CS771 - Intro to ML (Autumn 2024): Mini-project 1

October 22, 2024

Sagili Harshitha - 220934 Sarath Kumar - 220754 Bhanu Prakash - 220583
Sree Sahithi - 220577 Dinesh Naik - 220866

---

1 Introduction

1.1 Problem Statement

In this project we are provided 3 binary classification datasets. Each of these datasets represents the same machine learning task and was generated from the same raw dataset. The 3 datasets only differ in terms of features being used to represent each input from the original raw dataset. Each of these 3 datasets further consists of a training set, a validation set, and a test set. The validation set can be used for selecting the best hyperparameters or any other analyses you may want to perform. The problem statement is Divided into 2 parts Task-1 and Task-2

1.1.1 Task-1

The first task is to train some binary classification models on each of the 3 datasets and pick the best possible binary classification model for each of the 3 datasets. Once identifying best models for each of the 3 datasets, have to predict the labels for the corresponding test sets using those models.

1.1.2 Task-2

After completing task-1, we have to combine the train 3 datasets together. The second task is to train some binary classification models on the combined dataset (3 datasets that given in task-1) and pick the best possible binary classification model. Once identifying best models for combined dataset, have to predict the labels for the corresponding test sets using that model.

1.2 Notation

In this report for our convenience we are splitting Task-1 into 3-tasks i.e.,

- **Task-1(a):** To identify best model for **Emoticons as Features Dataset**.
- **Task-1(b):** To identify best model for **Deep Features Dataset**.
- **Task-1(c):** To identify best model for **Text Sequence Dataset**.
- **Task-2:** To identify best model for **Combined Dataset**.

1.3 Aim

Basically we have to identify best binary classification model for the 3 models and also for the combination of the 3 models. Best model should associate with these properties:

- **Better Accuracy:** The model should demonstrate higher accuracy on the validation set.
- **Lower Amount of Training Data:** We experimented with varying the number of training examples by using the first {20%, 40%, 60%, 80%, 100%} of the training examples.

2 Dataset Description

The three datasets provided for this project contain the same inputs but differ in the way the features are represented. Below is a brief description of each dataset:

2.1 Emoticons as Features Dataset

This dataset represents each input as a sequence of 13 emoticons.

- **Example:** `input_emoticons: "weary, cry, smile, thinking, expressionless, smirk, angry, grinning, thumbsup, bike", label: "0"`
- **Label:** {0, 1}

2.2 Deep Features Dataset

In this dataset, each input is represented by deep neural network embeddings of size 13 $\times$ 786 matrix.

- **Example:** `features: [0.1, -0.3, 0.5, ...], label: "0"`
- **Label:** {0, 1}

2.3 Text Sequence Dataset

This dataset represents each input as a sequence of 50 digits, which provides a more complex sequence-based feature set compared to the other datasets.

- **Example:** `input_str: "02418214121265492664 61...", label: "0"`
- **Label:** {0, 1}

3 Analysis

3.1 Task-1(a)

3.1.1 Models Using One-Hot Encoding

Each unique emoticon was assigned an index, and a one-hot vector was created for every emoticon. This resulted in a 13 $\times$ 226 matrix for each input. The one-hot encoded features were used to train the following models:

(a) Logistic Regression:
Logistic Regression achieved a final validation accuracy of 88.96% using 80% of the training data, and 89.37% using 100% of the training data.

(b) K-Nearest Neighbors (KNN):
KNN achieved a very low validation accuracy of 59.30% using 100% of the training data.

(c) Least Weighted Prototypes (LWP):
LWP achieved a final validation accuracy of 82.41% using 100% of the training data.

(d) Support Vector Machine (SVM):

SVM with linear kernel achieved 88.34% validation accuracy using 100% of training data.

Note: Deep neural network is not used with this features because even adding 4 neurons in next layer exceed the limit of 10k trainable parameters.

3.1.2 Model Using Embeddings

Each emoticon in the dataset was label-encoded, assigning a unique integer value from 1 to 226, reflecting the total of 226 distinct emoticon types. As a result, the original emoticon sequences were transformed into numpy arrays, forming vectors of length 13, corresponding to the sequence length.

(a) Logistic Regression, LWP, KNN, SVM:

Logistic, KNN, LWP, and SVM models did not outperform those trained on one-hot encoded features.

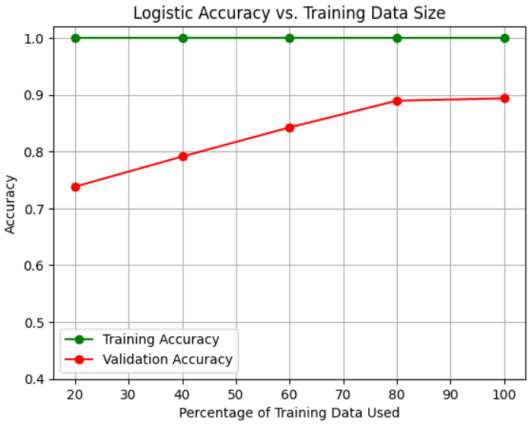
(b) Deep Neural Network (DNN):

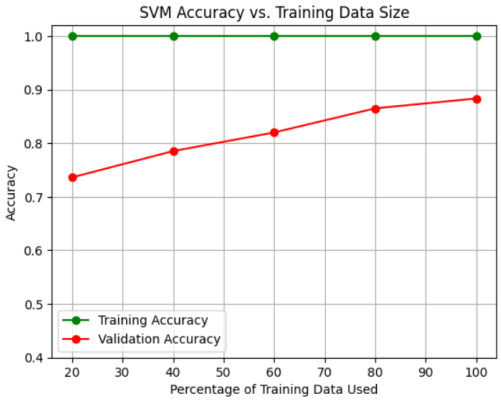
The DNN is a sequential neural network specifically designed for this dataset. It begins with an **Embedding layer** that converts the integer-encoded emoticons into dense 12-dimensional vectors, resulting in an output tensor of shape (None, 13, 12). The **Flatten layer** reshapes this output into a one-dimensional array of size 156, enabling compatibility with the subsequent dense layers. The first **Dense layer** consists of 24 neurons and employs the ReLU activation function, facilitating the learning of complex feature representations through non-linear transformations. The final **Output layer** contains a single neuron with a sigmoid activation function, generating a probability score for binary classification. The model is compiled with the Adam optimizer and uses binary crossentropy as the loss function, totaling 6517 trainable parameters. This structure effectively enables the model to learn from the training data, achieving high accuracy of 98.6% on the validation set.

Table 1: Training and Validation Accuracies Across Different Models

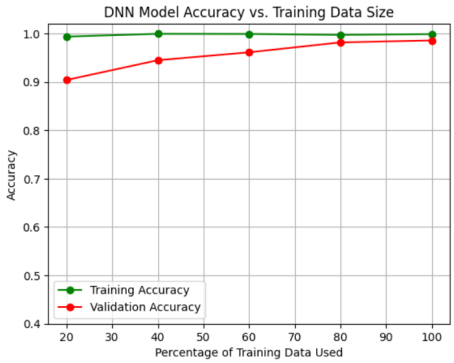
Model	Feature Extraction Method	Training %	Training Accuracy	Validation Accuracy
Logistic Regression	One-Hot Encoding	60%	100%	84.25%
		80%	100%	88.96%
		100%	100%	89.37%
SVM	One-Hot Encoding	60%	100%	82.00%
		80%	100%	86.50%
		100%	100%	88.34%
KNN	One-Hot Encoding	100%	72.22%	59.30%
LWP	One-Hot Encoding	100%	92.70%	82.41%
Deep Neural Network (DNN)	Embedding Layer	20%	99.36%	90.39%
		40%	99.93%	94.48%
		60%	99.90%	96.11%
		80%	99.72%	98.16%
		100%	99.87%	98.57%

Best Models Using One-Hot Encoding





Best Model Using Embeddings



3.1.3 Final Model:

Our analysis showed that the DNN model designed for emoticon sequence classification performed excellently, achieving a high validation accuracy of 98.6% on 100% of the training data with 6517 trainable parameters. By leveraging label-encoded inputs and an embedding layer, the model efficiently captures relationships between different emoticons.

3.2 Task-1(b)

3.2.1 Models using flattened feature set

Each 13×786 matrix is flattened into a vector of size 9984.

(a) Logistic Regression:
Logistic Regression achieved a final validation accuracy of 98.56% using 100% of the training data but with very high number of trainable parameters 9985.

(b) K-Nearest Neighbors (KNN):
KNN with 5 neighbors achieved a final validation accuracy of 86.91% using 100% of the training data.

(c) Least Weighted Prototypes (LWP):
LWP achieved a final validation accuracy of 93.66% using 100% of the training data.

(d) Support Vector Machine (SVM):
SVM achieved a final validation accuracy of 98.56% using 100% of the training data but with very high number of trainable parameters 9985.

3.2.2 Models using PCA feature extraction

Principal Component Analysis (PCA) reduces the 9984-dimensional flattened vector to 150 components, capturing the most important variance in the data while removing redundant or less significant features and reducing computational costs. The PCA-extracted features were used to train the following models:

(a) Logistic Regression:
Logistic Regression achieved a final validation accuracy of 96.73% using 40% of the training data, and 98.77% using 100% of the training data.

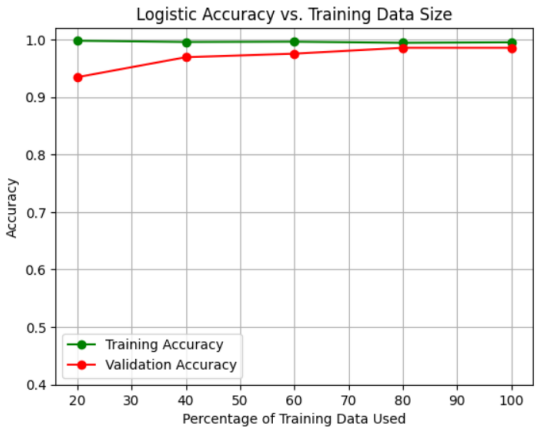
(b) Support vector machine:
SVM performed slightly better with PCA-transformed features than with flattened features. Achieving validation accuracy of 98.77% using 100% of the training data. However, **logistic regression** remains the **more practical choice** due to its simplicity and faster training times compared to **SVM**.

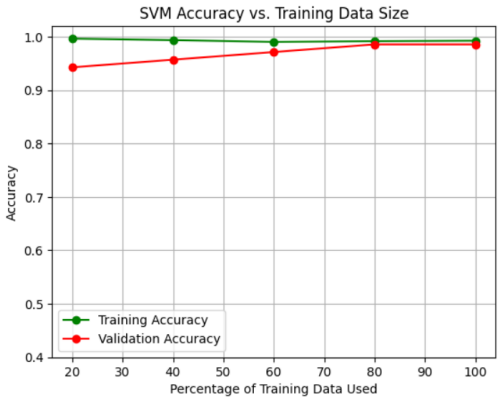
(c) Deep Neural Network (DNN):
DNN achieved high accuracy with PCA-reduced features. Achieving validation accuracy of 98.16% using 80% of the training data. However, **logistic regression** remains the **more practical choice** due to its simplicity and faster training times.

Table 2: Training and Validation Accuracies Across Different Models

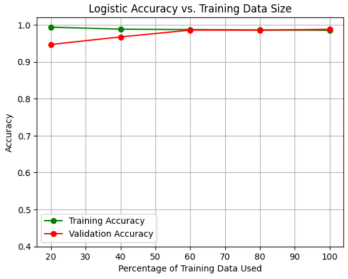
Model	Feature Extraction Method	Training %	Training Accuracy	Validation Accuracy
Logistic Regression	Flattening	80%	99.43%	98.56%
		100%	99.51%	98.56%
LWP	Flattening	100%	95.36%	93.66%
SVM	Flattening	100%	99.25%	98.56%
Logistic Regression	PCA (150)	60%	98.70%	98.57%
		80%	98.60%	98.57%
		100%	98.53%	98.77%
SVM	PCA (150)	100%	98.30%	98.77%
DNN	PCA (150)	60%	99.97%	97.34%
		80%	100%	98.16%
		100%	99.97%	97.75%

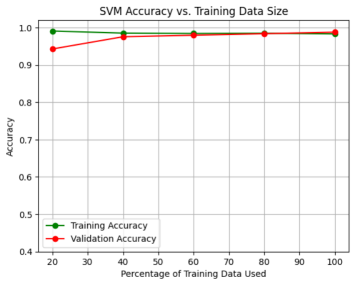
Best models using flattened feature set

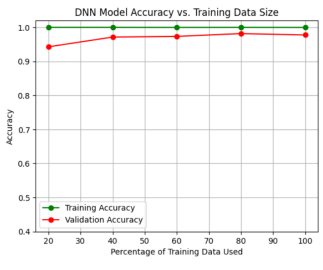




Best models using PCA feature extraction







3.2.3 Final Model:
Our analysis showed that the Logistic Regression model using PCA-reduced feature sets for Deep Features Dataset classification performed excellently, achieving a high validation accuracy of 98.77% on 100% of the training data with 151 trainable parameters.

3.3 Task-1(c)

3.3.1 Models using Single-Character One-Hot Encoding

In this dataset, the inputs are given as text sequences where the characters are from '0' to '9'. The single-character one-hot encoding method transforms each character in the sequence (digits 0-9) into a binary vector of length 10. Each sequence has a fixed length of 50, resulting in a 50×10 matrix. These matrices are then flattened into 500-dimensional vectors for the following models, except for the DNN.

(a) Logistic Regression:
Logistic regression achieves 64.8% validation accuracy at maximum with 100% of the training data.

(b) K-Nearest Neighbors (KNN):
Using 5 nearest neighbors works better than other KNN configurations, but the model only achieves a validation accuracy of 62%.

(c) Least Weighted Prototypes (LWP):
Accuracies are similar to KNN, with no significant improvement.

(d) Support Vector Machine (SVM):
This model performs the best among the non-DNN models trained so far, giving a maximum validation accuracy of 65.64% on 100% of the training set, with a training accuracy of 72.5%.

(e) Deep Neural Network (DNN):
The DNN model, using a CNN architecture with Conv1D layers and fewer parameters, reaching a validation accuracy up-to 78.12%. This significantly outperforms the other models, indicating that a deep learning approach is more effective for this problem.

3.3.2 Models using Overlapping Two-Character Encoding

In this dataset, the inputs are given as text sequences where the characters are from '0' to '9'. The double-character one-hot encoding method transforms the sequence to one-hots by taking two digits at a time (digits ranging from 0-99) into a binary vector of length 100. Each sequence has a fixed length of 50, resulting in a (50-1)×100 matrix. These matrices are then flattened into 500-dimensional vectors for the following models, except for the DNN.

(a) Logistic Regression:
Here, logistic regression performs better than the previous four basic models, achieving a validation accuracy of 74%, which gives us positive hope.

(b) K-Nearest Neighbors (KNN) and LWP:
These models' performances surprisingly decreased.

(c) Support Vector Machine (SVM):
This model also works similarly to logistic regression in terms of accuracies, achieving around 74.23% on 80% and 100% training sets.

(d) Deep Neural Network (DNN):
The Deep Neural Network (DNN) employs a CNN architecture to classify digit sequences. It consists of input reshaping to a 3D format, followed by two convolutional layers (16 filters each) with ReLU activation, max pooling, and dropout layers to mitigate overfitting. After flattening, it includes a dense layer with 16 neurons and a final sigmoid output layer for binary classification, with a total of 6,849 trainable

parameters. The model achieved a training accuracy of 95.64%, validation accuracy of 92.43%, and an F1 score of 0.9221, demonstrating its effectiveness in capturing patterns within the data.

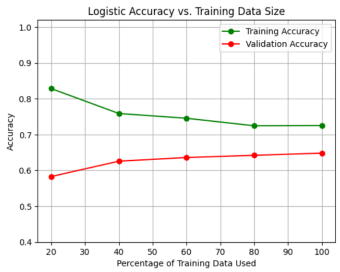
3.3.3 Embeddings-Based Feature Extraction

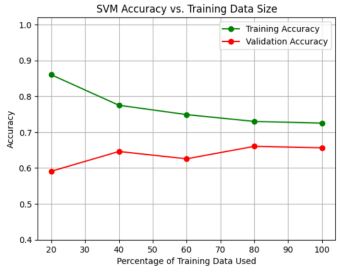
Similar to what we did in task-1(a) for emoticons, we are transforming the characters into labels (this can be done directly by changing characters to digits). In this case, we also trained all five models: logistic regression, KNN, LWP, SVM, and DNN. With embeddings as features, the DNN gives the highest training and validation accuracies of 76.8% and 66.87%, respectively, not even surpassing simple logistic regression on overlapping one-hot features, making these features less significant in this classification problem.

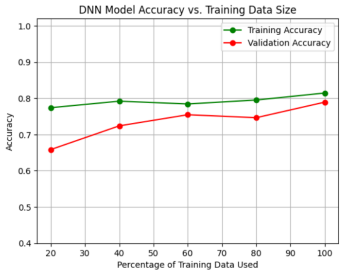
Table 3: Training and Validation Accuracies Across Different Models

Model	Feature Extraction Method	Training %	Training Accuracy	Validation Accuracy
Logistic Regression	Single Char One-Hot	80%	72.45%	64.21%
		100%	72.51%	64.82%
KNN	Single Char One-Hot	100%	75.16%	61.96%
SVM	Single Char One-Hot	80%	73.00%	66.05%
		100%	72.52%	65.64%
DNN	Single Char One-Hot	80%	79.51%	74.64%
		100%	81.45%	78.93%
Logistic Regression	Two-Char One-Hot	80%	96.02%	72.80%
		100%	94.74%	74.02%
DNN	Two-Char One-Hot	60%	95.64%	88.34%
		80%	95.25%	89.57%
		100%	94.53%	92.02%
DNN	Embedding	80%	76.39%	66.87%
		100%	76.39%	66.87%

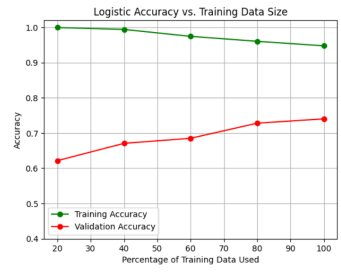
Best Models Using Single Character One-Hot Encoding

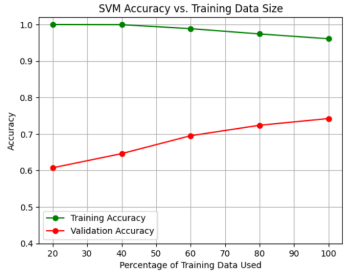


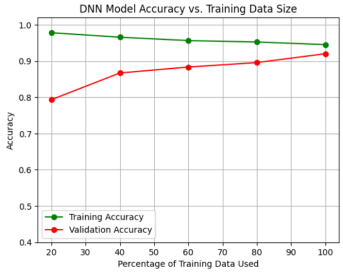




Best Models Using Overlapping Two-Character Encoding







3.3.4 Final Model:

The 1D Convolutional Neural Network (CNN) model using overlapping two character encoding feature extraction achieved a training accuracy of 95.64% and a validation accuracy of 92.43%, indicating its effectiveness in classifying sequences from the text dataset. With an F1 score of 0.9221, the model balances precision and recall well. Its lightweight architecture, featuring only 6,849 trainable parameters, ensures efficient computation while capturing complex patterns in the data. Overall, this model proves to be a reliable choice for accurate sequence classification tasks.

3.4 Task-2

3.4.1 Feature extraction

Emoticon Features

we utilize a flattened one-hot matrix of size 13×226 for each input, which has demonstrated high accuracy with simple models.

Deep Features

we apply Principal Component Analysis (PCA) on the flattened matrix (with dimensions 13×768) derived from the 9984 features, retaining the top 150 most variable features to reduce dimensionality.

Text Sequence Features

Text sequence features consist of a 50-digit numeric string. Overlapping character pairs are used to form a 49x100 one-hot encoded matrix, which is flattened to a vector of size 4900.

Combined Features

By combining these, we obtain a vector of size (2938 + 150 + 4900), totaling 7988 features. However, these high-dimensional features were verified with various sample features through PCA, showing no major change in accuracy. Therefore, we are proceeding with only 300 features after applying PCA. Here are a few models we have trained:

(a) Logistic Regression:

This basic model, with just 301 trainable parameters, achieved a validation accuracy of 98.77% and a training accuracy of 98.67%, indicating no overfitting. The model consistently delivered 90%+ validation accuracies across all percentages of training datasets. Given these points, we can select this model for our final predictions. The model effectively captured the underlying patterns of the sequences while maintaining a low computational cost, allowing for quick training and evaluation. Its simplicity makes it easy to interpret, providing insights into the importance of individual features. This makes Logistic Regression a strong baseline for further experimentation.

(b) K-Nearest Neighbors (KNN):

Delivered a validation accuracy of approximately 70% across various hyperparameter values for k, indicating difficulties in classifying high-dimensional data. This method may have suffered due to the curse of dimensionality, which often affects KNN performance.

(c) Least Weighted Prototypes (LWP):

LWP with Euclidean distance also performed well with these 300 features, reaching

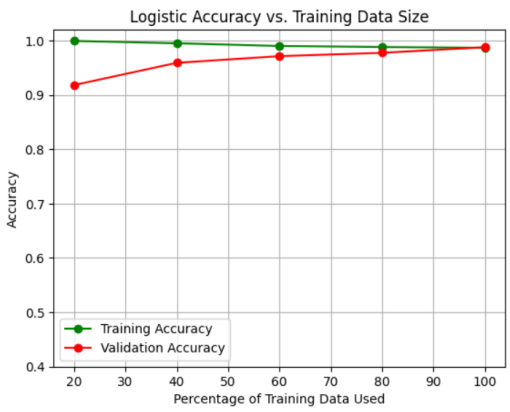
validation and training accuracies of up to 95%. However, it did not surpass the previously trained Logistic Regression model, providing a good balance between performance and model complexity.

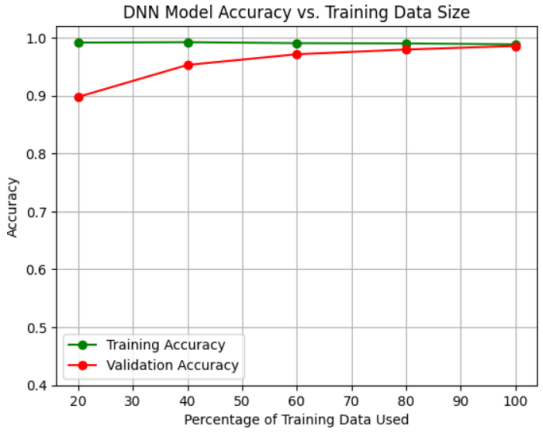
(d) Support Vector Machine (SVM):
Achieved validation accuracies of up to 98% and training accuracies of 99%. This model is computationally costlier than the previous ones but demonstrated a similar accuracy trend to the Logistic Regression model, albeit slightly lower. SVM effectively manages the complexity of the dataset and demonstrates its capability to handle non-linear relationships.

