# Part 3 – Simple Linear Regression

## EDA For Regression

When performing **EDA for Simple Linear Regression** it is important to explore:

* The Target Variable
* The Feature
* Feature-Target Relationship

#### Histogram

**sns.histplot(dataframe[‘price’]);** # shows distribution of prices

#### Box Plot

**sns.boxplot(x=dataframe[‘price’]** # quartile observations

**sns.despine();** # removes top and right borders

**sns.boxenplot(x=dataframe[‘charges’])** # more detail on distribution

## Simple Linear Regression

Ŷ = β0 + β1x # predicted

y = β0 + β1x + Ɛ # actual

**Least Squared Error**

The line that best fits the data, minimizing the least squared error. Also called **ordinary least squares**.

### Model Testing and Validation

#### Data Splitting

X and y must be split in the main data set. Use train-test-split to develop X\_train, X\_test, y\_train, and y\_test.

#### Bias – Variance Tradoff

**Bias** – How much the model fails to capture the relationships in the training data.

**Variance** – How much the model fails to generalize the test data.

### Simple Linear Regression in Python

**SciKit Learn**

**## Importing the libraries**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import pandas as pd**

**import seaborn as sns**

**## Importing the dataset**

**dataset = pd.read\_csv('Data/Salary\_Data.csv')**

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

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

**or**

**df = pd.read\_csv("Advertising.csv")**

**df.head()**

**X = df['total\_spend']**

**y = df['sales']**

**sns.regplot(data=df, x = 'total\_spend', y ='sales')**

**## Splitting the dataset into the Training set and Test set**

**from sklearn.model\_selection import train\_test\_split**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 11)**

**## Training the Simple Linear Regression model on the Training set**

**from sklearn.linear\_model import LinearRegression**

**regressor = LinearRegression()**

**regressor.fit(X\_train, y\_train)**

**## Predicting the Test set results**

**y\_pred = regressor.predict(X\_test)**

**## Visualising the Training set results**

**plt.scatter(X\_train, y\_train, color = 'red')**

**plt.plot(X\_train, regressor.predict(X\_train), color = 'blue')**

**plt.title('Salary vs Experience (Training set)')**

**plt.xlabel('Years of Experience')**

**plt.ylabel('Salary')**

**plt.show()**

**## Visualising the Test set results**

**plt.scatter(X\_test, y\_test, color = 'red')**

**plt.plot(X\_train, regressor.predict(X\_train), color = 'blue')**

**plt.title('Salary vs Experience (Test set)')**

**plt.xlabel('Years of Experience')**

**plt.ylabel('Salary')**

**plt.show()**

**## Predicting a single observation**

**print(regressor.predict([[12]]))**

**## Getting the final linear regression equation with the values of the coefficients**

**print(regressor.coef\_)**

**print(regressor.intercept\_)**