Detecting Drowsy Drivers Using Machine Learning Algorithms

Majid Aksari
University of Wisconsin-Madison
majid@cs.wisc.edu

ABSTRACT

Driver drowsiness contributes to many car crashes and fatalities in the United States. Machine learning algorithms have shown to help in detecting driver drowsiness. We try different machine learning algorithms on a dataset collected by the NADS-1 [1] simulator to detect driver drowsiness. We generate different aggregation features on the time-series data and evaluate the performance of the feature sets. We find that aggregation at the more fine-grained "per-event" level improves the outcome of classification compared to the high-level "per-run" aggregation.

1. INTRODUCTION

National Highway Safety Administration reports that almost 36,061 fatal vehicle crashes happen each year in United States. Alcohol intoxication, distraction and drowsiness have a considerable proportion in these crashes (by respectively 31%, 29%, and 2.5%) [2, 3]. However, lack of physical traceability of distraction and drowsiness cause their roles to be underestimated.

Each year more than 80,000 crashes occur in Nation's highways because of drowsy driving and result at least 850 fatalities and many injuries.
Using data from the 100-car naturalistic study [4], researchers have found that drowsy driving is the reason of 22% to 24% of crashes and near-crashes observed.

National Sleep Foundation's 2012 annual Sleep in America survey shows that 20 percent of workers had driven drowsy at least once per month in the past year [5]. Approximately 40 percent of train operators said they have driven drowsy at least once in the past month. More than 25 percent of those who had driven drowsy had fallen asleep. Studies show that sleepiness can impair driving performance as much as or more than alcohol [6].

These facts and their consequences are reasons for concern. Saving lives lost due to drowsy driving needs that research in this field is continued. Prior research in alcohol impairment detection shows that the machine learning algorithms can help to prevent and reduce these crashes and their consequences. The difference between an impaired driver from an unimpaired driver can be recognized reliably by these algorithms with use of driving performance metrics based on signature pattern of lane position and steering.

In this work, given a set of driving runs by drowsy and non-drowsy drivers we try to detect the drowsy drivers.

2. RELATED WORKS

The most popular algorithm for detecting drowsiness is PERCLOS. This algorithm is developed by Wierwille et al. [7]. PERCLOS measures percentage of time that eyes are closed over a window. The window can be as short as 1 minute. Experiments have shown that PERCLOS can achieve accuracy of 90% in simulator studies and in practice.

McDonald et al. [8] develop a random forest model using the steering wheel angle measurement to detect drowsiness related lane departures. This random forest algorithm has higher accuracy and Area Under the receiver operating characteristics Curve (AUC) than the commonly used PERCLOS algorithm.

In their later work, McDonald et al. [3] use a Hidden Markov model to detect drowsiness in time-series data. They find that using this model they can significantly increase the detection ability of the classifier, and reduce the rate of false positives.

3. DATA SET

3.1 Data Collection

Data collection was done by the NADS-1 driving simulator [2]. Drivers were volunteers from different age groups. They were asked to drive in different types of roadways (urban, freeway, rural) in different times of day.

The simulator can be seen in the below figure:

In each run the driver drives through a pre-defined route, and certain characteristics of the driver and the car are measured.

Some of these measures are:

- Accelerator pedal position
- Brake pedal position
- Steering angle
- Speed
- Lane deviation
- Eye closure fraction
- Percent of 2 minute window where the eye is more than 80% closed.

3.2 Our data set

We were not involved in the data collection process. The dataset that we use for this project is composed of different runs of driving. Each run is stored in one file.

Each run has 27 columns and about 150000 rows which correspond to measurement at different point in times of the drive. Thus, the data is time-series data.

Our data consists of 144 runs. 72 runs of them are during the day, and 72 of them are during late night.

3.3 Ground Truth

We consider 2 runs for each driver. One is during the day and one is during the late night. The experiments are done such that the driver is awake all day before performing the late night experiment. Therefore, we assume that the drivers were drowsy at the late night run.

We understand that this may be a strong assumption, and it can affect our results, but based on our discussion with the person who supplied the data we found that this is a reasonable assumption for this project's purpose. We will address more fine-grained drowsiness detection in the future work.

3.4 The Classification Task

Based on the above data set and the way we define the ground truth, the classification task is to find the runs where the driver is drowsy; i.e. the late night runs. Thus, we will use supervised learning with 2 classes: drowsy, not drowsy.

