Support Vector Machines and Decision Trees

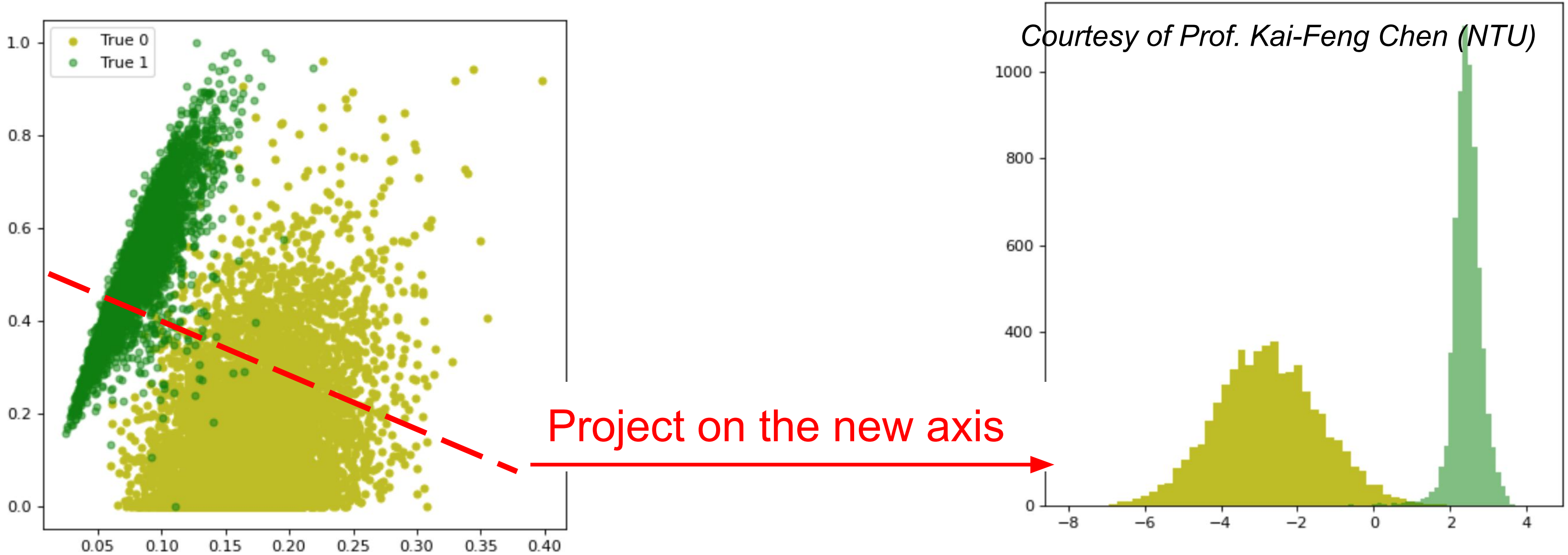
PHYS591000 2022.03.16

Warming up

- As usual, take 3 mins to introduce yourself to your teammate for this week!
  – “It’s been a month! How are you doing this semester?”
  – “What do you think we can do to make this week nice and easy?”

Review: Classification with LDA

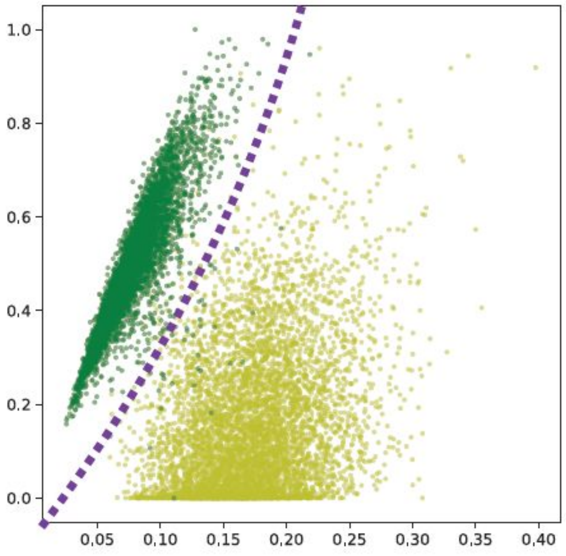
- Recall how LDA transforms the information on a 2D plane to the projection on a new axis that maximizes the separation:

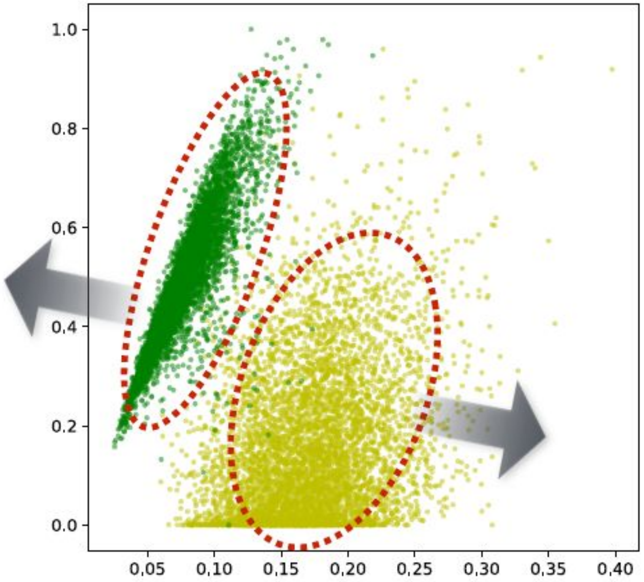


Motivation: Classification with a ‘border line’

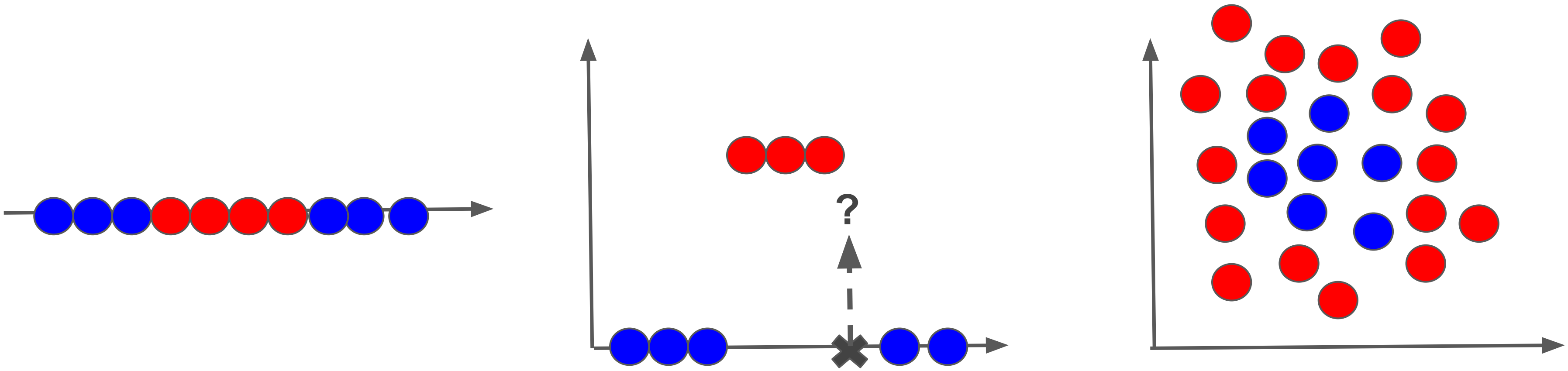
- Why don’t we just use a ‘border line’ to classify?
  – Support vector machines (SVM)

Courtesy of Prof. Kai-Feng Chen (NTU)





Motivation: Non-linear classification/regression

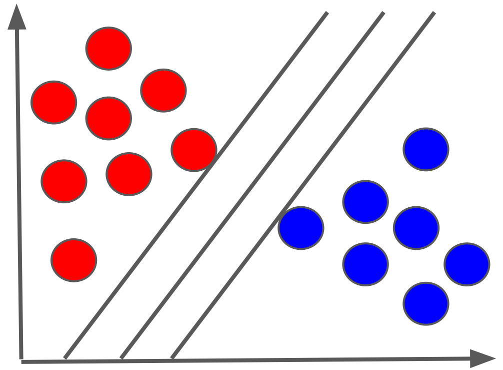


- LDA/linear hypothesis doesn’t seem to work…
- Besides SVM, Decision Trees and Random Forests are also popular algorithms that are applicable to non-linear cases.

Part I: Support Vector Machine

Classification: which line?

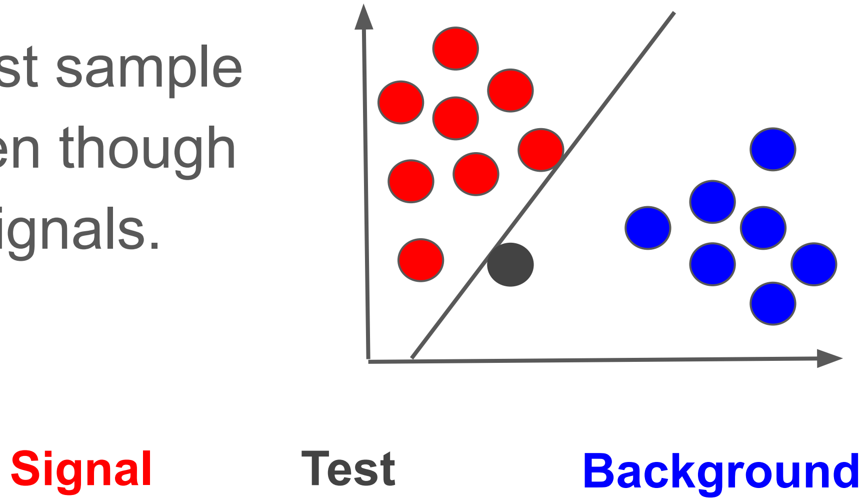
- All three lines can be used to classify signal (red) and background (blue) events. Which one to use?



