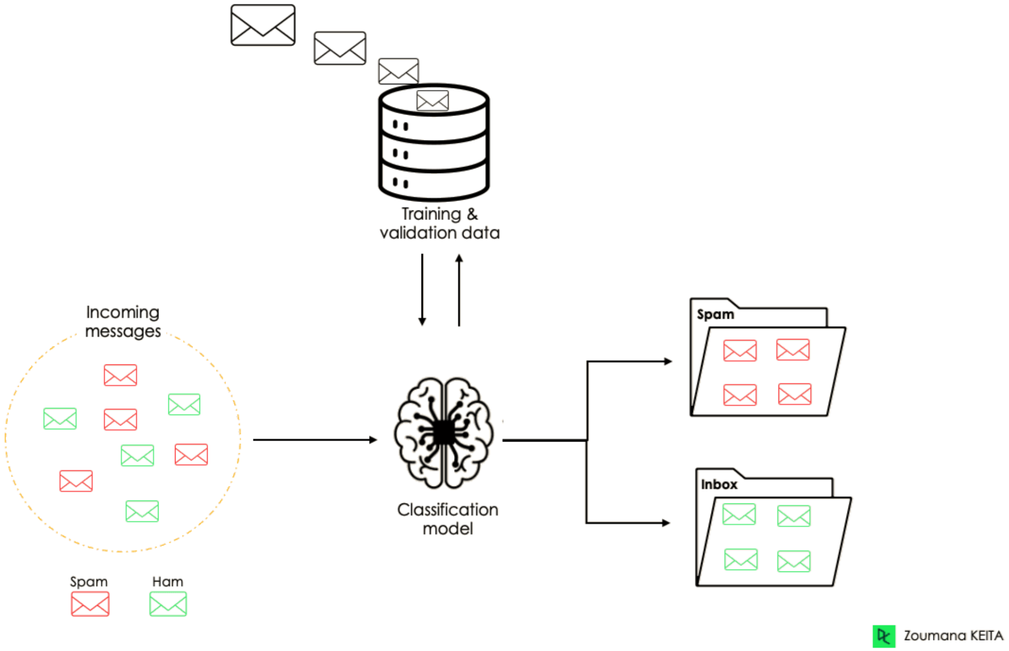
Classification

Dr Ossama Alshabrawy

ossama.alshabrawy@northumbria.ac.uk

What is Classification?

- Classification is a supervised machine learning method where the model tries to predict the correct label of a given input data.
- In classification, the model is fully trained using the training data, and then it is evaluated on vlidation data before being used to perform prediction on new unseen data.
- Type: Binary and Multi-Class Classification

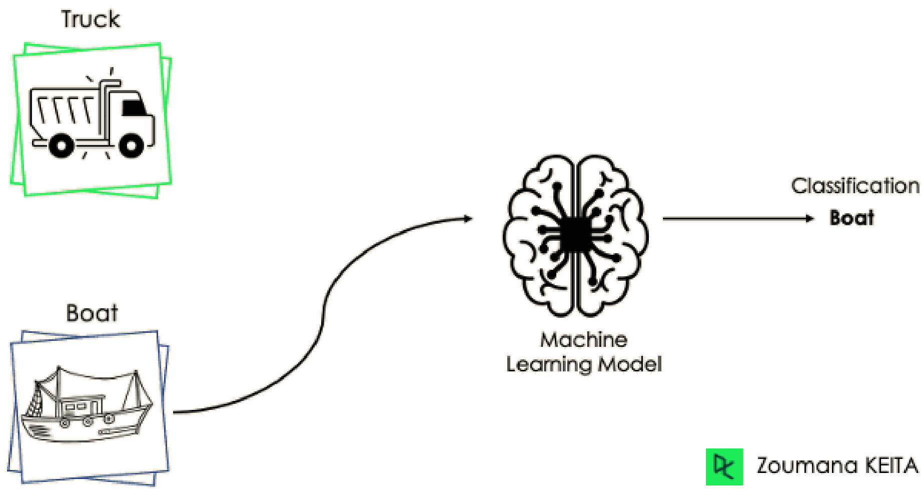


Model Construction, Validation and Testing

- **Model construction**
  - Each sample is assumed to belong to a predefined class (shown by the **class label**)
  - The set of samples used for model construction is **training set**
  - Model: Represented as decision trees, rules, mathematical formulas, or other forms
- **Model Validation and Testing**:
  - **Validation**: validation set is used to select, refine models or fine-tune hyperparameters of the model, it is called **validation** (or development)
  - **Test:** Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - *Accuracy:* % of test set samples that are correctly classified by the model
    - Test set is independent of training set and validation set.
- **Model Deployment:** If the accuracy is acceptable, use the model to classify new data
  - The ability to generalise

