Loading Libraries and Dataset

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprec
ated. Use the functions in the public API at pandas.testing instead.
  import pandas.util.testing as tm
```

In [2]:
```
dfx = pd.read_excel('/Movie300 Revised V1.xlsx')
dfx.head()
```

Out[2]:

	Movie_name	Movie_Genre	Movie_Genre_Num	Movie_Certification	Movie_Certification_Num	Release_Date	Release_Month	Release_Month_Nur
0	Kannum Kannum Kollaiyadithaal	Thriller	15	U	2	28 Feb 2020	Feb	
1	Oh My Kadavule	Comedy	14	UA	1	14 Feb 2020	Feb	
2	Psycho	Thriller	15	A	3	24 Jan 2020	Jan	
3	Dharala Prabhu	Comedy	14	UA	1	13 Mar 2020	Mar	
4	Gypsy	Drama	17	UA	1	06 Mar 2020	Mar	

Dataset Cleaning and some preliminary steps

In [3]:
```
dfx['Release_Month'] = dfx['Release_Date'].apply(lambda x: x.split(' ')[1])
dfx['Release_Month'].head()
```

Out[3]:
```
0    Feb
1    Feb
2    Jan
3    Mar
4    Mar
Name: Release_Month, dtype: object
```

In [4]:
```
dfx.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 16 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Movie_name              300 non-null    object 
 1   Movie_Genre             300 non-null    object 
 2   Movie_Genre_Num         300 non-null    int64  
 3   Movie_Certification     300 non-null    object 
 4   Movie_Certification_Num  300 non-null    int64  
 5   Release_Date            300 non-null    object 
 6   Release_Month           300 non-null    object 
 7   Release_Month_Num       300 non-null    int64  
 8   Runtime_Duration        300 non-null    object 
 9   Runtime_Minutes         300 non-null    int64  
 10  Lead_Actor              271 non-null    object 
 11  Lead_Actress            265 non-null    object 
 12  Movie_Critic_Rating     300 non-null    float64
 13  Movie_User_Rating       300 non-null    float64
 14  Movie_Synopsis          300 non-null    object 
 15  Movie_Full_Cast         300 non-null    object 
dtypes: float64(2), int64(4), object(10)
memory usage: 37.6+ KB
```

In [5]:
```
dfx.fillna
```

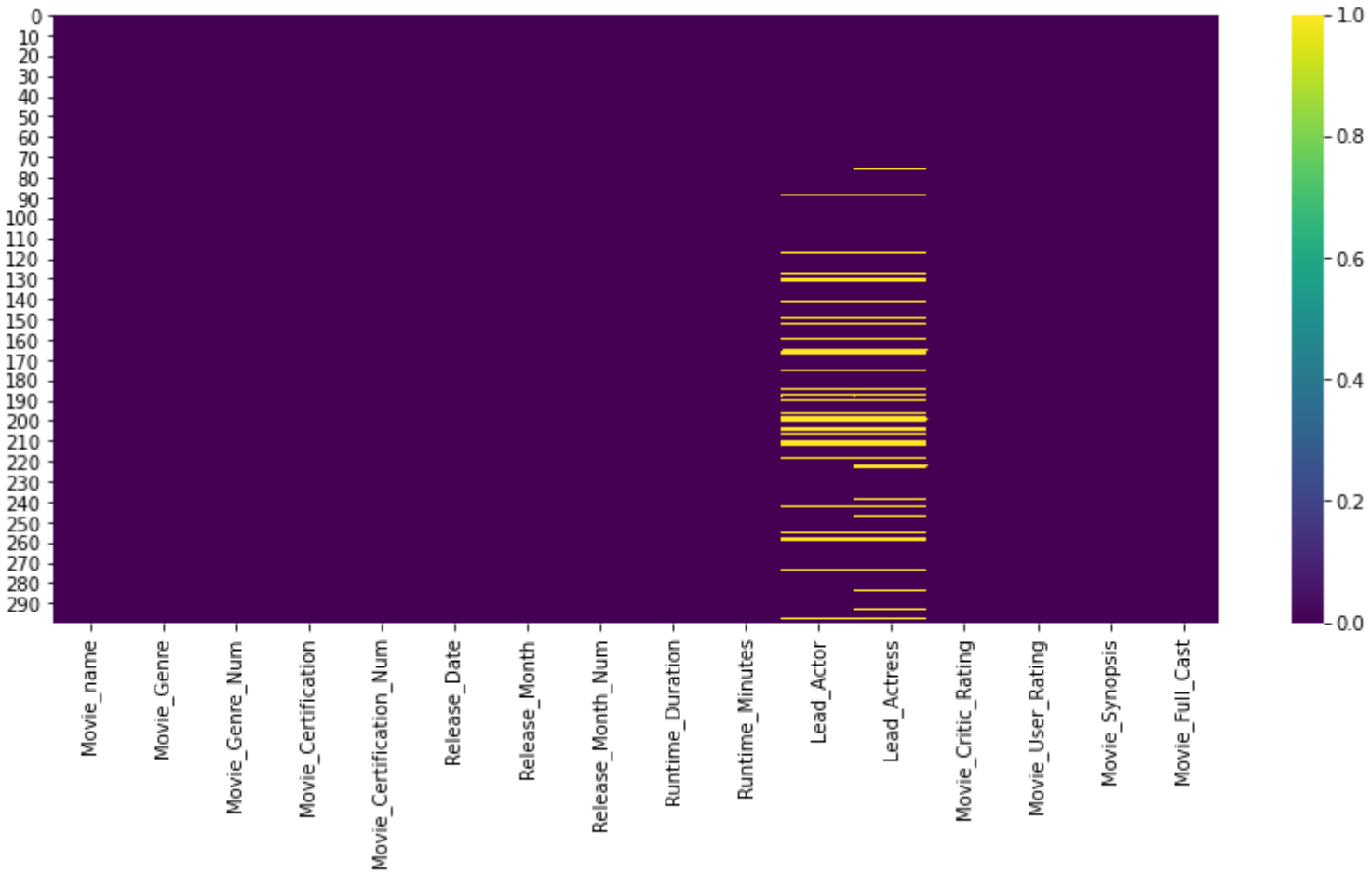
Out[5]:
```
<bound method DataFrame.fillna of                       Movie_name  ...                                  Movie_Fu
ll_Cast
0    Kannum Kannum Kollaiyadithaal  ...  Dulquer Salmaan, Ritu Varma, Gautham Vasudev M...
1                  Oh My Kadavule  ...            Ashok Selvan, Ritika Singh, Vani Bhojan
2                         Psycho  ...  Udhayanidhi Stalin, Aditi Rao Hydari, Nithya M...
3                  Dharala Prabhu  ...                  Harish Kalyan, Tanya Hope, Vivek
4                           Gypsy  ...        Jiiva, Natasha Singh, Lal Jose, Sunny Wayne
..                            ...  ...                                                ...
295                  Nootrenbadhu  ...  Siddharth, Priya Anand, Nithya Menen, Mouli, G...
296                Ponnar Shankar  ...  Prashanth, Divya Parameswaran, Pooja Chopra, S...
297             Nadunisi Naaygal  ...         Veera, Sameera Reddy, Deva, Swapna Abraham
298                     Ilaignan  ...  Pa Vijay, Kushboo, Meera Jasmine, Ramya Nambee...
299                    Mappillai  ...  Dhanush, Hansika Motwani, Manisha Koirala, Viv...

[300 rows x 16 columns]>
```

In [6]:
```
dfx.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 16 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Movie_name               300 non-null    object 
 1   Movie_Genre              300 non-null    object 
 2   Movie_Genre_Num          300 non-null    int64  
 3   Movie_Certification      300 non-null    object 
 4   Movie_Certification_Num  300 non-null    int64  
 5   Release_Date             300 non-null    object 
 6   Release_Month            300 non-null    object 
 7   Release_Month_Num        300 non-null    int64  
 8   Runtime_Duration         300 non-null    object 
 9   Runtime_Minutes          300 non-null    int64  
 10  Lead_Actor               271 non-null    object 
 11  Lead_Actress             265 non-null    object 
 12  Movie_Critic_Rating      300 non-null    float64
 13  Movie_User_Rating        300 non-null    float64
 14  Movie_Synopsis           300 non-null    object 
 15  Movie_Full_Cast          300 non-null    object 
dtypes: float64(2), int64(4), object(10)
memory usage: 37.6+ KB
```

In [7]:
```
plt.figure(figsize=(14,6))
sb.heatmap(dfx.isnull(), cmap="viridis")
```

Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbf6c294eb8>



In [8]:
```
dfx['Lead_Actor'].isnull().sum()
```

Out[8]: 29

In [9]:
```
dfx['Lead_Actress'].isnull().sum()
```

Out[9]:
```
35
```

In [10]:
```
dfy = dfx.copy()
```

In [11]:
```
dfy.head()
```

Out[11]:

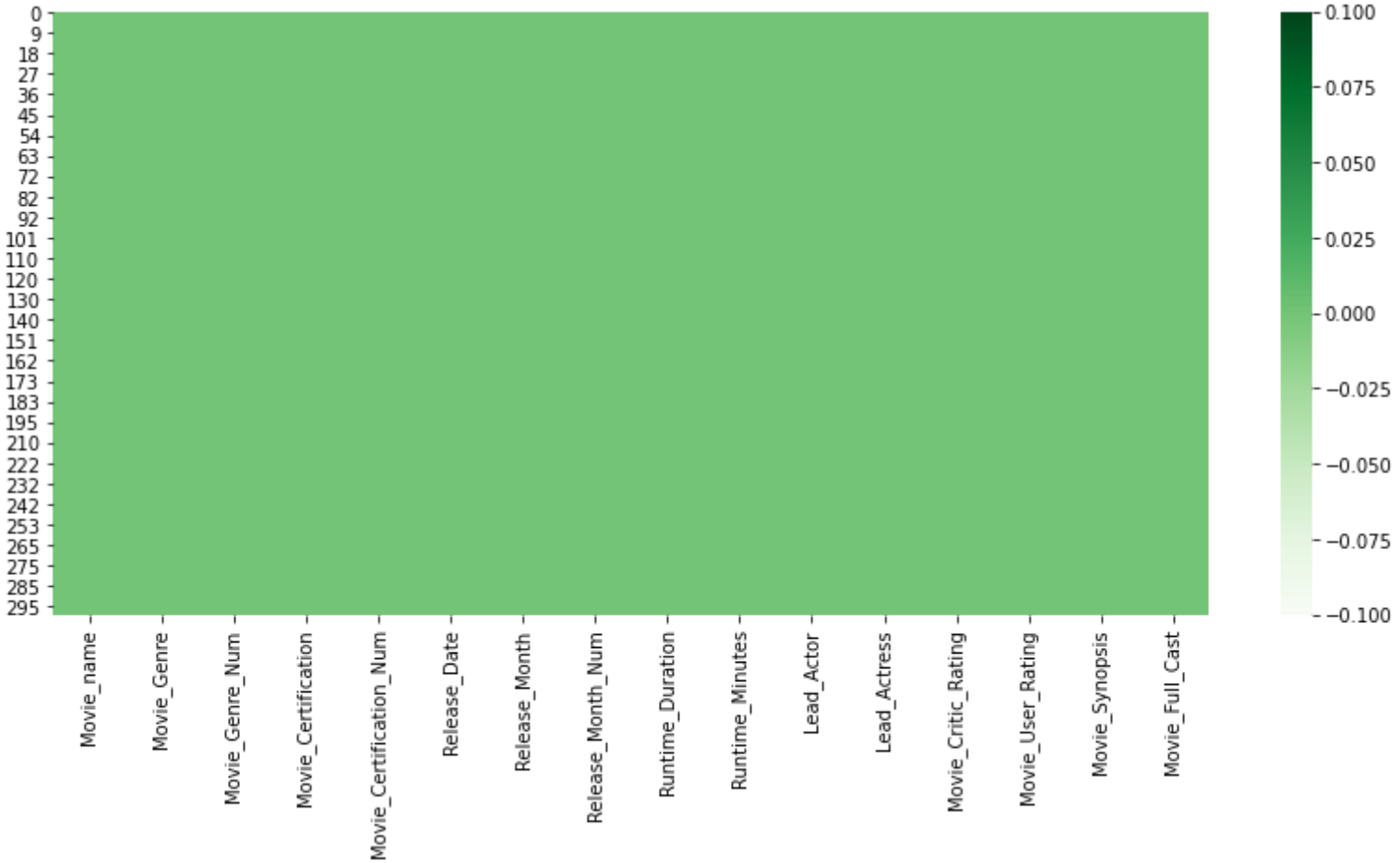
	Movie_name	Movie_Genre	Movie_Genre_Num	Movie_Certification	Movie_Certification_Num	Release_Date	Release_Month	Release_Month_Nur
0	Kannum Kannum Kollaiyadithaal	Thriller	15	U	2	28 Feb 2020	Feb	
1	Oh My Kadavule	Comedy	14	UA	1	14 Feb 2020	Feb	
2	Psycho	Thriller	15	A	3	24 Jan 2020	Jan	
3	Dharala Prabhu	Comedy	14	UA	1	13 Mar 2020	Mar	
4	Gypsy	Drama	17	UA	1	06 Mar 2020	Mar	

In [12]:
```
dfy.dropna(inplace=True)
```

In [13]:

```
plt.figure(figsize=(14,6))
sb.heatmap(dfy.isnull(), cmap="Greens")

#'Accent', 'Accent_r', 'Blues', 'Blues_r', 'BrBG', 'BrBG_r', 'BuGn', 'BuGn_r', 'BuPu', 'BuPu_r', 'CMRmap',
#'CMRmap_r', 'Dark2', 'Dark2_r', 'GnBu', 'GnBu_r', 'Greens', 'Greens_r', 'Greys', 'Greys_r', 'OrRd', 'OrRd_r',
#'Oranges', 'Oranges_r', 'PRGn', 'PRGn_r', 'Paired', 'Paired_r', 'Pastel1', 'Pastel1_r', 'Pastel2', 'Pastel2_r',
#'PiYG', 'PiYG_r', 'PuBu', 'PuBuGn', 'PuBuGn_r', 'PuBu_r', 'PuOr', 'PuOr_r', 'PuRd', 'PuRd_r', 'Purples', 'Purples_r',
#'RdBu', 'RdBu_r', 'RdGy', 'RdGy_r', 'RdPu', 'RdPu_r', 'RdYlBu', 'RdYlBu_r', 'RdYlGn', 'RdYlGn_r', 'Reds', 'Reds_r',
#'Set1', 'Set1_r', 'Set2', 'Set2_r', 'Set3', 'Set3_r', 'Spectral', 'Spectral_r', 'Wistia', 'Wistia_r', 'YlGn', 'YlGnB
u',
#'YlGnBu_r', 'YlGn_r', 'YlOrBr', 'YlOrBr_r', 'YlOrRd', 'YlOrRd_r', 'afmhot', 'afmhot_r', 'autumn', 'autumn_r', 'binar
y',
#'binary_r', 'bone', 'bone_r', 'brg', 'brg_r', 'bwr', 'bwr_r', 'cividis', 'cividis_r', 'cool', 'cool_r', 'coolwarm',
#'coolwarm_r', 'copper', 'copper_r', 'cubehelix', 'cubehelix_r', 'flag', 'flag_r', 'gist_earth', 'gist_earth_r',
#'gist_gray', 'gist_gray_r', 'gist_heat', 'gist_heat_r', 'gist_ncar', 'gist_ncar_r', 'gist_rainbow', 'gist_rainbow_r',
#'gist_stern', 'gist_stern_r', 'gist_yarg', 'gist_yarg_r', 'gnuplot', 'gnuplot2', 'gnuplot2_r', 'gnuplot_r', 'gray',
#'gray_r', 'hot', 'hot_r', 'hsv', 'hsv_r', 'icefire', 'icefire_r', 'inferno', 'inferno_r', 'jet', 'jet_r', 'magma',
#'magma_r', 'mako', 'mako_r', 'n...
