Lab 2: MNIST Digit Classification Using Naïve Bayes and Logistic Regression

Aazain Ullah Khan

Information and Communication Engineering Technology (ICET)

COMP 247: Supervised Learning

Merlin James

February 10, 2024

1. Introduction

In my exploration of machine learning models for MNIST Handwritten Digit Classification, I will be examining Naïve Bayes and Logistic Regression's ability to categorize digits after transforming the dataset into three broad categories. This analysis report emphasizes the models' predictive capabilities and the optimizing methods that will be tested, specifically comparing the "saga" and "lbfgs" solvers within the Logistic Regression framework. My findings will shed light on the models' strengths and weaknesses in handling complex pattern recognition tasks.

2. Problem Definition and Algorithm

2.1 Task Definition

The objective is to evaluate the performance of each algorithm in accurately predicting the categories of handwritten digits from the mnist dataset, using metrics such as accuracy and confusion matrices to assess and compare their effectiveness. Additionally, this analysis will compare the optimization techniques "saga" and "lbfgs" solvers to determine their impact on Logistic Regression's performance.

2.2 Algorithm Definition

Naïve Bayes Algorithm

The Naïve Bayes classifier applies Bayes' theorem with the "naive" assumption of independence between every pair of features. Despite its simplicity, it can yield surprisingly accurate models

for certain types of data. For the MNIST dataset, the Multinomial Naïve Bayes variant is suitable for feature vectors representing frequencies or counts.

$$P\left(\frac{B}{A}\right) = \left(\frac{P(A \cap B)}{P(A)}\right)$$

Logistic Regression

Logistic Regression, despite its name, is a linear classification model rather than regression. It predicts probabilities for binary outcomes but can be extended to multiclass classification through techniques like the One vs. Rest (OvR) scheme or using the Multinomial option for direct multiclass classification.

$$y = \frac{e^{(b_0 + b_1 X)}}{1 + e^{(b_0 + b_1 X)}}$$

3. Experimental Evaluation

3.1 Methodology

Hypotheses Tested:

Naïve Bayes with its assumption of feature independence, will efficiently classify the simplified MNIST dataset, despite potential limitations in handling feature interdependencies. On the other hand, Logistic Regression and its capacity for handling complex relationships between features,

will outperform Naïve Bayes in classifying the MNIST digits, especially in the transformed three-category scheme.

Evaluation Criteria:

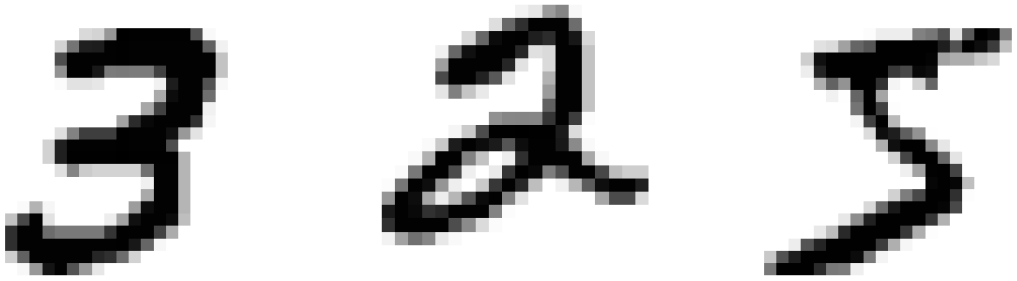
- Accuracy: The primary metric for evaluating the effectiveness of both Naïve Bayes and Logistic Regression models. It measures the proportion of correctly identified digits out of all predictions.
- Confusion Matrix: Provides a detailed breakdown of the classification performance, showing the correct and incorrect predictions for each digit category.

Training/Test Data:

- The MNIST dataset, containing 70,000 28x28 pixel grayscale images of handwritten digits, was split into a training set of 50,000 images and a test set of 20,000 images. This split ensures enough data for training the models while leaving a substantial portion for unbiased evaluation.
- The dataset was transformed to classify digits into three categories (0-3, 4-6, 7-9), aiming to explore model performance under a simplified output space.

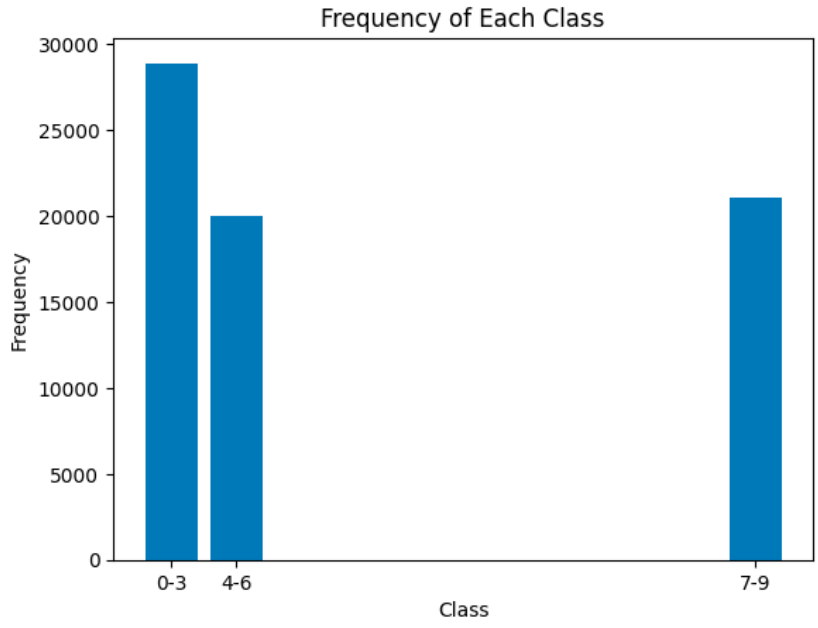
3.2 Results

Graphical plots



- some_digit1
- some_digit2
- some_digit3

Frequency of target classes



- Class 0 (digits 0-3): 28,911 instances
- Class 1 (digits 4-6): 20,013 instances
- Class 9 (digits 7-9): 21,076 instances

Naïve Bayes 3-Fold Cross-Validation

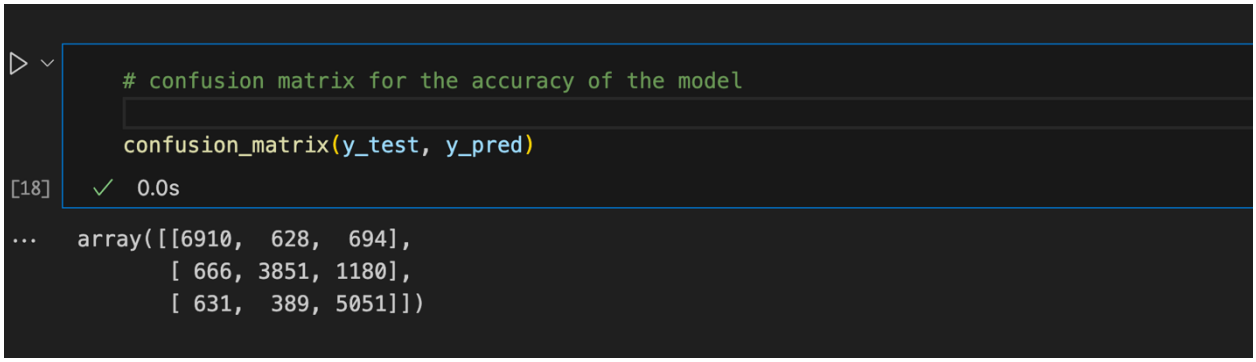
```
    # cross validation score

    cross_val_score(NB_clf_aazain, X_train, y_train, cv=3, scoring="accuracy")
[16] ✓ 0.5s
...  array([0.79138417, 0.77404452, 0.78549142])
```

- First fold: 0.79138417
- Second fold: 0.77404452
- Third fold: 0.78549142

These results showcase a relatively consistent level of accuracy across the folds, indicating that the Naïve Bayes model is stable and reliable when applied to different subsets of the MNIST dataset. The model is not overfitting to a specific part of the training data. Instead, it is learning generalizable patterns that apply across the dataset.

