Capstone Project
PODS
Srikar Iyer

I would like to thank Spotify for the dataset and chance to help analyze anything or interesting, and I have found much insight in the patterns. I have completed reports of the ten particular questions that you have asked of me, after preprocessing the data. All code was done in python. Also, I thank Prof. Pascal, the TAs, and everyone else for helping me understand a lot of new things better. The document is long due to images, and further explanation / analysis than required for some questions.

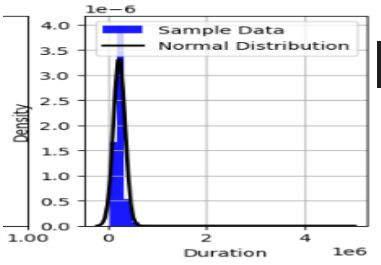
I was given a text file in the CSV (comma separated values) format, which acts functionally as a table. By this, I mean the first row consists of titles corresponding to properties of all songs (name, loudness, etc), and the other rows each have information of each individual song, organized by the aforementioned properties. For the most basic preprocessing, I just imported the python module pandas and read the file as a DataFrame, which acts like a dictionary / table (with key/value pairs), and randomized the seed for uniqueness with my N number. I didn't remove outliers since each datapoint was uniquely representative and influential on the total distribution. Even if one value looks out of place, it's still a real song with important impact. But for this entire project, there was never 1 value; there were at least 5 values in 'outlier' clumps, so removing the clump would not fairly assess songs like it. Used extensive use of numpy, scipy, matplotlib, pandas, scikit-learn, and seaborn for specific analyses and plots. And for the complex intricacies and debugging, ChatGPT was used. Importantly, I didn't see a requirement for confidence intervals. This is because we use a high sample size (52k samples per feature), so each statistical test and regression would have a really narrow or imprecise confidence interval, which wouldn't have added insight that would be there without the confidence interval. And even if the confidence intervals aren't super narrow, their importance is still dwarfed by analyzing the result of the underlying test / regression more thoroughly.

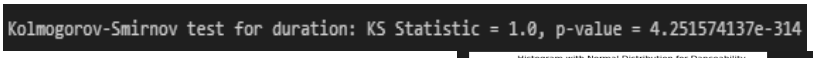
Task #1 : Consider the 10 song features: duration, danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence and tempo. Are any of these features reasonably distributed normally? If so, which one?

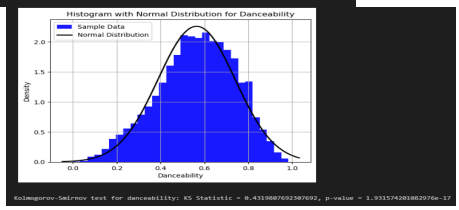
First, we plot the distributions of all songs for each of the requested features. Doing so gives us a plot for each feature, some of which looked interesting; for example, duration and tempo had more values near the center, while instrumentalness had a bimodal distribution, perhaps since some songs used a lot of instruments to great effect like jazz, while others might have used none like acapella. Thankfully, I was able to find the mean and standard deviation of the data for each feature and superimpose the resulting normal distribution onto each plot. Then, we can see if the distribution's mean is relatively central like the normal distribution and see if the tails are symmetric and decrease at the correct rate (exp(-x^2)). We also want to utilize a statistical test to compare this intuition to. In this case, since we want to compare the similarity of shape of two distributions (normal vs actual) feature, we use the Kolmogorov Smirnoff (KS) test, which measures the largest deviation at any point, and gives a p value that states the probability that

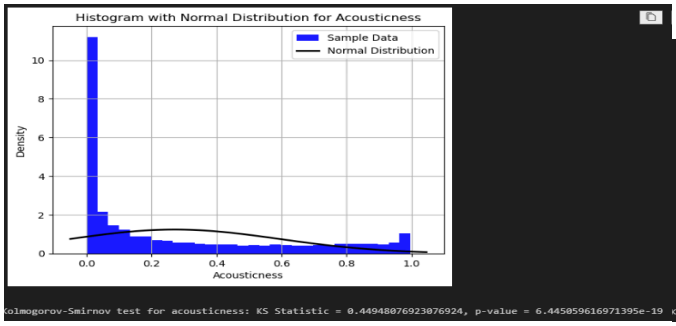
the two plots could potentially be the same. To note: the KS test is extremely sensitive to noise (and discrete data in general for this instance), so simply the ones with higher p values should be taken into account. However, I did not remove any outliers or change the data itself since the normal distribution must be compared at all values.

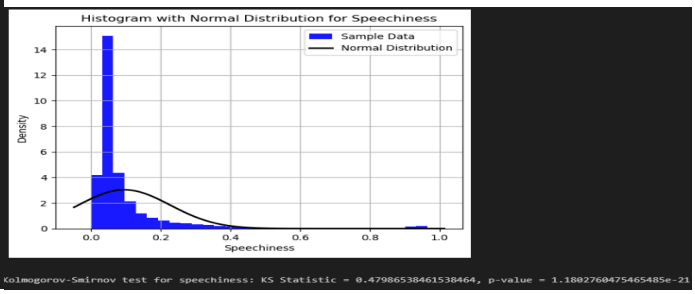
Please see the resulting plots (and KS test results) :



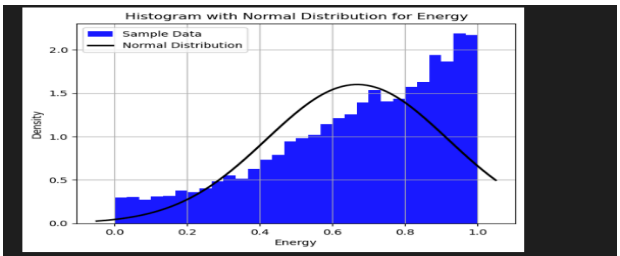


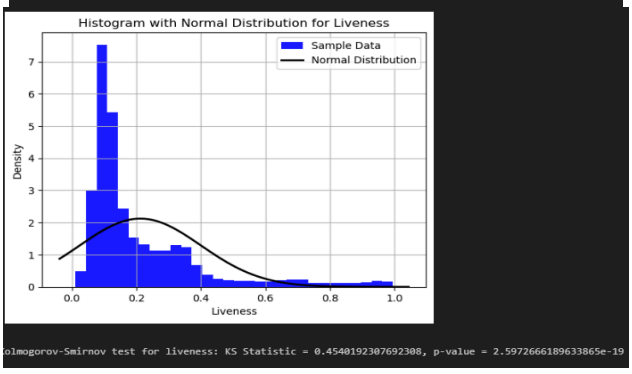


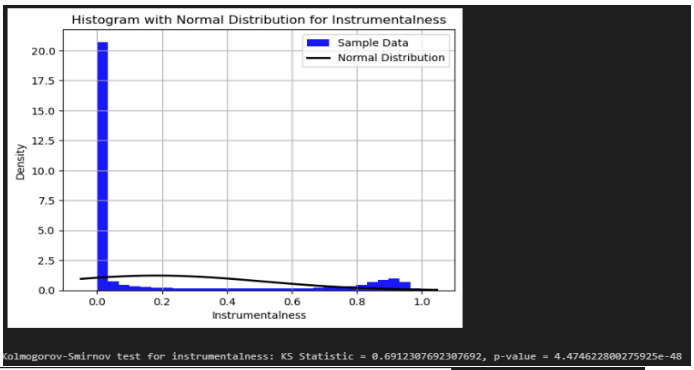


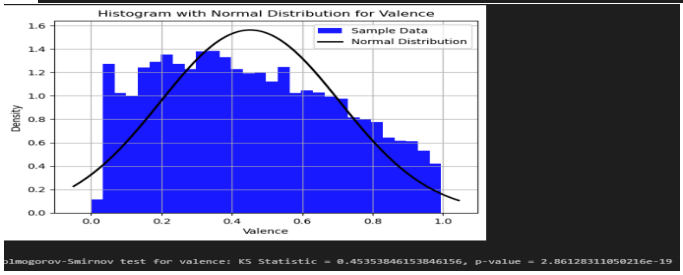


The distributions that end up with the highest p value are energy (around 5*10^-18), danceability (10^-17), then acousticness, liveness, and valence (O(10^-19)). Out of these, only danceability seems to follow its respective normal, with just a bit of right skew; this makes sense since the p value is twice the next highest. So out of these, danceability is the closest to represent the normal distribution, while the others fail visually. Most have no central tendency to the mean, and the ones that do (aside from danceability) follow a more uniform, ultra-skewed, or even erratic distribution like in the case of Tempo. This is because the KS test is unable to discern shapes or relations that the distribution shows, only able to find the maximum of how much both distributions differ by.



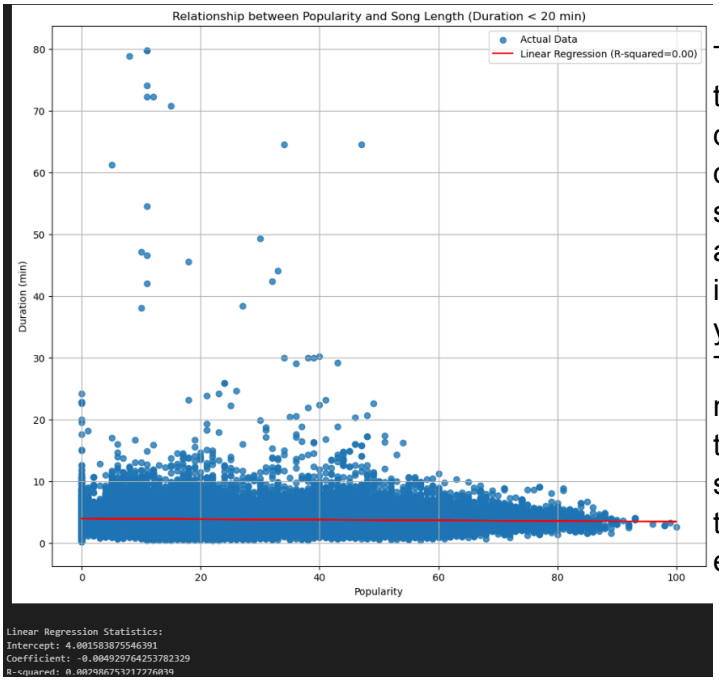






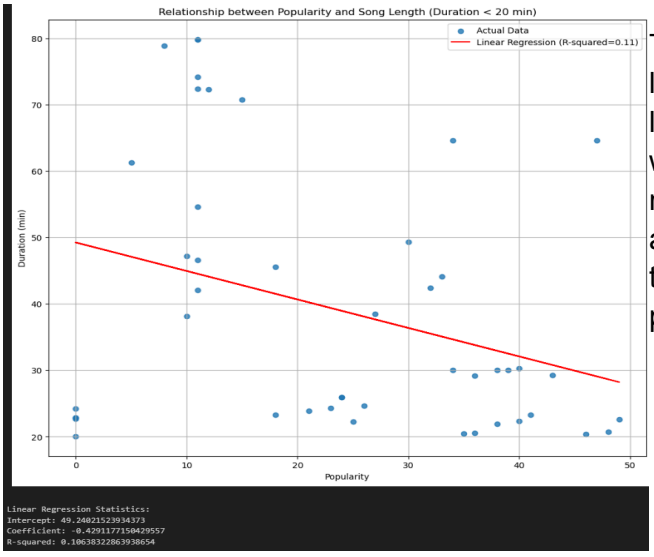
Task #2 : Is there a relationship between song length and popularity of a song? If so, is the relationship positive or negative?

Both attributes of each song, represented as numerical data, can be plotted to see what is roughly happening, and can be linearly fit, to see if there is a positive, negative, or no affine relationship (i.e. modeled by a line y=mx+b). Doing this, we have the scatterplot:



There is no clear relationship looking at the data since as popularity increases, the dense mass of song duration values doesn't really increase or decrease; it just seems to make a cone that narrows near around 4 minutes (which doesn't seem inaccurate). The line that we get is y=4.000-0.0049x with a R^2 of 0.0029. This tells us that, really, there is pretty much no correlation (not negative enough to matter), and the correlation itself isn't super representative of its data. So far, there is no linear relationship; but let us explore that cone idea.

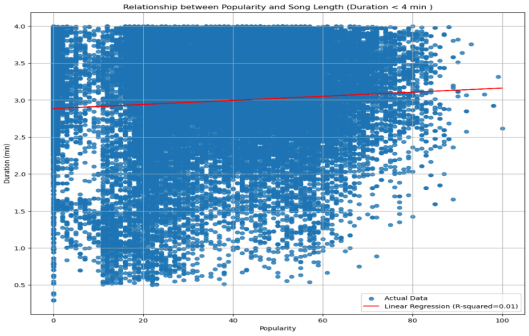
So, for some, the duration values > 20 might seem like outliers, but it really seems to imply that those long songs are really more unpopular than the general trend since none of them cross popularity=50. If we look at only values between 20 and 80, we get:

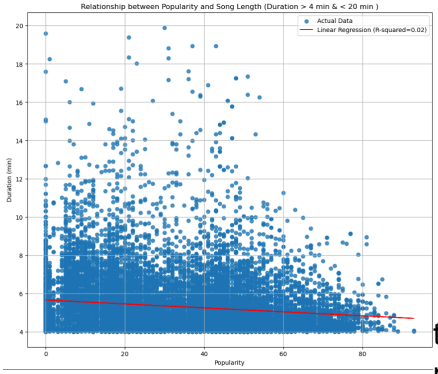


This is interesting since the line decreases for long songs; i.e. as the songs go from super long to pretty long, the popularity increases, with an R^2 of 0.11 which is comparatively more significant (and expected since there are a lot of natural outliers; killing them would kill the majority of natural distribution / valid data pts, so i'm not doing it).

We can then explore the top and bottom of the cone (songs from 0 to 4 mins and songs from 4 to 10000 (or max value) mins) separately to see if my guess is right. My hypothesis is that above 4 mins, we get a negative relationship (like with the super long songs) and below it, we get a positive relationship.

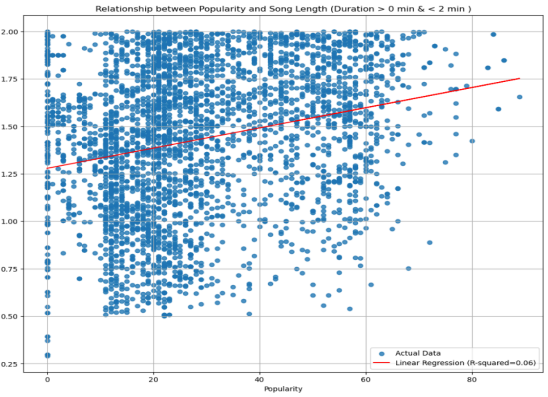
Both plots return :

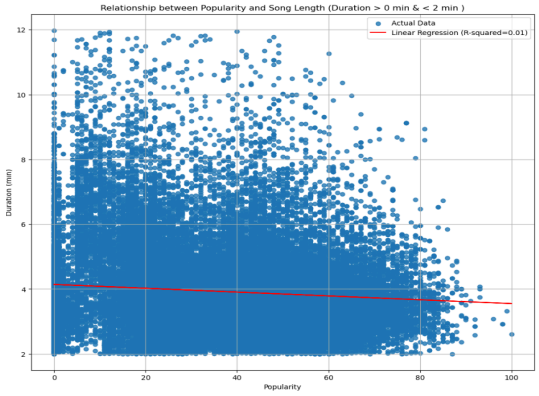


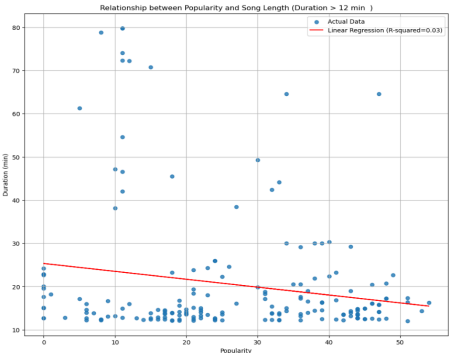


Ok, so the cone idea is still promising, just needs revision; I expect more of a bullet distribution. This is because the R^2 aren't good enough, and that there is a clear visual change of relation at the extremes of duration at both sides.

For example, for values between 2.0 and 12.0, we'll get no clear linear relation (constant), above 12.0 returns a negative relationship, and below 2.0 returns a positive relationship. Let's test that!

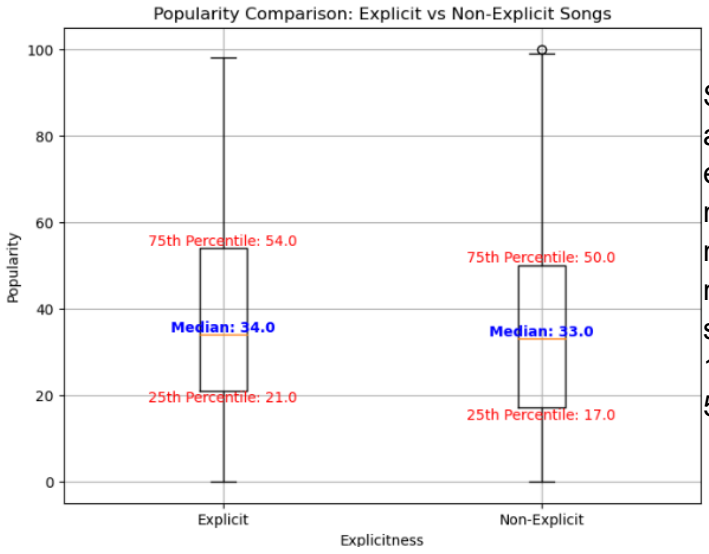






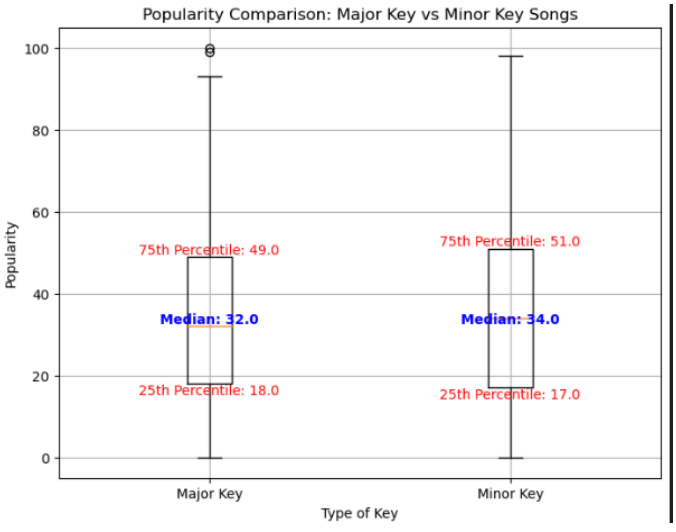
So, the data doesn't distribute like a bullet exactly (since the R^2s <0.1), but the piecewise relationship remains true: from 0 to 1.5 or 2 mins, popularity increases with time, from 2 mins to 11 or 12 mins, popularity remains relatively constant with time, and above 12 minutes (especially above 18 or 20 mins), popularity decreases with time. Therefore, the closer we get to the 3-10 min range, the more popular the songs get on average, and the farther we go, the less popular the songs get. But, there is no clear positive or negative linear relationship over all song durations.

