# Mini-Project 1

CS771 - Introduction to Machine Learning

October 25, 2024

## Group No: 69

DCCLXXI

Team Members: Rythm Kumar(220925), Viral Chitlangia(221198), Arunkumar Duraisamy(220211)

Shreyash Dwivedi(221035), Subham Anand(221093)

**Abstract**

This project focuses on training binary classification models on three different datasets that differ in their feature representations. We aim to identify the best models for each dataset based on accuracy and training data size, and also investigate whether combining these datasets can yield a more accurate model. Performance is evaluated on validation accuracy across different training data sizes.

# Contents

**1 Introduction** **3**

**2 Task 1: Individual Dataset Models** **3**
2.1 Task Description . . . 3
2.2 Dataset 1: Emoticons as Features . . . 3
2.2.1 Description of Dataset . . . 3
2.2.2 Analysis of Data . . . 3
2.2.3 Models and Their Respective Accuracy . . . 3
2.3 Dataset 2: Deep Features . . . 4
2.3.1 Description of Dataset . . . 4
2.3.2 Analysis of Data . . . 4
2.3.3 Models and Their Respective Accuracy . . . 4
2.4 Dataset 3: Text Sequence Dataset . . . 6
2.4.1 Description of Dataset . . . 6
2.4.2 Analysis of Data . . . 6
2.4.3 Models and Their Respective Accuracy . . . 7

**3 Task 2: Combined Dataset Model** **8**
3.1 Task Description . . . 8
3.2 Combination of Data and Results . . . 8

**4 Conclusion** **8**

# 1 Introduction

In this project, we tackle a binary classification problem using three datasets, each derived from the same underlying data but with different feature representations. The first dataset, referred to as **Dataset-1**, represents features as emoticons. **Dataset-2** contains deep neural network-extracted features, and **Dataset-3** consists of numeric text sequences. The project is divided into two tasks -

In **Task 1**, we focus on training and evaluating a variety of binary classification models, each tailored to the specific feature representation of the respective dataset. The goal is to identify the best-performing model for each dataset based on validation accuracy and use these models to generate predictions on the test set.

In **Task 2**, we aim to enhance our models by combining the three datasets, leveraging their different feature representations to improve overall model performance. We then compare the accuracy of this combined model with the individual models from Task 1 to assess the impact of dataset integration on prediction quality.

# 2 Task 1: Individual Dataset Models

## 2.1 Task Description

The objective of **Task 1** is to develop a binary classification model for each dataset separately, aiming for high accuracy with minimal training data usage. We will evaluate the models based on their validation accuracy, using different percentages of training data.

## 2.2 Dataset 1: Emoticons as Features

### 2.2.1 Description of Dataset

Dataset-1 comprises 7,080 examples, each represented by a sequence of 13 emoticons. These emoticons serve as categorical features, with 214 unique emoticons identified across the dataset. Each instance is associated with a binary classification label (0 or 1), with approximately 50% of the examples labeled as 0 and the remaining 50% labeled as 1, ensuring a balanced dataset.

### 2.2.2 Analysis of Data

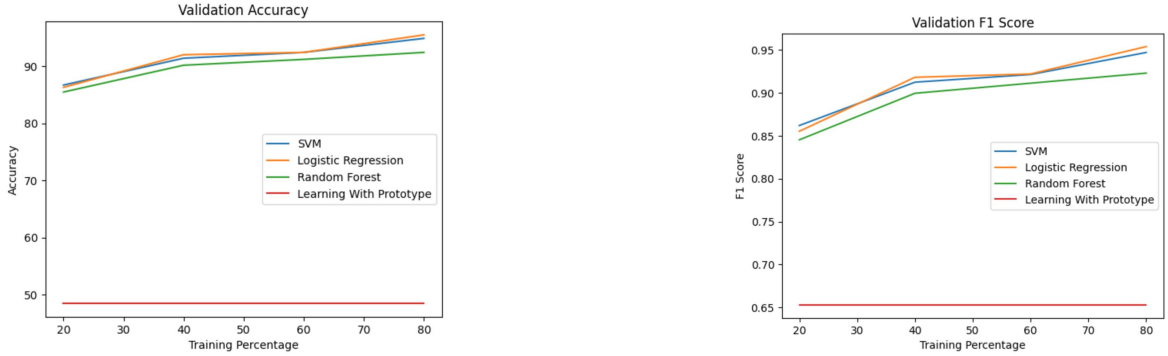
We proceeded to find the frequencies of all unique emoticons for each position over all 7,080 input sequences. It was found that a certain set of 213 emoticons appeared in positions 1-8, while another set of 91 emoticons appeared in positions 9-13. Moreover, it was observed that the first 8 positions always contained the following set of 6 emoticons: ['😜','😑','🛜','😢','🍸','🚾'] while the last 5 positions always contained this set of 4 emoticons: ['😑','😢','🍸','🔪'], in different permutations.

