Introduction to Machine Learning with Python

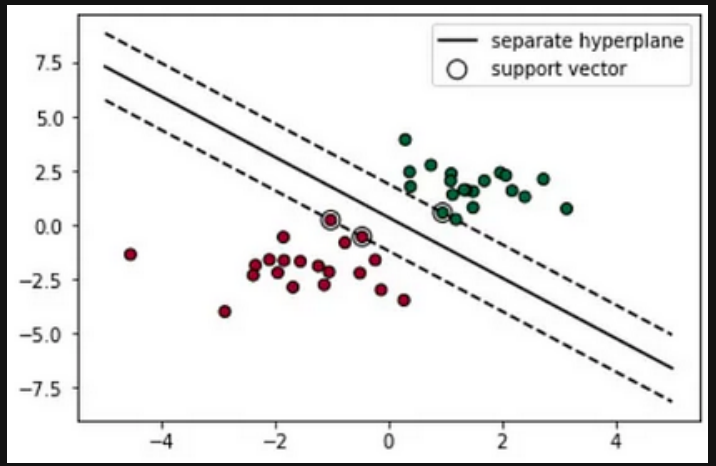
David Schaupp | WS2025

Content

```
Introduction to Machine Learning:
  - What is Machine Learning?
  - Machine Learning Project Checklist
Intro´s:
  - Colab
  - Python
  - Numpy
  - Matplotlib
Supervised Learning:
  - Classification (Binary|Multiclass)
  - Regression
  - Support Vector Machines
  - Decision Trees (Random Forest)
Unsupervised Learning:
  - Dimensionality Reduction
  - Clustering (k-means, DBSCAN)
Performances Measures:
  - Accuracy
  - Precision
  - Recall
  - F1-Score
  - ROC Curve
  - Confusion Matrix
```

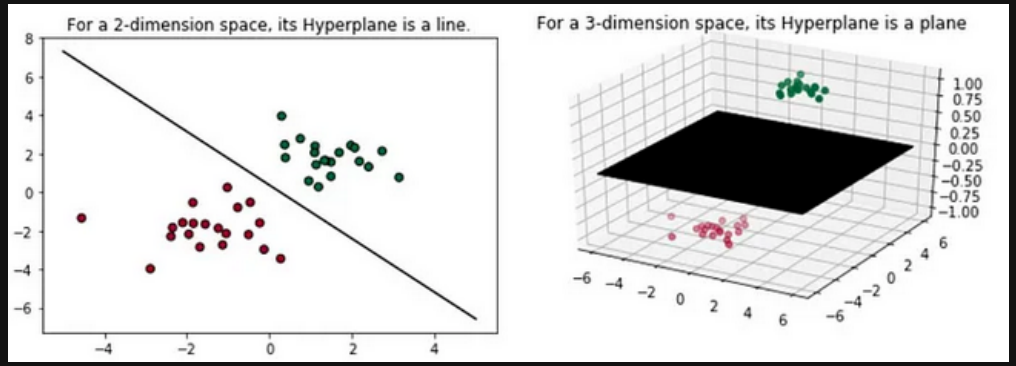
Support Vector Machine

- Supervised Learning Algorithm
- Used for Classification and Regression (mostly classification)
- Works well on small and medium sized datasets
  - training time increases rapidly with the size of the training set
- SVM are based on the idea of finding a hyperplane that best divides a dataset into two classes and maximizes the margin
  - Margin: distance between the hyperplane and the nearest data point from either class
  - Support Vectors: data points that are closest to the hyperplane



