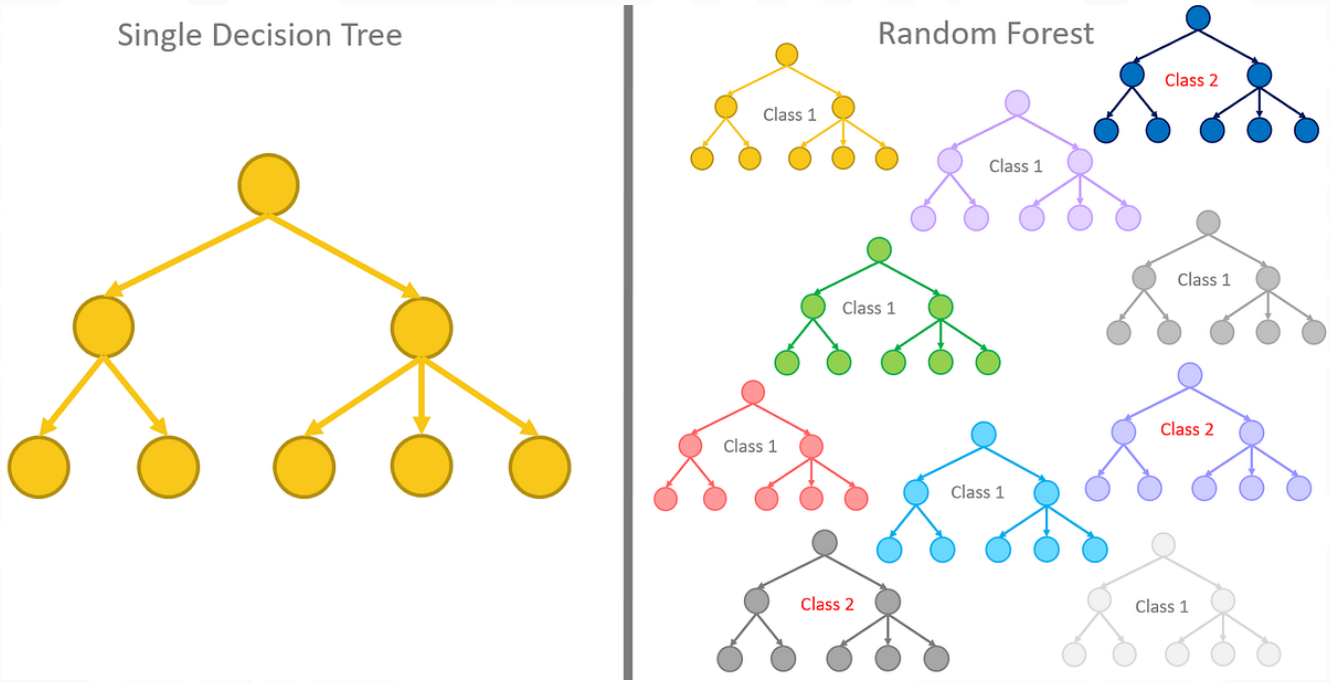
SVM and Tree-based Models

Dana Golden, Lilia Maliar



Data Science and Machine Learning - **November 30, 2024**

Presentation Outline

1. Introduction and Background
2. Support Vector Machines
3. Tree-based Models
4. Boosting

Review of Classification

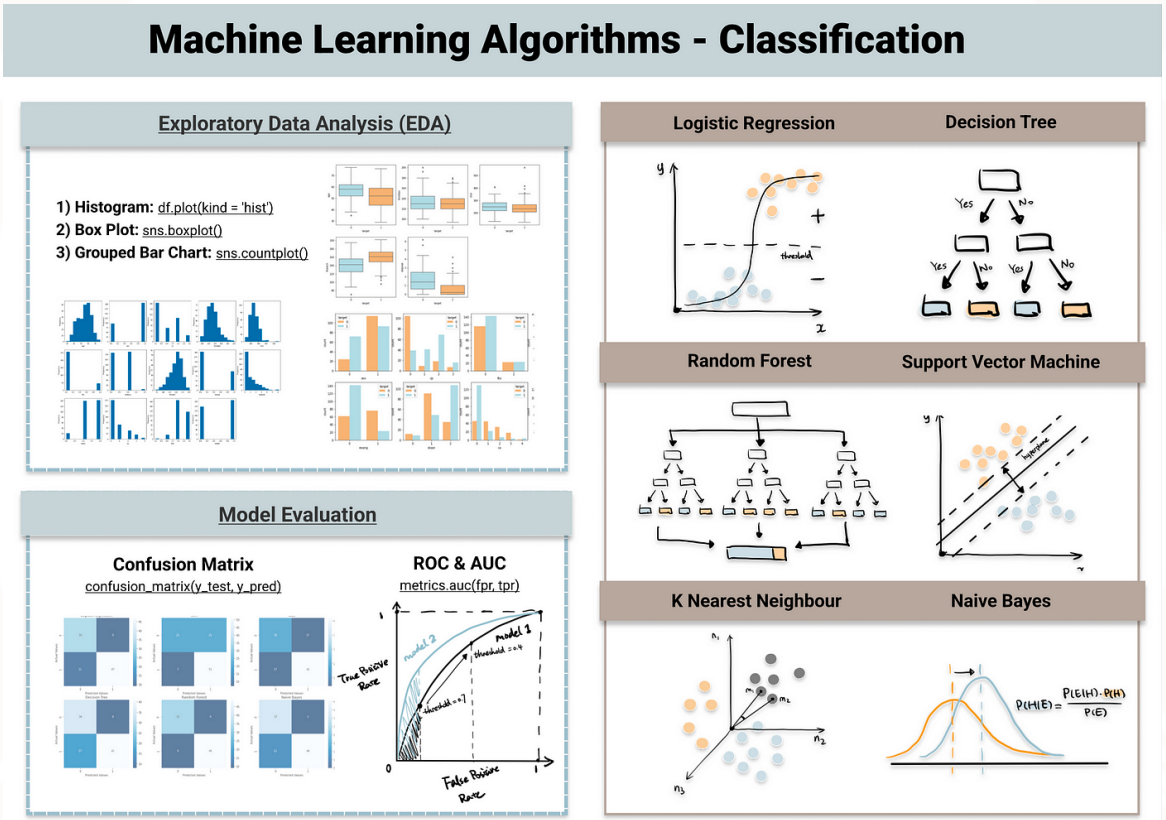


Figure 1: Classification Overview

SVM: Purpose and Overview

- SVM finds a linear separating hyperplane
- The goal is to find a hyperplane in n-d space that best separates two classes
- Valuable because it provides a more versatile way to classify in high-dimmensional spaces
- Computationally and memory intensive to run

SVM visualized

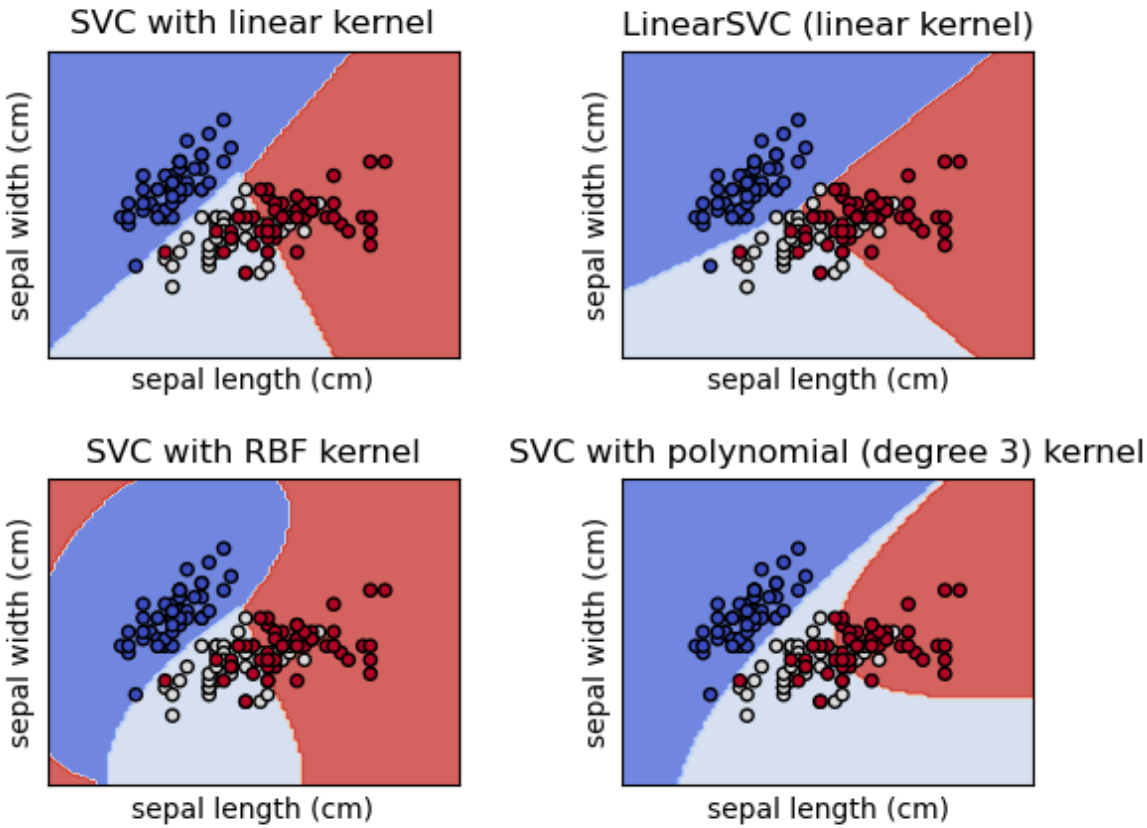


Figure 2: SVM Visualized

Determining Best

- SVM finds the margin that maximizes the distance between the closest datapoints from both classes
- Margin distance between the hyperplane and the observations closest to the hyperplane (support vectors)

Mathematical Formulation

- SVM finds the weight and bias that maximize margin
- Very similar to perceptron

$$\max_{w,b} \gamma(w, b) s.t. \forall i y_i(w^T) \geq 0 \tag{1}$$

$$\max_{w,b} \frac{1}{||w||_2} \min_{x_i \in D} |w^T x_i + b| s.t. \forall i y_i(w^T) \geq 0 \tag{2}$$

Kernel Trick

- What if there is a nonlinear boundary?

Kernel Trick

- What if there is a nonlinear boundary?
- Transform data into space of the type of boundary to make boundry linear in that space

$$k(x, z) =< \phi(x), \phi(z) > \tag{3}$$

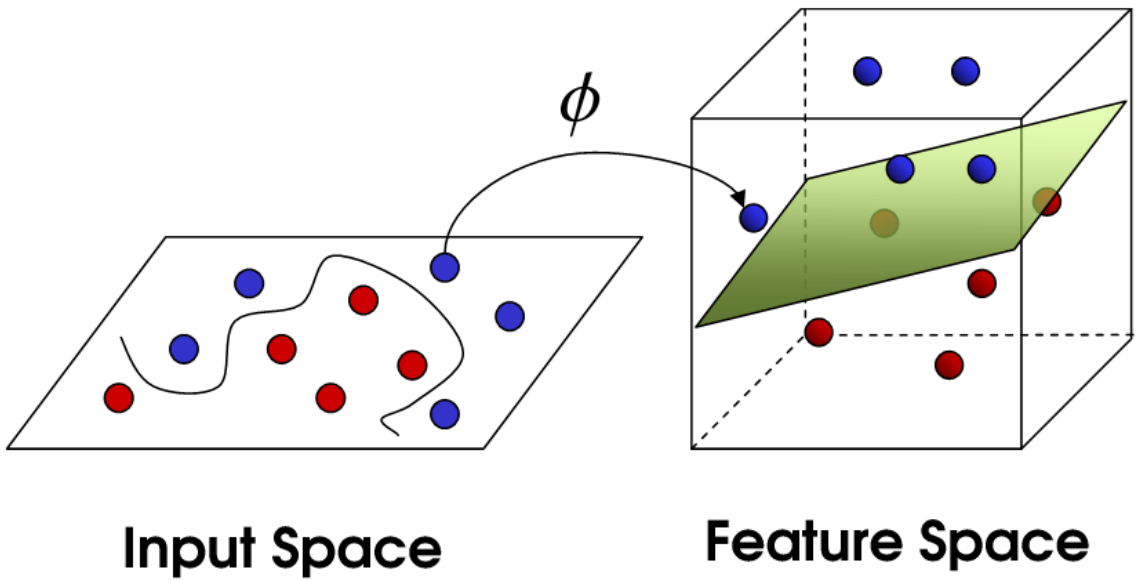


Figure 3: Kernel Trick

Multi-class SVM

- Two possibilities for dealing with multiple classes in SVM:

Multi-class SVM

- Two possibilities for dealing with multiple classes in SVM:
- 1 V rest: Perform SVM iteratively between one class and each other class, take the class with most confidence

Multi-class SVM

- Two possibilities for dealing with multiple classes in SVM:
- 1 V rest: Perform SVM iteratively between one class and each other class, take the class with most confidence
- One v. one: Compare each pair of classes, take class with most confidence

SVM Anomaly Detection

- One-class SVM finds the smallest-radius ball that surrounds all seen datapoints
- All datapoints outside of ball considered anomalies

