Artificial Neural Network

Objective:
You are working as a car salesman and you would like to develop a model to predict the total amount that customers are willing to pay given the following attributes:

Customer Name
Customer e-mail
Country
Gender
Age
Annual Salary
Credit Card Debt
Net Worth

The Model should predict:

Car Purchase Amount

Libraries Import

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

Import Dataset

In [2]:

```
car_df =  pd.read_csv('G:/TCS Study/TCS Git Hub Projects/Artificial Neural Network Proj
car_df.head()
```

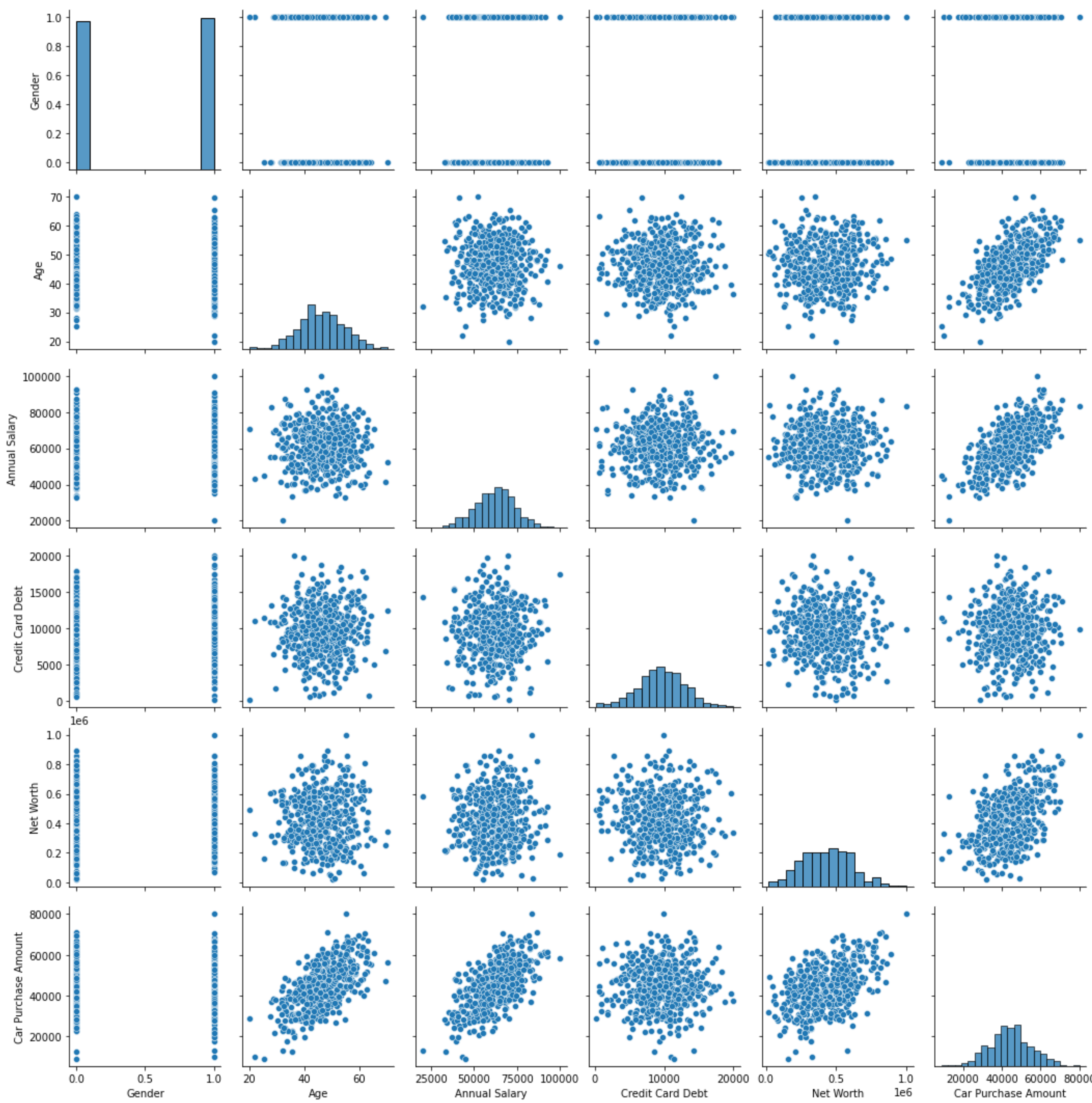
Out[2]:

	Customer Name	Customer e-mail	Country	Gender	Age	Annual Salary
0	Martina Avila	cubilia.Curae.Phasellus@quisaccumsanconvallis.edu	Bulgaria	0	41.851720	62812.09301
1	Harlan Barnes	eu.dolor@diam.co.uk	Belize	0	40.870623	66646.89292
2	Naomi Rodriquez	vulputate.mauris.sagittis@ametconsectetueradip...	Algeria	1	43.152897	53798.55112
3	Jade Cunningham	malesuada@dignissim.com	Cook Islands	1	58.271369	79370.03798
4	Cedric Leach	felis.ullamcorper.viverra@egetmollislectus.net	Brazil	1	57.313749	59729.15130

Visualize Dataset

In [3]:
```
sns.pairplot(car_df)
```

Out[3]:
```
<seaborn.axisgrid.PairGrid at 0x22b08430fc8>
```



Create Testing and Training Dataset/Data Cleaning

In [4]:
```
X = car_df.drop(['Customer Name', 'Customer e-mail', 'Country', 'Car Purchase Amount'],
X.head()
```

Out[4]:

	Gender	Age	Annual Salary	Credit Card Debt	Net Worth
0	0	41.851720	62812.09301	11609.380910	238961.2505
1	0	40.870623	66646.89292	9572.957136	530973.9078
2	1	43.152897	53798.55112	11160.355060	638467.1773
3	1	58.271369	79370.03798	14426.164850	548599.0524
4	1	57.313749	59729.15130	5358.712177	560304.0671

In [5]:

```
y = car_df['Car Purchase Amount']
y.head()
```

Out[5]:

```
0    35321.45877
1    45115.52566
2    42925.70921
3    67422.36313
4    55915.46248
Name: Car Purchase Amount, dtype: float64
```

In [6]:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
X_scaled
```

Out[6]:

```
array([[0.        , 0.4370344 , 0.53515116, 0.57836085, 0.22342985],
       [0.        , 0.41741247, 0.58308616, 0.476028  , 0.52140195],
       [1.        , 0.46305795, 0.42248189, 0.55579674, 0.63108896],
       ...,
       [1.        , 0.67886994, 0.61110973, 0.52822145, 0.75972584],
       [1.        , 0.78321017, 0.37264988, 0.69914746, 0.3243129 ],
       [1.        , 0.53462305, 0.51713347, 0.46690159, 0.45198622]])
```

In [7]:

```
scaler.data_max_
```

Out[7]:

```
array([1.e+00, 7.e+01, 1.e+05, 2.e+04, 1.e+06])
```

In [8]:

```
scaler.data_min_
```

Out[8]:

```
array([    0.,    20., 20000.,   100., 20000.])
```

In [9]:

```
y_new = y.values.reshape(-1,1)
y_scaled = scaler.fit_transform(y_new)
y_scaled
```

Out[9]:

```
array([[0.37072477],
       [0.50866938],
       [0.47782689],
       [0.82285018],
       [0.66078116],
       [0.67059152],
       [0.28064374],
       [0.54133778],
```

```
       [0.54948752],
       [0.4111198 ],
       [0.70486638],
       [0.46885649],
       [0.27746526],
       [0.56702642],
       [0.57056385],
       [0.61996151],
       [0.46217916],
       [0.49157341],
       [0.50188722],
       [0.64545808],
       [0.59339372],
       [0.48453965],
       [0.53860366],
       [0.53007738],
       [0.50814651],
       [0.49841668],
       [0.3966416 ],
       [0.56467566],
       [0.6950749 ],
       [0.49287831],
       [0.12090943],
       [0.50211776],
       [0.80794216],
       [0.62661214],
       [0.43394857],
       [0.60017103],
       [0.42223485],
       [0.01538345],
       [0.37927499],
       [0.64539707],
       [0.51838974],
       [0.45869677],
       [0.26804521],
       [0.2650104 ],
       [0.84054134],
       [0.84401542],
       [0.35515157],
       [0.406246  ],
       [0.40680623],
       [0.55963883],
       [0.2561583 ],
       [0.77096325],
       [0.55305289],
       [0.5264948 ],
       [0.3236476 ],
       [0.55070832],
       [0.54057623],
       [0.45669016],
       [0.41053254],
       [0.33433524],
       [0.39926954],
       [0.5420261 ],
       [0.57366948],
       [0.43793831],
       [0.46897896],
       [0.61908354],
       [0.55076214],
       [0.48846357],
       [0.61560519],
```

```
[0.56891394],
[0.30761974],
[0.56863909],
[0.46343171],
[0.50787179],
[0.61959897],
[0.59095889],
[0.45573492],
[0.49908055],
[0.4155271 ],
[0.45382041],
[0.41407692],
[0.45713635],
[0.30133556],
[0.47019791],
[0.42256447],
[0.14863766],
[0.50939895],
[0.38056275],
[0.59067519],
[0.05486922],
[0.4219045 ],
[0.59466257],
[0.23904164],
[0.72775122],
[0.63486085],
[0.43668833],
[0.74075381],
[0.42962519],
[0.61231998],
[0.46743215],
[0.68227386],
[0.19270023],
[0.34705263],
[0.31747576],
[0.72480623],
[0.51744133],
[0.36343414],
[0.36116341],
[0.26178477],
[0.64750739],
[0.56538749],
[0.70197943],
[0.68037083],
[0.60481233],
[0.29916352],
[0.81860421],
[0.47053558],
[0.45706646],
[0.47313925],
[0.35945319],
[0.46779854],
[0.46357095],
[0.71008706],
[0.59721993],
[0.64444254],
[0.53723155],
[0.78016463],
[0.39792327],
[0.61500514],
[0.49297475],
```

```
       [0.59931944],
       [0.4118826 ],
       [0.43333307],
       [0.43771229],
       [0.33988067],
       [0.55806565],
       [0.54497514],
       [0.42831636],
       [0.344062  ],
       [0.33380674],
       [0.75865395],
       [0.28768775],
       [0.49885366],
       [0.3893741 ],
       [0.62895125],
       [0.52080458],
       [0.41555485],
       [0.54839351],
       [0.72143981],
       [0.42155708],
       [0.26493265],
       [0.54372318],
       [0.46981253],
       [0.31409216],
       [0.46999496],
       [0.32165106],
       [0.24646921],
       [0.41088501],
       [0.42863953],
       [0.40442699],
       [0.67782275],
       [0.52751195],
       [0.49091635],
       [0.65623527],
       [0.47160596],
       [0.44900269],
       [0.16412978],
       [0.37237754],
       [0.38187033],
       [0.41101838],
       [0.45107076],
       [0.26605566],
       [0.48907732],
       [0.68212351],
       [0.45217002],
       [0.56409653],
       [0.45444407],
       [0.78232624],
       [0.28242388],
       [0.3059129 ],
       [0.41919878],
       [0.42720002],
       [0.33251345],
       [0.69078257],
       [0.63925743],
       [0.38927052],
       [0.42988917],
       [0.47856826],
       [0.73050372],
       [0.52648517],
       [0.67013948],
```

```
[0.47569515],
[0.40675569],
[0.50997782],
[0.665203  ],
[0.58387492],
[0.57353959],
[0.31967601],
[0.56647918],
[0.52378641],
[0.38150608],
[0.48259225],
[0.44592089],
[0.60117759],
[0.50035241],
[0.55660425],
[0.51838459],
[0.6457801 ],
[0.33068236],
[0.46773647],
[0.64966102],
[0.54907635],
[0.48459001],
[0.50109082],
[0.40485272],
[0.5465529 ],
[0.5048829 ],
[0.53018684],
[0.66991531],
[0.46018938],
[0.73406295],
[0.39864179],
[0.53369389],
[0.66841888],
[0.51422109],
[0.26233016],
[0.52661271],
[0.28171911],
[0.5965593 ],
[0.46494647],
[0.61485077],
[0.49905236],
[0.52189581],
[0.69345654],
[0.47873651],
[0.58735466],
[0.53534616],
[0.56901367],
[0.47883335],
[0.49908428],
[0.5257114 ],
[0.53305254],
[0.66900144],
[0.47568675],
[0.61005611],
[0.3540794 ],
[0.72905982],
[0.80505204],
[0.31289637],
[0.523032  ],
[0.67567861],
[0.2138602 ],
```

```
       [0.56450168],
       [0.39568902],
       [0.4845291 ],
       [0.28298776],
       [0.55421359],
       [0.34170423],
       [0.45531219],
       [0.56811431],
       [0.5972613 ],
       [0.31338011],
       [0.48730944],
       [0.55352142],
       [0.63399262],
       [0.41795296],
       [0.39544824],
       [0.40771621],
       [0.45521229],
       [0.81533714],
       [0.04981603],
       [0.43026944],
       [0.61562087],
       [0.6268025 ],
       [0.60728039],
       [0.41838955],
       [0.54964825],
       [0.71104101],
       [0.37903726],
       [0.45118709],
       [0.60183344],
       [0.62002621],
       [0.33560612],
       [0.28757249],
       [0.6825565 ],
       [0.58368485],
       [0.45880771],
       [0.52693631],
       [0.54380447],
       [0.8715253 ],
       [0.65554063],
       [0.63167261],
       [0.4352791 ],
       [0.50332973],
       [0.5343264 ],
       [0.27381426],
       [0.41053523],
       [0.47531745],
       [0.2911385 ],
       [0.76110147],
       [0.76406707],
       [0.46931782],
       [0.49948317],
       [0.81820046],
       [0.18438195],
       [0.43693203],
       [0.66298048],
       [0.56960734],
       [0.47179899],
       [0.39556023],
       [0.60375334],
       [0.37628608],
       [0.43511174],
```

```
       [0.37719946],
       [0.47698891],
       [1.        ],
       [0.72573208],
       [0.71491172],
       [0.43107389],
       [0.69917902],
       [0.61949147],
       [0.58685749],
       [0.71302854],
       [0.19197549],
       [0.4526464 ],
       [0.62671101],
       [0.38794277],
       [0.48597146],
       [0.3119255 ],
       [0.3172683 ],
       [0.31103807],
       [0.51220225],
       [0.22891454],
       [0.43505067],
       [0.60902435],
       [0.43537481],
       [0.51912329],
       [0.30740999],
       [0.42849005],
       [0.36167369],
       [0.20447774],
       [0.27793926],
       [0.70558126],
       [0.58012487],
       [0.3754806 ],
       [0.52673735],
       [0.32804493],
       [0.56449711],
       [0.56829414],
       [0.45672673],
       [0.21439436],
       [0.49893066],
       [0.72380375],
       [0.47597174],
       [0.53430602],
       [0.69953618],
       [0.40905353],
       [0.42634618],
       [0.64234067],
       [0.4237175 ],
       [0.54907212],
       [0.5222931 ],
       [0.30935321],
       [0.37643596],
       [0.56429808],
       [0.56275857],
       [0.39694511],
       [0.53113416],
       [0.61816285],
       [0.29231919],
       [0.73184274],
       [0.4362737 ],
       [0.41613807],
       [0.67273869],
```

```
       [0.76168793],
       [0.65775822],
       [0.3854533 ],
       [0.61236387],
       [0.58164331],
       [0.39802597],
       [0.54529522],
       [0.28930803],
       [0.72629843],
       [0.38204913],
       [0.68033622],
       [0.60453629],
       [0.37815253],
       [0.47470876],
       [0.65035196],
       [0.24788603],
       [0.63371047],
       [0.54888405],
       [0.48791067],
       [0.46027877],
       [0.76253593],
       [0.30644588],
       [0.79707352],
       [0.40664378],
       [0.47773971],
       [0.19154167],
       [0.86759109],
       [0.48228989],
       [0.41040247],
       [0.30168732],
       [0.77280198],
       [0.50863303],
       [0.49805458],
       [0.14824364],
       [0.57734658],
       [0.7427002 ],
       [0.4615406 ],
       [0.52679628],
       [0.39967091],
       [0.34882593],
       [0.3051776 ],
       [0.60642838],
       [0.30614901],
       [0.4287247 ],
       [0.41091372],
       [0.44492764],
       [0.74688327],
       [0.5558489 ],
       [0.43795845],
       [0.571387  ],
       [0.31561446],
       [0.54534475],
       [0.37724541],
       [0.47754279],
       [0.55657526],
       [0.51540401],
       [0.32481193],
       [0.32687852],
       [0.37288737],
       [0.28900791],
       [0.6538108 ],
```

```
[0.46675557],
[0.58506904],
[0.36510463],
[0.49152498],
[0.42443705],
[0.45278122],
[0.21316327],
[0.4747199 ],
[0.42114943],
[0.27669569],
[0.60775236],
[0.81194719],
[0.47759537],
[0.56687473],
[0.25779114],
[0.54746234],
[0.71808681],
[0.51086565],
[0.        ],
[0.52129727],
[0.33756622],
[0.56035434],
[0.51865585],
[0.43805795],
[0.3726359 ],
[0.2895323 ],
[0.41187412],
[0.499023  ],
[0.59220313],
[0.61366712],
[0.73808769],
[0.27413582],
[0.26177748],
[0.54900684],
[0.26992761],
[0.85449963],
[0.55003734],
[0.39949626],
[0.50001154],
[0.36815842],
[0.6502447 ],
[0.55469987],
[0.37779655],
[0.38757338],
[0.62128001],
[0.62041473],
[0.17564949],
[0.50726309],
[0.65320953],
[0.6691568 ],
[0.54146119],
[0.45760058],
[0.33173992],
[0.46457217],
[0.7118085 ],
[0.4556686 ],
[0.61669253],
[0.71996731],
[0.54592485],
[0.77729956],
[0.56199216],
```

```
       [0.31678049],
       [0.77672238],
       [0.51326977],
       [0.50855247]])
```

Training the Model

In [10]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_size = 0.2
```

In [11]:

```
X_train.shape
```

Out[11]:

```
(375, 5)
```

In [25]:

```
import tensorflow.keras
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(5, input_dim = 5, activation = 'relu'))
model.add(Dense(5, activation = 'relu'))
model.add(Dense(1, activation = 'linear'))
```

In [26]:

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_4 (Dense)              (None, 5)                 30        
_________________________________________________________________
dense_5 (Dense)              (None, 5)                 30        
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 6         
=================================================================
Total params: 66
Trainable params: 66
Non-trainable params: 0
_________________________________________________________________
```

In [27]:

```
model.compile(optimizer = 'adam', loss = 'mean_squared_error')
```

In [28]:

```
epochs_hist = model.fit(X_train,  y_train, epochs = 100, batch_size = 50, verbose = 1,
```

```
Train on 300 samples, validate on 75 samples
Epoch 1/100
300/300 [==============================] - 0s 959us/step - loss: 0.3144 - val_loss: 0.33
05
Epoch 2/100
300/300 [==============================] - 0s 113us/step - loss: 0.2968 - val_loss: 0.31
37
Epoch 3/100
```

```
300/300 [==============================] - 0s 100us/step - loss: 0.2812 - val_loss: 0.29
89
Epoch 4/100
300/300 [==============================] - 0s 80us/step - loss: 0.2673 - val_loss: 0.285
4
Epoch 5/100
300/300 [==============================] - 0s 77us/step - loss: 0.2550 - val_loss: 0.273
1
Epoch 6/100
300/300 [==============================] - 0s 73us/step - loss: 0.2439 - val_loss: 0.262
0
Epoch 7/100
300/300 [==============================] - 0s 80us/step - loss: 0.2340 - val_loss: 0.251
7
Epoch 8/100
300/300 [==============================] - 0s 73us/step - loss: 0.2247 - val_loss: 0.242
1
Epoch 9/100
300/300 [==============================] - 0s 80us/step - loss: 0.2160 - val_loss: 0.232
9
Epoch 10/100
300/300 [==============================] - 0s 83us/step - loss: 0.2076 - val_loss: 0.223
9
Epoch 11/100
300/300 [==============================] - 0s 70us/step - loss: 0.1994 - val_loss: 0.215
2
Epoch 12/100
300/300 [==============================] - 0s 80us/step - loss: 0.1914 - val_loss: 0.206
7
Epoch 13/100
300/300 [==============================] - 0s 80us/step - loss: 0.1833 - val_loss: 0.198
3
Epoch 14/100
300/300 [==============================] - 0s 73us/step - loss: 0.1754 - val_loss: 0.189
7
Epoch 15/100
300/300 [==============================] - 0s 87us/step - loss: 0.1676 - val_loss: 0.181
4
Epoch 16/100
300/300 [==============================] - 0s 100us/step - loss: 0.1599 - val_loss: 0.17
33
Epoch 17/100
300/300 [==============================] - 0s 73us/step - loss: 0.1525 - val_loss: 0.165
3
Epoch 18/100
300/300 [==============================] - 0s 87us/step - loss: 0.1452 - val_loss: 0.157
5
Epoch 19/100
300/300 [==============================] - 0s 83us/step - loss: 0.1381 - val_loss: 0.149
7
Epoch 20/100
300/300 [==============================] - 0s 90us/step - loss: 0.1310 - val_loss: 0.142
2
Epoch 21/100
300/300 [==============================] - 0s 77us/step - loss: 0.1238 - val_loss: 0.134
8
Epoch 22/100
300/300 [==============================] - 0s 87us/step - loss: 0.1166 - val_loss: 0.127
6
Epoch 23/100
300/300 [==============================] - 0s 90us/step - loss: 0.1098 - val_loss: 0.120
```

```
4
Epoch 24/100
300/300 [==============================] - 0s 97us/step - loss: 0.1029 - val_loss: 0.113
4
Epoch 25/100
300/300 [==============================] - 0s 93us/step - loss: 0.0960 - val_loss: 0.106
4
Epoch 26/100
300/300 [==============================] - 0s 100us/step - loss: 0.0893 - val_loss: 0.09
97
Epoch 27/100
300/300 [==============================] - 0s 103us/step - loss: 0.0827 - val_loss: 0.09
31
Epoch 28/100
300/300 [==============================] - 0s 90us/step - loss: 0.0766 - val_loss: 0.086
7
Epoch 29/100
300/300 [==============================] - 0s 80us/step - loss: 0.0707 - val_loss: 0.080
4
Epoch 30/100
300/300 [==============================] - 0s 83us/step - loss: 0.0652 - val_loss: 0.074
2
Epoch 31/100
300/300 [==============================] - 0s 67us/step - loss: 0.0599 - val_loss: 0.068
3
Epoch 32/100
300/300 [==============================] - 0s 100us/step - loss: 0.0549 - val_loss: 0.06
27
Epoch 33/100
300/300 [==============================] - 0s 100us/step - loss: 0.0501 - val_loss: 0.05
72
Epoch 34/100
300/300 [==============================] - 0s 110us/step - loss: 0.0453 - val_loss: 0.05
18
Epoch 35/100
300/300 [==============================] - 0s 80us/step - loss: 0.0411 - val_loss: 0.046
6
Epoch 36/100
300/300 [==============================] - 0s 103us/step - loss: 0.0377 - val_loss: 0.04
16
Epoch 37/100
300/300 [==============================] - 0s 87us/step - loss: 0.0347 - val_loss: 0.037
5
Epoch 38/100
300/300 [==============================] - 0s 110us/step - loss: 0.0321 - val_loss: 0.03
42
Epoch 39/100
300/300 [==============================] - 0s 90us/step - loss: 0.0300 - val_loss: 0.031
7
Epoch 40/100
300/300 [==============================] - 0s 87us/step - loss: 0.0284 - val_loss: 0.029
9
Epoch 41/100
300/300 [==============================] - 0s 97us/step - loss: 0.0271 - val_loss: 0.028
5
Epoch 42/100
300/300 [==============================] - 0s 110us/step - loss: 0.0261 - val_loss: 0.02
75
Epoch 43/100
300/300 [==============================] - 0s 103us/step - loss: 0.0250 - val_loss: 0.02
68
```

```
Epoch 44/100
300/300 [==============================] - 0s 107us/step - loss: 0.0239 - val_loss: 0.02
62
Epoch 45/100
300/300 [==============================] - 0s 87us/step - loss: 0.0226 - val_loss: 0.025
9
Epoch 46/100
300/300 [==============================] - 0s 70us/step - loss: 0.0214 - val_loss: 0.025
9
Epoch 47/100
300/300 [==============================] - 0s 83us/step - loss: 0.0205 - val_loss: 0.025
9
Epoch 48/100
300/300 [==============================] - 0s 70us/step - loss: 0.0198 - val_loss: 0.025
5
Epoch 49/100
300/300 [==============================] - 0s 67us/step - loss: 0.0192 - val_loss: 0.024
9
Epoch 50/100
300/300 [==============================] - 0s 77us/step - loss: 0.0187 - val_loss: 0.023
9
Epoch 51/100
300/300 [==============================] - 0s 73us/step - loss: 0.0183 - val_loss: 0.023
2
Epoch 52/100
300/300 [==============================] - 0s 97us/step - loss: 0.0179 - val_loss: 0.023
0
Epoch 53/100
300/300 [==============================] - 0s 97us/step - loss: 0.0176 - val_loss: 0.022
8
Epoch 54/100
300/300 [==============================] - 0s 97us/step - loss: 0.0174 - val_loss: 0.022
6
Epoch 55/100
300/300 [==============================] - 0s 77us/step - loss: 0.0170 - val_loss: 0.022
2
Epoch 56/100
300/300 [==============================] - 0s 63us/step - loss: 0.0168 - val_loss: 0.021
9
Epoch 57/100
300/300 [==============================] - 0s 90us/step - loss: 0.0166 - val_loss: 0.021
5
Epoch 58/100
300/300 [==============================] - 0s 73us/step - loss: 0.0164 - val_loss: 0.021
3
Epoch 59/100
300/300 [==============================] - 0s 97us/step - loss: 0.0161 - val_loss: 0.021
1
Epoch 60/100
300/300 [==============================] - 0s 77us/step - loss: 0.0160 - val_loss: 0.020
9
Epoch 61/100
300/300 [==============================] - 0s 97us/step - loss: 0.0158 - val_loss: 0.020
7
Epoch 62/100
300/300 [==============================] - 0s 63us/step - loss: 0.0156 - val_loss: 0.020
3
Epoch 63/100
300/300 [==============================] - 0s 90us/step - loss: 0.0154 - val_loss: 0.020
1
Epoch 64/100
```

```
300/300 [==============================] - 0s 87us/step - loss: 0.0153 - val_loss: 0.019
9
Epoch 65/100
300/300 [==============================] - 0s 97us/step - loss: 0.0151 - val_loss: 0.019
8
Epoch 66/100
300/300 [==============================] - 0s 77us/step - loss: 0.0149 - val_loss: 0.019
6
Epoch 67/100
300/300 [==============================] - 0s 90us/step - loss: 0.0148 - val_loss: 0.019
3
Epoch 68/100
300/300 [==============================] - 0s 77us/step - loss: 0.0146 - val_loss: 0.019
1
Epoch 69/100
300/300 [==============================] - 0s 63us/step - loss: 0.0145 - val_loss: 0.019
0
Epoch 70/100
300/300 [==============================] - 0s 63us/step - loss: 0.0143 - val_loss: 0.018
7
Epoch 71/100
300/300 [==============================] - 0s 67us/step - loss: 0.0141 - val_loss: 0.018
4
Epoch 72/100
300/300 [==============================] - 0s 60us/step - loss: 0.0140 - val_loss: 0.018
2
Epoch 73/100
300/300 [==============================] - 0s 67us/step - loss: 0.0138 - val_loss: 0.018
0
Epoch 74/100
300/300 [==============================] - 0s 63us/step - loss: 0.0136 - val_loss: 0.017
8
Epoch 75/100
300/300 [==============================] - 0s 73us/step - loss: 0.0135 - val_loss: 0.017
6
Epoch 76/100
300/300 [==============================] - 0s 60us/step - loss: 0.0133 - val_loss: 0.017
4
Epoch 77/100
300/300 [==============================] - 0s 57us/step - loss: 0.0131 - val_loss: 0.017
1
Epoch 78/100
300/300 [==============================] - 0s 60us/step - loss: 0.0129 - val_loss: 0.016
8
Epoch 79/100
300/300 [==============================] - 0s 60us/step - loss: 0.0127 - val_loss: 0.016
7
Epoch 80/100
300/300 [==============================] - 0s 63us/step - loss: 0.0126 - val_loss: 0.016
5
Epoch 81/100
300/300 [==============================] - 0s 73us/step - loss: 0.0124 - val_loss: 0.016
2
Epoch 82/100
300/300 [==============================] - 0s 73us/step - loss: 0.0122 - val_loss: 0.015
9
Epoch 83/100
300/300 [==============================] - 0s 60us/step - loss: 0.0121 - val_loss: 0.015
6
Epoch 84/100
300/300 [==============================] - 0s 67us/step - loss: 0.0119 - val_loss: 0.015
```

```
5
Epoch 85/100
300/300 [==============================] - 0s 67us/step - loss: 0.0118 - val_loss: 0.015
4
Epoch 86/100
300/300 [==============================] - 0s 63us/step - loss: 0.0116 - val_loss: 0.015
2
Epoch 87/100
300/300 [==============================] - 0s 83us/step - loss: 0.0114 - val_loss: 0.014
9
Epoch 88/100
300/300 [==============================] - 0s 70us/step - loss: 0.0113 - val_loss: 0.014
7
Epoch 89/100
300/300 [==============================] - 0s 80us/step - loss: 0.0111 - val_loss: 0.014
4
Epoch 90/100
300/300 [==============================] - 0s 63us/step - loss: 0.0109 - val_loss: 0.014
2
Epoch 91/100
300/300 [==============================] - 0s 63us/step - loss: 0.0107 - val_loss: 0.014
2
Epoch 92/100
300/300 [==============================] - 0s 73us/step - loss: 0.0105 - val_loss: 0.014
0
Epoch 93/100
300/300 [==============================] - 0s 63us/step - loss: 0.0103 - val_loss: 0.013
6
Epoch 94/100
300/300 [==============================] - 0s 67us/step - loss: 0.0101 - val_loss: 0.013
4
Epoch 95/100
300/300 [==============================] - 0s 77us/step - loss: 0.0099 - val_loss: 0.013
1
Epoch 96/100
300/300 [==============================] - 0s 67us/step - loss: 0.0097 - val_loss: 0.012
7
Epoch 97/100
300/300 [==============================] - 0s 60us/step - loss: 0.0095 - val_loss: 0.012
5
Epoch 98/100
300/300 [==============================] - 0s 63us/step - loss: 0.0093 - val_loss: 0.012
4
Epoch 99/100
300/300 [==============================] - 0s 103us/step - loss: 0.0091 - val_loss: 0.01
22
Epoch 100/100
300/300 [==============================] - 0s 77us/step - loss: 0.0089 - val_loss: 0.011
8
```

Model Evaluation

In [23]:

```
epochs_hist.history.keys()
```

Out[23]:

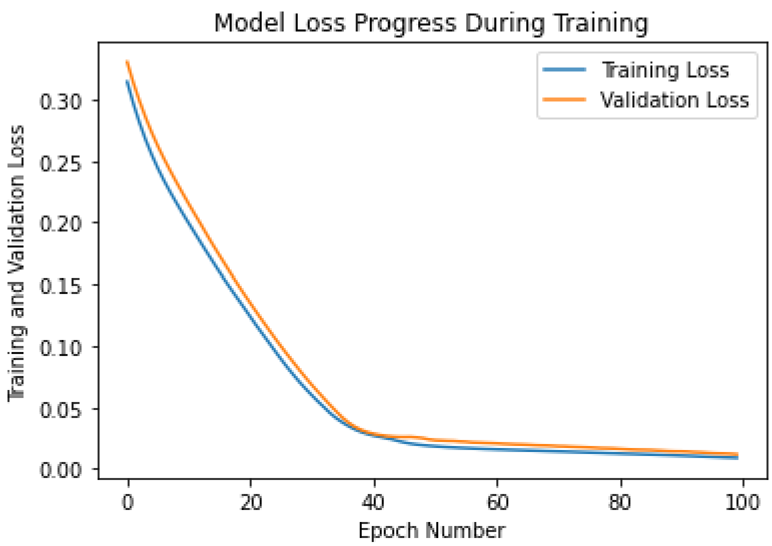
```
dict_keys(['val_loss', 'loss'])
```

In [29]:

```
plt.plot(epochs_hist.history['loss'])
```

```
plt.plot(epochs_hist.history['val_loss'])
plt.title('Model Loss Progress During Training')
plt.ylabel('Training and Validation Loss')
plt.xlabel('Epoch Number')
plt.legend(['Training Loss', 'Validation Loss'])
```

Out[29]: <matplotlib.legend.Legend at 0x22b13a7d488>



In [30]:

```
# Gender, Age, Annual Salary, Credit Card Debt, Net Worth
X_testing = np.array([[1, 50, 50000, 10000, 600000]])
y_predict = model.predict(X_test)
```

In [31]:

```
print('Expected Purchase Amount', y_predict)
```

```
Expected Purchase Amount [[34385.254]]
```

In [ ]: