MISSISSIPPI STATE UNIVERSITY

Data Analysis One

Emily Hubbard

October 8, 2025

HW 1

SECTION ONE

Executive Summary:

A. Introduction:

Understanding the factors that drive consumer brand choice is essential in predicting market behavior. Questions such as: Do discounts on a particular brand increase the likelihood of purchase? Does brand loyalty outweigh price sensitivity? How does a change in list price influence a customer's final decision? These are the kinds of questions we will find answers to using the methods utilized in this report Analyzing the OJ data within R, we will find the factors that most influence a customer's choice between Minute Maid (MM) and Citrus Hill (CH) orange juice brands. We will do so by fitting the data to both Ridge Regression Models(RRM) and Lasso Regression Models(LRM).

B. Data Collection

- **"Purchase"** A factor with levels CH and MM indicating whether the customer purchased Citrus Hill or Minute Maid Orange Juice.
- **"WeekofPurchase"** Week of Purchase
- **"StoreID"** Store ID
- **"PriceCH"** Price charged for CH

- **"PriceMM"** Price charged for MM
- **"DiscCH"** Discount offered for CH
- **"DiscMM"** Discount offered for MM
- **"SpecialCH"** Indicator of special on CH
- **"SpecialMM"** Indicator of special on MM
- **"LoyalCH"** Customer brand loyalty for CH
- **"SalePriceMM"** Sale price for MM
- **"SalePriceCH"** Sale price for CH
- **"PriceDiff"** Sale price of MM less sale price of CH
- **"Store7"** A factor with levels No and Yes indicating whether the sale is at Store 7.
- **"PctDiscMM"** Percentage discount for MM
- **"PctDiscCH"** Percentage discount for CH
- **"ListPriceDiff"** List price of MM less list price of CH
- **"STORE"** Which of 5 possible stores the sale occured at

C. Summary Information

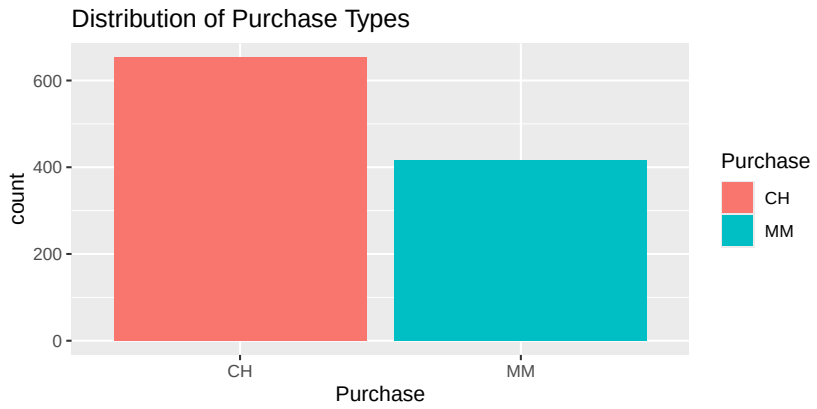

Figure 0.1: Basic Summary of Leuk Data

This figure visualizes the proportion of the response variables that fall in the "CM" category versus the "MM" category.
Correlation Chart for OJ Numerical Data:

