DAT-4253 LM 4.2 - NB

AUTHOR
Aaron Younger

```
options(scipen = 999)
suppressWarnings(RNGversion("3.5.3"))
```

```
##Libraries
library(readxl)
library(tidyverse)
library(DataExplorer)
library(SmartEDA)
library(ggplot2)
library(caret)
library(pROC)
library(gains)
library(klaR)
library(gmodels)
```

```
library(readxl)
Online_Retail <- read_excel("jaggia_ba_2e_ch12_data.xlsx", sheet = "OnlineRetail")
View(Online_Retail)
```

Business Understanding

An online retailer is offering a new line of running shoes. The retailer plans to send out an e-mail with a discount offer to some of its existing customers and wants to know if it can use data mining analysis to predict whether or not a customer might respond to its e-mail offer. The retailer prepares the accompanying data file of 170 existing customers who had received online promotions in the past, which includes the following variables: Purchase (1 if purchase, 0 otherwise); Age (1 for 20 years and younger, 2 for 21 to 30 years, 3 for 31 to 40 years, 4 for 41 to 50 years, and 5 for 51 and older); Income (1 for $0 to $50K, 2 for $51K to $80K, 3 for $81K to $100K, 4 for $100K+); and PastPurchase (1 for no past purchase, 2 for 1 or 2 past purchases, 3 for 3 to 6 past purchases, 4 for 7 or more past purchases).

Business Goal

- The goal of the model is to predict whether a customer will make a purchase in response to a promotional e-mail.

Data Understanding

EDA

```
## Brief dataset exploration
Online_Retail %>% head()
```

```
# A tibble: 6 × 4
  Purchase   Age Income PastPurchase
     <dbl> <dbl>  <dbl>        <dbl>
1        1     4      3            4
2        1     4      1            1
3        1     2      1            2
4        0     5      4            2
5        1     4      1            3
6        0     4      4            2
```

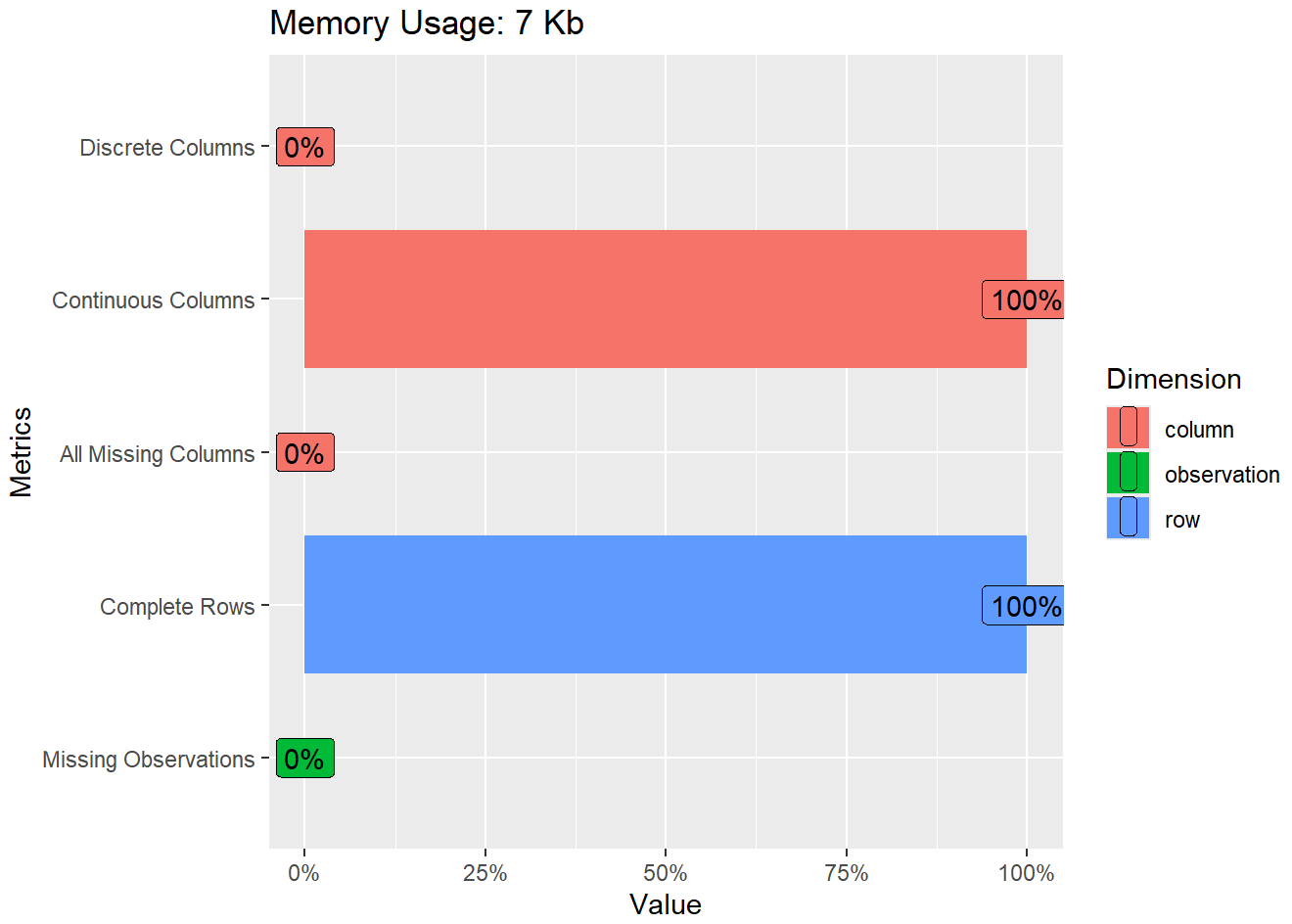
```
Online_Retail %>% tail()
```

```
# A tibble: 6 × 4
  Purchase   Age Income PastPurchase
     <dbl> <dbl>  <dbl>        <dbl>
1        0     2      3            2
2        1     4      4            2
3        1     4      1            3
4        0     4      3            1
5        1     2      2            1
6        1     3      4            3
```

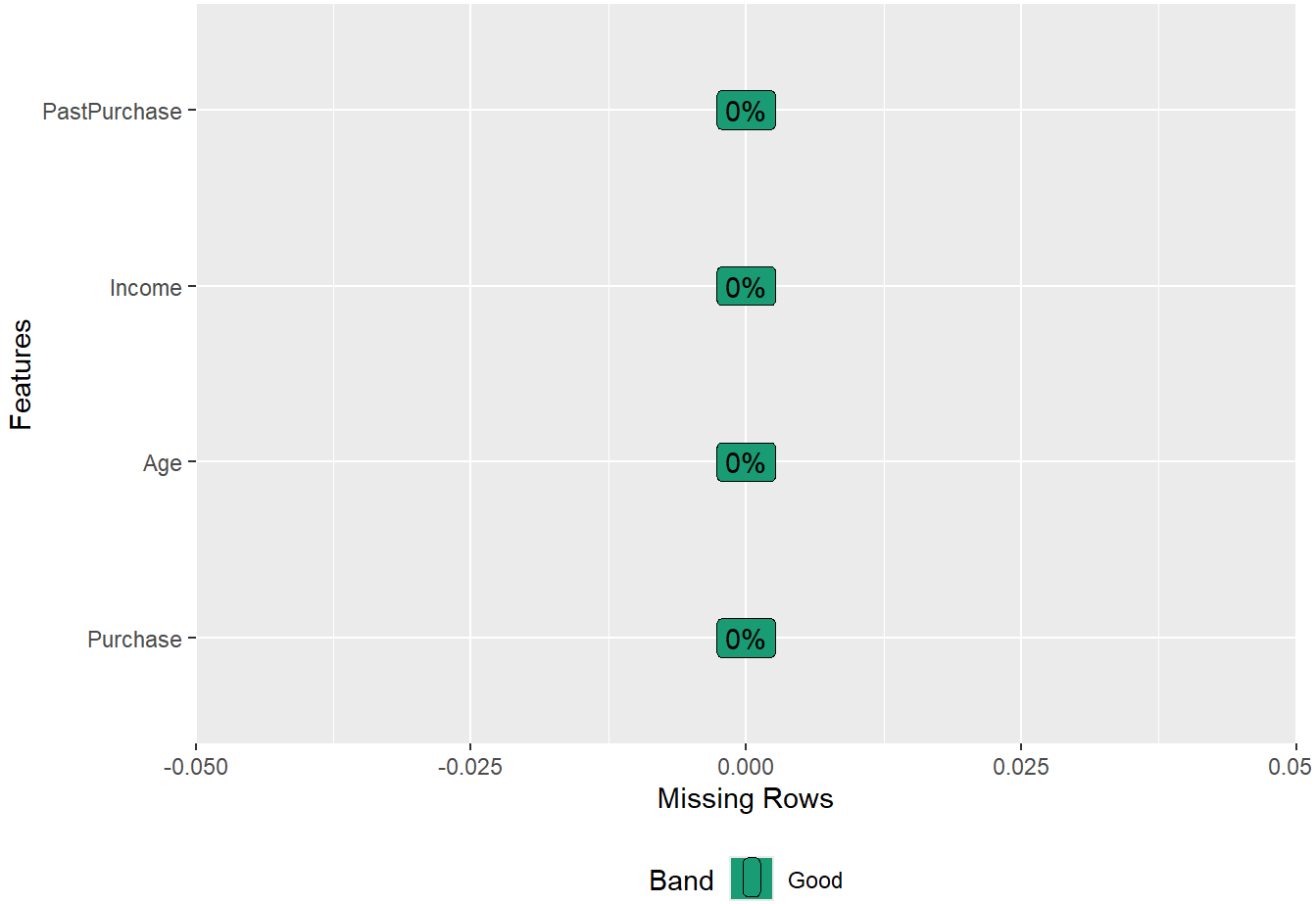
```
Online_Retail %>% str()
```

```
tibble [170 × 4] (S3: tbl_df/tbl/data.frame)
 $ Purchase    : num [1:170] 1 1 1 0 1 0 1 1 1 1 ...
 $ Age         : num [1:170] 4 4 2 5 4 4 4 4 4 4 ...
 $ Income      : num [1:170] 3 1 1 4 1 4 4 1 1 3 ...
 $ PastPurchase: num [1:170] 4 1 2 2 3 2 3 4 2 2 ...
```

```
Online_Retail %>% plot_intro()
```




```
Online_Retail %>% plot_missing()
```



Comments about EDA:

This dataset contains no missing variables. **Important note** this dataset is pre-binned, refer back to business understanding to see what certain bins mean for each variable.

