Hindi Vidya Prachar Samiti's
RAMNIRANJAN JHUNJHUNWALA COLLEGE OF ARTS, SCIENCE & COMMERCE
(EMPOWERED AUTONOMUS)

Data Engineering

Name: Dheeraj Mishra

Roll No.: 712

Class: MSc Data Science and Artificial Intelligence Part-I

Ramniranjan Jhunjhunwala College of Arts, Science and Commerce

Department of Data Science and Artificial Intelligence

CERTIFICATE

This is to certify Dheeraj Mishra of MSc. Data Science and Artificial Intelligence, roll no. 712 has successfully completed the practical of DATA ENGINEERING during the Academic Year 2023-2024.

Date:

Prof. Dr. Neha Ansari
(Prof-in-charge)

External Examiner

INDEX

Prac. No	Practical Name	Date	Signature
1.	Demonstrate Web Scrapping		
2.	Demonstrate XML Parsing		
3.	Create ML Pipeline using Python libraries		
4.	Install & Demonstrate the working of Kafka		
5.	Using PySpark, Demonstrate data manipulation on RDD (Resilient Distributed Dataset)		
6.	Demonstrate Data Exploration using PySpark		
7.	a) Deploy ML Pipeline for classification using Azure ML		
	b) Deploy ML Pipeline for regression using Azure ML		
	c) Deploy ML Pipeline for clustering using Azure ML		
8.	Working with different HBase commands to handle column oriented NoSQL HBase Db		

PRACTICAL NO.: 01

Course Code: RJOECDSA121 **Date:**

AIM: Demonstrate Web Scrapping

```
import requests # help to requests the website
from bs4 import BeautifulSoup # tool for webscrapping
import pandas as pd
url = 'https://www.espncricinfo.com/records/highest-career-batting-average-282910'
source=requests.get(url) #.get(url) fetch source content
print(f"Status Code : {source.status_code}") #to know the status code
#200 accessible, 404 not found
```

```
Status Code : 200
```

```
soup = BeautifulSoup(source.content, features='html.parser')
#source.content contains content to parse
soup
```

```
<!DOCTYPE html>
<html lang="en"><head><meta charset="utf-8"/><link as="font" crossorigin="anonymous" href="https://wassets.hscicdn.com/static/fonts/CiIcons/ci-icons-v2.22/fonts/icomoon.woff?gencn3" rel="preload" type="font/woff2"/><link as="font" crossorigin="anonymous"
href="https://wassets.hscicdn.com/static/fonts/BentonSans/BentonSans-Bold/BentonSans-Bold.woff" rel="preload" type="font/woff"/><link as="font"
crossorigin="anonymous" href="https://wassets.hscicdn.com/static/fonts/BentonSans/BentonSans-Regular/BentonSans-Regular.woff" rel="preload"
type="font/woff"/><link as="font" crossorigin="anonymous" href="https://wassets.hscicdn.com/static/fonts/BentonSans/BentonSans-Medium/BentonSans-Medium.woff" rel="preload" type="font/woff"/><link crossorigin="anonymous" href="https://wassets.hscicdn.com" rel="preconnect"/><link
crossorigin="anonymous" href="https://img1.hscicdn.com" rel="preconnect"/><script type="text/javascript">
        /*! js-cookie v3.0.5 | MIT */
        !function(e,t){"object"==typeof exports&&"undefined"!=typeof module?module.exports=t():"function"==typeof define&&define.amd?define(t):
(e="undefined"!=typeof globalThis?globalThis:e||self,function(){var n=e.Cookies,o=e.Cookies=t();o.noConflict=function(){return e.Cookies=n,o}}())}
(this,(function(){"use strict";function e(e){for(var t=1;t<arguments.length;t++){var n=arguments[t];for(var o in n)e[o]=n[o]}return e}var t=functio
t(n,o){function r(t,r,i){if("undefined"!=typeof document){"number"==typeof(i=e({},o,i)).expires&&(i.expires=new
Date(Date.now()+864e5*i.expires)),i.expires&&(i.expires=i.expires.toUTCString()),t=encodeURIComponent(t).replace(/%
(2[346B]|5E|60|7C)/g,decodeURIComponent).replace(/[()]/g,escape);var c="";for(var u in i)i[u]&&(c+="; "+u,!0!==i[u]&&(c+="="+i[u].split(";")
[0]));return document.cookie=t+"="+n.write(r,t)+c}}return Object.create({set:r,get:function(e){if("undefined"!=typeof document&&
(!arguments.length||e)){for(var t=document.cookie?document.cookie.split("; "):[],o={},r=0;r<t.length;r++){var
i=t[r].split("="),c=i.slice(1).join("=");try{var u=decodeURIComponent(i[0]);if(o[u]=n.read(c,u),e===u)break}catch(e){}}return e?
o[e]:o}},remove:function(t,n){r(t,"",e({},n,{expires:-1}))},withAttributes:function(n){return
t(this.converter,e({},this.attributes,n))},withConverter:function(n){return t(e({},this.converter,n),this.attributes)}},{attributes:
{value:Object.freeze(o)},converter:{value:Object.freeze(n)}})}({read:function(e){return'"'===e[0]&&(e=e.slice(1,-1)),e.replace(/(%[dA-F]
{2})+/gi,decodeURIComponent)},write:function(e){return encodeURIComponent(e).replace(/%(2[346BF]|3[AC-
F]|40|5[BDE]|60|7[BCD])/g,decodeURIComponent)}},{path:"/"});return t}));
      </script><script type="text/javascript">
```

```
print(soup.prettify())
#prettify shows the identation tag fetched from html.parser
```

```
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/framework-9f4513c321d3852d6c6f.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/main-b5b3cd09702046af1c78.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/pages/_app-66d110f64aa0f841fe91.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/5510-e03ccb96ba09e45e695a.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/6514-336dee1dfe1016c2ca93.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/9173-8a67afd75c813f8f4ea8.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/3355-0725df97cd512e623692.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/770-54f348a3c367381ff10d.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/5778-4ea7588a0715661335f05.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/chunks/pages/records/CiRecordFormatResultsPage-d04e60eaaa36e5f2863b.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/wdFYxSbA1Ws1-rCfByTIx/_buildManifest.js">
    </script>
    <script defer="" src="https://wassets.hscicdn.com/_next/static/wdFYxSbA1Ws1-rCfByTIx/_ssgManifest.js">
    </script>
```

```
soup.body()
```

```
[<div id="__next"><script>
     (function() {
       let theme = window?.localStorage?.getItem('ci-theme-preference') || 'LIGHT';

       // check ci-app-theme cookie required for native app theme support
       const appTheme = Cookies.get('ci_app_theme') || Cookies.get('ci-app-theme');

       if(appTheme) {
         theme = appTheme;
         window.hsciapp = { theme: appTheme };
       }

       if (theme === 'DARK') {
         document.body.dataset.colorTheme = 'dark';
       }

     })()
```

```
table1 = soup.find('table')
print(table1)
```

```
<table class="ds-w-full ds-table ds-table-xs ds-table-auto ds-w-full ds-overflow-scroll ds-scrollbar-hide"><thead class="ds-bg-fill-content-alternate
```

```
columns=[]
table_header = table1.find('thead')
table_header
```

```
<thead class="ds-bg-fill-content-alternate ds-text-left"><tr class=""><td class="ds-min-w-max"><div class="ds-popper-wrapper"><span class="ds-cursor-
pointer">Player</span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">Span</span>
</div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">Mat</span></div></td><td class="ds-
min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">Inns</span></div></td><td class="ds-min-w-max ds-text-right">
<div class="ds-popper-wrapper"><span class="ds-cursor-pointer">NO</span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-
wrapper"><span class="ds-cursor-pointer">Runs</span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-
cursor-pointer">HS</span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">
<strong>Ave</strong></span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">BF</span>
</div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">SR</span></div></td><td class="ds-
min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">100</span></div></td><td class="ds-min-w-max ds-text-right">
<div class="ds-popper-wrapper"><span class="ds-cursor-pointer">50</span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-
wrapper"><span class="ds-cursor-pointer">0</span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-
cursor-pointer">4s</span></div></td><td class="ds-min-w-max ds-text-right"><div class="ds-popper-wrapper"><span class="ds-cursor-pointer">6s</span>
</div></td></tr></thead>
```

```
for span in table_header.find_all('span'): #column names are present in the span
tag
    columns.append(span.text)
print(columns)
```

```
['Player', 'Span', 'Mat', 'Inns', 'NO', 'Runs', 'HS', 'Ave', 'BF', 'SR', '100', '50', '0', '4s', '6s']
```

```
table_body = table1.find('tbody')
table_rows = table_body.find_all('tr')
data = []
for table1_row in table_rows:
    raw_data=[]
    for row_data in table1_row.find_all('td'):
        raw_data.append(row_data.text)
    data.append(raw_data)
data
```

```
[['DG Bradman (AUS)',
  '1928-1948',
  '52',
  '80',
  '10',
  '6996',
  '334',
  '99.94',
  '9800+',
  '58.60',
  '29',
  '13',
  '7',
  '626+',
  '6'],
 ['HC Brook (ENG)',
  '2022-2023',
  '12',
  '20',
  '1',
  '1181',
  '186',
  '62.15',
  '1287',
  '91.76',
  '4',
  '7',
  '1',
  '141',
  '23'],
```

```
DataF = pd.DataFrame(data)
DataF
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	DG Bradman (AUS)	1928-1948	52	80	10	6996	334	99.94	9800+	58.60	29	13	7	626+	6
1	HC Brook (ENG)	2022-2023	12	20	1	1181	186	62.15	1287	91.76	4	7	1	141	23
2	AC Voges (AUS)	2015-2016	20	31	7	1485	269*	61.87	2667	55.68	5	4	2	186	5
3	RG Pollock (SA)	1963-1970	23	41	4	2256	274	60.97	1707+	54.48	7	11	1	246+	11
4	GA Headley (WI)	1930-1954	22	40	4	2190	270*	60.83	416+	56.00	10	5	2	104+	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
58	KD Walters (AUS)	1965-1981	74	125	14	5357	250	48.26	8662+	49.16	15	33	4	525+	23
59	GC Smith (ICC/SA)	2002-2014	117	205	13	9265	277	48.25	15525	59.67	27	38	11	1165	24
60	WH Ponsford (AUS)	1924-1934	29	48	4	2122	266	48.22	3118+	44.77	7	6	1	119+	0
61	SJ McCabe (AUS)	1930-1938	39	62	5	2748	232	48.21	3217+	60.02	6	13	4	241+	5+
62	DR Jardine (ENG)	1928-1934	22	33	6	1296	127	48.00	2110+	25.59	1	10	2	53+	0

63 rows × 15 columns

```
DataF.to_csv('average_data',index=False) #this help to change the Dataframe into CSV file
```

PRACTICAL NO.: 02

Course Code: RJOECDSA121 **Date:**

AIM: Demonstrate XML Parsing

```
import pandas as pd
import xml.etree.ElementTree as et # used to parse and create XML documents
xtree=et.parse(r"C:\Users\User20\Desktop\sample_xml.xml")
xtree
```

```
<xml.etree.ElementTree.ElementTree at 0x1a92673c940>
```

```
xroot=xtree.getroot()
xroot.tag #to get root tag
```

```
'data'
```

```
rows=[]
for node in xroot:
    sname=node.attrib.get("name") if node is not None else None
    smail=node.find("email").text if node is not None else None
    sroll=node.find("roll_no").text if node is not None else None
    rows.append({"name":sname,"mail":smail,"roll_no":sroll})
output_df=pd.DataFrame(data=rows)
output_df
```

	name	mail	roll_no
0	John	john@gmail.com	1
1	Pratap	pratap@gmail.com	2
2	Lydia	lydia@gmail.com	3
3	Subash	subash@gmail.com	4
4	Carter	carter@gmail.com	5

