p229278

April 15, 2024

1 Assignment # 1

2 Muhammad Shafeen

3 22P-9278

4 BS-AI-4A

4.1 IMPORTING LIBRARIES AND IMPORTING DATASET

[ ]:
```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import Normalizer
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import numpy as np
import matplotlib.pyplot as plt
import random
path="/home/shafeenkhan/Documents/My-all-programs--/Semester-4/Aritificial␣
 ↪Intelligence/Lab-Assignments/Assignment 01/total_data_na.csv"
df=pd.read_csv(path)
df
```

[ ]:

	PLAYER	Mat.x	Inns.x	NO	Runs.x	HS	Avg.x	BF	SR.x	\
0	Aaron Finch	10	9	1	134	46	16.75	100	134.00	
1	AB de Villiers	12	11	2	480	90	53.33	275	174.54	
2	Abhishek Sharma	3	3	2	63	46	63	33	190.90	
3	Ajinkya Rahane	15	14	1	370	65	28.46	313	118.21	
4	Alex Hales	6	6	0	148	45	24.66	118	125.42	
..	…	…	…	..	…	..	…	…	…	
138	Siddarth Kaul	0	0	0	0	0	0	0	0.00	
139	Trent Boult	0	0	0	0	0	0	0	0.00	
140	Umesh Yadav	0	0	0	0	0	0	0	0.00	
141	ViOy Kumar	0	0	0	0	0	0	0	0.00	
142	Yuzvendra Chahal	0	0	0	0	0	0	0	0.00	

	X100	…	Ov	Runs.y	Wkts	BBI	Avg.y	Econ	SR.y	X4w	X5w	y

```
0        0  …   0.0       0     0    0      0   0.00      0     0     0  0
1        0  …   0.0       0     0    0      0   0.00      0     0     0  0
2        0  …   0.0       0     0    0      0   0.00      0     0     0  0
3        0  …   0.0       0     0    0      0   0.00      0     0     0  0
4        0  …   0.0       0     0    0      0   0.00      0     0     0  0
..     …  …   …     …    …  …     …    …    …  …   … ..
138      0  …  66.0     547    21    0  26.04   8.28  18.85     0     0  0
139      0  …  52.4     466    18    0  25.88   8.84  17.55     0     0  0
140      0  …  53.1     418    20    0   20.9   7.86  15.95     0     0  0
141      0  …   3.5      65     2    0   32.5  16.95   11.5     0     0  0
142      0  …  50.0     363    12    0  30.25   7.26     25     0     0  0

[143 rows x 25 columns]
```

4.2 CLEANING OF DATA

[ ]:
```
df=df.replace("-",value=None)
# df.info()
df.isnull().sum()
df=df.ffill()
df.isnull().sum()
# df.describe()
```

[ ]:
```
PLAYER     0
Mat.x      0
Inns.x     0
NO         0
Runs.x     0
HS         0
Avg.x      0
BF         0
SR.x       0
X100       0
X50        0
X4s        0
X6s        0
Mat.y      0
Inns.y     0
Ov         0
Runs.y     0
Wkts       0
BBI        0
Avg.y      0
Econ       0
SR.y       0
X4w        0
X5w        0
```

```
y         0
dtype: int64
```

[ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 143 entries, 0 to 142
Data columns (total 25 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   PLAYER  143 non-null    object
 1   Mat.x   143 non-null    int64
 2   Inns.x  143 non-null    int64
 3   NO      143 non-null    int64
 4   Runs.x  143 non-null    int64
 5   HS      143 non-null    int64
 6   Avg.x   143 non-null    object
 7   BF      143 non-null    int64
 8   SR.x    143 non-null    float64
 9   X100    143 non-null    int64
 10  X50     143 non-null    int64
 11  X4s     143 non-null    int64
 12  X6s     143 non-null    int64
 13  Mat.y   143 non-null    int64
 14  Inns.y  143 non-null    int64
 15  Ov      143 non-null    float64
 16  Runs.y  143 non-null    int64
 17  Wkts    143 non-null    int64
 18  BBI     143 non-null    int64
 19  Avg.y   143 non-null    object
 20  Econ    143 non-null    float64
 21  SR.y    143 non-null    object
 22  X4w     143 non-null    int64
 23  X5w     143 non-null    int64
 24  y       143 non-null    int64
dtypes: float64(3), int64(18), object(4)
memory usage: 28.1+ KB
```

4.3 Changing some strings to correct form for processing

[ ]:
```
# def preprocess_column(column):
#     # Remove non-numeric characters using regular expression
#     column = column.str.replace(r'[^0-9.]', '')
#     # Convert data type to float
#     column = pd.to_numeric(column, errors='coerce')  # Set errors='coerce' to␣
 ↪handle non-numeric values
#     return column
```

[ ]:
```
df2=df
encoder=LabelEncoder()
df2["Encoded_Names"] = encoder.fit_transform(df["PLAYER"])
# df2['SR.x'] = preprocess_column(df['SR.x'])
df2=df2.drop(columns="PLAYER")
scalar=StandardScaler()
df2.isnull().count()
df2=df2.ffill()
df3=scalar.fit_transform(df2)
df3
```

[ ]:
```
array([[ 0.4480029 ,  0.54491687, -0.1550601 , …, -0.08391814,
         0.        , -1.69575554],
       [ 0.77823185,  0.90989631,  0.46087308, …, -0.08391814,
         0.        , -1.71998062],
       [-0.7077984 , -0.55002147,  0.46087308, …, -0.08391814,
         0.        , -1.67153046],
       …,
       [-1.20314182, -1.09749065, -0.77099328, …, -0.08391814,
         0.        ,  1.52617999],
       [-1.20314182, -1.09749065, -0.77099328, …, -0.08391814,
         0.        ,  1.55040507],
       [-1.20314182, -1.09749065, -0.77099328, …, -0.08391814,
         0.        ,  1.71998062]])
```

5 Using the elbow method to find the optimal number of clusters

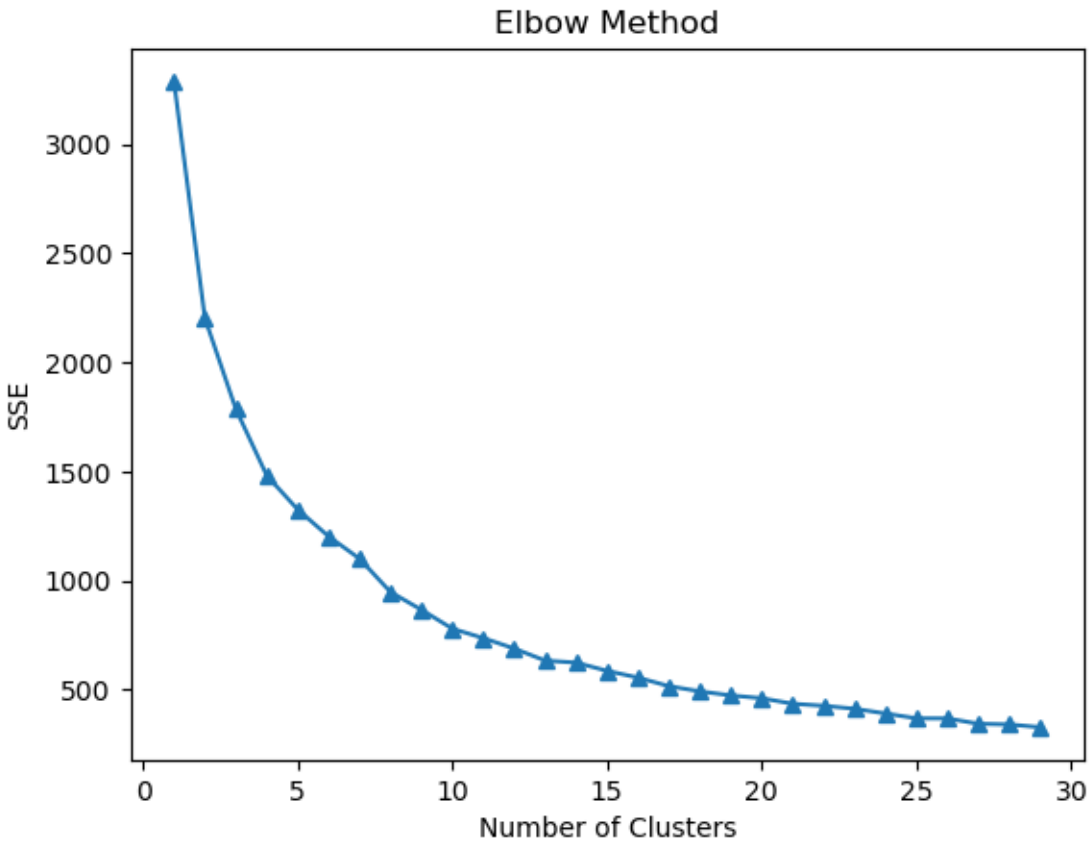
[ ]:
```
sse = []
for k in range(1, 30):
    kmeans = KMeans(n_clusters=k, max_iter=300, random_state=0)
    kmeans.fit(df3)
    sse.append(kmeans.inertia_)
number_clusters = range(1,30)
plt.plot(number_clusters, sse,marker='^')
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('SSE')
plt.show()
```

```
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
```

```
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
```

```
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
```

```
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/shafeenkhan/miniconda3/lib/python3.9/site-
packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



6 For producing same output

[ ]:
```
# random.seed(0)
```

6.1 Training the model

[ ]:
```
# Specifying the number of clusters
kmeans = KMeans(n_clusters=4,n_init=10)

# Fitting the model to the data
kmeans.fit(df3)

# Getting the centroids of the clusters
centroids = kmeans.cluster_centers_
print("Centroids:")
print(centroids)

# Getting the labels assigned to each data point
labels = kmeans.labels_
```

```
print("\nLabels:")
print(labels)
```

```
Centroids:
[[ 0.15617267  0.26481636 -0.09776399  0.04960099  0.40323538  0.34834346
   0.11520601  0.53016329 -0.1600461  -0.10268772  0.03513852 -0.02921066
  -0.633541   -0.79407724 -0.77432818 -0.80690122 -0.70317015  0.
  -0.85330807 -0.75407942 -0.86294227 -0.18172434 -0.08391814  0.
   0.03154894]
 [ 0.52621502  0.14151853  0.49329061 -0.2442881  -0.17003748 -0.11213204
  -0.26478405  0.3194559  -0.1600461  -0.38010296 -0.28538668 -0.14912175
   1.28195842  1.3369747   1.27043671  1.25924887  1.1488059   0.
   0.65782432  0.49718977  0.78785314  0.3311687  -0.08391814  0.
   0.20081316]
 [-1.09431637 -1.00624578 -0.64500695 -0.7356715  -0.96221136 -0.90700732
  -0.76415715 -1.11901828 -0.1600461  -0.489542   -0.67209629 -0.67969994
  -0.14055226 -0.04363911 -0.02434875  0.03040398 -0.00653103  0.
   0.64284067  0.80413012  0.56034363 -0.03407331  0.18881581  0.
  -0.26097017]
 [ 1.19101803  1.52833371  0.76883966  2.19553619  1.74775573  1.62169836
   2.1517139   0.79446991  1.11143125  2.24440733  2.161443    2.04608236
  -0.84932541 -0.81886648 -0.77272992 -0.80513778 -0.7294968   0.
  -0.92167035 -1.21386229 -0.97150119 -0.18172434 -0.08391814  0.
   0.13862129]]

Labels:
[0 3 0 0 0 3 1 1 1 1 1 1 0 0 3 3 2 2 1 0 2 0 2 0 1 0 3 1 3 0 0 1 1 1 0 0 0
 0 1 1 3 0 3 0 0 0 1 1 3 0 0 0 0 0 1 0 1 2 0 2 1 3 0 0 1 0 0 1 0 1 1 1 0 3
 0 0 0 3 0 1 3 1 3 1 0 1 3 0 0 1 3 3 1 2 1 3 1 0 0 0 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 1]
```

6.2 ASSIGNING LABELS TO THE PLAYRS

[ ]:
```
df_new=df
df_new["Labels"]=kmeans.labels_
df_new
```

[ ]:

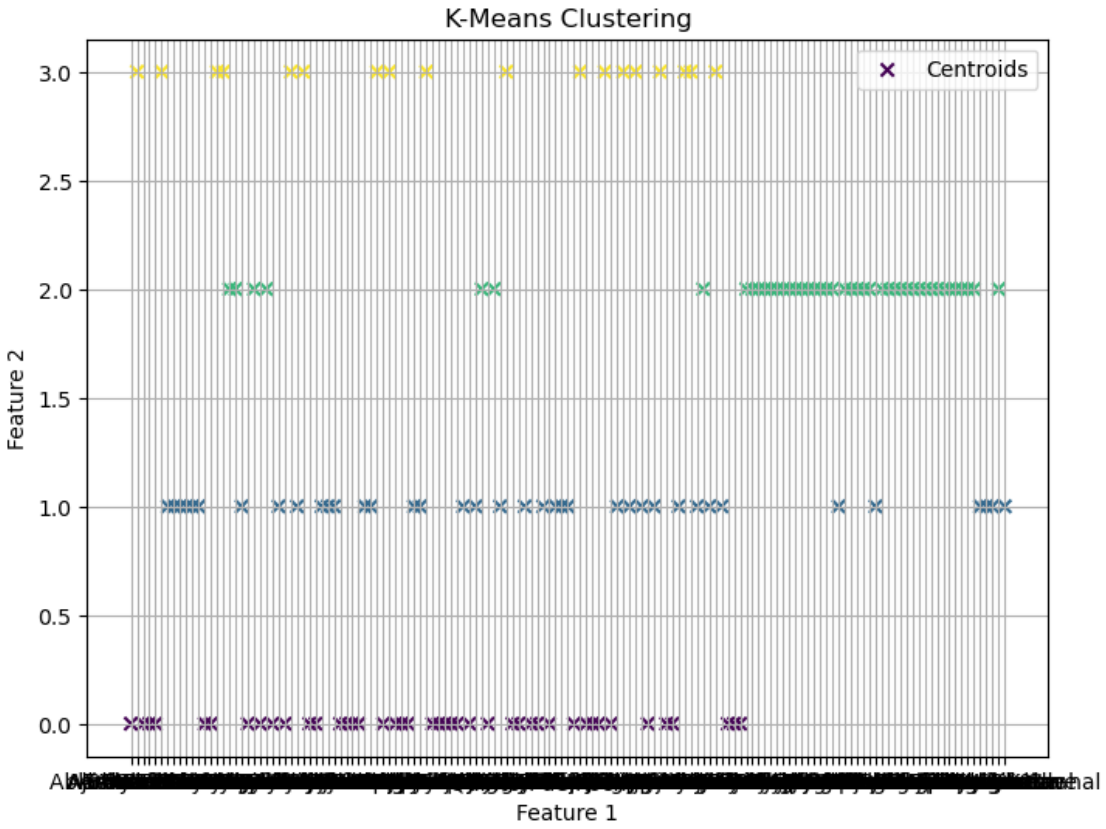
	PLAYER	Mat.x	Inns.x	NO	Runs.x	HS	Avg.x	BF	SR.x	\
0	Aaron Finch	10	9	1	134	46	16.75	100	134.00	
1	AB de Villiers	12	11	2	480	90	53.33	275	174.54	
2	Abhishek Sharma	3	3	2	63	46	63	33	190.90	
3	Ajinkya Rahane	15	14	1	370	65	28.46	313	118.21	
4	Alex Hales	6	6	0	148	45	24.66	118	125.42	
..	…	…	…	..	…	..	…	…	…	
138	Siddarth Kaul	0	0	0	0	0	0	0	0.00	
139	Trent Boult	0	0	0	0	0	0	0	0.00	
140	Umesh Yadav	0	0	0	0	0	0	0	0.00	
141	Vi0y Kumar	0	0	0	0	0	0	0	0.00	

```
142  Yuzvendra Chahal      0       0   0       0   0      0    0    0.00

     X100  …  Wkts  BBI  Avg.y   Econ   SR.y  X4w  X5w  y  Encoded_Names  \
0       0  …     0    0      0   0.00      0    0    0  0               1
1       0  …     0    0      0   0.00      0    0    0  0               0
2       0  …     0    0      0   0.00      0    0    0  0               2
3       0  …     0    0      0   0.00      0    0    0  0               3
4       0  …     0    0      0   0.00      0    0    0  0               5
..    …  …   …  …    …    …    …   …  …  … ..            …
138     0  …    21    0  26.04   8.28  18.85    0    0  0             126
139     0  …    18    0  25.88   8.84  17.55    0    0  0             133
140     0  …    20    0   20.9   7.86  15.95    0    0  0             134
141     0  …     2    0   32.5  16.95   11.5    0    0  0             135
142     0  …    12    0  30.25   7.26     25    0    0  0             142

    Labels
0        0
1        3
2        0
3        0
4        0
..     …
138      1
139      1
140      1
141      2
142      1

[143 rows x 27 columns]
```

