STATS769 Lab02

Yujie Zhang 130011770 yzhb915

August 11, 2019

The data are trips on electric scooters and bikes in Austin, Texas.The data came in the form of three CSV files, one per month. Each file contains 5000 trips.The following code reads the CSV files into R and combines them to create a single data frame.

1.The following code imported the eleven CSV files into R and combine them into a single data frame.

```
file_path="/course/Labs/Lab02/"
myfiles = list.files(path=file_path, pattern="trips-20", full.names=TRUE)
trips <- do.call(rbind, lapply(myfiles, read.csv))
```

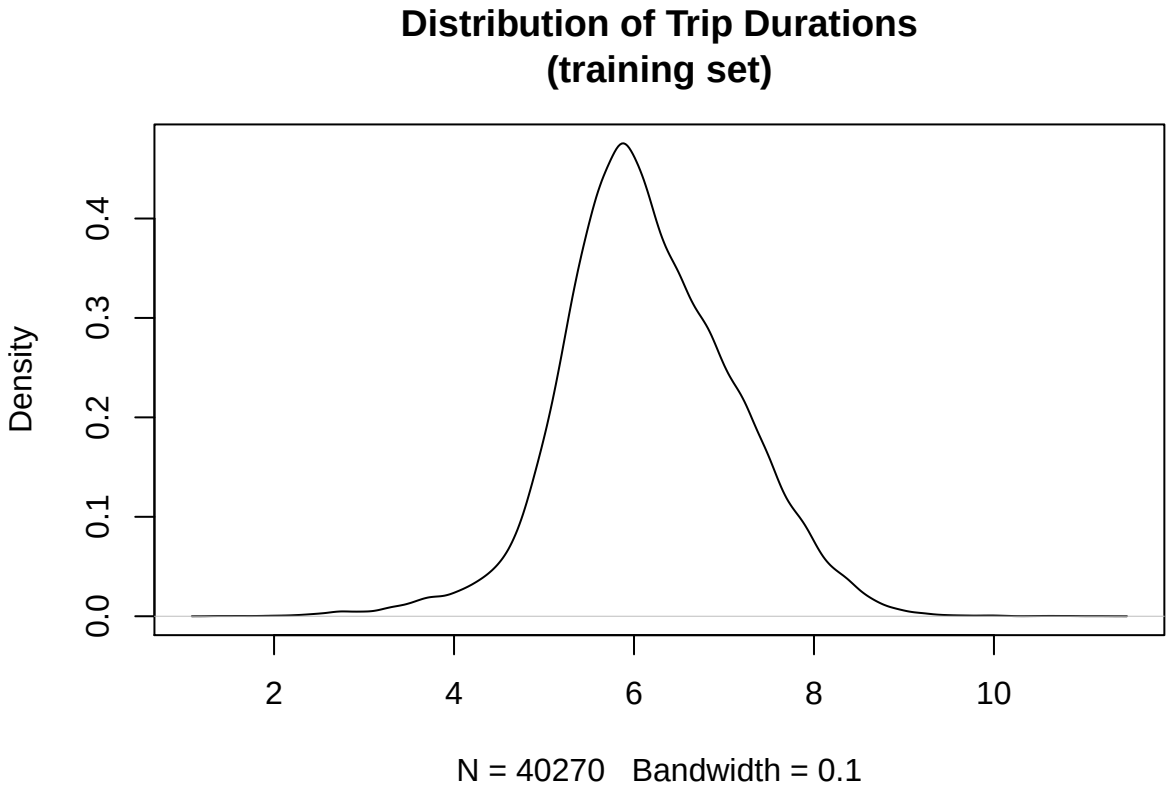
2.A test set was generated by selecting 1000 rows at random from each month. The following code extracted a subset of 1000 rows from each month to use as a test set (a total of 11,000 rows); the remaining 44,000 rows are the training set.

```
testIndex <- unlist(lapply(1:11,function(i) {sample(1:5000, 1000) + (i - 1)*5000}))
test_data = trips[testIndex,]
train_data = trips[-testIndex,]
```

Exploratory plots of trip durations and trip distances shows they are reasonably unimodal and symmetric.

