40. Decision Tree (Regression)

- When data is non-linear and cannot separated through straight line.
- In left side of figure (A), data can be separated through simple linear regression
- but in right side of figure (B), data cannot be separated through simple linear regression, so we apply decision tree regression
-

No description has been provided for this image

In [ ]:

No description has been provided for this image

In [ ]:

No description has been provided for this image

In [ ]:

In [2]:

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

In [3]:

```
dataset = pd.read_csv(r'Data/salary_data.csv')
dataset.head(3)
```

Out[3]:

	Age	Experience	Salary
0	53	21	274930.685866
1	39	19	217753.696272
2	32	19	166660.977435

In [4]:

```
dataset.isnull().sum()
```

Out[4]:

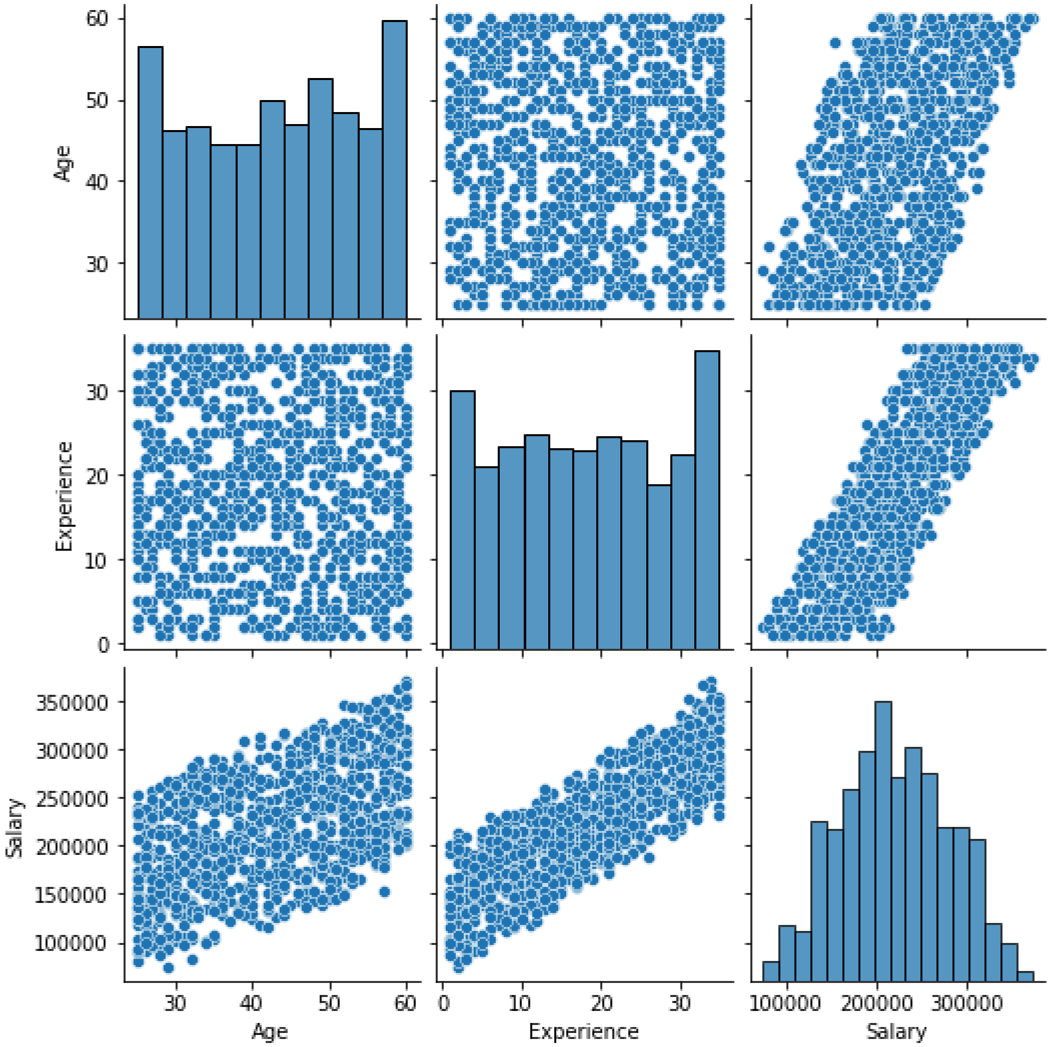
```
Age           0
Experience    0
Salary        0
dtype: int64
```

Check the data if it is linear or non-linear through graph

In [5]:

```
sns.pairplot(data=dataset)
plt.show()
```

```
C:\Users\rashi\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\axis
grid.py:123: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



Split the data into depdent and independent variables

- The data is linear and we can apply simple linear regression
- but to demonstrate linear regression through decision tree, we will apply decision tree regression

In [7]:

```
x = dataset.iloc[:,:-1]
x
```

Out[7]:

	Age	Experience
0	53	21
1	39	19
2	32	19
3	45	29
4	43	18
...	...	...
995	31	32
996	34	1
997	31	23
998	57	8
999	47	13

1000 rows × 2 columns

In [8]:
```
y = dataset['Salary']
y
```

Out[8]:
```
0      274930.685866
1      217753.696272
2      166660.977435
3      281857.674921
4      221357.621324
           ...
995    246721.167856
996     98140.456867
997    207088.257665
998    231458.172881
999    213710.389200
Name: Salary, Length: 1000, dtype: float64
```

Split the data into train and test dataset

In [9]:
```
from sklearn.model_selection import train_test_split
```

In [10]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_st
```

Build model through decision tree regressor

In [12]:
```
from sklearn.tree import DecisionTreeRegressor, plot_tree
```

In [13]:
```
dt = DecisionTreeRegressor()
dt.fit(x_train, y_train)
```

Out[13]:

```
▾ DecisionTreeRegressor

DecisionTreeRegressor()
```

Check accuracy of built model

In [15]:

```
dt.score(x_test, y_test)*100
```

Out[15]:

```
94.73975868182897
```

Check if model is over-fit

In [17]:

```
dt.score(x_train, y_train)*100
```

Out[17]:

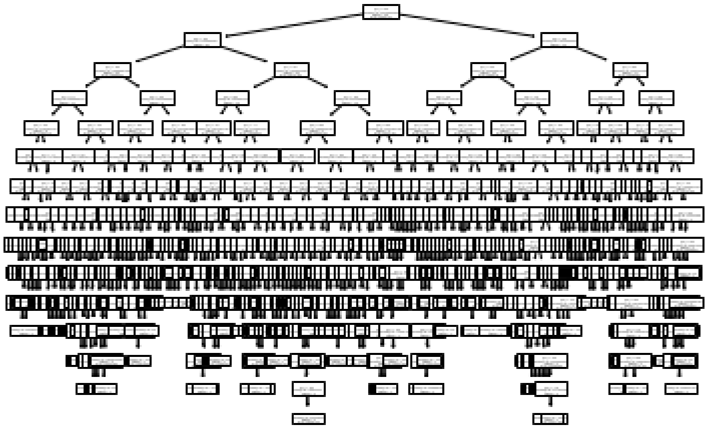
```
99.20845616821404
```

- It is slightly over-fit

Plot tree

In [16]:

```
plot_tree(dt)
plt.show()
```



In [ ]:

```

```