**Prac 1** AIM: Install, configure, and run Hadoop and HDFS and explore HDFS

Command 1: sudo apt-get update

Command 2: sudo apt-get install openjdk-8-jdk

Command 3:java -version

Command 4: sudo addgroup Hadoop

Command 5: sudo adduser --ingroup hadoop monika

Command 6: sudo usermod -aG sudo monika

Command 7:su monika

Command 8: ssh-keygen -t rsa -P ""

Command 9: cat $HOME/.ssh/id\_rsa.pub >> $HOME/.ssh/authorized\_keys

Command 10: sudo nano /etc/sysctl.conf

you need to press Ctrl+X, Shift Y and Enter

Now we will download the Hadoop

Command 11: cd /usr/local

Command 12: sudo wget <https://mirrors.estointernet.in/apache/hadoop/common/hadoop3.3.1/hadoop-3.3.1.tar.gz>

Command 13: sudo tar xzf hadoop-3.3.1.tar.gz

Command 14: sudo mv hadoop-3.3.1 hadoop

Command 15: sudo chown -R monika:hadoop hadoop

Command 16: nano $HOME/.bashrc

#Hadoop Related Options

export HADOOP\_HOME=/usr/local/Hadoop

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64

export HADOOP\_INSTALL=$HADOOP\_HOME

export HADOOP\_MAPRED\_HOME=$HADOOP\_HOME

export HADOOP\_COMMON\_HOME=$HADOOP\_HOME

export HADOOP\_HDFS\_HOME=$HADOOP\_HOME

export YARN\_HOME=$HADOOP\_HOME

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_HOME/lib/native

export PATH=$PATH:$HADOOP\_HOME/sbin:$HADOOP\_HOME/bin

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_HOME/lib"

after you add the lines, you need to press Ctrl+X, Shift Y and Enter)

Command 17: source ~/.bashrc

Command 18: cd /usr/local/hadoop/etc/hadoop

Command 19: nano hadoop-env.sh

Add the end of the file(Scroll Down):

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64

after you add the lines, you need to press Ctrl+X, Shift Y and Enter)

Command 20: sudo mkdir -p /app/hadoop/tmp

Command 21: sudo chown monika:hadoop /app/hadoop/tmp

Command 22: nano core-site.xml

Command 23: nano mapred-site.sh

Command 24: nano hdfs-site.xml

Command 25: hadoop namenode -format

Command 26: ssh

Command 27: sudo apt-get install ssh

Command 28: sudo service ssh restart

Command 29: ssh localhost

Command 30: cd /usr/local/hadoop/sbin

Command 31: start-all.sh Command 32: jps

**Prac 2** AIM: Implement word count/frequency programs using MapReduce.

Command : su monika

Command : cd

Command 1: mkdir MapReduce

Command 2: cd MapReduce

Command 3: touch word\_count\_data.txt

Command 4: nano word\_count\_data.txt

(Enter below Text and Press Ctrl+S and then Ctrl+X)

bus car train

ship ship train

bus Ship car

Command 5: cat word\_count\_data.txt

Command 6: touch mapper.py

Command 7: nano mapper.py

(Enter below Code and Press Ctrl+S and then Ctrl+X)

import sys

for line in sys.stdin:

line = line.strip()

words = line.split()

for word in words:

print('%s\t%s' % (word, 1))

For OUTPUT:

Command 8: cat word\_count\_data.txt | python3 mapper.py

Command 9: touch reducer.py

Command 10: nano reducer.py

(Enter below code and Press Ctrl+S and then Ctrl+X)

from operator import itemgetter

import sys

current\_word = None

current\_count = 0

word = None

for line in sys.stdin:

line = line.strip()

word, count = line.split('\t', 1)

try:

count = int(count)

except ValueError:

continue

if current\_word == word:

current\_count += count

else:

if current\_word:

print('%s\t%s' % (current\_word, current\_count))

current\_count = count

current\_word = word

if current\_word == word:

print('%s\t%s' % (current\_word, current\_count))

So now you have your output file in Home & output directory.

Command 13: hdfs dfs -cat /output/op.txt

**Prac 3** : **AIM**: Install MongoDB, Mongo shell and create Database and view it in Mongo Shell.

**Step A**: Install Mongo database

Step 1: Download MongoDB Community Server

<https://fastdl.mongodb.org/windows/mongodb-windows-x86_64-6.0.6-signed.msi>

Step 2: Once the download is complete open the msi file. Click next on the start-up screen. Choose the default settings and keep clicking Next and finally Finish.

