**Project Report – Deep Learning part I**

Rishab Lokray(9357-3447), Sagnik Ghosh(3343-6044)

**Introduction:**

The goal of the project is to evaluate the performances of given classifiers and regressors on a simple tic-tac-toe data set. Set of classifiers(KNN, MLP, SVM) and a set of regressors(KNN, MLP, Linear Regression) are trained on the given datasets. The classifiers are evaluated using confusion matrix & prediction accuracy; The regressors are evaluated on prediction accuracy. After evaluating the performances of all the regressors, the regressive model based on MLP was chosen as the best preforming model for our game.

**How to Run The Program:**

All the dataset files, the program files and the pickle files need to be present in the same directory for the application to run.

Dataset files: *tictac\_final.txt, tictac\_multi.txt, tictac\_single.txt*

Program files: tictactoe.py, classifiersandregressors.py

Trained Model Weight Files: Files that contain the pretrained models that can be loaded by the program.

***classifiers\_and\_regressors.py:***  Run this file using the below command

* Python3 classifiersandregressors.py

Follow the instructions on the command line to test the performances of the Classifiers or the Regressors separately.

Performance output would be printed on the console.

***tic\_tac\_toe.py:***  Run this file using the below command to play the game

* python3 tictactoe.py

By executing the above command, we can start playing a new game from scratch, with a new blank board.

We can also pass a <Listobj> as a command line argument which will be taken as the current status of the board.

* python3 tictactoe.py <ListObj>

For Example: python3 tictactoe.py [1,1,0,1,0,0,0,0,-1]

Follow the instructions on the command line to play the game using the regressor of your choice.

**NOTE:** All models are already trained and weights will be automatically imported so that we don’t have to train the model every time. This is achieved using python’s pickle module. To train the models on a different dataset we have to change the global variable action to “Train”in the classifiers\_and\_regressors.py file.

action = “Train”

Change action to “Test” to import weight and test the models.

**Implementation:**

For the *classifiers\_and\_regressors.py* file we have organized the classifiers in one class and regressors in another.

Class Classifiers(): Contains six methods, 3 for dataset1 and 3 for dataset2.

The methods are: knn\_dataset1, knn\_dataset2, mlp\_dataset1, mlp\_dataset2, svm\_dataset1, svm\_dataset2.

Class Regressor(): Contains 3 methods, and each performs regression on dataset3. The methods are liner\_regression, mlp and knr.

\*\*\*Note – We use the below nomenclature across the code and the project report.

Dataset1 is tictac\_final.txt

Dataset2 is tictac\_single.txt

Dataset3 is tictac\_multi.txt

* The data has been split into Training set and Testing set with 4:1 ratio.
* The test set has been kept separately for evaluating performance of the respective models.
* The training set is used to perform K Fold validation to train the models, with K=10.
* The KFold validation helps in choosing the best parameters for the models.
* The best performing model from the kfold validation is used to test accuracy against the testing set.
* In linear regression as we are using a regression algorithm to perform classification we have used the columns with the highest value as 1 and rest as 0.

For the *tictactoe.py* file we have organized the Class PlayTictactoe() as follows:

Method display\_board: Used to print the tic tac toe board on the console.

Method start\_game: This is where the main logic is written. It takes board as input and uses the regressor of choice to power the computer player(-1). While the input from the human is taken from the command line (+1).

**Evaluation Results:**

We have used accuracy\_score() and confusion\_matrix() of the sklearn.metrics class to measure the model performance.

Note - For testing accuracy, we keep a X\_test testing set separate while X\_train is used in Kfold cross validation for training and validation.

CLASSIFICATION RESULTS:

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset | MLP | KNN | SVM |
| Dataset1 | 97.91% | 100% | 97.9% |
| Dataset2 | 92.06% | 85.3% | 35.4% |

From our observations we can see that MLP performs the best for Dataset2 while KNN performs the best for Dataset1.

For Dataset1 although KNN performs better, it should not be the preferred choice of algorithm for larger datasets as it calculates everything from ground zero for every new prediction. MLP is a more obvious choice as it performs better for larger datasets and is better for nonlinear data.

For Dataset2, Although KNN can be used for multiclass classification, its performance here is worse than mlp as the classes are not clearly separable.

REGRESSION RESULTS:

|  |  |  |
| --- | --- | --- |
| KNR | MLP | Linear Regression |
| 93.94% | 96.06% | 72.7% |

MLP performs the best for Dataset2 and liner regression performs the worst. This is expected because the data we are working with is nonlinear and we are using a regression algorithm to perform classification in the case of liner regression.

**Evaluation Results For 1/10th Data:**

The models are trained and tested on a smaller dataset size. Following are the results.

CLASSIFICATION RESULTS:

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset | MLP | KNN | SVM |
| Dataset1 | 79% | 80% | 98% |
| Dataset2 | 70% | 50% | 49% |

As SVM uses linear separation between data to classify, it’s accuracy does not change as it does not depend on the number of datapoints.

**Results and Graphs:**

Graphs showing the value of K and its associated accuracy level. We chose our K value for the KNN and KNR algorithms based on these results.

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Below are the confusion matrixes for the three classifiers on dataset1 and dataset2

A picture containing graphical user interface

Description automatically generated A picture containing text

Description automatically generated

A screenshot of a cell phone

Description automatically generated Calendar

Description automatically generated

A picture containing graphical user interface

Description automatically generated A close up of a keyboard

Description automatically generated

**Evaluation Of TicTacToe:**

Using MLP it was quite difficult to win against the computer. We could beat the computer 1 in 15 times. While using the liner regression it was comparatively easy to win(5 in 15 times). This satisfies our accuracy observations.

We observed that the computer might take more moves to win the game. This is because the datasets train the computer to win the game but not to stop us from winning. The computer would be much harder to beat if it was trained well.

\*\*\*