Libraries

```
library(tidyverse)
# library(psych)          # describe()
library(DataExplorer)     # plot_missing() | drop_columns()
library(caret)            # nearZeroVar() | knnreg()
# library(inspectdf)      # inspect_cat() | show_plots()
# library(ggstance)       # geom_boxploth()
# library(corrplot)       # corrplot() | cor()
# library(ggpubr)         # ggscatter()
library(MASS)             # stepAIC()
library(regclass)         # vif()
# library(leaps)          # regsubsets()
library(ggplot2)          # ggplot()
library(glmtoolbox)       # hltest()
# library(purrr)          # map()
library(GGally)           # ggcorr() | ggpairs()
library(lindia)           # gg_cooksd() | gg_scalelocation
library(gridExtra)        # grid.arrange
# library(FNN)            # knn.reg()
# library(Metrics)        # mse()
library(glmnet)           # cv.glmnet()
library(ROCR)             # prediction() | performance()
library(stats)            # logLik()
library(MLmetrics)        # LogLoss()
################CLUSTERS#################
library(mvtnorm)
library(RColorBrewer)
library(pheatmap)
library(cluster)

library(jtools)           # interact_plot()
library(broom)            # augment()
```

Import Data

```
getwd()
```

```
## [1] "C:/Users/dnguy/Desktop/2 Applied Stats/Project 2/Statistcs2-project-2"
```

```
df = read.csv("Bank_Personal_Loan_Modelling.csv")
```

EDA

```
str(df)
```

```
## 'data.frame':    5000 obs. of  14 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
##  $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
##  $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
##  $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

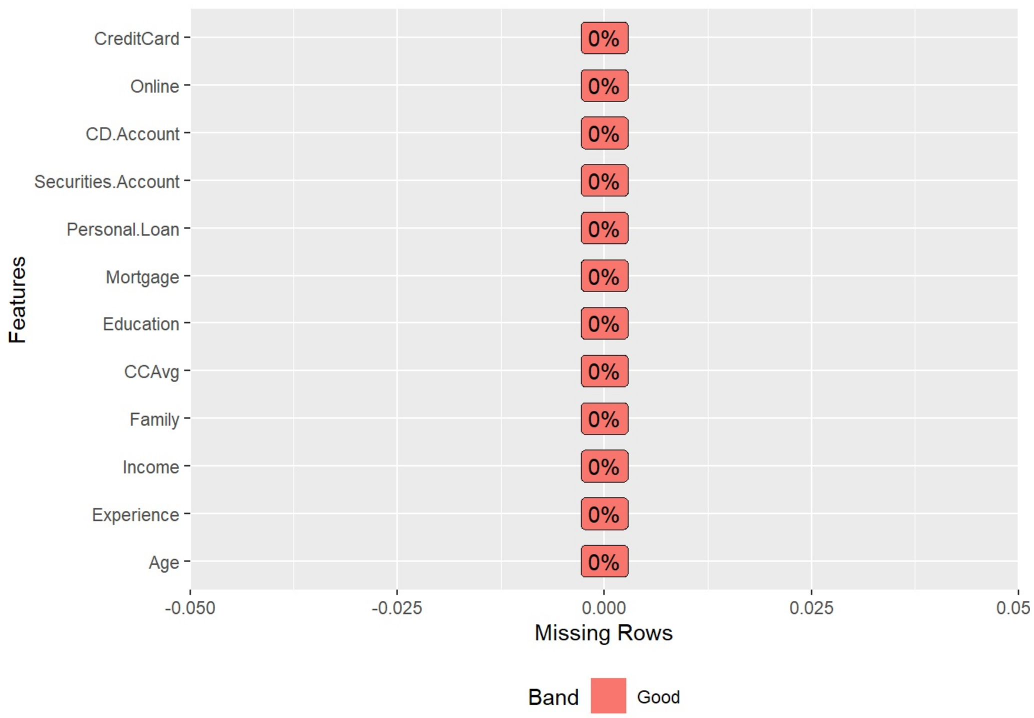
```
# Identification Columns (ID and ZIP.Code)
df = df[-c(1,5)]
str(df)
```

```
## 'data.frame':	5000 obs. of  12 variables:
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
##  $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
##  $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

```
# Naturally Factor Variables
factor_vars = c("Family", "Education", "Personal.Loan",
                "Securities.Account", "CD.Account", "Online", "CreditCard")
df[factor_vars] = lapply(df[factor_vars], as.factor)
str(df)
```

```
## 'data.frame':	5000 obs. of  12 variables:
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ Family            : Factor w/ 4 levels "1","2","3","4": 4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
##  $ CD.Account        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Online            : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 1 2 1 ...
##  $ CreditCard        : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
```

```
# missing values
plot_missing(df)
```




```
# near zero variance
nearZeroVar(df, names = TRUE)
```

```
## [1] "Mortgage"
```

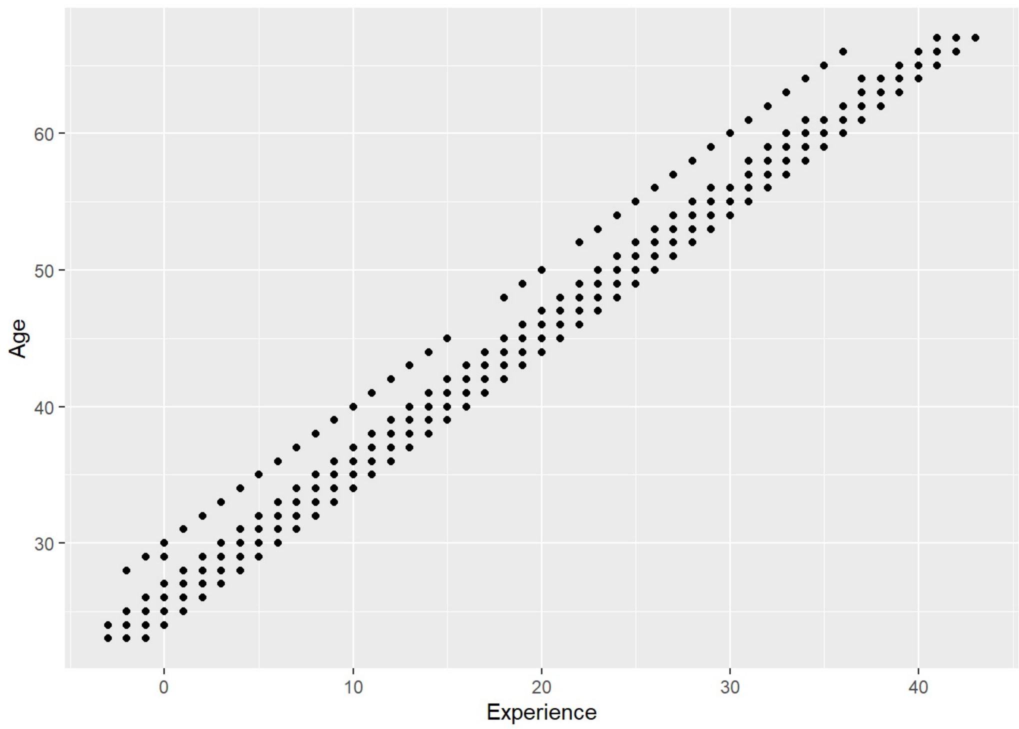
```
#df = df[-c(nearZeroVar(df))] # Removed Mortgage
str(df)
```

```
## 'data.frame':    5000 obs. of  12 variables:
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ Family            : Factor w/ 4 levels "1","2","3","4": 4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
##  $ CD.Account        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Online            : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 1 2 1 ...
##  $ CreditCard        : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
```

```
# multicollinearity
ggcorr(df, label = T)
```

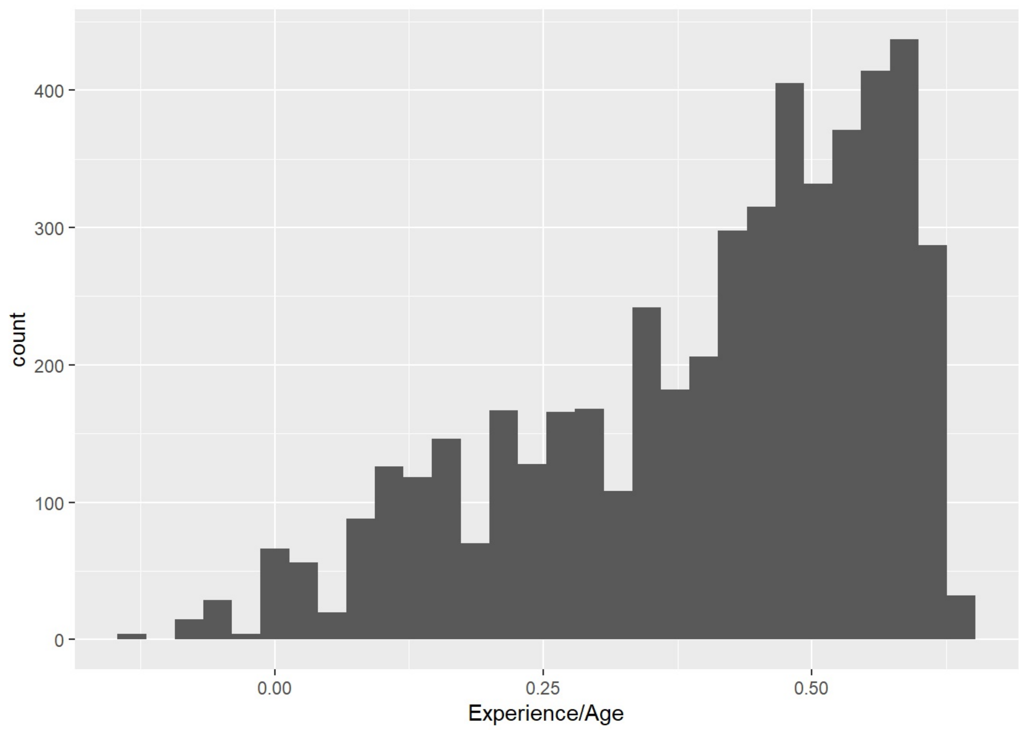
```
## Warning in ggcorr(df, label = T): data in column(s) 'Family', 'Education',
## 'Personal.Loan', 'Securities.Account', 'CD.Account', 'Online', 'CreditCard' are
## not numeric and were ignored
```


```
## Age and Experience have correlation of 1
ggplot( df, aes(Experience, Age)) + geom_point()
```




```
ggplot(df, aes(Experience/Age)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

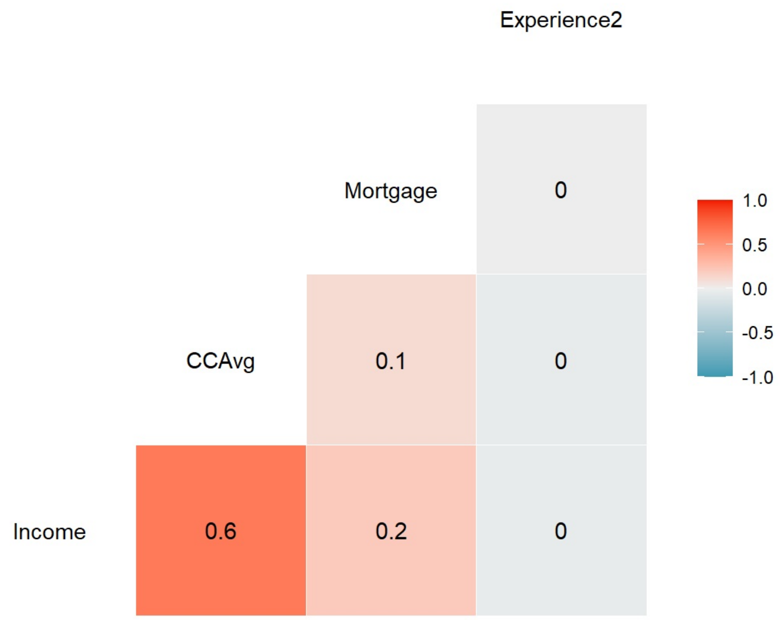



```
df = df %>% mutate(Experience2 = Experience/Age)
df = df[-c(1,2)] # getting rid of Age and Experience
str(df)
```

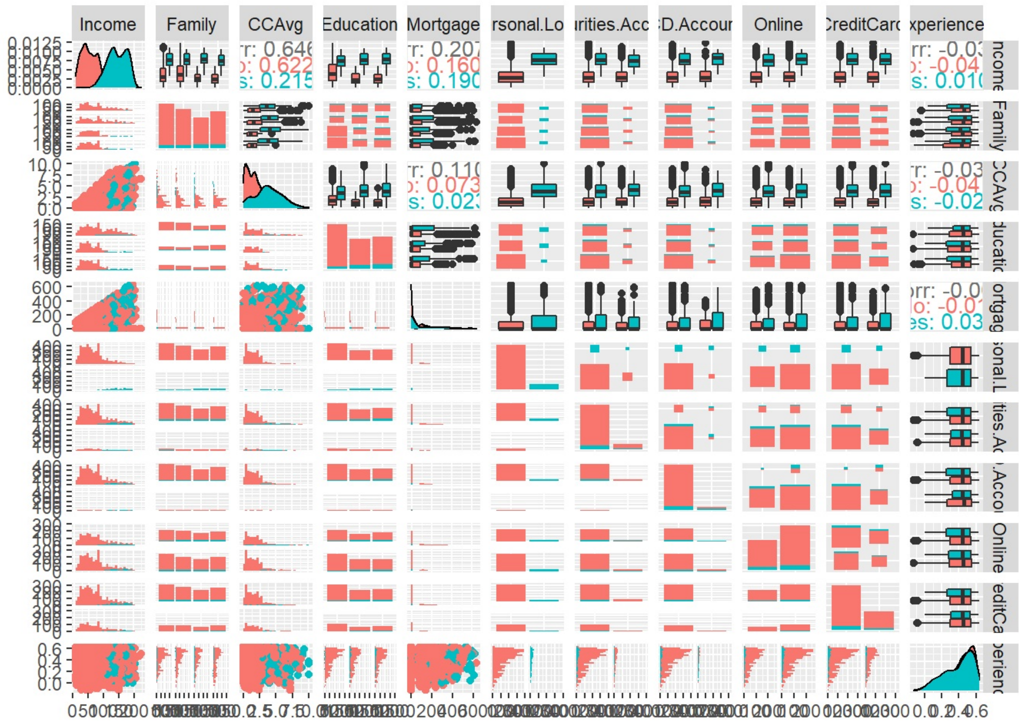
```
## 'data.frame':    5000 obs. of  11 variables:
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ Family            : Factor w/ 4 levels "1","2","3","4": 4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
##  $ CD.Account        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Online            : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 1 2 1 ...
##  $ CreditCard        : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
##  $ Experience2       : num  0.04 0.422 0.385 0.257 0.229 ...
```

