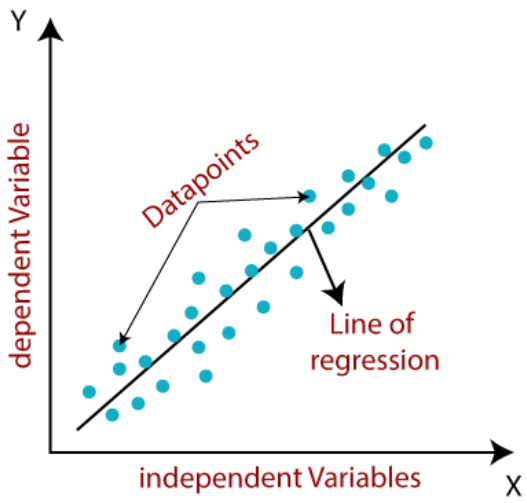
3. Linear Regression Algorithm

3.1 Definition:

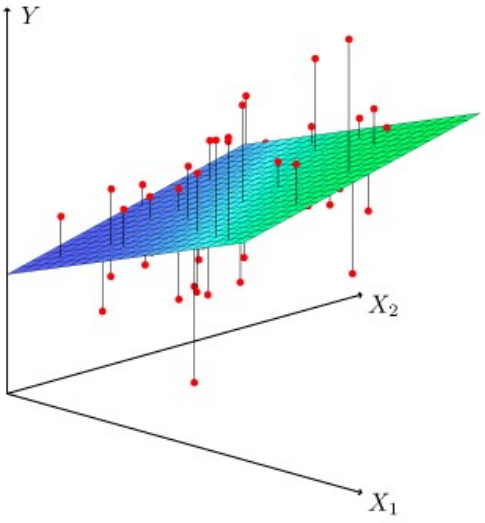
Linear regression is one of the simplest and most commonly used machine learning algorithms. It is used to find a linear relationship between two or more variables. It is used to predict a continuous outcome based on input variables that are either continuous or categorical.

Regression line



Simple linear regression

Regression plane



Multiple linear regression

3.2 How it Works:

Linear regression tries to model the linear relationship between input(s) x_i variables and an output y variable by fitting a straight line (or hyperplane) to the data called hat-y ($\hat{y}$). The general linear relationship between inputs and output can be represented as:

$$\hat{y} = \hat{w}_0 + \sum_{i=1}^{n} \hat{w}_j x_j = \hat{w}_0 + \hat{w}_1 x_1 + \hat{w}_2 x_2 + \ldots + \hat{w}_n x_n$$

Where:

$\hat{w}_0$: is the basis or y-intercept

x_i: is the input features

The goal of linear regression is to choose the values of $\hat{w}_0$ and $\hat{w}_i$ that best model the relationship between x_i and y. What does this mean is that the coefficients $\hat{w}_0$ and $\hat{w}_i$ are learned from the data using an optimization algorithm such as ***Gradient Descent*** or ***Normal Equation***. This involves iteratively updating the coefficient values to reduce the error or residual sum of squares between the actual data and the fitted line. This is known as the least squares optimization objective.

Once the model is trained, it can be used to make predictions for new input data points. The predicted output (ŷ) is calculated using the learned coefficient values and the input data.

3.3 Simple Linear Regression:

Simple linear regression is a regression technique in which the independent variable has a linear relationship with the dependent variable.

