Adrian Yan

Pascal Wallisch

Principles of Data Science

14 May 2024

Final Capstone Project

I began the data cleaning preprocessing step by first finding the columns with numerical data and filling in the missing data with the median of each column. I did the same for the categorical columns but instead of using the mean, I filled in the missing data with the most frequent value, or the mode. I finished by removing the duplicate rows from the data using drop_duplicates(). The random number generator is also seeded with my N#18196483 in the beginning of the code file and train test splits are used with random_states of '18196483' throughout the code.

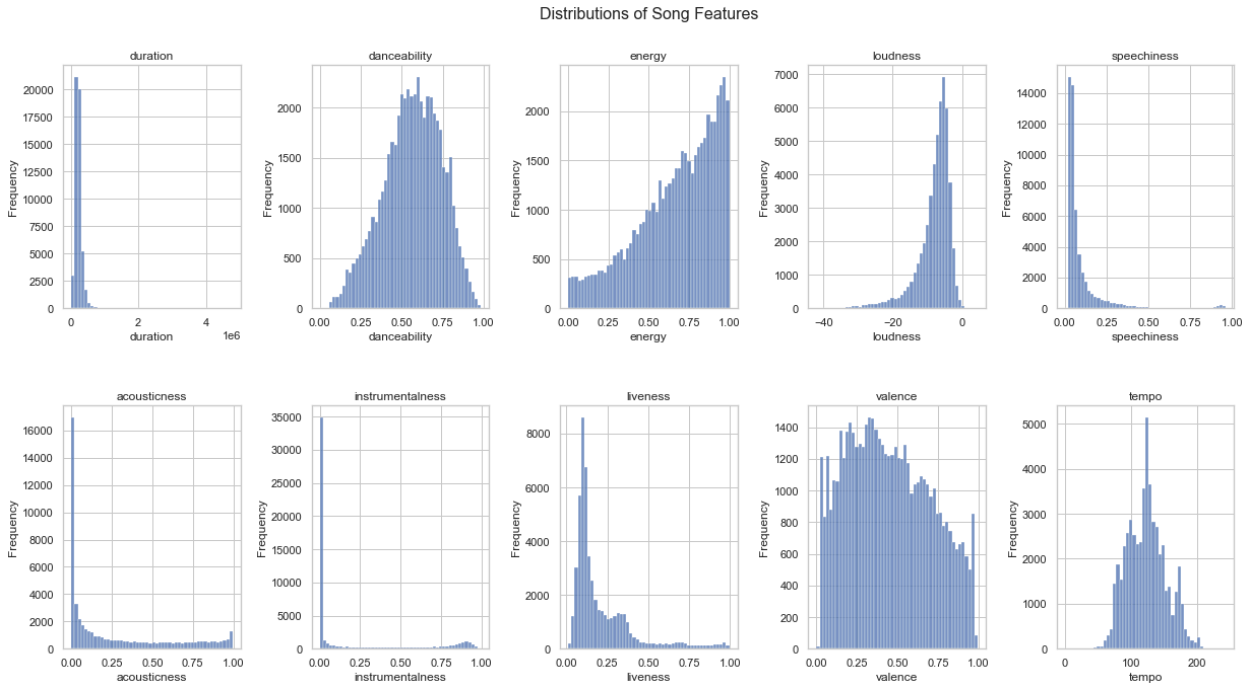
```
random.seed(18196483)
data = pd.read_csv('spotify52kData.csv')
data.describe()

#Fill missing numerical data with median
num_cols = data.select_dtypes(include=['float64', 'int64']).columns
data[num_cols] = data[num_cols].fillna(data[num_cols].median())

#Fill missing categorical columns with mode
cat_cols = data.select_dtypes(include=['object', 'bool']).columns
for col in cat_cols:
    data[col] = data[col].fillna(data[col].mode()[0])

#Return with duplicate rows removed
data = data.drop_duplicates()
```

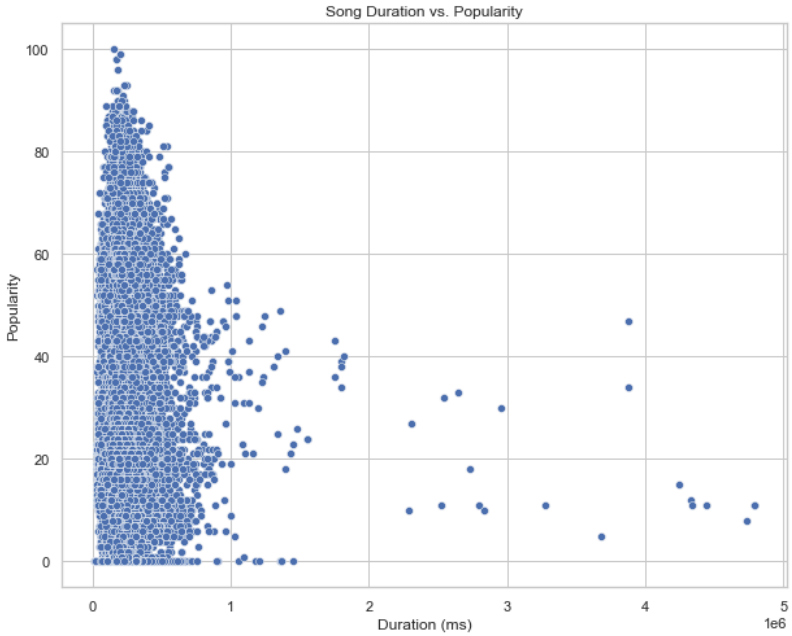
1) In order to first visualize the distributions of the 10 song features listed (duration, danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence and tempo), I implemented a 2x5 figure with histograms for each feature using pyplot. The X axis for each histogram contains the feature and the Y axis contains the frequency.



From first glance, it did not seem that any of the features were reasonably normally distributed. I then conducted a Shapiro-Wilk Test and for all features got a $p < 0.05$, indicating that none of the features were normally distributed.

	Statistics	p-value
duration	0.630809	0.0000000000000000000000000000000000000000000000000...
danceability	0.989345	0.0000000000000000000000000000000000000000000000000...
energy	0.931866	0.0000000000000000000000000000000000000000000000000...
loudness	0.833006	0.0000000000000000000000000000000000000000000000000...
speechiness	0.496423	0.0000000000000000000000000000000000000000000000000...
acousticness	0.797084	0.0000000000000000000000000000000000000000000000000...
instrumentalness	0.596098	0.0000000000000000000000000000000000000000000000000...
liveness	0.743081	0.0000000000000000000000000000000000000000000000000...
valence	0.965998	0.0000000000000000000000000000000000000000000000000...
tempo	0.989757	0.0000000000000000000000000000000000000000000000000...

2) To see if there is a relationship between song length and popularity, I constructed a scatter plot with song duration (ms) on the X-axis and popularity on the Y-axis.



I then used the corr() function to compute the Pearson correlation coefficient between song duration and popularity and got r = -0.05465, which means there is an extremely small negative relationship between the two variables, since the correlation coefficient was negative, but extremely close to 0. It can be concluded that there is a very weak negative correlation between song duration and popularity.

3) We need now to see if explicitly rated songs are more popular than non-explicit songs. Because song popularity isn't normally distributed, we cannot opt to use a parametric t-test. I opted to use the Mann-Whitney U Test, a non parametric test since we don't have a normal distribution, and it is good for two independent groups (explicit and non-explicit songs). This test is also robust in that it is not prone to extreme values, making it suitable for this comparison. Our null hypothesis is that the distribution of popularity scores for explicit songs is equal to or less than that of non-explicit songs. Our alternative hypothesis is that the distribution of popularity scores for explicit songs is greater than that of non-explicit songs. I achieved the following result:

```
Mann-Whitney U Test Statistic: 139361273.5, P-value: 1.5339599669557339e-19
```

Therefore, since $p < 0.05$, we reject the null hypothesis in favor of the alternative hypothesis and can conclude that explicitly rated songs are more popular than non-explicit songs.

