APSSDC

Andhra Pradesh State Skill Development Corporation

Day02 Machine Learning Using Python

Day02 Objectives

Sklearn Package, and Linear Regression using Machine Learning

- Applications of Machine Learning
- Types of regressions
- Correlation and types
- Linear regression with one variable
- Created ML model to predict the salary based on YearsOfExperience
- Evaluation Metrics in Regression Models

Linear Regression With One Variable

Business Use Case

Identify the salary of a particular Person based on Years of Experience

Data Exploration

1. Get the Data
2. Analyze the Data
3. Clean the Data
4. Preprocess the Data

In [3]:

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

In [36]:

```
df = pd.read_csv('https://raw.githubusercontent.com/AP-State-Skill-Development-Corporation/Datasets/master/Regression/Salary_Data.csv ',)
```

In [10]:

```
df
```

Out[10]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0
13	4.1	57081.0
14	4.5	61111.0
15	4.9	67938.0
16	5.1	66029.0
17	5.3	83088.0
18	5.9	81363.0
19	6.0	93940.0
20	6.8	91738.0
21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

In [8]:

```
df.shape
```

Out[8]:

```
(30, 2)
```

In [6]:

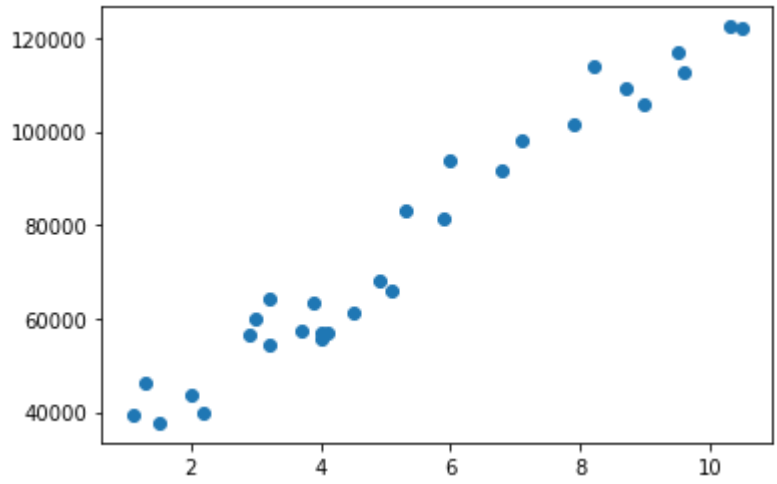
```
df.dtypes
```

Out[6]:

```
YearsExperience    float64
Salary             float64
dtype: object
```

In [7]:

```
plt.scatter(df['YearsExperience'], df['Salary'])
plt.show()
```



From the scatter plot we identified that YoE is having `Linearly Strong Prositive Corelation` with the salary

3. Selection of Algorithm

Linear Regression

4. Build the Model

- Train the Model
- Test the Model

In [19]:

```
x = df['YearsExperience']
y = df['Salary']
```

In [20]:

```
x
```

Out[20]:

```
0      1.1
1      1.3
2      1.5
3      2.0
4      2.2
5      2.9
6      3.0
7      3.2
8      3.2
9      3.7
10     3.9
11     4.0
12     4.0
13     4.1
14     4.5
15     4.9
16     5.1
17     5.3
18     5.9
19     6.0
20     6.8
21     7.1
22     7.9
23     8.2
24     8.7
25     9.0
26     9.5
27     9.6
28    10.3
29    10.5
Name: YearsExperience, dtype: float64
```

In [13]:

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

In [14]:

```
model = LinearRegression()
```

Train the Model

In [15]:

```
model.fit(x, y)
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
<ipython-input-15-d3dc977168f5> in <module>
----> 1 model.fit(x, y)

~\anaconda3\lib\site-packages\sklearn\linear_model\_base.py in fit(self,
 X, y, sample_weight)
    503
    504         n_jobs_ = self.n_jobs
--> 505         X, y = self._validate_data(X, y, accept_sparse=['csr', 'cs
c', 'coo'],
    506                                    y_numeric=True, multi_output=Tr
ue)
    507

~\anaconda3\lib\site-packages\sklearn\base.py in _validate_data(self, X,
 y, reset, validate_separately, **check_params)
    430                 y = check_array(y, **check_y_params)
    431             else:
--> 432                 X, y = check_X_y(X, y, **check_params)
    433             out = X, y
    434

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*arg
s, **kwargs)
     71                           FutureWarning)
     72         kwargs.update({k: arg for k, arg in zip(sig.parameters, ar
gs)})
---> 73         return f(**kwargs)
     74     return inner_f
     75

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in check_X_y(X,
 y, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_fini
te, ensure_2d, allow_nd, multi_output, ensure_min_samples, ensure_min_feat
ures, y_numeric, estimator)
    794         raise ValueError("y cannot be None")
    795
--> 796     X = check_array(X, accept_sparse=accept_sparse,
    797                     accept_large_sparse=accept_large_sparse,
    798                     dtype=dtype, order=order, copy=copy,

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*arg
s, **kwargs)
     71                           FutureWarning)
     72         kwargs.update({k: arg for k, arg in zip(sig.parameters, ar
gs)})
---> 73         return f(**kwargs)
     74     return inner_f
     75

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in check_array(a
rray, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_fi
nite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estima
tor)
    618             # If input is 1D raise error
    619             if array.ndim == 1:
--> 620                 raise ValueError(
    621                     "Expected 2D array, got 1D array instead:\narr
```

```
ay={}.\n"
    622                     "Reshape your data either using array.reshape
(-1, 1) if "

ValueError: Expected 2D array, got 1D array instead:
array=[ 1.1  1.3  1.5  2.   2.2  2.9  3.   3.2  3.2  3.7  3.9  4.   4.
 4.1
  4.5  4.9  5.1  5.3  5.9  6.   6.8  7.1  7.9  8.2  8.7  9.   9.5  9.6
 10.3 10.5].
Reshape your data either using array.reshape(-1, 1) if your data has a sin
gle feature or array.reshape(1, -1) if it contains a single sample.
```

In [21]:

```
x = x.values.reshape(-1, 1)
x
```

Out[21]:

```
array([[ 1.1],
       [ 1.3],
       [ 1.5],
       [ 2. ],
       [ 2.2],
       [ 2.9],
       [ 3. ],
       [ 3.2],
       [ 3.2],
       [ 3.7],
       [ 3.9],
       [ 4. ],
       [ 4. ],
       [ 4.1],
       [ 4.5],
       [ 4.9],
       [ 5.1],
       [ 5.3],
       [ 5.9],
       [ 6. ],
       [ 6.8],
       [ 7.1],
       [ 7.9],
       [ 8.2],
       [ 8.7],
       [ 9. ],
       [ 9.5],
       [ 9.6],
       [10.3],
       [10.5]])
```

In [22]:

```
model.fit(x, y)
```

Out[22]:

```
LinearRegression()
```

In [23]:

```
model.coef_
```

Out[23]:

```
array([9449.96232146])
```

In [24]:

```
model.intercept_
```

Out[24]:

```
25792.20019866871
```

$$Salary = 9449.9623 * YrsOfExp + 25792.200 + 5592.55$$

Test the Model

In [25]:

```
model.predict([[1.5]])
```

Out[25]:

```
array([39967.14368085])
```

In [27]:

```
model.predict([[1.3]])
```

Out[27]:

```
array([38077.15121656])
```

In [28]:

```
y_pred = model.predict(x)
y_pred
```

Out[28]:

```
array([ 36187.15875227,  38077.15121656,  39967.14368085,  44692.12484158,
        46582.11730587,  53197.09093089,  54142.08716303,  56032.07962732,
        56032.07962732,  60757.06078805,  62647.05325234,  63592.04948449,
        63592.04948449,  64537.04571663,  68317.03064522,  72097.0155738 ,
        73987.00803809,  75877.00050238,  81546.97789525,  82491.9741274 ,
        90051.94398456,  92886.932681  , 100446.90253816, 103281.8912346 ,
       108006.87239533, 110841.86109176, 115566.84225249, 116511.83848464,
       123126.81210966, 125016.80457395])
```

Evaluate the Model

In [29]:

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

In [31]:

```
r2_score(y, y_pred) * 100
```

Out[31]:

95.69566641435085

In [32]:

```
mean_squared_error(y, y_pred) ** 0.5
```

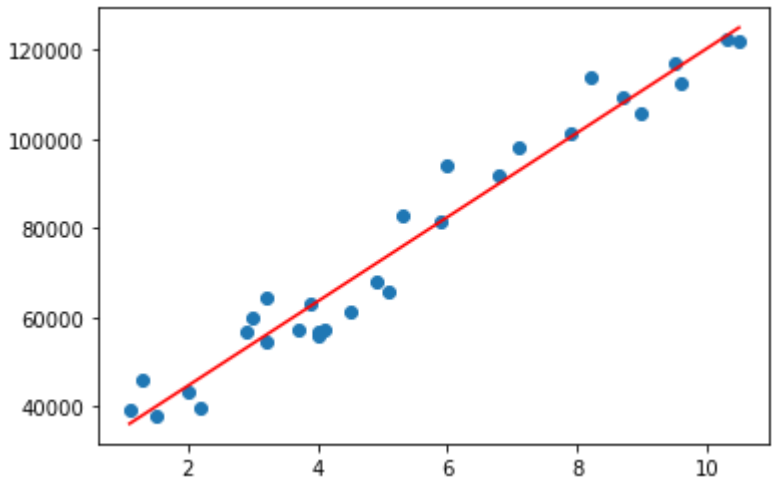
Out[32]:

5592.043608760662

In [34]:

```
plt.scatter(df['YearsExperience'], df['Salary'])
plt.plot(df['YearsExperience'], y_pred, color = 'r')

plt.show()
```



- Types of Data based on Statics
- Applications of Ml
- Differen Algorithms will'll discuss in 8 days
- What is regression
- Types of regression
- Corelation
- Steps in involved in Regression
- Linear regression for predicting the salary of a preson based on years on of experience