Handwritten Digits Recognition: A Comprehensive Analysis

Arshdeep Singh
2019csm1001
Sam Zabdiel Samuel
2019csm1012

Indian Institute of Technology
Ropar, India

Abstract

Our project implements a handwritten digits recognition system with the famous MNIST dataset. We have separated our project into two modules. The first module gives the complete analysis of the MNIST dataset. The second module is followed by experimenting with different machine learning algorithms to learn first-hand what works well and how techniques compare with each other.

1 Introduction

Handwritten digit recognition is a task that is done by sighted, literate humans accurately and effortlessly out of habit. Recognition of handwritten digits by machines has potential to greatly improve productivity of society. Postal codes on mail, amount in cheques, forms filled by hand, etc could be translated directly to the machine if such recognition were to be human-like accurate. And without the need for manual intervention and using pipe-lined processes, such tasks like data entry from forms or cheques, routing of mails, etc could be done faster akin to assembly lines.

The understanding of the underlying mechanisms of the ability to recognise handwritten digits (or for that matter even characters) by humans is still vague. But we can naturally tell that there are certain features of handwritten digits that help us recognize it. Like, 1 looks like a stick, or a capitalized i. 0 resembles an O or a donut, flattened vertically. But there are variations also, like 1 could also be written as it is printed as it exhibits in this sentence (a slant mark connecting to the upper end of the stick, and a line segment at the base). Our algorithms are limited in scope compared to our brain. Such analysis using features cannot be obviously translated to computer algorithms.

Here we use the MNIST dataset, which is 60,000 training images of handwritten digits 0-9, existing as 28x28 pixels. Now the algorithms may be able to work around this representation to recognise patterns and eventually predict the identity of handwritten digits. This project is the study of certain algorithms applied on the MNIST data, a comparative analysis.

We have at first analysed the MNIST dataset, to observe and evaluate approaches that would best fit this situation. In this project we have used six different algorithms for classification and with the analysis of the MNIST dataset. A sample collection of images from the MNIST dataset is shown in Figure 1.

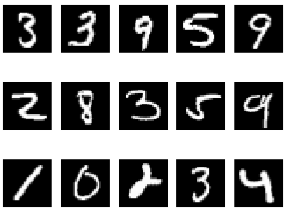


Figure 1: Sample MNIST images

2 Literature Survey

MNIST is a popular dataset used throughout Data Science and Machine Learning community. There are a lot of classifiers which have used MNIST dataset to analyse their model performances. As of today, the dataset has somewhat regarded as a benchmark for analysing the performances of various machine learning models. Popular research papers have used MNIST dataset to publish their classifier results. Various metrics are used in analysing the performance of the machine learning models. Among that, 'test error rate' is quite popular. The metric defines the how many unknown samples the model is able to correctly classify.

Many test-error-rate benchmarks have been set under the MNIST dataset [8]. There are various linear models proposed under which they are able to give test error rate ranging between 7.6%-12%. The K-nearest neighbour (KNN) approach is able to give an error range between 1.1%-5%. The neural network (NN) approach behaves in similar fashion giving an error range of 1.6%-4.7%. Moving further, Support Vector Machines (SVMs) improves the range even more by reducing the test-error-rate to 0.8%-1.4%. The Convolutional Neural Network (CNN) approach also gives similar result with respect to Support Vector Machines (SVMs) give the test-error-range of 0.7%-1.7%. Moreover, to make model more robust, data augmentation is also carried out on the dataset. Data augmentation is the distortion of the original image by some degree guided by many techniques to carry out the augmentation. Adding additional augmented data makes the model more robust. As a result, the Convolutional Neural Network was able to perform really well. The rate of test error estimation was 0.95%. That means, with the CNN model, we are able to efficiently get a test accuracy of 99% with 3-layer model. It is proved that we are able to efficiently achieve a better model performance with image segmentation. Lecun *et al.* believes that there is no need for the model to aim for full feature set. As human eye does not perceive all the image features all at once, then we can train the model with augmented data with distorted features to better generalise the model.

