Descriptive Insights on

Movie dataset

Using pandas,seaborn,plotly,matplotlib in python

ABOUT DATA:

DATA SOURCE: https://github.com/Fathima33/statistics-project

DATASET DESCRIPTION:

- Title= title of the movie
- Year=year the movie released
- Genre= Category like drama,comedy etc
- Duration=duration of the movie
- Director=director of the movie
- Rating=Rating the movie got

OBJECTIVE:

To perform descriptive statistical measures.

This project aims to harness the power of descriptive statistical techniques for in-depth exploration of media-related data. By analyzing attributes like titles, genres, ratings, and more, we seek to unveil patterns and trends.

SAMPLE DATA OF DATASET:

	Title	Year	Genre	Duration	Director	Rating	Popularity
0	What Is It?	2005	Drama	72	Crispin Glover	5.6	21.83
1	Glitter	2001	Drama	104	Vondie Curtis-Hall	2.2	81.69
2	The Attic Expeditions	2001	Comedy	100	Jeremy Kasten	5.0	42.08
3	Men in Black II	2002	Action	88	Barry Sonnenfeld	6.2	98.60
4	Star Wars: Episode II - Attack of the Clones	2002	Action	142	George Lucas	6.5	99.58

UNIVARIATE ANALYSIS

MEASURES OF CENTRAL TENDENCY:

Using measures like mean,median,mode can find

Average rating

```
1 Average rating of movies
```

```
1 round(df['Rating'].mean(),3)
```

```
5.38
```

Most common rating of the movies

```
1 df['Rating'].mode()
```

```
0    5.6
```

Similarly it can be applied to the popularity and duration column

```
Average duration of movies
```

```
round(df['Duration'].mean(),3)
```

```
95.235
```

From this we can conclude the average duration of movies is 95 minutes.

```
df['Duration'].mode()
```

```
0    90
Name: Duration, dtype: int64
```

Most of the movies are 90 minutes long.Measures of central tendency helps in univariate analysis,i.e it helps in understanding particular variable

MEASURES OF DISPERSION:

Measures of dispersion provide insights into the spread or variability of data points within a dataset.

For Rating column:

RANGE: span of ratings in your dataset.

```
df["Rating"].max()
```

```
9.7
```

```
df.Rating.min()
```

```
1.1
```

```
df["Rating"].max()-df.Rating.min()
```

```
8.6
```

Range of 8.6 indicates diverse variability in the ratings column means there are movies that are highly rated as well as lowly rated.

Limitations of Range:

It can be easily affected by outlier and thus can sometimes misinterpret the data.

VARIANCE:

```
1 round(df.Rating.var(),2)
```

```
1.56
```

The variance is the average of the squared differences between each data point and the mean.

It is expressed in squared units of the original data.

It measures the average extent to which individual data points deviate from the mean.

Variance provides a measure of the overall variability in the dataset.

It's sensitive to outliers because it involves squaring the differences from the mean.

variance of 1.55 suggests that the ratings in the dataset are relatively consistent and not too far from the mean.

STANDARD DEVIATION:

```
1 round(df.Rating.std(),2)
```

```
1.25
```

The standard deviation is the square root of the variance.

It is expressed in the same units as the original data.

It measures the average amount by which individual data points deviate from the mean.

Standard deviation provides a measure of the typical or average deviation from the mean.

It's used more commonly than variance because it's in the same units as the data and is easier to interpret.

standard deviation of 1.25 suggests that the ratings in the dataset are consistent and not too dispersed.

To analyze top 3 genres average rating and duration.

Genre	Rating	Duration
Action	4.838083	96.390146
Comedy	5.551317	94.711377
Drama	5.793260	97.441449

PROBABILITY DENSITY FUNCTION:

a way to describe the likelihood of different outcomes in a random experiment

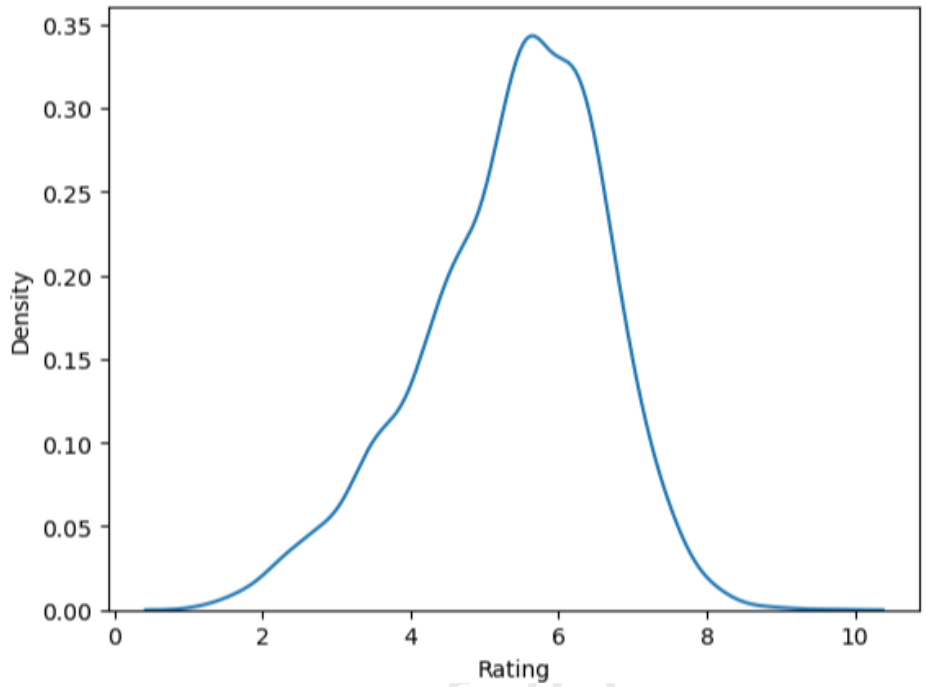
and is applied to describe continuous probability distributions.

CUMULATIVE DISTRIBUTIVE FUNCTION:

The CDF tells the chance of getting a value up to a certain point. It adds up the probabilities as we move along the values.

```
1 sns.kdeplot(df.Rating)
```

```
<AxesSubplot:xlabel='Rating', ylabel='Density'>
```



Rating column follows the normal distribution similarly the popularity column and duration column also follows the normal distribution.

Questions :

1. Probability of movies getting more than rating 8.
2. Probability of movies getting popularity between 80 and 90.
3. to find the probability of the movies having more than 80 minutes

Using scipy.stats library to solve the questions.

```
1 import scipy.stats as stats
```

```
1 prob=1-stats.norm.cdf(8,loc=df.Rating.mean(),scale=df.Rating.std())
```

```
1 print(f"percentage of getting more than rating 8: {round(prob*100,2)}")
```

```
percentage of getting more than rating 8: 1.79
```

```
prob=1-stats.norm.cdf(80,loc=df.Duration.mean(),scale=df.Duration.std())
```

```
print(f"percentage of movies having more than 80 minutes: {round(prob*100,2)}")
```

```
percentage of movies having more than 80 minutes: 88.36
```

to find the probability movies having popularity between 80 and 90

```
prob = stats.norm.cdf(90, loc=df.Popularity.mean(), scale=df.Popularity.std()) - \
       stats.norm.cdf(80, loc=df.Popularity.mean(), scale=df.Popularity.std())

```

```
print(f"percentage of movies having popularity between 80 and 90: {round(prob*100,2)}")
```

```
percentage of movies having popularity between 80 and 90: 6.64
```

PROBABILITY MASS FUNCTION:

is used to describe discrete probability distributions, that is it shows the exact probabilities for each outcome.

Using plotly to analyze rating column

```
from collections import Counter as c
```

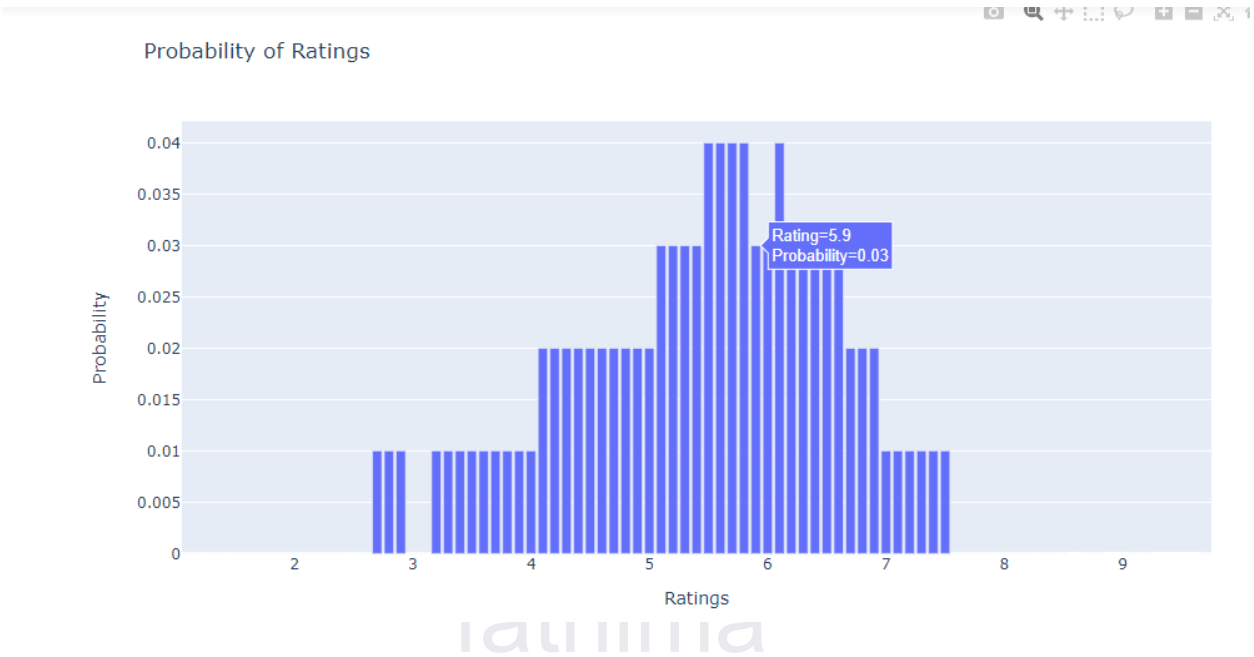
```
pairs=c(df['Rating'])
```

```
pmf={i:round(j/len(df['Rating']),2) for i,j in pairs.items()}
```

