Predictive Modeling Using Telematics

Robert Bear, Joseph Marker, Kailan Shang and Hai You

ReservePrism

May 2016

Introduction

With the fast growth of applying telematics to usage based insurance (UBI) and driver behavior analysis, risk assessment and pricing of auto insurance has become more complex but also rewarding. Telematics[1] provides details of driving trips such as the longitude and latitude of the vehicle every few seconds. Predictive models have been widely used in the pricing, reserving and risk quantification of insurance products. Even though the non-traditional geolocation data collected from telematics have different forms from those required for predictive models, they can be transformed and adjusted for meaningful predictions of driving behaviors. With more accurate and personalized analysis of driving behaviors, the risk assessment of the auto insurance business is more thorough and credible.

This article shows a simplified example of using a random forest model to identify trips that are not driven by the insured driver. Identifying different drivers of a vehicle can help us understand the usage of the vehicle by different drivers. This is useful for risk assessment and auto insurance pricing such as setting UBI rates and discounts for good driving behaviors. Core R codes are also provided below for education purpose.

Data

Driving trips represented as the coordinates of the vehicle position every second are provided in the dataset. The data was made available through a Kaggle competition[2]. Each driver has hundreds of driving trips with few of them not driven by the driver. This is an unsupervised learning problem because the trips that are not driven by the driver are not identified but we know that the number is small. Table 1 shows part of a driving path.

Table 1. Sample Driving Path Data

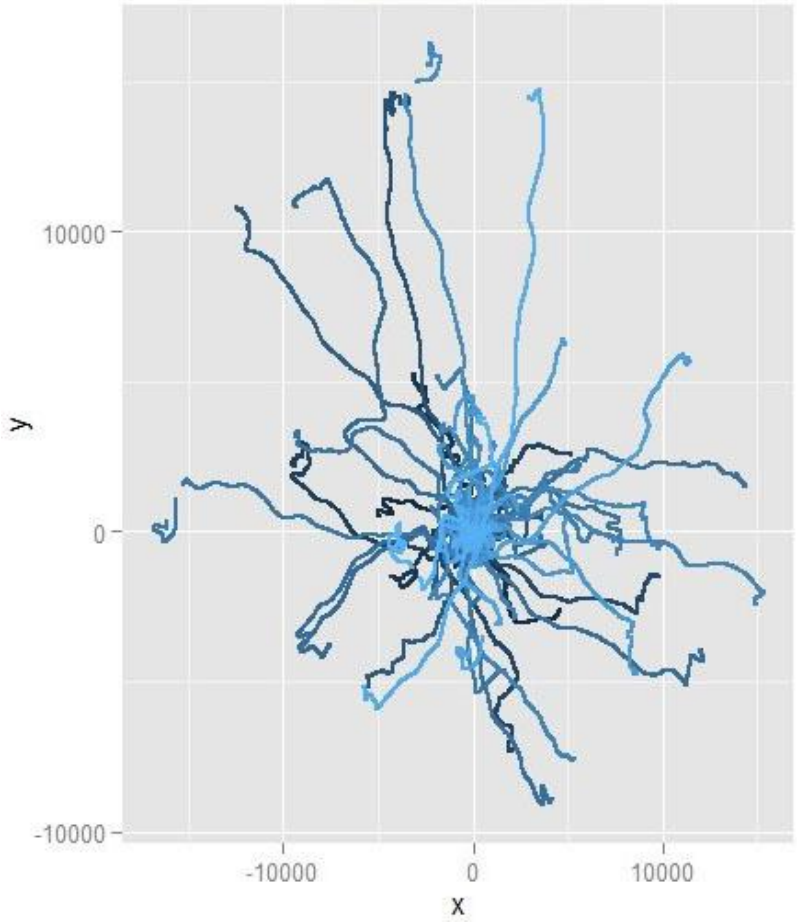
X	y
0	0
-7.4	-7.5
-15	-14.8
-22.6	-22.1
-29.9	-28.9
-37.7	-35.3
-44.7	-42.3

[1] https://en.wikipedia.org/wiki/Telematics
[2] https://www.kaggle.com/c/axa-driver-telematics-analysis

-51.5	-49.6
…	…

Each row contains the coordinates of the vehicle with the starting point as (0,0). For example, one second later, the vehicle moved to (-7.4, -7.5), which is 7.4m south and 7.5m west of the starting point. In this way, actual longitude and latitude are normalized to remove sensitive information. Figure 1 illustrates the driving paths of a driver.

Figure 1. Sample Driving Paths



Data Processing

The dataset only contains geolocation information that is hard to use directly in predictive models. The first step is to extract features from the data that can be used. The following features are extracted and used in the example and are denoted as explanatory variables X. The explained variable Y is a categorical variable that describe whether a driving trip in a driver's dataset was driven by the driver. In practice, other features can be created from geolocation data to solve other problems.

Table 2. Features Built on Geolocation Data

Feature	Explanation
Time	The time of a trip, by counting the number of rows in a trip file.
Speedavg	Average speed
Speedvol	Standard deviation of the speed
Speedmin	Minimum speed
Speedmax	Maximum speed
Speed10	10th percentile of speed
Speed30	30th percentile of speed
Speed70	70th percentile of speed
Speed90	90th percentile of speed
accelerationavg	Average acceleration
accelerationvol	Standard deviation of the acceleration
accelerationmin	Minimum acceleration
accelerationmax	Maximum acceleration
Acceleration10	10th percentile of acceleration
Acceleration30	30th percentile of acceleration
Acceleration70	70th percentile of acceleration
Acceleration90	90th percentile of acceleration
Nofast	Number of accelerations greater than 2
Noslow	Number of decelerations less than -2
segdistanceavg	Average distance per second
segdistancevol	Standard deviation of the distance
segdistancemin	Minimum distance
segdistancemax	Maximum distance
Segdistance10	10th percentile of distance
Segdistance30	30th percentile of distance
Segdistance70	70th percentile of distance
Segdistance90	90th percentile of distance
Length	Length of the trip (average speed * time)
Indicator (Y)	Whether the trip belongs to the driver. At initial, it is assumed that the 200 trips in a driver's folder all belong to the driver.

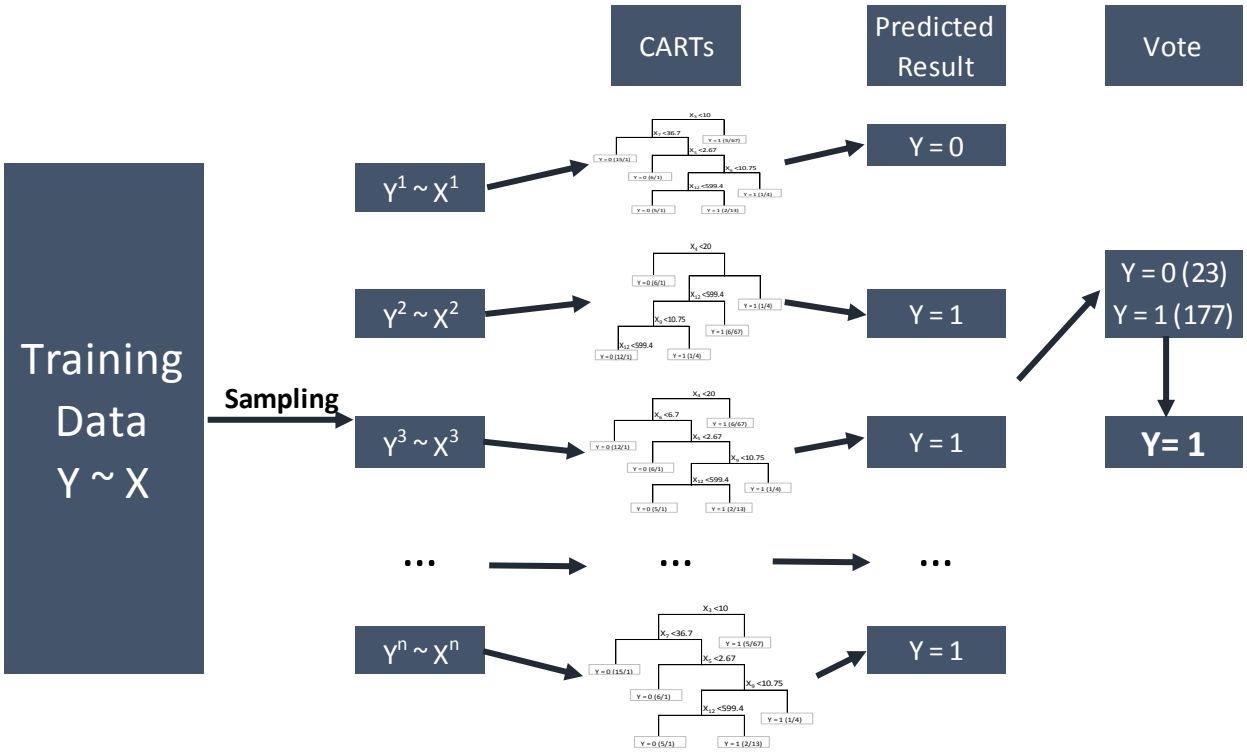
Random Forest Model

A random forest model is an ensemble model that has a collection of classification and regression trees (CARTs)[3] using random inputs. For each CART, a bootstrap sample of the data is used to train the model. The bootstrap sample is created randomly with replacement from the dataset. With these trained trees, the final result is based on the majority vote of these trees. Figure 2 illustrates the basic structure of a random forest model. N smaller subsets of the training data are sampled and each subset is used to train a CART. In the example of driving trips, the CART is a classification tree with the new features extracted from geolocation data used in splitting the tree at each node. Terminal nodes without further splitting shows the most frequent value of Y which is whether the driving trip was driven by the driver.

[3] A good introduction of CART with examples and R codes can be found at http://www.stat.cmu.edu/~cshalizi/350/lectures/22/lecture-22.pdf.

With the trained CARTs, Y is predicted for new driving trips with Y = 0 meaning that the trip was not driven by the driver and Y = 1 meaning that the trip was driven by the driver. The final conclusion of whether a driving trip was driven by the driver is based on the majority of the predicted results. In this illustration, Y = 1 is the final model output for a specific driving trip.

Figure 2. Random Forest Model Structure



Random forest models normally improve the prediction accuracy by reducing the variance of the predictors. Other predictive models such as generalized linear models and artificial neural network models can also be used to predict whether a trip belongs to a driver using the features built on geolocation data.

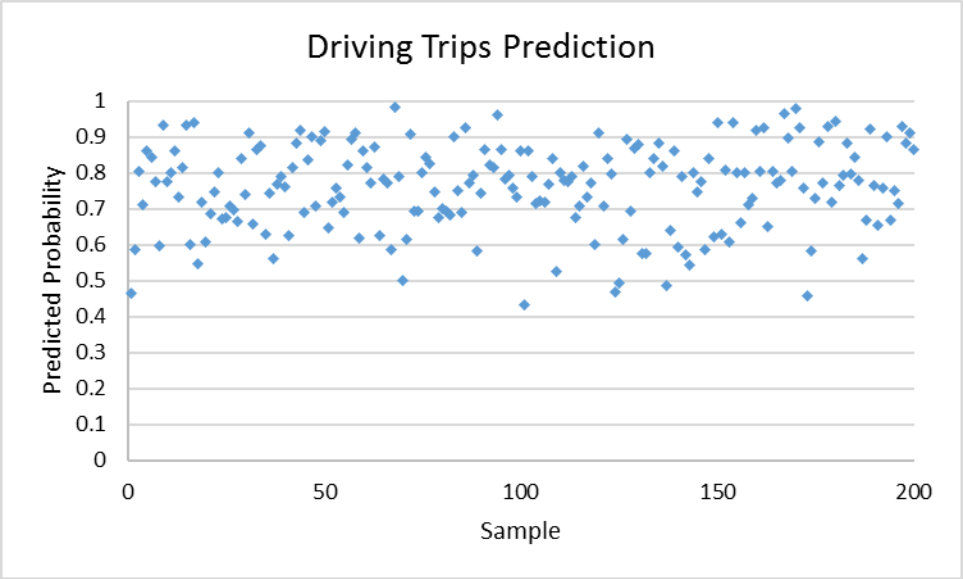
Prediction

In a real business world, for the driving trips of a specific driver, how many and which trips were driven by other drivers are unknown, at least at the early stage of the implementation. The geolocation data is normally collected from a hardware such as a dongle installed in the vehicle or a mobile app that records the driving path. Therefore, it is an unsupervised learning problem which is rare in the actuarial and risk management area. Most predictive models are not useful for unsupervised learning. To solve this problem, we need to transform the unsupervised learning problem into a supervised learning problem. Each trip used for model calibration needs to be labeled as either driven by the driver (right trips) or not (wrong trips). Each driver's dataset contains trips most of which were driven by the driver and they are all labeled as right trips. To include some wrong trips in the training data, 500 trips were randomly drawn from other drivers' dataset and added to the driver's dataset. The added trips from other drivers' dataset were very unlikely to be driven by the driver and they are labeled as wrong trips.

The unsupervised learning problem became a supervised learning problem with few errors in the training set.

The random forest model introduced above was used to predict the probability that a trip belongs to a driver. The process was repeated multiple times by redrawing the wrong trips randomly and then calibrating the random forest model. Average probability based on all the fitted models results was used to make the final prediction. If the average probability is less than 0.5, it was concluded that the trip does not belong to the driver. Figure 3 shows the prediction result for a driver's dataset. Out of 200 trips, 6 trips have a predicted probability less than 0.5, which means that the trips do not belong to the driver. The result is reasonable considering that we assume all the 200 trips belong to the driver during the model training.

Figure 3. Sample Driving Trips Prediction



Validation

The validation also needs some non-traditional adjustments to address the issue of unsupervised learning. The calibrated model of a driver can be used to test some random trips by other drivers. For example, 200 trips can be drawn from other drivers' datasets to see how many of them will be mistakenly predicted to be driven by the driver rather than the other drivers. A low rate of correct prediction would indicate a low accuracy of the calibrated model. However, it cannot test how accurate the model is for predicting the trips that were driven by the driver. A better approach is to split the driver's dataset into training dataset and validation dataset. For example, the training dataset can be composed of 80% of trips in the driver's dataset and 500 trips of other drivers. The validation dataset can be composed of the remaining 20% trips and 200 trips of other drivers. Using the model calibrated based on the training dataset, the trips in the validation dataset can be predicted as either right or wrong trips. The model can then be tested against not only its power to detect wrong trips but also its power to discover right trips. Table 4 shows the structure of the confusion matrix for validating the model.

Table 4. Confusion Matrix for Model Validation

Total Trips	Predicted Right Trips	Predicted Wrong Trips
Actual Right Trips	True Right Trips	False Wrong Trips
Actual Wrong Trips	False Right Trips	True Wrong Trips

With the actual confusion matrix, a few measures can be calculated to assess the accuracy of the prediction. Precision rate measures the Type I error and recall rate measures the Type II error. F-measure is the harmonic mean of precision and recall and can be considered as an aggregate measure of Type I error and Type II error.

Table 5. Sample Confusion Matrix for Model Validation

Total Trips	Predicted Right Trips	Predicted Wrong Trips
Actual Right Trips	43	7
Actual Wrong Trips	15	185

$$Precision = \frac{True\ Right\ Trips}{True\ Right\ Trips + False\ Right\ Trips} = \frac{43}{43+15} = 74.1\%$$

$$Recall = \frac{True\ Right\ Trips}{True\ Right\ Trips + False\ Wrong\ Trips} = \frac{43}{43+7} = 86\%$$

$$F - Measure = 2 \times \frac{Precision \times Recall}{Precision + recall} = 2 \times \frac{74.1\% \times 86\%}{74.1\% + 86\%} = 79.6\%$$

A disadvantage of this approach is that the training dataset will have less data of right trips. Given that each driver has a small dataset, it may significantly affect the credibility of model calibration.

Conclusion

Predictive modeling using geolocation data collected through telematics requires some non-traditional treatments such as feature extraction, transformation from unsupervised learning to supervised learning, and small-scale validation. With a little creativity, predictive models used for traditional actuarial and risk management problems can be applied to analyzing geolocation data as well.

Core R Codes

R codes used to implement the random forest model are listed below.

```
### R package "randomForest" is used.
library(randomForest)
## Create the formula for random forest models
#FeaturesDataset is the dataset with all the new features extracted from the geolocation data, as shown in Table 2.
Xnames <- colnames(FeaturesDataset)

#Remove driver no and trip no which are not useful for prediction and remove Y (indicator) from the list of explanatory variables
Xnames <- Xnames[!Xnames %in% c("driverno","trip","indicator")]

#Set the formula
f<-as.formula(paste("as.factor(indicator)~",paste(Xnames,collapse="+")))
```

```
#add randomly chosen trips from other drivers to the driver's existing trips to facilitate the training.
AddWrongTrips = function(driver, NoWrongtrips){
        Correcttrips = FeaturesDataset[which(FeaturesDataset[,'driverno']==driver), ]
        Wrongtrips = FeaturesDataset[sample(which(FeaturesDataset[,'driverno']!=driver), size = NoWrongtrips), ]
        Wrongtrips[,'indicator'] = 0
        Trainingdata = rbind(Correcttrips, Wrongtrips)
        return(Trainingdata)
}

# Using random forest to predict the wrong trips out of the 200 trips for each driver. The data used is the 200 trips + randomly chosen wrong trips。The program will run 10 times so that the different set of wrong trips will be used in the training. The results of the 10 trainings are used to predict the prob. of being a wrong trip.
NoWrongTrips = 500

# The predicted probability for a driver
TripProb = numeric(NoTrips)

# The predicted probability for a driver for all iterations
TrainProbIter = matrix(data=NA, nrow=NoTrips, ncol=Iteration)

# The predicted probability for all drivers in a vector
TripProbTotal = numeric(NoTrips * NoDrivers)

# Random forest model calibration for all drivers and all iterations
for (driver in DriveData){
        TrainData = AddWrongTrips(driver, NoWrongTrips)
        rfmodel = randomForest(f , ntree = 50, nodesize=5, importance = TRUE,  data=TrainData)
        rfpred = predict(rfmodel, newdata = TrainData[1:200, ], type = "prob")[,2]
        TrainProbIter[,iter] = rfpred

        TripProb=apply(X=TrainProbIter, 1, mean)
        indicator_pred=ifelse(TripProb>=0.5,1,0)
        FeaturesDataset[seq((i-1)*200+1, i*200 ),"indicator"]=indicator_pred
        TripProbTotal[seq((i-1)*200+1, i*200 )]=TripProb
}
```