APPLIANCES ENERGY PREDICTION & BANK TERM LOAN DEPOSIT PREDICTION USING ARTIFICIAL NEURAL NETWORKS & K – NEAREST NEIGHBOURS

SUDHARSANA RAJASEKARAN

APPLIED MACHINE LEARNING

Assignment 3

Table of Contents

ARTIFICIAL NEURAL NETWORKS

Varying Mini Batch Sizes

By varying the size of the mini-batches that each epoch takes we can achieve different levels of test and train accuracy. Here in the graphs, you will be able to see that the as the minibatch sizes increases the test and train accuracies also increase till a certain mini-batch size and decreases as the minibatch size increases beyond a certain point. This is because for smaller minibatch size, the step taken towards the global minimum is small and as the size of the minibatch size increases the step towards the global minimum increases leading better to the global minimum. On the other hand, with the larger mini-batch size the step towards global minimum goes beyond the global minimum point thus not giving us the right global optimal solution. Thus, taking the right size of minibatch is important towards building a good Neural Network. Here in the below two models, mini-batch size of 500-1000 is optimal for better model performance.

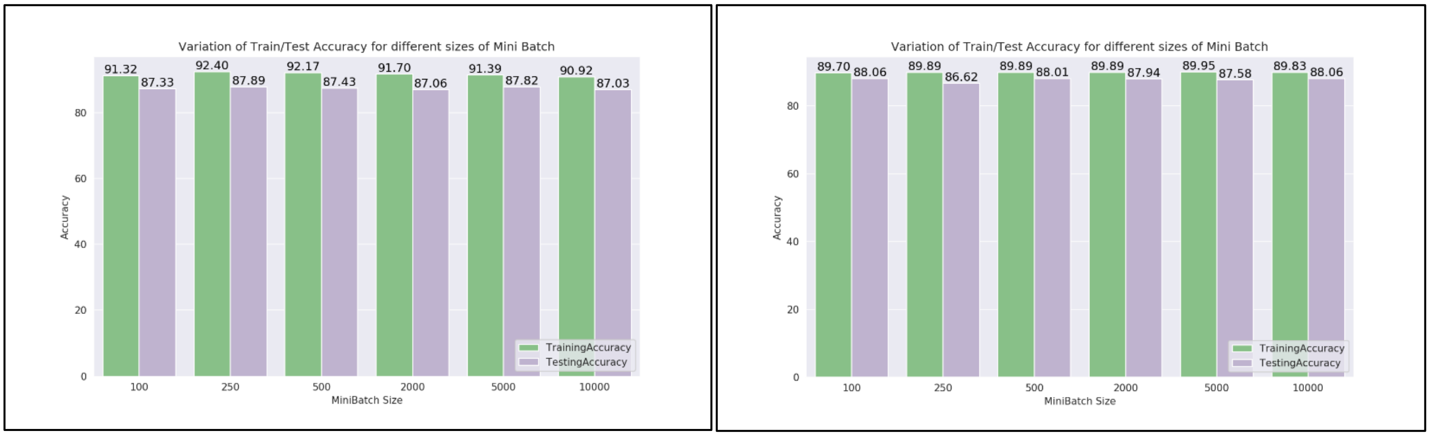


Figure 1. Variation of Train/Test accuracy for different mini batch for both the datasets

It can be observed that as the mini-batch sizes increase cost also increases. This is because with smaller mini-batches the number of steps taken for each epoch is higher meaning faster convergence towards global minimum. Also, the with smaller mini-batch sizes we can reach as closer to a global minimum as possible. Hence choosing the mini-batch size is essentially important towards the attaining global minimum. Here in the below two models, mini-batch size of 500-1000 is optimal for better model performance.

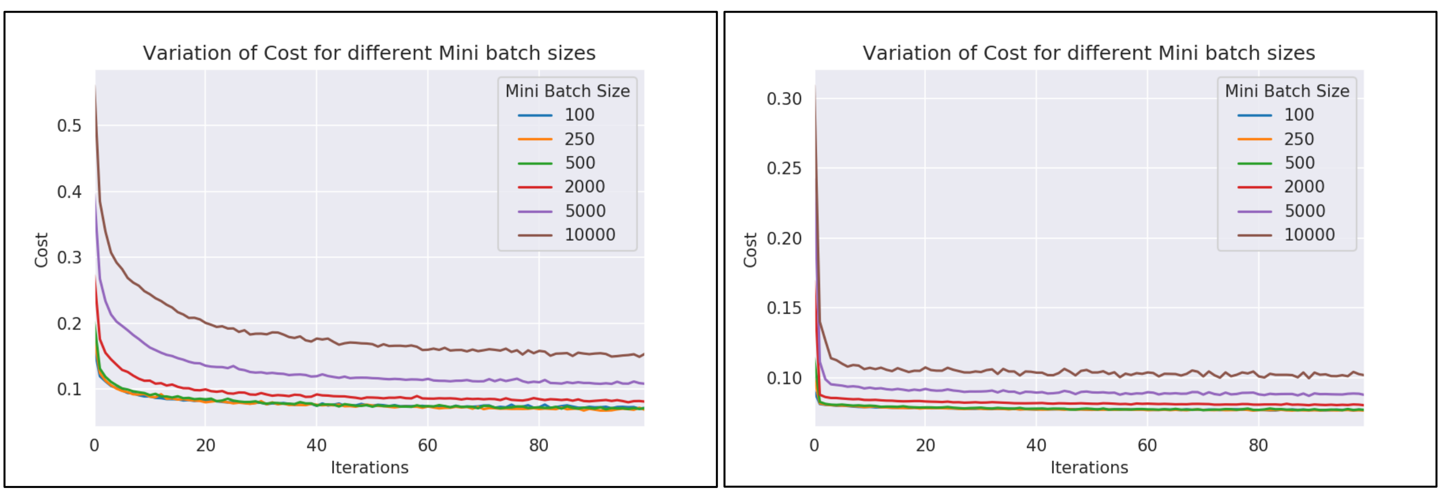


Figure 2. Variation of cost Vs iterations for different mini batch for both the datasets

Varying Hidden Layer width

Width of the hidden layer plays a very important role in deciding the output performance of the neural network. Here from the neural network, we will be able to see that larger the width of each of the neural network better the performance of the model. But it can also be observed that the beyond a certain width of the neural network the performance of the neural network degrades. Thus, it is essential to determine the width of each of the hidden layers of the neural network. Here in the below two models, hidden layer width of 200, 100, 50 has better performance over other hidden layer widths of the neural network.

	Layer number	i/p	HL1	HL2	HL3	o/p	Accuracy	Accuracy_val
0	1	27	20	10	5	1	TrainingAccuracy	91.15
1	2	27	200	100	50	1	TrainingAccuracy	97.61
2	3	27	500	250	100	1	TrainingAccuracy	73.56
3	4	27	1000	500	250	1	TrainingAccuracy	73.56
0	1	27	20	10	5	1	TestingAccuracy	87.28
1	2	27	200	100	50	1	TestingAccuracy	87.52
2	3	27	500	250	100	1	TestingAccuracy	73.69
3	4	27	1000	500	250	1	TestingAccuracy	73.69

	Layer number	i/p	HL1	HL2	HL3	o/p	Accuracy	Accuracy_val
0	1	13	20	10	5	1	TrainingAccuracy	90.13
1	2	13	200	100	50	1	TrainingAccuracy	91.01
2	3	13	500	250	100	1	TrainingAccuracy	88.28
3	4	13	1000	500	250	1	TrainingAccuracy	88.28
0	1	13	20	10	5	1	TestingAccuracy	86.60
1	2	13	200	100	50	1	TestingAccuracy	88.35
2	3	13	500	250	100	1	TestingAccuracy	88.36
3	4	13	1000	500	250	1	TestingAccuracy	88.36

