Assignment 1

Gandhar Vaidya

September 30th , 2016

Contents

Abstract

In this report ridge regression is applied to the task of predicting wine quality is explored. The wine quality dataset from the UCI machine learning repository is used, and its accuracy obtained using ridge regression is compared to the results from a recent publication.

The wine data is composed of two datasets - one for white wines, and one for reds. All the analyses is performed on both. The features for this dataset are standardized, using the following PyML script:

1 Introduction

Ridge Regression is a technique for analyzing multiple regression data that suffer from multi-collinearity. When multi-collinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors. It is hoped that the net effect will be to give estimates that are more reliable.

2 Ridge Regression

This section shows data extraction and segregation of samples and quality(labels).The implementation of the code for Ridge Regression is explained in the following section along with the explanation. Ridge Regression is given by the following formula.

$$\beta ridge = (X^{\mathrm{T}}X + \lambda I)^{\text{-}1}X^{\mathrm{T}}y \tag{1}$$

2.1 Data Extraction

The data is extracted by using the guidelines provided wherein 80% of data is taken as training data and the rest is taken for testing the regression to implement RMSE and MAD errors. Python Code of the extraction :

```
# Creating Simple Data
if __name__=='__main__' :
    X=np.genfromtxt("red.txt", delimiter=";")
    X_red = X[:,0:11]
    y_red = X[:,11]
    a = int(0.8*len(X))

    X_white=np.genfromtxt("white.txt", delimiter=";")
    X_red = standardize(X_red)
    y_red = standardize(y_red)
    X_train = X_red[0:a,:]
    y_train = y_red[0:a]
    X_test = X_red[a:,:]
    y_test = y_red[a:]
```

2.2 RMSE and MAD errors for red and white wine samples

The RMSE and MAD errors were implemented by defining following functions. RMSE and MAD errors are being computed by using the testing data which has not been trained. RMSE and MAD errors are computed after using the training data to obtain weight vector "w" using Ridge Regression.

The codes used for implementing below plots is given in appendix 1.

2.2.1 White wine data

The below code is used for calculating Root Mean Square Error(RMSE).The hypothesis used for calculating the error is taken as the product of transpose of weight vector matrix with the features. The h varialbe in the program is the hypothesis function.

```
#Calculating RMSE for the given data set.
def rmse(X,y,w):
    temp = 0
    h = np.zeros(len(X))
    for i in range(len(X)):
        h[i] = np.dot(w.transpose(),X[i])
        temp +=((y[i] - h[i])**2)
    return ma.sqrt(temp/len(X))
```

The following code is used to import the math library for taking the square root for RMSE

```
import math as ma
```

The below code is used for calculating Maximum Absolute Deviation(MAD).The hypothesis function used for calculating the error is the same which is used for RMSE. The h varialbe in the program is the hypothesis function.

```
def mad(X,y,w):
    temp=0
    h = np.zeros(len(X))
    for i in range(len(X)):
        h[i] = np.dot(w.transpose(),X[i])
        temp +=abs(y[i] - h[i])
    return temp/len(X)
```

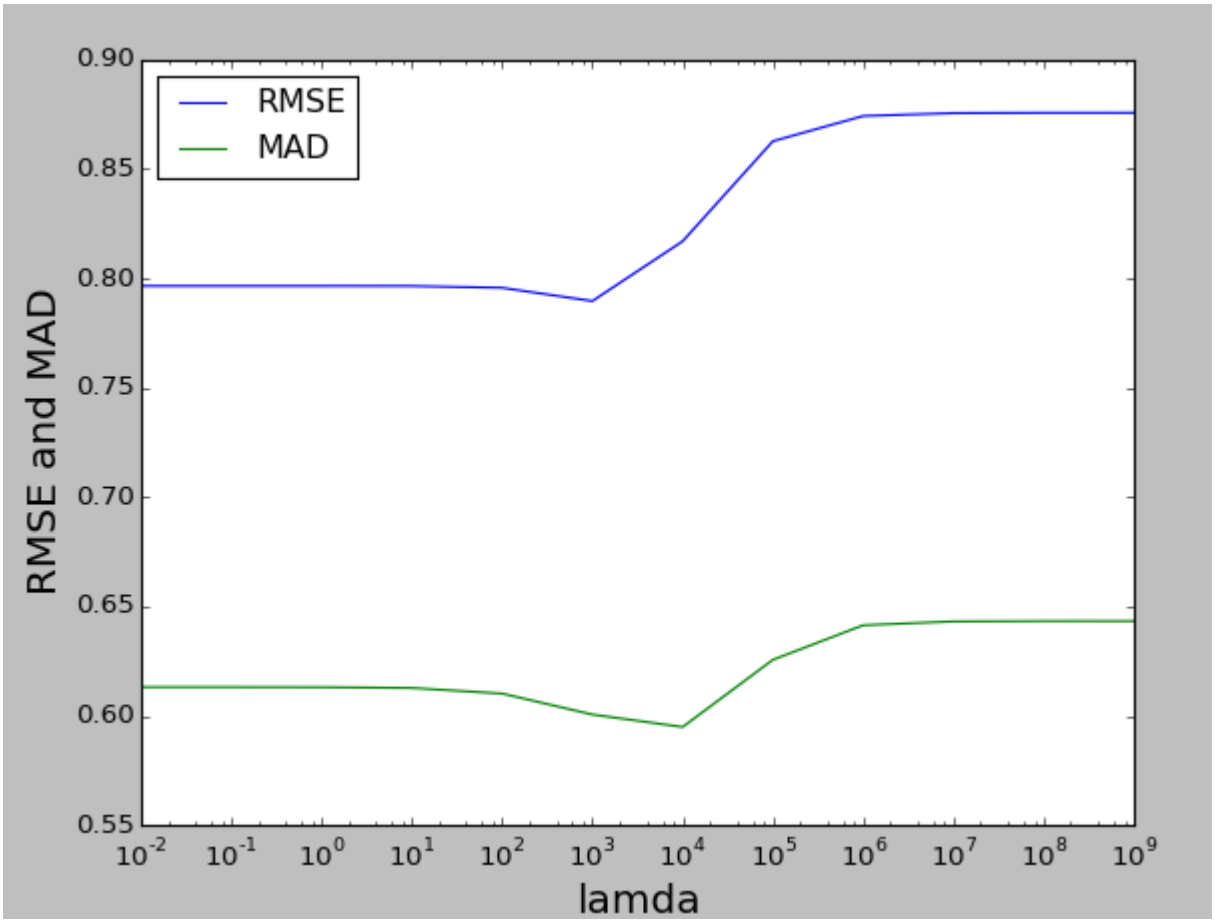

Figure 1: RMSE for white wine

The above figure shows the plot for RMSE vs MAD for the white wine data. As you can clearly see from the graph, both the errors stay constant for the initial phase i.e. till $\lambda = 10^2$.
After that there is a steep increase in the value of error. The error again gets saturated after$\lambda = 10^4$. But the MAD is always less than RMSE.
The least value of RMSE is obtained at $\lambda = 10^4$ and the value of $\lambda = 10^3$ gives the least MAD.

2.2.2 Red Wine data

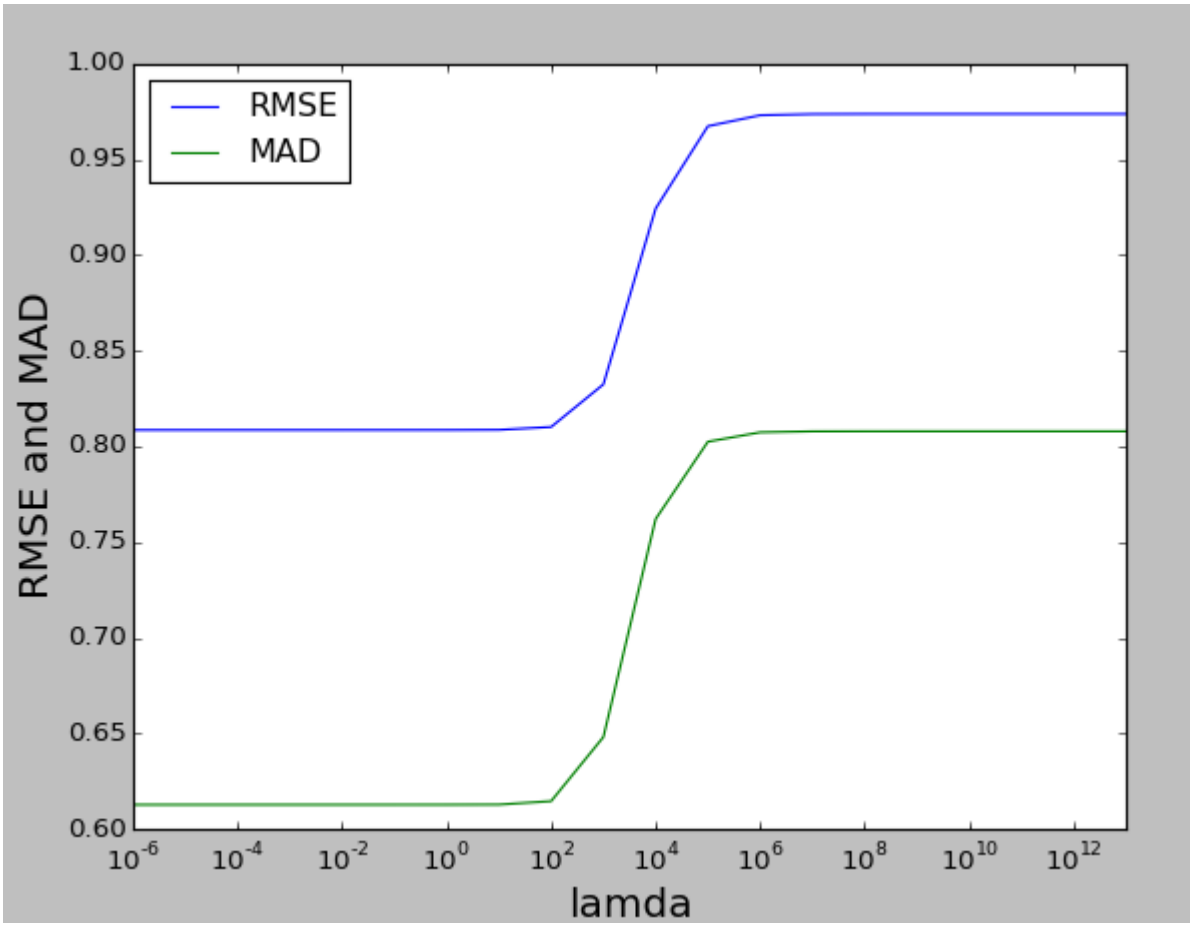

Figure 2: RMSE vs white for red wine

The above figure gives us a measure of the RMSE vs MAD deviation for the red wine dataset. RMSE and MAD are calculated using the same code as mentioned above. Both the curves exhibit similar shape with the MAD error being slightly lesser in magnitude than the RMSE error. For both, RMSE as well as MAD, $\lambda^{\min} = 10^2$

2.2.3 Discussion and Comparison

It is seen that, MAD and RMSE increase for increase in lambda after some value of lambda(>=100). For the significant small amount of lambda, the plots are not affected by lambda. It is also observed that the RMSE and MAD plots for lambda equals to or greater than 1000000 saturates. ***At such large lambda values the regularization parameter dominates over loss function and it is significantly larger than the largest feature in the loss function for finding RMSE or MAD values. It is observed that RMSE is always larger than MAD irrespective of the lambda.It is evident from Figure 1 that the ridge regression analysis gives MAD in the range of 0.60 to 0.65 for white wine and MAD in the range of 0.60 to 0.80 for red wine with insignificant standard error over lambda from 0.01 to 10. These values of MAD are statistically significant above MAD values obtained for SVM method in report [1]. But, these are not significantly different than those obtained for MR and NN methods used in the report[1]. The RMSE range for the red wine data set is 0.82 to 0.98 whereas that for the white wine data set is 0.80 to 0.90.***

2.3 REC curves

The Regression Error Characteristic (REC) curves of a representative classifier was created for both red wine and white wine datasets. ***The REC curves give more informaton compared to RMSE and MAD curves. From REC curves, one can learn about error tolerance and can choose error tolerance in the study of interest for required classifier accuracy.***

The code used to create REC curves is given in Appendix 2. The curves were created using absolute error and square error measures.. The created plots for Red wine and White wine are given below in Fig3 and Fig4 respectively.

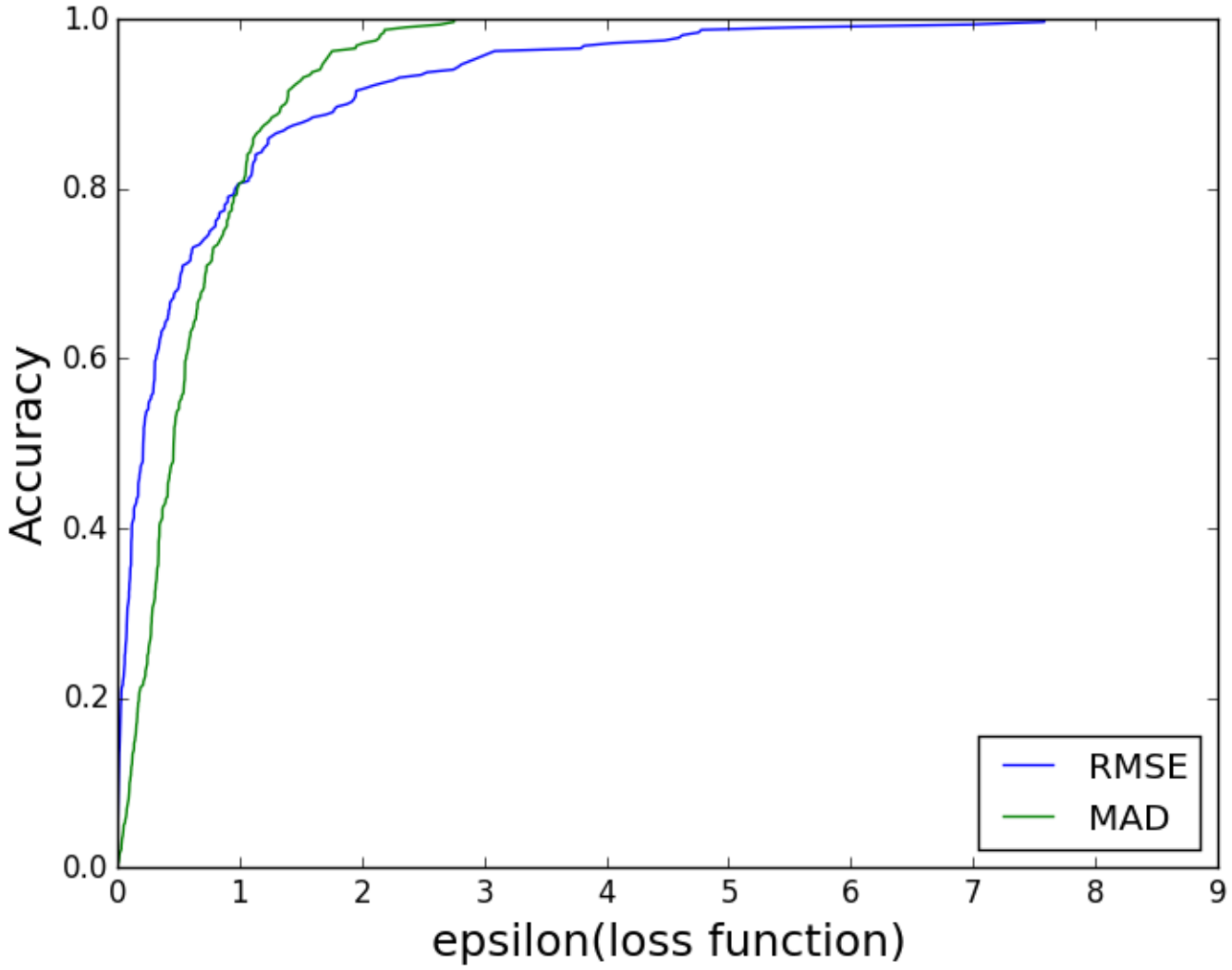

Figure 3: REC curve for red wine data set

For the above curve the value of λ is taken as 10^2 since that is the value for which we get minimum error as mentioned in the document above.

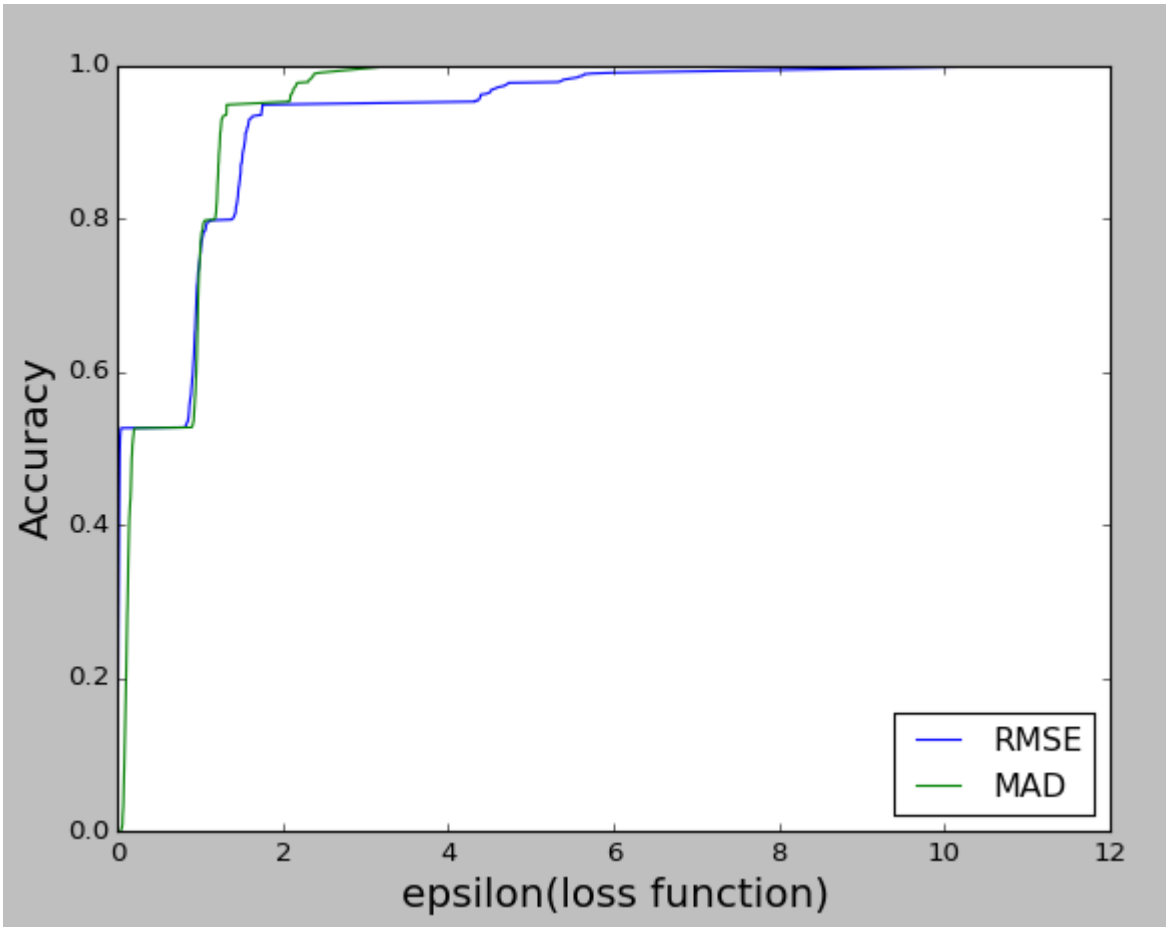

Figure 4: REC curve for white wine data set

For the above curve the value of λ is taken as 10^4 since that is the value for which we get minimum error as mentioned above in the document.

The Regression Error Characteristic (REC) curves of a representative classifier was created for both red wine and white wine datasets. ***The REC curves give more informaton compared to RMSE and MAD curves. From REC curves, one can learn about error tolerance and can choose error tolerance in the study of interest for required classifier accuracy.***

2.3.1 Discussion of MAD and RMSE plots

By seeing the MAD and RMSE plots for REC curve , it is clearly seen that sqaured error analysis is of more help when one requires a wider range of error tolerance to compare a data set. The use of sqaure error analysis expands the range of tolerance by a factor of 2 over the range of error tolerance generated from absolute deviation error analysis. As can be seen from the REC curves above, ***The value of Accuracy using RMSE converges to a value which is very close to the square of the value of epsilon in which accuracy of MAD converges to 1.***

3 Weight Vector Analysis

The magnitude of the weight vector can be interpreted as a measure of feature importance. The ridge regression classifier was trained on a subset of the data set that was reserved for training. It was done using the following for both red wine and white wine data:

```
1 if __name__=='__main__' :
```

```
X=np.genfromtxt("red.txt", delimiter=";")
X_red = X[:,0:11]
y_red = X[:,11]
X2=np.genfromtxt("white.txt", delimiter=";")
X_white = X2[:,0:11]
y_white = X2[:,11]
a = int(0.8*len(X2))
X_white = standardize(X_white)
X_train = X_white[0:a,:]
y_train = y_white[0:a]
w = rr(X_train,0.01,y_train)
X_test = X_white[a:,:]
y_test = y_white[a:]
```

In the above code the X matrix stands for the Red wine data set and X2 stands for white wine data set. Slicing is used to segregate data into training and testing samples for implementing ridge regression and weight vector analysis. As you can see the variable "a" is used to split the data in the ratio 80:20 in favor of training data.The relationship between the magnitude of weight vector components and their relevance to the classification task was explored in several ways.

3.1 Weight vector component against the Pearson correlation coefficient

Each feature is associated with a component of the weight vector. It can also be associated with the correlation of that feature with the vector of labels. A scatter plot of the weight vector component against the Pearson correlation coefficient of a feature against the labels was created. The code used for this study is given in Appeindix 3. The resultant plots are given in Figure 5 and Figure 6.

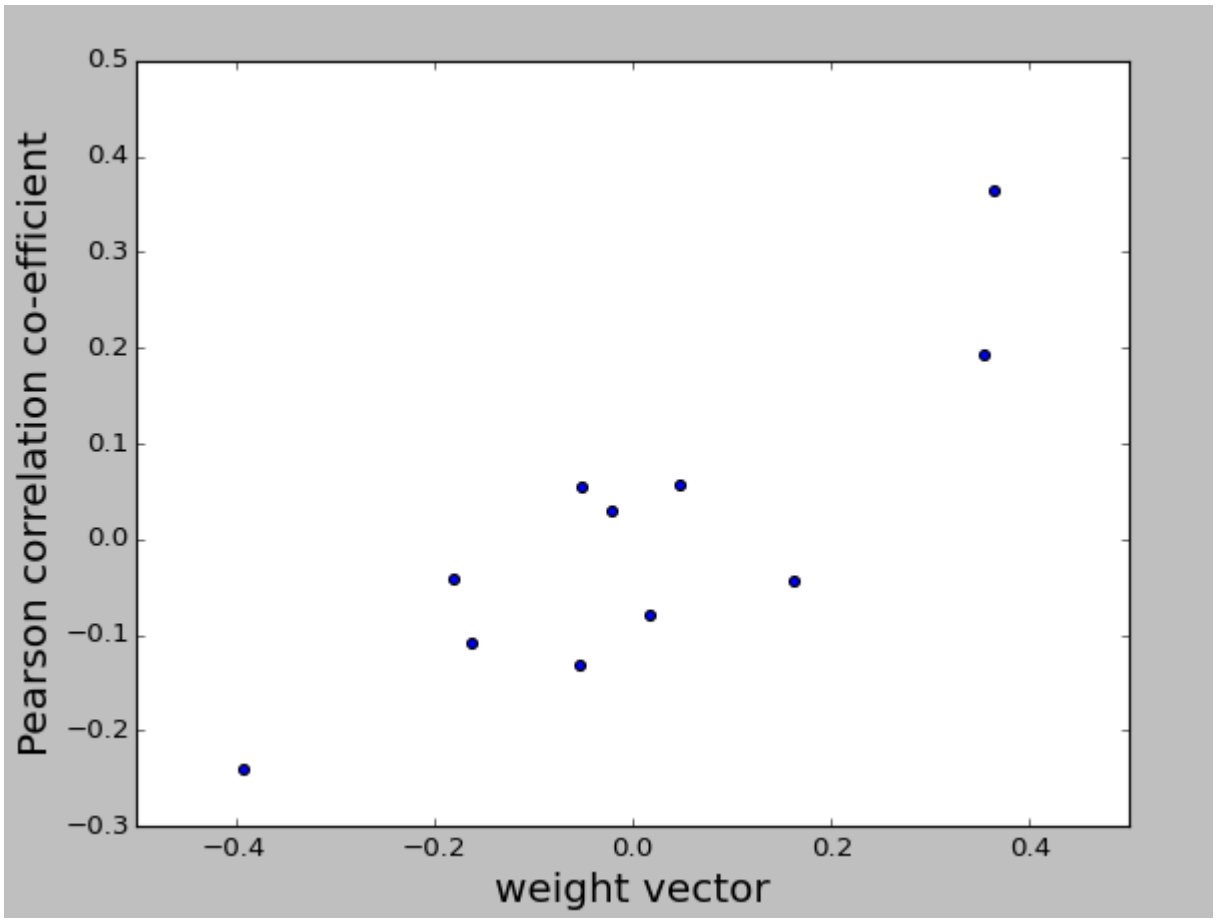

Figure 5: Scatter plot of Pearson co-relation co-efficient for red wine data set($\lambda = 100$)

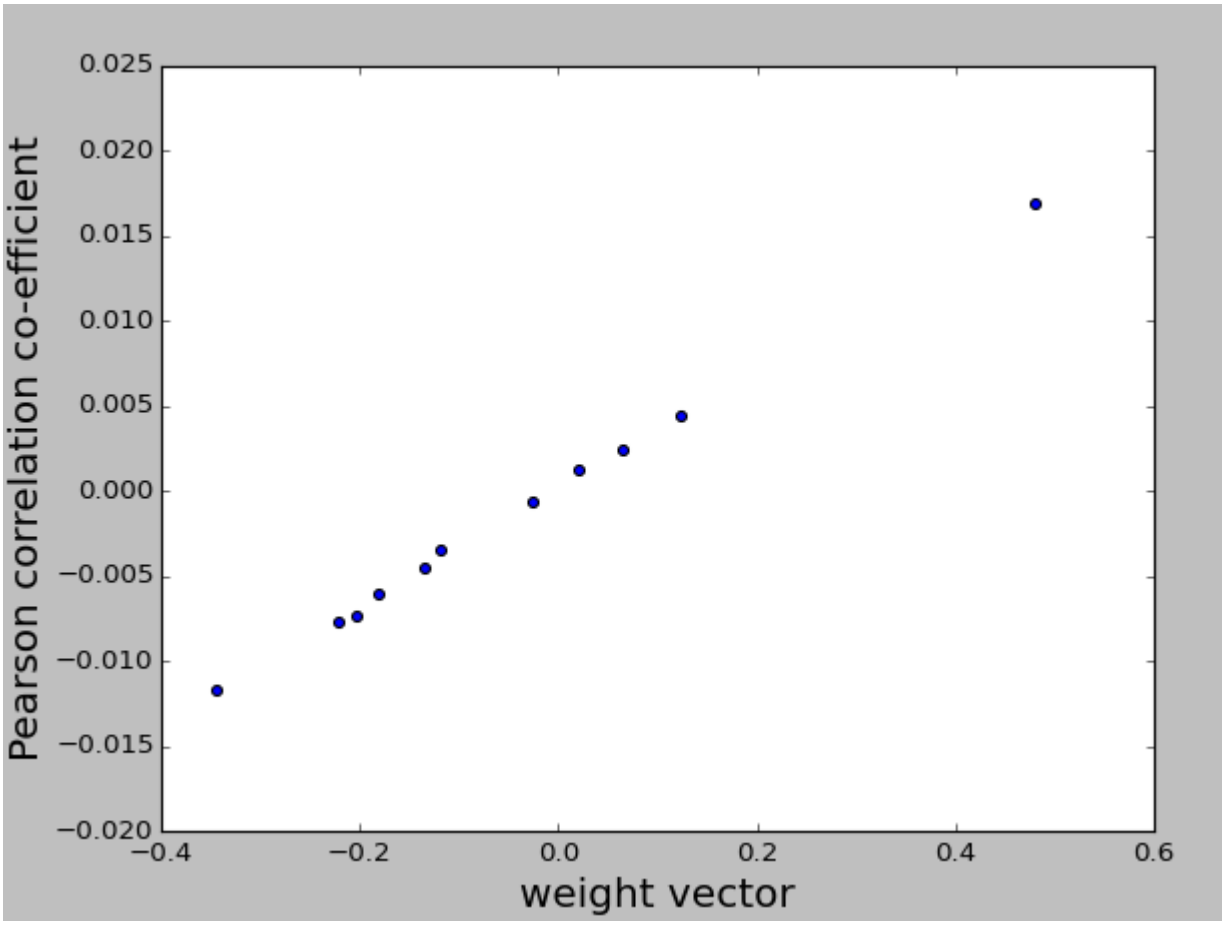

Figure 6: Scatter plot for white win data set of Pearson correlation co-efficient(for $\lambda = 10000$)

This proves that for the value of λ when RMSE and MAD is least you get a better correlation.

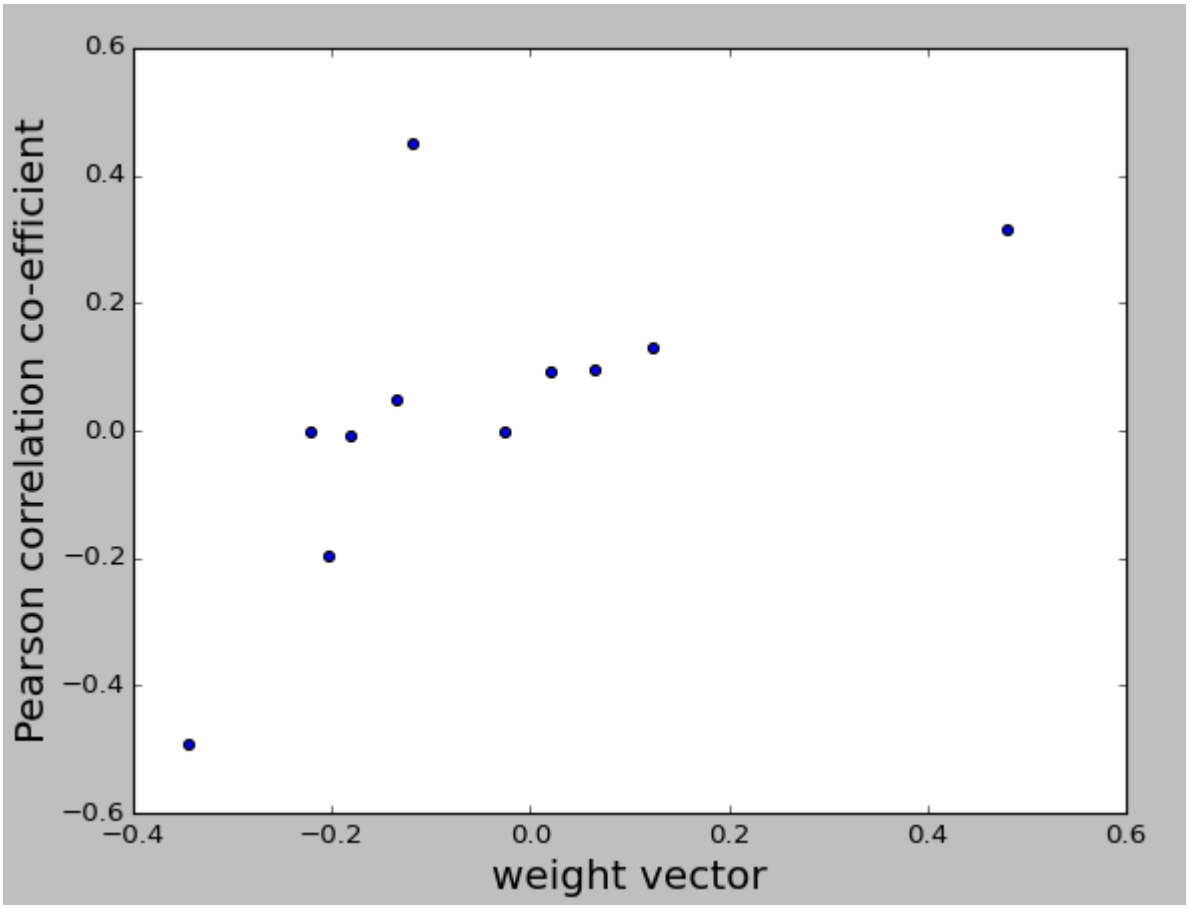

Figure 7: Scatter plot of Pearson co-relation co-efficient for white wine data set

It is evident from the above figures that the features with the larger values of weight vector components have relatively larger correlation with the created labels. Thus, the weight vector component and Pearson correlation coefficients are a measure of importance of features.

3.2 Weight vector removal experiment

The feature with the lowest absolute value of the weight vector were incrementally removed and the ridge regression classifier was retained. Plots of RMSE and MAD as a function of the number of features that remain on the test set is given in Figure 7 and 8 for red and white wine data set respectively. The code used to remove features and create this plot is given in Appendix IV. It is clear that features were eliminated from the test data set one by one.

In this experiment , we remove the features corresponding to the lowest value of weight vector and by re-training the ridge regression on a new data set, we obtain RMSE and MAD. The whole idea behind this experiment is to check the impact of weight vector components having larger values on RMSE and MAD errors.

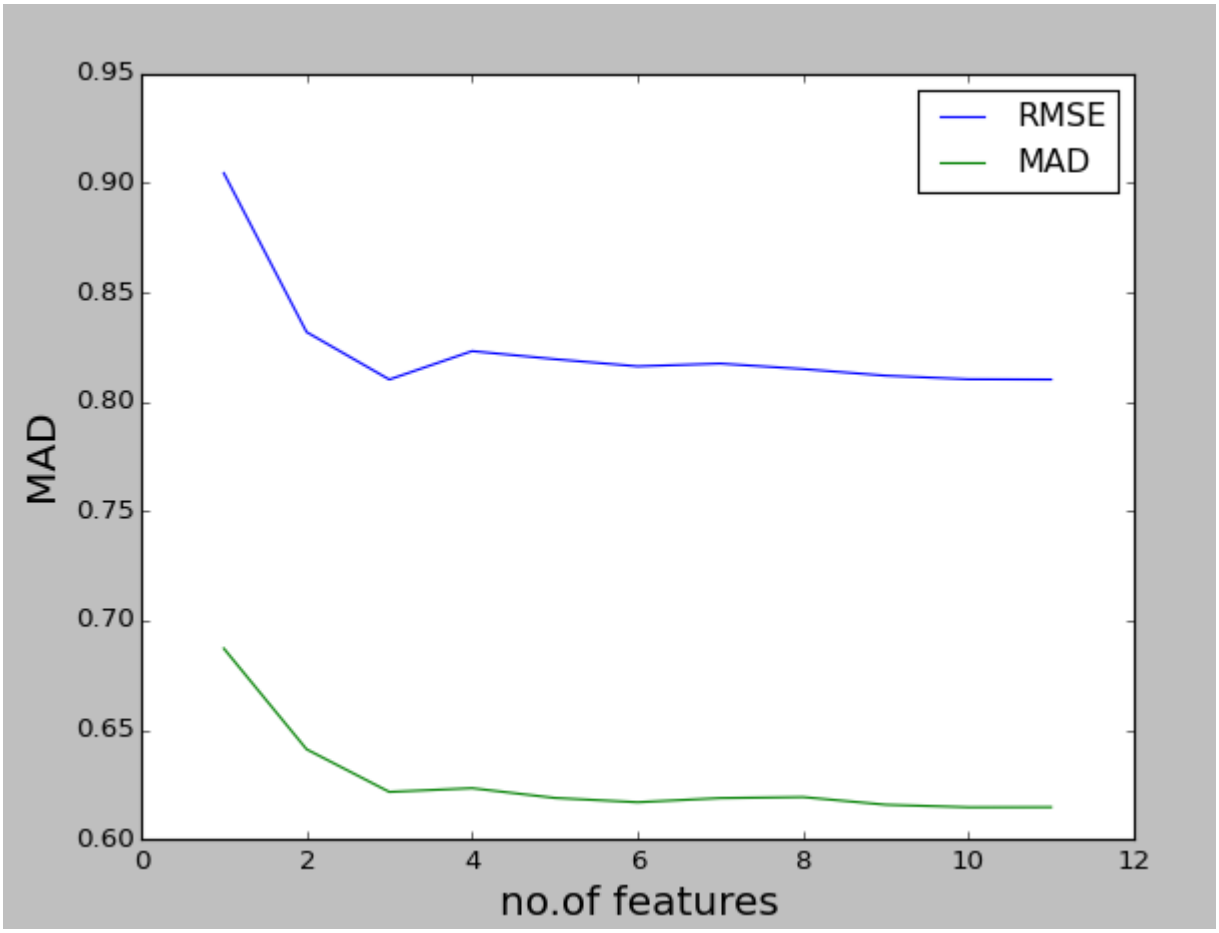

Figure 8: RMSE and MAD curves vs no.of features for red wine data set

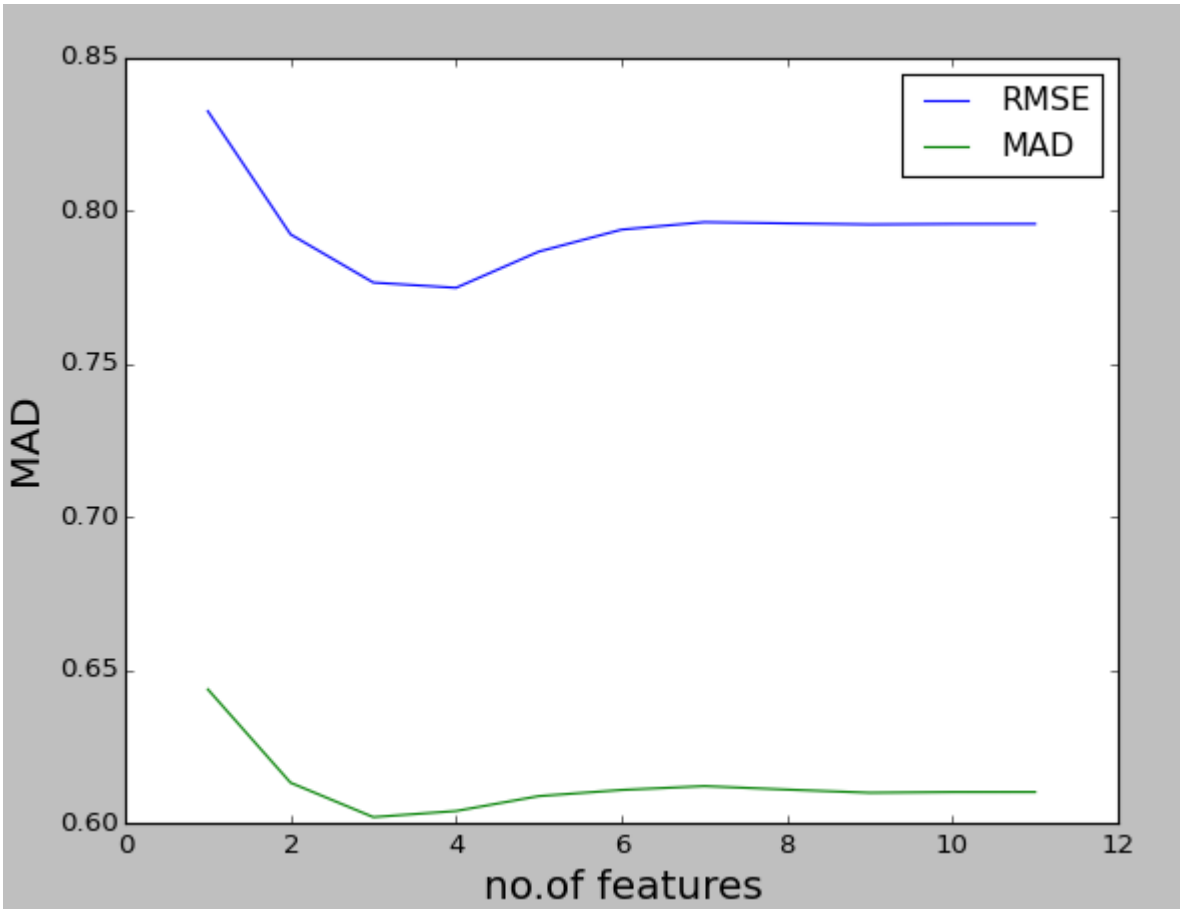

Figure 9: RMSE and MAD curves vs no.of features for white wine data set

In the plots in Figure 7 and 8, the RMSE and MAD values for the last 3 features are significantly higher.As we start removing the above features , the error goes on increasing. It is evident that these values of RMSE and MAD are significantly higher than the previous values. Thus, removing features with smaller weight vector component from the dataset has smaller effect on the RMSE/MAD values than removing the features with larger weight vector.

Paper[1] considers sulphates [feature 10] as most important and then 'alcohol' and 'residual sugar' have successive importance for white wine. In contrary, this study considers 'density' and 'residual sugar' as important as they have larger importance percentages.

Appendix 1 - Code for creating MAD and RMSE plots

The code inserted here is for white wine data. The similar code was used for red wine data as well.

```
def plot(X_train,y_train,X_test,y_test):
    lamb = np.zeros(20)
    r = np.zeros(20)
    m = np.zeros(20)
    j = 0
    for i in range(-6,14):
        lamb[j]=10**i
        w = rr(X_train,lamb[j],y_train)
        r[j] = rmse(X_test,y_test,w)
        m[j] = mad(X_test,y_test,w)
        j = j + 1

    plt.plot(lamb,r,label='RMSE')
    plt.xscale('log',basex=10)
    plt.xlabel('lamda',fontsize=18)
    plt.ylabel('RMSE and MAD',fontsize=18)
    plt.show()
    plt.plot(lamb,m, label='MAD')
    plt.xscale('log',basex=10)
    plt.legend(loc=2)
    plt.show()
```

In the above code, I have taken a sufficient range of indices for lambda to obtain RMSE and MAD curves.

Appendix 2 - Code for creating REC curves

The code inserted here is for RMSE. The similar code was used for MAD as well.

```
def REC_rmse(X,y,w):
    h = np.zeros(len(X))
    loss = np.zeros(len(X))
    for i in range(len(X)):
        h[i] = np.dot(w.transpose(),X[i])
        loss[i] = (h[i]-y[i])**2
    loss2 = np.sort(loss)
    eps=[]
    eps2 = 0.0
    hit=0.0
    accuracy =[]
    for i in range(len(X)):
        if (loss2[i]>eps2):
            accuracy.append(hit/len(X))
            eps2 = loss2[i]
            eps.append(eps2)
        hit = hit + 1
    accuracy.append(hit/len(X))
    eps.append(eps2)
    print 'eps',eps
    plt.xlim(0,9)
    plt.plot(eps,accuracy,label = 'RMSE')
    plt.xlabel('epsilon(loss function)',fontsize=18)
    plt.ylabel('Accuracy',fontsize=18)
    plt.legend(loc=4)
    plt.show()
```

In the above code loss function is taken as squared error means. The hit variable counts the number of samples that fall in the tolerance level and thus we get the accuracy after dividing it by the total number of samples.

Appendix 3 - Code for creating scatter plot

The code inserted here is for white wine data. The similar code was used for red wine data as well.

```
def pearson(X,y,w):
    pm = np.zeros(len(X[0]))
    s1 = np.std(X,0)
    s2 = np.std(y)
    for i in range(len(X[0])):
        cv = np.cov(X[:,i],y)
        print 'cv1',cv
        pm[i] = (cv[0][1])/s1[i]*s2
    print 'cv',cv[0][1]
    plt.scatter(pm,w)
    print 'pm',pm,'w',w
    print np.amax(pm)
    print np.amax(w)
    plt.xlabel('weight vector',fontsize=18)
    plt.ylabel('Pearson correlation co-efficient',fontsize=18)
    plt.show()
```

The pm matrix stands for all the pearson correlation co-efficients. The covariance array is denoted by cv.

Appendix 4 - Code for creating RMSE and MAD with successive feature removal

The code inserted here is for white wine data. The similar code was used for red wine data as well.

```
def weightremoval(w,lamb,y_train,X_train,X_test,y_test):
    rm = np.zeros(11)
    ma = np.zeros(11)
    ind = []
    values = np.arange(11,0,-1)
    for i in range(0,11):
        w = rr(X_train,lamb,y_train)

        if (w!=[]):
            j=(np.argmin(abs(w)))
            print j
        ind.append(j)
        rm[i] = rmse(X_test,y_test,w)
        ma[i] = mad(X_test,y_test,w)
        X_train = np.delete(X_train,j,1)
        X_test = np.delete(X_test,j,1)
        print 'X',np.shape(X_train)
    plt.plot (values,rm, label='RMSE')
    plt.xlabel('no.of features',fontsize='18')
    plt.ylabel('RMSE',fontsize='18')
    plt.show()
    plt.plot (values,ma, label='MAD')
    plt.xlabel('no.of features',fontsize='18')
    plt.ylabel('MAD',fontsize='18')
    plt.legend(loc=1)
    plt.show()
```

The rr function in the program does the ridge regression and returns the weight vector which then is used to calculate RMSE and MAD errors. The np.delete() method is used to delete the least significant feature from the weight vector.

4 References

1.Cortez et al, "Modeling wine preferences by data mining from physicochemical properties", Elsevier, May, 2009. 2. https://www.biomedware.com/files/documentation/Preparing_data/Why_standardize_variables.htm
3. Lecture slides from CS 545
4. Mostafa, Y.,& Ismail, M. (2012). Learning from data: A short course.