ALY 6015: Intermediate Analytics

Final Project: Draft Report

INSTRUCTOR NAME: **Nina R. Nasab** Date: **May 17, 2023**

Group Members:

Clyde Antonio Dias Do Rosario (NUID:002686677)

Raja Raghu Ram Pydi (NUID:002637033)

Shreyansh Chandrakant Bhalodiya (NUID:002664707)

Surpreet Kaur Mann (NUID:002660066)

Introduction

In this assignment, we have worked on the attrition dataset to develop models to predict the attrition and monthly income variables using advanced data modeling techniques. We have utilized excel and R programming to successfully conduct an exploratory data analysis and to create advanced logistic and linear regression models. Lastly, to illustrate our findings, we have attached code outputs and visualizations generated in the process.

About the Dataset

The attrition dataset describes employees if they left or retained in the company along with variables such as their gender, age, income, working hours and so on. Below are a few important points to consider before moving forward.

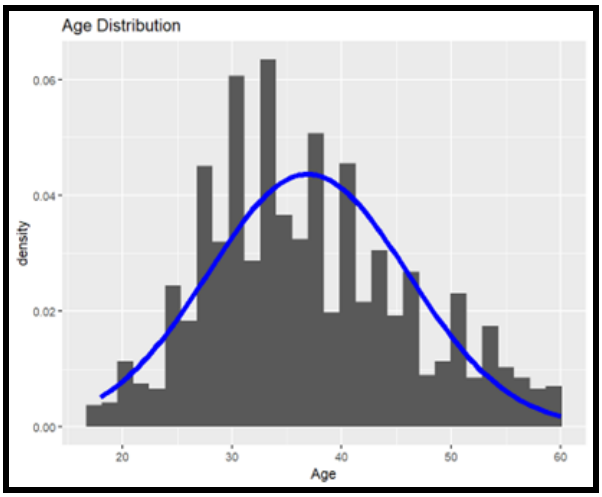
- Dataset Structure: The dataset has 20 variables and 1470 observations
- Data types: the dataset has a combination of only factors and numeric variables
- Missing Values: there are no missing values in the dataset
- Target Variables:
    - Attrition: We will create a logistic regression model using the attrition variable as the dependent variable to predict whether an employee left or retained in the company
    - MonthlyIncome: We will create a linear regression model using the MonthlyIncome variable as the dependent variable to predict the income of an employee.
- Attrition Imbalance: 1237 (84%) of the employees retained in the company whereas 237 (16%) did leave, creating an imbalance in the dataset. Necessary steps will be taken to approach this imbalance.

Business Problem Trying To Solve

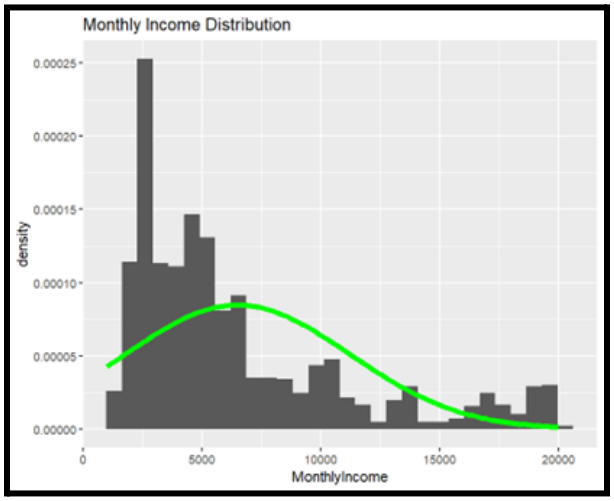
- Find major reasons responsible for the attrition rate of the company.
- Building a model using all the given variables predicting attrition of the company.
- Building another model predicting monthly income of the employees.