(d) 1-Dimensional DNN:
Utilized a CNN architecture, yielding a validation accuracy of 97.96% and an F1 score of 0.9791. This indicates strong performance in recognizing complex patterns within the data. The DNN architecture allowed for the extraction of hierarchical features, enabling the model to learn from both local and global patterns in the sequences. The model also exhibited robustness to noise, which is critical in real-world applications. Furthermore, its ability to learn non-linear relationships provided a significant advantage over traditional models, enhancing classification accuracy.

Table 4: Training and Validation Accuracies Across Different Models

Model	Feature Extraction Method	Training %	Training Accuracy	Validation Accuracy
Logistic Regression	PCA + Concatenation	40%	99.50%	95.91%
		60%	99.01%	97.13%
		80%	98.81%	97.75%
		100%	98.67%	98.77%
KNN	PCA + Concatenation	100%	79.46%	67.89%
LWP	PCA + Concatenationt	80%	94.57%	95.09%
		100%	94.33%	94.47%
SVM	PCA + Concatenation	60%	98.82%	97.54%
		80%	98.83%	97.95%
		100%	98.77%	97.34%
DNN	PCA + Concatenation	60%	99.10%	97.34%
		80%	98.99%	98.16%
		100%	98.92%	98.16%





3.4.2 Final Model:

The final model, Logistic Regression, demonstrated exceptional performance with a validation accuracy of 98.77% using 100% of the training data and only 301 trainable pa-

rameters. Its simplicity, efficiency, and ability to capture underlying patterns make it an ideal choice for accurate predictions in this dataset.

4 Final Models:

Table 5: Training and Validation Accuracies for the Final Models of Each Task

Task	Model	Feature Extraction Method	Training %	Training Accuracy	Validation Accuracy
Task-1(a)	Deep Neural Network (DNN)	Deep Neural Network (DNN)	40%	99.93%	94.48%
			60%	99.90%	96.11%
			80%	99.72%	98.16%
			100%	99.87%	98.57%
Task-1(b)	Logistic Regression	PCA (150)	60%	98.70%	98.57%
			80%	98.60%	98.57%
			100%	98.53%	98.77%
Task-1(c)	DNN	Two-Char One-Hot	60%	95.64%	88.34%
			80%	95.25%	89.57%
			100%	94.53%	92.02%
Task-2	Logistic Regression	PCA + Concatenation	40%	99.50%	95.91%
			60%	99.01%	97.13%
			80%	98.81%	97.75%
			100%	98.67%	98.77%

5 References:

- For Sklearn: https://scikit-learn.org/stable/index.html
- For Tensorflow: https://www.tensorflow.org/ and TensorFlow YouTube Playlist
- For Python and its libraries: https://www.w3schools.com/python/
- To know more about feature extractions we used: Feature Extraction Paper
- For various deep learning neural networks: Introduction to Deep Neural Networks

Acknowledgement

We would like to express our sincere gratitude to our professor, Dr. Piyush Rai, for his invaluable support and guidance throughout this mini-project. His expertise and encouragement were instrumental in our exploration of the three binary classification datasets. We appreciate the opportunity to delve into various machine learning techniques and enhance our collaborative skills. This experience has enriched our understanding and has been a significant part of our academic journey. Thank you, Dr. Rai, for your mentorship and for fostering a conducive learning environment.