Final Capstone Report

University of California, Berkeley

Modeling Interactions between Autonomous Vehicles and Pedestrians

---

Authors:

Salma Benslimane
Dan Xu
Mubarak Abdul Kader
Moyu Li
Dayi Wang

Advisers:

Dr. Gabriel Gomes - Faculty, Mechanical Engineering, UC Berkeley
Dr. Ching-Yao Chan - Research Engineering, California PATH, UC Berkeley

Submitted: May 4, 2018

We affirm that we are the sole authors of this report and we give due credit to all used sources.

Salma Benslimane — March 30, 2018 3/30/2018
Salma Benslimane

Dan Xu — March 30, 2018 3/30/2018
Dan Xu

Mubarak Abdul Kader — March 30, 2018 3/30/2018
Mubarak Abdul Kader

Moyu Li — March 30, 2018 3/30/2018
Moyu Li

Dayi Wang — March 30, 2018 3/30/2018
Dayi Wang

Turnitin Report

The initial Turnitin score was 59%. After we excluded our previous submissions, the score dropped to 2%. We went through the matches that caused the 2% similarity.

Page 8 contained two matches, this was paraphrased to convey what we wanted to through the source that we read.

Abstract

Automation in the automobile industry is growing rapidly these present years, and it will completely replace traditional vehicles in the next few decades. Being a new technology, where the vehicle has the capability to guide itself without human involvement, the autonomous vehicles have to gradually earn market shares from human-driven vehicles. Autonomous vehicles (AVs) have higher efficiency and accuracy because of the absence of human errors which account for 90% of the accidents on road. It is also expected to significantly reduce traffic congestion, energy consumption and increase shared cars ownership.

A large challenge faced by autonomous driving is how the autonomous vehicle interact with pedestrians, especially at intersections and in situations where jaywalkers exist. The aim of the project is to develop a model that would mimic human behavior for autonomous cars, such that it would provide and bring a feeling of safety on the road for the riders of autonomous cars as well as all the pedestrians, especially the jaywalkers. This project involves trajectory extraction by utilizing Traffic Intelligence Code by Nicolas Saunier. The extracted trajectory is then used to compute the velocity and acceleration of the vehicles and pedestrians. These features are further manipulated to extract the angle between interacting objects, which include one pedestrian and one vehicle per interaction. From this, rate of change of angle between these objects as well as the time taken for the relative distance between the objects to reach zero is computed. These features are then used to predict whether a vehicle would brake or not brake given certain values for all these features.

The two classification algorithms used are Logistic Regression and Random Forest Classifier. The overall results indicate that the models are able to predict correctly when the vehicle is not braking but has a lot of misclassification in predicting whether the vehicle is braking. Our team delivers a pipeline from data collection to model tuning so that in the future course of this project, the models can be improved further by collecting more data and a balanced one at that having close to equal number of samples for each class.

Contents

1 Introduction

In the current scenario, there are vehicles with varying amounts of automation in the industry. Broadly, they can be classified as semi-autonomous and fully autonomous vehicles. Autonomous vehicles currently being developed are equipped with GPS, laser range finders, radar as well as image processing to make the vehicle highly sensitive to the surroundings and enabling quick feedback to increase safety and functionality. However, the AVs face increased challenges when dealing with uncertainty, specifically when it comes to interactions with pedestrians and other vehicles in the environment because it cannot perfectly predict the behaviours involved. Recent incidents involving autonomous cars of Tesla and Uber have raised concerns about the maturity of the technology. These challenges validate the need for a model that can understand and predict the pedestrians' attitude based on experience and intuition.

2 Background

2.1 Conflicts in Consumers' Expectations from Autonomous Vehicles

Traffic accidents will go down in the era of autonomous vehicles, but sometimes, the AVs may have to choose between two worse situations. Some of the questions that seem to be going around are, if the survival rate is higher for the passenger in an autonomous vehicle than for a motorcycle rider, is it acceptable for the autonomous vehicle to swerve into a wall to avoid hitting the motorcycle? Should the age of pedestrians and passengers be a deciding factor while choosing during these situations? And if different versions of moral algorithm are offered by a manufacturer and knowingly, a buyer chooses one of them, will the blame fall on the buyer if the algorithm's decisions have any harmful consequences? (Bonnefon et al, Social Dilemma of autonomous vehicles, 2016)

People expect AVs to be utilitarian, in the sense that they should choose an option that would save more people even if that means harming its own passengers. This was validated by a study conducted by Jean-François Bonnefon et all. It was shown that more than 50% of the people surveyed wanted the vehicle to sacrifice its passengers if it meant saving more people, even when asked to imagine themselves to be a passenger. But the paradox was that even though they voted for an AV that ranks high on morality, they themselves were not open to buying an AV that would jeopardize its passengers to save more people. (Bonnefon et al, Social Dilemma of autonomous vehicles, 2016)

For now, until the debate on what kind of moral principles the AVs should follow for decision making during critical decisions are decided, the aim would be in mitigating the likelihood of an ethically controversial situation. For this, we'll strive to make the algorithm have an accuracy as high as possible. In the future, a good amount

of simulated and actual public testing can slowly change the negative opinions that people have about the autonomous vehicles and realize them for the true benefits that they provide.