4. Experiments

In this section, we will explain different experiments that we did with the dataset generated above, and report their results. We will explain the implications of the results after each of the experiments.

All the experiments were done with Weka [9], which is a collection of machine learning algorithms for data mining tasks.

We tried a variety of machine learning algorithms from the different packages available in Weka and will report the results.

4.1 Algorithms

Our classification task is suitable for supervised learning because the output classes are known. The machine learning algorithms that we tried are as follows:

- Bayes -> Naïve Bayes
- Functions -> Logistic regression, SMO
- Trees -> J48, Random Forest

This variety allows us to compare performance of different classes of machine learning algorithms.

4.2 Attribute Selection

For each experiment, we ran the algorithm with and without attribute selection and compare the results.

For AttributeSelectedClassifier, we used InfoGainAttributeEval as evaluator and Ranker for search. We limited the number of features to 100 out of 1900 for per-event feature set and 10 out of 76 for per-run feature set (described below).

5. Data Pre-processing

5.1 Removing Columns

We removed the following columns: Run name, frames, and event_id. These columns where identifiers and did not provide any value for classification.

We found that we should remove these columns after viewing the attributes selected by Weka. When we chose the AttributeSelectedClassifier, it ranked these columns high. This meant that even though they do not contain valuable information, the machine learning algorithms found a correlation between these columns and the run class.

It was interesting for us to note that the quality measures degraded after removing these features. It is clear that these identifiers where giving extra information to the machine learning algorithms, but we had to remove them because they were not relevant to the classification task.

5.2 Missing Values

In the case of per-event aggregate features, some of the runs did not have all the events. Therefore, we had missing value for some of the features. It appeared that those events were not significant in the classification, but we did not want to remove the events from the data. Also, we did not want Weka to try to fill those missing values for us. Therefore, after discussing with the person who we obtained the data from, we agreed that it is reasonable to put 0 for the missing values.

6. Features

Due to the time-series nature of data, we need to somehow aggregate the values in the time-series to generate features for this data set. This can be done in many ways. Therefore, we created 2 different set of features:

1. Per-run aggregate features
2. Per-event aggregate features

Below you can find the feature description and the experiment results for each set of features.

6.1 Per-run aggregate features

The most simple set of features is to aggregate all the values for each column in the run. For each column, we generated 4 features:

- Mean of all values in the column
- Minimum of all values in the column
- Maximum of all values in the column
- Standard deviation of all values in the column

For example, for the speed column, the mean, min, max, and std. of the speed in the run would be 4 different features. This totals to 76 features for this feature set.

The above 4 features for each column are generated based on the hypothesis of how a drowsy person "may" act. We do not have the exact answer but the hope is that the machine learning algorithms can find it out.

For example, for the speed column one could hypothesize that if a person is drowsy he may drive slower/faster in average (mean). He may have very slow/fast speeds in cases where he falls asleep (min, max). Or he may deviate from his average speed because of his impairment (standard deviation).

6.2 Per-event aggregate features

Each run is composed of a set of events. For example, the driver first pulls out of a driveway, then drives in an urban area, then enters the highway and so on. There are certain events where the driver is more prone to become drowsy. For example, we hypothesize that when driving in the highway or driving in the straight rural road which are more boring experiences, it is more likely to become drowsy. Also, one would hypothesize that the driver will be more likely to get drowsy in the later part of the trip when he gets close to the destination.

With the per-run aggregate features, this information is lost and not available to the machine learning algorithms. Therefore, we introduce a new set of features: Per-event aggregate features.

For this feature set, we aggregate the values per event in each column. This means that we generate these 4 features per-event per-column:

- Mean of all values in the event in column
- Minimum of all values in the event in column
- Maximum of all values in the event in column
- Standard deviation of all values in the event in column

There are 25 total events and 19 columns. This totals to 1900 features in the dataset.

7. Results

7.1 Testing

10 fold cross validation was used for testing. Data rows were shuffled to remove any possible meaningful ordering.

7.2 Metrics

We report precision, recall, F1 measure, as well as the area under the ROC curve (named as ROC area) for each of the experiment settings below. These measures are reported for detecting when the driver is drowsy. Detecting when the driver is not drowsy is not our focus for this classification task.

For precision and recall generally the higher measures are better. However, there is tradeoff between high precision and high recall. Generally with improving recall, precision is compromised.

High precision and low recall means that we will have low number of false positives and high number of false negatives. False negatives are not desired because for our task if we miss detecting drowsiness, we may endanger the life of the driver.

On the other hand, high recall and low precision means that there are many false positives, but we are less likely to miss detecting a drowsy driver. This is also not desirable for our task because high number of false positives will cause the driver to ignore the warnings.

The area under an ROC curve is a robust indicator of how well the classifier is able to distinguish between the drowsy and not drowsy classes. An area of 0.5 means that the algorithm is behaving like tossing a coin for detecting the classes. A rough guide for the goodness of classifier based on the ROC area is [10]:

- 90-1 = excellent (A)
- .80-.90 = good (B)
- .70-.80 = fair (C)
- .60-.70 = poor (D)
- .50-.60 = fail (F)

7.3 Per-run aggregate features

For these features time-series data for each column was aggregated over the whole run.

Without Attribute Selection

Algorithm	Precision	Recall	F1	ROC
Naïve Bayes	0.65	0.542	0.591	0.68
Logistic Regression	0.611	0.611	0.611	0.644
SMO	0.687	0.639	0.662	0.674
J48	0.628	0.681	0.653	0.635
Random Forest	0.636	0.389	0.483	0.656

With Attribute Selection (10 out of 76)

Algorithm	Precision	Recall	F1	ROC
Naïve Bayes	0.662	0.597	0.628	0.658
Logistic Regression	0.662	0.708	0.685	0.713
SMO	0.662	0.597	0.628	0.658
J48	0.523	0.472	0.496	0.523
Random Forest	0.623	0.597	0.61	0.67

Discussion

For the per-run aggregate features without attribute selection, SMO is doing better than all other approaches with 0.66 F1 score and an ROC area of 0.67. With this ROC are this classifier is performing poorly with a D score.

Selecting 10 attributes out of 76 improves the performance of Logistic regression and Random Forest, but hurts the other classifiers. However, now the best area under ROC curve is for the logistic regression (0.71) which means that it is doing a fair job in distinguishing between drowsy and non-drowsy drivers.

It is interesting to look at the selected attributes:

1 Steering_Angle_Rate_std
2 Steering_Angle_Rate_max
3 Lane_Departure_std
4 Perclos_mean
5 Perclos_min
6 Lane_Deviation_4_std
7 Time_std
8 Lane_Departure_max
9 Lane_Departure_min
10 Perclos_max

We noted that the selected attributes include Steering Angle (std and max) and all the Perclos measures which from previous works we knew that they are very important in this classification task. The Perclos attributes measures percent of 2 minute window where the eye is more than 80% closed. So it is intuitive that it would be a good indicator of drowsiness.

7.4 Per-event aggregate features

With the per-run aggregate features, the best we get is a fair classifier based on the area under the ROC curve. In this section we try to improve on the per-run aggregate features by using more fine-grained features that take into account different events in a run.

Without Attribute Selection

Algorithm	Precision	Recall	F1	ROC
Naïve Bayes	0.553	0.361	0.437	0.596
Logistic Regression	0.722	0.722	0.722	0.763
SMO	0.718	0.708	0.713	0.715
J48	0.606	0.597	0.601	0.609
Random Forest	0.571	0.722	0.638	0.607

With Attribute Selection (100 out of 1900)

Algorithm	Precision	Recall	F1	ROC
Naïve Bayes	0.687	0.639	0.662	0.703
Logistic Regression	0.652	0.625	0.638	0.723
SMO	0.797	0.764	0.78	0.785
J48	0.675	0.722	0.698	0.707
Random Forest	0.619	0.722	0.667	0.705

Discussion

Even without attribute selection the per-event aggregate features are performing better than the per-run aggregate features. This is because more information is available to the algorithms and we are not eliminating useful information by averaging over different events.

"Without" attribute selection, logistic regression performs best with a 0.72 F1 score and 0.76 ROC area. This is compared to 0.68 F1 and 0.71 ROC area for per-run features "with" attribute selection. This shows that per-event aggregate features give us better performance.

With selecting 100 attributes out of 1900, SMO is the winner with 0.78 F1 and 0.78 ROC area. This is a slight improvement over without attribute selection, but is a major improvement over the best of the per-run aggregate features. This classifier is close to getting a B (Good) score for the area under ROC curve.

It is interesting to note that in all scenarios, SMO and logistic regression which find a function over the attributes are performing better than other algorithms.

Also, it is interesting to note that Random Forest which is a collection of trees has the same ROC area as the J48 which builds a single tree.

Without attribute selection Naïve Bayes has the poorest performance of all, with ROC area of 0.59 which gets a fail score. It may be because it has a hard time in filtering out the bad attributes. This hypothesis is verified because when selecting attributes, it significantly improves to an ROC area of 0.7 which is a fair classifier and stands by J48 and Random Forest.

Note that this is not the case for the per-run aggregate features. Naïve Bayes is actually hurt when selecting attributes. This may be because there are only 76 features in the per-run aggregate features, whereas here we have 1900 features to choose from.

Let's look at the top 10 features selected for this feature set:

1 Speed_311_std
2 Brake_Pedal_Pos_307_max
3 Acc_2_311_max
4 Acc_1_307_min
5 Perclos_101_mean
6 Acc_2_311_min
7 Lane_Deviation_2_102_mean
8 Acc_2_311_std
9 Brake_Pedal_Pos_307_std
10 Lane_Deviation_4_306_mean

Note that among the selected attributes, we can see Brake Pedal position, Speed standard deviation and again Perclos (precent eye closure) attributes. We knew from previous works that these attributes are important.

But the more interesting fact among the selected attributes is that many of them refer to the event id 311. This event id, refers to driving a in a straight rural road, which is boring for the driver and we knew from previous works that drivers may get drowsy in this

section. Also, this part of the route is towards the end of the trip, and thus it is more likely that the driver shows drowsiness signs in this road.

Selecting features corresponding to the straight rural road event (311) shows the effectiveness of the Per-event aggregate features. It shows that in fact we are giving the algorithms more information for detecting drowsiness, and by aggregating over all the events we will lose this information. Also, we get more intuitive output from the algorithms.

8. Future Work

In this work, for both of the features sets, we used simple mean, min, max, and standard deviations for our aggregation functions. It remains future work to try other aggregate functions such as Fourier transforms and wavelets to see if they can improve classification.

For the ground truth, we assumed that if the driving is done at late night, then the driver is drowsy. This is a strong assumption and may not be the case in the real world. A driver may be drowsy at parts of the run and awake on other parts of the run. Therefore, for future work, we would like to try and detect parts of the run where the driver is drowsy. This is a more challenging task and requires more complicated features.

9. Conclusions

In this work, we tried to detect drowsy drivers using supervised machine learning algorithms. Because of the time-series nature of the data, we had to do aggregation over the time-series to generate features. We tried to generate aggregate features in the granularity of per-run and per-event in each run. We found that the per-event aggregate features result to better classifiers with respect to area under ROC curve. The per-event aggregate features are also more informative and intuitive when analyzing the selected attributes from the dataset. Also, function-based classifiers such as Logistic Regression and SMO performed better than Bayes and Trees for this classification task.

10. References

1. Brown, T., Lee, J., Schwarz, C., Dary Fiorentino, D., & McDonald, A. (2014, February). Assessing the feasibility of vehicle-based sensors to detect drowsy driving. (Report No. DOT HS 811 886). Washington, DC: National Highway Traffic Safety Administration. Available at: http://www.nhtsa.gov
2. National Highway Traffic Safety Administration. Advanced Countermeasures for Multiple Impairments, February 2014, Available at: http://www.nhtsa.gov
3. McDonald, Anthony D., Schwarz, Chris, Lee, John D., Brown & Timothy L., 2013, Impairment as a hidden state: How Hidden Markov Models improve drowsiness detection and may differentiate between distraction, drowsiness, and alcohol impairment
4. Vicki L. Neale, Thomas A. Dingus, Sheila G. Klauer, Jeremy Sudweeks, Michael Goodman, An overview of the 100-car naturalistic study and findings
5. National Sleep Foundation. 2012 Sleep in America Poll, Available at: www.sleepfoundation.org
6. National Sleep Foundation (NSF). Consequences of Drowsy Driving, Available at: www.sleepfoundation.org
7. Wierwille, W., Ellsworth, L., Wreggit, S., Fairbanks, R., & Kim, C. (1994). Final Report: Research on vehicle-based driver status/performance monitoring: development, validation, and refinement of algorithms for detection of driver drowsiness. Washington DC.
8. McDonald, Anthony D., Schwarz, Chris, Lee, John D., Brown, Timothy L., Real-Time Detection of Drowsiness Related Lane Departures Using Steering Wheel Angle, Proceedings of the Human Factors and Ergonomics Society 56th annual meeting, 2012
9. Weka, available at: http://www.cs.waikato.ac.nz/ml/weka/
10. The area under an ROC curve: http://gim.unmc.edu/dxtests/roc3.htm