Signal Background

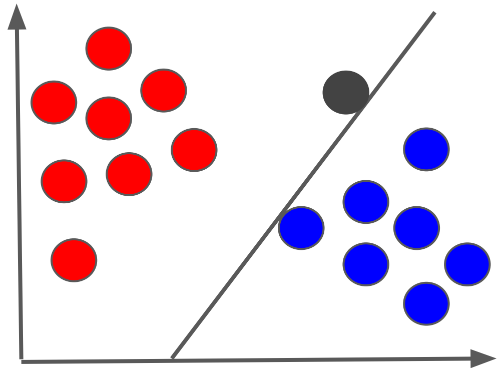
Classification: which line?

- This line will classify the test sample (black) as background, even though it is closer to the block of signals.



Classification: which line?

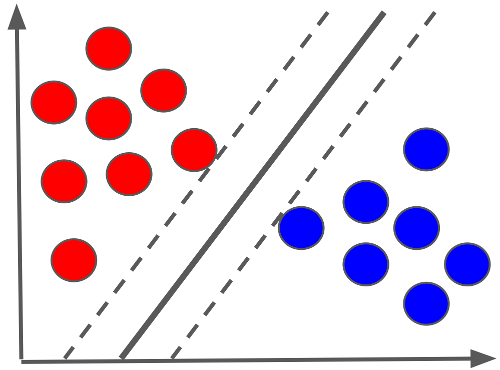
- Similarly the test sample will be classified as signal, despite being closer to the background events.



Signal Test Background

Classification: which line?

- Apparently this ‘central’ line is the best choice: the one with max **margin**
  - Margin: the shortest distance between the observations (dots) and the threshold (line)

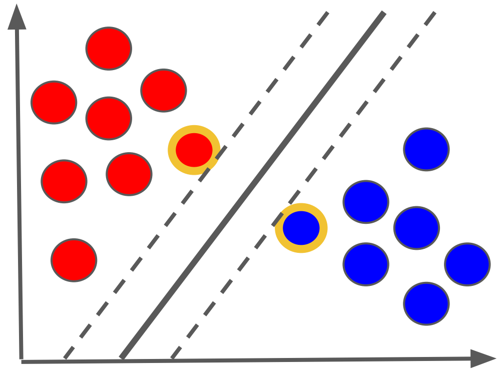


Signal

Background

Support Vector Machine (SVM)

- The max-margin hyperplane (when more features are used) is determined by the data points which lie nearest to it.
  → **Support vectors**

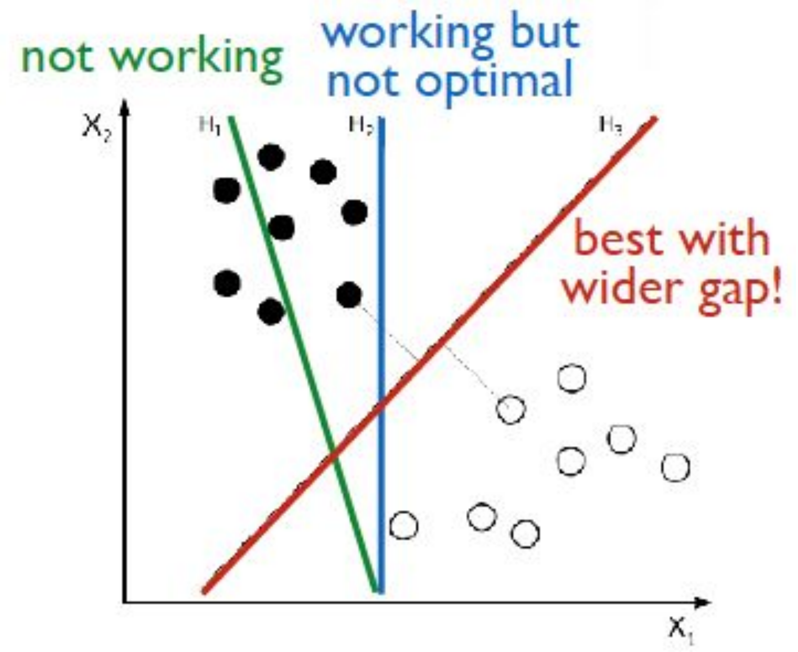


Signal **Background**

Support Vector Machine (SVM)