Step 3: Once the download is complete, set the path. Go to MyComputer properties -> advanced tab -> environment variables -> new tab of user variable -> write path in variable name -> write path of bin folder in variable value -> ok -> ok -> ok

Step 4: Check for successful installation using the command. Open PowerShell as Administrator (Start menu > PowerShell > right-click > Run as Administrator) Command: mongod –version

**Step B**: Download and Install Mongo shell

Step 1: Download Mongo Shell (https://www.mongodb.com/try/download/shell) and extract it.

Step 2: Go to the bin folder of mongo shell and run the mongosh application. Once you open you’ll be asked to set a connection string. You can define the value as mongodb://localhost:27017/

Step 3: Use the show dbs command to view all the databases

**Step C**: Creating and viewing Database

step 1: Open MongoDb click on + give name to database.

step 2: Open Mongoshell run

show dbs command

**Prac 4 : AIM**: Configure the Hive and implement the application in Hive.

PRE-REQUISITES:

1. Start WSL: Open PowerShell as Administrator (Start menu > PowerShell > right-click > Run as Administrator)

Command: wsl --install

2. Java Installation - Check whether the Java isinstalled or not using the following command. <https://www.java.com/download/ie_manual.jsp>

command: java -version

3. Hadoop must be installed on your system before installing Hive. Let us verify the Hadoop installation using the following

command: hadoop version

Step 1: Start hadoop

Command 1: su monika

Command 2:sudo service ssh restart

Command 3: ssh localhost

Command 4: start-all.sh

Note: If you have preinstalled Hive on yoursystem then uninstall the existing version:

Command 1: cd /usr/local

Command 2: sudo rm -r hive

Command 3: hdfs dfs -rm -r -f /tmp

Command 4: hdfs dfs -rm -r /usr/hive

Step 2: Change the current working directory to / home/monika

Command 5: cd /home/monika

Step 3: Create a text file sample.txt

Command 6:sudo nano sample.txt

Step 4: Add following detailsin the sample.txt1

Sanvi 20000 Lecturer

2 Manoj 25000 Accountant

3 Rahul 30000 IT

Step 5: Save “Ctrl S” and exit the file by pressing “Ctrl X

Download HIVE Now we will download and setup hive:

Change the current working directory to /usr/local

Command 7: cd /usr/local

Command 8: sudo wget

<https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz>

Command 9: sudo tar -xvzf apache-hive-3.1.2-bin.tar.gz

Command 10: ls

Command 11: sudo mv apache-hive-3.1.2-bin hive

Command 12: ls -l

Command 13: sudo chmod 777 hive

Command 14: cd /home/monika

Command 15: nano .bashrc

Command 16: source .bashrc

Command 17: cd /usr/local/hive/bin

Command 18: sudo nano hive-config.sh

V. Create Hive Directories in HDFS:

Command 19: hdfs dfs -mkdir /tmp

Command 20: hdfs dfs -chmod g+w /tmp

Command 21: hdfs dfs -ls /

Command 22: hdfs dfs -mkdir -p /user/hive/warehouse

Command 23: hdfs dfs -chmod g+w /user/hive/warehouse

Command 24: hdfs dfs -ls /user/hive

Command 25: /usr/local/hive/bin/schematool -dbType derby -initSchema

Command 26: ls $HIVE\_HOME/lib

Command 27: ls $HADOOP\_HOME/share/hadoop/hdfs/lib

Command 28: sudo rm /usr/local/hive/lib/guava-19.0.jar

Command 29: sudo cp $HADOOP\_HOME/share/hadoop/common/lib/guava-27.0-jre.jar /usr/local/hive/lib/

Command 30: /usr/local/hive/bin/schematool -initSchema -dbType derby

VII. Launch Hive Client Shell on Ubuntu

Command 31: cd $HIVE\_HOME

Command 32: hive

Command 33: CREATE TABLE IF NOT EXISTS employee ( eid int, name String, salary String,designation String)COMMENT 'Employee details' ROW FORMAT DELIMITED FIELDS TERMINATED BY ' ' LINES TERMINATED BY '\n' STORED AS TEXTFILE;

Command 34: Load data local inpath '/home/monika/sample.txt' overwrite into table employee;

Command 35: select \* from employee;

Command 36: quit;

**Prac 5 :** AIM: Implement an application that stores big data in Pig.

PRE-REQUISITES:

1. owerShell > right-click > Run as Administrator)

2. Check Java & Hadoop Installation –

Command: wsl --install -d ubuntu

command: java -version

command: hadoop version

Step 1:

Start hadoop

Command 1: su monika

Command 2: sudo service ssh restart

Command 3: ssh localhost

Command 4: start-all.sh

Command 5: cd /home/monika

Command 6: sudo nano sample.txt

Add following details in the sample.txt: (comma separated values)

1, Sanvi, 20000, Lecturer

2, Manoj, 25000, Accountant

3, Rahul, 30000, IT

Save “Ctrl S” and exit the file by pressing “Ctrl X

Download Pig

Now we will download and setup Pig:

Command 7: cd /usr/local

Command 8: sudo wget <https://downloads.apache.org/pig/pig-0.17.0/pig-0.17.0.tar.gz>

Command 9: sudo tar -xvzf pig-0.17.0.tar.gz

Command 10: ls

Command 11: sudo mv pig-0.17.0 pig

Command 12: ls

Command 13: sudo chmod 777 pig

Command 14: cd /home/monika

Command 15: nano .bashrc

Append the following Pig environment variables to the .bashrc file:

export PATH=$PATH:$HADOOP\_HOME/bin

export PATH=$PATH:$HADOOP\_HOME/sbin

export PATH=$PATH:/usr/local/pig/bin

export PIG\_HOME=/usr/local/pig

export PIG\_CLASSPATH=/usr/local/hadoop/etc/Hadoop

Save “Ctrl S” and exit the file by pressing “Ctrl X” followed by “Y” and “Enter” keys.

Command 16: source .bashrc

Command 17: pig -version

To start Pig in Cluster Mode, execute the below command:

Command 18: pig

Command 19: Quit

Command 20: pig -x local

Command 21: Employees = LOAD 'sample.txt' USING PigStorage(',') as (id:int, firstname:chararray, salary:int, dept:chararray);

Command 22: Dump Employees;

Command 23: Employees\_Order = ORDER Employees BY salary DESC;

Command 24: Dump Employees\_Order ;

Command 25: quit

**Prac 6 Aim**: Implement the Decision Tree classification technique.

command 1:Import pandas as pd

Import numpy as np

From sklearn.datasets import load\_iris

From sklearn.metrics import accuracy\_score

From matplotlib import pyplot as plt

From sklearn import datasets

From sklearn import tree

command 2:iris\_data = load\_iris()

iris=pd.DataFrame(iris\_data.data)

command 3:print(“Features Name : “, iris\_data.feature\_names)

command 4:x = iris\_data.data

print(x)

command 5:y = iris\_data.target

print(Y)

command 6:from sklearn.model\_selection import train\_test\_split

command 7:x\_train, x\_test, y\_train,y\_test = train\_test\_split(x,y, random\_state = 50,test\_size = 0.3)

command 8:print(x\_train.shape)

print(x\_test.shape)

print(y\_train.shape)

print(y\_test.shape)

command 9:from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier(random\_state=100)

clf.fit(x\_train,y\_train)

command 10:y\_pred = clf.predict(x\_test)

print(Y\_pred)

command 11:print(“Accuracy:”,accuracy\_score(y\_test, y\_pred))

command 12:from sklearn.metrics import confusion\_matrix

cm=np.array(confusion\_matrix(y\_test,y\_pred))

cm

command 13:from sklearn.metrics import plot\_confusion\_matrix

plot\_confusion\_matrix(clf,x\_test,y\_test)

plt.show()

command 14: tree.plot\_tree(clf)

**Prac 7** Aim: Implement SVM Classification Technique.

Command 1 : Import numpy as np

Import pandas as pd

From sklearn import datasets

Command 2 : iris = datasets.load\_iris()

Command 3 : iris[“feature\_names”]

Command 4 : df = pd.DataFrame(iris[“data”], columns = iris[“feature\_names”])

Command 5 : df.head()

Command 6 : x = df

y = iris[“target”]

Command 7 :print(x)

Command 8 : Print(y)

Command 9 : from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train,y\_test = train\_test\_split(x,y, test\_size = 0.3,random\_state = 0)

Command 10 : from sklearn.preprocessing import standardScaler

Sc = StandardScalar()

x\_train = sc.fit\_trainsform(x\_train)

x\_test = sc.transform(x\_test)

Command 11: from sklearn.svm import svc

svc\_clf = SVC(kernel = ‘linear’, random\_state = 0)

svc\_clf.fit(x\_train, y\_train)

Command 12 : y\_pred = svc\_clf.predict(x\_test)

Print(y\_pred)

Command 13: from sklearn.svm import SVC

svcclassifier = SVC(kernel = ‘poly’, random\_state = 0)

svcclassifier.fit(x\_train,y\_train)

Command 14: y\_pred = svc\_clf.predict(x\_test)

Print(y\_pred)

Command 15: from sklearn.svm import SVC

svcclassifier = SVC(kernel = ‘rbf’, random\_state = 0)

svcclassifier.fit(x\_train,y\_train)

Command 16: y\_pred = svc\_clf.predict(x\_test)

Print(y\_pred)

Command 17 : from sklearn.metrics import mean\_squared\_error

accuracy = svc\_clf.score(x\_test,y\_test)\*100

print(“Accuracy: ”,accuracy)

**Prac 8** Aim: Implement Logistic Regression on Iris Dataset

import numpy as np

import pandas as pd

from sklearn import datasets

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import mean\_squared\_error

iris = datasets.load\_iris()

df = pd.DataFrame(iris["data"],columns = iris["feature\_names"])

df.head()

df.describe()

x = df

y = iris["target"]

print(x)

print(y)

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size =

0.3,random\_state = 0)

model = LogisticRegression()

# Training model

model.fit(x\_train, y\_train)

# Testing model

y\_pred = model.predict(x\_test)

y\_pred

accuracy = model.score(x\_test,y\_test)\*100

accuracy

# mean squared error

mse=mean\_squared\_error(y\_test, y\_pred)

# Root mean squared error

rmse=np.sqrt(mse)

rmse

**Prac 9** : Aim: Multiple Linear Regression using the Diabetes Dataset

Command 1 : Import numpy as np

Import matplotlib.pyplot as plt

Import pandas as pd

From sklearn import metrics

Command 2 : datasets = pd.read\_csv(‘50\_startups.csv’)

x = dataset.iloc[:,:-1].values

y = dataset.iloc[:,-1].values

Command 3: print(x)

Command 4 : print(y)

Command 5 : from sklearn.compose import ColumnTransformer

From sklearn.preprocessing import OneHotEncoder

Ct = ColumnTransformer(transformers=[(‘encoder’, OneHotEncoder(),[3])], remainder=’passthrough’)

X = np.array(ct.fit\_transform(x))

Command 6: print(x)

Command 7 : from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train,y\_test = train\_test\_split(x,y,test\_size = 0.2, random\_state = 0)

Command 8 : from sklearn.linear\_model import LinearRegression

Regressor = LinearRegression()

Regressor.fit(x\_train, y\_train)

Command 9: y\_pred = regressor.predict(x\_test)

Print(“prediction {}:”.format(y\_train)

Command 10: mlr\_diff = pd.DataFrame({‘Actual value’: y\_test, ‘predicted value’: y\_pred})

Mlr\_diff.head()

Command 11 : print(‘R squared: {: .2f}’ .format(regressor.score(x,y)\*100))

Print(‘Mean Absolute Error:’ , metrics.mean\_absolute\_error(y\_test, y\_pred))

Print(‘Mean Squared Error:’, metrics.mean\_squared\_error(Y\_test,y\_pred))

Print(‘Root Mean Squared Error’, metrics.mean\_squared\_error(y\_test,Y\_pred, squared =False))

**Prac 10**

Aim: Implement K-Means Clustering using Iris Datasets

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from sklearn import datasets

iris = datasets.load\_iris()

x = iris.data

y = iris.target

print(x)

print(y)

from sklearn.cluster import KMeans

kmeans = KMeans(n\_clusters = 3, init = 'k-means++', max\_iter = 100, n\_init = 10, random\_state = 0) y\_kmeans = kmeans.fit\_predict(x)

y\_kmeans

plt.scatter(x[y\_kmeans == 0, 0], x[y\_kmeans == 0, 1], c = 'red', label = 'setosa')

plt.scatter(x[y\_kmeans == 1, 0], x[y\_kmeans == 1, 1], c = 'blue', label = 'versicolour') plt.scatter(x[y\_kmeans == 2, 0], x[y\_kmeans == 2, 1], c = 'green', label = 'virginica')

#Plotting the centroids of the clusters

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:,1], s = 100, c = 'yellow', label = 'Centroids')

plt.legend()