PROJECT TITLE

"OPTITANKS: PREDICTIVE ULLAGE MODELING AND REFILL OPTIMIZATION FOR EFFICIENT TANK MANAGEMENT"

SUBMITTED BY:

JAGANNATH DUTTA

COLLEGE: - HALDIA INSTITUTE OF TECHNOLOGY, HALDIA

UNDER THE GUIDANCE OF:

Mr. DEBAL BISWAS, CHIEF MANAGER (Information System)

Mr. SAUMIK BARUA, MANAGER (Information System)

ACKNOWLEDGEMENT

I would like to take this opportunity to thank everyone whose cooperation and encouragement throughout the on-going course of this project remains invaluable to us.

We are sincerely grateful to our guide **Mr. DEBAL BISWAS, CHIEF MANAGER (Information System) and Mr. SAUMIK BARUA, MANAGER (Information System)** for their wisdom; guidance and inspiration that helped us to go through with this project and take it to where it stands now. Last but not the least, we would like to extend our warm regards to our families and peers who have kept supporting us and always had faith in our work.

JAGANNATH DUTTA

TABLE OF CONTENTS

- SUMMARY 4
- INTRODUCTION 5
- OBJECTIVE 6
- SYSTEM ARCHITECTURE 7
- FEATURES 8

  1.1 CREATING THE SAMPLE CSV FILE FOR TANK ULLAGE PREDICTION 8

  1.2 TANK ULLAGE PREDICTION MODEL (CODE) 10

  1.3 TANK ULLAGE PREDICTION MODE (OUTPUT) 12

  1.4 CREATING THE SAMPLE CSV FILE FOR TANK OPTIMAL FILLING PREDICTION 13

  1.5 TANK OPTIMAL REFILLING PREDICTION MODEL (CODE) 15

  1.6 TANK OPTIMAL REFILLING PREDICTION MODEL (OUTPUT) 18

- CHALLENGES AND SOLUTIONS 20
- CONCLUSION 21
- FUTURE SCOPE 22
- ACKNOWLEDGEMENT 23

SUMMARY

This project aims to develop two predictive models for optimizing operations in a tank management system: Tank Ullage Prediction and Tank Refill Optimization Prediction.

1. **Tank Ullage Prediction:**
    - Utilizes temperature, humidity, pressure, and fill level as features to predict tank ullage (the empty space in the tank).
    - A Linear Regression model is trained on historical data to predict ullage values.
    - Model performance is evaluated using Mean Squared Error (MSE) and R-squared (R^2) score.
    - Actual and predicted ullage values are visualized using a scatter plot for comparison.
2. **Tank Refill Optimization Prediction:**
    - Considers factors such as petrol level, diesel level, petrol drain rate, diesel drain rate, and time since the last refill to predict the optimal refill amount for both tanks.
    - Synthetic time data is generated to incorporate the time since the last refill.
    - Another Linear Regression model is trained on historical data to predict optimal refill amounts.
    - Model performance is evaluated using MSE and visualized using scatter plots comparing actual vs. predicted refill amounts.
    - Additionally, the distribution of actual and predicted refill amounts and the distribution of residuals are plotted for further analysis.

In both parts of the project, Linear Regression models are employed for simplicity. However, more complex models or algorithms could be explored for potential improvements. Moreover, incorporating actual time data and refining feature engineering based on domain knowledge could enhance the models' accuracy and relevance to real-world scenarios. Overall, these predictive models offer insights and tools for optimizing tank management operations, ensuring efficient utilization of resources and minimizing costs.

INTRODUCTION

Effective management of tank systems, particularly in industries such as petroleum, chemicals, and logistics, is crucial for ensuring operational efficiency and cost-effectiveness. This project focuses on developing predictive models to optimize two key aspects of tank management: tank ullage prediction and tank refill optimization.

Tank ullage, which represents the empty space in a tank, is a critical parameter to monitor to prevent overfilling or underutilization of tank capacity. By accurately predicting tank ullage based on environmental factors and fill level, operators can optimize inventory management and avoid costly disruptions.

Additionally, optimizing the refill process for tanks, especially when dealing with multiple tanks containing different substances (e.g., petrol and diesel), is essential for maintaining continuous operations while minimizing refill costs. By considering factors such as current levels, drain rates, and time since the last refill, predictive models can recommend the optimal refill amounts for each tank, ensuring sufficient inventory without unnecessary waste or downtime.

In this project, we will explore historical data on tank operations, including environmental conditions, fill levels, refill amounts, and time intervals between refills. We will develop machine learning models, specifically Linear Regression, to predict tank ullage and optimize refill amounts. The performance of these models will be evaluated using metrics such as Mean Squared Error and R-squared score. Additionally, visualizations will be utilized to analyze the distribution of actual and predicted values, providing insights into the models' accuracy and potential areas for improvement.

Overall, this project aims to provide practical tools and insights for optimizing tank management operations, ultimately enhancing operational efficiency, reducing costs, and ensuring uninterrupted supply chain operations.

OBJECTIVE

This project aims to develop predictive models to optimize tank management operations with the following objectives:

1. **Tank Ullage Prediction:**
   - Develop a model to estimate tank ullage based on environmental factors and fill level.
   - Facilitate effective inventory management and prevent overfilling or underutilization of tank capacity.
2. **Tank Refill Optimization Prediction:**
   - Develop a model to recommend optimal refill amounts considering current levels, drain rates, and time since the last refill.
   - Minimize refill costs while ensuring continuous operations and sufficient inventory levels.
3. **Model Evaluation and Visualization:**
   - Evaluate model performance using metrics like Mean Squared Error and R-squared score.
   - Visualize predicted vs. actual values and analyse residuals to assess model accuracy and identify areas for improvement.
4. **Practical Application:**
   - Provide actionable insights for stakeholders to optimize inventory management, reduce costs, and ensure uninterrupted supply chain operations.
   - Enable informed decision-making regarding tank refilling schedules and inventory levels based on model predictions.

SYSTEM ARCHITECTURE

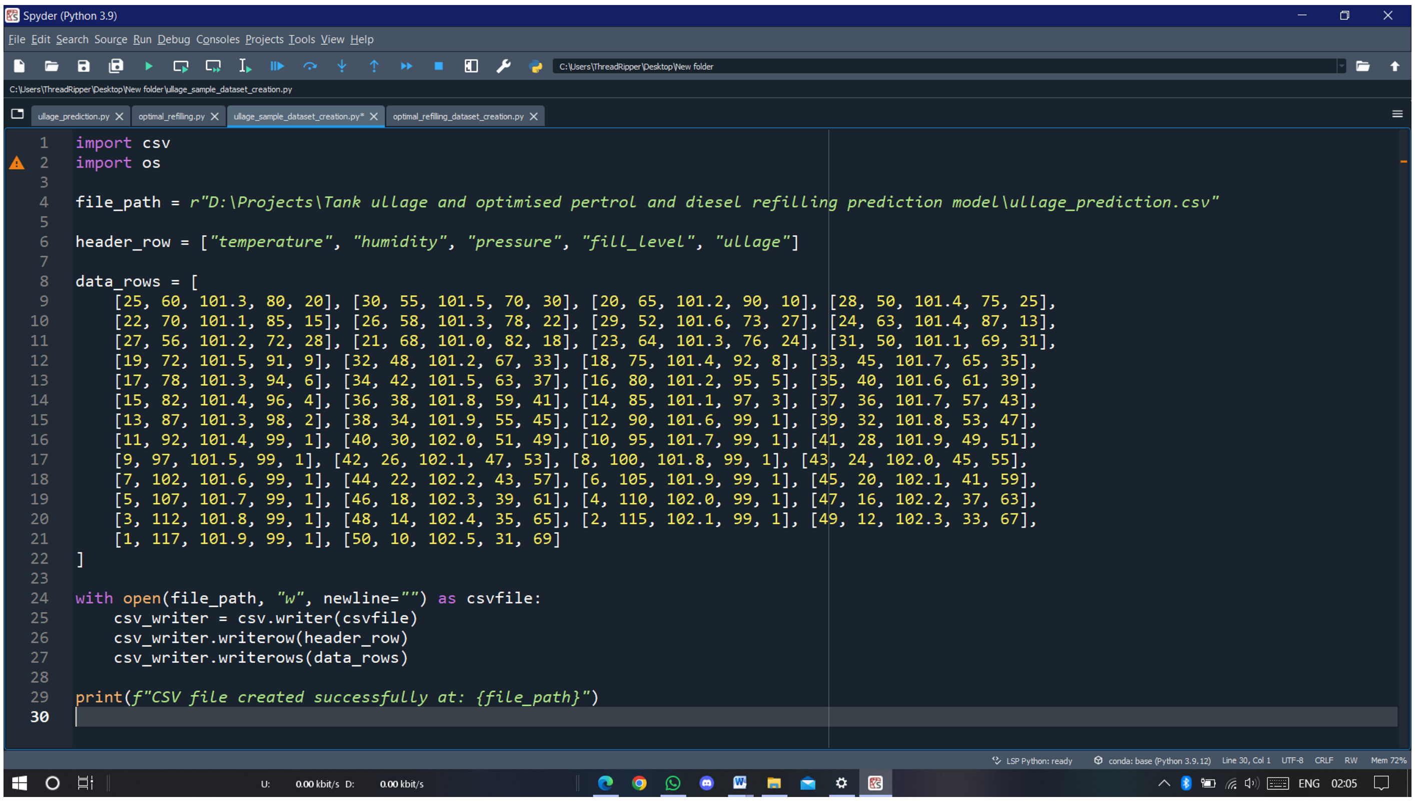
The architectural backbone of the this project is a sophisticated multi-tier system meticulously designed for optimal performance and scalability.