```

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbf6868c438>



In [14]:
```
dfy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 265 entries, 0 to 299
Data columns (total 16 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Movie_name              265 non-null    object 
 1   Movie_Genre             265 non-null    object 
 2   Movie_Genre_Num         265 non-null    int64  
 3   Movie_Certification     265 non-null    object 
 4   Movie_Certification_Num  265 non-null    int64  
 5   Release_Date            265 non-null    object 
 6   Release_Month           265 non-null    object 
 7   Release_Month_Num       265 non-null    int64  
 8   Runtime_Duration        265 non-null    object 
 9   Runtime_Minutes         265 non-null    int64  
 10  Lead_Actor              265 non-null    object 
 11  Lead_Actress            265 non-null    object 
 12  Movie_Critic_Rating     265 non-null    float64
 13  Movie_User_Rating       265 non-null    float64
 14  Movie_Synopsis          265 non-null    object 
 15  Movie_Full_Cast         265 non-null    object 
dtypes: float64(2), int64(4), object(10)
memory usage: 35.2+ KB
```

Genre Analysis

In [15]:
```
dfx['Movie_Genre'].unique()
```

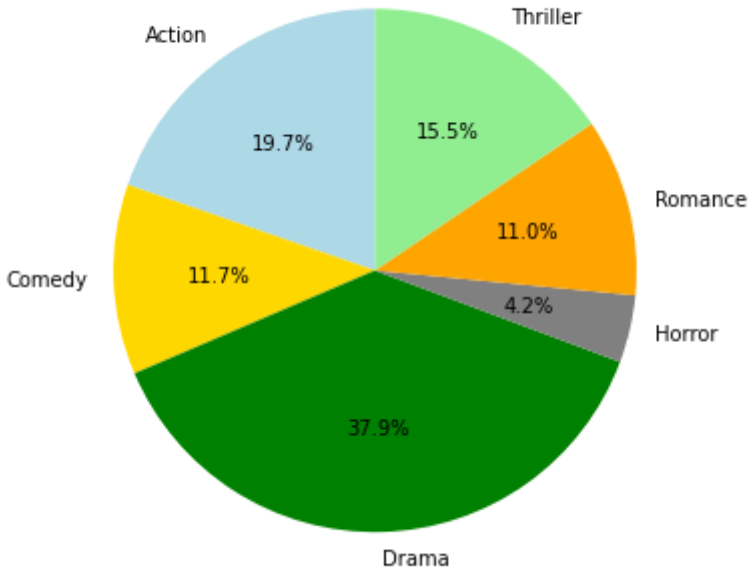
Out[15]:
```
array(['Thriller', 'Comedy', 'Drama', 'Action', 'Family', 'Crime',
       'Adventure', 'Musical', 'Biography', 'Sports', 'Romance', 'Sci-Fi',
       'Mystery', 'Horror', 'Documentary', 'History', 'Fantasy'],
      dtype=object)
```

In [16]:
```
genre = dfx.groupby('Movie_Genre')['Movie_Genre'].count()
genre
```

Out[16]:
```
Movie_Genre
Action          52
Adventure        2
Biography        2
Comedy          31
Crime           10
Documentary      1
Drama          100
Family           2
Fantasy          2
History          3
Horror          11
Musical          2
Mystery          3
Romance         29
Sci-Fi           5
Sports           4
Thriller        41
Name: Movie_Genre, dtype: int64
```

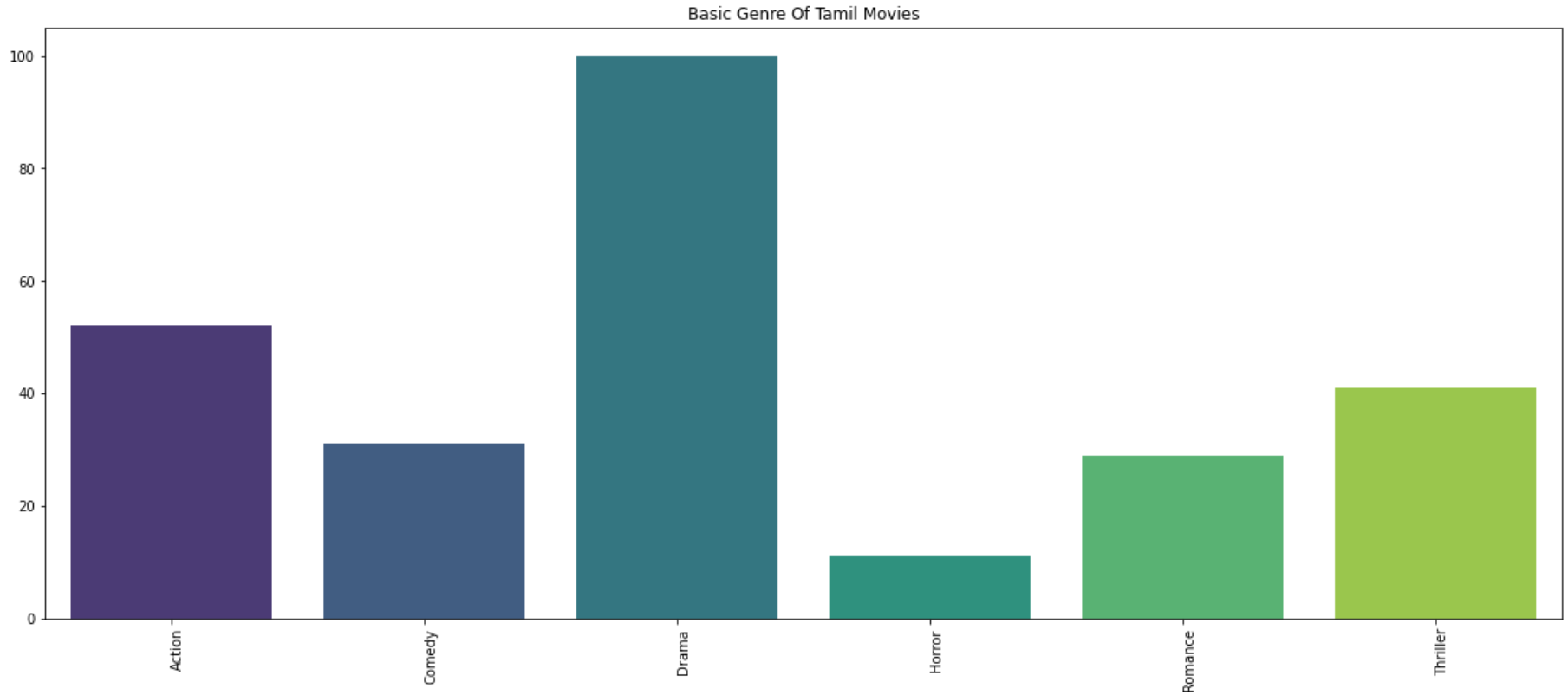
Pie Chart Representation of basic genre

In [17]:
```
genre_general = ['Action', 'Comedy', 'Drama', 'Horror', 'Romance', 'Thriller']
genre_general_values = [52, 31, 100, 11, 29, 41]

#pie chart
colors = ['lightblue', 'gold', 'green','grey', 'orange', 'lightgreen' ]
plt.subplots(figsize=(14,6))
plt.pie(genre_general_values,labels=genre_general, colors=colors, startangle = 90, autopct='%.1f%%')
