CS 514 – Applied Artificial Intelligence

Kaggle Project on All State Claims Severity

Description:

All state is a currently developing automated method for predicting the cost, and hence severity, of the insurance claims. It consists of train and test data. The given train data has 132 columns and 188318 rows of which 116 are categorical values and 14 continuous values. The loss needs to be predicted. Since most machine learning algorithms work in the empirical space, we need to transform the given categorical data into continuous. The different steps performed are as described below:

Installation:

The platform used is Python 3.5. I installed it and also the basic libraries such a pip, numpy, pandas, scikit-learn, and Anaconda for Windows. In order to do preprocessing on the data, the Pandas library has been installed as it handles data frames very well.

Data Preprocessing:

The categorical data needs to be converted to continuous data. Hence, One Hot Encoding technique as well as a Regular Transformation method has been used for the transformation of data. The Regular Transformation Method involves assigning a unique value to each of the dataset values using a dictionary. The transformed data was stored in a data frame along with the continuous values. The user is given an option to choose with the Regular Transformation method or the One Hot Encoding method.

Data Learning:

The transformed data frame is then passed to the splitData function where we divide the data to 80% training and 20% testing. This way of division of data works faster than doing cross validation on the data frame. Hence, I've used the 80-20 way of splitting. 80% of the training set was sent for learning and 20% for testing. The training frame was sent to the algorithms to do learning of the dataset. The evaluation metric used is Mean Absolute Error.

Results:

The following eight machine learning algorithms have been considered. The Mean Absolute Error between the actual values of the loss column vs. the predicted values is used as comparison of the results among the various machine learning algorithms. The duration taken to run each algorithm is also calculated and displayed in the output snippets. The statistics of the results of Mean Absolute Error obtained for all the eight algorithms are as shown below:

Machine Learning Algorithm	Mean Absolute Error (One Hot Encoding)	Mean Absolute Error (Regular Transformation)
Linear Regression	6789.90	1340.04
Lasso Regression	1307.81	1339.31
Elastic Net Regression	1447.04	1420.68
Ridge Regression	1315.80	1331.80
K Neighbors	1413.52	1762.37
Ada Boost	4178.76	3608.73
Gradient Boosting	1329.62	1253.75
XG Boost	**1200.67**	**1195.28**

The best running algorithm was **Lasso Regression** for **One Hot Encoding** and **XG Boost** for Regular Transformation method as shown above.

Best Algorithm for One Hot Encoding & Regular Transformation Method -XGBoost:

XGBoost is a short hand for "Extreme Gradient Boosting". XGBoost is based on Gradient Boosting as the original model. It is a regression analysis method that is used for supervised learning problems, where we use the training data (with multiple features) to predict the target variable.

It takes as an input the training frame and the testing frame obtained on splitting. Learning is performed on the training set and the values are predicted for the testing data set. The mean absolute error is calculated for the actual values versus the predicted values. The mean absolute error obtained using One Hot Encoding method is **1205.47**.

Output: The screenshot of the results is as shown below for One Hot Encoding

```
C:\Users\User\Desktop\Sem 3\Applied AI\HW5\Project\github Upload>python extendedAllStateSeverity.py
Kindly select the type of transformation you would like to run:
1. Regular Encoding Method
2. One Hot Encoding Method
Please select the option (1/2)
2
Executing One Hot Encoding....
Splitting data....
Train data size: 150654
Test data size: 37664
Executing XGBoost...
XGRegressor:1200.67
Check your folder for the file 'Predicted_loss_values.csv' to see the final predicted results for the Test Data
-------- 4089.1516355566846 seconds-------
```

The mean absolute error obtained using Regular Transformation is **1195.28**.

Output: The screenshot of the results is as shown below for Regular Transformation

```
C:\Users\User\Desktop\Sem 3\Applied AI\HW5\Project\github Upload>python extendedAllStateSeverity.py
Kindly select the type of transformation you would like to run:
1. Regular Encoding Method
2. One Hot Encoding Method
Please select the option (1/2)
1
Executing Regular Tranformation....
Splitting data....
Train data size: 150654
Test data size: 37664
Executing XGBoost...
XGRegressor:1195.28
Check your folder for the file 'Predicted_loss_values.csv' to see the final predicted results for the Test Data
-------- 416.074153333359 seconds-------
```

Code and running details:

The code is attached in the file named, allStateSeverity.py. It's a python file.

To run this, you would need to have:

- Python 3.5 installed.
- The python code file, the train data from Kaggle in the same folder.
- Install the packages as mentioned above.
- Open the command prompt.
- Change the path to the folder containing these files.
- Run the command **python allStateSeverity.py** in the command prompt.
- After the code runs, it shows the train and test data size, followed by the result of MAE of Lasso regression for One Hot Encoding method of transformation and the result of MAE of Gradient Boosting for Regular Transformation Method. Also a new file named 'Predicted_loss_values_OneHotEncoding.csv' and 'Predicted_loss_values_RegularTranformation.csv' would be created in the same folder respectively depending upon the choice selected by the user. These files contain the results predicted for the loss column along with the index column.

Hardware Specifications:

- i7 Processor
- Quad Core
- 8GB RAM

Operating System:

- Windows 10 Professional

Threats to Validity:

The results obtained can vary due to the random selection of the split data for training and testing. Also, due to different specifications of the running environment such as the RAM, processor speed etc. The

running times might also vary from one platform to another. My running time is around 6 mins for One Hot encoding and around 2 mins for Regular Transformation Method. The MAE for the algorithm can vary with +- (10) units.

Snippets of the csv file generated:

1. **'Predicted_loss_values_OneHotEncoding'**

Index	Loss
233824	3139.143
69520	2472.782
81359	3450.959
11498	670.868
555461	2009.412
21012	7537.544
543488	2468.104
10892	940.353
212816	1718.494
189942	2241.367
40280	1422.318
440359	1420.529
561292	2900.245
133011	6566.744
566118	4566.611
442408	2567.614

2. **'Predicted_loss_values_RegularTransformation.csv'**

A	B
Index	Loss
233824	2519.08
69520	1945.974
81359	3169.658
11498	1157.643
555461	2208.352
21012	6841.332
543488	2230.346
10892	1388.084
212816	1983.084
189942	2202.826
40280	1867.051
440359	1907.504
561292	3188.892
133011	5376.006
566118	4678.975
442408	2055.093

Thank you!