Session 9
Support Vector Machines

PSYC 560

Heungsun Hwang

Support Vector Machine

- The Support Vector Machine (SVM) is an approach for classification, which was developed in the computer science community in the 1990s and has been popular since then.

- SVM has been shown to perform well in many settings and is often considered one of the best “out of the box” classifiers.
  - Successfully applied to various problems including character recognition and text classification, speech and speaker verification, face detection, junk mail classification, credit rating analysis, and cancer and diabetes classification (Attewell et al. 2015; Byun & Lee 2002).

Support Vector Machine

- SVM is a generalization of simple and intuitive classifiers, called the maximal margin classifier and the support vector classifier.

- More specifically, SVM is an extension of the support vector classifier (SVC), an extension of the maximal margin classifier, to accommodate non-linear class boundaries.
  - Both the maximal margin classifier and support vector classifier can deal with linear class boundaries only.

Support Vector Machine

- People often loosely refer to the maximal margin classifier, the support vector classifier, and the support vector machine as “support vector machines”.

- To avoid confusion, we will carefully distinguish among these three methods.

Maximal Margin Classifier

- Suppose that we have N training observations on P predictors/features (X’s) and the binary response y (-1 or 1).

- Our goal is to develop a classifier based on the training data that will correctly classify test observations using their feature measurements.
  - We have seen many approaches for this task, including linear discriminant analysis, logistic regression, classification trees, bagging, random forests, and boosting.

- We will now see a new approach that is based on the concept of a separating hyperplane.

Maximal Margin Classifier: Hyperplane

- A hyperplane is a subspace of dimension P-1, in a P-dimensional space.
  - For instance, in two dimensions, a hyperplane is a flat one-dimensional subspace, i.e., a line. In three dimensions, a hyperplane is a flat two-dimensional subspace, i.e., a plane. In P > 3 dimensions, it can be hard to visualize a hyperplane, but the notion of a (P − 1)-dimensional flat subspace still applies.

- Mathematically, in two dimensions, a hyperplane is defined by the equation

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0,$$

for parameters β_0, β_1, and β_2.

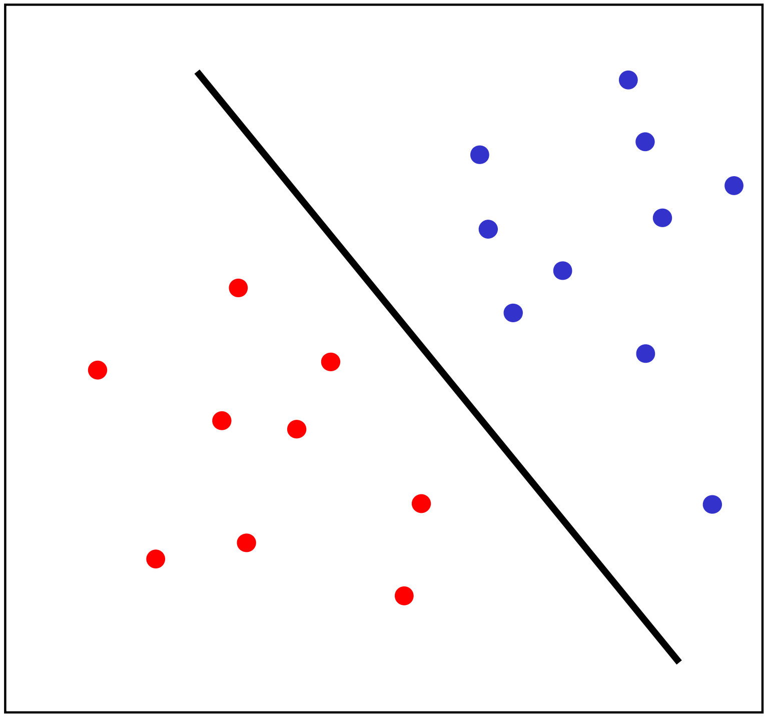
Maximal Margin Classifier: Hyperplane

- The equation for P = 2 can be easily extended to the P-dimensional setting:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_P X = 0.$$

Maximal Margin Classifier: Separating Hyperplane

- Then, a separating hyperplane has the property that



$y_i = \ 1$ (blue)
$y_i = -1$ (red)

Maximal Margin Classifier: Separating Hyperplane

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p > 0 \text{ if } y_i = 1,$$

- and

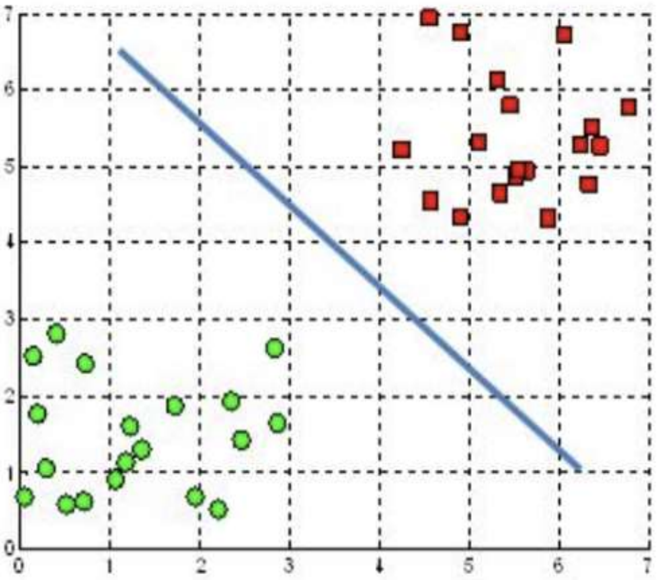
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p < 0, \text{ if } y_i = -1.$$

- Equivalently,

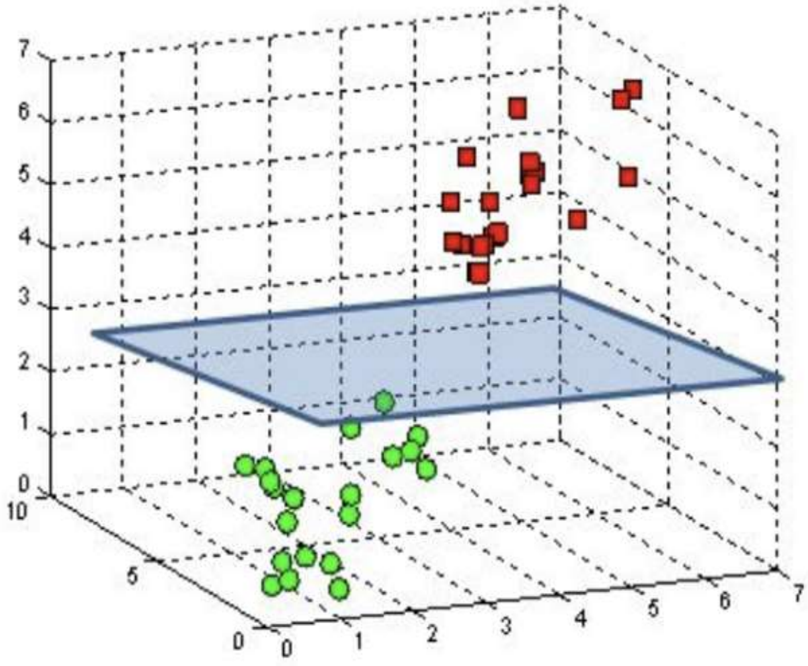
$$y_i(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p) = y_i f(x) > 0.$$

Maximal Margin Classifier: Separating Hyperplane

A hyperplane in $\mathbb{R}^2$ is a line



A hyperplane in $\mathbb{R}^3$ is a plane



https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

Maximal Margin Classifier: Separating Hyperplane

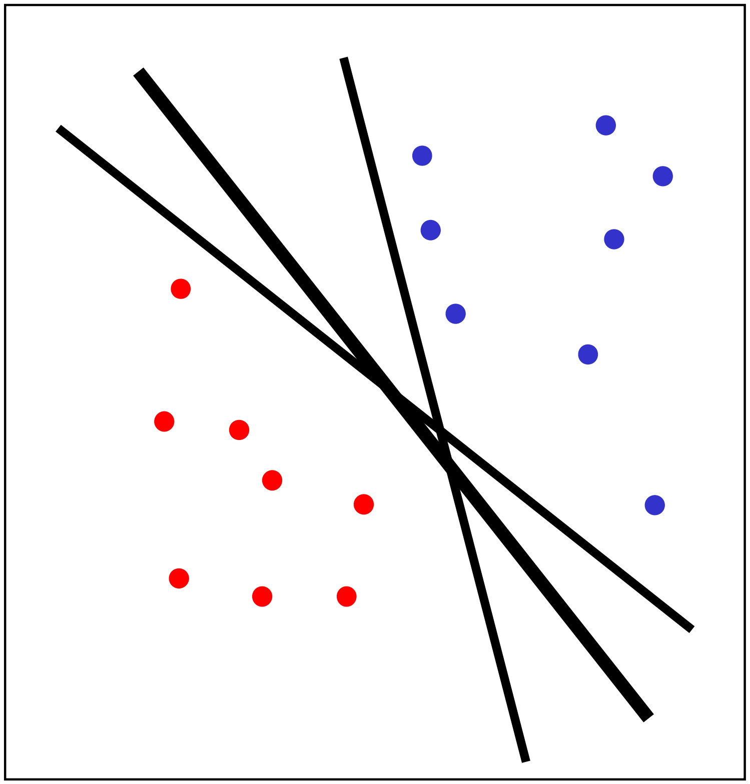
- If a separating hyperplane exists, we can use it to construct a very natural classifier. That is, we classify each observation based on the sign of f(x). If f(x) is positive, then we assign the observation to class 1, and if f(x) is negative, then we assign it to class -1.

- A classifier that is based on a separating hyperplane leads to a linear decision boundary.

Maximal Margin Classifier: Separating Hyperplane

- In general, if our data can be perfectly separated using a hyperplane, there will exist an infinite number of such hyperplanes.

- We must have a reasonable way to decide which of the infinite possible separating hyperplanes to use.

