ADDIS ABABA UNIVERSITY
ADDIS ABABA INSTITUTE OF TECHNOLOGY

Introduction to AI

Assignment 3: Report

Group Members

1. Yohannes Solomon UGR/3323/13 Section 1
2. Dawit Belay UGR/8622/14 Section 1
3. Sifan Fita UGR/8856/14 Section 2
4. Ahmed Muhammed UGR/8920/14 Section 1

MNIST Digit Dataset

For the MNIST dataset, we applied both logistic regression and naïve bayes algorithm for training. We tested different hyperparameters for each algorithm and recorded their performance, accuracy and confusion matrix.

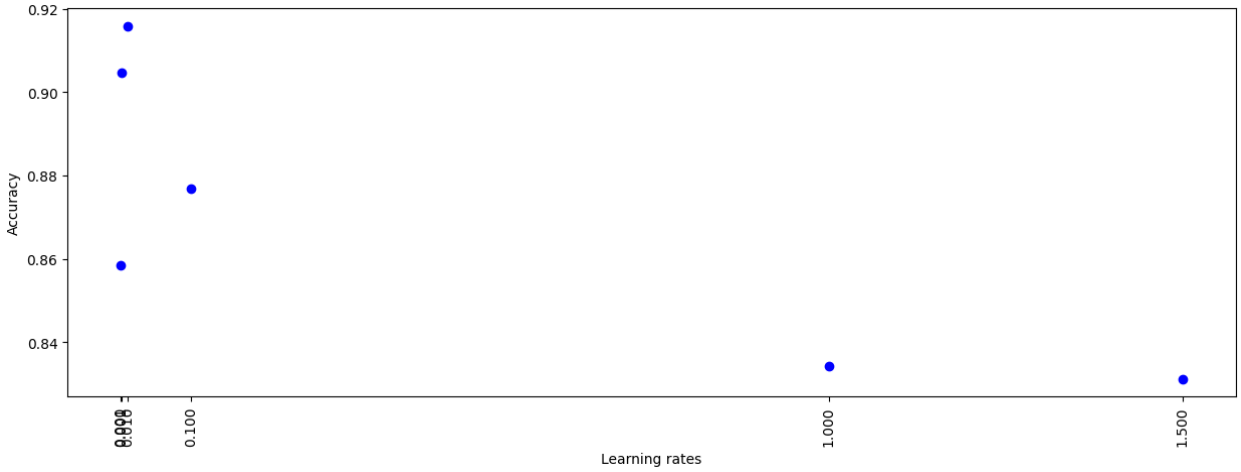
Logistic regression

For the logistic regression, we experimented with different feature extraction methods, and the following learning rates, (0.0001, 0.001, 0.01, 0.1, 1.0, 1.5).

Using all pixels as feature

The first attempt to classify MNIST dataset was feeding the whole pixel data to the logistic function and let it determine the weights of each pixels(features). We normalized the pixels between range of 0 – 1.

We had 784 pixels thus it was computationally expensive. The following shows learning rate vs accuracy graph using these features:



The following was the weights assigned for each pixel by the logistic regression function. Some of the weights resemble the number themselves. The common thing is pixels at the center of the image has more weight.

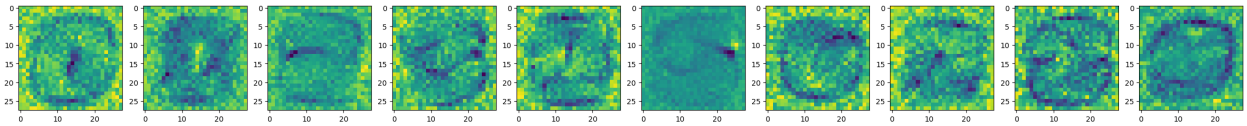


Figure 1 Weights after training

Apart from the accuracy, we evaluated the confusion matrix for each learning rate. The confusion matrices were as follows:

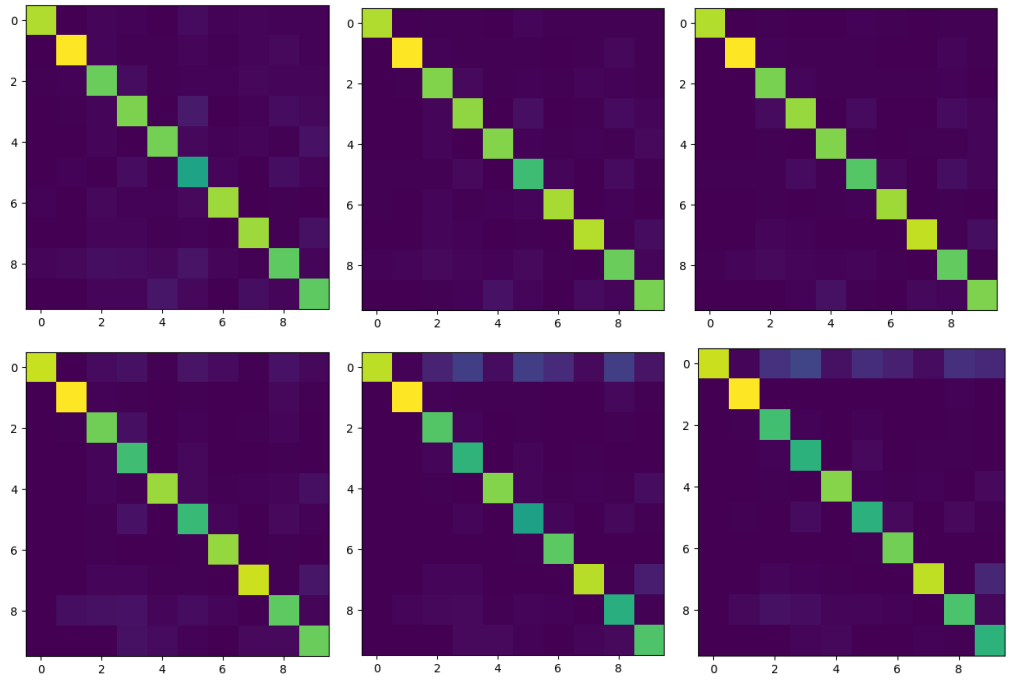
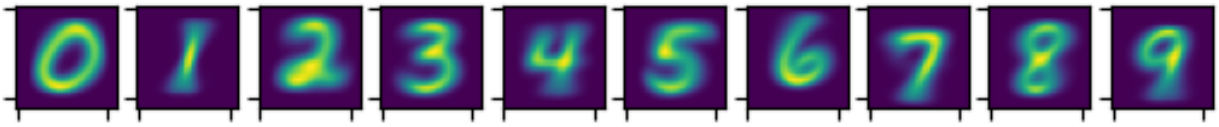


Figure 2 Confusion matrix for learning rates (0.0001, 0.001, 0.01, 0.1, 1, 1.5) [Brighter color represent higher number]

Using selected portion of image

The following image shows all samples that were in our training dataset of similar class overlayed on top of each other.



We can notice that the samples are almost perfectly aligned and most of the information required to classify them is located at the center of the image. The corner pixels are relatively less important for the classification. Thus, we filtered pixels at the center of the image and used them as feature for our samples. This extraction method decreased the dimension of feature from 784 to 215 pixels.

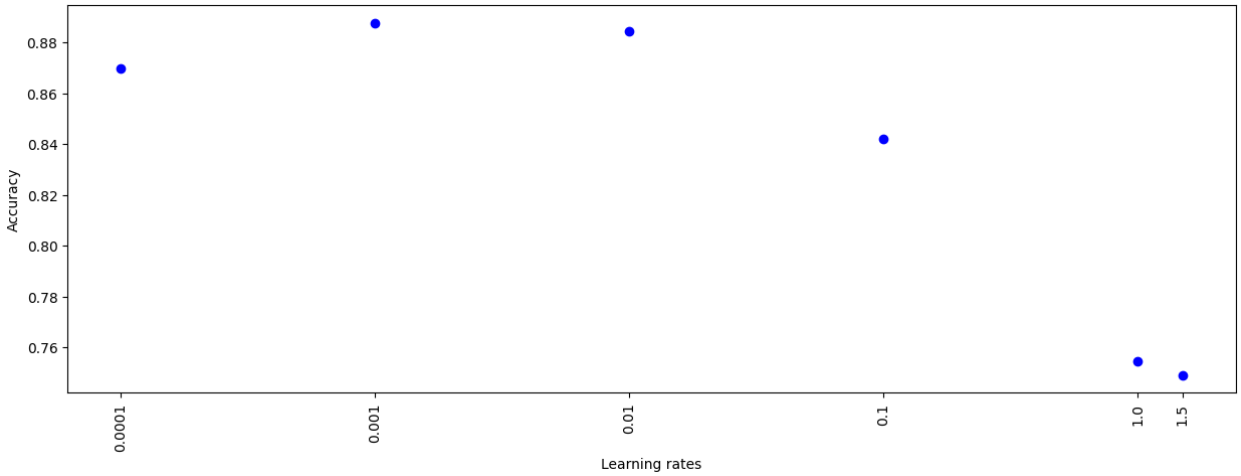


Figure 3 Accuracy vs Learning rate

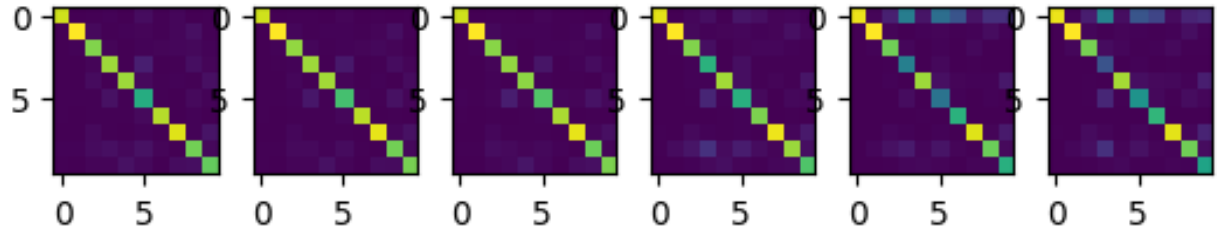


Figure 4 Confusion matrix for learning rates (0.0001, 0.001, 0.01, 0.1, 1, 1.5) [Brighter color represent higher number]

Naïve Bayes

As we did with the logistic regression, we experimented with different feature extraction methods for naïve bayes, and the following Laplace smoothing, (0.1, 0.5, 1.0, 10, 100).

Using all pixels as feature

The naïve bayes classifier achieved a peak accuracy of 75% with Laplace smoothing 0.1.