1. **Data Collection:** Collect tank operations data from sensors or historical records.
2. **Pre-processing:** Clean and pre-process data to handle missing values and outliers, and engineer relevant features.
3. **Ullage Prediction Module:** Use machine learning (e.g., Linear Regression) to predict tank ullage based on environmental factors and fill level.
4. **Refill Optimization Module:** Develop models to recommend optimal refill amounts considering current levels, drain rates, and time since the last refill.
5. **Evaluation and Validation:** Assess model performance using metrics like MSE and R-squared score, and validate with cross-validation.
6. **Visualization and Analysis:** Generate visualizations to compare predicted vs. actual values and analyze residuals.
7. **Deployment and Integration:** Deploy models into production, integrate with the tank management system, and develop APIs/interfaces.
8. **Monitoring and Maintenance:** Continuously monitor model performance; conduct maintenance, and update models as needed.
9. **Feedback Loop:** Gather feedback from stakeholders, incorporate improvements iteratively, and optimize tank management operations.

FEATURES

1.1 Creating the sample CSV file for tank ullage prediction

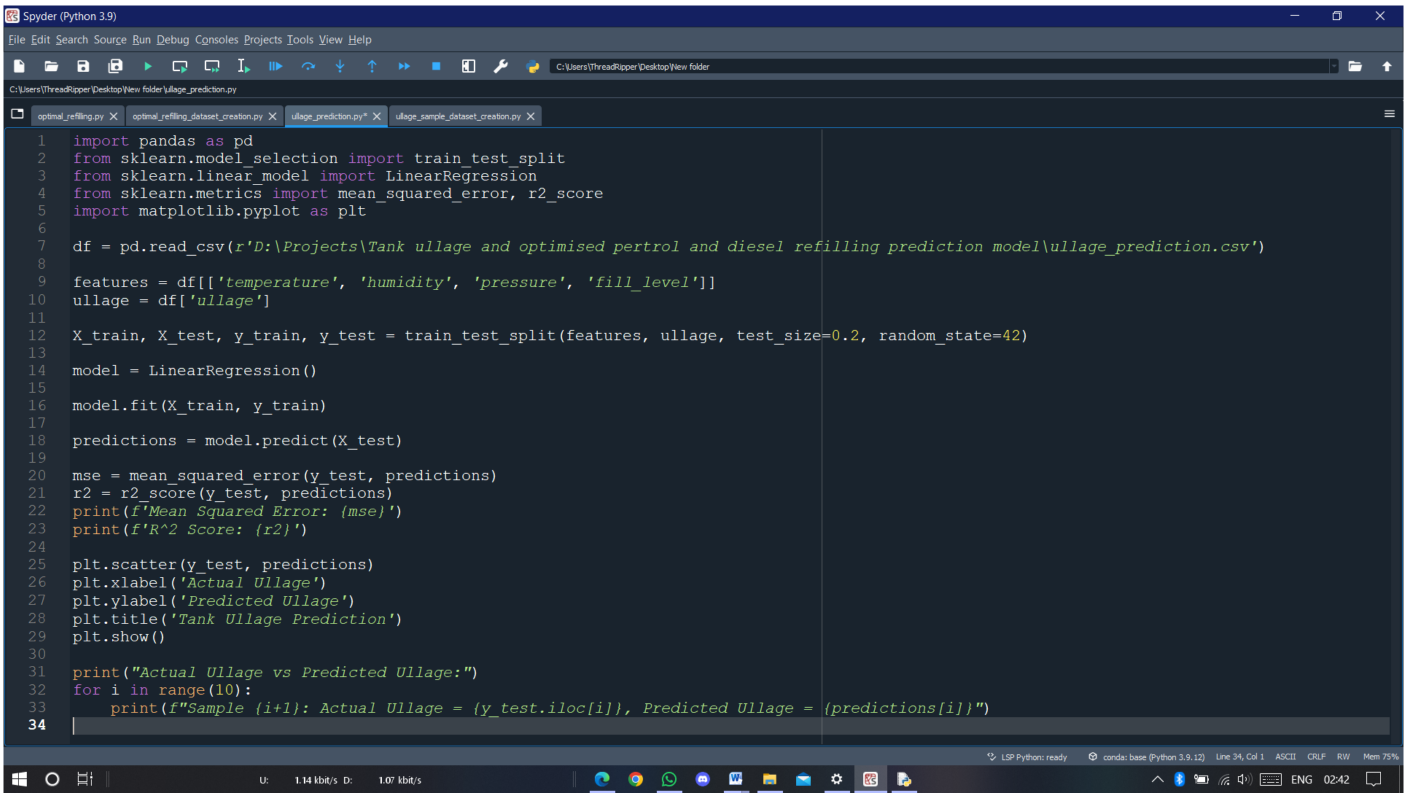
```
import csv
import os

file_path = r"D:\Projects\Tank ullage and optimised pertrol and
diesel refilling prediction model\ullage_prediction.csv"

header_row = ["temperature", "humidity", "pressure", "fill_level",
"ullage"]

data_rows = [
[25, 60, 101.3, 80, 20], [30, 55, 101.5, 70, 30], [20, 65, 101.2, 90, 10],
[28, 50, 101.4, 75, 25],
[22, 70, 101.1, 85, 15], [26, 58, 101.3, 78, 22], [29, 52, 101.6, 73, 27],
[24, 63, 101.4, 87, 13],
[27, 56, 101.2, 72, 28], [21, 68, 101.0, 82, 18], [23, 64, 101.3, 76, 24],
[31, 50, 101.1, 69, 31],
[19, 72, 101.5, 91, 9], [32, 48, 101.2, 67, 33], [18, 75, 101.4, 92, 8], [33,
45, 101.7, 65, 35],
[17, 78, 101.3, 94, 6], [34, 42, 101.5, 63, 37], [16, 80, 101.2, 95, 5], [35,
40, 101.6, 61, 39],
[15, 82, 101.4, 96, 4], [36, 38, 101.8, 59, 41], [14, 85, 101.1, 97, 3], [37,
36, 101.7, 57, 43],
[13, 87, 101.3, 98, 2], [38, 34, 101.9, 55, 45], [12, 90, 101.6, 99, 1], [39,
32, 101.8, 53, 47],
[11, 92, 101.4, 99, 1], [40, 30, 102.0, 51, 49], [10, 95, 101.7, 99, 1], [41,
28, 101.9, 49, 51],
[9, 97, 101.5, 99, 1], [42, 26, 102.1, 47, 53], [8, 100, 101.8, 99, 1], [43,
24, 102.0, 45, 55],
[7, 102, 101.6, 99, 1], [44, 22, 102.2, 43, 57], [6, 105, 101.9, 99, 1], [45,
20, 102.1, 41, 59],
[5, 107, 101.7, 99, 1], [46, 18, 102.3, 39, 61], [4, 110, 102.0, 99, 1], [47,
16, 102.2, 37, 63],
[3, 112, 101.8, 99, 1], [48, 14, 102.4, 35, 65], [2, 115, 102.1, 99, 1], [49,
12, 102.3, 33, 67],
[1, 117, 101.9, 99, 1], [50, 10, 102.5, 31, 69]
]

with open(file_path, "w", newline="") as csvfile:
csv_writer = csv.writer(csvfile)
csv_writer.writerow(header_row)
csv_writer.writerows(data_rows)

print(f"CSV file created successfully at: {file_path}")
```


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\ThreadRipper\Desktop\New folder
C:\Users\ThreadRipper\Desktop\New folder\ullage_sample_dataset_creation.py
ullage_prediction.py
optimal_refilling.py
ullage_sample_dataset_creation.py*
optimal_refilling_dataset_creation.py
LSP Python: ready
conda: base (Python 3.9.12)
Line 30, Col 1
UTF-8
CRLF
RW
Mem 72%
ENG
02:05

```python
import csv
import os

file_path = r"D:\Projects\Tank ullage and optimised pertrol and diesel refilling prediction model\ullage_prediction.csv"

header_row = ["temperature", "humidity", "pressure", "fill_level", "ullage"]

data_rows = [
    [25, 60, 101.3, 80, 20], [30, 55, 101.5, 70, 30], [20, 65, 101.2, 90, 10], [28, 50, 101.4, 75, 25],
    [22, 70, 101.1, 85, 15], [26, 58, 101.3, 78, 22], [29, 52, 101.6, 73, 27], [24, 63, 101.4, 87, 13],
    [27, 56, 101.2, 72, 28], [21, 68, 101.0, 82, 18], [23, 64, 101.3, 76, 24], [31, 50, 101.1, 69, 31],
    [19, 72, 101.5, 91, 9], [32, 48, 101.2, 67, 33], [18, 75, 101.4, 92, 8], [33, 45, 101.7, 65, 35],
    [17, 78, 101.3, 94, 6], [34, 42, 101.5, 63, 37], [16, 80, 101.2, 95, 5], [35, 40, 101.6, 61, 39],
    [15, 82, 101.4, 96, 4], [36, 38, 101.8, 59, 41], [14, 85, 101.1, 97, 3], [37, 36, 101.7, 57, 43],
    [13, 87, 101.3, 98, 2], [38, 34, 101.9, 55, 45], [12, 90, 101.6, 99, 1], [39, 32, 101.8, 53, 47],
    [11, 92, 101.4, 99, 1], [40, 30, 102.0, 51, 49], [10, 95, 101.7, 99, 1], [41, 28, 101.9, 49, 51],
    [9, 97, 101.5, 99, 1], [42, 26, 102.1, 47, 53], [8, 100, 101.8, 99, 1], [43, 24, 102.0, 45, 55],
    [7, 102, 101.6, 99, 1], [44, 22, 102.2, 43, 57], [6, 105, 101.9, 99, 1], [45, 20, 102.1, 41, 59],
    [5, 107, 101.7, 99, 1], [46, 18, 102.3, 39, 61], [4, 110, 102.0, 99, 1], [47, 16, 102.2, 37, 63],
    [3, 112, 101.8, 99, 1], [48, 14, 102.4, 35, 65], [2, 115, 102.1, 99, 1], [49, 12, 102.3, 33, 67],
    [1, 117, 101.9, 99, 1], [50, 10, 102.5, 31, 69]
]

with open(file_path, "w", newline="") as csvfile:
    csv_writer = csv.writer(csvfile)
    csv_writer.writerow(header_row)
    csv_writer.writerows(data_rows)

print(f"CSV file created successfully at: {file_path}")

```

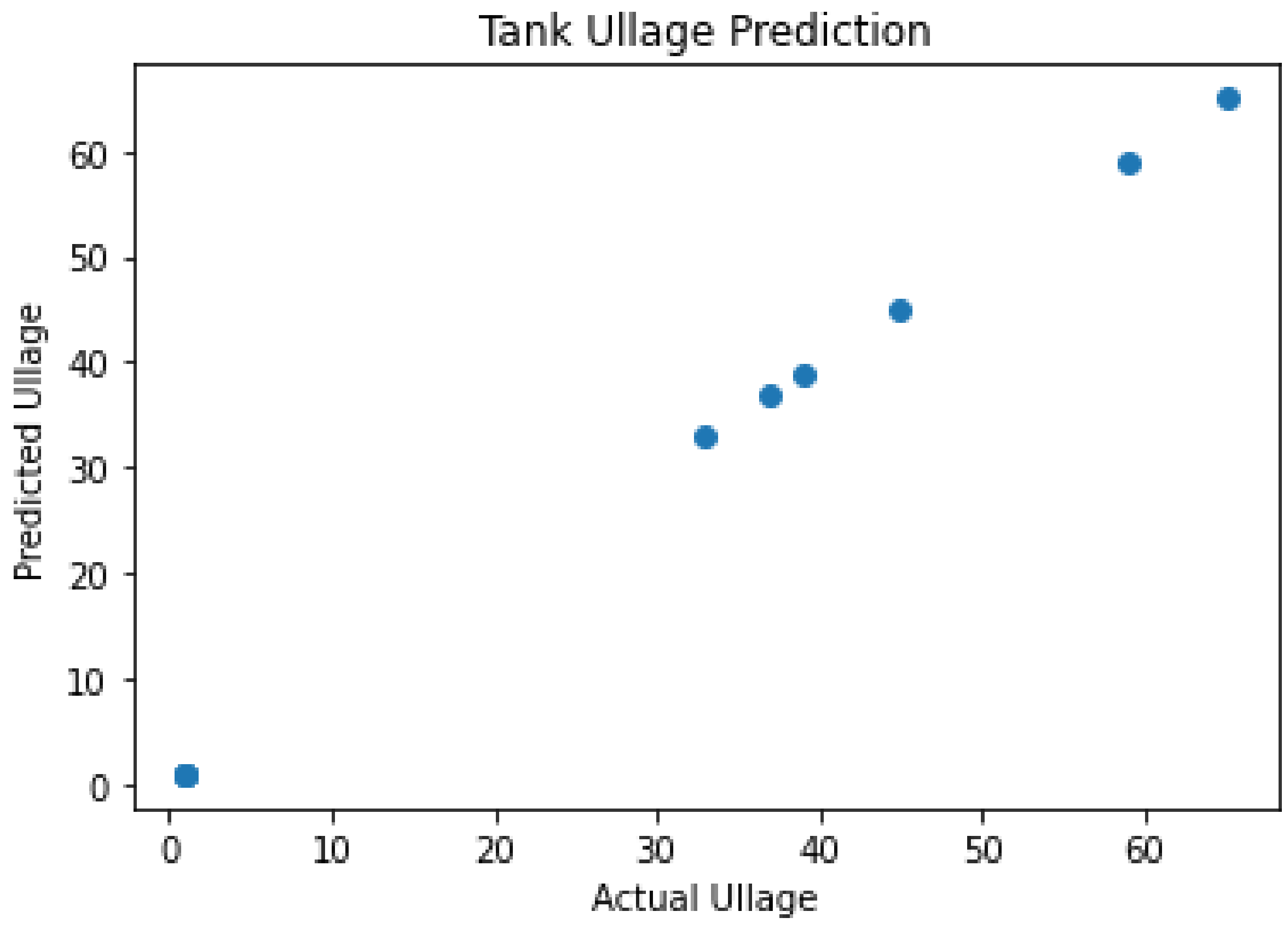
1.1 Tank ullage prediction model(code):

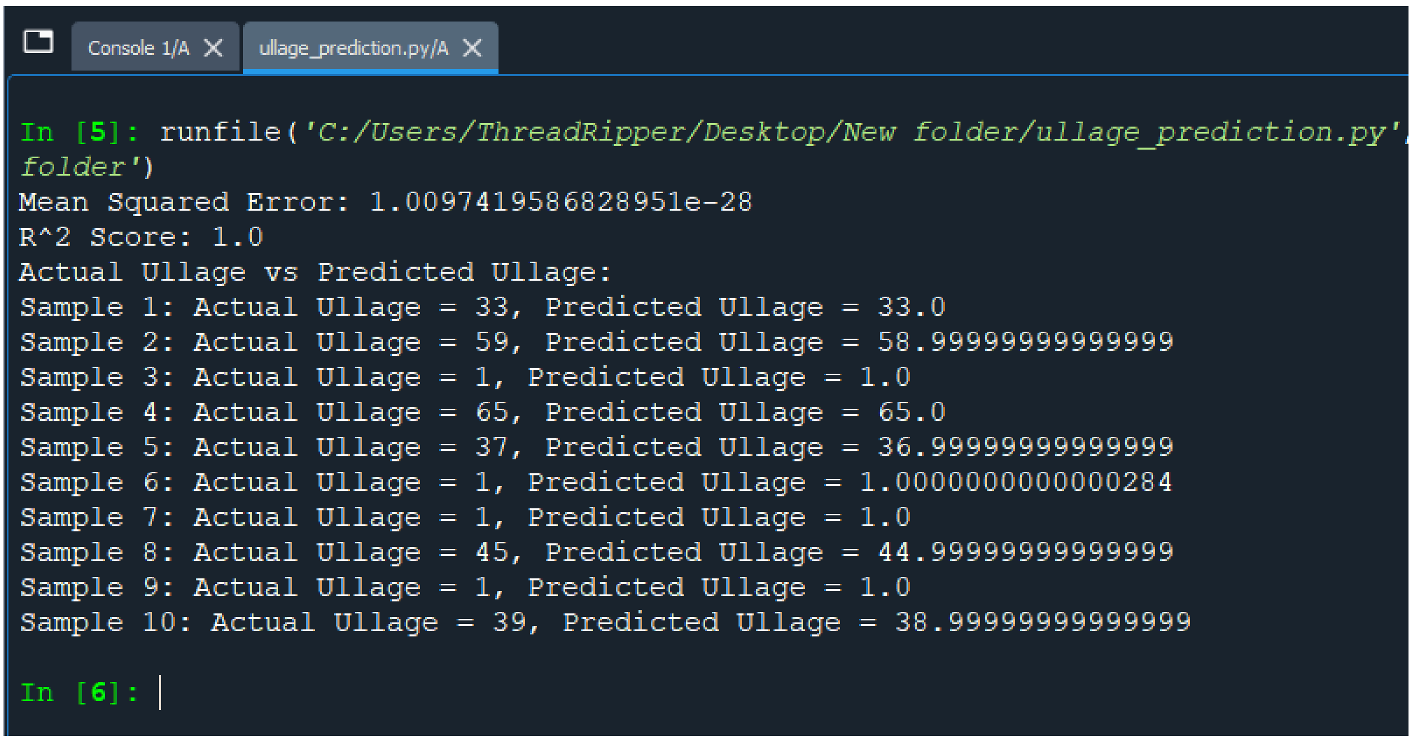
```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Read the dataset
df = pd.read_csv(r'D:\Projects\Tank ullage and optimised pertrol
and diesel refilling prediction model\ullage_prediction.csv')

# Select features and target variable
features = df[['temperature', 'humidity', 'pressure', 'fill_level']]
ullage = df['ullage']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, ullage,
test_size=0.2, random_state=42)

# Create a linear regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
predictions = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, predictions)
r2 = r2_score(y_test, predictions)
print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')

# Visualize predictions vs. actual values
plt.scatter(y_test, predictions)
plt.xlabel('Actual Ullage')
plt.ylabel('Predicted Ullage')
plt.title('Tank Ullage Prediction')
plt.show()

# Print actual and predicted ullage values for the first 10 samples
print("Actual Ullage vs Predicted Ullage:")
for i in range(10):
    print(f"Sample {i+1}: Actual Ullage = {y_test.iloc[i]}, Predicted
Ullage = {predictions[i]}")
```


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\ThreadRipper\Desktop\New folder
C:\Users\ThreadRipper\Desktop\New folder\ullage_prediction.py
optimal_refilling.py
optimal_refilling_dataset_creation.py
ullage_prediction.py*
ullage_sample_dataset_creation.py
LSP Python: ready
conda: base (Python 3.9.12)
Line 34, Col 1
ASCII
CRLF
RW
Mem 75%
ENG
02:42

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

df = pd.read_csv(r'D:\Projects\Tank ullage and optimised pertrol and diesel refilling prediction model\ullage_prediction.csv')

features = df[['temperature', 'humidity', 'pressure', 'fill_level']]
ullage = df['ullage']

X_train, X_test, y_train, y_test = train_test_split(features, ullage, test_size=0.2, random_state=42)

model = LinearRegression()

model.fit(X_train, y_train)

predictions = model.predict(X_test)

mse = mean_squared_error(y_test, predictions)
r2 = r2_score(y_test, predictions)
print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')

plt.scatter(y_test, predictions)
plt.xlabel('Actual Ullage')
plt.ylabel('Predicted Ullage')
plt.title('Tank Ullage Prediction')
plt.show()

print("Actual Ullage vs Predicted Ullage:")
for i in range(10):
    print(f"Sample {i+1}: Actual Ullage = {y_test.iloc[i]}, Predicted Ullage = {predictions[i]}")
```

1.2 Tank ullage prediction model(Output):



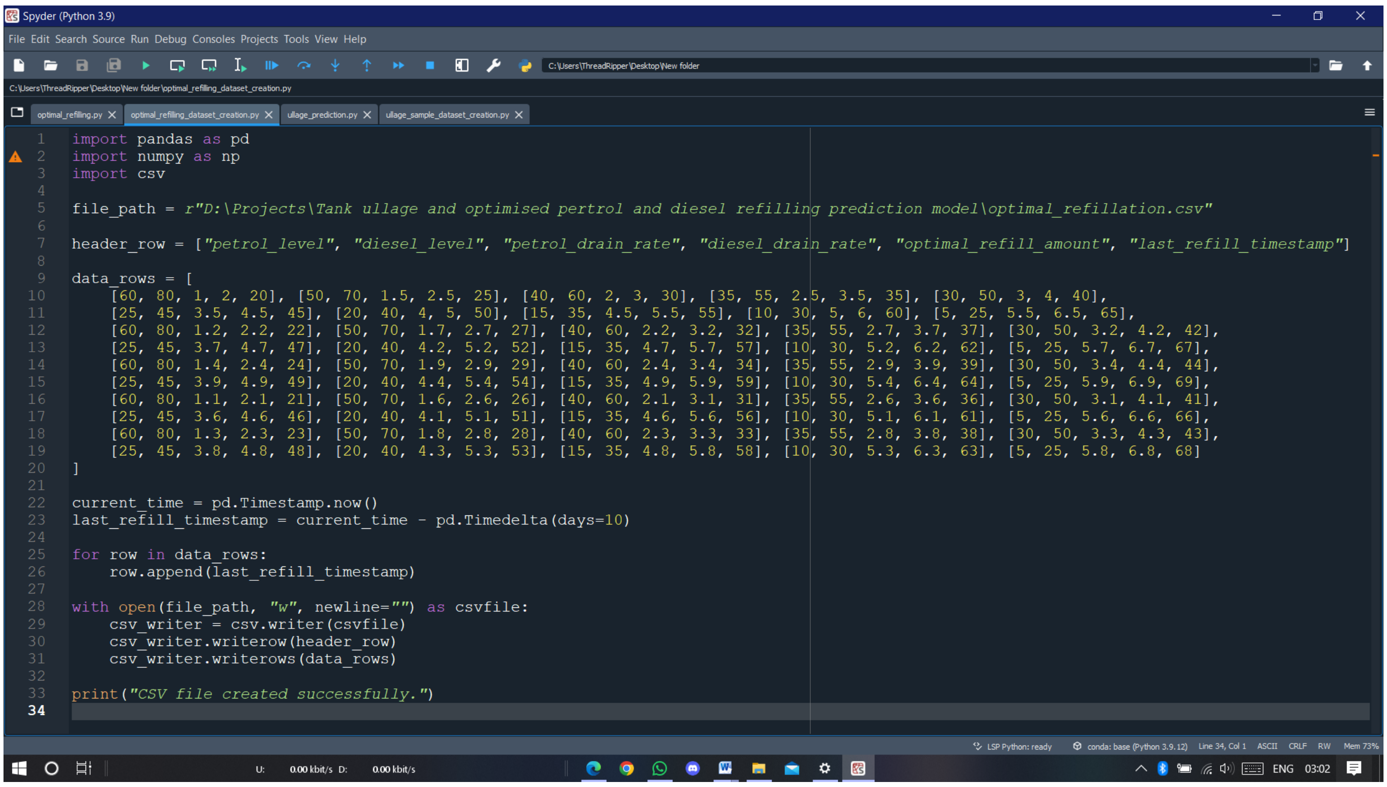


```
In [5]: runfile('C:/Users/ThreadRipper/Desktop/New folder/ullage_prediction.py',
folder')
Mean Squared Error: 1.0097419586828951e-28
R^2 Score: 1.0
Actual Ullage vs Predicted Ullage:
Sample 1: Actual Ullage = 33, Predicted Ullage = 33.0
Sample 2: Actual Ullage = 59, Predicted Ullage = 58.99999999999999
Sample 3: Actual Ullage = 1, Predicted Ullage = 1.0
Sample 4: Actual Ullage = 65, Predicted Ullage = 65.0
Sample 5: Actual Ullage = 37, Predicted Ullage = 36.99999999999999
Sample 6: Actual Ullage = 1, Predicted Ullage = 1.0000000000000284
Sample 7: Actual Ullage = 1, Predicted Ullage = 1.0
Sample 8: Actual Ullage = 45, Predicted Ullage = 44.99999999999999
Sample 9: Actual Ullage = 1, Predicted Ullage = 1.0
Sample 10: Actual Ullage = 39, Predicted Ullage = 38.99999999999999

In [6]:
```

1.3 Creating the sample CSV file for optimal tank filling prediction

```
import pandas as pd
import numpy as np
import csv

file_path = r"D:\Projects\Tank ullage and optimised pertrol and diesel refilling prediction model\optimal_refillation.csv"

header_row = ["petrol_level", "diesel_level", "petrol_drain_rate", "diesel_drain_rate", "optimal_refill_amount", "last_refill_timestamp"]

data_rows = [
   [60, 80, 1, 2, 20], [50, 70, 1.5, 2.5, 25], [40, 60, 2, 3, 30], [35, 55, 2.5, 3.5, 35], [30, 50, 3, 4, 40],
   [25, 45, 3.5, 4.5, 45], [20, 40, 4, 5, 50], [15, 35, 4.5, 5.5, 55], [10, 30, 5, 6, 60], [5, 25, 5.5, 6.5, 65],
   [60, 80, 1.2, 2.2, 22], [50, 70, 1.7, 2.7, 27], [40, 60, 2.2, 3.2, 32], [35, 55, 2.7, 3.7, 37], [30, 50, 3.2, 4.2, 42],
   [25, 45, 3.7, 4.7, 47], [20, 40, 4.2, 5.2, 52], [15, 35, 4.7, 5.7, 57], [10, 30, 5.2, 6.2, 62], [5, 25, 5.7, 6.7, 67],
   [60, 80, 1.4, 2.4, 24], [50, 70, 1.9, 2.9, 29], [40, 60, 2.4, 3.4, 34], [35, 55, 2.9, 3.9, 39], [30, 50, 3.4, 4.4, 44],
   [25, 45, 3.9, 4.9, 49], [20, 40, 4.4, 5.4, 54], [15, 35, 4.9, 5.9, 59], [10, 30, 5.4, 6.4, 64], [5, 25, 5.9, 6.9, 69],
   [60, 80, 1.1, 2.1, 21], [50, 70, 1.6, 2.6, 26], [40, 60, 2.1, 3.1, 31], [35, 55, 2.6, 3.6, 36], [30, 50, 3.1, 4.1, 41],
   [25, 45, 3.6, 4.6, 46], [20, 40, 4.1, 5.1, 51], [15, 35, 4.6, 5.6, 56], [10, 30, 5.1, 6.1, 61], [5, 25, 5.6, 6.6, 66],
   [60, 80, 1.3, 2.3, 23], [50, 70, 1.8, 2.8, 28], [40, 60, 2.3, 3.3, 33], [35, 55, 2.8, 3.8, 38], [30, 50, 3.3, 4.3, 43],
   [25, 45, 3.8, 4.8, 48], [20, 40, 4.3, 5.3, 53], [15, 35, 4.8, 5.8, 58], [10, 30, 5.3, 6.3, 63], [5, 25, 5.8, 6.8, 68]
]

current_time = pd.Timestamp.now()
last_refill_timestamp = current_time - pd.Timedelta(days=10)
```

```
for row in data_rows:
    row.append(last_refill_timestamp)

with open(file_path, "w", newline="") as csvfile:
    csv_writer = csv.writer(csvfile)
    csv_writer.writerow(header_row)
    csv_writer.writerows(data_rows)

print("CSV file created successfully.")
```



```
import pandas as pd
import numpy as np
import csv

file_path = r"D:\Projects\Tank ullage and optimised pertrol and diesel refilling prediction model\optimal_refillation.csv"

header_row = ["petrol_level", "diesel_level", "petrol_drain_rate", "diesel_drain_rate", "optimal_refill_amount", "last_refill_timestamp"]

data_rows = [
    [60, 80, 1, 2, 20], [50, 70, 1.5, 2.5, 25], [40, 60, 2, 3, 30], [35, 55, 2.5, 3.5, 35], [30, 50, 3, 4, 40],
    [25, 45, 3.5, 4.5, 45], [20, 40, 4, 5, 50], [15, 35, 4.5, 5.5, 55], [10, 30, 5, 6, 60], [5, 25, 5.5, 6.5, 65],
    [60, 80, 1.2, 2.2, 22], [50, 70, 1.7, 2.7, 27], [40, 60, 2.2, 3.2, 32], [35, 55, 2.7, 3.7, 37], [30, 50, 3.2, 4.2, 42],
    [25, 45, 3.7, 4.7, 47], [20, 40, 4.2, 5.2, 52], [15, 35, 4.7, 5.7, 57], [10, 30, 5.2, 6.2, 62], [5, 25, 5.7, 6.7, 67],
    [60, 80, 1.4, 2.4, 24], [50, 70, 1.9, 2.9, 29], [40, 60, 2.4, 3.4, 34], [35, 55, 2.9, 3.9, 39], [30, 50, 3.4, 4.4, 44],
    [25, 45, 3.9, 4.9, 49], [20, 40, 4.4, 5.4, 54], [15, 35, 4.9, 5.9, 59], [10, 30, 5.4, 6.4, 64], [5, 25, 5.9, 6.9, 69],
    [60, 80, 1.1, 2.1, 21], [50, 70, 1.6, 2.6, 26], [40, 60, 2.1, 3.1, 31], [35, 55, 2.6, 3.6, 36], [30, 50, 3.1, 4.1, 41],
    [25, 45, 3.6, 4.6, 46], [20, 40, 4.1, 5.1, 51], [15, 35, 4.6, 5.6, 56], [10, 30, 5.1, 6.1, 61], [5, 25, 5.6, 6.6, 66],
    [60, 80, 1.3, 2.3, 23], [50, 70, 1.8, 2.8, 28], [40, 60, 2.3, 3.3, 33], [35, 55, 2.8, 3.8, 38], [30, 50, 3.3, 4.3, 43],
    [25, 45, 3.8, 4.8, 48], [20, 40, 4.3, 5.3, 53], [15, 35, 4.8, 5.8, 58], [10, 30, 5.3, 6.3, 63], [5, 25, 5.8, 6.8, 68]
]

current_time = pd.Timestamp.now()
last_refill_timestamp = current_time - pd.Timedelta(days=10)

for row in data_rows:
    row.append(last_refill_timestamp)

with open(file_path, "w", newline="") as csvfile:
    csv_writer = csv.writer(csvfile)
    csv_writer.writerow(header_row)
    csv_writer.writerows(data_rows)

print("CSV file created successfully.")
```

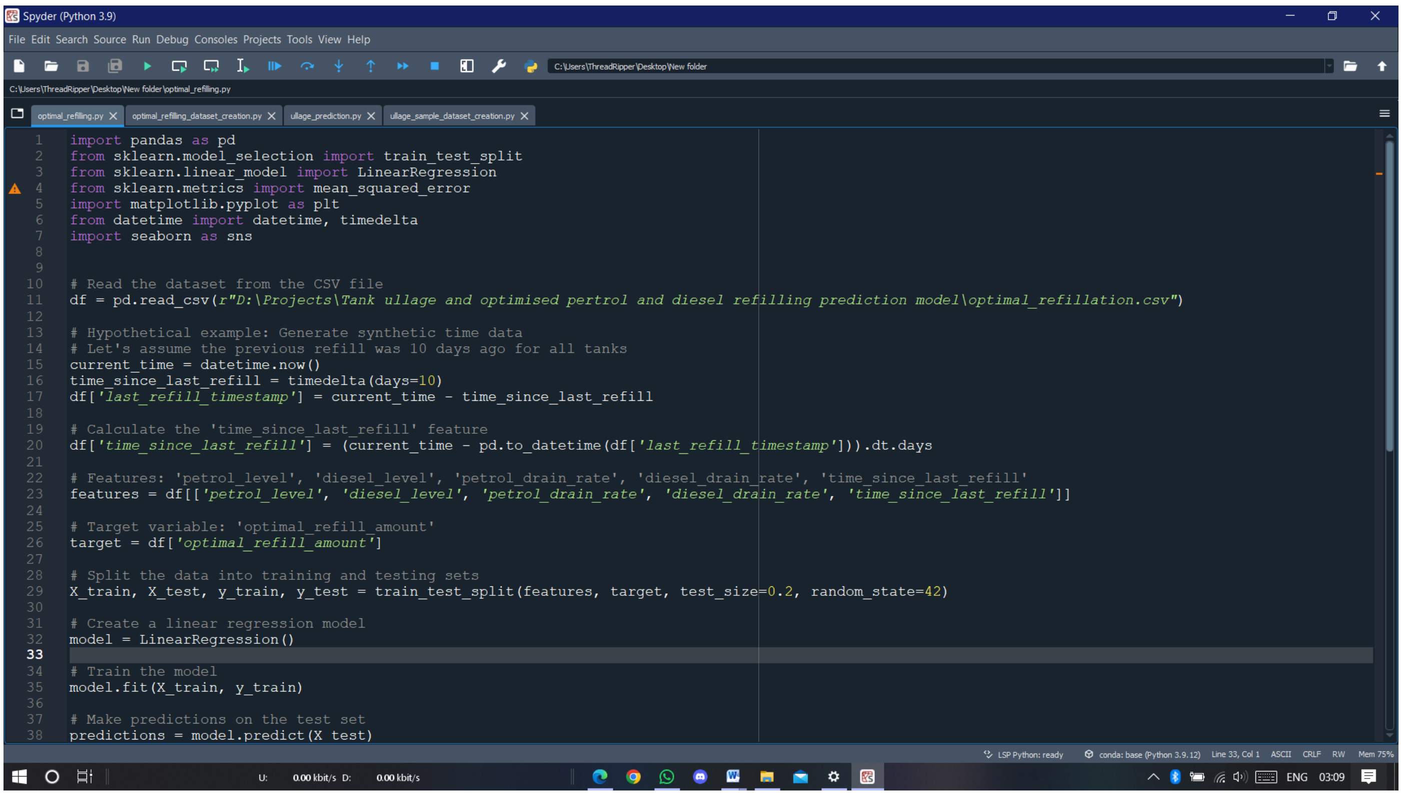
Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\ThreadRipper\Desktop\New folder
C:\Users\ThreadRipper\Desktop\New folder\optimal_refilling_dataset_creation.py
optimal_refilling.py
optimal_refilling_dataset_creation.py
ullage_prediction.py
ullage_sample_dataset_creation.py
LSP Python: ready
conda: base (Python 3.9.12)
Line 34, Col 1
ASCII
CRLF
RW
Mem 73%
ENG
03:02

1.4 Tank optimal refilling prediction model(code):

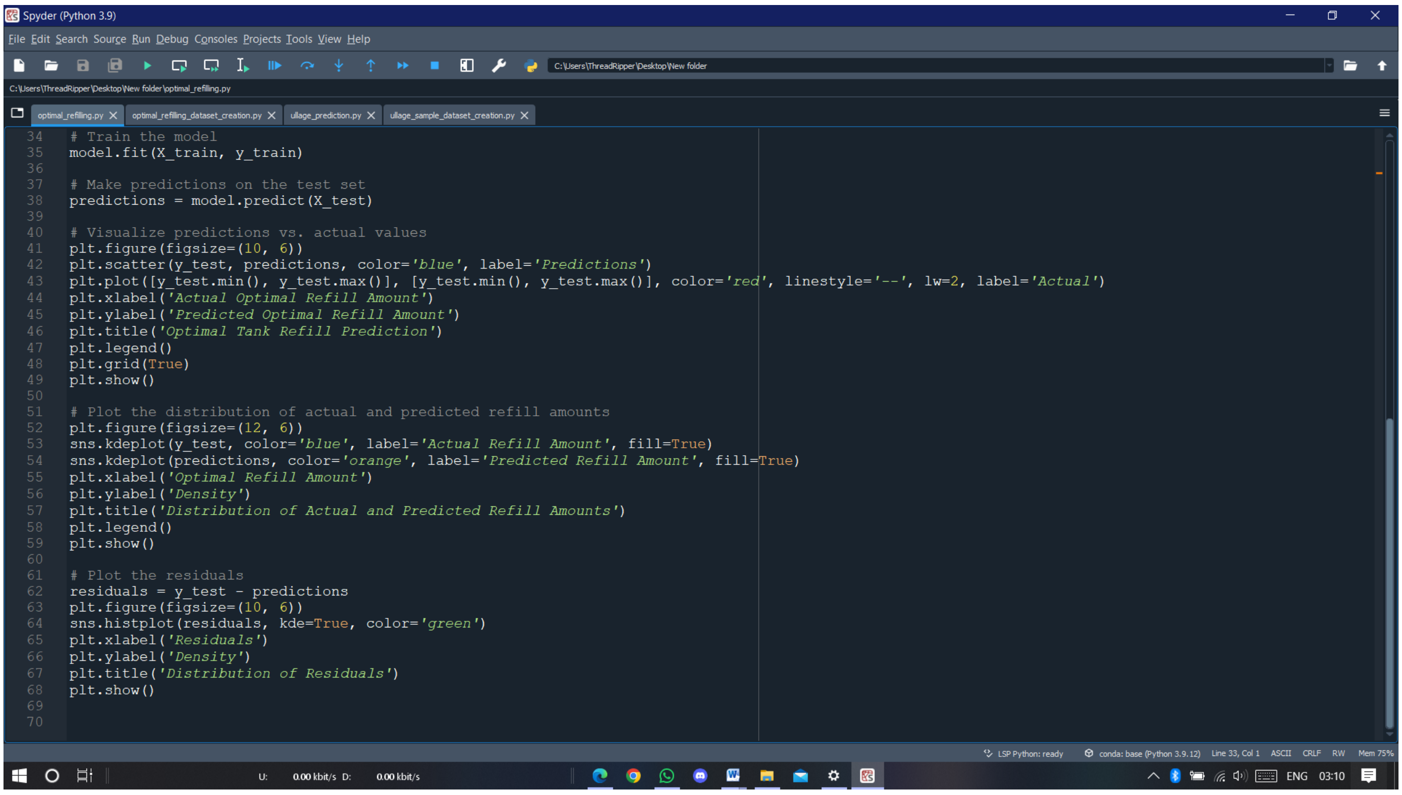
```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
import seaborn as sns


# Read the dataset from the CSV file
df = pd.read_csv(r"D:\Projects\Tank ullage and optimised
pertrol and diesel refilling prediction
model\optimal_refillation.csv")

# Hypothetical example: Generate synthetic time data
# Let's assume the previous refill was 10 days ago for all tanks
current_time = datetime.now()
time_since_last_refill = timedelta(days=10)
df['last_refill_timestamp'] = current_time -
time_since_last_refill

# Calculate the 'time_since_last_refill' feature
df['time_since_last_refill'] = (current_time -
pd.to_datetime(df['last_refill_timestamp'])).dt.days

# Features: 'petrol_level', 'diesel_level', 'petrol_drain_rate',
'diesel_drain_rate', 'time_since_last_refill'
features = df[['petrol_level', 'diesel_level', 'petrol_drain_rate',
'diesel_drain_rate', 'time_since_last_refill']]

# Target variable: 'optimal_refill_amount'
target = df['optimal_refill_amount']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features,
target, test_size=0.2, random_state=42)
```

```
# Create a linear regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
predictions = model.predict(X_test)

# Visualize predictions vs. actual values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, predictions, color='blue', label='Predictions')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()],
color='red', linestyle='--', lw=2, label='Actual')
plt.xlabel('Actual Optimal Refill Amount')
plt.ylabel('Predicted Optimal Refill Amount')
plt.title('Optimal Tank Refill Prediction')
plt.legend()
plt.grid(True)
plt.show()

# Plot the distribution of actual and predicted refill amounts
plt.figure(figsize=(12, 6))
sns.kdeplot(y_test, color='blue', label='Actual Refill Amount',
fill=True)
sns.kdeplot(predictions, color='orange', label='Predicted Refill
Amount', fill=True)
plt.xlabel('Optimal Refill Amount')
plt.ylabel('Density')
plt.title('Distribution of Actual and Predicted Refill Amounts')
plt.legend()
plt.show()
# Plot the residuals
residuals = y_test - predictions
plt.figure(figsize=(10, 6))
sns.histplot(residuals, kde=True, color='green')
plt.xlabel('Residuals')
plt.ylabel('Density')
```

plt.title('Distribution of Residuals')
plt.show()

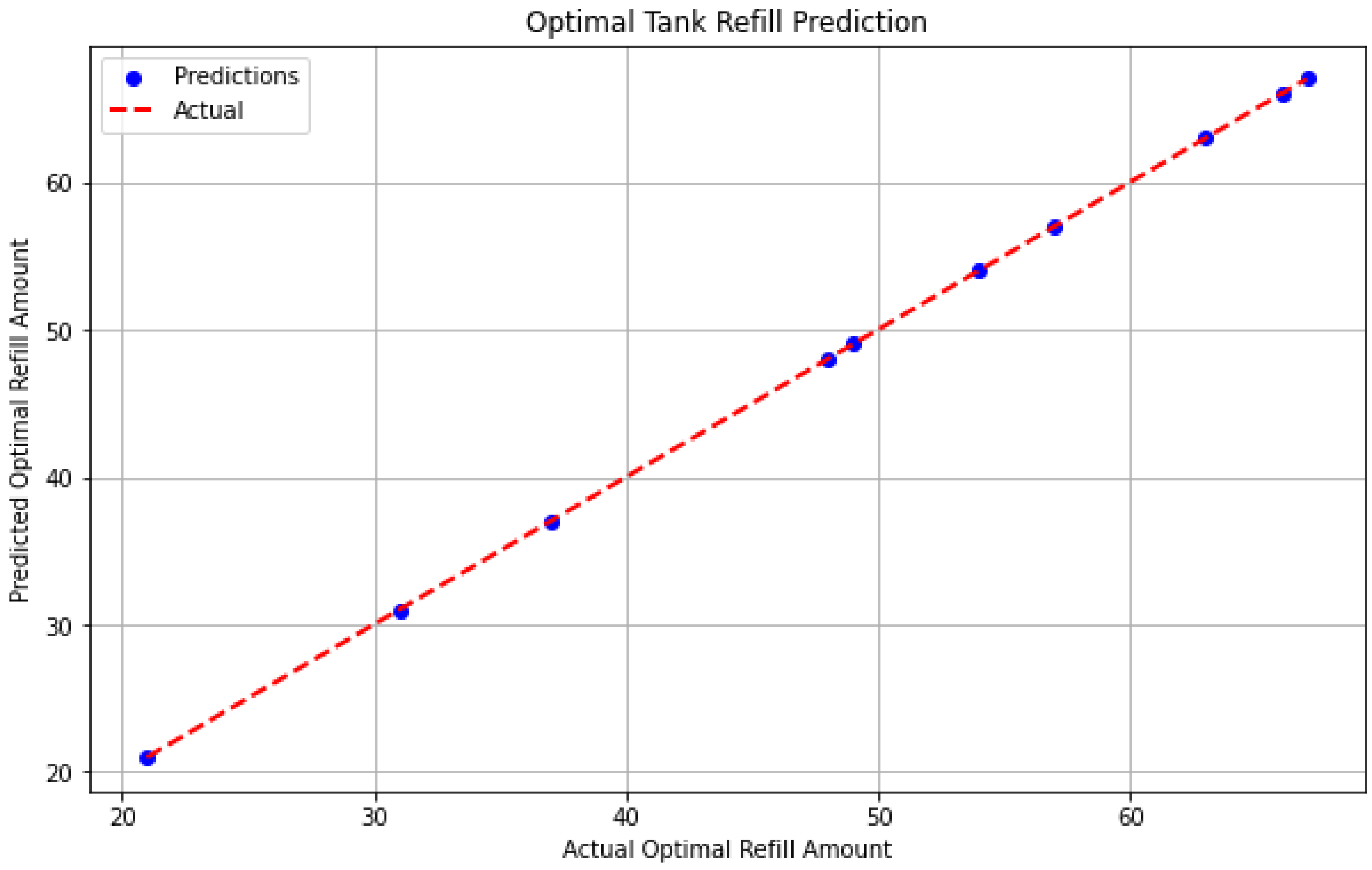


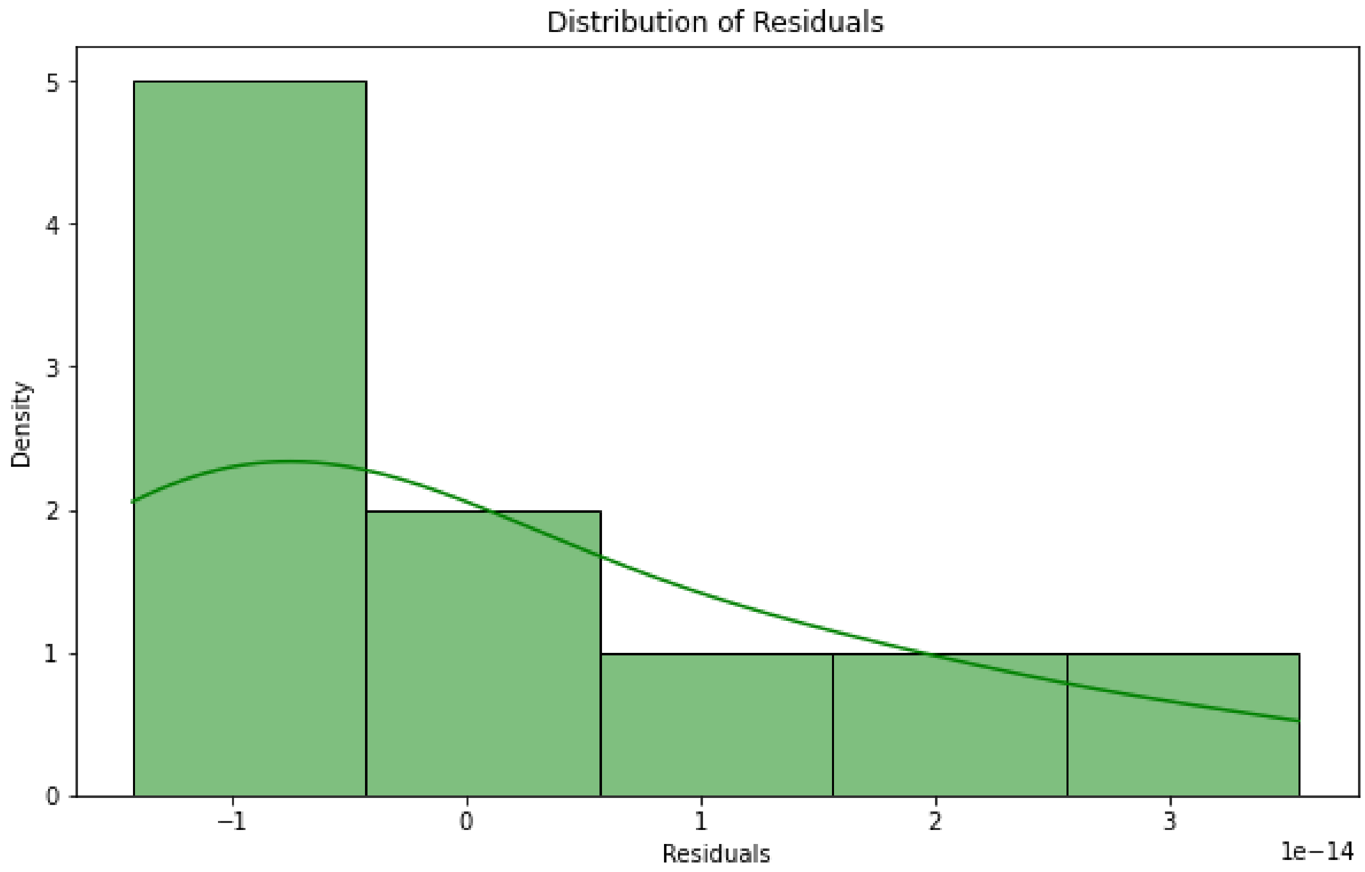
```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
import seaborn as sns


# Read the dataset from the CSV file
df = pd.read_csv(r"D:\Projects\Tank ullage and optimised pertrol and diesel refilling prediction model\optimal_refillation.csv")

# Hypothetical example: Generate synthetic time data
# Let's assume the previous refill was 10 days ago for all tanks
current_time = datetime.now()
time_since_last_refill = timedelta(days=10)
df['last_refill_timestamp'] = current_time - time_since_last_refill

# Calculate the 'time_since_last_refill' feature
df['time_since_last_refill'] = (current_time - pd.to_datetime(df['last_refill_timestamp'])).dt.days

# Features: 'petrol_level', 'diesel_level', 'petrol_drain_rate', 'diesel_drain_rate', 'time_since_last_refill'
features = df[['petrol_level', 'diesel_level', 'petrol_drain_rate', 'diesel_drain_rate', 'time_since_last_refill']]

# Target variable: 'optimal_refill_amount'
target = df['optimal_refill_amount']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)

