Alice Yang
Pascal Wallisch
Principles of Data Science
13 May 2024

Capstone Project

Introduction

Dimension Reduction:

For dimension reduction, I **standardized** the required data for specific questions first. Then, I performed the **PCA** (principal component analysis) and drew out several components according to the requirements of the questions. At last, I used the corresponding eigenvector multiplying the original data(Z-Scored) to get the new data in the dimension I needed.

Data Cleaning:

Since no missing data is included in the raw data, I kept all of them without using the data cleaning method. For the zeros in the 'popularity' column, I kept them as well and regarded them as low-popularity songs as Spotify calculates popularity factoring in some scale of time.

Data Transformation:

I imported the data using pandas and then the data was stored as a panda dataframe. Also, I converted the index of data for easier calculations as follows since the raw data file indexes from 1, but Python indexes from 0.

```
data = pd.read_csv("spotify52kData.csv")
#index
data.reset_index(drop=True)
```

Figure 1: Convert the index of csv file

Questions

1) Consider the 10 songs features duration, danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, and tempo. Is any of these features reasonably distributed normally? If so, which one?

We plotted the distribution of all of these 10 features respectively.

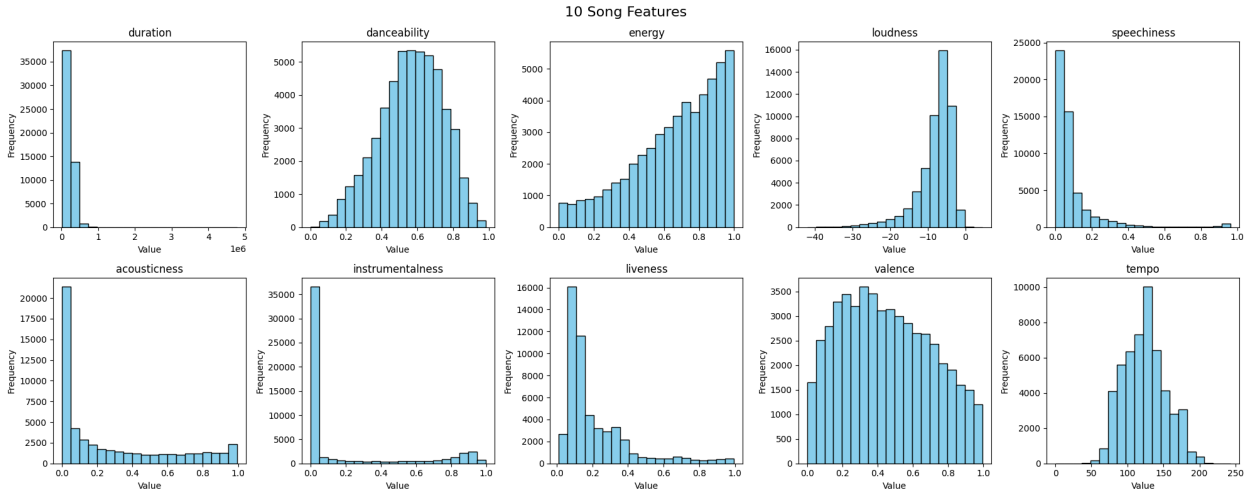


Figure 2: Distribution of 10 songs features

Only the graphs of **danceability** and **tempo** are distributed normally. Because these two graphs perform a symmetric, unimodal, and asymptotic feature.

2) Is there a relationship between song length and popularity of a song? If so, is the relationship positive or negative?

There is a negative relationship between song length and popularity of a song. From the scatter plot in Figure 3 below, we can see that the relationship is **negative** with a correlation coefficient of -0.05.

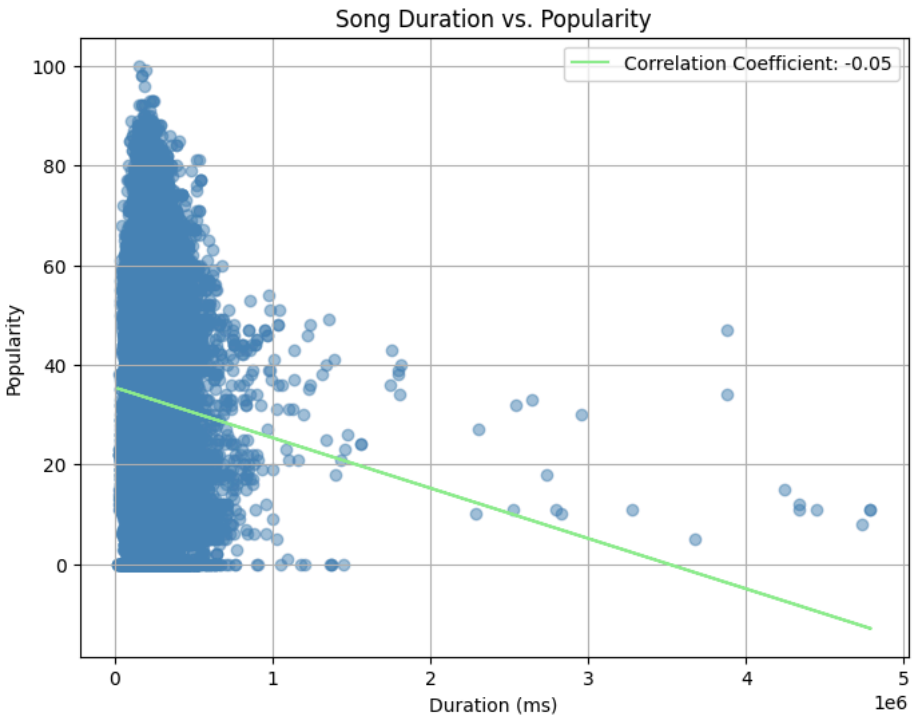


Figure 3: Relationship between duration and popularity

3) Are explicitly rated songs more popular than songs that are not explicit?

Since the 'explicit' column only contains 'True' and 'False' values, which means that it is categorical, while the popularity is numerical, we decided to use a nonparametric test-MannWhitneyU. Also, the distribution of two samples might **not** be normally distributed as shown in Figure 4.

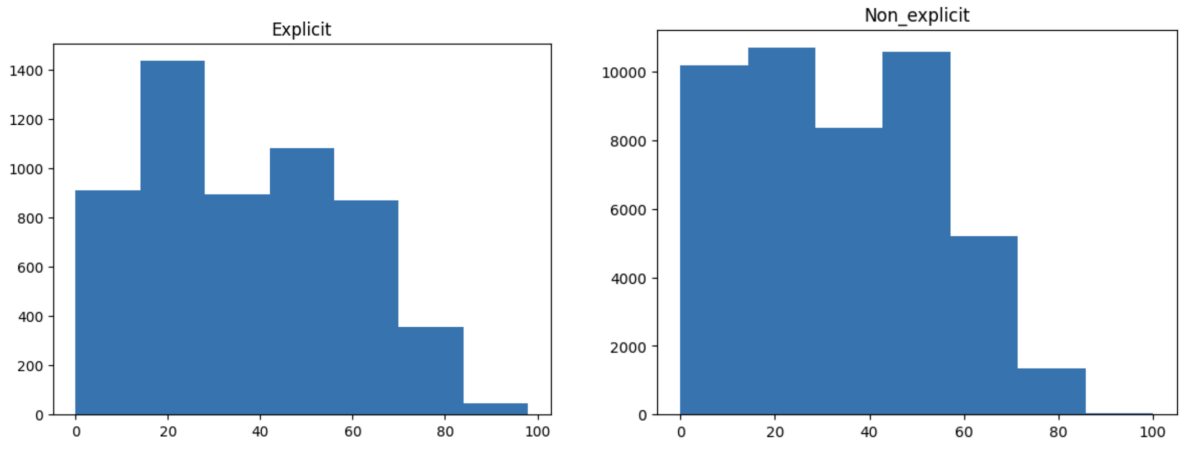


Figure 4: Distribution of explicit and non-explicit songs

The **null hypothesis** is that explicit songs are equally popular as nonexplicit songs. That is to say, the median of popularity for explicit and nonexplicit songs is the same.

The **alternative hypothesis** is that explicit songs are more popular than nonexplicit songs. The median of popularity for explicit songs is higher than that of nonexplicit songs.

Then I used a method in the scipy.stats package to perform a one-tailed **MannWhitney U** test between explicit songs and nonexplicit songs as shown below.

```
explicit_popularity = data[data['explicit'] == True]['popularity']
non_explicit_popularity = data[data['explicit'] == False]['popularity']

stat, pvalue1 = mannwhitneyu(explicit_popularity, non_explicit_popularity, alternative='greater')

alpha = 0.05
print(pvalue1)

if pvalue1 < alpha:
    print("The difference in popularity between explicit and non-explicit songs is statistically significant.")
else:
    print("There is no statistically significant difference in popularity between explicit and non-explicit songs.")
```

```
1.5339599669557339e-19
The difference in popularity between explicit and non-explicit songs is statistically significant.
```

Figure 5: Calculation of p-value test for question 3

As we can see in Figure 5, the p-value is approximately 0. Since we chose to use an alpha value of 0.05 as significance level, as $p < 0.05$, we could **reject** the null hypothesis. Therefore, the difference in popularity between explicit and non-explicit songs is statistically significant.

