Lab1: Naive Bayes Classifier

Mamoru Ota, S7784976

***Abstract*—This report describes an approach to data classification using the Naive Bayes classifier. The Naive Bayes classifier was applied to the weather and Iris datasets, incorporating Laplace smoothing and log probability to improve handling of missing data and to enhance computational stability. We also employed binarization for continuous features in the Iris dataset. This assignment aims to solidify the understanding of Naive Bayes classifiers and the importance of various optimizations such as smoothing and log-transformations.**

I. Introduction

In this report, we tackle the problem of classifying data using the Naive Bayes classifier, focusing on the weather and Iris datasets. The Naive Bayes classifier is a probabilistic classifier that applies Bayes' theorem with the assumption that features are conditionally independent given the class. Despite this strong assumption, it performs well for many real-world applications, particularly when working with small datasets or when computational efficiency is crucial.

This report explores several aspects of the classifier:

- The basic Naive Bayes algorithm and how it classifies data.
- The introduction of Laplace smoothing to address missing or unseen data in the training set.
- Log-transformation of probabilities to improve numerical stability during computation.
- Binarization of continuous features for use in the classifier.

II. Methods

A. Naive Bayes Classifier

The Naive Bayes classifier operates on the basis of prior and conditional probabilities for each feature. Given a new data instance, it calculates the posterior probability for each possible class using the formula:

$$P(C|X) \propto P(C) \prod_{i=1}^{n} P(X_i|C) \tag{1}$$

Where $P(C|X)$ is the posterior probability of class C given feature values X, $P(C)$ is the prior probability of class C, and $P(X_i|C)$ is the conditional probability of feature X_i given class C. The classifier assigns the data to the class with the highest posterior probability.

B. Laplace Smoothing

One of the limitations of Naive Bayes is that if a feature value does not appear in the training set for a given class, the conditional probability $P(X_i|C)$ becomes zero, which leads to the entire product becoming zero. To overcome this, "Laplace smoothing" is applied:

$$P(X_i|C) = \frac{n_{i,C} + \alpha}{n_C + \alpha \cdot v} \tag{2}$$

Where $n_{i,C}$ is the number of occurrences of feature X_i in class C, n_C is the total number of instances of class C, v is the number of possible values for X_i, and α is the smoothing parameter (typically set to 1). This ensures that no probability is zero.

C. Log Probabilities

When dealing with small probabilities, numerical underflow can occur due to multiplying many small numbers together. To avoid this, we take the logarithm of the probabilities, which allows us to sum the log-probabilities rather than multiplying them:

$$\log P(C|X) = \log P(C) + \sum_{i=1}^{n} \log P(X_i|C) \tag{3}$$

This approach maintains the same result while providing computational stability.

D. Binarization of Continuous Data

In the Iris dataset, the features are continuous. To apply the Naive Bayes classifier, we first "binarized" these continuous features. Binarization was done by calculating the mean of each feature and assigning 1 if the feature value was above the mean, and 0 otherwise.

III. Experiments

A. Weather Data

We used the weather dataset to train the Naive Bayes classifier. The dataset consists of categorical features. The data was randomly split into 10 training instances and 4 test instances. We compared the performance of the classifier with and without Laplace smoothing, as well as with log-transformed probabilities.

B. Iris Data

For the Iris dataset, which consists of continuous data, we applied binarization. Each feature (sepal length, sepal width, petal length, and petal width) was binarized based on whether the value was greater than the mean for that feature. After binarization, we applied the Naive Bayes classifier and evaluated its performance with log probabilities.

IV. Results

A. Weather Data: Naive Bayes without Smoothing

The Naive Bayes classifier without Laplace smoothing resulted in an error rate of **0.25** on the weather dataset.

B. Weather Data: Naive Bayes with Laplace Smoothing

When Laplace smoothing was applied, the error rate remained **0.25**. This suggests that the dataset did not suffer from missing or unseen values that would benefit from smoothing.

C. Weather Data: Naive Bayes with Log Probabilities

Using log probabilities, the error rate on the weather dataset remained at **0.25**, indicating that there were no numerical instability issues in this case.

D. Iris Data: Naive Bayes with Log Probabilities

For the Iris dataset, after binarization, the Naive Bayes classifier with log probabilities achieved an error rate of **0.17**. The reduction in error compared to the weather dataset may be due to the richer feature space provided by the Iris data.

V. Conclusions

In this report, we successfully implemented and evaluated the Naive Bayes classifier on two datasets: the weather dataset and the Iris dataset. Our experiments demonstrated the classifier's robustness and simplicity, even with strong assumptions of conditional independence between features.

For the weather dataset, applying Laplace smoothing did not change the error rate, indicating that the dataset did not contain any previously unseen feature values in the test set. This result underscores that while Laplace smoothing is crucial in many cases, its impact depends on the specific dataset. For the Iris dataset, the classifier performed better, achieving a lower error rate after binarizing the continuous features. This highlights the importance of discretization techniques when applying Naive Bayes to continuous data.

We also introduced log probabilities to prevent numerical underflow when multiplying small probabilities, but in this case, the effect on performance was minimal due to the manageable size of the dataset. However, the use of log probabilities ensures numerical stability, particularly in scenarios with large datasets or very small probabilities.

In future work, it would be valuable to explore different methods of handling continuous data, such as using probability density functions for continuous variables instead of binarization. Additionally, experimenting with more complex smoothing techniques or trying alternative classification algorithms could provide further insights into improving classification performance.

Overall, the Naive Bayes classifier, with the appropriate optimizations, proves to be an efficient and effective tool for classification tasks, especially with small or well-structured datasets.