RASHTREEYA SIKSHANA SAMITHI TRUST ® RV INSTITUTIONS

STATISTICS FOR DATA SCIENCE - AI235AT

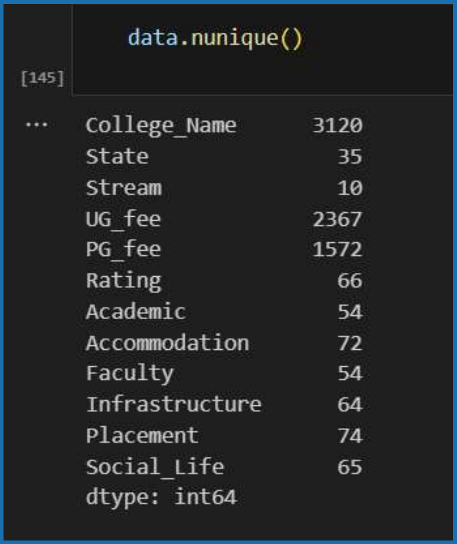
EXPERIENTIAL LEARNING

TOPICS : **INDIAN UNIVERSITIES**

QS WORLD RANKING

INDIAN UNIVERSITY

INTRODUCTION

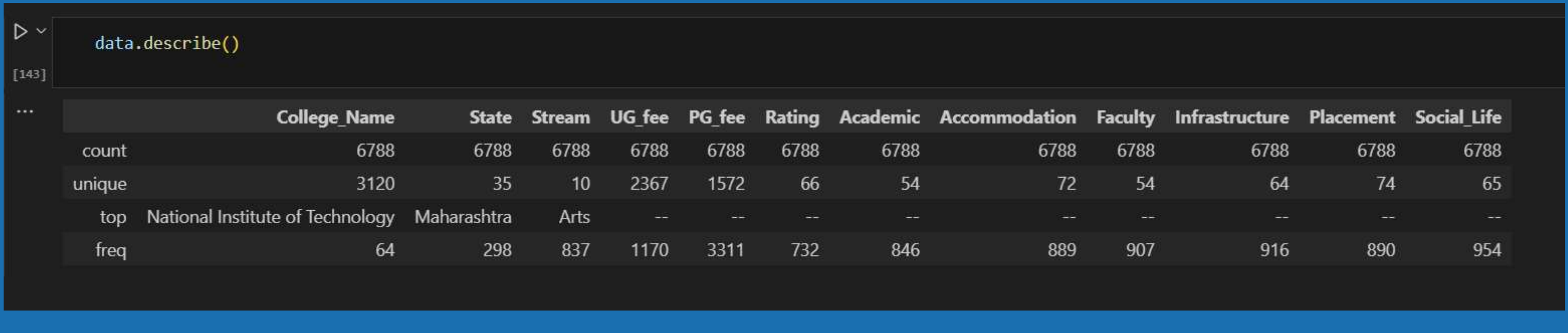


```
data.nunique()
```

[145]

```
College_Name       3120
State                35
Stream               10
UG_fee             2367
PG_fee             1572
Rating               66
Academic             54
Accommodation        72
Faculty              54
Infrastructure       64
Placement            74
Social_Life          65
dtype: int64
```

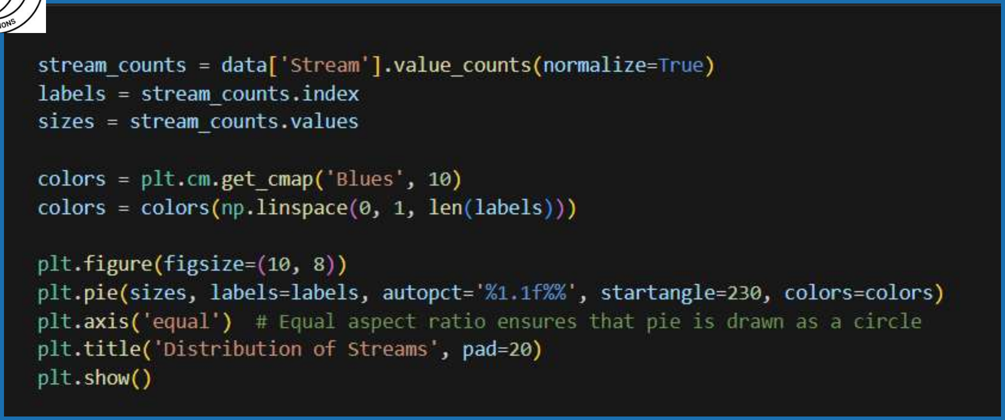
- The dataset includes information about universities/ colleges in India.
- It contains details like college name, state, different streams, undergraduate (UG) fees, postgraduate (PG) fees, and ratings.
- Ratings are provided for aspects like academic quality, accommodation, faculty, infrastructure, placement, and social life.
- The dataset can be used to compare different colleges based on these factors.
- The dataset contains 6788 rows and 12 columns.



```
data.describe()
```

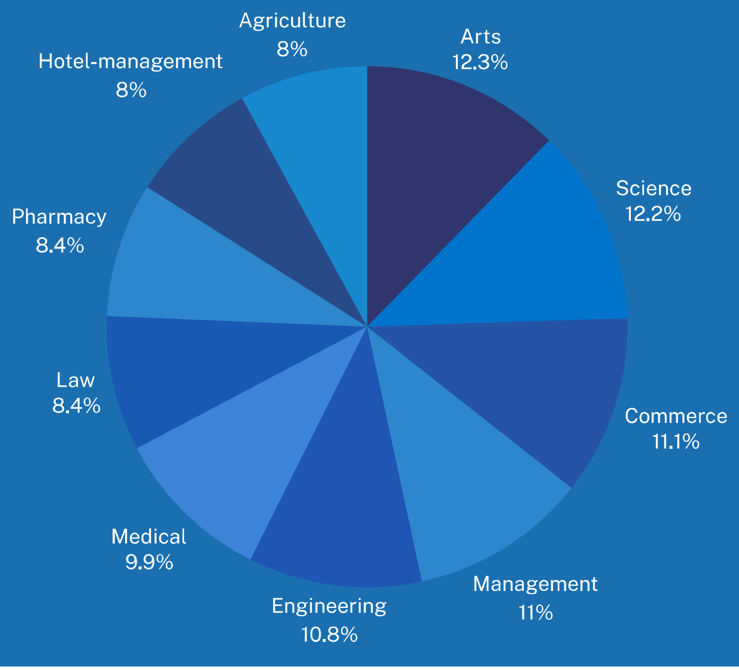
[143]

	College_Name	State	Stream	UG_fee	PG_fee	Rating	Academic	Accommodation	Faculty	Infrastructure	Placement	Social_Life
count	6788	6788	6788	6788	6788	6788	6788	6788	6788	6788	6788	6788
unique	3120	35	10	2367	1572	66	54	72	54	64	74	65
top	National Institute of Technology	Maharashtra	Arts	--	--	--	--	--	--	--	--	--
freq	64	298	837	1170	3311	732	846	889	907	916	890	954