Exploratory Data Analysis

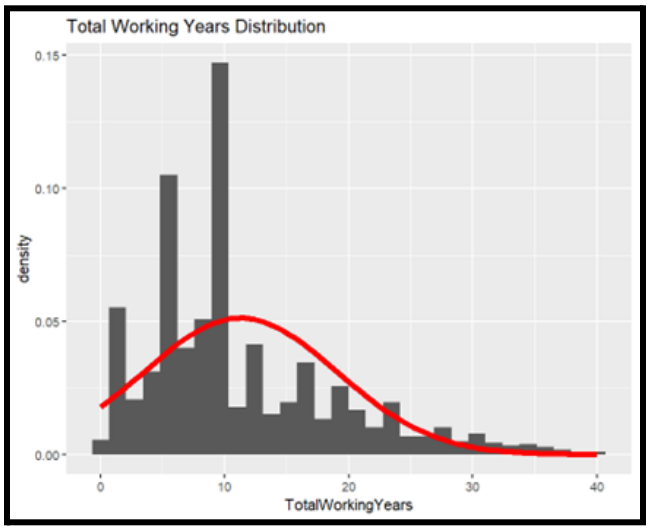
Univariate Analysis:



The age variable ranges from 18 to 60 years. The average age in the dataset is 36.92. Further from the visualization we can observe an almost normal distribution with slight positive skewness of 0.41.



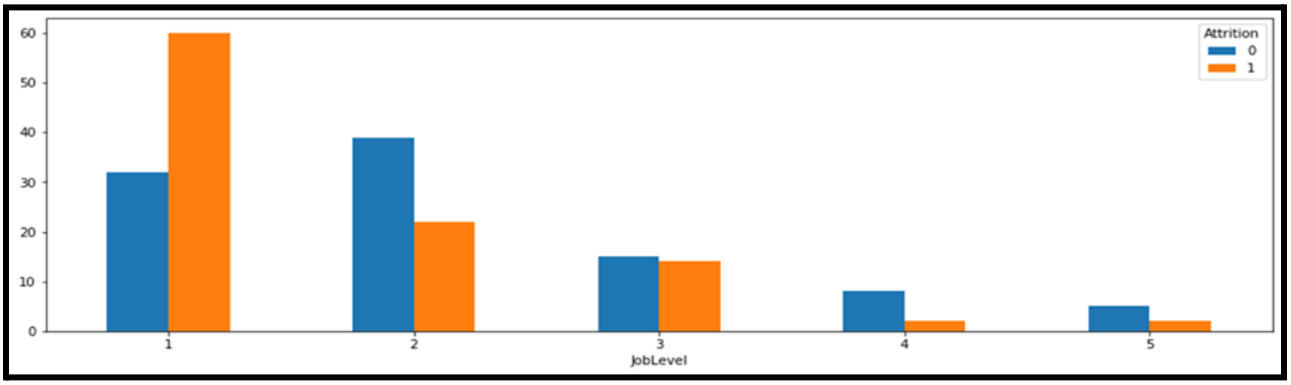
The monthly income variable ranges from 1009 to 19999. The average monthly income of individuals in the organization is 6503. The distribution of the variable is visible skewed to the right. This is evidently because the total number of employees under low-income groups are much higher than high income groups.



The minimum and maximum number of years worked by individuals in the dataset is 0 to 40 years. The average total working years is 11.28. The distribution of the variable is positively skewed.

Bivariate Analysis:

- JOB LEVEL BARPLOT BY ATTRITION:



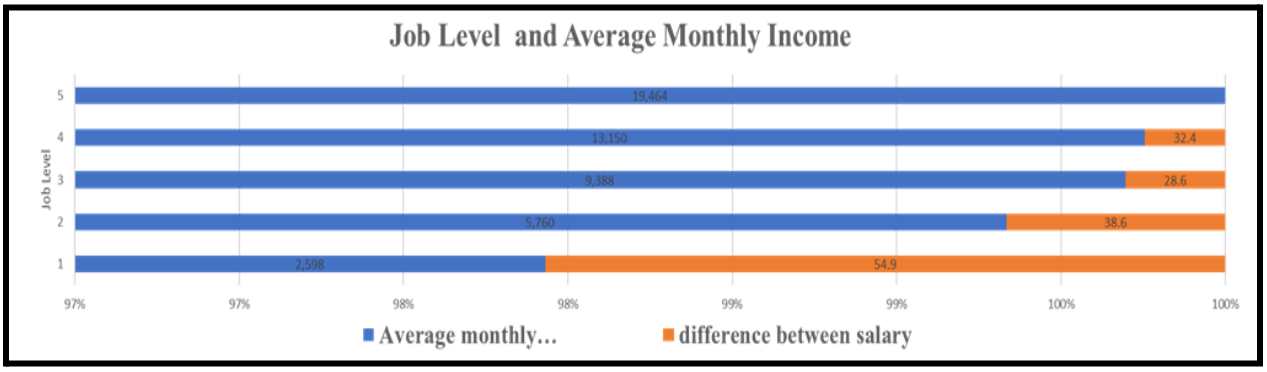
The visualization above is a barplot of the job level variable categorized by attrition. From the visualization we can immediately see that the total number of employees start reducing at higher job levels. The attrition ratio among lower job levels are much higher comparatively to higher job levels. From this we can infer that employees have a higher tendency to leave the organization at lower job levels.