4) Are songs in major key more popular than songs in minor key?

For this question, we chose to look at the 'Mode' column. Since it is a binary categorical variable where '1' represents the song in major and 0 represents the song in minor, we decided to use a nonparametric test-MannWhitneyU as well.

The **null hypothesis** is that songs in major key are equally popular as songs in minor key. That is to say, the median of popularity for songs in major key or minor key is the same.

The **alternative hypothesis** is that there is a difference in popularity between songs in major key and songs in minor key. The median popularity for songs in major key is different from songs in minor key.

Similar to what we did in question 3, I used a method in the scipy.stats package to perform a one-tailed **MannWhitney U** test between classical arts and modern arts.

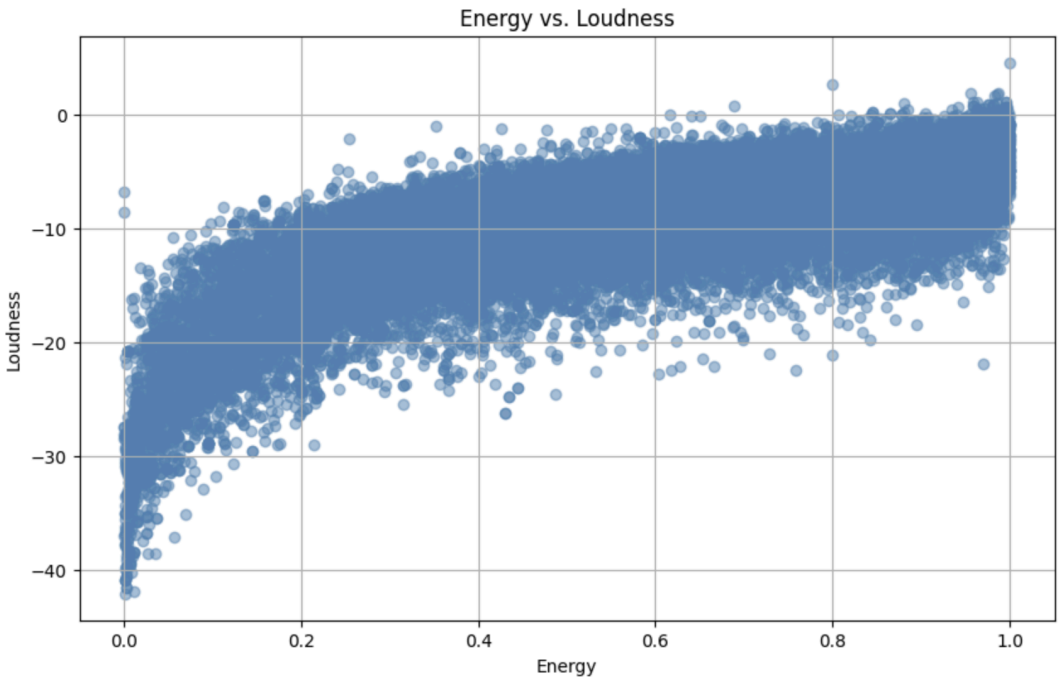
```
major_popularity = data[data['mode'] == 1]['popularity']
minor_popularity = data[data['mode'] == 0]['popularity']

stat2, pvalue2 = mannwhitneyu(major_popularity, minor_popularity)

print(f"P-value: {pvalue2:.10f}")

if pvalue2 < alpha:
    print("The difference in popularity between songs in major and minor keys is statistically significant.")
else:
    print("There is no statistically significant difference in popularity between songs in major and minor keys.")
```

```
P-value: 0.0000020175
The difference in popularity between songs in major and minor keys is statistically significant.
```

Figure 6: Calculation of p-value test for question 4

As we can see in Figure 6, the p-value is approximately 0 as well. Since we still chose to use an alpha value of 0.05, as $p < 0.05$, we chose to **reject** the null hypothesis. Therefore, the difference in popularity between songs in major key or minor key is statistically significant and songs in major key is more popular.

