# AUTOMATED DEPLOYMENT

Bhavin G Chennur || Nishanth S Shastry || Monish S

December 10, 2020

## 1 Introduction

This project presents an innovative way of integrating machine intelligence to an automated deployment procedure. The master node keeps getting better in understanding the computers in its cluster as it has an inbuilt Machine Learning model which uses k - Nearest Neighbours Algorithm. All the computers in the cluster could be of varied processing powers, the model learns how many tasks each computer could perform in a given amount of time. Even if any of the processors get damaged and stops working the model can learn that and take appropriate actions, like indicating a failure of a particular node and assigning the jobs to computers based on their previous history. The following sections will explain how big data technologies have been used to apply the above concept.

## 2 Project Code

### 2.1 Code Explanation

**FrontEnd-Database interaction** (index.php, ui.php, connection.php, display.php) :

> Using front-end web technologies such as **HTML, CSS and JS** a User-friendly website is developed. This website will accept the inputs from the user **i.e.,(User name, User id, No. of tasks)**

> The input taken from the first step is stored in the database with the help of **PHP**, a popular general-purpose scripting language and using **SQL** (a the query language).

> **Phpmyadmin** is used to handle the administration of **MySQL** (a database management system) over the Web.

> Now, in the terminal suitable **MYSQL** Queries are executed directly from the command line and by doing so the data in the user input data will further be transferred to a **CSV** file which is the main **dataset**(input provider) for the spark program**(bdtest1.py)**

**bdtest1.py**

>This code is a spark program,this program takes the input from the dataset which con-

tains four columns **INPUT:(Sl.no, User name, User id, No. of tasks)**.

>After considering the input,the code above calculates the average number of tasks for each user and returns the tuple which contains **OUTPUT:(User id,Average no. of tasks).**

**bdknn1.py**

>This file is the main component of our project.We have made use of the **KNN(K-Nearest Neighbour)**algorithm. KNN algorithm is a ML algorithm which is based on supervised learning.

>**Supervised learning** is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.So these input-output pairs have been fed into the model by us using the **training dataset**.

>This file receives input from the spark program above in the form of tuple.The tuple consists of **(User id, Avg No. of tasks)**.With the help of the training dataset which has been fed into the model the KNN algorithm is applied,the nearest value to the average number of tasks is determined and that node is predicted for so and so user.So, the output will be in the form of user id and the node predicted for every user.**(NOTE: Here,Node refers to Computer)**

>This file also does perform mapping operation.By mapping every Node which has been assigned gets mapped with 1 and the output will be of the form Eg: Node1,1. This output will later be fed as an input to the reducer.

**reducer1.py**

> It will count the number of users assigned to each computer in the cluster, the node is not printed if it is not assigned at all.

> It also acts as a verification step to show that the project really works and the workload is split. It is not split in a balanced way but is split in a logical way.

**Hive**

> It is used to backup the datasets used for input and training in the project.

> Hive is compliant with HDFS, if our project gets to be applied in a real world application then we can exploit the power of parallel processing using distributed computing by storing all the datasets in Hive and retrieving them as needed.

**PageRank**

> This project establishes a system for processing, we wanted to virtually run something

on this system.

> As a toy example, we assume a user gives a task of finding pageranks. As this job is executed we demonstrate how the system our project has created works, for instance, which node was assigned to the user and which node performs this task.

