say that it is a good model.

2.3.1 Multivariate Linear Regression - Degree 1 and 2

In multivariate regression we seek a set of parameters (or weights) Θ associated with features $\mathbf{X}$ for the hypothesis $\mathbf{H}$.
Hypothesis is $\mathbf{H} = \mathbf{X}\Theta$
with Θ as

$$\Theta = [\theta_0 \quad \theta_1 \quad ...\, \theta_i \quad ...\, \theta_m]^T \tag{1}$$

The parameters Θ are chosen so as to minimize the cost function. The cost function is $\mathbf{J}(\Theta) = \frac{1}{2m}(\mathbf{X}\Theta - \mathbf{Y})^T(\mathbf{X}\Theta - \mathbf{Y})$
Generally for very large problems due to memory requirements gradient descent algorithm is used for minimization but since here we are working with relatively small dataset with smaller number of feature vectors X, we have used direct method for finding the minima. This involves invoking the closed-form solution to linear regression

$$\Theta = (X^T X)^{-1} X^T Y \tag{2}$$

The advantages of linear regression are that they are computationally efficient, simple and easy to interpret. However, the algorithm fails to capture non-linear behavior.

Here, two different multi-variate linear regression models are tested, one with degree 1 and the other with degree 2 polynomial. With higher degree polynomial, the number of features increases and the model becomes more complex. Higher degrees may reduce bias errors but increase variance errors leading to overfitting. The degree 2 model is called a linear model because in the feature space, it varies linearly with the new features (X_1^2, X_1X_2, X_2^2), even though the regression equation when viewed from perspective of its relationship with original variables X_1, X_2 is non-linear. In machine learning, the linearity is viewed from the perspective of parameters Θ associated with the feature $\mathbf{X}$. Degree 2 polynomial is deemed suitable as shown from results below.

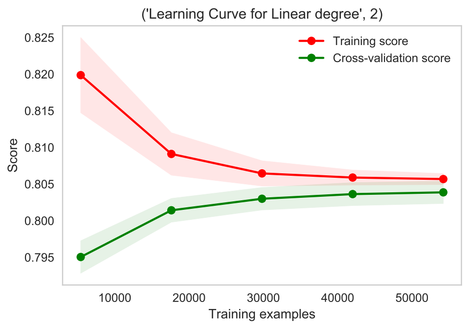
Figure 3 shows both the learning curve on cross-validation dataset and the predicted fit vs measured observation on hidden test dataset for degree 2 linear regression. The learning curve and fit plots are shown for both the car and bike traffic count data. For car-data, learning curve reveals that final validation and training scores have high values in range of 0.8-0.81. For car test data, linear regression with degree 2 polynomial is providing decent fit between the predicted and observed (as seen in Figure3) and the corresponding mean absolute error (MAE) in trip count is 222, while a higher MAE of 293 was obtained for degree 1 linear regression. For the bike-data, learning curve reveals that the linear regression might be underfitting (or showing higher bias) as both validation and training scores have lower scores in vicinity of 0.35-0.36, with the training score being slightly higher. For the bike test data, linear regression is not able to provide a good fit and the corresponding mean absolute daily error of bike trip count is 5.4 (for degree 2 polynomial regression). For degree 1 regression, the MAE is higher at 6.381 daily error. For bike data, we will need to test with a different estimator with more complexity. The linear regressor models are compared to other complex machine learning models in the next sections.

2.3.2 Tree based Regression

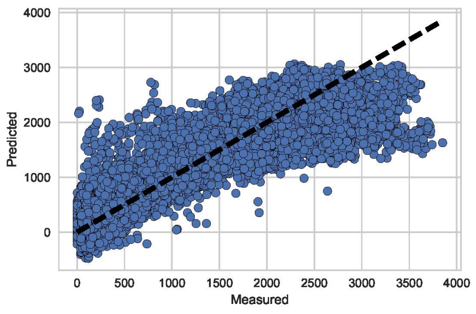
If the relationship between dependent and independent variables has high non-linearity and complexity, then a tree model is expected to outperform a classical linear regression method. The following three tree models have been tested : Decision Tree, Random Forest and Gradient boosting. A brief description of these models along with the parameters that have been used in them are given below :

Decision Tree

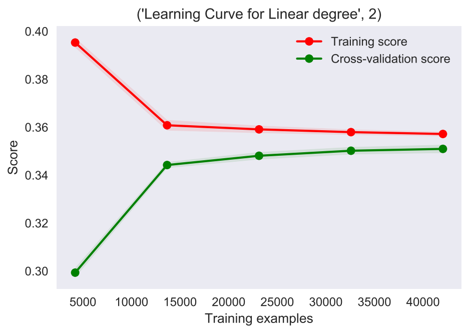
Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Some of the disadvantages of decision trees are that the algorithm can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble. The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement. There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems. Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance



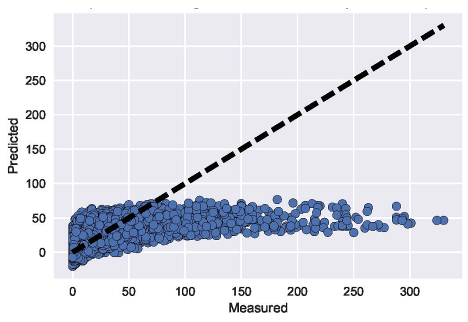
(a) Car



(b) Car



(c) Bike



(d) Bike

Figure 3: Linear Fit and Learning curve for car and bike data

the dataset prior to fitting with the decision tree. The advantage of this algorithm is that the results are highly interpretable. The important parameters in a decision tree model used here are :

1. The maximum depth of the tree (higher the depth, lesser the bias related errors) - 8 selected here.
2. Maximum features - the maximum number of features selected here to consider for a split = number of features.
3. Minimum sample leaf size - the minimal sample required to be at a leaf node - 3 selected here.
4. Minimum samples for a node split to control overfitting.
5. Minimum samples for a terminal node.

Random Forest
Random forest model creates a set of decision trees from randomly selected subset of training set and subset of features. At each branch in a decision tree, the Random Forest sub-samples the features in addition to the training examples. This process further de-correlates the individual trees reducing the variance errors. It then aggregates the votes from different decision trees to decide the final outcome. This is supposed to control over-fitting. The parameters used in this work are similar as decision tree with some additional parameters. The additional parameter are :

1. Maximum number of trees in the random forest (higher the trees, lesser the variance)- 55 selected here.
2. The maximum depth of the tree (higher the depth, lesser the bias related errors).
3. Maximum features - the maximum number of features the Random Forest is allowed to try in an individual tree is selected as square root of number of features. It helps in variance reduction by reducing the correlation between trees in the ensemble. By randomly selecting 'm' attributes at each split, some randomness is fit in to the ensemble and this reduces the correlation between trees.
4. Minimum sample leaf size = 3.

Gradient boosting
Gradient boosting tree based regression involves an ensemble of weak shallow decision trees. The shallow decision trees have higher bias errors and lower variance errors, which suits boosting procedure. Boosting procedure manipulates the training set to focus on areas of high error. Initially, it trains the model on an original training data and initialize the overall predictor as just this single model.It then assesses the error of the overall predictor and modify the training data by redefining the supervised prediction target to be some kind of residual between the ground truth and the overall predictor. It then trains a new model on the modified training data, and add to the overall predictor. This process continues. The parameters used here are:

1. Learning rate (0.12),

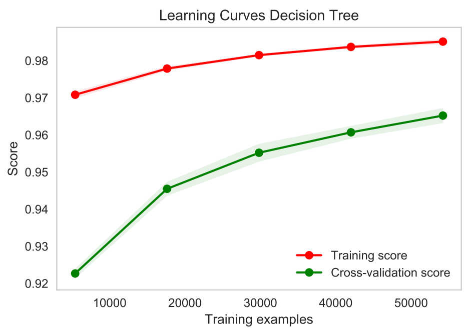
2. Maximum number of estimators (150),
3. Maximum depth (8).

Performance of the tree based regression models
Amongst the tree based models, the performance of Random forest and Gradient boost are in similar range in terms of accuracy (score range 0.95-0.985). Figure4 shows the learning curve for the four models (random-forest,decision tree,gradient boost,ensemble model) obtained from cross-validation, and Figure 5 shows the predicted fit vs measured observation on the test data for the four models. From car-data random forest learning curve (Figure 4) , it can be seen that random forest estimator have validation and training scores in range of 0.955-0.975. Performance of random forest is much better than that predicted by linear regression with degree 2 polynomial (which has 0.8 score). For random forest, the training score is mildly higher than the cross-validation score so a bit of over-fitting is there, which is in acceptable limit. Random forest is also seen to be giving good fit with the observed traffic count for the car test data (as seen in Figure 5) and the corresponding mean absolute error (MAE) is 39. The Random forest with a MAE of 39 daily trips is better than Gradient boosting (MAE=43), Decision tree (MAE=92) and Ensemble (MAE=53) models. But, these test performance errors for all the four tree based regression models are much lower than that reported by degree 2 linear regression (MAE of 222). So for the car traffic count, the random forest is performing the best.

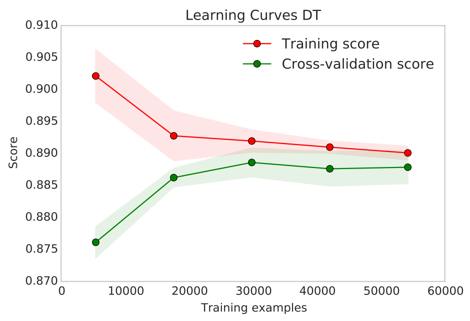
For bike data as well, the learning curve (Figure4) shows that the random forest validation score (around 0.62) is much better than the linear regression validation score (around 0.35). But, a bigger variation between the final training score (0.82) and the final validation score (0.62). This suggests that the estimator is over-fitting (has a higher variance error) and adding more training samples will be beneficial as it might help the model to generalize better (thus improving the validation score). Like car, for bike test data as well, the Random forest is seen to be giving good fit with the observed traffic count as seen in Figure 5. Performance on bike test data reveals an mean absolute error in daily trips (MAE) of 2.04 (for Random Forest), 2.08 (for Gradient boosting) , 2.33 (for decision tree) and 2.04 (Ensemble). This test performance error is much lower than that reported by linear regression (MAE of 5.4). So for the bike traffic count, the random forest is performing better and its generalizing ability can be improved further with more data.

2.3.3 ϵ-Support Vector Regression (SVR)

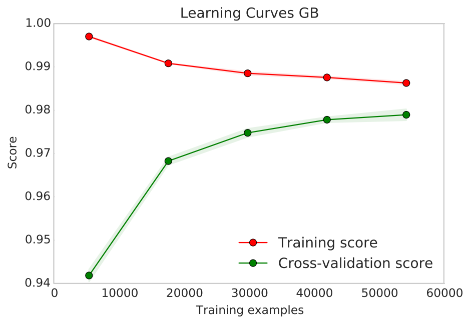
Theory
The SVR performs linear regression in a higher dimensional space. It constructs a hyperplane (decision surface) in a high or infinite-dimensional space. In SVR, the support vectors (SV) are the data points that lie closest to this decision surface (or hyperplane) and in-fact the support vector specifies the hyperplane. In ϵ-SVR, the hyperplane is found in such a way that there can be utmost ϵ deviation from the actually obtained targets y_i for all the training data and at the same time is flat as possible in feature space. In other words we do not care about errors as long as they are less than ϵ but will not accept any deviation larger than this. By flatness in SVR, we mean small weights w which mean to minimize the Euclidean norm $w^T w$. Smaller values of weights mean that the constructed SVM model is less sensitive to errors in measurement/random shocks/non-stationarity of the features. Mathematically, the equations are represented as follows - Given a set of data points,$\{(x_1, z_1), \ldots, (x_l, z_l)\}$, such



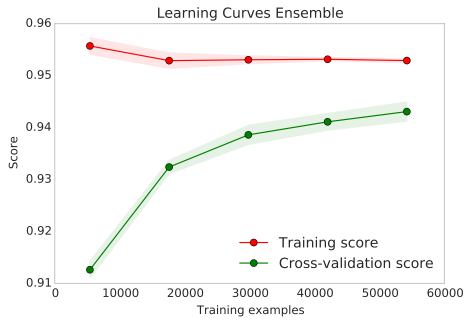
(a) Car-RF



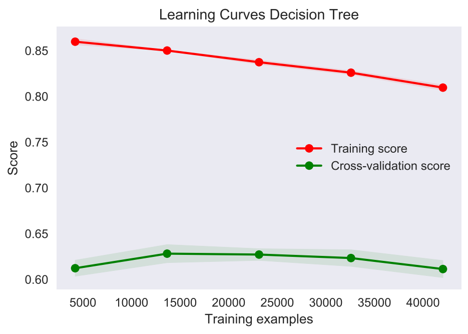
(b) Car-DT



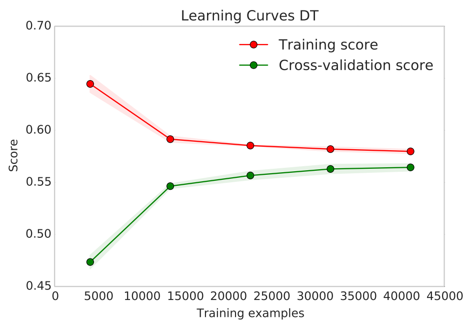
(c) Car-GB



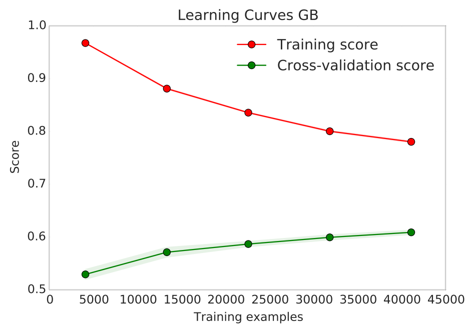
(d) Car-Ensemble



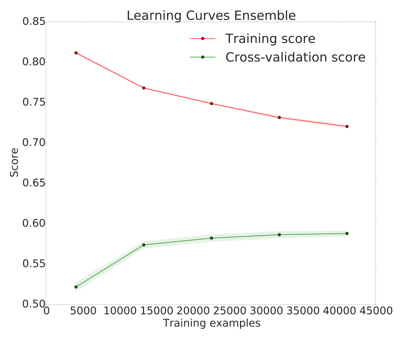
(e) Bike-RF



(f) Bike-DT

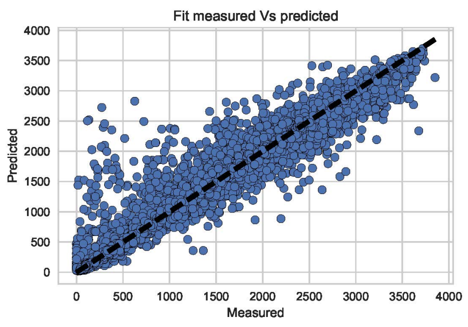


(g) Bike-GB

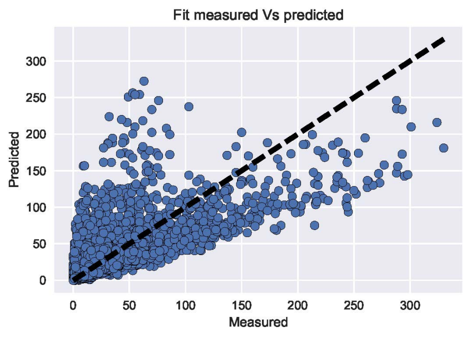


(h) Bike-Ensemble

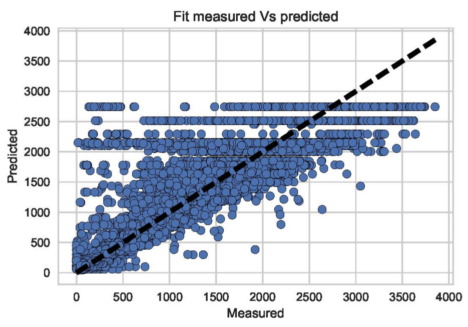
Figure 4: Learning curves from four models for car and bike data



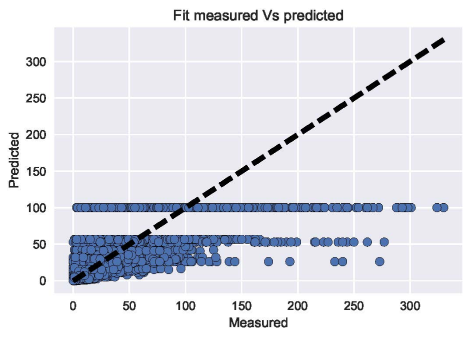
(a) Car-RF



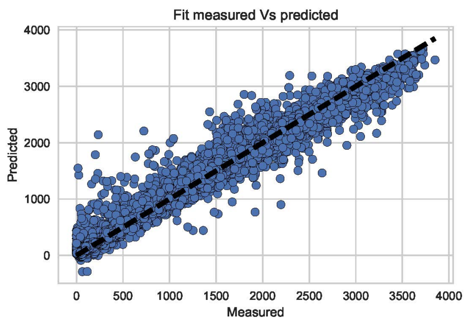
(b) Bike-RF



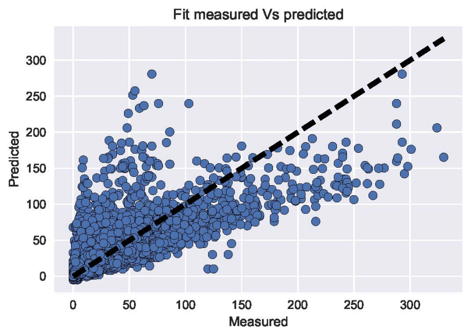
(c) Car-DT



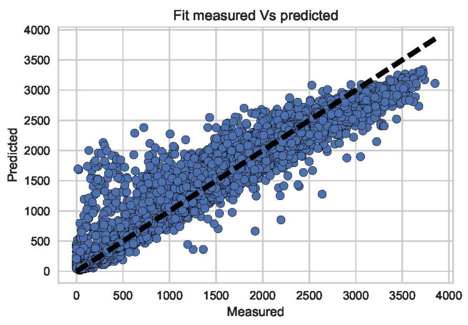
(d) Bike-DT



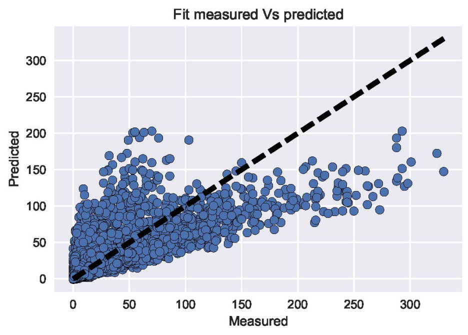
(e) Car-GB



(f) Bike-GB



(g) Car-Ensemble



(h) Bike-Ensemble

Figure 5: Decision Tree, Gradient Boost, Ensemble Fit with Random Forest for car and bike data

that $x_i \in R^n$ is an input and $z_i \in R^1$ is a target output, the standard form of support vector regression is:

$$\begin{aligned} \min_{w,b,\xi,\xi^*} \quad & \frac{1}{2}w^T w + C\sum_{i=1}^{l}\xi_i + C\sum_{i=1}^{l}\xi_i^* \\ & z_i - w^T\phi(x_i) - b \leq \epsilon + \xi_i, \\ & w^T\phi(x_i) + b - z_i \leq \epsilon + \xi_i^*, \\ & \xi_i, \xi_i^* \geq 0, i = 1, \dots, l. \end{aligned}$$

To make the optimization feasible, we introduce slack variables ξ, ξ_* which is analogous to creating a soft-margin loss function. The constant C determines the trade of between the flatness f and the amount up to which deviations ϵ larger can be tolerated. It also helps in controlling overfitting and underfitting as mentioned below. To the equation above, two procedures are performed -

1. Kernel trick - It helps in transforming the data to higher dimensions for accommodating non-linearity. This involves replacing the dot product with a non-linear kernel function.
2. The representation of the solution/model in the dual rather than in the primal.

As a result of these two procedures, the final model is represented as combinations of the training points rather than a function of the features and some weights (as shown in equations below). In other words, w can be completely described as a linear combination of the training patterns x_i. In a sense, the complexity of SVR model is independent of the dimensionality of the input space X and depends only on the number of SVs.

The dual is:

$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + \epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{l} z_i(\alpha_i - \alpha_i^*) \\ & \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0, 0 \leq \alpha_i, \alpha_i^* \leq C, i = 1, \dots, l, \end{aligned} \tag{3}$$

where $Q_{ij} = K(x_i, x_j) \equiv \phi(x_i)^T\phi(x_j)$. The optimization will provide α_i values. The SVMs decision function thus depends on some subset of the training data, called the support vectors. As noted, the weight is not explicitly obtained in this non-linear transformation. The decision function is:

$$f(x) = \sum_{i=1}^{l}(-\alpha_i + \alpha_i^*)K(x_i, x) + b.$$

The performance of a SVM model (including overfitting/underfitting) can be controlled by choosing the right hyper-parameters. These parameters are typically - the kernel functions $K(x_i, x)$ (linear, gaussian, radial basis functions), regularization parameter (C). The parameter C, common to all SVM kernels, trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth and model can have high bias error (underfitting), while a high C aims at classifying all training examples correctly with the model susceptible to high variance error (overfitting). When training an SVM with the Radial Basis Function (RBF) kernel, additional γparameter must be accounted. γ defines how much influence a single training example has. The larger γ is, the closer other examples must be to be affected. Hence, we select the parameters over a grid-search space.

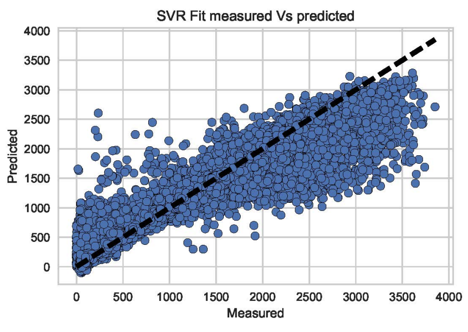
Parameter selection for best SVM model
The best SVM model in following parameter space has been explored - kernel types (Radial basis function (RBF), linear), Penalty parameter C of the error term (C from 10 to 100) and Kernel coefficient for RBF (γ from 0.1 to 50). The results from the cross-validation study indicated that the best model fit for car data is for SVR parameters : RBF Kernel, C=100, γ=1, and for the bike data, the best SVR parameters are: RBF Kernel, C=100, γ=0.1.

Performance of the best SVM model
For both car and bike data, Figure6 shows the learning curve obtained by applying the support vector machine (SVM) model and the predicted SVM fit vs measured observation for the test data. From learning curve plot for the car data (Figure 6) , it can be seen that SVM has final validation and training scores in range of 0.84-0.86. Performance of SVM is not as good as random forest (around 0.95) but is much better than that predicted by linear regression with degree 2 polynomial (0.8 score). Even for SVM (like random forest), slight over-fitting is there with training score more than validation score but it is acceptable. SVM is also seen to be giving decent fit with the observed traffic count for the car test data (as seen in Figure 6) and the corresponding performance error in terms of mean absolute error (MAE) is around 83 daily trip errors, which is worse than that predicted by Random forest (39) and Gradient boosting (43) but much better than linear regression (MAE of 222). So for the car traffic count prediction too - considering the cross-validation accuracy and generalization ability, the random forest is performing better than SVM and linear regression.

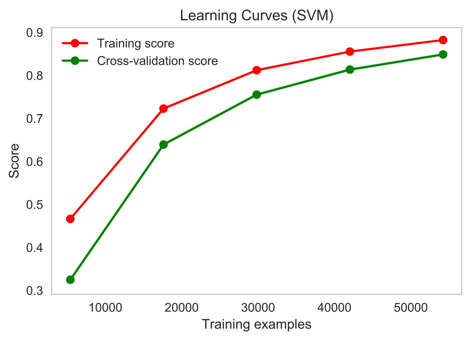
From learning curve plot for the bike data (Figure 6), it can be seen that SVM has final validation and training scores in range of 0.46-0.48. The trend of models (Linear regression, decision trees and now SVM) giving lower accuracies for the bike data continue. Amongst these models, the performance of SVM is not as good as random forest (score around 0.62) but SVM is much better than linear regression with degree 2 polynomial (which gives 0.35 score). Even for SVM (just like random forest), slight over-fitting is seen with training score more than validation score but it is acceptable. For bike test data, the SVM predictions are showing wide diversion from the observed bike traffic count (as seen in Figure 6) and the corresponding performance error in terms of mean absolute error (MAE) is around 2.09 daily trips, which is slightly less than that predicted by Random forest (MAE=2.04) but much better than linear regression (MAE of 5.4). So for the bike traffic count prediction too - considering the cross-validation accuracy and generalization ability, the random forest is performing better than SVM and linear regression.

2.3.4 Artificial Neural Network

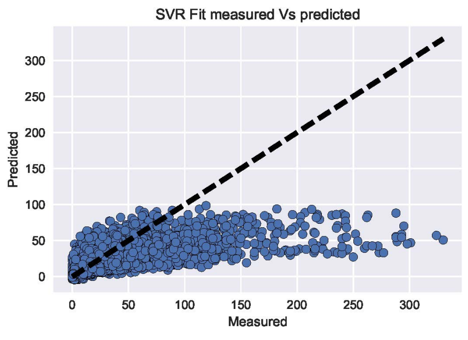
Theory
An artificial neural network is a network of simple elements called neurons that can approximate non-linear behaviour. It comprises an input layer, an output layer and many hidden layers. The input layer, consists of a set of neurons $x_1, x_2, ..., x_m$ representing the input features. Then, the neurons in the hidden layers, receives inputs from neurons in previous layer, and transform these inputs using a weighted linear summation $w_1x_1 + w_2x_2 + ... + w_mx_m$ followed by an activation function $g(\cdot) : R \rightarrow R$, and thus an neuron output is generated depending on the inputs, weights, and the activation. The weights here refer to the weights



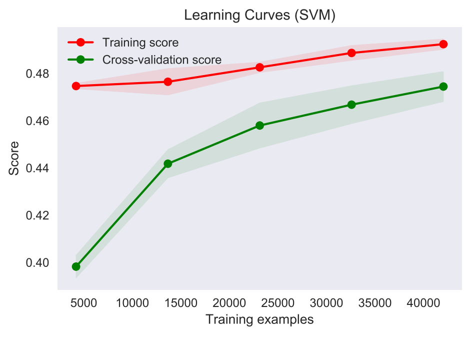
(a) Car



(b) Car



(c) Bike



(d) Bike

Figure 6: SVM Fit and Learning Curve for car and bike data

associated with each connection, and the choice of activation function is responsible for imparting non-linearity to the function. These neuron outputs become the input of other neurons in the next successive layer, thus forming a directed and weighted graph. The final output layer receives the values from the last hidden layer and transforms them into output values. All the connections have weights associated with them. The weights and the activation functions can be modified by a process called learning, which is governed by a learning rule. For regression purposes, the objective function is defined below and it involves an average sum-of-squares error term, while for classification purposes, the objective function involves a Cross-Entropy loss function term. The regression objective function is below. Further, for regression purposes, the ANN has no activation function in the output layer, which can also be seen as using the identity function as activation function in the outer layer.

For a given a set of m data points,$\{(x_1, y_1), \ldots, (x_m, y_m)\}$, we have to minimize the following objective function :

$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\left\|h_{W,b}(x^{(i)}) - y^{(i)}\right\|^2\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2$$

The first term in the definition of $J(W,b)$ is an average sum-of-squares error term $(lossfunction)$. The second term is a regularization term (R) (also called a weight decay term) that tends to decrease the magnitude of the weights and helps prevent overfitting. The solving of this objective function involves finding out the associated weights. The method involves stochastic gradient descent and the gradients are calculated using the back-propagation technique, and the weights are updated accordingly as shown below : $w \leftarrow w - \eta(\alpha\frac{\partial R(w)}{\partial w} + \frac{\partial Loss}{\partial w}$

where η is the learning rate which controls the step-size in the parameter space search.

The performance of an ANN model (including overfitting/underfitting) can be controlled by choosing the right hyper-parameters. These parameters are typically - number of hidden layers, number of neurons in each hidden layers, activation functions (logistic, RELU, tanh), learning rate, etc. Hence, we select the parameters over a grid-search space.

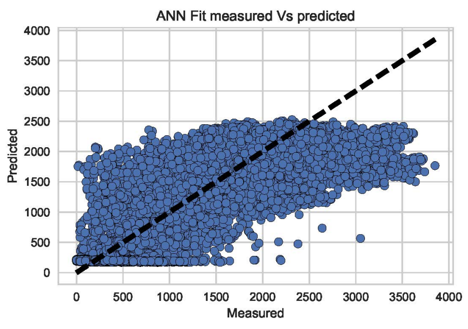
Parameter selection for best ANN model

For Artificial Neural Network, the following parameter space was evaluated in order to identify the parameters that will give the best Artificial Neural Network model : two different activation functions for the hidden layer (logistic sigmoid function to rectified linear unit function[RELU]); the L2 penalty (regularization term) parameter α varied from 0.1 to 10 ; and the ANN architecture with number of layers (varied from 2 to 5) and varied the number of neurons in the hidden layer (from 25 to 35). For both, the bike and car dataset, the best ANN model was obtained with the following parameters : RELU activation, α= 0.1, 5 number of layers, size of each hidden layer.

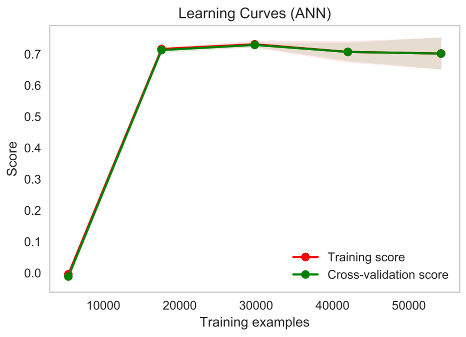
Performance of best ANN model

For both car and bike data, Figure7 shows the learning curve obtained by applying the artificial neural network (ANN) model, and the predicted ANN fit vs measured observation for the test data. From learning curve plot for the car data (Figure 7), it can be seen that ANN has final validation and training scores in range of 0.7 and both the validation and training score converge to the same value. This suggests that there is no over-fitting, and perhaps to improve accuracy, we can add one or more layer to the architecture to make the model more complex and reduce bias related error. Performance of ANN is not as good as random forest (around 0.95) or SVM (around 0.85 score) or linear regression with degree 2 polynomial (0.8 score). For test data, a wider deviation in ANN prediction and observed traffic count is seen (in Figure 7) and the corresponding performance error in terms of mean absolute error (MAE) is around 205 daily trip errors, which is worse than that predicted by Random forest (39) , Gradient boosting (43) and SVM (83) but slightly better than linear regression (MAE of 222). So for the car traffic count prediction, the random forest is performing better than SVM, ANN and linear regression.

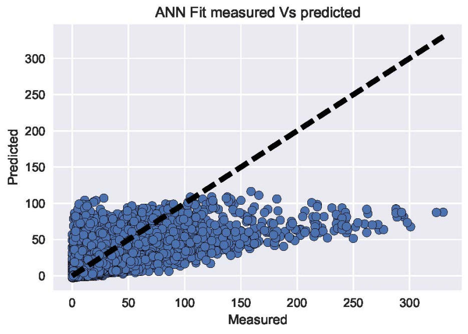
From learning curve plot for the bike data (Figure 7) , it can be seen that ANN has final validation and training scores in range of 0.42-0.43. The trend of all models tested so far giving lower accuracies for the bike data continue. The performance of ANN is comparable to SVM for bike data but it is not as good as random forest (score around 0.62) but is better than linear regression with degree 2 polynomial (which gives 0.35 score). For bike test data, the ANN predictions are showing much wider diversions from the observed bike traffic count (as seen in Figure 7) and the corresponding performance error in terms of mean absolute error (MAE) is around 3.82 daily trips, which is worse than that predicted by Random forest (MAE=2.04), SVM (MAE=2.09) but much better than linear regression (MAE=5.4). So considering the cross-validation accuracy and generalization ability, the random forest is performing better than SVM,ANN and linear regression.



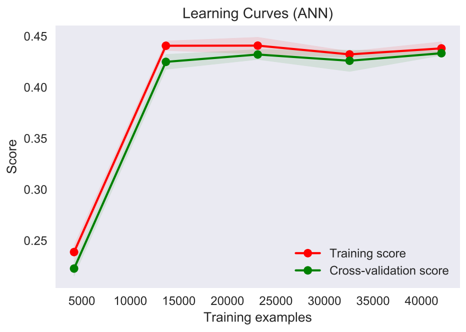
(a) Car



(b) Car



(c) Bike



(d) Bike

Figure 7: ANN Fit and Learning Curve for car and bike data

2.3.5 Model selection results

Based on the cross-validation performances of six different models, the Random forest model is able to provide the best accuracy. Hence, this model is chosen for predicting the traffic counts and its relationship with different variables. The next section discusses the most important variables influencing traffic counts and the differences in sensitivity of car and bike counts with regards to the weather related independent variables.

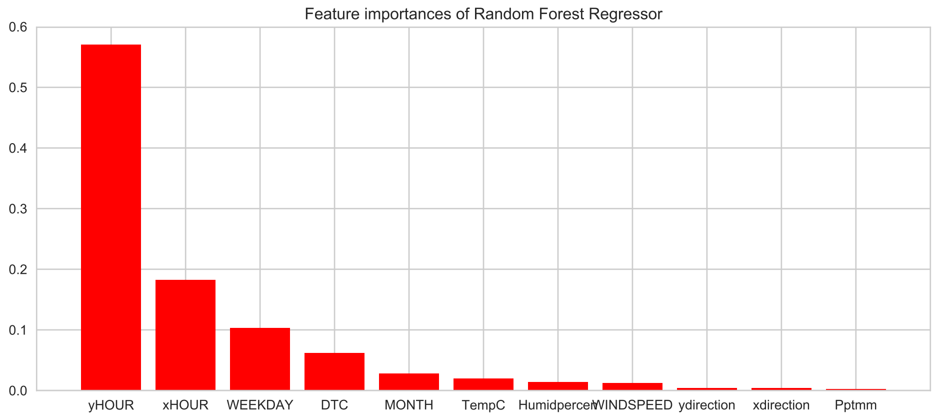
3 Results and Discussions

3.1 Differences in most important variables - Car and Bike count

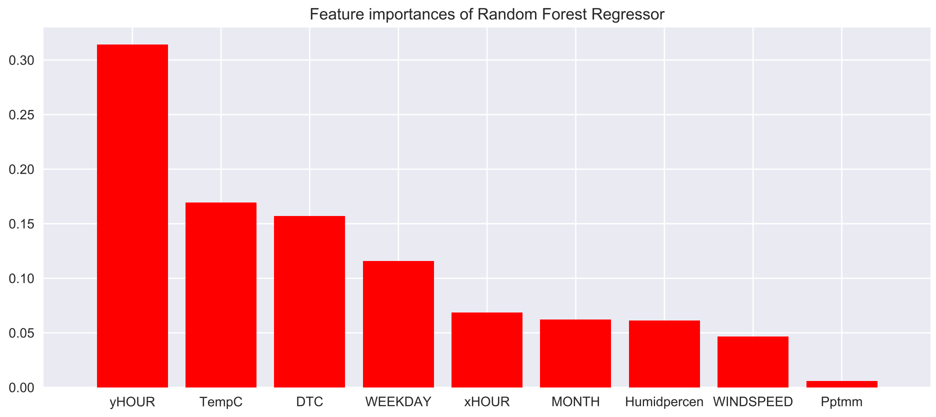
Figure 8 shows the most important variables influencing the traffic counts of car and bike. The car traffic count is influenced dominantly by variables like *hour of the day* (70% relative influence for combined xhour and yhour variables amongst other variables), followed by weekday (10 % relative influence), followed by distance of location from centre (5 % relative influence), followed by month (2-3 % relative influence) but not so much by the weather related variables (temperature, humidity, windspeed, winddirection, rainfall). The weather related variables, all combined together in total has approximately 10% relative influence, with temperature being the dominant ones among them (with around 2% influence). We had seen earlier that temperature is mildly correlated with the month of the year. Considering all this, the pattern suggests that weather is not as highly influencing factor in Oslo city for car drivers, as compared to the bike riders. The bike count, in contrast to the car count, is influenced more by weather (upto 30% relative influence amongst all variables, with temperature influencing to the extent of 17%, followed by humidity 7% and wind speed 4%). The most dominant influence even on bike count remains the $hour of the day$ with closer to 40% relative influence (combined xhour and yhour), but this influence is significantly lesser than the influence of the same variable on car count (70% relative influence as mentioned earlier). This pattern makes good sense as a bike rider is more likely to be exposed to weather than the car driver. Machine learning tools has helped us to quantify this pattern. Next, we analyze the influence of each of these variables on the observed counts one by one.

3.2 Variation with hour of the day and weekday - Car and Bike count

Figure 9 shows the variation in car and bike count with *hour of the day* for different weekdays. Some commonalities as well as differences are seen in the variation of bike and car count. The trends of variation with the *hour of the day* can be categorized into weekday trend and weekend trend. On weekdays (from Monday to Friday), for both car and bikes, the highest traffic counts are seen during the morning times (6-9 AM) and during the evening times (around 3-4 PM for car count and around 4-5 PM for bike counts). The early peak-time for car counts could be because of its usage in picking up school going kids on the weekdays. The important difference is that for the bike count, the observed bimodal distribution is very strong while it is milder for the car count. The pattern suggests that most of bike users are commuting mostly during these bimodal peak times, most probably office-going and home-coming trips. But for the car users, though they are also mostly travelling in these bimodal peak times but a substantial car count is also seen in non-peak times between 10 AM to 2 PM suggesting a more wider purposes for the car usage. Due to this, the bimodal distribution for cars is not that strong during weekdays. During weekends, the trends are similar for both car and bikes with less usage in the mornings and usage picking up to reach the peak at around 1230 PM to 1530 PM.

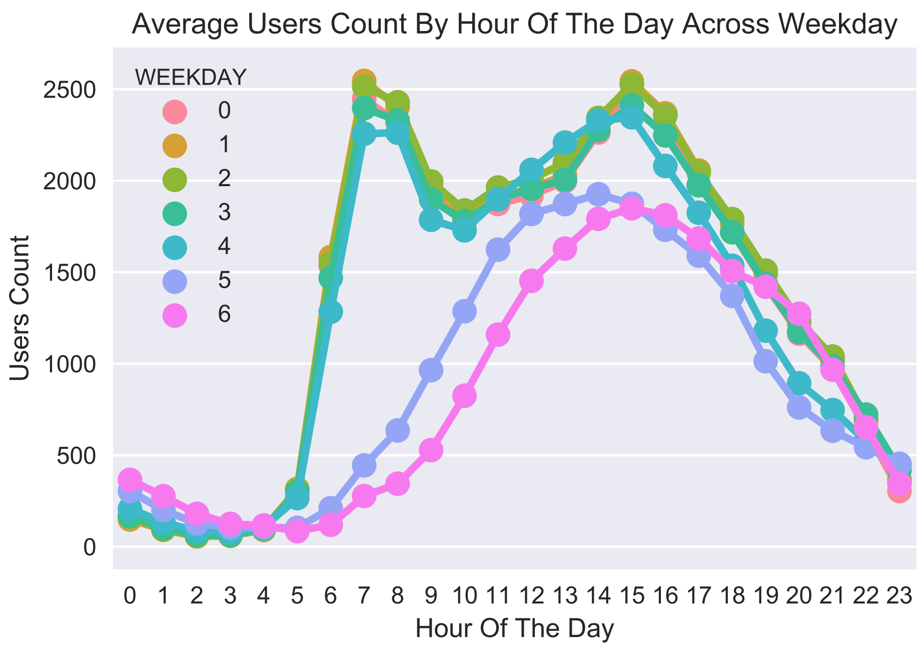


(a) Car

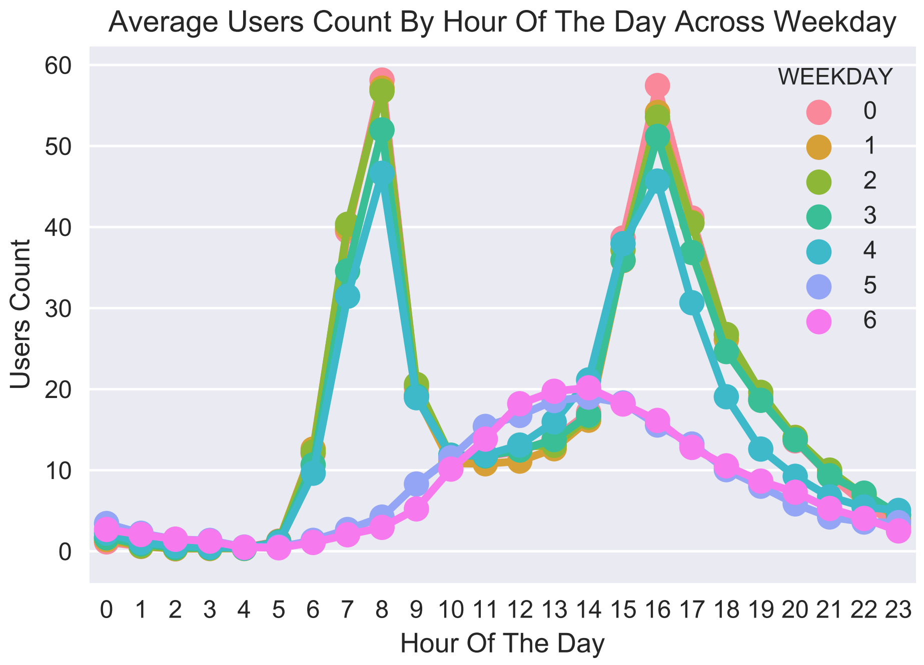


(b) Bike

Figure 8: Importance



(a) Car

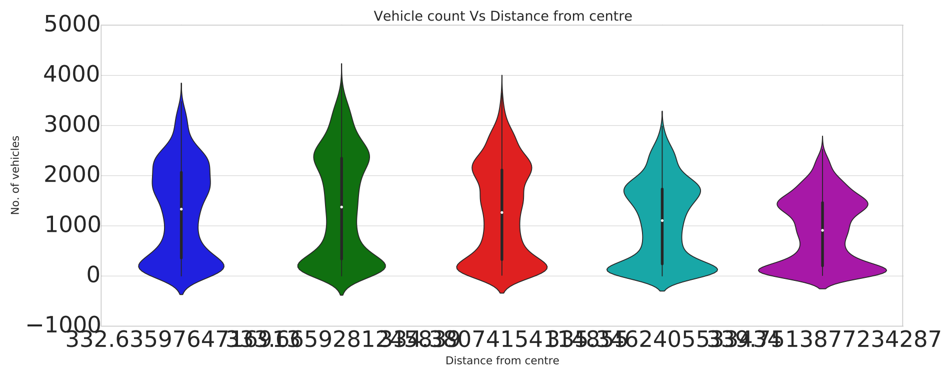


(b) Bike

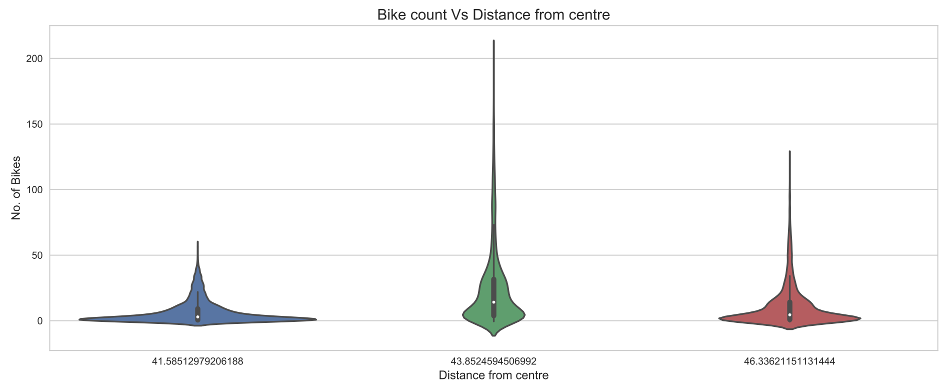
Figure 9: Weekday and hour Influence

3.3 Variation with distance from centre - Car and Bike count

Figure 10 shows the variation of car and bike counts with the location of counting stations relative to the city centre (i.e. measured as distance from the centre (DTC)). For the car data with five station locations, a drop in traffic count is seen as one moves away from the city centre. For the bike count with only 3 stations, too nearer to the city (DTC=41.58), the bike count is less, then it increases as DTC increases too 43.58 and drops down again as we are too far from the centre (DTC=46.3). This pattern could be reflective of both the population density as well as traffic rules and parking lots availability, which changes with distance from city centre.



(a) Car

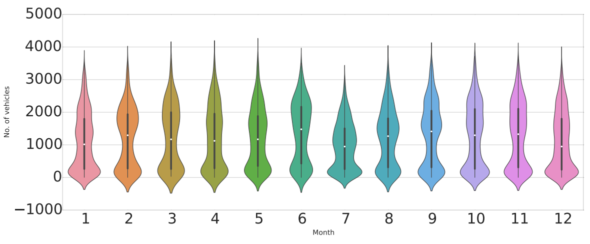


(b) Bike

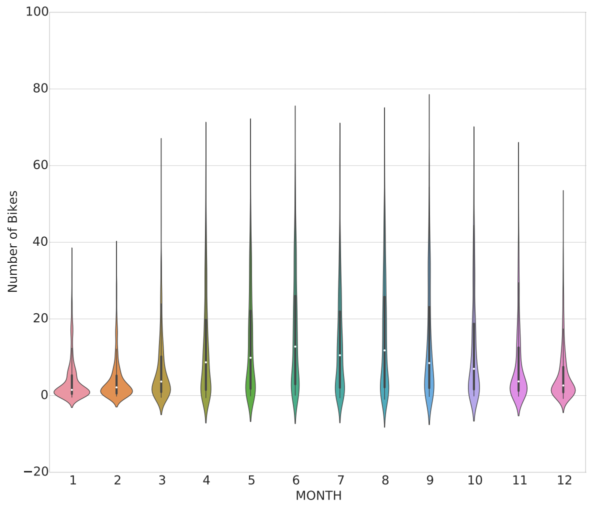
Figure 10: Distance from centre influence

3.4 Variation with month - Car and Bike count

Figure 11 shows the variation of car and bike counts with month of the year. For the bike counts, there is higher increase in bike count during summer season than during winter season. As was seen in the correlation plot that month and temperatures are mildly related, so this could be explained by favourable temperatures for biking during the summer time. People tend to bike in summer season both for office commuting and as a leisure activity. For cars, the count does not seem to be influenced much by seasons (and associated weather) but more by periods of vacations. The lowest car counts are seen during months of July and December. These two months are mostly the vacation periods, which will see drop of office commuting travellers. The car-counts are fairly constant during the months of February to May, and August to November with a mild high seen in June month.



(a) Car



(b) Bike

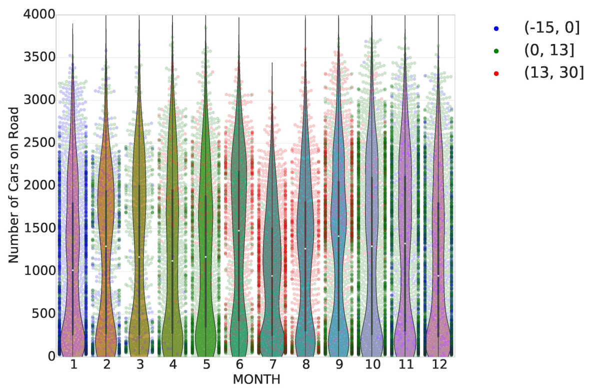
Figure 11: Month Influence

3.5 Variation with Temperature and Month - Car and Bike count

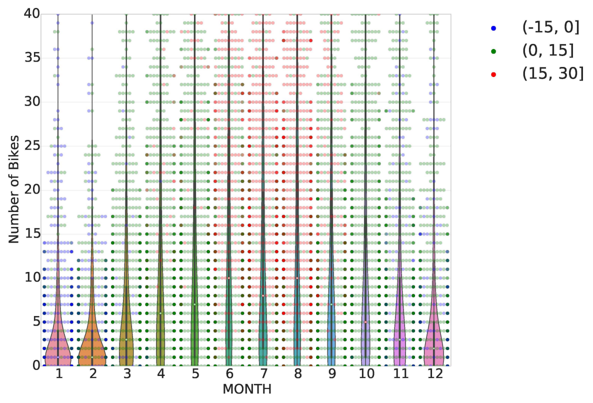
Since we had seen a mild correlation between month and temperature, so we have re-plotted violin plots of variation of bike counts with month and overlaid it with scatter-plot with each data point colored by temperature level hues (as shown in Figure 12). As noted previously in variation of bike count with month, the car count does not seem to change vastly as a result of seasonal temperature variation, but the bike counts seems to do so. A higher bike count is seen during the summer times (higher temperature period) than in the winter months (lower temperature period). The bikers will feel thermally more comfortable to bike during the summer time than during the winter time.

3.6 Variation with Humidity - Car and Bike count

Figure 13 shows the variation of car and bike counts with humidity percentage level. Bike count seems to decrease with increasing humidity percentage levels. This pattern could be



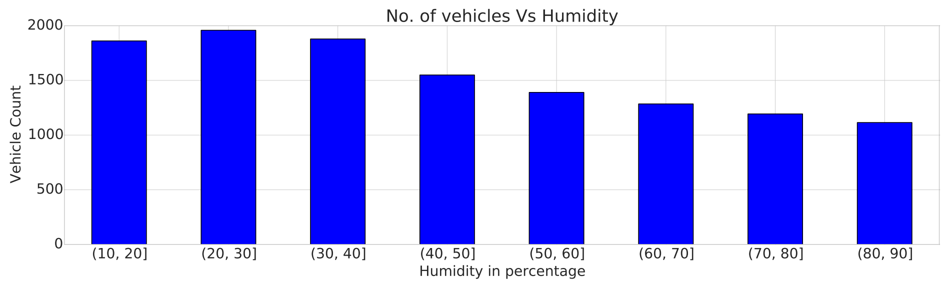
(a) Car



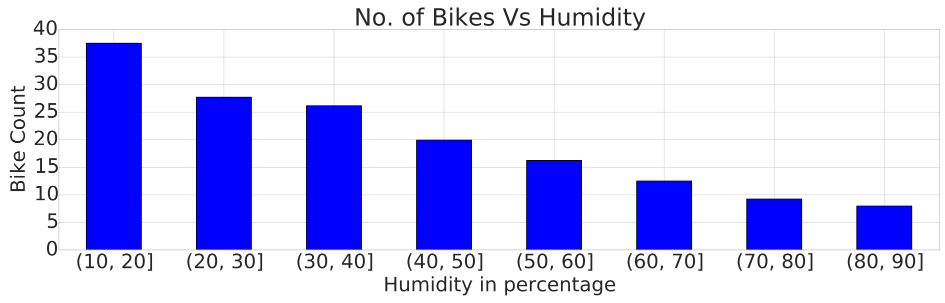
(b) Bike

Figure 12: Month Influence

result of bikers feeling uncomfortable due to sweat while biking at high humidity. The car count is also lowest at higher humidity level of 90-100% (signifying rainy conditions influencing car trip plans) but the decrease in car count with humidity levels is not as strong as for the bike counts.



(a) Car



(b) Bike

Figure 13: Humidity Influence

4 Conclusion and future work

Following conclusions can be drawn from this analysis :

1. Amongst the six machine learning models compared, the random forest model has been able to provide the highest accuracy as compared to the other five (linear regression, decision tree, gradient boosting, support vector machine and artificial neutral network).

2. Bike traffic count is more influenced by the weather related variables than the car traffic count, while the car traffic count seems to be highly influenced by the *hour of the day*. Bike count seems to be varying strongly with humidity and temperature, while the car count does not show any strong variation with them. This pattern is expected as bikers are more directly exposed to the weather than the car drivers.

3. For the bike count variation on different weekdays at different *hour of the day*, a strong bimodal distribution is seen with the highest traffic counts during morning times (6-9 AM) and during evening times (around 4 PM). These bimodal peak times most probably represents the office-going and home-coming bike trips. But for the car users, a less strong bimodal distribution exists suggesting a wider-purpose for the use of cars than the usual office-home routine.

4. Variation of bike counts with different months is influenced more by the weather associated with the month (seasonal variations), while the monthly car counts are influenced by the vacation periods in the month.

5. For the observed variation of counts with location of station - in regions nearer to and far away from the city centre, both the bike and car counts are lower. The counts are at a peak at the stations in-between. This seems to be representative of traffic rules and parking availability in the city vicinity and population density away from the city.

There is scope to take this work further. The future work could involve developing the predictor to guide municipal planners on road/bike-route planning and maintenance by giving advice on which routes are most preferred during a given time. The predictor can also be developed to encourage a cyclist to go biking often by giving alerts on possibility of conducive biking circumstances for end-to-end destination travel. But for doing this, we will need to create a larger biking traffic database. Also, there is scope for better car to cycle traffic comparison and analysis if the future traffic data collection focusses on simultaneous collection of cycle and bike data at similar locations (i.e. at bike lane location being located besides the road with both traffic moving in same direction). For bigger dataset, more advanced machine learning methodologies like deep learning can be employed to provide more accurate predictions.

5 Acknowledgment

We would like to thank the personnel from MET department for providing the data for this study, especially Anne Solveig for the valuable data collection part and both Eirik Johan Førland and Hans Olav Hygen for coordinating the data collection. The traffic data collection is attributed to CLIMAMOB project members, especially to Gretar Ævarsson for passing on the data. The authors acknowledge the financial support from the Norwegian Research council's CLIMAMOB and BIKE-TO-ZERO projects, which are both funded by the Research Council of Norway through the Climate Research Programme KLIMAFORSK (project no. 244137) and Energy Research Programme ENERGI-X (project no. 255628).

6 APPENDIX

The car and cycle volume data which have been used in this project are extracted from the dataset created for the following project run by the Institute of Transport economics (TØI): Project number 4334 – Efficient and climate friendly urban transport systems for the future.

6.1 Car traffic counts

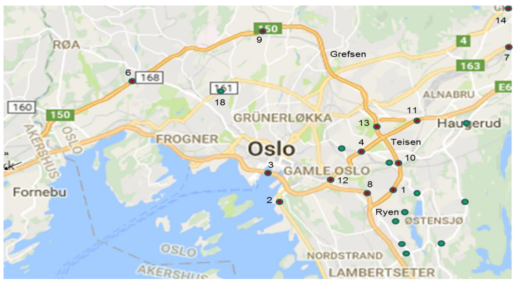
Data on car traffic volumes and bicycle traffic volumes was provided by The Norwegian Public Roads Administration, The Norwegian Public Roads Administration, Eastern Region and Municipality of Oslo (see figure 14). In most cases, data was collected from Level 1 counting stations . Level 1 stations continuously register the traffic through the year. Additionally, certain extra counting stations were set up to record the traffic data serving the purposes of Project 4334. We used the cohesive dataset collected over a period of 1 year in this project. Out of this only five stations were selected for analysis due to the quality of data (no missing data). These five stations are : R3 Grefsen, R3 Taysentunnelen, R3 Nydalsbrua, R3 Ã˜kern-Sinsen, R3 Smestad brannstasjon (Smestad V).

6.2 Cycle traffic counts

The Norwegian Public Roads Administration, Eastern Region and Municipality of Oslo have assisted in collecting data on the cycle traffic volume (figure 15-16). Data was, wherever possible, gathered from established counting stations which collected data throughout the year. There are relatively few cycle counting stations in Oslo which deliver data for an entire year's period. In the following table, all the counting stations are presented, and the stations with available yearly count which were further used in our analyses have been highlighted. For the cycle data, the following four stations were selected as they did not have any missing data : Maridalsvn, Ekebergvn, Akerselva and Ullevalsvn 19.

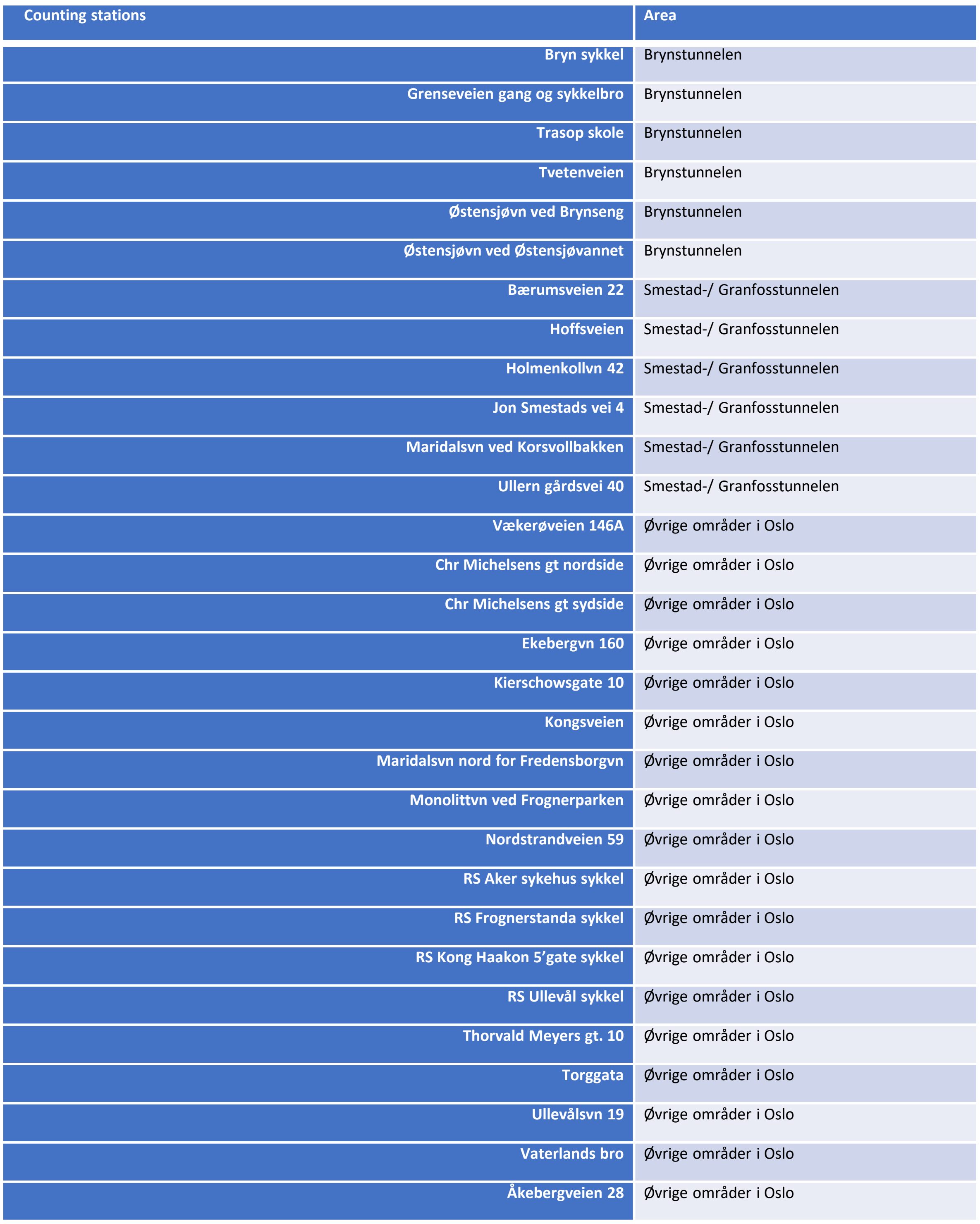
Nr.	Name of the counting station	Counting station nr.	Owner	Nr.	Name of the counting station	Counting station nr.	Owne r
1	E6 Manglerud	300001	SVRØ	15	Hellerudveien	300350	BYM
2	E18 Kongshavn	300018 300141	SVRØ	16	General Ruges vei	305870	BYM
3	E18 Bjørvikatunnelen	300029	SVRØ	17	Ensjøveien	306540	BYM
4	E6 Helsfyr	300030	SVRØ	18	Ring 2 Marienlyst	306578	BYM
5	E6 Skullerud	300039	SVRØ	19	Enebakkveien	306630	BYM
6	Ring 3 Smestad brannstasjon	300083	SVRØ	20	Lambertseterveien	306632	BYM
7	Rv 163 Grorud stasjon	300086	SVRØ	21	Plogveien	306633	BYM
8	E6 Svartdalstunnelen	300098	SVRØ	22	Tvetenveien v/Haugerud	306634	BYM
9	Ring 3 Tåsentunnelen	300099	SVRØ	23	Vekterveien	306635	BYM
10	E6 Bryn	300142	SVRØ	24	Østensjøveien v/Brynseng	306635	BYM
12	E6 Lodalen	300159	SVRØ	25	Rv 23 Oslofjordtunnelen	200244	SVRØ
13	Rv 150 Hovin	300165	SVRØ	26	E18 Ramstadsletta	200804	SVRØ
14	Rv 4 Ammerud	300231	SVRØ	27	E16 Brovoll	209570	SVRØ

(a) Car counting stations list



(b) Car counting stations location on the map

Figure 14: Car data counting stations and map of area covered



Counting stations	Area
Bryn sykkel	Brynstunnelen
Grenseveien gang og sykkelbro	Brynstunnelen
Trasop skole	Brynstunnelen
Tvetenveien	Brynstunnelen
Østensjøvn ved Brynseng	Brynstunnelen
Østensjøvn ved Østensjøvannet	Brynstunnelen
Bærumsveien 22	Smestad-/ Granfosstunnelen
Hoffsveien	Smestad-/ Granfosstunnelen
Holmenkollvn 42	Smestad-/ Granfosstunnelen
Jon Smestads vei 4	Smestad-/ Granfosstunnelen
Maridalsvn ved Korsvollbakken	Smestad-/ Granfosstunnelen
Ullern gårdsvei 40	Smestad-/ Granfosstunnelen
Vækerøveien 146A	Øvrige områder i Oslo
Chr Michelsens gt nordside	Øvrige områder i Oslo
Chr Michelsens gt sydside	Øvrige områder i Oslo
Ekebergvn 160	Øvrige områder i Oslo
Kierschowsgate 10	Øvrige områder i Oslo
Kongsveien	Øvrige områder i Oslo
Maridalsvn nord for Fredensborgvn	Øvrige områder i Oslo
Monolittvn ved Frognerparken	Øvrige områder i Oslo
Nordstrandveien 59	Øvrige områder i Oslo
RS Aker sykehus sykkel	Øvrige områder i Oslo
RS Frognerstanda sykkel	Øvrige områder i Oslo
RS Kong Haakon 5'gate sykkel	Øvrige områder i Oslo
RS Ullevål sykkel	Øvrige områder i Oslo
Thorvald Meyers gt. 10	Øvrige områder i Oslo
Torggata	Øvrige områder i Oslo
Ullevålsvn 19	Øvrige områder i Oslo
Vaterlands bro	Øvrige områder i Oslo
Åkebergveien 28	Øvrige områder i Oslo

Figure 15: Bike data counting stations list

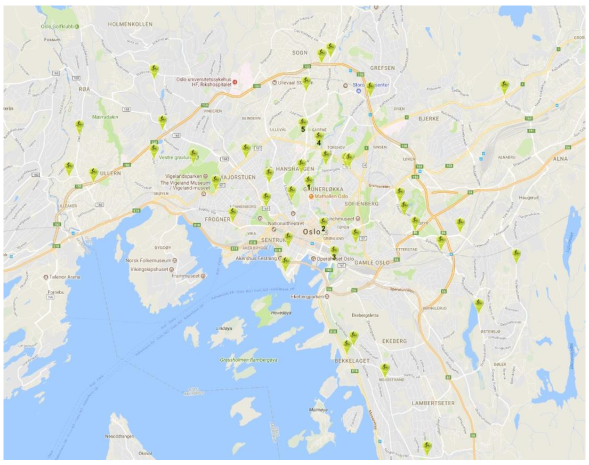


Figure 16: Bike data counting stations and map of area covered