This task preprocesses the Titanic dataset by loading, inspecting, and cleaning the data, handling missing values and outliers, and transforming categorical and numerical features. The processed data is then saved for further analysis or machine learning applications.

WEEK 1

INTRODUCTION TO ML

ML-WEEK 1.2

Noor Ul Ain

TASK 1.2:DATA CLEANING AND PREPARATION

OBJECTIVE

Learn the importance of preprocessing data to prepare for any machine learning model.

DATASET

Titanic Dataset from Kaggle. This dataset includes passenger information from the Titanic, such as age, fare, cabin, survival status, etc.

• Link to dataset

Titanic Dataset on Kaggle : https://www.kaggle.com/c/titanic

ACTIVITIES

1. **Load and Inspect the Dataset**

Load the data into a pandas DataFrame.

Inspect the data for missing values, potential errors, and outliers.

2. **Data Cleaning**

Handle missing values by filling them with the median or mode, or by using other appropriate imputation methods.

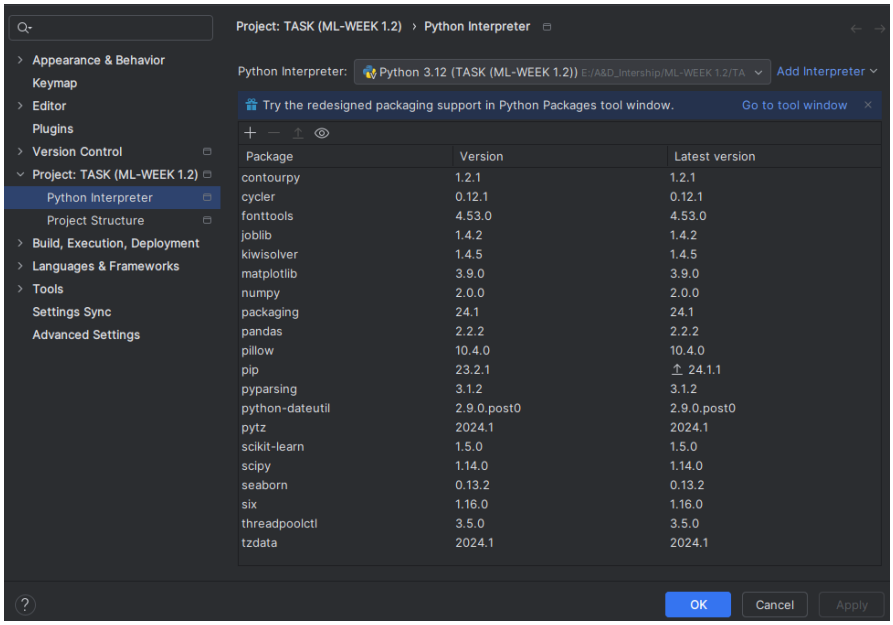
Remove outliers if necessary or treat them appropriately.

3. **Data Transformation**

Convert categorical data into numeric format using one-hot encoding or label encoding.

Normalize or standardize the numerical values if required for later modeling.