2.2 Solving the lack of intuition for Autonomous Vehicles' response to jaywalkers can reduce possibility of accidents

Having shown that autonomous vehicles can reduce accidents, traffic congestion and improve driving efficiency in comparison with human-driven vehicles. One aspect that is up for contention is the intuition that human-drivers have and the AVs lack. This aspect plays a huge role when it comes to interacting with objects in the environment, especially pedestrians because of their unpredictable behaviour. Our project is focused on solving this problem by collecting past data on interactions between human-driven vehicles and pedestrians to understand behaviours. This understanding can be modelled into an algorithm that can be used to teach an autonomous vehicle to compensate for the intuition it would lack.

3 Approach

Our model will take a pedestrian's varying position and speed as input, and instruct the vehicle on the actions it should take. The model is being trained on interaction data coming from video footages of Hearst Ave in front of Etcheverry Hall. The output will be a simulation on the test data which is unseen by the model. We imagine the output format to be in the form of a video where a car is identified and the action recommended by our algorithm is shown as an overlay to compare visually and see what the vehicle actually did, and hence, the accuracy.

Broadly, our approach can be dissected into six phases, (1) Video data processing, (2) Trajectory Extraction, (3) Database Creation & Modeling, (4) Video Overlay Output.

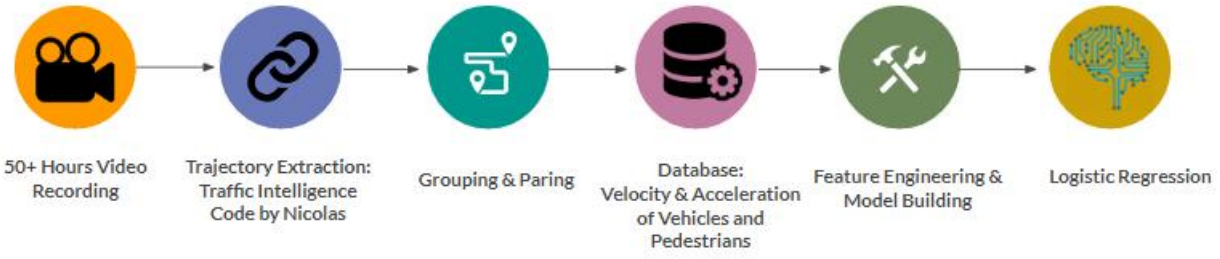

Figure 1: Flow Diagram indicating the approach of the project

4 Literature Review

In the research of transportation and vehicle behavior study, video data is commonly used, especially in the study of road users' behavior analysis. Video-based traffic analysis usually consists of four main steps: data collection, data preprocessing, data processing and data analysis.(Ismail, K, 2010) When we conduct behavior analysis, computer vision techniques are used to generate road user trajectories. Trajectories are then analyzed to yield traffic measurements (vehicle position and speed, pedestrian position and speed, lane change rate, etc.), driver's behavior and pedestrian's behavior. One important step of our analysis is the integration of a set of open source automatic tracking algorithms for data processing. The algorithm of vehicle and pedestrian trajectory extraction is based on Nicolas Saunier's paper and intelligence traffic code. Nicolas Saunier is the associate professor in transportation engineering at Polytechnique Montreal in Canada and his research focuses on intelligent transportation, road safety and data science for transportation.

5 Trajectory Extraction

5.1 Trajectory Extraction can be performed based on Nicolas Saunier's Papers and Algorithms

To extract pedestrian and vehicle trajectory data from video, the data collection system is integrated with the open source software called Traffic Intelligence, which is a feature-based tracking algorithm. The algorithm consists of two steps: (1) detect individual pixels with same feature as they move over time frames, which forms trajectory to be recorded. Fig.2 [Bradski, G, 2008]. The Kanade–Lucas–Tomasi feature tracking algorithm is applied here. It is a method of feature extraction in computer vision and it uses spatial intensity information to direct the search for the position that has the best match. For every object, there are several features on it and then the objects can be classified and grouped. (2) Feature trajectories are grouped on the basis of consistent common motion. The parameters of the grouping algorithm are tuned based on trials, which then leads to a trade-off between "oversegmentation (one object being tracked as many)" and "overgrouping (many objects tracked as one)". [Saunier, N., and T. Sayed. 2006]. Time is measured in frames and the feature position is recorded in every frame. Then we are able to conduct calculation by using the constant recording frame rate. For instance, 50 frames per second means that a displacement of 1 meter from one data point to the next point represents an object traveling at a speed of 50 meter per second. The trajectories are extracted and smoothened with the use of a moving average.

