Vehicular Classification using GPS Footprints

Swadesh Vaibhav, Vipin Baswan

2 November, 2019

Contents

1 Abstract

Vehicle Classification has emerged as an important field of study because of its importance in traffic management, surveillance, security systems, etc. GPS enabled location trackers can store the location history of terrestrial vehicles via satellite. This data can further be used for various analytical purposes. ***One such purpose is determining the class of vehicles.***

2 Objective

The prime objective of this project is to determine, as accurately as possible, the class of vehicles in the given dataset through various machine learning techniques.

3 Previous Work

There has been previous aiming at classifying vehicles using GPS data. A Support Vector Machine (SVM) model was used on the variances of acceleration and deceleration rates of vehicles by *Sun, Zhanbo* and *Ban, Xuegang* in their study, *'Vehicle Classification using GPS Data'* to achieve a test misclassification rate of 4 percent. Major observations from this paper are as follows:

1. Features related to the acceleration and deceleration are the most effective features in terms of vehicle classification using GPS Data.

2. Features related to speed (like maximum speed, average speed, variance in speed) are less important factors for classification because they are highly sensitive to the level of congestion on the roads.

3. The maximum acceleration and deceleration are not very salient features for classification.

Hence, variance in acceleration and deceleration are the most prominent features to classify the vehicles.

A similar paper was published by a group of Fleetmatics researchers titled under *'Vehicle Classification from Low Frequency GPS Data'*, which used an SVM model to get results.

4 Data Collection and Description

The dataset used was provided by MapMyIndia. It contained the coordinates of vehicles at irregular intervals of time, collected through the *Fastrackerz* service. The training data provided to us had **3168** files in total.

Each file had the following information:

1. Vehicle ID
2. Latitude and Longitude
3. Time Stamps (Date and time format)
4. Speed of vehicle
5. Engine status (0 means engine is off, 1 means it is on)
6. Vehicle label

The data was divided into two parts: Training Data and Test Data, with meanings standard to the Machine Learning paradigm.

The format used was *'csv'* format. The data had multiple csv files, with each file containing the locations of multiple vehicles for a particular interval of time.

Given below are the categories of vehicles along with their proportion in the dataset:

1. Car : 2549 (78%)
2. Ambulance : 588 (18%)
3. Motor Cycle : 18 (0.5%)
4. Bus : 16 (0.49%)
5. JCB : 1 (0.03%)
6. Truck : 58 (1.7%)
7. Tractor : 15 (0.46%)
8. Mini_Bus : 1 (0.03%)
9. School_ Bus : 1 (0.03%)
10. Not Available : 34 (1.04%, Removed from the training dataset)

As it is evident from the above statistics, the training data primarily had two major classes of vehicles: **Car** and **Ambulance**. Both of these classes formed 96% of the dataset. Since not enough data was present for training of vehicles of sother classes, we expected the classification to be almost binary.

5 Method

5.1 Data Preprocessing

Data preprocessing was done in multiple steps.

1. The first step was to read all the files in the training data and create 'csv' files corresponding to each vehicle ID. At the end, we had **3258** files.

2. The geolocations of vehicles at various timestamps was used to calculate the distance travelled by each of them in various intervals of time using standard formulae.

3. Then we estimated the mean and the variance of the speed and acceleration, for each vehicle.

We collected 6 features from the dataset:

1. Average of the speed of the vehicle

2. Average of the change in the position of the vehicle for timestamp *t1* to timestamp *t2* (called as *speed_2*)

3. Variance of the speed of the vehicle

4. Variance of the change in the position of the vehicle for timestamp *t1* to timestamp *t2* (called as *speed_2*)

5. Average acceleration of the vehicle

6. Variance of acceleration of the vehicle

To know about the contribution of each factor in the variance of the dataset, we did Prinicpal Compnent Analysis (PCA). The results are as following:

1. 0.611 explained_variance_ratio using only 1st feature

2. 0.907 explained_variance_ratio using first two featuress

3. 0.960 explained_variance_ratio using first three features
4. 0.9998 explained_variance_ratio using first four features
5. 0.9999 explained_variance_ratio using first five features
6. 1.0 explained_variance_ratio using all features

To understand correlation between features of the model, we went for many visualizations of the data. We wanted to begin with better understanding of each feature individually. So we plotted histogram for each feature:

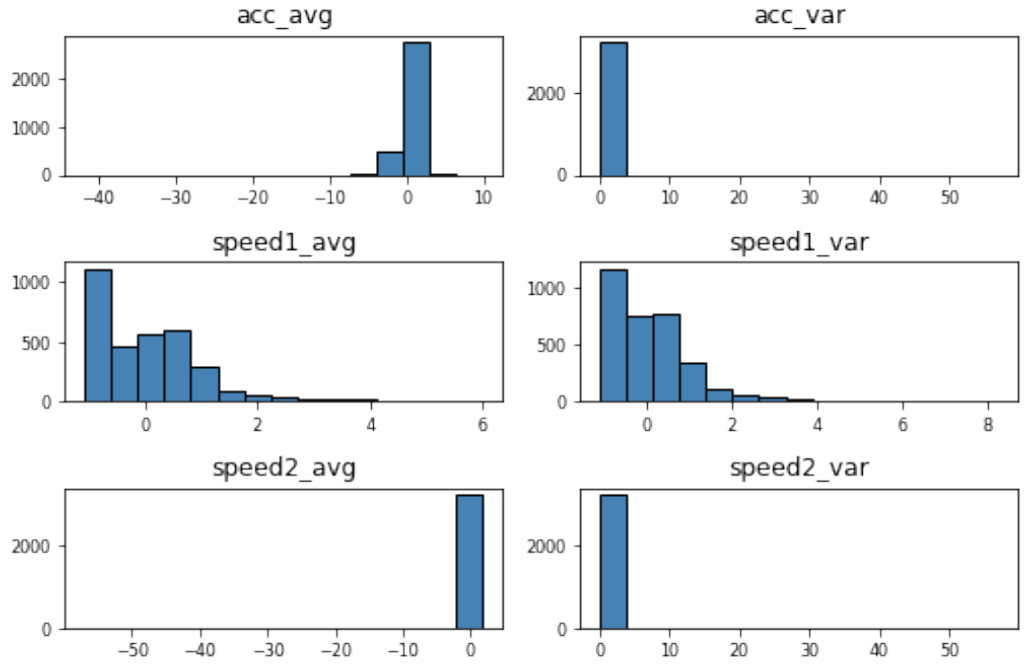

Figure 1: Histogram for each feature

Now to study correlation of one feature with others we used Heat Maps. We found that *speed2_avg* was correlated to *acc_avg*. But since out aim was not feature reduction but merely understanding of data, we didn't remove the correlated features.

