Predictive Analysis of Ethanol Prices with Machine Learning

Benjamin M. Schilling

Computer Science, Minnesota State University Moorhead, 1104 7th Avenue South, Moorhead, MN 56563

Introduction

Using historical data regarding Ethanol and Corn prices, a predictive regression will be built using a train/test machine learning algorithm. This process will consist of ETL data prep, data split, regression training, and model review. A successful model will have a strong coefficient of determination as well as a low mean squared error.

Prep and Validation of Raw Data

Data extraction, transformation, and loading is done in primary process of model creation (fig. 1). Correlation between key variables of each dataset is calculated to determine whether or not a predictive model will function (fig 2). Each price metric showed moderate correlation and percent change showed weaker correlation. Though the correlation of percent change between corn and ethanol prices is not ideal, this figure is still significant enough to continue with predictive model creation.

(fig. 1) Merge/Scrub of Corn & Ethanol Commodity Data

```
In [6]: ethanolCornDF = cornPriceDF.merge(ethanolPriceDF, left_on='cornDate', right_on='ethanolDate')

In [7]: ethanolCornDF['cornChange']=ethanolCornDF['cornChange'].map(lambda x: x.rstrip('%'))
        ethanolCornDF['ethanolChange']=ethanolCornDF['ethanolChange'].map(lambda x: x.rstrip('%'))
        ethanolCornDF['cornVol']=ethanolCornDF['cornVol'].map(lambda x: x.rstrip('K'))
        ethanolCornDF['ethanolVol']=ethanolCornDF['ethanolVol'].map(lambda x: x.rstrip('K'))
        ethanolCornDF['cornChange'] = pd.to_numeric(ethanolCornDF['cornChange'])
        ethanolCornDF['ethanolChange'] = pd.to_numeric(ethanolCornDF['ethanolChange'])
```

(fig. 2) Pearson Correlation Coefficient of Corn & Ethanol Commodity Prices

```
In [8]: ethanolCornDF.corr(method="pearson")
```

Out[8]:

	cornPrice	cornOpen	cornHigh	cornLow	cornChange	ethanolPrice	ethanolOpen	ethanolHigh	ethanolLow	ethanolChange
cornPrice	1.000000	0.998398	0.999181	0.999331	0.033308	0.518872	0.516805	0.522461	0.512152	0.010337
cornOpen	0.998398	1.000000	0.999306	0.999243	-0.014461	0.518168	0.517176	0.522330	0.511885	-0.014357
cornHigh	0.999181	0.999306	1.000000	0.999022	0.008632	0.520787	0.519354	0.524795	0.514271	-0.002129
cornLow	0.999331	0.999243	0.999022	1.000000	0.010044	0.515579	0.514119	0.519421	0.509217	-0.000783
cornChange	0.033308	-0.014461	0.008632	0.010044	1.000000	0.014188	-0.001068	0.006473	0.007979	0.384273
ethanolPrice	0.518872	0.518168	0.520787	0.515579	0.014188	1.000000	0.998396	0.999070	0.999236	0.046864
ethanolOpen	0.516805	0.517176	0.519354	0.514119	-0.001068	0.998396	1.000000	0.999205	0.998944	0.016638
ethanolHigh	0.522461	0.522330	0.524795	0.519421	0.006473	0.999070	0.999205	1.000000	0.998435	0.030195
ethanolLow	0.512152	0.511885	0.514271	0.509217	0.007979	0.999236	0.998944	0.998435	1.000000	0.033912
ethanolChange	0.010337	-0.014357	-0.002129	-0.000783	0.384273	0.046864	0.016638	0.030195	0.033912	1.000000

(fig. 3) Aggregation of Independent Variables and Split for Train & Test

```
y = ethanolCornDF.iloc[:,6].values
X = ethanolCornDF.iloc[:,[0,1,3,4,5]].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.05, random_state = 0)
```

Data Split for Ethanol Price Prediction

Data is then split into dependent and independent variables, X and y respectively. The independent variables initially consisted of all Corn related price columns, including cornPrice, cornOpen, cornHigh, cornLow, cornVol, and cornChange. The dependent variable and target for prediction is ethanolPrice. This split data will serve two pursposes: first to analyze which of the independent variables are inhibiting the model from an accurate prediction, and also to be split further into data for training and testing the predictive regression algorithm. The clustering of independent values and split of all data into test and training datasets can be seen in figure 3.

Review of Statistic Summary of Input Variables

Using the aggregated independent variables in a summary against the dependent variable, the R-squared value is reviewed as well as the P-value for each of the independent variables (fig 4). Independent variables with a significance (P-value) greater than .05 are deemed as insignificant and removed from the dataset. This process is repeated until all independent variables have are statistically significant as it will produce the most accurate results for our predictive model.

(fig. 4) Summary Statistic of All Independent Variables vs Dependent Variable

```
X2 = sm.add_constant(X)
X2 = X2[:,[0,1,3,4,5]]
est = sm.OLS(y,X2)
est2 = est.fit()
print(est2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.546
Model:                            OLS   Adj. R-squared:                  0.546
Method:                 Least Squares   F-statistic:                     983.4
Date:                Wed, 15 Apr 2020   Prob (F-statistic):               0.00
Time:                        12:29:32   Log-Likelihood:                -839.77
No. Observations:                3275   AIC:                             1690.
Df Residuals:                    3270   BIC:                             1720.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.9253      0.021     43.708      0.000       0.884       0.967
x1             0.0044      0.001      3.769      0.000       0.002       0.007
x2             0.0079      0.001      8.265      0.000       0.006       0.010
x3            -0.0101      0.001     -9.311      0.000      -0.012      -0.008
x4            -0.0009   6.91e-05    -13.383      0.000      -0.001      -0.001
==============================================================================
Omnibus:                     2294.558   Durbin-Watson:                   0.082
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            50686.036
Skew:                           3.055   Prob(JB):                         0.00
Kurtosis:                      21.278   Cond. No.                     3.19e+03
==============================================================================
```

Training the Regression and Reviewing Results

Once the significant P-Values have been determined, the predictive regression can be trained (fig 5). The predictive regression is then calculated by passing the independent test variables into the regression determined by training variables. The results are then output (fig 6) to show our coefficient of determination and mean square error. There is a strong coefficient of determination showing that our predictive model accurately depicts our real data. Our mean squared error also shows our model was accurate with a value of .05. With our dependent values ranging from 1.15 to 4.23, our mean squared shows that our model predicted on average within 1.6% of the actual values.

(fig. 5) Training Linear Regression and Obtaining Dependent Predictive Regression

```
regressor = LinearRegression()
regressor.fit(X_train, y_train)

y_pred = regressor.predict(X_test)
```

(fig. 6) Output of Coefficient of Determination, Mean Squared Error, and Plot Visualization

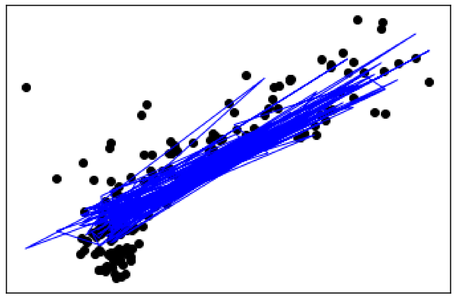
```
print('Coefficients: \n', regressor.coef_)
# The mean squared error
print('Mean squared error: %.2f'
      % mean_squared_error(y_test, y_pred))
# The coefficient of determination, 0 lowest - 1 Highest
print('Coefficient of determination: %.2f'
      % r2_score(y_test, y_pred))

#
plt.scatter(X_test[:,0], y_test,  color='black')
plt.plot(X_test[:,0], y_pred, color='blue', linewidth=1)

plt.xticks(())
plt.yticks(())

plt.show()
```

```
Coefficients:
 [ 0.00865584  0.00154195  0.00499421 -0.01307548 -0.00092288 -0.01011531]
Mean squared error: 0.05
Coefficient of determination: 0.73
```



References

1. Bonner, A. (2019, April 6). The complete beginner's guide to machine learning: simple linear regression in four lines of code! Retrieved from https://towardsdatascience.com/simple-linear-regression-in-four-lines-of-code-d690fe4dba84
2. Brownlee, J. (2020, March 17). How to Calculate Correlation Between Variables in Python. Retrieved from https://machinelearningmastery.com/how-to-use-correlation-to-understand-the-relationship-between-variables/
3. Ethanol History | Markets Insider. (2020, March 14). Retrieved March 14, 2020, from https://markets.businessinsider.com/commodities/historical-prices/ethanol-price/usd/4.1.2006_4.2.2020
4. Feed Grains: Yearbook Tables. (2020, March 14). Retrieved March 14, 2020, from https://www.ers.usda.gov/data-products/feed-grains-database/feed-grains-yearbook-tables/
5. R, V. (2020, February 23). Feature selection - Correlation and P-value. Retrieved from https://towardsdatascience.com/feature-selection-correlation-and-p-value-da8921bfb3cf
6. Raj, Y. (2020, February 23). ML: Multiple Linear Regression using Python. Retrieved from https://www.geeksforgeeks.org/ml-multiple-linear-regression-using-python/