INDIAN INSTITUTE OF TECHNOLOGY, DELHI

ASSIGNMENT REPORT

COL774 : Assignment 2

APAR AHUJA | ENTRY NO. 2019CS10465

Course - COL774 | Prof. Parag Singla

Compiled on October 10, 2021

Contents

Chapter 1

Text Classification

1.1 *a*

Training Time = 248.4 sec
Training Set Accuracy = 70.12%
Training Set Macro F1 = 0.49
Test Set Accuracy = 66.21%
Test Set Macro F1 = 0.27
Vocabulary Size = 159112

1.2 *b*

Training Set Accuracy:
Random Guess Model = 20.02%
Maximum Frequency Model = 51.86%

Test Set Accuracy:
Random Guess Model = 19.97%
Maximum Frequency Model = 66.09%

Test Set Macro F1:
Random Guess Model = 0.14
Maximum Frequency Model = 0.16

Our algorithm has significantly better accuracy than the random guess model i.e. the model is learning. But our model is not significantly better than the maximum frequency model i.e. our model is not learning very well. This is probably due to the skewed dataset.

1.3 *c*

Confusion Matrix: $\begin{bmatrix} 18 & 0 & 32 & 41 & 137 \\ 3 & 3 & 36 & 128 & 156 \\ 1 & 2 & 59 & 473 & 551 \\ 5 & 2 & 70 & 845 & 2186 \\ 19 & 12 & 95 & 782 & 8344 \end{bmatrix}$

Category 5 has the highest diagonal entry. That means among all the correctly classified data most of the data belongs to the fifth category. We also observe that for rows 1, 2, 3, 4 the value at the fifth column is the highest i.e. Pr(predicted category = 5 | actual category = 1, 2, 3 or 4) is high. Thus, every misclassified data point from the first 4 categories is mostly predicted as category 5 i.e. Pr(predicted category = 5 | misclassified) is high. We can note that the value at row 5 is the highest for columns 1, 2, 3 and second highest for column 4 i.e. Pr(actual category = 5 | predicted category = 1, 2, 3 or 4) is high. Thus, if the model predicts classes 1, 2, 3 or 4 then chances are it actually belongs to class 5 i.e. Pr(actual category = 5 | prediction) is high.

1.4 *d*

Training Time = 500.99 sec
Training Set Accuracy = 69.01%
Training Set Macro F1 = 0.49
Test Set Accuracy = 65.61%
Test Set Macro F1 = 0.27
Vocabulary Size = 132286

1.5 *e*

I tried the model on fours different features - bigrams, word count, character count and average word length. I used average word length = character count / word count. It was not a good feature so I've commented it in code and excluded it below.

Methodology
1. Bigrams - I appended consecutive words together and added them into the vocabulary forming a bag of bigrams. This gave poor results so I added back individual words in the vocabulary, forming a bag of words and bigrams. This was my best model for the dataset.
2. Character/Word Count and Average Word length - I divided these features into classes A, B, C, D, E depending on the range in which the value lies, eg. class A if word count < 200 and so on. I used naive bayes to learn the class occurrences for different labels and used it for prediction.

A few other observations on different features -

1. Baseline (*part d*): Accuracy = 65.61 % and Macro F1 = 0.27
2. Only bigrams: Accuracy = 62.91% and Macro F1 = 0.26
3. Single words and bigrams: Accuracy = 66.29% and Macro F1 = 0.20
4. Single words and character count: Accuracy = 66.08% and Macro F1 = 0.26
5. Single words and word count: Accuracy = 66.09% and Macro F1 = 0.26

1.6 *f*

Compared to the baseline bigrams along with individual words helped improve accuracy the most, but it has a poor F1 score. Character count and word count also improved accuracy with less drop in F1 values. More details on the features are given below.

F1 Scores for bigrams:
Class 1 = 0.00823
Class 2 = 0.00000
Class 3 = 0.00717
Class 4 = 0.20249
Class 5 = 0.80441
Macro F1 score = 0.20446

F1 Scores for character count:
Class 1 = 0.17629
Class 2 = 0.00578
Class 3 = 0.05542
Class 4 = 0.27143
Class 5 = 0.80383
Macro F1 score = 0.26255

F1 Scores for word count:
Class 1 = 0.16508
Class 2 = 0.00575
Class 3 = 0.06113
Class 4 = 0.26432
Class 5 = 0.80370
Macro F1 score = 0.25999

Macro F1 score looks like a more appropriate metric as -
1. False Negatives and False Positives are crucial in the model assessment on the given dataset.
2. Macro F1-score is better as there is imbalance in categories. Accuracy would have been appropriate if the class distribution was uniform/similar.

1.7 *g*

Modelling - Append summary field to the review during training. During prediction calculate log probabilities of summary and review separately and take weighted sum with summary weight equal to 6 due to its smaller length. As a result we note an increase in test accuracy and F1 score.
Training Time = 67.55 sec
Macro F1 Score = 0.3
Testing Set Accuracy: 66.99%

Chapter 2

MNIST Digit Classification

2.1 *Binary Classification*

2.1.1 *a*

Training Time = 36.71 sec
Threshold for alpha = 10^{-4}
Training Set Accuracy = 99.90%
Testing Set Accuracy = 99.30%
Number of Support Vectors = 233

2.1.2 *b*

Training Time = 32.17 sec
Threshold for alpha = 1.5x10^{-3}
Training Set Accuracy = 100%
Testing Set Accuracy = 99.19%
Number of Support Vectors = 1477

2.1.3 *c*

Train Accuracy
Linear Kernal: 99.9%
Gaussian Kernal: 100%

Test Accuracy
Linear Kernal: 99.29%
Gaussian Kernal: 99.19%

Weights
Linear Kernal:
$max(abs(\mathrm{w}_{cvxopt} - \mathrm{w}_{libsvm})) = 0.0007$

Gaussian Kernal:
Since weights can't be calculated explicitly we compare the support vector indices and the alpha values. I found that $max(abs(\alpha_{cvxopt} - \alpha_{libsvm})) = 0.001$. I also compared the support vector indices and they matched.

Bias
Linear Kernal:
$b_{cvxopt} = 1.635289423984918$
$b_{libsvm} = 1.635577983817867$
$abs(b_{cvxopt} - b_{libsvm}) = 0.0003$

Gaussian Kernal:
$b_{cvxopt} = 0.1406893622985296$
$b_{libsvm} = 0.1411406514271091$
$abs(b_{cvxopt} - b_{libsvm}) = 0.0004$

Number of Support Vectors
Linear Kernal CVXOPT = 233
Linear Kernal LIBSVM = 233
Gaussian Kernal CVXOPT = 1477
Gaussian Kernal LIBSVM = 1477

Computational Costs:
Linear Kernal CVXOPT = 36.71 sec
Linear Kernal LIBSVM = 1.03 sec
Gaussian Kernal CVXOPT = 32.17 sec
Gaussian Kernal LIBSVM = 3.15 sec

2.2 *Multi-Class Classification*

2.2.1 *a*

Macro F1 score = 0.70163
Accuracy = 68.93%

Confusion Matrix:

$$\begin{bmatrix} 976 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 1 & 1129 & 0 & 4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 230 & 138 & 603 & 10 & 7 & 0 & 7 & 4 & 32 & 1 \\ 130 & 112 & 2 & 735 & 0 & 1 & 0 & 2 & 28 & 0 \\ 193 & 112 & 0 & 0 & 608 & 0 & 5 & 0 & 52 & 12 \\ 346 & 64 & 1 & 24 & 2 & 347 & 2 & 0 & 105 & 1 \\ 211 & 67 & 0 & 0 & 1 & 0 & 664 & 0 & 15 & 0 \\ 99 & 250 & 11 & 3 & 1 & 0 & 0 & 641 & 13 & 10 \\ 134 & 181 & 0 & 2 & 0 & 1 & 2 & 2 & 652 & 0 \\ 196 & 140 & 0 & 4 & 1 & 0 & 0 & 6 & 124 & 538 \end{bmatrix}$$

F1 scores:
Class 0 = 0.55835
Class 1 = 0.67808
Class 2 = 0.73135
Class 3 = 0.82031
Class 4 = 0.75905
Class 5 = 0.55923
Class 6 = 0.81025
Class 7 = 0.76173
Class 8 = 0.65298
Class 9 = 0.68491

2.2.2 *b*

Training Time = 119.38 sec
Total number of Support Vectors = 10493
Training Set Accuracy = 99.92%
Test Set Accuracy = 97.23%
Testing Macro F1 Score = 0.97214
F1 scores:
Class 0 = 0.98376
Class 1 = 0.98897
Class 2 = 0.96339
Class 3 = 0.97188
Class 4 = 0.97665
Class 5 = 0.97303
Class 6 = 0.97813
Class 7 = 0.96717
Class 8 = 0.95927
Class 9 = 0.95919

2.2.3 *c*

Confusion Matrix:
Binary Classification: For classes 5 and 6

$$\begin{bmatrix} 951 & 7 \\ 8 & 884 \end{bmatrix}$$

Multi-Class Classification:

$$\begin{bmatrix} 969 & 0 & 1 & 0 & 0 & 3 & 4 & 1 & 2 & 0 \\ 0 & 1121 & 3 & 2 & 1 & 2 & 2 & 0 & 3 & 1 \\ 4 & 0 & 1000 & 4 & 2 & 0 & 1 & 6 & 15 & 0 \\ 0 & 0 & 8 & 985 & 0 & 4 & 0 & 6 & 5 & 2 \\ 0 & 0 & 4 & 0 & 962 & 0 & 6 & 0 & 2 & 8 \\ 2 & 0 & 3 & 6 & 1 & 866 & 7 & 1 & 5 & 1 \\ 6 & 3 & 0 & 0 & 4 & 4 & 939 & 0 & 2 & 0 \\ 1 & 4 & 19 & 2 & 4 & 0 & 0 & 987 & 2 & 9 \\ 4 & 0 & 3 & 10 & 1 & 5 & 3 & 3 & 942 & 3 \\ 4 & 4 & 3 & 8 & 13 & 4 & 0 & 9 & 12 & 952 \end{bmatrix}$$

Observations:
0 is misclassified most often as 6
1 is misclassified most often as 2 or 8
2 is misclassified most often as 8
3 is misclassified most often as 7
4 is misclassified most often as 9
5 is misclassified most often as 3
6 is misclassified most often as 0
7 is misclassified most often as 9
8 is misclassified most often as 3
9 is misclassified most often as 4
We can notice in Fig 2.1 below that the misclassified digits are unrecognizable even to a human eye. Thus, the model seems to be performing very well.

Figure 2.1: Examples of misclassified digits

2.2.4 *d*

C = 0.00001
Test Accuracy = 9.8 %

Best Validation Accuracy = 17.15 %
Average Validation Accuracy = 12.04 %

C = 0.001
Test Accuracy = 9.8 %
Best Validation Accuracy = 17.15 %
Average Validation Accuracy = 12.04 %

C = 1
Test Accuracy = 97.0 %
Best Validation Accuracy = 97.53 %
Average Validation Accuracy = 97.32 %

C = 5
Test Accuracy = 97.22 %
Best Validation Accuracy = 97.63 %
Average Validation Accuracy = 97.43 %

C = 10
Test Accuracy = 97.22 %
Best Validation Accuracy = 97.63 %
Average Validation Accuracy = 97.43 %

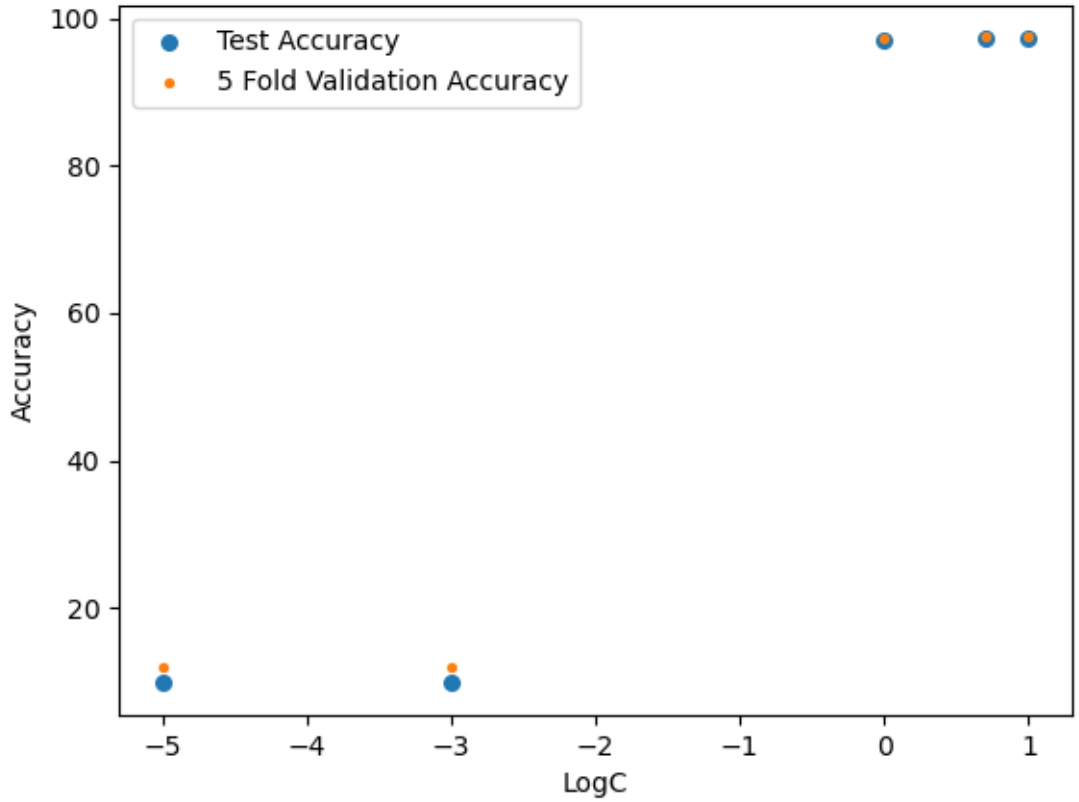

Figure 2.2: Accuracy vs LogC Plot

Observations:

1. There is a sudden jump in accuracy from C = 0.001 to C = 1.
2. Test accuracy and 5 fold validation accuracy are very close to each other.
3. The accuracy increase as C increases for the given dataset.
4. C = 5 and C = 10 give us the best accuracy on the data for 5 fold validation as well as testing.