NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

Q1. Download the Oil Spill Dataset and perform Data cleaning and Data Pre-Processing if Necessary.

Answer:

```python
import pandas as pd
df = pd.read_csv('oil_spill.csv')

# Display the first few rows of the dataset
print("Initial dataset:")
print(df.head())
```

```
PS C:\Users\vaibh\Downloads> & C:/Users/vaibh/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/vaibh/Downloads/Oil Spill"
Initial dataset:
   f_1   f_2     f_3    f_4  f_5      f_6   f_7   f_8  ...    f_43    f_44   f_45  f_46     f_47   f_48  f_49  target
0    1  2558 1506.09 456.63   90  6395000 40.88  7.89  ...  763.16  135.46   3.73     0 33243.19  65.74  7.95       1
1    2 22325   79.11 841.03  180 55812500 51.11  1.21  ... 9593.48 1648.80   0.60     0 51572.04  65.73  6.26       0
2    3   115 1449.85 608.43   88   287500 40.42  7.34  ...  150.00   45.13   9.33     1 31692.84  65.81  7.84       1
3    4  1201 1562.53 295.65   66  3002500 42.40  7.97  ...  453.21  144.97  13.33     1 37696.21  65.67  8.07       1
4    5   312  950.27 440.86   37   780000 41.43  7.03  ...  512.54  109.16   2.58     0 29038.17  65.66  7.35       0

[5 rows x 50 columns]
PS C:\Users\vaibh\Downloads>
```

```python
import pandas as pd
df = pd.read_csv('oil_spill.csv')

# Data Cleaning

# Check for missing values
print("\nMissing values:")
print(df.isnull().sum())

# Fill missing numerical values with mean
df.fillna(df.mean(), inplace=True)
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

```
PS C:\Users\vaibh\Downloads> & C:/Users/vaibh/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/vaibh/Downloads/Oil Spill"

Missing values:
f_1       0
f_2       0
f_3       0
f_4       0
f_5       0
f_6       0
f_7       0
f_8       0
f_9       0
f_10      0
f_11      0
f_12      0
f_13      0
f_14      0
f_15      0
f_16      0
f_17      0
f_18      0
f_19      0
f_20      0
f_21      0
f_22      0
f_23      0
f_24      0
f_25      0
f_26      0
f_27      0
f_28      0
f_29      0
f_30      0
f_31      0
f_32      0
f_33      0
f_34      0
f_35      0
f_36      0
f_37      0
f_38      0
f_39      0
f_40      0
f_41      0
f_42      0
f_43      0
f_44      0
f_45      0
f_46      0
f_47      0
f_48      0
f_49      0
target    0
dtype: int64
PS C:\Users\vaibh\Downloads>
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

Q2. Use various methods such as Handling null values, One-Hot Encoding, Imputation, and Scaling of Data Pre-Processing where necessary.

Answer:

a. Handling null values:

```python
import pandas as pd

# Load the dataset
df = pd.read_csv('oil_spill.csv')

# Drop rows with null values
df.dropna(inplace=True)

# Display the dataset after handling null values
print(df.head())
```

```
2   3   115  1449.85  608.43   88   287500  40.42  7.34  ...   150.00   45.13   9.33    1  31692.84  65.81  7.84      1
3   4  1201  1562.53  295.65   66  3002500  42.40  7.97  ...   453.21  144.97  13.33    1  37696.21  65.67  8.07      1
4   5   312   950.27  440.86   37   780000  41.43  7.03  ...   512.54  109.16   2.58    0  29038.17  65.66  7.35      0

[5 rows x 50 columns]
PS C:\Users\vaibh\Downloads>
```

b. One-hot Encoding

```python
import pandas as pd
from sklearn.preprocessing import OneHotEncoder

# Load the dataset
df = pd.read_csv('oil_spill.csv')

# Extract categorical columns
categorical_columns = df.select_dtypes(include=['object', 'category']).columns.tolist()

# Initialize OneHotEncoder
encoder = OneHotEncoder()

# Fit and transform the categorical columns
encoded_features = encoder.fit_transform(df[categorical_columns])

# Create a DataFrame for the encoded features
df_encoded = pd.DataFrame(encoded_features.toarray(), columns=encoder.get_feature_names_out(categorical_columns))

# Concatenate the encoded features with the original DataFrame
df = pd.concat([df.drop(columns=categorical_columns), df_encoded], axis=1)

print(df.head())
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

```
PS C:\Users\vaibh\Downloads> & C:/Users/vaibh/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/vaibh/Downloads/Oil Spill"
   f_1    f_2      f_3     f_4  f_5      f_6    f_7   f_8  ...     f_43     f_44   f_45  f_46      f_47   f_48  f_49  target
0    1   2558  1506.09  456.63   90   6395000  40.88  7.89  ...   763.16   135.46   3.73     0  33243.19  65.74  7.95       1
1    2  22325    79.11  841.03  180  55812500  51.11  1.21  ...  9593.48  1648.80   0.60     0  51572.04  65.73  6.26       0
2    3    115  1449.85  608.43   88    287500  40.42  7.34  ...   150.00    45.13   9.33     1  31692.84  65.81  7.84       1
3    4   1201  1562.53  295.65   66   3002500  42.40  7.97  ...   453.21   144.97  13.33     1  37696.21  65.67  8.07       1
4    5    312   950.27  440.86   37    780000  41.43  7.03  ...   512.54   109.16   2.58     0  29038.17  65.66  7.35       0

[5 rows x 50 columns]
PS C:\Users\vaibh\Downloads>
```

c. Imputation

```python
import pandas as pd
from sklearn.impute import SimpleImputer

# Load the dataset
df = pd.read_csv('oil_spill.csv')

# Display the first few rows of the dataset
print("Initial dataset:")
print(df.head())

# Imputation

# Extract numerical columns
numerical_columns = df.select_dtypes(include=['int64', 'float64']).columns.tolist()

# Initialize SimpleImputer
imputer = SimpleImputer(strategy='mean')

# Fit and transform the numerical columns
df[numerical_columns] = imputer.fit_transform(df[numerical_columns])

print("\nDataset after imputation:")
print(df.head())
```

```
PS C:\Users\vaibh\Downloads> & C:/Users/vaibh/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/vaibh/Downloads/Oil Spill"
Initial dataset:
   f_1    f_2      f_3     f_4  f_5      f_6    f_7   f_8  ...     f_43     f_44   f_45  f_46      f_47   f_48  f_49  target
0    1   2558  1506.09  456.63   90   6395000  40.88  7.89  ...   763.16   135.46   3.73     0  33243.19  65.74  7.95       1
1    2  22325    79.11  841.03  180  55812500  51.11  1.21  ...  9593.48  1648.80   0.60     0  51572.04  65.73  6.26       0
2    3    115  1449.85  608.43   88    287500  40.42  7.34  ...   150.00    45.13   9.33     1  31692.84  65.81  7.84       1
3    4   1201  1562.53  295.65   66   3002500  42.40  7.97  ...   453.21   144.97  13.33     1  37696.21  65.67  8.07       1
4    5    312   950.27  440.86   37    780000  41.43  7.03  ...   512.54   109.16   2.58     0  29038.17  65.66  7.35       0

[5 rows x 50 columns]

Dataset after imputation:
   f_1     f_2      f_3     f_4   f_5        f_6    f_7   f_8  ...     f_43     f_44   f_45  f_46      f_47   f_48  f_49  target
0  1.0  2558.0  1506.09  456.63  90.0  6395000.0  40.88  7.89  ...   763.16   135.46   3.73   0.0  33243.19  65.74  7.95     1.0
1  2.0 22325.0    79.11  841.03 180.0 55812500.0  51.11  1.21  ...  9593.48  1648.80   0.60   0.0  51572.04  65.73  6.26     0.0
2  3.0   115.0  1449.85  608.43  88.0   287500.0  40.42  7.34  ...   150.00    45.13   9.33   1.0  31692.84  65.81  7.84     1.0
3  4.0  1201.0  1562.53  295.65  66.0  3002500.0  42.40  7.97  ...   453.21   144.97  13.33   1.0  37696.21  65.67  8.07     1.0
4  5.0   312.0   950.27  440.86  37.0   780000.0  41.43  7.03  ...   512.54   109.16   2.58   0.0  29038.17  65.66  7.35     0.0

[5 rows x 50 columns]
PS C:\Users\vaibh\Downloads>
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

---

d. Scaling of Data Pre-Processing

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler

df = pd.read_csv('oil_spill.csv')

# Extract numerical columns
numerical_columns = df.select_dtypes(include=['int64', 'float64']).columns.tolist()

# Initialize StandardScaler
scaler = StandardScaler()

# Fit and transform the numerical columns
df[numerical_columns] = scaler.fit_transform(df[numerical_columns])

# dataset after scaling
print("\nDataset after scaling:")
print(df.head())
```

```
Dataset after scaling:
       f_1       f_2       f_3       f_4       f_5       f_6  ...      f_45      f_46      f_47      f_48      f_49    target
0 -1.240922  1.152390  1.346434 -0.793007  0.129657  1.469091  ... -0.255448 -0.383248  3.686767  0.388730 -0.058377  4.674790
1 -1.225524 11.389546 -1.033273 -0.057342  2.114766 14.374844  ... -0.878152 -0.383248  6.362181  0.387769 -0.639664 -0.213913
2 -1.210126 -0.112818  1.252645 -0.502492  0.085544 -0.125929  ...  0.858654  2.609278  3.460466  0.395456 -0.096212  4.674790
3 -1.194727  0.449611  1.440556 -1.101091 -0.399705  0.583114  ...  1.654442  2.609278  4.336762  0.382004 -0.017102  4.674790
4 -1.179329 -0.010794  0.419520 -0.823188 -1.039352  0.002691  ... -0.484237 -0.383248  3.072971  0.381043 -0.264751 -0.213913

[5 rows x 50 columns]
PS C:\Users\vaibh\Downloads>
```

Q3. Derive some insights from the dataset.

Answer:

```python
# Summary Insights
print("\nSummary Insights:")
print("- Dataset was loaded, duplicates were removed.")
print("- Missing values in numerical columns were imputed with column means.")
print("- Categorical features were one-hot encoded.")
print("- Relevant features and target columns were selected.")
print("- Numerical features were scaled using StandardScaler.")
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

```
Summary Insights:
- Dataset was loaded, duplicates were removed.
- Missing values in numerical columns were imputed with column means.
- Categorical features were one-hot encoded.
- Relevant features and target columns were selected.
- Numerical features were scaled using StandardScaler.
PS C:\Users\vaibh\Downloads>
```

Q4. Apply various Machine Learning techniques to predict the output in the target column, make use of Bagging and Ensemble as required, and find the best model by evaluating the model using Model evaluation techniques.

Answer:

```
import pandas as pd
df = pd.read_csv("oil_spill.csv")
# Display the first few rows
print(df.head())
# Remove duplicates
df = df.drop_duplicates()
# Handle missing values
df.fillna(df.mean(), inplace=True)

from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
# Define feature columns and target column
features = ['f_1', 'f_2']
target = 'target'
# Separate features and target
X = df[features]
y = df[target]
# Define preprocessing steps
numeric_features = ['f_1', 'f_2']
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='mean')),
    ('scaler', StandardScaler())])
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))])
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features)])

from sklearn.model_selection import train_test_split
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
# Define the models
models = {
    'Logistic Regression': LogisticRegression(),
    'Random Forest': RandomForestClassifier(),
    'Gradient Boosting': GradientBoostingClassifier()
}
# Train the models
trained_models = {}
for name, model in models.items():
    pipeline = Pipeline(steps=[('preprocessor', preprocessor), ('classifier', model)])
    trained_models[name] = pipeline.fit(X_train, y_train)

from sklearn.metrics import accuracy_score, roc_auc_score
# Evaluate the models
for name, model in trained_models.items():
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    auc = roc_auc_score(y_test, model.predict_proba(X_test)[:, 1])
    print(f"{name} - Accuracy: {accuracy:.4f}, AUC: {auc:.4f}")
```

```
# Make predictions with the best model
best_model = trained_models['Random Forest']
y_pred = best_model.predict(X_test)

# Show predictions
print(y_pred)
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