· MEAN DIFFERENCES CATEGORIZED BY ATTRITION

Row Labels	Average of Years At Company	Average of Monthly	Average of Distance	Average of Years Since Last Promotion	Average of TotalWorking	Average of Years With Company	Average of WorkLife
0	4.24	2854	8.33	1.21	6.26	2.69	2.73
1	3.08	2598	9.48	1.15	4.85	1.89	2.67

The table above illustrates the difference in means of a few variables that are categorized by attrition. This table is useful to identify potential predictor variables to predict the attrition variable. For example, the worklife variable does not have much of a difference in the mean values categorized by attrition. Therefore, it would be hard for the worklife variable to explain the variation between employees who left or retained in the organization. On the other hand, variables like monthly income, total working hours and years at the company can be potential predictors as the difference in the means indicate its ability to explain the variation in the attrition variable.

· JOB LEVEL BAR PLOT BY MONTHLY INCOME



The visualization above is a horizontal bar plot of the job level variable categorized by monthly income. From the visualization we can see the percentage increase of the average monthly income at every job level. For example, at job level 1 the average salary is 2598 and the difference increase to job level 2 is 54.9. Further, we can also observe that the average difference reduces at higher job levels.

Methods

MODEL 1: **Logistic Regression and Xgboost**

DEPENDENT VARIABLE: ATTRITION

We are predicting attrition using a logistic regression method. We have 1 and 0 values as two categories with attrition yes and another one with attrition no. Current ratio for 0 and 1 is 1233:237 which is 83:17 hence our data is bit unbalanced hence we will first try to make a good balance between 0 and 1 and will pick a sample from 0 of 900 values and than our new ratio will be 900:237 that is 79:21 which is good ratio for building logistic and xgboost regression.

ONE-HOT ENCODING/CREATING DUMMIES

data <- model.matrix(~.-1, data = data): This line performs one-hot encoding on the "data" dataframe, creating dummy variables for the categorical columns. The formula ~.-1 specifies that all columns except the intercept should be used for encoding.

TRAIN-TEST SPLIT

- We are using set.seed(999) sets a seed for the random number generator, ensuring reproducibility of results.
- train_index <- createDataPartition(data$Attrition, p=0.70, list=FALSE) creates an index for splitting the data. The createDataPartition function from the caret package is used, considering the "Attrition" column, and the parameter p=0.70 specifies a 70% portion for the training set.
- train <- data[train_index,] creates the training set by subsetting the original dataset (data) using the train_index created in the previous step.
- test <- data[-train_index,] creates the test set by selecting the remaining portion of the original dataset (data) that was not included in the training set.

FITTING LOGISTIC REGRESSION

model_1 <- glm(Attrition~., data=data, family = binomial(link = 'logit')): This line fits a logistic regression model (glm) using the "Attrition" column as the response variable and all other columns in the dataset (represents all predictors) as the explanatory variables. The family = binomial(link = 'logit') specifies the binomial family with a logit link function for logistic regression.

PREDICTING MODEL AND CONVERTING PROBABILITIES.

probs_train <- predict(model_1, newdata=train, type='response'): This line generates predicted probabilities for the training data (train) using the logistic regression model (model_1).

predict <- as.factor(ifelse(probs_train >= 0.25, 1, 0)): This line assigns predicted classes based on a threshold of 0.25. If the predicted probability (probs_train) is greater than or equal to 0.25, it is classified as 1, otherwise as 0. The as.factor function converts the resulting values into a factor variable.

RESULTS

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 509  44
         1 121 122

               Accuracy : 0.7927
                 95% CI : (0.7629, 0.8204)
    No Information Rate : 0.7915
    P-Value [Acc > NIR] : 0.486

                  Kappa : 0.4637

 Mcnemar's Test P-Value : 3.287e-09

            Sensitivity : 0.8079
            Specificity : 0.7349
         Pos Pred Value : 0.9204
         Neg Pred Value : 0.5021
             Prevalence : 0.7915
         Detection Rate : 0.6394
   Detection Prevalence : 0.6947
      Balanced Accuracy : 0.7714

       'Positive' Class : 0
```

Train

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 220  16
         1  50  55

               Accuracy : 0.8065
                 95% CI : (0.7605, 0.847)
    No Information Rate : 0.7918
    P-Value [Acc > NIR] : 0.277

                  Kappa : 0.501

 Mcnemar's Test P-Value : 4.865e-05

            Sensitivity : 0.8148
            Specificity : 0.7746
         Pos Pred Value : 0.9322
         Neg Pred Value : 0.5238
             Prevalence : 0.7918
         Detection Rate : 0.6452
   Detection Prevalence : 0.6921
      Balanced Accuracy : 0.7947

       'Positive' Class : 0
```

Test

- Clearly we got decent results for our logistic regression model. Talking about the confusion matrix we got 509 for train and 220 test results which were actually 0 and got predicted as 0 which is our true positive for train and test results.
- In the similar way we have 122 for train and 55 for tests which were actually 1 and predicted as 1 which are our true positive results.
- If we look at our false positive and false negative values we can find using recall and precision rates. Specificity rate is decent enough for our train and test results.
- Overall we got accuracy around 80% and balanced accuracy around 78%.
- We got the ROC value as 0.86 which indicates the performance of a binary classification model in terms of its ability to discriminate between the two classes.

XGBOOST MODEL FITTING

XGBoost (Extreme Gradient Boosting) is a popular machine learning algorithm known for its high performance and accuracy in various prediction tasks. Here's an explanation of the provided code line:

model_xgboost <- xgboost(data = data.matrix(train_m), label = train_y, max.depth = 6, eta = 2, nthread = 4, nrounds = 2, objective = "binary:logistic")

- xgboost: This function is used to train an XGBoost model.data.matrix(train_m). The data.matrix function converts the training dataset (train_m) into a matrix format that XGBoost expects as input.
- label = train_y: This specifies the response variable (train_y) that the XGBoost model will try to predict.
- max.depth = 6: It sets the maximum depth of each tree in the boosting process. Increasing this value can make the model more complex and potentially overfit the data.
- eta = 2: The eta (learning rate) parameter controls the step size shrinkage at each boosting iteration. A higher learning rate allows for faster learning but may lead to overfitting.

- nthread = 4: This sets the number of threads or cores to be used for parallel processing during training.
- nrounds = 2: It determines the number of boosting iterations or rounds to be performed during model training.
- objective = "binary:logistic": The objective function specifies the loss function to be optimized. In this case, "binary:logistic" indicates that XGBoost is used for binary classification, optimizing the logistic loss.

RESULTS

The accuracy of the model is 0.7135, indicating that approximately 71.35% of the predictions are correct. Sensitivity (true positive rate) is 0.7824, indicating that 78.24% of the actual positive cases are correctly identified, while specificity (true negative rate) is 0.4875, indicating that only 48.75% of the actual negative cases are correctly identified.

The positive predictive value (PPV) is 0.8333, indicating that 83.33% of the predicted positive cases are accurate. The negative predictive value (NPV) is 0.4062, indicating that only 40.62% of the predicted negative cases are accurate.

The Kappa statistic, measuring agreement beyond chance, is 0.2524, suggesting fair agreement. Overall, the model's performance appears to have limitations, as indicated by the modest accuracy and relatively low specificity and NPV values. Further analysis and potentially model improvements may be needed to enhance the predictive performance.

