

Machine Learning with Pyspark

Student Name: Prem Kumar Pulluri

Student Id: 2021470494

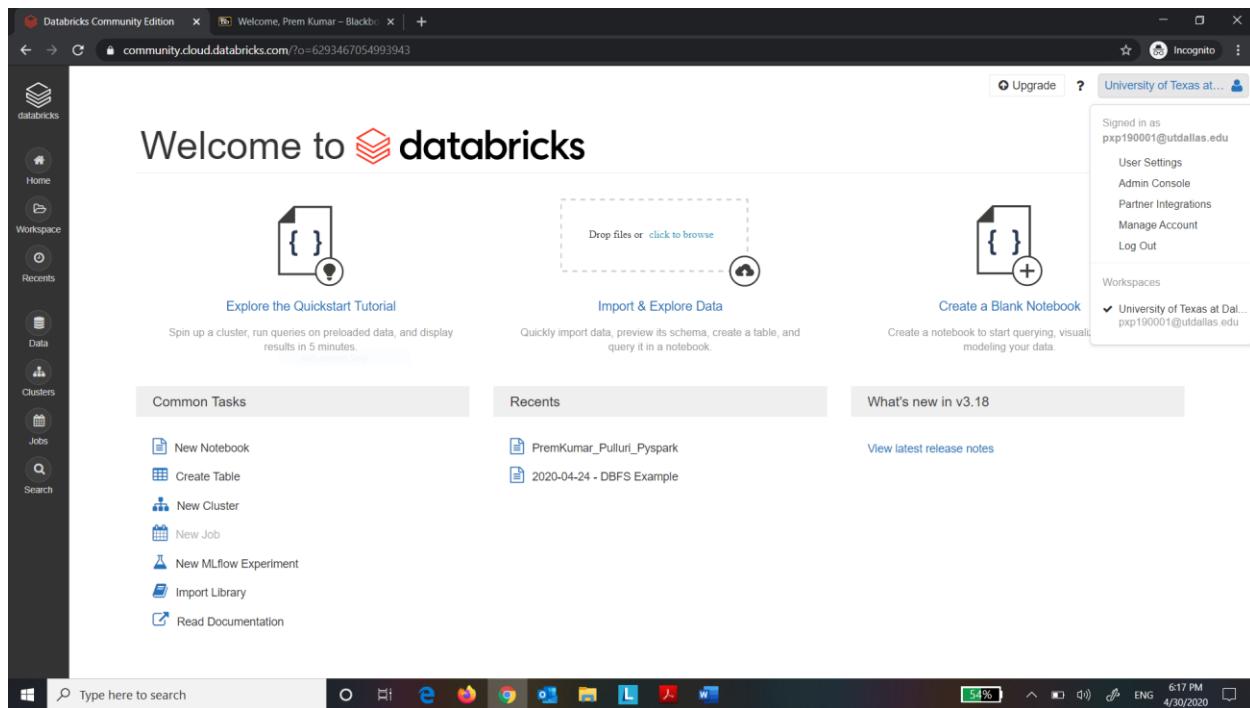
Date: 04/30/2020

Objective:

PySpark is a great language for performing exploratory data analysis and building machine learning pipelines. MLlib is a Spark implementation of useful machine learning (ML) functionality. The objective of this case study is to become familiar with Pyspark. We will load the data in Pyspark and build a machine learning model using the K-Means algorithm.

Part 1: Create a databricks account.

Q. Login to your databricks account and take a screenshot of the Home page output and paste it below.



Part 2: Create a cluster node.

Q. Take a screenshot of the Create Cluster page and paste it below.

The screenshot shows the 'Create Cluster' interface on the Databricks web interface. The page title is 'Create Cluster - Databricks Community'. The main heading is 'New Cluster'. A 'Create Cluster' button is prominently displayed. Below it, the configuration settings are listed:

- Cluster Name:** PremKumar_Pulluri_CaseStudy
- DataBricks Runtime Version:** Runtime: 6.4 (Scala 2.11, Spark 2.4.5)
- Instance:** Free 15GB Memory. As a Community Edition user, your cluster will automatically terminate after an idle period of two hours. For more configuration options, please upgrade your Databricks subscription.
- Availability Zone:** us-west-2c

The left sidebar includes links for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The bottom of the screen shows the Windows taskbar with various pinned icons and system status information.

Step 5: Green sign shows that a cluster is created and running successfully.

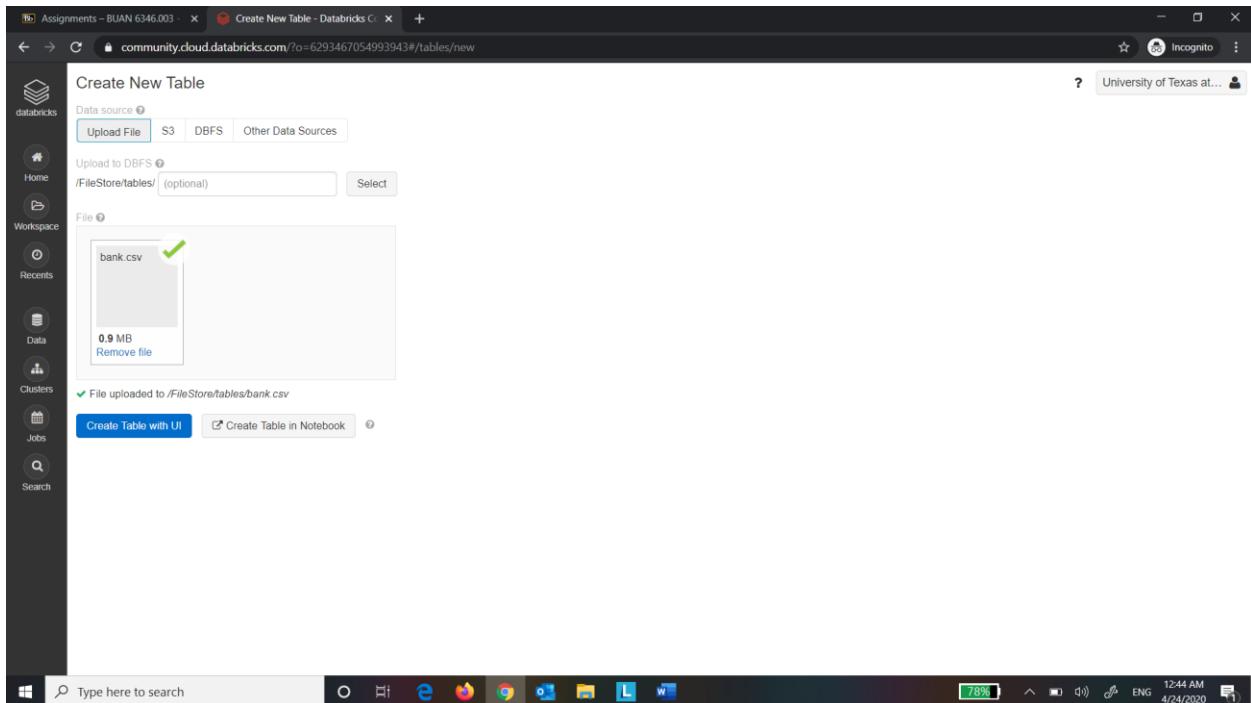
Q. Take a screenshot of the running Cluster Page and paste it below.

The screenshot shows the Databricks Cluster configuration page for a cluster named "PremKumar_Pulluri_CaseStudy". The top navigation bar includes tabs for "Edit", "Clone", "Restart", "Terminate", and "Delete". Below the navigation are sections for "Configuration", "Notebooks (0)", "Libraries", "Event Log", "Spark UI", "Driver Logs", "Metrics", "Apps", and "Spark Cluster UI - Master". The "Configuration" tab is selected. Under "Configuration", the "Databricks Runtime Version" is set to "6.4 (includes Apache Spark 2.4.5, Scala 2.11)". A note indicates that this version supports only Python 3. The "Driver Type" is set to "Community Optimized" with "15.3 GB Memory, 2 Cores, 1 DBU". In the "Instance" section, a message states: "Free 15GB Memory: As a Community Edition user, your cluster will automatically terminate after an idle period of two hours. For more configuration options, please upgrade your Databricks subscription." The "Availability Zone" is set to "us-west-2c". At the bottom of the page, there are tabs for "Instances", "Spark", "JDBC/ODBC", and "Permissions". The left sidebar contains icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The bottom of the screen shows a taskbar with various application icons and system status indicators.

Part 3: Create Datatable.

Step 7: Click on browse and select the “bank.csv” file that you downloaded from eLearning.

Take a screenshot of the uploaded file and paste it below.



Step 9: The file location can now be seen.

Q. Take a screenshot of the table created and paste it below.

Copy the file location and paste it here.

```
# File location and type
```

```
file_location = "/FileStore/tables/bank.csv"
```

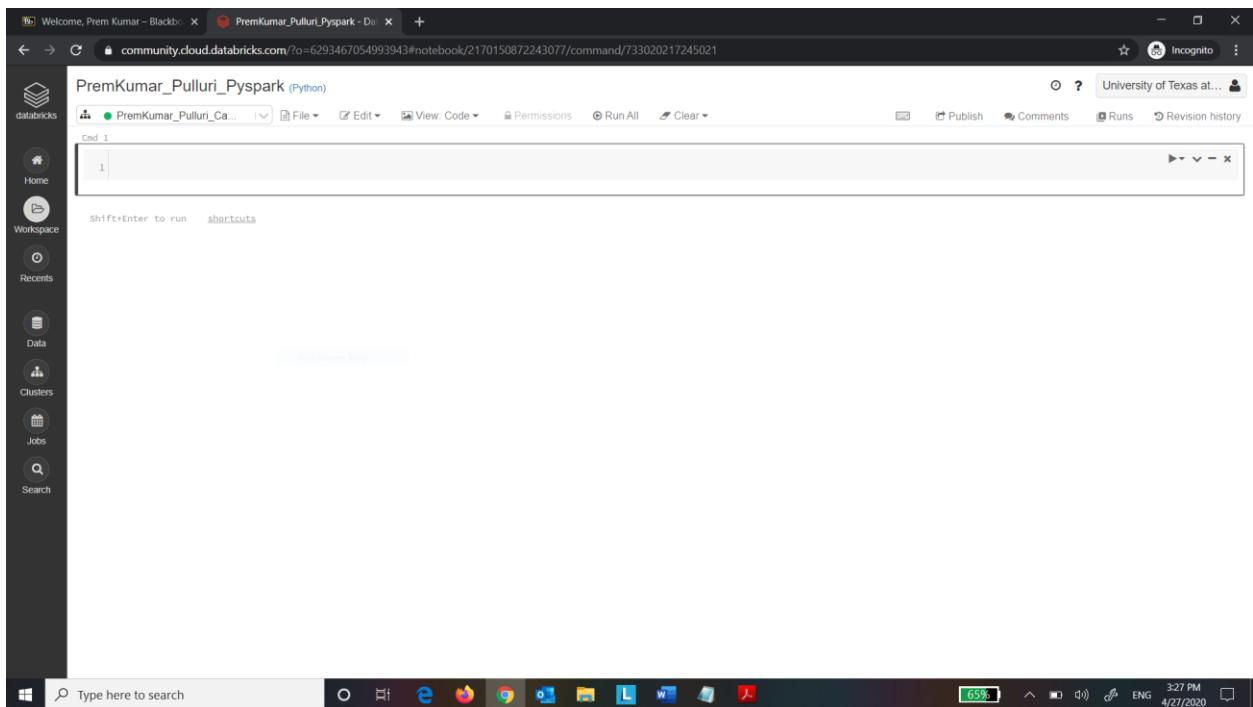
```
file_type = "csv"
```

```
1 # File location and type
2 file_location = "/FileStore/tables/bank.csv"
3 file_type = "csv"
4
5 # CSV options
6 infer_schema = "false"
7 first_row_is_header = "false"
8 delimiter = ","
9
10 # The applied options are for CSV files. For other file types, these will be ignored.
11 df = spark.read.format(file_type) \
12 .option("inferSchema", infer_schema) \
13 .option("header", first_row_is_header) \
14 .option("sep", delimiter) \
15 .load(file_location)
16
17 display(df)
```

```
1 # Create a view or table
2
3 temp_table_name = "bank_csv"
```

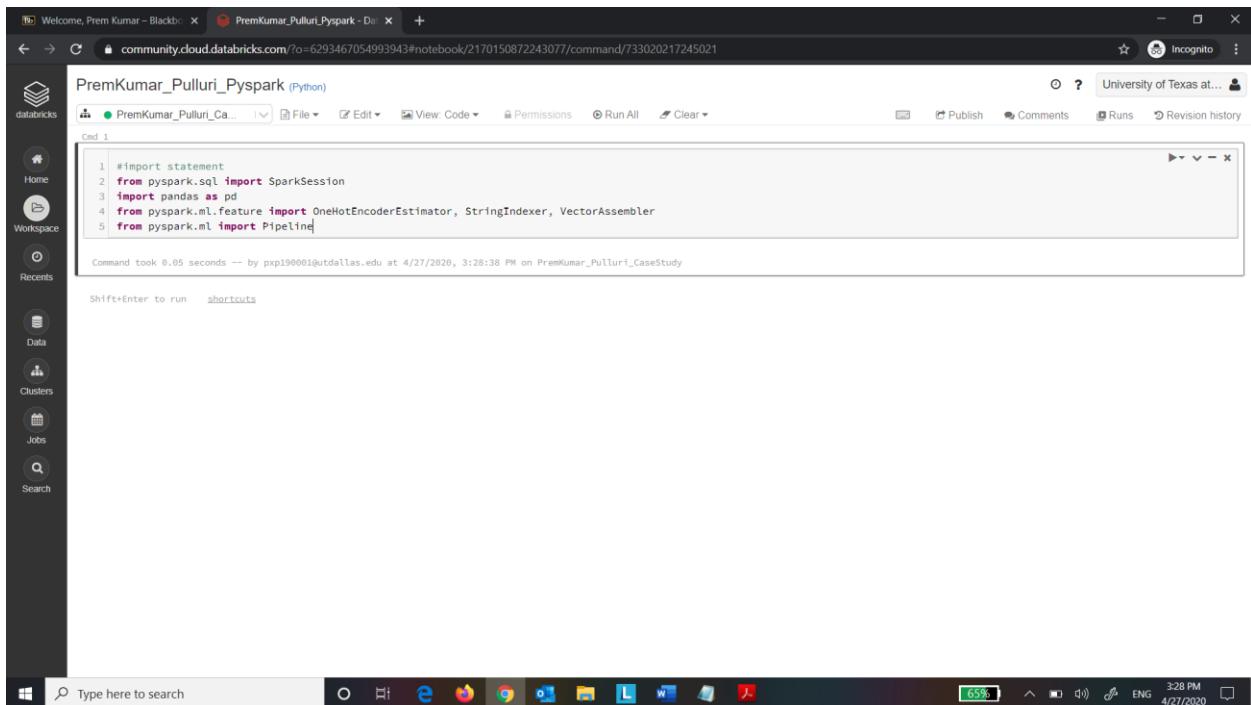
Part 4: DataFrame Operations.

Q. Take a screenshot of the blank notebook and paste it below.



Step 11: Import the required libraries from Spark.

Q. Take a screenshot of the output of the code and paste it below.



The screenshot shows a Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area has a title bar "PremKumar_Pulluri_Pyspark - Databricks" and a URL "community.cloud.databricks.com/?o=6293467054993943#notebook/2170150872243077/command/733020217245021". Below the title bar are navigation buttons (File, View, Code, Permissions, Run All, Clear), a publish button, and a revision history link. The notebook content is titled "Cmd 1" and contains the following Python code:

```
1 #import statement
2 from pyspark.sql import SparkSession
3 import pandas as pd
4 from pyspark.ml.feature import OneHotEncoderEstimator, StringIndexer, VectorAssembler
5 from pyspark.ml import Pipeline
```

A message at the bottom of the code cell says "Command took 0.05 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 3:28:38 PM on PremKumar_Pulluri_CaseStudy". A note below the code says "Shift+Enter to run shortcuts". At the bottom of the screen is a taskbar with various icons and a search bar. The system tray shows battery level (65%), network status, volume, and system time (3:28 PM, 4/27/2020).

Step 12:

Q. Take a screenshot of the output of the code and paste it below.

Here our output variable id is “deposit”. List all the input variables.

```
age: integer (nullable = true) job: string (nullable = true) marital: string  
(nullable = true) education: string (nullable = true) default: string  
(nullable = true) balance: integer (nullable = true) housing: string  
(nullable = true) loan: string (nullable = true) contact: string (nullable =  
true) day: integer (nullable = true) month: string (nullable = true)  
duration: integer (nullable = true) campaign: integer (nullable = true)  
pdays: integer (nullable = true) previous: integer (nullable = true)  
poutcome: string (nullable = true)
```

The screenshot shows a Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Recent, Data, Clusters, Jobs, and Search. The main area has a title bar "PremKumar_Pulluri_Pyspark - Databricks". Below the title bar are tabs for "File", "Edit", "View", "Permissions", "Run All", "Clear", "Publish", "Comments", "Runs", and "Revision history". The notebook content is in a "Cmd 1" cell:

```
1 #import statement
2 from pyspark.sql import SparkSession
3 import pandas as pd
4 from pyspark.ml.feature import OneHotEncoderEstimator, StringIndexer, VectorAssembler
5 from pyspark.ml import Pipeline
```

Below the cell, a message says "Command took 0.05 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 3:28:38 PM on PremKumar_Pulluri_CaseStudy". The next cell, "Cmd 2", contains:

```
1 df = spark.read.csv("/FileStore/tables/bank.csv", header = True, inferSchema = True)
2 df.printSchema()
```

The output of this command is shown in a collapsible panel:

```
root
|-- age: integer (nullable = true)
|-- job: string (nullable = true)
|-- marital: string (nullable = true)
|-- education: string (nullable = true)
|-- default: string (nullable = true)
|-- balance: integer (nullable = true)
|-- housing: string (nullable = true)
|-- loan: string (nullable = true)
|-- contact: string (nullable = true)
|-- day: integer (nullable = true)
|-- month: string (nullable = true)
|-- duration: integer (nullable = true)
|-- campaign: integer (nullable = true)
|-- pdays: integer (nullable = true)
|-- previous: integer (nullable = true)
|-- poutcome: string (nullable = true)
|-- deposit: string (nullable = true)
```

At the bottom of the screen is a taskbar with icons for File, Home, Recent, Data, Clusters, Jobs, and Search. A search bar says "Type here to search". On the right side of the taskbar are system status icons: battery level (64%), signal strength, volume, and date/time (4/27/2020 3:30 PM).

Step 13: Type the code below in the next cell and Run the cell.

Q. Take a screenshot of the output of the code and paste it below.

i. Describe “Select”, “Group by” operations in one line.

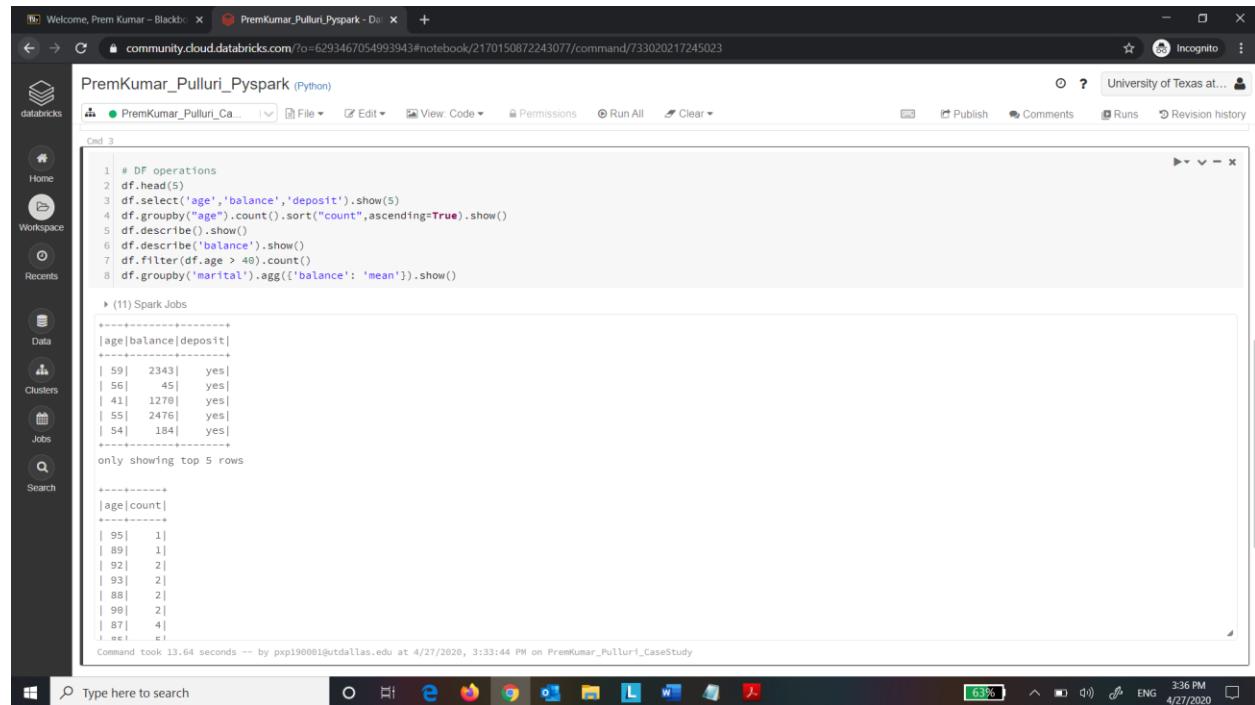
“Select” will select the mentioned columns and “Group by” will aggregate all the rows by the mentioned columns

ii. How many customers are there with an age greater than 40?

