UMKC

UNIVERSITY OF MISSOURI-KANSAS CITY

Bigdata Hadoop Programming

# Lab 4 Assignment

Team Members:

Pragathi Thammaneni

Sridevi Mallipudi

Introduction:

The main core concept for executing the Lab 4 Assignment is to implement Spark classification task and Spark Streaming Task**.**

Objectives:

To code for the 2 questions the below concepts are implemented.

- Implementing the Classification algorithms – Naive Bayes, Decision Tree and Random Forest
- Twitter Streaming Data using Spark

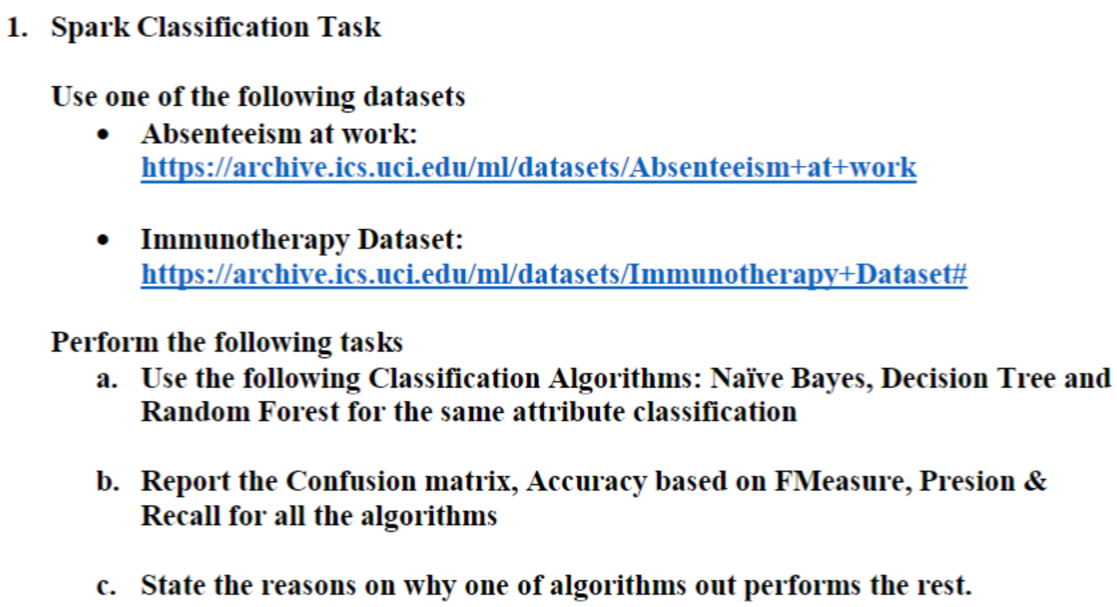
Approaches /Methods:

PySpark , PyCharm

Workflow &Datasets/Parameters and Evaluation:

The below each question will follow different approaches to solve. Coding is done to perform the evaluation of each individual snippet to execute the datasets which are provided as the input parameters.

Question 1:



1. **Spark Classification Task**

   Use one of the following datasets

   - **Absenteeism at work:** **https://archive.ics.uci.edu/ml/datasets/Absenteeism+at+work**
   - **Immunotherapy Dataset:** **https://archive.ics.uci.edu/ml/datasets/Immunotherapy+Dataset#**

   Perform the following tasks

   a. **Use the following Classification Algorithms: Naïve Bayes, Decision Tree and Random Forest for the same attribute classification**

   b. **Report the Confusion matrix, Accuracy based on FMeasure, Presion & Recall for all the algorithms**

   c. **State the reasons on why one of algorithms out performs the rest.**

Solution:

Implemented the three classification algorithms -Naïve Bayes, Decision Tree and Random Forest

Data Set: Absenteeism

Training and testing split ratio :70 and 30

Naïve Bayes Classification

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a feature in a class is unrelated to the presence of any other feature.

