STA 141A Project Report

Customer Segmentation Classification

Group Members	Contribution	E-mail
Jake Gwo	Introduction, Exploratory analysis	jzgwo@ucdavis.edu
Qinyi Qiu	Logistic regression models	qyqiu@uncdavis.edu
Mingrui Gao	Logistic regression models	mrgao@ucdavis.edu
Guanghao He	KNN models	ghhe@ucdavis.edu
Qiangwei Li	LDA，Conclusion	qwkli@ucdavis.edu

Instructor: Emanuela Furfaro

STA 141A - Fundamentals of Statistical Data Science

University of California, Davis

December 8, 2021

Introduction

An automobile company plans to enter new markets, and they have deduced that the behavior of the new market is similar to their existing market, in which the sales team had classified all customers into 4 segments (A, B, C, D) and found this strategy to work exceptionally well. In this study our aim is to analyze data of the previous customers and try to classify potential customers from the new automobile industry market into the four segments.

Data Description and Questions of Interest

The training dataset includes 8068 observations from the existing market customers and 11 variables that are: ID, gender of the customer, marital status, age, graduation status, profession, work experience in years, spending score, number of family members, and an anonymised category provided by the company (Var_1). Most of the variables in the dataset are categorical except ID, age, work experience and number of family members. The test data contains 2627 observations and the same variables.

By utilizing the variables along with different methodologies such as logistic regression, K-nearest neighbors algorithm, and linear discriminant analysis, our goal is to find the most suitable model to classify the customers. Key questions that we also wish to address are what are the variables that play the biggest role in prediction, and what are the limitations of our methods.

Data Exploration

As mentioned, the training dataset contains 3 numerical variables and 6 categorical variables as the potential predictors. We first check the distributions of numerical variables. From Figure 1 we see that age is approximately normally distributed with mean around 40. Work experience is heavily right skewed as the majority are 0 or 1. Family size distribution is also right skewed with the most common being 2. To analyze the relationships between them, we plot a scatter plot matrix with the correlation coefficients displayed in Figure 2. The plot shows low correlation for all pairs.

For the categorical variables, including the response (Segmentation), we created pie charts to inspect the classes each variable has, as well as the percentage distributions. We see that the customers are: approximately half-and-half in terms of gender distribution; forty-sixty in terms of marital history and graduation status; working in nine total professions, with health care being the most; mostly having average spending score; mostly classified into the second anonymised category by the company; evenly distributed in terms of final segmentation.

It is also important to discover interactions between the predictors for modeling, so we created side-by-side boxplots of the categorical predictors versus the numerical predictors, shown in Figure 4 to 6. From the plots, the key interactions for age are spending score, marital history, graduation, and profession. Specifically, we see that customers who have high spending scores are having a median of 60 in age, which is older compared to the average ages of both average and low spending score customers. Also, customers who were married at least once are generally older than those who were never married, customers who have graduated are generally older than those who have not graduated, and customers that are lawyers are generally the oldest in the group while those who work in health care are generally the youngest. Besides these, the other interactions shown are not as significant.

Finally, we check how the classes in segmentation are distributed over the three quantitative variables. In Figure 7, it shows that distribution of classes changes over different ages. For example, the proportion of class D customers is the highest around the age of 20, and the proportion of all classes is about the same as age gets larger. For different values of work experience and family size, the distribution of segmentations remains relatively the same.

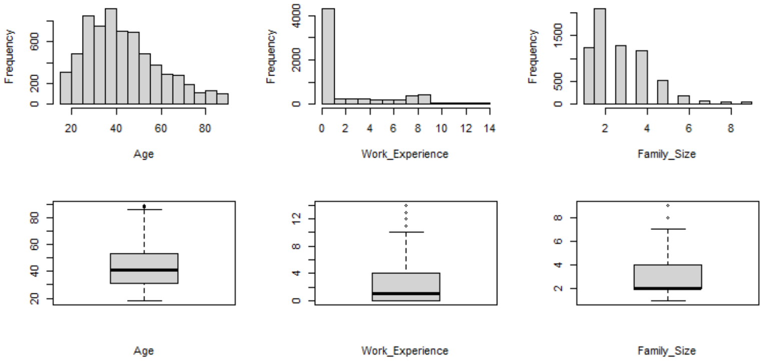


Figure 1: Histograms and boxplots of the numericals

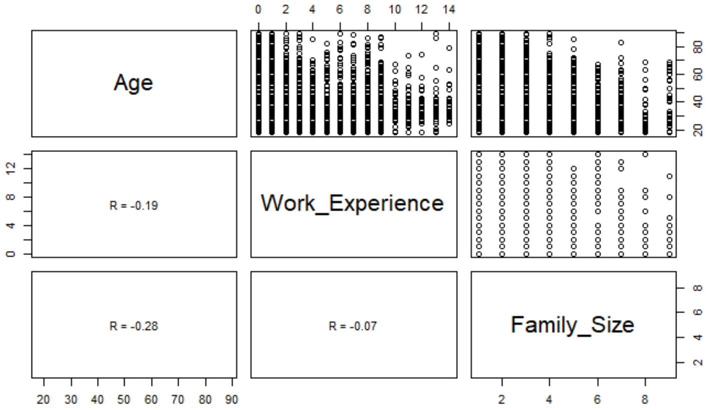


Figure 2: scatterplot matrix with correlation

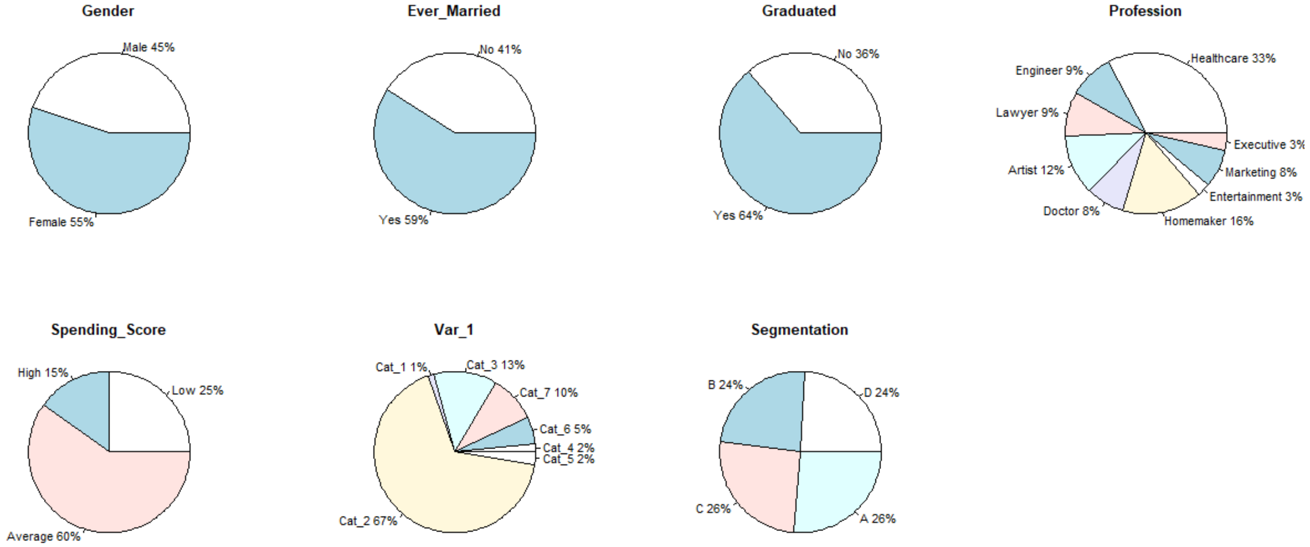


Figure 3: Pie charts of the categoricals

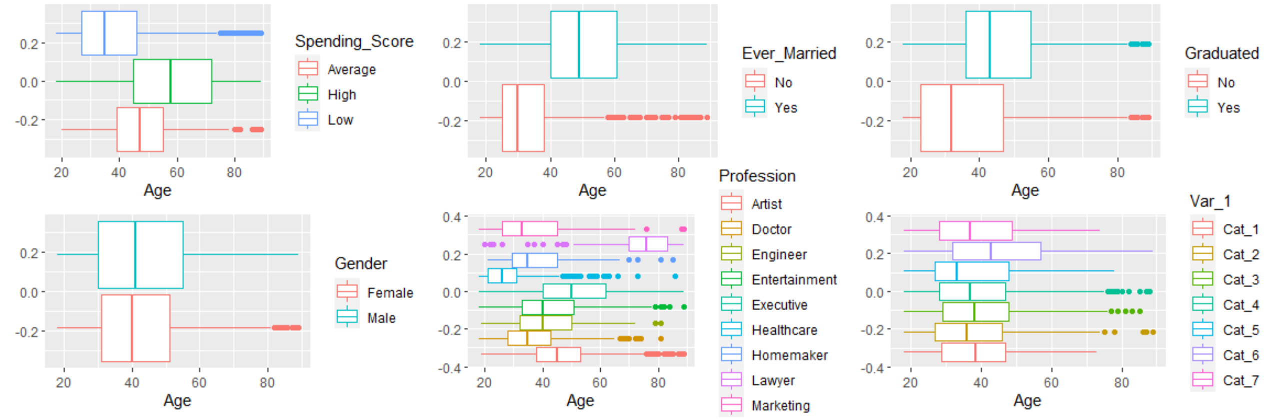


Figure 4: Side-by-side boxplots of the categoricals against Age

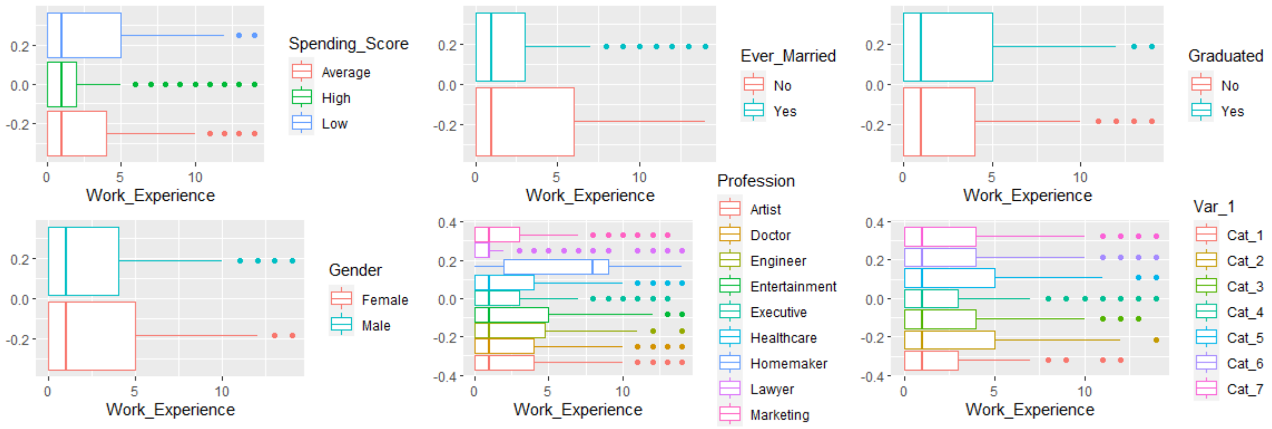


Figure 5: Side-by-side boxplots of the categoricals against Work Experience

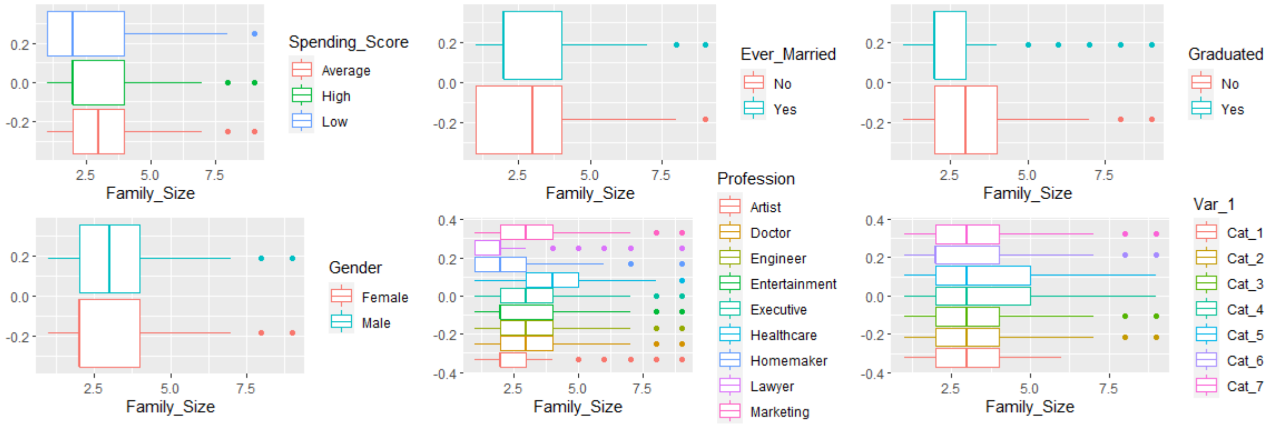


Figure 6: Side-by-side boxplots of the categoricals against Family Size

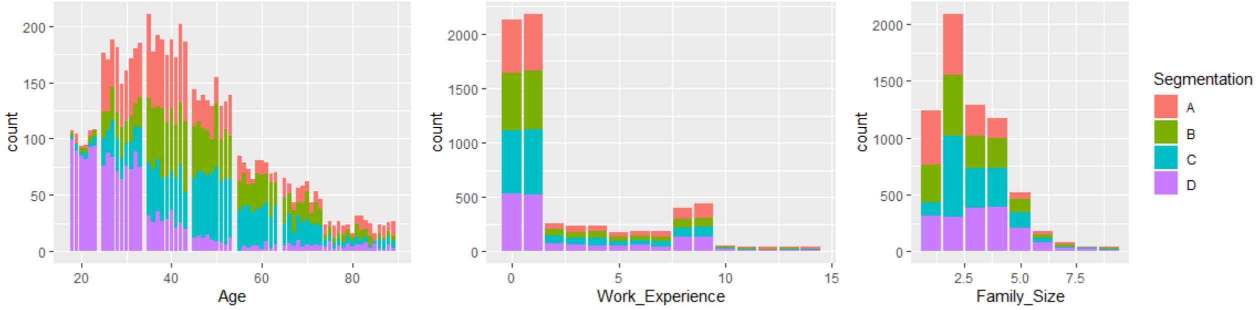


Figure 7: Composition of cases with respect to segmentation

Methods

1. Logistic Regression

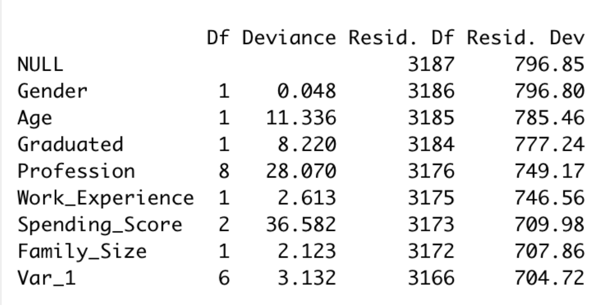
Logistic regression is a statistical model that uses a logistic function to model a binary dependent_variable. The output of the regression model is the likelihood of an event happening. When using logistic regression to classify binary categorical response variables, the usual method is to replace the response variable with 0 and 1 before modeling.

The dataset in our project contains four different segmentations. If we still want to try logistic regression for classification, the only way is to apply logistic regression three times and classify one segmentation each time. However, the error rate would become very high. So, in this project, we separated the dataset into six binary groups, performed logistic regression on each group, and used them to find the variables that play the biggest role in prediction.

For each model, we first fitted a full model on every variable, and used the stepwise regression procedures to choose the best model. Specifically, we use the backward elimination, which starts from a full model, and delete variables that give the biggest improvement of the criterion regarding AIC, until the criterion cannot be improved anymore. Finally, we investigated the ANOVA table of each model, and made conclusions of the variables that play the biggest role in prediction.

Model 1: segmentations A & B

The accuracy of this model is 0.6480552. As shown in the table (Figure 8), when sequentially adding the Age, Graduated, Profession, Spending_Score variables in the model, the residual deviance decreases more than adding other variables in the model. This indicates that these four variables play a relatively more important role in the model.



```
                Df Deviance Resid. Df Resid. Dev
NULL                             3187     796.85
Gender           1    0.048      3186     796.80
Age              1   11.336      3185     785.46
Graduated        1    8.220      3184     777.24
Profession       8   28.070      3176     749.17
Work_Experience  1    2.613      3175     746.56
Spending_Score   2   36.582      3173     709.98
Family_Size      1    2.123      3172     707.86
Var_1            6    3.132      3166     704.72
```

Figure 8: ANOVA table of the first logistic model

Model 2: segmentations A & C

The accuracy of this model is 0.7419065. From the ANOVA table we can observe that when sequentially adding variables into the model, the residual deviance decreases more when adding the variables Graduated, Profession, Spending_Score. This indicates that these three var iables play a relatively more important role in the model.

Model 3: segmentations A & D

The accuracy of this model is 0.7571894. From the ANOVA table we can observe that when sequentially adding variables into the model, the residual deviance decreases more when adding the variables Ever_Married, Age, Graduated, Profession. This indicates that these four variables play a relatively more important role in this model.

Model 4: segmentations B & C

The accuracy of this model is 0.6479344. From the ANOVA table we can observe that when sequentially adding variables into the model, the residual deviance decreases more when adding the variables Profession and Spending_Score. This indicates that these two variables play a relatively more important role in this model.

Model 5: segmentations B & D

The accuracy of this model is 0.8224692. From the ANOVA table we can observe that when sequentially adding variables into the model, the residual deviance decreases more when adding the variables Ever_Married, Age, Graduated, Profession, Spending_Score. This indicates that these five variables play a relatively more important role in this model.

Model 6: segmentations C & D

The accuracy of this model is 0.851021. From the ANOVA table we can observe that when sequentially adding variables into the model, the residual deviance decreases more when adding the variables Age, Graduated, Profession, Spending_Score. This indicates that these four variables play a relatively more important role in this model.

From the six logistic regression model, we can observe that generally, the Graduated, Profession, Spending_Score variables play important roles in segmentation for A,B,C and D. Besides, since the accuracies for performing logistic regression on A&D, B&D, C&D are pretty high than the others, we can make the conclusion that the segmentation D is more easily to be segmented from the others.

2. k-NN

k-NN is the abbreviation for K-Nearest Neighbors. It is a non-parametric method not only used for classification but also used for regression. The reason for non-parametric is for that it does not set up any assumptions on the underlying data distribution. In general, we use k-NN classification when predicting categorical outcome and apply k-NN regression when predicting a continuous outcome. The principle of the k-NN algorithm is when we make predictions or classification, there are nonlinear boundaries separating labels or classes of interest. K-NN would examine the classes or labels of points or neighbors which are surrounding the point we are interested in and classify the interest points or data based on the similar measure. Specifically, the similar measure is defined as the distance between two data and the popular method to calculate the distance is Euclidean distance. Besides that, based on different data properties, the distance measure could be different, such as Manhattan, Minkowski, Jaccard distance and so on. Thus, efficiency of the k-NN algorithm mainly depends on the optimal number of k and generally we could apply a cross-validation approach to get the relatively optimal K. In short, the method allows us to resampling train data so that it evaluates performance of different possible k based on those data applying the best k into the model, which could minimize the validation error. In our study, in order to train a model to better predict the segmentation (A,B,C,D) of the customer, we designed three k–NN models based on different scenarios. Two of them separately apply different distance measures, Euclidean distance and Jaccard distance.

As we all know, we need to apply numeric variables for calculating the distance between data points. Thus, we firstlt include all three quantitative variables as predictor variables in the first k-NN model. They are “Age”, “Work_Experience” and “Family_Size” and our outcome variable is “Segmentation”. As three predictor variables are different from each in scale, we used the scale function to standardize these predictors. After that, before we use the train function of the class package we learned this semester to run k-NN classification, we initially apply 10-fold cross validation in the training data with 75% training percentage sampling to find the optimal number of neighbors. According to the train function in Caret package(Caret, pp.163-169), we could achieve that goal and generate a plot.

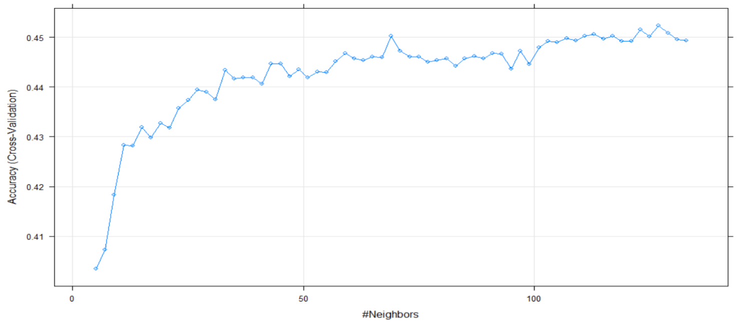


Figure 9

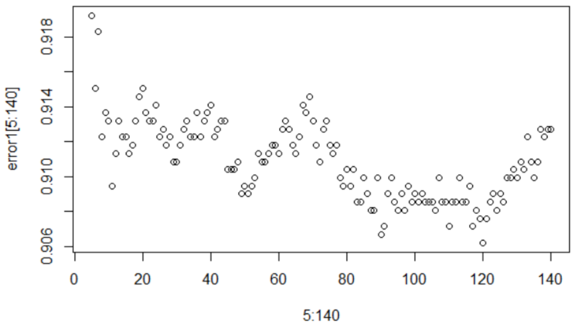


Figure 10

Based on it, we are aware the optimal k for the final model to use is 127 (Figure 9). In order to test that, we use the train function of the class package to run classification and plot the final model prediction error rate in an interval including the optimal k. We can see the lowest error rate of the model appears when k is roundly equal to 120, though the error rate is pretty high, almost 90% (Figure 10). It is reasonable to predict, as this model only has three quantitative variables and does not contain enough information.

In the second k-NN model, we decide to transform all of the categorical variables into numeric. Same as the first model procedure, we apply 10-fold cross validation in the training data with 75% training percentage sampling to get the optimal k equals to 47 (Figure 11).