```
ggcorr(df, label = T)
```

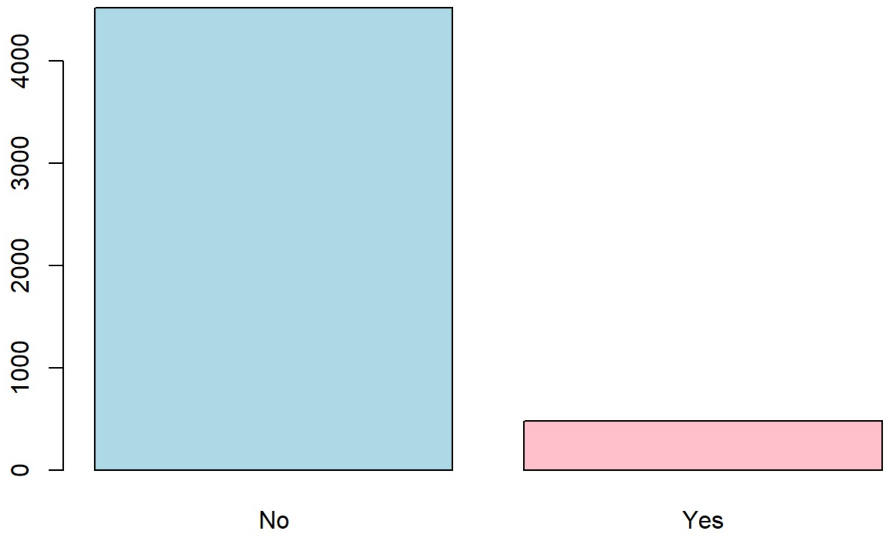
```
## Warning in ggcorr(df, label = T): data in column(s) 'Family', 'Education',
## 'Personal.Loan', 'Securities.Account', 'CD.Account', 'Online', 'CreditCard' are
## not numeric and were ignored
```



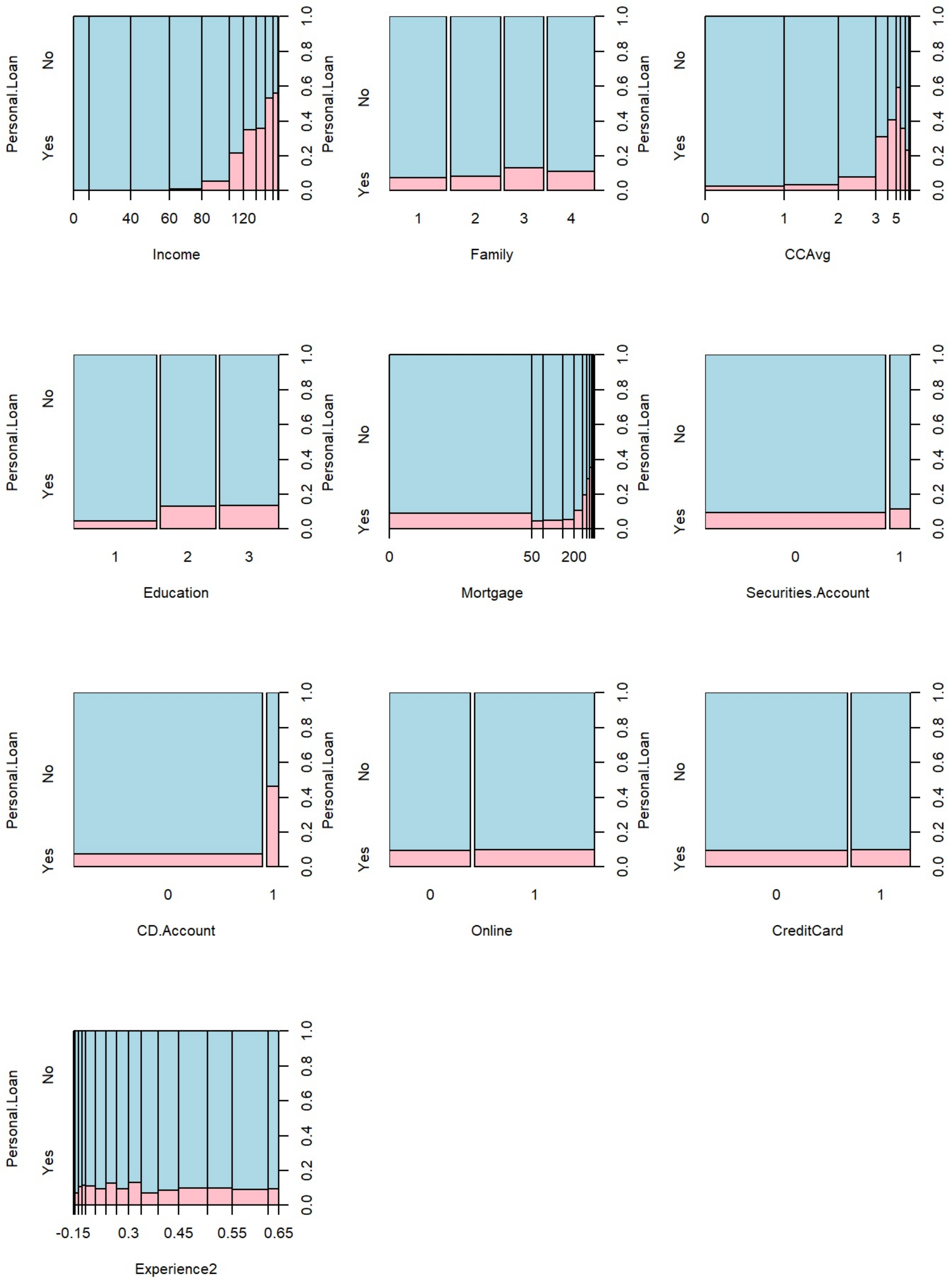

```
# pairs plots
#newAuto$mpg<-factor(ifelse(Auto$mpg>median(Auto$mpg),"High","Low"),levels=c("Low","High"))   # used for numeric
outcome into categorical outcome (using median)
              # kept for future reference
levels(df$Personal.Loan) = c("No", "Yes")
ggpairs(df, aes(colour = Personal.Loan))
```



```
## Using the trick of already knowing what my stepwise logistic regression model consists of in terms of coeffici
ents (which are Income, Family, CCAvg, Education, Securities.Account, CD.Account, Online, and CreditCard) I can p
retend to say that the following variables can be considered in our model for Objective 1 to predict whether if a
customer will accept a personal loan offer or not.

## We can see that, for variables with multiple levels, the levels with even a slight change compared to the refe
rence level (1st level) are found as significant to our stepwise logistic regression model.

# This determines green is yes.
plot(df$Personal.Loan, col= c("lightblue","pink"))
```




```
par(mfrow=c(2,3))
plot(Personal.Loan ~ ., data = df, col= c("pink","lightblue"))
```



EDA: Exploring Interactions

```
#interact_plot()
```

EDA: Heatmaps (Unit 13)

EDA: Cluster Analysis (Unit 13)

Train Test Split

```
set.seed(123)

split = sample(nrow(df), nrow(df)*0.7)

train = df[split,]
test = df[-split,]
```

Objective 1: Logistic Regression Model

```r
premodel = glm(Personal.Loan ~ ., data = train, family = "binomial")

# feature selection - stepwise
stepAIC(premodel, direction = "both")
```

```
## Start:  AIC=820.41
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard + Experience2
## 
##                      Df Deviance     AIC
## - Experience2         1   792.42  818.42
## - Mortgage            1   794.06  820.06
## <none>                    792.41  820.41
## - Securities.Account  1   797.09  823.09
## - CCAvg               1   798.20  824.20
## - CreditCard          1   805.06  831.06
## - Online              1   809.23  835.23
## - Family              3   865.63  887.63
## - CD.Account          1   879.79  905.79
## - Education           2  1083.03 1107.03
## - Income              1  1358.72 1384.72
## 
## Step:  AIC=818.42
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard
## 
##                      Df Deviance     AIC
## - Mortgage            1   794.07  818.07
## <none>                    792.42  818.42
## + Experience2         1   792.41  820.41
## - Securities.Account  1   797.10  821.10
## - CCAvg               1   798.32  822.32
## - CreditCard          1   805.07  829.07
## - Online              1   809.25  833.25
## - Family              3   865.64  885.64
## - CD.Account          1   879.96  903.96
## - Education           2  1083.11 1105.11
## - Income              1  1358.75 1382.75
## 
## Step:  AIC=818.07
## Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account + 
##     CD.Account + Online + CreditCard
## 
##                      Df Deviance     AIC
## <none>                    794.07  818.07
## + Mortgage            1   792.42  818.42
## + Experience2         1   794.06  820.06
## - Securities.Account  1   798.62  820.62
## - CCAvg               1   799.43  821.43
## - CreditCard          1   807.04  829.04
## - Online              1   810.78  832.78
## - Family              3   867.01  885.01
## - CD.Account          1   882.58  904.58
## - Education           2  1083.88 1103.88
## - Income              1  1387.09 1409.09
```

```
## 
## Call:  glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education + 
##     Securities.Account + CD.Account + Online + CreditCard, family = "binomial", 
##     data = train)
## 
## Coefficients:
##        (Intercept)              Income             Family2  
##          -12.54761             0.06381            -0.20616  
##            Family3             Family4               CCAvg  
##            1.92186             1.39674             0.12679  
##         Education2          Education3  Securities.Account1  
##            4.01753             4.17725            -0.71361  
##        CD.Account1             Online1          CreditCard1  
##            3.54524            -0.81711            -0.88671  
## 
## Degrees of Freedom: 3499 Total (i.e. Null);  3488 Residual
## Null Deviance:       2164 
## Residual Deviance: 794.1     AIC: 818.1
```

```
model1 = glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
    Securities.Account + CD.Account + Online + CreditCard, family = "binomial",
    data = train)
```

Hypothesis Testing

```
summary(model1)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
##     Securities.Account + CD.Account + Online + CreditCard, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9123  -0.1813  -0.0649  -0.0194   4.1340
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -12.547610   0.663989 -18.897  < 2e-16 ***
## Income                0.063807   0.003717  17.167  < 2e-16 ***
## Family2              -0.206161   0.281441  -0.733 0.463853
## Family3               1.921856   0.295621   6.501 7.97e-11 ***
## Family4               1.396735   0.290604   4.806 1.54e-06 ***
## CCAvg                 0.126789   0.055110   2.301 0.021412 *
## Education2            4.017534   0.334135  12.024  < 2e-16 ***
## Education3            4.177245   0.333415  12.529  < 2e-16 ***
## Securities.Account1  -0.713607   0.348807  -2.046 0.040771 *
## CD.Account1           3.545241   0.403945   8.777  < 2e-16 ***
## Online1              -0.817114   0.201944  -4.046 5.20e-05 ***
## CreditCard1          -0.886710   0.258336  -3.432 0.000598 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2163.69  on 3499  degrees of freedom
## Residual deviance:  794.07  on 3488  degrees of freedom
## AIC: 818.07
##
## Number of Fisher Scoring iterations: 8
```

```
# As the p-values of all variables used in model1, aside from Family2, are all less than 0.05, none of them are i
nsignificant in our logistic regression model.
```

Criterion

```
AIC(model1)          #  AIC = 818.07
```

```
## [1] 818.0738
```

```
BIC(model1)          #  BIC = 892
```

```
## [1] 892
```

Verify Predictions Manually

```
# Holding the upcoming predictions accountable
prop.table(table(df$Personal.Loan))
```

```
##
##    No   Yes
## 0.904 0.096
```

```
prop.table(table(train$Personal.Loan))
```

```
##
##         No        Yes
## 0.90714286 0.09285714
```

```
prop.table(table(test$Personal.Loan))
```

```
##
##        No       Yes
## 0.8966667 0.1033333
```

```
# This means that,
# it is preferred that our predictions are 90% no loan and 10% yes loan.

# The general idea is, for a bank problem like this where we are trying to find profit from the highest number of
customers who will accept a personal loan offer as we can, we want to have an as-low-as-possible chance of predic
ting customers saying no but they actually do want to say yes because, not calling an interested customer will co
st us valuable profits. However, calling a disinterested customer will not hurt that much where they will simply
assume that it's a cold call. Unless our decisions mean that the bank can forcibly and automatically give a custo
mer a loan despite them not being interested in a loan, or rather a more realistic example like using software to
determine a patient to have cancer even though they do not, and that patient will wastefully go through a surgery
process, like an actionable decision from our predictions, we should be fine with a low specificity (with the oth
er proportion of low specificity meaning a high chance of predicting yes to people saying no, which is perfectly
OK and can be overlooked).

pred.step = predict(model1, test, type = "response")

step.cutoff = 0.426
class.step = as.factor(if_else(pred.step < step.cutoff, "No", "Yes"))
#pred = as.factor(if_else(pred < 0.3, 0, 1))
prop.table(table(class.step))
```

```
## class.step
##    No   Yes
## 0.914 0.086
```

```
# Confusion Matrix
confusionMatrix(class.step, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1334   37
##        Yes   11  118
##
##                Accuracy : 0.968
##                  95% CI : (0.9578, 0.9763)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8135
##
##  Mcnemar's Test P-Value : 0.000308
##
##             Sensitivity : 0.9918
##             Specificity : 0.7613
##          Pos Pred Value : 0.9730
##          Neg Pred Value : 0.9147
##              Prevalence : 0.8967
##          Detection Rate : 0.8893
##    Detection Prevalence : 0.9140
##       Balanced Accuracy : 0.8766
##
##        'Positive' Class : No
##
```

```
                          #  Threshold   = 0.426
                          #  Accuracy    = 0.968
                          #  Sensitivity = 0.7613
                          #  Specificity = 0.9919
```

Assumptions

```
# Linearity
## Predict the probability (p) of personal loan offer
probabilities <- predict(model1, type = "response")
length(probabilities)
```

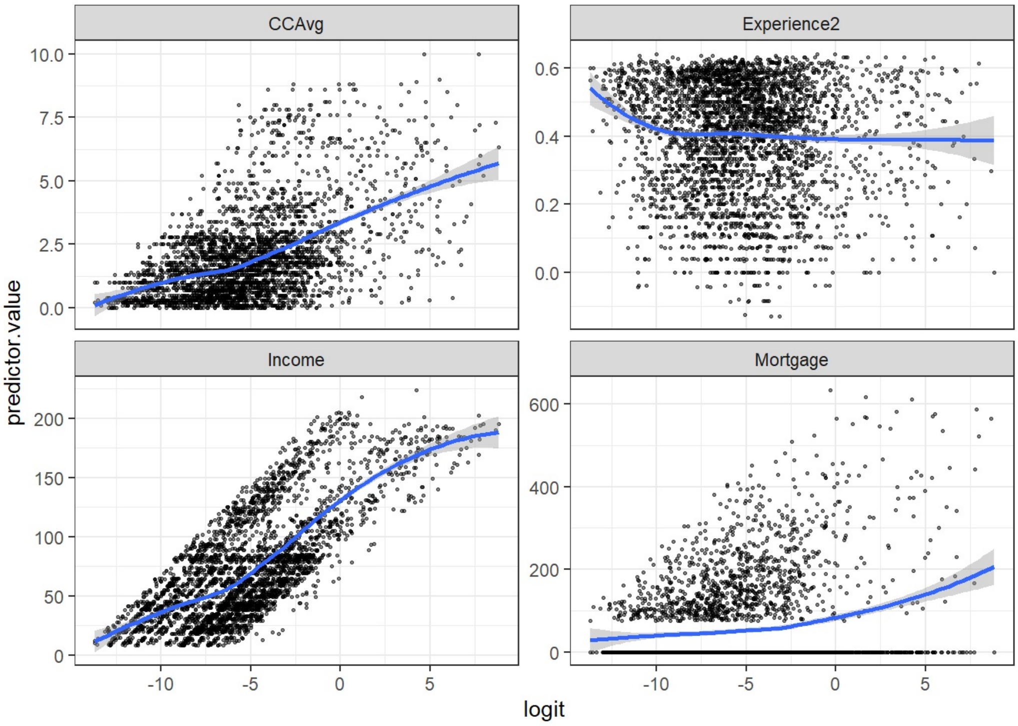
```
## [1] 3500
```

