Sales Prediction using Genetic Programming

1st Aamir Shoeb Alam Khan
Faculty of Computer Science
Dalhousie University
Halifax, Canada
am754815@dal.ca

***Abstract*—Prediction in the business domain is a widely used approach for economic planning, inventory production planning, and perfecting industrial processes. Sales prediction can help businesses plan operating expenses, inventory production, and marketing. The prediction could be regression or forecasting. In regression, the predicted value depends on the input whereas, In forecasting, the predicted value depends on time and inputs. There are many machine learning and statistical approaches to deal with Regression and Forecasting problems. In this paper we use Linear Genetic Programming (LGP), which is a type of Genetic Programming (GP), to solve the Sales prediction problem. We have discussed how LGP can be used to solve both the Regression and Forecasting problems. The results show that LGP does reasonably well for both Regression and Forecasting tasks.**

***Index Terms*—Prediction, Regression, Forecasting, Genetic Algorithm (GA), Genetic Programming (GP), Linear Genetic Programming (LGP), Population, Selection, Variation, Replacement, Crossover, Mutation, Sales Data**

I. Introduction

Machine Learning is capable of solving a variety of problems. Popular problems are Classification, Regression, Forecasting, and Clustering among others. Classification, Regression, and Forecasting are different types of Prediction problems. There are various Machine Learning and Statistical algorithms to tackle these problems. These algorithms predict future data points based on historical data. In this study, we have looked at Regression and Forecasting problems. Specifically, we are trying to predict sales by converting the problem to Regression and Forecasting problems. The purpose of predicting sales is to provide a basis for operating expenses planning, inventory production planning, and marketing planning.

A Genetic Algorithm (GA) [1] is an optimization algorithm that is inspired by Charles Darwins theory of natural algorithm. This algorithm mirrors the process of natural selection where the fittest individuals are selected for reproduction to produce offsprings for the next generation. The GA is one of the evolutionary algorithms and it is a population-based search algorithm. Population-based search algorithms start the search with multiple starting points unlike a gradient-based search. The GA is often used for nonlinear problems where the search spaces are very large.

A GA involves Population Initialization, Selection, Variation, and Replacement. We start with a randomly selected set of solutions. Each solution is called an individual. The solution set is a population and the process is called Population Initialization. Selection is the process of selecting individuals for Variation. Variation is where the offsprings are generated from the selected individuals. There are two variation techniques, Crossover and Mutation. Once the new offsprings are generated then we replace the prior population with the new offsprings and we repeat the Selection, Variation, and Replacement steps until we get the desired optimal solutions.

A. Population Initialization

The population is a set of solutions/individuals. Each individual corresponds to a chromosome. There are various techniques to represent a chromosome but the most common one is a binary representation. In a binary representation, the solution is represented as ones and zeros. E.g., 1100 is a binary representation of a chromosome. A chromosome is a set of genes. Each gene is a binary string in a GA individual. E.g., In 1100, there are four genes, 1, 1, 0, and 0. This is how an individual is represented in a GA setup.

B. Selection

Selection is the process of selected individuals for variation. The selection is based on an objective function (also called fitness function). This objective function identifies how good an individual is. Thus, individuals with higher fitness values have a bigger probability of contributing offsprings to the next generation. The fitness function depends on the problem we are solving.

C. Variation

This is the core part of a GA. This where the offsprings are produced. Variation is where evolution happens. There are two main variation operators, Crossover and Mutation.

1) Crossover: Crossover is a process where individuals of the last population are mated at random. After each mating, a pair of offsprings is generated. These offsprings are a combination of genes from two parents individuals, which hopefully have improved fitness values.

2) Mutation: Mutation brings in randomness to the population. It is the occasional random alteration of the value of a gene. E.g. In a bit-flip mutation, one or more bits are flipped, i.e. 1 becomes 0 or 0 becomes 1 in a bit representation of a chromosome.

