Classification

Valari Graham and Jeffrey Li

2022-09-20

This notebook performs linear classification on the **Accelerometer** data set from the UCI Machine Learning Repository. A link to the data can be found here (https://archive.ics.uci.edu/ml/datasets/Accelerometer)

Linear Models for Classification

Linear classification in R is a model that makes decisions to place an observation in a discrete class based on its explanatory variables. Advantages of using classification include the simplicity and efficiency of the models. Some disadvantages of classification is that the model can be prone to over fitting if the number of observations is less than the number of features.

Read in Data Set and Split

```
acc <- read.csv("accelerometer.csv", stringsAsFactors = T)
sample <- sample(1:nrow(acc), nrow(acc)*0.8, replace=FALSE)
train <- acc[sample,]
test <- acc[-sample,]
```

Exploratory Analysis using 5 Functions

```
# First 6 rows of data
head(train)
```

```
##        wconfid pctid     x      y      z
## 147821       3    95 1.070  0.242  0.035
## 109469       3    30 0.980  0.105 -0.156
## 4875         1    25 0.957 -0.129 -0.156
## 24062        1    60 1.023 -0.168 -0.016
## 151995       3   100 0.840 -0.094 -0.328
## 32817        1    70 1.094 -0.375 -0.359
```

```
# Find summary of the training data
summary(train)
```

```
##     wconfid      pctid             x                y
##  Min.   :1   Min.   : 20.00   Min.   :-8.0000   Min.   :-8.00000
##  1st Qu.:1   1st Qu.: 40.00   1st Qu.: 0.9450   1st Qu.:-0.07800
##  Median :2   Median : 60.00   Median : 0.9920   Median : 0.00800
##  Mean   :2   Mean   : 59.98   Mean   : 0.9952   Mean   : 0.00616
##  3rd Qu.:3   3rd Qu.: 80.00   3rd Qu.: 1.0390   3rd Qu.: 0.10500
##  Max.   :3   Max.   :100.00   Max.   : 7.9960   Max.   : 7.99200
##        z
##  Min.   :-5.8670
##  1st Qu.:-0.1720
##  Median :-0.1250
##  Mean   :-0.1185
##  3rd Qu.:-0.0660
##  Max.   : 6.0860
```

```
# Find null values in the data set
sum(is.na(train))
```

```
## [1] 0
```

```
# Find total count of each class (red, blue, green)
table(acc$wconfid)
```

```
##
##     1     2     3
## 51000 51000 51000
```

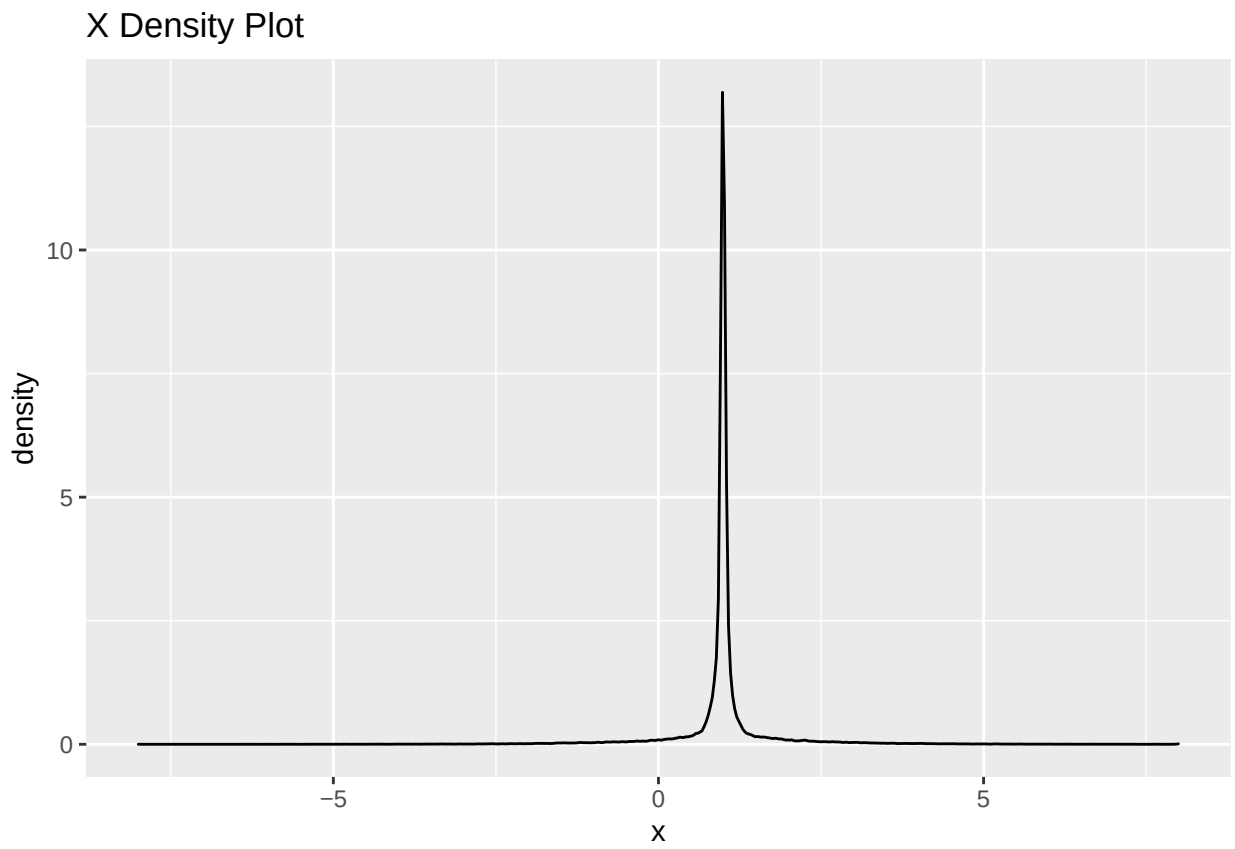
```
# Check for correlation between independent variables
cor(train[, c('x', 'y', 'z')])
```

```
##             x           y           z
## x  1.00000000  0.02361876 -0.08682264
## y  0.02361876  1.00000000 -0.02700724
## z -0.08682264 -0.02700724  1.00000000
```

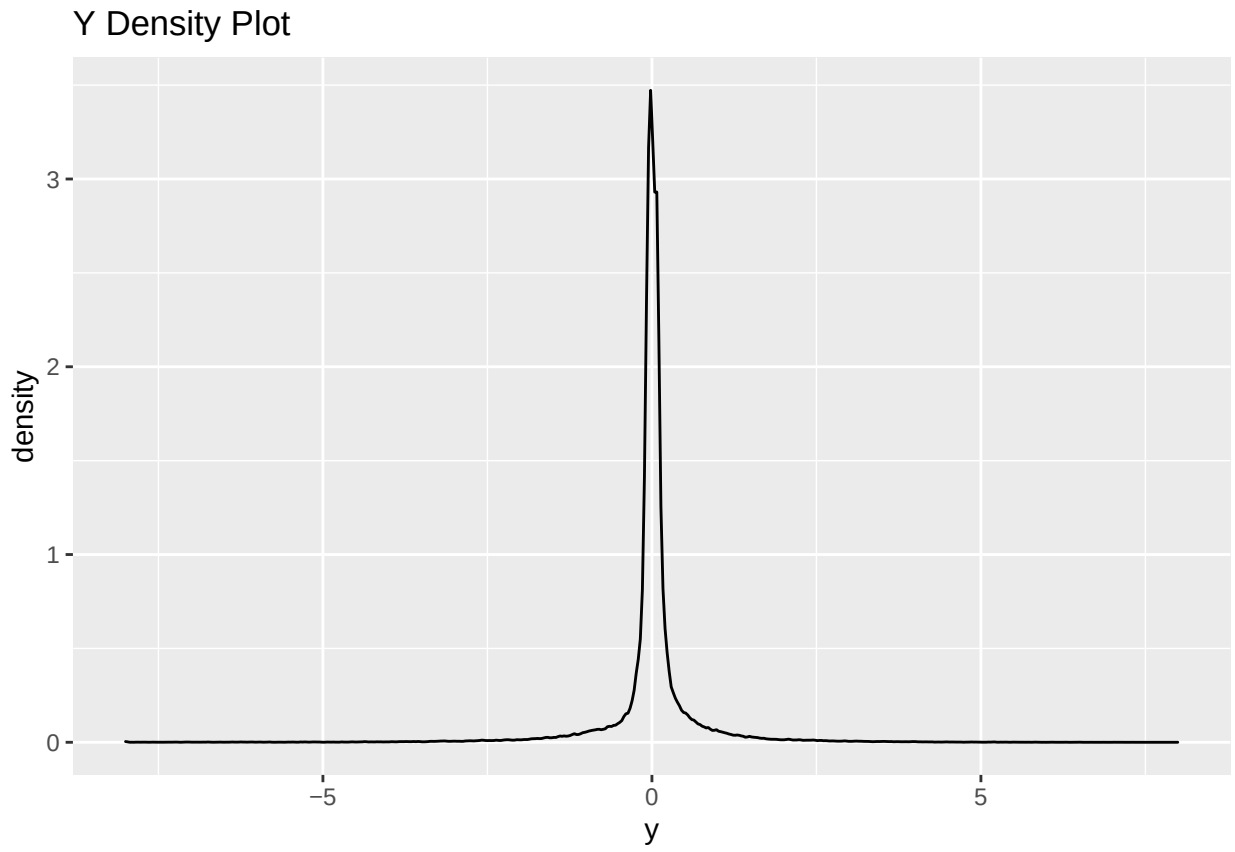
Visuals

Density Plot

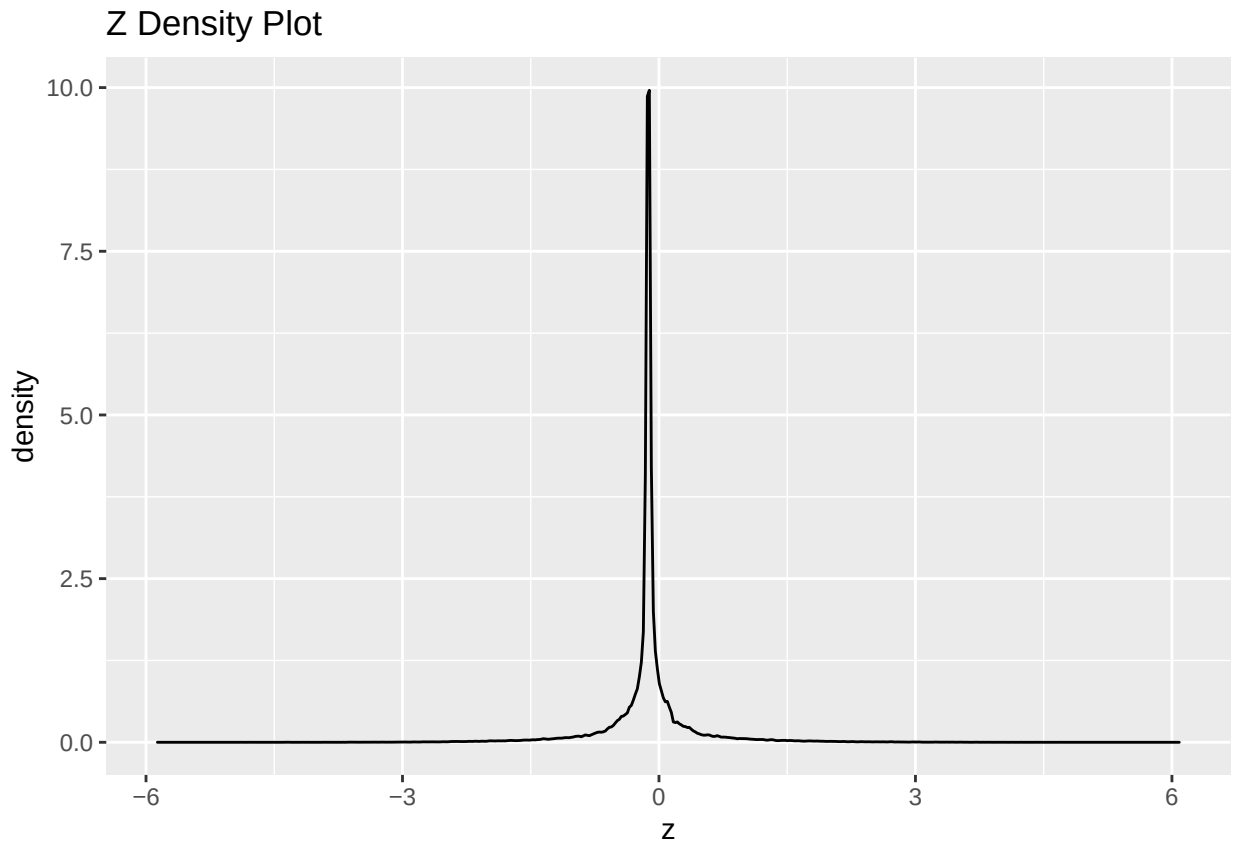
```
library(ggplot2)
# Density plot of each accelerometer value compared
ggplot(train, aes(x = x)) + geom_density() + ggtitle("X Density Plot")
```




```
ggplot(train, aes(x = y)) + geom_density() + ggtitle("Y Density Plot")
```

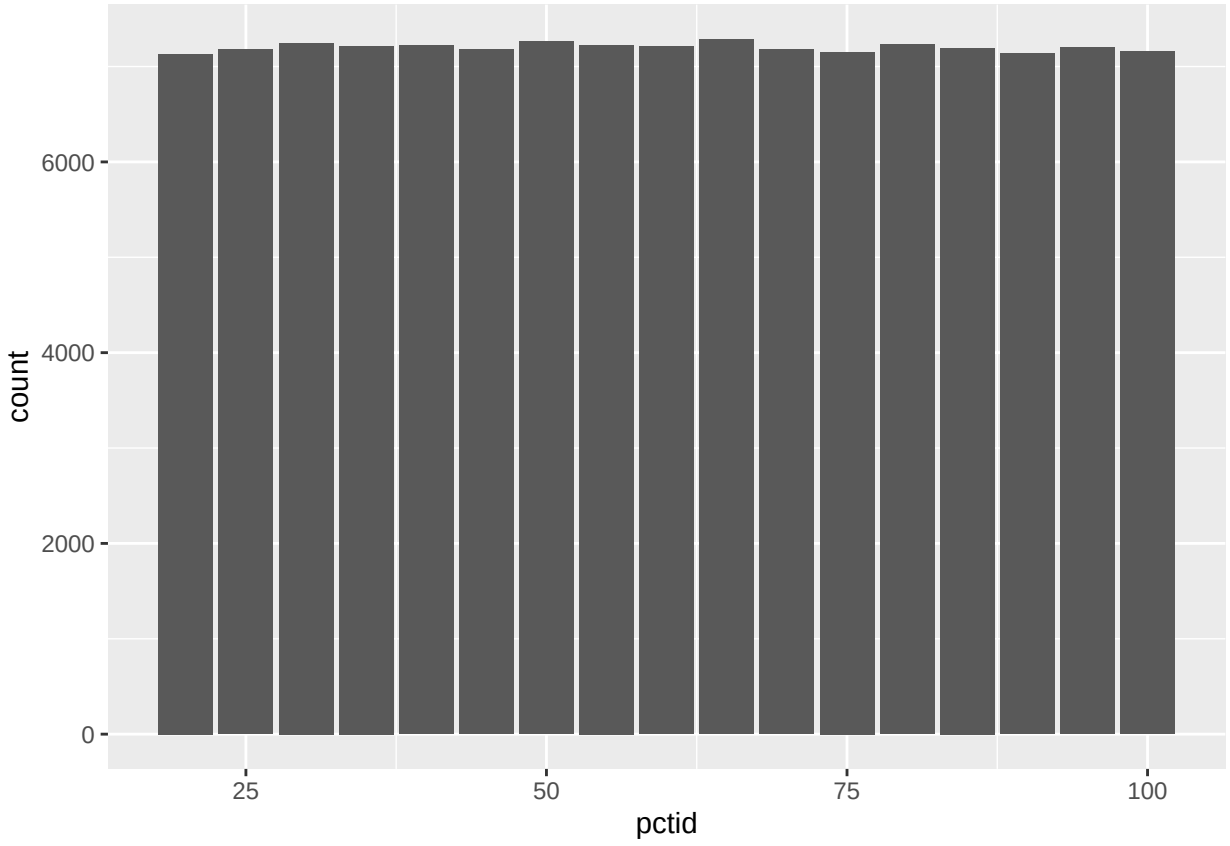



```
ggplot(train, aes(x = z)) + geom_density() + ggtitle("Z Density Plot")
```



Bar Plot

```
# Overview of cooler fan speed percentage
ggplot(train, aes(x = pctid)) +geom_bar()
```



Reclassfication

```
# Reclassify as normal configuration or not (Red)
acc_red <- acc
acc_red$wconfid <- as.factor(ifelse (acc_red$wconfid == 1, 1, 0))

# Reclassify as perpendicular configuration or not (Blue)
acc_blue <- acc
acc_blue$wconfid <- as.factor(ifelse (acc_blue$wconfid == 2, 1, 0))

# Reclassify as opposite configuration or not (Green)
acc_green <- acc
acc_green$wconfid <- as.factor(ifelse (acc_green$wconfid == 3, 1, 0))
```

Logistic Regression Model

```
library(caret)
```

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

```r
fun <- function(df, i){
    train <- df[i,]
    test <- df[-i,]
    glm1 <- glm(wconfid~., data=train, family="binomial")
    probs <- predict(glm1, newdata=test)
    pred <- ifelse(probs>0.5, 1, 0)
    acc <- mean(pred==test$wconfid)
    print(paste("accuracy = ", acc))
    table(pred, test$wconfid)
}
```

```r
set.seed(1234)
i <- sample(1:150, 100, replace=FALSE)
# Evaluate test set on normal configuration
fun(acc_red, i)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
## [1] "accuracy =  0.667102681491171"
```

```
##
## pred      0      1
##    0 102000  50900
```

```r
glm_red <- glm(wconfid~ pctid + x + y + z, data = acc_red, family = 'binomial')
summary(glm_red)
```

```
##
## Call:
## glm(formula = wconfid ~ pctid + x + y + z, family = "binomial",
##     data = acc_red)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9999  -0.9009  -0.8996   1.4803   1.6061
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.062e-01  1.594e-02 -44.291  < 2e-16 ***
## pctid        1.311e-06  2.214e-04   0.006  0.99528
## x            1.022e-02  7.039e-03   1.452  0.14649
## y           -2.583e-02  7.284e-03  -3.547  0.00039 ***
## z           -2.423e-02  1.054e-02  -2.299  0.02153 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 194773  on 152999  degrees of freedom
## Residual deviance: 194753  on 152995  degrees of freedom
## AIC: 194763
```

```
##
## Number of Fisher Scoring iterations: 4
```

```
# Evaluate test set on perpendicular configuration
fun(acc_blue, i)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
## [1] "accuracy =  0.666448659254415"
```

```
##
## pred      0      1
##    0 101900  51000
```

```
glm_blue <- glm(wconfid~ pctid + x + y + z, data = acc_blue, family = 'binomial')
summary(glm_blue)
```

```
##
## Call:
## glm(formula = wconfid ~ pctid + x + y + z, family = "binomial",
##     data = acc_blue)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0357  -0.9009  -0.8995   1.4799   1.5942
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.964e-01  1.596e-02 -43.639  < 2e-16 ***
## pctid       -1.502e-05  2.215e-04  -0.068 0.945917
## x            8.085e-03  7.042e-03   1.148 0.250906
## y            2.451e-02  7.342e-03   3.338 0.000844 ***
## z            3.540e-02  1.054e-02   3.359 0.000783 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 194773  on 152999  degrees of freedom
## Residual deviance: 194751  on 152995  degrees of freedom
## AIC: 194761
##
## Number of Fisher Scoring iterations: 4
```

```
# Evaluate test set on opposite configuration
fun(acc_green, i)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
## [1] "accuracy =  0.666448659254415"
```

```
##
## pred      0      1
##    0 101900  51000
```

```
glm_green <- glm(wconfid~ pctid + x + y + z, data = acc_green, family = 'binomial')
summary(glm_green)
```

```
##
## Call:
## glm(formula = wconfid ~ pctid + x + y + z, family = "binomial",
##     data = acc_green)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9714  -0.9007  -0.8998   1.4820   1.4868
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.763e-01  1.598e-02 -42.316  < 2e-16 ***
## pctid        1.316e-06  2.214e-04   0.006  0.99526
## x           -1.834e-02  7.042e-03  -2.604  0.00922 **
## y            1.543e-03  7.313e-03   0.211  0.83291
## z           -1.120e-02  1.055e-02  -1.062  0.28811
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 194773  on 152999  degrees of freedom
## Residual deviance: 194766  on 152995  degrees of freedom
## AIC: 194776
##
## Number of Fisher Scoring iterations: 4
```

The logistic regression model built predicts how well the given observations fit to the categories 1, 2, and 3. In logistic regression, the model assume the data following a sigmoid function. For example with the normal configuration, when using a decision threshold, the model chooses whether the observation belongs to a normal configuration or not. This process was repeated above with the perpendicular configuration and opposite configuration as well. Looking at the results provided, it shows that each prediction of the test data held above a 60% accuracy level. Any accuracy less than 50% and we begin to question the significance of the model. Each prediction table produced similar results.

Naive Bayes classifier

```
library(e1071)
library(caTools)
# Building Naive Bayes model and training

classifier_cl <- naiveBayes(wconfid ~ ., data = train)
classifier_cl
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         1         2         3
## 0.3331209 0.3334232 0.3334559
##
## Conditional probabilities:
##    pctid
## Y       [,1]     [,2]
##   1 59.98909 24.46951
##   2 59.95504 24.44981
##   3 59.98248 24.46245
##
##    x
## Y        [,1]       [,2]
##   1 0.9989023 1.08041569
##   2 0.9977655 0.78423198
##   3 0.9888479 0.07736055
##
##    y
## Y           [,1]       [,2]
##   1 -0.001271693 1.07682734
##   2  0.014063831 0.68920511
##   3  0.005681588 0.08501723
##
##    z
## Y         [,1]       [,2]
##   1 -0.1240170 0.69052193
##   2 -0.1122504 0.55929228
##   3 -0.1191046 0.08797411
```

```
# Predicting on test data
y_pred <- predict(classifier_cl, newdata = test)
```

```
# Confusion matrix
cm <- table(test$wconfid, y_pred)
cm
```

```
##    y_pred
##        1    2    3
##   1 2474 2668 5084
##   2 1679 3345 5165
##   3    0  595 9590
```

```
# Overall model evaluation
confusionMatrix(cm)
```

```
## Confusion Matrix and Statistics
```

```
##
##    y_pred
##        1    2    3
##   1 2474 2668 5084
##   2 1679 3345 5165
##   3    0  595 9590
##
## Overall Statistics
##
##                Accuracy : 0.5036
##                  95% CI : (0.4979, 0.5092)
##     No Information Rate : 0.6483
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2557
##
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
## Sensitivity           0.59571   0.5062   0.4834
## Specificity           0.70689   0.7147   0.9447
## Pos Pred Value        0.24193   0.3283   0.9416
## Neg Pred Value        0.91759   0.8401   0.4980
## Prevalence            0.13572   0.2159   0.6483
## Detection Rate        0.08085   0.1093   0.3134
## Detection Prevalence  0.33418   0.3330   0.3328
## Balanced Accuracy     0.65130   0.6105   0.7140
```

The Naive Bayes regression is a non-linear classification model. This model applies Bayes' theorem with strong independence assumptions on the features. Based on the combination of the features, the model decides the weight configuration scenario of the fan on the accelerometer. Compared to the logistic regression, we see the accuracy of the model drop. The Naive Bayes model holds an accuracy of 0.4953, which is concerning that it is below 50%. These low results could be a cause of any dependency among the features. Specificity is quite high for all three classes. This means the model predicted a high number of true negatives and that the model was better at predicting when an observation was not in a class. Overall, the logistic regression model was the better fit for this classification due to the better accuracy and prediction of observations.

Advantages vs. Disadvantages

Certain advantages of logistic regression are seen in the implementation of the model itself. The model is efficient and easy to use, which makes it a more comfortable choice for beginners. Logistic regression also can easily extend multiple classes making it a good fit for multinomial classification. On the other hand, a disadvantage of it is the assumption of linearity between the dependent variable and predictors. The model also requires little to no multicollinearity between independent variables. concerning Naive Bayes, an advantage is the conditional probabilities are easier to evaluate. The model itself is also efficient and gives great results when the assumptions hold. The disadvantages of this model are that it requires the conditional independence assumption does not always follow through. As for most cases, the predictors show some form of a dependent relationship. Another disadvantage is that the model is better suited for categorical variables than numerical.

Classification Metrics

As seen above, the classification metrics include the confusion matrix, accuracy, specificity, sensitivity, precision and the f1-score. Accuracy, specificity and sensitivity are helpful in that they reassure validity of the results. We would rather see these values higher as a means of showing the model was overall performing well on the test data. The limitations with accuracy are as follows: class imbalance and misleading accuracy when model predicts just one class.The confusion matrix shows the actual class vs. the predicted matrix. The advantage of this matrix is that is shows in a visual perspective how the model did overall in assign observations to the correct class. Similar to accuracy, the confusion matrix has a disadvantage when the model predicts that each point belongs to a major class and this causes imbalance. As opposed to accuracy, the f1 score gives more importance to understanding false negatives and false positives. F1-score is a better metric when there are imbalanced classes.