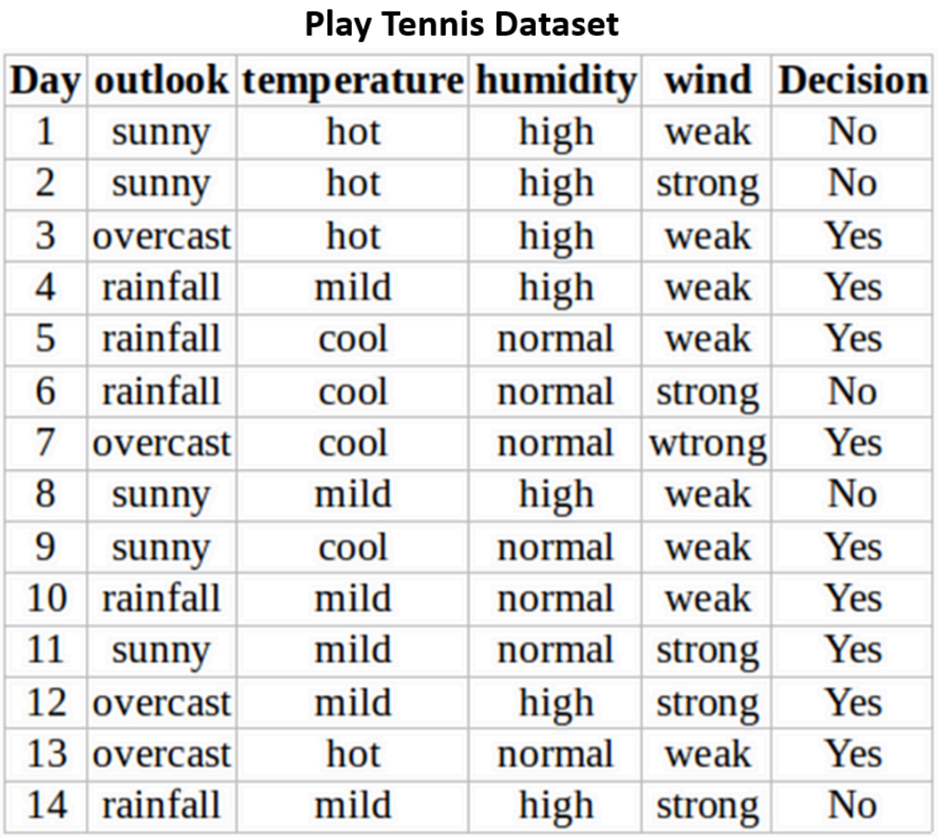
Open in Colab YouTube

Video Lecture
محاظرة مسجلة

Introduction to Classification using Decision Trees

A **decision tree** is a popular supervised learning algorithm used for both classification and regression tasks.
In classification, the goal is to predict the class label of an observation by splitting the data based on feature values in a tree-like structure.

For example, to decision if we can paly tennis or not on a give day, we can use weather information to build a classification model to assist us in making the decision.



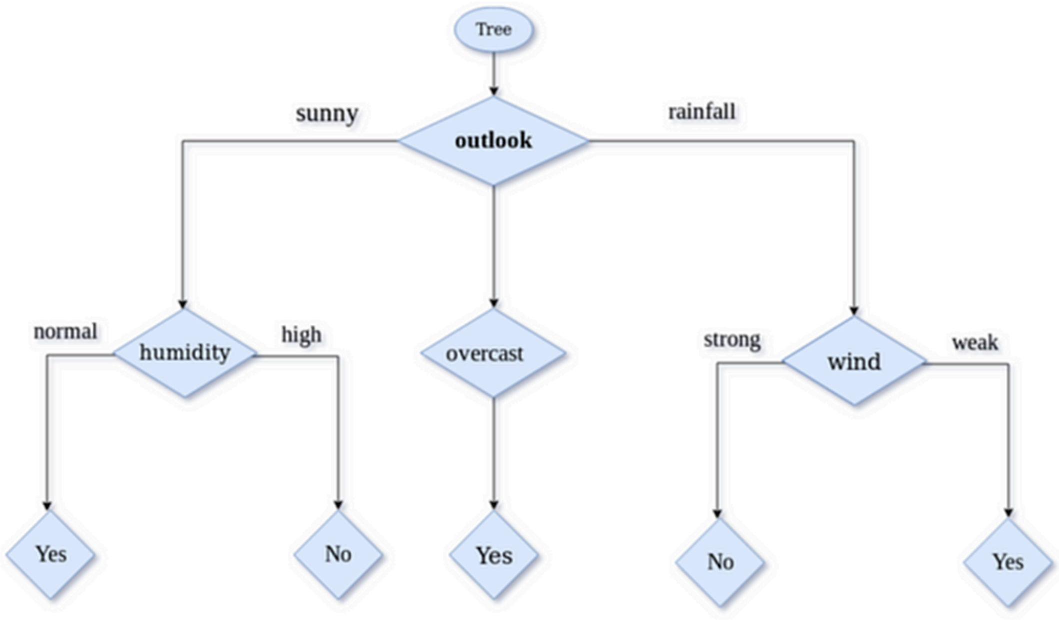
Play Tennis Dataset

Day	outlook	temperature	humidity	wind	Decision
1	sunny	hot	high	weak	No
2	sunny	hot	high	strong	No
3	overcast	hot	high	weak	Yes
4	rainfall	mild	high	weak	Yes
5	rainfall	cool	normal	weak	Yes
6	rainfall	cool	normal	strong	No
7	overcast	cool	normal	wtrong	Yes
8	sunny	mild	high	weak	No
9	sunny	cool	normal	weak	Yes
10	rainfall	mild	normal	weak	Yes
11	sunny	mild	normal	strong	Yes
12	overcast	mild	high	strong	Yes
13	overcast	hot	normal	weak	Yes
14	rainfall	mild	high	strong	No

Components of a Decision Tree

- **Root Node**: The first node in the decision tree.
- **Internal Nodes**: Nodes that represent decisions based on features.
- **Leaf Nodes**: Terminal nodes that contain the predicted class.

