Project 2: Supervised and Unsupervised Learning

SDS322E - By Sam Queralt, Avi Thakor, Karim Aliyev, Reuben Rapose

Introduction

In this project, we have decided to collect information on Spotify and Apple Music subscription costs in different countries. In particular, we chose to look at their premium individual plans, which is the most basic paid plan they offer. Our primary goal was to confirm our belief that both services charge approximately the same price in each country. For further analysis, we have also collected information on the country's mean income, size, population, and their continent. In the end, we have found 68 countries which offered both Apple Music and Spotify services as well as had other necessary data readily available on the web. Part two of our project aims to identify the correlation between the difference in price of the two services and the variables in our data set.

```
# Reading Spotify data from the website and converting it
# to an R dataframe
raw_spotify = read_html("https://howtospotify.com/spotify-premium-price/")
spotify_r = raw_spotify %>%
    html_nodes("table") %>%
    html_table()
spotify_r = spotify_r[2:6] %>%
    Reduce(rbind, .)

# renaming variable to our standardized format
spotify = spotify_r %>%
    dplyr::select(X2, X3) %>%
    rename(c(country = X2, spotify_price = X3)) %>%
    filter(country != "country") %>%
    mutate(spotify_price = as.numeric(str_remove_all(spotify_price,
        "[^0-9.]")))

# renaming countries to match other datasets during the
# merge
spotify$country[which(spotify$country == "USA")] = "United States"
spotify$country[which(spotify$country == "Türkiye")] = "Turkey"
spotify$country[which(spotify$country == "España")] <- "Spain"

# Reading Apple data from a csv
apple_raw = read_csv("apple.csv")

apple = apple_raw %>%
    dplyr::select(Country, "USD Price") %>%
    rename(apple_music_price = "USD Price", country = Country) %>%
    mutate(apple_music_price = as.numeric(str_remove_all(apple_music_price,
        "[^0-9.]")))

# reading Income data from a csv
income_raw = read_csv("medincome.csv")

# renaming variables to our format
income = income_raw %>%
    dplyr::select(country, meanIncome) %>%
    rename(mean_income = meanIncome)

# reading the population data from the web
pop_raw = read_html("https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population
_density")

pop_r = pop_raw %>%
    html_nodes("table") %>%
    html_table()
pop_r = pop_r[[1]]

# using regex to remove extra characters and renaming
# variables to our format
pop = pop_r %>%
    dplyr::select("Country, territory or dependency", "Population",
        "Area") %>%
    rename(country = "Country, territory or dependency") %>%
    filter(country != "Country, territory or dependency") %>%
```

```r
    filter(str_detect(country, "\\)") == FALSE, str_detect(country,
        "\\[") == FALSE) %>%
    mutate(Population = str_remove_all(Population, ","), Area = str_remove_all(Area,
        ",")) %>%
    rename(c(population = Population)) %>%
    mutate(population = as.numeric(population), Area = as.numeric(Area)) %>%
    rename(`Area km^2` = Area)

# reading the data from a csv
country_cont_raw <- read_csv("countryContinent.csv")

# renaming variables to our format
country_cont <- country_cont_raw %>%
    dplyr::select(country, continent)

# renaming USA to match other datasets
country_cont$country[which(country_cont$country == "United States of America")] = "United States"

# join by country
final <- spotify %>%
    inner_join(apple) %>%
    inner_join(income) %>%
    inner_join(pop) %>%
    inner_join(country_cont) %>%
    arrange(country) %>%
    rename(area = `Area km^2`)

# Calculating Population Density
final = final %>%
    mutate(density = population/area) %>%
    mutate(price_diff = spotify_price - apple_music_price, price_diff_bin = ifelse(spotify_price -
        apple_music_price > 0, 1, 0))

# Renaming Oceania to Australia
final$continent[which(final$continent == "Oceania")] = "Australia"

final %>%
    rename(Country = country, `Spotify Price` = spotify_price,
        `Apple Music Price` = apple_music_price, `Mean Income` = mean_income,
        Population = population, `Area (km^2)` = area, Continent = continent,
        `Population Density` = density, `Price Difference` = price_diff,
        `Price Difference > 0?` = price_diff_bin) %>%
    head() %>%
    kable(digits = 2) %>%
    kable_styling(full_width = TRUE) %>%
    column_spec(1, bold = TRUE, border_right = TRUE, color = "white",
        background = "steelblue")
```

Country	Spotify Price	Apple Music Price	Mean Income	Population	Area (km^2)	Continent	Population Density	Price Difference	Price Difference > 0?
Argentina	2.01	6.49	6935	45276780	2780400	Americas	16.28	-4.48	0
Armenia	4.99	6.49	2810	2790974	29743	Asia	93.84	-1.50	0

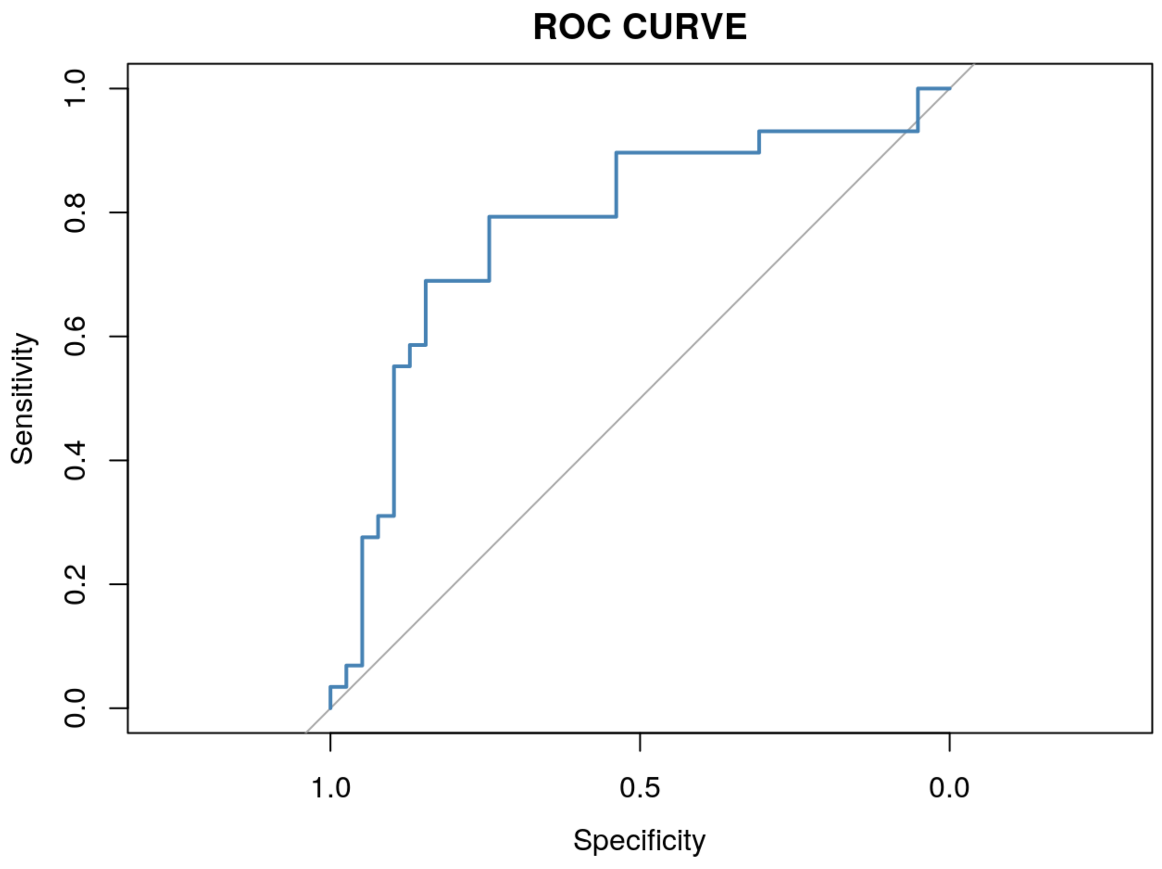
Country	Spotify Price	Apple Music Price	Mean Income	Population	Area (km^2)	Continent	Population Density	Price Difference	Price Difference > 0?
Australia	8.81	8.31	21329	25921089	7692024	Australia	3.37	0.50	1
Azerbaijan	4.99	5.49	3851	10312992	86600	Asia	119.09	-0.50	0
Belarus	4.99	5.99	7359	9578167	207600	Europe	46.14	-1.00	0
Belize	5.99	6.49	3246	400031	22966	Americas	17.42	-0.50	0

Logistic Regression

```
reg_glm = glm(price_diff_bin ~ mean_income + density + population,
    data = final, family = binomial)

pred = predict(reg_glm, final, type = "response")

true_prop = mean(final$price_diff_bin)

pred_bin = ifelse(pred > true_prop, 1, 0)

accuracy = mean(pred_bin == final$price_diff_bin) * 100

con_matrix = confusionMatrix(data = factor(pred_bin, levels = c(1,
    0)), reference = factor(final$price_diff_bin, levels = c(1,
    0)))
con_matrix = con_matrix$table

sensitivity = sensitivity(con_matrix)
specificity = specificity(con_matrix)
```

We have decided to predict whether the price difference is larger than 0 using mean income, population, and population density. We have found our prediction model to have:
- Accuracy of 77.94%
- Sensitivity of 0.6897
- Specificity of 0.8462

With the accuracy of over 77.94%, we feel that the regression model is doing fairly well in terms of predicting the outcome.



```
##
## Call:
## roc.default(response = final$price_diff_bin, predictor = pred,     plot = TRUE, main = "ROC CURV
E", col = "steelblue")
##
## Data: pred in 39 controls (final$price_diff_bin 0) < 29 cases (final$price_diff_bin 1).
## Area under the curve: 0.7772
```

Using our 80-20 split for training purposes, we get the following statistics in 5 trials:

```r
acc_v = c()
sense_v = c()
spec_v = c()
auc_v = c()

for (i in seq(1:5)) {

    set.seed(i)
    train_idx = createDataPartition(final$price_diff_bin, p = 0.8)[[1]]
    price_train = final[train_idx, ]
    price_test = final[-train_idx, ]

    reg_glm = glm(price_diff_bin ~ mean_income + density, data = price_train,
        family = binomial)

    pred = predict(reg_glm, price_test, type = "response")

    pred_bin = ifelse(pred > true_prop, 1, 0)

    acc = mean(pred_bin == price_test$price_diff_bin) * 100

    acc_v = c(acc_v, acc)

    con_matrix = confusionMatrix(data = factor(pred_bin, levels = c(1,
        0)), reference = factor(price_test$price_diff_bin, levels = c(1,
        0)))
    con_matrix = con_matrix$table

    sense = sensitivity(con_matrix)
    spec = specificity(con_matrix)

    sense_v = c(sense_v, sense)
    spec_v = c(spec_v, spec)

    auc = auc(price_test$price_diff_bin, pred)
    auc_v = c(auc_v, auc)

}
```

Test 1
Accuracy: 53.8462
Sensitivity: 0.6667
Specificity: 0.5
AUC: 0.7

Test 2
Accuracy: 92.3077
Sensitivity: 0.75
Specificity: 1
AUC: 1

Test 3
Accuracy: 69.2308
Sensitivity: 0.3333
Specificity: 0.8
AUC: 0.5667

Test 4
Accuracy: 92.3077
Sensitivity: 0.75
Specificity: 1
AUC: 1

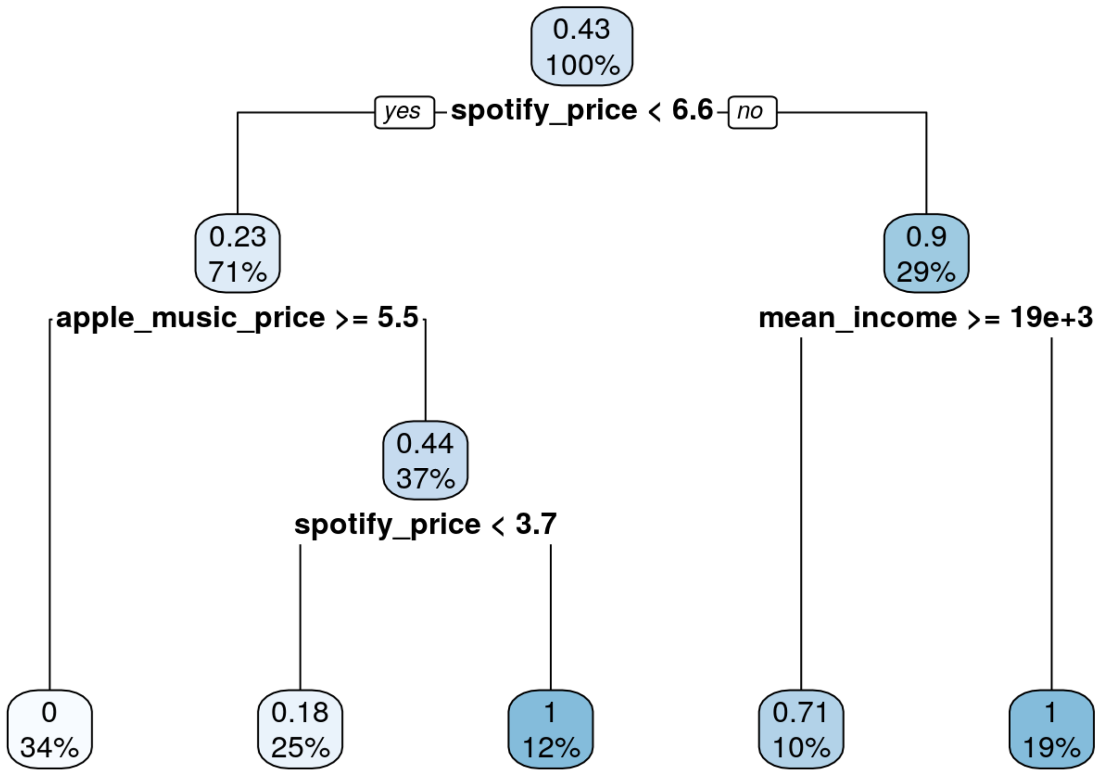
Test 5
Accuracy: 69.2308
Sensitivity: 0.5
Specificity: 0.7778
AUC: 0.8056

As the above train/test split testing shows, our data is inconsistent. Since the full data set is less than 70 entries, the 80% training set may result in drastic changes in the testing phase. Overall, the model seems to be doing an average job at predicting the testing outcomes. Since the data is binary, a fully random model would be expected to predict the correct result about 50% of the time. All but one of the tests resulted in an accuracy significantly above 50%, so we can safely assume that our model is better than a random selection. Additionally, the model's accuracy does not change significantly when using train/test splits when compared to our original glm fit. This means that our model is not overfitting.

Tree-based Classifiers

Firstly, we use the random forest fit in order to predict whether price difference is bigger than 0.

```
tree_price = final %>%
    dplyr::select(-country, -continent, -price_diff, -area)

fit = rpart(price_diff_bin ~ ., data = tree_price)
rpart.plot(fit)
```



```
forest_fit = randomForest(price_diff_bin ~ ., data = tree_price)

forest_pred = predict(forest_fit, data = tree_price)

forest_pred_bin = ifelse(forest_pred > true_prop, 1, 0)

con_matrix = confusionMatrix(data = factor(forest_pred_bin, levels = c(1,
    0)), reference = factor(tree_price$price_diff_bin, levels = c(1,
    0)))
con_matrix = con_matrix$table

acc = mean(forest_pred_bin == tree_price$price_diff_bin) * 100
sense = sensitivity(con_matrix)
spec = specificity(con_matrix)
auc = auc(tree_price$price_diff_bin, forest_pred)
```

Key Statistics for Random Forest Prediction:
Accuracy: 79.4118
Sensitivity: 0.7931
Specificity: 0.7949
AUC: 0.9001

```
acc_v = c()
sense_v = c()
spec_v = c()
auc_v = c()

for (i in seq(1:5)) {

    set.seed(i)
    train_idx = createDataPartition(tree_price$price_diff_bin,
        p = 0.8)[[1]]
    price_train = tree_price[train_idx, ]
    price_test = tree_price[-train_idx, ]

    fit = rpart(price_diff_bin ~ ., data = price_train)

    forest_pred = predict(fit, newdata = price_test)
    forest_pred_bin = ifelse(forest_pred > true_prop, 1, 0)

    accuracy = mean(forest_pred_bin == price_test$price_diff_bin) *
        100
    acc_v = c(acc_v, accuracy)

    con_matrix = confusionMatrix(data = factor(forest_pred_bin,
        levels = c(1, 0)), reference = factor(price_test$price_diff_bin,
        levels = c(1, 0)))
    con_matrix = con_matrix$table

    sense = sensitivity(con_matrix)
    sense_v = c(sense_v, sense)

    spec = specificity(con_matrix)
    spec_v = c(spec_v, spec)

    if (!(is.na(sense) | is.na(spec))) {
        auc = auc(price_test$price_diff_bin, forest_pred)
        auc_v = c(auc_v, auc)
    }

}

acc = mean(acc_v, na.rm = TRUE)
sesne = mean(sense_v, na.rm = TRUE)
spec = mean(spec_v, na.rm = TRUE)
auc = mean(auc_v, na.rm = TRUE)
```

Secondly, we split the data into 80% training and 20% testing sets and compute average statistics over 5 simulations of this process.
Average Statistics for 5 Train/Test Simulations:
Accuracy: 66.1538
Sensitivity: 0.75
Specificity: 0.62
AUC: 0.8194

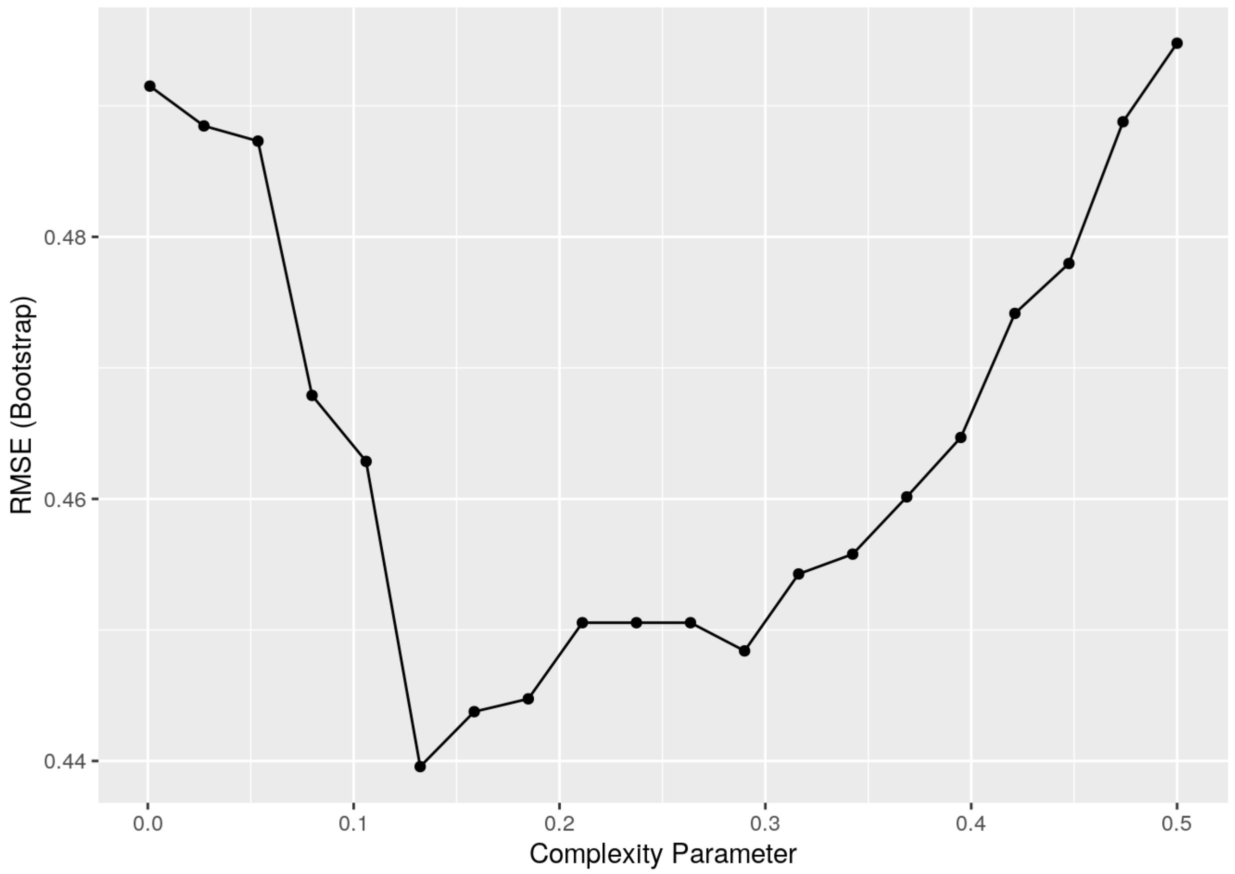
With an average accuracy of 66 percent, our model does a poor job of modeling the data. Since the sensitivity is higher than the specificity, the model does a better job of predicting true positives than true negatives. Since the area under the ROC curve is low, our model is not very useful.

This model does not overfit the data because the average accuracy is so low that it barely fits the data at all. If our model overfit, this accuracy would be much higher until we changed the data to a train/test split. Our tree based model performed more poorly than our logistic model on average. I believe that the tree is more intuitive than other models, but, in this case, is less effective.

Finding the appropriate complexity parameter for the tree.