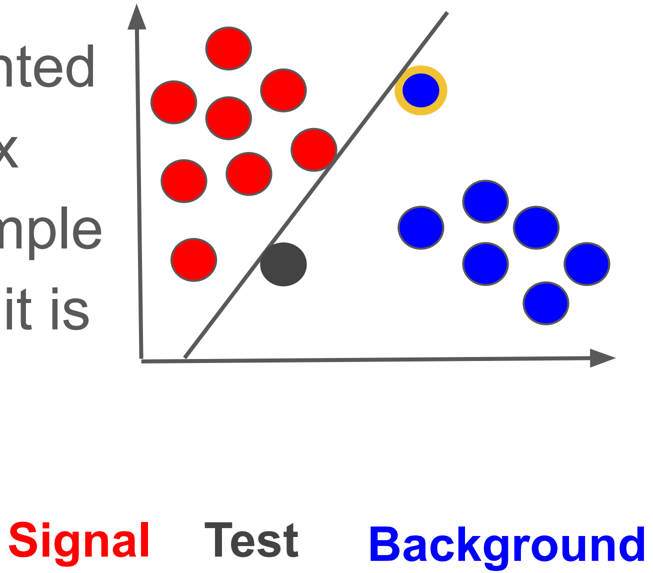
Courtesy of Prof. Kai-Feng Chen (NTU)

- The goal of SVM is to divide the categories with a gap (margin) as wide as possible



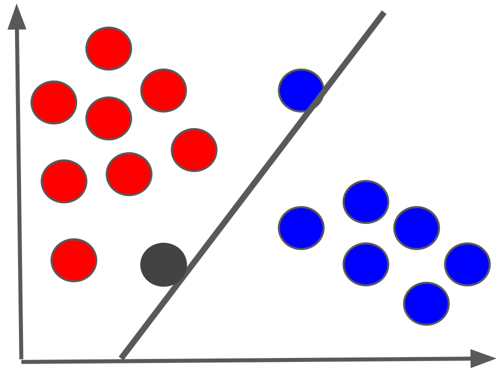
Soft margin

- When there are outliers (the highlighted blue dot), choosing the line with max *hard* margin will classify the test sample (black) as background even though it is closer to *most* of the signal events.



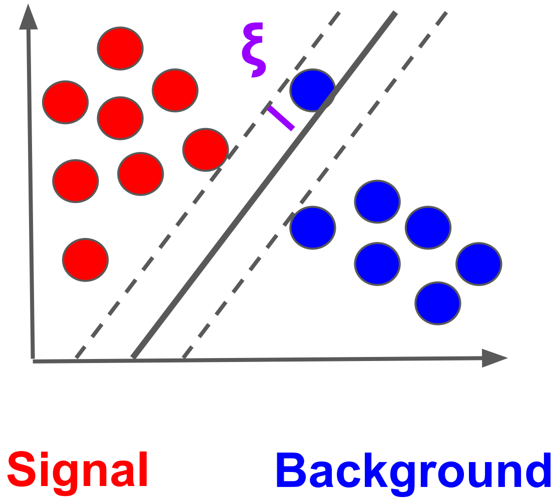
Soft margin

- Instead we choose the line with max **soft margin** by allowing a little bit misclassification, so that the model is less sensitive to outliers in the training data.



Soft margin

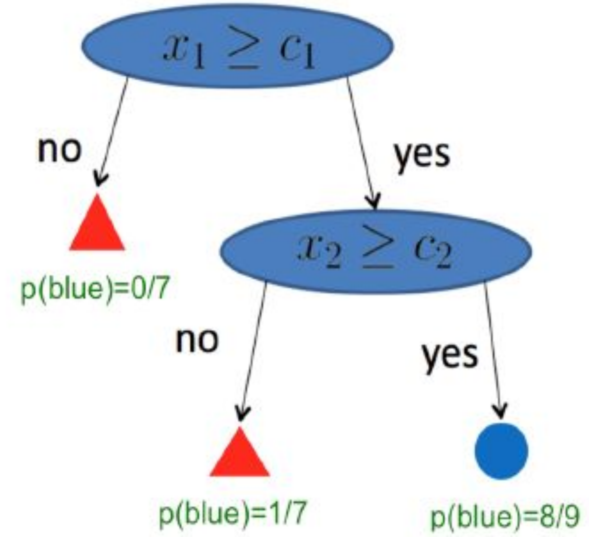
- The amount of misclassification can be represented by the distance **ξ**
- A ‘regularization’ term $C\Sigma_i \xi_i$ can be added to the SVM fit. C is a hyperparameter to be determined.

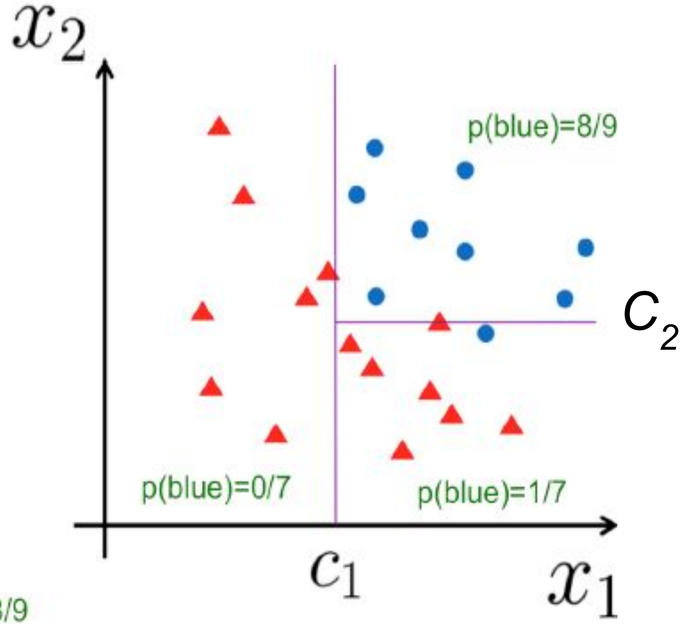


Part II: Decision Tree

Decision Tree

- Ask a series of Y/N questions and separate the data based on the answers

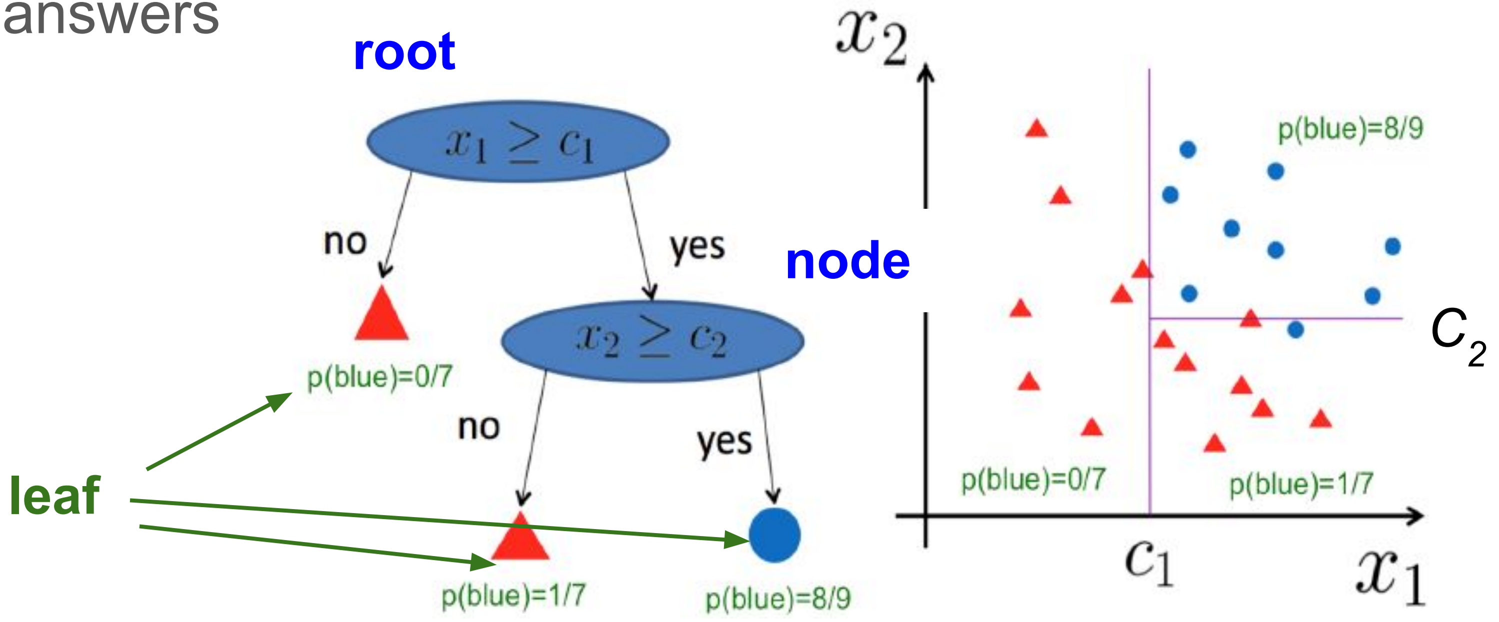




M. Kagan

Decision Tree

- Ask a series of Y/N questions and separate the data based on the answers



Decision Tree

- Easy to train but prone to overfitting (in principle can keep splitting until every training event is correctly classified)

