Exploratory Data Analysis

Dataset Used : Coursera Course Dataset

URL : https://www.kaggle.com/datasets/siddharthm1698/coursera-course-dataset

Data Brief

Course dataset scrapped from Coursera website. This dataset contains mainly 6 columns and 890 course data. The detailed description:

1. **course_title : Contains the course title.**
2. **course_organization : It tells which organization is conducting the courses.**
3. **courseCertificatetype : It has details about what are the different certifications available in courses.**
4. **course_rating : It has the ratings associated with each course.**
5. **course_difficulty : It tells about how difficult or what is the level of the course.**
6. **coursestudentsenrolled : It has the number of students that are enrolled in the course.**

Data Loading and Basic Review

Required Modules

In [58]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as sps
```

Data Loading and Basic Exploration

In [59]:

```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [60]:

```
df=pd.read_csv("/kaggle/input/coursera-course-dataset/coursea_data.csv")
df.head()
```

In [61]:

```
df=df.drop("Unnamed: 0",axis=1)
df.info()
```

So, 1 numarical object only. But, we can turn some others to numarical too.

In [62]:

```
df.describe()
```

Mean course rating is 4.677329. Quite high, as the rating can be given from 0-5. Minimum is 3.3, highest is 5 - proves so.

Initial plan for data exploration

Data Exploration

1. **Ploting course_rating to get a overview of the distribution.**
2. **analyzing course Certificate types values.**

Data Cleaning

1. **Deleting first Unnamed column**
2. **Deleting course name - not necessary now; as all the values are unique**

Data Exploration

Basic Rating distribution :

In [63]:

```
# Ploting course_rating to get a overview of the distribution.
plt.boxplot(df['course_rating'])
```

In [64]:

```
# Ploting course_rating to get a overview of the distribution.
df['course_rating'].hist()
```

Findings:

Average course rating is quite higher, compared to lowest and maximum value.

Rating distribution per course difficulty :

In [65]:

```
g = df.groupby('course_difficulty')['course_rating']
fig, axes = plt.subplots(g.ngroups, sharex=True, figsize=(4, 6))

for i, (type,rating) in enumerate(g):
    ax = rating.plot.hist('course_rating',ax=axes[i], legend=False,title=type)
fig.tight_layout()
```

Insight:

Advanced courses' rating has some ups-and downs; maybe due to low frequency.
Beginner course has distribution quite similiar to total rating chart.
Intermidiate course's rating top is not as sharp of others, that may say - as the participants has some knowledge on the topic, they can judge better and being critical.

Rating distribution per course type :

In [66]:

```
g = df.groupby('course_Certificate_type')['course_rating']
fig, axes = plt.subplots(g.ngroups, sharex=True, figsize=(4, 6))

for i, (type,rating) in enumerate(g):
    ax = rating.plot.hist('course_rating',ax=axes[i], legend=False,title=type,bins=10)
fig.tight_layout()
```

In [67]:

```
g.describe()
```

Findings and Insight:

1. **Specializations has lower mean value than courses, but the distribution is interesting. specialization has good distribution values on right, but normal courses are on left.**

Combined

In [68]:

```
g = df.groupby(['course_difficulty','course_Certificate_type'])['course_rating']
fig, axes = plt.subplots(g.ngroups, sharex=True, figsize=(4, 20))

for i, (type,rating) in enumerate(g):
    axes[i].set_ylim(0, 100)
    ax = rating.plot.hist('course_rating',ax=axes[i], legend=False,title=type[0]+"-"+typ
e[1].lower(),bins=10)
fig.tight_layout()
```

Analyzing course Certificate types values.

In [69]:

```
df.groupby('course_difficulty').course_difficulty.value_counts().unstack().plot.barh()
```

In [70]:

```
df.groupby('course_Certificate_type').course_Certificate_type.value_counts().unstack().p
lot.barh()
```

Data Cleaning

1. **Deleting first Unnamed column**
2. **Deleting course name - not necessary now; as all the values are unique**

In [71]:

```
df=df.drop(['course_title'],axis=1)
```

Feature Engineering

1. **Modifying course_students_enrolled column**

In [72]:

```
df_fe1=df.copy()
```

In [73]:

```
def course_students_enrolled_modifier(x):
    return x[:-2]
```

In [74]:

```
df_fe1['course_students_enrolled_modified']=df_fe1['course_students_enrolled'].apply(cour
se_students_enrolled_modifier)
df_fe1['course_students_enrolled_modified']=df_fe1['course_students_enrolled_modified'].a
pply(pd.to_numeric)
df_fe1 =df_fe1.drop(['course_students_enrolled'],axis=1)
df_fe1
```

1. **Modifying course_difficulty column to numarical**

In [75]:

```
def course_difficulty_modifier(x):
    if x=="Beginner":
        return "0"
    elif x=="Intermediate":
        return "1"
    elif x=="Mixed":
        return "0.5"
    elif x=="Advanced":
        return "2"
    else:
        return "0"
"""as most courses are beginner level, we are assuming undefined will be beginner too."""
```

In [76]:

```
df_fe1['course_difficulty_modified']=df_fe1['course_difficulty'].apply(course_difficulty_
modifier)
df_fe1['course_difficulty_modified']=df_fe1['course_difficulty_modified'].apply(pd.to_num
eric)
df_fe1 =df_fe1.drop(['course_difficulty'],axis=1)
df_fe1
```

Data Exploration of newly engineered columns

In [77]:

```
df_fe1[['course_difficulty_modified','course_students_enrolled_modified']].describe()
```

course_students_enrolled_modified has some empty columns, so we have to fill them.

In [78]:

```
df_fe1[['course_students_enrolled_modified']].plot.hist()
```

so , most of the frequencies are in between 0-10, so, using average-1; so avoid the effect of outliers.

In [79]:

```
df_fe1['course_students_enrolled_modified'].fillna((df_fe1['course_students_enrolled_modi
fied'].mean()-1), inplace=True)
df_fe1[['course_difficulty_modified','course_students_enrolled_modified']].describe()
```

In [80]:

```
df_numaric=df_fe1.select_dtypes(include=np.number)
```

Finding relation between columns

In [81]:

```
corrM = df_numaric.corr()
corrM
```

In [82]:

```
df_numaric.plot.scatter(x='course_rating', y='course_difficulty_modified',c='DarkBlue')
```

Findings :

No effective coorelation.

Key Findings and Insights

1. **Average course rating is quite higher, compared to lowest and maximum value. So, the cours quality is being maintained.**
2. **Advanced courses' rating has some ups-and downs; maybe due to low frequency.**
3. **Beginner course has distribution quite similiar to total rating chart, as big portion of the data is from them, and he number of beginner level courses are high.**
4. **Intermidiate course's rating top is not as sharp of others, that may say - as the participants has some knowledge on the topic, they can judge better and being critical.**
5. **Specializations has lower mean value than courses, but the distribution is interesting. specialization has good distribution values on right, but normal courses are on left.**
6. **No effective coorelation between course_difficulty,course_students_enrolled, course rating.**