# PHASE 4:

## AI-DRIVEN EXPLORATION AND PREDICTION OF COMPANY REGISTRATION TRENDS WITH REGISTER OF COMPANIES

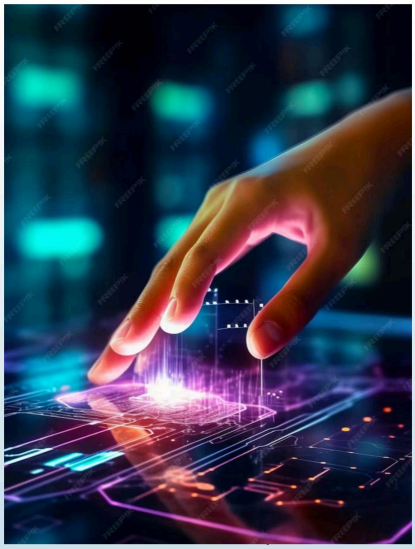
**By**

**E.Manu jeeva**

**Reg.No:513421106026**

**University college of engineering kanchipuram**

# INTRODUCTION:



To build the project by performing feature engineering activity, model training and evaluation

## FEATURE ENGINEERING:

Feature engineering includes remodeling raw data into a format that successfully represents the underlying patterns within the data. It involves selecting, combining, and crafting attributes that capture the relationships between variables, enhancing the predictive power of machine learning models. These engineered features act as the input for algorithms, using progressed performance and robustness.

# Code:

```
data={‘company name’ :[‘Niko resource limited’, ‘Tata and Lyte industry’, ‘Oil and gas exploration company’, ‘Advantmed’, ‘Nippon signal co ltd’] , “Year of reg’ :[‘1998’, ‘2001’, ‘2002’, ‘2004’, ‘2006’,], ‘Registered state’ :[‘Gujarat’, ‘Gujarat’, ‘Gujarat’, ‘Gujarat’, ‘Gujarat’], ‘salary’ :[‘25000’, ‘30000’, ‘40000’, ‘25000’, ‘100000’]}
df = pd.DataFrame(data)
df
```

let us start by building a function to calculate the coefficients using standard formula for calculation using linear regression model

```
Import matplotlib.pyplot as plt
import numpy as np
def simple_linear_regression(x,y):
n=np.size(x)
mean_x = np.mean(x)
mean_y = np.mean(y)
xy = np.sum(y*x) - n*mean_y*mean_x
xx = np.sum(x*x) – n*mean_x*mean_x
m = xy / xx
c = mean_y – m*mean_x
return m,c
```

```
x= df['length'].to_numpy()
y= df['price'].to_numpy()
m,c = simple_linear_regression(x,y)
y_pred = c+m*x
plt.plot(x, y_pred, color = "g", label='ssalary
prediction')
plt.scatter(df['length']
plt.ylabel('salary')
plt.legend(bbox_to
-anchor=(1,1))
plt.show()
```

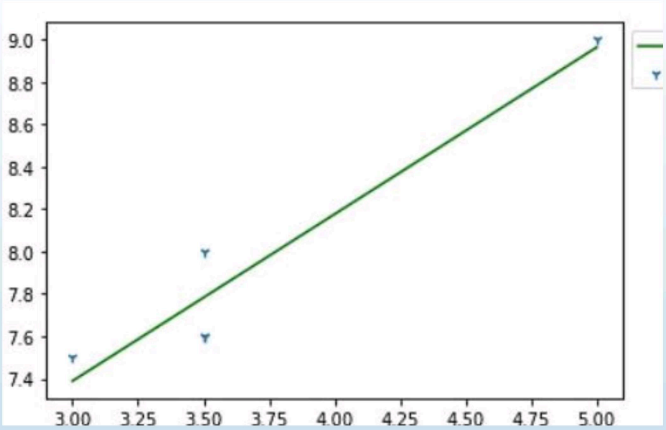
df[‘size’]= df[‘breadth’]*df[‘length’]
df[[‘company name’, ‘salary’, ‘year of reg’]]

we now use this feature to build a new simple linear regression model

x=df[‘year of reg].to_numpy()
y=df[‘price’].to_numpy()
m,c = simple_linear_regression(x,y)
y_pred = c+m*x

```
Plt.plot(x, y_pred, color='g',
label='salary prediction')
plt.scatter(df['year of reg'].to
numpy(),y, marker='1', label='training
set')
plt.xlabel('year of reg')
plt.ylabel('salary;)
plt.legend(bbox_to_anchor=(1,1))
plt.show()
```

The graphic image of the prediction :



# Model training:

1.begin with existing data
2.analyze data to identify patterns
3.make predictions

x_train: It is used to represent features for the training data. Once the model is trained enough with the relevant training data, it is tested with the test data. We can understand the whole process of training and testing in three steps, which are as follows:

Feed: Firstly, we need to train the model by feeding it with training input data.
Define: Now, training data is tagged with the corresponding outputs (in Supervised Learning), and the model transforms the training data into text vectors or a number of data features.
Test: In the last step, we test the model by feeding it with the test data/unseen dataset. This step ensures that the model is trained efficiently and can generalize well.

For training and testing data we use

```
xtrain.shape
xtest.shape
ytrain.shape
ytest.shape
from sklearn.metrics import accuracy_score
from sklearn import svm
clf = svm.svc()
clf.fit(x,y)
y_prediction = clf.predict(x test)
score = accuracy_score(ytest,y_pred)
print(score)
```

## Evaluation:

The Dataset Evaluation Form is used to gather information on prospective datasets for inclusion on the Portal. Data Custodians and Data Owners can use the completed form to recommend whether the identified dataset, or portions of the dataset would be suitable for public release. Once completed, the Evaluation Form may be retained within your branch/division for future reference.

## Conclusion:

To build a project in the dataset of company registration was done and the activities such as featured engineering, model training, evaluation was also done.