MLDM 2021 Coursework Group: The Algorithms

Rosmin Ann Raju rr00775@surrey.ac.uk
Sebastian Angel SA01929@SURREY.AC.UK
Sephy Sabu SS03583@SURREY.AC.UK
Sahla Marjan S-000282SURREY.AC.UK
KEYUR ATUL BILGI KB01205@SURREY.AC.UK

Abstract

Machine learning is a subdivision of Artificial Intelligence. It deals with training models with data. This technique can be used in many industries to formulate the best results. In health industry a self-learning neural network algorithm can improve the diagnosis and treatment efficiency. The first dataset includes health records of patients. The aim of choosing this dataset is to predict heart failure of the patients based on the health record attributes. The second dataset contains data about the results of diagnostic tests of human beings. There are different diagnostic parameters which would determine if the person has diabetics. The algorithms used are SVM, Decision Tree, MLP, Naïve Bayes, KNN and Reinforcement Learning.

1. Project Definition

Machine learning technique has increased the efficiency in healthcare sector. This has helped to automate a lot of diagnostic procedures. Data related to this sector has been taken and used in these experiments. The algorithms compared are Support vector machines, Neural network, Reinforcement learning, Decision tree, Multi-layered perceptron, and Naive bayes. Hyperparameter tuning has also been implemented before passing the parameters to the algorithms to tune the performance of the algorithms.

Hyperparameter tuning has been done using GridSearchCV. This is a library in sklearn model_selection package. Using this will help to select the best hyperaramets for the model. This algorithm selects a list of values and is turned as the input for the parameters. Passing these inputs will return a set of values which are the best parameters for the estimated model.

It is assumed that Tree-based algorithms will work best on classification problems. Decision tree is selected along with other algorithms for experimenting. Hyperparameter tuning is assumed to produce positive results which in turn will improve the performance the algorithms.

The results for each algorithm are discussed using confusion matrix and the usage of accuracy, precision, recall, f1-score and support. The comparison of algorithms has been done using k-fold cross validation and plotting ROC curve.

The first dataset contains several health attributes of people which will predict the chances of heart attack. Heart attacks are one of the leading causes of heart disease and the need for early intervention in the event of a heart attack cannot be underestimated. Thus building a model to predict chances of heart attack will be of great help.

1.1 Data Preparation

Dataset 1: Heart Dataset

The dataset contains health records of patients. The fields in the dataset are used to predict if the patient has a chance of heart failure. The target column 'HeartDisease' depicts 1 or 0. The other columns are

(i) Age
(ii) Sex
(iii) ChestPainType: It depicts the type of chest pain.
(iv) RestingBP
(v) Cholesterol
(vi) FastingBS
(vii) RestingECG
(viii) MaxHR
(ix) ExerciseAngina
(x) Oldpeak
(xi) ST_slope

Reference
https://www.kaggle.com/datasets/fedesoriano/heart-Data preparation

Dataset 2: Diabetes Dataset

This dataset contains patient records of females above the age of 21. The target column here is ‘Outcome’ which is a class variable of 1 or 0. Other fields include

(i) Pregnancies
(ii) Glucose
(iii) Bloodpressure
(iv) Skin Thickeness
(v) Insulin
(vi) BMI
(vii) DiabetesPedigree function
(viii) Age

Reference:
https://www.kaggle.com/datasets/mathchi/diabetes-data-set?resource=download

1.2 Data Exploration

Dataset1:

- The field ChestPainType is denoted by TA for Typical Angina, ATA for Atypical Angina, NAP for Non-Anginal Pain and ASY for Asymptomatic.
- RestingECG: resting electrocardiogram results [Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria]
- There were no missing values in the dataset.
- There were no duplicate values in the dataset.
- Correlation matrix was plotted to see the relation of attributes.

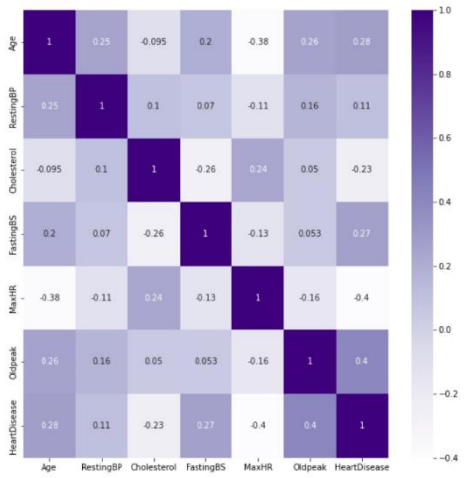
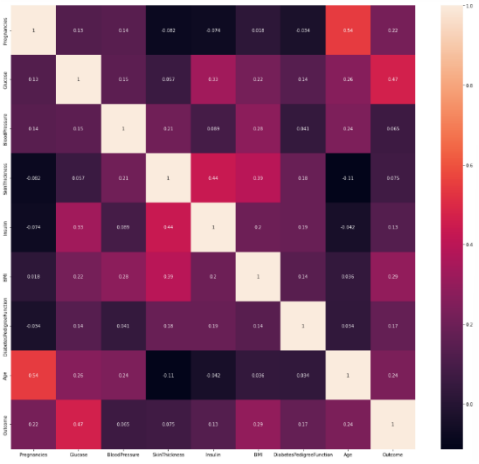


Fig 1.2.1 Correlation matrix

It is shown that Oldpeak has positive correlation to ‘HeartDisease’ and cholesterol has negative correlation to ‘HeatyDataset’.

Dataset 2:

- Every attribute is numeric valued.
- Decription of some attributes:Pregnancies(total number of times pregnant), BMI(Body mass index(weight in kg/(height in m)^2)),Insulin (mu U/ml)
- There was no null value in the dataset
- We checked for duplicates there came to be none
- Correlation matrix was plotted to see the relation of the attributes



- From the plot it is shown that Pregnancies has positive correlation to ‘Age and SkinThickness has negative correlation to Age.

1.3 Data Pre-processing

Dataset 1

One hot Encoding

Columns ‘Sex’, ‘ChestpainType’, ‘RestingECG’, ‘ExerciseAngina’, ‘ST_slope’ has text values. These fields were converted to numerical values using one hot encoding. Values 0 and1 for ‘sex’, 0 to 3 for ‘ChestPainType’, 0 to 1 for fastingBS, 0 to 2 for ‘RestingECG’, 0 to 1 for ST_slope were factorized into the dataset.

Standardization

Standardization was used here as the dataset contained data which varied in ranges. The fields were measured in different units. StandardScaler from sklearn library was used here.

Dataset 2

All values in this dataset were numerical values so therefore there was no need of one-hot encoding.

Standardization was also done on the dataset but it caused decrement of accuracy, thus standardization was removed.

1.4 Data Visualization

Scatterplot

Scatterplot was plotted to understand the relationship of attributes of the dataset. This is used to represent how much one variable is dependent on another by depicting the points on horizontal and vertical axis. Fig 1.4.1 shows the scatterplot for the dataset1 and Fig 1.4.2 shows the scatter plot for the second dataset.

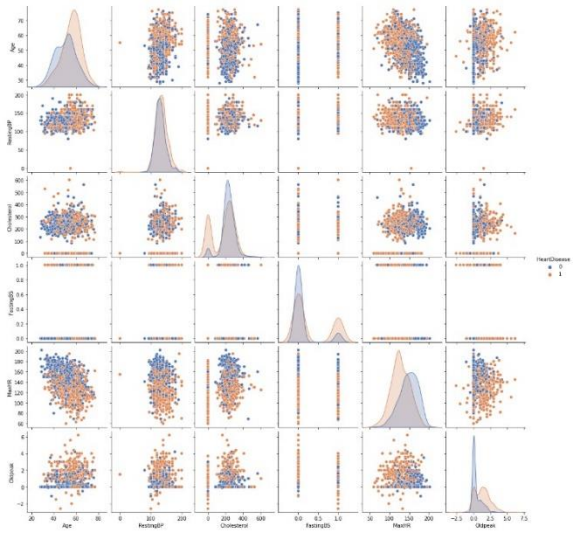


Fig 1.4.1 Scatterplot

Fig 1.4.2

Histogram

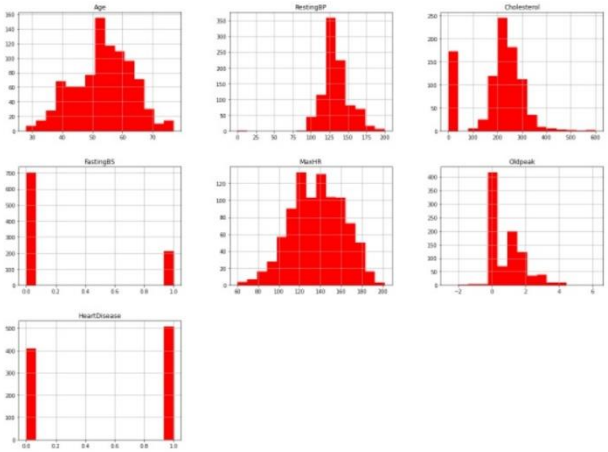


Fig 1.4.3

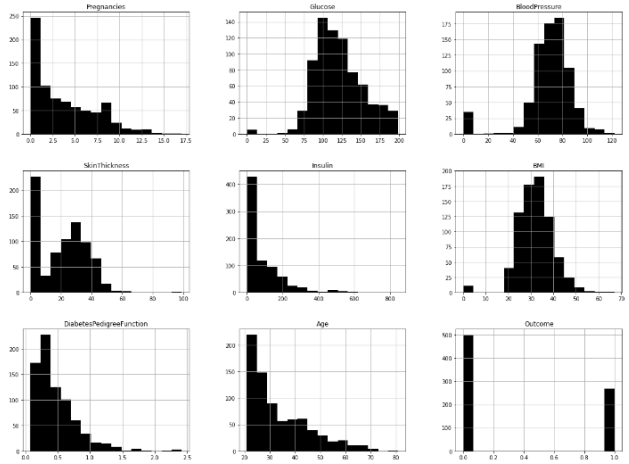


Fig 1.4.4

Histogram was plotted to depict the distribution of attributes in each field.

Barplot

A barplot was plotted to understand the cholesterol rate over the age of the patients. This shows that there is no significant distribution on any particular age. Fig 1.4.3 shows the barplot.

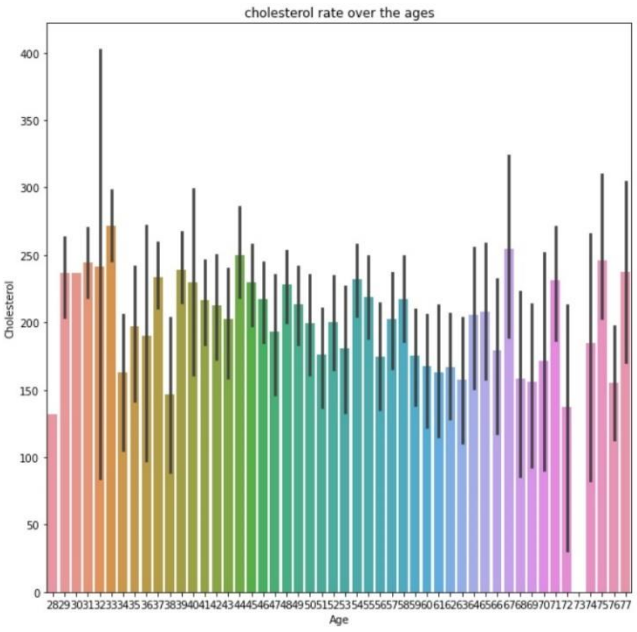


Fig 1.4.5 Barplot

Linegraph

This was plotted in Dataset 2 to see the variations between the diabetic and normal person.

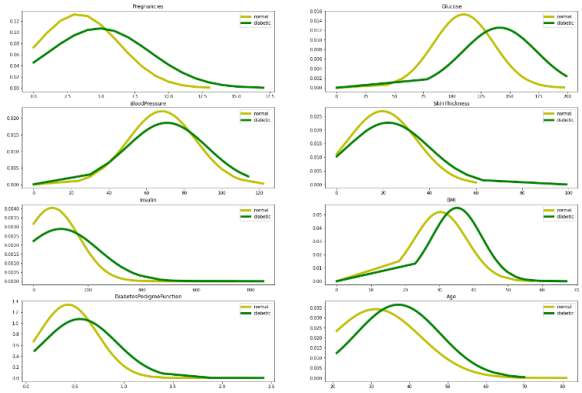


Fig 1.4.6 Linegraph

3.

Model Development

Several classifications and regression algorithms have been used in the dataset. The tested algorithms and the parameters used are explained in detail below.

3.1 Support Vector Machine

Library: sklearn

Version: 1.0.2

Module: sklearn.svm

An SVM classifier is used for two group classification problems. It works by capturing already labeled data points and ouputs the hyperplane which seperates the tags. This is called decision boundary. This particular decisions boundary maximises margines from both tags.

The advantages of SVM are that this algorithm works best on dataset with many features. It works more efficienly than other algorithms if the decision boundary is clear. The chances of overfitting is less in case of SVM.

The disadvantages are it doesn't perform well in dataset that contain large amount of noise. SVM also doesn't perform well in huge amount of data as it takes lot of time to function.

Hyperparameter tuning

C = [0.1,1,10].

This depicts misclassification or error. This parameter tunes the amount of error that is acceptable in the SVM. Large C value represents small margin of hyperplane

Kernel = ["linear","rbf", "poly", "sigmoid"]

This parameter decides what type of hyperplane is to be created. This hyperplane is used to separate classes. If linear is selected, it creates a linear hyperplane, otherwise not.

3.2 Decision Tree Classifier

Library: sklearn

Version: 1.0.2

Module: sklearn.tree

A decision tree consists of an initial node, internal nodes, leaf nodes and branches. At the initial node and internal nodes, an attribute (X_i) is tested for which one value is selected by a branch to the next node. Eventually, a leaf node is reached which signifies the prediction of Y. The decision tree aims to find the function that transforms the initial X_i into Y. The algorithm used in decision trees is the ID3 algorithm. The ID3 algorithm is a greedy top-down learning algorithm that uses entropy to measure the homogeneity of the training example samples to determine which branch to follow from node to node. A pro of using decision trees is that they are very easy to

read however one of the cons is that they are prone to overfitting.

Hyperparameter tuning

Criterion- gini, entropy

This parameter measures the quality of the split made by the trees

max_depth- (1,10), (20,50), (100,200,25)

This parameter decides the depth of the decision tree.

min_samples_split- (1,5), (1,10), (100,200)

This parameter shows the minimum number of samples needed to fit.

3.3 Multi-Layered Perceptron

Library: sklearn

Vesion: 1.0.2

Module: sklearn

The multilayer perceptron, or MLP, is a type of neural network. They can be used for classification and regression and are very non-linear. MLP uses the backpropagation algorithm which is very slow to train as it can take thousands of iterations however after the network has been trained, using it is fast. A couple of drawbacks of this method are poor generalization and lack of explainability.

Hyperparameter tuning

Hidden_layer_sizes=[10,30,10], [20,]

This depicts how many hidden layers are present in the neural network.

activation= [tanh, relu]

This parameter calculates which neuron is to be activated.

solver=[sgd, adam]

This parameter is used for weight optimization.

alpha=[0.0001, 0.05]

This parameter represents the strength of the regularization term.

learning_rate= ['constant','adaptive']

This parameter is the schedule for eight updates. Constant gives a constant learning rate and adaptive changes the learning rate is the training loss is decreased.

3.4 Naïve Bayes

Library: sklearn

Version: 1.0.2

Module: sklearn.naive_bayes

Naives Bayes is a classifier that assumes independence between features. This assumption is often incorrect but regardless, naïve bayes works well most of the time. No estimated posteriors are needed in this method.

3.5 KNN

Library: sklearn

Version:1.0.2

Module: sklearn.neighbors

KNN (k-Nearest neighbor) is a form of instance-based learning. Advantages of KNN are that training the model is very fast and it can easily learn complex target functions. Another advantage is that the model doesn't lose information. The disadvantages are that the query time is slow, it requires a lot of storage and the algorithm can be thrown off by irrelevant attributes.

Hyperparameter tuning

n_neighbour [3,5,10,20]

This decides the k value which is the number of neighbours.

weights=[uniform, distance]

This calculates how much its neighbours affect a certain data point.

3.6 Logistic Regression

Logistic regression is the method of calculating the probability of a distinct result when we give an input variable. Mostly the logistic regression models give a binary outcome, which can be true /false, yes/no and likewise. It's an analysis tool which is can be used easily for determining if a new data fits best into the classification task.

4. Model evaluation / Experiments

Several algorithms have been tested with various hyperparameter tuning across both datasets. Null hypothesis and results have been discussed in detail on following subsections.

Both datasets were trained with atleast four algorithms, gridsearchcv has been implemented wherever necessary.

4.1 Experiment 1 (dataset 1)

4.1.1 NULL HYPOTHESIS 1

The hypothesis for this experiment is that random tree algorithms and SVM will perform efficiently as the dataset is basically two-group classification data.

4.1.2 MATERIAL & METHODS 1

Five models have been used to study the dataset. The dataset has split into 70:25 ratio. K-cross validation was

used with K=10. This was used to split the dataset into 10 to reduce bias and overfitting.

4.1.3 Results & Discussion 1

After comparing the results of different algorithms, it is evident that multi-layer perceptron has the most accuracy which is 85.56%. It is also visible that accuracy of KNN also comes to near the accuracy of MLP.

	Classification Algorithms	Accuracy Score	Recall	Precision	F1-score
0	SVM Classifier	0.834783	0.892562	0.812030	0.850394
1	Decision Tree Classifier	0.778261	0.842975	0.761194	0.800000
2	MLP Classifier	0.856522	0.917355	0.828358	0.870588
3	Naive Bayes Classifier	0.808696	0.842975	0.803150	0.822581
4	KNN Classifier	0.847826	0.884298	0.835938	0.859438

Fig 4.1.3.1 performance score for dataset 1

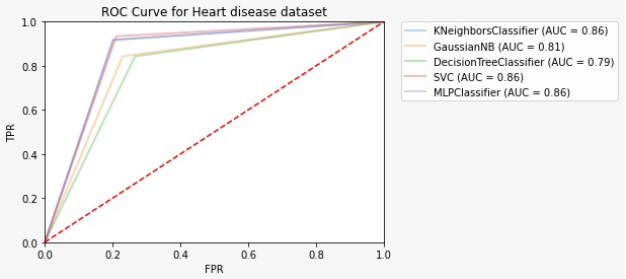


Fig 4.1.3.2 ROC curve for dataset 1

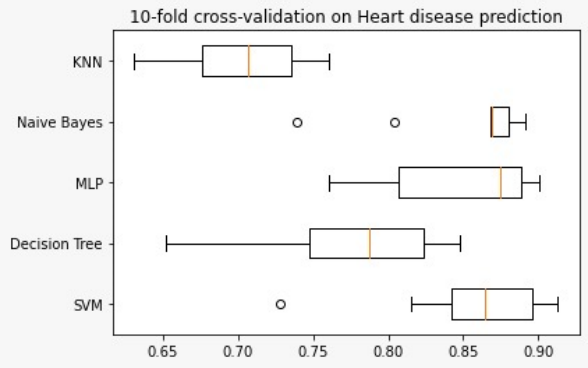


Fig 4.1.3.3 cross validation (dataset 1)

4.2 Experiment 2

4.2.1 Null Hypothesis 1

The hypothesis for this experiment is that random tree algorithms and SVM will perform efficiently as the dataset is basically two-group classification data.

4.2.2 Material & Methods 1

Five models have been used to study the dataset. The dataset has split into 70:25 ratio. K-cross validation was used with K=10. This was used to split the dataset into 10 to reduce bias and overfitting.

4.2.3 Results & Discussion 1

The modelling of different algorithms in dataset has yield lesser accuracy than dataset 1. The most accurate algorithm is Naïve Bayyes and Logistic regression. It has same accuracy.

	Classification Algorithms	Accuracy Score	Recall	Precision	F1-score
0	SVM Classifier	0.729167	0.461538	0.638298	0.535714
1	Decision Tree Classifier	0.687500	0.523077	0.539683	0.531250
2	MLP Classifier	0.697917	0.430769	0.571429	0.491228
3	Naive Bayes Classifier	0.744792	0.538462	0.648148	0.588235
4	KNN Classifier	0.713542	0.338462	0.647059	0.444444
5	Logistic Regression	0.744792	0.461538	0.681818	0.550459

Fig 4.2.3.1 Performance score for dataset 2

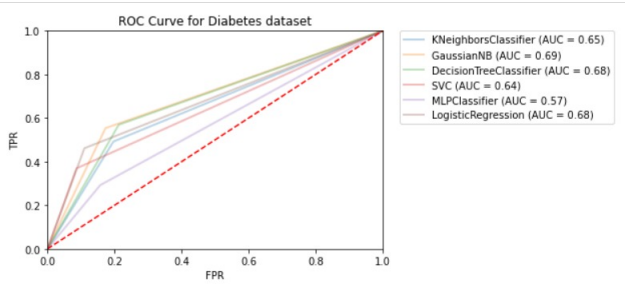


Fig 4.2.3.2 ROC curve for dataset 2

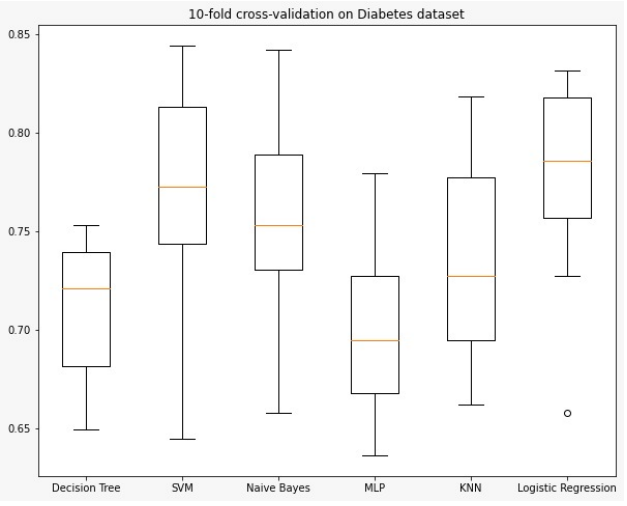


Fig 4.2.3.3 cross validation (dataset2)

4.3 Experiment 3 (Reinforcement Learning)

Optimisation of a Cartpole problem has been taken. Reinforcement learning (RL) is an area of machine learning in which an AI-powered system educates through trial and error while getting feedback on its activities. This input can be negative or positive, and it can be sent as a punishment or a reward to increase the reward function. Training can be done on the dataset without any prior knowledge of labels or answer keys. Using the gym library, we created the 'Cartpole' environment. The OpenAI Gym library provides a large number of agents that can be trained. In the environment, there are four possible states: cart position, cart velocity, pole angle and pole angular velocity. A cart is attached to a pole and drives along a frictionless track. The Cartpole environment is performed by coupling the pole to the cart so that the joint cannot move. A uniformly random value has been assigned to all the observations. In order to prevent the pole from dropping, it needs to be kept in place when it has started to stand. We use a point-based system here, with a payout of +1 for every timestep the pole remains upright. When the pole is titled +-12 degrees from a vertical or the cart position moves +-2.4 units from the centre, or the length of the episode is more than 500, the episode ends.

OpenAI Gym library lets to play around with the simulated environment. Think of an episode as one full game within the environment. Some environments have fixed episode lengths (Cartpole has 200 frames, others are continuous). The main environment function are env.reset() which resets the environment and obtain initial observations, env.render() visualises the environment, env.stop() applies an action to the environment and env.close() which closes down the render frame.

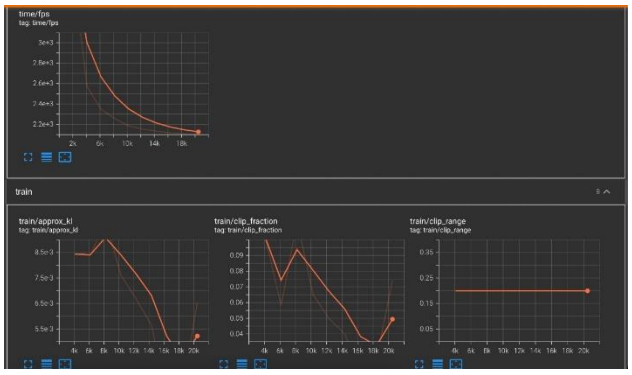


Fig 4.3.1 Reward graph

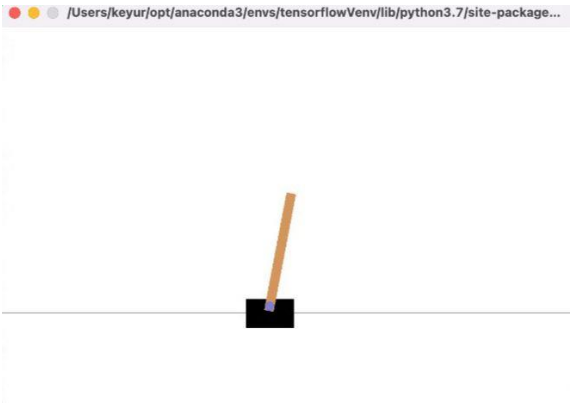


Fig 4.3.2 Cartpole before reinforcement learning

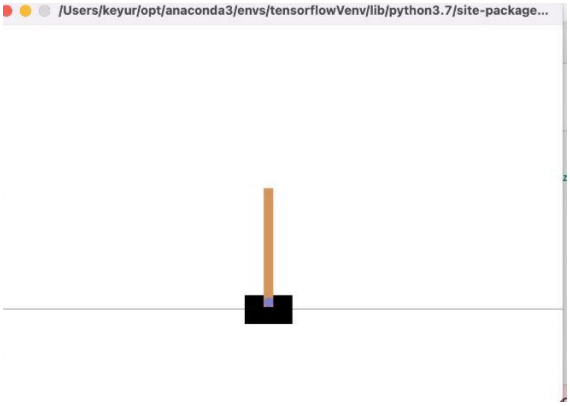


Fig 4.3.3 Cartpole after reinforcement learning

The final phase was possible because the algorithm was given time to train the surrounding.

5. Discussion of the results, interpretation and critical assessment

On the dataset, different models perform at different levels; nonetheless, we can anticipate which model is better for a certain type of task. The performance is determined by various factors, including the nature and amount of the data, its cleanliness, and the dataset's linearity.

By working on both datasets, it has been evident that hyperparameter tuning has not yielded much of a change in the result. Therefore, the observation is that further

increment of hyperparameters will not capture any great result and also will lead to usage of more computational power. Further discussion of result in detail would help us understand the data better.

The accuracy value of SVM is pretty high in both datasets. Dataset 1 has an accuracy of 83 and 72 in Dataset 2. Since accuracy score cannot be fully depended to check the performance. Since the dataset is imbalanced other performance scores like recall, precision, and F1 score should also be considered. The resukt for dataset 1 has yielded high scores for all of these performance scores but in dataset 2 has yielded very poor scores.

The accuracy value for decision tree is 68 for dataset2 and 77 for dataset 1. The hypothesis that tree classifiers will perform better on classification data has been not validated using any of the experiment as decision tree algorithm has performed poorly.

MLP classifier has given accuracy of 85 on dataset 1 and 69 on dataset 2. Other performance scores have also performed poorly on dataset 2 but has moderate performance on dataset1.

Naïve Bayes is used in NLP for text analysis, yet this algorithm has produced better results on dataset 1. Recall (0.84) Precision (0.83) and F1 (0.82). Accuracy on dataset 2 is fairly low which is 74.

KNN which uses a neural network has produced better results on Dataset1. It has an accuracy of 84. Other performance scores have yielded good results as well. But dataset two has performed not that better on KNN.

Reinforcement learning has performed fairly accurate. The algorithm uses what it has discovered after 2000 steps through exploration in order to raise the average reward. The final phase was possible because the algorithm was given time to train the surroundings and the model is able to balance the pole better than initially and made sure it stays straight

6. Conclusions

Five different alogorithms were tested on the dataset 1 and 6 different algorithms on dataset 2. MLP and KNN has performed well on Dataset 1 and Naïve Bayes and Logistic regression has performed well on Dataset 2. The dataset 1 had more level of accuracies than dataset 2 .

For further improvement elaborate tuning of individual parameters may lead to more improved results. Both of the datasets were also imbalanced, therefore working with more balanced and stable data can also improve performance of the models. Dataset containing Diverse data and more features could be used to produce high efficient models

Contributions

Rosmin Ann raju (6693898)

Attended all the project meetings, participated in finding the two datasets, implemented various algorithms, and helped with reinforcement problems and final report

Sebastian Angel (6531678)

Attended all the project meetings, participated in finding the two datasets, implemented various algorithms, and helped with reinforcement problems and final report.

Sephy Sabu (6708935)

Attended all the project meetings, participated in finding the two datasets, implemented various algorithms, and helped with reinforcement problems and final report.

Sahla Marjan (6711257)

Attended all the project meetings, participated in finding the two datasets, implemented various algorithms, and helped with reinforcement problems and final report.

Keyur Belgi Athul (6707875)

Attended all the project meetings, participated in finding the two datasets, implemented various algorithms, and helped with reinforcement problems and final report.

References:

[1] https://www.gymlibrary.ml/environments/classic_control/cart_pole/

[2] https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction

[3] https://stable-baselines3.readthedocs.io/en/master/

[4] https://www.kaggle.com/datasets/mathchi/diabetes-data-set?resource=download

[5] https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/

[6]. https://realpython.com/pandas-plot-python/

[7]. https://www.roboti.us/index.html

[8]

https://panjeh.medium.com/scikit-learnhyperparameter-optimization-for-mlpclassifier-4d670413042b