BCSE0133 and BCSE0701

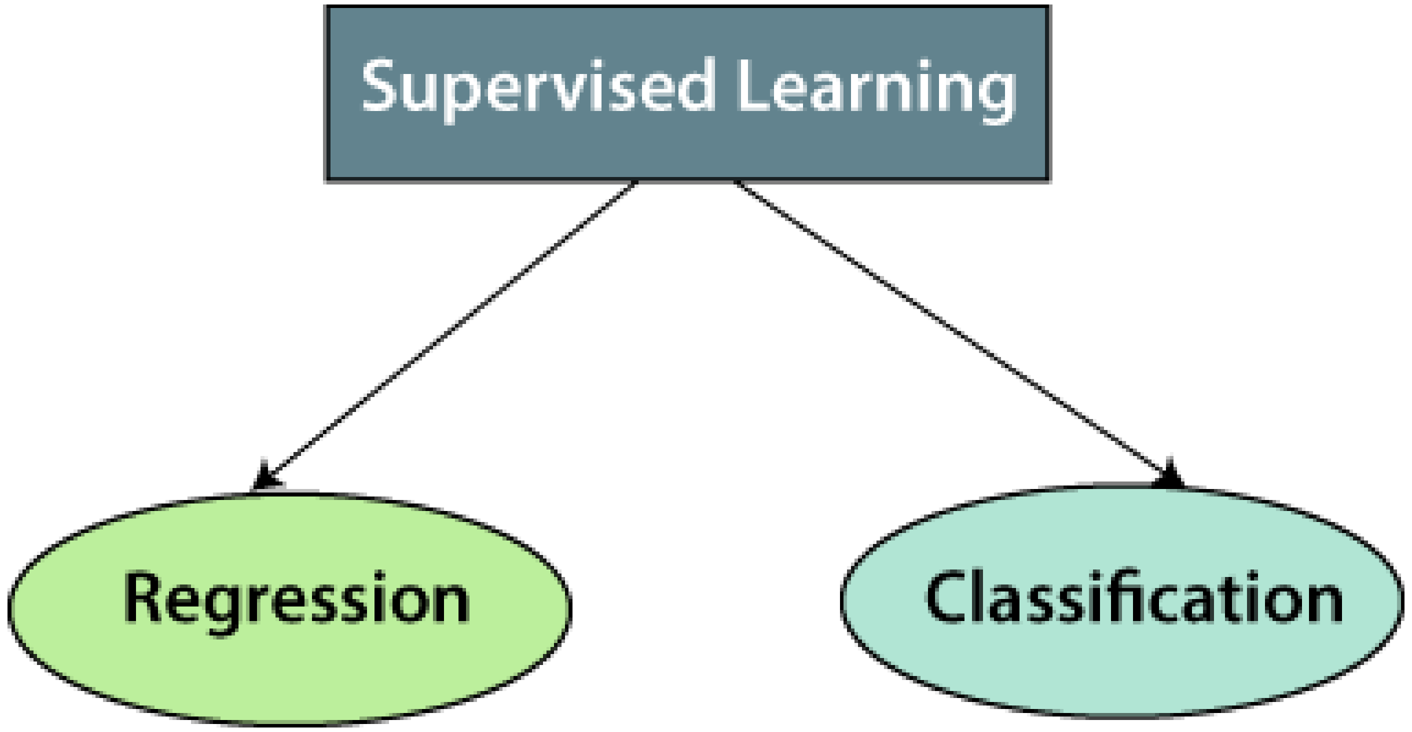
Linear Regression

Machine Learning Lab

DR GAURAV KUMAR

ASST. PROF, CEA, GLA UNIVERSITY

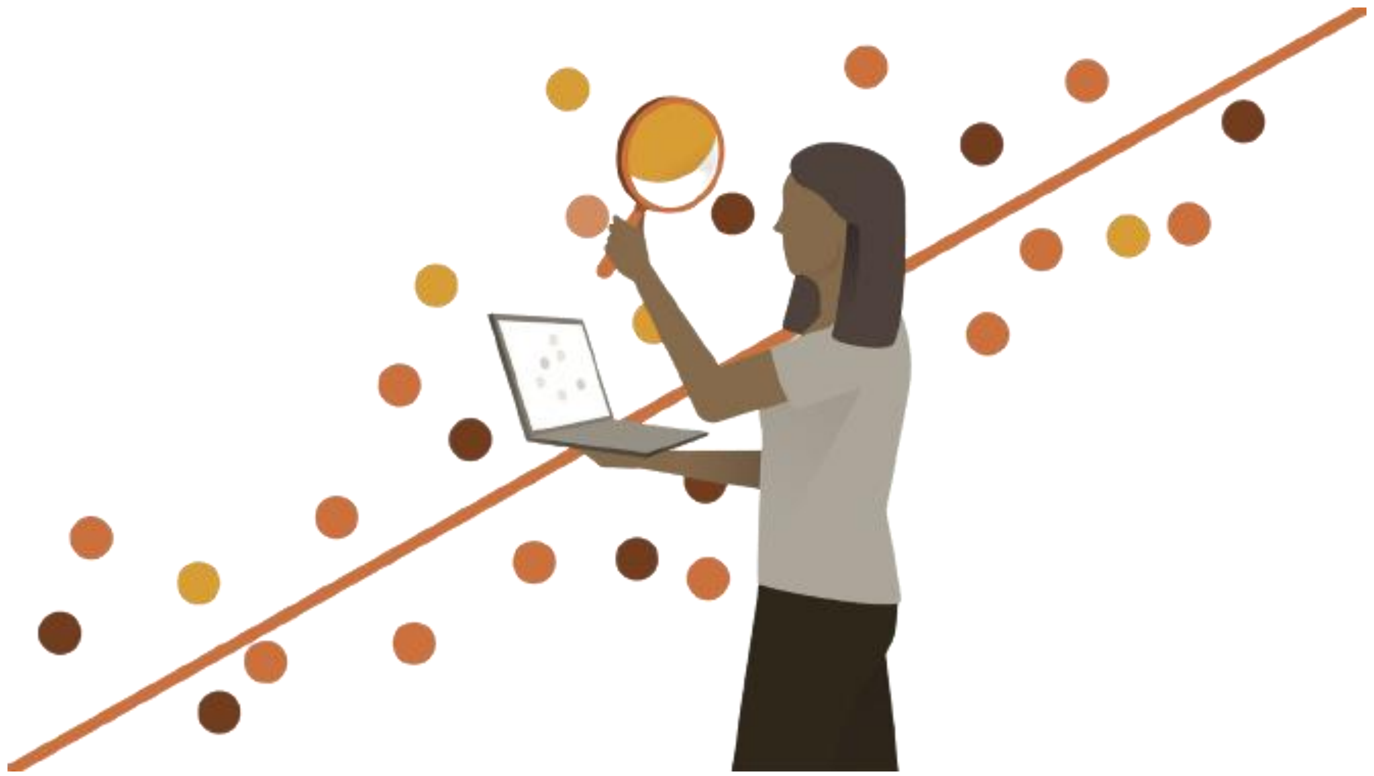
Supervised Machine learning algorithms



Regression

- It is commonly used to make projections, such as for sales revenue for a given business, weather forecasting, stock price prediction and so on.

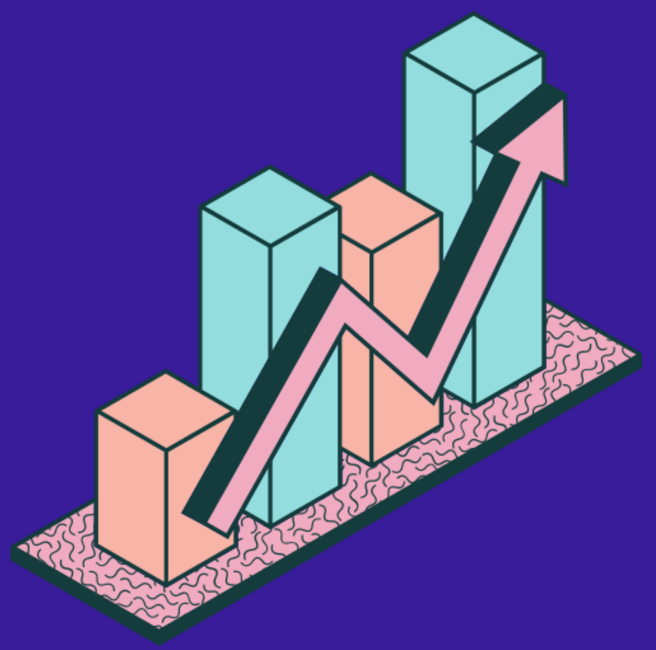
- It is used to understand the relationship between dependent and independent variables.
- Linear regression, logistical regression, and polynomial regression are popular regression algorithms.



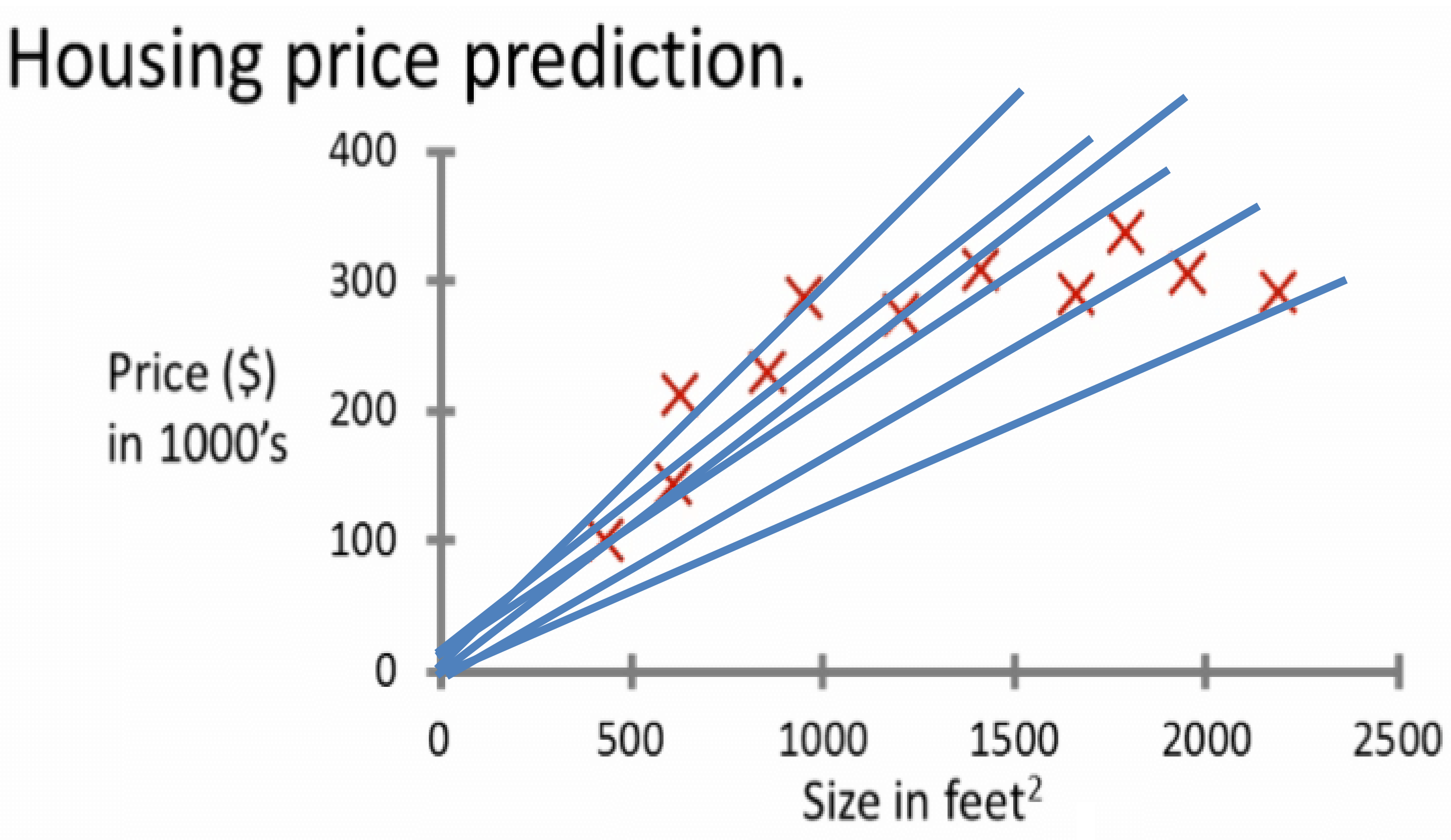
GLA UNIVERSITY MATHURA
Accredited with A Grade by NAAC
12-B Status from UGC

