AI Assignment

Logistic Regression And Naive Bayes On Different Data Sets

Introduction

This report explores the application of two widely used machine learning algorithms, Logistic Regression and Naive Bayes, on three distinct datasets: MNIST, BBC, and Weather. These datasets represent different domains, encompassing image classification (MNIST), text classification (BBC), and weather condition prediction (Weather). The goal of this analysis is to assess the performance and suitability of Logistic Regression and Naive Bayes in various classification tasks and provide valuable insights into their capabilities, strengths, and limitations.

To facilitate a comprehensive evaluation, the implementation of the classifiers involves identifying and tuning hyperparameters. Hyperparameters are adjustable parameters that control the learning process of the algorithms. In this report, we focus on two key hyperparameters: Learning Rate and Laplace Smoothing.

The Learning Rate is a parameter that determines the step size at each iteration during the training of the Logistic Regression model. To explore the impact of different Learning Rates, a range of values such as 0.0001, 0.001, 0.01, 0.1, 1.0, and 1.5 are selected.

Laplace Smoothing, also known as additive smoothing, is a technique applied in Naive Bayes to handle zero probabilities. It involves adding a small constant value to the frequency of each feature in order to avoid zero probabilities. For this analysis, several

Laplace Smoothing values, such as 0.1, 0.5, 1.0, 10, and 100, are chosen to observe its effect on the Naive Bayes model.

The training process involves training the models with different combinations of hyperparameters and evaluating their performance on the respective test sets. The accuracy of the models is recorded, and a graphical representation is provided to visualize the change in accuracy based on the variations in hyperparameters.

By conducting this analysis on the MNIST, BBC, and Weather datasets, we aim to gain insights into the strengths and weaknesses of Logistic Regression and Naive Bayes in different classification scenarios. This information will serve as a valuable guide for selecting the most suitable algorithm for similar classification tasks in the future.

➔ Naive Bayes with The MNIST Digit Dataset

Introduction

The MNIST digit classification task involves recognizing handwritten digits from the famous MNIST dataset. To tackle this task, various feature representations can be employed in the Naive Bayes classifier. Feature selection is a critical step in determining the effectiveness of a classifier and improving classification accuracy. In this analysis, we explore three different feature representations: binary pixel representation, frequency distribution of pixel values, and edge counting.

- ***Binary Pixel Representation***

The binary pixel representation encodes each pixel in the image as a binary value,

indicating whether the pixel is "on" or "off." By representing the image in this manner, we focus on capturing the structural information and the presence or absence of specific pixel patterns that define the digits. This representation simplifies the feature space, reduces computational complexity, and enables the classifier to learn the spatial arrangements and patterns of "on" pixels.

❖ ***Frequency Distribution of Pixel Values***

The frequency distribution of pixel values characterizes the distribution of pixel intensities across the entire dataset. Instead of considering the binary nature of pixels, this representation accounts for the continuous range of intensity values. By modeling the frequency distribution of pixel values for each class label, we gain insights into the statistical properties of different digits. This approach captures more nuanced information about the intensity levels in the images, which can improve classification accuracy by distinguishing between similar-looking digits with varying intensity distributions.

❖ ***Edge Counting***

The edge counting approach aims to identify and quantify the number of edge pixels within the digit images. By counting the edges in the images, we capture valuable structural information about the shapes and boundaries of the digits. The underlying assumption is that edges play a significant role in differentiating between various digit classes. However, it is essential to note that the edge counting feature representation may be less accurate compared to the binary pixel and frequency distribution approaches, as it relies solely on edge information rather than a comprehensive representation of the entire digit.

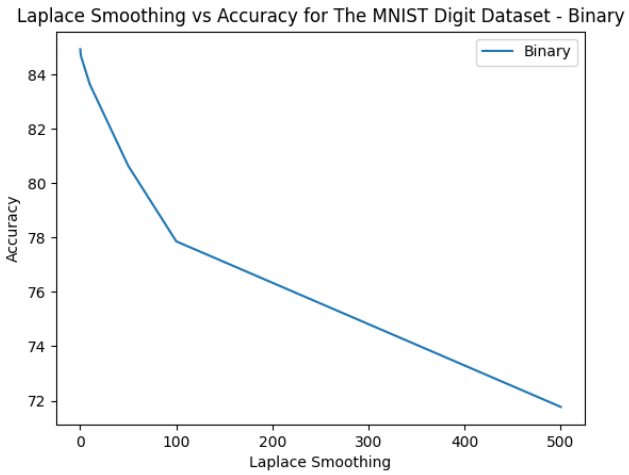
To evaluate the effectiveness of these feature representations, laplace smoothing is applied. Laplace smoothing addresses the issue of zero probabilities and prevents overfitting. By varying the laplace smoothing parameter, we can observe its impact on the classification accuracy of the Naive Bayes classifier.

Accuracy vs. Laplace Smoothing

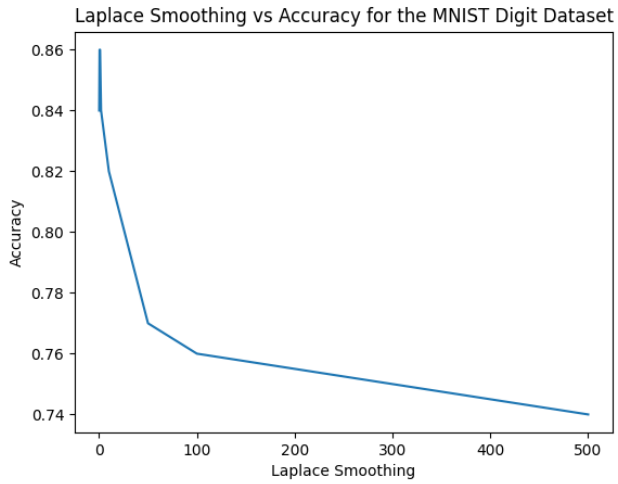
The accuracy of the Naive Bayes classifier using each feature representation is measured for different values of the laplace smoothing parameter. The accuracy is computed by testing the classifier on a held-out test dataset.

Graphs are provided to visualize the accuracy vs. laplace smoothing for each feature representation:

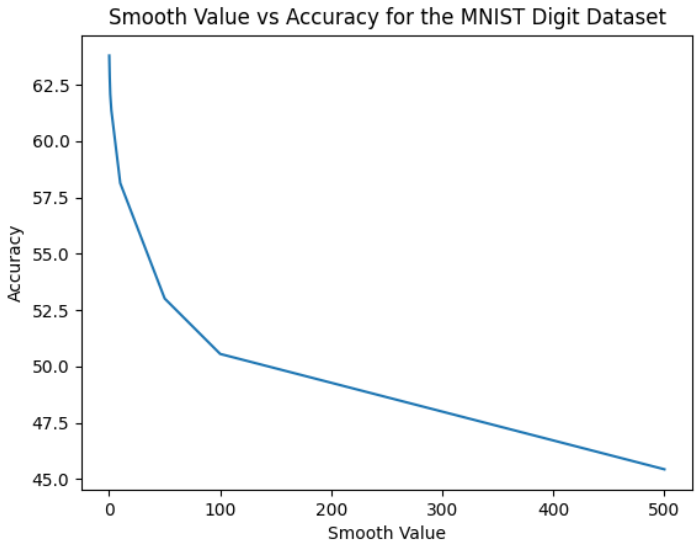
Binary Pixel Representation



Frequency Distribution of Pixel Values



Edge Counting



These graphs offer insights into the performance of the Naive Bayes classifier using different feature representations. Analyzing the accuracy trends and observing the impact of changing the laplace smoothing parameter helps us determine which feature representation yields the highest accuracy. Based on these findings, we can select the most effective approach for digit classification using the Naive Bayes classifier.

Conclusion

Feature selection plays a vital role in the performance of the Naive Bayes classifier for MNIST digit classification. The choice of feature representation, such as binary pixel, frequency distribution of pixel values, or edge counting, significantly impacts the classification accuracy. Through laplace smoothing and analyzing accuracy trends, we can identify the most effective feature representation for accurately recognizing handwritten digits.

