Data Preprocessing Solution: Titanic Dataset Analysis

Overview

This comprehensive data preprocessing solution demonstrates essential techniques for cleaning and preparing the Titanic dataset for machine learning applications. The solution covers the complete workflow from initial data exploration to final preparation for model building.

Step-by-Step Implementation

1. Environment Setup and Data Loading

The solution begins with importing essential Python libraries for data manipulation and visualization:

python

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

The Titanic dataset from Kaggle serves as the primary data source for this preprocessing exercise.

2. Initial Data Exploration

Understanding the dataset structure is crucial before any preprocessing steps. The solution includes comprehensive data inspection:

python

```
df = pd.read_csv('titanic.csv')
```

Data Preprocessing Solution: Titanic Dataset Analysis

df.info()

df.describe()

df.isnull().sum()

This approach helps identify data types, statistical distributions, and missing value patterns within the dataset.

3. Missing Value Treatment

The solution implements targeted strategies for handling missing data:

- Age column: Filled with median values to maintain distribution integrity
- Embarked column: Filled with mode (most frequent value)
- Cabin column: Dropped entirely due to excessive missing values

python

df['Age'].fillna(df['Age'].median(), inplace=True)

df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)

df.drop(columns=['Cabin'], inplace=True)

4. Categorical Variable Encoding

Categorical features require numerical transformation for machine learning algorithms. The solution uses one-hot encoding:

python

```
df = pd.get_dummies(df, columns=['Sex', 'Embarked'], drop_first=True)
```

This technique converts categorical variables like 'Sex' and 'Embarked' into binary numerical columns while avoiding multicollinearity by dropping the first category.

5. Feature Scaling Implementation

Numerical features with different scales require standardization. The solution applies StandardScaler to normalize features:

```
python
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df[['Age', 'Fare']] = scaler.fit_transform(df[['Age', 'Fare']])
```

This ensures features like 'Age' and 'Fare' contribute equally to model training.

6. Outlier Detection and Removal

The solution includes robust outlier detection using the Interquartile Range (IQR) method:

```
python
sns.boxplot(df['Fare'])
# Remove outliers in 'Fare'
Q1 = df['Fare'].quantile(0.25)
```

Data Preprocessing Solution: Titanic Dataset Analysis

```
Q3 = df['Fare'].quantile(0.75)

IQR = Q3 - Q1

df = df[(df['Fare'] >= Q1 - 1.5*IQR) & (df['Fare'] <= Q3 + 1.5*IQR)]
```