D. Replacement

Replacement is the process of replacing some or all the individuals in the previous population with the new offsprings. There are various techniques to do this. We have used the Breeder model, which will be discussed later.

Genetic Programming (GP) [2] is a technique of evolving programs. It is based on GA, i.e., it includes all the evolutionary steps like Population Initialization, Selection, Variation, and Replacement. GP starts with a population of unfit programs and solves a particular task by evolving these programs. Each program is an individual. A program is a set of instructions. So, each instruction can be considered a gene in the chromosome (program). The complexity of a genetic program is usually measured as the number of instructions it holds [3]. There are various types of GP depending on their structural representation. A GP can solve problems like Classification, Regression, and Forecasting, among others. Linear Genetic Programming (LGP) [3] is a type of GP where the program is represented as a linear set of instructions. It does not mean that the method itself is linear, i.e., may solve linearly separable problems only. GP normally represent highly non-linear solutions due to their inherent power of expression. An LGP can be seen as a data flow graph produced by multiple usages of register content. Like a GP, an LGP also follows the evolutionary approach of Population Initialization, Selection, Variation, and Replacement. A simple LGP individual looks like this:

TABLE I
A SIMPLE LINEAR GENETIC PROGRAM

R[0] = R[0] + R[1]
R[1] = R[0] * 2
R[2] = R[2] / 2
R[3] = R[2] X[0]

In Table I, we can see that there are four registers R[0], R[1], R[2], and R[3]. There are four types of operations addition, subtraction, multiplication by 2, and division by 2. The input data is X[0]. During training, the value of these registers helps in fitness evaluation. The above program evolves with each generation/iteration of LGP training. We have used LGP to solve Regression and Forecasting problems. The aim of this study to understand the capability of LGP to solve the Sales Prediction problem. We have performed Regression and Forecasting using LGP. In the following sections, we will see the Methodology, Experiments, and Results.

II. METHODOLOGY

A. Dataset

The dataset was taken from Kaggle [4]. It includes the historical sales data for 45 Walmart stores located in different regions. Each store contains several departments, and the participants were supposed to predict the department-wide weekly sales for each store. The data covers all the sales that were done between 2010-02-05 and 2012-11-01. The following fields are available in the dataset:

- Store - the store number
- Dept - the department number
- Date - the week
- Weekly_Sales - sales for the given department in the given store
- IsHoliday - whether the week is a special holiday week
- Temperature - average temperature in the region
- Fuel_Price - cost of fuel in the region
- MarkDown1-5 - anonymized data related to promotional markdowns that Walmart is running. MarkDown data is only available after Nov 2011, and is not available for all stores all the time. Any missing value is marked with an NA.
- CPI - the consumer price index
- Unemployment - the unemployment rate

In addition to the above information, some information about special periods like Super Bowl, Labor Day, Thanksgiving, and Christmas was given.

B. Process

1) Data Preprocessing: The data preprocessing step involves data cleaning, exploratory data analysis, encoding, scaling, and feature engineering. Data preprocessing for Regression is slightly different from that of Forecasting. The key difference between the Regression and Forecasting is the predicted output. Regression predicts the actual weekly sales value whereas Forecasting will predict the direction of sale, i.e., if there will be a positive change or negative change of sales next week. Most of the preprocessing steps were taken from [5] and [6] since they gave a very good result. From the correlation plot of all the fields in [5], it was observed that the Discounts are correlated and higher unemployment means lower Consumer Price Index. It can be seen that there is a positive correlation between department numbers and sales. Moreover, larger stores generate more sales, discounts generally generate higher sales values and larger unemployment results in a bit fewer sales. It seems that there is hardly any relationship between holidays, temperatures or fuel prices with the weekly sales.

a) Common Steps: Although there were lots of finer preprocessing steps, we will mention the key ones.

- There were a few dates that had anomalous sales, i.e., the sales were significantly higher on these dates. These dates correspond to two days before Christmas and Black Friday. To make it significant, [5] engineered these features and named it as Black_Friday and Pre_christmas.
- Missing Values: There were missing values in Markdown1-5, CPI, and Unemployment rate fields. For Markdown1-5, we engineered five fields corresponding to each markdown. These fields indicated if there was a markdown or not. The missing values in Markdown1-5 were filled with zeros. For CPI and Unemployment, we

took the mean and filled the missing values with the mean.
- Encoding categorical variables: One hot encoding was done on each of the categorical fields. Encoded fields are Store, Dept, Type, and Month. All the boolean (True/False) fields were changed to ones and zeros.
- Scaling: All the continuous fields like Unemployment, Temperature, CPI, Fuel_Price, Size, and Markdowns1-5 were scaled using the Standard scaling method.

b) Regression: For Regression, we added some additional features as given in [6]. These features are several days before Christmas, Day, and Year. Since these features are continuous, they were scaled using the Standard scaling approach.

c) Forecasting: Since forecasting will be predicting the direction of sales and not the sales itself, there were some additional steps as mentioned in [5] that were taken to prepare the data for LGP training.

- To reduce noise, we took the median of the weekly sales in a month. The median was taken for Type, Dept, Store, Month, and IsHoliday attributes.
- To calculate the Sales Difference, which is our field to be predicted here, we calculate Lagged Variables of Weekly_Sales. A Lagged Variable(Sales) in this context is the previous weeks sale. If the previous weeks sales data were not given then we take the median value as the Lagged Sales value.

2) LGP Formulation: After a thorough preprocessing cycle, we formulate the prediction problem as a GP problem. The data processing done for Regression and Forecasting keeps one variable to predict and the other variables as input vectors. So, the LGP formulation for both Regression and Prediction will be the same.

a) Parameters: To represent the Regression and Forecasting problems as an LGP, we use the following parameters:

- Test data proportion (tdp): this field is used for training and test split of the dataset. The split is done in such a way that the distribution of classes is uniform across train and test set.
- Gap percentage (gap): Since our LGP uses a Breeder selection-replacement method [7], the gap percentage allows users to enter the percentage of replacement to be done in each generation. In each iteration (generation), a given gap percentage of weak individuals are removed from the population and are replaced by a given gap percentage of offsprings.
- Population size (p): The size of the population
- Number of generations (g): The number of times the selection, variation and replacement will be done in the LGP algorithm.
- Number of registers (r): The number of registers that can be used by the individuals (programs).
- Sample size (t): The number of training samples that will be drawn from the training partition.
- Resampling period (rp): Resampling period is the frequency of resampling during training(while iterating).

b) Sampling: Sampling is how the training samples are chosen periodically (based on *rp*) during training. We have used a uniform sampling approach where we uniformly sample t training samples from the complete training dataset.

c) Population initialization: Before the population was initialized, we represented the registers, operators, select, source and target in the form of integers. Each of them is described below:

- Registers: The register representation depends on the *r* parameter. If *r* is 4 then Register 0 will be represented as 0, Register 1 with 1, Register 2 with 2, and Register 3 with 3.
- Operator: The operator representation was fixed as we have four operators. The operator +(addition) was represented as 0, -(subtraction) as 1, *2(multiplication by 2) as 2, and /2(division by 2) as 3.
- Select: The select is a binary selector(1/0). It is chosen randomly and is used to select source. A select value of 0 represents input attributes and 1 represents registers.
- Target: Target is the output register of an instruction. Since target is a register, the representation is same as register representation.
- Source: This could either be a dataset field or a register. The choice is made depending on the value of Select. If it is registered then the representation will be the same as registers else if it is dataset attribute then the range of representation will be between 0 and number of attributes 1. E.g., If there are four attributes in a preprocessed dataset then the representation will be between 0 and 3 corresponding to each attribute.

After the representation, we initialize the population. The steps are:

1) Create a program counter that randomly (uniform) chooses a program size from 15-45 range for each individual in the population and stores it in a list.
2) Using the program counter list, the list of individuals is generated with the assigned length. The individual is a program. Each program will have a list of instructions as per the program counter list. Each instruction is randomly generated with uniform probability. An instruction is represented as [select, target, operator, source]. E.g. [1 0 1 1] is generated randomly (uniform probability) and it corresponds to R0 = R0 R1. The source is a register because the first value in [1 0 1 1] is 1. Similarly, [0 2 1 1] corresponds to instruction R2 = R2 X1. Here X1 represents the second input attribute and it is selected because the first value in [0 2 1 1] is 0. Another example for multiplication by 2 where the source does not matter, [0 1 2 1] corresponds to R1 = R1 * 2.
3) The population generation process continues until all the programs are initialized depending on the population size.

d) Selection: Selection is the process of selecting the individuals for mating. To do this, we first evaluate the fitness of each individual. Fitness evaluation: For each individual or program, we estimate the fitness score. The fitness function is the Mean Absolute Error (MAE). The process is as:

1) For each program in population:
2) For each row in training data:
3) Read the row data
4) Initialize the registers to 0
5) Execute the program
6) Check the prediction: the maximum value of all the registers with highest value is the predicted value.
7) Store the predicted value in predicted values list.
8) If it was the last row in training data then stop else move to next row.
9) Compare the predicted values against the true values. Calculate the MAE score. This MAE score is the fitness score.
10) If it was the last program then stop else move to next program (and record fitness_score).

Please note that the fitness score of the program is the MAE score. Since MAE is an error score, the lower the MAE the fitter will be the program.

e) Variation: Before replacement, the variation opeartors are applied.

Crossover: We have used a two-point crossover method to perform crossover between two parents. The points chosen are random and do not allow the program to increase in size. To prevent the program to increase in size, we have a Max program size limit to check if the crossover is possible or not. If the crossover will make the size of the program greater than the Max program size then the crossover will be ignored in this generation.

Mutation: We have used a bit flip mutation approach. We randomly(uniformly) pick the instruction to be mutated in a program. Then we randomly(uniformly) pick the bit for flipping. This could either be a target or operator or source. Finally, we flip the chosen bit randomly(uniformly).

f) Replacement: Selection and Replacement follows the Breeder model. The steps are:

1) In each generation rank the population by their fitness scores. The lower fitness value, the fitter will be the program since our fitness function is an error score.
2) Remove the worst Gap percentage programs.
3) Apply variation operators to define Gap percentage offspring choosing the parents from the 1 Gap percentage programs remaining in the population after step 2 with uniform probability.
4) Add the offspings in place of the removed Gap percentage programs.

The fitness scores for all the programs are calculated only for the first generation. From, second generation onwards, the fitness scores are only calculated for the offsprings. This makes the LGP algorithm run significantly faster.

g) Stop criterion: The stop criterion is based on the *g* parameter. The *g* parameter has a default value of 500. This parameter is however configurable. As the *g* loop ends, the LGP algorithm stops.

h) Metrics: Once the GP iterations end, the metrics are measured. The metric used to determine the Accuracy MAE. The measurement is done on the whole training and test dataset for selection of champion (best) predictor. Once the champion predictor is identified based on MAE. MAE is an arithmetic average of the absolute errors [8].

III. Experiments and Results

In this section, we evaluate our LGP Predictor with the various experiments that were conducted with different parameters.

A. Setup

The LGP for Regression and Forecasting was setup on Ubuntu 18.04 operating system. The device used was an MSI GS40-6QE laptop with the following key specifications: Intel Core i7-6700HQ processor, 16GB DDR4-2133 RAM, and 1TB Samsung 840 Evo Solid-State Drive. We used the following versions of the tools and libraries to set up the proposed system: Python version 3.6, pandas, NumPy, and scikit-learn. All the required tools are open source.

B. Dataset

Since the Walmart sales dataset is huge and contains 421,570 training rows and 115,064 test rows, the experiments would require more time to run. The actual dataset contains sales data of 45 stores and 99 departments of each store. For a quick experimental setup, we took three stores and five departments of each store. This reduced the training set to 1716 rows and test set to 429 rows. All our evaluations were done on this reduced training and test set.

C. Parameters

For our evaluation, we have chosen our standard set of parameters. Table II shows the standard parameters. These are not the best parameters. These standard parameters are chosen for reasonably good results and most importantly for faster training.

TABLE II
Standard Parameters

Parameter	Value
Test data proportion (tdp)	20
Gap percentage (gap)	20
Population size (p)	250
Number of generations (g)	500
Number of registers (r)	4
Sample size (t)	200
Resampling period (rp)	5
Max program size	64

The evaluation is done by varying some of the above parameters.

1) Regression:

TABLE III
MAE SCORE OF DIFFERENT MODELS

Model	MAE
Linear Regression	14566
KNN Regressor	8769
Decision Tree Regressor	2375
Random Forest Regressor (g)	1854
XGB Regressor	2291
LGP	5525

a) MAE against the existing approaches: In Table III, we have compared our LGP predictor with other models. These models were trained on the entire dataset and the training and reporting was done by [6]. Since we didn't train on the entire dataset, the results doesn't show the true comparison. From the comparison we can see that LGP lies in the middle in terms of testing accuracy. It is better than Linear Regression and KNN Regressor. However, Decision Tree, Random Forest, and XGB outperforms LGP by a good margin.

b) Effect of variation in the number of generations: Figure 1 shows the computed MAE for four different runs with the varying number of generations, 100, 500, 1000 and 5000. From the figure, we find out that the MAE decreases with the increase in the number of generations. It could even decrease further as we increase the number of generations. This is expected since with more number of generations, the LGP gets more opportunities to find the optimal solution. Thus, more number of generations improve the chances of getting better results in this case.

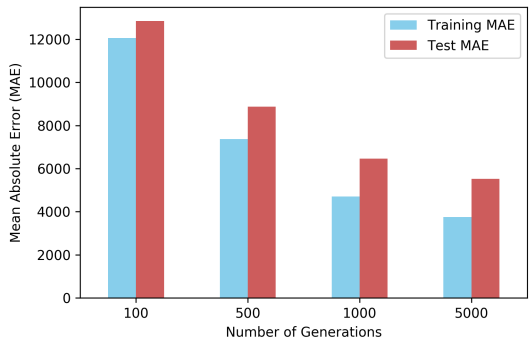

Fig. 1. MAE with varying number of generations

c) Effect of variation in population size: Figure 2 shows the computed MAE for four different runs with the varying population size, 100, 250, 500, and 1000. From the figure, we find out that theres slight hint of decreases in MAE when the population size becomes 1000. It could even decrease further as we increase the population size. This is possible since more number of individuals increase the population search that in turn would help increase the chances of finding the optimal solution.

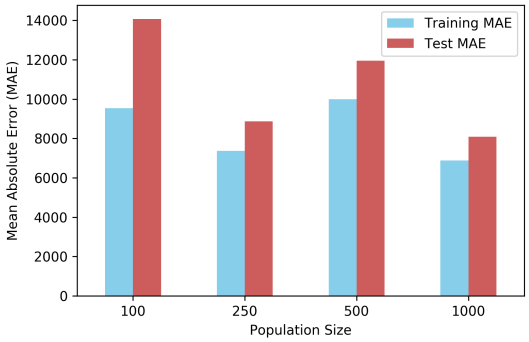

Fig. 2. MAE with varying population size

d) Effect of variation in the number of registers: Figure 3 shows the computed MAE for four different runs with the varying number of registers, 2, 4, 8, and 16. From the figure, we find out that there is a reduction in MAE with the increase in the number of registers up to a certain point, i.e. 8. However, when we increase the number of registers to 16, the MAE increases sharply. Hence, when the number of registers is up to 8, the LGP predictor does better than when it is greater than 8.

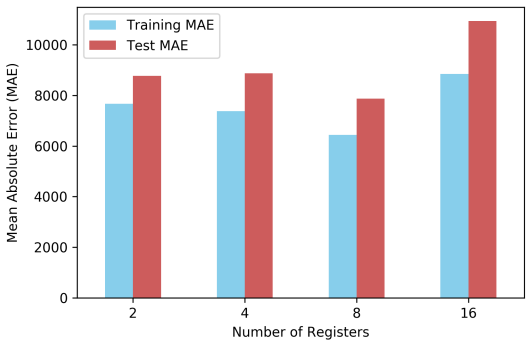

Fig. 3. MAE with varying number of registers

e) Effect of variation in the program size: Figure 4 shows the computed MAE for three different runs with varying program size, 64, 96, and 112. From the figure, we find out that increasing the program size is not helping our LGP predictor much. In fact, with an increase in program size, the complexity of the LGP predictor increases. It also takes more time to compute the fitness due to the execution of more number of steps.

f) Effect of variation in the training sample size: Figure 5 shows the computed MAE for four different runs with varying training sample size, 200, 300, 400, and 500. From the figure, we find out that the LGP predictor does well when the sample size is 400. However, with a sample size of 500, the MAE of LGP increases again. Hence, it is difficult to conclude if an increase in the training sample really helps.

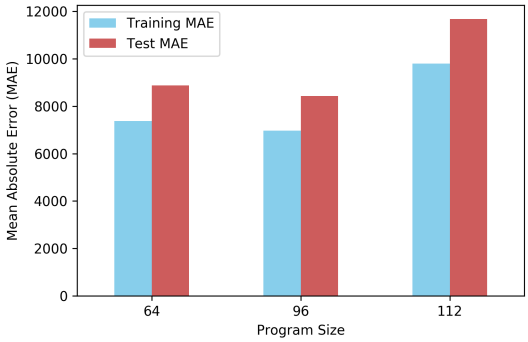

Fig. 4. MAE with varying program size

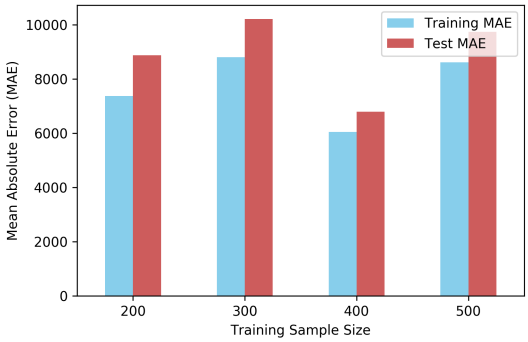

Fig. 5. MAE with varying training sample size

g) The best setting: After analysing the above, we came up with our best configurations. The following parameters were chosen as the best settings:

TABLE IV
BEST PARAMETERS FOR REGRESSION

Parameter	Value
Test data proportion (tdp)	20
Gap percentage (gap)	20
Population size (p)	250
Number of generations (g)	1000+
Number of registers (r)	8
Sample size (t)	400
Resampling period (rp)	5
Max program size	64

With the above settings, we ran our LGP predictor training and got reasonable results. We got a Training MAE score of 5645.69 and Testing MAE score of 6054.85. This is higher when compared to 3750.66 and 5525.38 Training and Testing MAE respectively when we ran with standard configuration and kept the number of generations as 5000. However, the ability of the LGP with the best settings to generalize better is a positive result. The difference between Training and Test MAE is quite low compared to the standard-setting (with 5000 generations). Figure 6 shows 50 samples of the weekly sales prediction by our LGP predictor and the true weekly sales of the test dataset.

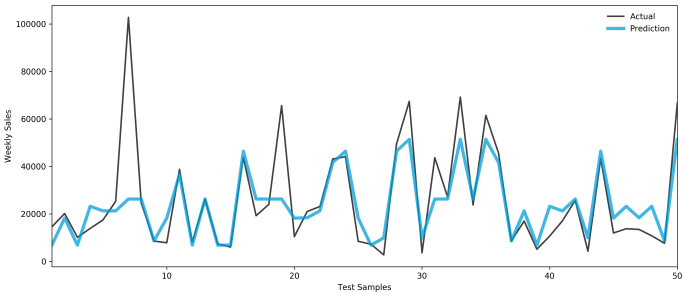

Fig. 6. Weekly sales prediction with the best settings

2) Forecasting:

TABLE V
MAE SCORE OF DIFFERENT MODELS

Model	MAE
Random Forest	1447.06
LGP	2488.56

a) MAE against the existing approaches: In Table V, we have compared our LGP direction predictor with other models. These models were trained on the entire dataset and the training and reporting was done by [5]. Since we didn't train on the entire dataset, the results doesn't show the true comparison. From the comparison we can see that Random Forest outperforms LGP in testing. However, LGP did reasonably well with a decent test MAE score of 2488.56.

b) Effect of variation in the number of generation: Figure 7 shows the computed MAE for three different runs with the varying number of generations, 100, 500, and 1000. From the figure, we find out that there is a slight reduction in MAE with the increase in the number of generations. It could even decrease further with the increase in the number of generations.

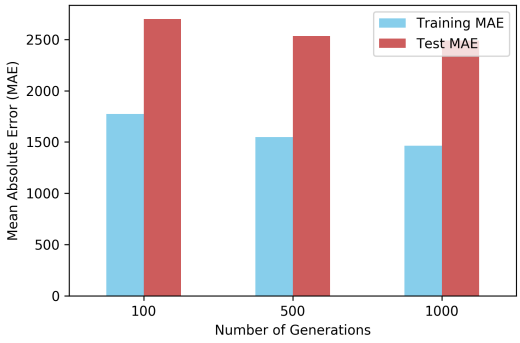

Fig. 7. MAE with varying number of generations

c) Effect of variation in population size: Figure 8 shows the computed MAE for four different runs with varying population sizes, 100, 250, 500, and 1000. From the figure, we find out that there is not much change in the MAE with varying population sizes.

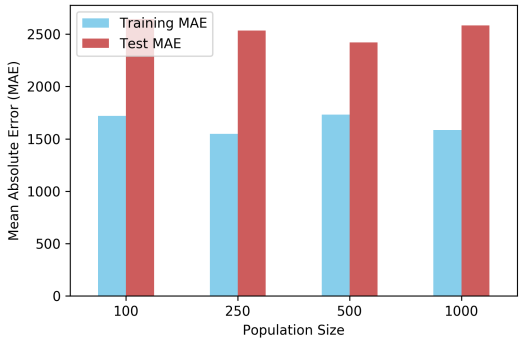

Fig. 8. MAE with varying population size

d) Effect of variation of the other parameters: The other parameters like training sample size, Max program size, and the number of registers did not affect the MAE score. There was not a clear pattern.

e) The best setting: The following parameters were chosen as the best settings:

TABLE VI
BEST PARAMETERS FOR FORECASTING

Parameter	Value
Test data proportion (tdp)	20
Gap percentage (gap)	20
Population size (p)	250
Number of generations (g)	1000+
Number of registers (r)	4
Sample size (t)	200
Resampling period (rp)	5
Max program size	64

With the above settings, we ran our LGP predictor training and got good results. We got a Training MAE score of 1464.22 and a Testing MAE score of 2488.56. The MAE score is better than the Regression model as Forecasting LGP is not predicting a real value. Figure 9 shows 50 samples of the sales difference prediction by our LGP predictor.

Since we are forecasting the direction of sales, we have categorized the direction (sales difference from weekly sales median) to positive and negative directions where the positive indicated increase in sales and negative indicates a decrease in sales. We have encoded positive to 1 and negative to 0. Hence it can be treated like a binary classifier. Figure 10 shows the confusion matrix of the direction classifier. The Accuracy was measured to be 67.83 percent.

IV. Conclusion and Future Work

In this paper, we have studied the use of Genetic Programming, especially Linear Genetic Programming for Regression and Forecasting tasks on the Sales Prediction dataset. The process consists of data preprocessing and all the steps of a Genetic Algorithm, Population initialization, Selection, Variation, and Replacement. The LGP predictors for Regression and Forecasting were evaluated with different parameter settings. The training was done on part of the actual dataset and not the complete dataset. The results were reasonably better if not best and show the potential of Genetic Programming in prediction tasks. A possible future direction is to improve preprocessing steps and to train the LGP predictor for more number of generations to improve the accuracy of the LGP predictor. Training on the complete dataset might give better results. Moreover, additional operators like trigonometric functions could be tried to improve the accuracy of prediction.

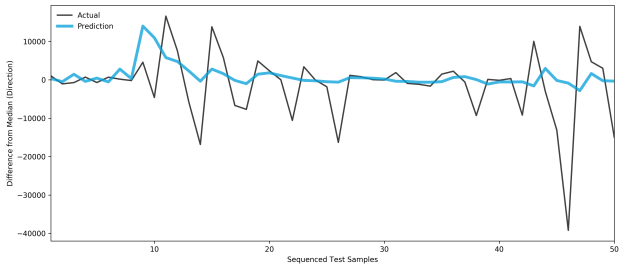

Fig. 9. Sales difference from median prediction with the best settings

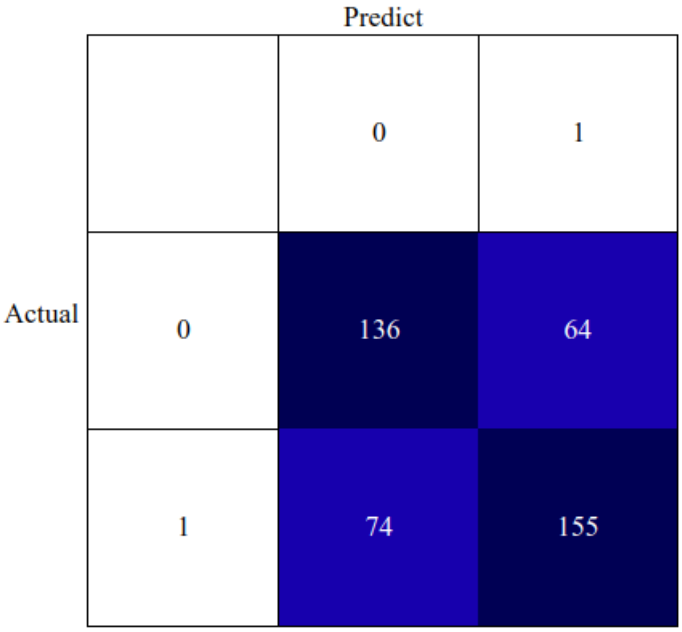

Fig. 10. Confusion matrix of positive/negative direction classifier with the best settings

References

[1] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Publishing Company, 10 1989.

[2] W. Banzhaf, *Genetic programming : an introduction ; on the automatic evolution of computer programs and its applications*. San Francisco, Calif. Kaufmann, 12 1997.
[3] A. M. Brameier and W. Banzhaf, *Linear genetic programming*. Springer, 12 2006.
[4] "Walmart recruiting - store sales forecasting."
[5] F. Lasso, "Cracking the walmart sales forecasting challenge."
[6] Aditya, "Walmart sales forecasting," 10 2019.
[7] H. Mhlenbein and D. Schlierkamp-Voosen, "The science of breeding and its application to the breeder genetic algorithm (bga)," *Evolutionary Computation*, vol. 1, pp. 335–360, 12 1993.
[8] W. Contributors, "Mean absolute error," 08 2019.