**1.**

**Data cleaning and pre-processing:** At this phase, the data are prepared for analysis by being examined for missing values, outliers, and irregularities. By making sure the data is reliable, consistent, and comprehensive, this step tries to prepare the data for analysis. You may carry out the following tasks at this stage:

• Look for any missing values, then add them or recalculate them.

• Look for outliers and respond to them appropriately.

• Verify and rectify any data irregularities, such as misspellings.

• Rename columns and choose pertinent options

• If necessary, convert data types. The folder structure for this stage can be organized as follows:

Text

Description automatically generated with low confidence

**Exploratory data analysis:** To get insights and identify patterns, the data must be explored at this point. The objective of this phase is to comprehend the data better and produce hypotheses for additional investigation. You may carry out the following tasks at this stage:

Calculate statistical summaries.

Visualize your ideas

Identification of trends and relationships

Look for suppositions.

Create hypotheses for additional research. The folder structure for this stage can be organized as follows:

Text

Description automatically generated

**Modelling and inference:** Building models and testing theories are part of this phase. Making predictions and deriving inferences from the data are the goals of this stage. You may carry out the following tasks at this stage:

Choose suitable models.

test your theories.

Verify models

Make assumptions.

The folder structure for this stage can be organized as follows:

Shape

Description automatically generated with medium confidence

**Reproducible workflow:**

import pandas as pd

import os

# Set up folder structure

data\_folder = 'data'

raw\_data\_folder = os.path.join(data\_folder, 'raw\_data')

cleaned\_data\_folder = os.path.join(data\_folder, 'cleaned\_data')

analysis\_folder = 'analysis'

exploratory\_folder = os.path.join(analysis\_folder, 'exploratory')

hypotheses\_folder = os.path.join(analysis\_folder, 'hypotheses')

if not os.path.exists(raw\_data\_folder):

os.makedirs(raw\_data\_folder)

if not os.path.exists(cleaned\_data\_folder):

os.makedirs(cleaned\_data\_folder)

if not os.path.exists(exploratory\_folder):

os.makedirs(exploratory\_folder)

if not os.path.exists(hypotheses\_folder):

os.makedirs(hypotheses\_folder)

# Load raw data

raw\_data = pd.read\_csv(os.path.join(raw\_data\_folder, 'my\_data.csv'))

# Data cleaning and preprocessing

cleaned\_data = raw\_data.dropna() # remove missing values

# Save cleaned data

cleaned\_data.to\_csv(os.path.join(cleaned\_data\_folder, 'my\_data\_cleaned.csv'))

# Exploratory data analysis

summary\_stats = cleaned\_data.describe() # compute summary statistics

summary\_stats.to\_csv(os.path.join(exploratory\_folder, 'my\_data\_summary\_stats.csv'))

import matplotlib.pyplot as plt

import seaborn as sns

# Create visualizations

sns.scatterplot(data=cleaned\_data, x='study\_time', y='final\_grade')

plt.savefig(os.path.join(exploratory\_folder, 'my\_data\_visualizations', 'scatterplot.png'))

sns.histplot(data=cleaned\_data, x='final\_grade')

plt.savefig(os.path.join(exploratory\_folder, 'my\_data\_visualizations', 'histogram.png'))

sns.barplot(data=cleaned\_data, x='gender', y='final\_grade')

plt.savefig(os.path.join(exploratory\_folder, 'my\_data\_visualizations', 'barplot.png'))

sns.boxplot(data=cleaned\_data, y='final\_grade')

plt.savefig(os.path.join(exploratory\_folder, 'my\_data\_visualizations', 'boxplot.png'))

# Modeling and inference

from sklearn.linear\_model import LinearRegression

# Fit linear regression model

model = LinearRegression()

model.fit(cleaned\_data[['study\_time']], cleaned\_data['final\_grade'])

# Make predictions

predictions = model.predict(cleaned\_data[['study\_time']])

results = pd.DataFrame({'actual': cleaned\_data['final\_grade'], 'predicted': predictions})

results.to\_csv(os.path.join(hypotheses\_folder, 'my\_hypothesis\_results.csv'))