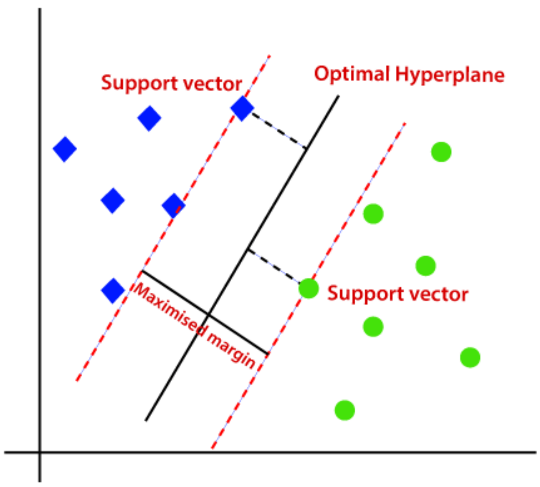
Exploring the Impact of Different Kernels in SVM

1. Support Vector Machine

Though it may also be utilised in regression situations, Support Vector Machine (SVM) is a strong supervised learning method mostly employed in classification task. Finding the ideal hyperplane that divides data points of many classes in a high-dimensional feature space is SVM's central goal. This hyperplane guarantees the best possible separation between the many classes by acting as a decision boundary separating the data. SVM seeks to maximise the margin, which is the distance between the nearest data points of every class to the hyperplane, hence accomplishing this. We call these nearest points support vectors. Maximising the margin will help the classifier to generalise better to unprocessed data, hence lowering the overfitting risk. Since they are the fundamental data points affecting the hyperplane's location and orientation, support vectors are important to SVM.

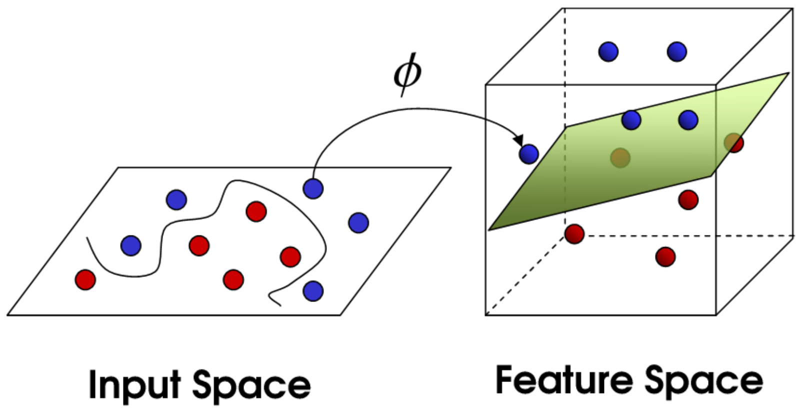
SVM uses the kernel approach in situations when the data is not linearly separable. SVM may transfer the data into a higher-dimensional space where a linear hyperplane can be found to distinguish the classes by utilising kernel functions such polynomial, radial basis function (RBF), or sigmoid. For many real-world applications, SVM is a flexible and strong classification technique because of its capacity to manage non-linear decision limits.



When the data is not linearly separable by means of translating the input data into a higher-dimensional space using a kernel function, SVM is very successful. The kernel trick is this mechanism that lets SVM generate non-linear decision bounds. Common kernels include Radial Basis Functions (RBF), linear, polynomial, and widely used in fields like image recognition, text classification, and bioinformatics, SVM is well-known for its robustness—especially in high-dimensional spaces—and simplicity.

2. Kernel Trick

The kernel trick is a method used in Support Vector Machines (SVM) to handle non-linearly separable data. Instead of explicitly mapping the input data into a higher-dimensional space, the kernel function computes the dot product between data points in that higher-dimensional space without performing the transformation. This allows SVM to find complex decision boundaries while avoiding the computational cost of high-dimensional mapping. Common kernel functions include the linear, polynomial, and radial basis function (RBF) kernels, each suitable for different types of data relationships, making SVM highly flexible and effective for a wide range of classification problems.



3. Types of Kernels

By converting input into a higher-dimensional space, the kernel function is fundamental in Support Vector Machines (SVM) in establishing the decision boundary. Commonly utilised multiple kinds of kernels for categorisation activities are:

Linear Kernel: Usually employed when the data is linearly separable, the simplest and most efficient kernel available in Support Vector Machines (SVM) is linear kernel. Calculating the dot product between two data points in the original feature space drives it. The linear kernel in SVM seeks the ideal hyperplane to optimally divide data points of many classes. A decision boundary maximising the margin between the nearest data points—known as support vectors—from every class defines the hyperplane. This maximised margin guarantees that the classifier extends effectively to unprocessed data.

The linear kernel is very efficient when the data is already well-separated by a linear boundary as it minimises pointless computation and generates rapid model training with excellent performance.

Polynomial Kernel: In Support Vector Machines (SVM), the polynomial kernel is used when the data shows non-linear correlations, especially in cases where the decision boundary follows a polynomial form. Unlike the linear kernel, this one transfers the input data into a higher-dimensional space therefore enabling SVM to identify more intricate decision boundaries. Increasing the polynomial degree lets the model identify more complex patterns in the data, therefore translating it into a space where a linear decision boundary may be drawn—even for non-linear situations. The SVM then seeks to find in this higher-dimensional space the hyperplane that maximises the margin separating the many classes. For datasets where the link between features and class labels is complicated and non-linear, this kernel offers a flexible method of categorisation.

Radial Basis Function (RBF) Kernel: Because of its adaptability in managing difficult, non-linear decision boundaries, the Radial Basis Function (RBF) kernel is among the most often used ones in Support Vector Machines (SVM). It works by turning two data points—measuring their similarity—into a higher-dimensional space. With larger values of gamma, the decision border becomes more sensitive to local data points, hence this function finds the degree of effect a data point has on the decision boundary.

By evaluating the similarity between all pairs of data points, the RBF kernel may provide somewhat complex decision boundaries in classification, therefore allowing SVM to classify data that is not linearly separable. Changing gamma helps the complexity of the decision boundary to be precisely regulated, therefore enabling the model to capture complex patterns in the data and enhance classification performance on challenging datasets.

Sigmoid Kernel: Often seen in neural networks, the sigmoid function's mathematical form is the basis of the sigmoid kernel in Support Vector Machines (SVM). Like other non-linear kernels, the sigmoid kernel converts the data into a higher-dimensional space; yet, its decision boundary works like the activation function of a neural network, enabling it to simulate non-linear interactions between data points. Though it may efficiently categorise data with non-linear boundaries, in many real-world applications the sigmoid kernel performs less effectively than the Radial Basis Function (RBF) kernel. This is so because the sigmoid kernel's decision limits might be less smooth and more prone to overfitting while its sensitivity to the choice of parameters is higher. Consequently, while less often utilised in reality than the RBF kernel, the sigmoid kernel might nevertheless be valuable in some situations when neural network-like behaviour is sought for.

Every kernel offers advantages; the decision will rely on the specific data and the type of problem.

4. Effect of Different Kernels

4.1. Linear Kernel

```
svc_linear = SVC(kernel='linear')
svc_linear.fit(x_train, y_train)
y_pred_linear = svc_linear.predict(x_test)
accuracy_linear = accuracy_score(y_test, y_pred_linear)
print(f'Accuracy with Linear Kernel: {accuracy_linear * 100:.2f}%')
```

With a linear kernel, 90.32% shows that the Support Vector Machine (SVM) is performing satisfactorily on the dataset. This great precision implies that the data is either nearly linearly separable or linearly separable, hence the linear kernel is a good option. The accuracy, recall, and f1-score statistics highlight yet another aspect of the model's performance. While the recall of 0.80 implies that 20% of class 0 cases were misclassified, the accuracy of 1.00 indicates that all expected positive examples were accurately classified. The recall of 1.00 for class 1 shows flawless recognition of class 1 events; nonetheless, the accuracy of 0.84 indicates some misclassification of class 1 as class 0. The weighted average (0.90) and macro average (0.90) F1 scores show a balanced performance across both courses generally. Though the accuracy is really high, the differences in precision and recall across the classes might indicate some space for development, particularly for class 1, which would benefit from another kernel or more hyperparameter tweaking.

```
Classification Report for Linear Kernel:
              precision    recall  f1-score   support

           0       1.00      0.80      0.89        15
           1       0.84      1.00      0.91        16

    accuracy                           0.90        31
   macro avg       0.92      0.90      0.90        31
weighted avg       0.92      0.90      0.90        31
```

4.2. Polynomial Kernel

```
svc_poly = SVC(kernel='poly')
svc_poly.fit(x_train, y_train)
y_pred_poly = svc_poly.predict(x_test)
accuracy_poly = accuracy_score(y_test, y_pred_poly)
print(f'Accuracy with Polynomial Kernel: {accuracy_poly * 100:.2f}%')
```

With the Polynomial kernel, 90.32% suggests that the SVM model performs similarly to the linear kernel, indicating that the data may not need a complicated non-linear decision boundary. Although the Polynomial kernel is made to manage more complex connections, its performance is similar to the linear kernel, suggesting that the data could still be essentially separable in the original feature space. Precision stays perfect at 1.00 for class 0 according the classification report; recall for this class is 0.80, suggesting a 20% misclassification rate. By contrast, for class 1 the recall is flawless (1.00) but the accuracy is 0.84, suggesting some misclassification of class 1 as class 0. With a macro average and weighted average F1 score of 0.90, one finds a balanced performance. Although the Polynomial kernel adds flexibility, overall, the findings imply that the data may not require such complexity as the linear kernel produces almost comparable accuracy.

Classification Report for Polynomial Kernel:

	precision	recall	f1-score	support
0	1.00	0.80	0.89	15
1	0.84	1.00	0.91	16
accuracy			0.90	31
macro avg	0.92	0.90	0.90	31
weighted avg	0.92	0.90	0.90	31

4.3. RBF Kernel

```
svc_rbf = SVC(kernel='rbf')
svc_rbf.fit(x_train, y_train)
y_pred_rbf = svc_rbf.predict(x_test)
accuracy_rbf = accuracy_score(y_test, y_pred_rbf)
print(f'Accuracy with RBF Kernel: {accuracy_rbf * 100:.2f}%')
```

With the RBF kernel, the accuracy of 93.55% shows a development above both the linear and polynomial kernels, thereby stressing its efficiency in capturing non-linear correlations in the data. With a precision of 1.00 and a recall of 0.87, the accuracy, F1-score for class 0 indicates a modest amount of misclassified class 0 events. Although there is a small misclassification rate, the accuracy of class 1 is 0.89 and the recall is perfect (1.00), indicating that most class 1 cases were correctly diagnosed. The weighted average (0.94) and macro average (0.93) F1 scores show a well-balanced model that routinely succeeds across both classes. Higher accuracy and more balanced precision and recall follow from the RBF kernel's apparent better capture of the complicated decision boundary between the classes than from the polynomial kernel. This implies that, especially with regard to complicated, non-linear decision boundaries, the RBF kernel is better appropriate for this particular classification problem.

```
Classification Report for RBF Kernel:
              precision    recall  f1-score   support

           0       1.00      0.87      0.93        15
           1       0.89      1.00      0.94        16

    accuracy                           0.94        31
   macro avg       0.94      0.93      0.93        31
weighted avg       0.94      0.94      0.94        31
```

4.4. Sigmoid Kernel

```
svc_sigmoid = SVC(kernel='sigmoid')
svc_sigmoid.fit(x_train, y_train)
y_pred_sigmoid = svc_sigmoid.predict(x_test)
accuracy_sigmoid = accuracy_score(y_test, y_pred_sigmoid)
print(f'Accuracy with Sigmoid Kernel: {accuracy_sigmoid * 100:.2f}%')
```

Among the kernels examined, the accuracy of 87.10% with the sigmoid kernel is the lowest; this suggests that the sigmoid kernel may not be appropriate for this specific classification job. With 1.00, the precision for class 0 indicates ideal prediction for occurrences of class 0. The recall for class 0, however, is 0.73, indicating that 27% of class 0 events were misclassified. The recall for class 1 is perfect (1.00), although the accuracy is lower at 0.80, meaning some occurrences of class 1 were mistakenly projected as class 0. From weighted average F1 scores of 0.87, the macro average indicates that the model suffers from uneven performance across the two classes. Though theoretically able to represent non-linear decision boundaries, the sigmoid kernel generally performs worse in real-world applications than other kernels such as RBF. This suggests that the sigmoid kernel's capacity to capture the decision boundary is less successful for this dataset, therefore reducing the accuracy generally.

```
Classification Report for Sigmoid Kernel:
              precision    recall  f1-score   support

           0       1.00      0.73      0.85        15
           1       0.80      1.00      0.89        16

    accuracy                           0.87        31
   macro avg       0.90      0.87      0.87        31
weighted avg       0.90      0.87      0.87        31
```

5. Conclusion

The study of different kernels in Support Vector Machines (SVM) reveals unique performance characteristics for every one of them. Though minor imbalances in precision and recall point to potential for development, the linear kernel attained an amazing accuracy of 90.32%, suggesting its fit for linearly separable data. Although it gave flexibility for non-linear boundaries, the polynomial kernel delivered identical results to the linear kernel, suggesting that the dataset may not need such complexity. With an accuracy of 93.55%, the RBF kernel beat the other kernels showing its efficiency in managing non-linear connections and generating balanced precision and recall. On the other hand, the sigmoid kernel had the lowest accuracy at 87.10%, thereby stressing its difficulties in efficiently drawing the decision boundary for this dataset. Though the sigmoid kernel was less successful in this instance, generally the RBF kernel had the highest performance, especially for datasets with complex, non-linear connections.

References

Cárcamo, J., Cuevas, A. and Rodríguez, L.-A. (2024) 'A uniform kernel trick for high and infinite-dimensional two-sample problems', Journal of Multivariate Analysis, 202, p. 105317. doi:10.1016/j.jmva.2024.105317.

Kreßel, U. (1998) 'Pairwise classification and Support Vector Machines', Advances in Kernel Methods [Preprint]. doi:10.7551/mitpress/1130.003.0020.

Lin, Y. (2002) Data Mining and Knowledge Discovery, 6(3), pp. 259–275. doi:10.1023/a:1015469627679.

Pavlidis, P., Wapinski, I. and Noble, W.S. (2004) 'Support Vector Machine Classification on the web', Bioinformatics, 20(4), pp. 586–587. doi:10.1093/bioinformatics/btg461.

Sun, Q. and DeJong, G. (2005) 'Explanation-augmented SVM', Proceedings of the 22nd international conference on Machine learning -ICML '05, pp. 864–871. doi:10.1145/1102351.1102460.

Vieira, C.P. and Digiampietri, L.A. (2020) 'A study about explainable articial intelligence: Using decision tree to explain SVM', Revista Brasileira de Computação Aplicada, 12(1), pp. 113–121. doi:10.5335/rbca.v12i1.10247.