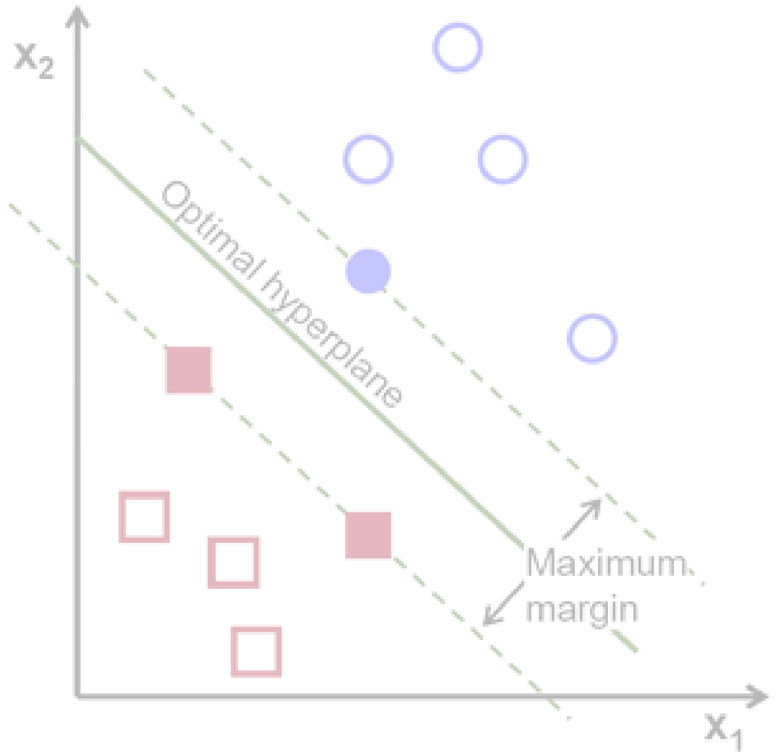
Support Vector Machine (SVM)



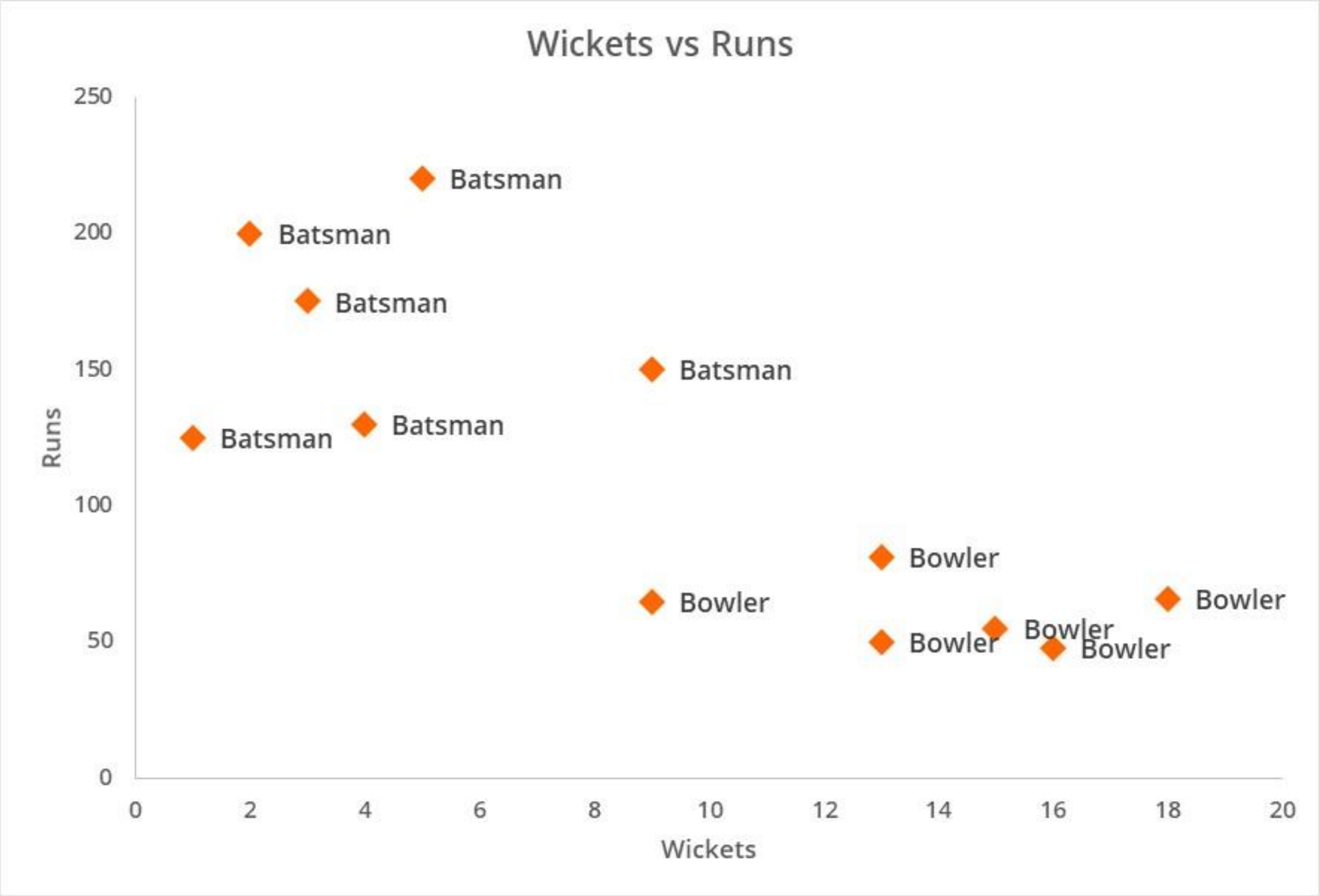
Contents

Support Vector Machine

- **SVM** is a Supervised ML algorithm used for classification and regression problems.
- SVM generates partitions between classes by generating **two parallel lines**.
- Divides the 2 categories by a clear gap that should be as wide as possible, by a plane called **hyperplane**.
- This hyperplane have the **largest margin** to separate given data into classes.
- The *larger the margin, the lower is the generalization error* of the classifier.

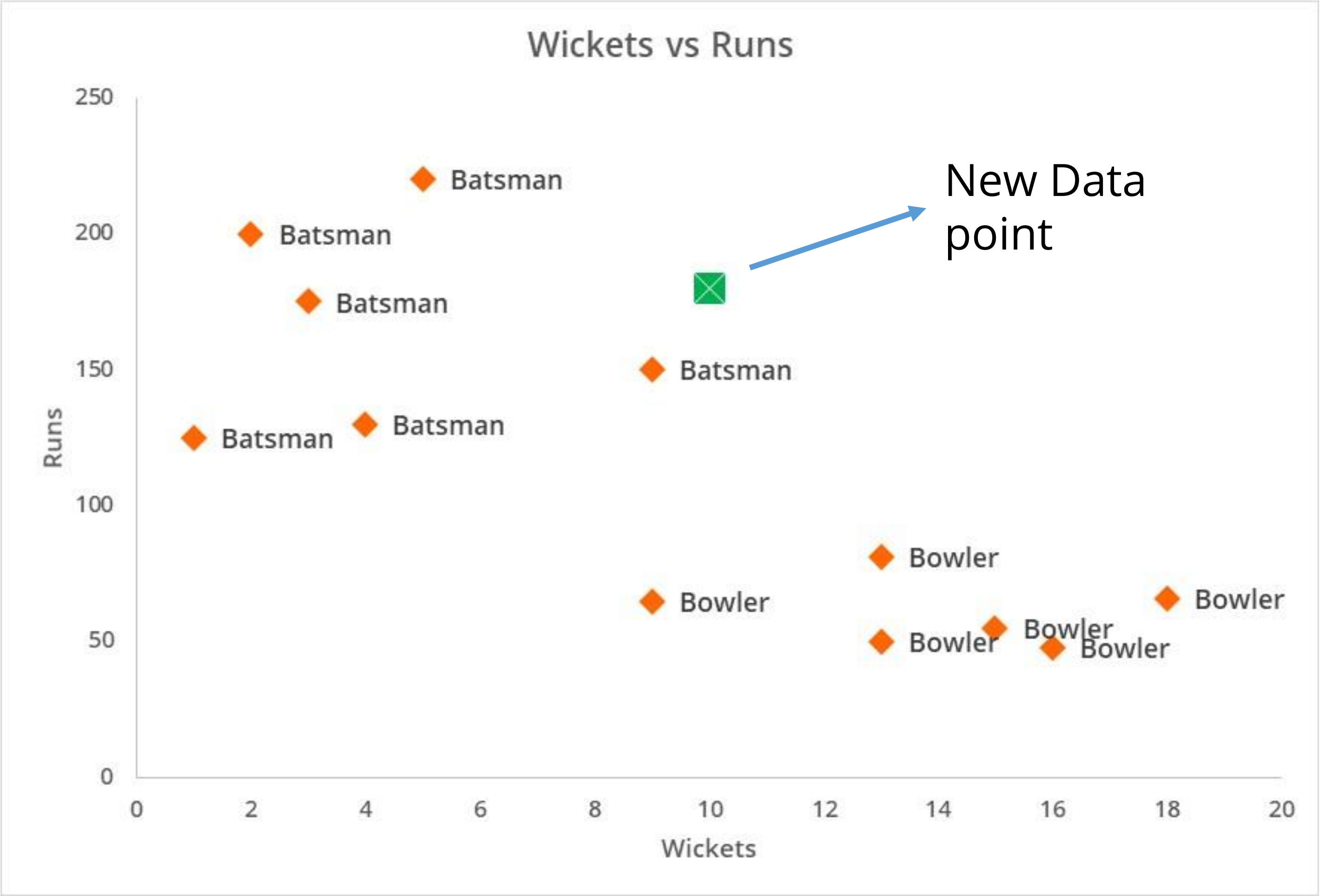
SVM - Working

Wickets	Runs	Player
1	125	Batsman
4	130	Batsman
9	150	Batsman
5	220	Batsman
2	200	Batsman
3	175	Batsman
9	65	Bowler
15	55	Bowler
16	48	Bowler
18	66	Bowler
13	50	Bowler
13	81	Bowler