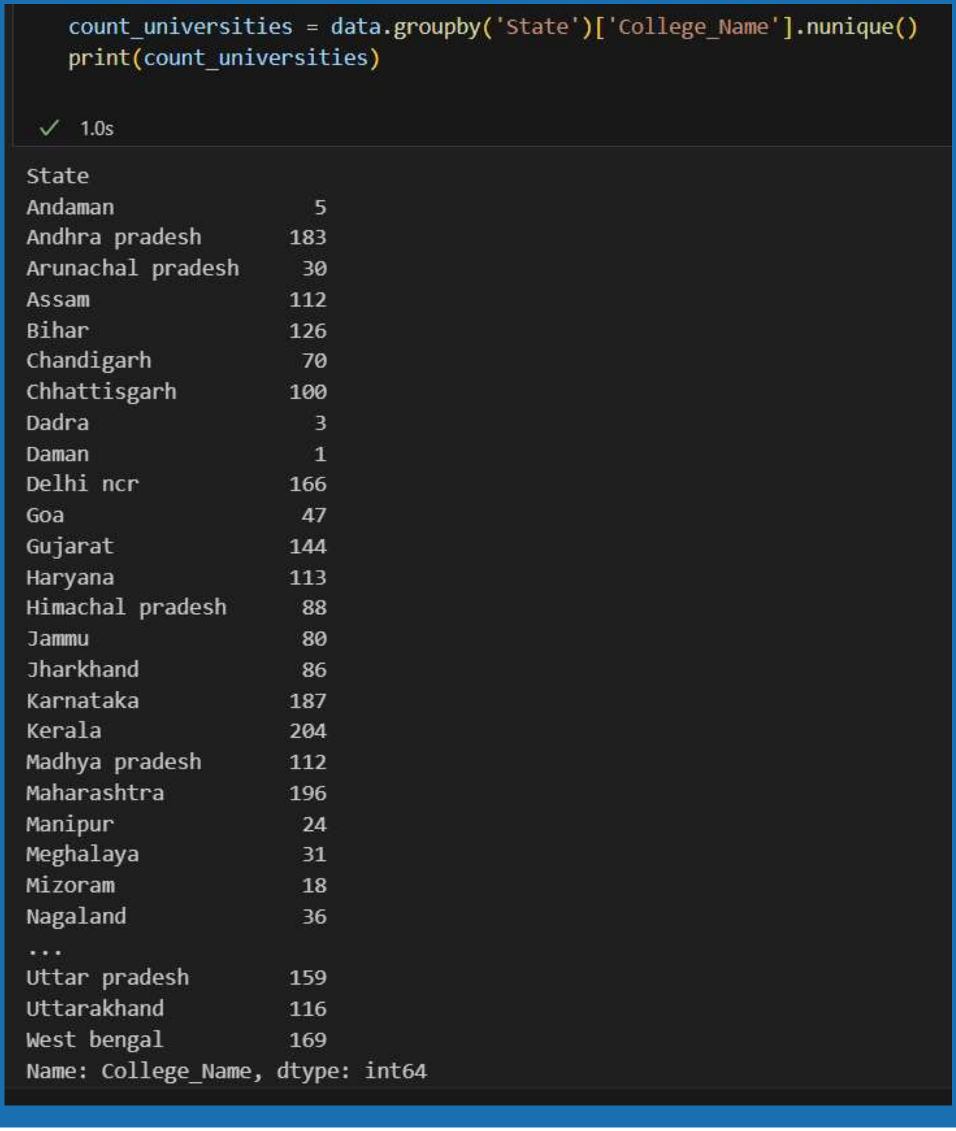


```
stream_counts = data['Stream'].value_counts(normalize=True)
labels = stream_counts.index
sizes = stream_counts.values

colors = plt.cm.get_cmap('Blues', 10)
colors = colors(np.linspace(0, 1, len(labels)))

plt.figure(figsize=(10, 8))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=230, colors=colors)
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle
plt.title('Distribution of Streams', pad=20)
plt.show()
```

DISTRIBUTION OF STREAMS



DISTRIBUTION OF UNIVERSITIES ACROSS DIFF STATES

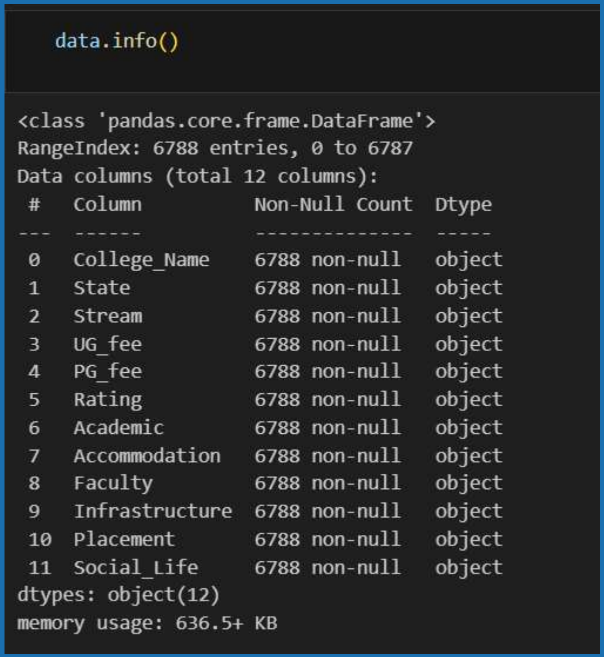


```
count_universities = data.groupby('State')['College_Name'].nunique()
print(count_universities)
```

