# **Homework 3**

AMATH 482

Christian Diangco

## **Abstract**

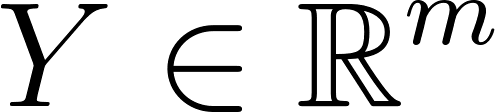
This paper covers the theory and use of supervised machine learning, linear regression, ridge regression, and cross validation..The process is discussed in Sec. 3. Algorithm Implementation and Development, after which results and visualizations are displayed in Sec. 4. Computational Results.

## **Sec. 1. Introduction and Overview**

In this assignment we are tasked with developing an algorithm that predicts the quality of wine from a series of chemical measurements. Each wine is given a score from 0 to 10 based on 11 different features. We will utilize linear and ridge regression in order to give scores for a new batch of wines. 10-fold Cross validation will be used to find the optimal parameters to use for the Gaussian and Laplacian kernels for ridge regression.

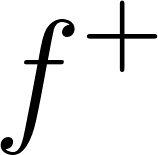
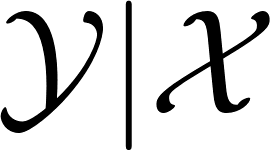
## **Sec. 2. Theoretical Background**

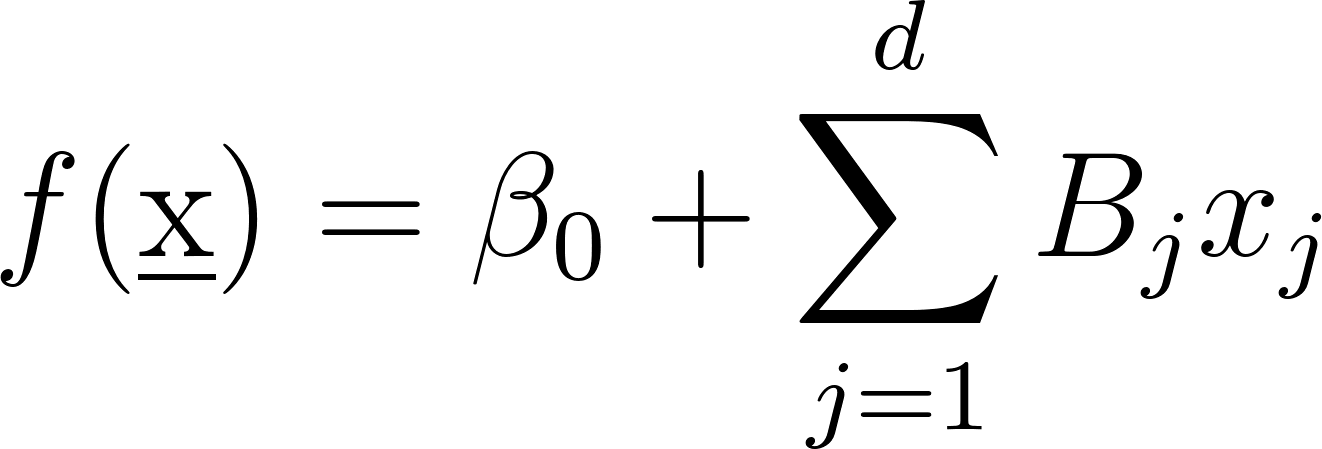
This homework requires us to utilize supervised machine learning techniques, In standard machine learning terminology, we consider a set of points of features or inputs, where [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=X%20%3D%20%5Cmathbb%7BR%7D%5Ed#0). We can thus again think of as a matrix [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=X%20%5Cin%20%5Cmathbb%7BR%7D%5E%7Bd%20%5Ctimes%20N%7D#0).

Associated with these features/inputs are a collection of outputs/responses . is often thought of as either a Euclidean space [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=Y%20%5Cin%20%5Cmathbb%7BR%7D%5Em#0) or space of categories e.g. , in this case the features are the wine scores. The pair of input and output sets is referred to as a training dataset. The goal of supervised learning is, given the training dataset, predict responses/output for any new input/feature (.

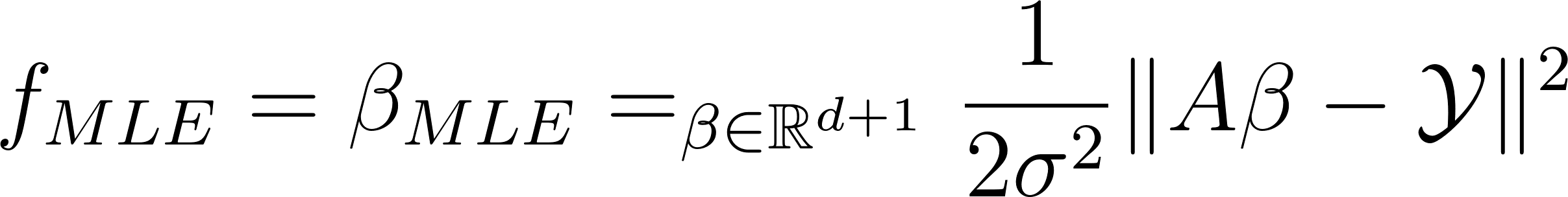
We will also be fitting our data to different regression models. The first one being Linear/Least Squares Regression. We assume there exists a function

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=f%5E%2B%3A%5Cmathcal%7BX%7D%5Crightarrow%5Cmathcal%7BY%7D%20%5Ctext%7B%20so%20that%20%7D%20y_j%3Df(%5Cunderbar%7Bx%7D_j)%2B%5Cvarepsilon_j#0)

Where are some noise. We can formulate an optimization problem for funding [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=f%5E%2B#0) by maximizing the likelihood of [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=%5Cmathcal%7BY%7CX%7D#0). This is called a maximum likelihood estimator (MLE). We make some assumptions on the form of for the linear regression model,

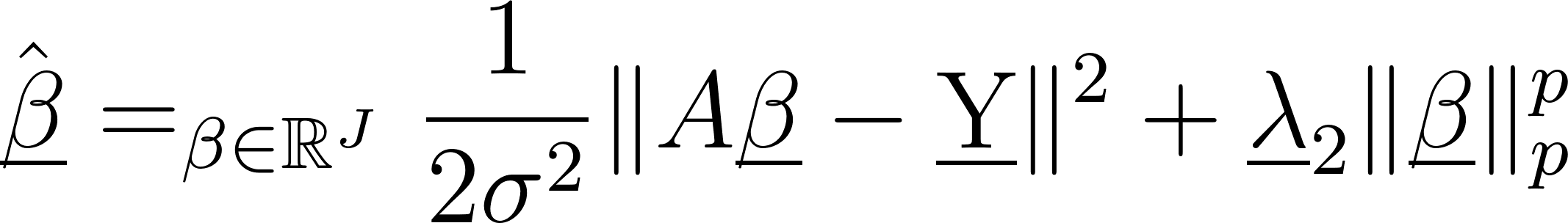
[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=f(%5Cunderbar%7Bx%7D)%3D%5Cbeta_0%2B%5Csum_%7Bj%3D1%7D%5EdB_jx_j#0)

If we assume is an affine transformation of [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=%5Cunderbar%7Bx%7D#0), then our MLE takes the form

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=f_%7BMLE%7D%3D%5Cbeta_%7BMLE%7D%3D%5Cargmin_%7B%5Cbeta%5Cin%5Cmathbb%7BR%7D%5E%7Bd%2B1%7D%7D%5Cfrac%7B1%7D%7B2%5Csigma%5E2%7D%5C%7CA%5Cbeta-%5Cmathcal%7BY%7D%5C%7C%5E2#0)

MLE is a least squares solution to .

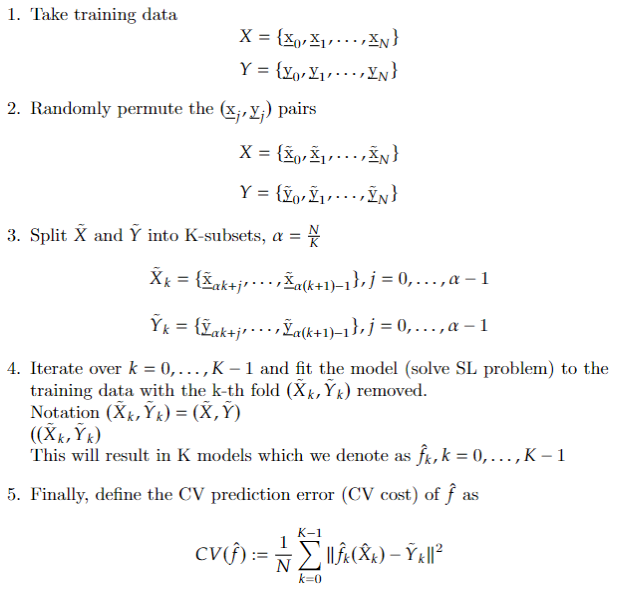
We will also be utilizing Ridge Regression. Consider

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=%5Chat%7B%5Cunderbar%7B%5Cbeta%7D%7D%3D%5Cargmin_%7B%5Cbeta%5Cin%5Cmathbb%7BR%7D%5EJ%7D%5Cfrac%7B1%7D%7B2%5Csigma%5E2%7D%5C%7CA%5Cunderbar%7B%5Cbeta%7D-%5Cunderbar%7BY%7D%5C%7C%5E2%2B%5Cunderbar%7B%5Clambda%7D_2%5C%7C%5Cunderbar%7B%5Cbeta%7D%5C%7C_p%5Ep#0),

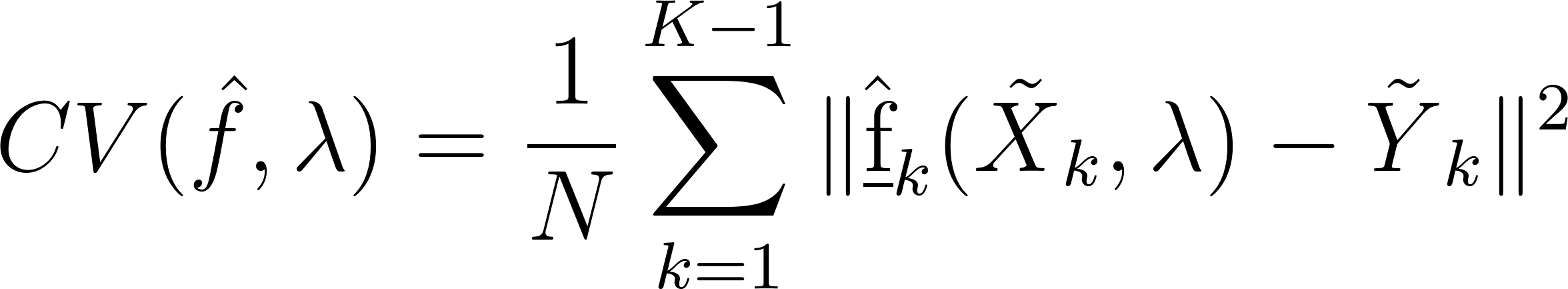
Where is called the regularization/penalty parameter and denotes the choice of the norm on [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=%5Cunderbar%7B%5Cbeta%7D#0). When , this case is known as Ridge Regression. We will be using both the Gaussian (RBF) kernel as well as the Laplacian kernel as described below,



In order to choose an optimal value for and for Ridge Regression, we will also be using K-fold Cross Validation (CV), in this case . The idea of CV is to split the training set into pieces, where some pieces will be used for training and others are used for testing. The steps for CV are described as follows:



In order to find the optimal choice for for instance, we minimize

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=CV(%5Chat%7Bf%7D%2C%5Clambda)%3D%5Cfrac%7B1%7D%7BN%7D%5Csum_%7Bk%3D1%7D%5E%7BK-1%7D%5C%7C%5Chat%7B%5Cunderbar%7Bf%7D%7D_k(%5C~%7BX%7D_k%2C%5Clambda)-%5C~%7BY%7D_k%5C%7C%5E2#0).

## **Sec. 3. Algorithm Implementation and Development**

After loading the data, the first step we must do is to normalize and center the input features and outputs for the training, test, and new batch data. For example, we use the following code to do this for the training features

X\_train\_normal = (X\_train - repmat(X\_train\_mean, X\_train\_N,

1))/repmat(X\_train\_std, X\_train\_N, 1)

Where X\_train is the original data, X\_train\_std is the standard deviation of the original data, and X\_train\_N is the number of rows. repmat() comes from NumPy’s matrix library.

Once we have the normalized data, we can use them to fit our regression models. We use sci-kit learn’s LinearRegression() and KernelRidge() for Linear Regressions and Kernel Ridge Regression respectively.

In order to perform cross validation, we use cross\_val\_score() from sci-kit learn, setting the cv parameter to 10. I used the helper code from Lecture 16 to find the optimal values for and

K\_sgm = 20

K\_lmbd = 20

sgm = np.linspace(-5, 5, K\_sgm)

lmbd = np.linspace(-5, 5, K\_lmbd)

scores = np.zeros((K\_sgm, K\_lmbd))

scores\_std = np.zeros((K\_sgm, K\_lmbd))

KRR\_CV = kernel\_ridge.KernelRidge(kernel='rbf')

for i in range(K\_sgm):

KRR\_CV.gamma = 1/(2\*(2\*\*sgm[i])\*\*2)

for j in range(K\_lmbd):

KRR\_CV.alpha = (2\*\*lmbd[j])

this\_score = model\_selection.cross\_val\_score(KRR\_CV, X\_train\_normal, Y\_train\_normal, scoring= 'neg\_mean\_squared\_error', cv=10, n\_jobs=-1)

scores[i,j] = (np.mean(this\_score))

scores\_std[i,j] = (np.std(this\_score))

ij\_max = np.array( np.where( scores == scores.max() ), dtype=int).flatten()

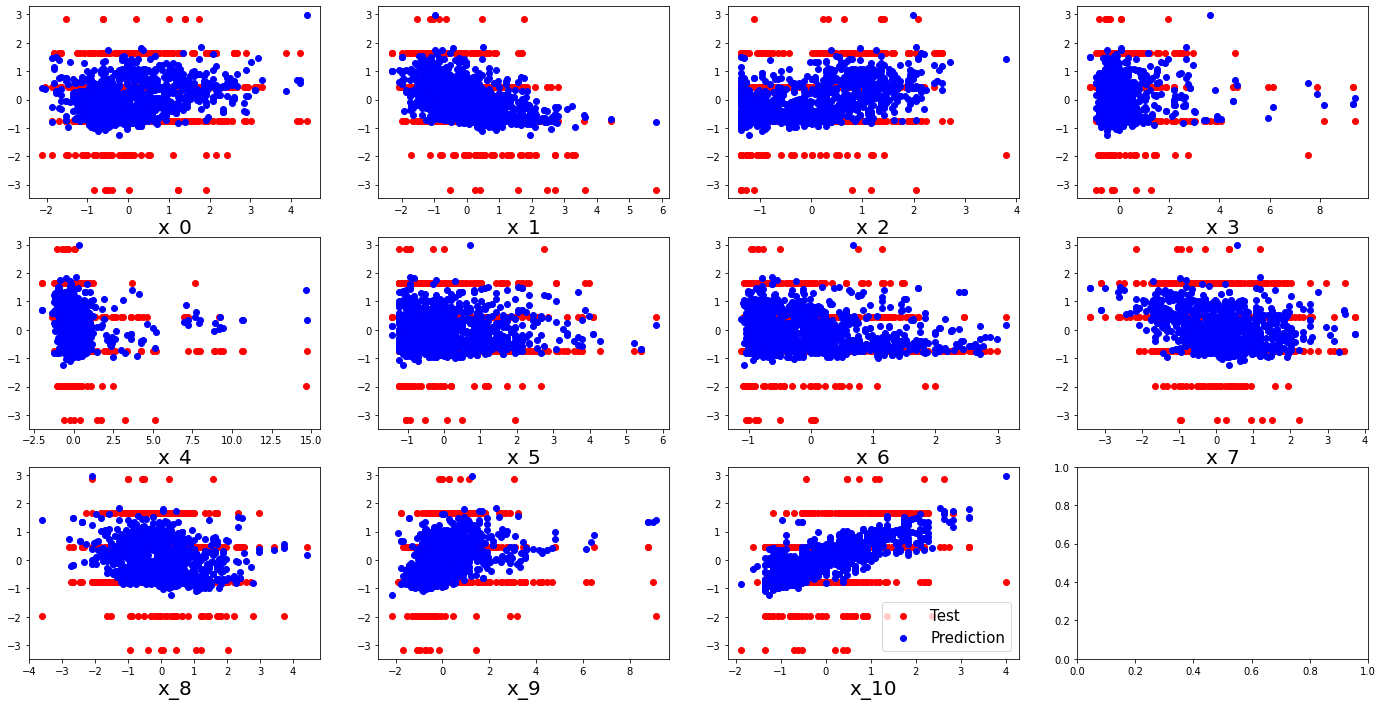
print(ij\_max)

print('log\_2 sg:', sgm[ij\_max[0]], 'log\_2\_lmbd: ', lmbd[ij\_max[1]])

print("sigma: ", 2\*\*sgm[ij\_max[0]], "lambda: ", 2\*\*lmbd[ij\_max[1]])

## **Sec. 4. Computational Results**

After fitting the normalized data to the Linear Regression Model, we compare the test data to the model’s predicted values.



(Figure 1) Plot comparing test values (red) and predicted values (blue) for the test data.

The optimal parameters I found for the Gaussian and Laplacian kernels are described in the table below.

|  | Gaussian Kernel | Laplacian Kernel |
| --- | --- | --- |
|  | 3.585328384551423 | 1.7284437865632103 |
|  | 0.27891447944038594 | 0.27891447944038594 |

We see that is greater for the Gaussian Kernel than the Laplacian Kernel. Interestingly, the value for I obtained for both the Gaussian and Laplacian Kernels are the same.

The training and test mean squared errors (MSE) I obtained for each model are displayed below:

|  | Linear Regression | Ridge Regression (Gaussian Kernel) | Ridge Regression (Laplacian Kernel) |
| --- | --- | --- | --- |
| Training MSE | 0.613286664489905 | 0.8204280967317644 | 0.05139203022413222 |
| Test MSE | 0.6577236288498409 | 0.7688150701221311 | 0.5782050859027209 |

We see that the Gaussian Kernel has the highest MSE, while the Laplacian Kernel has the lowest. Notably, the training MSE for the Laplacian Kernel is very low.

The scores for the quality of the new batch of wines predicted by each model are shown below:

|  | Wine 1 | Wine 2 | Wine 3 | Wine 4 | Wine 5 |
| --- | --- | --- | --- | --- | --- |
| Linear Regression | 6.04844329 | 5.51261797 | 5.45387107 | 6.43191258 | 6.14397344 |
| Ridge Regression (Gaussian Kernel) | 5.78733751 | 5.59711445 | 5.61717949 | 5.8751159 | 5.82293072 |
| Ridge Regression (Laplacian Kernel) | 5.81073403 | 5.638363 | 5.5886698 | 5.79239741 | 5.72115072 |

## **Sec. 5. Summary and Conclusions**

Using 10-fold cross validation we found the “optimal” values for and for ridge regression using Gaussian and Laplacian kernels. We found that was the same for both. While each model gave relatively similar scores to each of the wines, we found that ridge regression with the Laplacian kernel had the lowest MSE among the three models.

## **Acknowledgments**

I would like to acknowledge the students in the AMATH 582/482 Discord who allowed me to have fruitful discussion about this assignment.

## **References**

Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature

585, 357–362 (2020). <https://doi.org/10.1038/s41586-020-2649-2>

J. D. Hunter, "Matplotlib: A 2D Graphics Environment," in Computing in Science &

Engineering, vol. 9, no. 3, pp. 90-95, May-June 2007, doi: 10.1109/MCSE.2007.55.

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.