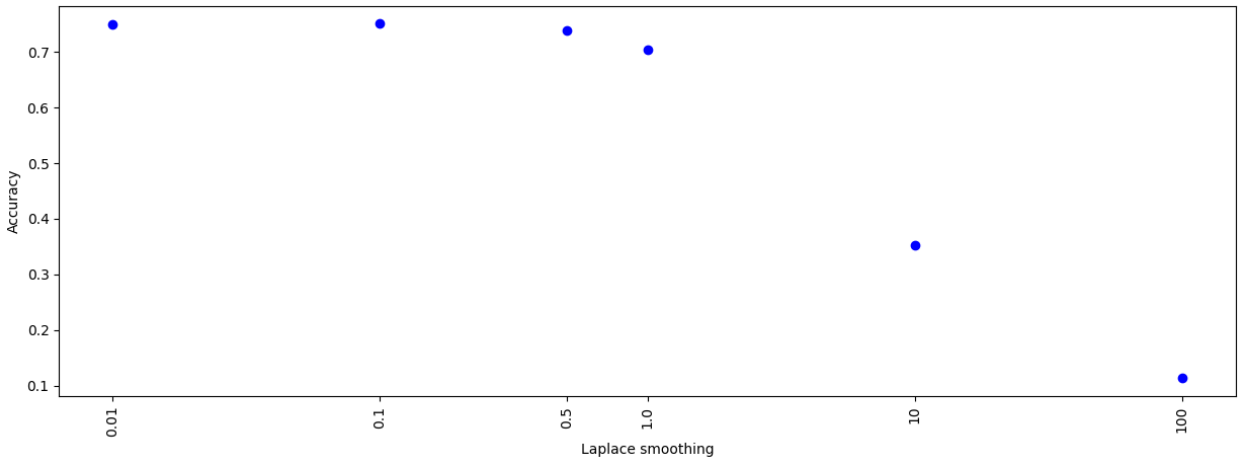


Figure 5 Laplace smoothing vs Accuracy (Naive bayes)

From the confusion matrix, we noticed that number 1 was usually easily classified while 5 is the most difficult for the model to classify accurately. Other confusions were 4 with 9 and 3 with 8. This seems to be the result of their shape similarity. Large Laplace smoothing was shown to be problematic for this dataset.

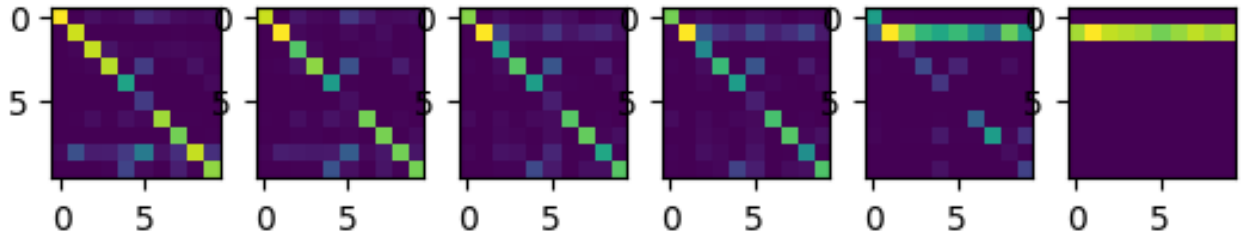


Figure 6 Confusion matrix (Laplace smoothing left to right 0.01, 0.1, 0.5, 1.0, 10, 100)

Using selected portion of image

In this experiment, the naïve bayes algorithm achieved a peak accuracy of 69.5% with Laplace smoothing 0.01.

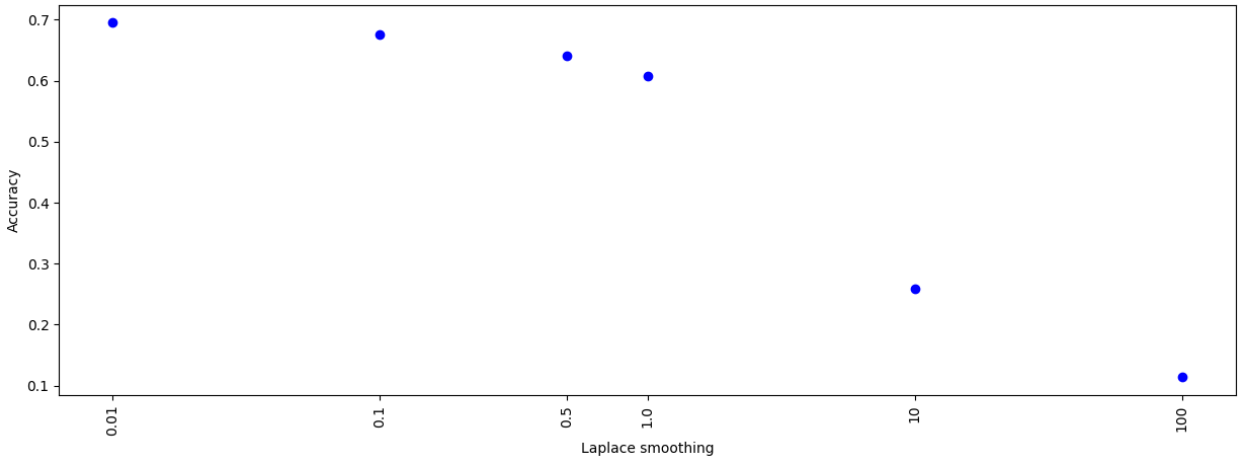


Figure 7 Laplace smoothing vs Accuracy (Naive bayes with pixel filtering)

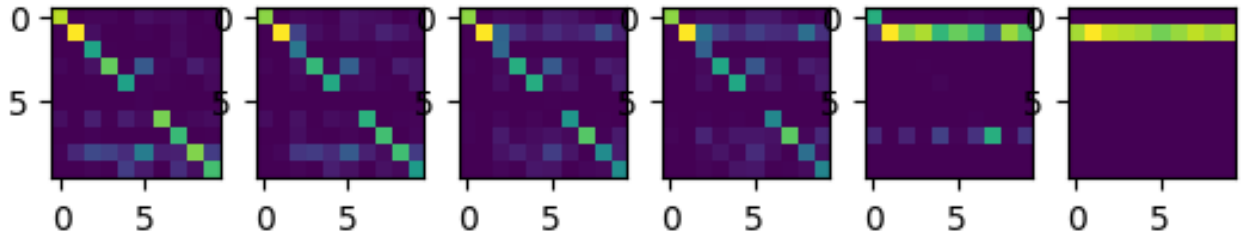


Figure 8 Confusion matrix (Naive bayes with pixel filtering)

BBC Text Dataset

For BBC Text dataset we used both naïve bayes and logistic regression algorithms. We used different hyperparameters and recorded the algorithms and recorded their performance and accuracy.

Logistic regression

Using word count and bag of words

We vectorized the sample documents using their word count and used these vectors as feature. This method reached a peak accuracy of 94% with learning rate of 0.01 that was allowed to train for 100 epochs. Higher learning rates showed decrease in performance. The following graph and confusion matrix summarize the experiment.

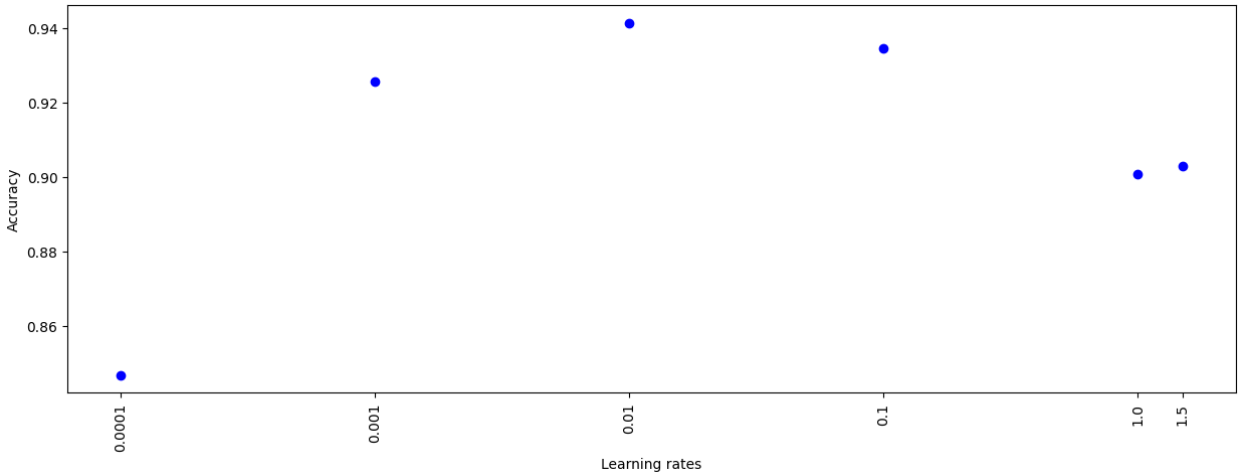
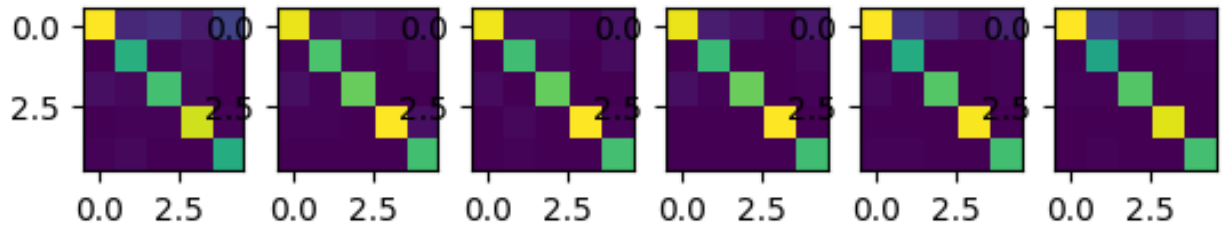
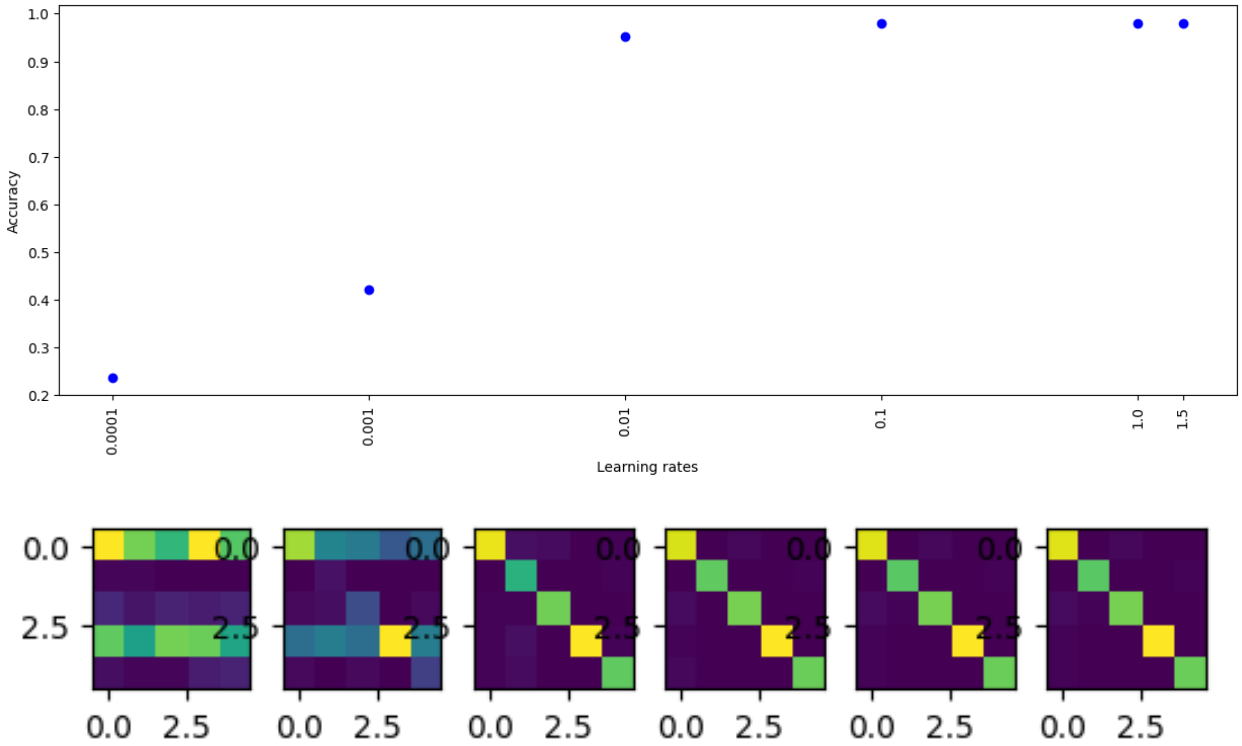


Figure 9 Learning rate vs Accuracy (Logistic regression on BBC Dataset)



Using TF-IDF

We calculated the TF-IDF for each word in the document to measure how important a term is within a document relative to a collection of documents. Using this method achieved 98% peak accuracy, with learning rate 1.5 and 100 epochs. As we can see from the accuracy graph, TF-IDF performed poorly at lower learning rates.



Naïve Bayes

We experimented the naïve bayes algorithm with different feature extraction methods, and the following Laplace smoothing, (0.1, 0.5, 1.0, 10, 100).

Using unigrams as feature

We used each word as feature to predict the category of a given text. The naïve bayes algorithm reached a peak of 99% accuracy with small Laplace smoothing factors of 0.1, 0.5 and 1.0. A larger Laplace smoothing factor like 100 significantly decreased the performance of the algorithm.

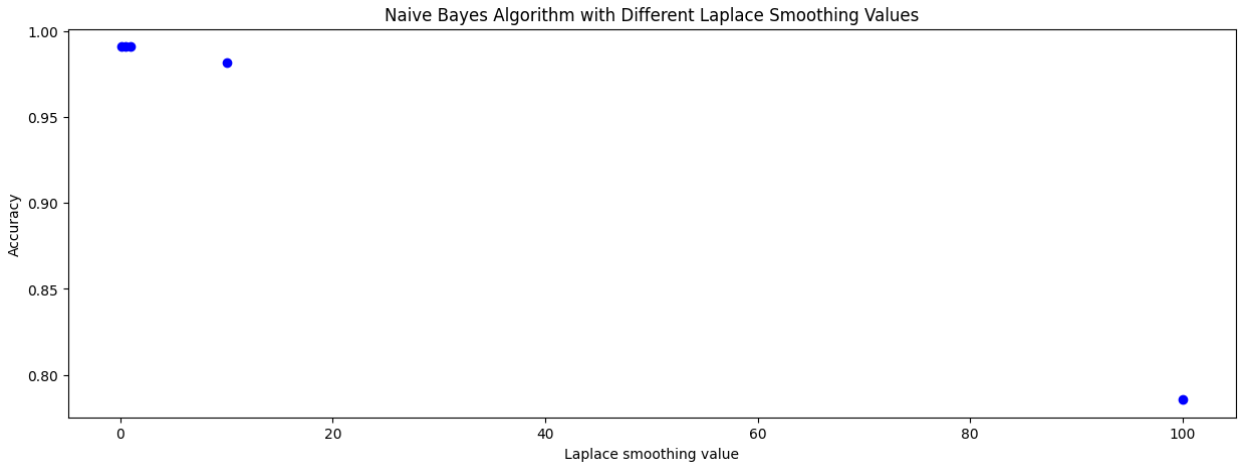


Figure 9 Laplace smoothing vs Accuracy (Naive bayes unigrams as feature)

Using bigrams as feature

We used two adjacent words of the training data as feature. Two adjacent words in a sentence often carry important contextual information that can help in distinguishing between different categories.

The naïve bayes algorithm reached a peak of 97% accuracy with small Laplace smoothing factor of 0.1. A larger Laplace smoothing factor highly affected the performance of the algorithm.

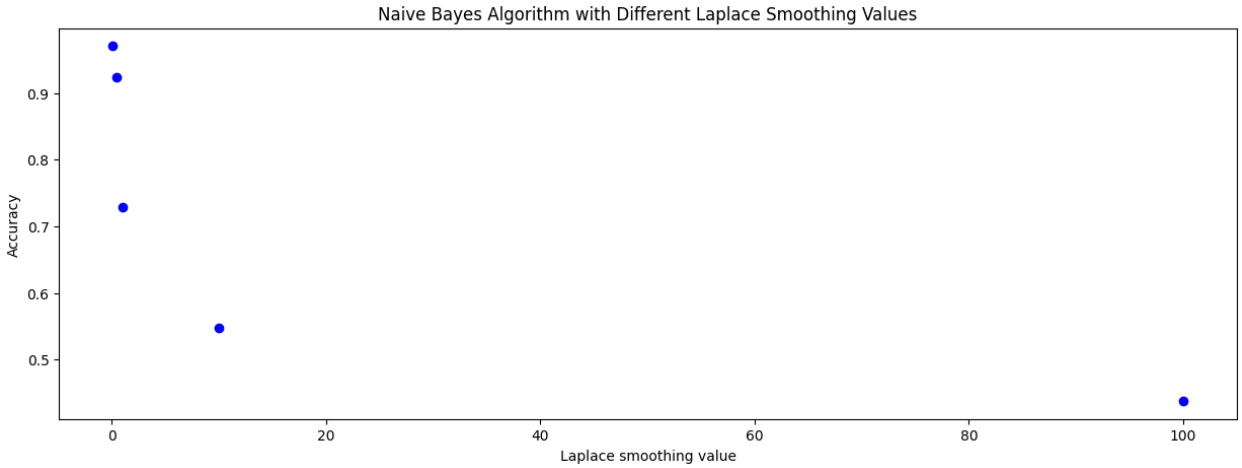


Figure 10 Laplace smoothing vs Accuracy (Naive bayes bigrams as feature)

Weather Dataset

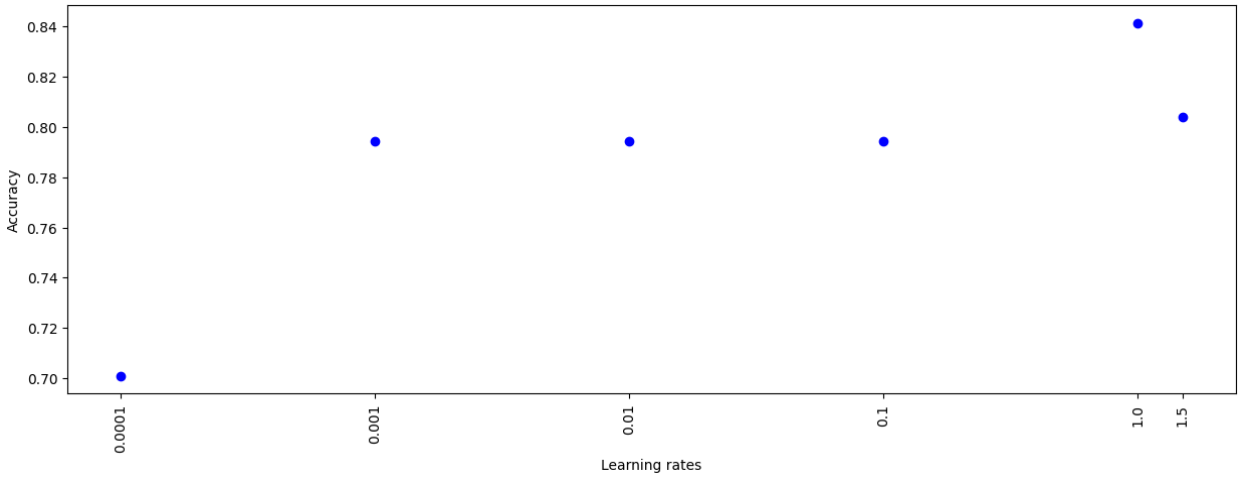
The weather dataset has the following features, continent, season, wind speed, location and weather. We were required to predict temperature based on these features.

Logistic regression

One hot encoding of features

The features in the dataset were categorical, thus to convert them into numerical features we used one hot encoding. We had 3 unique continents, 4 seasons, 3 wind speeds, 3 locations, and 4 weather. Using one hot encoding each sample was represented by vector of size 17.

This method achieved a peak of 84% accuracy with learning rate 1.0. The following graph shows accuracy as a function of learning rate.



Naïve bayes

Using one hot encoding with naïve bayes achieved a peak of 75.7% accuracy with learning rate 1.0. The following graph shows accuracy as a function of learning rate.