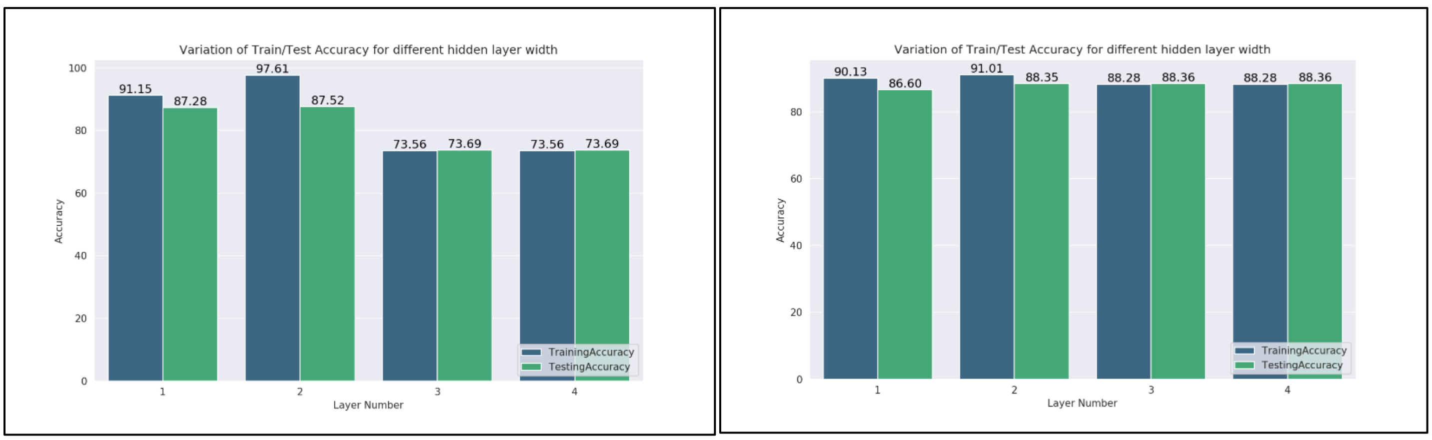


Figure 3. Variation of Train/Test accuracy for different hidden layer width for both the datasets

Varying Number of Hidden Layers

Deeper the neural network, better the performance of the network, here in the below two models the number of hidden layers increases the test and training accuracies also increases. Therefore, the neural network needs to have deeper layers for better performance. But deeper the network, the more the computational expensive to update the model parameters. Therefore, the number of hidden layers is a tradeoff between performance and the computational performance of the system.

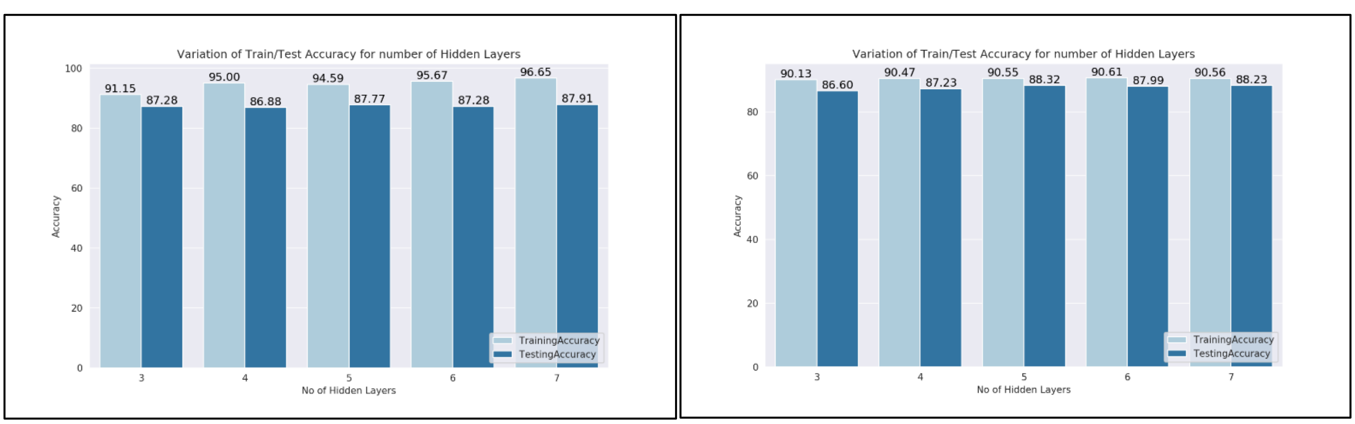


Figure 4. Variation of Train/Test accuracy for different hidden layers for both the datasets

