การทำนายราคารถยนต์ในอนาคต

โดย

63010022 นายกกฤต รุ่งโรจน์กิจกุล
63010042 นายกฤษติภูมิ เทียนงาม
63010052 นายก้องเกียรติ ชุนงาม
63010332 นายณัฐพล จำปานนท์

โครงงานนี้เป็นส่วนหนึ่งของการศึกษา

วิชา 01076032 ELEMENTARY DIFFERENTIAL EQUATIONS AND

LINEAR ALGEBRA

ปีการศึกษา 2564

บทคัดย่อ

โครงงานเรื่อง การทำนายราคารถยนต์ในอนาคต จัดทำขึ้นเพื่อแก้ปัญหาว่าถ้าหากสนใจรถรุ่นใดรุ่นหนึ่งแล้วในอนาคตรถนั้นจะมีราคาเท่าใดถ้าถูกขายแบบมือสองหรือราคาที่ลดตามกาลเวลา โดยการนำความรู้ในเรื่องของ Vector และ Matrix มาประยุกต์เพื่อทำนายหรือคาดเดาราคาในอนาคต โดยผ่านกระบวนการคำนวณต่างๆ จนได้โปรแกรมที่สามารถรับข้อมูลจากผู้ใช้งานแล้วแสดงราคาของรถในอนาคตออกมา

คำสำคัญ: Vector, Matrix, ทำนาย

สารบัญ

บทที่ 1

บทนำ

1.1 ที่มาของโครงงาน

จากคำถามทั่วไปตามสื่อโซเชียลมีเดียต่างๆไปจนถึงบทสนทนาระหว่างครอบครัว หนึ่งในหัวข้อที่มักจะถูกหยิบมาพูดคุยกันอยู่บ่อยครั้งคือระหว่าง “บ้าน” และ “รถ” จะเลือกซื้ออะไรเป็นอย่างแรกถึงจะคุ้มค่ากว่า เป็นที่แน่นอนว่าต่างคนต่างก็มีเหตุผลของตัวเองอย่างเช่น คนที่เลือกซื้อบ้านก่อนเพราะการที่บ้านตอบโจทย์ในการใช้ชีวิตมากกว่า การมีพื้นที่ให้ได้พักผ่อนอย่างเต็มที่หรือ อีกเหตุผลคือที่ดินนั้นไม่มีค่าเสื่อมราคายิ่งเวลาผ่านไปค่าของมันก็ยิ่งเพิ่มทำให้ถ้ามีการซื้อขายก็จะคงได้กำไรแน่นอน เป็นต้น ส่วนคนที่เลือกซื้อรถนั้นอาจเป็นเพราะในการทำงานต้องเดินทางบ่อย หรือการที่มีบ้านแล้วแต่ระยะทางจากบ้านไปที่ทำงานมีระยะทางที่ไกล ทำให้การเลือกที่จะซื้อรถนั้นตอบโจทย์ในการใช้ชีวิตมากกว่า จากเหตุผลที่กล่าวมาเป็นแค่เหตุผลบ้างเหตุผลเท่านั้น ยังมีผู้คนจำนวนมากที่ถึงจะไม่เลือกซื้อบ้านแต่ก็มีค่าใช้จ่ายในชีวิตประจำวันที่มากอยู่แล้ว การที่มีความจำเป็นจะซื้อรถนั้นอาจต้องไปซื้อรถมือ 2 มาแทนแต่ถ้าไม่อยากซื้อรถมือ 2 แล้วอย่างจะทราบราคาของรถรุ่นที่สนใจในอนาคตว่าจะมีราคาเท่าใดจะสามารถทราบได้หรือไม่

จากเหตุผลและคำถามที่ได้กล่าวมาข้างต้น การที่จะสามารถทราบราคาของรถมือสองหรือราคาขอรถรุ่นที่สนใจในอนาคตได้นั้น สามารถทำได้โดยการนำข้อมูลเกี่ยวกับรถยนต์มาทำการวิเคราะห์ด้วยกระบวนการทางคณิตศาสตร์ที่เรียกว่า linear regression โดยเมื่อผ่านการคำนวณเสร็จจะได้ Model ที่สามารถทำนายราคาของรถยนต์ในอนาคตได้ ด้วยเหตุนี้ทางคณะผู้จัดทำได้ตัดสินใจทำโครงงานเรื่องนี้เพื่อให้เป็นประโยชน์แก่ผู้ที่ต้องการทราบราคาของรถยนต์และ ผู้ที่ต้องการหาข้อมูลไปใช้ในการประกอบการตัดสินใจของตนเอง

1.2 จุดประสงค์โครงงาน

1.2.1 เพื่อให้เป็นประโยชน์แก่ผู้ที่ต้องการทราบราคาของรถยนต์และผู้ที่ต้องการหาข้อมูลไปใช้ในการประกอบการตัดสินใจของตนเอง

1.2.2 ประยุกต์ใช้ความรู้ที่ได้จากการศึกษามาใช้กับการทำงานจริงและก่อประโยชน์สูงสุด

บทที่ 2

ภาพรวมการออกแบบระบบ

2.1 ภาพรวมขั้นตอนการทำงานของระบบ

ระบบจะวิเคราะห์ข้อมูลที่นำเข้ามาโดยใช้การเขียนโปรแกรมภาษา Python โดยจะทำงานที่ Google Colab

2.1.1 การนำข้อมูลเข้าสู่ระบบ

นำข้อมูลที่เป็นไฟล์ csv เข้าสู่ระบบโดยการอัปโหลดไฟล์ไว้ใน GitHub จากนั้นทำการ import ข้อมูลเข้าสู่ระบบ

2.1.2 การปรับแต่งข้อมูล

ข้อมูลที่นำเข้าสู่ระบบนั้นยังไม่สามารถนำไปวิเคราะห์ได้เพราะว่าอาจมีข้อมูลบางตัวผิดพลาด หรือ มีข้อมูลบางอย่างในชุดข้อมูลที่ไม่ได้นำมาวิเคราะห์

2.1.3 การวิเคราะห์ชุดข้อมูล

นำข้อมูลที่ได้หลังจากการปรับแต่งข้อมูลมาวิเคราะห์โดยใช้วิธีการ Linear Regression

2.1.4 Model การทำนายของโครงงาน

เมื่อวิเคราะห์ชุดข้อมูลสำเร็จจะได้ Model ที่ใช้ในการทำนายราคารถยนต์ในอนาคต โดยเมื่อนำค่าที่ได้จาก Model มาหาค่าความใกล้เคียงโดยใช้ Cosine Similarity จะมีค่าอยู่ที่ 0.9684%

2.2 รายละเอียดข้อมูลที่เกี่ยวข้อง

ชุดข้อมูลนั้นได้มากจากเว็บไซต์ Kaggle ที่ถูกเก็บเมื่อปี ค.ศ. 2018 ที่ประเทศอินเดียจำนวน 301 ชุด โดยที่ชุดข้อมูลประกอบด้วยข้อมูลต่อไปนี้

1. Name คือ ชื่อรุ่นของรถยนต์
2. Year คือ ปีที่ซื้อรถยนต์
3. Selling price คือ ราคาของรถยนต์ที่ขายจากบุคคลสู่บุคคล
4. Present price คือ ราคาของรถยนต์จากโชว์รูมรถยนต์
5. Km driven คือ ระยะทางที่รถยนต์นั้นวิ่ง
6. Fuel คือ ประเภทของเชื้อเพลิง
7. Seller type คือ ประเภทการขาย
8. Transmission คือ ประเภทของเกียร์
9. Owner คือ จำนวนเจ้าของรถก่อนหน้านี้

2.2.1 Name

```
[ ]  data["Car_Name"].value_counts()
```

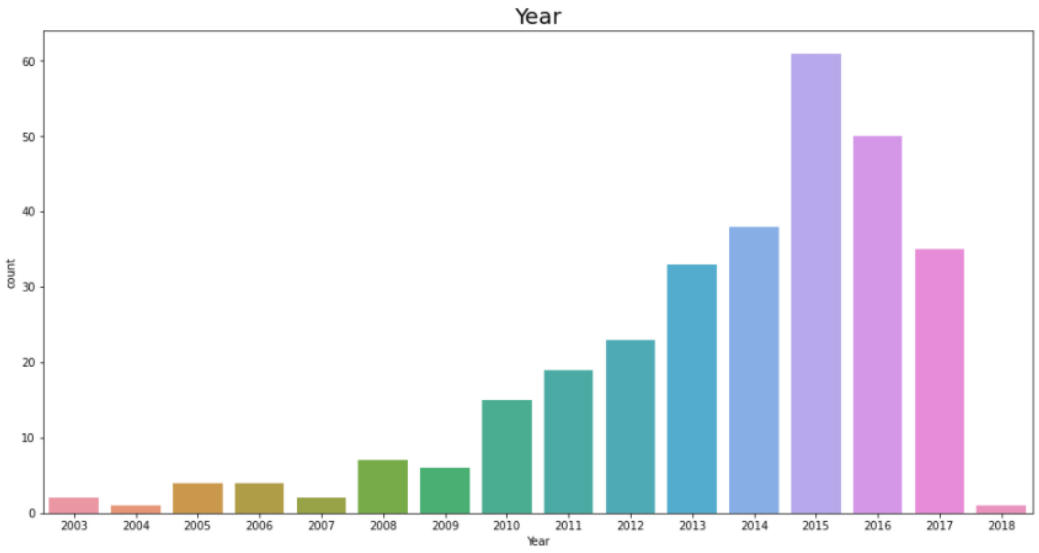
```
city                    26
corolla altis           16
verna                   14
fortuner                11
brio                    10
                        ..
Mahindra Mojo XT300      1
Honda Dream Yuga         1
TVS Sport                1
Bajaj Pulsar 135 LS      1
Honda CB Trigger         1
Name: Car_Name, Length: 98, dtype: int64
```

```
[ ]  print("Unique Car_Name values : ",len(data["Car_Name"].unique()))
```

จากข้อมูลจะเห็นว่ารถยนต์นั้นมีชื่อรุ่นที่ไม่ซ้ำกันถึง 98 ชื่อ

2.2.2 Year

```
[ ]  data["Year"].value_counts()

2015    61
2016    50
2014    38
2017    35
2013    33
2012    23
2011    19
2010    15
2008     7
2009     6
2006     4
2005     4
2007     2
2003     2
2018     1
2004     1
Name: Year, dtype: int64
```



2.2.3 Selling price

```
[ ]  data["Selling_Price"].value_counts()
```

```
0.60     8
0.45     8
5.25     7
4.50     7
4.75     6
        ..
19.99    1
4.35     1
0.80     1
0.27     1
7.20     1
Name: Selling_Price, Length: 156, dtype: int64
```