5) Energy is believed to largely reflect the "loudness" of a song. Can you substantiate (or refute) that this is the case?




```
correlation between energy and loudness:  0.7748808291850185
```

Figure 7: correlation between energy and loudness

We made a scatter plot for the correlation between energy and loudness of songs as shown in Figure 7. Since the correlation coefficient we calculated is around 0.7748, which is close to 1, we concluded that the energy and loudness of a song are **closely** related, which means energy is largely reflects the loudness of the song.

6) Which of the 10 song features in question 1 predicts popularity best? How good is this model?

We used a **for loop** to calculate the correlation coefficient for all of the 10 songs features as shown in Figure 8.

```
correlation coefficient between popularity and duration: -0.054651195936376386
correlation coefficient between popularity and danceability: 0.03715781135740383
correlation coefficient between popularity and energy: -0.05592469066526968
correlation coefficient between popularity and loudness: 0.06021003481482005
correlation coefficient between popularity and speechiness: -0.04853267708421366
correlation coefficient between popularity and acousticness: 0.026233391738937697
correlation coefficient between popularity and instrumentalness: -0.14497227053733283
correlation coefficient between popularity and liveness: -0.04384602990976282
correlation coefficient between popularity and valence: -0.03576878910256896
correlation coefficient between popularity and tempo: -0.0026318311756676612
The feature that best predicts popularity is:  instrumentalness
Correlation coefficient:  -0.14497227053733283
RMSE: 21.456607642834474
R-squared: 0.02110466494367702
```

Figure 8: Correlation between popularity and 10 songs features

The largest **absolute** correlation coefficient we got was 0.145 from **'instrumentalness'**, which means instrumentalness predicts popularity best. To evaluate this model, we did some further calculations and ran a linear regression between instrumentalness and popularity. Then, we got a **r squared** of **0.02**, which is small and shows that the model is weak as it does not account for all random variables and the variance.

7) Building a model that uses all of the song features in question 1, how well can you predict popularity? How much (if at all) is this model improved compared to the model in question 7). How do you account for this?

```
X = data[features]
y = data['popularity']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Build and train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Evaluate the model's performance on the testing data
y_pred = model.predict(X_test)

rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print("Root Mean Squared Error:", rmse)
print("R-squared:", r2)
```

```
Root Mean Squared Error: 21.111348243501684
R-squared: 0.05235411982671212
```

Figure 8: multiple regression model between features and popularity

Similarly, I ran a **multiple regression** since there are several predictor variables. I trained the model using the training data after splitting the raw data. Then, this model is evaluated by calculating the variance, r squared, as shown in Figure 8. Then, we got a **r squared** of **0.05**. Since it is also small, our model is weak and still doesn't account for much of the variance.

8) When considering the 10 song features above, how many meaningful principal components can you extract? What proportion of the variance do these principal components account for?

For this question, we did **feature extraction** and used the dimension reduction method PCA (principal component analysis). First, the data is standardized to zscoredData and we performed principal component analysis to find the eigenvalues. To extract meaningful principal components, we drew the graph of eigenvalues as shown in Figure 9 below.

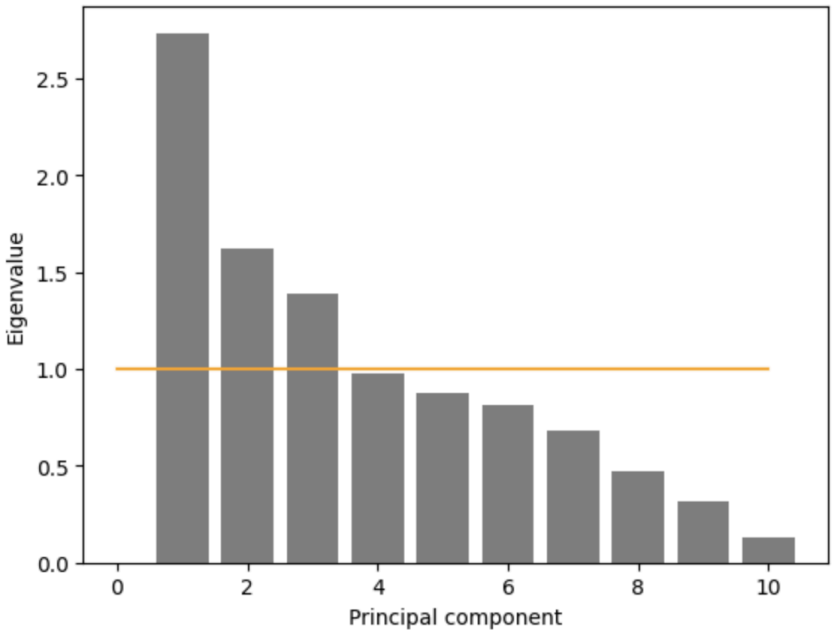


Figure 9: Eigenvalues of principal components

I used **Kaiser criterion**, which is marked as an orange line in Figure 9. Therefore, **three** meaningful principal components are drawn out since they have eigenvalues above 1.

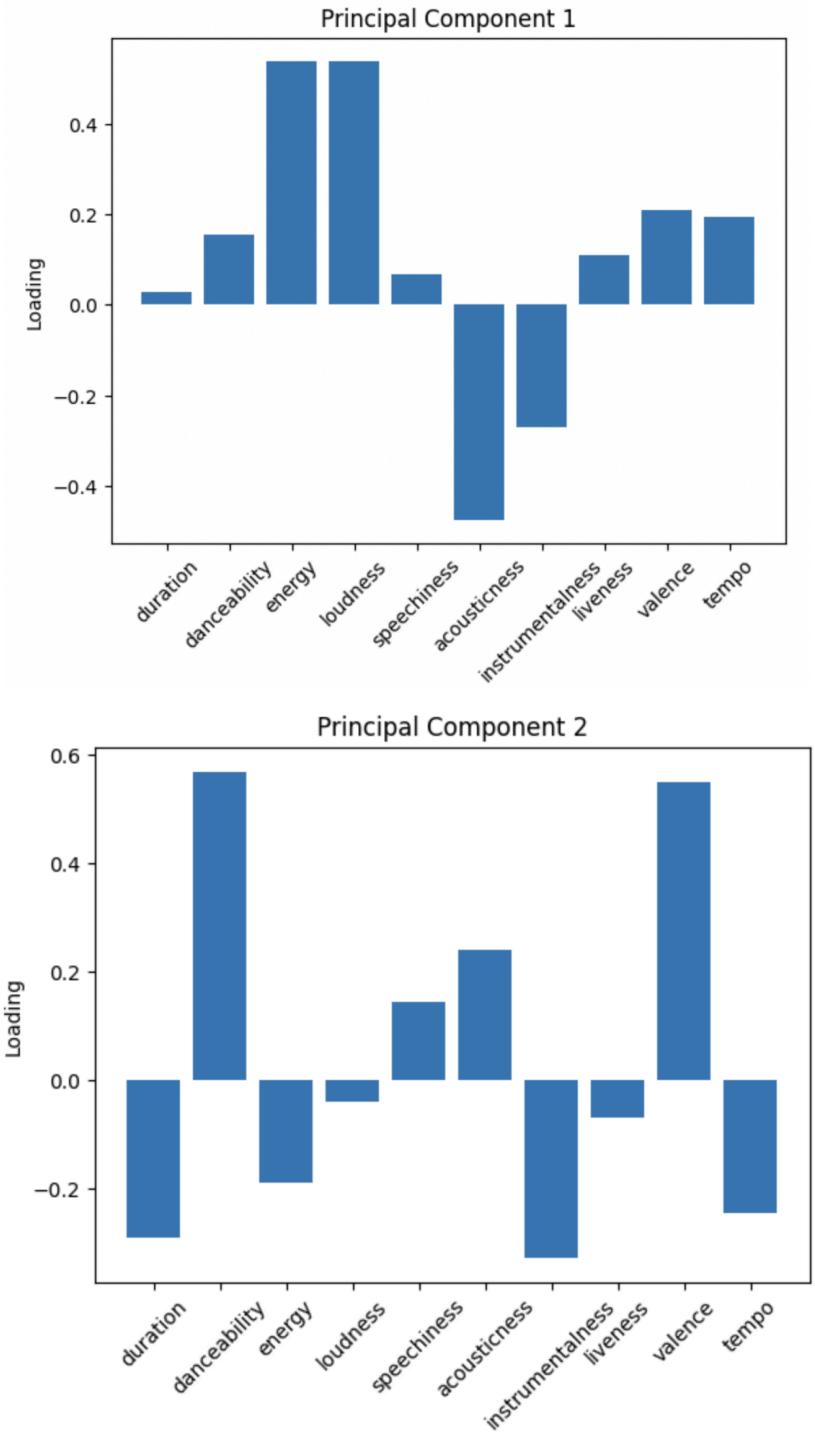
Then, I calculated the percent of variance explained by each feature (corresponds to its eigenvalues) as shown in Figure 10 below. It means that principal component 1, principal component 2, and principal component 3 account for 27.339%, 16.174%, and 13.846 of the variance respectively.

```
zscoredData = stats.zscore(data[features])
pca = PCA().fit(zscoredData)
eigVals = pca.explained_variance_
loadings = pca.components_
rotatedData = pca.fit_transform(zscoredData)
varExplained= eigVals/sum(eigVals)*100
for i in range(len(varExplained)):
    print(varExplained[i].round(3))
```

```
27.339
16.174
13.846
9.796
8.752
8.148
6.783
4.716
3.131
1.316
```

Figure 10: Proportion of variance that each feature account for

Next, we plotted the eigenvectors for each of those three principal components we extracted by Kaiser criterion to examine them as shown in Figure 11.



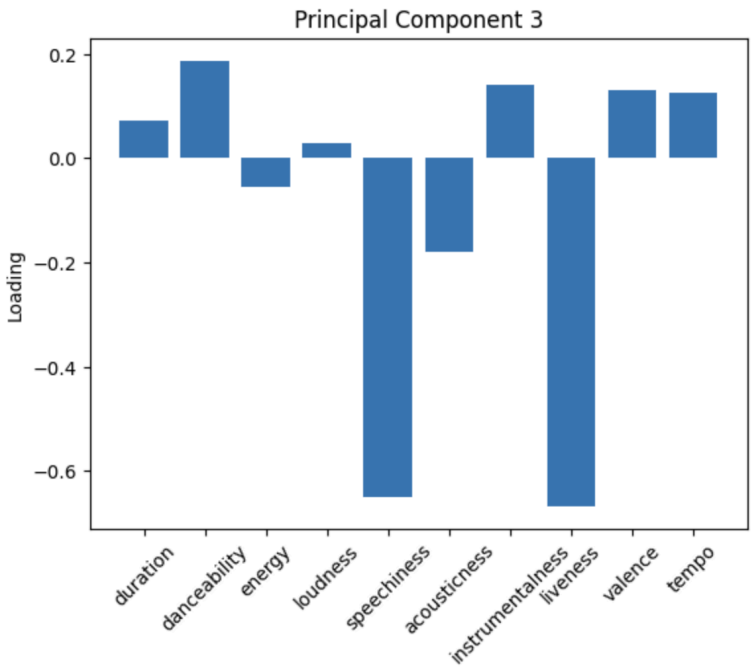


Figure 11: bar plots of loadings for each feature

To interpret the loadings graph, from the graph for principal component 1, just as we previously discussed, energy and loudness are closely correlated. Since the increase in volume makes us excited, we can get more energy from it. Then this may account for the **excitement** of tracks. Also, we can see that dancebility and valence are highly correlated for principal component 2. Since high valence gives off high positive **mood**, which is uplifting and raises the dancibility of the song. For principal component 3, speechiness and liveness is closely correlated. As more words are spoken during the song, it is more likely that the song was live in front of audience instead of recording. We can categorize it as the **narrative** of a song for this component.

9) Can you predict whether a song is in major or minor key from valence? If so, how good is this prediction? If not, is there a better predictor?

Since there are only two classes in the 'mode' column, major and minor key, and this column is binary, we performed a **logistic regression** to predict. Then, we did a train test split to avoid overfitting and visualized it in Figure 12 below.

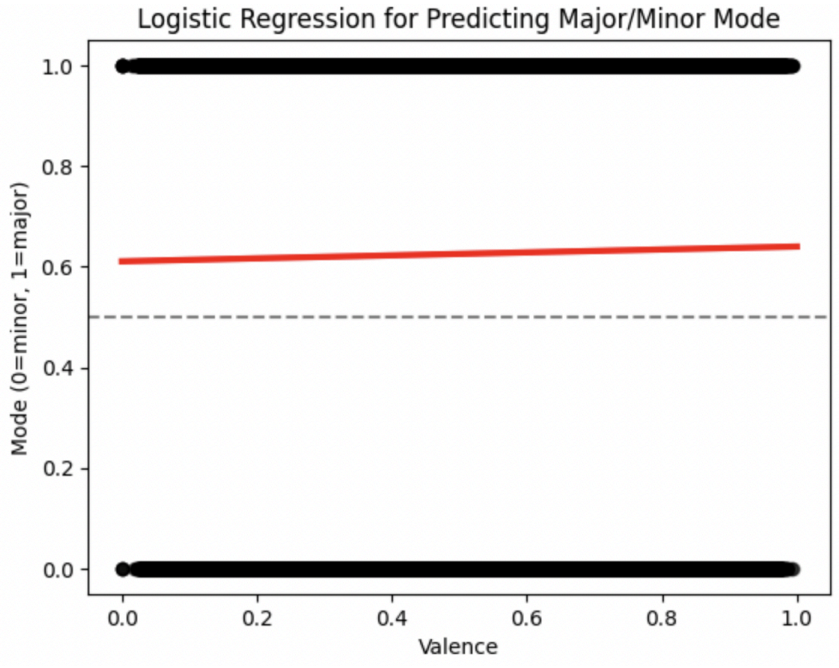


Figure 11: logistic regression for question 9

From the graph, since the slope of our regression line is very small, and the dots we have do not clearly show us the difference between major and minor key, we concluded that valence is **not** a good predictor to predict whether a song is in major or minor key. To further verify our model, I calculated the **accuracy** of the model and got a small value of 0.619. This means that only this much of predictions are correct when using valence as the predictor.

Then, we iterated over each feature to perform logistic regression for all of them, and stored the auroc value of auc roc model in a dictionary I created. In this way, we found out the auroc scores for all of them and got the best predictor for predicting major or minor key is **speechiness** with the largest auroc score of 0.5702 as shown in Figure 12. Therefore, speechiness is a greater predictor to estimate whether a song is in major or minor key.

```
AUROC for duration: 0.490304299616636
AUROC for danceability: 0.5518890213117899
AUROC for energy: 0.5489876119308623
AUROC for loudness: 0.5306242753192909
AUROC for speechiness: 0.5702059807111586
AUROC for acousticness: 0.5514625078380431
AUROC for instrumentalness: 0.5346514449531318
AUROC for liveness: 0.49989453841554893
AUROC for valence: 0.500681068480768
AUROC for tempo: 0.50765502460718
The feature with the largest AUROC score is 'speechiness' with a score of 0.5702059807111586.
```

Figure 12: AUROC for 10 songs features

10) Which is a better predictor of whether a song is classical music – duration or the principal components you extracted in question 8?

First, we need to convert the qualitative genre label into a **binary** numerical label. As shown in Figure 13, we used a function to create a panda series 'is_classical' by making 'classical' be 1 and all other 'non-classical' be 0.

```
data['is_classical'] = data['track_genre'].apply(lambda x: 1 if x == 'classical' else 0)
```

Figure 13: convert genre into binary label

Then, we performed a **logistic regression** model between duration and classical music. Then we assessed the model using auroc curve and got the value of 0.558 when using duration as a predictor as shown in Figure 14 below.

```
X_duration = data[['duration']]
y = data['is_classical']

# Split the data into training and testing sets for duration
X_train_dur, X_test_duration, y_train, y_test = train_test_split(X_duration, y, test_size=0.2, random_state=42)

# Train logistic regression model with duration as the predictor
model_duration = LogisticRegression()
model_duration.fit(X_train_dur, y_train)

# Predict probabilities on the test set for duration
y_pred_prob_duration = model_duration.predict_proba(X_test_duration)[:, 1]

# Calculate AUC-ROC score
auc_roc_duration = roc_auc_score(y_test, y_pred_prob_duration)
print("AUC-ROC using duration as predictor:", auc_roc_duration)
```

```
AUC-ROC using duration as predictor: 0.5582064656517253
```

Figure 14: Logistic regression model between duration and is_classical

Similarly, we did another logistic regression for the principal components we extracted from question 8. As we evaluated the model, three principal components were drawn out and pca was performed for standardized data we processed before. At last, we got the auc-roc score of 0.9404 for this model as shown in Figure 16 below.

```
X = data[features]
y = data['is_classical']

pca = PCA(n_components=3)
X_pca = pca.fit_transform(zscoredData)

X_train_pca, X_test_pca, y_train, y_test = train_test_split(X_pca, y, test_size=0.2, random_state=42)

# Train logistic regression model with duration as the predictor
model_pca = LogisticRegression()
model_pca.fit(X_train_pca, y_train)

# Predict probabilities on the test set for duration
y_pred_prob_pca = model_pca.predict_proba(X_test_pca)[:, 1]

# Calculate AUC-ROC score
auc_roc_pca = roc_auc_score(y_test, y_pred_prob_pca)
print("AUC-ROC using principal components as predictor:", auc_roc_pca)
```

```
AUC-ROC using principal components as predictor: 0.9404150233201866
```

Figure 15: Logistic regression model between principal components and is_classical

Since the auc roc score of the logistic regression model for three principal components we extracted and classical music is much larger larger, we concluded that **those principal components** are a greater predictor of whether a song is classical music or not.