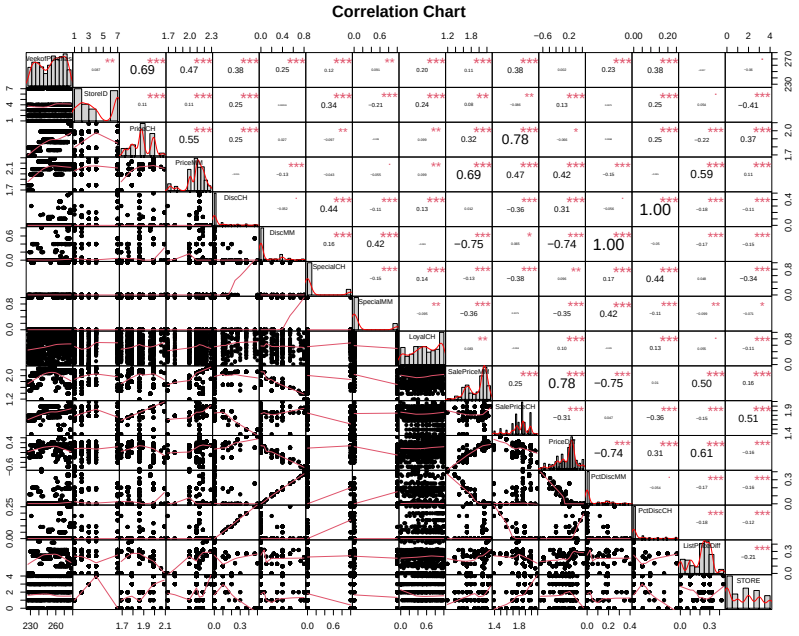

Figure 0.2: Correlation Chart

Observations:

This plot provides a dizzying amount of information. The only valuable piece I want to point out is the myriad of high-correlation calculations. There are 15 that are greater than |0.5| which indicates a very high multicollinearity across the various variables in the data set. Reducing multicollinearity is one of the many reasons to use ridge regression or lasso regression.

D. Report Body

1. First, we will create a training set containing a random sample of 800 observations, and a test set containing the remaining observations. Then we will find the best lambda for a ridge regression model(RRM), fit the model, and interpret its coefficients. After that, we will calculate its Mean Squared Error(MSE) as well as produce a confusion matrix and misclassification rate to analyze the model's reliability and usefulness.

   Before we get started, I want to define the response variable:

$$Y = \begin{cases} 0 & \text{if Citrus Hill is purchased} \\ 1 & \text{if Minute Maid is purchased} \end{cases}$$

Now that we have defined the response variable, we can use cross-validation to find the best lambda for our RRM. The following figure shows the lambda value at various levels of binomial deviance.

Cross-Validation Curve:

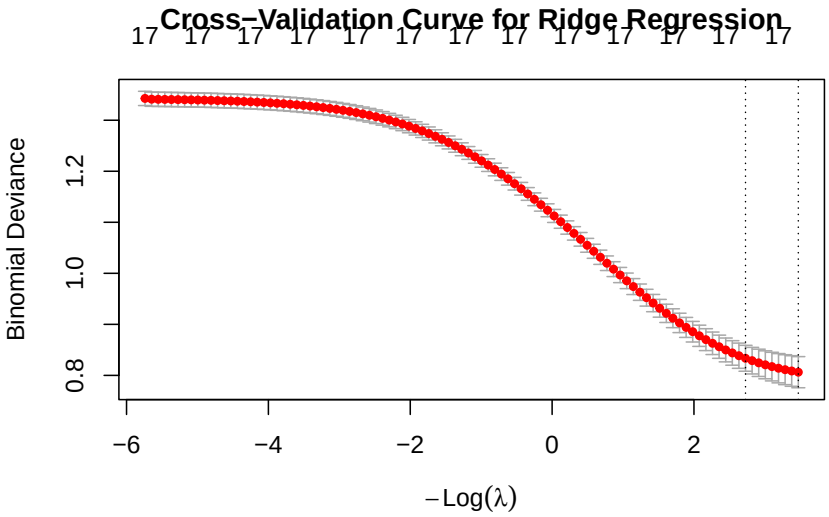

Figure 0.3: R Summary Output

Explanation:
Notice that lowest point is the best lambda value because it allows for the lowest binomial deviation. This is the number that we will use in our RRM.

$$\lambda = 0.031$$

$$-\text{Log}(0.031) = 3.473$$

(The -Log is added for ease of Figure 0.3 interpretation)

After calculating the best λ value, it is now appropriate to fit the RRM to our data. The model produced the following coefficients:

Ridge Regression Coefficients:

```
(Intercept)     2.864068444
WeekofPurchase -0.008487537
StoreID        -0.086819381
PriceCH         0.642651730
PriceMM        -0.217430282
DiscCH         -0.950638639
DiscMM          0.495394873
SpecialCH      -0.015598926
SpecialMM       0.352662996
LoyalCH        -4.362040787
SalePriceMM    -0.418196465
SalePriceCH     0.962467411
PriceDiff      -0.617295907
Store7Yes      -0.320099626
PctDiscMM       0.754160320
PctDiscCH      -1.687326926
ListPriceDiff  -0.902182209
STORE           0.016997052
```

Figure 0.4: Ridge Regression Coefficients

Observations:

Because we are using the ridge regression method instead of lasso, the covariates deemed less influential compared to the others have values that near zero. In a lasso method, these values would simply be taken out. For this model, the corresponding covariate to any coefficient close to zero can be interpreted to be non-influential in predicting the response variable.

This leaves us to analyze the larger numbers more closely to figure out which are the most influential. The following are the top 5 most significant influencers:

$$\begin{aligned}
&\text{LoyalCH} = -4.362\\
&\text{PctDiscCH} = -1.687\\
&\text{DiscCH} = -0.951\\
&\text{ListPriceDiff} = -0.902\\
&\text{SalePriceCH} = +0.963
\end{aligned}$$

By far, the customer brand loyalty for CH is the greatest influence on the response variable. We can interpret this to mean that if a person has a brand loyalty to CH, the log-odds of a Minute Maid(MM) purchase fall 4.362 times per one unit increase in brand loyalty (though I am not sure how this qualitative covariate is measured). This coefficient makes perfect sense because a person loyal to Citrus Hill(CH) will have a lower probability to purchase MM. The same can be said of the next three covariates. The 5th covariate, unlike the rest, makes a positive impact on the probability of a MM purchase. It makes sense that an increase to the price of CH would constitute a larger probability of a MM purchase.

So, now that we have fit our model and understand the influence of individual covariates, it is now time to check the accuracy. Is this a reliable model? Will it be able to accurately predict whether or not MM will be purchased?

To answer these questions, we will do two things:

1. Calculate the MSE.
2. Produce a confusion matrix and subsequent accuracy and misclassification rates.

1. Calculating MSE:

$$\text{MSE} = \frac{(\text{Actual} - \text{Predicted})^2}{\text{number of observations}}$$
$$\text{MSE} = 0.132$$

This value really comes in handy when comparing the model to other methods and their corresponding MSE's.

2. Producing Confusion Matrix and Rates:

Using the remaining observations that were not used to train the model, we will ask the RRM to predict the values of those observations and then compare that with its actual value. This will produce the information we need to build a confusion matrix and calculate relevant rates.

The following is the confusion matrix and rates for our current RRM.

```
> print(conf.mat.ridge)
           y_test
ridge_pred  CH  MM
        CH 155  34
        MM  13  68
> print(accuracy.ridge)
[1] 0.8259259
> print(misclassification.rate.ridge)
[1] 0.1740741
```

Figure 0.5: Confusion Matrix

Explanation: Overall, this model does a fair job at predicting the response variable with an accuracy rate of approximately 82.6 percent. The confusion matrix allows us to see which way the model is incorrectly guessing. For example, of the misclassifications, it is more likely to wrongly predict a purchase of CH when the actual purchase was MM ($\frac{34}{47}$ – the proportion of CH predictions over total misclassifications).

Summary of the RRM:

1. It keeps all 17 covariates.
2. It has an MSE of 0.132.
3. It has a fairly high accuracy rate.
4. It is more likely to wrongly predict a CH purchase.

2. Now, we will repeat this process for a different model – Lasso Regression Model(LRM). (Note: the response variable remains the same.)

   As we did with the RRM, we can use cross-validation to find the best lambda for LRM. The following figure shows the lambda value at various levels of binomial deviance.

   Cross-Validation Curve:

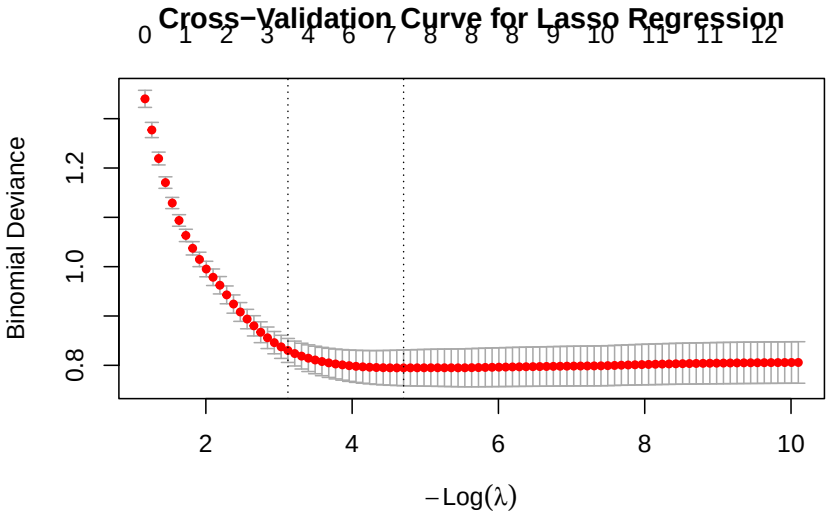

Figure 0.6: R Summary Output

Explanation:
As with the RRM, the lambda with the lowest binomial deviance becomes the best fit lambda for our model. As can be observed, this value falls around 4.7 which is the negative log value of lambda.

$$\lambda = 0.009$$

$$-\text{Log}(0.009) = 4.711$$

(The -Log is added for ease of Figure 0.6 interpretation)

After calculating the best λ value, it is now appropriate to fit the LRM to our data. The model produced the following coefficients:

Lasso Regression Coefficients:

```
                              s0
(Intercept)     2.6879967940
WeekofPurchase -0.0002469293
StoreID        -0.0477303273
PriceCH         .
PriceMM         .
DiscCH         -0.8246557737
DiscMM          .
SpecialCH       .
SpecialMM       0.2165750076
LoyalCH        -5.5724973930
SalePriceMM     .
SalePriceCH     0.2247243008
PriceDiff      -2.1584654249
Store7Yes      -0.5214292113
PctDiscMM       .
PctDiscCH       .
ListPriceDiff   .
STORE           .
```

Figure 0.7: Lasso Regression Coefficients

Observations:

This model trims the covariates down to 7. It chose the most influential covariates, kept them, and got rid of the rest. This reduction in covariates helps immensely when it comes interpretation. It is more simple to draw conclusions from a smaller set of covariates.

This leaves us to analyze the larger numbers more closely to figure out which are the most influential. The following are the top 5 most significant influencers:

$$\text{LoyalCH} = -5.572$$
$$\text{PctDiff} = -2.158$$
$$\text{DiscCH} = -0.825$$
$$\text{Store7Yes} = -0.521$$
$$\text{SalePriceCH} = +0.225$$

My immediate observation is how different these covariates are from the RRM covariates. Loyalty to CH remains the same, but the majority of the covariates have shifted. However, this is normal. Of the 17 covariates to beging with, 4 or 5 are derived from the same information. These lead to having similar influences on the prediction, so these correlations must be dealt with somehow. Both models deal with multicollinearity differently. RRM shrinks coefficients and balances out the duplicate

emphsis. LRM simply chooses one and gets ride of the rest. With that being said, it makes sense that the covariates would be a little different.

As we did for the RRM, we will now check the accuracy of the model.

We will use the same methods to analyze reliability:

1. Calculate the MSE.
2. Produce a confusion matrix and subsequent accuracy and misclassification rates.

1. Calculating MSE:

$$\text{MSE} = \frac{(\text{Actual} - \text{Predicted})^2}{\text{number of observations}}$$
$$\text{MSE} = 0.126$$

When compared to the RRM MSE, it is smaller. The smaller the MSE, the better.

2. Producing Confusion Matrix and Rates:

Using the same process as before, we will use the leftover observations to test the model and produce the following information:

The following is the confusion matrix and rates for our current LRM.

```
> print(conf.mat.lasso)
          y_test
lasso_pred  CH  MM
        CH 156  33
        MM  12  69
> print(accuracy.lasso)
[1] 0.8333333
> print(misclassification.rate.lasso)
[1] 0.1666667
```

Figure 0.8: Confusion Matrix

Explanation:

Overall, this model does a slightly better job at predicting the response variable with an accuracy rate of approximately 83.3 percent, 0.7 percent more than the RRM. Similar to the RRM, the LRM is more likely to wrongly predict a purchase of CH when the actual purchase was MM ($\frac{33}{45}$ – the proportion of CH predictions over total misclassifications).