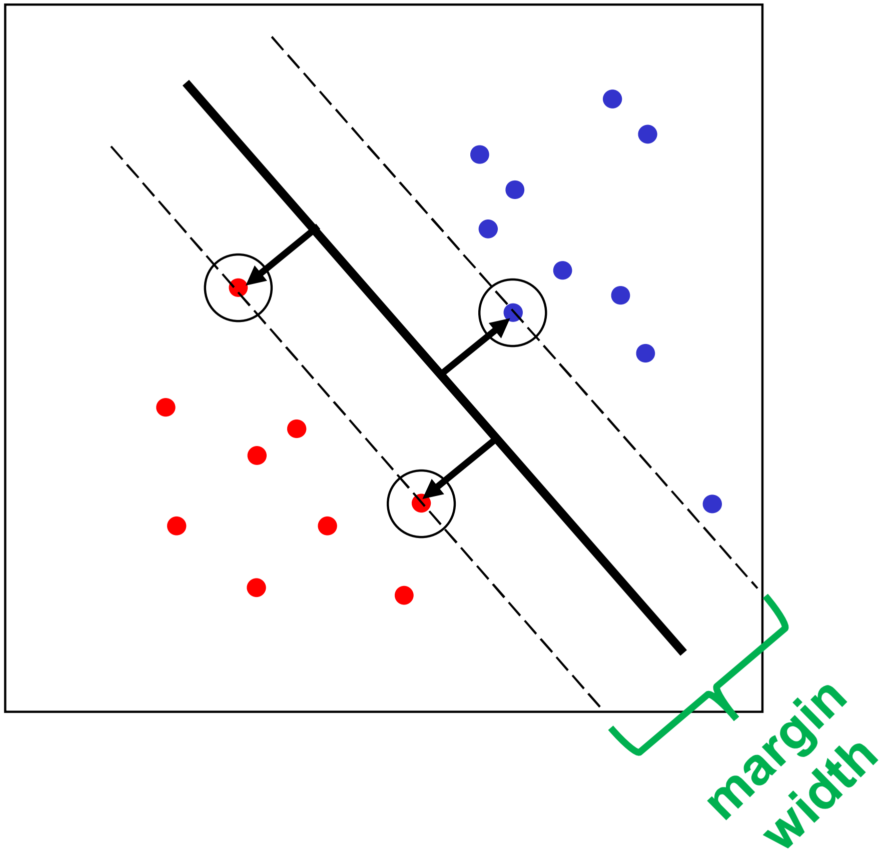


Maximal Margin Classifier

- A natural choice is the maximum margin hyperplane (or the optimal separating hyperplane), which is the separating hyperplane that is farthest from training observations.
  - We can compute the (perpendicular) distance from each training observation to a given separating hyperplane; the smallest such distance is the minimal distance from the observations to the hyperplane and is called the margin.

- We can classify based on the maximal margin hyperplane, which is the separating hyperplane for which the margin is largest. This is known as the maximal margin classifier.

Maximal Margin Classifier



- The maximal margin hyperplane represents the mid-line of the widest slab that we can insert between two classes.

- The 3 observations on the dashed lines are called support vectors, which support the maximal margin hyperplane in the sense that if these points were moved slightly then the maximal margin hyperplane would move as well.

Maximal Margin Classifier

- The maximal margin hyperplane depends directly on the support vectors, but not on the other observations: a movement to any of the other observations would not affect the separating hyperplane, provided that the observation’s movement does not cause it to cross the boundary set by the margin.

- The fact that the maximal margin hyperplane depends directly on only a small subset of the observations is an important property.

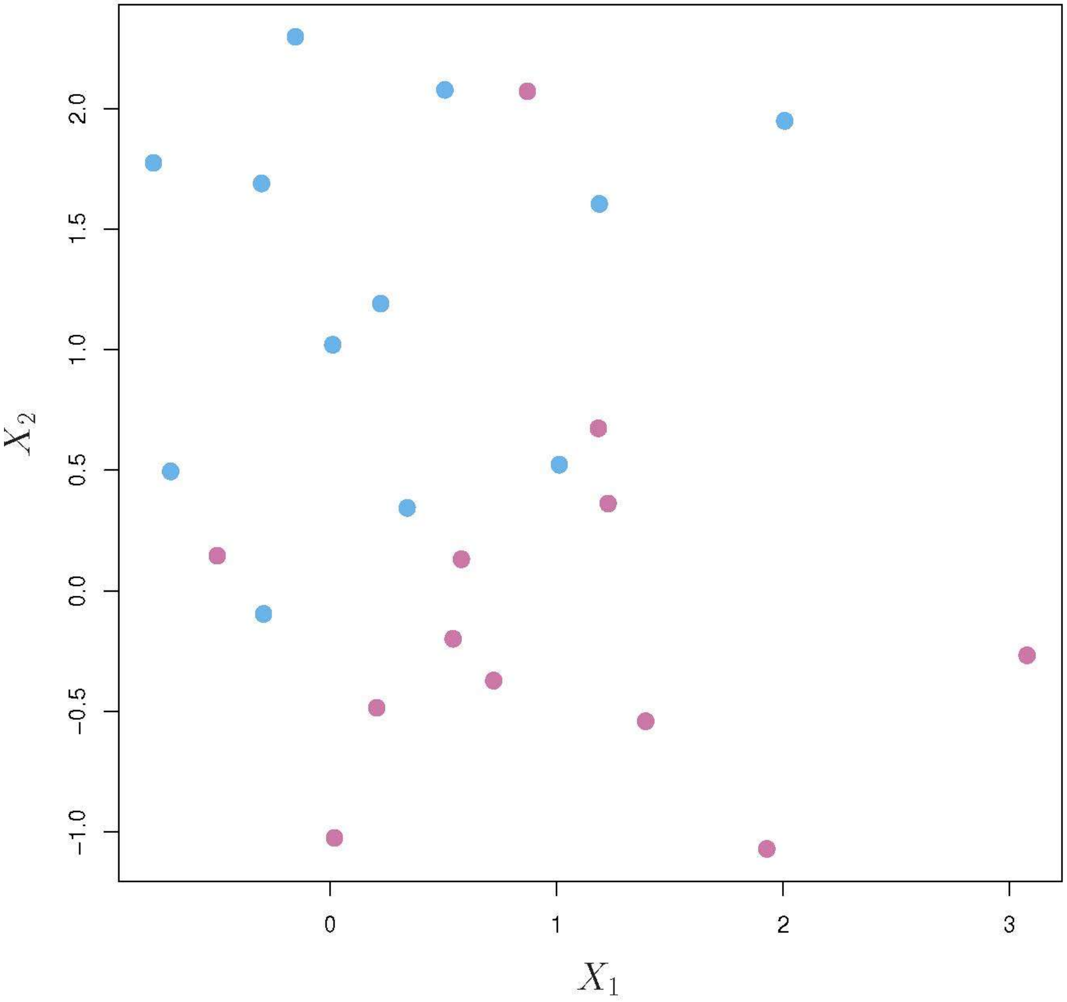
Maximal Margin Classifier

- Technically, we aim to choose the coefficients $\beta_0, \beta_1, .., \beta_p$ to maximize the margin of our hyperplane under some constraints.
  - Refer to Hastie et al. (2009, Chapter 12)

- The function svm() in the R package e1071 solves this optimization problem efficiently.

Support Vector Classifier

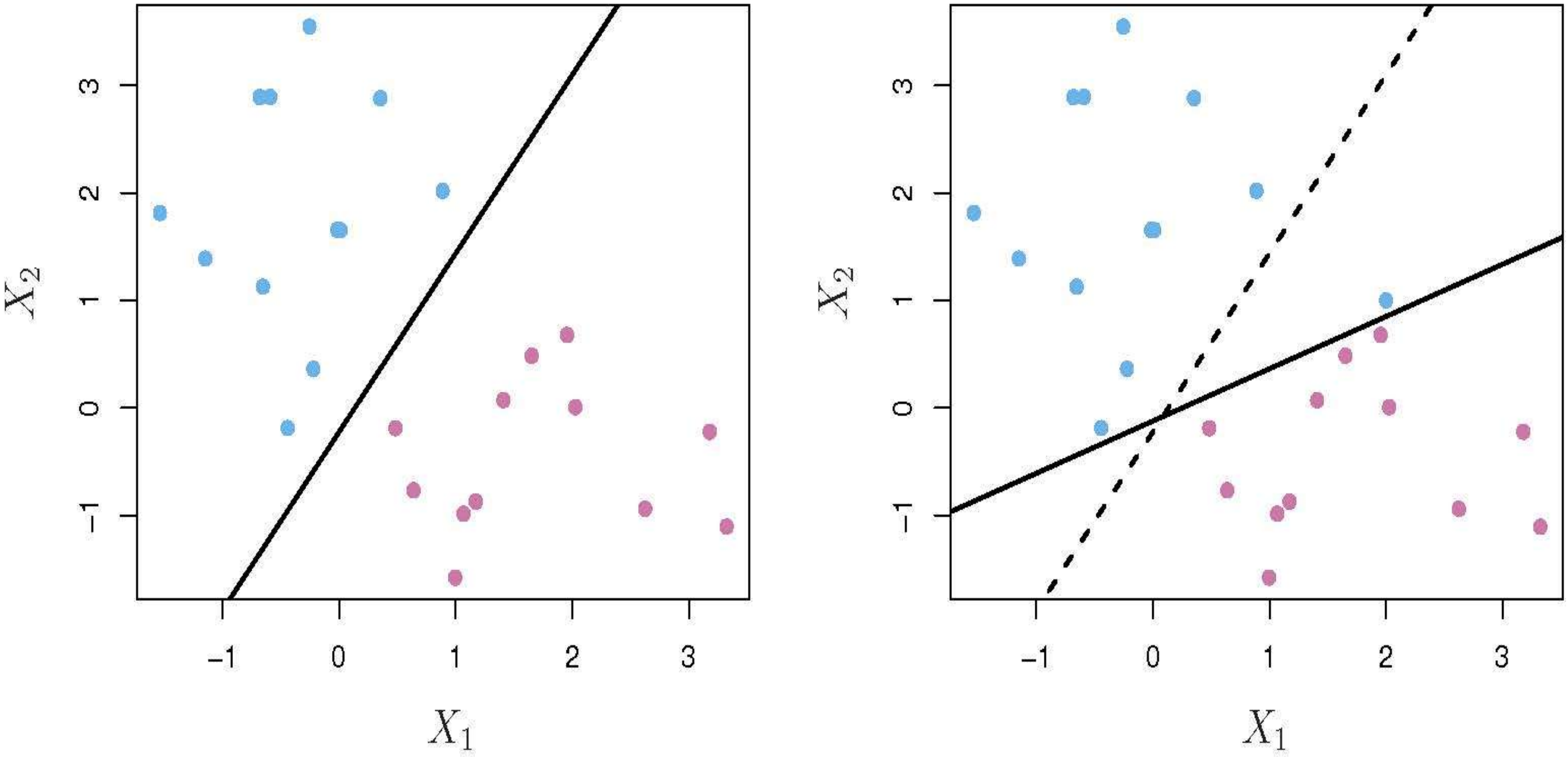
- The maximal margin classifier is a very natural way to perform classification if a separating hyperplane exists. However, in many cases, no separating hyperplane exists and so there is no maximal margin classifier.



James et al. (2021, Figure 9.4)

Support Vector Classifier

- Also, the maximal margin classifier can be very sensitive to a change in a single observation.



James et al. (2021, Figure 9.5)

Support Vector Classifier

- In these cases, we might be willing to consider a classifier based on a hyperplane that does not perfectly separate the two classes, in the interest of
  - Greater robustness to individual observations, and
  - Better classification of <u>most</u> of the training observations.
- That is, it could be worthwhile to misclassify a few training observations in order to do a better job of classifying the remaining observations.

Support Vector Classifier

- We can extend the concept of a separating hyperplane in order to develop a hyperplane that almost separates the classes, using a so-called soft margin.

- This generalization of the maximal margin classifier is known as the support vector classifier, also called a soft margin classifier.

