Data Preprocessing

```
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt
```

```
df = pd.read_csv('/content/Global YouTube Statistics.csv', encoding='unicode_escape', on_bad_lines='skip')
```

```
df.head()
```

	rank	Youtuber	subscribers	video views	category	Title	uploads	Cc
0	1	T-Series	245000000	2.280000e+11	Music	T-Series	20082	
1	2	YouTube Movies	170000000	0.000000e+00	Film & Animation	youtubemovies	1	
2	3	MrBeast	166000000	2.836884e+10	Entertainment	MrBeast	741	
3	4	Cocomelon - Nursery Rhymes	162000000	1.640000e+11	Education	Cocomelon - Nursery Rhymes	966	
4	5	SET India	159000000	1.480000e+11	Shows	SET India	116536	

5 rows × 28 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 995 entries, 0 to 994
Data columns (total 28 columns):
 #   Column                                   Non-Null Count  Dtype  
---  ------                                   --------------  -----  
 0   rank                                     995 non-null    int64  
 1   Youtuber                                 995 non-null    object 
 2   subscribers                              995 non-null    int64  
 3   video views                              995 non-null    float64
 4   category                                 949 non-null    object 
 5   Title                                    995 non-null    object 
 6   uploads                                  995 non-null    int64  
 7   Country                                  873 non-null    object 
 8   Abbreviation                             873 non-null    object 
 9   channel_type                             965 non-null    object 
 10  video_views_rank                         994 non-null    float64
 11  country_rank                             879 non-null    float64
 12  channel_type_rank                        962 non-null    float64
 13  video_views_for_the_last_30_days         939 non-null    float64
 14  lowest_monthly_earnings                  995 non-null    float64
 15  highest_monthly_earnings                 995 non-null    float64
 16  lowest_yearly_earnings                   995 non-null    float64
 17  highest_yearly_earnings                  995 non-null    float64
 18  subscribers_for_last_30_days             658 non-null    float64
 19  created_year                             990 non-null    float64
 20  created_month                            990 non-null    object 
 21  created_date                             990 non-null    float64
 22  Gross tertiary education enrollment (%)  872 non-null    float64
 23  Population                               872 non-null    float64
 24  Unemployment rate                        872 non-null    float64
 25  Urban_population                         872 non-null    float64
 26  Latitude                                 872 non-null    float64
 27  Longitude                                872 non-null    float64
dtypes: float64(18), int64(3), object(7)
memory usage: 217.8+ KB
```

Data Reduction 1 - Feature Subset Selection

Features `Latitude` and `Longitude` is not likely to have a significant relation with the rank of a youtube channel. And the feature `subscribers_for_last_30_days` have significant amount of missing values. Hence we drop the three columns from the dataframe.

```
df = df.iloc[:, 0:26]
df = df.drop(['subscribers_for_last_30_days'], axis=1)
```

Data Cleaning 1 - Drop examples with missing values

Now we drop the samples with missing values.

```
df.dropna(inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 808 entries, 0 to 994
Data columns (total 25 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   rank                                    808 non-null    int64
 1   Youtuber                                808 non-null    object
 2   subscribers                             808 non-null    int64
 3   video views                             808 non-null    float64
 4   category                                808 non-null    object
 5   Title                                   808 non-null    object
 6   uploads                                 808 non-null    int64
 7   Country                                 808 non-null    object
 8   Abbreviation                            808 non-null    object
 9   channel_type                            808 non-null    object
 10  video_views_rank                        808 non-null    float64
 11  country_rank                            808 non-null    float64
 12  channel_type_rank                       808 non-null    float64
 13  video_views_for_the_last_30_days        808 non-null    float64
 14  lowest_monthly_earnings                 808 non-null    float64
 15  highest_monthly_earnings                808 non-null    float64
 16  lowest_yearly_earnings                  808 non-null    float64
 17  highest_yearly_earnings                 808 non-null    float64
 18  created_year                            808 non-null    float64
 19  created_month                           808 non-null    object
 20  created_date                            808 non-null    float64
 21  Gross tertiary education enrollment (%) 808 non-null    float64
 22  Population                              808 non-null    float64
 23  Unemployment rate                       808 non-null    float64
 24  Urban_population                        808 non-null    float64
dtypes: float64(15), int64(3), object(7)
memory usage: 164.1+ KB
```

Data type of `created_year` is float. Convert data type into int.

```
df = df.astype({'created_year':int})
```

```
df['created_year'].describe()
```

```
count     808.000000
mean     2012.235149
std         4.287575
min      1970.000000
25%      2009.000000
50%      2013.000000
75%      2015.000000
max      2022.000000
Name: created_year, dtype: float64
```

Data Cleaning 2 - Drop incorrect data

The minimum value for the attribute `created_year` is 1970. This is not possible since Youtube was created in 2005. Remove the examples with `created_year` value less than 2005.

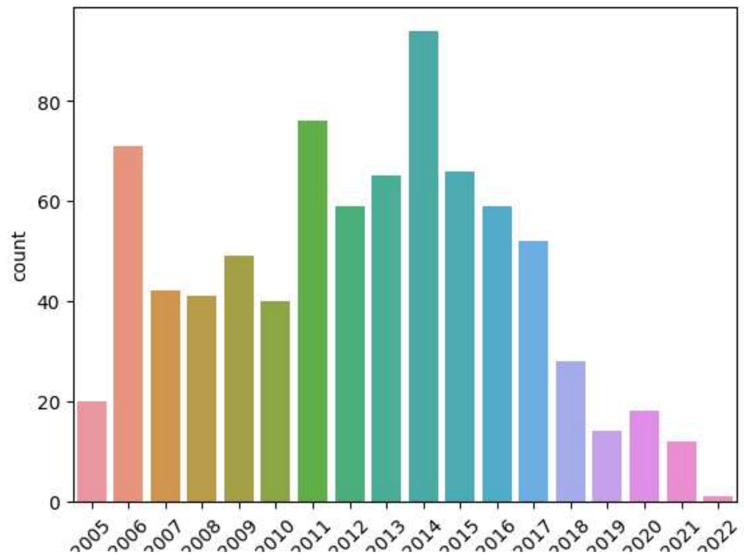
```
df.drop(df.loc[df['created_year'] < 2005].index, inplace=True)
df.shape
```

```
(807, 25)
```

Plot a countplot of the number of youtube channels that were created every year starting from 2005.

```
year_plot = sns.countplot(x=df['created_year'], data=df['Youtuber'])
year_plot.set_xticklabels(year_plot.get_xticklabels(), rotation=45)
```

```
[Text(0, 0, '2005'),
 Text(1, 0, '2006'),
 Text(2, 0, '2007'),
 Text(3, 0, '2008'),
 Text(4, 0, '2009'),
 Text(5, 0, '2010'),
 Text(6, 0, '2011'),
 Text(7, 0, '2012'),
 Text(8, 0, '2013'),
 Text(9, 0, '2014'),
 Text(10, 0, '2015'),
 Text(11, 0, '2016'),
 Text(12, 0, '2017'),
 Text(13, 0, '2018'),
 Text(14, 0, '2019'),
 Text(15, 0, '2020'),
 Text(16, 0, '2021'),
 Text(17, 0, '2022')]
```

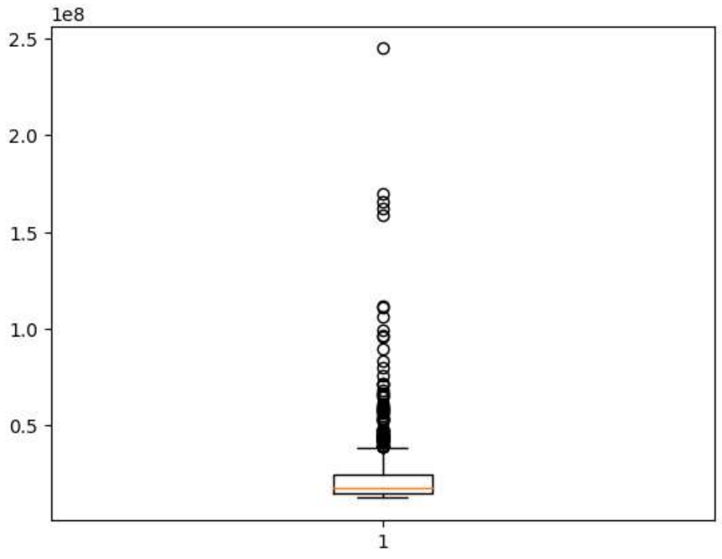


Data Cleaning 3 - Remove outliers

Plot the box plot for `subscribers`

```
plt.boxplot(df['subscribers'])
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7c9b859182b0>,
  <matplotlib.lines.Line2D at 0x7c9b85918550>],
 'caps': [<matplotlib.lines.Line2D at 0x7c9b859187f0>,
  <matplotlib.lines.Line2D at 0x7c9b85918a90>],
 'boxes': [<matplotlib.lines.Line2D at 0x7c9b85ae3fd0>],
 'medians': [<matplotlib.lines.Line2D at 0x7c9b85918d30>],
 'fliers': [<matplotlib.lines.Line2D at 0x7c9b85918fd0>],
 'means': []}
```



Remove samples with `subscribers` greater than 0.35e8.

```
df.drop(df.loc[df['subscribers'] > 0.35e8].index, inplace=True)
df.shape
```

```
(715, 25)
```

Normalization

The attribute `subscribers` can vary drastically depending on the size of the country. Hence we have to normalize it before applying data analysis techniques. Use **Min-Max Normalization** to map the feature values to the range [0, 1].

```
df_scaled = df.copy()
column = 'subscribers'
df_scaled[column] = (df_scaled[column] - df_scaled[column].min()) / (df_scaled[column].max() - df_scaled[column].min())
```

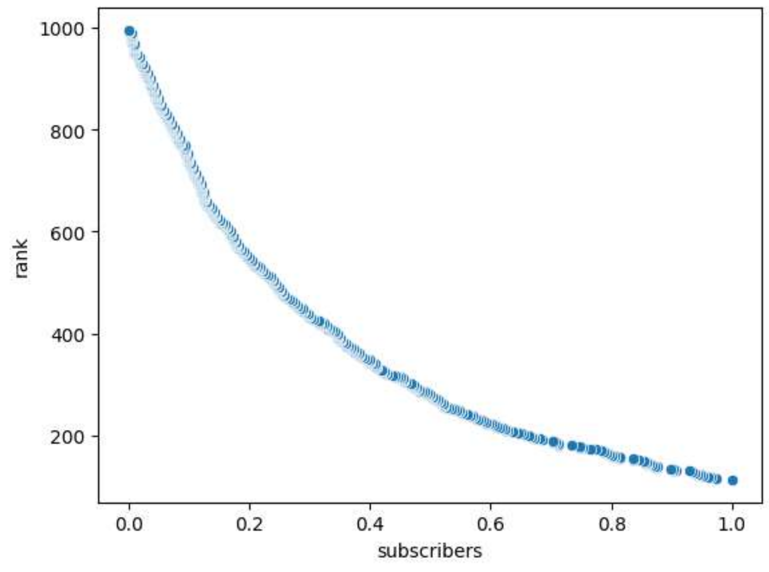
```
df_scaled['subscribers'].describe()
```

```
count    715.000000
mean       0.272195
std        0.244397
min        0.000000
25%        0.084071
50%        0.185841
75%        0.398230
max        1.000000
Name: subscribers, dtype: float64
```

Plot a scatter plot of `rank` vs. `subscribers`.

```
col1 = 'subscribers'
col2 = 'rank'
sns.scatterplot(x=col1, y=col2, data=df_scaled)
df_scaled[col1].corr(df_scaled[col2])
```

```
-0.9347265183592881
```



`rank` and `subscribers` have a strong negative correlation of -0.9347 which indicates that the larger the number of subscribers, the channel is very likely to have a high ranking.

Data Transformation

The plot is not a straight line. Let's plot the logarithm of `rank` vs. `subscribers` to get a better correlation.

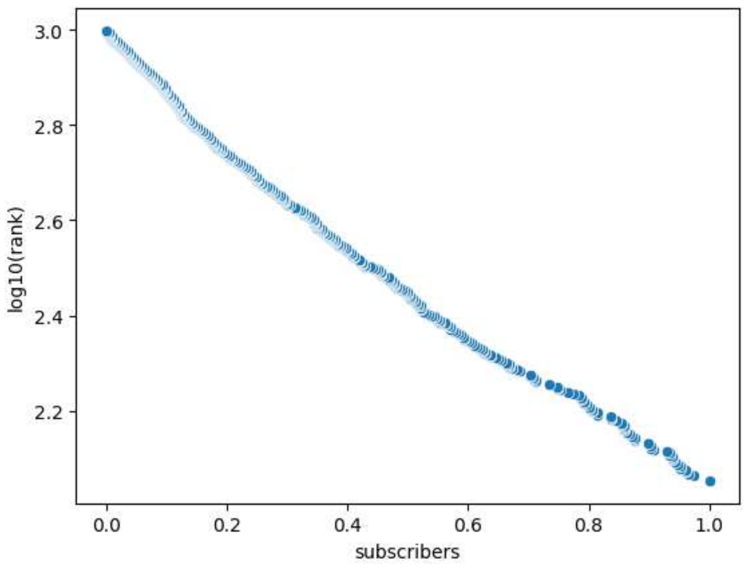
The logarithm of `rank` and `subscribers` have a very strong negative correlation of -0.9943 as shown in the below plot. Thus, as the number of subscribers increases the log of rank is very likely to decrease.

```
col3 = 'log10(rank)'
df_scaled[col3] = np.log10(df_scaled[col2])
df_scaled[col1].corr(df_scaled[col3])
```

```
-0.9943406749095963
```

```
sns.scatterplot(x=col1, y=col3, data=df_scaled)
```

```
<Axes: xlabel='subscribers', ylabel='log10(rank)'>
```


3.0
2.8
2.6
2.4
2.2
log10(rank)
0.0
0.2
0.4
0.6
0.8
1.0
subscribers