PATTERN RECOGNITION HOMEWORK-2

Name: Anvesh Muppeda

R #11840667

Classical classifiers typically get the greatest results when given little datasets. However, their performance usually declines while working with larger datasets. However, when few datasets are used, deep learning algorithms often perform worse. Nevertheless, as dataset numbers increase, so does their performance. The experimental results are provided in terms of test accuracy to demonstrate performance.

Experiment: Performance of **Small datasets**

For smaller datasets:

Dataset Used: pima-indians-diabetes-dataset (768 datapoints)

Algorithms: Deep Learning and SVM

Observations:

With an accuracy of **80.72%**, SVM outperforms the Deep Learning method in this instance. The Deep Learning algorithm has an accuracy level **of 72.73%.**

When it comes to tiny datasets, SVM outperforms Deep Learning in terms of efficiency.

Screenshots:

Deep Learning:

```
anveshmuppeda@Anveshs-MacBook-Air small % python3 small-ds-dnn.py
/opt/homebrew/lib/python3.11/site-packages/keras/src/layers/core/dense.py:86: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 1s 665us/step - accuracy: 0.6487 - loss: 0.7305
Epoch 2/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 513us/step - accuracy: 0.6280 - loss: 0.6648
Epoch 3/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 484us/step - accuracy: 0.6603 - loss: 0.5964
Epoch 4/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 484us/step - accuracy: 0.6863 - loss: 0.5629
Epoch 5/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 485us/step - accuracy: 0.7433 - loss: 0.5456
Epoch 6/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 485us/step - accuracy: 0.7752 - loss: 0.4911
Epoch 7/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 480us/step - accuracy: 0.7702 - loss: 0.4875
Epoch 8/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 476us/step - accuracy: 0.7562 - loss: 0.4811
Epoch 9/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 483us/step - accuracy: 0.7497 - loss: 0.4852
Epoch 10/10
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 490us/step - accuracy: 0.7821 - loss: 0.4403
Test Accuracy: 0.7273
anveshmuppeda@Anveshs-MacBook-Air small %
```

SVM:

```
anveshmuppeda@Anveshs-MacBook-Air small % python3 small-ds-svm.py
Confusion Matrix is:
 [[120  11]
 [ 26  35]]
The accuracy of SVM is:  0.8072916666666666
Classification report is:
              precision    recall  f1-score   support

           0       0.82      0.92      0.87       131
           1       0.76      0.57      0.65        61

    accuracy                           0.81       192
   macro avg       0.79      0.74      0.76       192
weighted avg       0.80      0.81      0.80       192

anveshmuppeda@Anveshs-MacBook-Air small %
```

For larger datasets:

Dataset Used: churn-modelling (10001 datapoints)

Algorithms: Deep Learning and SVM

Observations:

In this case, Deep Learning outperforms SVM method with an accuracy of 85.92%. The SVM method has an accuracy rating of 79.64%.

Compared to SVM, Deep Learning is more effective with tiny datasets.

Screenshots:

Deep Learning:

```
anveshmuppeda@Anveshs-MacBook-Air final % python3 big-ds-dnn.py
/opt/homebrew/lib/python3.11/site-packages/keras/src/layers/core/dense.py:86: UserWarning: Do not pass an `input_shap
e`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first
layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 1s 557us/step - accuracy: 0.7402 - loss: 0.5717
Epoch 2/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 429us/step - accuracy: 0.8001 - loss: 0.4504
Epoch 3/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 437us/step - accuracy: 0.8213 - loss: 0.4187
Epoch 4/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 443us/step - accuracy: 0.8335 - loss: 0.3955
Epoch 5/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 437us/step - accuracy: 0.8475 - loss: 0.3771
Epoch 6/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 428us/step - accuracy: 0.8470 - loss: 0.3744
Epoch 7/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 439us/step - accuracy: 0.8579 - loss: 0.3562
Epoch 8/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 436us/step - accuracy: 0.8517 - loss: 0.3624
Epoch 9/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 440us/step - accuracy: 0.8576 - loss: 0.3503
Epoch 10/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 0s 440us/step - accuracy: 0.8497 - loss: 0.3676
Test Accuracy: 0.8592
anveshmuppeda@Anveshs-MacBook-Air final %
anveshmuppeda@Anveshs-MacBook-Air final %
```

SVM:

```
anveshmuppeda@Anveshs-MacBook-Air final % python3 big-ds-svm.py
/opt/homebrew/lib/python3.11/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning: Precision
 and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter
to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/homebrew/lib/python3.11/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning: Precision
 and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter
to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/homebrew/lib/python3.11/site-packages/sklearn/metrics/_classification.py:1469: UndefinedMetricWarning: Precision
 and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter
to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
Confusion matrix is:
 [[1991    0]
 [ 509    0]]
The accuracy of SVM is: 0.7964
Classification report is:
               precision    recall  f1-score   support

           0       0.80      1.00      0.89      1991
           1       0.00      0.00      0.00       509

    accuracy                           0.80      2500
   macro avg       0.40      0.50      0.44      2500
weighted avg       0.63      0.80      0.71      2500

anveshmuppeda@Anveshs-MacBook-Air final %
```

Analysis:

It may be deduced that classical classifiers tend to perform worse as dataset sizes grow.

Experiment: Robustness to Overfitting

Dataset: pima-indians-diabetes-dataset (768 datapoints)

Models: Deep Learning and SVM

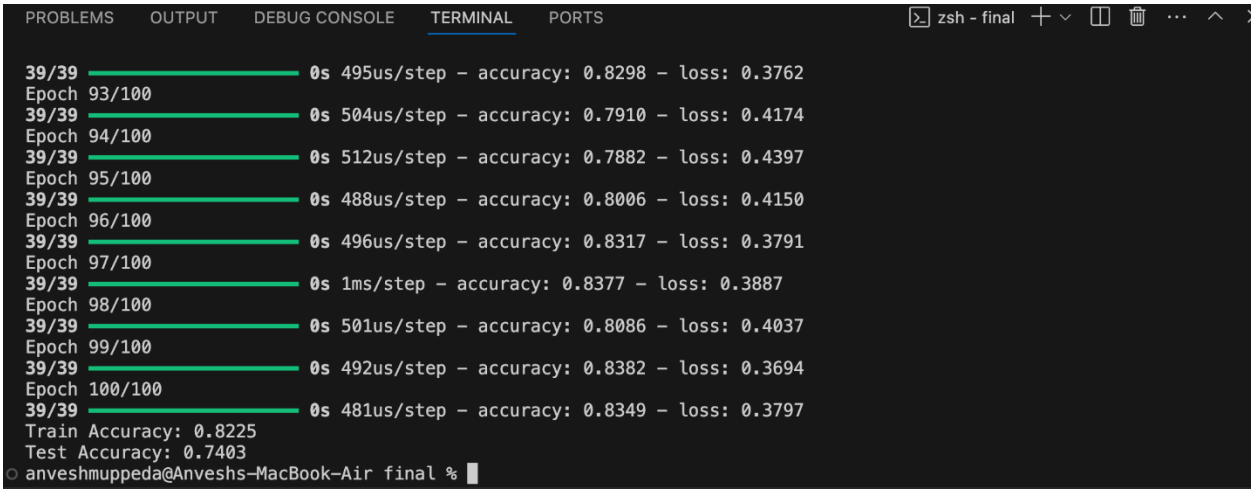
Observations:

The SVM model consistently produces predicted outcomes when trained correctly, indicating that overfitting is less of a problem for conventional machine learning methods.

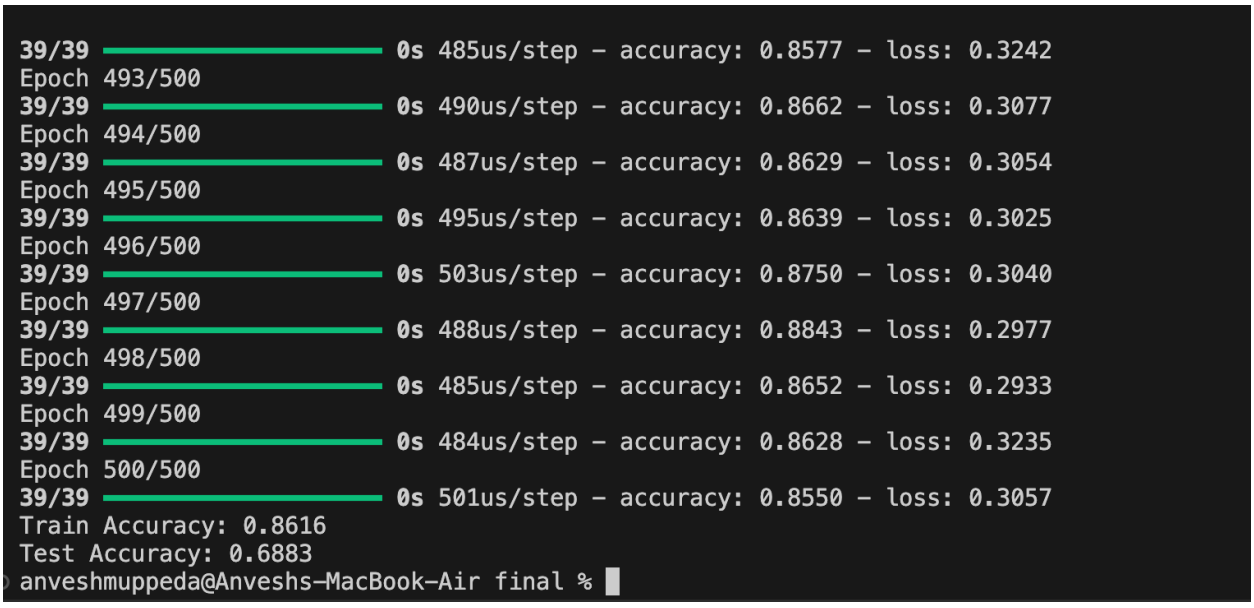
However, with the Deep Learning model, the test accuracy score rises to and the training accuracy score approaches 100% as we increase the epochs value from 100 to 1000.

Screenshots:

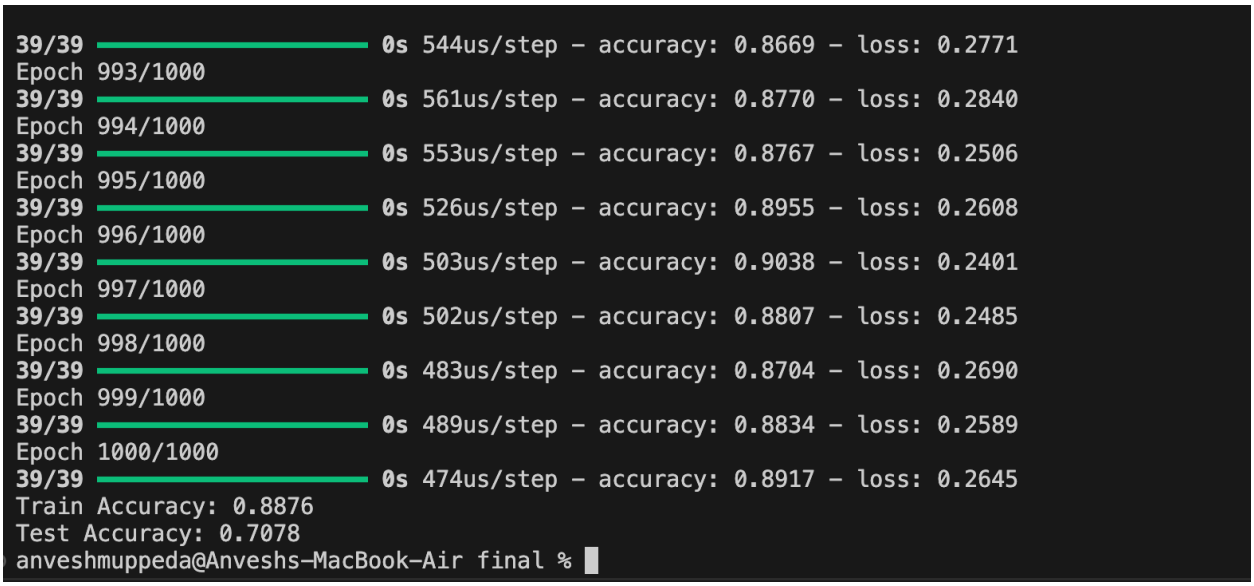
Epochs: 100



Epochs: 500



Epochs: 1000


39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 544us/step - accuracy: 0.8669 - loss: 0.2771
Epoch 993/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 561us/step - accuracy: 0.8770 - loss: 0.2840
Epoch 994/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 553us/step - accuracy: 0.8767 - loss: 0.2506
Epoch 995/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 526us/step - accuracy: 0.8955 - loss: 0.2608
Epoch 996/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 503us/step - accuracy: 0.9038 - loss: 0.2401
Epoch 997/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 502us/step - accuracy: 0.8807 - loss: 0.2485
Epoch 998/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 483us/step - accuracy: 0.8704 - loss: 0.2690
Epoch 999/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 489us/step - accuracy: 0.8834 - loss: 0.2589
Epoch 1000/1000
39/39 ━━━━━━━━━━━━━━━━━━━━ 0s 474us/step - accuracy: 0.8917 - loss: 0.2645
Train Accuracy: 0.8876
Test Accuracy: 0.7078
anveshmuppeda@Anveshs-MacBook-Air final %