### 2.2.3 Models and Their Respective Accuracy

The consistent presence of these 7 specific emoticons across all training examples led us to infer that they did not contribute meaningfully to the classification task. These emoticons were deemed superfluous, as their distribution appeared independent of the labels. Consequently, we removed these 7 emoticons from the original set of 214 unique emoticons, reducing the feature set to 207 unique emoticons.

The remaining 207 emoticons were then encoded using a one-hot encoding scheme, converting the categorical emoticons into binary feature vectors. Given that the task at hand is a binary classification problem, this encoding provided a suitable input representation for our machine learning models.
For Dataset 1, Logistic Regression achieved the best performance, reaching 94.07% accuracy and an F1 score of 0.9395 with 80% of the training data, while maintaining strong performance (90.18% accuracy) even with just 40% of the data. SVM closely followed, attaining 93.66% accuracy and an F1 score of 0.9345 with 80% of the data. Random Forest showed consistent performance, peaking at

92.23% accuracy with 80% of the data. Both Logistic Regression and SVM demonstrated significant improvements as training data increased. In contrast, the LwP model performed poorly, maintaining a constant 48.47% accuracy across all data splits.




(a) Accuracy scores (b) F1 scores

Figure 1: Validation Scores on various models

We selected Logistic Regression as the model for making the final prediction for this dataset training it on 100% of the data since validation accuracy improved as the training data increased. After completing the training, the model produced weights of length 624.

## 2.3 Dataset 2: Deep Features

### 2.3.1 Description of Dataset

The **Deep Features Dataset** contains input features extracted using a simple deep neural network. Each input is represented as a $13 \times 786$ matrix, where:

1. Each of the 13 rows corresponds to a distinct emoticon feature.
2. Each row contains a 786-dimensional embedding representing the deep features of the emoticon.

The embeddings for each emoticon were obtained through a deep feature extraction process, where each input's emoticon-based features were transformed into high-dimensional vectors. The objective is to convert this matrix representation into a suitable feature vector for further processing. The dataset is provided in the `.npz` format, with the following structure:

1. **features:** a list of $13 \times 786$ floating-point matrices.
2. **label:** a binary label (0 or 1) indicating the class.

Example: **features:** [[0.1, -0.3, 0.5, ...], ...] ,**label:** "0"

### 2.3.2 Analysis of Data

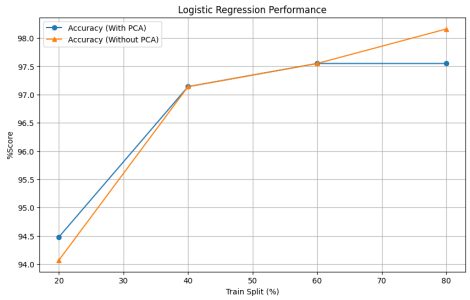
The dataset consists of 7,080 training examples, each represented as a $13 \times 786$-dimensional matrix. Upon closer analysis, it was discovered that the feature vectors at positions $\{2, 3, 4, 5, 6, 9, 10, 11, 12, 13\}$ were redundant across both the training and validation sets. These redundant vectors did not provide any meaningful contribution to model performance, suggesting that they do not capture significant variation in the data. To streamline the input representation and reduce computational complexity, these redundant vectors were removed. As a result, each input matrix has been reduced from $13 \times 786$ to a more compact $3 \times 786$-dimensional matrix, retaining only the essential vectors at positions 1, 7, 8 for further processing.

