Set up Spark session

python

CopyEdit

from pyspark.sql import SparkSession

- Import SparkSession – the main entry point to use Spark.

python

CopyEdit

from pyspark.sql.functions import when, col, trim, lower

- Import useful SQL functions:
    - when: for conditional logic (like if-else)
    - col: to refer to DataFrame columns
    - trim: removes spaces
    - lower: makes text lowercase

python

CopyEdit

from pyspark.ml.feature import VectorAssembler

- Tool to combine multiple columns into a **single feature vector** (needed for ML models).

python

CopyEdit

from pyspark.ml.regression import LinearRegression

- Import Linear Regression from MLlib to predict traffic conditions.

python

CopyEdit

from pyspark.sql.types import DoubleType

- We use this to convert columns to type Double, which MLlib models require.

2️⃣ Create a Spark session

python

CopyEdit

spark = SparkSession.builder \

  .appName("TrafficPrediction") \

.getOrCreate()

- Starts a new Spark app named "TrafficPrediction".

3️⃣ Load the CSV file

python

CopyEdit

df = spark.read.csv("/content/Traffic.csv", header=True, inferSchema=True)

- Loads a file named Traffic.csv.
- header=True: treats first row as column names.
- inferSchema=True: guesses the type of each column automatically.

python

CopyEdit

print("Total rows before processing:", df.count())

- Shows total number of rows loaded from the file.

4️⃣ Clean and prepare 'Traffic Situation' column

python

CopyEdit

df = df.withColumn("Traffic Situation", trim(lower(col("Traffic Situation"))))

- Removes extra spaces and converts text in the "Traffic Situation" column to lowercase (e.g., " Low " → "low").

python

CopyEdit

df.select("Traffic Situation").distinct().show(truncate=False)

- Prints all unique values in the "Traffic Situation" column to help check for spelling or formatting issues.

5️⃣ Convert text labels to numbers

python

CopyEdit

df = df.withColumn(

```
    "Traffic Situation",
    when(col("Traffic Situation") == "low", 0)
    .when(col("Traffic Situation") == "moderate", 1)
    .when(col("Traffic Situation") == "heavy", 2)
    .otherwise(None)
)
```

- Converts:
  - "low" → 0
  - "moderate" → 1
  - "heavy" → 2
- Everything else → None (null)

python

CopyEdit

```
df = df.dropna(subset=["Traffic Situation"])
print("Rows after mapping:", df.count())
```

- Removes rows where the Traffic Situation was None (invalid or unmapped).

6️⃣ Convert label to numeric type

python

CopyEdit

```
df = df.withColumn("Traffic Situation", col("Traffic Situation").cast(DoubleType()))
```

- Converts "Traffic Situation" to **Double**, required by MLlib.

7️⃣ Prepare features for the model

python

CopyEdit

```
feature_cols = ["CarCount", "BikeCount", "BusCount", "TruckCount", "Total"]
```

- These are the input features used to predict traffic.

python

CopyEdit

```
assembler = VectorAssembler(inputCols=feature_cols, outputCol="features")
```

df = assembler.transform(df)

- Combines the 5 columns into a single **feature vector** column.

8️⃣ Split data for training and testing

python

CopyEdit

train_df, test_df = df.randomSplit([0.8, 0.2], seed=42)

print("Train rows:", train_df.count())

print("Test rows:", test_df.count())

- Randomly splits the data:
    - 80% for training
    - 20% for testing
- seed=42: to make the split reproducible

9️⃣ Train a Linear Regression model

python

CopyEdit

lr = LinearRegression(featuresCol="features", labelCol="Traffic Situation")

model = lr.fit(train_df)

- Creates and fits a Linear Regression model using the training data.

🔟 Predict on test data

python

CopyEdit

predictions = model.transform(test_df)

- Uses the trained model to predict traffic on test data.

1️⃣1️⃣ Convert numeric predictions to categories

python

CopyEdit

predictions = predictions.withColumn(

```
    "Predicted Traffic Situation",
    when(col("prediction") < 0.5, 0)
    .when((col("prediction") >= 0.5) & (col("prediction") < 1.5), 1)
    .otherwise(2)
)
```

- Maps predicted values back to traffic categories:
    - < 0.5 → low (0)
    - 0.5 - 1.49 → moderate (1)
    - ≥ 1.5 → heavy (2)

---

12. Show some results

python

CopyEdit

```
predictions.select("features", "Traffic Situation", "prediction", "Predicted Traffic Situation").show(10, truncate=False)
```

- Displays 10 rows showing:
    - The input features
    - Actual traffic situation
    - Predicted value from the model
    - Final mapped traffic category

---

13. Stop Spark session

python

CopyEdit

```
spark.stop()
```

- Gracefully shuts down the Spark session.