Accuracy, precision and recall:

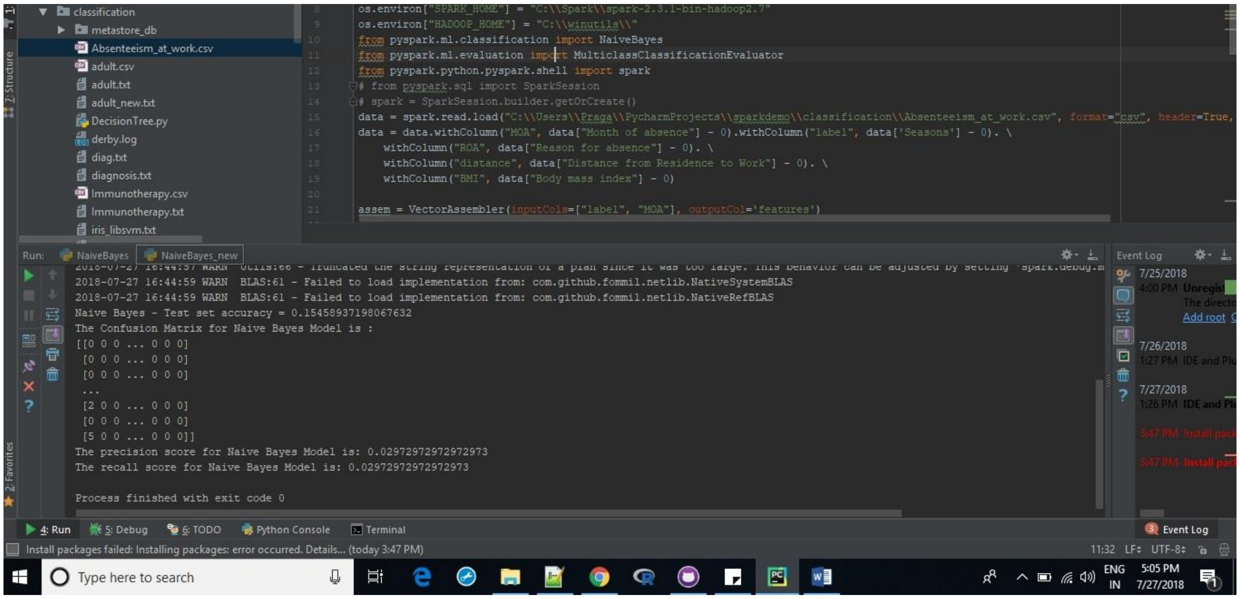
Accuracy refers to the closeness of a measured value to a standard or known value. Precision refers to the closeness of two or more measurements to each other. Recall referred to as the true positive rate or sensitivity.

Accuracy – 0.1545

Precision – 0.0297

Recall –0.0297

Output Screens:



```
os.environ["SPARK_HOME"] = "C:\\Spark\\spark-2.3.1-bin-hadoop2.7"
os.environ["HADOOP_HOME"] = "C:\\winutils\\"
from pyspark.ml.classification import NaiveBayes
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.python.pyspark.shell import spark
# from pyspark.sql import SparkSession
# spark = SparkSession.builder.getOrCreate()
data = spark.read.load("C:\\Users\\Praga\\PycharmProjects\\sparkdemo\\classification\\Absenteeism_at_work.csv", format="csv", header=True,
data = data.withColumn("MOA", data["Month of absence"] - 0).withColumn("label", data['Seasons'] - 0). \
    withColumn("ROA", data["Reason for absence"] - 0). \
    withColumn("distance", data["Distance from Residence to Work"] - 0). \
    withColumn("BMI", data["Body mass index"] - 0)

assem = VectorAssembler(inputCols=["label", "MOA"], outputCol='features')
```

```
2018-07-27 16:44:57 WARN  Utils:66 - Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.debug.m
2018-07-27 16:44:59 WARN  BLAS:61 - Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
2018-07-27 16:44:59 WARN  BLAS:61 - Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
Naive Bayes - Test set accuracy = 0.15458937198067632
The Confusion Matrix for Naive Bayes Model is :
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [2 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [5 0 0 ... 0 0 0]]
The precision score for Naive Bayes Model is: 0.02972972972972973
The recall score for Naive Bayes Model is: 0.02972972972972973

Process finished with exit code 0
```

Install packages failed: Installing packages: error occurred. Details... (today 3:47 PM)

Decision Tree:

Decision Tree - Classification. Decision tree builds classification or regression models in the form of a tree structure. The result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy).

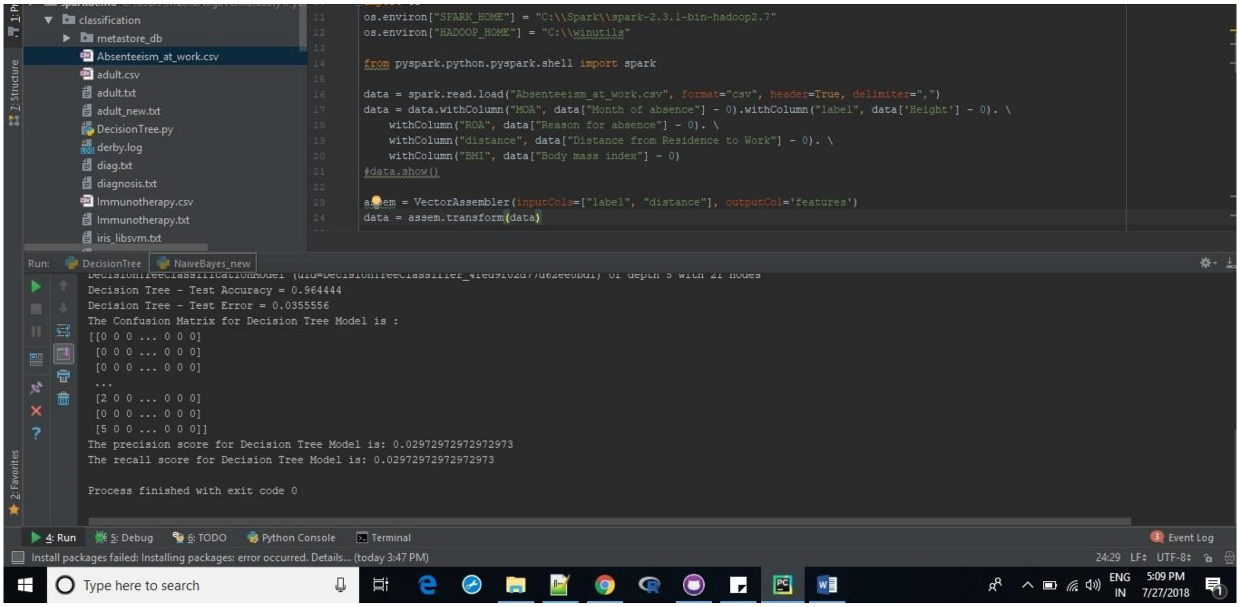
Accuracy, precision and recall:

Accuracy refers to the closeness of a measured value to a standard or known value. Precision refers to the closeness of two or more measurements to each other. Recall referred to as the true positive rate or sensitivity.

Accuracy – 0.96444

Precision – 0.0297

Recall –0.0297



```
os.environ["SPARK_HOME"] = "C:\\Spark\\spark-2.3.1-bin-hadoop2.7"
os.environ["HADOOP_HOME"] = "C:\\winutils"

from pyspark.python.pyspark.shell import spark

data = spark.read.load("Absenteeism_at_work.csv", format="csv", header=True, delimiter=",")
data = data.withColumn("MOA", data["Month of absence"] - 0).withColumn("label", data['Height'] - 0). \
    withColumn("ROA", data["Reason for absence"] - 0). \
    withColumn("distance", data["Distance from Residence to Work"] - 0). \
    withColumn("BMI", data["Body mass index"] - 0)
#data.show()

assem = VectorAssembler(inputCols=["label", "distance"], outputCol='features')
data = assem.transform(data)
```

```
DecisionTreeClassificationModel (uid=DecisionTreeClassifier_4fed9102d77d62ee0bd1) of depth 5 with 21 nodes
Decision Tree - Test Accuracy = 0.964444
Decision Tree - Test Error = 0.0355556
The Confusion Matrix for Decision Tree Model is :
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [2 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [5 0 0 ... 0 0 0]]
The precision score for Decision Tree Model is: 0.02972972972972973
The recall score for Decision Tree Model is: 0.02972972972972973

Process finished with exit code 0
```

Install packages failed: Installing packages: error occurred. Details... (today 3:47 PM)

Random Forest:

Random forest (or random forests) is a trademark term for an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the classes output by individual trees. Random forests are collections of trees, all slightly different. It randomizes the algorithm, not the training data.

Accuracy, precision and recall:

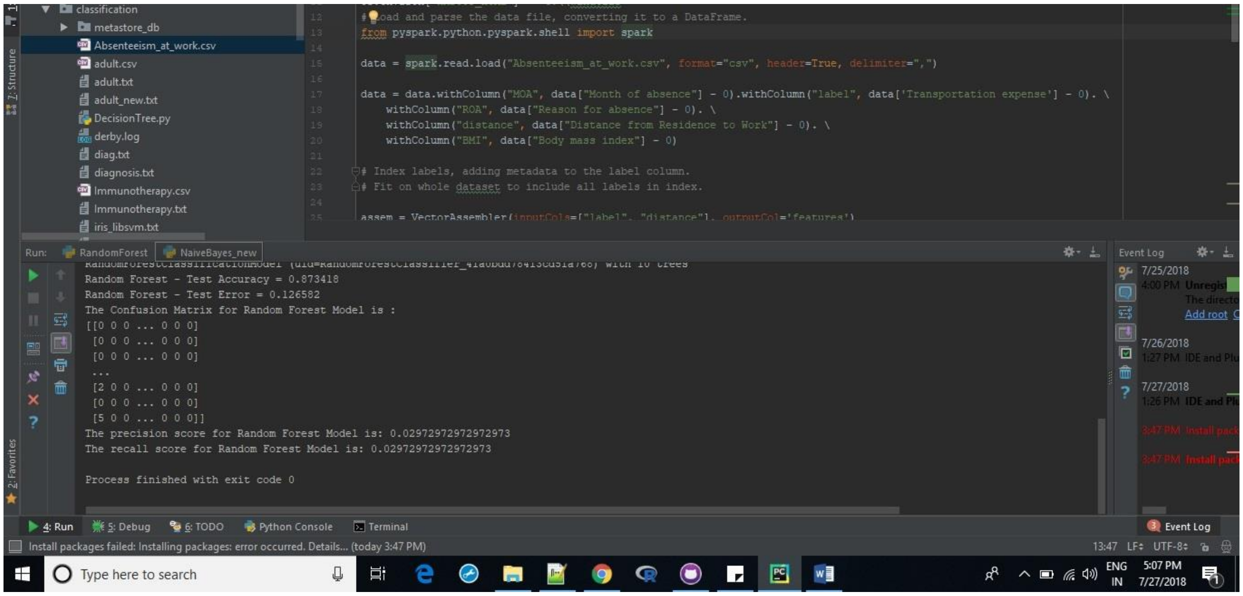
Accuracy refers to the closeness of a measured value to a standard or known value. Precision refers to the closeness of two or more measurements to each other. Recall referred to as the true positive rate or sensitivity.

Accuracy – 0.873

Precision – 0.0297

Recall –0.0297

Output Screen Shots:



```
#Load and parse the data file, converting it to a DataFrame.
from pyspark.python.pyspark.shell import spark

data = spark.read.load("Absenteeism_at_work.csv", format="csv", header=True, delimiter=",")

data = data.withColumn("MOA", data["Month of absence"] - 0).withColumn("label", data['Transportation expense'] - 0). \
    withColumn("ROA", data["Reason for absence"] - 0). \
    withColumn("distance", data["Distance from Residence to Work"] - 0). \
    withColumn("BMI", data["Body mass index"] - 0)

# Index labels, adding metadata to the label column.
# Fit on whole dataset to include all labels in index.

assem = VectorAssembler(inputCols=["label", "distance"], outputCol='features')
```

```
RandomForestClassificationModel (uid=RandomForestClassifier_41a0bdd78413cd51a768) with 10 trees
Random Forest - Test Accuracy = 0.873418
Random Forest - Test Error = 0.126582
The Confusion Matrix for Random Forest Model is :
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [2 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [5 0 0 ... 0 0 0]]
The precision score for Random Forest Model is: 0.02972972972972973
The recall score for Random Forest Model is: 0.02972972972972973

Process finished with exit code 0
```

```
Install packages failed: Installing packages: error occurred. Details... (today 3:47 PM)
```

Final Observations:

As Random forest (or random forests) is a trademark term for an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the classes output by individual trees. But in this scenario Decision tree out performs with 96% accuracy may be due to "**A single decision tree will train much more quickly, and computing a prediction is also much quicker" as random forest randomize the algorithm.**

Question 2:

2. Spark Streaming Task

Perform Word-Count on Twitter Streaming Data using Spark.

https://www.linkedin.com/pulse/apache-spark-streaming-twitter-python-laurent-weichberger/

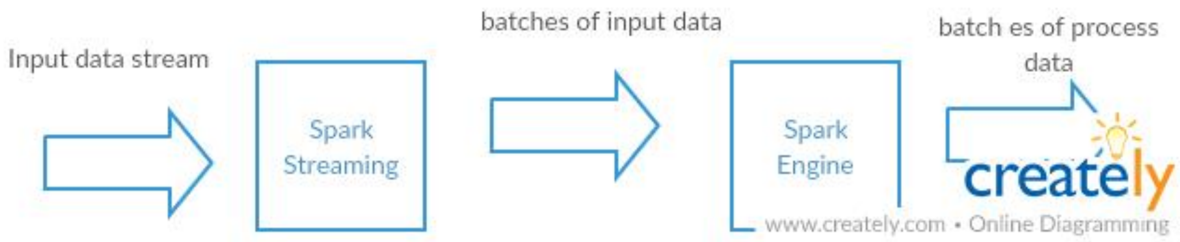
https://github.com/stefanobaghino/spark-twitter-stream-example

Solution:

Twitter Stream is a simple plugin designed to simply show a user's Twitter timeline. It includes file caching to stop overuse of Twitter's API. You can also choose how many updates to return (maximum of 200). It also includes auto linking for URL's found within the timeline.

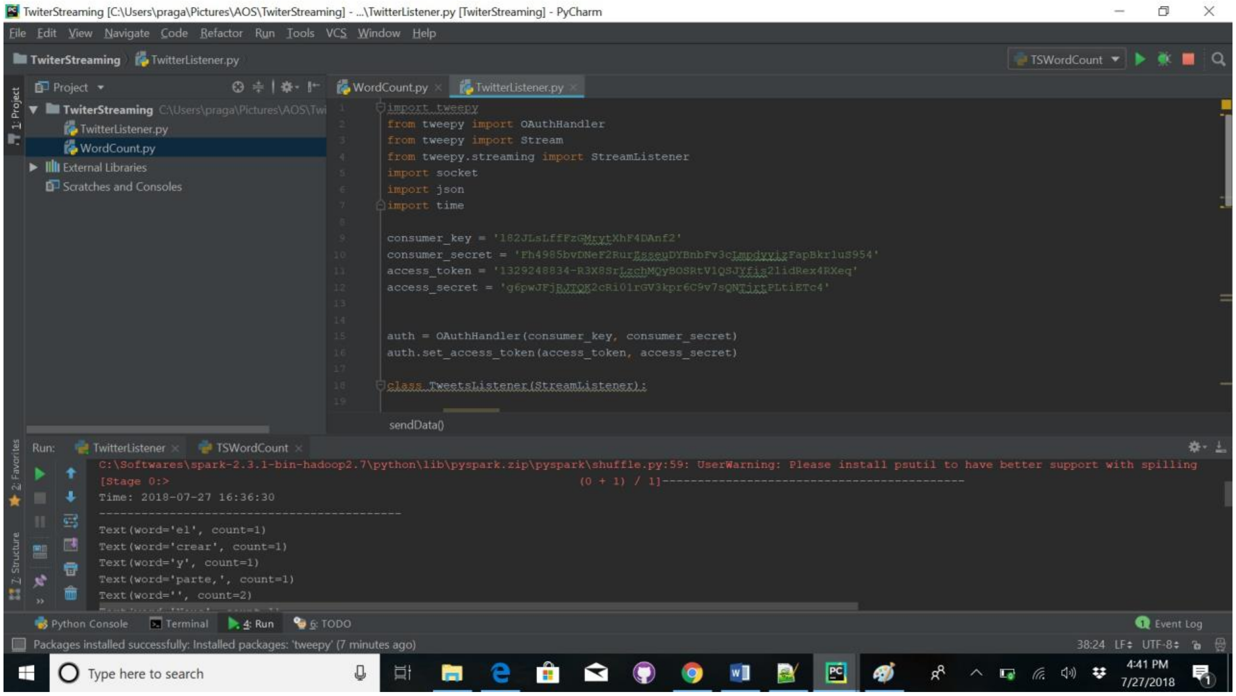
- Firstly, we have created a developer account in Twitter using below link. https://apps.twitter.com/
- Below are the variables that contains the user credentials to access Twitter API

  ➢ API ACCESS_TOKEN = " 1329248834-R3X8SrLzchMQyBOSRtV1QSJYfis2lidRex4RXeq"
  ➢ ACCESS_SECRET = " g6pwJFjRJTQK2cRi01rGV3kpr6C9v7sQNTjrtPLtiETc4"
  ➢ CONSUMER_KEY = " l82JLsLffFzGMrytXhF4DAnf2"
  ➢ CONSUMER_SECRET = " Fh4985bvDNeF2RurZsseuDYBnbFv3cLmpdyyizFapBkr1uS954"
- We have written python program that is used to fetch tweets in JSON format.
- The tweet data is collected and performed word count on the streaming data
  Spark streaming ,the data can be -ingested – from different sources like twitter and can perform high level complex algorithms like queries and the processed data can be pushed out to the file systems.To stream the twitter data twitter.utils contains all the built in functionality -to-stream data from twitter



Code Snippet :

Twitter Streaming

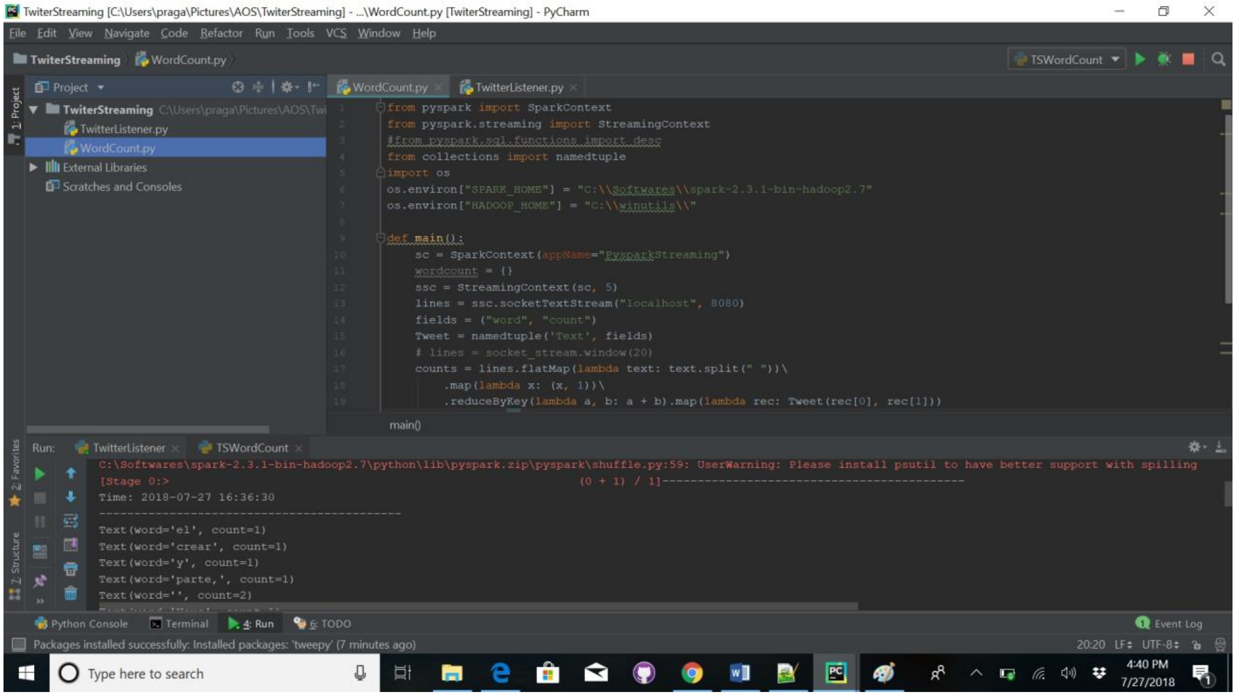


```python
import tweepy
from tweepy import OAuthHandler
from tweepy import Stream
from tweepy.streaming import StreamListener
import socket
import json
import time

consumer_key = '182JLsLffFzGMrytXhF4DAnf2'
consumer_secret = 'Fh4985bvDNeF2RurZsseuDYBnbFv3cLmpdyyizFapBkr1uS954'
access_token = '1329248834-R3X8SrLzchMQyBOSRtV1QSJYfis2lidRex4RXeq'
access_secret = 'g6pwJFjRJTQK2cRi01rGV3kpr6C9v7sQNTjrtPLtiETc4'


auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)

class TweetsListener(StreamListener):

```

```
C:\Softwares\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support with spilling
[Stage 0:>                                                            (0 + 1) / 1]------------------------------------------
Time: 2018-07-27 16:36:30
-------------------------------------------
Text(word='el', count=1)
Text(word='crear', count=1)
Text(word='y', count=1)
Text(word='parte,', count=1)
Text(word='', count=2)
```

Packages installed successfully: Installed packages: 'tweepy' (7 minutes ago)

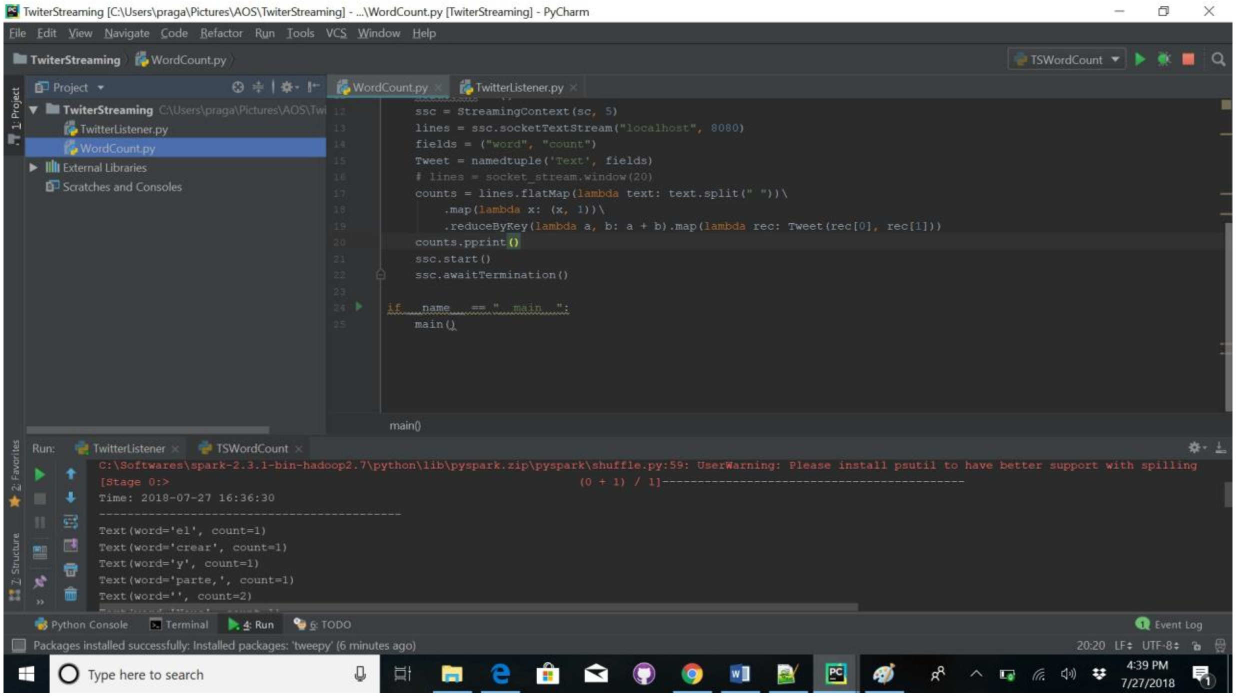
Wordcount


TwiterStreaming [C:\Users\praga\Pictures\AOS\TwiterStreaming] - ...\WordCount.py [TwiterStreaming] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
TwiterStreaming
WordCount.py
TSWordCount
Project
TwiterStreaming C:\Users\praga\Pictures\AOS\Twi
TwitterListener.py
WordCount.py
External Libraries
Scratches and Consoles
WordCount.py
TwitterListener.py
main()
Run:
TwitterListener
TSWordCount
Python Console
Terminal
4: Run
6: TODO
Event Log
Packages installed successfully: Installed packages: 'tweepy' (7 minutes ago)
20:20 LF÷ UTF-8÷
Type here to search
4:40 PM
7/27/2018

```
from pyspark import SparkContext
from pyspark.streaming import StreamingContext
#from pyspark.sql.functions import desc
from collections import namedtuple
import os
os.environ["SPARK_HOME"] = "C:\\Softwares\\spark-2.3.1-bin-hadoop2.7"
os.environ["HADOOP_HOME"] = "C:\\winutils\\"

def main():
    sc = SparkContext(appName="PysparkStreaming")
    wordcount = {}
    ssc = StreamingContext(sc, 5)
    lines = ssc.socketTextStream("localhost", 8080)
    fields = ("word", "count")
    Tweet = namedtuple('Text', fields)
    # lines = socket_stream.window(20)
    counts = lines.flatMap(lambda text: text.split(" "))\
        .map(lambda x: (x, 1))\
        .reduceByKey(lambda a, b: a + b).map(lambda rec: Tweet(rec[0], rec[1]))
```

```
C:\Softwares\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support with spilling
[Stage 0:>                                                          (0 + 1) / 1]------------------------------------------
Time: 2018-07-27 16:36:30
-------------------------------------------

Text(word='el', count=1)
Text(word='crear', count=1)
Text(word='y', count=1)
Text(word='parte,', count=1)
Text(word='', count=2)
```

Output Screen shot :


TwiterStreaming [C:\Users\praga\Pictures\AOS\TwiterStreaming] - ...\WordCount.py [TwiterStreaming] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
TwiterStreaming
WordCount.py
TSWordCount
Project
TwiterStreaming C:\Users\praga\Pictures\AOS\Twi
TwitterListener.py
WordCount.py
External Libraries
Scratches and Consoles
WordCount.py
TwitterListener.py
main()
Run:
TwitterListener
TSWordCount
Python Console
Terminal
4: Run
6: TODO
Event Log
Packages installed successfully: Installed packages: 'tweepy' (6 minutes ago)
20:20 LF÷ UTF-8÷
Type here to search
4:39 PM
7/27/2018

```
    ssc = StreamingContext(sc, 5)
    lines = ssc.socketTextStream("localhost", 8080)
    fields = ("word", "count")
    Tweet = namedtuple('Text', fields)
    # lines = socket_stream.window(20)
    counts = lines.flatMap(lambda text: text.split(" "))\
        .map(lambda x: (x, 1))\
        .reduceByKey(lambda a, b: a + b).map(lambda rec: Tweet(rec[0], rec[1]))
    counts.pprint()
    ssc.start()
    ssc.awaitTermination()

if __name__ == "__main__":
    main()
```

```
C:\Softwares\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support with spilling
[Stage 0:>                                                          (0 + 1) / 1]------------------------------------------
Time: 2018-07-27 16:36:30
-------------------------------------------

Text(word='el', count=1)
Text(word='crear', count=1)
Text(word='y', count=1)
Text(word='parte,', count=1)
Text(word='', count=2)
```

Conclusion:

As stated, the above workflow describes Spark classification task and Spark Streaming Task.

Source code link:

https://github.com/PragathiThammaneni/Bigdata-Programming--Hadoop-Spark/tree/master/Lab%204/Source%20Code

Video Link: Provided in wiki link https://youtu.be/UNqLOIlCUL0

Wiki Link:
https://github.com/PragathiThammaneni/Bigdata-Programming--Hadoop-Spark/wiki/Lab-4-Assignment