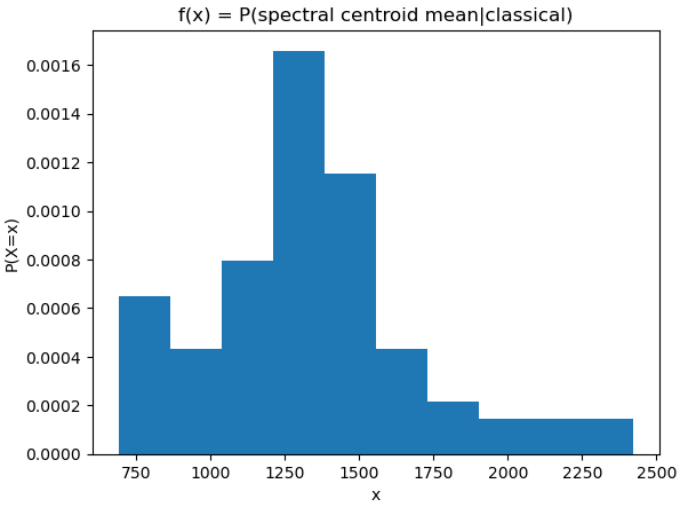
MACHINE LEARNING ASSIGNMENT 1

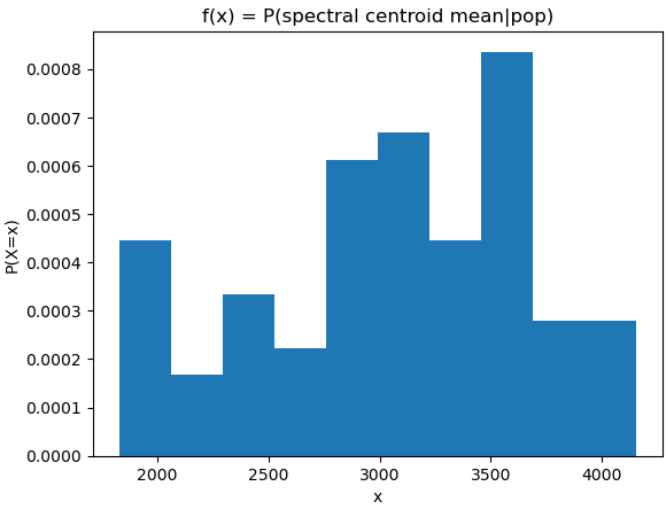
Can Senyurt 1079752

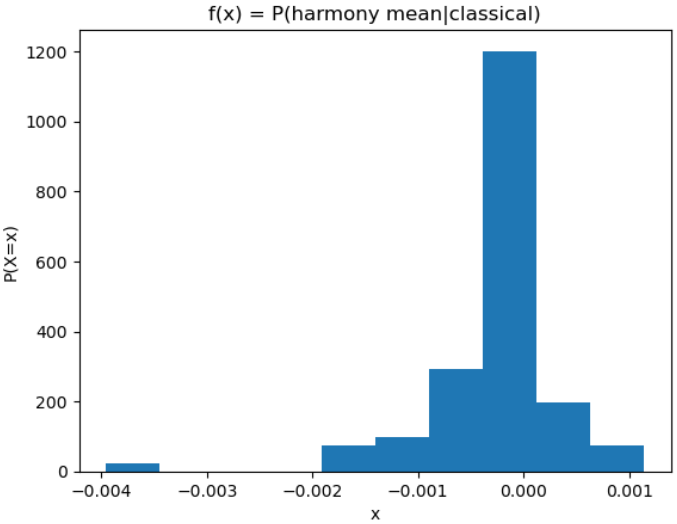
TASK 1

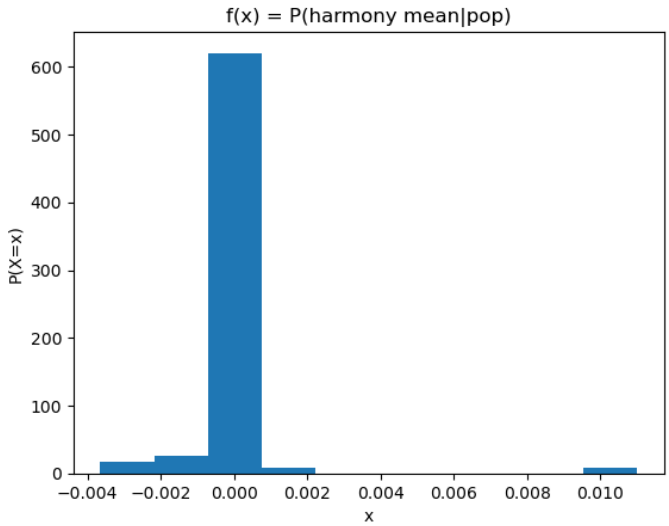
Q1: Treating classical as the positive class, my model returned a 97% accuracy, 95% precision and 100% recall which shows that the model is working really well when it is just comparing classical and pop songs to each other.