### 2.3.3 Models and Their Respective Accuracy

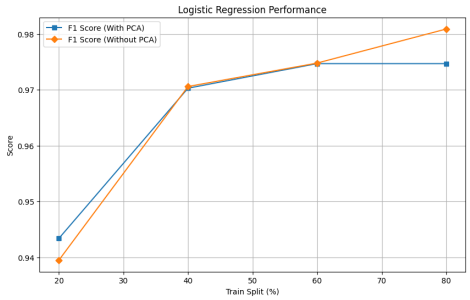
We applied standardized scaling on our dataset that scales it to mean zero and standard deviation of 1 . We also used PCA, i.e., Principal Component Analysis, which simplifies the dataset by transforming its features into new variables, capturing the most important patterns, and helps in reducing complexity while keeping the essential information.

- **Method 1** - We applied PCA on the 3 x 786 input . We flattened the input matrix into a 3*786 dimensional vector and then applied PCA on it which reduces the dimensions.
- **Method 2** (Without PCA)- We directly flattened it into a 3*786 dimensional vector and applied our models.

Graphical representation of the results obtained on the validation set provided from training various machine learning models on different training dataset splits are as follows:

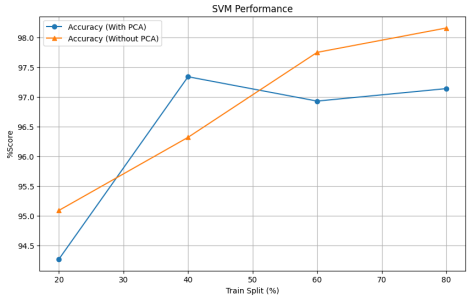



(a) Accuracy scores

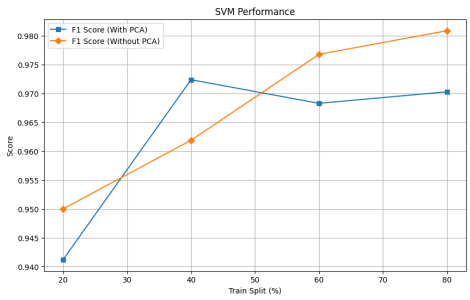



(b) F1 scores

Figure 2: Validation scores of Logistic Regression

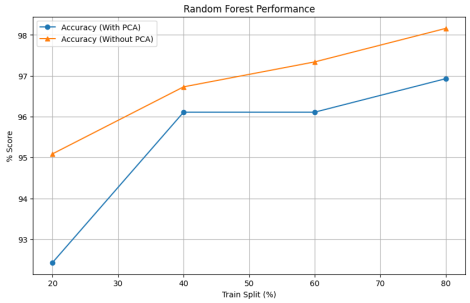



(a) Accuracy scores

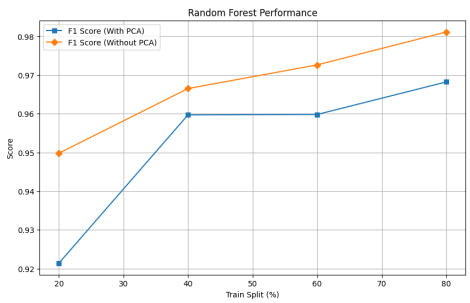



(b) F1 scores

Figure 3: Validation scores of SVM




(a) Accuracy scores




(b) F1 scores

Figure 4: Validation scores of Random Forest

For Dataset 2, the best performance was achieved with Logistic Regression, where using 80% of the training data without PCA resulted in an accuracy of 98.16% and an F1 score of 0.9809. Even with just 40% of the data, it reached 97.14% accuracy. SVM followed closely, achieving 97.14% accuracy and 0.9809 F1 score without PCA on 80% of the data. Random Forest performed well too, reaching 96.11% accuracy with 40% of the data and 96.93% accuracy with 80% of the data.

Overall, Logistic Regression provided the best results, particularly without PCA, suggesting that the full feature set from the deep embeddings was highly effective for this classification task.

We selected the Logistic Regression model and trained it on 80% of the data, resulting in 2,305 training weights, which is well below the trainable weights limit of 10,000 for the final predictions.

## 2.4 Dataset 3: Text Sequence Dataset

### 2.4.1 Description of Dataset

Dataset-3 consists of input sequences in the form of a character string of length 50, where each character represents a numeric digit (0-9). These sequences contain only numeric digits as the input features. Each sequence is associated with a binary classification label (0 or 1), and the dataset remains balanced in terms of label distribution. This data has `.csv` format unlike the Dataset-2. The data in this dataset corresponds to the same underlying information as the Emoticons and Deep Features datasets but is encoded as numeric sequences rather than emoticons or embeddings.

### 2.4.2 Analysis of Data

We began by analyzing the frequency of digits (0-9) across all 50 positions in the input sequences. A frequency table was generated to observe the distribution of digits at each position. From this analysis, we immediately identified that the first three positions contained only the digit '0' across all examples, rendering these positions redundant for model training. As a result, we removed these positions from the dataset.

Next, we analyzed the occurrence of digits at various positions based on the assigned binary labels (0 or 1).

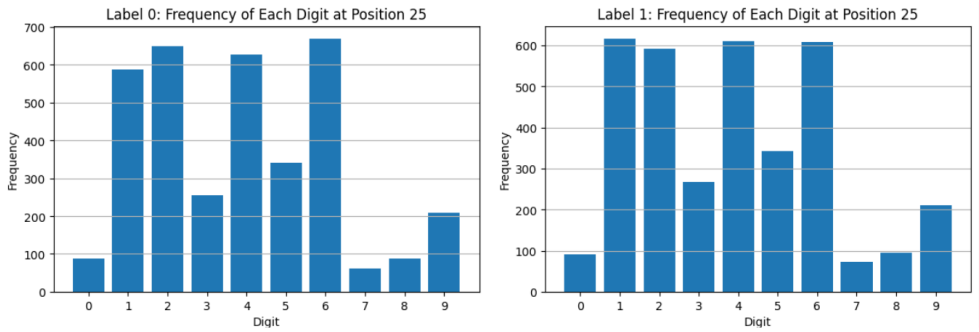



Figure 5: Frequencies of each digit for position 25

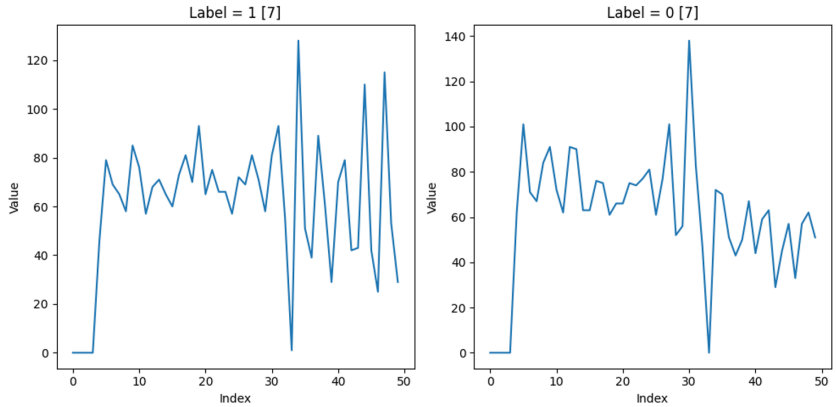



Figure 6: Frequency Graph of Digit 7 for each position

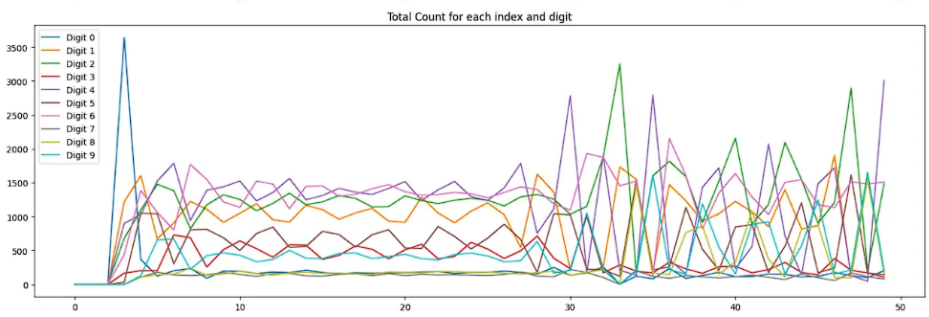



Figure 7: Frequencies of each digit over position in the input string

To explore potential patterns, we plotted the frequency of each digit for both labels across all positions. This helped us investigate whether certain digits might be more strongly associated with a particular label, and whether specific positions could favor certain digits depending on the