UMC-203 Assignment-2

Aditya Gupta
SR No: 22205

1 Support Vector Machine

1.1 The binary case

We are given a dataset with 10 different labels. We choose two labels, which turn out to be 5 and 9 according to my SR No. We name them 1 and -1 respectively. Before proceeding, we augment the data with a 1 to account for the bias term. Then we solve the following optimization problems.

1.1.1 Primal slack linear SVM

The primal slack linear SVM problem is defined as follows:

$$\begin{aligned}\min_{w,b,\xi} \quad & \frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \dots, n \\ & \xi_i \geq 0, \quad i = 1, 2, \dots, n\end{aligned}$$

The cvxopt library solves the quadratic minimisation problem as:

$$\begin{aligned}\min_{x} \quad & \frac{1}{2}x^T P x + q^T x \\ \text{subject to} \quad & Gx \leq h \\ & Ax = b\end{aligned}$$

To solve this, we first define the variable x as $[w_1, w_2, \dots, w_{785}, \xi_1, \xi_2, \dots, \xi_{1000}]^T$. Here $n = 1000$ and dimension of w is 785 (784 + 1 due to augmentation).

Since we have no equalities here, A and b are 0. We define the matrices P, q, G, h as follows:

$$P = \begin{bmatrix} I_{784\times784} & 0_{784\times1001} \\ 0_{1001\times784} & 0_{1001\times1001} \end{bmatrix}$$

$$q = [0_{784}, 0_{1000}]^T$$

$$G = \begin{bmatrix} -y_1 x_1^T & \\ \vdots & -I_{1000\times1000} \\ -y_{1000} x_{1000}^T & \\ 0_{1000\times785} & -I_{1000\times1000} \end{bmatrix}$$

$$h = [-1_{1000}, 0_{1000}]^T$$

The accuracies received for different values of C are:

C	Accuracy
1	97.5 %
10	97.5 %
100	97.5 %

1.1.2 Dual slack linear SVM

The dual slack linear SVM problem is defined as follows:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{subject to} \quad & 0 \le \alpha_i \le C, \quad i = 1, 2, \ldots, n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & w = \sum_{i=1}^{n} \alpha_i y_i x_i \end{aligned}$$

To solve this in cvxopt, we first define the variable x as $[\alpha_1, \alpha_2, \ldots, \alpha_{1000}]^T$. We define the matrices P, q, G, h, A, b as follows:

$$P = \begin{bmatrix} y_1 y_1 x_1^T x_1 & y_1 y_2 x_1^T x_2 & \cdots & y_1 y_{1000} x_1^T x_{1000} \\ y_2 y_1 x_2^T x_1 & y_2 y_2 x_2^T x_2 & \cdots & y_2 y_{1000} x_2^T x_{1000} \\ \vdots & \vdots & \ddots & \vdots \\ y_{1000} y_1 x_{1000}^T x_1 & y_{1000} y_2 x_{1000}^T x_2 & \cdots & y_{1000} y_{1000} x_{1000}^T x_{1000} \end{bmatrix}$$

$$q = [-1_{1000}]^T$$

$$G = \begin{bmatrix} -I_{1000\times1000} \\ I_{1000\times1000} \end{bmatrix}$$

$$h = \begin{bmatrix} 0_{1000} \\ C_{1000} \end{bmatrix}$$

$$A = y^T$$

$$b = [0]$$

The accuracies received for different values of C are:

C	Accuracy
1	65 %
10	65 %
100	65 %

The reduced accuracy of the dual can be attributed to the fact that we do not have a good way to get the value of the bias for the dual problem. (the accuracy is higher for some other random SR No values, since the bias term changes for every dataset)

1.1.3 Kernelized SVM

The kernelized SVM problem is defined as follows:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned}$$

The accuracies received for different values of C are:

C	Accuracy
1	50 %
10	50 %
100	50 %

1.1.4 Scikit-learn linear SVM

The accuracies received for different values of C are:

C	Accuracy
1	100 %
10	100 %
100	100 %

1.2 Multiclass SVM

1.2.1 Classifier

The multiclass classifier can be given as:

$$h(x) = \underset{i \in \{1,\ldots,k\}}{argmax}\{w_i^T x\}$$

1.2.2 Lagrangian Dual

To solve the multiclass Lagrangian, we introduce the Kronecker delta in the primal SVM problem. The primal problem can be given as:

$$\begin{aligned} \min_{w_i,\xi_i} \quad & \frac{1}{2}\sum_{k=1}^{K} w_k^T w_k + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & w_{y_i}^T x_i - w_k^T x_i \geq 1 - \xi_i - \delta_{y_i,k} \quad (x_i, y_i) \in D \\ & k = 1, 2, \ldots, K \end{aligned}$$

