BSDS-205 - Hadoop for Data Science

Course Instructor - Ms. Manpreet Bhatia

Gauri Sharan - BSc Data Science (Sem 3)

January 6, 2024

Outline

Starting a PySpark Session

```
!pip install pyspark

from pyspark.sql import SparkSession

spark =
SparkSession.builder.appName("mySparkSession")
.getOrCreate()
```

Reading a File

```
# Assuming a CSV file
df = spark.read.csv("path/to/your/file.csv",
header=True, inferSchema=True)
```

Data Preprocessing

- filter and orderBy
- Multiple filtering
- Dropping null values
- Filling missing values
- Imputer function
- Grouping and aggregating

Filter and OrderBy

```
df_filtered = df.filter(df['column_name'] > 50)
.orderBy('column_name')
```

Multiple Filtering

```
df_filtered = df.filter((df['column1'] > 50) &
(df['column2'] == 'value'))
```

Dropping Null Values

```
df_no_nulls = df.dropna()
```

Filling Missing Values

```
df_filled = df.fillna(value=0)
```

Imputer Function

```
from pyspark.ml.feature import Imputer

imputer = Imputer(strategy='mean', inputCols=
['column1', 'column2'], outputCols=['column1',
'column2_imputed'])
df_imputed = imputer.fit(df).transform(df)
```

Grouping and Aggregating

```
df_grouped = df.groupBy('grouping_column')
.agg({'agg_column': 'sum'})
```

Exploratory Data Analysis

- UDF (User-Defined Functions)
- Summary table
- Identifying numeric features
- Extracting numeric features
- Correlation

User-Defined Functions (UDF)

```
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType

# Define the Python function
def my_udf_function(value):
    return str(value) + '_processed'

# Register the UDF
udf_function = udf(my_udf_function, StringType())

# Apply the UDF
df_udf_applied = df.withColumn('new_column',
udf_function(df['existing_column']))
```

Summary Table

```
df_summary = df.describe()
```

Identifying Numeric Functions

```
numeric_columns = [col[0] for col in df.dtypes
if col[1] in ['int', 'double']]
df_numeric = df.select(numeric_columns)
```

Extracting Numeric Features

```
from pyspark.ml.feature import VectorAssembler

feature_columns = ['feature1', 'feature2',
'feature3']
assembler = VectorAssembler(inputCols=feature_columns
, outputCol='features')
df_assembled = assembler.transform(df)
```

Correlation

```
from pyspark.ml.stat import Correlation
from pyspark.ml.feature import VectorAssembler

feature_columns = ['feature1', 'feature2',
'feature3']
assembler = VectorAssembler(inputCols=feature_columns
, outputCol='features')
df_assembled = assembler.transform(df)

correlation_matrix =
Correlation.corr(df_assembled, 'features').head()
```