✓ 1.0s

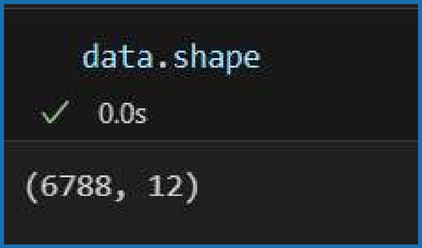
```
State
Andaman                  5
Andhra pradesh         183
Arunachal pradesh       30
Assam                  112
Bihar                  126
Chandigarh              70
Chhattisgarh           100
Dadra                    3
Daman                    1
Delhi ncr              166
Goa                     47
Gujarat                144
Haryana                113
Himachal pradesh        88
Jammu                   80
Jharkhand               86
Karnataka              187
Kerala                 204
Madhya pradesh         112
Maharashtra            196
Manipur                 24
Meghalaya               31
Mizoram                 18
Nagaland                36
...
Uttar pradesh          159
Uttarakhand            116
West bengal            169
Name: College_Name, dtype: int64
```

DATA DESCRIPTION



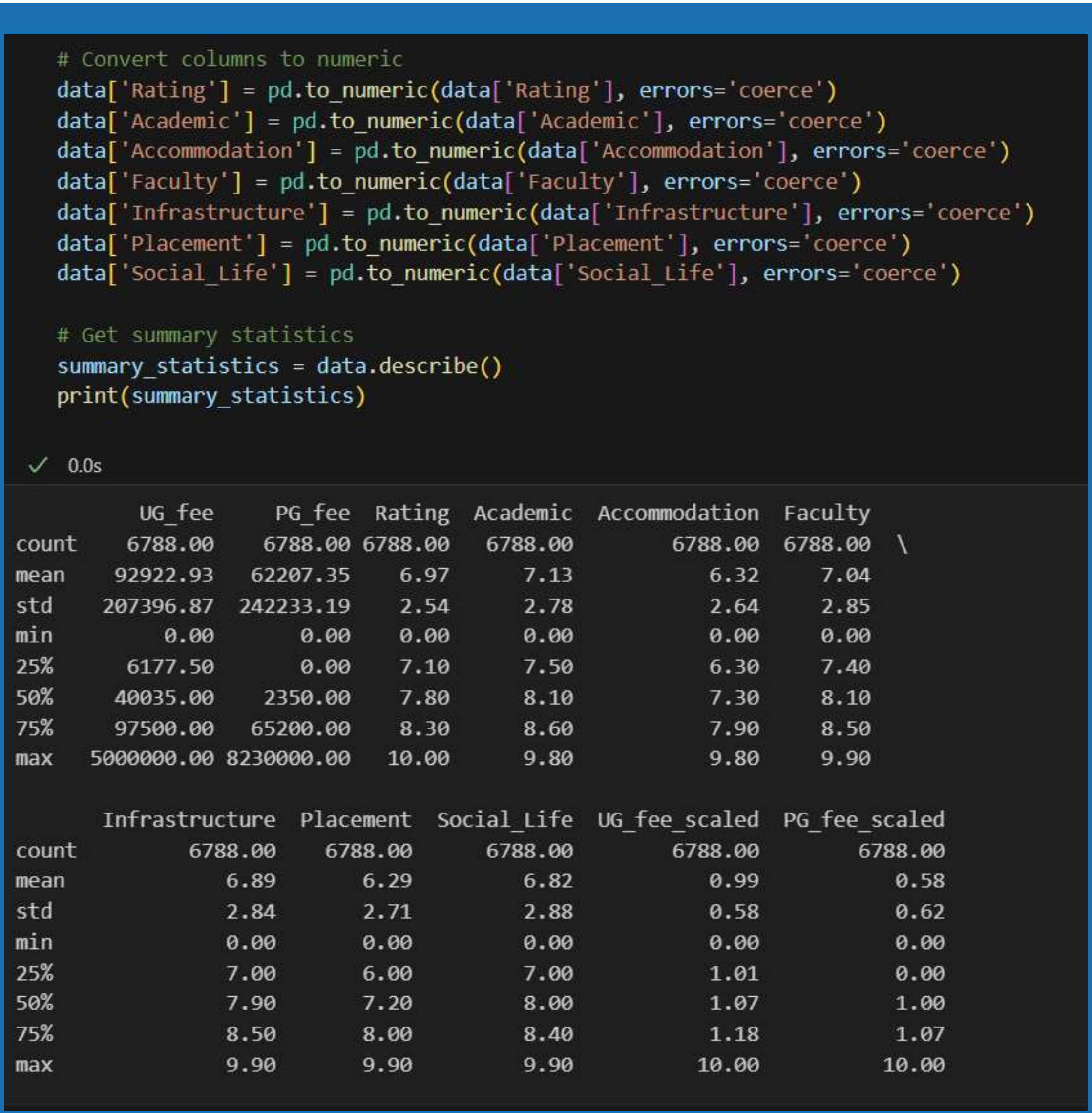
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6788 entries, 0 to 6787
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   College_Name    6788 non-null   object
 1   State           6788 non-null   object
 2   Stream          6788 non-null   object
 3   UG_fee          6788 non-null   object
 4   PG_fee          6788 non-null   object
 5   Rating          6788 non-null   object
 6   Academic        6788 non-null   object
 7   Accommodation   6788 non-null   object
 8   Faculty         6788 non-null   object
 9   Infrastructure  6788 non-null   object
 10  Placement       6788 non-null   object
 11  Social_Life     6788 non-null   object
dtypes: object(12)
memory usage: 636.5+ KB
```