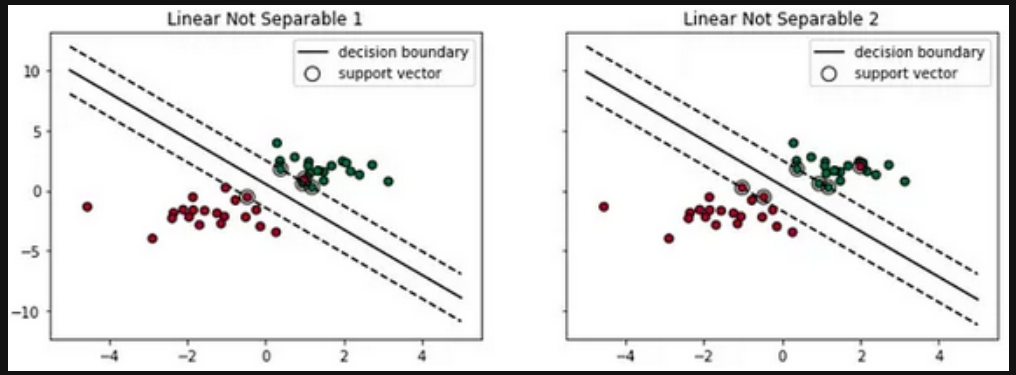
SVM linear separable data - Hyperplane & Margin

- Linearly separable data:
  - Data can be separated by a single straight line
- Hyperplane:
  - In a 2D space: a line
  - In a 3D space: a plane
  - In a n-dimensional space: a n-1 dimensional plane



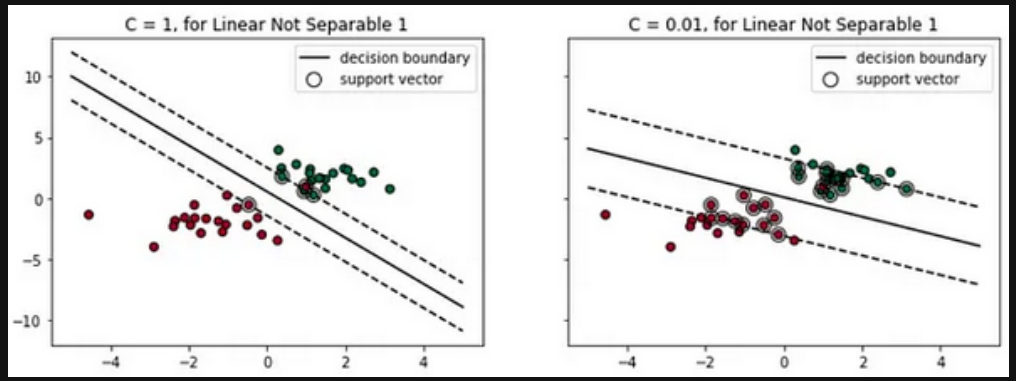
SVM linear non-separable data & Soft Margin

- Linearly non-separable data:
  - Data cannot be separated by a single straight line
  - In reality most of the data is not linearly separable
- Solutions:
  - Soft Margin: Try to find a line to separate, but allow some points to be on the wrong side
  - Kernel Trick: Try to find a non-linear decision boundary
  - (We'll focus on Soft Margin first)
- Two types of misclassification tolerated:
  - Points on the wrong side of the hyperplane but on the correct side of the margin
  - Points on the wrong side of the hyperplane and on the wrong side of the margin



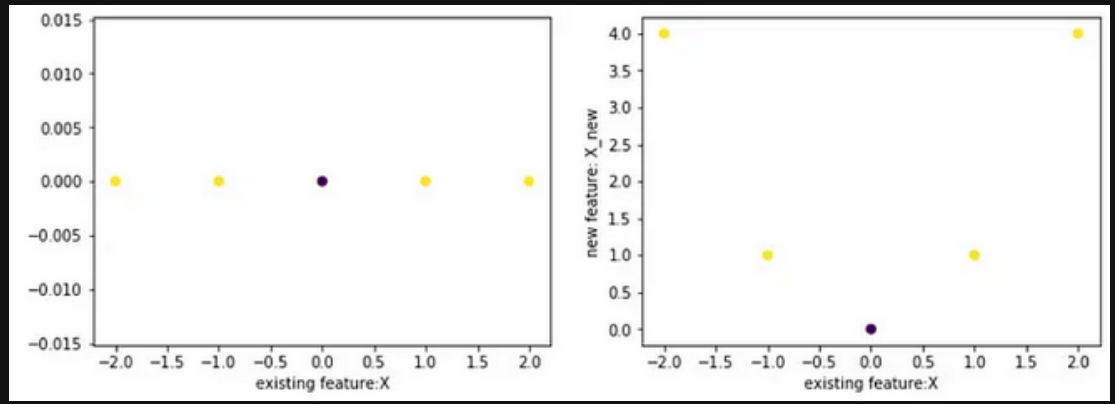
SVM linear non-separable data: Soft Margin