The authors in [10] have carried out various popular machine learning algorithms in their paper with an attempt to analyse its behaviour on same data but different models. Discriminant models such as Linear Discriminant Analysis and Quadratic Discriminant Analysis are applied. Among the unsupervised approach, Gaussian Mixture Models were applied which were able to give better accuracy than any other unsupervised approach. The supervised learning was further carried out by K-nearest neighbour and Support Vector Machines. Hyper-parameter tuning was also carried on all the models. For example, KNN approach at K = 3 was giving the highest test accuracy among all other inputs for K.

The authors in [4] proposed a hybrid approach incorporating supervised and unsupervised learning. They performed Deep Convolutional Neural Network on the output of K-

Means clustering on the MNIST dataset. The model performance was optimal and there was reduction of correlated parameters. The model performed near equal to the typical CNN approach giving the test error rate of 0.5%. Similarly, the authors in [9] were able to efficiently acquire a test error rate metric between 0.54%-0.83% by initially applying SVM approach to the dataset and then applying the CNN model. SVM helped in becoming a good feature extractor. Reverse procedure was employed by the authors in [7]. Here CNN approach was employed for the the extraction of the features and then Support Vector Model was trained on the improved dataset.

Dimensionality reduction techniques were applied in [2] and [3] and reduced dataset was then fed into the various supervised classifiers and metrics were calculated. The author were able to achieve a very low test error with range between 0.5%-0.62%. Principal Component Analysis was majorly employed for the dimensionality reduction. Principal Component Analysis approach redistributes the data along axes of the decreasing variance and is an efficient feature extraction mechanism. Linear discriminant Analysis behaves in a similar but supervised way.

The authors in [1] have explained the whole procedure for Multi-Layer-Perceptron for the MNIST dataset. The paper lays down the formulation of efficient back-propagation and forward-propagation in terms of MNIST Dataset. The paper is able to achieve a test accuracy of 99.08%. The paper also analyses the accuracy of all the ten digits. Digit 5 was having the highest test accuracy of 99.08% with the lowest falling just short of 0.02% which was of Digit 2. The model was able to achieve good test accuracy predictions for all the classes.

The authors in [6] published various results in applying SVMs on MNIST with different kernel methods. The authors tested the Support Vector Model on various kernel methods such as Linear Kernel, Polynomial Kernel, Quadratic kernel. Further SVMs were analysed alongside multi-layer-perceptron model for better judgement. Among all the kernels tested, the linear model was able to aim and simultaneously achieve the highest test accuracy of 94.8%.

The authors in [10] describe the scalability of applications of Handwritten Digit Classification at the expanse of input data, processing capability and output mechanism. Due to this, the dataset has become an area of great interest for many researchers. MNIST is a popular dataset used throughout Data Science and Machine Learning community. There are a lot of classifiers which have used MNIST dataset to analyse their model performances. As of today, The dataset has somewhat regarded as a benchmark for analysing the performances of various machine learning models. The dataset is affected by class invariance. There are different ways and style to write a single digit in the handwriting. Hence, for better generalization of the model data augmentation is done. The distorted image data is appended alongside to the original data. Various libraries developed are available in Tensorflow for data augmentation. The dataset make the model style invariant. The paper mentions the integration of generative models such as Gaussian Mixture Models, Multivariate Models with discriminative models such as Suppot Vector Machines, Neural Nets, KNN classifiers. The integration is helpful in achieving low test error rate

The authors in [5] proposed a hybrid model comprising of Support Vector Model and K Nearest Neighbour. The K nearest neighbour model sometime suffers from bias-variance trade-off which hampers the the accuracy of the model. Support Vector Model on the other hand takes up too much of time in finding out out the efficient hyper-plane in the latent space. So, the paper proposes an hybrid solution which take the best of both the models and is able to achieve better results against the established benchmarks. The approach generalises efficiently in the categorizing the multi-class data.

3 MNIST Data Analysis

Before applying classification algorithms, we will do Exploratory Data Analysis of our MNIST data. Prior data analysis is useful in analysing how different models will behave with the MNIST data. After that, preliminary results are published of the carried out classification algorithms.

- **Exploring data in terms of pixels**
  Here, we explore how the pixel values (0-255) are distributed in the dataset. Histogram is plotted for all the pixel values.

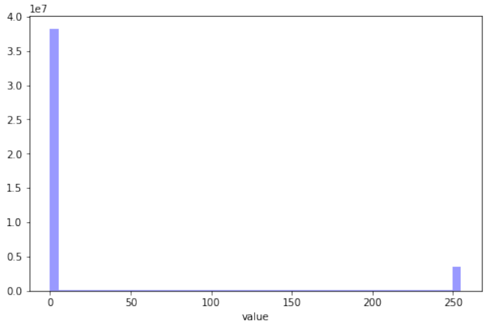


Figure 2: histogram of pixel values

Observation : Most pixels in the dataset are completely black (pixel value = 0). The pixels corresponding to the various digits are mostly white with pixel value of 255. The remaining pixels (very few) are between value 0-255.

- **Variability within each digit label**
  Here, we will analyse the digit variance. There are different ways and style to write a single digit in handwriting. So we take the mean of all the same class image and plot the resulting image of all the 10 digits. Digits having high proportion of white pixels will signify better class invariability.

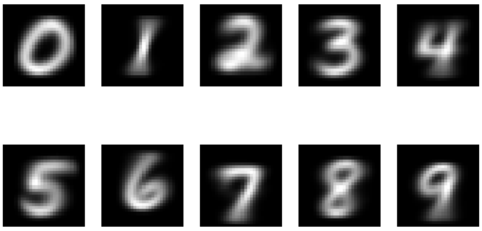


Figure 3: The average images of 10 MNIST digits

Observation : Digits like 2, 3, 7, 0, 1 are having high proportion of white pixels. That means these digits are particularly written in same style by people. However, 3, 4, 5, 6, 8, 9 are having high style variance. People's style trends in writing these digits are quite variable. These digits might have low test accuracy than the other former batch.

- **Atypical instances**
  Here, we will analyse atypical instances of our class labels. These are the instances which have high variability from their typical structure. We have used Euclidean distance which is square root of the sum of squares to label each image to its centroid. The boxplot of Euclidean distances of each class label is plotted in **Figure 4**.

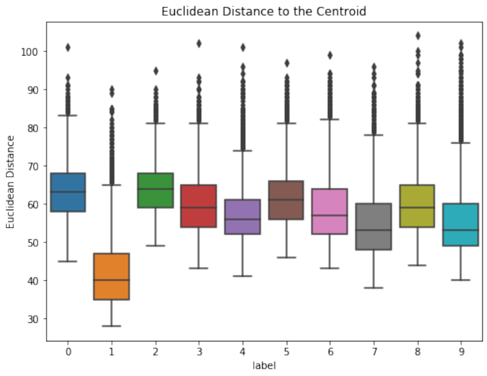


Figure 4: Euclidean distances to the digit centroids

Observation : Digits like 1, 4, 7, 8 and 9 are showing maximum variability with maximum outliers. Maximum outliers indicates maximum variability and hence maximum Atypical Instances. The mean image of these digit classes will have least amount of white pixel density.

Top five highly atypical instances (based on Euclidean Distance) belonging to each class label are plotted below.

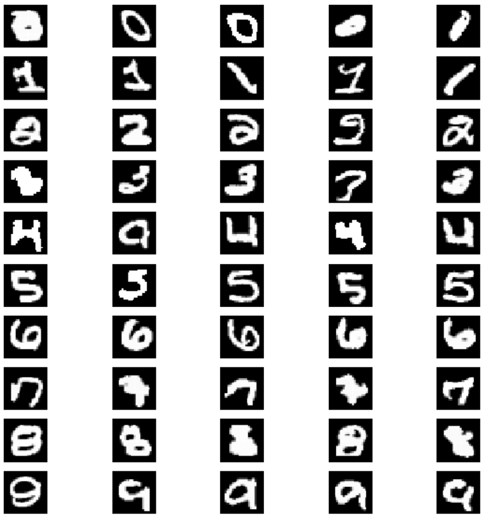


Figure 5: Top five most atypical instances of each class

4 Dimensionality Reduction and Visualization

Here, the visualization of MNIST dataset is carried out. Our dataset has 784 Dimensions. As visualization in high dimensional space is nearly impossible, we will apply dimensionality reduction mechanism to our data set. Here first, Principal Component Analysis is carried out. The *n_components* of the PCA method in Sklearn library is set to 2. After that the data is plotted on the 2D graph. The plot is shown below.

