Part 1

1. ***Consider the following customer: Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Education_3 = 0, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1, and Credit Card = 1. Perform a k-NN classification with all predictors except ID and ZIP code using k = 1. How would this customer be classified?***

After having split the data with the first 3000 observations being the training, normalizing the dataset with min-max normalization and creating the record, the new customer was classified as a customer with personal loan = 0. This means that that the customer would not be approved

2. ***With the same customer above, classify the customer using the best k. Explain intuitively how this customer belongs to the group.***

After having normalized the testing observations, (last 2000), and using a for loop to test for various k values (the number of neighbors we look at) I determined that the optimal value of k was 3. The for loop ran from 1 to 100 and looked at which value had the lowest MSE.

When classifying the new customer with a value of k = 3, the result of the classification did not change the application was rejected; personal loan = 0. This means that when looking at where the new customer is located compared to others most of the 3 closest customers got rejected. When looking at why and quickly analyzing what sort of customers generally get accepted I would guess that augmenting the income as well as seeing higher education (education3 = 1).

3. ***Explain your thought process in selecting the best k for question 2.***

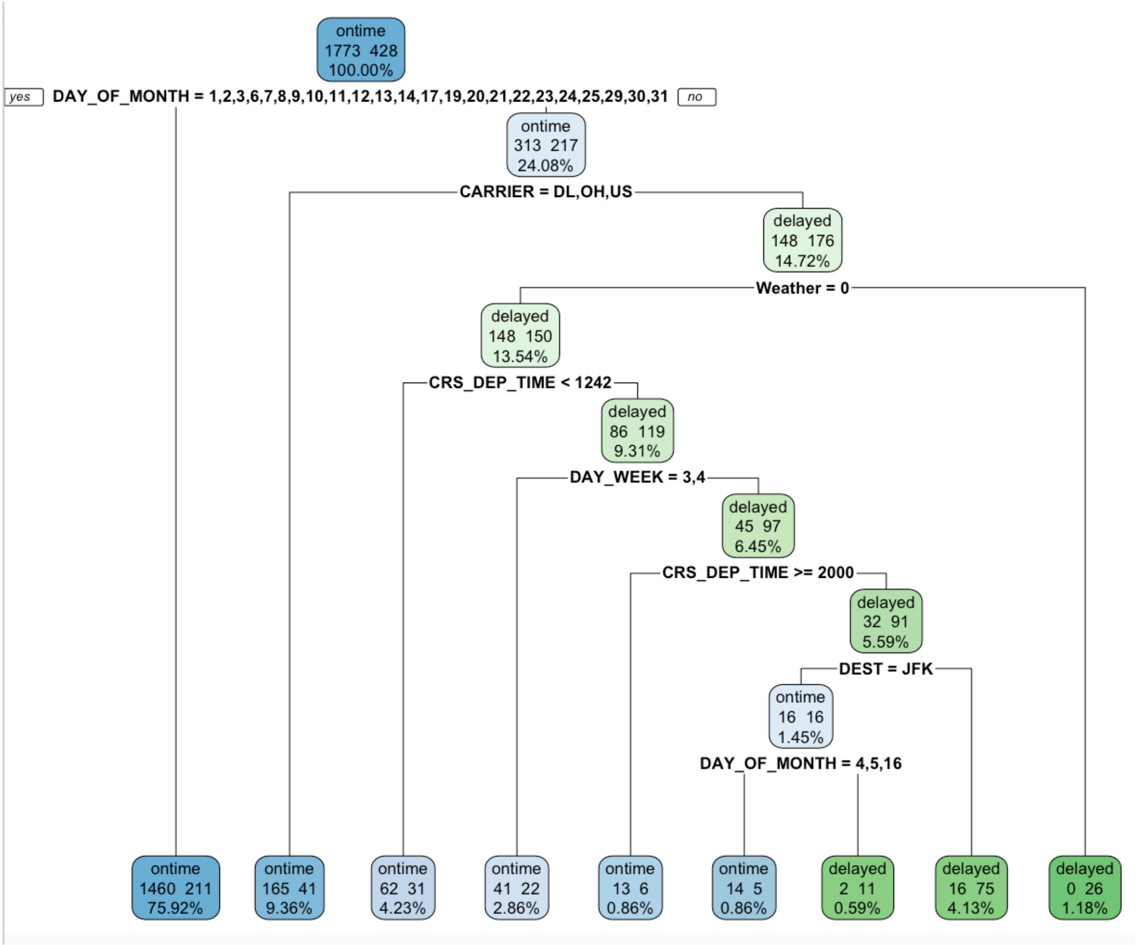
To determine the ideal k value, I started by determining the possible range for k. if k is too small, the model might be overfitting, and outliers might really affect the classification. However, if k is too big then local interesting values might be overlooked,

