**The links that I read**

1] Wikipedia page on linear regression: <https://en.wikipedia.org/wiki/Linear_regression>

this is a comprehensive overview of linear regression including its definition, the mathematics behind it, various type of linear regression, applications and examples.

* Modelling the relationship between a scalar response and one or more explanatory variables
* Models are often fitted using the least squares approach
* Formula y = b0 + b1\*x
* Formula y = b0 + b1*x1 + b2*x2 + .... + bn\*xn
* Makes a lot of assumptions such as that the variables are related linearly, constant variance and many more
* Application- widely used in biological, behavioral and social sciences to describe the possible relationships between variables.

2] A gentle introduction to linear regression by analytics Vidhya: https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/

Covered basic concepts, assumptions, evaluation metrics and implementation in python. It also gave an explanation of the line of best fit and gradient descent

3]Introduction to linear regression by statistics solution: <https://www.statisticssolutions.com/introduction-to-linear-regression-analysis/>

This article provides a comprehensive introduction to linear regression and its applications. It covers the basics of linear regression, including its definition, assumptions and the least squares method. It also covers a different type of linear regression models, such as simple linear regression, and multiple linear regression.

4] a gentle introduction to linear regression by Jason wang:

<https://towardsdatascience.com/a-gentle-introduction-to-linear-regression-9a15a0ddbb86>

this article provides a gentle introduction to linear regression, including its definition and assumptions. It covers the basics of linear regression, including the least square method and provides a simple example to illustrate the concepts. The article also covers the different types of linear regression.

5]linear regression for machine learning by Jason Brownlee:

<https://machinelearningmastery.com/linear-regression-for-machine-learning/>

this article provides a practical guide to linear regression for machine learning. It covers the basics of linear regression, including applications in machine learning. This article also covers the implementation of linear regression using python and the scikit-learn library

6] linear regression in python by DataCamp: <https://www.datacamp.com/community/tutorials/linear-regression-python>

This article provides a hands-on guide to linear regression in Python. It covers the basics of linear regression, including its definition and assumptions, and provides a step-by-step guide to implementing linear regression in Python using the scikit-learn library. This article also covers the evaluation of linear regression models and the interpretation of regression coefficients.

**Important points**

* *Linear regression is a linear model which assumes the relationship between the input(X) and the output(Y). i.e Y can be calculated with the X value.*
* *If there is a single input it is referred to as simple linear regression, whereas if there are multiple inputs it is referred to as multiple linear regression.*

Linear regression is a basic and widely used statistical technique that is often used as a first step in understanding the relationship between variables. It can be used to predict future values of a response variable based on the values of predictor variables. Linear regression is used in many different fields, including economics, finance.

* The equation for a linear regression model with one independent variable (also known as simple linear regression) is:
* y = b0 + b1\*x
* where:
* y is the dependent variable (the variable being predicted)
* x is the independent variable (the variable used to predict y)
* b0 is the y-intercept (the value of y when x = 0)
* b1 is the slope (the change in y for a one-unit change in x)
* For multiple linear regression (more than one independent variable), the equation would be:
* y = b0 + b1*x1 + b2*x2 + .... + bn\*xn
* where x1, x2, ...., xn are the independent variables and b1, b2, ..., bn are the coefficients.

**BEST-FITTING LINE**

* There are several methods that can be used to find the best-fitting line in a linear regression model, including:

1. Least Squares Method: The most common method for determining the best-fitting line, it minimizes the sum of the squared differences between the predicted values and the actual values.
2. Maximum Likelihood Method: This method uses probability theory to find the line that is most likely to have generated the data.
3. Gradient Descent: This is an optimization algorithm that finds the best-fitting line by iteratively adjusting the coefficients in the direction of the steepest downhill gradient until the minimum of the cost function is reached.
4. Least Absolute Deviations Method: Similar to the least squares method, but instead of the sum of the squared differences, it minimizes the sum of the absolute differences between the predicted and actual values.
5. Ridge Regression: This method is used to reduce the complexity of the model by adding a penalty term to the cost function.
6. Lasso Regression: Similar to Ridge Regression but instead of adding a penalty term to the cost function, it adds a penalty term on the absolute value of the coefficients.
7. ElasticNet: This is a combination of Ridge and Lasso Regression. It adds a penalty term to the cost function which is a linear combination of L1 and L2 regularization.
8. Bayesian Regression: This method uses Bayesian statistics to determine the best-fitting line.

**Why is linear regression used?**

* Linear regression can be used for making predictions and understanding the relationship between variables in the following ways:

1. Prediction: Once a linear regression model is trained, it can be used to make predictions on new data. For example, given a set of independent variables, the model can predict the value of the dependent variable.
2. Understanding the relationship between variables: Linear regression can be used to understand the relationship between a dependent variable and one or more independent variables
3. Identifying important variables: By looking at the magnitude and sign of the coefficients of the model, we can determine which independent variables have the greatest impact on the dependent variable.
4. Model evaluation: Linear regression can be used to evaluate the model's performance. The R-squared value can be used to measure the proportion of the variance in the dependent variable that is predictable from the independent variables. The smaller the mean squared error, the better the model fits the data.
5. Identifying outliers: Linear regression models can be used to identify outliers in the data by analyzing residuals, which are the differences between the predicted values and the actual values. Outliers are data points that have a large residual value, indicating that they do not fit well with the model.

**Evaluation of linear regression model:**

1. R-squared: R-squared is a statistical measure that represents the proportion of the variance in the dependent variable that is predictable from the independent variables. A higher R-squared value indicates a better fit of the model to the data.
2. Mean Squared Error (MSE): MSE is a measure of the average of the squared differences between the predicted values and the actual values. The smaller the MSE, the better the model fits the data.
3. Mean Absolute Error (MAE): MAE is a measure of the average of the absolute differences between the predicted values and the actual values. The smaller the MAE, the better the model fits the data.
4. Root Mean Squared Error (RMSE): RMSE is a measure of the average of the squared differences between the predicted values and the actual values, but it is expressed in the same units as the dependent variable. The smaller the RMSE, the better the model fits the data.

**Limitation of linear regression**

1. **Linearity assumption**: Linear regression assumes that the relationship between the independent variables and the dependent variable is linear. If the relationship is non-linear, the model may not fit the data well and the predictions may be inaccurate.
2. **Outliers**: Linear regression is sensitive to outliers in the data. Outliers can have a large impact on the model coefficients and can lead to biased and inefficient estimates.
3. **Multicollinearity**: Linear regression assumes that the independent variables are not highly correlated with one another. If there is a high correlation among the independent variables, it can lead to unstable estimates of the model parameters.
4. **Non-normality of errors**: Linear regression assumes that the errors are normally distributed. If the errors are not normally distributed, the model's assumptions are violated and the results may not be reliable.
5. **Limited ability to handle categorical variables**: Linear regression is designed to work with continuous variables, and it can be challenging to handle categorical variables. One solution is to create binary/dummy variables from categorical variables.
6. **Limited ability to handle non-linear relationships:** Linear regression can only model linear relationships between variables, so it is not well-suited for modelling non-linear relationships.
7. **Overfitting**: Linear regression can be prone to overfitting if there are too many independent variables, which can lead to poor generalization performance.
8. **Not robust to missing values**: Linear regression is not robust to missing values and the model’s performance might be affected if there are missing values in the data.

BUILD A LINEAR REGRESSION MODEL WITHOUT IN BUILD METHODS

1] TowardsDatascience <https://towardsdatascience.com/simple-linear-regression-in-python-numpy-only-130a988c0212>

This link provides a comprehensive explanation of simple linear regression and how to implement it from scratch in python using Numpy library. It covers the concepts of cost function gradient descent and hypothesis functions and provides code example to illustrate it.

#using Least square method for modelling

|  |
| --- |
| # Building the model |
|  | X\_mean = np.mean(X) |
|  | Y\_mean = np.mean(Y) |
|  |  |
|  | num = 0 |
|  | den = 0 |
|  | for i in range(len(X)): |
|  | num += (X[i] - X\_mean)\*(Y[i] - Y\_mean) |
|  | den += (X[i] - X\_mean)\*\*2 |
|  | m = num / den |
|  | c = Y\_mean - m\*X\_mean |
|  |  |
|  | print (m, c) |

#making prediction—showing the best-fit line in the map

|  |
| --- |
| # Building the model |
|  | X\_mean = np.mean(X) |
|  | Y\_mean = np.mean(Y) |
|  |  |
|  | num = 0 |
|  | den = 0 |
|  | for i in range(len(X)): |
|  | num += (X[i] - X\_mean)\*(Y[i] - Y\_mean) |
|  | den += (X[i] - X\_mean)\*\*2 |
|  | m = num / den |
|  | c = Y\_mean - m\*X\_mean |
|  |  |
|  | print (m, c) |

3] <https://www.edureka.co/blog/least-square-regression/>

# Mean X and Y

mean\_x = np.mean(X)

mean\_y = np.mean(Y)

# Total number of values

n = len(X)

# Using the formula to calculate 'm' and 'c'

numer = 0

denom = 0

for i in range(n):

numer += (X[i] - mean\_x) \* (Y[i] - mean\_y)

denom += (X[i] - mean\_x) \*\* 2

m = numer / denom

c = mean\_y - (m \* mean\_x)

# Printing coefficients

print("Coefficients")

print(m, c)

Coefficients

0.26342933948939945 325.57342104944223

# Plotting Values and Regression Line

max\_x = np.max(X) + 100

min\_x = np.min(X) - 100

# Calculating line values x and y

x = np.linspace(min\_x, max\_x, 1000)

y = c + m \* x

# Ploting Line

plt.plot(x, y, color='#58b970', label='Regression Line')

# Ploting Scatter Points

plt.scatter(X, Y, c='#ef5423', label='Scatter Plot')

plt.xlabel('Head Size in cm3')

plt.ylabel('Brain Weight in grams')

plt.legend()

plt.show()

EVALUATION METRIX

# Calculating Root Mean Squares Error

rmse = 0

for i in range(n):

    y\_pred = c + m \* X[i]

    rmse += (Y[i] - y\_pred) \*\* 2

rmse = np.sqrt(rmse/n)

print("RMSE")

print(rmse)

RMSE

72.1206213783709

# Calculating R2 Score

ss\_tot = 0

ss\_res = 0

for i in range(n):

    y\_pred = c + m \* X[i]

    ss\_tot += (Y[i] - mean\_y) \*\* 2

    ss\_res += (Y[i] - y\_pred) \*\* 2

r2 = 1 - (ss\_res/ss\_tot)

print("R2 Score")

print(r2)

R2 Score

0.6393117199570003

2] Kaggle: <https://www.kaggle.com/code/hoshi7/from-scratch-simple-linear-regression>

This executes linear regression using least square method

def Simple\_Linear\_Regression(x, y):

n = len(x)

x\_mean = x.mean()

y\_mean = y.mean()

B1\_numerator = sum(((x-x\_mean) \* (y-y\_mean)))

B1\_denominator = sum((x-x\_mean)\*\*2)

B1\_hat = B1\_numerator/B1\_denominator

B0\_hat = y\_mean - B1\_hat \* x\_mean

regression\_line = f'y = **{**B0\_hat**}** + **{**B1\_hat**}**X'

return B0\_hat, B1\_hat, regression\_line