MODEL 2 :LASSO Regression

We are using lasso regression to predict the monthly income. We will initially fit all the variables to fit and then the model will find the best performing coefficients. We have a log of minimum lambda value as 4.15 and log of lambda 1se as 5.46. We will fit both the models and compare the coefficient results we got.

```
> coef(model.min)
28 x 1 sparse Matrix of class "dgCMatrix"
                                        s0
(Intercept)                     -1801.771862
Age                                     .
Attrition                               .
`BusinessTravelNon-Travel`              .
BusinessTravelTravel_Frequently         .
BusinessTravelTravel_Rarely           112.016745
`DepartmentResearch & Development`      .
DepartmentSales                      -200.866130
`EducationFieldLife Sciences`           .
EducationFieldMarketing                 .
EducationFieldMedical                   .
EducationFieldOther                     .
`EducationFieldTechnical Degree`        .
EnvironmentSatisfaction                 .
GenderMale                              .
JobInvolvement                          .
JobLevel                             3804.802034
JobSatisfaction                         .
MaritalStatusMarried                    .
MaritalStatusSingle                     .
OverTimeYes                             .
TotalWorkingYears                     43.134427
WorkLifeBalance                         .
YearsAtCompany                          .
YearsInCurrentRole                      .
YearsSinceLastPromotion                 .
YearsWithCurrManager                  -1.588088
DistanceFromHome                      -6.678437
```

```
> coef(model.1se)
28 x 1 sparse Matrix of class "dgCMatrix"
                                        s0
(Intercept)                     -1515.85422
Age                                     .
Attrition                               .
`BusinessTravelNon-Travel`              .
BusinessTravelTravel_Frequently         .
BusinessTravelTravel_Rarely             .
`DepartmentResearch & Development`      .
DepartmentSales                         .
`EducationFieldLife Sciences`           .
EducationFieldMarketing                 .
EducationFieldMedical                   .
EducationFieldOther                     .
`EducationFieldTechnical Degree`        .
EnvironmentSatisfaction                 .
GenderMale                              .
JobInvolvement                          .
JobLevel                             3696.91963
JobSatisfaction                         .
MaritalStatusMarried                    .
MaritalStatusSingle                     .
OverTimeYes                             .
TotalWorkingYears                     34.01656
WorkLifeBalance                         .
YearsAtCompany                          .
YearsInCurrentRole                      .
YearsSinceLastPromotion                 .
YearsWithCurrManager                    .
DistanceFromHome                        .
```

Overall, when comparing the two models' coefficients, it appears that **model.1se** has less no of predictors as lasso regression minimized the predictors to zero. Hence it is clear that total working years and job level in this model has a very high correlation with monthly income. We got excellent results by just using these two variables as independent.

```
> eval_results(train_y, preds.train , train_x)
      RMSE   Rsquare
1 1474.784 0.9080536
> eval_results(test_y, preds.test , test_x)
     RMSE   Rsquare
1 1337.98 0.9028541
>
```

The evaluation results for the train and test datasets provide insights into the performance of the model in predicting attrition.

Train Set:

- RMSE (Root Mean Squared Error): The RMSE value of 1474.784 indicates the average difference between the actual attrition values and the predicted values on the train dataset.
- R-square: The R-square value of 0.9080536 represents the proportion of the variance in the train dataset's attrition that is explained by the model. The model accounts for approximately 90.8% of the variance, indicating a good fit to the train dataset.

Test Set:

- RMSE (Root Mean Squared Error): The RMSE value of 1337.98 represents the average difference between the actual attrition values and the predicted values on the test dataset. Similar to the train dataset, a lower RMSE indicates better performance in accurately predicting attrition. The model has a slightly lower RMSE on the test dataset compared to the train dataset, suggesting that it generalizes well to unseen data.
- R-square: The R-square value of 0.9028541 indicates the proportion of the variance in the test dataset's attrition that is explained by the model. This indicates that the model performs well in capturing the patterns and trends in the test dataset.

Conclusion

In our project, we performed deep exploratory data analysis (EDA) on an attrition dataset and developed models for predictions. We fitted two linear regression models, namely Lasso regression and logistic regression, along with XGBoost. The predictor variable used in both the XGBoost and logistic regression models was attrition, while the predictor variable for the Lasso regression model was monthly income.

Through in-depth exploratory data analysis (EDA), we identified several significant factors contributing to attrition within the dataset. Our findings revealed that the primary reasons for attrition were as follows: Firstly, approximately 66.6% of the individuals experiencing attrition were at job level 1, indicating that lower-level positions were more susceptible to turnover.

Additionally, a noteworthy observation was that the maximum number of iterations occurred when an employee had been with the company for less than or equal to 2 years. Another contributing factor was the marital status, with singles predominantly working at job level 1 and married individuals at job level 3. Other influential factors included overtime, monthly income, and the number of years under the current manager. These insights shed light on the key drivers behind attrition and can guide organizations in taking proactive measures to address these factors, improve employee satisfaction, and mitigate turnover.

Based on the available information, our models achieved an accuracy of approximately 80% for both XGBoost and logistic regression. However, it is worth noting that the specificity, a measure of correctly identifying non-attrition cases, ranged between 55% to 70%. This suggests that the models were relatively better at predicting non-attrition cases than attrition cases.

In the case of Lasso regression, the R-square value was notably high at 90%, indicating that around 90% of the variance in monthly income could be explained by the predictor variables used in the model. In conclusion, our models demonstrated reasonable accuracy in predicting attrition, but they exhibited lower specificity. Additionally, the Lasso regression model showed strong predictive power for monthly income. Further analysis and evaluation, such as considering other performance metrics, feature importance, and potential improvements, would be necessary for a comprehensive assessment of the models' effectiveness and reliability.

References

- *Medium*. (n.d.-b). Medium. https://towardsdatascience.com/ridge-regression-l2 -regularization -9e8a3ac7f3c
- Biswal, A. (2023). The Complete Guide on Overfitting and Underfitting in Machine Learning. *Simplilearn.com*. https://www.simplilearn.com/tutorials/machine-learning-tutorial/overfitti ng-and-underfitting
- *Regression Analysis: Step by Step Articles, Videos, Simple Definitions*. (2023, February 22). Statistics How To. https://www.statisticshowto.com/probability-and-statistics/regression-a nalysis/
- *Regularization — ML Glossary documentation*. (n.d.). https://ml-cheatsheet.readthedo cs.io/en/latest/regularization.html
- Thieme, C. (2022, March 30). Identifying Outliers in Linear Regression — Cook's Distance. *Medium*. https://towardsdatascience.com/identifying-outliers-in-linear-regression-cooks-di stance-9e212e9136a