Varying Number of Epochs

With the increasing number of epochs increases the accuracy of the model. This is because with each epoch the model takes one more extra towards the global minimum. Therefore, increasing the number of epochs is a good way to increase the model performance. But increasing the number of epochs is computationally expensive and the model takes more time to train. Hence, increasing the number of epochs is a tradeoff between model performance and the system capability to run the model. Here in the below two models, it can be observed that a model trained for 3000 epochs has a better say over the final performance of the neural network than the ones below it.

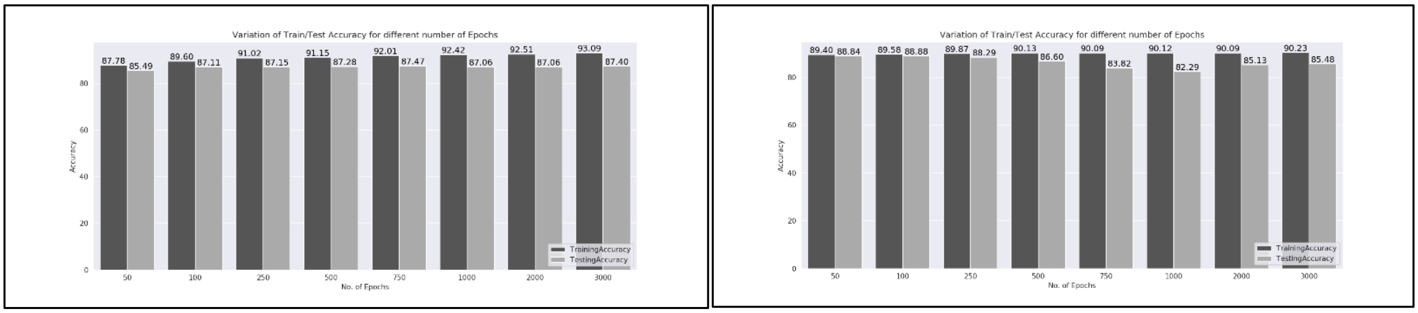


Figure 5. Variation of Train/Test accuracy for different epochs for both the datasets

Different Learning Rates

Learning rate is a very important tuning parameter in any model. This is primarily because it measures the step size taken for each epoch. The tuning learning rate is essential, as the step size is large the model will never converge to the global minimum and will oscillate around it, on the other hand, smaller the step size it takes infinitely small steps and it will take forever to converge. Here in the below two models, the model performance increases as the learning rate decrease but it stops after some rate, here it stops after 0.005. Therefore any learning rate less than that decreases the performance of the final model.

