UMC 203 Assignment 2

Debanjan Saha
23607

Question 1

1.1

A plot between the misclassification rate and the number of iterations for the perceptron algorithm as defined in Task 1.
Task 1: Perceptron: Run the perceptron algorithm on your data. Report whether it converges, or appears not to. If it doesn't seem to converge, make certain that you are reasonably sure.

Ans. The misclassification rates plot for both train and test data are given in Figure 1. Doing FLD in 1D we can easily visualize why it does not converge. (Figure 2). After using linear SVM and removing the support vectors we get a clearly separable training set. (Figure 3). Similarly we can get a plot with misclassifications removed.

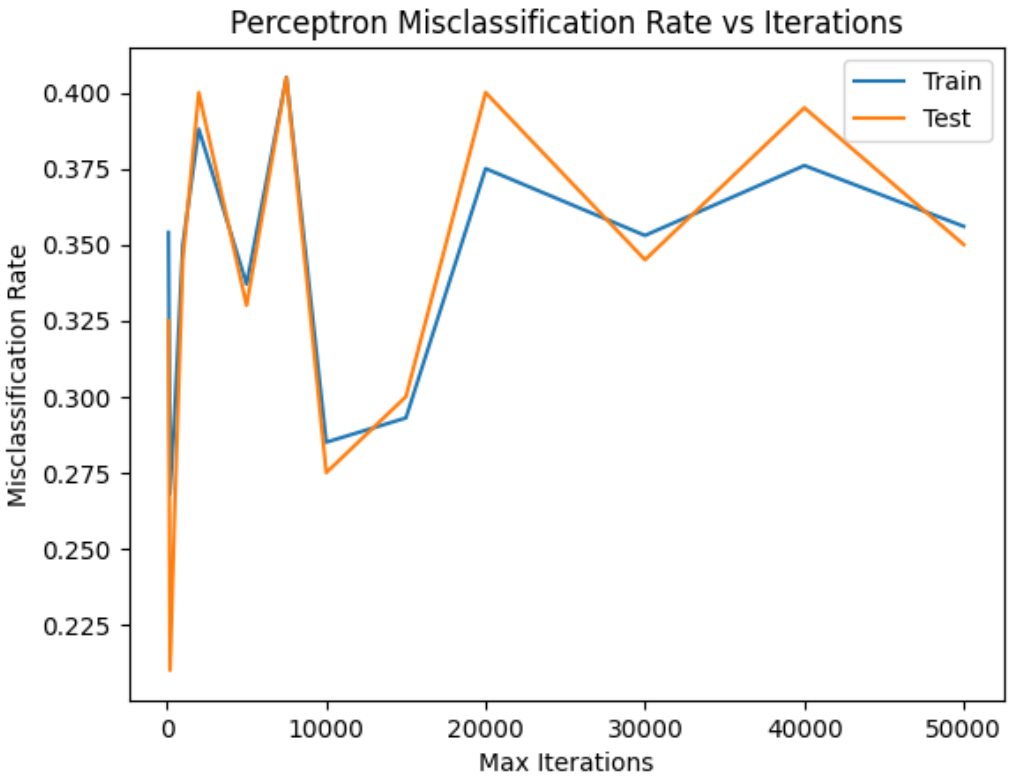


Figure 1: Misclassification rate vs Iterations of perceptron (Q.1.1)

1.2

Which, between the primal and dual, is solved faster for Task 2. Report the times taken for running both and justify any patterns you see.
Task 2: Linear SVM: Construct a slack support vector machine with the linear kernel. Solve both the primal

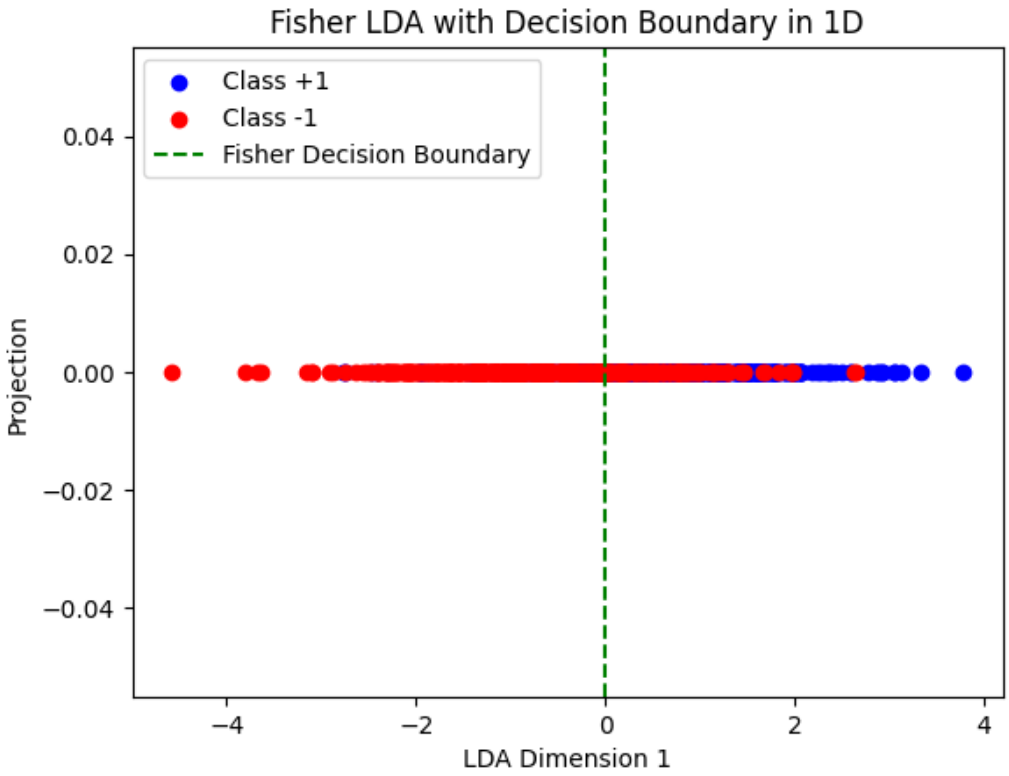


Figure 2: FLD plot of the train data (notice that it overlaps heavily) (Q.1.1)

and dual versions of the SVM quadratic programs using cvxopt. Use the SVM's solution to isolate the sources of non-separability.

Ans. In all the cases the dual is being solved faster. Refer to Figure 4. It seems that the dual is solved faster due to optimized kernel computation and having only one variable to optimize.

1.3

The images that cause non-separability (indices).

Ans. Please refer to attached file "inseparable_23607.csv". Indices of the datapoints causing inseparability for linear SVM in the training set are given in the file.

1.4

The final misclassification rate for the kernelized SVM for Task 3.
Task 3: Kernelized SVM: Repeat the previous construction, but with the Gaussian kernel this time. Choose your hyperparameters such that non-separability is no longer an issue, and your decision boundary is consistent with the training data's labels.

Ans. Choosing $C = 60$ and $\gamma = 0.1$ I am getting number of training set misclassified points = 265 and 40 on test set. So, misclassification rate is 26.5% on training set and 20% on test set.

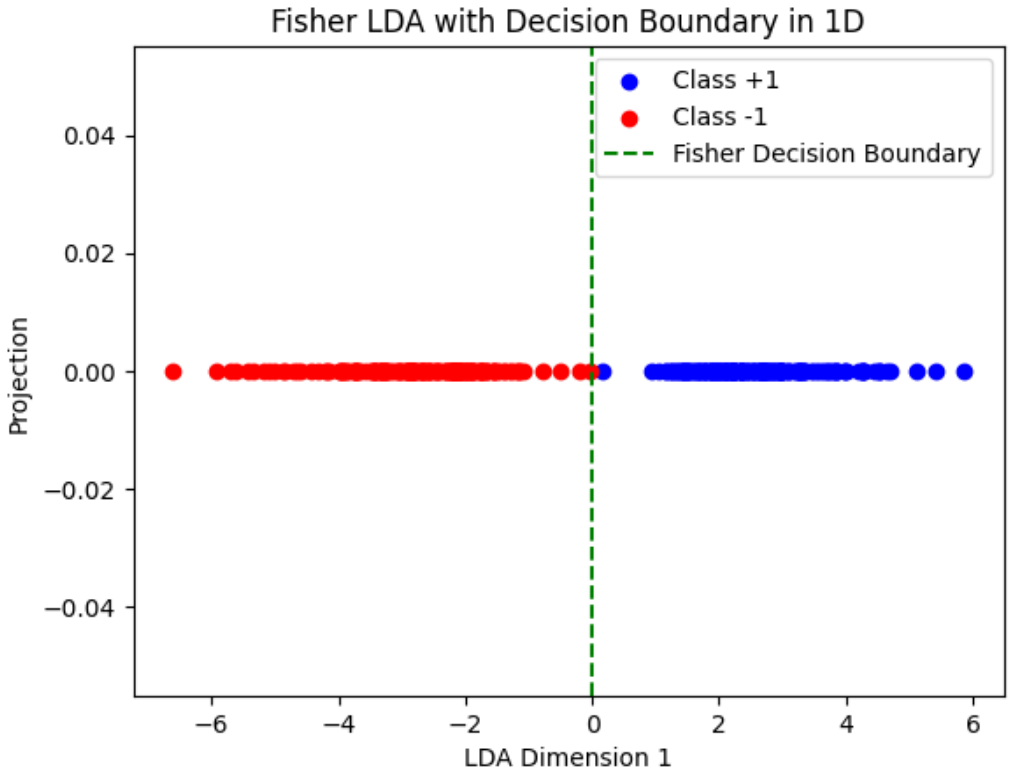


Figure 3: FLD plot of the train data with support vectors removed (Q.1.1)

