Problem Statement

Title::

Predicting Employee Salary Based on Experience

Background::

In the corporate world, employee compensation is a crucial factor for both the employers and the employees. Determining a fair and competitive salary based on an employee's experience is important for maintaining job satisfaction, motivation, and retention. This dataset contains data on employees' years of experience and theircorresponding salaries'.

Objective::

In [ ]:
```
The objective of this analysis is to build a predictive model that can accurat
forecast sales based on the amount of money spent on TV, Radio, and Newspaper
This model will help in understanding the impact of each advertising channel o
to maximize sales.
```

DataSet Description::

The dataset consists of the following columns:

1:Experience_Years:

In [ ]:
```
Number of years of experience the employee has.
```

2:Salary::

In [ ]:
```
Salary of the employee (in dollars).
```

In [ ]:

Importing Libraries

In [ ]:
```
Numpy ia apython library used for working with arrays.It also has functions fo
fourier transform and matrice.Pandas is a python library used for working with
exploring.manipulatingseaborn is used for visualization on matplotlib.
```

In [8]:
```
#import salary_exp dataset
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

Load the data

In [9]:
```
s=pd.read_csv("salary_exp.csv")
s
```

Out[9]:

	Experience Years	Salary
0	1.1	39343
1	1.2	42774
2	1.3	46205
3	1.5	37731
4	2.0	43525
5	2.2	39891
6	2.5	48266
7	2.9	56642
8	3.0	60150
9	3.2	54445
10	3.2	64445
11	3.5	60000
12	3.7	57189
13	3.8	60200
14	3.9	63218
15	4.0	55794
16	4.0	56957
17	4.1	57081
18	4.3	59095
19	4.5	61111
20	4.7	64500
21	4.9	67938
22	5.1	66029
23	5.3	83088
24	5.5	82200
25	5.9	81363
26	6.0	93940
27	6.2	91000
28	6.5	90000
29	6.8	91738
30	7.1	98273
31	7.9	101302
32	8.2	113812
33	8.5	111620
34	8.7	109431
35	9.0	105582

	Experience Years	Salary
36	9.5	116969
37	9.6	112635
38	10.3	122391
39	10.5	121872

Columns

In [ ]:
```
The 'columns' is used in pandas DataFrame to handle and manipulate the columns
```

In [10]:
```
s.columns
```

Out[10]:
```
Index(['Experience Years', 'Salary'], dtype='object')
```

head()

In [ ]:
```
The 'head()'function in pandas is used to quickly view the firstfew rows of a
```

In [11]:
```
s.head()
```

Out[11]:

	Experience Years	Salary
0	1.1	39343
1	1.2	42774
2	1.3	46205
3	1.5	37731
4	2.0	43525

tile()

In [ ]:
```
the tile method is a powerful tool for array manipulation especially when deal
```

In [12]:
```
s.tail()
```

Out[12]:

	Experience Years	Salary
35	9.0	105582
36	9.5	116969
37	9.6	112635
38	10.3	122391
39	10.5	121872

info()

In [ ]:
```
The info() method in pandas is used to quickly gather a summary of a DataFrame
structure and category
```

In [13]:
```
s.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 2 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Experience Years  40 non-null     float64
 1   Salary            40 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 772.0 bytes
```

shape

In [ ]:
```
shape attribute is used to get the dimensions of a DataFrame or series
```

In [14]:
```
s.shape
```

Out[14]:
```
(40, 2)
```

Asking Questions From Data

In [ ]:
```
1.)what are the total count in the dataset?
2.)what is minimum salary?
3.)maximum years experience?
4.)what is maximum salary?
5.)minimum years experience?
6.)Is there any relationship between years & salary?
7.)what is the mean of the salary?
8.)median of the salary?
9.)median of the years experience?
10.)How many employees get below 50000 salary?
11.)how many employees have above and equal 5 years experience?
```

Data Visualization

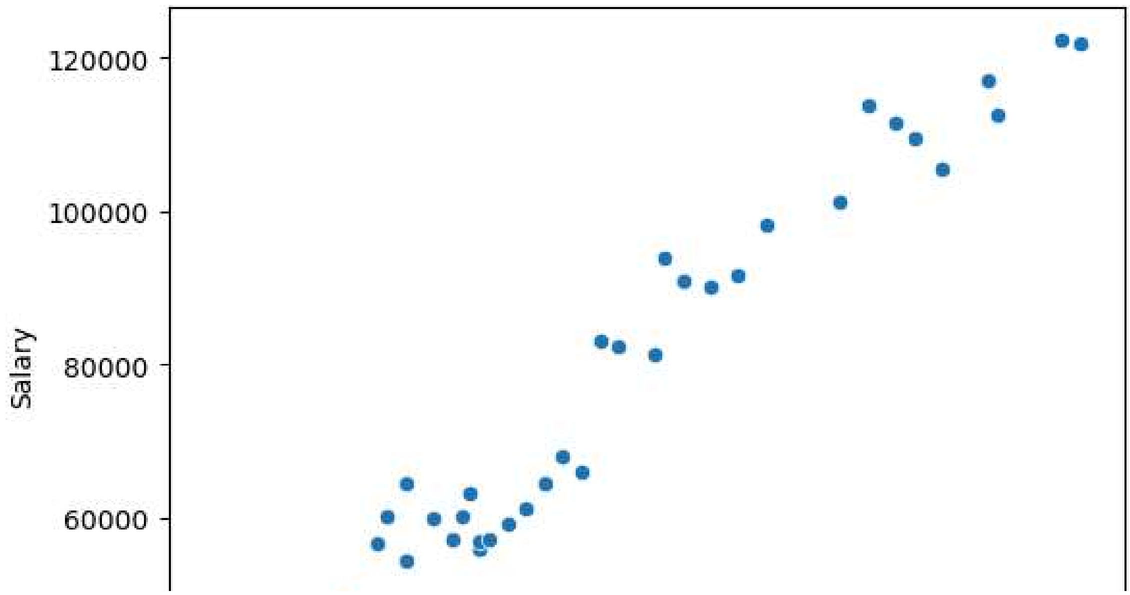
In [ ]:
```
Data Visualization is the graphical represention of information and data.By us
Data Visualization tools provide an accessible way to see and understand trend
```

we can find the salaries based on experience by using scatterplot

In [ ]:

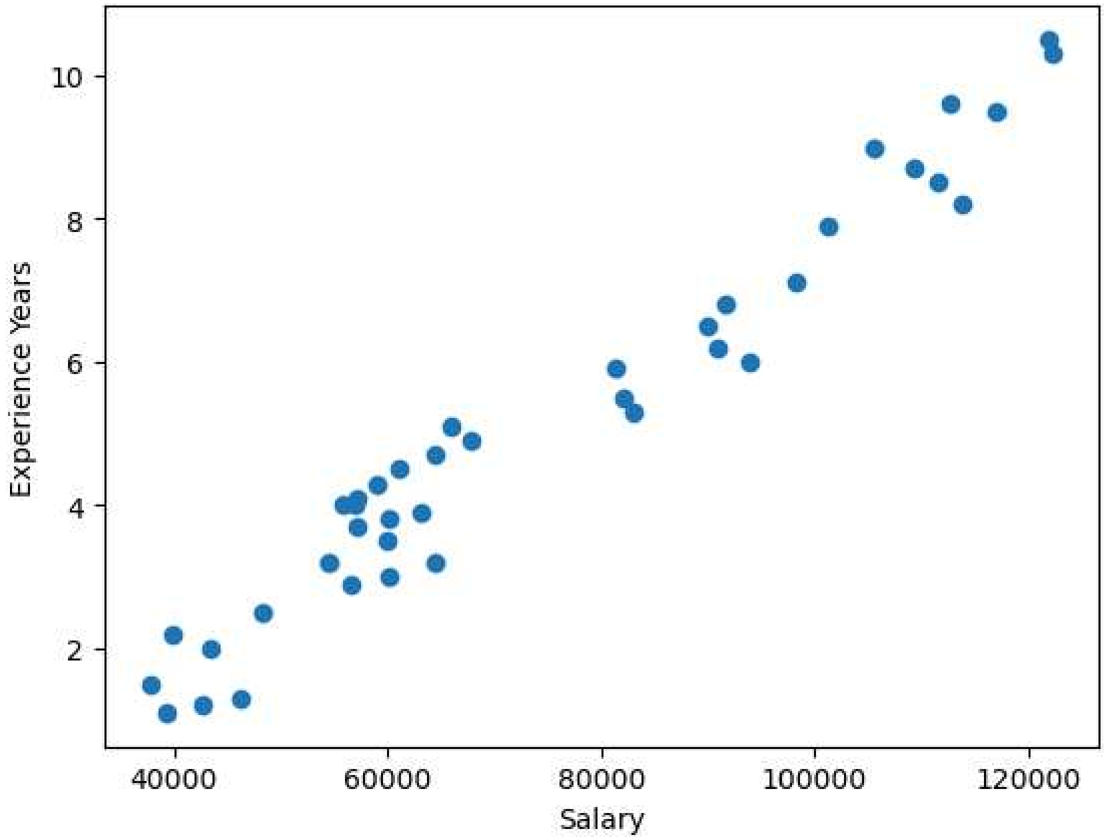
In [15]:
```
sns.scatterplot(data=s,x="Experience Years",y="Salary")
```

Out[15]:
```
<Axes: xlabel='Experience Years', ylabel='Salary'>
```



In [16]:
```
plt.scatter(s['Salary'],s['Experience Years'])
plt.xlabel('Salary')
plt.ylabel('Experience Years')
```

Out[16]:
```
Text(0, 0.5, 'Experience Years')
```



Machine Learning

In [ ]:
```
Machine learning is a subfield of artificial intelligence ,which is broadly de
to imitate intelligent human behaviour .Artificial intelligence systems are us
that is similar to how human solve problems
```

Load the Data

In [17]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

In [18]:
```
s=pd.read_csv("salary_exp.csv")
s
```

Out[18]:

	Experience Years	Salary
0	1.1	39343
1	1.2	42774
2	1.3	46205
3	1.5	37731
4	2.0	43525
5	2.2	39891
6	2.5	48266
7	2.9	56642
8	3.0	60150
9	3.2	54445
10	3.2	64445
11	3.5	60000
12	3.7	57189
13	3.8	60200
14	3.9	63218
15	4.0	55794
16	4.0	56957
17	4.1	57081
18	4.3	59095
19	4.5	61111
20	4.7	64500
21	4.9	67938
22	5.1	66029
23	5.3	83088
24	5.5	82200
25	5.9	81363
26	6.0	93940
27	6.2	91000
28	6.5	90000
29	6.8	91738
30	7.1	98273
31	7.9	101302
32	8.2	113812
33	8.5	111620
34	8.7	109431
35	9.0	105582

	Experience Years	Salary
36	9.5	116969
37	9.6	112635
38	10.3	122391
39	10.5	121872

Correlation

In [ ]:
```
correlation is a statistical measure that express the extent to which two vari
It is a common tool for describing simple relationships without making a state
```

In [19]:
```
s.corr()
```

Out[19]:

	Experience Years	Salary
Experience Years	1.000000	0.977692
Salary	0.977692	1.000000

Create a mapping

In [ ]:
```
Feature mapping is technique used in data analysis and machine learning to tra
from a lower dimensional space to a higherdimensional space
```

In [20]:
```
Salary_mapping={1:0,2:1,3:2}
s["Salary_encoded"]=s["Salary"].map(Salary_mapping)
s.head()
```

Out[20]:

	Experience Years	Salary	Salary_encoded
0	1.1	39343	NaN
1	1.2	42774	NaN
2	1.3	46205	NaN
3	1.5	37731	NaN
4	2.0	43525	NaN

In [21]:
```
Salary_mapping={1:0,2:1,3:2}
s["Salary_encoded"]=s["Salary"].map(Salary_mapping)
s.tail()
```

Out[21]:

	Experience Years	Salary	Salary_encoded
35	9.0	105582	NaN
36	9.5	116969	NaN
37	9.6	112635	NaN
38	10.3	122391	NaN
39	10.5	121872	NaN

In [22]:
```
Salary_mapping={1:0,2:1,3:2}
s["Salary_encoded"]=s["Salary"].map(Salary_mapping)
s.shape
```

Out[22]: (40, 3)

In [24]:
```
Salary_mapping={1:0,2:1,3:2}
s["Salary_encoded"]=s["Salary"].map(Salary_mapping)
s.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Experience Years  40 non-null     float64
 1   Salary            40 non-null     int64
 2   Salary_encoded    0 non-null      float64
dtypes: float64(2), int64(1)
memory usage: 1.1 KB
```

iloc

In [ ]:
```
iloc function is used to select data in a DataFrame by integer location
```

In [62]:
```
x=s.iloc[:,0:1]
y=s.iloc[:,0:1]
```

In [63]:
```
x
```

Out[63]:

	Experience Years
0	1.1
1	1.2
2	1.3
3	1.5
4	2.0
5	2.2
6	2.5
7	2.9
8	3.0
9	3.2
10	3.2
11	3.5
12	3.7
13	3.8
14	3.9
15	4.0
16	4.0
17	4.1
18	4.3
19	4.5
20	4.7
21	4.9
22	5.1
23	5.3
24	5.5
25	5.9
26	6.0
27	6.2
28	6.5
29	6.8
30	7.1
31	7.9
32	8.2
33	8.5
34	8.7
35	9.0

	Experience Years
36	9.5
37	9.6
38	10.3
39	10.5

In [64]:
```
y
```

Out[64]:

	Experience Years
0	1.1
1	1.2
2	1.3
3	1.5
4	2.0
5	2.2
6	2.5
7	2.9
8	3.0
9	3.2
10	3.2
11	3.5
12	3.7
13	3.8
14	3.9
15	4.0
16	4.0
17	4.1
18	4.3
19	4.5
20	4.7
21	4.9
22	5.1
23	5.3
24	5.5
25	5.9
26	6.0
27	6.2
28	6.5
29	6.8
30	7.1
31	7.9
32	8.2
33	8.5
34	8.7
35	9.0

	Experience Years
36	9.5
37	9.6
38	10.3
39	10.5

Type *Markdown* and LaTeX: α^2

In [ ]:

train_test_split

In [ ]:
```
train_test_split function is a method used to split a dataset into training an
```

In [66]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=
x.head()
```