Linear Regression

- It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s).
- We establish the relationship between independent and dependent variables by fitting a best line.



Supervised Machine learning algorithms



Linear Regression

Example- House Prediction

- A list of houses with size and price is given.
- **Need to find best fit line to predict the price.** This best fit line is known as regression line and represented by a linear equation **Y= a *X + b.**

In this equation:
Y – Dependent Variable (Predicted Price of House)
a – Slope
X – Independent variable (Size of House (Predictor))
b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.

Implementing Linear Regression for Salary Prediction

1. Download and read the data set

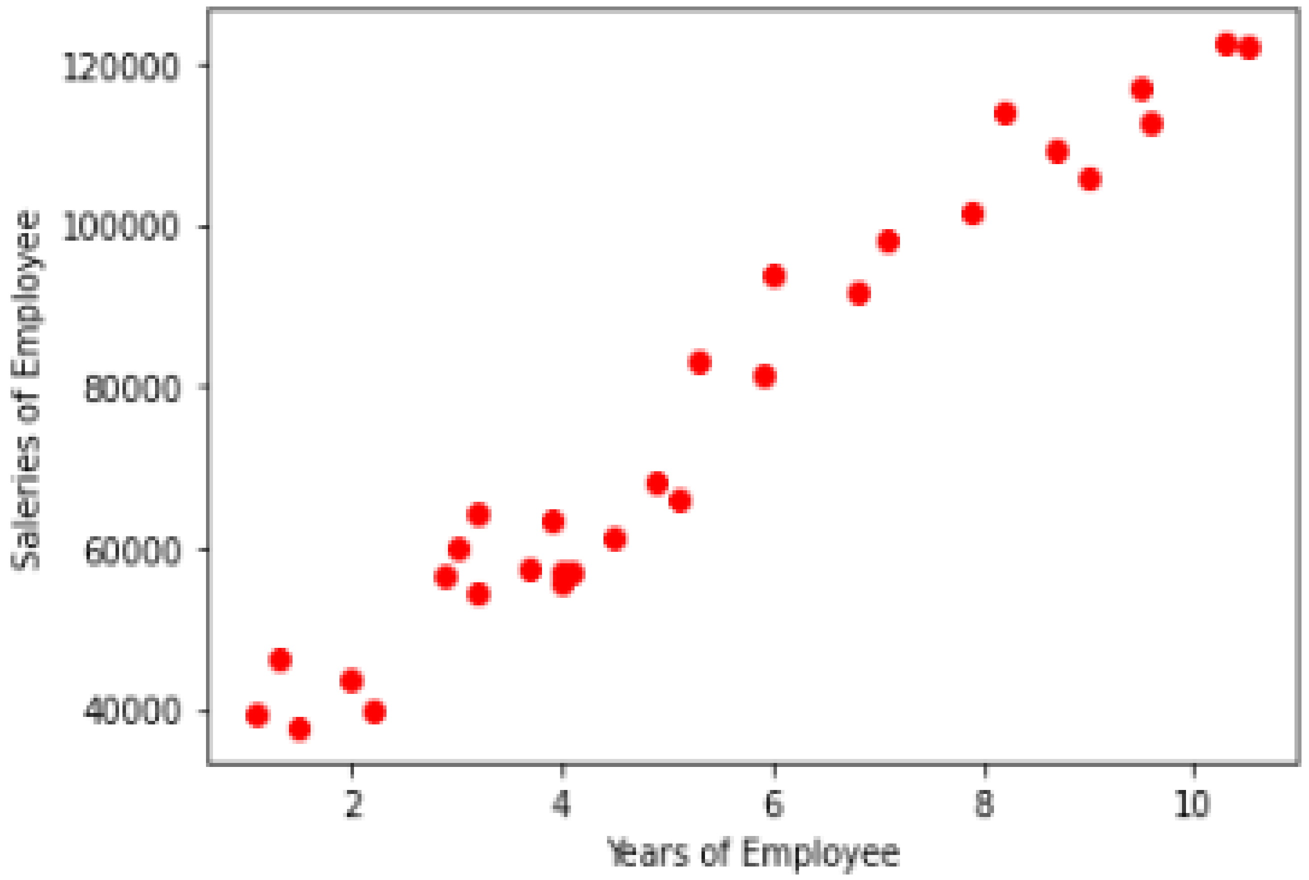
It has 2 columns — "Years of Experience" and "Salary" for 30 employees in a company. So in this example, we will train a Simple Linear Regression model to learn the correlation between the number of years of experience of each employee and their respective salary.

```
import pandas as pd
dataset = pd.read_csv('Salary_Data.csv')
```