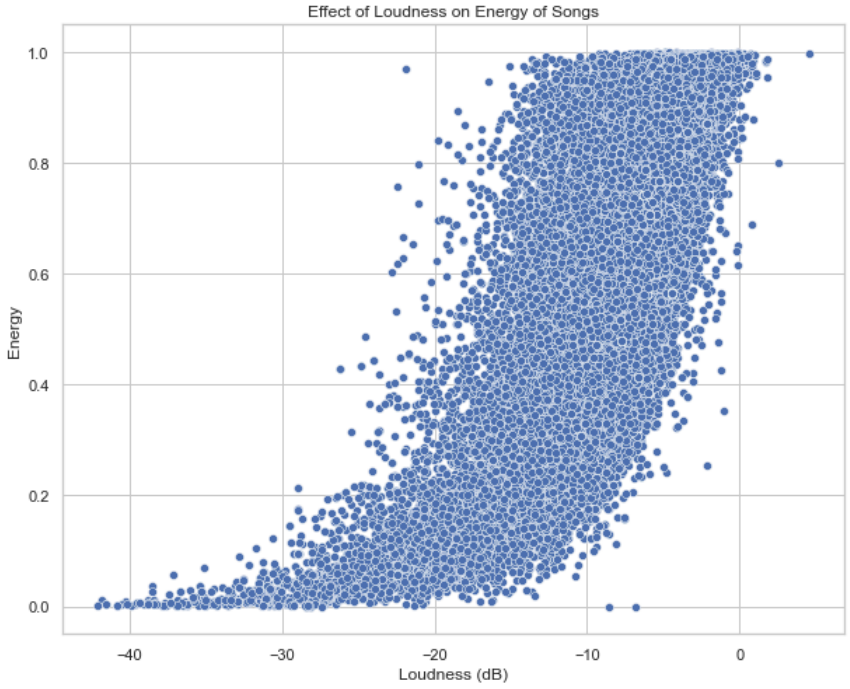
4) To see if songs in major key are more popular than songs in minor key, we can again use the Mann-Whitney U Test for the same reasons. We have two independent groups and the test is robust to extreme values, making it suitable for this comparison. The null hypothesis is that the distribution of popularity scores for songs in major key is equal to or less than that of the songs

in minor key. Our alternative hypothesis is that the distribution of popularity scores for songs in major key is greater than that of the songs in minor key. This time, running the test resulted in:

```
Mann-Whitney U Statistic: 309702373.0, P-value: 0.9999989912386331
```

Since $p > 0.05$, we do not reject the null hypothesis and cannot conclude that songs in major key are more popular than songs in minor key.

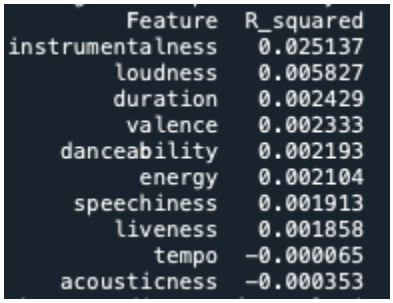
5) To determine if the energy of a song reflects the loudness of the song, I constructed a scatterplot with loudness (dB) on the X-axis and energy level on the Y-axis using pyplot.



Using the corr() function, I calculated the Pearson correlation coefficient between loudness and energy and got $r = 0.775$, meaning there is a relatively strong positive correlation between loudness and energy of a song. This substantiates the claim that energy largely reflects the "loudness" of a song.

6) To evaluate which of the 10 features from problem 1 predict popularity best, we run a simple linear regression model on each of the features and see which model comes out with the greatest

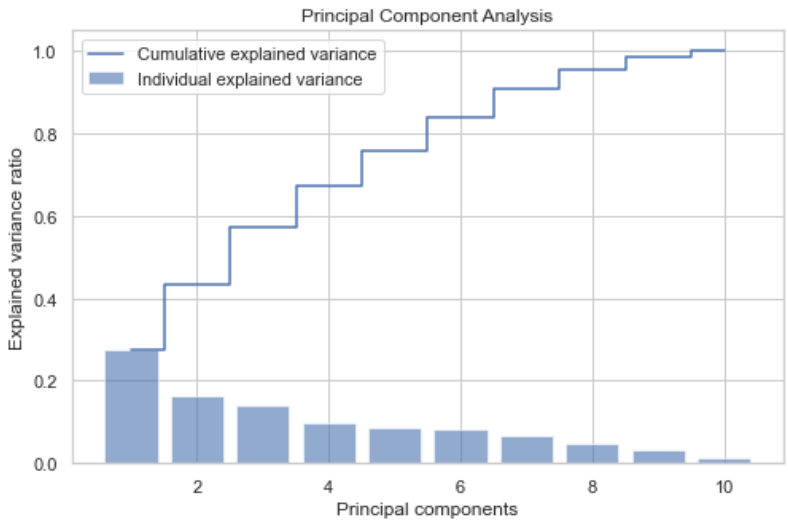
R^2 (coefficient of determination), which is the proportion of variance in the popularity that is predictable from the feature. We first split the data into training and testing sets using a random_state of '18196483', which is the seed for my random number generator using my N#. Random splitting ensures that the training and testing datasets are representative of the overall dataset. This is done for each of the 10 features and the R^2 values from each respective linear regression model are recorded each time. From this figure here, we can see that instrumentalness has the highest R^2 value of 0.025137, indicating that this feature out of the ten predicts popularity best. However, this means that instrumentalness only accounts for about 2.5% of the variance in popularity, so a model that only uses instrumentalness to predict popularity isn't the best.