Task #3 : Are explicitly rated songs more popular than songs that are not explicit?
First, we sort songs based on whether or not they are explicit into two groups. Thankfully, explicitly is boolean so it is simple. Then, if we compare the popularity meaningfully between both groups, we can see if one of these groups is more popular than the other (then, explicit songs are more popular than if not, or the converse) or if both groups aren't substantially different popularity wise. Popularity is ordinal since one song can be more popular than another 'objectively', as more people could listen to it. But it is not cardinal; a song of popularity 50 is probably more popular than 2 songs of popularity 25, or at least is not definitively equal. So, we need to use a nonparametric hypothesis test with these two groups. The data isn't categorical, and we are comparing numbers of ordinal flavor but not cardinal, so the medians matter more, as medians measure the central tendency of ordinality. So, we use the Mann Whitney U Test, letting the default null hypothesis be that the default distribution of explicit and non explicit groups are equally popular. The U test results in the U statistic of 139361273, and a p value of around 3*10^-17, which is less than 0.05. The p value is probably that small due to the high sample size, which results in high power. So, explicitly rated songs are definitely distributed differently than clean songs. We can actually plot the distribution ordinally with the Box and Whisker plots, which are shown here :



So, while the distributions of both plots are different which justifies the U test, explicit songs are clearly, by percentile margin comparison, more popular than non-explicit songs. Though, this is not by much, since the popularity of explicit songs only beats non-explicit songs by 4, 1, and 4 percentile points at the 75th, 50th, and 25th percentiles.

Task #4 : Are songs in major key more popular than songs in minor key?
We can sort songs based on major or minor key, which is boolean. Then, if we compare the popularity meaningfully between both keys, we can see if one of the keys is more popular than the other or if both keys are equally popular. Popularity is ordinal but not cardinal as discussed before. So, we need to use a nonparametric hypothesis test with these two groups.

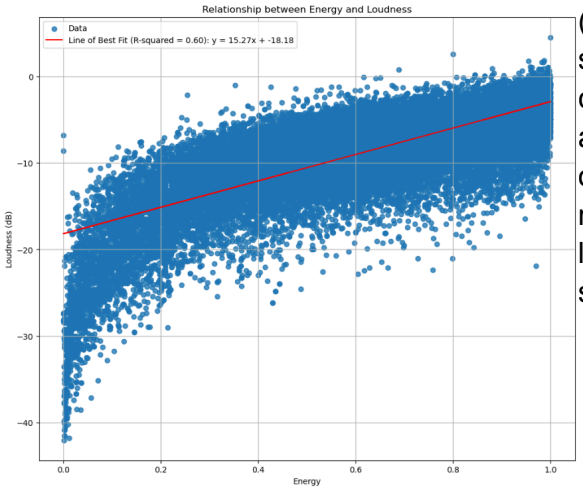


The data isn't categorical, and we are comparing numbers of ordinal flavor but not cardinal, so the medians matter more, as medians measure the central tendency of ordinality. So, we use the Mann Whitney U Test, letting the default null hypothesis be that the default distribution of songs in major key and those in minor key are equally popular. The U test results in the U statistic of 3*10^9, and a p value of around 2*10^-6, which is less than 0.05. So, songs in both keys are distributed differently popular wise. We can actually plot the distribution ordinally with the Box and Whisker plots, which are shown here :

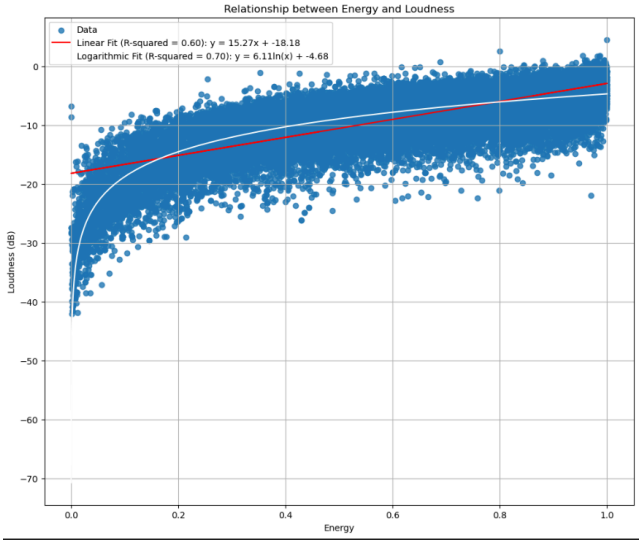
So, while the distributions of both plots are different which justifies the U test, it's not clear which is more popular just looking at quartiles. This is because the median of the population of songs of major key is 2 less than that of songs of minor key, and the quartiles have the opposite relationship. Though the minor key does have a more top heavy IQR, minor key songs seem to have the slight edge for more popular songs, while amongst the less popular songs, major key songs are more popular. But since we generally regard popular songs as the ones higher on the popularity scale, minor key songs win the contest marginally.

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

Since both values are numerical for songs, we can look at a scatterplot. Here it is:
(I added linear regression just for correlation's sake). The R^2 and coefficients are both high (0.60, and significant positive slope) so yes, we can say that energy and loudness have a positive correlation. But, we can show more. Since decibels are measured in the log base 10 of the power level of an electric (or audible) signal, a logarithmic regression should be more apt; also this graph looks logarithmic, increasing quickly then increasing slowly.

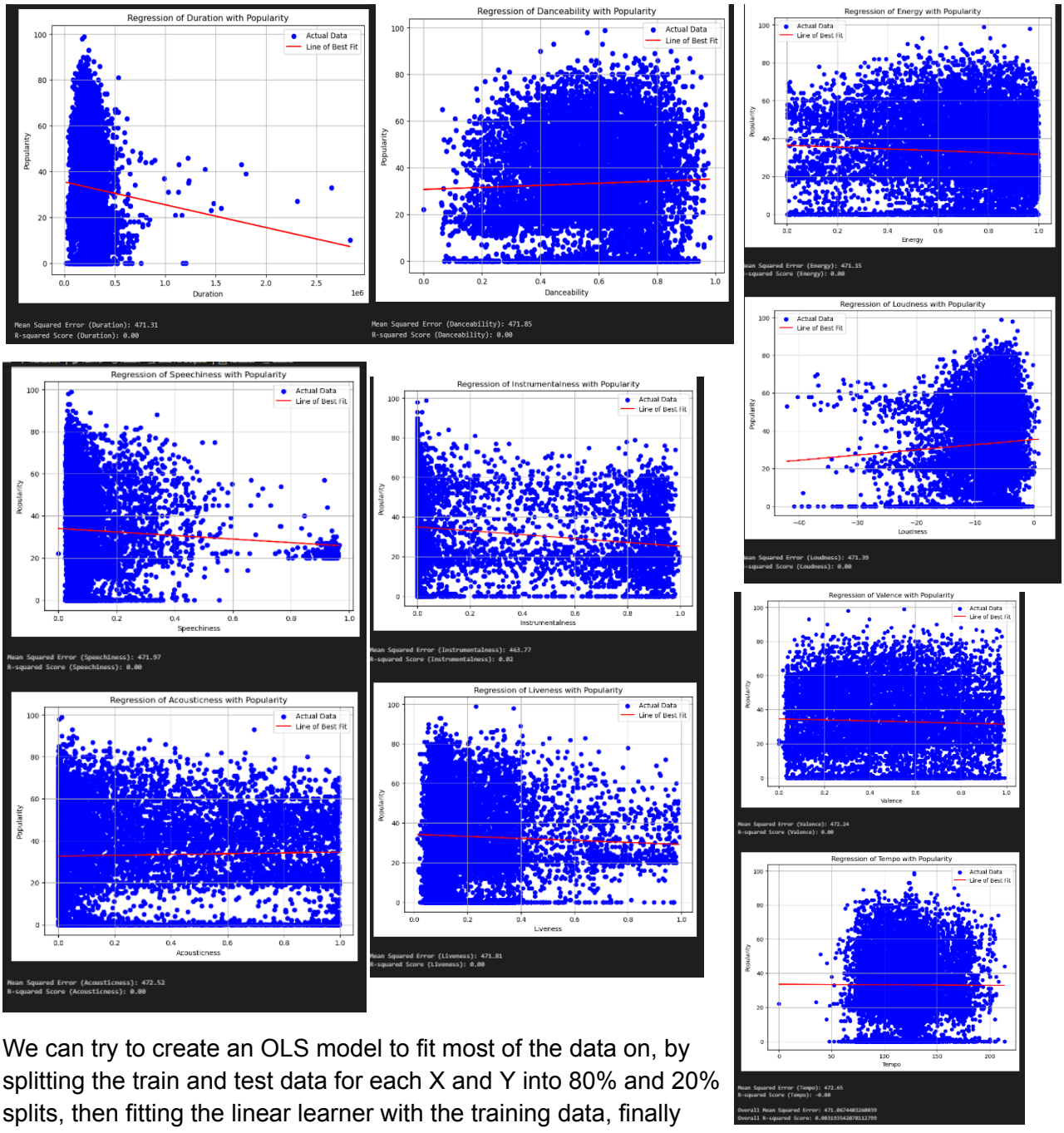


So, we try logarithmic regression and end up with the following graph :
Clearly, the adjusted R^2 is 0.7, and it captures the essence of the data better; as energy increases from 0 to 0.1, loudness sharply increases; after that, loudness increases slowly, with the middle seemingly following the logarithmic curve of best fit perfectly. So, energy



does reflect the ‘loudness’ of the song, in a logarithmic relationship.
Task # 6: Which of the 10 song features in question 1 predicts popularity best? How good is this model?

Since all features including popularity are numerical with cardinality, ordinality, and nominality, we can use a regression model for each feature alongside with a scatterplot to see what is actually happening.
Here are the plots :



We can try to create an OLS model to fit most of the data on, by splitting the train and test data for each X and Y into 80% and 20% splits, then fitting the linear learner with the training data, finally evaluating with the test data. (X represents each feature, Y represents popularity) In fact, the

Mean Squared Error, R^2, and Mean Actual Error were calculated with the Y predicted from testing the trained model with the test Y's and actual Y's. Upon plotting each of the 10 features, we see that none of the features correlate well on first glance or result in a good R^2 value at all relative to popularity. So out of all these regression plots, only one has a R^2 value > 0.01; that one is Instrumentalness vs Popularity with a still abysmal R^2 score of 0.02. We cannot do anything about that though, since all of the data is representative of the overall trends (or lack thereof), without any significant coincidences. For instance, it's no surprise that energy for instance has no effect, as on popularity, as there are a lot of peppy and slow songs that get millions of plays where people love them, but also peppy and slow songs that people don't like / never get popular. In fact, only a few plots at all have any non-constant (or not super low) relationship with popularity, those being Instrumentalness, Speechiness, Loudness, and Duration with negative, negative, positive, and negative relationships respectively. The model's MSE also represents this well : Every model except Instrumentalness features a MSE from 471 to 473 while Instrumentallness has a MSE of 463. I am only using MSE and not MAE or anything similar since the R^2 error is the best approximator of a linear model. So, instrumentalness is the best predictor of Popularity amongst the other numerical variables, but still not very good.

Task #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?

So, I simply tried multivariate least squares linear regression, lasso regression, and ridge regression models. OLS, like in the previous question, can simply find the line in multidimensional space (in this case, 11D space since there are 10 input features vs 1 output feature) that optimally tracks popularity. The idea is that since the individual features alone can't predict popularity, some combination of them might work better. Lasso and ridge simply account for multicollinearity; namely, if some features are more correlated, we can reduce their impact by subtracting their impact with L2 regression in the case of ridge regression, and L1 (which could kill useless dimensions) in the case of lasso regression. The results are here :

```
Linear Regression - Mean Squared Error (All Features): 453.05078693651126
Linear Regression - R-squared Score (All Features): 0.0413178421493674

Ridge Regression - Mean Squared Error: 453.050448781303
Ridge Regression - R-squared Score: 0.04131855770571158

Lasso Regression - Mean Squared Error: 453.9884925802496
Lasso Regression - R-squared Score: 0.03933360175979328
```

So, we can simply select Ridge regression for being better than OLS and Lasso with minimizing MSE and maximizing R^2. Thankfully, this result is better in every way compared to the best model comparing each feature with Popularity, so the combined Ridge model (and the others too) are better than the Instrumentalness vs Popularity model and all of the other worse ones since they have a far lower MSE (453 vs 462) and better R^2 (0.04 vs 0.02). I account for this since, for instance, instrumental songs with smaller duration might be more popular than less instrumental songs or long songs, which is accounted for in the multivariate model, but cannot be accounted for in OLS models since the latter can accommodate for 2 features / dims at most. Combining this

logic for all combinations of each feature with popularity, it's no surprise that this model has more information and context, and can approximate popularity better.

Task #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?

Given we have all 10 feature vectors, we can create a correlation matrix that is symmetric by nature. So, when we eigendecompose this matrix, we get the eigenvectors (principal components) and corresponding eigenvalues, which tell us how influential each component is. By the Spectral theorem of symmetric, full rank matrices, we know that each principal component is orthonormal, so each is independent of each other; combined, they constitute the full data rotated on the eigenbasis. These eigenvectors and eigenvalues are :

```
Eigenvalues:
[1.38491215e+10 8.54782569e+02 2.32737930e+01 1.04695822e-01
 6.91384911e-02 5.68605002e-02 4.07546017e-02 9.00790086e-03
 1.58354759e-02 2.08348416e-02]

Eigenvectors:
[[-1.00000000e+00  9.84979346e-06 -3.13614009e-07  4.46506145e-07
  -5.10088733e-09  1.67995457e-07  1.45629142e-08  3.04841658e-08
  -3.37195203e-08 -3.77451295e-08]
 [ 7.56483635e-08  5.36665479e-04  6.79507503e-03  1.72984223e-01
  -3.36158381e-01  2.26445393e-01 -1.25794631e-01 -3.14892764e-01
   5.22433337e-02  8.29481113e-01]
 [-1.46418442e-07 -2.08336528e-03  3.83116899e-02 -2.06964577e-01
  -2.54075894e-01 -1.92626036e-01  3.05096550e-01 -6.55124099e-01
  -5.48628875e-01 -1.75411104e-01]
 [-6.50969615e-07 -3.48497920e-02  9.97414104e-01 -4.11033870e-03
   3.95795454e-02  4.30345026e-02 -6.47575479e-03  1.69307998e-02
   1.35673712e-02  1.59162188e-03]
 [ 5.14086142e-08  1.98683556e-04  1.08057947e-03  5.38214335e-02
   3.05497617e-02 -3.87637069e-02  3.38630635e-01  5.94006702e-01
  -6.47060403e-01  3.29339185e-01]
 [ 3.13836268e-07  2.33167485e-03 -3.89859947e-02  3.93249542e-01
   5.72762246e-01  6.10310608e-01  1.63190176e-01 -2.74108920e-01
  -1.85755940e-01 -8.37963124e-02]
 [-2.42829647e-07  3.56564345e-04 -2.83158054e-02 -7.94103681e-01
...
   5.02790808e-04  5.23273200e-04 -2.93303243e-05  6.22339427e-05
   1.52052005e-04  7.52621392e-04]]
```

Then, we can choose the principal components with the elbow method and the kaiser method; clearly since the first eigenvector is 10^8 times more important than the second eigenvector, by the elbow method, it alone is the sole principle component, and surely accounts for 99.5% + of variance and information. But, I want to be super conservative (almost unnecessarily) and use the Kaiser criterion, choosing only principal components with eigenvalues > 1. Then, we have

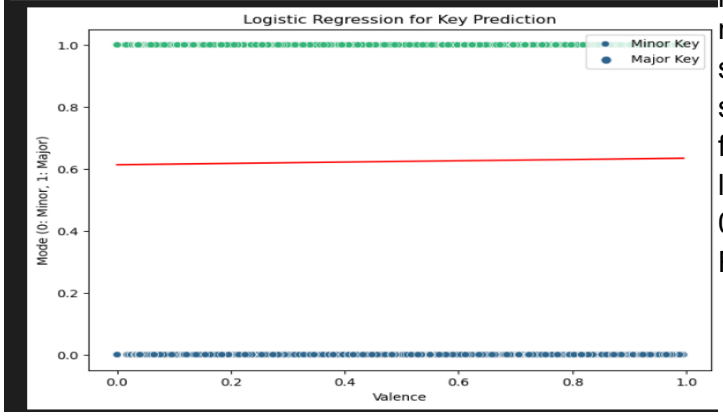
```
Eigenvalues:
[1.38491215e+10 8.54782569e+02 2.32737930e+01]

Eigenvectors:
[[-1.00000000e+00  9.84979346e-06 -3.13614009e-07]
 [ 7.56483635e-08  5.36665479e-04  6.79507503e-03]
 [-1.46418442e-07 -2.08336528e-03  3.83116899e-02]
 [-6.50969615e-07 -3.48497920e-02  9.97414104e-01]
 [ 5.14086142e-08  1.98683556e-04  1.08057947e-03]
 [ 3.13836268e-07  2.33167485e-03 -3.89859947e-02]
 [-2.42829647e-07  3.56564345e-04 -2.83158054e-02]
 [-2.82645732e-08  1.73773668e-05  3.67947164e-03]
 [ 2.53039094e-07 -2.95790414e-04  9.61712553e-03]
 [-9.83220482e-06 -9.99387399e-01 -3.49608267e-02]]

Total Variance Explained by Important Principal Components: 0.9999999999771013
```

Our total variance explained by principal components is around 0.9999999999, or less than 10 billionths off the total variance explained by the entirety of the original data (1) while only consisting of 3/10 the amount of information / features. In fact, these 3 features (even 1 if you want to save a lot of information) represent the data as a whole as the rest are functionally meaningless. We calculate the total variance explained by summing the eigenvalues of the PCA divided by the trace of the eigenvalue matrix, or eigensum of the covariance matrix. This dominating result is pretty clear since the first principal component is, again, 100 million times as influential relative to the rest.

Task #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?

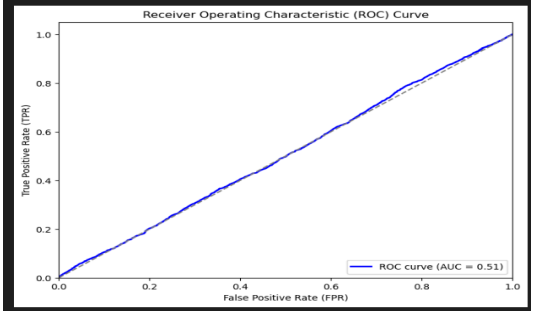
We want to make a model that predicts categorical (binary) data (minor/major key) from numerical data with ordinality, nominality, cardinality (valence). To do this, we can approximate the relationship with logistic regression, then select points predicted above and equal to 0.5 as 1 (minor key), and below 0.5 to predict major key. When we try this for valence, we get:

```
Accuracy: 0.6241346153846153
Confusion Matrix:
 [[   0 3909]
 [   0 6491]]
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      3909
           1       0.62      1.00      0.77      6491

    accuracy                           0.62     10400
   macro avg       0.31      0.50      0.38     10400
weighted avg       0.39      0.62      0.48     10400
```



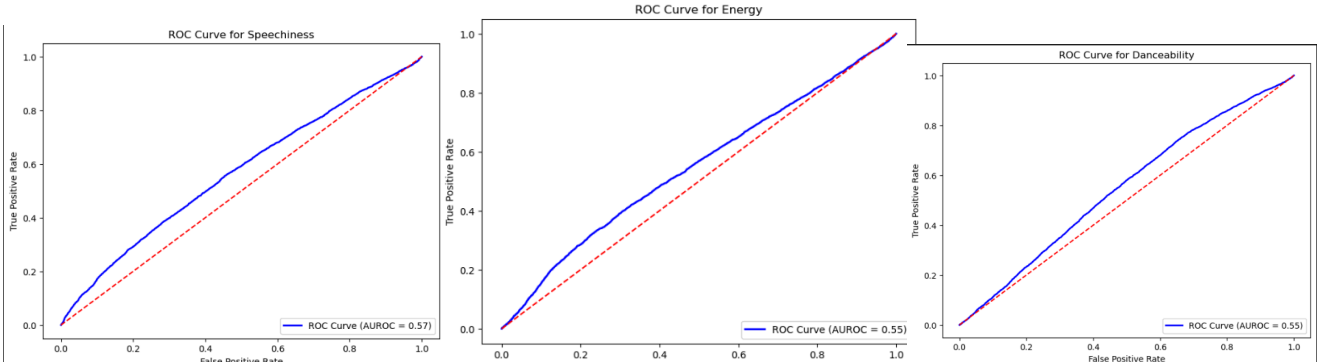
So, the predictor seems to have a lot of difficulty getting anywhere with classifying, so the valence doesn't predict songs with minor key (true positive) or songs with major key (true negative). There is no FP/FN recorded since the entire regression is over 0.5, so all values are predicted to be true.. In fact, it's stretched long enough to look like a terrible linear regression between 0 and 1, but we can make sure with the ROC curve.

```
Confusion Matrix:
[[   0 4961]
 [   0 8039]]
True Negatives (TN): 0
False Positives (FP): 4961
False Negatives (FN): 0
True Positives (TP): 8039
AUC: 0.505604894943841
```

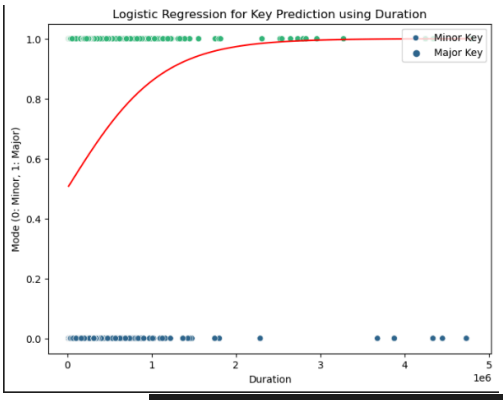


Clearly, the ROC curve (and AUROC) agree with the sentiment that valence doesn't predict key at all, since the ROC is very close to chance (50% random guessing correct or wrong), which the AUROC (51%) substantiates. Regardless, this is a result of the logistic model predicting everything as a minor key, so because there are spread out, near uniform distributions between minor and major key songs choosing with this criteria is similar to chance. So, you can try to predict key from valence, but it wouldn't work at all.

Let us look at the best possible example from the 10 numerical features; well, to start, they all share the same failures of false positive/negative as a result of logistic regression only being weighted > 0.5. But, some features predict popularity well, just assuming that everything is minor key (then, there is an implied relationship between the feature and minor key). Let us see some examples : (AUROC = 0.57, 0.55, 0.55); all of these have a very similar logistic regression graph not worth showing.

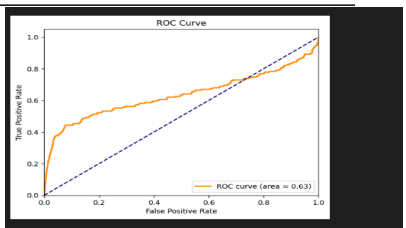


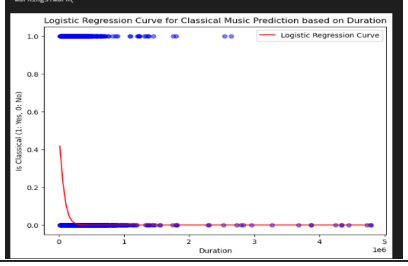
So, speechiness is a slightly better predictor of key, though none of them are really good. To note : the most interesting logistic regression was that of Duration, though its AUROC was around 0.5. I didn't change or omit any data points as each one is important to influence the ROC independently, so the data speaks for itself (there are really no outliers in this case, as fuzzy results naturally come from fuzzy data).



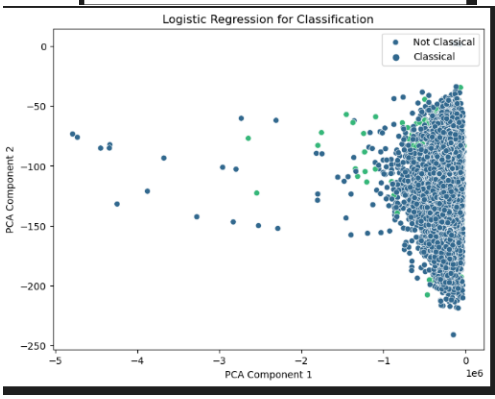
Task #10 : Which is a better predictor of whether a song is classical music – duration or the principal components you extracted in question 8?