IMPORTING LIBRARIES



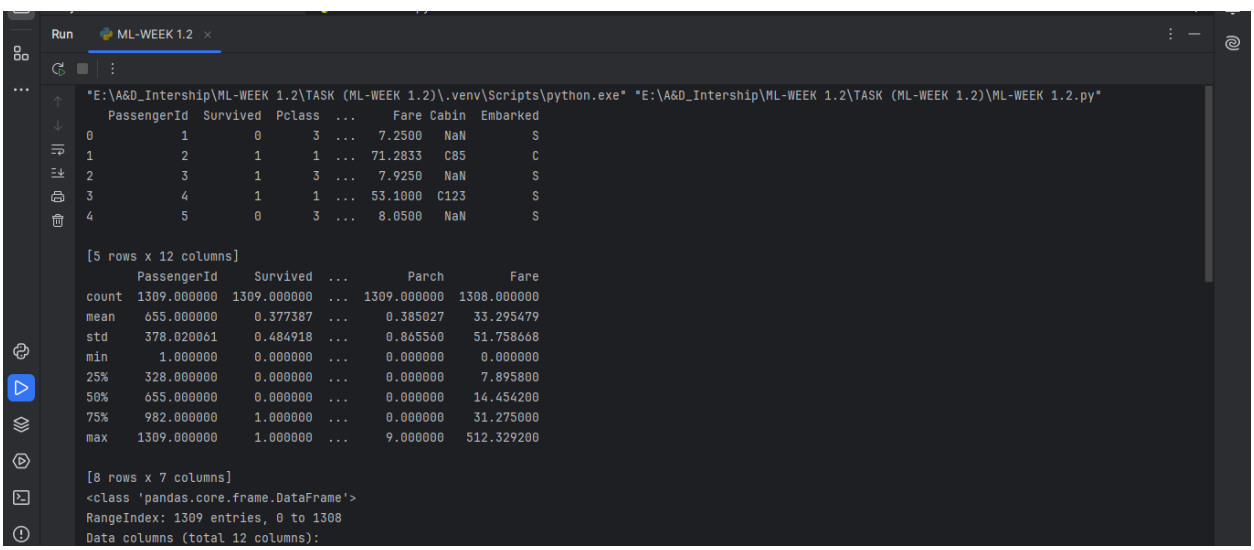
CODE

```
# <<<--------ML-WEEK 1.2---------->>>

# Importing libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# 1. Load & Inspect dataset:
train_path = r'E:\A&D_Intership\ML-WEEK 1.2\TASK (ML-WEEK 1.2)\train.csv'
test_path = r'E:\A&D_Intership\ML-WEEK 1.2\TASK (ML-WEEK 1.2)\test.csv'
gender_submission_path = r'E:\A&D_Intership\ML-WEEK 1.2\TASK (ML-WEEK
1.2)\gender_submission.csv'

df_train = pd.read_csv(train_path)
df_test = pd.read_csv(test_path)
df_gender_submission = pd.read_csv(gender_submission_path)
```

```
# Add 'Survived' column to the test set from gender_submission
df_test['Survived'] = df_gender_submission['Survived']

# Combine train and test sets
titanic_df = pd.concat([df_train, df_test], ignore_index=True)

print(titanic_df.head())
print(titanic_df.describe())
print(titanic_df.info())
print(titanic_df.isnull().sum())

# Visualize missing values
sns.heatmap(titanic_df.isnull(), cbar=False, cmap='viridis')
plt.show()

# 2. Data Cleaning
# Handle missing values by filling them with the median or mode
titanic_df['Age'] = titanic_df['Age'].fillna(titanic_df['Age'].median())
titanic_df['Embarked'] = titanic_df['Embarked'].fillna(titanic_df['Embarked'].mode()[0])
titanic_df['Fare'] = titanic_df['Fare'].fillna(titanic_df['Fare'].median())

# Drop the 'Cabin' column as it has too many missing values
titanic_df = titanic_df.drop(columns=['Cabin'])

# Check for outliers in 'Fare' using boxplot
sns.boxplot(x=titanic_df['Fare'])
plt.show()

# Handle outliers (e.g., cap them at the 99th percentile)
fare_cap = titanic_df['Fare'].quantile(0.99)
titanic_df['Fare'] = np.where(titanic_df['Fare'] > fare_cap, fare_cap, titanic_df['Fare'])

# 3. Data Transformation
# Convert categorical data into numeric format using one-hot encoding or label encoding
categorical_features = ['Sex', 'Embarked']
numeric_features = ['Age', 'Fare']

# One-hot encode categorical features
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numeric_features),
        ('cat', OneHotEncoder(), categorical_features)])
```

```
# Apply the transformations
titanic_transformed = preprocessor.fit_transform(titanic_df)

# Convert the transformed data back to a DataFrame
transformed_df = pd.DataFrame(titanic_transformed, columns=['Age', 'Fare'] +

list(preprocessor.transformers_[1][1].get_feature_names_out(categorical_features)))

# Add the target column 'Survived' to the transformed DataFrame
transformed_df['Survived'] = titanic_df['Survived'].values

print(transformed_df.head())

# Save the cleaned and transformed DataFrame to a new CSV file
output_path = r'E:\A&D_Intership\ML-WEEK 1.2\TASK (ML-WEEK 1.2)\cleaned_titanic.csv'
transformed_df.to_csv(output_path, index=False)
```

