# Lasso Linear Regression and SVM Classification

*Abstract* - **The first objective was to compare the results of Lasso and Ordinary Least Squares (OLS) when predicting commodity values. Through this, the commodity that best reflects the broad market and the minimum number of commodities to track can be determined for amazing.inc. The second objective was to use an SVM with an unscaled Gaussian kernel to classify non-linearly separable data. Lasso and OLS were implemented on a dataset provided by the U.S Federal Reserve Bank of St. Louis Economic Data (FRED) website. Both methods were used to predict the value of the 'Lead' commodity and had their predicted values graphed compared to the actual values. Before applying the SVM with a Gaussian kernel, the data set needed to be dimensionally reduced to a 2 dimensional set. Once reduced the data set was run through the SVM and the values were classified. By observing Figure 3, it is apparent that Lasso was able to better predict the 'Lead' commodity values compared to OLS due to OLS's susceptibility to outliers. As well Figure 2 shows that the commodity that best represents the broad market is B2: 'Copper' as it is non zero at the highest lambda value. Figure 2 also shows that the minimum number of commodities to track is 6 (B1: 'Coal', B4: 'Fish meal', B6: 'Hides', B12: 'Tin', B13: 'Uranium', and B15: 'Zinc') as these variables have a non-zero value at the minimum MSE lambda. By observing Figure 4, it can be seen that applying an SVM with an unscaled Gaussian kernel was able to achieve an accuracy of 100% when classifying the non-linearly separable data. Overall Lasso proved to be a more effective prediction method compared to OLS as it generated a more general solution that was less affected by outliers at a lower computational cost. As well, SVM with an unscaled Gaussian kernel was an effective classification tool when dealing with non-linearly separable data.**

## INTRODUCTION

The first objective was to compare Lasso Linear Regression and Ordinary Least Squares (OLS) when predicting commodity values and determine the commodity that best reflects the broad market along with the minimum number of commodities that amazing.inc should track. The second objective was to apply Gaussian SVM in order to classify non-linearly separable data.

The data set analyzed was from the U.S Federal Reserve Bank of St. Louis Economic Data (FRED) website and included global prices for frequently traded commodities. Missing values were handled using 'Mean substitution'. The mean was applied over a 5 quarter window in order to avoid inconsistent bias, which can be an issue when using 'Mean substitution' [1]. As the values are not strictly random, the window allows for the calculated mean value to more greatly reflect the current behaviour of the commodity value. Lasso will be applied to this filled dataset in order to determine the commodity that best reflects the broad market and find the minimum number of commodities to track. Lasso stands for Least Absolute Shrinkage and Selection Operator and is a linear least squares problem. Its main purpose is feature selection and regularization of data models. The method regularizes model parameters by shrinking the regression coefficients, reducing some of them to zero. After this shrinkage, feature selection occurs where every non-zero value is selected to be used in the model. Lasoo is able to offer high prediction accuracy as the shrinkage of coefficients reduces variance and minimizes bias.

SVM is a linear model used for classification and regression problems. The basic principle is that the SVM generates a hyperplane within the data and uses this hyperplane to classify the data points. There are instances where data may not be linearly separable, meaning a linear hyperplane cannot be generated to classify the data. A method for handling this data is to map each data vector to a higher dimensional vector space. The new mapped vector may be linearly separated in the higher vector space. This can be done by applying an SVM with an unscaled Gaussian kernel to classify the data.

The effectiveness of lasso compared to OLS will be evaluated by comparing how accurately each method predicts the value of the 'Lead' commodity. This will be tested by viewing the resulting predicted values to the known values and comparing which method gives the best modeling. The effectiveness of

SVM with a Gaussian kernel function will be evaluated by calculating the accuracy of the classification on a small data set as a percentage value.

## METHODS

Lasso is a linear least squares problem that is constrained by the L1 norm of the weight vector. The lasso is simply the least squares objective function with a threshold constraint θ written as $\overline{w}^* = argmin[X\overline{w} - \overline{y}]^T[X\overline{w} - \overline{y}]$ such that $||\overline{w}^*|| <= \theta$. The change to using the L1 norm has a substantial effect on the minimization. If the ordinary least squares (OLS) solution is not feasible for the lasso then the constrained least squares (CLS) produces a weight vector that is on the boundary of the level set described by the constraint. Geometrically the level set of the constraint in CLS is a hypersphere of radius $\sqrt{\theta}$ around the origin whereas, the level set of the constraint in the lasso is a hypercube of width θ that is centered at the origin. Lasso works by setting certain weight values to exactly zero and ensuring that the sum of the absolute values of the other weight values meet the imposed constraint. When comparing CLS and lasso it is apparent that there are numerical differences between the CLS optimal weight vector and the lasso optimal weight vector. Computation of the lasso is not straightforward, considering the 2D case, the constraint can be written as:

$$w1 + w2 <= \theta$$
$$-w1 + w2 <= \theta$$
$$w1 - w2 <= \theta$$
$$-w1 - w2 <= \theta$$