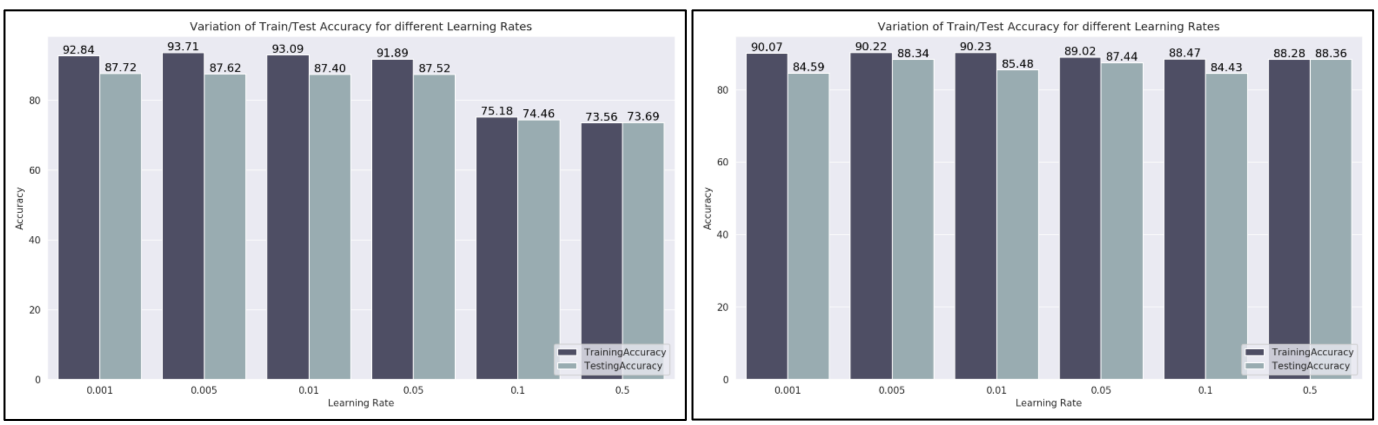
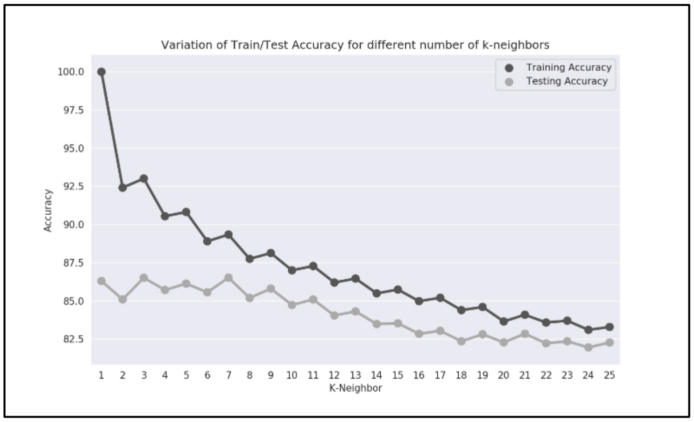


Figure 6. Variation of Train/Test accuracy for different learning rates for both the datasets

KNN

Varying Number of Nearest Neighbor

Number of k-nearest neighbor is an important tuning parameter of the KNN algorithm. Ran the model for different k-nearest neighbors and measured the train and test accuracies. It can be observed from the below graphs that dataset 1 has better performance 3 nearest neighbors and dataset 2 has better training accuracy for a 2 nearest neighbors.



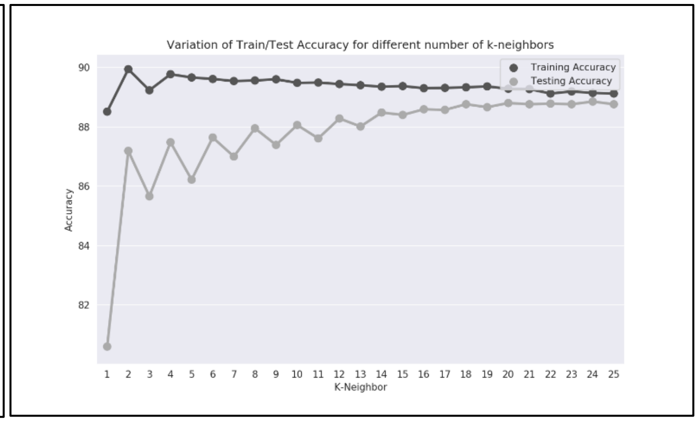
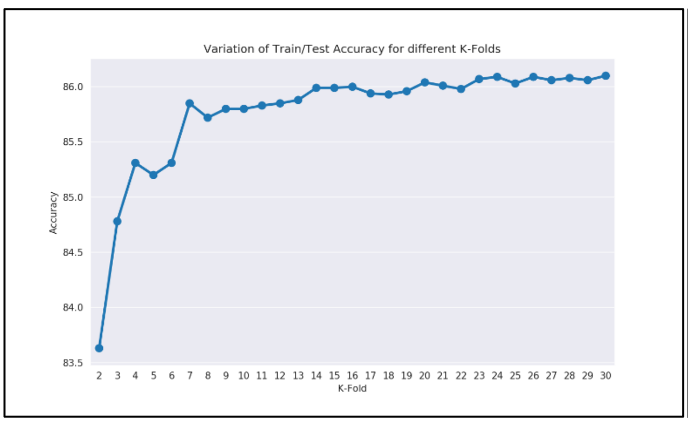


Figure 7. Variation of Train/Test accuracy for different k-nearest neighbors for both the datasets

Varying Number of K-Folds in Cross Validation

Varying the number of folds for cross validation has an important metric in building a robust machine learning model. Here it can be observed from the dataset 1 that as the number of folds increases the accuracy of the model also increases. Incase of dataset 2 has peak performance for 15 folds in the dataset. Hence number of folds is important in measuring the performance of the model.



