Assignment 7: Data Analytics

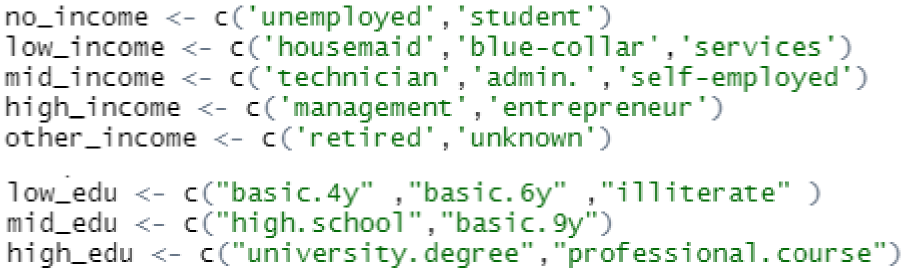
Dataset 1: Bank and Marketing:

http://archive.ics.uci.edu/ml/datasets/Bank+Marketing

The dataset contains a total of 23 feature columns and one label column, indicating whether a bank client has subscribed to the term deposit program at the bank. The goal for this project is to extract relevant features and create predictive models to give suggestions on how to increase the program subscription rate.

EDA and feature engineering:

First thing I do is to verify that there is no NA value within the dataset. Then, I print out the number of levels of each categorical variables, and realize that the “job” and “education” column, with 12 and 8 levels, may cause rank deficient or convergence issues in later modeling with their outstanding number of levels. So, I decide to aggregate these two columns by assorting them into fewer buckets. Specifical details are shown in the code snippets below.



```
no_income <- c('unemployed','student')
low_income <- c('housemaid','blue-collar','services')
mid_income <- c('technician','admin.','self-employed')
high_income <- c('management','entrepreneur')
other_income <- c('retired','unknown')

low_edu <- c("basic.4y"  ,"basic.6y"  ,"illiterate" )
mid_edu <- c("high.school","basic.9y")
high_edu <- c("university.degree","professional.course")
```

Figure 1.Income and Education Bracket

Next, I observe the distributions of the target label and several features as shown in the figures below. The highly uneven distribution of labels in “y” with a yes: no ratio of 0.113 needs to be noted in later model interpretation. Other than that, we can see that most of the levels in the factor columns have a decent number of samples.

Column distributions:

Target label “y”, marital Status, education, income

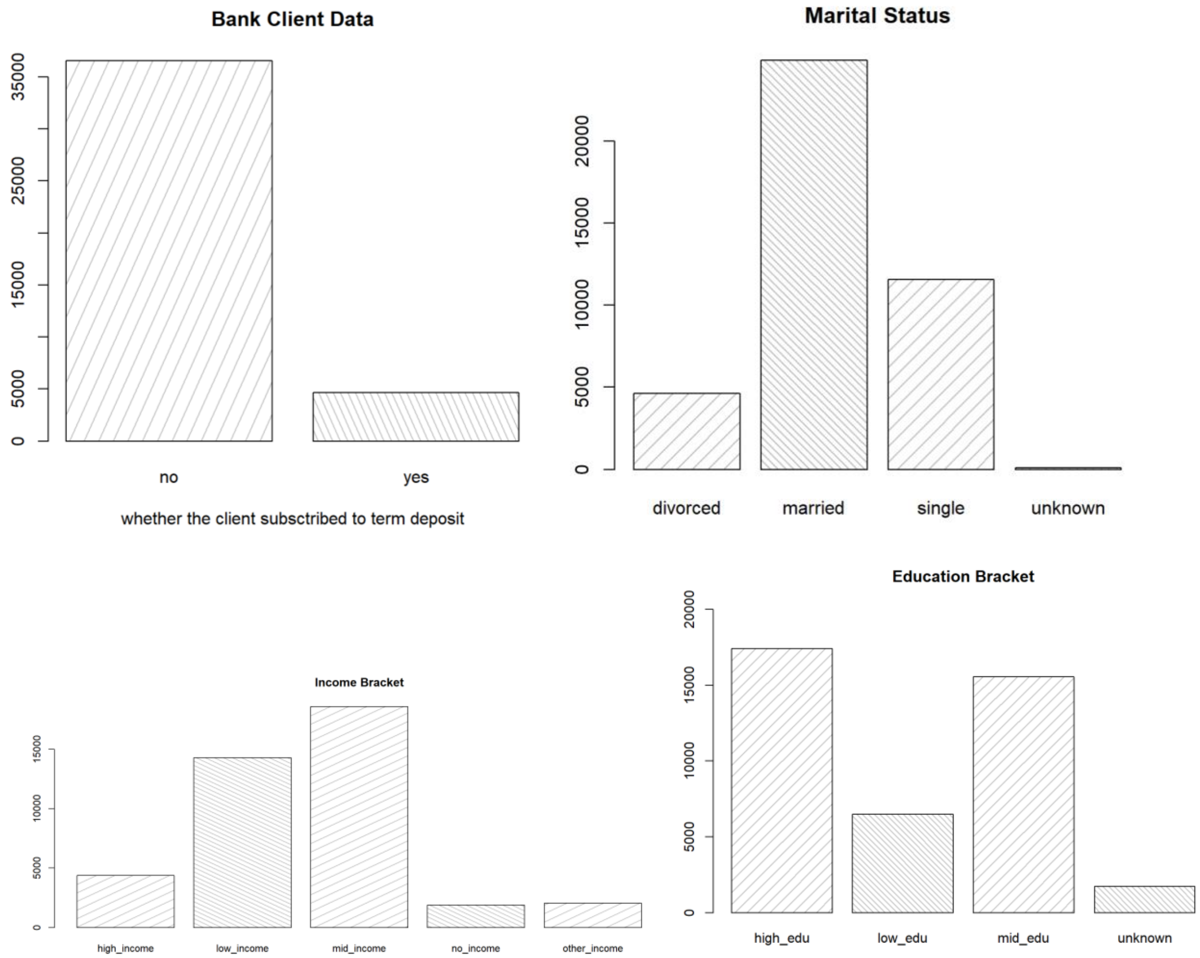
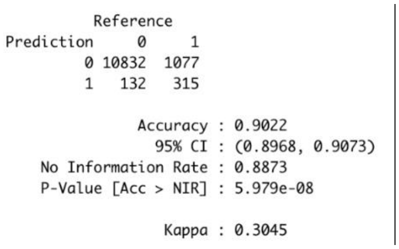


Figure 2.Feature Distribution

Modeling:

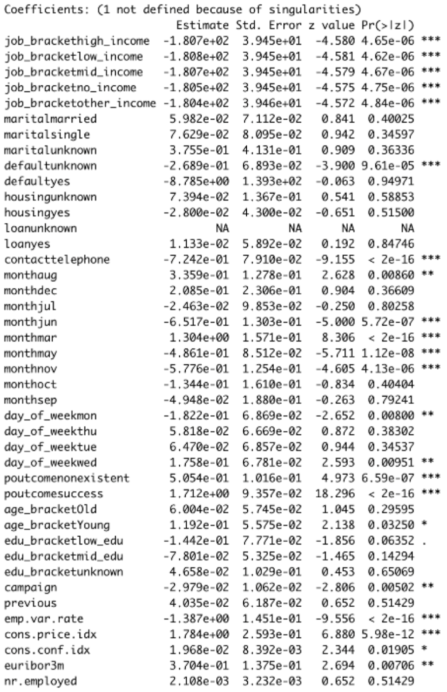
Logistic Regression:

Logistic regression is first to be analyzed since it could give us a sense of what features are significantly correlated with the target and help me to narrow down feature selections in the later models. For the actual algorithm, I use standard glm() function, including all of the feature columns except “duration”, which is mentioned not to be used for more realistic result in the dataset documentation. Also, interpret is excluded from the model to increase features’ significances. The model yields a pseudo-R-squared of 0.6, and a testing accuracy of 0.902. The confusion matrix is shown as below



```
          Reference
Prediction     0     1
         0 10832  1077
         1   132   315

               Accuracy : 0.9022
                 95% CI : (0.8968, 0.9073)
    No Information Rate : 0.8873
    P-Value [Acc > NIR] : 5.979e-08

                  Kappa : 0.3045
```

Figure 3. Confusion Matrix of Logit

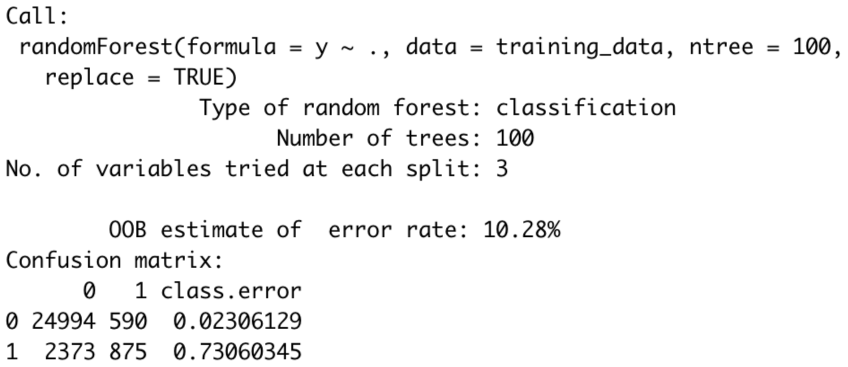


```
Coefficients: (1 not defined because of singularities)
                         Estimate Std. Error z value Pr(>|z|)    
job_brackethigh_income  -1.807e+02  3.945e+01  -4.580 4.65e-06 ***
job_bracketlow_income   -1.808e+02  3.945e+01  -4.581 4.62e-06 ***
job_bracketmid_income   -1.807e+02  3.945e+01  -4.579 4.67e-06 ***
job_bracketno_income    -1.805e+02  3.945e+01  -4.575 4.75e-06 ***
job_bracketother_income -1.804e+02  3.946e+01  -4.572 4.84e-06 ***
maritalmarried           5.982e-02  7.112e-02   0.841  0.40025    
maritalsingle            7.629e-02  8.095e-02   0.942  0.34597    
maritalunknown           3.755e-01  4.131e-01   0.909  0.36336    
defaultunknown          -2.689e-01  6.893e-02  -3.900 9.61e-05 ***
defaultyes              -8.785e+00  1.393e+02  -0.063  0.94971    
housingunknown           7.394e-02  1.367e-01   0.541  0.58853    
housingyes              -2.800e-02  4.300e-02  -0.651  0.51500    
loanunknown                     NA         NA      NA       NA    
loanyes                  1.133e-02  5.892e-02   0.192  0.84746    
contacttelephone        -7.242e-01  7.910e-02  -9.155  < 2e-16 ***
monthaug                 3.359e-01  1.278e-01   2.628  0.00860 ** 
monthdec                 2.085e-01  2.306e-01   0.904  0.36609    
monthjul                -2.463e-02  9.853e-02  -0.250  0.80258    
monthjun                -6.517e-01  1.303e-01  -5.000 5.72e-07 ***
monthmar                 1.304e+00  1.571e-01   8.306  < 2e-16 ***
monthmay                -4.861e-01  8.512e-02  -5.711 1.12e-08 ***
monthnov                -5.776e-01  1.254e-01  -4.605 4.13e-06 ***
monthoct                -1.344e-01  1.610e-01  -0.834  0.40404    
monthsep                -4.948e-02  1.880e-01  -0.263  0.79241    
day_of_weekmon          -1.822e-01  6.869e-02  -2.652  0.00800 ** 
day_of_weekthu           5.818e-02  6.669e-02   0.872  0.38302    
day_of_weektue           6.470e-02  6.857e-02   0.944  0.34537    
day_of_weekwed           1.758e-01  6.781e-02   2.593  0.00951 ** 
poutcomenonexistent      5.054e-01  1.016e-01   4.973 6.59e-07 ***
poutcomesuccess          1.712e+00  9.357e-02  18.296  < 2e-16 ***
age_bracketOld           6.004e-02  5.745e-02   1.045  0.29595    
age_bracketYoung         1.192e-01  5.575e-02   2.138  0.03250 *  
edu_bracketlow_edu      -1.442e-01  7.771e-02  -1.856  0.06352 .  
edu_bracketmid_edu      -7.801e-02  5.325e-02  -1.465  0.14294    
edu_bracketunknown       4.658e-02  1.029e-01   0.453  0.65069    
campaign                -2.979e-02  1.062e-02  -2.806  0.00502 ** 
previous                 4.035e-02  6.187e-02   0.652  0.51429    
emp.var.rate            -1.387e+00  1.451e-01  -9.556  < 2e-16 ***
cons.price.idx           1.784e+00  2.593e-01   6.880 5.98e-12 ***
cons.conf.idx            1.968e-02  8.392e-03   2.344  0.01905 *  
euribor3m                3.704e-01  1.375e-01   2.694  0.00706 ** 
nr.employed              2.108e-03  3.232e-03   0.652  0.51429    
```

From the coefficient reports on the left, we are able to determine that features “nr.employed”, “previous”, “marital,education”, “loan”, “housing” are not significantly correlated enough to explain variance in the target. So, I remove these variables from my feature list and move on to the next model.

Random Forest:

For random forest algorithm, I include 12 feature columns which were shown significantly correlated from previous regression. I choose 100 trees to be included in the forest and a sampling with replacement method to balance between model efficiency and train time. The confusion matrix is shown as below.



```
Call:
 randomForest(formula = y ~ ., data = training_data, ntree = 100,
   replace = TRUE)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 3

        OOB estimate of  error rate: 10.28%
Confusion matrix:
      0   1 class.error
0 24994 590  0.02306129
1  2373 875  0.73060345
```

Figure 4. Confusion Matrix of Random Forest

Also, I generated feature importance to show the relative importance of splits in the forest. As shown in the histogram below, we can see that the forest considers feature “euribor3m”, the 3-month Euribor rate, to be the most important in reducing entropy, following by “poutcome”, the outcome of previous marketing campaign. In overall, the model yields a testing accuracy of 0.899.

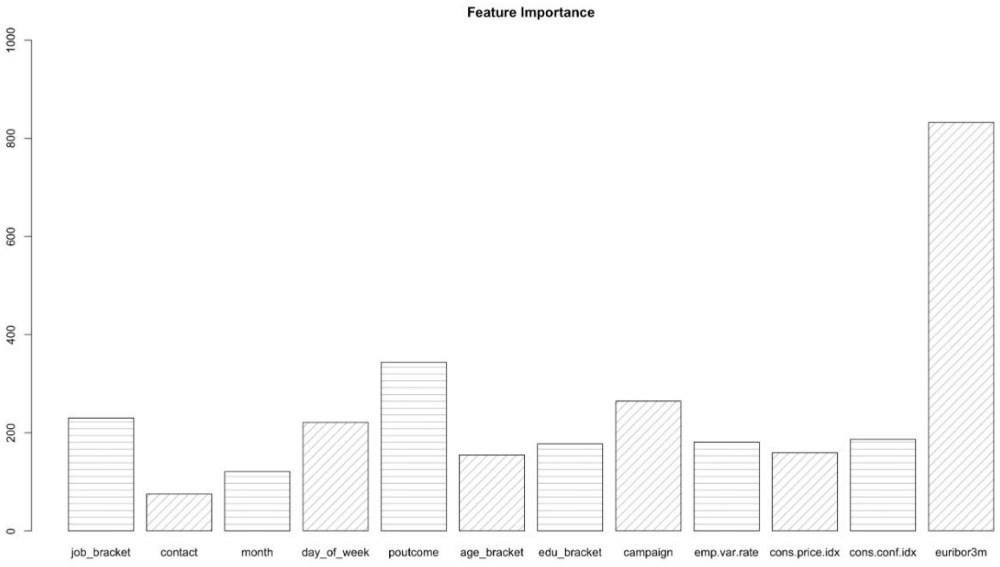


Figure 5. Feature Importance of RF

In order to further visualize feature importance, I also create a decision tree to show how the split are chosen. We can see that indeed the “euribor3m” and “poutcome” create the first two split, meaning that they reduce entropy the most among all features.

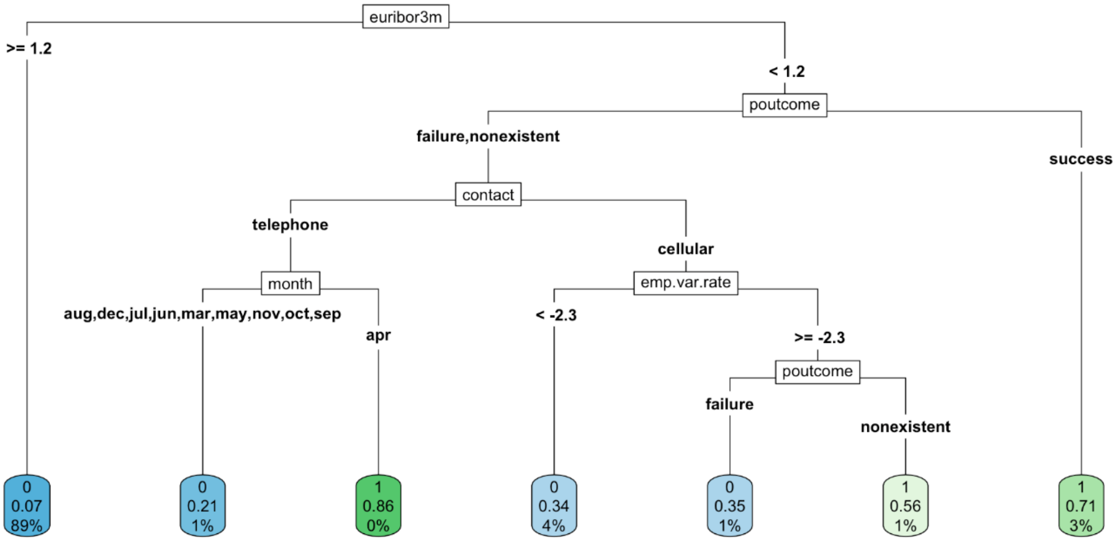


Figure 6. Sample Tree

KNN:

Next, I choose the KNN classifier. In order to select the optimal k value, I use a grid search method with 3-fold cross validation to try different k values, starting from k = 1 to k = 19. Then I plot out the ROC curve along with elbow method to determine the optimal k to be 15. For the actual k = model, I use 10-fold cross validation for more robust accuracy measures. For the final result, the in-sample accuracy is 0.894 and the out-of-sample accuracy is 0.896, indicating no over/under fitting.

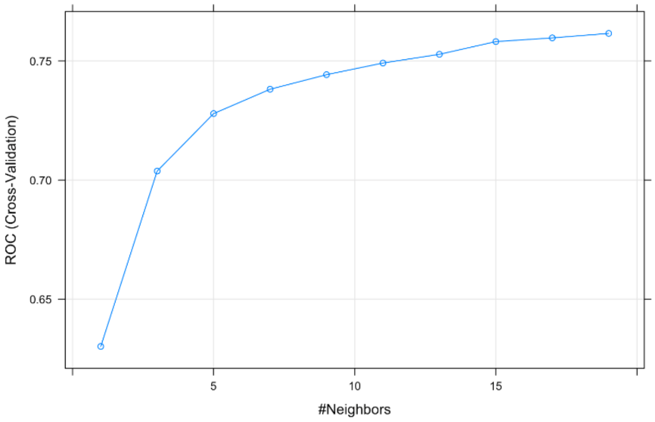
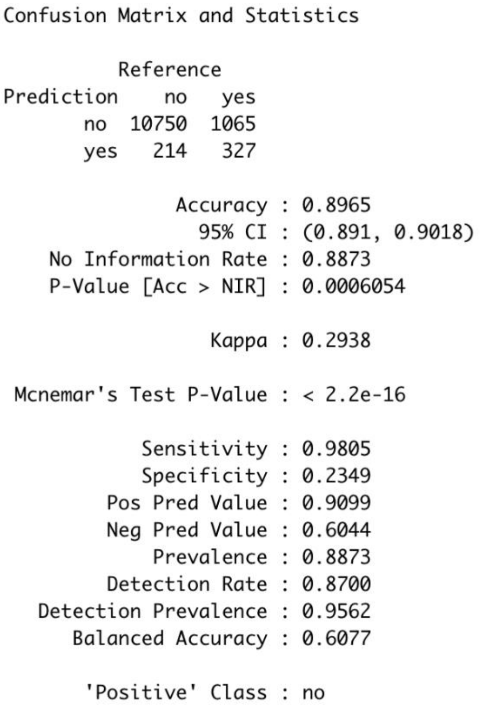


Figure 7. ROC



```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  10750  1065
       yes   214   327

               Accuracy : 0.8965
                 95% CI : (0.891, 0.9018)
    No Information Rate : 0.8873
    P-Value [Acc > NIR] : 0.0006054

                  Kappa : 0.2938

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9805
            Specificity : 0.2349
         Pos Pred Value : 0.9099
         Neg Pred Value : 0.6044
             Prevalence : 0.8873
         Detection Rate : 0.8700
   Detection Prevalence : 0.9562
      Balanced Accuracy : 0.6077

       'Positive' Class : no
```

Figure 8. Confusion Matrix of KNN

Conclusion & Discussion

Considering 87% of target label in original dataset is 0, all of the three models make a slight improvement. Among the three models, Logistic regression gives the highest accuracy of 90.2%. However, due to rank deficient concerns, meaning either many data points in the training sets are duplicate or some features are colinear to each other, we should not solely rely on logistic regression because the model simply does not have enough information to make statistically reliable predictions.

On the other hand, random forest algorithm with 89.9% of accuracy provides some interesting insights.

Random forest suggests that “euribor3m” has the greatest effect on client decision about whether to subscribe to the service. Combining with logistic regression which indicates a positive relationship between “euribor3m” and target label, we can suggest the bank to invest more campaign during the period when Euribor index is around but smaller than 1.2. Also, the forest shows that most of the clients who were converted successfully in the previous campaign are more susceptible to future campaigns. A sensitivity of 0.979 from the random forest suggests that if the bank follows the branch and split criterion, and target heavily to the clients mentioned above, 97.9% of the time the client will decide to subscribe to their service.

KNN ranks last in accuracy among the three in this project. But I believe that this is due to not selecting high enough k number. The original data set has around 41,000 data rows. So, the optimal k value in theory should be around sqrt (41,000) = 640. But it would take too long to converge the model with the limiting resource of a laptop and the inefficiency in memory allocation of Rstudio. Also, KNN is a distance-based algorithm, meaning that it would be hard to interpret feature importance. If the goal is to build a model with the highest accuracy, I would suggest to run KNN with a more powerful machine under a better R framework to speed up the process. However, in a business standpoint, the interpretability of random forest will give it unique upper hand against any distance-based algorithm.

Dataset 2: Cervical Cancer Behavior Risk Data Set

This dataset has 1 target label column indicating whether the respondent has Cervical Cancer, and 18 feature columns including the respondent's health related behavior and hygiene level. The goal is to build models to find underlying relationship between humans' behaviors and Cervical Cancer, hoping to provide suggestions to avoid getting the cancer.

<u>EDA and feature engineering:</u>

First thing I do is to verify that there is no NA value within the dataset. Then, I plot out some of the distributions to examine skewness. And I quickly realize that some of feature columns are poorly variated, meaning that they may bias the classifiers or simply provide no significant insights. So, in this problem, selecting the correct features to include is essentially important to improve models' generalizability.

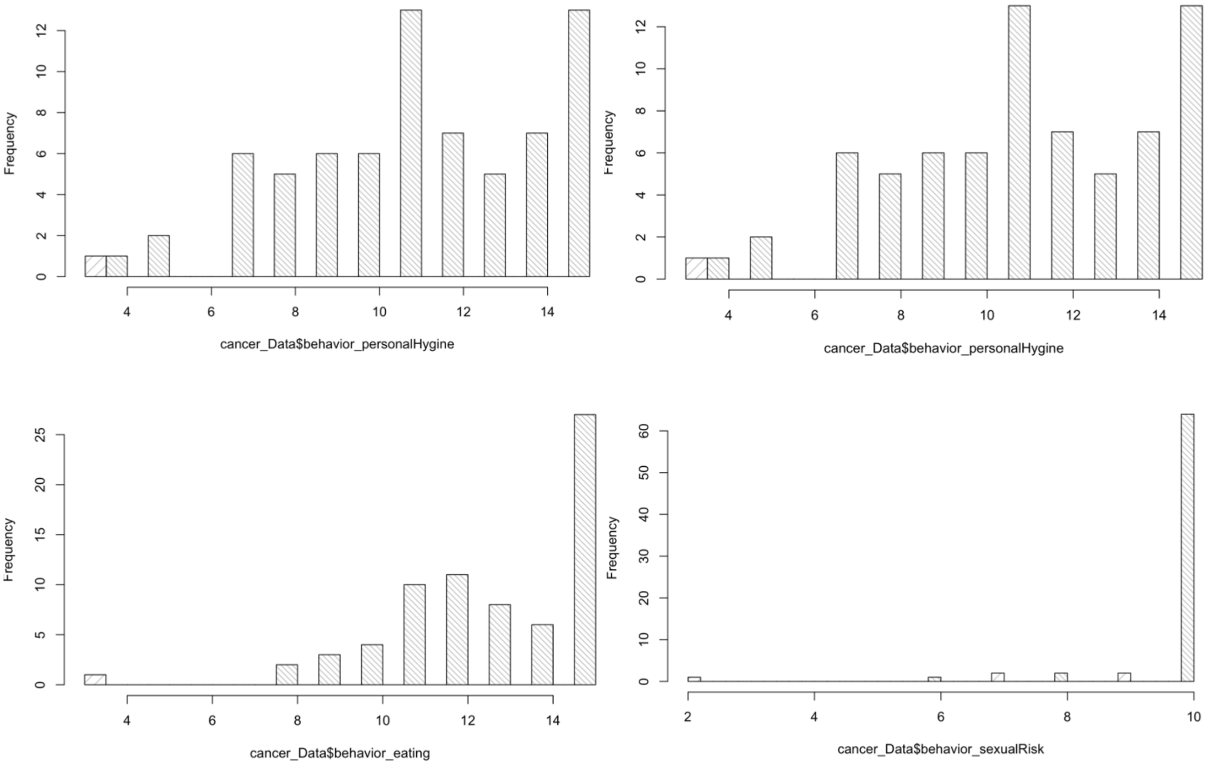


Figure 9. Distribution of Features- Cervical Cancer

From the heatmap, we can clearly observe that only a few features have correlations with the y label. However, there exists clear squares in the heatmap, meaning some of the features have internal linear correlation. We need to be careful about including all of them in the model since they may introduce collinearity.

So, I generate the correlation list of label y, sort it, and save it for feature selection purpose.

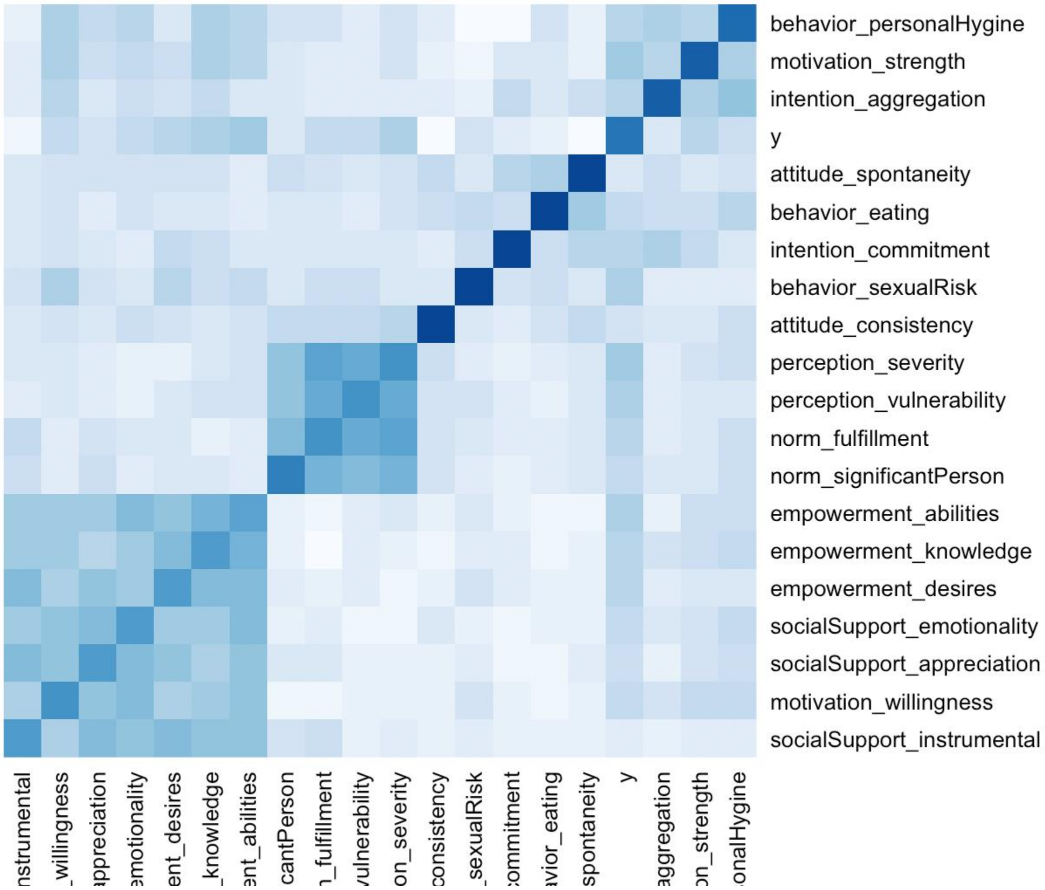


Figure 10. Heatmap of Features

<u>Modeling:</u>

Logistic Regression:

Logistic regression is first to be analyzed. When including all the features in the model, the regression will fail to converge. So, I use a for loop to iteratively increase the number of features to be included in the regression, trying to find the maximum number of features that allows convergence without error. The result is shown in the image below.

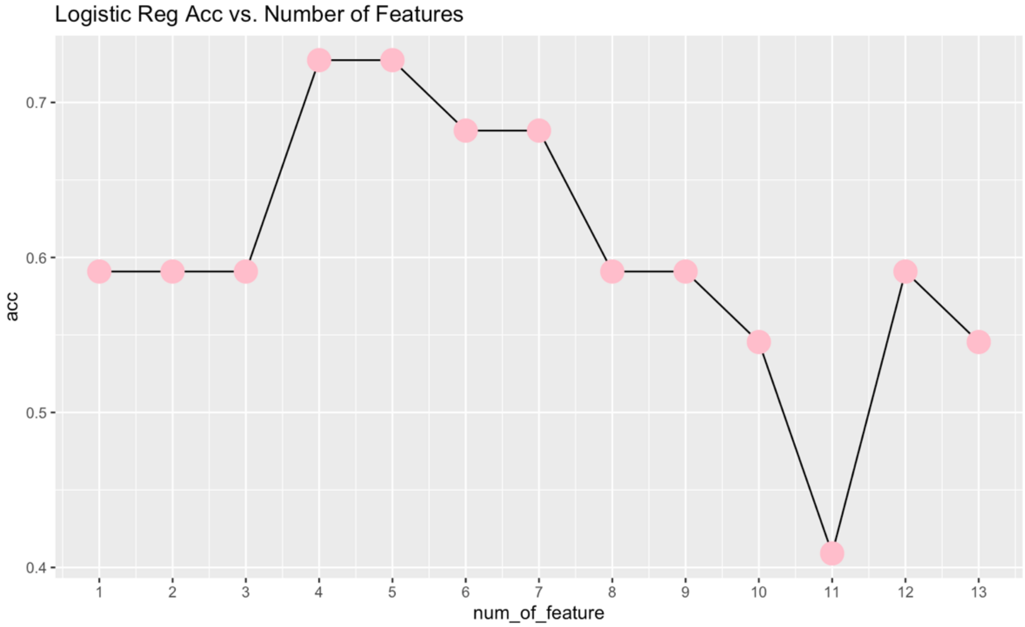
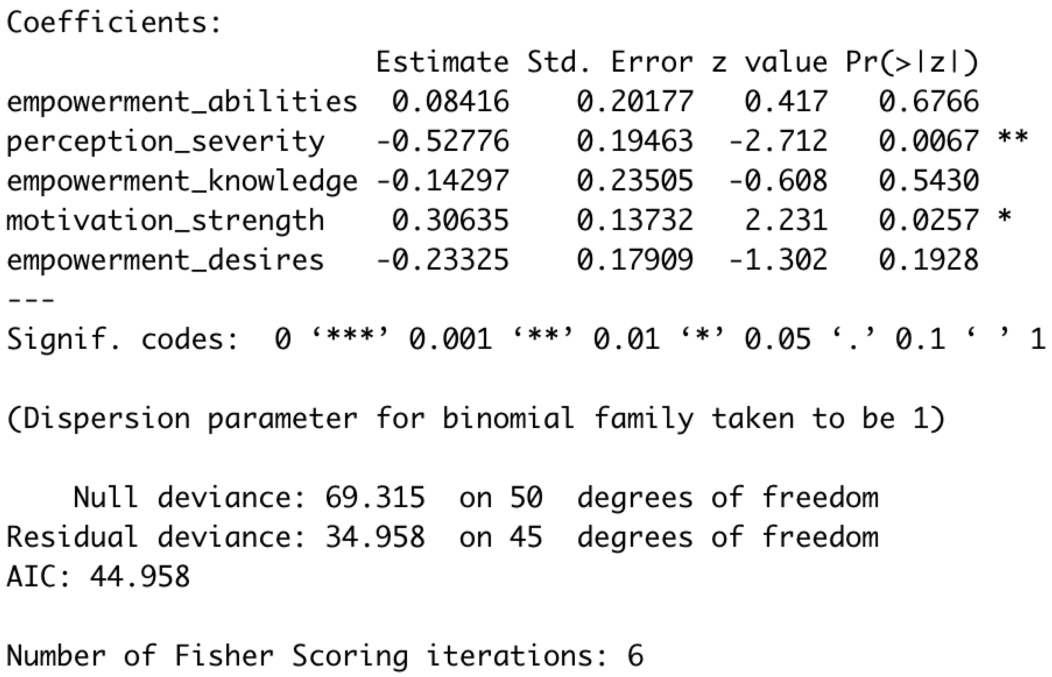


Figure 11. Acc vs. Number of Features

We can see that when including the 4 or 5 most correlated features into the model, the regression yields the highest accuracy of 0.727. After including more than 13 features, the model starts failing to converge. In my analysis, I choose 5 to be the optimal number of features to include because theoretically a logistic model with more features has higher pseudo r-squared, incorporating more variances of the training data. The summary of the actual model is shown as below, indicating that "perception_severity" is significantly negatively correlated, and "motivation_strength" is significantly positively correlated to the cancer, with a testing accuracy of 0.545.

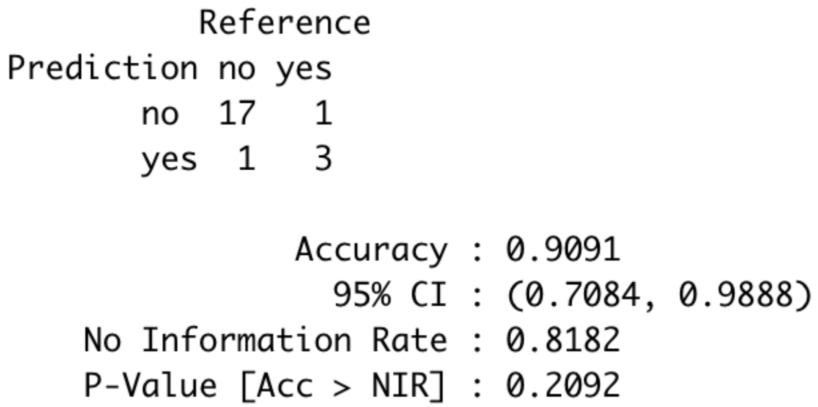


```
Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
empowerment_abilities  0.08416    0.20177   0.417   0.6766
perception_severity   -0.52776    0.19463  -2.712   0.0067 **
empowerment_knowledge -0.14297    0.23505  -0.608   0.5430
motivation_strength    0.30635    0.13732   2.231   0.0257 *
empowerment_desires   -0.23325    0.17909  -1.302   0.1928
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 69.315  on 50  degrees of freedom
Residual deviance: 34.958  on 45  degrees of freedom
AIC: 44.958

Number of Fisher Scoring iterations: 6
```

Figure 12. Summary Report of Logit- Cervical Cancer

KNN:

For KNN classifier, I use similar approach to determine the optimal k as in the previous problem. According to the ROC result shown below, I determine k=3 to be the optimal parameter as it yields the highest accuracy. With a training accuracy of 0.798 and testing accuracy of 0.909.



```
          Reference
Prediction no yes
       no  17   1
       yes  1   3

               Accuracy : 0.9091
                 95% CI : (0.7084, 0.9888)
    No Information Rate : 0.8182
    P-Value [Acc > NIR] : 0.2092
```

Figure 13. Confusion Matrix of KNN - Cervical Cancer

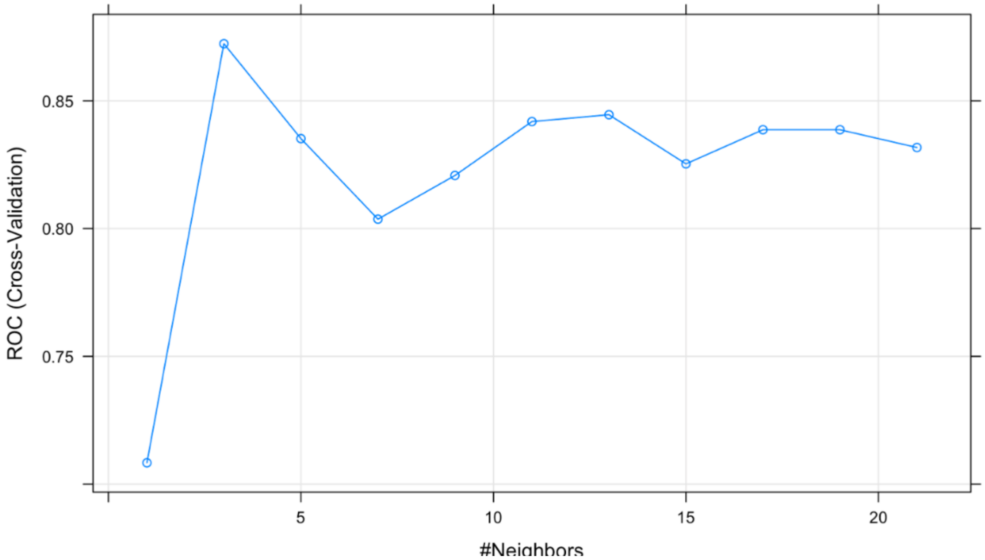
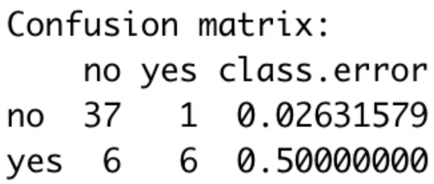


Figure 14. ROC of KNN - Cervical Cancer

Random Forest:

For random forest algorithm. I choose 100 trees to be included in the forest and a sampling with replacement method to balance between model efficiency and train time. The confusion matrix and feature importance are shown as below. The training accuracy is 0.4 while the testing accuracy is a whooping 0.95.



```
Confusion matrix:
    no yes class.error
no  37   1  0.02631579
yes  6   6  0.50000000
```

Figure 15. Confusion Matrix of Random Forest - Cervical Cancer

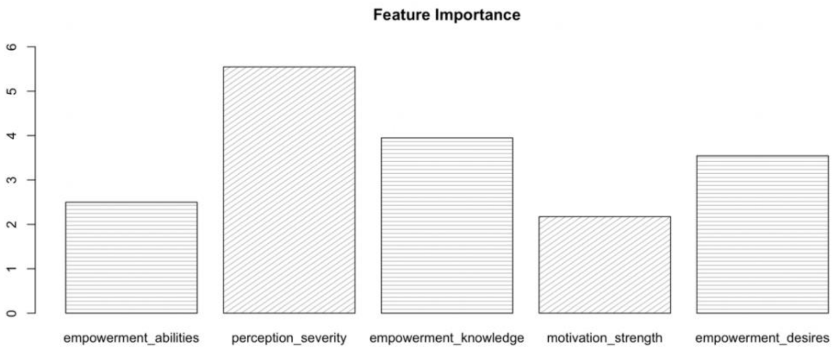


Figure 16. Feature Importance of RF - Cervical Cancer

Conclusion & Discussion

Even though the same three models provide some useful insight and generate reliable predictions for the banking dataset, they do not seem to perform well under this one. The main reason may be the lack of training data. The Cervical Cancer dataset only contains 72 entries, which is way too small for machine learning model to produce reliable predictions. All we can conclude from this study is that “perception_severity” and “motivation_strength” are statistically correlated to the Cervical Cancer, but we cannot rely on the prediction that the models make. The lack of training data causes the KNN model to be extremely over-fitted, having out-of-sample accuracy significantly smaller than the in-sample accuracy. It also causes the random forest model to make weird judgement when determining the most important split. For the forest model, different random state will lead to totally different result, which is saying that the model is not robust nor generalized enough to make reliable predictions.