Feature	R_squared
instrumentalness	0.025137
loudness	0.005827
duration	0.002429
valence	0.002333
danceability	0.002193
energy	0.002104
speechiness	0.001913
liveness	0.001858
tempo	-0.000065
acousticness	-0.000353

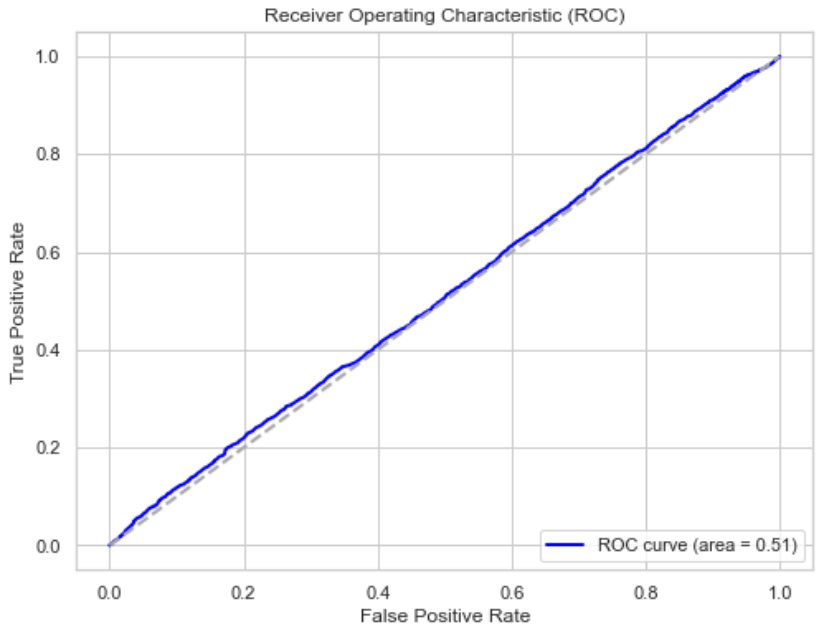
7) Now we want to build a model that uses all of the 10 features to predict song popularity. We can do this by creating a multiple regression model and again, using the R^2 value to determine if the model has improved at all from the previous best model. I again seeded the random_state with my N# in order to split the data into training and test sets. We use model.fit() once again and this time, we get an R^2 value of 0.057. This means that these 10 features account for around 5.7% of the variance in popularity. This is about a 3.2% increase in variance accounted for compared to the model that only uses instrumentalness to predict popularity. The improvement can be attributed to the multidimensional nature of data in real-world scenarios. Each feature might capture only a part of the overall variance, and when combined, they provide a more complete representation of the underlying processes affecting popularity. However, this is still a very low percentage, thus our model still has a low predictive power.

8) To get how many principal components we can extract from these 10 features, we perform principal component analysis. I first used StandardScaler() in order to standardize the data, which is important in PCA as it gives each feature equal importance and PCA is sensitive to the variances of the initial features. I use PCA from sklearn.decomposition to calculate the explained variance ratio per principal component as well as the cumulative explained variance. From this we can make a plot that can help visualize these evaluations:

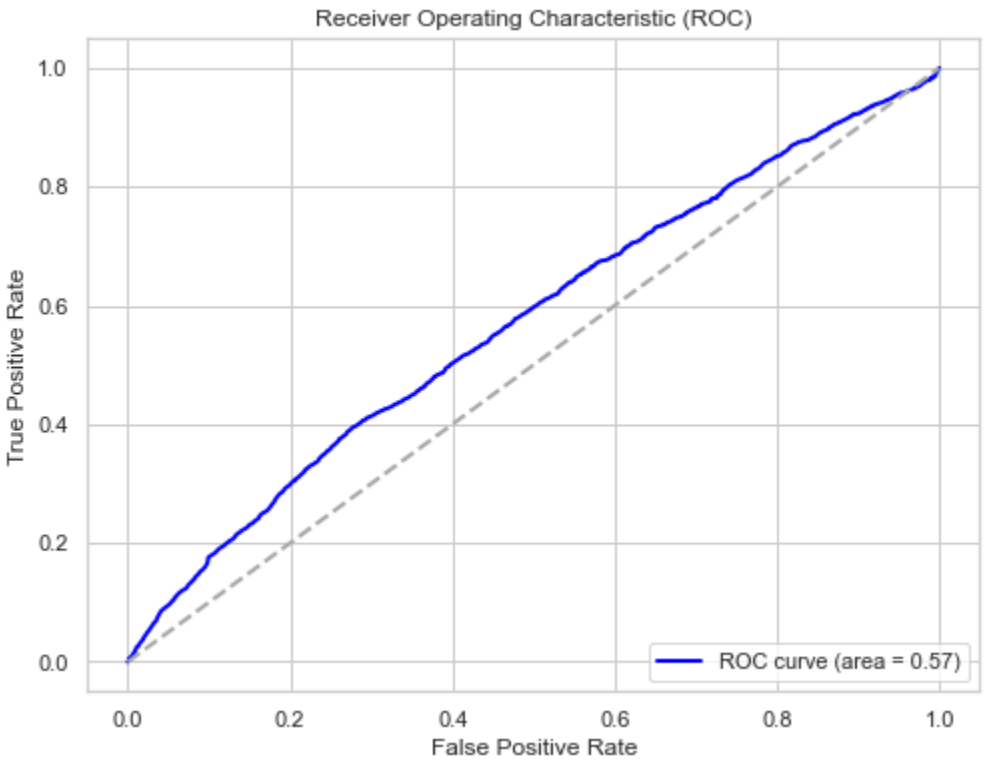


I set my threshold for principal components to be 90%, so I look for the number of principal components that first goes over 90% in the cumulative explained variance. Thus, from the plot we can see to extract 7 meaningful components that account for around 90.8% of the variance.

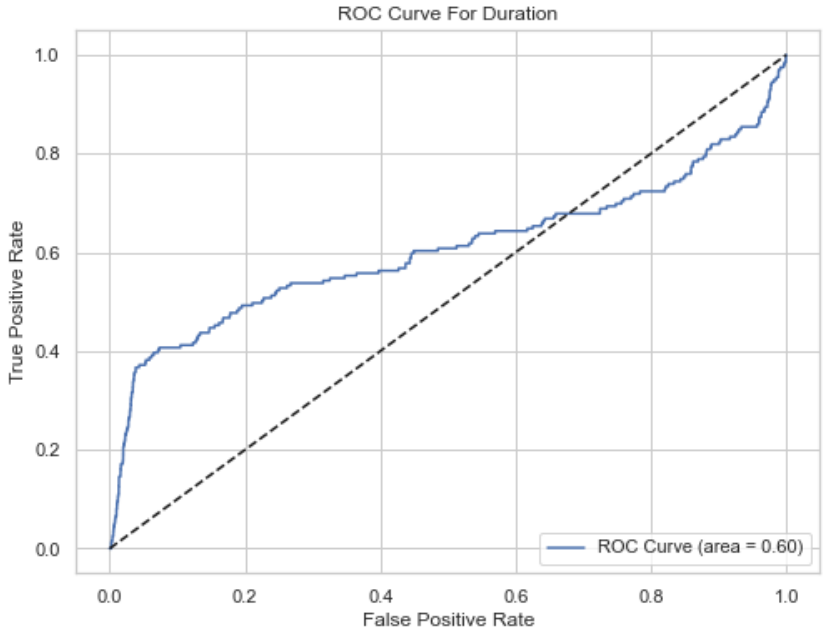
9) Since a song being in major or minor key is a binary dependent variable, I will be using logistic regression in this model to predict major or minor key from valence. We again do a train and test split seeding the random_state with my N#. This time we use LogisticRegression() from sklearn.linear_model and fit our data. To determine how well our model predicts, I use AUC to quantify it. To visualize the ROC, I created a plot with the false positive rate on the X-axis and the True Positive Rate on the Y-axis, and our ROC curve.



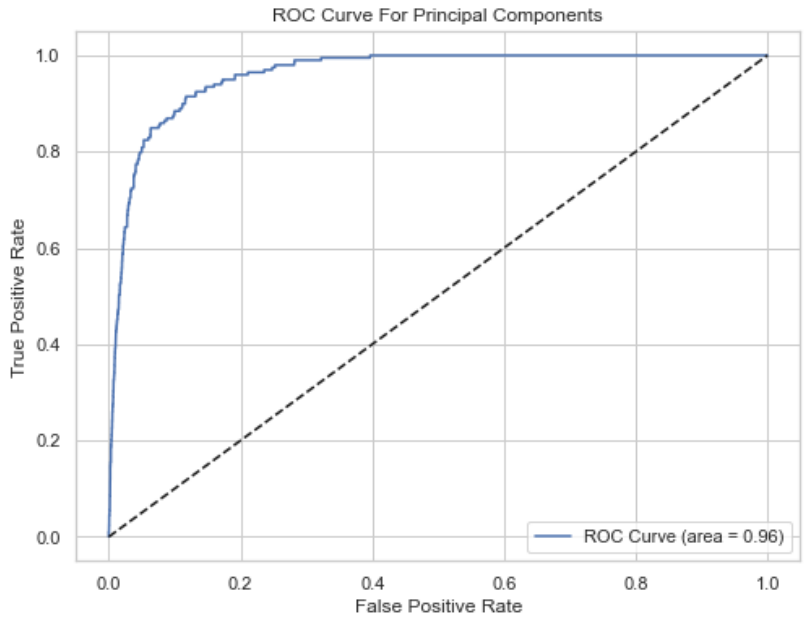
We evaluate that the AUC for our logistic regression model is 0.51, which means our model has no discriminative ability whatsoever—it is effectively guessing whether a song is minor key or major key by chance, meaning using valence is not a strong predictor for this. However, a feature with a slightly higher AUC that can be used to predict major or minor key better is speechiness, with an AUC of 0.57, slightly greater than 0.51.



10) To evaluate which is a better predictor of whether a song is classical music – duration or the principal components you extracted in question 8, we must first convert the genre label to a binary numerical label that states whether a song is classical or not. For this, I created a new column named 'is_classical' that gives each of the genres a binary number that tells us whether the song is either classical or not classical. I used the same train test split as before, but now using duration, using my N# for random_state. Again, I used logistic regression since we have a binary dependent variable in whether a song is classical or not. To test how good of a predictor duration is, I used AUC and plotted it once again:



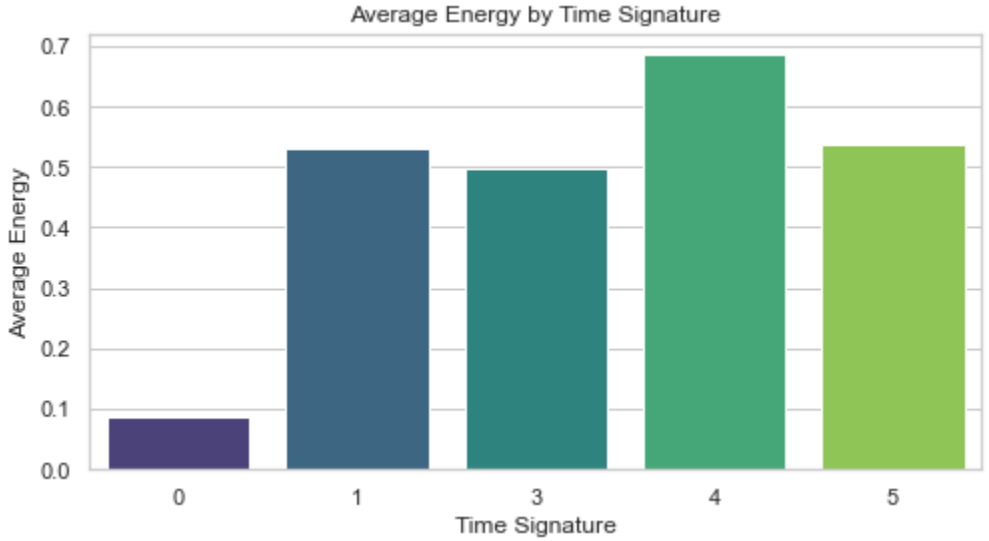
I then did the same train test split process with the principal components we extracted from question 8, the 7 principal components that were seen to have made up the 90% threshold for cumulative explained variance. This is the ROC curve for the principal components:



We can see that the AUC for the principal components is much higher than that of just the duration, with 0.96 > 0.60. Thus, we can conclude that using the extracted principal components is a better predictor of whether a song is classical or not than using just the duration.

Extra Credit:

I am interested to see whether the number of beats per measure (time signature) has any correlation with the energy associated with the song. I first created a bar plot with time signature on the X-axis and average energy on the Y-axis.



From this we see that songs with 4 beats per measure have the highest average energy out of all time signatures. However, when computing the Pearson correlation coefficient, we find that $r = 0.144$, meaning that there is a small, if any, positive correlation between time signature and average energy of a song. Thus, the time signature of a song is not a strong predictor of the energy associated with a song.