YearsExperience	Salary
1.1	39343.00
1.3	46205.00
1.5	37731.00
2.0	43525.00
2.2	39891.00
2.9	56642.00
3.0	60150.00
3.2	54445.00
3.2	64445.00
3.7	57189.00
3.9	63218.00
4.0	55794.00
4.0	56957.00
4.1	57081.00
4.5	61111.00
4.9	67938.00
5.1	66029.00
5.3	83088.00
5.9	81363.00
6.0	93940.00
6.8	91738.00

Implementing Linear Regression for Salary Prediction

2. Visualize the data set




```python
#Visualize the dataset
plt.scatter(dataset['Experiences'],dataset['Salary'],color='red')

plt.title("linear Regression Salary Vs Experience")
plt.xlabel("Years of Employee")
plt.ylabel("Salaries of Employee")
plt.show()
```

Experiences	Salary
1.1	39343.00
1.3	46205.00
1.5	37731.00
2.0	43525.00
2.2	39891.00
2.9	56642.00
3.0	60150.00
3.2	54445.00
3.2	64445.00
3.7	57189.00
3.9	63218.00
4.0	55794.00
4.0	56957.00
4.1	57081.00
4.5	61111.00
4.9	67938.00
5.1	66029.00
5.3	83088.00
5.9	81363.00
6.0	93940.00
6.8	91738.00

GLA UNIVERSITY MATHURA
Accredited with A Grade by NAAC
12-B Status from UGC

Implementing Linear Regression for Salary Prediction

3. Divide the dataset

Categorized dataset into Independent and Dependent variable

X = dataset.iloc[:, :-1].values

OR

X = dataset.iloc[:, 0].values

OR

X = dataset.iloc[:, [0]].values

OR

X = dataset[Experience].values

y = dataset.iloc[:,1].values

Independent Variable

Dependent Variable

Experience	Salary
1.1	39343.00
1.3	46205.00
1.5	37731.00
2.0	43525.00
2.2	39891.00
2.9	56642.00
3.0	60150.00
3.2	54445.00
3.2	64445.00
3.7	57189.00
3.9	63218.00
4.0	55794.00
4.0	56957.00
4.1	57081.00
4.5	61111.00
4.9	67938.00
5.1	66029.00
5.3	83088.00
5.9	81363.00
6.0	93940.00
6.8	91738.00

