PRACTICAL REPORT

ON

PPSIT2P1: BIG DATA ANALYTICS

SUBMITTED BY

MANSI M. BAIKAR

ROLL NO : 01

SUBMITTED TO

Ms. RASIKA SAWANT

MSc. (INFORMATION TECHNOLOGY) SEM – II

2022 – 2023

CONDUCTED AT

CHIKITSAK SAMUHA'S

S. S. & L.S. PATKAR COLLEGE OF ARTS & SCIENCE

AND

V. P. VARDE COLLEGE OF COMMERCE & ECONOMICS

An Autonomous college,

Affiliated to University of Mumbai

GOREGAON (W). MUMBAI -400062

CHIKITSAK SAMUHA'S

SIR SITARAM & LADY SHANTABAI PATKAR COLLEGE OF ARTS & SCIENCE

&

V.P. VARDE COLLEGE OF COMMERCE & ECONOMICS

GOREGAON (WEST), MUMBAI - 400 104.

An Autonomous College, University of Mumbai

CERTIFICATE

Certified that such of the experiments as have been duly signed were performed by Mr./Miss _____

Roll No. _____ *of* _____ *class* _____

Division _____ *in the* _____ *Laboratory of this college during the year* _____

Professor-in-Charge **Examiner** **Co-ordinator**

Date: _____ _____ *Department*

INDEX

Practical No	Practical Aim	Date	Sign
1	Solve the following:	3/1/2023	
A	REGRESSION MODEL Import a data from web storage. Name the dataset and now do Logistic Regression to find out relation between variables that are affecting the admission of a student in an institute based on his or her GRE score, GPA obtained and rank of the student. Also check the model is fit or not. require (foreign), require(MASS).		
B	LINEAR REGRESSION MODEL Apply multiple regressions, if data have a continuous independent variable. Apply on above dataset.		
2	Implement the following:	11/1/2023	
A	Implement Decision tree classification techniques		
B	Implement SVM classification techniques		
3	Install, configure, and run Hadoop and HDFS adexplore HDFS	2/3/2023	
4	Implement word count / frequency programs using MapReduce.	2/3/2023	
5	Implement an application that stores big data in MongoDB and manipulate it using R / Python	24/3/2023	
6	Solve the Following:	20/3/2023	
A	CLASSIFICATION MODEL a. Install relevant package for classification. b. Choose classifier for classification problem. c. Evaluate the performance of classifier		
B	CLUSTERING MODEL a. Clustering algorithms for unsupervised classification. b. Plot the cluster data using R visualizations.		
7	Configure the Hive and implement the application in Hive.	21/3/2023	
8	Implement an application that stores big data in Pig.	27/3/2023	

Practical No : 01

A] Logistic Regression Model

Aim: Import data from web storage. Name the dataset and now do Logistic Regression to find out relation between variables that are affecting the admission of a student in an institute based on his or her GRE score, GPA obtained and rank of the student. Also check if the model is fit or not. require (foreign), require(MASS).

Theory:

WHAT ARE REGRESSION MODELS?

- Regression deals with numerical target values.
- Regression model helps predict the numerical value of a target value based on the training dataset.

WHAT IS LOGISTIC REGRESSION?

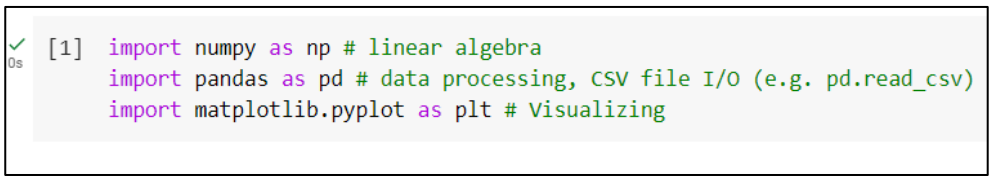
- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- Logistic Regression is much similar to Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

OBJECTIVE: In this practical the goal is to predict whether a student can enter his/her desired university or not.

The data set contains the information of students with 4 attributes(columns) including the Graduate Record Examinations (GRE), the Grade Point Average(GPA) scores, rank of the student, and also whether the student has at least one research or not.

1. Importing Basic Libraries.

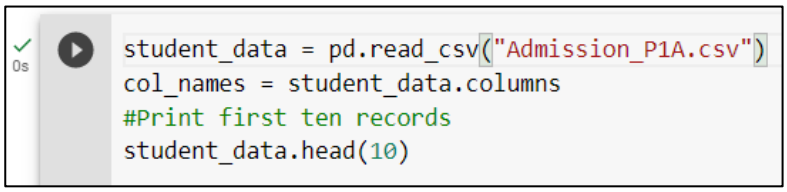
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt # Visualizing
```



```
[1]  import numpy as np # linear algebra
     import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
     import matplotlib.pyplot as plt # Visualizing
```

2. Importing Dataset.

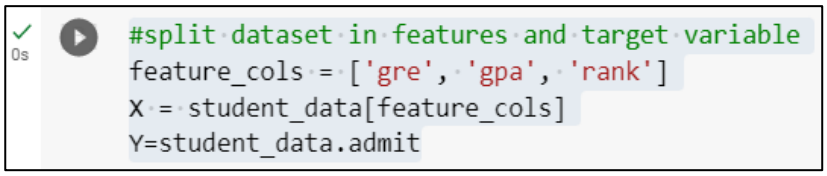
```
student_data = pd.read_csv("Admission_P1A.csv")
col_names = student_data.columns
#Print first ten records
student_data.head(10)
```



	admit	gre	gpa	rank
0	0	380	3.61	3
1	1	660	3.67	3
2	1	800	4.00	1
3	1	640	3.19	4
4	0	520	2.93	4
5	1	760	3.00	2
6	1	560	2.98	1
7	0	400	3.08	2
8	1	540	3.39	3
9	0	700	3.92	2

3. Split the dataset into features and target variables.

```
#split dataset in features and target variable
feature_cols = ['gre', 'gpa', 'rank']
X = student_data[feature_cols]
Y=student_data.admit
```



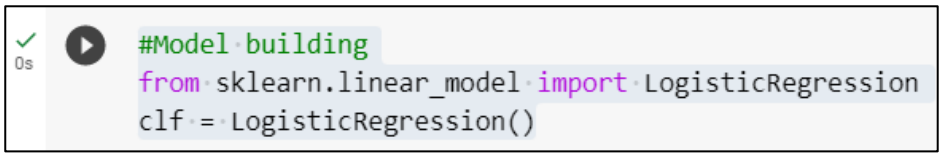
4. Split the data into Training and Testing dataset.

```
from sklearn.model_selection import train_test_split
# 70% training and 30% test
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_s
tate=1)
```

```
from sklearn.model_selection import train_test_split
# 70% training and 30% test
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=1)
```

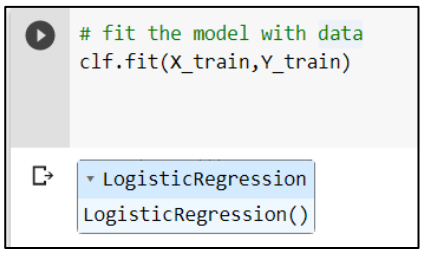
5. Model Building

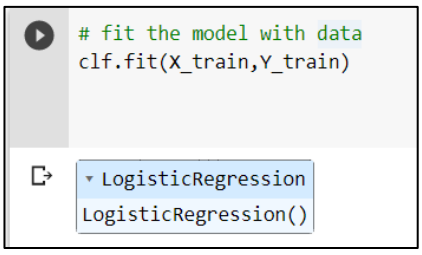
```
#Model building
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression()
```



6. Fit the Model.

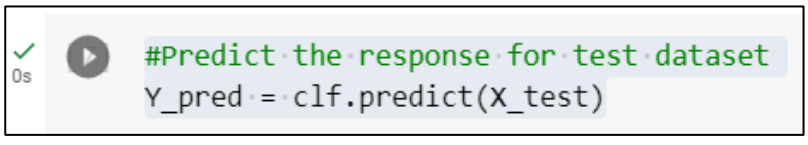
```
 # fit the model with data
clf.fit(X_train,Y_train)
# Train Decision Tree Classifer
clf = clf.fit(X_train,Y_train)
```





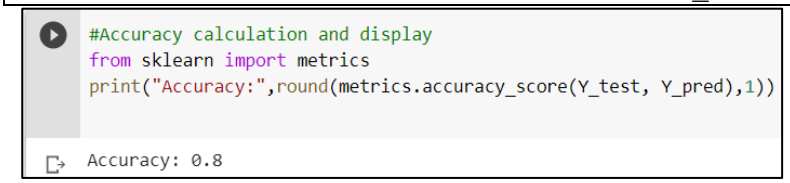
7. Predict the response for the test dataset.

```
#Predict the response for test dataset
Y_pred = clf.predict(X_test)
```



8. Calculate Accuracy of Model.

```
#Accuracy calculation and display
from sklearn import metrics
print("Accuracy:",round(metrics.accuracy_score(Y_test, Y_pred),1))
```



9. Prediction

```
# Prediction
new={'gre':[260],'gpa':[2.67],'rank':[1] }
sc2 = pd.DataFrame(new,columns= ['gre','gpa','rank'])
Y_pred=clf.predict(sc2)
print (sc2)
print ("Forecast is:",Y_pred)
```

```
# Prediction
new={'gre':[260],'gpa':[2.67],'rank':[1] }
sc2 = pd.DataFrame(new,columns= ['gre','gpa','rank'])
Y_pred=clf.predict(sc2)
print (sc2)
print ("Forecast is:",Y_pred)
```

```
   gre   gpa  rank
0  260  2.67     1
Forecast is: [0]
```

Conclusion: Forecast 0 indicates that students with 260 gre 2.67 gpa and 1 rank in undergrad are less likely to gain admission to graduate programs.

B] LINEAR REGRESSION MODEL

Aim: Apply linear regressions, if data have a continuous independent variable. Apply on the above dataset.

Theory:

WHAT IS SIMPLE LINEAR REGRESSION?

- Simple Linear Regression basically defines the relation between a one feature and the outcome variable.
- This can be specified using the formula $y = \alpha + \beta x$ which is similar to the slope-intercept form, where y is the value of the dependent variable, α is the intercept β denotes the slope and x is the value of the independent variable.
- Suppose we are given the value of the independent variable x, the regression model will compute the value of α and β such that the absolute difference between the predicted y value and actual y value is minimal.
- As a result of the difference between the predicted y value and actual y value , we will be able to understand whether our model is performing well or needs fine tuning.

OBJECTIVE: Predicting the Chances of getting admitted to the graduate school based on the GRE Score.

The dataset version that has been used here is Admission_Predict.csv The dataset contains several parameters which are considered important during the application for master's Programs. The parameters included are: GRE Scores (out of 340) TOEFL Scores (out of 120) University Rating (out of 5) Statement of Purpose and Letter of Recommendation Strength (out of 5) Undergraduate GPA (out of 10) Research Experience (either 0 or 1) Chance of Admit (ranging from 0 to 1) Our goal here would be to predict the "Chance of Admit" based on the different parameters that are provided

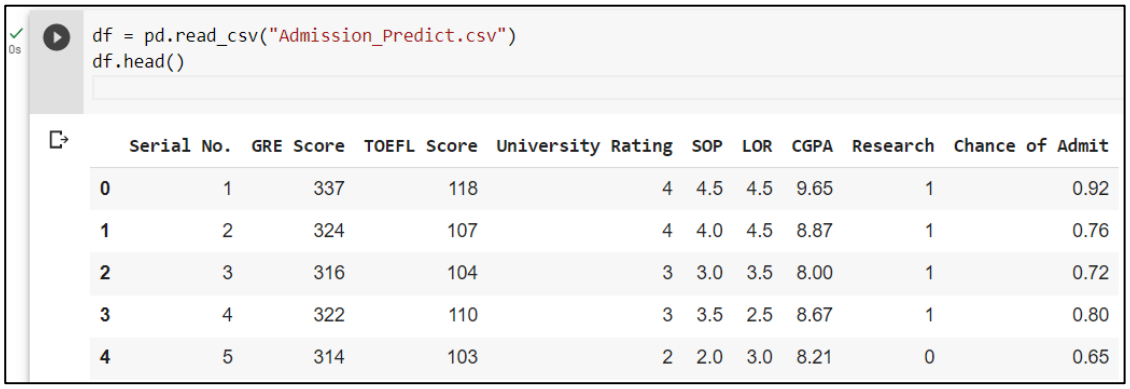
in the dataset. We will achieve this goal by using the Simple Linear Regression model. Based on the data that we have; we will split out data into training and testing sets. The Training set will have features and labels on which our model would be trained. The label here is the "Chance of Admit". If you think from a non-technical standpoint then labels are basically the output that we want, and features are the parameters that drive us towards the output. Once our model is trained, we will use the trained model and run it on the test set and predict the output. Then we will compare the predicted results with the actual results that we have to see how our model performed.

This whole process of training the model using features and known labels and later testing it to predict the output is called Supervised Learning.

1. Import the Data.

```
import pandas as pd
```


```
df = pd.read_csv("Admission_Predict.csv")
df.head()
```



	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65

2. Explore the Data.

```
df.info()
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   serial_no.         400 non-null    int64  
 1   gre_score          400 non-null    int64  
 2   toefl_score        400 non-null    int64  
 3   university_rating  400 non-null    int64  
 4   sop                400 non-null    float64
 5   lor                400 non-null    float64
 6   cgpa               400 non-null    float64
 7   research           400 non-null    int64  
 8   chance_of_admit    400 non-null    float64
dtypes: float64(4), int64(5)
memory usage: 28.2 KB
```

3. Format the Column Names a bit by making all the Column Names to Lowercase and Removing the spaces between the Column Names by adding "_" and also replacing any ")" or "(" with no spaces.

```
df.columns = df.columns.str.strip().str.lower().str.replace(' ', '_').str.replace('(', '').str.replace(')', '')
```

```
<ipython-input-3-c24f79c43405>:1: FutureWarning: The default value of regex will change from True to False in a future version. In addition, sing
  df.columns = df.columns.str.strip().str.lower().str.replace(' ', '_').str.replace('(', '').str.replace(')', '')
<ipython-input-3-c24f79c43405>:1: FutureWarning: The default value of regex will change from True to False in a future version. In addition, sing
  df.columns = df.columns.str.strip().str.lower().str.replace(' ', '_').str.replace('(', '').str.replace(')', '')
```

```
df.info()
```

```
df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   serial_no.         400 non-null    int64  
 1   gre_score          400 non-null    int64  
 2   toefl_score        400 non-null    int64  
 3   university_rating  400 non-null    int64  
 4   sop                400 non-null    float64
 5   lor                400 non-null    float64
 6   cgpa               400 non-null    float64
 7   research           400 non-null    int64  
 8   chance_of_admit    400 non-null    float64
dtypes: float64(4), int64(5)
memory usage: 28.2 KB
```

Note that in our data set we don't have any categorical values. Categorical values are mostly multiclass in a dataset. Firstly, while building a model, categorical variables should be converted into dummy variables through a dummy encoding process for better accuracy. The reason is most of the machine learning algorithms perform matrix operations for model evaluation and with categorical variables the results won't be accurate.

```
df_dummies = pd.get_dummies(df, drop_first=True)
df_dummies.head()
```

```
df_dummies = pd.get_dummies(df, drop_first=True)
df_dummies.head()
```

	serial_no.	gre_score	toefl_score	university_rating	sop	lor	cgpa	research	chance_of_admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65

4. Remove any Possible Order Effects in the Data by shuffling the rows of the Data before splitting the data into Features(X) and Dependent Variables(Y).

```
from sklearn.utils import shuffle
df_shuffled = shuffle(df_dummies,random_state = 44)
df_shuffled.head()
```

```
from sklearn.utils import shuffle
df_shuffled = shuffle(df_dummies,random_state = 44)
df_shuffled.head()
```

	serial_no.	gre_score	toefl_score	university_rating	sop	lor	cgpa	research	chance_of_admit
201	202	315	110	2	3.5	3.0	8.46	1	0.72
258	259	326	102	4	5.0	5.0	8.76	1	0.77
114	115	311	105	3	3.5	3.0	8.45	1	0.59
270	271	306	105	2	2.5	3.0	8.22	1	0.72
7	8	308	101	2	3.0	4.0	7.90	0	0.68

5. Split the data into Features (X) and Dependent Variables (Y).

```
DV = 'chance_of_admit'
X = df_shuffled.drop(DV,axis=1)
y = df_shuffled[DV]
```

```
DV = 'chance_of_admit'
X = df_shuffled.drop(DV,axis=1)
y = df_shuffled[DV]
```

6. Split the data into a Training and Testing Set. The testing size that we are taking here is 30% of the total size of the Dataset.

```
from sklearn.model_selection import train_test_split
```

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.33, random
_state = 42)
X_train.head()
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.33, random_state = 42)
X_train.head()
```

	serial_no.	gre_score	toefl_score	university_rating	sop	lor	cgpa	research
346	347	304	97	2	1.5	2.0	7.64	0
391	392	318	106	3	2.0	3.0	8.65	0
385	386	335	117	5	5.0	5.0	9.82	1
202	203	340	120	5	4.5	4.5	9.91	1
192	193	322	114	5	4.5	4.0	8.94	1

```
y_train.head()
```

```
y_train.head()

346    0.47
391    0.71
385    0.96
202    0.97
192    0.86
Name: chance_of_admit, dtype: float64
```

7. Fit the Simple Linear Regression Model and Determine the Intercept and Coefficient.

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

8. Fit the model to the "gre_score" Column of the Training data which is the feature column and extract the intercept and coeff.

```
model.fit(X_train[['gre_score']],y_train)
```

```
model.fit(X_train[['gre_score']],y_train)

▾ LinearRegression
LinearRegression()
```

9. Extract the value of the Intercept using the below.

```
intercept = model.intercept_
print("Intercept: ",intercept)

coeff = model.coef_
print("Coefficient: ",coeff)
```

```
intercept = model.intercept_
print("Intercept: ",intercept)

coeff = model.coef_
print("Coefficient: ",coeff)
```

```
Intercept:  -2.3144531329626408
Coefficient:  [0.00959563]
```

10. Print out the whole formula which will be used for predicting the "chance_of_admit".

```
print('Chance of Admit = {0:0.2f} + ({1:0.2f} x gre_score)'.format(intercept,
coeff[0]))
```

```
print('Chance of Admit = {0:0.2f} + ({1:0.2f} x gre_score)'.format(intercept,coeff[0]))
```

```
Chance of Admit = -2.31 + (0.01 x gre_score)
```

11. Generate the prediction on the test data.

```
y_pred = model.predict(X_test[['gre_score']])
print("Predictions: ",y_pred)
```

```
y_pred = model.predict(X_test[['gre_score']])
print("Predictions: ",y_pred)
```

```
Predictions:  [0.59302133 0.67938196 0.90967696 0.71776446 0.90967696 0.67938196
 0.94805947 0.63140383 0.89048571 0.53544758 0.64099946 0.79452946
 0.90967696 0.71776446 0.56423445 0.57383008 0.74655133 0.76574259
 0.74655133 0.71776446 0.92886822 0.75614696 0.55463883 0.5834257
 0.70816883 0.66978633 0.77533821 0.72736008 0.78493384 0.68897758
 0.63140383 0.82331634 0.82331634 0.55463883 0.69857321 0.80412509
 0.59302133 0.80412509 0.79452946 0.71776446 0.77533821 0.65059508
 0.83291196 0.65059508 0.53544758 0.83291196 0.73695571 0.69857321
 0.61221258 0.61221258 0.64099946 0.74655133 0.64099946 0.84250759
 0.88089009 0.71776446 0.76574259 0.93846384 0.56423445 0.66978633
 0.84250759 0.66019071 0.64099946 0.80412509 0.81372071 0.72736008
 0.66978633 0.79452946 0.62180821 0.89048571 0.89048571 0.75614696
 0.73695571 0.70816883 0.83291196 0.72736008 0.66019071 0.76574259
 0.80412509 0.79452946 0.62180821 0.73695571 0.82331634 0.67938196
 0.85210321 0.66978633 0.94805947 0.72736008 0.87129446 0.81372071
 0.75614696 0.68897758 0.61221258 0.90967696 0.5066607  0.65059508
 0.79452946 0.5450432  0.56423445 0.53544758 0.84250759 0.68897758
 0.66978633 0.89048571 0.63140383 0.71776446 0.76574259 0.76574259
 0.70816883 0.60261696 0.86169884 0.63140383 0.66978633 0.68897758
 0.68897758 0.61221258 0.77533821 0.5834257  0.80412509 0.67938196
 0.5450432  0.70816883 0.67938196 0.86169884 0.66978633 0.86169884
 0.76574259 0.64099946 0.77533821 0.78493384 0.69857321 0.76574259]
```

12. Compare the Actual Chance of Admit Vs The Predicted Chance of Admit

```
df1 = pd.DataFrame({'Chance of Admit Actual value': y_test, 'Chance of Admit Predicted value': y_pred})
df1.head()
```

```
df1 = pd.DataFrame({'Chance of Admit Actual value': y_test, 'Chance of Admit Predicted value': y_pred})
df1.head()
```

	Chance of Admit Actual value	Chance of Admit Predicted value
19	0.62	0.593021
80	0.50	0.679382
71	0.96	0.909677
294	0.61	0.717764
372	0.95	0.909677

```
from sklearn import metrics
import numpy as np
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 :       ",np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
print("Mean Absolute Percentage Error: ",metrics.mean_absolute_percentage_error(y_test,y_pred))
```

```
print("R-
squared:                    ",metrics.explained_variance_score(y_test,y_pre
d))
```

```
Mean Absolute Error :              0.06315764414298021
Mean Squared Error :               0.007059203468378346 5
Root Mean Squared Error :          0.08401906610037002
Mean Absolute Percentage Error:    0.1001757190276413
R-squared:                         0.6621061878477427
```

- **Mean Absolute Error** is the absolute difference between the predicted value (predictions) and the actual value(y_test).

- **Mean squared error (MSE)** is the average of the squared differences between the predicted and actual values.

- **Root mean squared error (RMSE)** is the square root of the MSE.

- **R-squared** is a statistical measure which says how close the data are to the fitted regression line.R-squared is always between 0 and 100%: For instance, R-Squared value of 0% indicates that the model explains none of the variability of the response data around its mean. Similarly, 100% indicates that the model explains all the variability of the response data around its mean.

Conclusion:

In conclusion the R-squared value for the Simple Linear Regression was 66.8% which explains that the “gre_score” explained 66.8% of variance in the “chance_of_admit”. As a result of this, it is evident that our model still needs improvement. And also the predictions were within the range of ±0.64 unit.

Practical No: 02

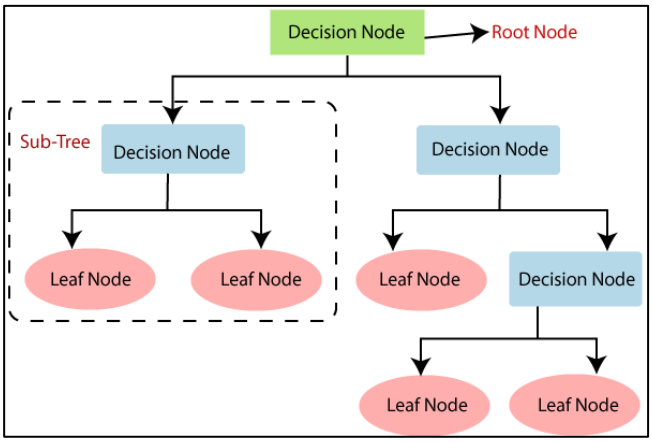
A] Aim: Implement Decision tree classification techniques.

Theory:

Decision Tree

A decision tree is a flowchart-like tree structure where an internal node represents a feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome.

The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in a recursive manner called recursive partitioning. This flowchart-like structure helps you in decision-making. It's visualised like a flowchart diagram which easily mimics human level thinking. That is why decision trees are easy to understand and interpret.



- **How Does the Decision Tree Algorithm Work?**

The basic idea behind any decision tree algorithm is as follows:

1. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2. Make that attribute a decision node and break the dataset into smaller subsets.
3. Start tree building by repeating this process recursively for each child until one of the conditions will match:

- All the tuples belong to the same attribute value.
- There are no more remaining attributes.
- There are no more instances.

Dataset: Pima Indian Diabetes dataset

The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage. The datasets consist of several medical predictor variables and one target variable, Outcome. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

1. Importing Required Libraries.

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split fun
ction
from sklearn import metrics #Import scikit-
learn metrics module for accuracy calculation
```

```
# Load libraries
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
```

2. Loading Data.

```
col_names = ['pregnant', 'glucose', 'bp', 'skin', 'insulin', 'bmi', 'pedigree
', 'age', 'label']
# load dataset
pima = pd.read_csv("diabetes.csv", names=col_names,header=1)
```

```
col_names = ['pregnant', 'glucose', 'bp', 'skin', 'insulin', 'bmi', 'pedigree', 'age', 'label']

# load dataset
pima = pd.read_csv("diabetes.csv", names=col_names,header=1)
```

```
pima.head()
```

```
pima.head()
```

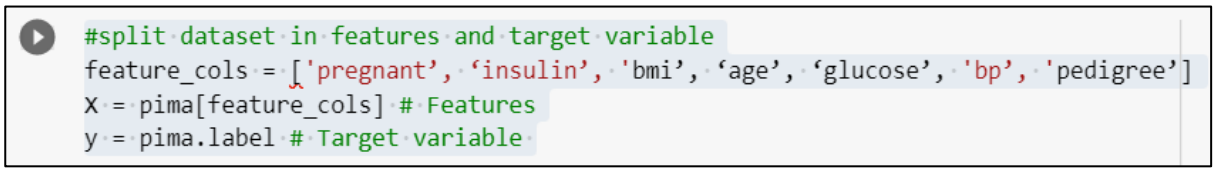
	pregnant	glucose	bp	skin	insulin	bmi	pedigree	age	label
0	1	85	66	29	0	26.6	0.351	31	0
1	8	183	64	0	0	23.3	0.672	32	1
2	1	89	66	23	94	28.1	0.167	21	0
3	0	137	40	35	168	43.1	2.288	33	1
4	5	116	74	0	0	25.6	0.201	30	0

3. Feature Selection

Feature Selection Divide given columns into two types of variables dependent(or target variable) and independent variable(or feature variables).

