2020

Fraudulent Payments at Acme Corp

A CASE STUDY

PREPARED BY SCOTT R. SILER FOR TEKPARTNERS & POINT B

Introduction

The purpose of this document is to highlight three elements: 1) my ability to address a business problem through statistical and machine learning frameworks, 2) my coding skills, and 3) my ability to communicate complicated material in a succinct, meaningful way. The data used in this is case study was randomly generated and the subsequent analysis is my own.

Situation

Acme Corp is experiencing issues with fraudulent payments. It has identified a random set of payments from its system and Acme Corp's Mr. Manager has a hunch the fraudulent payments are associated with invalid email accounts. He believes his hunch to be correct and wants to build a model to flag payments associated with an invalid email address. Before he can make his case to his boss, Mr. Manager asks you to investigate. He provides you a .csv file of payment transactions which includes the payment id, first name, last name, email address, whether the email is valid (1) or invalid (0), and whether payment was fraudulent (1) or not fraudulent (0). You are asked to determine if a relationship exists between the invalid email accounts and the fraudulent payments. In other words, using the data provided, *determine if an invalid email a good predictor of a fraudulent payment*?

Investigating the Data

To investigate this problem, you decide you will use R and R Studio. You open R Studio and start by loading your regular packages and libraries and set up your working directory on your desktop.

```
pacman::p_load(dplyr, plyr, tidyr, stringr, splitstackshape, gsheet, purrr,lubridate, readxl, forecast, tibble, sqldf, ggplot2, caret, mlbench)

wd <- c("C:/Users/siles/C:/Users/siles/Desktop/DSKTOP/JOB/TekPartners")
```

The name of the .csv file is AML_Example. You read it into your global environment.

```
AML_Example <- read.csv("C:/Users/siles/Desktop/DSKTOP/JOB/TekPartners/AML_Example.csv", header=T, sep=",", stringsAsFactors = FALSE)
```

You make a comment to clarify the purpose of this investigation.

```
#######-Is having an invalid email address a good predictor of a fraudulent payment?-######
```

You start by examining the data using the *View, names, str* and *summary* functions. From the *View* function you can see the data is already "tidy" in that each row is an observation (i.e. a payment as designated by PMT_ID) and each column is a variable.

```
View(AML_Example)
names(AML_Example)
```

```
[1] "PMT_ID"      "First_Name"  "Last_Name"   "Email"       "Valid_Email" "Valid_PMT"
```

```
str(AML_Example)
```

```
'data.frame':   932 obs. of  6 variables:
 $ PMT_ID     : int  39759 12142 28106 29822 59359 95784 32394 4779 16307 11636 ...
 $ First_Name : chr  "A" "B" "C" "A" ...
 $ Last_Name  : chr  "X" "Y" "Z" "X" ...
 $ Email      : chr  "A_X37tekpartners.com" "B_Y53tekpartners.com" "C_Z540tekpartners.com" "A_X390tekpartners.com" ...
```

```
 $ Valid_Email: int  1 1 1 1 1 0 1 0 1 0 ...
 $ Valid_PMT  : int  0 0 1 1 0 0 0 0 1 1 ...
```

From *names* you can see of the column heads. *Str* tells you the data set contains 932 observations and 6 variables or features. 3 of the variables (PMT_ID, Valid_Email, and Valid_PMT) are integers while the other three (First_Name, Last_Name, and Email) are characters.

You have been at Acme Corp long enough to know the observations in PMT_IDs are randomly assigned ID numbers. You also know the first name, last name, and email observations are all characters. Therefore, you only need summary statistics for the valid email and valid payments observations. In getting the summary of these two variables, you notice the mean of valid emails is 0.4828 and the mean of valid payments is 0.5118. Essentially, about 48% of emails are valid while 52% are invalid. As well, 51% of payments are valid while 49% of payments are invalid.

```
summary(AML_Example)
```

```
summary(AML_Example$Valid_Email)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0000  0.0000  0.4828  1.0000  1.0000
```

```
summary(AML_Example$Valid_PMT)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0000  1.0000  0.5118  1.0000  1.0000
```

You'd like to see the how the counts of each valid and invalid email and payment split into table form using the *table* function. You then print the results.

