Transportation Mode Identification

Introduction:

Many different forms of transportation are widely used by people every day. It has become easier to get data of people's form of transportation because almost everyone carries a smart phone with them, which contains sensors. Transportation mode identification is a growing topic of research, with many applications in the field of the Internet of Things. It identifies a user's transportation modes through observations of the user or observation of the environment. This is useful because transportation mode detection can provide context information useful to offer appropriate services based on the user's needs and possibilities of interaction. Transportation mode detection can help with smart parking, traffic monitoring, urban transportation planning and so much more. We will explore different models that can be used to identify a user's transportation based on a given data set.

Data Set:

The data set used is Transportation Mode Detection with Unconstrained Smartphones Sensors and can be found at kaggle.com. The data set consists of time and different sensors that identify a user's transportation modes through observations of the user, or observation of the environment. Nine sensors are used which are accelerometer, game rotation vector, gyroscope, gyroscope uncalibrated, linear acceleration, orientation, rotation vector, sound, and speed. Each sensor has a measurement of mean, minimum, maximum, and standard deviation. There are five target variables, which are bus, car, still, train, and walking. The data set was split into three different sets, Dfirst, Dsecond, and Dthird. The first dataset, Dfirst, is formed by three sensors and time, thirteen features. Sensors included in the first set are accelerometer, sound, and gyroscope. These three sensors have the highest values of accuracy taken individually. The second dataset, Dsecond, is formed by eight sensors and time, thirty-three features. The third dataset, Dthird, is formed by all nine relevant sensors and time, thirty-seven features. Dthird differs from the previous set, Dsecond because it includes speed. All the data sets contain 5894 rows and each row is considered as a sample. We normalized the Dthird data set based on z-scores, because it has all the features from both previous data sets. Normalizing the data makes it easier to compare the different measurements used for the different features. We used the first 4000 rows as our training data set and used the remaining 1894 rows as our testing data set.

Methods Tried:

Logistic Regression

We began our analysis with logistic regression. As logistic regression is best used to describe the relationship between a binary target variable and other independent variables, it would be a good method for our predictive analysis. The glaring issue was that our target variable was not binary. Therefore, we combined several transportation modes into one class and completed binary

classification that way. By completing this iteratively, we got five different regressions that describe the data. The 8 most important sensors used in the creations of the regressions were the means of acceleration, gyroscope, sound, game rotation, linear acceleration, orientation, rotation vector, and speed. In order to test how accurate the regressions were, we completed R^2 tests. With the R^2 statistic, we would be able to see how much of our data is explained by the regressions. We aimed for the highest possible R^2 values. Our finalized R^2 values are as follows: Walking: 0.693, Bus: 0.303, Train: 0.344, Still: 0.495, and Car: 0.365. On our test data, the regressions were fairly accurate, with classifications being 72% accurate. Based on our confusion matrix comparing actual vs predicted, the most misclassifications occured when classifying still as train.

	predictions				
	Bus	Car	Still	Train	Walking
Bus	235	41	33	54	18
Car	34	269	15	53	10
Still	18	5	285	82	7
Train	25	42	47	246	13
Walking	20	12	4	8	317

Support Vector Machine

The next method we tried was support vector machine. We thought this would be a good method to use because it is used to analyze data for classification and regression analysis. For our analysis we are trying to create predictive models based on sensors and time that would predict the transportation mode of bus, car, train, still and walking, and support vector machine is able to do this. We started by looking at only the means of the sensors and time in order to test for tuning parameters and to determine which kernel would be the best for the test. After doing multiple tests, we found that the polynomial kernel was the best to use and produced the highest accuracy. In order to choose the best tuning parameters, we used cross validation to determine which parameters had the smallest error, and those parameters provided the highest accuracy. When testing for the tuning parameters, we tested for the degree and cost parameters, since the polynomial kernel was the one we decided to choose for our kernel. After picking the kernel and tuning parameters, we started to look at the correlations, boxplots and ANOVA tests for the feature variables and targets to see if those tests would help narrow down the number of features used in the support vector machine tests. We also looked at normalizing the data with the max-normalization and the z-score-normalization. We used the Dthird data set to normalize, because it contained all of the features that were in both Dfirst and Dsecond. After doing multiple testing based on high positive correlations, the correlations did not seem to be too helpful in determining the best features because many of the accuracies of the tests were pretty low, in the ranges of 40%-60%, which can be viewed in Analysis by Stephanie.R file. The features found by the ANOVA tests seemed to be somewhat helpful in picking the best feature variables. We did multiple tests, but only a few got very high accuracy. The test that got the

highest accuracy used the Dthird data set, that has been normalized. It uses all the features, time, all the sensors and all their measurements which in total is 37 feature variables.

```
svm1 <- svm(target~., data=z_completetrain, method="C-classification", scale = FALSE, kernal="polynomial", cost=100, degree=
3) # polynomial kernel
summary(svm1) # summary of svm
```

```
##
## Call:
## svm(formula = target ~ ., data = z_completetrain, method = "C-classification",
##     kernal = "polynomial", cost = 100, degree = 3, scale = FALSE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  100
##
## Number of Support Vectors:  1355
##
##  ( 296 242 292 217 308 )
##
##
## Number of Classes:  5
##
## Levels:
##  Bus Car Still Train Walking
```

```
##          pred1
## truth1    Bus Car Still Train Walking
##   Bus     345   4     2    20      10
##   Car      22 346     3     5       5
##   Still     5   2   369    14       7
##   Train    11   6    14   338       4
##   Walking   9   3     4     0     345
```

```
# Accuracy for testing
correct <- 0
for (i in 1:5) {
  correct <- correct + table(truth1, pred1)[i,i]
}
correct/nrow(z_completetest) # 92.08% accuracy
```

```
## [1] 0.9207607
```

This test had 1355 support vectors and provided a 92.07% accuracy for the test data set and for the training data set, the accuracy was 97.6%. This is a high accuracy and should be high considering it uses all the feature variables. Also seen in the truth table, only a few were misclassified for each target. With support vector machine, it is better to have more features, than fewer. The next test that we did used only the means of the sensors and time. We wanted to see how the different measurements for the sensors would affect the accuracy of the tests. This test got the second highest accuracy and again used the Dthird data set, that has been normalized. It uses all time and all the sensors, but means only which in total is 10 feature variables.

```
svm2 <- svm(target~., data=z_normaltrain, method="C-classification", scale = FALSE, kernal="polynomial", cost=100, degree=4)
# polynomial kernel
summary(svm2) # summary of svm
```

```
##
## Call:
## svm(formula = target ~ ., data = z_normaltrain, method = "C-classification",
##     kernal = "polynomial", cost = 100, degree = 4, scale = FALSE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  100
##
## Number of Support Vectors:  1343
##
##  ( 317 207 271 246 302 )
##
##
## Number of Classes:  5
##
## Levels:
##  Bus Car Still Train Walking
```

```
##          pred2
## truth2    Bus Car Still Train Walking
##   Bus     340  15     3    16       7
##   Car      36 327     9     9       0
##   Still     1   4   379    10       3
##   Train    13   8    19   330       3
##   Walking  13   1     7     1     339
```

```
# Accuracy for testing
correct <- 0
for (i in 1:5) {
  correct <- correct + table(truth2, pred2)[i,i]
}
correct/nrow(z_normaltest) # 90.59% accuracy
```

```
## [1] 0.9059694
```

This test had 1343 support vectors and provided a 90.59% accuracy for the test data set and for the training data set, the accuracy was 95%. The accuracy for this test is a little lower compared to the previous test, however it uses 27 less feature variables. It is interesting to see that the accuracy went down for the test data set by not more than 2%, even though a lot less features are used. Also seen in the truth table, there are more misclassified targets, but not as many for walking. It seems that walking is a little easier to classify compared to the other targets. The third test that we did used only sensors that were found to be significant from ANOVA tests and boxplots, which were found to be their means only. This test got the third highest accuracy and again used the Dthird data set, that has been normalized. It uses only seven sensors, but means only which in total is seven feature variables.

```
svm3 <- svm(as.formula(paste(colnames(z_completetrain)[38], "~", paste(colnames(z_completetrain)[c(2, 6, 10, 22, 26, 30, 34)], collapse = "+"), sep = "")), data=z_completetrain, method="C-classification", scale = FALSE, kernal="polynomial", cost=100, degree=4) # polynomial kernel
summary(svm3) # summary of svm
```

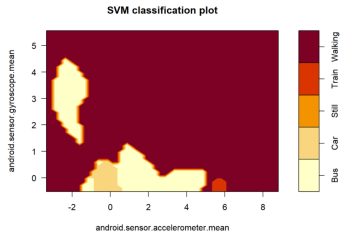
```
##
## Call:
## svm(formula = as.formula(paste(colnames(z_completetrain)[38], "~",
##     paste(colnames(z_completetrain)[c(2, 6, 10, 22, 26, 30, 34)],
##         collapse = "+"), sep = "")), data = z_completetrain, method = "C-classification",
##     kernal = "polynomial", cost = 100, degree = 4, scale = FALSE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  100
##
## Number of Support Vectors:  1571
##
##  ( 375 216 298 323 359 )
##
##
## Number of Classes:  5
##
## Levels:
##  Bus Car Still Train Walking
```

```
##          pred3
## truth3    Bus Car Still Train Walking
##   Bus     314  33     4    20      10
##   Car      33 317    19    12       0
##   Still     6   3   370    15       3
##   Train    18   7    31   313       4
##   Walking   9   5     6     2     339
```

```
# Accuracy for testing
correct <- 0
for (i in 1:5) {
  correct <- correct + table(truth3, pred3)[i,i]
}
correct/nrow(z_completetest) # 87.32% accuracy
```

```
## [1] 0.8732171
```

This test had 1571 support vectors and provided an 87.32% accuracy for the test data set and for the training data set, the accuracy was 90.3%. The accuracy for this test is a little lower compared to the previous two tests, however it uses 30 less feature variables than the first. It is interesting to see that the accuracy went down for the test data set by not more than 5%, even though a lot less features are used. Also seen in the truth table, there are more misclassified, but again not as many for walking. It seems that walking is a little easier to classify compared to the other targets. A graph below shows the support vector machine plot for this test.



The plot uses gyroscope mean and accelerometer mean as the x and y axes. These features were chosen because they provide the most significance when testing for the target variables. As seen in the graph, walking is the most distinct out of all the targets. This relates to the truth table, because walking had the least misclassified. Bus is also pretty visible in the plot, while car and train are somewhat visible. Still is very difficult to see in the plot, this is because still is the hardest target variable to separate compared to the other targets, while walking is the best. After doing many different types of tests, the above three were the tests with the highest accuracies for support vector machines. Comparing the different data sets, Dfirst, Dsecond, Dthird and normalized Dthird, the accuracy increases with each data set because more features are being added. Normalizing the data set also helps to increase the accuracy because it helps to make it easier to compare the different sensors used. The support vector machine test that is preferred depends on some factors. If the maximum accuracy was required, then the first test will be preferred, but if the highest accuracy with the least amount of feature variables used was required, then the third test will be preferred.

Decision Tree

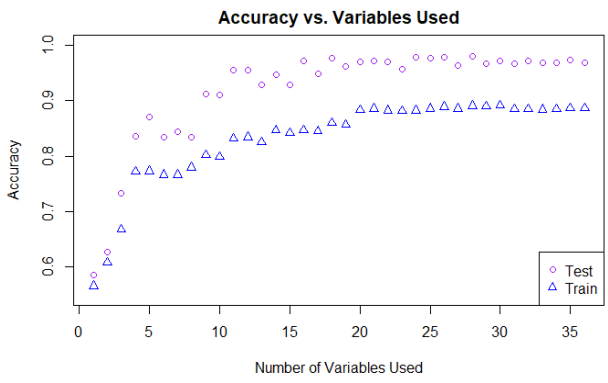
We decided to also build a decision tree model because it is a non-parametric technique that can accommodate irregular separation planes. The flexibility of decision trees to adapt to diverse separation planes and ease of interpretation are its biggest advantages. However, it is also highly dependent on input and is prone to overfitting. We chose a post-pruning approach to construct

the classification because we prioritized accuracy over complexity. Since there is no easy way to separate the data points, the decision tree necessarily had to be large to achieve high accuracy. Thus, we first set the cost parameter to be zero and minimum split to equal to 5 to grow the tree to full depth. A minimum split of 5 was used over 2 because it yielded more stable results, probably due to the fact that there are 5 groups in the response variable. We then used cross validation to select the cost parameter with the lowest cross validation error and prune the tree accordingly.

```
          pred
truth      Bus Car Still Train Walking
  Bus      335  15     3    12      14
  Car       20 343    15     4      12
  Still      6   5   305    20       9
  Train     16   6    13   364       6
  Walking   13   8     6    10     333
[1] 0.8874802
```

As the confusion matrix illustrates, the accuracy of the best performing tree predicts the correct transportation method close to 88% of the time. Among the five methods, train and walking were the easiest to identify while car was the hardest because there is significant overlap between car, bus, and still.

The biggest downside to using a decision tree is that it requires many nodes to achieve high accuracy, which significantly increases the complexity of the model. The decision is even too large to be printed and viewed normally. Nevertheless, we tried to simplify the model by eliminating unuseful feature variables. We ordered all 37 variables by their importance in generating the original large tree, and iteratively deleted the least significant variable and constructed a new tree to assess the accuracy of the model without the deleted variable. Though this was a computation-intensive step, R was able to generate the output within about a minute, which speaks to the convenience and efficiency of using a decision tree model. As shown in the figure below, a decision using just 4 variables was already 77% accurate, and using past 20 variables doesn't change the accuracy at all. With this information, we conclude that it is efficient to incorporate only the most important 20 variables, which come from 6 unique sensors and time, into the final decision tree. However, even with a reduced number of variables, the decision tree is still too large to be printed.



Issues:

The accuracy of logistic regression could have been improved by incorporating a nonlinear classifier. Originally, our classes were separated by linear boundaries. If we implemented logistic regression as a non linear classifier, we could have improved the test accuracy. We would need a better idea of the shape of the decision boundary, which we did not know. Unfortunately, this was not easy to incorporate and with more exploration can be achieved in the future.

Support vector machine had some minor issues with runtime. When finding tuning parameters, the run time was very long and felt like it would not even load sometimes. In order to fix this issue in the future, we can reduce the number of features used in the cross validation test to help reduce the runtime since there will be less features to go through. This may however not provide the absolute best tuning parameters because not all the features that are being used will be counted.

Final Results:

Support vector machine had the highest accuracy results out of all the methods tried. It also has the highest accuracy with the least amount of feature variables used, which was 87.32% accuracy with only 7 features used. The maximum accuracy got from the support vector machine tests, however, used the most feature variables compared to the other two methods tried. The most important features used in the support machine tests were rotation vector, speed, orientation, linear acceleration, sound, gyroscope and accelerometer.

Conclusion:

After doing multiple tests, it is clear that the data is not linearly separable, so logistic regression and clustering do not work well. Support vector machine yielded the best performing model, but the run-time was longer. Decisions trees work the best when the number of sensors are limited, but it does require more variables. The decision tree model is the preferred model because although the accuracy is slightly lower than support vector machines, the run-time is a lot faster and can use less sensors. The model can also be built upon by employing a random forest to make the test more accurate in the future.

Contribution:

Task	**Participants**	**Estimated Percentage**
Data preparation	Stephanie Boehm Bella Patel Albert Yen	25% 25% 50%

Logistic Regression	Stephanie Boehm Bella Patel Albert Yen	0% 100% 0%
Support Vector Machine	Stephanie Boehm Bella Patel Albert Yen	100% 0% 0%
Decision Trees	Stephanie Boehm Bella Patel Albert Yen	0% 0% 100%
K-Means	Stephanie Boehm Bella Patel Albert Yen	0% 0% 100%
Presentation	Stephanie Boehm Bella Patel Albert Yen	33.33% 33.33% 33.33%
Report	Stephanie Boehm Bella Patel Albert Yen	33.33% 33.33% 33.33%

Code:

All data and code is organized in the Github repository:
https://github.com/lsugyen/Stat-385-final-project