Out[66]:

	Experience Years
0	1.1
1	1.2
2	1.3
3	1.5
4	2.0

Type *Markdown* and LaTeX: α^2

In [67]:
```
x_test.head()
```

Out[67]:

	Experience Years
27	6.2
9	3.2
14	3.9
0	1.1
2	1.3

In [68]:
```
y_test.head()
```

Out[68]:

	Experience Years
27	6.2
9	3.2
14	3.9
0	1.1
2	1.3

In [69]:
```
x_train.head()
```

Out[69]:

	Experience Years
17	4.1
37	9.6
38	10.3
29	6.8
24	5.5

In [70]:
```
y_train.head()
```

Out[70]:

	Experience Years
17	4.1
37	9.6
38	10.3
29	6.8
24	5.5

Linear Regerssion

Importing Linear Regression

In [46]:
```
from sklearn.linear_model import LinearRegression
```

In [47]:
```
lr=LinearRegression()
```

fit()

In [ ]:
```
fit() function is used to train a machine learning model on training dataset
```

In [48]:
```
lr=LinearRegression()
```

In [71]:
```
lr.fit(x_train,y_train)
```

Out[71]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [72]:
```
lr.fit(x_test,y_test)
```

Out[72]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Predict()

In [ ]:
```
The predict method is used to obtainthe predicted values based on the input da
```

In [73]:
```
lr.predict([[10.3]])
```

Out[73]:
```
array([[10.3]])
```

In [75]:
```
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
```

In [76]:
```
y_pred=lr.predict(x_test)
y_test.values
```

Out[76]:
```
array([[6.2],
       [3.2],
       [3.9],
       [1.1],
       [1.3],
       [7.1],
       [3.8],
       [9.5]])
```

In [77]:
```
#mean absolute error
mean_absolute_error(y_test,y_pred)
```

Out[77]:
```
8.881784197001252e-16
```

In [78]:
```
#mean_squared_error
mean_squared_error(y_test,y_pred)
```

Out[78]: 1.0846837446788912e-30

In [81]:
```
lr.predict([[9.5]])
```

Out[81]: array([[9.5]])