Summary of the LRM:

1. It keeps 7 covariates.

2. It has an MSE of 0.126.
3. It has a fairly high accuracy rate.
4. It is more likely to wrongly predict a CH purchase.

3. Finally, we will compare our two models and decide which one is superior.

Conclusion:

The lasso model makes correct predictions 0.7 percent more times than the ridge model. Is this a huge difference? No, but in a much larger population, this percentage could make a significant impact. The lasso also has a slightly lower MSE (smaller by 0.006). This also indicates a superior model. Therefore, the lasso is the superior test according to the accuracy rate and MSE calculation.

Overall, which of these models is best?

The lasso regression model is the superior choice. It achieves greater accuracy with a smaller MSE while also keeping fewer covariates. This indicates a simpler model maintaining high accuracy with greater ease of interpretability.

SECTION TWO

Executive Summary:

A. Introduction:

In this section of the report, we will be transforming a data set into its Principle Components and analyzing the output. These principle components are made up of all of the variables in varying proportions in an attempt to capture as much of the total variance in the data as possible using fewer dimensions.

B. Data

```
> str(USArrests)
'data.frame':   50 obs. of  4 variables:
 $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
 $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
 $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
 $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
```

Figure 0.9: Variable Type of USArrests Data

- **"Murder"** numeric Murder arrests (per 100,000)
- **"Assault"** numeric Assault arrests (per 100,000)
- **"UrbanPop"** numeric Percent urban population
- **"Rape"** numeric Rape arrests (per 100,000)

C. Summary Information

PCA is particularly useful for the USArrests dataset because the four variables—Murder, Assault, UrbanPop, and Rape —are likely correlated and may contain overlapping information. By applying PCA, we can reduce these correlated variables into a smaller number of uncorrelated components that capture most of the total variance.

In doing so, we shift our perspective from isolated variables to composite measures that reflect the most important sources of variation in the dataset. While this shift reduces the ability to interpret each variable individually, it allows for a more holistic view of the underlying structure.

D. Report Body

a) We will perform PCA on the USArrests data set, which is part of the base ISLR package.

Output and Summary of PCA:

```
> pca.fit
Standard deviations (1, .., p=4):
[1] 1.5748783 0.9948694 0.5971291 0.4164494

Rotation (n x k) = (4 x 4):
               PC1        PC2        PC3         PC4
Murder   -0.5358995 -0.4181809  0.3412327  0.64922780
Assault  -0.5831836 -0.1879856  0.2681484 -0.74340748
UrbanPop -0.2781909  0.8728062  0.3780158  0.13387773
Rape     -0.5434321  0.1673186 -0.8177779  0.08902432
> summary(pca.fit)
Importance of components:
                          PC1    PC2     PC3     PC4
Standard deviation     1.5749 0.9949 0.59713 0.41645
Proportion of Variance 0.6201 0.2474 0.08914 0.04336
Cumulative Proportion  0.6201 0.8675 0.95664 1.00000
```

Figure 0.10: Results and Summary of PCA

Interpretation of the Summary:

This output gives us a wealth of information. First, take a look at the cumulative proportion on the last line. These values tell us that nearly 87 percent of the variance is taken into account with the first two principle components alone. For this reason, we will be using PC_1 and PC_2 to make our interpretations and conclusions.

Now that we have decided that, we can take a closer look at PC_1 and PC_2. Below, are the weighted sums of PC_1 and PC_2. This indicates the proportion of information that is provided by the corresponding covariate when calculating the principle component, also known as "loadings".

Principle Components Weighted Sums:

$$\text{PC}_1 = -0.536(\text{Murder}) - 0.583(\text{Assault}) - 0.278(\text{UrbanPop}) - 0.543(\text{Rape})$$
$$\text{PC}_2 = -0.418(\text{Murder}) - 0.188(\text{Assault}) + 0.872(\text{UrbanPop}) + 0.167(\text{Rape})$$

Why do these weighted sums matter? What can they tell us about crime rates in the 50 US states?

They tell us a lot actually. PC_1 has coefficients that have similar weights. This means that this component takes those 3 variables into pretty equal account. It has a smaller urban pop coefficient which simply means it doesn't focus on the effect of urbanization as much as, say, another principle component like PC_2. With equal weightings on the crimes and a smaller emphasis on urban population, this PC would be a good indicator for overall crime rates.

On the other hand, PC_2 puts a large emphasis on urban population making it a good indicator for the effect of urbanization on the crime rate. This principle component gives us a more detailed look at which crimes are more common in states with low or high urbanization.

Findings like these help us make connections between covariates that we likely never would have seen with a simpler model.

Visualization of PCA:

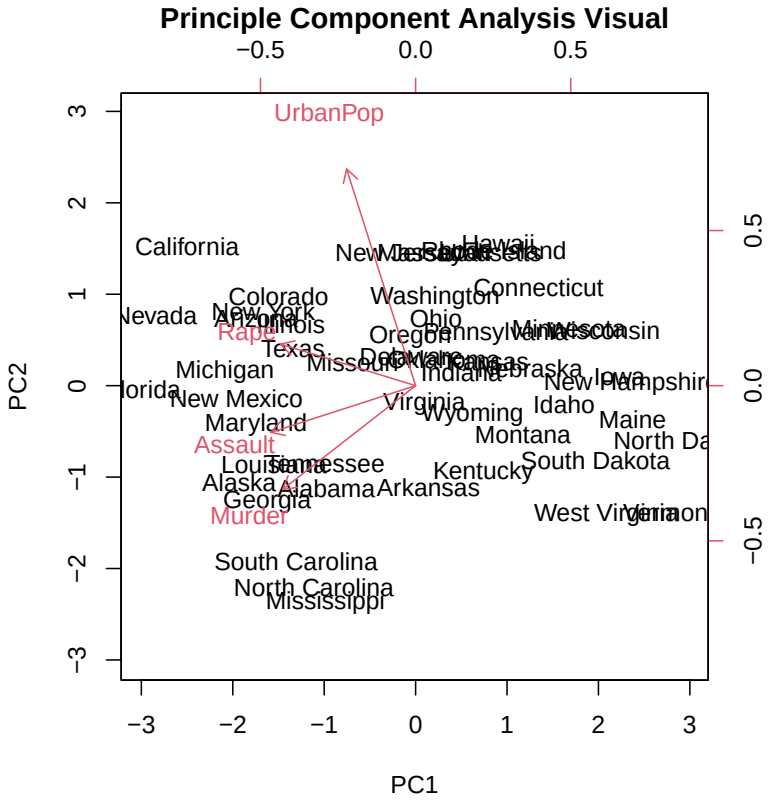

Figure 0.11: Principle Component Analysis Visual

Interpretation of the Visual:

Let's start by looking at the arrows. The direction of the arrows tells us whether that variable has a positive or negative load on the component. For example, all 4 variables point left reletive to the x-axis which corresponds to the fact that all of those loads are negative for PC_1. Similarly, if the plot were viewed 90° clockwise, the relative directions of the arrows would align with the loadings on PC_2. The

length of the arrows explains the magnitude of the effect on the component. For ease of explanation, we will primarily focus on PC_1. All three crimes have similar lengths which supports the similar coefficients found in the weighted sum of PC_1. States that fall in the direction of the arrows indicate a higher number for those variables. This means that states farther left have higher crime rates because that is the direction that murder, rape, and assault point. Similarly, you could say that states on the right have lower crime rates.

Overall Conclusions:

This PCA transformation works really well for this data set. It gives us unique information that would not have been captured in other methods or models. Do we lose interpretibiliy of individual variables? Yes. However, in a sense, we are gaining a new set of interpretations that have their own inherent value.

APPENDIX

R code:

```
#### HW 4 ####
#### 1 ####
library(ISLR)
data(OJ)
View(OJ)
?OJ
summary(OJ)
str(OJ)

### Visuals of Data
library(ggplot2)

ggplot(OJ, aes(x = Purchase, fill = Purchase)) +
  geom_bar() +
  labs(title = "Distribution of Purchase Types")

### Numerical Data
num_vars <- OJ[, sapply(OJ, is.numeric)]

library(PerformanceAnalytics)
chart.Correlation(num_vars,method="spearman",histogram=TRUE,pch=16, main = "Correla

#### a ####
set.seed (1)
sample.index = sample (1:nrow(OJ), 800)
train <- OJ[sample.index,]
test <- OJ[-sample.index, ]

#### b ####
### Ridge Regression
library (glmnet )

#Split Data intor Train and Test Data
x_train <- model.matrix(Purchase ~ ., data = train)[, -1]
y_train <- train$Purchase

x_test  <- model.matrix(Purchase ~ ., data = test)[, -1]
y_test  <- test$Purchase

#Determine Best Lambda Using Cross-Validation
```

```
cv_ridge <- cv.glmnet(x_train, y_train, alpha = 0, family = "binomial")
plot(cv_ridge, main = "Cross-Validation Curve for Ridge Regression")

#Best Lambda
cv_ridge$lambda.min
# =0.031

#Ridge Regression Given Best Lambda
ridge_mod <- glmnet(x_train, y_train, alpha = 0, family = "binomial", lambda = cv_r
ridge_mod

ridge_probs <- predict(ridge_mod, s = cv_ridge$lambda.min, newx = x_test, type = "r
ridge_probs
ridge_pred <- ifelse(ridge_probs > 0.5, "MM", "CH")
ridge_pred <- factor(ridge_pred, levels = levels(y_test))

conf.mat.ridge <- table(ridge_pred, y_test)
accuracy.ridge <- mean(ridge_pred == y_test)
misclassification.rate.ridge <- 1 - accuracy.ridge

print(conf.mat.ridge)
print(accuracy.ridge)
print(misclassification.rate.ridge)

#### c ####
coef(ridge_mod)

#### d ####
y_test_num <- ifelse(y_test == "MM", 1, 0)
#turning the factors into binary outcomes so that it can
#the predicted probabilities can be subtracted and a mean residual error can be fou
ridge_mse <- mean((y_test_num - ridge_probs)^2)
ridge_mse
#=0.132

#### e ####
### Lasso Regression
#Determine Best Lambda Using Cross-Validation
cv_lasso <- cv.glmnet(x_train, y_train, alpha = 1, family = "binomial")
plot(cv_lasso, main = "Cross-Validation Curve for Lasso Regression")

#Best Lambda
cv_lasso$lambda.min
#=0.009
```

```
#Ridge Regression Given Best Lambda
lasso_mod <- glmnet(x_train, y_train, alpha = 1, family = "binomial", lambda = cv_la
lasso_mod

coef(lasso_mod)

lasso_probs <- predict(lasso_mod, s = cv_lasso$lambda.min, newx = x_test, type = "re
lasso_probs
lasso_pred <- ifelse(lasso_probs > 0.5, "MM", "CH")
lasso_pred <- factor(lasso_pred, levels = levels(y_test))

conf.mat.lasso <- table(lasso_pred, y_test)
accuracy.lasso <- mean(lasso_pred == y_test)
misclassification.rate.lasso <- 1 - accuracy.lasso

print(conf.mat.lasso)
print(accuracy.lasso)
print(misclassification.rate.lasso)

lasso_mse <- mean((y_test_num - lasso_probs)^2)
lasso_mse
#=0.126

#### f ####
cv_lasso$lambda.min
#=0.009

#### g ####
accuracy.diff.pct <- (accuracy.lasso - accuracy.ridge)*100
accuracy.diff.pct
#0.74% This means the lasso model makes correct predictions 0.74% more times than t
#Is it a huge difference? No, but in a much larger population, this percentage coul
#Therefore, the lasso is the superior test according to the accuracy estimate alone

#The lasso also has a slightly lower MSE (smaller by 0.006). This also indicates a

#### h ####
#Overall, which of these models is best? The lasso regression model achieves greate
#keeping a smaller number of covariates. This indicates a simpler model with greate
```

```
#### 2 ####
data(USArrests)
View(USArrests)
?USArrests
summary(USArrests)
str(USArrests)

library (pls)
?pcr
?prcomp

#Splitting Data into Principle Components
pca.fit <- prcomp(USArrests, scale. = TRUE)
pca.fit
summary(pca.fit)

#Visual for PCA
biplot(pca.fit, scale=0, main = "Principle Component Analysis Visual")
```