Naïve Bayes Confusion Matrix



True Positives (Correct Predictions):

- Class 1: 6910 instances were correctly identified.
- Class 2: 3851 instances were correctly identified.
- Class 3: 5051 instances were correctly identified.

Misclassifications (Errors):

- Class 1 was misclassified as Class 2 in 628 instances and as Class 3 in 694 instances.
- Class 2 was misclassified as Class 1 in 666 instances and as Class 3 in 1180 instances.
- Class 3 was misclassified as Class 1 in 631 instances and as Class 2 in 389 instances.

The 1180 instances where Class 2 is misclassified as Class 3 suggest similarities between these classes that the model struggles to differentiate. The model confuses certain classes more than others, likely due to shared or similar features.

Naïve Bayes Test Data Accuracy

```
# accuracy score against test data

y_pred = NB_clf_aazain.predict(X_test)
accuracy_score(y_test, y_pred)
```

[17] ✓ 0.0s

```
0.7906
```

79.06%.

This level of accuracy reflects the model's ability to generalize well from the patterns it learned during the training phase to new, unseen data.

Naïve Bayes Prediction of 3 Variables

```
# predict the 3 variables

print(NB_clf_aazain.predict([some_digit1, some_digit2, some_digit3]))
```

[19] ✓ 0.0s

```
[0 0 0]
/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packa
  warnings.warn(
```

When predicting the categories of three pre-selected digit images (as defined in point 7), the Naïve Bayes classifier outputted predictions of [0, 0, 0]. This result deviates from the expected [0, 0, 1]. This discrepancy indicates that the model misclassified the third-digit image.

Logistic Regression Training (lbfgs solver)

```
LR_clf_aazain = LogisticRegression(
    solver='lbfgs', max_iter=1200, tol=0.1, multi_class='multinomial')
LR_clf_aazain.fit(X_train, y_train)
```

```
ConvergenceWarning: lbfgs failed to converge (status=1):

STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

This optimizer is well-suited for relatively small datasets due to its limited memory usage. LBFGS optimization may struggle if the logistic regression model is too complex for the given dataset. The LBFGS optimizer may require more iterations to converge if the model is not close to the optimal solution within the specified number of iterations.

Logistic Regression Training (saga solver)

```
LR_clf_aazain = LogisticRegression(
    solver='saga', max_iter=1200, tol=0.1, multi_class='multinomial')
LR_clf_aazain.fit(X_train, y_train)
```

Completed successfully.

Logistic Regression 3-Fold Cross-Validation (saga solver)

```
    # cross validation score

    cross_val_score(LR_clf_aazain, X_train, y_train, cv=3, scoring="accuracy")
[88]  ✓  4.8s
...  array([0.89242215, 0.88834223, 0.88515541])
```

- First fold: 0.89242215
- Second fold: 0.88834223
- Third fold: 0.88515541

These results illustrate the model's generalizability and consistency across different samples of your data. The slight variations in accuracy scores across folds can be attributed to the differences in the distribution of data points within each fold. The overall close range of these scores suggests that the Logistic Regression model, optimized with the 'saga' solver, performs robustly across various segments of the dataset, with minimal overfitting or underfitting to specific subsets.

Logistic Regression Confusion Matrix (saga solver)

```
# confusion matrix for the accuracy of the model
# saga

confusion_matrix(y_test, y_pred)
```

```
[90]  ✓ 0.0s
```

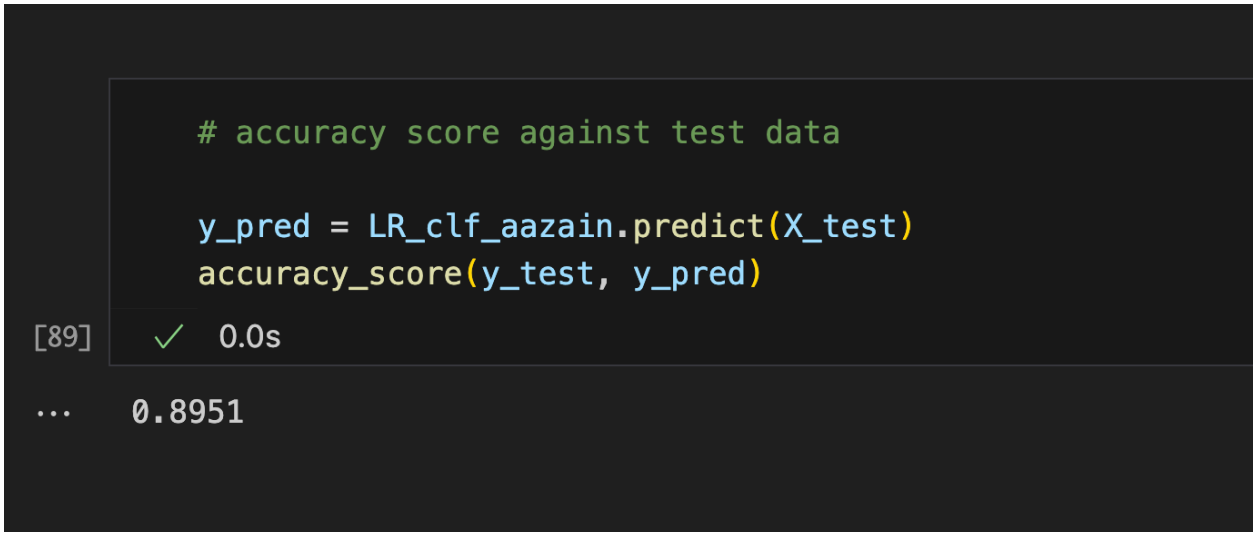
```
array([[7575,  277,  380],
       [ 374, 4966,  357],
       [ 417,  293, 5361]])
```

First Row: Represents the predictions for the first class. Out of these, 7575 were correctly identified as the first class (true positives), while 277 and 380 were incorrectly predicted as the second and third classes, respectively (false positives).

Second Row: Shows predictions for the second class, with 4966 correctly predicted (true positives). The model incorrectly classified 374 instances as the first class and 357 instances as the third class (false positives).

Third Row: For the third class, 5361 instances were correctly identified (true positives), but 417 and 293 instances were misclassified as the first and second classes, respectively (false positives).

Logistic Regression Test Data Accuracy (saga solver)



89.51%

The model correctly predicted about 89.51% of the outcomes in the test dataset, demonstrating a high level of performance in classifying the given data.

Logistic Regression Prediction of 3 Variables (saga solver)

The prediction of 3 variables resulted in an output of [0, 0, 0]. The expected output was [0, 0, 1]. This expectation was based on the transformation of the target variable,

where digits 3 and 2 fall into the first category (0-3 range) and are expected to be predicted as 0, while digit 5 falls into the second category (4-6 range) and is expected to be predicted as 1.

The expected outcome indicates a nuanced understanding of the model's classification boundaries, but the actual prediction suggests that the model may have not correctly learned or differentiated the features.

SAGA vs LBFGS solver

The 'saga' solver, known for handling large datasets efficiently and supporting L1 regularization, showed robust performance with a slight variance in 3-Fold Cross-Validation scores (0.892, 0.888, 0.885) indicating consistent model generalizability. On the other hand, the 'lbfgs' solver, ideal for smaller datasets, encountered convergence issues during training, suggesting a struggle with the dataset's complexity or size. This highlights 'saga's' superior adaptability to complex classification tasks in the Logistic Regression framework, as evidenced by its higher test accuracy (89.51%) and consistent performance across different data folds.

4. Conclusion

In conclusion, the comparative analysis of Logistic Regression (LR) and Naïve Bayes (NB) for MNIST Handwritten Digit Classification revealed LR's superior performance, particularly when optimized with the 'saga' solver. This work

highlighted the strengths of LR in handling complex data structures and achieving higher accuracy, emphasizing the importance of solver choice in model optimization. The study underscores LR's adaptability and effectiveness in image classification tasks over NB