The Lagrangian can then be given as:

$$L(w_i, \xi_i, \lambda) = \frac{1}{2}\sum_{k=1}^{K} w_k^T w_k + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\sum_{k=1}^{K}\lambda_{i,k}(1 - \xi_i - \delta_{y_i,k} - w_{y_i}^T x_i + w_k^T x_i)$$

To solve for optimality, we take derivatives with respect to w_i and ξ_i and set them to 0. We get the following equations:

$$\begin{aligned} \frac{\partial L}{\partial \xi_i} = 0 &\implies \sum_{k=1}^{K}\lambda_{i,k} = C \\ \frac{\partial L}{\partial w_k} = 0 &\implies w_k = -\sum_{i=1}^{n}\lambda_{i,k}x_i + \sum_{i=1}^{n}\left(\sum_{k=1}^{K}\lambda_{i,k}\right)x_i \end{aligned}$$

We can solve for w_k further to get:

$$\begin{aligned} w_k &= -\sum_{i=1}^{n}\lambda_{i,k}x_i + C\sum_{i=1, y_i=k}^{n} x_i \\ &= -\sum_{i=1}^{n}\lambda_{i,k}x_i + C\sum_{i=1}^{n}\delta_{y_i,k}x_i \\ &= \sum_{i=1}^{n}(C\delta_{y_i,k} - \lambda_{i,k})x_i \end{aligned}$$

Substituting these values back in L, we get:

$$L(\lambda) = -\frac{1}{2}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^T x_j \sum_{k=1}^{K}(C\delta_{y_i,k} - \lambda_{i,k})(C\delta_{y_j,l} - \lambda_{j,l})\right] - \left(\sum_{i=1}^{n}\sum_{k=1}^{K}\lambda_{i,k}\delta_{y_i,k}\right)$$

Define $\alpha_i = Ce_i - \lambda_i = (C\delta_{y_i,1} - \lambda_{i,1}, \ldots, C\delta_{y_i,K} - \lambda_{i,K})^T$

$$\therefore L(\lambda) = -\frac{1}{2}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i^T x_j)(\alpha_i^T \alpha_j)\right] - \sum_{1}^{n} \alpha_i^T e_i$$

where $e_i = (0, \ldots, 1, \ldots, 0)^T$ with 1 at the i^{th} position.

Now the dual optimization problem can be given by:

$$\begin{aligned} \min_{\alpha_1,\ldots,\alpha_n} \quad & \frac{1}{2}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i^T x_j)(\alpha_i^T \alpha_j)\right] + \sum_{1}^{n} \alpha_i^T e_i \\ \text{subject to} \quad & \sum_{k=1}^{K} \alpha_{i,k} = 0 \quad \forall i \in \{1, \ldots, n\} \\ & \alpha_i^k \leq 0, \text{ if } y_i \neq k \\ & \alpha_i^k \leq C, \text{ if } y_i = k \\ & w_k = \sum_{i=1}^{n} \alpha_i^k x_i \end{aligned}$$

2 Regression

2.1 Linear Regression

After loading the dataset, I have omitted the first two columns, ie ID and Date. Then the first 80% of the dataset was used as training data and the remaining 20% was used as testing data. The labels are the prices of the houses and the features are the remaining columns.

Before applying regression, the data was augmented with a 1 to account for the bias term. The regression was then applied to the training data to obtain w using the following formula:

$$\begin{aligned} w_{LR} &= (X^T X)^{-1} X^T Y \\ w_{RR} &= (X^T X + \lambda I)^{-1} X^T Y \end{aligned}$$

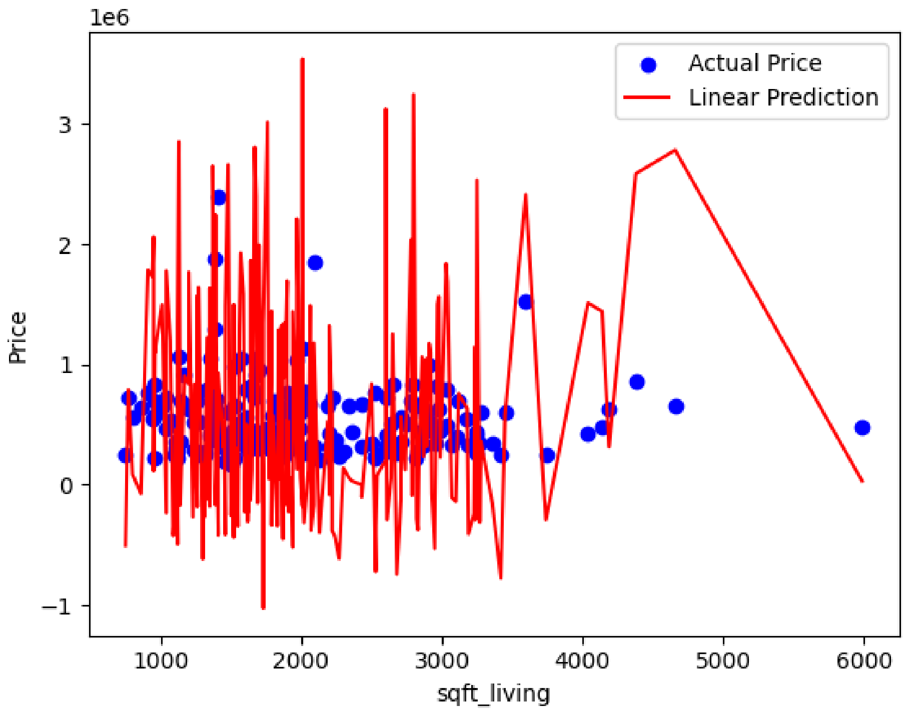
where X is the augmented training data and Y is the true labels.

On solving the regression, the MSE was:

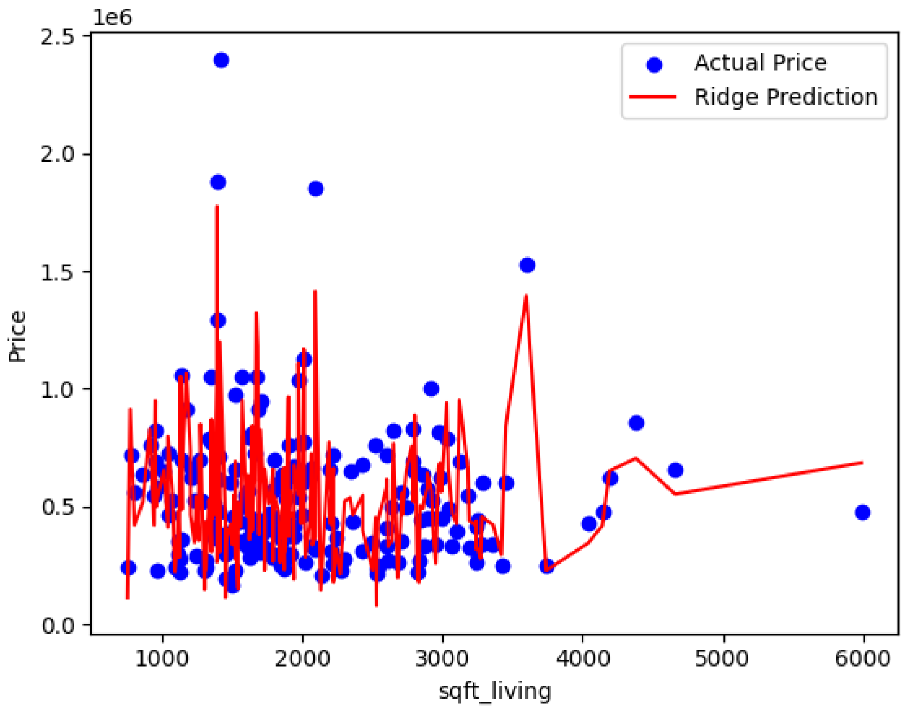
$$\begin{aligned} MSE_{LR} &= 7 \times 10^{11} \\ MSE_{RR} &= 2 \times 10^{10} \end{aligned}$$

Since w is not a 1-D number, we can not directly draw a line with slope w on a 2-D plane. To show the line, we connect all the points predicted by the regression and plot it against the actual values.

The plot for the actual values vs the prediction by the linear regression model is:

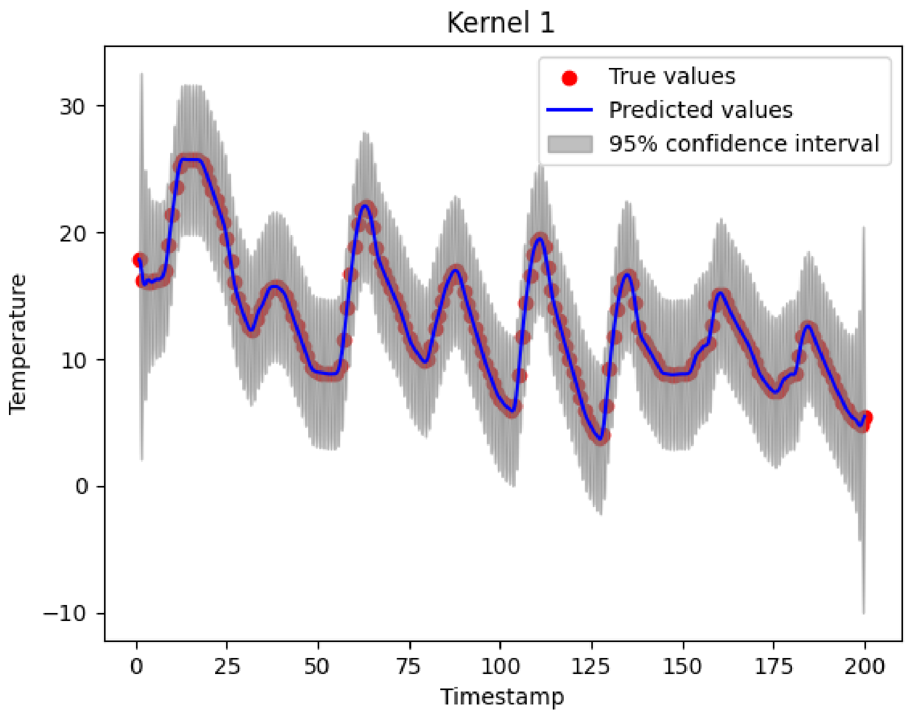


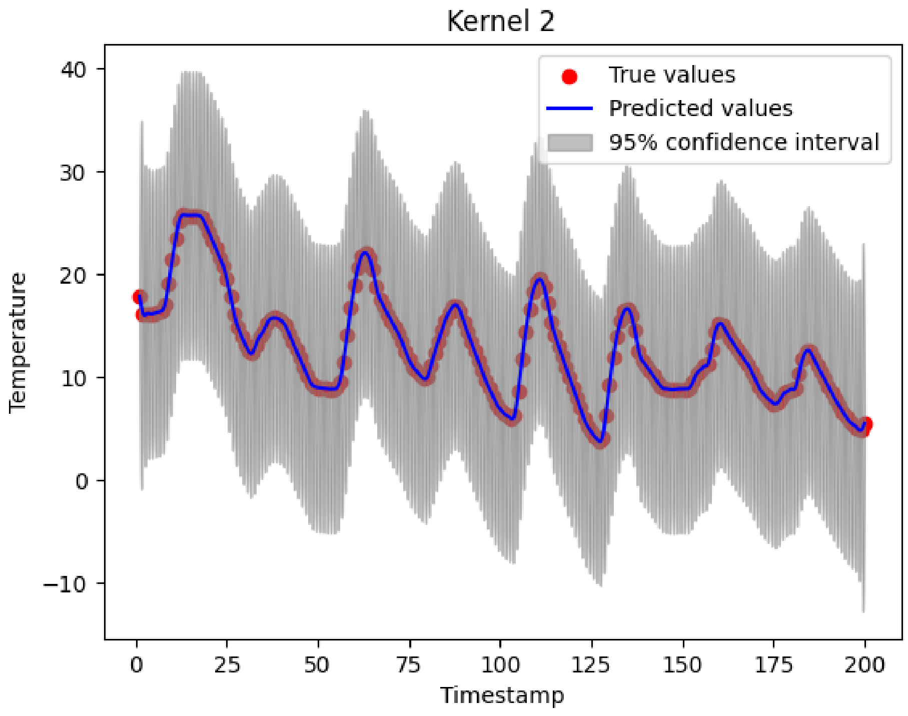
The plot for the actual values vs the prediction by the ridge regression model is:

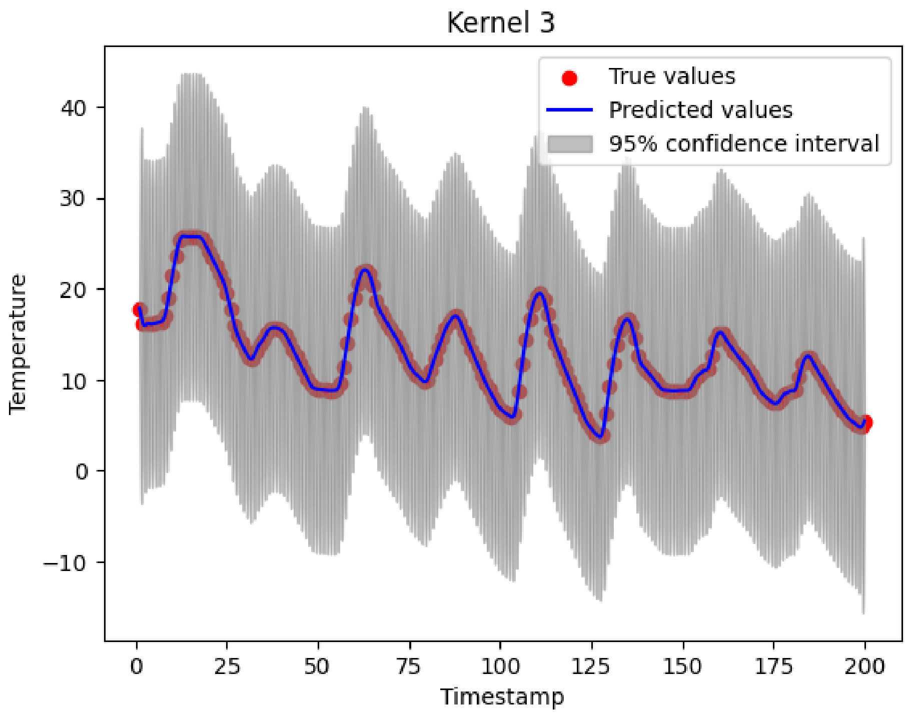


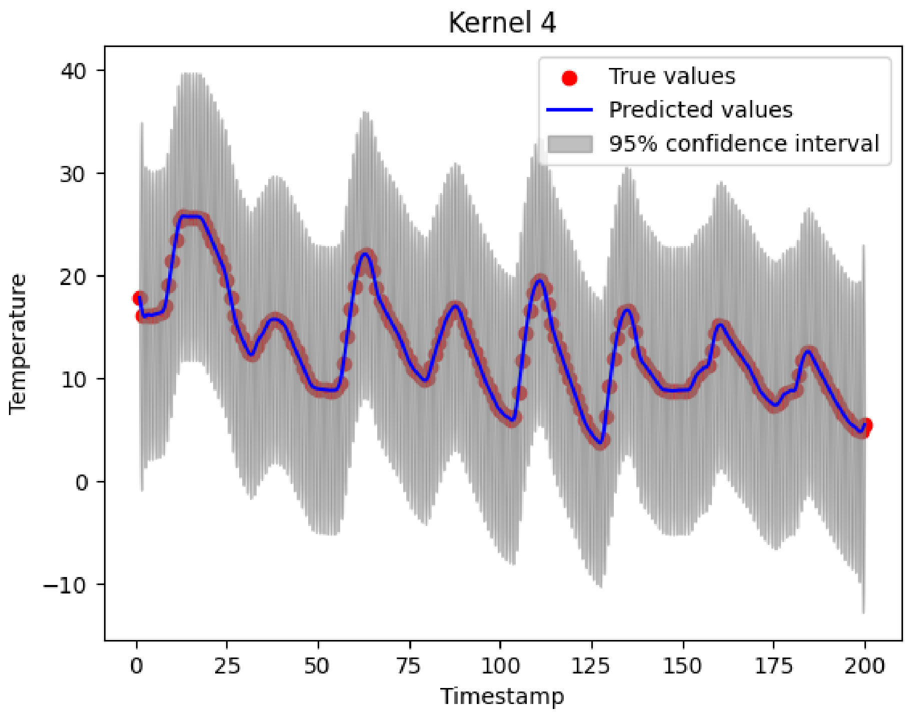
2.2 Gaussian Process Regression

The Gaussian Process Regression was applied to the same dataset with different kernels. The results were as follows:


Kernel 1
True values
Predicted values
95% confidence interval
Temperature
Timestamp
30
20
10
0
−10
0
25
50
75
100
125
150
175
200


Kernel 2
True values
Predicted values
95% confidence interval
Temperature
Timestamp
40
30
20
10
0
−10
0
25
50
75
100
125
150
175
200


Kernel 3
True values
Predicted values
95% confidence interval
Temperature
Timestamp
40
30
20
10
0
−10
0
25
50
75
100
125
150
175
200


Kernel 4
True values
Predicted values
95% confidence interval
Temperature
Timestamp
40
30
20
10
0
−10
0
25
50
75
100
125
150
175
200