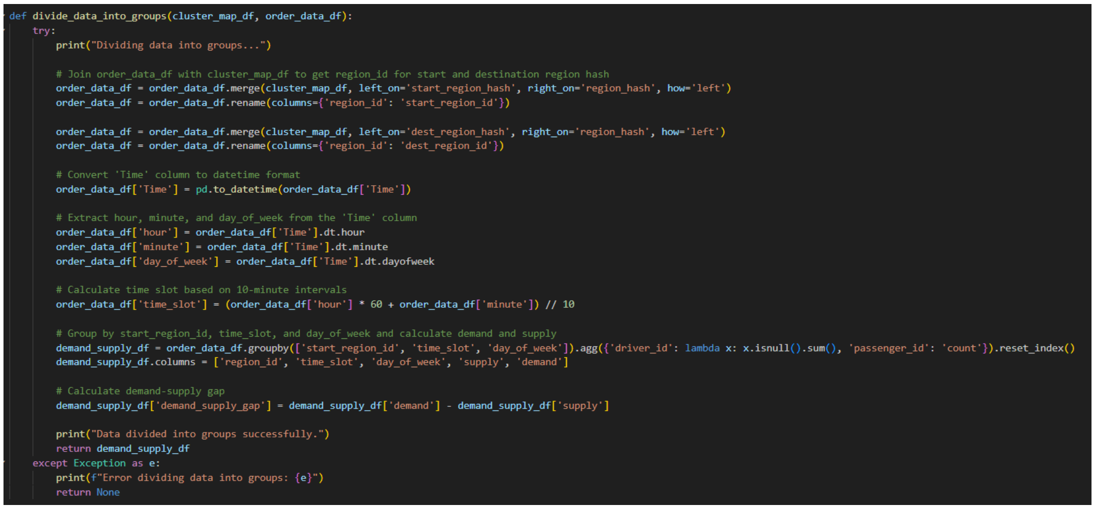
Muhammad Mahad Sheikh

21i1239

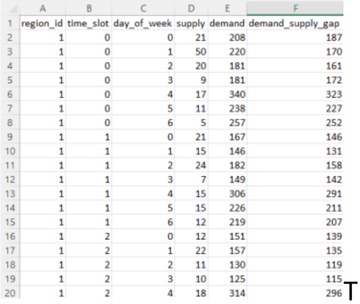
SE-P

1. **Data Preparation**:

    - The dataset was divided into groups based on a 10-minute timestamp and regions.
    - Each group was analyzed to find the supply-demand gap.

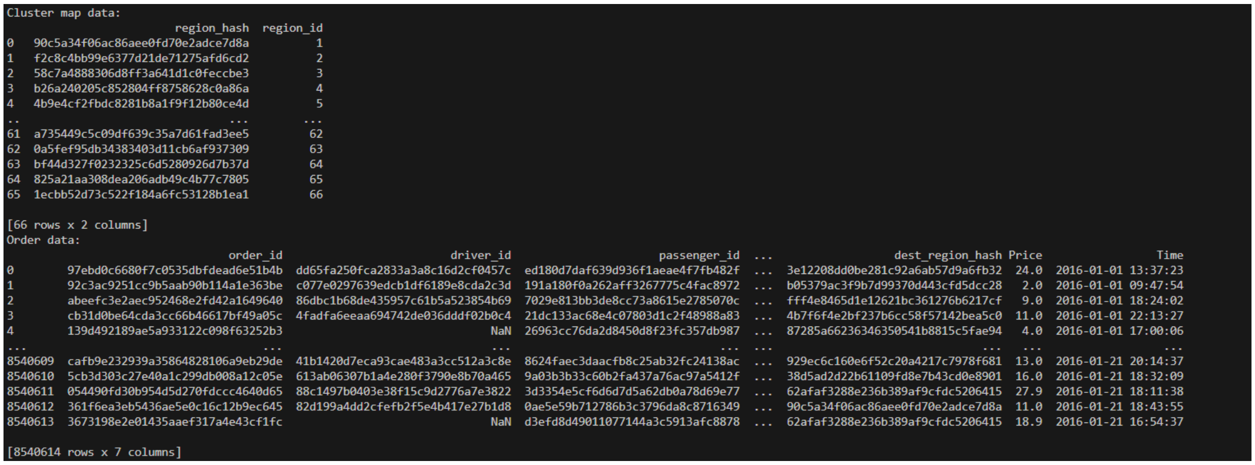


```python
def divide_data_into_groups(cluster_map_df, order_data_df):
    try:
        print("Dividing data into groups...")

        # Join order_data_df with cluster_map_df to get region_id for start and destination region hash
        order_data_df = order_data_df.merge(cluster_map_df, left_on='start_region_hash', right_on='region_hash', how='left')
        order_data_df = order_data_df.rename(columns={'region_id': 'start_region_id'})

        order_data_df = order_data_df.merge(cluster_map_df, left_on='dest_region_hash', right_on='region_hash', how='left')
        order_data_df = order_data_df.rename(columns={'region_id': 'dest_region_id'})

        # Convert 'Time' column to datetime format
        order_data_df['Time'] = pd.to_datetime(order_data_df['Time'])

        # Extract hour, minute, and day_of_week from the 'Time' column
        order_data_df['hour'] = order_data_df['Time'].dt.hour
        order_data_df['minute'] = order_data_df['Time'].dt.minute
        order_data_df['day_of_week'] = order_data_df['Time'].dt.dayofweek

        # Calculate time slot based on 10-minute intervals
        order_data_df['time_slot'] = (order_data_df['hour'] * 60 + order_data_df['minute']) // 10

        # Group by start_region_id, time_slot, and day_of_week and calculate demand and supply
        demand_supply_df = order_data_df.groupby(['start_region_id', 'time_slot', 'day_of_week']).agg({'driver_id': lambda x: x.isnull().sum(), 'passenger_id': 'count'}).reset_index()
        demand_supply_df.columns = ['region_id', 'time_slot', 'day_of_week', 'supply', 'demand']

        # Calculate demand-supply gap
        demand_supply_df['demand_supply_gap'] = demand_supply_df['demand'] - demand_supply_df['supply']

        print("Data divided into groups successfully.")
        return demand_supply_df
    except Exception as e:
        print(f"Error dividing data into groups: {e}")
        return None
```



region_id	time_slot	day_of_week	supply	demand	demand_supply_gap
1	0	0	21	208	187
1	0	1	50	220	170
1	0	2	20	181	161
1	0	3	9	181	172
1	0	4	17	340	323
1	0	5	11	238	227
1	0	6	5	257	252
1	1	0	21	167	146
1	1	1	15	146	131
1	1	2	24	182	158
1	1	3	7	149	142
1	1	4	15	306	291
1	1	5	15	226	211
1	1	6	12	219	207
1	2	0	12	151	139
1	2	1	22	157	135
1	2	2	11	130	119
1	2	3	10	125	115
1	2	4	18	314	296

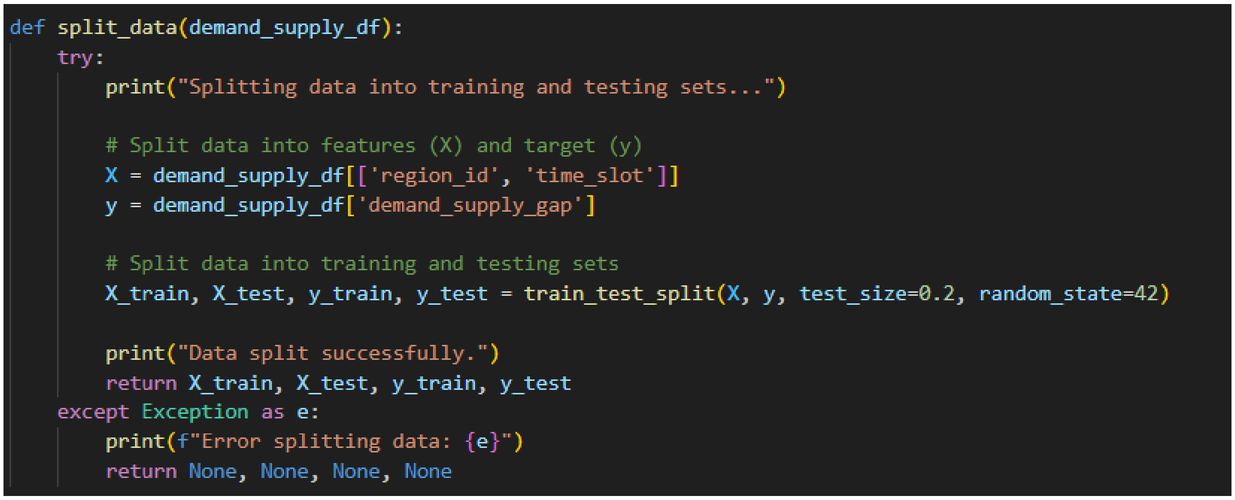
There is more data as well 0-60192 groups



```
Cluster map data:
                         region_hash  region_id
0   90c5a34f06ac86aee0fd70e2adce7d8a          1
1   f2c8c4bb99e6377d21de71275afd6cd2          2
2   58c7a4888306d8ff3a641d1c0feccbe3          3
3   b26a240205c852804ff8758628c0a86a          4
4   4b9e4cf2fbdc8281b8a1f9f12b80ce4d          5
..                               ...        ...
61  a735449c5c09df639c35a7d61fad3ee5         62
62  0a5fef95db34383403d11cb6af937309         63
63  bf44d327f0232325c6d5280926d7b37d         64
64  825a21aa308dea206adb49c4b77c7805         65
65  1ecbb52d73c522f184a6fc53128b1ea1         66

[66 rows x 2 columns]
Order data:
                                 order_id                         driver_id                      passenger_id  ...                   dest_region_hash Price                Time
0        97ebd0c6680f7c0535dbfdead6e51b4b  dd65fa250fca2833a3a8c16d2cf0457c  ed180d7daf639d936f1aeae4f7fb482f  ...  3e12208dd0be281c92a6ab57d9a6fb32  24.0 2016-01-01 13:37:23
1        92c3ac9251cc9b5aab90b114a1e363be  c077e0297639edcb1df6189e8cda2c3d  191a180f0a262aff3267775c4fac8972  ...  b05379ac3f9b7d99370d443cfd5dcc28   2.0 2016-01-01 09:47:54
2        abeefc3e2aec952468e2fd42a1649640  86dbc1b68de435957c61b5a523854b69  7029e813bb3de8cc73a8615e2785070c  ...  fff4e8465d1e12621bc361276b6217cf   9.0 2016-01-01 18:24:02
3        cb31d0be64cda3cc66b46617bf49a05c  4fadfa6eeaa694742de036dddf02b0c4  21dc133ac68e4c07803d1c2f48988a83  ...  4b7f6f4e2bf237b6cc58f57142bea5c0  11.0 2016-01-01 22:13:27
4        139d492189ae5a933122c098f63252b3                               NaN  26963cc76da2d8450d8f23fc357db987  ...  87285a66236346350541b8815c5fae94   4.0 2016-01-01 17:00:06
...                                   ...                               ...                               ...  ...                               ...   ...                 ...
8540609  cafb9e232939a35864828106a9eb29de  41b1420d7eca93cae483a3cc512a3c8e  8624faec3daacfb8c25ab32fc24138ac  ...  929ec6c160e6f52c20a4217c7978f681  13.0 2016-01-21 20:14:37
8540610  5cb3d303c27e40a1c299db008a12c05e  613ab06307b1a4e280f3790e8b70a465  9a03b3b33c60b2fa437a76ac97a5412f  ...  38d5ad2d22b61109fd8e7b43cd0e8901  16.0 2016-01-21 18:32:09
8540611  054490fd30b954d5d270fdccc4640d65  88c1497b0403e38f15c9d2776a7e3822  3d3354e5cf6d6d7d5a62db0a78d69e77  ...  62afaf3288e236b389af9cfdc5206415  27.9 2016-01-21 18:11:38
8540612  361f6ea3eb5436ae5e0c16c12b9ec645  82d199a4dd2cfefb2f5e4b417e27b1d8  0ae5e59b712786b3c3796da8c8716349  ...  90c5a34f06ac86aee0fd70e2adce7d8a  11.0 2016-01-21 18:43:55
8540613  3673198e2e01435aaef317a4e43cf1fc                               NaN  d3efd8d49011077144a3c5913afc8878  ...  62afaf3288e236b389af9cfdc5206415  18.9 2016-01-21 16:54:37

[8540614 rows x 7 columns]
```

2. **Feature Engineering**:

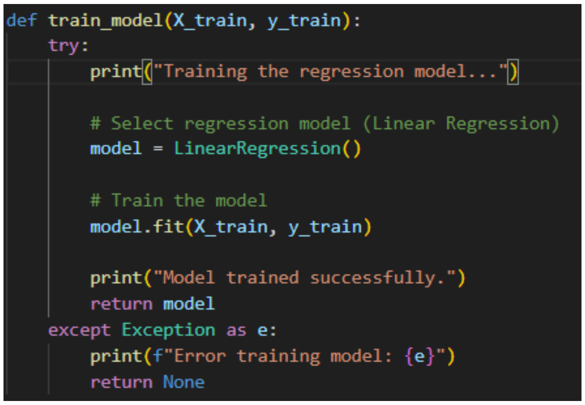
    - Each region_id and time_slot were assigned as the independent variable (X).
    - The demand_supply_gap was considered as the dependent variable (y).



```python
def split_data(demand_supply_df):
    try:
        print("Splitting data into training and testing sets...")

        # Split data into features (X) and target (y)
        X = demand_supply_df[['region_id', 'time_slot']]
        y = demand_supply_df['demand_supply_gap']

        # Split 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)

        print("Data split successfully.")
        return X_train, X_test, y_train, y_test
    except Exception as e:
        print(f"Error splitting data: {e}")
        return None, None, None, None
```

3. **Model Training**:

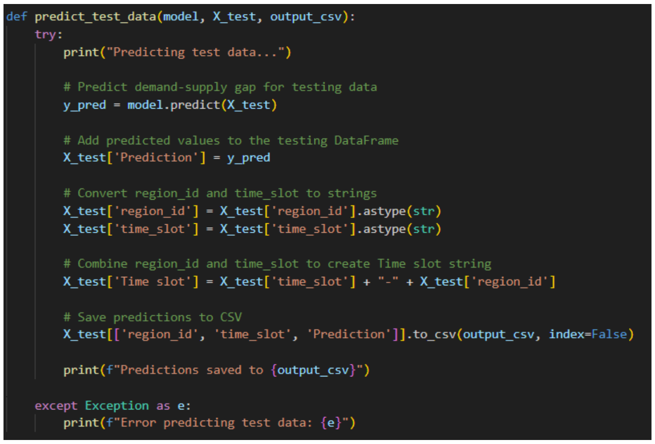
    - Sklearn was utilized to train and test the model.
    - A regression model was chosen for training.
    - The model was fitted using the training data.



```python
def train_model(X_train, y_train):
    try:
        print("Training the regression model...")

        # Select regression model (Linear Regression)
        model = LinearRegression()

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

        print("Model trained successfully.")
        return model
    except Exception as e:
        print(f"Error training model: {e}")
        return None
```

4. **Testing**:

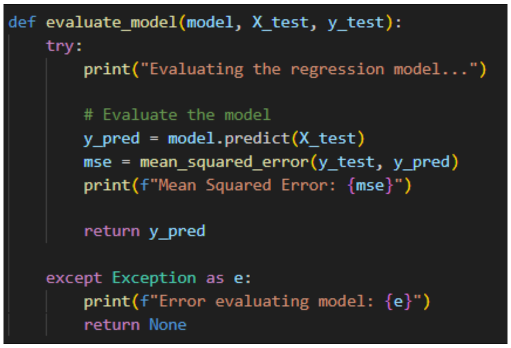
    - The trained model was used to predict data on the testing set for all regions.
    - Predictions were made for each region ID and time slot combination.
    - The output data contained three fields:
        - **Region ID**: String representing the region mapping ID.
        - **Time Slot**: String representing the timestamp (e.g., 2016-01-23-1 for the first time slot on Jan. 23rd, 2016).
        - **Prediction**: Double value representing the predicted supply-demand gap.



