databricks ACT20

(https://databricks.com)

```
filePath = """/databricks-datasets/learning-spark-v2/sf-airbnb/sf-airbnb-clean.parquet/"""
airbnbDF = spark.read.parquet(filePath)
airbnbDF.select("neighbourhood_cleansed", "room_type", "bedrooms", "bathrooms",
"number_of_reviews", "price").show(5)
```

```
+--------------------+---------------+--------+---------+----------------+-----+
|neighbourhood_cleansed|      room_type|bedrooms|bathrooms|number_of_reviews|price|
+--------------------+---------------+--------+---------+----------------+-----+
|      Western Addition|Entire home/apt|     1.0|      1.0|           180.0|170.0|
|        Bernal Heights|Entire home/apt|     2.0|      1.0|           111.0|235.0|
|        Haight Ashbury|   Private room|     1.0|      4.0|            17.0| 65.0|
|        Haight Ashbury|   Private room|     1.0|      4.0|             8.0| 65.0|
|      Western Addition|Entire home/apt|     2.0|      1.5|            27.0|785.0|
+--------------------+---------------+--------+---------+----------------+-----+
only showing top 5 rows
```

```
airbnbDF.display()
```

Visualization

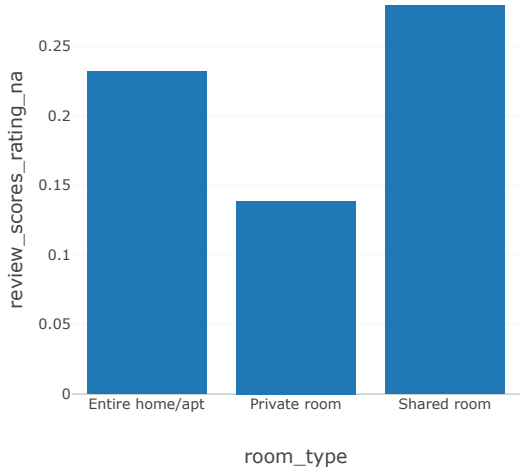
New result table: ON

Truncated to 1,000 rows.

```
airbnbDF.display()
```

Visualization

New result table: ON



3 rows

```
trainDF, testDF = airbnbDF.randomSplit([.8, .2], seed=42)
print(f"""There are {trainDF.count()} rows in the training set,
and {testDF.count()} in the test set""")
#There are 5780 rows in the training set, and 1366 in the test set
```

```
There are 5780 rows in the training set,
and 1366 in the test set
```

```
# In Python
from pyspark.ml.feature import VectorAssembler
vecAssembler = VectorAssembler(inputCols=["bedrooms"], outputCol="features")
#VectorAssembler takes a list of input columns and creates a new DataFrame with an additional column
vecTrainDF = vecAssembler.transform(trainDF)
vecTrainDF.select("bedrooms", "features", "price").show(10)
```

```
+--------+--------+-----+
|bedrooms|features|price|
+--------+--------+-----+
|     1.0|   [1.0]|200.0|
|     1.0|   [1.0]|130.0|
|     1.0|   [1.0]| 95.0|
|     1.0|   [1.0]|250.0|
|     3.0|   [3.0]|250.0|
|     1.0|   [1.0]|115.0|
|     1.0|   [1.0]|105.0|
|     1.0|   [1.0]| 86.0|
|     1.0|   [1.0]|100.0|
|     2.0|   [2.0]|220.0|
```

```
+--------+--------+-----+
only showing top 10 rows
```

```
from pyspark.ml.regression import LinearRegression
lr = LinearRegression(featuresCol="features", labelCol="price")
lrModel = lr.fit(vecTrainDF)
```

```
m = round(lrModel.coefficients[0], 2)
b = round(lrModel.intercept, 2)
print(f"""The formula for the linear regression line is
price = {m}*bedrooms + {b}""")
#price=123.68*bedrooms+47.51
```

```
The formula for the linear regression line is
price = 123.68*bedrooms + 47.51
```

```
from pyspark.ml import Pipeline
pipeline = Pipeline(stages=[vecAssembler, lr])
pipelineModel = pipeline.fit(trainDF)
```

```
predDF = pipelineModel.transform(testDF)
predDF.select("bedrooms", "features", "price", "prediction").show(10)
#Task finished!
```

```
+--------+--------+------+------------------+
|bedrooms|features| price|        prediction|
+--------+--------+------+------------------+
|     1.0|   [1.0]|  85.0|171.18598011578285|
|     1.0|   [1.0]|  45.0|171.18598011578285|
|     1.0|   [1.0]|  70.0|171.18598011578285|
|     1.0|   [1.0]| 128.0|171.18598011578285|
|     1.0|   [1.0]| 159.0|171.18598011578285|
|     2.0|   [2.0]| 250.0|294.86172649777757|
|     1.0|   [1.0]|  99.0|171.18598011578285|
|     1.0|   [1.0]|  95.0|171.18598011578285|
|     1.0|   [1.0]| 100.0|171.18598011578285|
|     1.0|   [1.0]|2010.0|171.18598011578285|
+--------+--------+------+------------------+
only showing top 10 rows
```

What is the format of the data source in this problem?

Parkett

What are the features and label in this problem?

Price is the label and all of the other columns are features

How many rows are there in the training and testing datasets respectively?

There are 5780 rows in the training set, and 1366 in the test set

What is the purpose of using the Vector Assembler in this problem?

VectorAssembler takes a list of input columns and creates a new DataFrame with an additional vector column. In other words, The modeling algorithms in Spark MLlib will only accept a vectorized column as input. This is done for reasons of efficiency and scaling.

What are the stages in the Pipeline?

The stages are:

1. Data extraction
2. Model Training
3. Model Evaluation
4. Model Deployment

What is the expected price of a 4 bedroom Airbnb rental in the San Francisco area?

The price is 542.23.
47.51 + 123.68 + 123.68 + 123.68 + 123.68 = 542.23
It is four bedrooms at 123.68 each in acordance to our model.

Here we terminate the cluster