EED363 – APPLIED MACHINE LEARNING (SP2021)

Final Project Report

Handwritten Digit Recognition

Submitted by

Team Member 1: **Rithish Jakkireddy**

1810110188

Rj976

Team Member 2: **Rohith Kambampati**

1710110168

Ka517

Under supervision of

Prof Madan Gopal

Electrical Engineering

Department of Electrical Engineering

School of Engineering

SHIV NADAR UNIVERSITY

Contents

Abstract

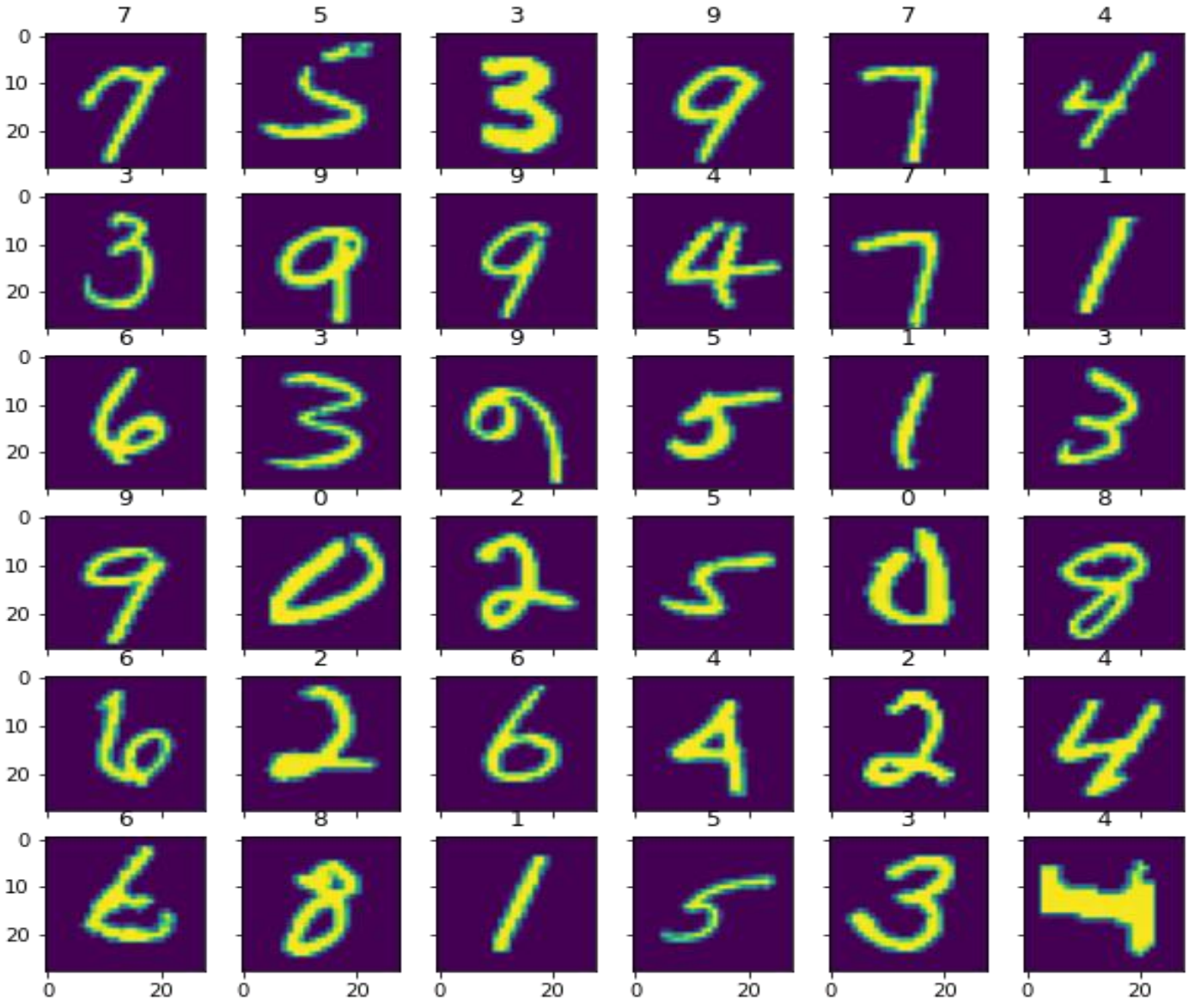
The main objective of this project is to provide efficient and reliable techniques for recognition of handwritten numerals by comparing various existing classification models. In this project, multiple learning techniques are examined, and a new accuracy level for recognition of the MNIST dataset is reported. The proposed framework involves pre-processing, Exploratory Data Analysis (EDA), feature extraction and classification. As will be seen, pre-processing and feature extraction play crucial roles in this experiment to reach the highest accuracy. The classification task is performed by a few classifiers namely, Support Vector Machine (SVM), K-Nearest Neighbors (K-NN) and Random Forest (RF) to determine which classifier has the highest accuracy rate in this experiment. Moreover, few experimental results are analysed and evaluated by a series of tools such as the confusion matrix, error rates. Python libraries like sklearn, seaborn, pytorch, numpy, kears, tensorflow are being used.

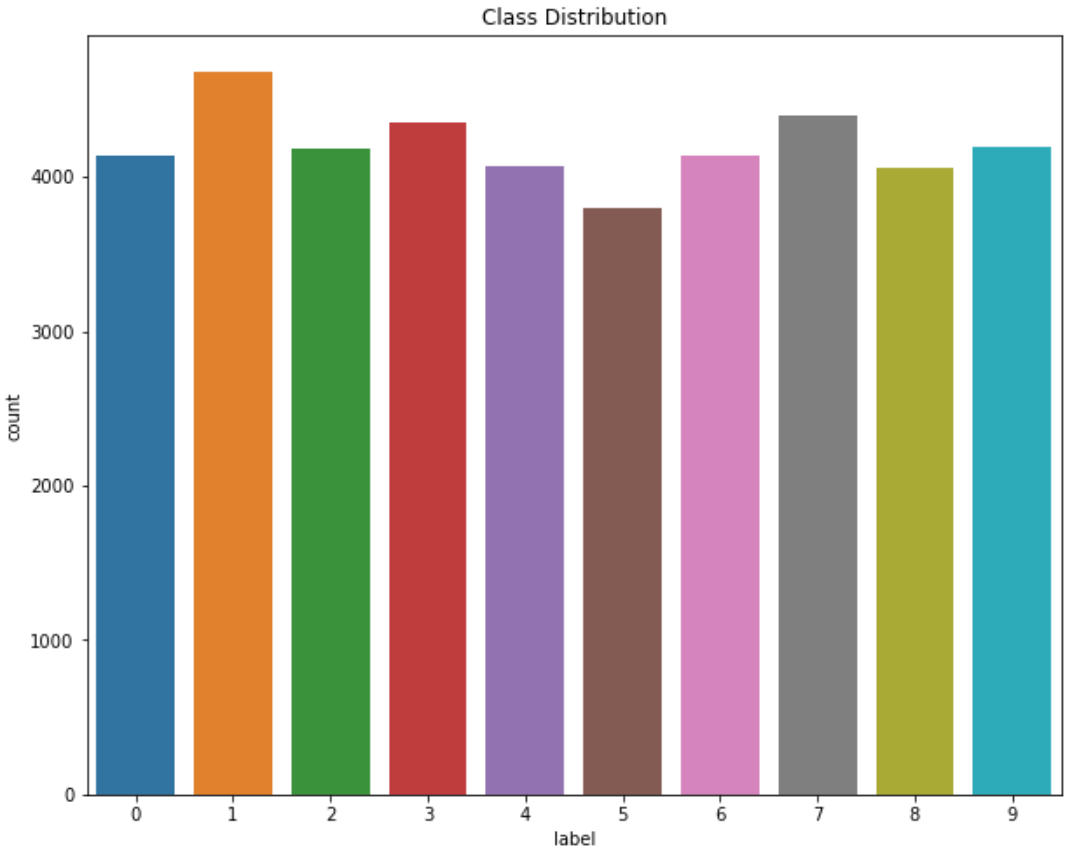
Introduction

The problem of handwritten numerals recognition has. been widely studied in recent years and the large quantity of pre-processing methods and classification algorithms have been developed. However, handwritten numerals recognition is still a challenge for us. The main difficulty of handwritten numerals recognition is highly corelated to image size, translation, stroke thickness, rotation, and deformation of the numeral image because of handwritten digits are written by different users and their writing style is different from one user to another user. The problem of handwritten digit recognition has long been an open problem in the field of pattern classification. The main objective of this project is to provide efficient and reliable techniques for recognition of handwritten numerals by comparing various existing classification models. This study focuses on feature extraction and classification. The performance of a classifier can rely as much on the quality of the features as on the classifier itself.

Description of Dataset

In this project, we used the MNIST database consisting handwritten digits ranging from 0-9. The MNIST dataset is an acronym that stands for the Modified National Institute of Standards and Technology dataset. The task is to classify a given image of a handwritten digit into one of 10 classes representing integer values from 0 to 9, inclusively. It is a widely used and deeply understood dataset and, for the most part, is “solved.” Top-performing models are deep learning convolutional neural networks that achieve a classification accuracy of above 99%, with an error rate between 0.4 %and 0.2% on the hold out test dataset. The total number of digit image samples (62,000), the total number for training (42000) and testing (10,000), and the subtotal number for each digit are shown in the bar graph below. Each digit is a grey-level fixed-size image with a size of 28 x 28 (or 784 pixels) in total as the features.





EDA (Exploratory data analysis):

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations. It's a method of visualizing, summarizing, and analysing data that's hidden behind rows and columns. EDA is a critical step in data science because it helps us to gain specific information and statistical measurements that are critical for business continuity, stockholders, and data scientists. It performs to define and refine our important features variable selection, that will be used in our model. In EDA for our dataset, we will first identify various parameters for all the features such as their probability distribution, mean, variance and their correlation with the output. Also, we will be identifying and removing the outliners in each feature using boxplot and identify and remove null values from the data set if any.

```
df_train.isnull().any().describe()
```

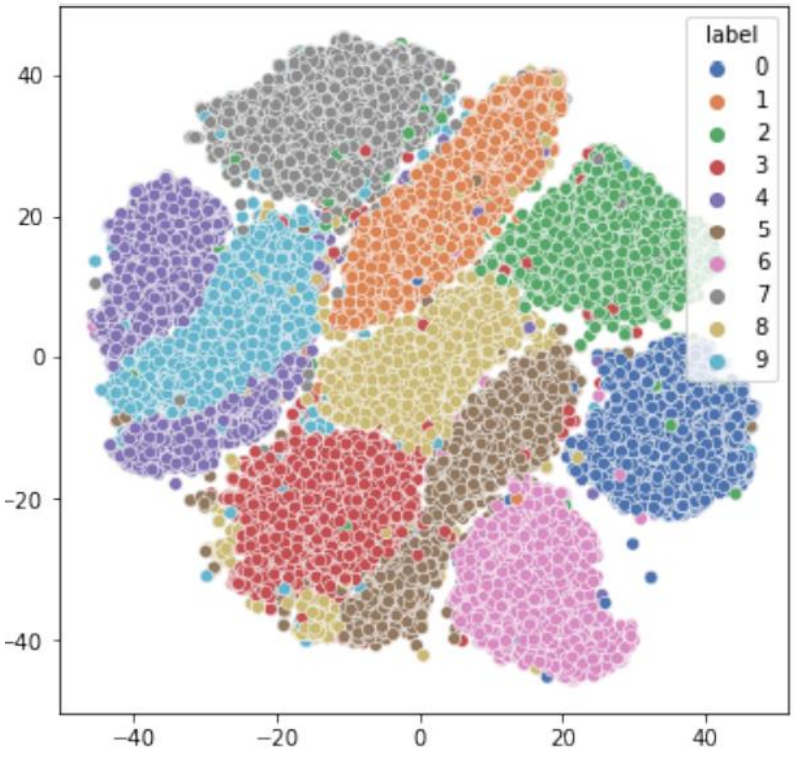
```
count       785
unique        1
top       False
freq        785
dtype: object
```

```
df_test.isnull().any().describe()
```

```
count       784
unique        1
top       False
freq        784
dtype: object
```

T - distributed Stochastic Neighbor Embedding.

t-SNE is a tool to visualize high-dimensional data. It converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data. t-SNE has a cost function that is not convex, i.e., with different initializations we can get different results. t-Distributed stochastic neighbor embedding (t-SNE) minimizes the divergence between two distributions: a distribution that measures pairwise similarities of the input objects and a distribution that measures pairwise similarities of the corresponding low-dimensional points in the embedding. From the graph we can see the two components hold some information, especially for specific digits, but clearly not enough to set all of them apart. Luckily there is another technique that we can use to reduce the number of dimensions that may prove more helpful. The method we will be exploring is known as t-SNE (t-Distributed Stochastic Neighbouring Entities).

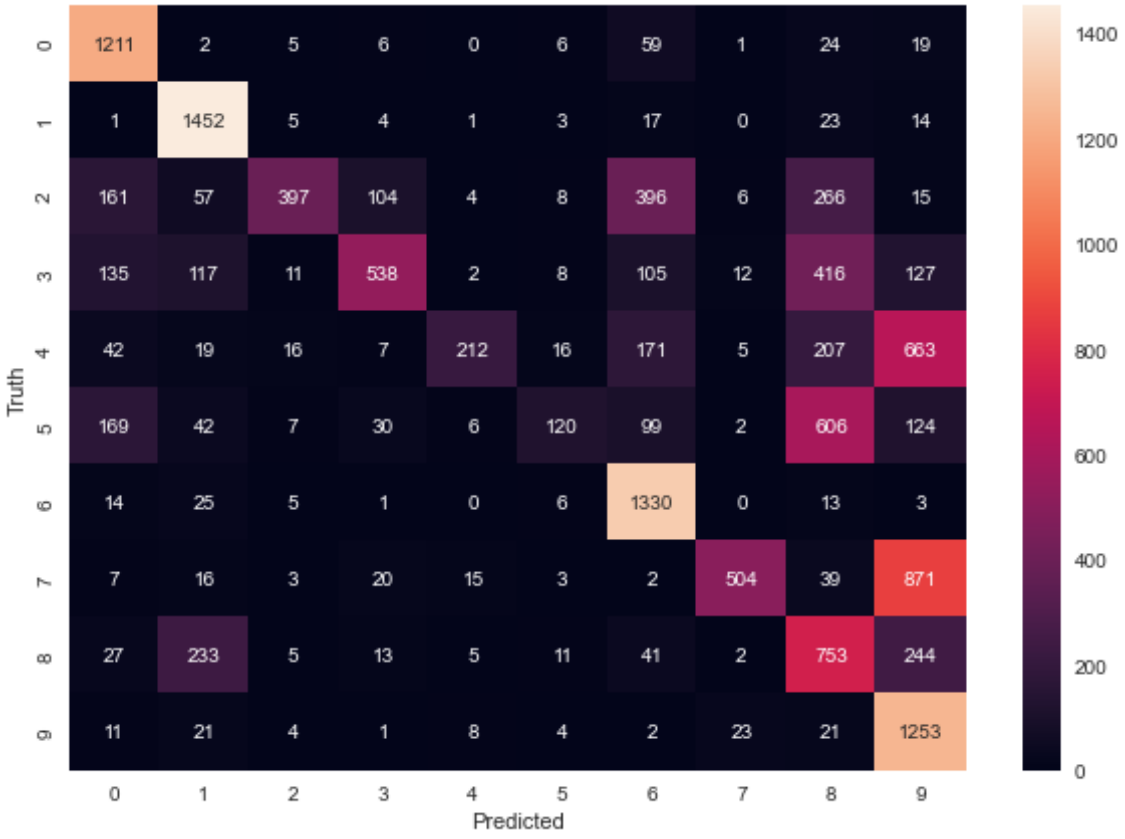


Work Done and Results:

- **Naïve Bayes:** Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, accurate and reliable algorithm. Naive Bayes classifiers have high accuracy and speed on large datasets.

```
Accuracy: 56.060606060606055
```

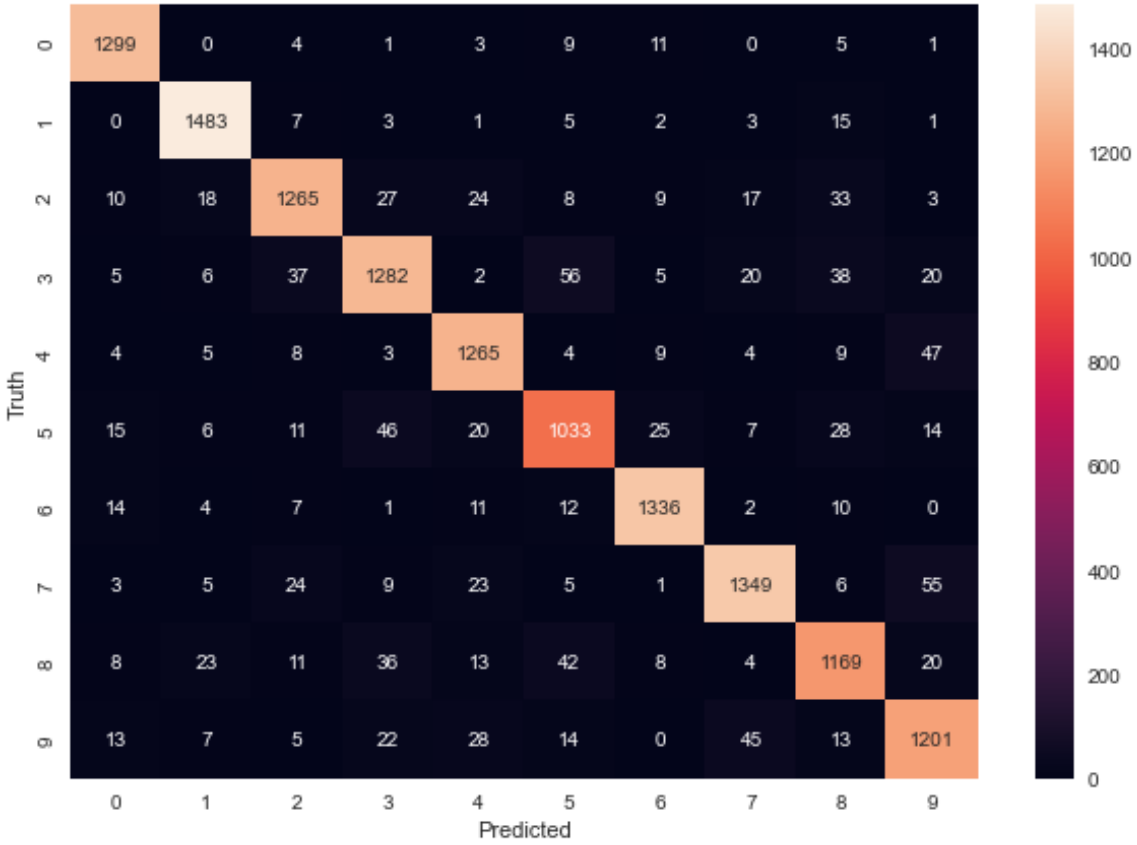
Confusion Matrix:



- **Logistic Regression:** Logistic regression may be a supervised learning classification algorithm want to predict the probability of a target variable. It's one among the only ML algorithms which will be used for various classification problems like spam detection, Diabetes prediction, cancer detection etc. Logistic regression is simpler to implement, interpret, and efficient to coach.

```
Accuracy: 0.9658008658008658
```

Confusion Matrix for Logistic Regression:



- **KNN (K nearest neighbors):** KNN is the non-parametric method or classifier used for classification as well as regression problems. This is the lazy or late learning classification algorithm where all of the computations are derived until the last stage of classification, as well as this, is the instance-based learning algorithms where the approximation takes place locally. Being simplest and easiest to implement there is no explicit training phase earlier and the algorithm does not perform any generalization of training data. KNN explains categorical value using majority votes of K nearest neighbors where the value for K can differ, so on changing the value of K, the value of votes can also vary.

Algorithm

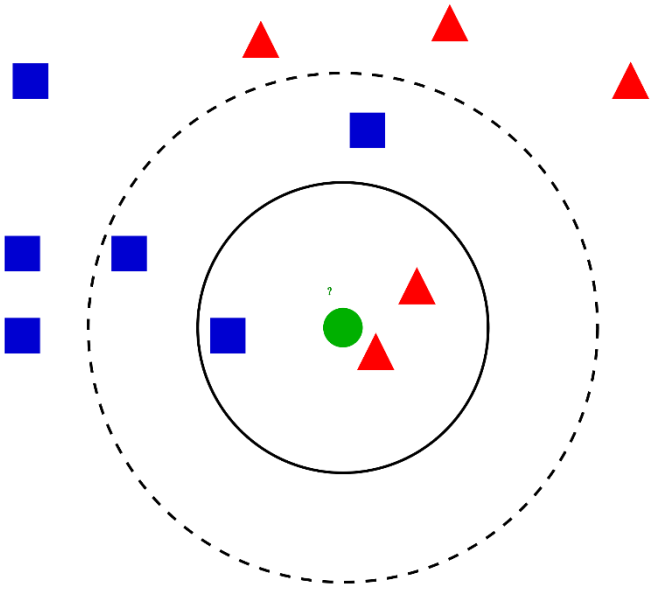
- Compute the distance metric between the test data point and all labelled data points.
- Order the labelled data points in increasing order of distance metric.
- Select the top K labelled data points and look at class labels.
- Look for the class labels that majority of these K labelled data points have and assign it to test data points.

Distance functions

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

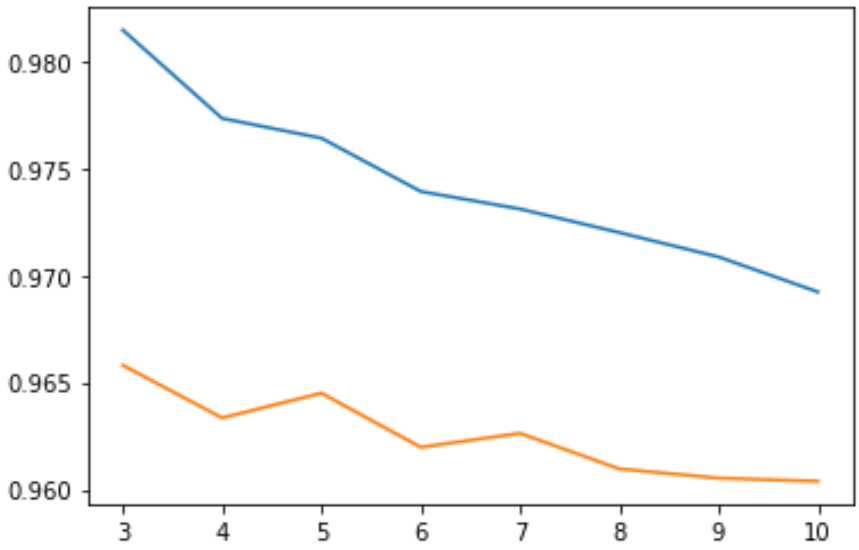
Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$



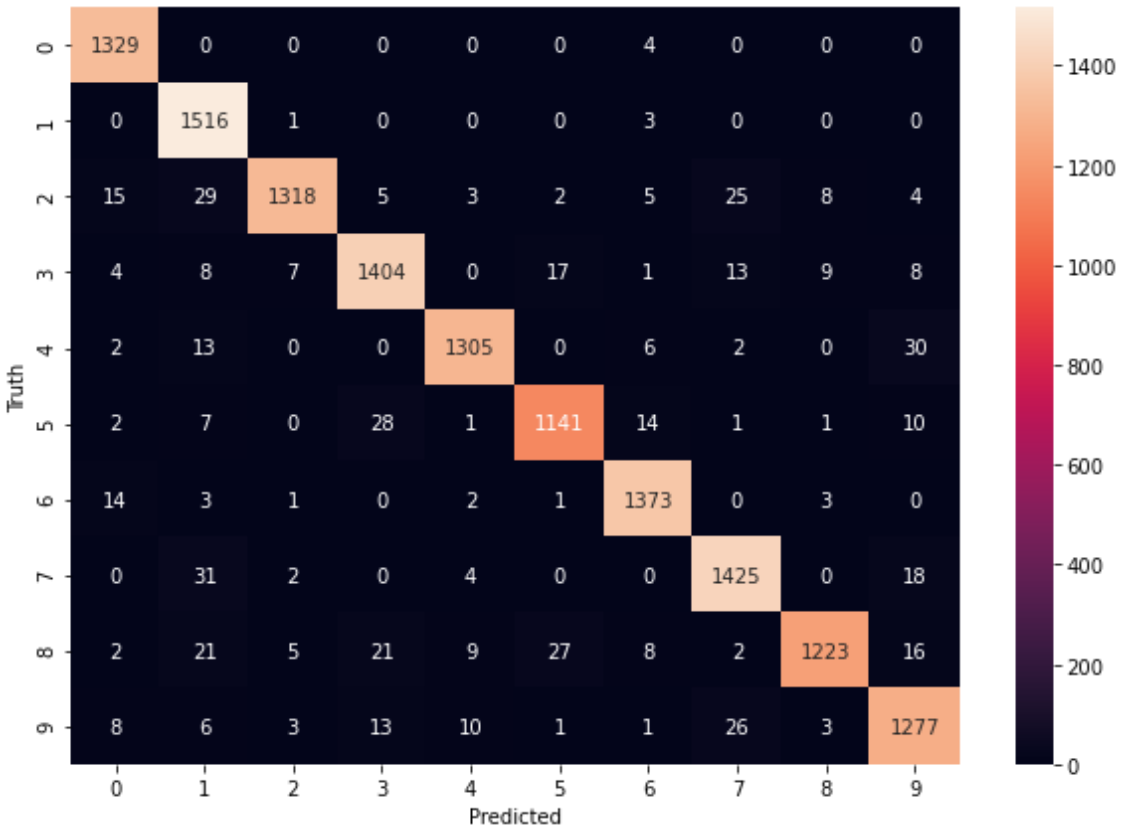
For changing the value of K, the output for the test data point can also vary. So, it's necessary to choose the value of K wisely. The large value for K can reduce the overall noise but there is no guarantee.

The k-Nearest Neighbors algorithm is a simple and effective way to classify data. But the algorithm has to carry around the full dataset; for large datasets, this implies a large amount of storage. In addition, you need to calculate the distance measurement for every piece of data in the database, and this can be cumbersome.

```
Accuracy: 96.58008658008658
```



Confusion Matrix for KNN:



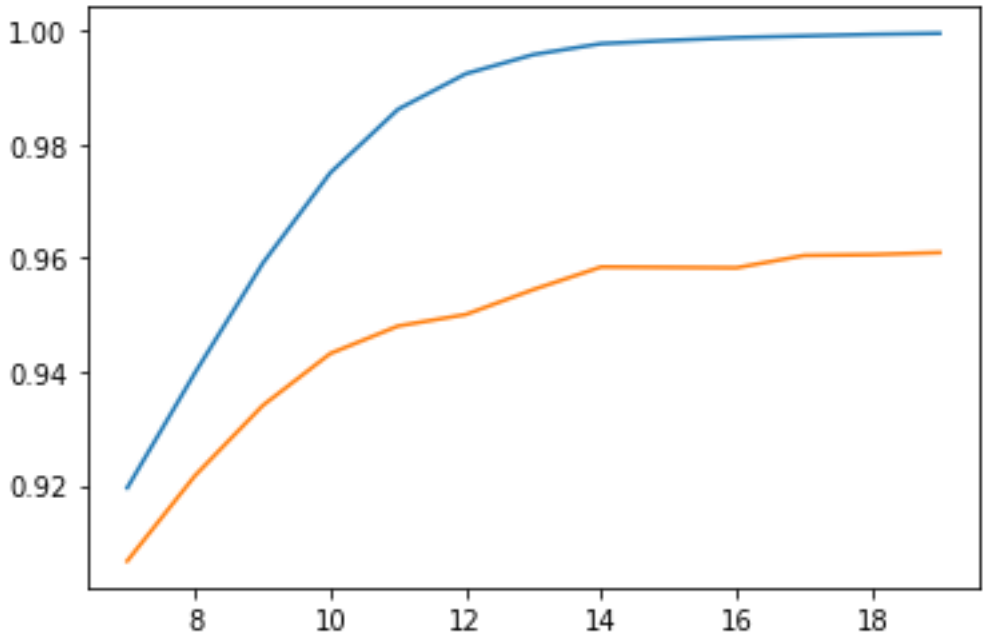
- **Random Forest:** Actually, Random Forest is a general term for classifier combination that uses L tree-structured classifiers {h (x, Θk), k = 1, ...L} where Θk are independent identically distributed random vectors and x is an input. With respect to this definition, one can say that Random Forest is a family of methods in which we can find several algorithms, such as the Forest-RI algorithm proposed by Breiman in and cited as the reference method in all RF related papers. In Forest-RI algorithm, Bagging is used in tandem with a random feature selection principle. The training stage of this method consists in building multiple trees, each one trained on a bootstrap sample of the original training set i.e., the Bagging principle – and with a CART-like induction algorithm [4]. This tree induction method, sometimes called RamdomTree, is a CART-based algorithm that modifies the feature selection procedure at each node of the tree, by introducing a random pre-selection i.e., the RandomSubspace principle.

  Each tree is grown as follows:

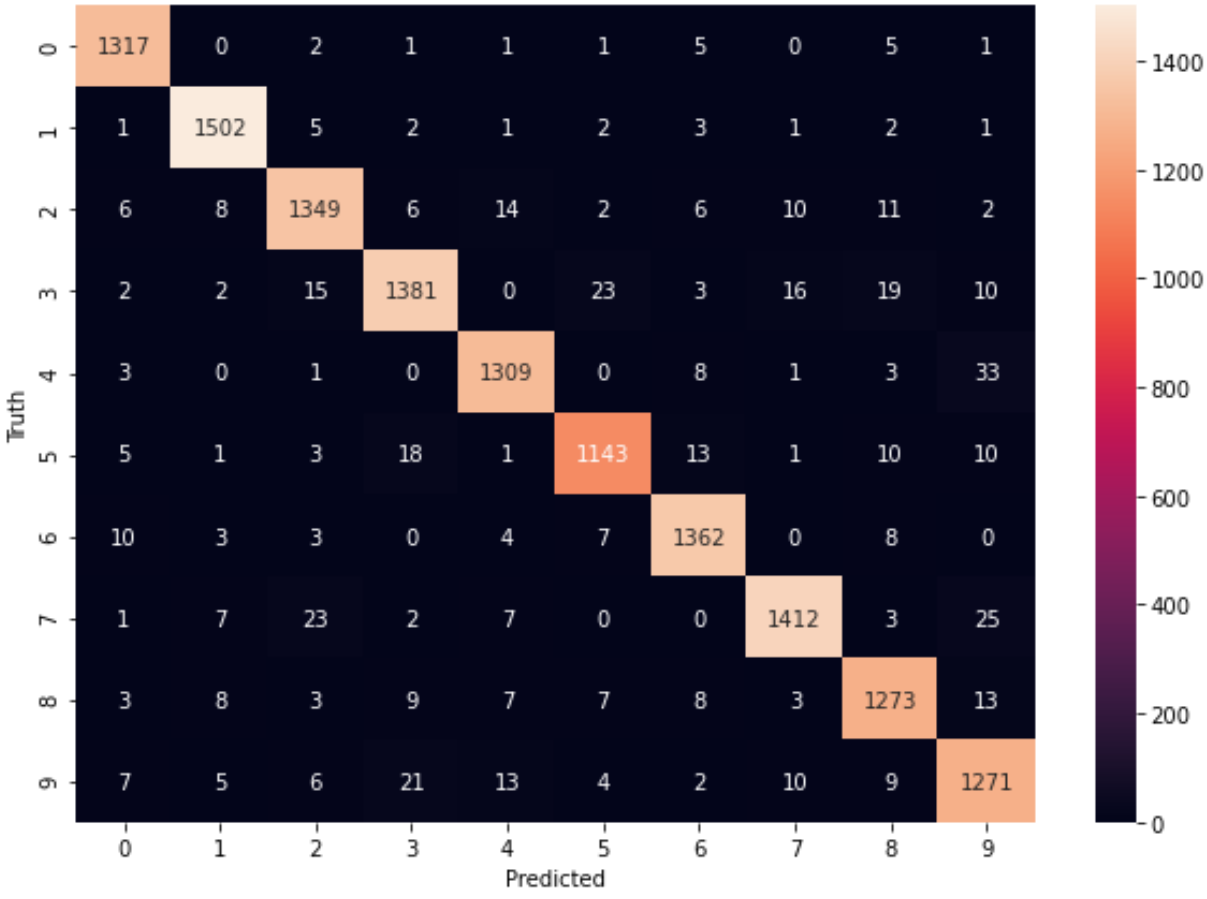
  - For N instances in the training set, sample N cases at random with replacement. The resulting set will be the training set of the tree.
  - For M input features, a number K << M is specified such that at each node, a subset of K features is drawn at random, and among which the best split is selected.
  - Each tree is grown to its maximum size and unpruned.

The idea of our experiments is to tune the RF main parameters in order to analyse the correlation between the RF performances and the parameter values. In this section, we first detail the parameters studied in our experiments and we explain the way they have been tuned. We then present our experiment protocol, by describing the MNIST database, the test procedure, the results recorded, and the features extraction technique used.

```
Accuracy: 96.09668109668109
```



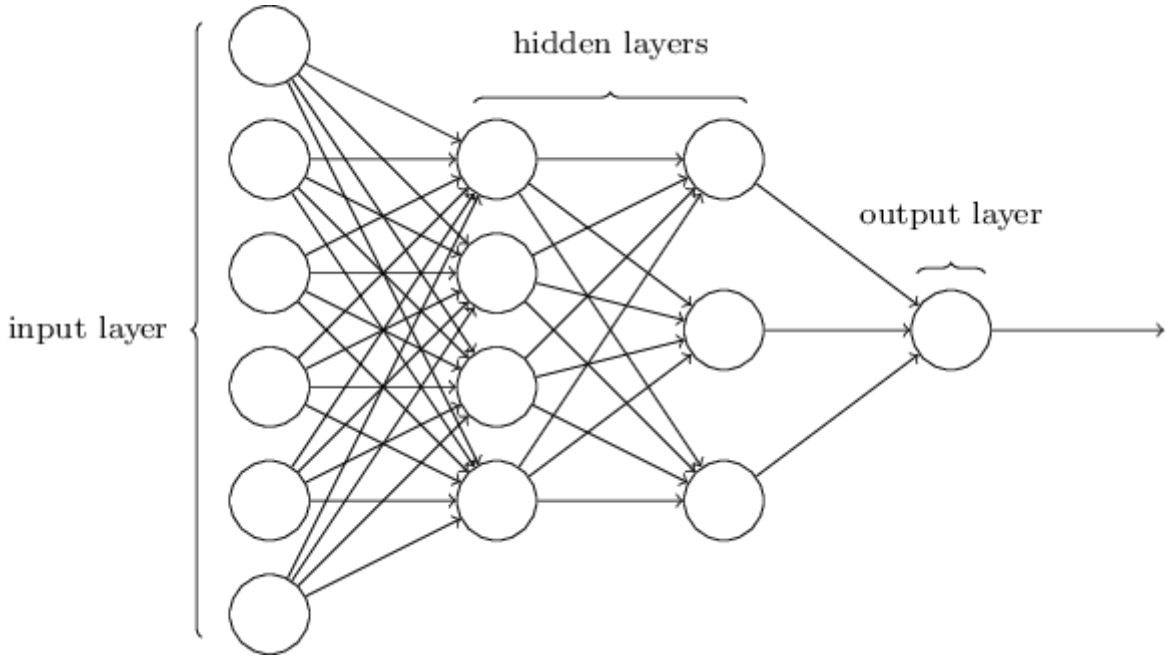
Confusion Matrix for Random Forest:



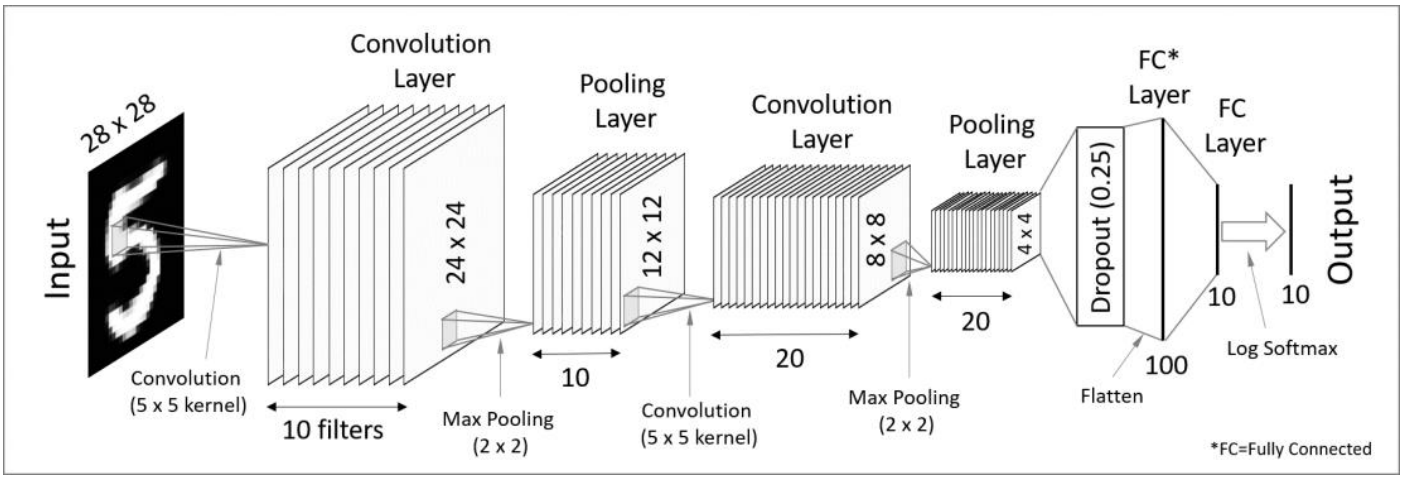
- **ANN (Artificial Neural Network):** Neural Networks mimics the working of how our brain works. They have emerged a lot in the era of advancements in computational power. Deep learning is the acronym for Neural Networks, the network connected with multilayers. The layers are composited form nodes. A node is just a perception which takes an input performs some computation and then passed through a node's activation function, to show that up to what context signal progress proceeds through the network to perform classification. It intended to simulate the behavior of biological systems composed of "neurons". ANNs are computational models inspired by an animal's central nervous systems. It is capable of machine learning as well as pattern recognition. These presented as systems of interconnected "neurons" which can compute values from inputs.

  A neural network may contain the following 3 layers:

  - Input layer – The activity of the input units represents the raw information that can feed into the network.
  - Hidden layer – To determine the activity of each hidden unit. The activities of the input units and the weights on the connections between the input and the hidden units. There may be one or more hidden layers.
  - Output layer – The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.



Artificial Neural network is typically organized in layers. Layers are being made up of many interconnected 'nodes' which contain an 'activation function.



$$Accuracy = (TP + TN) / N, \text{ where } N \text{ is sum of } TP, TN, FN, FP.$$

$$Expected\_accuracy = ((TP + FN) * (TP+FP) + (FP+TN) * (FN+TN)) / N$$

- TP – True Positive
- TN – True Negative
- FP – False Positive
- FN – False Negative

TP is a correct identification of positive labels, TN is a correct identification of negative labels, FP is incorrect identification of positive labels, FN is incorrect identification of negative labels. Overall effectiveness of classifier best defines the accuracy or portion of true results (meaning with the true positives and true negatives) from the total. The maximum value that accuracy can achieve is 1. This happens when classifier exactly classify two groups (i.e., FP = 0 and FN = 0). Remember that, The total number of True positive is TP+FN. The total number of a True negative is TN+FP.