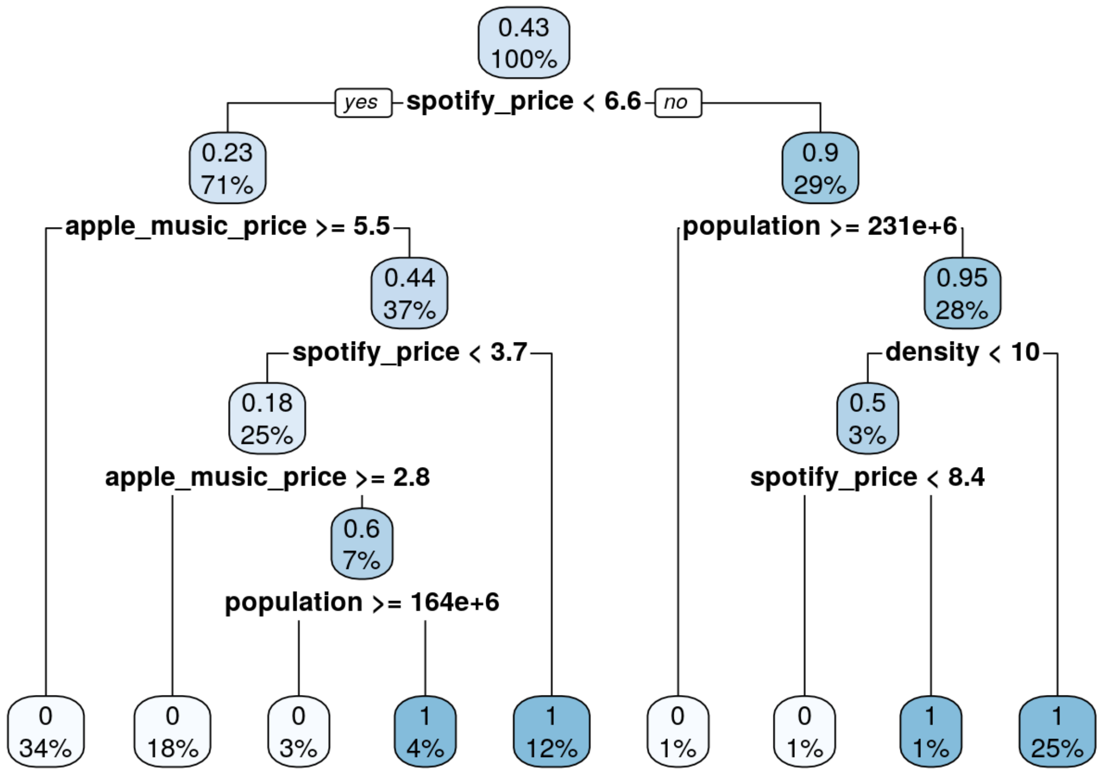
Testing a set of possible complexity parameters, we have found that the appropriate one for our model is around 0.1323.

```
set.seed(80085)
possible_cps <- data.frame(cp = seq(from = 0.001, to = 0.5, length = 20))
control <- rpart.control(minsplit = 2)

train_control <- rpart(price_diff_bin ~ ., data = tree_price,
    cp = possible_cps, control = control)
tuned_rpart <- train(price_diff_bin ~ ., data = tree_price, method = "rpart",
    tuneGrid = possible_cps, control = control)

ggplot(tuned_rpart)
```



Using our found cp, we get the following tree:

```
rpart.plot(train_control)
```

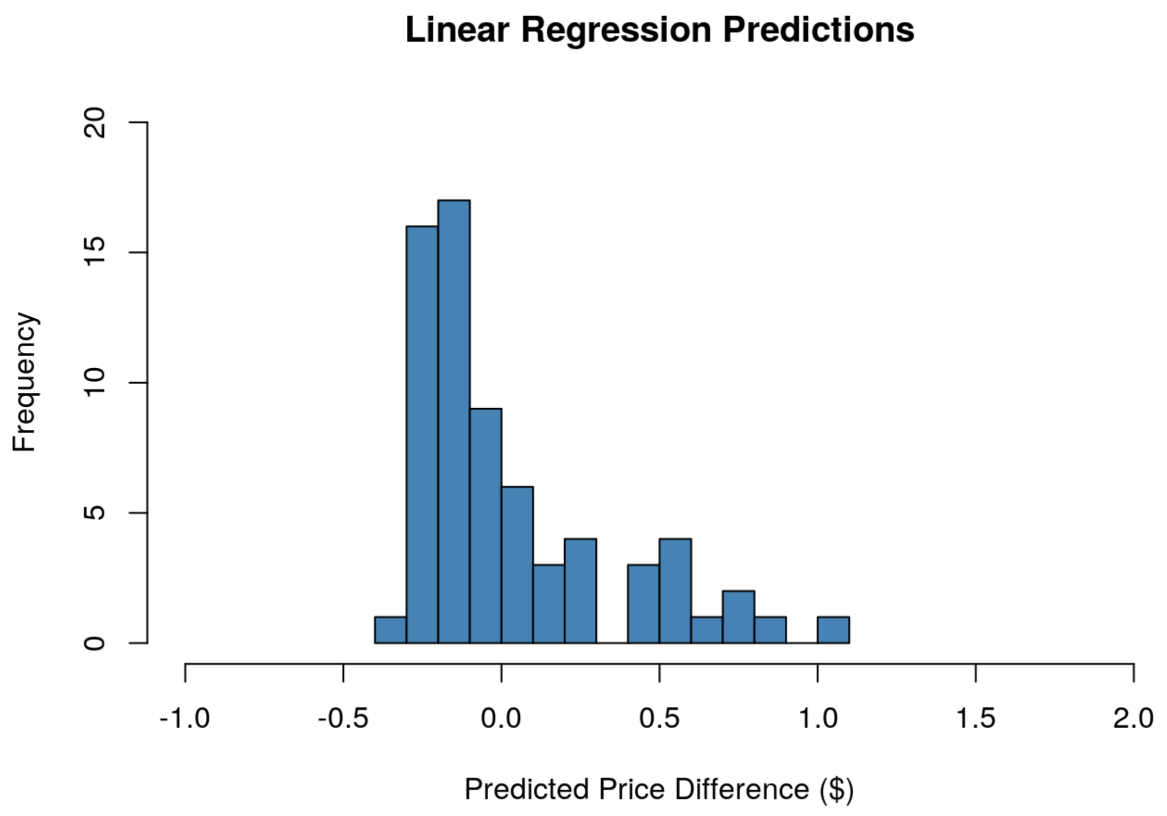


As shown above, the tree shows that `spotify_price` is the most impactful parameter in our data to predict price difference between the two services. When looking at a Spotify price greater than 6.6, we see that the area is a major influencer. In fact, the larger areas account for larger price difference. This is likely to be due to the fact that in larger areas, there is more variability of income, resulting in a greater discrepency between services. When Spotify price is less than 6.6, the influencing factors become slightly more predictable. Indeed, the difference is then based on the individual prices of the services themselves, alternating between the two services. This could be due to there being just enough competition in lower income societies. When we look further, population takes a role in deciding price difference between services, as indicated by the percentages and quantities of 0s on the left, with the larger percentages of 1s on the right. In other words, with greater population and greater Spotify price, the tree indicates that we might expect larger price differences.

Regression/Numeric Prediction

We have decided to predict price difference using mean income and population density.

```
price_diff_lm = lm(data = final, price_diff ~ mean_income + density)
pred_price_diff = predict(price_diff_lm)
lm_rmse = RMSE(pred = pred_price_diff, obs = final$price_diff)

hist(pred_price_diff, xlim = range(-1:2), ylim = range(0:20),
    breaks = 18, col = "steelblue", xlab = "Predicted Price Difference ($)",
    main = "Linear Regression Predictions")
```



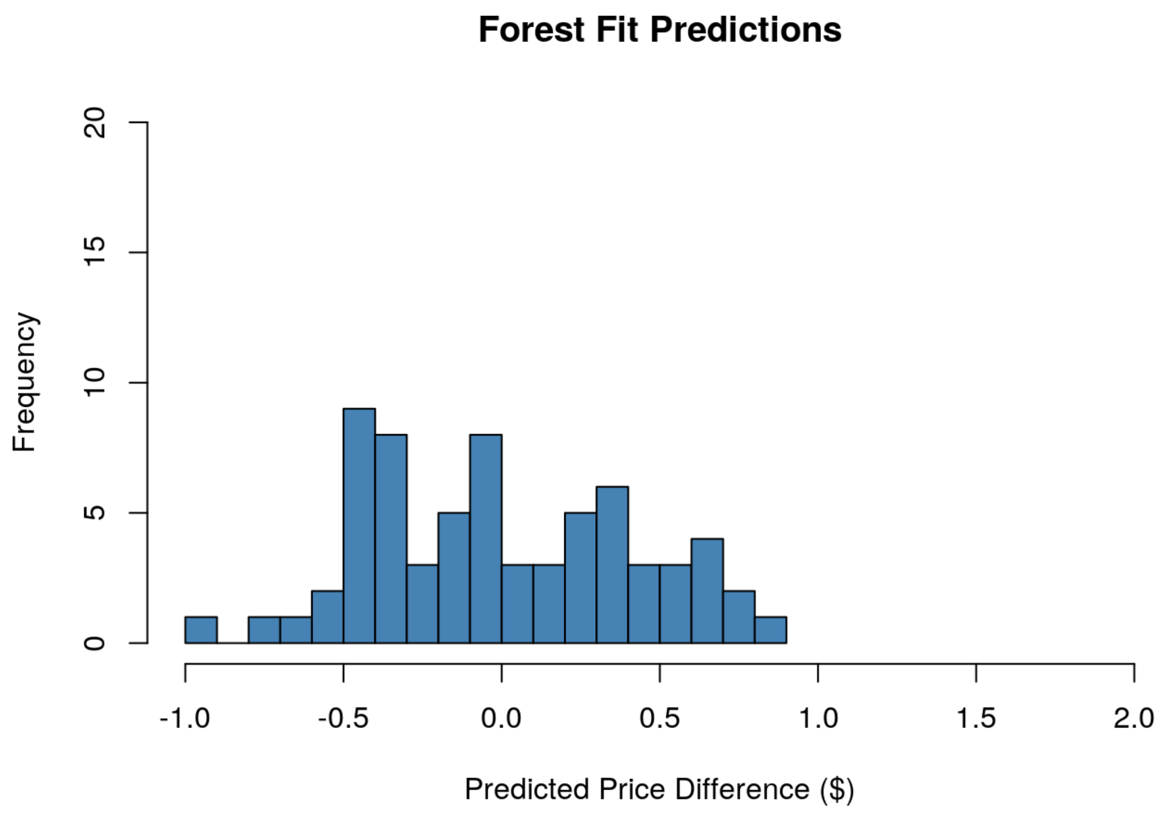
```
set.seed(80085)
train_control <- trainControl(method = "cv", number = 20)
train_price_diff_lm = train(price_diff ~ mean_income + density,
    data = final, method = "lm", trControl = train_control)

klm_rmse = train_price_diff_lm$results[[2]]
```

Our linear regression model had an RMSE of 0.8946.
Using the K-fold cross-validation, however, the Linear RMSE is 0.737

```
forest_fit = randomForest(price_diff ~ mean_income + density,
    data = final)
forest_pred = predict(forest_fit, data = final)
train_price_diff_forest = train(price_diff ~ mean_income + density,
    data = final, method = "rf", trControl = train_control)
```

```
## note: only 1 unique complexity parameters in default grid. Truncating the grid to 1 .
```

```
kforest_rmse = train_price_diff_forest$results[[2]]

hist(forest_pred, xlim = range(-1:2), ylim = range(0:20), breaks = 18,
    col = "steelblue", xlab = "Predicted Price Difference ($)",
    main = "Forest Fit Predictions")
```



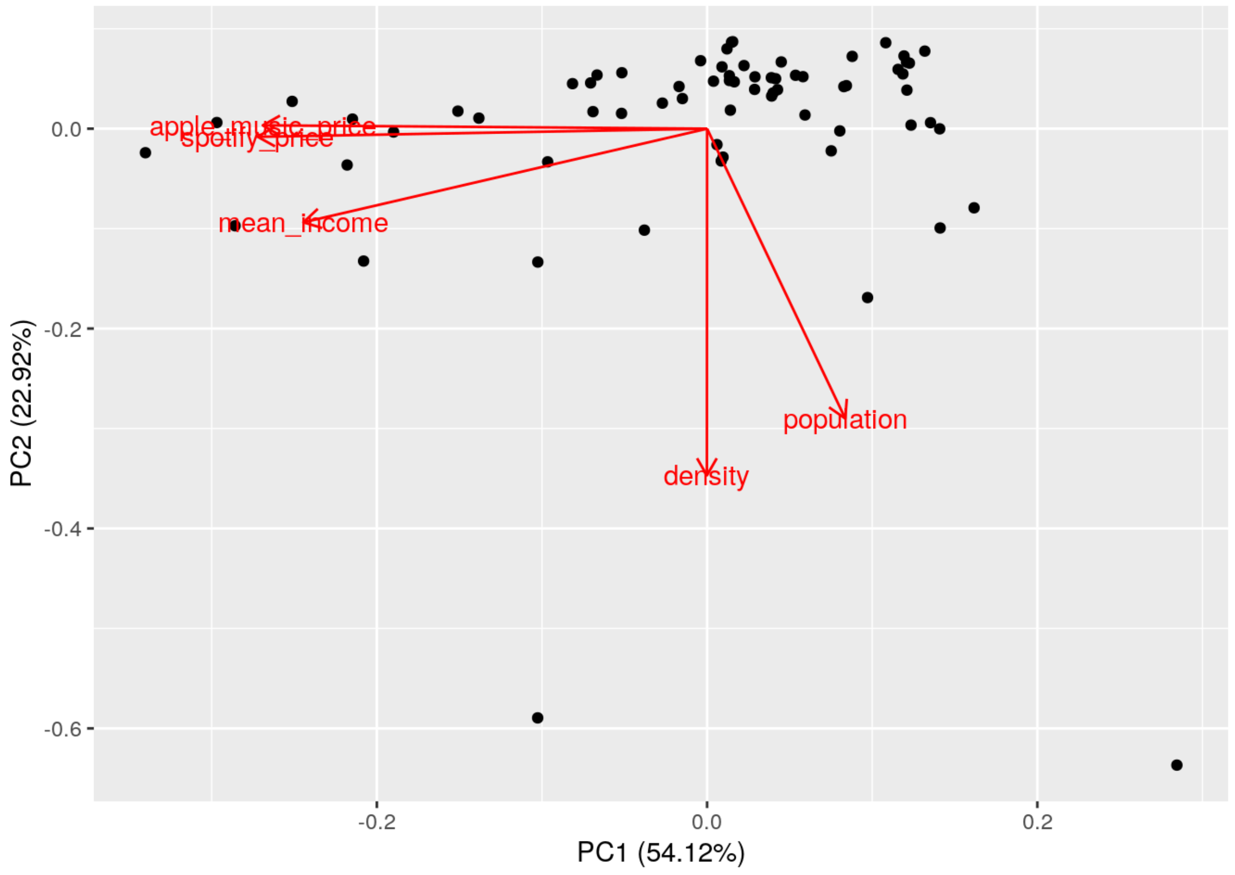
Using the K-fold cross-validation on our Forest Fit, we got an RMSE of 0.7698.

Even thought the RMSE for the linear regression model was high, the k-fold cross validation results in a smaller RMSE for the linear model than the forest model. The k-fold cross validation is a method similar to the train/test algorithm we tested earlier. Our cross validation separates the data into 20 groups. It then iterates through the groups, choosing 1 to be a testing group and the other 19 to be the training group. This provides 19 different fits from which we may choose the best performing model. This is why the accuracy increases from the linear regression to the k-fold cross validation. The linear k-fold model seems to perform best.