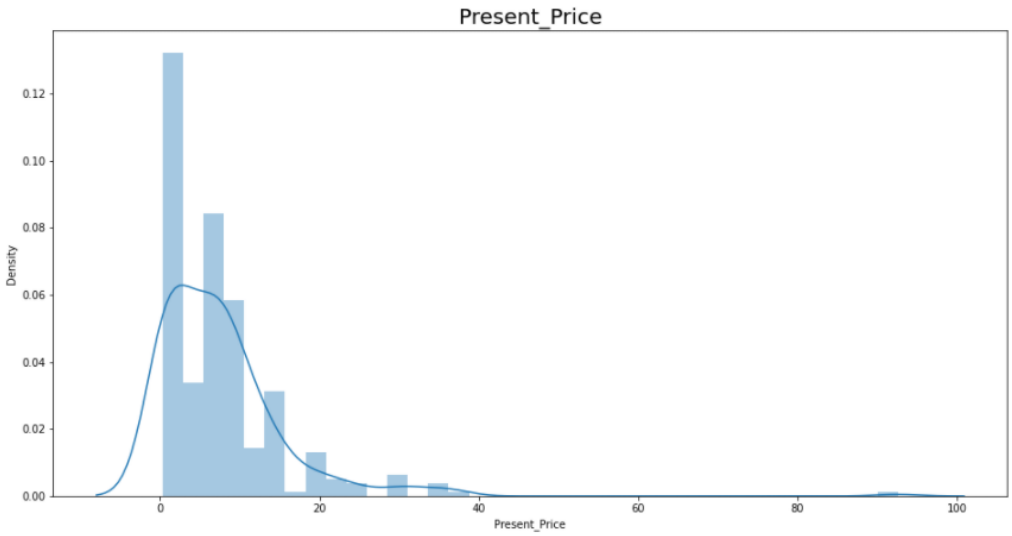
```
[ ]  data["Selling_Price"].describe()
```

```
count    301.000000
mean       4.661296
std        5.082812
min        0.100000
25%        0.900000
50%        3.600000
75%        6.000000
max       35.000000
Name: Selling_Price, dtype: float64
```

2.2.4 Present price

```
[ ] data["Present_Price"].value_counts()

9.40     15
13.60    13
5.70      8
4.43      7
1.47      7
         ..
9.29      1
92.60     1
13.70     1
1.17      1
0.65      1
Name: Present_Price, Length: 147, dtype: int64
```

```
[ ] data["Present_Price"].describe()

count    301.000000
mean       7.628472
std        8.644115
min        0.320000
25%        1.200000
50%        6.400000
75%        9.900000
max       92.600000
Name: Present_Price, dtype: float64
```



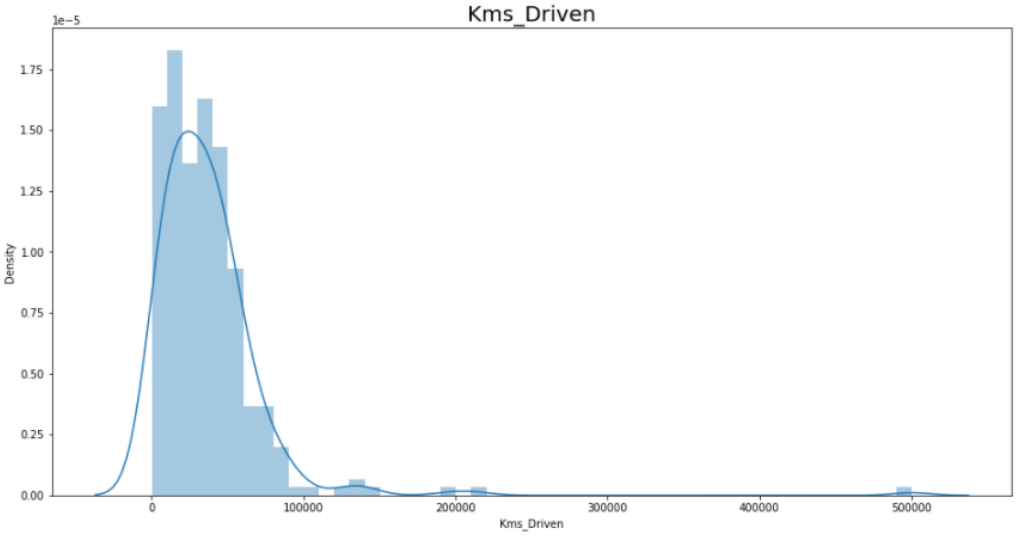
2.2.5 Km driven

```
[ ]  data["Kms_Driven"].value_counts()
```

```
15000     9
45000     9
35000     5
25000     5
50000     5
         ..
1000      1
500000    1
11800     1
5400      1
4100      1
Name: Kms_Driven, Length: 206, dtype: int64
```

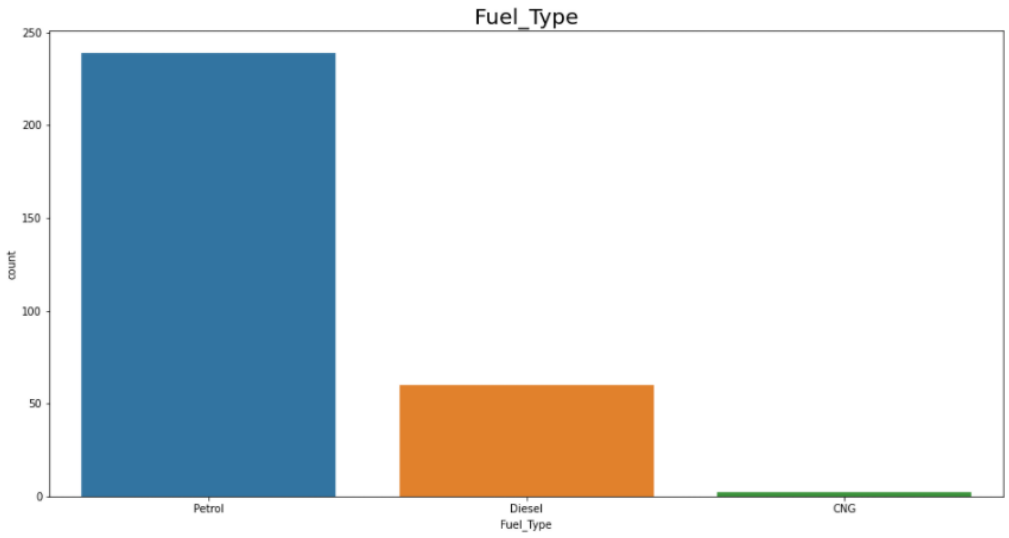
```
[ ]  data["Kms_Driven"].describe()
```

```
count       301.000000
mean      36947.205980
std       38886.883882
min         500.000000
25%       15000.000000
50%       32000.000000
75%       48767.000000
max      500000.000000
Name: Kms_Driven, dtype: float64
```



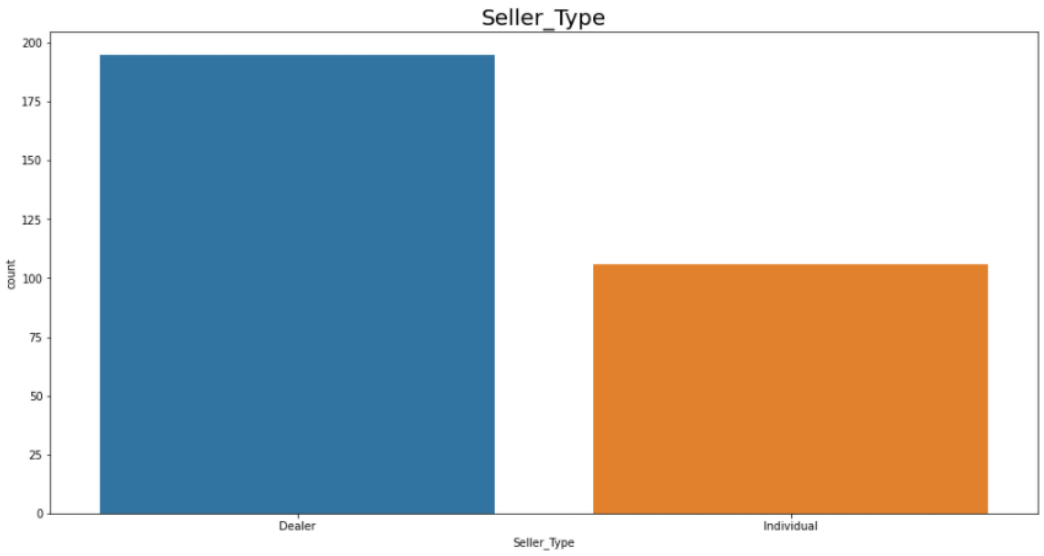
2.2.6 Fuel

```
[ ]  data["Fuel_Type"].value_counts()

     Petrol    239
     Diesel     60
     CNG         2
     Name: Fuel_Type, dtype: int64
```



2.2.7 Seller type

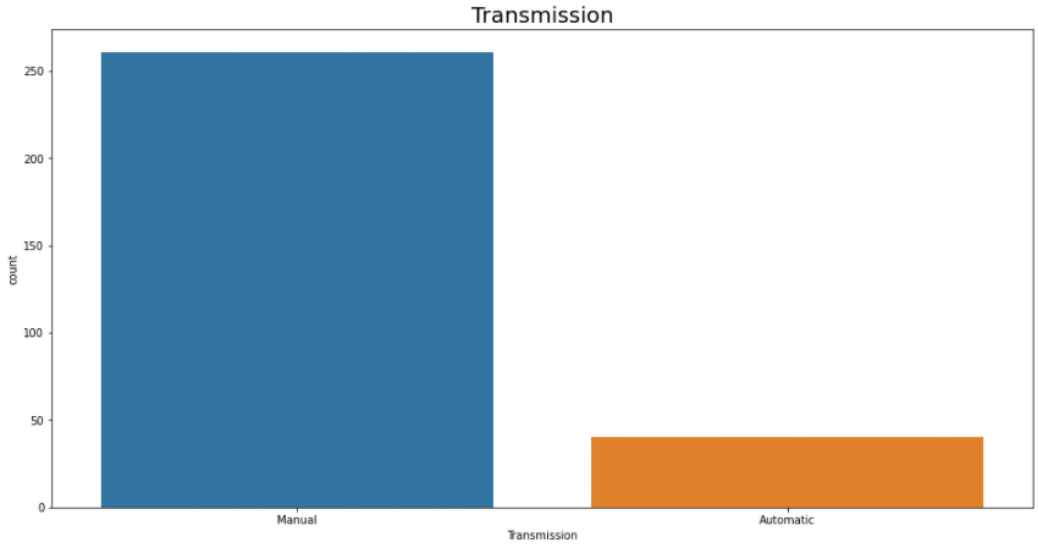
```
[ ]  data["Seller_Type"].value_counts()

     Dealer        195
     Individual    106
     Name: Seller_Type, dtype: int64
```



2.2.8 Transmission

```
[ ]  data["Transmission"].value_counts()
```

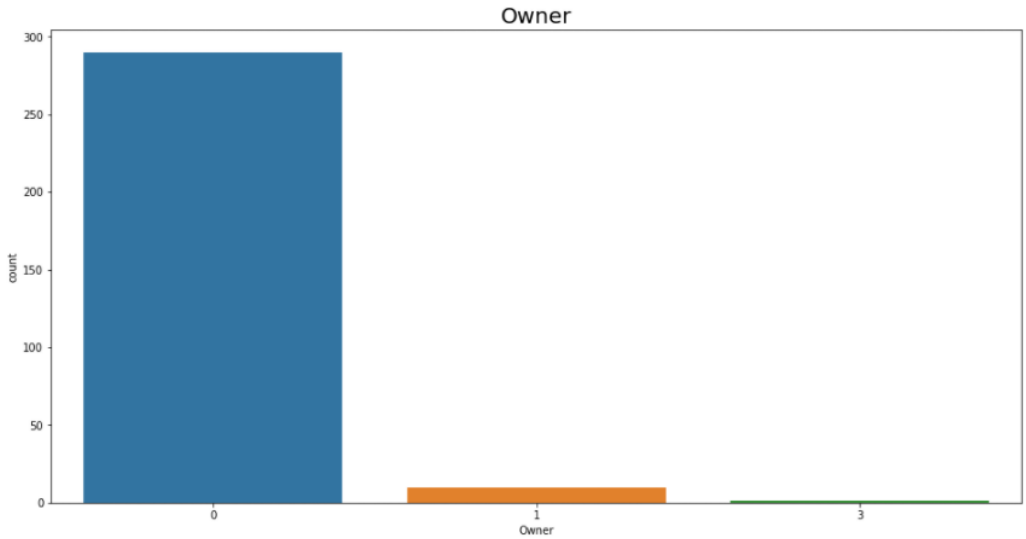
```
Manual       261
Automatic     40
Name: Transmission, dtype: int64
```



2.2.9 Owner

```
[ ]  data["Owner"].value_counts()
```

```
0    290
1     10
3      1
Name: Owner, dtype: int64
```



2.3 อธิบายขั้นตอนย่อยแต่ละขั้น

2.3.1 การนำข้อมูลเข้าสู่ระบบ

```
[ ]  url = "https://raw.githubusercontent.com/KirttiphoomEarth/Car_pre_data/main/car%20data.csv"    # last updated 2020
```

นำเข้าชุดข้อมูลจากชุดข้อมูลที่ถูกอัพโหลดไว้ใน GitHub

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

```
[ ]  #Additional library
     from sklearn.linear_model import LinearRegression
     from sklearn.model_selection import train_test_split
     from sklearn.metrics import mean_squared_error
     from sklearn.preprocessing import RobustScaler
     import math
```

เพิ่ม library ต่างๆที่ใช้ในการคำนวณในระบบ

```
[ ] data = pd.read_csv(url)
    data.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

อ่านชุดข้อมูลและแสดงข้อมูลเบื้องต้น

```
[ ] data.count()
```

```
Car_Name         301
Year             301
Selling_Price    301
Present_Price    301
Kms_Driven       301
Fuel_Type        301
Seller_Type      301
Transmission     301
Owner            301
dtype: int64
```

นับจำนวนข้อมูล

2.3.2 การปรับแต่งข้อมูล

```
[ ] data.drop('Car_Name', axis=1, inplace=True)
    data.head()
```

	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

นำข้อมูล Name ออกเพราะมีความแตกต่างกันมากเกินไป

```
[ ] data['Year'].replace({2010 : 1, 2011 : 2, 2012 : 3, 2013 : 4, 2014 : 5, 2015 : 6, 2016 : 7, 2017 : 8, 2018 : 9}, inplace=True)
    data.loc[(data['Year'] >= 2003) & (data['Year']<= 2009), 'Year'] = 0
    data['Year'].value_counts()
```

```
6    61
7    50
5    38
8    35
4    33
0    26
3    23
2    19
1    15
9     1
Name: Year, dtype: int64
```

การปรับจากปีที่ผลิตต่างๆเป็นเลขเพื่อที่ให้ง่ายต่อการคำนวณในระบบ

ถ้าอยู่ในช่วงปี ค.ศ. 2003 ถึง ค.ศ. 2009 กำหนดให้เป็น 0

ปี ค.ศ. 2010 กำหนดให้เป็น 1

ปี ค.ศ. 2011 กำหนดให้เป็น 2

ปี ค.ศ. 2012 กำหนดให้เป็น 3

ปี ค.ศ. 2013 กำหนดให้เป็น 4

ปี ค.ศ. 2014 กำหนดให้เป็น 5

ปี ค.ศ. 2015 กำหนดให้เป็น 6

ปี ค.ศ. 2016 กำหนดให้เป็น 7

ปี ค.ศ. 2017 กำหนดให้เป็น 8

ปี ค.ศ. 2018 กำหนดให้เป็น 9

```
[ ]  data.loc[(data['Kms_Driven'] >= 500) & (data['Kms_Driven'] <= 15000), 'Kms_Driven'] = 0
     data.loc[(data['Kms_Driven'] > 15000) & (data['Kms_Driven'] <= 32000), 'Kms_Driven'] = 1
     data.loc[(data['Kms_Driven'] > 32000) & (data['Kms_Driven'] <= 49000), 'Kms_Driven'] = 2
     data.loc[(data['Kms_Driven'] > 49000), 'Kms_Driven'] = 3
     data["Kms_Driven"].value_counts()
```

```
0    77
2    76
1    75
3    73
Name: Kms_Driven, dtype: int64
```

ปรับข้อมูลระยะที่รถวิ่งได้โดยกำหนดเลขในช่วงใดช่วงหนึ่งเพื่อให้ง่ายต่อการคำนวณในระบบ

ในช่วง 500 ถึง 15000 กิโล/เมตร กำหนดให้เป็น 0

ในช่วง 15000 ถึง 32000 กิโล/เมตร กำหนดให้เป็น 1

ในช่วง 32000 ถึง 49000 กิโล/เมตร กำหนดให้เป็น 2

ตั้งแต่ 49000 กิโล/เมตร กำหนดให้เป็น 3

```
[ ]  numFuel = {"Petrol":0,"Diesel":1,"CNG":2}
     data["Fuel_Type"].replace(numFuel, inplace = True)
     data["Fuel_Type"].value_counts()
```

```
0    239
1     60
2      2
Name: Fuel_Type, dtype: int64
```

การปรับข้อมูลประเภทของเชื้อเพลิง โดยกำหนดให้

เครื่องยนต์เบนซิน เป็น 0

เครื่องยนต์ดีเซล เป็น 1

เครื่องยนต์ก๊าซธรรมชาติ เป็น 2

```
[ ]  numSeller = {"Individual":0,"Dealer":1}
     data["Seller_Type"].replace(numSeller, inplace = True)
     data["Seller_Type"].value_counts()
```

```
1    195
0    106
Name: Seller_Type, dtype: int64
```

ประเภทของการขายกำหนดให้ ขายผ่าน Dealer เป็น 0 ขายผ่าน Individual เป็น 1

```
[ ]  numTrans = {"Manual":0,"Automatic":1}
     data["Transmission"].replace(numTrans, inplace = True)
     data["Transmission"].value_counts()
```

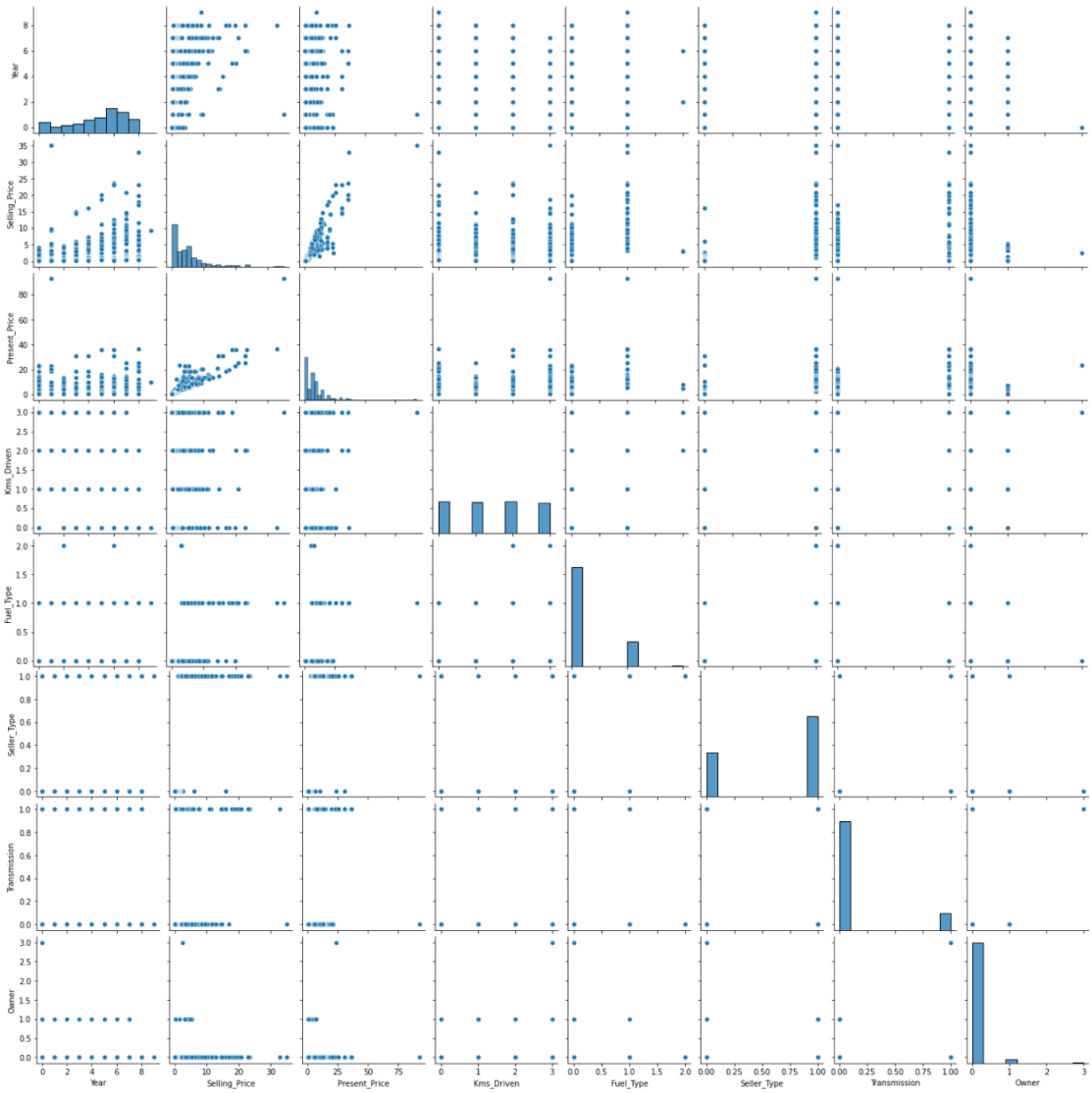
```
0    261
1     40
Name: Transmission, dtype: int64
```

ประเภทของเกียร์กำหนดให้เกียร์ประเภท Manual เป็น 0 และเกียร์ประเภท Automatic เป็น 1

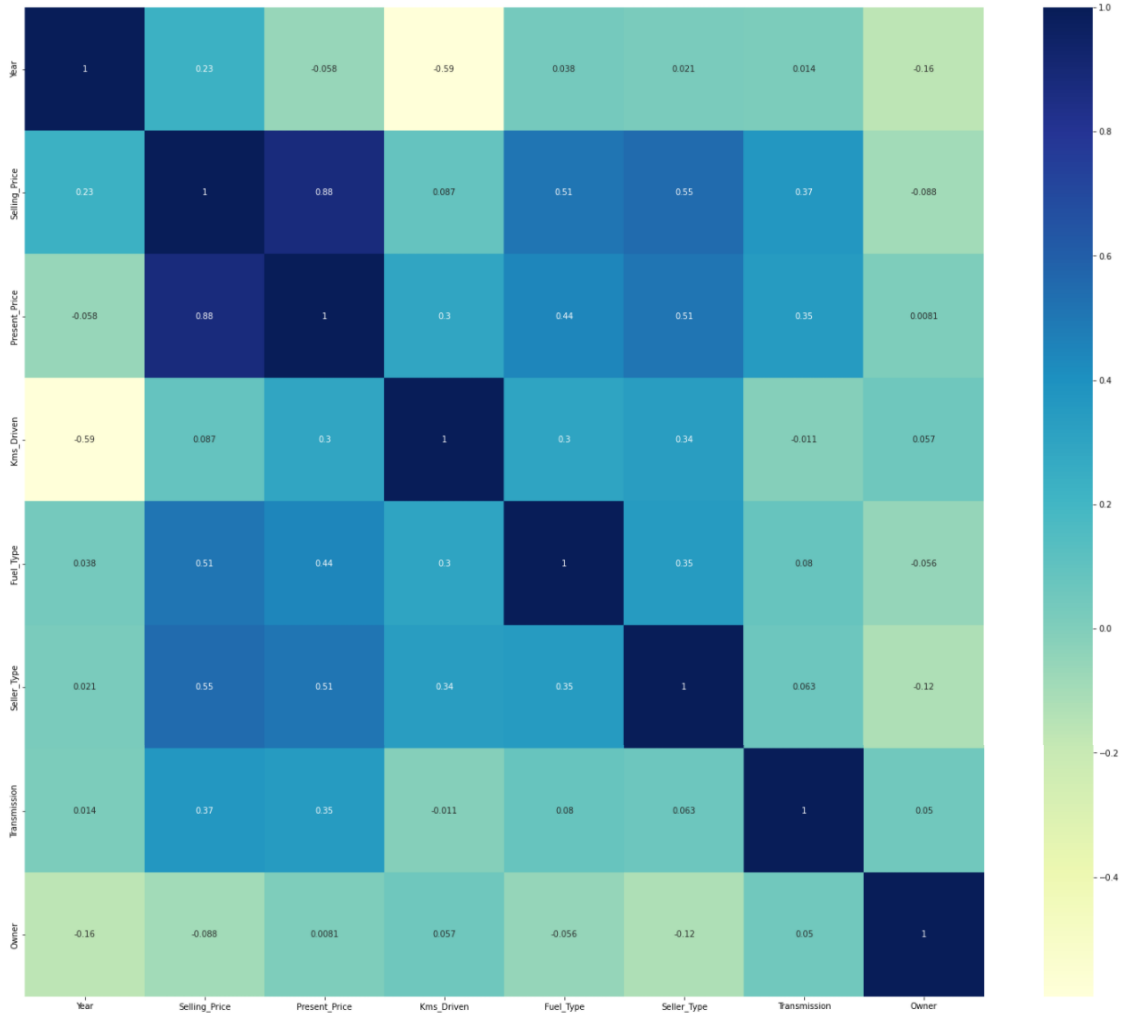
	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	5	3.35	5.59	1	0	1	0	0
1	4	4.75	9.54	2	1	1	0	0
2	8	7.25	9.85	0	0	1	0	0
3	2	2.85	4.15	0	0	1	0	0
4	5	4.60	6.87	2	1	1	0	0

ตัวอย่างชุดข้อมูลหลังจากการปรับแต่งข้อมูล

2.3.3 การวิเคราะห์ชุดข้อมูล



การเปรียบทียบข้อมูลแต่ละตัวในชุดข้อมูล



การนำชุดข้อมูลมาหา Correlations และแสดงอยู่ในรูปของ Matrix