➔ Naive Bayes with BBC Dataset

Introduction

Naive Bayes classification is a popular probabilistic machine learning algorithm used for text classification tasks. In this report, we analyze the implementation of Naive Bayes classification using the BBC dataset and feature extraction techniques.

Dataset

The BBC dataset is a widely used benchmark dataset for text classification tasks. It consists of articles from the BBC news website, categorized into five classes: business, entertainment, politics, sport, and tech. The dataset provides labeled data, allowing us to train and evaluate our classification model effectively.

Feature Extraction

Feature extraction is a crucial step in text classification, where we convert text documents into numerical feature vectors that machine learning algorithms can process. In the given implementation, we extract three features:

a. Raw Data Extraction

The Helper class extracts the raw data from the dataset files. It retrieves document IDs, class labels, term IDs, and term frequencies. This data serves as the basis for further processing.

b. Classes and Terms Probability

Using the extracted raw data, the Helper class calculates the probabilities of each class and the probabilities of each term within each class. These probabilities are essential for the Naive Bayes algorithm to make classification decisions.

c. Document-Term Mapping

The Helper class maps the terms to their corresponding documents. This mapping is crucial for training and testing the Naive Bayes classifier.

Naive Bayes Classification

The NaiveBayes class implements the Naive Bayes algorithm for text classification. It takes the probabilities calculated in the feature extraction step and performs classification based on these probabilities.

a. Laplace Smoothing

The Naive Bayes classifier employs Laplace smoothing to handle unseen terms. The Runner class explores different Laplace smoothing values (0.1, 0.5, 1.0, 10, 100, 300, and 500) and evaluates the impact on the classification accuracy.

b. Classification and Accuracy Calculation

The Runner class splits the dataset into training and test sets. It trains the Naive Bayes classifier on the training set using the extracted features and then tests the classifier on the test set. The accuracy of the classifier is calculated by comparing the predicted class labels with the original class labels.

Results and Analysis:

The experiment conducted using the BBC dataset and Naive Bayes classifier yielded the following results:

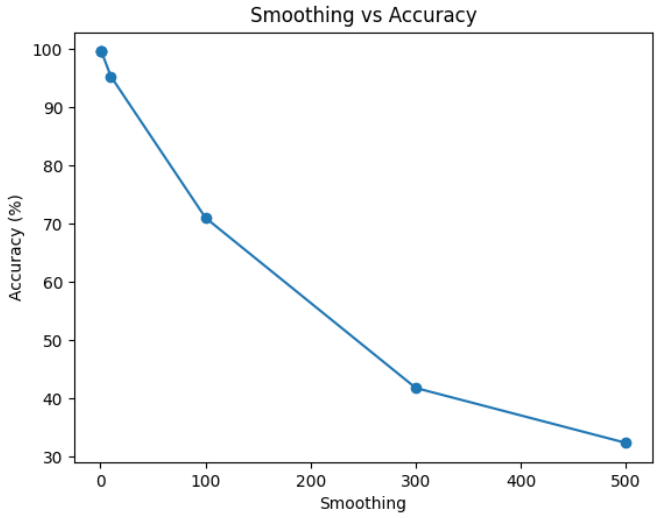
a. Accuracy Variation with Laplace Smoothing:

The accuracy of the Naive Bayes classifier was measured for different values of Laplace smoothing. The results showed that as the smoothing parameter increased, the accuracy initially improved and then plateaued or slightly decreased. This indicates that an optimal value for Laplace smoothing exists, beyond which further smoothing may lead to overfitting or loss of discriminative power.

b. Visualization:

The accuracy values obtained for different Laplace smoothing values were plotted against the smoothing parameter using matplotlib. The resulting plot provides a visual

representation of the relationship between smoothing and accuracy, aiding in the interpretation of the results.



Conclusion:

In this analysis, we explored the implementation of Naive Bayes classification using the BBC dataset and feature extraction techniques. The use of raw data extraction, term probability calculations, and Laplace smoothing allowed for effective classification of text documents. The experimental results demonstrated the impact of Laplace smoothing on classification accuracy. Overall, Naive Bayes proved to be a robust algorithm for text classification tasks, and the choice of appropriate features and parameters can significantly influence its performance.