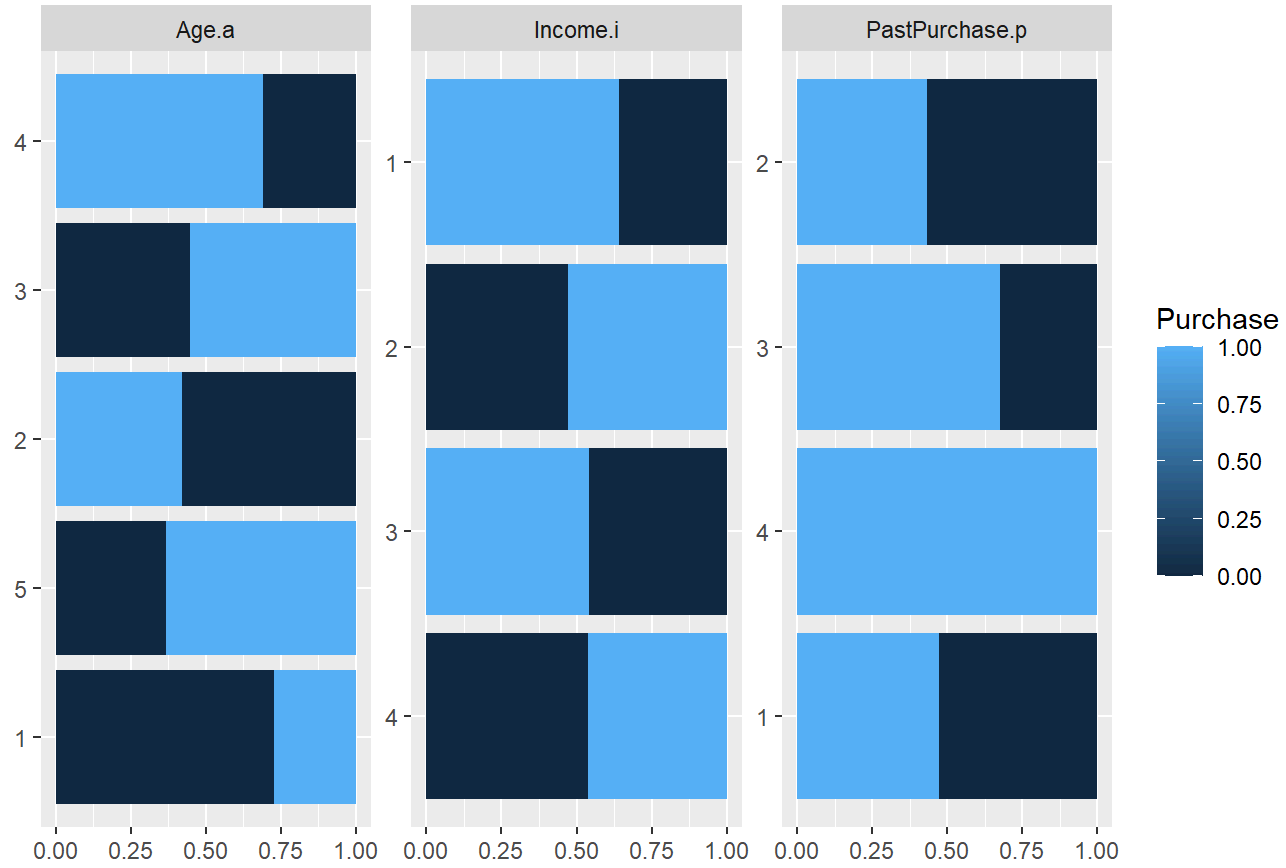
Variable EDA

```
## Variable manipulation in preparation for Exploration and Modeling

## Leave this variable as a factor since it is the dependent variable
Online_Retail$Age.a <- as.character(Online_Retail$Age)
Online_Retail$Income.i <- as.character(Online_Retail$Income)
Online_Retail$PastPurchase.p <- as.character(Online_Retail$PastPurchase)
View(Online_Retail)


Online_Retail %>% str()
```

```
tibble [170 × 7] (S3: tbl_df/tbl/data.frame)
 $ Purchase      : num [1:170] 1 1 1 0 1 0 1 1 1 1 ...
 $ Age           : num [1:170] 4 4 2 5 4 4 4 4 4 4 ...
 $ Income        : num [1:170] 3 1 1 4 1 4 4 1 1 3 ...
```

```
$ PastPurchase  : num [1:170] 4 1 2 2 3 2 3 4 2 2 ...
$ Age.a         : chr [1:170] "4" "4" "2" "5" ...
$ Income.i      : chr [1:170] "3" "1" "1" "4" ...
$ PastPurchase.p: chr [1:170] "4" "1" "2" "2" ...
```

```
## Potential Issue customers with a past purchase amount of four dont have a "0 purchase case"
        which might affect the model.
plot_bar(Online_Retail, by = "Purchase")
```




```
CrossTable(Online_Retail$Purchase, format = "SPSS")
```