```
output df.to csv("sample") #changes the dataframe into .csv file
```

PRACTICAL NO.: 03

Course Code: RJOECDSA121 **Date:**

AIM: Create ML Pipeline using Python libraries

```
!pip install pyspark #installing spark, using for distributed data processing
```

```
Requirement already satisfied: pyspark in c:\users\user20\anaconda3\lib\site-packages (3.5.0)
Requirement already satisfied: py4j==0.10.9.7 in c:\users\user20\anaconda3\lib\site-packages (from pyspark) (0.10.9.7)
```

#importing required library

```
import pyspark
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import score
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
```

```
from sklearn.datasets import load_breast_cancer #importing inbuilt dataset
dataset=load_breast_cancer()
df = pd.DataFrame(dataset.data, columns = dataset.feature_names) #dataset.data = input features
df.sample(3)#displaying random 3 datas from dataframe
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	worst area
541	14.47	24.99	95.81	656.4	0.08837	0.12300	0.10090	0.03890	0.1872	0.06341	...	16.22	31.73	113.50	808.9
5	12.45	15.70	82.57	477.1	0.12780	0.17000	0.15780	0.08089	0.2087	0.07613	...	15.47	23.75	103.40	741.6
475	12.83	15.73	82.89	506.9	0.09040	0.08269	0.05835	0.03078	0.1705	0.05913	...	14.09	19.35	93.22	605.8

3 rows × 30 columns

```
df['class'] = dataset.target #giving name to target column
df.sample(3)
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness
80	11.45	20.97	73.81	401.5	0.11020	0.09362	0.04591	0.02233	0.1842	0.07005	...	32.16	84.53	525.1	0.1557
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622
390	10.26	12.22	65.75	321.6	0.09996	0.07542	0.01923	0.01968	0.1800	0.06569	...	15.65	73.23	394.5	0.1343

3 rows × 31 columns

```
df.info() #getting to know about the info
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   mean radius             569 non-null    float64
 1   mean texture            569 non-null    float64
 2   mean perimeter          569 non-null    float64
 3   mean area               569 non-null    float64
 4   mean smoothness         569 non-null    float64
 5   mean compactness        569 non-null    float64
 6   mean concavity          569 non-null    float64
 7   mean concave points     569 non-null    float64
 8   mean symmetry           569 non-null    float64
 9   mean fractal dimension  569 non-null    float64
 10  radius error            569 non-null    float64
 11  texture error           569 non-null    float64
 12  perimeter error         569 non-null    float64
 13  area error              569 non-null    float64
 14  smoothness error        569 non-null    float64
 15  compactness error       569 non-null    float64
 16  concavity error         569 non-null    float64
 17  concave points error    569 non-null    float64
 18  symmetry error          569 non-null    float64
 19  fractal dimension error 569 non-null    float64
 20  worst radius            569 non-null    float64
 21  worst texture           569 non-null    float64
 22  worst perimeter         569 non-null    float64
 23  worst area              569 non-null    float64
 24  worst smoothness        569 non-null    float64
```

```
df.shape
```

```
(569, 31)
```

```
logit_pl = Pipeline([('scaling' , StandardScaler()),
                     ('pca', PCA(n_components=3)),
                      ('model_logit' , LogisticRegression())])
dtree_pl = Pipeline([('scaling' , StandardScaler()),
                     ('pca', PCA(n_components=3)),
                      ('model_logit' , DecisionTreeClassifier())])
naive_pl =  Pipeline([('scaling' , StandardScaler()),
                     ('pca', PCA(n_components=3)),
                      ('model_logit' , GaussianNB())])
x = df.iloc[:,:-1] #storing the desired features into x variable
y = df.iloc[:,-1] #storing the target feature into y variable

xtrain,xtest,ytrain,ytest = train_test_split(x,y,random_state = 1, test_size = 0.25)
#spliting the datas into train and test
my_pipeline = [logit_pl,dtree_pl,naive_pl]
pipeline_dict = {0:'Logistic_Regression', 1:'Decision_Tree',2:'Naive_Bayes'}
for i in my_pipeline:
    i.fit(xtrain,ytrain)
```

```
for i, model in enumerate(my_pipeline): #enumerate allows you to loop over an iterable (such as a list, tuple, or string) while keeping track of the index and the value at that index. It returns tuples containing the index and corresponding item.
```

```
    print(f"{pipeline_dict[i]}'s trainig accuracy is :
{model.score(xtrain,ytrain)}")
```

```
Logistic_Regression's trainig accuracy is : 0.9671361502347418
Decision_Tree's trainig accuracy is : 1.0
Naive_Bayes's trainig accuracy is : 0.9225352112676056
```

```
for i, model in enumerate(my_pipeline):
    print(f"{pipeline_dict[i]}'s trainig accuracy is :
{model.score(xtest,ytest)}")
```

```
Logistic_Regression's trainig accuracy is : 0.9370629370629371
Decision_Tree's trainig accuracy is : 0.9440559440559441
Naive_Bayes's trainig accuracy is : 0.9090909090909091
```

PRACTICAL NO.: 05

Course Code: RJOECDSA121 **Date:**

AIM: Using PySpark, Demonstrate data manipulation on RDD (Resilient Distributed Dataset)

```
!pip install pyspark
```

```
Collecting pyspark
  Downloading pyspark-3.5.0.tar.gz (316.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 316.9/316.9 MB 3.9 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.10/dist-packages (from pyspark) (0.10.9.7)
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.5.0-py2.py3-none-any.whl size=317425345 sha256=59c23d07e7852a56d29bf78b722658a2d2caf378b870a03a7de5542
  Stored in directory: /root/.cache/pip/wheels/41/4e/10/c2cf2467f71c678cfc8a6b9ac9241e5e44a01940da8fbb17fc
Successfully built pyspark
Installing collected packages: pyspark
Successfully installed pyspark-3.5.0
```

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("DataProcessingRDD").getOrCreate()
df_iris =
spark.read.format("csv").option("mutliline",True).option("header",True).option("in
ferschema",True).load("/content/Iris.csv")
df_iris.show() #to display the dataset
```

```
+---+-------------+------------+-------------+------------+-----------+
| Id|SepalLengthCm|SepalWidthCm|PetalLengthCm|PetalWidthCm|    Species|
+---+-------------+------------+-------------+------------+-----------+
|  1|          5.1|         3.5|          1.4|         0.2|Iris-setosa|
|  2|          4.9|         3.0|          1.4|         0.2|Iris-setosa|
|  3|          4.7|         3.2|          1.3|         0.2|Iris-setosa|
|  4|          4.6|         3.1|          1.5|         0.2|Iris-setosa|
|  5|          5.0|         3.6|          1.4|         0.2|Iris-setosa|
|  6|          5.4|         3.9|          1.7|         0.4|Iris-setosa|
|  7|          4.6|         3.4|          1.4|         0.3|Iris-setosa|
|  8|          5.0|         3.4|          1.5|         0.2|Iris-setosa|
|  9|          4.4|         2.9|          1.4|         0.2|Iris-setosa|
| 10|          4.9|         3.1|          1.5|         0.1|Iris-setosa|
| 11|          5.4|         3.7|          1.5|         0.2|Iris-setosa|
| 12|          4.8|         3.4|          1.6|         0.2|Iris-setosa|
| 13|          4.8|         3.0|          1.4|         0.1|Iris-setosa|
| 14|          4.3|         3.0|          1.1|         0.1|Iris-setosa|
| 15|          5.8|         4.0|          1.2|         0.2|Iris-setosa|
| 16|          5.7|         4.4|          1.5|         0.4|Iris-setosa|
| 17|          5.4|         3.9|          1.3|         0.4|Iris-setosa|
| 18|          5.1|         3.5|          1.4|         0.3|Iris-setosa|
| 19|          5.7|         3.8|          1.7|         0.3|Iris-setosa|
| 20|          5.1|         3.8|          1.5|         0.3|Iris-setosa|
+---+-------------+------------+-------------+------------+-----------+
only showing top 20 rows
```

```
df_iris.count()
```

```
150
```

```
df_iris.summary().show()#display the 5 summary
```

```
+-------+-----------------+-----------------+-------------------+-----------------+------------------+-------------+
|summary|               Id|     SepalLengthCm|       SepalWidthCm|     PetalLengthCm|      PetalWidthCm|      Species|
+-------+-----------------+-----------------+-------------------+-----------------+------------------+-------------+
|  count|              150|              150|                150|              150|               150|          150|
|   mean|             75.5| 5.843333333333335| 3.0540000000000007|3.758666666666693|1.1986666666666672|         NULL|
| stddev|43.445367992456916|0.8280661279778637|0.43359431136217375| 1.764420419952262|0.7631607417008414|         NULL|
|    min|                1|              4.3|                2.0|              1.0|               0.1|  Iris-setosa|
|    25%|               38|              5.1|                2.8|              1.6|               0.3|         NULL|
|    50%|               75|              5.8|                3.0|              4.3|               1.3|         NULL|
|    75%|              113|              6.4|                3.3|              5.1|               1.8|         NULL|
|    max|              150|              7.9|                4.4|              6.9|               2.5|Iris-virginica|
+-------+-----------------+-----------------+-------------------+-----------------+------------------+-------------+
```

```
df_iris.select("SepalLengthCm").summary().show()
```

```
+-------+------------------+
|summary|     SepalLengthCm|
+-------+------------------+
|  count|               150|
|   mean| 5.843333333333335|
| stddev|0.8280661279778637|
|    min|               4.3|
|    25%|               5.1|
|    50%|               5.8|
|    75%|               6.4|
|    max|               7.9|
+-------+------------------+
```

```
df_iris.select('SepalLengthCm',"PetalLengthCm").show()
```

```
+-------------+-------------+
|SepalLengthCm|PetalLengthCm|
+-------------+-------------+
|          5.1|          1.4|
|          4.9|          1.4|
|          4.7|          1.3|
|          4.6|          1.5|
|          5.0|          1.4|
|          5.4|          1.7|
|          4.6|          1.4|
|          5.0|          1.5|
|          4.4|          1.4|
|          4.9|          1.5|
|          5.4|          1.5|
|          4.8|          1.6|
|          4.8|          1.4|
|          4.3|          1.1|
|          5.8|          1.2|
|          5.7|          1.5|
|          5.4|          1.3|
|          5.1|          1.4|
|          5.7|          1.7|
|          5.1|          1.5|
+-------------+-------------+
only showing top 20 rows
```

```
df_iris.createOrReplaceTempView("iristab")
df_iris1 = spark.sql("""select * from iristab where PetalLengthCm<2.5""")
df_iris1.show()
```

```
+---+-------------+------------+-------------+------------+-----------+
| Id|SepalLengthCm|SepalWidthCm|PetalLengthCm|PetalWidthCm|    Species|
+---+-------------+------------+-------------+------------+-----------+
|  1|          5.1|         3.5|          1.4|         0.2|Iris-setosa|
|  2|          4.9|         3.0|          1.4|         0.2|Iris-setosa|
|  3|          4.7|         3.2|          1.3|         0.2|Iris-setosa|
|  4|          4.6|         3.1|          1.5|         0.2|Iris-setosa|
|  5|          5.0|         3.6|          1.4|         0.2|Iris-setosa|
|  6|          5.4|         3.9|          1.7|         0.4|Iris-setosa|
|  7|          4.6|         3.4|          1.4|         0.3|Iris-setosa|
|  8|          5.0|         3.4|          1.5|         0.2|Iris-setosa|
|  9|          4.4|         2.9|          1.4|         0.2|Iris-setosa|
| 10|          4.9|         3.1|          1.5|         0.1|Iris-setosa|
| 11|          5.4|         3.7|          1.5|         0.2|Iris-setosa|
| 12|          4.8|         3.4|          1.6|         0.2|Iris-setosa|
| 13|          4.8|         3.0|          1.4|         0.1|Iris-setosa|
| 14|          4.3|         3.0|          1.1|         0.1|Iris-setosa|
| 15|          5.8|         4.0|          1.2|         0.2|Iris-setosa|
| 16|          5.7|         4.4|          1.5|         0.4|Iris-setosa|
| 17|          5.4|         3.9|          1.3|         0.4|Iris-setosa|
| 18|          5.1|         3.5|          1.4|         0.3|Iris-setosa|
| 19|          5.7|         3.8|          1.7|         0.3|Iris-setosa|
| 20|          5.1|         3.8|          1.5|         0.3|Iris-setosa|
+---+-------------+------------+-------------+------------+-----------+
only showing top 20 rows
```

```
df_iris1.count()
```

```
50
```

```
df_iris2 = spark.sql("""select * from iristab where PetalLengthCm between 2.5 and 4.5""")
df_iris2.show()
```

```
+---+------------+-----------+------------+-----------+--------------+
| Id|SepalLengthCm|SepalWidthCm|PetalLengthCm|PetalWidthCm|       Species|
+---+------------+-----------+------------+-----------+--------------+
| 52|         6.4|        3.2|         4.5|        1.5|Iris-versicolor|
| 54|         5.5|        2.3|         4.0|        1.3|Iris-versicolor|
| 56|         5.7|        2.8|         4.5|        1.3|Iris-versicolor|
| 58|         4.9|        2.4|         3.3|        1.0|Iris-versicolor|
| 60|         5.2|        2.7|         3.9|        1.4|Iris-versicolor|
| 61|         5.0|        2.0|         3.5|        1.0|Iris-versicolor|
| 62|         5.9|        3.0|         4.2|        1.5|Iris-versicolor|
| 63|         6.0|        2.2|         4.0|        1.0|Iris-versicolor|
| 65|         5.6|        2.9|         3.6|        1.3|Iris-versicolor|
| 66|         6.7|        3.1|         4.4|        1.4|Iris-versicolor|
| 67|         5.6|        3.0|         4.5|        1.5|Iris-versicolor|
| 68|         5.8|        2.7|         4.1|        1.0|Iris-versicolor|
| 69|         6.2|        2.2|         4.5|        1.5|Iris-versicolor|
| 70|         5.6|        2.5|         3.9|        1.1|Iris-versicolor|
| 72|         6.1|        2.8|         4.0|        1.3|Iris-versicolor|
| 75|         6.4|        2.9|         4.3|        1.3|Iris-versicolor|
| 76|         6.6|        3.0|         4.4|        1.4|Iris-versicolor|
| 79|         6.0|        2.9|         4.5|        1.5|Iris-versicolor|
| 80|         5.7|        2.6|         3.5|        1.0|Iris-versicolor|
| 81|         5.5|        2.4|         3.8|        1.1|Iris-versicolor|
+---+------------+-----------+------------+-----------+--------------+
only showing top 20 rows
```

```
df_iris2.count()
```

```
37
```

```
spark.sql(""" select Species, Count(*) from iristab group by Species """ ).show()
```

```
+---------------+--------+
|        Species|count(1)|
+---------------+--------+
| Iris-virginica|      50|
|    Iris-setosa|      50|
|Iris-versicolor|      50|
+---------------+--------+
```

```
spark.sql(""" select SepalLengthCm, SepalWidthCm, Species from iristab where
Species = 'Iris-setosa' """).show()
```

```
+------------+-----------+-----------+
|SepalLengthCm|SepalWidthCm|    Species|
+------------+-----------+-----------+
|         5.1|        3.5|Iris-setosa|
|         4.9|        3.0|Iris-setosa|
|         4.7|        3.2|Iris-setosa|
|         4.6|        3.1|Iris-setosa|
|         5.0|        3.6|Iris-setosa|
|         5.4|        3.9|Iris-setosa|
|         4.6|        3.4|Iris-setosa|
|         5.0|        3.4|Iris-setosa|
|         4.4|        2.9|Iris-setosa|
|         4.9|        3.1|Iris-setosa|
|         5.4|        3.7|Iris-setosa|
|         4.8|        3.4|Iris-setosa|
|         4.8|        3.0|Iris-setosa|
|         4.3|        3.0|Iris-setosa|
|         5.8|        4.0|Iris-setosa|
|         5.7|        4.4|Iris-setosa|
|         5.4|        3.9|Iris-setosa|
|         5.1|        3.5|Iris-setosa|
|         5.7|        3.8|Iris-setosa|
|         5.1|        3.8|Iris-setosa|
+------------+-----------+-----------+
only showing top 20 rows
```

```
df_iris2 = spark.sql("""Select min(SepalLengthCm) as min_sep_len_species,species
from iristab group by species""")
df_iris2.show()
```

```
+------------------+--------------+
|min_sep_len_species|       species|
+------------------+--------------+
|               4.9| Iris-virginica|
|               4.3|    Iris-setosa|
|               4.9|Iris-versicolor|
+------------------+--------------+
```

```
df_iris.groupBy("Species").min("SepalLengthCm").show()
```

```
+-----------+------------------+
|    Species|min(SepalLengthCm)|
+-----------+------------------+
|Iris-setosa|               4.3|
+-----------+------------------+
```

```
spark.sql(""" Select Species, min(PetalLengthCm) as
min_petal_len,max(PetalLengthcm) as Max_petal_length, avg(PetalLengthCm) from
iristab group by Species""").show()
#Multiple Agg Queries
```

```
+--------------+-------------+---------------+-----------------+
|       Species|min_petal_len|Max_petal_length|avg(PetalLengthCm)|
+--------------+-------------+---------------+-----------------+
| Iris-virginica|          4.5|            6.9|            5.552|
|    Iris-setosa|          1.0|            1.9|            1.464|
|Iris-versicolor|          3.0|            5.1|             4.26|
+--------------+-------------+---------------+-----------------+
```

```
spark.sql(""" select PetalLengthCm, PetalWidthCm, Species from iristab group by
Species, PetalLengthCm, PetalWidthCm having mean(PetalLengthCm)>4 """ ).show()
```

```
+-------------+------------+--------------+
|PetalLengthCm|PetalWidthCm|       Species|
+-------------+------------+--------------+
|          4.6|         1.4|Iris-versicolor|
|          5.8|         1.8| Iris-virginica|
|          5.4|         2.3| Iris-virginica|
|          5.4|         2.1| Iris-virginica|
|          5.5|         2.1| Iris-virginica|
|          6.1|         1.9| Iris-virginica|
|          6.4|         2.0| Iris-virginica|
|          5.9|         2.1| Iris-virginica|
|          5.7|         2.5| Iris-virginica|
|          4.2|         1.5|Iris-versicolor|
|          4.4|         1.3|Iris-versicolor|
|          4.6|         1.3|Iris-versicolor|
|          4.3|         1.3|Iris-versicolor|
|          4.7|         1.5|Iris-versicolor|
|          4.8|         1.4|Iris-versicolor|
|          5.0|         2.0| Iris-virginica|
|          5.3|         1.9| Iris-virginica|
|          4.1|         1.3|Iris-versicolor|
|          5.8|         2.2| Iris-virginica|
|          5.7|         2.3| Iris-virginica|
+-------------+------------+--------------+
only showing top 20 rows
```

```
df_iris.where('Species=="Iris-setosa"').show()
```

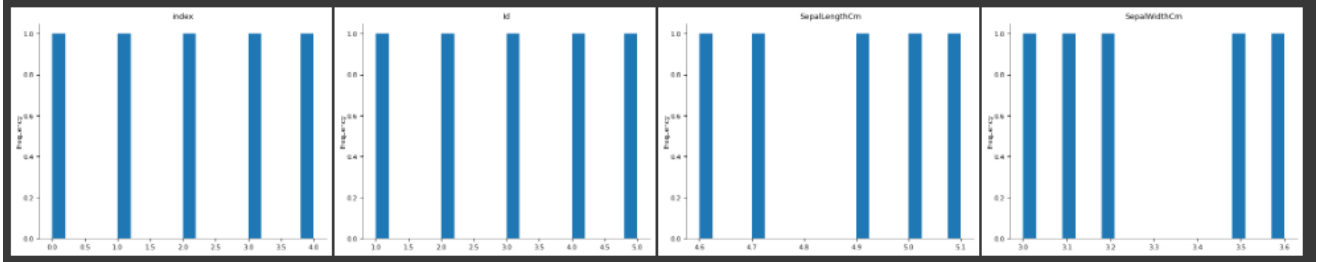
```
+---+-------------+------------+-------------+------------+-----------+
| Id|SepalLengthCm|SepalWidthCm|PetalLengthCm|PetalWidthCm|    Species|
+---+-------------+------------+-------------+------------+-----------+
|  1|          5.1|         3.5|          1.4|         0.2|Iris-setosa|
|  2|          4.9|         3.0|          1.4|         0.2|Iris-setosa|
|  3|          4.7|         3.2|          1.3|         0.2|Iris-setosa|
|  4|          4.6|         3.1|          1.5|         0.2|Iris-setosa|
|  5|          5.0|         3.6|          1.4|         0.2|Iris-setosa|
|  6|          5.4|         3.9|          1.7|         0.4|Iris-setosa|
|  7|          4.6|         3.4|          1.4|         0.3|Iris-setosa|
|  8|          5.0|         3.4|          1.5|         0.2|Iris-setosa|
|  9|          4.4|         2.9|          1.4|         0.2|Iris-setosa|
| 10|          4.9|         3.1|          1.5|         0.1|Iris-setosa|
| 11|          5.4|         3.7|          1.5|         0.2|Iris-setosa|
| 12|          4.8|         3.4|          1.6|         0.2|Iris-setosa|
| 13|          4.8|         3.0|          1.4|         0.1|Iris-setosa|
| 14|          4.3|         3.0|          1.1|         0.1|Iris-setosa|
| 15|          5.8|         4.0|          1.2|         0.2|Iris-setosa|
| 16|          5.7|         4.4|          1.5|         0.4|Iris-setosa|
| 17|          5.4|         3.9|          1.3|         0.4|Iris-setosa|
| 18|          5.1|         3.5|          1.4|         0.3|Iris-setosa|
| 19|          5.7|         3.8|          1.7|         0.3|Iris-setosa|
| 20|          5.1|         3.8|          1.5|         0.3|Iris-setosa|
+---+-------------+------------+-------------+------------+-----------+
only showing top 20 rows
```

```
df_iris2 = df_iris.withColumnRenamed("PetalLengthCm","petal_len")
df_iris2.columns
```

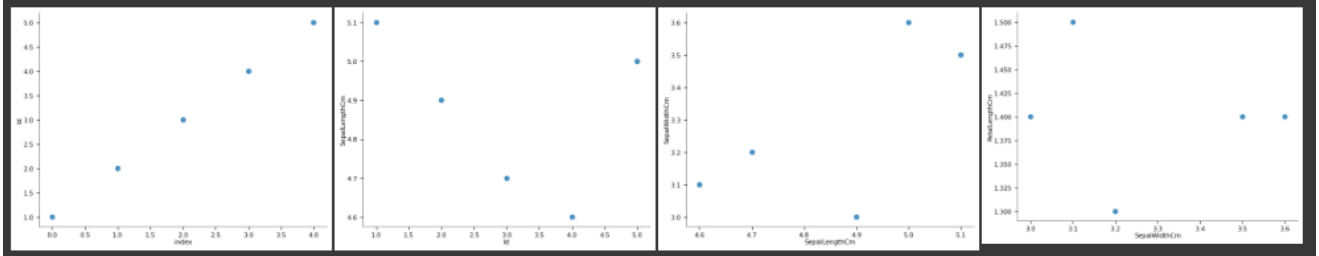
```
['Id', 'SepalLengthCm', 'SepalWidthCm', 'petal_len', 'PetalWidthCm', 'Species']
```

```
pd_iris = df_iris.toPandas()
pd_iris.head()
```

