Linear Regression

Analysing

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import math
import sklearn
%matplotlib inline

unicorn_data = pd.read_csv('./data/Unicorn_Companies.csv')
unicorn_data.head(5)
```

Out[1]:

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	F
0	Bytedance	140.0	04/07/2017	China	Beijing	Artificial intelligence	Sequoia Capital China, SIG Asia Investments, S...	2012	$
1	SpaceX	100.3	12/01/2012	United States	Hawthorne	Other	Founders Fund, Draper Fisher Jurvetson, Rothen...	2002	$(
2	Stripe	95.0	1/23/2014	United States	San Francisco	Fintech	Khosla Ventures, LowercaseCapital, capitalG	2010	$2
3	Klarna	45.6	12/12/2011	Sweden	Stockholm	Fintech	Institutional Venture Partners, Sequoia Capita...	2005	$3
4	Epic Games	42.0	10/26/2018	United States	Cary	Other	Tencent Holdings, KKR, Smash Ventures	1991	$4

In [2]:
```
unicorn_data.tail(5)
```

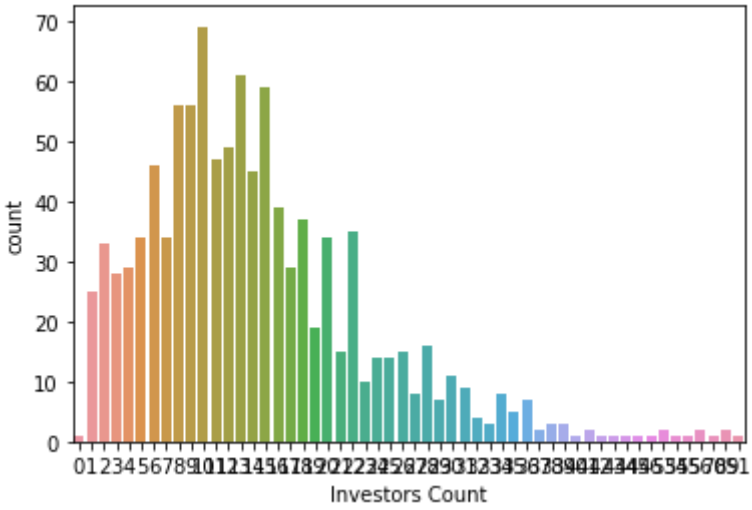
Out[2]:

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year
1032	Timescale	1.0	2/22/2022	United States	New York	Internet software & services	New Enterprise Associates, Benchmark, Two Sigm...	2015
1033	Scalapay	1.0	2/23/2022	Italy	Milan	Fintech	Fasanara Capital, Tiger Global Management, Bal...	2019

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	
1034	Omada Health	1.0	2/23/2022	United States	San Francisco	Health	U.S. Venture Partners, dRx Capital, Andreessen...	2011	$
1035	BlueVoyant	1.0	2/23/2022	United States	New York	Cybersecurity	8VC, Liberty Strategic Capital, Eden Global Pa...	2017	
1036	Veev	1.0	2/24/2022	United States	San Mateo	Internet software & services	Zeev Ventures, Bond, Fifth Wall Ventures	2008	

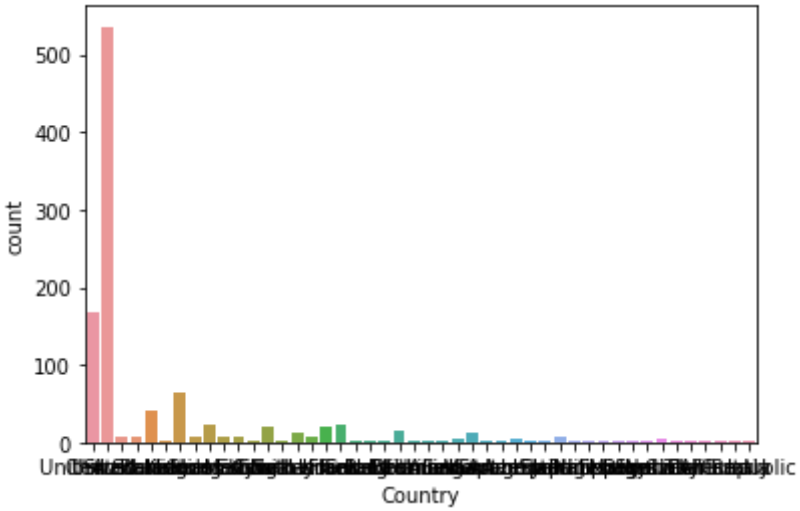
In [3]:
```
sns.countplot(x="Investors Count", data=unicorn_data)
```

Out[3]: <AxesSubplot:xlabel='Investors Count', ylabel='count'>



In [4]:
```
sns.countplot(x="Country", data=unicorn_data)
```

Out[4]: <AxesSubplot:xlabel='Country', ylabel='count'>



In [5]:
```
unicorn_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1037 entries, 0 to 1036
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Company            1037 non-null   object
 1   Valuation ($B)     1037 non-null   float64
 2   Date Joined        1037 non-null   object
 3   Country            1037 non-null   object
 4   City               1037 non-null   object
 5   Industry           1037 non-null   object
 6   Select Inverstors  1037 non-null   object
 7   Founded Year       1037 non-null   int64
 8   Total Raised       1037 non-null   object
 9   Financial Stage    1037 non-null   object
 10  Investors Count    1037 non-null   int64
 11  Deal Terms         1037 non-null   int64
 12  Portfolio Exits    1037 non-null   object
dtypes: float64(1), int64(3), object(9)
memory usage: 105.4+ KB
```

Preprocessing

Change countries column to 2 different boolean columns: "In US" & "Outside US"

In [6]:
```
unicorn_data.isnull()
```

Out[6]:

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	Total Raised	Financia Stage
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	..
1032	False	False	False	False	False	False	False	False	False	False
1033	False	False	False	False	False	False	False	False	False	False
1034	False	False	False	False	False	False	False	False	False	False
1035	False	False	False	False	False	False	False	False	False	False
1036	False	False	False	False	False	False	False	False	False	False

1037 rows × 13 columns

In [7]:
```
unicorn_data.isnull().sum()
```

Out[7]:
```
Company             0
Valuation ($B)      0
Date Joined         0
Country             0
City                0
Industry            0
```

```
Select Inverstors    0
Founded Year         0
Total Raised         0
Financial Stage      0
Investors Count      0
Deal Terms           0
Portfolio Exits      0
dtype: int64
```

In [8]:
```
finStage = pd.get_dummies(unicorn_data["Financial Stage"])
finStage.head(5)
```

Out[8]:

	Acq	Acquired	Asset	Corporate	Divestiture	IPO	Management	None	Reverse	Take
0	0	0	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0	1	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0	0	0
4	0	1	0	0	0	0	0	0	0	0

In [9]:
```
indust = pd.get_dummies(unicorn_data["Industry"])
indust.head(5)
```

Out[9]:

	500 Global, Rakuten Ventures, Golden Gate Ventures	Andreessen Horowitz, DST Global, IDG Capital	Artificial Intelligence	Artificial intelligence	Auto & transportation	B Capital Group, Monk's Hill Ventures, Dynamic Parcel Distribution	Consumer & retail	Cyberse
0	0	0	0	1	0	0	0	
1	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	

5 rows × 33 columns

In [10]:
```
cntry = pd.get_dummies(unicorn_data["Country"])
cntry.head(5)
```

Out[10]:

	Argentina	Australia	Austria	Bahamas	Belgium	Bermuda	Brazil	Canada	Chile	China	...	S K
0	0	0	0	0	0	0	0	0	0	1	...	
1	0	0	0	0	0	0	0	0	0	0	...	
2	0	0	0	0	0	0	0	0	0	0	...	

	Argentina	Australia	Austria	Bahamas	Belgium	Bermuda	Brazil	Canada	Chile	China	...	S K
3	0	0	0	0	0	0	0	0	0	0	...	
4	0	0	0	0	0	0	0	0	0	0	...	

5 rows × 46 columns

In [11]:
```
unicorn_data.drop("City", axis=1, inplace=True)
```

In [12]:
```
unicorn_data.drop("Select Inverstors", axis=1, inplace=True)
```

In [13]:
```
invest = pd.get_dummies(unicorn_data["Portfolio Exits"])
```

In [14]:
```
invest.head(5)
```

Out[14]:

	1	2	3	5	None
0	0	0	0	1	0
1	0	0	0	0	1
2	1	0	0	0	0
3	1	0	0	0	0
4	0	1	0	0	0

In [15]:
```
comp = pd.get_dummies(unicorn_data["Company"])
comp.head(5)
```

Out[15]:

	1047 Games	1KMXC	1Password	4Paradigm	56PINGTAI	58 Daojia	6Sense	ABL Space Systems	AIWAYS	ASAPP
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0

5 rows × 1035 columns

In [ ]:

In [ ]:

In [ ]:

```
```

In [ ]:

```
```

In [16]:

```
unicorn_data.drop("Financial Stage", axis=1, inplace=True)
unicorn_data.drop("Industry", axis=1, inplace=True)
unicorn_data.drop("Country", axis=1, inplace=True)
unicorn_data.drop("Portfolio Exits", axis=1, inplace=True)
unicorn_data.drop("Company", axis=1, inplace=True)
unicorn_data.drop("Date Joined", axis=1, inplace=True)
unicorn_data.drop("Total Raised", axis=1, inplace=True)
```

In [17]:

```
unicorn_data=pd.concat([unicorn_data,finStage,indust,cntry,comp],axis=1)
```

In [18]:

```
unicorn_data.isnull().sum()
```

Out[18]:

```
Valuation ($B)     0
Founded Year       0
Investors Count    0
Deal Terms         0
Acq                0
                  ..
iTutorGroup        0
o9 Solutions       0
reddit             0
solarisBank        0
wefox              0
Length: 1128, dtype: int64
```

In [19]:

```
unicorn_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1037 entries, 0 to 1036
Columns: 1128 entries, Valuation ($B) to wefox
dtypes: float64(1), int64(3), uint8(1124)
memory usage: 1.1 MB
```

Making the model

In [20]:

```
#x=unicorn_data.drop(['Valuation ($B)'], axis=1)
#x=np.array(unicorn_data.drop(['Valuation ($B)'], axis=1))
x=np.array(unicorn_data["Deal Terms"]).reshape(-1, 1)
y=np.array(unicorn_data['Valuation ($B)']).reshape(-1, 1)

x[0:1]
```

Out[20]:

```
array([[28]])
```

In [21]:

```
y[0:5]
```

Out[21]:

```
array([[140. ],
       [100.3],
       [ 95. ],
       [ 45.6],
       [ 42. ]])
```

In [22]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

X_train, X_test, y_train, y_test = train_test_split(x,y,test_size=0.3,random_state=1
linModel = LinearRegression()

pipe = make_pipeline(StandardScaler(), linModel)
pipe.fit(X_train, y_train)
```

Out[22]:
```
Pipeline(steps=[('standardscaler', StandardScaler()),
                ('linearregression', LinearRegression())])
```

In [23]:
```
pipe.score(X_test,y_test)
```

Out[23]:
```
0.07642008176817283
```

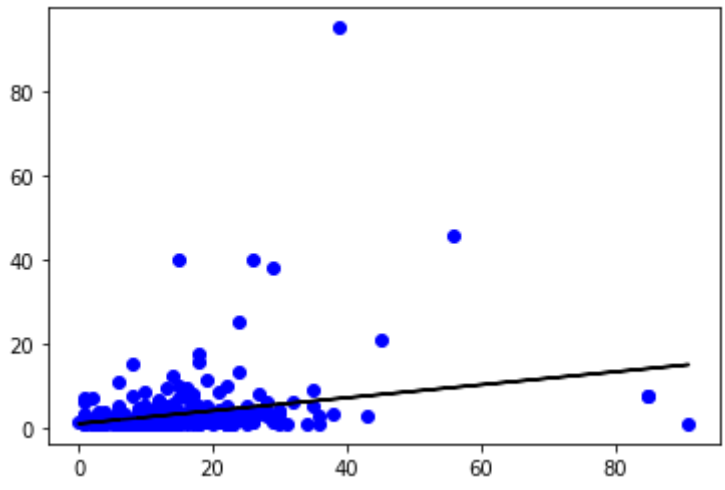
In [24]:
```
y_pred = pipe.predict(X_test)
```

In [25]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred)
```

Out[25]:
```
0.07642008176817283
```

In [26]:
```
plt.scatter(X_test, y_test, color ='b')
#plt.plot(X, y, color='k')
plt.plot(X_test, y_pred, color ='k')
```

Out[26]:
```
[<matplotlib.lines.Line2D at 0x7fcc5bc5cbe0>]
```



In [ ]:
```
```