**Name: -** Shivam Borse

**Roll No.: -** BEAD20119

**Subject: -** CL IV (BUSINESS INTELLIGENCE)

**Assignment No. 10**

**Problem Statement:** Perform the data classification algorithm using any Classification algorithm

**Code-**

import numpy as np

import pandas as pd

data=pd.read\_csv("gender\_classification\_v7.csv")

data.head()

data=pd.DataFrame(data)

data.info()

data["gender"].value\_counts()

data

f=data[data.gender=="Female"]

m=data[data.gender=="Male"]

data.gender=[1 if each=="Male" else 0 for each in data.gender]

data["gender"].value\_counts()

# x=feature

# y=class

x=data.drop(["gender"],*axis*=1)

y=data.gender.values

x

y

x=(x-np.min(x))/(np.max(x)-np.min(x))

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

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,*test\_size*=0.2,*random\_state*=42)

"""# Logistic Regression"""

from sklearn.linear\_model import LogisticRegression

lr = LogisticRegression()

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

print("test accuracy: {} ".format(lr.fit(x\_train, y\_train).score(x\_test, y\_test)))

print("train accuracy: {} ".format(lr.fit(x\_train, y\_train).score(x\_train, y\_train)))

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

y\_true=y\_test

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(y\_true,y\_pred)

cm

import seaborn as sns

import matplotlib.pyplot as plt

f,ax = plt.subplots(*figsize*=(5,5))

sns.heatmap(cm,*annot* = True,*linewidths*=0.5,*linecolor*="red",*fmt* = ".0f",*ax*=ax)

plt.xlabel("y\_pred")

plt.ylabel("y\_true")

plt.show()

"""# Naive Bayes"""

from sklearn.naive\_bayes import GaussianNB

nb=GaussianNB()

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

prediction=nb.predict(x\_test)

print("Score: ",nb.score(x\_test,y\_test))

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

y\_true=y\_test

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(y\_true,y\_pred)

f,ax = plt.subplots(*figsize*=(5,5))

sns.heatmap(cm,*annot* = True,*linewidths*=0.5,*linecolor*="red",*fmt* = ".0f",*ax*=ax)

plt.xlabel("y\_pred")

plt.ylabel("y\_true")

plt.show()

**Output-**

****

****

**Name: -** Shivam Borse

**Roll No.: -** BEAD20119

**Subject: -** CL IV (BUSINESS INTELLIGENCE)

**Assignment No. 11**

**Problem Statement:** Perform the data clustering algorithm using any Clustering algorithm

**Code-**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

import seaborn as sns

data = pd.read\_csv('Iris.csv')

data

data.columns

for i,col in enumerate(data.columns):

    print(*f*'Column number {1+i} is {col}')

data.dtypes

data.drop('Id', *axis*=1, *inplace*=True)

data.head()

data.isna().sum()

target\_data = data.iloc[:,4]

target\_data.unique()

clustering\_data = data.iloc[:,[0,1,2,3]]

clustering\_data.head()

fig, ax = plt.subplots(*figsize*=(15,7))

sns.set(*font\_scale*=1.5)

ax = sns.scatterplot(*x*=data['SepalLengthCm'],*y*=data['SepalWidthCm'], *s*=70, *color*='#f73434',

*edgecolor*='#f73434', *linewidth*=0.3)

ax.set\_ylabel('Sepal Width (in cm)')

ax.set\_xlabel('Sepal Length (in cm)')

plt.title('Sepal Length vs Width', *fontsize* = 20)

plt.show()

from sklearn.cluster import KMeans

wcss=[]

for i in range(1,11):

    km = KMeans(i)

km.fit(clustering\_data)

wcss.append(km.inertia\_)

np.array(wcss)

kms = KMeans(*n\_clusters*=3, *init*='k-means++')

kms.fit(clustering\_data)

KMeans(*n\_clusters*=3)

clusters = clustering\_data.copy()

clusters['Cluster\_Prediction'] = kms.fit\_predict(clustering\_data)

clusters.head()

kms.cluster\_centers\_

import plotly.express as px

cluster0 = clusters[clusters['Cluster\_Prediction'] == 0]

cluster1 = clusters[clusters['Cluster\_Prediction'] == 1]

cluster2 = clusters[clusters['Cluster\_Prediction'] == 2]

fig = px.scatter(clusters, *x*='SepalLengthCm', *y*='SepalWidthCm', *color*='Cluster\_Prediction',

