AMS 598 Project 3

Jiecheng Song
111783762
Oct 2021

1. Introduction

There are two datasets under gpfs/projects/AMS598/Projects/project3:

Train_project3.csv and Test_project3.csv

The goal is to predict the sale price (variable SalePrice) of a particular piece of heavy equipment at auction based on its usage, equipment type, and configuration. The data is sourced from bulldozers Kaggle competition.

Use Train.csv to train your model and apply your model to Test.csv to forecast sale prices.

Use Spark (e.g. pySpark) for this project. Submit a .csv file that contains your predicted sale price fro each SalesID, and also write a report about your analyses.

2. Methods

Since SalePrice is numerical variable, so I decide to use a regression method, and we have lots of string variable, so the project is separated into two parts, data preprocessing and model fitting.

2.0 Replace ‘,’ in fiModelDesc with ‘|’

Since ‘.csv’ file is separated by comma, but there are also commas in fiModelDesc, so texteditor was used in this step, and replace the commas in fiModelDesc with vertical line to avoid the data reading problem.

2.1 Data glimpse

There are **53 variables** in train data, including one ID variable, one response variable (SalePrice) and 51 predictor variables. And there are **52 variables** in test data (no SalePrice).

There are **300000 data samples** in train dataset and **101125 data samples** in test data set.

In the 51 predictor variables, there are 4 integers type variables (MachineID, ModelID, datasource, YearMade) , and others are strings type.

We viewed the **distinct values** for each variable (**details table in jupyter notebook**), most variables only contain a few (less than 20) distinct values (categories), even some string variable contain many categories (a few thousands), but still small compare to the sample size (300000). So we decided to treat **most variables as categorical**, except **a few special variables**.

2.2 Special variables

a. MachineID
   MachineID is an integer variable, and there are **341027 distinct value** of MachineID in the **401125 total samples** (train + test). By common sense, the ID shouldn't have a linear relationship with SalePrice and according to the distinct data, we also shouldn't treat it as categorical (too less data in one category). So we delete MachineID variable.

b. ModelID, datasource, YearMade
   These 3 variables are all interger type. By common sense, the relation between these 3 predictors and SalePrice may not be linear, and the distinct value in these 3 variable are at most a few thousand, far less than data samples, so we treat these 3 variables as categorical.

c. MachineHoursCurrentMeter
   MachineHoursCurrentMeter is string type in the dataset, but as our ob servation, MachineHoursCurrentMeter is actually integer type and the re are more than 10k distinct values, also it should be correlated with SalePrice, so treat it as numerical variables.

d. SaleDate
   SaleDate is string type, but it is not reasonable to treat it as categorical, so we extract the month and year out, generate two new variables, month and year, and delete the variable SaleDate. The month and year are treated as categorical.

2.3 Imputation

There are plenty missing values in the dataset, to avoid coding error, we imputed the missing values in the **categorical type variable** with a special string **'NA'**, and for **numerical type variable**, we use **0** for imputation.

2.4 Generate dummy variables

To avoid the situation that some categories only show up in train data, but not in test dataset, so we union two data set as a new dataframe 'df' and generate dummy variable with the code by Wenqiang Feng in the reference.[1]

Store dummy variables in one dataframe df_dummy, and use inner join with the train data and test data on SaleID, then we can get the train_dummy dataframe for train model, and test_dummy dataframe for prediction.

There are 3 columns in train_dummy dataframe, 'SaleID', 'feature' (the dummy vector), and 'label' (SalePrice) and 2 columns in test_dummy dataframe 'SaleID' and 'feature'.

2.5 Fit model

At first ramdom forest regressor was selected, but due to hardware limitation, (my laptop out of memory due to the large dummy vectors), so finally **ordinary linear regression** was adopted.

3. Evaluation

After fit train data set, we got following summary result.

Mean squared error: 86634417.443971, RMSE: 9307.761140
Multiple R-squared: 0.828548, Total iterations: 100
Adjusted R-squared: 0.820742

Although RMSE looks large, this may be caused by the large range of SalePrice. From R2 and adj R2, we can find the model fits well in training dataset.

Detailed information and corresponding PySpark code for each step was stored in the jupyter notebook (project3.ipynb) and prediction for test data is stored in the .csv file. (prediction.csv)

[1] The dummy variable code can be found in https://runawayhorse001.github.io/LearningApacheSpark/pyspark.pdf