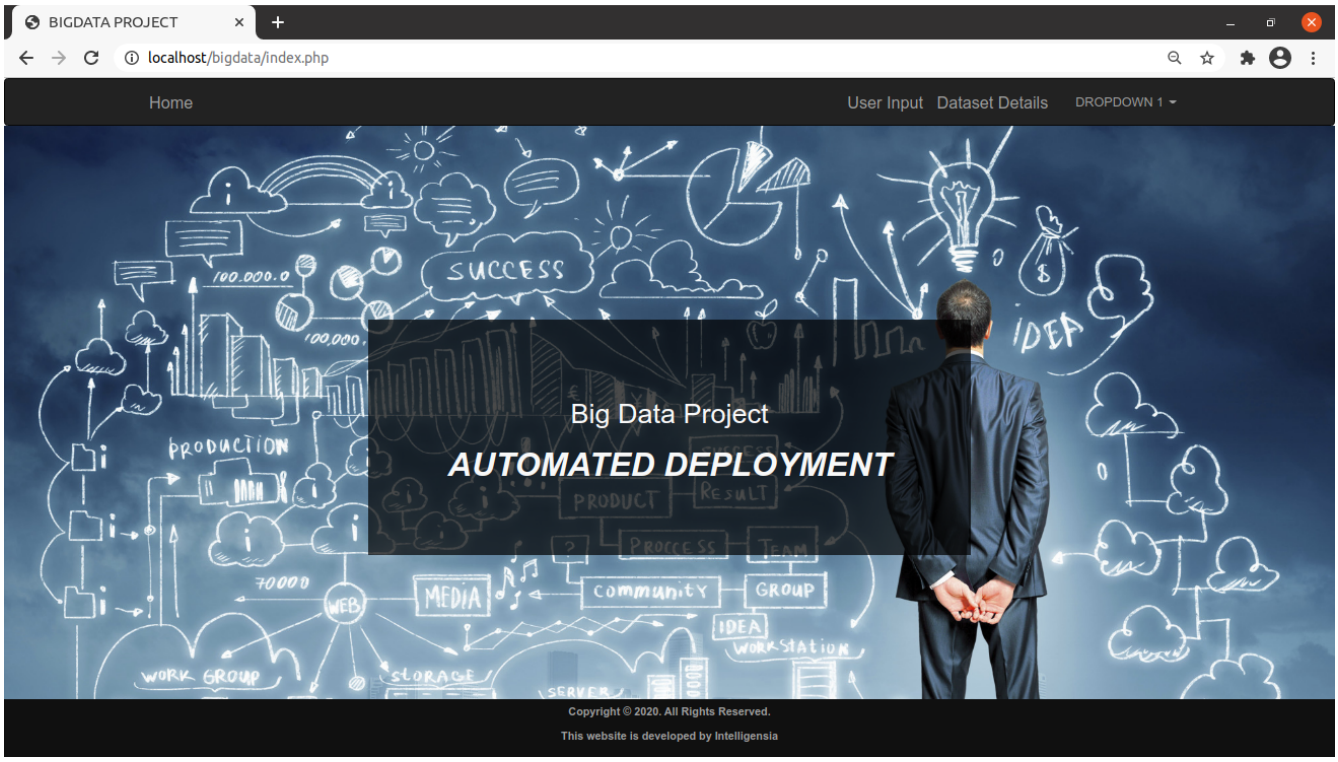
### Bash Script

> When there are so many software platforms being used many commands have to be executed and from different directories.

> The bash script can execute many commands at once, so one only has to execute a bash script to do it all.

## 2.2 CODES

*HOME PAGE*



This is the main page of the website. Generally called the **Landing Page**.

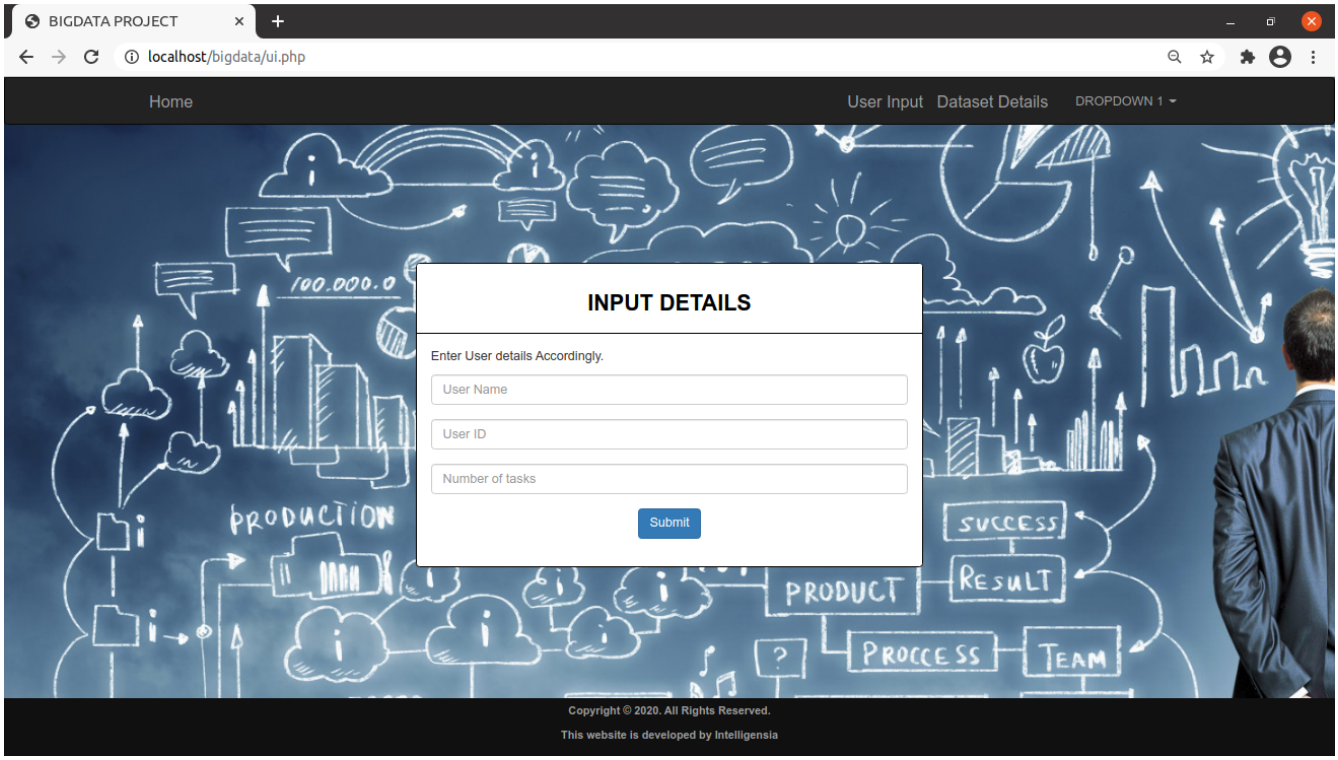
**FILE NUMBER 1) index.php**

```
<!DOCTYPE html>
<head>
    <!--<link rel="shortcut icon" href="img/projecticon.jpg" />  tilte bar
    icon -->
    <title>BIGDATA PROJECT</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- responsiveness -->
    <!-- latest compiled and minified CSS -->
    <link rel="stylesheet" href="bootstrap/css/bootstrap.min.css"
    type="text/css">
    <!-- jquery library -->
    <script type="text/javascript"
    src="bootstrap/js/jquery-3.2.1.min.js"></script>
    <!-- Latest compiled and minified javascript -->
    <script type="text/javascript"
    src="bootstrap/js/bootstrap.min.js"></script>
    <!-- External CSS own css from scratch -->
    <link rel="stylesheet" href="css/style.css" type="text/css">
</head>
<body>
    <div>
        <nav class="navbar navbar-inverse navabar-fixed-top">
            <div class="container">
                <div class="navbar-header">
                    <button type="button" class="navbar-toggle"
                    data-toggle="collapse" data-target="#myNavbar">
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                    </button>

                    <a href="index.php" class="navbar-brand">Home</a>
                </div>
                <div class="collapse navbar-collapse" id="myNavbar">
                    <ul class="nav navbar-nav navbar-right">
                        <a href="ui.php" class="navbar-brand">User Input</a>
                        <a href="display.php" class="navbar-brand">Dataset
                        Details</a>
                        <li class="dropdown">
                        <a class="dropdown-toggle" data-toggle="dropdown"
                        href="#">DROPDOWN 1
                        <span class="caret"></span></a>
                        <ul class="dropdown-menu">
                        <li><a href="#">TEST 1</a></li>
```

```
                        <li><a href="#">TEST 2</a></li>
                        </ul>
                        </li>
                    </ul>

                </div>
            </div>
        </nav>
        <div id="bannerImage" style="background-image: url('pic.jpg');">
            <div class="container">
                <center>
                    <div id="bannerContent">
                        <h2>Big Data Project</h2><h1><b><i>AUTOMATED
                        DEPLOYMENT</i></b></h1>
                    </div>
                </center>
            </div>
        </div>
    </div>

<footer class="footer">
        <div class="container">
        <center>
            <p>Copyright &copy 2020. All Rights Reserved.</p>
            <p>This website is developed by Intelligensia</p>
        </center>
        </div>
</footer>
</body>
</html>
```

```
\pagebreak

\textbf{ FILE NUMBER 1) index.php}
```

*User INPUT PAGE*




This is the web-page where the user is asked to enter the details regarding, **Username**, **UserID** and the **Number of tasks** to be performed.

**FILE NUMBER 2) ui.php**

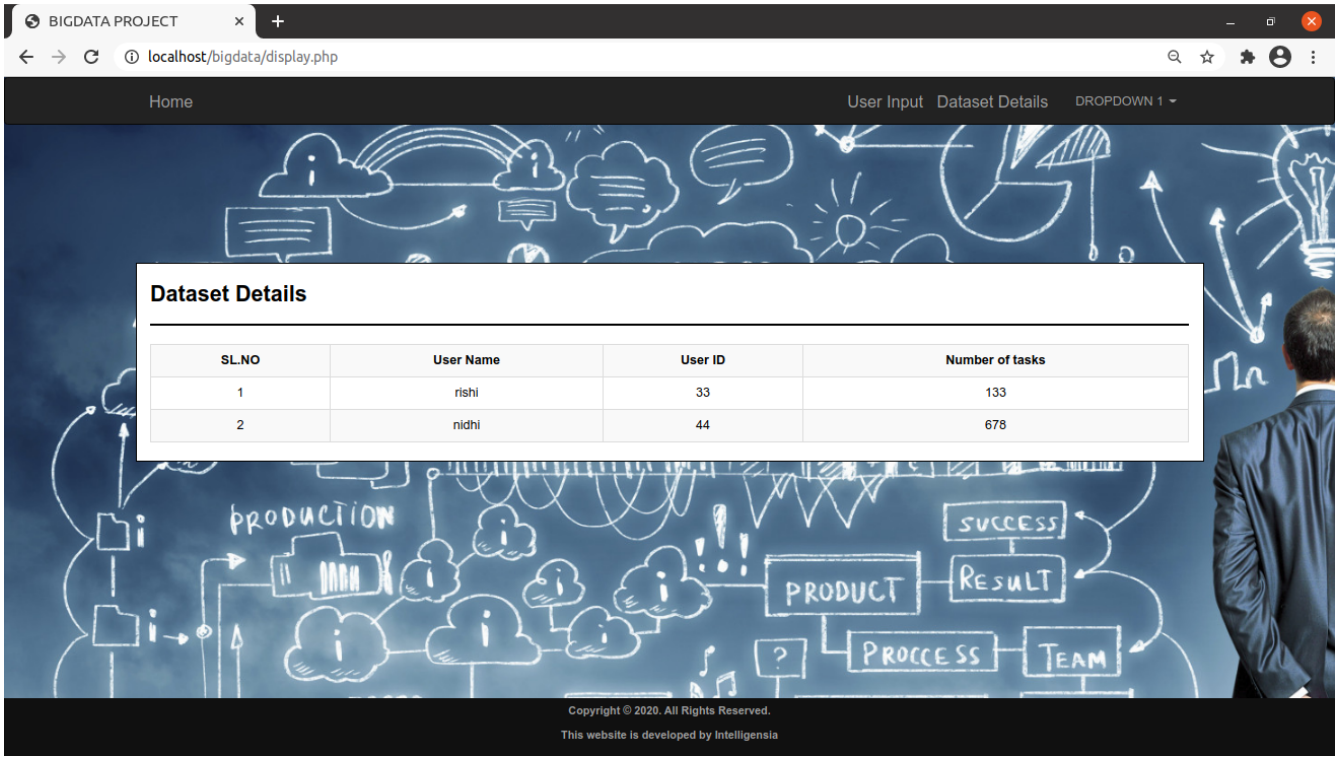
```php
<?php
$servername = "localhost";
$username = ""; // username for the MYSQL server account
$password = ""; // password for the MYSQL server account
$dbname = "";  // database name for the table with input details.

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$user_name = $_POST['uname'];
$userid = $_POST['userid'];
$numoftasks = $_POST['numoftasks'];

$sql = "INSERT INTO data (username, userid, numoftasks)
VALUES ('$user_name', '$userid', '$numoftasks')";
$result = $conn->query($sql);

?>

<!DOCTYPE html>
<head>
    <!--<link rel="shortcut icon" href="img/projecticon.jpg" />  tilte bar
    icon -->
    <title>BIGDATA PROJECT</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- responsiveness -->
    <!-- latest compiled and minified CSS -->
    <link rel="stylesheet" href="bootstrap/css/bootstrap.min.css"
    type="text/css">
    <!-- jquery library -->
    <script type="text/javascript"
    src="bootstrap/js/jquery-3.2.1.min.js"></script>
    <!-- Latest compiled and minified javascript -->
    <script type="text/javascript"
    src="bootstrap/js/bootstrap.min.js"></script>
    <!-- External CSS own css from scratch -->
    <link rel="stylesheet" href="css/style.css" type="text/css">
</head>
<body style="background-image: url('pic.jpg');">
    <div>
        <nav class="navbar navbar-inverse navabar-fixed-top">
            <div class="container">
```

```
            <div class="navbar-header">
                <button type="button" class="navbar-toggle"
                data-toggle="collapse" data-target="#myNavbar">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>

                <a href="index.php" class="navbar-brand">Home</a>
            </div>
            <div class="collapse navbar-collapse" id="myNavbar">
                <ul class="nav navbar-nav navbar-right">
                    <a href="ui.php" class="navbar-brand">User Input</a>
                    <a href="display.php" class="navbar-brand">Dataset
                    Details</a>
                    <li class="dropdown">
                    <a class="dropdown-toggle" data-toggle="dropdown"
                    href="#">DROPDOWN 1
                    <span class="caret"></span></a>
                    <ul class="dropdown-menu">
                    <li><a href="#">TEST 1</a></li>
                    <li><a href="#">TEST 2</a></li>
                    </ul>
                    </li>
                </ul>

            </div>
        </div>
    </nav>
    <br/ ><br/ ><br/ ><br/ >
    <br/ ><br/ ><br/ ><br/ >
    <div>
            <div class="container">
            <div class="row">
                <div class="col-xs-6 col-xs-offset-3">