- Should specify parameters such as
  - Max Tree depth
  - Min samples used in a node
  - Max number of leaves

  …

  by yourself

Hyperparameter Optimization

- There are many ‘free parameters’ when building a model: e.g. coefficient of the regularization term, number of neighbors in kNN, number of clusters in K-means → **hyperparameters**

- In most of the machine learning algorithms, hyperparameter optimization is a step need to be carried out to get the optimal performance.

Hyperparameter Optimization

- The tuning can be carried out by simply trying several reasonable (based on experience) setups, or make an exhaustive searching in the allowed parameter space.

  E.g.
  ```
  from sklearn.model_selection import GridSearchCV
  ```

Part III: Random Forests

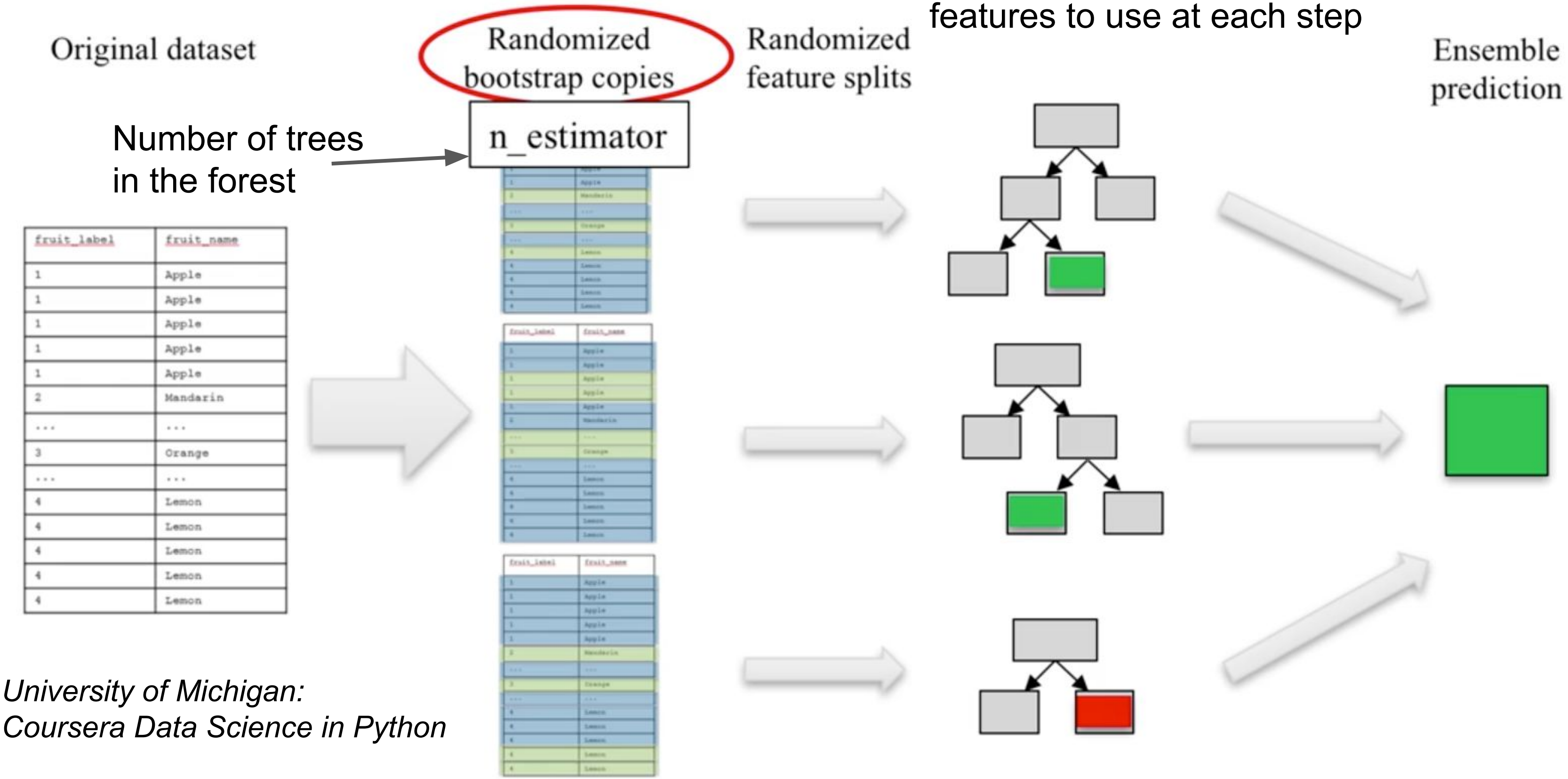
Ensembles of Decision Tree

- To make a decision tree flexible (easier to generalize) we can combine many trees and take the ‘average’ output:
  - Boosted Decision Trees: Train N models in sequence and give more weights to wrongly classified events each time.
    Output = weighted votes from each model
  - Random Forest: Train a set of trees by *randomly* selecting a subset of features and samples.
    Outcome = the output which gets the most ‘votes’ by the trees

Random Forests

Max_features: number of (the subset of) features to use at each step

Number of trees in the forest



University of Michigan:
Coursera Data Science in Python

In-class exercise

- For the in-class exercise this week we're going to use the MNIST dataset again!
  - 28*28 images; each pixel is associated with a number from 0-255 (~ the amount of 'ink')
  - We will start by using SVM, Decision Tree, and Random Forest to separate 0's and 1's with full/center average pixel densities.
  - And we'll inject all digits, 0-9, to the models (i.e. A 10-class classification!)

Lab for this week

- For the Lab this week, we'll use the same c2D project data as last week, and do
  - SVM
  - Decision Tree
  - Random Forest
- Then compare the results among the three and from last week (in the 'Discuss' session. Please elaborate on what you've *learned*, not just what you've observed.)

Backup

SVM: Nonlinear kernel

- SVM can be used for non-linear separation

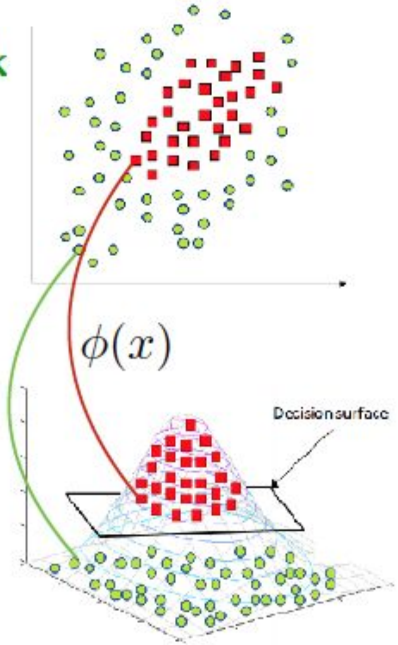
Courtesy of Prof. Kai-Feng Chen (NTU)

- The idea is to transform the data with a **kernel trick** and allows the algorithm to fit the margin hyperplane in a **transformed feature space**. The classifier finds a hyperplane in the transformed space, the plane can be be nonlinear in the original space. Some common kernels:
  - *Polynomial*

$$k(\vec{x}_i, \vec{x}_j) = (\gamma \vec{x}_i \cdot \vec{x}_j + \eta)^d$$

  - *Gaussian / Radial basis function (RBF)*

$$k(\vec{x}_i, \vec{x}_j) = \exp(-\gamma|\vec{x}_i - \vec{x}_j|^2)$$



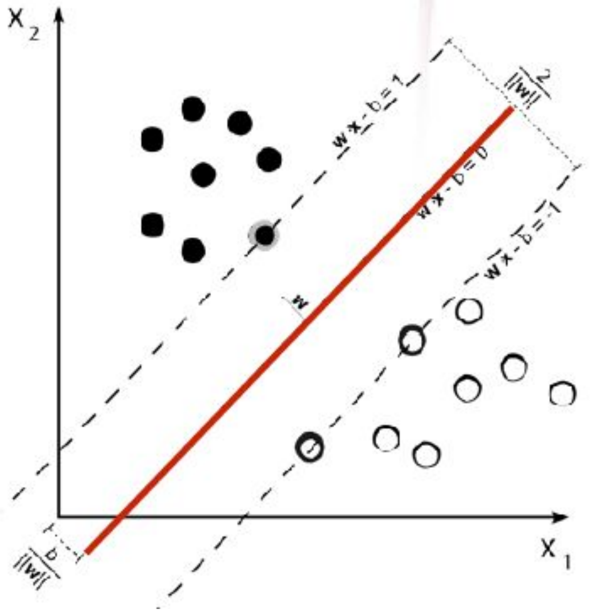
SVM hyperplane

Courtesy of Prof. Kai-Feng Chen (NTU)

- Consider a training data set of n points (*vectors*):

$$(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n) \quad \text{where } y_i = \pm 1$$

We want to find the "maximum-margin hyperplane" to separate the groups of ***y*=+1** and **–1**.



- A hyperplane can be expressed as

$$\vec{w} \cdot \vec{x} - b = 0$$

where w is the normal vector to the hyperplane, and the parameter $b/|w|$ determines the offset of the hyperplane from the origin.

Hard margin

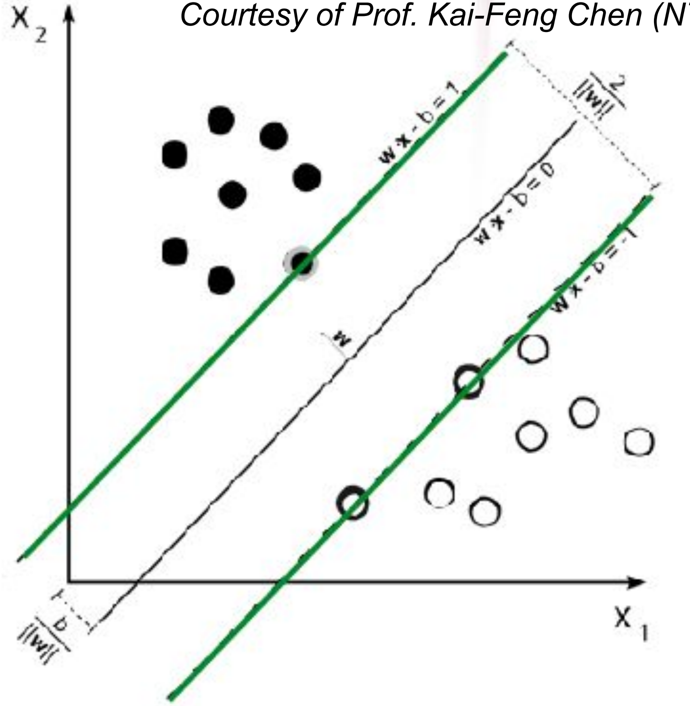
- These hyperplanes can be described by the following equations:

$$\vec{w} \cdot \vec{x} - b = \pm 1$$

- We have to prevent data points from falling into the margin, thus the following constraints apply:

$$\vec{w} \cdot \vec{x}_i - b \geq +1, \quad \text{if } y_i = +1$$
$$\vec{w} \cdot \vec{x}_i - b \leq -1, \quad \text{if } y_i = -1$$

Courtesy of Prof. Kai-Feng Chen (NTU)



Hard margin

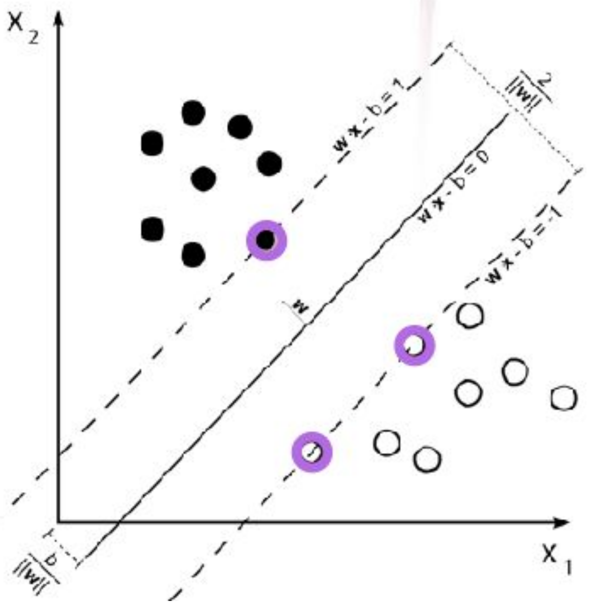
Courtesy of Prof. Kai-Feng Chen (NTU)

- The constraints imply each data point must lie on the correct side of the margin. One can put this together to formulate an optimization problem:

$$\text{Minimize } \frac{1}{2}|w|^2 \text{ subject to}$$

$$y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1$$

$$\text{for all } 1 \leq i \leq n$$



- A consequence of this geometric description is that the max-margin hyperplane is completely determined by those data points which lie nearest to it ⇒ **support vectors**.

Soft margin

- The max soft margin is determined by

$$\text{Minimize } \frac{1}{2}|w|^2 + C\sum_i \xi_i$$ subject to

$$\vec{w} \cdot \vec{x}_i - b \geq +1 - \xi_i$$ For black dots (y=1)

$$\vec{w} \cdot \vec{x}_i - b \leq -1 + \xi_i$$ For white dots (y=-1)

- C is a (regularization) hyperparameter