Group 132: Run or Walk Prediction

First Name	Last Name	Monday or Tuesday class
LAVANYA	SURIKAPURAM	Tuesday

Table of Contents

1. Introduction

This dataset aims to detect whether the person is running or walking based on deep neural network and sensor data collected from iOS device.

This dataset has been accumulated with help of "Data Collection" iOS app specially developed for this purpose.

A dataset containing labeled sensor data from accelerometer and gyroscope

2. Data

Currently, the dataset contains a single file which represents 88588 sensor data samples collected from accelerometer and gyroscope from iPhone 5s in 10 seconds' interval and ~5.4/second frequency.

This data is represented by following columns (each column contains sensor data for one of the sensor's axes):

- acceleration_x
- acceleration_y
- acceleration_z
- gyro_x
- gyro_y
- gyro_z

There is an activity type represented by "activity" column which acts as label and reflects following activities:

- "0": walking
- "1": running

Apart of that, the dataset contains "wrist" column which represents the wrist where the device was placed to collect a sample on:

- "0": left wrist
- "1": right wrist

Additionally, the dataset contains "date", "time" and "username" columns which provide information about the exact date, time and user which collected these measurements.

Application or Domain the Data Belongs:

The data set is obtained from Kaggle.
It belongs to health and fitness domain.

Data Set Collected from the below link:

Link: https://www.kaggle.com/vmalyi/run-or-walk

Size of the Data set:

88589 Records
11 fields

3. Problems to be Solved

The problems I want to solve, in this project:

1. **Problem 1:**
   Currently, the dataset contains 88588 sensor data samples collected from accelerometer and gyroscope from iPhone 5s in 10 seconds' interval and ~5.4/second frequency. Assuming both acceleration_x between the person who is walking and the person who is running are approximately normal with equal variances, is there a difference in mean yield (□ = 0.05)? (**2 sampled Independent hypothesis testing**). Hence, we use 2 sampled Independent hypothesis testing to predict the statistically significant difference in mean of acceleration_x between the person who is walking and the person who is running using a 5% level of significance.

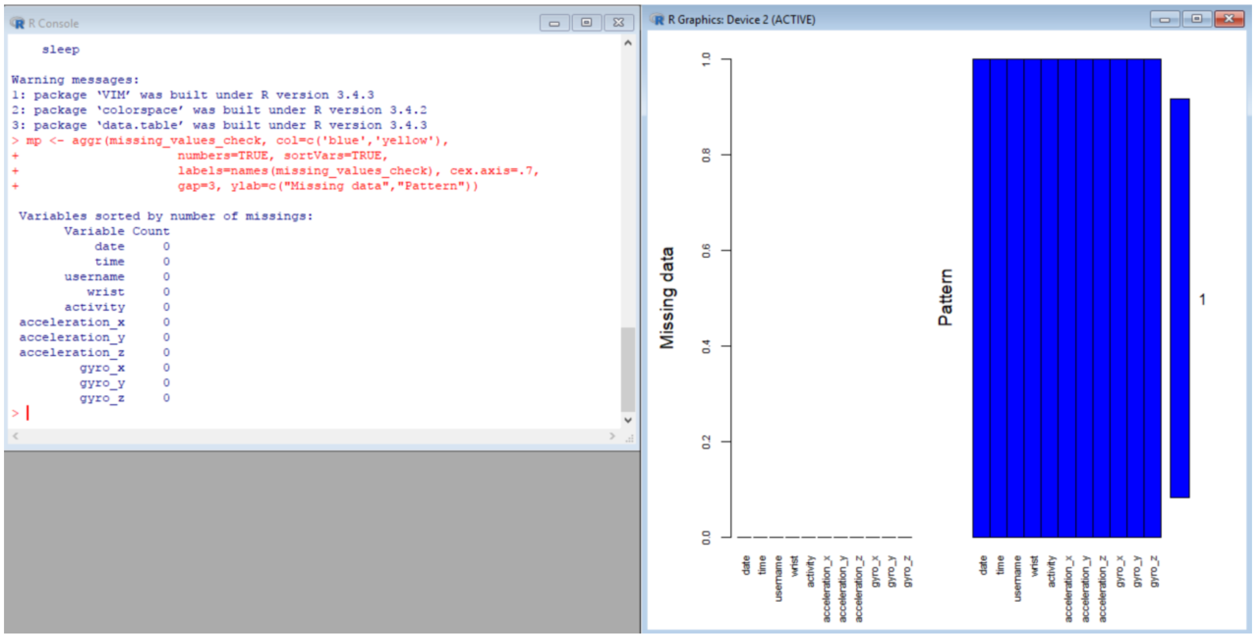
2. **Problem 2:**
   I was told the average Axcellaration_x for the person who is running is -0.09F. But I do not think so?
   Solution as on which method it can solved: **One sample Two Tailed hypothesis testing**

3. **Problem 3:**
   I was told the average Axcellaration_x for the person who is running is less than -0.09F. But I do not think so?
   Solution as on which method it can solved: **One sample one Tailed hypothesis testing**

4. **Problem 4:**
   The main problem in this project is to detect whether the person is running or walking based on deep neural network and sensor data collected from iOS device. Solution as on which method it can solved: (**Logistic Regression** with activity being the dependent variable (run or walk) as well we can compare the model with other classification models like DECISION TREES, RANDOM FOREST, k-Nearest Neighbor Predictions)

4. Data Processing

- Handled missing values
  - ➔ Checked if there were any missing values in my data



```
    sleep

Warning messages:
1: package ‘VIM’ was built under R version 3.4.3
2: package ‘colorspace’ was built under R version 3.4.2
3: package ‘data.table’ was built under R version 3.4.3
> mp <- aggr(missing_values_check, col=c('blue','yellow'),
+                      numbers=TRUE, sortVars=TRUE,
+                      labels=names(missing_values_check), cex.axis=.7,
+                      gap=3, ylab=c("Missing data","Pattern"))

 Variables sorted by number of missings:
       Variable Count
           date     0
           time     0
       username     0
          wrist     0
       activity     0
 acceleration_x     0
 acceleration_y     0
 acceleration_z     0
         gyro_x     0
         gyro_y     0
         gyro_z     0
> 
```

This code, visualize the patterns of NAs and from the plot there are no null values in the dataset. Hence the data is clean.

- Splitted the data into testing (20%) and training (80%) sets.

5. Methods and Process

The list of methods used to solve the above listed problem:

- ➔ **Hypothesis Testing**
  - Two sampled independent Hypothesis Testing
  - One sampled two tailed Hypothesis Testing
  - One sampled one tailed Hypothesis Testing
- ➔ **Logistic Regression**
  - Backward Elimination
  - Step wise backward selections
  - Step wise Forward
- ➔ **Decision Trees**
- ➔ **Random Forest**
- ➔ ***k*-Nearest Neighbor Predictions**

Detailed Explanation about the methods used to solve the problem to predict the potential predictive model:

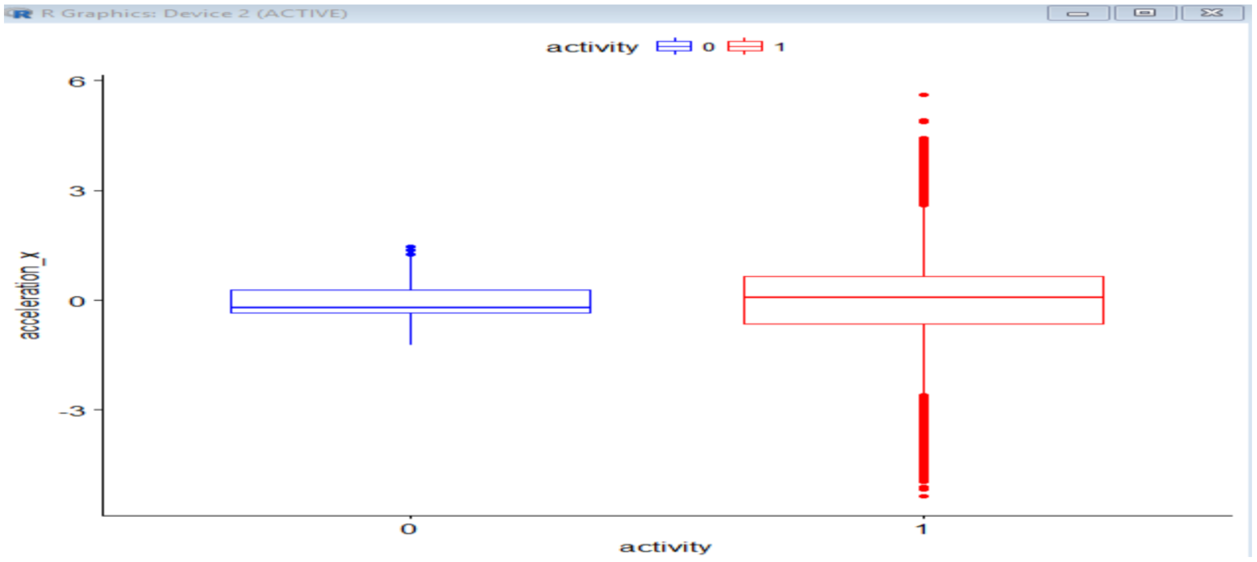
Method to solve Problem 1:

Problem: Assuming both acceleration_x between the person who is walking and the person who is running are approximately normal with equal variances, is there a difference in mean yield?