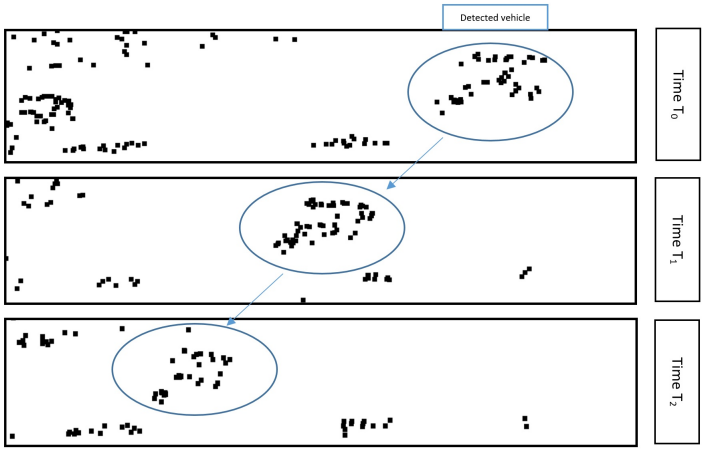

Figure 2: Plot of moving object in time using background extraction from Saunier's code

5.2 Code and Parameters Modifications on top of the Traffic Intelligence Code were required to extract the trajectories

When setting up the environment for the open source code, an old version (2.4.13) of library OpenCV, which is required by the Traffic Intelligence code, is not available on our Windows Operating System. Hence, Dr. Xiao Zhou, one of our advisors from California PATH, modifies the code so that it is compatible with a newer version (3.2) of OpenCV. Besides, as stated in the home page of Traffic Intelligence, the code requires other additional libraries to support its calculations and data storage.

As mentioned in the previous subsection, tracking part of the code generates trajectories of different feature points. Since some features belong to a same object, say a pedestrian or a vehicle, Traffic Intelligence then groups those features together as one object. In our case, the parameters used for grouping are adjusted to avoid oversegmentation and overgrouping problems of objects. [Saunier, N., and T. Sayed. 2006].

5.3 Transforming Pixels into Real World Distance is required to analyze trajectories effectively

Homography transformation is applied for perspective correction, as cameras have certain orientation. It is transformation which projects coordinates onto homogeneous plane. [Dubrofsky, E. 2009]. In our case, the homography matrix H is used to transform pixel coordinates S into homogeneous coordinates D in real distance

by

$$\begin{bmatrix} D_x \\ D_y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} S_x \\ S_y \\ 1 \end{bmatrix}$$

Four pairs of S and D are required to determine H, but more pairs can be used to calibrate the values. [Dubrofsky, E. 2009].

5.4 Visualising the results validates the correctness of the code

As explained previously, the openCv code allows to identify and track objects and is supposed to classify them. Most importantly, it gives their positions and velocities as well as their properties size, width, number ... and allows to plot the trajectories on the videos.

Figure 3: Overlay of plot of objects' trajectories (red lines) on the processed videos

Noting that Nicolas' code is from an opencv library, we faced lot of difficulties setting up the environment, running the code and getting consistent results. However, we verified that in fact the positions are those of the moving objects on the videos as shown in Fig.3

6 Database Creation and Modeling

6.1 Modeling Drivers' Behavior requires Data Preprocessing

After extracting trajectories of moving objects from the videos, we would obtain a database, as an SQLITE file, containing all the information that we would a-priori need to build the model. The database contains 4 tables with the following information

- **Objects** which contains object IDs and types [vehicles, pedestrians, bicycles...] (*Note: object classification does not function properly from OpenCv code - in the following paragraphs, we will present an alternative solution to classification*),

- **Objects Features** which contains object ID and trajectory ID,
- **Positions** which contains trajectory ID, frame number, x positions and y positions,
- **Velocities** which contains trajectory ID, frame number, velocities along x axis, velocities along y axis,

To pre-process the data, we use Pandas and Python. We first convert all the SQLITE files into Pandas Dataframes for clarity and practicality. Then, we classify the target objects: vehicles and pedestrians

6.2 Manual Classification can be based on threshold and direction of trajectories

Our aim is to extract, from all moving objects in the database, only the vehicles and the pedestrians. As explained previously, the openCv code fails to classify those. Thus, we use the properties that vehicles only move along the x axis and most of the time crosses the whole road and that the pedestrians crosses the street perpendicularly to the vehicles.
We define a threshold, and set a criterion to classify vehicles and pedestrians:

- Vehicles move more than 10 meters along x direction: $Veh \quad if \quad \Delta(x) \geq 10m$
- Pedestrians move more than 2 meters along y direction: $Ped \quad if \quad \Delta(y) \geq 2m$

We apply the criterion and get the IDs of pedestrians and vehicles that we arrange in a dataframe.
At this stage, we decide to save the dataframes using python's function 'pickle'(meaning we save database containing the IDs of vehicles and pedestrians in files). It is important to save intermediate files because it allows to debug the code more efficiently and allows to work simultaneously which saves time. The next step is to arrange these vehicles and pedestrians into pairs.

6.3 Each interaction need to be labeled manually to model the behavior

An interaction represents the amount of time when a jaywalker is crossing while a car is heading his or her direction. The definition of an interaction entails having both the data of the vehicles and pedestrians matching and sequenced the same way. Our data collection contains many pedestrians' and vehicles' trajectories together with assigned object ID in synchronized time frame. However, we do not know which pair of pedestrian and vehicle, denoted by pair of object IDs, is actually interacting in an interaction. Therefore, we use the trajectories to reconstruct the video based on time frame, so that we can pair pedestrians and vehicles together manually for each interaction. The pair form would be (1, A), which means that trajectory of pedestrian 1 actually interacts with trajectory of vehicle A at certain time.

7 Meaningful Features Are Required For Machine Learning model to work

Feature engineering is the process of using collected data to create features that make machine learning algorithms work. Feature engineering is fundamental to the application of machine learning because a right feature engineering is able to increase the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning modeling process.

The selected feature will directly affect the predictive model and the results. Better feature mean flexibility, simpler models and better results. Good feature with flexibility will allow you to use less complicated model which is easily built and maintained. With good feature engineering, there is no need to pick the most right model and the most optimized parameters. A representation of all the data is used to characterize the underlying issue.

The selection of feature engineering is a iterative process and the basic steps of designing feature engineering include: (1)Design a set of features; (2)Run experiment and analyze the result; (3)Change feature set; (4)Go to step 1. During our capstone project, we selected two feature engineerings. The first feature engineering we tested include relative speed and relative distance of interacted vehicle and pedestrian. The diagram is shown in Figure 4. The second feature engineering involves the angle between vehicle and pedestrian, rate of the angle and relative distance, which is shown in Figure 5.

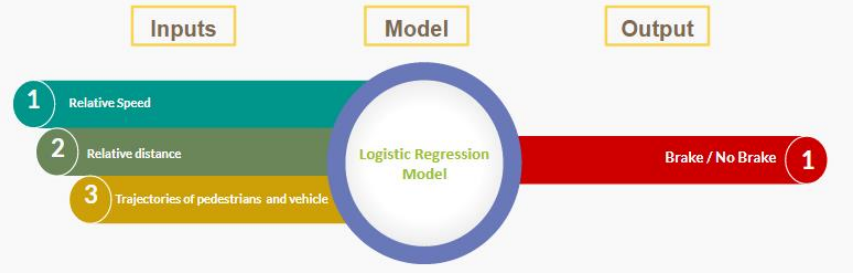

Figure 4: Feature Engineering I:Relative Speed and Relative Distance

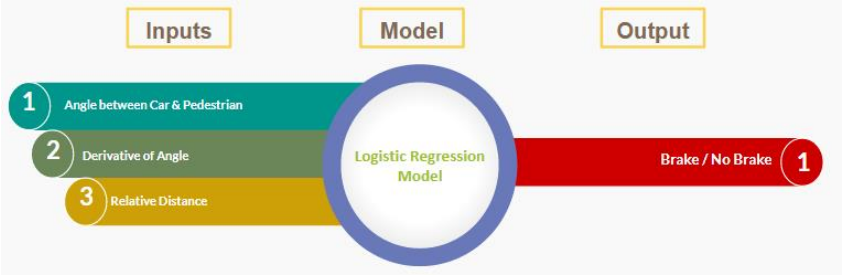

Figure 5: Feature Engineering II:Angle Between Vehicle and Pedestrian,Rate of Angle and Relative Distance

In the study of driver's behavior of brake or not brake when a pedestrian is crossing the street, we realize the main parameters are the relative speed and relative distance. Absolute speed and distance don't give us much information. Also, we need to take the relative positions of vehicle and pedestrian into account. For example, if the pedestrian is in front of vehicle and looking for opportunity to cross the street, the driver must observe the behavior of pedestrian and decide to brake or not brake. In contrast, if the pedestrian is behind the vehicle, there is no need for the driver to brake. Thus, we create a feature called the angle between vehicle and pedestrian. If the angle falls into the range of 0 and Pi/2, it means the pedestrian is in front of car and the driver need to make a decision to brake or not brake. If the angle is larger than Pi/2, the pedestrian is behind the vehicle and the driver's decision should always be "Not Brake".

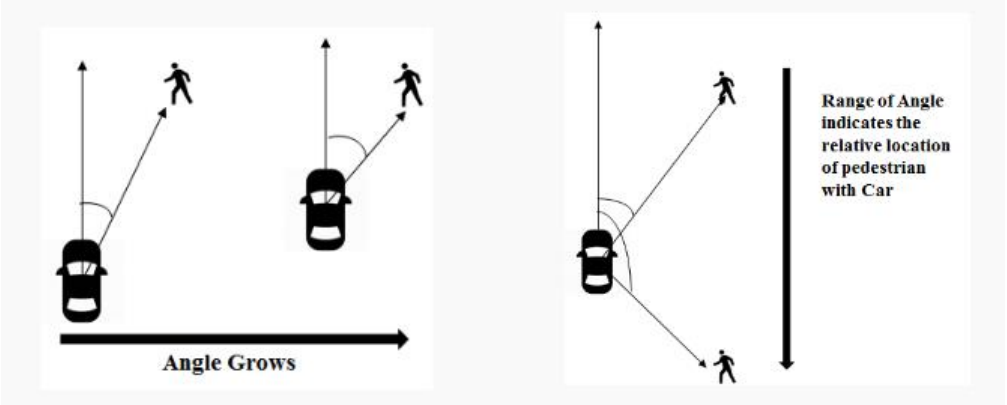

Figure 6: Range of Angle Between Vehicle and Pedestrian Indicates the Relative Location

8 Behavior Prediction Using the Algorithm of Logistic Regression

The first model we are planning to use to model the interactions between jaywalkers and cars is logistic regression (which is a statistical method that allows to analyze datasets where there are multiple independent variables determining an outcome). The decision to use this algorithm was influenced by factors involving time, complexity and practicality. For each training data point, the output is defined as brake or not brake and the features as defined previously. However, we are facing two challenges. First, determining whether we should convert the varying 300+ data points per car into a single data point and if yes, figure out how we should do it without losing valuable information from the raw data. The second challenge is to obtain the pedestrians' data and successfully map them to the corresponding vehicle.

8.1 Machine learning models can be used for pattern recognition in interactions between jaywalkers and vehicles

The second part of our project is constructing the machine learning models allowing the pattern recognition and analysis of the interactions. The first algorithm we used, - as stated previously, is logistics regression because of the simplicity of the code and the possibility to use limited data to get preliminary results. Then, other algorithms such as Support Vector Machine and Decision Tree are exploited to account for non-linearity in the dataset and have better interpretability of our results. The results would be compared to pick the right model for our final solution. Furthermore, we also expect to explore regression and clustering as a solution to our problem as well.

8.2 Drivers' Behavior can be modeled using Logistic Regression

The input to our model would be a matrix of training data containing our features and the output vector.
The features we used are the relative distance in x and y axis and relative speed in x and y axis at each time step and the output file would be a binary variable [0,1] referring to the drivers' actions [No Brake, Brake].
To get our preliminary results, the input only consists of relative distance and speed at a time t. However, in reality, a driver's reaction to a jaywalker includes time steps prior to his braking reaction, in other words, the trajectory of the pedestrian at time steps prior to time t. Our model would then have more input data in terms of features, including times before brake time.
Another consideration is the reaction time of a driver between the intention to brake and the action. We decided to include that reaction time by translating the output vectors in time to correspond to the input data of the intention.
Incorporate with the logistic regression model below

$$h_\theta(x) = g(\theta^{Tx}) = 1/(1 + e^{-\theta^T x}) \tag{1}$$

where h_θ (x) is the hypotheses, the parameters are signified using θ and the mapping of linear functions from X to Y happens using θ The two classes that are predicted are separated by a decision boundary which is θ^{Tx}=0 's solution and if x is one dimension, it will be a point and it will be a line if it has two. In this project, X are vectors of relative speed and distance as discussed above, y is the vectors of brake/not brake label. And the first trial result is shown in Figure4.