```
step.cutoff = 0.3
predicted.classes <- ifelse(probabilities > step.cutoff, "Yes", "No")
head(predicted.classes)
```

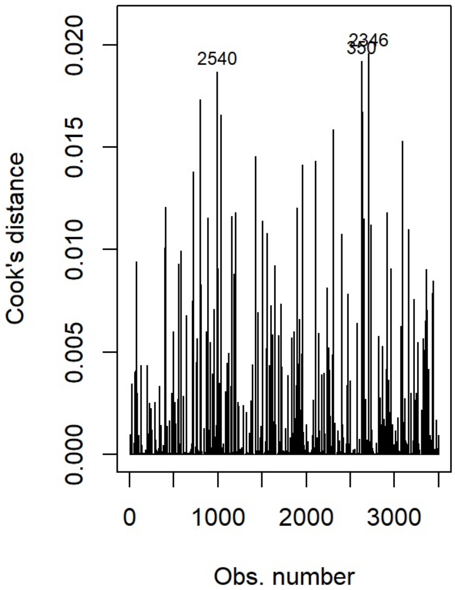
```
## 2463 2511 2227  526 4291 2986
## "No" "No" "No" "No" "No" "No"
```

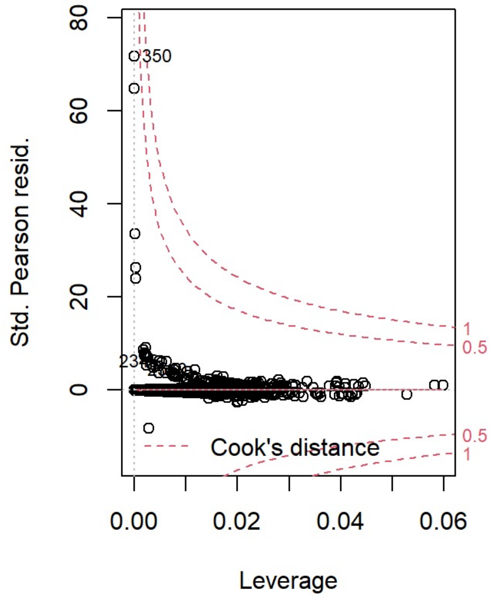
```
## Select only numeric predictors
mydata <- train %>% select_if(is.numeric)
predictors <- colnames(mydata)
## Bind the logit and tidying the data for plot
mydata <- mydata %>%
  mutate(logit = log(probabilities/(1-probabilities))) %>%
  gather(key = "predictors", value = "predictor.value", -logit)
## Create scatter plots
ggplot(mydata, aes(logit, predictor.value))+
  geom_point(size = 0.5, alpha = 0.5) +
  geom_smooth(method = "loess") +
  theme_bw() +
  facet_wrap(~predictors, scales = "free_y")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
# Influential Points
par(mfrow = c(1, 2))
## Cook's Distance Plot
plot(model1, 4, 3)
## Standardized Residuals vs Leverage
plot(model1, 5, 3)
```

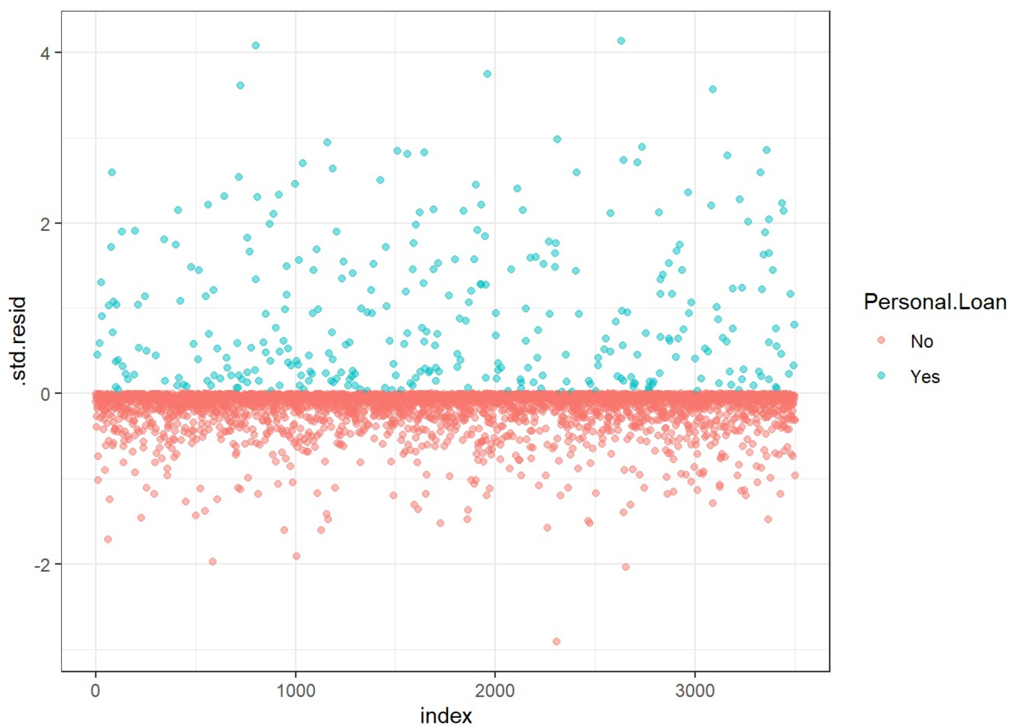





```
par(mfrow = c(1, 1))
## Extract model results
model.data <- augment(model1) %>%
  mutate(index = 1:n())
model.data %>% top_n(3, .cooksd)
```

```
## # A tibble: 3 x 17
##   .rownames Persona~1 Income Family CCAvg Educa~2 Secur~3 CD.Ac~4 Online Credi~5
##   <chr>     <fct>      <int> <fct>  <dbl> <fct>   <fct>   <fct>   <fct>  <fct>
## 1 2540      Yes           98 1        4.2 1       1       1       0      0
## 2 350       Yes           60 2        3   1       0       0       0      0
## 3 2346      Yes           89 1        4.1 1       0       1       1      0
## # ... with 7 more variables: .fitted <dbl>, .resid <dbl>, .std.resid <dbl>,
## #   .hat <dbl>, .sigma <dbl>, .cooksd <dbl>, index <int>, and abbreviated
## #   variable names 1: Personal.Loan, 2: Education, 3: Securities.Account,
## #   4: CD.Account, 5: CreditCard
## # i Use `colnames()` to see all variable names
```

```
ggplot(model.data, aes(index, .std.resid)) +
  geom_point(aes(color = Personal.Loan), alpha = .5) +
  theme_bw()
```



```
## Culprit Outlier Observations
outliers = model.data %>% filter(abs(.std.resid) > 3)
outliers$.rownames
```

```
## [1] "1127" "1070" "976"  "350"  "2159"
```

```
# Multicollinearity
VIF(model1)
```

```
##                         GVIF Df GVIF^(1/(2*Df))
## Income              2.940809  1        1.714879
## Family              1.529409  3        1.073381
## CCAvg               1.516750  1        1.231564
## Education           2.323075  2        1.234570
## Securities.Account 1.291648  1        1.136507
## CD.Account          1.936714  1        1.391659
## Online              1.143566  1        1.069376
## CreditCard          1.383602  1        1.176266
```

Interpretations and Confidence Intervals

```
# Coefficients
coef(model1)
```

```
##         (Intercept)              Income             Family2             Family3
##        -12.54761027          0.06380744         -0.20616070          1.92185637
##             Family4               CCAvg          Education2          Education3
##          1.39673534          0.12678909          4.01753412          4.17724518
## Securities.Account1         CD.Account1             Online1         CreditCard1
##         -0.71360741          3.54524089         -0.81711404         -0.88670963
```

```
# interpret as log odds & confidence intervals
format(exp(cbind("Odds Ratio" = coef(model1),
                 confint.default(model1, level = 0.95))),
       scientific = F)
```

```
##                     Odds Ratio          2.5 %               97.5 %
## (Intercept)         "  0.0000035533836" "  0.0000009670612" "  0.0000130566048"
## Income              "  1.0658871357374" "  1.0581505454691" "  1.0736802915192"
## Family2             "  0.8137023043692" "  0.4687078501608" "  1.4126314289566"
## Family3             "  6.8336324635477" "  3.8284108534361" " 12.1978895250861"
## Family4             "  4.0419826931018" "  2.2868175478207" "  7.1442621677030"
## CCAvg               "  1.1351775688223" "  1.0189519542112" "  1.2646603281260"
## Education2          " 55.5639227354387" " 28.8653266780218" "106.9570264763547"
## Education3          " 65.1860299588602" " 33.9118239040492" "125.3019747277599"
## Securities.Account1 "  0.4898738322890" "  0.2472743352565" "  0.9704863681571"
## CD.Account1         " 34.6480306876299" " 15.6978347366024" " 76.4746253654833"
## Online1             "  0.4417045599866" "  0.2973287142396" "  0.6561859281298"
## CreditCard1         "  0.4120091864564" "  0.2483194731810" "  0.6836015216606"
```

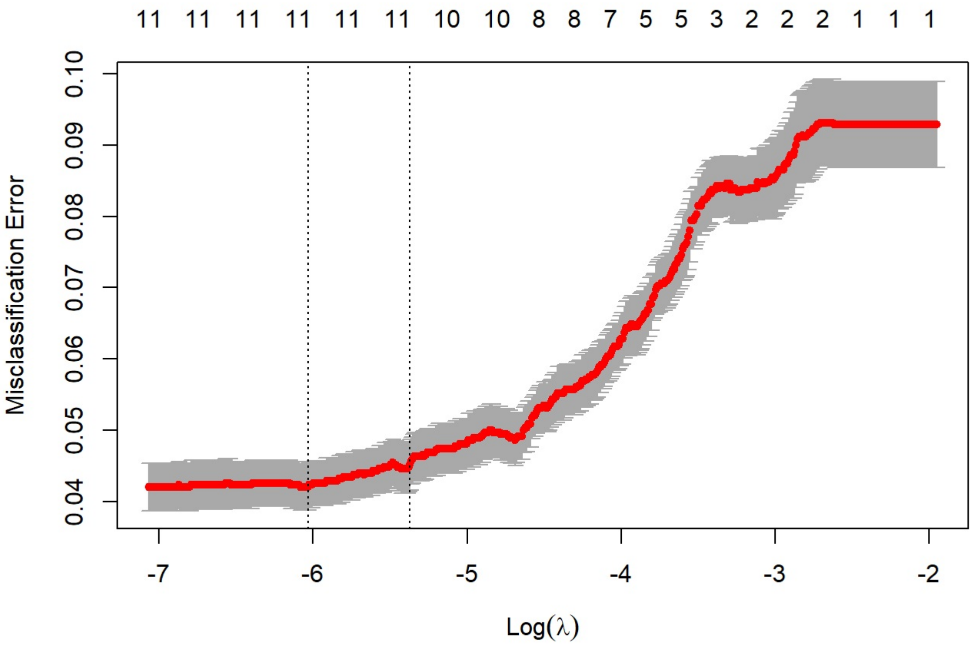
```
# Holding all other variables constant,
### an increase of $1,000 in a customer's income is associated with an increase of 6.58871% in the odds of them a
ccepting a personal loan offer.
### customers with a family size of 2 have around 0.814 times the odds of accepting a personal loan offer than th
ose who don’t.
### ...
### customers with a securities account have a .49 times the odds of those who don't of accepting a personal loan
offer.
### ...
```

Objective 1: LASSO Penalized Logistic Regression Model

```
str(train)
```

```
## 'data.frame':    3500 obs. of  11 variables:
##  $ Income            : int  23 52 98 79 95 63 91 143 59 38 ...
##  $ Family            : Factor w/ 4 levels "1","2","3","4": 3 4 1 2 2 4 1 3 3 1 ...
##  $ CCAvg             : num  0.4 1.3 5.4 2.8 0 3.6 0.1 2.9 0.9 1.5 ...
##  $ Education         : Factor w/ 3 levels "1","2","3": 1 2 1 1 3 3 2 3 3 2 ...
##  $ Mortgage          : int  0 0 0 179 0 0 199 0 199 116 ...
##  $ Personal.Loan     : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 2 1 1 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ CD.Account        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Online            : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 2 1 ...
##  $ CreditCard        : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 2 1 1 2 ...
##  $ Experience2       : num  0.538 0.613 0.04 0.594 0.636 ...
```

```
dat.train.x = model.matrix(Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account + CD.Account
+ Online + CreditCard + Experience2, train)
dat.train.y = train$Personal.Loan

cvfit = cv.glmnet(dat.train.x, dat.train.y, family = "binomial", type.measure = "class", nlambda = 1000)
plot(cvfit)
```




```
coef(cvfit, s = "lambda.min")
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                              s1
## (Intercept)         -10.67329812
## (Intercept)           .
## Income                0.05403179
## Family2              -0.14483106
## Family3               1.52190649
## Family4               1.05043583
## CCAvg                 0.08593637
## Education2            3.15116632
## Education3            3.26676559
## Securities.Account1  -0.29480894
## CD.Account1           2.68760355
## Online1              -0.51980937
## CreditCard1          -0.52133702
## Experience2           .
```

```
# CV misclassification error rate is little below .10
cvfit$cvm[which(cvfit$lambda==cvfit$lambda.min)]
```

```
## [1] 0.042
```

```
# Optimal penalty
cvfit$lambda.min
```

```
## [1] 0.002395327
```

```
# For final model predictions go ahead and refit lasso using entire data set
LASSOmodel = glmnet(dat.train.x, dat.train.y, family = "binomial",lambda=cvfit$lambda.min)
coef(LASSOmodel, s = "lambda.min")
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                               s1
## (Intercept)         -10.67454150
## (Intercept)           .
## Income                0.05403933
## Family2              -0.14480803
## Family3               1.52200412
## Family4               1.05046564
## CCAvg                 0.08592698
## Education2            3.15171466
## Education3            3.26728389
## Securities.Account1  -0.29482677
## CD.Account1           2.68779906
## Online1              -0.51987111
## CreditCard1          -0.52136272
## Experience2           .
```

```
# Predict
dat.test.x = model.matrix(Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account + CD.Account +
Online + CreditCard + Experience2, test)
fit.pred.lasso = predict(LASSOmodel, newx = dat.test.x, type = "response")

LASSO.cutoff = 0.44
class.lasso = as.factor(if_else(fit.pred.lasso < LASSO.cutoff, "No", "Yes"))

# Confusion Matrix for Lasso
conf.lasso = table(class.lasso, test$Personal.Loan)
conf.lasso
```

```
##
## class.lasso   No  Yes
##         No  1340   50
##         Yes    5  105
```

```
# Accuracy of LASSO
sum(diag(conf.lasso))/sum(conf.lasso)
```

```
## [1] 0.9633333
```

```
# Sensitivity & Specificity of LASSO
cm = confusionMatrix(class.lasso, test$Personal.Loan)
cm$byClass
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.9962825            0.6774194            0.9640288
##       Neg Pred Value            Precision               Recall
##            0.9545455            0.9640288            0.9962825
##                   F1           Prevalence       Detection Rate
##            0.9798903            0.8966667            0.8933333
## Detection Prevalence    Balanced Accuracy
##            0.9266667            0.8368509
```

```
                          #  Threshold   = 0.44
                          #  Accuracy    = 0.9633
                          #  Sensitivity = 0.6774
                          #  Specificity = 0.9963
```

Objective 1: Erin’s Model based on Intuition

```
mod.erin = glm(formula = Personal.Loan ~ Income + Family + Education + CD.Account + CreditCard,
           family = "binomial", data = train)
summary(mod.erin)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ Income + Family + Education + CD.Account +
##     CreditCard, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7993  -0.1856  -0.0677  -0.0215   4.2636
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -12.902065   0.650445 -19.836  < 2e-16 ***
## Income        0.066222   0.003506  18.888  < 2e-16 ***
## Family2      -0.160351   0.277510  -0.578  0.56339
## Family3       1.986369   0.289837   6.853 7.21e-12 ***
## Family4       1.405825   0.283468   4.959 7.07e-07 ***
## Education2    3.875382   0.324913  11.927  < 2e-16 ***
## Education3    4.042373   0.322560  12.532  < 2e-16 ***
## CD.Account1   2.829847   0.333789   8.478  < 2e-16 ***
## CreditCard1  -0.703303   0.245175  -2.869  0.00412 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2163.69  on 3499  degrees of freedom
## Residual deviance:  818.88  on 3491  degrees of freedom
## AIC: 836.88
##
## Number of Fisher Scoring iterations: 8
```

```
# Criterion
AIC(mod.erin)             #  AIC = 836.8792
```

```
## [1] 836.8792
```

```
BIC(mod.erin)             #  BIC = 892.3239
```

```
## [1] 892.3239
```

```
pred.erin = predict(mod.erin, test, type = "response")
erin.cutoff = 0.5
class.erin = as.factor(if_else(pred.erin < erin.cutoff, "No", "Yes"))
prop.table(table(class.erin))
```

```
## class.erin
##    No   Yes
## 0.928 0.072
```

```
confusionMatrix(class.erin, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1338   54
##        Yes    7  101
##
##                Accuracy : 0.9593
##                  95% CI : (0.9481, 0.9688)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7466
##
##  Mcnemar's Test P-Value : 3.869e-09
##
##             Sensitivity : 0.9948
##             Specificity : 0.6516
##          Pos Pred Value : 0.9612
##          Neg Pred Value : 0.9352
##              Prevalence : 0.8967
##          Detection Rate : 0.8920
##    Detection Prevalence : 0.9280
##       Balanced Accuracy : 0.8232
##
##        'Positive' Class : No
##
```

```
                          #  Threshold   = 0.5
                          #  Accuracy    = 0.9593
                          #  Sensitivity = 0.6516
                          #  Specificity = 0.9948
```

Objective 1: Origin Model (Income Only)

```
model_income = glm(formula = Personal.Loan ~ Income, family = "binomial", data = train)
summary(model_income)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ Income, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0919  -0.3066  -0.1796  -0.1166   2.7881
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.03738    0.21904  -27.56   <2e-16 ***
## Income       0.03619    0.00163   22.20   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2163.7  on 3499  degrees of freedom
## Residual deviance: 1407.4  on 3498  degrees of freedom
## AIC: 1411.4
##
## Number of Fisher Scoring iterations: 6
```

```
pred.income = predict(model_income, test, type = "response")

income.cutoff = 0.3
class.income = as.factor(if_else(pred.income < income.cutoff, "No", "Yes"))
prop.table(table(class.income))
```

```
## class.income
##        No       Yes
## 0.8873333 0.1126667
```

```
confusionMatrix(class.income, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1258   73
##        Yes   87   82
##
##                Accuracy : 0.8933
##                  95% CI : (0.8766, 0.9085)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : 0.6827
##
##                   Kappa : 0.4465
##
##  Mcnemar's Test P-Value : 0.3041
##
##             Sensitivity : 0.9353
##             Specificity : 0.5290
##          Pos Pred Value : 0.9452
##          Neg Pred Value : 0.4852
##              Prevalence : 0.8967
##          Detection Rate : 0.8387
##    Detection Prevalence : 0.8873
##       Balanced Accuracy : 0.7322
##
##        'Positive' Class : No
##
```