```
PS C:\Users\vaibh\Downloads> & C:/Users/vaibh/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/vaibh/Downloads/Oil Spill"
   f_1    f_2      f_3     f_4  f_5       f_6    f_7   f_8  ...     f_43     f_44   f_45  f_46      f_47   f_48  f_49  target
0    1   2558  1506.09  456.63   90   6395000  40.88  7.89  ...   763.16   135.46   3.73     0  33243.19  65.74  7.95       1
1    2  22325    79.11  841.03  180  55812500  51.11  1.21  ...  9593.48  1648.80   0.60     0  51572.04  65.73  6.26       0
2    3    115  1449.85  608.43   88    287500  40.42  7.34  ...   150.00    45.13   9.33     1  31692.84  65.81  7.84       1
3    4   1201  1562.53  295.65   66   3002500  42.40  7.97  ...   453.21   144.97  13.33     1  37696.21  65.67  8.07       1
4    5    312   950.27  440.86   37    780000  41.43  7.03  ...   512.54   109.16   2.58     0  29038.17  65.66  7.35       0

[5 rows x 50 columns]
Logistic Regression - Accuracy: 0.9681, AUC: 0.8132
Random Forest - Accuracy: 0.9574, AUC: 0.7454
Gradient Boosting - Accuracy: 0.9415, AUC: 0.8265
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
 0 0 0]
PS C:\Users\vaibh\Downloads>
```

Q5. Save the best model and load the model.

Answer:

```python
# Save the best model
best_model = trained_models[best_model_name]
joblib_file = "best_model.pkl"
joblib.dump(best_model, joblib_file)
print(f"Best model saved as {joblib_file}")

# Load the best model
loaded_model = joblib.load(joblib_file)

# Use the loaded model to make predictions
y_pred_loaded = loaded_model.predict(X_test)

# Show predictions
print(y_pred_loaded)
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

```
Logistic Regression - Accuracy: 0.9681, AUC: 0.8132
Random Forest - Accuracy: 0.9521, AUC: 0.7495
Gradient Boosting - Accuracy: 0.9415, AUC: 0.8265
Best model: Gradient Boosting with AUC: 0.8265
Best model saved as best_model.pkl
[0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0]
PS C:\Users\vaibh\Downloads>
```

Q6. Take the original data set and make another dataset by randomly picking 20 datapoints from the oil spill data set and applying the saved model to the same.

Answer:

```python
# Train Multiple Models and Save the Best Model
models = {
    'Logistic Regression': LogisticRegression(),
    'Random Forest': RandomForestClassifier(),
    'Gradient Boosting': GradientBoostingClassifier()
}

trained_models = {}
best_model_name = None
best_auc = 0

for name, model in models.items():
    pipeline = Pipeline(steps=[('preprocessor', preprocessor), ('classifier', model)])
    trained_models[name] = pipeline.fit(X_train, y_train)

    # Model evaluation
    y_pred = pipeline.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    auc = roc_auc_score(y_test, pipeline.predict_proba(X_test)[:, 1])
    print(f"{name} - Accuracy: {accuracy:.4f}, AUC: {auc:.4f}")
    if auc > best_auc:
        best_auc = auc
        best_model_name = name
```