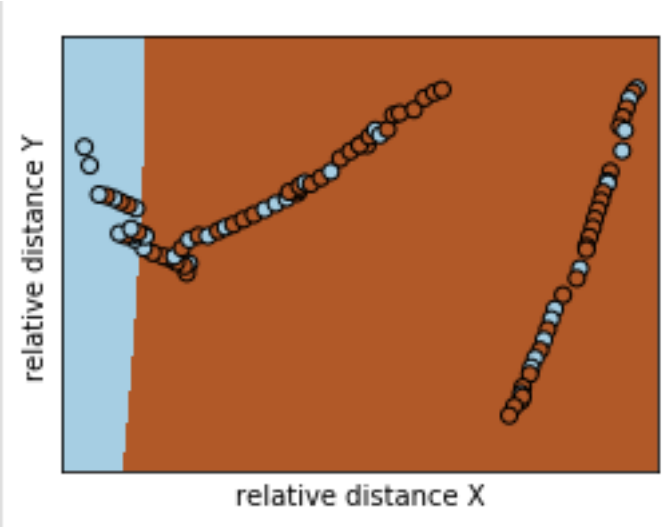

Figure 7: Logistic Regression Classification trial result

8.3 Drivers' Behavior modeled using Random Forest Classifier is more interpretable

The Random Forest Classifier is a tree-based algorithm which splits the samples based on features that give the highest homogeniety in these splits. The advantages of this classifier is that it models non-linear relationships really well. It has a higher interpretability because one can visualize the tree and see how the algorithm classifies the samples at each stage. Further, this algorithm gives us the benefit of extracting the importance of each of the features.

- The full dataset had 8000 samples and three features that were formed out of the feature engineering space
- The response variable has binary values - 0 and 1 where 0 means "Not Brake" and 1 means "Brake"
- A Weighted Random Forest Classifier was used along with oversampling from the minority sample to account for the class imbalance problem (more samples from class 0 as compared to class1)
- The model was tuned using a series of hyperparameters such as number of terminal nodes, minimum impurity for splitting a node, depth of the tree, and maximum number of trees to be created
- The model does well in predicting whether a certain vehicle will not brake with an F-1 score for class 0 of 0.72 but not so well in predicting whether the vehicle will brake given a low F-1 score for class 1 of 0.31
- The F-1 score is the harmonic mean of Precision and Recall

$$F1Score = 2 \times \frac{(Precision \times Recall)}{Precision + recall} \tag{2}$$