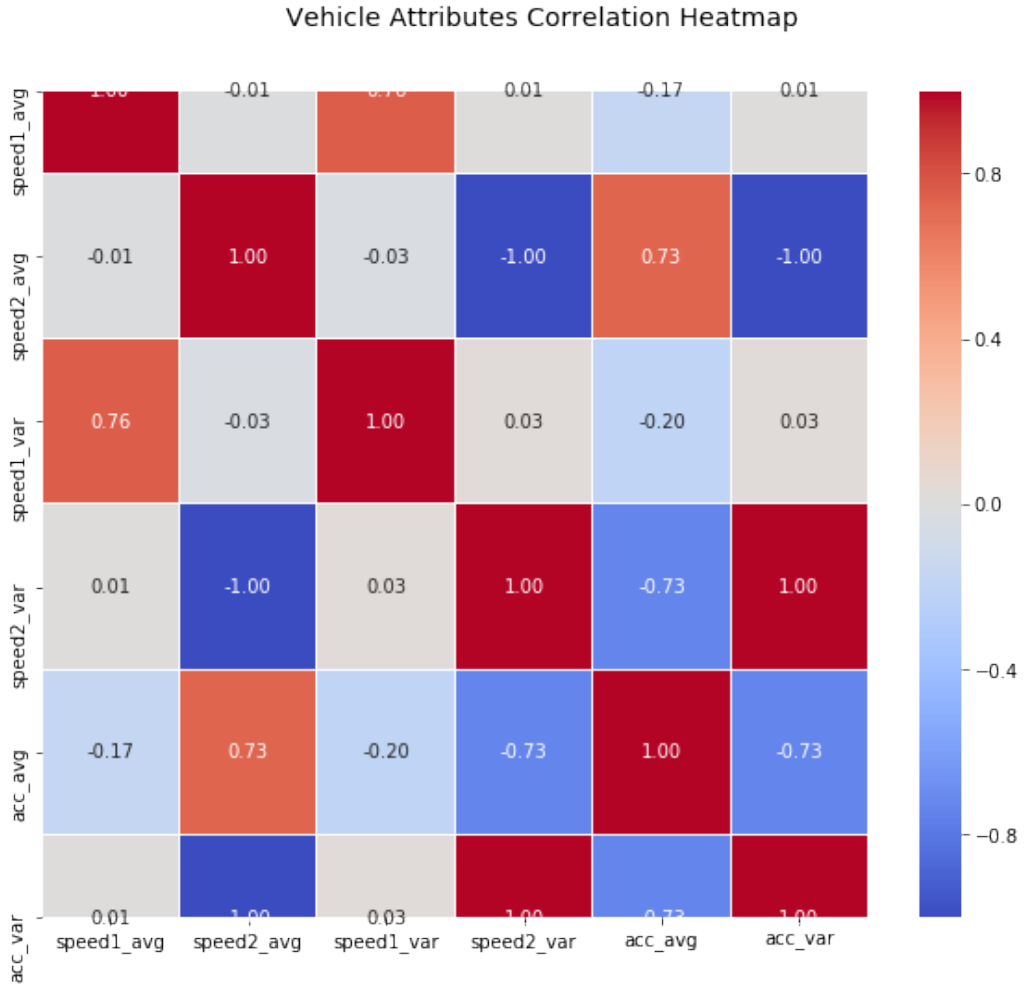

Figure 2: Histogram for each feature

5.2 Training: Classical ML Models

For training, we used the classical Machine Learning models. We got best performance (in terms of accuracy) from **Random Forest Classifier** and **eXtreme Gradient Boost (XGB) Classifier,** with accuracy of 93% and 93.4% respectively.

MODEL	Fold1	Fold2	Fold3	Fold4	Fold5	Avg
XGB Classifier	93.86	91.86	92.31	91.08	92.46	92.31
Random Forest	93.86	91.86	92.31	91.38	92.77	92.44
Bagging Classifier	92.93	90.17	91.23	91.08	92.31	91.54
Decision Tree Classifier	92.93	90.78	91.69	90.15	92.0	91.51
Gradient Boosting Classifier	92.63	90.63	91.85	89.69	91.38	91.24
Ada Boost Classifier	88.63	85.25	79.08	87.54	88.77	85.85
SVC	84.64	84.33	82.46	81.08	82.15	82.93
KNN	83.41	82.03	81.38	79.38	82.62	81.76
Bernaulli Naive Bayes	79.57	80.49	77.23	76.15	79.38	78.57
SGD Classifier	79.26	80.03	77.23	75.54	79.23	78.26

Table 1: Various Models used along with their fold and average accuracy (using 5-fold cross validation)

As evident from the Table 1 above, XGB Classifier gave an accuracy of 93.4%. To look at the *class-wise accuracy* of the trained model, given below is the confusion matrix for the same:

Class	**Ambulance**	**Bus**	**Car**	**Motor Cycle**	**Tractor**	**Truck**
Ambulance	107	0	16	0	0	0
Bus	1	0	4	0	0	0
Car	4	0	500	0	0	0
Motor Cycle	2	0	0	1	0	0
Tractor	3	0	0	0	0	0
Truck	8	0	5	0	0	0

Table 2: Confusion Matrix for XGB Classifier

From the confusion matrix above, we can infer the following:

1. 4 cars out of 504 cars have been misclassified as ambulances. This gives us misclassification rate for cars as 0.7%.

2. 16 ambulances out of 123 total have been misclassified as cars. This gives us an error of 13% in the classification of ambulances.

3. Not enough data was present for classification of vehicles of other classes (other than car and ambulances).

Class	Precision	Recall	f1-score	Support
Car	0.95	0.99	0.97	504
Ambulance	0.86	0.87	0.86	123
Bus	0.00	0.00	0.00	5
Motor Cycle	1.00	0.33	0.50	3
Tractor	0.00	0.00	0.00	3
Truck	0.00	0.00	0.00	13
Accuracy			**93%**	651
macro avg	0.47	0.37	0.39	651
weighted avg	0.90	0.93	0.92	651

Table 3: Classification report for XGB Classifier

The above table is for 80:20 divison of dataset. Model was trained on 80% (i.e. 2607 tuples) and tested on 20% (i.e. 651 tuples). We can see the class-wise accuracy in the above table. **Car** had accuracy of 97% and **Ambulance** had an accuracy of 86%. It can be observed that the accuracy is commensurate with the proportion of training data available for that class.

5.3 Training: Dense Neural Network

As the next step, we trained our model using Dense Neural Networks given by the Keras framework. It gave a test accuracy of 89.2%.
The model architecture consisted of 6 fully connected Neural layers, with 8, 12, 16, 20, 15 and 10 neural nodes respectively. All the layers were activated using the 'relu' function. The final output was obtained using the softmax function.