Following the rule of thumb, k = sqrt(n), I realized that the "ideal" k would be sqrt(5000) or 71 approximately. I therefore decided that I would run the for loop from 1 to 100 to include the 71 and see if really that was the ideal k.

I was originally expecting a value close to 71 but smaller to avoid the problems that arrived with a k too large, but to my surprise the lowest MSE came with a value of k=3.

Part 2

4. ***Construct a decision tree using all relevant predictors (except DEP_TIME). Express the tree as a set of rules.***



This is the tree that is generated when including the relevant variables. The non-relevant variables were chosen to be DEP_TIME as indicated in the question, FL_DATE as it is a variable with too many categories, and this can lead to overfitting, and FL_NUM as it also has too many levels, and is almost an identifier). To generate this tree the relevant variables that qualified were transformed as factor.

Tree Expressed As a Set of Rules:

Rule number: 2 [yval=ontime cover=1671 (76%) prob=0.13]

DAY_OF_MONTH=1,2,3,6,7,8,9,10,11,12,13,14,17,19,20,21,22,23,24,25,29,30,31

Rule number: 6 [yval=ontime cover=206 (9%) prob=0.20]

DAY_OF_MONTH=4,5,15,16,18,26,27,28

CARRIER=DL,OH,US

```
Rule number: 28 [yval=ontime cover=93 (4%) prob=0.33]

 DAY_OF_MONTH=4,5,15,16,18,26,27,28

 CARRIER=CO,DH,MQ,RU,UA

 Weather=0

 CRS_DEP_TIME< 1242

Rule number: 58 [yval=ontime cover=63 (3%) prob=0.35]

 DAY_OF_MONTH=4,5,15,16,18,26,27,28

 CARRIER=CO,DH,MQ,RU,UA

 Weather=0

 CRS_DEP_TIME>=1242

 DAY_WEEK=3,4

Rule number: 118 [yval=ontime cover=19 (1%) prob=0.32]

 DAY_OF_MONTH=4,5,15,16,18,26,27,28

 CARRIER=CO,DH,MQ,RU,UA

 Weather=0

 CRS_DEP_TIME>=1242

 DAY_WEEK=1,2,5,7

 CRS_DEP_TIME>=2000

Rule number: 476 [yval=ontime cover=19 (1%) prob=0.26]

 DAY_OF_MONTH=4,5,15,16,18,26,27,28

 CARRIER=CO, DH,MQ,RU,UA

 Weather=0

 CRS_DEP_TIME>=1242

 DAY_WEEK=1,2,5,7

 CRS_DEP_TIME< 2000
```