Support Vector Classifier

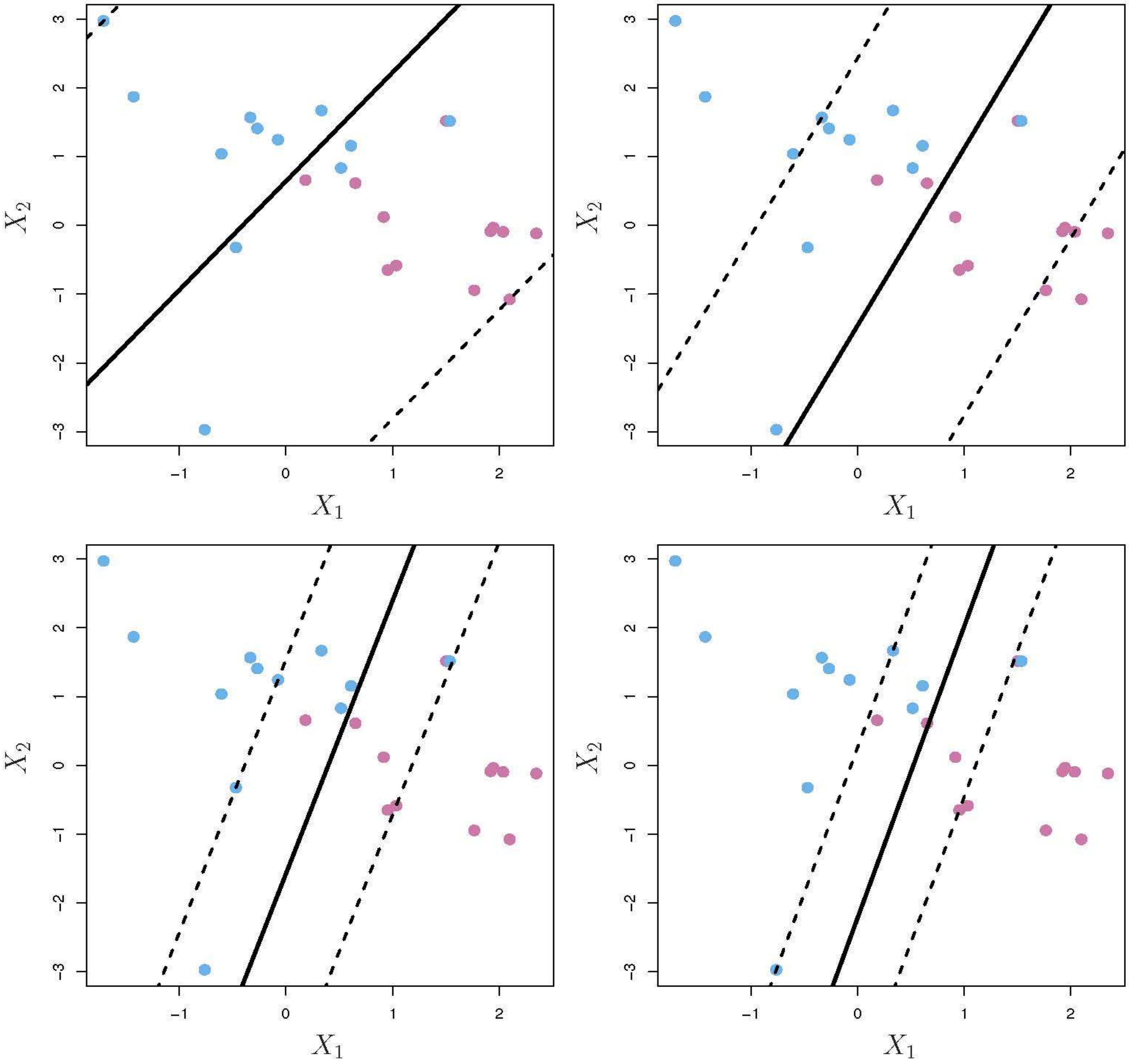
- The support vector classifier (SVC) allows some observations to be on the incorrect side of the margin or the hyperplane. The margin is soft because it can be violated by some training observations.

- Observations on the wrong side of the hyperplane correspond to training observations that are misclassified by the support vector classifier.
  - In fact, when there is no separating hyperplane, such a situation is inevitable.

Support Vector Classifier

- When applying the support vector classifier, we need to decide on a nonnegative tunning parameter, which represents the amount that the margin can be violated by observations.

- As the tunning parameter increases, we become more tolerant of violations to the margin, and so the margin will widen. Conversely, as it decreases, we become less tolerant of violations to the margin and so the margin narrows.
  - If the tunning parameter = 0, SVC amounts to the maximal margin classifier.



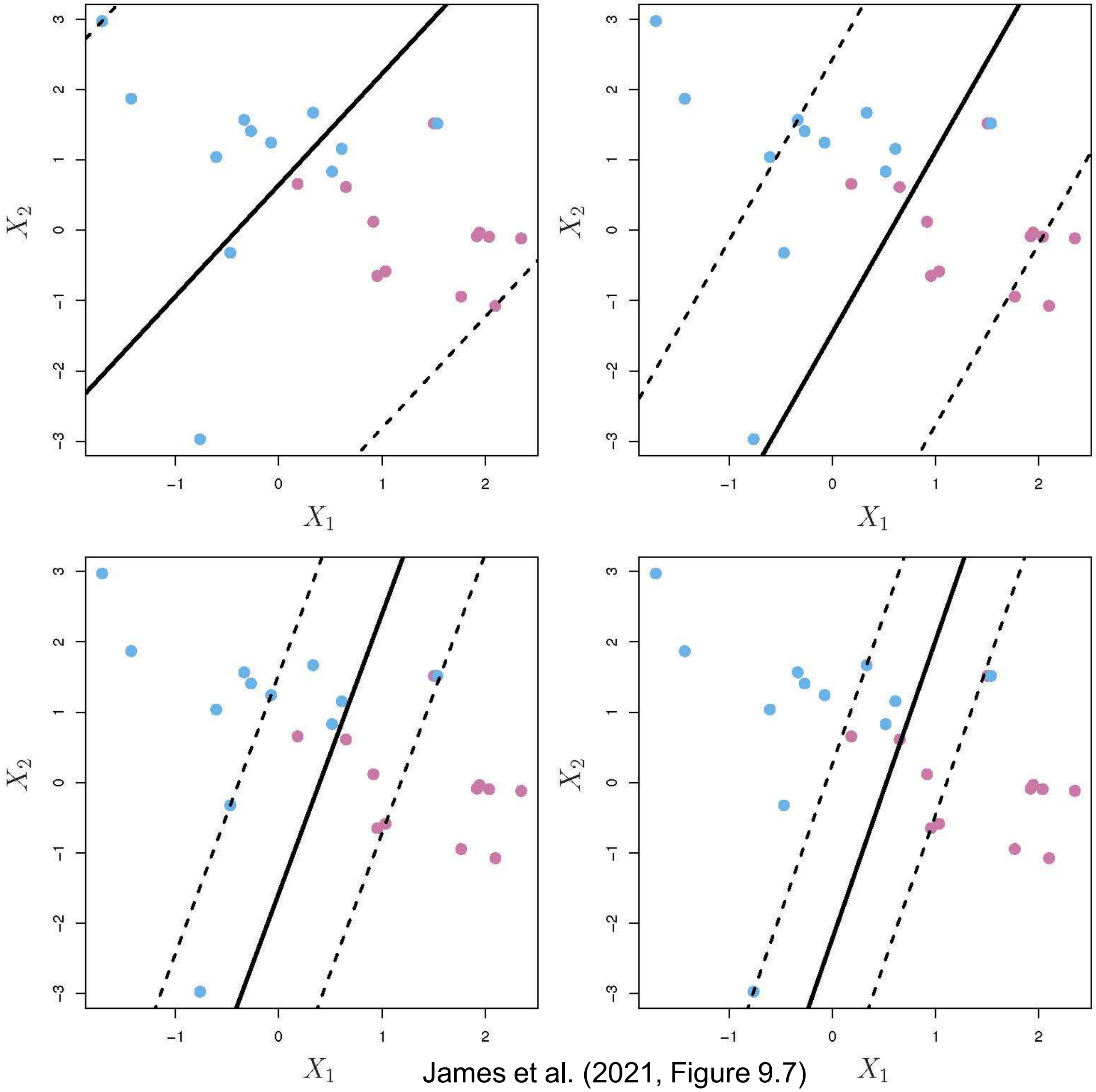
James et al. (2021, Figure 9.7)

Support Vector Classifier

- In practice, the tuning parameter is chosen via cross validation. As with the tuning parameters that we have seen for other methods, this tuning parameter controls the bias-variance tradeoff of the statistical learning technique.
    - When it is small, we seek narrow margins that are rarely violated; this amounts to a classifier that is highly fit to the data, which may have low bias but high variance. Conversely, when it is larger, the margin is wider, and we allow more violations to it; this amounts to fitting the data less hard and obtaining a classifier that is potentially more biased but may have lower variance.

Support Vector Classifier

- In SVC, observations that lie directly on the margin, or on the wrong side of the margin for their class, are known as support vectors. Only these observations affect the support vector classifier.



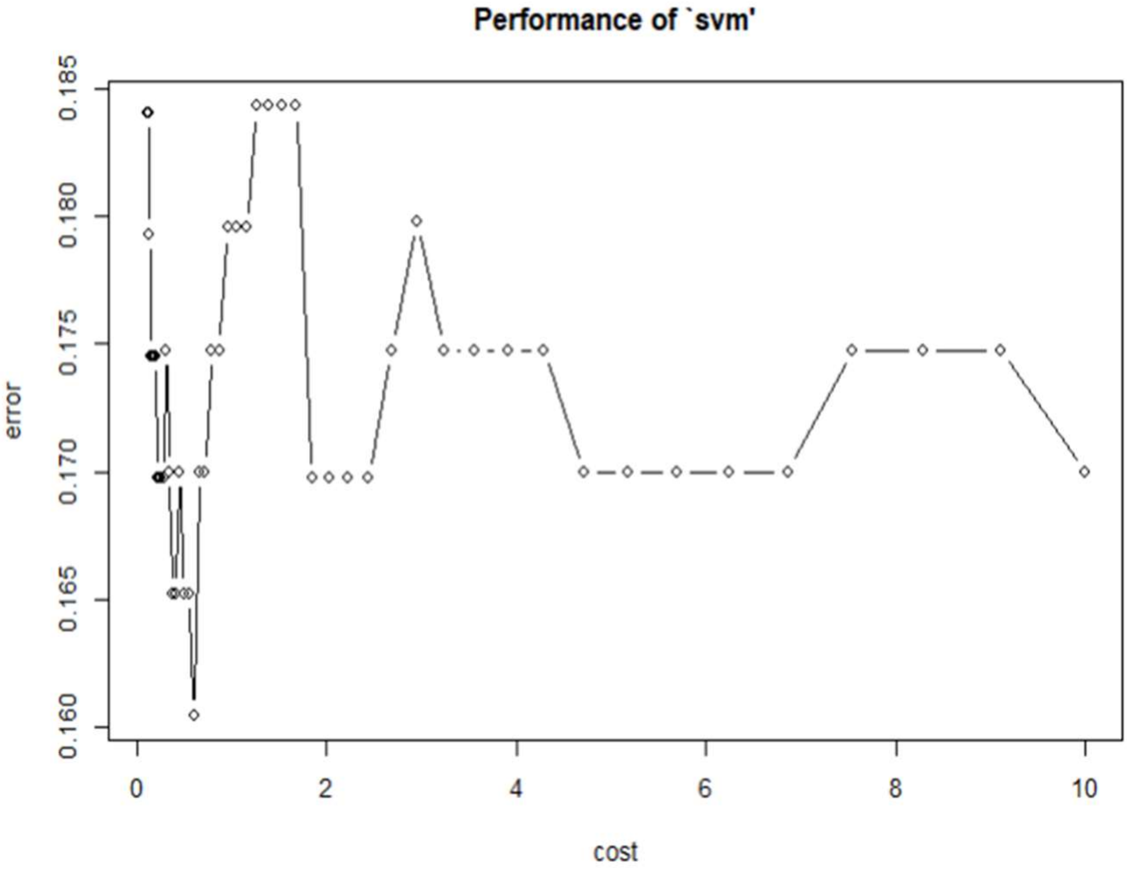
James et al. (2021, Figure 9.7)

Example: Support Vector Classifier

- The Heart data (Heart_training.csv & Heart_test.csv)
- Y = HD (1 = heart disease, 0 = no heart disease)
- X = 13 predictors, including Age, Sex, Chol (a cholesterol measurement), chest pain, etc. (refer to Heart_README.txt)

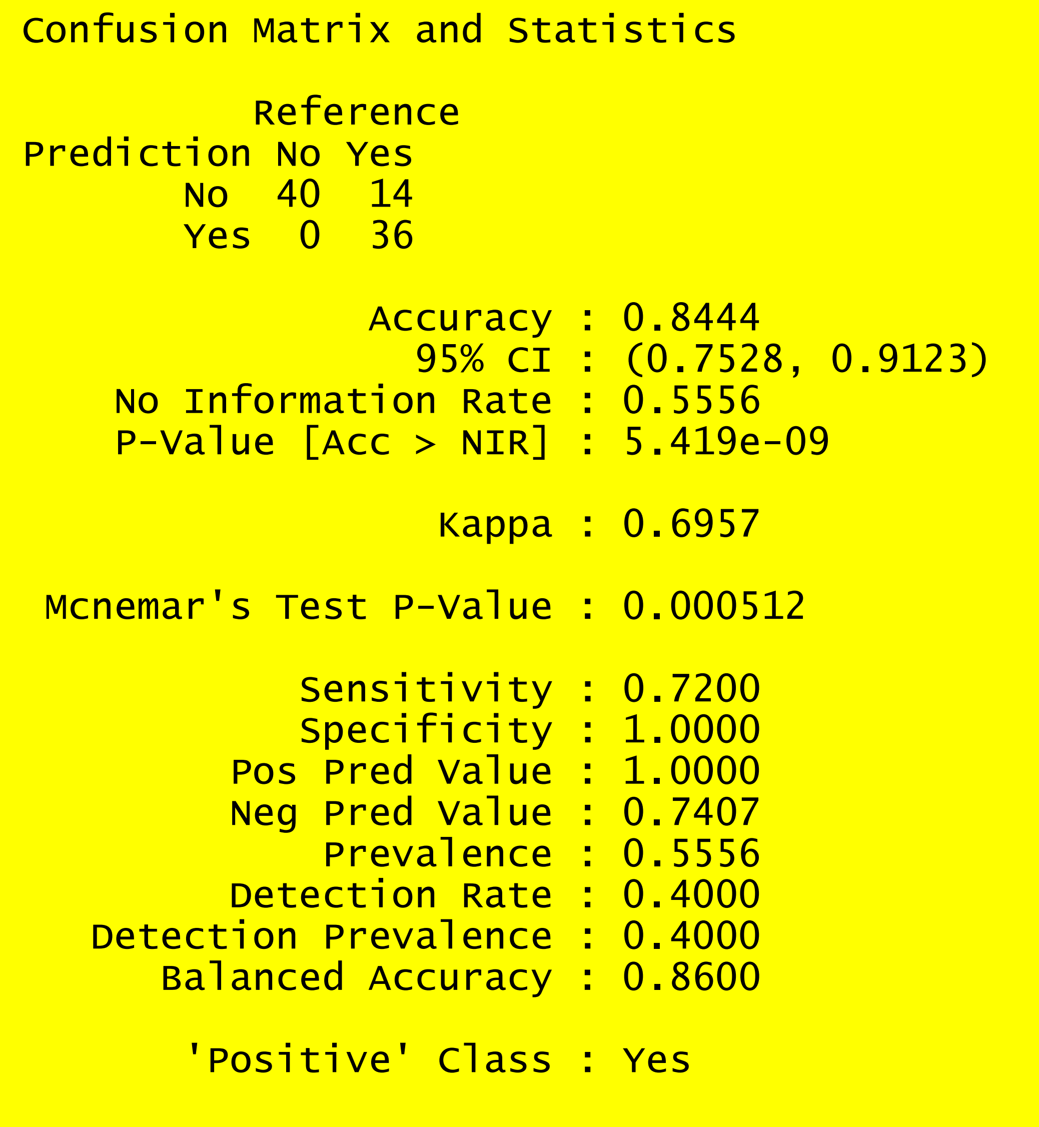
*Janosi, Andras, Steinbrunn, William, Pfisterer, Matthias, Detrano, Robert & M.D., M.D.. (1988). Heart Disease. UCI Machine Learning Repository.

Example: Support Vector Classifier

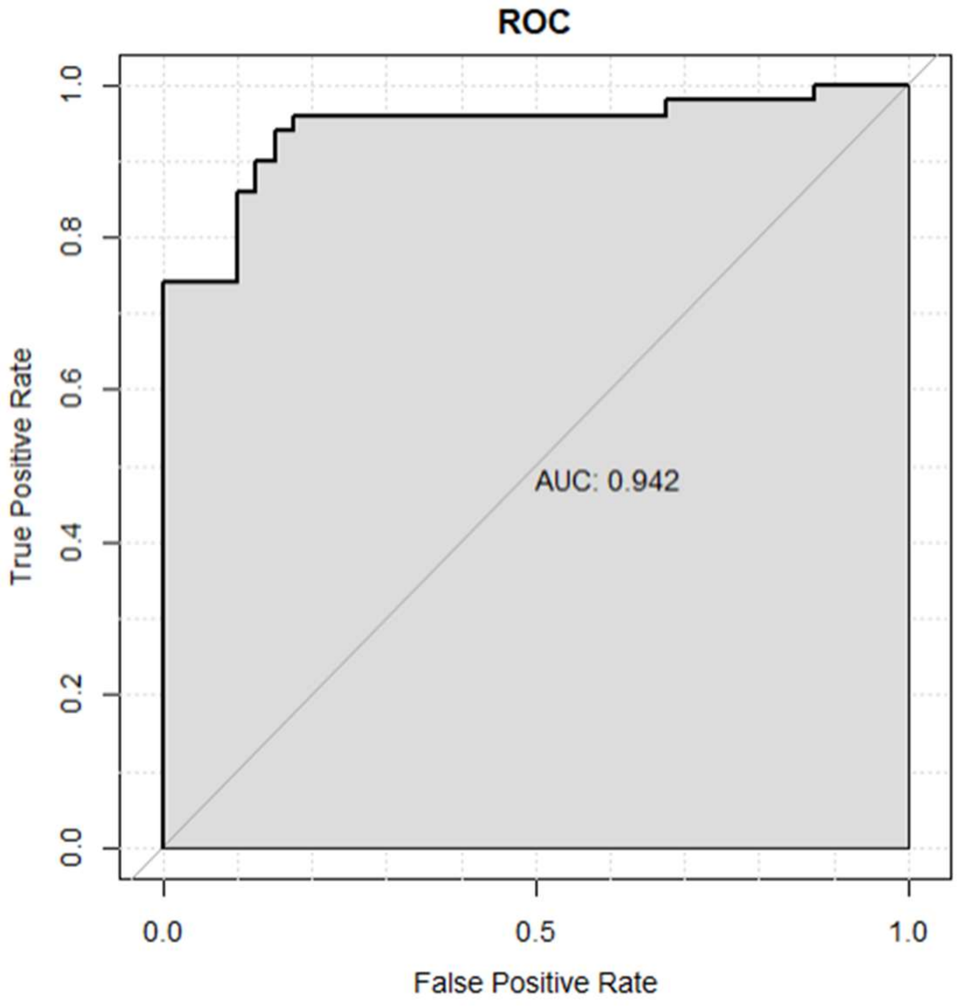


```
- sampling method: 10-fold cross
validation

- best parameters:
      cost
   0.5963623

- best performance: 0.1604762

- Detailed performance results:
         cost              error
dispersion
1  10.0000000 0.1700000 0.07892896
2   9.1029818 0.1747619 0.08168976
3   8.2864277 0.1747619 0.08168976
4   7.5431201 0.1747619 0.08168976
5   6.8664885 0.1700000 0.07892896
6   6.2505519 0.1700000 0.07892896
7   5.6898660 0.1700000 0.07892896
8   5.1794747 0.1700000 0.07892896
                …
```

Example: Support Vector Classifier



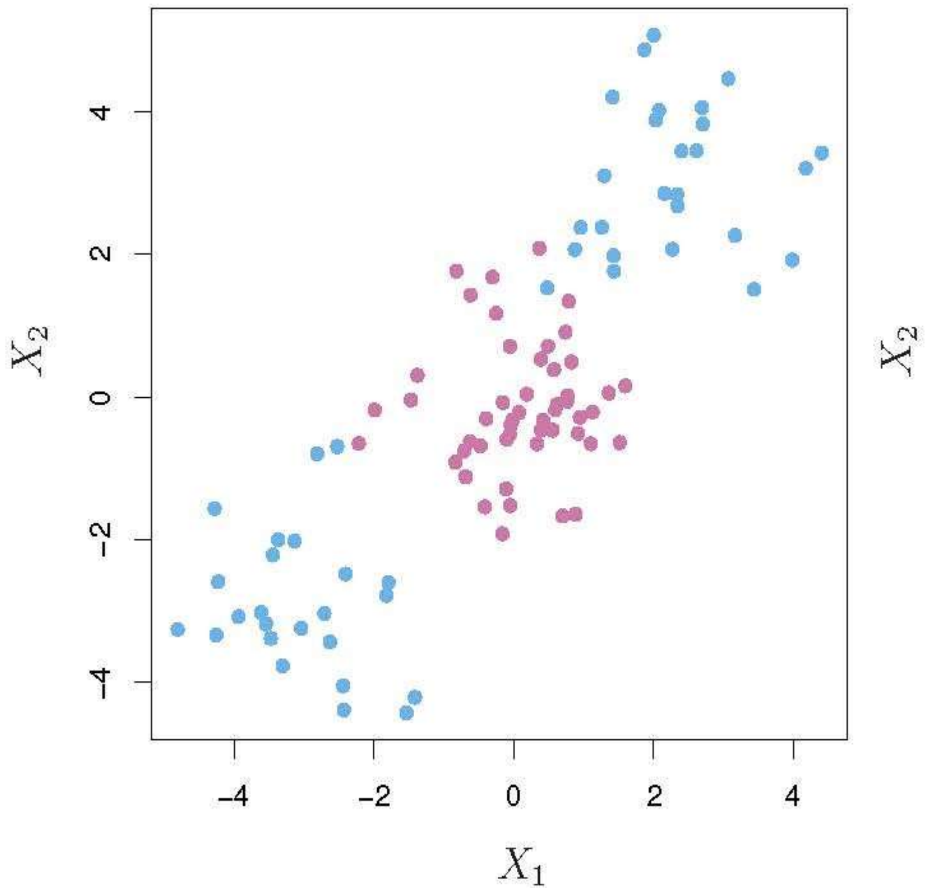
```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  40  14
       Yes  0  36

               Accuracy : 0.8444
                 95% CI : (0.7528, 0.9123)
    No Information Rate : 0.5556
    P-Value [Acc > NIR] : 5.419e-09

                  Kappa : 0.6957

 Mcnemar's Test P-Value : 0.000512

            Sensitivity : 0.7200
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.7407
             Prevalence : 0.5556
         Detection Rate : 0.4000
   Detection Prevalence : 0.4000
      Balanced Accuracy : 0.8600

       'Positive' Class : Yes
```



Support Vector Machine

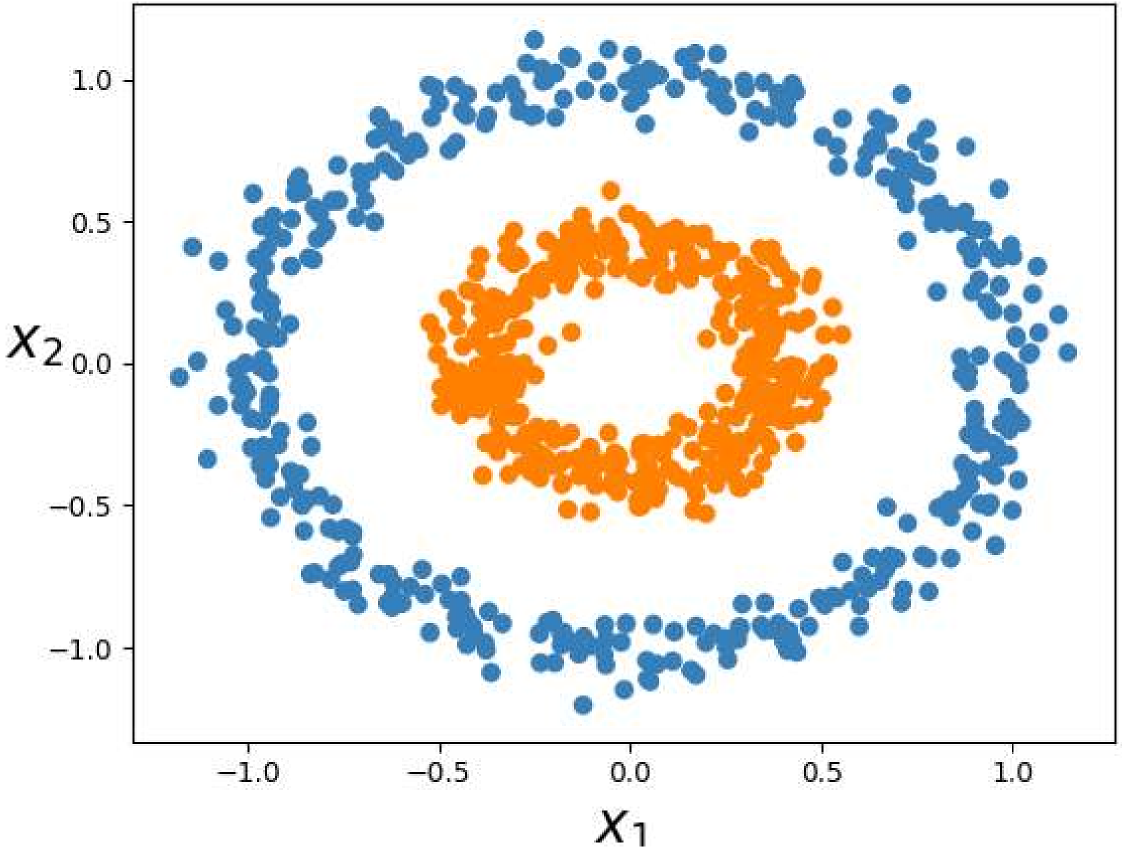
- The support vector classifier is a natural approach for classification in the two-class setting if the boundary between the two classes is linear.

- However, in practice, we are sometimes faced with non-linear class boundaries.



James et al. (2021, Figure 9.8)

Support Vector Machine



https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f

Support Vector Machine

- In linear regression, when there is a nonlinear relationship between the predictors and the outcome, we consider enlarging the predictor/feature space using functions of the predictors, such as quadratic and cubic terms, in order to address this non-linearity.

- In the case of the support vector classifier, we could address the problem of possibly non-linear boundaries between classes in a similar way, by enlarging the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors.

Support Vector Machine

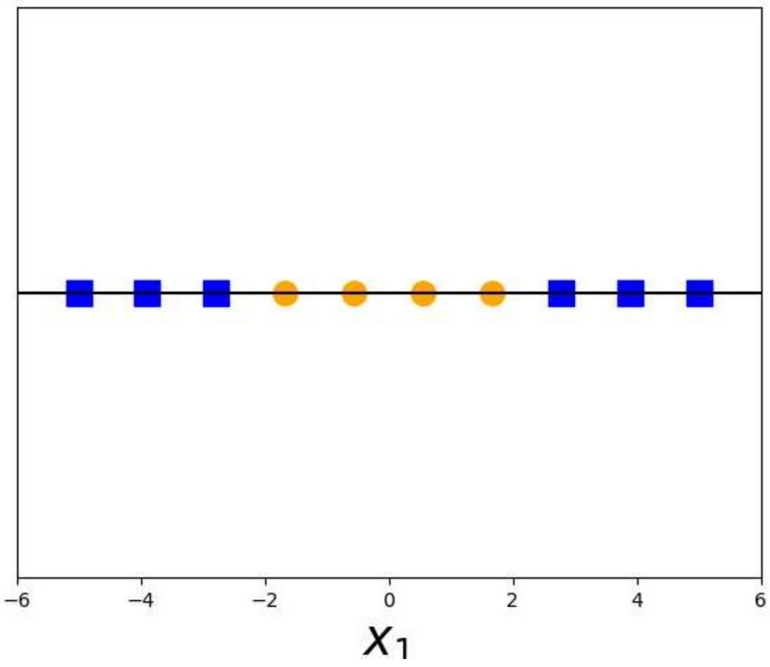
- If the data are not linearly separable in the original, or input, space, then we apply transformations to the data or create nonlinear combinations of the original features, which map the data from the original space into a higher dimensional feature space.

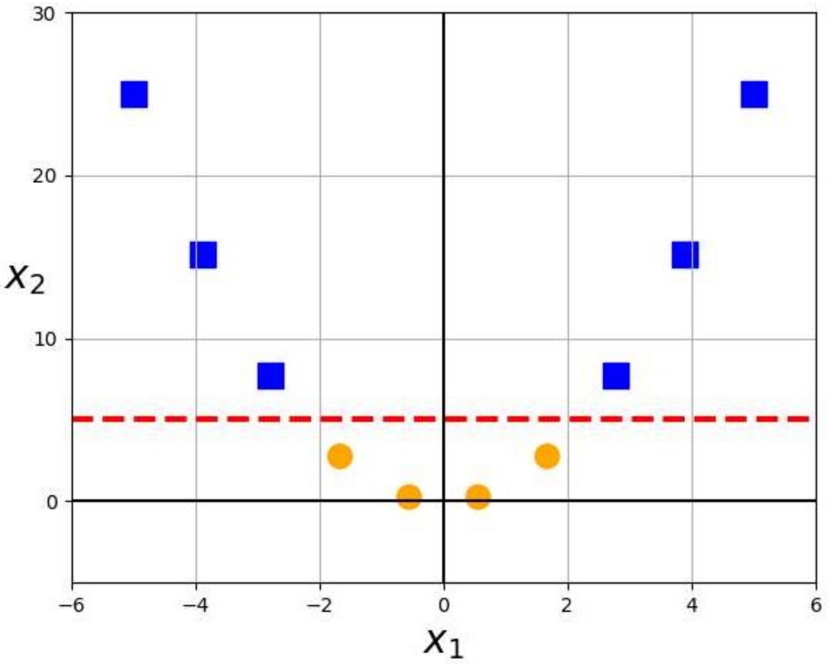
- The goal is that after the transformation to the higher dimensional space, the classes are now linearly separable in this higher dimensional feature space.

- We can then fit a hyperplane to separate the classes and make predictions.

Support Vector Machine

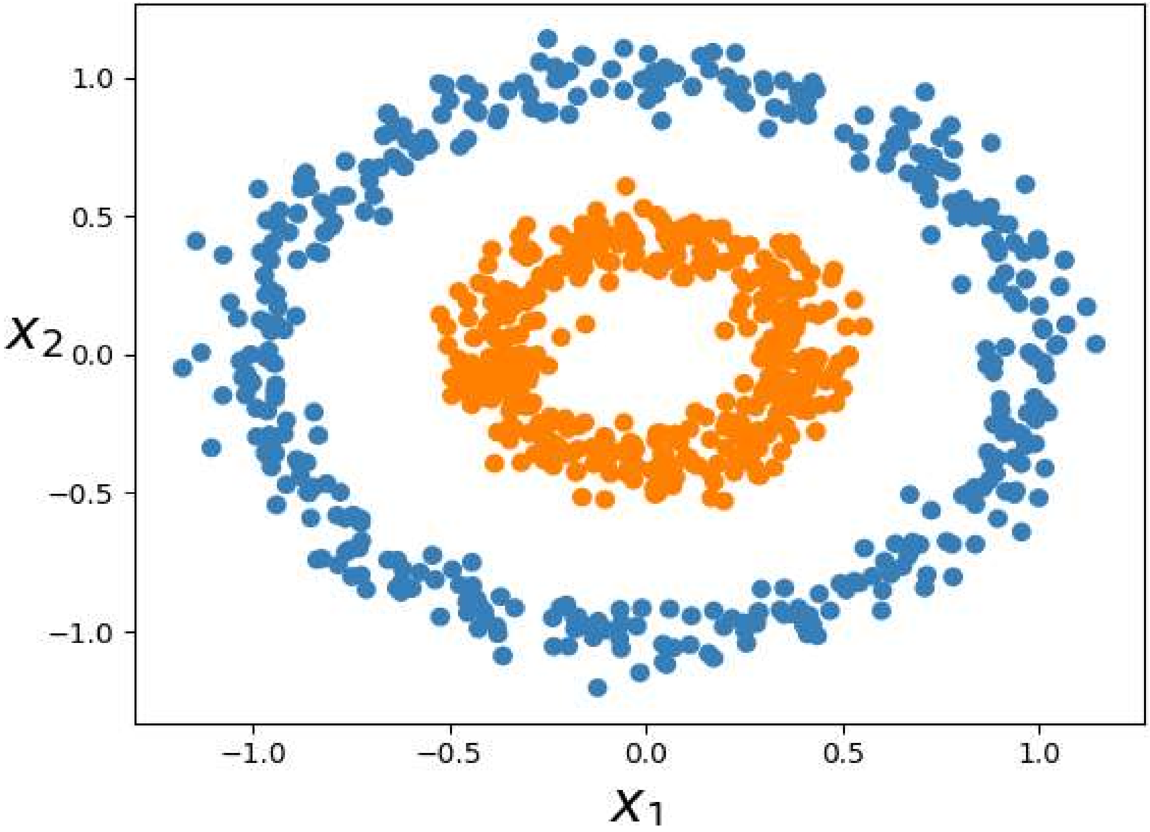
$$\phi(x) = \begin{pmatrix} x \\ x^2 \end{pmatrix} \rightarrow \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$$





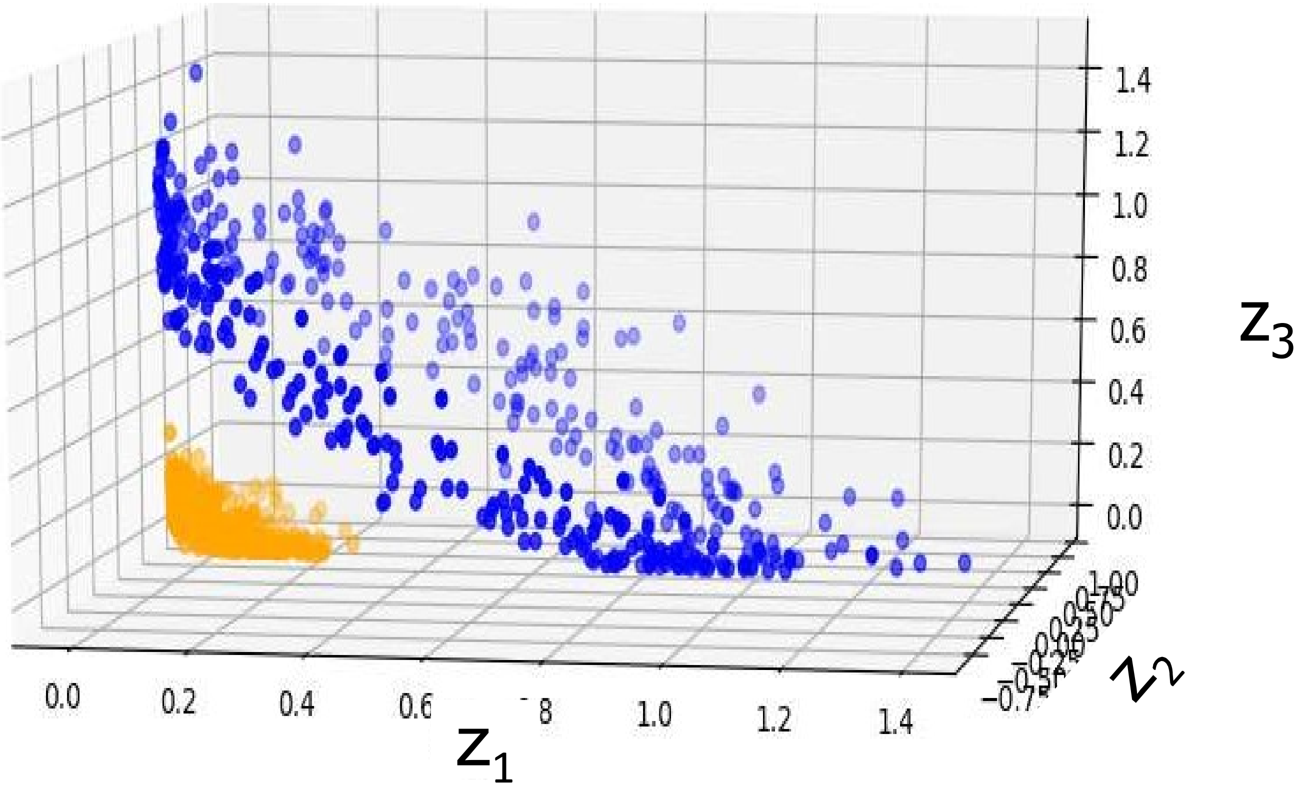
https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f

Support Vector Machine

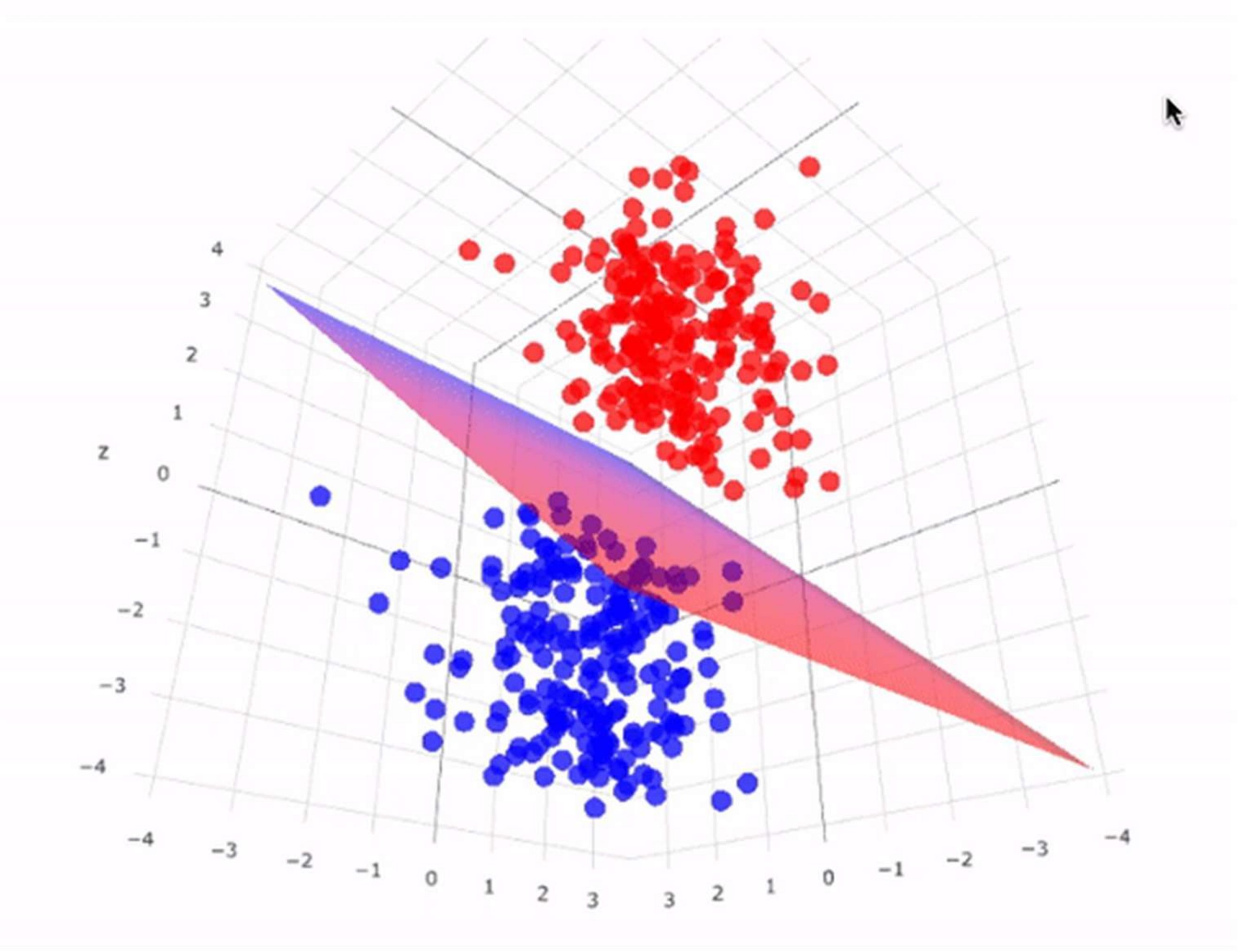


https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f

$$\phi(\boldsymbol{x}) = \phi\begin{pmatrix}x_1\\ x_2\end{pmatrix} = \begin{pmatrix}x_1^2\\ \sqrt{2}x_1x_2\\ x_2^2\end{pmatrix} \Rightarrow \begin{pmatrix}z_1\\ z_2\\ z_3\end{pmatrix}$$

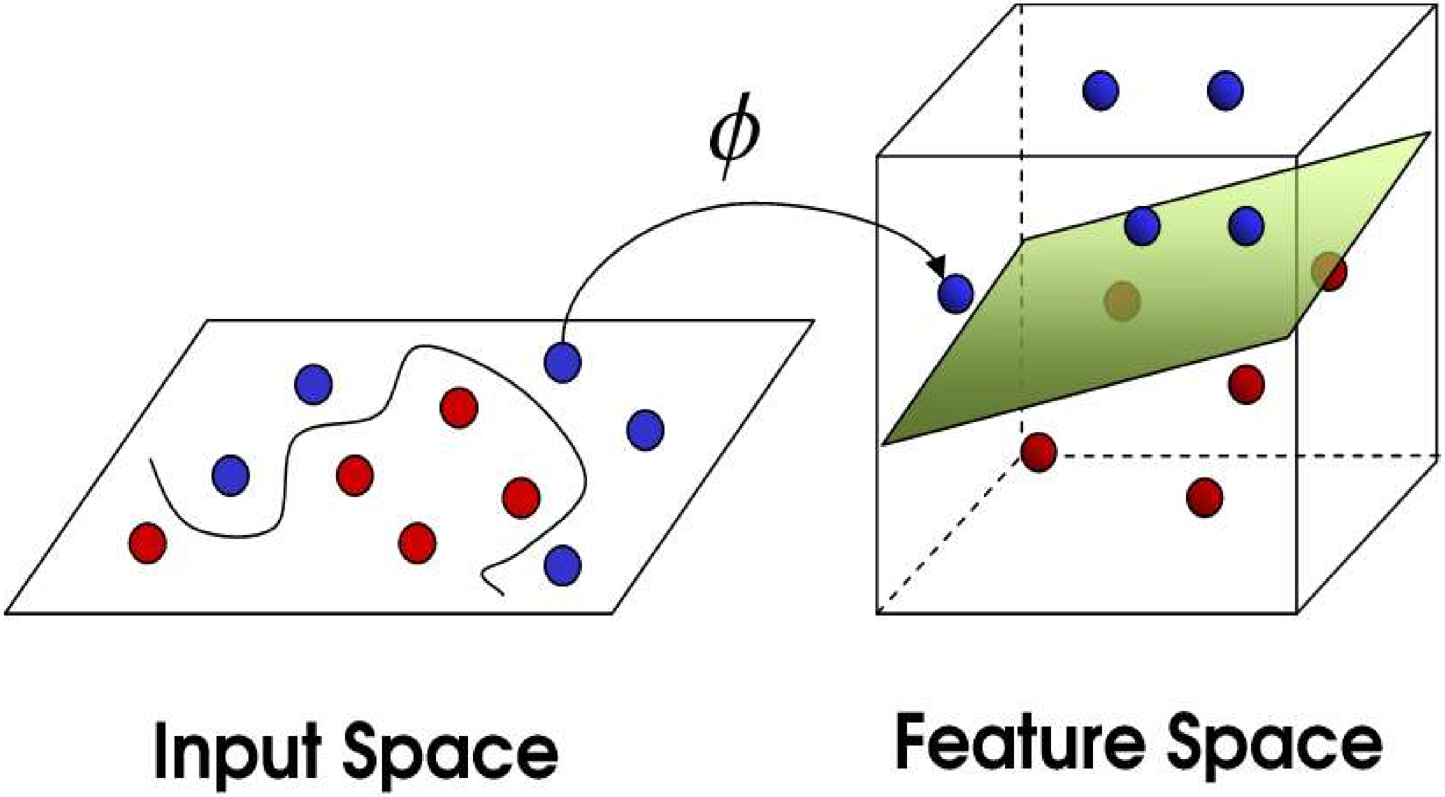


https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f



https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

Support Vector Machine



Miller, Sacchet, & Gotlib (2020, *Emotion Review*)

Support Vector Machine

- There are many possible ways to enlarge the feature space (transform the features), and unless we are careful, we could end up with a huge number of features. Then computations would become unmanageable.
  - In real applications, there might be many features in the data and applying transformations that involve many polynomial combinations of these features will lead to extremely high and impractical computational costs.

- The support vector machine allows us to enlarge the feature space used by the support vector classifier in a way that leads to efficient computations, called the kernel trick/approach.

Support Vector Machine

- The “trick” is to represent the data only through a set of pairwise similarity comparisons between the original observations **x**, instead of explicitly applying the transformations ϕ(**x**) and representing the data by these transformed coordinates in the higher dimensional feature space.

- The data set X is represented by an N x N kernel matrix of pairwise similarity comparisons where the entries (i, i') are defined by the kernel function: $K\left(x_i, x_{i'}\right)$ $(i = 1,\ldots,N)$.

Support Vector Machine

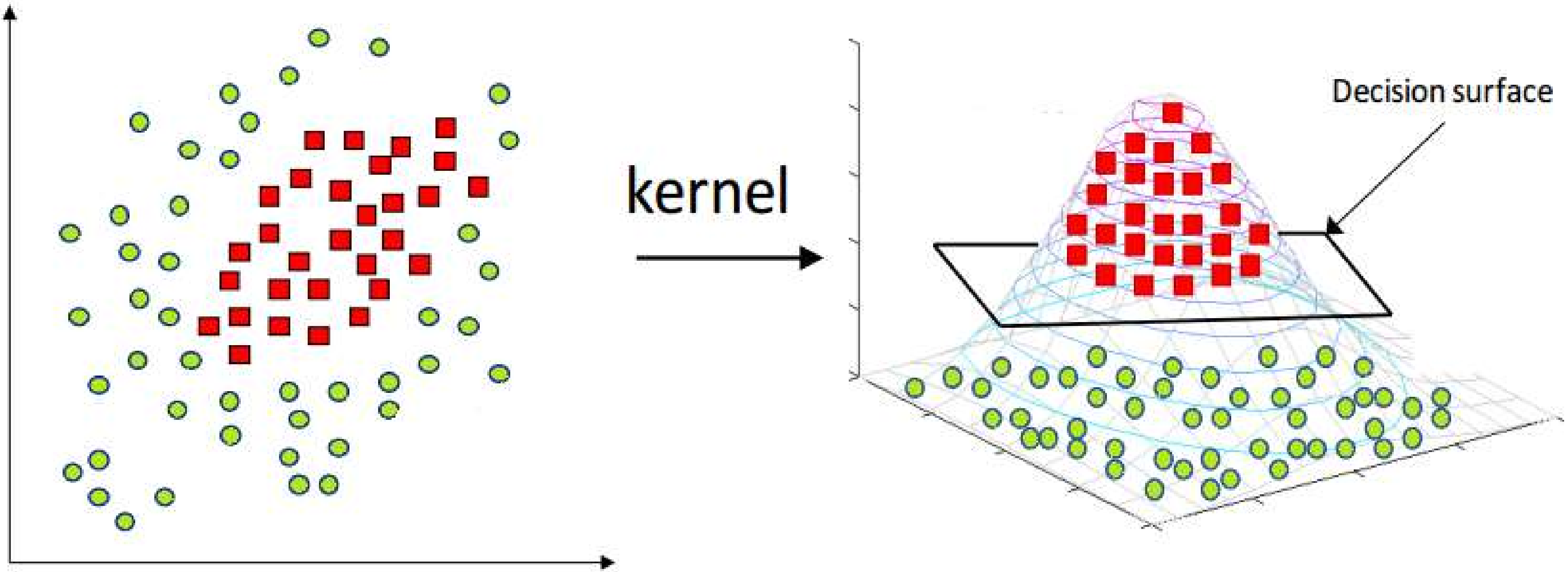
- A kernel is a function that quantifies the similarity of two observations. For example,
  - Linear kernel: $K\left(x_i, x_{i'}\right) = \sum_{j=1}^{P} x_{ij} x_{i'j}$
    - The linear kernel essentially quantifies the similarity of pair of observations using Pearson correlation.
  - Polynomial kernel: $K\left(x_i, x_{i'}\right) = (1 + \sum_{j=1}^{P} x_{ij} x_{i'j})^{\mathbf{d}}$
    - A kernel polynomial kernel of d > 1 amounts to fitting a support vector classifier in a higher-dimensional space involving polynomials of degree d.
  - Radial/Gaussian kernel: $K\left(x_i, x_{i'}\right) =$ $\exp(-\frac{1}{2\sigma} \sum_{j=1}^{P} (x_{ij} - x_{i'j})^2) = \exp(-\boldsymbol{\gamma} \sum_{j=1}^{P} (x_{ij} - x_{i'j})^2)$
    - σ is a positive constant.

Support Vector Machine

- When the support vector classifier is combined with a non-linear kernel such as the polynomial or radial kernel, the resulting classifier is known as a support vector machine (SVM).

- Our data are only linearly separable as the vectors $\phi(\mathbf{x})$ in the higher dimensional space, and SVM aims to find the optimal separating hyperplane in this higher dimensional space without having to calculate or even know anything about $\phi(\mathbf{x})$.

Support Vector Machine



https://medium.com/@apurvjain37/support-vector-machine-s-v-m-classifiers-and-kernels-9e13176c9396

Support Vector Machine

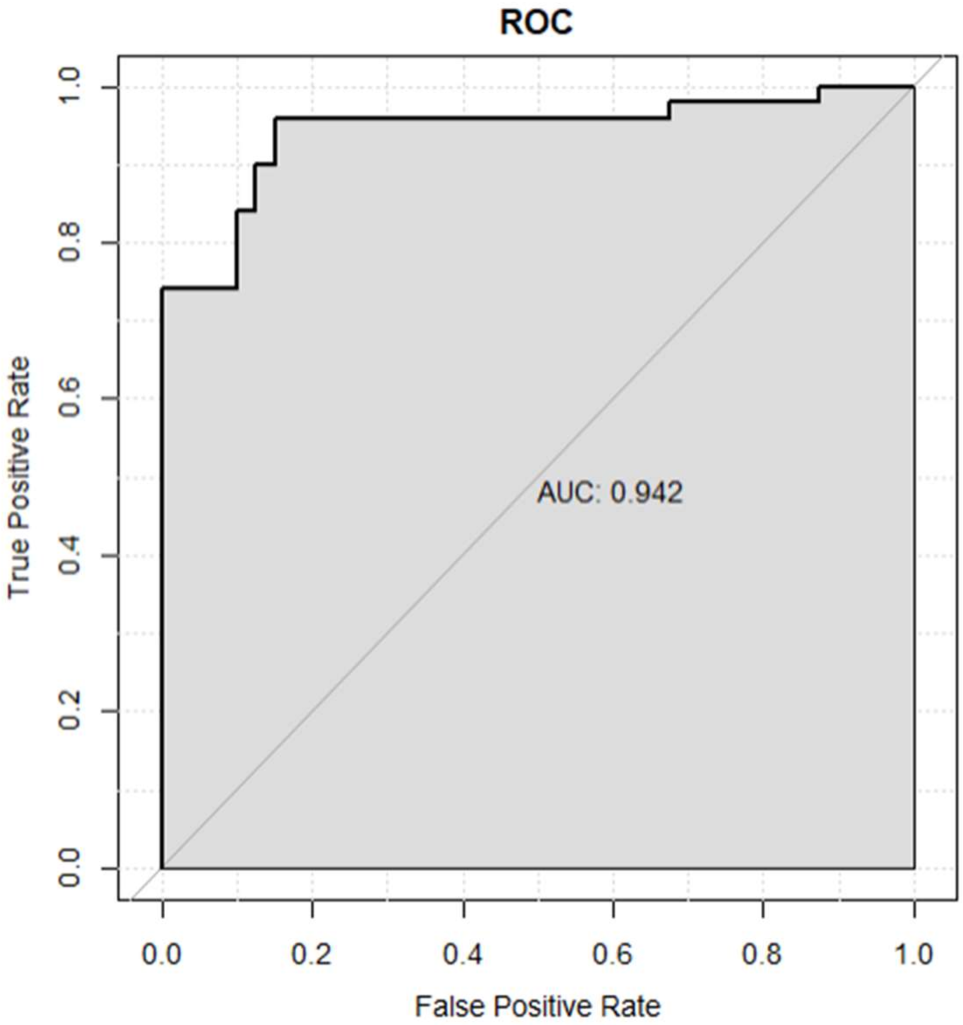
- SVM visualization: https://www.youtube.com/watch?v=OdlNM96sHio&ab_channel=udiprod

Example: Support Vector Machine with Polynomial Kernel

```
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
  degree     cost
   1        5.689866

- best performance: 0.1680952

- Detailed performance results:
    degree      cost     error  dispersion
1        1 10.0000000 0.1830952 0.07367243
2        2 10.0000000 0.1876190 0.06814129
3        3 10.0000000 0.1876190 0.08160181
4        1  9.1029818 0.1878571 0.07571320
5        2  9.1029818 0.1876190 0.07517344
6        3  9.1029818 0.1971429 0.08135443
       …
```

Example: Support Vector Machine with Polynomial Kernel

```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  40  14
       Yes  0  36

               Accuracy : 0.8444
                 95% CI : (0.7528, 0.9123)
    No Information Rate : 0.5556
    P-Value [Acc > NIR] : 5.419e-09

                  Kappa : 0.6957

 Mcnemar's Test P-Value : 0.000512

            Sensitivity : 0.7200
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.7407
             Prevalence : 0.5556
         Detection Rate : 0.4000
   Detection Prevalence : 0.4000
      Balanced Accuracy : 0.8600

       'Positive' Class : Yes
```