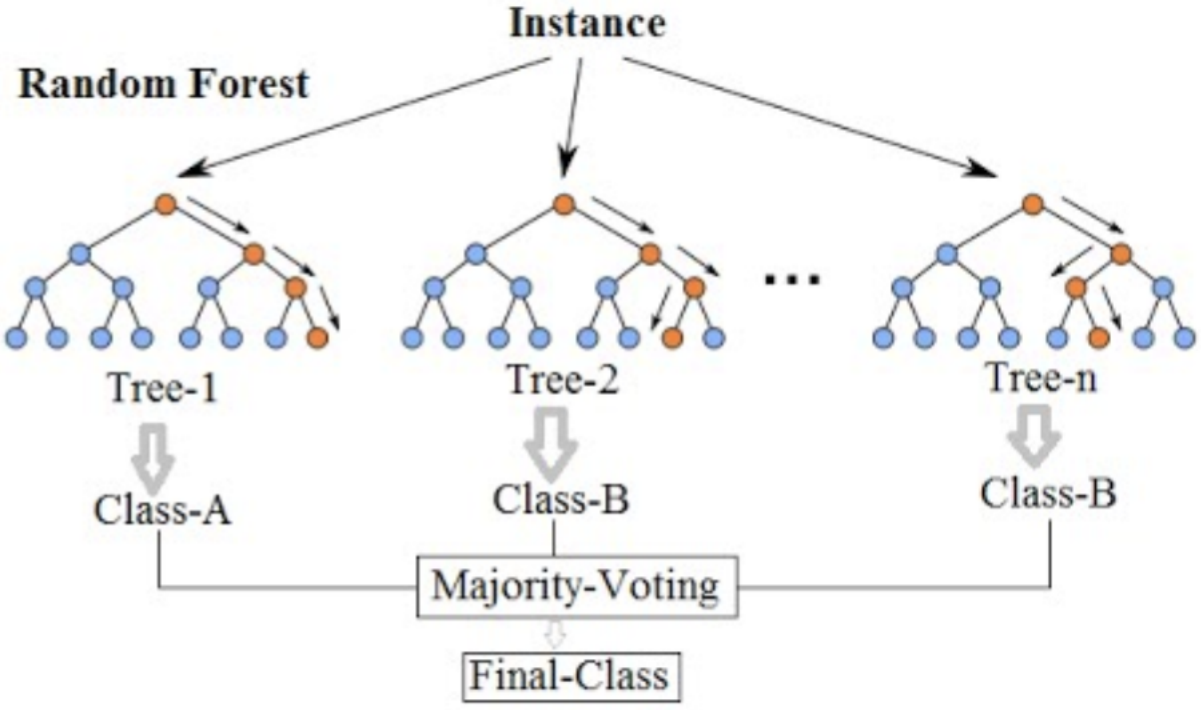

Figure 8: Graph of the overview of Random Forest

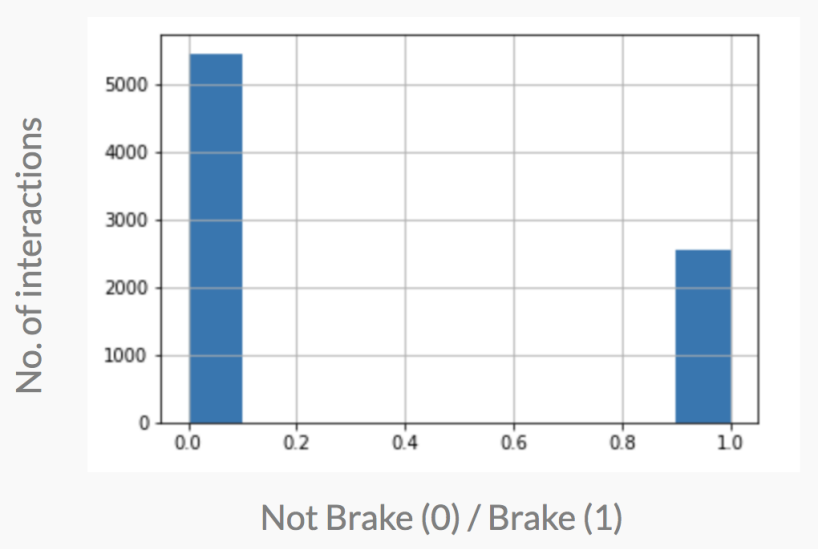

Figure 9: Histogram of our Dataset

- The feature importance as predicted by the model is given below, where ttped is the time for the vehicle to reach the pedestrian, alpha is the angle between the vehicle and the pedestrian, and deltaalpha is the derivative of that angle