By writing out the constraint as a set of linear inequalities, it can be seen that $2^n$ inequalities are needed to constrain the weight vector $\overline{w} \in R^n$. Current methods use solutions that are linear in the number of dimensions n of the data space. Lasso can be effectively computed using the inequalities sequentially, which is a form of coordinate descent. Current solutions use an interior point method or an alternating direction method.

In order to classify the 5 dimensional data using an SVM with an unscaled Gaussian kernel the data needed to be dimensionally reduced. This reduction will change the data from 5 dimensional to 2 dimensional by using singular value decomposition. This was done by extracting the z-scores of the data and running these scores through singular value decomposition. The data from the right singular vector was extracted and multiplied by the vector containing the z-scores. After transposing, this produced the x-matrix that represents the dimensionally reduced data. The dimensionally reduced data could now be run through the SVM using an unscaled Gaussian kernel. The principal method for managing non-linearly separable data is to map each data vector to a higher dimensional vector space. In many instances, data vectors that seem to be non linearly separable can be mapped to another vector space where they are, this is known as embedding. If the embedding function has certain properties, it is possible to perform the necessary computations for the SVM without actually embedding the data vector $\overline{x}_i$ in a higher dimensional space. Looking at the primal Lagrange formula, $L(\overline{w}, b, \overline{\alpha}) = \frac{1}{2}\overline{w}^T\overline{w} + \overline{\alpha}^T[\overline{1} - YX\overline{w} - b\overline{y}]$, the objective term, $\frac{1}{2}\overline{w}^T\overline{w}$, would have the vector $\overline{w}$ replaced by the higher dimensional vector $\widehat{w}$. This makes the embedded objective function equal to $\frac{1}{2}\widehat{w}^T\widehat{w}$. The inequality constraints, which are managed by using the KKT conditions to introduce the Lagrange multipliers would have the weight vector $\overline{w}$ replaced by the higher dimensional vector $\widehat{w}$ and each data vector replaced by the higher dimensional vector, which produce a higher dimensional design

matrix $\widehat{X}$. This would generate the revised constraint term as $\bar{\alpha}^T[\bar{1} - Y\widehat{X}^T\widehat{w} - b\bar{y}]$. From these equations the dual formulation can be derived with an objective function of $-\frac{1}{2}\bar{\alpha}^T Y\widehat{X}^T\widehat{X}Y\bar{\alpha}$. The insight for the 'kernel trick' in training an SVM with a design matrix X, is that embedding does not need to be computed. In optimizing the objective function, dot products of the embedded vectors are needed. It is possible to compute the dot products without computing the embedding for certain useful embeddings. We can see that the matrix $XX^T$ is symmetric and, if the data vectors are linearly independent, it is also positive semidefinite. The entry (i, j) of the symmetric matrix is $\bar{x}_i^T\bar{x}_j$. If we embed the data vector $\bar{x}_i$ in a higher dimensional space as $\widehat{x}_i$, and embed $\bar{x}_j$ as $\widehat{x}_j$, the dot products of the equation would be $\widehat{x}_i^T\widehat{x}_j = [\bar{\phi}(\bar{x}_i)]^T\bar{\phi}(\bar{x}_j)$. A kernel function is a function that is symmetric and positive semidefinite. For an appropriate kernel function κ(. , .) that is defined for the vector space $R^n$, the mapping can be avoided and the dot product can be computed directly. For any design matrix X that is full rank, and any kernel function κ(. , .) defined on the vectors that are columns of X, the Gram matrix is the matrix K that has entries $K_{ij} = \kappa(\bar{x}_i, \bar{x}_j)$. With all this in mind the "kernel trick" can now be described. For any appropriately defined kernel function κ, the objective term is replaced with the Gram matrix of X such that the objective is equal to $\frac{1}{2}\bar{\alpha}^T YKY\bar{\alpha}$. This equation can be easily incorporated into many algorithms and methods.

For the purposes of classifying the small data set, an unscaled Gaussian kernel will be used. In a Gaussian kernel, the square of the norm of the distance between vectors can be used in the unscaled normal distribution function $e^{-x^2/2}$. This is usually presented with variance $\sigma^2$ with a kernel function $\kappa(\bar{u}, \bar{v}) = exp\left(\frac{-||\bar{u}-\bar{v}||^2}{2\sigma^2}\right)$. This variance of $\sigma^2$ is an important hyper parameter in the SVM. A small variance causes the Gram matrix to approach the identity matrix whereas a large variance causes the entries in the Gram matrix to approach a constant value. Training an SVM with a kernel function has the effect of computing a separating hyperplane that was embedded in a higher dimensional space. With the calculated separating hyperplane computed it is important to score and classify the new data. Note that when working with the primal Lagrange equation the weight vector can be calculated as $\bar{w} = X^T Y\bar{\alpha}$. By expanding this term and converting the products to a summation then the score for a linear SVM can be calculated as $z(\bar{x}_j) = \left(\sum_{i=1}^{m} \alpha_i y_i(\bar{x}_i \cdot \bar{x}_j)\right) + b$. Replacing the dot product with the kernel function $\kappa(\bar{x}_j, \bar{x}_i)$ gives a score for any kernel function such that $z(\bar{x}_j) = \left(\sum_{i=1}^{m} \alpha_i y_i \kappa(\bar{x}_j, \bar{x}_i)\right) + b$. This equation can also be written as $z(\bar{x}_j) = \bar{\alpha}^T Y\bar{k}_j + b$. As the non-support vectors have a Lagrange multiplier of zero it means the equation can be improved upon. This can be done by letting $N_s$ be the set of support vectors in the design matrix X. The score for a vector $\bar{x}_j$ of a kernel SVM can be calculated as $z(\bar{x}_j) = \left(\sum_{i \in N_s} \alpha_i y_i \kappa(\bar{x}_i, \bar{x}_j)\right) + b$. Any vector $\bar{x} \in R^n$ whether it is given data vector $\bar{x}_j$ or a new vector can be classified. The $\bar{x}$ is in Class +1 if its score is non-negative and is in Class -1 otherwise. Thus, the decision of how to classify $\bar{x}$ can be written as the function $q(\bar{x}) = sign(z(\bar{x}))$.

The effectiveness of lasso compared to OLS will be evaluated by plotting the values when attempting to predict the 'Lead' commodity price. This plot will include the predicted values using lasso, the predicted values using OLS, and the actual values. In order to evaluate the accuracy of classification using an SVM with an unscaled Gaussian kernel a percentage accuracy value will be reported.

## RESULTS

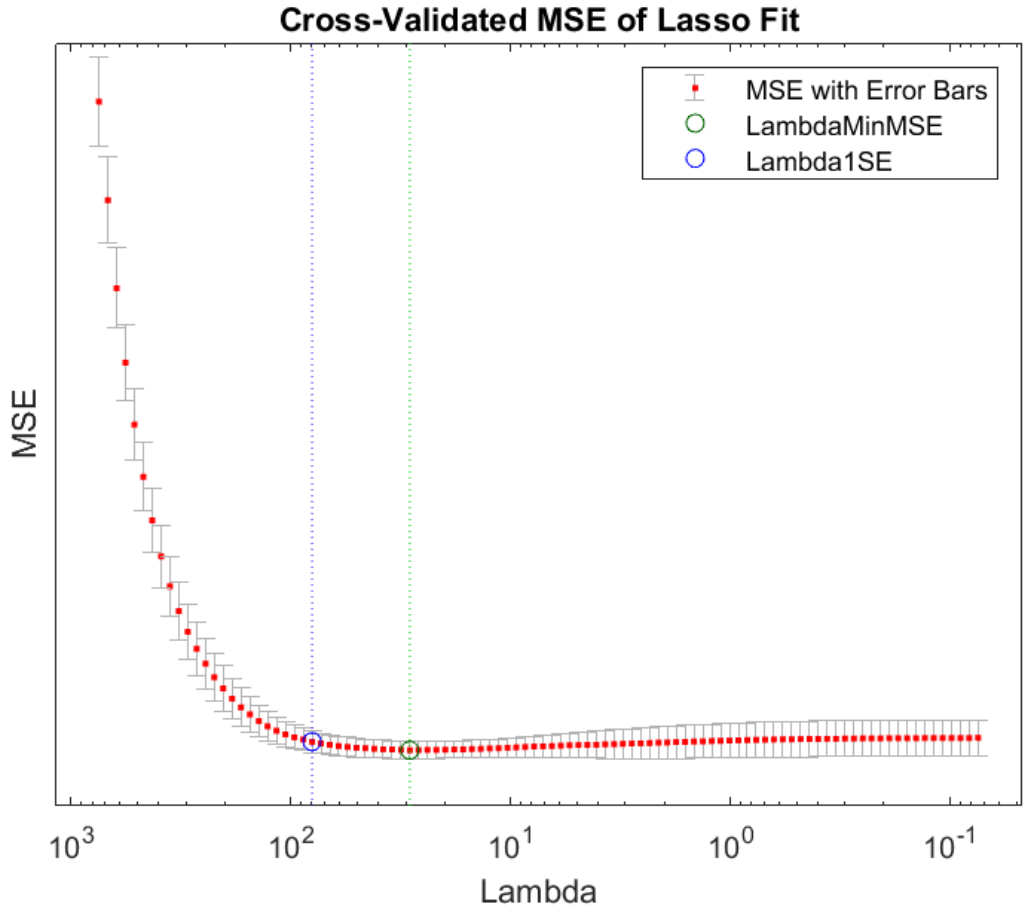


Figure 1: The following figure displays the mean MSE (red dot) across all 5 folds at each lambda value. At each fold error bars are evaluated and displayed. The green circle represents the minimum MSE and the blue circle represents the MSE one standard deviation away from the green circle.

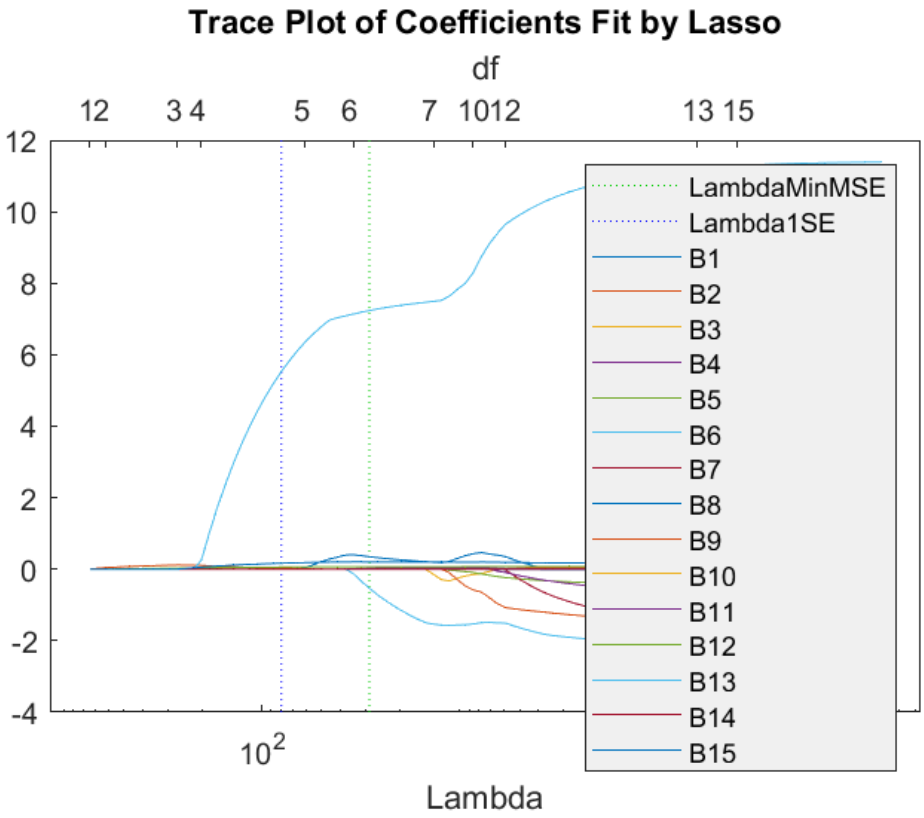


Figure 2: The following figure displays the coefficient values of each variable at each lambda value.

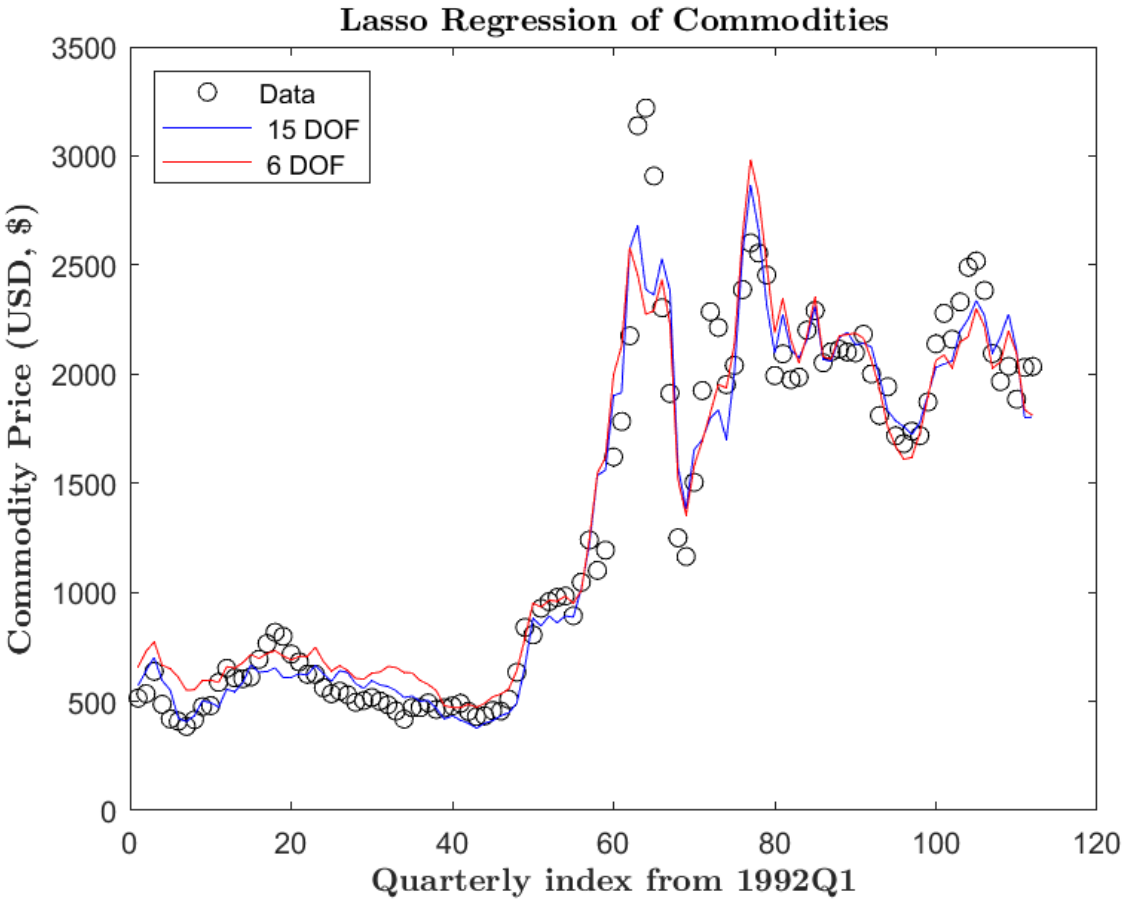


Figure 3: The following figure shows the known data points for the 'Lead' commodity as black circles. The blue line represents the OLS solution when predicting the 'Lead' commodity value. The red line indicates the lasso solution when predicting the 'Lead' commodity value.

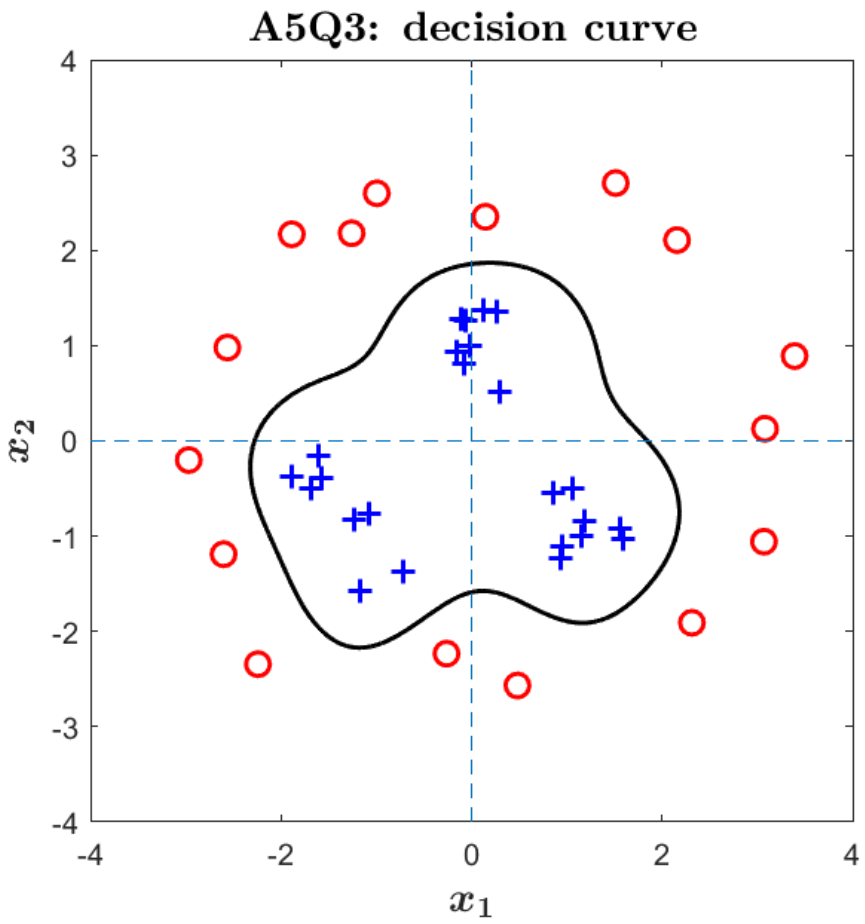


Figure 4: Data in 5D projected to 2D with classification hyperplane (black) with an accuracy of 100%. The blue crosses represent the classification class of +1. The red circles represent the classification class of -1.

## DISCUSSION

By observing **Figure 2**, it can be seen that at the 'LambdaMinMSE' value the variables B1, B4, B6, B12, B13, and B15 have non-zero values when applying lasso to the data set. **Figure 2** also displays that at the highest lambda value, only variable B2 has a non-zero value. As well **Figure 3**, displays the lasso predicted values with 6 degrees of freedom, the ordinary least squares (OLS) predicted values with 15 degrees of freedom, and the actual values for the 'Lead' commodity. The generated hyperplane when classifying the non-linearly separable data can be seen in **Figure 4**.

At the value 'LambdaMinMSE' the variables that had a non zero value represent the minimum number of other commodities to track. The minimum number of commodities was 6 and they were B1: 'Coal', B4: 'Fish meal', B6: 'Hides', B12: 'Tin', B13: 'Uranium', and B15: 'Zinc' **(Figure 2)**. The commodity that best reflects the broad market is B2: 'Copper' as it has a non-zero value at the highest lambda value **(Figure 2)**. As seen in **Figure 3**, lasso and OLS were able to accurately predict the 'Lead' commodity values. When comparing the methods, lasso was able to predict these values with only 6 degrees of freedom compared to OLS's 15 degrees of freedom. This makes lasso a much less computationally expensive prediction method as it is able to filter out redundant data. As well, **Figure 3** displays that OLS is more affected by outliers compared to lasso. As lasso allows for fewer variables, it is able to generate a more generalized solution meaning it is less affected by outliers. Overall, lasso provides

a better prediction as it is less computationally expensive and is less affected by outliers compared to OLS.

By observing **Figure 4**, it can be seen that using an SVM with an unscaled Gaussian kernel was able to accurately classify the non-linearly separable data with an accuracy of 100%. The data presented in **Figure 4** is not linearly separable, meaning that it cannot be separated by a single linear hyperplane. A method for classifying this non-linearly separable data is by pushing it up a dimension by using a mapping function. The mapping function used was a Gaussian kernel function. By using this Gaussian kernel, the points can be shifted up to a dimension where a hyperplane can be fit to separate the data. The hyperplane and points are then brought back to the second dimension.

Lasso provided a more accurate estimate of the 'Lead' commodity compared to OLS as it is less affected by outliers and less computationally expensive as it is able to eliminate redundant data. As well, lasso was able to determine that 'Copper' is the commodity that best represents the broad market and that the minimum number of other commodities to track is 6 (Coal, Fish meal, Hides, Tin, Uranium, and Zinc). Overall, SVM with an unscaled Gaussian kernel was an effective method for classifying the 5 dimensional data and achieved an accuracy of 100%.

## REFERENCES

[1] Kang, Hyun. "The Prevention and Handling of the Missing Data." *PubMed Central (PMC)*, 24 May 2013, www.ncbi.nlm.nih.gov/pmc/articles/PMC3668100.