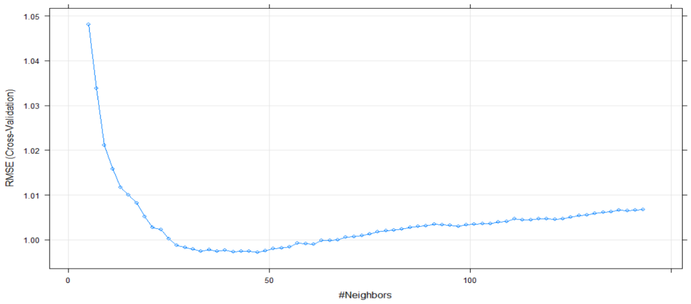


Figure 11

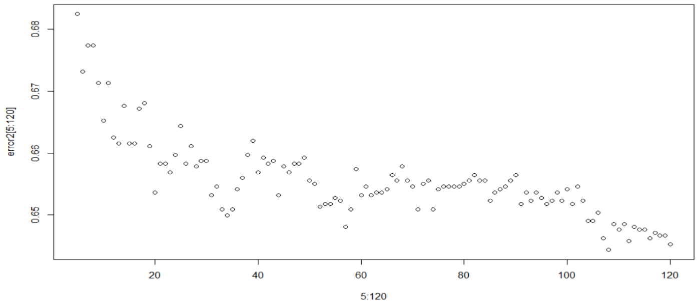


Figure 12

Then, we run a classification function and plot the error rate interval using the same procedure as mentioned in the first model. At this time, we notice that the overall error rate improves a lot and when k equals to 47, the error rate approximately equals to 65%. However, we do find the error rate of k that equals to 47 obtained from cross validation approach is relatively higher than the error rate of 64% obtained from the k around 100 (Figure 12). In fact, the error can be expected, as the k value of cross validation approach is trained on the 75% training percentage data and tested on the remaining 25% data. Thus, when we apply the k-NN classification with the optimal k value and test on the new test data, the final result may cause a certain error. Although the error rate improves a lot, the model we generated does not obtain practical meaning. As in the train function of the class package, the default distance measure method is Euclidean distance, which calculates the straight distance between two data points. However, the categorical variable does not fit for that distance approach. For example, the “Profession” variable has 9 different levels so that we convert each of them with a number from 1 to 9. However, this data of “Profession” does not contain these order relationships.

Therefore, in the third k-NN model, we found a method to calculate the distance of categorical variables. We first convert each of these categorical variables into dummy variables and then we could apply the Jaccard distance measure (Teknomo). It could measure the asymmetric information of binary and the distance defined as the size of interaction of two data divided by the union of them. We learned a knn function of the package named “Neighbr” that could apply k–NN classification with Jaccard distance(Neighbr, pp.3). However, when we apply that function to our dataset, the computational time of cost is too long and even costs a whole day to test one k. Thus, we have to sample a portion of data from our converted train data and test data. Lastly, we use 1000 samples as our training data and 300 samples as our test data. After applying the k-NN classification with Jaccard distance, we plot the final model prediction error rate in a wide interval of k, from 10 to 200. We can see the best error rate is relatively the lowest 66 % when k equals to around 50 (Figure 13).

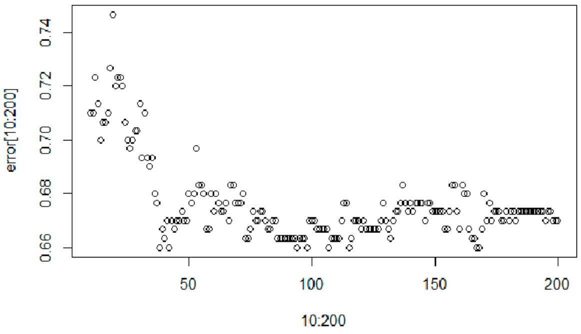


Figure 13

3. LDA

Linear Discriminant Analysis (LDA) is a sound machine learning technique and classification method for predicting categorical. The Linear Discriminant Analysis has two functions: dimensionality reduction and classification. So, this method can project multi-dimensional data to a low-dimensional plane, making our data categories easily distinguishable. And the dimension of the data after dimensionality reduction is called the discriminant function. After using LDA, only a few discriminant functions are left in the original data with many features. The primary problem with too high data dimensions (too many predictor variables) is that many dimensions cannot provide helpful information to the model and even interfere with model performance. When the number of dimensions increases, the data becomes sparse. Linear Discriminant Analysis is to help us reduce computational costs for classification tasks, but it can also be helpful by avoiding overfitting via minimizing the error in parameter estimation. In our case, by fitting the LDA model with our training data, we got the prior probability of groups, group means, Coefficients of linear discriminants, and Proportion of trace. According to the result, the Group means can give us some information like “GenderMale,” “Work_Experience,” “Family_Size” and “Var_1Cat_7” have almost the same value among the four segmentations (A, B, C, D), which means these variables are not that useful for predicting. However, “ProfessionHealthcare” has an especially high mean in D and low in others, meaning that this variable is very useful for predicting the D group. Importantly, we can see that the LDA model helps us to reduce our dimensions by classifying our 23 variables into only three features: “LD1”, “LD2,” and “LD3”. In other words, LDA uses fewer variables to predict ABCD four segmentation. The Coefficients of linear discriminants show us the correlation of 23 variables and the three LD, respectively. Furthermore, we calculate the confusion matrix to help us evaluate the performance of the classification model, and we further get the misclassification error rate: 0.6722377. The high error rate implies that linear function cannot perform well in helping us separate our clusters with our data.

Results and Conclusion

In summary, we created models adopting three different methodologies: logistic regression, k-nearest neighbor, and Bayes' classifier. Here we compare their performance and answer our questions of interest.

1）What is the most suitable model to classify the customers？

According to the result we get from the three different models, we found that logistic is the most efficient method to do the classification for this dataset, since it has the highest accuracy rate. However, the logistic model can only analyze two segmentations simultaneously. Comparison between LDA and k-NN, though the third k-NN model seems to have a relatively higher accuracy rate 0.34 than that of LDA 0.33. However, as the k-NN model here does not include the whole dataset, instead it samples a proportion of training data and test data. Overall, the LDA model is more comprehensive.

2）What are the variables that play the biggest role in prediction?

According to the logistic regression models, different groups of segmentation have different significant variables. So we cannot give an absolute idea of what are the variables that play the biggest role in classifying the customers into four segments A, B, C and D. However, by summing up the all six models, we can observe that the Graduated, Profession, Spending_Score variables have high influence in deviance reduction for all six models. Hence, we can conclude that these are the variables that play the biggest role in prediction. According to the Group means in the LDA model, we can see that Profession, Ever_Married and Age have more influence on predicting the four segmentations A, B, C and D. However, we cannot tell who plays the biggest role in prediction only based on the Group means.

3）What are the limitations of our methods？

The limitations using the logistic model mainly is that it can only classify two different categories simultaneously. As a result, we can only analyze between two of the four segmentations which may lead to ignoring the relationship between them and cause errors on the classification.

The third k-NN model requires too much time and computational power, so our suggestion is that if the dataset is mostly full of categorical variables and also would like to consider the notion of distance. It may be better to consider "partitioning around medoids (PAM)".

As shown above, the highest accuracies we achieved using our selected methods were around thirty percent. This could be due to either due to the low ceiling of the predictive ability of the data, or the inadequacy of the methods, which we did find limitations of. If the latter is true, methodologies such as neural networks or decision trees may be more suitable for the task.

References

(1) Data source
https://www.kaggle.com/kaushiksuresh147/customer-segmentation

(2) Teknomo, Kardi. *Jaccard's Coefficient*
https://people.revoledu.com/kardi/tutorial/Similarity/Jaccard.html.

(3) Package ‘Caret’
https://cran.r-project.org/web/packages/caret/caret.pdf.

(4) Package ‘neighbr’
https://cran.r-project.org/web/packages/neighbr/neighbr.pdf

STA 141A Project Code

Customer Segmentation Classification

Authors: Jake Gwo, Qinyi Qiu, Mingrui Gao, Guanghao He, Qiangwei Li

```r
library(readr)
library(ggplot2)
library(gridExtra)

#loading data
train.c <- read_csv("Train.csv")
test.c <- read_csv("Test.csv")

train <- na.omit(train.c)
test <- na.omit(test.c)
str(train)
```

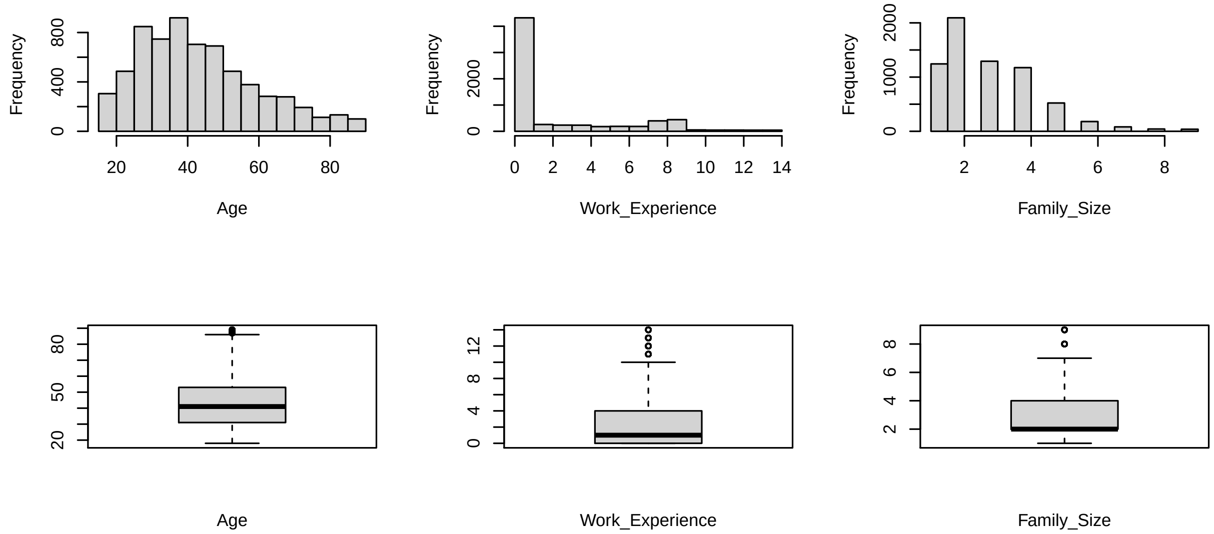
```
## tibble [6,665 x 11] (S3: tbl_df/tbl/data.frame)
##  $ ID             : num [1:6665] 462809 466315 461735 461319 460156 ...
##  $ Gender         : chr [1:6665] "Male" "Female" "Male" "Male" ...
##  $ Ever_Married   : chr [1:6665] "No" "Yes" "Yes" "Yes" ...
##  $ Age            : num [1:6665] 22 67 67 56 32 33 61 55 26 19 ...
##  $ Graduated      : chr [1:6665] "No" "Yes" "Yes" "No" ...
##  $ Profession     : chr [1:6665] "Healthcare" "Engineer" "Lawyer" "Artist" ...
##  $ Work_Experience: num [1:6665] 1 1 0 0 1 1 0 1 1 4 ...
##  $ Spending_Score : chr [1:6665] "Low" "Low" "High" "Average" ...
##  $ Family_Size    : num [1:6665] 4 1 2 2 3 3 3 4 3 4 ...
##  $ Var_1          : chr [1:6665] "Cat_4" "Cat_6" "Cat_6" "Cat_6" ...
##  $ Segmentation   : chr [1:6665] "D" "B" "B" "C" ...
##  - attr(*, "na.action")= 'omit' Named int [1:1403] 2 5 13 14 25 34 40 44 46 48 ...
##   ..- attr(*, "names")= chr [1:1403] "2" "5" "13" "14" ...
```

