Semester Minor Project Report
On

Salary Prediction (In LPA) Using CGPA

Submitted in partial fulfilment of the requirements for the award of the degree of

M.C.A. (Software Engineering)

By

Praful Dhakde

Under the supervision of
DR. C. S. RAI
(Professor, USIC&T)

University School of Information Communication and Technology

Guru Gobind Singh Indraprastha University
Dwarka, Sector 16C, New Delhi

DECLARATION

I hereby declare that the semester progress report entitled "**Salary Prediction (In LPA) Using CGPA"** submitted to USICT, GGSIPU is arecord of original work done by me, under the guidance and supervision of **Prof. C. S. Rai** and it has not formed the basis for the award of any Degree/Diploma/Associateship/ Fellowship or other similar title to any candidate ofany University.

Date: -

Signature

(Praful Dhakde)

CERTIFICATE

This is to certify that Praful Dhakde registered in the MCA(SE) programme of USICT, GGSIPU is permitted to carry out the proposed work presented in this Semester progress report under my guidance. The matter embodied in this proposal has not been submitted earlier.

Signature of Supervisor

DR. C. S. RAI
(Professor, USIC&T**)**

ACKNOWLEDGEMENT

With candour and pleasure, I take the opportunity to express my sincere thanks and obligation to my esteemed guide Prof. C. S. Rai. It is because of her able and mature guidance and co-operation without which it would not have been possible for me to complete my project.

It is my pleasant duty to thank all the staff members of the computer Centre who never hesitated from time to time during the project.

Finally, I gratefully acknowledge the support, encouragement & patience of my family, and as always, nothing in my life would be possible without God, Thank You!

Date: -

Signature

INDEX

1. INTRODUCTION: -

In the modern educational and professional landscape, students often face the challenge of making informed decisions about their future careers. The "Salary Prediction" system, utilizing the power of regression in machine learning within the framework of Linear Programming Approach (LPA), emerges as a sophisticated solution to predict potential job offers or compensation salary based on academic performance, specifically Cumulative Grade Point Average (CGPA).

Regression is used when the target variable is numerical, and the aim is to find the best-fitting line or curve that minimizes the difference between the predicted and actual values. This involves learning the mapping function from the input features to the continuous output. The output can represent various real-world phenomena, such as stock prices, temperatures, sales figures, or any other measurable quantity.

There various types of regression which are comes under the supervised machine learning namely- linear regression, polynomial regression, multiple linear regression, logistic regression etc. But we use linear regression for the prediction of salary (in lpa) because we have limited data (CGPA). So, we use only one independent variable and one dependent variable.

2. OBJECTIVES: -

1. To study concept of regression & its types. Specifically linear regression.
2. To study different algorithms of regression in supervised machine learning.
3. Collect and pre-process datasets (including CGPA and Package) for training and testing classes.
4. Practical implementation of linear regression for Package Prediction (In LPA) using the CGPA of students.
5. Learn about basic python & their libraries for the implementation of above regression algorithms.

3. REGRESSION: -

Regression is a process of finding the correlations between dependent and independent variables. It helps in predicting the continuous variables such as prediction of Market Trends, prediction of House prices, etc.

The task of the Regression algorithm is to find the mapping function to map the input variable(x) to the continuous output variable(y).

Example: Suppose we want to do weather forecasting, so for this, we will use the Regression algorithm. In weather prediction, the model is trained on the past data, and once the training is completed, it can easily predict the weather for future days.

Types of Regression: -

1. Linear Regression.

2. Polynomial Regression.

3. Logistic Regression.

3.1. Linear Regression: - Linear regression is a way to identify a relationship between two or more variable and use this relationship to predict values of one variable for given values(s) of the other variable(s).

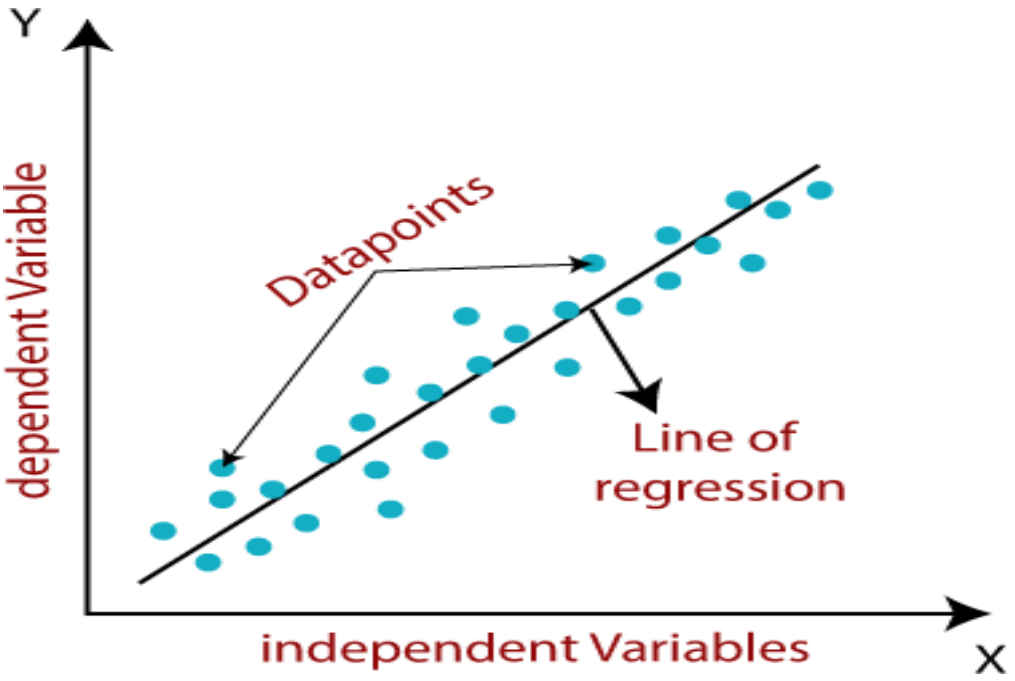
Linear regression assumes the relationship between variables can be modelled through linear equation or an equation of line.

Equation of line will be **Y = AX + B**

Where, Y is a dependent/regressed variable

X is an independent/regressor variable. A and B is a constant for line which are called intercept & slope respectively. A sloped straight line represents the linear regression model.

$$A = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2} \quad and \quad B = \bar{y} - A\bar{x}$$



Loss Function/Cost Function:-

The loss is the error in our predicted value of m and c. Our goal is to minimize this error to obtain the most accurate value of m and c.

We will use the Mean Squared Error function to calculate the loss. There are three steps in this function:

1. Find the difference between the actual y and predicted y value(y = mx + c), for a given x.
2. Square this difference.
3. Find the mean of the squares for every value in X.

$$E = \frac{1}{n}\sum_{i=0}^{n}(y_i - \bar{y}_i)^2$$

Mean Squared Error Equation

Here y_i is the actual value and $\bar{y}_i$ is the predicted value. Lets substitute the value of $\bar{y}_i$:

$$\frac{1}{n}\sum_{i=0}^{n}(y_i - (mx_i + c))^2$$

Substituting the value of $\bar{y}_i$

So we square the error and find the mean. hence the name Mean Squared Error. Now that we have defined the loss function, lets get into the interesting part — minimizing it and finding m and c.

The Gradient Descent Algorithm:-

Gradient descent is an iterative optimization algorithm to find the minimum of a function. Here that function is our Loss Function. Understanding Gradient Descent –