```
#split dataset in features and target variable
```

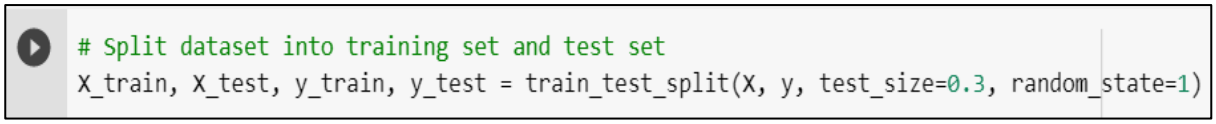
```
feature_cols = ['pregnant', 'insulin', 'bmi', 'age', 'glucose', 'bp', 'pedigree']
X = pima[feature_cols] # Features
y = pima.label # Target variable
```



4. Splitting Dataset

To understand model performance, dividing the dataset into a training set and a test set is a good strategy. Let's split the dataset by using the function train_test_split(). You need to pass 3 parameters: features, target, and test_set size.

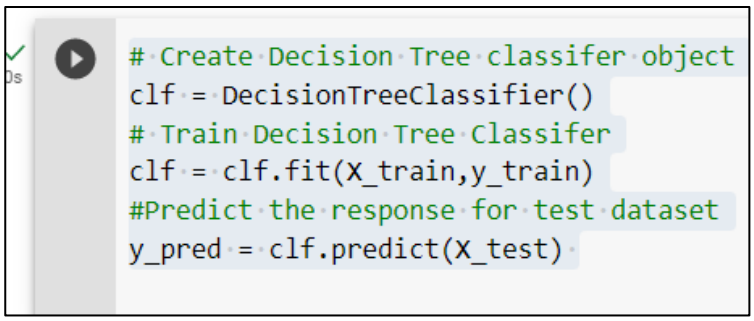
```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```



5. Building Decision Tree Model.

DecisionTreeClassifier is a class capable of performing multi-class classification on a dataset.

```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()
# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)
#Predict the response for test dataset
y_pred = clf.predict(X_test)
```



6. Evaluating Model

Let's estimate how accurately the classifier or model can predict the type of cultivars. Accuracy can be computed by comparing actual test set values and predicted values.

```
# Model Accuracy, how often is the classifier correct?
```

```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.70995670995671
```

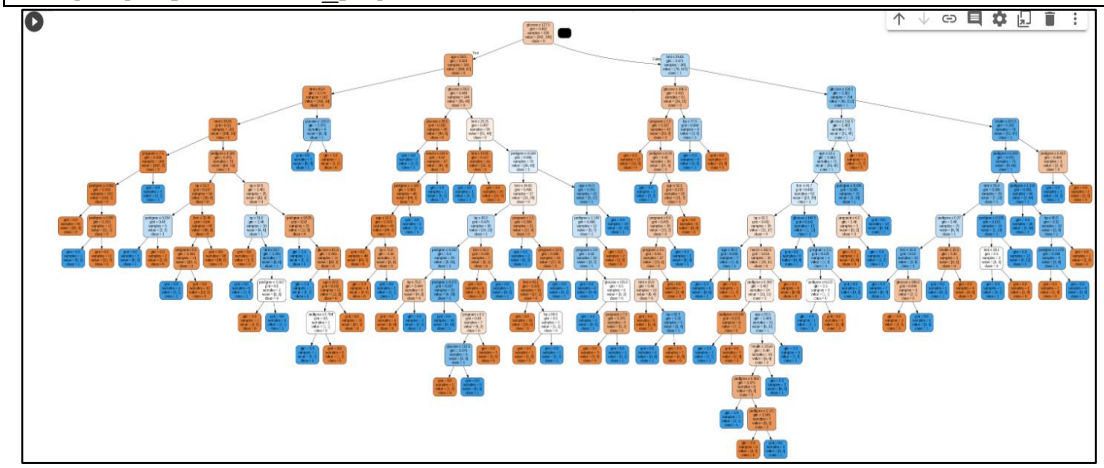
7. Visualizing Decision Trees.

Scikit-learn's export_graphviz function for displaying the tree within a notebook. For plotting trees, install graphviz and pydotplus.

```
pip install graphviz
pip install pydotplus
```

export_graphviz function converts decision tree classifier into dot file and pydotplus convert this dot file to png or displayable form.

```
from sklearn.tree import export_graphviz
from six import StringIO
from IPython.display import Image
import pydotplus
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data, filled=True, rounded=True,special_cha
racters=True,
feature_names = feature_cols,
class_names=['0','1’])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())
```

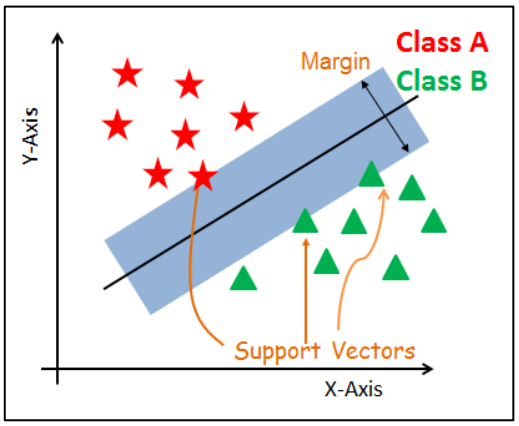


2B] Aim: Implement SVM classification techniques.

Theory:

Support Vector Machines

Generally, Support Vector Machines is considered to be a classification approach, but can be employed in both types of classification and regression problems. It can easily handle multiple continuous and categorical variables. SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplanes in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane(MMH) that best divides the dataset into classes.



Support Vectors

Support vectors are the data points, which are closest to the hyperplane. These points will define the separating line better by calculating margins. These points are more relevant to the construction of the classifier.

Hyperplane

A hyperplane is a decision plane which separates between a set of objects having different class memberships.

Margin

A margin is a gap between the two lines on the closest class points. This is calculated as the perpendicular distance from the line to support vectors or closest points. If the margin is larger in between the classes, then it is considered a good margin, a smaller margin is a bad margin.

- **How does SVM work?**

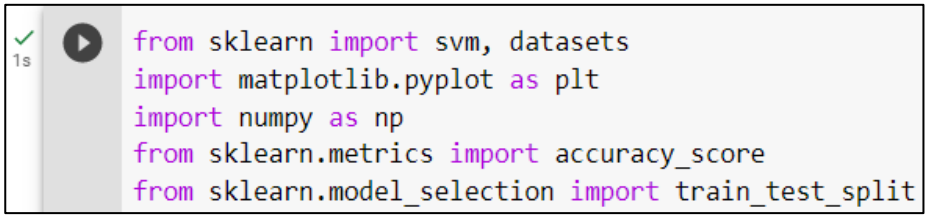
The main objective is to segregate the given dataset in the best possible way. The distance between the nearest points is known as the margin. The objective is to select a hyperplane with the maximum possible margin between support vectors in the given dataset. SVM searches for the maximum marginal hyperplane in the following steps:

- Generate hyperplanes which segregate the classes in the best way. Left-hand side figure showing three hyperplanes black, blue and orange. Here, the blue and orange have higher classification error, but the black is separating the two classes correctly.
- Select the right hyperplane with the maximum segregation from the nearest data points as shown in the right-hand side figure.

Program on SVM for performing classification and finding its accuracy on the given data:
1. Import libraries.

1. We import svm and datasets from the sklearn Library
2. Numpy for carrying out efficient mathematical computations
3. accuracy_score from sklearn.metrics to predict the accuracy of the model
4. from sklearn.model_selection import train_test_split for splitting the data into a training set and testing set.

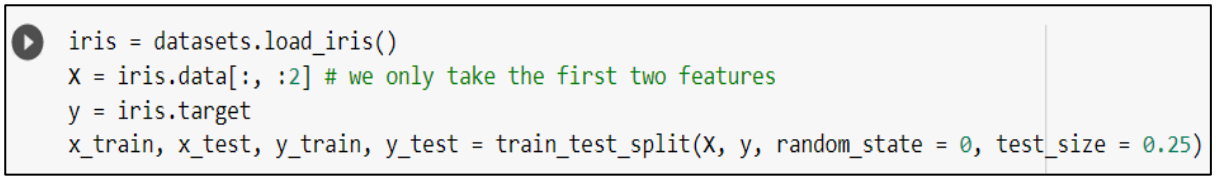
```
from sklearn import svm, datasets
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
```



2. Add datasets, insert the desired number of features and train the model.

Here is the code for importing inbuilt dataset. “iris” is the variable name in which we will be loading our required dataset. In the next step X variable is loaded with iris.datal[:, :2], in this we only take the first two features as input for training.
And the Y variable is loaded with iris.target, that is the output of the original data.
Then we Split arrays or matrices into a random train and test subset using train_test_split(). We provide the proportion of data to use as a test set and we can provide the parameter random_state, which is used to ensure repeatable results.
Test_size is used to decide how much data to be given for testing to the model.

```
iris = datasets.load_iris()
X = iris.data[:, :2] # we only take the first two features
y = iris.target
x_train, x_test, y_train, y_test = train_test_split(X, y, random_state = 0, t
est_size = 0.25)
```



3. Define classifier.

We will be using the SVC (support vector classifier) SVM (support vector machine). Our kernel is going to be linear, and C is equal to 1. C is a valuation of “how badly" you want to properly classify, or fit, everything. We are going to just stick with 1 for now, which is a nice default parameter.

```
clf = svm.SVC(kernel= 'linear', C=1).fit(x_train, y_train)
```

4. Predicting the output and printing the accuracy of the model.

In this step we use the clf.predict(x_test), that is the classifier to predict the test results and then we print the accuracy score.

```
classifier_predictions = clf.predict(x_test)
print(accuracy_score(y_test, classifier_predictions)*100)
```

```
classifier_predictions = clf.predict(x_test)
print(accuracy_score(y_test, classifier_predictions)*100)

76.31578947368422
```

5. Finally plotting the classifier for our program.

In this step, we will be plotting our classifier. Here the np.meshgrid() function is used to create a rectangular grid out of two given one-dimensional arrays representing the Cartesian indexing or Matrix indexing.

- NumPy provides the reshape() function on the NumPy array object that can be used to reshape the data.

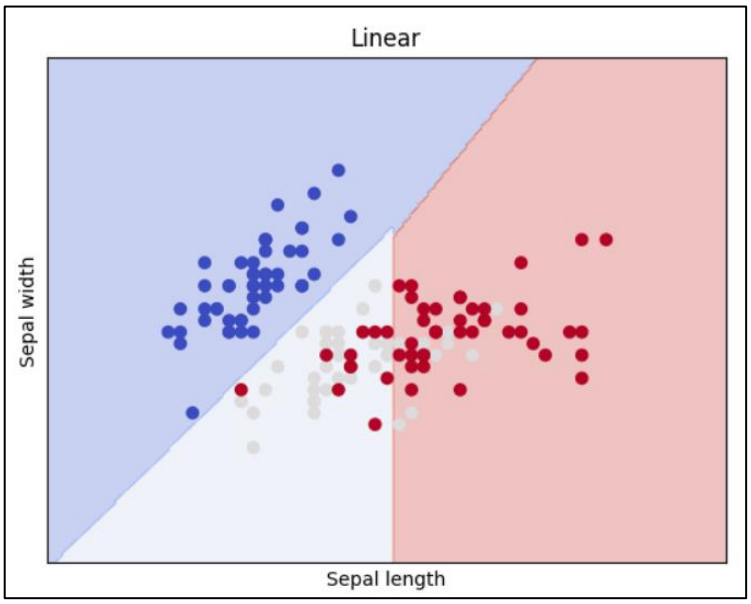
In the case of reshaping a one-dimensional array into a two-dimensional array with one column, the tuple would be the shape of the array as the first dimension and 1 for the second dimension.

- plt.scatter() is used to plot the points on the graph and plt.show() displays the graph.
- The numpy.ravel() functions returns contiguous flattened array(1D array with all the input-array elements and with the same type as it)

The contourf() function in the pyplot module of matplotlib library is used to plot contours. But contourf draw filled contours, while contourf draws contour lines.

```
h = 0.02
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1

xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                     np.arange(y_min, y_max, h))
xx.shape

Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.3)

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.coolwarm)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
```

```
plt.yticks(())
plt.title("Linear")
plt.show()
```



Practical No: 03

Aim: Install, configure, and run Hadoop and HDFS and explore HDFS.

Prerequisites: Windows Subsystem for Linux (WSL) on Windows 10

Theory:

- **Windows Subsystem for Linux (WSL) on Windows 10**

You can run Linux alongside Windows 10 without the need for a second device or virtual machine using the Windows Subsystem for Linux, and here's how to set it up.
On Windows 10, the Windows Subsystem for Linux (WSL) is a feature that creates a lightweight environment that allows you to install and run supported versions of Linux (such as Ubuntu, OpenSuse, Debian, etc.) without the complexity of setting up a virtual machine or different computer.

To install the Windows Subsystem for Linux using PowerShell, use these steps:
You can now install everything you need to run Windows Subsystem for Linux (WSL) by entering this command in an administrator PowerShell or Windows Command Prompt and thenrestarting your machine.

- **wsl --install**

This command will enable the required optional components, download the latest Linux kernel,set WSL 2 as your default, and install a Linux distribution for you (Ubuntu by default, see below to change this).
The first time you launch a newly installed Linux distribution, a console window will open andyou'll be asked to wait for files to decompress and be stored on your machine.

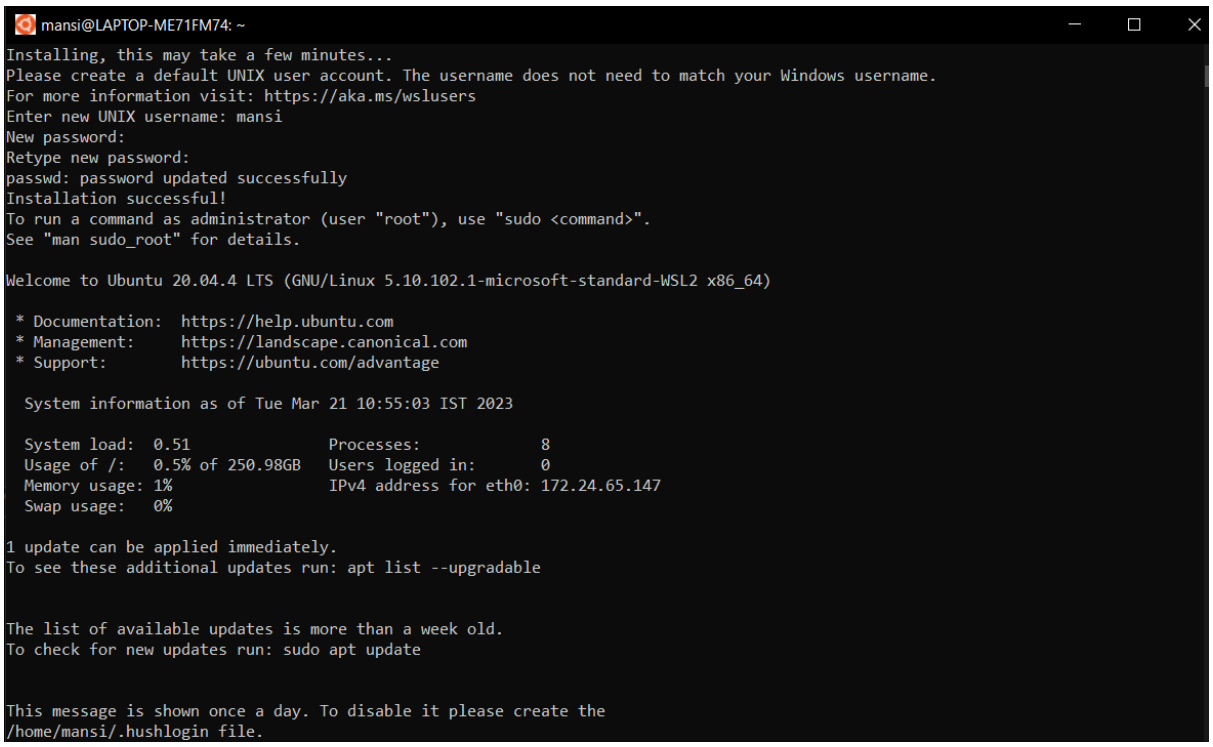
Steps:

Step 1: Open Start and Search for PowerShell, right-click the top result, and select the Run as administrator option.

Step 2: Type the following command to enable the Linux subsystem and press Enter:

```
wsl --install
```

Step 3: Open the app. It will load the Ubuntu environment and will ask you for a usernameand password. Here I've given the mansi name as a username with password mansi123.

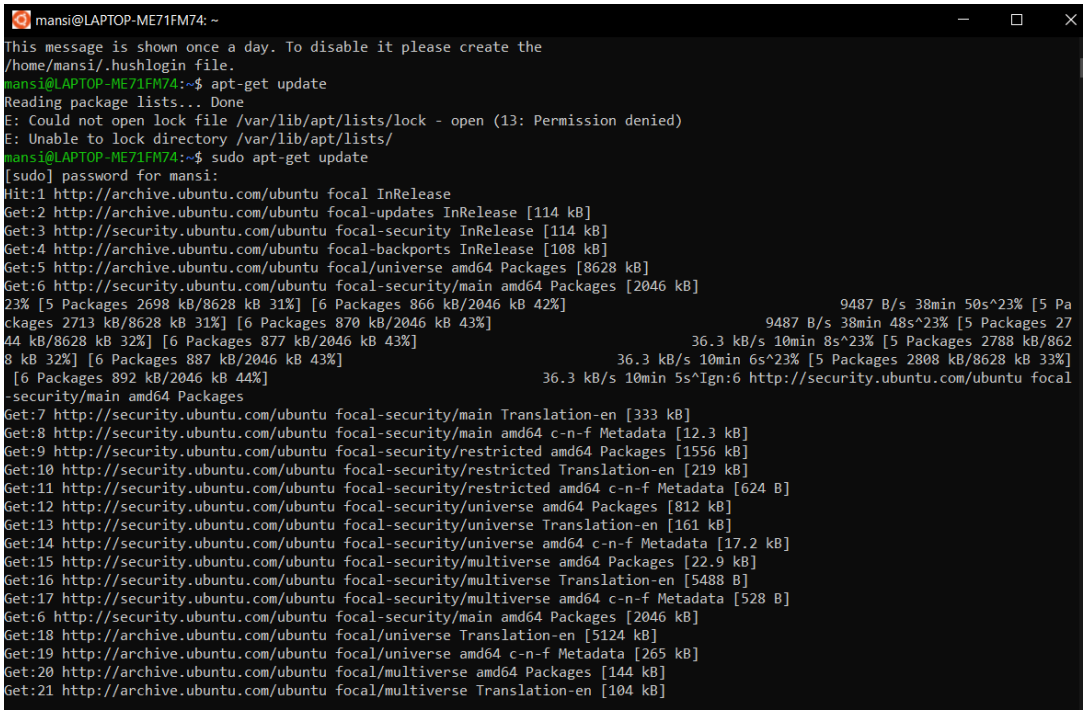


- **Installation of Hadoop in Ubuntu distribution:**

First you need to perform some prerequisites

Command 1: apt-get update

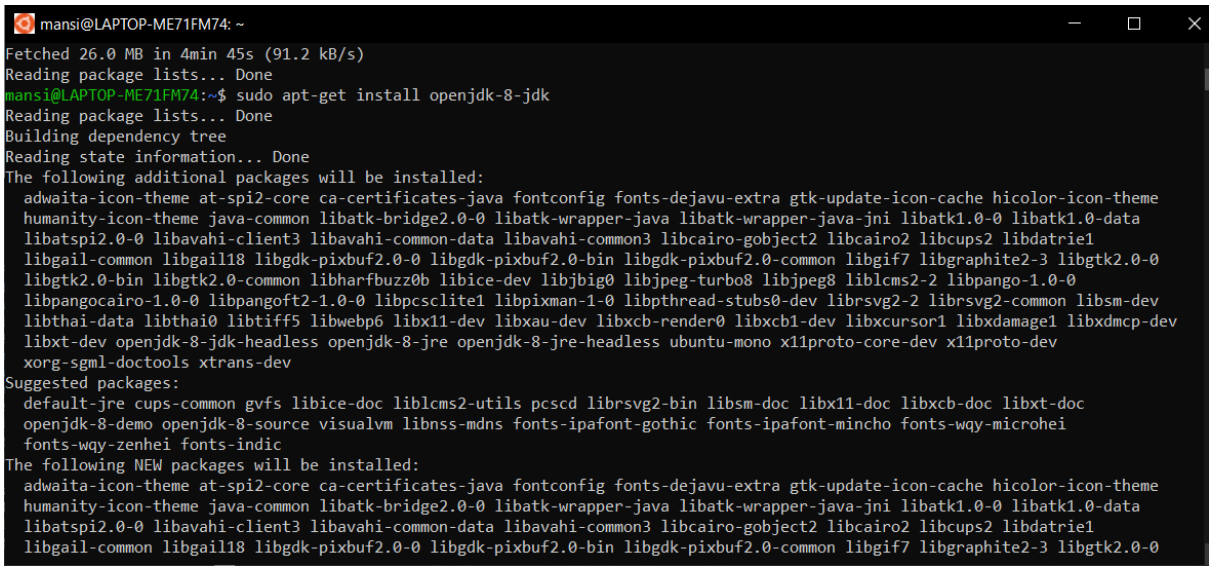
sudo apt-get update fetches the latest version of the package list from your distro's software repository, and any third-party repositories you may have configured. if you get permissiondenied error then try with sudo apt-get update.

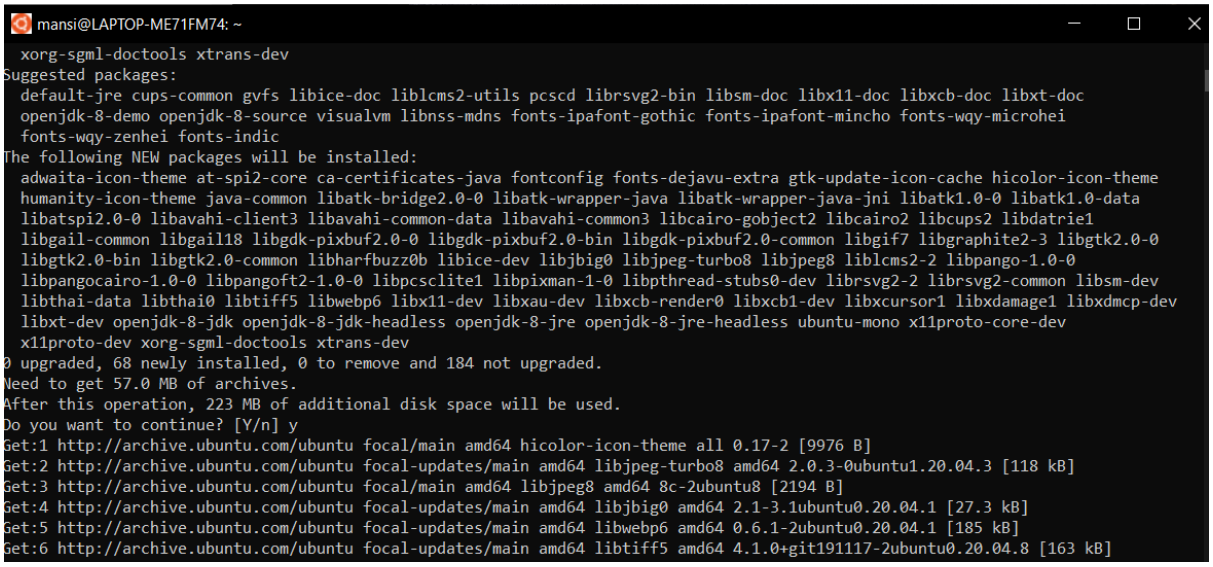


Command 2: sudo apt-get install openjdk-8-jdk

Enter Y if it asks you –

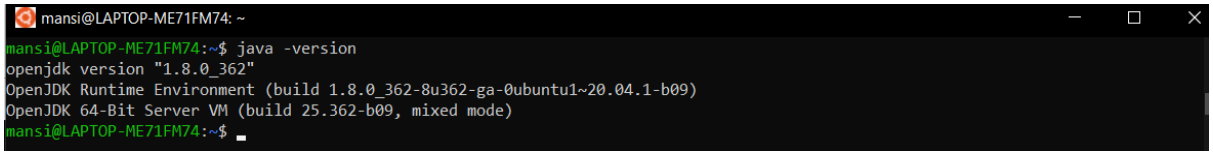
Do you want to continue? [Y/n] Y





Command 3: java -version

check java installation

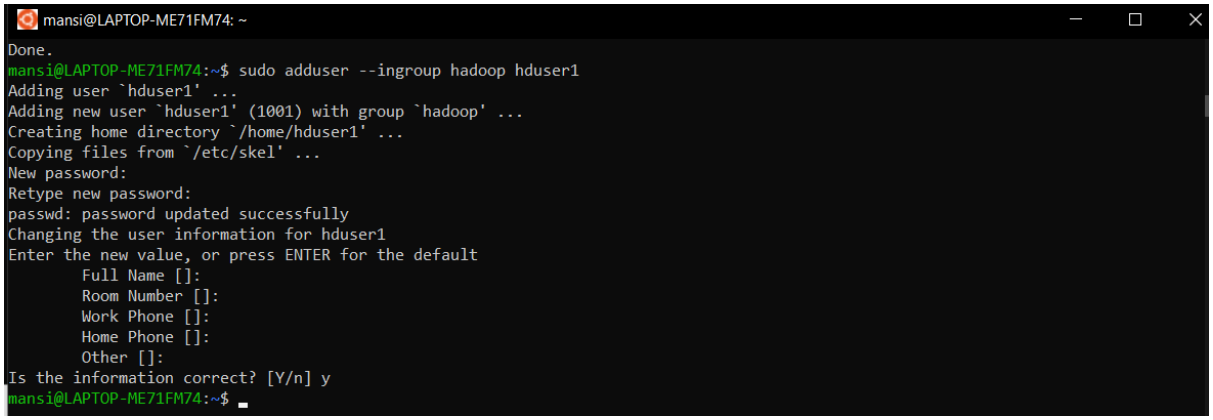


Command 4: sudo add group hadoop

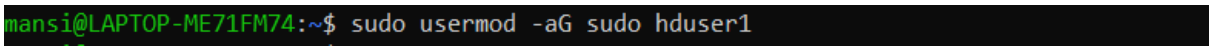
Above command is creating a new group with the name "hadoop".

Command 5: sudo adduser --ingroup hadoop hduser1

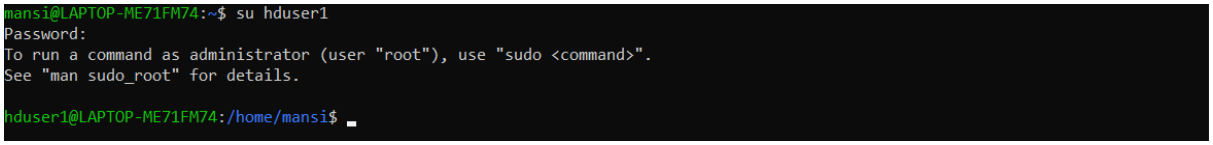
Create a new user hduser1 and add it into hadoop group. Here I've given hduser1 name as a username with password mansi123.



Command 6: usermod -aG sudo hduser1



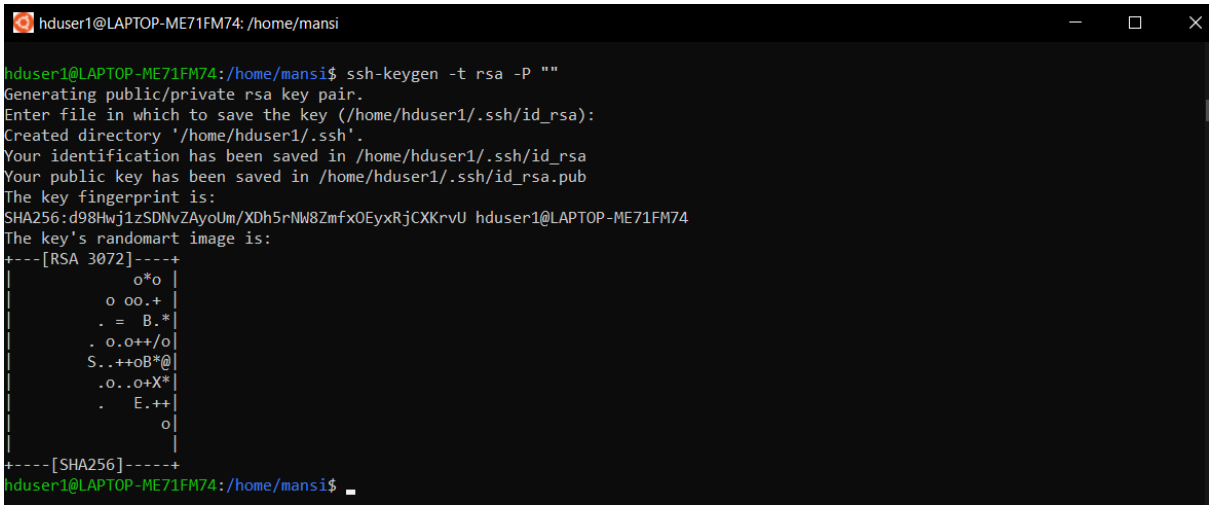
Command 7: su hduser1



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

Generating public/private rsa key pair.

Press Enter key if it asks you – Enter file in which to save the key(/home/hduser/.ssh/id_rsa):



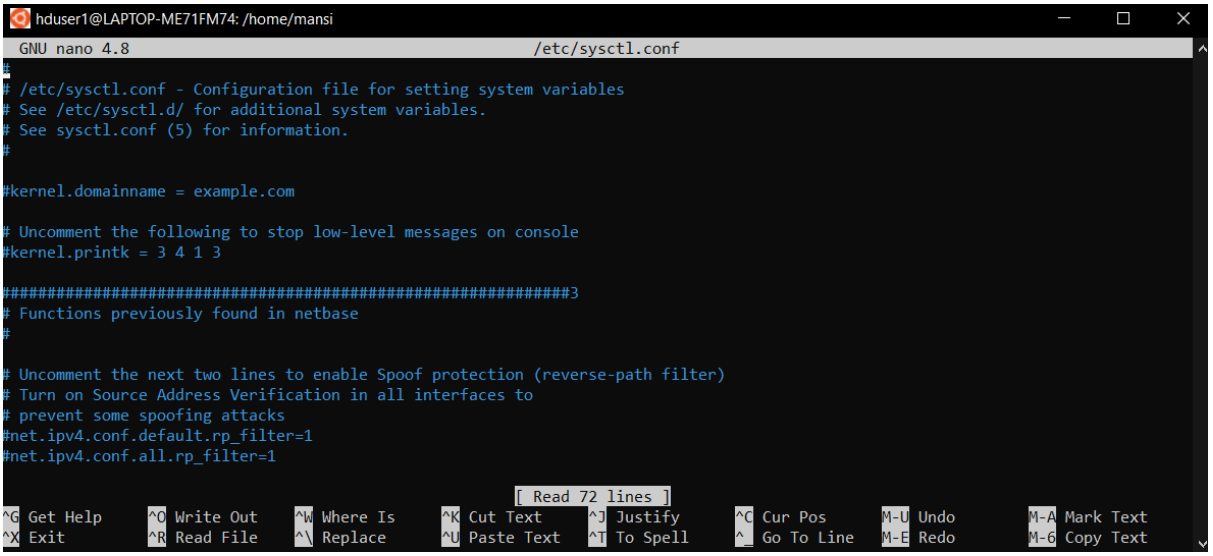
Command 9: cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

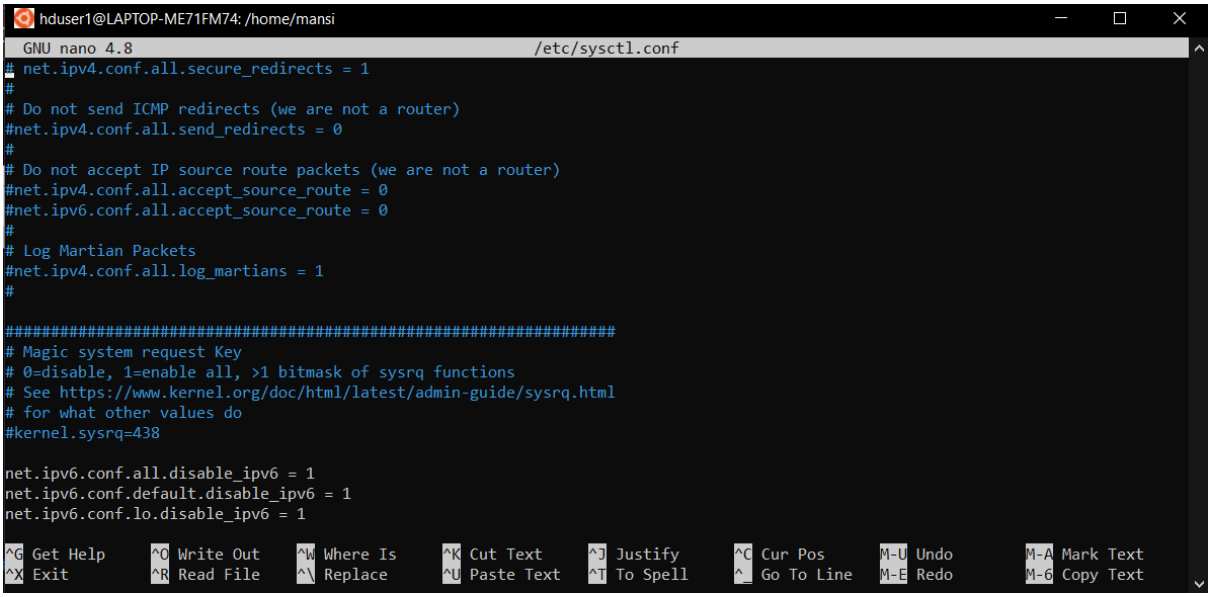
Now you need to disable IPv6. Open the /etc/sysctl.conf file and add the following lines to the end of the file and save it. (One way of opening the file is sudo nano /etc/sysctl.conf, afteryou add the lines, you need to press Ctrl+X, Shift Y and Enter)

Command 10: sudo nano /etc/sysctl.conf

```
hduser1@LAPTOP-ME71FM74:/home/mansi$ sudo nano /etc/sysctl.conf
[sudo] password for hduser1:
hduser1@LAPTOP-ME71FM74:/home/mansi$ sudo nano /etc/sysctl.conf
hduser1@LAPTOP-ME71FM74:/home/mansi$
```

net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1





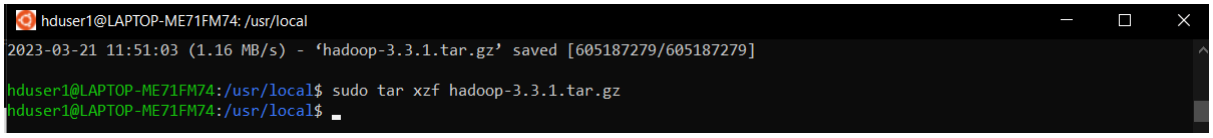
Step 4: Now we will download Hadoop.

Command 11: cd /usr/local

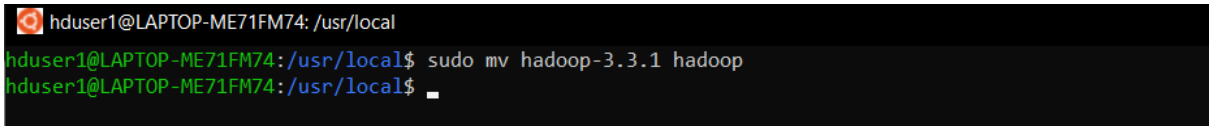
```
hduser1@LAPTOP-ME71FM74: /usr/local
hduser1@LAPTOP-ME71FM74:/home/mansi$ cd /usr/local
hduser1@LAPTOP-ME71FM74:/usr/local$
```

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

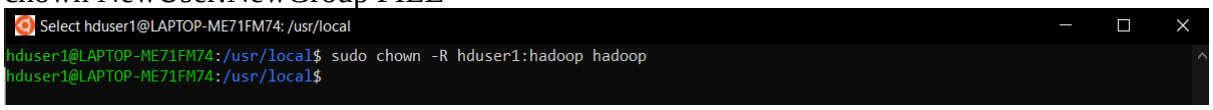
Command 13: sudo tar xzf hadoop-3.3.1.tar.gz



Command 14: sudo mv hadoop-3.3.0 hadoop



Command 15: sudo chown -R hduser1:hadoop hadoop
chown NewUser:NewGroup FILE



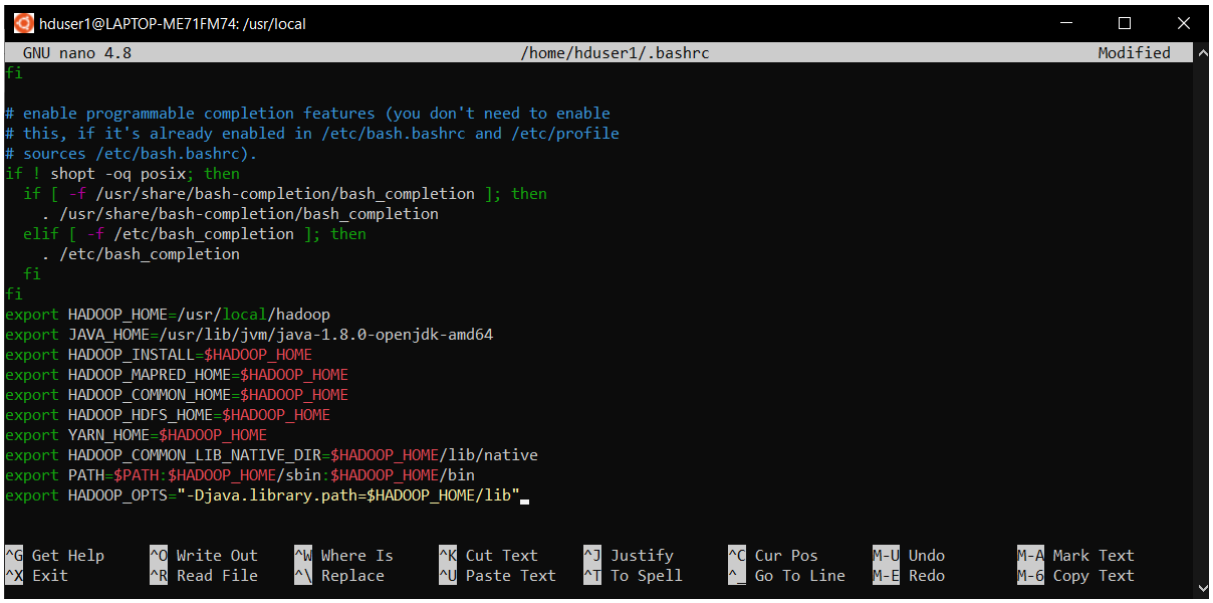
Step 5: Now open $HOME/.bashrc and add the following lines:
Command 16: nano $HOME/.bashrc

Define the Hadoop environment variables by adding the following content to the end of thefile:

#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

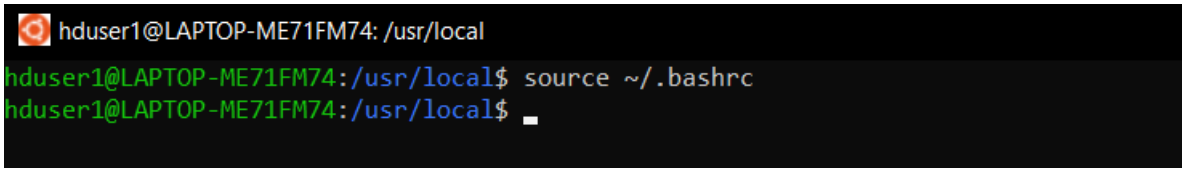
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"



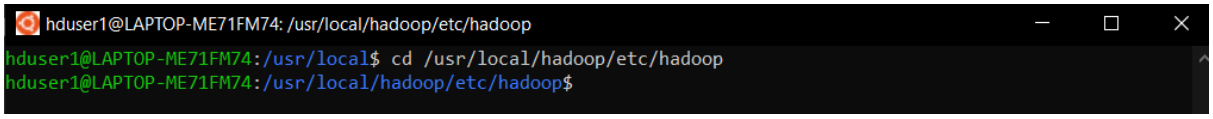
Step 6: Enter the following Commands.

Command 17: source ~/.basher

Above command used for activate environment.

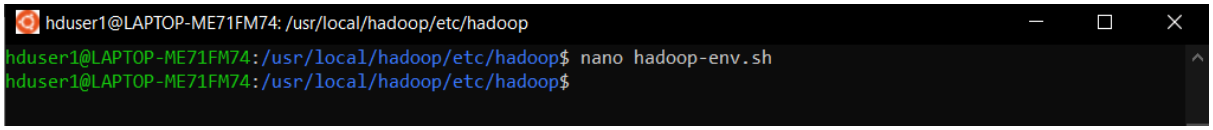


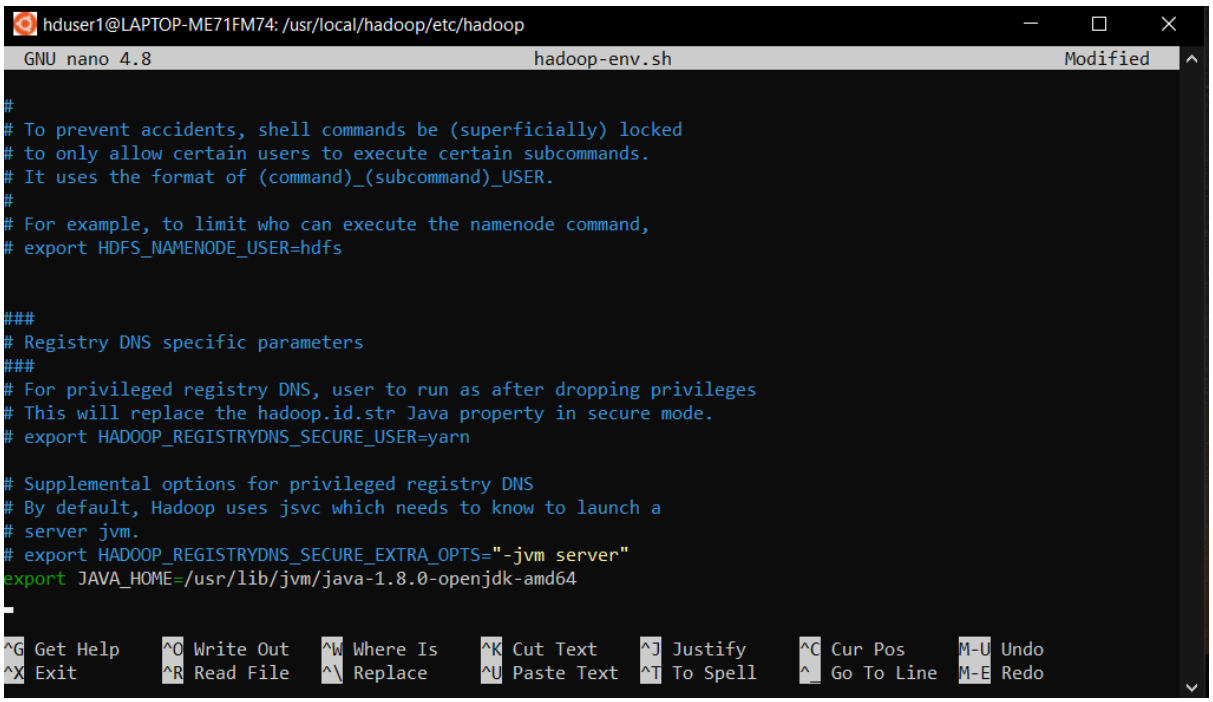
Command 18: cd /usr/local/hadoop/etc/hadoop



Step 7: Add the following line to hadoop-env.sh
exportJAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64

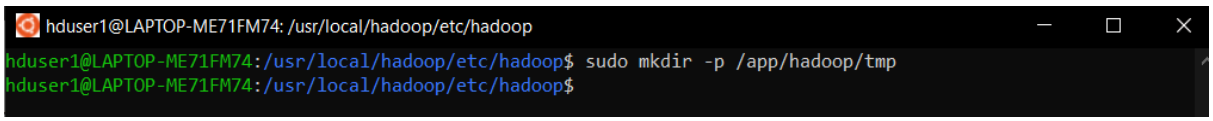
Command 19: nano hadoop-env.sh





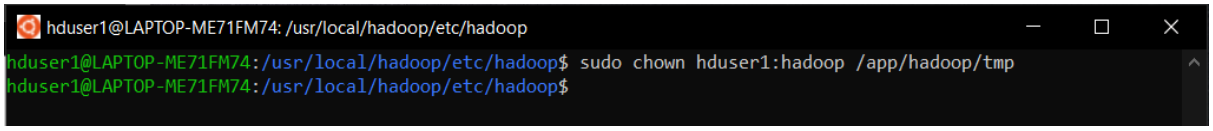
Step 8: Run the following commands:

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



The mkdir command creates one or more directory elements. -p. Creates missing intermediatepath name directories. If the -p flag is not specified, the parent directory of each-newly created directory must already exist.

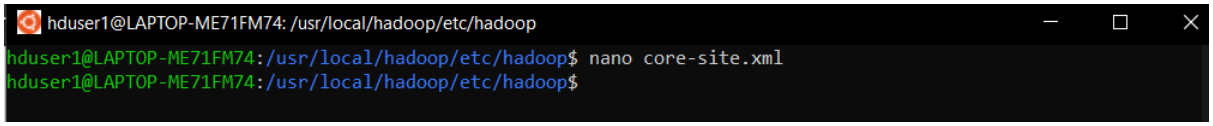
Command 21: sudo chown hduser1:hadoop /app/hadoop/tmp



Step 9: Make the following changes in core-site.xml (this file is present in /usr/local/hadoop/etc/hadoop)

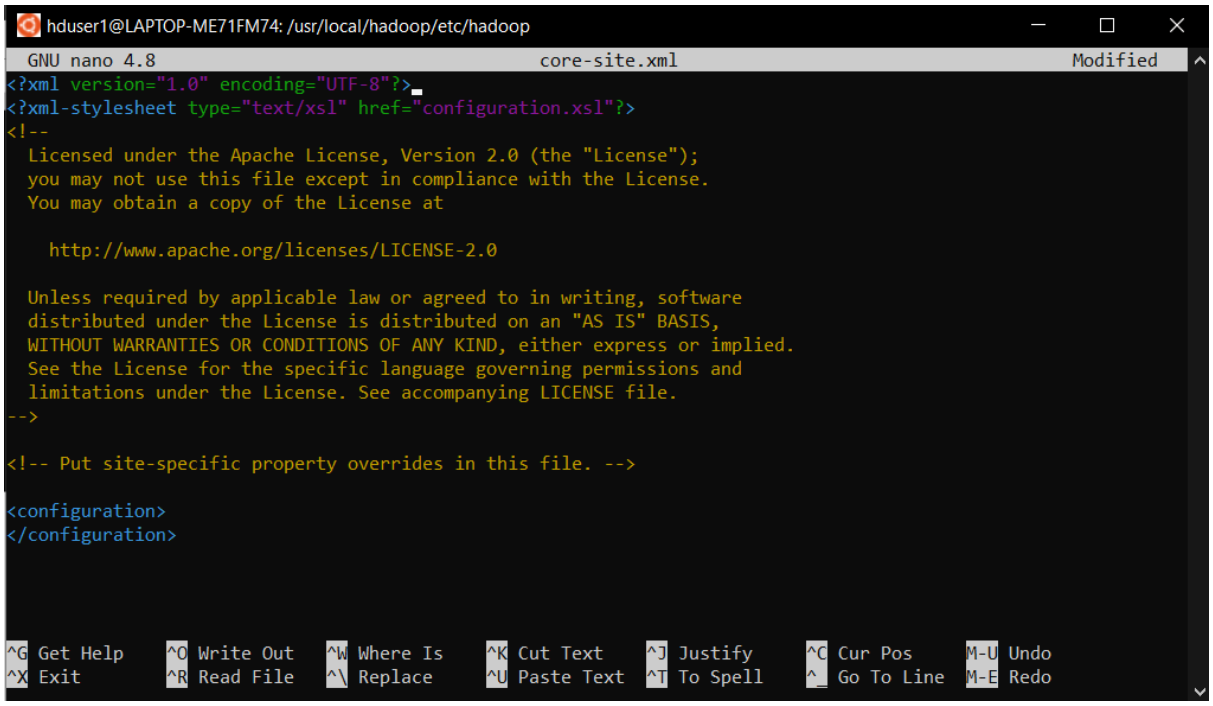
Edit The core-site.xml file which defines HDFS and Hadoop core properties. To set up Hadoopin a pseudo-distributed mode, we need to specify the URL for our NameNode, and the temporary directory Hadoop uses for the map and reduce process

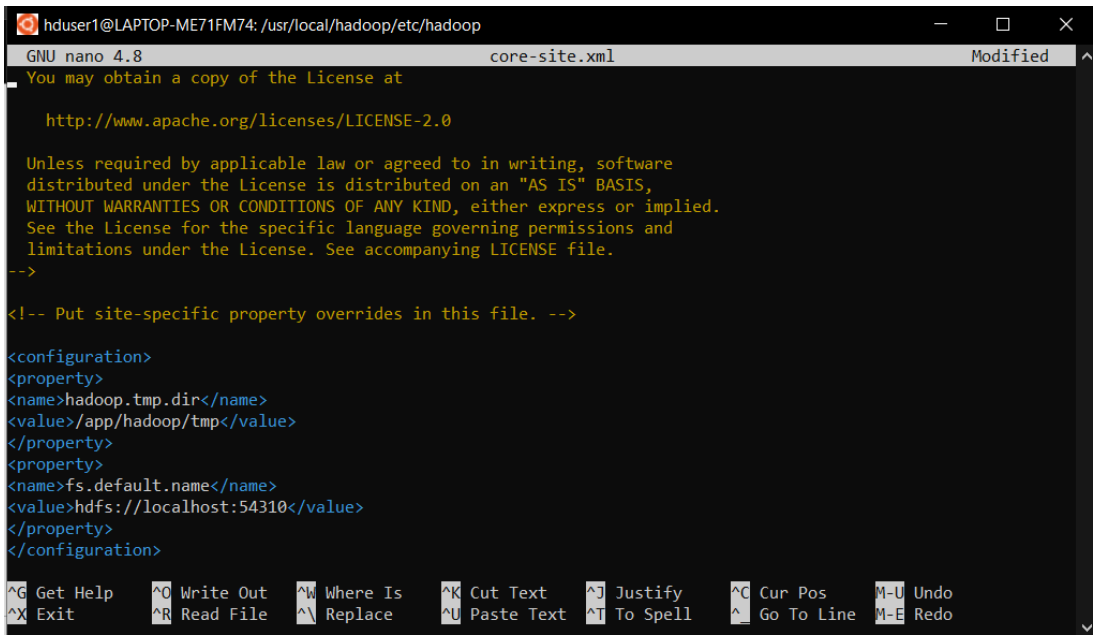
Command 22: nano core-site.xml



Add the following between <configuration> and </configuration>

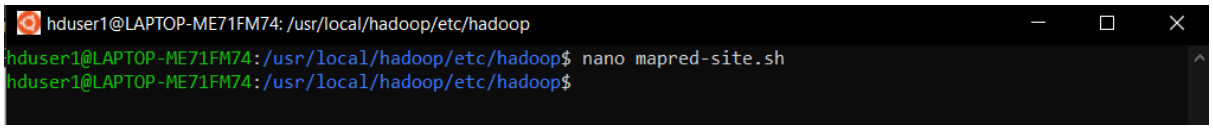
```
<configuration>
<property>
<name>hadoop.tmp.dir</name>
<value>/app/hadoop/tmp</value>
<description>A Base for other temporary
directories.</description>
</property>

<property>
<name>fs.default.name</name>
<value>hdfs://127.0.0.1:54310</value>
</property>
</configuration>
```





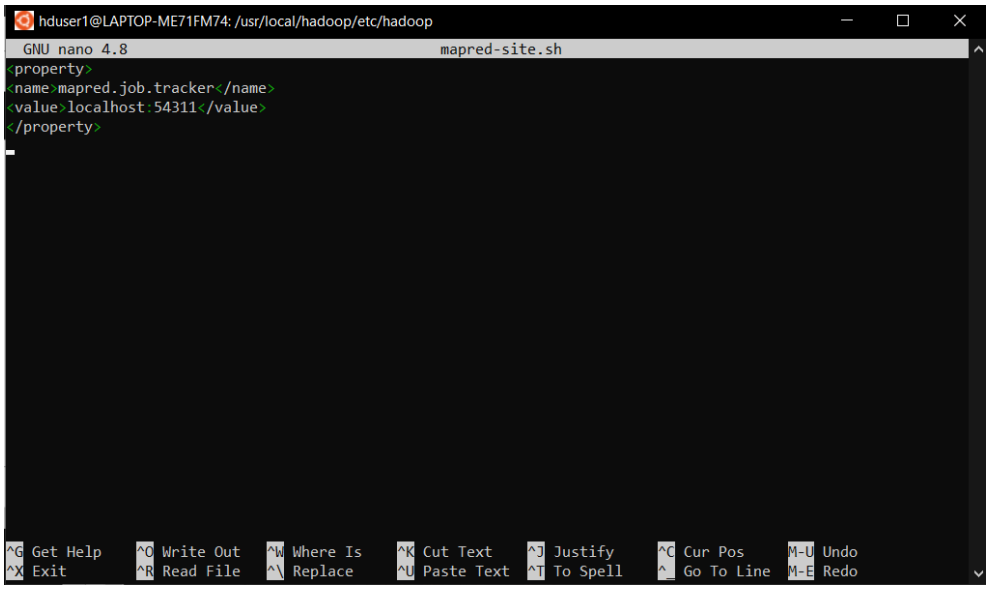
Step 10: In the file mapred-site.xml

Command 23: nano mapred-site.sh



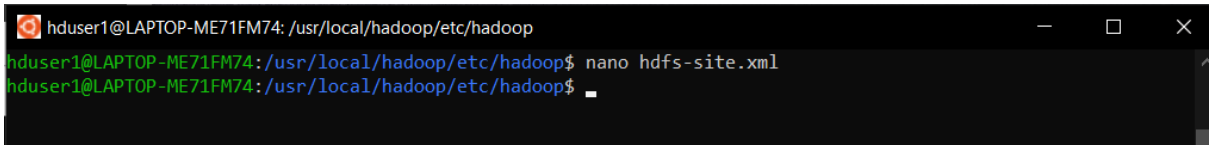
Add the following between <configuration> and </configuration>

```
<property>
<name>mapred.job.tracker</name>
<value>hdfs://127.0.0.1:54311</value>
</property>
```



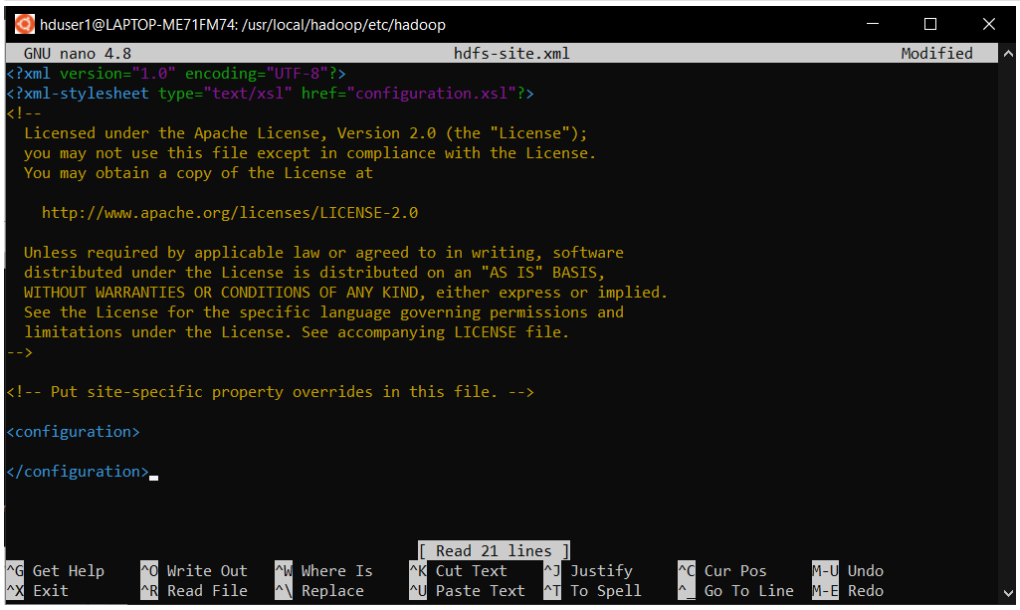
Step 11: In the file hdfs-site.xml

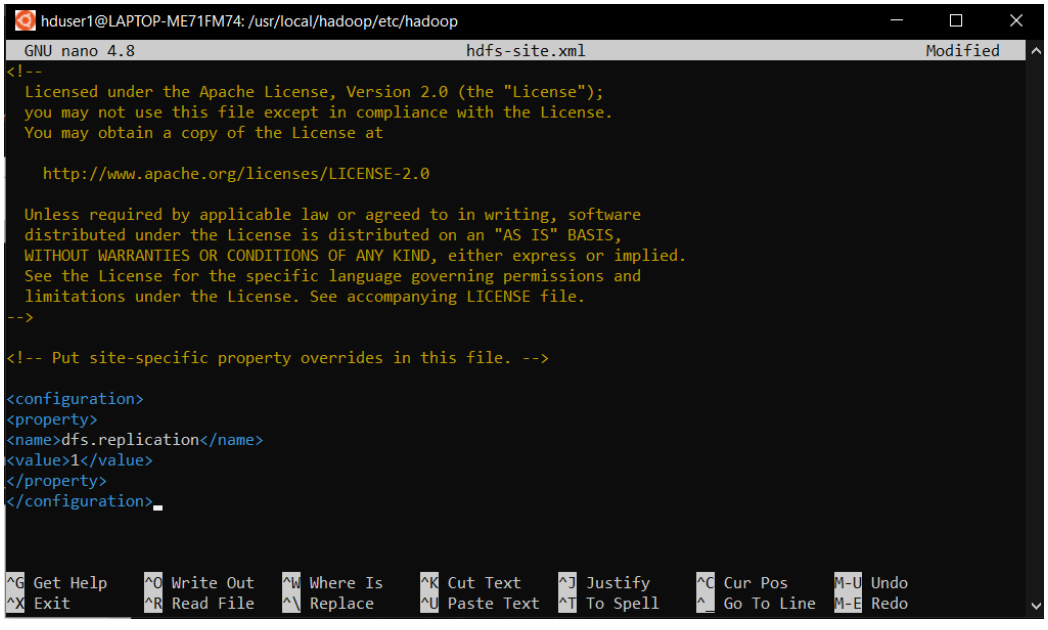
Command 24: nano hdfs-site.xml



Add the following between <configuration> and </configuration>.

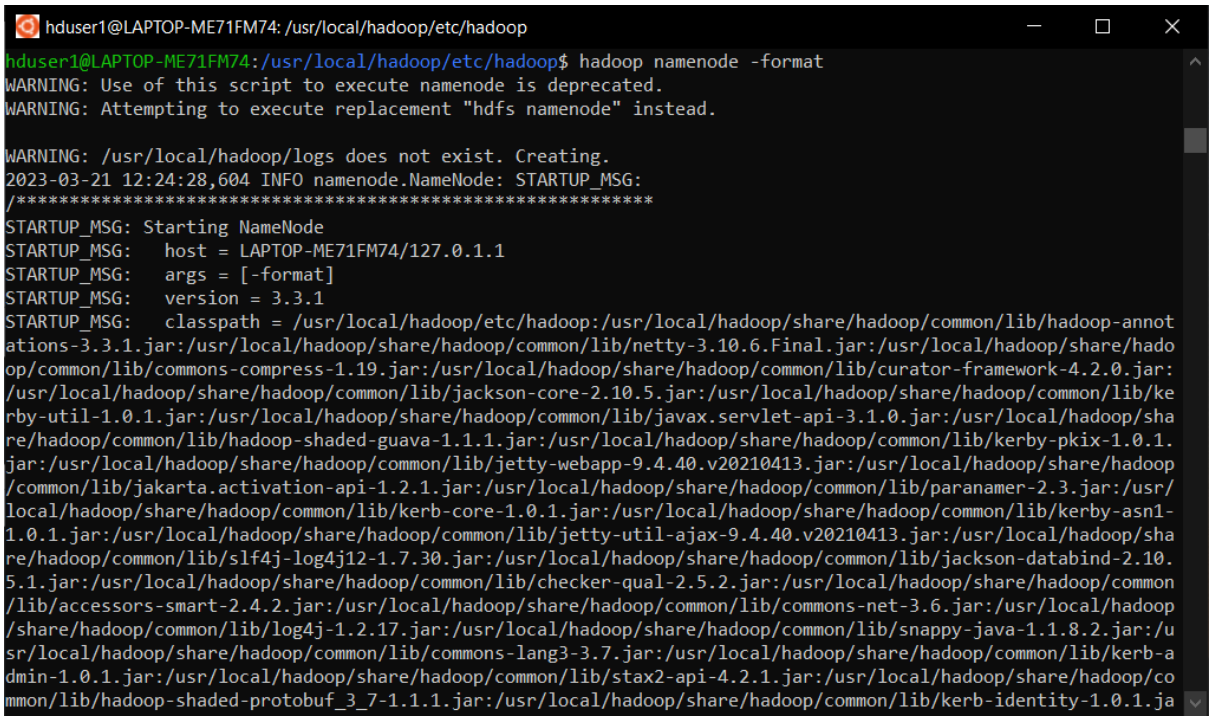
```
<configuration>
        <property>
                <name>dfs.replication</name>
                <value>1</value>
        </property>
</configuration>
```





Step 12: Finally, we format namenode by the following commands:

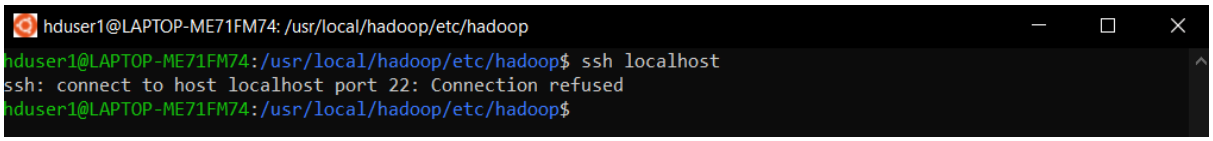
Command 25: hadoop namenode -format



THUS, WE HAVE SUCCESSFULLY INSTALLED HADOOP.

Step 13: Now to start hadoop, we need to run command:

Command 26: ssh localhost

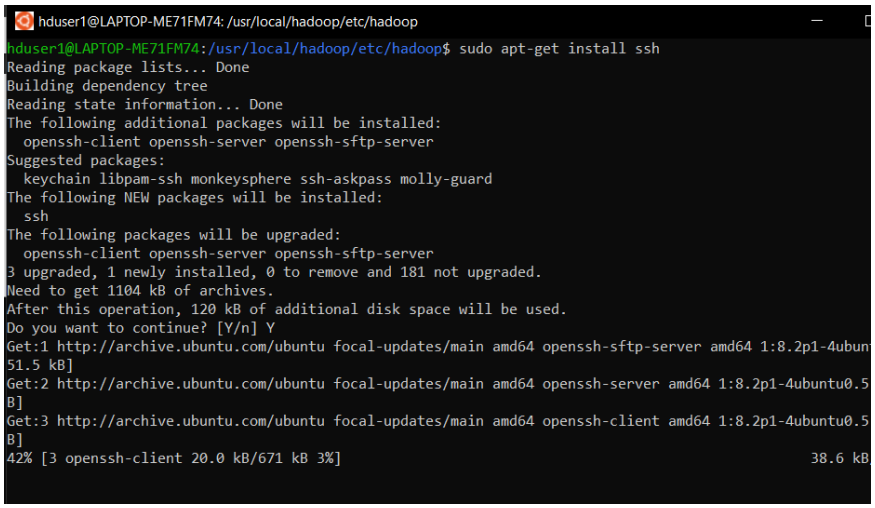


if error occurred with

ssh: connect to localhost port 22: Connection refused

try to install ssh command

Command 27: sudo apt-get install ssh



And then restart the service: by below command

Command 28: sudo service ssh restart

```
hduser1@LAPTOP-ME71FM74: /usr/local/hadoop/etc/hadoop
hduser1@LAPTOP-ME71FM74:/usr/local/hadoop/etc/hadoop$ sudo service ssh restart                    [ OK ]
hduser1@LAPTOP-ME71FM74:/usr/local/hadoop/etc/hadoop$
```

Command 29: ssh localhost

```
hduser1@LAPTOP-ME71FM74: ~
hduser1@LAPTOP-ME71FM74:/usr/local/hadoop/etc/hadoop$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:ZqwY1qituf5E2Y+fvXfiZipS56l5h3OOutShxZFE04k.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.10.102.1-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Mar 21 12:31:28 IST 2023

  System load:  0.23              Processes:             13
  Usage of /:   1.4% of 250.98GB  Users logged in:       0
  Memory usage: 4%                IPv4 address for eth0: 172.24.65.147
  Swap usage:   0%

189 updates can be applied immediately.
144 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

Step 14: This command is to start hadoop services:

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

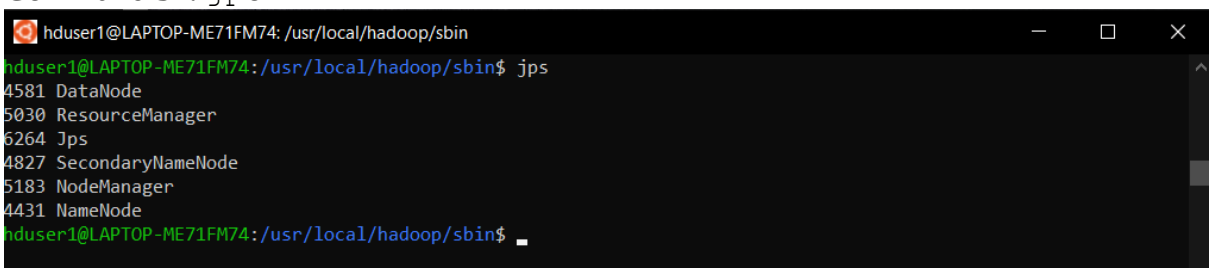
```
hduser1@LAPTOP-ME71FM74: /usr/local/hadoop/sbin
hduser1@LAPTOP-ME71FM74:~$ cd /usr/local/hadoop/sbin
hduser1@LAPTOP-ME71FM74:/usr/local/hadoop/sbin$
```

Command 31: `start-all.sh`

```
hduser1@LAPTOP-ME71FM74: /usr/local/hadoop/sbin
Starting nodemanagers
hduser1@LAPTOP-ME71FM74:/usr/local/hadoop/sbin$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hduser1 in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
localhost: namenode is running as process 4431.  Stop it first and ensure /tmp/hadoop-hduser1-namenode.pid fil
e is empty before retry.
Starting datanodes
localhost: datanode is running as process 4581.  Stop it first and ensure /tmp/hadoop-hduser1-datanode.pid fil
e is empty before retry.
Starting secondary namenodes [LAPTOP-ME71FM74]
LAPTOP-ME71FM74: secondarynamenode is running as process 4827.  Stop it first and ensure /tmp/hadoop-hduser1-s
econdarynamenode.pid file is empty before retry.
2023-03-21 12:35:13,658 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform...
using builtin-java classes where applicable
Starting resourcemanager
resourcemanager is running as process 5030.  Stop it first and ensure /tmp/hadoop-hduser1-resourcemanager.pid
file is empty before retry.
Starting nodemanagers
localhost: nodemanager is running as process 5183.  Stop it first and ensure /tmp/hadoop-hduser1-nodemanager.p
id file is empty before retry.
```

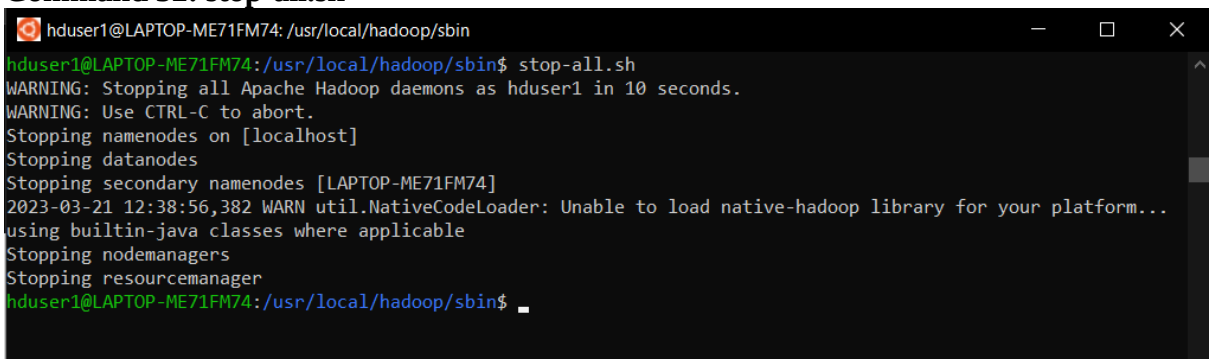
Step 15: This command is to check that all hadoop services are running (6 services should appear)

Command 32: `jps`



Step 16: This command is to stop hadoop services:

Command 32: stop-all.sh



Practical No: 04

Aim: Implement word count / frequency programs using MapReduce.

Theory:

In MapReduce word count example, we find out the frequency of each word. Here, the role ofMapper is to map the keys to the existing values and the role of Reducer is to aggregate the keys of common values. So, everything is represented in the form of a Key-value pair.

The WordCount example reads text files and counts how often words occur. The input is text files and the output is text files, each line of which contains a word and the count of how oftenit occurred, separated by a tab.

Each mapper takes a line as input and breaks it into words. It then emits a key/value pair of theword and each reducer sums the counts for each word and emits a single key/value with the word and sum.

As an optimization, the reducer is also used as a combiner on the map outputs. This reduces the amount of data sent across the network by combining each word into a single record.

Pre-requisites:

- **Java Installation -** Check whether the Java is installed or not using the following command.

java —-version

- **Hadoop Installation -** Check whether the Hadoop is installed or not using the following command.

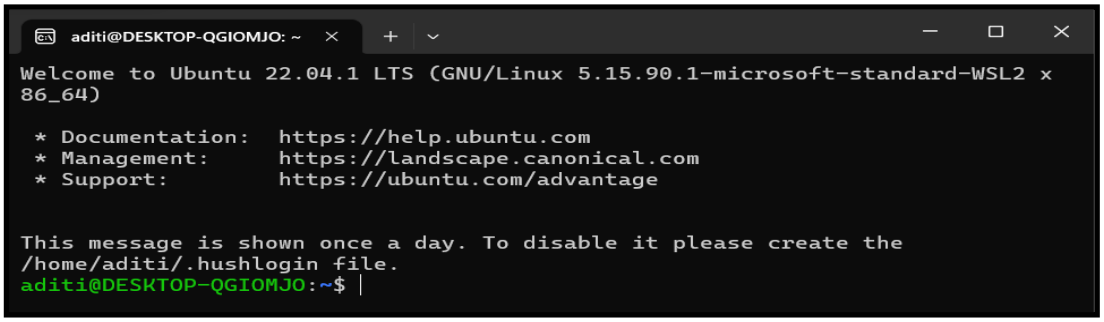
hadoop version

txt

```
lo wordcount MapReduce Hadoop program.This
my first MapReduce program.
```

Steps:

1) Start Ubuntu Subsystem.



Command 1: su hduser1

```
aditi@DESKTOP-QGIOMJO:/home/hduser1$ su hduser1
Password:
hduser1@DESKTOP-QGIOMJO:~$
```

2) Restart the SSH server/daemon.

Command 2: sudo service ssh restart

```
hduser1@DESKTOP-QGIOMJO:~$ sudo service ssh restart
[sudo] password for hduser1:
 * Restarting OpenBSD Secure Shell server sshd
                                        [ OK ]
```

3) Create a connection to your own machine, to the current user i.e., hduser1.

Command 3: ssh localhost

```
hduser1@DESKTOP-QGIOMJO:~$ ssh localhost
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.90.1-microsoft-standard-WSL2 x86_6
4)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Last login: Mon Mar 27 10:23:55 2023 from 127.0.0.1
```

4) Start Hadoop.

Command 4: /usr/local/hadoop/sbin/start-all.sh

```
hduser1@DESKTOP-QGIOMJO:~$ /usr/local/hadoop/sbin/start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hduser1 in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [DESKTOP-QGIOMJO]
2023-03-02 14:35:46,301 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting resourcemanager
Starting nodemanagers
```

5) To check all the services

Command 5: jps

```
hduser1@DESKTOP-QGIOMJO:~$ jps
609 SecondaryNameNode
836 ResourceManager
948 NodeManager
375 DataNode
264 NameNode
1134 Jps
```

Note: If you are performing this practical for the first time you don't need to do this step. (Every time when you run this, you need to remove existing files before starting the execution)

To remove existing file from HDFS Command 1:

hdfs dfs -rm /bda.txt

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -rm /bda.txt
2023-03-28 11:12:10,228 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
```

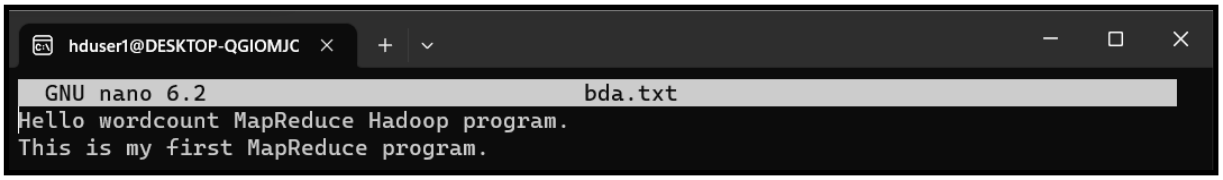
Clear Output of previous run at default HDFS location

Command 2: hdfs dfs -rm -r /output

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -rm -r /output
2023-03-28 11:13:50,960 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Deleted /output
```

6) Create a text file, write some text into it (try to include same and repeated words)

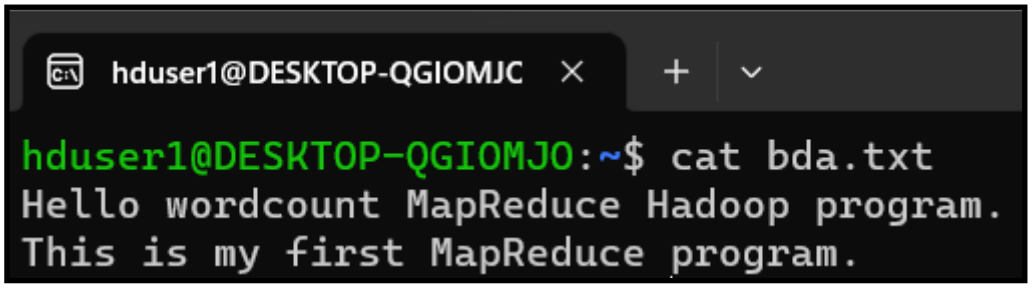
Command 6: sudo nano bda.txt



(Press Ctrl+X then Ctrl+S and enter)

7) Check the text written in the bda.txt file.

Command 7: cat bda.txt



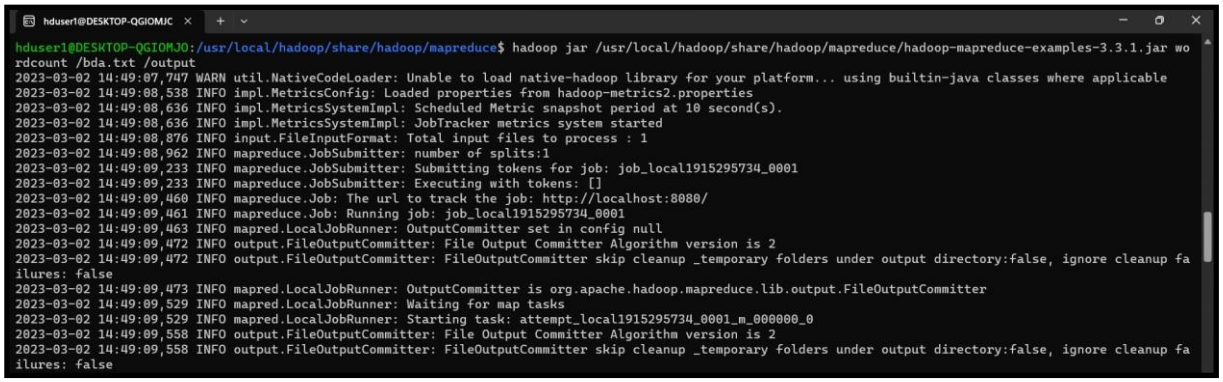
8) Move the bda.txt file to HDFS.

Command 8: hdfs dfs -put /home/hduser1/bda.txt /

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -put /home/hduser1/bda.txt /
2023-03-02 14:43:41,568 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using buil
tin-java classes where applicable
```

9) Running MapReduce for wordcount file bda.txt.