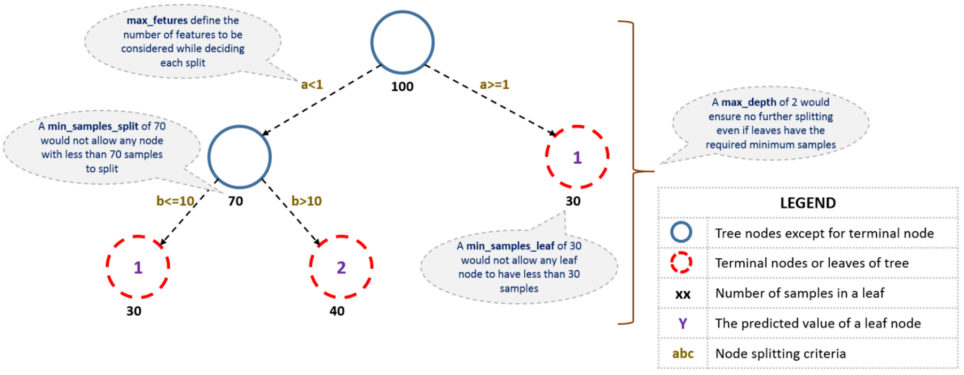

Figure 10: Plot of parameters tunning

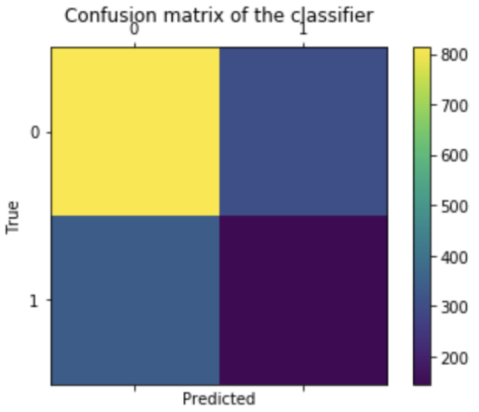

Figure 11: Confusion Matrix of our model

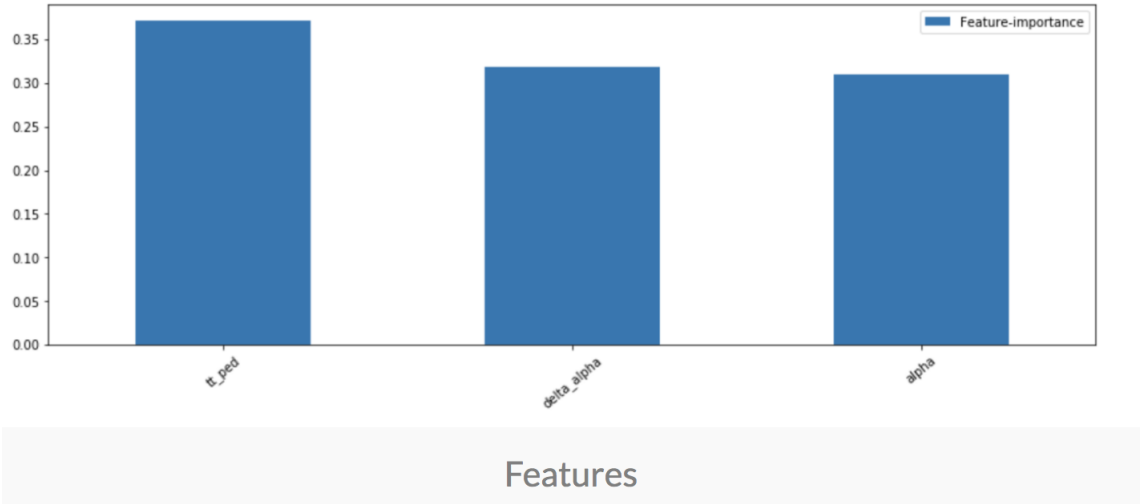

Figure 12: Feature Importance scores using RF

8.4 Drivers' Behavior can also be modeled using a Regression Model

After using classification to model drivers' behavior, we would have a better understanding of the validity of our data and features. We can then build a regression

model which would warrant a solution that is more close to the real world scenarios where the output would be continuous and not discrete such as brake/not brake. The output from these regression models can guide us in implementing control algorithms that can control the speed of the vehicle and constantly modify it based on the trained model.

9 Conclusion

A pipeline starting from data collection and ending with model tuning is built up, so that the future works for developing better models could be done based on this pipeline. This project involved trajectory extraction by utilizing Traffic Intelligence Code by Nicolas Saunier. The extracted trajectories were then used to compute the velocity and acceleration of the vehicles and pedestrians. These features were further manipulated to extract the angle between interacting objects, which include one pedestrian and one vehicle per interaction. From this, rate of change of angle between these objects as well as the time taken for the relative distance between the objects to reach zero is computed. These features were then used to predict whether a vehicle would brake or not brake given certain values for all these features. The two classification algorithms used were Logistic Regression and Random Forest Classifier. The overall results indicated that the models are able to predict correctly when the vehicle is not braking but have a lot of misclassification in predicting whether the vehicle is braking. We believe that this drawback can be overcome in the future course of this project by collecting more data and a balanced one at that having close to equal number of samples for each class.

Further, considering the real world situation, we should not try to model the behavior of every interaction because there can be unsafe and incorrect behaviors. Instead, we should try to separate these extreme behaviors from our dataset and use only the correct and safe behavior to train our model. We can use clustering algorithms such as k-means to detect and label these different behaviors.

References

Jean-François Bonnefon, Azim Shariff, and Iyad Rahwan (2016). The Social Dilemma of autonomous vehicles. Science, 252(62593), 1573-6. doi:10.1126/science.aaf2654.

Do Autonomous Vehicles Spell Doom for Auto Insurance? Frost & Sullivan Market Insight, March 2017.

NHTSA[Press Release].Retrieved from https://www.nhtsa.gov/press-releases/nhtsa-data -shows -traffic-deaths-77-percent-2015.

Li, Q., Chen, L., Li, M., Shaw, S. L., & Nuchter, A. (2014). A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios. IEEE Transactions on Vehicular Technology, 63(2), 540-555.

Glaser, Sébastien, et al. "Maneuver-based trajectory planning for highly autonomous vehicles on real road with traffic and driver interaction." IEEE Transactions on Intelligent Transportation Systems 11.3 (2010): 589-606.

Ismail, K. Application of Computer Vision Techniques for Automated Road Safety Analysis and Traffic Data Collection. PhD dissertation. University of British Columbia, Vancouver, Canada. 2010. http://hdl.handle.net/2429/29546.

Bradski, G., and A. Kaehler. Learning OpenCV: Computer Vision with the OpenCV Library.O'Reilly Media, Inc. 2008. http://oreilly.com/catalog/9780596516130/index.html.

Saunier, N., and T. Sayed. A Feature-Based Tracking Algorithm for Vehicles in Intersections. Presented at Third Canadian Conference on Computer and Robot Vision, IEEE, Quebec City, Quebec, Canada, June 2006.

Saunier, N., T. Sayed, and C. Lim. Probabilistic Collision Prediction for Vision-Based Automated Road Safety Analysis. Presented at 10th International IEEE Conference on Intelligent Transportation Systems, Seattle, Wash., 2007.

Beymer, D., McLauchlan, P., Coifman, B., & Malik, J. (1997, June). A real-time computer vision system for measuring traffic parameters. In Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on (pp. 495-501). IEEE.

Dubrofsky, E. Homography Estimation. The University of British Columbia, Vancouver, March 2009.