plt.show()
```



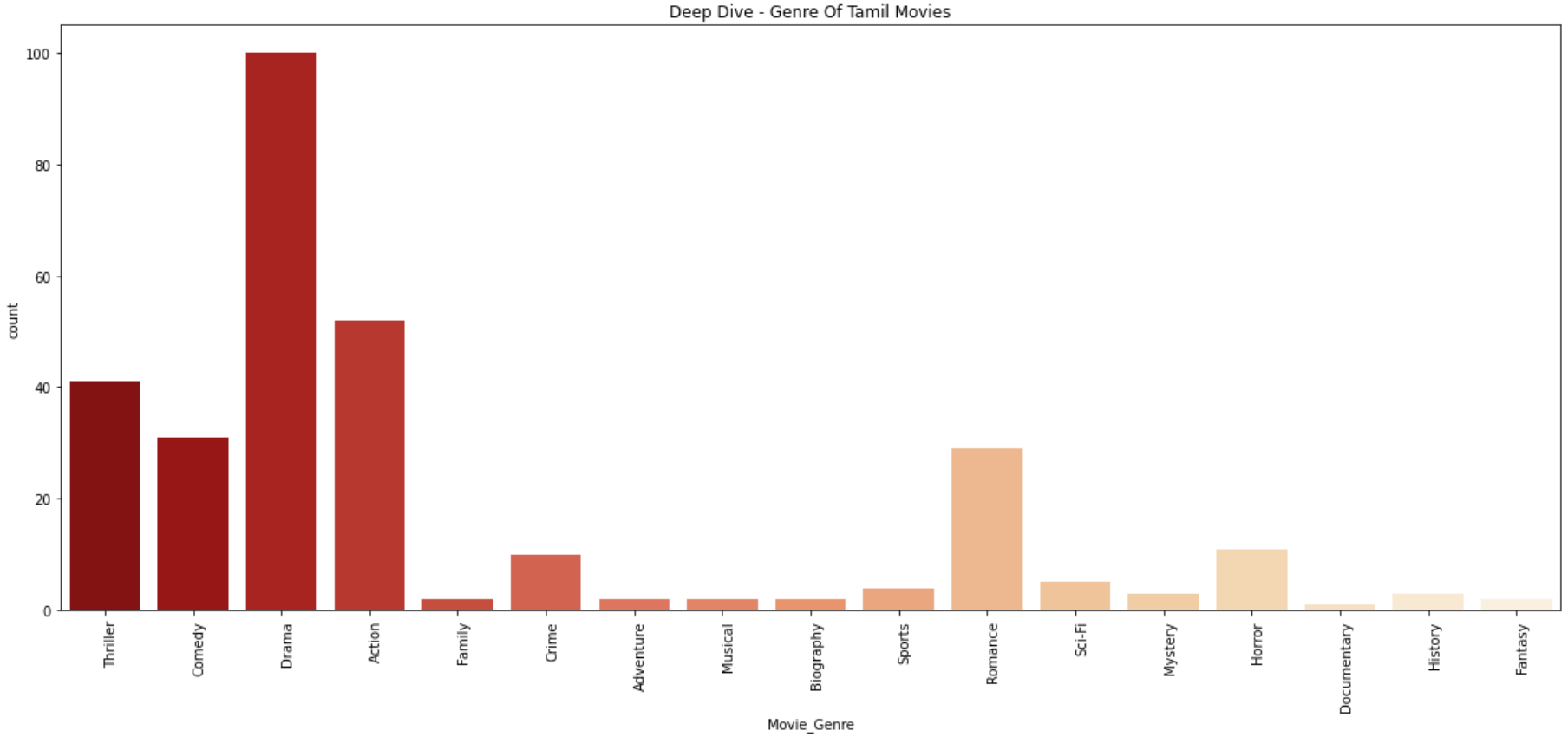
Bar Chart Representation of basic genre

In [18]:
```
plt.figure(figsize=(20,8))
x = genre_general
y = genre_general_values
chart1 = sb.barplot(x, y, palette='viridis')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90, horizontalalignment='left')
chart1.set_title('Basic Genre Of Tamil Movies')
plt.show()
```



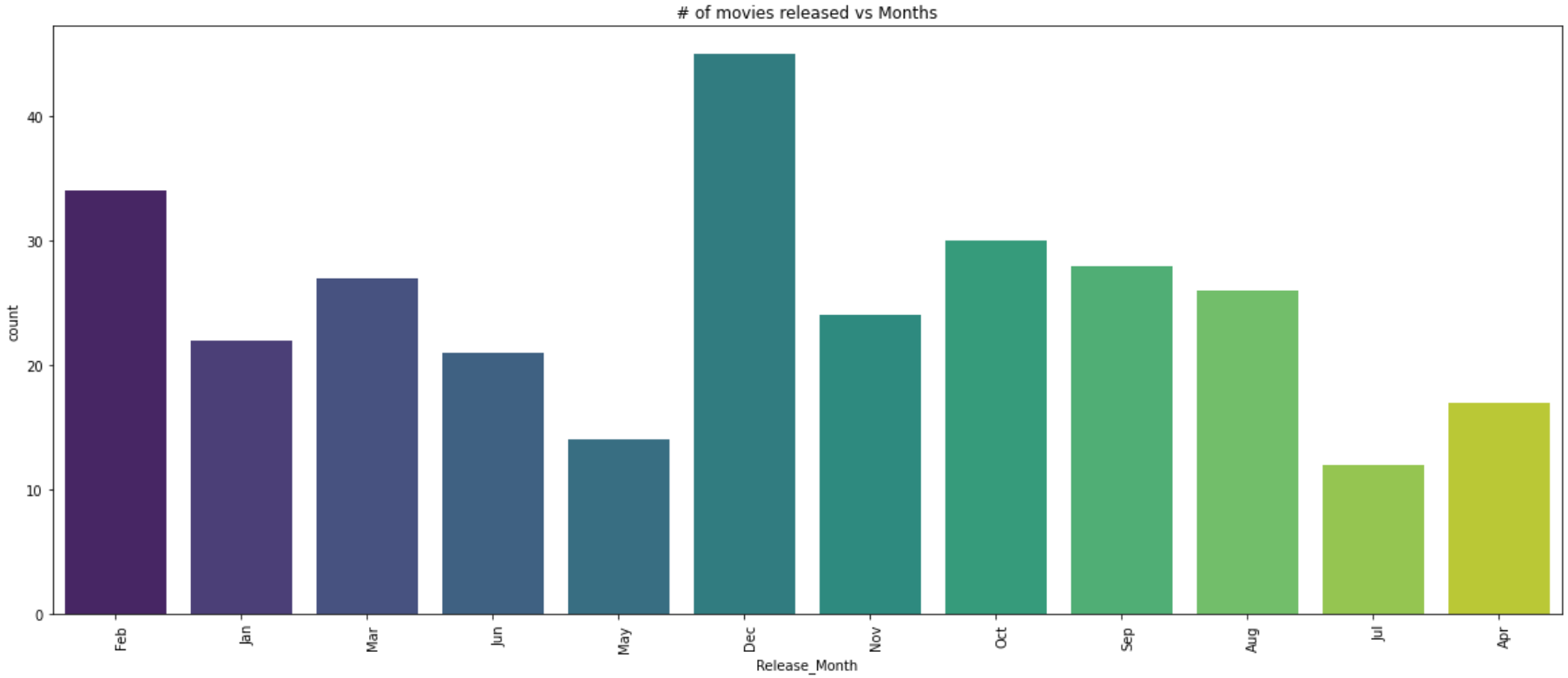
Depth Analysis of Genre

In [19]:
```
plt.figure(figsize=(20,8))
x = dfx['Movie_Genre']
chart1 = sb.countplot(x, data=dfx, palette='OrRd_r')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90, horizontalalignment='left')
chart1.set_title('Deep Dive - Genre Of Tamil Movies')
plt.show()
```



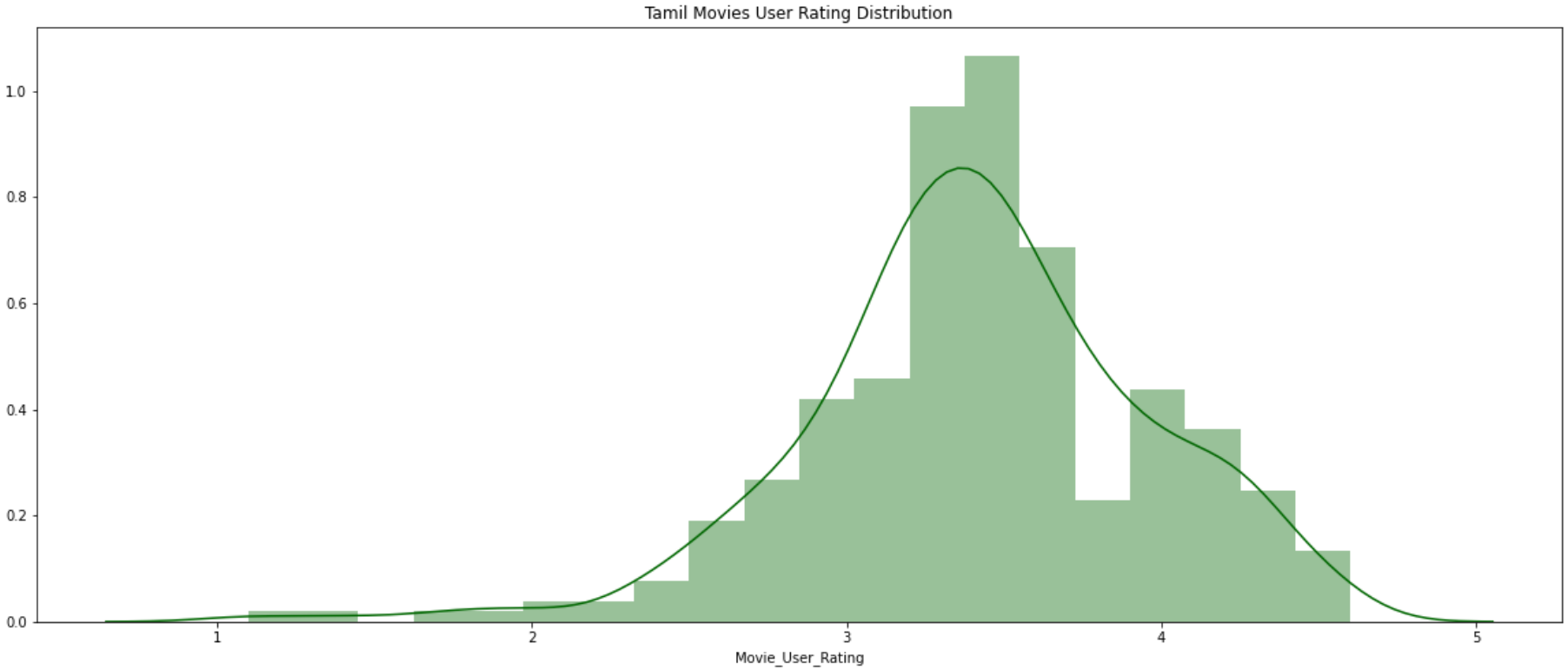
Month-wise Visualization of Movie releases

In [20]:
```
plt.figure(figsize=(20,8))
chart1 = sb.countplot(x=dfx['Release_Month'], data=dfx, palette='viridis')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90, horizontalalignment='left')
chart1.set_title('# of movies released vs Months')
plt.show()
```



User Rating Distribution Plot

In [21]:
```
plt.figure(figsize=(20,8))
chart2 = sb.distplot(dfx['Movie_User_Rating'],  color="#006600")
chart2.set_title('Tamil Movies User Rating Distribution')
plt.show()
```

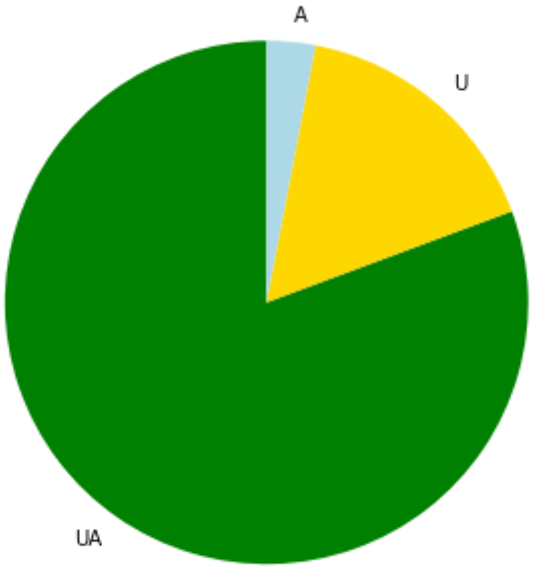


Movie Certifications Analysis

Pie Chart Representation of Movie Censorboard Certification

In [22]:
```
mc = dfx.groupby('Movie_Certification')['Movie_Certification'].count()
mc_entities = ['UA', 'U', 'A']
mc_values = [242,49,9]
```

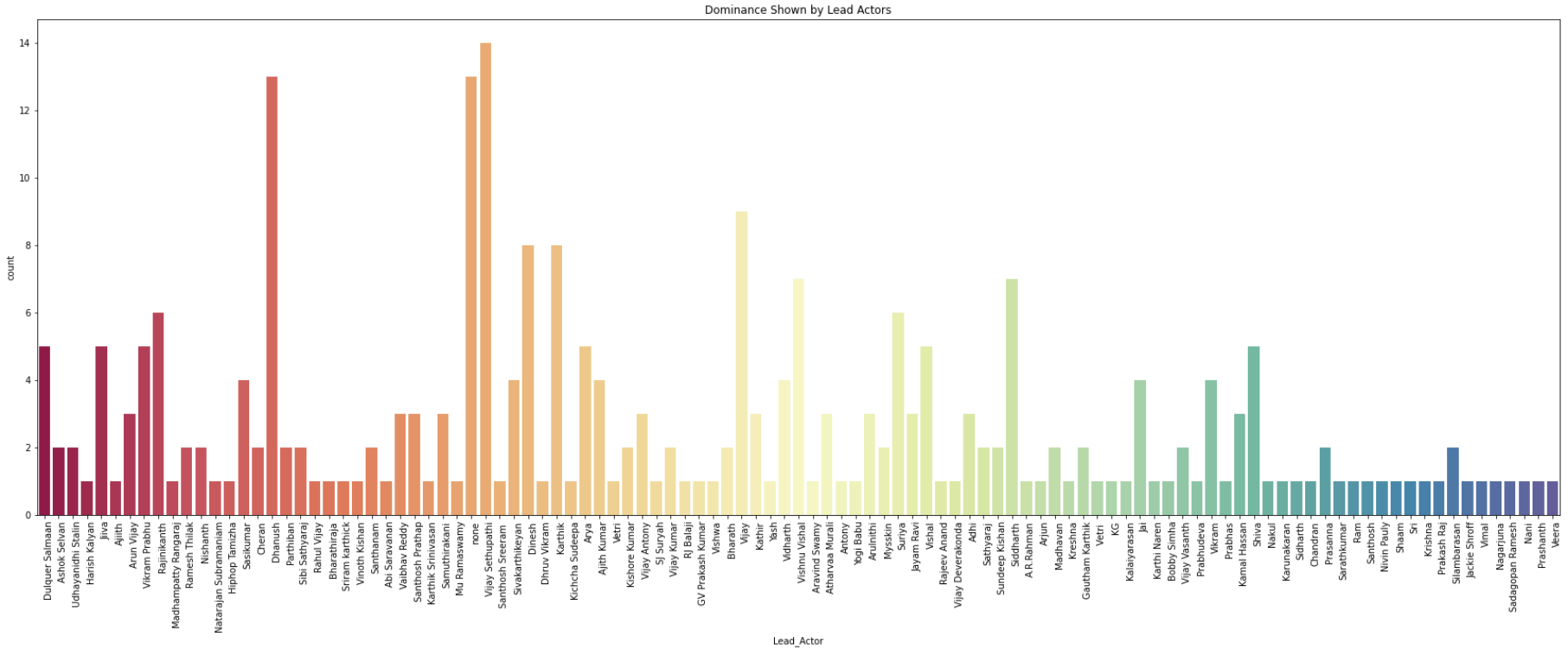
In [23]:
```
plt.subplots(figsize=(14,6))
colors = ['green', 'gold', 'lightblue']
plt.pie(mc_values, labels = mc_entities, colors=colors, startangle = 90)
plt.show()
```



Analysis of Lead Actors and Lead Actresses of the Decade

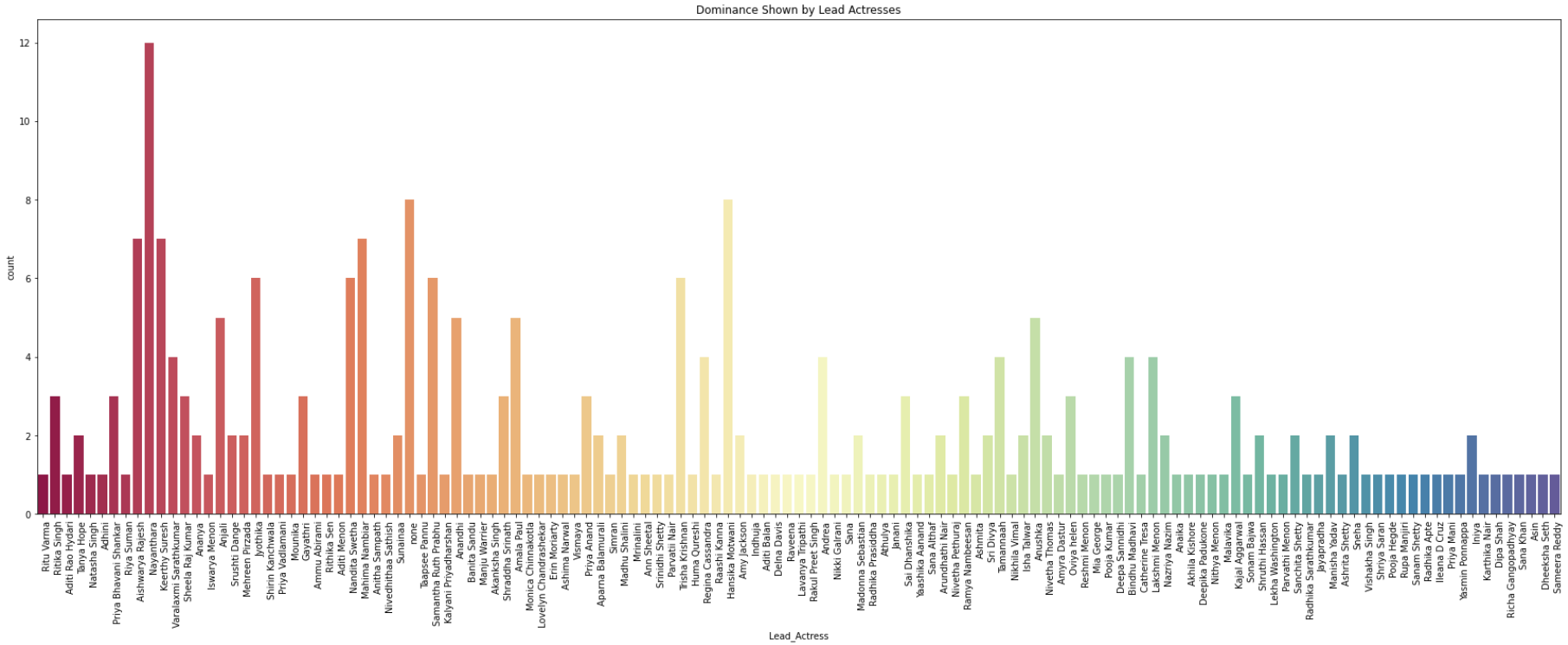
Dominance comparison of Lead Actors

In [24]:
```
plt.figure(figsize=(30,10))
chart1 = sb.countplot(x=dfx['Lead_Actor'], data=dfx, palette='Spectral')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90, horizontalalignment='left')
chart1.set_title('Dominance Shown by Lead Actors')
plt.show()
```



Dominance comparison of Lead Actresses

In [25]:
```
plt.figure(figsize=(30,10))
chart1 = sb.countplot(x=dfx['Lead_Actress'], data=dfx, palette='Spectral')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90, horizontalalignment='left')
chart1.set_title('Dominance Shown by Lead Actresses')
plt.show()
```



Conclusion

Presenting the Rockstars of Kollywood in the decade 2011-2020

Machine Learning Implementation

In [26]:
```
dfx.head(10)
```

Out[26]:

	Movie_name	Movie_Genre	Movie_Genre_Num	Movie_Certification	Movie_Certification_Num	Release_Date	Release_Month	Release_Month_Nur
0	Kannum Kannum Kollaiyadithaal	Thriller	15	U	2	28 Feb 2020	Feb	
1	Oh My Kadavule	Comedy	14	UA	1	14 Feb 2020	Feb	
2	Psycho	Thriller	15	A	3	24 Jan 2020	Jan	
3	Dharala Prabhu	Comedy	14	UA	1	13 Mar 2020	Mar	
4	Gypsy	Drama	17	UA	1	06 Mar 2020	Mar	
5	Baaram	Drama	17	A	3	21 Feb 2020	Feb	
6	Mafia: Chapter 1	Drama	17	UA	1	21 Feb 2020	Feb	
7	Seeru	Action	16	UA	1	07 Feb 2020	Feb	
8	Vaanam Kottattum	Drama	17	U	2	07 Feb 2020	Feb	
9	Darbar	Action	16	UA	1	09 Jan 2020	Jan	

In [27]:
```
'''path = 'C:/Users/gkish/Jupyter Notebooks/BDB/DAY - 3/Movie300 Revised V1.xlsx'
dfx.to_excel(path)'''
```

Out[27]:
```
"path = 'C:/Users/gkish/Jupyter Notebooks/BDB/DAY - 3/Movie300 Revised V1.xlsx'\ndfx.to_excel(path)"
```

In [28]:
```
dfx.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 16 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Movie_name               300 non-null    object 
 1   Movie_Genre              300 non-null    object 
 2   Movie_Genre_Num          300 non-null    int64  
 3   Movie_Certification      300 non-null    object 
 4   Movie_Certification_Num  300 non-null    int64  
 5   Release_Date             300 non-null    object 
 6   Release_Month            300 non-null    object 
 7   Release_Month_Num        300 non-null    int64  
 8   Runtime_Duration         300 non-null    object 
 9   Runtime_Minutes          300 non-null    int64  
 10  Lead_Actor               271 non-null    object 
 11  Lead_Actress             265 non-null    object 
 12  Movie_Critic_Rating      300 non-null    float64
 13  Movie_User_Rating        300 non-null    float64
 14  Movie_Synopsis           300 non-null    object 
 15  Movie_Full_Cast          300 non-null    object 
dtypes: float64(2), int64(4), object(10)
memory usage: 37.6+ KB
```

In [29]:
```
dfx.columns
```

Out[29]:
```
Index(['Movie_name', 'Movie_Genre', 'Movie_Genre_Num', 'Movie_Certification',
       'Movie_Certification_Num', 'Release_Date', 'Release_Month',
       'Release_Month_Num', 'Runtime_Duration', 'Runtime_Minutes',
       'Lead_Actor', 'Lead_Actress', 'Movie_Critic_Rating',
       'Movie_User_Rating', 'Movie_Synopsis', 'Movie_Full_Cast'],
      dtype='object')
```

In [30]:
```
dfxml = dfx[['Movie_name','Movie_Genre_Num','Movie_Certification_Num','Release_Month_Num','Runtime_Minutes','Movie_Critic_Rating','Movie_User_Rating']]
dfxml.head()
```

Out[30]:

	Movie_name	Movie_Genre_Num	Movie_Certification_Num	Release_Month_Num	Runtime_Minutes	Movie_Critic_Rating	Movie_User_Rating
0	Kannum Kannum Kollaiyadithaal	15	2	2	122	3.5	4.3
1	Oh My Kadavule	14	1	2	151	3.5	3.4
2	Psycho	15	3	1	134	3.5	3.3
3	Dharala Prabhu	14	1	3	122	3.0	3.3
4	Gypsy	17	1	3	145	3.0	3.2

In [31]:
```
#target variable
y = dfxml['Movie_User_Rating']

#input dataframe
x = dfxml[['Movie_Genre_Num','Movie_Certification_Num','Release_Month_Num','Runtime_Minutes','Movie_Critic_Rating']]
```

In [32]:
```
from sklearn.model_selection import train_test_split
```

In [33]:
```
from sklearn.linear_model import LinearRegression
```

In [34]:
```
x1,x2,y1,y2 = train_test_split(x,y,test_size = 0.1)
```

In [35]:
```
lr = LinearRegression()
```

In [36]:
```
lr.fit(x1,y1)
```

Out[36]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [37]:
```
lr.coef_
```

Out[37]:
```
array([ 0.00177994,  0.05436386, -0.01117045,  0.00173541,  0.77982082])
```

In [38]:
```
lr.intercept_
```

Out[38]:
```
0.5602708187272603
```

In [39]:
```
pd.DataFrame(lr.coef_, index=x.columns, columns=['myval'])
```

Out[39]:

	myval
Movie_Genre_Num	0.001780
Movie_Certification_Num	0.054364
Release_Month_Num	-0.011170
Runtime_Minutes	0.001735
Movie_Critic_Rating	0.779821

In [40]:
```
t = np.array(dfxml.loc[8][['Movie_Genre_Num','Movie_Certification_Num','Release_Month_Num','Runtime_Minutes','Movie_Critic_Rating']])
```

In [41]:
```
lr.predict([t])
```

Out[41]:
```
array([3.25586581])
```

In [43]:
```
f = []
k = []
for i in range(0,300):
    b = np.array(dfxml.loc[i][['Movie_Genre_Num','Movie_Certification_Num','Release_Month_Num','Runtime_Minutes','Movie_Critic_Rating']])
    f.append(lr.predict([b]))
    k.append(np.array(dfxml.loc[i][['Movie_name']]))
```

In [44]:
```
K = pd.DataFrame(k,columns=['Movie_name'])
K
```

Out[44]:

	Movie_name
0	Kannum Kannum Kollaiyadithaal
1	Oh My Kadavule
2	Psycho
3	Dharala Prabhu
4	Gypsy
...	...