                    <div class="panel panel-primary" style="border-color:
                    black;">
                        <div class="panel-heading" style="border-color:
                        black;background-color: white;color: black;">
                            <center><h3><b>INPUT DETAILS</b></h3></center>
                        </div>
                        <div class="panel-body">
                            <p>Enter User details Accordingly.</p>
                            <form method="post"> <!--
                            action="data_submit.php" -->
                                <div class="form-group">
                                    <input type="text"
                                    class="form-control" name="uname"
```

```
                                        placeholder="User Name"
                                        required="true">
                                    </div>
                                    <!-- <?php //echo $username ?> -->
                                    <div class="form-group">
                                        <input type="number" c

                                        lass="form-control" name="userid"
                                        placeholder="User ID" required="true">
                                    </div>
                                    <!-- <?php //echo $userid ?>
                                    -->
                                    <div class="form-group">
                                        <input type="number"
                                        class="form-control" name="numoftasks"
                                        placeholder="Number of tasks"
                                        required="true">
                                    </div>
                                    <!-- <?php //echo $numoftasks ?> -->
                                    <div class="form-group">

                                        <center><input type="submit"
                                        value="Submit" class="btn
                                        btn-primary"></center>
                                    </div>
                                </form>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

<footer class="footer">
        <div class="container">
        <center>
            <p>Copyright &copy 2020. All Rights Reserved.</p>
            <p>This website is developed by Intelligensia</p>
        </center>
        </div>
</footer>
</body>
</html>
```

In the above code PHP establishes connection with MYSQL SERVER using PHPMYADMIN

*Dataset Details PAGE*




This web page shows the user input details which ultimately form the Dataset for our project.

