Group Assignment (ANN)

This project sought to analyse the data of over 1000 companies using Artificial Neural Network in order to forecast the valuation of another company given specific variables.

Importing nessesury Module and library

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()

import tensorflow as tf
from tensorflow import keras

import sklearn
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import r2_score, classification_report, confusion_matrix
```

```
2022-05-06 11:22:03.290149: W tensorflow/stream_executor/platform/default/dso_loade
r.cc:64] Could not load dynamic library 'libcudart.so.11.0'; dlerror: libcudart.so.1
1.0: cannot open shared object file: No such file or directory
2022-05-06 11:22:03.290176: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ign
ore above cudart dlerror if you do not have a GPU set up on your machine.
```

Importing dataset

In [2]:
```
#import diabetes dataset
df = pd.read_csv("/home/md12.islam/ ML_Module/Untitled Folder/Unicorn_Companies.csv"
#show size of rows and columns
df.shape
#show first 5 rows in dataset
df.head()
```

Out[2]:

	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

Data Exploration

In [3]:
```
df.info()
df.describe(include='all')
```

```
<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
```

Out[3]:

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	Ra
count	1037	1037.000000	1037	1037	1037	1037	1037	1037.000000	
unique	1035	NaN	623	46	256	33	1006	NaN	
top	Bolt	NaN	7/13/2021	United States	San Francisco	Fintech	None	NaN	N
freq	2	NaN	9	536	145	205	17	NaN	
mean	NaN	3.290154	NaN	NaN	NaN	NaN	NaN	1929.251688	
std	NaN	7.310122	NaN	NaN	NaN	NaN	NaN	401.499520	
min	NaN	1.000000	NaN	NaN	NaN	NaN	NaN	0.000000	
25%	NaN	1.070000	NaN	NaN	NaN	NaN	NaN	2011.000000	
50%	NaN	1.600000	NaN	NaN	NaN	NaN	NaN	2014.000000	
75%	NaN	3.100000	NaN	NaN	NaN	NaN	NaN	2016.000000	
max	NaN	140.000000	NaN	NaN	NaN	NaN	NaN	2021.000000	

- this dataset has total 13 columns and 1037 row.
- only Valuation ($B), Investors Count,Deal Terms are numerical value rests
- rest are objects(needs to be modified to numeric)

In [4]:
```
# Check if any column has null variables
df.isnull().sum()
```

Out[4]:
```
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 [5]:

```
#Check if any column has None Value

for column in df.columns:
    if 'None' in df[column].unique():
        print(column)
```

```
Select Inverstors
Total Raised
Financial Stage
Portfolio Exits
/tmp/ipykernel_613377/2188549246.py:4: FutureWarning: elementwise comparison failed;
returning scalar instead, but in the future will perform elementwise comparison
  if 'None' in df[column].unique():
```

In [6]:

```
#Check if any column has None Value

for column in df.columns:
    if 0 in df[column].unique():
        print(column)
```

```
Founded Year
Investors Count
Deal Terms
```

In [7]:

```
# Exploring Industry columns
print(df["Industry"].value_counts())
```

```
Fintech                                                          205
Internet software & services                                     192
E-commerce & direct-to-consumer                                  107
Artificial intelligence                                           71
Health                                                            69
Supply chain, logistics, & delivery                               57
Other                                                             56
Cybersecurity                                                     49
Data management & analytics                                       41
Mobile & telecommunications                                       37
Hardware                                                          33
Auto & transportation                                             29
Edtech                                                            28
Consumer & retail                                                 25
Travel                                                            14
Artificial Intelligence                                            7
Finttech                                                           1
Sequoia Capital China, Shunwei Capital Partners, Qualgro           1
B Capital Group, Monk's Hill Ventures, Dynamic Parcel Distribution 1
Andreessen Horowitz, DST Global, IDG Capital                       1
Vertex Ventures SE Asia, Global Founders Capital, Visa Ventures    1
Mundi Ventures, Doqling Capital Partners, Activant Capital         1
SingTel Innov8, Alpha JWC Ventures, Golden Gate Ventures           1
Dragonfly Captial, Qiming Venture Partners, DST Global             1
Sequoia Capital China, ING, Alibaba Entrepreneurs Fund             1
Sequoia Capital, Thoma Bravo, Softbank                             1
500 Global, Rakuten Ventures, Golden Gate Ventures                 1
```

```
Hopu Investment Management, Boyu Capital, DC Thomson Ventures          1
Vision Plus Capital, GSR Ventures, ZhenFund                            1
Jungle Ventures, Accel, Venture Highway                                1
Tiger Global Management, Tiger Brokers, DCM Ventures                   1
Kuang-Chi                                                              1
Temasek, Guggenheim Investments, Qatar Investment Authority            1
Name: Industry, dtype: int64
```

column(Industry) has few unique value which accured only once, so we are going to get rid of those rare row so that our Model doesn't gets confused.