```
[ ] #Training (Selling_Price)
    x1 = data.drop(['Selling_Price'], axis=1)        #Drop both Price makes Linear Regression score lower
    y1 = data['Selling_Price']

    x1 = RobustScaler().fit_transform(x1)

    x1Train, x1Test, y1Train, y1Test = train_test_split(x1, y1, test_size = 0.2,random_state = 42)
    x1Train.shape, x1Test.shape, y1Train.shape, y1Test.shape

    ((240, 7), (61, 7), (240,), (61,))
```

นำชุดข้อมูลมาผ่านการ Training

```
[ ] #Linear Regression (Selling_Price)
    aLR = LinearRegression()
    aLR.fit(x1Train, y1Train)
    yPredict = aLR.predict(x1Test)
    print("Linear regression score : ",aLR.score(x1Test, y1Test))
    print("Mean squared error : ", mean_squared_error(y1Test, yPredict))

    Linear regression score :  0.8592697783382605
    Mean squared error :  3.2418029156481825
```

นำชุดข้อมูลที่ผ่านการ Training มาเข้ากระบวนการ Linear Regression

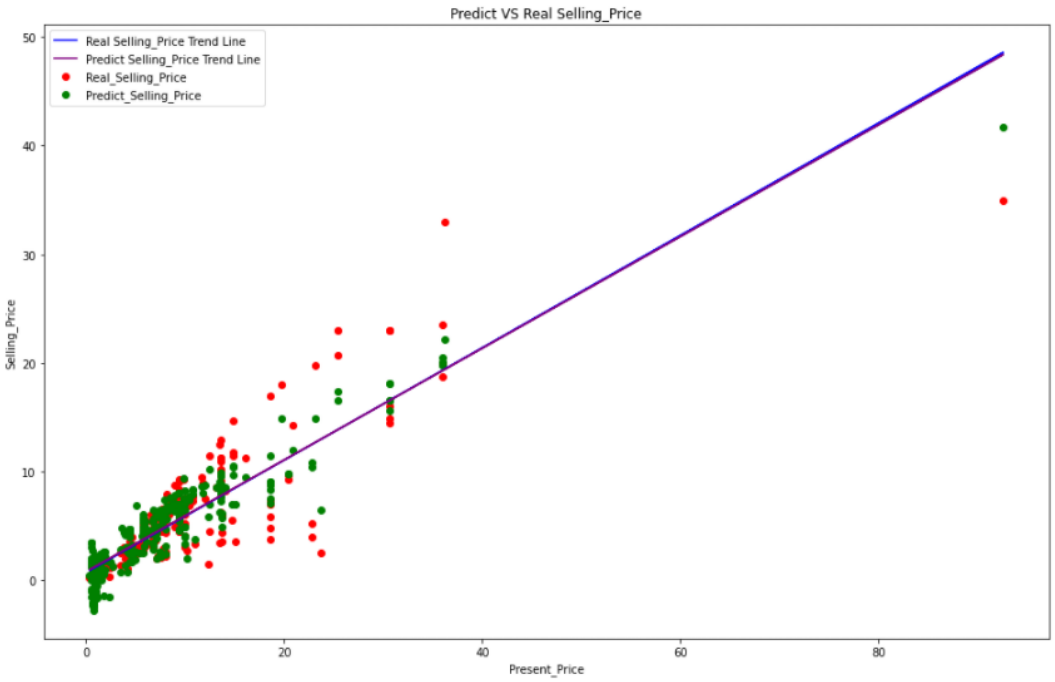
```
[ ] #Linear Regression (Selling_Price)
    yPredict = aLR.predict(x1)
    print("Linear regression score : ",aLR.score(x1, y1))
    print("Mean squared error : ", mean_squared_error(y1, yPredict))

    Linear regression score :  0.8852375928700549
    Mean squared error :  2.955033613791859
```

นำชุดข้อมูลที่ไม่ได้ผ่านการ Training มาเข้ากระบวนการ Linear Regression

```
[ ]  yPredict = aLR.predict(x1)
     np.set_printoptions(precision=2,suppress=True)
     print("Predict Y(Selling_Price)\n",yPredict)

     #for i in range(0,len(yPredict)-3,3):
     #  a_format = '{0:.3g}'.format(yPredict[i])
     #  b_format = '{0:.3g}'.format(yPredict[i+1])
     #  c_format = '{0:.3g}'.format(yPredict[i+2])
     #  print(a_format+"   "+b_format+"   "+c_format)
```

```
Predict Y(Selling_Price)
 [ 4.08  6.55  7.53  2.57  5.77  9.4   5.61  6.95  7.83  7.08  4.77  7.38
  8.22  5.42  2.06  8.32  7.91  8.32  6.86  3.16  4.17  5.7   4.69  2.79
  2.45  2.59  3.1   4.31  1.38  7.73  4.21  1.85  6.04  7.13  6.3   6.06
  4.85 -1.5   4.43  2.41  7.36  3.02  2.04  6.3   1.28  8.74  2.66  0.71
  5.42  8.05 16.58 18.15 14.86 15.6   2.01  6.99  4.88  7.49  3.28 20.11
  9.09  5.07 19.76 20.52 22.15  5.8  14.88  9.81  5.77 12.02  5.37  7.03
  8.74  2.56  6.33  4.67  7.    5.8  10.85 16.58 10.48  6.61 17.35  9.09
  6.31  6.52 41.76  6.18  3.53  4.25  7.11  9.5   4.94 18.15 10.42  8.33
 16.59 11.43  9.09  9.81  2.22  2.59  2.58  2.5   2.44  1.67  0.77  0.83
  1.7   2.44  2.17  2.05  1.63  1.3   1.28  1.28  0.11  1.82  1.28  1.
  1.94  0.01  1.9   0.11  0.53 -1.39  2.21  1.74  2.17  2.16  2.17  0.99
  2.21  1.8   2.15  1.3   0.75  0.16  1.77  1.05  0.34  1.39  1.    1.4
  0.66 -0.24  0.92  0.3  -1.34  1.45 -0.29  1.28  0.19 -0.1   0.96  3.47
  2.02  1.    2.03  3.47 -0.58  0.58  0.19 -0.99  1.62  3.08  1.62  0.23
  0.49  1.21  0.12  1.22  0.85  3.47  0.59 -1.35 -0.54  2.74  1.91 -1.64
 -1.07  0.55  0.07  0.02 -2.46  0.25  0.28  0.04  0.06 -2.26 -2.18 -1.32
 -2.81 -0.88 -2.16  0.42 -0.82 -1.53 -2.08 -2.26 -2.18  2.78  4.53  1.72
  3.97  4.31  6.33  5.64  6.76  6.01  2.48  9.68  8.44  3.39  5.35  4.6
  4.73  4.73  5.42  4.6   4.43  5.66  5.74  6.95  6.47  2.42  4.53  1.65
  5.73  4.57  6.48  6.99 10.37  4.5   4.88  6.99  4.73  8.81  4.66  2.75
  6.08  3.84  4.41  6.13  6.48  5.73  3.45  5.82  3.23  5.78  9.56  4.88
  6.49  7.11  4.88  2.46  7.74  7.68  8.03  4.35  8.44  5.41  4.23  5.83
  4.76  6.68  4.35  8.05  5.39  6.44  4.11  3.3   6.44  2.6   4.61  9.9
  8.03  8.03  6.49  7.28  4.97  2.24  8.58  7.99  4.16  7.64  5.91  6.44
  7.68  8.79  4.44  4.71  6.49  4.01  4.26  8.58  8.68  3.92  3.74 10.18
  5.38]
```

ค่าที่ได้จากการ Training Model การทำนาย



กราฟเปรียบเทียบค่าที่ได้จาก Model การทำนาย และ ค่าจริง

```
[ ]  #Cosine Similarity
     real = data['Selling_Price'].values.tolist()                          #List of real values
     predict = yPredict.tolist()                                           #List of predict values
     dot = np.dot(real,predict)                                            #Dot product
     magReal = np.linalg.norm(real)                                        #magnitude of real values
     magPredict = np.linalg.norm(predict)                                  #magnitude of predict values
     cosine = dot/(magReal*magPredict)                                     #cosine value
     degree = float("{0:.3f}".format((math.acos(cosine)*180)/math.pi))     #value in degree

     print("Cosine =","{0:.4f}".format(cosine))
     print("Degree =",degree,"°")

     Cosine = 0.9684
     Degree = 14.447 °
```

หาค่าของ Cosine Similarity เพื่อวัดประสิทธิภาพของ Model การทำนาย

2.3.4 Model การทำนายของโครงงาน

```
[ ] #Predict Output from Input
    #Use : Y = aLR.predict(X)
    print("  *** Car Selling Price Prediction ***\nInput :")
    cYear = int(input("Year : "))
    cPP = int(input("Present Price (THB) : "))
    cKM = int(input("Kms Driven (km) : "))
    cFuel = int(input("Fuel type (Select one: Petrol = 0 / Diesel = 1 / CNG = 2): "))
    cSeller = int(input("Seller Type (Select one: Individual = 0 / Dealer = 1): "))
    cTrans = int(input("Car Transmission (Select one: Manual = 0 / Automatic = 1): "))
    cOwner = int(input("Number of Car Owner (Select one: First hand = 0 / Second hand = 1 / Third hand = 2 / Fourth hand or more = 3): "))

    #Convert input
    if cYear < 2009 : cYear = 0
    elif cYear == 2010 : cYear = 1
    elif cYear == 2011 : cYear = 2
    elif cYear == 2012 : cYear = 3
    elif cYear == 2013 : cYear = 4
    elif cYear == 2014 : cYear = 5
    elif cYear == 2015 : cYear = 6
    elif cYear == 2016 : cYear = 7
    elif cYear == 2017 : cYear = 8
    elif cYear == 2018 : cYear = 9
    elif cYear >= 2019 : cYear = cYear - 2009

    if cKM >= 0 and cKM <= 15000 : cKM = 0
    elif cKM > 15000 and cKM <= 32000 : cKM = 1
    elif cKM > 32000 and cKM <= 49000 : cKM = 2
    elif cKM > 49000 : cKM = 3

    cPP = (cPP/0.44)/pow(10,5)
    cPP = "{0:.2f}".format(cPP)
    #Check var
    #print(cYear)
    #print(cPP)
    #print(cKM)

    #Calculte
    inpData = {'Year':[cYear],'Present_Price':[cPP],'Kms_Driven':[cKM],'Fuel_Type':[cFuel],'Seller_Type':[cSeller],'Transmission':[cTrans],'Owner':[cOwner]}
    newDF = pd.DataFrame(inpData)
    newX = data.drop('Selling_Price',axis=1)
    newX = newX.append(newDF,ignore_index=True)
    inpX = RobustScaler().fit_transform(newX)
    outputPredict = aLR.predict(inpX)
    opPred = outputPredict[len(outputPredict)-1]
    opPred = str(opPred*pow(10,5)*0.44)
    opPred = "{0:.2f}".format(float(opPred))
    #ans = opPred

    #Output
    print("Output :\nPredict selling price : "+opPred+" THB")
```

การรับข้อมูลจากผู้ใช้งานเข้ามาสู่ Model การทำนายและแสดงค่าการทำนาย

```
  *** Car Selling Price Prediction ***
Input :
Year : 2014
Present Price (THB) : 2000000
Kms Driven (km) : 20000
Fuel type (Select one: Petrol = 0 / Diesel = 1 / CNG = 2): 0
Seller Type (Select one: Individual = 0 / Dealer = 1): 1
Car Transmission (Select one: Manual = 0 / Automatic = 1): 0
Number of Car Owner (Select one: First hand = 0 / Second hand = 1 / Third hand = 2 / Fourth hand or more = 3): 0
Output :
Predict selling price : 956549.20 THB
```

การแสดงผล

บทที่ 3

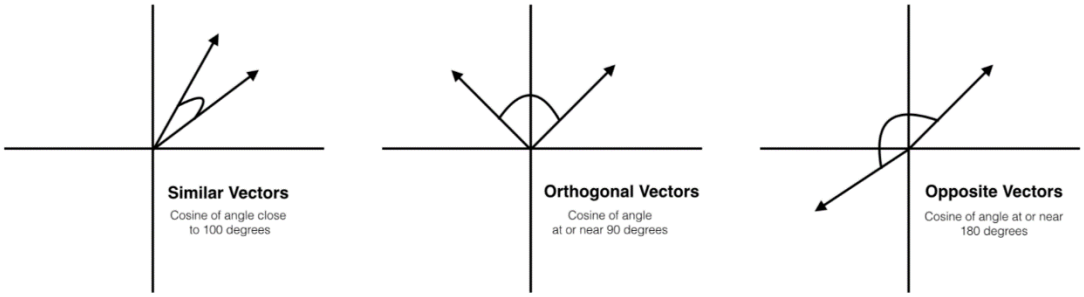
การประยุกต์ใช้ทฤษฎี

3.1 การประยุกต์ใช้ทฤษฎีเวกเตอร์

3.1.1 Cosine similarity

การนำ Cosine similarity มาใช้เพื่อวัดประสิทธิภาพของ Model การทำนายว่ามีค่าใกล้เคียงกับค่าจริงมากแค่ไหน

Cosine similarity หรือ ความคล้ายคลึงโคไซน์ ค่าจะขึ้นอยู่กับมุมที่เวกเตอร์ 2 เวกเตอร์กระทำต่อกัน



จากรูปจะเห็นว่าเมื่อเวกเตอร์ 2 เวกเตอร์ทำมุมต่อกันน้อยกว่า 100 องศาเวกเตอร์จะมีทิศทางใกล้เคียงกันต่างจากการที่เวกเตอร์ 2 เวกเตอร์ตั้งฉากและมีมุมมากกว่า 100 องศา ที่ทิศทางนั้นไม่มีความใกล้เคียงกัน

โดยที่ Cosine similarity จะมีสูตรการคำนวณดังนี้

$$D_{Cosine} = cos\theta = \frac{u \cdot v}{|u||v|}$$

จากสมการหา Cosine similarity สามารถนำไปเขียนให้อยู่ในรูปของผลรวมได้คือ

$$\frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

จากสมการการคำนวณหา Cosine similarity ที่ได้กล่าวมาเมื่อนำมาใช้วนระบบมีการเขียนโปรแกรมคำนวณดังนี้

```
[ ]  #Cosine Similarity
     real = data['Selling_Price'].values.tolist()                          #List of real values
     predict = yPredict.tolist()                                           #List of predict values
     dot = np.dot(real,predict)                                            #Dot product
     magReal = np.linalg.norm(real)                                        #magnitude of real values
     magPredict = np.linalg.norm(predict)                                  #magnitude of predict values
     cosine = dot/(magReal*magPredict)                                     #cosine value
     degree = float("{0:.3f}".format((math.acos(cosine)*180)/math.pi))     #value in degree

     print("Cosine =","{0:.4f}".format(cosine))
     print("Degree =",degree,"°")

     Cosine = 0.9684
     Degree = 14.447 °
```

จากรูปจะเห็นว่าค่า Cosine similarity ของ Model การทำนายมีความใกล้เคียงกับค่าจริงอยู่ที่ 14.447 องศาหรือประมาณ 97 เปอร์เซ็น

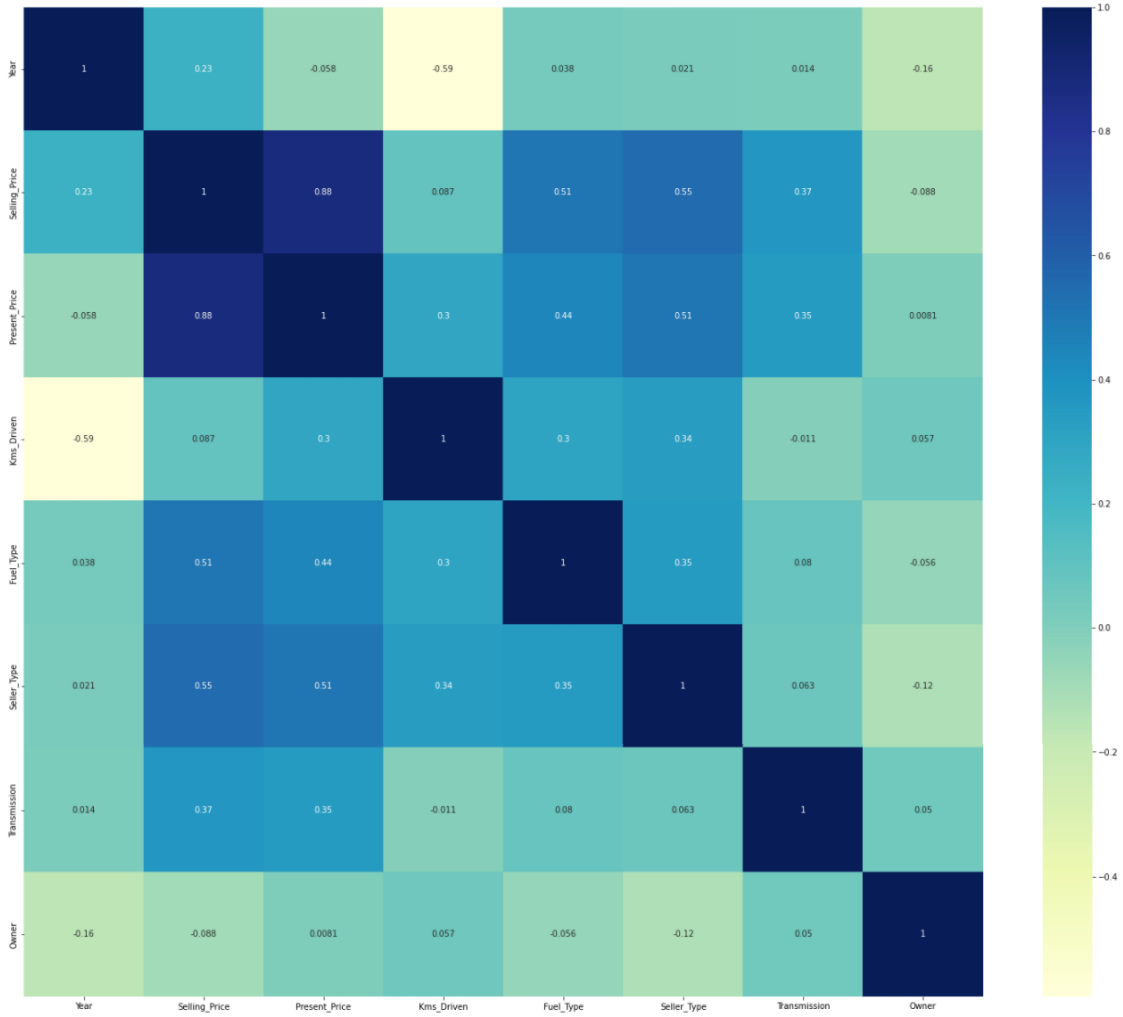
3.2 การประยุกต์ใช้ทฤษฎีเมทริกซ์

3.2.1 Correlation

Correlation หรือ สหสัมพันธ์ คือการศึกษาหาความสัมพันธ์ระหว่างตัวแปร 2 ตัวแปรขึ้นไปสามารถเขียนในรูปสมการคือ

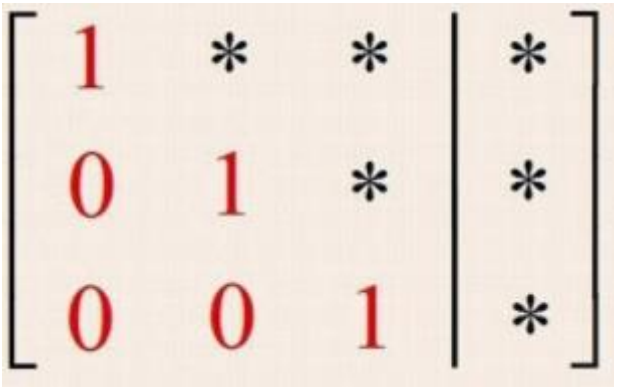
$$r_{xy} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

เมื่อนำมาใช้ในระบบจะใช้ในการหาความสัมพันธ์ของข้อมูลที่แต่ละตัวในชุดข้อมูลมีความสัมพันธ์มากน้อยแค่ไหน โดยจะแสดงอยู่ในรูปของ Matrix ดังรูปต่อไปนี้



3.2.2 Gaussian elimination

คือวิธีการในการแก้ปัญหาของสมาการ Liner โดยการทำให้เลขที่อยู่เหนือสมาชิกนำมีค่าเป็นศูนย์ทั้งหมด ซึ่งถ้าทำเฉพาะส่วนแรก ก็จะได้เมทริกซ์ที่มีรูปแบบขั้นบันได



โดยจะนำไปคำนวณในระบบเพื่อทำ Linear regression ต่อไป

[	301	1449	446	64	195	40	13	2296.17	|	1403.05
	1449	8743	1665	320	946	196	33	10689.389	|	7622.18
	446	1665	1036	138	343	58	24	4258.175	|	2227.86
	64	320	138	68	63	12	1	974.53	|	629.11
	195	946	343	63	195	29	4	2122.83	|	1310.73
	40	196	58	12	29	40	3	612.62	|	376.8
	13	33	24	1	4	3	19	104.35	|	27.2
	2296.17	10689.389	4258.175	974.53	2122.83	612.62	104.35	39932.48733	|	22288.92846]
[	1	4.813953488	1.481727575	0.2126245847	0.6478405316	0.1328903654	0.04318936877	7.628471761	|	4.661295681
	1449	8743	1665	320	946	196	33	10689.389	|	7622.18
	446	1665	1036	138	343	58	24	4258.175	|	2227.86
	64	320	138	68	63	12	1	974.53	|	629.11
	195	946	343	63	195	29	4	2122.83	|	1310.73
	40	196	58	12	29	40	3	612.62	|	376.8
	13	33	24	1	4	3	19	104.35	|	27.2
	2296.17	10689.389	4258.175	974.53	2122.83	612.62	104.35	39932.48733	|	22288.92846]
[	1	4.813953488	1.481727575	0.2126245847	0.6478405316	0.1328903654	0.04318936877	7.628471761	|	4.661295681
	0	1767.581395	-482.0232558	11.90697674	7.279069767	3.441860465	-29.58139535	-364.2665814	|	867.9625581
	0	-482.0232558	375.1495017	43.16943522	54.06312292	-1.26910299	4.737541528	855.8765947	|	148.9221262
	0	11.90697674	43.16943522	54.39202658	21.53820598	3.495016611	-1.764119601	486.3078073	|	330.7870764
	0	7.279069767	54.06312292	21.53820598	68.67109635	3.086378738	-4.42192691	635.2780066	|	401.7773422
	0	3.441860465	-1.26910299	3.495016611	3.086378738	34.68438538	1.272425249	307.4811296	|	190.3481728
	0	-29.58139535	4.737541528	-1.764119601	-4.42192691	1.272425249	18.43853821	5.17986711	|	-33.39684385
	0	-364.2665814	855.8765947	486.3078073	635.2780066	307.4811296	5.17986711	22416.21933	|	11585.80116]

[	1	4	1	0	0	0	0	7	|	4
	0	1767	-483	11	7	3	-30	-365	|	867
	0	-483	375	43	54	-2	4	855	|	148
	0	11	43	54	21	3	-2	486	|	330
	0	7	54	21	68	3	-5	635	|	401
	0	3	-2	3	3	34	1	307	|	190
	0	-30	4	-2	-5	1	18	5	|	-34
	0	-365	855	486	635	307	5	22416	|	11585]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-0.273344652	0.006225240521	0.003961516695	0.001697792869	-0.01697792869	-0.2065647991	|	0.4906621392
	0	0	242.9745331	46.00679117	55.91341256	-1.179966044	-4.200339559	755.229202	|	384.9898132
	0	0	46.00679117	53.93152235	20.95642332	2.981324278	-1.813242784	488.2722128	|	324.6027165
	0	0	55.91341256	20.95642332	67.97226938	2.98811545	-4.881154499	636.4459536	|	397.565365
	0	0	-1.179966044	2.981324278	2.98811545	33.99490662	1.050933786	307.6196944	|	188.5280136
	0	0	-4.200339559	-1.813242784	-4.881154499	1.050933786	17.49066214	-1.196943973	|	-19.28013582
	0	0	755.229202	488.2722128	636.4459536	307.6196944	-1.196943973	22340.60385	|	11764.09168]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	242	46	55	-2	-5	755	|	384
	0	0	46	53	20	2	-2	488	|	324
	0	0	55	20	67	2	-5	636	|	397
	0	0	-2	2	2	33	1	307	|	188
	0	0	-5	-2	-5	1	17	-2	|	-20
	0	0	755	488	636	307	-2	22340	|	11764]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0.1900826446	0.2272727273	-0.00826446281	-0.02066115702	3.119834711	|	1.58677686
	0	0	0	44.25619835	9.545454545	2.380165289	-1.049586777	344.4876033	|	251.0082645
	0	0	0	9.545454545	54.5	2.454545455	-3.863636364	464.4090909	|	309.7272727
	0	0	0	2.380165289	2.454545455	32.98347107	0.958677686	313.2396694	|	191.1735537
	0	0	0	-1.049586777	-3.863636364	0.958677686	16.89669421	13.59917355	|	-12.0661157
	0	0	0	344.4876033	464.4090909	313.2396694	13.59917355	19984.52479	|	10565.98347]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	44	9	2	-2	344	|	251
	0	0	0	9	54	2	-4	464	|	309
	0	0	0	2	2	32	0	313	|	191
	0	0	0	-2	-4	0	16	13	|	-13
	0	0	0	344	464	313	13	19984	|	10565]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0.2045454545	0.04545454545	-0.04545454545	7.818181818	|	5.704545455
	0	0	0	0	52.15909091	1.590909091	-3.590909091	393.6363636	|	257.6590909
	0	0	0	0	1.590909091	31.90909091	0.09090909091	297.3636364	|	179.5909091
	0	0	0	0	-3.590909091	0.09090909091	15.90909091	28.63636364	|	-1.590909091
	0	0	0	0	393.6363636	297.3636364	28.63636364	17294.54545	|	8602.636364]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0	0	-1	7	|	5
	0	0	0	0	52	1	-4	393	|	257
	0	0	0	0	1	31	0	297	|	179
	0	0	0	0	-4	0	15	28	|	-2
	0	0	0	0	393	297	28	17294	|	8602]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0	0	-1	7	|	5
	0	0	0	0	1	0.01923076923	-0.07692307692	7.557692308	|	4.942307692
	0	0	0	0	0	30.98076923	0.07692307692	289.4423077	|	174.0576923
	0	0	0	0	0	0.07692307692	14.69230769	58.23076923	|	17.76923077
	0	0	0	0	0	289.4423077	58.23076923	14323.82692	|	6659.673077]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0	0	-1	7	|	5
	0	0	0	0	1	0	-1	7	|	4
	0	0	0	0	0	30	0	289	|	174
	0	0	0	0	0	0	14	58	|	17
	0	0	0	0	0	289	58	14323	|	6659]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0	0	-1	7	|	5
	0	0	0	0	1	0	-1	7	|	4
	0	0	0	0	0	1	0	9.633333333	|	5.8
	0	0	0	0	0	0	14	58	|	17
	0	0	0	0	0	0	58	11538.96667	|	4982.8]

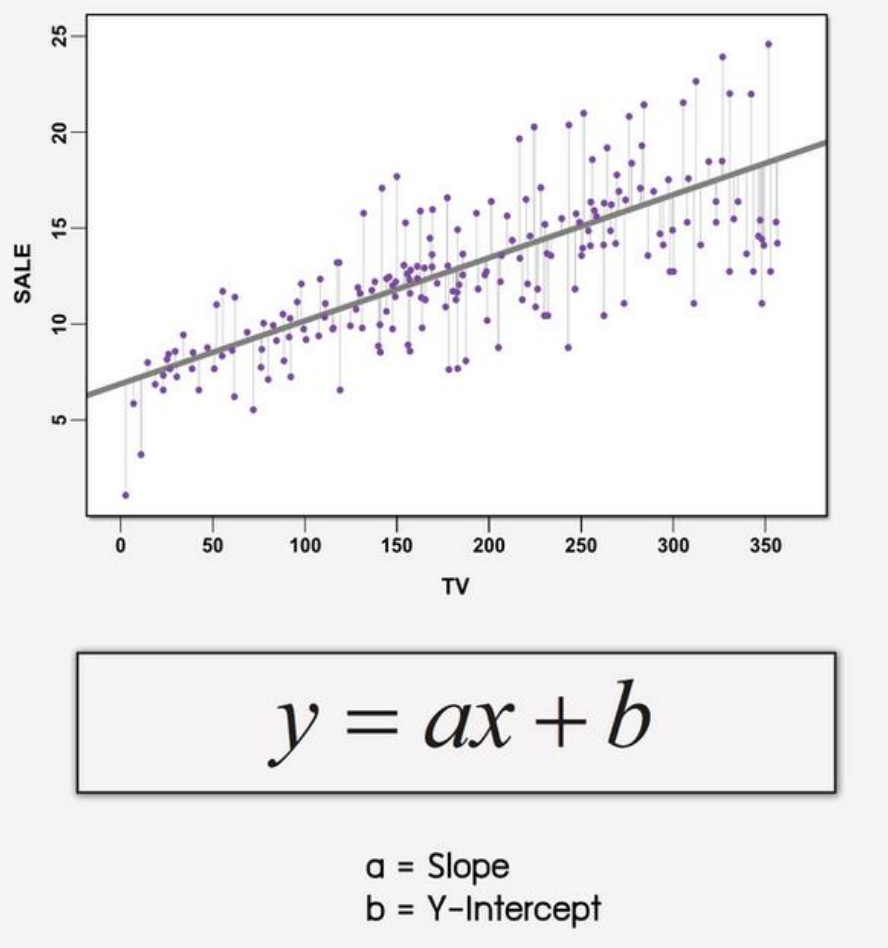
[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0	0	-1	7	|	5
	0	0	0	0	1	0	-1	7	|	4
	0	0	0	0	0	1	0	9.633333333	|	5.8
	0	0	0	0	0	0	1	4.142857143	|	1.214285714
	0	0	0	0	0	0	0	11298.68095	|	4912.371429]

[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0	0	-1	7	|	5
	0	0	0	0	1	0	-1	7	|	4
	0	0	0	0	0	1	0	9	|	5
	0	0	0	0	0	0	1	4	|	1
	0	0	0	0	0	0	0	11298	|	4912]

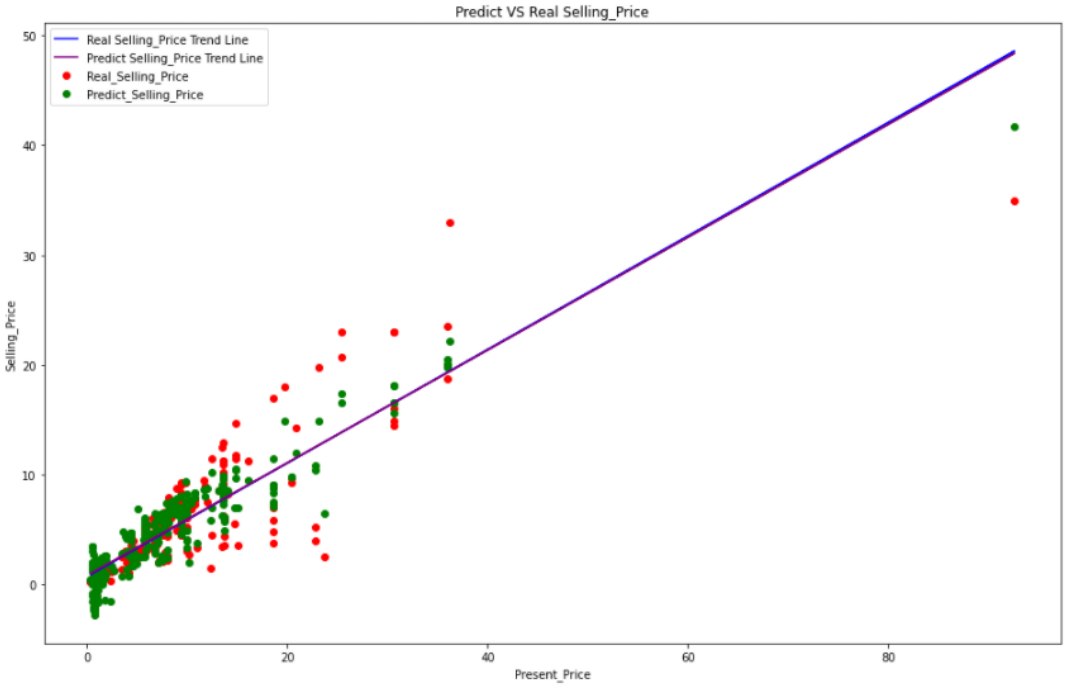
[	1	4	1	0	0	0	0	7	|	4
	0	1	-1	0	0	0	-1	-1	|	0
	0	0	1	0	0	-1	-1	3	|	1
	0	0	0	1	0	0	-1	7	|	5
	0	0	0	0	1	0	-1	7	|	4
	0	0	0	0	0	1	0	9	|	5
	0	0	0	0	0	0	1	4	|	1
	0	0	0	0	0	0	0	1	|	0.4347672154]

3.2.3 Linear regression

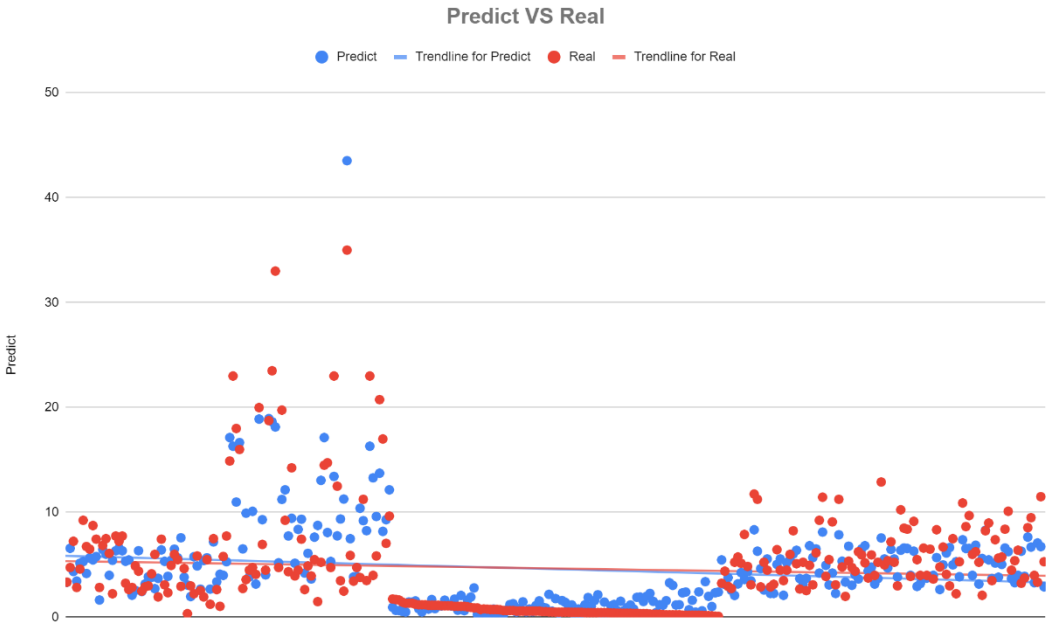
Linear Regression หรือ การวิเคราะห์การถดถอย เป็นการศึกษาความสัมพันธ์ระหว่างตัวแปรตั้งแต่ 2 ตัวขึ้นไป ซึ่งได้แก่ตัว ประมาณการ (Predictor, X) และตัวตอบสนอง (Response, y) โดยเป็นความสัมพันธ์แบบเชิงเส้น (Linear) ทั้งนี้ในขั้นตอนการทำ Regression ต้องมีการเก็บจำนวน Sample space จำนวนมากพอ นั้นคือ มี x และ y ที่มีความสัมพันธ์กันหลายๆ ครั้ง เพื่อนำมาหาสมการความสัมพันธ์



โดยนำ Linear regression มาสร้างโมเดลการทำนายราคาของรถยนต์ในอนาคต



กราฟ Linear regression ที่ได้จากการเขียนโปรแกรม



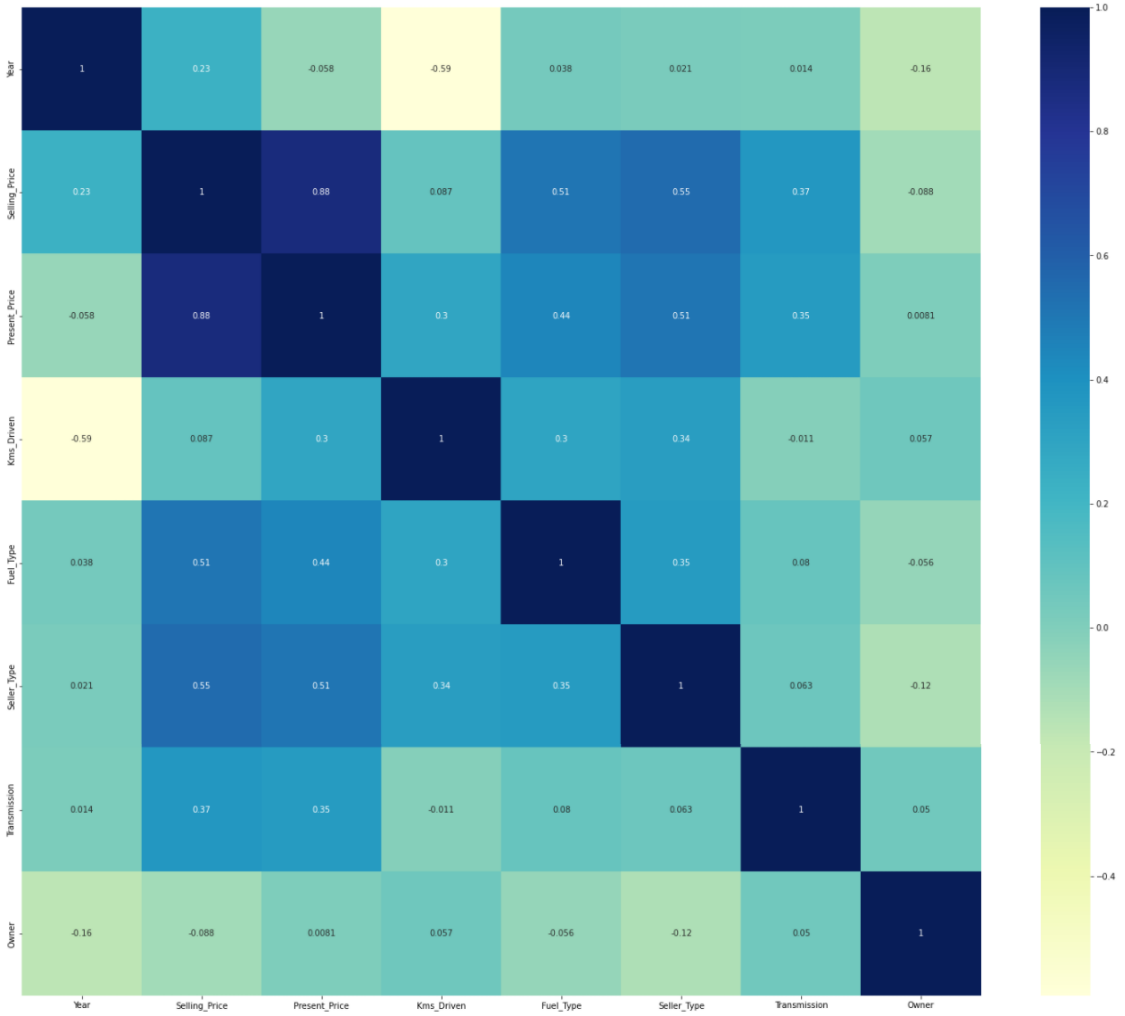
กราฟ Linear regression ที่ได้จากการคนวณใน Google Sheets

บทที่ 4

ผลการทดลอง

จากการขั้นตอนการทำงานทั้งหมด การนำเข้าข้อมูลจนถึงการ Training ชุดข้อมูลเพื่อสร้าง Model การทำนาย มีผลการทดลองดังนี้

4.1 Correlations



ผลการทดลอง จากรูป Matrix จะเห็นว่าค่าของ Correlations ที่แต่ละข้อมูลในชุดข้อมูลเทียบกันนั้นมีค่าใกล้เคียงกันมาก ดั้งนั้นชุดข้อมูลชุดนี้จึงสามารถนำไปสร้าง Model ที่มีค่าความผิดพลาดน้อยได้

4.2 Linear Regression

```
[ ]  #Linear Regression (Selling_Price)
     aLR = LinearRegression()
     aLR.fit(x1Train, y1Train)
     yPredict = aLR.predict(x1Test)
     print("Linear regression score : ",aLR.score(x1Test, y1Test))
     print("Mean squared error : ", mean_squared_error(y1Test, yPredict))
```

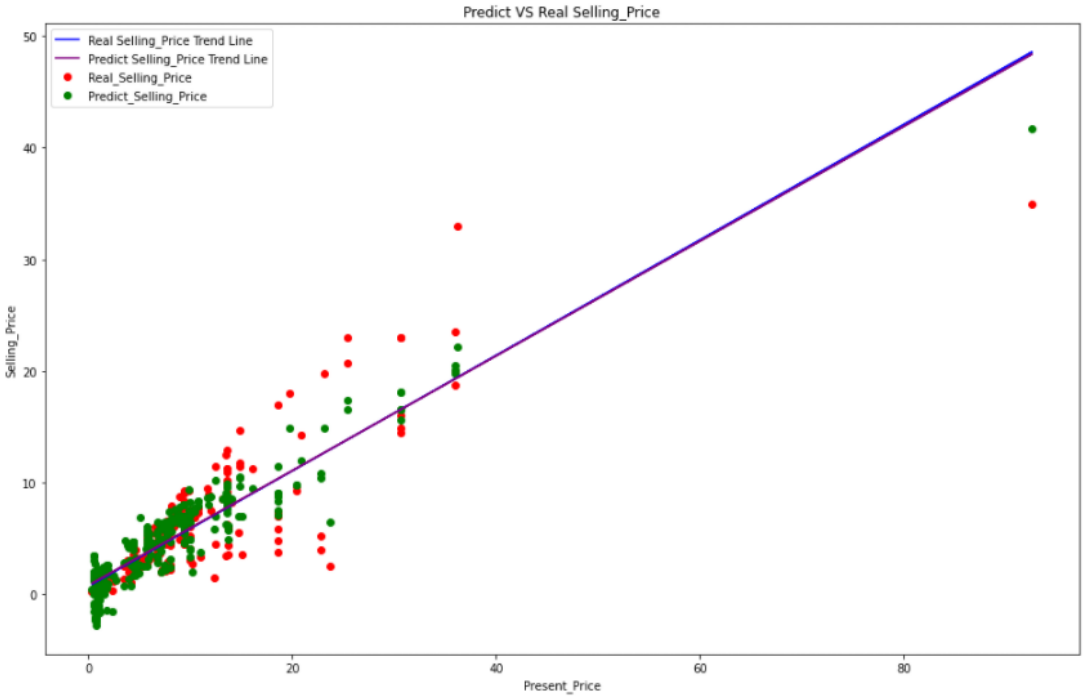
```
Linear regression score :  0.8592697783382605
Mean squared error :  3.2418029156481825
```

```
[ ]  #Linear Regression (Selling_Price)
     yPredict = aLR.predict(x1)
     print("Linear regression score : ",aLR.score(x1, y1))
     print("Mean squared error : ", mean_squared_error(y1, yPredict))
```

```
Linear regression score :  0.8852375928700549
Mean squared error :  2.955033613791859
```

ผลการทดลอง จาชุดข้อมูลนำไปเข้ากระบวนการ Linear Regression จะเห็นว่า Linear Regression score ประมาณ 88 เปอร์เซนและ Mean squared error ประมาณ 2.95 ซึ้งแสดงให้เห็นว่าตัวชุดข้อมูลนั้นมีค่าความผิดพลาดที่น้อย

4.3 Model การทำนายราคารถยนต์ในอนาคต




```
[ ]  #Cosine Similarity
     real = data['Selling_Price'].values.tolist()                          #List of real values
     predict = yPredict.tolist()                                           #List of predict values
     dot = np.dot(real,predict)                                            #Dot product
     magReal = np.linalg.norm(real)                                        #magnitude of real values
     magPredict = np.linalg.norm(predict)                                  #magnitude of predict values
     cosine = dot/(magReal*magPredict)                                     #cosine value
     degree = float("{0:.3f}".format((math.acos(cosine)*180)/math.pi))     #value in degree

     print("Cosine =","{0:.4f}".format(cosine))
     print("Degree =",degree,"°")

     Cosine = 0.9684
     Degree = 14.447 °
```

จากรูปกราฟ Linear Regression จะเห็นว่าค่าจริงและค่าที่ทำนายนั้นไปในทิศทางเดียวกัน และเมื่อหา Cosine Similarity เพื่อวัดประสิทธิภาพของ Model พบว่าใกล้เคียงถึง 96 เปอร์เซน

4.4 ตัวอย่างการทำงานของ Model กาทำนายราคารถยนต์ในอนาคต

```
  *** Car Selling Price Prediction ***
Input :
Year : 2014
Present Price (THB) : 2000000
Kms Driven (km) : 20000
Fuel type (Select one: Petrol = 0 / Diesel = 1 / CNG = 2): 0
Seller Type (Select one: Individual = 0 / Dealer = 1): 1
Car Transmission (Select one: Manual = 0 / Automatic = 1): 0
Number of Car Owner (Select one: First hand = 0 / Second hand = 1 / Third hand = 2 / Fourth hand or more = 3): 0
Output :
Predict selling price : 956549.20 THB
```

```
  *** Car Selling Price Prediction ***
Input :
Year : 2017
Present Price (THB) : 500000
Kms Driven (km) : 1000
Fuel type (Select one: Petrol = 0 / Diesel = 1 / CNG = 2): 1
Seller Type (Select one: Individual = 0 / Dealer = 1): 0
Car Transmission (Select one: Manual = 0 / Automatic = 1): 1
Number of Car Owner (Select one: First hand = 0 / Second hand = 1 / Third hand = 2 / Fourth hand or more = 3): 0
Output :
Predict selling price : 428311.60 THB
```

```
  *** Car Selling Price Prediction ***
Input :
Year : 2010
Present Price (THB) : 459000
Kms Driven (km) : 0
Fuel type (Select one: Petrol = 0 / Diesel = 1 / CNG = 2): 1
Seller Type (Select one: Individual = 0 / Dealer = 1): 1
Car Transmission (Select one: Manual = 0 / Automatic = 1): 0
Number of Car Owner (Select one: First hand = 0 / Second hand = 1 / Third hand = 2 / Fourth hand or more = 3): 0
Output :
Predict selling price : 281664.60 THB
```

เริ่มจากการให้ผู้ใช้กรอกข้อมูลของรถยนต์ที่สนใจเช่น ราคาที่ขายจากโชว์รูม ประเภทของเครื่องยนต์ ประเภทเกียร์จากนั้นระบบจะคำนวณ ราคารถยนต์ในอนาคตแล้วแสดงผลออกมาดังรูป

บทที่ 5

สรุปผลการทดลองและข้อเสนอแนะ

5.1 สรุปผลการทดลอง

จากการดำเนินการทั้งหมดที่ได้กล่าวมาโครงงานเรื่องนี้ได้ประยุกต์ใช้ความรู้ที่ได้ศึกษามาอย่างเต็มที่ ทั้งในเรื่องของ Vector และ Matrix คณะผู้จัดจึงได้เห็นว่าความรู้ต่างๆที่ได้ศึกษามานั้นสามารถนำไปประยุกต์ใช้ให้ก่อประโยชน์ได้ จนเกิดเป็นโครงงานเรื่องนี้ขึ้นมา

การหาชุดข้อมูล นำเข้าชุดข้อมูล ปรับแต่งชุดข้อมูล จนไปถึงการ Training Model เมื่อผ่านการดำเนินทั้งหมดแล้วทางคณะผู้จัดทำจึงได้ Model การทำนายราคารถยนต์ในอนาคตที่สามารถนำไปใช้ได้จริง ทางคณะผู้จัดทำหวังว่าโครงงานเรื่องนี้จะก่อประโยชน์กับบุคคลที่ต้องการหาราคาของรถยนต์ในอนาคตรวมถึงผู้ที่ต้องการศึกษาการ Model การทำนายอย่างสูงสุด

5.2 ข้อเสนอแนะ

5.2.1 ปัญหาที่พบ

- ข้อมูลมีความแตกกันมากเกินไป แก้ไขโดยการปรับชุดข้อมูลใหม่
- ชุดข้อมูลมีค่าของ Correlations มาเกินไป แก้ไขโดยการปรับชุดข้อมูลใหม่

5.2.2 ข้อเสนอแนะ

- การใช้ชุดข้อมูลที่มากว่า 301 ชุดอาจได้ผลลัพธ์ที่ละเอียดกว่านี้
- อาจทำตัวโปรแกรมออกมาในรูปแบบของ Web หรือ App ที่สามารถใช้และรับข้อมูลเพื่อการทำนายได้ง่ายกว่านี้

รายการอ้างอิง

Nehal Birla. (25 มิถุนายน 2561). *Vehicle dataset*. เข้าถึงได้จาก kaggle.com: https://www.kaggle.com/nehalbirla/vehicle-dataset-from-cardekho/metadata

Phanpaporn Z. (26 สิงหาคม 2564). *ทำความรู้จัก "Linear Regression" Algorithm ที่คนทำ Machine Learning ยังไงก็ต้องได้ใช้!* เข้าถึงได้จาก borntodev: https://www.borntodev.com/2021/08/26-linear-regression-algorithm

Supalerk Pisitsupakarn. (17 มีนาคม 2564). *เมื่อสาย DATA อยากจะกิน Pizza (โดยใช้ Jaccard Similarity และ Cosine Similarity)*. เข้าถึงได้จาก medium.com: https://medium.com/data-cafe-thailand//เมื่อสาย-data-อยากจะกิน-pizza-โดยใช้-jaccard-similarity-และ-cosine-similarity-f921fa4ab043

Thanat Lapthawan. (30 พฤษภาคม 2562). *สร้างโมเดลความสัมพันธ์ของข้อมูลทางธุรกิจด้วยเทคนิค linear regression*. เข้าถึงได้จาก bigdataexperience: http://bigdataexperience.org/business-data-relation-with-linear-regression/

wikipedia. (2564). *Cosine similarity*. เข้าถึงได้จาก wikipedia: https://en.wikipedia.org/wiki/Cosine_similarity

wikipedia. (2564). *สหสัมพันธ์*. เข้าถึงได้จาก wikipedia: https://th.wikipedia.org/wiki/สหสัมพันธ์

การกำจัดแบบเกาส์ (Gaussian elimination). (ม.ป.ป.). เข้าถึงได้จาก teamsnb: https://teamsnb.com/1960-gaussian-elimination/

สถิติเบื้องต้นง่ายๆ ที่จะทำให้คุณเข้าใจการวิเคราะห์มากขึ้น (ตอนที่ 2). (ม.ป.ป.). เข้าถึงได้จาก coraline: https://www.coraline.co.th/single-post/basic-statistic-2

ภาคผนวก

ภาคผนวก ก

ข้อมูลโครงงาน

[1] ข้อมูลที่ใช้

Vehicle dataset | Kaggle

[2] Source code หรือ File ที่ใช้ในการคำนวณ

การทำนายราคารถยนต์.ipynb - Colaboratory (google.com)

การทำนายราคารถยนต์ - Google Sheets

[3] ไฟล์ประกอบอื่นๆ

แผนการดำเนินการ .pdf - Google Drive

ภาคผนวก ข

วิดีโอและสไลด์นำเสนอโครงงาน

นำเสนอความคืบหน้า ครั้งที่ 1 - Google Slides

นำเสนอความคืบหน้าครั้งที่ 2 - Google Slides

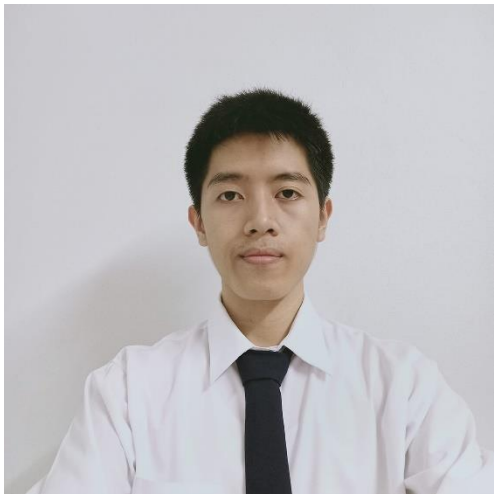
นำเสนอโครงงาน - Google Slides

วิดีโอนำเสนอ-ทำนายราคารถ - Google Drive

สมาชิก

Capybaras

63010022 นายกฤต รุ่งโรจน์กิจกุล (ฟิลด์)



63010052 นายก้องเกียรติ ชุนงาม (ก้อง)

กอดคอลงเรือ

63010042 นายกฤษติภูมิ เทียนงาม (เอิร์ท)

63010332 นายณัฐพล จำปานนท์ (โตโต้)