Car price prediction

: Linear Algebra for Data Science

Team member

ณัฐวิทย์	จันทร์เจริญ	65056035
พรรษมน	มะโนมัย	65056065
ศุภวิชญ์	รุจิเมธาภาส	65056086
ภัทรพล	พูนสมบัติ	65056069

Source Code

https://github.com/natthawit-jan/car_price_prediction_proj

Open in Colab

Data

An automobile company wants to predict the car price that the customers are willing to pay for a new car.

Given a few customer features such as age, annual salary and net worth, we can build a linear regression model to predict the expected car price.

Data Source: https://www.kaggle.com/datasets/dev0914sharma/car-purchasing-model

Import Library

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

Import Data

```
url = r'https://raw.githubusercontent.com/natthawit-jan/car_price_prediction_proj/master/Car_Purchasing_Data.csv'
df = pd.read_csv(url)
```

```
df.head()
```

	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	USA	0	42	62812.09301	11609.380910	238961.2505	35321.45877
1	Harlan Barnes	eu.dolor@diam.co.uk	USA	0	41	66646.89292	9572.957136	530973.9078	45115.52566
2	Naomi Rodriquez	vulputate.mauris.sagittis@ametconsectetueradip...	USA	1	43	53798.55112	11160.355060	638467.1773	42925.70921
3	Jade Cunningham	malesuada@dignissim.com	USA	1	58	79370.03798	14426.164850	548599.0524	67422.36313
4	Cedric Leach	felis.ullamcorper.viverra@egetmollislectus.net	USA	1	57	59729.15130	5358.712177	560304.0671	55915.46248

Check Data Missing

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 9 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Customer Name        500 non-null    object 
 1   Customer e-mail      500 non-null    object 
 2   Country              500 non-null    object 
 3   Gender               500 non-null    int64  
 4   Age                  500 non-null    int64  
 5   Annual Salary        500 non-null    float64
 6   Credit Card Debt     500 non-null    float64
 7   Net Worth            500 non-null    float64
 8   Car Purchase Amount  500 non-null    float64
dtypes: float64(4), int64(2), object(3)
memory usage: 35.3+ KB
```

Exploratory Data Analysis (EDA)

```
df.describe()
```

	Gender	Age	Annual Salary	Credit Card Debt	Net Worth	Car Purchase Amount
count	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000
mean	0.506000	46.224000	62127.239608	9607.645049	431475.713625	44209.799218
std	0.500465	7.990339	11703.378228	3489.187973	173536.756340	10773.178744
min	0.000000	20.000000	20000.000000	100.000000	20000.000000	9000.000000
25%	0.000000	41.000000	54391.977195	7397.515792	299824.195900	37629.896040
50%	1.000000	46.000000	62915.497035	9655.035568	426750.120650	43997.783390
75%	1.000000	52.000000	70117.862005	11798.867487	557324.478725	51254.709517
max	1.000000	70.000000	100000.000000	20000.000000	1000000.000000	80000.000000

Data Preparation

```
df.rename(columns={'Gender': 'Is_Male'}, inplace=True)
```

```
df.head()
```

	Customer Name	Customer e-mail	Country	Is_Male	Age	Annual Salary	Credit Card Debt	Net Worth	Car Purchase Amount
0	Martina Avila	cubilia.Curae.Phasellus@quisaccumsanconvallis.edu	USA	0	42	62812.09301	11609.380910	238961.2505	35321.45877
1	Harlan Barnes	eu.dolor@diam.co.uk	USA	0	41	66646.89292	9572.957136	530973.9078	45115.52566
2	Naomi Rodriquez	vulputate.mauris.sagittis@ametconsectetueradip...	USA	1	43	53798.55112	11160.355060	638467.1773	42925.70921
3	Jade Cunningham	malesuada@dignissim.com	USA	1	58	79370.03798	14426.164850	548599.0524	67422.36313
4	Cedric Leach	felis.ullamcorper.viverra@egetmollislectus.net	USA	1	57	59729.15130	5358.712177	560304.0671	55915.46248

Rename the column of gender to is_male, where 1 indicates male and female otherwise

Data Preparation (remove columns)

```python
## Drop Country, Customer Name and Customer e-mail columns since they are not relevant to the model
if 'Country' in df.columns:
    df.drop(columns='Country', inplace=True)
if 'Customer Name' in df.columns:
    df.drop(columns='Customer Name', inplace=True)
if 'Customer e-mail' in df.columns:
    df.drop(columns='Customer e-mail', inplace=True)
```

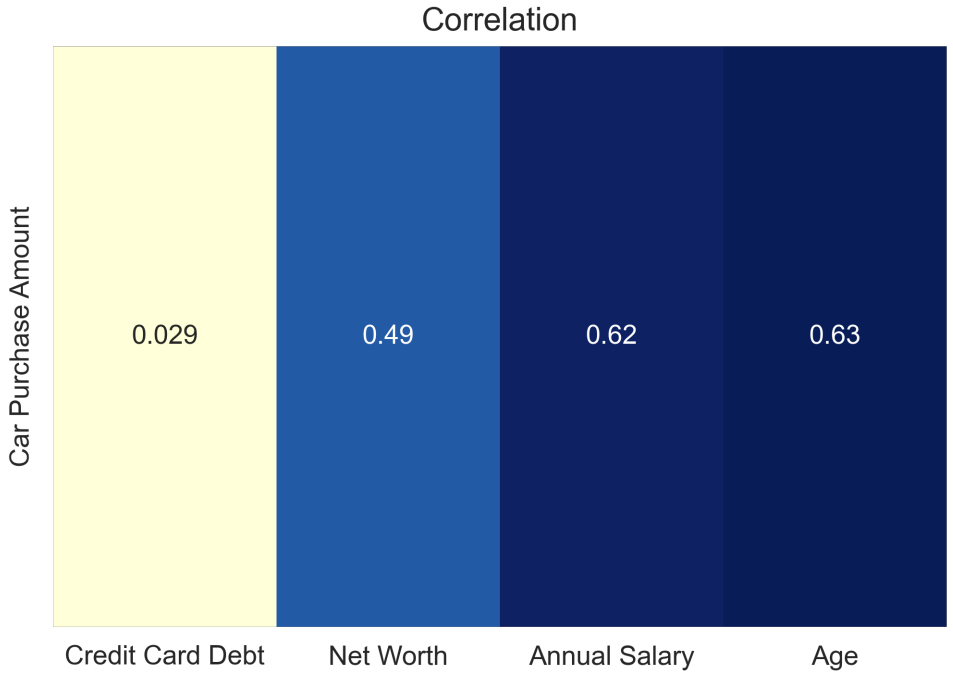
```python
df.head()
```

	Is_Male	Age	Annual Salary	Credit Card Debt	Net Worth	Car Purchase Amount
0	0	42	62812.09	11609.38	238961.25	35321.46
1	0	41	66646.89	9572.96	530973.91	45115.53
2	1	43	53798.55	11160.36	638467.18	42925.71
3	1	58	79370.04	14426.16	548599.05	67422.36
4	1	57	59729.15	5358.71	560304.07	55915.46

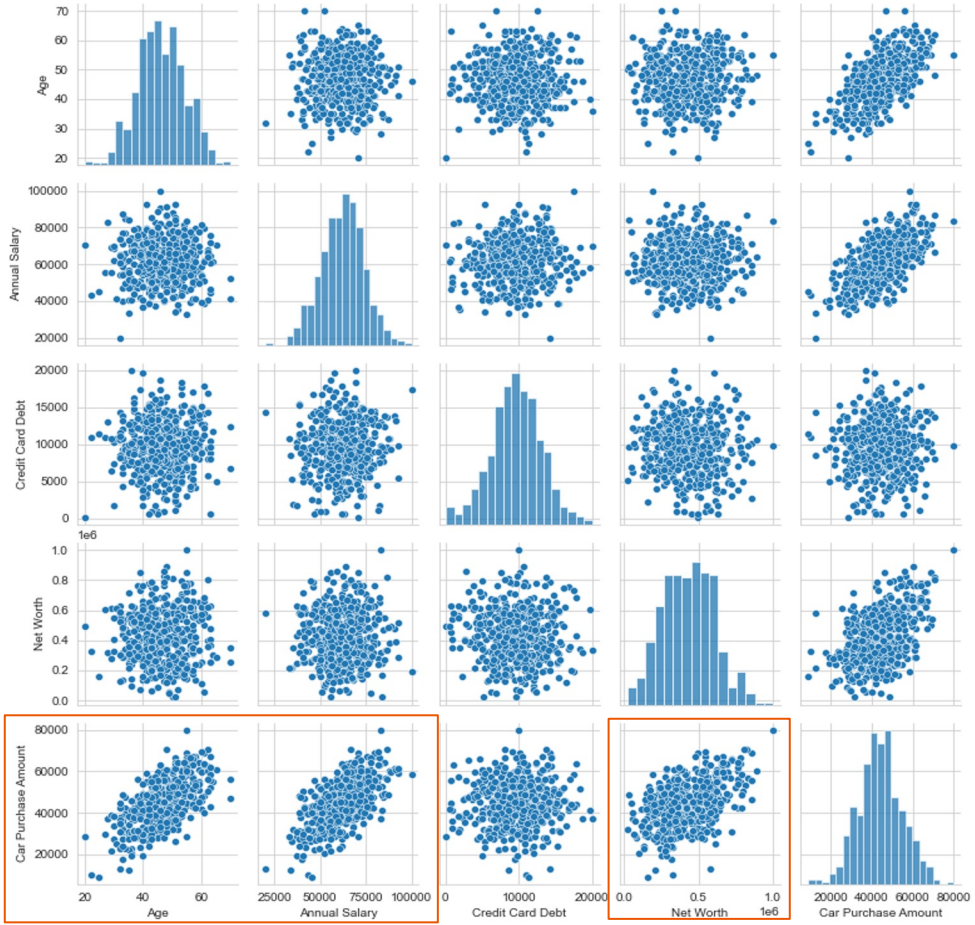
5 rows × 6 columns Open in new tab

Drop Country, Customer Name and Customer e-mail columns

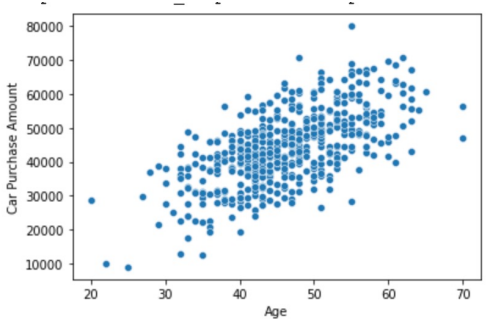
Correlation Heat Map

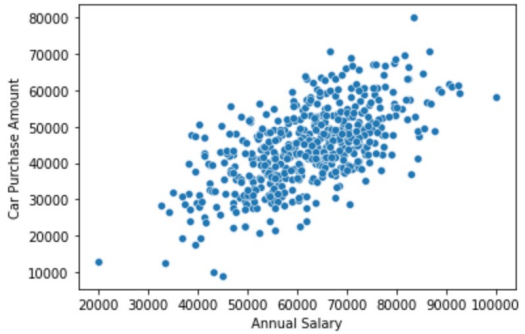


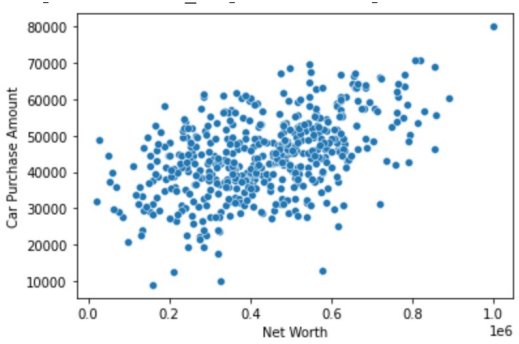
Finding Correlation between X and Y



Correlation of 3 linear features



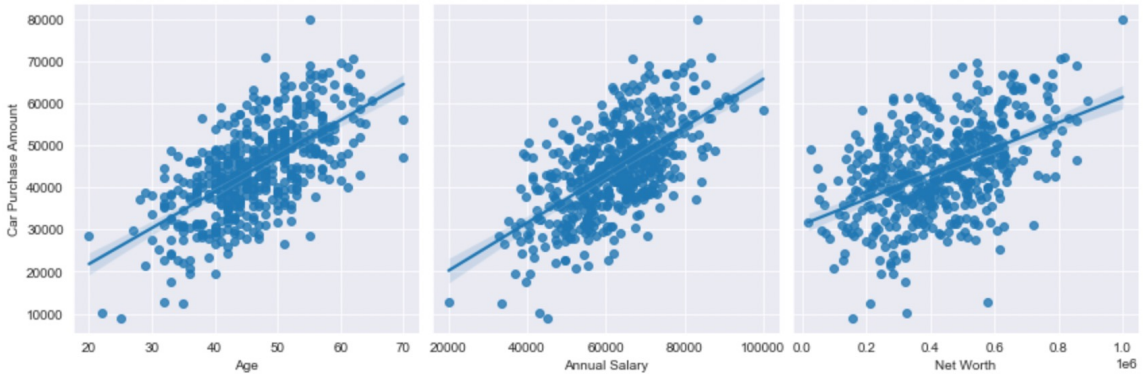




Regression plots for 3 linear features

```
x_elements = ['Age', 'Annual Salary', 'Net Worth']
y_elements = 'Car Purchase Amount'
sns.pairplot(data=df, x_vars=x_elements, y_vars=y_elements, kind="reg", height=4, )
```

```
<seaborn.axisgrid.PairGrid at 0x289c3e6b0>
```



Therefore, looking from the plots, we can see that there are mainly 3 features that look linear and can be used to train the model. ***Age*, *Net Worth* and *Annual Salary*.**

Standardize and Split Data

```
from sklearn.preprocessing import StandardScaler

S = StandardScaler()
S.fit(X)
K = StandardScaler()
K.fit(y)

X = S.transform(X)
y = K.transform(y)
```

Divide the data into train and test data. We'll use 70% for the train dataset and 30% for the test dataset.

```
[132] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.3)
```

Import Linear Regression model + train data

```
[ ]  model = LinearRegression()
     model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

Get all coefficients from the trained model

```
[ ]  print(f'The intercept b0 = {model.intercept_}')
```

```
The intercept b0 = [5.76240327e-05]
```

```
Y = S.feature_names_in_

for ind, coeff in enumerate(model.coef_.T):
    print(f'The value of b{ind+1} {Y[ind]} = {coeff}')
    ind += 1
```

```
The value of b1 Age = [0.62268822]
The value of b2 Annual Salary = [0.61035803]
The value of b3 Net Worth = [0.46688715]
```

The regression equation

$$y = 0.00025 + 0.62(\text{Age}) + 0.61(\text{Annual Salary}) + 0.47(\text{Net Worth})$$

If `Age` goes up by 1 unit, the car price will also go up by 0.62

If `Annual Salary` goes up by 1 unit, the car price will also go up by 0.61

If `Net worth` goes up by 1 unit, the car price will also go up by 0.47

Predict the outcome with test data + Calculate error

```
[ ]  y_pred = model.predict(X_test)
     print('MAE:', metrics.mean_absolute_error(y_test, y_pred))
     print('MSE:', metrics.mean_squared_error(y_test, y_pred))
     print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
MAE: 0.021072551634077896
MSE: 0.0005792296471567664
RMSE: 0.024067190263027515
```

From the RMSE result, our model in general can be wrong by 0.024 (std.) (around 44,000 dollars off)

ANOVA TABLE (Analysis of Variance)

```
X_train = pd.DataFrame(X_train)
X_train.columns = S.feature_names_in_
X_With_Constant = sm.add_constant(X_train)
sm_model = sm.OLS(y_train, X_With_Constant).fit()
sm_model.summary()
```

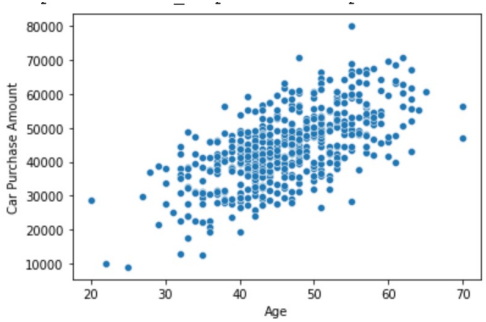
OLS Regression Results

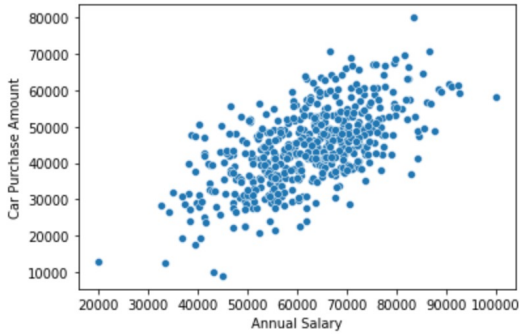
Dep. Variable:	y	**R-squared:**	1.000
Model:	OLS	**Adj. R-squared:**	1.000
Method:	Least Squares	**F-statistic:**	2.638e+05
Date:	Tue, 11 Oct 2022	**Prob (F-statistic):**	0.00
Time:	23:05:21	**Log-Likelihood:**	843.70
No. Observations:	350	**AIC:**	-1679.
Df Residuals:	346	**BIC:**	-1664.
Df Model:	3		
Covariance Type:	nonrobust		

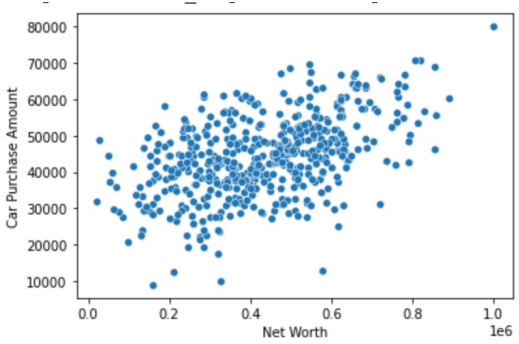
	coef	std err	t	P>|t|	[0.025	0.975]
const	5.762e-05	0.001	0.049	0.961	-0.002	0.002
Age	0.6227	0.001	546.302	0.000	0.620	0.625
Annual Salary	0.6104	0.001	533.129	0.000	0.608	0.613
Net Worth	0.4669	0.001	382.242	0.000	0.464	0.469

Omnibus:	151.771	**Durbin-Watson:**	1.913
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	19.717
Skew:	-0.062	**Prob(JB):**	5.23e-05
Kurtosis:	1.844	**Cond. No.**	1.13

Correlation of 3 linear features







Residual vs Fitted