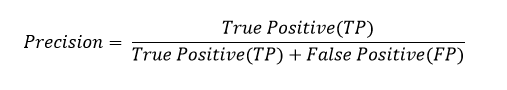
**Instructions**: Please complete and submit your work to the appropriate folder in LumiNUS. You may work in study groups, but each student must be responsible for their own submission.

Please submit all the following documents as a single zip file named StudentID-Name-H4.zip:

1. Completed Word file named as StudentID-Name-H4.docx (with all results)
2. Print preview of ipynb file named as StudentID-Name-H4.pdf (with all results)
3. Working ipynb file named as StudentID-Name-H4.ipynb
4. Consider the metrics accuracy, precision, and recall.
   1. Give one example when accuracy would not be a good performance metric. Give a numerical example.
   2. Given one example of a supervised machine learning classification problem when higher precision is desired. Please give a different example than the ones given in class. This need not be a numerical example but must be clearly defined classification problem and dataset.
   3. Given one example of a supervised machine learning classification problem when higher recall is desired. Please give a different example than the ones given in class. This need not be a numerical example but must be clearly defined classification problem and dataset.

**Ans 1:**

1. Suppose we consider a two-class problem for COVID-19 detection for example, where there is a machine learning classifier model which performs testing on 1000 people to check on whether they are COVID-19 positive or negative. If there are 990 people who are detected COVID-19 positive, which is depicted by class 1, and 10 people who are detected COVID-19 negative, which is depicted by class 0, the accuracy of the model will be 990/1000 which is 99% which identifies COVID-19 positive patients with 99% accuracy. The model will not correctly predict any sample in class 0 and hence, in this case accuracy is misleading and does not give a good picture of model quality. If the model simply identifies most patients belonging to class 1 i.e., COVID-19 positive and still have 99% accuracy, which is incorrect and biased as they misclassify the class 0 cases.
2. Precision (positive predictive value) is the ratio between the True Positives and all the Positives identified by the machine learning model, and it describes how many detected items are truly relevant which is identified by the equation:



This is my own example which has come up on the top of my mind. For the case of criminal punishment systems in jail, precision would be more important than recall in this case as we really do not care about false negatives but focus more on false positives. In this scenario, if we miss out to punish a criminal is still alright with not so severe consequences in the long run and this serves the purpose of a jail as well ultimately, (low recall as this is emphasizing on a true positive), but incriminating an innocent person is highly undesirable (this is false positive in this case, as the person has wrongly been implicated as a criminal). For this machine learning model classification system, the objective is to find out the right criminal who is actually a criminal, out of all the people where the system has classified the suspects as criminals (all positive detected by the classifier), and we must ensure that the right person is punished, not the wrong person (false positive), hence, precision is more important here. The dataset can be from the jail’s history sheeter and criminal records and behavioral dataset from people’s profiles in the city where the jail is located, with current criminal cases in the city that the jail is located and neighborhood crime data, so that they can link up certain current incidents to known criminal facts.

1. Recall is the ratio between the True Positives and all the Actual Positives identified by the machine learning model (true positive rate) which is illustrated by the equation:

Text

Description automatically generated with medium confidence

This is another example which I have thought about carefully. Recall is important when false negatives are catastrophic, and you want to detect all positive cases. One particular example that I can bring out is the fraudulent banking transactions classification model which identifies whether a particular bank transaction is fraudulent or not. This is particularly important in today’s world where everything is digitized and almost everybody has a bank account to store their precious life savings and earnings, so a fraudulent transaction can spell doom for the customers money using the bank platform (especially high net worth individuals) and fraudulent transactions results in severe damages for the bank worth millions (they might need to refund a lot of money to customers). In this model, it is still alright if I classify a proper or an honest transaction as fraudulent (which is a false positive in this case) as you can always reverify transactions through additional checks from auditors such as KPMG & EY, and top management officials, so it would be better to be safe than sorry here. However, if a fraudulent bank transaction has been wrongly classified as honest or legitimate (which is a false negative in this case as the fraudulent bank transaction has been wrongly classified as not fraudulent as specified by negative), and this can cause severe damages for the bank and customers money in the long run which might result in a police or litigation case, which can result in people’s loss of life savings due to a fraudulent transaction, and this could even arise to become a national security issue as well. For this case, the dataset are all records of the bank transactions over different times of the day, month and year, along with the personal profile data of the bank’s customers such as their occupation, address, income etc..., and possible suspicious transactions in the past so that the machine learning classifier model can identify the fraudulent transactions based on these past data. Those fraudulent transactions can be labeled as positive (class 1) and honest ones as negative (class 0) using supervised learning methods.

1. Suppose you are given the same test dataset and two binary classifiers. Give a numerical example such that Classifier 1 has higher accuracy than Classifier 2, but Classifier 2 has both higher precision and higher recall than Classifier 1? Hint: Give a hypothetical 2x2 confusion matrix for each classifier.

**Ans 2:**

Let us assume that there is a dataset involving all sickle cell anemia cases for Sierra Leone in West Africa, for the year 2021.

Imagine that there are a total of 1000 cases detected by the Machine Learning Classifier Model, with patients identified with sickle cell anemia as class 1 and class 0 for patients who are not identified with sickle cell anemia. Suppose that there are 300 positive cases and 700 negative cases for both classifiers 1 & 2.

**For Classifier 1**, it has the following confusion matrix results:

|  |  |
| --- | --- |
| True Positive: 100 | False Negative: 200 |
| False Positive: 300 | True Negative: 400 |

**From classifier 1**, we obtain the following:

* Accuracy: (100+400)/(100+400+200+300) = **50%**
* Precision: 100/(100+300) = **25%**
* Recall: 100/(100+200) = **33.33%.**

**For Classifier 2**, it has the following confusion matrix results:

|  |  |
| --- | --- |
| True Positive: 250 | False Negative: 50 |
| False Positive: 650 | True Negative: 50 |

**From classifier 2**, we obtain the following:

* Accuracy: (250+50)/(250+50+650+50) = **30%**
* Precision: 250/(250+650) = **27.78%**
* Recall: 250/(250+50) = **83.33%.**

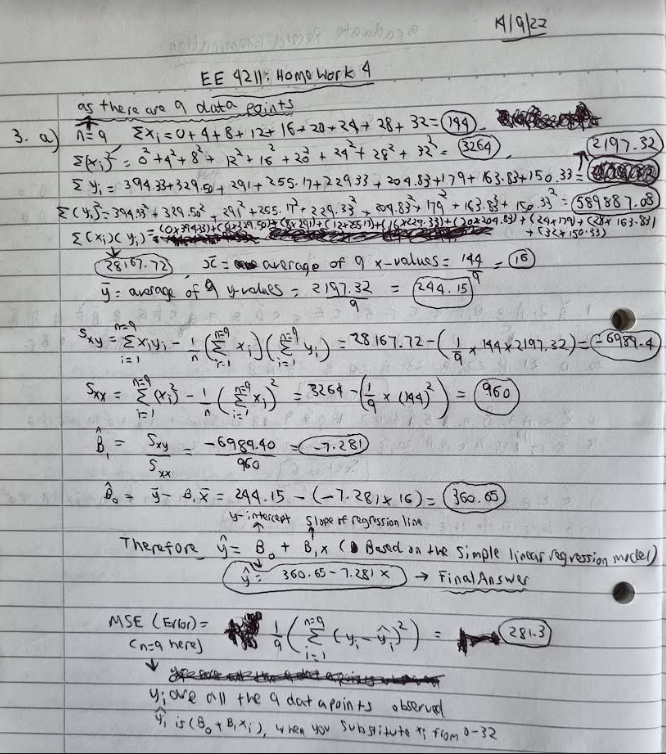
1. Consider the tire tread versus mileage problem we discussed in the lecture. The problem is to predict the tire tread depth from the mileage. The dataset, which has nine pairs of points, is reproduced below. This is an individual assignment to be done by every student. You may work in a group, but I expect every student to solve the problem and implement the code themselves.

|  |  |
| --- | --- |
| Mileage  (in 1000 miles) | Tire Tread Depth  (in mils) |
| 0 | 394.33 |
| 4 | 329.50 |
| 8 | 291.00 |
| 12 | 255.17 |
| 16 | 229.33 |
| 20 | 204.83 |
| 24 | 179.00 |
| 28 | 163.83 |
| 32 | 150.33 |

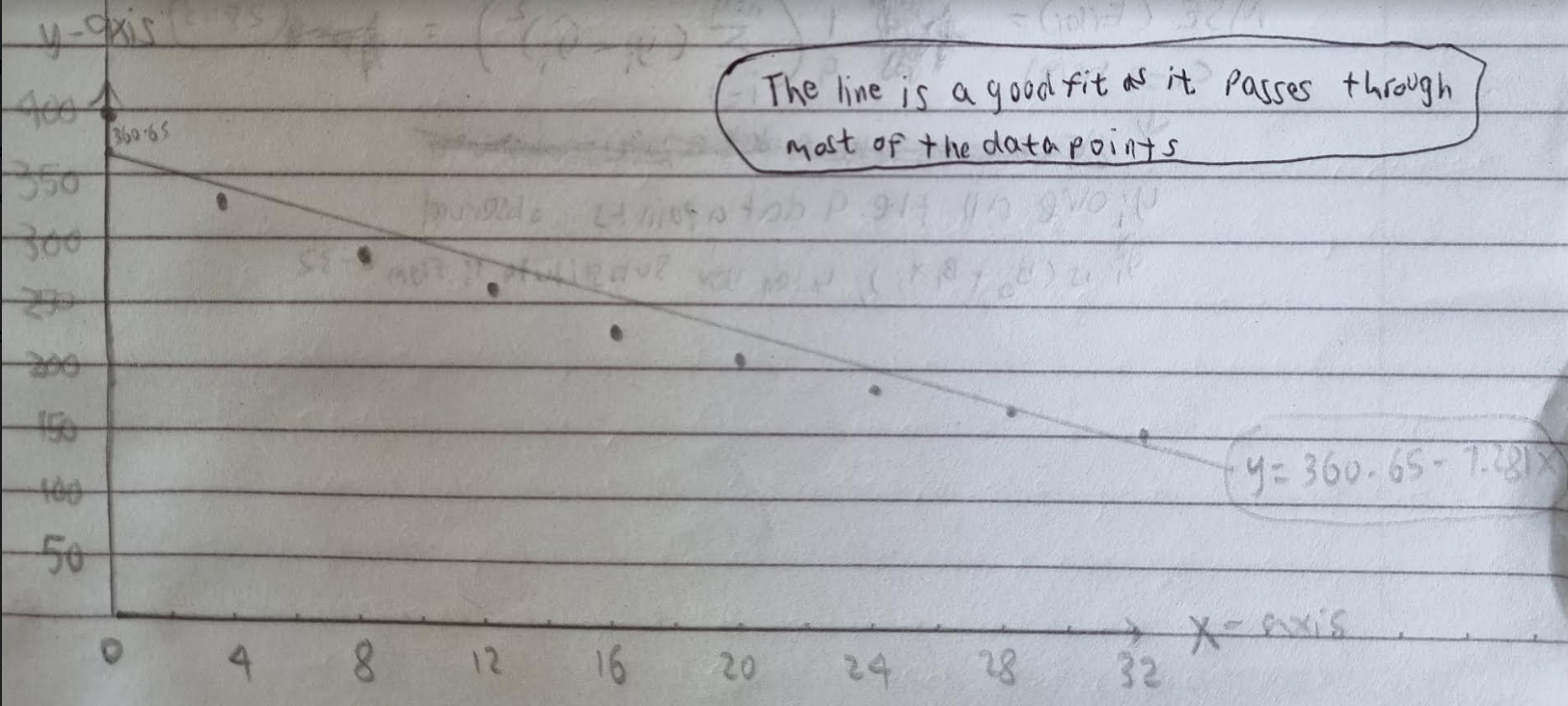
1. Compute the linear regression solution (i.e., best fit line) for this dataset. Use the entire dataset to train and find the best fit line. Give the expression for the best fit line and compute the error performance on the training dataset. Recall that the error performance is measured by the sum of squared errors. For this question, you can use Python to do the computations, but you may not use Scikit-learn to do the regression for parts a, b, & c.
2. Plot the best fit line over the data points and comment on whether the fit is good.
3. Leave out the last sample (x=32) and use it as a test data point. Use the remaining samples to train and find the best fit line. Give the expression for the best fit line. Compute the training error and the test error performance.
4. Using the entire dataset to train, find the linear regression solution using Scikit-learn and compare to the solution you got in part (a). The two solutions might be different. Can you explain why? Try doing this several times if the answers are the same.

**Ans 3:**

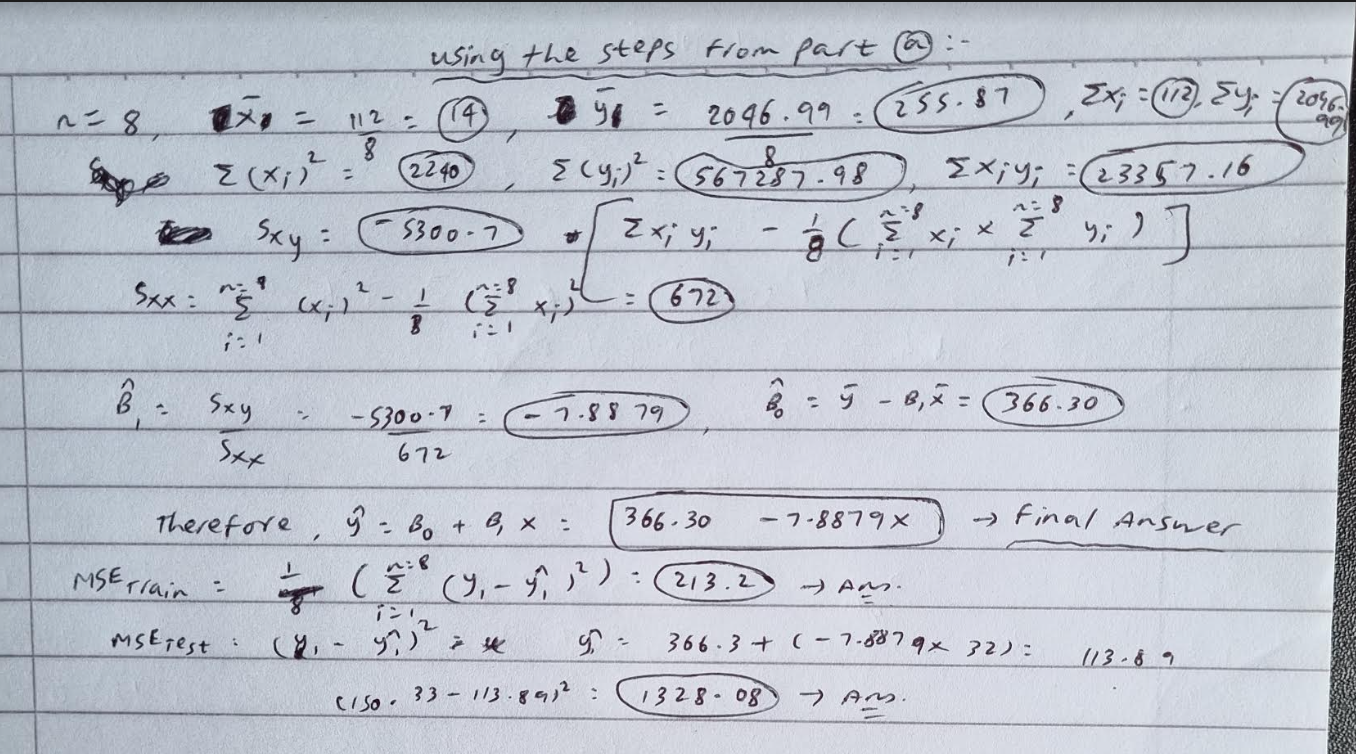
1. Below is my answer:



1. Below is my answer:



1. Below is my answer:



1. Code is attached in my .ipynb file.

Chart, scatter chart

Description automatically generated

Regarding my solution as shown above, both the calculated (black dashed line) and the sklearn-generated (purple dot) solution are similar after multiple tries for this dataset. For other datasets, however, these results might differ because sklearn does not directly implement the batch matrix calculation method as illustrated in the first part of this question i.e., part (a). Instead, it uses an approximation to minimize cost using the gradient descent method. This approximation is the one that leads to the difference in results because the gradient descent method can get easily “trapped” in local minima, which becomes largely visible in larger datasets.