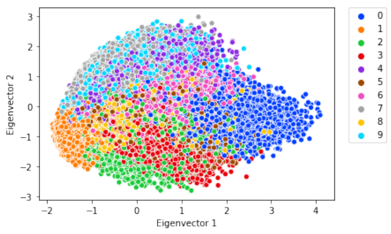


Figure 6: 2D PCA on Train Set

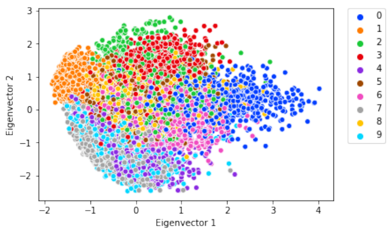


Figure 7: 2D PCA on Test Set

Observation : As we only few digits which are separately visible. Rest of the digits are forming a mixture pattern. There is considerable loss of variance converting the 784 dimensional MNIST to 2 dimensional. Among all the digits, digit 0 is mostly separated. This should reflect in the training of ML algorithms with digit 0 generating the highest precision and recall.

Next, we will apply another dimensionality technique, i.e, t-SNE. As we know TSNE is a good dimensionality reduction and visualization technique. The important advantage of TSNE is that it maintains and preserves the pair-wise property of similarity between the data samples. The similarities can be in terms of distance or any other measure metric. The similarities in the high dimensions will be reflected back into the low dimensional state space. The plot of 2D TSNE is shown below.

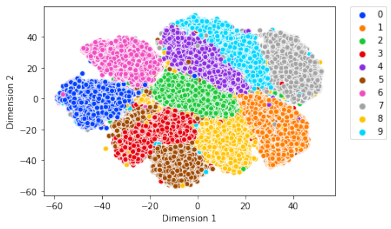


Figure 8: 2D TSNE on Train Set

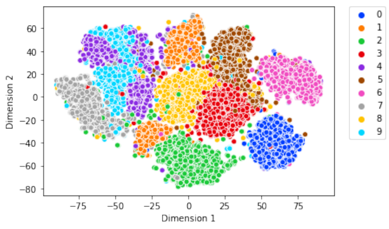


Figure 9: 2D TSNE on Test Set

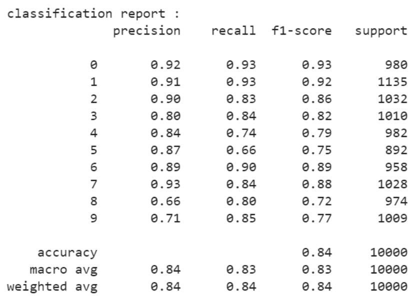
Observation : The TSNE plot is clearly showing the separated clusters. The separated clusters indicates that the dataset is also distinguishable in the 784 dimensional space state. The behaviour and output of the TSNE plot should reflect in the classification of different ML algorithms. The resulting plot of TSNE indicates that getting accuracy of more than 95% is feasible.

5 Experimental Results and Analysis

We have used six different algorithms for classification analysis. The algorithms are : Gaussian Naive Bayes, Logistic Regression, K-Nearest Neighbours, Random Forrest, Support Vector Machines and Convolutional Neural Network. After fitting the models on the Train data, we have calculated our results in the form of model **accuracy** (on Test data). In order to get an idea of True Positive and False Positive values predicted by our model, we have calculated **Confusion Matrix** of each model. Also we have used sklearn's **classification_report** method to generate model report.

5.1 Bayes' Classifier

The Bayes' classifier works best in the sense if we consider that the features are independent of each other. It is a generative model and uses Naive Bayes assumption for classification and generalises very well. Here, we are using Multinomial Naive Bayes for our implementation. **Model accuracy achieved : 83.57%**.

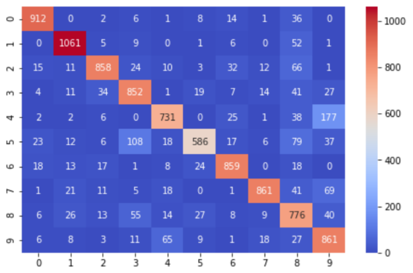


```
classification report :
              precision    recall  f1-score   support

           0       0.92      0.93      0.93       980
           1       0.91      0.93      0.92      1135
           2       0.90      0.83      0.86      1032
           3       0.80      0.84      0.82      1010
           4       0.84      0.74      0.79       982
           5       0.87      0.66      0.75       892
           6       0.89      0.90      0.89       958
           7       0.93      0.84      0.88      1028
           8       0.66      0.80      0.72       974
           9       0.71      0.85      0.77      1009

    accuracy                           0.84     10000
   macro avg       0.84      0.83      0.83     10000
weighted avg       0.84      0.84      0.84     10000
```

Figure 10: Classification Report



	0	1	2	3	4	5	6	7	8	9
0	912	0	2	6	1	8	14	1	36	0
1	0	1061	5	9	0	1	6	0	52	1
2	15	11	858	24	10	3	32	12	66	1
3	4	11	34	852	1	19	7	14	41	27
4	2	2	6	0	731	0	25	1	38	177
5	23	12	6	108	18	586	17	6	79	37
6	18	13	17	1	8	24	859	0	18	0
7	1	21	11	5	18	0	1	861	41	69
8	6	26	13	55	14	27	8	9	776	40
9	6	8	3	11	65	9	1	18	27	861

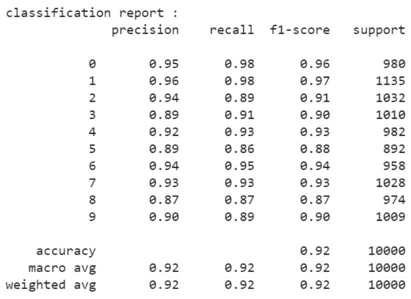
Figure 11: Confusion Matrix

Observation : The Naive bayes model is giving the lowest test accuracy among all the trained models. This indicates that the Naive bayes assumption of the feature set of the MNIST is not entirely independent. That indicates the presence of the correlation between the feature space of the 784 dimensional dataset. That is why Naive Bayes model is generating a high test error rate of 16.43%.

5.2 Logistic Regression

Logistic regression is a supervised machine learning algorithm. Logistic Regression uses linear weighted combination of the data and uses probability methods for predicting unseen data. As we have to implement a model that can predict values for digits from 0 to 9, we are using a Multinomial Logistic Regression i.e. one with multiple outputs, each of which is used to predict a single output class. **Model accuracy achieved : 91.98%**.

Observation : The Logistic regression model significantly reduces test error rate to 8.02%. Logistic regression tries to find the linear discriminant boundary that separates one class from the others. This method called 'ovr' or one-vs-rest, is applied to each class and then the data is predicted according to the best value. This means that logistic regression would need a linear discriminanting boundary from each case vs every other and such linear

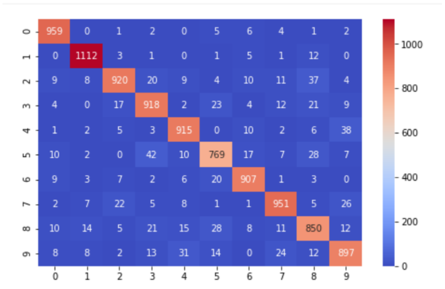


```
classification report :
              precision    recall  f1-score   support

           0       0.95      0.98      0.96       980
           1       0.96      0.98      0.97      1135
           2       0.94      0.89      0.91      1032
           3       0.89      0.91      0.90      1010
           4       0.92      0.93      0.93       982
           5       0.89      0.86      0.88       892
           6       0.94      0.95      0.94       958
           7       0.93      0.93      0.93      1028
           8       0.87      0.87      0.87       974
           9       0.90      0.89      0.90      1009

    accuracy                           0.92     10000
   macro avg       0.92      0.92      0.92     10000
weighted avg       0.92      0.92      0.92     10000
```

