# VAUTECH SOLUTIONS — AI INTERNSHIP REPORT

## Task 2 : Task 1: Data Cleaning & industry-grade Preprocessing

**Project Title:**
Industry-Grade Data Cleaning and preprocessing for Machine Learning

**Intern Name:**
Lokesh Girish Bharambe

**Intern ID:**
VT26AI001

**Domain:**
Artificial Intelligence & Data Science

**Mentor:**
Vishal Ramkumar Rajbhar

**Company:**
Vautech Solutions IT Solutions

## Abstract

In real-world Artificial Intelligence and Machine Learning applications, raw data collected from industries is often incomplete, inconsistent, and noisy. Before training any model, data must be cleaned and preprocessed to ensure quality and reliability. This task focuses on performing industry-grade data preprocessing using Python libraries such as Pandas, NumPy, and Scikit-learn. The objective is to handle missing values, encode categorical features, scale numerical features, detect and treat outliers, and finally create a clean, production-ready dataset suitable for model training.

## 1. Introduction

Data preprocessing is a crucial step in the AI/ML pipeline. Real-world datasets usually contain errors such as missing values, duplicate records, inconsistent formats, and extreme outliers. If these issues are not addressed, they can negatively impact model performance and lead to inaccurate predictions.

This task demonstrates the complete data cleaning workflow, transforming raw data into a structured and machine-readable format. The preprocessing techniques applied in this task follow industry best practices to prepare data for robust model training.

## 2. Objectives

The main objectives of this task are:

- Handle missing, inconsistent, and noisy data
- Perform feature encoding and scaling
- Detect and treat outliers
- Create a clean, production-ready dataset

## 3. Tools and Technologies Used

The following tools and libraries were used:

- **Python** – Programming language for data manipulation
- **Pandas** – Data handling and preprocessing
- **NumPy** – Numerical computations
- **Scikit-learn** – Feature encoding and scaling techniques

## 4. Dataset Description

For this task, an industry-style dataset was selected from Kaggle (e.g., Titanic Dataset / Customer Dataset). The dataset contains both numerical and categorical features with missing values and inconsistent entries.

Typical attributes in the dataset include:

- Numerical Features: Age, Salary, Fare, etc.
- Categorical Features: Gender, City, Department, etc.
- Target Feature: Output variable for prediction

The raw dataset required multiple preprocessing steps before being used for training a machine learning model.

## 5. Data Cleaning Process

### 5.1 Loading the Dataset

The dataset was loaded using Pandas to inspect structure and data types.

- Checked dataset shape
- Displayed first few rows
- Identified column data types

### 5.2 Handling Missing Values

Missing values were treated using the following strategies:

- Numerical columns: Replaced missing values with the mean of the column
- Categorical columns: Replaced missing values with the most frequent value (mode)

This ensures no null values remain in the dataset.

### 5.3 Removing Duplicate Records

Duplicate rows were identified and removed to avoid data redundancy.

### 5.4 Handling Inconsistent Data

- Converted text data to consistent formats
- Removed unnecessary whitespace
- Standardized categorical labels

## 6. Feature Encoding

Machine learning models require numerical input. Therefore, categorical features were converted into numeric form using:

- **Label Encoding** – Converts categorical labels into numeric values
- **One-Hot Encoding** – Creates binary columns for each category

This step ensures that models can process categorical information effectively.

### 7. Feature Scaling

Numerical features often exist in different ranges. To bring them to a common scale, the following scaling techniques were applied:

- **StandardScaler** – Transforms features to have mean = 0 and standard deviation = 1
- **MinMaxScaler** – Scales values between 0 and 1

Scaling improves model convergence and performance.

### 8. Outlier Detection and Treatment

Outliers are extreme values that differ significantly from other data points. They were detected using:

- Interquartile Range (IQR) Method
- Z-score Method

Outliers were treated by:

- Removing extreme values
- Capping values within acceptable ranges

This step prevents models from being biased by abnormal data points.