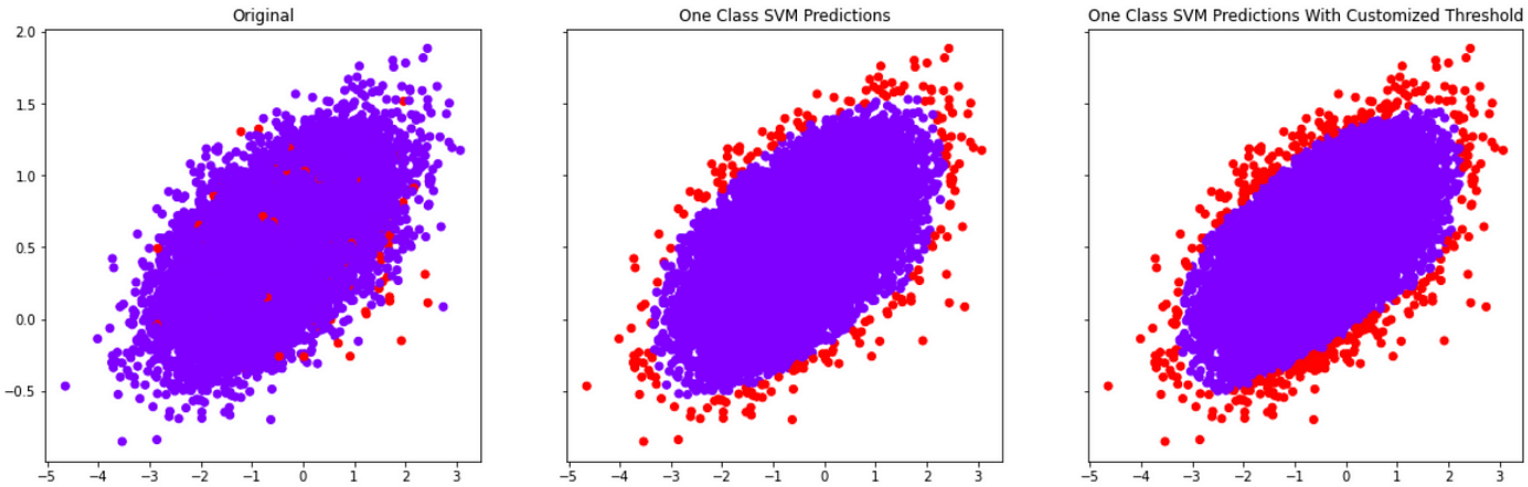


Figure 4: One-class SVM

Classification and Regression Trees

- Classification and regression trees separate classes or datapoints based on cutoffs in data
- Classification trees separate classes based on the values of features that provide most data, majority class taken in end
- Regression trees do the same but take the average in the end as prediction

When to use Decision Trees?

- Linear regression outperforms decision trees when
- Decision trees outperform linear regression when relationships in data are nonlinear
- Particularly, decision trees outperform when relationship involves jumps or thresholds

Impurity Measures

- Gini Index: Gini index measures the probability that a random observation would be misclassified $Gini = 1 - P(i)^2$
- Entropy: entropy focuses on the amount of randomness in the sample, it is minimized when all share one class and maximized when half observations are from each class, splits performed to maximize change in entropy or "information gain"

$$E = -\sum_{i=1}^{n} p_i log_2(p_i) \tag{4}$$

- Variance reduction: Most regression trees focus on variables that most reduce variance and either take median or average or experiment with range of values for split

Pruning

- Pruning reduces overfitting by deciding to restrict the level of complexity or granularity of the tree
- Pre-pruning (before tree created):
  - Choose maximum depth
  - Choose minimum samples per leaf
  - Choose minimum samples per split
  - Choose maximum features
- Post-pruning (after tree created):
  - Cost-complexity pruning: Price of accuracy and complexity, only keeps subtree that minimizes cost
  - Reduced Error Pruning: Remove nodes that do not increase accuracy
  - Minimum leaf size: Removes leaf nodes with samples below threshold

Classification Tree Visualization

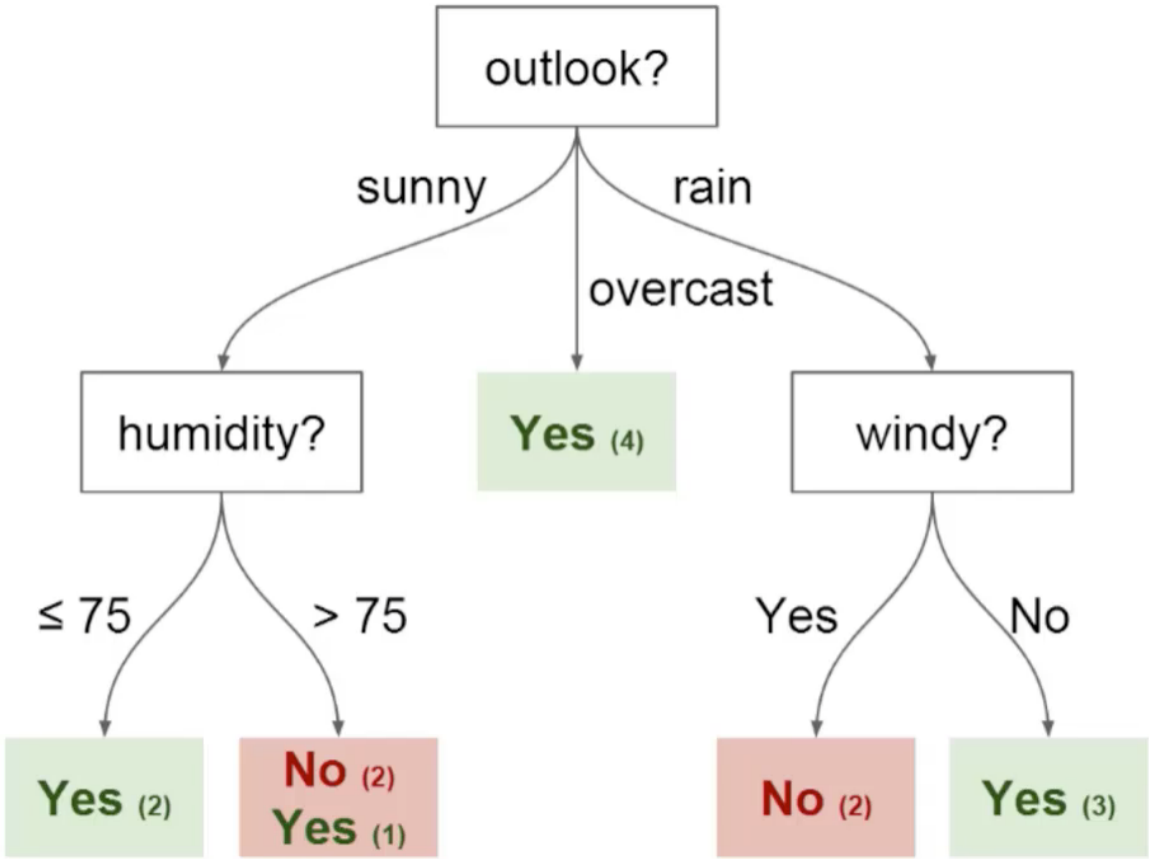


Figure 5: Decision Tree visual

Bootstrapping

- When bootstrapping, sample with replacement from original dataset
- Bootstrapping increases sample size and reduces bias by increasing variance
- Common in statistics and in econometrics, particularly for finding standard errors

Random Forest

- Random forest generates decision trees using bootstrapping and randomly selects a subset of features
- Random forest then builds a decision tree on the new dataset and repeats for a set number of iterations
- A majority vote decides what the class of any datapoint will be
- In case of regression, an average is taken
- The idea of random forest is to sample the range of possible decision trees rather than using the boundaries that naturally come from data

Random Forest Visualization

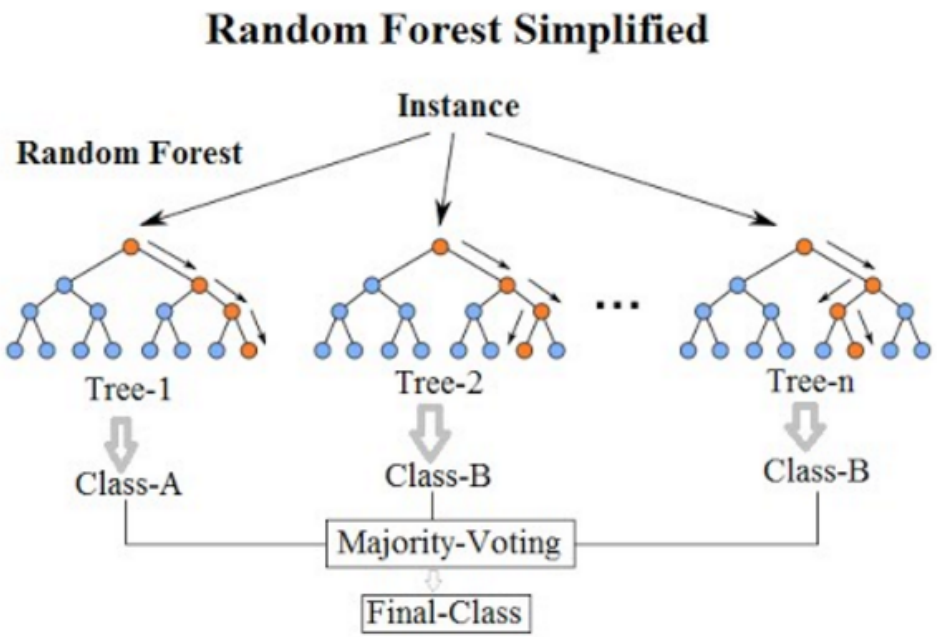


Figure 6: Random Forest

When to use Random Forest

- Random forest have higher variance but lower bias than decision trees
- Random forest is less likely to overfit with few datapoints
- Random forests are less interpretable than a single decision tree
- Random forests can be more difficult to create and sensitive to poor draws

The Idea of Boosting

- An "ensemble" model composed of different simple models may outperform a more complex model
- A set of simple models and rules are created to create a stronger model
- Works because individual models learn different rules

The Idea of Boosting

- An "ensemble" model composed of different simple models may outperform a more complex model
- A set of simple models and rules are created to create a stronger model
- Works because individual models learn different rules
- In classification, weighted majority rule taken. In regression, weighted average

Boosting Visualized

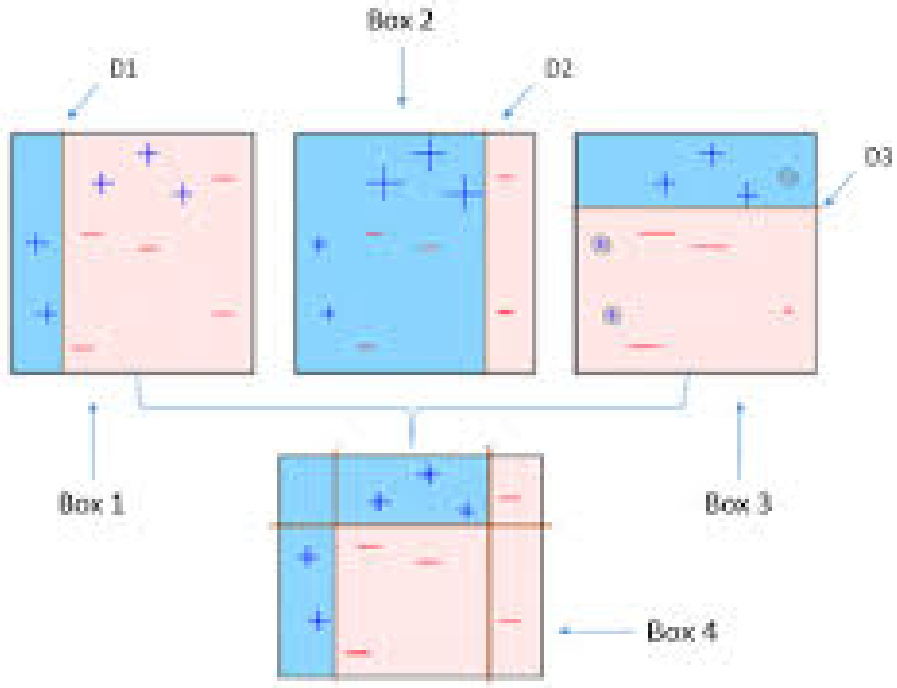


Figure 7: Boosting Visual

Boosting v. Bagging

- In boosting, multiple models are created focusing on different features of the dataset
- In bagging, a single model is trained using data sampled without replacement

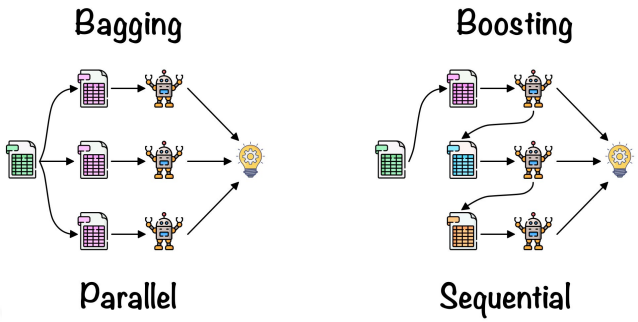


Figure 8: Boosting Vs. Bagging

Adaboost Algorithm

1. For data set of n samples, weight $w_i = \frac{1}{n}$
2. For M iterations:
3. Fit classifier $K(x)$ on all samples
4. $\epsilon = \frac{\sum_{y_i \neq K_m(x_i)} w_i^{(m)}}{\sum_{y_i} w_i^{(m)}}$
5. Compute $\alpha_m = \frac{1}{2} ln(\frac{1-\epsilon}{\epsilon})$
6. Update all weights: $w_i^{(m+1)} = w_i^{(m)} e^{-\alpha y K_m(x)}$
7. New predictions: $K(x) = sign[\sum_{m=1}^{M} \alpha_m K_m(x)]$

Adaboost Example

- Consider the following dataset: The first two coordinates represent the value of two features, and the last coordinate is the binary label of the data.
- $X1 = (1, 0, +1), X_2 = (0.5, 0.5, +1), X_3 = (0, 1, 1), X_4 = (0.5, 1, 1), X_5 = (1, 0, +1), X_6 = (1, 1, +1), X_7 = (0, 1, 1), X_8 = (0, 0, 1)$

Adaboost Example Data

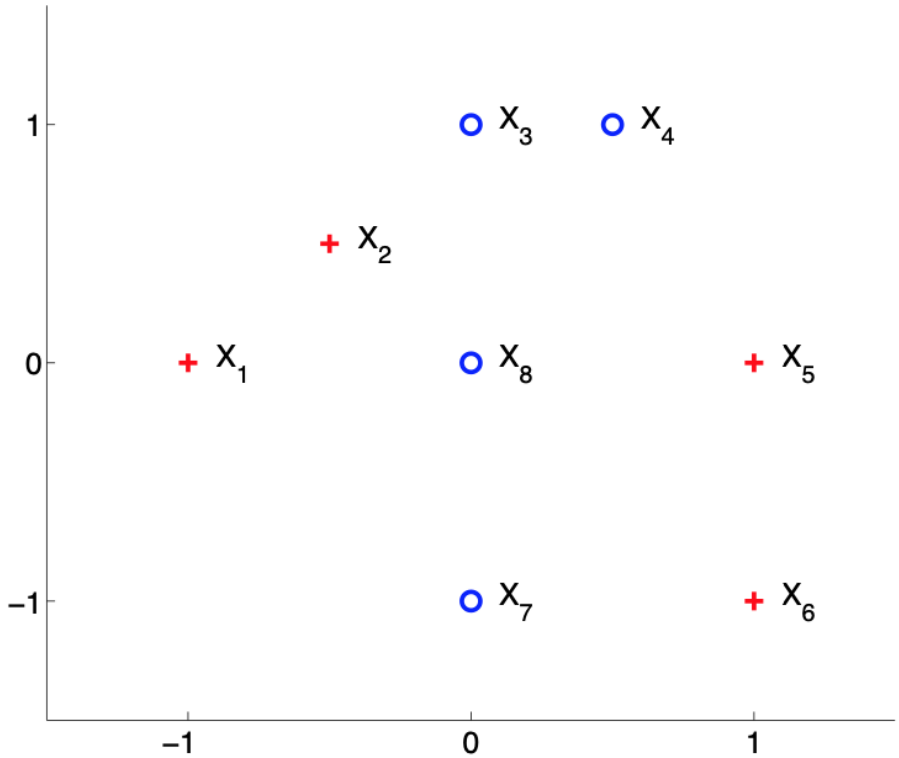


Figure 9: Adaboost Data

Adaboost Solved

ϵ_t	α_t	Z_t	$D_T(1)$	$D_T(2)$	$D_T(3)$	$D_T(4)$	$D_T(5)$	$D_T(6)$	$D_T(7)$	$D_T(8)$
$\frac{1}{4}$	.5493	.866	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$
.166	0.8071	0.745	0.0833	0.083	0.083	0.083	0.2499	0.2499	0.083	0.083
.35	.309	.807	0.25	.25	.0499	.0499	0.1496	0.1496	0.0499	0.0499

Decision Stump 1

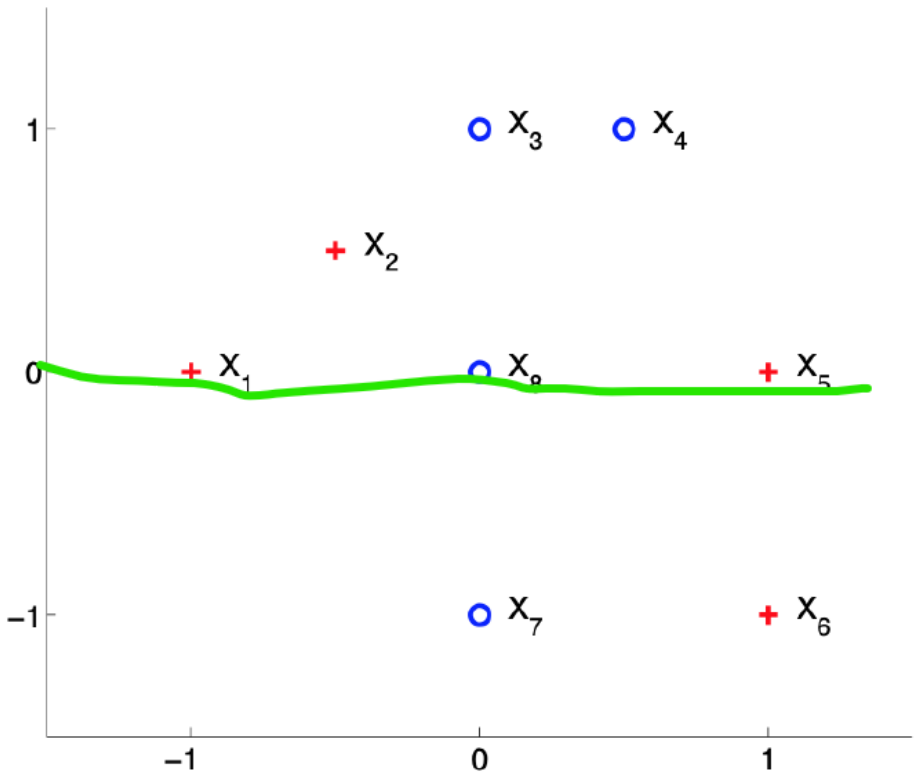


Figure 10: Decision Stump 1

Decision Stump 2

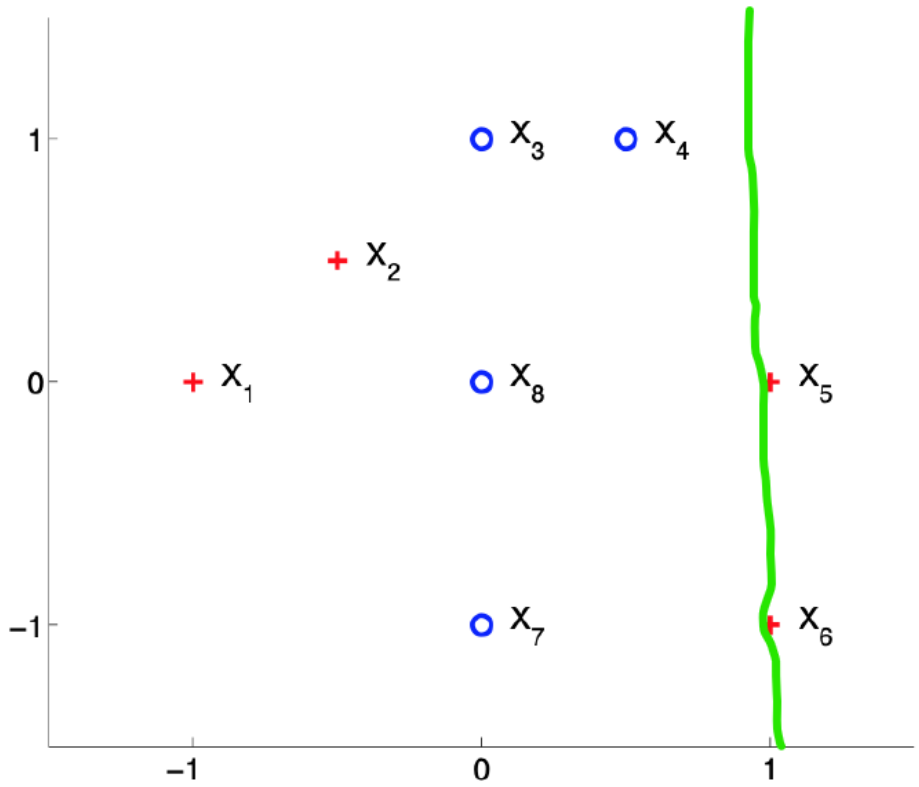


Figure 11: Decision Stump 2

Decision Stump 3

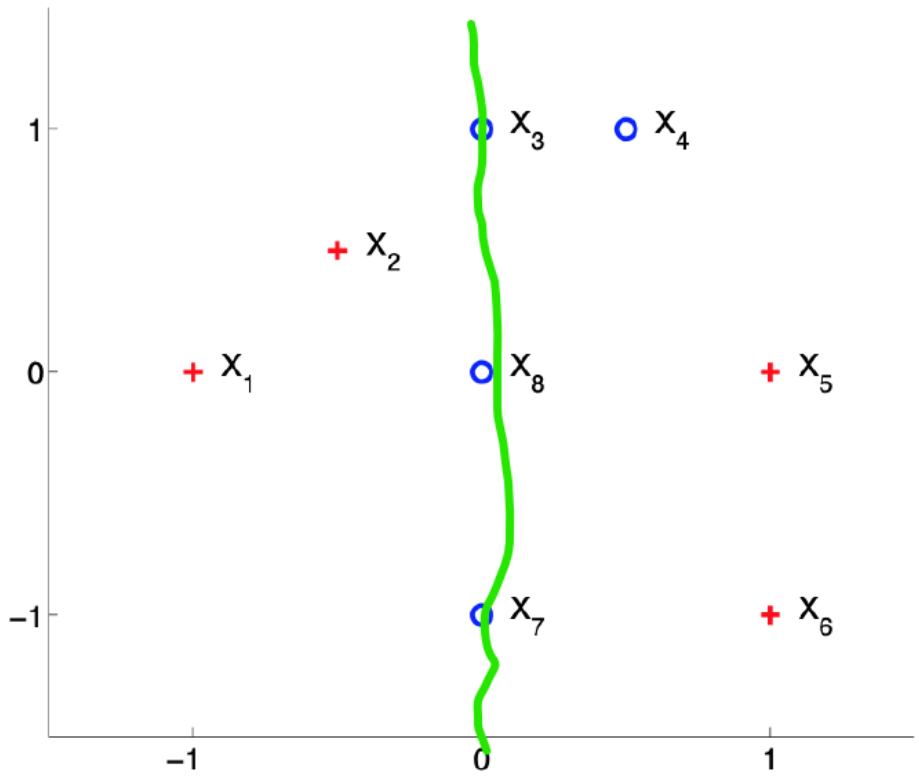


Figure 12: Decision Stump 3

Final Classification

- Point 1: .5493*1+.8071*(-1)+.309*1=.051
- Point 2: .5493*1+.8071*(-1)+.309*-1=-0.567
- Point 3: .5493*(-1)+.8071*(-1)+.309*-1=-1.665
- Point 4: .5493*(-1)+.8071*(-1)+.309*-1=-1.665
- Point 5: .5493*(-1)+.8071*(1)+.309*1=.5668
- Point 6: .5493*(-1)+.8071*(1)+.309*1=.5668
- Point 7: .5493*(-1)+.8071*(-1)+.309*1=-1.047
- Point 8: .5493*(-1)+.8071*(-1)+.309*1=-1.047
- Only one point is misclassified, better than any individual classifiers