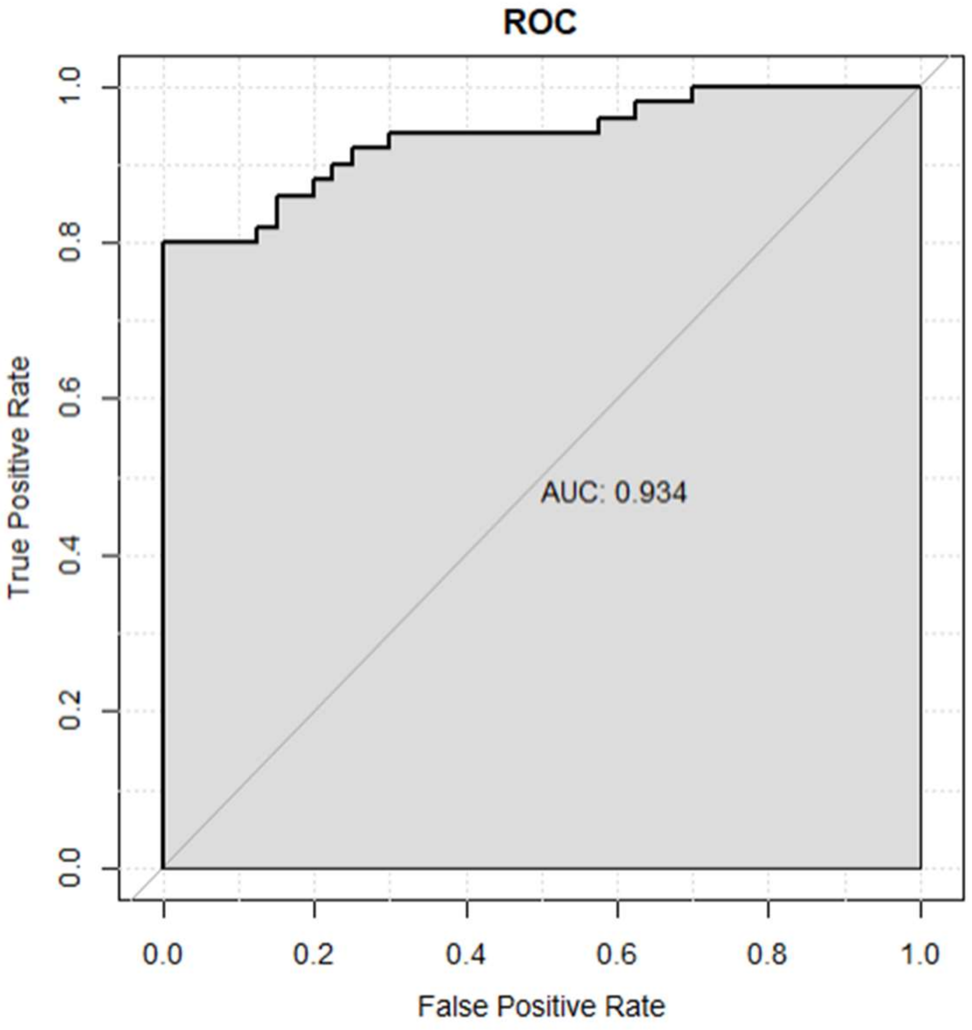


Example: Support Vector Machine with Radial Kernel

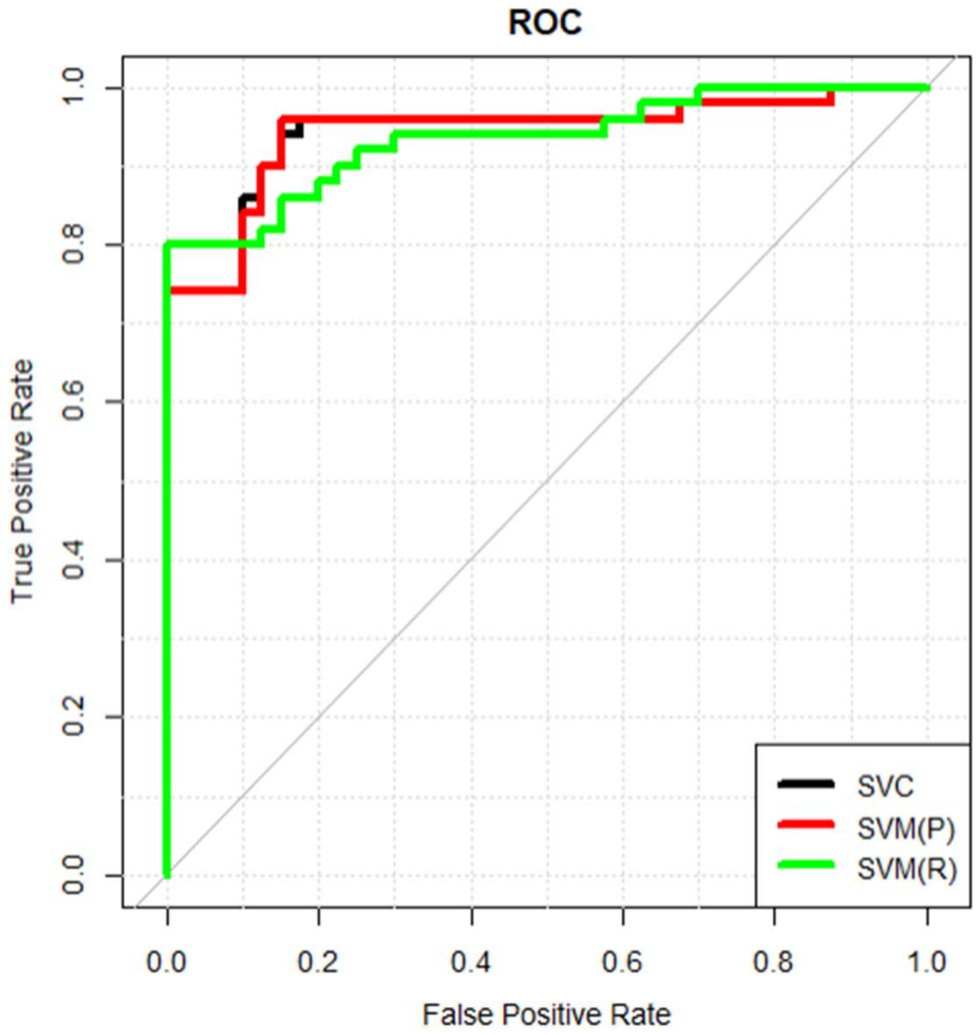
```
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
  gamma     cost
  0.01      3.237458

- best performance: 0.1654762

- Detailed performance results:
    gamma       cost    error  dispersion
1   0.100 10.0000000 0.2478571 0.10737746
2   0.010 10.0000000 0.1892857 0.07983393
3   0.001 10.0000000 0.1802381 0.10050296
4   0.100  9.1029818 0.2480952 0.11039337
5   0.010  9.1029818 0.1845238 0.07469810
6   0.001  9.1029818 0.1802381 0.10050296
               …
```

Example: Support Vector Machine with Radial Kernel

```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  40  13
       Yes  0  37

               Accuracy : 0.8556
                 95% CI : (0.7657, 0.9208)
    No Information Rate : 0.5556
    P-Value [Acc > NIR] : 1.203e-09

                  Kappa : 0.7167

 Mcnemar's Test P-Value : 0.0008741

            Sensitivity : 0.7400
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.7547
             Prevalence : 0.5556
         Detection Rate : 0.4111
   Detection Prevalence : 0.4111
      Balanced Accuracy : 0.8700

       'Positive' Class : Yes
```

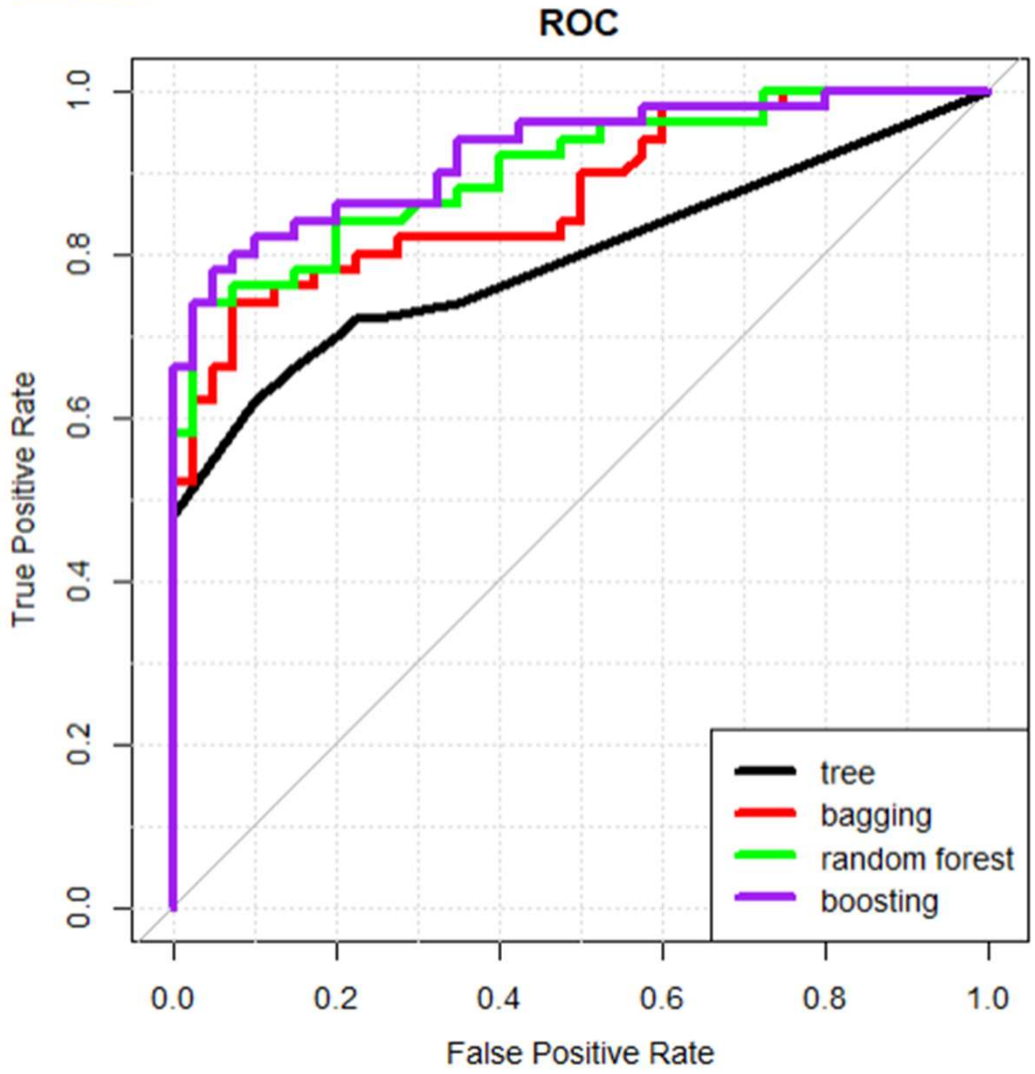


Example: SVC vs. SVM



```
AUC(SVC)              :  0.9425
AUC(SVM(Polynomial)):  0.9425
AUC(SVM(Radial))     :  0.934
```

Example: Tree-based Methods



```
AUC(tree)          :  0.79525
AUC(bagging)       :  0.87225
AUC(random forest):  0.90175
AUC(boosting)      :  0.9225
```