NAME: DIVYA STUTI
E-MAIL: divyastuti.DS@gmail.com
COURSE NAME: Data Science with Python Career Program (ChatGPT Included)
ASSIGNMENT: Machine Learning (MAJOR)

```python
print(f"Best model: {best_model_name} with AUC: {best_auc:.4f}")

# Save the best model
best_model = trained_models[best_model_name]
joblib_file = "best_model.pkl"
joblib.dump(best_model, joblib_file)
print(f"Best model saved as {joblib_file}")

# 3. Randomly Select 20 Data Points
df_sample = df.sample(n=20, random_state=42)

# 4. Apply the Saved Model to the Selected Data Points
X_sample = df_sample[features]

# 5. Load the Best Model and Make Predictions
loaded_model = joblib.load(joblib_file)
y_pred_sample = loaded_model.predict(X_sample)

# Display Results
df_sample['predicted_target'] = y_pred_sample
print(df_sample)
```

```
Best model saved as best_model.pkl
     f_1  f_2     f_3     f_4  f_5     f_6    f_7    f_8  ...    f_44   f_45  f_46     f_47   f_48   f_49  target  predicted_target
321   29  105  881.92 1128.79   83  262500  38.90   8.51  ...   84.74   4.21     0  3425.75  65.97   7.04       0                 0
70    60  111 1153.32 1283.44   41  277500  41.25   5.98  ...  140.92   2.40     0  5915.80  66.12   7.34       0                 0
209   17  867 1059.49  581.31   46 2167500  31.08   8.26  ...  408.01   4.93     0  5679.31  65.74   7.42       0                 0
656    9   85   71.06  469.47  140  688500  70.85  11.28  ...   87.51   3.95     0  6376.53  65.98   6.22       0                 0
685   38   15   32.47  582.13  156  121500  73.27  12.11  ...   84.66   6.47     0  3285.95  66.11   5.98       0                 0
96    86   86  769.73 1761.26   55  215000  37.55   6.27  ...   59.34  14.93     1 15720.91  66.30   6.71       0                 0
468   36  462  904.13 2689.99  129  649687  29.80   8.99  ...    0.00   0.00     0 40916.70  36.71  14.53       0                 0
86    76  128 1378.47  929.73   51  320000  39.80   5.20  ...   94.32   8.43     0  9183.53  65.98   7.73       0                 0
532   38  294   11.49 1559.36   40  413437  38.12  22.22  ...    0.00   0.00     0 10484.87  36.02  14.82       0                 0
327   37   98 1326.06 1109.08   72  245000  41.31   7.53  ...   33.61   3.71     0  7233.16  66.02   7.54       0                 0
528   34  151  465.77 1736.15   73  212343  28.96   8.14  ...    0.00   0.00     0  8415.67  36.35  14.83       0                 0
247  138  144 1341.72   78.22  110  360000  31.12   6.88  ...   60.43  11.49     1  6824.45  65.55   7.90       0                 0
250  156  260 1080.89  833.29  111  650000  30.52   7.95  ...  161.45   5.93     0  4667.21  65.86   7.36       0                 0
485   53   84  575.19 1558.81  153  118125  30.94   8.89  ...    0.00   0.00     0 10674.79  36.41  14.92       0                 1
467   35   74  619.18 1622.32    5  104062  26.45   5.92  ...    0.00   0.00     0 11277.47  36.44  14.90       0                 0
723   76   10   30.80  348.90  153   81000  70.50   8.93  ...  146.97   3.82     0 11172.62  65.80   6.22       0                 0
483   51   60  743.88 1250.60  127   84375  33.03  11.87  ...  109.90   2.95     0  9370.56  36.51  15.08       0                 0
886  154   10  182.50  460.00   90   81000  57.60   8.68  ...    0.00   4.00     0  6004.08  66.01   6.58       0                 0
809   77   13  160.77  420.23   63  105300  51.15  10.66  ...   56.92  20.62     0  3719.47  65.95   6.55       0                 0
244  118  308 1313.18  791.35   61  770000  29.13   7.14  ...  181.66   4.29     0  6636.30  65.87   7.63       0                 0

[20 rows x 51 columns]
PS C:\Users\vaibh\Downloads>
```