Decision Tree in Machine Learning

Decision trees are a powerful tool in machine learning for predicting outcomes based on input data. They break down complex decisions into a series of simpler choices.

Introduction to Decision Trees

Decision trees are used in many applications, including medical diagnosis, financial risk assessment, and customer segmentation.

1 **Tree Structure**

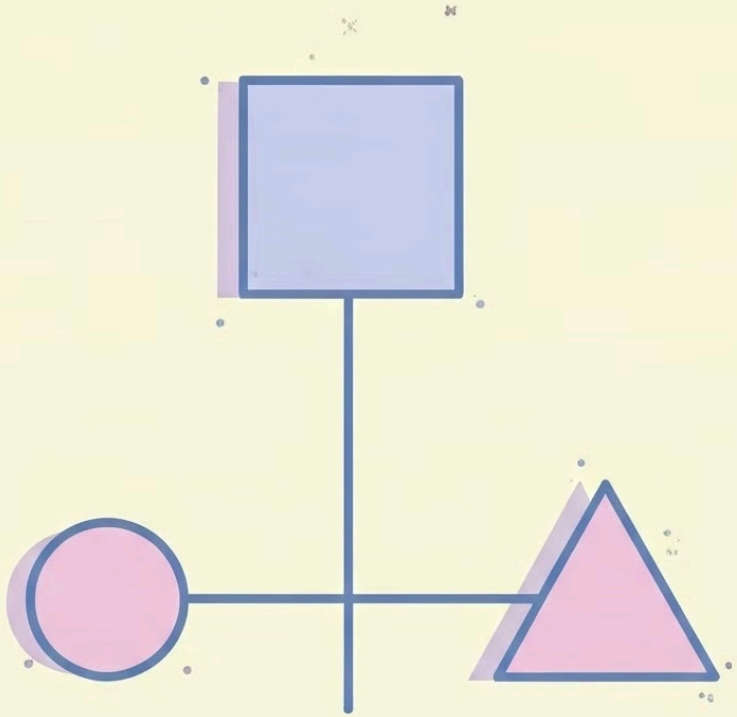
Decision trees resemble a branching tree structure.

2 **Decision Nodes**

Each node represents a decision or attribute, and the branches represent possible outcomes.

3 **Leaf Nodes**

Leaf nodes represent the final predictions or classifications.



Key Concepts and Terminology

Understanding key concepts is crucial for effective decision tree implementation.

Root Node	Represents the starting point of the tree.
Internal Node	Represents a decision based on an attribute.
Leaf Node	Represents the final prediction.
Branch	Represents a possible outcome of a decision.
Decision Boundary	Represents the line that separates different classifications.

Building a Decision Tree

Decision trees are built using algorithms like ID3, C4.5, and CART.

1

Data Preparation

Clean and pre-process the data to prepare it for tree construction.

2

Attribute Selection

Choose the best attribute to split the data at each node.

