Linear Regression and Cross Validation of Regression on Financial Datasets

ABSTRACT

PURPOSE: To explore the use of Linear Regression and K-Fold Cross Validation to evaluate the best proxy commodity to analyze a market for a given financial dataset.

METHODS: Linear Regression and K-Fold Cross Validation algorithms are explored in this report. There are 2 prime functions called by the parent function, a2q1 and a2q2.

The first uses the input dataset, goods.csv, to output the the Root Mean Square (henceforth referred to as RMS) Error values of the linear regressions for each commodity based on predicted values calculated using other commodities as independent variables, and then find the index of the commodity with the lowest RMS Error value.

The second conducts a 5-fold Cross Validation producing training and testing sets from different partitions of data, and outputs their respective RMS Errors.

The use of an intercept term and non-standardized data, and their implications are discussed at length

RESULTS: The results intuitively demonstrate the effectiveness of the model for the given dataset with some variance in RMS Errors across training and testing sets using data in Tables 1, 2 and Figure 1.

CONCLUSIONS: We concluded that this algorithmic model is well suited for the variables in the given dataset, goods.csv, but likely not entirely suitable for datasets with high variance which would produce relatively skewed output, biasing interpretation of real-world data

Word Count: 218

INTRODUCTION

The purpose of this exploration was to explore the use of Linear Regression and 5-Fold Cross Validation to evaluate the best proxy commodity to analyze a market for a given financial dataset.

Linear regression is a tool that models a relationship between a dependent variable and one or more independent variables, to produce a line of best fit that typically minimizes the gap between the actual and predicted values. Real life applications of it are endless, since there exist countless relationships between features in the real-world which can be modelled by this tool. K-Fold Cross Validation is used to evaluate the effectiveness of data analysis models based on dividing a model into training and testing sets and simulating a model that generalizes new data based on previously fed data.

The question being explored is the effectiveness of a linear regression algorithm in conjunction with a 5-Fold Cross Validation model in evaluating the best proxy commodity using which a market of associated goods can be modelled and simulated.

METHODS

Both functions, a2q1 and a2a2, are called by the parent function a2_20292366. They both initially read data from goods.csv, skipping the first row and column, and store the size of the dataset in a vector.

Function a2q1 initializes the vector rmsvars with zeroes, and then iterates through all columns and finds the RMS errors of linear regression for each commodity by treating them as dependent variables and the rest of the goods as independent variables. It starts the iteration of columns by extracting the dependent variable vector (y_vec) and extracting the independent variable matrix (X_mat) from the data. It concurrently adds an intercept term by adding a column of ones at the beginning of the matrix. This decision and its implications are discussed in detail later.

The iterator then calculates the weight vector (w_vec) by running a regression on X_mat and y_vec. The result of the RMS Error analysis using the rms function is stored in the rmsvars vector. After the iteration is completed the index of the commodity with the lowest RMS Error value is stored in lowndx. The predicted values for this commodity are then calculated in the same manner as the function. The plot function is then used to graph the results of our computation of the regression.

Function a2q2 initializes X_mat using all commodities except the one with the lowest index, and y_vec stores the commodity vector for Uranium. The use of rng('default'); resets the random number generator to ensure that any future calls to random number generation functions will output the same sequence every time the function is called, which helps in output moderation and consistent interpretation.

rmsTrain and rmsTest vectors are initialized to store the RMS Error values calculated for the training and testing sets later. foldSize is computed by roughly dividing all observations into 5 partitions using the floor function that rounds to the closest integer.

The iterator then goes through each fold that is randomly generated using the testIdx variable that randomizes the start and end indices for each testing set, ensuring shuffled data and consequently more accurate output. The trainIdx uses the setdiff builtin to store the complement of all stored in testIdx.

The training and testing sets for X_mat and y_vec are then used to calculate the weight vector w_vec, the predicted values for the training and test vectors. The RMS Errors between these is calculated and stored in rmsTrain and rmsTest vectors respectively. Finally, those vectors are returned as rmstrain and rmstest.

Function mykfold was not utilized in this exploration. The code was checked multiple times by printing out intermediate results and intuitively comparing them to results from previous iterations of testing.

The approach this exploration takes into the analysis of goods.csv using linear regression and 5-Fold Cross Validation algorithms sets it apart from others in that it does not standardize data in the given dataset.

Standardizing data is a common practice in most algorithms for the simple reason that it transforms features that may have varying scales into standardized data with zero mean and unit variance, ensuring smooth output from algorithms that may be sensitive to the scale or units of measurement. Doing so often greatly improves algorithm performance and increases overall stability.

However, standardizing data can sometimes be a hindrance since it is not a good strategy for datasets that do not follow a roughly normal distribution, as it may produce skewed output, which may be worsened by the fact that standardization amplifies the weight of outliers.

This exploration made the choice to not randomize the dataset to preserve the original units for the commodities which would lead to output that is more meaningful in practice since interpretation of standardized output isn't intuitive or easily achieved. Output data in terms of the original units is much more helpful in drawing conclusions about not only the accuracy and performance of the model, but even the dataset, as discussed in the initial paragraphs of this section.

Though the above is true, standardizing our data may have improved the performance of our model for this dataset as it may have produced more accurate results with less variance between the rmstrain and rmstest output vectors. This may have been beneficial if our dataset was much larger as well as it would have reduced computational costs.

A drawback of this for our model would have been that there would have been a loss in the weight biases that may have contributed to a more effective interpretation of the output we got.

Independent variables such as Nickel, Copper, and Zinc, have the greatest weights in terms of their quantity scale, which can be seen with a cursory glance at goods.csv, which would intuitively mean that they would be the most important actors in the calculation of the training sets, consequently contributing to the overfitting of the data.

Another interesting diverging aspect of the approach this exploration takes in doing the linear regression is the use of an intercept term which is done by adding a column of 1s to X_mat (independent variables matrix).

The use of an intercept term helped establish a baseline prediction and increases the interpretability of the relationships between the independent variables and the dependent vector. This may have also led to a decreased impact of outliers in our dataset, since they would only be able to affect the slope and not the intercept, but it wasn't a problem for this dataset as it had relatively low variance.

However, the use of an intercept term may have contributed to the model overfitting the data, which could explain the high variance in the testing sets.

RESULTS

Table 1 contains the RMS Error values between actual and predicted datapoints for each commodity upon application of a linear regression model based on all other goods.

Table 2 contains the RMS Error values for the training and testing sets from 5 different folds created randomly from the dependent vector and the independent variable matrix.

Figure 1 outlined below shows the linear regression plot for commodity best modelled by the rest of the commodities, which in our dataset turns out to be Uranium.

Table 1: RMS Error values, scaled to proper units by multiplying them by 1.0e+03, between actual and predicted datapoints for each commodity upon application of a linear regression model based on all other commodities

RMS Error Values from Linear Regression for each Commodity	
Commodity	**RMS Error Value**
Zinc	0.2328
WTI_Crude	0.0074
Uranium	0.0062
Tin	1.2200
Copper	0.3152
Hard_Logs	0.0304
Soft_Logs	0.0125
Hides	0.0095
Lead	0.1911
Nickel	2.2748
Rubber	0.0099
Soft_Sawn	0.0178
Fish_Meal	0.1578
Cotton	0.0093
Coal	0.0107
Iron_Ore	0.0115
Hard_Sawn	0.0472
Zinc	0.2328

Table 2: RMS Error values, in terms of the raw commodity units, for the training and testing sets from the 5 different folds created randomly from the dependent variable vector (y_vec) of the best modelled commodity and the independent variable matrix (X_mat) with the rest of the goods .

	Cross-Validation RMS Errors in testing and training sets				
foldNumber	1	2	3	4	5
rmstrain	5.7229	5.7203	5.7821	6.2058	6.4089
rmstest	9.3635	12.25	8.6243	7.2156	6.1905

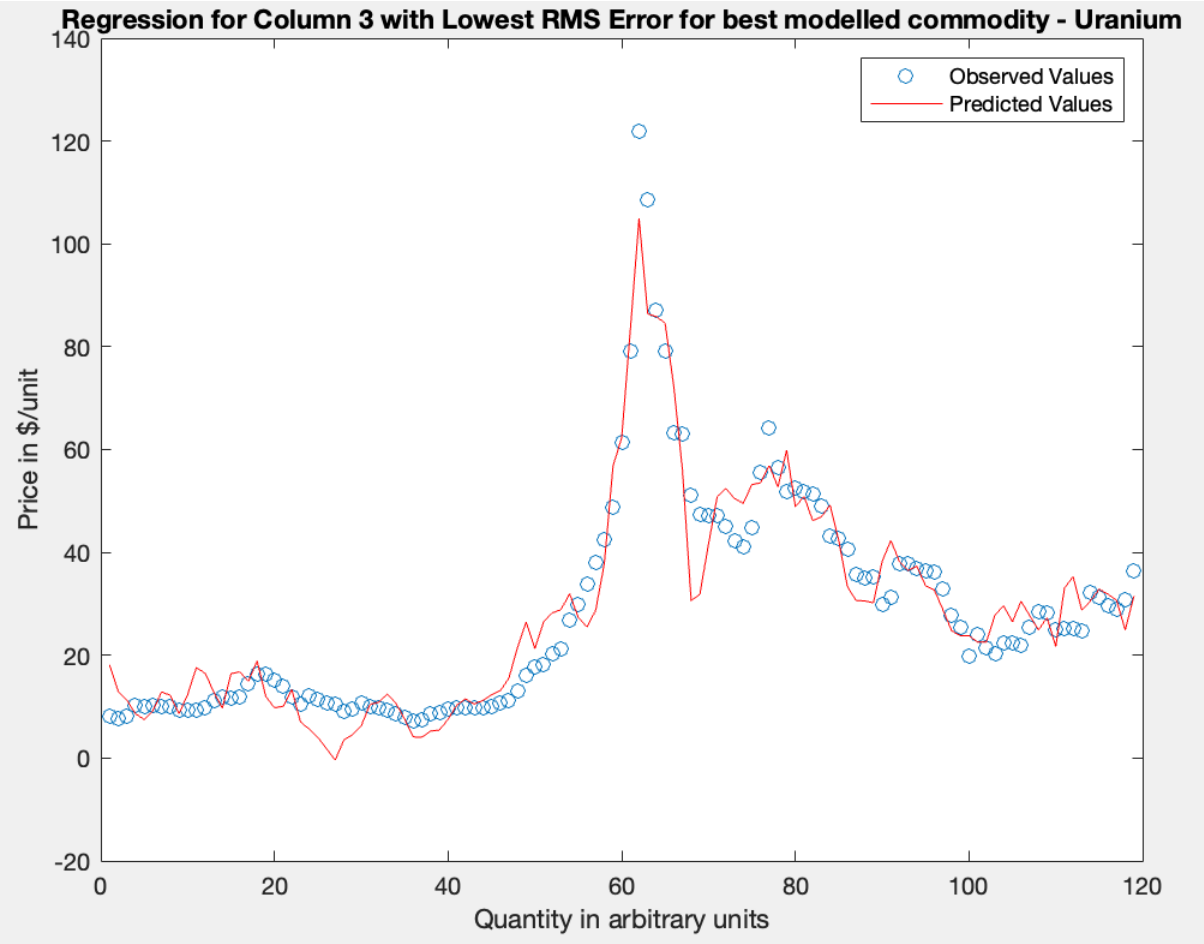

Figure 1: Linear regression plot for commodity best modelled by the all the rest of the commodities as independent variables (lowest RMS Error values upon application of linear regression), which turns out to be Uranium.

DISCUSSION

The results section shows the raw output from the analysis done on the goods.csv dataset using linear regression models, a simple cross validation algorithm, and RMS Errors which are useful in drawing meaningful conclusions.

Table 1 of the results section shows RMS Error values as output by the a2q1 function in the rmsvars vector ranging from 0.0062, for Uranium, to 2.2748 for Nickel. These values can be scaled to proper raw data units by multiplication with 1.0e+03. In their raw form, they are highly difficult to interpret intuitively, which is why MATLAB scales them logarithmically to produce a more meaningful and palatable output.

The mean of the RMS Error values for the linear regression for the commodities in goods.csv is approximately 0.2665*1.0e+03, with a standard deviation of approximately 0.5769*1.0e+03. This in its raw form indicates that there is minimal variance in the dataset. Minimal noise and outliers in the dataset would contribute to the RMS Errors for the models being small and would result in the linear regression and cross validation algorithms producing more accurate and meaningful data.

Our function a2q1 determined variable index 3 (lowndx) in our dataset, Uranium to have the smallest RMS Error between the actual values from the dataset and the predicted values calculated through linear regression using the other commodities as independent variables.

As seen in Figure 1, the actual data points in circular blue markers for Uranium fit the red regression line modelled on the other commodities as independent variables relatively well, showing graphically that Uranium best modelled by our model in terms of price in $ against arbitrary quantity units. The independent variable matrix even models the spike at around 60 arbitrary quantity units quite accurately.

This was unexpected as Uranium is not a highly transacted good with multiple barriers to entry in a standard government-moderated market. I would have expected a commodity such as Iron Ore or Coal to have been the best proxy good that fits the regression that our model produces for each dependent variable, since they comprise a huge subset of raw material transactions in the real world which are exponentially higher than the other ones described.

Using the second-best fitting commodity might be of interest in real-world analytics and may possibly suit practical interpretations better as WTI_Crude or crude oil, which has an RMS Error value of 0.0074, is one of the most heavily traded commodities internationally and possibly the best practical proxy good that demonstrates the trends in a goods market.

The best proxy good could be hypothesized intuitively to have the greatest weight bias in a dataset which may be another good exploration for this dataset.

Our function a2q2 determines the RMS Error values for the training and testing sets that we create using X_mat (independent variable matrix) that contains datapoints for every commodity other than Uranium, and y_vec (dependent variable vector) which contains the actual datapoints for Uranium. The output from this function, seen in Table 2, shows the RMS Error values between the training and testing sets.

A brief glance shows the RMS Errors in the testing sets to be approximately 2.45 raw data units higher than those for the training sets at average, with the average variance between training and testing RMS Errors to be approximately 6.5 raw data units.

Testing data is the 'mini-dataset' that we use to evaluate the model's accuracy in predicting new data. The model likely overfitted the training data which subsequently lead to higher RMS Errors.

Another explanation is that the testing data sets had varying statistical properties which in our case seems unlikely since the rows in the testing and training data folds are randomized decreasing the likelihood of biased testing sets.

The 5^{th} fold has a standard deviation of 0.15 compared to the 2^{nd} fold which has one as high as 4.62. This striking difference fits our overfitted model hypothesis but might be contributed to by some noise in the testing datasets, accounting properly for which is a considerable task.

We cannot effectively say that this linear regression and Cross-Validation model would be a good predictive tool in making decisions about price vs quantity in a commodities market in the real world, since the relationship, best modelled by Uranium, is not entirely free of bias and has considerable variance between training and testing sets.

This model is good for expanding datasets of stable variance but adding new ones with higher variance may render it inaccurate, and the interpretations drawn from it, ineffectual in practice. This may be counteracted to some extent using a greater number of folds in training and testing the model.

We can conclude that this algorithmic model is well suited for the variables in the given dataset, goods.csv, but not entirely suitable for datasets with high variance which would produce relatively skewed output, biasing interpretation of real-world data.