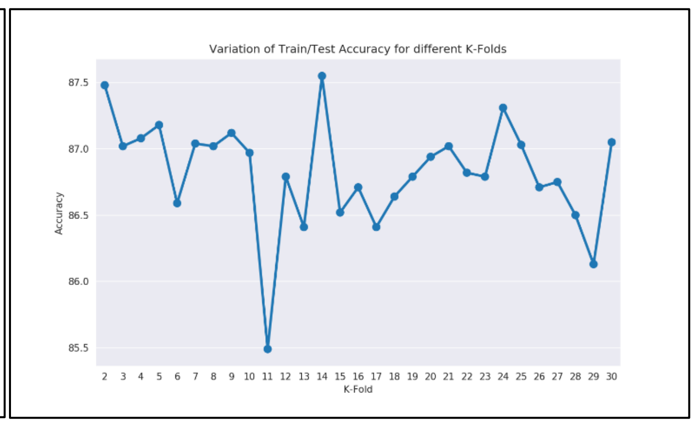


Figure 8. Variation of Train/Test accuracy for different k-folds validation for both the dataset

Cross Validated K-Nearest Neighbor

A Grid search cross validated k-nearest search is performed to find the model that gives best parameters. It can be observed that the dataset 1 has a maximum accuracy of 85.8% for 3 nearest neighbors and dataset 2 has a accuracy of 88.75% for a 16 nearest neighbors.

```
Cross Validated K-Nearest Neighbour
Accuracy: 85.8%
Number of k-Nearest Neighbor: {'n_neighbors': 3}
```

```
Cross Validated K-Nearest Neighbour
Accuracy: 88.75%
Number of k-Nearest Neighbor: {'n_neighbors': 16}
```

Figure 9. Cross Validated k-nearest neighbor for both the datasets

Cross Validated K-Nearest Neighbor and Distance

A Grid search cross validated k-nearest neighbor and distance search is performed to find the model that gives best parameters. It can be observed that the dataset 1 has a maximum accuracy of 86.4% for 4 nearest neighbors as compared to 3 nearest neighbors when just the nearest neighbors was the selection criteris and dataset 2 has a accuracy of 88.75% for a 16 nearest neighbors with a uniform search.

```
Cross Validated K-Nearest Neighbour and Distance
Accuracy: 86.4%
Number of k-Nearest Neighbor: {'n_neighbors': 4, 'weights': 'distance'}
```

```
Cross Validated K-Nearest Neighbour and Distance
Accuracy: 88.75%
Number of k-Nearest Neighbor: {'n_neighbors': 16, 'weights': 'uniform'}
```

Figure 10. Cross Validated k-nearest neighbor and weights for both the datasets

Comparison of performance between Hyperparameter Tuned ANN and KNN

After the fine tuning of ANN and KNN models, the hyperparameter tuned models were run. The result was that the ANN model outperformed KNN model in the dataset 1 and KNN outperformed ANN in dataset 2 in terms of testing accuracy. Therefore, whether it is KNN or ANN it all boils down to the dataset under consideration.

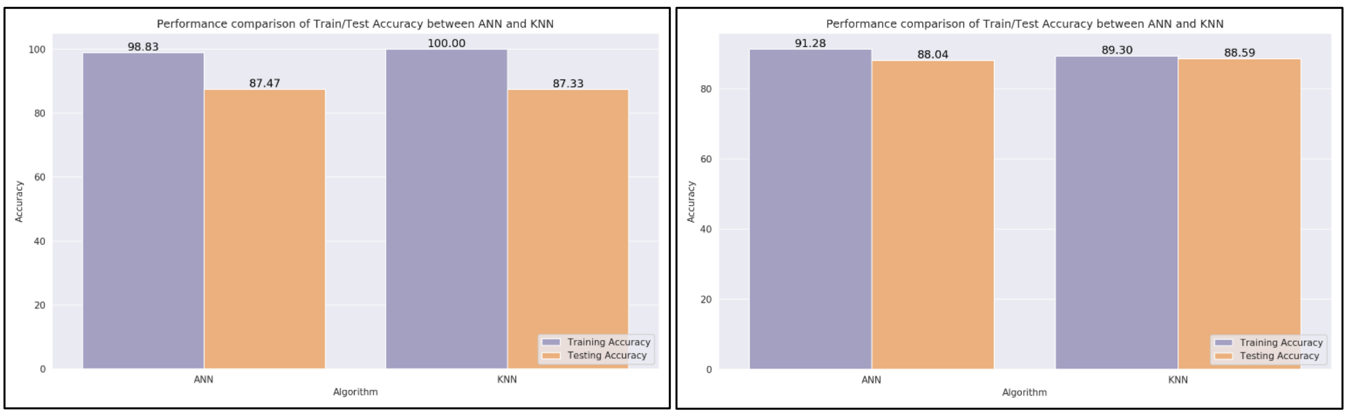


Figure 11. Comparison of Performance of model between ANN and KNN

Comparison of performance between different Machine Learning models

A person comparison of which of the machine learning models performs better on the dataset is carried out. It can be observed that the boosted decision tree out performs all other machine learning models in the dataset 1 and decision tree with no boosting performs the worst. In the dataset 2 Decision tree with no boosting performs the best and Ada boosted decision tree performs the worst. The boosted decision tree performing worst may be primarily because of overfitting in the test dataset, as decision trees are prone to overfitting.

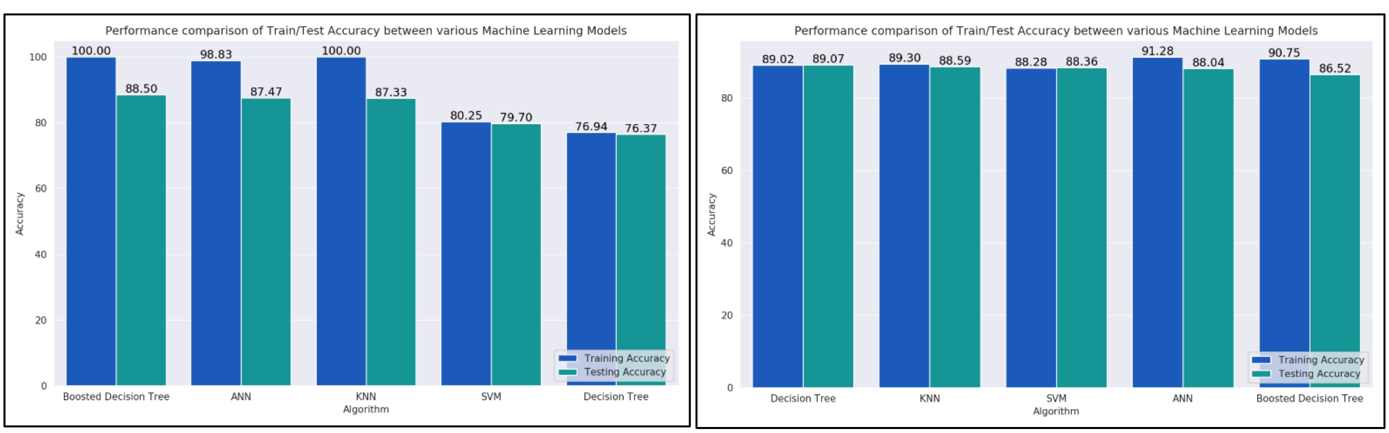


Figure 12. Comparison of Performance between different machine learning models

Appendix

Dataset 1 – Appliance Energy prediction

Dataset 2 – Bank Loan Term Deposit Prediction