Command 9: hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/Hadoop- mapreduce-examples-3.3.1.jar wordcount /bda.txt /output



This will create a directory output and two files within it viz. _SUCCESS and part-r-00000

10) Check the contents within the HDFS directories. For example, if one has to determine the directories within the output directory, simply type the following command:

Command 10: hdfs dfs -ls /output

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -ls /output
2023-03-02 14:51:06,197 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hduser1 supergroup          0 2023-03-02 14:49 /output/_SUCCESS
-rw-r--r--   1 hduser1 supergroup         77 2023-03-02 14:49 /output/part-r-00000
```

11) To display the content of the files.

Command 11: hdfs dfs -cat /bda.txt /output/part-r-00000

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -cat /bda.txt /output/part-r-00000
2023-03-02 14:51:25,487 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Hello wordcount MapReduce Hadoop program.
This is my first MapReduce program.
Hadoop  1
Hello   1
MapReduce       2
This    1
first   1
is      1
my      1
program.        2
wordcount       1
```

12) Check/display output at default output location.

Command 12: hdfs dfs -head /output/part-r-00000

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -head /output/part-r-00000
2023-03-02 14:51:33,829 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Hadoop  1
Hello   1
MapReduce       2
This    1
first   1
is      1
my      1
program.        2
wordcount       1
```

13) To get output in .txt file in HDFS(optional)

Note: Everytime when you perform this practical, kindly give a different output file name.

Command 13: hdfs dfs -mv /output/part-r-00000 /output/op.txt

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -mv /output/part-r-00000 /output/op.txt
2023-03-02 14:51:45,995 WARN util.NativeCodeLoader: Unable to load native-ha
doop library for your platform... using builtin-java classes where applicabl
e
```

14) To check whether the output got saved in a text file, list the files present in theoutput directory.

Command 14: hdfs dfs -ls /output

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -ls /output
2023-03-02 14:51:53,711 WARN util.NativeCodeLoader: Unable to load native-ha
doop library for your platform... using builtin-java classes where applicabl
e
Found 2 items
-rw-r--r--   1 hduser1 supergroup          0 2023-03-02 14:49 /output/_SUCCE
SS
-rw-r--r--   1 hduser1 supergroup         77 2023-03-02 14:49 /output/op.txt
```

15) To get output in a .txt file in default file location(optional)

Command 15: hdfs dfs -get /output/op.txt /home/hduser1

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -get /output/op.txt /home/hduser1
2023-03-02 14:52:55,550 WARN util.NativeCodeLoader: Unable to load native-ha
doop library for your platform... using builtin-java classes where applicabl
e
```

16) You can check whether the file got saved in the home directory of the user bylisting its content.

Command 16: ls

```
hduser1@DESKTOP-QGIOMJO:~$ ls
bda.txt  op.txt
```

17) By using cat command view the content of wordcountop.txt file.

Command 17: cat op.txt

```
hduser1@DESKTOP-QGIOMJO:~$ cat op.txt
Hadoop  1
Hello   1
MapReduce       2
This    1
first   1
is      1
my      1
program.        2
wordcount       1
```

18) To view content of HDFS location/structure, use the following command.

Command 18: hdfs dfs -ls /

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -ls /
2023-03-02 14:53:26,976 WARN util.NativeCodeLoader: Unable to load native-ha
doop library for your platform... using builtin-java classes where applicabl
e
Found 2 items
-rw-r--r--   1 hduser1 supergroup         78 2023-03-02 14:43 /bda.txt
drwxr-xr-x   - hduser1 supergroup          0 2023-03-02 14:51 /output
```

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

Command 19: hdfs dfs -cat /output/op.txt

```
hduser1@DESKTOP-QGIOMJO:~$ hdfs dfs -cat /output/op.txt
2023-03-02 14:53:34,867 WARN util.NativeCodeLoader: Unable to load native-ha
doop library for your platform... using builtin-java classes where applicabl
e
Hadoop  1
Hello   1
MapReduce       2
This    1
first   1
is      1
my      1
program.        2
wordcount       1
hduser1@DESKTOP-QGIOMJO:~$
```

Practical No: 05

Aim: Implement an application that stores big data in MongoDB and manipulate it using R / Python.

Requirement:

a. PyMongo
b. Mongo Database
c. Mongo shell

Theory:

- **Mongo Database**

MongoDB is an open-source document-oriented database that is designed to store a large scaleof data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables. The MongoDB database is developed and managed by MongoDB.Inc under SSPL(Server Side Public License) and initially released in February 2009.It also provides official driver support for all the popular languages like C, C++, C#, and .Net,Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid. So, you can createan application using any of these languages. Nowadays there are so many companies that use MongoDB like Facebook, Nokia, eBay, Adobe, Google, etc. to store their large amounts of data.

- **Mongo Shell**

The mongo shell is an interactive JavaScript interface to MongoDB. You can use the mongoshell to query and update data as well as perform administrative operations.

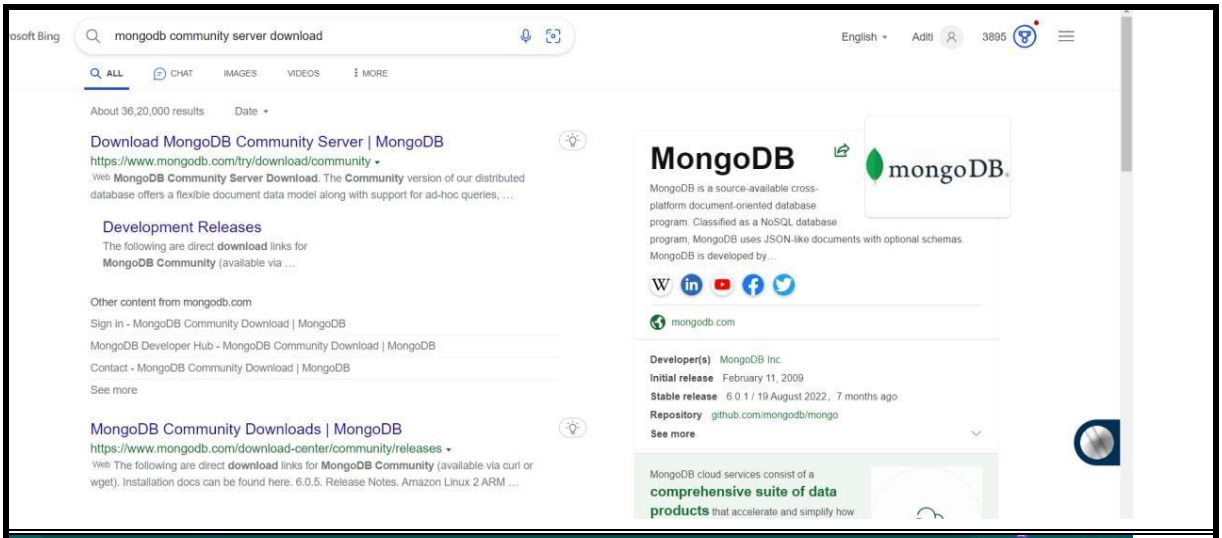
- **PyMongo**

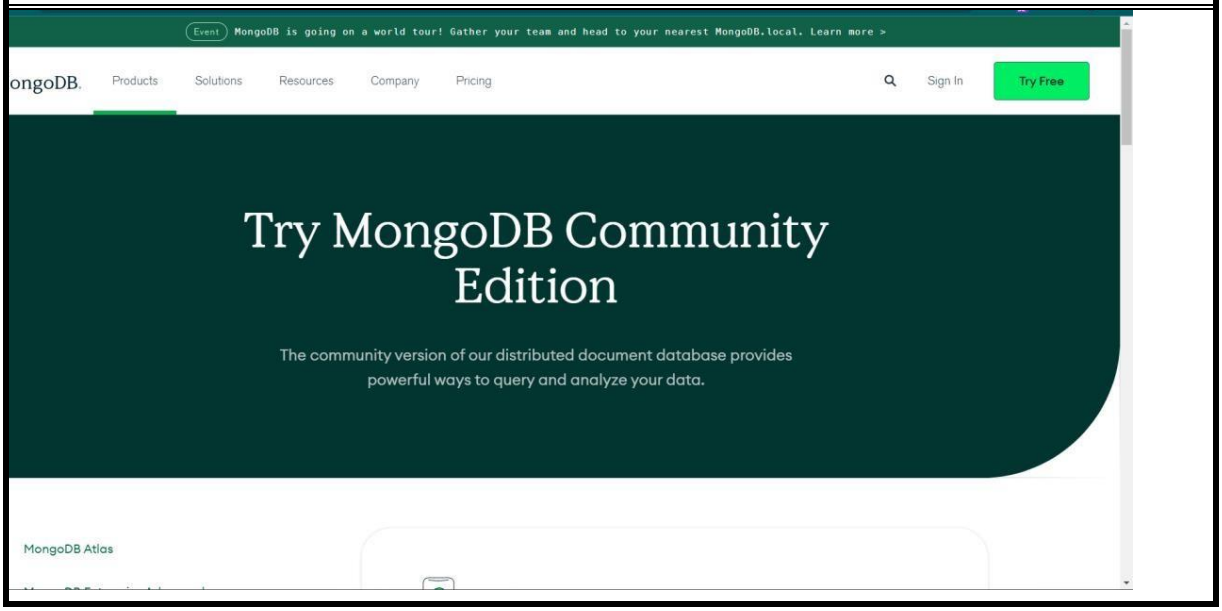
PyMongo is the official Python driver that connects to and interacts with MongoDB databases.The PyMongo library is being actively developed by the MongoDB team.

Steps:

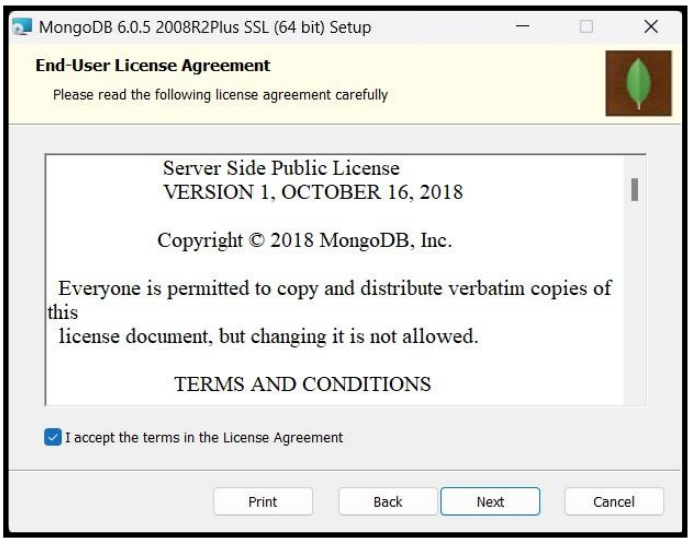
Step A: Install Mongo database

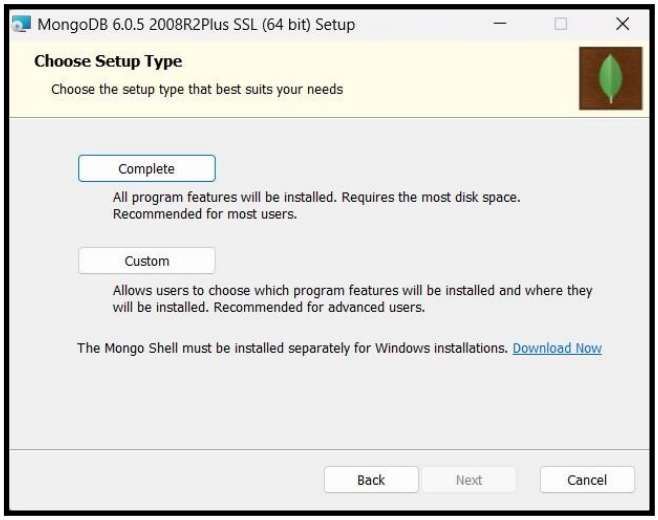
1) Go to (https://www.mongodb.com/download-center/community) and Download MongoDB Community Server. We will install the 64-bit version for Windows.

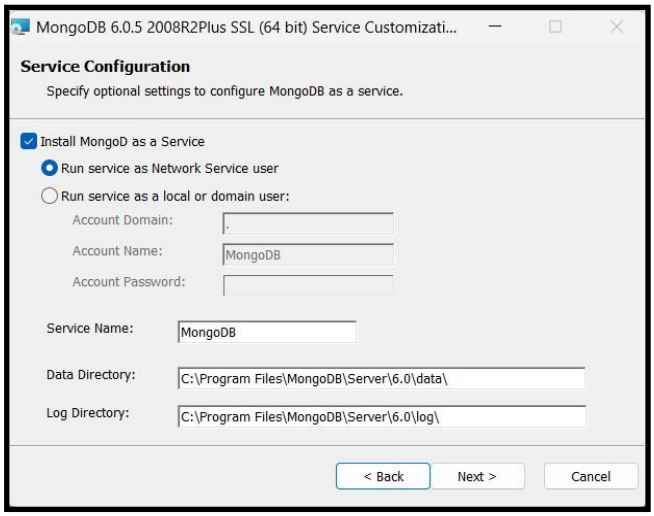


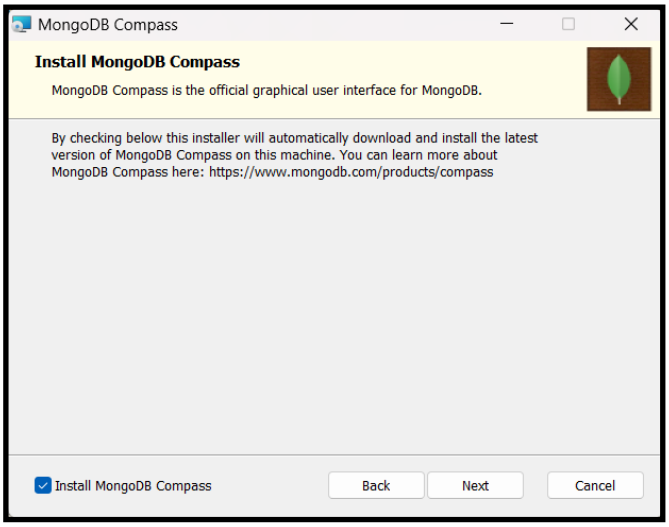


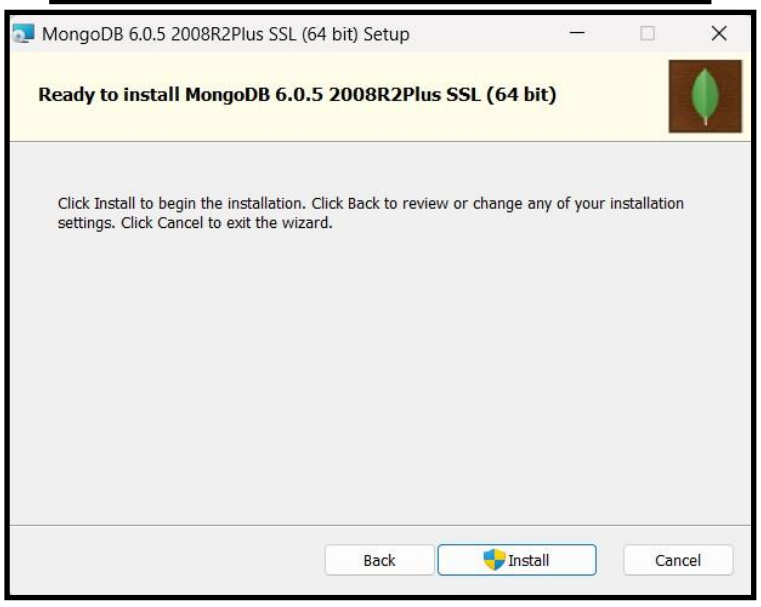
2) Once download is complete, open the msi file. Click Next in the startup screen.

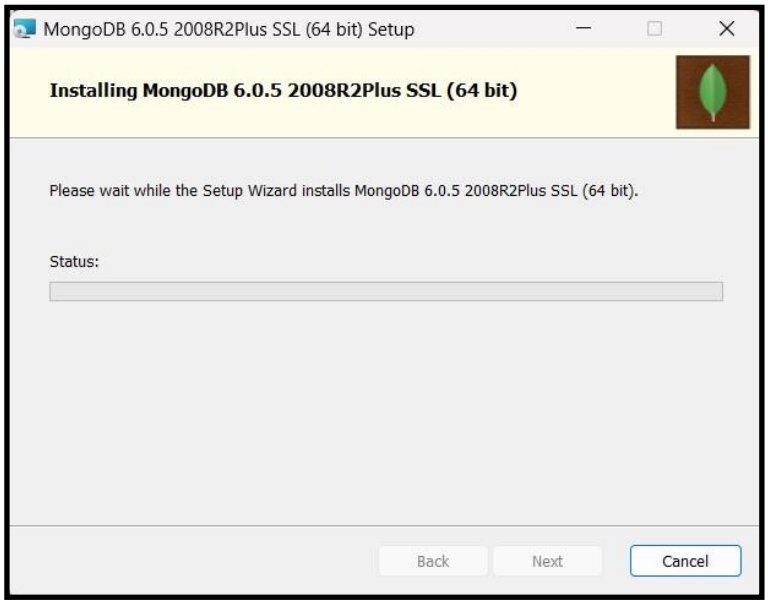




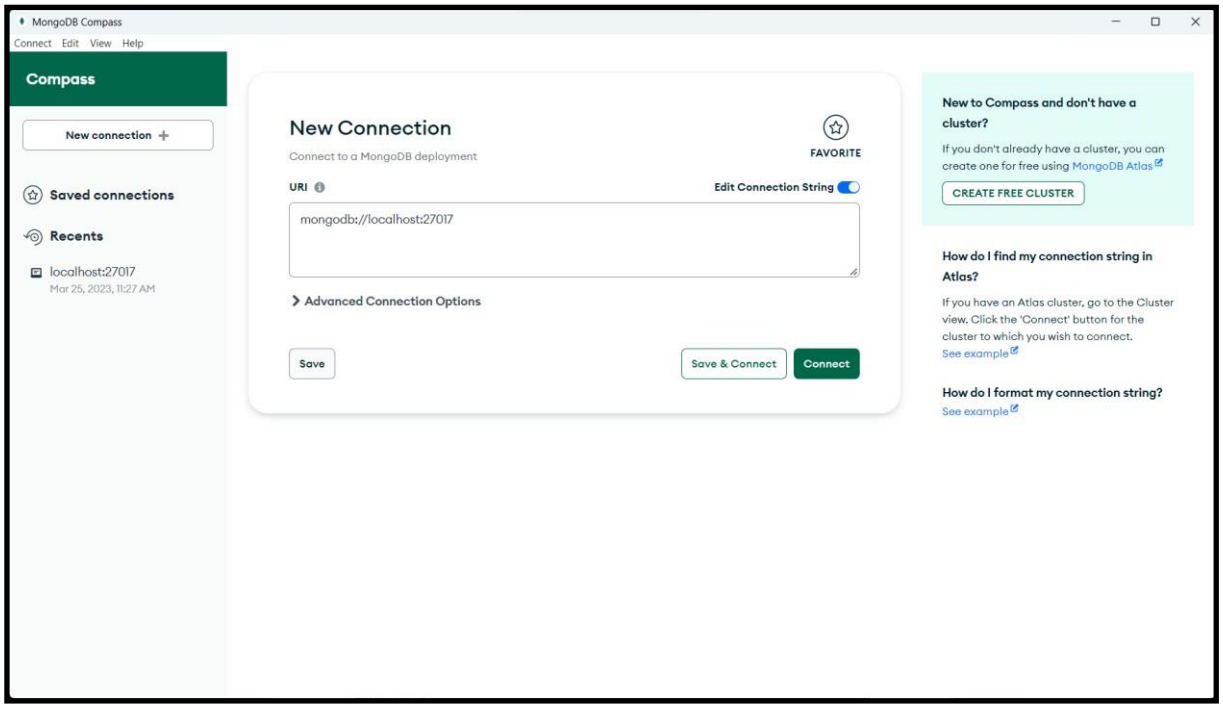

MongoDB 6.0.5 2008R2Plus SSL (64 bit) Service Customizati...
Service Configuration
Specify optional settings to configure MongoDB as a service.
Install MongoD as a Service
Run service as Network Service user
Run service as a local or domain user:
Account Domain:
Account Name:
MongoDB
Account Password:
Service Name:
MongoDB
Data Directory:
C:\Program Files\MongoDB\Server\6.0\data\
Log Directory:
C:\Program Files\MongoDB\Server\6.0\log\
< Back
Next >
Cancel


MongoDB Compass
Install MongoDB Compass
MongoDB Compass is the official graphical user interface for MongoDB.
By checking below this installer will automatically download and install the latest version of MongoDB Compass on this machine. You can learn more about MongoDB Compass here: https://www.mongodb.com/products/compass
Install MongoDB Compass
Back
Next
Cancel

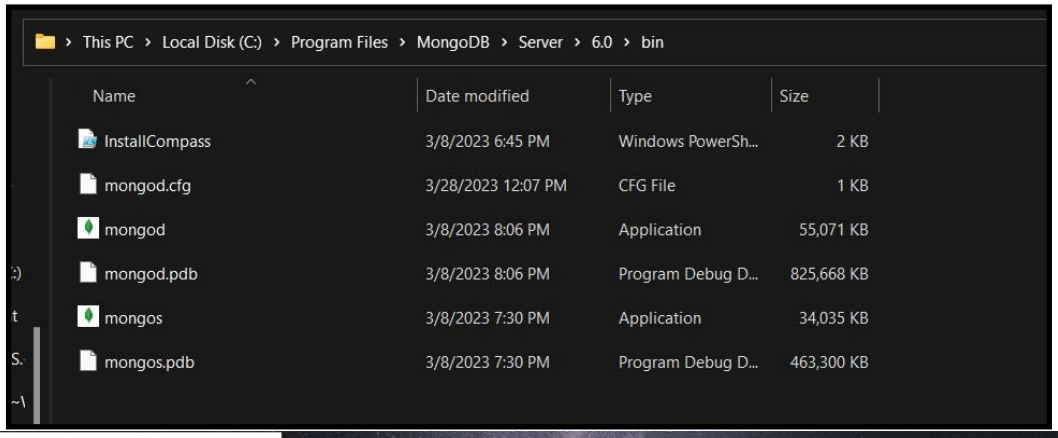

MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup
Ready to install MongoDB 6.0.5 2008R2Plus SSL (64 bit)
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.
Back
Install
Cancel

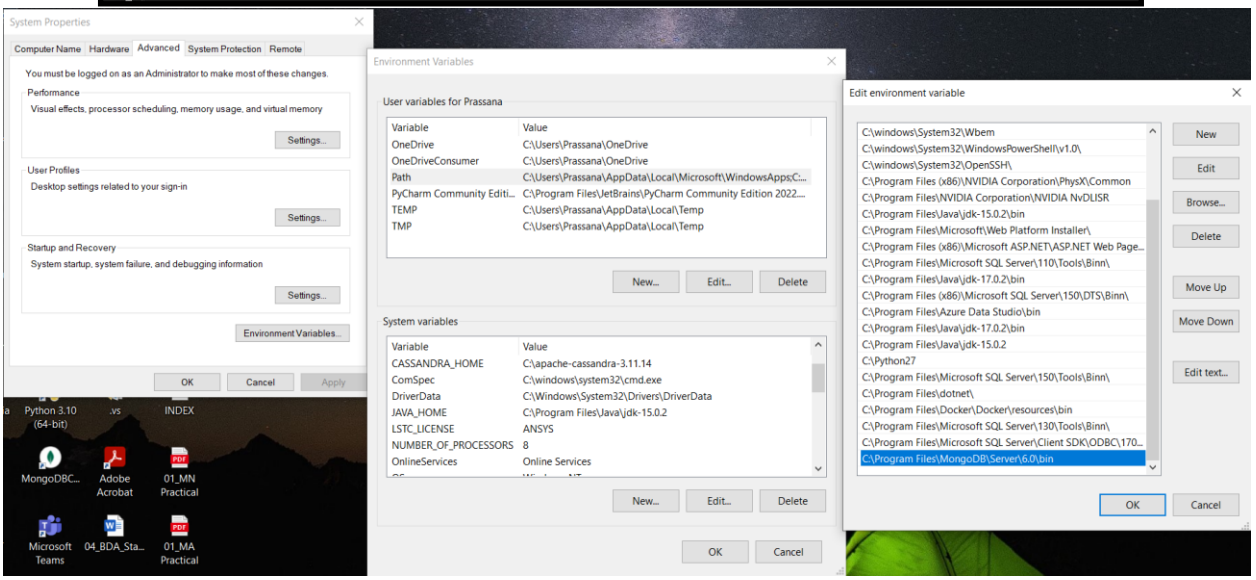

MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup
Installing MongoDB 6.0.5 2008R2Plus SSL (64 bit)
Please wait while the Setup Wizard installs MongoDB 6.0.5 2008R2Plus SSL (64 bit).
Status:
Back
Next
Cancel


MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup
Completed the MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup Wizard
Click the Finish button to exit the Setup Wizard.
Back
Finish
Cancel


MongoDB Compass
Connect Edit View Help
Compass
New connection
Saved connections
Recents
localhost:27017
Mar 25, 2023, 11:27 AM
New Connection
Connect to a MongoDB deployment
FAVORITE
URI
Edit Connection String
mongodb://localhost:27017
Advanced Connection Options
Save
Save & Connect
Connect
New to Compass and don't have a cluster?
If you don't already have a cluster, you can create one for free using MongoDB Atlas
CREATE FREE CLUSTER
How do I find my connection string in Atlas?
If you have an Atlas cluster, go to the Cluster view. Click the 'Connect' button for the cluster to which you wish to connect.
See example
How do I format my connection string?
See example

Set the path:

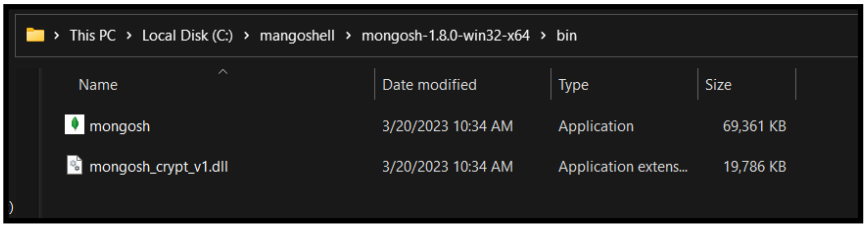

This PC › Local Disk (C:) › Program Files › MongoDB › Server › 6.0 › bin
Name
Date modified
Type
Size
InstallCompass 3/8/2023 6:45 PM Windows PowerSh... 2 KB
mongod.cfg 3/28/2023 12:07 PM CFG File 1 KB
mongod 3/8/2023 8:06 PM Application 55,071 KB
mongod.pdb 3/8/2023 8:06 PM Program Debug D... 825,668 KB
mongos 3/8/2023 7:30 PM Application 34,035 KB
mongos.pdb 3/8/2023 7:30 PM Program Debug D... 463,300 KB


System Properties
Environment Variables
Edit environment variable
C:\Program Files\MongoDB\Server\6.0\bin

