Report

Homework 1

Goal: Develop models for 10-class classification problems with medium and large input space

Andrea Massignan
1796802

November 27, 2023

Project Overview

The goal is to provide two different solution for a 10-class classification problem.

1 Dataset

The datasets used in this project consists in training sets and blind test sets. The training sets are composed by 50000 samples, each one with 100 features for Dataset 1 and 1000 features for Dataset 2. The blind test sets are composed by 10000 samples, each one with 100 features for Dataset 1 and 1000 features for Dataset 2. The labels are 10, one for each class.

- `X_train`: 50000 samples for each dataset.
- `n_features`: 100 for Dataset 1 and 1000 for Dataset 2.

2 Data Exploration

The datasets, that are in a csv file, the elements $V^{(i)}$ belonging to the X column are feature vectors that represent the input data.

The Y column contains the associated labels C_k(k=0,...9).

Thus, each line i in column X is a feature vector structured as:

$[V_1^{(i)}$, $V_2^{(i)}$, ..., $V_j^{(i)}$, ... , $V_d^{(i)}]$

Where d is the size of the feature vector with sizes described before.

And $i = 1, ..., n$, where n is the number of samples in the dataset as specified before.

These are loaded with an helper function called `load_data` that returns the training set in the X matrix and the labels in the Y vector.

Then the training set is split into `X_train` and `Y_train` with a test size of 0.2 and different random states.

Since the dataset is already vectorized, no further preprocessing is needed at the moment.

3 Metrics

The metrics used to evaluate the models are the accuracy and the confusion matrix.

3.1 Precision

Precision is a measure of the accuracy of the positive predictions made by a model. It is the ratio of true positive predictions to the total number of positive predictions made by the model (both true positives and false positives). Precision is calculated using the formula:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Precision provides insight into the model's ability to avoid making false positive predictions. A high precision indicates that when the model predicts a positive class, it is likely to be correct.

3.2 Recall (Sensitivity or True Positive Rate)

Recall measures the ability of a model to correctly identify all relevant instances of the positive class. It is the ratio of true positive predictions to the total number of actual positive instances in the dataset (including both true positives and false negatives). Recall is calculated using the formula:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Recall is especially important when the cost of false negatives (missing a positive instance) is high, as it focuses on minimizing false negatives.

3.3 F1-Score

The F1-score is the harmonic mean of precision and recall. It provides a balanced measure that considers both false positives and false negatives. The F1-score is calculated using the formula:

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1-score is useful when there is an uneven class distribution or when false positives and false negatives have different consequences.

3.4 Support

Support refers to the number of actual instances of each class in the dataset. It is the count of true positive and true negative instances for each class. Support is not directly involved in the calculation of precision, recall, or F1-score, but it provides context by showing how many instances belong to each class.

Classification

4 Data Loading and Splitting

4.1 Loading Dataset 1

The data loading process begins with dataset 1, where the feature matrix X and label vector Y are extracted using the `load_data` function. An informative message is printed to indicate the commencement of this operation. The resulting shape of matrix X is then presented, providing insights into the number of samples and features within dataset 1.

4.2 Loading Dataset 2

An analogous procedure is employed for dataset 2. The feature matrix $X2$ and label vector $Y2$ are obtained through the `load_data` function. A corresponding message is printed, and the shape of $X2$ is displayed to convey the dataset's sample and feature dimensions.

4.3 Splitting Data

The subsequent steps involve splitting the loaded data into training and testing sets:

```
X_train1, X_test1, Y_train1, Y_test1 = train_test_split(X, Y, ...)
  ...
X_train4, X_test4, Y_train4, Y_test4 = train_test_split(X2, Y2, ...)
```

Utilizing the `train_test_split` function, both dataset 1 and dataset 2 are partitioned into training and testing sets. Each dataset undergoes this process twice, resulting in four sets per dataset. The `test_size` parameter dictates the proportion allocated for testing (20%), while `random_state` ensures reproducibility.

The code concludes by acknowledging the completion of data loading and splitting.

The resulting variables (X_train1, X_test1, Y_train1, Y_test1, etc.) are now ready for subsequent machine learning tasks.

Model Selection

After the data is loaded and splitted, the user is prompted to choose which of the dataset to use and then what type of model to use for the classification task. The user can choose between:

5 Bernoulli

BernoulliNB is a Naive Bayes classifier specifically tailored for multivariate Bernoulli models. Similar to MultinomialNB, this classifier is well-suited for handling discrete data.

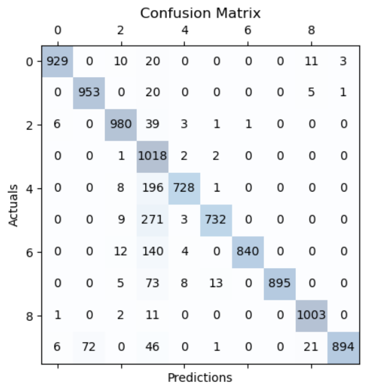
The key distinction lies in their data types: MultinomialNB deals with occurrence counts, while BernoulliNB is optimized for binary or boolean features. BernoulliNB is preferred over other Naive Bayes classifiers when dealing with datasets where features are binary or boolean.

It is well-suited for problems involving discrete, binary decision-making based on the presence or absence of features. Additionally, BernoulliNB is a simpler model with fewer parameters, making it advantageous for scenarios with limited training data and leading to faster training and prediction times.

5.1 Classification

Overall the model performs well, with an average accuracy of 90%, however class 3 has a lower recall of 71%, meaning that the model struggles to capture all instances of the class.

Even class 4 has a lower precision of 78%, meaning that when the model predicts this class it's not always correct.



	0		2		4		6		8	
0	929	0	10	20	0	0	0	0	11	3
	0	953	0	20	0	0	0	0	5	1
2	6	0	980	39	3	1	1	0	0	0
	0	0	1	1018	2	2	0	0	0	0
4	0	0	8	196	728	1	0	0	0	0
	0	0	9	271	3	732	0	0	0	0
6	0	0	12	140	4	0	840	0	0	0
	0	0	5	73	8	13	0	895	0	0
8	1	0	2	11	0	0	0	0	1003	0
	6	72	0	46	0	1	0	0	21	894

Figure 1: Confusion Matrix for BernoulliNB

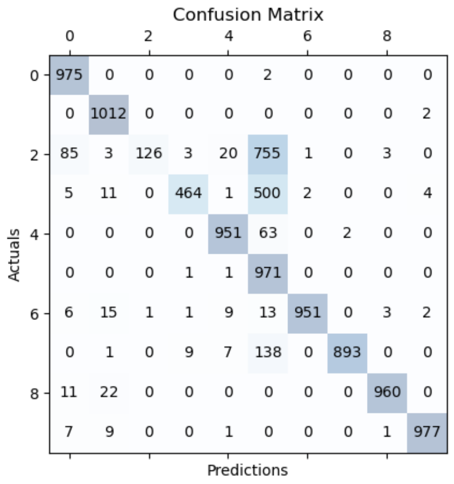
6 SVM

Support Vector Machine is a supervised machine learning algorithm used for classification and regression tasks. It's particularly powerful in high-dimensional spaces and is effective in cases where the margin between different classes is clear.

The SVM model performs well in terms of precision and recall for most classes, but there are variations across different digits.
In this particular case we are using a linear kernel, which is a good choice for high-dimensional data, however, it may not be the best choice for this particular dataset.

- The accuracy is around 83% which is lower than the BernoulliNB model.
- Some classes, like class 5, have lower support (973), while others, like class 7, have higher support (1048).
- The F1-scores are generally high, however, class 2 stands out with a lower F1-score (0.22).
- Notably, class 5 has a recall of 1.00, on the other hand, classes 2 and 3 have lower recall values.
- For most classes, precision is relatively high, however, there are exceptions, such as class 5, where precision is notably lower (0.40).

The lower performance in certain classes, especially in terms of recall for classes 2 and 3, indicates areas where the model may benefit from further improvement.



	0		2		4		6		8	
0	975	0	0	0	0	2	0	0	0	0
	0	1012	0	0	0	0	0	0	0	2
2	85	3	126	3	20	755	1	0	3	0
	5	11	0	464	1	500	2	0	0	4
4	0	0	0	0	951	63	0	2	0	0
	0	0	0	1	1	971	0	0	0	0
6	6	15	1	1	9	13	951	0	3	2
	0	1	0	9	7	138	0	893	0	0
8	11	22	0	0	0	0	0	0	960	0
	7	9	0	0	1	0	0	0	1	977

Figure 2: Confusion Matrix for SVM