label. Note that the variation in frequencies is less in positions 1-30, while it is significant in positions 31-50 for all digits. The frequencies corresponding to digits were plotted for each position, for each label, and it was found that the plots were similar for both labels for all positions (Such a bar graph for position 25 is included as an example). Similarly, line graphs of frequencies corresponding to position were plotted for each digit, for each label. Here, all were similar except for that of digit '7', which appeared comparatively lesser than the other digits.

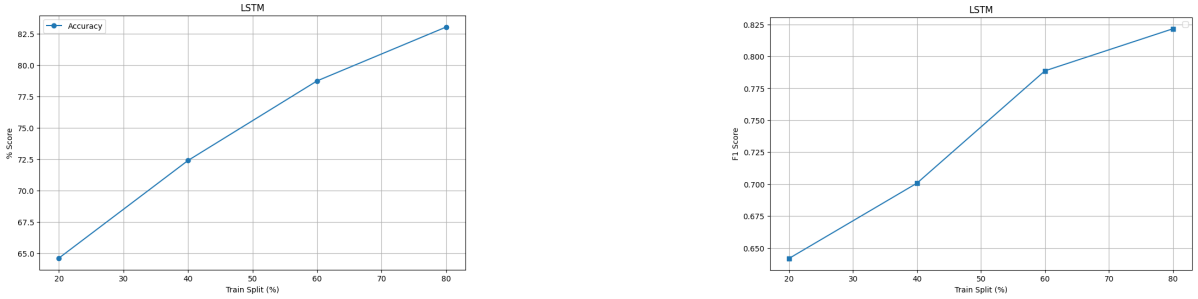
### 2.4.3 Models and Their Respective Accuracy

As described earlier, we began by removing the first three characters from each input sequence, as these positions were deemed redundant. We then applied one-hot encoding to the remaining digits, transforming the sequences into a structured format suitable for model training.

We initially employed Logistic Regression, SVM, and Random Forest due to their simplicity and effectiveness with structured data. However, these models assume that features are independent and do not capture the sequential dependencies between the digits. As a result, their performance was limited, achieving only 60-65% accuracy on the dataset.

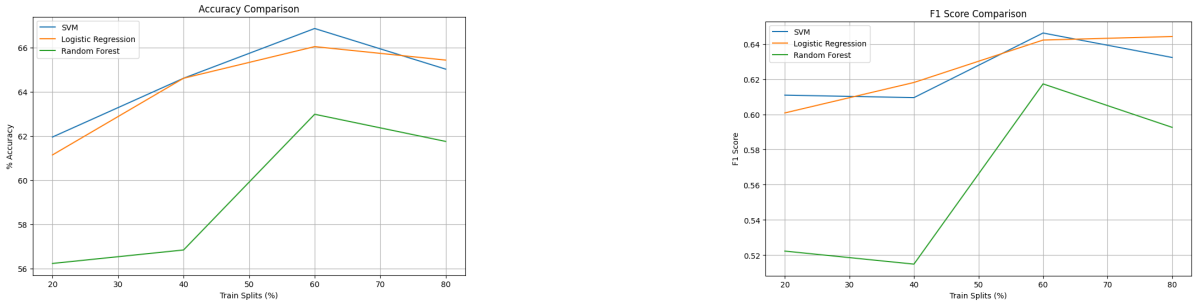
To address this, we implemented a Long Short-Term Memory (LSTM) model, a type of Recurrent Neural Network (RNN) which is specifically designed to handle sequential data by retaining information from earlier positions in the sequence. After training on 80% of the data, the LSTM model significantly outperformed the traditional approaches, achieving an accuracy of 83%. This improvement highlights the LSTM's ability to capture causal dependencies, making it more effective for this classification task.

