

# **Titanic Survival Prediction**

A Machine Learning Approach using Random Forest





#### **Team Members**

Ayushman Bhowmik Harshita Kumari

Roll No: 202401100300089 Roll No: 202401100300119

Harshit Singh Patel

Harshita Sharma

Roll No: 202401100300118 Roll No: 202401100300120

Date: 5/27/2025



# Problem Statement



Develop a classification model to predict survival of passengers on the Titanic using machine learning techniques.

## Why it matters

Real-world Application Practical use of classification models in historical data analysis

Peature Influence

Understanding how different factors affected survival outcomes

Learning Benchmark Industry-standard dataset for machine learning education











## **What is Random Forest?**

### **Core Concept**

An **ensemble learning method** that combines multiple decision trees to make predictions

Prediction = Majority Vote of Multiple Trees

## Key Features

Bootstrap Aggregating: Uses random samples with replacement

Feature Randomness: Considers random subset of features at each split

Voting: Final prediction based on majority vote

## Algorithm Process

#### 1. Bootstrap Sampling

Create multiple random samples from training data with replacement

#### 2. Tree Building

Train individual decision trees on each bootstrap sample

#### 3. Feature Selection

At each split, randomly select subset of features to consider

#### 4. Voting

Combine predictions from all trees using majority voting

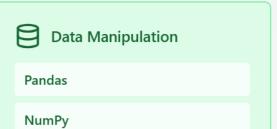
#### **®** Why Choose Random Forest?

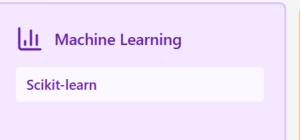
- · Reduces overfitting compared to single decision trees
- Handles both numerical and categorical features
- · Provides feature importance rankings
- Robust to outliers and missing values

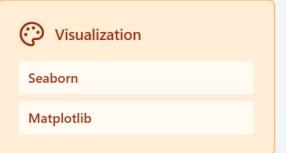


# 

**>** Programming Language Python







Nevelopment Stack

**Data Science** 

Pandas + NumPy for data manipulation and numerical computing

**Machine Learning** 

Scikit-learn for model training and evaluation

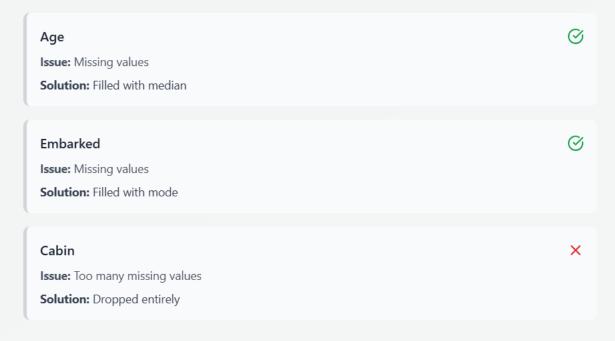
Visualization

Seaborn + Matplotlib for creating insightful charts



# **Data Cleaning**

## ⚠ Handled Missing Values



## × Dropped Irrelevant Features



# **Proposition** Feature Engineering

## **Data Transformations**

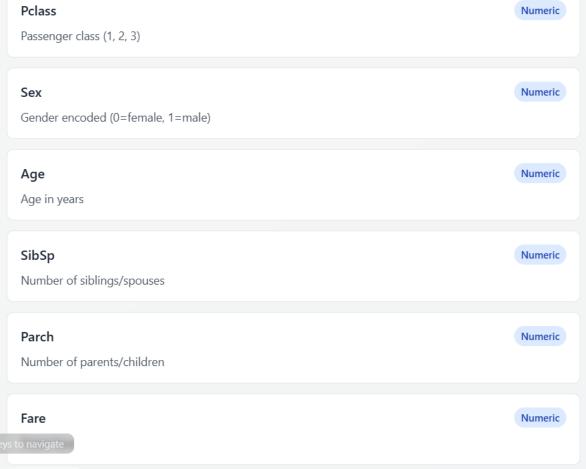






Machine learning algorithms require numeric input. Converting categorical variables to numbers enables the model to process them effectively.

### **≡** Final Feature Set



# **X** Data Splitting

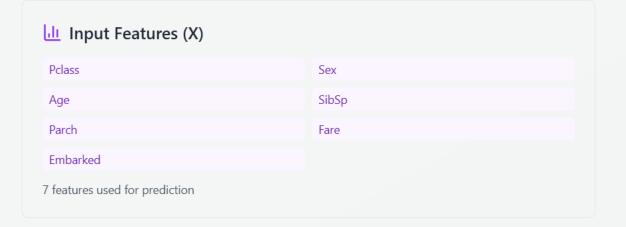
## **Split Strategy**

80% 20% **Training Set Testing Set** ~713 records ~178 records train\_test\_split(test\_size=0.2, random\_state=42)

#### **6** Why 80/20 Split?

- Industry standard for medium-sized datasets
- · Sufficient training data for model learning
- Adequate test data for reliable evaluation
- Maintains class distribution in both sets

## Q Data Structure











## A Random Forest Model



Import Model

from sklearn.ensemble import RandomForestClassifier

Import Random Forest classifier from sklearn

Initialize

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

Create model with 100 trees and fixed random state

Train

model.fit(X\_train, y\_train)

Train ensemble of decision trees on training data

**Predict** 



Algorithm Type

Ensemble method combining multiple decision trees with voting

> Training Process

Builds multiple trees using bootstrap sampling and random feature selection

Prediction Logic

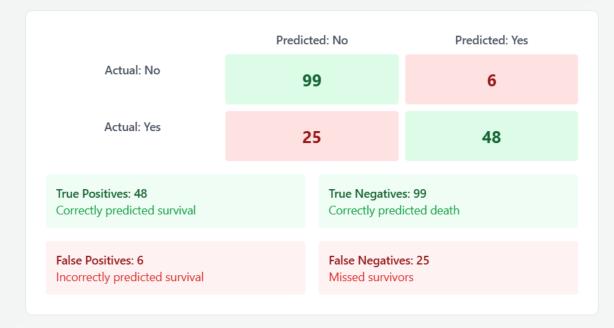
Uses majority voting from all trees to determine final prediction

- **@** Model Advantages
- **High Accuracy:** Often outperforms single models
- Overfitting Resistant: Ensemble reduces variance
- Feature Importance: Provides insights into data
- Robust: Handles missing values and outliers well

Use ← → arrow keys to navigate predictions = model.predict(X\_test)

## Results & Performance

## © Confusion Matrix



## **□** Performance Metrics

Accuracy Overall correct predictions	82.6%
Precision True positive rate	88.9%
Recall Sensitivity	65.8%
F1-Score Harmonic mean	75.4%



# **!** Insights

**Most Influential Features** 

Gender Impact

Females more likely to survive

Women and children first policy was evident in survival rates

Class Privilege

1st class had higher survival

Upper class passengers had better access to lifeboats

Age Factor

**Children had higher chances** 

Younger passengers were prioritized during evacuation

✓ e Model Insights

• Naive Bayes worked well despite its simplicity

• Feature independence assumption was reasonable

Gender was the strongest predictor

• Socioeconomic factors significantly influenced survival

Key Takeaways

**Model Performance** 

77.1% accuracy demonstrates Naive Bayes effectiveness for binary classification

Feature Importance

Demographic features (sex, class, age) were more predictive than family relationships

Historical Context

Use ← → arrow keys to navigate hm Choice

Naive Bayes proved suitable for this dataset size and feature types

These insights reflect the social hierarchies and maritime protocols of 1912, where class and ge

1.5+









**Kaggle Competition** 

Original dataset and competition details

https://www.kaggle.com/c/titanic

### Scikit-learn Documentation

Official Documentation

Machine learning library documentation and examples

https://scikit-learn.org/

### Python Data Science Libraries

**Multiple Sources** 

Pandas, Matplotlib, Seaborn official documentation



**Educational Resources** 

Academic resources and online learning materials

## **\*\*** Technology Stack

**Pandas** Data manipulation and analysis

NumPy 1.24+

Numerical computing

Scikit-learn 1.2+

Machine learning algorithms

Matplotlib 3.6+

Data visualization

Seaborn 0.12 +

Statistical visualization

#### Acknowledgments

• Kaggle for providing the Titanic dataset

arn contributors for the ML framework