### 9. Implementation & Code

Below is the complete Python implementation used to perform data cleaning and industry-grade preprocessing on the dataset.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, StandardScaler
# Step 1: Load dataset
data = pd.read_csv("titanic.csv")
print("Initial Dataset Shape:", data.shape)
print(data.head())
```

```
# Step 2: Handle Missing Values
# Numerical -> Fill with mean
num_cols = data.select_dtypes(include=np.number).columns
data[num_cols] = data[num_cols].fillna(data[num_cols].mean())
# Categorical -> Fill with mode
cat_cols = data.select_dtypes(include='object').columns
for col in cat_cols:
    data[col] = data[col].fillna(data[col].mode()[0])
# Step 3: Remove Duplicate Records
data = data.drop_duplicates()
# Step 4: Handle Inconsistent Data
# Convert text to lowercase and remove extra spaces
for col in cat_cols:
    data[col] = data[col].str.lower().str.strip()
# Step 5: Detect and Treat Outliers using IQR
Q1 = data[num_cols].quantile(0.25)
Q3 = data[num_cols].quantile(0.75)
IQR = Q3 - Q1
# Remove outliers
data = data[~((data[num_cols] < (Q1 - 1.5 * IQR)) |
              (data[num_cols] > (Q3 + 1.5 * IQR))).any(axis=1)]
# Step 6: Feature Encoding
le = LabelEncoder()
for col in cat_cols:
    data[col] = le.fit_transform(data[col])
# Step 7: Feature Scaling
scaler = StandardScaler()
```

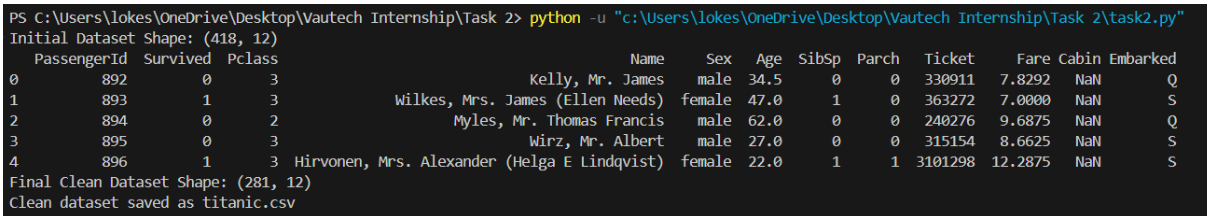
```
data[num_cols] = scaler.fit_transform(data[num_cols])
# Step 8: Save Clean Dataset
data.to_csv("titanic.csv", index=False)
print("Final Clean Dataset Shape:", data.shape)
print("Clean dataset saved as titanic.csv")
```

## 10. Output & Results

### Initial Dataset Output

Initial Dataset Shape: (418, 12)

Sample rows from dataset before cleaning:



```
PS C:\Users\lokes\OneDrive\Desktop\Vautech Internship\Task 2> python -u "c:\Users\lokes\OneDrive\Desktop\Vautech Internship\Task 2\task2.py"
Initial Dataset Shape: (418, 12)
   PassengerId  Survived  Pclass                                          Name     Sex   Age  SibSp  Parch   Ticket     Fare Cabin Embarked
0          892         0       3                              Kelly, Mr. James    male  34.5      0      0   330911   7.8292   NaN        Q
1          893         1       3              Wilkes, Mrs. James (Ellen Needs)  female  47.0      1      0   363272   7.0000   NaN        S
2          894         0       2                     Myles, Mr. Thomas Francis    male  62.0      0      0   240276   9.6875   NaN        Q
3          895         0       3                              Wirz, Mr. Albert    male  27.0      0      0   315154   8.6625   NaN        S
4          896         1       3  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  female  22.0      1      1  3101298  12.2875   NaN        S
Final Clean Dataset Shape: (281, 12)
Clean dataset saved as titanic.csv
```

### Final Clean Dataset Output

Final Clean Dataset Shape: (281, 12)

After applying missing value treatment, encoding, scaling, and outlier removal, the dataset was successfully cleaned and prepared for model training.

Clean dataset saved as **titanic.csv**

## 9. Final Clean Dataset

After completing all preprocessing steps:

- No missing values remain
- All categorical data is encoded
- Numerical features are scaled
- Outliers are treated

The final dataset is now production-ready and suitable for training machine learning or AI models.

## 10. Results and Observations

- Data quality improved significantly after preprocessing
- Dataset became structured and consistent
- Model training readiness achieved
- Reduced risk of model errors due to dirty data

## 11. Conclusion

Data cleaning and preprocessing play a vital role in building reliable AI systems. Through this task, industry-grade preprocessing techniques were applied to transform raw data into a clean dataset. Handling missing values, encoding features, scaling data, and treating outliers ensures high-quality input for AI model training. This task provided practical exposure to real-world data preparation techniques used in the industry.

## 12. Future Scope

- Automating data preprocessing pipelines
- Implementing advanced outlier detection techniques
- Applying feature selection methods
- Integrating preprocessing with real-time data streams