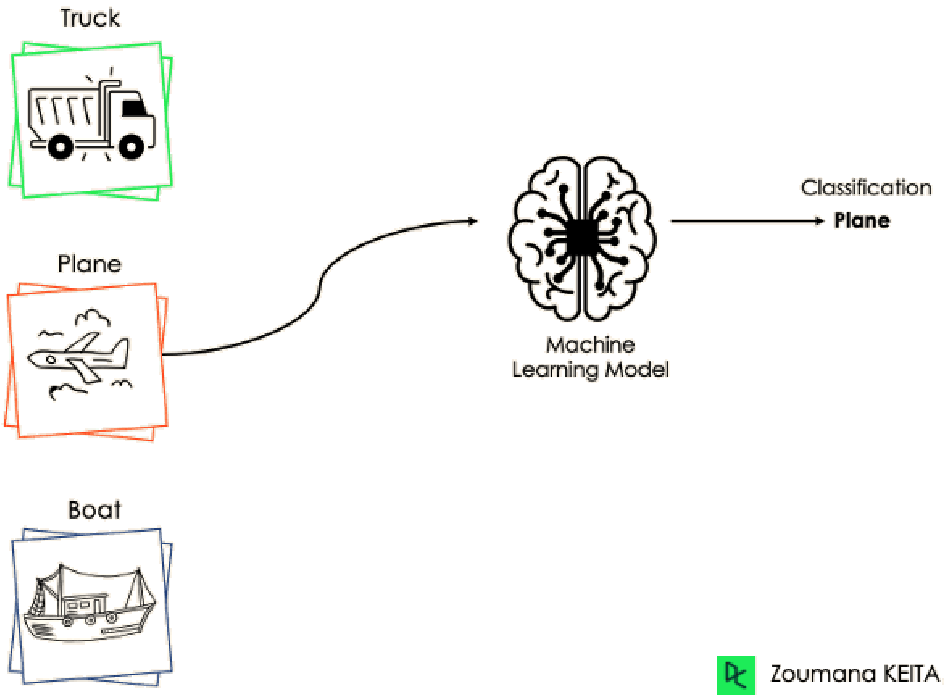
Binary Classification

- To classify the input data into two mutually exclusive categories.
  - The training data in such a situation is labelled in a binary format: true and false; positive and negative; 0 and 1; spam and not spam, etc. depending on the problem being tackled.



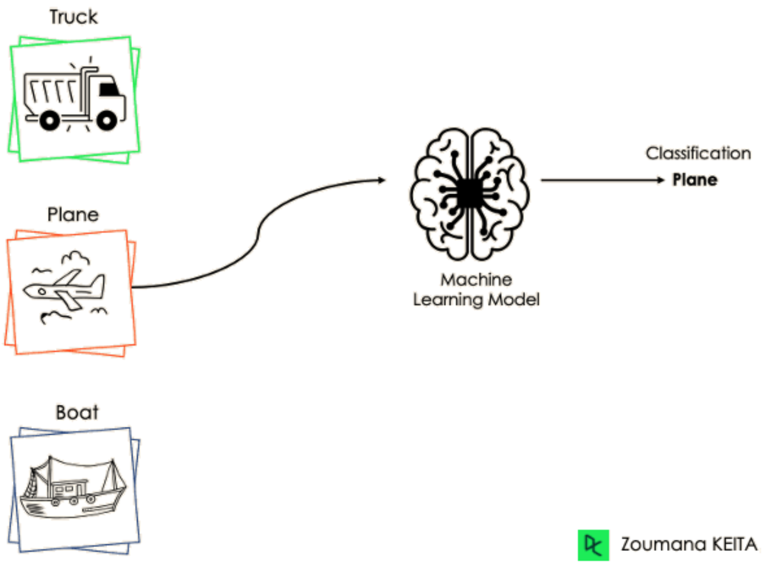
Multi-class Classification

The multi-class classification, on the other hand, has at least two mutually exclusive class labels, where the goal is to predict to which class a given input example belongs to.



Multi-Class Classification

- has at least two mutually exclusive class labels, where the goal is to predict to which class a given input example belongs to.
- Algorithms:
  - Logistic Regression.
  - Decision Trees
  - Random Forest
  - K-Nearest Neighbours
  - SVM
  - Naive Bayes
  - Gradient Boosting



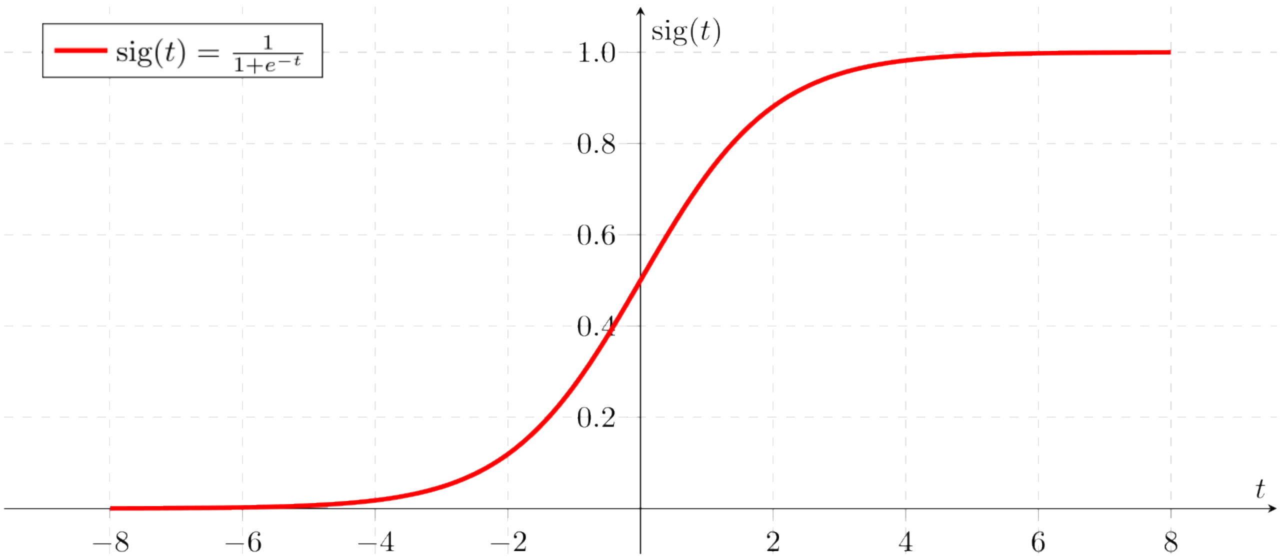
Logistic regression

- Linear regression assumption:

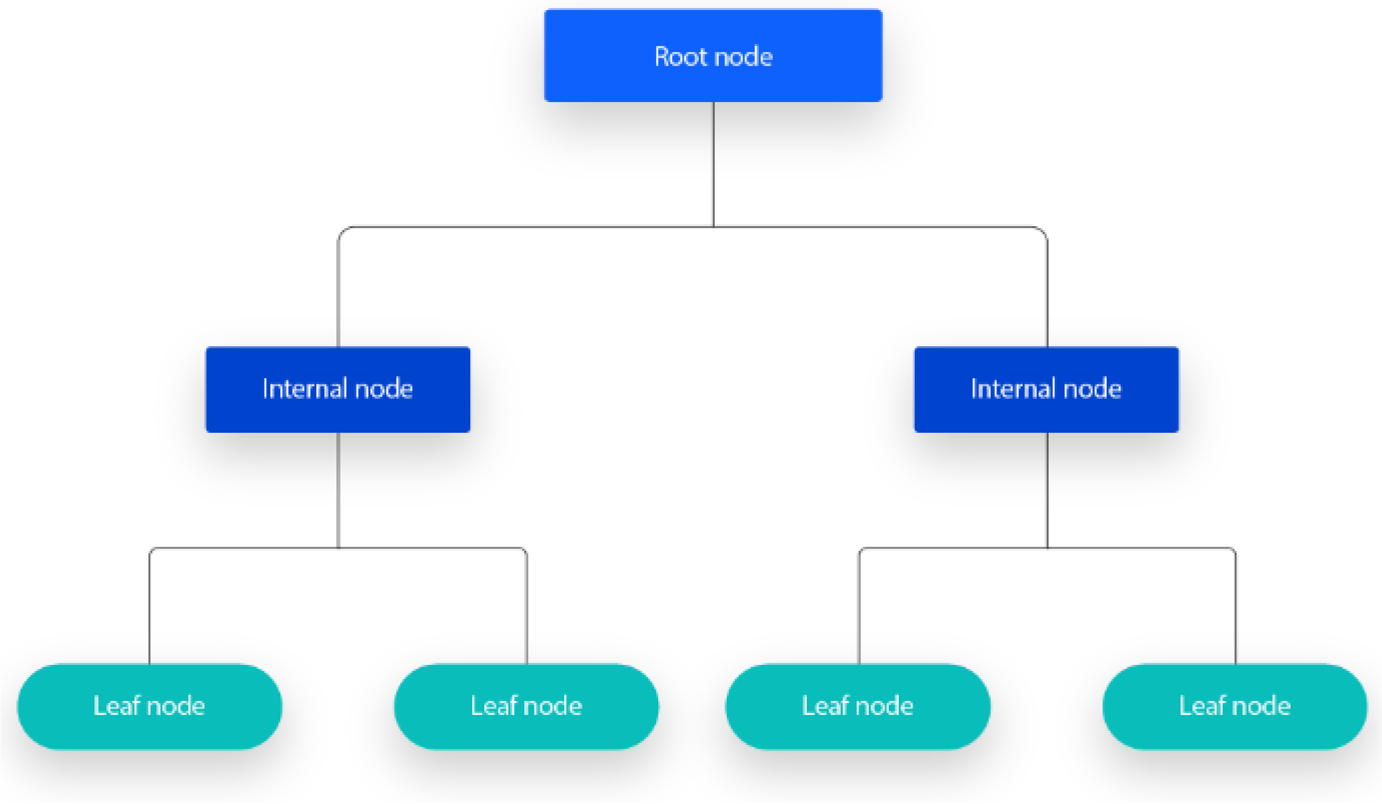
$$f_{\theta}(\boldsymbol{x}) = \boldsymbol{\theta_0} + \boldsymbol{\theta_1 x_1} + \boldsymbol{\theta_2 x_2} + \ \ldots + \boldsymbol{\theta_d x_d}$$