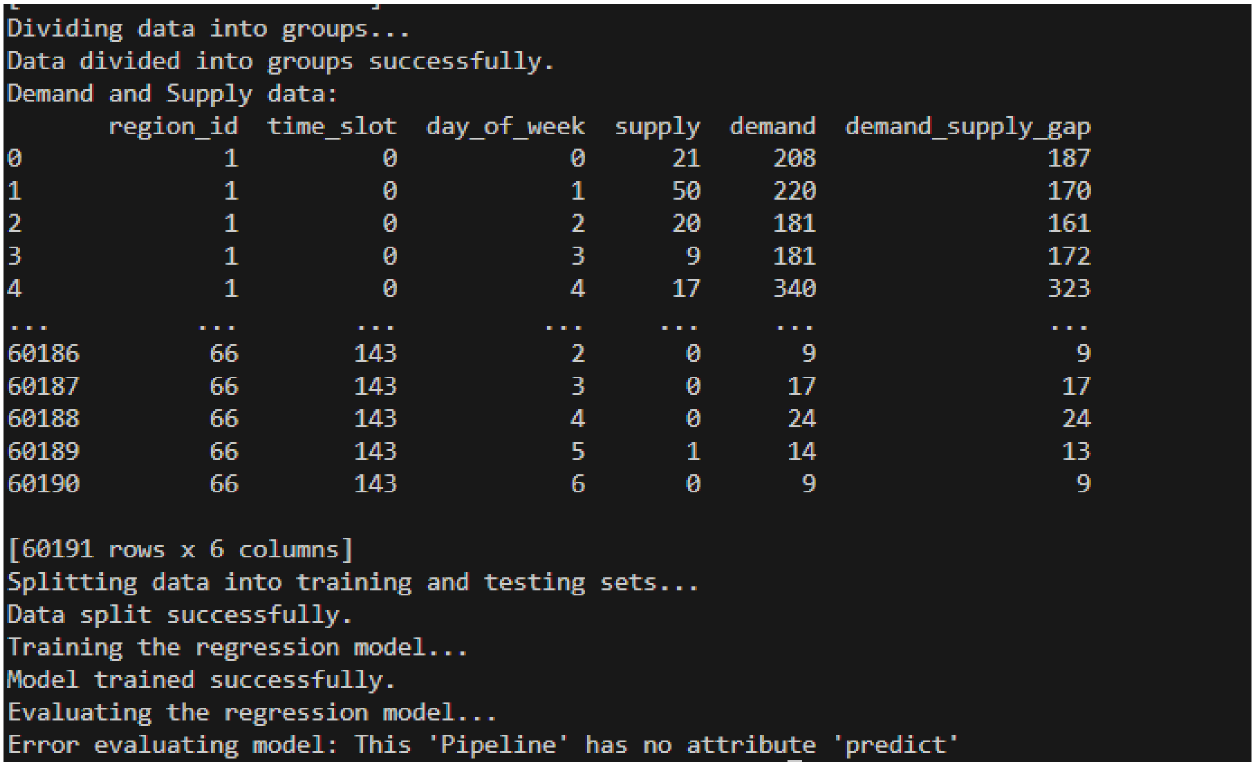
```python
def predict_test_data(model, X_test, output_csv):
    try:
        print("Predicting test data...")

        # Predict demand-supply gap for testing data
        y_pred = model.predict(X_test)

        # Add predicted values to the testing DataFrame
        X_test['Prediction'] = y_pred

        # Convert region_id and time_slot to strings
        X_test['region_id'] = X_test['region_id'].astype(str)
        X_test['time_slot'] = X_test['time_slot'].astype(str)

        # Combine region_id and time_slot to create Time slot string
        X_test['Time slot'] = X_test['time_slot'] + "-" + X_test['region_id']

        # Save predictions to CSV
        X_test[['region_id', 'time_slot', 'Prediction']].to_csv(output_csv, index=False)

        print(f"Predictions saved to {output_csv}")

    except Exception as e:
        print(f"Error predicting test data: {e}")
```

5. **Accuracy Evaluation**:

    - Mean squared error (MSE) was calculated to evaluate the accuracy of the model.



```python
def evaluate_model(model, X_test, y_test):
    try:
        print("Evaluating the regression model...")

        # Evaluate the model
        y_pred = model.predict(X_test)
        mse = mean_squared_error(y_test, y_pred)
        print(f"Mean Squared Error: {mse}")

        return y_pred

    except Exception as e:
        print(f"Error evaluating model: {e}")
        return None
```



```
Dividing data into groups...
Data divided into groups successfully.
Demand and Supply data:
       region_id  time_slot  day_of_week  supply  demand  demand_supply_gap
0              1          0            0      21     208                187
1              1          0            1      50     220                170
2              1          0            2      20     181                161
3              1          0            3       9     181                172
4              1          0            4      17     340                323
...          ...        ...          ...     ...     ...                ...
60186         66        143            2       0       9                  9
60187         66        143            3       0      17                 17
60188         66        143            4       0      24                 24
60189         66        143            5       1      14                 13
60190         66        143            6       0       9                  9

[60191 rows x 6 columns]
Splitting data into training and testing sets...
Data split successfully.
Training the regression model...
Model trained successfully.
Evaluating the regression model...
Error evaluating model: This 'Pipeline' has no attribute 'predict'
```

Overall, the assignment involved data preprocessing, feature engineering, model training, prediction, and accuracy evaluation using mean squared error. The process aimed to develop a regression model to predict the supply-demand gap for different regions and timestamps.