The linear relationship represent a straight line model as following:

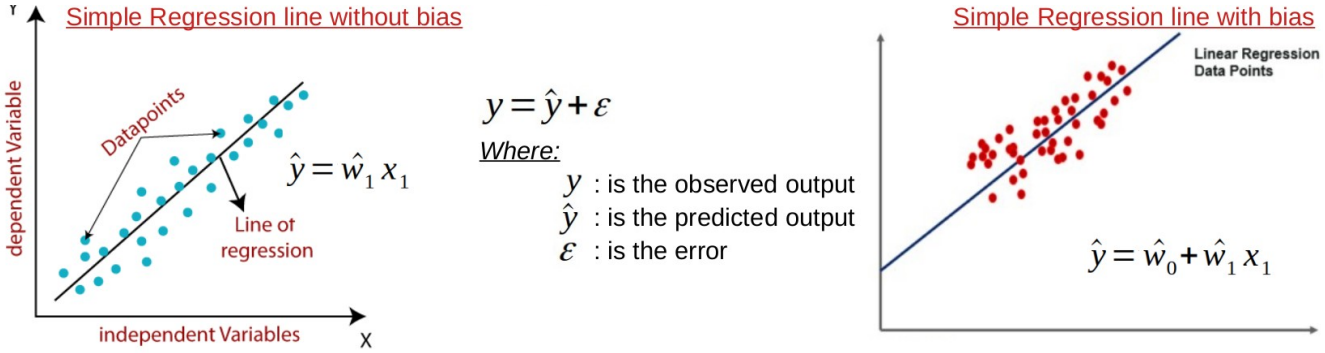
$$\hat{y} = \hat{w}_0 + \hat{w}_1 x$$

Where:

$\hat{w}_0$: is the bias or y-intercept

$\hat{w}_1$: is the slope of the line

x: is the input feature



The straight line in the diagram is the best fit line. The main goal of the simple linear regression is to consider the given data points and plot the best fit line to fit the model in the best way possible to ***minimize some measure of error or cost function***.

3.4 The Mean Square Error (MSE) Cost Function:

In machine learning the most popular measure of error is the ***mean squared error (MSE),*** which is the average of squared error occurred between the predicted values $\hat{Y}_i$ and actual or observed values Y_i.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Where:

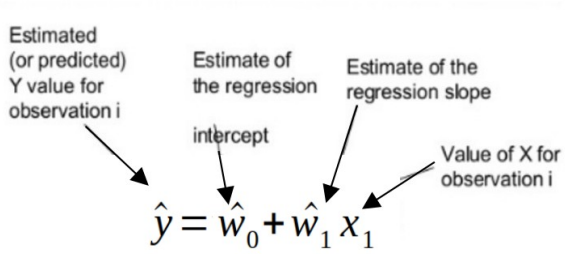
MSE = mean square error

n = number of data points

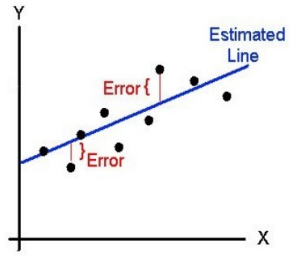
Y_i = observed values

$\hat{Y}_i$ = predicted values

In linear regression, both ***gradient descent*** and the ***least squares method*** are commonly used for estimating the parameters of the regression model that minimize the ***MSE***.



Simple Linear Regression Mathematical Definition



Simple Linear Regression Line Visualization

$$MSE = \frac{\sum(\overbrace{y_i - \hat{y}_i}^{\text{Error } \varepsilon})^2}{n}$$

3.4.1 Gradient Decent Method:

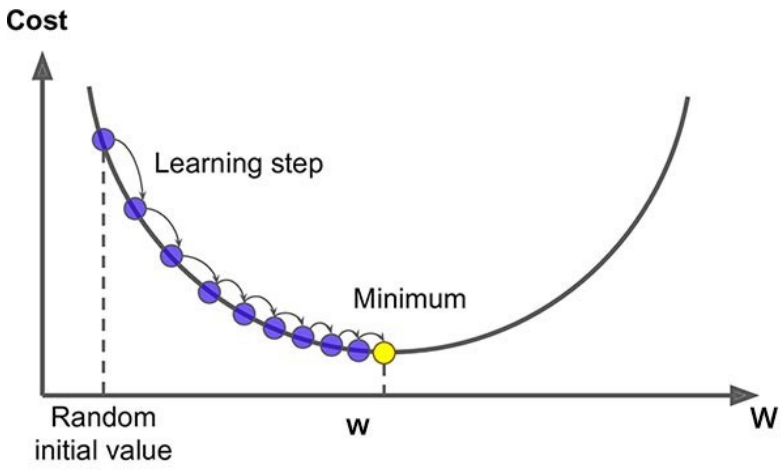
It is a method of updating $\hat{w}_0$ and $\hat{w}_1$ values to reduce the ***MSE***. The idea behind this is to keep iterating the $\hat{w}_0$ and $\hat{w}_1$ values until we reduce the ***MSE*** to the minimum.

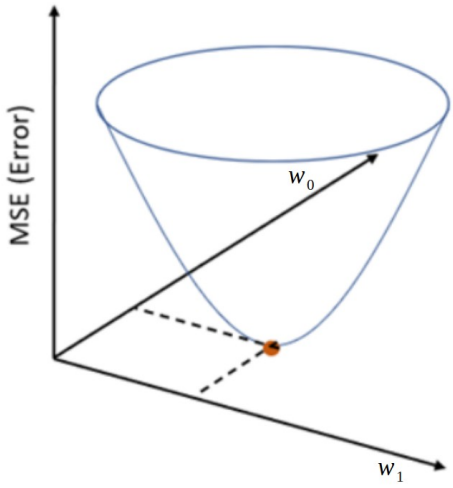
To update $\hat{w}_0$ and $\hat{w}_1$, we take gradients from the cost function. To find these gradients, we take partial derivatives with respect to $\hat{w}_0$ and $\hat{w}_1$. These partial derivatives are the gradients and are used to update the values of $\hat{w}_0$ and $\hat{w}_1$.

$$D_{w_0} = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i) \qquad D_{w_1} = \frac{-2}{n}\sum_{i=1}^{n} x_i(y_i - \hat{y}_i)$$

A smaller learning rate η takes closer to the minimum, but it takes more time. In case of a larger learning rate, the time taken is sooner but there is a chance to overshoot the minimum value.

$$w_0 = w_0 - \eta D_{w_0} \qquad w_1 = w_1 - \eta D_{w_1}$$





3.4.2 Least Square Method

The least squares method is a straightforward approach to linear regression. It aims to minimize the sum of the squared differences between the observed target values and the predicted values. The goal is to find the parameters values that minimize the overall residual error.

The least squares method estimates the coefficients by minimizing the sum of squared residuals *SSR*, which is the sum of the squared differences between the observed target values Y_i and the predicted values $\hat{Y}_i$:

$$SSR = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

The least squares solution for the coefficients involves finding the values of $\hat{w}_0$ and $\hat{w}_1$ that minimize the *SSR*.This can be achieved analytically using matrix algebra or be using optimization techniques as following:

$$\hat{w}_1 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

$$\hat{w}_0 = \bar{y} - \hat{w}_1 \bar{x}$$

Where:

x_i = independent variables value

$\bar{x}$ = average of independent variables

y_i = dependent variables value

$\bar{y}$ = average of dependent variables

Example:

As shown in table the data points represent the dependency of the salary on the years of experience. Using ML algorithm, create the fit model to predict the salary according the given year of experience.

Solution:

Years of Experience x_i	Salary (in 1000$) y_i	$(x_i - \bar{x})$	$(y_i - \bar{y})$	$(x_i - \bar{x})(y_i - \bar{y})$	$(x_i - \bar{x})^2$
2	15	-5.56	-30.44	169.24	30.91
3	28	-4.56	-17.44	79.53	20.79
5	42	-2.56	-3.44	8.81	6.55
13	64	5.44	18.56	100.97	29.59
8	50	0.44	4.56	2.01	0.19
16	90	8.44	44.56	376.09	71.23
11	58	3.44	12.56	43.21	11.83
1	8	-6.56	-37.44	245.61	43.03
9	54	1.44	8.56	12.33	2.07
$\bar{x}$ =7.56	$\bar{y}$ = 45.44			∑ = 1037.8	∑ = 216.19

$$\hat{w}_1 = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2} \Rightarrow \hat{w}_1 = \frac{1037.8}{216.19} \Rightarrow \hat{w}_1 = 4.8$$

$$\hat{w}_0 = \bar{y} - \hat{w}_1 \bar{x} \Rightarrow \hat{w}_0 = 45.44 - 4.8 \times 7.56 \Rightarrow \hat{w}_0 = 9.15$$