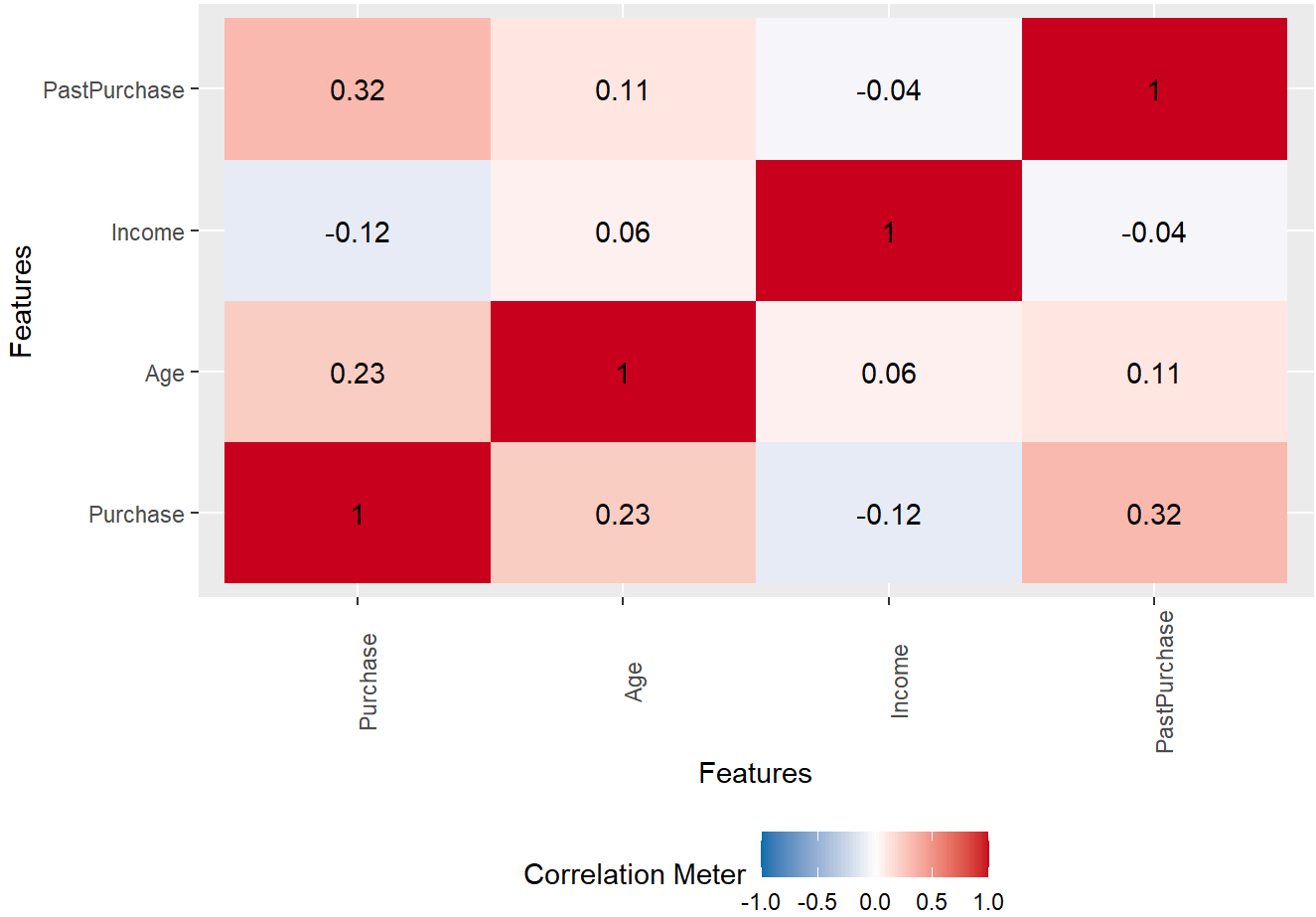
```
   Cell Contents
|-------------------------|
|                   Count |
|             Row Percent |
|-------------------------|

Total Observations in Table:  170

          |        0  |        1  |
          |-----------|-----------|
          |       75  |       95  |
```

```
|   44.118% |   55.882% |
|-----------|-----------|
```

```
##Confirming that a Past purchase amount of four has 0 purchase cases.
Online_Retail %>%
  group_by(PastPurchase) %>%
  count(Purchase)
```

```
# A tibble: 7 × 3
# Groups:   PastPurchase [4]
  PastPurchase Purchase     n
         <dbl>    <dbl> <int>
1            1        0    10
2            1        1     9
3            2        0    50
4            2        1    38
5            3        0    15
6            3        1    31
7            4        1    17
```

```
Online_Retail <- Online_Retail %>%
  dplyr::select(-c(5: 7))

View(Online_Retail)

Online_Retail %>% plot_correlation()
```



Variable EDA Comments:

Frequency Distribution for the dependent variable is relatively balanced with a 45/55% split between 0's and 1's. Past Purchases of four have no examples where a customer did not make a purchase. This can mess up the model so Laplace smoothing will need to be considered when modeling.

Data Preparation

```
## Transform data type for correct type for NB
Online_Retail$Purchase <- as.factor(Online_Retail$Purchase)
Online_Retail$Age <- as.factor(Online_Retail$Age)
Online_Retail$Income <- as.factor(Online_Retail$Income)
Online_Retail$PastPurchase <- as.factor(Online_Retail$PastPurchase)
```

Data Preperation Comments

For Naive Bayes the dependent variable has to be made into a factor for it to run proper classification. The predictor variables were also transformed to factors. Converting these binned variables into factors allows the model to interpret their true meaning as each of their numbers represent a category.

Modeling

```
set.seed(1)
my_index <- createDataPartition(Online_Retail$Purchase, p=0.6, list=FALSE)
trainset1 <- Online_Retail[my_index,]
testset1 <- Online_Retail[-my_index,]
```

Before running the KNN model the data is partitioned into a training dataset and a test dataset. The training dataset will be used to fit the model. The test set is used to evaluate to see how well the model generalizes.

```
cv_tctrl <- trainControl(method = "cv", number=10)
set.seed(1)
nb_fit <- train(Purchase~., data = trainset1, method = "nb", trControl = cv_tctrl)
nb_fit
```

```
Naive Bayes

102 samples
  3 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 91, 91, 92, 91, 93, 93, ...
Resampling results across tuning parameters:

  usekernel  Accuracy   Kappa
  FALSE            NaN         NaN
   TRUE      0.5874747  0.06888492

Tuning parameter 'fL' was held constant at a value of 0
Tuning
 parameter 'adjust' was held constant at a value of 1
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were fL = 0, usekernel = TRUE and adjust
 = 1.
```

```
cv_tctrl <- trainControl(method = "cv", number=10)
set.seed(1)
nb_fit_adjusted <- train(Purchase ~ ., data = trainset1, method = "nb", trControl = cv_tctrl,
          tuneGrid = data.frame(fL = 1, usekernel = TRUE, adjust = 1))
nb_fit_adjusted
```

```
Naive Bayes

102 samples
  3 predictor
```

```
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 91, 91, 92, 91, 93, 93, ...
Resampling results:

  Accuracy   Kappa
  0.5874747  0.06888492

Tuning parameter 'fL' was held constant at a value of 1
Tuning
 parameter 'usekernel' was held constant at a value of TRUE
Tuning
 parameter 'adjust' was held constant at a value of 1
```

Comments over Modeling

Two models were made one without Laplace smoothing and one with Laplace smoothing. When the models were run the Laplace smoothing showed no improvement in accuracy and kappa however the model adjusted to Laplace smoothing was used throughout the rest of the analysis. The Naive Bayes model was then evaluated on the test dataset.

Model Evaluation

```
nb_class_predict <- predict(nb_fit_adjusted, newdata = testset1)
confusionMatrix(nb_class_predict, testset1$Purchase, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0  2  0
         1 28 38

               Accuracy : 0.5882
                 95% CI : (0.4623, 0.7063)
    No Information Rate : 0.5588
    P-Value [Acc > NIR] : 0.3588

                  Kappa : 0.0739

 Mcnemar's Test P-Value : 0.0000003352

            Sensitivity : 1.00000
            Specificity : 0.06667
         Pos Pred Value : 0.57576
```

```
          Neg Pred Value : 1.00000
              Prevalence : 0.55882
          Detection Rate : 0.55882
    Detection Prevalence : 0.97059
       Balanced Accuracy : 0.53333

        'Positive' Class : 1
```

Confusion Matrix Comments

A confusion matrix was made to help show the results of how well the model performed. The model showed an accuracy of 58.8% which is the proportion of all correct predictions. The 95% CI shows that with 95% confidence the accuracy will lie within the range of (0.4623, 0.7063). The No Information Rate shows the model does not perform better than trivial guessing as the accuracy 0.5882 is equal to the NIR value of 0.5588. The P-Value shows the model is not statistically significant 0.3588 > 0.05.

The sensitivity rate of 1.0000 shows the model identifies all positive cases (Purchase = 1). The specificity rate of 0.0667 shows the model almost completely fails to identify negative cases (Purchase = 0). This shows very poor balance and strong bias toward the majority positive predictions.

The Positive Predictive Value shows that of those predicted as positive, only 57.6% were correct, which is weak precision. The Negative Predictive Value shows that of those predicted as negative, 100% were correct, but this is misleading since the model predicted almost no negatives.

A prevalence of 0.5588 shows the proportion of positive cases in the dataset, around 56%. The Detection Rate of 0.5588 shows the proportion of all records that were correctly predicted as positives. Detection Prevalence, 0.9706, shows the records predicted as positive, which is much higher than the actual prevalence. This confirms the model is over-predicting the positive class.

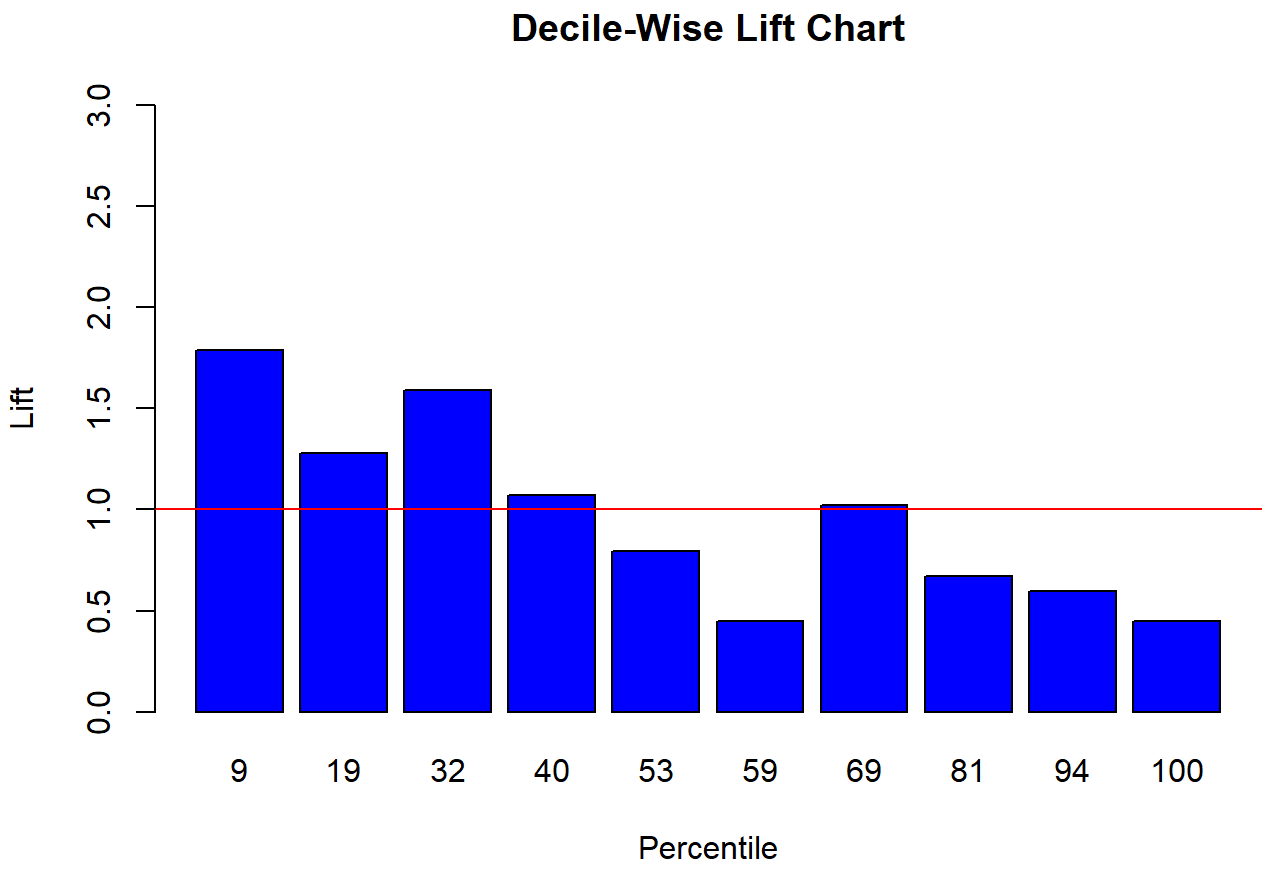
Finally, the Balanced Accuracy of 0.5333 shows the model is only slightly better than random guessing and is not performing well. Overall, this model is not good and suffers from severe imbalance issues, over predicting positives while failing to detect negatives.

```r
## Gains Chart
nb_class_predict <- predict(nb_fit_adjusted, newdata = testset1, type = 'prob')
testset1$Purchase <- as.numeric(as.character(testset1$Purchase))
gains_table <- gains(testset1$Purchase, nb_class_predict[,2])
gains_table
```

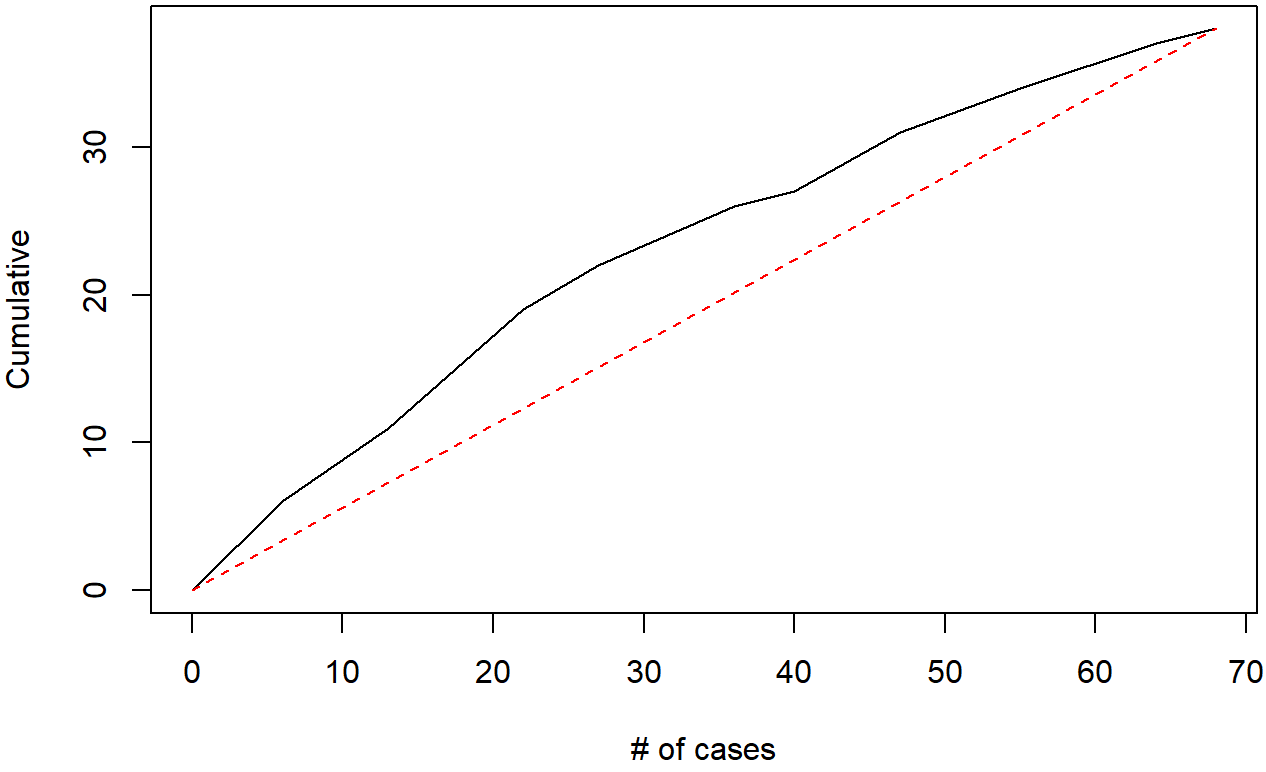
Depth of File	N	Cume N	Mean Resp	Cume Mean Resp	Cume Pct of Total Resp	Lift Index	Cume Lift	Mean Model Score
9	6	6	1.00	1.00	15.8%	179	179	0.97
19	7	13	0.71	0.85	28.9%	128	151	0.94
32	9	22	0.89	0.86	50.0%	159	155	0.92