*size\_max*=30, *opacity*=0.7, *title*='Clusters', *labels*={'SepalLengthCm': 'Sepal Length (in cm)', 'SepalWidthCm': 'Sepal Width (in cm)'})

fig.add\_scatter(*x*=kms.cluster\_centers\_[:, 0], *y*=kms.cluster\_centers\_[:, 1],

*mode*='markers', *marker*=dict(*size*=20, *color*='yellow', *line*=dict(*color*='black', *width*=1)),

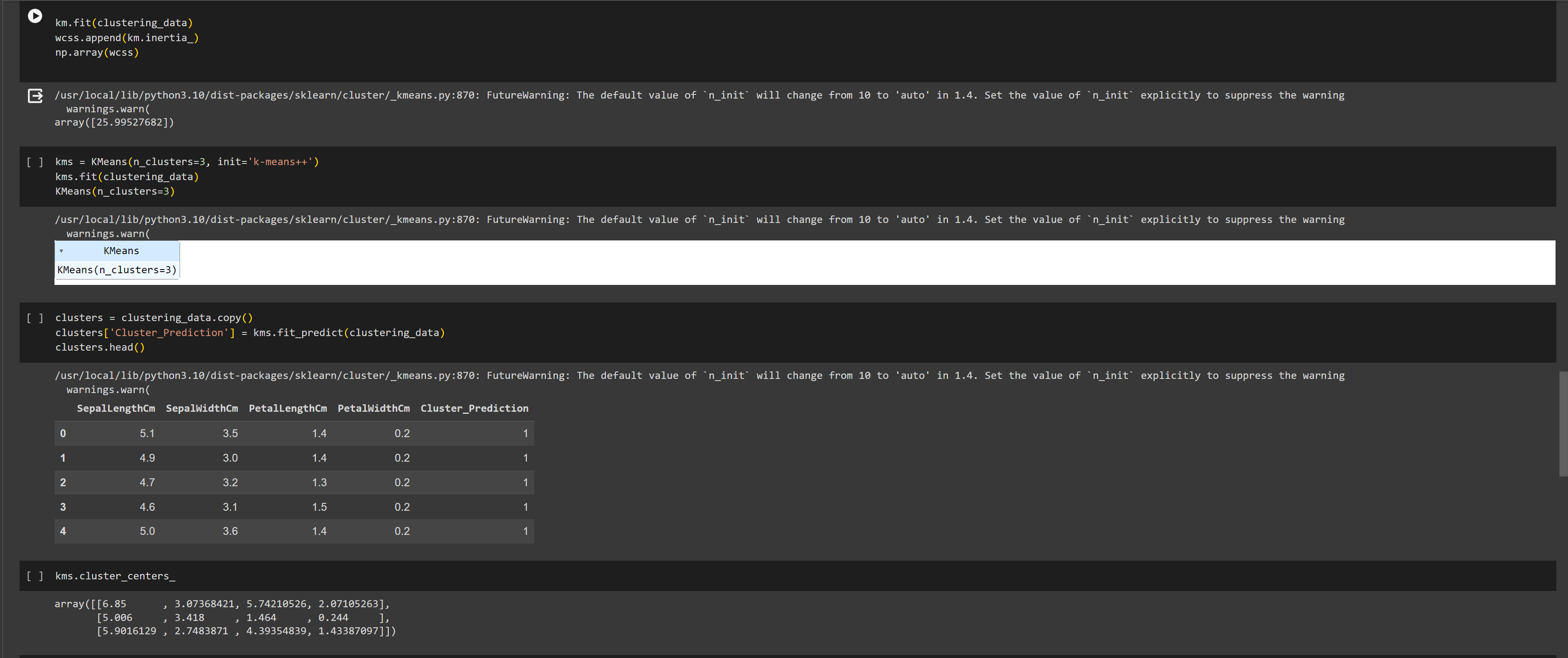
*name*='Centroids')

fig.for\_each\_trace(*lambda* *t*: *t*.update(*name*='Cluster ' + str(*t*.name)))

fig.update\_layout(*legend\_title\_text*='Clusters', *xaxis\_range*=[4, 8], *yaxis\_range*=[1.8, 4.5], *xaxis\_title*='Sepal Length (in cm)', *yaxis\_title*='Sepal Width (in cm)')

fig.show()

**Output-**

****

****

**Name: -** Shivam Borse

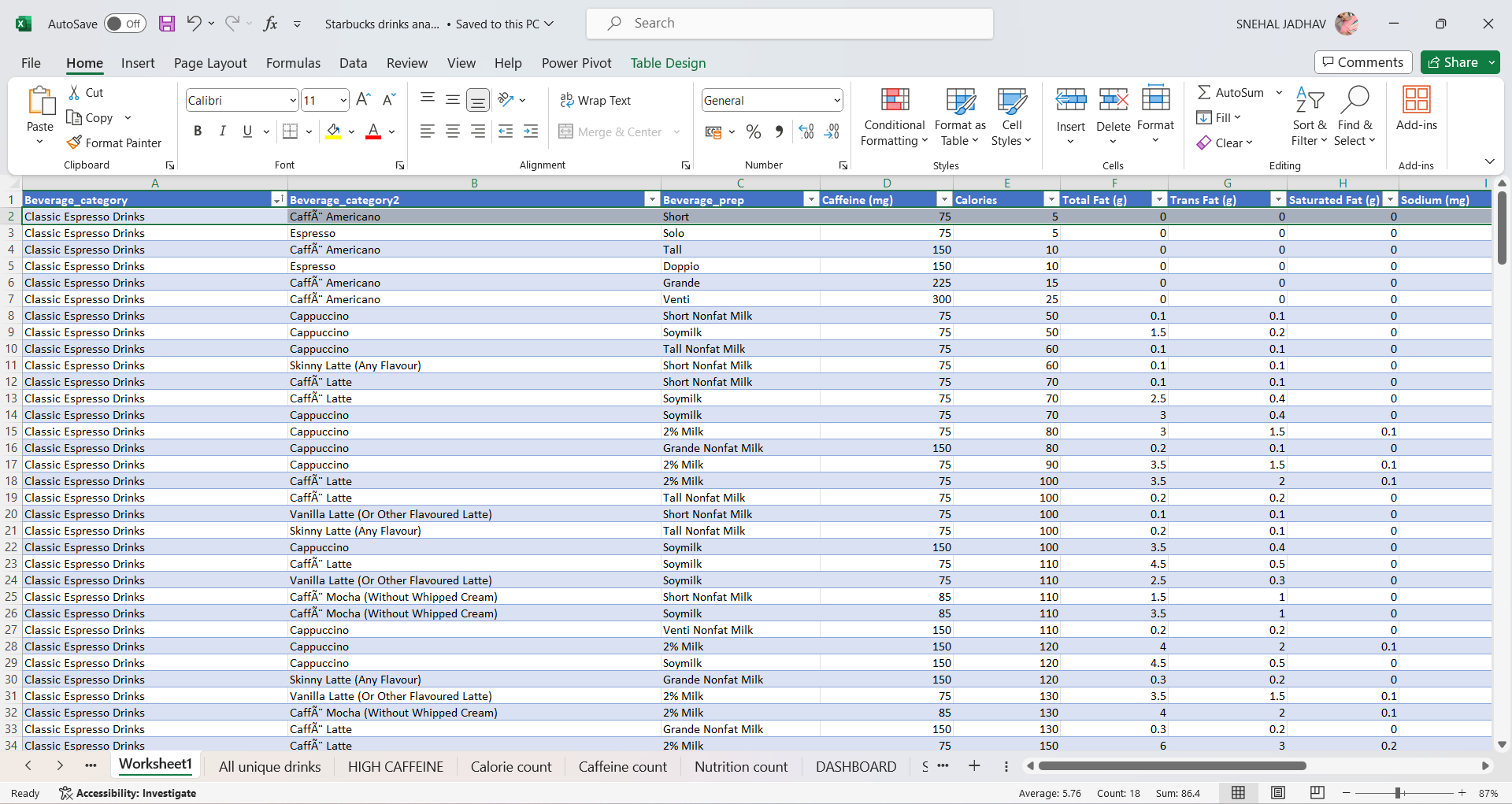
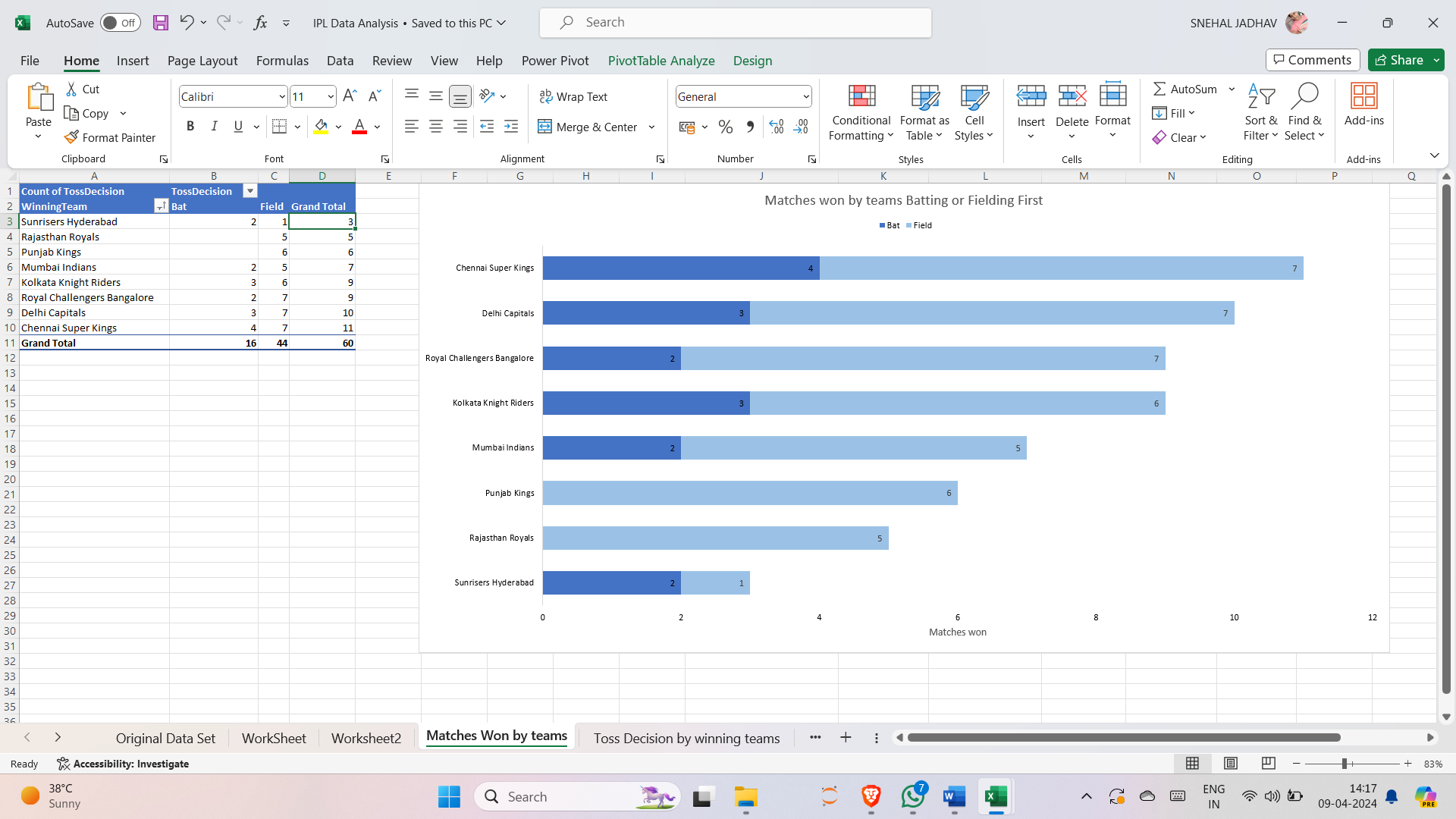
**Roll No.: -** BEAD20119

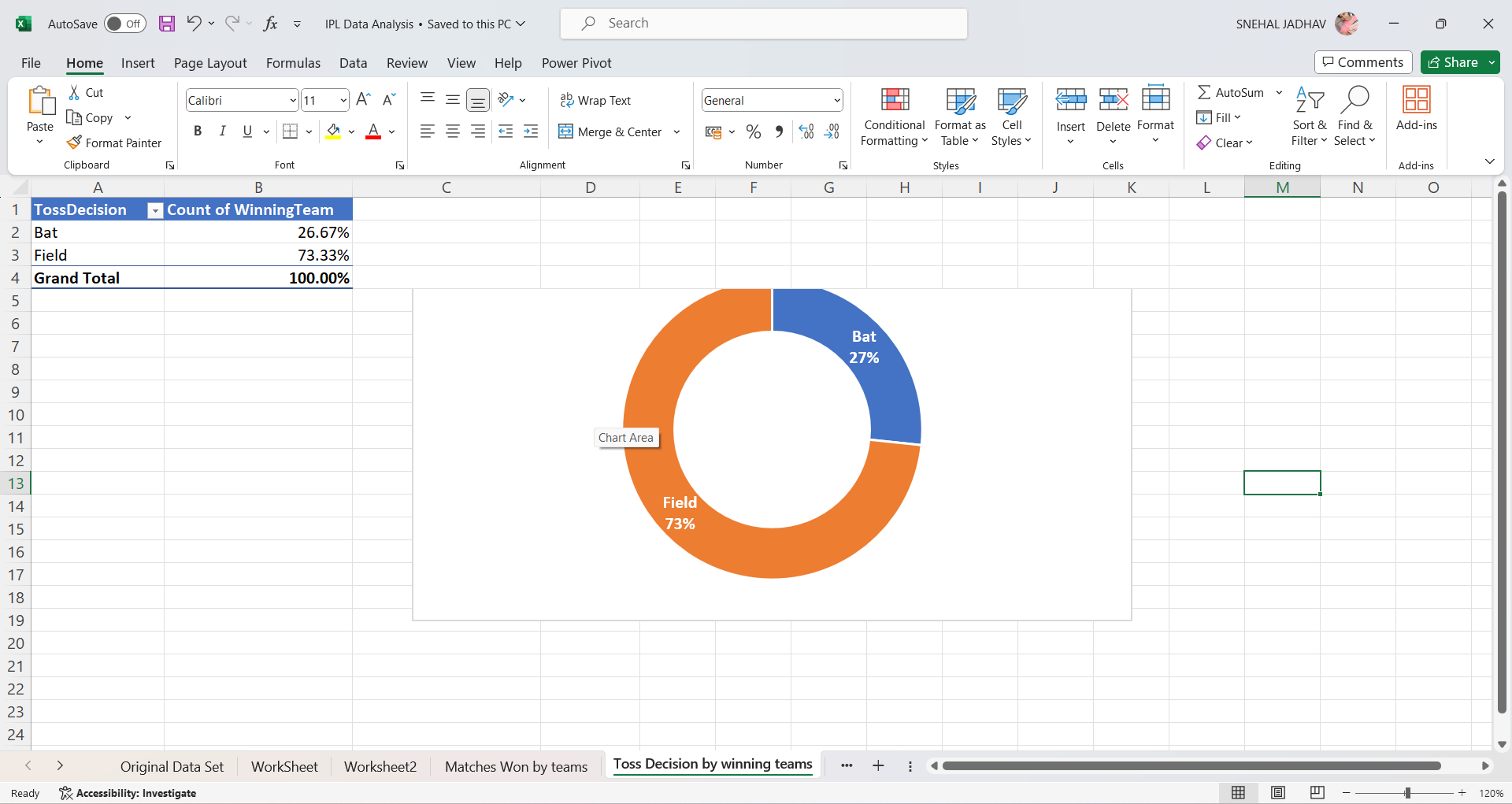
**Subject: -** CL IV (BUSINESS INTELLIGENCE)

**Assignment No. 12**

**Problem Statement:** Data Analysis and Visualization using Advanced Excel.

**Output-  
Dataset:**

****

****