PCA Analysis

For the PCA analysis, we have chosen to use mean income, population, spotify price, apple music price, and density.

```
pca_price = final %>%
    dplyr::select(population, mean_income, spotify_price, density,
        apple_music_price) %>%
    scale() %>%
    prcomp()

autoplot(pca_price, data = final, loadings.label = TRUE, loadings = TRUE)
```



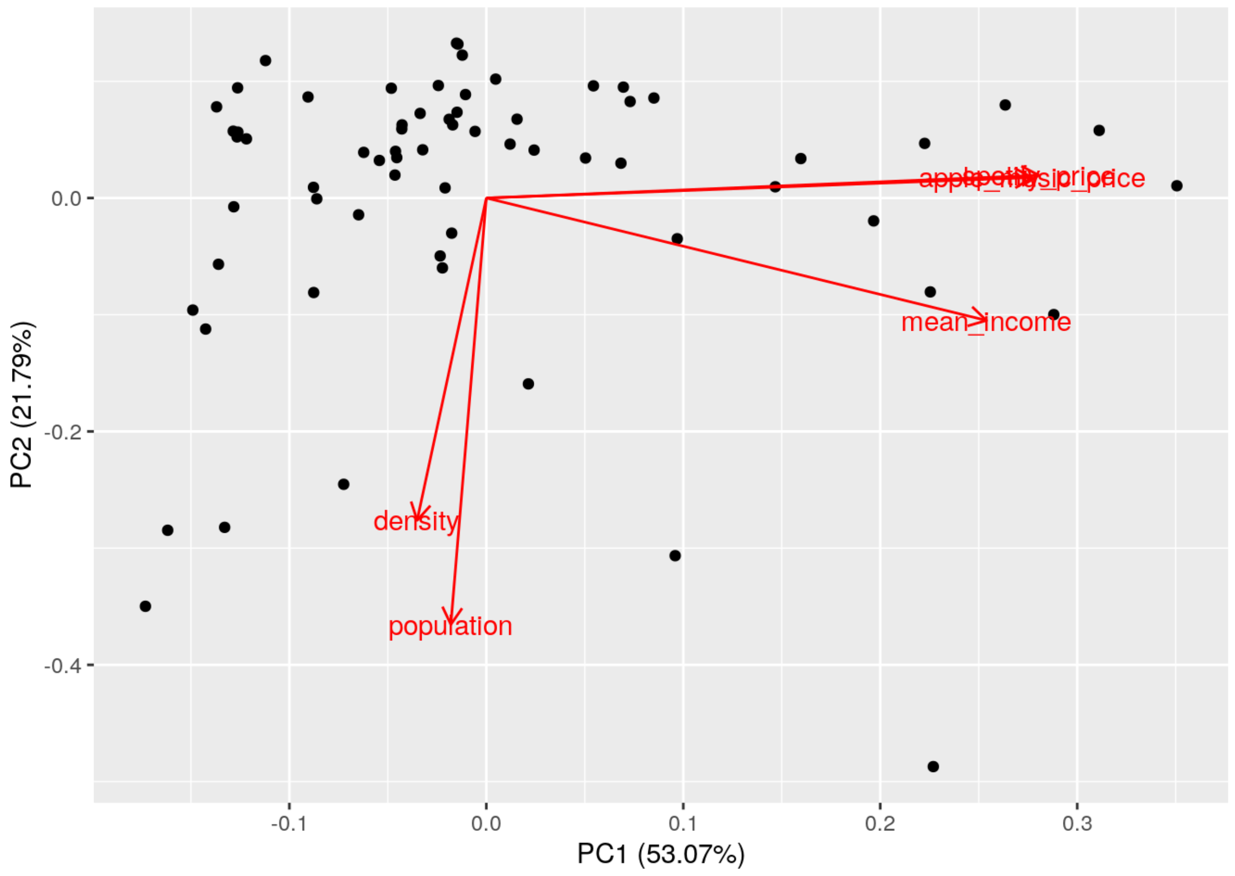
From the graph we see that PC1 is influenced by mean income, apple music price, and spotify price. This variable predicts the largest amount of variability in the data. This is fairly intuitive since these three variables are directly and fairly closely related. The PC2 is influenced by income and population. An interesting observation that can be made from the graph is that density has no relationship with apple music or Spotify price and mean income has no relationship with the population. In conclusion, the variables influencing the first principal component have almost no relation to the variables influencing the second principal component.

```
pca_price_per_explained = pca_price$sdev^2/sum(pca_price$sdev^2)
pca1_2 = round((pca_price_per_explained[1] + pca_price_per_explained[2]) *
    100, 2)
```

The first two principal components campture around 77.04% of the variance in the data. This is a fairly high number and suggests that they have a significant influence on the price difference between the two music streaming applications.

Furthermore, we have removed two outliers towards the bottom of the graph (India and Malta) to get the following PCA plot.

```
pca_price_2 = final %>%
    filter(country != "India") %>%
    filter(country != "Malta") %>%
    dplyr::select(population, mean_income, spotify_price, density,
        apple_music_price) %>%
    scale() %>%
    prcomp()

final_mod = final %>%
    filter(country != "India") %>%
    filter(country != "Malta")

autoplot(pca_price_2, data = final_mod, loadings.label = TRUE,
    loadings = TRUE)
```



We observe that the principal components are still influenced by the variables and explain roughly the same amount of variability. A major change that occurred with the removal of the outliers: the values of PC1 switched from mostly negative to mostly positive.

References

https://howtospotify.com/spotify-premium-price/ (https://howtospotify.com/spotify-premium-price/)
https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population_density
(https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population_density)
https://www.kaggle.com/datasets/statchaitya/country-to-continent?resource=download
(https://www.kaggle.com/datasets/statchaitya/country-to-continent?resource=download)
https://www.cashnetusa.com/blog/which-countries-pay-most-and-least-apple-music/

(https://www.cashnetusa.com/blog/which-countries-pay-most-and-least-apple-music/)
https://worldpopulationreview.com/country-rankings/median-income-by-country
(https://worldpopulationreview.com/country-rankings/median-income-by-country)

Acknowledgements

We did the project as a group, but each of us specialized in one part:
Sam - Logistic Regression
Reuben - Tree-Based Classifiers
Avi - Regression
Karim - PCA