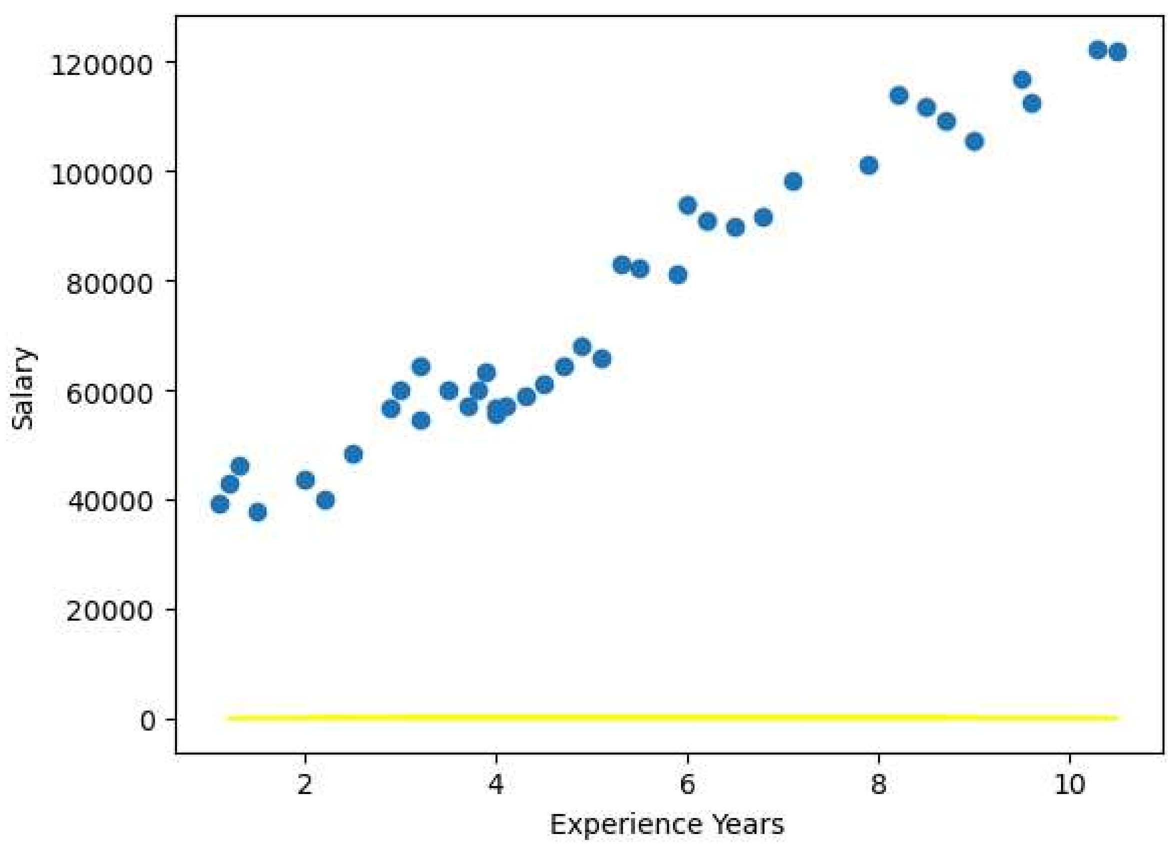
In [92]:
```
s.head()
```

Out[92]:

	Experience Years	Salary	Salary_encoded
0	1.1	39343	NaN
1	1.2	42774	NaN
2	1.3	46205	NaN
3	1.5	37731	NaN
4	2.0	43525	NaN

In [94]:
```
plt.scatter(s['Experience Years'],s['Salary'])
plt.plot(x_train,lr.predict(x_train),color='yellow')
plt.xlabel('Experience Years')
plt.ylabel('Salary')
```

Out[94]: Text(0, 0.5, 'Salary')



Type *Markdown* and LaTeX: α^2

Report

In [ ]:
```
In this project we can predicting employee salaries based on experience.
Simple linear regression is being used to solve this problem.and it shows
the model predicts well to make future decision  from the above graph is clear
that model predicts the salary well enough.
```