Implementing Linear Regression for Salary Prediction

4. Training and Testing (Splitting again)

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test =train_test_split(X,y, test_size=0.3, random_state=0)

X_train
X_train.shape
X_test
X_test.shape
y_train
y_train.shape
y_test
y_test.shape
```

Also Calculate Shape and Size

- With **random_state=None** , we get different train and test sets across different executions and the shuffling process is out of control. i.e., every time you run your code again, it will generate a different test set.
- With **random_state=0 , (or 42) we get the same train and test sets across different executions**.

Implementing Linear Regression for Salary Prediction

5. Apply Linear Regression Model

```
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit(X_train,y_train)

#for predict the test values
y_prdict=reg.predict(X_test)
```

Note- If reg.fit shows error, then convert X_train 1-D data into 2-D Array using reshape(-1,1)
Use this X_train=X_train.shape(-1,1) then execute reg.fit

Implementing Linear Regression for Salary Prediction

6. Analyzing the Results

Understand the result, Print Actual and Predicted Values

X_test

y_test

y_prdict

diff_pred= y_test – y_prdcit

```
diff_pred
```

```
array([ 3086.78327049,   797.08258899,  8073.46261459,    64.41035735,
       -1269.12643996, -1219.33546892,  4000.89968866,  8424.43648597,
       -6701.22384198])
```

Implementing Linear Regression for Salary Prediction

6. Analyzing the Results

```
diff_pred
```

```
array([ 3086.78327049,   797.08258899,  8073.46261459,    64.41035735,
       -1269.12643996, -1219.33546892,  4000.89968866,  8424.43648597,
       -6701.22384198])
```

res_df = pd.concat([pd.Series(y_pred),pd.Series(y_test), pd.Series(diff_pred)], axis=1)

```
res_df.columns=['Prediction','Original Data','Diff']
```

```
res_df
```

	Prediction	Original Data	Diff
0	40817.783270	37731	3086.783270
1	123188.082589	122391	797.082589
2	65154.462615	57081	8073.462615
3	63282.410357	63218	64.410357
4	115699.873560	116969	-1269.126440
5	108211.664531	109431	-1219.335469
6	116635.899689	112635	4000.899689
7	64218.436486	55794	8424.436486
8	76386.776158	83088	-6701.223842

Axis=1 indicates column

Axis=0 indicates row

Use Flatten() to convert 2D array into 1D array

y_pred=y_pred.flatten()

Implementing Linear Regression for Salary Prediction

7. Visualize the Training data

```
plt.scatter(X_train, y_train,color='red')
plt.plot(X_train, reg.predict(X_train), color='blue')
plt.title("Training of linear Regression Salary Vs Experience")
plt.xlabel("Years of Employee")
plt.ylabel("Salaries of Employee")
plt.show()
```

Implementing Linear Regression for Salary Prediction

8. Visualize the Testing data

```
plt.scatter(X_test, y_test, color='red')
plt.plot(X_train, reg.predict(X_train), color='blue')
plt.title("Linear Regression Salary Vs Experience")
plt.xlabel("Years of Employee")
plt.ylabel("Salaries of Employee")
plt.show()
```

Implementing Linear Regression for Salary Prediction

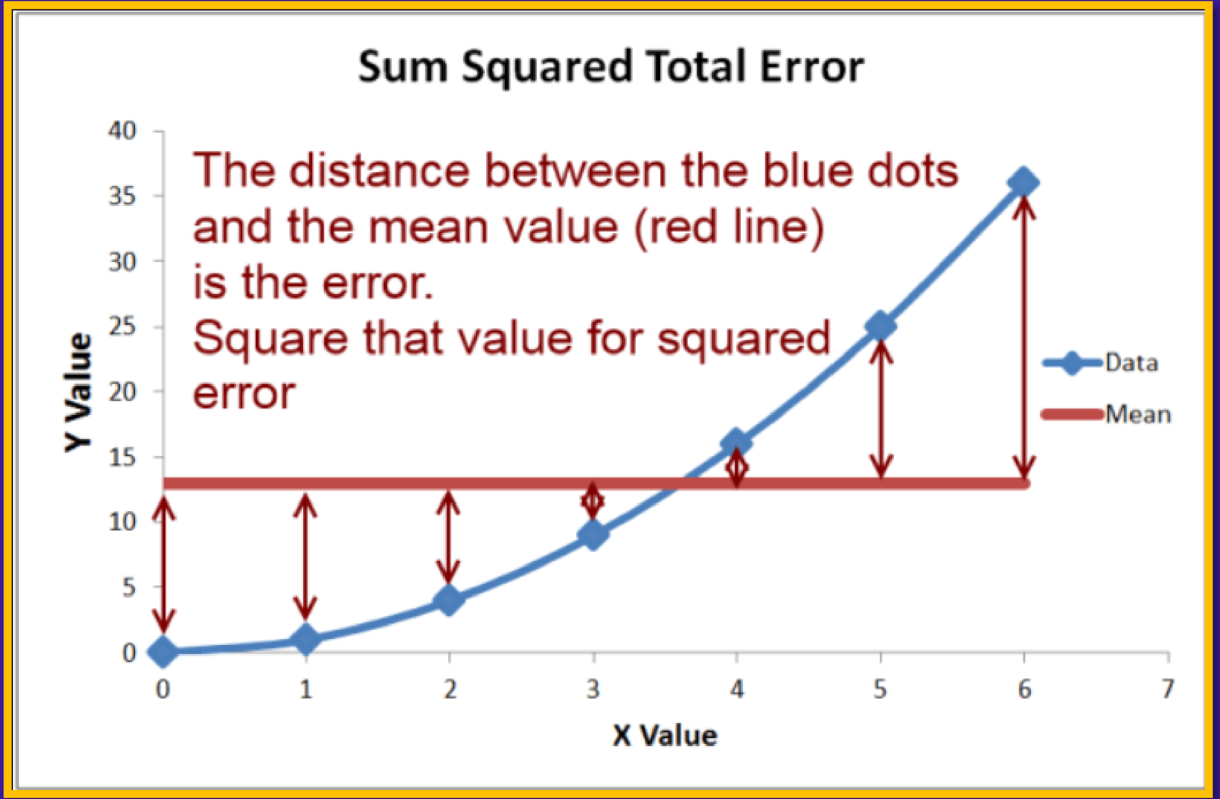
9. Model Evaluation

Root Mean Squared Error (RMSE) — The square root of average of the squares of the difference between the true values and the predicted values.

- The basic idea is to **measure how bad/erroneous the model's predictions are when compared to actual observed values**. So a high RMSE is “bad” and a low RMSE is “good”.

- The lower the difference the better the performance of the model. This is a common metric used for regression analysis.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$



GLA UNIVERSITY MATHURA
Accredited with A Grade by NAAC
12-B Status from UGC

9. Model Evaluation

Explained Variance Score — A measurement to examine how well a model can handle the variation of values in the dataset. Or

Explained variance (also called explained variation) is used to measure the discrepancy between a model (predicted value) and actual data (test value).

The explained variance score explains the **dispersion of errors of a given dataset**. A score of 1.0 is the perfect score.

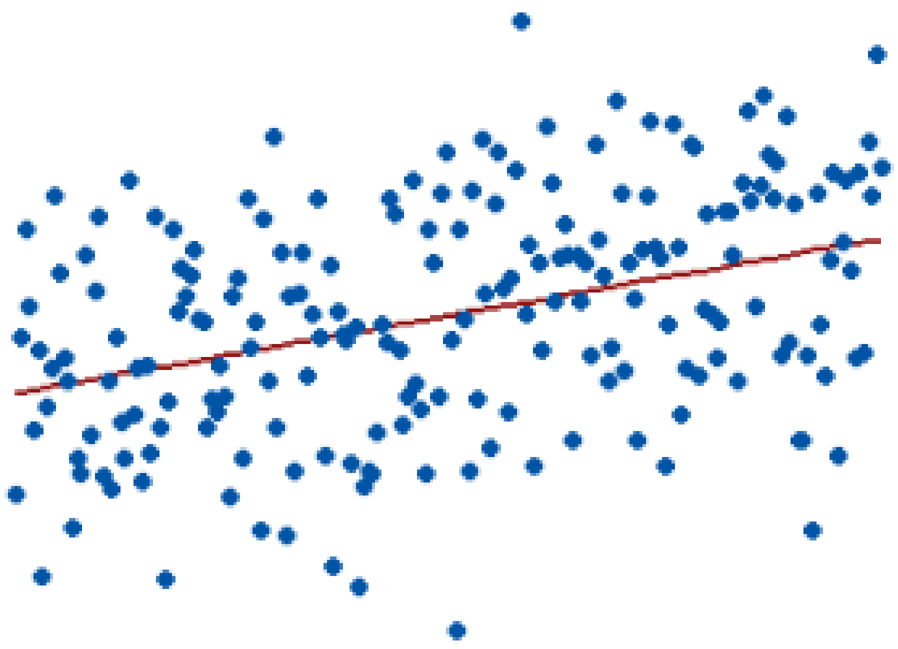
$$explained\ variance(y, \hat{y}) = 1 - \frac{Var(y - \hat{y})}{Var(y)}$$

Note: y= actual data points, and y cap is predicted data point

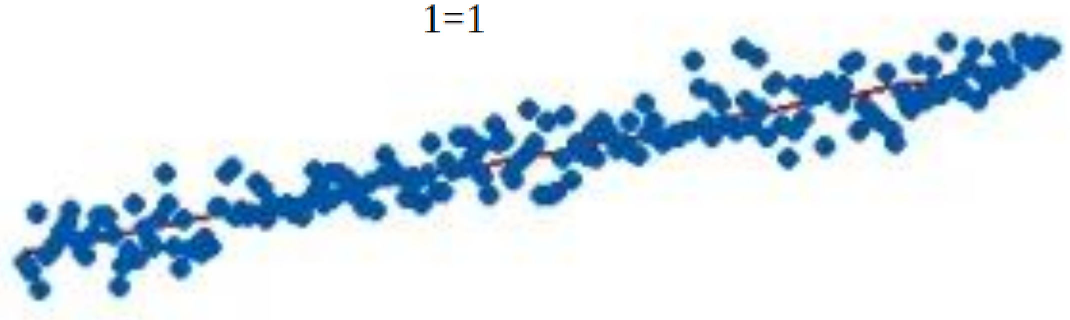
9. Model Evaluation

R^2 Score **(R Squared Value, also called goodness-of-fit measure)** — A measurement to examine how well our model can predict values based on the test set (unknown samples). (Percentage of the dependent variable variation that a linear model explains) The perfect score is 1.0. More than 70% score is good.

$$R^2 = 1 - \frac{\sum_{1=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{1=1}^{N} (y_i - \bar{y})^2}$$



The R-squared value = 15%



The R-squared value = 85%

Note: y= actual data points, y bar is mean value and y cap is predicted data point

Implementing Linear Regression for Salary Prediction

9. Model Evaluation

```
import sklearn.metrics as sm
import numpy as np
print("Root Mean squared error =", round(np.sqrt(sm.mean_squared_error(y_test, y_prdict)), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, y_prdict), 2))
print("R2 score =", round(sm.r2_score(y_test, y_prdict), 2))
```

```
Root Mean squared error = 4834.26
Explain variance score = 0.98
R2 score = 0.97
```

GLA UNIVERSITY MATHURA
Accredited with A Grade by NAAC
12-B Status from UGC

10. Salary Prediction

```
new_salary_pred = reg.predict([[15]])
print (new_salary_pred)
```

Exploring the Different Variations

Assignment

1. Changing the Testing Size
2. Increasing and Decreasing the Size of Data Set

Analyze the results of different possible scenario you can think of

THANKYOU

gaurav.kumar@gla.ac.in 8586968801