Appendix

```
# Final Project: Initial Analysis Report

# Exploratory Data Analysis

data <- read.csv("attrition.csv", header=TRUE,sep=",")

library(psych)


head(data)

summary(data)

str(data)

describe(data)


which(is.na(data))

# No missing values in the dataset


table(data$Attrition)

# Attriton variable is imbalanced with 1233 0's and 237 1's


hist(data$Age)

hist(data$TotalWorkingYears)

hist(data$MonthlyIncome)


data$Gender <- as.factor(data$Gender)

library(tidyverse)


data%>%
```

```
  ggplot(aes(x=MonthlyIncome, fill=Gender, color=Gender))+
  geom_density(alpha=0.4)+
  theme_bw()+
  labs(title = "DENSITY GRAPH OF INCOME BY GENDER")


library(corrplot)
library(RColorBrewer)


#creating subset with continous variables


data_cont <- subset(data,
select=c(Age,MonthlyIncome,TotalWorkingYears,


YearsAtCompany,YearsInCurrentRole,YearsSinceLastPromotion,


YearsWithCurrManager,DistanceFromHome))


cors <- cor(data_cont, use="pairwise")
corrplot(cors, type="upper", col=brewer.pal(n=8, name="RdYlBu"))


##########################################
install.packages('caret')
library(caret)


set.seed(999)
train_index <- createDataPartition(data$Attrition, p=0.70,
list=FALSE)
train <- data[train_index,]
```

```
test <- data[-train_index,]


model_1 <- glm(Attrition~., data=data, family = binomial(link =
'logit'))

summary(model_1)

coef(model_1)

exp(coef(model_1))


# Making predictions and confusion matrix


# Train

probs_train <- predict(model_1, newdata=train, type='response')

probs_train

predict <- as.factor(ifelse(probs_train >= 0.5,1,0))

predict

confusionMatrix(predict, as.factor(train$Attrition))


# Test

probs_test <- predict(model_1, newdata=test, type='response')

probs_test

predict_test <- as.factor(ifelse(probs_test >= 0.5,1,0))

confusionMatrix(predict_test,as.factor(test$Attrition))


str(predict_test)


# ROC plot and AUC
```

```
library(pROC)
roc_1 <- roc(test$Attrition,probs_test)
plot(roc_1, col='purple', main="ROC")
auc(roc_1)

#### LASSO REGRESSION

install.packages("psych")

install.packages("caret")

install.packages("glmnet")

install.packages("corrplot")

install.packages("RColorBrewer")

install.packages("Metrics")

library(psych)
library(Metrics)
library(psych)
library(glmnet)
library(Metrics)

# Final Project: Initial Analysis Report
```

```
# Exploratory Data Analysis


data <- read.csv("attrition.csv", header=TRUE,sep=",")


head(data)

summary(data)

str(data)

#describe(data)



##########################################


class(data)

# Convert categorical variables to factors

categorical_cols <- c("BusinessTravel", "Department",
"EducationField", "Gender", "MaritalStatus", "OverTime")

data[categorical_cols] <- lapply(data[categorical_cols],
as.factor)


# Perform one-hot encoding

data <- model.matrix(~.-1, data = data)


data <- as.data.frame(data)


head(data)
```

```
df <- data


##########################################

# Split data into train and test sets

##########################################

# seed() function in R is used to create reproducible results
when writing code that involves creating variables

# that take on random values. By using the set. seed() function,
you guarantee that the same random values are produced

# each time you run the code. # The value inside the seed()
function does not matter

set.seed(123)

trainIndex <- sample(x = nrow(df), size = nrow(df) * 0.7)



train <- df[trainIndex,]

test <- df[-trainIndex,] # Returns all rows that are not in
training set


head(train)



train_x <- model.matrix(MonthlyIncome ~ ., train)[,-1]

test_x <- model.matrix(MonthlyIncome ~ ., test)[,-1]

# model.matrix is more advanced than just matrix function.

# E.g. Model.matrix automatically converts categorical variables
to dummy variables



head(train_x)

class(train_x)
```

```
colnames(train_x)


train_y <- train$MonthlyIncome

test_y <- test$MonthlyIncome


# Find best values of lambda

############################################################

# Find the best lambda using  corss-validation

set.seed(123)

#?cv.glmnet

cv.lasso <- cv.glmnet(train_x, train_y, alpha = 1, nfolds = 10)
# alpha = 1 Lasso

plot(cv.lasso) # Values on top shows the number of non-zero
coefficients

# Red dots are error estimates with their confidence interval

# dotted line on the left side represents the minimum value of
Lambda; which retains

#all 15 parameters

# dotted line on the right side represents maximum value within
one standard error

#of the minimum; it has 9 non-zero coefficients in the model. So
it sets the coefficient of

# one of the parameters to zero.


# lambda min - minimizes out of sample loss

# lambda 1se - largest value of lambda within 1 standard error
of lambda min

log(cv.lasso$lambda.min)
```

```
log(cv.lasso$lambda.1se)


cv.lasso$lambda.min

cv.lasso$lambda.1se


cv.lasso$cvm # Returns Mean Square Error


plot(cv.lasso$cvm ~ cv.lasso$lambda)

plot(cv.lasso$cvm ~ log(cv.lasso$lambda))


############################################################

# Fit models based on lambda

############################################################

# Fit the final model on the training data using lambda.min

# alpha = 1 for Lasso (L2)

# alpha = 0 for Ridge (L1)


model.lasso <- glmnet(train_x, train_y, alpha = 1)

model.lasso


model.min <- glmnet(train_x, train_y, alpha = 1, lambda =
cv.lasso$lambda.min)

model.min


# Display regression coefficients

coef(model.min)
```

```
# fit the final model on training data using lambda.1se

model.1se <- glmnet(train_x, train_y, alpha = 1, lambda =
cv.lasso$lambda.1se)

model.1se


# Display regression coefficients

coef(model.1se)


# Make predictions on the test data using lambda.min

preds.train <- predict(model.1se, newx = train_x) #
predict.glmnet

train.rmse <- rmse(train_y, preds.train) # 7.456


preds.test <- predict(model.1se, newx = test_x)

test.rmse <- rmse(test_y, preds.test)


# Compare rmse values

train.rmse

test.rmse


# Make predictions on the test data using lambda.min

preds.train <- predict(model.min, newx = train_x) #
predict.glmnet

train.rmse <- rmse(train_y, preds.train)

preds.test <- predict(model.min, newx = test_x)

test.rmse <- rmse(test_y, preds.test)


# Compare rmse values
```

```
train.rmse
test.rmse


eval_results <- function(true, predicted, df) {
  SSE <- sum((predicted - true)^2)
  SST <- sum((true - mean(true))^2)
  R_square <- 1 - SSE / SST
  RMSE = sqrt(SSE/nrow(df))



  # Model performance metrics
  data.frame(
    RMSE = RMSE,
    Rsquare = R_square
  )
}
#Evaluating results of test set
eval_results(test_y, preds.test , test_x)

#Evaluating results of train set
eval_results(train_y, preds.train , train_x)
```