INDIAN INSTITUTE OF TECHNOLOGY

BOMBAY

COMPUTER SCIENCE AND ENGINEERING

FOUNDATIONS OF MACHINE LEARNING

CS 725

BIGMART SALE PREDICTION

STUDENT ID :	*STUDENT NAME :*
193050037	Gourab Dipta Ghosh
193050099	Raj Krishna Srivastava
193050059	Vinod Kushwaha
193050071	Rohit Kumar

Contents

1 INTRODUCTION

BigMart has collected 2013 sales data for 1559 products across 10 stores in different cities. Also, certain attributes of each product and store have been defined. The aim is to build a predictive model and find out the sales of each product at a particular store. Using this model, BigMart will try to understand the properties of products and stores which play a key role in increasing sales.

2 LITERATURE SURVEY

It is important to understand how does the company expect to use and benefit from this model? This first helps to determine how to frame the problem, what algorithms to select and measure the performance of each one.

For now, we can categorize our Machine Learning (ML) system as:

Supervised Learning task: We are given labeled training data (e.g. we already know some of the sales, for some products and stores);

Regression task: Our algorithm is expected to predict the sale for a given product and store.

Plain Batch learning: Since there is no continuous flow of data coming into our system, there is no particular need to adjust to changing data rapidly, and the data is small enough to fit in memory, so plain batch learning should work.

Performance Measure: Usually for regression problems the typical performance measure is the Root Mean Square Error (RMSE). This function gives an idea of how much error the system makes in its predictions with higher weight for large errors.

This dataset is a quite common one in ML competitions. A lot of work were done on this dataset and accuracy around 60% is what people could bring for this dataset. Apart from traditional ML algorithms, we have used ensembling and compared the results.

3 APPROACHES APPLIED

i. LINEAR REGRESSION

Linear regression is a basic and commonly used type of predictive analysis. The overall idea of regression is to examine two things:
(1) Does a set of predictor variables do a good job in predicting an outcome (dependent) variable?
(2) Which variables in particular are significant predictors of the outcome variable, and in what way do they–indicated by the magnitude and sign of the beta estimates–impact the outcome variable?
These regression estimates are used to explain the relationship between one dependent variable and one or more independent variables.

ii. NEURAL NETWORK

Neural Networks are also a very powerful tool in extracting out hidden information from a dataset. Our dataset has 12 attributes and each attribute either takes in nominal or ordinal values. We have used a Neural Network model with 2 hidden and 1 output layers. There are 32 nodes in each hidden with ReLU as activation function and 1 node in the output layer using sigmoid as activation function which gives the predicted output.
Because of its high dynamicity, Neural network models, after being fine tuned with proper parameters(Number of hidden layers, Number of nodes in each hidden layer, learning rate, etc),they are expected to train the data the best and give better outputs as compared to other machine learning algorithms.

iii. ENSEMBLE LEARNING

Ensemble learning is used in our model to bring out the best of both worlds. We have used bootstrap aggregation for the same. Predicted output from both linear regression and neural network has been used to produce a weighted average for the final prediction.

4 EXPERIMENTAL SETUP

We are using python as language and PyCharm as IDE. We have used packages scikit-learn and keras for model training, pandas for reading data, and numpy for numerical manipulations.
Link to code:
`https://github.com/kushwahaVINOD/Big-Mart-Sales-Prediction?fbclid=IwAR1q09hT7RBvuZs`
`AiVBPypMIIY`

The dataset have 11 attributes on which sale (predicted-variable) depends. The description of the attributes is mentioned below:

Item_Identifier : Unique product ID

Item_Weight : Weight of product

Item_Fat_Content : Whether the product is low fat or not

Item_Visibility : The display percentage area of all products in a store
Item_Type : The category to which the product belongs

Item_MRP : Maximum Retail Price (list price) of the product

Outlet_Identifier : Unique store ID

Outlet_Establishment_Year : The year in which store was established

Outlet_Size : The size of the store in terms of ground area covered

Outlet_Location_Type : The type of city in which the store is located

Outlet_Type : Whether the outlet is just a grocery store or some sort of supermarket

Item_Outlet_Sales : Sales of the product in the particular store. This is the outcome variable to be predicted.

Pre-processing

Item_Identifier was dropped since much of the information was captured by Item_Type itself.
Some values from Item_Weight column were missing. We filled them with the mean of available values of Item_Weight.
Similarly, in Item_Visibility some values were 0. But 0 visibility means the item was

not visible to the customer. Still those items had some sales value which indicated that the 0 values were error in recording. We filled the 0 values with the mean of available non zero values.
In Outlet_Size, some values were missing. This is a nominal variable, so we cannot perform mean smoothing. So we did modal analysis and assigned the modal value to the missing ones. In Item_Fat_Content, there were some values like Low Fat, and some were LF, both indicating the same. Similarly, there were Regular Fat and Reg. We had to do data cleaning in the preprocessing stage. The Outlet_Establishment_Year had very less impact on the algorithms' performances. So instead of dropping it, we converted it to Years_of_Operation by subtracting the year of establshment from 2013(Since this dataset is from 2013). Year of operation was of more significance than year of establishment. For training, we used 80% of the data available in train.csv file and the rest 20% for testing. The train test division was made randomly each time. For Linear Regression, we didn't set aside any data for validation since the epoch was not set much high. But for Neural Networks, out of the training part, 90% was actually used for training and the rest 10% for validation.

4.1 Linear Regression

In Linear Regression, we used root mean square error and used gradient descent to update the weights involved. We used sklearn linear regressor model for training.

4.2 Neural Network

1 input layer consisting of the data features. 2 hidden layers with 32 nodes each using ReLU as activation function. 1 output layer with a single node predicting the predicted output value, using sigmoid activation function. The epoch used was 100.

5 RESULT

5.1 Method 1

Here we trained the model using only Linear Regression. We achieved an accuracy of 61%.

5.2 Method 2

Here we trained the model using only Neural Networks with the above specified parameters. We achieved an accuracy of 63-65%(since each time we selected the training data randomly).

5.3 Method 3

Here we trained both Linear Regression and Neural Network model separately and independently by random shuffling the dataset. Then we averaged the predicted output and got an accuracy of 63% approximately.

5.4 Method 4

In this method, we used boosting concept of ensemble learning. First we trained using Linear Regression and then fed only those data features where the error in prediction came to be more than 800 to the Neural Network model. We considered less than 800 error to be close and more than 800 to be far apart. We got an average accuracy of 61% in Linear Regression and 53% in Neural Network. The final accuracy after taking 80-20 weighted average was 59%.

5.5 Method 5

In this method, we did basically the same as the above but this time we used Neural Networks both the times using the same parameters. The accuracy we got the first time was 64.2% and 51.5% in the second time. Average weighted output had an accuracy of 58.7%.

5.6 Method 6

Here, we found out how much of the training data was Neural Network giving close error. This time, we squeezed the close error definition as those errors which were less than 200. Percentage of data having close error came out to be 9.9%. The first round gave us 61.1% accuracy. We fed the rest 90.1% of the data which had high error(¿200) to another Neural Network model with different parameters. Here we got an accuracy of 63.7%. Then while taking the weighted average, instead of giving both the rounds equal weights, we gave weights based on the ratio found out earlier,

i.e. 9.9% of the first output and 90.1% of the second output to find out the final output. The accuracy we achieved was 63.8%.

6 CONCLUSION

We saw that both Linear Regression and Neural Networks were giving accuracy in the range of 60-65%. This is the best one can get on this dataset since for each attribute in the dataset, there are a lot of classes/values possible. Even in online competitions, this was the accuracy range achieved.
Ensembling the models did give us better results than just blind mean of the accuracies. In the second approach where we fed the high error data only to the next model, we saw accuracy dropping by 7-8%. One reason behind that can be the fact that high error predictions were less in number and thus the second model had less training data to work with. Another reason could be that the high error data features might have been primarily made up of outliers and hence converging the error was difficult. The bottomline is that the BigMart dataset is best approximated by Neural Networks.

7 EFFORT

7.1 Fraction of time spent in different parts of the project

Understanding objective and dataset: 10%
Pre-processing data: 40%
Tuning parameters for best results: 25%
Trying out different methods: 25%

7.2 Most challenging part

Understanding why the accuracy for different methods were fluctuating was the most challenging part.

7.3 Contribution

Raj Krishna Srivastava- Neural Networks and its variants in ensembling(coding).
Vinod Kushwaha- Background study, Linear Regression Model and Boosting(Coding).
Gourab Dipta Ghosh- Preprocessing, Ensembling(support), Latex and Slides.
Rohit Kumar- Background study, Latex.