**FILE NUMBER 3) display.php**

```
<?php
    session_start();
    require 'connection.php';
    $query = "select slno, username, userid, numoftasks
    from data;";
    $query_result=mysqli_query($con,$query) or die(mysqli_error($con));

?>

<!DOCTYPE html>
<head>
    <!--<link rel="shortcut icon" href="img/projecticon.jpg" />  tilte bar
    icon -->
    <title>BIGDATA PROJECT</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- responsiveness -->
    <!-- latest compiled and minified CSS -->
    <link rel="stylesheet" href="bootstrap/css/bootstrap.min.css"
    type="text/css">
    <!-- jquery library -->
    <script type="text/javascript"
    src="bootstrap/js/jquery-3.2.1.min.js"></script>
    <!-- Latest compiled and minified javascript -->
    <script type="text/javascript" src="bootstrap/js/bootstrap.min.js"></script>
    <!-- External CSS own css from scratch -->
    <link rel="stylesheet" href="css/style.css" type="text/css">
</head>
<body style="background-image: url('pic.jpg');">
    <div>
        <nav class="navbar navbar-inverse navabar-fixed-top">
            <div class="container">
                <div class="navbar-header">
                    <button type="button" class="navbar-toggle"
                    data-toggle="collapse" data-target="#myNavbar">
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                    </button>

                    <a href="index.php" class="navbar-brand">Home</a>
                </div>
                <div class="collapse navbar-collapse" id="myNavbar">
                    <ul class="nav navbar-nav navbar-right">
                        <a href="ui.php" class="navbar-brand">User Input</a>
                        <a href="display.php" class="navbar-brand">Dataset
                        Details</a>
```

```
                    <li class="dropdown">
                    <a class="dropdown-toggle" data-toggle="dropdown"
                    href="#">DROPDOWN 1
                    <span class="caret"></span></a>
                    <ul class="dropdown-menu">
                    <li><a href="#">TEST 1</a></li>
                    <li><a href="#">TEST 2</a></li>
                    </ul>
                    </li>
                </ul>

            </div>
        </div>
    </nav>
    <br/ ><br/ ><br/ ><br/ >
    <br/ ><br/ ><br/ ><br/ >
    <div>
    <div class="container" style="border-color: black;background-color:
    white;color: black;border: 1px solid black;">

<h3><b>Dataset Details</b></h3>

<hr style="border: 1px solid black;">
<table class="table table-bordered table-striped">
    <tbody>
        <tr>
            <th style="text-align:center">SL.NO</th>
            <th style="text-align:center;">User Name</th>
            <th style="text-align:center;">User ID</th>
            <th style="text-align:center;">Number of tasks</th>
        </tr>
        <?php
            $query_result=mysqli_query($con,$query)
            or die(mysqli_error($con));
            while($row=mysqli_fetch_array($query_result)){
        ?>
        <tr>
            <td style="text-align:center"><?php echo $row['slno']?></td>
            <td style="text-align:center;">
            <?php echo $row['username']?></td>
            <td style="text-align:center;">
            <?php echo $row['userid']?></td>
            <td style="text-align:center;">
            <?php echo $row['numoftasks']?></td>
        </tr>
            <?php }?>
    </tbody>
```

```
    </table>
    </div>
    </div>

<footer class="footer">
        <div class="container">
        <center>
            <p>Copyright &copy 2020. All Rights Reserved.</p>
            <p>This website is developed by Intelligensia</p>
        </center>
        </div>
</footer>
</body>
</html>
```

**FILE NUMBER 4) connection.php**

This PHP code allows the website to connect to the database present in PHPMYADMIN which in turn is present in the MYSQL server.

```
<?php
$con=mysqli_connect("localhost","","","") or die(mysqli_error($con));
//$con=mysqli_connect("localhost","user","password","database")
or die(mysqli_error($con));
?>
```

**FILE NUMBER 5) scriptsample.sh**

```
#i/bin/sh

export MYSQL_PWD=mysql_password
# command used to store the MYSQL password for further mysql queries

full_path=$(realpath $0)

dir_path=$(dirname $full_path)

cat output.csv | tr "\\t" "," > output.csv

cat testoutput.csv | tr "\\t" "," > testoutput.csv

cat dataset.csv | tr "\\t" "," > dataset.csv

# above 3 commands are used to delete old data and create empty csv's

mysql -u root -h localhost testdb -e "SELECT * FROM testdb.data" >> output.csv
# command to retrieve the user input data and store in csv file.

tail -n +2 output.csv >> testoutput.csv
# command to store only the data and not the headers

awk 'NR{$1=$1}1' OFS="," testoutput.csv >> dataset.csv
# command to convert any tab spaces to comma separated

cd /path-to-file-spark-home/

cat bd_dataset_1.csv | tr "\\t" "," > bd_dataset_1.csv
# command to update the old dataset with new dataset

cd

cp dataset.csv /path-to-file-spark-home/bd_dataset_1.csv

cd ..
cd ..

cd /path-to-file-spark-home/

sudo ./bin/spark-submit bd_test_1.py > /path-to-file/bd_output_1.txt

sudo ./bin/spark-submit bd_test_1.py > /path-to-file/final_output.txt

cd
```

```
cd /path-to-file-(where below python files exist)/

python3 bd_knn_1.py | sort | python3 reducer_1.py

bash ./pagerank.sh

cat final_output.txt
cat output  # Page Rank output
cd ~/hive-home/apache-hive-3.1.2-bin

hive
# select * from bigdata;
# select * from knn;
```

**FILE NUMBER 6) bd test 1.py**

```
from pyspark import SparkConf, SparkContext

conf = SparkConf().setMaster("local").setAppName("F")
sc = SparkContext(conf = conf)

def parseLine(line):
        fields = line.split(',')
        a = int(fields[2])
        n = int(fields[3])
        return (a, n)

lines = sc.textFile("/path-to file-in-spark-home/bd_dataset_1.csv")

rdd = lines.map(parseLine)

t = rdd.mapValues(lambda x: (x, 1)).reduceByKey
(lambda x, y: (x[0] + y[0], x[1] + y[1]))

a = t.mapValues(lambda x: x[0] / x[1])

results = a.collect()

for result in results:
        print(result[0],float(result[1]))
        #print(result)
```

**FILE NUMBER 7) bdknn1.py**

```
# Importing libraries

import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsRegressor


import csv



# K Nearest Neighbors Regression

class K_Nearest_Neighbors_Regressor() :

        def __init__( self, K ) :

                self.K = K

        # Function to store training set

        def fit( self, X_train, Y_train ) :

                self.X_train = X_train

                self.Y_train = Y_train

                # no_of_training_examples, no_of_features

                self.m, self.n = X_train.shape

        # Function for prediction

        def predict( self, X_test ) :

                self.X_test = X_test

                # no_of_test_examples, no_of_features

                self.m_test, self.n = X_test.shape

                # initialize Y_predict
```

```
        Y_predict = np.zeros( self.m_test )

        for i in range( self.m_test ) :

                x = self.X_test[i]

                # find the K nearest neighbors
                from current test example

                neighbors = np.zeros( self.K )

                neighbors = self.find_neighbors( x )

                # calculate the mean of K nearest neighbors

                Y_predict[i] = np.mean( neighbors )

        return Y_predict

# Function to find the K nearest neighbors to current test example

def find_neighbors( self, x ) :

        # calculate all the euclidean distances between current test
        # example x and training set X_train

        euclidean_distances = np.zeros( self.m )

        for i in range( self.m ) :

                d = self.euclidean( x, self.X_train[i] )

                euclidean_distances[i] = d

        # sort Y_train according to euclidean_distance_array and
        # store into Y_train_sorted

        inds = euclidean_distances.argsort()

        Y_train_sorted = self.Y_train[inds]

        return Y_train_sorted[:self.K]

# Function to calculate euclidean distance

def euclidean( self, x, x_train ) :
```

```
                return np.sqrt( np.sum( np.square( x - x_train ) ) )

# Driver code

def main() :

        # Importing dataset

        df = pd.read_csv( "bd_knn_1.csv" )

        X = df.iloc[:,:-1].values

        Y = df.iloc[:,1].values


        # Splitting dataset into train and test set

        X_train, X_test, Y_train, Y_test = train_test_split(
        X, Y, test_size = 1/3, random_state = 0 )

        # Model training

        model = K_Nearest_Neighbors_Regressor( K = 3 )

        model.fit( X_train, Y_train )

        model1 = KNeighborsRegressor( n_neighbors = 3 )

        model1.fit( X_train, Y_train )

        # Prediction on test set

        Y_pred = model.predict( X_test )

        Y_pred1 = model1.predict( X_test )

        file1 = open('/path-to-file/bd_output_1.txt', 'r')
        Lines1 = file1.readlines()
        Lines = []
        Lines.append(Lines1[2])
        Lines.append(Lines1[3])

        rows, cols = (len(Lines), 1)

        arr = [[0 for i in range(cols)] for j in range(rows)]


        count = 0
```

```
# Strips the newline character
for line in Lines:
        if line[0] == '(':

# String to store the resultant String
                res = "";

    # Traverse the words and
    # remove the first and last letter

                res = line[1:len(line) - 2]

                s1 = res.split(",")

                s2 = s1[1]

                s3 = s2[1:len(s2)-1]

                arr[count][0] = float(s3)

                arr[count][0] = int(arr[count][0])

                count = count + 1


temp = model.predict(np.array(arr))

output = []

for i in range(len(temp)):
        if(temp[i] - int(temp[i]) < 0.5):
                output.append( int(temp[i]))
        else:
                output.append(int(temp[i])+1)


for i in range(len(output)):
        print(output[i],"\t",1)

        # node number is output and mapper function

file1 = open('final_output.txt', 'a')
for i in range(len(output)):
        s3 = "user " + str(i) + " : Node" + str(output[i]) + "\n"
        file1.writelines(s3)
file1.close()
```

```
if __name__ == "__main__" :

        # Opening a file
        file1 = open('final_output.txt', 'a')

        L = ["\n\nThe above is the output of the spark program \n\n",
         "The 3rd user input is the user_id  \n",
         "And the 4th input is the number of jobs given by the user at
         that instace   \n\n",
         "The spark program outputs the user
         id and the average number of jobs
         given by the user\n\n\n",
         "Now,\n",
         "we'll use a K-Nearest Neighbours algorithm to select nodes
         appropriately\n\n"
         "Every worker node has a capacity, \n\ni.e.,
         it can execute a certain number of jobs in ,say,
         1 hour depending on it's capacity. \n\n"
         "We have user_id's and their average number of jobs\n\n"
         "We have to train the KNN Model, the training data is:\n\n",
         "No. of jobs per hour \t Node no.\n"]

        file1.writelines(L)

        with open('bd_knn_1.csv', mode = 'r') as file:
                csvFile = csv.reader(file)

                for lines in csvFile:
                        s1 = lines[0] + "\t\t\t\t\t\t\t"+lines[1]
                        file1.writelines(s1)
                        file1.writelines("\n")


        L1 = ["\n\nUsing averages from spark output we predict the node to be
        assigned,according to its capacity, using kNN algorithm: \n\n"]

        file1.writelines(L1)

        file1.close()

        main()
```

**FILE NUMBER 8) reducer 1.py**

```
"""reducer_1.py"""

from operator import itemgetter
import sys
import csv

current_word = None
current_count = 0
word = None

file1 = open('final_output.txt', 'a')

L = ["\n\n The no. of tasks deployed to every node using map reduce:\n\n",
"Computer no. \t No. of Users assigned to it\n\n"]

file1.writelines(L)

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    word, count = line.split('\t', 1)

    # convert count (currently a string) to int
    try:
        count = int(count)
    except ValueError:
        # count was not a number, so silently
        # ignore/discard this line
        continue

    # this IF-switch only works because Hadoop sorts map output
    # by key (here: word) before it is passed to the reducer
    if current_word == word:
        current_count += count
    else:
        if current_word:
            # write result to STDOUT


            s1 =  str(current_word) + "\t\t\t" + str(current_count) + "\n"

            file1.writelines(s1)
```

```
        current_count = count
        current_word = word

# do not forget to output the last word if needed!
if current_word == word:


        s1 =  str(current_word) + "\t\t\t" + str(current_count) + "\n"

        file1.writelines(s1)

file1.close()
```