# APSSDC

Andhra Pradesh State Skill Development Corporation

# Day03 Machine Learning Using Python

## Day03 Objectives

## Polynomial Regression

- Linear Regression with Multiple Variables
- Train/Test splitting of data
- Under fitting, Overfitting, Best fit
- Polynomial Features
- Non-Linear Regression with One variable
- Non-Linear Regression with Multiple Variable

## [Co2 Fuel Consumption (https://raw.githubusercontent.com/AP-State-Skill-Development-Corporation/Datasets/master/Regression/FuelConsumptionCo2.csv](https://raw.githubusercontent.com/AP-State-Skill-Development-Corporation/Datasets/master/Regression/FuelConsumptionCo2.csv)

## 1. Predicting Co2 Emissions of a Car

In [1]:

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

In [2]:

```
df = pd.read_csv('https://raw.githubusercontent.com/AP-State-Skill-Development-Corporation/

df.head()
```

Out[2]:

| | MODELYEAR | MAKE | MODEL | VEHICLECLASS | ENGINESIZE | CYLINDERS | TRANSMISSION |
|---|---|---|---|---|---|---|---|
| **0** | 2014 | ACURA | ILX | COMPACT | 2.0 | 4 | AS5 |
| **1** | 2014 | ACURA | ILX | COMPACT | 2.4 | 4 | M6 |
| **2** | 2014 | ACURA | ILX HYBRID | COMPACT | 1.5 | 4 | AV7 |
| **3** | 2014 | ACURA | MDX 4WD | SUV - SMALL | 3.5 | 6 | AS6 |
| **4** | 2014 | ACURA | RDX AWD | SUV - SMALL | 3.5 | 6 | AS6 |

In [3]:

```
df.shape
```

Out[3]:

```
(1067, 13)
```

In [4]:

```
df.columns
```

Out[4]:

```
Index(['MODELYEAR', 'MAKE', 'MODEL', 'VEHICLECLASS', 'ENGINESIZE', 'CYLINDER
S',
       'TRANSMISSION', 'FUELTYPE', 'FUELCONSUMPTION_CITY',
       'FUELCONSUMPTION_HWY', 'FUELCONSUMPTION_COMB',
       'FUELCONSUMPTION_COMB_MPG', 'CO2EMISSIONS'],
      dtype='object')
```

In [7]:

```
df['MODELYEAR'].value_counts()
```

Out[7]:

```
2014    1067
Name: MODELYEAR, dtype: int64
```

In [12]:

```
df['MAKE'].value_counts()
```

Out[12]:

```
FORD             90
CHEVROLET        86
BMW              64
MERCEDES-BENZ    59
AUDI             49
GMC              49
TOYOTA           49
PORSCHE          44
VOLKSWAGEN       42
DODGE            39
MINI             36
KIA              33
NISSAN           33
CADILLAC         32
JEEP             31
MAZDA            27
HYUNDAI          24
```

In [10]:

```
df['MODEL'].value_counts()
```

Out[10]:

```
F150 FFV 4X4                8
F150 FFV                    8
FOCUS FFV                   6
ACCORD                      6
BEETLE                      6
                           ..
CLS 63 AMG S 4MATIC         1
X3 xDRIVE28i                1
CADENZA                     1
RANGE ROVER EVOQUE COUPE    1
CLA 45 AMG 4MATIC           1
Name: MODEL, Length: 663, dtype: int64
```

In [14]:

```
df['VEHICLECLASS'].value_counts(), len(df['VEHICLECLASS'].value_counts())
```

Out[14]:

```
(MID-SIZE                    178
 COMPACT                     172
 SUV - SMALL                 154
 SUV - STANDARD              110
 FULL-SIZE                    86
 TWO-SEATER                   71
 SUBCOMPACT                   65
 PICKUP TRUCK - STANDARD      62
 MINICOMPACT                  47
 STATION WAGON - SMALL        36
 VAN - PASSENGER              25
 VAN - CARGO                  22
 MINIVAN                      14
 PICKUP TRUCK - SMALL         12
 SPECIAL PURPOSE VEHICLE       7
 STATION WAGON - MID-SIZE      6
 Name: VEHICLECLASS, dtype: int64,
 16)
```

In [15]:

```
df['FUELTYPE'].value_counts()
```

Out[15]:

```
X    514
Z    434
E     92
D     27
Name: FUELTYPE, dtype: int64
```

In [16]:

```
df.columns
```

Out[16]:

```
Index(['MODELYEAR', 'MAKE', 'MODEL', 'VEHICLECLASS', 'ENGINESIZE', 'CYLINDER
S',
       'TRANSMISSION', 'FUELTYPE', 'FUELCONSUMPTION_CITY',
       'FUELCONSUMPTION_HWY', 'FUELCONSUMPTION_COMB',
       'FUELCONSUMPTION_COMB_MPG', 'CO2EMISSIONS'],
      dtype='object')
```

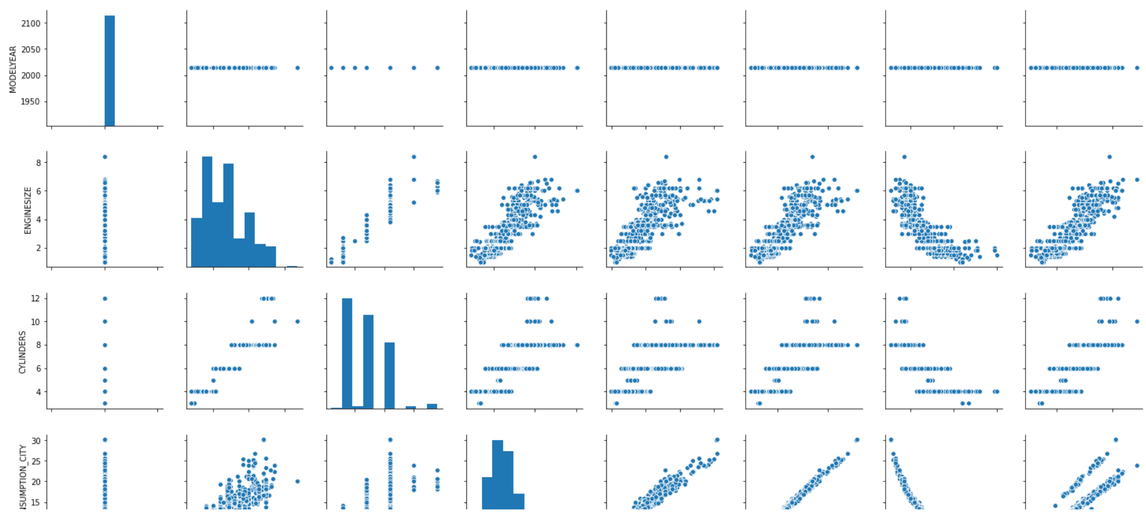
In [7]:

```python
import seaborn as sns

sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x1ed68702550>
```




In [95]:

```python
x = df[['FUELCONSUMPTION_CITY', 'FUELCONSUMPTION_HWY', 'FUELCONSUMPTION_COMB']]
y = df['CO2EMISSIONS']
```

## 3. Selection of Algo

#### Linear Regression with Multiple Variables

$$Y = M * X + C$$

$$Y = M_1 * X_1 + M_2 * X_2 + M_3 * X_3 + \ldots\ldots\ldots\ldots M_n * X_n + C$$

splitting of our data into two sub groups training set, testing set,

- 70:30
- 75:25

In [19]:

```python
from sklearn.model_selection import train_test_split
```

In [96]:

```
xtr, xtt, ytr, ytt = train_test_split(x, y, test_size = 0.3, random_state = 42)

xtr.head()
```

Out[96]:

| | FUELCONSUMPTION_CITY | FUELCONSUMPTION_HWY | FUELCONSUMPTION_COMB |
|---|---|---|---|
| **673** | 18.8 | 13.6 | 16.5 |
| **716** | 9.4 | 6.4 | 8.1 |
| **277** | 12.5 | 8.1 | 10.5 |
| **941** | 20.0 | 13.0 | 16.9 |
| **948** | 9.6 | 7.1 | 8.5 |

In [94]:

```
from sklearn.linear_model import LinearRegression
```

In [54]:

```
model = LinearRegression()
```

In [55]:

```
model.fit(xtr, ytr)
```

Out[55]:

```
LinearRegression()
```

## Testing and Evaluation of Model

In [97]:

```
xtt.head(1)
```

Out[97]:

| | FUELCONSUMPTION_CITY | FUELCONSUMPTION_HWY | FUELCONSUMPTION_COMB |
|---|---|---|---|
| **557** | 20.6 | 13.6 | 17.5 |

In [31]:

```
ytt.head(1)
```

Out[31]:

```
478    294
Name: CO2EMISSIONS, dtype: int64
```

In [33]:

```
model.predict(xtt.head(1).values)
```

Out[33]:

```
array([372.84136547])
```

In [63]:

```
xtt.iloc[1,:]
```

Out[63]:

```
FUELCONSUMPTION_CITY    10.5
FUELCONSUMPTION_HWY      8.5
FUELCONSUMPTION_COMB     9.6
Name: 719, dtype: float64
```

In [40]:

```
ytt.values[1]
```

Out[40]:

```
221
```

In [64]:

```
model.predict([[10.5, 8.5, 9.5]])
```

Out[64]:

```
array([213.27193105])
```

In [51]:

```
from sklearn.metrics import r2_score, mean_squared_error
```

In [57]:

```
pred_tt = model.predict(xtt)
pred_tr = model.predict(xtr)
```

In [65]:

```
pred_tt
```

Out[65]:

```
array([372.84136547, 215.2255471 , 276.81135357, 274.4644438 ,
       228.0238145 , 245.26627885, 187.76248787, 335.74765196,
       261.71915213, 379.0925037 , 192.34374855, 200.06151009,
       256.32944053, 286.69163045, 282.35999236, 174.47722771,
       232.89542107, 283.11546754, 363.61926097, 343.96465868,
       270.13280571, 231.40993794, 279.46086167, 208.35035237,
       256.77570999, 144.85165314, 267.60810891, 211.2683429 ,
       424.4409436 , 239.82359153, 239.89542709, 187.76248787,
       278.52759946, 221.09564404, 323.5545812 , 208.54699923,
       239.82359153, 371.28404677, 313.90506708, 257.76194793,
       263.0079884 , 276.86432931, 294.80268571, 294.1190461 ,
       185.4155781 , 205.8786313 , 230.62034687, 261.71915213,
       219.55718518, 277.18578747, 223.81698771, 218.30246482,
       276.79249374, 334.74255418, 264.88676525, 403.58459667,
       230.17407741, 231.53474924, 348.36813233, 252.37223634,
       189.26683083, 200.57961511, 218.30246482, 181.3335626 ,
       279.40788594, 200.43594398, 256.84754556, 181.53020947,
```

In [66]:

```
r2_score(ytt, pred_tt) * 100
```

Out[66]:

```
81.19697012621197
```

In [59]:

```
r2_score(ytr, pred_tr)
```

Out[59]:

```
0.8044710344049928
```

In [60]:

```
mean_squared_error(ytt, pred_tt) ** 0.5
```

Out[60]:

```
27.14190538722293
```

In [67]:

```
model.coef_
```

Out[67]:

```
array([  4.46269457, -10.39213672,  19.53616047])
```

In [68]:

```
model.intercept_
```

Out[68]:

```
69.15327576124645
```

### Non Linear Regression with one variable

# China GDP (https://raw.githubusercontent.com/AP-State-Skill-Development-Corporation/Datasets/master/Regression/china_gdp.csv)

In [69]:

```
gdp = pd.read_csv('https://raw.githubusercontent.com/AP-State-Skill-Development-Corporation
gdp.head()
```

Out[69]:

| | Year | Value |
|---|---|---|
| **0** | 1960 | 5.918412e+10 |
| **1** | 1961 | 4.955705e+10 |
| **2** | 1962 | 4.668518e+10 |
| **3** | 1963 | 5.009730e+10 |
| **4** | 1964 | 5.906225e+10 |

In [70]:

```
gdp.shape
```

Out[70]:

```
(55, 2)
```

In [71]:

```
gdp.tail()
```

Out[71]:

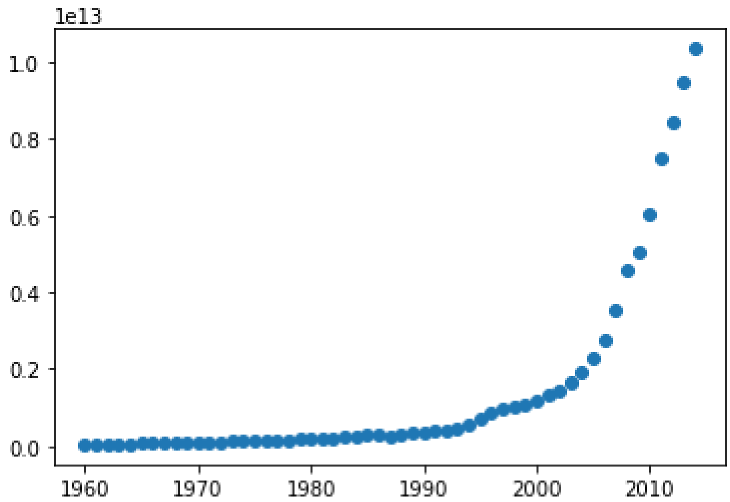
| | Year | Value |
|---|---|---|
| **50** | 2010 | 6.039659e+12 |
| **51** | 2011 | 7.492432e+12 |
| **52** | 2012 | 8.461623e+12 |
| **53** | 2013 | 9.490603e+12 |
| **54** | 2014 | 1.035483e+13 |

In [72]:

```
plt.scatter(gdp['Year'], gdp['Value'])
```

Out[72]:

```
<matplotlib.collections.PathCollection at 0x143cfa24d90>
```




- Convert data to Poly Nomial Features
- Linear Regression

$$Y = m_0 * x^0 + m_1 * x^1 + M_2 * X^2 + M_3 * x^3 + \ldots\ldots\ldots + m_n * x^n$$

p1 = X^1 p2 = x^2 p3 = x^3

$$Y = M_1 * p_1 + M_2 * P_2 + M_3 * P_3 + C$$

In [100]:

```
x = gdp['Year'].values.reshape(-1,1)
y = gdp['Value']
```

In [78]:

```
from sklearn.preprocessing import PolynomialFeatures
```

In [79]:

```
poly = PolynomialFeatures(degree = 3) # Concerting input features to degree polynomial of 3
```

In [80]:

```
x_poly = poly.fit_transform(x)

x_poly # transformed input features x^0, x^0, x^1, x^2, x^3
```

Out[80]:

```
array([[1.00000000e+00, 1.96000000e+03, 3.84160000e+06, 7.52953600e+09],
       [1.00000000e+00, 1.96100000e+03, 3.84552100e+06, 7.54106668e+09],
       [1.00000000e+00, 1.96200000e+03, 3.84944400e+06, 7.55260913e+09],
       [1.00000000e+00, 1.96300000e+03, 3.85336900e+06, 7.56416335e+09],
       [1.00000000e+00, 1.96400000e+03, 3.85729600e+06, 7.57572934e+09],
       [1.00000000e+00, 1.96500000e+03, 3.86122500e+06, 7.58730712e+09],
       [1.00000000e+00, 1.96600000e+03, 3.86515600e+06, 7.59889670e+09],
       [1.00000000e+00, 1.96700000e+03, 3.86908900e+06, 7.61049806e+09],
       [1.00000000e+00, 1.96800000e+03, 3.87302400e+06, 7.62211123e+09],
       [1.00000000e+00, 1.96900000e+03, 3.87696100e+06, 7.63373621e+09],
       [1.00000000e+00, 1.97000000e+03, 3.88090000e+06, 7.64537300e+09],
       [1.00000000e+00, 1.97100000e+03, 3.88484100e+06, 7.65702161e+09],
       [1.00000000e+00, 1.97200000e+03, 3.88878400e+06, 7.66868205e+09],
       [1.00000000e+00, 1.97300000e+03, 3.89272900e+06, 7.68035432e+09],
       [1.00000000e+00, 1.97400000e+03, 3.89667600e+06, 7.69203842e+09],
       [1.00000000e+00, 1.97500000e+03, 3.90062500e+06, 7.70373438e+09],
       [1.00000000e+00, 1.97600000e+03, 3.90457600e+06, 7.71544218e+09],
```

In [81]:

```
model1 = LinearRegression()
```

In [82]:

```
model1.fit(x_poly, y)
```

Out[82]:

```
LinearRegression()
```

In [84]:

```
model1.predict(poly.fit_transform([[2015]]))
```

Out[84]:

```
array([1.04552986e+13])
```

In [85]:

```
pred = model1.predict(x_poly)
```

In [87]:

```
r2_score(y, pred)
```

Out[87]:

```
0.96451010485102
```

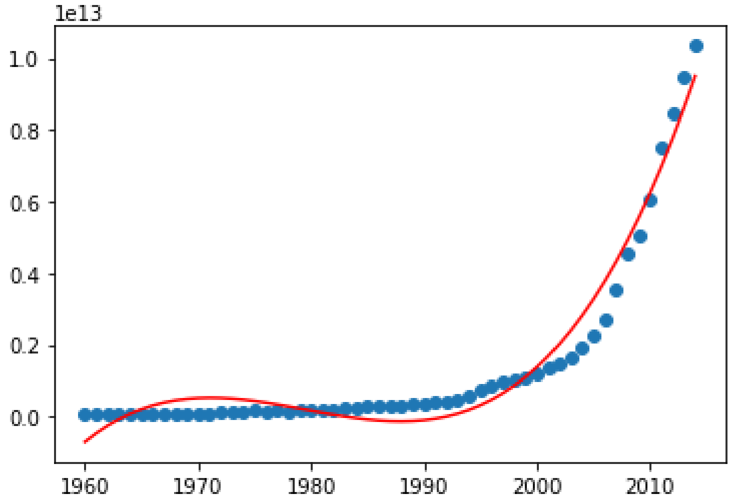
In [89]:

```
plt.scatter(gdp['Year'], gdp['Value'])

plt.plot(gdp['Year'], pred, c = 'r')
```

Out[89]:

```
[<matplotlib.lines.Line2D at 0x143d49e8eb0>]
```




In [90]:

```
model1.coef_
```

Out[90]:

```
array([ 0.00000000e+00,  3.24332951e+15, -1.63848615e+12,  2.75908680e+08])
```

In [102]:

```
acc = []
for i in range(2, 15):
    poly = PolynomialFeatures(degree = i)

    x_poly = poly.fit_transform(x)

    model1 = LinearRegression()

    model1.fit(x_poly, y)

    model1.predict(poly.fit_transform([[2015]]))

    pred = model1.predict(x_poly)

    acc.append(r2_score(y, pred))
```

In [92]:

```
acc
```

Out[92]:

```
[0.8424443474171589,
 0.96451010485102,
 0.965120230219237,
 0.9657218086560847,
 0.966314902123602,
 0.9668995728719105,
 0.9674758835411509,
 0.9680438971192882,
 0.968603676937293,
 0.9691552866497548,
 0.9696987902246914,
 0.9702342519164264,
 0.970761736268388]
```

In [112]:

```
plt.figure(figsize=(14,7))
plt.plot(range(2,15,1), acc)
plt.title('Degree Polynomial vs r2_score')
plt.xlabel('Degree Polynomial')
plt.ylabel('r2_score')
plt.grid()
plt.xlim(1, 15, 1)
plt.legend(['Degree Polynomial vs r2_score'])
plt.show()
```