In [1]:
```
import pandas as pd
```

In [2]:
```
df=pd.read_csv('C:/Users/STUDENT/Documents/dsp project\Video_Games_Sales_as_at_22_Dec_2016.csv')
dfa=df.copy()
df
```

Out[2]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Criti
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24	
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	
...	...	...	...	...	...	...	...	...	...	...	
16714	Samurai Warriors: Sanada Maru	PS3	2016.0	Action	Tecmo Koei	0.00	0.00	0.01	0.00	0.01	
16715	LMA Manager 2007	X360	2006.0	Sports	Codemasters	0.00	0.01	0.00	0.00	0.01	
16716	Haitaka no Psychedelica	PSV	2016.0	Adventure	Idea Factory	0.00	0.00	0.01	0.00	0.01	
16717	Spirits & Spells	GBA	2003.0	Platform	Wanadoo	0.01	0.00	0.00	0.00	0.01	
16718	Winning Post 8 2016	PSV	2016.0	Simulation	Tecmo Koei	0.00	0.00	0.01	0.00	0.01	

16719 rows × 16 columns

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16719 entries, 0 to 16718
Data columns (total 16 columns):
Name               16717 non-null object
Platform           16719 non-null object
Year_of_Release    16450 non-null float64
Genre              16717 non-null object
Publisher          16665 non-null object
NA_Sales           16719 non-null float64
EU_Sales           16719 non-null float64
JP_Sales           16719 non-null float64
Other_Sales        16719 non-null float64
Global_Sales       16719 non-null float64
Critic_Score       8137 non-null float64
Critic_Count       8137 non-null float64
User_Score         10015 non-null object
User_Count         7590 non-null float64
Developer          10096 non-null object
Rating             9950 non-null object
dtypes: float64(9), object(7)
memory usage: 2.0+ MB
```

In [4]:
```
df.isnull().sum()
```

Out[4]:
```
Name                  2
Platform              0
Year_of_Release     269
Genre                 2
Publisher            54
NA_Sales              0
EU_Sales              0
JP_Sales              0
Other_Sales           0
Global_Sales          0
Critic_Score       8582
Critic_Count       8582
User_Score         6704
User_Count         9129
Developer          6623
Rating             6769
dtype: int64
```

In [5]:
```
df.describe()
```

Out[5]:

	Year_of_Release	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Count
count	16450.000000	16719.000000	16719.000000	16719.000000	16719.000000	16719.000000	8137.000000	8137.000000	7590.000000
mean	2006.487356	0.263330	0.145025	0.077602	0.047332	0.533543	68.967679	26.360821	162.229908
std	5.878995	0.813514	0.503283	0.308818	0.186710	1.547935	13.938165	18.980495	561.282326
min	1980.000000	0.000000	0.000000	0.000000	0.000000	0.010000	13.000000	3.000000	4.000000
25%	2003.000000	0.000000	0.000000	0.000000	0.000000	0.060000	60.000000	12.000000	10.000000
50%	2007.000000	0.080000	0.020000	0.000000	0.010000	0.170000	71.000000	21.000000	24.000000
75%	2010.000000	0.240000	0.110000	0.040000	0.030000	0.470000	79.000000	36.000000	81.000000
max	2020.000000	41.360000	28.960000	10.220000	10.570000	82.530000	98.000000	113.000000	10665.000000

In [6]:
```
dfa = dfa[['Name','Platform','Genre','Publisher','Year_of_Release','Critic_Score','Global_Sales']]
dfa = dfa.dropna().reset_index(drop=True)
dfa
```

Out[6]:

	Name	Platform	Genre	Publisher	Year_of_Release	Critic_Score	Global_Sales
0	Wii Sports	Wii	Sports	Nintendo	2006.0	76.0	82.53
1	Mario Kart Wii	Wii	Racing	Nintendo	2008.0	82.0	35.52
2	Wii Sports Resort	Wii	Sports	Nintendo	2009.0	80.0	32.77
3	New Super Mario Bros.	DS	Platform	Nintendo	2006.0	89.0	29.80
4	Wii Play	Wii	Misc	Nintendo	2006.0	58.0	28.92
...	...	...	...	...	...	...	...
7977	Breach	PC	Shooter	Destineer	2011.0	61.0	0.01
7978	Bust-A-Move 3000	GC	Puzzle	Ubisoft	2003.0	53.0	0.01
7979	Mega Brain Boost	DS	Puzzle	Majesco Entertainment	2008.0	48.0	0.01
7980	STORM: Frontline Nation	PC	Strategy	Unknown	2011.0	60.0	0.01
7981	15 Days	PC	Adventure	DTP Entertainment	2009.0	63.0	0.01

7982 rows × 7 columns

In [7]:
```
s=dfa.drop("Name",axis=1)
s.rename(columns={'Global_Sales':'Hit'},inplace='True')
s.loc[s['Hit']>=1,"Hit"]=int(1)
s.loc[s['Hit']<1,"Hit"]=int(0)
s=s.astype({'Hit':int})
s
```

Out[7]:

	Platform	Genre	Publisher	Year_of_Release	Critic_Score	Hit
0	Wii	Sports	Nintendo	2006.0	76.0	1
1	Wii	Racing	Nintendo	2008.0	82.0	1
2	Wii	Sports	Nintendo	2009.0	80.0	1
3	DS	Platform	Nintendo	2006.0	89.0	1
4	Wii	Misc	Nintendo	2006.0	58.0	1
...	...	...	...	...	...	...
7977	PC	Shooter	Destineer	2011.0	61.0	0
7978	GC	Puzzle	Ubisoft	2003.0	53.0	0
7979	DS	Puzzle	Majesco Entertainment	2008.0	48.0	0
7980	PC	Strategy	Unknown	2011.0	60.0	0
7981	PC	Adventure	DTP Entertainment	2009.0	63.0	0

7982 rows × 6 columns

In [8]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from pandas import get_dummies
df_copy = pd.get_dummies(s)
df_copy
```

Out[8]:

	Year_of_Release	Critic_Score	Hit	Platform_3DS	Platform_DC	Platform_DS	Platform_GBA	Platform_GC	Platform_PC	Platform_PS
0	2006.0	76.0	1	0	0	0	0	0	0	(
1	2008.0	82.0	1	0	0	0	0	0	0	(
2	2009.0	80.0	1	0	0	0	0	0	0	(
3	2006.0	89.0	1	0	0	1	0	0	0	(
4	2006.0	58.0	1	0	0	0	0	0	0	(
...	...	...	...	...	...	...	...	...	...	..
7977	2011.0	61.0	0	0	0	0	0	0	1	(
7978	2003.0	53.0	0	0	0	0	0	1	0	(
7979	2008.0	48.0	0	0	0	1	0	0	0	(
7980	2011.0	60.0	0	0	0	0	0	0	1	(
7981	2009.0	63.0	0	0	0	0	0	0	1	(

7982 rows × 334 columns

In [9]:
```
import seaborn as sns
from math import ceil
import matplotlib.pyplot as plt
n=ceil(0.05 * len(s['Hit']))
sns.regplot(x="Critic_Score", y="Hit", data=s.sample(n=n),logistic=True)
plt.grid()
```

In [10]:
```
dfb = df_copy
y = dfb['Hit'].values
dfb = dfb.drop(['Hit'],axis=1)
X = dfb.values
```

In [11]:
```
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size=0.3, random_state=2)
```

In [12]:
```
from sklearn.ensemble import RandomForestClassifier
model=RandomForestClassifier().fit(Xtrain,ytrain)
y_pred=model.predict(Xtest)
print(classification_report(ytest,y_pred))
```

```
C:\Users\STUDENT\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default value
of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)

              precision    recall  f1-score   support

           0       0.89      0.95      0.92      1997
           1       0.61      0.41      0.49       398

    accuracy                           0.86      2395
   macro avg       0.75      0.68      0.70      2395
weighted avg       0.84      0.86      0.85      2395
```

In [13]:
```
model = LogisticRegression().fit(Xtrain, ytrain)
y_pred = model.predict(Xtest)
print(classification_report(ytest, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.89      0.97      0.93      1997
           1       0.71      0.40      0.52       398

    accuracy                           0.87      2395
   macro avg       0.80      0.69      0.72      2395
weighted avg       0.86      0.87      0.86      2395


C:\Users\STUDENT\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solv
er will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

In [14]:
```
import re
l=[]
m=["Platform","Genre","Publisher","Year of release","Critic score"]
for i in range(5):
    print("Enter",m[i],":",end="")
    u=input()
    l.append(u)
if(re.match('[a-zA-Z\s]+$', l[0])) and (re.match('[a-zA-Z\s]+$', l[1])) and (re.match('[a-zA-Z\s]+$', l[2])) and
    x1=pd.get_dummies(pd.DataFrame({'Platform':[l[0]],'Genre':[l[1]],'Publisher':[l[2]],'Year_of_Release':[l[3]]
    y1=(model.predict(x1.reindex(columns=dfb.columns,fill_value=0)))
    if(y1):
        print('\033[1m********Hit********')
    else:
        print('\033[1m********Not Hit********')
else:
    print("Wrong input")
```

```
Enter Platform :PC
Enter Genre :Sports
Enter Publisher :DTP Entertainments
Enter Year of release :2009
Enter Critic score :76
********Not Hit********
```

In [15]:
```
hit_values=s[s["Hit"]==1]
```

In [16]:
```
hit_values
```

Out[16]:

	Platform	Genre	Publisher	Year_of_Release	Critic_Score	Hit
0	Wii	Sports	Nintendo	2006.0	76.0	1
1	Wii	Racing	Nintendo	2008.0	82.0	1
2	Wii	Sports	Nintendo	2009.0	80.0	1
3	DS	Platform	Nintendo	2006.0	89.0	1
4	Wii	Misc	Nintendo	2006.0	58.0	1
...	...	...	...	...	...	...
1337	PSP	Action	Sony Computer Entertainment	2010.0	86.0	1
1338	XB	Fighting	Namco Bandai Games	2003.0	92.0	1
1339	PS	Sports	Sony Computer Entertainment	2000.0	73.0	1
1340	PS2	Action	Tecmo Koei	2002.0	72.0	1
1341	PS3	Platform	Ubisoft	2011.0	87.0	1

1342 rows × 6 columns

In [17]:
```
k=hit_values["Critic_Score"].min()
```

In [18]:
```
l=hit_values["Critic_Score"].max()
```

In [19]:
```
print("Range:(",k,",",l,")")
```

```
Range:( 20.0 , 98.0 )
```