MACHINE LEARNING ASSIGNMENT 2

Birla Institute of Technology and Science,
Hyderabad

Kush Desai(2021B4A73158H)
Manav Sahni(2021A8PS2199H)
Om Narayan(2021A7PS1324H)

May 1, 2023

Introduction

In this report, our group has trained machine learning models using several supervised learning techniques in order to predict the outputs for the problem statement. These machine-learning approaches are providing good results for decision-making in academic fields. this report consists of several machine learning algorithms trained using some given data in order to predict (a) if the salary of an adult is more than $50K using Naïve Bayes classifier and (b) to recognise a handwritten digit in the mnist database using artificial neural network models. The report is divided into 2 parts: the first part includes the description of the naïve bayes learning algorithm used on the adult database. The second part consists of the explanation for the neural network used for the mnist database, with a comparative study of the fifteen trained models.

General Procedure

Although every algorithm is different from one another, there are a few steps that are common while performing any machine-learning algorithm, which are as follows:

Feature engineering (optional): Feature engineering refers to manipulation — addition, deletion, combination, mutation — of your data set to improve machine learning model training, leading to better performance and greater accuracy.

Train test split: we are a set of data, a portion of which is used to train our machine learning model, and once the model is finalized, the remaining data called the testing data, is used to test how accurate our trained machine learning model actually is.

A model with a fit and predict functions: this includes the implementation of the machine learning algorithm. The fit function is used to train the classifier, while the predict function is used to predict the output for new inputs once the model is trained.

Classifier: A classifier in machine learning is an algorithm that automatically orders or categorizes data into one or more of a set of “classes.”

Calculation of accuracy: the final step in our techniques is to calculate how accurate our model is for a new set of training data.

Part A: Performing naïve Bayes algorithm on adult database

The Naïve Bayes algorithm is a supervised learning algorithm based on the Bayes theorem and used for solving classification problems. It predicts on the basis of the probability of an object. Bayes' theorem is used to determine the probability of a hypothesis with prior knowledge. This algorithm works on the concept of conditional probability.

The formula for Bayes' theorem is given as:

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

Where,

P(A|B) is Posterior probability: Probability of hypothesis A, given the event B.

P(B|A) is Likelihood probability: The probability of the event B given that the probability of hypothesis A is true.

P(A) is Prior Probability: The probability of the hypothesis before observing the evidence.

P(B): Probability of Event B.

We use the naïve Bayes classifier on the adult database to check if the individual earns more than $50k annually.

We calculate the prior probability of each class in the training dataset and the conditional probability of each feature of each class in the dataset. We also predict the class of a given instance using the classifier we built and calculate the accuracy of the classifier on the testing set. We have given the results of the same below…

Results:

The accuracy of our Naïve Bayes classifier is 0.7832, precision is 0.6452, recall is 0.3045, F-1 score is 0.4138.

Confusion matrix:

[[822, 452]

[1877, 7594]]

Part B: Training a neural network with the MNIST database to recognize handwritten digits

A neural network is a machine learning algorithm that implements the functioning of the human brain into a network of differentially activated “neurons” having activations between 0 and 1, which guide the probability of classification. It tries to achieve this by minimizing a “cost function” to its local minima using the gradient descent algorithm so that we have the minimum error in the training examples.

We have trained 15 different models using this technique, and here are the accuracy scores we managed to get on the testing examples using those models:

Model 1:

Activation function: ReLU

No. of hidden layers: 2

No. of neurons: 100

Layer-wise split: 40/60

Accuracy: 96.86%

Confusion matrix:

```
[[ 968    0    1    0    1    1    6    0    3    0]
 [   0 1119    5    1    0    1    3    0    6    0]
 [   3    1  997    4    3    3    4    7   10    0]
 [   0    1    5  960    0   18    1    6   13    6]
 [   0    2    4    0  945    1    6    2    6   16]
 [   2    0    0    9    0  866    5    1    6    3]
 [   6    2    2    1    7   10  924    0    6    0]
 [   1    6    7    1    0    0    0  999    8    6]
 [   1    2    2    4    3    8    0    3  950    1]
 [   2    2    0    9    9   13    1    6    9  958]]
```

Model 2:

Activation function: ReLU

No. of hidden layers: 2

No. of neurons: 100

Layer-wise split: 50/50

Accuracy: 97.13%

Confusion matrix:

```
[[ 971    0    2    0    0    1    3    0    2    1]
 [   0 1120    4    1    0    1    3    1    5    0]
 [   4    0 1012    3    1    0    2    4    6    0]
 [   1    1    4  984    1    7    0    2    9    1]
 [   1    3    5    2  949    0    2    3    2   15]
 [   4    0    0   23    1  842    8    1   10    3]
 [   4    4    1    1    2    2  940    0    4    0]
 [   1    9   13    4    0    2    0  995    3    1]
 [   5    5    6   10    5    3    1    2  934    3]
 [   3    4    1    8   11    3    0    4    9  966]]
```

Model 3:

Activation function: ReLU

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 50/50/50

Accuracy: 97.34%

Confusion matrix:

```
[[ 970    0    0    2    0    0    2    0    4    2]
 [   0 1122    4    2    0    0    3    0    4    0]
 [   3    2 1000   13    2    0    3    6    3    0]
 [   0    1    3  983    0   13    0    2    5    3]
 [   2    0    1    0  962    0    4    3    1    9]
 [   3    0    0   14    0  864    4    0    6    1]
 [   7    2    3    1    3    6  935    0    1    0]
 [   3    3    9    3    1    1    0  997    4    7]
 [   2    1    4    4    5    7    1    4  942    4]
 [   3    4    0   12   11    7    1    4    8  959]]
```

Model 4:

Activation function: ReLU

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 70/50/30

Accuracy: 97.36%

Confusion matrix:

```
[[ 970    0    0    1    0    1    3    2    2    1]
 [   0 1115    3    1    1    2    3    2    8    0]
 [   4    2  995    8    4    1    4    6    8    0]
 [   0    0    3  991    0    6    0    3    2    5]
 [   1    0    4    1  957    1    2    0    2   14]
 [   2    0    0   17    0  863    2    1    5    2]
 [   3    2    1    0    2    7  938    0    4    1]
 [   2    2    7    4    0    0    0 1006    2    5]
 [   2    1    6   11    5    6    0    4  933    6]
 [   1    2    1   13   10    7    0    6    1  968]]
```

Model 5:

Activation function: ReLU

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 30/40/80

Accuracy: 96.83%

Confusion matrix:

```
[[ 968    0    2    0    1    4    2    1    1    1]
 [   1 1122    5    0    1    0    2    2    2    0]
 [   5    0 1003    6    1    1    3    8    5    0]
 [   0    1    7  977    0    7    0    5    8    5]
 [   1    0    6    0  953    1    6    1    0   14]
 [   3    0    1   20    1  849   10    1    3    4]
 [   9    4    3    0    8    9  924    0    1    0]
 [   2    2   16    3    2    0    0  995    4    4]
 [   6    1    7    9    6    6    4    4  927    4]
 [   2    3    1    6   10    8    0    5    9  965]]
```

Model 6:

Activation function: sigmoid

No. of hidden layers: 2

No. of neurons: 100

Layer-wise split: 60/40

Accuracy: 97.07%

Confusion matrix:

```
[[ 969    0    2    1    0    4    1    2    1    0]
 [   0 1118    4    2    0    0    5    1    5    0]
 [   2    0 1009    4    1    1    2    8    5    0]
 [   0    0   12  972    0    8    1    9    4    4]
 [   1    1    6    0  957    0    3    3    4    7]
 [   3    0    0   17    3  850    6    5    7    1]
 [   3    3    7    1    8    4  928    0    4    0]
 [   2    2   10    5    1    0    0 1001    2    5]
 [   3    3    7    7    3    5    1    4  935    6]
 [   2    3    1    6   12    8    0    4    5  968]]
```

Model 7:

Activation function: sigmoid

No. of hidden layers: 2

No. of neurons: 100

Layer-wise split: 50/50

Accuracy: 97.02%

Confusion matrix:

```
[[ 965    1    4    1    0    2    4    1    1    1]
 [   0 1120    6    2    0    0    2    0    5    0]
 [   6    2 1000    8    1    0    1    8    5    1]
 [   0    0    9  985    1    5    1    3    3    3]
 [   0    0    8    0  954    2    3    2    1   12]
 [   4    0    2   15    4  852    5    1    4    5]
 [   5    3    5    1    4    6  932    0    2    0]
 [   1    5   10    4    1    0    1  999    2    5]
 [   5    2    4    9    4    7    3    4  929    7]
 [   2    5    0    7   15    7    0    4    3  966]]
```

Model 8:

Activation function: sigmoid

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 50/50/50

Accuracy: 96.9%

Confusion matrix:

```
[[ 968    0    2    1    1    1    2    0    1    4]
 [   0 1116    7    1    0    0    3    1    6    1]
 [   4    1 1000   10    4    1    1    7    4    0]
 [   1    1    7  977    0   10    0    4    1    9]
 [   1    1    4    1  949    0    9    4    2   11]
 [   3    1    2   14    3  856    2    5    3    3]
 [   8    4    3    2    1    7  931    0    2    0]
 [   2    5   13    6    1    0    0  995    0    6]
 [   3    2    1   10    1    6    3    4  939    5]
 [   2    3    2    9   16    4    0   10    4  959]]
```

Model 9:

Activation function: sigmoid

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 70/50/30

Accuracy: 97.27%

Confusion matrix:

```
[[ 965    0    1    1    1    4    4    1    1    2]
 [   0 1123    4    0    0    0    3    0    5    0]
 [   5    1  994   10    5    1    2    8    6    0]
 [   0    2    4  980    0   12    1    6    4    1]
 [   0    0    3    1  957    0    7    1    3   10]
 [   4    0    0    9    0  865    7    0    2    5]
 [   4    2    0    0    3    8  940    0    1    0]
 [   0    4    9    9    3    0    0  997    2    4]
 [   1    2    4    3    3    8    4    3  940    6]
 [   0    3    0   11    9    6    2    6    6  966]]
```

Model 10:

Activation function: sigmoid

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 30/40/80

Accuracy: 96.06%

Confusion matrix:

```
[[ 963    0    3    2    0    2    5    0    2    3]
 [   0 1114    5    2    0    2    3    3    6    0]
 [   5    1  984   12    7    0    3   13    7    0]
 [   2    3    1  967    0   13    0    6   14    4]
 [   1    1    5    1  948    0    2    5    3   16]
 [   2    0    1   21    1  840    7    3   10    7]
 [   9    3    2    0    7    6  930    0    1    0]
 [   0    3   16    9    5    2    0  979    2   12]
 [   4    2    5    6    9   11    1    3  927    6]
 [   5    5    0    7   19    4    1    5    9  954]]
```

Model 11:

Activation function: tanh

No. of hidden layers: 2

No. of neurons: 100

Layer-wise split: 60/40

Accuracy: 96.83%

Confusion matrix:

```
[[ 958    0    2    1    2    3    5    3    5    1]
 [   0 1106    5    3    2    0    4    3    8    4]
 [   2    0  993   14    2    0    3    7   11    0]
 [   0    0    5  984    0    6    0    7    6    2]
 [   0    2    5    1  953    1    4    1    0   15]
 [   3    0    1   17    1  849    9    1    7    4]
 [   5    2    0    0    7    6  933    0    5    0]
 [   2    2   11    7    3    0    0 1001    0    2]
 [   1    0    5   13    8    4    1    3  934    5]
 [   0    4    0    6   11    3    0    8    5  972]]
```

