28462000 - Bourez Nicolas
38502000 - Demaret Dimitri
77102000 - Lhernould Benoit

# LINMA2472 - Homework 3 - Part II - Group 9

## 1 Q1 Testing time with default parameters

| Method | Testing Time (seconds) | Accuracy |
|---|---|---|
| Linear SVM | 53.018 | 0.907 |
| Kernel SVM with Gaussian Kernel | 171.756 | 0.956 |
| Linear SVM with Random Fourier Features | 26.296 | 0.918 |

Table 1: Comparison of SVM classification times and accuracies

The kernel trick, as used in the Kernel SVM with a Gaussian Kernel, operates implicitly in a high-dimensional space, which is computationally intensive. This is reflected in the longer testing time of 171.756 seconds. In contrast, the Linear SVM operates in the original feature space and is less computationally demanding, which is consistent with its shorter testing time of 53.018 seconds.

The Linear SVM with Random Fourier Features represents a balance between these two approaches. It explicitly approximates the high-dimensional space, thereby allowing for a linear classification in this approximated space. The result of 26.296 seconds for this method shows it's significantly faster than both the Gaussian Kernel SVM and the regular Linear SVM. This confirms the theoretical expectation that the RFF method would lead to reduced computational complexity ($\mathcal{O}(D+n)$), where $D$ is the dimension of the approximated space) while still maintaining relatively high accuracy.

## 2 Q2 Varying Parameters

### 2.1 Accuracy of the linear SVM with RFF

By increasing the dimensionality of the approximated kernel, we are expanding the space in which the SVM searches for the optimal decision boundary. A higher dimensional makes the data more separable, allowing the linear SVM to find a better decision boundary that separates the classes more accurately. We can see that from the 500th dimension, the accuracy is not improving as much as before. So, we don't need to choose excessively high dimensions because the gain is not so significant and might lead to overfitting.

As we tend to a perfect approximation of the kernel function when $D \to \infty$ :

$$k(\mathbf{x}, \mathbf{y}) \approx \langle z(\mathbf{x}), z(\mathbf{y}) \rangle = \frac{1}{D} \sum_{m=1}^{D} \sqrt{2} \cos\left(\omega_m^T \mathbf{x} + b_m\right) \sqrt{2} \cos\left(\omega_m^T \mathbf{x} + b_m\right)$$

We can conclude that the results of the graph are coherent with the theory. As we are bound with the number of data, we will never reach a perfect approximation but we can still be very close.