- Degree of tolerance:
  - Important parameter of SVM
  - C parameter:
    - Controls the width of the margin
    - Large C: narrow margin, less tolerance for misclassification (shown in left)
      - pro: more accurate
      - con: overfitting
    - Small C: wide margin, more tolerance for misclassification (shown in right)
      - pro: less overfitting
      - con: less accurate



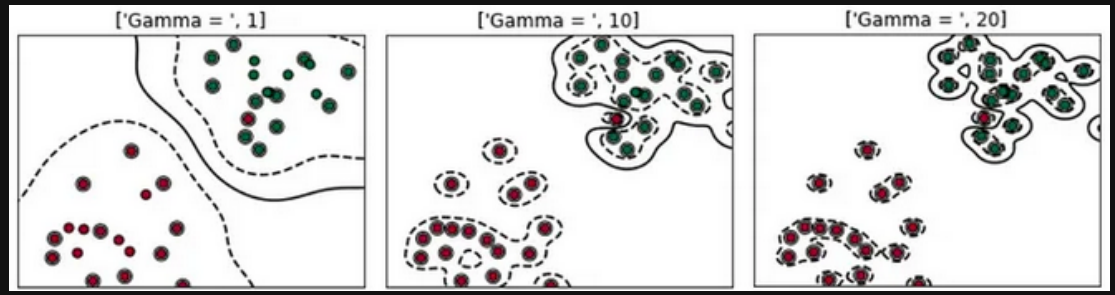
SVM linear non-separable data: Kernel Trick

- Utilizes existing features, applies transformations and creates new features
- New features are called kernels
- Most popular kernels:
  - Polynomial Kernel
  - Radial Basis Function (RBF)
- Polynomial Kernel Example:
  - Impossible to find a linear hyperplane to separate the data (shown in left)
  - Solution:
    - Apply polynomial kernel to the data (shown in right)
    - Transform the data into a higher dimensional space



SVM linear non-separable data: RBF Kernel

- Radial Basis Function (RBF):
  - Impossible to find a linear hyperplane to separate the data
  - Solution:
    - Apply RBF kernel to the data
    - Transform the data into a higher dimensional space
- Hyperparameter gamma:
  - Controls the smoothness of the decision boundary
  - Large gamma: more complex decision boundary
    - pro: more accurate
    - con: overfitting
  - Small gamma: smoother decision boundary
    - pro: less overfitting
    - con: less accurate

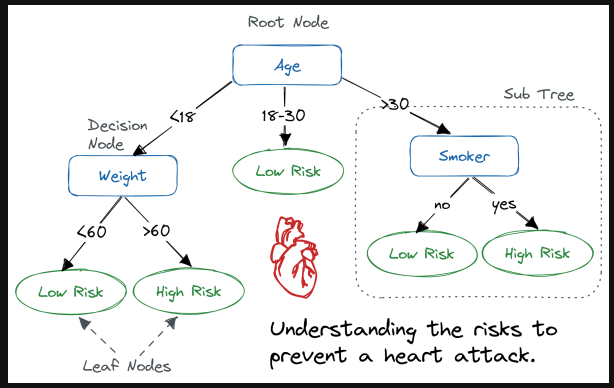


Content

```
Introduction to Machine Learning:
  - What is Machine Learning?
  - Machine Learning Project Checklist
Intro´s:
  - Colab
  - Python
  - Numpy
  - Matplotlib
Supervised Learning:
  - Classification (Binary|Multiclass)
  - Regression
  - Support Vector Machines
  - Decision Trees (Random Forest)
Unsupervised Learning:
  - Dimensionality Reduction
  - Clustering (k-means, DBSCAN)
Performances Measures:
  - Accuracy
  - Precision
  - Recall
  - F1-Score
  - ROC Curve
  - Confusion Matrix
```

Decision Trees

- Supervised Learning Algorithm
- Used for Classification and Regression (mostly classification)
- Pros:
  - Easy to understand and interpret
  - Feature selection not required
  - Can handle both numerical and categorical data
- Cons:
  - Prone to overfitting, unbalanced datasets create biased trees
- Terminology:
  - Root Node: topmost node in the tree
  - Decision Node: node that has child nodes
  - Leaf Node: node that does not have any child nodes



Decision Trees

- Example:
  - Play golf based on weather conditions
  - Binary Classification (True/False) based on 4 features
    - Outlook, Temperature, Humidity, Wind

Overlook	Temperature	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcoast	Hot	High	False	Yes

Decision Trees

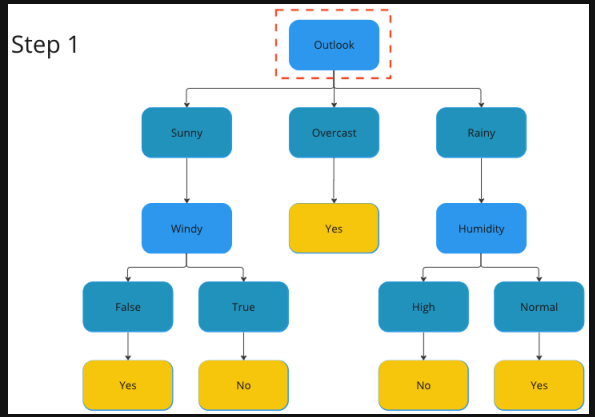
- Entropy:
  - Measures the impurity/randomness in a dataset
  - Low entropy (→0): Most samples belong to the same class
  - High entropy (→1): Samples are evenly split between classes

- Information Gain:
  - Measures how much a feature helps in classification
  - Higher gain = Better feature for splitting data

- Decision Making:
  - Calculate: Information Gain = Parent Entropy - Children Entropy
  - Choose feature with highest information gain

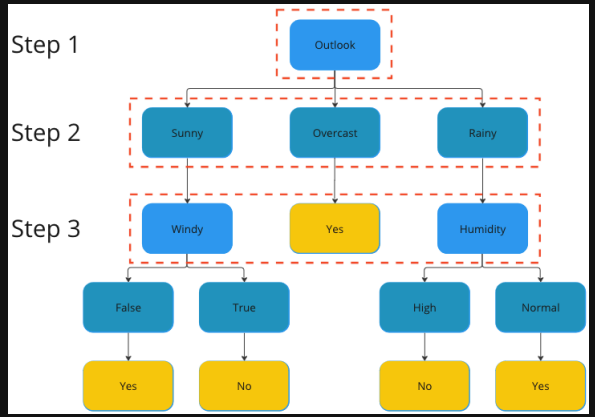
Decision Trees - C4.5

```
Steps:
1. Select Root Node
   - Calculate information gain for each feature
   - Choose feature that best splits the data
   - Example: "Outlook" splits data most effectively
2. Split Dataset
   - Divide data based on root node values
   - Example: Split on "Sunny", "Rainy", "Overcast"
3. Repeat for Each Branch
   - Calculate information gain for remaining features
   - Choose best feature for each subset
4. Continue until reaching leaf nodes (final decisions)
```



Decision Trees - C4.5

```
Steps:
1. Select Root Node
   - Calculate information gain for each feature
   - Choose feature that best splits the data
   - Example: "Outlook" splits data most effectively
2. Split Dataset
   - Divide data based on root node values
   - Example: Split on "Sunny", "Rainy", "Overcast"
3. Repeat for Each Branch
   - Calculate information gain for remaining features
   - Choose best feature for each subset
4. Continue until reaching leaf nodes (final decisions)
```



Random Forest

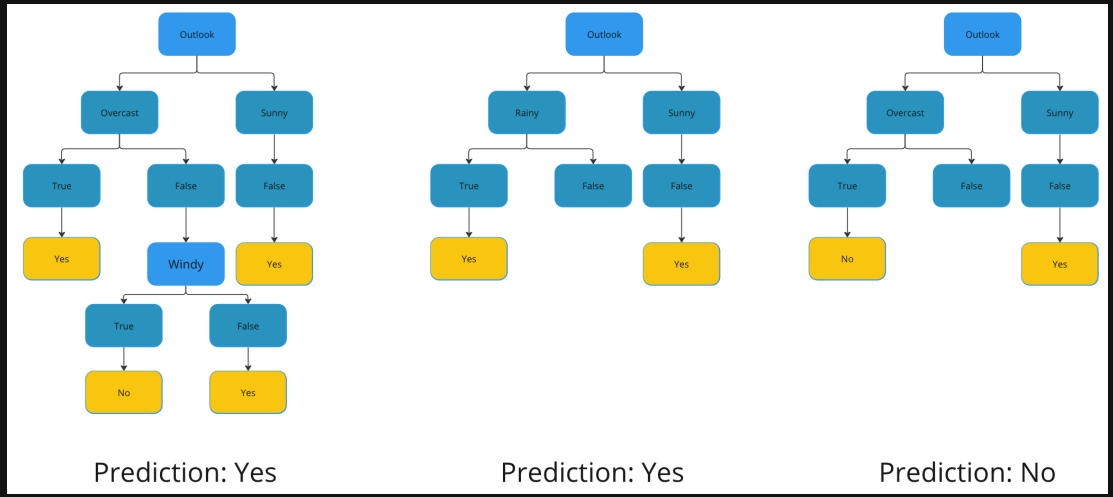
- Supervised Learning Algorithm
- Used for Classification and Regression (mostly classification)
- Ensemble Learning Method
  - Uses multiple Decision Trees
- Pros:
  - More accurate than a single Decision Tree
  - Robust to overfitting
  - Feature selection not required (feature selection is done automatically)
- Cons:
  - Less interpretable than a single Decision Tree
  - Slower than a single Decision Tree

Random Forest

- Steps in building a random forest:
  - 1. Pick a random subset of the training data
  - 2. Build a decision tree based on the subsets
  - 3. Make a prediction based on the majority vote
    - F.e. 60 trees predict class "Play Golf", 40 trees predict class "Don´t Play Golf"
    - Prediction: class "Play Golf"

Random Forest

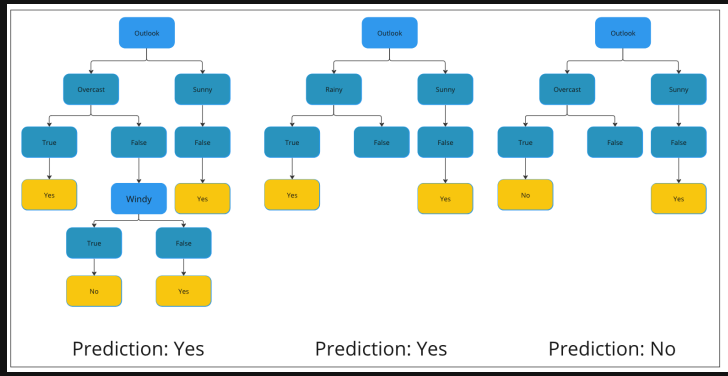
```
- Steps in building a random forest:
  - 3. Prediction based on majority vote
    - 2 trees predicted "Yes"
    - 1 tree predicted "No"
    - Prediction based on majority vote: "Yes"
```

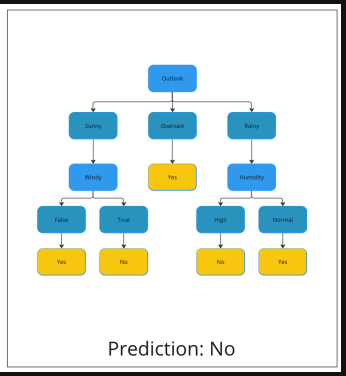


Summary

- Decision Trees:
  - A interpretable supervised learning method, effective in both classification and regression
  - Utilizes entropy and information gain to make decisions
  - Pros: Easy to understand and implement, automatic feature selection, and handles various data types
  - Cons: Prone to overfitting and can be biased in unbalanced datasets

- Random Forest:
  - An ensemble learning method that combines multiple Decision Trees to enhance prediction accuracy
  - Builds various trees on different data subsets and aggregates their predictions.
  - Pros: Higher accuracy, robust against overfitting, and automatic feature selection
  - Cons: Less interpretable than individual Decision Trees and slower in training and prediction





Let´s get our hands dirty