The LSTM model, trained on 80% of the data, produced 8,769 training weights, staying below the limit of 10,000 weights.




(a) Accuracy scores (b) F1 scores

Figure 8: Validation scores of LSTM




(a) Accuracy scores (b) F1 scores

Figure 9: Validation scores on Other Models

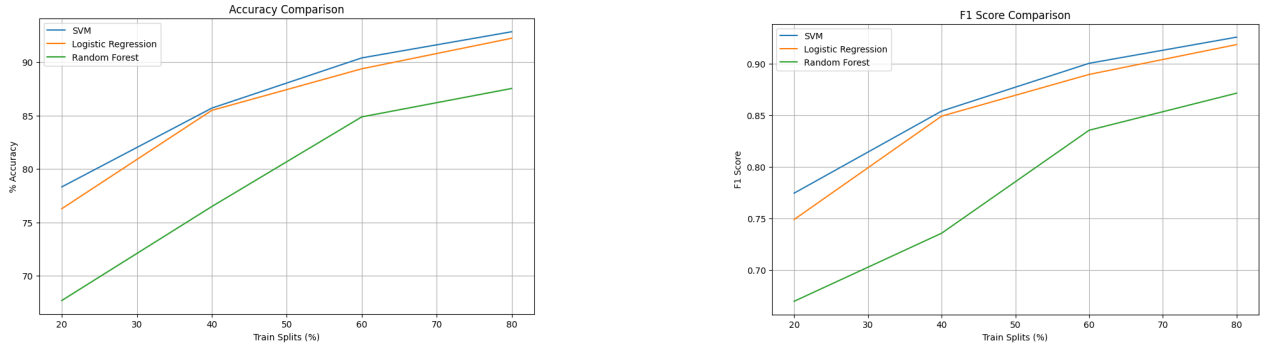
# 3 Task 2: Combined Dataset Model

## 3.1 Task Description

Task 2 explores whether combining the feature representations of the three datasets leads to a better-performing classification model. The goal is to assess the generalization of this model on the validation set.

## 3.2 Combination of Data and Results

In Dataset 3, the first three positions of each input sequence were removed because they contained only zeros and were considered redundant. At the same time, in Dataset 1, after removing the redundant emoticons, each example was left with exactly three emoticons. This prompted us to combine the two datasets by replacing the first three positions in Dataset 3 with the corresponding three emoticons from Dataset 1. By integrating these emoticons back into the sequence, we enriched the feature representation and prepared the combined dataset for model training. We chose to go with the following models: logistic regression, SVM and random forest, from which SVM gave the best performance of 93% when trained on 80% of the training data while Logistic Regression gave its best of 92% at 80% training data and Random forest gave a maximum accuracy of 87% when trained on 80% of the training dataset.




(a) Accuracy scores (b) F1 scores

Figure 10: Validation scores of Various Models

# 4 Conclusion

This mini-project explored binary classification models across three distinct datasets, each representing the same underlying data with different feature representations. Our analysis yielded several important insights:

**Dataset Performance:** Among the individual datasets, Dataset 2 (Deep Features) consistently outperformed the others. Using Logistic Regression, we achieved a remarkable 97% accuracy while training on only 40% of the data, demonstrating the effectiveness of the deep feature representations.

**Model Efficiency:** For Dataset 1 (Emoticons as Features), Logistic Regression reached 94.07% accuracy with 80% of the training data. Dataset 2 also demonstrated efficiency with 97% accuracy on 40% of the data, while Dataset 3 (Text Sequence) required an LSTM model, achieving 83% accuracy.

**Feature Engineering:** Our analysis revealed the significance of feature selection and preprocessing. Removing redundant features, such as consistently appearing emoticons or digits, improved model performance across all datasets.

**Combined Dataset Approach:** Contrary to our initial hypothesis, combining datasets did not yield superior results. The combined model achieved a maximum accuracy of 93% with SVM on 80% of the data, which fell short of Dataset 2's individual performance.

**Optimal Model:** The most effective model was from Dataset 2, utilizing Logistic Regression for an optimal balance of 97% accuracy with 40% of the training data.

**Model Complexity:** All models maintained fewer than 10,000 trainable parameters, achieving high performance with limited complexity.