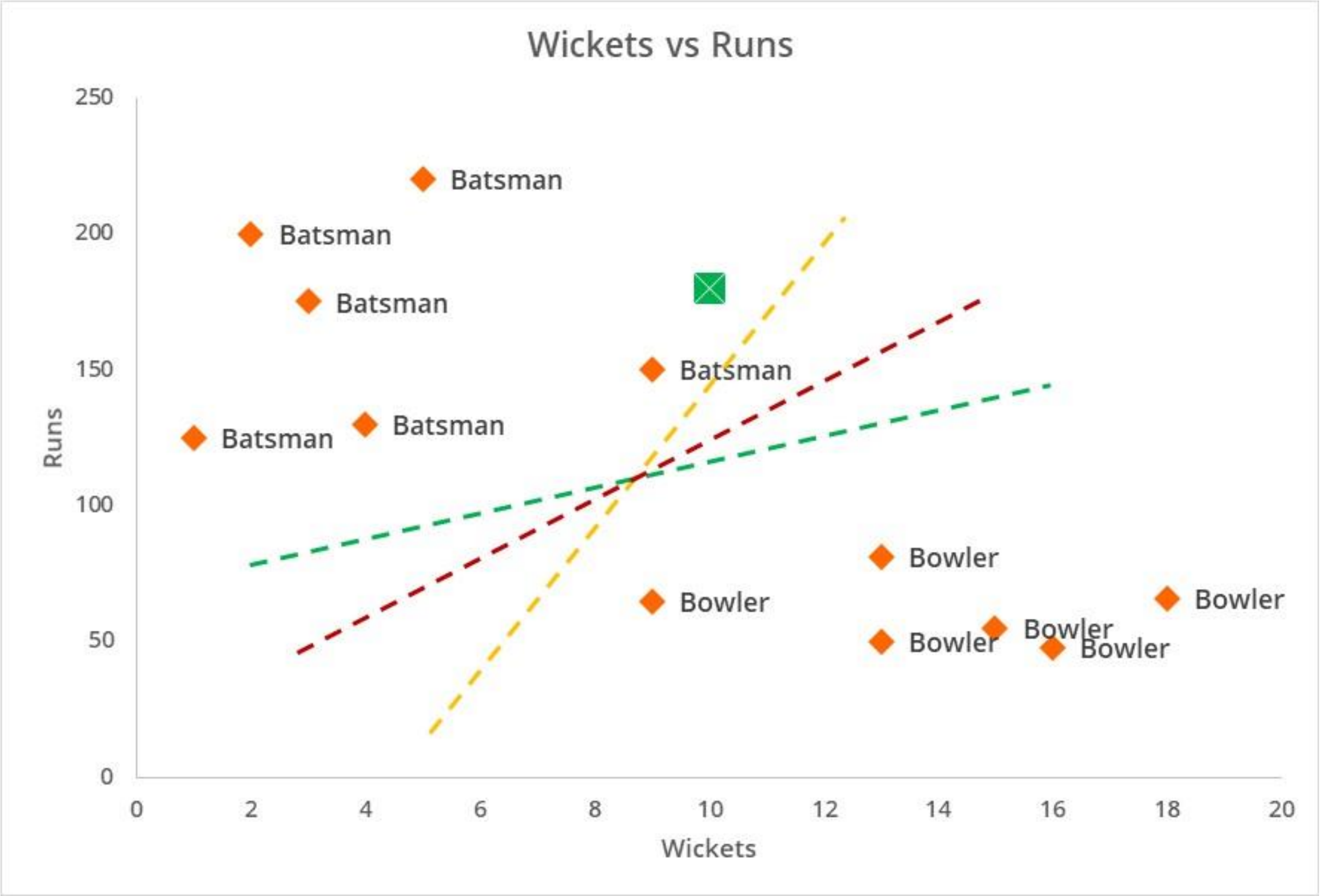
SVM - Working

- We want to classify the new data point as a batsman or a bowler.
- To achieve this, a **decision boundary** is required in order to classifty new data.



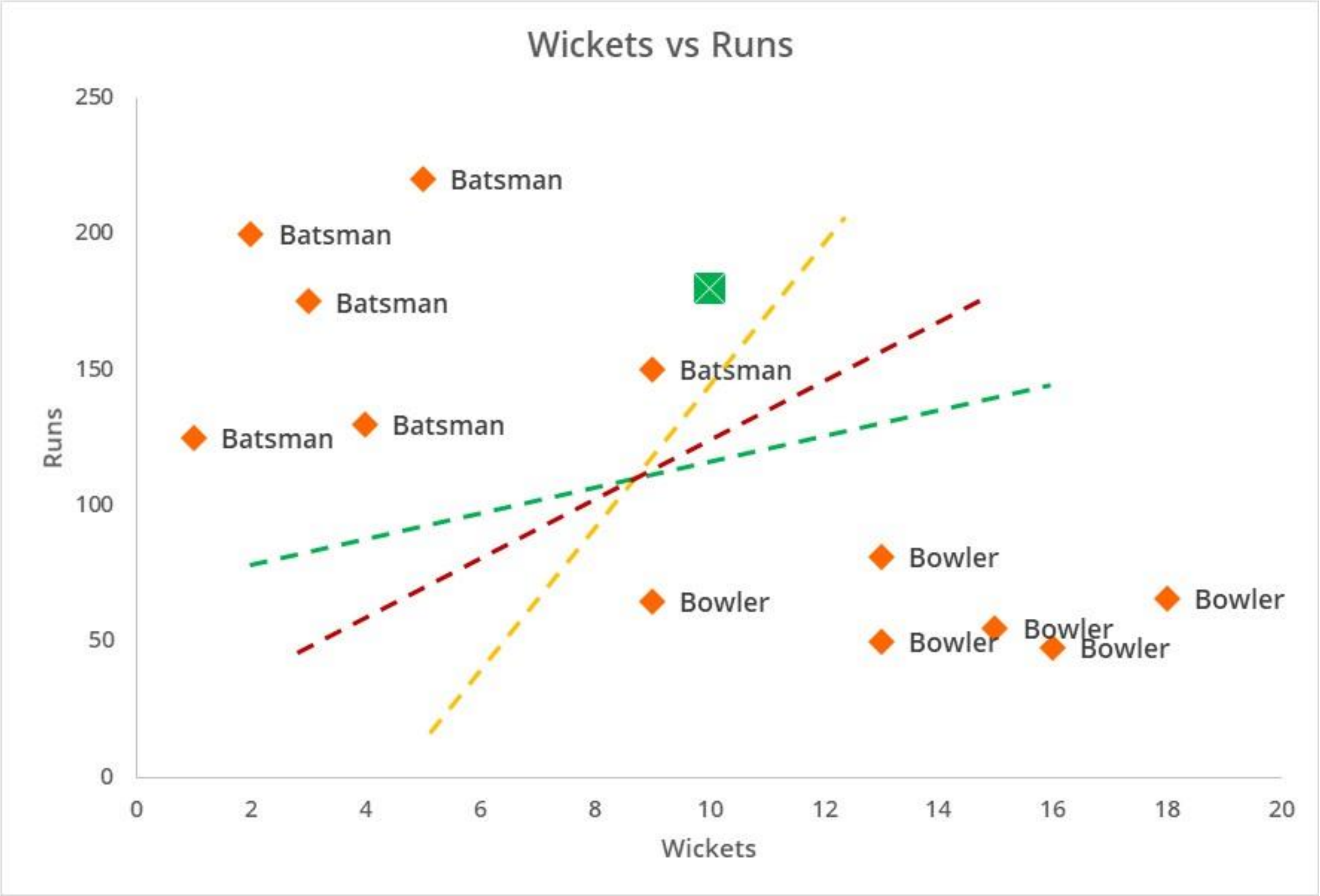
SVM - Working

- The decision boundary is separation between the two classes.
- Infinite number of lines can be drawn in the xy-space.
- This aspect is close to linear regression where we find the best fit line.



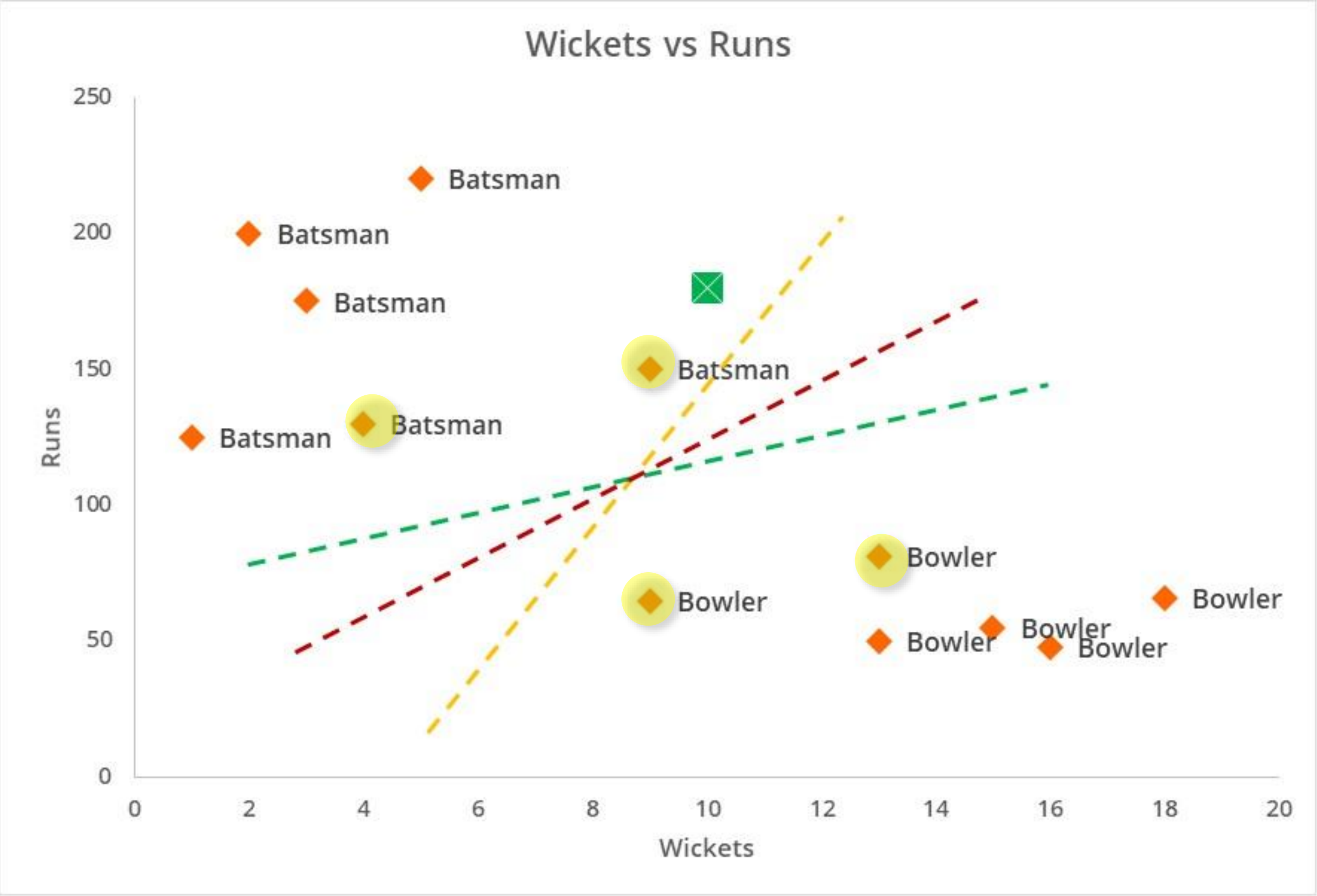
SVM - Working

- Out of these lines.. Which one to select?
- The class of new data point changes based on the line.
- The best line is selected by computing the **maximum margin** from equidistant **Support Vectors.**