Extra Credit

1) Relationship between the key and popularity of a song

Since it's commonly believed that certain keys evoke different emotions or have different effects on listeners, we wanted to know that if there's certain keys that would make songs more popular. Therefore, we generated a bar plot of popularity across different key as shown in Figure 16. For this graph, we used groupby method to calculate the average popularity of each key.

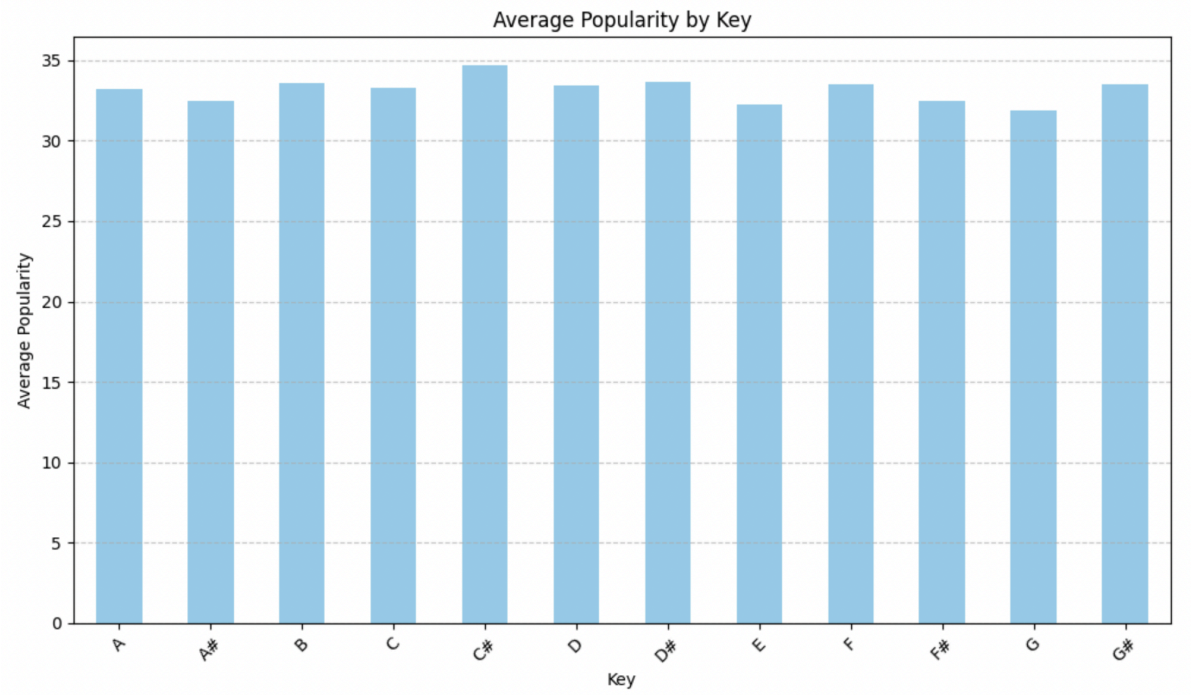


Figure 16: bar plot of average popularity across different keys

As we can see in the graph, there is no noticeable difference between the popularity of songs in different keys. However, in order to analyze it statistically, we calculated and compared the standard deviations of the popularity values for each key group. Similarly, we used groupby method as well to generate the standard deviation of each key as shown below in Figure 17 below.

```
# Calculate the standard deviation of popularity for each key group using groupby method
key_std_dev = data.groupby('key')['popularity'].std()

# Map the numeric key values back to their corresponding musical keys
key_std_dev.index = key_std_dev.index.map(key_mapping)

# Print the standard deviation for each key
print("Standard Deviation of Popularity for Each Key:")
print(key_std_dev)
```

```
Standard Deviation of Popularity for Each Key:
key
A     21.423002
A#    22.131801
B     21.043936
C     22.628343
C#    21.244904
D     22.097113
D#    21.592120
E     21.093304
F     23.172387
F#    21.425097
G     22.033170
G#    22.163202
```

Figure 17: standard deviation of popularity for each key

Since there's no significant difference in standard deviation between different key groups, it suggests that the low variability in popularity differs across keys. Therefore, the difference in average popularity between keys is **not** statistically significant. Contrary to what we suspected, the key of songs does not really affect the popularity of songs.