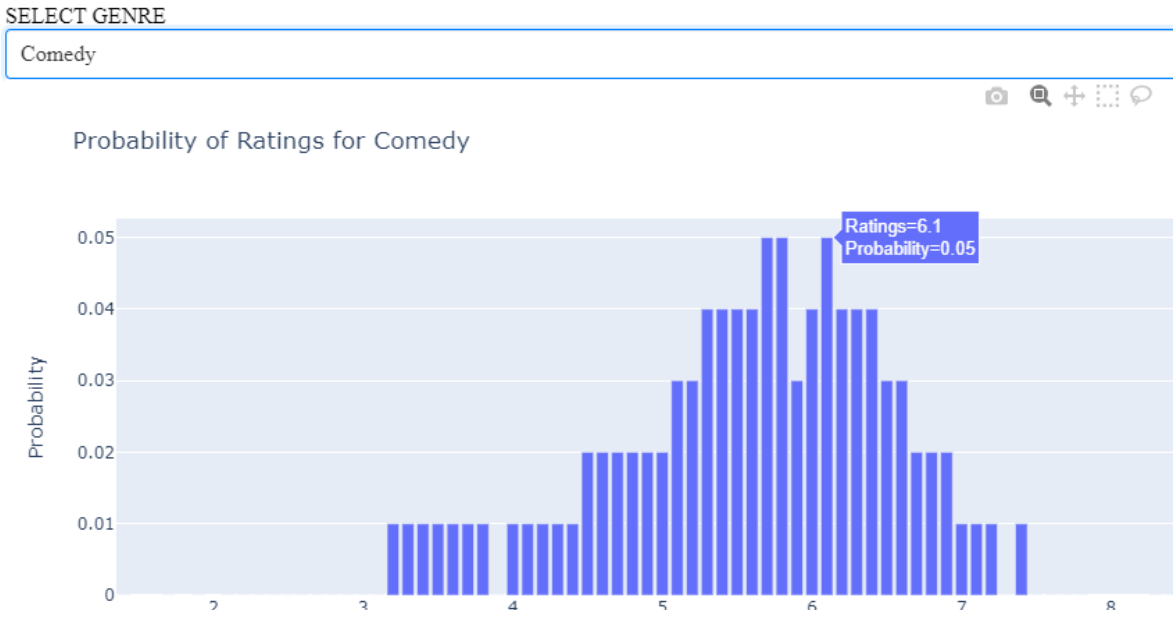
```
import plotly.express as px

pmf_df = [{"Outcome": outcome, "Probability": probability} for outcome, probability in pmf.items()]
fig = px.bar(pmf_df, x="Outcome", y="Probability", title="Probability of Ratings")
fig.update_xaxes(title="Ratings")
fig.update_yaxes(title="Probability")
fig.show()
```

It results in interactive plotly visual where we can analyze the ratings probability.



To analyze for different genre



BERNOULLI THEOREM:

The Bernoulli distribution is suitable for situations where you're dealing with a single trial or experiment that has two possible outcomes (success/failure)

to analyze whether a movie's rating is above (7) a certain threshold (success) or not (failure)

```
#using scipy library
from scipy.stats import bernoulli
threshold = 7
prob_success = sum(df['Rating'] > threshold) / len(df)
bernoulli_dist = bernoulli(prob_success)
prob_high_rating = bernoulli_dist.pmf(1)

print(f"Probability of a movie having a rating above {threshold}: {prob_high_rating:.2f}")
```

```
Probability of a movie having a rating above 7: 0.07
```

Conclusion:

7% chance of randomly selecting a movie with a rating above 7 from this dataset. This might suggest that, in this specific dataset, highly-rated movies are relatively uncommon.

BINOMIAL THEOREM:

The binomial distribution is a statistical distribution that describes the number of successes in a fixed number of independent trials or experiments, where each trial has only two possible outcomes: success or failure.

To find the probability of selecting at least 100 movies in 150 having rating more than 5

```
prob=sum(df["Rating"]>5)/len(df)
success=100
n_trails=150
probability=1-binom.cdf(success-1,n_trails,prob)
f"probability of selecting 100 movies having rating more than 5 is {probability:.4f}"
```

```
'probability of selecting 100 movies having rating more than 5 is 0.4048'
```

40% chance in selecting atleast 100 movies having rating more than 5.