```
data.shape
✓ 0.0s
```

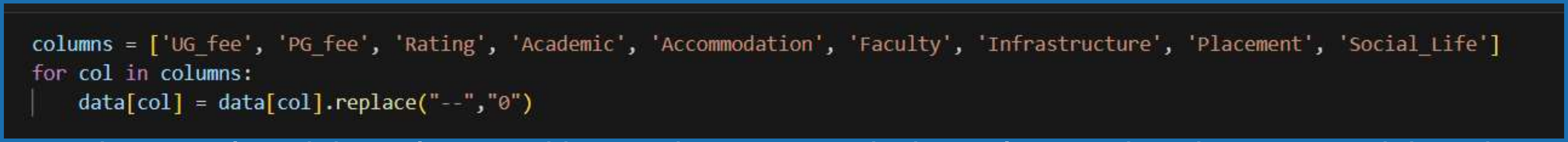
```
(6788, 12)
```



```
# Convert columns to numeric
data['Rating'] = pd.to_numeric(data['Rating'], errors='coerce')
data['Academic'] = pd.to_numeric(data['Academic'], errors='coerce')
data['Accommodation'] = pd.to_numeric(data['Accommodation'], errors='coerce')
data['Faculty'] = pd.to_numeric(data['Faculty'], errors='coerce')
data['Infrastructure'] = pd.to_numeric(data['Infrastructure'], errors='coerce')
data['Placement'] = pd.to_numeric(data['Placement'], errors='coerce')
data['Social_Life'] = pd.to_numeric(data['Social_Life'], errors='coerce')

# Get summary statistics
summary_statistics = data.describe()
print(summary_statistics)
✓ 0.0s
```

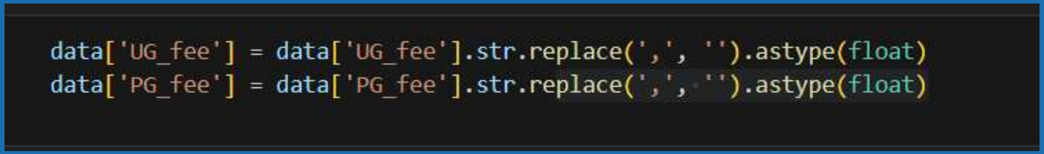
```
           UG_fee      PG_fee  Rating  Academic  Accommodation  Faculty  \
count     6788.00     6788.00 6788.00   6788.00        6788.00  6788.00   
mean     92922.93    62207.35    6.97      7.13           6.32     7.04   
std     207396.87   242233.19    2.54      2.78           2.64     2.85   
min          0.00        0.00    0.00      0.00           0.00     0.00   
25%       6177.50        0.00    7.10      7.50           6.30     7.40   
50%      40035.00     2350.00    7.80      8.10           7.30     8.10   
75%      97500.00    65200.00    8.30      8.60           7.90     8.50   
max    5000000.00  8230000.00   10.00      9.80           9.80     9.90   

       Infrastructure  Placement  Social_Life  UG_fee_scaled  PG_fee_scaled  
count         6788.00    6788.00      6788.00        6788.00        6788.00  
mean             6.89       6.29         6.82           0.99           0.58  
std              2.84       2.71         2.88           0.58           0.62  
min              0.00       0.00         0.00           0.00           0.00  
25%              7.00       6.00         7.00           1.01           0.00  
50%              7.90       7.20         8.00           1.07           1.00  
75%              8.50       8.00         8.40           1.18           1.07  
max              9.90       9.90         9.90          10.00          10.00  
```

DATA CLEANING



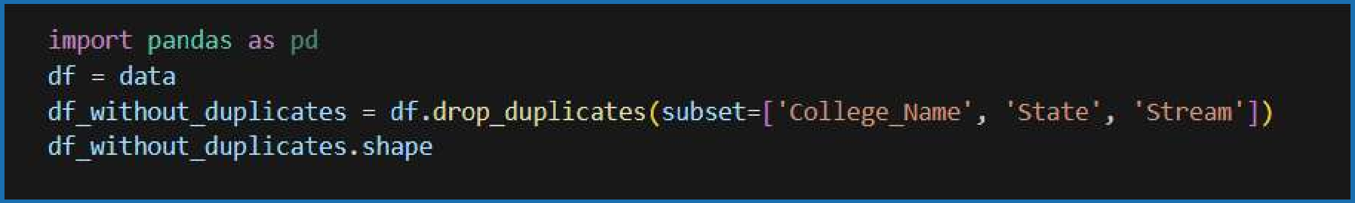
```
columns = ['UG_fee', 'PG_fee', 'Rating', 'Academic', 'Accommodation', 'Faculty', 'Infrastructure', 'Placement', 'Social_Life']
for col in columns:
    data[col] = data[col].replace("--","0")
```

- We have replaced the values -- with 0 as those were missing values and we haven’t used them in any analysis by keeping condition greater than 0.

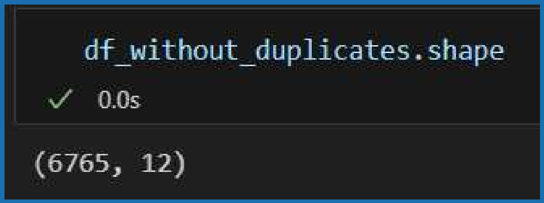


```
data['UG_fee'] = data['UG_fee'].str.replace(',', '').astype(float)
data['PG_fee'] = data['PG_fee'].str.replace(',', '').astype(float)
```

- As UG_fee and PG_fee data was having commas, it was getting considered as string so we removed it using the above code.



```
import pandas as pd
df = data
df_without_duplicates = df.drop_duplicates(subset=['College_Name', 'State', 'Stream'])
df_without_duplicates.shape
```



```
df_without_duplicates.shape
✓ 0.0s

(6765, 12)
```

- we have encountered some data which has same college name,state and stream so we deleted duplicate values . So after cleaning the data we have 6765 rows and 12 columns.It keeps row that was encountered first and doesn’t consider others.

THANK YOU