Spring 5-18-2015

Driver Telematics Analysis

Karthik Vakati
San Jose State University

Driver Telematics Analysis

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment of the

Requirements for the

Degree of Master of Computer Science

By

Karthik Vakati

Spring 2015

Driver Telematics Analysis

by

Karthik Vakati

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

_____

Dr. Robert Chun, Department of Computer Science

_____

Dr. T Y Lin, Department of Computer Science

_____

Dr. Chris Tseng, Department of Computer Science

ABSTRACT

For automobile insurance firms, telemetric analysis represents a valuable and growing way to identify the risk associated with each driver. The pricing decisions of an insurer are best accounted for if they are made considering the driver's behavior instead of just the vehicle characteristics and the best way to understand a driver's behavior is to leverage the telemetric analysis. Decisions made on such factors can eventually lead to increased premium or reduced liability for unsafe or reckless drivers and can also help in transitioning the burden to the policies that lead to increased liability.

The dataset provided for this project by AXA has 50000 trips from anonymized drivers. A small and varied number of false trips (trips that do not belong to the particular driver of interest) are accounted for each driver. These trips are picked from drivers not given in the dataset, in order to avoid similarity with the given drivers. Neither the number of such false trips, nor the trips with a labeled set of true positives are given. However, it is given that most of the trips do belong to the particular driver of interest. The goal of this project is to develop a driving signature of each driver and identify the trips which are not driven by the particular driver of interest by predicting a probability for each trip that is accounted for the driver.

ACKNOWLEDGEMENT

I would like to dedicate my sincere thanks to Dr. Robert Chun for accepting my project and providing constant support and technical guidance for this writing project. I would also like to thank my committee members Dr. T Y Lin and Dr. Chris Tseng for their valuable time.

TABLE OF CONTENTS

1. INTRODUCTION

Driving, in essence, is a multi-factor cognitive task that is perceived only in the context of its surrounding environment like the underlying traffic conditions, road layout, social behavior and weather conditions. Therefore the driving style of an individual driver is majorly influenced by the actions of motorcycle riders and other drivers that share the environment: the driver first understands the surrounding environment which includes the behavior of other vehicles sharing the environment like the distance from the leading vehicle, traffic conditions and road layout and utilizes this understanding in decision making about braking, steering and acceleration. As the driver gains experience, he develops a unique individual driving style that could impact road congestion, fuel economy, safety, and other factors. For instance, a driver may maintain a comfortable distance and time gap to the vehicle in front by monitoring his/her own behavior based on surrounding environment.

Being able to analyze and recognize dynamically, the driving style doesn't yield any valuable information for automobile insurers. More precisely, being able to analyze the driving style coupled with the contextual behavior of its surrounding environment, allows the automobile insurers to assess the risk associated with each driver. For instance, if the acceleration patterns of the drivers in specific road curvatures can be categorized taking into account the vehicle dynamics and compared with the speed limit and accident statistics of the particular road segment under study, confident decisions can be made which can eventually lead to increased premium or reduced liability for unsafe or reckless

drivers and can also help in transitioning the burden to the policies that lead to increased liability.

2. RELATED WORK

The recent advancements in data mining and machine learning algorithms can be put to the best use to explore huge time-series data so as to discover underlying patterns and establish spatio-temporal associations among them. Also, the recent advancements in hardware/software help automotive on-board diagnostic systems and communications to capture the real time information about drivers and their styles.

Some studies examined the relationship between the aggressive behavior of the driver and road characteristics like number of lanes, speed limit, presence of curbs and the type of road. Aggressive behaviors include lane violations, failure to stop, speeding, sudden raise of acceleration and severe other violations. Driving styles, most of the times, can also vary depending on weather/road conditions and the driving culture of the particular country where the data is recorded from. In such cases, the application of data mining to discover patterns and classify driving styles is highly preferable.

Some studies modeled the drivers' speed profiles at a curvature with average angular velocity and the particular location of maximum acceleration calculated based on Bayesian methods. It is revealed that speed profiles of different drivers differ significantly, just like the average speeds at the curves. The modeled speed profiles of the drivers can also be used to derive acceleration profiles, which in turn can be used to determine the aggressive behaviors of the drivers. The work of Spiegel et al. [6], utilizes a bottom-up Singular Valued Decomposition algorithm that identifies, time series segments which are internally homogeneous. The recurring patterns are recognized, by grouping the time series segments using agglomerative hierarchical clustering. Consequently, recurring series

of grouped segments that can be considered as classes containing high-level driving contexts can thus be retrieved.

Some studies introduced two models which predict the recognition of driving events: the first model used multiple linear regression approach to predict whether a driver steers or eases up on the brake or accelerator; the second model used Bayesian Network to predict drivers' decelerating patterns. The predictions on three driving actions made by the two proposed models were successful with the accuracy over 70%.

Although majority of the studies dealt the prediction of driving style with the classification approach, dealing the problem with a regression approach is treated very useful. The reason is that it provides valuable insights into the recognition problem by providing an estimation of every factor that contributes to the event that needs to be recognized.

3. PROJECT DESCRIPTION

The dataset provided for this project by AXA has nearly 50000 trips from anonymized drivers. These trips are divided among 2736 drivers with each driver accounting for 200 trips of which majority of them have actually taken by the driver. The trips are provided in the form of .csv files.

Each trip is recorded by capturing every second, the position of the car (in meters). To protect the drivers' privacy of their locations, the trips were randomly rotated, centered to initiate at the origin (0, 0) and data accounting for short trip lengths were removed from the trips' start/end positions.

A small and varied number of false trips (trips that do not belong to the particular driver of interest) are accounted for each driver. These trips are picked from drivers not given in the dataset, in order to avoid similarity with the given drivers. Neither the number of such false trips, nor the trips with a labeled set of true positives are given. However, it is given that most of the trips do belong to the particular driver of interest.

The intent of this project is to develop a driving signature to answer the following questions about each driver. Does he drive short trips? Does he drive long trips? Does he take Highways? Does he take back roads? Does he accelerate hard and sudden from stops? At what speed does he take turns? The answers to the above questions can be combined to form an aggregate and unique driving profile that differs from other drivers and eventually help to identify the trips which are not driven by the particular driver of interest by predicting a probability for each trip that is accounted for the driver.

A sample trip looks like the following:

x	y
0	0
18.6	-11.1
36.1	-21.9
53.7	-32.6
70.1	-42.8
86.5	-52.6
101.7	-62.3
117	-71.6
131.2	-80.4
145.5	-88.7
159.7	-96.8
171.7	-104
182.5	-111
193.4	-117.7
202	-124
211.8	-130.4
221.5	-137
231.3	-143.6
239.8	-150.3
250.6	-157.1
260.4	-163.8
270.1	-170.8
279.8	-177.8
289.5	-185
302.6	-193.1
314.5	-201.2
325.2	-209.4
337.2	-218.7
347.8	-228.1
357.5	-237.1
367	-246
376.6	-255.1
386.2	-264.4
395.7	-273.8
404.1	-282.9
410.3	-291.6
416.6	-299.8
421.8	-306.9
422.5	-313
425.4	-319
428.4	-324.8
430.3	-330.8
431	-337
432.8	-344.4
433.3	-353.1
436	-363.1
.	
.	
.	
.	
-531	1021.1
-531	1021.1
-531.1	1020.9
-531.1	1020.9
-531.1	1020.9
-532.2	1020.9
-532.2	1020.9
-532.2	1020.9
-532.2	1020.8
-532.2	1020.8
-532.2	1020.8
-532.2	1020.7
-533.3	1020.7
-533.3	1020.7
-533.3	1020.7
-533.3	1020.6
-533.3	1020.6
-533.3	1020.6
-533.3	1020.5
-533.3	1020.5
-533.3	1020.5
-533.3	1020.5
-534.4	1020.6
-534.4	1020.5
-534.4	1020.5
-534.4	1020.5
-534.4	1020.5
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.4
-535.5	1020.3
-535.5	1020.3
-535.5	1020.3
-535.5	1020.3

Figure 1: Sample Input .csv file

The following are the trips driven by the driver (with id =1), as per the given dataset.

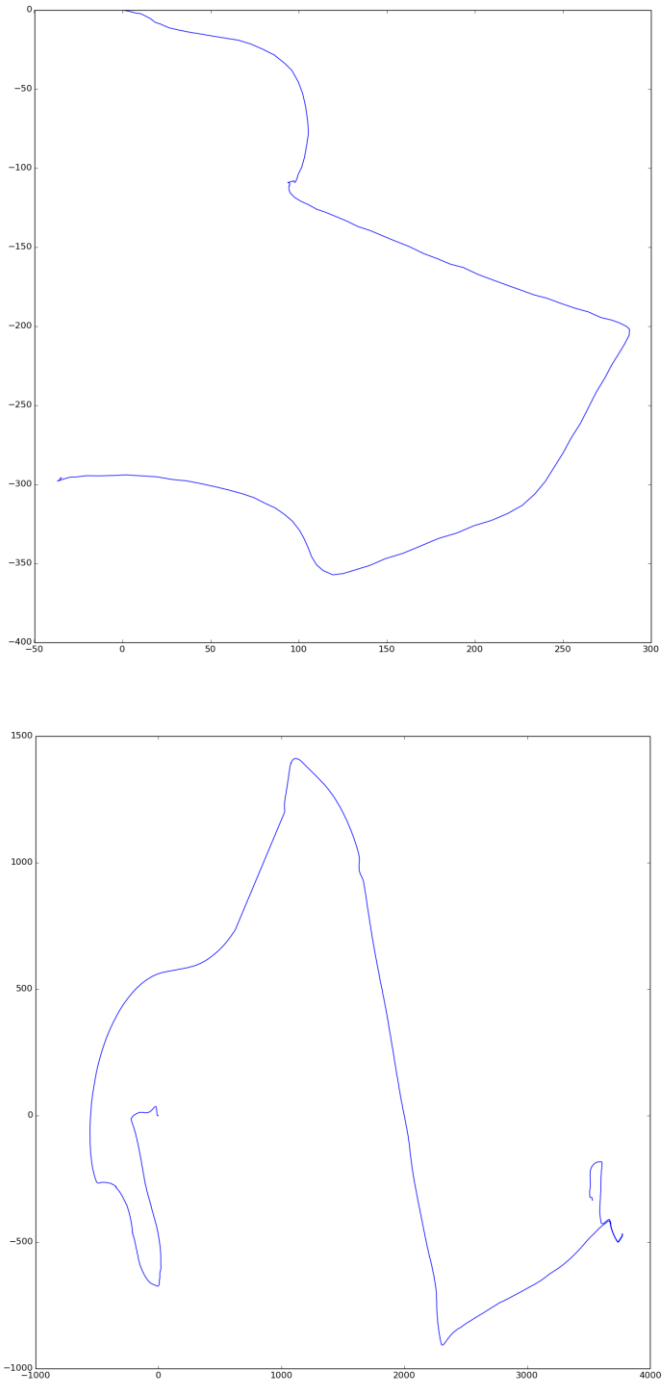

Figure 2: Plots showing the trips taken by the driver with id = 1

4. TOOLS AND PACKAGES

The following are the tools and packages used to accomplish the project.

4.1 PyDev with Eclipse: This project uses Eclipse 4.3 which is configured with PyDev plug-in to support the development of python modules. All the code for this project is developed using python with the python interpreter version of 2.7.

4.2 Packages: This project uses the following packages available in python to accomplish a variety of tasks.

4.2.1 SciPy: *SciPy* is a python library which is a collection of many open source software packages like NumPy, Matplotlib among many others to provide scientific computing in Python.

```
hduser@KarthikVakati:~$ python
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import scipy
>>> scipy.__version__
'0.14.0'
>>>
```

Figure 3a: Scipy Version Information

4.2.2 NumPy: *NumPy* is a fundamental package used for numerical computation and is one of the core packages available in SciPy. It provides methods to represent data in matrix types and numerical arrays and support methods to perform basic operations on them.

```
hduser@KarthikVakati:~$ python
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy
>>> numpy.__version__
'1.8.1'
>>>
```

Figure 3b: Numpy Version Information

4.2.3 Matplotlib: *Matplotlib* is one of the most popular plotting packages available as one of the core packages in SciPy. It provides 2D plotting in python with the quality worth of publication. It also provides rudimentary 3D plotting.

```
hduser@KarthikVakati:~$ python
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import matplotlib
>>> matplotlib.__version__
'1.3.1'
>>>
```

Figure 3c: Matplotlib Version Information

4.2.4 Multiprocessing: *Multiprocessing* is a python package which provides both remote and local concurrency by spawning processes. It uses an API which is similar to the "threading" module in python to achieve this.

```
hduser@KarthikVakati:~$ python
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import multiprocessing as mp
>>> mp.__version__
'0.70a1'
>>>
```

Figure 3d: Multiprocessing Version Information

4.2.5 Scikit-learn: *Scikit-learn* is an open source python library that features various machine learning algorithms of supervised learning (Classification and Regression) and unsupervised learning (Clustering).

```
hduser@KarthikVakati:~$ python
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import sklearn
>>> sklearn.__version__
'0.15.2'
>>>
```

Figure 3e: Scikit-Learn Version Information

5. MACHINE LEARNING ALGORITHMS

The following machine learning algorithms are used to accomplish this project.

5.1 Random Forest: Random Forests is an ensemble of many classification or regression trees. The class label of a new data point is obtained by passing the input vector down each classification or regression tree of the forest. A classification is obtained from each individual tree and a vote is obtained for that particular class. The forest finally chooses the class label which has the highest number of votes.

Each tree in the forest is generated as follows:

i. If N is the number of entrees in the training data, then a random sample with replacement is chosen from the initial training data and this small random sample serves as the training data for this tree.

ii. If M is the number of features or input attributes, a small number m (far less than M, m<<M) is chosen in such a way that at each individual node of the tree, m out of M features are chosen at random and the node is split based on the best value of m. m is treated as constant during the process of growing forest.

iii. Each tree of the forest is developed to the largest depth possible. No pruning is applied.

The error rate of the forest depends on two important things:

i. *Correlation:* If any two trees are correlated in the forest, then it results in the increase of forest error rate.

ii. *Strength:* Weak classifiers tend to increase the error rate. So as the trees become strong, error rate of the forest decreases.

Selecting a small m reduces both strength and correlation and large m increases both of them. Random Forest works at its best when an optimal value of m is determined. This is done by leveraging the out-of-bag error rate. This is the only parameter which is adjustable and to which random forests are sensitive to an extent.

5.2 Gradient Boosting: The basic idea of Gradient Boosting is to generate a series of simple classification trees, wherein the prediction residuals from the preceding tree are used to build the successive tree. Each tree that is built is a binary tree which partitions the data into two different samples at every node split. At every step of boosting, a best and simple partitioning of the input data is determined. Each partition results in a residue which helps in computing the standard deviations of input values. The next successive tree is grown to fit the residuals from the previous tree, to derive a new partition of the data that further reduces the error rate of the data.

It is proved that this process of building the trees with weights added in each step of boosting can eventually result in a best fit of the observed values to the predicted values. This is true even if there is complex or non-linear relationship between dependant and independent variables.

5.3 Linear Regression: In the field of statistics, **linear regression** is a methodology to model a relation between a variable *y*, which is a scalar and dependant on one or more independent variables denoted by *X*. Linear Regression is Simple if the dependant

variable depends only on one independent variable or Multiple (not Multivariate) if the dependant variable depends on more than one independent variable.

In this algorithm, the parameters of model are estimated by modeling the data with predictor functions which are linear. Models thus estimated are known as *linear models*. In majority of the cases, linear regression relies on *y*'s conditional mean which is an affine transformation of *X* and in some cases it relies on *y*'s conditional median or some measure which is a linear transformation of *X*. Similar to various kinds of regression analysis, the focus of *linear regression* is on the probability distribution of the dependant variable *y* given the independent vector *X* which is conditional, rather than on the combined probability distributions of dependant variable *y* and independent vector *X*, which forms the cause for multivariate regression analysis.

5.4 Support Vector Machines (SVM): SVM belongs to the class of classifiers that achieve classification by the construction of hyperplanes which separate data points of different categories or classes in a multidimensional space. SVM is capable of performing both classification and regression tasks and also handling multiple categorical and continuous variables. For categorical attributes, a variable (often dummy variable) is assigned with class label either 1 or 0. Thus, a dependent categorical variable depending on three attributes, (X, Y, Z), can be represented as X: {0 1 0}, Y: {0 0 1}, Z: {1 0 0}.

SVM utilizes an iterative approach in its training algorithm, which minimizes the error function in order to obtain a hyperplane which is optimal.

5.5 Ramer Douglas Peucker (RDP) Algorithm: The RDP algorithm reduces the number of points of a curve by approximating common sets of points. The algorithm works by iteratively or recursively drawing lines between current and last points of the curve. At each iterative step, it checks for the points that are farthest from the line drawn above. It deletes all these points from the resulting curve whose distance from the line is smaller than a threshold value "epsilon". If the distance of the points from the line is greater than the threshold value "epsilon", the curve is divided into two parts:

i. Starting from the first point and including these distant points
ii. The distant and rest of the points

6. SYSTEM ARCHITECTURE

The input data, as described above, is provided in .csv files. In order to apply any machine learning algorithm to this data, features need to be identified and extracted from this data. A feature is any measurable property of the phenomenon that is being observed. A group of features can represent the whole phenomenon as such. The process of deriving the values of such features is called Feature Extraction. Feature Extraction is intended to reduce any redundant information and require fewer resources to represent the whole set of data. The features of this project fall into various categories like the Movement features, Trip features and Segment features.

The data thus obtained by deriving the values for the features is divided into two sets of values. The first set of data is called the “train data” and the second set of data is called the “test data”. The train data serves as the actual input data to the machine learning algorithm. The algorithm analyses the input data for any repeated patterns and builds models that capture these patterns. The models built for this project made use of the machine learning algorithms like Random Forest, Gradient Boosting, Support Vector Machines and Linear Regression to study the driving behavior of all the drivers.

Once the models are built, the goal is to predict some phenomenon of interest from the test data. It does so by leveraging the models that are built on the train data. The goal of this project is to predict the probabilities of each trip that belong to the drivers.

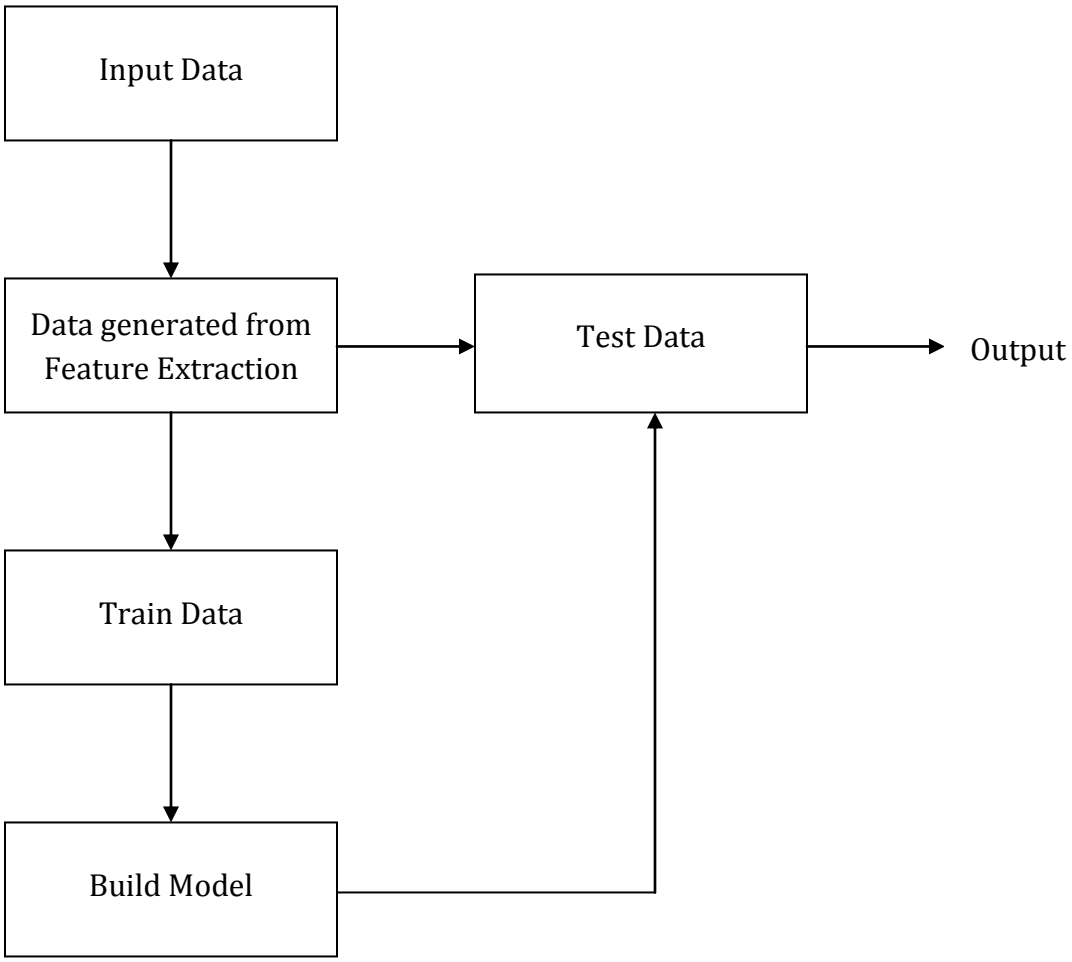

Figure 4: System Architecture

7. FEATURES

Features are any measurable properties that best describe or represent any phenomenon. Feature Extraction (the process of deriving values for the features) is one of the crucial steps to predict the behavior of interest about the phenomenon. For any analytics problem, if the extracted features can capture the exact and complete behavior of the phenomenon, then majority of the problem is solved. The rest of the problem lies on building an appropriate model out of these features.

For Driver Telematics problem, features need to be extracted that best describe the driving behavior of the driver and also the road conditions. The research about the driving behavior has started in 1978 when Karstens and Kuhler [7] introduced 10 behavior parameters that account for aggregate driving profile: mean speed, mean speed excluding the stops, mean acceleration, mean deceleration, average length of a trip, mean number of acceleration/deceleration changes within a trip, standstill time proportion, acceleration time proportion, deceleration time proportion and constant speed time proportion. Later on, many studies also revealed several other interesting features pertaining to the driving behavior. Although all such features are taken into consideration, in the context of this problem, some features were excluded and the resulting features formed the final feature set for building the model. These features fall into 3 categories.

7.1 Trip Features

These are the features that represent the properties of the trip. The features that fall under this category are as follows.

7.1.1 Ride Length: This gives the total length of the trip in meters. It is calculated as follows. The distance between every two consecutive points is calculated and summed up to give the final length of the trip. The distance measure used to calculate the distance between two points is the Euclidean distance measure.

7.1.2 Ride Speed: This is the average speed for the total trip. It is calculated by dividing the total length of ride (calculated from above) with the total trip time (calculated by summing up all the rows in the excel file except for the first row which are labels).

7.1.3 Ride Length without stops: This gives the total length of the trip for which there are no stops. In order to measure this, Euclidean distances are first measured between every two consecutive points and distances which are less than certain threshold like 2 meters (as they can be considered as stops) are removed. The resulting distances are summed up to give the total length of the trip without stops.

7.1.4 Ratio of Stops: This gives the ratio of total number of stops over length of the trip for a particular trip. It is calculated by dividing the total number of distances whose Euclidean distance measure is less than certain threshold like 2 meters with the length of the trip.

7.1.5 Angle: This gives the angle of the trip at a particular position with respect to the coordinate system. It is calculated using the standard geometric formulas.

7.1.6 Other Features: These features include percentiles and histograms over speeds, angles, accelerations, speed*angles.

7.2 Trip segment features

A good idea to identify matching trips is to use Euclidean distances. But a better idea to identify matching trips is to identify common road segments between trips. This approach should eventually identify redundant trips. The algorithm that is helpful in this regard is the Ramer-Douglas-Peucker algorithm which simplifies trajectories.

7.3 Driving features

These features can best describe the driving style of the driver, although to some extent they can also capture certain road features and junction shapes. The features that fall under this category are as follows.

7.3.1 Mean acceleration: This gives the mean acceleration for each acceleration window.

7.3.2 Mean deceleration: This gives the mean acceleration for each deceleration window.

7.3.3 Average number of acceleration/deceleration changes: This feature gives the average number of changes from acceleration to deceleration and vice-versa.

7.3.4 Other Features: Other features include standard deviations of average, minimum and maximum values of speed*acceleration and mean and standard deviation of acceleration and deceleration windows.

ride_length	ride_speed	ride_length_no_stops	stops_ratio	euclidian_distance	angles
5.399812144611642907e+03	9.764578923348359751e+00	5.381682314222243804e+03	2.278481012658227778e-01	4.351405228102872570e+03	5.372466157096698058e-01
4.608272466372889539e+03	1.807165673087407498e+01	4.606258180919899132e+03	7.843137254901960675e-03	2.518063623744566939e+03	3.904931466458646527e-01
3.341949342124683426e+03	5.894090550484450297e+00	3.293104779861437692e+03	2.980599647266313768e-01	1.773208393586655575e+03	9.223468471654350243e-01
1.441302686417888935e+03	2.854064725579978035e+00	1.401676245113044615e+03	6.217821782178217793e-01	4.909105878474928772e+02	2.302351840738207756e+00
3.187333205644308691e+03	7.535066680010186424e+00	3.165498799121940010e+03	3.711583924349881602e-01	1.664791785318428765e+03	6.289270208903304793e-01
2.014012299215853773e+03	5.164134100553471107e+00	1.953677050588731618e+03	4.974358974358974450e-01	1.445581203009871842e+03	1.511583099878919700e+00
3.062866110961160302e+03	6.992845002194429505e+00	3.030805975363965445e+03	4.018264840182648179e-01	1.260460075960190807e+03	8.548271330173676796e-01
7.834283392643784282e+03	5.611950854329358584e+00	7.699561676484045165e+03	3.216332378223495936e-01	5.416226469709254161e+03	5.359552452094999975e-01
1.462367957180873600e+04	1.325809571333520864e+01	1.455993432903629946e+04	1.867633726201269329e-01	1.169338070046144639e+04	6.550332286088141887e-01
1.170421047736383844e+04	1.153124184961954590e+01	1.166801079399949049e+04	1.330049261083743883e-01	9.764443934330214688e+03	3.021268027865034915e-01
1.979271085911885393e+03	7.761847395732884181e+00	1.968453797933602800e+03	2.549019607843137081e-01	1.419158265077531269e+03	5.992815512116311805e-01
1.143084890549911324e+04	1.423517920983700336e+01	1.140176090860234581e+04	1.095890410958904049e-01	1.025141690359869062e+04	4.145530739040047541e-01
2.120933222394660334e+03	6.219745520189448900e+00	2.111793960069020613e+03	1.847507331378299145e-01	1.061346400742762398e+03	7.335335575451116252e-01
7.102946462195164713e+03	1.066508477807081867e+01	7.080199175662810376e+03	1.306306306306306286e-01	5.450911190354903520e+03	3.133727315474877373e-01
6.464470595033350946e+03	9.520575250417305568e+00	6.432907693601628125e+03	1.531664212076583320e-01	5.444145871461436400e+03	5.242810496195031700e-01
1.918246443946970885e+03	3.341892759489496267e+00	1.863370564329137551e+03	6.010452961672473338e-01	1.121382938107028622e+02	1.195546715191847298e+00
5.853883294899191725e+01	3.383747569305891045e-01	5.370571718310135800e+00	9.768786127167630173e-01	7.294365389799800425e+00	4.736415193097155196e+00
9.032523148383647822e+02	4.032376405528414587e+00	8.782150966960875849e+02	5.625000000000000000e-01	1.938006995183114896e+02	7.877571571261501315e-01
8.551458081740187481e+03	1.134145634183048656e+01	8.517177070657122385e+03	1.697612732095490629e-01	5.549412901846082605e+03	5.656153011552447207e-01
8.416530361044240635e+03	8.906381334438350450e+00	8.354838319189948855e+03	2.190476190476190466e-01	5.593259397264777363e+03	3.542282793656978956e-01
2.087619770280929970e+03	6.250358593655478856e+00	2.082274107558520882e+03	1.736526946107784519e-01	1.044387143644855087e+03	6.150977250640143890e-01
1.184509243850606708e+03	4.914976115562683212e+00	1.170187118357021291e+03	3.651452282157676144e-01	4.163434870877205185e+02	7.234297954681299103e-01
1.495057543592417687e+03	6.229406431635073460e+00	1.478172969793084576e+03	4.000000000000000222e-01	1.036414476370851162e+03	7.544477668448499319e-01
3.689608707451288410e+03	7.499204689941643132e+00	3.660021463757037964e+03	3.353658536585366057e-01	1.675419579103420574e+03	8.896394003270216011e-01
1.600683767387760236e+04	1.852643249291389083e+01	1.597620526266369416e+04	8.796296296296296502e-02	1.429478815031502563e+04	2.074496331331905918e-01
2.855956935184035572e+01	5.912954317151212363e-02	0.000000000000000000e+00	9.979296066252587583e-01	2.999868666815364460e+00	6.562554203793430929e+01
9.003372499174416589e+03	1.064228427798394350e+01	8.973554537699770663e+03	1.595744680851063912e-01	5.551357672827015449e+03	4.456373005362527229e-01
8.185859220751199246e+03	9.463421064452253617e+00	8.108581027304549934e+03	2.219653179190751446e-01	5.636511792563710515e+03	4.880515069677018869e-01
2.696386974160041518e+03	5.836335441904851429e+00	2.662648636212013116e+03	3.160173160173160301e-01	1.717061949500961191e+03	8.116958188678700115e-01
9.769082630706108830e+03	8.599544569283546380e+00	9.712555040237099092e+03	2.411971830985915610e-01	5.586453378874358350e+03	3.837461066495583273e-01
1.207993831905533625e+03	6.565188690518129448e+00	1.188174894769681259e+03	3.369565217391304324e-01	4.073871665009899061e+02	5.732695442180415490e-01
6.980055450669924539e+03	9.509612330612975839e+00	6.957611185964442484e+03	1.934604904632152567e-01	5.401086007684348260e+03	3.942179454009354100e-01
9.661906267503068193e+02	3.818935283598050567e+00	9.256343416033444100e+02	5.296442687747036082e-01	3.303968090815153573e+02	2.435211110380452126e+00
5.897988950129180012e+03	8.135157172591972241e+00	5.859372766524073366e+03	2.165517241379310287e-01	2.397207604330430513e+03	5.820548167346549784e-01
8.463313347433082527e+03	9.169353572516882878e+00	8.390995435762488341e+03	3.228602383531961051e-01	5.591879094340798474e+03	4.917264413898684849e-01

Figure 5: Sample input feature file for Train data

8. PROJECT IMPLEMENTATION

This project is implemented as follows. For every driver, 180 out of 200 trips are chosen at random and labeled as "1" (probability that this trip was taken by the driver of interest). Then a driver is randomly selected and 180 from 200 trips of this driver are chosen at random and labeled as "0" (probability that this trip was **not** taken by the driver of interest). The features were then extracted for these trips and the data that is generated forms the Train data. The remaining 20 trips from both the drivers form the Test data.

To further improve the efficiency of the model that is built, some modifications to this execution plan were carried out. Instead of just taking 180 trips from a single driver, 180 trips from more than 4 drivers are chosen. Since this modification results in unbalanced training data, data from the driver of interest is duplicated.

For the purpose of cross validation, a slightly modified approach can be followed. In this approach, the process of model building (from 180 out of 200 trips from the current driver and 180 out of 200 trips from the other drivers) is repeated 10 times so that every trip falls into train and test datasets atleast once. This removes any factor of bias that was generated from the previous approach.

Once the Train data is generated, features of each category type are extracted. Individual models are built by running machine learning algorithms like the Random Forest Classifier, Gradient Boosting Classifier, Linear Regression and Support Vector Machine on each set of features.

8.1 Ensemble:

Ensembling is the process of joining individual machine learning models, with each individual model assigned a weight, in order to gain better prediction results. This project uses a linear model to combine all the individual models and proved to be successful.

8.2 Caching:

It is always a good idea to cache the results from the models that are built. Caching enables easier access to the models built earlier and save huge amounts of time by eliminating the need to rebuild the model. As the data provided is very huge, at times, caching definitely improves the efficiency of the project by reducing the build time of the project.

9. EVALUATION

The performance metric used to evaluate this project is the **area measured under ROC curve**. ROC is an acronym for Receiver Operating Characteristic. ROC curve is a graphical plot, which elucidates the performance measure of a binary classifier against its corresponding discrimination threshold. The curve is generated by plotting the rates of true positive against false positive at various settings of the threshold. (In machine learning, the true-positive rate is termed as "recall" and the false positive rate is termed as "false alarm rate"). The ROC curve is thus the recall as a function of false alarm rate. Generally, once the probability distributions of both recall and false alarm are generated, ROC curve can be plotted with the cumulative distribution (area measurable under the probability distribution curve from $-\infty$ to $+\infty$) of the probability of recall on y-axis against the cumulative distribution of the probability of false alarm on x-axis.

The Scikit-learn package in python provides the methods roc_curve() and auc() as part of the module sklearn.metrics. The method signature of roc_curve() is as follows.

roc_curve(*test_true*, *test_score*, *position_label=None*, *sample_weight=None*)

where *test_true* is an array of n samples containing true binary labels either in the range {0, 1} or in the range {-1, 1}. Position_label should be given explicitly if the provided labels are not binary.

test_score is an array of n samples containing the predicted scores which can be probability estimates of either the positive class or the confidence values.

position_label is an integer representing the positive label while all the other values represent negative values

sample_weight is an array of n samples which contains sample weights

This method returns the tuple (fpr, tpr, thresholds)

fpr is an array containing the increasing rates of false positives such that ith element represents the predictions' false positive rate with predicted_score is greater than or equals to thresholds [i]

tpr is an array containing the increasing rates of true positive such that ith element represents the predictions' true positive rate with predicted_score is greater than or equals to thresholds [i]

thresholds is an array containing n thresholds decreasing with the decision function which is used to compute tpr and fpr.

In this project, once the model is built, it is tested on the test data to form the predictions which are the probabilities of each trip taken by the driver of interest. The predictions along with the test labels (180 out of 200 trips by the driver of interest and other drivers selected at random are labeled as '1' and the remaining 20 trips from each of those drivers are labeled as '0') are fed as input to the roc_curve() method which returns the parameters fpr (false positive rate) and tpr (true positive rate). These parameters are fed as input to the method auc() which measures and returns the value of the area under ROC curve.

Since this project was hosted by kaggle.com, many users submitted their final entries and the entry which secured the first place has an auc score of 0.97984. However,

this project has an auc score of around o.9161 which could have stood around 92nd place out of 1528 teams that participated.

10. RESULTS

The algorithm developed in this project predicts the probabilities of each trip from all the drivers. For a particular driver of interest, in order to differentiate the trips which are actually taken from the trips which are not taken, a good measure of the probability >= 0.85 is chosen. This section of the report illustrates the results of the algorithm by comparing trips of a driver from the actual data with those derived from the algorithm.

This section flows as follows. In the first phase, it considers two trips from the driver with id = 2 (random driver chosen to illustrate the results) whose probability < 0.85 which indicates that these trips are not taken by this driver. It then considers two other trips from the same driver whose probability >= 0.85 which indicates that these trips are actually taken by this driver. In the second phase, it illustrates the trips which are actually taken and which are not taken by the drivers with id = 1 and id = 3 in different plots.

The following two plots represent trips with probability < 0.85 for the driver with id= 2.

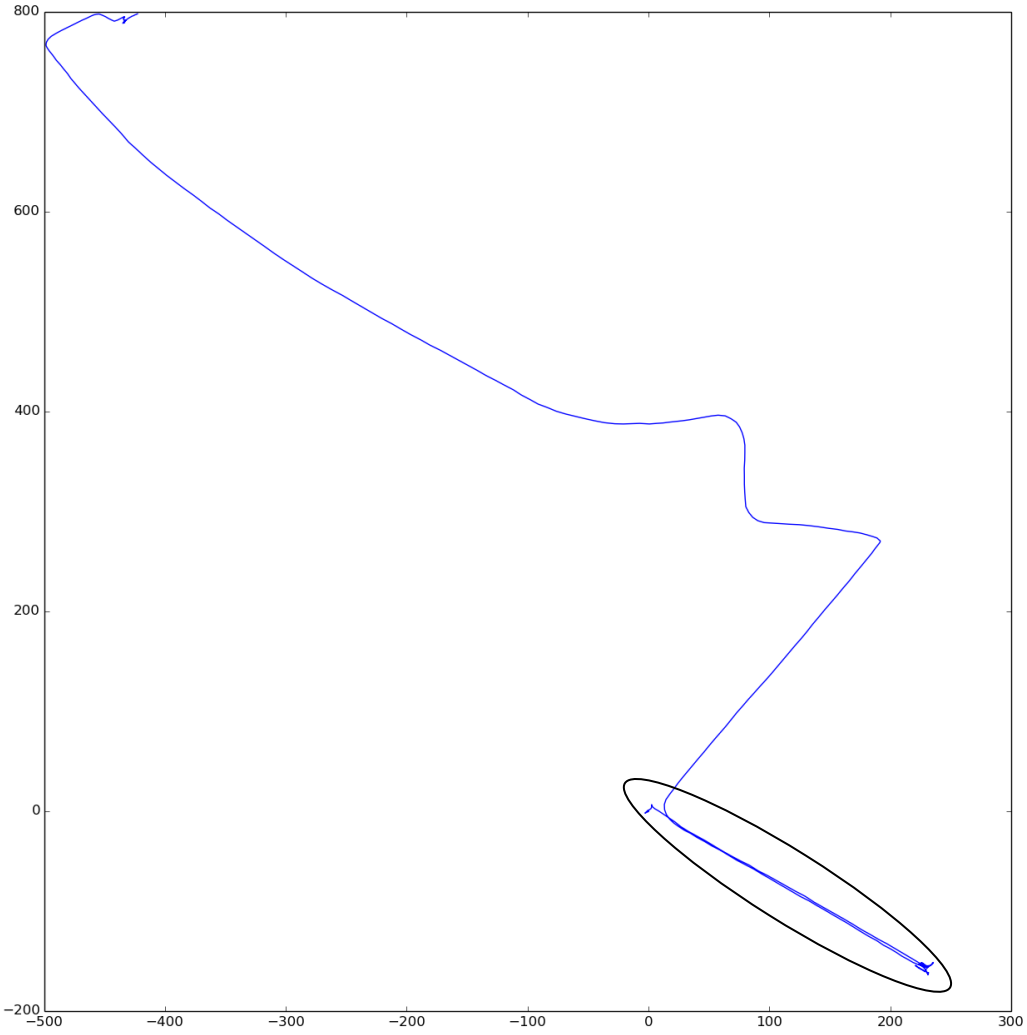

Figure 6a: This plot represents the trip with id = 47, which belong to the driver with id = 2 and whose probability < 0.85. The part of the trip shown in oval contains repetitive data points which indicate that it was driven multiple times.

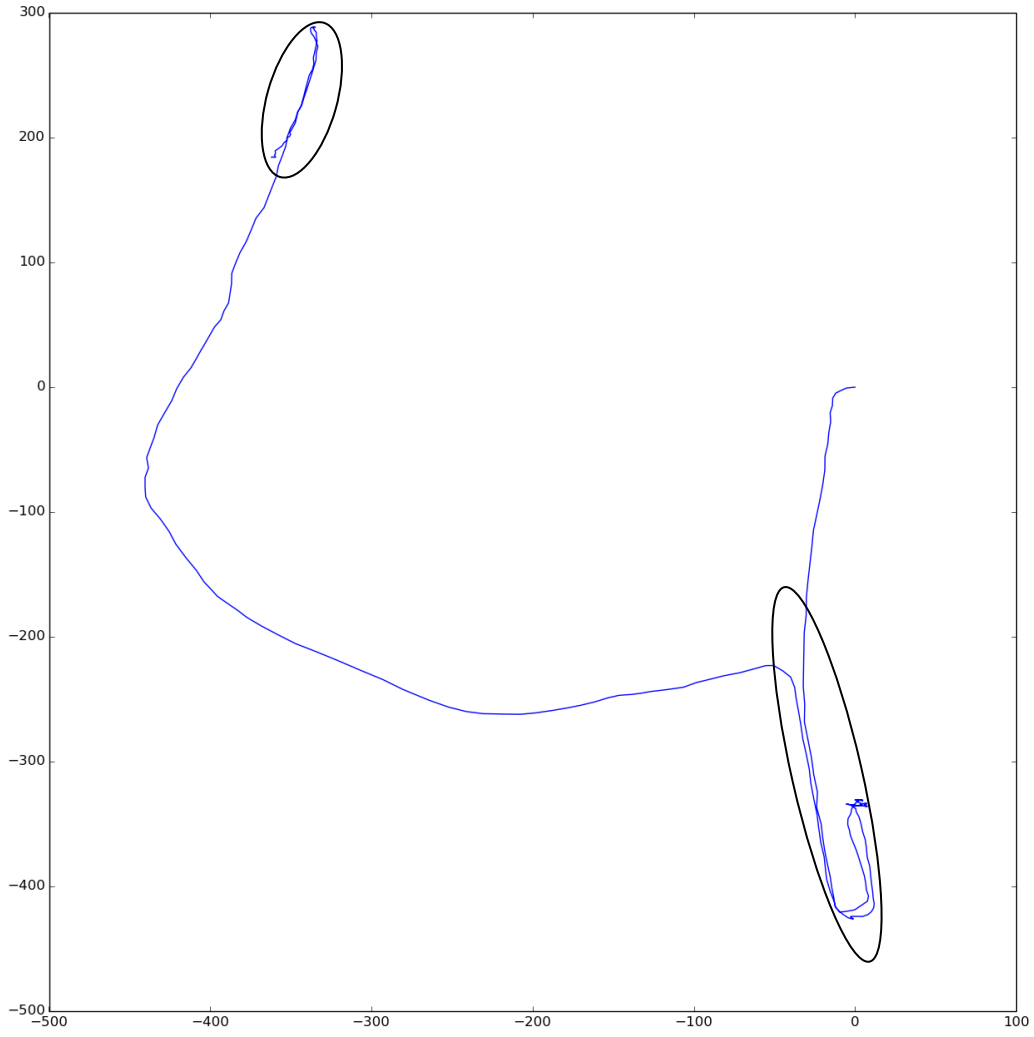

Figure 6b: This plot represents the trip with id = 153, which belongs to the driver with id=2 and whose probability < 0.85. The parts of the trip shown in ovals contain repetitive data points which indicate that they were driven multiple times.

The following two plots represent trips with probability >= 0.85 for the driver with id= 2.

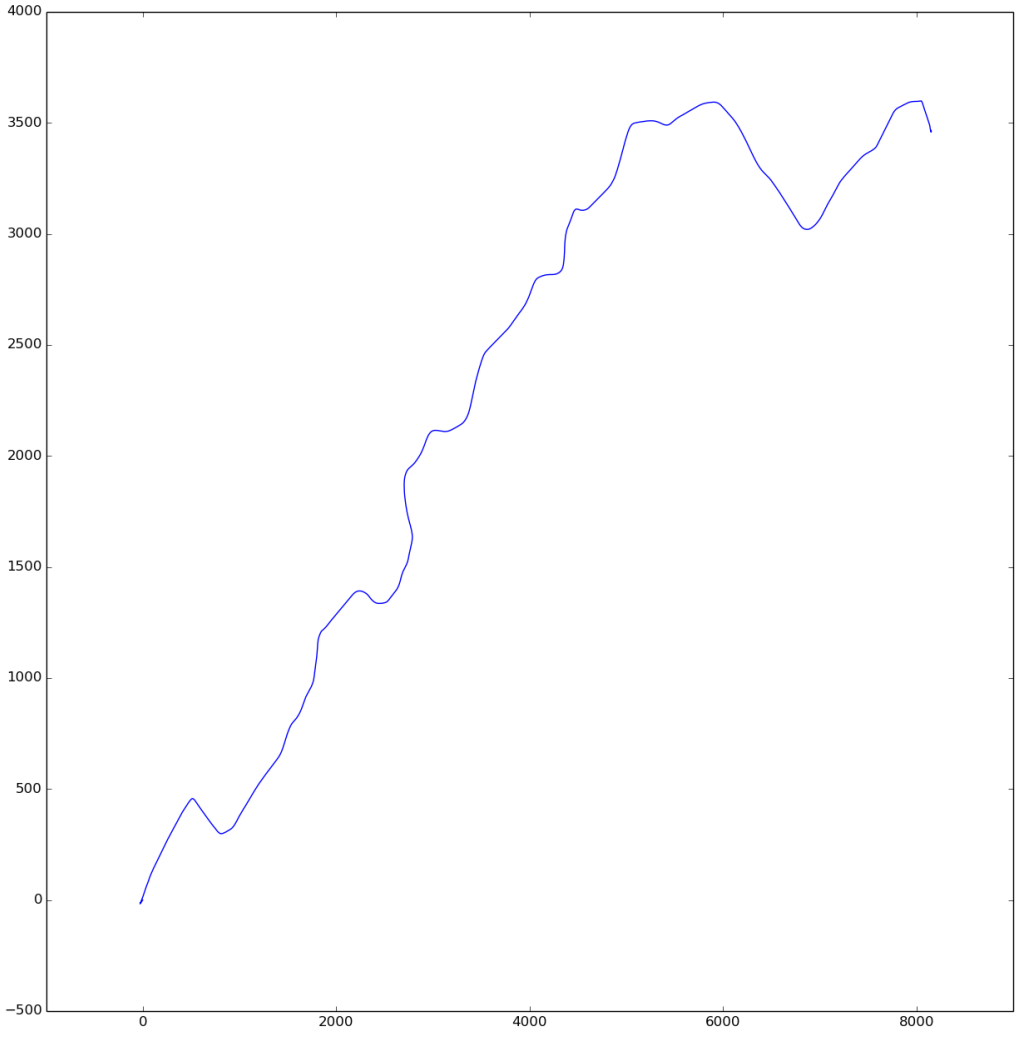

Figure 6c: This plot represents the trip with id = 25, which belong to the driver with id = 2 and whose probability >= 0.85

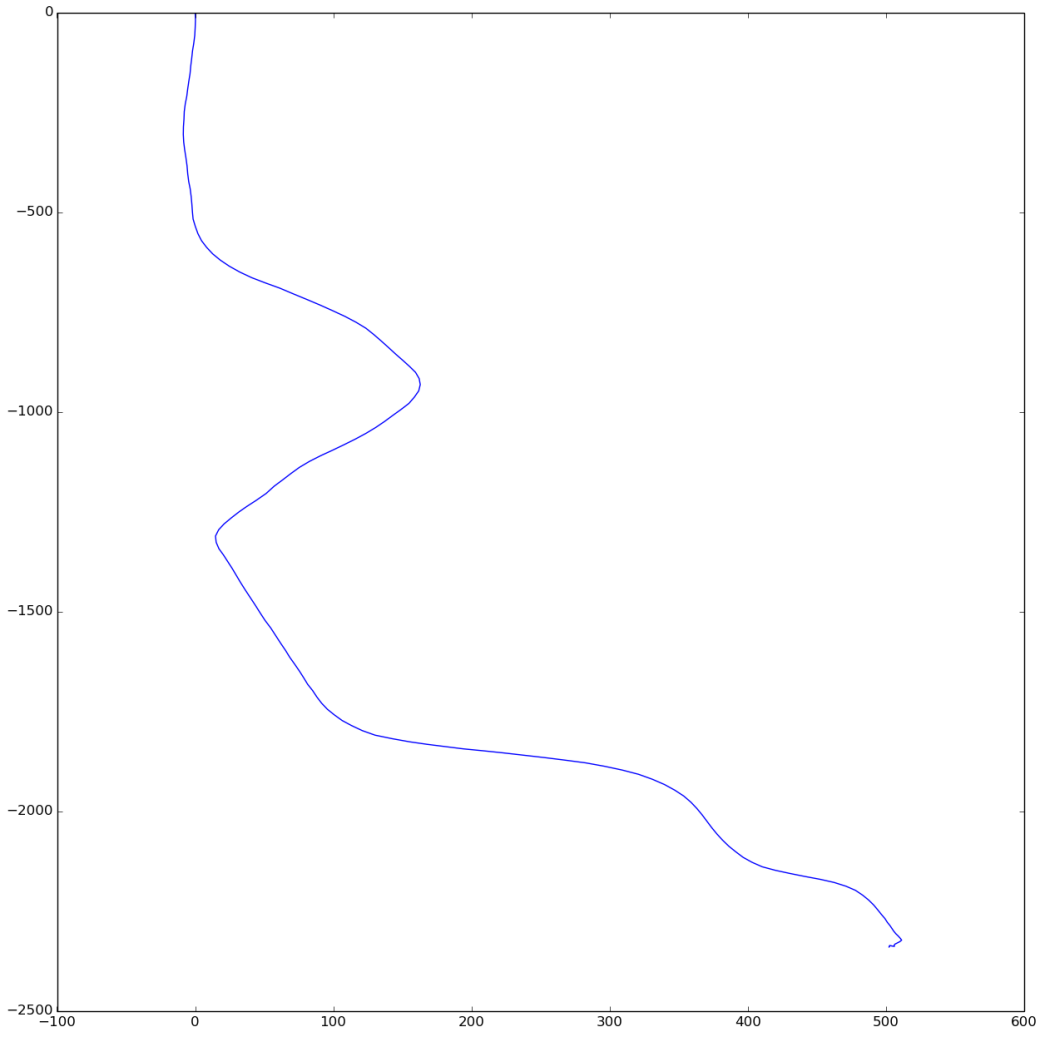

Figure 6d: This plot represents the trip with id = 61, which belong to the driver with id = 2 and whose probability >= 0.85

From the figures Figure 6a, Figure 6b, Figure 6c and Figure 6d it is clear from Figure 6a and Figure 6b that the driver with id = 2 doesn't take shorter trips which have repetitive data points. On the other hand the same driver takes longer trips which don't have repetitive data points.

The following section illustrates trips from drivers with id = 1 and id = 3.

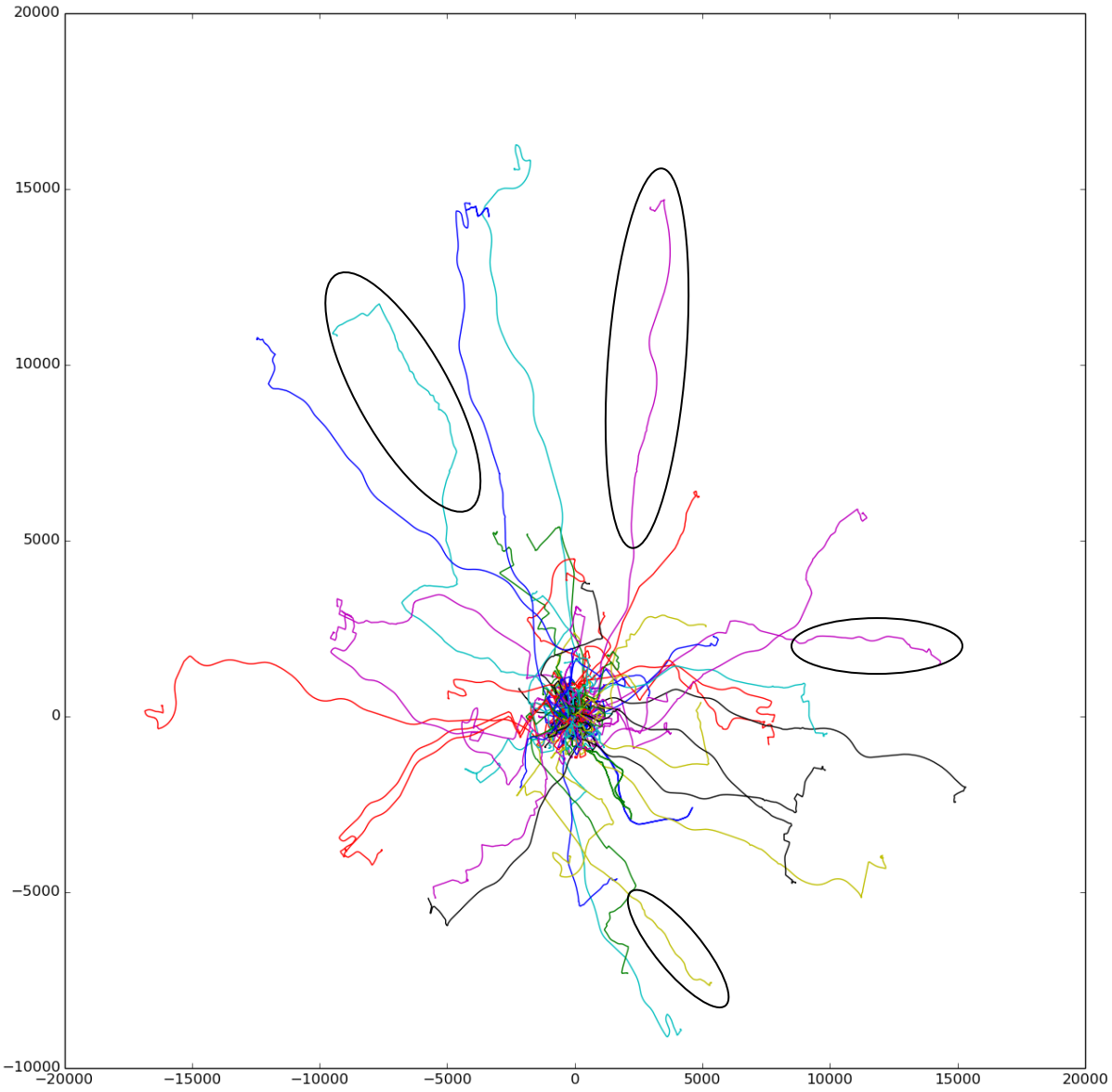

Figure 5: This plot shows all the trips taken by the driver with id = 1 as per the given dataset. The trips shown in ovals are the trips removed from the next plot which contains all the trips whose probability >= 0.85

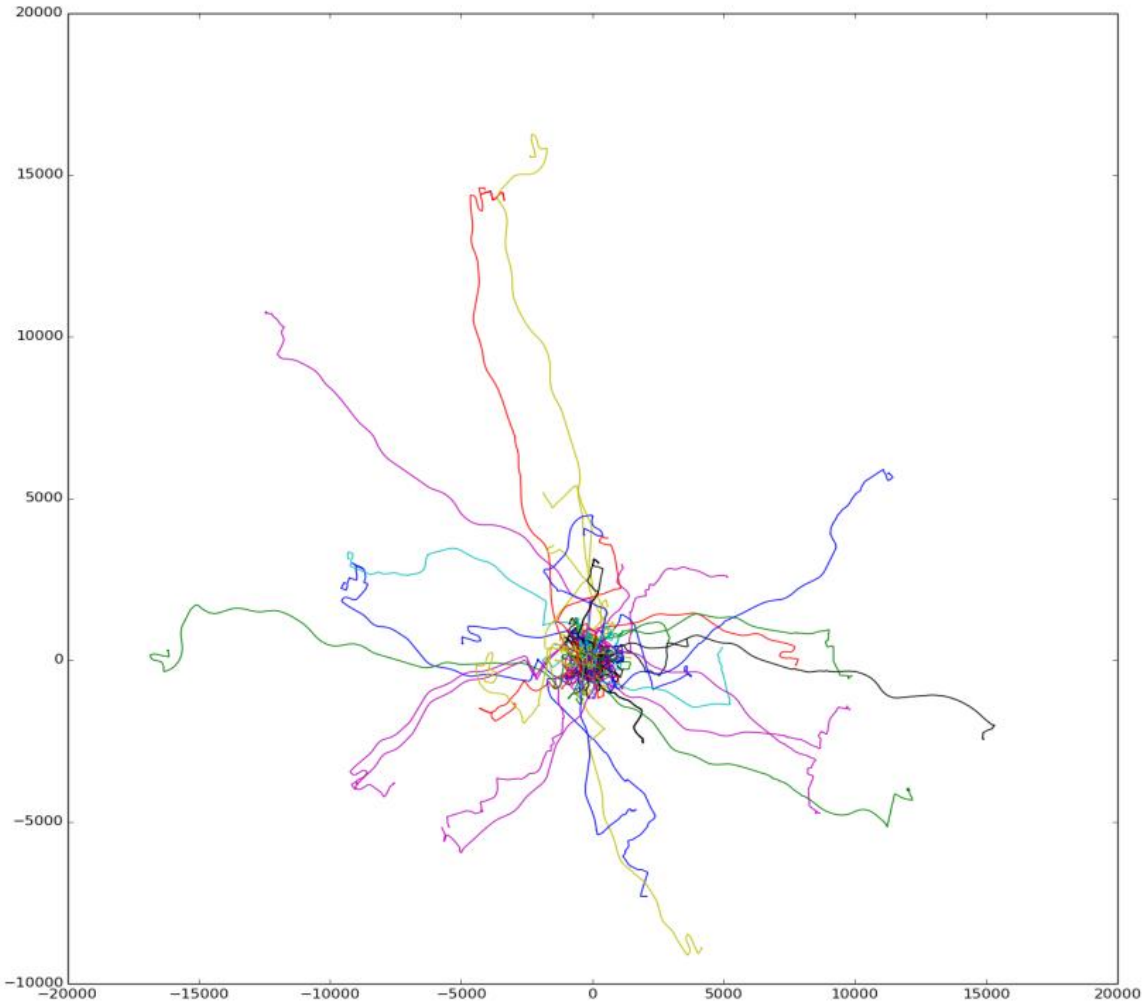

Figure 5: This plot shows all the trips of the driver with id = 1 whose probability >= 0.85. The probabilities are obtained by applying the algorithm on the test data. This plot also shows that this driver is open to both short and long trips.

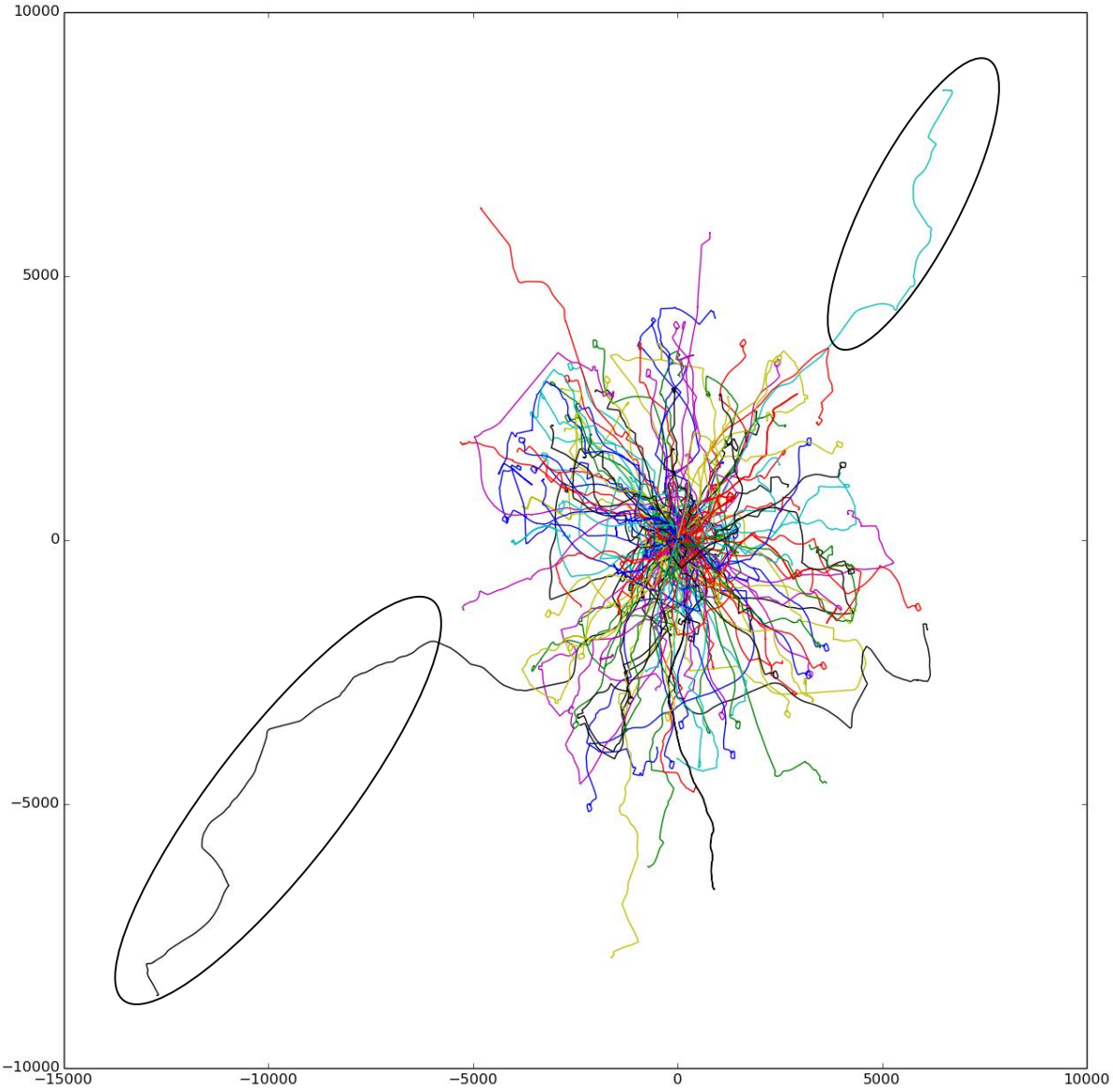

Figure 5: This plot shows all the trips taken by the driver with id = 3 as per the given dataset. The trips shown in ovals are the trips removed from the next plot which contains all the trips whose probability >= 0.85

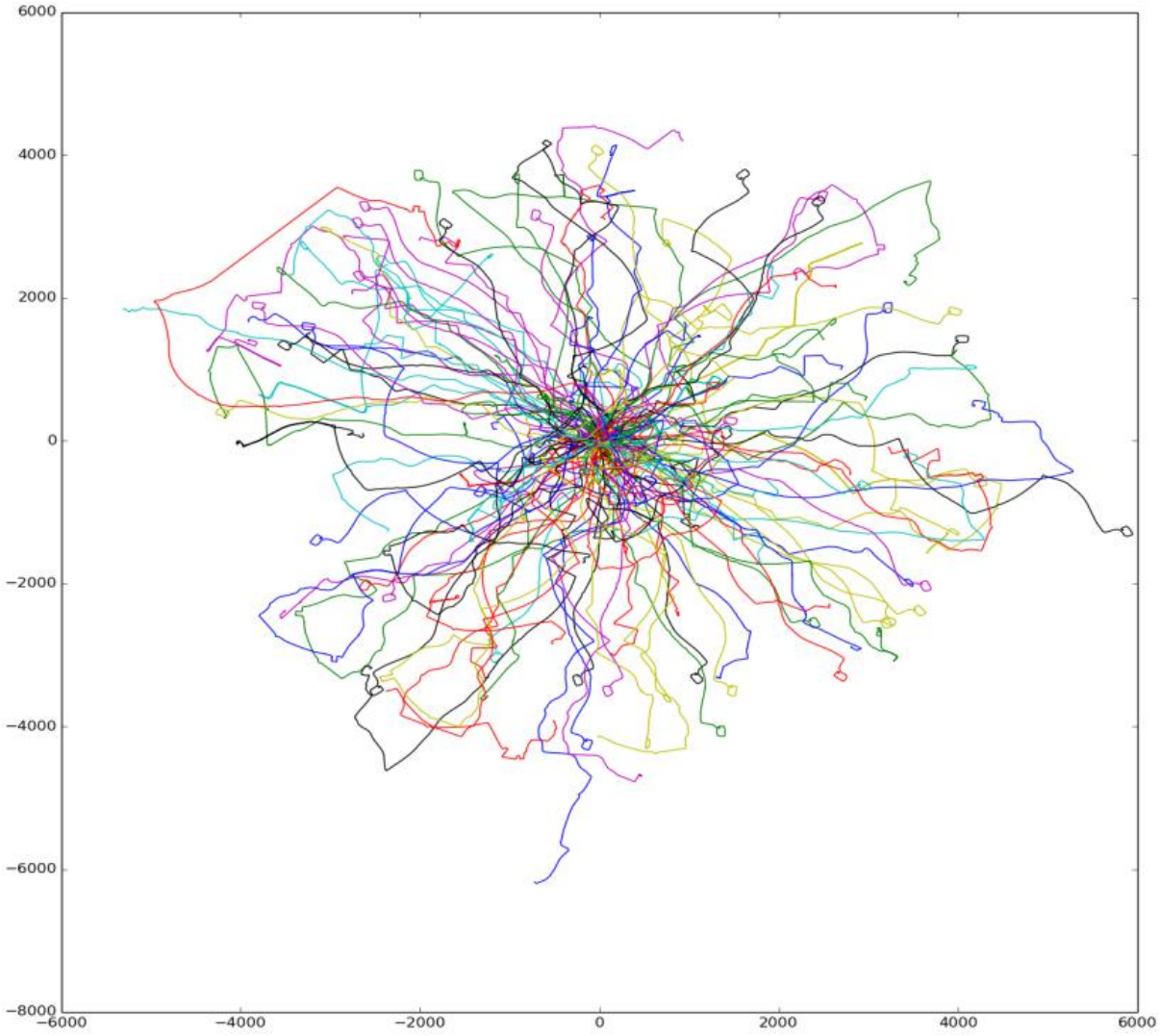

Figure 5: This plot shows all the trips of the driver with id = 1 whose probability >= 0.85

The probabilities are obtained by applying the algorithm on the test data. This plot also reveals that this driver doesn't take long trips.

11. CONCLUSION

The machine learning model built in this project has achieved an auc measure of 0.9161. Though this algorithm is not the best among all the entries, it achieved a decent score when compared with the others. This algorithm could have done even better if more diverse data was provided. Though the data is limited in terms of diversity, this model has performed better. Based on this model built, automobile insurers can know the routes taken by the driver and if the routes are riskier like the ones on steep mountains and make decisions on premium and other policies. On the other hand, if a driver always takes safer routes and when combined this profile with his/her experience in driving, the premium can be reduced for the driver or the liability can be increased with the same premium.

12. FUTURE WORK

The road segment features in this project, which are used to identify common road segments by a driver of interest, are generated using the RDP algorithm. One drawback with the RDP algorithm is the difference in segmentation process of similar curves because of the effect of movement noise on the threshold used by the algorithm. The future work of this project might include the way to encode each trip as a list of segments along with the driving instructions just like the directions of Google Maps. This advancement of the feature extraction of segments along with the driving instructions helps in obtaining better results.

13. REFERENCES

1. Driving Style Recognition for Co-operative Driving: A Survey by Angelos Amditis Anastasia Bolovinou, Francesco Bellotti and Mikko Tarkiainen
2. Driver Identification by Driving Style by Zhan Fan Quek, Eldwin Ng
3. Review of Driving Conditions Prediction and Driving Style Recognition Based Control Algorithms for Hybrid Electric Vehicles by Rui Wang and Srdjan M. Lukic
4. Driving Behavior Improvement and Driver Recognition Based on Real-Time Driving by Kexin Nie, Luyan Wu and Jiafan Yu
5. Mining GPS Data for Trajectory Recommendation by Peifeng Yin, Mao Ye, Wang-Chien Lee and Zhenhui Li
6. Pattern recognition and classification for multivariate time series by S. Spiegel, J. Gaebler, A. Lommatzsch, E. De Luca, and S. Albayrak
7. Improved Driving Cycle for Testing Automotive Exhaust Emissions by M. Kuhler and D. Karstens
8. http://www.scipy.org/about.html
9. http://scikit-learn.org/dev/tutorial/basic/tutorial.html
10. https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm
11. http://www.statsoft.com/Textbook/Boosting-Trees-Regression-Classification
12. http://en.wikipedia.org/wiki/Linear_regression
13. http://www.statsoft.com/textbook/support-vector-machines
14. http://karthaus.nl/rdp/
15. http://en.wikipedia.org/wiki/Receiver_operating_characteristic