Final Exam INFO 3130

Cody Frisby

12/11/2017

1.

Eric,

First we need business understanding. What does the business do?
We need to look at the data and understand it. What's it look like? Are there any rows in your spreadsheet that are far away from the others?
What data preparation steps are needed? Do we need to recode any of the variables?
We will build a model. What is the best model for the question we want to answer? That will depend on the structure and types of data that we have. We will then need to evaluate the model and test how "good" it is. If it needs some tuning we can go back a step and build it again by tweaking one or two things. When we are happy with the model's predictions/categorizations then we can deploy it into the wild :).

2.

We are going to use an association model to investigate "rules" or associations among the elements. Using $supp = 0.005$ and $conf = 0.25$ we get the following association rules using `R` package `arules`.

lhs	rhs	support	confidence
{Bottle/First,Trek/Book} =>	{Safety Boot}	0.007915567	0.3750000
{Safety Boot,Trek/Book} =>	{Bottle/First}	0.007915567	0.3750000
{Bottle/First} =>	{Safety Boot}	0.042216359	0.3720930
{Safety Boot} =>	{Bottle/First}	0.042216359	0.3555556

We can see, with around 37.5% confidence that if *Bottle/First, Trek/Book* then *Saftey Boot. The* support* for this rules is 0.00792. What we mean by support is that this is the proportion of times we observed this rule in our data set. While what we mean by confidence is based on the model, this is how confident we are to see the left-hand side predict the right-hand side.

3.

A decent model of choice, since we'd like to predict a numerical variable, is a linear regression model. Below is a summary of the model. When we see a large *t value* or a small *p value* then we want these variables in our model as they are "good" predictors of **store revenue**. If we see those with large *p value*s than these are variables we do not need to include in our model. The reason we can get rid of them is they are not important to explain all the variation we see in **store revenue**.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	731725.717787	716996.77884	1.0205425	0.3113136
Efficiency.Score	-31622.080671	7997.04786	-3.9542193	0.0001946
Square.Footage	9.503476	22.92813	0.4144897	0.6799015
Monthly.Returns	5989.519046	614.57991	9.7457124	0.0000000
Monthly.Employee.Turnover	1939.070940	13072.81715	0.1483285	0.8825499

	Estimate	Std. Error	t value	Pr(>|t|)
Number.of.Employees	-25954.143707	10928.63444	-2.3748753	0.0205630

My monthly predicted revenue for

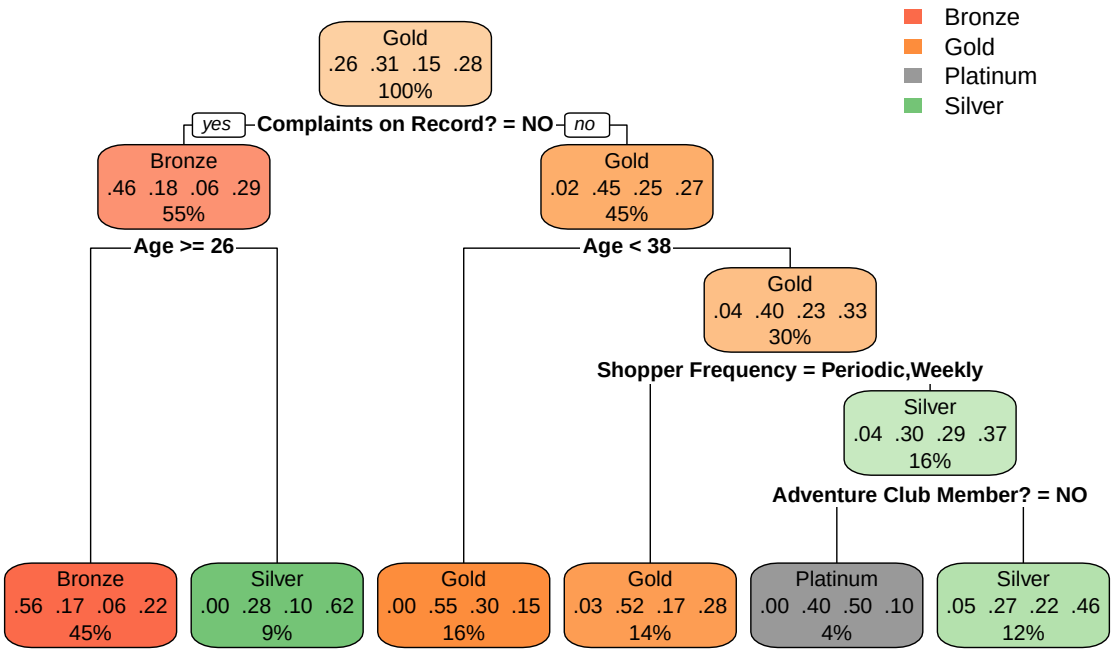
$$Efficiency\ Score = 20,\ Square\ Footage = 5,\ Monthly\ Returns = 550, ...$$

$$Monthly\ Employee\ Turnover = 21, Number\ of\ Employees = 39$$

is **2422076**.

4.

Using the tree below, we can follow the stems and leaves to the prediction. We would arrive at a buyer category of **gold**.



5.

Using the sheet "Employee Performance Data" we can build a logistic regression model since the outcome is binary (employee was either fired or not).

The data does not appear to be very helpful in predicting who to keep and who to let go. None of our variables are strongly predictive of **Retain?**. In the below table we can see that none of the variables are statistically significant predictors.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.0239724	1.3102170	0.7815288	0.4344915
`1st Review Rating`	-0.1223448	0.3121625	-0.3919265	0.6951125
`2nd Review Rating`	-0.1653071	0.3071956	-0.5381167	0.5904965
`3rd Review Rating`	0.1686405	0.3061242	0.5508891	0.5817097

	Estimate	Std. Error	z value	Pr(>|z|)
`Weeks Employed`	-0.0022368	0.0061117	-0.3659921	0.7143710
Salary	-0.0000122	0.0000171	-0.7147531	0.4747616

6.

Using data sheet "Supplier Data" we build an ANN using *Relationship Value* as the dependent variable and the others (excluding *SupplierID*) as the predictor variables.

(**Note**: My computer is erroring out when I try to run the `nnet` function on the data. Not sure if this is due to when I rebuild my operating system and now I'm missing something that I should have. Anyways, I cannot fit the ANN model. I can try to predict the categories using another one, such as Naive Bayes and I can figure out later why it's doing this. #sorry.)

Here are what the first few predictions look like when applying our model to the "Potential New Suppliers" data set.

High	Low	Lowest	Medium
0.0000000	0.0000543	0	0.9999457
0.0000000	0.6550576	0	0.3449424
0.0000000	0.9998179	0	0.0001821
0.0000000	0.9999224	0	0.0000776
0.0583621	0.0000000	0	0.9416379
0.0163636	0.0000000	0	0.9836364

	High	Low	Lowest	Medium
[38,]	0.0000000	0.999992	0	0.0000080
[39,]	0.9992957	0.000000	0	0.0007043
[40,]	0.0128658	0.000000	0	0.9871342
[41,]	0.9999790	0.000000	0	0.0000210
[42,]	0.0000005	0.000000	0	0.9999995
[43,]	0.0000000	0.000000	0	1.0000000

Below is the table with the count of each predicted class using Naive Bayes. (I'd like to see how this compares with ANN :(. What happened to my machine?)

```
## b
##   High    Low Lowest Medium
##      6     13      1     23
```

7.

I compute the correlation matrix (using the default as I've run out of time to examine whether or not we should use the more robust non-parametric methods) amongst all the variables.

There is a strong correlation between monthly returns and monthly revenue (0.9411376). Efficiency score and monthly revenue are strongly negatively correlated (-0.79338). And efficiency score and monthly returns are negatively correlated as well (-0.7601500).

8.

The top words are call, rockwear, and card with 13, 12, and 10 counts respectively.

words	freq
call	13
rockwear	12
card	10
day	9
order	9
credit	9
told	8
payment	7
ship	7
custom	6

R Code

```
## packages used:
library(readxl)

## Read the data from the provided excel file
path <- "~/Documents/school/info3130/data/finalexamdata.xlsx"
sheets <- excel_sheets(path)
df1 <- read_excel(path, sheets[1])
## df1 has an extra row that shouldn't be there
df1 <- df1[-380, ]
df2 <- read_excel(path, sheets[2])
df3 <- read_excel(path, sheets[3])
df4 <- read_excel(path, sheets[4])
df5 <- read_excel(path, sheets[5])
df6 <- read_excel(path, sheets[6])
df7 <- read_excel(path, sheets[7])
##########################################################
# Here's a function I wrote to do some repetitive work.
## Split the data into a test and training set
CrossVal <- function(df, p = 2/3) { # takes an R dataframe
  n <- dim(df)[1] # number of rows in df
  s <- sample(1:n, size = n * p) # simple random sample of size n*p
  train <- df[s, ] # training dataset
  test <- df[-s, ] # cross validation training set.
  return(list(train = train, test = test))
}

##### 2 #####
library(arules)
df_rules <- as.data.frame(df1[, 11:13])
df_rules[df_rules == FALSE] <- NA
rules <- apriori(df_rules, parameter = list(supp=0.005, conf=0.25))
rules <- sort(rules, by = "confidence", decreasing = TRUE)
```

```r
inspect(rules)
###### 3 ########
test <- data.frame(`Efficiency Score` = 20, `Square Footage` = 5,
                   `Monthly Returns` = 550,
                   `Monthly Employee Turnover` = 21, `Number of Employees` = 39)
names(df2) <- c("ID", names(test), "Monthly.Revenue")
fit <- lm(Monthly.Revenue ~ ., data = df2[-1])
temp <- summary(fit)
knitr::kable(temp$coefficients)
p <- predict(fit, test)
###### 4 #######
library(rpart)
library(rpart.plot)
# sheet 6
fittree <- rpart(`Buyer Category` ~ ., data = df6[-1], cp = 0.01)
rpart.plot::rpart.plot(fittree, uniform = TRUE, main = "")
###### 5 #######
fit <- glm(`Retain?` ~ ., data = df3, family = binomial(link = "logit"))
###### 6 #######
## Naive Bayes
nb <- e1071::naiveBayes(`Relationship Value` ~ ., data = df4[-1])
pnb <- predict(nb, df5[-1], type = "raw")
knitr::kable(head(pnb))
knitr::kable(tail(pnb))
classes <- colnames(pnb)
p <- apply(pnb, 1, function(x) which.max(x))
b <- vector()
for(i in 1:length(p)) {
  b[i] <- classes[p[i]]
}
table(b)
library(quanteda)
df <- readLines("~/Documents/school/info3130/data/finalcomplaints.txt")
c1 <- corpus(df)
mod <- dfm(c1, remove = stopwords("english"), remove_punct = TRUE,
           stem = TRUE)
x <- as.data.frame(topfeatures(mod))
names(x) <- "freq"
x$words <- row.names(x)
x <- x[, c("words", "freq")]
knitr::kable(x, row.names = FALSE)
```