In [8]:
```
uselessnames = ["Finttech","Sequoia Capital China, Shunwei Capital Partners, Qualgro
for i in uselessnames:
    df.drop( df[ df["Industry"] == i ].index , inplace=True)
```

In [9]:
```
print(df["Industry"].value_counts())
```

```
Fintech                                205
Internet software & services           192
E-commerce & direct-to-consumer        107
Artificial intelligence                 71
Health                                  69
Supply chain, logistics, & delivery     57
Other                                   56
Cybersecurity                           49
Data management & analytics             41
Mobile & telecommunications             37
Hardware                                33
Auto & transportation                   29
Edtech                                  28
Consumer & retail                       25
Travel                                  14
Artificial Intelligence                  7
Name: Industry, dtype: int64
```

Same Process for the column(Finalcial Stage)

In [10]:
```
print(df["Financial Stage"].value_counts())
```

```
None           973
Acquired        21
Divestiture      8
IPO              7
Acq              6
Asset            1
Take             1
Management       1
Reverse          1
Corporate        1
Name: Financial Stage, dtype: int64
```

In [11]:
```
uselessnames = ["Asset","Take","Management","Reverse","Corporate"]
for i in uselessnames:
    df.drop( df[ df["Financial Stage"] == i ].index , inplace=True)
```

In [12]:
```
print(df["Financial Stage"].value_counts())
```

```
None           973
Acquired        21
Divestiture      8
IPO              7
```

```
Acq              6
Name: Financial Stage, dtype: int64
```

Data Modifidation

column(Date Joined, Founded year) has Date object type data. which is not good for machine learning. We are going to convert this data into age data as two new column as ("Age" and "Unicorn Age")

In [13]:
```
from datetime import datetime, date

df['Date Joined']
```

Out[13]:
```
0       04/07/2017
1       12/01/2012
3       12/12/2011
4       10/26/2018
5       01/08/2018
           ...
1032     2/22/2022
1033     2/23/2022
1034     2/23/2022
1035     2/23/2022
1036     2/24/2022
Name: Date Joined, Length: 1015, dtype: object
```

In [14]:
```
df['Date Joined'] = pd.to_datetime(df['Date Joined'], format='%m/%d/%Y')
# Create 'year_joined column'
df['year joined'] = df['Date Joined'].dt.year
```

In [15]:
```
df['age'] = (2022-(df['Founded Year']))
df['Unicorn age'] = (2022-(df['year joined']))
df.head()
```

Out[15]:

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	Total Raised
0	Bytedance	140.0	2017-04-07	China	Beijing	Artificial intelligence	Sequoia Capital China, SIG Asia Investments, S...	2012	$7.44B
1	SpaceX	100.3	2012-12-01	United States	Hawthorne	Other	Founders Fund, Draper Fisher Jurvetson, Rothen...	2002	$6.874B
3	Klarna	45.6	2011-12-12	Sweden	Stockholm	Fintech	Institutional Venture Partners, Sequoia Capita...	2005	$3.472B
4	Epic Games	42.0	2018-10-26	United States	Cary	Other	Tencent Holdings, KKR, Smash Ventures	1991	$4.377B

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	Total Raised
5	Canva	40.0	2018-01-08	Australia	Surry Hills	Internet software & services	Sequoia Capital China, Blackbird Ventures, Mat...	2012	$571.26M

Converting "Total Raised" Strings into integer Type

In [16]:
```
df['Total Raised']
```

Out[16]:
```
0         $7.44B
1        $6.874B
3        $3.472B
4        $4.377B
5       $571.26M
          ...
1032    $181.06M
1033       $700M
1034    $449.72M
1035     $525.5M
1036     $604.5M
Name: Total Raised, Length: 1015, dtype: object
```

In [17]:
```
def multiply_by_amount_raised(amt):
    try:
        amount = {'B': 1000000000, 'M': 1000000, 'K': 1000}
        return float(amt[:-1]) * amount[amt[-1:]]
    except TypeError:
        pass
```

In [18]:
```
df['Total Raised'] = df['Total Raised'].str.replace('$','')
df['Total Raised'] = df['Total Raised'].replace('None',0)
df['Total Raised'] = df['Total Raised'].apply(multiply_by_amount_raised)
```

```
/tmp/ipykernel_613377/2827083612.py:1: FutureWarning: The default value of regex wil
l change from True to False in a future version. In addition, single character regul
ar expressions will *not* be treated as literal strings when regex=True.
  df['Total Raised'] = df['Total Raised'].str.replace('$','')
```

In [19]:
```
df.head()
```

Out[19]:

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	Total Rais
0	Bytedance	140.0	2017-04-07	China	Beijing	Artificial intelligence	Sequoia Capital China, SIG Asia Investments, S...	2012	7.440000e+

	Company	Valuation ($B)	Date Joined	Country	City	Industry	Select Inverstors	Founded Year	Total Rais
1	SpaceX	100.3	2012-12-01	United States	Hawthorne	Other	Founders Fund, Draper Fisher Jurvetson, Rothen...	2002	6.874000e+
3	Klarna	45.6	2011-12-12	Sweden	Stockholm	Fintech	Institutional Venture Partners, Sequoia Capita...	2005	3.472000e+
4	Epic Games	42.0	2018-10-26	United States	Cary	Other	Tencent Holdings, KKR, Smash Ventures	1991	4.377000e+
5	Canva	40.0	2018-01-08	Australia	Surry Hills	Internet software & services	Sequoia Capital China, Blackbird Ventures, Mat...	2012	5.712600e+

In [20]:
```
print(df["Industry"].value_counts())
```

```
Fintech                                203
Internet software & services           192
E-commerce & direct-to-consumer        107
Artificial intelligence                 71
Health                                  69
Supply chain, logistics, & delivery     57
Other                                   55
Cybersecurity                           49
Data management & analytics             41
Mobile & telecommunications             36
Hardware                                33
Auto & transportation                   29
Edtech                                  28
Consumer & retail                       24
Travel                                  14
Artificial Intelligence                  7
Name: Industry, dtype: int64
```

converting strinng type value to seprate column

get_dummies() function is used to convert categorical variable into dummy/indicator variables. Data of which to get dummy indicators. String to append DataFrame column names.

In [21]:
```
df= pd.get_dummies(data=df, columns=["Industry"])
```

Dropping all useless column

In [22]:
```
useless_columns=["Company","Date Joined", "Country","City","Financial Stage","Select

for item in useless_columns:
```

```
    df.drop(item , axis=1, inplace=True)
```

In [23]:

```
df.info()
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1015 entries, 0 to 1036
Data columns (total 22 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Valuation ($B)                                1015 non-null   float64
 1   Total Raised                                  991 non-null    float64
 2   Investors Count                               1015 non-null   int64
 3   Deal Terms                                    1015 non-null   int64
 4   age                                           1015 non-null   int64
 5   Unicorn age                                   1015 non-null   int64
 6   Industry_Artificial Intelligence              1015 non-null   uint8
 7   Industry_Artificial intelligence              1015 non-null   uint8
 8   Industry_Auto & transportation                1015 non-null   uint8
 9   Industry_Consumer & retail                    1015 non-null   uint8
 10  Industry_Cybersecurity                        1015 non-null   uint8
 11  Industry_Data management & analytics          1015 non-null   uint8
 12  Industry_E-commerce & direct-to-consumer      1015 non-null   uint8
 13  Industry_Edtech                               1015 non-null   uint8
 14  Industry_Fintech                              1015 non-null   uint8
 15  Industry_Hardware                             1015 non-null   uint8
 16  Industry_Health                               1015 non-null   uint8
 17  Industry_Internet software & services         1015 non-null   uint8
 18  Industry_Mobile & telecommunications          1015 non-null   uint8
 19  Industry_Other                                1015 non-null   uint8
 20  Industry_Supply chain, logistics, & delivery  1015 non-null   uint8
 21  Industry_Travel                               1015 non-null   uint8
dtypes: float64(2), int64(4), uint8(16)
memory usage: 71.4 KB
```

Out[23]:

	Valuation ($B)	Total Raised	Investors Count	Deal Terms	age	Unicorn age	Industry_Artificial Intelligence	Industry_Artificial intelligence	t
0	140.0	7.440000e+09	28	8	10	5	0	1	
1	100.3	6.874000e+09	29	12	20	10	0	0	
3	45.6	3.472000e+09	56	13	17	11	0	0	
4	42.0	4.377000e+09	25	5	31	4	0	0	
5	40.0	5.712600e+08	26	8	10	4	0	0	

5 rows × 22 columns

In This Stage We have all columns with numarical values, no more object type values

Scaling all values in range of 0 to 1

In [24]:

```
cols_to_scale= ["Valuation ($B)","Total Raised","Investors Count","Deal Terms","age"

from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
```

```
df[cols_to_scale]= scaler.fit_transform(df[cols_to_scale])
df.describe()
```

Out[24]:

	Valuation ($B)	Total Raised	Investors Count	Deal Terms	age	Unicorn age	Industry_Artific Intelliger
count	1015.000000	991.000000	1015.000000	1015.000000	1015.000000	1015.000000	1015.0000
mean	0.015775	0.076594	0.157906	0.157584	0.045312	0.145419	0.0068
std	0.048457	0.098406	0.109246	0.114701	0.198468	0.133541	0.0827
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	0.000719	0.030390	0.087912	0.052632	0.002474	0.066667	0.0000
50%	0.004317	0.050604	0.142857	0.157895	0.003464	0.066667	0.0000
75%	0.015108	0.084871	0.197802	0.210526	0.004948	0.200000	0.0000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000

8 rows × 22 columns

This is the final form of our data.

- 22 column , 1015 row
- all data in numarical form and between (0 to 1)

Spliting data into Train and Test dataset

In [25]:
```
X = df.drop("Valuation ($B)",axis ="columns")
y = df["Valuation ($B)"]
```

In [26]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size= 0.2, random_state
```

In [27]:
```
for i in(X_train, X_test, y_train, y_test):
    print(i.shape)
```

```
(812, 21)
(203, 21)
(812,)
(203,)
```

Building the Model

Useing keras.Sequential Model to make ANN with 21 Input layer and 1 output layer. inside it has 15 hidden layer

In [28]:
```
model = keras.Sequential([
    keras.layers.Dense(21, input_shape=(21,), activation='relu'),
    keras.layers.Dense(15, activation='relu'),
    keras.layers.Dense(1, activation='sigmoid')
])
```

```
# opt = keras.optimizers.Adam(learning_rate=0.01)

model.compile(optimizer='adam',
              loss='mean_squared_error',
              metrics=['accuracy'])


model.fit(X_train, y_train, epochs=10)
```

```
2022-05-06 11:22:04.741049: W tensorflow/stream_executor/platform/default/dso_loade
r.cc:64] Could not load dynamic library 'libcuda.so.1'; dlerror: libcuda.so.1: canno
t open shared object file: No such file or directory
2022-05-06 11:22:04.741080: W tensorflow/stream_executor/cuda/cuda_driver.cc:269] fa
iled call to cuInit: UNKNOWN ERROR (303)
2022-05-06 11:22:04.741098: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:15
6] kernel driver does not appear to be running on this host (csctcloud): /proc/drive
r/nvidia/version does not exist
2022-05-06 11:22:04.741326: I tensorflow/core/platform/cpu_feature_guard.cc:142] Thi
s TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to
use the following CPU instructions in performance-critical operations:  AVX2 AVX512F
FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler
flags.
2022-05-06 11:22:04.815505: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.
cc:185] None of the MLIR Optimization Passes are enabled (registered 2)
Epoch 1/10
26/26 [==============================] - 0s 900us/step - loss: nan - accuracy: 0.215
5
Epoch 2/10
26/26 [==============================] - 0s 838us/step - loss: nan - accuracy: 0.215
5
Epoch 3/10
26/26 [==============================] - 0s 843us/step - loss: nan - accuracy: 0.215
5
Epoch 4/10
26/26 [==============================] - 0s 848us/step - loss: nan - accuracy: 0.215
5
Epoch 5/10
26/26 [==============================] - 0s 841us/step - loss: nan - accuracy: 0.215
5
Epoch 6/10
26/26 [==============================] - 0s 841us/step - loss: nan - accuracy: 0.215
5
Epoch 7/10
26/26 [==============================] - 0s 844us/step - loss: nan - accuracy: 0.215
5
Epoch 8/10
26/26 [==============================] - 0s 835us/step - loss: nan - accuracy: 0.215
5
Epoch 9/10
26/26 [==============================] - 0s 863us/step - loss: nan - accuracy: 0.215
5
Epoch 10/10
26/26 [==============================] - 0s 817us/step - loss: nan - accuracy: 0.215
5
```

Out[28]: <keras.callbacks.History at 0x7f3eac6ce2e0>

In [29]:

```
# evaluate the keras model
_, accuracy = model.evaluate(X_test, y_test)
print('Accuracy: %.2f' % (accuracy*100))
```

```
7/7 [==============================] - 0s 863us/step - loss: nan - accuracy: 0.3005
Accuracy: 30.05
```

Evaluation

With this Neural Network we get 30.05 Accuracy, which is something but not has a lot to improve

GradientBoostingRegressor

Gradient boosting is a regression and classification machine learning technique that generates a prediction model in the form of an ensemble of weak prediction models.

In [30]:
```
X_train = X_train.replace((np.inf, -np.inf, np.nan), 0).reset_index(drop=True)
X_test = X_test.replace((np.inf, -np.inf, np.nan), 0).reset_index(drop=True)
```

In [31]:
```
from sklearn.ensemble import GradientBoostingRegressor

GradientBoost_model = GradientBoostingRegressor(
    n_estimators=1000,
    learning_rate=0.1,
    random_state=0)
GradientBoost_model.fit(X_train, y_train)


GradientBoost_model.score(X_test, y_test)
```

Out[31]: 0.740816285311199

Using this Model we can see it produced much better accuracy for our dataset.

with 74% Accuracy our model is definately usable in real world Problem