Homework6 - Group 6

Shahzad Anasari, Jordan Daugherty and Karen Ochie

10/18/2021

Learning Objective: Data wrangling, regression modeling and analysis

Problem 1 - Online Retail Sales Prediction

(a) Data Preparation and Modeling

i. Data Understanding

```
ggplot(data=training_data, aes(y=revenue, x=operatingSystem))+
  geom_point(color="blue")+
  xlab("Operating Sysem")+
  ylab("Revenue")+
  labs(title="Figure 1: Revenue vs Operating System")+
  dark_theme_bw()
```

```
## Inverted geom defaults of fill and color/colour.
## To change them back, use invert_geom_defaults().
```

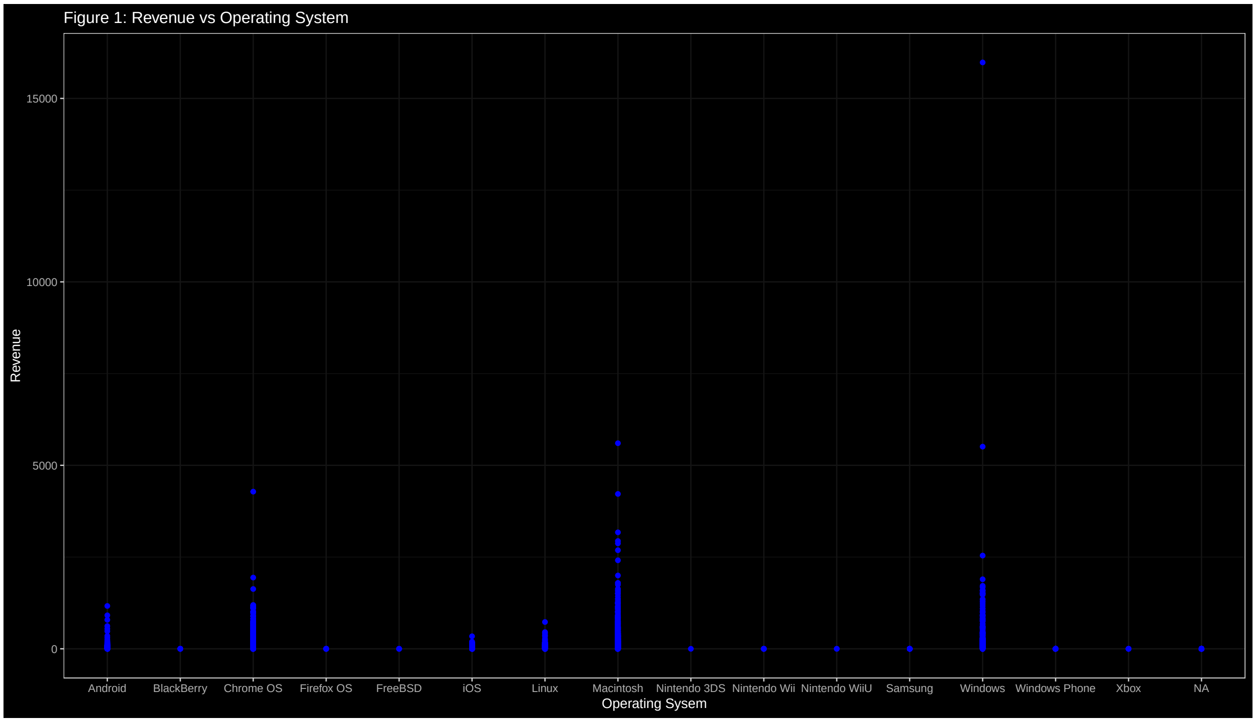


Figure 1 illustrates the relationship between different operating systems used and the amount of revenue gained from each operating system. We can take away some good information from this graph that can help us predict future revenue based on what operating system was used.

```r
ggplot(data=training_data, aes(y=revenue, x=subContinent))+
  geom_point(aes(color=channelGrouping))+
  labs(x="Sub-Continent",y="Customer Revenue",title="Figure 2: Revenue vs. Sub-Continent",
       size=10)+
  scale_colour_discrete("Channel\nGrouping")+
  dark_theme_light()+
  theme(axis.text.x = element_text(angle=45, vjust=1, hjust=1))
```

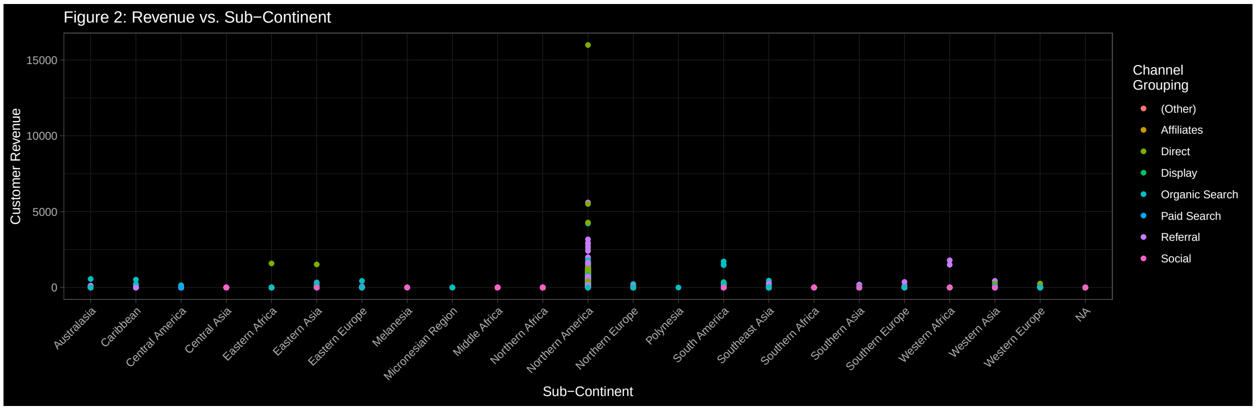


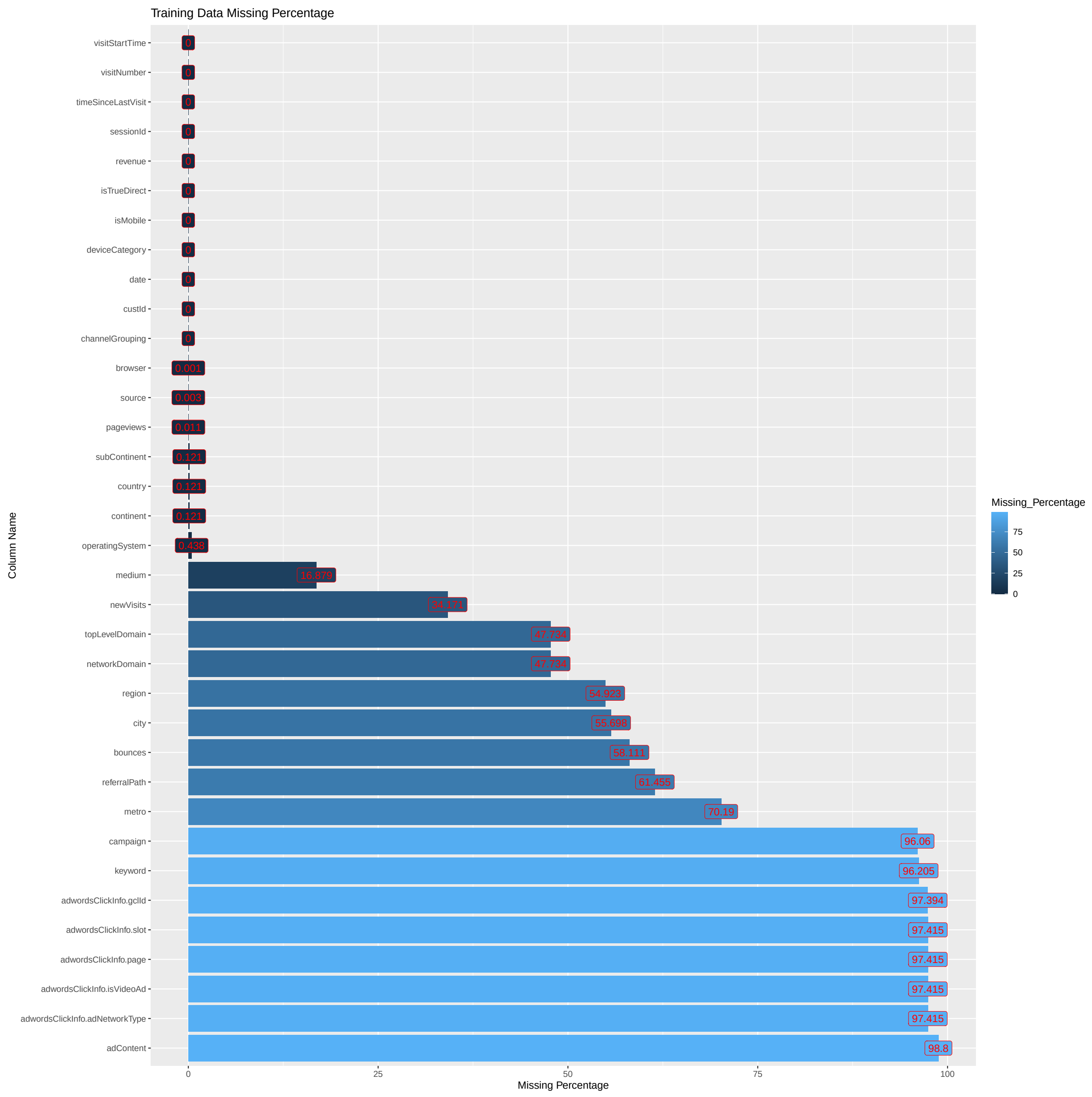
Figure 2 gives us information from a different angle, showing us the amount of revenue gained based on the continent the user is from. Channel grouping was also added to show what category was used by the user.

Overall, we can learn a lot about what different factors have high influences on the amount of revenue gained by a customer. From Figure 1, we can see that a majority of revenue is gained when the operating systems Macintosh, Chrome, Windows and Android are used. Figure 2 illustrates that Northern America is by far the sub-continent with the most revenue gained. The channel grouping also provides us with more incite on which channel groupings are used more frequently for purchases. We can also see that there variables that contain many factor levels that dont typically produce any revenue.

ii. Data Preparation

```r
#Finding percentage of missing data in each variable
percentagesDf = data.frame(colnames(training_data))
for(i in 1:ncol(training_data)) {       # for-loop over columns
  percentagesDf[i,2] = (round((sum(is.na(training_data[,i])))
                              /dim(training_data)[1],5))*100
}
names(percentagesDf)[1] = "Column_Name"
names(percentagesDf)[2] = "Missing_Percentage"
percentagesDf = as.data.frame(percentagesDf)

ggplot(data = percentagesDf, mapping = aes(x = reorder(Column_Name, -Missing_Percentage),
                                           Missing_Percentage)) +
  geom_bar(stat = "identity",aes(fill=Missing_Percentage), position = 'dodge') +
  coord_flip()+
  geom_label(aes(label = Missing_Percentage,fill=Missing_Percentage),color = "Red")+
  xlab("Column Name")+
  ylab("Missing Percentage")+
  ggtitle("Training Data Missing Percentage")
```



```
percentagesDf = data.frame(colnames(testing_data))
for(i in 1:ncol(testing_data)) {        # for-loop over columns
  percentagesDf[i,2] = (round((sum(is.na(testing_data[,i])))
                              /dim(testing_data)[1],5))*100
}
names(percentagesDf)[1] = "Column_Name"
names(percentagesDf)[2] = "Missing_Percentage"
percentagesDf = as.data.frame(percentagesDf)
```

Data Preparation

1. Factor Selection: After visualizing the data set and seeing how much of each factor contains NA values, we began to select which we would keep. As a general rule any factor containing over 20 percent of the missing information we determined to be not worth imputing as any imputation methods we would use may be inaccurate or in some cases, it would be impossible to create an accurate imputation i.e city. There would be no way for us to know which city they originated from nor what metro etc. We also removed information that we determined to be of little value such as bounces, sessionId,visitStartTime, and date. There would be no real meaningful information gained from knowing what time a session started and modeling confirmed that some of these factors were not of high value.

2. Imputations: Once we have decided on which factors are worth keeping we need to then impute the missing values in those factors. Many of the factors were categorical in nature meaning any numerical method used to impute such as mean value imputation would be out of the question. For the categorical variables such as operating system, browser and medium the imputation method used was a probabilistic method. Taking the occurrences of each category within the column you could create a probability distribution. When an NA is encountered randomly generate a value using the probability distributions to select which of the existing categories will be used to impute the NA. For NewVisits and pageViews a more numerical method could be used, since there was no way of knowing if a visit was a new visit we would just assume it wasn't a new visit. For the page views, we would determine the average page views using mean value imputation and rounding down. Within the location data country, continent and subcontinent there were 84 values in which all 3 were unknown, since there was no possible way to know what these three could be a new level was added called unknown and they have imputed it as such.

3. Factor Collapsing Before any modeling is done we need to deal with the fact that some categories have too many levels. subContinent, country, browser, country, and operatingSystem all have many levels and some of these levels may only have a few entries. For the Subcontinent continents and countries, we determined that over 50 percent of entries come from the united states, using factor lump we consolidated the categorical columns into two levels each. The North American Continent, the north subcontinent, and the United States. The levels for browser were reduced to the top 8 levels and 1 other level and the operating system column was lumped into the top 6 levels and one other level.

4. Outlier Resolution: After the initial data analysis, the revenue variable has a very significant outlier. This outlier was a value of 15980.79 and came from one single customer Id. This customer visited the site 6 times, but only made a purchase one time. In order to avoid the prediction model being skewed, the outlier was removed.

5. Data Aggregation: After the initial data preparation, the training and testing data was aggregated to summarize the parameter variables and modify the output variable to the natural log of the total revenue plus one for each customer Id. The summarized parameters include the average page views, the maximum number of visits, the average visit gap time and the mode for the browser, operatingSystem, country and subContinent.

OLS Model

Initially we tried the linear model from the stats package as a first attempt. This did not produce a very good RMSE value. The caret package train function was then tested with a K-fold cross validation re-sampling approach was also implemented. For the K-fold CV a fold value of 20 and a k value of 5 was used. This produced a better RMSE value.

PLS Model

A PLS model was then developed from the caret package using the train function with the pls method. This model also used K-fold cross validation re-sampling approach with the same fold=20 and k=5 from the OLS train model and produced an RMSE that was worse than our OLS model. From there, the pls package was tested to compare similar models from different packages would perform. The oscorespls method was used for this model, with a ncomp of 30 and a cross validation approach. This model produced a similar value to the OLS model.

Elastic Net Model

Here, two different caret package train methods were tested. The first a glmnet method with K-fold cross validation with 5 folds and k=5. This produced an RMSE of 0.937, which wasn't better than our OLS or our PLS models. Next, the enet method was tested on the Elastic Net model and along with a K-fold cross validation with 5 folds and a k-value of 10. This produced a significantly worse RMSE value of 1.02 with a lambda of 0.5 and a fraction of 0.3.

```
################################################################################
#                           ELASTIC NET MODEL                                  #
################################################################################

#TRAINING ELASTIC NET REGRESSION MODEL

control <- trainControl(method = "repeatedcv",
                        number = 5,
                        repeats = 5,
                        search = "random",
                        verboseIter = TRUE)

elastic_model <- train(data=t1,log(totalRevenue + 1) ~ .-custId,
                       method = "glmnet",
                       na.action = na.pass,
                       preProcess = c("center", "scale"),
                       tuneLength = 10,
                       trControl = control)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

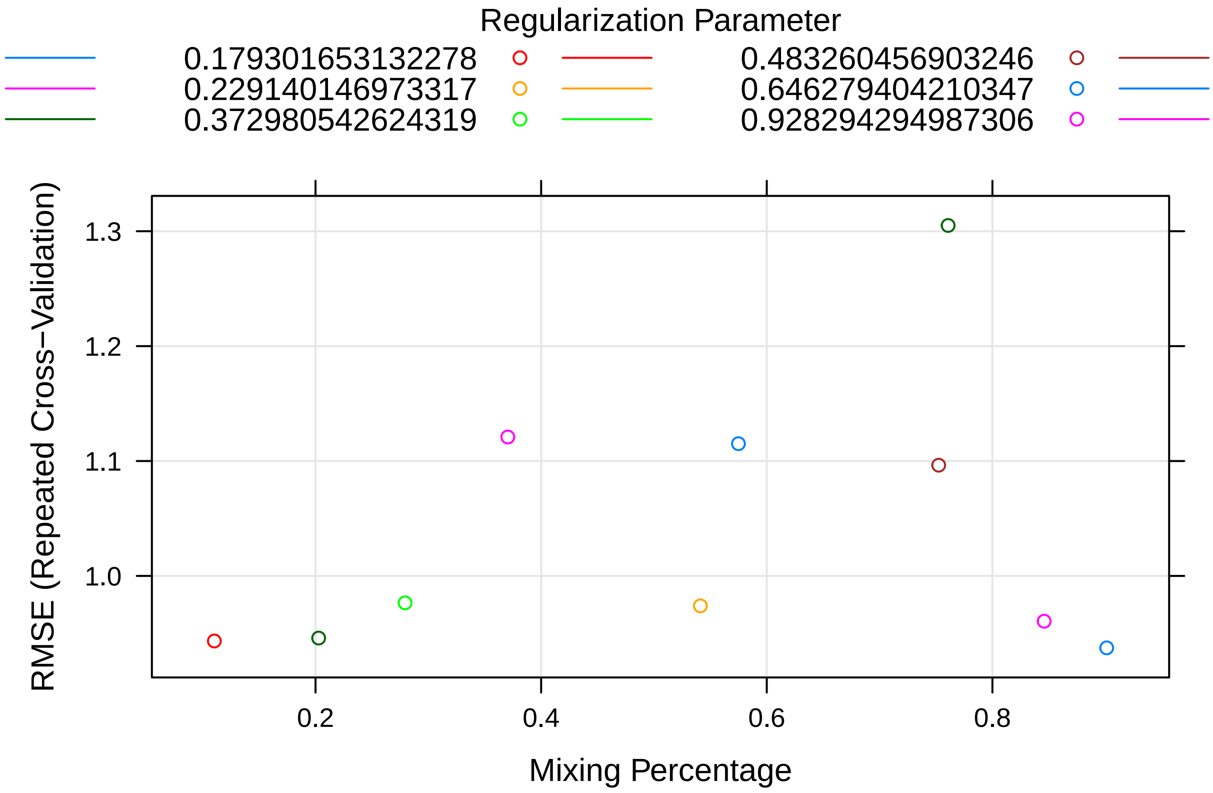
```
grid <- expand.grid(
  lambda   = seq(0.5, 0.7, by=0.1),
  fraction = seq(0, 1, by=0.1)
)

ctrl <- trainControl(
  method     = 'repeatedcv',
  number     = 5,  #folds
  repeats    = 10, #repeats
  classProbs = FALSE
)


enetTune <- train(data=t1, log(totalRevenue + 1) ~ .-custId,
  method = 'enet',
  metric = 'RMSE',
  tuneGrid = grid,
  trControl = ctrl
)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
plot(elastic_model)
```



```
ELASTIC_RMSE<-min(elastic_model$results$RMSE)
ELASTIC_R2 <- elastic_model$results$Rsquared[2]


pred_elastic <- predict(elastic_model,newdata=t2)
```

LASSO Model

Next, a LASSO model was tested with the caret package train function. Here a K-fold cross validation was used with 20 fold and a k-value of 5. A tune grid was also used to test 100 different fraction values. This model produced our an RMSE value better than the 3 previous model types tested.

Ridge Regression

Another model was then tested using the caret package train function with the glment method, a tune grid where multiple lambda values were tested, and K-fold cross validation was used with 5 folds and a k-value of 5. This model produced a RMSE that was better than the elastic net, but not better than any other model tested.

SVM Model

Lastly, an SVM model was tested using the e1071 package and the eps-regression model type. This model produced a significant improvement to the RMSE with an RMSE of 0.74.

Conclusion

Our modeling approach was to test a few variations of each model and from there look for models that produced the best RMSE from our cleaned data. A lot of the models that were tested, were tested with trial and error to see how different hyperparameters would impact the RMSE values. Multiple grid techniques were used to test multiple hyperparameter values to achieve an optimal result. One of the main problems that occurred during out model testing was the issue of run time with the SVM model. SVM produced the best result, but also took the longest time to compute, which made it very time consuming to trial different variations of this model.

```
summary_data
```

```
##        Model Method Package      Hyperparameter                 Value   CV_RMSE
## 1        OLS     lm   stats                  NA                    NA 0.9323020
## 2        PLS   plsr     pls               ncomp                    31 0.9327614
## 3      LASSO  lasso   caret            fraction             0.5405051 0.9324956
## 4 elasticNet glmnet   caret        alpha,lambda 0.9012876,0.004112218 0.9373690
## 5      RIDGE glmnet   caret       alpha, lambda                   0,0 0.9363191
## 6        SVM    svm   e1071 cost, gamma,epsilon         1,0.03125,0.1 0.7468956
##       CV_R2
## 1 0.4961491
## 2 0.4847112
## 3 0.4966205
## 4 0.4841742
## 5 0.4879861
## 6 0.6734063
```