Internship Report

Car Sales Purchase Prediction

Name: Ankita Gupta

Course: ML1119

Duration: 2 Weeks

Problem Statement: You are working as a car salesman and you would like to develop a Machine Learning model to predict the total amount that customers are willing to pay.

The model should predict: Car Purchase Amount

Github link: https://github.com/Ankita30-ui/Car-Sales-Purchase-Prediction

Dataset used

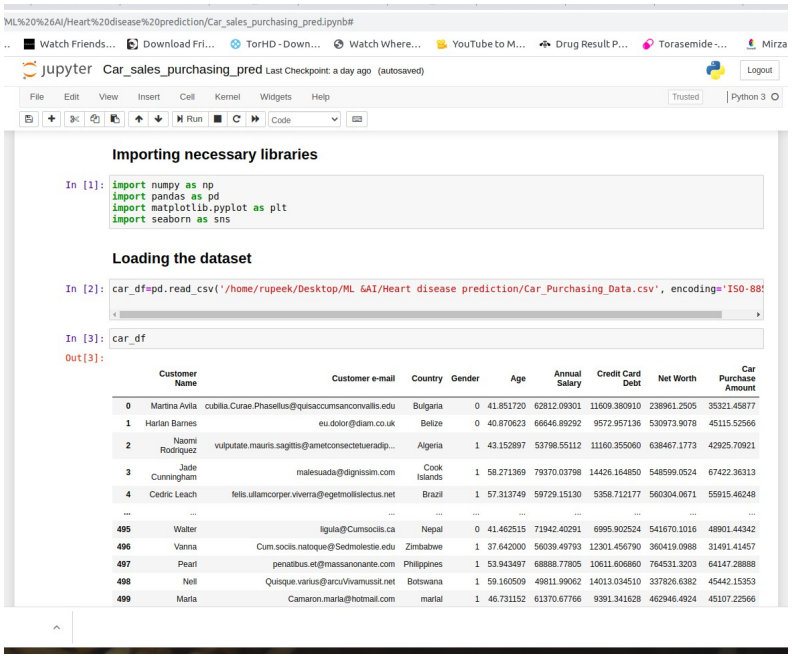
The data source used for this project is car.csv The car.csv Data Set contains attributes like: Customer Name, Customer e-mail, Country, Gender, Age, Annual Salary, Credit Card Debt, Net Worth.

Link: https://drive.google.com/drive/folders/1Dzj0gD6irtFA97BkBlpd3lnwWRPzHBHp?usp=sharing

Applying algorithms

Sequential modelling

1.Importing the libraries:



ML%20%26AI/Heart%20disease%20prediction/Car_sales_purchasing_pred.ipynb#

Watch Friends... Download Fri... TorHD - Down... Watch Where... YouTube to M... Drug Result P... Torasemide -... Mirza

jupyter Car_sales_purchasing_pred Last Checkpoint: a day ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

Importing necessary libraries

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

Loading the dataset

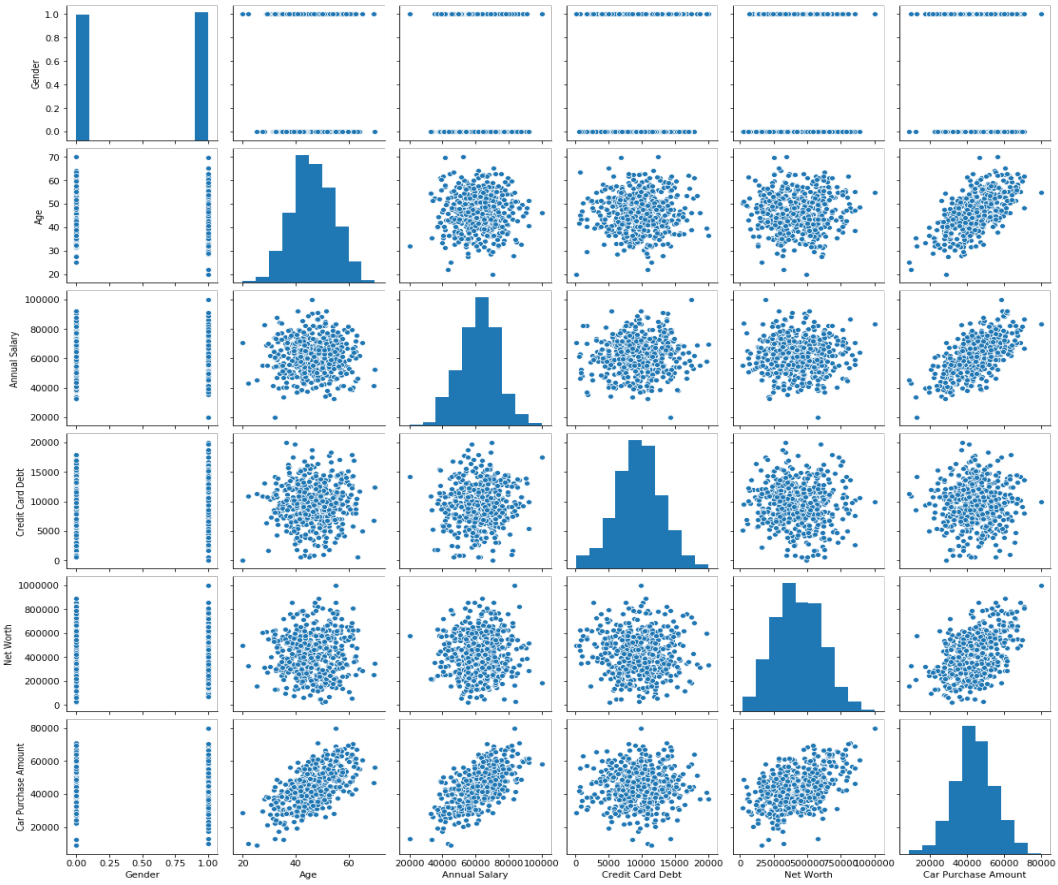
In [2]:
```
car_df=pd.read_csv('/home/rupeek/Desktop/ML &AI/Heart disease prediction/Car_Purchasing_Data.csv', encoding='ISO-885
```

In [3]:
```
car_df
```

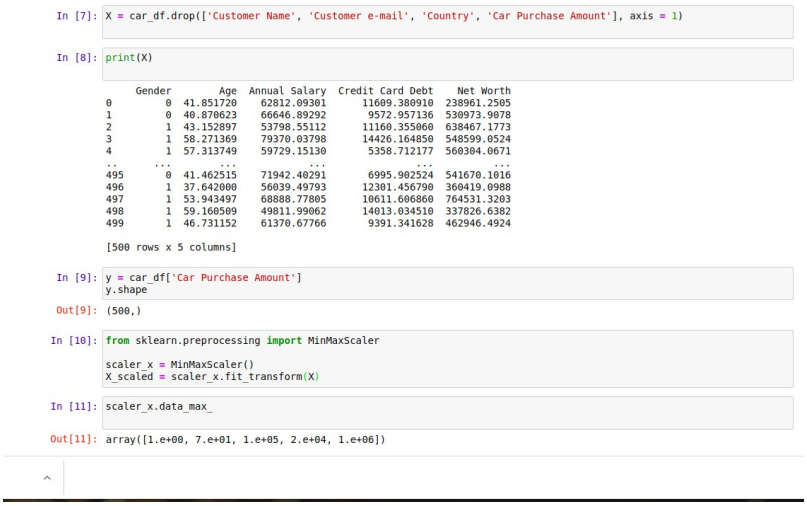
Out[3]:

	Customer Name	Customer e-mail	Country	Gender	Age	Annual Salary	Credit Card Debt	Net Worth	Car Purchase Amount
0	Martina Avila	cubilia.Curae.Phasellus@quisaccumsanconvallis.edu	Bulgaria	0	41.851720	62812.09301	11609.380910	238961.2505	35321.45877
1	Harlan Barnes	eu.dolor@diam.co.uk	Belize	0	40.870623	66646.89292	9572.957136	530973.9078	45115.52566
2	Naomi Rodriquez	vulputate.mauris.sagittis@ametconsectetueradip...	Algeria	1	43.152897	53798.55112	11160.355060	638467.1773	42925.70921
3	Jade Cunningham	malesuada@dignissim.com	Cook Islands	1	58.271369	79370.03798	14426.164850	548599.0524	67422.36313
4	Cedric Leach	felis.ullamcorper.viverra@egetmollislectus.net	Brazil	1	57.313749	59729.15130	5358.712177	560304.0671	55915.46248
...	...	...	...	...	...	...	...	...	...
495	Walter	ligula@Cumsociis.ca	Nepal	0	41.462515	71942.40291	6995.902524	541670.1016	48901.44342
496	Vanna	Cum.sociis.natoque@Sedmolestie.edu	Zimbabwe	1	37.642000	56039.49793	12301.456790	360419.0988	31491.41457
497	Pearl	penatibus.et@massanonante.com	Philippines	1	53.943497	68888.77805	10611.606860	764531.3203	64147.28888
498	Nell	Quisque.varius@arcuVivamussit.net	Botswana	1	59.160509	49811.99062	14013.034510	337826.6382	45442.15353
499	Marla	Camaron.marla@hotmail.com	marlal	1	46.731152	61370.67766	9391.341628	462946.4924	45107.22566

3. Visualizing data:

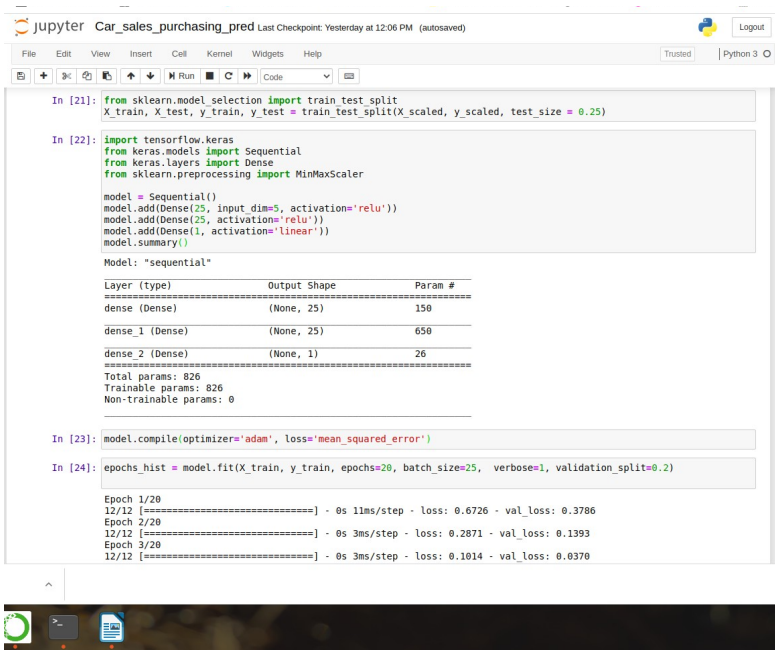


3. creating testing and training dataset



```
In [7]: X = car_df.drop(['Customer Name', 'Customer e-mail', 'Country', 'Car Purchase Amount'], axis = 1)

In [8]: print(X)

     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
..      ...        ...            ...               ...          ...
495       0  41.462515    71942.40291        6995.902524  541670.1016
496       1  37.642000    56039.49793       12301.456790  360419.0988
497       1  53.943497    68888.77805       10611.606860  764531.3203
498       1  59.160509    49811.99062       14013.034510  337826.6382
499       1  46.731152    61370.67766        9391.341628  462946.4924

[500 rows x 5 columns]

In [9]: y = car_df['Car Purchase Amount']
        y.shape

Out[9]: (500,)

In [10]: from sklearn.preprocessing import MinMaxScaler

         scaler_x = MinMaxScaler()
         X_scaled = scaler_x.fit_transform(X)

In [11]: scaler_x.data_max_

Out[11]: array([1.e+00, 7.e+01, 1.e+05, 2.e+04, 1.e+06])
```

4. Training the model



jupyter Car_sales_purchasing_pred Last Checkpoint: Yesterday at 12:06 PM (autosaved)

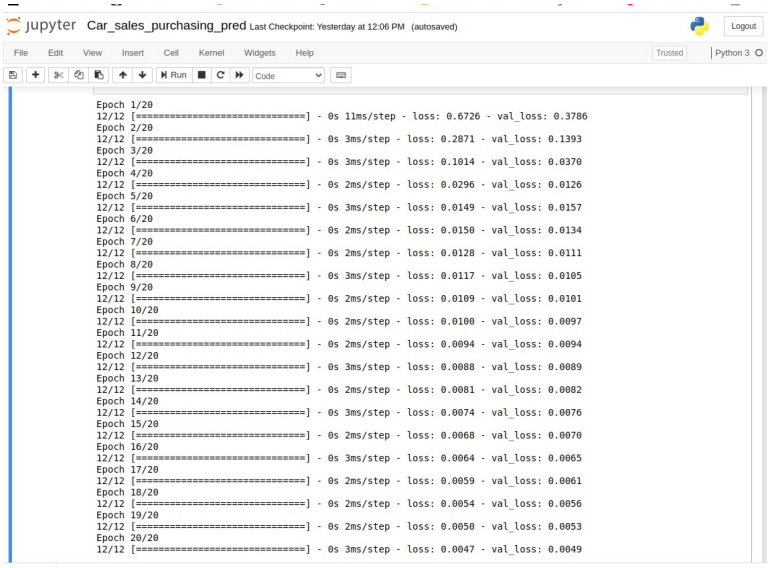
Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
In [21]: 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 = 0.25)

In [22]: import tensorflow.keras
         from keras.models import Sequential
         from keras.layers import Dense
         from sklearn.preprocessing import MinMaxScaler

         model = Sequential()
         model.add(Dense(25, input_dim=5, activation='relu'))
         model.add(Dense(25, activation='relu'))
         model.add(Dense(1, activation='linear'))
         model.summary()

Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 25)                150
_________________________________________________________________
dense_1 (Dense)              (None, 25)                650
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 26
=================================================================
Total params: 826
Trainable params: 826
Non-trainable params: 0
_________________________________________________________________

In [23]: model.compile(optimizer='adam', loss='mean_squared_error')

In [24]: epochs_hist = model.fit(X_train, y_train, epochs=20, batch_size=25,  verbose=1, validation_split=0.2)

Epoch 1/20
12/12 [==============================] - 0s 11ms/step - loss: 0.6726 - val_loss: 0.3786
Epoch 2/20
12/12 [==============================] - 0s 3ms/step - loss: 0.2871 - val_loss: 0.1393
Epoch 3/20
12/12 [==============================] - 0s 3ms/step - loss: 0.1014 - val_loss: 0.0370
```



5. Evaluating the model:

6. Final output: