Table of Content:

1. ABSTRACT

The report is focused on conducting classification of multinominal or multi-class data using four different types of classifiers. The report also focuses on various experimentation and parameter fine tuning conducting to derive the best classification model. Further it also explores topics such as pre-processing the data examining the data set provided. The outcome obtained from this report is the predicated labels for the test data provided and checking the accuracy of this prediction.

2. INTRODUCTION

a. Aim and Objective

The goal of this report is to exploit various machine learning classification-based algorithms to perform a multi-level classification of the data provided. The data is provided into 2 sets training and testing data. The training data contains about 30,000 samples with the labels provided for the classification and the testing data contains about 5000 samples with no label. The goal is to classify and provide the labels for the testing data.

The assignment forces you inculcate into critical thinking and analytical skills by making you perform various classification algorithms on the same data and understand what is working for the data and what can you do to improve the classification. It is like a lot of real-world implementations of machine learning based classification algorithms. We can also divide the goal and scope of the assignment on a microscopic level as:

- Learning about the data set
- Identifying Pre-processing techniques that would help enhance the classification
- Determining the classifiers to be used for the given classification problem
- Fine tuning the parameters of these classifiers
- Comparing the performance of the various classifiers implemented
- Obtaining the best possible predicted labels for the given test dataset.

a. Understanding the Dataset

The dataset consists of grayscale images of the size 28x28 that can be classified into a set of categories. The dataset consists of a training set of 30,000 examples and a test set of 5,000 examples belonging to these 10 categories.

The training dataset 30000 samples contains both the feature and label in the data, however upon inspection it is found that the testing dataset does not have any labels. So, for the purpose of understanding the accuracy of the classification models the training data is split into training and testing data into a ratio of 75% training (2 1 and 25% testing (7500 samples). Eventually the working model will be i the final output test data (5000 samples) to predict test labels.

The following is an example of the samples in the dataset:

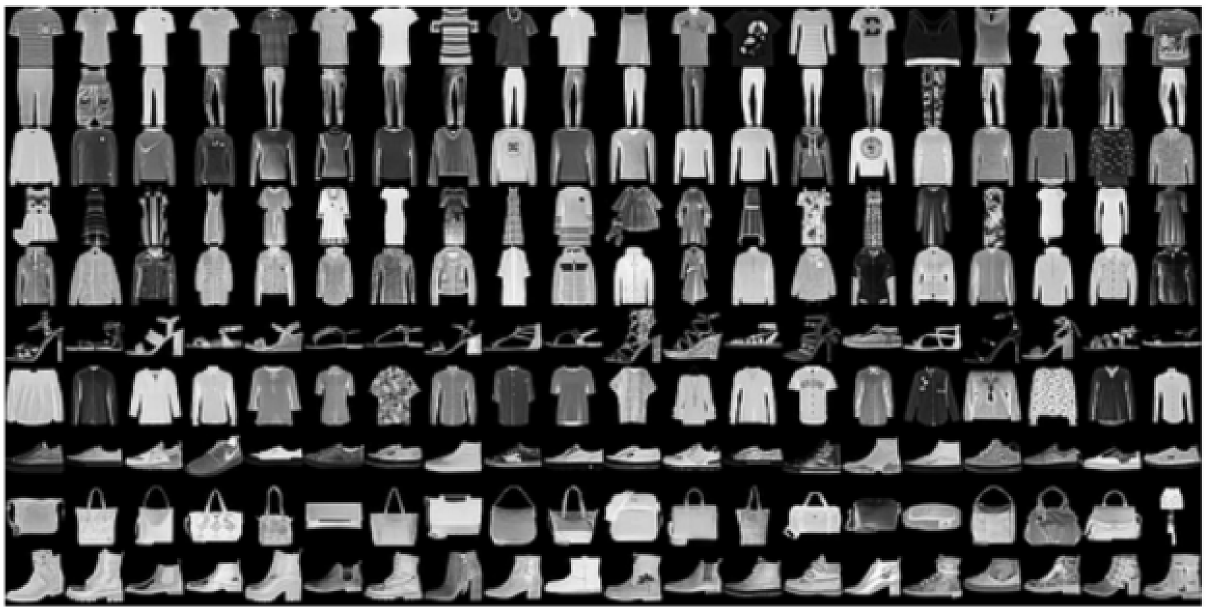


Fig 1: Examples of samples from the dataset

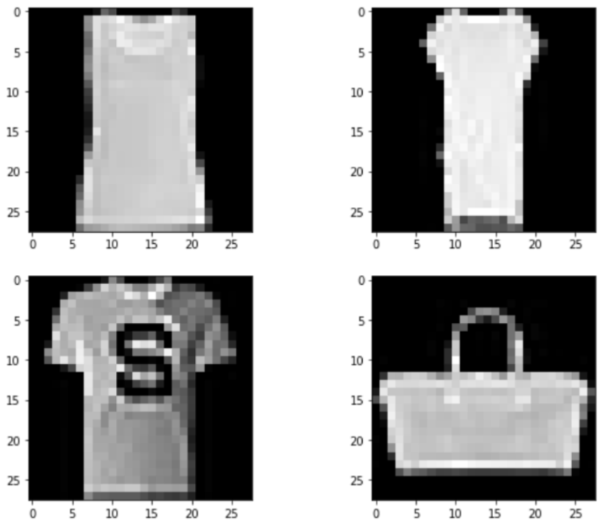


Fig 2: Examples of samples from the dataset

These are the 10 label categories for classification:

0: T-shirt/Top	1: Trouser	2: Pullover	3: Dress	4: Coat

5: Sandal	6: Shirt	7: Sneaker	8: Bag	9: Ankle boot

3. METHODS

a. Data Pre-processing

Data pre-processing includes data preparation, compounded by integration, cleaning, normalization and transformation of data; and data reduction tasks, such as feature selection, instance selection, discretization, etc. The result expected after a reliable chaining of data pre-processing tasks is a final dataset, which can be considered correct and useful for further data mining algorithms.

For this assignment we have used the following pre-processing techniques:

i. Normalization:

We cannot say for sure that measurement unit for a data is not affected during analysis. However, it necessary to express all attributes in the same measurement units which is where normalizing the data to give all attributes is a usual statistical learning method The measurement unit used can affect the data analysis. We have used the technique of Min-Max normalization it aims to scale all the numerical values v of a numerical attribute A to a specified range denoted by [new − min A, new – max A]. Thus, a transformed value is obtained by applying the following expression to v in order to obtain the new value v' [1]:

$$v' = \frac{v - min_A}{max_A - min_A}(new - max_A - new - min_A) + new - min_A,$$

where max A and min A are the original maximum and minimum attribute values respectively.

ii. (PCA)Principal Component Analysis

It's a commonly known fact that it is critical to reduce the number of factors or features to a few important ones to arrive at the right decision. This process is called dimensionality reduction. In machine learning, the problem of high dimensionality is dealt in two ways [2]:

- Feature selection — is carefully selecting the important features by filtering out the irrelevant features

- Feature extraction — is creating new and more relevant features from the original features

Principal Component Analysis (PCA) is one of the key techniques of feature extraction [2]. PCA basically assists you find an axis along the entire dataset reserving the maximum information. For this assignment we have used covariance matrix to deduce the eigen value and eigen vectors. The spread of the data is along the principal components is examined b exploiting these eigen vectors.

b. Classifiers Applied

For this project the following are 4 classifiers used these classifiers are selected since they are useful for classifying multinominal data. Here is a basic functionality of the algorithm explained in simple words

i. (KNN)K-Nearest Neighbours Classifier:

The KNN classifier is a non-parametric method used for classification for a set of given N vectors. The algorithm basically identifies the k-nearest neighbours for the N-vectors, where in it identifies the class of the unknown feature vector.

There are various distance calculation methods that can be used to evaluate the distance between any given data point and its n-nearest neighbours. For the purpose of the project we have computed the (D)Euclidean distance which is mathematically given as:

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

where X and Y are data points.

ii. Naive Bayes Classifier

Naïve Bayes classifies using Bayes' Theorem of probability [3]. Bayes' theorem calculates the posterior probability of an event (A) given some prior probability of event B represented by P(A/B) as follows:

$$P(A/B) = P(B/A)\ P(A)\ P(B)$$

where,

• A and B are events. • P(A) and P(B) are the probabilities of observing A and B independent of each other. • P(A/B) is the conditional probability, i.e., Probability of observing A, given B is true. • P(B/A) is the probability of observing B, given A is True. [3]

The joint probability distribution for a binary class problem as described by the Naïve Bayes classifier is:

$$P\,(X = x, Y = c) = P\,(Y = c)\; P^{\cdot}\,(X = x \mid Y = c)$$

iii. (SVM)Support Vector Machine Classifier 4

SVM is a supervised learning algorithm. It works on the (in calculation. This algorithm plots each data item as a point in n-dimensional is the number of features, we have in our dataset). The value of each feature is the value of the corresponding coordinate. It classifies the data into different classes by finding a line (hyper plane) which separates the training datasets into classes. It works by maximizing the distances between the nearest data point (in both classes) and the hyper plane that we can call as margin [3].

iv. Random Forest Classifier

The ensemble learning algorithm used for this project is Random Forest Classifier which can be used for both classification as well as regression models. This algorithm creates a set of decision trees by selecting a random subset of data using the approach of bagging [3]. The Random Forest Algorithm has two stages one is creating a random forest and the second one is to make predictions based on the classifier created in the first step

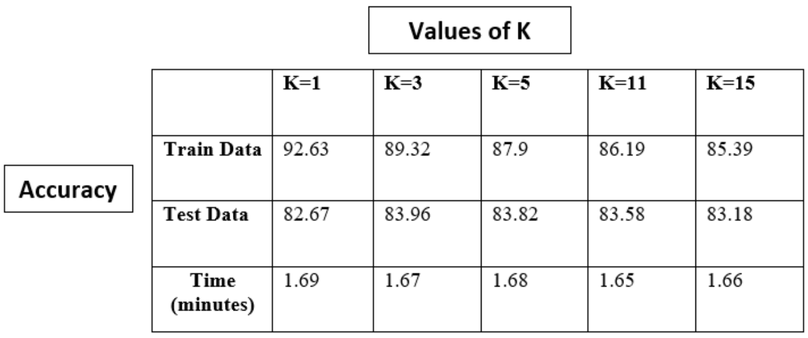
4. EXPERIMENTATION AND RESULTS

a. Experimentation

i. KNN Algorithm

The KNN model was created using required NumPy, pandas' libraries. The classifier was trained using a training data and testing data obtained after splitting the train data into train and test and finally the classifier was used to generate the output label file contains the class for input test feature data.

Further the values of K were altered manually to detect best parameters. K=1 shows the highest accuracy however it is ignored due to over fitting. Grid Search based hyper parameter fine tunning was also conducted to obtain the best parameters. The following table shows the train and test accuracy scores for the different values of k:



Values of K

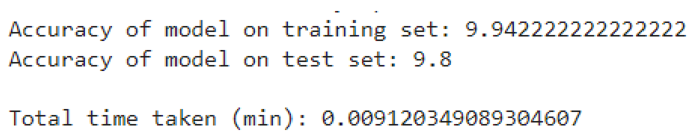
Accuracy

	K=1	K=3	K=5	K=11	K=15
Train Data	92.63	89.32	87.9	86.19	85.39
Test Data	82.67	83.96	83.82	83.58	83.18
Time (minutes)	1.69	1.67	1.68	1.65	1.66

Fig : Accuracy scores for different values of K

ii. Naïve Bayes Algorithm

The Naïve Bayes Algorithm was created by exporting the 5 es. It was trained and tested based on the training data and testing data. T ance of this model is as followed:



```
Accuracy of model on training set: 9.942222222222222
Accuracy of model on test set: 9.8

Total time taken (min): 0.009120349089304607
```

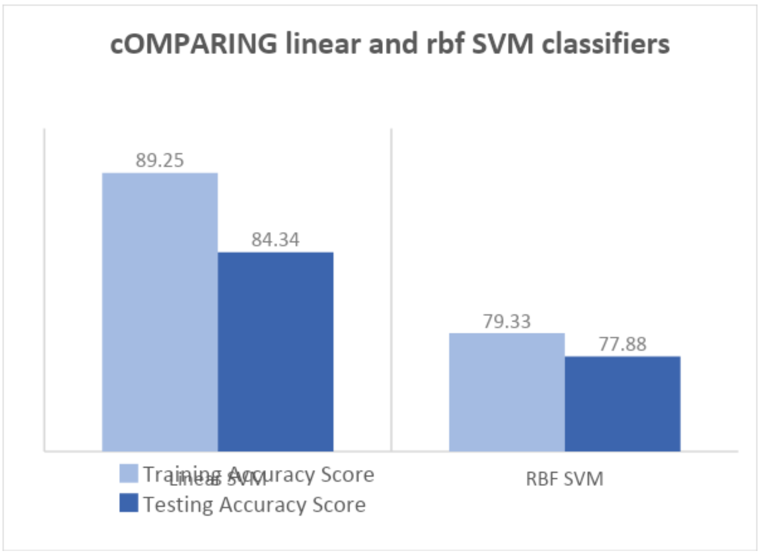
Fig: NVB Output

As we can observe that this algorithm has a very low accuracy hence, we tried to fine tune it by changing the learning rate which did not create much difference. Therefore, this model won't be considered for further analysis.

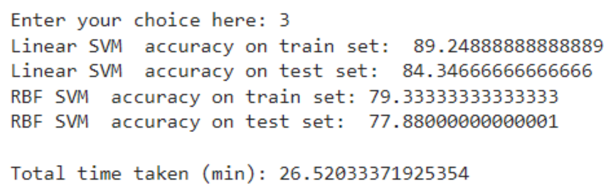
iii. SVM Algorithm

The SVM algorithm was created by importing the required libraries. The SVM classification was implemented on the test data and the train data. I have also tried to create check the linearity of the classification of data by implementing linear SVM that is based on parameters and RBF kernel based SVM. As we can observe from the fig. below since our data is linearly classifiable the accuracy of the linear classification was higher. Note: The RBF based kernel was also more expensive and took longer to implement the classification that the linear SVM.

Fig: Comparing Linear and RBF SVM Classifiers



The following is the sinpet of the accurracy scores obtained using both the SVM models:



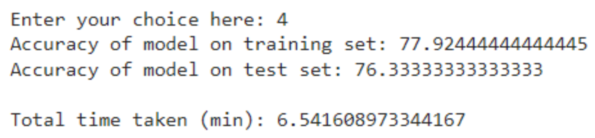
```
Enter your choice here: 3
Linear SVM  accuracy on train set:  89.24888888888889
Linear SVM  accuracy on test set:  84.34666666666666
RBF SVM  accuracy on train set: 79.33333333333333
RBF SVM  accuracy on test set:  77.88000000000001

Total time taken (min): 26.52033371925354
```

6

Fig: SVM Output

iv. Random Forest Algorithm

Random Forest Algorithm is selected as an ensemble method. All the required numpy and pandas libraries were imported for the classifier implementation, the following are the accuracy scores obtained on the training and testing classifier using this ensemble method:



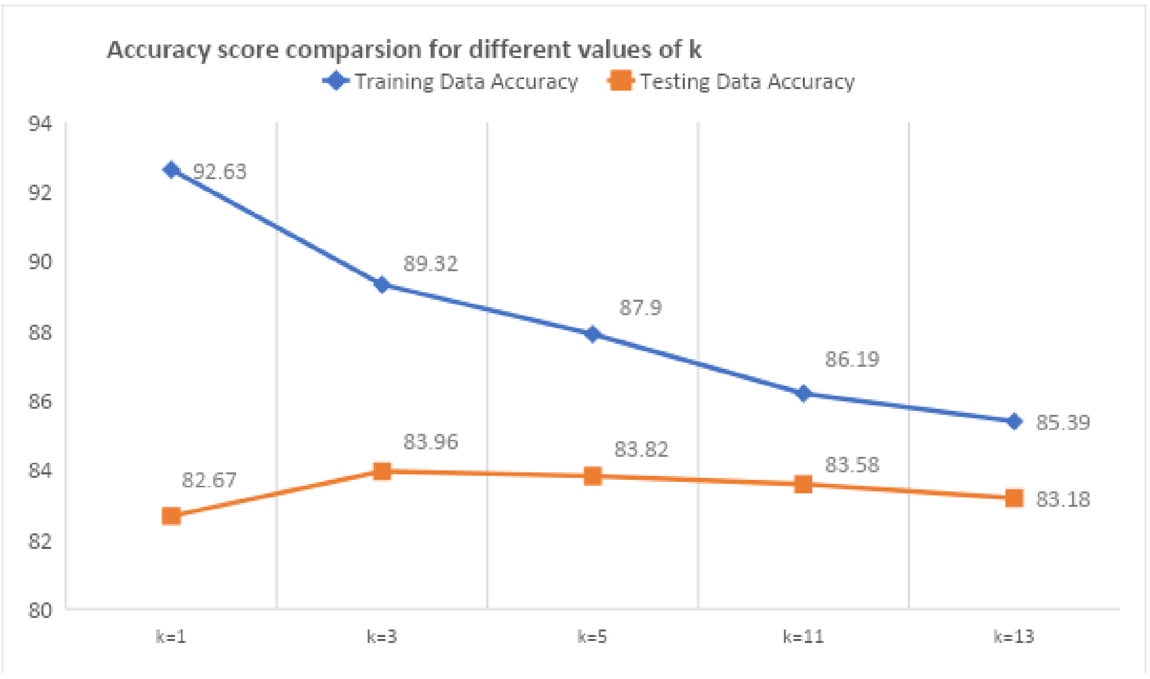
```
Enter your choice here: 4
Accuracy of model on training set: 77.92444444444445
Accuracy of model on test set: 76.33333333333333

Total time taken (min): 6.541608973344167
```