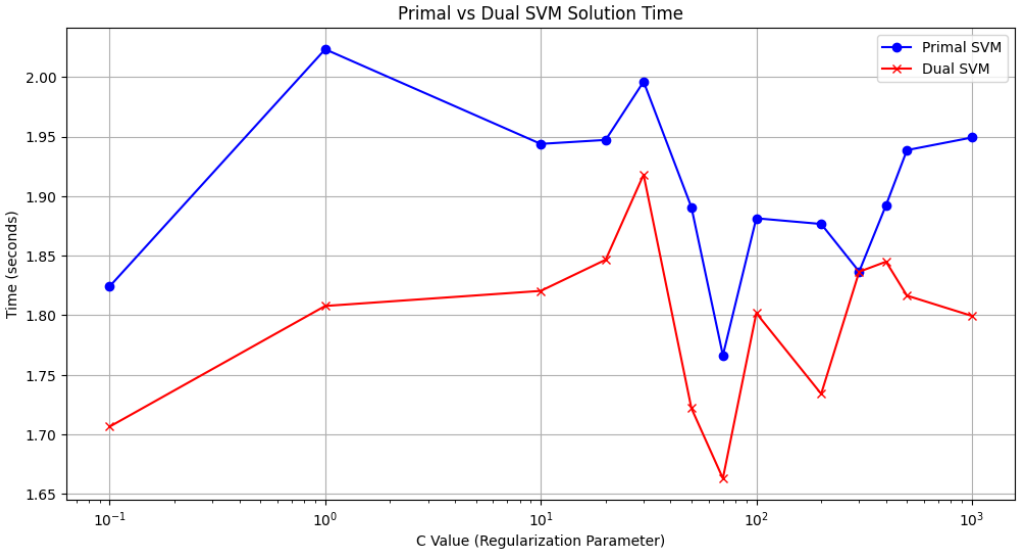


Figure 4: Primal and Dual solution times vs C (Q.1.2)

1.5

A plot between misclassification rate and iterations for the perceptron for Task 4.
Perceptron, again: Retrain the perceptron, after removing the sources of non-separability isolated by the linear SVM. Verify that it converges.

Ans. Please refer to Figure 5. We see that when we remove all support vectors, perceptron converges faster than the case of removing the misclassified points. Although testing, former perceptron (sv removed) gives **40/200** misclassifications whereas later one (misclassifications removed) gives **36/200** misclassifications.

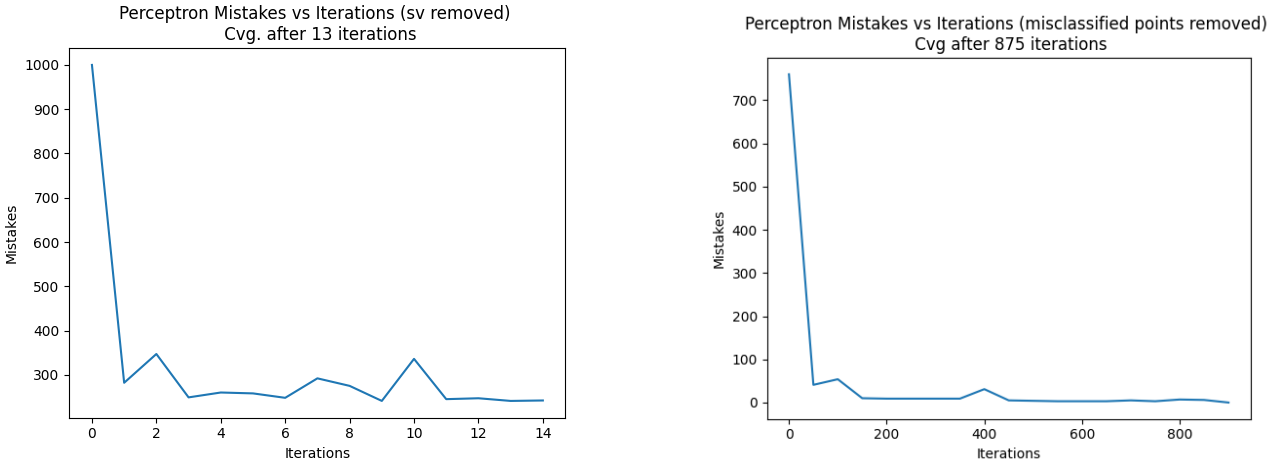


Figure 5: Training Misclassifications vs iterations with non-separabilities removed. (Q.1.5)

Question 2

Tasks:

2.1. Multi Layer Perceptron:

The MLP model gets saved as 'mlp_model.pth'. I have defined test_image_mlp(path, model) taking an image-path and MLP model and returning the predicted label and class-probabilities list.

Example: 'q2_data/3/50.jpg' (Figure 6) gives correct prediction 3 with probability **0.984**.

2.2. CNN:

Similar to part 2.1, model gets saved as cnn_model.pth and testing using the defined function 'test_image_cnn(path, model).

Example: 'q2_data/3/50.jpg' (Figure 6) gives correct prediction 3 with probability **1.000**.

2.3. PCA:

Please refer to attached .py file.

2.4. MLP with PCA:

Similarly did PCA with K=100. Wrote function 'test_image_mlp_pca(path, model, mean, eigenvectors, k)'. Testing for the same image (Figure 6) as above gives class-probability **0.9653**.

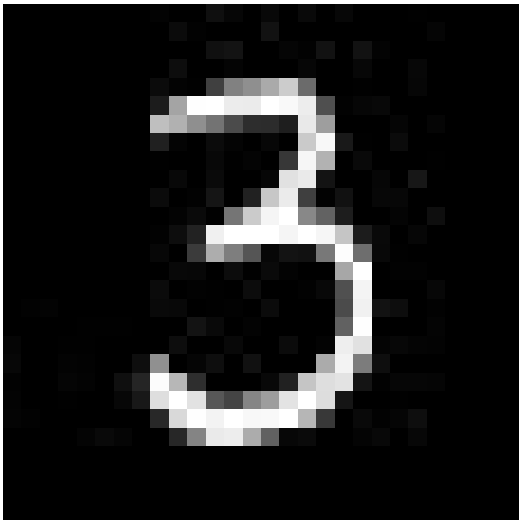


Figure 6: q2_data/3/50.jpg (Q.2)

2.5. Logistic Regression with PCA:

Gives class probability **0.7726**. (For Figure 6)

Deliverables

2.1. Image reconstruction:

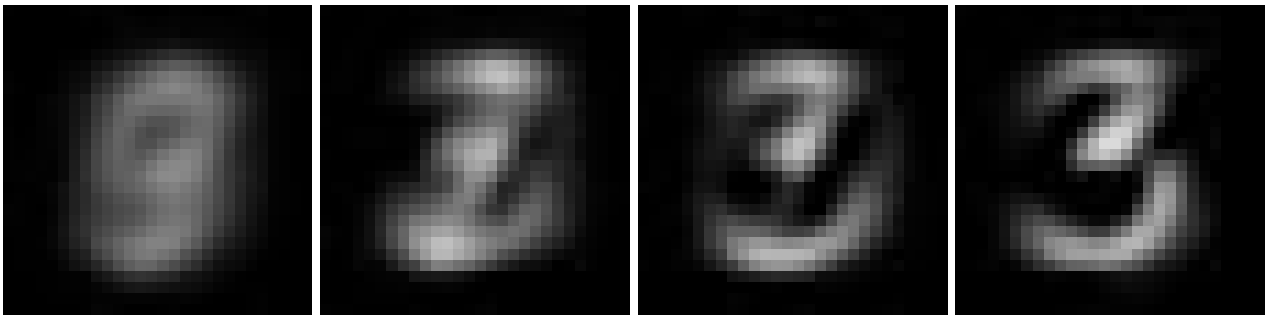
I have taken the same image as Figure 6 and applied PCA for varied K values and reconstructed those. Please refer to figure 7. From the trend we can conclude that as we take more principal components into consideration while reconstructing, the image gets sharper.

2.2. Confusion Matrices of all multi-class classifiers:

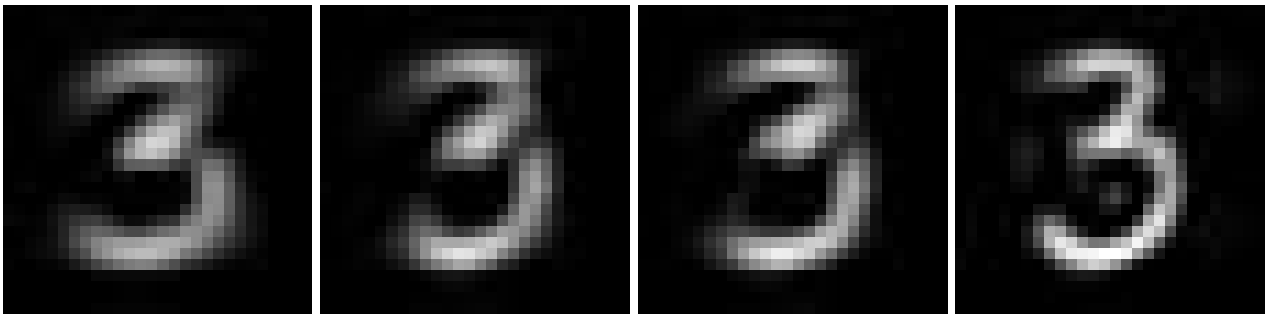
MLP confusion matrix: Please refer to following confusion matrix of test data and Table 1 for metrics.

$$MLP\ CM = \begin{pmatrix} 188 & 0 & 1 & 1 & 0 & 3 & 0 & 0 & 1 & 0 \\ 0 & 211 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 2 & 193 & 5 & 4 & 0 & 1 & 2 & 6 & 0 \\ 0 & 0 & 1 & 193 & 0 & 4 & 0 & 5 & 3 & 3 \\ 0 & 0 & 1 & 0 & 195 & 0 & 1 & 1 & 0 & 10 \\ 1 & 1 & 0 & 7 & 0 & 164 & 2 & 0 & 3 & 3 \\ 3 & 3 & 3 & 0 & 3 & 8 & 178 & 0 & 4 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 161 & 0 & 13 \\ 1 & 2 & 0 & 1 & 1 & 2 & 0 & 0 & 197 & 3 \\ 0 & 0 & 0 & 4 & 3 & 1 & 0 & 1 & 2 & 185 \end{pmatrix}$$

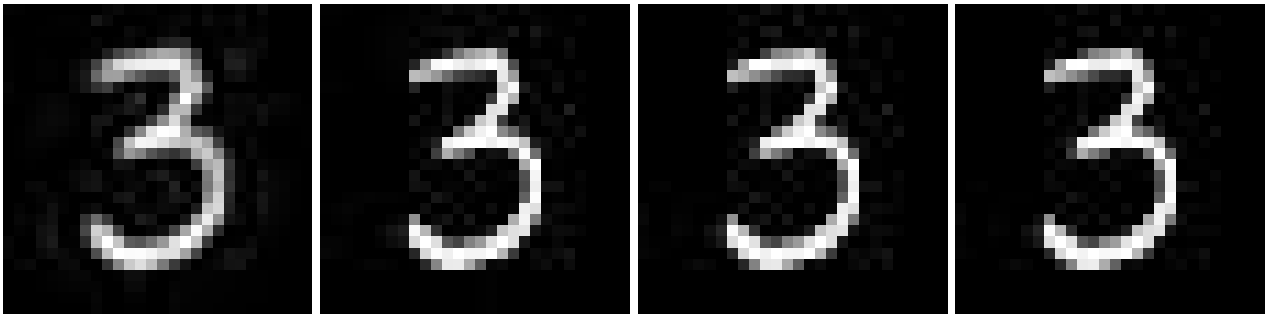
CNN confusion matrix: Please refer to following confusion matrix of test data and Table 2 for metrics.



(a) K=1 (b) K=2 (c) K=5 (d) K=8



(e) K=10 (f) K=15 (g) K=20 (h) K=50



(i) K=100 (j) K=400 (k) K=784 (l) Original

Figure 7: Image reconstruction after PCA (Q.2.1)

Table 1: MLP Classification Metrics per Class (Q.2.2)

Class	Accuracy	Precision	Recall	F1 Score
0	0.9935	0.9641	0.9691	0.9666
1	0.9955	0.9635	0.9953	0.9791
2	0.9855	0.9650	0.8977	0.9301
3	0.9825	0.9104	0.9234	0.9169
4	0.9880	0.9466	0.9375	0.9420
5	0.9825	0.9011	0.9061	0.9036
6	0.9860	0.9780	0.8812	0.9271
7	0.9880	0.9471	0.9148	0.9306
8	0.9855	0.9120	0.9517	0.9314
9	0.9780	0.8486	0.9439	0.8937

$$CNN\ CM = \begin{pmatrix} 204 & 0 & 0 & 0 & 0 & 1 & 3 & 1 & 0 & 0 \\ 0 & 170 & 0 & 1 & 1 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 193 & 0 & 0 & 0 & 0 & 2 & 3 & 0 \\ 1 & 0 & 0 & 203 & 0 & 3 & 0 & 1 & 2 & 3 \\ 0 & 2 & 0 & 0 & 182 & 0 & 2 & 1 & 2 & 1 \\ 0 & 0 & 0 & 1 & 0 & 212 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 201 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 203 & 1 & 2 \\ 1 & 0 & 1 & 2 & 0 & 1 & 1 & 1 & 195 & 1 \\ 0 & 0 & 0 & 2 & 4 & 1 & 0 & 4 & 0 & 176 \end{pmatrix}$$

Table 2: CNN Classification Metrics per Class (Q.2.2)

Class	Accuracy	Precision	Recall	F1 Score
0	0.9965	0.9903	0.9761	0.9831
1	0.9965	0.9884	0.9714	0.9798
2	0.9960	0.9847	0.9747	0.9797
3	0.9920	0.9713	0.9531	0.9621
4	0.9935	0.9733	0.9579	0.9655
5	0.9955	0.9636	0.9953	0.9792
6	0.9955	0.9710	0.9853	0.9781
7	0.9910	0.9398	0.9760	0.9575
8	0.9915	0.9559	0.9606	0.9582
9	0.9910	0.9617	0.9412	0.9514

MLP with PCA confusion matrix: Please refer to following confusion matrix of test data and Table 3 for metrics.

$$MLP\ with\ PCA\ CM = \begin{pmatrix} 204 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 193 & 1 & 0 & 1 & 0 & 0 & 0 & 6 & 0 \\ 2 & 1 & 181 & 0 & 1 & 0 & 3 & 1 & 3 & 0 \\ 0 & 0 & 5 & 187 & 0 & 5 & 0 & 1 & 1 & 2 \\ 0 & 0 & 1 & 0 & 199 & 0 & 3 & 0 & 0 & 6 \\ 1 & 1 & 1 & 1 & 0 & 179 & 0 & 0 & 1 & 2 \\ 1 & 1 & 2 & 0 & 0 & 1 & 206 & 0 & 0 & 0 \\ 0 & 0 & 6 & 1 & 2 & 3 & 0 & 189 & 0 & 3 \\ 3 & 0 & 5 & 1 & 1 & 3 & 4 & 0 & 193 & 2 \\ 0 & 0 & 0 & 5 & 4 & 1 & 0 & 2 & 0 & 163 \end{pmatrix}$$

Logistic Regression with PCA: Please refer to following confusion matrix of test data and Table 4 for metrics.

$$logistic\ reg\ CM = \begin{pmatrix} 199 & 0 & 0 & 1 & 3 & 0 & 2 & 1 & 1 & 1 \\ 0 & 192 & 1 & 0 & 1 & 2 & 1 & 0 & 5 & 0 \\ 4 & 6 & 155 & 4 & 3 & 1 & 7 & 4 & 8 & 0 \\ 1 & 2 & 9 & 167 & 0 & 10 & 2 & 5 & 2 & 3 \\ 0 & 2 & 1 & 0 & 191 & 3 & 4 & 0 & 2 & 6 \\ 9 & 3 & 1 & 10 & 2 & 141 & 6 & 5 & 7 & 2 \\ 4 & 1 & 5 & 0 & 2 & 4 & 195 & 0 & 0 & 0 \\ 0 & 7 & 3 & 2 & 3 & 1 & 0 & 180 & 2 & 6 \\ 4 & 9 & 4 & 8 & 4 & 14 & 1 & 4 & 161 & 3 \\ 0 & 1 & 1 & 7 & 9 & 0 & 0 & 11 & 2 & 144 \end{pmatrix}$$

Table 3: MLP with PCA classification Metrics on Test Set per Class (Q.2.2)

Class	Accuracy	Precision	Recall	F1 Score
0	0.9940	0.9623	0.9808	0.9714
1	0.9940	0.9847	0.9554	0.9698
2	0.9835	0.8916	0.9427	0.9165
3	0.9885	0.9541	0.9303	0.9421
4	0.9905	0.9567	0.9522	0.9544
5	0.9900	0.9323	0.9624	0.9471
6	0.9920	0.9493	0.9763	0.9626
7	0.9905	0.9793	0.9265	0.9521
8	0.9850	0.9461	0.9104	0.9279
9	0.9860	0.9106	0.9314	0.9209

Table 4: Logistic regression with PCA classification metrics per Class (Q.2.2)

Class	Accuracy	Precision	Recall	F1 Score
0	0.9845	0.9005	0.9567	0.9277
1	0.9795	0.8610	0.9505	0.9035
2	0.9690	0.8611	0.8073	0.8333
3	0.9670	0.8392	0.8308	0.8350
4	0.9775	0.8761	0.9139	0.8946
5	0.9600	0.8011	0.7581	0.7790
6	0.9805	0.8945	0.9242	0.9091
7	0.9730	0.8571	0.8824	0.8696
8	0.9600	0.8474	0.7594	0.8010
9	0.9740	0.8727	0.8229	0.8471

Result comparison: Overall the CNN model has the highest accuracy followed by MLP with PCA and then MLP. The Logistic regression with PCA has least accuracy.

2.3. AUC score and ROC curve:

Please refer to Figure 8 for the ROC curve of the classification of one vs. rest (the AUC scores are shown in the legend) and Figure 9 for multiclass classification.

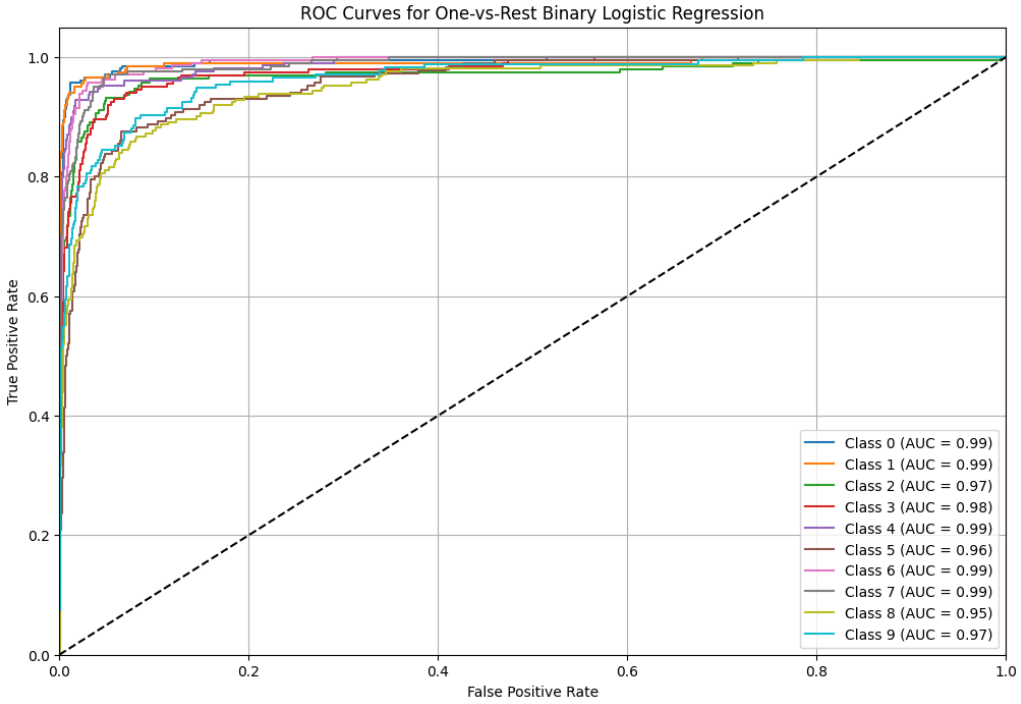


Figure 8: Logistic reg. ROC curve for one vs rest classification (Q.2.3)

Question 3

3.1. Linear Regression:

3.1.1.

Can you do OLS if X does not have full column rank?

Ans. If X doesn't have full column rank then $X \cdot X^T$ will not be invertible, i.e. more than one solution would exist. For inverting $X \cdot X^T$ we can use pseudo-inverse and thus be able to do OLS. In fact, the given D_2 has non-invertible XX^T.

3.1.2.

Report MSE on D_1^{train}. Report w_1^{ols} and w_1^{rr}.

Ans. OLS MSE on D_1 is 2.7685343 and RR($\alpha = 100$) MSE is 1.6587716.
(Please refer to RR MSE vs α curve in Figure 10).

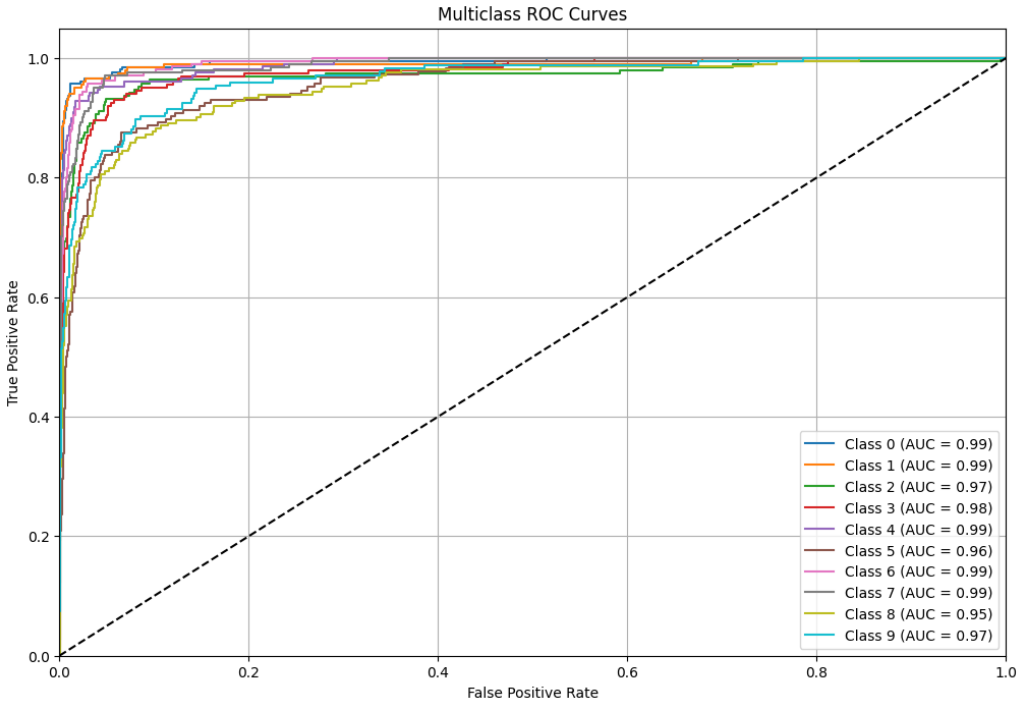


Figure 9: Logistic reg. ROC curve for multiclass classification (Q.2.3)

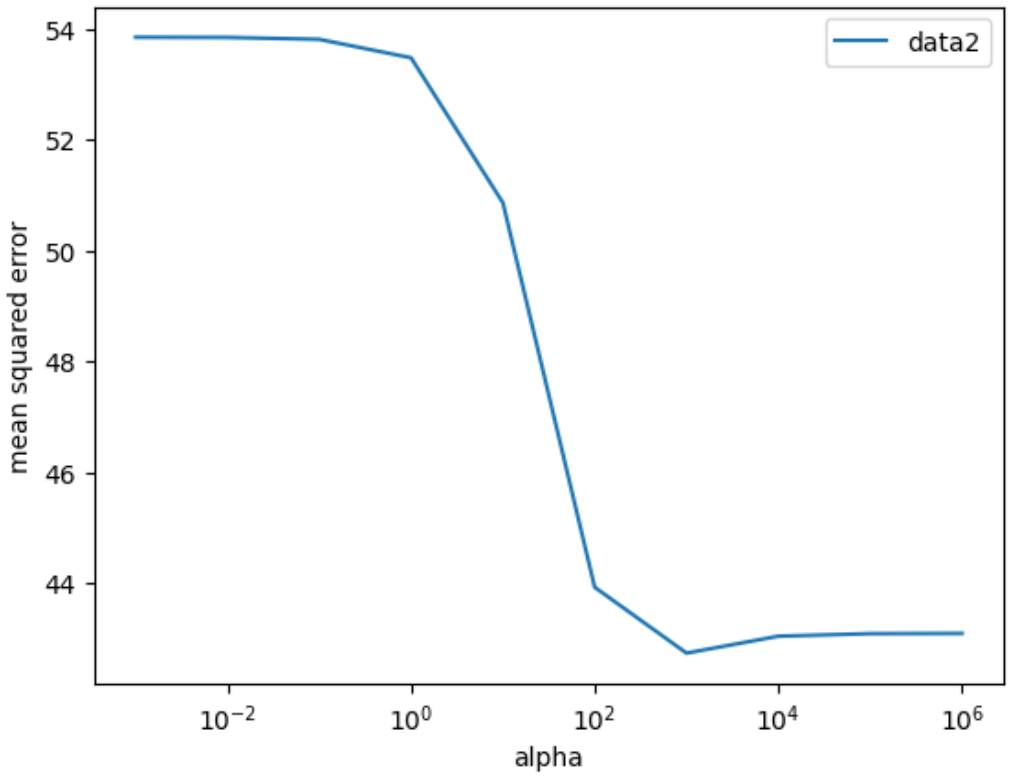


Figure 10: RR MSE vs α (Q.3.1)

$$w_1^{ols} = \begin{bmatrix} 0.35655766 \\ 0.32183454 \\ 0.00351676 \\ 0.8842572 \\ 0.16073379 \\ 0.07412291 \end{bmatrix} \qquad w_1^{rr} = \begin{bmatrix} 0.10344289 \\ 0.04432345 \\ -0.02761292 \\ 0.13022108 \\ 0.0014239 \\ 0.00560629 \end{bmatrix}$$

3.1.3.

Report MSE on D_2^{train}. Attach w_2^{ols} and w_2^{rr} as csv files.

Ans. OLS MSE on D_2 is 53.852755 and RR($\alpha = 100$) MSE is 43.931703. For w_2^{ols} and w_2^{rr} please refer to attached 'w_ols_23607.csv' and 'w_rr_23607.csv' respectively.

3.2. SVR

Train SVR for $t \in \{7, 30, 90\}$ and $\gamma \in \{1, 0.1, 0.01, 0.001\}$

Ans. My stock was AEP. Linear SVR predictions are in Figures 11, 12, 13. Gaussian SVR predictions are in Figures 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25.

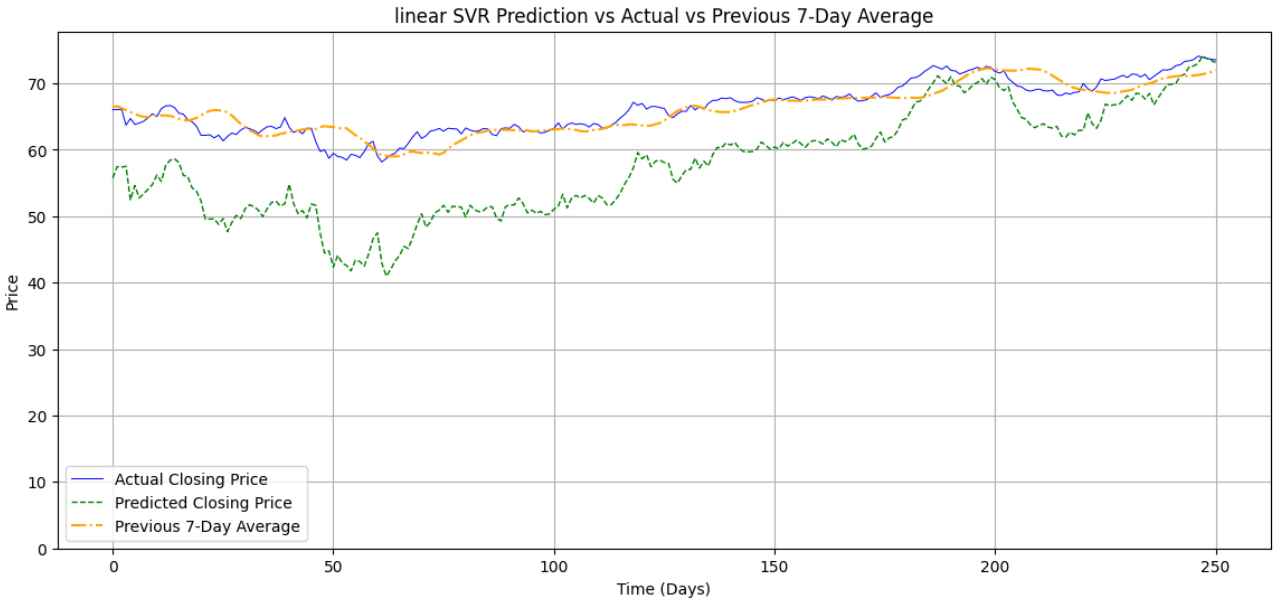


Figure 11: Linear SVR, t=7

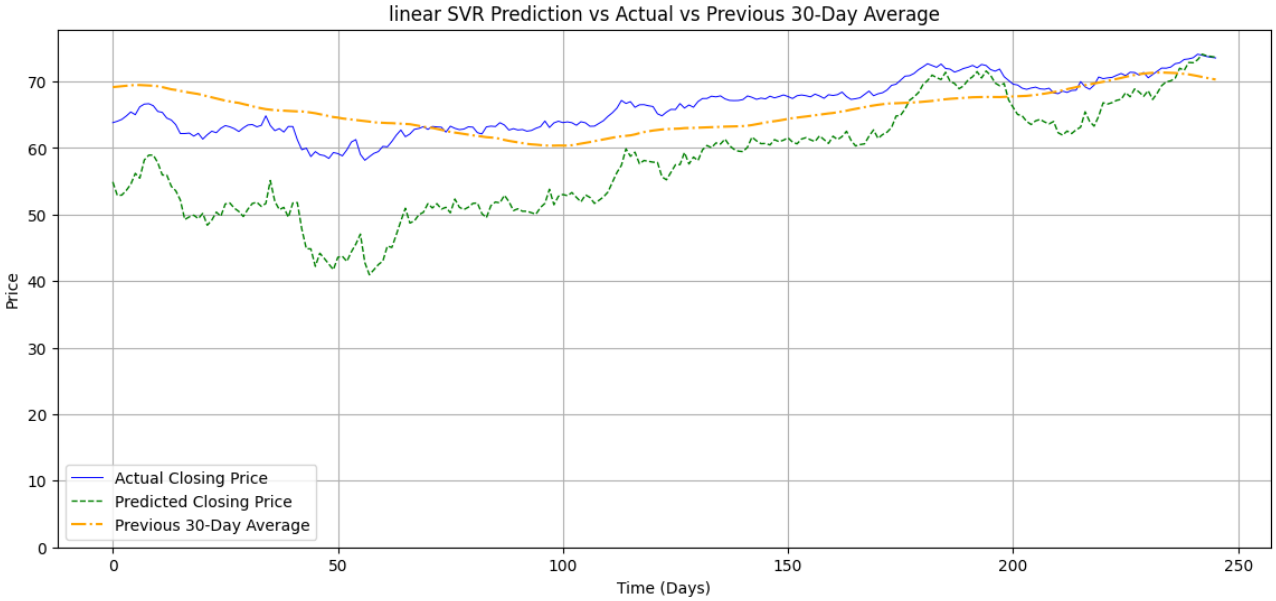


Figure 12: Linear SVR, t=30

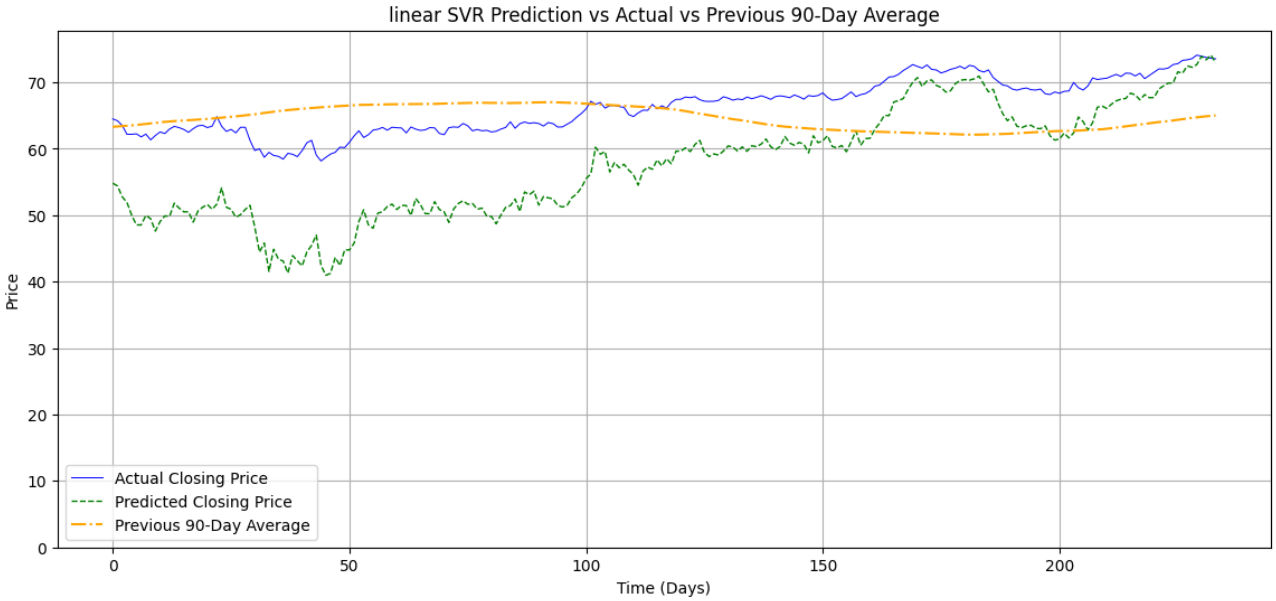


Figure 13: Linear SVR, t=90

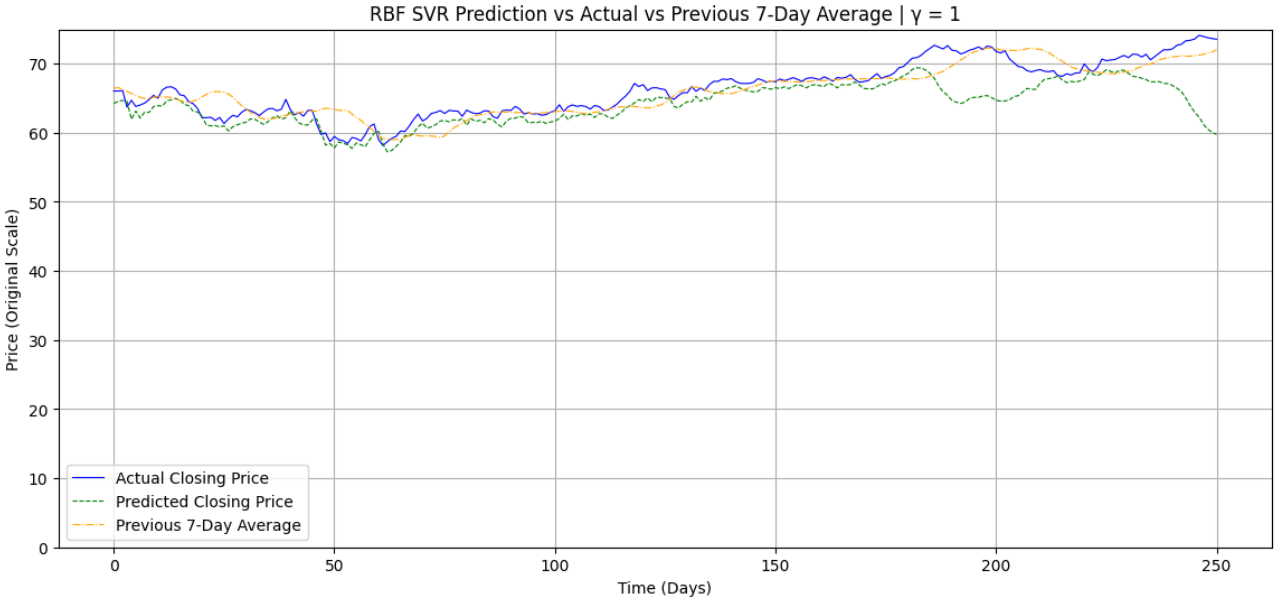


Figure 14: Gaussian SVR, t=7, $\gamma = 1$