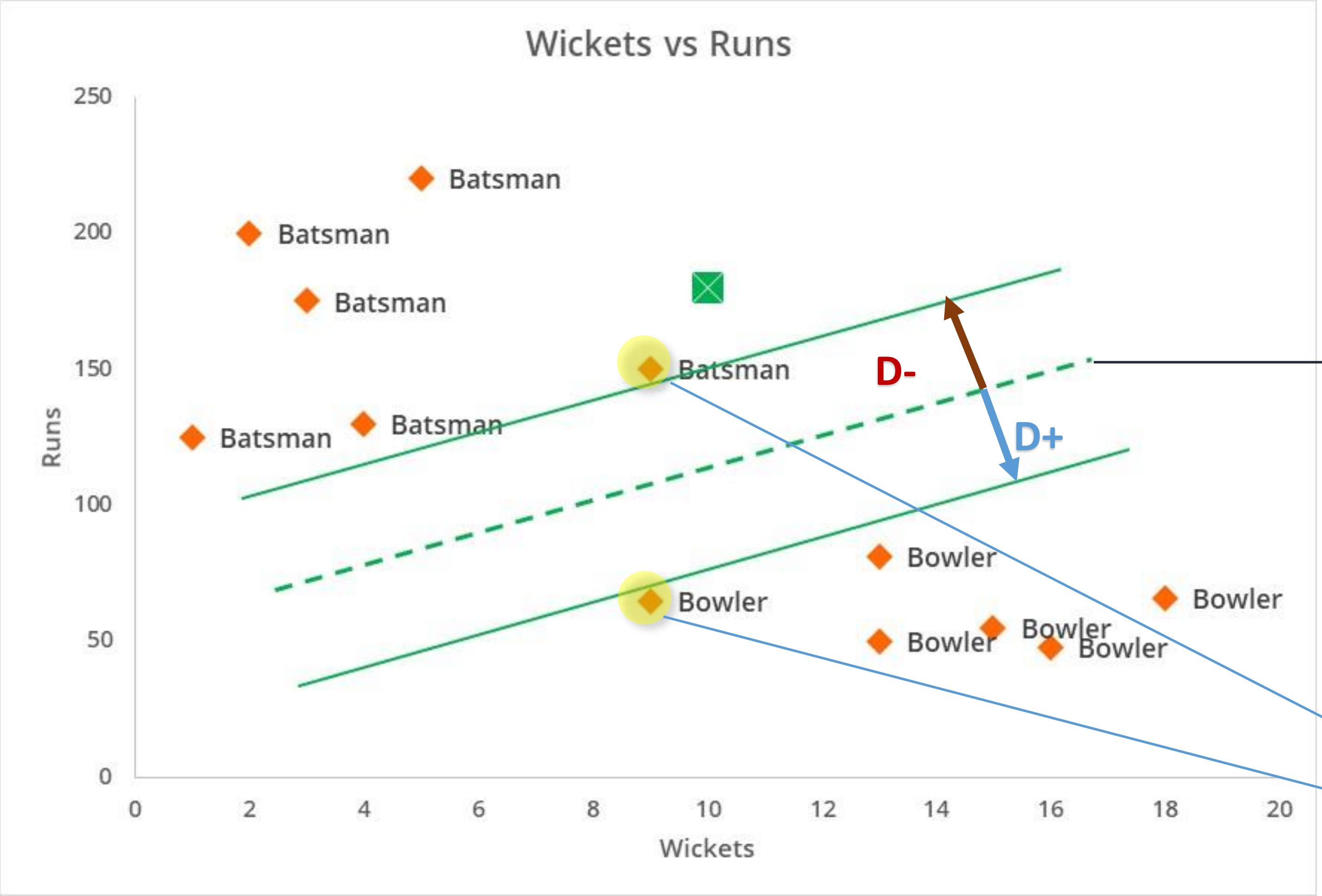


SVM - Working

- **Support vectors** – which are closest together and maximizes the distance between groups and support the algorithm.
- In simple terms, the points which are very close to the dividing line.
- Used to select the best dividing line.



SVM - Linear

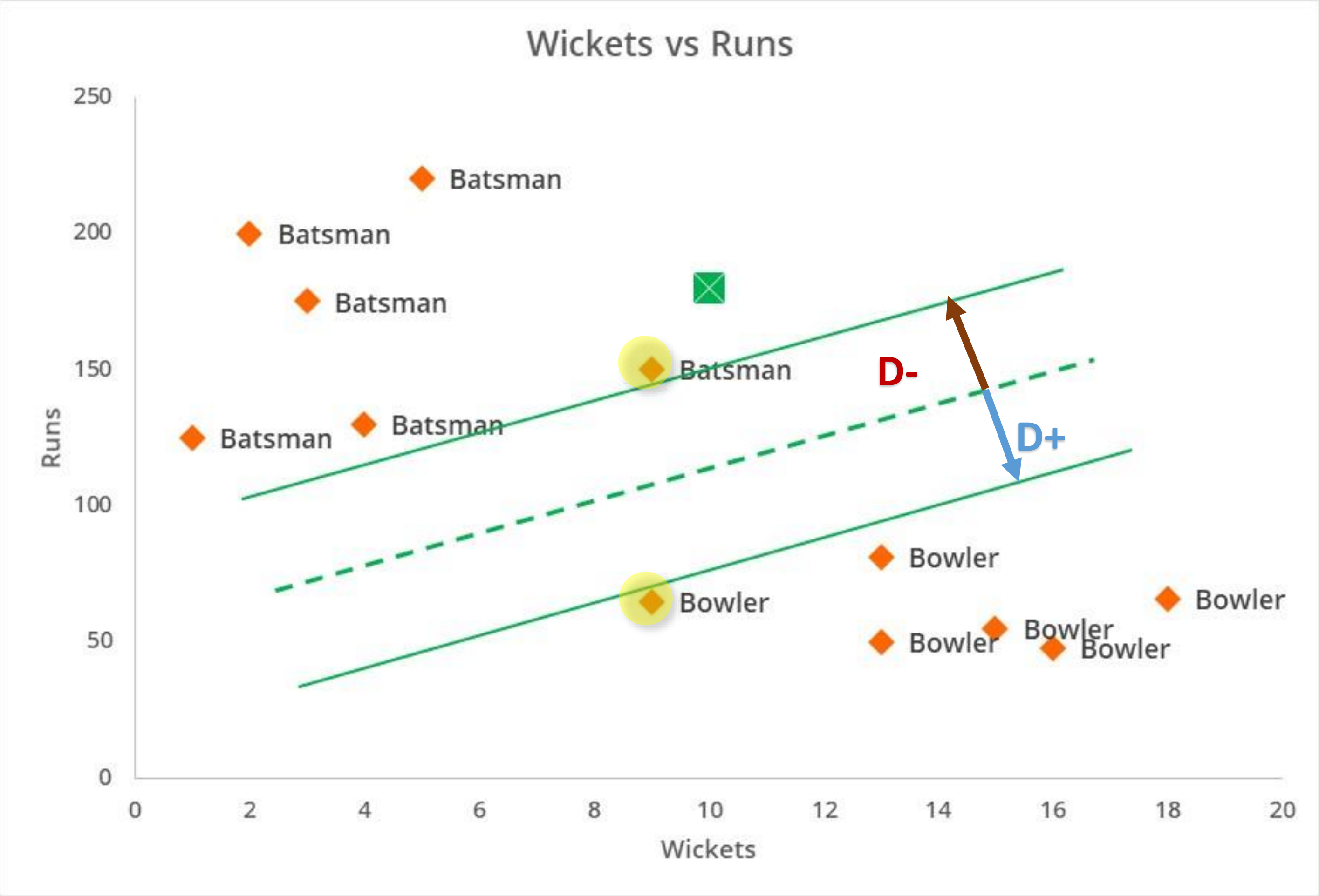


Hyperplane

- **D+** Closest distance to the closest positive point.
- **D-** Closest distance to the closest negative point.
- Sum of D+ & D- is called **Distance margin.**

Support Vectors

SVM - Linear



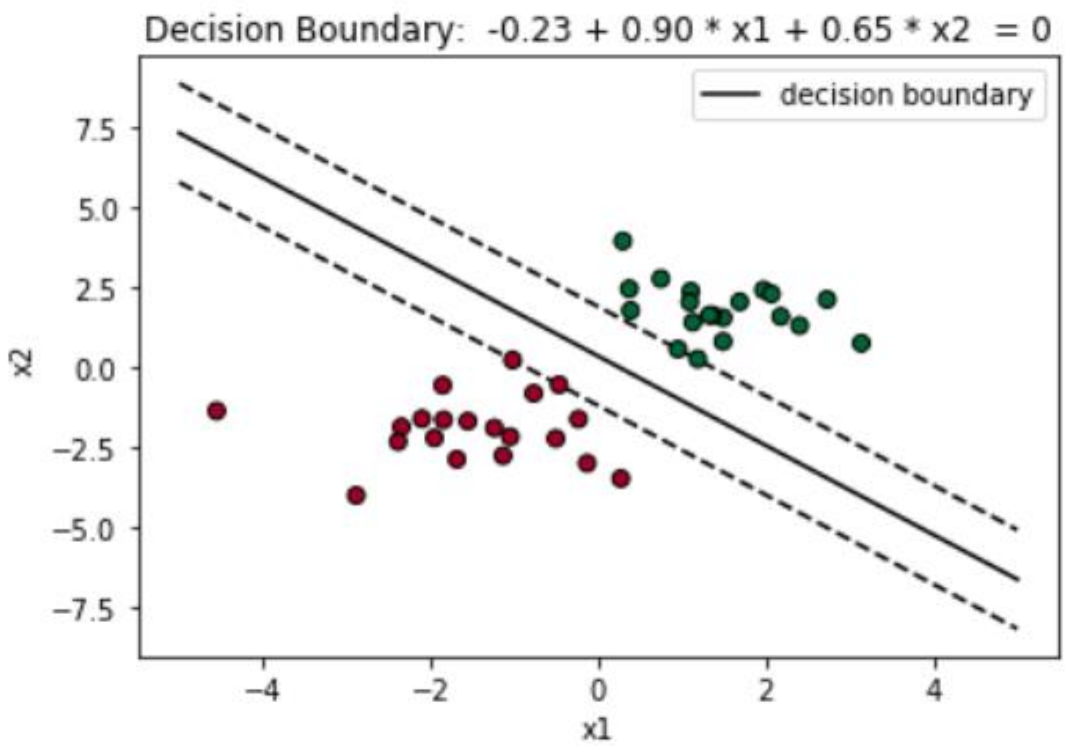
Any Hyperplane can be written mathematically as:

$$\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + ... + \beta_n * x_n = 0$$

For 2-dimensional space, the Hyperplane, which is the line, could be written as:

$$\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 = 0$$

SVM - Linear



The data points above this line can be written as:

$$\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 > 0$$

The data points below this line can be written as:

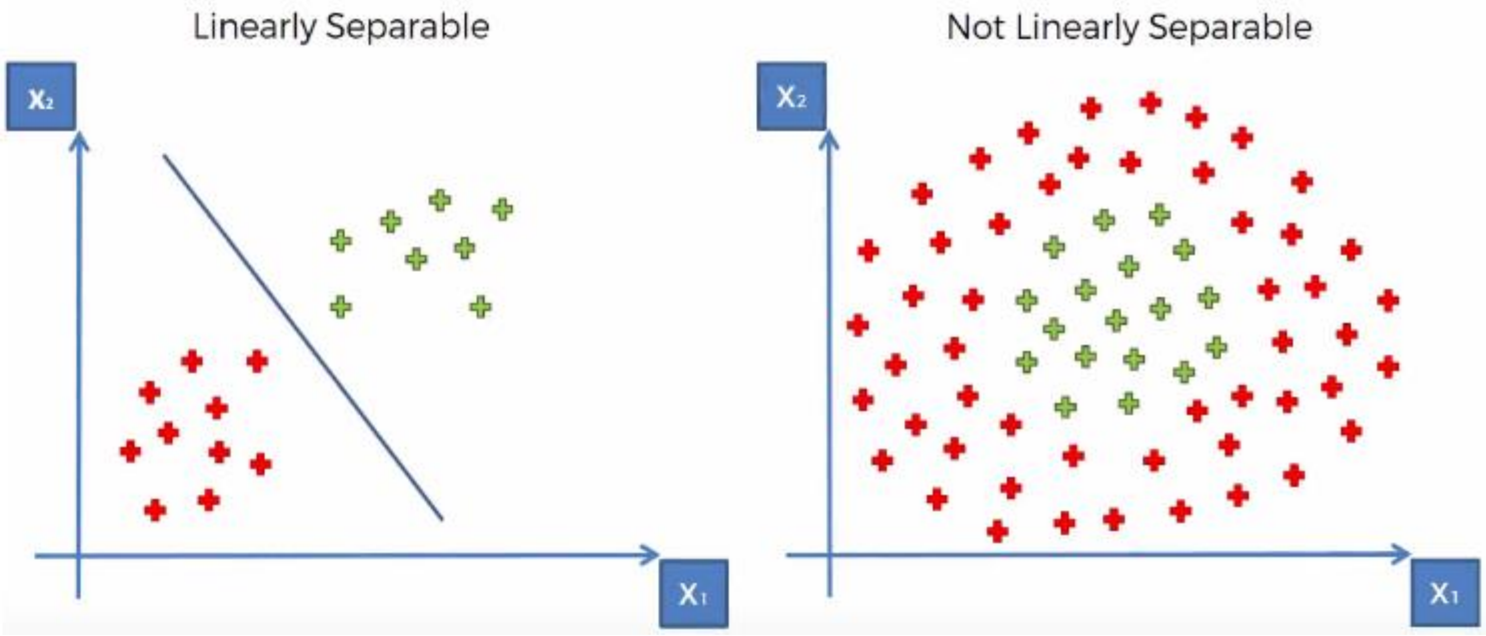
$$\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 < 0$$

```
from sklearn import svm
clf = svm.SVC(kernel='linear')
clf.fit(X, Y)
```

```
# the coefficient of the line
print('the coefficient of the line', clf.coef_)
```

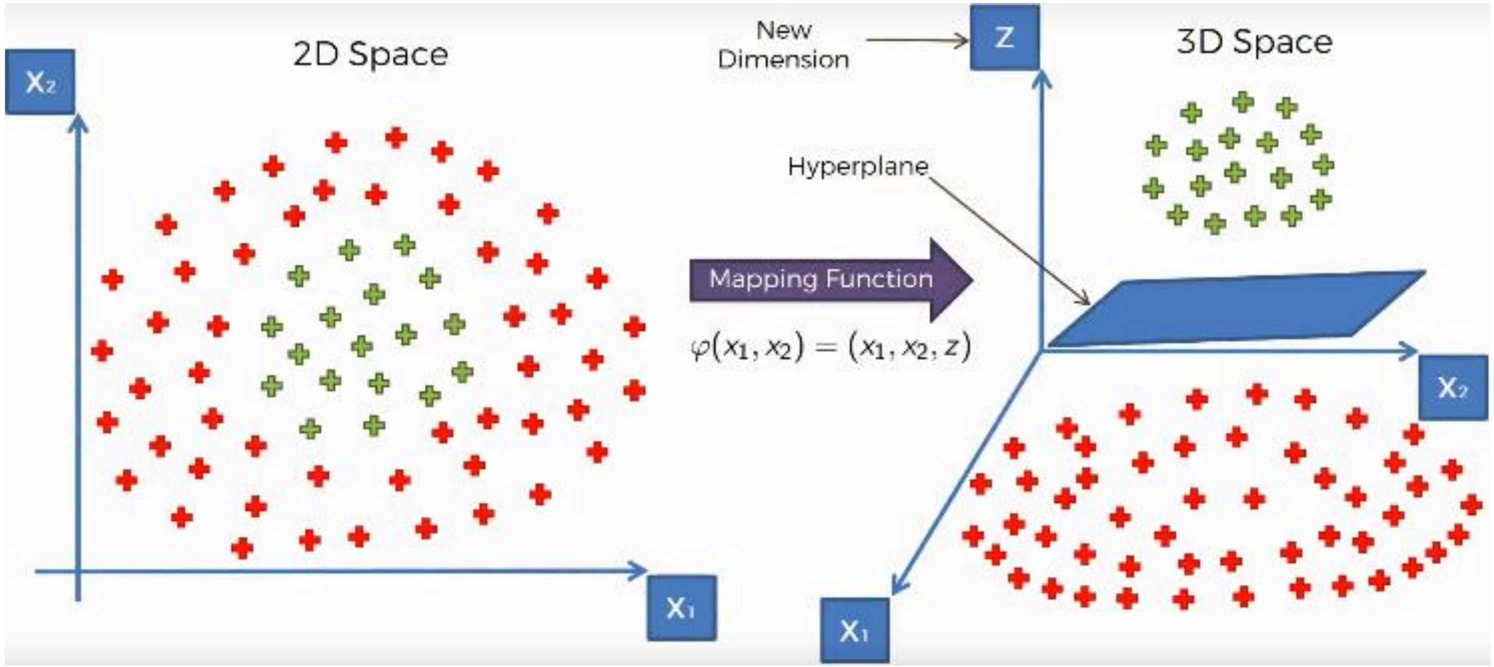
SVM - Kernel

- Consider the data sown in the image 2.
- What if our 2-d data looks like this.?
- This can not be separated by a hyperplane.



SVM - Kernel

- In these scenarios, SVM uses a Kernel function.
- This function transforms the data into higher dimension.
- Thereby enabling the option of fitting a plane to separate the classes.



Kernel functions

- Available kernel fucntions Sklearn (svm.SVC()) - `'linear'`, `'poly'`, `'rbf'`, `'sigmoid'`, `'precomputed‘`.
- The most used function is **RBF (Radial Basis Function).**
- It is a general-purpose kernel, used when there is no prior knowledge about the data.
- It is a transformer to generate new features by measuring the distance between all other dots to a specific dot/dots — centers.

Kernel functions

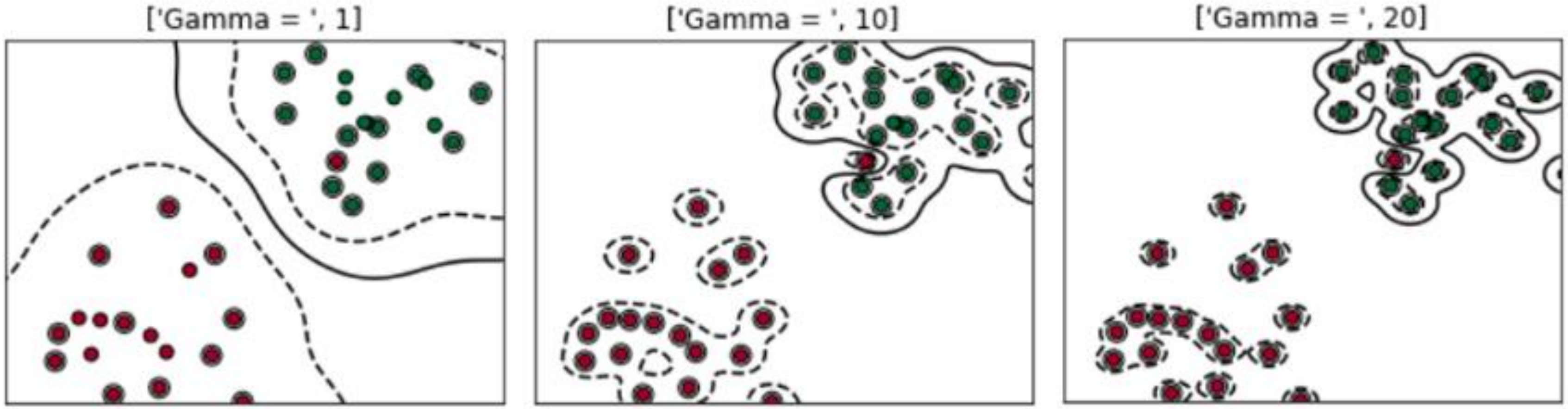
- The most popular/basic RBF kernel is the Gaussian Radial Basis Function:

$$\phi(x, center) = exp(-\gamma \|x - center\|^2)$$

- **gamma** controls the influence of new features — **Φ(x, center)** on the decision boundary.
- The higher the gamma, the more influence of the features will have on the decision boundary, more wiggling the boundary will be.
- Gamma is a hyperparameter that we can tune for SVM.

Kernel functions

- Effect of Gamma on the decision boundaries – (Example)



Pros & Cons

Pros:

- SVMs are effective when the number of features is quite large.
- It works effectively even if the number of features are greater than the number of samples.
- Non-Linear data can also be classified using customized hyperplanes built by using kernel trick.

Cons:

- The biggest limitation of Support Vector Machine is the choice of the kernel. The wrong choice of the kernel can lead to an increase in error percentage.
- SVMs have good generalization performance but they can be extremely slow in the test phase.
- SVMs have high algorithmic complexity and extensive memory requirements due to the use of quadratic programming.