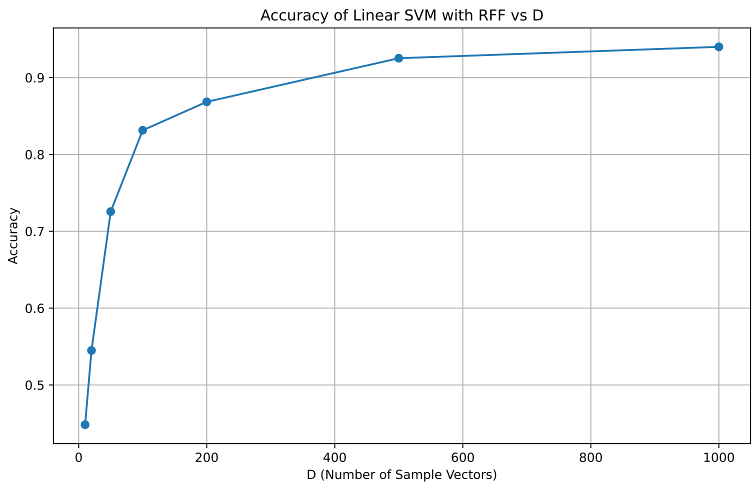

Figure 1: Accuracy of Linear SVM with RFF vs D

## 2.2 Time taken to create the RFF

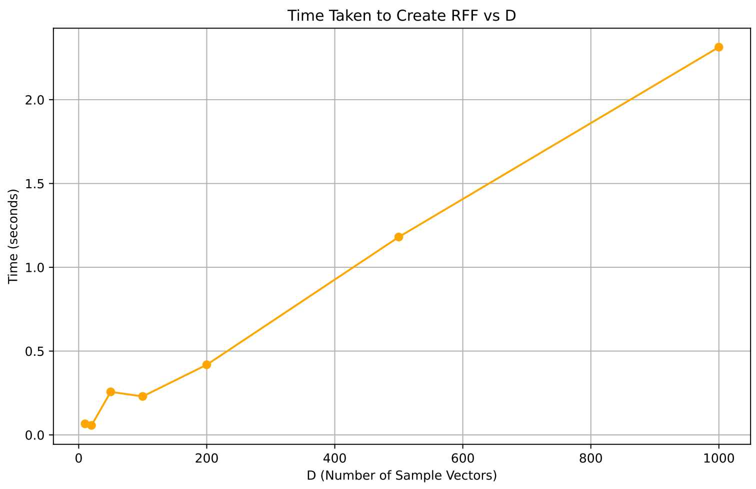

Figure 2: Time Taken to Create RFF vs $D$

The time to create RFF increases linearly with $D$, reflecting the direct relationship between the dimensionality of the transformed feature space and the computational effort required to construct it. Taking high dimensions is not terrible in terms of complexity, indeed a linear relationship is one of the best relations we can find.

## 2.3 Time taken to train the SVM with RFF

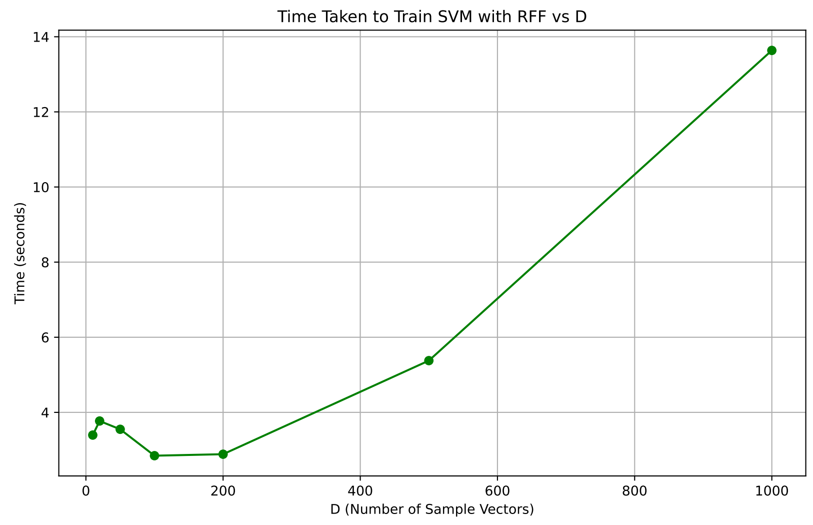

Figure 3: Accuracy of Linear SVM with RFF vs $D$

Training time remains relatively low for smaller $D$ but grows significantly as $D$ approaches 1000, indicating a linear increase in complexity with dimensionality.

## 2.4 Time taken by the classifier to classify the testing instances

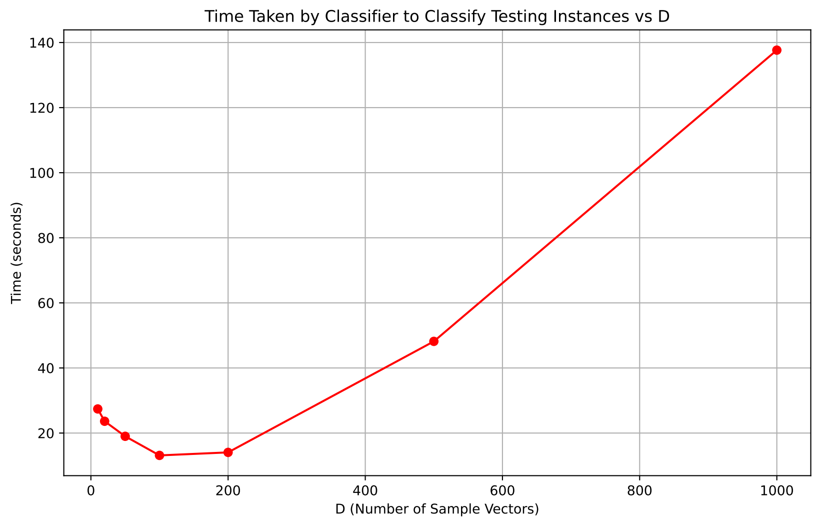

Figure 4: Time Taken by Classifier to Classify Testing Instances vs $D$

The classification time decreases initially, then rises sharply with increasing $D$. This suggests that while initial increases in D may help the model to learn more efficiently, beyond a certain point, the additional complexity outweighs the benefits, leading to longer classification times.

## 2.5 Conclusion

These results demonstrate the trade-off between dimensionality and performance in RFF. Initially, higher dimensions can improve accuracy, but beyond a certain point, the cost in terms of computation time becomes significant. With RFF, we can still work with relatively high dimensions without increasing the computation time drastically. Understanding this balance is key to leveraging RFF effectively in SVMs.