Solution:

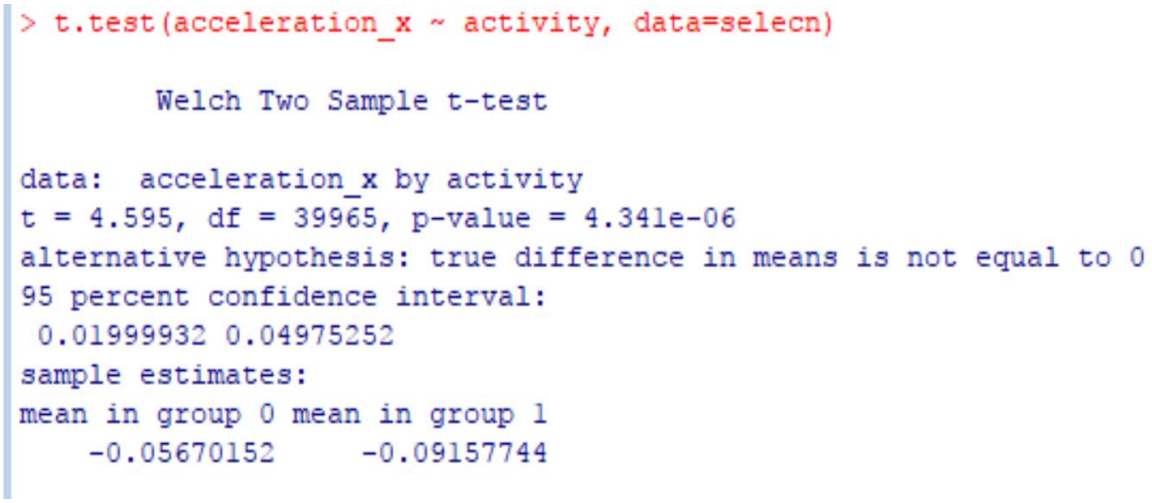
Two sampled independent Hypothesis Testing

Box Plot:



0 represents box plot for walking people

1 represents box plot for running people

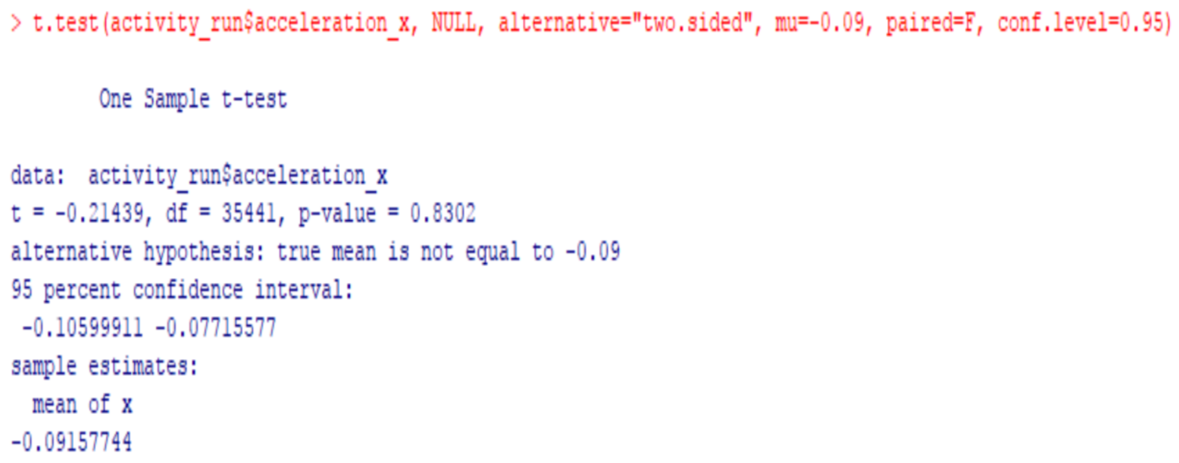


```
> t.test(acceleration_x ~ activity, data=selecn)

        Welch Two Sample t-test

data:  acceleration_x by activity
t = 4.595, df = 39965, p-value = 4.341e-06
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.01999932 0.04975252
sample estimates:
mean in group 0 mean in group 1
    -0.05670152     -0.09157744
```

From the result, it is confirmed there is a difference in mean as the alternative hypothesis is true.

Method to solve Problem 2:

Problem: I was told the average Axcellaration_x for the person who is running is -0.09F. But I do not think so?



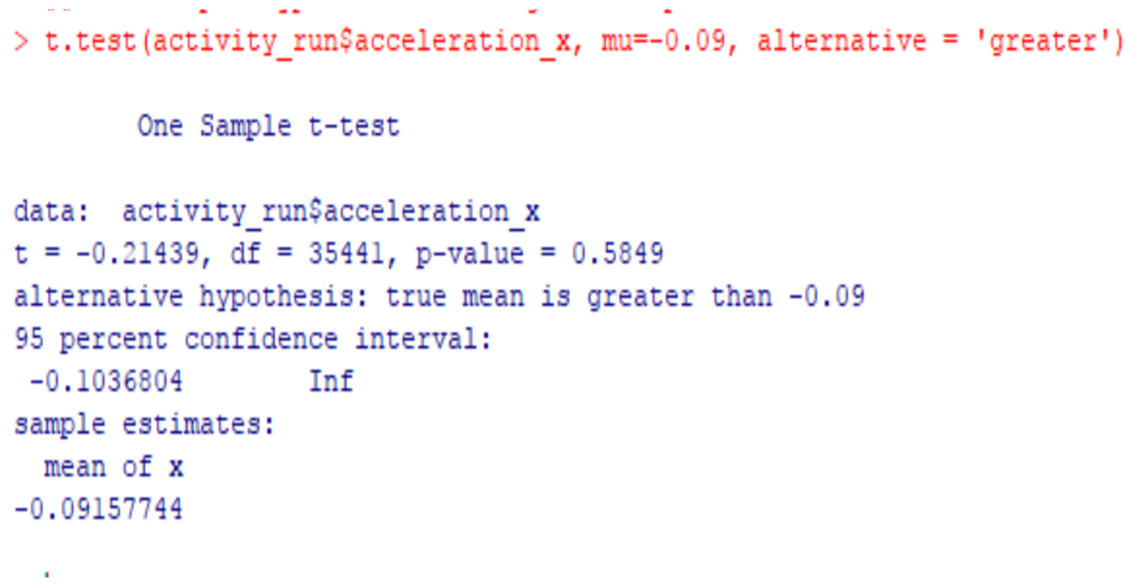
```
> t.test(activity_run$acceleration_x, NULL, alternative="two.sided", mu=-0.09, paired=F, conf.level=0.95)

	One Sample t-test

data:  activity_run$acceleration_x
t = -0.21439, df = 35441, p-value = 0.8302
alternative hypothesis: true mean is not equal to -0.09
95 percent confidence interval:
 -0.10599911 -0.07715577
sample estimates:
  mean of x
-0.09157744
```

Hence from the result it is known that the average of Axcellaration_x for the person who is running is not equal to -0.09F as the alternative hypothesis is true.

(Hence my Assumption was right that the average of Axcellaration_ x for the person who is running is not equal to -0.09F)

Method to solve Problem 3:

Problem: I was told the average Axcellaration_x for the person who is running is less than -0.09F. But I do not think so?



```
> t.test(activity_run$acceleration_x, mu=-0.09, alternative = 'greater')

	One Sample t-test

data:  activity_run$acceleration_x
t = -0.21439, df = 35441, p-value = 0.5849
alternative hypothesis: true mean is greater than -0.09
95 percent confidence interval:
 -0.1036804        Inf
sample estimates:
  mean of x
-0.09157744
```

Hence from the result it is known that the Axcellaration_x for the person who is running is not less than -0.09F

(Hence my Assumption was right that the average of Axcellaration_x is not less than -0.09F)

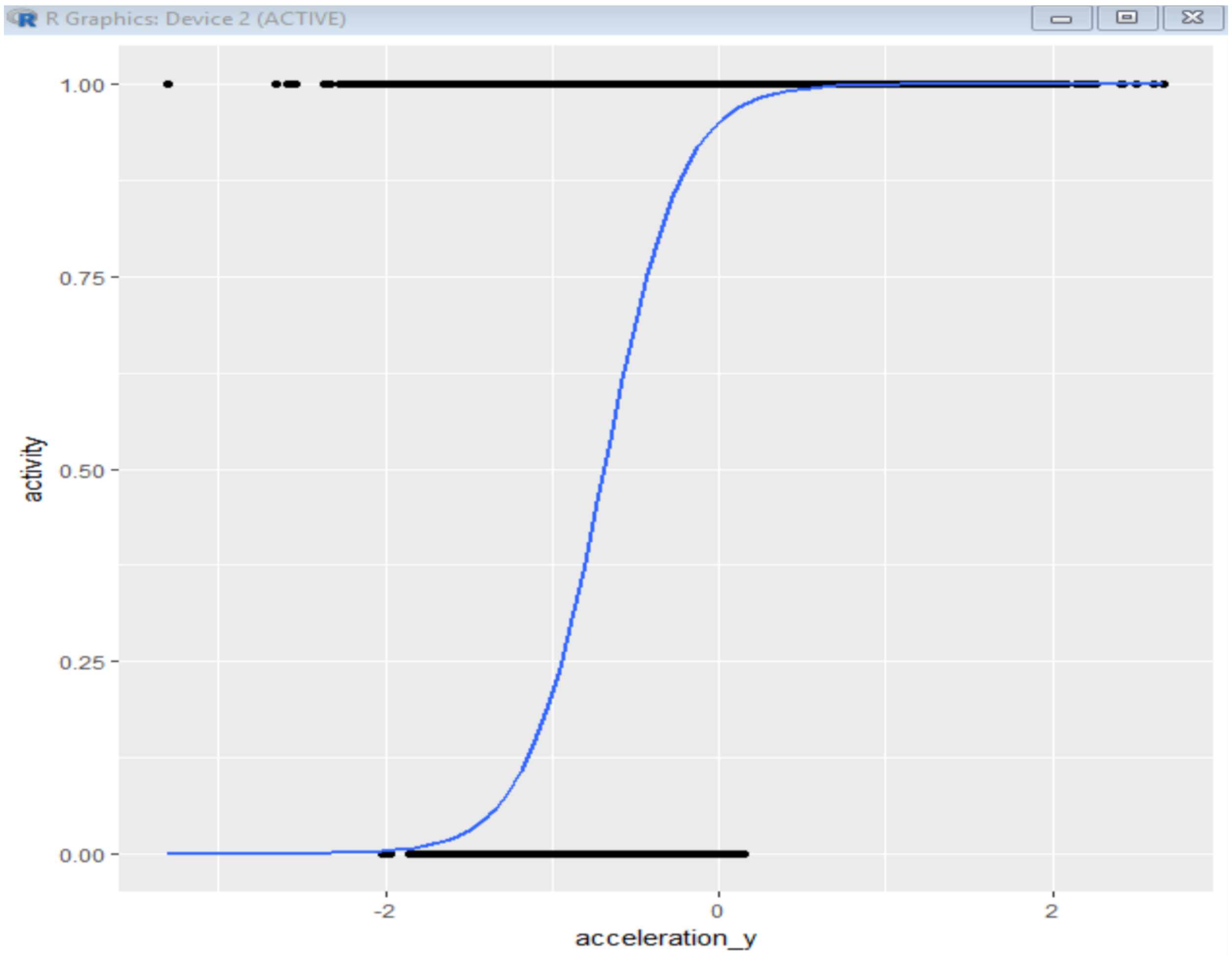
Method to solve Problem 4:

Problem: To examine if the person is walking or running, I need to use logical regression Model. (with activity being the dependent variable (run or walk))

- Split the data into training and testing
- Fit a logistic regression model using training data
- Use the fitted model to do predictions for the test data using the below methods,

i. **BACKWARD ELEMINATION**
ii. STEPWISE BACKWARD REGRESSION
iii. STEPWISE FOWARD REGRESSION

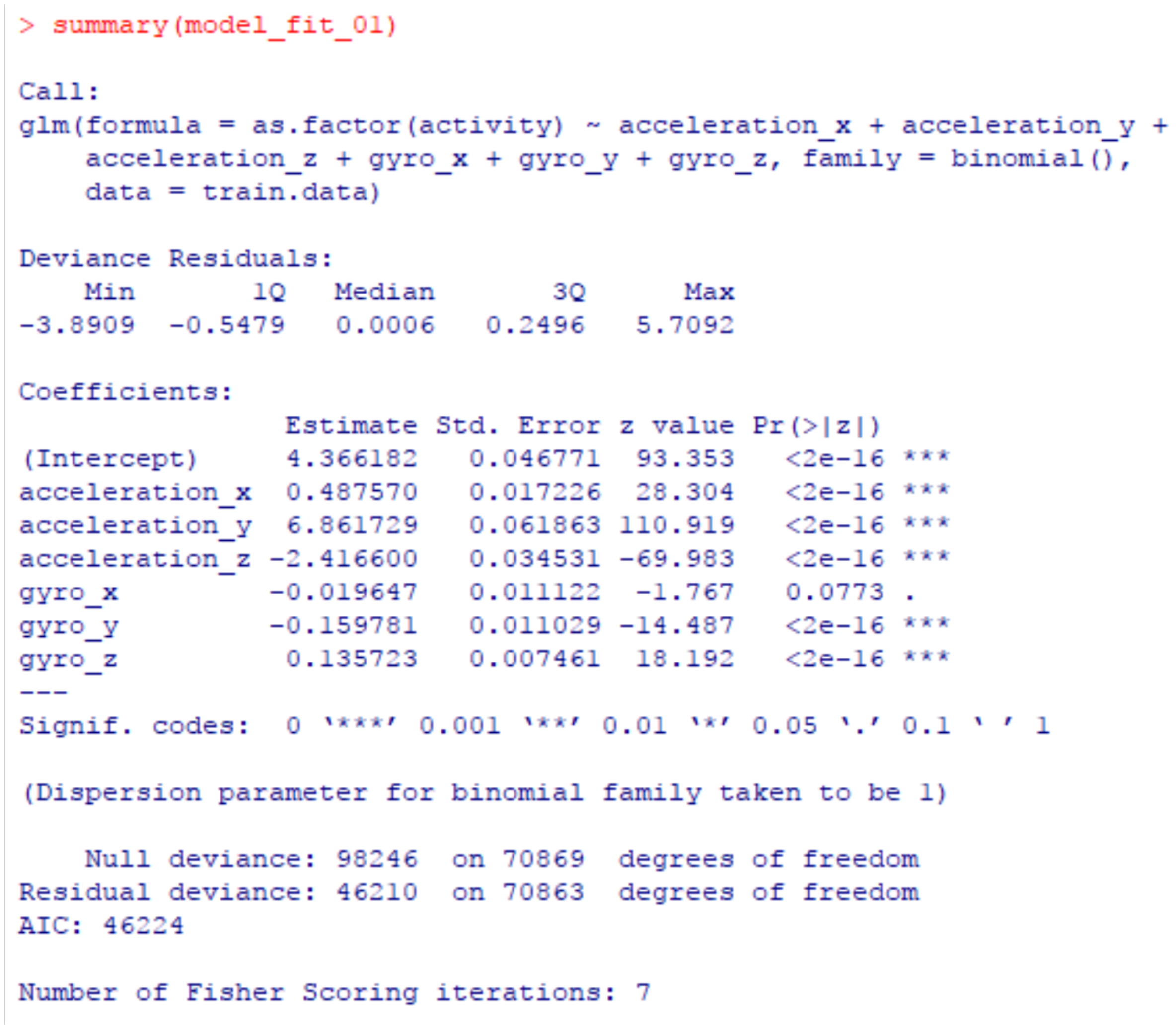
Axcellaration_y curve



This curve confirms us that the data is suitable in order to perform logistic regression

i. **BACKWARD ELEMINATION**: Start with the full model and eliminates one variable at the time until a reasonable candidate regression model is found. It typically uses a criterion based on the goodness-of-fit F-test. The metric could be p-value, AIC, BIC.

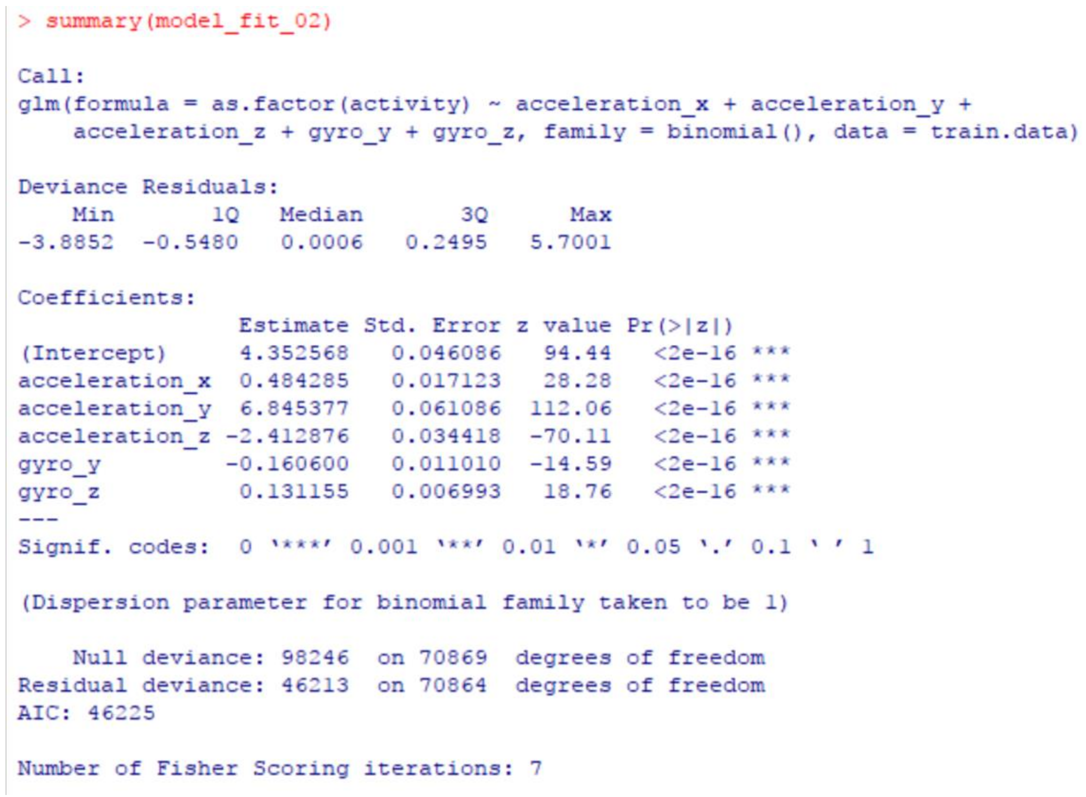
Model 1:



```
> summary(model_fit_01)

Call:
glm(formula = as.factor(activity) ~ acceleration_x + acceleration_y +
    acceleration_z + gyro_x + gyro_y + gyro_z, family = binomial(),
    data = train.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.8909  -0.5479   0.0006   0.2496   5.7092

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     4.366182   0.046771  93.353   <2e-16 ***
acceleration_x  0.487570   0.017226  28.304   <2e-16 ***
acceleration_y  6.861729   0.061863 110.919   <2e-16 ***
acceleration_z -2.416600   0.034531 -69.983   <2e-16 ***
gyro_x         -0.019647   0.011122  -1.767   0.0773 .
gyro_y         -0.159781   0.011029 -14.487   <2e-16 ***
gyro_z          0.135723   0.007461  18.192   <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: 98246  on 70869  degrees of freedom
Residual deviance: 46210  on 70863  degrees of freedom
AIC: 46224

Number of Fisher Scoring iterations: 7
```

As gyro_x has p value > 0.05, we need to procced with model 2 by removing gyro_x

Model 2:

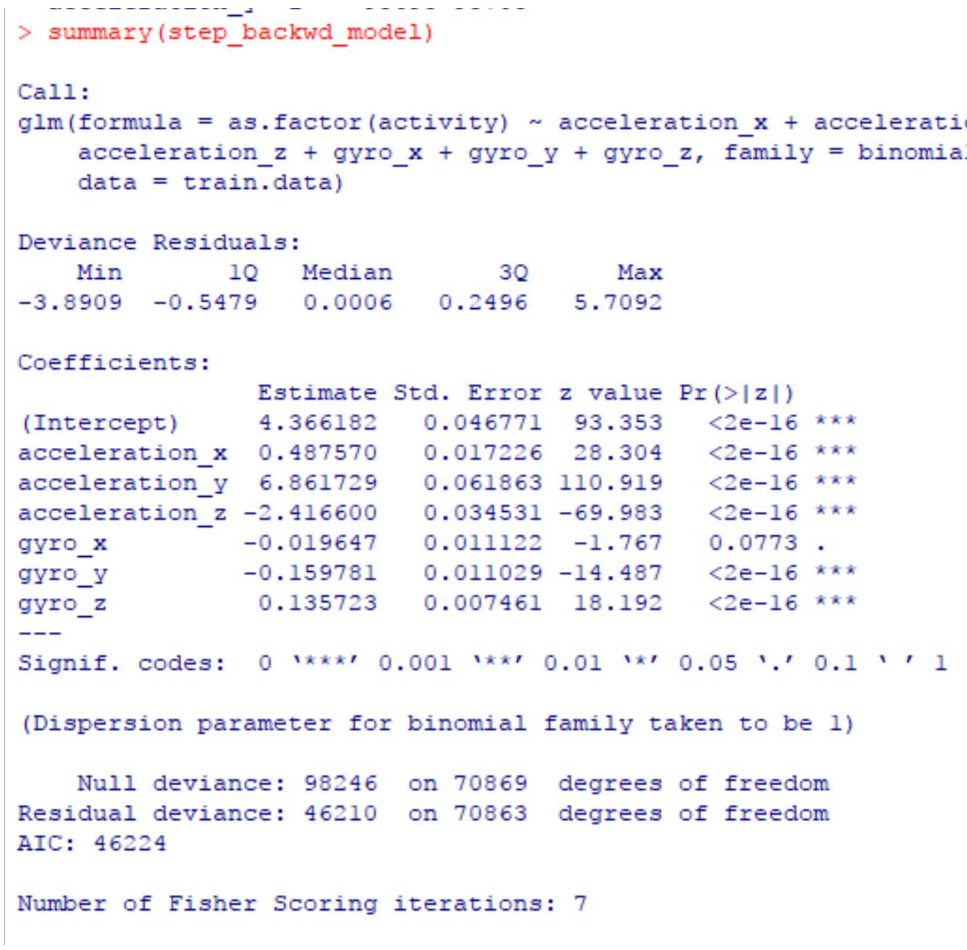


```
> summary(model_fit_02)

Call:
glm(formula = as.factor(activity) ~ acceleration_x + acceleration_y +
    acceleration_z + gyro_y + gyro_z, family = binomial(), data = train.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.8852  -0.5480   0.0006   0.2495   5.7001

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     4.352568   0.046086   94.44   <2e-16 ***
acceleration_x  0.484285   0.017123   28.28   <2e-16 ***
acceleration_y  6.845377   0.061086  112.06   <2e-16 ***
acceleration_z -2.412876   0.034418  -70.11   <2e-16 ***
gyro_y         -0.160600   0.011010  -14.59   <2e-16 ***
gyro_z          0.131155   0.006993   18.76   <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: 98246  on 70869  degrees of freedom
Residual deviance: 46213  on 70864  degrees of freedom
AIC: 46225

Number of Fisher Scoring iterations: 7
```

As all the p values are less than .05 we can stop the backward elimination and say this model is fit under backward elimination.

ii. STEPWISE BACKWARD REGRESSION

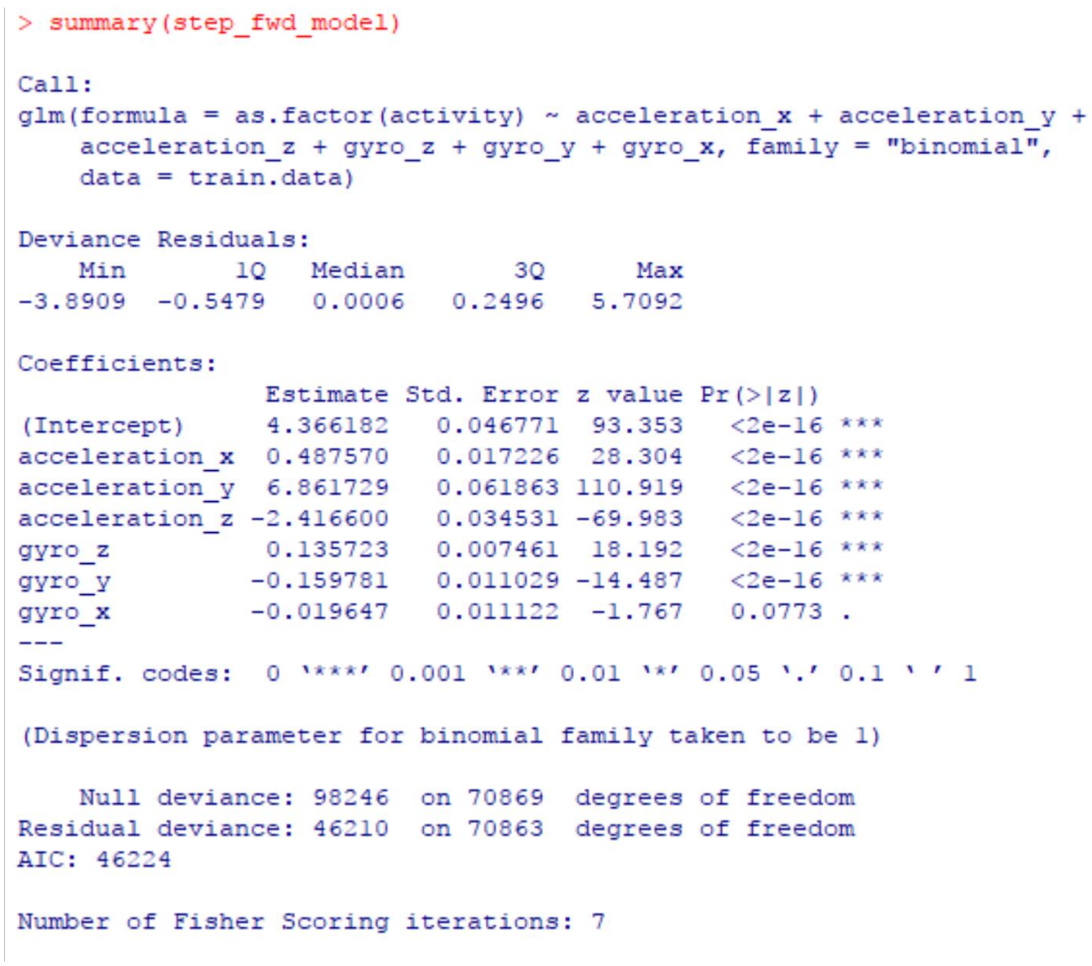
Starts with all of the predictors in the model. The variable that is least significant--that is, the one with the largest P value--is removed and the model is refitted. Each subsequent step removes the least significant variable in the model until all remaining variables have individual P values smaller than some value, such as 0.05



```
> summary(step_backwd_model)

Call:
glm(formula = as.factor(activity) ~ acceleration_x + acceleration_y +
    acceleration_z + gyro_x + gyro_y + gyro_z, family = binomial(),
    data = train.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.8909  -0.5479   0.0006   0.2496   5.7092

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     4.366182   0.046771  93.353   <2e-16 ***
acceleration_x  0.487570   0.017226  28.304   <2e-16 ***
acceleration_y  6.861729   0.061863 110.919   <2e-16 ***
acceleration_z -2.416600   0.034531 -69.983   <2e-16 ***
gyro_x         -0.019647   0.011122  -1.767   0.0773 .
gyro_y         -0.159781   0.011029 -14.487   <2e-16 ***
gyro_z          0.135723   0.007461  18.192   <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: 98246  on 70869  degrees of freedom
Residual deviance: 46210  on 70863  degrees of freedom
AIC: 46224

Number of Fisher Scoring iterations: 7
```

iii. STEPWISE FOWARD REGRESSION

Starts with an empty model. The variable that has the smallest P value when it is the only predictor in the regression equation is placed in the model. Each subsequent step adds the variable that has the smallest P value in the presence of the predictors already in the equation. Variables are added one-at-a-time as long as their P values are small enough, typically less than 0.05.



```
> summary(step_fwd_model)

Call:
glm(formula = as.factor(activity) ~ acceleration_x + acceleration_y +
    acceleration_z + gyro_z + gyro_y + gyro_x, family = "binomial",
    data = train.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.8909  -0.5479   0.0006   0.2496   5.7092

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     4.366182   0.046771  93.353   <2e-16 ***
acceleration_x  0.487570   0.017226  28.304   <2e-16 ***
acceleration_y  6.861729   0.061863 110.919   <2e-16 ***
acceleration_z -2.416600   0.034531 -69.983   <2e-16 ***
gyro_z          0.135723   0.007461  18.192   <2e-16 ***
gyro_y         -0.159781   0.011029 -14.487   <2e-16 ***
gyro_x         -0.019647   0.011122  -1.767   0.0773 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 98246  on 70869  degrees of freedom
Residual deviance: 46210  on 70863  degrees of freedom
AIC: 46224

Number of Fisher Scoring iterations: 7
```

Additional Classification models were built to test which model in best,

To build other model for **problem 4**, we can use Classification methods like,

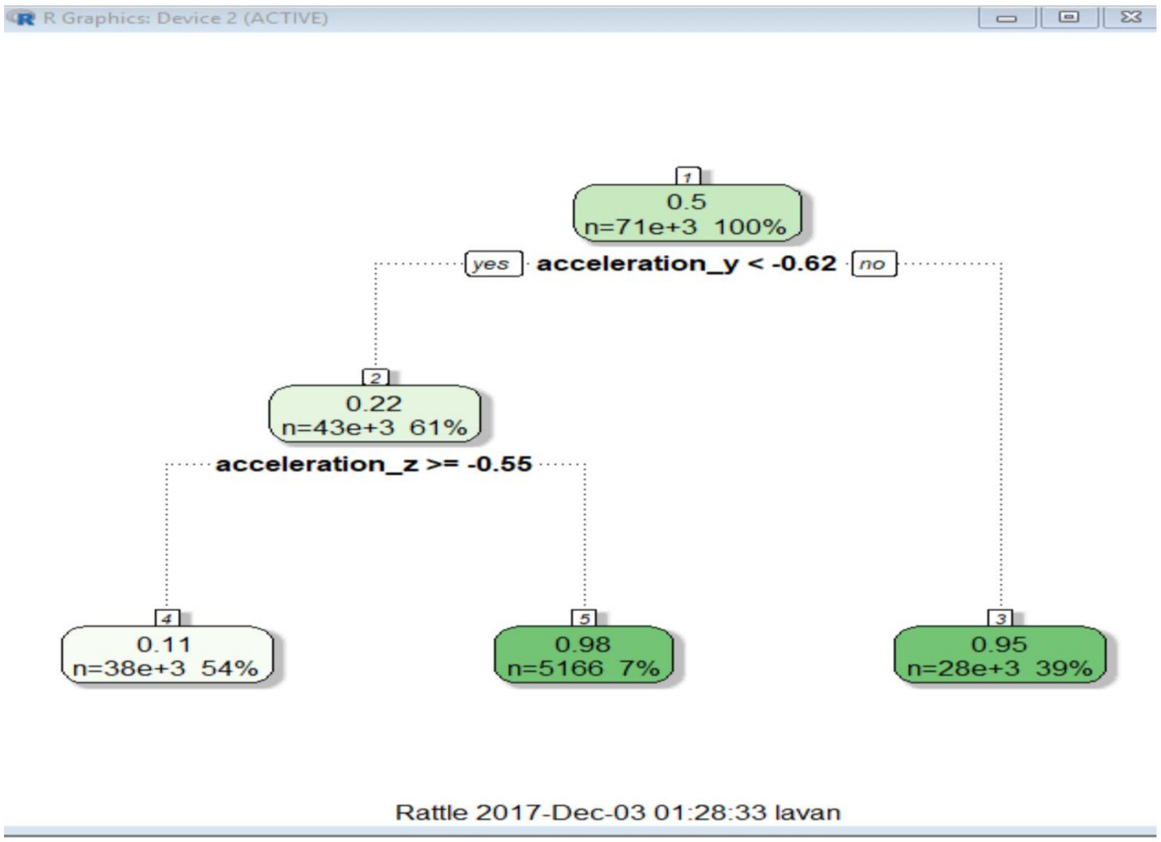
i. DECISION TREES

ii. RANDOM FOREST

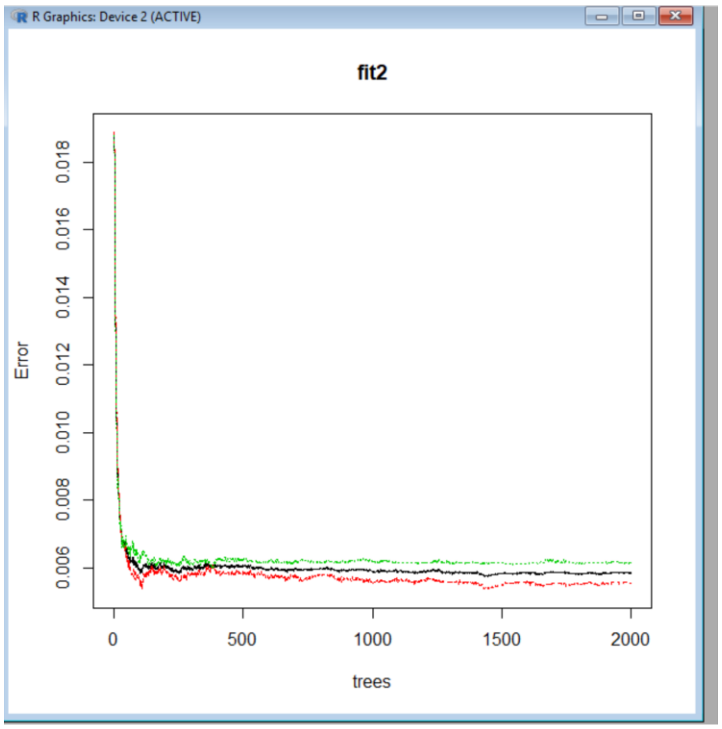
iii. k-Nearest Neighbor Predictions

i. DECISION TREES

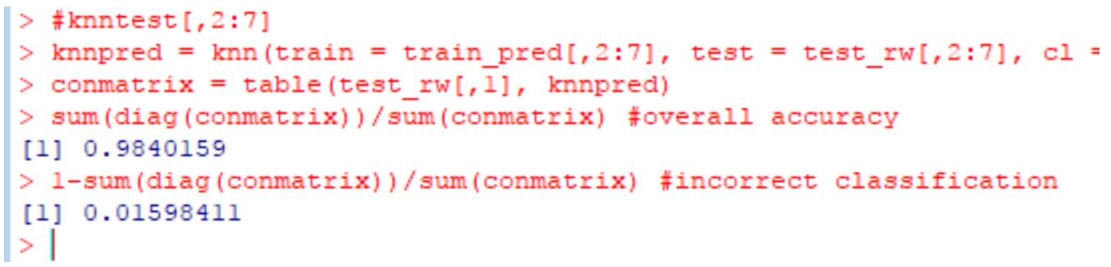
A decision tree is a graph that uses a branching method to illustrate every possible outcome of a decision



ii. RANDOM FOREST



iii. k-Nearest Neighbor Predictions



```
> #knntest[,2:7]
> knnpred = knn(train = train_pred[,2:7], test = test_rw[,2:7], cl =
> conmatrix = table(test_rw[,1], knnpred)
> sum(diag(conmatrix))/sum(conmatrix) #overall accuracy
[1] 0.9840159
> 1-sum(diag(conmatrix))/sum(conmatrix) #incorrect classification
[1] 0.01598411
> |
```

6. Evaluations and Results

6.1. Evaluation Methods

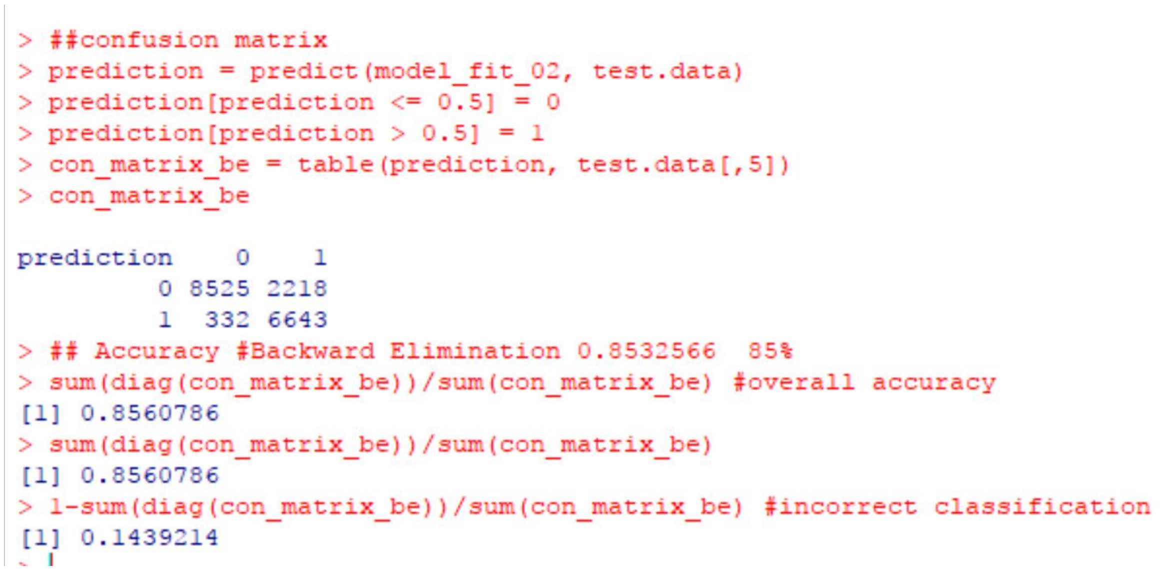
The model evaluation methods which I am using are liste das below,

1. **Confusion matrix(Accuracy)**
2. **AIC & Adj Rsqaure**
3. **Hosmer and Lemeshow goodness of fit (GOF) test**
4. **chi square**
5. **ROC curve and the AUC**
6. **Overdispersion**
7. **threshold value(cut off)**

6.1.1 Confusion **matrix(Accuracy)**

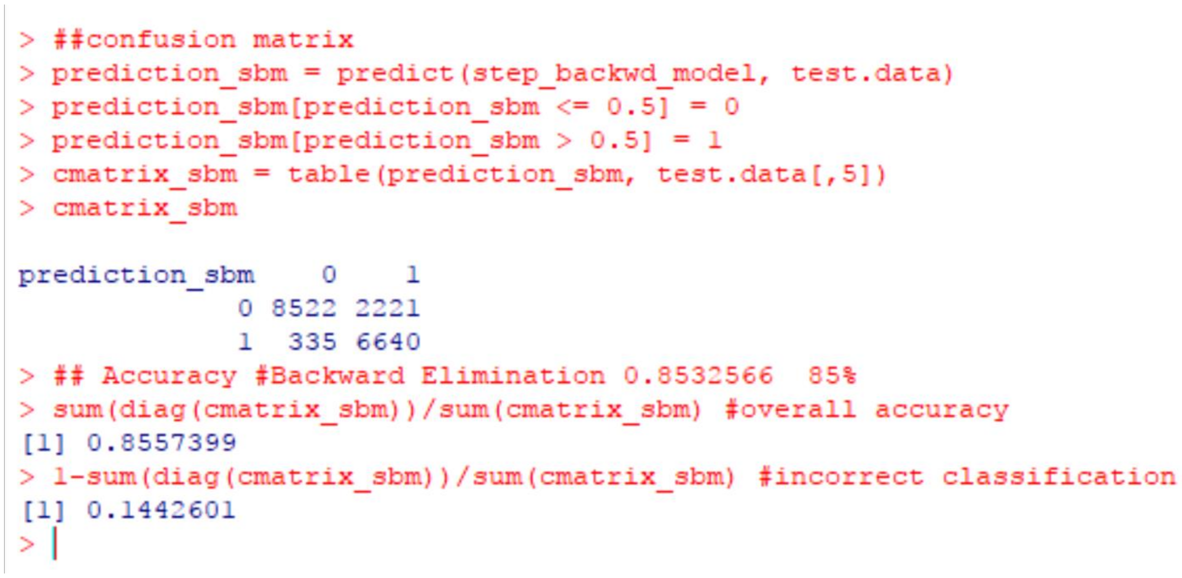
This is used to measure the accuracy:

1. **Backward Elimination 85%**



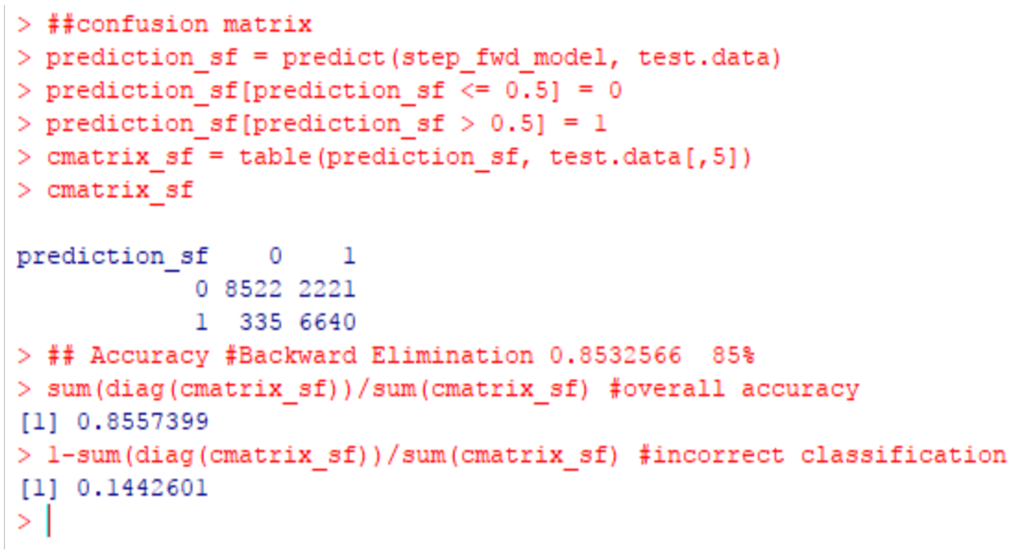
```
> ##confusion matrix
> prediction = predict(model_fit_02, test.data)
> prediction[prediction <= 0.5] = 0
> prediction[prediction > 0.5] = 1
> con_matrix_be = table(prediction, test.data[,5])
> con_matrix_be

prediction    0    1
         0 8525 2218
         1  332 6643
> ## Accuracy #Backward Elimination 0.8532566  85%
> sum(diag(con_matrix_be))/sum(con_matrix_be) #overall accuracy
[1] 0.8560786
> sum(diag(con_matrix_be))/sum(con_matrix_be)
[1] 0.8560786
> 1-sum(diag(con_matrix_be))/sum(con_matrix_be) #incorrect classification
[1] 0.1439214
```

2. **Step wise backward selection** 85%



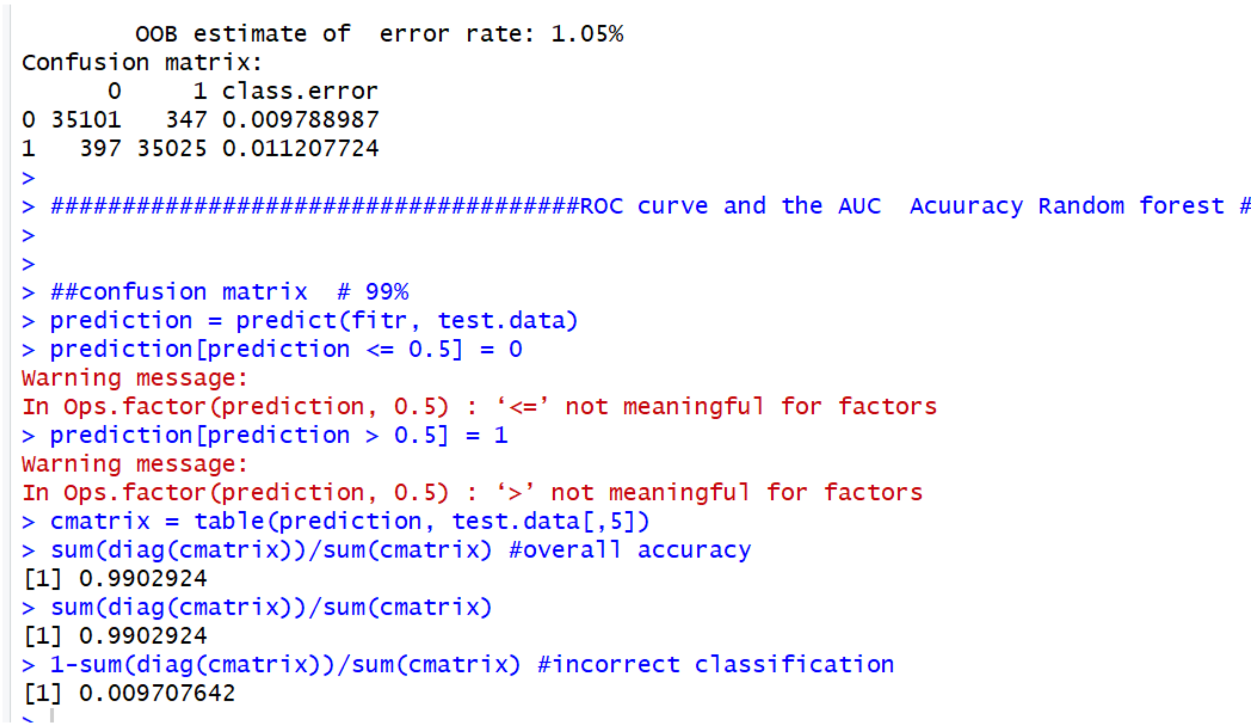
```
> ##confusion matrix
> prediction_sbm = predict(step_backwd_model, test.data)
> prediction_sbm[prediction_sbm <= 0.5] = 0
> prediction_sbm[prediction_sbm > 0.5] = 1
> cmatrix_sbm = table(prediction_sbm, test.data[,5])
> cmatrix_sbm

prediction_sbm    0    1
             0 8522 2221
             1  335 6640
> ## Accuracy #Backward Elimination 0.8532566  85%
> sum(diag(cmatrix_sbm))/sum(cmatrix_sbm) #overall accuracy
[1] 0.8557399
> 1-sum(diag(cmatrix_sbm))/sum(cmatrix_sbm) #incorrect classification
[1] 0.1442601
> 
```

3. **step wise Forward 85%**



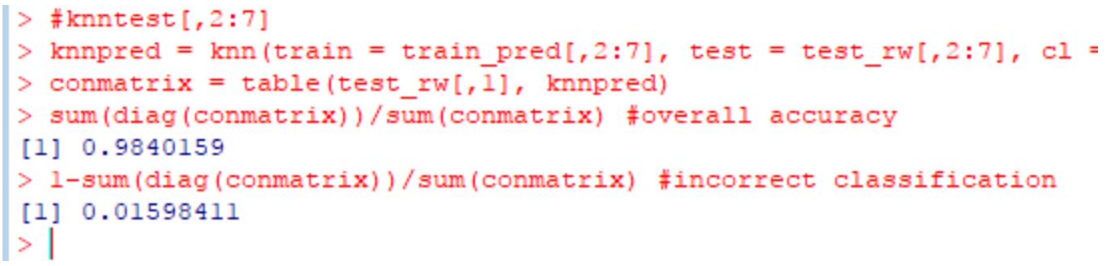
```
> ##confusion matrix
> prediction_sf = predict(step_fwd_model, test.data)
> prediction_sf[prediction_sf <= 0.5] = 0
> prediction_sf[prediction_sf > 0.5] = 1
> cmatrix_sf = table(prediction_sf, test.data[,5])
> cmatrix_sf

prediction_sf    0    1
            0 8522 2221
            1  335 6640
> ## Accuracy #Backward Elimination 0.8532566  85%
> sum(diag(cmatrix_sf))/sum(cmatrix_sf) #overall accuracy
[1] 0.8557399
> 1-sum(diag(cmatrix_sf))/sum(cmatrix_sf) #incorrect classification
[1] 0.1442601
> 
```

4. **Random Forest 99%**


```
        OOB estimate of  error rate: 1.05%
Confusion matrix:
      0     1 class.error
0 35101   347 0.009788987
1   397 35025 0.011207724
>
> #####################################ROC curve and the AUC  Acuuracy Random forest #
>
>
> ##confusion matrix  # 99%
> prediction = predict(fitr, test.data)
> prediction[prediction <= 0.5] = 0
Warning message:
In Ops.factor(prediction, 0.5) : ‘<=’ not meaningful for factors
> prediction[prediction > 0.5] = 1
Warning message:
In Ops.factor(prediction, 0.5) : ‘>’ not meaningful for factors
> cmatrix = table(prediction, test.data[,5])
> sum(diag(cmatrix))/sum(cmatrix) #overall accuracy
[1] 0.9902924
> sum(diag(cmatrix))/sum(cmatrix)
[1] 0.9902924
> 1-sum(diag(cmatrix))/sum(cmatrix) #incorrect classification
[1] 0.009707642
```

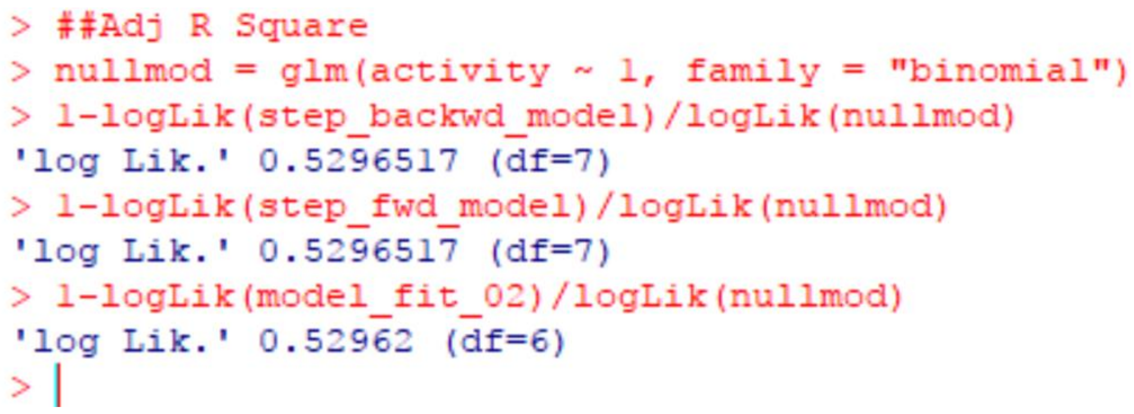
5. **Decision Trees ~ 94%**

6. **KNN 98%**


```
> #knntest[,2:7]
> knnpred = knn(train = train_pred[,2:7], test = test_rw[,2:7], cl =
> conmatrix = table(test_rw[,1], knnpred)
> sum(diag(conmatrix))/sum(conmatrix) #overall accuracy
[1] 0.9840159
> 1-sum(diag(conmatrix))/sum(conmatrix) #incorrect classification
[1] 0.01598411
>
```

Hence based on accuracy Random forest comes first with 99% accuracy

6.1.2 **AIC & Adj Rsqaure**



```
> ##Adj R Square
> nullmod = glm(activity ~ 1, family = "binomial")
> 1-logLik(step_backwd_model)/logLik(nullmod)
'log Lik.' 0.5296517 (df=7)
> 1-logLik(step_fwd_model)/logLik(nullmod)
'log Lik.' 0.5296517 (df=7)
> 1-logLik(model_fit_02)/logLik(nullmod)
'log Lik.' 0.52962 (df=6)
> |
```

Backward Elimination

AIC 46225

Adj Rsqar 0.5296

Step wise backward selections

AIC 46224

Adj Rsqar 0. 5296

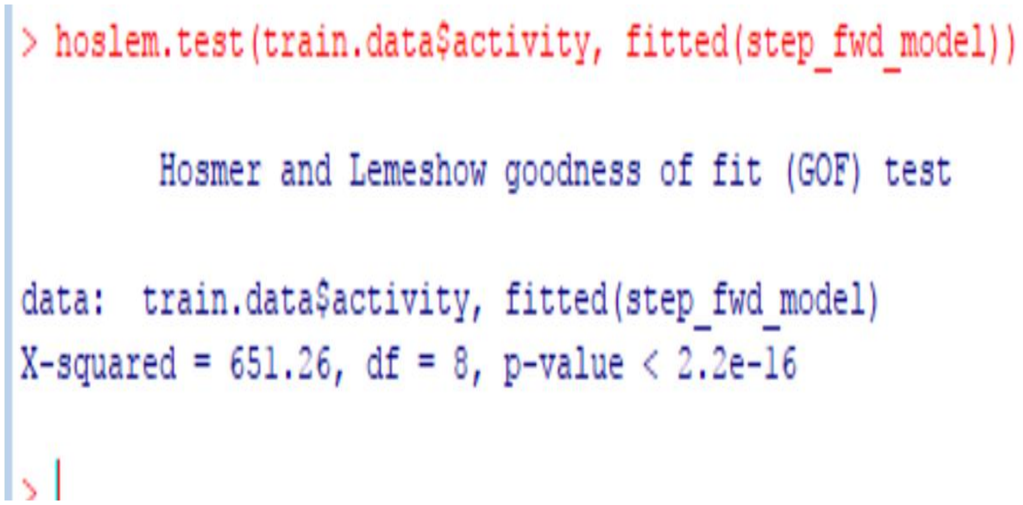
Step wise Forward

AIC 46224

Adj Rsqar 0. 5296

As AIC is lower by one for step wise forward and step wise backward compared to backward elimination. Either stepwise forward or step wise backward is chosen as the fit model for **linear regression**

6.1.3 **Hosmer and Lemeshow goodness of fit (GOF)**
This test is to validate the logical regression fit model(stepwise forward**)**

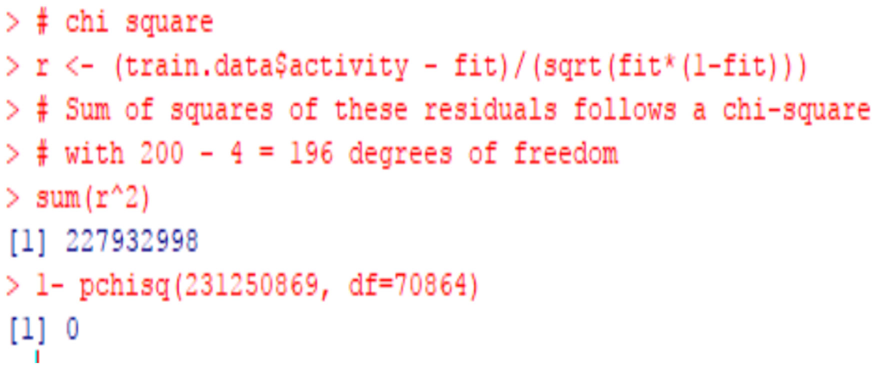


```
> hoslem.test(train.data$activity, fitted(step_fwd_model))

        Hosmer and Lemeshow goodness of fit (GOF) test

data:  train.data$activity, fitted(step_fwd_model)
X-squared = 651.26, df = 8, p-value < 2.2e-16

>
```

The result of the test does not confirm the model is correct as we have significant difference between the model and the observed data (i.e. the p-value is below 0.05). hence we move forward with some other test .

6.1.4 **chi square**

This test is to validate the logical regression fit model(stepwise forward**)**

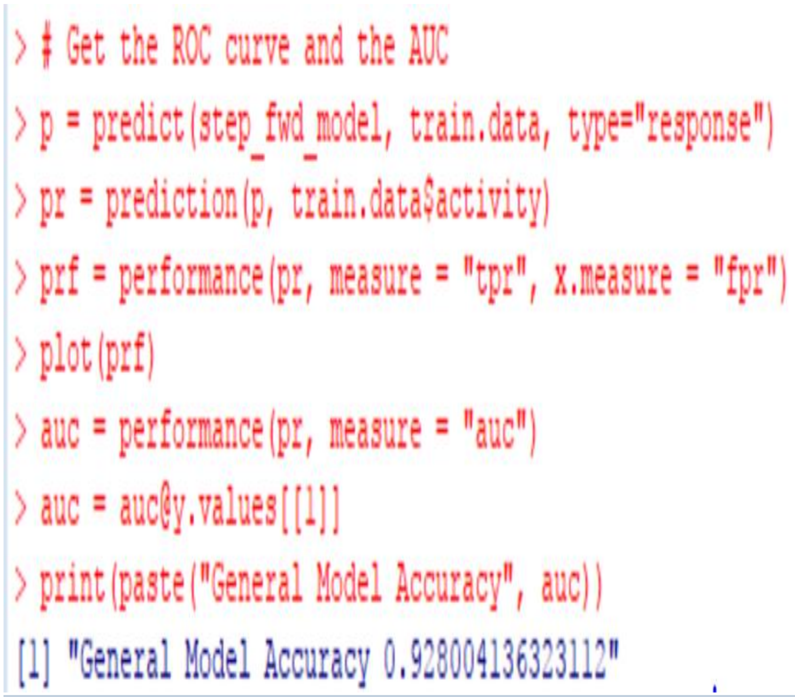


```
> # chi square
> r <- (train.data$activity - fit)/(sqrt(fit*(1-fit)))
> # Sum of squares of these residuals follows a chi-square
> # with 200 - 4 = 196 degrees of freedom
> sum(r^2)
[1] 227932998
> 1- pchisq(231250869, df=70864)
[1] 0
```

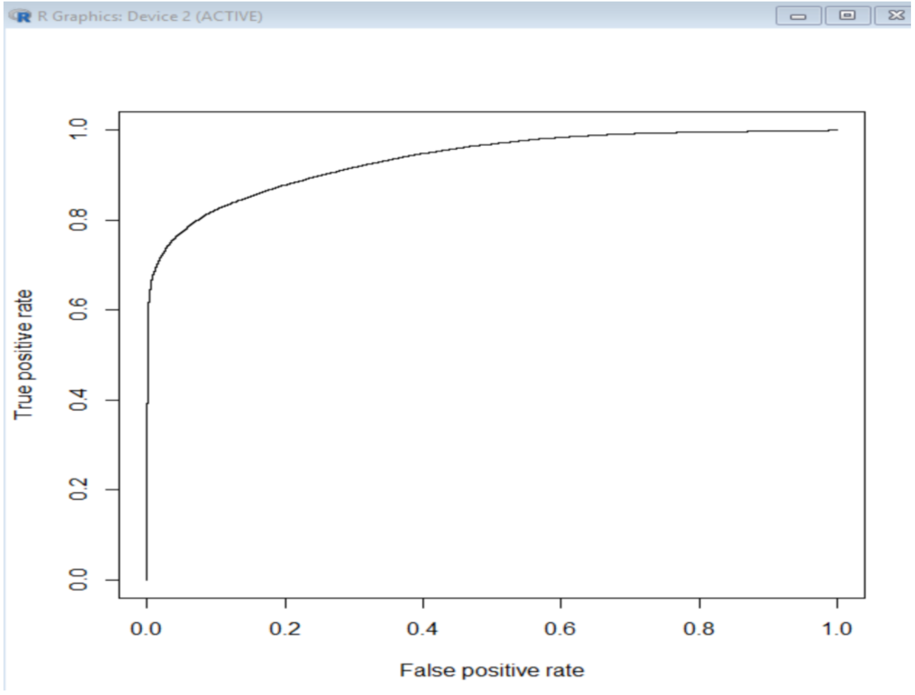
The P-Value is < 0.00001. The result is significant at $p < 0.05$. Null Hypothesis is true (which means there really is no difference)

6.1.5 **ROC curve and the AUC**

1. **Logistic regression**

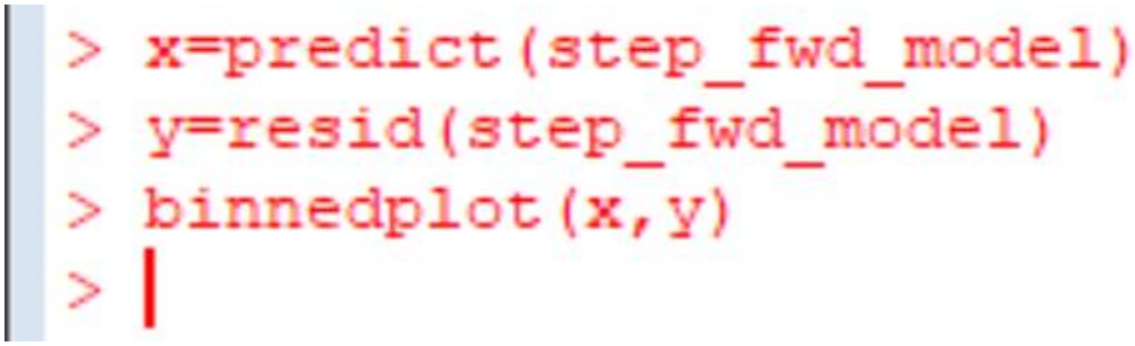


```
> # Get the ROC curve and the AUC
> p = predict(step_fwd_model, train.data, type="response")
> pr = prediction(p, train.data$activity)
> prf = performance(pr, measure = "tpr", x.measure = "fpr")
> plot(prf)
> auc = performance(pr, measure = "auc")
> auc = auc@y.values[[1]]
> print(paste("General Model Accuracy", auc))
[1] "General Model Accuracy 0.928004136323112"
```



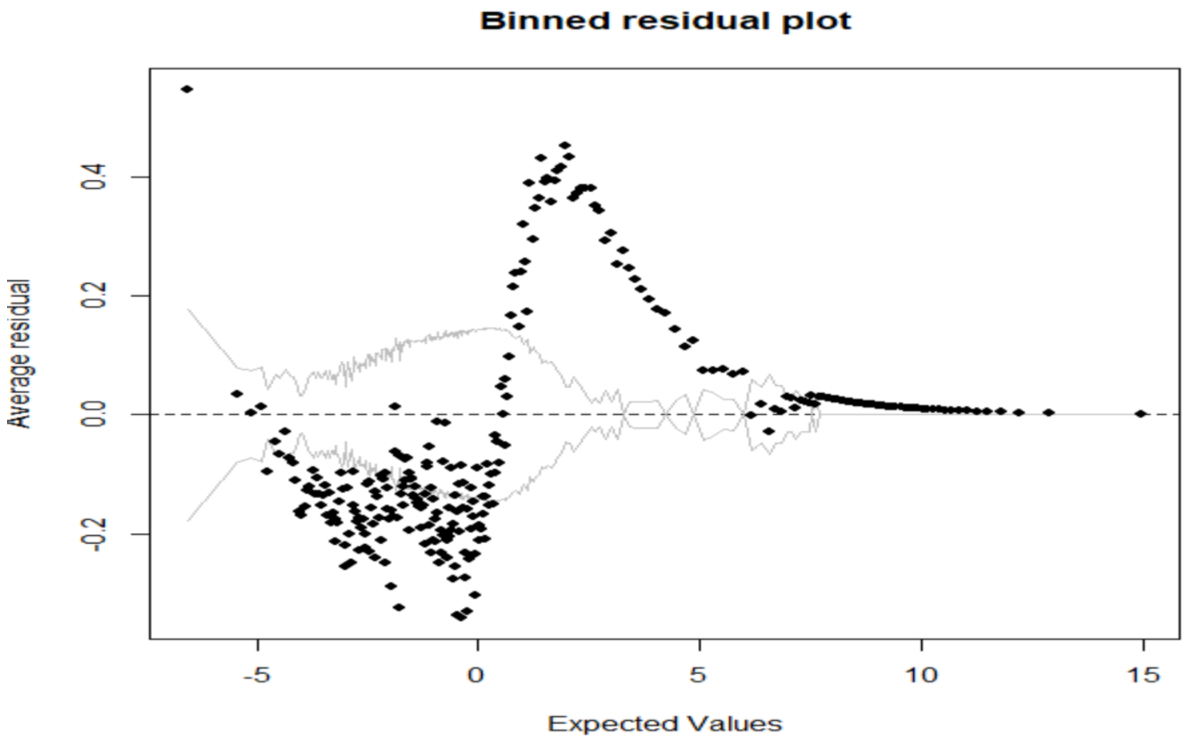
The guaranties us that our logistic step wise forward model is a correct model

6.1.6 **Overdispersion**



```
> x=predict(step_fwd_model)
> y=resid(step_fwd_model)
> binnedplot(x,y)
> |
```

1 For logistic regression model



Hence the curve is not above 2 or -2 which says our model is correct.

6.2. Results and Findings

AS we see from the methods used to evaluate the Predicted models above,

All the 4 models are fit models

- ➔ **Logistic Regression**
  - Step wise Forward
- ➔ **Decision Trees**
- ➔ **Random Forest**
- ➔ ***k*-Nearest Neighbor Predictions**

But when we compare which is the best among them, we can conclude that Random forest is the best model as it has the highest confusion matrix Accuracy of 99 %(**Random Forest** 99%)

Hence the best model of all is the **Random forest model**.

7. Conclusions and Future Work

7.1. Conclusions

As Proposed the main outcome of this project is to predict whether the person is running or walking based on deep neural network and sensor data collected from iOS device was predicted using four models over which the best model which can predict with 99% of accuracy was the Random forest model.

This dataset seemed to be relay interesting, as we can do wrist prediction as well as to which wrist the user is wearing the watch which has the sensors.

7.2. Limitations

It was overall a good data set, I did not find any limitations over this.

7.3. Potential Improvements or Future Work

In future, Collection and Processing of Data from Wrist Wearable Devices in Heterogeneous and Multiple-User Scenarios can be predicted based on this scenario.