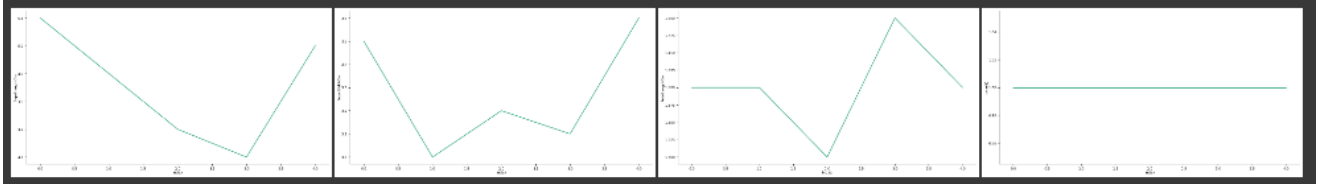
Distributions



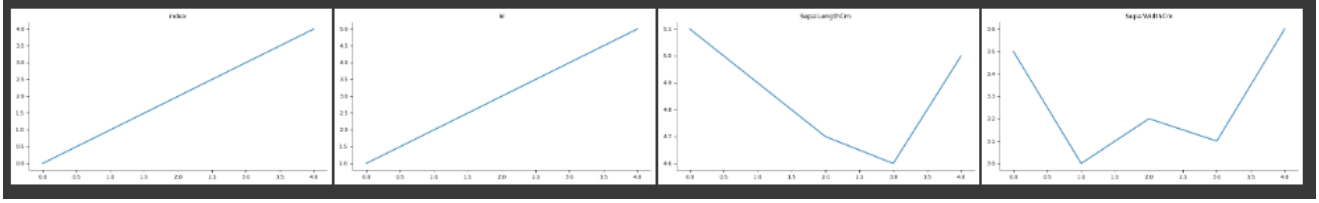
2-d distributions



Time series



Values



PRACTICAL NO.: 06

Course Code: RJOECDSA121 **Date:**

AIM: Demonstrate Data Exploration using PySpark

```
!pip install pyspark
```

```
Collecting pyspark
  Downloading pyspark-3.5.0.tar.gz (316.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 316.9/316.9 MB 2.6 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.10/dist-packages (from pyspark) (0.10.9.7)
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.5.0-py2.py3-none-any.whl size=317425345 sha256=8049b486b749afffd80dabd4db7420646b5fc29655fcbc0f10bebb97af365c5e
  Stored in directory: /root/.cache/pip/wheels/41/4e/10/c2cf2467f71c678cfc8a6b9ac9241e5e44a01940da8fbb17fc
Successfully built pyspark
Installing collected packages: pyspark
Successfully installed pyspark-3.5.0
```

```
from google.colab import drive
drive.mount('/content/drive')
from pyspark.context import SparkContext
#Find & Initialize PySpark Home Variable
sc = SparkContext("local","iris")
iris1 = sc.textFile("/Iris.csv",2) #2 partitions to be made while import
#Retrieve Records from the DataSet
iris1.collect() #will collect and divide the sample into 2
```

```
['Id,SepalLengthCm,SepalWidthCm,PetalLengthCm,PetalWidthCm,Species',
 '1,5.1,3.5,1.4,0.2,Iris-setosa',
 '2,4.9,3.0,1.4,0.2,Iris-setosa',
 '3,4.7,3.2,1.3,0.2,Iris-setosa',
 '4,4.6,3.1,1.5,0.2,Iris-setosa',
 '5,5.0,3.6,1.4,0.2,Iris-setosa',
 '6,5.4,3.9,1.7,0.4,Iris-setosa',
 '7,4.6,3.4,1.4,0.3,Iris-setosa',
 '8,5.0,3.4,1.5,0.2,Iris-setosa',
 '9,4.4,2.9,1.4,0.2,Iris-setosa',
 '10,4.9,3.1,1.5,0.1,Iris-setosa',
 '11,5.4,3.7,1.5,0.2,Iris-setosa',
 '12,4.8,3.4,1.6,0.2,Iris-setosa',
 '13,4.8,3.0,1.4,0.1,Iris-setosa',
 '14,4.3,3.0,1.1,0.1,Iris-setosa',
 '15,5.8,4.0,1.2,0.2,Iris-setosa',
 '16,5.7,4.4,1.5,0.4,Iris-setosa',
 '17,5.4,3.9,1.3,0.4,Iris-setosa',
 '18,5.1,3.5,1.4,0.3,Iris-setosa',
 '19,5.7,3.8,1.7,0.3,Iris-setosa',
 '20,5.1,3.8,1.5,0.3,Iris-setosa',
```

```
#Displaying the Number of Records in Each Partitions
partitioned_data = iris1.glom().collect() #glom will merege into list which is
partitioned to retain the partition boundary
for i, partition in enumerate(partitioned_data):
    print(f"Partiton-{i+1} has {len(partition)} records")
```

```
Partiton-1 has 79 records
Partiton-2 has 72 records
```

```
#parallelizing data!! We use this when we want to make use of local data to convert it into rdd
iris2 = sc.parallelize(iris1.collect(),8)
iris2.getNumPartitions()
```

```
8
```

```
for i in iris2.collect():
  print(i)
```

```
Id,SepalLengthCm,SepalWidthCm,PetalLengthCm,PetalWidthCm,Species
1,5.1,3.5,1.4,0.2,Iris-setosa
2,4.9,3.0,1.4,0.2,Iris-setosa
3,4.7,3.2,1.3,0.2,Iris-setosa
4,4.6,3.1,1.5,0.2,Iris-setosa
5,5.0,3.6,1.4,0.2,Iris-setosa
6,5.4,3.9,1.7,0.4,Iris-setosa
7,4.6,3.4,1.4,0.3,Iris-setosa
8,5.0,3.4,1.5,0.2,Iris-setosa
9,4.4,2.9,1.4,0.2,Iris-setosa
10,4.9,3.1,1.5,0.1,Iris-setosa
11,5.4,3.7,1.5,0.2,Iris-setosa
12,4.8,3.4,1.6,0.2,Iris-setosa
13,4.8,3.0,1.4,0.1,Iris-setosa
14,4.3,3.0,1.1,0.1,Iris-setosa
15,5.8,4.0,1.2,0.2,Iris-setosa
16,5.7,4.4,1.5,0.4,Iris-setosa
17,5.4,3.9,1.3,0.4,Iris-setosa
```

```
str_to_igr = 'Id,SepalLengthCm,SepalWidthCm,PetalLengthCm,PetalWidthCm,Species'
iris2 = iris2.filter(lambda x: x!=str_to_igr)
iris2.collect()
```

```
['1,5.1,3.5,1.4,0.2,Iris-setosa',
 '2,4.9,3.0,1.4,0.2,Iris-setosa',
 '3,4.7,3.2,1.3,0.2,Iris-setosa',
 '4,4.6,3.1,1.5,0.2,Iris-setosa',
 '5,5.0,3.6,1.4,0.2,Iris-setosa',
 '6,5.4,3.9,1.7,0.4,Iris-setosa',
 '7,4.6,3.4,1.4,0.3,Iris-setosa',
 '8,5.0,3.4,1.5,0.2,Iris-setosa',
 '9,4.4,2.9,1.4,0.2,Iris-setosa',
 '10,4.9,3.1,1.5,0.1,Iris-setosa',
 '11,5.4,3.7,1.5,0.2,Iris-setosa',
 '12,4.8,3.4,1.6,0.2,Iris-setosa',
 '13,4.8,3.0,1.4,0.1,Iris-setosa',
 '14,4.3,3.0,1.1,0.1,Iris-setosa',
 '15,5.8,4.0,1.2,0.2,Iris-setosa',
```

```
#Map Transformation
iris3 = iris2.map(lambda x: x*2) #repeating the same string 2 times
iris3.collect()
```

```
['1,5.1,3.5,1.4,0.2,Iris-setosa1,5.1,3.5,1.4,0.2,Iris-setosa',
 '2,4.9,3.0,1.4,0.2,Iris-setosa2,4.9,3.0,1.4,0.2,Iris-setosa',
 '3,4.7,3.2,1.3,0.2,Iris-setosa3,4.7,3.2,1.3,0.2,Iris-setosa',
 '4,4.6,3.1,1.5,0.2,Iris-setosa4,4.6,3.1,1.5,0.2,Iris-setosa',
 '5,5.0,3.6,1.4,0.2,Iris-setosa5,5.0,3.6,1.4,0.2,Iris-setosa',
 '6,5.4,3.9,1.7,0.4,Iris-setosa6,5.4,3.9,1.7,0.4,Iris-setosa',
 '7,4.6,3.4,1.4,0.3,Iris-setosa7,4.6,3.4,1.4,0.3,Iris-setosa',
 '8,5.0,3.4,1.5,0.2,Iris-setosa8,5.0,3.4,1.5,0.2,Iris-setosa',
 '9,4.4,2.9,1.4,0.2,Iris-setosa9,4.4,2.9,1.4,0.2,Iris-setosa',
 '10,4.9,3.1,1.5,0.1,Iris-setosa10,4.9,3.1,1.5,0.1,Iris-setosa',
 '11,5.4,3.7,1.5,0.2,Iris-setosa11,5.4,3.7,1.5,0.2,Iris-setosa',
 '12,4.8,3.4,1.6,0.2,Iris-setosa12,4.8,3.4,1.6,0.2,Iris-setosa',
 '13,4.8,3.0,1.4,0.1,Iris-setosa13,4.8,3.0,1.4,0.1,Iris-setosa',
 '14,4.3,3.0,1.1,0.1,Iris-setosa14,4.3,3.0,1.1,0.1,Iris-setosa',
 '15,5.8,4.0,1.2,0.2,Iris-setosa15,5.8,4.0,1.2,0.2,Iris-setosa',
 '16,5.7,4.4,1.5,0.4,Iris-setosa16,5.7,4.4,1.5,0.4,Iris-setosa',
 '17,5.4,3.9,1.3,0.4,Iris-setosa17,5.4,3.9,1.3,0.4,Iris-setosa',
 '18,5.1,3.5,1.4,0.3,Iris-setosa18,5.1,3.5,1.4,0.3,Iris-setosa',
```

```
#Flattening  a text file
iris4 = iris2.flatMap(lambda x: x.split(","))
iris4.collect()
```

```
['1',
 '5.1',
 '3.5',
 '1.4',
 '0.2',
 'Iris-setosa',
 '2',
 '4.9',
 '3.0',
 '1.4',
 '0.2',
 'Iris-setosa',
```

```
partitions = iris4.glom().collect()
print("Total partitions:", len(partitions))
```

```
Total partitions: 8
```

```
partitions[0]
```

```
['1',
 '5.1',
 '3.5',
 '1.4',
 '0.2',
 'Iris-setosa',
 '2',
 '4.9',
 '3.0',
 '1.4',
 '0.2',
 'Iris-setosa',
 '3',
 '4.7',
 '3.2',
 '1.3',
 '0.2',
```

```
iris5 = iris4.map(lambda x: x*2)
iris5 = iris4.map(lambda x: x*2)
```

```
['11', '5.15.1', '3.53.5', '1.41.4', '0.20.2']
```

```
iris4.take(5)
```

```
['1', '5.1', '3.5', '1.4', '0.2']
```

```
#Filtering
iris6 = iris1.flatMap(lambda x: x.split(",")).filter(lambda x: x=='Iris-
virginica').map(lambda x: (x,1))
partitions = iris6.glom().collect()
partitions
```

```
[[],
 [('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
  ('Iris-virginica', 1),
```

```
species = ['Iris-virginica','Iris-setosa','Iris-versicolor']
species1 = iris1.flatMap(lambda x: x.split(",")).filter(lambda x: x in species)
species1.collect()
```

```
['Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
 'Iris-setosa',
```

```
species2 = species1.groupBy(lambda x: x).mapValues(lambda x: len(x))
species2.collect()
```