Fig: Random Forest Output

b. Impact of Fine-tuning parameters of algorithms

- For K-NN Algorithm fine tuning of parameters was done by using 2 techniques, one the value of K was manually changed with the values k=1,3,5,11 and 15. The parameters were used to calculate the accuracy score on the test data and train data and the following output was obtained

Fig: Accuracy score comparison for different values of K



To further enhance the performance of a KNN classifier when is selected as the best model for classification of the given data we have performed hyper-parameter fine tunning using Grid search based on 10-fold cross validation. The following is the grid-search implementation with the output results:

Fig: Grid search implementation snippet for KNN Classifier
(Note: Please zoom in to see the detailed image)

- For SVM classifier we tried to perform hyper-parameter fine tuning using cross validation-based grid search, however it took a lot of time more than 4 hrs and hence this approach for discarded.
- For Naïve Bayes classifier we tried to perform hyper-parameter fine-tuning by varying the learning rate, however it did not show any major difference in the accuracy scores and hence it was discarded.

c. Results

Upon inspection it was derived that Linear SVM and KNN classifier produced the best accuracy scores in comparison to the rest of the models. The accuracy score comparison can be seen in the diagram below:

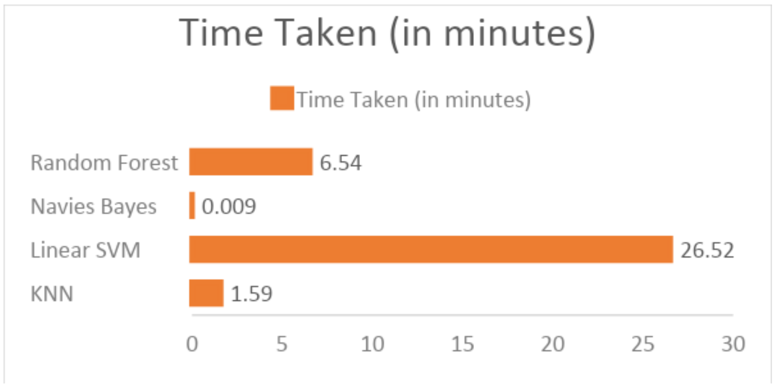


Fig: Time taken by each classifier for execution

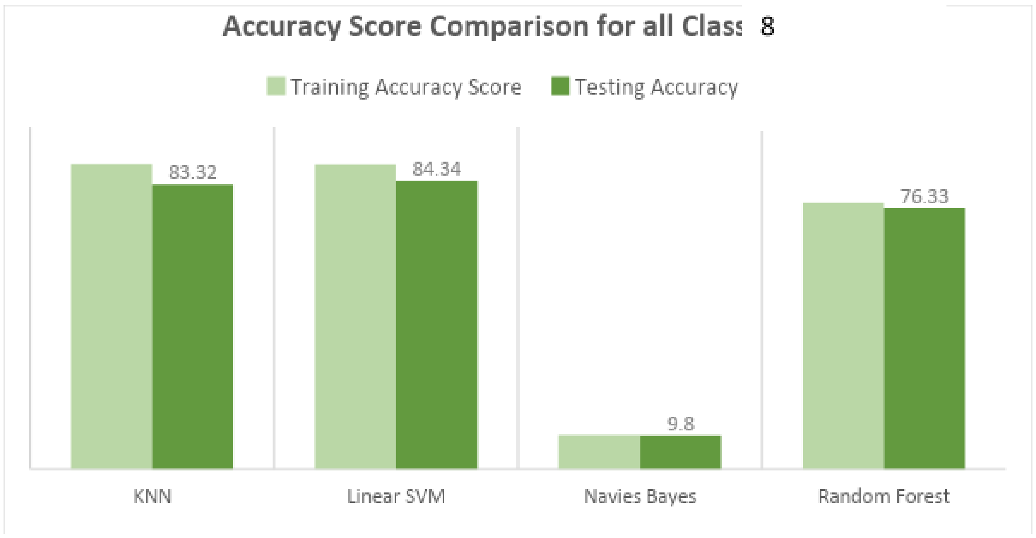


Fig: Accuracy Score comparison for all classifiers

Upon analysing the above results Linear SVM took precedence over KNN in terms of the accuracy scores however in terms of the time taken KNN is much faster than SVM and the difference in the testing accuracy scores is merely 1% therefore KNN is selected as the best classifier model.

Finally, to enhance the KNN classifier and identify the best hyper-parameters a 10-fold cross validation is performed on the KNN classifier.

After obtaining these parameters the KNN classifier was implemented on the test data set containing 5000 unlabelled samples. The output was uploaded on Kaggle the accuracy score obtained was 84.15%