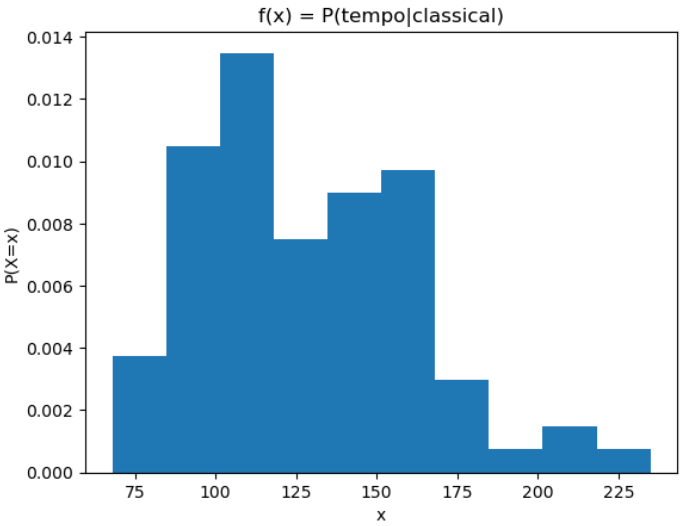
Q2:

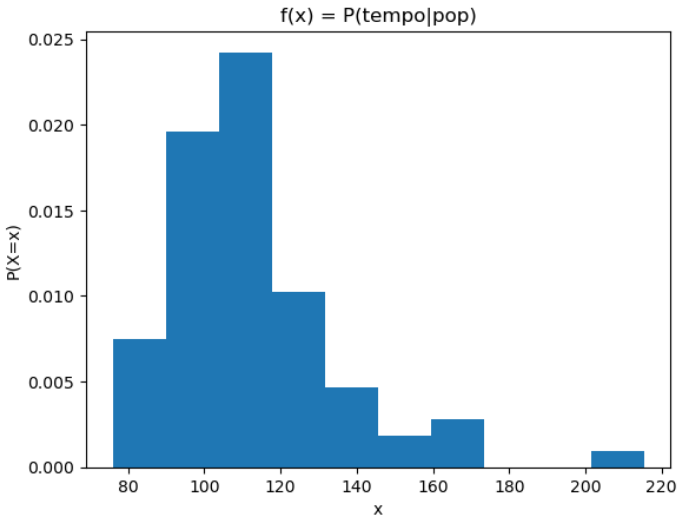












When looking at the graphs, the first thing that got my attention was that the harmony mean graphs' y axis were not [0,1) and after looking into it I realized the code's only requirement when making a pdf graph is the area underneath has to one, which means that I would have to alter the x axis values so I decided this was not a good idea, so I left it that way. When I was choosing which feature to chose out of these 3, I wanted to pick one that had means in substantially separate spots between the classical and pop graphs and as little overlap as possible. The graphs that fit this criterion the best is spectral centroid mean as the x values are in almost completely different ranges, which makes it a very reliable way of determining class. Whereas the other features had similar distributions to each other, meaning they do not change as much from genre to genre.

TASK 2

Q6: I modified the code to skip missing values if they ever came up and used Jaeden's code from https://stackoverflow.com/questions/68478052/how-to-randomly-delete-10-of-attributes-values-from-a-pandas-dataframe to randomly delete 0.01, 0.05 , 0.1 , 0.2, 0.5, 0.85 and all of the features from the test set, and then used my evaluation function to look at how many answers it got right. Here are my results;

1st run, no values deleted:

The code ran with 0.495 accuracy.

2nd run, 0.01 of values deleted:

The code still ran with 0.495 accuracy on average, which was expected as I only deleted a very small number of entries.

3rd run, 0.05 pf values deleted:

Code is still running with around 0.49 accuracy, lowest accuracy 0.47 and highest was 0.5 so in some cases it ran better than no values deleted.

4th run, 0.1 of entries deleted:

Still working with similar accuracy, highest recorded this time was 0.51 and lowest was again 0.47 accuracy.

5th run, 0.2 of entries deleted:

To my surprise the code is still running around 0.49 accuracy, which shows that the algorithm is not susceptible to missing values.

6th run, 0.5 of entries deleted:

The algorithm is still working with roughly the same accuracy, however the variation is considerably higher, the lowest accuracy I got was 0.45 and highest was 0.51. Compared to the other runs this is a big change, but overall, it is not changing much.

7th run, 0.85 of entries deleted:

To my surprise I am still getting similar accuracies, which is very interesting.

8th run, ALL of the entries deleted:

This time I was sure the accuracy would be VERY low, but the lowest I got was 0.44. This is due to one of or a combination of three things I think:

a) The algorithm is not susceptible to missing data almost at all.
b) I made a logic error somewhere along my code, however I could not find anything.
c) The Naïve Bayes algorithm might not be the best option for this task as I think it would make sense that some of these features could be dependent on each other quit heavily, but this is just a guess.