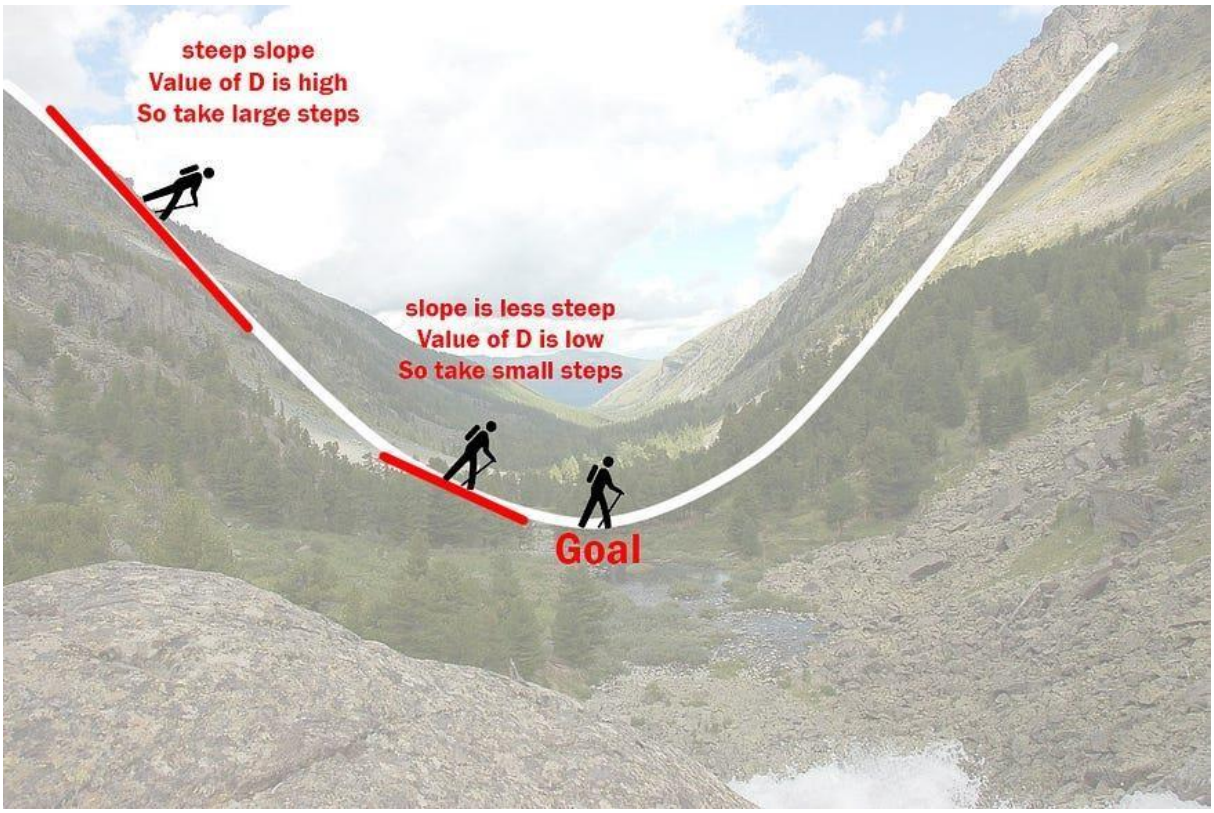


Illustration of how the gradient descent algorithm works

Imagine a valley and a person with no sense of direction who wants to get to the bottom of the valley. He goes down the slope and takes large steps when the slope is steep and small steps when the slope is less steep. He decides his next position based on his current position and stops when he gets to the bottom of the valley which was his goal.

Let's try applying gradient descent to m and c and approach it step by step:

1. Initially let m = 0 and c = 0. Let L be our learning rate. This controls how much the value of m changes with each step. L could be a small value like 0.0001 for good accuracy.
2. Calculate the partial derivative of the loss function with respect to m, and plug in the current values of x, y, m and c in it to obtain the derivative value D.

$$D_m = \frac{1}{n}\sum_{i=0}^{n} 2(y_i - (mx_i + C))(-x_i)$$

$$D_m = \frac{-2}{n} \sum_{i=0}^{n} x_i (y_i - \bar{y}_i)$$

Derivative with respect to m

D_m is the value of the partial derivative with respect to m. Similarly lets find the partial derivative with respect to c, Dc :

$$D_c = \frac{-2}{n} \sum_{i=0}^{n} (y_i - \bar{y}_i)$$

Derivative with respect to c

3. Now we update the current value of m and c using the following equation: $m = m - L * D_m$

$c = c - L * D_m$

4. We repeat this process until our loss function is a very small value or ideally 0 (which means 0 error or 100% accuracy). The value of m and c that we are left with now will be the optimum values.

Now going back to our analogy, m can be considered the current position of the person. D is equivalent to the steepness of the slope and L can be the speed with which he moves. Now the new value of m that we calculate using the above equation will be his next position, and L×D will be the size of the steps he will take. When the slope is more step (D is more) he takes longer steps and when it is less steep (D is less), he takes smaller steps. Finally he arrives at the bottom of the valley which corresponds to our loss = 0.

Now with the optimum value of m and c our model is ready to make predictions !

3.2. Polynomial Regression:- It can handle non-linear relationship among variables by using nth degree of a polynomial. Instead of applying transformation, polynomial regression can be directly used to deal with different levels of curvilinearity.

For example, the second-degree polynomial (called quadratic transformation) is given as: $y = a_0 + a_1x + a_2x^2$ and third-degree polynomial (called cubic transformation) is given as: $y = a_0 + a_1x + a_2x^2 + a_3x^3$.

Generally, polynomials maximum degree 4 are used, as higher order polynomials take some strange shapes & make the curve more flexible. It leads to a situation of overfitting & hence is avoided.

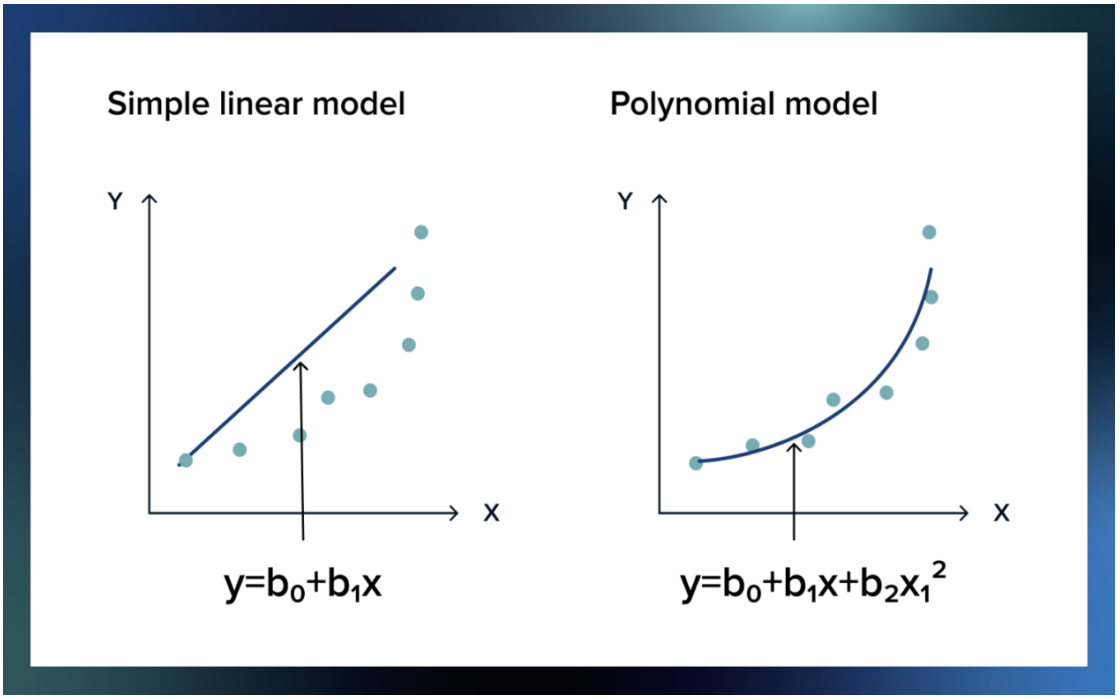
Generalized polynomial equation –

$Y = a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots\ldots\ldots + a_nx^n$.

Consider the polynomial of 2nd degree. And equation is given by:

$$Y = a_0 + a_1x + a_2x^2$$

Where coefficients a_0, a_1, and a_2 are calculated using the formula: $a = x^{-1} B$.



Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as nth degree polynomial.

It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression. It is a linear model with some modification in order to increase the accuracy. The dataset used in Polynomial regression for training is of non-linear nature. It makes use of a linear regression model to fit the complicated and non-linear functions and datasets. Hence, "In Polynomial regression, the original features are converted into Polynomial features of required degree (2,3,..,n) and then modelled using a linear model."

Need of Polynomial Regression:

1. If we apply a linear model on a linear dataset, then it provides us a good result as we have seen in Simple Linear Regression, but if we apply the same model without any modification on a non-linear dataset, then it will produce a drastic output. Due to which loss function will increase, the error rate will be high, and accuracy will be decreased.
2. So for such cases, where data points are arranged in a non-linear fashion, we need the Polynomial Regression model. We can understand it in a better way using the below comparison diagram of the linear dataset and non-linear dataset.

3.3. Logistic Regression:- Logistic regression is typically used for binary classification to predict two discrete classes, e.g., pregnant or not pregnant. To do this, the sigmoid function is added to compute the result and convert numerical results into an expression of probability between 0 and 1.

Sigmoid Function: - Is this person a “potential customer” or “not a potential customer?” Machine learning accommodates such questions through logistic equations, and specifically through what is known as the sigmoid function. The sigmoid function produces an S-shaped curve that can convert any number and map it into a numerical value between 0 and 1, but it does so without ever

reaching those exact limits. A common application of the sigmoid function is found in logistic regression. Logistic regression adopts the sigmoid function to analyse data and predict discrete classes that exist in a dataset. Although logistic regression shares a visual resemblance to linear regression, it is technically a classification technique. Whereas linear regression addresses numerical equations and forms numerical predictions to discern relationships between variables, logistic regression predicts discrete classes.

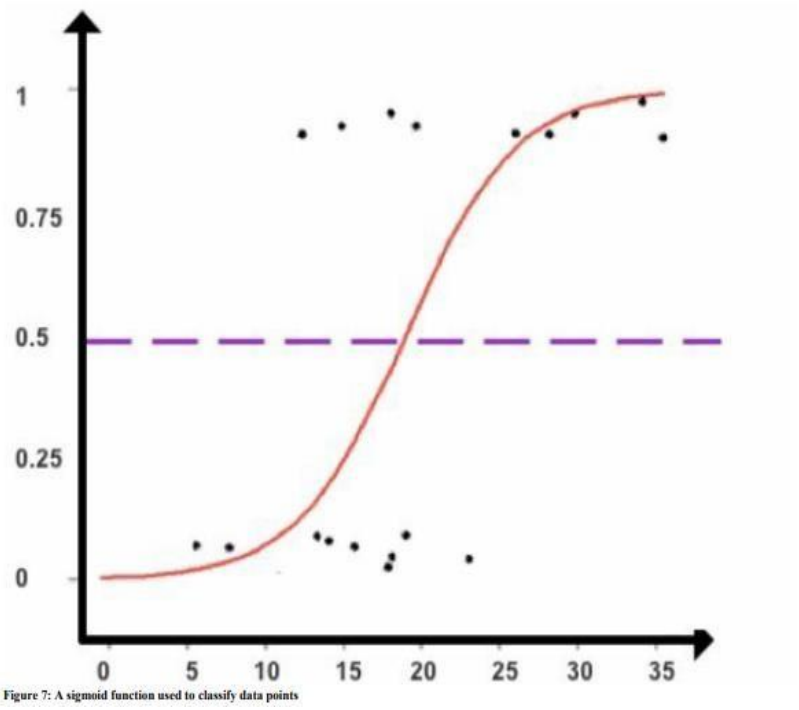


Figure 7: A sigmoid function used to classify data points

This is required sigmoid function (logistic regression)- $$y = \frac{1}{1+e^{-x}}$$

Where: y = dependent variable, x = independent variable, e = Euler's constant, In a binary case, a value of 0 represents no chance of occurring, and 1 located between 0 and 1 can be calculated according to how close they rest to 0 (impossible) or 1 (certain possibility) on the scatterplot.

NOTE:- We use Linear Regression in this model because we have only one independent variable (in form of CGPA).

4. METHODOLOGY:-

4.1. Data Collection:-

(a). Define the Problem: Understand the problem you are trying to solve using linear regression. Determine what you want to predict (the dependent variable) and what features you'll use to make that prediction (the independent variables).

(b). Identify Data Sources: Determine where you can obtain the necessary data. This could be from databases, APIs, data files, or manual data entry.

(c). Data Gathering: Once you've identified potential sources, collect the data. This might involve querying databases, using web scraping techniques to gather data from websites, or extracting data from files.

4.2. Data Preprocessing:-

(a). Split the data into training and testing sets. The training set is used to train the model, while the testing set is used to evaluate its performance.

(b). Check for missing values in the data and decide how to handle them (e.g., imputation, removal, or using algorithms that can handle missing values).

(c). Clean the data by removing outliers, correcting errors, and standardizing formats.

(d). Create new features or transform existing ones to improve model performance. This could involve scaling, normalization, or creating polynomial features.

(e). Select the most relevant features for your model. This could be done through domain knowledge, statistical tests, or automated feature selection algorithms.

5. Implementation:-

The implementation phase involves coding the system using a programming language, such as Python, and relevant libraries like scikit-learn. The dataset is split into training and testing sets, and the linear regression model is trained on the former.

Describe the steps taken to handle missing values, outliers, and noise in the dataset. Explain if any normalization or standardization techniques were applied to scale the features. Detail any feature engineering techniques used to enhance the predictive power of the model.

5.1. Algorithm and Python Implementation:-

(1). Importing required libraries:

[121]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

(2). Reading the data:

[122]:
```
df = pd.read_csv('placement.csv')
```

[123]:
```
df.head()
```

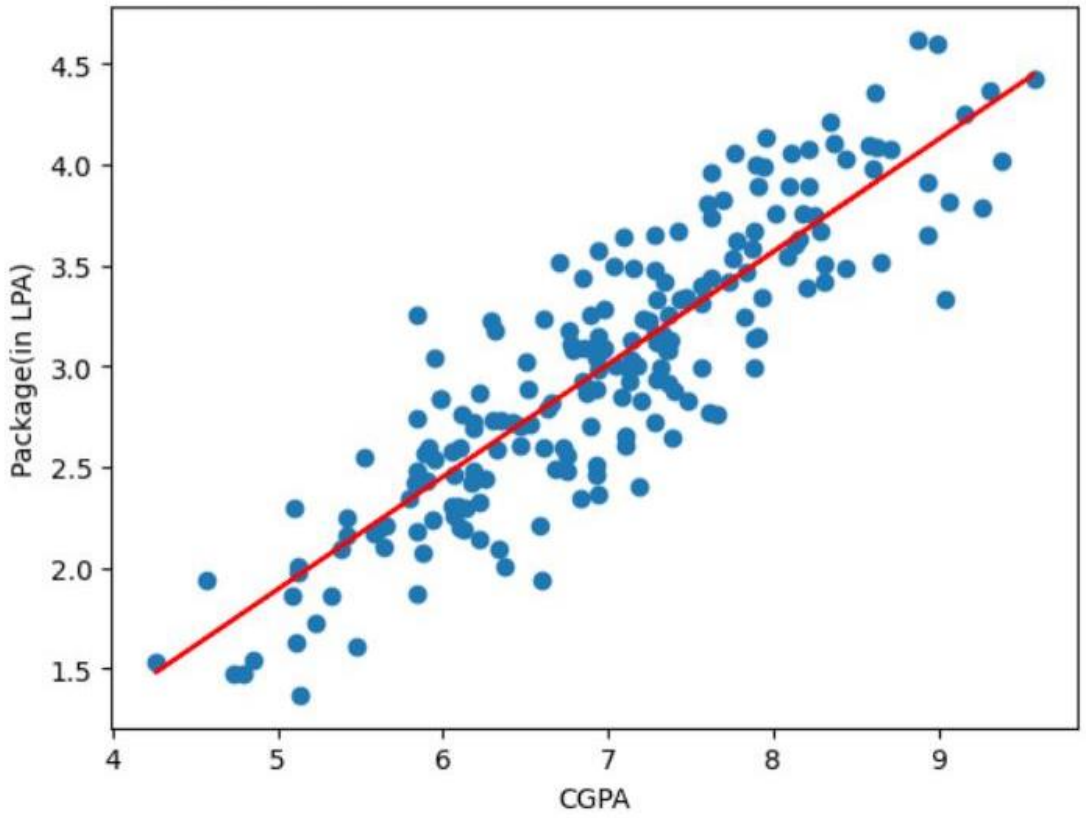
[123]:

	cgpa	package
0	6.89	3.26
1	5.12	1.98
2	7.82	3.25
3	7.42	3.67
4	6.94	3.57

(3). Plot CGPA and Package(in LPA) on X and Y Respectively:

```
plt.scatter(df['cgpa'],df['package'])
plt.plot(X_train,lr.predict(X_train),color='red')
plt.xlabel('CGPA')
plt.ylabel('Package(in LPA)')
```

```
Text(0, 0.5, 'Package(in LPA)')
```



(4). Fetching X and Y:

```
X = df.iloc[:,0].values
Y = df.iloc[:,1].values
```

(5). Splitting the data into training and testing set:

```
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=2)
```

(6). Creating a class to bundle all this into a single box, initialized m and b with 0.

[70]:

```
class MYLR:
    def __init__(self):
        self.m = None
        self.b = None
```

(7). Created a fit method that will take the data and use the formula mentioned above for m and b.

```
def fit(self,X_train,Y_train):

    num = 0
    den = 0

    for i in range(X_train.shape[0]):

        num = num + ((X_train[i] - X_train.mean())*(Y_train[i] - Y_train.mean()))
        den = den + ((X_train[i] - X_train.mean())*(X_train[i] - X_train.mean()))

    self.m = num/den
    self.b = Y_train.mean() - (self.m * X_train.mean())
    print(self.b)
    print(self.m)
```

(8). Created a prediction method that will take a test point and apply the value of m and b we got (Formula : y = mx + b)

```
def predict(self,X_test):
    print(X_test)
    return (self.m * X_test + self.b)
```

6. Results:-

Created an object for the class, using which we can call or use all the methods (fit and predict). Using the fit method which applies data points to the formula. Predicting using the prediction method.

Here we first create an object(name of object – "lr") for the class (name of class – "MYLR") after that we fit it for X_tarin and Y_train and find the value of m and b then we predicted the salary (in LPA) by the help of test part of data. At last we are becomes able to predict the salary (in LPA) for a random CGPA (9.8). As shown in below picture

```
[80]: lr = MYLR()

[81]: lr.fit(X_train,Y_train)

-0.8961119222429152
0.5579519734250721

[86]: print(lr.predict(X_test[1]))

7.15
3.0932446877463504

[85]: print(lr.predict(9.8))

9.8
4.571817417322792
```

Clearly state why linear regression was chosen for this task, emphasizing the suitability of linear regression due to having limited data (CGPA) with one independent variable (CGPA) and one dependent variable (Salary).

7. Conclusion:-

In this study, we explored the predictive power of CGPA in estimating salaries of students using linear regression. The objective was to develop a model that can effectively predict salaries based on CGPA, providing valuable insights for students and recruiters in the job market.

In conclusion, this study underscores the value of CGPA in predicting salaries and provides a foundational framework for further research in the field of predictive modelling in education and employment. By leveraging CGPA as a predictive factor, students can make informed decisions about their career paths, while recruiters can optimize their hiring processes to identify top talent effectively.

8. REFERENCES: -

[1]. Neural Networks & Machine Learning by “Simon Haykin”.

[2]. Machine Learning by “Saikat Dutt, Subramanian Chandramouli, Amit Kumar Das”.

[3]. Machine Learning for Absolute Beginners by “Oliver Theobald”.

[4]. Hands-On Machine Learning with Scikit-Learn, Kera’s, and TensorFlow" by Aurélien Géron.

[5]. Research paper namely “Kaggle” for datasets.