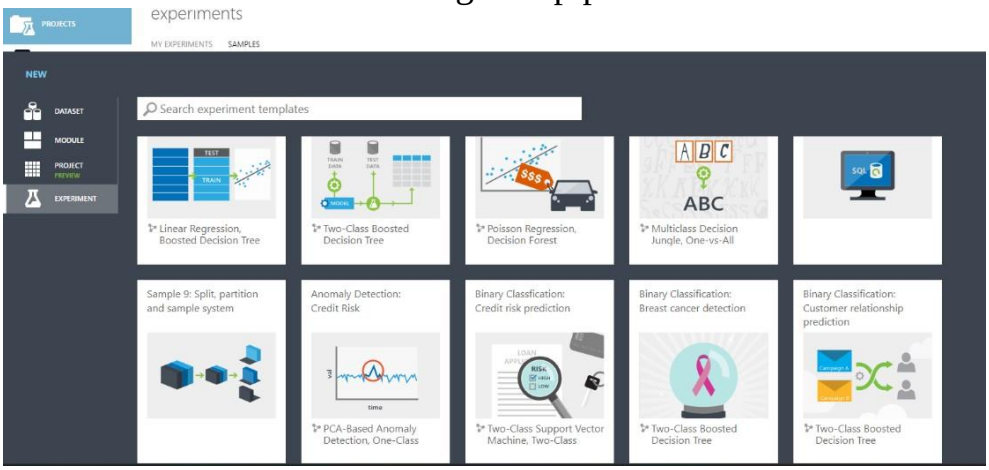
```
[('Iris-setosa', 50), ('Iris-virginica', 50), ('Iris-versicolor', 50)]
```

PRACTICAL NO.: 07 (a)

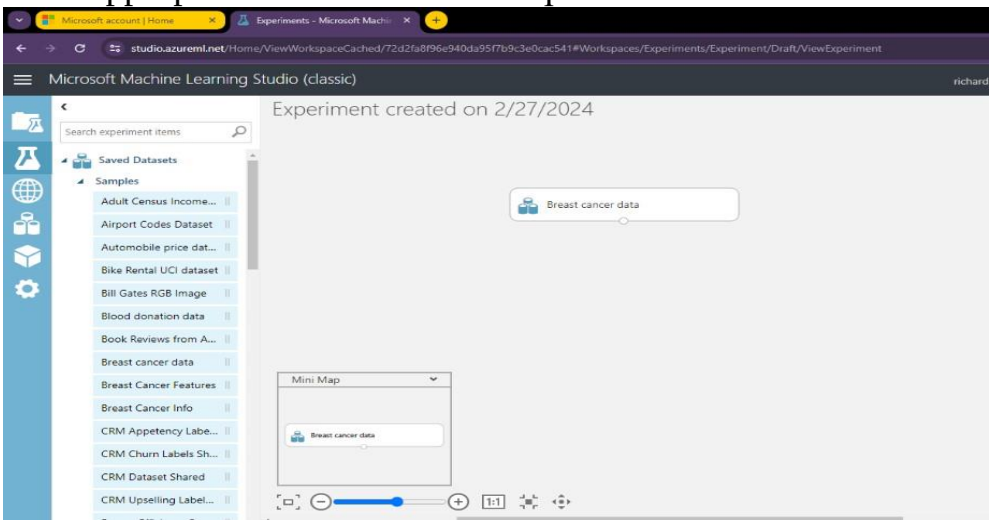
Course Code: RJOECDSA121 **Date:**

AIM: Deploy ML Pipeline for classification using Azure ML

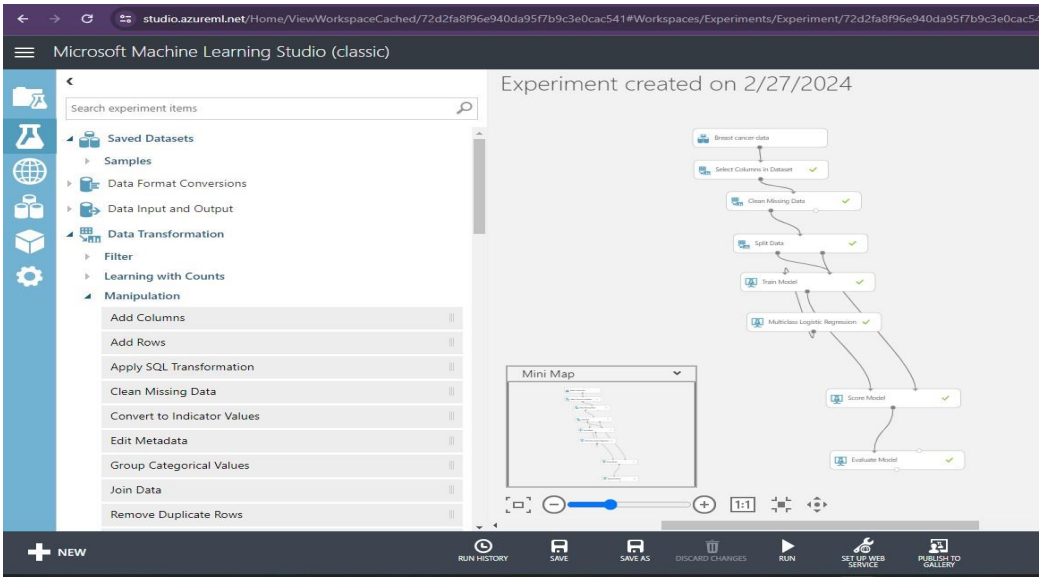
- Go to azureml.net >> log in using Microsoft account >> Go to “Experiment” section >> click on “NEW” for making new pipeline

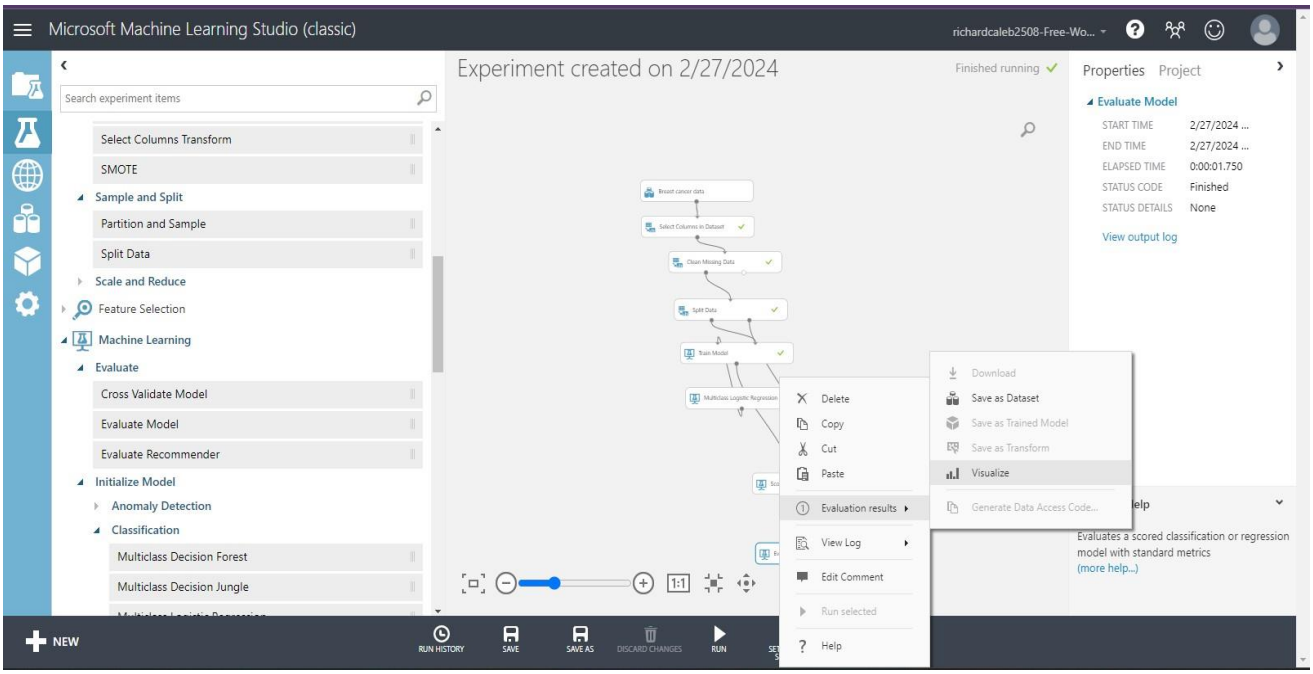


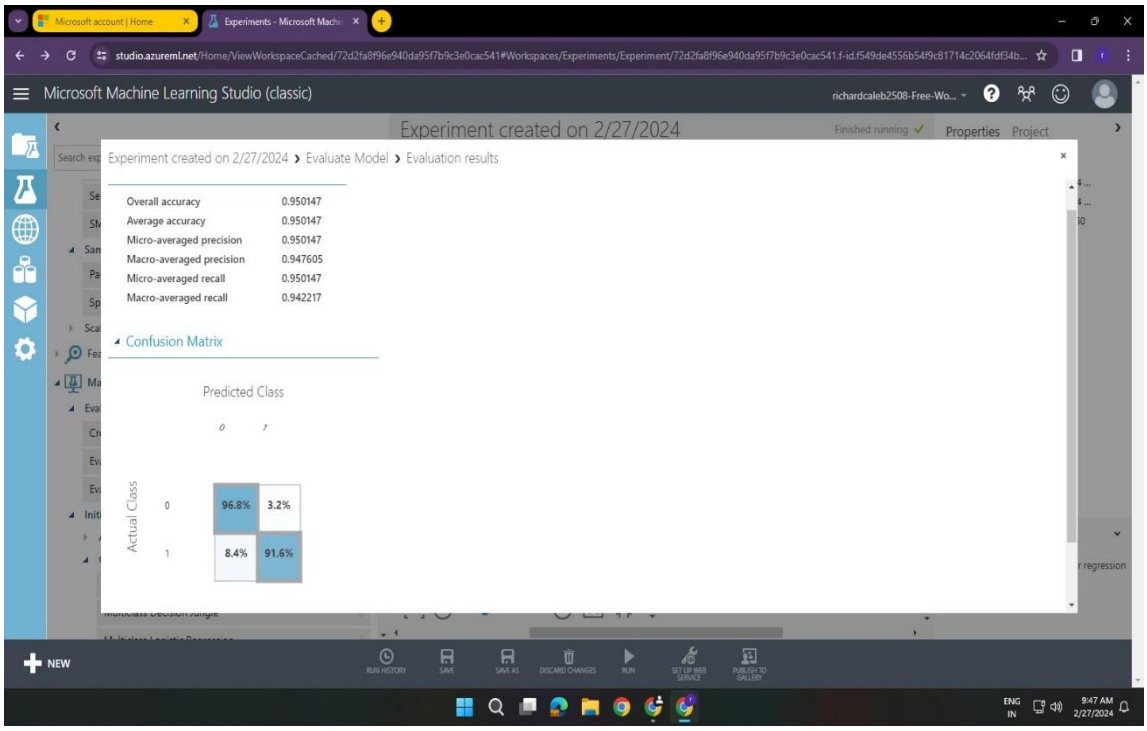
- Load appropriate dataset from “Sample Dataset” section from left hand side



- Drag and drop the necessary elements for constructing the pipeline from left side and create the final model




Microsoft Machine Learning Studio (classic)
richardcaleb2508-Free-Wo...
Experiment created on 2/27/2024
Finished running
Properties
Project
Search experiment items
Select Columns Transform
SMOTE
Sample and Split
Partition and Sample
Split Data
Scale and Reduce
Feature Selection
Machine Learning
Evaluate
Cross Validate Model
Evaluate Model
Evaluate Recommender
Initialize Model
Anomaly Detection
Classification
Multiclass Decision Forest
Multiclass Decision Jungle
Evaluate Model
START TIME 2/27/2024 ...
END TIME 2/27/2024 ...
ELAPSED TIME 0:00:01.750
STATUS CODE Finished
STATUS DETAILS None
View output log
Delete
Copy
Cut
Paste
Evaluation results
View Log
Edit Comment
Run selected
Help
Download
Save as Dataset
Save as Trained Model
Save as Transform
Visualize
Generate Data Access Code...
Evaluates a scored classification or regression model with standard metrics
(more help...)
NEW
RUN HISTORY
SAVE
SAVE AS
DISCARD CHANGES
RUN

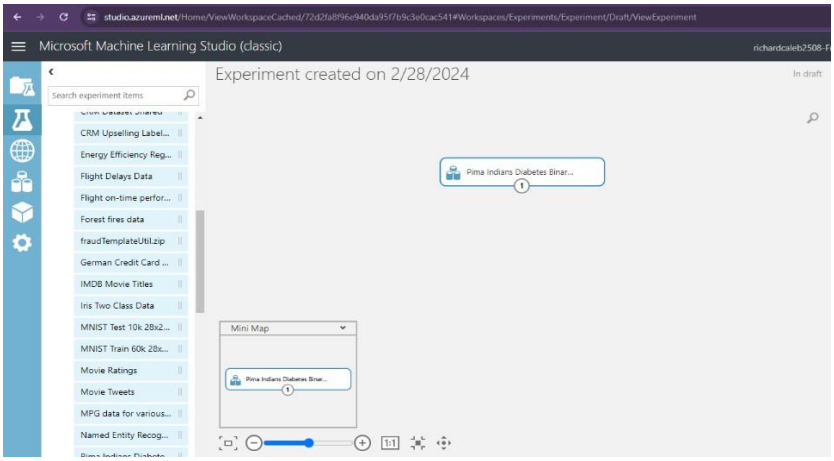

Microsoft Machine Learning Studio (classic)
Experiment created on 2/27/2024
Experiment created on 2/27/2024 › Evaluate Model › Evaluation results
Overall accuracy 0.950147
Average accuracy 0.950147
Micro-averaged precision 0.950147
Macro-averaged precision 0.947605
Micro-averaged recall 0.950147
Macro-averaged recall 0.942217
Confusion Matrix
Predicted Class
Actual Class
96.8%
3.2%
8.4%
91.6%
NEW

PRACTICAL NO.: 07 (c)

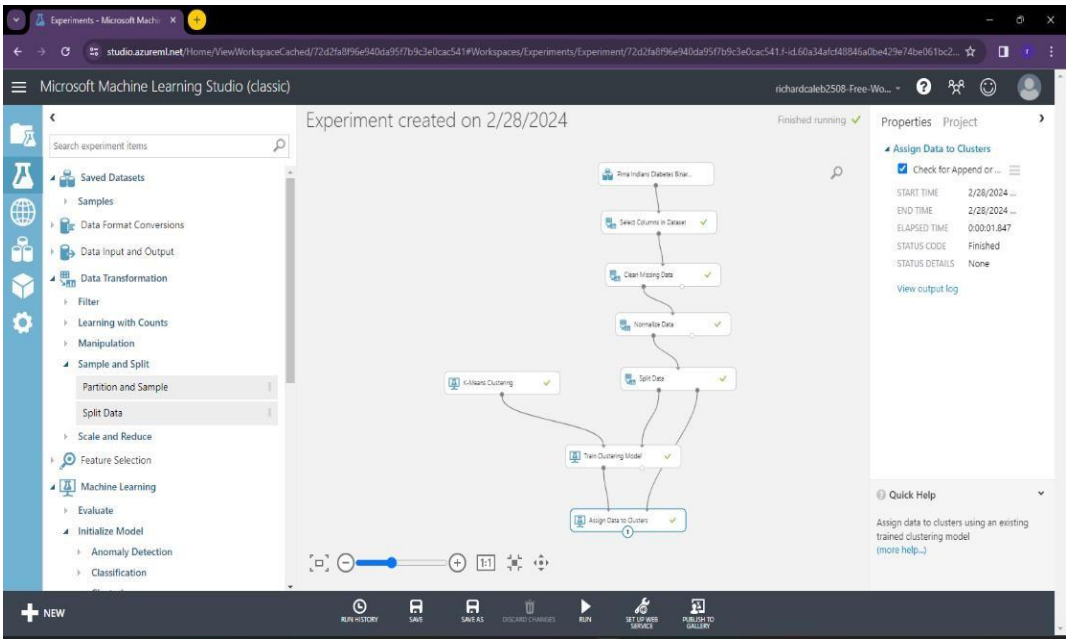
Course Code: RJOECDSA121 **Date:**

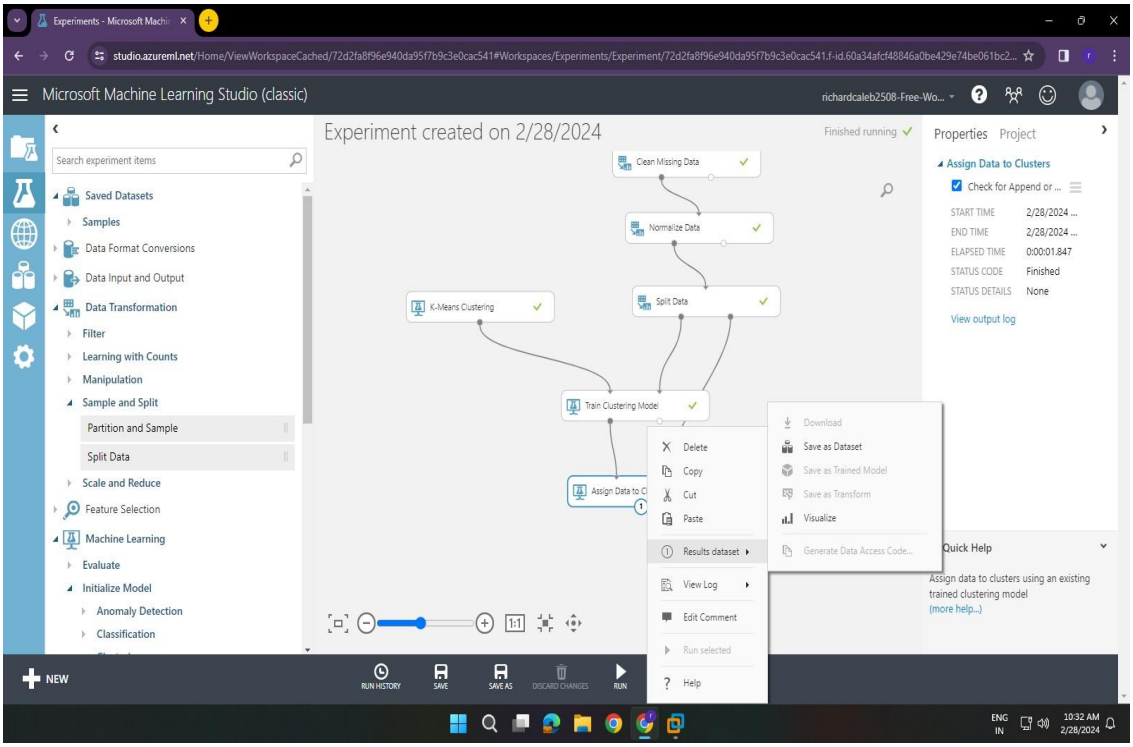
AIM: Deploy ML Pipeline for clustering using Azure ML

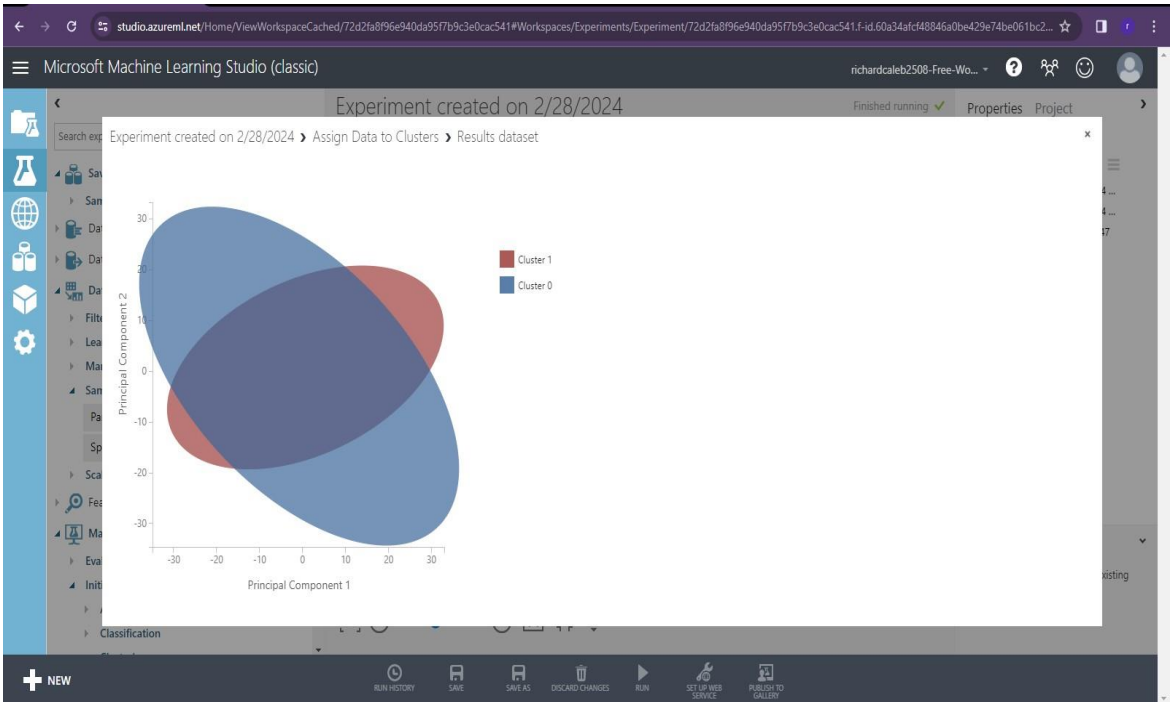
- Go to azureml.net >> log in using Microsoft account >> Go to “Experiment” section >> click on “NEW” for making new pipeline
- Load appropriate dataset from “Sample Dataset” section from left hand side



- Drag and drop the necessary elements for constructing the pipeline from left side and create the final model




Microsoft Machine Learning Studio (classic)
Experiment created on 2/28/2024
Finished running
Clean Missing Data
Normalize Data
K-Means Clustering
Split Data
Train Clustering Model
Assign Data to Clusters
Properties
Project
Assign Data to Clusters
Check for Append or ...
START TIME 2/28/2024 ...
END TIME 2/28/2024 ...
ELAPSED TIME 0:00:01.847
STATUS CODE Finished
STATUS DETAILS None
View output log
Quick Help
Assign data to clusters using an existing trained clustering model
(more help...)
Saved Datasets
Samples
Data Format Conversions
Data Input and Output
Data Transformation
Filter
Learning with Counts
Manipulation
Sample and Split
Partition and Sample
Split Data
Scale and Reduce
Feature Selection
Machine Learning
Evaluate
Initialize Model
Anomaly Detection
Classification
Delete
Copy
Cut
Paste
Results dataset
View Log
Edit Comment
Run selected
Help
Download
Save as Dataset
Save as Trained Model
Save as Transform
Visualize
Generate Data Access Code...
NEW
RUN HISTORY
SAVE
SAVE AS
DISCARD CHANGES
RUN


Microsoft Machine Learning Studio (classic)
Experiment created on 2/28/2024
Experiment created on 2/28/2024 › Assign Data to Clusters › Results dataset
Cluster 1
Cluster 0
Principal Component 2
Principal Component 1
30
20
10
0
-10
-20
-30
Properties
Project
Classification
NEW
RUN HISTORY
SAVE
SAVE AS
DISCARD CHANGES
RUN
SET UP WEB SERVICE
PUBLISH TO GALLERY