3

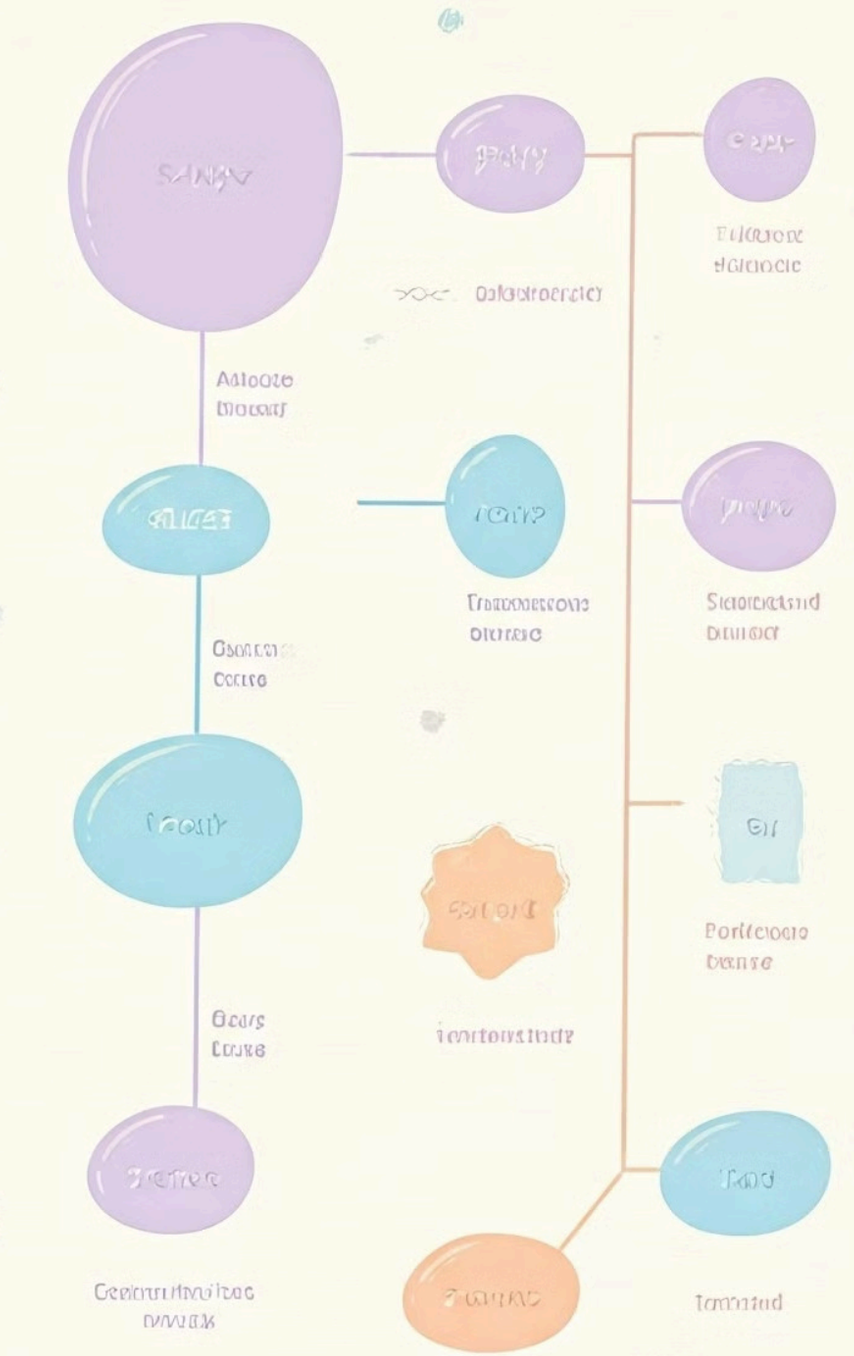
Tree Growth

Recursively split the data until a stopping criterion is met.

4

Pruning

Simplify the tree to avoid overfitting and improve generalization.



Advantages of Decision Trees

Decision trees offer a variety of advantages for classification and regression tasks.

Interpretability

They are easily understood and visualized, making the decision process transparent.

Non-parametric

They can handle both categorical and numerical data, making them flexible.

Robustness

They are relatively resistant to outliers and noisy data.

Limitations and Challenges

Despite their advantages, decision trees have certain limitations.

1

Overfitting

Trees can become too complex, leading to poor performance on unseen data.

2

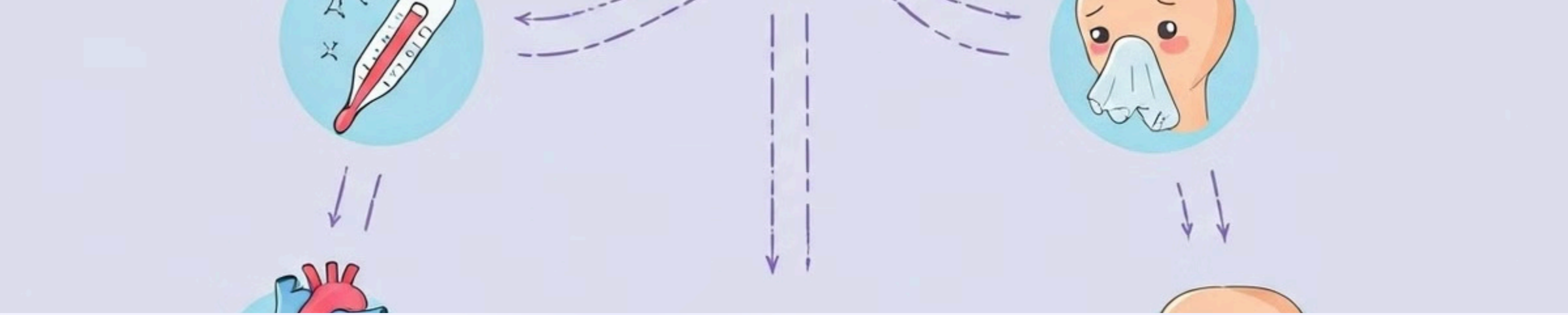
Instability

Small changes in the data can lead to significant changes in the tree structure.

3

Bias

Decision trees can be biased towards attributes with more categories.



Practical Applications of Decision Trees

Decision trees are widely used in various industries, including healthcare, finance, and marketing.

Medical Diagnosis

Predicting diseases and identifying treatments based on patient symptoms.

Financial Risk Assessment

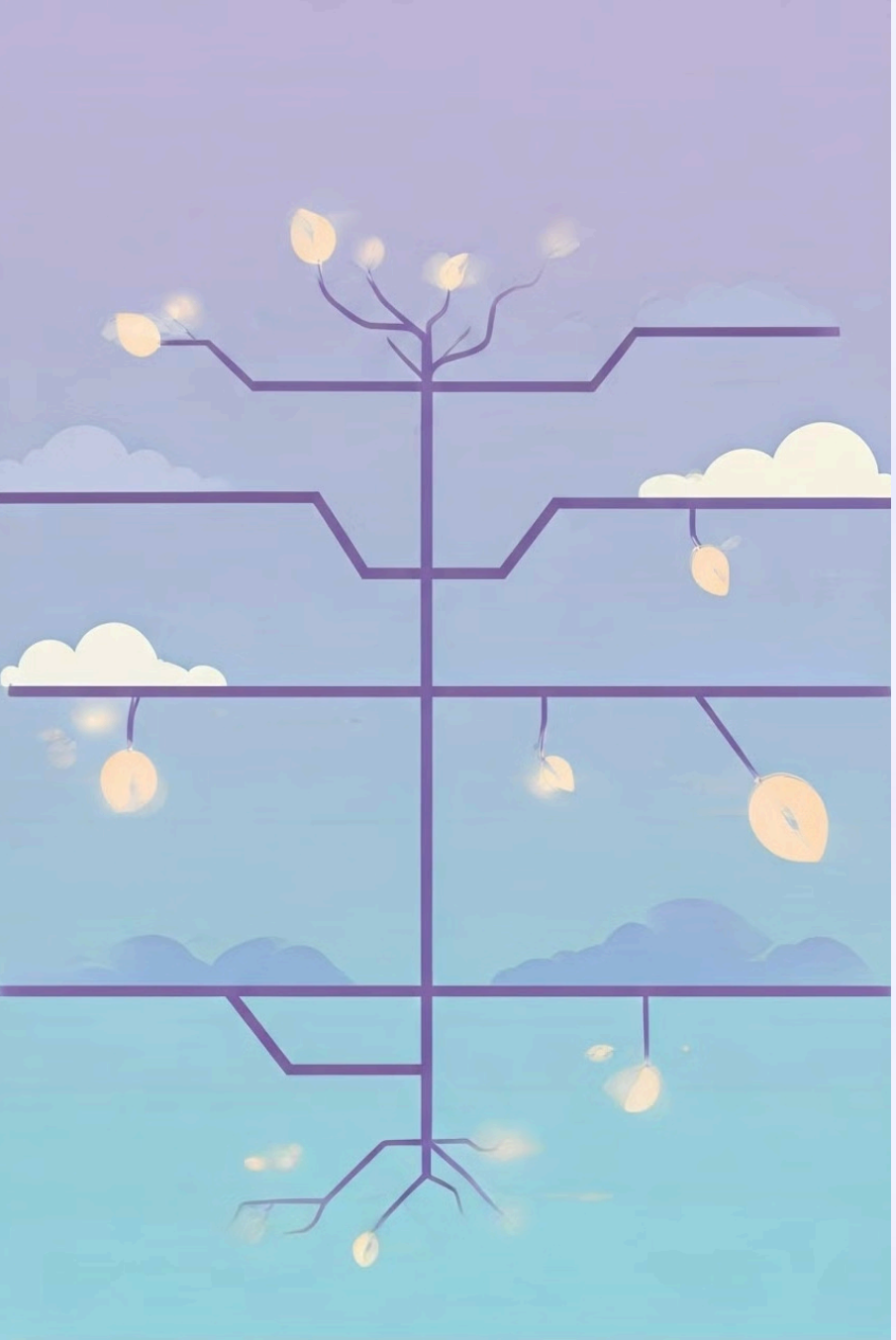
Evaluating creditworthiness and predicting loan defaults.

Customer Segmentation

Classifying customers into groups based on their preferences and behavior.

Machine Learning Automation

Automating tasks like fraud detection and image classification.



Conclusion and Future Trends

Decision trees are valuable tools with a bright future.

Ensemble Methods

Combining multiple decision trees to improve accuracy and reduce overfitting.

Deep Learning Integration

Using decision trees as part of deep learning architectures for complex tasks.

Explainable AI

Making decision trees more transparent and interpretable for ethical considerations.

Decision tree model for the given problem

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
import matplotlib.pyplot as plt
```

```
# Create the dataset
data = {
    'a1': [True, True, False, False, False, True, True, True, False, False],
    'a2': ['Hot', 'Hot', 'Hot', 'Cool', 'Cool', 'Cool', 'Hot', 'Hot', 'Cool', 'Cool'],
    'a3': ['High', 'High', 'High', 'Normal', 'Normal', 'High', 'High', 'Normal', 'Normal', 'High'],
    'Classification': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'No', 'Yes', 'Yes', 'Yes']
}
```

```
# Convert to DataFrame
df = pd.DataFrame(data)
```

```
# Convert categorical features into numeric using one-hot encoding
df_encoded = pd.get_dummies(df[['a1', 'a2', 'a3']])
```

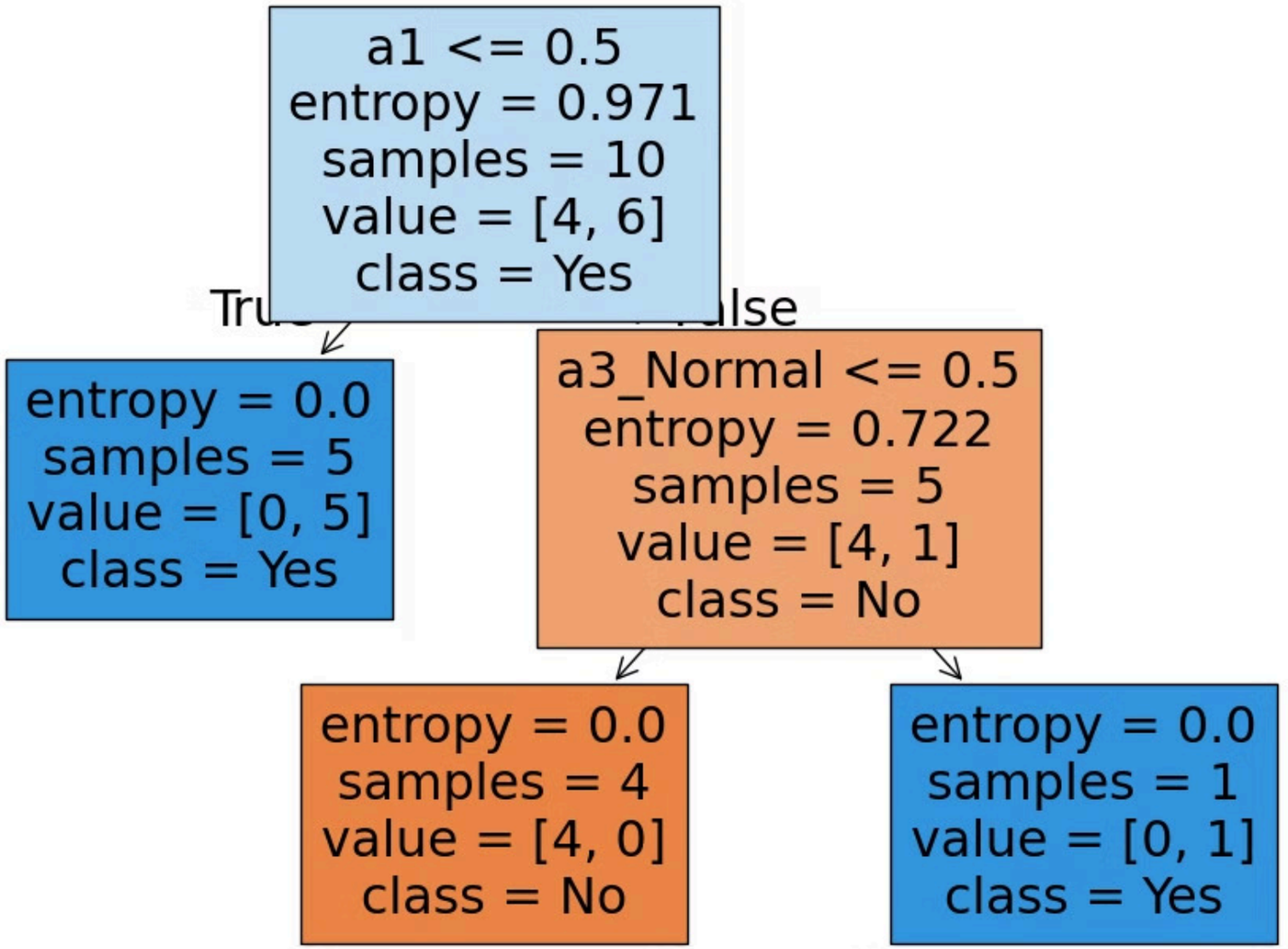
```
# Map classification to numeric
df['Classification'] = df['Classification'].map({'Yes': 1, 'No': 0})
```

```
# Split data into features and target
X = df_encoded
y = df['Classification']
```

```
# Create and train the decision tree model
clf = DecisionTreeClassifier(criterion='entropy')
clf.fit(X, y)
```

```
DecisionTreeClassifier(criterion='entropy')
```

```python
# Visualize the decision tree
plt.figure(figsize=(12,8))
tree.plot_tree(clf, feature_names=X.columns, class_names=['No', 'Yes'], filled=True)
plt.show()
```



```
# Predict on the training data (for simplicity)
y_pred = clf.predict(X)
```

ell (Ctrl+Enter)
as not been executed in this session

```
# Calculate accuracy
accuracy = (y_pred == y).mean()
print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
Accuracy: 100.00%
```

Done by

SARANKUMAR.S - 22AM055

AJINESH - 22AM007

GOBIKA - 22AM069