Titanic Survival Prediction

A Machine Learning Approach using Random Forest

Machine Learning Data Science

Team Members

Ayushman Bhowmik
Roll No: 202401100300089

Harshita Kumari
Roll No: 202401100300119

Harshit Singh Patel
Roll No: 202401100300118

Harshita Sharma
Roll No: 202401100300120

Date: 5/27/2025

Problem Statement

Objective

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

Feature Influence
Understanding how different factors affected survival outcomes

Learning Benchmark
Industry-standard dataset for machine learning education

Introduction to the Titanic Dataset

Historical Context

Historical dataset from the 1912 Titanic tragedy - one of the most famous maritime disasters in history.

891 training records

Target Variable

0 = No	**1 = Yes**
Did not survive	Survived

Key Features

- **Age** Passenger age in years
- **Sex** Gender of passenger
- **Pclass** Passenger class (1st, 2nd, 3rd)
- **Fare** Ticket fare paid
- **SibSp** Number of siblings/spouses
- **Parch** Number of parents/children
- **Embarked** Port of embarkation

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

Tools & Libraries

Programming Language

Python

Data Manipulation

Pandas

NumPy

Machine Learning

Scikit-learn

Visualization

Seaborn

Matplotlib

Development 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

Age

Issue: Missing values

Solution: Filled with median

Embarked

Issue: Missing values

Solution: Filled with mode

Cabin

Issue: Too many missing values

Solution: Dropped entirely

Dropped Irrelevant Features

Name

Not predictive of survival

Ticket

Unique identifiers, no pattern

PassengerId

Random ID, not relevant

🎯 Cleaning Strategy

- Preserve data integrity while maximizing usable records
- Use statistical methods (median, mode) for imputation
- Remove features that don't contribute to prediction

Feature Engineering

Data Transformations

Sex

Original		Encoded
`male, female`	→	`0, 1`

Method: Label Encoding

Embarked

Original		Encoded
`C, Q, S`	→	`0, 1, 2`

Method: Label Encoding

Why Encode?

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

Final Feature Set

Pclass — Numeric

Passenger class (1, 2, 3)

Sex — Numeric

Gender encoded (0=female, 1=male)

Age — Numeric

Age in years

SibSp — Numeric

Number of siblings/spouses

Parch — Numeric

Number of parents/children

Fare — Numeric

Use ← → arrow keys to navigate

Data Splitting

📊 Split Strategy

80%

Training Set

~713 records

20%

Testing Set

~178 records

```
train_test_split(test_size=0.2, random_state=42)
```

🎯 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

🔍 Data Structure

Input Features (X)

Pclass	Sex
Age	SibSp
Parch	Fare
Embarked	

7 features used for prediction

🎯 Target Variable (y)

Survived

0 or 1

Binary classification target

✅ Split Benefits

Use ← → arrow keys to navigate

…g: Model learns patterns

…nbiased performance evaluation

Random Forest Model

🚀 Implementation Steps

1 Import Model

```
from sklearn.ensemble import RandomForestClassifier
```

Import Random Forest classifier from sklearn

2 Initialize

```
model = RandomForestClassifier(n_estimators=100, random_state=42)
```

Create model with 100 trees and fixed random state

3 Train

```
model.fit(X_train, y_train)
```

Train ensemble of decision trees on training data

4 Predict

```
predictions = model.predict(X_test)
```

Make predictions using majority voting

⚙️ Model Characteristics

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

Results & Performance

Confusion Matrix

	Predicted: No	Predicted: Yes
Actual: No	99	6
Actual: Yes	25	48

True Positives: 48
Correctly predicted survival

True Negatives: 99
Correctly predicted death

False Positives: 6
Incorrectly predicted survival

False Negatives: 25
Missed survivors

Performance Metrics

Accuracy 82.6%
Overall correct predictions

Precision 88.9%
True positive rate

Recall 65.8%
Sensitivity

F1-Score 75.4%
Harmonic mean

Model Performance Summary

82.6%
Overall Accuracy

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

🚢 Historical Context

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

🤖 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

hm Choice

Naive Bayes proved suitable for this dataset size and feature types

Use ← → arrow keys to navigate

References

📚 Key Resources

Titanic Dataset
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

Machine Learning Course Materials
Educational Resources

Academic resources and online learning materials

🛠️ Technology Stack

Pandas 1.5+

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

• ...rn contributors for the ML framework

Use ← → arrow keys to navigate