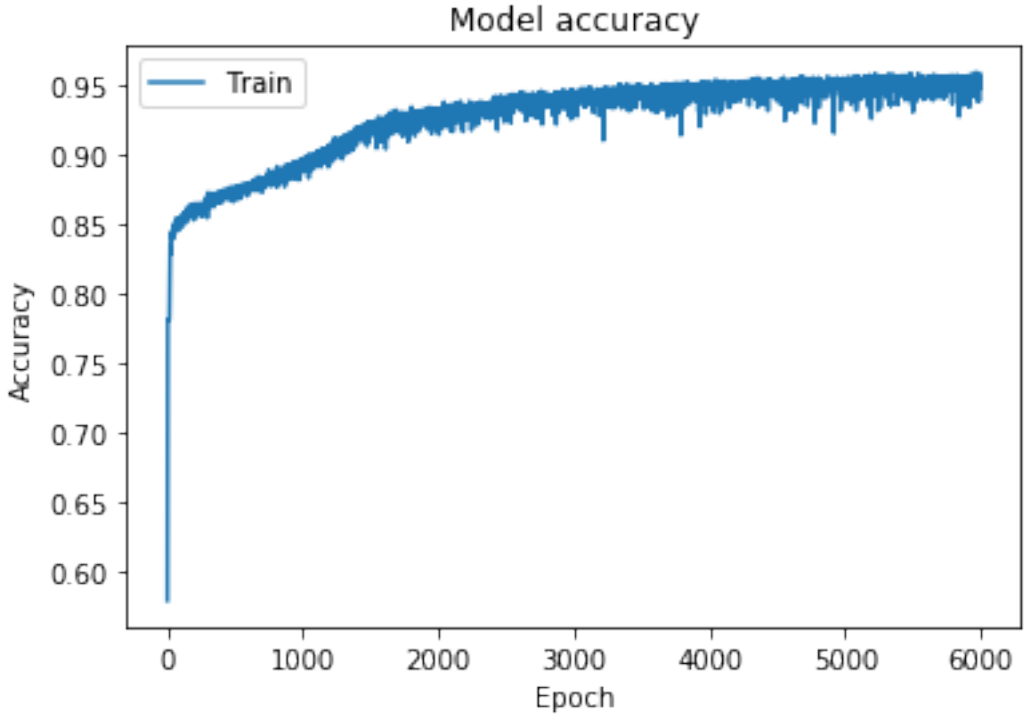

Figure 3: Model Accuracy of Neural Nets as a function of the number of Epochs

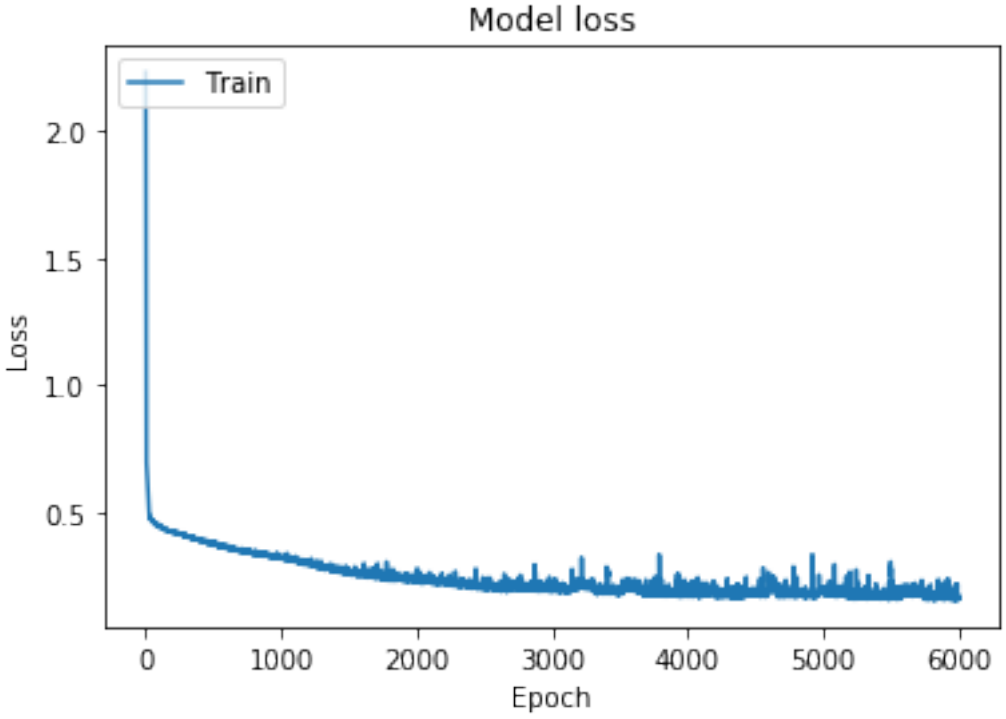

Figure 4: Model Loss of Neural Nets as a function of the number of Epochs

The above graphs show that the training model accuracy saturates at approximately 95% (number of Epochs as 6000). Also, as expected, the loss is also minimum at 6000 epochs. So, we trained our model for **epochs = 6000** and **batch_size=64.**