```
Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Prassana> mongod --version
db version v6.0.5
Build Info: {
    "version": "6.0.5",
    "gitVersion": "c9a99c120371d4d4c52cbb15dac34a36ce8d3b1d",
    "modules": [],
    "allocator": "tcmalloc",
    "environment": {
        "distmod": "windows",
        "distarch": "x86_64",
        "target_arch": "x86_64"
    }
}
PS C:\Users\Prassana>
```

Step B: Download and Install Mongo shell

https://www.mongodb.com/try/download/shell

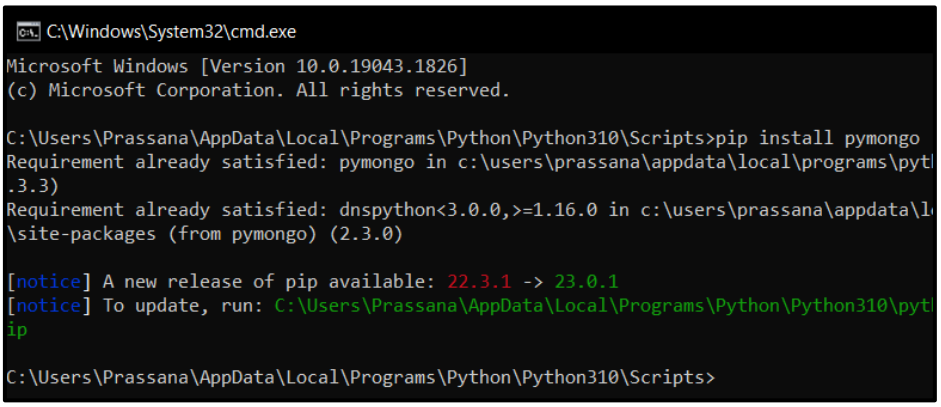



```
mongosh mongodb://127.0.0.
Please enter a MongoDB connection string (Default: mongodb://localhost/):

Current Mongosh Log ID: 64228dfc96dfb3899f8691cf
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1
.8.0
Using MongoDB:          6.0.5
Using Mongosh:          1.8.0

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2023-03-28T12:07:25.885+05:30: Access control is not enabled for the database. Read and write access to data and conf
iguration is unrestricted
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------

test>
```

```
test> show dbs
admin   40.00 KiB
config  84.00 KiB
local   72.00 KiB
```

Step C: Install PyMongo

python -m pip install pymongo



Program 1: Creating a Database, Collection and inserting one collection.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")print(client)
mydb = client["BigData"] print(client.list_database_names())
collection = mydb["student"]
dictionary = {'name':'Mansi','dept':'ITCS'}collection.insert_one(dictionary)
print(mydb.list_collection_names())
```

```
test> show dbs
BigData   72.00 KiB
admin     40.00 KiB
config   108.00 KiB
local     72.00 KiB
```

```
test> use BigData
switched to db BigData
```

```
BigData> show collections
student
```

```
BigData> db.student.find()
[
  {
    _id: ObjectId("642292fa8b3c8705a3197555")
    name: 'Aditi',
    dept: 'ITCS'
  }
]
```

Program 2: Inserting more collections

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")print(client)
mydb = client["BigData"] print(client.list_database_names())
collection = mydb["student"]

mylist=[{'name': 'Hrishi','dept':'BAF'},
{'name': 'Narayan','dept':'Mechanics'},
{'name': 'Vaishnavi','dept':'Commerce'},
{'name': 'Kunal','dept':'BSCIT'},]
x=collection.insert_many(mylist)

print(mydb.list_collection_names())
```

Mydb.py - C:\Users\ADITI\AppData\Local\Programs\Python\Python310\Mydb.py (3.10.8)

File Edit Format Run Options Window Help

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
print(client)
mydb = client["BigData"]
print(client.list_database_names())
collection = mydb["student"]

mylist=[{'name': 'Hrishi','dept':'BAF'},
        {'name': 'Narayan','dept':'Mechanics'},
        {'name': 'Vaishnavi','dept':'Commerce'},
        {'name': 'Kunal','dept':'BSCIT'},]
x=collection.insert_many(mylist)

print(mydb.list_collection_names())
```

mongosh mongodb://127.0.0.

```
BigData> db.student.find()
[
  {
    _id: ObjectId("642292fa8b3c8705a3197555"),
    name: 'Aditi',
    dept: 'ITCS'
  },
  {
    _id: ObjectId("6422956c41c3f3df68530965"),
    name: 'Hrishi',
    dept: 'BAF'
  },
  {
    _id: ObjectId("6422956c41c3f3df68530966"),
    name: 'Narayan',
    dept: 'Mechanics'
  },
  {
    _id: ObjectId("6422956c41c3f3df68530967"),
    name: 'Vaishnavi',
    dept: 'Commerce'
  },
  {
    _id: ObjectId("6422956c41c3f3df68530968"),
    name: 'Kunal',
    dept: 'BSCIT'
  }
]
BigData>
```

Program 3: To show all the collections.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")print(client)
mydb = client["BigData"]
print(client.list_database_names())collection =
mydb["student"]

alldocs=collection.find()for item in
alldocs:
print(item)
```

Mydb.py - C:\Users\ADITI\AppData\Local\Programs\Python\Python310\Mydb.py (3.10.8)

Edit Format Run Options Window Help

```
port pymongo

ient = pymongo.MongoClient("mongodb://localhost:27017/")
int(client)
db = client["BigData"]
int(client.list_database_names())
llection = mydb["student"]

ldocs=collection.find()
r item in alldocs:
  print(item)
```

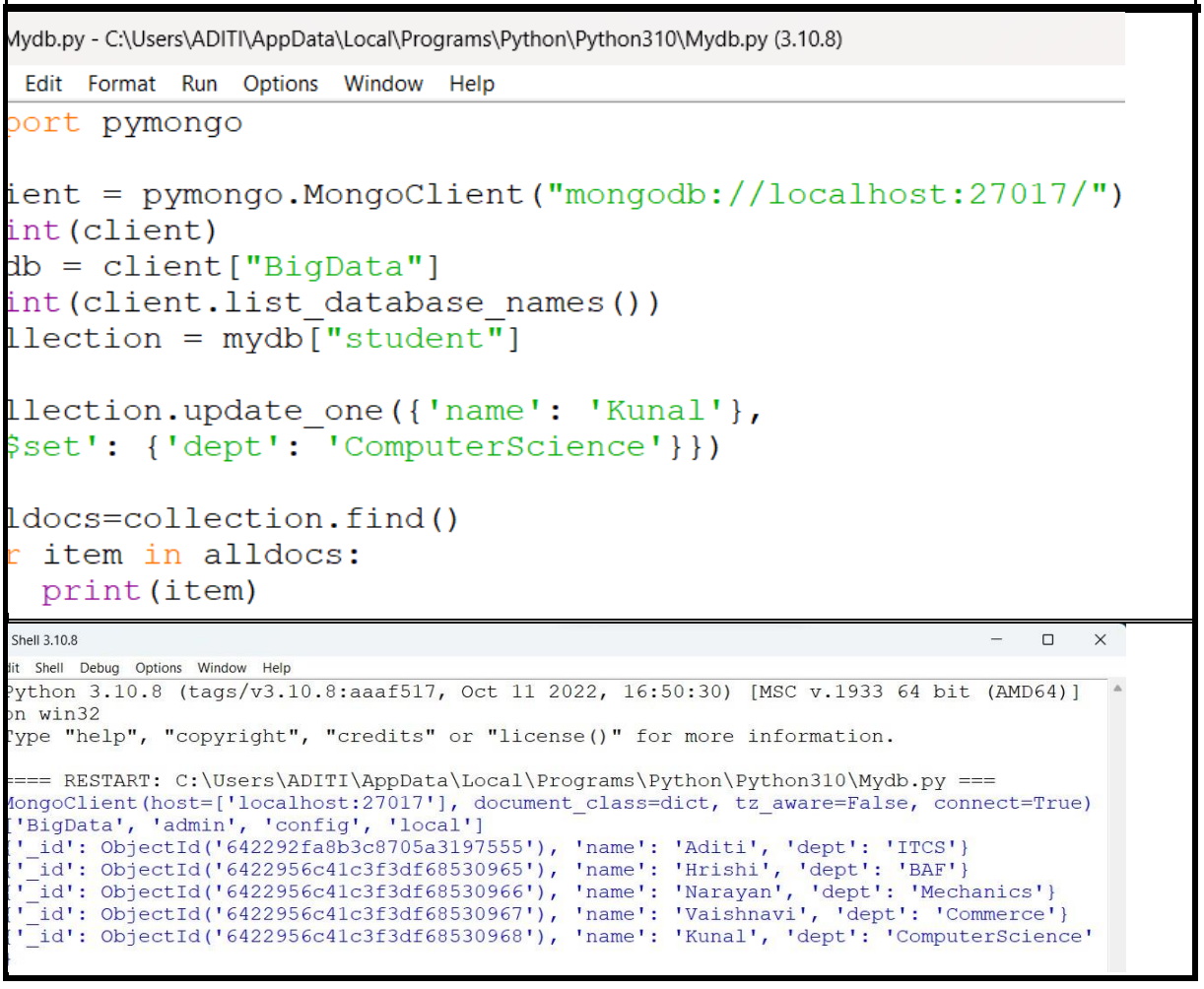
Shell 3.10.8

dit Shell Debug Options Window Help

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (AMD64)
on win32
Type "help", "copyright", "credits" or "license()" for more information.

==== RESTART: C:\Users\ADITI\AppData\Local\Programs\Python\Python310\Mydb.py ===
MongoClient(host=['localhost:27017'], document_class=dict, tz_aware=False, connect=Tru
['BigData', 'admin', 'config', 'local']
{'_id': ObjectId('642292fa8b3c8705a3197555'), 'name': 'Aditi', 'dept': 'ITCS'}
{'_id': ObjectId('6422956c41c3f3df68530965'), 'name': 'Hrishi', 'dept': 'BAF'}
{'_id': ObjectId('6422956c41c3f3df68530966'), 'name': 'Narayan', 'dept': 'Mechanics'}
{'_id': ObjectId('6422956c41c3f3df68530967'), 'name': 'Vaishnavi', 'dept': 'Commerce'}
{'_id': ObjectId('6422956c41c3f3df68530968'), 'name': 'Kunal', 'dept': 'BSCIT'}
```

Program 4: To update one collection

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")print(client)
mydb = client["BigData"]
print(client.list_database_names())collection =
mydb["student"]

collection.update_one({'name': 'Kunal'},
{'$set': {'dept': 'ComputerScience'}})

alldocs=collection.find()for item in
alldocs:
print(item)
```



Program 5: To delete one collection

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")print(client)
mydb = client["BigData"]
print(client.list_database_names())collection =
mydb["student"]

collection.delete_one({"name": "Aditi"})

alldocs=collection.find()
for item in alldocs:
print(item)
```

Mydb.py - C:\Users\ADITI\AppData\Local\Programs\Python\Python310\Mydb.py (3.10.8)

Edit Format Run Options Window Help

```
port pymongo

ient = pymongo.MongoClient("mongodb://localhost:27017/")
int(client)
db = client["BigData"]
int(client.list_database_names())
llection = mydb["student"]

llection.delete_one({"name": "Aditi"})

ldocs=collection.find()
r item in alldocs:
  print(item)
```

Shell 3.10.8

it Shell Debug Options Window Help

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (AMD64)] o
 win32
Type "help", "copyright", "credits" or "license()" for more information.

=== RESTART: C:\Users\ADITI\AppData\Local\Programs\Python\Python310\Mydb.py ===
MongoClient(host=['localhost:27017'], document_class=dict, tz_aware=False, connect=True)
['BigData', 'admin', 'config', 'local']
{'_id': ObjectId('6422956c41c3f3df68530965'), 'name': 'Hrishi', 'dept': 'BAF'}
{'_id': ObjectId('6422956c41c3f3df68530966'), 'name': 'Narayan', 'dept': 'Mechanics'}
{'_id': ObjectId('6422956c41c3f3df68530967'), 'name': 'Vaishnavi', 'dept': 'Commerce'}
{'_id': ObjectId('6422956c41c3f3df68530968'), 'name': 'Kunal', 'dept': 'ComputerScience'}
```

Practical No: 06

A] Classification Model

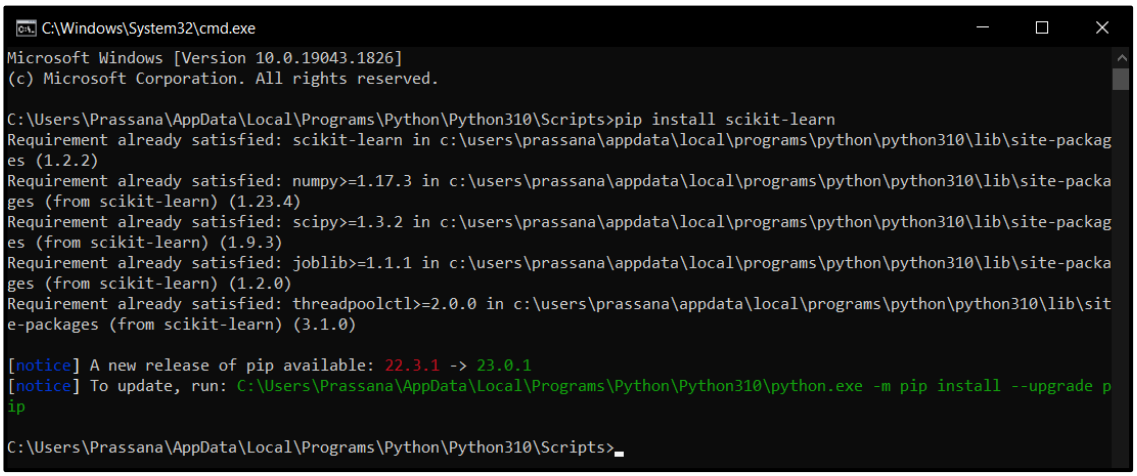
Aim: a) Install relevant packages for classification.
b) Choose Classifier for a classification problem.
c) Evaluate the performance of the classifier.

Prerequisites:

1. Scam.csv
2. Scikit-learn Library

Theory:

sklearn.feature_extraction.text.CountVectorizer	Convert a collection of text documents to a matrix of token counts.
sklearn.naive_bayes.MultinomialNB	The multinomial Naive Bayes classifier is suitable for classificationwith discrete features(e.g., wordcounts for text classification). The m ultinomial distribution normallyrequires integer feature counts.However, in practice, fractionalcounts such as tf-idf may also work.
Scikit-learn Library	Scikit-learn(Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python.



Building a spam classification model

1. Split the data into train and test sets.
2. Use Sklearn built-in classifiers to build the models.
3. Train the data on the model.
4. Make predictions on new data.

Program Code:

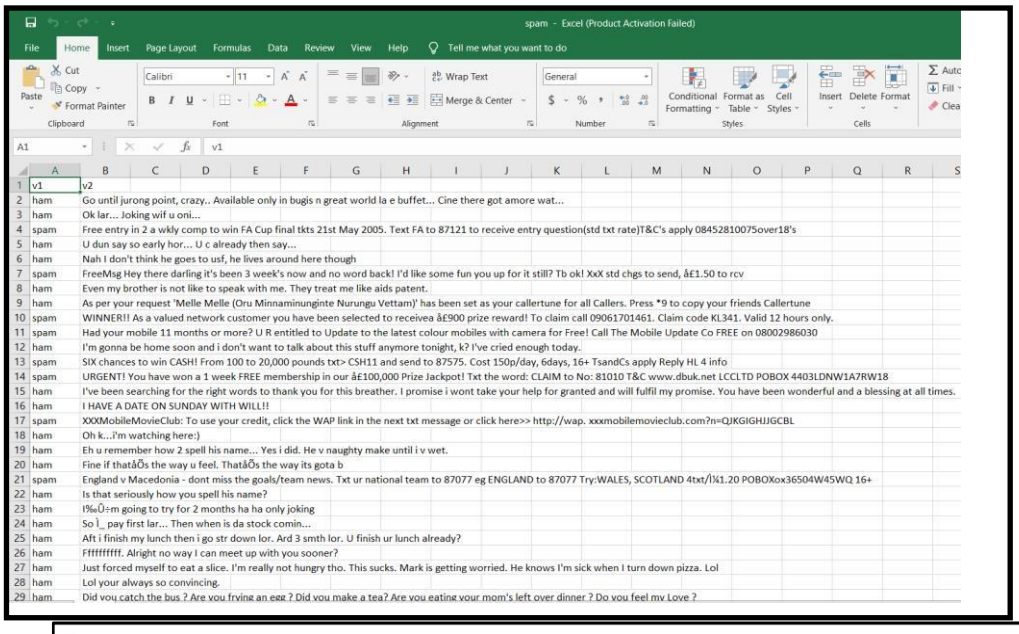
```
import numpy as np
import pandas as
pdimport os
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB

#Load Dataset
df = pd.read_csv('spam.csv',encoding='latin-1')

#Keep only necessary columns
df = df[['v2','v1']]

#Rename columns
df.columns=
['SMS','Type']

# Let's process the text data
# Instantiate count
vectorizercountvec =
CountVectorizer(
ngram_range = (1,4), stop_words =
'english', strip_accents =
'unicode',max_features = 1000
)

# Create bag of words
bow = countvec.fit_transform(df.SMS)

# Prepare training data
X_train = bow.toarray()
y_train = df.Type.values

# Instantiate
classifiermnb =
MultinomialNB()

# Train the classifier/Fit the model
mnb.fit(X_train,y_train)

# Testing
text = countvec.transform(['Free gifts for all'])
print(mnb.predict(text))
```

Pract6.py - I:/Msc part I/Semester II/BDA/BDA practical/practical 06/Pract6.py (3.10.8)

File Edit Format Run Options Window Help

```
import numpy as np
import pandas as pd
import os
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB

#Load Dataset
df = pd.read_csv("I:\Msc part 1\Semester II\BDA\BDA practical\practical 06\spam.csv",
                 encoding='latin-1')

#Keep only necessary columns
df = df[['v2','v1']]

#Rename columns
df.columns = ['SMS','Type']

#Let's process the text data
#Instantiate count vectorizer
countvec = CountVectorizer(ngram_range=(1,4),stop_words='english',
                           strip_accents='unicode',max_features=1000)

#Create bag of words
bow = countvec.fit_transform(df.SMS)

#Prepare training data
X_train = bow.toarray()
y_train = df.Type.values

#Instantiate classifier
mnb = MultinomialNB()

#Train the classifier/Fit the model
mnb.fit(X_train,y_train)

#Testing
text = countvec.transform(['free gifts for all'])
print(mnb.predict(text))
```

Output:



spam - Excel (Product Activation Failed)

	A	B
1	v1	v2
2	ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
3	ham	Ok lar... Joking wif u oni...
4	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's
5	ham	U dun say so early hor... U c already then say...
6	ham	Nah I don't think he goes to usf, he lives around here though
7	spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, å£1.50 to rcv
8	ham	Even my brother is not like to speak with me. They treat me like aids patent.
9	ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callertune
10	spam	WINNER!! As a valued network customer you have been selected to receivea å£900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.
11	spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
12	ham	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.
13	spam	SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info
14	spam	URGENT! You have won a 1 week FREE membership in our å£100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403LDNW1A7RW18
15	ham	I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonderful and a blessing at all times.
16	ham	I HAVE A DATE ON SUNDAY WITH WILL!!
17	spam	XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap. xxxmobilemovieclub.com?n=QJKGIGHJJGCBL
18	ham	Oh k...i'm watching here:)
19	ham	Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.
20	ham	Fine if thatâ€s the way u feel. Thatâ€s the way its gota b
21	spam	England v Macedonia - dont miss the goals/team news. Txt ur national team to 87077 eg ENGLAND to 87077 Try:WALES, SCOTLAND 4txt/ì¼1.20 POBOXox36504W45WQ 16+
22	ham	Is that seriously how you spell his name?
23	ham	I‰Û÷m going to try for 2 months ha ha only joking
24	ham	So Ì_ pay first lar... Then when is da stock comin...
25	ham	Aft i finish my lunch then i go str down lor. Ard 3 smth lor. U finish ur lunch already?
26	ham	Ffffffffff. Alright no way I can meet up with you sooner?
27	ham	Just forced myself to eat a slice. I'm really not hungry tho. This sucks. Mark is getting worried. He knows I'm sick when I turn down pizza. Lol
28	ham	Lol your always so convincing.
29	ham	Did you catch the bus ? Are you frying an egg ? Did you make a tea? Are you eating your mom's left over dinner ? Do you feel my Love ?

IDLE Shell 3.10.8

File Edit Shell Debug Options Window Help

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: I:/Msc part 1/Semester II/BDA/BDA practical/practical 06/Pract6.py =
['spam']
>>>
```

B] CLUSTERING MODEL

Aim: Use Clustering algorithm for unsupervised classification & plot the cluster data.

Theory:

- **Clustering**

Clustering is a set of techniques used to partition data into groups, or clusters. Clusters are loosely defined as groups of data objects that are more similar to other objects in their cluster than they are to data objects in other clusters.

- **K-Means Algorithm**

It is an iterative algorithm that divides the unlabelled dataset into k different clusters in such away that each dataset belongs to only one group that has similar properties.

- **K-Means Clustering**

→ K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabelled dataset into different clusters. Here K defines the number of predefined clusters that need to becreated in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.
→ It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabelled dataset on its own without the need for any training.
→ It is a centroid-based algorithm, where each cluster is associated with a centroid. The mainaim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.
→ K-means is the most frequently used form of clustering due to its speed and simplicity. Thek-means algorithm captures the insight that each point in a cluster should be near to the centreof that cluster.

```
# import statements
from sklearn.datasets import make_blobsimport
numpy as np
import matplotlib.pyplot as plt
```

make_blobs(), a convenience function in scikit-learn used to generate synthetic clusters. It isused to generate data points.
make_blobs() uses these parameters:

- n_samples is the total number of samples to generate.
- n_featuresint is the number of features for each sample.
- centers is the number of centers to generate.
- cluster_std is the standard deviation.
- random_state determines random number generation for dataset creation. Pass an int for reproducible output across multiple function calls.

make_blobs() returns a tuple of two values:

1. A two-dimensional NumPy array with the x- and y-values for each of the samples
2. A one-dimensional NumPy array containing the cluster labels for each sample.

```
# create blobs
data = make_blobs(n_samples=400, n_features=2, centers=4,
cluster_std=1.6,
random_state=50)
# create np array for data points
points = data[0]
```

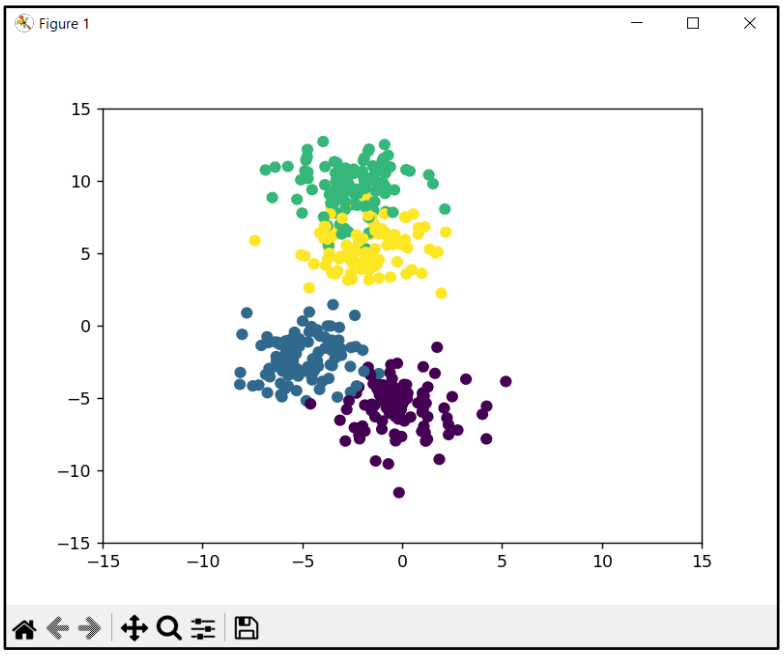
we'll use the first 2 columns (0,1) of our data.

- cmaps.viridis is a matplotlib.colors.ListedColormap
- cmap stands for colormap and it's a colormap instance or registered colormap name (cmap will only work if c is an array of floats)
- maps.viridis is a matplotlib.colors.ListedColormap

```
# create scatter plot

plt.scatter(data[0][:,0], data[0][:,1], c=data[1],
cmap='viridis')
plt.xlim(-15,15)
plt.ylim(-15,15)
```

Output:

You can see our "blobs" below:



We have four coloured clusters, but there is some overlap with the two clusters on top, as wellas the two clusters on the bottom.

Now we apply the k-means algorithm on our blob.

Applying K-Means Algorithm:

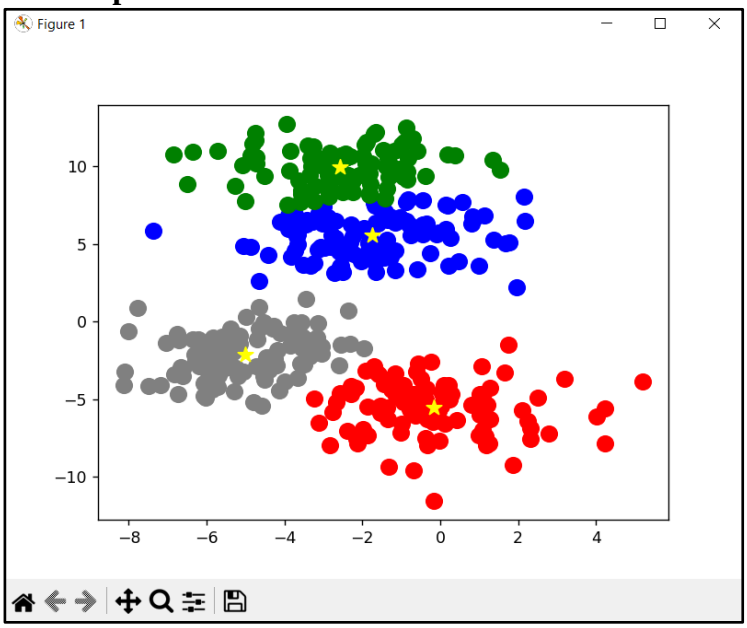
1. Initialize centroids – This is done by randomly choosing K no of points, the points can be present in the dataset or also random points.
2. Assign Clusters – The clusters are assigned to each point in the dataset by calculating their distance from the centroid and assigning it to the centroid with minimum distance.
3. Re-calculate the centroids – Updating the centroid by calculating the centroid of each clusterwe have created.

```
# import KMeans

from sklearn.cluster import KMeans
# create kmeans object
kmeans = KMeans(n_clusters=4)
# fit kmeans object to data
kmeans.fit(points)
# print location of clusters learned by kmeans object
```

```
[[-1.74809641  5.54583068]
 [-0.17419501 -5.53888403]
 [-5.01621736 -2.11522242]
 [-2.58575308  9.9704047 ]]
```

```
Visualize the Clusters
# save new clusters for chart
y_km = kmeans.fit_predict(points)
plt.scatter(points[y_km ==0,0], points[y_km == 0,1], s=100,
c='red')
plt.scatter(points[y_km ==1,0], points[y_km == 1,1], s=100,
c='blue')
plt.scatter(points[y_km ==2,0], points[y_km == 2,1], s=100,
c='grey')
plt.scatter(points[y_km ==3,0], points[y_km == 3,1], s=100,
c='green')
plt.scatter(kmeans.cluster_centers_[:, 0],
kmeans.cluster_centers_[:, 1],s=100,color='yellow',
marker="*")
```

Output:



It works pretty decently and shows 4 clusters of different coloured data points with their centroids marked by '*'.

Note: You can change sample size, number of clusters, graph colours

Practical No: 07

Aim: Configure the Hive and implement the application in Hive.

Theory:

Apache Hive is an enterprise data warehouse system used to query, manage, and analyse data stored in the Hadoop Distributed File System.

The Hive Query Language (HiveQL) facilitates queries in a Hive command-line interface shell. Hadoop can use HiveQL as a bridge to communicate with relational database management systems and perform tasks based on SQL-like commands.

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analysing easy.

Initially Hive was developed by Facebook, later the Apache Software Foundation took it up and developed it further as an open source under the name Apache Hive. It is used by different companies. For example, Amazon uses it in Amazon Elastic MapReduce.

Pre-requisites:

• Java must be installed.
• Hadoop must be installed and the Hadoop cluster must be configured

Steps:

Verifying JAVA Installation

Java must be installed on your system before installing Pig. Let us verify java installation

command: java –version

```
venom16@comp196: /home/venom
venom@comp196:~$ java -version
openjdk version "1.8.0_362"
OpenJDK Runtime Environment (build 1.8.0_362-8u362-ga-0ubuntu1~22.04-b09)
OpenJDK 64-Bit Server VM (build 25.362-b09, mixed mode)
```

Verifying Hadoop Installation

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

command: hadoop version

```
venom16@comp196:/home/venom$ hadoop version
Hadoop 3.3.1
Source code repository https://github.com/apache/hadoop.git -r a3b9c37a397ad4188041
dd80621bdeefc46885f2
Compiled by ubuntu on 2021-06-15T05:13Z
Compiled with protoc 3.7.1
From source with checksum 88a4ddb2299aca054416d6b7f81ca55
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-3.3.
1.jar
```

Step 1: Start hadoop

Command 1: su venom16

```
venom16@comp196: ~
venom@comp196:~$ su venom16
Password:
```

Command 2: sudo service ssh restart

```
venom16@comp196:/home/venom$ sudo service ssh restart
[sudo] password for venom16:
 * Restarting OpenBSD Secure Shell server sshd                          [ OK ]
venom16@comp196:/home/venom$ ssh localhost
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 4.4.0-19041-Microsoft x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Last login: Wed Mar  8 14:07:46 2023 from 127.0.0.1
```

Command 3: ssh localhost

```
venom16@comp196:/home/venom$ ssh localhost
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 4.4.0-19041-Microsoft x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Last login: Wed Mar  8 14:07:46 2023 from 127.0.0.1
```

Command 4: start-all.sh

```
venom16@comp196:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as venom16 in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [comp196]
Starting resourcemanager
Starting nodemanagers
```

Note: If you have preinstalled Hive on your system 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/hduser1

Command 5: cd /home/venom16

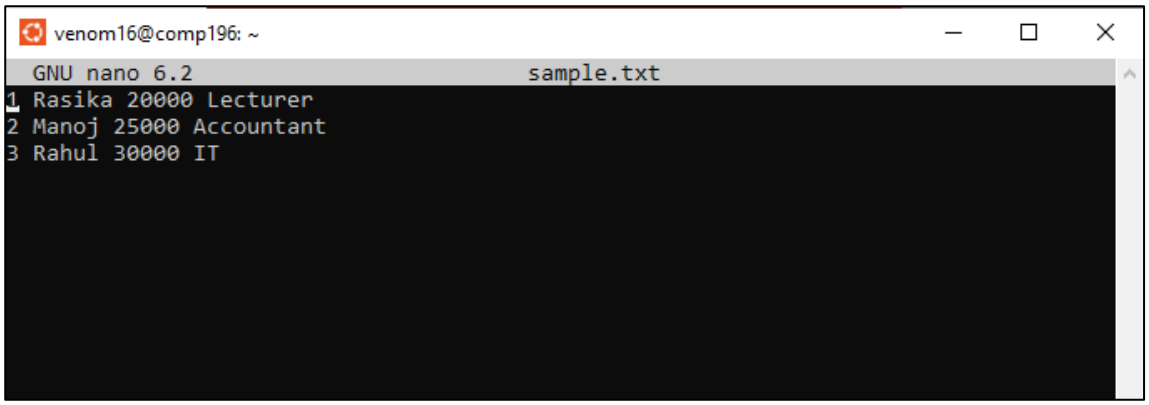
```
venom16@comp196:~$ cd /home/venom16
```

Step 3: Create a text file sample.txt:
Command 6: sudo nano sample.txt

```
venom16@comp196:~$ sudo nano sample.txt
[sudo] password for venom16:
```

Step 4: Add following details in the sample.txt: (comma separated values)
1 Rasika 20000 Lecturer
2 Hrishi 25000 Accountant
3 Rahul 30000 SoftwareEngineer

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



Now we will download and setup Hive:
Step 6: Change the current working directory to /usr/local
Command 7: cd /usr/local

```
venom16@comp196:~$ cd /usr/local
```

Step 7: Download the new release of Hive.
Download the compressed Hive files using and the wget command followed by the download path:

Command 8: sudo wget https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz

```
venom16@comp196:/usr/local$ sudo wget https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
--2023-03-21 11:16:30--  https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 88.99.95.219, 135.181.214.104, 2a01:4f8:10a:201a::2, ...
Connecting to downloads.apache.org (downloads.apache.org)|88.99.95.219|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 278813748 (266M) [application/x-gzip]
Saving to: 'apache-hive-3.1.2-bin.tar.gz.1'

apache-hive-3.1.2-bi 100%[===================>] 265.90M  1.01MB/s    in 4m 32s

2023-03-21 11:21:02 (1002 KB/s) - 'apache-hive-3.1.2-bin.tar.gz.1' saved [278813748/278813748]
```

Step 8: Once the download process is complete, untar the compressed hive package:

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

```
venom16@comp196:/usr/local$ sudo tar -xvzf apache-hive-3.1.2-bin.tar.gz
apache-hive-3.1.2-bin/LICENSE
apache-hive-3.1.2-bin/NOTICE
apache-hive-3.1.2-bin/RELEASE_NOTES.txt
apache-hive-3.1.2-bin/binary-package-licenses/asm-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.google.protobuf-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.ibm.icu.icu4j-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.sun.jersey-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.thoughtworks.paranamer-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/javax.transaction.transaction-api-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/javolution-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/jline-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/NOTICE
```

Step 9 : Use ls command to check if the file got extracted in the directory or not. The hive binary files are now located in the apache-hive-3.1.2-bin directory.

Command 10: ls

```
venom16@comp196:/usr/local$ ls
apache-hive-3.1.2-bin          bin     hadoop-3.3.1.tar.gz  man              pig-0.17.0.tar.gz.1
apache-hive-3.1.2-bin.tar.gz   etc     hive                 pig              sbin
apache-hive-3.1.2-bin.tar.gz.1 games   include              pig-0.17.0       share
apache-hive-3.1.2-bin.tar.gz.2 hadoop  lib                  pig-0.17.0.tar.gz src
venom16@comp196:/usr/local$
```

Step 10: Rename the extracted file from apache-hive-3.1.2-bin to hive.

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

```
venom16@comp196: ~
venom16@comp196:/usr/local$ sudo mv apache-hive-3.1.2-bin hive
```

Step 11: Use ls command to check if the file got renamed or not.

Command 12: ls

```
venom16@comp196: ~
venom16@comp196:/usr/local$ sudo mv apache-hive-3.1.2-bin hive
venom16@comp196:/usr/local$ ls
apache-hive-3.1.2-bin.tar.gz    etc     hadoop-3.3.1.tar.gz  lib   share
apache-hive-3.1.2-bin.tar.gz.1  games   hive                 man   src
bin                             hadoop  include              sbin
```

Step 12: Define or change permissions or modes on files and limit access to only those who are allowed access.

Command 13: sudo chmod 777 hive

```
venom16@comp196:/usr/local$ sudo chmod 777 hive
```

Step 13: Change the current working directory to / home/hduser
Command 14: cd /home/hduser1

```
venom16@comp196:/usr/local$ cd /home/venom16
```

Step 14: The $HIVE_HOME environment variable needs to direct the client shell to the apache-pig directory. Edit the .bashrc shell configuration file using a text editor of your choice (we will be using nano):
Command 15: nano .bashrc

```
venom16@comp196:~$ nano .bashrc
```

Step 15: Append the following Hive environment variables to the .bashrc file:
export HIVE_HOME=/usr/local/hive
export PATH=$PATH:$HIVE_HOME/bin

The Hadoop environment variables are located within the same file.

```
#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"
export HIVE_HOME=/usr/local/hive
export PATH=$PATH:$HIVE_HOME/bin
```

Step 16: Save "Ctrl S" and exit the file by pressing "Ctrl X" followed by "Y" and "Enter" keys.
Step 17: Save and exit the .bashrc file once you add the Pig variables. Apply the changes to the current environment with the following command:

Command 16: source .bashrc

```
venom16@comp196:~$ source .bashrc
```

IV. Edit hive-config.sh file

Apache Hive needs to be able to interact with the Hadoop Distributed File System. Access the hive-config.sh file using the previously created $HIVE_HOME variable:

Note: The hive-config.sh file is in the bin directory within your Hive installation directory.

Step 18: Then go to hive bin directory by command:
Command 17: cd /usr/local/hive/bin

```
venom16@comp196:~$ cd /usr/local/hive/bin
```

Step 19: Open and edit hive-config.sh file.
Command 18: sudo nano hive-config.sh

```
venom16@comp196:/usr/local/hive/bin$ sudo nano hive-config.sh
```

Step 20: Add the following line to hive-config.sh (Add the HADOOP_HOME variable and the full path to your Hadoop directory.)
export HADOOP_HOME=/usr/local/hadoop

```
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"
export HIVE_HOME=/usr/local/hive
export PATH=$PATH:$HIVE_HOME/bin

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

V. Create Hive Directories in HDFS:

Hive is installed now, but you need to first create some directories in HDFS for Hive to store its data. Create two separate directories to store data in the HDFS layer:

• The temporary, tmp directory is going to store the intermediate results of Hive processes.
• The warehouse directory is going to store the Hive related tables.

Create a tmp directory

Step 22: Create a tmp directory within the HDFS storage layer. This directory is going to store the intermediary data Hive sends to the HDFS:
Command 19: hdfs dfs -mkdir /tmp

```
venom16@comp196:~$ hdfs dfs -mkdir /tmp
mkdir: `/tmp': File exists
venom16@comp196:~$
```

Step 23: Add write and execute permissions to tmp group members:
Command 20: hdfs dfs -chmod g+w /tmp

```
venom16@comp196:~$ hdfs dfs -chmod g+w /tmp
venom16@comp196:~$ hdfs dfs -ls/
```

Step 24: Check if the permissions were added correctly:
Command 21: hdfs dfs -ls /
The output confirms that users now have write and execute permissions.

```
venom16@comp196:~$ hdfs dfs -ls /
Found 5 items
-rw-r--r--   1 venom16 supergroup         88 2023-03-08 14:13 /Mapreduce.txt
drwxr-xr-x   - venom16 supergroup          0 2023-03-08 14:14 /output
-rw-r--r--   1 venom16 supergroup         67 2023-03-27 13:56 /sample.txt
drwxrwxr-x   - venom16 supergroup          0 2023-03-27 14:28 /tmp
drwxr-xr-x   - venom16 supergroup          0 2023-03-21 11:25 /user
venom16@comp196:~$
```

Create warehouse Directory

Step 25: Create the warehouse directory within the /user/hive/ parent directory:
Command 22: hdfs dfs -mkdir -p /user/hive/warehouse

```
venom16@comp196:~$ hdfs dfs -mkdir -p /user/hive/warehouse
```

Step 26: Add write and execute permissions to warehouse group members
Command 23: hdfs dfs -chmod g+w /user/hive/warehouse

```
venom16@comp196:~$ hdfs dfs -chmod g+w /user/hive/warehouse
```

Step 27: Check if the permissions were added correctly:
Command 24: hdfs dfs -ls /user/hive

```
venom16@comp196:~$ hdfs dfs -ls /user/hive
Found 1 items
drwxrwxr-x   - venom16 supergroup          0 2023-03-21 11:25 /user/hive/warehouse
```

The output confirms that users now have write and execute permissions.

VI. Initialize Derby database

Apache Hive uses the Derby database to store metadata. Initialize the Derby database, from the Hive bin directory using the schematool command:

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

```
venom16@comp196:~$ HIVE_HOME/bin/schematool -dbType derby -initSchema
-bash: HIVE_HOME/bin/schematool: No such file or directory
```

Derby is the default metadata store for Hive.

Error: FUNCTION 'NUCLEUS_ASCII' already exists. (state=X0Y68,code=30000) org.apache.hadoop.hive.metastore.HiveMetaException: Schema initialization FAILED! Metastore state would be inconsistent !! * schemaTool failed *

Solution:
To avoid this just delete or move your metastore_db and try the below command.
$ mv metastore_db metastore_db.tmp
OR $ rm metastore_db

How to Fix guava Incompatibility Error in Hive

If the Derby database does not successfully initiate, you might receive an error with the following content:
"Exception in thread "main"
java.lang.NoSuchMethodError:com.google.common.base.Preconditions.checkArgument(ZLja
va/lang/String;Ljava/lang/Object;)V"
This error indicates that there is most likely an incompatibility issue between Hadoop and Hive guava versions.

Step 29: Locate the guava jar file in the Hive lib directory:
Command 26: ls $HIVE_HOME/lib

```
venom16@comp196:~$ ls $HIVE_HOME/lib
HikariCP-2.6.1.jar
ST4-4.0.4.jar
accumulo-core-1.7.3.jar
accumulo-fate-1.7.3.jar
accumulo-start-1.7.3.jar
accumulo-trace-1.7.3.jar
aircompressor-0.10.jar
ant-1.9.1.jar
ant-launcher-1.9.1.jar
antlr-runtime-3.5.2.jar
antlr4-runtime-4.5.jar
aopalliance-repackaged-2.5.0-b32.jar
apache-curator-2.12.0.pom
apache-jsp-9.3.20.v20170531.jar
apache-jstl-9.3.20.v20170531.jar
```

```
esri-geometry-api-2.0.0.jar
findbugs-annotations-1.3.9-1.jar
flatbuffers-1.2.0-3f79e055.jar
groovy-all-2.4.11.jar
gson-2.2.4.jar
guava-19.0.jar
hbase-client-2.0.0-alpha4.jar
hbase-common-2.0.0-alpha4-tests.jar
hbase-common-2.0.0-alpha4.jar
hbase-hadoop-compat-2.0.0-alpha4.jar
```

Step 30: Locate the guava jar file in the Hadoop lib directory as well:
Command 27: ls $HADOOP_HOME/share/hadoop/hdfs/lib
The two listed versions are not compatible and are causing the error.

```
hduser1@Rasika:~$ ls $HADOOP_HOME/share/hadoop/hdfs/lib
accessors-smart-2.4.2.jar
animal-sniffer-annotations-1.17.jar
asm-5.0.4.jar
audience-annotations-0.5.0.jar
avro-1.7.7.jar
checker-qual-2.5.2.jar
commons-beanutils-1.9.4.jar
```

```
failureaccess-1.0.jar
gson-2.2.4.jar
guava-27.0-jre.jar
hadoop-annotations-3.3.1.jar
hadoop-auth-3.3.1.jar
hadoop-shaded-guava-1.1.1.jar
hadoop-shaded-protobuf_3_7-1.1.1.jar
htrace-core4-4.1.0-incubating.jar
```

Step 31: Remove the existing guava file from the Hive lib directory:
Command 28: sudo rm /usr/local/hive/lib/guava-19.0.jar

```
venom16@comp196: ~
venom16@comp196:~$ sudo rm /usr/local/hive/lib/guava-19.0.jar
```

Step 32: Copy the guava file from the Hadoop lib directory to the Hive lib directory:
Command 29:
sudo cp $HADOOP_HOME/share/hadoop/common/lib/guava-27.0-jre.jar /usr/local/hive/lib/

```
venom16@comp196:~$ sudo cp $HADOOP_HOME/share/hadoop/common/lib/guava-27.0-jre.jar /usr/local/hive/lib/
```

Step 33: Use the schematool command once again to initiate the Derby database:
Command 30: /usr/local/hive/bin/schematool -initSchema -dbType derby

```
venom16@comp196:~$ /usr/local/hive/bin/schematool -initSchema -dbType derby
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Metastore connection URL:        jdbc:derby:;databaseName=metastore_db;create=true
Metastore Connection Driver :    org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:       APP
Starting metastore schema initialization to 3.1.0
Initialization script hive-schema-3.1.0.derby.sql
```

```
venom16@comp196: ~
Initialization script completed
schemaTool completed
```

VII. Launch Hive Client Shell on Ubuntu

Step 34: Start the Hive command-line interface using the following commands:
Command 31: cd $HIVE_HOME

```
venom16@comp196: ~
Initialization script completed
schemaTool completed
venom16@comp196:~$ cd $HIVE_HOME
```

Command 32: hive
You are now able to issue SQL-like commands and directly interact with HDFS.

```
venom16@comp196: ~
venom16@comp196:~$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 553ca4c8-a1a8-431f-9047-0ea362d0e938

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-3.1.2.jar!/hive-log4j2.properties Async: true
Hive Session ID = 7de7ff52-bbb1-4d8f-b1c3-ebb1aafccbdb
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X rele
ases.
```

Perform SQL commands on 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 ‘ ‘ LINES TERMINATED BY ‘\n’ STORED AS TEXTFILE;

```
hive> show tables;
OK
Time taken: 0.597 seconds
hive> 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;
OK
Time taken: 0.755 seconds
```

Command 34:

LOAD DATA LOCAL INPATH "/home/hduser1/sample.txt" overwrite into table employee;

```
hive> LOAD DATA LOCAL INPATH "/home/venom16/sample.txt" overwrite into table employee;
Loading data to table default.employee
OK
```

Command 35: select * from employee;

```
hive> select * from employee;
OK
1       Rasika  20000   Lecturer
2       Manoj   25000   Accountant
3       Rahul   30000   IT
NULL    NULL    NULL    NULL
Time taken: 2.054 seconds, Fetched: 4 row(s)
```

Step 36: Then stop hadoop and close.

Command 36: quit;

```
hive> quit
    > ;
```

Working with Hive:

```
venom16@comp196: ~
venom16@comp196:~$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 1b3bed17-4c05-4f17-ad99-eec644c5c598

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-3.1.2.jar!/hive-log4j2.properties Async: true
Hive Session ID = 32d630b4-448e-4197-b37a-3fc47bc49d54
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez)
ases.
hive> create database Company;\
    > ;
OK
Time taken: 0.548 seconds
NoViableAltException(-1@[])
        at org.apache.hadoop.hive.ql.parse.HiveParser.statement(HiveParser.java:1387)
        at org.apache.hadoop.hive.ql.parse.ParseDriver.parse(ParseDriver.java:220)
        at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:74)
        at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:67)
        at org.apache.hadoop.hive.ql.Driver.compile(Driver.java:616)
        at org.apache.hadoop.hive.ql.Driver.compileInternal(Driver.java:1826)
        at org.apache.hadoop.hive.ql.Driver.compileAndRespond(Driver.java:1773)
        at org.apache.hadoop.hive.ql.Driver.compileAndRespond(Driver.java:1768)
        at org.apache.hadoop.hive.ql.reexec.ReExecDriver.compileAndRespond(ReExecDriver.java:126)
        at org.apache.hadoop.hive.ql.reexec.ReExecDriver.run(ReExecDriver.java:214)
        at org.apache.hadoop.hive.cli.CliDriver.processLocalCmd(CliDriver.java:239)
        at org.apache.hadoop.hive.cli.CliDriver.processCmd(CliDriver.java:188)
        at org.apache.hadoop.hive.cli.CliDriver.processLine(CliDriver.java:402)
        at org.apache.hadoop.hive.cli.CliDriver.executeDriver(CliDriver.java:821)
        at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:759)
        at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:683)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:323)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:236)
FAILED: ParseException line 2:0 cannot recognize input near '<EOF>' '<EOF>' '<EOF>'
```

```
venom16@comp196: ~
hive> use Company;
OK
Time taken: 0.019 seconds
```

```
hive> create table employee(id int, name String, salary String);
OK
Time taken: 0.523 seconds
hive> describe employee;
OK
id                      int
name                    string
salary                  string
Time taken: 0.261 seconds, Fetched: 3 row(s)
```

```
hive> insert into employee values (101, 'Sophia', '1 Lakh');
Query ID = venom16_20230321115609_1d7bdb8b-7fed-4e16-a6d8-4355a15c1b0c
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2023-03-21 11:56:13,626 Stage-1 map = 0%,  reduce = 0%
2023-03-21 11:56:14,662 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local834427342_0001
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:54310/user/hive/warehouse/company.db/employee/.hive-staging_hive_2023-03-21_11-56-09_477_3699228217322367910-1/-ext-10000
Loading data to table company.employee
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 0 HDFS Write: 180 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 5.631 seconds
hive> insert into employee values(102, 'Shubham', '2 Lakh');
Query ID = venom16_20230321115641_0db9f2d7-f8d7-44ff-bed8-5390760e1eb6
Total jobs = 3
Launching Job 1 out of 3
```

```
Launching Job 1 out of 3
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2023-03-21 11:56:42,993 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local626568141_0002
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:54310/user/hive/warehouse/company.db/employee/.hive-staging_hive_2023-03-21_11-56-41_299_6444725407450948216-1/-ext-10000
Loading data to table company.employee
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 144 HDFS Write: 362 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 1.864 seconds
hive> select * from employee;
OK
101     Sophia  1 Lakh
102     Shubham 2 Lakh
Time taken: 0.11 seconds, Fetched: 2 row(s)
hive>
```

Practical No: 08

Aim: Implement an application that stores big data in Pig.

Theory:

Pig is a high-level platform or tool which is used to process the large datasets. It provides a high-level of abstraction for processing over the MapReduce. It provides a high-level scripting language, known as Pig Latin which is used to develop the data analysis codes. This language provides various operators using which programmers can develop their own functions for reading, writing, and processing data. First, to process the data which is stored in the HDFS, the programmers will write the scripts using the Pig Latin Language.

Internally Pig Engine (a component of Apache Pig) converted all these scripts into a specific map and reduce task. But these are not visible to the programmers in order to provide a high- level of abstraction. Pig Latin and Pig Engine are the two main components of the Apache Pig tool. The result of Pig always stored in the HDFS.

Need of Pig: One limitation of MapReduce is that the development cycle is very long. Writing the reducer and mapper, compiling packaging the code, submitting the job and retrieving the output is a time-consuming task. Apache Pig reduces the time of development using the multi-query approach. Also, Pig is beneficial for the programmers who are not from Java background. 200 lines of Java code can be written in only 10 lines using the Pig Latin language. Programmers who have SQL knowledge needed less effort to learn Pig Latin

Pre-requisites:

• Java must be installed.
• Hadoop must be installed and the Hadoop cluster must be configured

Steps:

Verifying JAVA Installation

Java must be installed on your system before installing Pig. Let us verify java installation

command: java –version

```
venom@comp196:~$ java -version
openjdk version "1.8.0_362"
OpenJDK Runtime Environment (build 1.8.0_362-8u362-ga-0ubuntu1~22.04-b09)
OpenJDK 64-Bit Server VM (build 25.362-b09, mixed mode)
venom@comp196:~$
```

Step 1: Start hadoop

Command 1: su venom16

```
venom16@comp196: ~
venom@comp196:~$ su venom16
Password:
```

Command 2: sudo service ssh restart

```
venom16@comp196:/home/venom$ sudo service ssh restart
[sudo] password for venom16:
 * Restarting OpenBSD Secure Shell server sshd                          [ OK ]
venom16@comp196:/home/venom$ ssh localhost
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 4.4.0-19041-Microsoft x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Last login: Wed Mar  8 14:07:46 2023 from 127.0.0.1
```

Command 3: ssh localhost

```
venom16@comp196:/home/venom$ ssh localhost
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 4.4.0-19041-Microsoft x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Last login: Wed Mar  8 14:07:46 2023 from 127.0.0.1
```

Command 4: start-all.sh

```
venom16@comp196:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as venom16 in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [comp196]
Starting resourcemanager
Starting nodemanagers
```

Step 2: Change the current working directory to / home/hduser1

Command 5: cd /home/venom16

```
venom16@comp196:~$ cd /home/venom16
```

Step 3: Create a text file sample.txt:

Command 6: sudo nano sample.txt

```
venom16@comp196:~$ sudo nano sample.txt
[sudo] password for venom16:
```

Step 4: Add following details in the sample.txt: (comma separated values)

1,Rasika,20000,Lecturer

2,Hrishi,25000,Accountant

3,Rahul,30000,SoftwareEngineer

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

```
venom16@comp196: ~
  GNU nano 6.2
1,Rasika,20000,Lecturer
2,Manoj,25000,Accountant
3,Rahul,30000,IT
```

Now we will download and setup Pig:

Step 6: Change the current working directory to /usr/local

Command 7: cd /usr/local

```
venom16@comp196:~$ cd /usr/local
venom16@comp196:/usr/local$
```

Step 7: Download the new release of Apache Pig.

In my case I have downloaded the pig-0.17.0.tar.gz version of Pig which is latest and about 220MB in size

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

```
venom16@comp196:/usr/local$ sudo wget https://downloads.apache.org/pig/pig-0.17.0/pig-0.17.0.tar.gz
[sudo] password for venom16:
--2023-03-28 10:44:32--  https://downloads.apache.org/pig/pig-0.17.0/pig-0.17.0.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 135.181.214.104, 88.99.95.219, 2a01:4f8:10a:
201a::2, ...
Connecting to downloads.apache.org (downloads.apache.org)|135.181.214.104|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 230606579 (220M) [application/x-gzip]
Saving to: 'pig-0.17.0.tar.gz.1'

pig-0.17.0.tar.gz.1       100%[=================================>] 219.92M  1.02MB/s    in 3m 39s

2023-03-28 10:48:11 (1.01 MB/s) - 'pig-0.17.0.tar.gz.1' saved [230606579/230606579]
```

Step 8: Once the download process is complete, untar the compressed pig package:

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

```
venom16@comp196: /usr/local
venom16@comp196:/usr/local$ sudo tar -xvzf pig-0.17.0.tar.gz
pig-0.17.0/
pig-0.17.0/bin/
pig-0.17.0/conf/
pig-0.17.0/contrib/
pig-0.17.0/contrib/piggybank/
pig-0.17.0/contrib/piggybank/java/
pig-0.17.0/contrib/piggybank/java/build/
pig-0.17.0/contrib/piggybank/java/build/classes/
pig-0.17.0/contrib/piggybank/java/build/classes/org/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/datetime/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/datetime/convert/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/datetime/diff/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/datetime/truncate
/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/decode/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/math/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/stats/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/string/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/util/
pig-0.17.0/contrib/piggybank/java/build/classes/org/apache/pig/piggybank/evaluation/util/apachelogpar
ser/
```

Step 9 : Use ls command to check if the file got extracted in the directory or not. The pig binary files are now located in the pig-0.17.0 directory.

Command 10: ls

```
venom16@comp196:/usr/local$ ls
apache-hive-3.1.2-bin.tar.gz    etc     hadoop-3.3.1.tar.gz  lib  pig-0.17.0          sbin
apache-hive-3.1.2-bin.tar.gz.1  games   hive                 man  pig-0.17.0.tar.gz   share
bin                             hadoop  include              pig  pig-0.17.0.tar.gz.1 src
venom16@comp196:/usr/local$
```

Step 10: Rename the extracted file from pig-0.17.0 to pig

Command 11: sudo mv pig-0.17.0 pig

```
venom16@comp196:/usr/local$ sudo mv pig-0.17.0 pig
```

Step 11: Use ls command to check if the file got renamed or not.

Command 12: ls -l

```
venom16@comp196:/usr/local$ ls -l
total 1585972
-rw-r--r-- 1 root    root   278813748 Aug 27  2019 apache-hive-3.1.2-bin.tar.gz
-rw-r--r-- 1 root    root   278813748 Aug 27  2019 apache-hive-3.1.2-bin.tar.gz.1
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 bin
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 etc
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 games
drwxr-xr-x 1 venom16 hadoop      4096 Mar  8 13:51 hadoop
-rw-r--r-- 1 root    root   605187279 Jun 15  2021 hadoop-3.3.1.tar.gz
drwxrwxrwx 1 root    root        4096 Mar 21 11:48 hive
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 include
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 lib
lrwxrwxrwx 1 root    root           9 Jan  4 03:10 man -> share/man
drwxrwxrwx 1 root    root        4096 Jun  2  2017 pig
drwxr-xr-x 1 root    root        4096 Jun  2  2017 pig-0.17.0
-rw-r--r-- 1 root    root   230606579 Jun 16  2017 pig-0.17.0.tar.gz
-rw-r--r-- 1 root    root   230606579 Jun 16  2017 pig-0.17.0.tar.gz.1
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 sbin
drwxr-xr-x 1 root    root        4096 Mar  2 14:03 share
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 src
```

Step 12: Define or change permissions or modes on files and limit access to only those who are allowed access.

Command 13: sudo chmod 777 pig

```
venom16@comp196:/usr/local$ sudo chmod 777 pig
venom16@comp196:/usr/local$ ls -l
total 1585972
-rw-r--r-- 1 root    root   278813748 Aug 27  2019 apache-hive-3.1.2-bin.tar.gz
-rw-r--r-- 1 root    root   278813748 Aug 27  2019 apache-hive-3.1.2-bin.tar.gz.1
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 bin
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 etc
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 games
drwxr-xr-x 1 venom16 hadoop      4096 Mar  8 13:51 hadoop
-rw-r--r-- 1 root    root   605187279 Jun 15  2021 hadoop-3.3.1.tar.gz
drwxrwxrwx 1 root    root        4096 Mar 21 11:48 hive
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 include
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 lib
lrwxrwxrwx 1 root    root           9 Jan  4 03:10 man -> share/man
drwxrwxrwx 1 root    root        4096 Jun  2  2017 pig
drwxr-xr-x 1 root    root        4096 Jun  2  2017 pig-0.17.0
-rw-r--r-- 1 root    root   230606579 Jun 16  2017 pig-0.17.0.tar.gz
-rw-r--r-- 1 root    root   230606579 Jun 16  2017 pig-0.17.0.tar.gz.1
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 sbin
drwxr-xr-x 1 root    root        4096 Mar  2 14:03 share
drwxr-xr-x 1 root    root        4096 Jan  4 03:10 src
```

Step 13: Change the current working directory to / home/hduser
Command 14: cd /home/hduser1

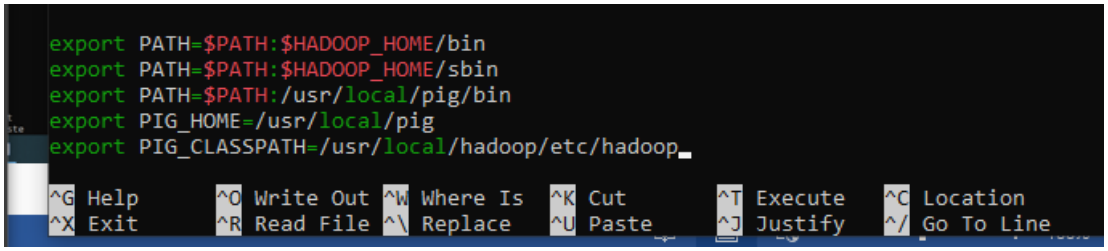
```
venom16@comp196:/usr/local$ cd /home/venom16
venom16@comp196:~$
```

Step 14: The $HIVE_HOME environment variable needs to direct the client shell to the apache-pig directory. Edit the .bashrc shell configuration file using a text editor of your choice (we will be using nano):
Command 15: nano .bashrc

```
venom16@comp196:~$ nano .bashrc
venom16@comp196:~$
```

Step 15: 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



The Hadoop environment variables are located within the same file.

Step 16: Save "Ctrl S" and exit the file by pressing "Ctrl X" followed by "Y" and "Enter" keys.
Step 17: Save and exit the .bashrc file once you add the Pig variables. Apply the changes to the current environment with the following command:
Command 16: source .bashrc

```
venom16@comp196:~$ source .bashrc
venom16@comp196:~$
```

Step 18: We have successfully installed Apache Pig in Ubuntu. Now check the Pig version to verify the installation. Execute the below command:
Command 17: pig -version

```
venom16@comp196:~$ pig -version
Apache Pig version 0.17.0 (r1797386)
compiled Jun 02 2017, 15:41:58
venom16@comp196:~$
```

Step 19: Start Apache Pig.

When we start Apache Pig, it opens a grunt shell.

We can start Pig in one of the following two modes mentioned below:

• Local Mode

• Cluster Mode

To start using pig in local mode '-x local' option is used whereas while executing only "pig" command without any options, Pig starts in the cluster mode.

While running pig in local mode, it can only access files present on the local file system.

Whereas, on starting pig in cluster mode pig can access files present in HDFS.

Step 20: To start Pig in Local Mode, execute the below command:

Command 18: pig -x local

And if you get the below output that means Pig started successfully in Local mode.

```
venom16@comp196:~$ pig -x local
2023-03-27 14:09:37,597 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2023-03-27 14:09:37,598 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType
2023-03-27 14:09:37,652 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.
0 (r1797386) compiled Jun 02 2017, 15:41:58
2023-03-27 14:09:37,652 [main] INFO  org.apache.pig.Main - Logging error messages t
o: /home/venom16/pig_1679906377650.log
2023-03-27 14:09:37,672 [main] INFO  org.apache.pig.impl.util.Utils - Default bootu
p file /home/venom16/.pigbootup not found
2023-03-27 14:09:37,819 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
```

Step 21: Execute the quit command in the grunt shell to come out of it.

Command 19: Quit

```
grunt> quit
2023-03-27 14:16:32,420 [main] INFO  org.apache.pig.Main - Pig script completed in
6 minutes, 54 seconds and 993 milliseconds (414993 ms)
```

Command 20: Pig

And if you get the below output that means Pig started successfully in Cluster mode.

```
venom16@comp196:~$ pig
2023-03-28 10:58:56,464 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2023-03-28 10:58:56,466 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
2023-03-28 10:58:56,467 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2023-03-28 10:58:56,518 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.0 (r1797386) compi
led Jun 02 2017, 15:41:58
2023-03-28 10:58:56,519 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/venom16/p
ig_1679981336512.log
2023-03-28 10:58:56,541 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/venom
16/.pigbootup not found
2023-03-28 10:58:56,852 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tr
acker is deprecated. Instead, use mapreduce.jobtracker.address
2023-03-28 10:58:56,852 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine -
 Connecting to hadoop file system at: hdfs://localhost:54310
2023-03-28 10:58:58,400 [main] INFO  org.apache.pig.PigServer - Pig Script ID for the session: PIG-de
fault-c78a6620-5f02-42ed-b663-4d17616825d3
2023-03-28 10:58:58,401 [main] WARN  org.apache.pig.PigServer - ATS is disabled since yarn.timeline-s
ervice.enabled set to false
grunt>
```

Step 21: Running in local mode: Start pig using following command:

Command 21: pig -x local

```
venom16@comp196:~$ pig -x local
2023-03-27 14:09:37,597 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2023-03-27 14:09:37,598 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType
2023-03-27 14:09:37,652 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.
0 (r1797386) compiled Jun 02 2017, 15:41:58
2023-03-27 14:09:37,652 [main] INFO  org.apache.pig.Main - Logging error messages t
o: /home/venom16/pig_1679906377650.log
2023-03-27 14:09:37,672 [main] INFO  org.apache.pig.impl.util.Utils - Default bootu
p file /home/venom16/.pigbootup not found
2023-03-27 14:09:37,819 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
```

Step 22: Load the data in the file named sample.txt as a relation named Employees.

Command 22: Employees = LOAD 'sample.txt' USING PigStorage(',') as (id:int, firstname:chararray,salary:int, Dept:chararray);

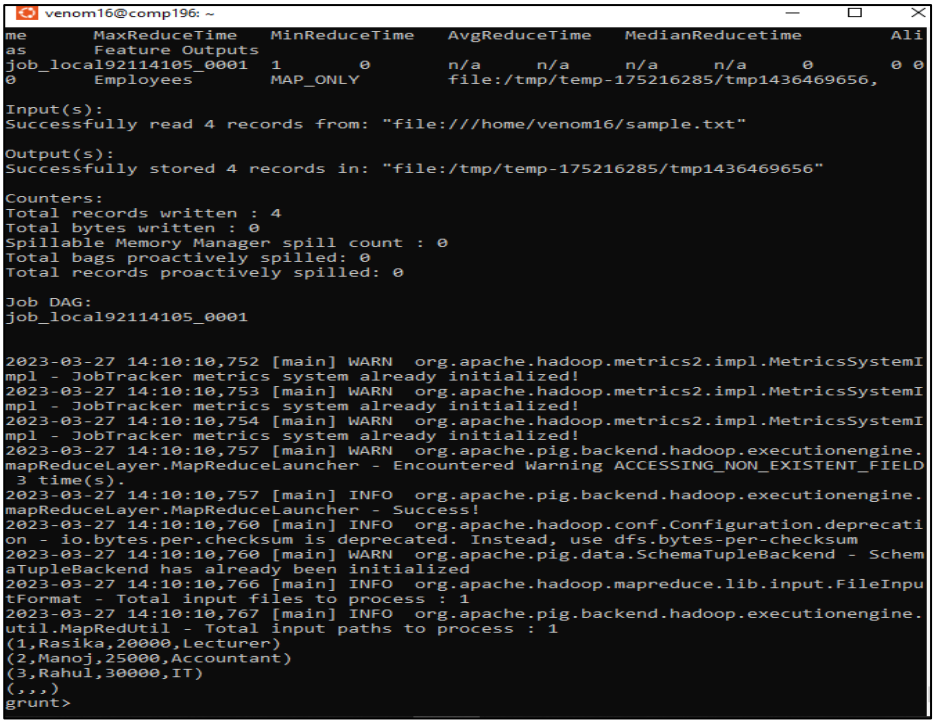
```
grunt> Employees = LOAD 'sample.txt' USING  PigStorage(',') as (id:int, firstname:chararray, salary:int, Dept:chararray);
2023-03-28 11:08:45,221 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated.
grunt>
```

Step 23: Display the Employees table on the screen.

The Dump operator is used to run the Pig Latin statements and display the results on the screen.

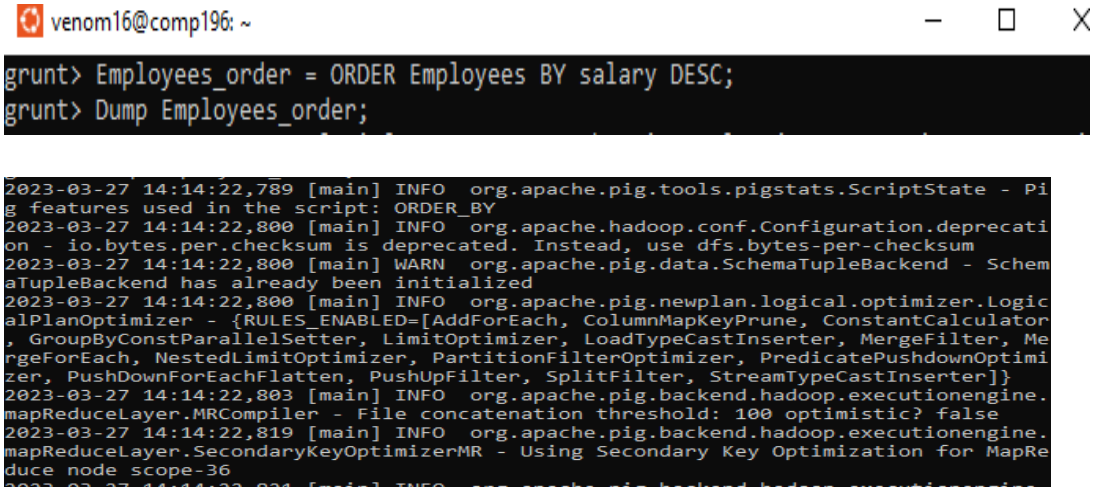
Command 23: Dump Employees;

```
grunt> Dump Employees;
2023-03-27 14:10:09,382 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-03-27 14:10:09,398 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pi
g features used in the script: UNKNOWN
2023-03-27 14:10:09,415 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-03-27 14:10:09,461 [main] INFO  org.apache.pig.newplan.logical.optimizer.Logic
alPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, ConstantCalculator
, GroupByConstParallelSetter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, Me
rgeForEach, NestedLimitOptimizer, PartitionFilterOptimizer, PredicatePushdownOptimi
zer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter]}
2023-03-27 14:10:09,530 [main] INFO  org.apache.pig.impl.util.SpillableMemoryManage
r - Selected heap (PS Old Gen) of size 699400192 to monitor. collectionUsageThresho
ld = 489580128, usageThreshold = 489580128
2023-03-27 14:10:09,599 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? false
2023-03-27 14:10:09,632 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2023-03-27 14:10:09,632 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
2023-03-27 14:10:09,651 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-03-27 14:10:09,739 [main] INFO  org.apache.hadoop.metrics2.impl.MetricsConfig
- Loaded properties from hadoop-metrics2.properties
2023-03-27 14:10:09,821 [main] INFO  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - Scheduled Metric snapshot period at 10 second(s).
2023-03-27 14:10:09,821 [main] INFO  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system started
2023-03-27 14:10:09,845 [main] INFO  org.apache.pig.tools.pigstats.mapreduce.MRScri
ptState - Pig script settings are added to the job
2023-03-27 14:10:09,851 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - mapred.job.reduce.markreset.buffer.percent is deprecated. Instead, use mapredu
ce.reduce.markreset.buffer.percent
2023-03-27 14:10:09,851 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
mapReduceLayer.JobControlCompiler - mapred.job.reduce.markreset.buffer.percent is n
ot set, set to default 0.3
2023-03-27 14:10:09,854 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - mapred.output.compress is deprecated. Instead, use mapreduce.output.fileoutput
format.compress
```



Step 24: Arrange the tuples of the relation in descending order, based on salary and store it as Employees_order.

Command 24: Employees_order = ORDER Employees BY salary DESC;



```
venom16@comp196: ~
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_local462565466_0002 ->      job_local1449750795_0003,
job_local1449750795_0003        ->      job_local1333784696_0004,
job_local1333784696_0004


2023-03-27 14:14:24,347 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,348 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,348 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,351 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,352 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,353 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,355 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,356 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,356 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:14:24,358 [main] WARN  org.apache.pig.backend.hadoop.executionengine.
mapReduceLayer.MapReduceLauncher - Encountered Warning ACCESSING_NON_EXISTENT_FIELD
 3 time(s).
2023-03-27 14:14:24,358 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
mapReduceLayer.MapReduceLauncher - Success!
2023-03-27 14:14:24,358 [main] INFO  org.apache.hadoop.conf.Configuration.deprecati
on - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-03-27 14:14:24,358 [main] WARN  org.apache.pig.data.SchemaTupleBackend - Schem
aTupleBackend has already been initialized
2023-03-27 14:14:24,360 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInpu
tFormat - Total input files to process : 1
2023-03-27 14:14:24,360 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
util.MapRedUtil - Total input paths to process : 1
(3,Rahul,30000,IT)
(2,Manoj,25000,Accountant)
(1,Rasika,20000,Lecturer)
(,,,)
grunt>
```

Command 25: quit

```
(,,,)
grunt> quit
2023-03-27 14:16:32,420 [main] INFO  org.apache.pig.Main - Pig script completed in
6 minutes, 54 seconds and 993 milliseconds (414993 ms)
```

Running in HDFS mode (pig can access data on HDFS).

Step 25: Move sample.txt to HDFS.

Command 26: hdfs dfs -put sample.txt /

```
venom16@comp196:~$ hdfs dfs -put sample.txt /
put: `/sample.txt': File exists
```

Step 26: Check sample.txt got saved to HDFS or not by listing files of HDFS.

Command 27: hdfs dfs -ls /

```
venom16@comp196:~$ hdfs dfs -ls /
Found 5 items
-rw-r--r--   1 venom16 supergroup         88 2023-03-08 14:13 /Mapreduce.txt
drwxr-xr-x   - venom16 supergroup          0 2023-03-08 14:14 /output
-rw-r--r--   1 venom16 supergroup         67 2023-03-27 13:56 /sample.txt
drwxrwxr-x   - venom16 supergroup          0 2023-03-27 14:04 /tmp
drwxr-xr-x   - venom16 supergroup          0 2023-03-21 11:25 /user
venom16@comp196:~$
```

Step 27: Start pig using pig command.

Command 28: pig

Step 28: Load the data in the file named sample.txt as a relation named Employee.
Command 29: Emp = LOAD 'hdfs://localhost:54310/sample.txt' USING PigStorage(',');

(NOTE: Give a space after and before assignment operator "=")

```
venom16@comp196: ~

Details at logfile: /home/venom16/pig_1679906849601.log
grunt> Emp = LOAD 'hdfs://localhost:54310/sample.txt' USING PigStorage(',');
```

Step 29: Display the Employee table on the screen.
Command 30: dump Emp;

```
venom16@comp196: ~

Details at logfile: /home/venom16/pig_1679906849601.log
grunt> Emp = LOAD 'hdfs://localhost:54310/sample.txt' USING PigStorage(',');
grunt> dump Emp;
2023-03-27 14:21:05,352 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pi
g features used in the script: UNKNOWN
2023-03-27 14:21:05,377 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [
pig.schematuple] was not set... will not generate code.
2023-03-27 14:21:05,421 [main] INFO  org.apache.pig.newplan.logical.optimizer.Logic
alPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, ConstantCalculator
, GroupByConstParallelSetter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, Me
rgeForEach, NestedLimitOptimizer, PartitionFilterOptimizer, PredicatePushdownOptimi
zer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter]}
2023-03-27 14:21:05,481 [main] INFO  org.apache.pig.impl.util.SpillableMemoryManage
r - Selected heap (PS Old Gen) of size 699400192 to monitor. collectionUsageThresho
ld = 489580128, usageThreshold = 489580128
```

```
venom16@comp196: ~
Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTime      AvgMapTime      MedianMapTi
me      MaxReduceTime   MinReduceTime   AvgReduceTime   MedianReducetime        Ali
as      Feature Outputs
job_local637023462_0001 1       0       n/a     n/a     n/a     n/a     0       0 0
0       Emp     MAP_ONLY        hdfs://localhost:54310/tmp/temp-760666114/tmp127369
1167,

Input(s):
Successfully read 4 records (5755849 bytes) from: "hdfs://localhost:54310/sample.tx
t"

Output(s):
Successfully stored 4 records (5755868 bytes) in: "hdfs://localhost:54310/tmp/temp-
760666114/tmp1273691167"

Counters:
Total records written : 4
Total bytes written : 5755868
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_local637023462_0001


2023-03-27 14:21:15,612 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:21:15,614 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:21:15,615 [main] WARN  org.apache.hadoop.metrics2.impl.MetricsSystemI
mpl - JobTracker metrics system already initialized!
2023-03-27 14:21:15,621 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
mapReduceLayer.MapReduceLauncher - Success!
2023-03-27 14:21:15,624 [main] WARN  org.apache.pig.data.SchemaTupleBackend - Schem
aTupleBackend has already been initialized
2023-03-27 14:21:15,632 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInpu
tFormat - Total input files to process : 1
2023-03-27 14:21:15,633 [main] INFO  org.apache.pig.backend.hadoop.executionengine.
util.MapRedUtil - Total input paths to process : 1
(1 Rasika 20000 Lecturer)
(2 Manoj 25000 Accountant)
(3 Rahul 30000 IT)
()
grunt>
```