# Create a linear regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
predictions = model.predict(X_test)
```

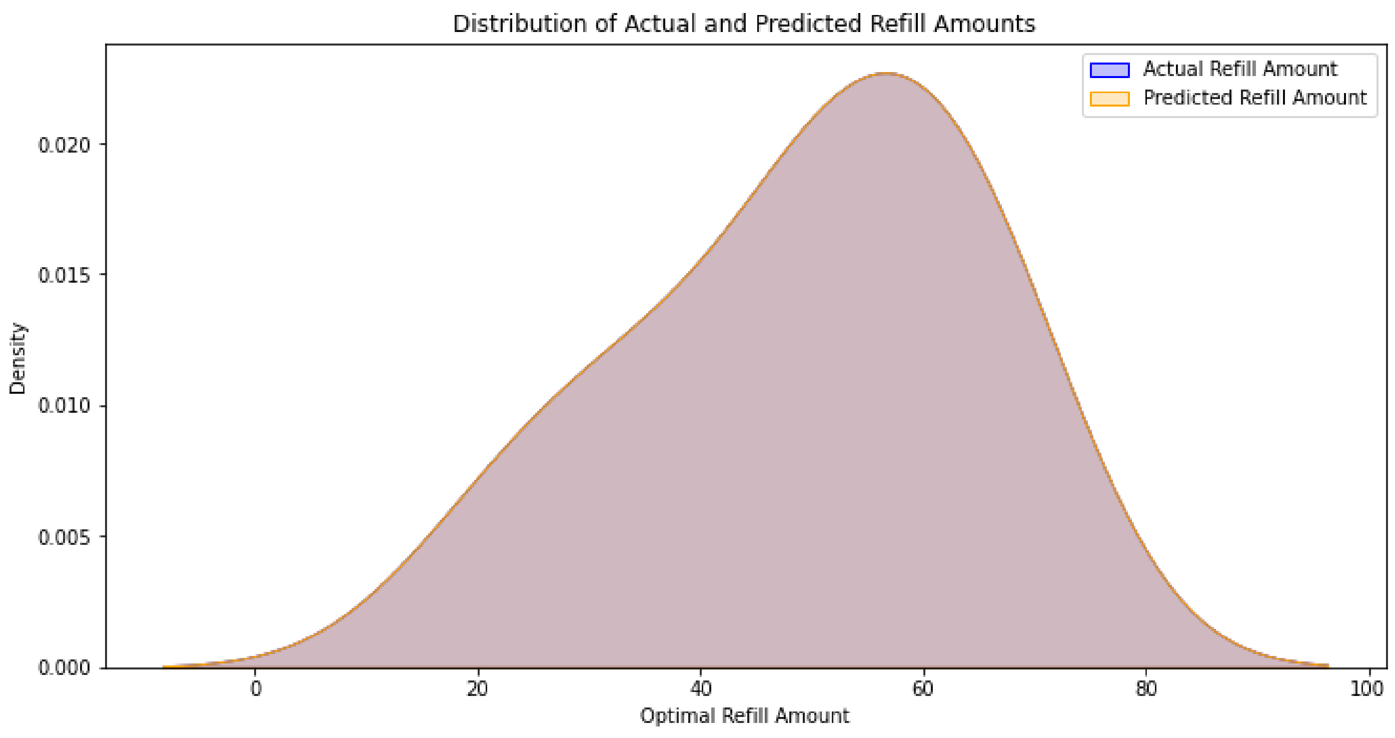


```python
# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
predictions = model.predict(X_test)

# Visualize predictions vs. actual values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, predictions, color='blue', label='Predictions')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color='red', linestyle='--', lw=2, label='Actual')
plt.xlabel('Actual Optimal Refill Amount')
plt.ylabel('Predicted Optimal Refill Amount')
plt.title('Optimal Tank Refill Prediction')
plt.legend()
plt.grid(True)
plt.show()

# Plot the distribution of actual and predicted refill amounts
plt.figure(figsize=(12, 6))
sns.kdeplot(y_test, color='blue', label='Actual Refill Amount', fill=True)
sns.kdeplot(predictions, color='orange', label='Predicted Refill Amount', fill=True)
plt.xlabel('Optimal Refill Amount')
plt.ylabel('Density')
plt.title('Distribution of Actual and Predicted Refill Amounts')
plt.legend()
plt.show()

# Plot the residuals
residuals = y_test - predictions
plt.figure(figsize=(10, 6))
sns.histplot(residuals, kde=True, color='green')
plt.xlabel('Residuals')
plt.ylabel('Density')
plt.title('Distribution of Residuals')
plt.show()
```

1.5 Tank optimal refilling prediction model(output):







NOTE:

- I've added a new feature called 'time_since_last_refill', which represents the duration of time since the last refill for each tank.
- This feature can be calculated based on the timestamps of previous refills and the current time.
- The model now takes into account this additional time-related feature when making predictions.
- The visualization remains the same, comparing the predicted optimal refill amounts to the actual values.

To make this code more realistic, you would need to incorporate actual time data and calculate 'time_since_last_refill' accordingly. Additionally, you may need to adjust the model and features based on the specifics of your dataset and scenario.

CHALLENGES AND SOLUTIONS

Challenges:

1. **Data Quality:** The quality of input data can significantly impact the performance of the model. Incomplete, noisy, or inaccurate data may lead to biased predictions and reduced model accuracy.
2. **Feature Selection:** Identifying the most relevant features that contribute to tank ullage prediction can be challenging. Selecting irrelevant features or excluding important ones may result in suboptimal model performance.
3. **Model Complexity:** Choosing an appropriate model complexity is crucial. Overly complex models may lead to overfitting, where the model learns noise in the training data and fails to generalize well to unseen data. On the other hand, overly simple models may underfit and fail to capture the underlying patterns in the data.
4. **Model Interpretability:** Linear regression models offer simplicity and interpretability, but more complex models may provide better predictive performance. Balancing between model interpretability and predictive accuracy is essential, especially in domains where interpretability is crucial.
5. **Deployment and Integration:** Integrating the predictive model into existing systems or processes can be challenging. Considerations such as real-time inference, scalability, and maintenance need to be addressed during deployment.

Solutions:

1. **Data Preprocessing:** Perform thorough data preprocessing steps such as handling missing values, outlier detection, normalization, and feature scaling to ensure data quality.
2. **Feature Engineering:** Conduct extensive feature engineering to extract relevant information from the available data. Techniques such as feature transformation, dimensionality reduction, and feature selection can help improve model performance.
3. **Model Selection and Evaluation:** Experiment with different machine learning algorithms and model architectures to find the best-performing model. Utilize techniques such as cross-validation and hyperparameter tuning to optimize model performance while avoiding overfitting.
4. **Model Explainability:** Use techniques such as feature importance analysis, partial dependence plots, and SHAP (SHapley Additive exPlanations) values to interpret and explain the model's predictions, even for more complex models.
5. **Deployment Strategies:** Consider deploying the model in a scalable and efficient manner, such as containerization using Docker or deploying as a web service using platforms like Flask or FastAPI. Ensure robust monitoring and maintenance procedures to keep the model performing well over time. Additionally, provide adequate documentation and support for integration into existing systems.

CONCLUSION

In conclusion, this project tackles the challenge of predicting tank ullage and optimizing refill operations for two tanks, one containing petrol and the other diesel. By employing machine learning techniques and data analysis, the model aims to provide accurate predictions while considering factors such as temperature, humidity, pressure, and fill level. Challenges such as data quality, feature selection, and model interpretability were addressed through rigorous preprocessing, feature engineering, and model evaluation. The project offers valuable insights and solutions for efficiently managing tank refilling operations in various industrial settings, contributing to enhanced decision-making and resource optimization.

FUTURE SCOPE

The journey of the this project doesn't end here. Future iterations hold the promise of even greater capabilities, including:

1. **Integration of Predictive Maintenance:** One significant future scope for this project is integrating predictive maintenance capabilities. By analysing historical data and sensor readings from the tanks, the system can predict potential equipment failures or maintenance needs. This proactive approach can help prevent costly downtime, improve equipment reliability, and ensure continuous operation of the tanks.
2. **Real-Time Optimization and Control:** Another crucial future scope is implementing real-time optimization and control mechanisms. By continuously monitoring factors such as fuel levels, temperature, and pressure in real-time, the system can dynamically adjust refill schedules and optimize tank operations. This real-time optimization can enhance efficiency, respond quickly to changing conditions, and adapt to fluctuations in demand, ultimately improving overall performance and resource utilization.

ACKNOWLEDGEMENT

The successful development and implementation of the tank ullage prediction and optimization project owe much to the collaborative efforts and contributions of various individuals and groups:

- Development Team: The dedication, expertise, and hard work of the development team were instrumental in designing, implementing, and refining the predictive models and optimization algorithms.

- Stakeholders: The invaluable input and guidance provided by stakeholders throughout the project lifecycle helped ensure that the solution met the specific needs and requirements of the users and stakeholders.

- User Community: The on-going support, feedback, and engagement from the user community played a vital role in shaping the project's direction and enhancing its usability and effectiveness.

Together, this collaborative effort has transformed the tank ullage prediction and optimization project into a pioneering solution, providing valuable insights and operational efficiencies in the domain of tank management and logistics.