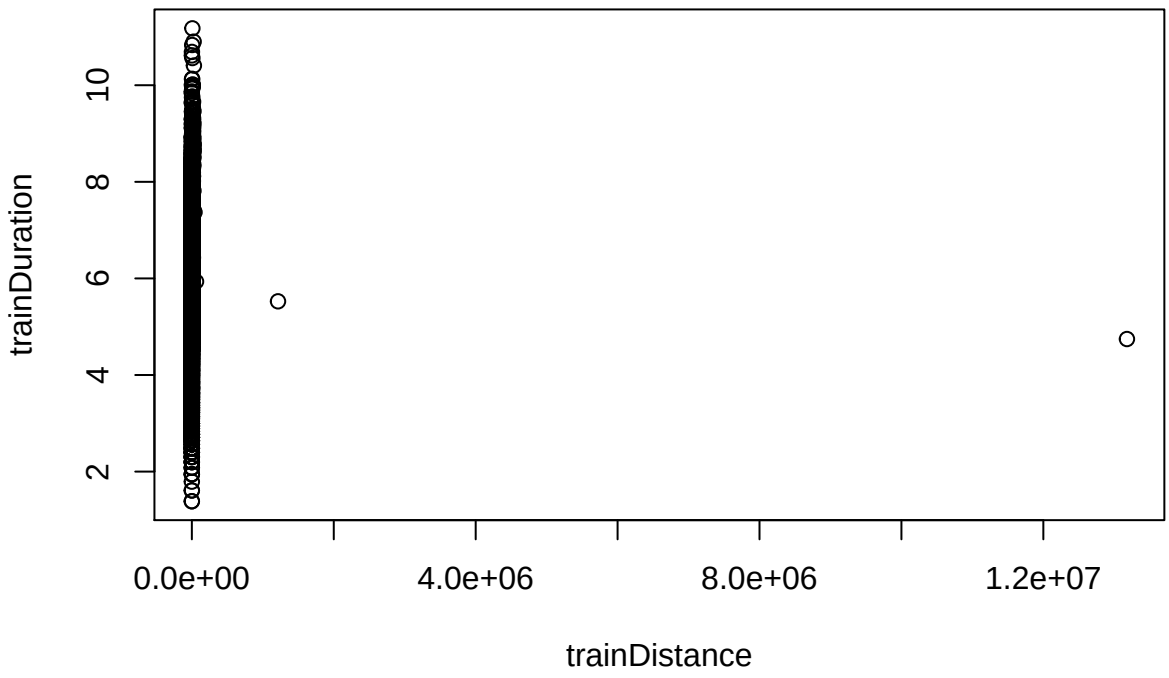
3.The following code transformed the training set by removing trips with a distance or duration that is non-positive, then log the duration variable. And then we plot the density of train duration.Exploratory plots of trip durations shows it is reasonably unimodal and symmetric.

```
trainIndex <- train_data$Trip.Duration>0 & train_data$Trip.Distance>0
trainsubset <- train_data[trainIndex,]
trainDuration <- log(trainsubset$Trip.Duration)
trainDistance <- trainsubset$Trip.Distance
plot(density(trainDuration),main="Distribution of Trip Durations\n(training set)")
```

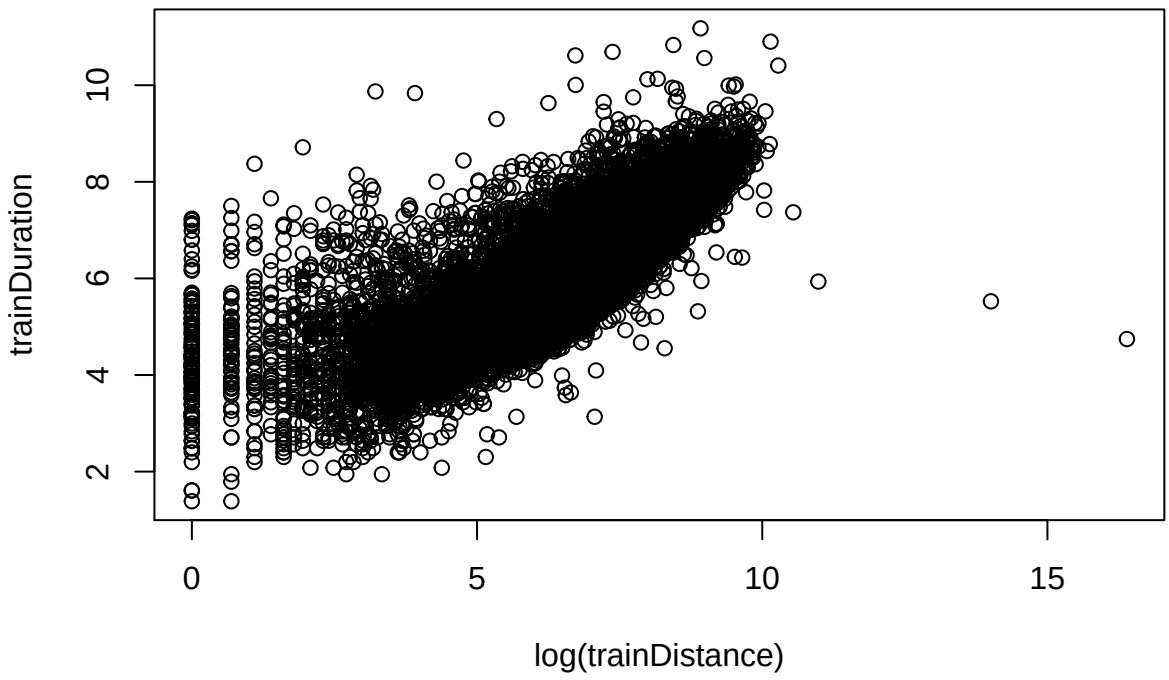


The following plot shows the relationship between duration and distance. The first plot indicates that there is outliers in the data. After trying with log transformation, it looks better.

```
plot(trainDuration ~ trainDistance)
```

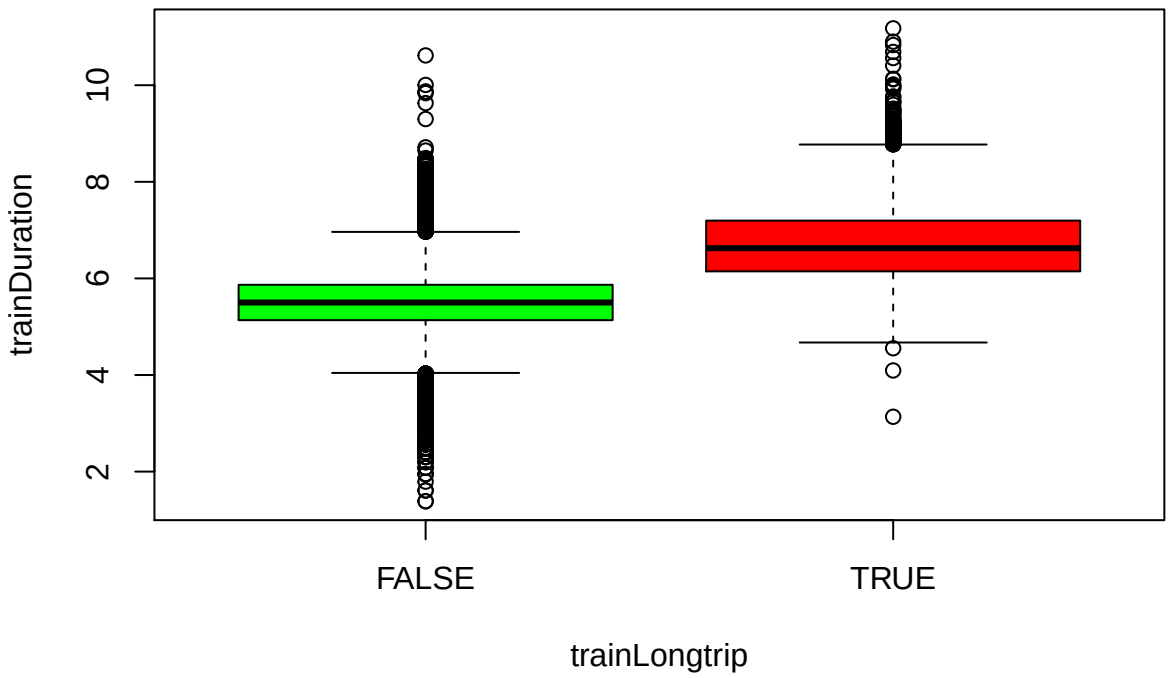



```
plot(trainDuration ~ log(trainDistance))
```



4.To predict the probobility of long trip, we need to transform the data of train distance to be a categorial value. The following code created a "long trip" variable that is TRUE if the trip distance is greater than 1000 (1km) and FALSE otherwise. And then show the relationship between long trip and train duration.

```
train_data$`long trip` <- train_data$Trip.Distance>1000
trainLongtrip <- train_data$`long trip`[trainIndex]
boxplot(trainDuration ~ trainLongtrip, col = c("green","red"))
```



5.The following code fitted a logistic regression model to predict the proportion of long trips based on trip duration using training dataset.

```
fitGLM <- glm(y ~ x, data.frame(x=trainDuration, y=trainLongtrip), family="binomial",na.action=na.exclu
summary(fitGLM)
```

```
##
## Call:
## glm(formula = y ~ x, family = "binomial", data = data.frame(x = trainDuration,
##     y = trainLongtrip), na.action = na.exclude)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.0657  -0.6169   0.1230   0.6384   3.9775
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -16.6023     0.1698  -97.77   <2e-16 ***
## x             2.7723     0.0282   98.31   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 55355  on 40269  degrees of freedom
## Residual deviance: 32519  on 40268  degrees of freedom
## AIC: 32523
```

```
##
## Number of Fisher Scoring iterations: 6
```

6.Before the prediction, we prepared and transformed the test set in the same way as we transformed the training set.

```
testIndex <- test_data$Trip.Duration>0 & test_data$Trip.Distance>0
testsubset <- test_data[testIndex,]
testDuration <- log(testsubset$Trip.Duration)
testDistance <- testsubset$Trip.Distance

test_data$`long trip` <- test_data$Trip.Distance>1000
testLongtrip <- test_data$`long trip`[testIndex]
```

7.We did the prediction and then visualized and evaluated the model on the test set.

```
predGLM <- predict(fitGLM, data.frame(x=testDuration), type="response")
predProp <- mean(testLongtrip, na.rm=TRUE)
breaks = seq(1,10,1)
testBlocks <- cut(testDuration, breaks=breaks)
testProps <- tapply(testLongtrip, testBlocks, mean)
plot(seq(1, 9, 1), testProps)
abline(h=.5, lty="dashed")
abline(h=predProp, col="red", lwd=3)
o <- order(testDuration)
lines(testDuration[o], predGLM[o])

table(testLongtrip, rep(predProp > .5, length(testLongtrip)))
```

```
##
## testLongtrip TRUE
##        FALSE 4488
##        TRUE  5574
```

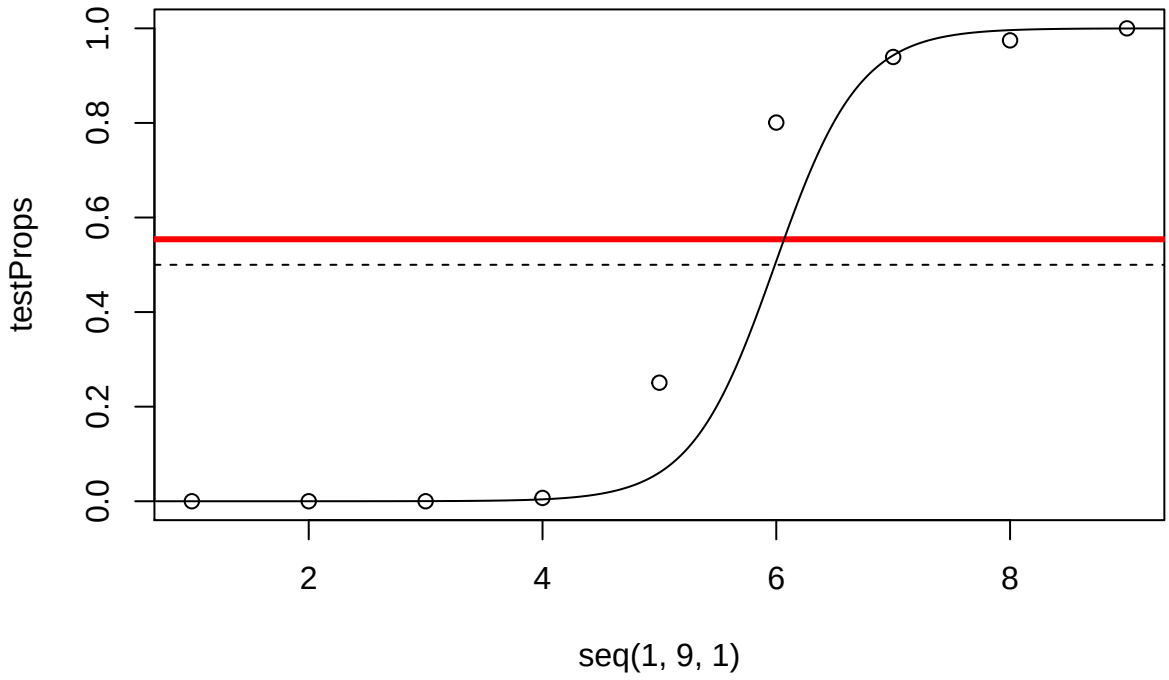
```
table(testLongtrip, predGLM > .5)
```

```
##
## testLongtrip FALSE TRUE
##        FALSE  3665  823
##        TRUE    902 4672
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```




```
confusionMatrix(factor(rep(predProp > .5, length(testLongtrip))),
                factor(testLongtrip))
```

```
## Warning in confusionMatrix.default(factor(rep(predProp > 0.5,
## length(testLongtrip))), : Levels are not in the same order for reference
## and data. Refactoring data to match.
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE     0    0
##      TRUE   4488 5574
##
##                Accuracy : 0.554
##                  95% CI : (0.5442, 0.5637)
##     No Information Rate : 0.554
##     P-Value [Acc > NIR] : 0.5041
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.000
##             Specificity : 1.000
##          Pos Pred Value :   NaN
##          Neg Pred Value : 0.554
```

```
##              Prevalence : 0.446
##          Detection Rate : 0.000
##    Detection Prevalence : 0.000
##       Balanced Accuracy : 0.500
##
##        'Positive' Class : FALSE
##
```

```
confusionMatrix(factor(predGLM > .5),
                factor(testLongtrip))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  3665  902
##      TRUE    823 4672
##
##                Accuracy : 0.8286
##                  95% CI : (0.8211, 0.8359)
##     No Information Rate : 0.554
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6537
##
##  Mcnemar's Test P-Value : 0.06038
##
##             Sensitivity : 0.8166
##             Specificity : 0.8382
##          Pos Pred Value : 0.8025
##          Neg Pred Value : 0.8502
##              Prevalence : 0.4460
##          Detection Rate : 0.3642
##    Detection Prevalence : 0.4539
##       Balanced Accuracy : 0.8274
##
##        'Positive' Class : FALSE
##
```

```
## Vary threshold
confusionMatrix(factor(predGLM > .4),
                factor(testLongtrip))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  3351  565
##      TRUE   1137 5009
##
##                Accuracy : 0.8308
##                  95% CI : (0.8234, 0.8381)
##     No Information Rate : 0.554
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                  Kappa : 0.6534
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7467
##             Specificity : 0.8986
##          Pos Pred Value : 0.8557
##          Neg Pred Value : 0.8150
##              Prevalence : 0.4460
##          Detection Rate : 0.3330
##    Detection Prevalence : 0.3892
##       Balanced Accuracy : 0.8226
##
##        'Positive' Class : FALSE
##
```

```
## Guessing model
confusionMatrix(factor(sample(c(TRUE, FALSE), length(testDistance), replace=TRUE)),
                factor(testLongtrip))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2231 2761
##      TRUE   2257 2813
##
##                Accuracy : 0.5013
##                  95% CI : (0.4915, 0.5111)
##     No Information Rate : 0.554
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0017
##
##  Mcnemar's Test P-Value : 1.241e-12
##
##             Sensitivity : 0.4971
##             Specificity : 0.5047
##          Pos Pred Value : 0.4469
##          Neg Pred Value : 0.5548
##              Prevalence : 0.4460
##          Detection Rate : 0.2217
##    Detection Prevalence : 0.4961
##       Balanced Accuracy : 0.5009
##
##        'Positive' Class : FALSE
##
```

Conclusion

The data for this exercise came in a CSV format, which presented no difficulties for importing into R. Generating a test set was slightly complicated by requiring sub-samples from each month, though this only required standard R data manipulation tools.

We required a log-transformation of duration variables to obtain unimodal, symmetric data for model fitting. This required excluding non-positive distances and durations (about 8% of the data),which means that our predictive model is only valid for trips that had a non-zero distance or duration. We are unlikely to be predicting zero-distance or zero-duration trips, so this is not a large concern.

We fitted a logistic regression model, which provided some predictive power, though there are clear opportunities for a more flexible model(or a more robust model) to perform better.

According to the result of evaluation, the accuracy, sensitivity and specificity are all about 0.82 respectively with the threshold of 0.5. The details of numbers are displayed in the above result. So the model is good for the analyze. We also tried to change the threshold to be 0.4, and there is little change for the accuracy, sensitivity and specificity.

We finished this experiment on linux virtual machine with commands like ssh, scp,ls,cd,grep,vim,R. The difference is is that, since there is no desktop of the virtual machine, we can not use RStudio. We need to generate the report by using linux command:

Rscript -e “rmarkdown::render(‘./STATS769_Lab02_YujieZhang.rmd’)”

Since linux system is widely used in different servers in the environment of production, it’s still convinent to process big files on the servers.