US Natural Gas Prices - Predicting Future Performances

1.0 Introduction

The growth of the energy sector is challenged by what would appear to be an enduring era of low cost natural gas. This project will evaluate the trend in natural gas supply and demand dynamics and the impact on natural gas prices in the next years. Understanding trends in natural gas prices, consumption and sales would help investors make informed decision on product investment and predict growth dynamics in the market. Therefore one of the main focuses of this project is to analyze historical natural gas prices, establish relationship between this parameter with sales volume and demand. The prices will be compared to crude oil prices. An assessment of the market behavior of these products will enable E&P companies make informed decision on future investment in the industry.

Natural gas is a gas that consists primarily of methane and is an important fuel source around the world. Natural gas is generally created using one of two mechanisms: biogenic and thermogenic. Biogenic is created by methanogenic lifeforms that exist in shallow waters or landfills while further down in the earth thermogenic gas is created from organic material that has been buried for a great deal of time.

Natural gas is a major source of power generation, especially for heating and cooling systems in the U.S. Generally, the product is used to either power gas turbines or burned to create steam and power steam turbines instead. The fuel is often preferred to coal or oil since it is far cleaner and produces less greenhouse gases than its other fossil fuel cousins.

2.0 Data For the Study

Data for this study will be obtained from the United States Energy information Administration ; Petroleum Product Data; Gas Price Data. Alternative data source is the Henry Hub Natural Gas Spot Price: Henry Hub Spot Price Data

The study will include a compilation and analysis of certain oil and gas information as reported by publicly In EIA and will include data on US exploration and production (E&P) results for the period between 1964 through 2015. Oil prices: Oil Price Data || Production Data

2.0.1 Performance Indicators

Key performance measures and indicators would be predicted using available data. Most of the measures would be calculated based on the natural gas and spot oil price information recorded in the data base. Descriptions of some of the metrics are given below:

2.0.1.1 Average Natural Gas Price per Mcf

Average natural gas price is the price companies pay for natural gas. Because the price of natural gas can vary dramatically with commodities market movements these companies will often hedge the risk of unexpected price movements by investing in commodities futures. By doing so they are able to effectively lock in a price for the commodity over a given time period and avoid unexpected increases in the commodities' price. As such, it can be useful to

predict future trends in the the average price these companies pay for the natural gas to help them in their planning.

2.0.1.2 Natural Gas Storage Days of Demand

It is important to us to compare the data available to access the performance of the working gas storage against the 5-year average. This is obtained by dividing the overall storage by days of demand and the measure gives a better sense of the relative adequacy of inventories to meet demand

2.0.1.3 Henry Hub index

It is also useful to generate indexes and compare them with Henry Hub index which until recently is the most widely used reference for US natural gas prices.

The Henry Hub is a distribution hub on the natural gas pipeline system in Erath, Louisiana, owned by Sabine Pipe Line LLC, a subsidiary of EnLink Midstream Partners LP who purchased the asset from Chevron Corporation in 2014. Due to its importance, it lends its name to the pricing point for natural gas futures contracts traded on the New York Mercantile Exchange (NYMEX) and the OTC swaps traded on Intercontinental Exchange (ICE)

2.1 Data Wrangling

In this section of the project, missing date values in the Pandas Dataframe column would be filled. The daily natural gas price data are stored using Data Frames. There are 5105 rows in the dataset. The Dataset snapshot is displayed here (Hub_Gas_Spot_Prices). The time series data do not contain the values for Saturday and Sunday. Hence missing values have to be filled.

The procedure for filling in the missing values will involve the use of set_index from column and then resample with ffill or bfill. The dataset is re-sampled and front-filled. Back fill is not used as back filling would imply having knowledge of the future when that is a false assumption. This results in a pandas dataframe (df) with 7414 rows having all the missing days front filled with data. The resulting data is used for full time series analysis.

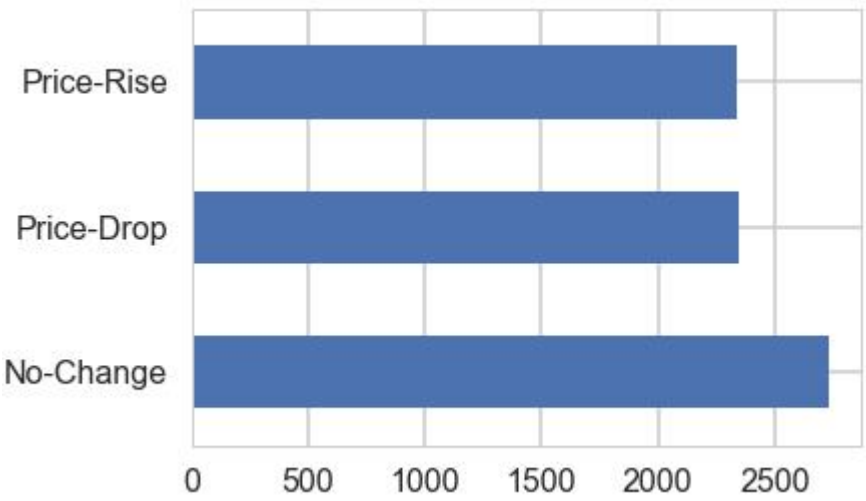
2.2 Data Visualization

Visualizing data is useful because it allows you to see relationships in data in a fast, intuitive way. It's especially helpful in exploring data and deciding what to dig into next, because it can point to places where there may be significant patterns.

2.2.1 Bar Charts

Bar charts are a visual way of presenting grouped data for comparison. Here the counts of price changes can be visualized with a bar chart from the .plot() method. To define the type of plot the .plot() is given a keyword called kind=. In this case, the keywords barh (for horizontal bar chart) is used.

In order to use the mathplot library and create plot in jupyter notebook, use %matplotlib inline. Plotting the frequency of the price changes, we have:

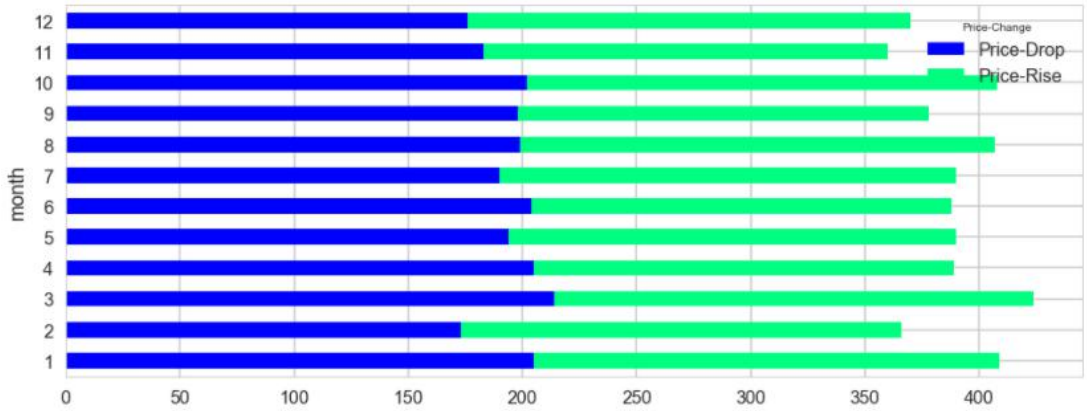


Out of the 2735 days that did not record price changes, it is worthy of note that 2310 of those days were artificial data created for the days when there were no prices.

The bar chart shows that, in principle, there is almost the same number of price drops as price rise. This indicates that there is equal likelihood of price rising or price dropping each day of sales.

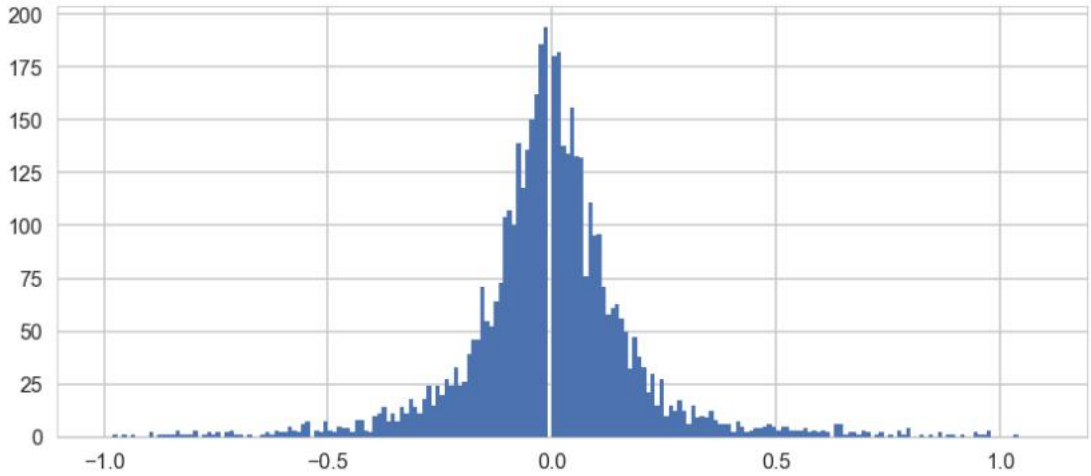
2.2.2 Plotting grouped data

Let us plot the price-drop and price-rise for every month. Use a new parameter in .plot() to stack the values vertically (instead of allowing them to overlap) called stacked=True:



2.2.3 Histogram

We can visually represent the distribution of gas price changes using a histogram. Histograms allow us to bucket the values into bins, or fixed value ranges, and count how many values fall in that bin. numpy .arange() We are interested in analyzing size of gas price changes and we want to put these sizes into bins that represent every 10 cents/MMbtu increase. We can use the numpy method .arange() to create a list of numbers that define those bins. The bins of 10 cents/MMbtu intervals will range from $10/MMbtu price drop (-10) to 10 cents/MMbtu price increase (10). The first bin will hold a count of gas price change between $10MMbtu and $9.90/MMbtu drop in price, then $9.90/MMbtu and $9.80/MMbtu, and so on.



We can see that the vast majority of price changes weren't more than 10 cents/MMbtu although the price is as likely to drop as it is likely to rise.

2.3 Data Exploration

2.3.1 Extracting Month from the DataFrame

Month is extracted from Pandas Datetime column for further monthly analysis and a new month column is created. We now want to investigate the months of the year when there is more price changes than others. This will enable us to determine whether there are special features of such months that determine price changes .

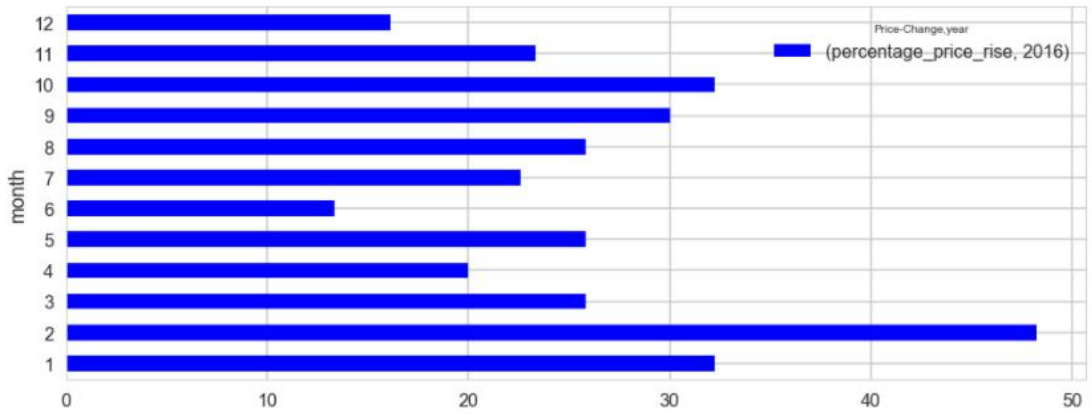
To compare price changes by months, we need to group the monthly records together. We will use the .groupby() function which allows us to group records into buckets by categorical values. Since we already have created a column in our data for the Price-Change, and also another column to indicate the month of the price change, we can simply pass those arguments into the groupby() function. This will create a segment for each unique combination of Price-Change and Month.

2.3.2 Yearly Price Volatility

Price spikes occur periodically in natural gas markets because supplies cannot quickly adjust to demand changes.

2000-2001

In 2000, natural gas supplies were constrained and demand skyrocketed, leading to the perfect environment for the price spike. 2000-2001 corresponds to the California crisis, in which prices ramped up to relatively high levels nationally, irrespective of the dynamics in California. Similar movement was observed in the actual average wellhead prices around the country.



2005-2006
Disruptions caused by severe weather, operating mishaps, or planned maintenance can also cause short-term tightness in natural gas supply. In the summer of 2005, hurricanes along the U.S. Gulf Coast (Katrina/Rita, and the 2005 Loss of Offshore Supply) caused more than 800 billion cubic feet (Bcf) of natural gas production to be shut in between August 2005 and June 2006. This is equivalent to about 5 percent of U.S. production over that period and about 22 percent of yearly natural gas production in the Federal Gulf of Mexico. As a result of these disruptions, natural gas spot prices at times exceeded $15 per million Btu (MMBtu) in many spot market locations and fluctuated significantly over the subsequent months, reflecting the uncertainty over supplies.

Mean
The mean, or the average, gives us a general idea of how many much the price changed for each year .pivot_table() calculates the mean of the aggregated values by default. We can pivot on the column year to see the mean price change aggregated by year:

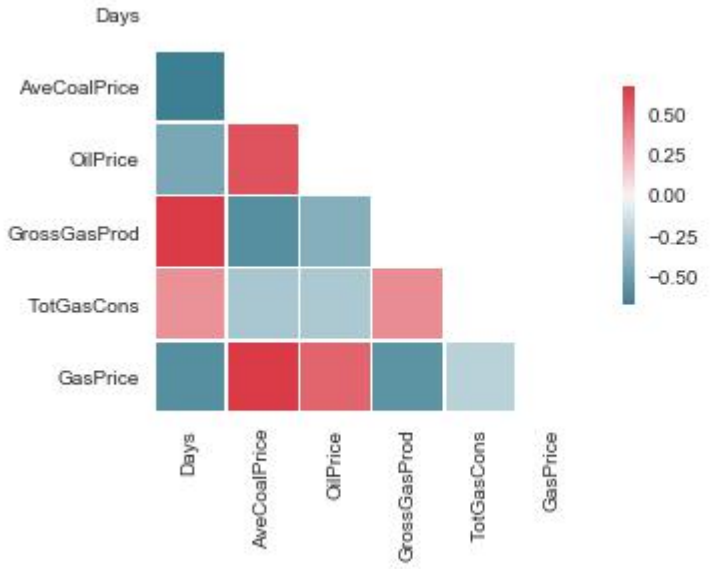
2.3.3 Basic statistics with .describe()
We will use .describe() method since we are working with numeric columns. The .describe() will enable us see a number of basic statistics about the column, such as the mean, min, max, and standard deviation. This can give us a quick overview of the shape of the data.

Before using describe(), we would select the Price-Difference series:

Here's a quick breakdown of the above as it relates to this particular dataset: count: there are 7414 rows in the dataset, which is filtered to show all the gas prices. mean: the average gas price. std: the standard deviation. min: the largest drop in the dataset. In this case, the gas price dropped the most. 25%: the 25th percentile. 25% of Gas price drop is lower than -0.03 $/MMbtu. 50%: the 50th percentile, or the median. 50% of the time Gas price dropped and 50% of the time gas price rose. 75%: the 75th percentile. 75% of gas price change is lower than 0.03 $/MMbtu. max: the highest gas price rise in the dataset: 6.50 $/MMbtu.

2.4 Correlation Matrix

Correlation matrix is used to investigate the dependence between multiple variables at the same time. The result is a table containing the correlation coefficients between each variable and the others.



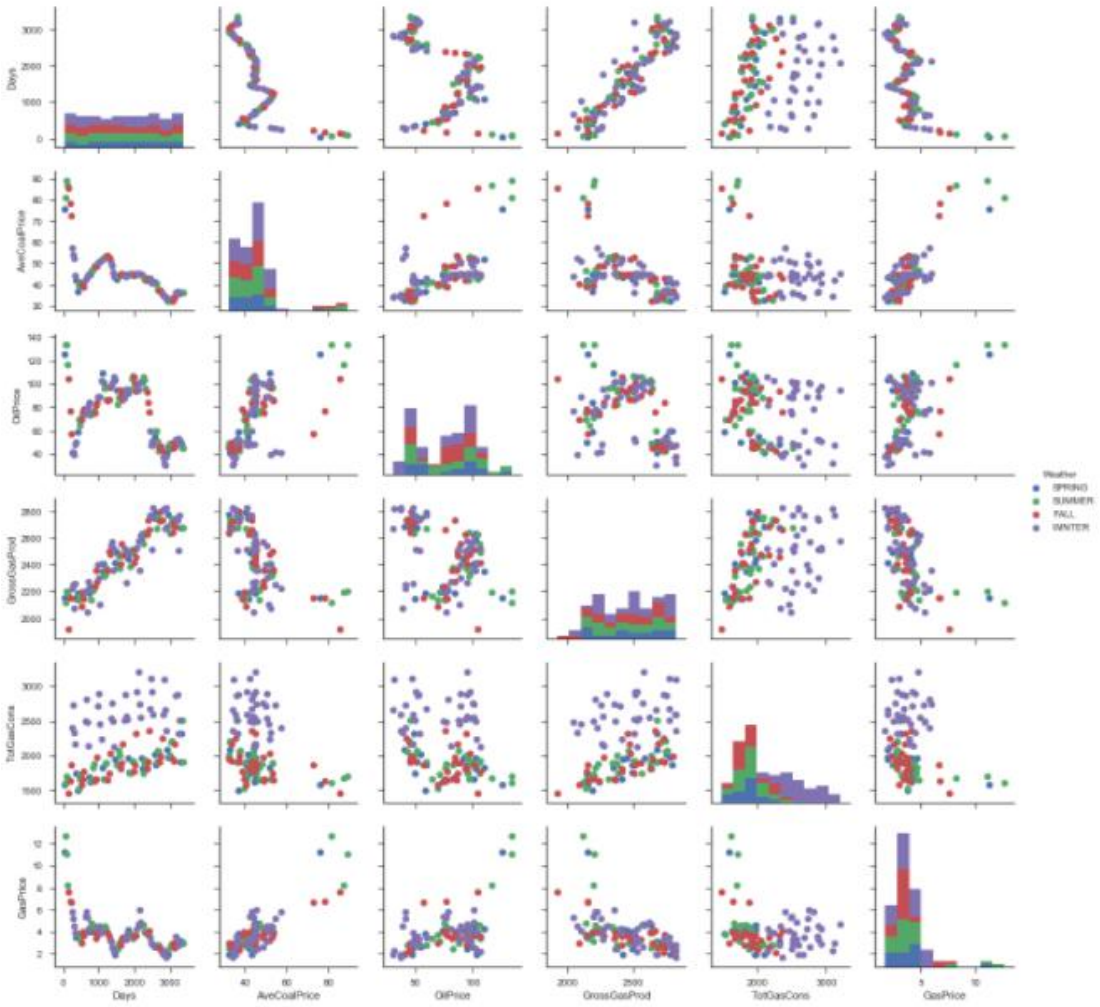
Positive correlations are displayed in red and negative correlations in blue color. Color intensity are proportional to the correlation coefficients. In the right side of the heatmap, the legend color shows the correlation coefficients and the corresponding colors.

From the heatmap above, the correlations between the variables ranges from weak 0.2 to moderate 0.6. The negative correlation between time (Days) and Gas price seems to suggest that gas price has been on the decrease since 2008. The matrix also suggests that there is negative correlation between gas price and gas production/ gas consumption. While it could be suggested that increased gas supply tends to drive down gas prices, the dynamics that maintains low gas price as total consumption increases are not well known.

2.5 Scatterplot Matrices

Scatterplot matrices are a great way to roughly determine if we have a linear correlation between multiple variables. This is particularly helpful in pinpointing specific variables that might have similar correlations to my key data.

The data set comprises multiple variables: Days, Date (Months), US Average coal price (AveCoalPrice - US$/tons), Cushing, OK WTI Spot Price FOB (Oil Price - (Dollars per Barrel)), U.S. Natural Gas Gross Withdrawals (GrossGasProd (MMMcf)), U.S. Natural Gas Total Consumption (TotGasCons-(MMMcf)), Henry Hub Natural Gas Spot Price (GasPrice (Dollars per Million Btu)), Weather and the Status of gas price (High or Low - Price higher than the average is High and below average price is Low).



In this scatterplot, it is probably safe to say that there is a correlation between Gas Price and Coal Price and also between Gas Price and Oil Price because the plots look like a line. Gas Production has steadily been on the increase, and there seems to be a logarithmic relationship between Gas production Gas price. There is probably less of a correlation between Oil Price and Total Gas Consumption in addition to Coal Price and Total gas production. More statistical analyses would be needed to confirm or deny this.

3.0 Hypothesis Testing

3.0.1 Introduction

There is one major major factor that affect natural gas prices in the United States of America. This can be developed in the form of hypothesis: Natural gas prices are mainly a function of market supply and demand.

This hypothesis is premised on the thought that there are limited short-term alternatives to natural gas as a fuel for heating and electricity generation during peak demand periods, changes in supply or demand over a short period may result in large price changes. Prices themselves often act to balance supply and demand.

3.0.1.1 Supply-Side Factors

These include natural gas production, net imports, and underground storage levels. Increases in supply tend to pull prices down, while decreases in supply tend to push prices up. Increases in prices tend to encourage production, imports, and sales from storage inventories. Declining prices tend to have the opposite effects.

3.0.1.2 Demand-Side Factors

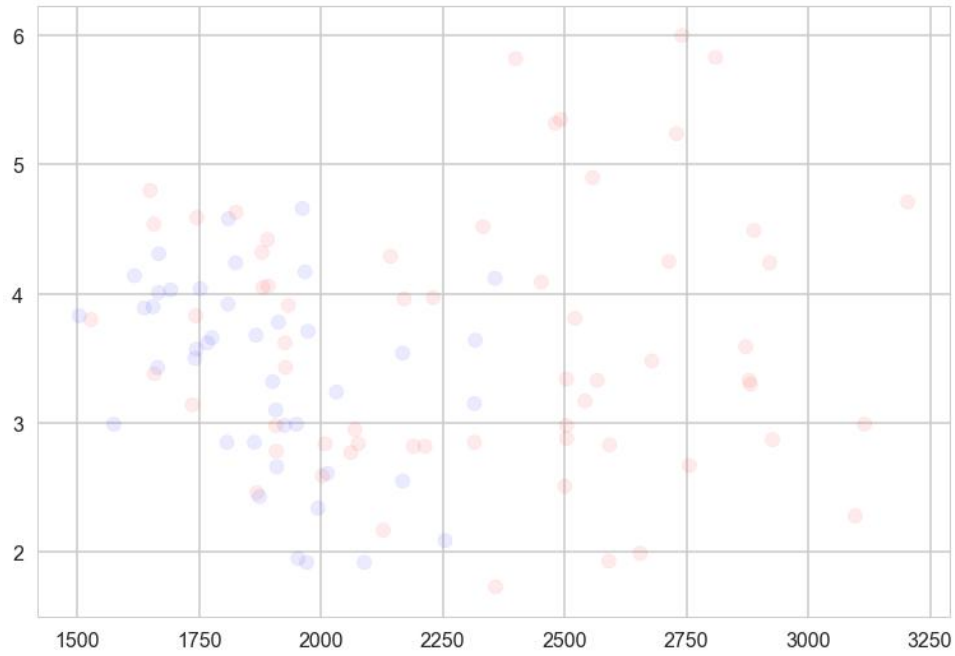
Factors on the demand-side include weather (temperatures), economic conditions, and petroleum prices. Cold weather (low temperatures) increases demand for heating, while hot weather (high temperatures) increases demand for cooling, which increases natural gas demand by electric power plants. Economic conditions influence demand for natural gas, especially by manufacturers. Demand may be moderated by petroleum fuel prices, which may be an economical substitute for natural gas for power generators, manufacturers, and large building owners. Higher demand tends to lead to higher prices, while lower demand can lead to lower prices. Increases and decreases in prices tend to reduce or increase demand.

3.0.1.3 Further Hypothesis

Existing theories about Natural gas prices performance has been based on the thinking that, as with other commodity prices, these prices are mainly driven by supply and demand fundamentals. It is believed that natural gas prices may also be linked to the price of crude oil and/or petroleum products. In the United States Natural gas prices had historically followed oil prices, but recent studies have shown that in recent years Natural gas prices have decoupled from oil and are now trending somewhat with coal prices. This is an alternative hypothesis that could be tested by the project. The forward outlook of Natural gas prices will be assessed and inferences drawn.

3.1.1 The Influence of Weather on Natural Gas Prices

In order to investigate the impact of weather on natural gas prices, we will first create two groups: The group that incorporates prices in the winter & summer (Expected High Demand Period) and the group that incorporates prices in the spring and fall (and Expected Low Demand Period). The winter & summer group are tagged (1) in weather-status variable (WSTAT) while the fall & spring group is labeled (0) in weather-status column.



The scatter plot above (Gas price vs Total Gas consumption) shows that gas price has always fallen within the $1/MMbtu and $5/MMbtu. Higher gas prices > $5/MMbtu were accompanied with higher demand (higher total consumption) of Natural Gas (>2.5 Bscf). From the scatter plot, it could be observed that during spring and fall, the price of gas falls within the normal range of period of low gas price $1/MMbtu and $5/MMbtu, while the period of high price (>$5), are during the winter and the summer. It is also evident that the period of spring and fall (blue shaded circles) are consistent with the period of low gas consumption while the winter and summer period are consistent with the time of high natural gas consumption.

Now that we have established that there is an apparent difference between the two groups, we want to check whether it might be due to chance.

We split the data frame into two thereby creating two groups x1 and x2 comprising data for (spring & fall) and (winter & summer) respectively.

3.1.2 Test Statistic

We'll look at the gas price and total gas consumed. The effect size we'll consider is the difference in the means. Later we will look at a correlation between variables or a coefficient in a linear regression. The number that quantifies the size of the effect is called the "test statistic". The actual difference in the means is 26 cent per MMbtu.

The null hypothesis is that there is no difference between the groups. We can model that by forming a pooled sample that includes gas prices in all the weathers. In this case the result is about 17%, which means that even if there is no difference between the groups, it is plausible that we could see a sample difference as big as 26 cents.

We conclude that the apparent effect might be due to chance, so we are not confident that it would appear in the general population, or in another sample from the same population.

3.1.3 Difference in Gas Demand

We ran DiffMeansPermute again to see if there is a difference in Gas Demand between the different weather classes. In this case, after 1000 attempts, we never see a sample difference as big as the observed difference, so we conclude that the apparent effect is unlikely under the null hypothesis. Under normal circumstances, we can also make the inference that the apparent effect is unlikely to be caused by random sampling. In this case, the p-value is less than 1/1000 or less than 0.001 as the apparent effect is not impossible under the null hypothesis; just unlikely.

4.0 Anomaly and Outlier Detection

The goal of anomaly detection is to identify cases that are unusual within our seemingly homogeneous data. Outliers are cases that are unusual because they fall outside the distribution that is considered normal for the data. The distance from the center of a normal distribution indicates how typical a given point is with respect to the distribution of the data. Each case can be ranked according to the probability that it is either typical or atypical.

Anomaly detection is a form of classification and is implemented as one-class classification, because only one class is represented in the training data. An anomaly detection model predicts whether a data point is typical for a given distribution or not. An atypical data point can be either an outlier or an example of a previously unseen class. Normally, a classification model must be trained on data that includes both examples and counter-examples for each class so that the model can learn to distinguish between them.

It is often used in preprocessing to remove anomalous data from the dataset. In supervised learning, removing the anomalous data from the dataset often results in a statistically significant increase in accuracy.

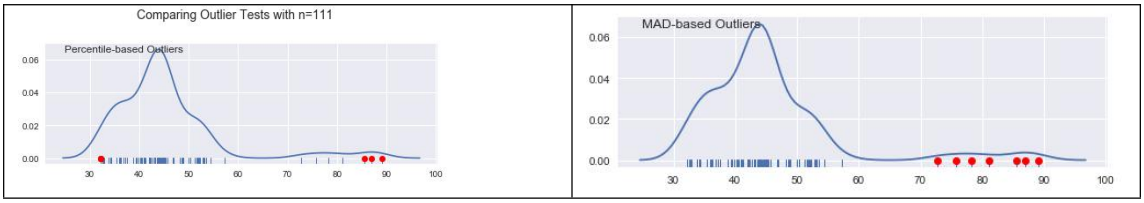
4.1.1 Outlier Detection

We will implement two types of outlier detection:

- Median-absolute-deviation (MAD) based outlier detection - which is a method that measures the distance of all points from the median in terms of median distance
- Percentile based outlier detection

The MAD-based classifier works correctly regardless of sample-size, while the percentile based classifier classifies more points the larger the sample size is, regardless of whether or not they are actually outliers. Therefore, in this work, MAD-based classifier is accepted as a more robust form of outlier detection.

Since these two methodologies deal with univariate data, we will look at the the following variables one at a time: AveCoalPrice, OilPrice, GrossGasProd, TotGasCons, and GasPrice.



The MAD-based outlier detection technique detects average coal price > 70 as outliers. These values are colored red.

4.1.2 Kernel Density Estimation

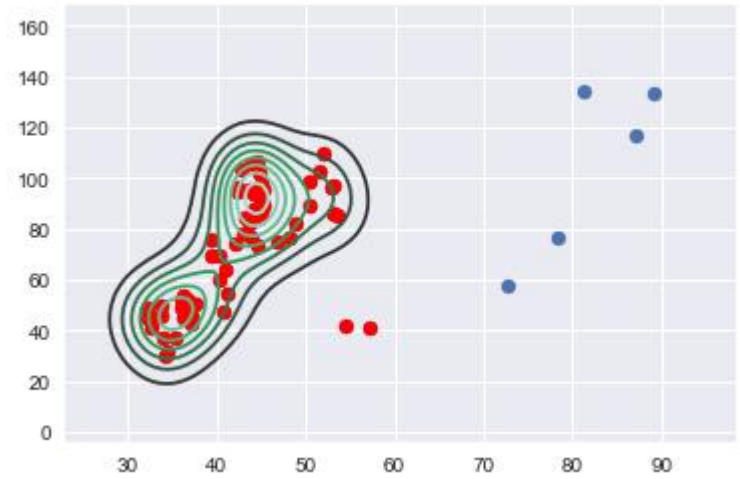
Another methodology used for novelty anomaly detection in this work is to fit and plot a bivariate kernel density estimate.Novelty Anomaly Detection
Firstly we will estimate a Gaussian distribution for each feature in the data. Recall that to define a probability distribution we need two things - mean and variance. To accomplish this we'll create a simple function that calculates the mean and variance for each feature in our data set. Now that we have our model parameters, we need to determine a probability threshold which indicates that an data point should be considered an anomaly. To do this, we need to use a set of labeled validation data and test the model's performance at identifying those anomalies given different threshold values.

Furthermore, we also need a way to calculate the probability that a data point belongs to a normal distribution given some set of parameters. Fortunately SciPy has this built-in.
We just calculated the probability that each of the first 50 instances of our data set's first dimension belong to the distribution that we defined earlier by calculating the mean and variance for that dimension. Essentially it's computing how far each instance is from the mean and how that compares to the "typical" distance from the mean for this data.

Let's compute and save the probability density of each of the values in our data set given the Gaussian model parameters we calculated above.

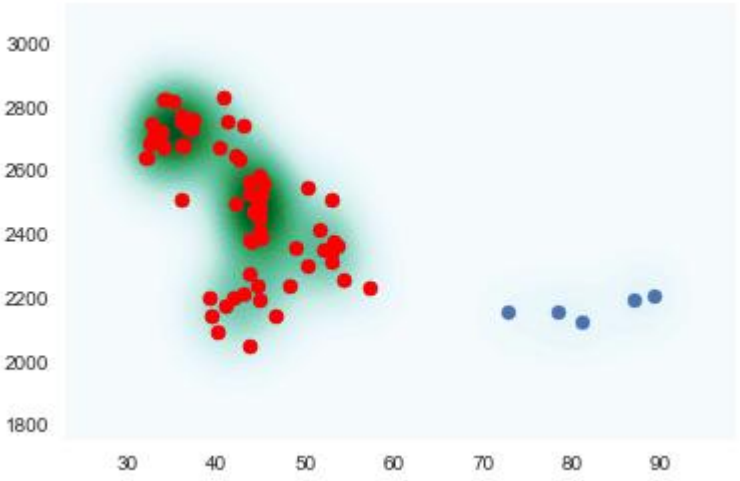
4.1.2.1 kde Plot of Average Coal Price vs Oil Price

A kde plot shows the points enclosed in the cluster, while the points outside the enclosure could be regarded as anomalies.



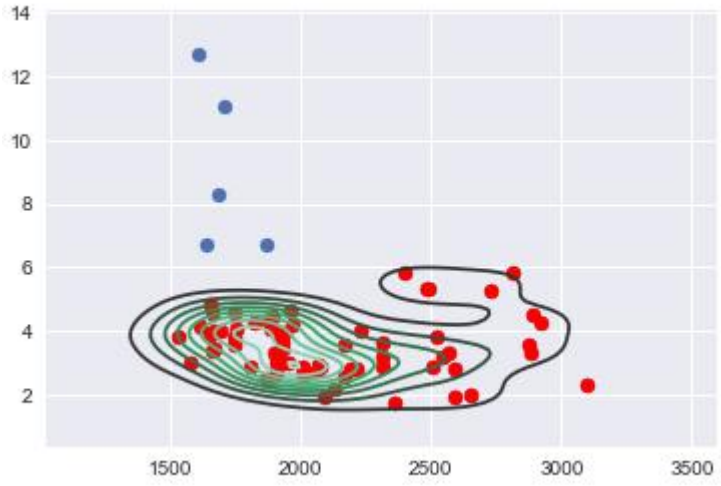
4.1.2.2 kde Plot of Average Coal Price vs Gross Gas Production

A kde plot shows the points enclosed in the cluster, while the points outside the enclosure could be regarded as anomalies.



4.1.2.3 kde Plot of Total Gas Consumption vs Gas Price

A kde plot shows that points outside the enclosure which may be considered as anomalies are driven primarily by the gas price.



4.1.2 Anomaly Detection - Conclusion

For further analysis, it would be required to remove gas prices higher than 7 and equivalent data for oil price higher than 70 in order the improve the accuracy of the model.

4.2.1 Variance Inflation Factor

The variance inflation factor (VIF) quantifies the severity of multicollinearity among variables in an ordinary least squares regression analysis. It provides an index that measures how much the variance (the square of the estimate's standard deviation) of an estimated regression coefficient is increased because of collinearity.

4.2.1.1 Collinearity

Colinearity is the state where two variables are highly correlated and contain similar information about the variance within a given dataset. Multicolinearity on the other hand is more troublesome to detect because it emerges when three or more variables, which are highly correlated, are included within a model. To make matters worst multicolinearity can emerge even when isolated pairs of variables are not colinear.

4.2.1.2 Steps for Implementing VIF

- Run a multiple regression.
- Calculate the VIF factors.
- Inspect the factors for each predictor variable, if the VIF is between 5-10, multicolinearity is likely present and you should consider dropping the variable.

As expected, the Gross gas production has been on the increase, therefore has a high variance inflation factor with days. Both variables are likely to "explain" the same variance within this dataset. We would need to discard one of these variables before moving on to model building or risk building a model with high multicolinearity.

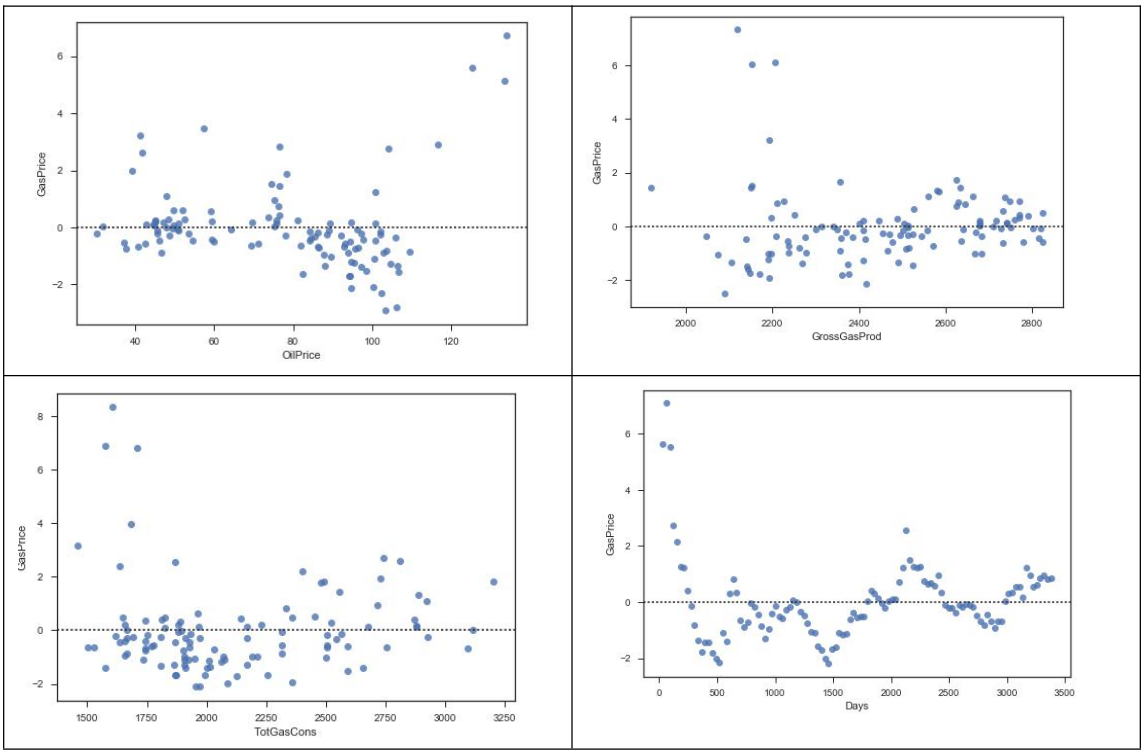
5.0 Linear Regression

5.1.1 Residuals

For linear relationships, the ideal plot of residuals with each of the predictor should be a random scatter because we assume that the residuals are uncorrelated with the predictor variables. Any noticeable pattern in such plots indicates violation of linear relationship assumption.

Below are plots of the residuals with each of the predictor variables in the given data set. As we can see the plots are random scatter plots. The plots are indicating linearity.

Therefore, linear regression analysis will be applied to the data.



5.1.2 Linear Regression

We will use machine learning techniques to analyze our natural gas data set. Our first insight will be through the simplest model - linear regression. The goal in regression problems is to predict the value of a continuous response variable.

We will first examine linear regression, which models the relationship between a response variable and each of the explanatory variable. Next, we will consider polynomial regression and regularization methods if necessary.

In this section, we will investigate the goodness of fit if we are presented with data different from the data set we have used to construct this model. This will help us in prediction. One of the solutions is to split our data set into two (one

group left out (for testing) and while the rest us used to train the model. Cross validation will be carried out after this split and analysis.

Let us start by splitting the data into random train and test subsets using the function train_test_split in sklearn.cross_validation. We then built a linear regression model using our new training data set. We will: (1) Fit a linear regression model to the training set (2) Predict the output on the test set. mean square error for X_test is 0.6306954835175395.

Let us compare the residuals in a residual plot

The accuracy score is 0.269062493854

5.1.1 Cross Validation

There are some inherent dangers in the train/test split which we carried out above: For instance, what if the split we made isn't random? What if one subset of our data has only data from a certain year, period (winter, summer, Autumn or spring) or the data is from certain features. This will result in over fitting, even though we're trying to avoid it! This is where cross validation comes in.

In this section, we apply the train/test split to more subsets. Meaning, we split our data into k subsets, and train on k-1 one of those subset. What we do is to hold the last subset for test. We're able to do it for each of the subsets. We will use K-Folds Cross Validation and the Leave One Out Cross Validation (LOOCV)

5.1.1.1 K-Folds Cross Validation

In K-Folds Cross Validation we split our data into k different subsets (or folds). We use k-1 subsets to train our data and leave the last subset (or the last fold) as test data. We then average the model against each of the folds and then finalize our model. After that we test it against the test set.

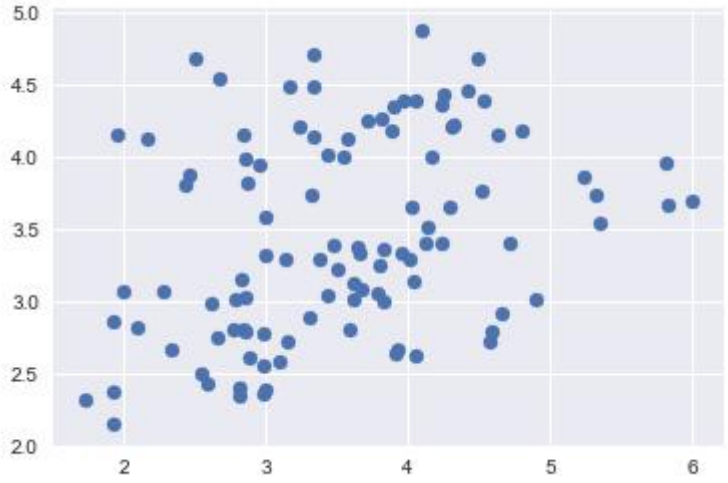
As we can see, the function splits the original data into different subsets of the data.

5.1.1.2 Leave One Out Cross Validation (LOOCV)

In this type of cross validation, the number of folds (subsets) equals to the number of observations we have in the dataset. We then average ALL of these folds and build our model with the average. We then test the model against the last fold. Because we would get a big number of training sets (equals to the number of samples), this method is very computationally expensive but we are using it because our dataset is small datasets.

The more folds we have, we will be reducing the error due the bias but increasing the error due to variance; the computational price would go up too, obviously—the more folds we have, the longer it would take to compute it and we would need more memory. With a lower number of folds, we're reducing the error due to variance, but the error due to bias would be bigger. It would also computationally cheaper. Therefore, in big datasets, k=3 is usually advised. In smaller datasets it's best to use LOOCV. We will use the cross_val_predict function to return the predicted values for each data point when it's in the testing slice.

6 fold cross validation improved the score of the original model—from 0.0528911 to 0.16772978. Plotting the new predictions, after performing cross validation, we have: Cross-Predicted Accuracy: -0.0688343792343

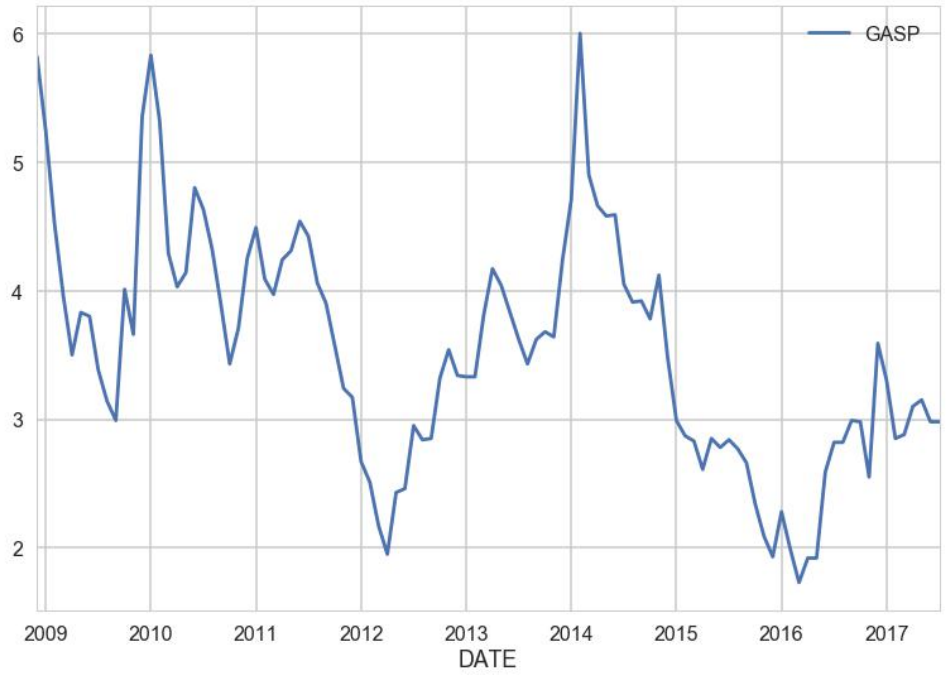


We can see our scatter plot is very different from the original plot (its now a near perfect correlation). It is six times as many points as the original plot because we used cv=6. Finally, let's check the R^2 score of the model (R^2 is a "number that indicates the proportion of the variance in the dependent variable that is predictable from the independent variable(s)". Basically, how accurate is our model):

6.0 Classification: Natural Gas Data Set

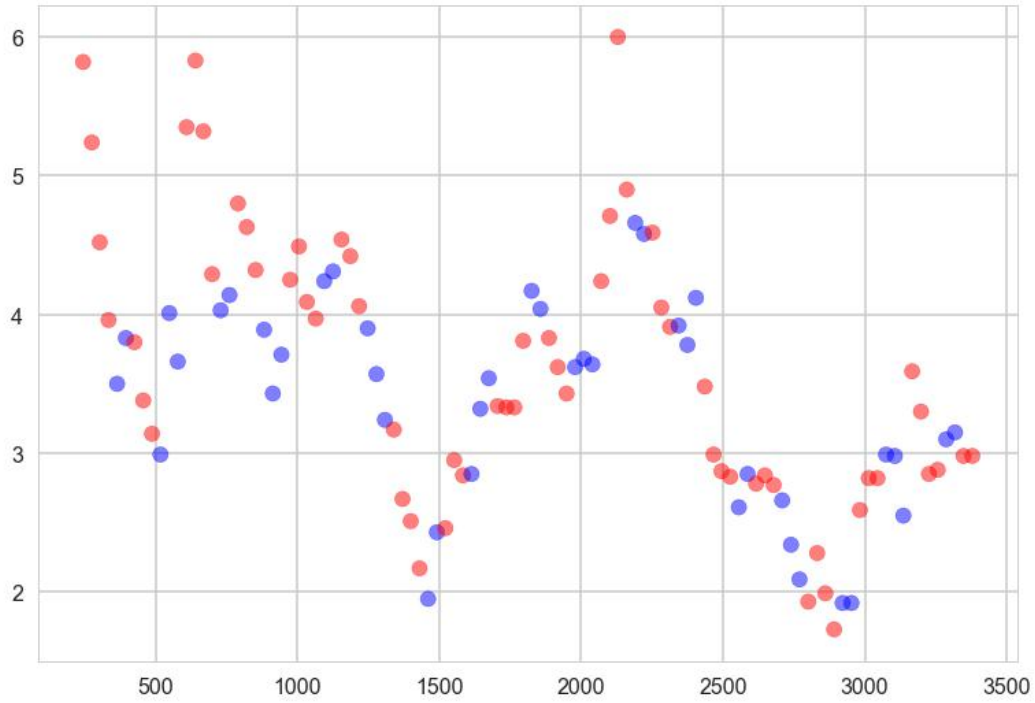
We'll classify our data using the dataset of natural gas prices. Classification tries to predict, which of a small set of classes, an observation belongs to. Mathematically, the aim is to find yy, a label based on knowing a feature vector xx . In our case we will consider predicting gas price by identifying the weather.

To have a machine do this well, we would typically feed the machine a bunch of gas prices which have been labelled "WSTAT = 0" or "WSTAT =1" (the training set), and have it learn the gas price in the dataset from the labels and the features used to determine gas prices. Then, given a new weather status, the trained algorithm returns us the gas price that is expected for such a weather Status. "0" is the used to denote when the weather is either SPRING or FALL while "1" denotes that period when the weather is WINTER or SUMMER.
We now generate a time series plot for gas price.



We also check whether gas price varies according to the weather.

```
plt.scatter(dflog.DAYS, dflog.GASP, c=[cm_bright.colors[i] for i in dflog.WSTAT == 0], alpha=0.5);
```



Observe that the periods of gas price peaks correspond to winter and summer months (in red). Let us further investigate the validity of the classification by plotting gas price as a function of other variables

6.1.1 Training and Test Datasets

When fitting models, we would like to ensure two things: (1) We have found the best model (in terms of model parameters). (2) The model is highly likely to generalize i.e. perform well on unseen data.

First, we try a basic Logistic Regression: (1) Split the data into a training and test (hold-out) set (2) Train on the training set, and test for accuracy on the testing set

Let us start with the relationship that has the most explicit classification: That is classification of Gas price and Gas Consumption with respect to weather Status.

In this case, the accuracy score is 0.58.

6.1.2 Tuning the Model

6.1.2.1 Logistic Regression

The model has some hyperparameters we can tune for hopefully better performance. For tuning the parameters of our model, we will use a mix of cross-validation and grid search. In Logistic Regression, the most important parameter to tune is the regularization parameter C. We will note that the regularization parameter is not always part of the logistic regression model.

The regularization parameter is used to control for unlikely high regression coefficients, and in other cases can be used when data is sparse, as a method of feature selection.

We will now implement some code to perform model tuning and selecting the regularization parameter C.

We will use the following cv_score function to perform K-fold cross-validation and apply a scoring function to each test fold. In this incarnation we use accuracy score as the default scoring function.

In order to find a good model using all the key independent variables, we will use a given list of possible values of C. For each C, we will create a logistic regression model with that value of C. We will find the average score for this model using the cv_score function only on the training set (Xlr, ylr) and we will then pick the C with the highest average score

The goal is to find the best model parameters based only on the training set, without showing the model test set at all.

Here we see that the maximum score is 0.70 while the best C is 1. We will now evaluate the accuracy score of our prediction.

6.1.2.1 Grid Search CV

Using grid search CV, we observe that the best C is 0.0001 and the maximum score is 0.718. Now we generate the accuracy score of our prediction. In these cases, we have used a squared error loss function along with Empirical Risk Minimization (ERM) to carry out regression. The idea there was to calculate this risk on the training set and minimize it. Then the hope was that on the population, or any testing set representative of it, the out-of-sample risk was similar in size to the in-sample training risk, and thus small. The question then is what might be an appropriate risk for classification? One immediately comes to mind: the fraction of misclassified samples. Since the accuray score for our model is 77%, it suggests that the risk that a portion of the sample is misclassified is 27%

7.0 Principal Component Analysis

7.1.1 The curse of dimensionality: Feature engineering

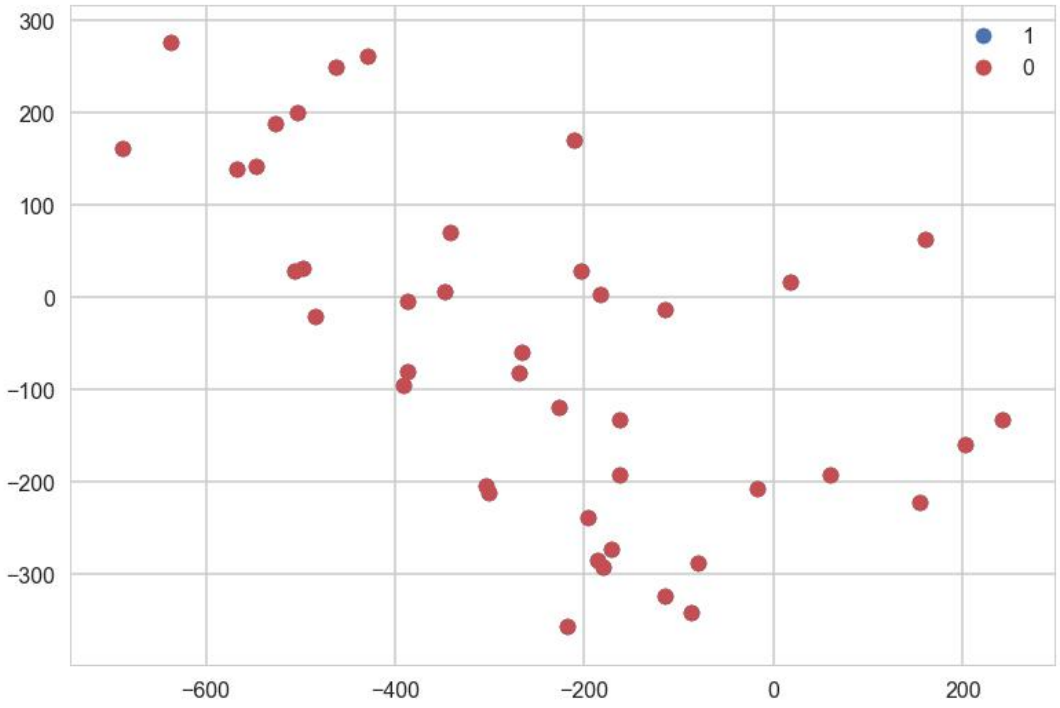
Let us recall that having too many features can lead to overfitting. We will engage in some a-priori feature selection that will reduce the dimensionality of the problem. The idea we'll use here is called Principal Components Analysis, or PCA. PCA is an unsupervized learning technique. The basic idea behind PCA is to rotate the co-ordinate axes of the feature space. We first find the direction in which the data varies the most. We set up one co-ordinate axes along this direction, which is called the first principal component. We then look for a perpendicular direction in which the data varies the second most. This is the second principal component. There are as many principal components as the feature dimension: all we have done is a rotation.

How does this then achieve feature selection? We decide on a threshold of variation; once the variation in a particular direction falls below a certain number, we get rid of all the co-ordinate axes after that principal component. For example, if the variation falls below 10% after the third axes, and we decide that 10% is an acceptable cutoff, we remove all dimensions from the fourth dimension onwards. In other words, we took our higher dimensional problem and projected it onto a 3 dimensional subspace.

Here we'll continue to focus on PCA. We'll reduce our dimensionality from 5 to 3. We choose 3 as a large apriori number: we dont know if the variation in the data will have gone below a reasonable threshold by then.

The explained variance ratio (0.999925003031) tells us how much of the variation in the features is explained by these 3 features. When we sum it up over the features, we see that more than 99% is explained: good enough to go down to a 3 dimensional space from a 5 dimensional one!

We can see the individual variances as we increase the dimensionality: The first dimension accounts for 81% of the variation, the second 19%, and it goes steadily down from there. Let us create a dataframe with these 3 features labeled pc1,pc2,pc3 and the labels of the sample:



plt.scatter(df[mask]['pc1'], df[mask]['pc2'], c=color, label=label)

7.1.2 Classifying In A Reduced Feature Space with kNN

Implicit in the notion of classification, is the idea that samples close to each other in feature-space share a label. kNN is a very simple algorithm to diretly use this idea to do classification. The basic notion is this: if a lot of samples in some area of the feature space belong to one class as compared to the other, we'll label that part of the feature space as "belonging" to that class. This

process will then classify the feature space into class-based regions. Then, given the point in feature space, we find which region its in and thus its class.

The way kNN does this is to ask for the k nearest neighbors in the training set of the new sample. To answer this question, we would define a distance in the feature space (Note that this distance is different from the error or risk measures we have seen earlier). This distance is typically defined as the Euclidean distance, the sum of the square of the difference of each feature value between any two samples.

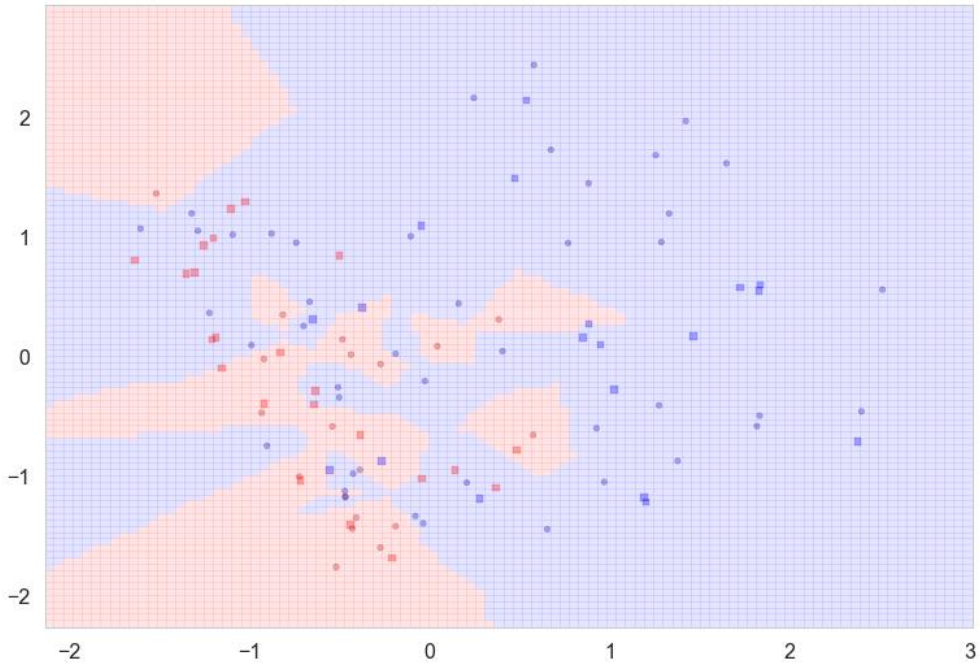
Once we have a distance measure, we can sort the distances from the current sample. Then we choose the k closest ones in the training set, where k is an odd number (to break ties) like 1,3,5,...19,. We now see how many of these k "nearest neighbors" belong to one class or the other, and choose the majority class amongst those neighbors as our sample's class. The training process thus simply consists of memorizing the data, perhaps using a database to aid in the fast lookup of the k nearest training set neighbors of any point in feature space.

We will use sklearn's simple api to write the classifier:

Lets see what happens when we choose $k=1$. On the training set, the 1NN classifier memorizes the training data. It will predict perfectly on the training set, and wont do too badly on the test set, especially deep in the regions of feature space where one or the other class dominates. This is because even one neighbor might be enough in those regions. However, the same classifier will do badly near the clasification boundaries on the test set, because you will need more than one neighbor to decide with any certainty of the class. The result of this is, as you might expect, the regions of feature space classfied one way or the other are quite jagged and mottled. Since we are choosing just one neighbor, we fit to the noise in the region rather than the trend. We are overfitting.

classify(Xs,ys,1)

- Accuracy on training data: 1.00
- Accuracy on test data: 0.57

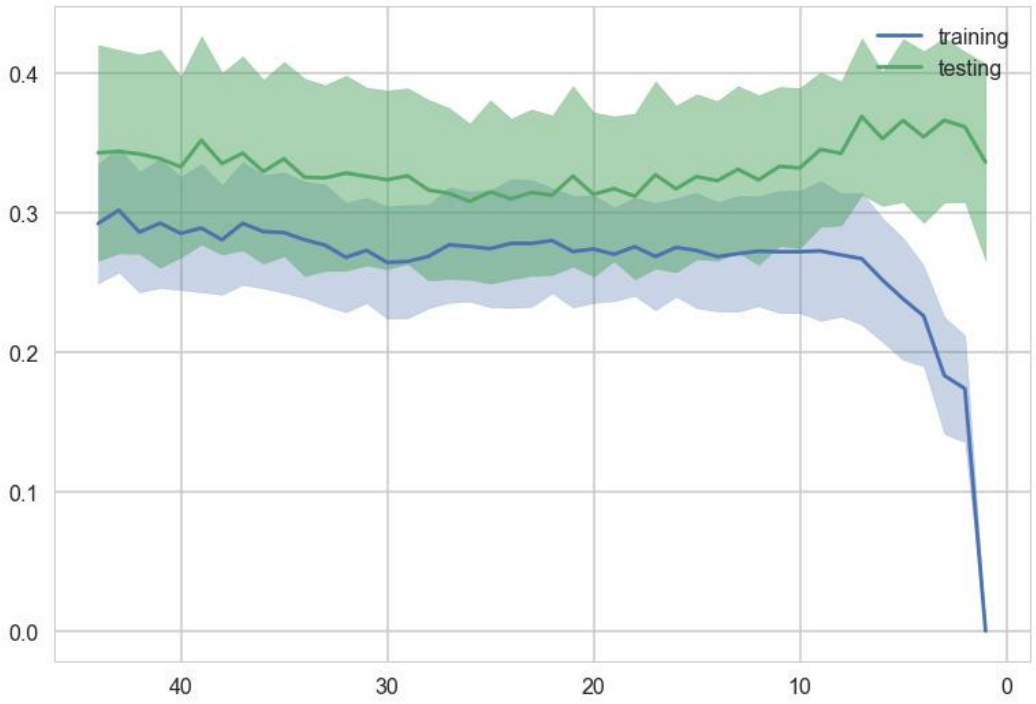


If we choose a large number for k, such as 3, we are wandering too far from our original sample,and thus we average over a large amount of the feature space. This leads to a very biased classification, depending on where our sample is, but extending far out from there. Our classification may even cover the entire feature space, then giving us the majority class. In terms of probabilities, such an underfit case gives us the base rate classifier. Imagine $k=N$. Then the probability is just the fraction of training set examples in a given class. Say this number for the blue class is 0.4 (that is, when we have uneven class memberships in the training set). Now, on any random test set, if we use the classifier which says "classify all as red", we will be correct, on average, 60% of the time if the test set and training sets are representative of the population of samples. Any classifier we create must do a better job than this!

classify(Xs,ys,3)#run this a few times

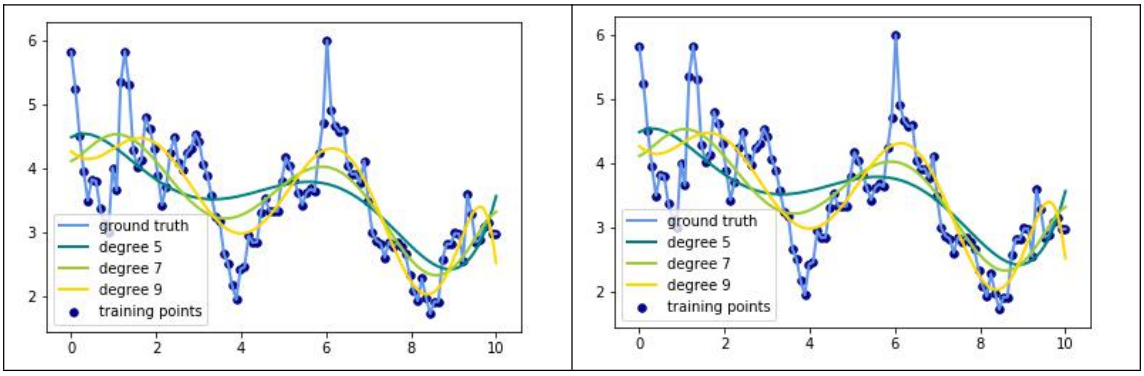
- Accuracy on training data: 0.81
- Accuracy on test data: 0.60

7.1.3 Error against complexity (k), and cross-validation



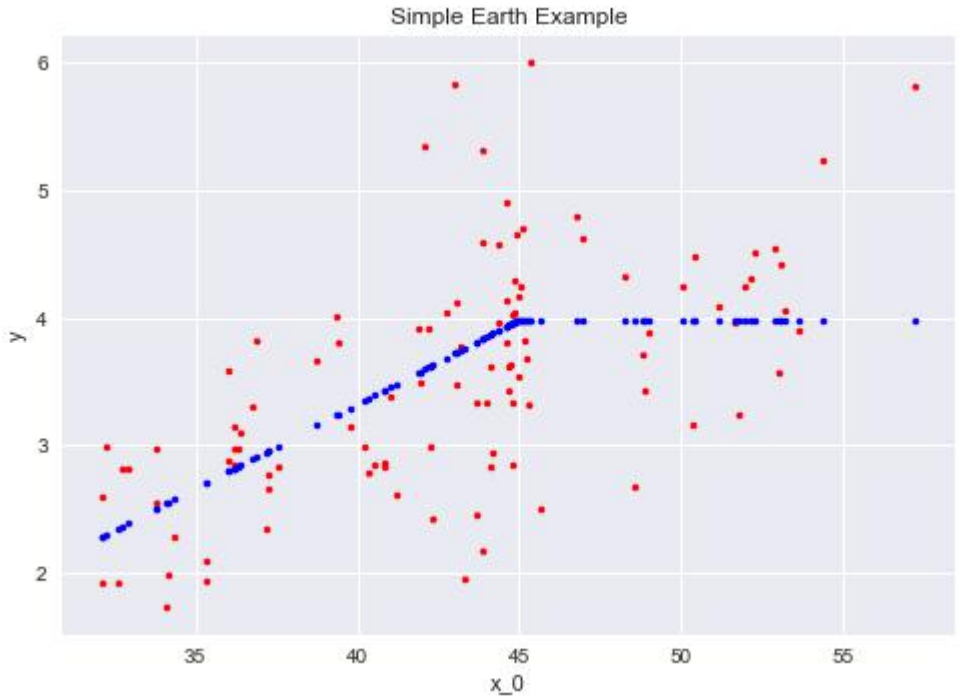
Once again, as before, we plot the test error and training errors against the number of neighbors k . Here k serves as a complexity parameter, with small k being more "wiggly" in the classification of neighborhoods and large k oversmoothing the classification. Notice that we plot k reversed on the x-axis so as to go from lower complexity to higher complexity. As expected, the training error drops with complexity, but the test error starts going back up.

8.0 Polynomial Regression



8.1.1 Multivariate Adaptive Spline

We used the Multivariate Adaptive Spline to carry out polynomial regression. In this case, gas price is regressed against average coal price by fitting an earth model:



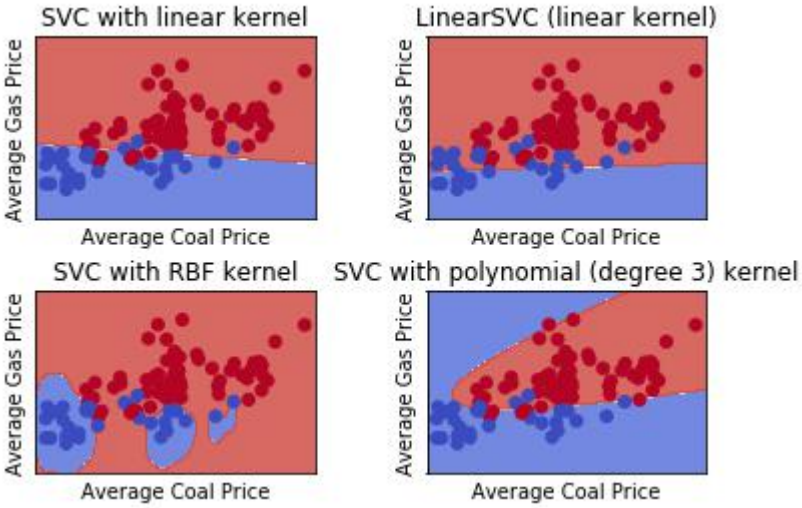
9.0 Support Vector Machine

SVM is a supervised machine learning algorithm which we are using for our regression problems. It uses a technique called the kernel trick to transform our data and then based on these transformations it would find an optimal boundary between the possible outputs. SVM could also be used for classification and outliers detection.

9.1.1 Classification

In this subsection, we will plot different SVM classifiers in the natural gas dataset. There are three common classes capable of performing multi-class classification on a dataset: These are SVC, NuSVC and LinearSVC. SVC and NuSVC are similar methods, but accept slightly different sets of parameters and have different mathematical formulations (see section Mathematical formulation). On the other hand, LinearSVC is another implementation of Support Vector Classification for the case of a linear kernel.

Features - Average Coal Price and Gas Price, Target: Weather Status Let us consider the features: average coal price (AveCoalPrice) and gas price (GasPrice) and classify them with respect to the Gas Price status: GPSAT

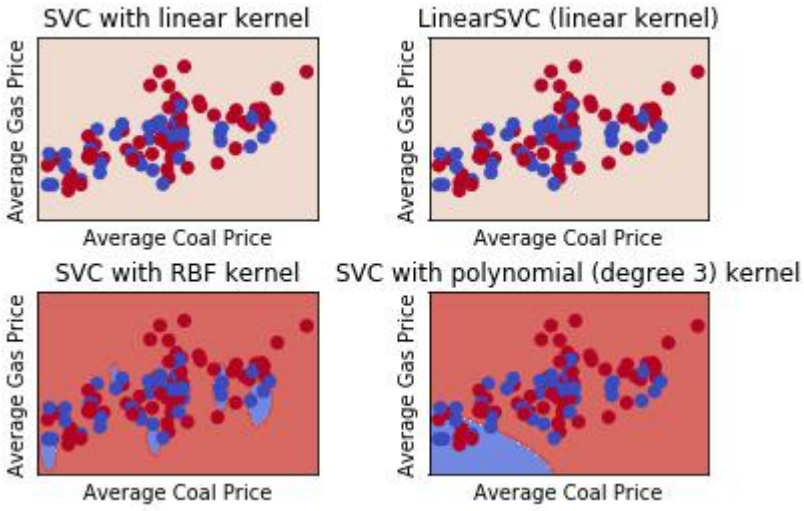


We will now consider one feature at a time and gas price (GasPrice). We will classify them with respect to the weather status: WSTAT. If the feature falls within the Winter and Summer it is assigned the value 1, and zero if it falls within Spring and Fall. Using two-dim dataset allows us to avoid this slicing.

Features - Average Coal Price and Gas Price, Target: Weather Status
Let us consider the features: average coal price (AveCoalPrice) and gas price (GasPrice) and classify them with respect to the weather status: WSTAT. We will now: (1) assign step size in the mesh (h) (2) create an instance of SVM and fit our data. (3) assign an SVM regularization parameter (C) We would not scale our data since we want to plot the support vectors

Let us fit

- LinearSVC()
- SVC(kernel='linear').
- Polynomial
- Gaussian RBF



The linear models LinearSVC() and SVC(kernel='linear') yield different decision boundaries. This can be due to any of the following reasons:

LinearSVC minimizes the squared hinge loss while SVC minimizes the regular hinge loss.
LinearSVC uses the One-vs-All (also known as One-vs-Rest) multiclass reduction while SVC uses the One-vs-One multiclass reduction.

Both linear models have linear decision boundaries (intersecting hyperplanes) while the non-linear kernel models (polynomial or Gaussian RBF) have more flexible non-linear decision boundaries with shapes that depend on the kind of kernel and its parameters.

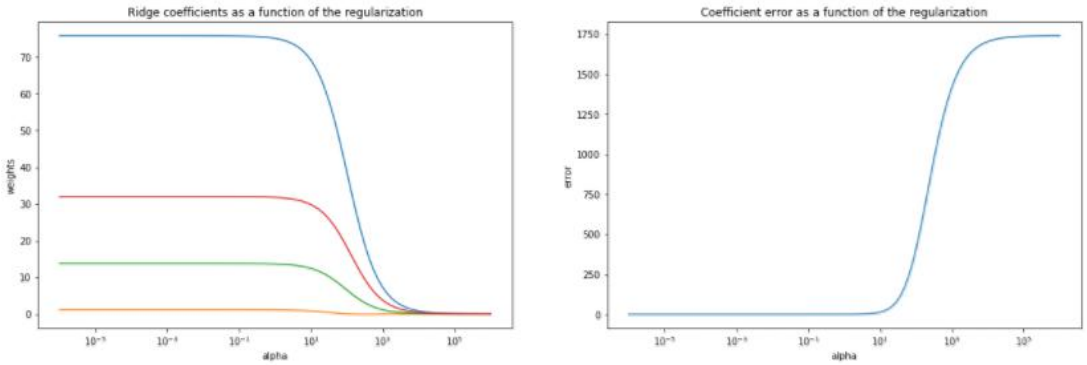
10.0 Ridge Regression
Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value.

We generated our estimator instance clf (classifier) and use it to fit a ridge model on the full data.

10.1.1 Plotting Ridge coefficients as a function of the L2 regularization
Each color in the left plot represents one different dimension of the coefficient vector, and this is displayed as a function of the regularization parameter. The right plot shows how exact the solution is. This shows how a well defined solution is found by Ridge regression and how regularization affects the coefficients and their values. The plot on the right shows how the difference of the coefficients from the estimator changes as a function of regularization.

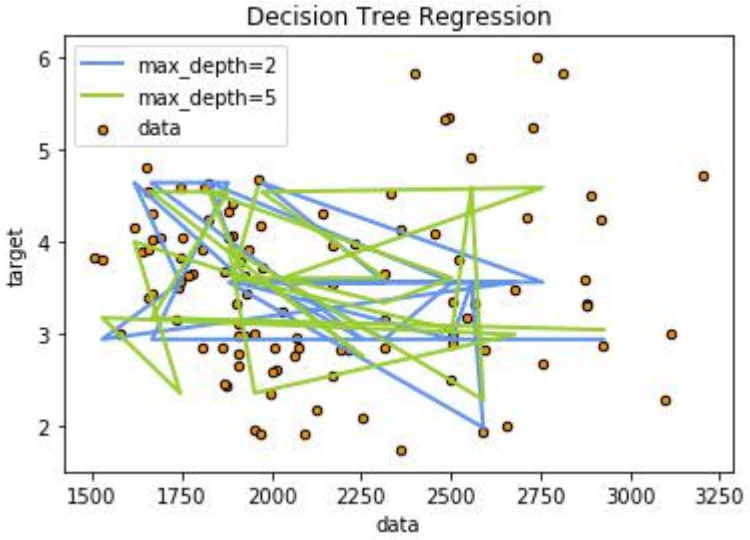
The dependent variable Y is set as a function of the input features: y = X*w + c.

The coefficient vector w is randomly sampled from a normal distribution, whereas the bias term c is set to a constant. As alpha tends toward zero the coefficients found by Ridge regression stabilize towards the randomly sampled vector w. For big alpha (strong regularization) the coefficients are smaller (eventually converging at 0) leading to a simpler and biased solution. These dependencies can be observed on the left plot. The right plot shows the mean squared error between the coefficients found by the model and the chosen vector w. Less regularized models retrieve the exact coefficients (error is equal to 0), stronger regularized models increase the error.



11.0 Decision Trees

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Using natural gas data set, decision trees are generated to learn from data to approximate a gas prices with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.



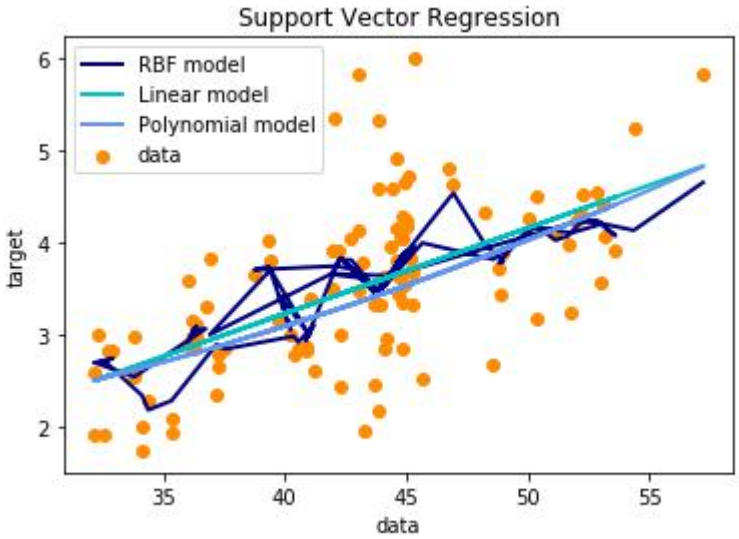
Using Decision Trees, the mean score is 0.67669515669515679

11.0 Support Vector Regression

The method of Support Vector Classification can be extended to solve regression problems. This method is called Support Vector Regression. Just like in support vector classification, the model produced by Support Vector Regression depends only on a subset of the training data, because the cost function for building the model ignores any training data close to the model prediction.

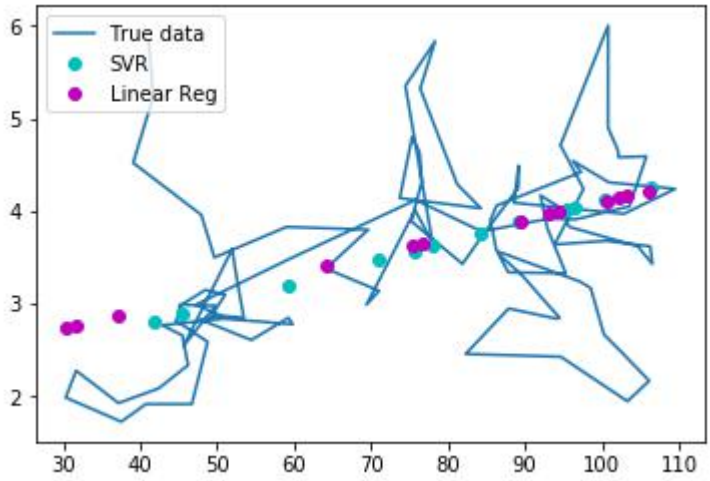
We will generate 1D regression using linear, polynomial and RBF kernels. Let us firstly import the required libraries and upload the natural gas dataset.

Regressing Gas Price against Average Coal Price¶



Estimating score for the natural gas data using SVR, the accuracy of the model is 0.63634401052718848

Oil Price vs Gas Price



12.0 Boosting Accuracy of the Machine Learning Model

In this section, we used different machine learning models to attempt to boost the accuracy score. The result is shown below:

Accuracy Score

Linear Regression	Decision Tree	Extremely Randomnized Tree	Random Forest	Extra Tree
0.47	0.68	0.69	0.78	0.73

Accuracy Score (Ada Boost = 0.70)
GradientBoostingClassifier (Score) - 92%

GradientBoostingRegression - MSE: 0.12

13.0 Conclusion and Recommendation

- Coal price remains the key indicator of increase in gas price. Therefore, it is a variable that investors should watch out for.
- Whereas Oil price was seen as the major determinant of natural gas prices, our study shows that its impact in the past decade is marginal.
- Gas demand is a key factor that determines US gas price. Gas demand is primarily driven by seasonal variation or occurrence of natural disaster.

- Seasonal weather, including tropical storms and hurricanes, can have a great influence on natural gas production. The seasonal effect of weather on gas price cannot be effectively analyse using logistic regression and classification. Time series Analysis is therefore recommended in order to explore the effect of seasonality.
- There are other factors that contribute to gas price variation that have not been captured in this project. Such variables as economic growth, natural gas storage and availability of gas pipelines have effect on gas prices. It is recommended that future studies include these variables.