```
                          #  Threshold   = 0.3
                          #  Accuracy    = 0.8933
                          #  Sensitivity = 0.5290
                          #  Specificity = 0.9353
# Criterion
AIC(model_income)          #  AIC = 1411.45
```

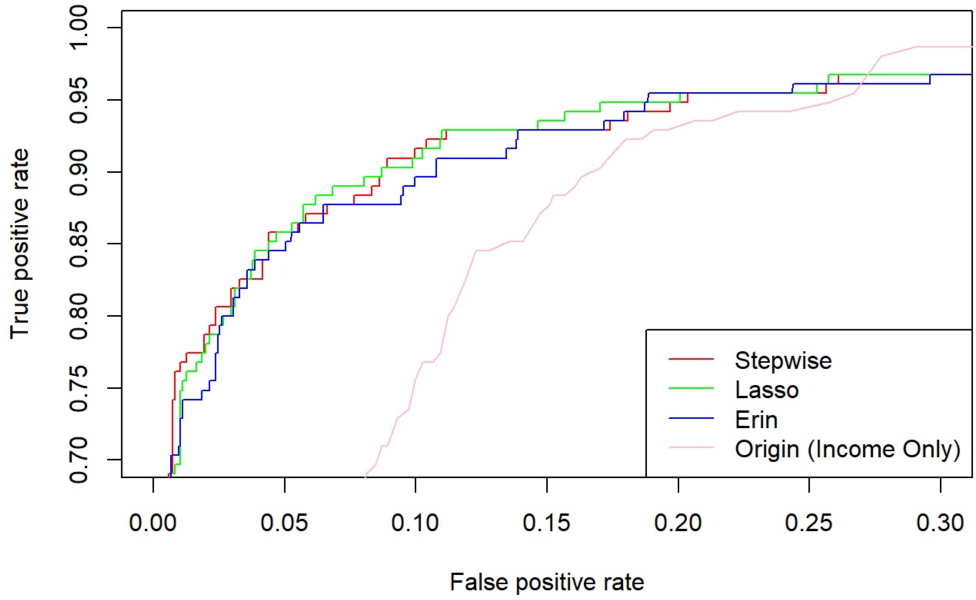
```
## [1] 1411.45
```

```
BIC(model_income)          #  BIC = 1423.771
```

```
## [1] 1423.771
```

Comparing ROCR Curves

```
# Stepwise
pred_prob = predict(model1, test, type = "response")
test_label = df[-split, "Personal.Loan"]
results.step = prediction(pred_prob, test_label)
roc.step = performance(results.step, measure = "tpr", x.measure = "fpr")

# LASSO
results.lasso = prediction(fit.pred.lasso,
                           test$Personal.Loan,
                           label.ordering=c("No", "Yes"))
roc.lasso = performance(results.lasso, measure = "tpr", x.measure = "fpr")

# Erin's Intuition
results.erin = prediction(pred.erin, test_label)
roc.erin = performance(results.erin, measure = "tpr", x.measure = "fpr")

# Origin (Income Only)
results.income = prediction(pred.income, test_label)
roc.income = performance(results.income, measure = "tpr", x.measure = "fpr")

plot(roc.step, col = "red", xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.lasso, col = "green", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.erin, col = "blue", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.income, col = "pink", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
legend("bottomright", legend = c("Stepwise", "Lasso", "Erin", "Origin (Income Only)"),
       col = c("red", "green","blue", "pink"),
       lty=1, lwd=1)
```




```
#abline(a=0, b= 1)
#abline(a=1, b= -1)

# Stepwise seems to be the better performing model according to the above ROC curves.
```

Objective 2: Adding Complexity

```
model.poly.income2 = glm(formula = Personal.Loan ~ poly(Income, 2) + Family + CCAvg + Education + Securities.Acco
unt + CD.Account + Online + CreditCard, family = "binomial", data = train)
pred.poly.income2 = predict(model.poly.income2, test, type = "response")
poly.income2.cutoff = 0.55
class.poly.income2 = as.factor(if_else(pred.poly.income2 < poly.income2.cutoff, "No", "Yes"))
prop.table(table(class.poly.income2))
```

```
## class.poly.income2
##         No        Yes
## 0.91133333 0.08866667
```

```
confusionMatrix(class.poly.income2, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1334   33
##        Yes   11  122
##
##                Accuracy : 0.9707
##                  95% CI : (0.9608, 0.9786)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8311
##
##  Mcnemar's Test P-Value : 0.001546
##
##             Sensitivity : 0.9918
##             Specificity : 0.7871
##          Pos Pred Value : 0.9759
##          Neg Pred Value : 0.9173
##              Prevalence : 0.8967
##          Detection Rate : 0.8893
##    Detection Prevalence : 0.9113
##       Balanced Accuracy : 0.8895
##
##        'Positive' Class : No
##
```

```
results.poly.income2 = prediction(pred.poly.income2, test_label)
roc.poly.income2 = performance(results.poly.income2, measure = "tpr", x.measure = "fpr")

model.poly.income3 = glm(formula = Personal.Loan ~ poly(Income, 3) + Family + CCAvg + Education + Securities.Acco
unt + CD.Account + Online + CreditCard, family = "binomial", data = train)
pred.poly.income3 = predict(model.poly.income3, test, type = "response")
poly.income3.cutoff = 0.55
class.poly.income3 = as.factor(if_else(pred.poly.income3 < poly.income3.cutoff, "No", "Yes"))
prop.table(table(class.poly.income3))
```

```
## class.poly.income3
##         No        Yes
## 0.91066667 0.08933333
```

```
confusionMatrix(class.poly.income3, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1333   33
##        Yes   12  122
##
##                Accuracy : 0.97
##                  95% CI : (0.9601, 0.978)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8278
##
##  Mcnemar's Test P-Value : 0.002869
##
##             Sensitivity : 0.9911
##             Specificity : 0.7871
##          Pos Pred Value : 0.9758
##          Neg Pred Value : 0.9104
##              Prevalence : 0.8967
##          Detection Rate : 0.8887
##    Detection Prevalence : 0.9107
##       Balanced Accuracy : 0.8891
##
##        'Positive' Class : No
##
```

```r
results.poly.income3 = prediction(pred.poly.income3, test_label)
roc.poly.income3 = performance(results.poly.income3, measure = "tpr", x.measure = "fpr")

model.poly.CCAvg2 = glm(formula = Personal.Loan ~ Income + Family + poly(CCAvg, 2) + Education + Securities.Accou
nt + CD.Account + Online + CreditCard, family = "binomial", data = train)
pred.poly.CCAvg2 = predict(model.poly.CCAvg2, test, type = "response")
poly.CCAvg2.cutoff = 0.55
class.poly.CCAvg2 = as.factor(if_else(pred.poly.CCAvg2 < poly.CCAvg2.cutoff, "No", "Yes"))
prop.table(table(class.poly.CCAvg2))
```

```
## class.poly.CCAvg2
##         No        Yes
## 0.92133333 0.07866667
```

```r
confusionMatrix(class.poly.CCAvg2, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1339   43
##        Yes    6  112
##
##                Accuracy : 0.9673
##                  95% CI : (0.957, 0.9757)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8029
##
##  Mcnemar's Test P-Value : 2.706e-07
##
##             Sensitivity : 0.9955
##             Specificity : 0.7226
##          Pos Pred Value : 0.9689
##          Neg Pred Value : 0.9492
##              Prevalence : 0.8967
##          Detection Rate : 0.8927
##    Detection Prevalence : 0.9213
##       Balanced Accuracy : 0.8591
##
##        'Positive' Class : No
##
```

```r
results.poly.CCAvg2 = prediction(pred.poly.CCAvg2, test_label)
roc.poly.CCAvg2 = performance(results.poly.CCAvg2, measure = "tpr", x.measure = "fpr")

model.poly.CCAvg3 = glm(formula = Personal.Loan ~ Income + Family + poly(CCAvg, 3) + Education + Securities.Accou
nt + CD.Account + Online + CreditCard, family = "binomial", data = train)
pred.poly.CCAvg3 = predict(model.poly.CCAvg3, test, type = "response")
poly.CCAvg3.cutoff = 0.55
class.poly.CCAvg3 = as.factor(if_else(pred.poly.CCAvg3 < poly.CCAvg3.cutoff, "No", "Yes"))
prop.table(table(class.poly.CCAvg3))
```

```
## class.poly.CCAvg3
##         No        Yes
## 0.91933333 0.08066667
```

```r
confusionMatrix(class.poly.CCAvg3, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1337   42
##        Yes    8  113
##
##                Accuracy : 0.9667
##                  95% CI : (0.9563, 0.9752)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8008
##
##  Mcnemar's Test P-Value : 3.058e-06
##
##             Sensitivity : 0.9941
##             Specificity : 0.7290
##          Pos Pred Value : 0.9695
##          Neg Pred Value : 0.9339
##              Prevalence : 0.8967
##          Detection Rate : 0.8913
##    Detection Prevalence : 0.9193
##       Balanced Accuracy : 0.8615
##
##        'Positive' Class : No
##
```

```
results.poly.CCAvg3 = prediction(pred.poly.CCAvg3, test_label)
roc.poly.CCAvg3 = performance(results.poly.CCAvg3, measure = "tpr", x.measure = "fpr")

model.poly.both2 = glm(formula = Personal.Loan ~ poly(Income, 2) + Family + poly(CCAvg, 2) + Education + Securiti
es.Account + CD.Account + Online + CreditCard, family = "binomial", data = train)
pred.poly.both2 = predict(model.poly.both2, test, type = "response")
poly.both2.cutoff = 0.55
class.poly.both2 = as.factor(if_else(pred.poly.both2 < poly.both2.cutoff, "No", "Yes"))
prop.table(table(class.poly.both2))
```

```
## class.poly.both2
##   No  Yes
## 0.91 0.09
```

```
confusionMatrix(class.poly.both2, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1334   31
##        Yes   11  124
##
##                Accuracy : 0.972
##                  95% CI : (0.9623, 0.9797)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8398
##
##  Mcnemar's Test P-Value : 0.00337
##
##             Sensitivity : 0.9918
##             Specificity : 0.8000
##          Pos Pred Value : 0.9773
##          Neg Pred Value : 0.9185
##              Prevalence : 0.8967
##          Detection Rate : 0.8893
##    Detection Prevalence : 0.9100
##       Balanced Accuracy : 0.8959
##
##        'Positive' Class : No
##
```

```
results.poly.both2 = prediction(pred.poly.both2, test_label)
roc.poly.both2 = performance(results.poly.both2, measure = "tpr", x.measure = "fpr")

model.poly.both3 = glm(formula = Personal.Loan ~ poly(Income, 3) + Family + poly(CCAvg, 3) + Education + Securiti
es.Account + CD.Account + Online + CreditCard, family = "binomial", data = train)
pred.poly.both3 = predict(model.poly.both3, test, type = "response")
poly.both3.cutoff = 0.55
class.poly.both3 = as.factor(if_else(pred.poly.both3 < poly.both3.cutoff, "No", "Yes"))
prop.table(table(class.poly.both3))
```

```
## class.poly.both3
##         No        Yes
## 0.91066667 0.08933333
```

```
confusionMatrix(class.poly.both3, test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1334   32
##        Yes   11  123
##
##                Accuracy : 0.9713
##                  95% CI : (0.9616, 0.9792)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8354
##
##  Mcnemar's Test P-Value : 0.002289
##
##             Sensitivity : 0.9918
##             Specificity : 0.7935
##          Pos Pred Value : 0.9766
##          Neg Pred Value : 0.9179
##              Prevalence : 0.8967
##          Detection Rate : 0.8893
##    Detection Prevalence : 0.9107
##       Balanced Accuracy : 0.8927
##
##        'Positive' Class : No
##
```

```
results.poly.both3 = prediction(pred.poly.both3, test_label)
roc.poly.both3 = performance(results.poly.both3, measure = "tpr", x.measure = "fpr")

confusionMatrix(class.poly.income2, test$Personal.Loan)$overall[1]
```

```
##  Accuracy
## 0.9706667
```

```
confusionMatrix(class.poly.income3, test$Personal.Loan)$overall[1]
```

```
## Accuracy
##     0.97
```

```
confusionMatrix(class.poly.CCAvg2, test$Personal.Loan)$overall[1]
```

```
##  Accuracy
## 0.9673333
```

```
confusionMatrix(class.poly.CCAvg3, test$Personal.Loan)$overall[1]
```

```
##  Accuracy
## 0.9666667
```

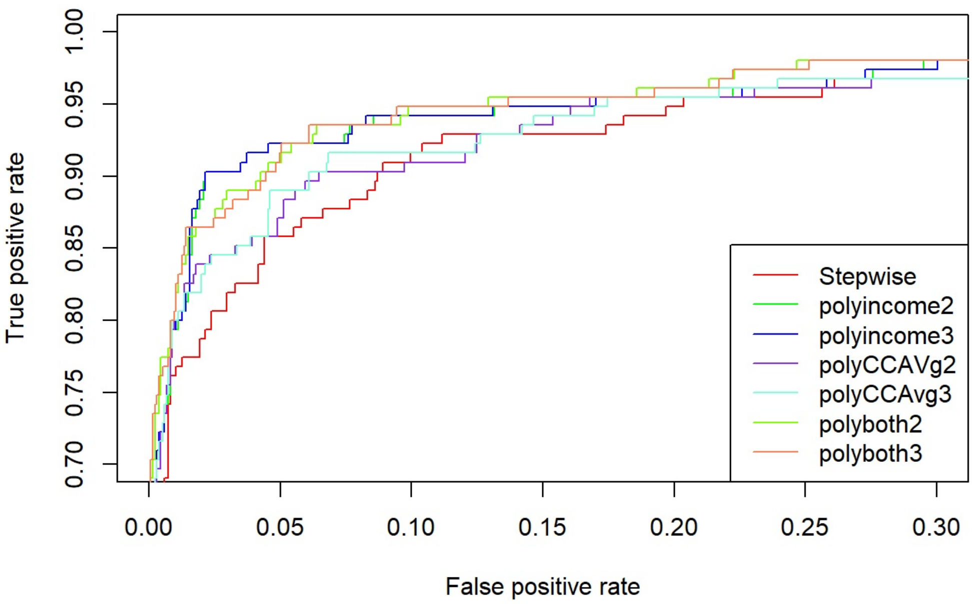
```
confusionMatrix(class.poly.both2, test$Personal.Loan)$overall[1]
```

```
## Accuracy
##    0.972
```

```
confusionMatrix(class.poly.both3, test$Personal.Loan)$overall[1]
```

```
##  Accuracy
## 0.9713333
```

```
plot(roc.step, col = "red", xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.poly.income2, col = "green", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.poly.income3, col = "blue", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.poly.CCAvg2, col = "blueviolet", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.poly.CCAvg3, col = "aquamarine", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.poly.both2, col = "chartreuse", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.poly.both3, col = "coral", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
legend("bottomright", legend = c("Stepwise", "polyincome2", "polyincome3", "polyCCAVg2", "polyCCAvg3", "polyboth2
", "polyboth3"),
       col = c("red", "green", "blue", "blueviolet", "aquamarine", "chartreuse", "coral"),
       lty=1, lwd=1)
```




```
# Polynomial Income^3 seems to be the best performing model according to the above plot.
```

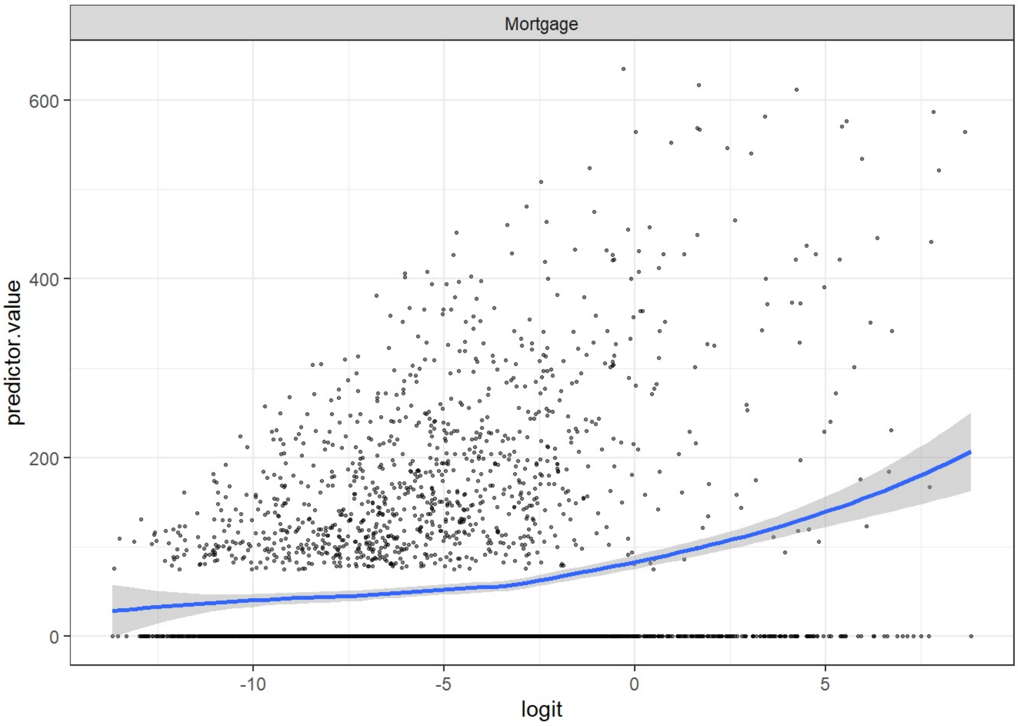
Objective 2: Working Towards A Best Model

```
str(train)
```

```
## 'data.frame':    3500 obs. of  11 variables:
##  $ Income           : int  23 52 98 79 95 63 91 143 59 38 ...
##  $ Family           : Factor w/ 4 levels "1","2","3","4": 3 4 1 2 2 4 1 3 3 1 ...
##  $ CCAvg            : num  0.4 1.3 5.4 2.8 0 3.6 0.1 2.9 0.9 1.5 ...
##  $ Education        : Factor w/ 3 levels "1","2","3": 1 2 1 1 3 3 2 3 3 2 ...
##  $ Mortgage         : int  0 0 0 179 0 0 199 0 199 116 ...
##  $ Personal.Loan    : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 2 1 1 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ CD.Account       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Online           : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 2 1 ...
##  $ CreditCard       : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 2 1 1 2 ...
##  $ Experience2      : num  0.538 0.613 0.04 0.594 0.636 ...
```

```
## Addressing Mortgage
facets = c("Mortgage")
ggplot(mydata[mydata$predictors %in% facets,], aes(logit, predictor.value))+
  geom_point(size = 0.5, alpha = 0.5) +
  geom_smooth(method = "loess") +
  theme_bw() +
  facet_wrap(~predictors, scales = "free_y")
```

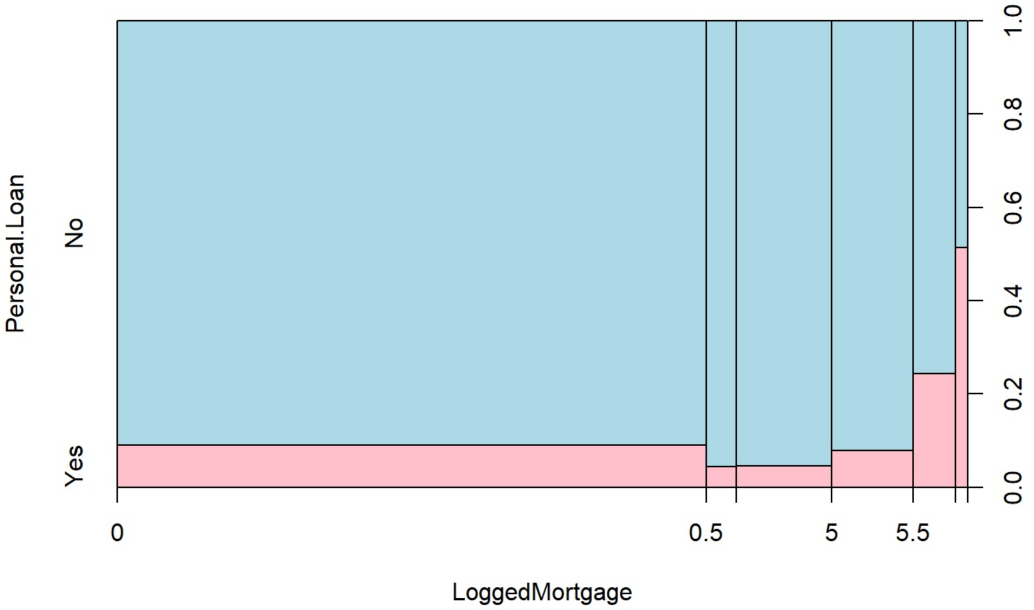
```
## `geom_smooth()` using formula 'y ~ x'
```




```
## Training Stepwise again but with logged Mortgage
str(df$Mortgage)
```

```
##  int [1:5000] 0 0 0 0 0 155 0 0 104 0 ...
```

```
final_df = mutate(df)
## Fixing Mortgage values
final_df$Mortgage[final_df$Mortgage == 0] = 1
final_df$LoggedMortgage = log(final_df$Mortgage)
plot(Personal.Loan ~ LoggedMortgage, data = final_df, col= c("pink","lightblue")) # Mortgage is now somewhat dist
ributed
```




```
set.seed(123)
split = sample(nrow(final_df), nrow(final_df)*0.7)
final_train = final_df[split,]
final_test = final_df[-split,]
```

```
# prefinalmodel0: Stepwise with logged Mortgage.
prefinalmodel0 = glm(Personal.Loan ~ ., data = final_train, family = "binomial")
## Feature selection - stepwise
stepAIC(prefinalmodel0, direction = "both")
```

```
## Start:  AIC=820.61
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard + Experience2 + 
##     LoggedMortgage
## 
##                      Df Deviance     AIC
## - Experience2         1   790.62  818.62
## - Mortgage            1   791.11  819.11
## - LoggedMortgage      1   792.42  820.42
## <none>                    790.61  820.61
## - Securities.Account  1   795.74  823.74
## - CCAvg               1   795.92  823.92
## - CreditCard          1   803.52  831.52
## - Online              1   807.83  835.83
## - Family              3   864.66  888.66
## - CD.Account          1   878.76  906.76
## - Education           2  1082.26 1108.26
## - Income              1  1333.24 1361.24
## 
## Step:  AIC=818.62
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage
## 
##                      Df Deviance     AIC
## - Mortgage            1   791.11  817.11
## - LoggedMortgage      1   792.43  818.43
## <none>                    790.62  818.62
## + Experience2         1   790.61  820.61
## - Securities.Account  1   795.74  821.74
## - CCAvg               1   796.01  822.01
## - CreditCard          1   803.52  829.52
## - Online              1   807.84  833.84
## - Family              3   864.66  886.66
## - CD.Account          1   878.95  904.95
## - Education           2  1082.32 1106.32
## - Income              1  1333.25 1359.25
## 
## Step:  AIC=817.11
## Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account + 
##     CD.Account + Online + CreditCard + LoggedMortgage
## 
##                      Df Deviance     AIC
## <none>                    791.11  817.11
## - LoggedMortgage      1   794.07  818.07
## + Mortgage            1   790.62  818.62
## + Experience2         1   791.11  819.11
## - Securities.Account  1   796.05  820.05
## - CCAvg               1   796.93  820.93
## - CreditCard          1   803.83  827.83
## - Online              1   808.18  832.18
## - Family              3   864.83  884.83
## - CD.Account          1   879.01  903.01
## - Education           2  1082.53 1104.53
## - Income              1  1381.04 1405.04
```

```
## 
## Call:  glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education + 
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage, 
##     family = "binomial", data = final_train)
## 
## Coefficients:
##        (Intercept)              Income             Family2  
##          -12.64983             0.06365            -0.24090  
##            Family3             Family4               CCAvg  
##            1.91321             1.38231             0.13262  
##         Education2          Education3  Securities.Account1  
##            4.03980             4.20467            -0.74697  
##        CD.Account1             Online1         CreditCard1  
##            3.52663            -0.82814            -0.88255  
##     LoggedMortgage  
##            0.06474  
## 
## Degrees of Freedom: 3499 Total (i.e. Null);  3487 Residual
## Null Deviance:       2164 
## Residual Deviance: 791.1     AIC: 817.1
```

```
finalmodel0 = glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
    Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
    family = "binomial", data = final_train)
summary(finalmodel0)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
##     family = "binomial", data = final_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8579  -0.1805  -0.0646  -0.0192   4.1186
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -12.649830   0.669463 -18.895  < 2e-16 ***
## Income                0.063648   0.003719  17.116  < 2e-16 ***
## Family2              -0.240898   0.282962  -0.851 0.394579
## Family3               1.913214   0.295328   6.478 9.28e-11 ***
## Family4               1.382308   0.290321   4.761 1.92e-06 ***
## CCAvg                 0.132625   0.055363   2.396 0.016595 *
## Education2            4.039801   0.335595  12.038  < 2e-16 ***
## Education3            4.204674   0.335101  12.547  < 2e-16 ***
## Securities.Account1  -0.746968   0.350753  -2.130 0.033204 *
## CD.Account1           3.526629   0.403314   8.744  < 2e-16 ***
## Online1              -0.828144   0.202636  -4.087 4.37e-05 ***
## CreditCard1          -0.882548   0.259549  -3.400 0.000673 ***
## LoggedMortgage        0.064738   0.037410   1.730 0.083544 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2163.69  on 3499  degrees of freedom
## Residual deviance:  791.11  on 3487  degrees of freedom
## AIC: 817.11
##
## Number of Fisher Scoring iterations: 8
```

```
AIC(finalmodel0)          #  AIC = 817.1093
```

```
## [1] 817.1093
```

```
BIC(finalmodel0)          #  BIC = 897.1968
```

```
## [1] 897.196
```

```
## Testing finalmodel0
pred.final0 = predict(finalmodel0, final_test, type = "response")
final0.cutoff = 0.4
class.final0 = as.factor(if_else(pred.final0 < final0.cutoff, "No", "Yes"))
prop.table(table(class.final0))
```

```
## class.final0
##         No        Yes
## 0.91266667 0.08733333
```

```
## Confusion Matrix
confusionMatrix(class.final0, final_test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1333   36
##        Yes   12  119
##
##                Accuracy : 0.968
##                  95% CI : (0.9578, 0.9763)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8146
##
##  Mcnemar's Test P-Value : 0.0009009
##
##             Sensitivity : 0.9911
##             Specificity : 0.7677
##          Pos Pred Value : 0.9737
##          Neg Pred Value : 0.9084
##              Prevalence : 0.8967
##          Detection Rate : 0.8887
##    Detection Prevalence : 0.9127
##       Balanced Accuracy : 0.8794
##
##        'Positive' Class : No
##
```

```
                          #  Threshold   = 0.4
                          #  Accuracy    = 0.968
                          #  Sensitivity = 0.76774
                          #  Specificity = 0.99108
```

```
# prefinalmodel0a: Stepwise with logged Mortgage and 3rd power Income.
prefinalmodel0a = glm(Personal.Loan ~ ., data = final_train, family = "binomial")
## Feature selection - stepwise
stepAIC(prefinalmodel0a, direction = "both")
```

```
## Start:  AIC=820.61
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage +
##     Securities.Account + CD.Account + Online + CreditCard + Experience2 +
##     LoggedMortgage
##
##                      Df Deviance     AIC
## - Experience2         1   790.62  818.62
## - Mortgage            1   791.11  819.11
## - LoggedMortgage      1   792.42  820.42
## <none>                    790.61  820.61
## - Securities.Account  1   795.74  823.74
## - CCAvg               1   795.92  823.92
## - CreditCard          1   803.52  831.52
## - Online              1   807.83  835.83
## - Family              3   864.66  888.66
## - CD.Account          1   878.76  906.76
## - Education           2  1082.26 1108.26
## - Income              1  1333.24 1361.24
##
## Step:  AIC=818.62
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage
##
##                      Df Deviance     AIC
## - Mortgage            1   791.11  817.11
## - LoggedMortgage      1   792.43  818.43
## <none>                    790.62  818.62
## + Experience2         1   790.61  820.61
## - Securities.Account  1   795.74  821.74
## - CCAvg               1   796.01  822.01
## - CreditCard          1   803.52  829.52
## - Online              1   807.84  833.84
## - Family              3   864.66  886.66
## - CD.Account          1   878.95  904.95
## - Education           2  1082.32 1106.32
## - Income              1  1333.25 1359.25
##
## Step:  AIC=817.11
## Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account +
##     CD.Account + Online + CreditCard + LoggedMortgage
##
##                      Df Deviance     AIC
## <none>                    791.11  817.11
## - LoggedMortgage      1   794.07  818.07
## + Mortgage            1   790.62  818.62
## + Experience2         1   791.11  819.11
## - Securities.Account  1   796.05  820.05
## - CCAvg               1   796.93  820.93
## - CreditCard          1   803.83  827.83
## - Online              1   808.18  832.18
## - Family              3   864.83  884.83
## - CD.Account          1   879.01  903.01
## - Education           2  1082.53 1104.53
## - Income              1  1381.04 1405.04
```

```
##
## Call:  glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
##     family = "binomial", data = final_train)
##
## Coefficients:
##        (Intercept)              Income             Family2
##          -12.64983             0.06365            -0.24090
##            Family3             Family4               CCAvg
##            1.91321             1.38231             0.13262
##         Education2          Education3  Securities.Account1
##            4.03980             4.20467            -0.74697
##        CD.Account1             Online1         CreditCard1
##            3.52663            -0.82814            -0.88255
##     LoggedMortgage
##            0.06474
##
## Degrees of Freedom: 3499 Total (i.e. Null);  3487 Residual
## Null Deviance:       2164
## Residual Deviance: 791.1     AIC: 817.1
```

```r
finalmodel0a = glm(formula = Personal.Loan ~ poly(Income, 3) + Family + CCAvg + Education +
    Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
    family = "binomial", data = final_train)
summary(finalmodel0a)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ poly(Income, 3) + Family + CCAvg +
##     Education + Securities.Account + CD.Account + Online + CreditCard +
##     LoggedMortgage, family = "binomial", data = final_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5046  -0.1067  -0.0085  -0.0004   4.7966
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -11.70723    1.13599 -10.306  < 2e-16 ***
## poly(Income, 3)1     357.28275   57.75651   6.186 6.17e-10 ***
## poly(Income, 3)2    -128.05811   41.07901  -3.117 0.001825 **
## poly(Income, 3)3      -4.65606   18.48375  -0.252 0.801118
## Family2               -0.17600    0.28292  -0.622 0.533884
## Family3                2.47418    0.33878   7.303 2.81e-13 ***
## Family4                1.59917    0.32235   4.961 7.01e-07 ***
## CCAvg                  0.16062    0.05588   2.874 0.004048 **
## Education2             3.88617    0.32583  11.927  < 2e-16 ***
## Education3             4.02449    0.32635  12.332  < 2e-16 ***
## Securities.Account1   -0.85671    0.40702  -2.105 0.035304 *
## CD.Account1            3.75879    0.47962   7.837 4.61e-15 ***
## Online1               -0.86468    0.22927  -3.771 0.000162 ***
## CreditCard1           -0.96375    0.29073  -3.315 0.000917 ***
## LoggedMortgage         0.09663    0.04152   2.327 0.019958 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2163.7  on 3499  degrees of freedom
## Residual deviance:  625.5  on 3485  degrees of freedom
## AIC: 655.5
##
## Number of Fisher Scoring iterations: 11
```

```r
AIC(finalmodel0a)          #  AIC = 655.5022
```

```
## [1] 655.5022
```

```r
BIC(finalmodel0a)          #  BIC = 747.9099
```

```
## [1] 747.9099
```

```r
## Testing finalmodela
pred.final0a = predict(finalmodel0a, final_test, type = "response")
final0a.cutoff = 0.33
class.final0a = as.factor(if_else(pred.final0a < final0a.cutoff, "No", "Yes"))
prop.table(table(class.final0a))
```

```
## class.final0a
##   No  Yes
## 0.89 0.11
```

```r
## Confusion Matrix
confusionMatrix(class.final0a, final_test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   No  Yes
##        No  1319   16
##        Yes   26  139
##                                           
##                Accuracy : 0.972           
##                  95% CI : (0.9623, 0.9797)
##     No Information Rate : 0.8967          
##     P-Value [Acc > NIR] : <2e-16          
##                                           
##                   Kappa : 0.8531          
##                                           
##  Mcnemar's Test P-Value : 0.1649          
##                                           
##             Sensitivity : 0.9807          
##             Specificity : 0.8968          
##          Pos Pred Value : 0.9880          
##          Neg Pred Value : 0.8424          
##              Prevalence : 0.8967          
##          Detection Rate : 0.8793          
##    Detection Prevalence : 0.8900          
##       Balanced Accuracy : 0.9387          
##                                           
##        'Positive' Class : No              
## 
```

```
                          #  Threshold   = 0.33
                          #  Accuracy    = 0.972
                          #  Sensitivity = 0.89677
                          #  Specificity = 0.98067
```

```
# prefinalmodel1: Stepwise with logged Mortgage and below observations removed.
model1_train = final_train[-c(350, 976, 1070, 1127, 2159),]
prefinalmodel1 = glm(Personal.Loan ~ ., data = model1_train, family = "binomial")
## Feature selection - stepwise
stepAIC(prefinalmodel1, direction = "both")
```

```
## Start:  AIC=817.42
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage +
##     Securities.Account + CD.Account + Online + CreditCard + Experience2 +
##     LoggedMortgage
##
##                      Df Deviance     AIC
## - Experience2         1   787.42  815.42
## - Mortgage            1   787.94  815.94
## - LoggedMortgage      1   789.22  817.22
## <none>                    787.42  817.42
## - Securities.Account  1   792.70  820.70
## - CCAvg               1   793.53  821.53
## - CreditCard          1   800.65  828.65
## - Online              1   805.10  833.10
## - Family              3   861.44  885.44
## - CD.Account          1   875.43  903.43
## - Education           2  1077.58 1103.58
## - Income              1  1331.23 1359.23
##
## Step:  AIC=815.42
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage
##
##                      Df Deviance     AIC
## - Mortgage            1   787.94  813.94
## - LoggedMortgage      1   789.23  815.23
## <none>                    787.42  815.42
## + Experience2         1   787.42  817.42
## - Securities.Account  1   792.70  818.70
## - CCAvg               1   793.60  819.60
## - CreditCard          1   800.66  826.66
## - Online              1   805.10  831.10
## - Family              3   861.44  883.44
## - CD.Account          1   875.67  901.67
## - Education           2  1077.63 1101.63
## - Income              1  1331.24 1357.24
##
## Step:  AIC=813.94
## Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account +
##     CD.Account + Online + CreditCard + LoggedMortgage
##
##                      Df Deviance     AIC
## <none>                    787.94  813.94
## - LoggedMortgage      1   790.73  814.73
## + Mortgage            1   787.42  815.42
## + Experience2         1   787.94  815.94
## - Securities.Account  1   793.03  817.03
## - CCAvg               1   794.58  818.58
## - CreditCard          1   800.99  824.99
## - Online              1   805.46  829.46
## - Family              3   861.63  881.63
## - CD.Account          1   875.74  899.74
## - Education           2  1077.85 1099.85
## - Income              1  1378.66 1402.66
```

```
##
## Call:  glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
##     family = "binomial", data = model1_train)
##
## Coefficients:
##        (Intercept)              Income             Family2
##          -12.63965             0.06374            -0.28917
##            Family3             Family4               CCAvg
##            1.88835             1.35862             0.14255
##         Education2          Education3  Securities.Account1
##            4.03491             4.19975            -0.75893
##        CD.Account1             Online1         CreditCard1
##            3.53185            -0.84114            -0.89448
##     LoggedMortgage
##            0.06285
##
## Degrees of Freedom: 3494 Total (i.e. Null);  3482 Residual
## Null Deviance:       2163
## Residual Deviance: 787.9     AIC: 813.9
```

```
finalmodel1 = glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
    Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
    family = "binomial", data = model1_train)
summary(finalmodel1)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
##     family = "binomial", data = model1_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8681  -0.1811  -0.0640  -0.0188   4.1208
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -12.639653   0.669128 -18.890  < 2e-16 ***
## Income                0.063743   0.003725  17.114  < 2e-16 ***
## Family2              -0.289167   0.284194  -1.017 0.308917
## Family3               1.888351   0.295229   6.396 1.59e-10 ***
## Family4               1.358621   0.290154   4.682 2.84e-06 ***
## CCAvg                 0.142553   0.055765   2.556 0.010578 *
## Education2            4.034915   0.335867  12.013  < 2e-16 ***
## Education3            4.199750   0.335388  12.522  < 2e-16 ***
## Securities.Account1  -0.758926   0.351432  -2.160 0.030810 *
## CD.Account1           3.531847   0.404355   8.735  < 2e-16 ***
## Online1              -0.841139   0.203271  -4.138 3.50e-05 ***
## CreditCard1          -0.894477   0.259920  -3.441 0.000579 ***
## LoggedMortgage        0.062850   0.037466   1.678 0.093442 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2162.72  on 3494  degrees of freedom
## Residual deviance:  787.94  on 3482  degrees of freedom
## AIC: 813.94
##
## Number of Fisher Scoring iterations: 8
```

```
AIC(finalmodel1)          #  AIC = 813.9434
```

```
## [1] 813.9434
```

```
BIC(finalmodel1)          #  BIC = 894.0116
```

```
## [1] 894.0116
```

```
## Testing finalmodel1
pred.final1 = predict(finalmodel1, final_test, type = "response")
final1.cutoff = 0.4
class.final1 = as.factor(if_else(pred.final1 < final1.cutoff, "No", "Yes"))
prop.table(table(class.final1))
```

```
## class.final1
##         No        Yes
## 0.91066667 0.08933333
```

```
## Confusion Matrix
confusionMatrix(class.final1, final_test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1331   35
##        Yes   14  120
##
##                Accuracy : 0.9673
##                  95% CI : (0.957, 0.9757)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8125
##
##  Mcnemar's Test P-Value : 0.004275
##
##             Sensitivity : 0.9896
##             Specificity : 0.7742
##          Pos Pred Value : 0.9744
##          Neg Pred Value : 0.8955
##              Prevalence : 0.8967
##          Detection Rate : 0.8873
##    Detection Prevalence : 0.9107
##       Balanced Accuracy : 0.8819
##
##        'Positive' Class : No
##
```

```
                          #  Threshold   = 0.4
                          #  Accuracy    = 0.9673
                          #  Sensitivity = 0.77419
                          #  Specificity = 0.98959
```

```
# prefinalmodel1a: Stepwise with logged Mortgage, below observations removed, and 3rd power Income.
model1_train = final_train[-c(350, 976, 1070, 1127, 2159),]
prefinalmodel1a = glm(Personal.Loan ~ ., data = model1_train, family = "binomial")
## Feature selection - stepwise
stepAIC(prefinalmodel1a, direction = "both")
```

```
## Start:  AIC=817.42
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard + Experience2 + 
##     LoggedMortgage
## 
##                      Df Deviance     AIC
## - Experience2         1   787.42  815.42
## - Mortgage            1   787.94  815.94
## - LoggedMortgage      1   789.22  817.22
## <none>                    787.42  817.42
## - Securities.Account  1   792.70  820.70
## - CCAvg               1   793.53  821.53
## - CreditCard          1   800.65  828.65
## - Online              1   805.10  833.10
## - Family              3   861.44  885.44
## - CD.Account          1   875.43  903.43
## - Education           2  1077.58 1103.58
## - Income              1  1331.23 1359.23
## 
## Step:  AIC=815.42
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage
## 
##                      Df Deviance     AIC
## - Mortgage            1   787.94  813.94
## - LoggedMortgage      1   789.23  815.23
## <none>                    787.42  815.42
## + Experience2         1   787.42  817.42
## - Securities.Account  1   792.70  818.70
## - CCAvg               1   793.60  819.60
## - CreditCard          1   800.66  826.66
## - Online              1   805.10  831.10
## - Family              3   861.44  883.44
## - CD.Account          1   875.67  901.67
## - Education           2  1077.63 1101.63
## - Income              1  1331.24 1357.24
## 
## Step:  AIC=813.94
## Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account + 
##     CD.Account + Online + CreditCard + LoggedMortgage
## 
##                      Df Deviance     AIC
## <none>                    787.94  813.94
## - LoggedMortgage      1   790.73  814.73
## + Mortgage            1   787.42  815.42
## + Experience2         1   787.94  815.94
## - Securities.Account  1   793.03  817.03
## - CCAvg               1   794.58  818.58
## - CreditCard          1   800.99  824.99
## - Online              1   805.46  829.46
## - Family              3   861.63  881.63
## - CD.Account          1   875.74  899.74
## - Education           2  1077.85 1099.85
## - Income              1  1378.66 1402.66
```

```
## 
## Call:  glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education + 
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage, 
##     family = "binomial", data = model1_train)
## 
## Coefficients:
##        (Intercept)              Income             Family2  
##          -12.63965             0.06374            -0.28917  
##            Family3             Family4               CCAvg  
##            1.88835             1.35862             0.14255  
##         Education2          Education3  Securities.Account1  
##            4.03491             4.19975            -0.75893  
##        CD.Account1             Online1         CreditCard1  
##            3.53185            -0.84114            -0.89448  
##     LoggedMortgage  
##            0.06285  
## 
## Degrees of Freedom: 3494 Total (i.e. Null);  3482 Residual
## Null Deviance:       2163 
## Residual Deviance: 787.9     AIC: 813.9
```

```
finalmodel1a = glm(formula = Personal.Loan ~ poly(Income, 3) + Family + CCAvg + Education +
    Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
    family = "binomial", data = model1_train)
summary(finalmodel1a)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ poly(Income, 3) + Family + CCAvg +
##     Education + Securities.Account + CD.Account + Online + CreditCard +
##     LoggedMortgage, family = "binomial", data = model1_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4994  -0.1072  -0.0085  -0.0004   4.7967
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -11.71235    1.13606 -10.310  < 2e-16 ***
## poly(Income, 3)1     357.74419   57.71361   6.199 5.70e-10 ***
## poly(Income, 3)2    -128.16066   40.98097  -3.127 0.001764 **
## poly(Income, 3)3      -4.09516   18.41518  -0.222 0.824018
## Family2               -0.18928    0.28343  -0.668 0.504250
## Family3                2.46348    0.33886   7.270 3.60e-13 ***
## Family4                1.59104    0.32230   4.936 7.96e-07 ***
## CCAvg                  0.16322    0.05603   2.913 0.003578 **
## Education2             3.88120    0.32577  11.914  < 2e-16 ***
## Education3             4.02008    0.32633  12.319  < 2e-16 ***
## Securities.Account1   -0.85804    0.40683  -2.109 0.034937 *
## CD.Account1            3.75405    0.47939   7.831 4.84e-15 ***
## Online1               -0.86466    0.22930  -3.771 0.000163 ***
## CreditCard1           -0.96647    0.29056  -3.326 0.000880 ***
## LoggedMortgage         0.09568    0.04151   2.305 0.021173 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2162.7  on 3494  degrees of freedom
## Residual deviance:  624.9  on 3480  degrees of freedom
## AIC: 654.9
##
## Number of Fisher Scoring iterations: 11
```

```
AIC(finalmodel1a)          #  AIC = 654.8965
```

```
## [1] 654.8965
```

```
BIC(finalmodel1a)          #  BIC = 747.2828
```

```
## [1] 747.2828
```

```
## Testing finalmodel1a
pred.final1a = predict(finalmodel1a, final_test, type = "response")
final1a.cutoff = 0.33
class.final1a = as.factor(if_else(pred.final1a < final1a.cutoff, "No", "Yes"))
prop.table(table(class.final1a))
```

```
## class.final1a
##   No  Yes
## 0.89 0.11
```

```
## Confusion Matrix
confusionMatrix(class.final1a, final_test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1319   16
##        Yes   26  139
##
##                Accuracy : 0.972
##                  95% CI : (0.9623, 0.9797)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8531
##
##  Mcnemar's Test P-Value : 0.1649
##
##             Sensitivity : 0.9807
##             Specificity : 0.8968
##          Pos Pred Value : 0.9880
##          Neg Pred Value : 0.8424
##              Prevalence : 0.8967
##          Detection Rate : 0.8793
##    Detection Prevalence : 0.8900
##       Balanced Accuracy : 0.9387
##
##        'Positive' Class : No
##
```

```
                          #  Threshold   = 0.33
                          #  Accuracy    = 0.972
                          #  Sensitivity = 0.89677
                          #  Specificity = 0.98067
```

Objective 2: Inspecting Interactions

```
# This is using the original dataset

library(sjPlot)    #For effect plotting
```

```
## Warning: package 'sjPlot' was built under R version 4.1.3
```

```
## Learn more about sjPlot with 'browseVignettes("sjPlot")'.
```

```
library(sjmisc)    #For effect plotting
```

```
## Warning: package 'sjmisc' was built under R version 4.1.3
```

```
##
## Attaching package: 'sjmisc'
```

```
## The following objects are masked from 'package:jtools':
##
##     %nin%, center
```

```
## The following object is masked from 'package:purrr':
##
##     is_empty
```

```
## The following object is masked from 'package:tidyr':
##
##     replace_na
```

```
## The following object is masked from 'package:tibble':
##
##     add_case
```

```
getwd()
```

```
## [1] "C:/Users/dnguy/Desktop/2 Applied Stats/Project 2/Statistcs2-project-2"
```

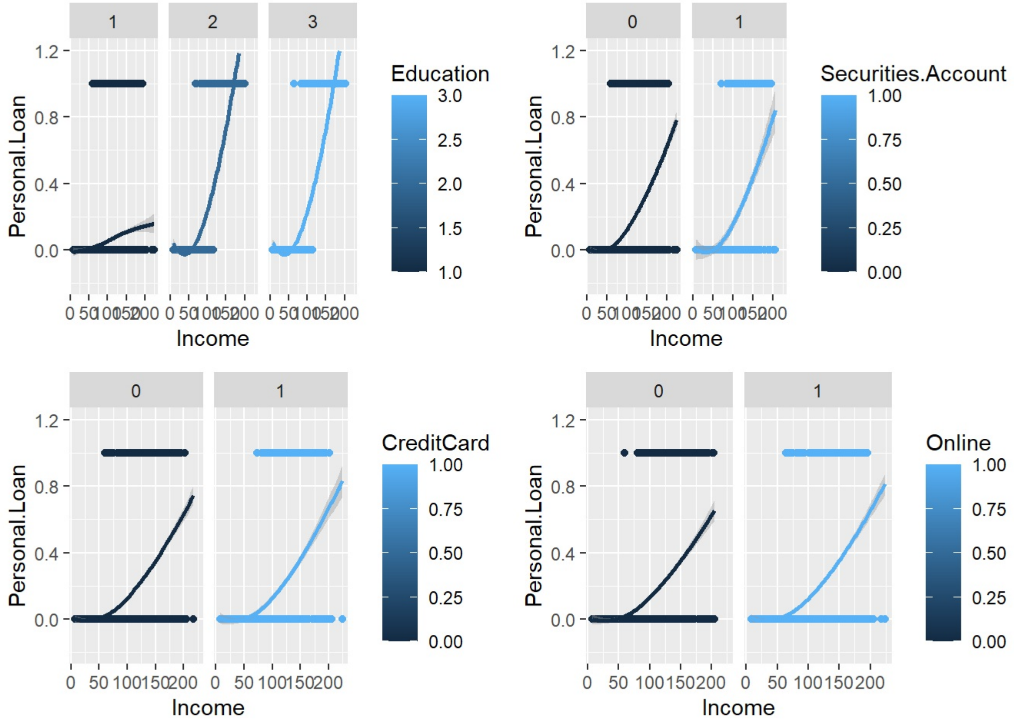
```
PersonalL=read.csv("Bank_Personal_Loan_Modelling.csv")

# Age is omitted for plotting since it's replaced by Experience2.

#Education,Family,CCAvg,Online,CreditCard, Securities.Account
a=ggplot(PersonalL,aes(x=Income,y=Personal.Loan,colour=Education))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Education)

b0=ggplot(PersonalL,aes(x=Income,y=Personal.Loan,colour=Family))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Family)

c=ggplot(PersonalL,aes(x=Income,y=Personal.Loan,colour=CCAvg))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~CCAvg)

d=ggplot(PersonalL,aes(x=Income,y=Personal.Loan,colour=Online))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Online)

e=ggplot(PersonalL,aes(x=Income,y=Personal.Loan,colour=CreditCard))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~CreditCard)

f=ggplot(PersonalL,aes(x=Income,y=Personal.Loan,colour=Securities.Account))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Securities.Account)


grid.arrange(a,f,e,d, ncol=2)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 14 rows containing missing values (geom_smooth).
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

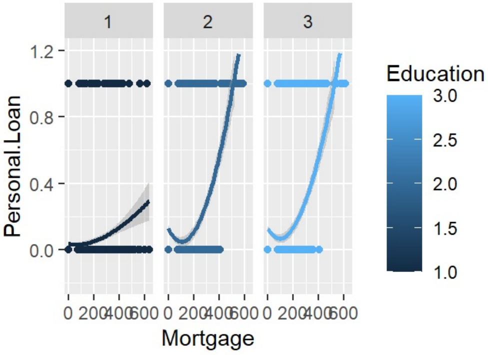


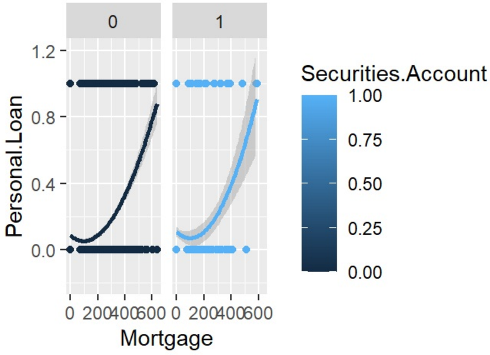

```
#Education,Family,CCAvg,Online,CreditCard, Securities.Account
a=ggplot(PersonalL,aes(x=Mortgage,y=Personal.Loan,colour=Education))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Education)

b1=ggplot(PersonalL,aes(x=Mortgage,y=Personal.Loan,colour=Family))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Family)

c=ggplot(PersonalL,aes(x=Mortgage,y=Personal.Loan,colour=CCAvg))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~CCAvg)

d=ggplot(PersonalL,aes(x=Mortgage,y=Personal.Loan,colour=Online))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Online)

e=ggplot(PersonalL,aes(x=Mortgage,y=Personal.Loan,colour=CreditCard))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~CreditCard)

f=ggplot(PersonalL,aes(x=Mortgage,y=Personal.Loan,colour=Securities.Account))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Securities.Account)

grid.arrange(a,f,e,d, ncol=2)
```

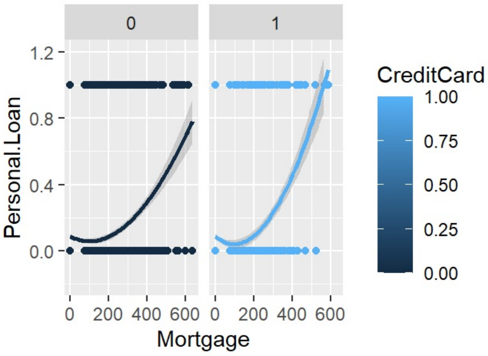
```
## `geom_smooth()` using formula 'y ~ x'
```

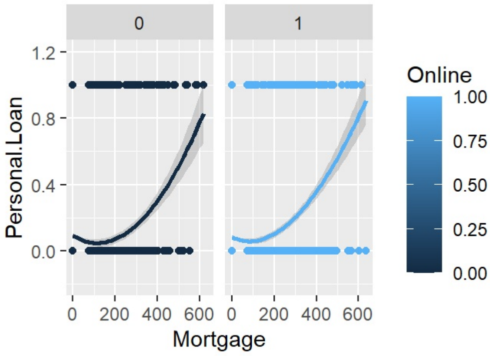
```
## Warning: Removed 11 rows containing missing values (geom_smooth).
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```










```
#Education,Family,CCAvg,Online,CreditCard, Securities.Account
a=ggplot(PersonalL,aes(x=CCAvg,y=Personal.Loan,colour=Education))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Education)

b2=ggplot(PersonalL,aes(x=CCAvg,y=Personal.Loan,colour=Family))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Family)

c=ggplot(PersonalL,aes(x=CCAvg,y=Personal.Loan,colour=CCAvg))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~CCAvg)

d=ggplot(PersonalL,aes(x=CCAvg,y=Personal.Loan,colour=Online))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Online)

e=ggplot(PersonalL,aes(x=CCAvg,y=Personal.Loan,colour=CreditCard))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~CreditCard)

f=ggplot(PersonalL,aes(x=CCAvg,y=Personal.Loan,colour=Securities.Account))+geom_point()+
  geom_smooth(method="loess",size=1,span=1.5)+
  ylim(-.2,1.2)+
  facet_wrap(~Securities.Account)
```

```
# prefinalmodel2: Stepwise with logged Mortgage, below observations removed, 3rd power Income, and Income*Family.
model1_train = final_train[-c(350, 976, 1070, 1127, 2159),]
prefinalmodel2 = glm(Personal.Loan ~ ., data = model1_train, family = "binomial")
## Feature selection - stepwise
stepAIC(prefinalmodel2, direction = "both")
```

```
## Start:  AIC=817.42
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard + Experience2 + 
##     LoggedMortgage
## 
##                      Df Deviance     AIC
## - Experience2         1   787.42  815.42
## - Mortgage            1   787.94  815.94
## - LoggedMortgage      1   789.22  817.22
## <none>                    787.42  817.42
## - Securities.Account  1   792.70  820.70
## - CCAvg               1   793.53  821.53
## - CreditCard          1   800.65  828.65
## - Online              1   805.10  833.10
## - Family              3   861.44  885.44
## - CD.Account          1   875.43  903.43
## - Education           2  1077.58 1103.58
## - Income              1  1331.23 1359.23
## 
## Step:  AIC=815.42
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage + 
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage
## 
##                      Df Deviance     AIC
## - Mortgage            1   787.94  813.94
## - LoggedMortgage      1   789.23  815.23
## <none>                    787.42  815.42
## + Experience2         1   787.42  817.42
## - Securities.Account  1   792.70  818.70
## - CCAvg               1   793.60  819.60
## - CreditCard          1   800.66  826.66
## - Online              1   805.10  831.10
## - Family              3   861.44  883.44
## - CD.Account          1   875.67  901.67
## - Education           2  1077.63 1101.63
## - Income              1  1331.24 1357.24
## 
## Step:  AIC=813.94
## Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account + 
##     CD.Account + Online + CreditCard + LoggedMortgage
## 
##                      Df Deviance     AIC
## <none>                    787.94  813.94
## - LoggedMortgage      1   790.73  814.73
## + Mortgage            1   787.42  815.42
## + Experience2         1   787.94  815.94
## - Securities.Account  1   793.03  817.03
## - CCAvg               1   794.58  818.58
## - CreditCard          1   800.99  824.99
## - Online              1   805.46  829.46
## - Family              3   861.63  881.63
## - CD.Account          1   875.74  899.74
## - Education           2  1077.85 1099.85
## - Income              1  1378.66 1402.66
```

```
## 
## Call:  glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education + 
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage, 
##     family = "binomial", data = model1_train)
## 
## Coefficients:
##        (Intercept)              Income             Family2  
##          -12.63965             0.06374            -0.28917  
##            Family3             Family4               CCAvg  
##            1.88835             1.35862             0.14255  
##         Education2          Education3  Securities.Account1  
##            4.03491             4.19975            -0.75893  
##        CD.Account1             Online1         CreditCard1  
##            3.53185            -0.84114            -0.89448  
##     LoggedMortgage  
##            0.06285  
## 
## Degrees of Freedom: 3494 Total (i.e. Null);  3482 Residual
## Null Deviance:       2163 
## Residual Deviance: 787.9     AIC: 813.9
```

```
finalmodel2 = glm(formula = Personal.Loan ~ poly(Income, 3) + Family + Income*Family + CCAvg + Education + Securi
ties.Account + CD.Account + Online + CreditCard + LoggedMortgage,
    family = "binomial", data = model1_train)
#finalmodel2 = glm(formula = Personal.Loan ~ poly(Income, 3) + Family + Income*Family + CCAvg + CCAvg*Family + Ed
ucation + CCAvg*Education + Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage + LoggedMortga
ge*Education + LoggedMortgage*Family, family = "binomial", data = model1_train)
summary(finalmodel2)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ poly(Income, 3) + Family + Income *
##     Family + CCAvg + Education + Securities.Account + CD.Account +
##     Online + CreditCard + LoggedMortgage, family = "binomial",
##     data = model1_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4936  -0.1100  -0.0175  -0.0015   4.2247
##
## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -8.684063   0.885151  -9.811  < 2e-16 ***
## poly(Income, 3)1    193.117742  44.684965   4.322 1.55e-05 ***
## poly(Income, 3)2    -53.501216  32.211896  -1.661 0.096731 .
## poly(Income, 3)3    -19.228152  16.524968  -1.164 0.244594
## Family2               0.104771   1.053352   0.099 0.920770
## Family3              -8.076632   2.321404  -3.479 0.000503 ***
## Family4             -10.301547   2.401639  -4.289 1.79e-05 ***
## Income                      NA         NA      NA       NA
## CCAvg                 0.232488   0.064788   3.588 0.000333 ***
## Education2            4.293987   0.359126  11.957  < 2e-16 ***
## Education3            4.271829   0.352929  12.104  < 2e-16 ***
## Securities.Account1  -0.824892   0.428669  -1.924 0.054316 .
## CD.Account1           3.571350   0.507698   7.034 2.00e-12 ***
## Online1              -0.863168   0.251394  -3.434 0.000596 ***
## CreditCard1          -1.027839   0.311468  -3.300 0.000967 ***
## LoggedMortgage        0.050310   0.045351   1.109 0.267277
## Family2:Income       -0.001187   0.007887  -0.151 0.880340
## Family3:Income        0.092702   0.021798   4.253 2.11e-05 ***
## Family4:Income        0.106746   0.021944   4.864 1.15e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2162.72  on 3494  degrees of freedom
## Residual deviance:  548.51  on 3477  degrees of freedom
## AIC: 584.51
##
## Number of Fisher Scoring iterations: 11
```

```
AIC(finalmodel2)          #  AIC = 549.3415
```

```
## [1] 584.5078
```

```
BIC(finalmodel2)          #  BIC = 678.6824
```

```
## [1] 695.3714
```

```
## Testing finalmodel2
pred.final2 = predict(finalmodel2, final_test, type = "response")
```

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

```
final2.cutoff = 0.34
class.final2 = as.factor(if_else(pred.final2 < final2.cutoff, "No", "Yes"))
prop.table(table(class.final2))
```

```
## class.final2
##        No       Yes
## 0.8906667 0.1093333
```

```
## Confusion Matrix
confusionMatrix(class.final2, final_test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1321   15
##        Yes   24  140
##
##                Accuracy : 0.974
##                  95% CI : (0.9646, 0.9814)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8632
##
##  Mcnemar's Test P-Value : 0.2002
##
##             Sensitivity : 0.9822
##             Specificity : 0.9032
##          Pos Pred Value : 0.9888
##          Neg Pred Value : 0.8537
##              Prevalence : 0.8967
##          Detection Rate : 0.8807
##    Detection Prevalence : 0.8907
##       Balanced Accuracy : 0.9427
##
##        'Positive' Class : No
##
```

```
                           #  Threshold   = 0.34
                           #  Accuracy    = 0.974
                           #  Sensitivity = 0.90323
                           #  Specificity = 0.98216
```

```
# prefinalmodel2a: Stepwise with logged Mortgage, 3rd power Income, and Income*Family.
prefinalmodel2a = glm(Personal.Loan ~ ., data = final_train, family = "binomial")
## Feature selection - stepwise
stepAIC(prefinalmodel2a, direction = "both")
```

```
## Start:  AIC=820.61
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage +
##     Securities.Account + CD.Account + Online + CreditCard + Experience2 +
##     LoggedMortgage
##
##                      Df Deviance     AIC
## - Experience2         1   790.62  818.62
## - Mortgage            1   791.11  819.11
## - LoggedMortgage      1   792.42  820.42
## <none>                    790.61  820.61
## - Securities.Account  1   795.74  823.74
## - CCAvg               1   795.92  823.92
## - CreditCard          1   803.52  831.52
## - Online              1   807.83  835.83
## - Family              3   864.66  888.66
## - CD.Account          1   878.76  906.76
## - Education           2  1082.26 1108.26
## - Income              1  1333.24 1361.24
##
## Step:  AIC=818.62
## Personal.Loan ~ Income + Family + CCAvg + Education + Mortgage +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage
##
##                      Df Deviance     AIC
## - Mortgage            1   791.11  817.11
## - LoggedMortgage      1   792.43  818.43
## <none>                    790.62  818.62
## + Experience2         1   790.61  820.61
## - Securities.Account  1   795.74  821.74
## - CCAvg               1   796.01  822.01
## - CreditCard          1   803.52  829.52
## - Online              1   807.84  833.84
## - Family              3   864.66  886.66
## - CD.Account          1   878.95  904.95
## - Education           2  1082.32 1106.32
## - Income              1  1333.25 1359.25
##
## Step:  AIC=817.11
## Personal.Loan ~ Income + Family + CCAvg + Education + Securities.Account +
##     CD.Account + Online + CreditCard + LoggedMortgage
##
##                      Df Deviance     AIC
## <none>                    791.11  817.11
## - LoggedMortgage      1   794.07  818.07
## + Mortgage            1   790.62  818.62
## + Experience2         1   791.11  819.11
## - Securities.Account  1   796.05  820.05
## - CCAvg               1   796.93  820.93
## - CreditCard          1   803.83  827.83
## - Online              1   808.18  832.18
## - Family              3   864.83  884.83
## - CD.Account          1   879.01  903.01
## - Education           2  1082.53 1104.53
## - Income              1  1381.04 1405.04
```

```
##
## Call:  glm(formula = Personal.Loan ~ Income + Family + CCAvg + Education +
##     Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
##     family = "binomial", data = final_train)
##
## Coefficients:
##        (Intercept)              Income             Family2
##          -12.64983             0.06365            -0.24090
##            Family3             Family4               CCAvg
##            1.91321             1.38231             0.13262
##         Education2          Education3  Securities.Account1
##            4.03980             4.20467            -0.74697
##        CD.Account1             Online1         CreditCard1
##            3.52663            -0.82814            -0.88255
##     LoggedMortgage
##            0.06474
##
## Degrees of Freedom: 3499 Total (i.e. Null);  3487 Residual
## Null Deviance:       2164
## Residual Deviance: 791.1     AIC: 817.1
```

```r
finalmodel2a = glm(formula = Personal.Loan ~ poly(Income, 3) + Family + Income*Family + CCAvg + Education + Securities.Account + CD.Account + Online + CreditCard + LoggedMortgage,
    family = "binomial", data = model1_train)

summary(finalmodel2a)
```

```
##
## Call:
## glm(formula = Personal.Loan ~ poly(Income, 3) + Family + Income *
##     Family + CCAvg + Education + Securities.Account + CD.Account +
##     Online + CreditCard + LoggedMortgage, family = "binomial",
##     data = model1_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4936  -0.1100  -0.0175  -0.0015   4.2247
##
## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -8.684063   0.885151  -9.811  < 2e-16 ***
## poly(Income, 3)1    193.117742  44.684965   4.322 1.55e-05 ***
## poly(Income, 3)2    -53.501216  32.211896  -1.661 0.096731 .
## poly(Income, 3)3    -19.228152  16.524968  -1.164 0.244594
## Family2               0.104771   1.053352   0.099 0.920770
## Family3              -8.076632   2.321404  -3.479 0.000503 ***
## Family4             -10.301547   2.401639  -4.289 1.79e-05 ***
## Income                      NA         NA      NA       NA
## CCAvg                 0.232488   0.064788   3.588 0.000333 ***
## Education2            4.293987   0.359126  11.957  < 2e-16 ***
## Education3            4.271829   0.352929  12.104  < 2e-16 ***
## Securities.Account1  -0.824892   0.428669  -1.924 0.054316 .
## CD.Account1           3.571350   0.507698   7.034 2.00e-12 ***
## Online1              -0.863168   0.251394  -3.434 0.000596 ***
## CreditCard1          -1.027839   0.311468  -3.300 0.000967 ***
## LoggedMortgage        0.050310   0.045351   1.109 0.267277
## Family2:Income       -0.001187   0.007887  -0.151 0.880340
## Family3:Income        0.092702   0.021798   4.253 2.11e-05 ***
## Family4:Income        0.106746   0.021944   4.864 1.15e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2162.72  on 3494  degrees of freedom
## Residual deviance:  548.51  on 3477  degrees of freedom
## AIC: 584.51
##
## Number of Fisher Scoring iterations: 11
```

```r
AIC(finalmodel2a)          #  AIC = 584.5078
```

```
## [1] 584.5078
```

```r
BIC(finalmodel2a)          #  BIC = 695.3714
```

```
## [1] 695.3714
```

```r
## Testing finalmodel2a
pred.final2a = predict(finalmodel2a, final_test, type = "response")
```

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

```r
final2a.cutoff = 0.28
class.final2a = as.factor(if_else(pred.final2a < final2a.cutoff, "No", "Yes"))
prop.table(table(class.final2a))
```

```
## class.final2a
##    No   Yes
## 0.886 0.114
```

```
## Confusion Matrix
confusionMatrix(class.final2a, final_test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1316   13
##        Yes   29  142
##
##                Accuracy : 0.972
##                  95% CI : (0.9623, 0.9797)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8555
##
##  Mcnemar's Test P-Value : 0.02064
##
##             Sensitivity : 0.9784
##             Specificity : 0.9161
##          Pos Pred Value : 0.9902
##          Neg Pred Value : 0.8304
##              Prevalence : 0.8967
##          Detection Rate : 0.8773
##    Detection Prevalence : 0.8860
##       Balanced Accuracy : 0.9473
##
##        'Positive' Class : No
##
```

```
                          #  Threshold   = 0.28
                          #  Accuracy    = 0.972
                          #  Sensitivity = 0.91613
                          #  Specificity = 0.97844
```

```
## prefinalmodel2b: Repeated K-Fold Cross Validation with logged Mortgage, 3rd power Income, and Income*Family.
ctrl <- trainControl(method = "repeatedcv", number = 10, savePredictions = TRUE)
mod_fit <- train(Personal.Loan ~ poly(Income, 3) + Family + Income*Family + CCAvg + Education + Securities.Accoun
t + CD.Account + Online + CreditCard + LoggedMortgage,
                data = final_df,
                method = "glm", family = "binomial",
                trControl = ctrl, tuneLength = 5)
```

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

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

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

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

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

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

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

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

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

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

```
pred.final2b = predict(mod_fit, newdata = final_test)
```

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

```
confusionMatrix(data=pred.final2b, final_test$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1336   23
##        Yes    9  132
##
##                Accuracy : 0.9787
##                  95% CI : (0.97, 0.9854)
##     No Information Rate : 0.8967
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8801
##
##  Mcnemar's Test P-Value : 0.02156
##
##             Sensitivity : 0.9933
##             Specificity : 0.8516
##          Pos Pred Value : 0.9831
##          Neg Pred Value : 0.9362
##              Prevalence : 0.8967
##          Detection Rate : 0.8907
##    Detection Prevalence : 0.9060
##       Balanced Accuracy : 0.9225
##
##        'Positive' Class : No
##
```

```
                          #  Accuracy    = 0.9787
                          #  Sensitivity = 0.8516
                          #  Specificity = 0.9933
```

```
## ROC Curve
test_label = final_df[-split, "Personal.Loan"]
results.model1a = prediction(pred.final1a, test_label)
length(pred.final1a)
```

```
## [1] 1500
```

```
length(test_label)
```

```
## [1] 1500
```

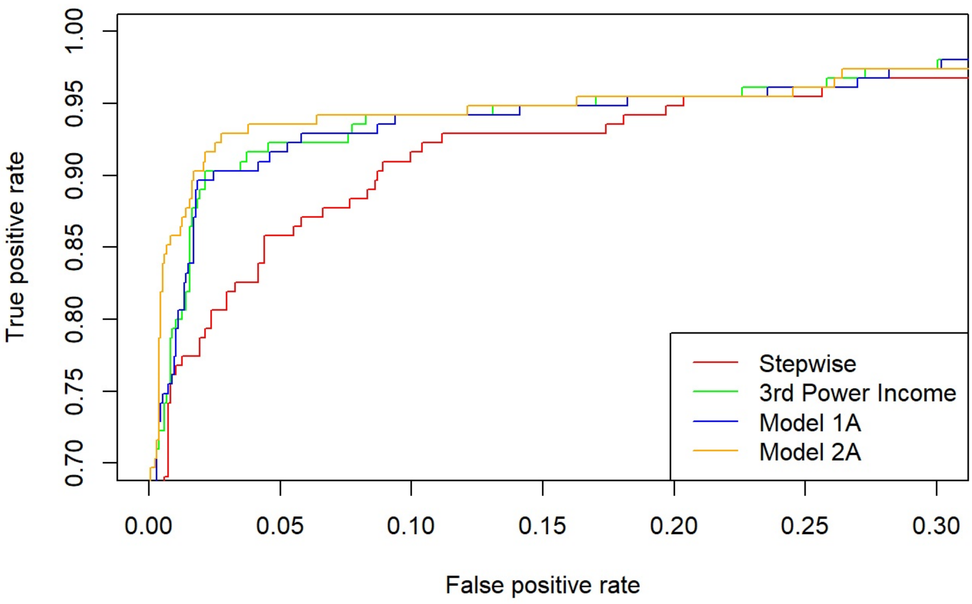
```
roc.model1a = performance(results.model1a, measure = "tpr", x.measure = "fpr")

results.model2a = prediction(pred.final2a, test_label)
length(pred.final2a)
```

```
## [1] 1500
```

```
length(test_label)
```

```
## [1] 1500
```

```
roc.model2a = performance(results.model2a, measure = "tpr", x.measure = "fpr")

plot(roc.step, col = "red", xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.poly.income3, col = "green", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.model1a, col = "blue", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.model2a, col = "orange", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
legend("bottomright", legend = c("Stepwise", "3rd Power Income", "Model 1A", "Model 2A"),
       col = c("red", "green", "blue", "orange"),
       lty=1, lwd=1)
```



Objective 2: LDA (continuous predictors only)

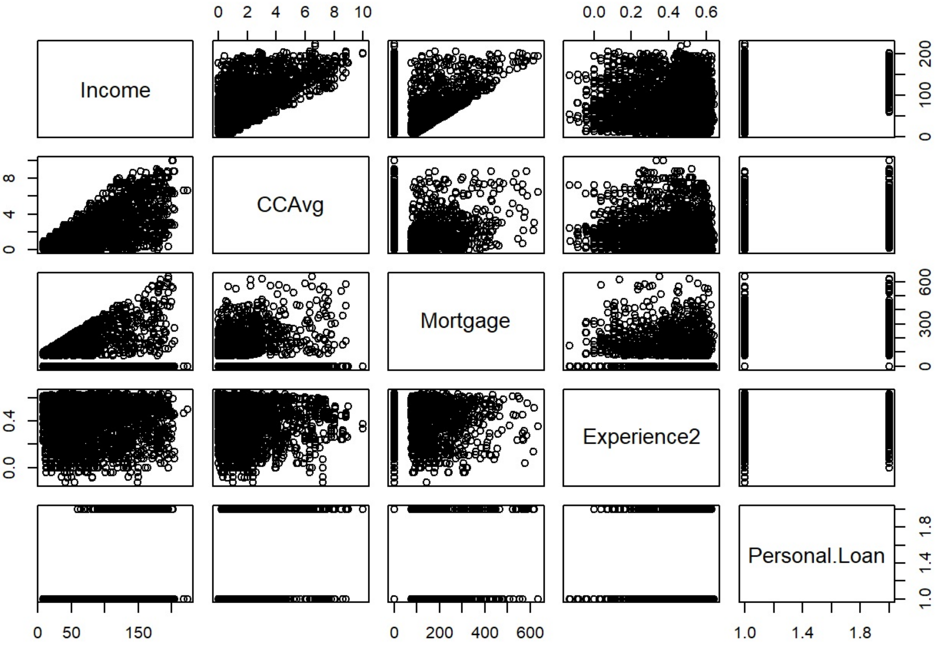
```
# Find only continuous predictors
str(train)
```

```
## 'data.frame':    3500 obs. of  11 variables:
##  $ Income           : int  23 52 98 79 95 63 91 143 59 38 ...
##  $ Family           : Factor w/ 4 levels "1","2","3","4": 3 4 1 2 2 4 1 3 3 1 ...
##  $ CCAvg            : num  0.4 1.3 5.4 2.8 0 3.6 0.1 2.9 0.9 1.5 ...
##  $ Education        : Factor w/ 3 levels "1","2","3": 1 2 1 1 3 3 2 3 3 2 ...
##  $ Mortgage         : int  0 0 0 179 0 0 199 0 199 116 ...
##  $ Personal.Loan    : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 2 1 1 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ CD.Account       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Online           : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 2 1 ...
##  $ CreditCard       : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 2 1 1 2 ...
##  $ Experience2      : num  0.538 0.613 0.04 0.594 0.636 ...
```

```
# Setting up for PCA then LDA
LDA_train = select_if(train, is.numeric) %>% mutate(Personal.Loan = train$Personal.Loan)
str(LDA_train)
```

```
## 'data.frame':    3500 obs. of  5 variables:
##  $ Income       : int  23 52 98 79 95 63 91 143 59 38 ...
##  $ CCAvg        : num  0.4 1.3 5.4 2.8 0 3.6 0.1 2.9 0.9 1.5 ...
##  $ Mortgage     : int  0 0 0 179 0 0 199 0 199 116 ...
##  $ Experience2  : num  0.538 0.613 0.04 0.594 0.636 ...
##  $ Personal.Loan: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 2 1 1 ...
```

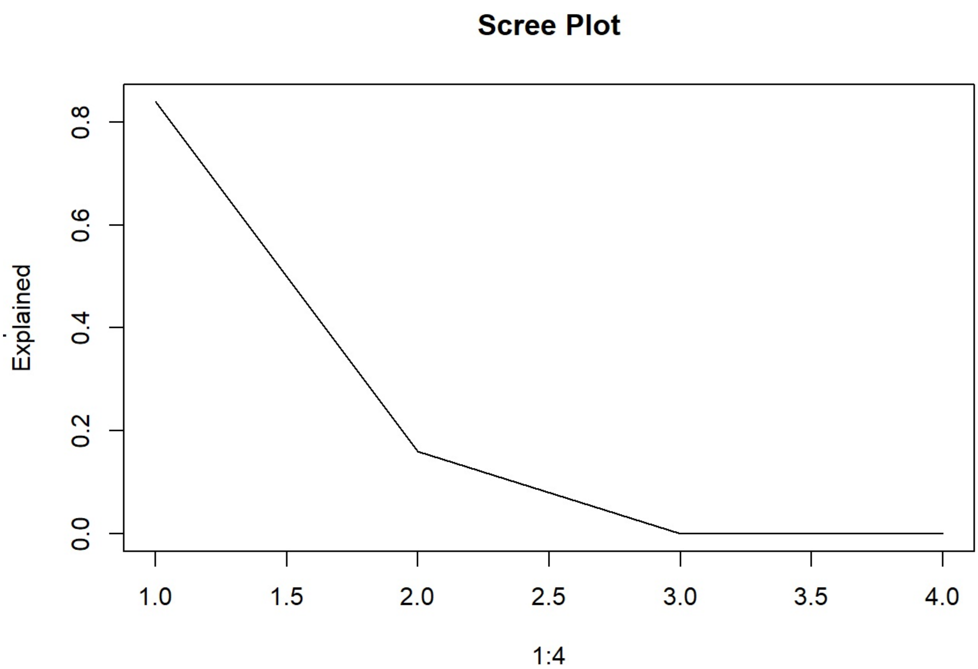
```
pairs(LDA_train)
```



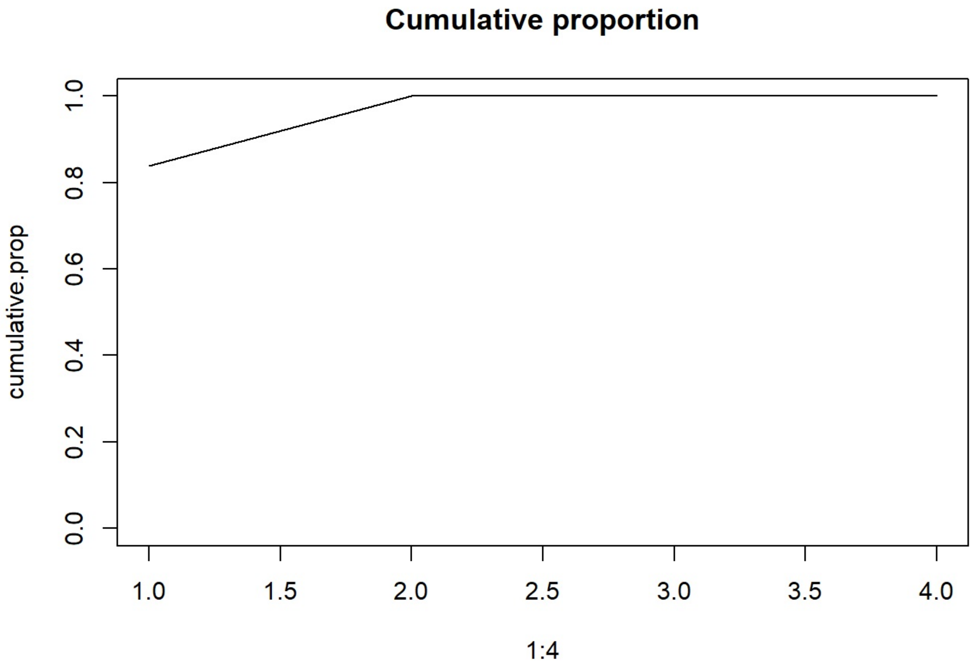

```
# PCA
reduced = LDA_train[-c(5)]
pc.result<-prcomp(reduced,scale=FALSE)
eigenvals<-(pc.result$sdev)^2
eigenvals
```

```
## [1] 1.038109e+04 1.988352e+03 1.730318e+00 2.731713e-02
```

```
plot(1:4,eigenvals/sum(eigenvals),type="l",main="Scree Plot",ylab="Prop. Var.
Explained")
```




```
cumulative.prop<-cumsum(eigenvals/sum(eigenvals))
plot(1:4,cumulative.prop,type="l",main="Cumulative proportion",ylim=c(0,1))
```



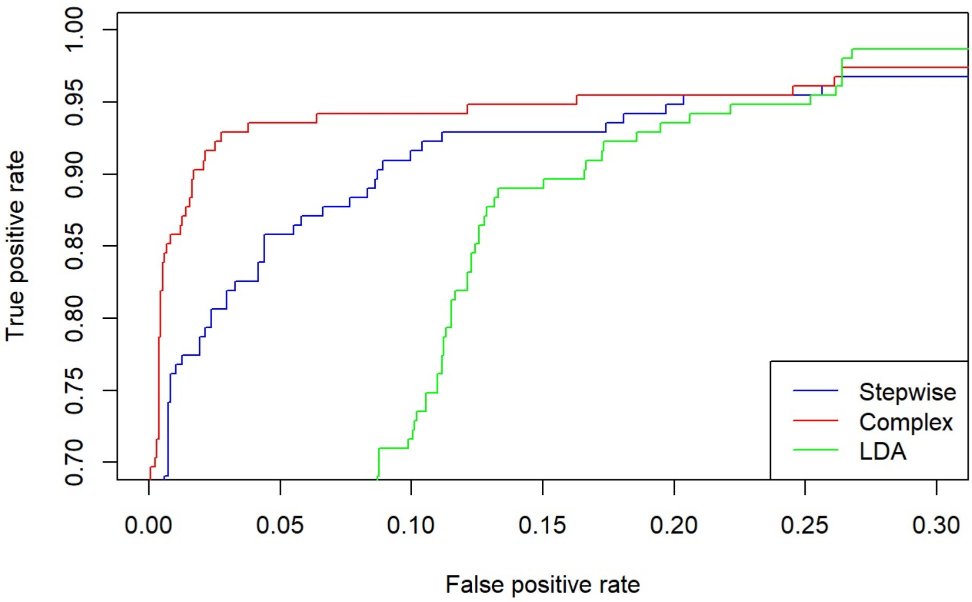

```
par(mfrow=c(1,1))

# The desired number of PCs looks to be 1, since 2 retains 0% of the total variation.

# Build Model
LDA.model = lda(Personal.Loan ~ ., LDA_train)
LDA.model
```

```
## Call:
## lda(Personal.Loan ~ ., data = LDA_train)
##
## Prior probabilities of groups:
##         No        Yes
## 0.90714286 0.09285714
##
## Group means:
##        Income    CCAvg  Mortgage Experience2
## No   66.27969 1.738101  50.66677   0.4069236
## Yes 143.66154 3.803631 103.09846   0.3945928
##
## Coefficients of linear discriminants:
##                    LD1
## Income      0.02262327
## CCAvg       0.07065000
## Mortgage    0.00119106
## Experience2 0.01861669
```

```
# Criteria
fit.p<-predict(LDA.model, newdata=test)
str(fit.p)
```

```
## List of 3
##  $ class    : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ posterior: num [1:1500, 1:2] 0.937 0.997 0.998 0.973 0.998 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:1500] "4" "6" "8" "9" ...
##   .. ..$ : chr [1:2] "No" "Yes"
##  $ x        : num [1:1500, 1] 0.586 -0.997 -1.344 0.132 -1.259 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:1500] "4" "6" "8" "9" ...
##   .. ..$ : chr "LD1"
```

```
# ROC Curves
results.model<-prediction(fit.p$posterior[,2], test$Personal.Loan,label.ordering=c("No","Yes"))
roc.lda_= performance(results.model, measure = "tpr", x.measure = "fpr")

plot(roc.step, col = "blue", xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.model2a, col = "red", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
plot(roc.lda_, col = "green", add = TRUE, xlim = c(0, 0.3), ylim = c(0.7, 1.0))
legend("bottomright", legend = c("Stepwise", "Complex", "LDA"),
       col = c("blue", "red", "green"),
       lty=1, lwd=1)
```




```
#fake<-train
#fake$Personal.Loan<-sample(fake$Personal.Loan,3500,replace=F)
#LDA.model.fake = lda(Personal.Loan ~ ., fake)
#LDA.model.fake

# Universally compare accuracy
confusionMatrix(class.step, test$Personal.Loan)$overall[1]
```

```
## Accuracy
##    0.968
```

```
confusionMatrix(class.final2a, test$Personal.Loan)$overall[1]
```

```
## Accuracy
##    0.972
```

```
round((mean(predict(LDA.model,newdata=test)$class==test$Personal.Loan)),3)
```

```
## [1] 0.899
```