295	Nootrenbadhu
296	Ponnar Shankar
297	Nadunisi Naaygal
298	Ilaignan
299	Mappillai

300 rows × 1 columns

In [45]:
```
F=pd.DataFrame(f,columns=['Machine_Predicted_Rating'])
F
```

Out[45]:

	Machine_Predicted_Rating
0	3.614450
1	3.608633
2	3.700809
3	3.157225
4	3.202479
...	...
295	2.726817
296	2.827430
297	1.979619
298	2.114093
299	1.657698

300 rows × 1 columns

In [46]:
```
J = dfxml[['Movie_Critic_Rating','Movie_User_Rating']]
```

In [47]:
```
final = pd.concat([K,J,F],axis=1)
final
```

Out[47]:

	Movie_name	Movie_Critic_Rating	Movie_User_Rating	Machine_Predicted_Rating
0	Kannum Kannum Kollaiyadithaal	3.5	4.3	3.614450
1	Oh My Kadavule	3.5	3.4	3.608633
2	Psycho	3.5	3.3	3.700809
3	Dharala Prabhu	3.0	3.3	3.157225
4	Gypsy	3.0	3.2	3.202479
...	...	...	...	...
295	Nootrenbadhu	2.5	2.4	2.726817
296	Ponnar Shankar	2.5	2.4	2.827430
297	Nadunisi Naaygal	1.5	1.7	1.979619
298	Ilaignan	1.5	1.4	2.114093
299	Mappillai	1.0	1.1	1.657698

300 rows × 4 columns

In [48]:
```
'''path = 'C:/Users/gkish/Jupyter Notebooks/BDB/DAY - 3/Machine_Predictions.xlsx'
final.to_excel(path)'''
```

Out[48]:
```
"path = 'C:/Users/gkish/Jupyter Notebooks/BDB/DAY - 3/Machine_Predictions.xlsx'\nfinal.to_excel(path)"
```

In [49]:
```
dfx = pd.concat([dfx,F],axis=1)
dfx.columns
```

Out[49]:
```
Index(['Movie_name', 'Movie_Genre', 'Movie_Genre_Num', 'Movie_Certification',
       'Movie_Certification_Num', 'Release_Date', 'Release_Month',
       'Release_Month_Num', 'Runtime_Duration', 'Runtime_Minutes',
       'Lead_Actor', 'Lead_Actress', 'Movie_Critic_Rating',
       'Movie_User_Rating', 'Movie_Synopsis', 'Movie_Full_Cast',
       'Machine_Predicted_Rating'],
      dtype='object')
```

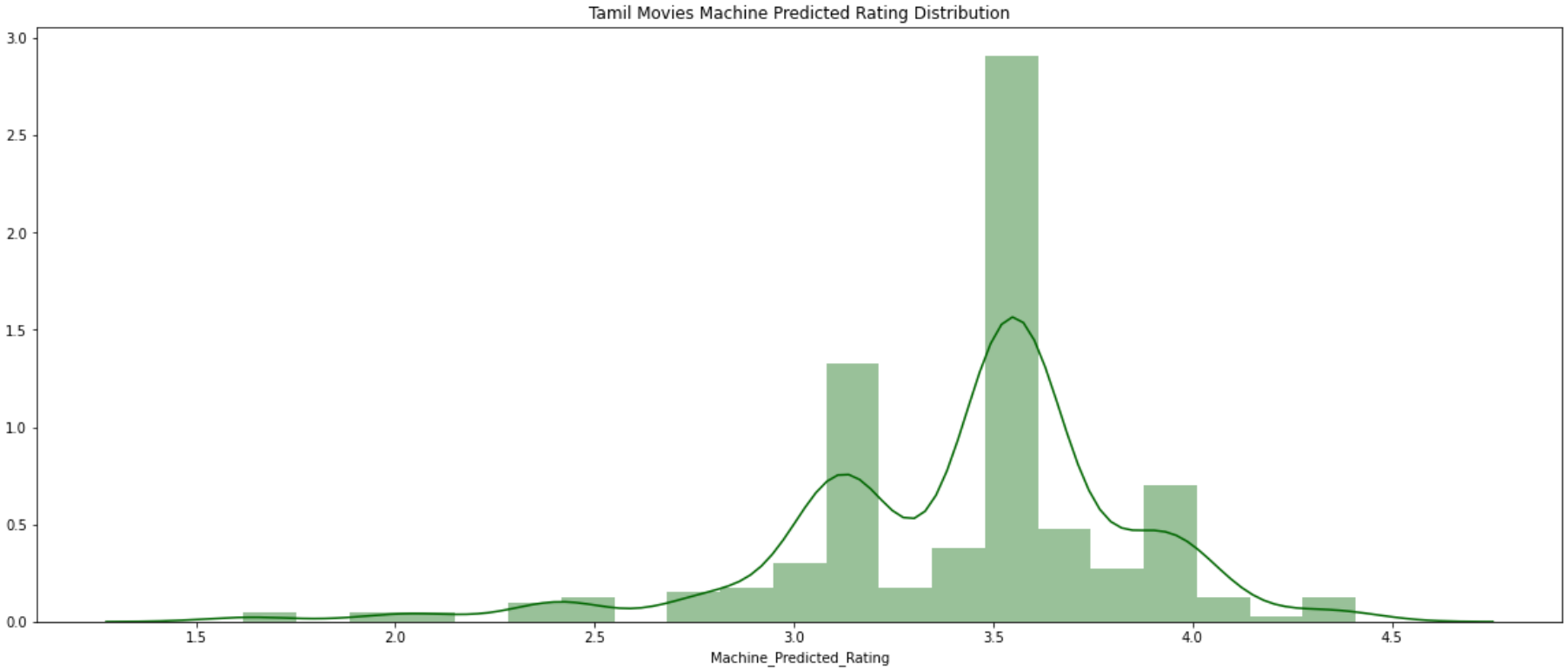
In [50]:
```
dfx = dfx[['Movie_name', 'Movie_Genre', 'Movie_Genre_Num', 'Movie_Certification',
       'Movie_Certification_Num', 'Release_Date', 'Release_Month',
       'Release_Month_Num', 'Runtime_Duration', 'Runtime_Minutes',
       'Lead_Actor', 'Lead_Actress', 'Movie_Critic_Rating',
       'Movie_User_Rating','Machine_Predicted_Rating', 'Movie_Synopsis', 'Movie_Full_Cast']]

dfx.head()
```

Out[50]:

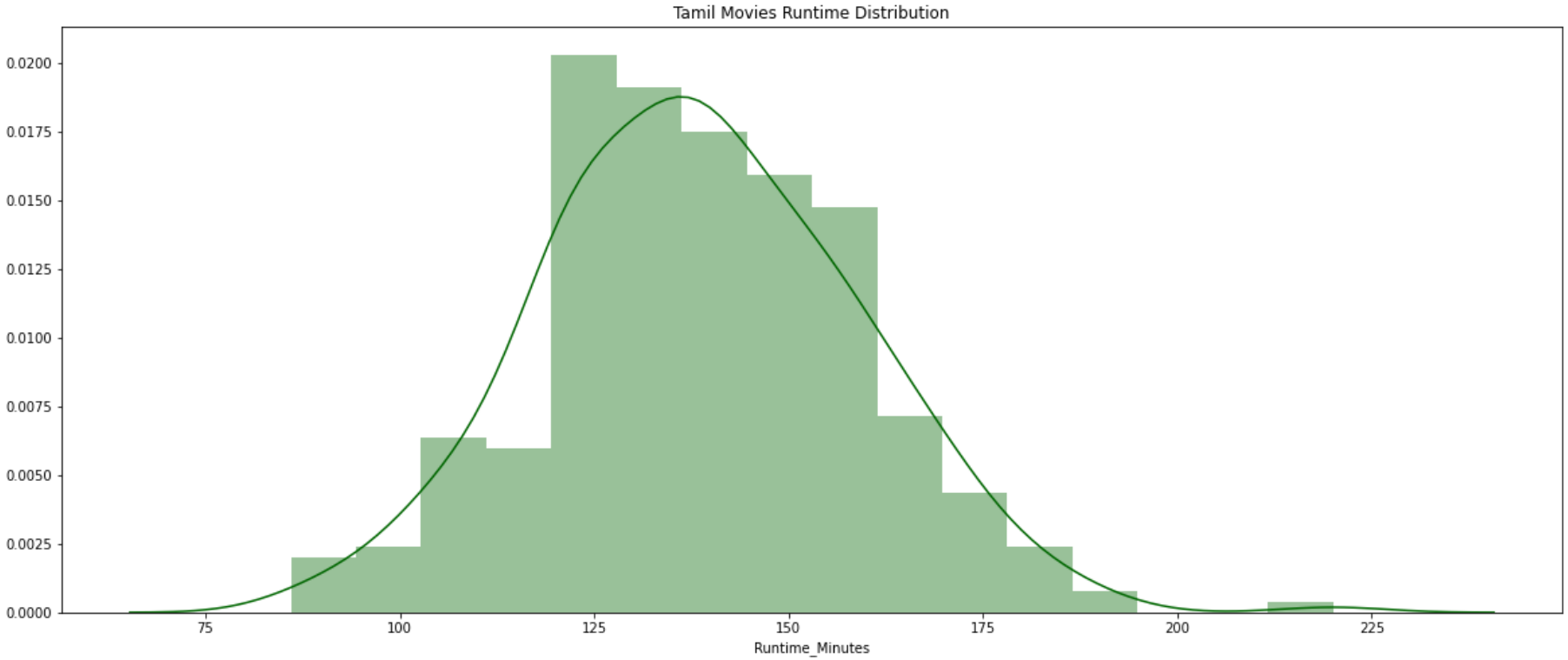
	Movie_name	Movie_Genre	Movie_Genre_Num	Movie_Certification	Movie_Certification_Num	Release_Date	Release_Month	Release_Month_Nur
0	Kannum Kannum Kollaiyadithaal	Thriller	15	U	2	28 Feb 2020	Feb	
1	Oh My Kadavule	Comedy	14	UA	1	14 Feb 2020	Feb	
2	Psycho	Thriller	15	A	3	24 Jan 2020	Jan	
3	Dharala Prabhu	Comedy	14	UA	1	13 Mar 2020	Mar	
4	Gypsy	Drama	17	UA	1	06 Mar 2020	Mar	

Advanced Visualizations

In [51]:
```
plt.figure(figsize=(20,8))
chart3 = sb.distplot(final['Machine_Predicted_Rating'],  color="#006600")
chart3.set_title('Tamil Movies Machine Predicted Rating Distribution')
plt.show()
```



In [52]:
```
plt.figure(figsize=(20,8))
chart4 = sb.distplot(dfx['Runtime_Minutes'],  color="#006600")
chart4.set_title('Tamil Movies Runtime Distribution')
plt.show()
```

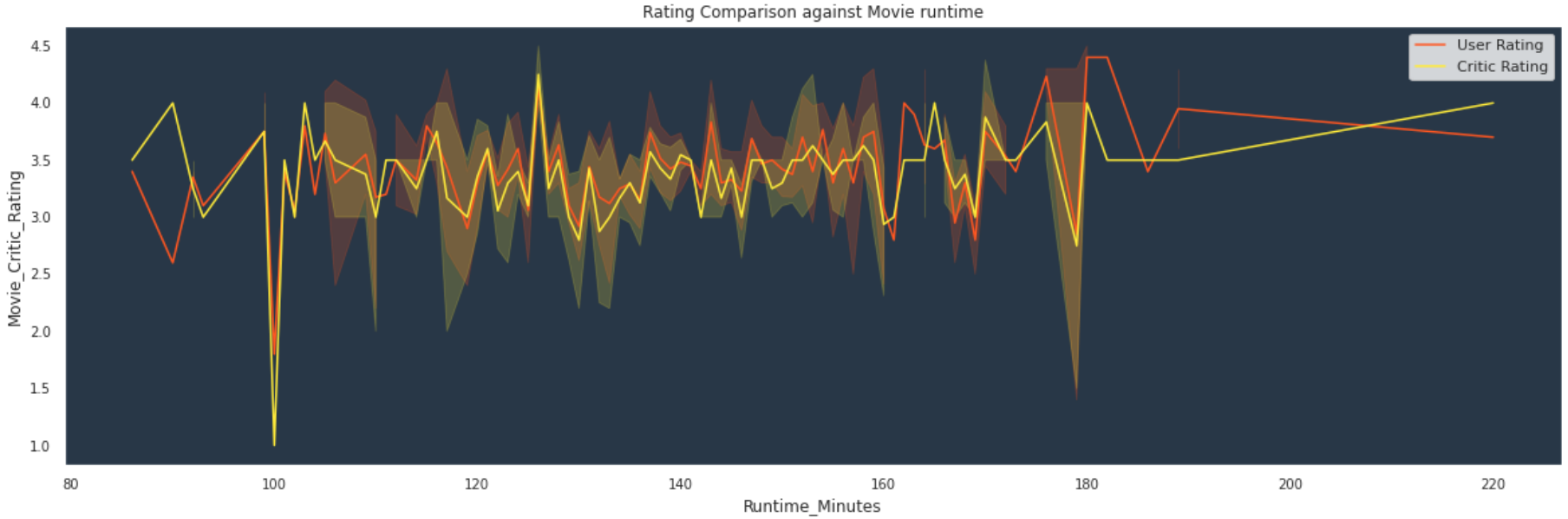


In [53]:
```
dfx['Runtime_Minutes'].mean()
```

Out[53]:
```
138.54666666666665
```

In [54]:
```
import matplotlib as mpl
```

In [55]:
```
plt.figure(figsize=(20,6))
sb.set(rc={"axes.facecolor":"#283747", "axes.grid":False,'xtick.labelsize':10,'ytick.labelsize':10})
chart5 = sb.lineplot(x=dfx.Runtime_Minutes,y=dfx.Movie_User_Rating,data=dfx, color="#FF5722", label='User Rating')
chart5 = sb.lineplot(x=dfx.Runtime_Minutes,y=dfx.Movie_Critic_Rating,data=dfx, color="#FFEB3B", label='Critic Rating')
chart5.set_title('Rating Comparison against Movie runtime')
legend = plt.legend()
frame = legend.get_frame()
frame.set_facecolor('white')
plt.show()
```



In [56]:
```
plt.figure(figsize=(20,6))
chart6 = sb.lineplot(x=dfx.Movie_Genre,y=dfx.Movie_User_Rating, data=dfx, color = "#FF5722", linewidth=2, label='User
 Rating')
chart6 = sb.lineplot(x=dfx.Movie_Genre,y=dfx.Machine_Predicted_Rating, data=dfx, color = "#FFEB3B", linewidth=2, label
='Machine Rating')
chart6.set_title('User Rating against Machine Predicted Rating based on genre')
legend = plt.legend()
frame = legend.get_frame()
frame.set_facecolor('white')
plt.show()
```