Assignment 1, Ian Chavez

1. How many parameters are you learning in your model?

1365 columns and thus 1365 parameters in my learning model.

```
TERMINAL
  1365 columns
  1365 columns
  1365 columns
  1365 columns
  1365 columns
  1365 columns
  1365 columns
  1365 columns
  --------------------
```

```
while (scanner.hasNextLine()) {
    String line = scanner.nextLine();
    String[] columns = line.split(regex:",");
    double[] data = new double[columns.length];
    print(columns.length + " columns"); // for report
```

2. What is the train set Accuracy as printed by your code?

The training set's accuracy was 0.58.

```
====================
Accuracy: 0.5823654894524769
```

3. What are the train set Precision, Recall and F1-measure (also called F1-score) of the positive (spam) class as printed by your code?

The training set's positive measurements were: Precision (0.07), Recall (0.31), and F1-score (0.12).

```
====================
Accuracy: 0.5823654894524769
--------------------
Precision (Positive): 0.07111935683364255
Precision (Negative): 0.9000768639508071
--------------------
Recall (Positive): 0.30666666666666664
Recall (Negative): 0.609261186264308
--------------------
F1 (Positive): 0.11546184738955824
F1 (Negative): 0.7266521874030406
--------------------
```

4. What are the train set Precision, Recall and F1-measure (also called F1-score) of the negative (ham) class as printed by your code?

The training set's negative measurements were: Precision (0.90), Recall (0.61), and F1-score (0.73).

```
==================
Accuracy: 0.5823654894524769
------------------
Precision (Positive): 0.07111935683364255
Precision (Negative): 0.9000768639508071
------------------
Recall (Positive): 0.30666666666666664
Recall (Negative): 0.609261186264308
------------------
F1 (Positive): 0.11546184738955824
F1 (Negative): 0.7266521874030406
------------------
```

5. What is the confusion matrix for the train set as printed by your code?

Confusion Matrix: [115 1502]
[260 2342]

```
------------------
Confusion Matrix:
[115 1502]
[260 2342]
==================
```

6. What is the test set Accuracy as printed by your code?

The test set's accuracy was 0.62.

```
==================
Accuracy: 0.6158245948522403
```

7. What are the test set Precision, Recall and F1-measure (also called F1-score) of the positive and the negative class as printed by your code?

The training set's positive measurements were as follows: precision (0.07), recall (0.29), and F1-measure (0.11).

The test set's negative measurements were as follows: precision (0.91), recall (0.65), and F1-measure (0.75).

```
--------------------
Precision (Positive): 0.07142857142857142
Precision (Negative): 0.9051094890510949
--------------------
Recall (Positive): 0.2857142857142857
Recall (Negative): 0.6471816283924844
--------------------
F1 (Positive): 0.1142857142857143
F1 (Negative): 0.7547169811320756
--------------------
```

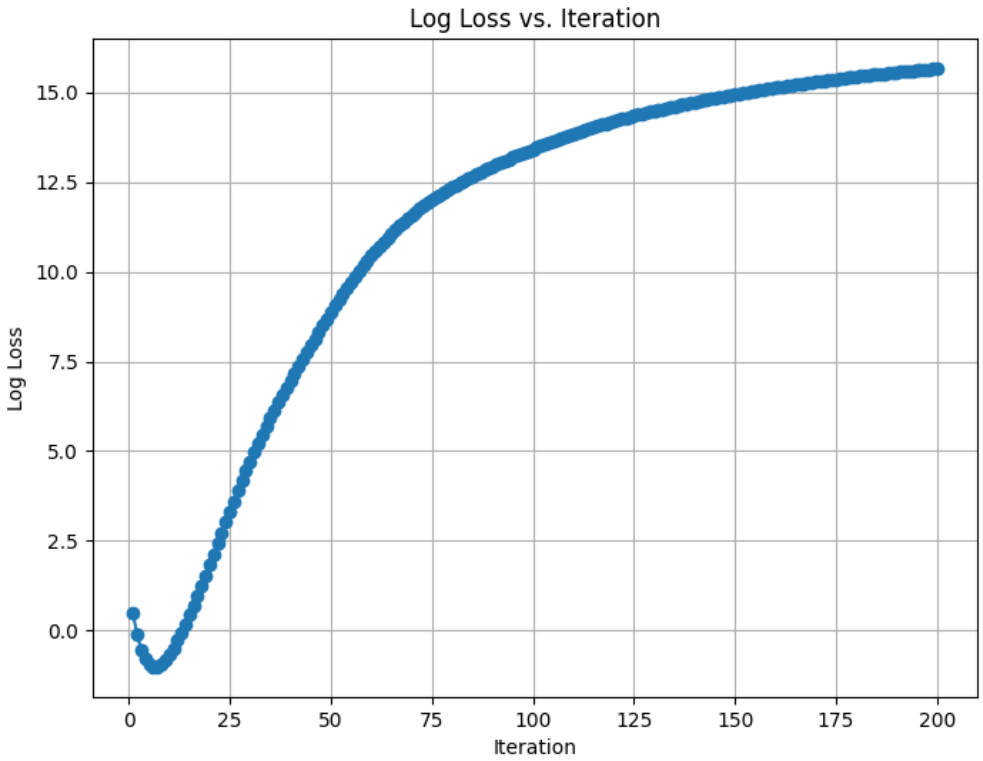
8. What is the confusion matrix for the test set as printed by your code?

Confusion Matrix: [26 338]

[65 620]

```
Confusion Matrix:
[26 338]
[65 620]
```

9. Draw a plot of log loss of the training data (y-axis) versus training iteration (x-axis), and include it in your report. You don't need to include the code for generating the plot.



What is the total cost of your model? Use the below formulas (ignore b parameter (bias)):

$$J(w,b) = \frac{1}{4459} \sum_{i=0}^{4459-1} [-y^i log(\frac{1}{1+e^{-x}}(w * x^i), y^i]$$

```
// double logLoss = -((y * Math.log(prob)) + ((1 - y) * Math.log(1 - prob)));
// totalLogLoss += logLoss;
```

```
with open("log_loss.txt", "r") as file:
    log_loss_values = [float(line.strip()) for line in file]

total_cost = sum(log_loss_values) / len(log_loss_values)

print("Total Cost:", total_cost)
```

```
ment 1> python .\calculateCost.py
Total Cost: 11.231654181483469
```

The total cost of my model is 11.23.

10. Count the number of positive and negative instances in your train file (you don't need to report them). Keeping those counts in mind, what can you do to improve the performance of your classifier on this corpus (apart from what you are already required to do in this homework)?

Positives: 375, Negatives: 4084.

The dataset seems to be heavily imbalanced in favor of negatives as it seems to be larger by a factor of 10. To compensate for this, we can do a couple of things:

- Undersample by randomly removing instances from the negative class until both are around the same (~375)
- Oversample by duplicating positives until we get around the same (~4084)
- Weighted instances where the minority's weight would be equal to the majority % / minority %.