Report

COS 314 Artificlical Intelligence – Assignemnt 3

TEAM MEMBERS:

Hlungwani T.

Makhene T.

Mapheto T.

Motsepe O.J.

Introduction

The purpose of this assignement was to implement and evaulate Machine Learning models, namely: Genetic Programming (GP), Multi-Layer Perceptron. (MLP) and . Decision tree.(DT) to predict (classification) whether a financial stock should be purchased based on historical data.

The models were each built with Java programming lnaguage ensuring no model was built using a supirioir language .

The models were trained and tested on the same inout data, nmely BTC_test and BTC_train, ensuring fairness across the models

This report includes a detailed design specification for each model – including specific paramters used as well as a results table to measure the accuracy (ACC) and F-Score of each model to evaluate their respective performance as well as a detailed Wilcoxon signed-rank test carried out between the GP and MLP evaluate the significance of the differences in their respective performances.

Genetic Programming (GP)

The GP model is built using Java with several classes namely:

Dataset:

- Loads data from file into an object where we can retrive data and lables.

GPAlgorithm:

Core Engine of GP model.

- Builds the GP algoritm using paremters mentioned below.
- Creates runs for Individuals and fills them into a list : populations.
- The algorithm initialized the populationa nd evaluates the fitness using respective functions.
- Selects two parents
- Applies cross overs and mutates the parents and replaces them with their child individuals .

Individual :

- Built using the Tree Object and the fitness evaluation.

GPNode:

- Creates Nodes for the tree
- Attributes a function to them.

Tree:

- Generates a random expression tree using the GP nodes and allows for mutation, crossover and evaluation.

Main:

Explained in detail below

Design Specifications:

The main Program executes as follows:

Input Collection

program asks user for input for:

- Seed value – for reproducability.
- Absolute paths to training and testing CSV datasets.

Data loading

- Dataset and Buffers objects are used to load both datasets.

Model Configuration

The GPAlgorithm is configured using the created GPAlgorithm object with the following parameters:

- Train Dataset Object
- Seed value.
- Population Size = 50
- Max Depth = 5
- Generations = 30
- Mutation Rate = 0,3
- Crossover Rate = 0,9
- TournamentSize = 5

Model Traininn

A best Individual is then created from running GPAlgorithm

Model Evaluation

- An Evaluation Function that uses the best individual and test Dataset as paremeters is used to compare the score of the individual and the predicated score.
- Data labels are also extracted from the test Dataset.

Metrics used are:

- **Accuracy**: Percentage of correctly classified instances.
- **F1 Score**: Weighted harmonic mean of precision and recall, especially useful for imbalanced datasets.

Results

Results are outputed to terminal.

Multi-Layer Perecptron (MLP)

Implemented in Java using the Weka machine learning library.

Design Specifications:

The model executes briefly in the following steps:

Input Collection

program asks user for input for:

- Seed value – for reproducability.
- Absolute paths to training and testing CSV datasets.

Data loading

- Instance Objects from Weka are used to load both datasets 3.
- The last coloumn is set to the class index and converted to a nominal value

Model Configuration

The MLP is configured using the Percepetron object with the following parameters:

- Seed value.
- The model sets Hillden layers- which prevents the MLP from modeling a linear relationshiop between data. The dimension parameters of the matrix are set to 10x10 indicating 10 hidden layer neurons and 10 input vectors to the hidden layer.
- The learing rate (α) is the parameter that controls the rate at which the model learns, is set to 0.3, this medium rate is to ensure steady convergence in the model.
- The training time variable which is used to denote the time the model should take to learn and find an optimal solution is set to 500ms.

Model Training

Model is then trained using the buildClassifier() built in function to test the performance on both training and testing datasets.

Model Evaluation

An Evaluation Object is created to evaluate the traiing and test data respectively.

Metrics used are:

- **Accuracy**: Percentage of correctly classified instances.
- **F1 Score**: Weighted harmonic mean of precision and recall, especially useful for imbalanced datasets.

Results

Results are outputed to terminal.

Decision Tree (DT)

Results

The results for each model were collected and tabulized in *figure 1*. The table represents the seed value used, accuracy(Acc) an F-score (Fi) for both training and test data for each model.

Model	Seed Value	Training Acc (%)	F1	Test Acc(%)	F2
GP	140233	65,33	0.6239	47.53	0.3551
MLP	141025	70.94	0.6759	95.82	0.9581
DT					

Wilcoxon signed-rank test

As mentioned earlier, the test was carried out between the GP and MLP evaluate the significance of the differences in their respective performance across the same dataset. This test is used to determine if there is a statistically significant difference in accuracy and F-Score of the respective models.

This was automated using the **wsr.py** program which uses pythons built wilcoxon library under the the /Wilcoxon/ folder

INPUTS:

Accuracy for GP = [0.7094, 0.9582]

Accuracy for MLP = [0.6533, 0.4753]

F-Score for GP = [0.6239, 0.3551]

F-Score for MLP = [0.6759, 0.9581]

H_0: Median difference between GP and MLP performance (accuracy or F-score) is zero - neither method consistently outperforms the other.

H_1 : The median difference is **not** zero - there *is* a consistent difference between GP and MLP

Program Output:

Conslusion

Conlusion