- Logistic regression assumption: $sigmoid(f_{\Theta}(x))$

- Output [0,1]



Decision Trees



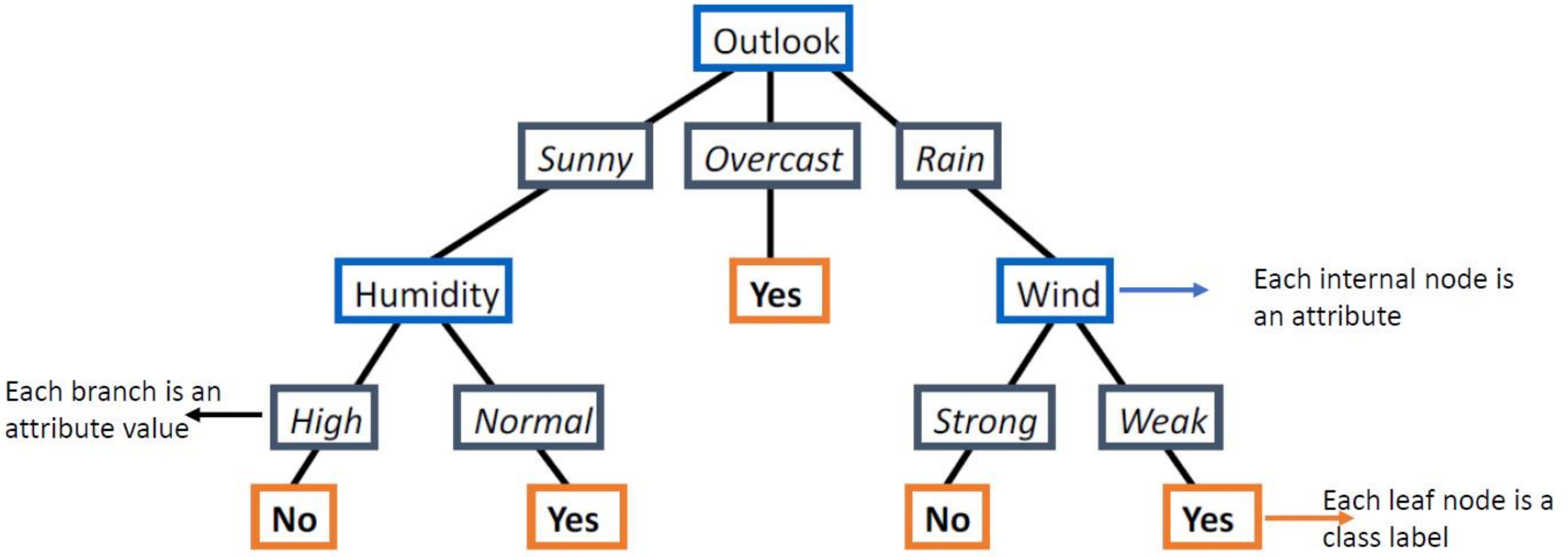
Decision Trees

Decision Trees

- Another example: If you want to play tennis and the possibilities of play
- Play decision:
  - Outlook=sunny && humidity=normal
  - Outlook=overcast
  - Outlook=rainy && wind=weak
  - Otherwise, no play