OUTPUT



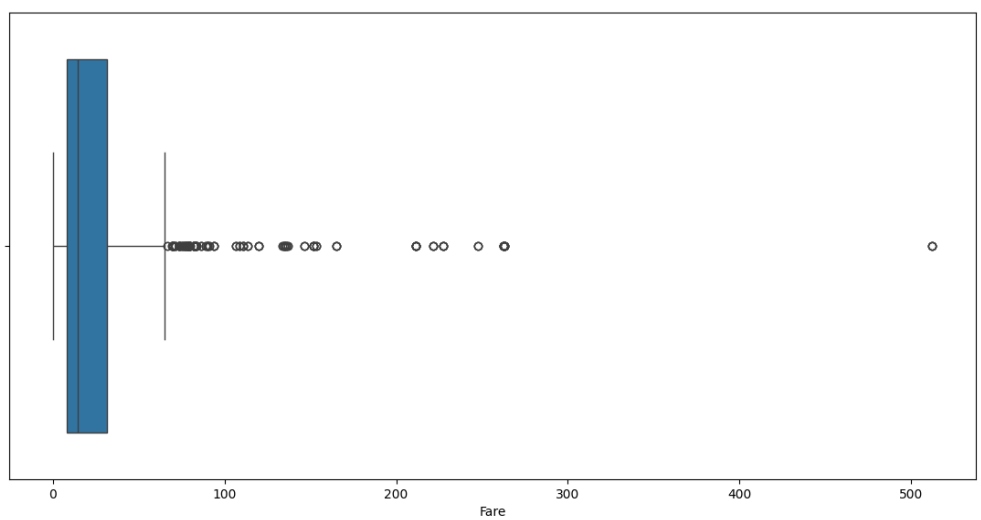
```
Run    ML-WEEK 1.2
"E:\A&D_Intership\ML-WEEK 1.2\TASK (ML-WEEK 1.2)\.venv\Scripts\python.exe" "E:\A&D_Intership\ML-WEEK 1.2\TASK (ML-WEEK 1.2)\ML-WEEK 1.2.py"
   PassengerId  Survived  Pclass  ...     Fare Cabin  Embarked
0            1         0       3  ...   7.2500   NaN         S
1            2         1       1  ...  71.2833   C85         C
2            3         1       3  ...   7.9250   NaN         S
3            4         1       1  ...  53.1000  C123         S
4            5         0       3  ...   8.0500   NaN         S

[5 rows x 12 columns]
       PassengerId     Survived  ...        Parch         Fare
count  1309.000000  1309.000000  ...  1309.000000  1308.000000
mean    655.000000     0.377387  ...     0.385027    33.295479
std     378.020061     0.484918  ...     0.865560    51.758668
min       1.000000     0.000000  ...     0.000000     0.000000
25%     328.000000     0.000000  ...     0.000000     7.895800
50%     655.000000     0.000000  ...     0.000000    14.454200
75%     982.000000     1.000000  ...     0.000000    31.275000
max    1309.000000     1.000000  ...     9.000000   512.329200

[8 rows x 7 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 12 columns):
```

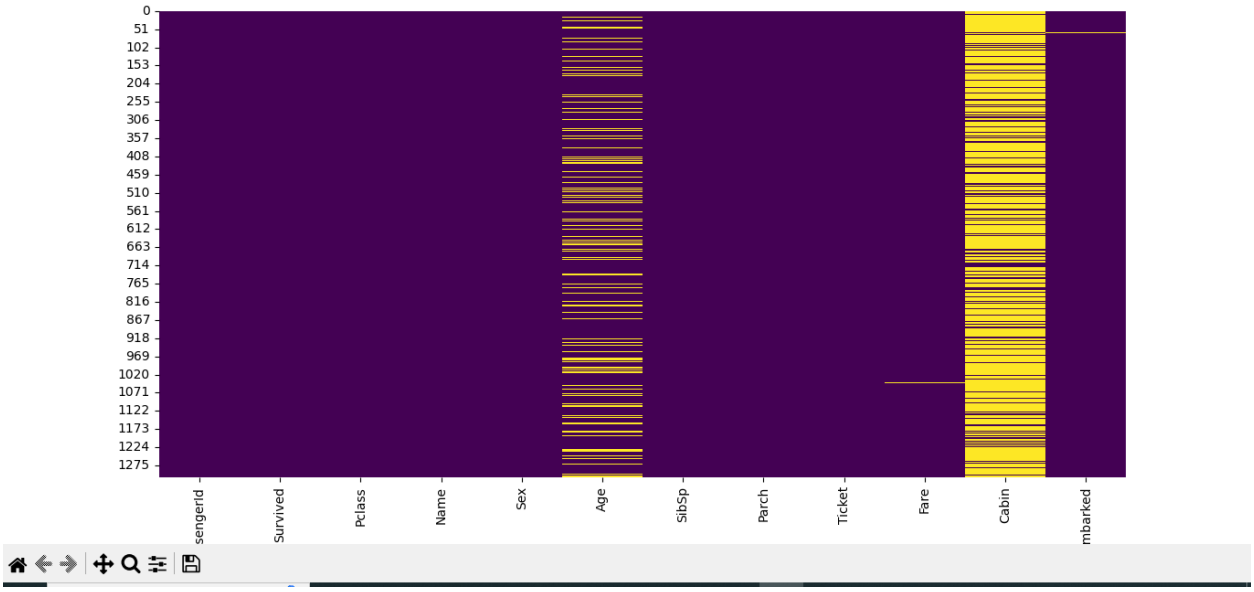
```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  1309 non-null   int64
 1   Survived     1309 non-null   int64
 2   Pclass       1309 non-null   int64
 3   Name         1309 non-null   object
 4   Sex          1309 non-null   object
 5   Age          1046 non-null   float64
 6   SibSp        1309 non-null   int64
 7   Parch        1309 non-null   int64
 8   Ticket       1309 non-null   object
 9   Fare         1308 non-null   float64
 10  Cabin        295 non-null    object
 11  Embarked     1307 non-null   object
dtypes: float64(2), int64(5), object(5)
memory usage: 122.8+ KB
None
PassengerId       0
Survived          0
Pclass            0
Name              0
Sex               0
```

TASK (ML-WEEK 1.2) > ML-WEEK 1.2.py 77:1 CRLF UTF-8 4 spaces Python 3.12 (TASK (ML-WEEK 1.2))

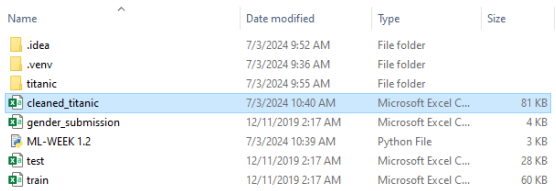
Run ML-WEEK 1.2

```
None
PassengerId       0
Survived          0
Pclass            0
Name              0
Sex               0
Age             263
SibSp             0
Parch             0
Ticket            0
Fare              1
Cabin          1014
Embarked          2
dtype: int64
        Age      Fare  Sex_female  ...  Embarked_Q  Embarked_S  Survived
0 -0.581628 -0.547116         0.0  ...         0.0         1.0         0
1  0.658652  0.839563         1.0  ...         0.0         0.0         1
2 -0.271558 -0.532499         1.0  ...         0.0         1.0         1
3  0.426099  0.445793         1.0  ...         0.0         1.0         1
4  0.426099 -0.529792         0.0  ...         0.0         1.0         0

[5 rows x 8 columns]

Process finished with exit code 0
```





New csv file named **cleaned_titanic** is created



Name	Date modified	Type	Size
.idea	7/3/2024 9:52 AM	File folder	
.venv	7/3/2024 9:36 AM	File folder	
titanic	7/3/2024 9:55 AM	File folder	
cleaned_titanic	7/3/2024 10:40 AM	Microsoft Excel C...	81 KB
gender_submission	12/11/2019 2:17 AM	Microsoft Excel C...	4 KB
ML-WEEK 1.2	7/3/2024 10:39 AM	Python File	3 KB
test	12/11/2019 2:17 AM	Microsoft Excel C...	28 KB
train	12/11/2019 2:17 AM	Microsoft Excel C...	60 KB

EXPLANATION

LIBRARIES USED

- `pandas` (pd): For data manipulation and analysis.
- `matplotlib.pyplot` (plt) and `seaborn` (sns): For data visualization.
- `numpy` (np): For numerical operations.
- `sklearn.preprocessing`: For data preprocessing (standard scaling and one-hot encoding).
- `sklearn.compose`: For combining multiple transformations.
- `sklearn.pipeline`: For creating a machine learning pipeline.

This code performs several essential tasks in data preprocessing for a Titanic dataset analysis. It begins by loading the training, testing, and gender submission datasets from specified file paths into data structures suitable for manipulation. After loading, it appends the 'Survived' column from the gender submission data to the test set and merges the train and test datasets into a single comprehensive dataset for easier handling.

The next step involves inspecting the combined dataset by displaying its first few rows, summary statistics, data types, and missing values count. A heatmap is used to visualize the locations of any missing values within the dataset. Following this, the code addresses the missing data by filling missing values in the 'Age' column with the median, in the 'Embarked' column with the mode, and in the 'Fare' column with the median. It also drops the 'Cabin' column due to its high proportion of missing values.

To identify and manage outliers, particularly in the 'Fare' column, a boxplot visualization is created. The code caps extreme outliers in the 'Fare' column at the 99th percentile to mitigate their impact.

In the data transformation phase, categorical features such as 'Sex' and 'Embarked' are converted into a numeric format using one-hot encoding, while numeric features like 'Age' and 'Fare' are standardized. These transformations are applied using a column transformer that combines scaling for numeric features and encoding for categorical features. The transformed data is then compiled back into a structured format, with appropriate column names and the 'Survived' target variable reintroduced.

Finally, the cleaned and transformed dataset is saved to a new file, ensuring that the data is ready for subsequent analysis or machine learning model training.