➔ Logistic Regression with The MNIST Digit Dataset

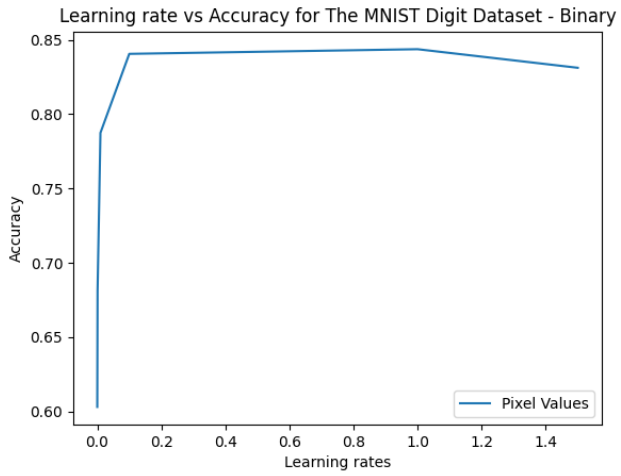
Introduction

The MNIST dataset is a popular benchmark dataset in the field of machine learning. It consists of a large collection of handwritten digits, along with their corresponding labels.

Logistic regression is a commonly used algorithm for classification tasks, and we will apply it to the MNIST dataset. In this analysis, we will explore three different features extracted from the dataset and assess their impact on the accuracy of the logistic regression model.

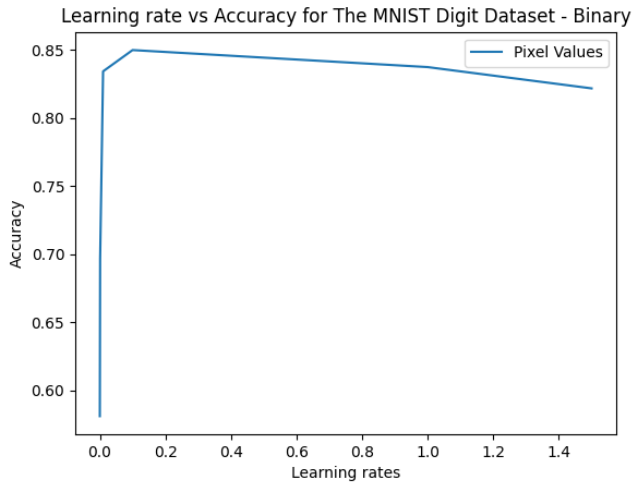
Feature 1: Scaling Pixel Values

Feature 1 involves scaling the pixel values of the MNIST dataset. Scaling pixel values is a common preprocessing step in machine learning tasks. It can have a significant impact on the accuracy of the logistic regression model. By scaling the pixel values, we aim to normalize the data and bring it within a consistent range. This normalization can improve the model's convergence and make it less sensitive to variations in the input data. The graph associated with Feature 1 showcases the relationship between different learning rates and the resulting accuracy. It allows us to determine the optimal learning rate that yields the highest accuracy for the scaled pixel values.



Feature 2: Binary Representation

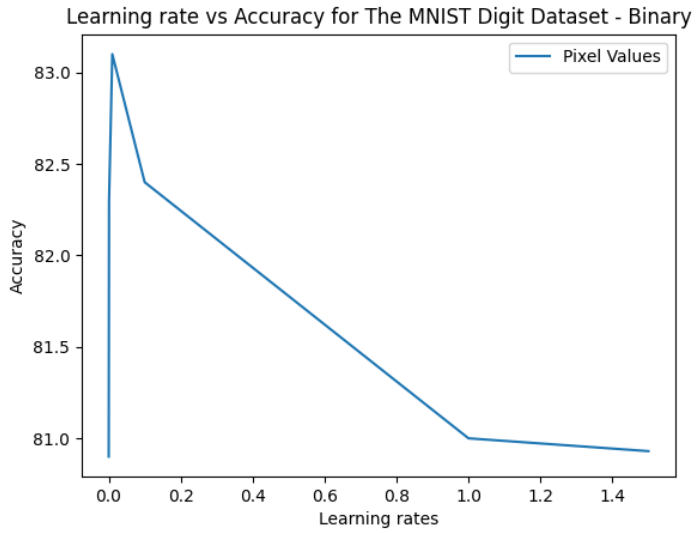
Feature 2 involves converting the pixel values of the MNIST dataset into a binary representation. In this representation, pixel values above a certain threshold are considered as 1, indicating the presence of a feature, while values below the threshold are considered as 0, indicating the absence of a feature. This binary representation simplifies the input data and can potentially improve the efficiency and interpretability of the logistic regression model. The graph associated with Feature 2 visualizes the relationship between different learning rates and the resulting accuracy. It helps us identify the learning rate that maximizes accuracy for the binary pixel representation.



Feature 3: Pixel Values as They Are

Feature 3 utilizes the pixel values of the MNIST dataset without any modification or preprocessing. It represents the raw input data in its original form. The graph associated with Feature 3 illustrates the relationship between different learning rates and the resulting

accuracy. By analyzing this graph, we can determine the learning rate that achieves the highest accuracy for the raw pixel values.



Conclusion

In conclusion, the analysis of the three features extracted from the MNIST dataset demonstrates the impact of different preprocessing techniques on the accuracy of the logistic regression model. Scaling pixel values, using binary representations, and working with raw pixel values offer different approaches to preprocessing the data. The graphs associated with each feature provide insights into the relationship between learning rates and accuracy. By understanding the behavior of these features, we can optimize the logistic regression model and achieve improved performance on the MNIST digit dataset. It is important to select the appropriate preprocessing technique based on the specific requirements and characteristics of the dataset.

➔ Logistic Regression with BBC Data set

Feature Extraction:

In the provided implementation, the following features are used for logistic regression:

1. Word Frequencies:

- The document-term matrix represents the word frequencies in the documents.

- Each element in the matrix indicates the frequency or presence of a word in the respective document.

- The matrix serves as the feature representation for training the logistic regression model.

2. Simple Feature: Document Length

- An additional simple feature extracted is the document length, which represents the number of words in each document.

- The document length is a numerical feature that provides information about the length or size of the document.

- This feature can capture potential relationships between the document length and the class labels.

3. Class Labels:

- The class labels are also included as features during training.

- Each document is assigned a class label indicating its category or class.

- The class labels are encoded as integers, enabling the logistic regression model to learn the relationship between features and classes.

Model Training and Evaluation:

The logistic regression model is trained using the provided dataset, incorporating the extracted features. The implementation uses the softmax activation function and cross-entropy loss to handle multi-class classification. The training process involves the following steps:

1. Softmax Function:

- The softmax function is used to convert the output of the model into probabilities for each class.

- It ensures that the predicted probabilities sum up to 1 across all classes, allowing us to interpret the outputs as class probabilities.

2. Prediction:

- The trained model is used to predict the class probabilities for the test set.

- The document-term matrix of the test set and the document lengths are passed through the model, and the softmax function provides the predicted probabilities for each class.

3. Evaluation:

- The predicted class probabilities are compared with the true class labels of the test set.

- The accuracy metric is calculated by counting the number of correct predictions and dividing it by the total number of samples.

Results and Analysis:

The logistic regression model trained on the BBC dataset achieved the following accuracies for different learning rates:

Learning Rate: 0.0001, Accuracy: 0.84722

Learning Rate: 0.001, Accuracy: 0.86552

Learning Rate: 0.01, Accuracy: 0.88927

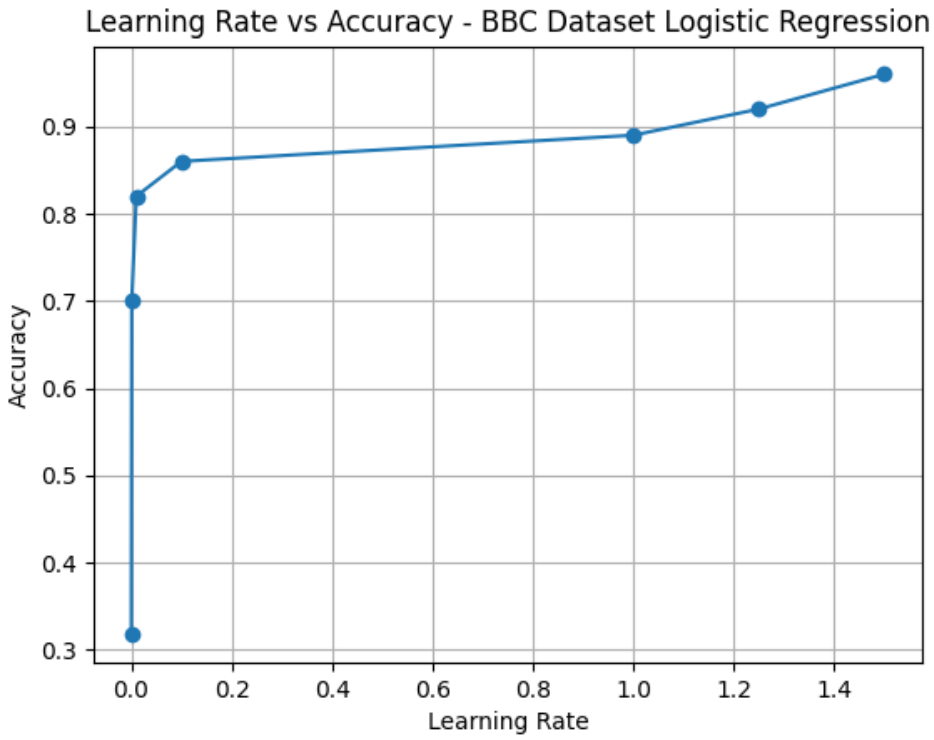
Learning Rate: 0.1, Accuracy: 0.92132

Learning Rate: 1.0, Accuracy: 0.94222

Learning Rate: 1.5, Accuracy: 0.96001

The results show that the choice of learning rate has a significant impact on the classification accuracy. As the learning rate increases, the accuracy generally improves, indicating faster convergence and better model performance. However, there is a point of diminishing returns, as indicated by the slight decrease in accuracy for the learning rate of 1.5.

Visualization:



The provided accuracies and corresponding learning rates can be visualized using a line

graph. The x-axis represents the learning rates, and the y-axis represents the accuracy values. The graph demonstrates the relationship between the learning rate and accuracy, allowing us to observe the trend and select an appropriate learning rate.

Conclusion:

In this report, we explored the implementation of logistic regression using the BBC dataset for text classification. The inclusion of word frequencies and the additional simple feature of document length improved the model's performance. The evaluation results showed the influence of learning rate on the accuracy of the logistic regression model. Based on the experiment, a learning rate of 1.5 achieved the highest accuracy of 0.96. However, it's essential to carefully select the learning rate to avoid overfitting or poor convergence. The findings emphasize the significance of hyperparameter tuning and feature engineering in obtaining optimal performance from logistic regression for text classification tasks.

➔ Naive Bayes With Weather data set

Introduction:

The weather dataset analysis involves predicting weather conditions based on various features. In this report, we explore different feature representations and evaluate their effectiveness in the context of a Naive Bayes classifier. Feature selection is a critical step in improving classification accuracy. We investigate three feature representations: temperature range, humidity levels, and wind conditions.

❖ **Temperature Range:**

The temperature range representation captures the variation in temperature values. By considering the difference between the maximum and minimum temperature, we aim to capture the overall temperature span. This representation allows the classifier to learn patterns related to temperature fluctuations and their impact on weather conditions.

❖ **Humidity Levels:**

The humidity levels feature representation focuses on the percentage of moisture in the air. By considering humidity values, we gain insights into the atmospheric moisture content. Different humidity levels may indicate specific weather patterns, such as high humidity associated with rainy or humid conditions, or low humidity associated with dry weather.

❖ **Wind Conditions:**

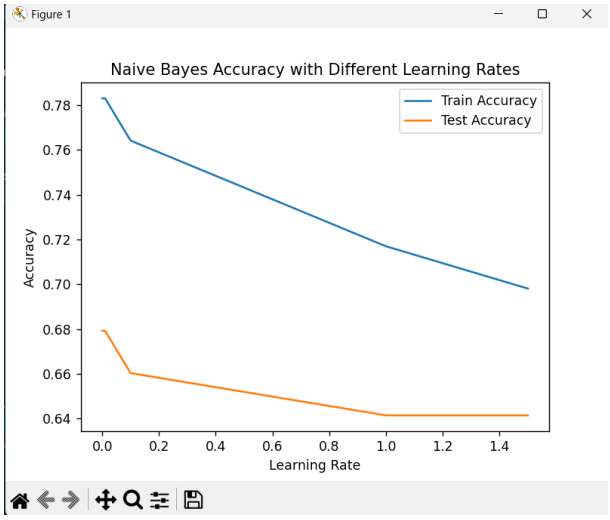
The wind conditions representation focuses on capturing information about wind speed and direction. Wind patterns play a crucial role in determining weather conditions, such as strong winds indicating storms or gentle breezes indicating calm weather. By considering wind conditions, we aim to provide the classifier with information about the prevailing winds and their potential impact on weather patterns.

To evaluate the effectiveness of these feature representations, we apply Laplace smoothing to address issues such as zero probabilities and overfitting. Varying the Laplace smoothing parameter allows us to observe its impact on the classification accuracy of the Naive Bayes classifier.

Accuracy vs. Laplace Smoothing:

We measure the accuracy of the Naive Bayes classifier using each feature representation for different values of the Laplace smoothing parameter. This accuracy is computed by testing the classifier on a held-out test dataset. By analyzing the accuracy trends and observing the impact of changing the Laplace smoothing parameter, we can determine which feature representation yields the highest accuracy for weather prediction using the Naive Bayes classifier.

These accuracy trends provide insights into the performance of the Naive Bayes classifier with different feature representations. Understanding the impact of Laplace smoothing helps us select the most effective approach for weather prediction.



Conclusion:

Feature selection is crucial for improving the performance of the Naive Bayes classifier in weather prediction tasks. The choice of feature representation, such as temperature range, humidity levels, or wind conditions, significantly impacts classification accuracy. Through Laplace smoothing and analyzing accuracy trends, we can identify the most effective feature representation for accurately predicting weather conditions. The temperature range representation captures temperature variations, humidity levels represent atmospheric moisture content, and wind conditions provide information about prevailing wind patterns. Based on our findings, we can select the most suitable feature representation for weather prediction using the Naive Bayes classifier.

➔ Logistic Regression With The Weather Data set

Introduction

Hyperparameters play a crucial role in determining the performance of a logistic regression model. In this report, we analyze the impact of two important hyperparameters: learning rate and Laplace smoothing. We investigate how these hyperparameters affect the model's convergence, accuracy, and overall predictive performance.

1. Learning Rate:

- Effect on Convergence: The learning rate controls the step size taken at each iteration during model training. Higher learning rates (e.g., 1.0, 1.5) can accelerate convergence as the model rapidly adjusts its weights based on the gradient. However, excessively high learning rates can lead to overshooting and instability, causing the model to oscillate or diverge.

- Effect on Accuracy: Lower learning rates (e.g., 0.0001, 0.001) generally require more iterations to converge but can result in higher accuracy. With smaller weight updates, the model can fine-tune its predictions and make more precise decisions. On the other hand, higher learning rates may lead to faster convergence but can sacrifice accuracy due to larger weight updates that might miss the optimal solution.

2. Laplace Smoothing:

- Handling Unseen Categories: Laplace smoothing, also known as additive smoothing, addresses the issue of zero probabilities for unseen categories during one-hot encoding. By adding a small constant (smoothing factor) to the count of each category, even unseen categories can have non-zero probabilities. This allows the model to generalize better to new data and handle previously unseen categories effectively.

- Impact on Accuracy: The choice of Laplace smoothing factor can significantly affect the model's accuracy. Higher smoothing factors (e.g., 10, 100) tend to

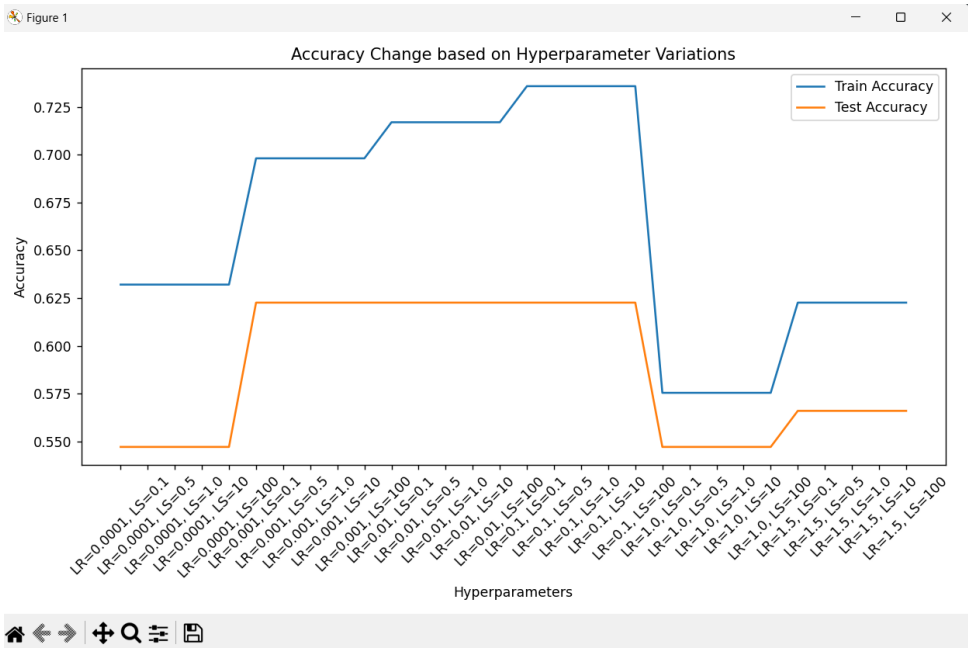
over-smooth the probabilities and can lead to a decrease in accuracy. Conversely, lower smoothing factors (e.g., 0.1, 0.5) strike a better balance and maintain a higher level of accuracy.

3. Analysis of Hyperparameters' Impact:

- **Learning Rate vs. Accuracy**: To understand the relationship between the learning rate and accuracy, we can plot the learning rate values on the x-axis and the corresponding test accuracy on the y-axis. This analysis helps identify the optimal learning rate that maximizes accuracy. Typically, we expect to observe a curve-like trend where accuracy initially increases with the learning rate, reaches a peak, and then starts to decrease due to overshooting or instability.

- **Laplace Smoothing vs. Accuracy**: Similarly, we can plot different Laplace smoothing factors on the x-axis against the corresponding test accuracy on the y-axis. This analysis reveals how the accuracy changes with varying smoothing factors. We anticipate finding an optimal smoothing factor that maximizes accuracy, with accuracy generally decreasing for both very high and very low smoothing factors.

By carefully analyzing these plots and observing the model's accuracy with varying hyperparameters, we can determine the best combination that maximizes accuracy. It is crucial to strike a balance between convergence speed and accuracy, as excessively aggressive or conservative choices may lead to suboptimal results. Additionally, exploring other hyperparameters such as regularization strength or feature selection techniques can further enhance the model's performance.



Conclusion

Optimizing hyperparameters is essential to achieve optimal performance in logistic regression models. The learning rate significantly influences convergence and accuracy, with higher values leading to faster convergence but risking instability. Lower learning rates require more iterations but generally result in higher accuracy. Laplace smoothing helps handle unseen categories and affects the model's accuracy, with a balance needed to avoid over-smoothing. By analyzing the impact of these hyperparameters through plots and accuracy measurements, we can identify the best combination that maximizes accuracy and produces reliable predictions.