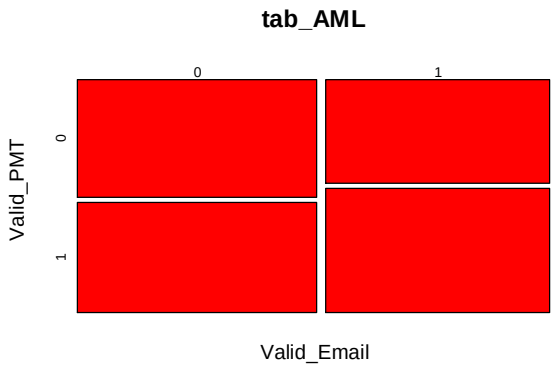
```
tab_AML <- table(AML_Example$Valid_Email, AML_Example$Valid_PMT)
print(tab_AML)
```

```
     0   1
  0 249 233
  1 206 244
```

It looks to be close to an even split which makes sense given the mean of each variable. You then examine the probability table using *prop.table().* This table simply divides the number of observations within the contingency table by the total number of observations (932) to calculate a probability. For example, if you were to randomly select an email and payment pair from the 932 observations, the probability the email would be valid but the payment invalid is 0.25 or 25%.

```
prop.table(tab_AML)

           0         1
  0 0.2671674 0.2500000
  1 0.2210300 0.2618026
```

Lastly, you want to visually examine the data. You use the *plot()* function on *tab_AML* to examine the table.



From the visual, the proportions are as you might expect. Now it's time to build a model and see if the relationship is truly significant.

Developing a Model(s)

Based off the descriptive statistical analysis you perform you are wary invalid emails are a good predictor of fraudulent payments. However, to test this null hypothesis, that invalid emails do not predict fraudulent payments, you decide to test this hypothesis with some candidate models.

Model 1 - Linear Regression Model

The first model you investigate is a simple, linear regression model. Using the *lm()* function you set Valid_PMT as your dependent or target variable and Valid_Email as your independent or feature variable. You then print the summary of the object *model.*

```
#Candidate Model 1 - Linear Regression
model <- lm(formula = Valid_PMT ~ Valid_Email, data = AML_Example)
summary(model)
```

```
Call:
lm(formula = Valid_PMT ~ Valid_Email, data = AML_Example)

Residuals:
    Min      1Q  Median      3Q     Max
-0.5422 -0.4834  0.4578  0.4725  0.5166

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.48340    0.02275  21.246   <2e-16 ***
Valid_Email  0.05882    0.03274   1.796   0.0728 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4995 on 930 degrees of freedom
Multiple R-squared:  0.003458,Adjusted R-squared:  0.002386
F-statistic: 3.227 on 1 and 930 DF,  p-value: 0.07277
```

Two things immediately draw your attention. One is the Residual Standard Error (RSE) which is .4995. Simply put this means the error of the model's estimation is very close to the average of 0 and 1 – the only values within each of the variables. Second, the Multiple R-Squared and the Adjusted R-Squared is approximately 0.3% and 0.2% respectively. This indicates that almost none of the variability for Valid_PMT is explained by the variability of Valid_Email. In short, invalid email is not

a good predictor of fraudulent payment as result of this model as the model is no better than chance (i.e. 50/50) at predicting if a payment is fraudulent based off the email being invalid.

Model 2 - Logistic Regression Model

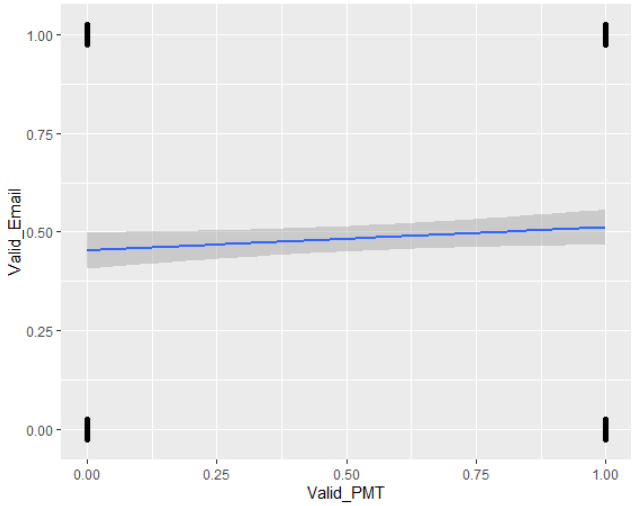
Knowing the data is both categorical and binomial, you next try glm(), a logistic regression model. Using similar syntax to the linear regression model you run the logistic regression and inspect its summary.

```
#Candidate Model 2 - Logistic Regression
gmodel <- glm(formula = Valid_PMT ~ Valid_Email, data = AML_Example, family = binomial)
summary(model)
```

Again, in looking at the summary of the logistic regression model, you see that the p-Value is the same as your linear regression model and despite changes in your estimate, standard error, and residuals. As well, when using the plot() function on your model, you can see the graphs returned look exactly like those generated by your linear model.

```
Call:
glm(formula = Valid_PMT ~ Valid_Email, family = binomial, data = AML_Example)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.250  -1.149   1.106   1.131   1.206

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.06641    0.09115  -0.729   0.4662
Valid_Email  0.23571    0.13138   1.794   0.0728 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1291.5  on 931  degrees of freedom
Residual deviance: 1288.3  on 930  degrees of freedom
AIC: 1292.3
```

A visual indication ofa this model, while different, fairs no better than the previous in determining invalid email as a predictor of fraudulent payment.



Model 3 – K-Nearest Neighbors (KNN) Model

Despite the lackluster results from the linear logistics regression model you decide to test one more model to see if you gain any additional lift or information gain from the data set Mr. Manager provides you. In this case you use a specific classification model – the K Nearest Neighbors (KNN) model. You know that KNN is used primarily to perform supervised training and offers compelling visuals as to how well your classification model performs.[1]

In using KNN, you first need to develop training and test sets for each of your candidate models. To do this you need to hold out a certain percentage of the overall population (a holdout sample) so you can eventually test your champion model (the best candidate model) on data the model has never seen before. As such you determine you will try three different training sets, each a different proportion of the overall data set you are provided. You will then test against the remaining data (excluding your holdout sample) and see which model performs best. You will then use the best model of the three to test against the hold out sample and see how well said model would classify a fraudulent payment given an invalid email address.

Model	Training Data	Test Data	Hold Out
pred1	20%	60%	20%
pred2	40%	40%	20%
pred3	50%	30%	20%

You start by creating comments to organize your labels, training data, test data, and prediction models as well as importing the library "class" for classification.

```
#Candidate Model KNN - K Nearest Neighbors
library(class)
```

You then write your code to train and test your models based on the table above.

```
######Candidate Model KNN 1
train_labels1 <- train1$Valid_Email == 'Invalid'
train_labels2 <- train2$Valid_Email == 'Invalid'
train_labels3 <- train3$Valid_Email == 'Invalid'

train1 <- AML_Example[1:186, ]
test1 <- AML_Example[187:746, ]
pred1 <- knn(train = train1[,c(5,6)], test = test1[,c(5,6)], cl = train_labels1, k = 1)
table(pred1, test1$Valid_PMT)

######Candidate Model KNN 2
train2 <- AML_Example[1:372, ]
test2 <- AML_Example[373:746, ]
pred2 <- knn(train = train2[,c(5,6)], test = test2[,c(5,6)], cl = train_labels2, k = 1)
table(pred2, test2$Valid_PMT)

######Candidate Model KNN 3
train3 <- AML_Example[1:466, ]
test3 <- AML_Example[467:746, ]
```

[1] To perform the KNN used Excel to transform the 0 and 1 populated fields with "Invalid' and "Valid" respectively.

```
pred3 <- knn(train = train3[,c(5,6)], test = test3[,c(5,6)], cl = train_labels3, k = 1)
table(pred3, test3$Valid_PMT)

#####Calculate Peformance of each Candidate model
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(table(pred1,test1$Valid_PMT))
accuracy(table(pred2,test2$Valid_PMT))
accuracy(table(pred3,test3$Valid_PMT))
```

After training and testing each of the models you determine their accuracy by calculating the percentage of times they correctly identify a payment outcome. For pred1, pred2, and pred3 their accuracy scores are 50.35%, 47.32% and 51.07% respectively. Based off this, you determine pred3 is the best model, relatively speaking, and decide to test pred3 on your holdout sample (labeled as the object "validate") to see if it performs well.

```
#####As pred3 is the Champion, validate on holdout sample to determine if pred3 should be put into production
validate = AML_Example[747:932,]
PredModel <- knn(train = train3[,c(5,6)], test = validate[,c(5,6)], cl = train_labels3, k = 1)
accuracy(table(PredModel,validate$Valid_PMT))
```

Unfortunately, when you run pred3 on the holdout sample it only predicts 46.23% of the invalid payments correctly. Based on this you determine that no model of those you tested establishes a statistical or predictive relationship between whether or not an email that is invalid determines whether a payment will be fraudulent.

Conclusion

You decide it's best to break the news to Mr. Manager gently that his hunch is likely not correct and that a model should not be implemented based on it by offering some additional data analysis and exploration activities that could help in model development given more time:

1. Collect a greater number of payment observations (thousands if not millions of records) that may indicate invalid email as a predictor of fraudulent payments.

2. Perform more extensive cross validation on the KNN (or other classification models) to ensure all predictive power and information gain is determined from the data

3. Identify additional features and attributes (numeric or categorical) which can be used to help predict fraudulent payments. These can include:

    a. Time of payment (i.e. odd hours)

    b. Location of payment relative to individual's home address, state, and or country

    c. If payment had been made to receiver of payment previously