The decision tree makes predictions by following a path from the root to a leaf node.



Create a Decision Tree from using Python

Note: We will use the Gini Index in the **scikit-learn** library to establish the tree and determine howto split nodes.

Click Here for Gini Index Explaination

Click Here for Gini Index Intuition

Video Lecture
محاضرة مسجلة

Video Lecture
محاضرة مسجلة

Stepts for creating a decision tree in Python:

1. Import Libraries.

2. Get Data.
3. Split data into X and y, independent and dependent variables.
4. We split the dataset into training and testing sets.
5. Initialize a decision tree classifier with the Gini Index as the criterion.
6. Train the model using the training set.
7. Use the trained model to make predictions on the test set.
8. Evaluate the model’s accuracy.
9. We plot the decision tree for visualization.

Video Lecture
محاضرة مسجلة

In [2]:

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt

# Load the dataset
url = "https://raw.githubusercontent.com/msfasha/307304-Data-Mining/main/datasets/l
data = pd.read_csv(url)

# Separate features and target variable
X = data.drop('loan_default', axis=1)  # Features
y = data['loan_default']               # Target variable

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta

# Initialize the Decision Tree Classifier with Gini index
model = DecisionTreeClassifier(criterion='gini', random_state=42)

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

result = pd.DataFrame({"Original Values": y_test , "Predictions":y_pred})
result.head(10)
```

Out[2]:

	Original Values	Predictions
521	0	0
737	0	0
740	1	1
660	0	0
411	0	0
678	0	0
626	0	1
513	0	0
859	0	0
136	0	0

Evauation the Model

In [5]:

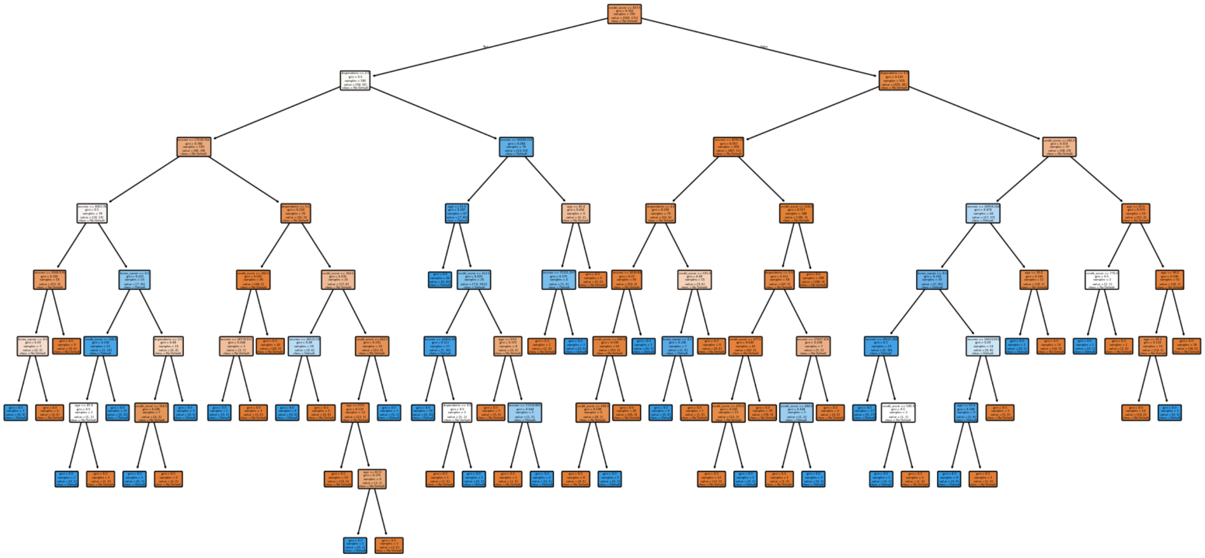
```
# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.86

Classification Report:
               precision    recall  f1-score   support

           0       0.91      0.92      0.91       236
           1       0.68      0.66      0.67        64

    accuracy                           0.86       300
   macro avg       0.79      0.79      0.79       300
weighted avg       0.86      0.86      0.86       300


Confusion Matrix:
 [[216  20]
 [ 22  42]]
```

Plot the Generated Decision Tree

In [6]:

```
# Plot the Decision Tree
plt.figure(figsize=(20, 10))
plot_tree(model, feature_names=X.columns, class_names=['No Default', 'Default'], fi
plt.title("Decision Tree for Loan Default Prediction")
plt.savefig("decision_tree_plot.png", format="png", dpi=300, bbox_inches='tight')
plt.show()
```



Decision Tree for Loan Default Prediction