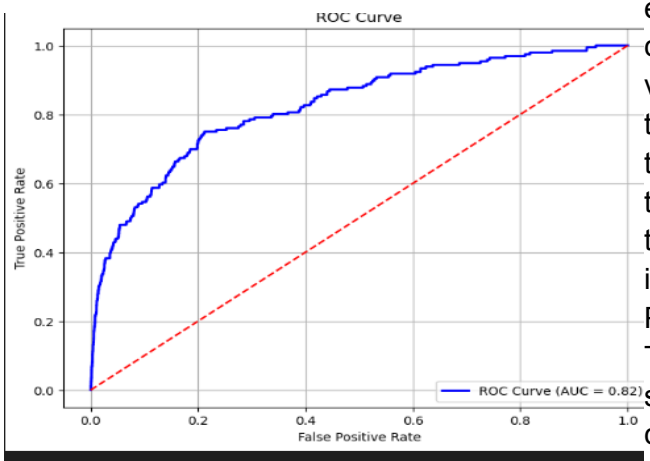
We can split the question into 2 parts : creating a model to predict classical music (boolean) from duration (numerical). We create a logistic regression model, and plot its unique ROC curve to get the graphs on the right ->
We get a decent AUROC of 0.63, but with a convenient logistic regression curve; it predicts all songs as not classical, and just because there happen to be more non-classical songs at super high duration, it fits better at that duration, while at low duration, it's just guessing not classical from a huge pool of all songs.



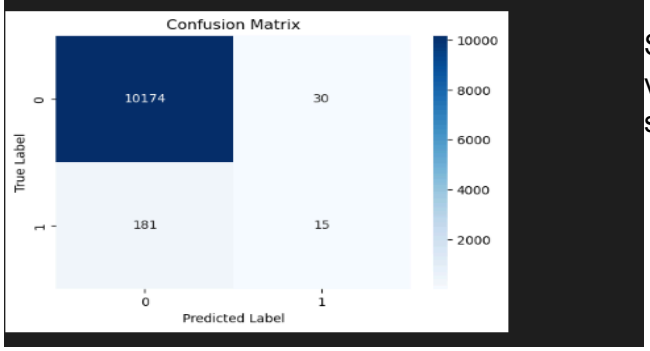


For the second part, we can map the principal components to the 10 numerical features by simply multiplying them; doing this gives us the projection of the principal components onto the feature matrix to get us our X data. The y data is simply what we're trying to predict, which is the same classical music boolean variable. The model we seek to achieve is a multivariable logistic regression between the 3 principal components extracted and the classical music variable. To the right is the mapping from PC1 to PC2 : It lets us know that the data is split reasonably between the two components; there is no autocorrelation or inherent pattern. We can assume the same for PC3, which matters less anyway.





Though we can't plot the logistic regression, since it is 4D, we can get the ROC curve and confusion matrix.

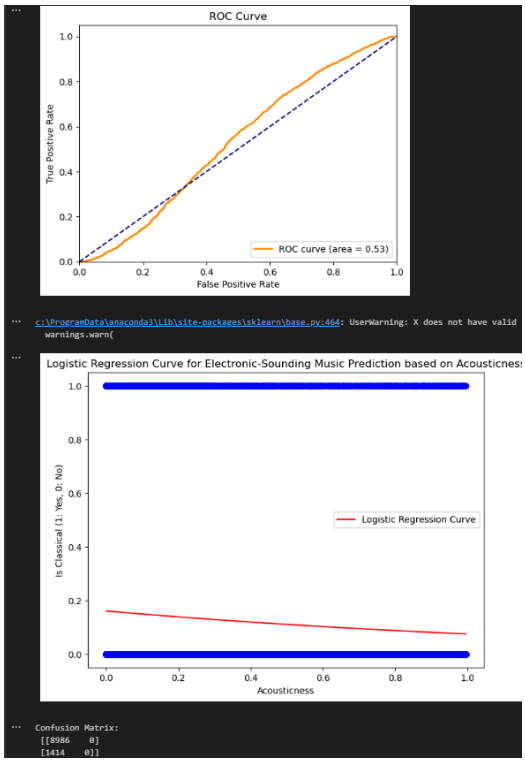


So, since the AUROC is 0.82, this model is a very good classifier; surely the best we have seen so far. This logistic regression model is

also more representative of the data since it clearly goes above and below 0.5 owing to the lack of 0 values in the TP, FP, TN, and FN boxes of the confusion matrix. The confusion matrix is also very promising, as true positives (non-classical songs) are identified reliably and effectively. One problem is that true negatives are far fewer than false positives, so it predicts classical songs very badly. Regardless, this is largely a consequence of mismatched sample sizes, which might be a consequence of removing some of the principal components, or just a consequence of how the logistic model cut off classical vs not classical data. Regardless, this model does a great job identifying non-classical data and is rewarded with a very solid ROC curve, so this model (derived from PCA) is by far better for predicting whether a song is classical or not.

Extra Credit : Acoustic instruments are usually thought to be utilized by some folk, country, and a few pop/dance songs to add some soulful flavor, or make it interesting. But, I don't think of them being in mostly electronic songs, including genres like trap and hip-hop. I want to test this theory.

We can try to use a logistic regression utilizing a boolean variable for the y variable, where the value would be 1 if the song has the genres I thought lacked acousticness and 0 else, and the acousticness column for the x variable. I originally included breakbeat, deep house, and disco, but they all use guitar and other instruments fundamentally so I omitted them. The genres left were : electronic, anime, chill, club, detroit-techno, dubstep, and edm. Upon trying the logistic regression, I got :



So, clearly my hypothesis is wrong since my guess of acousticness for the given genres has around a 50% chance of being right, which isn't too shocking. The logistic regression did me no favors by being below 0.5 from 0 to 1, so I can kind of expect 50-50. Perhaps this effect occurs because there are always some club songs that have 1 guitar riff that might add acousticness, or a hiphop beat might have a bass guitar-based hook. But even still, either I haven't listened to enough songs to experience the acoustic nature of some of these genres (possible), or I can't tell when jazz is being played vs synchronizer, and oftentimes, I can't pick out when they're being mixed. So, perhaps I chose my genres wrong, or there is really no relationship between music genre and acousticness, at least by how Spotify measures acousticness.