Figure 12: Classification Report



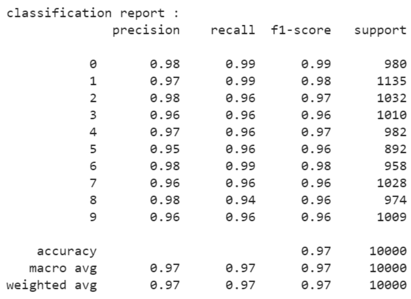
	0	1	2	3	4	5	6	7	8	9
0	959	0	1	2	0	5	6	4	1	2
1	0	1112	3	1	0	1	5	1	12	0
2	9	8	920	20	9	4	10	11	37	4
3	4	0	17	918	2	23	4	12	21	9
4	1	2	5	3	915	0	10	2	6	38
5	10	2	0	42	10	769	17	7	28	7
6	9	3	7	2	6	20	907	1	3	0
7	2	7	22	5	8	1	1	951	5	26
8	10	14	5	21	15	28	8	11	850	12
9	8	8	2	13	31	14	0	24	12	897

Figure 13: Confusion Matrix

boundary may not be possible or may not be able to accurately discriminate between every item in the dataset.

5.3 K Nearest Neighbour

K-Nearest Neighbour approach is a supervised classification approach. Under this, when we have to process a new test sample point, it is assigned a label closest to the training sample data mix which it is closest to in the original dimensional state space. The variable K indicates how many training data samples the model will verify out before assigning the label to the unseen new test sample point. After testing out on various efficient values of K for maximum efficiency, the maximum test accuracy was reported at K = 1. **Model accuracy achieved : 96.91%**.

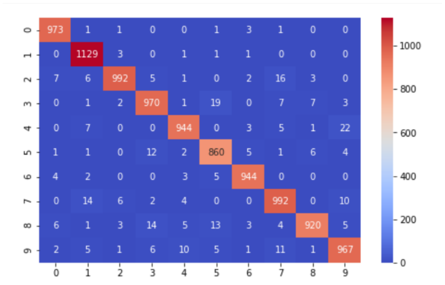


```
classification report :
              precision    recall  f1-score   support

           0       0.98      0.99      0.99       980
           1       0.97      0.99      0.98      1135
           2       0.98      0.96      0.97      1032
           3       0.96      0.96      0.96      1010
           4       0.97      0.96      0.97       982
           5       0.95      0.96      0.96       892
           6       0.98      0.99      0.98       958
           7       0.96      0.96      0.96      1028
           8       0.98      0.94      0.96       974
           9       0.96      0.96      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000
```

Figure 14: Classification Report



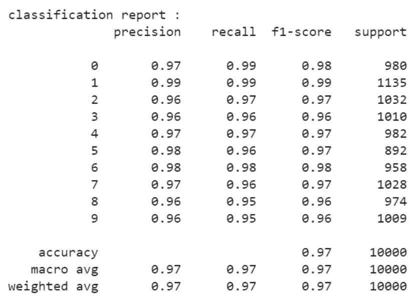
	0	1	2	3	4	5	6	7	8	9
0	973	1	1	0	0	1	3	1	0	0
1	0	1129	3	0	1	1	1	0	0	0
2	7	6	992	5	1	0	2	16	3	0
3	0	1	2	970	1	19	0	7	7	3
4	0	7	0	0	944	0	3	5	1	22
5	1	1	0	12	2	860	5	1	6	4
6	4	2	0	0	3	5	944	0	0	0
7	0	14	6	2	4	0	0	992	0	10
8	6	1	3	14	5	13	3	4	920	5
9	2	5	1	6	10	5	1	11	1	967

Figure 15: Confusion Matrix

Observation : K-NN predicts the data according to its nearest neighbour in the 784 (or 28x28) dimensions. And as we have seen clustering of similar digit or class on dimensionality reduction using PCA and t-SNE, we know that similar data are likely to be closer to each other in the high dimensions, euclidean distance wise. Therefore K-NN is expected to perform well on the MNIST dataset. The K-NN model is able to further reduce the test error rate to 3.09%.

5.4 Random Forrest

Random Forrest (RF) is an another supervised way of classification. Random Forrest are ensemble learning models and uses entropy to measure the quality of a particular class before refining it further. Random forest are basically an ensemble of several decision trees. Random Forrest are proven useful in overcoming overfitting. We can represent Random Forrest through instance bagging and feature bagging. Sklearn's RandomForrestClassifier() is a class capable of performing multi-class classification on a dataset. **Model accuracy achieved : 97.04%**.



```
classification report :
               precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.96      0.97      0.97      1032
           3       0.96      0.96      0.96      1010
           4       0.97      0.97      0.97       982
           5       0.98      0.96      0.97       892
           6       0.98      0.98      0.98       958
           7       0.97      0.96      0.97      1028
           8       0.96      0.95      0.96       974
           9       0.96      0.95      0.96      1009

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000
```

Figure 16: Classification Report

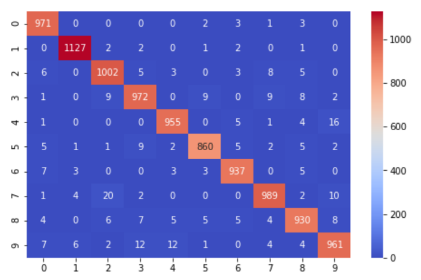
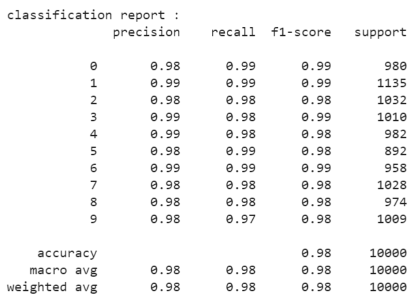


Figure 17: Confusion Matrix

Observation : The test error rate of random forest is 2.96%. Random Forrest are statistical models. The main advantage of the random forest approach is that it helps in reducing the variance of the model and thereby making the model less prone to overfitting.

5.5 Support Vector Machines

SVM employ kernel methods for efficient classification. The 'kernel trick' is to move the non-classifiable data to the higher dimensional latent space and make it classifiable. Kernel basis functions are used for converting data point to the latent space. We have used **rbf** kernel for our classification. The regularization parameter with value range between 100-200 is able to give high test accuracy of near about 98-99%. **Model accuracy achieved : 98.43%**.



```
classification report :
               precision    recall  f1-score   support

           0       0.98      0.99      0.99       980
           1       0.99      0.99      0.99      1135
           2       0.98      0.98      0.98      1032
           3       0.99      0.98      0.99      1010
           4       0.99      0.98      0.98       982
           5       0.98      0.99      0.98       892
           6       0.99      0.99      0.99       958
           7       0.98      0.98      0.98      1028
           8       0.98      0.98      0.98       974
           9       0.98      0.97      0.98      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000
```

Figure 18: Classification Report

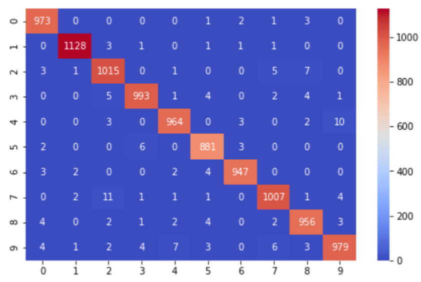
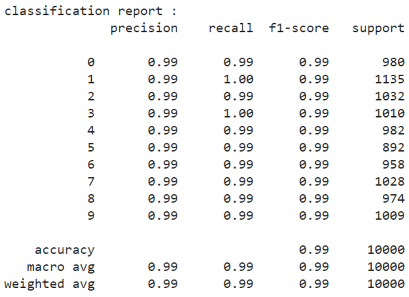


Figure 19: Confusion Matrix

Observation : SVM separates the data, class wise using OVR policy, in either the same dimensions as the input if possible or projects the data onto a higher dimension space where discriminating between the classes is more accurate. This helps in its high performance of classification. That is why, with SVM, we are able to achieve quite low test error rate of 1.57%.

5.6 Deep Learning Model

Deep Learning is shown to produce unprecedented performance on the MNIST dataset. So, for completeness, we have employed a Convolutional Neural Network with Maxpooling and Dropout which give an accuracy of 99.1%. Among all the implemented algorithms, CNN model gave the lowest test error rate of 0.85%. **Model accuracy achieved : 99.15%**.

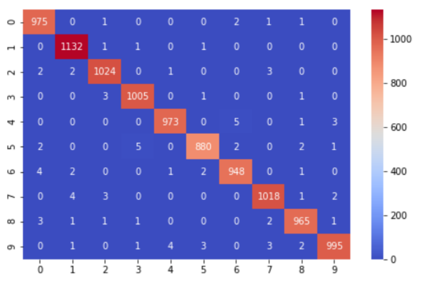


```
classification report :
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       980
           1       0.99      1.00      0.99      1135
           2       0.99      0.99      0.99      1032
           3       0.99      1.00      0.99      1010
           4       0.99      0.99      0.99       982
           5       0.99      0.99      0.99       892
           6       0.99      0.99      0.99       958
           7       0.99      0.99      0.99      1028
           8       0.99      0.99      0.99       974
           9       0.99      0.99      0.99      1009

    accuracy                           0.99     10000
   macro avg       0.99      0.99      0.99     10000
weighted avg       0.99      0.99      0.99     10000
```

Figure 20: Classification Report



	0	1	2	3	4	5	6	7	8	9
0	975	0	1	0	0	0	2	1	1	0
1	0	1132	1	1	0	1	0	0	0	0
2	2	2	1024	0	1	0	0	3	0	0
3	0	0	3	1005	0	1	0	0	1	0
4	0	0	0	0	973	0	5	0	1	3
5	2	0	0	5	0	880	2	0	2	1
6	4	2	0	0	1	2	948	0	1	0
7	0	4	3	0	0	0	0	1018	1	2
8	3	1	1	1	0	0	0	2	965	1
9	0	1	0	1	4	3	0	3	2	995

Figure 21: Confusion Matrix

6 Result

The comparison of the implemented algorithms is plotted below. Amongst all the implemented algorithms, SVM and CNN models are performing best. The SVM model is giving the test error rate of 1.57% whereas the test error rate of the CNN model is 0.85%.

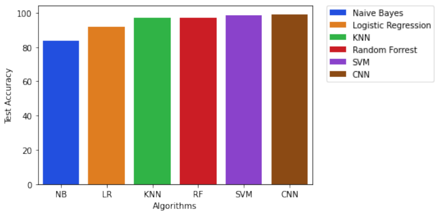


Figure 22: Test Accuracy

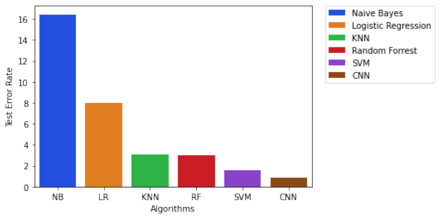


Figure 23: Test Error Rate

7 Conclusion

This project implements and analyse state-of-the-art machine learning models which are quite popular in the Data Science community. The project aims to utilize key aspects of data science to a real world problem of analysing handwritten digit images. The approach to implement the project was to first execute Exploratory Data Analysis of the data. The EDA of the MNIST revealed how different machine learning models will behave on this dataset. It was followed by experimental analysis of the dataset by implementing six different algorithms and comparing them on the basis of test accuracy metric and test error metric.

References

[1] Saeed Al Mansoori. Intelligent handwritten digit recognition using artificial neural network, 06 2015.

[2] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, 2013.

[3] T. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma. Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing*, 24(12): 5017–5032, 2015.

[4] Aysegul Dundar, Jonghoon Jin, and Eugenio Culurciello. Convolutional clustering for unsupervised learning. *CoRR*, abs/1511.06241, 2015. URL http://arxiv.org/abs/1511.06241.

[5] Hao Zhang, A. C. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2126–2136, 2006.

[6] Parveen Kumar, Nitin Sharma, and Arun Rana. Article: Handwritten character recognition using different kernel based svm classifier and mlp neural network (a comparison). *International Journal of Computer Applications*, 53(11):25–31, September 2012. Full text available.

[7] Kai Labusch, Erhardt Barth, and Thomas Martinetz. Simple method for high-performance digit recognition based on sparse coding. *IEEE Transactions on Neural Networks*, 19:1985–1989, 2008.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[9] Xiao-Xiao Niu and Ching Y Suen. A novel hybrid cnn–svm classifier for recognizing handwritten digits. *Pattern Recognition*, 45(4):1318–1325, 2012.

[10] M. Wu and Z. Zhang. Handwritten digit classification using the mnist data set. 2019.