There are 4967 customers whose age is greater than 40

iii. Why are there only 5 records viewable?

Only 5 records are viewable because we have passed 5 as parameter in show function



The screenshot shows a Databricks notebook interface with the following details:

- Header:** Welcome, Prem Kumar – Blackboard, PremKumar_Pulluri_Pyspark - Databricks, community.cloud.databricks.com/?o=6293467054993943#notebook/2170150872243077/command/733020217245023, Incognito.
- Sidebar:** dataricks, Home, Workspace, Recents, Data, Clusters, Jobs, Search.
- Toolbar:** File, View, Code, Permissions, Run All, Clear, Publish, Comments, Runs, Revision history.
- Code Cell:** Cmd 3, containing Python code:

```
1 # DF operations
2 df.head(5)
3 df.select('age','balance','deposit').show(5)
4 df.groupby("age").count().sort("count",ascending=True).show()
5 df.describe().show()
6 df.describe('balance').show()
7 df.filter(df.age > 40).count()
8 df.groupby('marital').agg({'balance': 'mean'}).show()
```
- Output:** (11) Spark Jobs, showing two tables:

age	balance	deposit
59	2343	yes
56	45	yes
41	1278	yes
55	2476	yes
54	184	yes

only showing top 5 rows

age	count
95	1
89	1
92	2
93	2
88	2
90	2
87	4
- Bottom:** Type here to search, taskbar with various icons, battery level 63%, 3:36 PM, ENG, 4/27/2020.

Step 14: Execute the following code:

Q. Take a screenshot of the output of the code and paste it below.

What is seen in the output of the transpose?

The rows and columns are interchanged.

The screenshot shows a Databricks notebook interface. The left sidebar contains icons for Home, Workspace, Records, Data, Clusters, Jobs, and Search. The main area has a header bar with tabs for 'Welcome, Prem Kumar - Blackboard' and 'PremKumar_Pulluri_Pyspark - Databricks'. Below the header are buttons for File, Edit, View, Code, Permissions, Run All, Clear, Publish, Comments, Runs, and Revision history. The notebook content shows a command cell (Cmd 4) with the following code:

```
1 #Panda Dataframe
2 pd.DataFrame(df.take(5), columns=df.columns).transpose()
```

Below the code, a note indicates '(1) Spark Jobs'. The output cell (Out[10]) displays a transposed DataFrame with 5 rows and 5 columns. The columns are labeled 0, 1, 2, 3, and 4. The data consists of various categorical variables like age, job, marital, education, default, balance, housing, loan, contact, day, month, duration, campaign, pdays, previous, poutcome, deposit, and their corresponding values. The original DataFrame had 17 columns, and the transpose operation has swapped the row and column dimensions.

Command took 0.38 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 3:37:33 PM on PremKumar_Pulluri_CaseStudy

The bottom of the screen shows a Windows taskbar with the Start button, a search bar, and various pinned application icons. The system tray shows battery level (62%), network connection, volume, and date/time (4/27/2020, 3:38 PM).

Step 15: Execute the following code:

Q. Take a screenshot of the output of the code and paste it below.

The screenshot shows a Databricks notebook interface. The left sidebar contains icons for Home, Workspace, Records, Data, Clusters, Jobs, and Search. The main area displays a Python code cell and its output. The code cell contains:1 numeric_features = [t[0] for t in df.dtypes if t[1] == 'int']
2 df.select(numeric_features).describe().toPandas().transpose()

```
Out[11]:
```

	0	1	2	3	4
summary	count	mean	stddev	min	max
age	11162	41.231947679827304	11.913369192215518	18	95
balance	11162	1528.5385235620856	3225.413325946149	-6847	81204
day	11162	15.658036194230425	8.420739541006462	1	31
duration	11162	371.9938183121043	347.12838571630687	2	3881
campaign	11162	2.508421429851281	2.7220771816614824	1	63
pdays	11162	51.33040673714388	108.75826197197717	-1	854
previous	11162	0.832568894463358	2.292007218670508	0	58

Command took 1.75 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 3:40:40 PM on PremKumar_Pulluri_CaseStudy

Shift+Enter to run shortcuts

The status bar at the bottom shows: 62%, ENG, 3:41 PM, 4/27/2020.

Part 5: Data Preprocessing:

Step 16: Execute the following code:

Q. Take a screenshot of the output of the code and paste it below.

The screenshot shows a Databricks notebook interface. The notebook is titled "PremKumar_Pulluri_Pyspark (Python)". The code cell contains the following Python code:

```
1 categoricalColumns = ['job', 'marital', 'education', 'default', 'housing', 'loan', 'contact', 'poutcome']
2 stages = []
3 for categoricalCol in categoricalColumns:
4     stringIndexer = StringIndexer(inputCol = categoricalCol, outputCol = categoricalCol + 'Index')
5     encoder = OneHotEncoderEstimator(inputCols=[stringIndexer.getOutputCol()], outputCols=[categoricalCol + "classVec"])
6     stages += [stringIndexer, encoder]
7 labelStringIdx = StringIndexer(inputCol = 'deposit', outputCol = 'label')
8 stages += [labelStringIdx]
9 numericCols = ["age", "balance", "duration", "campaign", 'pdays', 'previous']
10 assemblerInputs = [c + "classVec" for c in categoricalColumns] + numericCols
11 assembler = VectorAssembler(inputCols=assemblerInputs, outputCol="features")
12 stages += [assembler]
```

Below the code cell, a message indicates: "Command took 0.30 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 3:47:16 PM on PremKumar_Pulluri_CaseStudy".

The Databricks sidebar on the left includes links for Home, Workspace, Records, Data, Clusters, Jobs, and Search. The Windows taskbar at the bottom shows various application icons and the system tray.

Step 17: Execute the following code:

```
pipeline = Pipeline(stages = stages)
pipelineModel = pipeline.fit(df)
df = pipelineModel.transform(df)
df.printSchema()
```

Q. Take a screenshot of the output of the code and paste it below.

The screenshot shows a Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Data, Clusters, Jobs, and Search. The main area has a title bar "PremKumar_Pulluri_Pyspark - Databricks" and a URL "community.cloud.databricks.com/o=6293467054993943#notebook/2170150872243077/command/733020217245028". Below the title bar are navigation buttons and tabs for Permissions, Run All, Clear, Publish, Comments, Runs, and Revision history. The central workspace contains a code editor with the following Python code:

```
1 pipeline = Pipeline(stages = stages)
2 pipelineModel = pipeline.fit(df)
3 df = pipelineModel.transform(df)
4 df.printSchema()
```

Below the code editor, the notebook displays the output of the command:

```
1 (9) Spark Jobs
2 df: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 33 more fields]
3
4 root
5 |-- age: integer (nullable = true)
6 |-- job: string (nullable = true)
7 |-- marital: string (nullable = true)
8 |-- education: string (nullable = true)
9 |-- default: string (nullable = true)
10 |-- balance: integer (nullable = true)
11 |-- housing: string (nullable = true)
12 |-- loan: string (nullable = true)
13 |-- contact: string (nullable = true)
14 |-- day: integer (nullable = true)
15 |-- month: string (nullable = true)
16 |-- duration: integer (nullable = true)
17 |-- campaign: integer (nullable = true)
18 |-- pdays: integer (nullable = true)
19 |-- previous: integer (nullable = true)
20 |-- poutcome: string (nullable = true)
21 |-- deposit: string (nullable = true)
22 |-- jobIndex: double (nullable = false)
23 |-- jobClassVec: vector (nullable = true)
24 |-- maritalIndex: double (nullable = false)
25 |-- maritalClassVec: vector (nullable = true)
```

At the bottom of the notebook, a message indicates the command took 5.64 seconds. The status bar at the bottom right shows battery level (58%), signal strength, and system information (3:56 PM, ENG, 4/27/2020).

Step 18: Execute the following command:

```
pd.DataFrame(df.take(5), columns=df.columns).transpose()
```

Q. Take a screenshot of the output of the code and paste it below.

Define the purpose of the transpose function?

The purpose of the transpose function is to swap the rows and columns

The screenshot shows a Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area has a title bar "PremKumar_Pulluri_Pyspark (Python)" and a toolbar with "File", "View", "Code", "Permissions", "Run All", "Clear", "Publish", "Comments", "Runs", and "Revision history". A code cell at the top contains the command: `pd.DataFrame(df.take(5), columns=df.columns).transpose()`. Below it, under "Out[14]:", is a table with 5 rows of data. The columns are labeled 0, 1, 2, 3, and 4. The table includes columns for age, job, marital, education, default, balance, housing, loan, contact, day, month, duration, campaign, pdays, previous, poutcome, deposit, jobIndex, jobclassVec, maritalIndex, and maritalclassVec. The data shows various categorical values like "admin.", "married", "secondary", etc., and numerical values like 59, 5, 1042, etc.

	0	1	2	3	4
age	59	56	41	55	54
job	admin.	admin.	technician	services	admin.
marital	married	married	married	married	married
education	secondary	secondary	secondary	secondary	tertiary
default	no	no	no	no	no
balance	2343	45	1270	2476	184
housing	yes	no	yes	yes	no
loan	no	no	no	no	no
contact	unknown	unknown	unknown	unknown	unknown
day	5	5	5	5	5
month	may	may	may	may	may
duration	1042	1467	1389	579	673
campaign	1	1	1	1	2
pdays	-1	-1	-1	-1	-1
previous	0	0	0	0	0
poutcome	unknown	unknown	unknown	unknown	unknown
deposit	yes	yes	yes	yes	yes
jobIndex	3	3	2	4	3
jobclassVec	(0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...)	(0, 0, 0, 0, 1, 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, 1, 0, 0, 0, 0, 0, 0, 0, ...)	(0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...)
maritalIndex	0	0	0	0	0
maritalclassVec	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)

Step 19: Execute the following code:

```
train, test = df.randomSplit([0.8, 0.2], seed = 99999)
from pyspark.ml.clustering import KMeans
import numpy as np
cost = np.zeros(10)
for k in range(2,10):
    kmeans = KMeans().setK(k).setSeed(1)
    model = kmeans.fit(train)
    cost[k] = model.computeCost(train)
```

Q. Take a screenshot of the output of the code and paste it below.

The screenshot shows a Databricks notebook titled "PremKumar_Pulluri_Pyspark (Python)". The notebook contains the following Python code:

```
1 train, test = df.randomSplit([0.8, 0.2], seed = 99999)
2 from pyspark.ml.clustering import KMeans
3 import numpy as np
4 cost = np.zeros(10)
5 for k in range(2,10):
6     kmeans = KMeans().setK(k).setSeed(1)
7     model = kmeans.fit(train)
8     cost[k] = model.computeCost(train)

(58) Spark Jobs
(58) train: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 33 more fields]
(58) test: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 33 more fields]

Command took 1.24 minutes -- by pxp190001@utdallas.edu at 4/27/2020, 4:12:52 PM on PremKumar_Pulluri_CaseStudy
```

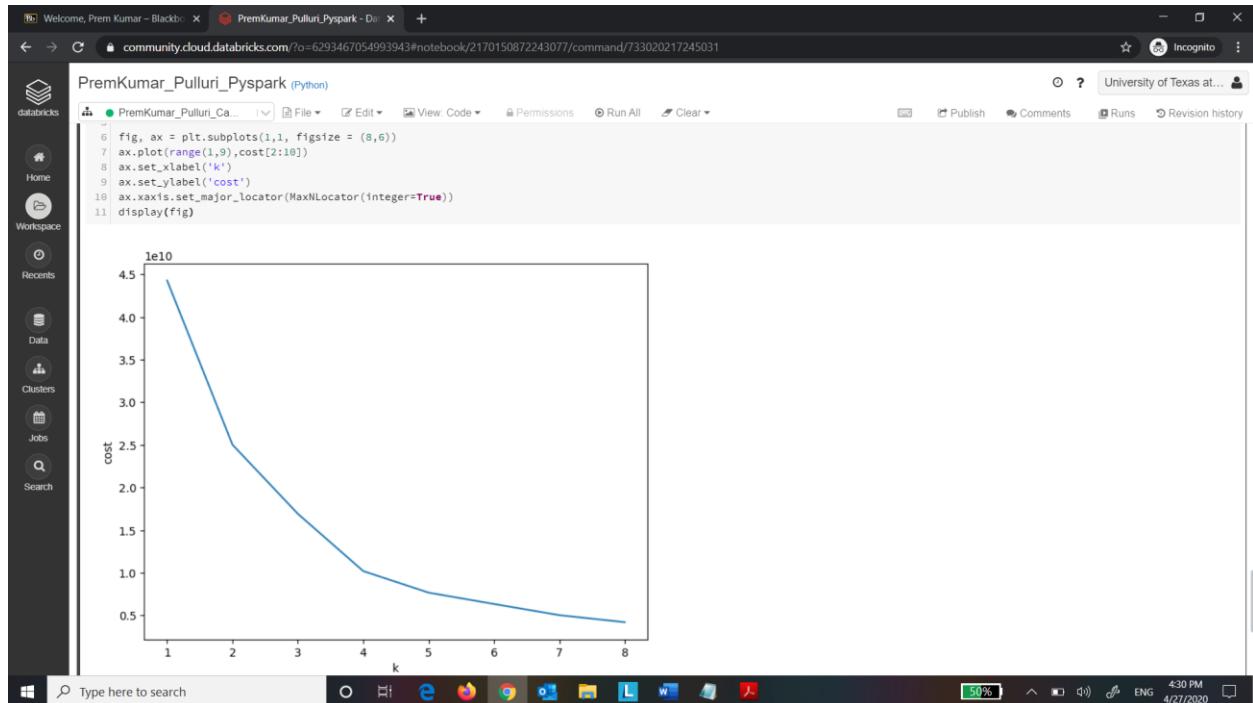
The status bar at the bottom indicates the command took 1.24 minutes and was run by pxp190001@utdallas.edu at 4/27/2020, 4:12:52 PM on PremKumar_Pulluri_CaseStudy.

Step 20: Execute the following code:

Q. Take a screenshot of the output of the code and paste it below.

Comment about the graph that is plotted here.

From the elbow plot, we can say that the cost decreases as number of clusters increases



Step 21: Execute the following code:

```
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
kmeans = KMeans().setK(8).setSeed(999)
model = kmeans.fit(train)
```

Q. Take a screenshot of the output of the code and paste it below.

The screenshot shows a Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Recent, Data, Clusters, Jobs, and Search. The main area has a header "PremKumar_Pulluri_Pyspark (Python)" and a toolbar with File, View, Code, Permissions, Run All, Clear, Publish, Comments, Runs, and Revision history. Below the toolbar is a command line interface (CLI) window titled "Cmd 11". It contains the following Python code:

```
1 from pyspark.ml.clustering import KMeans
2 from pyspark.ml.evaluation import ClusteringEvaluator
3 kmeans = KMeans().setK(8).setSeed(999)
4 model = kmeans.fit(train)
```

Below the code, the CLI displays the message: "Command took 1.57 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 4:20:24 PM on PremKumar_Pulluri_CaseStudy". A "Spark Jobs" section is visible below the CLI. At the bottom of the screen is a taskbar with various application icons and a system status bar showing battery level (47%), time (4:40 PM), and date (4/27/2020).

Step 22: Execute the following code:

```
evaluator = ClusteringEvaluator()
predictions = model.transform(train)
silhouette = evaluator.evaluate(predictions)
print("Training Dataset Performance = " + str(silhouette))
```

Q. Take a screenshot of the output of the code and paste it below.

PremKumar_Pulluri_Pyspark (Python)

```
1 from pyspark.ml.clustering import KMeans
2 from pyspark.ml.evaluation import ClusteringEvaluator
3 kmeans = KMeans().setK(8).setSeed(999)
4 model = kmeans.fit(train)

▶ (28) Spark Jobs
Command took 7.85 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 4:39:21 PM on PremKumar_Pulluri_CaseStudy

Cmd 12
1 evaluator = ClusteringEvaluator()
2 predictions = model.transform(train)
3 silhouette = evaluator.evaluate(predictions)
4 print("Training Dataset Performance = " + str(silhouette))

▶ (2) Spark Jobs
▶ 2 predictions: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 34 more fields]
Training dataset Performance = 0.7363543157476975

Command took 3.15 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 4:41:50 PM on PremKumar_Pulluri_CaseStudy
```

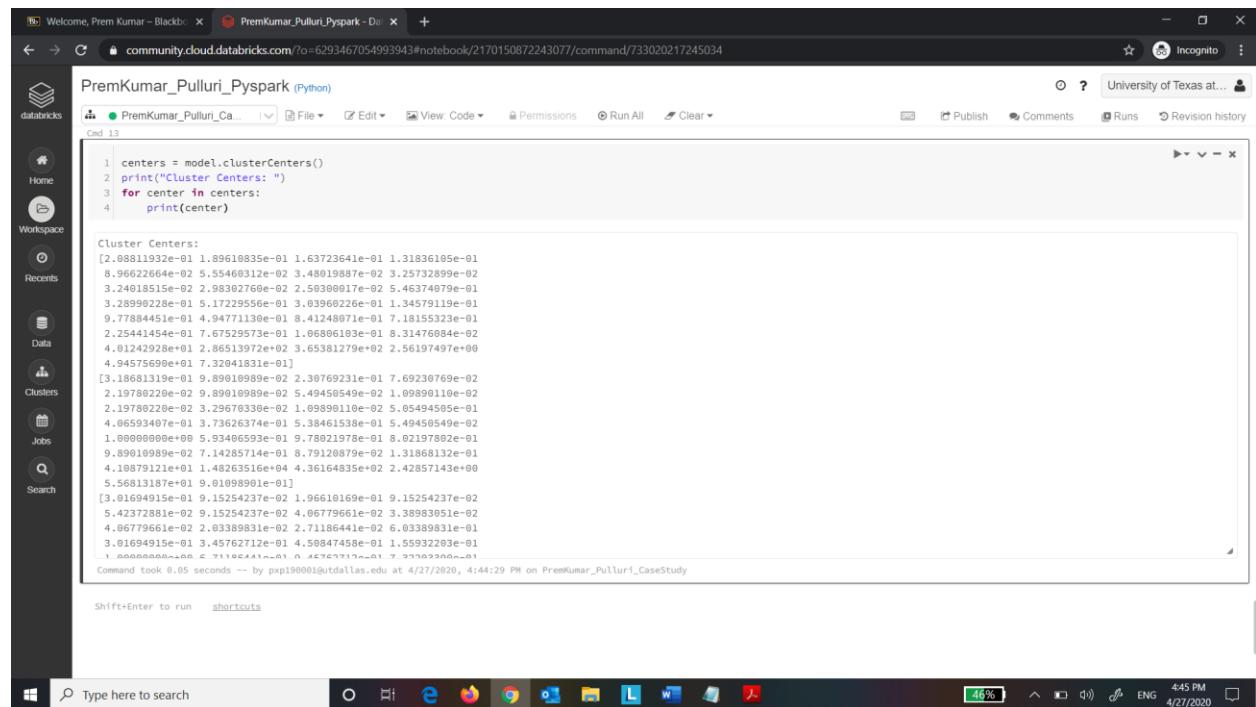
Step 23: Execute the following code:

```
centers = model.clusterCenters()
print("Cluster Centers: ")
for center in centers:
    print(center)
```

Q. Take a screenshot of the output of the code and paste it below.

What is Centroid and how is it calculated in K-means?

A centroid is a data point at the center of a cluster. Centroid is calculated by taking the mean of all the data points in a cluster



```
1 centers = model.clusterCenters()
2 print("Cluster Centers: ")
3 for center in centers:
4     print(center)

Cluster Centers:
[2.08811932e-01 1.896180835e-01 1.63723641e-01 1.31836105e-01
8.96622664e-02 5.55469312e-02 3.48019887e-02 3.25732899e-02
3.24018515e-02 2.98302760e-02 2.50300617e-02 5.46374979e-01
3.28999228e-01 5.17229556e-01 3.03966226e-01 1.34579119e-01
9.77884451e-01 4.94771130e-01 8.41248071e-01 7.18155323e-01
2.25441454e-01 7.67529573e-01 1.66806103e-01 8.31476884e-02
4.01242928e+01 2.86513972e+02 3.65381279e+02 2.56197497e+00
4.94575699e+01 7.32041831e-01
[3.18681319e-01 9.89019989e-02 2.30769231e-01 7.69230769e-02
2.19780226e-02 9.89019989e-02 5.49450549e-02 1.09890110e-02
2.19780226e-02 3.29679330e-02 1.09890110e-02 5.05494505e-01
4.06593407e-01 3.73626374e-01 5.38461538e-01 5.49450549e-02
1.00000000e+00 5.93406593e-01 9.78921978e-01 8.02197802e-01
9.89010989e-02 7.14285714e-01 8.79128879e-02 1.31868132e-01
4.18879121e+01 1.48263516e+04 4.36164835e+02 2.42857143e+00
5.56813187e+01 9.01098991e-01]
[3.01694915e-01 9.15254237e-02 1.96610169e-01 9.15254237e-02
5.42372881e-01 9.15254237e-02 4.66779561e-02 3.38983951e-02
4.66779561e-02 2.03389831e-02 2.71186441e-02 6.03389831e-01
3.01694915e-01 3.45762712e-01 4.59847458e-01 1.55932203e-01
... 0.00000000e+00 6.71192441e-01 0.45762712e-01 7.32029900e-01]
```

Command took 0.05 seconds - by pxp190001@tdallas.edu at 4/27/2020, 4:44:29 PM on PremKumar_Pulluri_CaseStudy

Step 24:

Execute the following code:

```
kmeans = KMeans().setK(8).setSeed(999)  
model = kmeans.fit(test)
```

Q. Take a screenshot of the output of the code and paste it below.

The screenshot shows a Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Records, Data, Clusters, Jobs, and Search. The main area has a header "PremKumar_Pulluri_Pyspark (Python)". Below the header is a code editor window containing the following Python code:

```
1 kmeans = KMeans().setK(8).setSeed(999)  
2 model = kmeans.fit(test)
```

Below the code editor is a "Cmd 14" section with the message "(17) Spark Jobs". At the bottom of the notebook window, it says "Command took 6.88 seconds -- by pxp190001@utdallas.edu at 4/27/2020, 4:46:29 PM on PremKumar_Pulluri_CaseStudy".

The bottom of the screen shows a Windows taskbar with various application icons and a search bar. The system tray indicates the battery is at 46%, the time is 4:46 PM, and the date is 4/27/2020.

Step 25: Execute the following code:

```
evaluator = ClusteringEvaluator()  
predictions = model.transform(test)  
silhouette = evaluator.evaluate(predictions)  
print("Testing Dataset Performance = " + str(silhouette))
```

Q. Take a screenshot of the output of the code and paste it below.

The screenshot shows a Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Data, Clusters, Jobs, and Search. The main area has a title bar 'PremKumar_Pulluri_Pyspark - Da' and 'Welcome, Prem Kumar - Black...' with tabs for 'File', 'Edit', 'View Code', 'Permissions', 'Run All', 'Clear', 'Publish', 'Comments', 'Runs', and 'Revision history'. Below the title bar is a command bar with 'Cmd 15' and a code editor containing the following Python code:

```
1 evaluator = ClusteringEvaluator()  
2 predictions = model.transform(test)  
3 silhouette = evaluator.evaluate(predictions)  
4 print("Testing Dataset Performance = " + str(silhouette))
```

After running the code, the output is displayed in the notebook:

```
(2) Spark Jobs  
predictions: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 34 more fields]  
Testing Dataset Performance = 0.6993505278355092
```

At the bottom of the notebook, there is a message: 'Command took 2.04 seconds --- by pxp190001@utdallas.edu at 4/30/2020, 6:55:35 PM on PremKumar_Pulluri_CaseStudy'. The status bar at the bottom of the screen shows 'Type here to search', a taskbar with various icons, battery level '63%', and system information '6:56 PM 4/30/2020'.

Copy the Train and Test performance results of all the clusters here. Compare the results and comment about the performance of each model. Select your best model and explain why you chose that number of clusters as the optimal level.

From the elbow plot, we can see that the cost function decreases significantly until K=4 but after that there isn't significant decrease. If K is less, it leads to overfitting and higher value of K leads to underfitting. So, 4 or 5 might be the optimal value for number of clusters K

For K=4

```
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
kmeans = KMeans().setK(4).setSeed(999)
model = kmeans.fit(train)
evaluator = ClusteringEvaluator()
predictions = model.transform(train)
silhouette = evaluator.evaluate(predictions)
print("Training Dataset Performance = " + str(silhouette))

# (25) Spark Jobs
# predictions: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 34 more fields]
Training Dataset Performance = 0.8651895328329998

Command took 9.31 seconds -- by pxpi190001@utdallas.edu at 4/30/2020, 7:02:01 PM on PremKumar_Pulluri_CaseStudy

centers = model.clusterCenters()
print("Cluster Centers: ")
for center in centers:
    print(center)

Cluster Centers:
[2.17816617e-01 1.81035607e-01 1.61992957e-01 1.23646798e-01
 8.80396504e-02 6.44319812e-02 3.59984340e-02 3.27377871e-02
 8.32594235e-02 2.88248337e-02 2.54326768e-02 5.61497326e-01
 3.20594757e-01 5.05934525e-01 3.16421025e-01 1.31994261e-01
 9.62522499e-01 5.14021130e-01 8.57571410e-01 7.21012130e-01
 2.14425468e-01 7.48271814e-01 1.12299465e-01 9.26846694e-02
 4.07520543e+00 6.71781923e+02 3.707721273e+02 2.51376827e+00
 5.21325160e+00 8.85539194e-01]
[2.72887324e-01 1.28521127e-01 1.73415493e-01 9.68309859e-02
 6.33802817e-02 1.03873239e-01 4.31338026e-02 3.16901408e-02
 2.72887324e-02 2.64084507e-02 2.64084507e-02 5.93309859e-01
 2.87852113e-01 4.06690141e-01 4.03169014e-01 1.41725352e-01]
```

PremKumar_Pulluri_Pyspark - Do | Welcome, Prem Kumar - Blackboard | +

community.cloud.databricks.com/?o=6293467054993943#notebook/2170150872243077/command/3067199472814659

Incognito

PremKumar_Pulluri_Pyspark (Python)

PremKumar_Pulluri_CaseStudy

File Edit View Code Permissions Run All Clear

University of Texas at...

Cluster Centers:

```
[2.17816617e-01 1.81035607e-01 1.61992957e-01 1.23646798e-01  
8.80396594e-02 6.44319812e-02 3.59894349e-02 3.27377707e-02  
3.32594235e-02 2.88248337e-02 2.54336768e-02 5.61497326e-01  
3.28594737e-01 5.05934525e-01 3.16421825e-01 1.31994261e-01  
9.85274999e-01 5.14921136e-01 8.57571416e-01 7.21812136e-01  
1.44254668e-01 7.48271814e-01 1.12994656e-01 9.26046694e-02  
4.07520543e+01 6.71781923e+02 3.76772127e+02 2.51376627e+00  
5.21325160e+01 8.65530194e-01]  
[2.72887324e-01 1.28521127e-01 1.73415493e-01 9.68309859e-02  
6.33982817e-02 1.63873239e-01 4.313386928e-02 3.16901408e-02  
2.72887324e-02 2.64684507e-02 2.64684597e-02 5.93399859e-01  
2.87852113e-01 4.06699141e-01 4.93169914e-01 1.41725352e-01  
9.99119718e-01 5.88028169e-01 9.37509080e-01 7.16549296e-01  
1.98140845e-01 7.11267606e-01 1.13556333e-01 1.26760563e-01  
4.41346831e+01 5.49795176e+03 4.00888282e+02 2.42957744e+00  
5.25052817e+01 9.79753521e-01]  
[3.49056694e-01 9.43396226e-02 2.07547170e-01 5.66937736e-02  
2.83018858e-02 1.13207547e-01 4.71698113e-02 1.88679245e-02  
1.88679245e-02 2.83018868e-02 1.88679245e-02 6.03773558e-01  
3.58490566e-01 3.49056694e-01 5.47169811e-01 6.69377358e-02  
1.00000000e+00 c 77241500e-01 0.43764111e-01 0.30764717e-01]
```

Command took 8.03 seconds -- by pxp190001@utdallas.edu at 4/30/2020, 7:02:13 PM on PremKumar_Pulluri_CaseStudy

Cmd 18

```
1 evaluator = ClusteringEvaluator()  
2 predictions = model.transform(test)  
3 silhouette = evaluator.evaluate(predictions)  
4 print("Testing Dataset Performance = " + str(silhouette))
```

▶ (2) Spark Jobs

▶ □ pending: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 34 more fields]

Testing Dataset Performance = 0.869985892233181

Command took 1.94 seconds -- by pxp190001@utdallas.edu at 4/30/2020, 7:02:20 PM on PremKumar_Pulluri_CaseStudy

For K=5

PremKumar_Pulluri_Pyspark - D1 Welcome, Prem Kumar - Blackboard

community.cloud.databricks.com/?o=6293467054993943#notebook/2170150872243077/command/3067199472814659

PremKumar_Pulluri_Pyspark (Python)

File Edit View Code Permissions Run All Clear

University of Texas at...

Incognito

Workspace Home Workspaces Recents Data Clusters Jobs Search

Cmd 15

```
1 from pyspark.ml.clustering import KMeans
2 from pyspark.ml.evaluation import ClusteringEvaluator
3 kmeans = KMeans().setK(5).setSeed(999)
4 model = kmeans.fit(train)
5 evaluator = ClusteringEvaluator()
6 predictions = model.transform(train)
7 silhouette = evaluator.evaluate(predictions)
8 print("Training Dataset Performance = " + str(silhouette))
```

(21) Spark jobs

predictions: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 34 more fields]

Training Dataset Performance = 0.820849454513673

Command took 8.63 seconds -- by pxp198001@tdallas.edu at 4/30/2028, 7:03:52 PM on PremKumar_Pulluri_CaseStudy

Cmd 17

```
1 centers = model.clusterCenters()
2 print("Cluster Centers: ")
3 for center in centers:
4     print(center)
```

Cluster Centers:

```
[2.14084376e-01 1.84794471e-01 1.62071481e-01 1.28519329e-01
 9.04449307e-02 5.96644785e-02 3.54485777e-02 3.25309993e-02
 3.28227571e-02 2.80807527e-02 2.59011743e-02 5.54648140e-01
 3.26914616e-01 5.13347921e-01 3.11159737e-01 1.32312181e-01
 9.88598109e-01 5.006565455e-01 8.49898059e-01 7.16411370e-01
 2.22173596e-01 7.56819840e-01 1.07958401e-01 8.94237783e-02
 4.03951860e+00 4.61864478e+02 3.70385616e+02 2.54633552e+00
 5.11264776e+01 7.7272642e+01]
[3.81818182e-01 5.45454545e-01 4.545454545e-01 5.4545454545e-02
 6.3636364e-02 1.63636364e-01 5.4545454545e-02 1.81818182e+02
 1.81818182e+02 1.81818182e+02 1.81818182e+02 6.545454545e-01
 3.27272727e-01 3.45454545e-01 5.4545454545e-01 5.4545454545e-02]
```

```

Cluster Centers:
[2.14004376e-01 1.84974471e-01 1.62871481e-01 1.28519329e-01
9.04449307e-02 5.96544785e-02 3.54485777e-02 3.25309993e-02
3.28227571e-02 2.88087527e-02 2.58911743e-02 5.54948140e-02
3.26914661e-01 5.13347921e-01 3.11159737e-01 1.32312181e-01
9.88598184e-01 5.08656455e-01 8.49899591e-01 7.16411379e-01
2.22173596e-01 7.56819840e-01 1.67959401e-01 8.94237783e-02
4.03951860e+00 4.61864478e-02 3.70305616e-02 2.54035525e+00
5.11264770e+01 7.72728642e-01]
[3.81818182e-01 5.45454545e-02 1.45454545e-01 5.45454545e-02
3.63636364e-02 1.63636364e-01 5.45454545e-02 1.81818182e-02
1.81818182e-02 1.81818182e-02 1.81818182e-02 6.54545455e-01
3.27272727e-01 3.45454545e-01 5.45454545e-01 5.45454545e-02
1.00000000e+00 6.98989891e-01 9.63636364e-01 8.00000000e-01
1.27272727e-01 8.00000000e-01 5.45454545e-02 7.27272727e-02
4.48545455e+01 2.32069818e+04 3.99945455e+02 2.96363636e+00
3.25099999e+01 6.72727273e-01]
[3.09333333e-01 9.60000000e-02 2.08000000e-01 9.06666667e-02
5.33333333e-02 8.53333333e-02 3.73333333e-02 2.93333333e-02
3.46666667e-02 2.40000000e-02 2.40000000e-02 5.73333333e-01
3.22666667e-01 3.54666667e-01 4.66666667e-01 1.36000000e-01
1.00000000e+00 2.33333333e-01 1.66666667e-01 3.43333333e-01]

Command took 0.63 seconds -- by pxp190001@utdallas.edu at 4/30/2020, 7:04:04 PM on PremKumar_Pulluri_CaseStudy

```

Cmd 18

```

1 evaluator = ClusteringEvaluator()
2 predictions = model.transform(test)
3 silhouette = evaluator.evaluate(predictions)
4 print("Testing Dataset Performance = " + str(silhouette))

▶ (2) Spark Jobs
▶ predictions: pyspark.sql.dataframe.DataFrame = [age: integer, job: string ... 34 more fields]
Testing Dataset Performance = 0.8328806996735177

Command took 2.53 seconds -- by pxp190001@utdallas.edu at 4/30/2020, 7:04:11 PM on PremKumar_Pulluri_CaseStudy

```

The train and test dataset performance is better for K = 4 compared to K = 5. Hence K = 4 is the best choice for the value of K