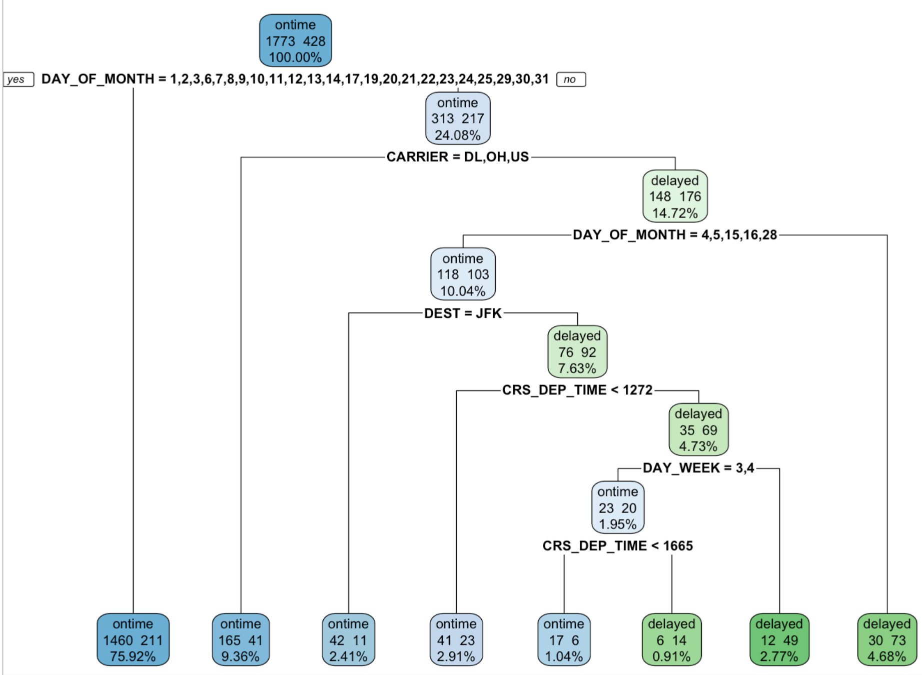
```
  DEST=JFK

  DAY_OF_MONTH=4,5,16

Rule number: 477 [yval=delayed cover=13 (1%) prob=0.85]

  DAY_OF_MONTH=4,5,15,16,18,26,27,28

  CARRIER=CO,DH,MQ,RU,UA

  Weather=0

  CRS_DEP_TIME>=1242

  DAY_WEEK=1,2,5,7

  CRS_DEP_TIME< 2000

  DEST=JFK

  DAY_OF_MONTH=18,26,27

Rule number: 239 [yval=delayed cover=91 (4%) prob=0.82]

  DAY_OF_MONTH=4,5,15,16,18,26,27,28

  CARRIER=CO,DH,MQ,RU,UA

  Weather=0

  CRS_DEP_TIME>=1242

  DAY_WEEK=1,2,5,7

  CRS_DEP_TIME< 2000

  DEST=EWR,LGA

Rule number: 15 [yval=delayed cover=26 (1%) prob=1.00]

  DAY_OF_MONTH=4,5,15,16,18,26,27,28

  CARRIER=CO,DH,MQ,RU,UA

  Weather=1
```

5. ***If you want to fly between these two airports on Monday at 7:00AM, how would you be able to use the tree constructed above? Do you need any additional information?***

To be able to determine whether or not we can classify this flight we must look at the top of the tree. The first node is the day of the month therefore it would be important to know what day of the month that Monday is. If the day is a 4,5,15,16,18,26,27,28 then we keep going in the tree, otherwise we have a 76% that the flight is on time. The next information we need is the carrier. We only keep going CO, DH, MQ, RU, UA. Next, we look at the weather and keep going if weather = 1.

Until now we have only used information that we did not have. Finally, we can use the 7:00 am. 7 am being before 12:42, we stop here and have a 4% chance of being ontime.

6. ***Construct another decision tree that excludes both DEP_TIME and Weather predictor. Compare the tree with the one generated in question 4. Do you notice any significant differences? Is the tree more/less useful? How?***



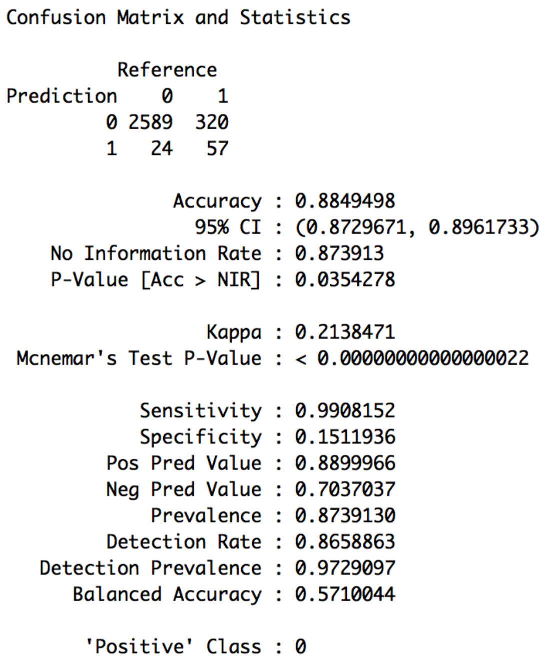
The above tree is relatively similar to the first tree. First and second nodes are identical. The first change arrives in the third node where it used to include the weather variable. After that the nodes vary and go through day of month again, destination, CRS_DEP_TIME the day of the week and then CRS_DEP_TIME again.

