**LOGISTIC REGRESSION**

Logistic Regression is a fundamental algorithm used for binary classification tasks. It's widely used in various fields including machine learning, statistics, and social sciences.

**1. Data Preparation:**

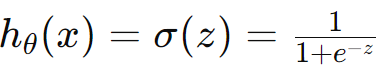
* Collect your dataset consisting of features (independent variables) and target labels (dependent variable).
* Ensure that your data is cleaned and preprocessed, handling missing values, outliers, and encoding categorical variables if necessary.
* Split your data into training and testing sets to evaluate the performance of the model.

**2. Model Initialization:**

* Initialize the parameters of the logistic regression model. These parameters include the weights (coefficients) and bias (intercept).
* Randomly initialize the weights, or initialize them with zeros.

**3. Hypothesis Function:**

* Define the hypothesis function that models the relationship between the features and the probability of the target variable.
* The logistic function (also known as the sigmoid function) is commonly used for this purpose:

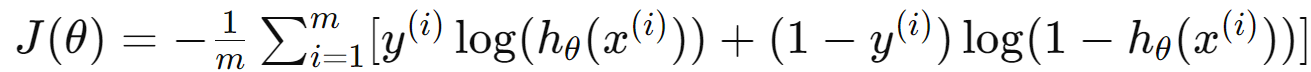




* Where *hӨ(x)* represents the predicted probability that the target variable equals 1 given the input features *x*, and Ө represents the model parameters.

**4. Cost Function:**

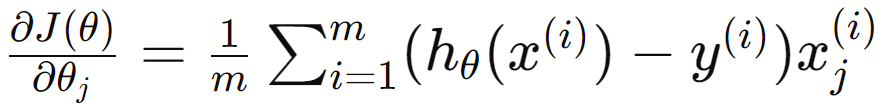
* Define a cost function to measure the error between the predicted probabilities and the actual labels.
* Logistic Regression commonly uses the binary cross-entropy loss function:



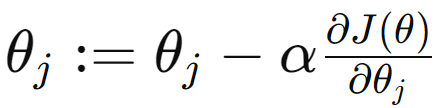
* Where *J(Ө)* represents the cost, *m* is the number of training examples, *y(i)* is the actual label of the training example, and *hӨ(x(i))* is the predicted probability of the *ith* example.

**5. Gradient Descent:**

* Use an optimization algorithm such as Gradient Descent to minimize the cost function.
* Compute the gradient of the cost function with respect to the parameters:



* Update the parameters iteratively using the gradient descent update rule:



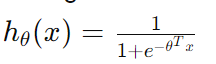
* Where α is the learning rate, controlling the step size of the updates.

**6. Training:**

Iterate the gradient descent process until convergence, i.e., until the change in the cost function becomes negligible or after a predefined number of iterations.

**7. Prediction:**

* Once the model is trained, use it to make predictions on new data.
* Given a new set of features , predict the probability of the target variable being 1 using the learned parameters:



* Classify the instance as 1 if ***hӨ(x)>0.5*** and as 0 otherwise.

**NAÏVE BAYES**

The Naive Bayes algorithm is a simple yet powerful probabilistic classifier based on Bayes theorem with an assumption of independence between features.

**1. Data Preparation:**

* Collect your dataset consisting of features (independent variables) and target labels (dependent variable).
* Ensure that your data is cleaned and preprocessed, handling missing values, outliers, and encoding categorical variables if necessary.
* Split your data into training and testing sets to evaluate the performance of the model.

**2. Calculate Class Priors:**

* Calculate the prior probability of each class in the training set.
* This is done by simply counting the number of occurrences of each class and dividing by the total number of instances.

**3. Calculate Likelihoods:**

* For each class, calculate the likelihood of each feature given the class.
* In Naive Bayes, the assumption is made that the features are conditionally independent given the class. This allows us to calculate the likelihood of each feature independently:



* This can be estimated from the training data using techniques such as frequency counts or probability density estimation (e.g., Gaussian Naive Bayes assumes a Gaussian distribution for continuous features).

**4. Calculate Class Posteriors:**

* Use Bayes theorem to calculate the posterior probability of each class given the features:



* In practice, the denominator *P(x)* is constant for all classes and can be ignored since it only serves as a normalization factor. Thus, the class posterior can be proportional to the product of prior and likelihood:

****

**5. Prediction:**

* Given a new instance with features *x*, calculate the posterior probability of each class using the formula obtained in the previous step.
* Assign the instance to the class with the highest posterior probability.
* Mathematically, this can be represented as:

