exp 8

Python

```
import pandas as pd
import numpy as np

# Load the dataset
data = pd.read_csv('titanic.csv')

# Select numerical features for outlier handling
numerical_features = ['Age', 'Fare']

# Calculate IQR for each numerical feature
Q1 = data[numerical_features].quantile(0.25)
Q3 = data[numerical_features].quantile(0.75)
IQR = Q3 - Q1

# Define lower and upper bounds for outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Identify outliers
outliers = ((data[numerical_features] < lower_bound) |
(data[numerical_features] > upper_bound)).any(axis=1)

# Handle Outliers (Choose ONE of the following methods)

# 1. Remove outliers
# data = data[~outliers]

# 2. Cap outliers
# for feature in numerical_features:
#     data[feature] = np.where(data[feature] < lower_bound[feature],
lower_bound[feature], data[feature])
#     data[feature] = np.where(data[feature] > upper_bound[feature],
upper_bound[feature], data[feature])

# 3. Transform data (log transformation)
# for feature in numerical_features:
#     data[feature] = np.log1p(data[feature])  # Use log1p to handle 0
```

```
values

# Display data after handling outliers
print("Data after handling outliers:")
print(data.head())
```

AIM: To identify and handle outliers in numerical variables of the Titanic dataset using the Interquartile Range (IQR) method to improve data quality and prevent distortion of analysis.

Theory:

Outliers are data points that significantly deviate from the other observations in a dataset. They can arise due to measurement errors, data entry mistakes, or genuine extreme values. Outliers can distort statistical analyses, affect model performance, and lead to misleading conclusions.

Interquartile Range (IQR) Method:

The IQR method is a common technique for identifying outliers. It uses the following steps:

1. **Calculate Quartiles:** Calculate the first quartile (Q1) and the third quartile (Q3) of the data.
2. **Calculate IQR:** Calculate the IQR as the difference between Q3 and Q1: `IQR = Q3 - Q1`.
3. **Define Bounds:** Define the lower and upper bounds for identifying outliers:
    - `Lower Bound = Q1 - 1.5 * IQR`
    - `Upper Bound = Q3 + 1.5 * IQR`
4. **Identify Outliers:** Any data point that falls below the lower bound or above the upper bound is considered an outlier.

Handling Outliers:

- **Removal:** Removing outliers from the dataset. This can be suitable if outliers are due to errors or if they significantly distort the analysis. However, it can lead to loss of information.
- **Capping:** Replacing outliers with a specified upper or lower limit. This preserves more data than removal but can affect the data distribution.
- **Transformation:** Applying a mathematical transformation (e.g., log transformation) to the data to reduce the influence of outliers. This can be useful if the data is skewed.

Algorithm:

1. **Load the dataset:** Import the dataset into a Pandas DataFrame.
2. **Select numerical features:** Identify the numerical columns to be checked for outliers.
3. **Calculate IQR:** Calculate the IQR for each numerical feature.

4. **Define bounds:** Define the lower and upper bounds for outlier detection.
5. **Identify outliers:** Find data points that fall outside the defined bounds.
6. **Handle outliers:** Apply one of the outlier handling techniques (removal, capping, or transformation).

Output:

- The code will print the first 5 rows of the dataset after handling the outliers.

Result:

The experiment demonstrates the application of the IQR method for outlier detection and shows different techniques for handling outliers in numerical variables. The choice of outlier handling method depends on the specific dataset and the goals of the analysis.

Learning Outcomes:

- Gained practical experience in identifying and handling outliers in numerical data.
- Understood the impact of outliers on data analysis and model building.
- Applied the IQR method for outlier detection.
- Explored different techniques for dealing with outliers (removal, capping, transformation).
- Developed skills in data cleaning and preprocessing using Pandas and NumPy.