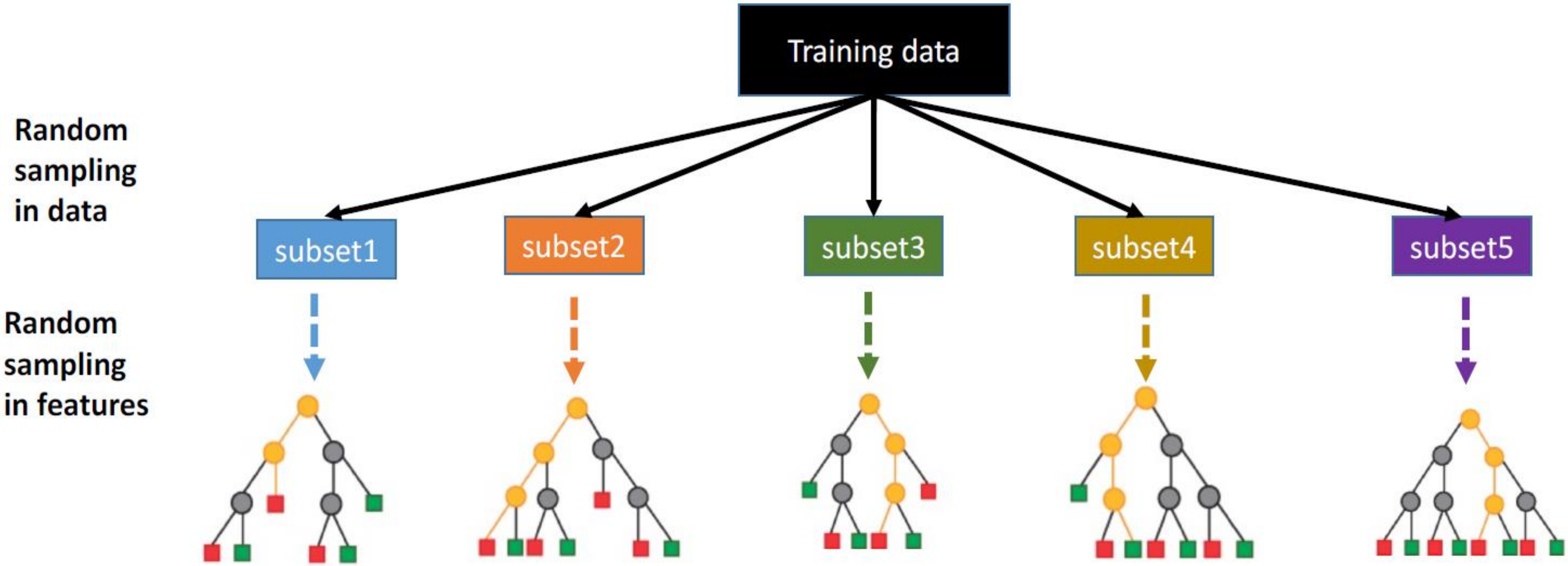
Attribute	Value
Outlook	Sunny, Overcast, Rain
Humidity	High, Normal
Wind	Strong, Weak
Temperature	Hot, Mild, Cool

Example—Decision Tree

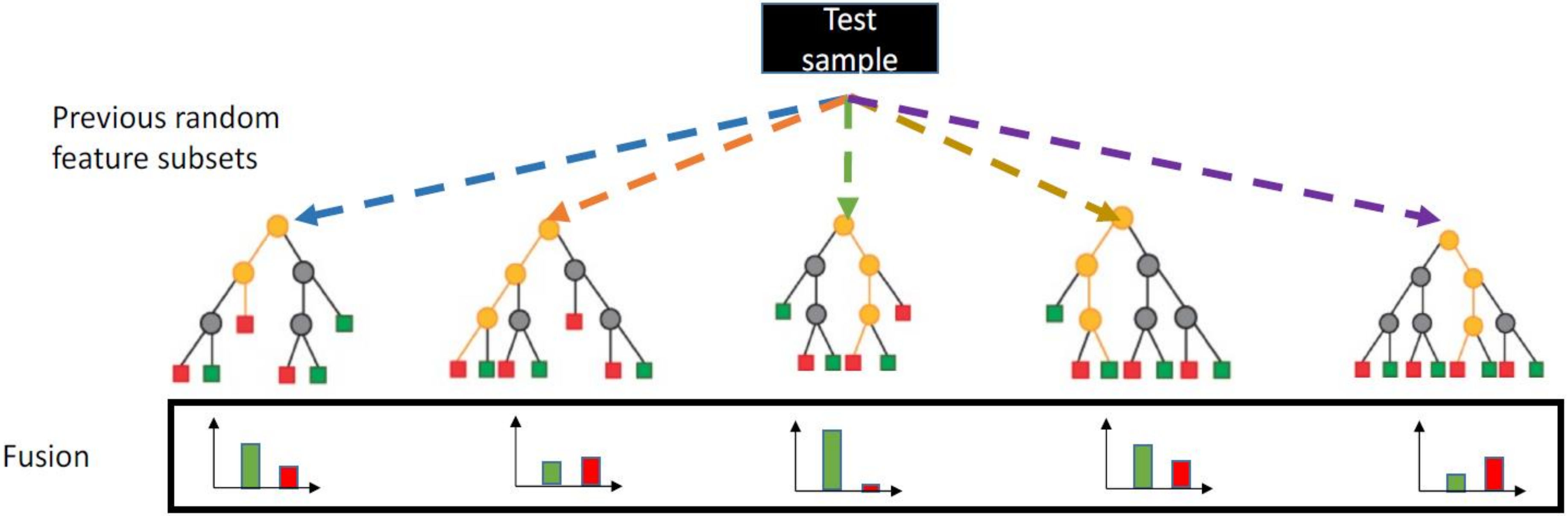


From Decision Tree to Random Forests (RF)

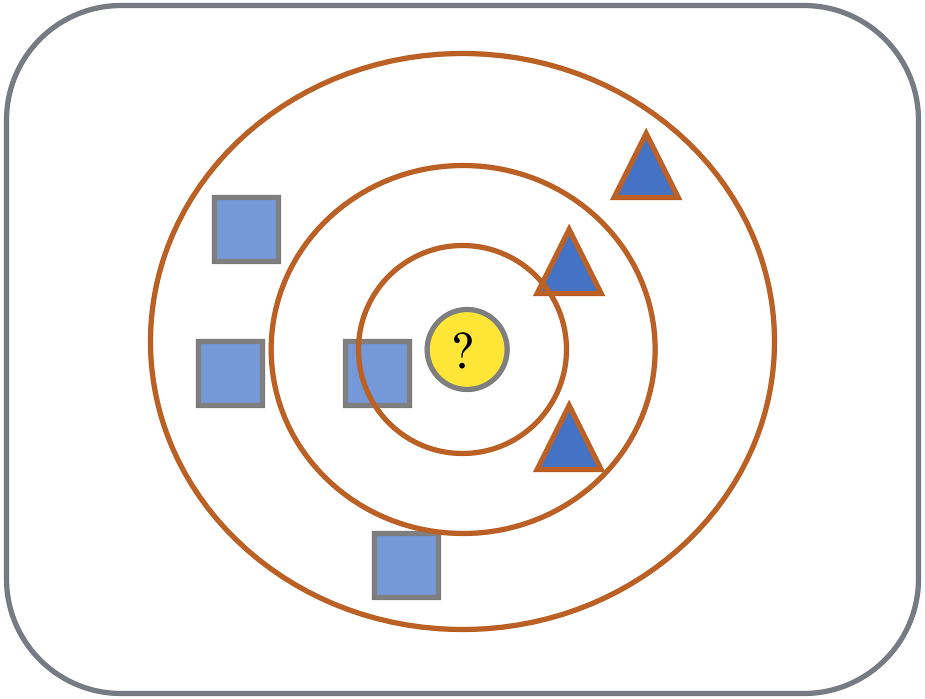
- Ensemble method
- Contains multiple classifier (decision trees)
- For each decision trees: Random sampling in features
- Random sampling in data (subset -> minibatch)



From Decision Tree to Random Forests (RF)



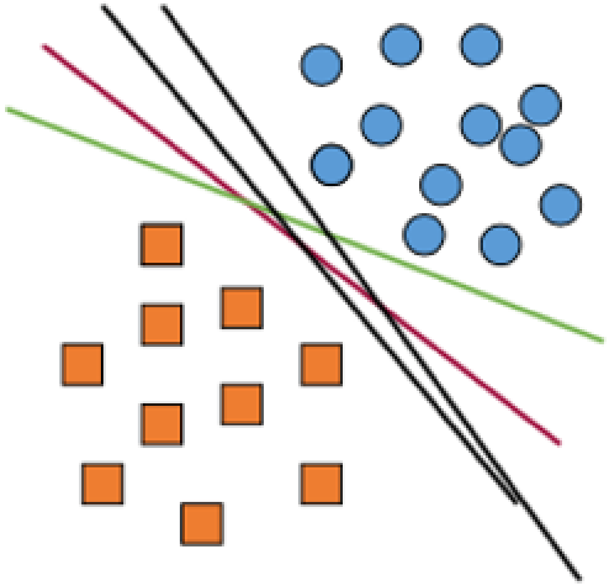
k Nearest Neighbor



- $k = 1$:
  - Belongs to square class
- $k = 3$:
  - Belongs to triangle class
- $k = 7$:
  - Belongs to square class

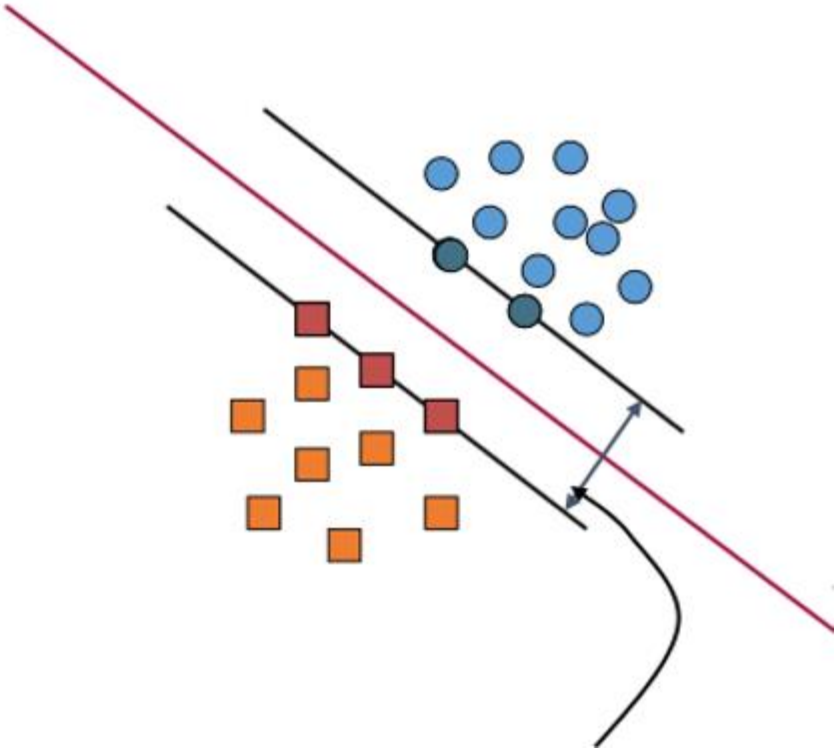
- Choosing the value of *k*:
  - If *k* is too small, sensitive to noise points
  - If *k* is too large, neighborhood may include points from other classes
  - Choose an odd value for *k*, to eliminate ties

Decision boundary for linearly separable data



- Which line is better?

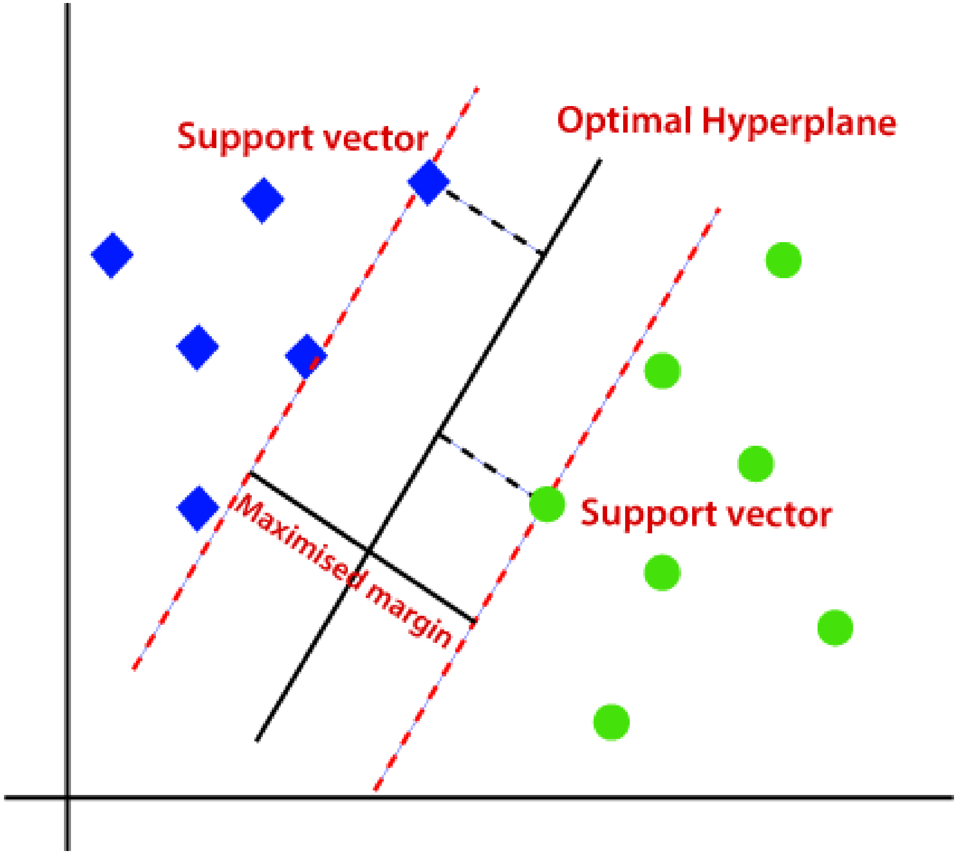
Decision boundary for linearly separable data



- The line with the largest margin

Support Vector Machines

- Known as the large margin classifier.



Model Evaluation

- Evaluation metrics
  - How can we measure accuracy?
  - Other metrics to consider?
- Use **validation or test set** of class-labeled tuples instead of training set when assessing accuracy
- Methods for estimating a classifier’s accuracy
  - Train-test method
  - Cross-validation

Confusion Matrix

- **Confusion Matrix:**

Actual class\Predicted class	C_1	$\neg C_1$
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

- In a confusion matrix w. *m* classes, $CM_{i,j}$ indicates # of tuples in class *i* that were labeled by the classifier as class *j*
  - May have extra rows/columns to provide totals
- **Example of Confusion Matrix:**

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

Accuracy, Error Rate, Sensitivity and Specificity

- **Classifier accuracy,** or recognition rate
  - Percentage of test set tuples that are correctly classified

  Accuracy = (TP + TN)/All

- **Error rate:** *1 – accuracy*, or

  Error rate = (FP + FN)/All

A\P	C (positive)	¬C (negative)	
C (positive)	**TP**	**FN**	**P**
¬C (negative)	**FP**	**TN**	**N**
	P’	**N’**	**All**

Precision and Recall, and F-measures

- **Precision**: Exactness: what % of tuples that the classifier labeled as positive are actually positive?
  - P = Precision = TP/(TP+FP)
- **Recall:** Completeness: what % of positive tuples did the classifier label as positive?
  - R = Recall = TP/(TP+FN)
- ***F* measure (**or ***F1*-score):** harmonic mean of precision and recall
  - F1 = 2P*R/(P+R)

Example

- Use the same confusion matrix, calculate the measure just introduced

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	90	**210**	300	30.00 (*sensitivity*)
cancer = no	**140**	9560	9700	98.56 (*specificity*)
Total	230	9770	10000	96.50 (*accuracy*)

- Accuracy = (TP + TN)/All = (90+9560)/10000 = 96.50%
- Precision = TP/(TP + FP) = 90/(90 + 140) = 90/230 = 39.13%
- Recall = TP/ (TP + FN) = 90/(90 + 210) = 90/300 = 30.00%
- F1 = 2 P × R /(P + R) = 2 × 39.13% × 30.00%/(39.13% + 30%) = 33.96%

Holdout & Cross-Validation

- **Holdout method**
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Repeated random sub-sampling validation: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained
- **Cross-validation** (*k*-fold, where k = 10 is most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use D_i as test set and others as training set
  - Leave-one-out: *k* folds where *k* = # of tuples, for small sized data
  - ***Stratified cross-validation***: folds are stratified so that class distribution, in each fold is approximately the same as that in the initial data