This tree seems to be more useful as removing the weather variable has made the tree simpler and thus easier to use. It has therefore made it simpler, but probably not less effective as in the original tree the weather was relatively insignificant as the probability of it being 0 was slim.

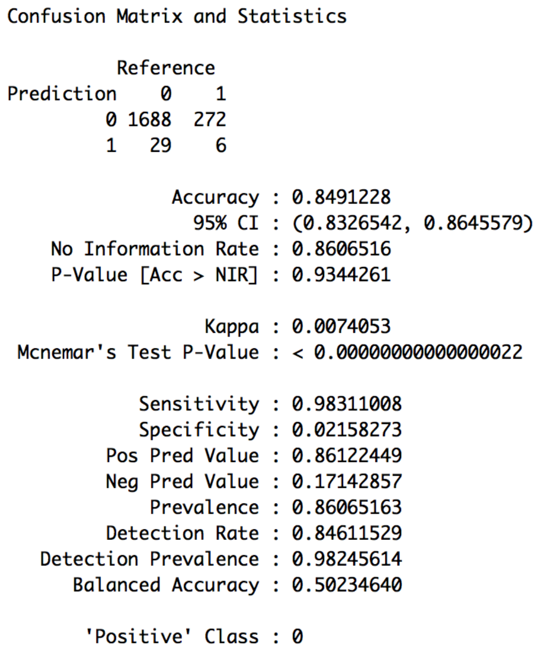
Part 3

7. ***Run a neural net model on these data, using a single hidden layer with 5 nodes. Can you interpret the business implications of the results?***

Training Dataset Matrix **Testing Dataset Matrix**



```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2589  320
         1   24   57

               Accuracy : 0.8849498
                 95% CI : (0.8729671, 0.8961733)
    No Information Rate : 0.873913
    P-Value [Acc > NIR] : 0.0354278

                  Kappa : 0.2138471
 Mcnemar's Test P-Value : < 0.00000000000000022

            Sensitivity : 0.9908152
            Specificity : 0.1511936
         Pos Pred Value : 0.8899966
         Neg Pred Value : 0.7037037
             Prevalence : 0.8739130
         Detection Rate : 0.8658863
   Detection Prevalence : 0.9729097
      Balanced Accuracy : 0.5710044

       'Positive' Class : 0
```



```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1688  272
         1   29    6

               Accuracy : 0.8491228
                 95% CI : (0.8326542, 0.8645579)
    No Information Rate : 0.8606516
    P-Value [Acc > NIR] : 0.9344261

                  Kappa : 0.0074053
 Mcnemar's Test P-Value : < 0.00000000000000022

            Sensitivity : 0.98311008
            Specificity : 0.02158273
         Pos Pred Value : 0.86122449
         Neg Pred Value : 0.17142857
             Prevalence : 0.86065163
         Detection Rate : 0.84611529
   Detection Prevalence : 0.98245614
      Balanced Accuracy : 0.50234640

       'Positive' Class : 0
```

From the two confidence matrices above, one can see that the accuracy for both models are extremely high: 88% for the training and 84% for the testing. It makes sense that the testing is smaller than the testing as it is completely new data. Furthermore, the fact that the testing data has 84% accuracy shows the model does not face the problem of overfitting.

The positive class being 0, means that the model is really good at identifying which customers will not buy phone. As a business, this is very important and useful as one can target the customers that are not classified as 0 and lower costs of targeting customers that would be useless to target.

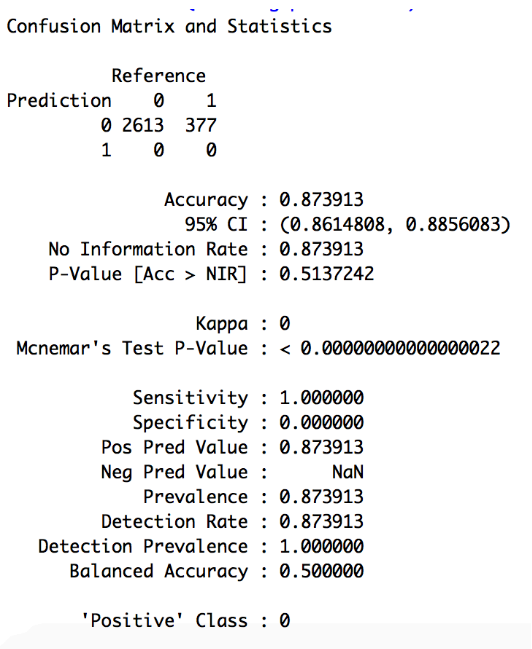
8. ***Comment on the difference between results based on training and validation data***

When looking at the two images in question 7, one can see 2 main differences: the p-value and the kappa values. In the training model, the p value is very low. When analyzing the p-value it is important to make sure that it is lower than the no information rate. While it is the case in the training, it is not the case in the testing. The p-value is extremely high and even higher than the no information rate.

The second big difference is the Kappa value. While in the training model the kappa value is 21%, in the testing the value is of 0.0074%. As this value represents the probability of an observation to be classified by chance and not by the model, it is important for it to be as low as possible,

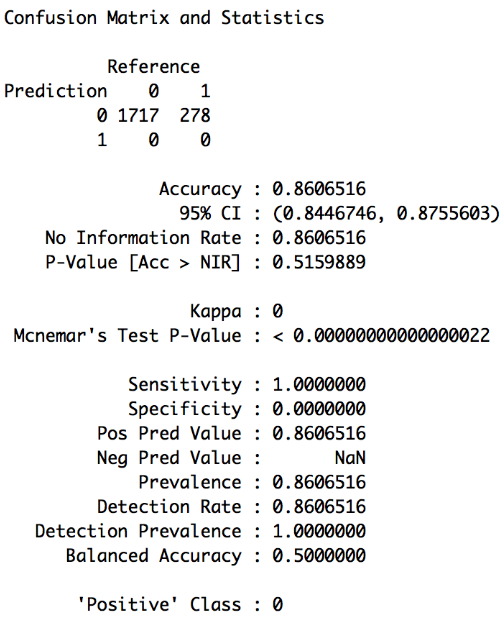
9. *Run another neural net model on the data where the number of hidden nodes is 1. Comment on the difference between this model and the model you run for question 7*

Training Dataset Matrix



```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2613  377
         1    0    0

               Accuracy : 0.873913
                 95% CI : (0.8614808, 0.8856083)
    No Information Rate : 0.873913
    P-Value [Acc > NIR] : 0.5137242

                  Kappa : 0
 Mcnemar's Test P-Value : < 0.00000000000000022

            Sensitivity : 1.000000
            Specificity : 0.000000
         Pos Pred Value : 0.873913
         Neg Pred Value :      NaN
             Prevalence : 0.873913
         Detection Rate : 0.873913
   Detection Prevalence : 1.000000
      Balanced Accuracy : 0.500000

       'Positive' Class : 0
```

Testing Dataset Matrix



```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1717  278
         1    0    0

               Accuracy : 0.8606516
                 95% CI : (0.8446746, 0.8755603)
    No Information Rate : 0.8606516
    P-Value [Acc > NIR] : 0.5159889

                  Kappa : 0
 Mcnemar's Test P-Value : < 0.00000000000000022

            Sensitivity : 1.0000000
            Specificity : 0.0000000
         Pos Pred Value : 0.8606516
         Neg Pred Value :       NaN
             Prevalence : 0.8606516
         Detection Rate : 0.8606516
   Detection Prevalence : 1.0000000
      Balanced Accuracy : 0.5000000

       'Positive' Class : 0
```

As one can see when changing the number of hidden layers, the accuracy is still high and the positive class is still 0. In fact, with one hidden layer, this model's outcome is very weird as it classifies everything as a 0.

However, one can see two differences: first the p-values of both models are smaller than the no information rate and the kappa value is zero for both models showing that no observation should be classified by luck.

Using this model compared to the one in question 7 is therefore not as good as it classifies everything as a 0 and the accuracy does not go up significantly and neither does the kappa in the testing set.

10. *If you are hired as a consultant for the phone company, what additional information would you recommend collecting to increase the performance of the classification model?*

If I were a consultant for the company, I would collect various other pieces of information. The main flaw I see in the collected information is that there is no information about the customers themselves. Variables such as past purchases, age, location of the bought ticket etc could be very useful. Income could be useful as well, but there is no real way of determining the income of the users.

While the current variables seem to be doing well it would be interesting to know and analyze the information on the customers and maybe even cluster them and target groups of customers differently.