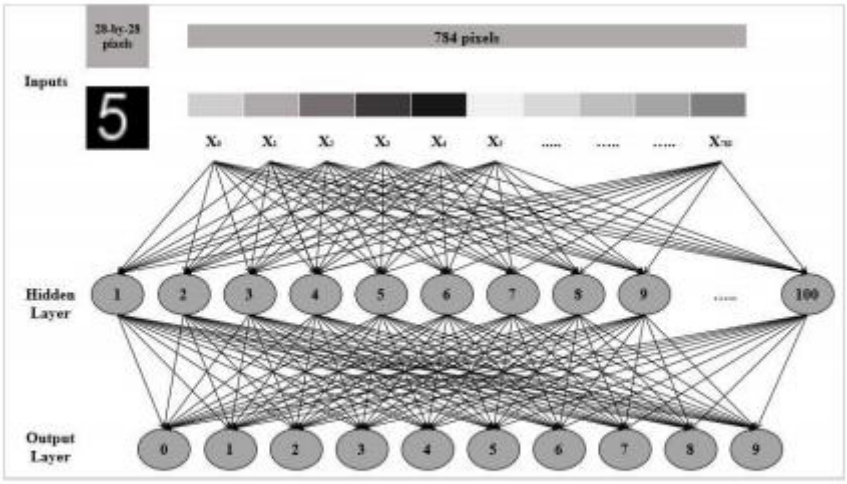
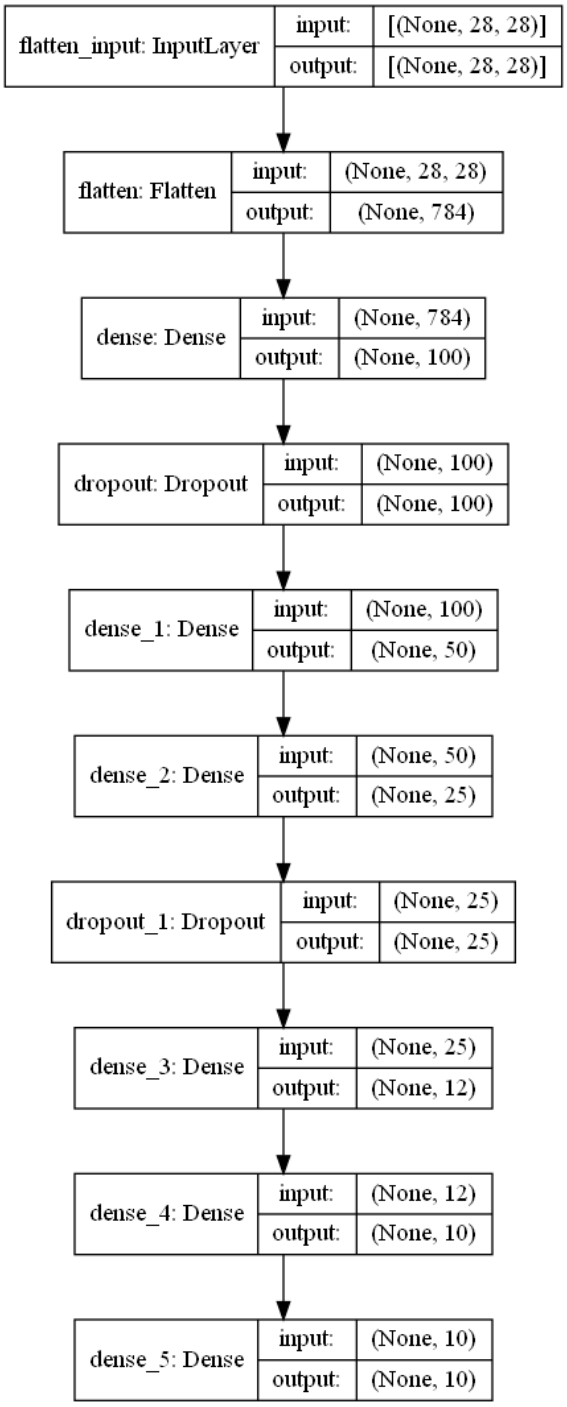
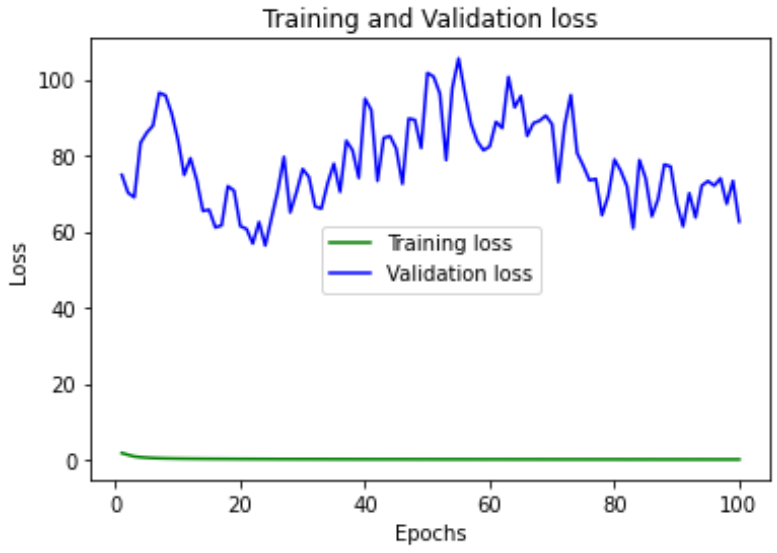


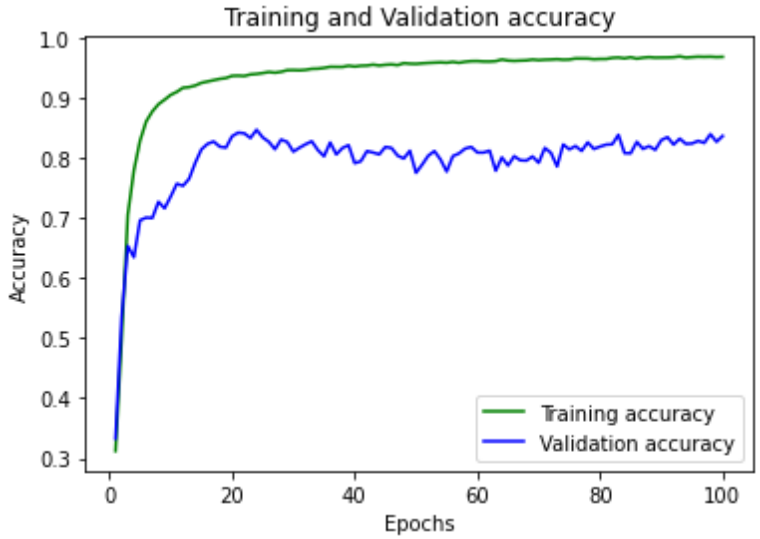
Figure 2. ANN architecture



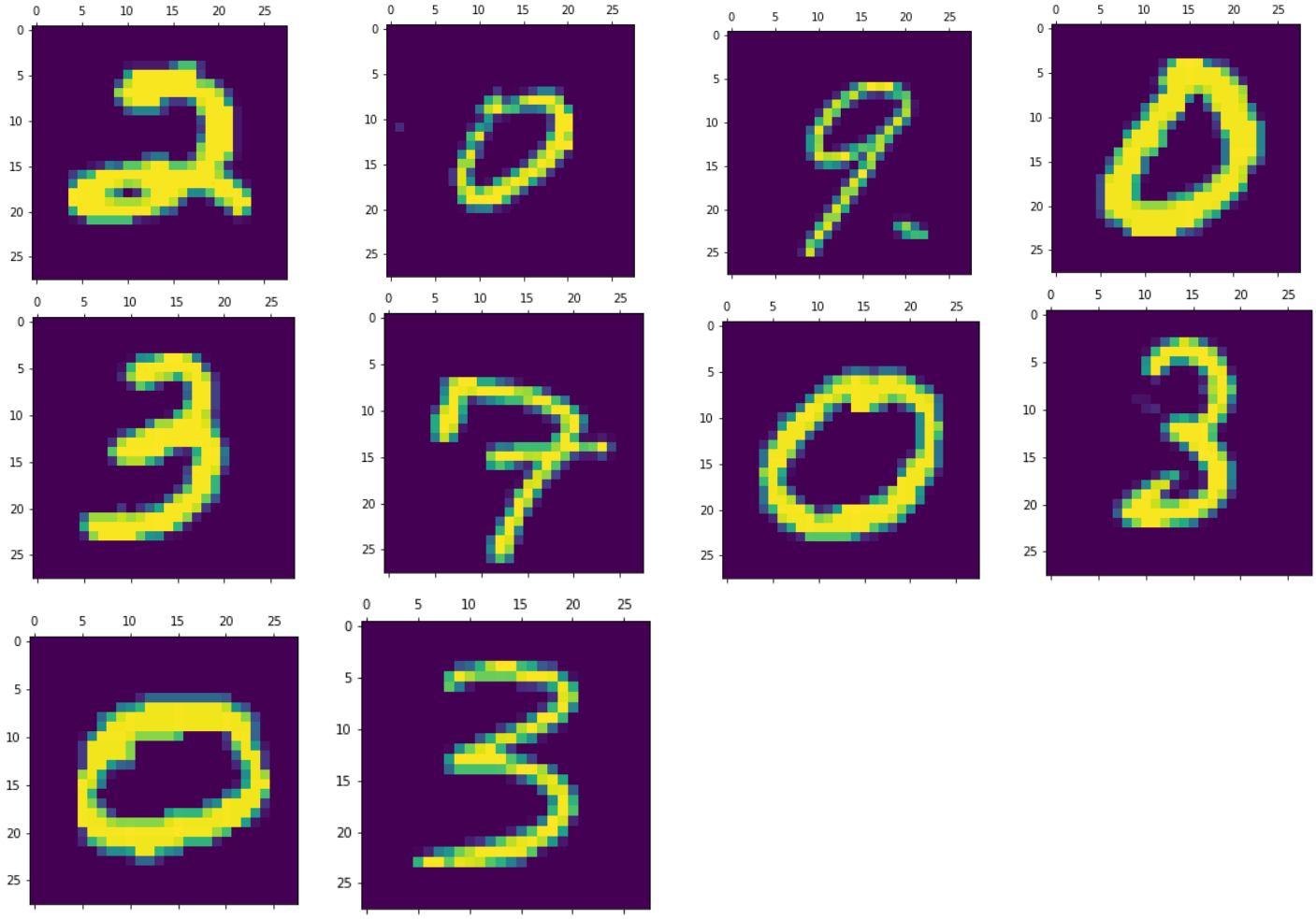
Model Summary:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 784)               0
_________________________________________________________________
dense (Dense)                (None, 100)               78500
_________________________________________________________________
dropout (Dropout)            (None, 100)               0
_________________________________________________________________
dense_1 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_2 (Dense)              (None, 25)                1275
_________________________________________________________________
dropout_1 (Dropout)          (None, 25)                0
_________________________________________________________________
dense_3 (Dense)              (None, 12)                312
_________________________________________________________________
dense_4 (Dense)              (None, 10)                130
_________________________________________________________________
dense_5 (Dense)              (None, 10)                110
=================================================================
Total params: 85,377
Trainable params: 85,377
Non-trainable params: 0
```





Predicted Images: range (10)



Actual Labels:

```
Label=[np.argmax(i) for i in Label]
Label[:10]
```

```
[2, 0, 9, 9, 3, 7, 0, 3, 0, 3]
```

Conclusions:

Implementations of different classifiers has been done and the test accuracies obtained here speak for the robustness of the algorithms chosen. Naïve Bayes classification method is based on Bayes' theorem that derives the probability of the given feature vector being associated with a label. Naïve Bayes has a naive assumption of conditional independence for every feature, which means that the algorithm expects the features to be independent which not always is the case. This leads to such a low accuracy, indicating that this is not the best algorithm out there. On the quest for better algorithms, an implementation of Logistic regression was also done. Logistic regression becomes a classification algorithm when the threshold value to decide the classes is brought into picture. It must be noted that the performance of logistic regression depends on the chosen threshold, which is dependent on the nature of the classification problem. K Nearest neighbours' algorithm was also considered to find the best output. The accuracy of the KNN algorithm varies deeply with the number of neighbours considered. If the number of neighbours chosen are less than the total number of classes, it would not mean much, because a few outliers might cause a lot of loss in accuracy. A value of 1- neighbours has been considered for now to reduce ambiguity for the model.

Due to computational constraint the accuracies for models with more neighbours was not investigated. Random forest algorithm is also one of the algorithms that was investigated. The literature survey has shown us that random forest algorithm outperforms the algorithms that were chosen earlier. The implementation we had done has proven our findings. We ignored decision trees while investigating random forests. The reason being decision trees tend to overfit. Random forests are an iterated version of decision trees that correct overfitting nature of decision trees. Due to this reason, they perform well compared to decision trees, KNN, Naïve Bayes and logistic regression. Due to our interest in neural networks, we ventured into investigating the problem statement by using artificial networks. The artificial neural networks have proven to be superior to all the above-mentioned algorithms. The concept of ANNs is based on how the human brain works.

The nodes and edges are modelled on the neurons and the synapses of the human brain. The weights are modified to adjust the learning. The elegance of the idea of ANNs is something that astonished us a lot. This has been the motivation behind using it in the project. The ANNs we used to give us an accuracy next to random forest algorithm. The ANNs are supposed to outperform random forests. But this has not happened in our case. The reason for this is very transparent. The performance of ANNs have a high correlation with the hyperparameters chosen. The problem with ANNs is that the algorithm might settle for a solution that gives a local minimum for the errors. By trying different learning rates and going through multiple iterations, the global optimum could be achieved. And we believe that with better computational power, we might be able to achieve the accuracies mentioned in the research papers, we had gone through. Upon thorough research and implementation of different algorithms, we have come to the conclusion that ANNs are better than all the other algorithms that were investigated earlier.

References:

- Offline Handwritten Digits Recognition Using Machine learning. October 2018.Conference: Proceedings of the International Conference on Industrial Engineering and Operations Management Washington DC, USA, September 27-29, 2018 .At: Washington DC, USAVolume: 133 https://www.researchgate.net/publication/328342141_Offline_Handwritten_Digits_Recogniti on_U sing_Machine_learning
- Recognition of Handwritten Digit using Convolutional Neural Network in Python with Tensorflow and Comparison of Performance for Various Hidden Layers https://arxiv.org/ftp/arxiv/papers/1909/1909.08490.pdf
- Handwritten Digit Recognition using Machine Learning Algorithms By S M Shamim, Mohammad Badrul Alam Miah, Angona Sarker, Masud Rana & Abdullah Al Jobair https://www.researchgate.net/publication/326408524_Handwritten_Digit_Recognition_Using _Ma chine_Learning_Algorithms

- Handwritten Digit Recognition Using Machine Learning: A Review, Publisher: IEEE https://ieeexplore.ieee.org/document/8976601

- Effective handwritten digit recognition based on multi-feature extraction and deep analysis, Publisher: IEEE https://ieeexplore.ieee.org/document/7381957

- S. Bernard, S. Adam and L. Heutte, "Using Random Forests for Handwritten Digit Recognition," *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, 2007, pp. 1043-1047, doi: 10.1109/ICDAR.2007.4377074.

  https://ieeexplore.ieee.org/document/4377074

- D. Gorgevik, D. Cakmakov and V. Radevski, "Handwritten digit recognition by combining support vector machines using rule-based reasoning," Proceedings of the 23rd International Conference on Information Technology Interfaces, 2001. ITI 2001., 2001, pp. 139-144 vol.1, doi: 10.1109/ITI.2001.938010.

  https://ieeexplore.ieee.org/document/938010

- Handwritten digits recognition with artificial neural network

  (PDF) Handwritten digits recognition with artificial neural network (researchgate.net)