PRACTICAL NO.: 08

Course Code: RJOECDSA121 **Date:**

AIM: Working with different HBase commands to handle column-oriented NoSQL HBase Db

- Open Cloudera >> Go to Terminal >> write “hbase shell” to go into it
- Use “Create” command to create table and column family
  By default first name is taken as ‘table_name’ and rest as ‘column_family’

```
hbase(main):002:0> create 'stud','studid','studname','studaddress'
0 row(s) in 1.2260 seconds

=> Hbase::Table - stud
hbase(main):003:0>
```

- Use “Put” command to put the values into the table

```
hbase(main):004:0> put 'stud','01','studname:fname','amit'
0 row(s) in 0.1050 seconds

hbase(main):005:0> put 'stud','01','studname:mname','Prakash'
0 row(s) in 0.0210 seconds

hbase(main):006:0> put 'stud','01','studname:lname','Vaze'
0 row(s) in 0.0080 seconds

hbase(main):007:0> put 'stud','01','studaddress:place','Mulund'
0 row(s) in 0.0070 seconds
```

put ‘table_name’, ‘row_key’, ‘column_family’:’column_name’,’value’

- “Scan” command to see the full values inside the table

```
hbase(main):031:0> scan 'stud'
ROW                                   COLUMN+CELL
 01                                   column=studaddress:place, timestamp=1709100151044, value=Mulund
 01                                   column=studname:fname, timestamp=1709100036127, value=amit
 01                                   column=studname:lname, timestamp=1709100105163, value=Vaze
 01                                   column=studname:mname, timestamp=1709100085953, value=Prakash
 02                                   column=studaddress:landmark, timestamp=1709100362194, value=Devi Dayal Rd
 02                                   column=studaddress:place, timestamp=1709100389571, value=Mulund W
 02                                   column=studname:fname, timestamp=1709100259118, value=Sumit
 02                                   column=studname:lname, timestamp=1709100298148, value=Kulkarni
 02                                   column=studname:mname, timestamp=1709100277098, value=Neeraj
 03                                   column=studaddress:landmark, timestamp=1709100699020, value=PK Rd
 03                                   column=studaddress:place, timestamp=1709100719642, value=Thane
 03                                   column=studaddress:state, timestamp=1709100767634, value=Maharastra
 03                                   column=studname:fname, timestamp=1709100552591, value=Alka
 03                                   column=studname:lname, timestamp=1709100594682, value=Yadav
 03                                   column=studname:mname, timestamp=1709100579454, value=Deepak
 04                                   column=studaddress:landmark, timestamp=1709100863068, value=Goshala Rd
 04                                   column=studaddress:place, timestamp=1709100887004, value=Mulund W
 04                                   column=studaddress:state, timestamp=1709100995620, value=Maharastra
 04                                   column=studname:fname, timestamp=1709100795494, value=Amit
 04                                   column=studname:lname, timestamp=1709100830499, value=Yadav
 04                                   column=studname:mname, timestamp=1709100815489, value=Neeraj
 05                                   column=studaddress:landmark, timestamp=1709100999035, value=JN Rd
 05                                   column=studaddress:place, timestamp=1709101026285, value=Thane W
 05                                   column=studaddress:state, timestamp=1709101039924, value=Maharastra
 05                                   column=studname:fname, timestamp=1709100928704, value=Veena
 05                                   column=studname:lname, timestamp=1709100973346, value=Mhatre
 05                                   column=studname:mname, timestamp=1709100951874, value=Pushkar
5 row(s) in 0.0640 seconds
```

Scan ‘table_name’

- “get” command for retrieving each particular value

```
hbase(main):032:0> get 'stud','01',{COLUMN => 'studaddress:place'}
COLUMN                                CELL
 studaddress:place                    timestamp=1709100151044, value=Mulund
1 row(s) in 0.0180 seconds

hbase(main):033:0> get 'stud','03',{COLUMN => 'studaddress'}
COLUMN                                CELL
 studaddress:landmark                 timestamp=1709100699020, value=PK Rd
 studaddress:place                    timestamp=1709100719642, value=Thane
 studaddress:state                    timestamp=1709100767634, value=Maharastra
3 row(s) in 0.0030 seconds
```

get ‘table_name’,’row_key’,{COLUMN => ‘column_family’:’column_name’

- Using “get” command with only row_id

```
hbase(main):034:0> get 'stud','01'
COLUMN                                CELL
 studaddress:place                    timestamp=1709100151044, value=Mulund
 studname:fname                       timestamp=1709100036127, value=amit
 studname:lname                       timestamp=1709100105163, value=Vaze
 studname:mname                       timestamp=1709100085953, value=Prakash
4 row(s) in 0.0120 seconds
```

- “count” command for retrieving the total no. of values present

```
hbase(main):042:0> count 'stud'
5 row(s) in 0.0450 seconds

=> 5
```

- Disabling the table using “disable” and checking it

```
hbase(main):043:0> disable 'stud'
0 row(s) in 2.2830 seconds

hbase(main):044:0> is_disabled 'stud'
true
0 row(s) in 0.0170 seconds
```

- Enabling the table using “enable” command checking it using “scan” command

```
hbase(main):045:0> enable 'stud'
0 row(s) in 1.2740 seconds

hbase(main):046:0> scan 'stud'
ROW                                     COLUMN+CELL
 01                                     column=studaddress:place, timestamp=1709100151044, value=Mulund
 01                                     column=studname:fname, timestamp=1709100036127, value=amit
 01                                     column=studname:lname, timestamp=1709100105163, value=Vaze
 01                                     column=studname:mname, timestamp=1709100085953, value=Prakash
 02                                     column=studaddress:landmark, timestamp=1709100362194, value=Devi Dayal Rd
 02                                     column=studaddress:place, timestamp=1709100389571, value=Mulund W
 02                                     column=studname:fname, timestamp=1709100259118, value=Sumit
 02                                     column=studname:lname, timestamp=1709100298148, value=Kulkarni
 02                                     column=studname:mname, timestamp=1709100277098, value=Neeraj
 03                                     column=studaddress:landmark, timestamp=1709100699020, value=PK Rd
 03                                     column=studaddress:place, timestamp=1709100719642, value=Thane
 03                                     column=studaddress:state, timestamp=1709100767634, value=Maharastra
 03                                     column=studname:fname, timestamp=1709100552591, value=Alka
 03                                     column=studname:lname, timestamp=1709100594682, value=Yadav
 03                                     column=studname:mname, timestamp=1709100579454, value=Deepak
 04                                     column=studaddress:landmark, timestamp=1709100863068, value=Goshala Rd
 04                                     column=studaddress:place, timestamp=1709100887004, value=Mulund W
 04                                     column=studaddress:state, timestamp=1709100905620, value=Maharastra
 04                                     column=studname:fname, timestamp=1709100795494, value=Amit
 04                                     column=studname:lname, timestamp=1709100830499, value=Yadav
 04                                     column=studname:mname, timestamp=1709100815489, value=Neeraj
 05                                     column=studaddress:landmark, timestamp=1709100999835, value=JN Rd
 05                                     column=studaddress:place, timestamp=1709101026285, value=Thane W
 05                                     column=studaddress:state, timestamp=1709101039924, value=Maharastra
 05                                     column=studname:fname, timestamp=1709100928784, value=Veena
 05                                     column=studname:lname, timestamp=1709100973346, value=Mhatre
 05                                     column=studname:mname, timestamp=1709100951874, value=Pushkar
5 row(s) in 0.0220 seconds
```

- Deleting particular value

```
hbase(main):047:0> delete 'stud','01','studname:mname'
0 row(s) in 0.0480 seconds

hbase(main):048:0> scan 'stud'
ROW                                     COLUMN+CELL
 01                                     column=studaddress:place, timestamp=1709100151044, value=Mulund
 01                                     column=studname:fname, timestamp=1709100036127, value=amit
 01                                     column=studname:lname, timestamp=1709100105163, value=Vaze
 02                                     column=studaddress:landmark, timestamp=1709100362194, value=Devi Dayal Rd
 02                                     column=studaddress:place, timestamp=1709100389571, value=Mulund W
 02                                     column=studname:fname, timestamp=1709100259118, value=Sumit
 02                                     column=studname:lname, timestamp=1709100298148, value=Kulkarni
 02                                     column=studname:mname, timestamp=1709100277098, value=Neeraj
 03                                     column=studaddress:landmark, timestamp=1709100699020, value=PK Rd
 03                                     column=studaddress:place, timestamp=1709100719642, value=Thane
 03                                     column=studaddress:state, timestamp=1709100767634, value=Maharastra
 03                                     column=studname:fname, timestamp=1709100552591, value=Alka
 03                                     column=studname:lname, timestamp=1709100594682, value=Yadav
 03                                     column=studname:mname, timestamp=1709100579454, value=Deepak
 04                                     column=studaddress:landmark, timestamp=1709100863068, value=Goshala Rd
 04                                     column=studaddress:place, timestamp=1709100887004, value=Mulund W
 04                                     column=studaddress:state, timestamp=1709100905620, value=Maharastra
 04                                     column=studname:fname, timestamp=1709100795494, value=Amit
 04                                     column=studname:lname, timestamp=1709100830499, value=Yadav
 04                                     column=studname:mname, timestamp=1709100815489, value=Neeraj
 05                                     column=studaddress:landmark, timestamp=1709100999835, value=JN Rd
 05                                     column=studaddress:place, timestamp=1709101026285, value=Thane W
 05                                     column=studaddress:state, timestamp=1709101039924, value=Maharastra
 05                                     column=studname:fname, timestamp=1709100928784, value=Veena
 05                                     column=studname:lname, timestamp=1709100973346, value=Mhatre
 05                                     column=studname:mname, timestamp=1709100951874, value=Pushkar
5 row(s) in 0.0470 seconds
```

- Dropping the table using “drop” command before that make use of “disable” and once dropping check with “exists” command

```
hbase(main):059:0> exists 'stud'
Table stud does exist
0 row(s) in 0.0120 seconds

hbase(main):060:0> disable 'stud'
0 row(s) in 2.2480 seconds

hbase(main):061:0> drop 'stud'
0 row(s) in 1.2510 seconds

hbase(main):062:0> list
TABLE
student
table1
2 row(s) in 0.0050 seconds

=> ["student", "table1"]
hbase(main):063:0> exists 'stud'
Table stud does not exist
0 row(s) in 0.0280 seconds
```

PRACTICAL NO.: 04

Course Code: RJOECDSA121 **Date:**

AIM: Install & Demonstrate the working of Kafka

- Download Kafka >> extract the file in “Downloads” >> Copy it in C:

DOWNLOAD

3.7.0 is the latest release. The current stable version is 3.7.0

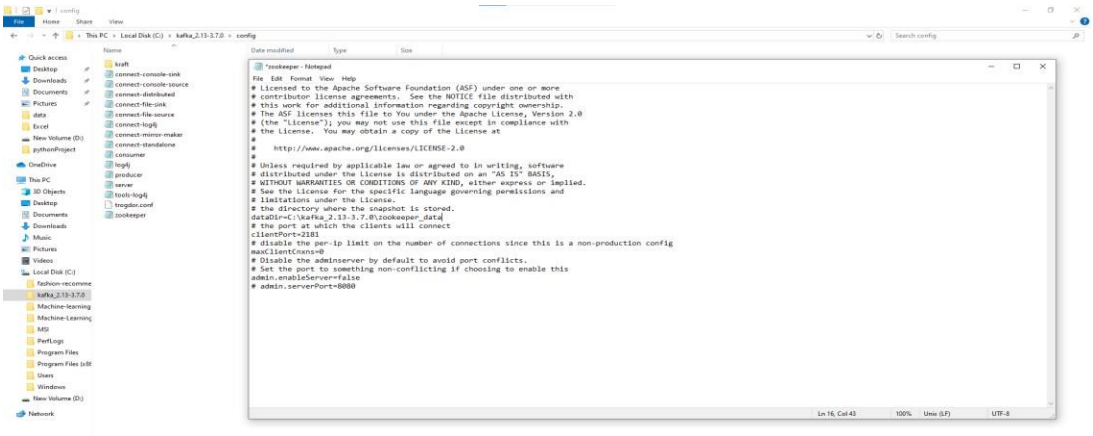
You can verify your download by following these procedures and using these KEYS.

3.7.0

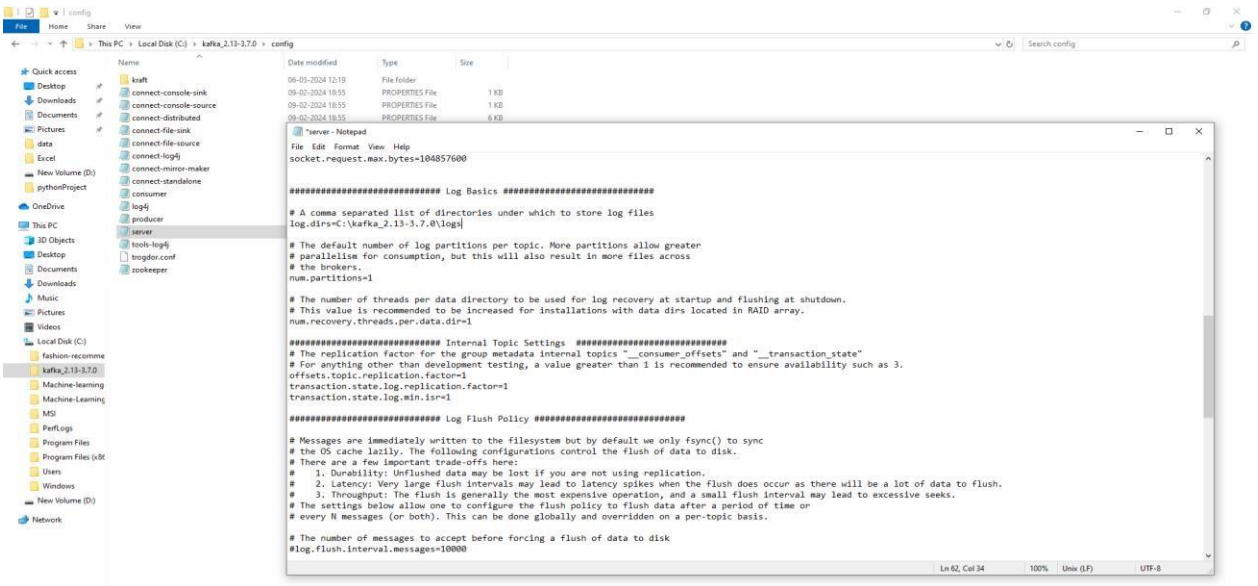
- Released Feb 27, 2024
- Release Notes
- Docker image: apache/kafka:3.7.0
- Source download: kafka-3.7.0-src.tgz (asc, sha512)
- Binary downloads:
  - Scala 2.12 - kafka_2.12-3.7.0.tgz (asc, sha512)
  - Scala 2.13 - kafka_2.13-3.7.0.tgz (asc, sha512)

We build for multiple versions of Scala. This only matters if you are using Scala and you want a version built for the same Scala version you use.
Otherwise any version should work (2.13 is recommended).

- Inside Kafka >> go to “Zookeeper” >> set “dataDir” >> paste the kafka path in this >> add “\zookeeper_data” in front



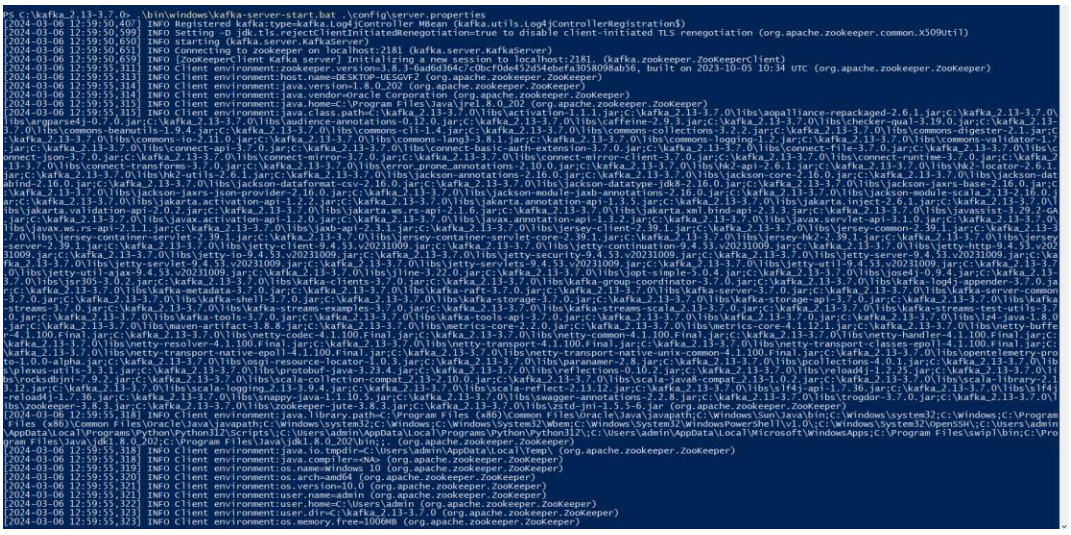
Again go inside Kafka >> go to “server” >> set “log.dirs” >> paste the kafka path in this >> add “\logs” in last



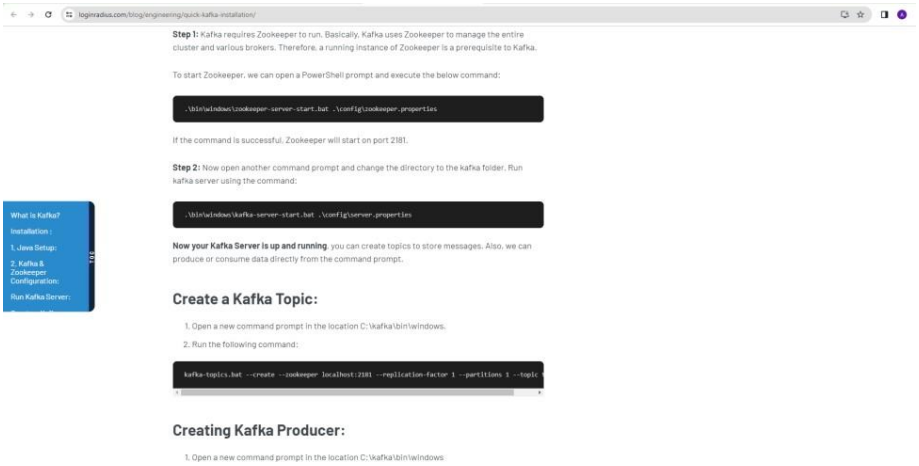
- Open kafka in “Powershell” >> activate the “zookeeper” by going inside “.\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties”

- Again open kafka in “Powershell” >> active the server.properties by “.\bin\windows\zookeeper-server-start.bat .\config\server.properties”

Or “\bin>windows\kafka-server-start.bat C:\kafka_2.13-3.7.0\config\server.properties”



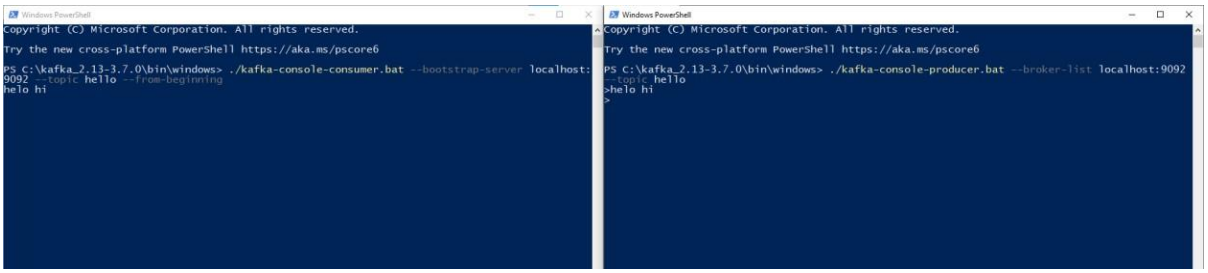
- Go to login radius kafka >> to get the command for creating topic , consumer and producer



- Go to windows inside kafka through bins >> open “Power Shell” >> type this command “kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test”

In place of –zookeeper write - -bootstrap and localhost port to 9092 and set a topic name of choice

- Go to windows inside kafka through bins >> open “Power Shell” >> use this command “kafka-console-producer.bat --broker-list localhost:9092 --topic test” for producer repeat the above and write “kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test --from-beginning” for consumer



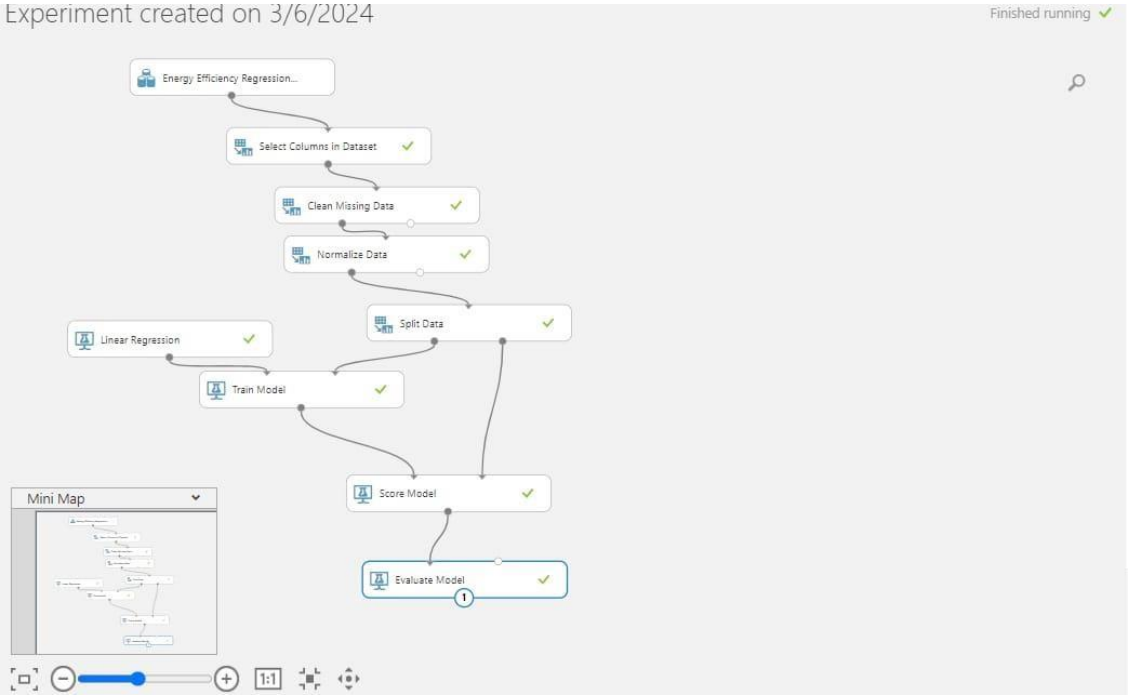
Then send any message via producer to consumer

PRACTICAL NO.: 07 (b)

Course Code: RJOECDSA121 **Date:**

AIM: Deploy ML Pipeline for regression using Azure ML

- Go to azureml.net >> log in using Microsoft account >> Go to "Experiment" section
  >> click on "NEW" for making new pipeline
- Load appropriate dataset from "Sample Dataset" section from left hand side panel



- Right Click on "Evaluate Model" >> To get the Evaluation Metrics