EDA

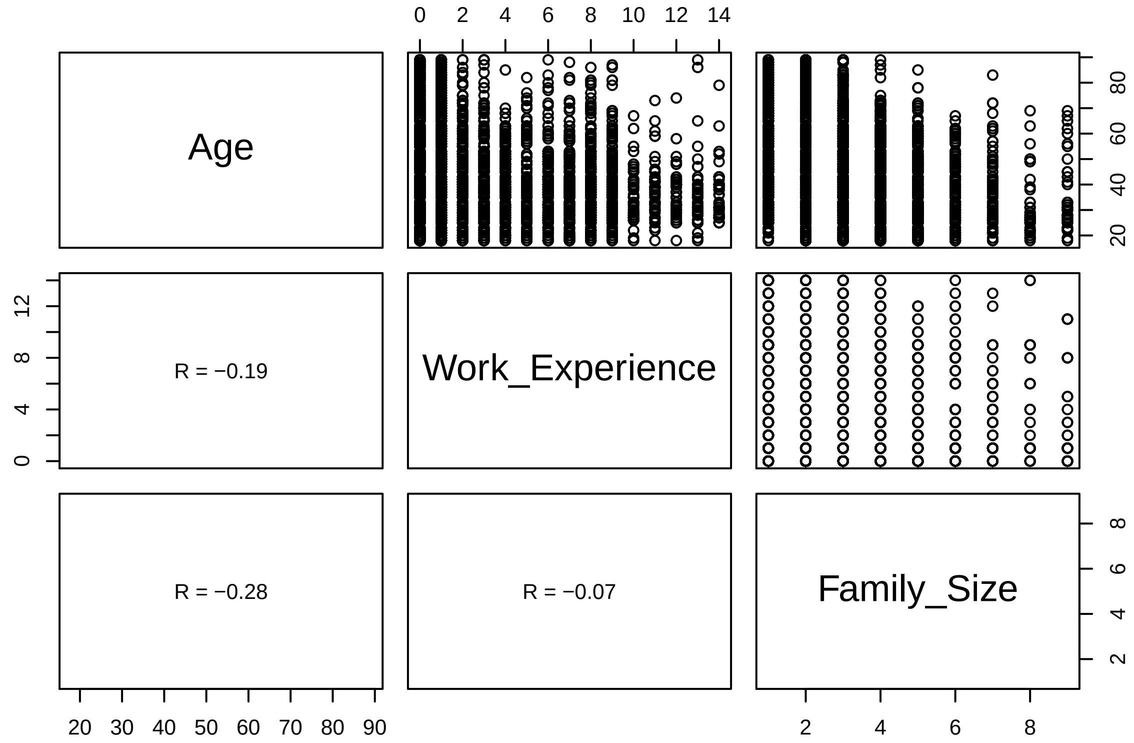
```r
summary(train[, c(4,7,9)])
```

```
##       Age        Work_Experience   Family_Size
##  Min.   :18.00   Min.   : 0.000   Min.   :1.000
##  1st Qu.:31.00   1st Qu.: 0.000   1st Qu.:2.000
##  Median :41.00   Median : 1.000   Median :2.000
##  Mean   :43.54   Mean   : 2.629   Mean   :2.841
##  3rd Qu.:53.00   3rd Qu.: 4.000   3rd Qu.:4.000
##  Max.   :89.00   Max.   :14.000   Max.   :9.000
```

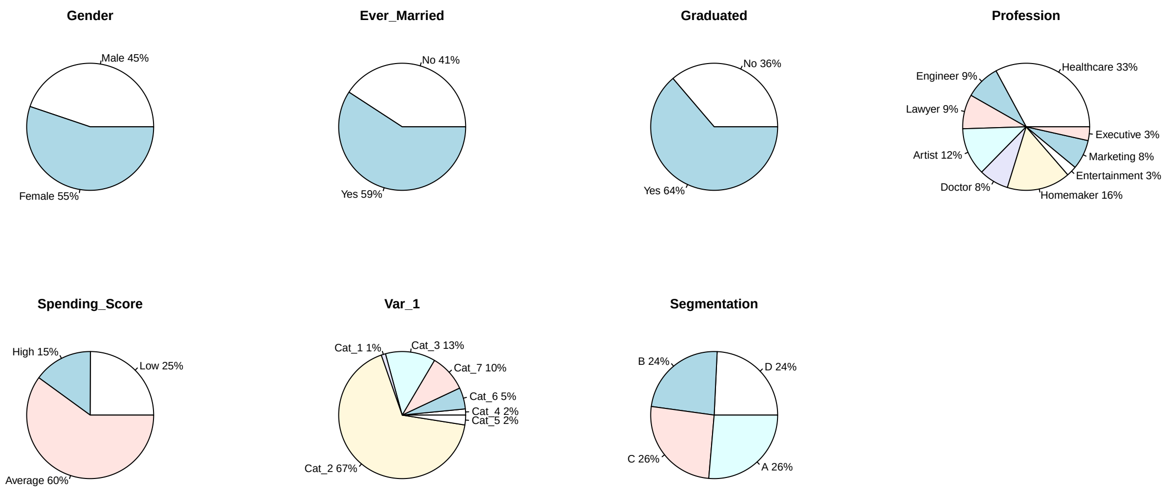
```r
layout(mat = matrix(c(1,2,3,4,5,6), ncol = 3))
for(j in 1:ncol(train)){
  if(j %in% c(4,7,9)) { #if a numerical variable
    hist(train[[j]], xlab = colnames(train)[j], main = "")
    boxplot(train[[j]], xlab = colnames(train)[j], main = "")
  }
}
```



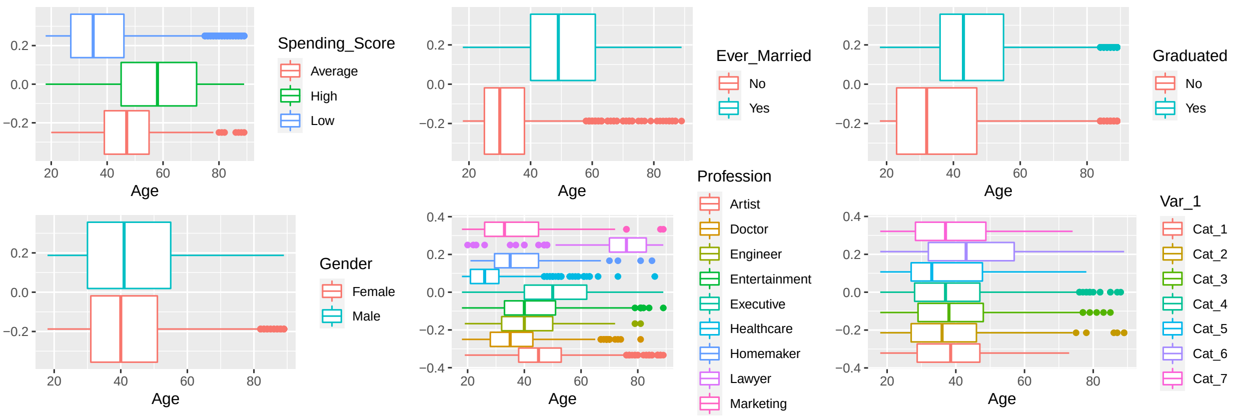
```r
panel.cor <- function(x, y) {
    par(usr = c(0, 1, 0, 1))
    r <- round(cor(x, y, use = "complete.obs"), 2)
    txt <- paste0("R = ", r)
    cex.cor <- 0.8/strwidth(txt)
    text(0.5, 0.5, txt, cex = cex.cor * r)
}
pairs(train[c("Age", "Work_Experience", "Family_Size")], lower.panel=panel.cor)
```



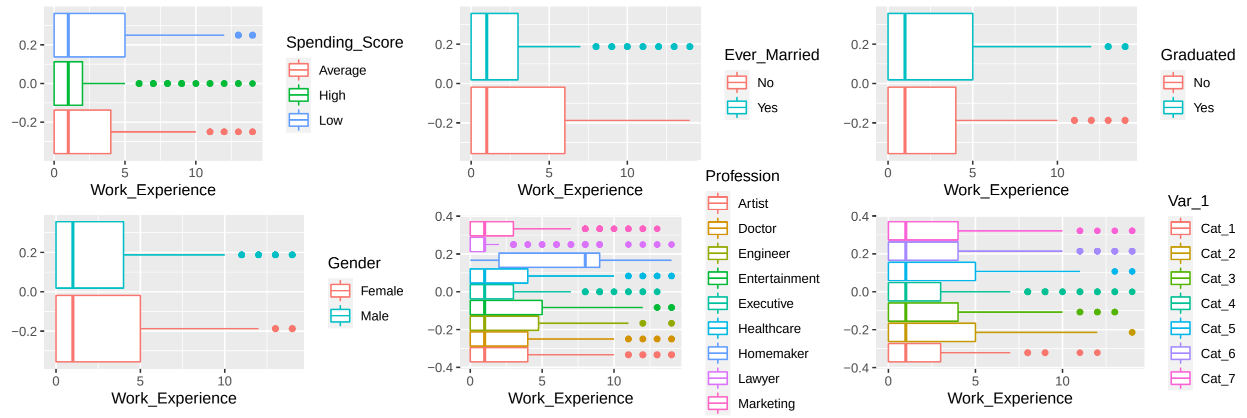
```
par(mfrow=c(2,4))
n <- nrow(train)
for(i in 1:ncol(train)){
  if(!i %in% c(1,4,7,9)) { #if not a numerical variable
    lbls <- unique(train[,i])[[1]]
    pct <- round(100*table(train[,i])/n)
    lab <- paste(lbls, pct)
    lab <- paste(lab,'%',sep='')
    pie(table(train[,i]), main = colnames(train)[i], labels=lab)
  }
}
```



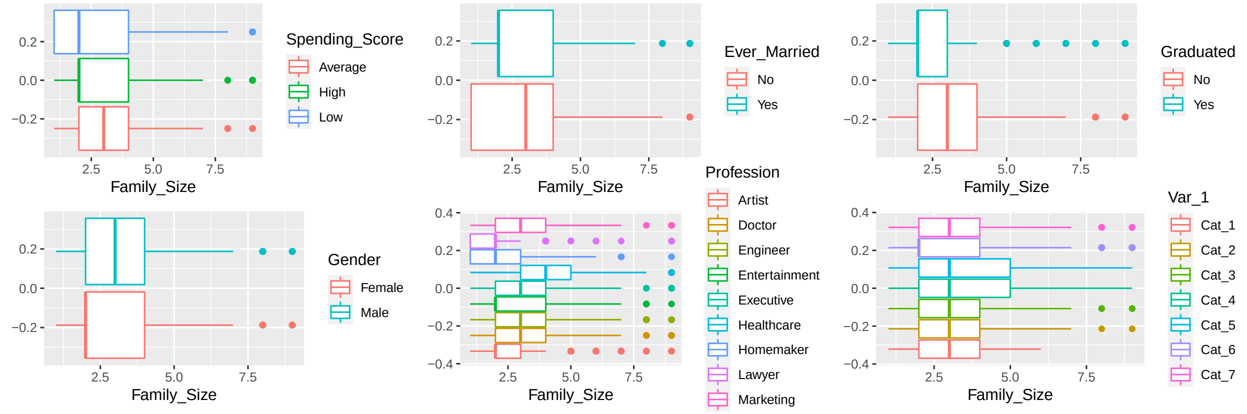
```
interaction_box <- function(quant_var) {
  name <- deparse(substitute(quant_var))
  name <- substring(name, 7)
  plots <- list(
    ggplot(train, aes(x=quant_var, color=Spending_Score))+geom_boxplot()+labs(x=name),
    ggplot(train, aes(x=quant_var, color=Ever_Married))+geom_boxplot()+labs(x=name),
    ggplot(train, aes(x=quant_var, color=Graduated))+geom_boxplot()+labs(x=name),
    ggplot(train, aes(x=quant_var, color=Gender))+geom_boxplot()+labs(x=name),
    ggplot(train, aes(x=quant_var, color=Profession))+geom_boxplot()+labs(x=name),
    ggplot(train, aes(x=quant_var, color=Var_1))+geom_boxplot()+labs(x=name))
  grid.arrange(grobs = plots, nrow = 2)
}

interaction_box(train$Age)
```



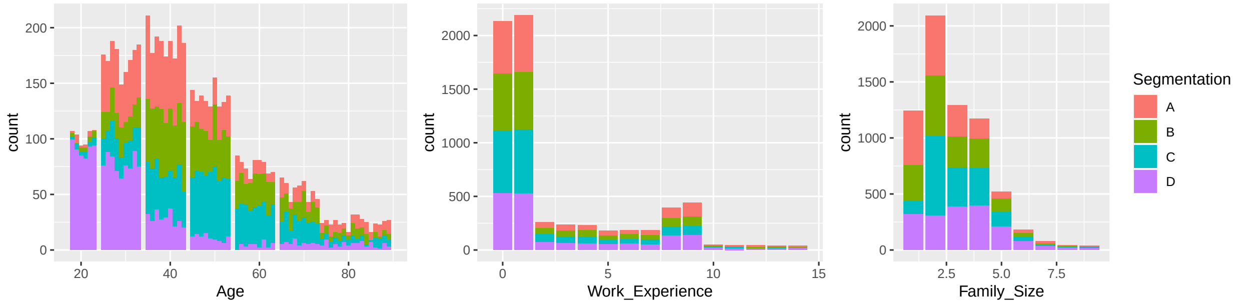
```
interaction_box(train$Work_Experience)
```



```
interaction_box(train$Family_Size)
```



```
bp1 <- ggplot(train, aes(x=Age))+geom_bar(aes(fill=Segmentation)) + theme(legend.position="none")
bp2 <- ggplot(train, aes(x=Work_Experience))+geom_bar(aes(fill=Segmentation)) + theme(legend.position="
bp3 <- ggplot(train, aes(x=Family_Size))+geom_bar(aes(fill=Segmentation))
grid.arrange(bp1,bp2,bp3, nrow = 1)
```



```
trainA <- train[which(train$Segmentation=='A'),]
trainB <- train[which(train$Segmentation=='B'),]
trainC <- train[which(train$Segmentation=='C'),]
trainD <- train[which(train$Segmentation=='D'),]

testA <- test[which(test$Segmentation=='A'),]
testB <- test[which(test$Segmentation=='B'),]
testC <- test[which(test$Segmentation=='C'),]
testD <- test[which(test$Segmentation=='D'),]
```

Model A&B

```
trainAB <- rbind(trainA,trainB)
trainAB$Segmentation <- ifelse(trainAB$Segmentation=="A",1,0)
glmAB <- glm(Segmentation~., data=trainAB, family='binomial')

stepAIC(glmAB, scope=list(upper=glmAB, lower=~1), direction = "backward",
    k = 2, trace = FALSE)
```

```
##
## Call:  glm(formula = Segmentation ~ Gender + Age + Graduated + Profession +
##     Work_Experience + Spending_Score + Family_Size + Var_1, family = "binomial",
##     data = trainAB)
##
## Coefficients:
##             (Intercept)               GenderMale                      Age
##                 0.30147                  0.27053                 -0.02304
##            GraduatedYes         ProfessionDoctor       ProfessionEngineer
##                -0.36533                  0.52883                  0.79281
## ProfessionEntertainment      ProfessionExecutive     ProfessionHealthcare
##                 0.91209                  0.26173                  0.05395
##     ProfessionHomemaker         ProfessionLawyer      ProfessionMarketing
##                 0.40270                  1.37157                  0.74073
##         Work_Experience       Spending_ScoreHigh        Spending_ScoreLow
##                 0.03822                  0.38152                  1.03880
##             Family_Size               Var_1Cat_2               Var_1Cat_3
##                -0.10022                 -0.43675                 -0.05828
##              Var_1Cat_4               Var_1Cat_5               Var_1Cat_6
##                 0.19515                 -0.50795                 -0.13380
##              Var_1Cat_7
##                 0.02278
##
## Degrees of Freedom: 3187 Total (i.e. Null);  3166 Residual
## Null Deviance:       4419
## Residual Deviance: 4030  AIC: 4074
```

```
glmAB_ <- glm(formula = Segmentation ~ Gender + Age + Graduated + Profession +
    Work_Experience + Spending_Score + Family_Size + Var_1, family = "binomial",
    data = trainAB)

predictAB_ <- ifelse(glmAB_$fitted.values>0.5,'A','B')
confusionAB_ <- table(predictAB_, ifelse(trainAB$Segmentation==1,'A','B'), dnn=c("Predicted","True"))
confusionAB_
```

```
##          True
## Predicted    A    B
##         A 1063  575
##         B  553  997
```

```r
sum(diag(confusionAB_)) / sum(confusionAB_)
```

```
## [1] 0.6461731
```

```r
anova(glmAB_)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Segmentation
##
## Terms added sequentially (first to last)
##
##
##                 Df Deviance Resid. Df Resid. Dev
## NULL                             3187     4418.9
## Gender           1    0.191      3186     4418.7
## Age              1   45.651      3185     4373.1
## Graduated        1   33.516      3184     4339.5
## Profession       8  116.839      3176     4222.7
## Work_Experience  1   10.890      3175     4211.8
## Spending_Score   2  157.936      3173     4053.9
## Family_Size      1    9.861      3172     4044.0
## Var_1            6   14.051      3166     4030.0
```

Model A&C

```r
trainAC <- rbind(trainA,trainC)
trainAC$Segmentation <- ifelse(trainAC$Segmentation=="A",1,0)
glmAC <- glm(Segmentation~., data=trainAC, family='binomial')

stepAIC(glmAC, scope=list(upper=glmAC, lower=~1), direction = "backward",
    k = 2, trace = FALSE)
```

```
##
## Call:  glm(formula = Segmentation ~ Gender + Age + Graduated + Profession +
##     Work_Experience + Spending_Score + Family_Size + Var_1, family = "binomial",
##     data = trainAC)
##
## Coefficients:
##             (Intercept)               GenderMale                      Age
##                 0.92325                  0.25494                 -0.02350
##            GraduatedYes         ProfessionDoctor       ProfessionEngineer
##                -0.73378                  0.60248                  1.60905
## ProfessionEntertainment      ProfessionExecutive     ProfessionHealthcare
##                 1.43516                  0.42152                 -0.53428
##     ProfessionHomemaker         ProfessionLawyer      ProfessionMarketing
##                 1.11633                  1.41283                  0.40739
##         Work_Experience       Spending_ScoreHigh        Spending_ScoreLow
##                 0.04111                 0.69753                 1.61618
##             Family_Size              Var_1Cat_2              Var_1Cat_3
##                -0.32641                -0.23116                 0.15753
##              Var_1Cat_4              Var_1Cat_5              Var_1Cat_6
##                 0.84200                -0.05978                -0.37823
##              Var_1Cat_7
##                -0.03324
##
## Degrees of Freedom: 3335 Total (i.e. Null);  3314 Residual
## Null Deviance:       4621
## Residual Deviance: 3503  AIC: 3547
```

```
glmAC_ <- glm(formula = Segmentation ~ Gender + Age + Graduated + Profession +
    Work_Experience + Spending_Score + Family_Size + Var_1, family = "binomial",
    data = trainAC)

predictAC_ <- ifelse(glmAC_$fitted.values>0.5,'A','C')
confusionAC_ <- table(predictAC_, ifelse(trainAC$Segmentation==1,'A','C'), dnn=c("Predicted","True"))
confusionAC_
```

```
##          True
## Predicted    A    C
##         A 1176  421
##         C  440 1299
```

```
sum(diag(confusionAC_)) / sum(confusionAC_)
```

```
## [1] 0.7419065
```

```
anova(glmAC_)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Segmentation
##
## Terms added sequentially (first to last)
##
##
##                 Df Deviance Resid. Df Resid. Dev
## NULL                             3335     4621.4
## Gender           1     0.06      3334     4621.4
## Age              1    84.99      3333     4536.4
## Graduated        1   150.53      3332     4385.9
## Profession       8   343.33      3324     4042.5
## Work_Experience  1    18.29      3323     4024.2
## Spending_Score   2   366.09      3321     3658.1
## Family_Size      1    83.54      3320     3574.6
## Var_1            6    71.40      3314     3503.2
```

Model A&D

```
trainAD <- rbind(trainA,trainD)
trainAD$Segmentation <- ifelse(trainAD$Segmentation=="A",1,0)
glmAD <- glm(Segmentation~., data=trainAD, family='binomial')
```

```
stepAIC(glmAD, scope=list(upper=glmAD, lower=~1), direction = "backward",
    k = 2, trace = FALSE)
```

```
##
## Call:  glm(formula = Segmentation ~ Gender + Ever_Married + Age + Graduated +
##     Profession + Work_Experience + Spending_Score + Family_Size,
##     family = "binomial", data = trainAD)
##
## Coefficients:
##             (Intercept)               GenderMale          Ever_MarriedYes
##                 1.13813                 -0.37499                  0.43830
##                     Age             GraduatedYes         ProfessionDoctor
##                 0.02198                  0.61117                 -1.12548
##      ProfessionEngineer  ProfessionEntertainment      ProfessionExecutive
##                -0.84361                 -0.65089                 -1.46281
##    ProfessionHealthcare      ProfessionHomemaker         ProfessionLawyer
##                -2.85500                 -1.56710                 -1.92905
##     ProfessionMarketing          Work_Experience      Spending_ScoreHigh
##                -2.24809                 -0.02241                 -0.09116
##       Spending_ScoreLow              Family_Size
##                -0.77079                 -0.12549
##
## Degrees of Freedom: 3372 Total (i.e. Null);  3356 Residual
## Null Deviance:       4670
## Residual Deviance: 3391  AIC: 3425
```

```
glmAD_ <- glm(formula = Segmentation ~ Gender + Ever_Married + Age + Graduated +
    Profession + Work_Experience + Spending_Score + Family_Size,
    family = "binomial", data = trainAD)

predictAD_ <- ifelse(glmAD_$fitted.values>0.5,'A','D')
confusionAD_ <- table(predictAD_, ifelse(trainAD$Segmentation==1,'A','D'), dnn=c("Predicted","True"))
confusionAD_
```

```
##          True
## Predicted    A    D
##         A 1279  482
##         D  337 1275
```

```
sum(diag(confusionAD_)) / sum(confusionAD_)
```

```
## [1] 0.7571894
```

```
anova(glmAD_)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Segmentation
##
## Terms added sequentially (first to last)
##
##
##                 Df Deviance Resid. Df Resid. Dev
## NULL                             3372     4670.1
## Gender           1     7.76      3371     4662.3
## Ever_Married     1   373.25      3370     4289.1
## Age              1   155.17      3369     4133.9
## Graduated        1   159.91      3368     3974.0
## Profession       8   537.50      3360     3436.5
## Work_Experience  1     1.92      3359     3434.6
## Spending_Score   2    25.30      3357     3409.3
## Family_Size      1    18.74      3356     3390.5
```

Model B&C

```
trainBC <- rbind(trainB,trainC)
trainBC$Segmentation <- ifelse(trainBC$Segmentation=="B",1,0)
glmBC <- glm(Segmentation~., data=trainBC, family='binomial')

stepAIC(glmBC, scope=list(upper=glmBC, lower=~1), direction = "backward",
    k = 2, trace = FALSE)
```

```
## 
## Call:  glm(formula = Segmentation ~ Gender + Age + Graduated + Profession + 
##     Spending_Score + Family_Size + Var_1, family = "binomial", 
##     data = trainBC)
## 
## Coefficients:
##             (Intercept)               GenderMale                      Age  
##                0.557458                 0.119099                -0.007687  
##            GraduatedYes         ProfessionDoctor       ProfessionEngineer  
##               -0.491081                 0.188615                 1.142981  
## ProfessionEntertainment      ProfessionExecutive     ProfessionHealthcare  
##                0.811586                 0.309467                -0.568501  
##     ProfessionHomemaker         ProfessionLawyer      ProfessionMarketing  
##                1.163062                 0.419431                -0.320363  
##      Spending_ScoreHigh        Spending_ScoreLow              Family_Size  
##                0.398582                 0.780922                -0.195134  
##              Var_1Cat_2               Var_1Cat_3               Var_1Cat_4  
##                0.130605                 0.171596                 0.806180  
##              Var_1Cat_5               Var_1Cat_6               Var_1Cat_7  
##                0.387076                -0.139419                 0.085298  
## 
## Degrees of Freedom: 3291 Total (i.e. Null);  3271 Residual
## Null Deviance:       4557 
## Residual Deviance: 4203  AIC: 4245
```

```
glmBC_ <- glm(formula = Segmentation ~ Gender + Age + Graduated + Profession +
    Spending_Score + Family_Size + Var_1, family = "binomial",
    data = trainBC)

predictBC_ <- ifelse(glmBC_$fitted.values>0.5,'B','C')
confusionBC_ <- table(predictBC_, ifelse(trainBC$Segmentation==1,'B','C'), dnn=c("Predicted","True"))
confusionBC_
```

```
##          True
## Predicted    B    C
##         B  930  517
##         C  642 1203
```

```r
sum(diag(confusionBC_)) / sum(confusionBC_)
```

```
## [1] 0.6479344
```

```r
anova(glmBC_)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Segmentation
##
## Terms added sequentially (first to last)
##
##
##                Df Deviance Resid. Df Resid. Dev
## NULL                            3291     4557.0
## Gender          1    0.039      3290     4557.0
## Age             1    5.684      3289     4551.3
## Graduated       1   47.564      3288     4503.7
## Profession      8  107.890      3280     4395.8
## Spending_Score  2  114.577      3278     4281.3
## Family_Size     1   29.763      3277     4251.5
## Var_1           6   48.745      3271     4202.8
```

Model B&D

```r
trainBD <- rbind(trainB,trainD)
trainBD$Segmentation <- ifelse(trainBD$Segmentation=="B",1,0)
glmBD <- glm(Segmentation~., data=trainBD, family='binomial')

stepAIC(glmBD, scope=list(upper=glmBD, lower=~1), direction = "backward",
    k = 2, trace = FALSE)
```

```
##
## Call:  glm(formula = Segmentation ~ Gender + Ever_Married + Age + Graduated +
##     Profession + Work_Experience + Spending_Score + Var_1, family = "binomial",
##     data = trainBD)
##
## Coefficients:
##            (Intercept)               GenderMale          Ever_MarriedYes
##                0.79601                 -0.58881                  0.33755
##                    Age             GraduatedYes          ProfessionDoctor
##                0.03693                  0.91022                 -1.56430
##      ProfessionEngineer  ProfessionEntertainment      ProfessionExecutive
##               -1.68466                 -1.57940                 -1.71245
##    ProfessionHealthcare      ProfessionHomemaker         ProfessionLawyer
##               -2.81064                 -1.94790                 -2.80849
##     ProfessionMarketing          Work_Experience       Spending_ScoreHigh
##               -2.95069                 -0.07615                 -0.42440
##       Spending_ScoreLow              Var_1Cat_2               Var_1Cat_3
##               -1.77249                  0.92486                  0.56443
##              Var_1Cat_4               Var_1Cat_5               Var_1Cat_6
##                0.01167                  0.79443                  0.41402
##              Var_1Cat_7
##                 0.55725
##
## Degrees of Freedom: 3328 Total (i.e. Null);  3307 Residual
## Null Deviance:       4605
## Residual Deviance: 2752  AIC: 2796
```

```
glmBD_ <- glm(formula = Segmentation ~ Gender + Ever_Married + Age + Graduated +
    Profession + Work_Experience + Spending_Score + Var_1, family = "binomial",
    data = trainBD)

predictBD_ <- ifelse(glmBD_$fitted.values>0.5,'B','D')
confusionBD_ <- table(predictBD_, ifelse(trainBD$Segmentation==1,'B','D'), dnn=c("Predicted","True"))
confusionBD_
```

```
##          True
## Predicted    B    D
##         B 1290  309
##         D  282 1448
```

```
sum(diag(confusionBD_)) / sum(confusionBD_)
```

```
## [1] 0.8224692
```

```
anova(glmBD_)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Segmentation
##
## Terms added sequentially (first to last)
##
##
##                 Df Deviance Resid. Df Resid. Dev
## NULL                             3328     4604.7
## Gender           1    10.31      3327     4594.4
## Ever_Married     1   784.05      3326     3810.3
## Age              1   181.46      3325     3628.9
## Graduated        1   227.83      3324     3401.0
## Profession       8   461.05      3316     2940.0
## Work_Experience  1    24.41      3315     2915.6
## Spending_Score   2   144.09      3313     2771.5
## Var_1            6    19.37      3307     2752.1
```

Model C&D

```
trainCD <- rbind(trainC,trainD)
trainCD$Segmentation <- ifelse(trainCD$Segmentation=="C",1,0)
glmCD <- glm(Segmentation~., data=trainCD, family='binomial')

stepAIC(glmCD, scope=list(upper=glmCD, lower=~1), direction = "backward",
    k = 2, trace = FALSE)
```

```
##
```

```
## Call:  glm(formula = Segmentation ~ Gender + Age + Graduated + Profession + 
##     Work_Experience + Spending_Score + Family_Size + Var_1, family = "binomial", 
##     data = trainCD)
## 
## Coefficients:
##             (Intercept)               GenderMale                      Age  
##                -0.04193                 -0.39914                  0.03749  
##            GraduatedYes         ProfessionDoctor       ProfessionEngineer  
##                 1.20850                 -1.54516                 -2.28253  
## ProfessionEntertainment      ProfessionExecutive     ProfessionHealthcare  
##                -1.95901                 -1.71081                 -2.33919  
##     ProfessionHomemaker         ProfessionLawyer      ProfessionMarketing  
##                -2.50197                 -2.44365                 -2.47576  
##         Work_Experience       Spending_ScoreHigh        Spending_ScoreLow  
##                -0.06224                 -0.64709                 -2.18601  
##             Family_Size               Var_1Cat_2               Var_1Cat_3  
##                 0.27567                  0.42948                  0.37791  
##              Var_1Cat_4               Var_1Cat_5               Var_1Cat_6  
##                -0.77128                 -0.17661                  0.55369  
##              Var_1Cat_7  
##                 0.32445  
## 
## Degrees of Freedom: 3476 Total (i.e. Null);  3455 Residual
## Null Deviance:       4820 
## Residual Deviance: 2533  AIC: 2577
```

```
glmCD_ <- glm(formula = Segmentation ~ Gender + Age + Graduated + Profession +
    Work_Experience + Spending_Score + Family_Size + Var_1, family = "binomial",
    data = trainCD)

predictCD_ <- ifelse(glmCD_$fitted.values>0.5,'C','D')
confusionCD_ <- table(predictCD_, ifelse(trainCD$Segmentation==1,'C','D'), dnn=c("Predicted","True"))
confusionCD_
```

```
##          True
## Predicted    C    D
##         C 1426  224
##         D  294 1533
```

```
sum(diag(confusionCD_)) / sum(confusionCD_)
```

```
## [1] 0.851021
```

```
anova(glmCD_)
```

```
## Analysis of Deviance Table
## 
## Model: binomial, link: logit
## 
## Response: Segmentation
## 
## Terms added sequentially (first to last)
## 
## 
##                 Df Deviance Resid. Df Resid. Dev
## NULL                            3476     4819.8
## Gender           1     9.51      3475     4810.2
## Age              1   913.15      3474     3897.1
## Graduated        1   381.73      3473     3515.4
## Profession       8   556.12      3465     2959.2
## Work_Experience  1    17.81      3464     2941.4
## Spending_Score   2   302.33      3462     2639.1
## Family_Size      1    44.38      3461     2594.7
## Var_1            6    61.24      3455     2533.5
```

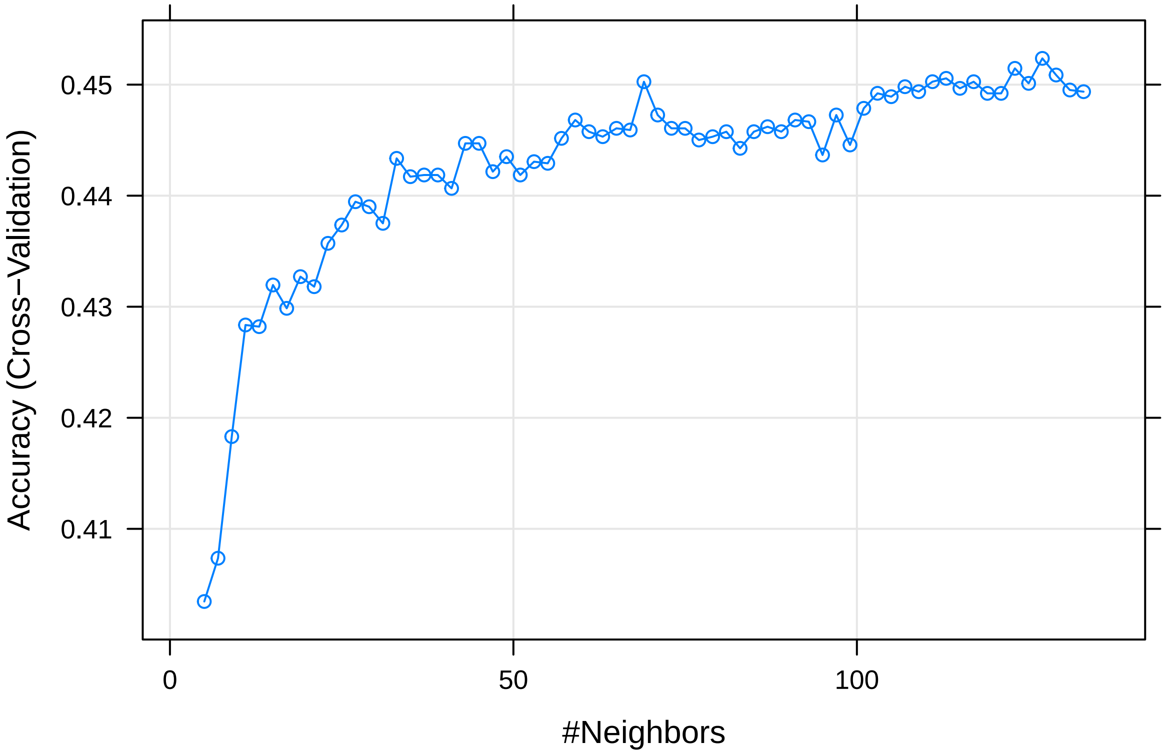
```
train <- na.omit(read_csv("Train.csv"))
test <- na.omit(read_csv("Test.csv"))
str(train)
```

First KNN Model:

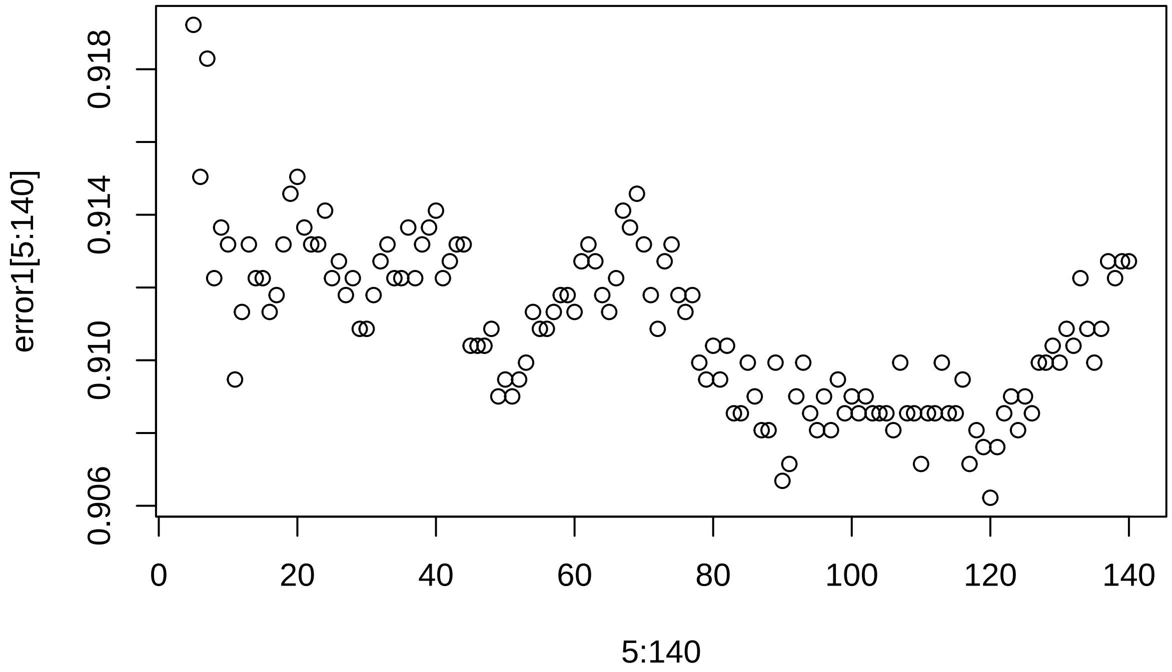
```
train.set <- train[,-1]
set.seed(123)
control <- trainControl(method = ’cv’,number = 10)
model <- train(Segmentation~.,train.set[,c(3,6,8,10)],
               method = ’knn’,preProcess="scale",
               trControl = control,tuneLength=65)
model
```

```
## k-Nearest Neighbors
##
## 6665 samples
##    3 predictor
##    4 classes: ’A’, ’B’, ’C’, ’D’
##
## Pre-processing: scaled (3)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5998, 5998, 5999, 5998, 5999, 5999, ...
## Resampling results across tuning parameters:
##
##   k    Accuracy   Kappa
##     5  0.4034577  0.2028580
##     7  0.4073558  0.2080819
##     9  0.4183075  0.2225901
##    11  0.4283561  0.2359381
##    13  0.4282068  0.2356714
##    15  0.4319577  0.2405476
##    17  0.4298576  0.2375914
##    19  0.4327107  0.2413202
##    21  0.4318057  0.2400950
##    23  0.4357087  0.2452556
##    25  0.4373597  0.2474201
##    27  0.4394580  0.2502838
##    29  0.4390050  0.2495916
##    31  0.4375096  0.2474977
##    33  0.4433612  0.2553345
##    35  0.4417123  0.2530781
##    37  0.4418633  0.2532456
##    39  0.4418615  0.2530975
##    41  0.4406614  0.2513887
##    43  0.4447123  0.2567816
##    45  0.4447137  0.2568130
##    47  0.4421627  0.2533445
##    49  0.4435123  0.2550757
##    51  0.4418622  0.2526841
##    53  0.4430643  0.2543205
##    55  0.4429139  0.2540701
##    57  0.4451623  0.2570583
##    59  0.4468138  0.2593029
##    61  0.4457634  0.2578076
##    63  0.4453134  0.2572632
##    65  0.4460644  0.2581716
##    67  0.4459131  0.2579924
##    69  0.4502636  0.2637778
##    71  0.4472642  0.2597490
##    73  0.4460646  0.2581522
##    75  0.4460630  0.2581127
##    77  0.4450144  0.2565884
##    79  0.4453127  0.2569816
##    81  0.4457641  0.2575397
##    83  0.4442644  0.2555583
##    85  0.4457634  0.2576333
##    87  0.4462129  0.2582048
##    89  0.4457636  0.2576397
##    91  0.4468147  0.2590224
##    93  0.4466641  0.2588592
##    95  0.4436635  0.2548757
##    97  0.4472647  0.2597095
##    99  0.4445660  0.2561623
##   101  0.4478664  0.2605754
##   103  0.4492173  0.2623792
##   105  0.4489159  0.2619709
##   107  0.4498145  0.2631752
##   109  0.4493643  0.2625749
##   111  0.4502647  0.2637210
##   113  0.4505646  0.2641212
##   115  0.4496639  0.2629309
##   117  0.4502629  0.2636678
##   119  0.4492139  0.2622584
##   121  0.4492144  0.2622508
##   123  0.4514653  0.2652959
##   125  0.4501148  0.2634681
##   127  0.4523653  0.2664563
##   129  0.4508656  0.2643161
##   131  0.4495156  0.2625419
##   133  0.4493643  0.2623087
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 127.
```

```
plot(model)
```



```
error1 <- numeric()
for(i in 5:140){
  pred <- class:::knn(as.data.frame(scale(train.set[c(3,6,8)])),
             as.data.frame(scale(test[c(4,7,9)])),
             as.matrix(train.set[,10]), k = i)
  tab<- table(pred, as.matrix(test[,10]))
  error1[i] <- (1- sum(diag(tab))/sum(tab))
}
plot(5:140,error1[5:140])
```



Third KNN model with Dummy Varible(As it computation cost of time is too long, we do not choose to knit it)

```
nrow(train)
train.set <- train[,-1]
test.set <- test[,-1]
#Convert Categorical varibale into dummy variable
train.setnew <- dummy_cols(train.set, select_columns = c("Gender","Ever_Married","Graduated","Professio
train.setnew[,c("Age","Family_Size")]<-scale(train.setnew[,c("Age","Family_Size")])
test.setnew <-dummy_cols(test.set, select_columns = c("Gender","Ever_Married","Graduated","Profession",
test.setnew[,c("Age","Family_Size")]<-scale(test.setnew[,c("Age","Family_Size")])

error <- numeric()
for(i in 10:200){
pred1 <- neighbr:::knn(train_set=train.setnew[1:1000,-c(1:3)],test_set=test.setnew[1:300,-c(1:4)],k=i,ca
pred1result <- pred1[["test_set_scores"]][["categorical_target"]]
tab<- table(pred1result, as.matrix(test.setnew[1:300,4]))
error[i] <- (1- sum(diag(tab))/sum(tab))
}
plot(10:200,error[10:200])
```

Convert categorical columns to numeric:

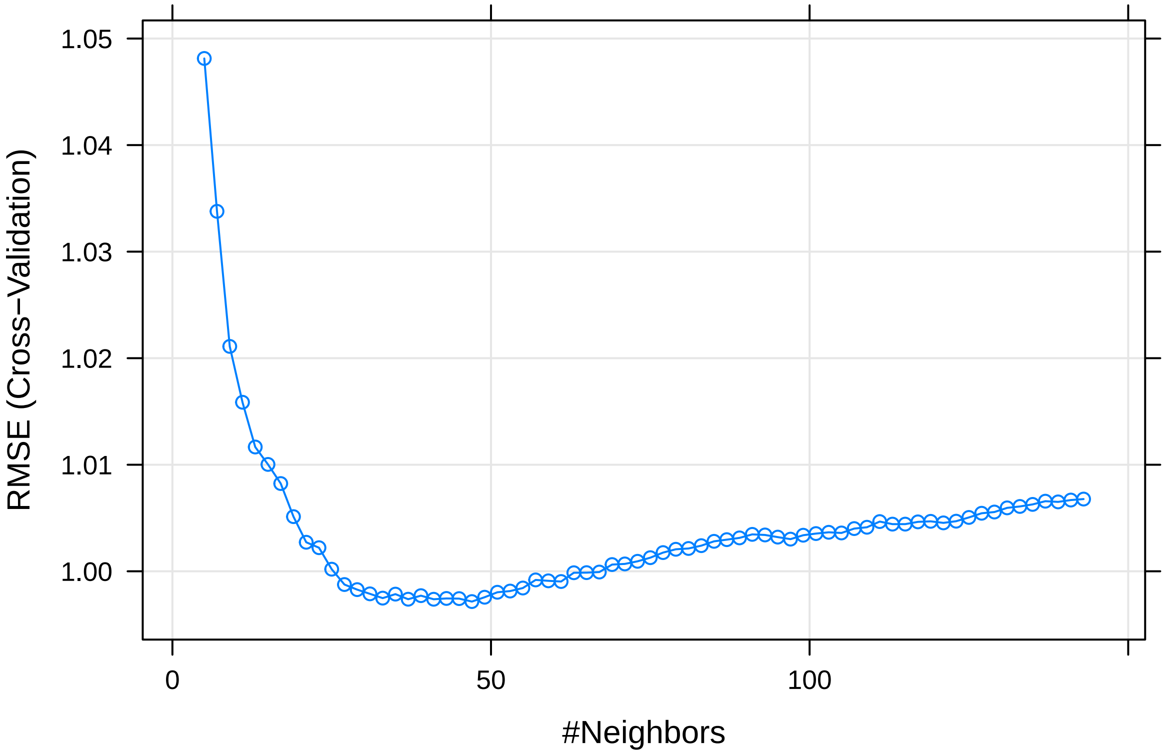
```
temp <- train
must_convert<-sapply(temp,is.character)
temp2<-sapply(temp[,must_convert],function(x){unclass(as.factor(x))})
train.num<-cbind(temp[,!must_convert],temp2)


temp <- test
must_convert<-sapply(temp,is.character)
temp2<-sapply(temp[,must_convert],function(x){unclass(as.factor(x))})
test.num<-cbind(temp[,!must_convert],temp2)
```

KNN second model

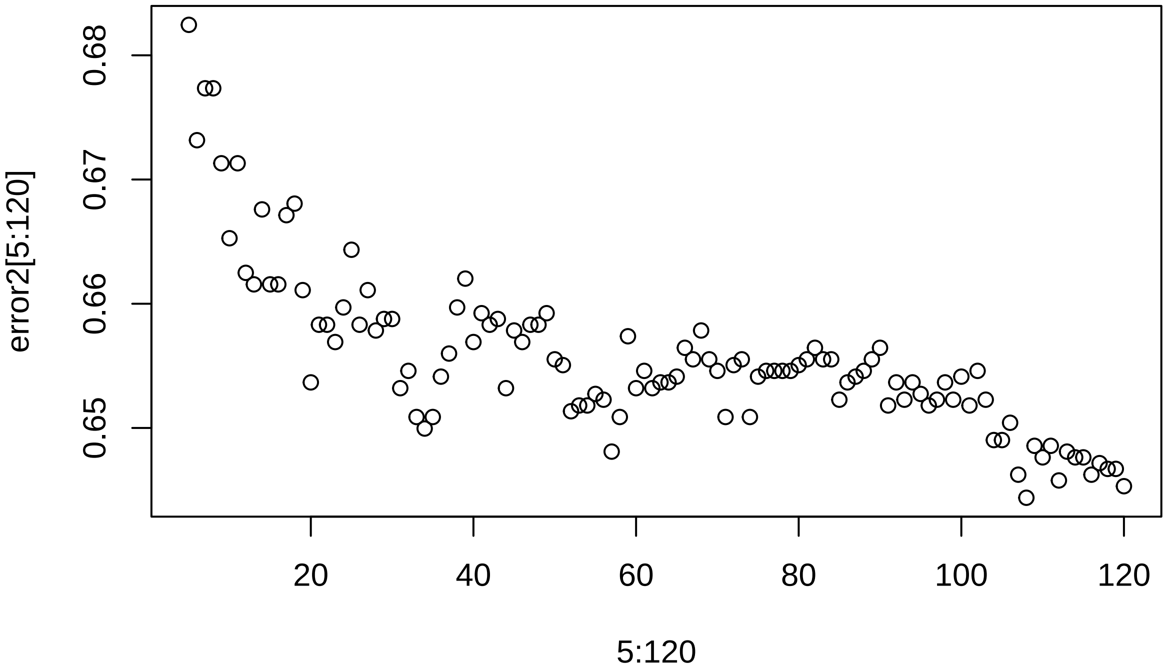
```
set.seed(123)
control <- trainControl(method = ’cv’,number = 10,)
model2 <- train(Segmentation~.,train.num[,c(2:11)],
               method = ’knn’,
               trControl = control,tuneLength=70)
model2
```

```
## k-Nearest Neighbors
##
## 6665 samples
##    9 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5998, 5999, 5998, 5999, 5999, 5998, ...
## Resampling results across tuning parameters:
##
##   k    RMSE       Rsquared   MAE
##     5  1.0481336  0.1665634  0.8166304
##     7  1.0337842  0.1762270  0.8131512
##     9  1.0211055  0.1880340  0.8091198
##    11  1.0158628  0.1924518  0.8099176
##    13  1.0116639  0.1965879  0.8101013
##    15  1.0100283  0.1977287  0.8108885
##    17  1.0082344  0.1993456  0.8114001
##    19  1.0051292  0.2031531  0.8103063
##    21  1.0027243  0.2062889  0.8089257
##    23  1.0022127  0.2066451  0.8098545
##    25  1.0001966  0.2092712  0.8095640
##    27  0.9987559  0.2112101  0.8097898
##    29  0.9982782  0.2116849  0.8097119
##    31  0.9978841  0.2119801  0.8105058
##    33  0.9974799  0.2124700  0.8113297
##    35  0.9978557  0.2118079  0.8121565
##    37  0.9973762  0.2123821  0.8123400
##    39  0.9977117  0.2117434  0.8135274
##    41  0.9973742  0.2121801  0.8143498
##    43  0.9974504  0.2119375  0.8152208
##    45  0.9974348  0.2118649  0.8158460
##    47  0.9971574  0.2123035  0.8159947
##    49  0.9975652  0.2116346  0.8168275
##    51  0.9980311  0.2108923  0.8176461
##    53  0.9981430  0.2106724  0.8182342
##    55  0.9984278  0.2101852  0.8189695
##    57  0.9991897  0.2089695  0.8199925
##    59  0.9991051  0.2090539  0.8202486
##    61  0.9990382  0.2091200  0.8207094
##    63  0.9998631  0.2078401  0.8217672
##    65  0.9998729  0.2077828  0.8224014
##    67  0.9999272  0.2076819  0.8227343
##    69  1.0006301  0.2065560  0.8239326
##    71  1.0006900  0.2064228  0.8243838
##    73  1.0009367  0.2059997  0.8252408
##    75  1.0012779  0.2054068  0.8258449
##    77  1.0017541  0.2046258  0.8266452
##    79  1.0020655  0.2041280  0.8273514
##    81  1.0021406  0.2039960  0.8277855
##    83  1.0024051  0.2035584  0.8283847
##    85  1.0028112  0.2028996  0.8288981
##    87  1.0029680  0.2026216  0.8295441
##    89  1.0031344  0.2023529  0.8300656
##    91  1.0034632  0.2018170  0.8306467
##    93  1.0034089  0.2018998  0.8309411
##    95  1.0032074  0.2022168  0.8311022
##    97  1.0030159  0.2025313  0.8311811
##    99  1.0033846  0.2019361  0.8317079
##   101  1.0035383  0.2016634  0.8322125
##   103  1.0036589  0.2015101  0.8324826
##   105  1.0035957  0.2015819  0.8328399
##   107  1.0040085  0.2009265  0.8335584
##   109  1.0041318  0.2007399  0.8340048
##   111  1.0046653  0.1998900  0.8344960
##   113  1.0044259  0.2002342  0.8345074
##   115  1.0044251  0.2002392  0.8348515
##   117  1.0046379  0.1998911  0.8351727
##   119  1.0046882  0.1997979  0.8354208
##   121  1.0045428  0.2000175  0.8356005
##   123  1.0047006  0.1997697  0.8359042
##   125  1.0050548  0.1991931  0.8365605
##   127  1.0054456  0.1985585  0.8369985
##   129  1.0055606  0.1983655  0.8373271
##   131  1.0059533  0.1977376  0.8378330
##   133  1.0060846  0.1975109  0.8380189
##   135  1.0062779  0.1971884  0.8383435
##   137  1.0065766  0.1967027  0.8387354
##   139  1.0065129  0.1967867  0.8389980
##   141  1.0066852  0.1964948  0.8392961
##   143  1.0067792  0.1963423  0.8396225
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 47.
```

```
plot(model2,)
```



```
error2 <- numeric()
for(i in 5:120){
  pred <- class:::knn(as.data.frame(train.num[c(2:10)]),
               as.data.frame(test.num[c(2:10)]),
               as.matrix(train.num[,11]), k = i)
  tab<- table(pred, as.matrix(test.num[,11]))
  error2[i] <- (1- sum(diag(tab))/sum(tab))
}
plot(5:120, error2[5:120])
```


error2[5:120]
0.65
0.66
0.67
0.68
20
40
60
80
100
120
5:120

LDA:

```
library(MASS)

lda_fit <- lda(Segmentation ~ ., data=train.set)
lda_fit
```

```
## Call:
## lda(Segmentation ~ ., data = train.set)
##
## Prior probabilities of groups:
##         A         B         C         D
## 0.2424606 0.2358590 0.2580645 0.2636159
##
## Group means:
##   GenderMale Ever_MarriedYes      Age GraduatedYes ProfessionDoctor
## A  0.5420792       0.5816832 44.42141    0.6342822       0.10396040
## B  0.5343511       0.7334606 48.11323    0.7321883       0.08078880
## C  0.5377907       0.8034884 49.32791    0.8337209       0.07151163
## D  0.5896414       0.2669323 32.95674    0.3636881       0.09903244
##   ProfessionEngineer ProfessionEntertainment ProfessionExecutive
## A         0.13675743              0.19801980          0.06064356
## B         0.10114504              0.12150127          0.09478372
## C         0.03720930              0.07267442          0.09534884
## D         0.07854297              0.09846329          0.05350028
##   ProfessionHealthcare ProfessionHomemaker ProfessionLawyer ProfessionMarketing
## A           0.05445545          0.02970297       0.09715347          0.02846535
## B           0.05407125          0.02798982       0.08460560          0.01526718
## C           0.07209302          0.01104651       0.07209302          0.01686047
## D           0.44393853          0.03642573       0.04894707          0.07626636
##   Work_Experience Spending_ScoreHigh Spending_ScoreLow Family_Size Var_1Cat_2
## A        2.888614         0.13242574         0.7004950    2.431931 0.04393564
## B        2.393766         0.20038168         0.4821883    2.682570 0.05661578
## C        2.224419         0.21511628         0.3226744    2.961628 0.05000000
## D        2.997154         0.05976096         0.8844622    3.241320 0.06602163
##   Var_1Cat_3 Var_1Cat_4  Var_1Cat_5 Var_1Cat_6 Var_1Cat_7
## A 0.10829208 0.15965347 0.009900990  0.6373762 0.02599010
## B 0.09541985 0.11959288 0.012722646  0.6787532 0.02353690
## C 0.06860465 0.05348837 0.009883721  0.7813953 0.02325581
## D 0.10870803 0.17700626 0.011952191  0.5890723 0.02675014
##
## Coefficients of linear discriminants:
##                                 LD1         LD2          LD3
## GenderMale              -0.27567501 -0.06744037 -0.055580025
## Ever_MarriedYes          0.21238480 -0.36086140  0.606531538
## Age                      0.02234178  0.00696333  0.023496705
## GraduatedYes             0.68117674  0.08652796 -0.308106238
## ProfessionDoctor        -0.70548986 -0.25836226 -0.005320386
## ProfessionEngineer      -0.86792383 -1.17444903  0.968693436
## ProfessionEntertainment -0.76830638 -1.17992758 -0.545072715
## ProfessionExecutive     -0.67735270 -0.05011896  0.877298992
## ProfessionHealthcare    -1.73700523  1.50622088  1.020471421
## ProfessionHomemaker     -1.13915853 -0.23702131  2.355337729
## ProfessionLawyer        -1.38220299 -0.59633827 -1.082904651
## ProfessionMarketing     -1.58508181  0.78521276 -0.549710604
## Work_Experience         -0.03096318 -0.01864889 -0.069340101
## Spending_ScoreHigh      -0.17896060 -0.71387570  0.819344826
## Spending_ScoreLow       -0.96433284 -1.32728372  0.384619439
## Family_Size              0.05755677  0.25138063 -0.064077802
## Var_1Cat_2               0.31634364 -0.10532162  1.461754618
## Var_1Cat_3               0.18130552 -0.28510612  0.669520252
## Var_1Cat_4              -0.10071548 -0.65989092  0.793716770
## Var_1Cat_5               0.38267613 -0.30263089  1.756152842
## Var_1Cat_6               0.24621495  0.12924502 -0.181461124
## Var_1Cat_7               0.20777755 -0.26907176  0.064541355
##
## Proportion of trace:
##    LD1    LD2    LD3
## 0.8260 0.1687 0.0053
```

```
lda_conf_mat <- table(predict(lda_fit, newdata = test)$class, test$Segmentation)
lda_conf_mat
```

```
##
##       A   B   C   D
##   A 250 165 119 179
##   B  98  68  73  67
##   C 183 135 150 147
##   D 161  82  39 238
```

```
#misclassification error rate
1-sum(diag(lda_conf_mat)) / sum(lda_conf_mat)
```

```
## [1] 0.6722377
```