**Name: Pujan Thakrar**

**UID: 805229060**

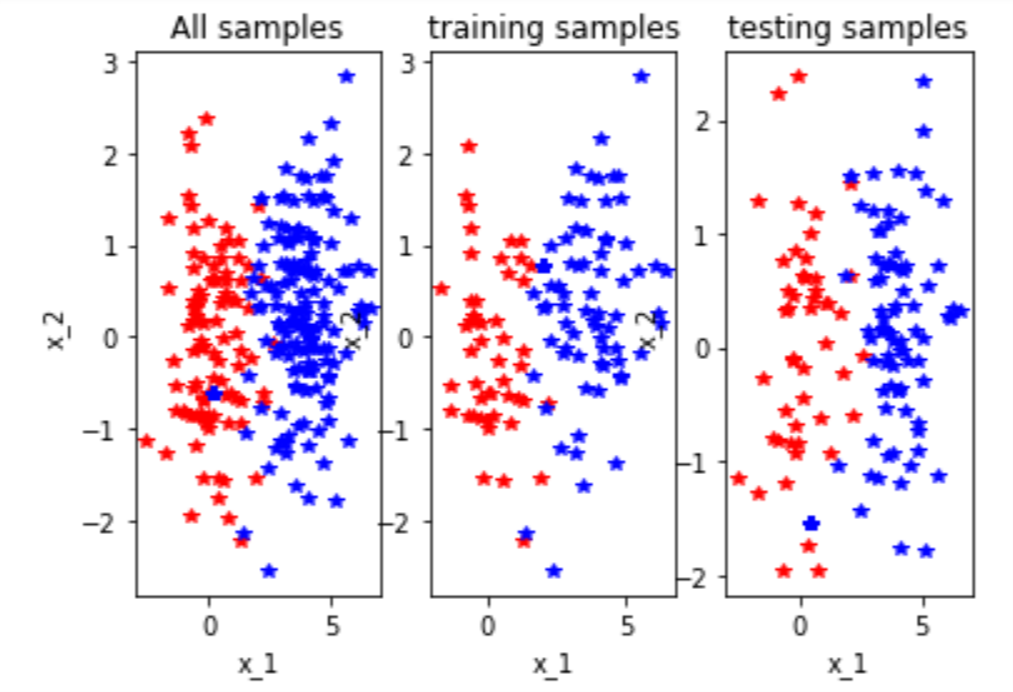
**Homework Assignment 3**

**PROBLEM 1: Logistic Regression**

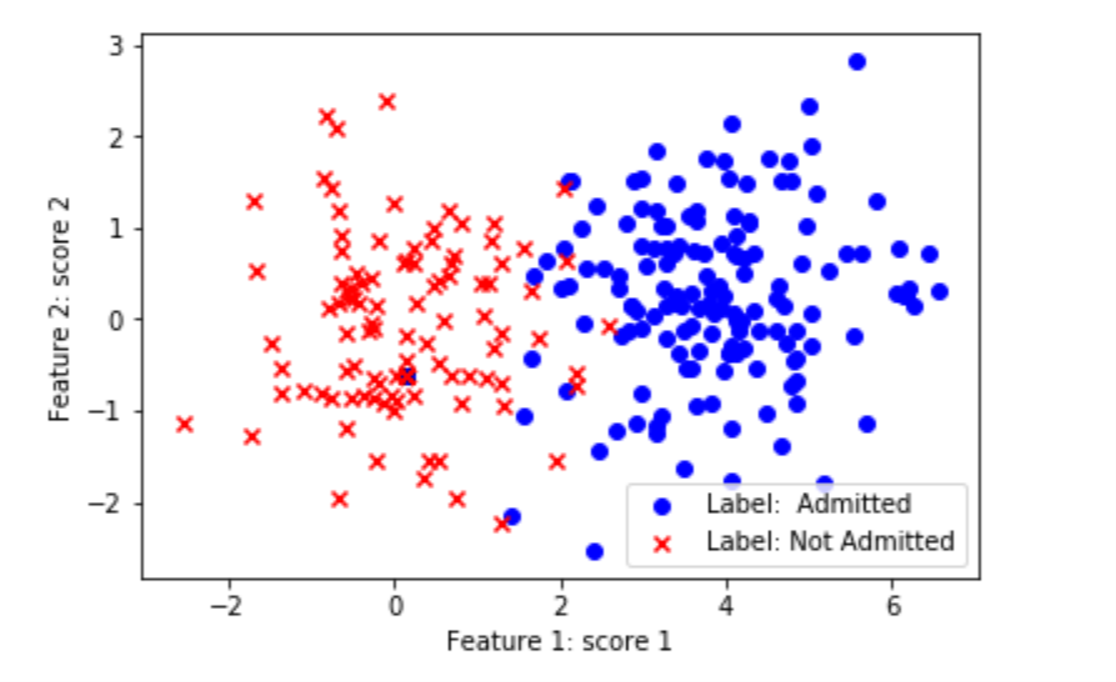
The first step was to generate the dataset, using the getDataSet() function. The dataset was composed of 250 samples (two features for each) that were associated with ground truth labels of either 0 or 1.

I split the dataset into a training dataset that consisted of 120 randomly selected samples, and the testing dataset which was composed of the 130 remaining samples.

I then plotted all the samples, the training samples and the testing samples in the following plots:



Next, I train a logistic model using my training data and the LogisticRegression() function. Then, I plot the data after assigning labels of “Admitted” and “Not Admitted”:



We have a clear division between samples that are admitted and samples that are not admitted.

We were given two ways to learn a logistic regression, the first was through the gradient descent function and the second was through the sklearn method. After testing it using both methods on the testing data, we find that the sklearn method is more accurate and gives an overall better prediction than that of the gradient descent. Here is the output:

**Scikit won.. :(**

**Your score: 0.8461538461538461**

**Scikits score: 0.9538461538461539**

The sklearn method was consistently better across several different trials.

In order to evaluate the results, we compare the predicted labels and true labels by computing the average and standard errors and get the following results:

**GD average error: 0.4336094674556213 (0.49557269616926414)**

**SK average error: 0.46579881656804734 (0.4988289075944315)**

**PROBLEM 2: Confusion Matrix**

Given the following table:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Image ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| True class | C | C | C | C | C | D | D | D | D | D | D | D | D | M | M | M | M | M | M | M |
| Predicted class | D | C | D | D | M | D | D | C | C | M | M | D | C | C | C | M | M | D | D | M |

Notes: C, cat; D, dog; M, monkey

I manually compute the following confusion matrix:

PREDICTION

GROUND TRUTH

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | 1 | 3 | 1 |
|  | 3 | 3 | 2 |
|  | 2 | 2 | 3 |

Then I compute the following accuracy, precision, and recall rates:

ACCURACY

Accuracy = True positive + True negative/ Total

Accuracy = (1 + 3 + 3 )/20

Accuracy = 7/20

**Accuracy = 0.35**

PRECISION

Cat Precision = 1/(1+3+2)

**Cat Precision = 1/6**

Dog Precision = 3/(3+3+2)

Dog Precision = 3/8

**Dog Precision = 0.375**

Monkey Precision = 3/(1+2+3)

Monkey Precision = 3/6

**Monkey Precision = 0.5**

RECALL

Cat Recall = 1/(1+3+1)

Cat Recall = 1/5

**Cat Recall = 0.2**

Dog Recall = 3/(3+3+2)

Dog Recall = 3/8

**Dog Recall = 0.375**

Monkey Recall = 3/(2+2+3)

**Monkey Recall = 3/7**

**PROBLEM 3: Comparative Studies**

Finally, I create a function in Python to compute the confusion matrix of our dataset along with the accuracy, precision, and recall rates. I then use it to compute the confusion matrix for both the gradient descent method and the sklearn method. I get the following results:

**GD Confusion Matrix:**

**Confusion Matrix:**

**[[ 0. 32. 16.]**

**[ 1. 0. 82.]]**

**Accuracy: 0.8769230769230769**

**Precision 0: 1.0**

**Precision 1: 0.8367346938775511**

**Recall 0: 0.6666666666666666**

**Recall 1: 1.0**

**SK Learn Confusion Matrix:**

**Confusion Matrix:**

**[[ 0. 45. 3.]**

**[ 1. 3. 79.]]**

**Accuracy: 0.9538461538461539**

**Precision 0: 0.9375**

**Precision 1: 0.9634146341463414**

**Recall 0: 0.9375**

**Recall 1: 0.9634146341463414**