Model 12:

Activation function: tanh

No. of hidden layers: 2

No. of neurons: 100

Layer-wise split: 50/50

Accuracy: 96.85%

Confusion matrix:

```
[[ 961    0    0    0    2    2    7    3    2    3]
 [   0 1117    5    0    0    2    2    2    6    1]
 [   4    1  997    4    4    0    4    6   12    0]
 [   0    2    6  977    0    8    1    2    9    5]
 [   0    0    1    1  963    0    5    2    3    7]
 [   3    1    1   16    0  849    9    1   10    2]
 [   9    2    1    1    2    2  935    0    6    0]
 [   2    2    7   10    3    1    0  990    5    8]
 [   2    2    5    7    4    8    2    3  937    4]
 [   3    7    0    8   19    5    0    6    2  959]]
```

Model 13:

Activation function: tanh

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 50/50/50

Accuracy: 96.88%

Confusion matrix:

```
[[ 967    0    1    0    1    1    5    1    3    1]
 [   0 1116    5    1    0    1    4    3    5    0]
 [   5    0  994    5    4    2    5   11    6    0]
 [   0    1    5  975    1    9    1    6    4    8]
 [   2    0    3    1  952    2    6    1    1   14]
 [   2    0    0   16    0  860    6    0    4    4]
 [   5    3    2    1    1    8  936    0    2    0]
 [   3    2   13    4    4    0    0  988    2   12]
 [   4    2    5    5    6    7    5    5  928    7]
 [   1    4    2    7    8    5    0    3    7  972]]
```

Model 14:

Activation function: tanh

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 70/50/30

Accuracy: 96.89%

Confusion matrix:

```
[[ 966    0    3    0    1    2    3    3    2    0]
 [   0 1121    4    3    0    0    3    0    3    1]
 [   1    1 1004   11    2    0    1    4    7    1]
 [   0    0    4  975    1    5    1    7   10    7]
 [   1    0    6    0  947    0    4    5    1   18]
 [   2    1    0   20    0  844   10    1    7    7]
 [   4    2    2    2    3    7  934    0    4    0]
 [   0    7   10    4    3    2    1  993    3    5]
 [   5    3    2    9    6    4    2    4  934    5]
 [   1    4    1    5    7    3    1    9    7  971]]
```

Model 15:

Activation function: tanh

No. of hidden layers: 3

No. of neurons: 150

Layer-wise split: 30/40/80

Accuracy: 96.39%

Confusion matrix:

```
[[ 963    1    1    1    1    0    8    1    2    2]
 [   0 1117    6    1    2    0    4    0    5    0]
 [   5    4  997    2    3    1    1   13    6    0]
 [   0    3    7  978    0    8    0    7    4    3]
 [   1    0   16    0  944    0    1    5    4   11]
 [   3    1    3   21    1  845    5    1    9    3]
 [   5    6   10    0    3    8  921    0    5    0]
 [   2    3   11    7    2    0    0  991    4    8]
 [   5    4   14    7    6   10    1    6  916    5]
 [   3    2    2    5   12    4    0    8    6  967]]
```

Comparative Study:

From the above 15 models, we conclude that the ReLU function gives the best test accuracy (97.36%), followed by the sigmoid function (97.27%), followed by the tanh function (96.89%).

We tested 5 models for each of the three activation functions. We then tested 2 out of 5 models for 100 neurons in 2 hidden layers and the other 3 out of 5 models for 150 neurons in 3 hidden layers respectively. We have also displayed the confusion matrix of each model above.

The other classifiers do not show a statistically significant difference from the best classifier in terms of performance as the task was simple, since we just had 100-150 neurons to train with. Hence, the training complexity was lesser as compared to what it would have been for 10000 neurons with more than 2/3 hidden layers. The lowest accuracy is 96.06% and the highest accuracy is 97.36%. Even though, we have taken 3 different activation functions, the test accuracy is not varying much. This is because the task is simple.