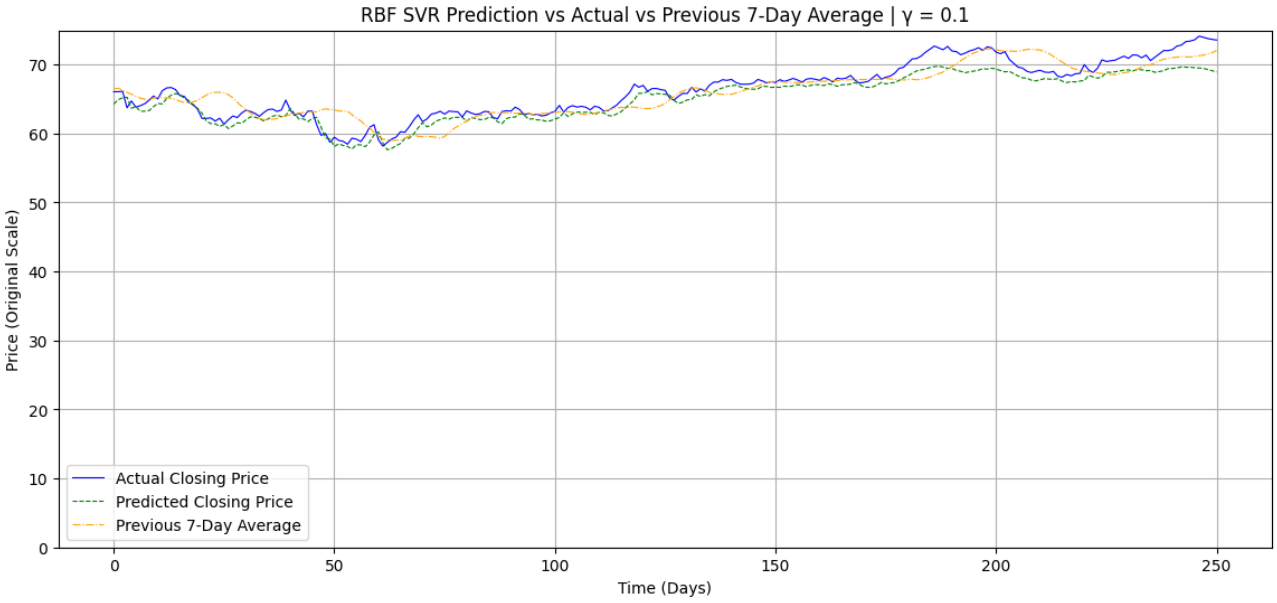


Figure 15: Gaussian SVR, t=7, $\gamma = 0.1$

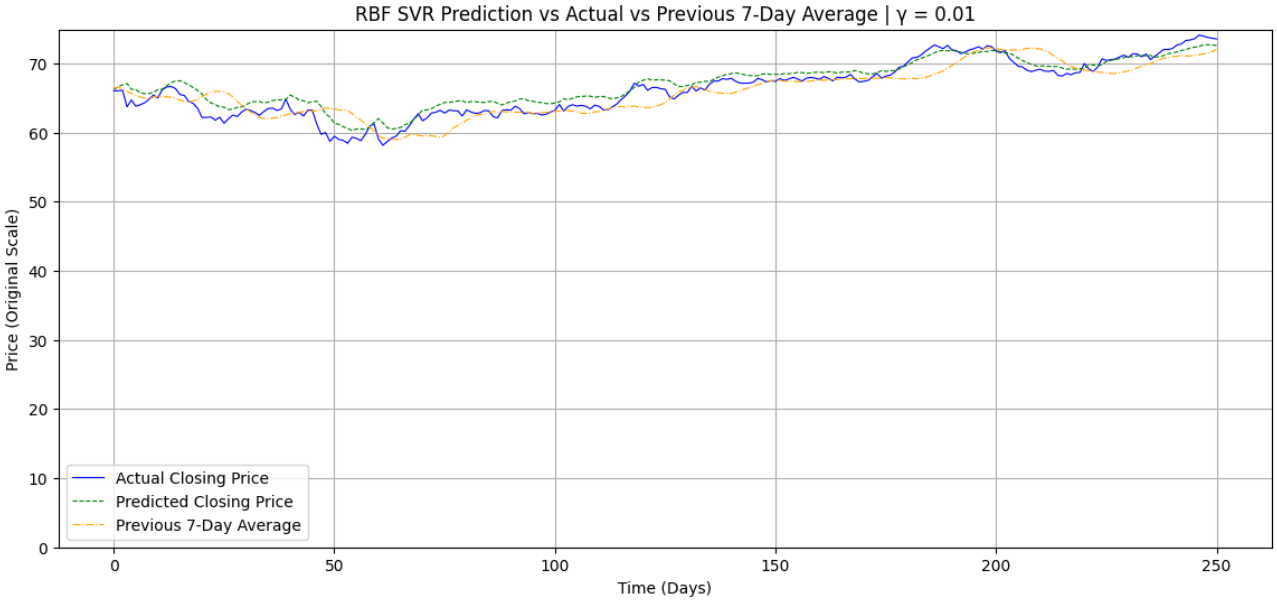


Figure 16: Gaussian SVR, t=7, $\gamma = 0.01$

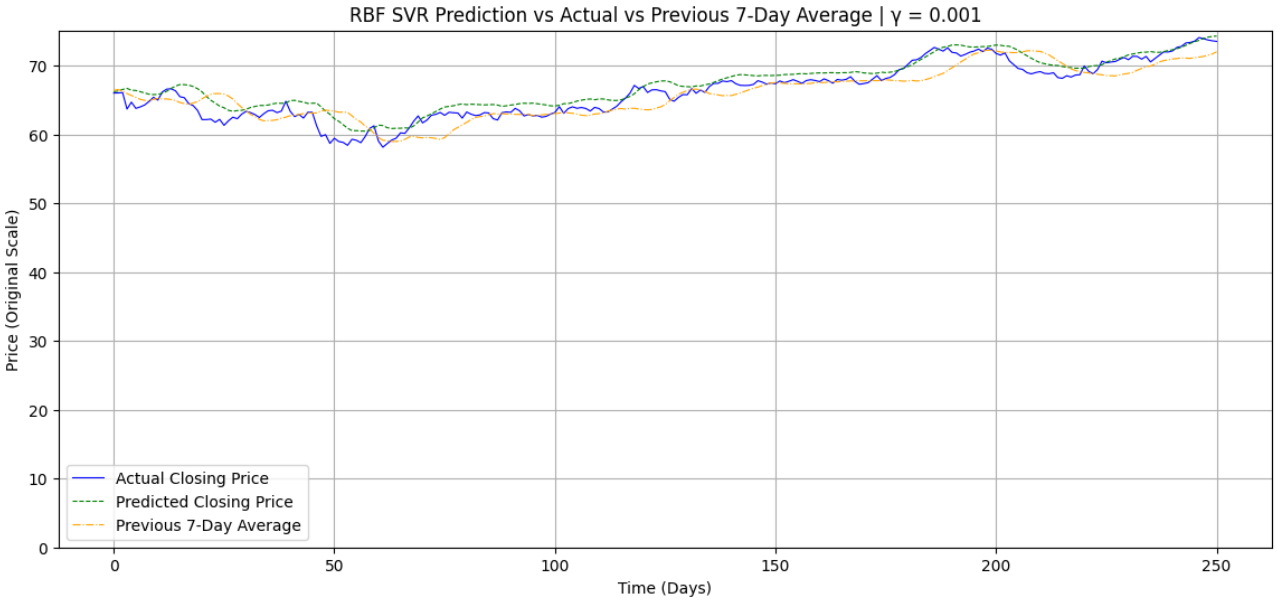


Figure 17: Gaussian SVR, t=7, $\gamma = 0.001$

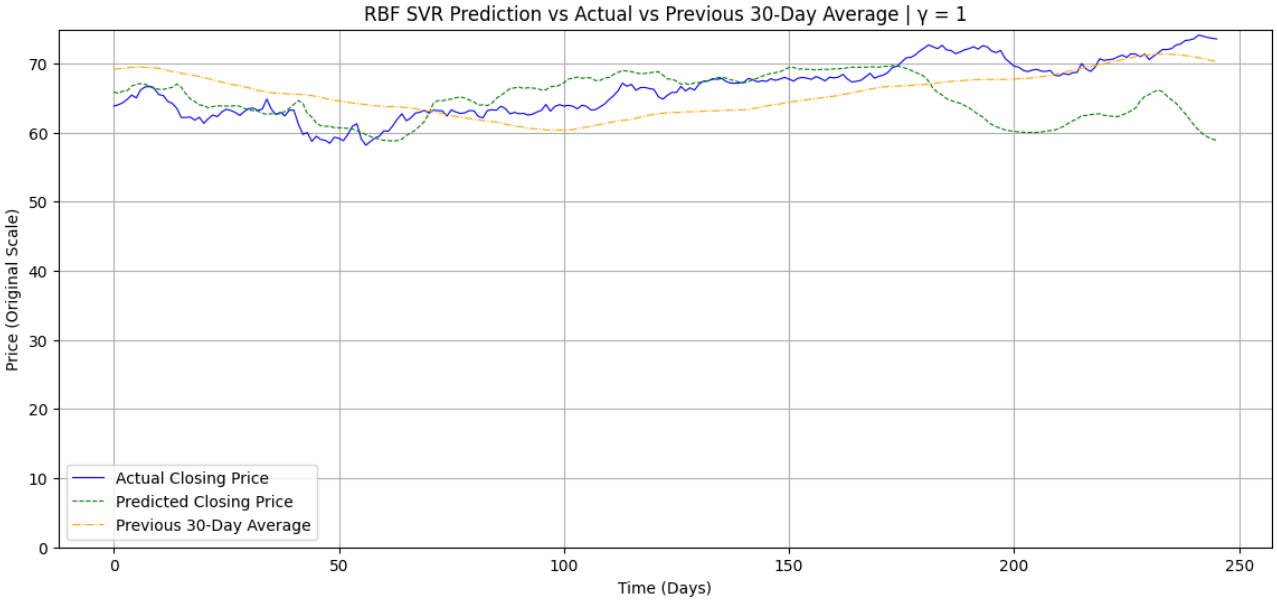


Figure 18: Gaussian SVR, t=30, $\gamma = 1$

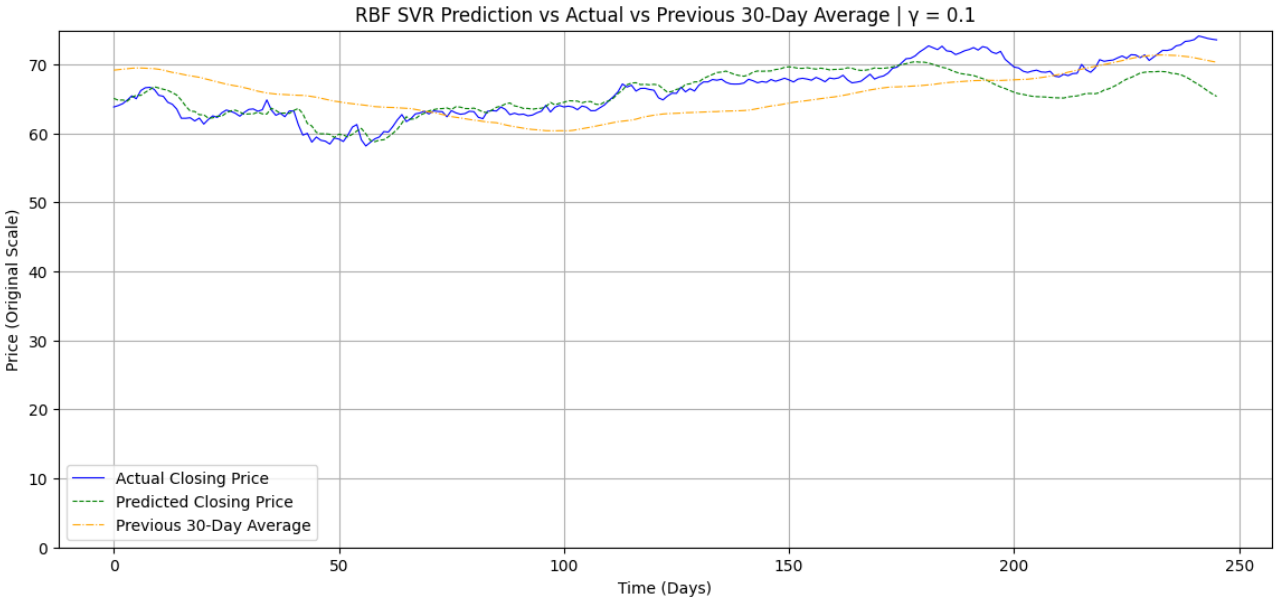


Figure 19: Gaussian SVR, t=30, $\gamma = 0.1$

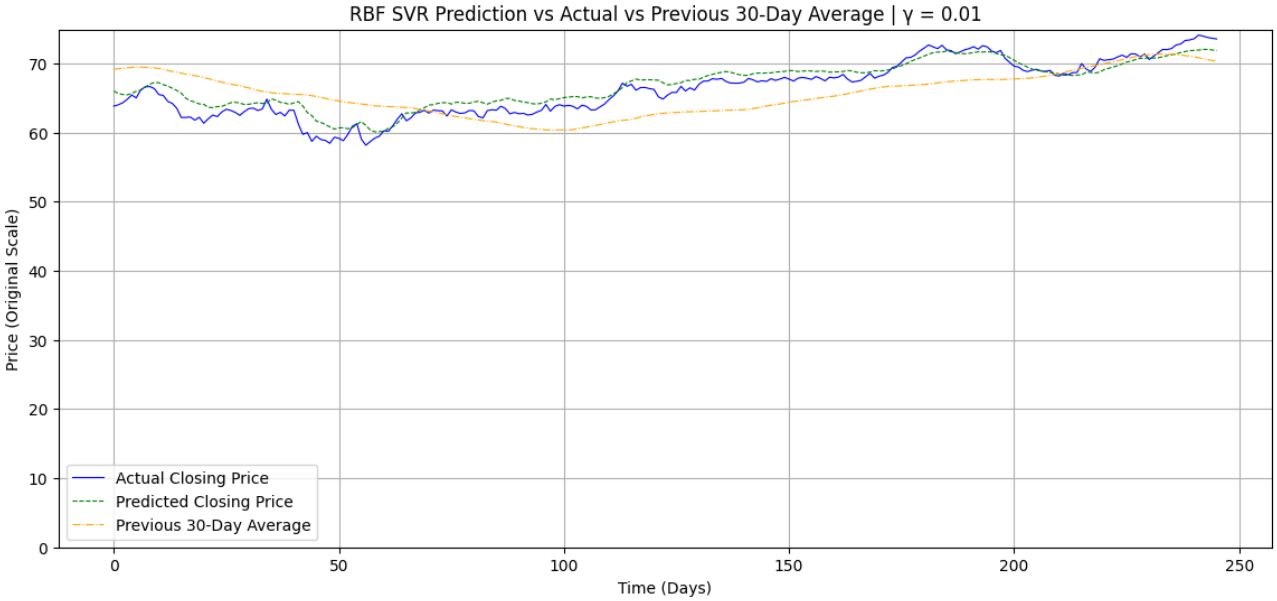


Figure 20: Gaussian SVR, t=30, $\gamma = 0.01$

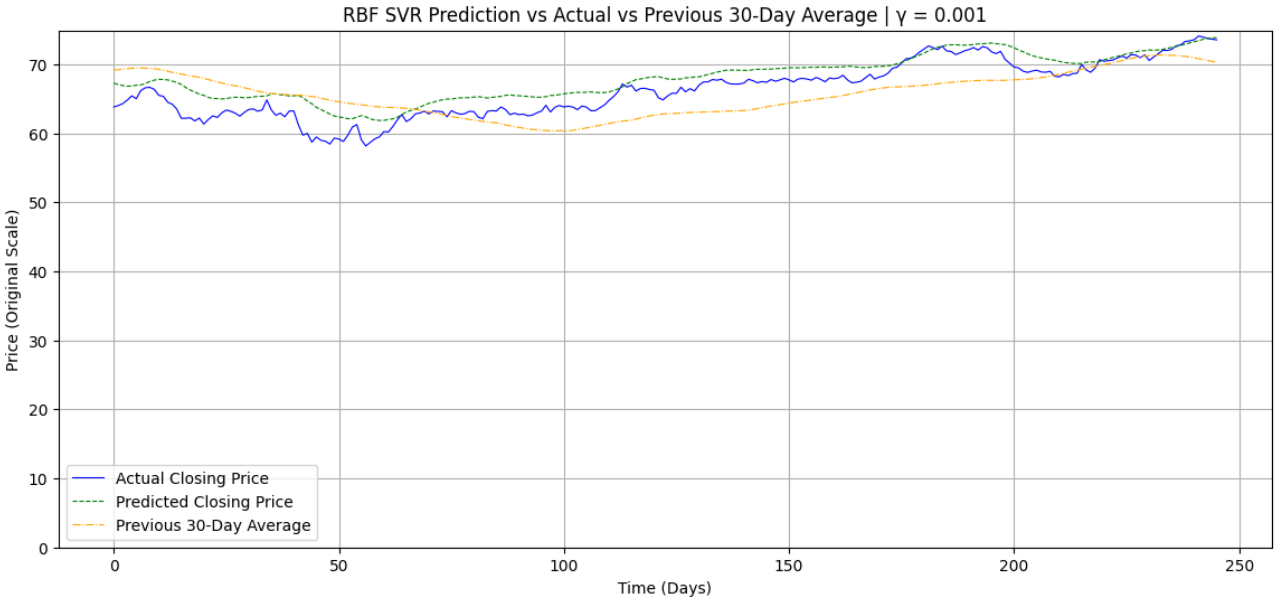


Figure 21: Gaussian SVR, t=30, $\gamma = 0.001$

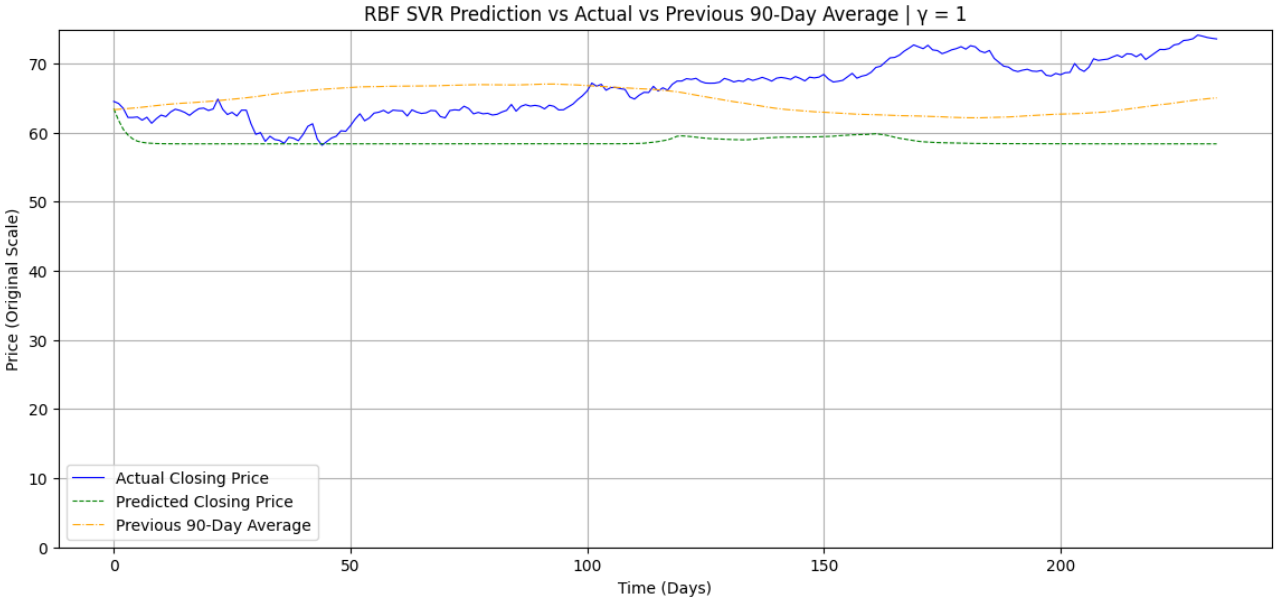


Figure 22: Gaussian SVR, t=90, $\gamma = 1$

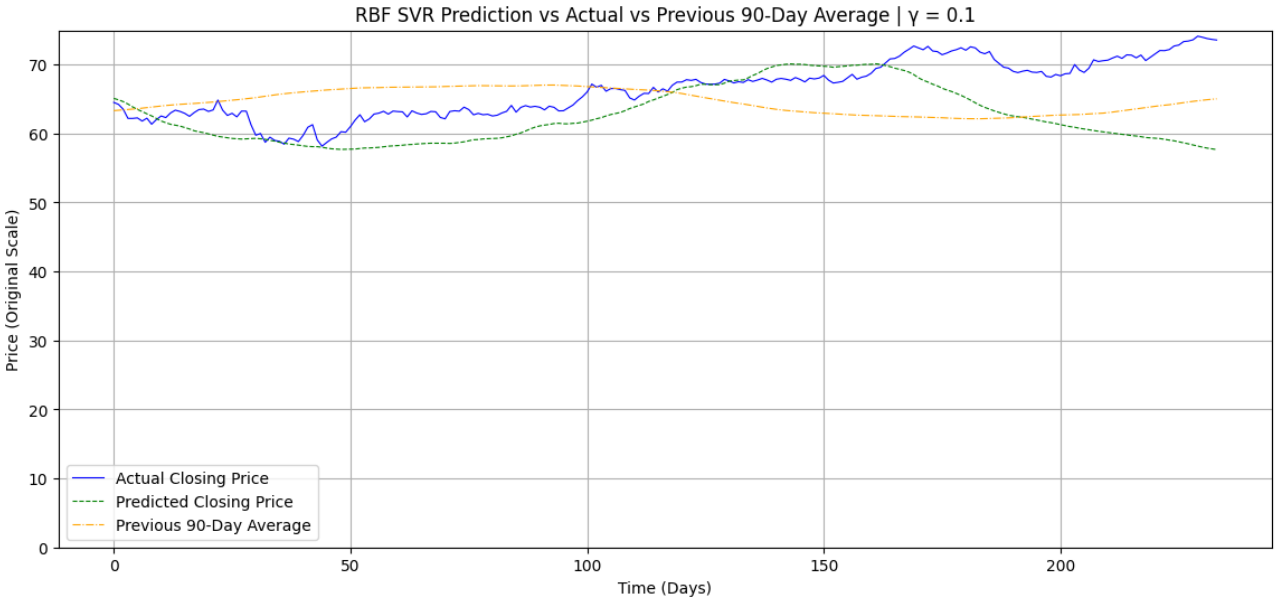


Figure 23: Gaussian SVR, t=90, $\gamma = 0.1$

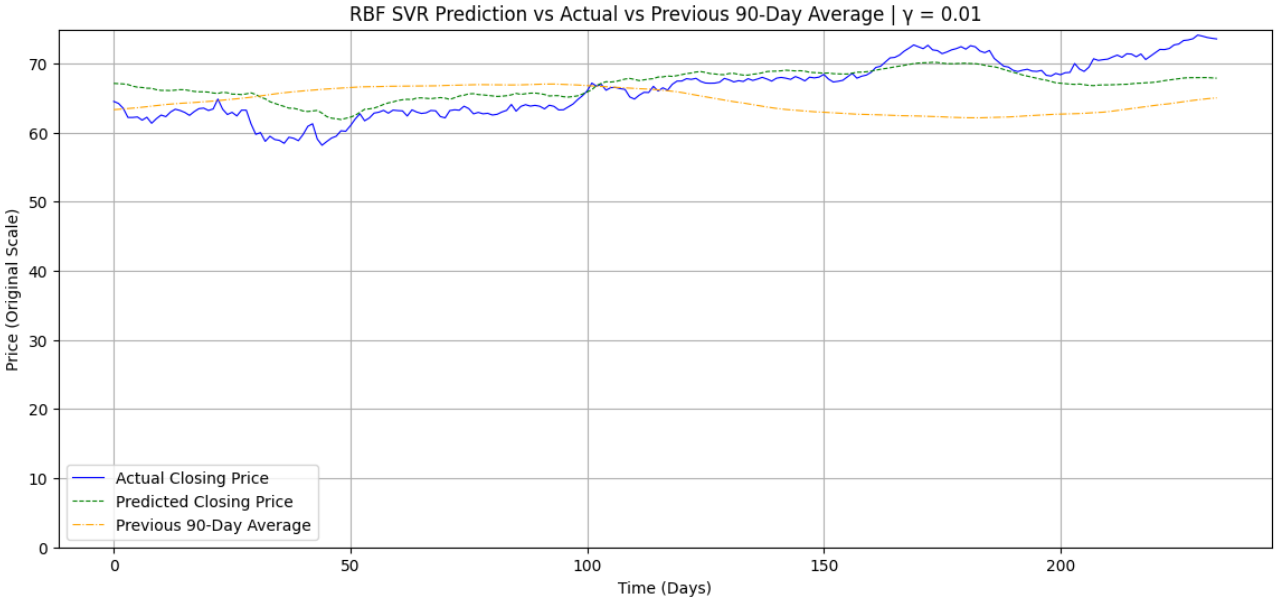


Figure 24: Gaussian SVR, t=90, $\gamma = 0.01$

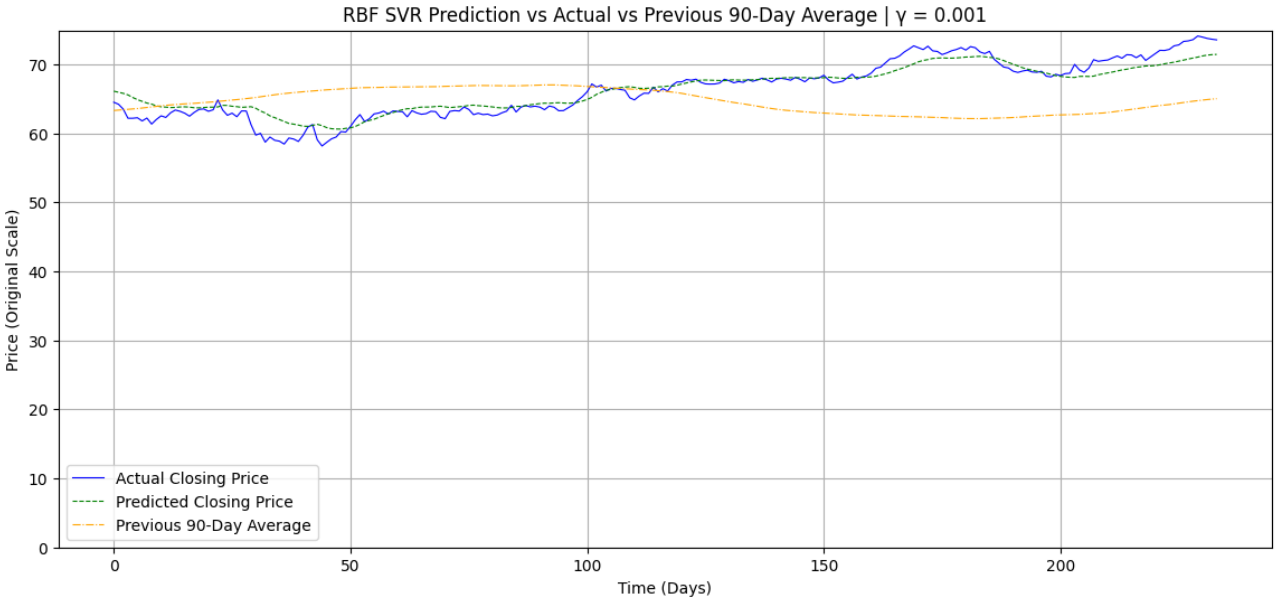


Figure 25: Gaussian SVR, t=90, $\gamma = 0.001$