Mandatory assignment 1 – Spotify dataset

By: Sebastian Sjøen-Tollaksvik

Code: Github

Introduction

This exercise is the first mandatory assignment in the machine learning course. Given a large dataset of spotify songs, I was supposed to make a classification system that could separate Classical and Pop songs only using two of the features, “liveness” and “loudness”. Since the original dataset was so large, it made it easier for the computer to handle a smaller dataset. This classification was made using supervised learning since the dataset gave the labels. Since the exercise said that the labels pop and classical should be relabeled 0 and 1. It made sense to use a logistic regression bound within the [0,1] range.

Process:

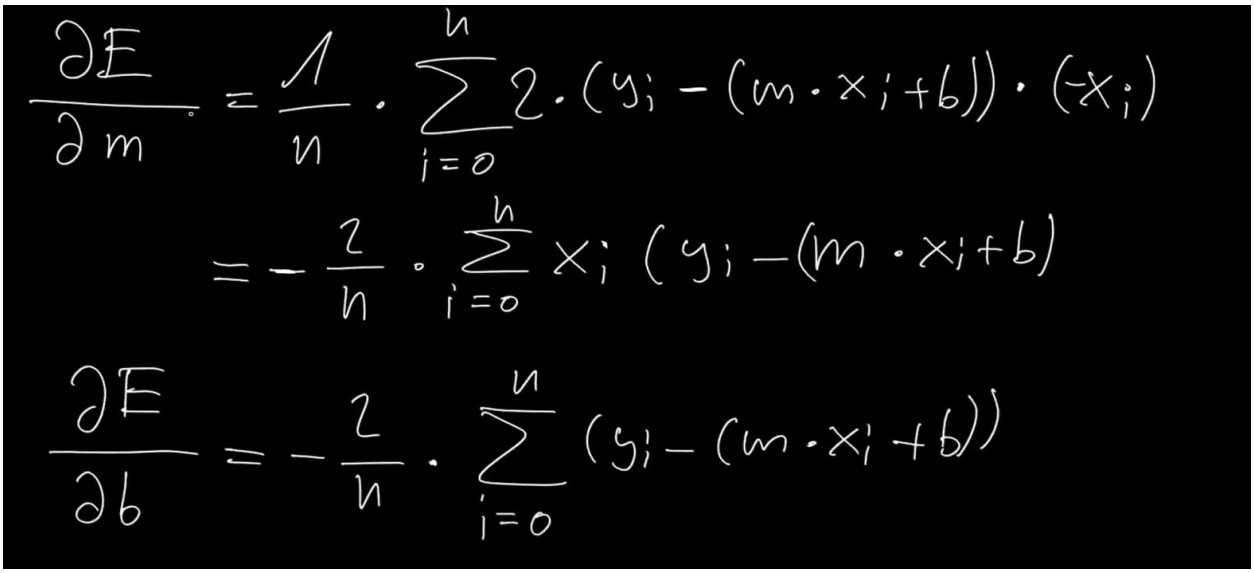
1) Data handling

The handling to the data was straightforward. Both numpy and panda are powerful imports that can handle complicated and taxing tasks with simple commands. With reading up on the librarys and a little help with chatgpt on reminding me on commands it didn’t take long before all the data was handled as the exercise explained.

2) Machine learning part

I decided to make a class for everything that was associated with the logistic regression, and in a separate file. This would make the code more readable and easier to code. The class has a learning rate and epoch, of course these variables can be adjusted afterwards to get a better result. I tried to make a class that does the same things as the sk.learn models do. Hence the variables names of the different functions within the class are the same. The functions are pretty similar to a linear regression, the only difference is that

there is a extra function called the sigmoid function that bounds the range from [0,1], since the labels is in this range. The gradient decent is just the derivative of the the error function of linear regression. The gradient decent for weight takes the derivative in respect to the weight, and vise versa for the bias. When we take the derivative of a function, we get the rate of increase at that exact point. So, each time the weight and biases move the opposite way of the growth. After enough epochs, the function should move to the lowest point of the loss function.



Screenshot taken from the Youtube channel NeauralNine

3) Evaluation and confusion matrix

   After the model is fitted to the data the process is nearly complete and it becomes simpler. Now you just got to make the model make new predictions using the given weight and biases. Afterwards you got to transform the values you get from the model into predictions. Since the model will spit out anything in the range [0,1] now, you got to set a threshold that splits the predictions. In this instance it makes sense to set it to 0,5 because we don't value on prediction higher than the other. In other instances where you value one prediction more than the other. It would make sense to adjust the threshold.

Afterwards I compare the predictions to the dataset. This is used to measure the accuracy of the model. I also make a confusion matrix using a few simple if statements and looping over the predictions and dataset.

Questions:

1) Load the SpotifyFeatures.csv file and report the number of samples (songs) as well as the number of features (song properties) in the dataset:

There are 232725 songs, and 18 features in the dataset.

2) Report how many samples belongs to the two classes:

Pop data: (9386, 3), Classical data: (9256, 3)

3)

Test your trained logistic regression classifier on the test set. Report the accuracy on the test set. Discuss any significant differences in performance between the training and test sets and what this might indicate (e.g., overfitting or underfitting).

Accuracy is around 0.92. I got this number after playing around with the learning rate and the number of epochs. I first tried using a high amount of epochs and a very low learning rate. This resulted in an accuracy of around 0.6. This indicates that those variables resulted in the model overfitting. The accuracy of 0.92 indicates that I found a good balance between learning rate and number of epochs. I'm quite impressed with the results of the model considering the limited access to features. The training set and test set is randomized, so that the balance between pop songs and classical songs is realistic.

4)

Create a confusion matrix for the classification results on the test set. Report the confusion matrix and explain what it shows in terms of correct and incorrect predictions (true positives, true negatives, false positives, and false negatives).

This is the array from the last time I ran the code:

	Predicted: 0 (Classical)	**Predicted: 1 (Pop)**
Actual: 0 (Classical)	True Negative (TN) = 1713	False Positive (FP) = 180
Actual: 1 (Pop)	False Negative (FN) = 86	True Positive (TP) = 1750

5)

You should now have two evaluation metrics for the performance of the classifier on the test set; accuracy and the confusion matrix. What information does the confusion matrix give you that the accuracy score does not?

The confusion matrix tells us how often the ML model predicts wrong/right and what it struggles to separate. For example; on our data, my model struggles to predict Classical songs since it predicted it wrong 180 times compared to 86 times wrongs on the pop songs. In this dataset we don’t value one of the predictions more than the other. But if we had model to predict a decease, we could switch this around to improve the chance of picking up a sick patience.

Reflection/conclusion

I would say that this exercise was a valuable lesson, I learned a lot about machine learning methods, and it was quite challenging in the start as I spend a couple of hours researching and understanding the theory behind the classification method. I also had a lot of problems using the numpy library. There were a few issues with data types. For the next assignment I will get more familiar with the numpy library. I am happy with the turnout of the code, and I got to answer all the questions.

Sources:

https://github.com/uitml/MLcourse/blob/main/code/loadDataTutorial.ipynb

NeuralNine. (2022, March 16). *Logistic Regression Explained - Machine Learning* [Video]. YouTube. https://www.youtube.com/watch?v=VmbA0pi2cRQ

NumPy. (n.d.). *NumPy*. https://numpy.org/

OpenAI. (2024). *ChatGPT* [Large Language Model]. https://chatgpt.com