Hence,

$$\hat{y} = \hat{w}_0 + \hat{w}_1 x \Rightarrow \hat{y} = 9.15 + 4.8x$$

3.5 The Coefficient of Determination (R-Squared):

R-squared or R^2 or coefficients of determination – is the statistical measure to show how close the data are to the fitted regression line.

It is defined as the proportion of variation of data points explained by the regression line or model. It can be calculated as a function of total variation of data points from the regression line (also termed as the Residual Sum of Squares *RSS*) and total variation of data points from the mean (also termed as Total Sum of Square *TSS*). *TSS* measures the total variability of the dependent variable, while *RSS* quantifies the unexplained variability or the sum of squared residuals.

$$R^2 = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

The value of the R-Squared lies in the range of 0 and 1. Closer the value of R-Squared to 1, the better is the regression model. The value of R-Squared increases with the addition of features.

It's important to note that R-squared alone should not be the sole determinant of the model's quality. It does not indicate the correctness of the model or whether the estimated coefficients are statistically significant. Therefore, it's advisable to consider other evaluation metrics and perform statistical tests to assess the overall performance and validity of the linear regression model.

3.6 Limitations:

Linear regression works best when the relationship between x and y is actually linear. It cannot model non-linear relationships. It also cannot handle model multicollinearity in the input variables well.

3.7 Important concepts:

- **Input variables -** The features used to make the predictions. Can be either continuous (numerical) or categorical (discrete) variables.
- **Output variable -** The variable we want to predict. It is always a continuous variable.
- **Coefficients -** The b and w values that define the line of best fit. Learned during the model training.
- **Linear fit -** The linear relationship between the input and output variables defined by the coefficients.
- **Optimization Objective -** The measure used to determine how well the model fits the data. In linear regression, it is the minimization of the sum of squared errors.
- **Gradient descent -** An optimization algorithm used to iteratively update the coefficient values and minimize the loss function.
- **Prediction equation -** The equation used to make predictions for new data points based on the learned coefficients.

3.8 Implementation:

1. Collect the data. You will need a set of input variables x and corresponding output variables y. This is your training data.
2. Choose a cost function. This measures how well your model fits the data. A common choice is mean squared error.
3. Initialize the model parameters $\hat{w}_0$ and $\hat{w}_1$. Usually they are initialized to 0 or small random values.
4. Make a prediction using the model. Calculate $\hat{y} = \hat{w}_0 + \hat{w}_1 x$.
5. Calculate the cost by comparing $\hat{y}$ with the actual y values.
6. Use an optimization algorithm like gradient descent to update the model parameters to minimize the cost. Take small steps in the direction that reduces the cost.
7. Repeat steps 4 through 6 until the cost is minimized or the maximum number of iterations is reached.
8. Make predictions on new data using the fitted model parameters from the training.

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
import matplotlib.pyplot as plt


# Read the data from the CSV file into a DataFrame
data = pd.read_csv('house_prices.csv')

# Separate the input features and output value
X = data.iloc[:,:-1]
y = data.iloc[:,-1]

# Convert the dataframes to numpy arrays
X = np.array(X)
y = np.array(y).reshape(-1,1)

# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)

# Create the model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make a prediction using the test set
y_pred = model.predict(X_test)

# Calculate the Mean Absolute Error
print("MAE:", metrics.mean_absolute_error(y_test, y_pred))

# Calculate the coefficient of determination (R^2) score
print("R^2 Score:", metrics.r2_score(y_test, y_pred))
```