40	5	27	0.60	0.81	57.9%	107	146	0.90
53	9	36	0.44	0.72	68.4%	80	129	0.88
59	4	40	0.25	0.68	71.1%	45	121	0.86
69	7	47	0.57	0.66	81.6%	102	118	0.81
81	8	55	0.38	0.62	89.5%	67	111	0.76
94	9	64	0.33	0.58	97.4%	60	103	0.64
100	4	68	0.25	0.56	100.0%	45	100	0.53

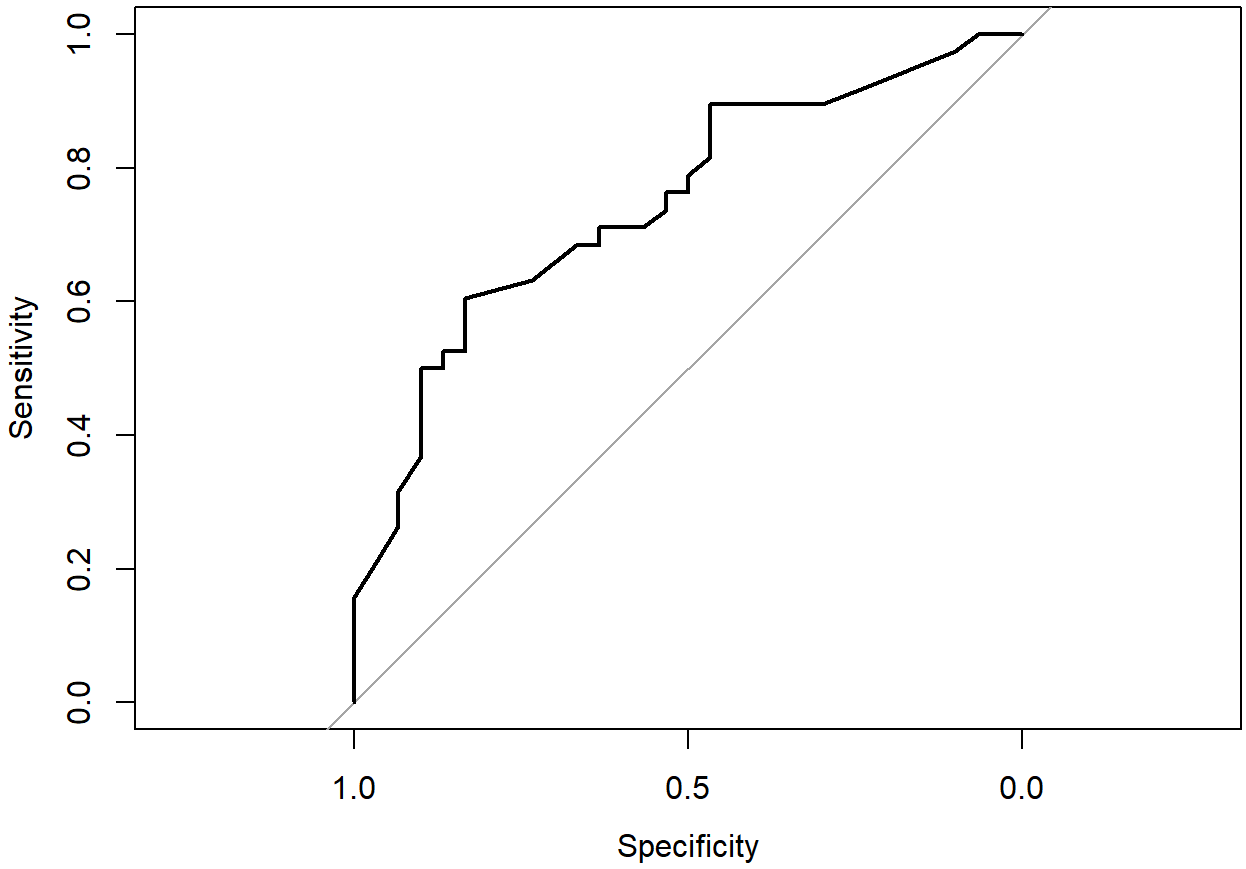
```
## Decile Wise Chart
barplot(gains_table$mean.resp/mean(testset1$Purchase), names.arg=gains_table$depth,
        xlab="Percentile", ylab="Lift", ylim=c(0,3), col="blue", main="Decile-Wise Lift Chart")
abline(h=c(1),col="red")
```




```
## Cumulative Lift Chart
plot(c(0, gains_table$cume.pct.of.total*sum(testset1$Purchase)) ~ c(0, gains_table$cume.obs),
         xlab = '# of cases', ylab = "Cumulative", type = "l")
lines(c(0, sum(testset1$Purchase))~c(0, dim(testset1)[1]), col="red", lty=2)
```




```
## Roc plot and Auc value
roc_object <- roc(testset1$Purchase, nb_class_predict[,2])
plot.roc(roc_object)
```




```
auc(roc_object)
```

```
Area under the curve: 0.7474
```

Evaluation Comments: Charts and Graphs

The model was further evaluated based of charts and graphs. Both the Decile-wise and Cumulative Lift Charts show that the model captures a high volume of positive cases in the top deciles of the data. The first 10–30% of the data contains a strong concentration of positives. The ROC Curve further supports this, as it rises well above the diagonal line, indicating good discriminating power between positive and negative classes. The AUC value is 0.7474 which shows the model is better than random chance at discriminating between positive and negative cases. The Gains Table shows that the top 50% of the dataset captures around 70–75% of the actual positives.

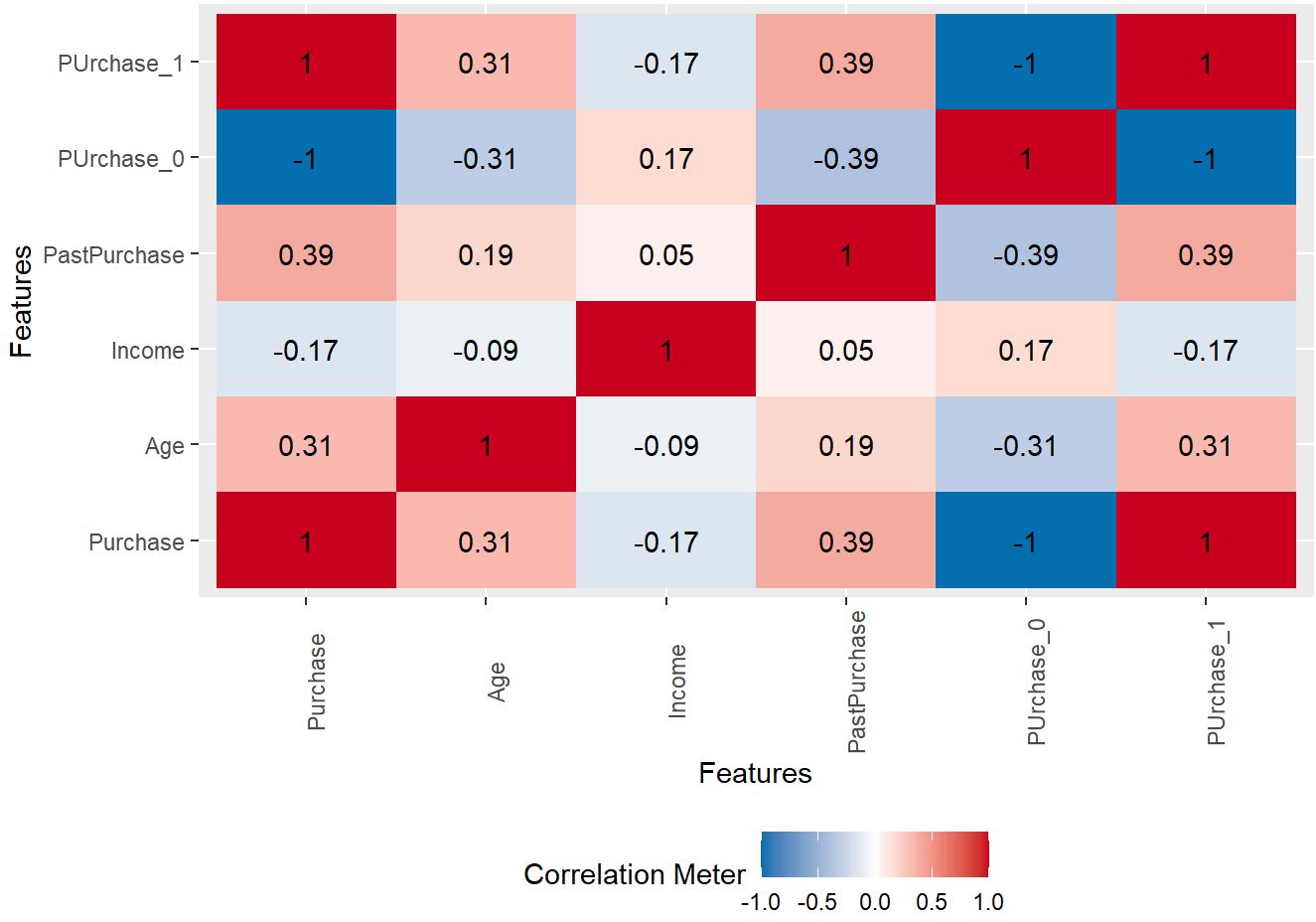
Deployment

```
testset1$PUrchase <- as.factor(testset1$Purchase)
testset1$Age <- as.numeric(testset1$Age)
testset1$PastPurchase <- as.numeric(testset1$PastPurchase)
```

```
testset1$Income <- as.numeric(testset1$Income)

testset1 %>%
  group_by(Purchase) %>%
  summarise(meanage = mean(Age),
            meanpospur = mean(PastPurchase),
            meanincome = mean(Income))
```

```
# A tibble: 2 × 4
  Purchase meanage meanpospur meanincome
     <dbl>   <dbl>      <dbl>      <dbl>
1        0    2.83       2.07       2.37
2        1    3.53       2.71       2
```

```
testset1 %>% plot_correlation()
```



Recommendations:

The model shows that Age and Past Purchases show weak positive correlation to Purchase, meaning that older people and people with higher past purchases tend to make purchases. This model showed bias prediction of classes so model's greatest value would be in segmenting the dataset. The model can help

direct where to segment the dataset so the retailer can select which existing customers emails should be sent to, conserving resources and time.