6.3 PLOTTING THE CLUSTERS

[ ]:
```
# Plotting
plt.figure(figsize=(8, 6))

# Scatter plot of data points colored by labels
plt.scatter(df["PLAYER"],labels ,c=labels , marker='x', label='Centroids')
# plt.scatter(centroids[:, 0], centroids[:, 1], c='red', marker='o',␣
 ↪edgecolor='black', s=100)  # Ensure markers are visible

# plt.scatter(centroids[:, 0], centroids[:, 1], c='blue', marker='x',␣
 ↪label='Centroids')
# plt.scatter(centroids[:, 0], centroids[:, 1], c='blue', marker='o',␣
 ↪edgecolor='black', s=100)  # Ensure markers are visible
#
# plt.scatter(centroids[:, 0], centroids[:, 1], c='green', marker='x',␣
 ↪label='Centroids')
```

```
# plt.scatter(centroids[:, 0], centroids[:, 1], c='green', marker='o',␣
 ↪edgecolor='black', s=100)  # Ensure markers are visible

plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('K-Means Clustering')

plt.legend()
plt.grid(True)
plt.show()
```



6.4 PREDICTING THE PLAYERS

6.5 Top players of the league

[ ]:
```
top_order = df_new[df_new['Labels'] == 3].nlargest(3, ['Runs.x', 'SR.x'])
# top_order
middle_order=df_new[df_new['Labels'] == 0].nlargest(2, ['Runs.x', 'SR.x'])
# middle_order
all_rounders2=df_new[df_new['Labels'] == 2].nlargest(3, ['Runs.x', 'SR.
 ↪x','Wkts','Avg.y','SR.y'])
```

```
# all_rounders2
bowlers2=df_new[df_new['Labels'] == 1].nlargest(4, ['Wkts'])
# bowlers2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[13], line 5
      3 middle_order=df_new[df_new['Labels'] == 0].nlargest(2, ['Runs.x', 'SR.
 ↪x'])
      4 # middle_order
----> 5␣
 ↪all_rounders2=df_new[df_new['Labels'] == 2].nlargest(3, ['Runs.x', 'SR.x','Wk s','Avg.y','S
      6 # all_rounders2
      7 bowlers2=df_new[df_new['Labels'] == 1].nlargest(4, ['Wkts'])

File ~/miniconda3/lib/python3.9/site-packages/pandas/core/frame.py:7631, in␣
 ↪DataFrame.nlargest(self, n, columns, keep)
   7512 def nlargest(
   7513     self, n: int, columns: IndexLabel, keep: NsmallestNlargestKeep =␣
 ↪"first"
   7514 ) -> DataFrame:
   7515     """
   7516     Return the first `n` rows ordered by `columns` in descending order.
   7517
   (…)
   7629     Brunei      434000    12128      BN
   7630     """
-> 7631     return␣
 ↪selectn.SelectNFrame(self, n=n, keep=keep, columns=columns).nlargest()

File ~/miniconda3/lib/python3.9/site-packages/pandas/core/methods/selectn.py:57␣
 ↪in SelectN.nlargest(self)
     55 @final
     56 def nlargest(self):
---> 57     return self.compute("nlargest")

File ~/miniconda3/lib/python3.9/site-packages/pandas/core/methods/selectn.py:
 ↪201, in SelectNFrame.compute(self, method)
    199     dtype = frame[column].dtype
    200     if not self.is_valid_dtype_n_method(dtype):
--> 201         raise TypeError(
    202             f"Column {repr(column)} has dtype {dtype}, "
    203             f"cannot use method {repr(method)} with this dtype"
    204         )
    206 def get_indexer(current_indexer, other_indexer):
    207     """
    208     Helper function to concat `current_indexer` and `other_indexer`
```

```
    209     depending on `method`
    210     """

TypeError: Column 'Avg.y' has dtype object, cannot use method 'nlargest' with␣
 ↪this dtype
```

[ ]:
```
# print("Best Openers : ",top_order)
# print("Best Middle-players : ",middle_order)
# print("Best all rounders : ",all_rounders2)
# print("Best Bowlers : ",bowlers2)
total_best = pd.concat([top_order, middle_order, all_rounders2, bowlers2],␣
 ↪ignore_index=False)
total_best.sort_values(["Runs.x"])
```

[ ]:
```
                  PLAYER  Mat.x  Inns.x  NO  Runs.x   HS  Avg.x   BF    SR.x  \
126  Mitchell McCleOghan      0       0   0       0    0      0    0    0.00
129     Mujeeb Ur Rahman      0       0   0       0    0      0    0    0.00
101           Amit Mishra      0       0   0       0    0      0    0    0.00
136       Sandeep Sharma      0       0   0       0    0      0    0    0.00
33          Hardik Pandya     13      13   4     260   50  28.88  195  133.33
52          Manish Pandey     15      13   2     284   62  25.81  246  115.44
43              Karun Oir     13      12   0     301   54  25.08  221  136.19
62             Nitish Ra0     15      15   2     304   59  23.38  232  131.03
6           Andre Russell     16      14   3     316   88  28.72  171  184.79
89            Sunil Orine     16      16   0     357   75  22.31  188  189.89
73           Rishabh Pant     14      14   1     684  128  52.61  394  173.60
42      Kane Williamson      17      17   3     735   84   52.5  516  142.44

     X100  …  Wkts  BBI  Avg.y  Econ   SR.y  X4w  X5w  y  Encoded_Names  \
126     0  …    14    0  23.71  8.30  17.14    0    0  0             85
129     0  …    14    0  20.64  6.99  17.71    0    0  0             91
101     0  …    12    0     22  7.13   18.5    0    0  0              7
136     0  …    12    0  27.75  7.56     22    0    0  0            113
33      0  …    18    0  21.16  8.92  14.22    0    0  0             46
52      0  …     0    0      0  0.00      0    0    0  0             78
43      0  …     0    0      0  0.00      0    0    0  0             63
62      0  …     4    0     11  7.13   9.25    0    0  0             94
6       0  …    13    0   27.3  9.38  17.46    0    0  0              8
89      0  …    17    0  27.47  7.65  21.52    0    0  0            128
73      1  …     0    0      0  0.00      0    0    0  0            108
42      0  …     0    0      0  0.00      0    0    0  0             61

    Labels
126      1
129      1
101      1
```

```
136      1
33       3
52       2
43       2
62       2
6        3
89       3
73       0
42       0

[12 rows x 27 columns]
```

6.6 12-Players Balanced Team

[ ]:
```
top_order_batsmen = []
middle_order_batsmen = []
all_rounders = []
bowlers = []
for i, player in enumerate(df_new['PLAYER']):
    if df_new["Labels"][i] == 3:
#         print([player])
        top_order_batsmen.append(player)
    elif df_new["Labels"][i]== 0:
        middle_order_batsmen.append(player)
    elif df_new["Labels"][i]== 2:
        all_rounders.append(player)
    elif df_new["Labels"][i]==1:
        bowlers.append(player)
total_players=(top_order_batsmen[:3])+(middle_order_batsmen[:3])+(all_rounders[:
 ↪2])+(bowlers[:4])
df_total=pd.
 ↪DataFrame(total_players,columns=["Players"],index=[1,2,3,4,5,6,7,8,9,10,11,12])
print("Top Order Batsmen:", top_order_batsmen[:3])
print("Middle Order Batsmen:", middle_order_batsmen[:3])
print("All-rounders:", all_rounders[:2])
print("Bowlers:", bowlers[:4])
df_total
```

```
Top Order Batsmen: ['Andre Russell', 'Andrew Tye', 'Axar Patel']
Middle Order Batsmen: ['AB de Villiers', 'Ajinkya Rahane', 'Ambati Rayudu']
All-rounders: ['Aaron Finch', 'Abhishek Sharma']
Bowlers: ['Chris Woakes', 'Corey Anderson', 'Dan Christian', 'Mohammad Obi']
```

[ ]:
```
             Players
1     Andre Russell
2        Andrew Tye
3        Axar Patel
```

```
4    AB de Villiers
5    Ajinkya Rahane
6     Ambati Rayudu
7       Aaron Finch
8   Abhishek Sharma
9      Chris Woakes
10   Corey Anderson
11    Dan Christian
12     Mohammad Obi
```

[ ]: