**Machine learning assignment**

**Linear Regression**

Linear regression is a data analysis technique that uses a known data value to predict the value of unknown data. It mathematically models the known and unknown variables as a linear equation.

**Linear regression can help with:**

* Foreseeing trends
* Determining future values
* Predicting the impacts of changes

**Type of linear regression**

Multiple linear regression (MLR) is a type of linear regression that establishes the relationship between multiple independent variables and a dependent variable. The independent variables can be continuous or categorical.

Logistic regression is another type of linear regression that involves one dependent variable and two independent variables. The target variable is discrete, such as a binary or ordinal value.

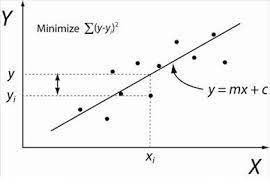
Gradient descent is an iterative optimization algorithm that helps make accurate predictions in linear regression. It's a way to adjust the prediction line so that it fits the data points well.

Linear regression is a statistical method used for modeling the relationship between a dependent variable (target) and one or more independent variables (features or predictors) by fitting a linear equation to the observed data. The goal of linear regression is to find the best-fitting line (or hyperplane in the case of multiple predictors) that minimizes the difference between the observed values and the values predicted by the model.

linear regression equation for a single independent variable is: *y*=*mx*+*b*

where:

* *y* is the dependent variable (target),
* *x* is the independent variable (feature),
* *m* is the slope of the line, and
* *b* is the y-intercept (the value of *y* when *x* is 0).



**Multiple linear regression (MLR)**

Multiple linear regression (MLR) is a statistical technique that analyzes the relationship between a dependent variable and two or more independent variables. The formula for MLR is:

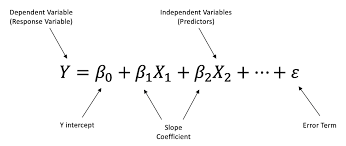
y=a+bx1+cx2+dx3+…

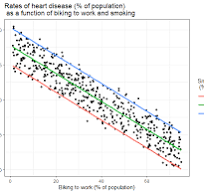
* The variable yi is dependent or predicted
* The slope of y depends on the y-intercept, that is, when xi and x2 are both zero, y will be β0.
* The regression coefficients β1 and β2 represent the change in y as a result of one-unit changes in xi1 and xi2.
* βp refers to the slope coefficient of all independent variables
* ε term describes the random error (residual) in the model.

Where ε is a standard error, this is just like we had for simple linear regression, except k doesn’t have to be 1.

Here are some detail:

* MLR is used when the relationship between variables is complex.
* MLR can account for nonlinear relationships and interactions between variables in ways that simple linear regression can't.
* The objective of MLR is to develop a model that describes a dependent variable y to more than one independent variable.
* Multiple regression analysis shows the correlation between each set of independent and dependent variables.
* To find the best-fit line for each independent variable, multiple linear regression calculates:
  + The regression coefficients that lead to the smallest overall model error
  + The t statistic of the overall model
  + The associated p value





**Logistic regression**

Logistic regression is used for binary [classification](https://www.geeksforgeeks.org/getting-started-with-classification/) where we use [sigmoid function](https://www.geeksforgeeks.org/derivative-of-the-sigmoid-function/), that takes input as independent variables and produces a probability value between 0 and 1.

For example, we have two classes Class 0 and Class 1 if the value of the logistic function for an input is greater than 0.5 (threshold value) then it belongs to Class 1 it belongs to Class 0. It’s referred to as regression because it is the extension of [linear regression](https://www.geeksforgeeks.org/ml-linear-regression/) but is mainly used for classification problems.

**Key Points:**

* Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value.
* It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
* In Logistic regression, instead of fitting a regression line, we fit an “S” shaped logistic function, which predicts two maximum values (0 or 1).

**Logistic Function – Sigmoid Function**

* The sigmoid function is a mathematical function used to map the predicted values to probabilities.
* It maps any real value into another value within a range of 0 and 1. The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the “S” form.
* The S-form curve is called the Sigmoid function or the logistic function.
* In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

**Types of Logistic Regression**

On the basis of the categories, Logistic Regression can be classified into three types:

1. Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
2. Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as “cat”, “dogs”, or “sheep”
3. Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as “low”, “Medium”, or “High”.

**Sigmoid Function**

Now we use the [sigmoid function](https://www.geeksforgeeks.org/derivative-of-the-sigmoid-function/) where the input will be z and we find the probability between 0 and 1. i.e. predicted y.



Sigmoid function

As shown above, the figure sigmoid function converts the continuous variable data into the [probability](https://www.geeksforgeeks.org/probability-gq/) i.e. between 0 and 1.

**The Naive Bayes algorithm**

The Naive Bayes algorithm is a probabilistic machine learning algorithm based on Bayes' theorem. It is primarily used for classification tasks, particularly in natural language processing and text classification. Despite its simplicity, Naive Bayes can be surprisingly effective and computationally efficient. The "Naive" in its name comes from the assumption that features are conditionally independent, given the class label, which simplifies the modeling process.

Here are the key components of the Naive Bayes algorithm:

Bayes' Theorem:

Bayes' theorem describes the probability of an event based on prior knowledge of conditions that might be related to the event. For a classification problem, it is expressed as:

P(Class∣Features)=P(Features)/P(Features∣Class)⋅P(Class)

where:

* *P*(Class∣Features) is the posterior probability of the class given the features.
* *P*(Features∣Class) is the likelihood of observing the features given the class.
* *P*(Class) is the prior probability of the class.
* *P*(Features) is the probability of observing the features.

Naive Bayes Algorithm:

* **Assumption of Feature Independence:**Naive Bayes assumes that features are conditionally independent given the class label. This simplifying assumption facilitates the modeling process.
* **Training:**The algorithm calculates the prior probabilities*P*(Class) and the class-conditional probabilities*P*(Features∣Class) based on the training data.
* **Feature Vector:**Given a new instance with a feature vector, the algorithm calculates the posterior probabilities for each class using Bayes' theorem.
* **Prediction:**The class with the highest posterior probability is predicted as the output.

Types of Naive Bayes Classifiers:

* **Gaussian Naive Bayes:**Assumes that the continuous values associated with each class are normally distributed.
* **Multinomial Naive Bayes:**Suitable for discrete data (e.g., text data) with multiple categories. It models the probability of observing a count of words.
* **Bernoulli Naive Bayes:**Applicable to binary data (e.g., presence or absence of a feature). It models the probability of binary events.

Advantages of Naive Bayes:

* **Simplicity and Speed:**Naive Bayes is computationally efficient and simple to implement.
* **Good Performance with Text Data:**Particularly effective for text classification tasks, such as spam filtering or sentiment analysis.
* **Handles High-Dimensional Data:**Performs well with a large number of features.
* **Applicability to Various Types of Data:**Suitable for both binary and multiclass classification problems.

Limitations of Naive Bayes:

**Assumption of Feature Independence:**The assumption of feature independence may not hold in some real-world scenarios.

**Sensitivity to Irrelevant Features:**Can be sensitive to irrelevant features in the dataset.

**Estimation Issues with Zero Probabilities:**If a feature has not been observed in the training data with a certain class, the conditional probability may become zero. Techniques like Laplace smoothing can be applied to address this issue.

**Decision tree algorithm**

A decision tree algorithm is a machine learning method used for both classification and regression tasks. It builds a tree-like structure by recursively making decisions based on input features. The goal is to create a model that predicts the target variable by learning simple decision rules inferred from the data.

Here are the key components and steps of the decision tree algorithm:

### **Components:**

* Root Node:
  + The topmost node of the tree represents the entire dataset. It is the starting point for making decisions.
* Decision Nodes (Internal Nodes):
  + These nodes represent decisions based on specific features. Each decision node tests the value of a particular feature.
* Branches:
  + The branches emanating from decision nodes represent the possible outcomes of the feature tests.
* Leaf Nodes:
  + The terminal nodes, or leaves, represent the final predicted outcome or value. Each leaf node corresponds to a specific class (in classification) or a predicted value (in regression).
* Decision Rules:
  + The decision tree learns decision rules based on the feature tests at each node. These rules guide the traversal from the root to a specific leaf node.

### **Decision Tree Algorithm Steps:**

* Feature Selection:
  + Choose the feature that best splits the data into subsets. The selection is based on criteria such as information gain, Gini impurity, entropy, or mean squared error, depending on the task (classification or regression).
* Splitting:
  + Split the dataset into subsets based on the selected feature. Each subset corresponds to a branch emanating from the decision node.
* Recursive Process:
  + Recursively repeat the feature selection and splitting process for each subset until a stopping criterion is met. This can be a predefined depth limit, a minimum number of samples per leaf, or other conditions.
* Leaf Node Assignment:
  + When a stopping criterion is reached, assign a leaf node to each subset. The majority class (for classification) or the mean value (for regression) of the target variable in the subset is assigned to the leaf node.
* Prediction:
  + Given a new instance, traverse the decision tree from the root to a leaf node based on the feature tests. The predicted outcome is the class or value associated with the reached leaf node.

### **Types of Decision Trees:**

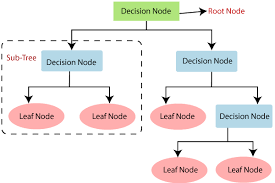
* Classification Trees:
  + Used for categorical target variables. The leaves represent class labels.
* Regression Trees:
  + Used for continuous target variables. The leaves represent predicted numerical values.
* Ensemble Methods (e.g., Random Forests):
  + Combine multiple decision trees to improve predictive performance and robustness.

### **Advantages of Decision Trees:**

* Interpretability:
  + Decision trees are easy to interpret and visualize, making them suitable for explaining decisions to non-experts.
* Non-linearity:
  + They can capture non-linear relationships between features and the target variable.
* Handling Missing Values:
  + Decision trees can handle missing values by selecting alternative paths during the traversal.
* Feature Importance:
  + They provide information about feature importance, which can aid in feature selection.

### **Limitations of Decision Trees:**

* Overfitting:
  + Decision trees can be prone to overfitting, especially when the tree is deep and captures noise in the training data.
* Sensitivity to Small Variations:
  + Small variations in the data can lead to different tree structures.
* Limited Expressiveness:
  + Decision trees may not be expressive enough to represent complex relationships in certain datasets.
* Global Optimality:
  + Decision trees aim for locally optimal splits at each node, which may not result in a globally optimal tree structure.



Decision Tree Splitting Criteria:

The decision tree algorithm uses a splitting criterion to determine the feature and the corresponding value to split the data at each internal node. Common splitting criteria include:

**Gini Impurity:**Measures the impurity or disorder in a dataset. The algorithm selects the feature and value that minimizes the Gini impurity for the resulting subsets.

**Entropy:**Measures the level of disorder or uncertainty in a dataset. The algorithm chooses the feature and value that minimizes the entropy for the resulting subsets.

**Information Gain:**Based on entropy, information gain quantifies the reduction in uncertainty achieved by splitting the data on a specific feature. The algorithm selects the feature that maximizes information gain.

**Gain Ratio:**Similar to information gain but addresses the bias towards features with a large number of values. It considers the size of each subset when selecting the feature.

**Support Vector Machines (SVM)**

Support Vector Machines (SVM) is a supervised machine learning algorithm used for classification and regression tasks. It's particularly effective in high-dimensional spaces and is widely used in various domains, including image classification, text categorization, and bioinformatics. The primary objective of SVM is to find a hyperplane that best separates data points belonging to different classes.

Concepts:

* Hyperplane:
  + In SVM, a hyperplane is a decision boundary that separates data points of one class from another. For a binary classification problem in a two-dimensional space, the hyperplane is a line. In higher dimensions, it becomes a flat affine subspace.
* Support Vectors:
  + Support vectors are the data points that lie closest to the hyperplane. They are crucial in defining the optimal hyperplane and determining the margin.
* Margin:
  + The margin is the distance between the hyperplane and the nearest data point from either class. SVM aims to find the hyperplane that maximizes this margin.
* Kernel Trick:
  + The kernel trick is used to transform the input features into a higher-dimensional space, allowing SVM to find a hyperplane that can handle non-linearly separable data. Common kernel functions include linear, polynomial, radial basis function (RBF or Gaussian), and sigmoid.

SVM for Classification:

* Linear SVM:
  + For linearly separable data, SVM seeks a linear hyperplane that separates the classes with the maximum margin. The decision boundary is defined by
  + **w**⋅**x**+*b*=0, where
  + **w** is the weight vector and
  + *b* is the bias term.
* Non-linear SVM:
  + In cases where the data is not linearly separable, SVM employs the kernel trick to map the input features into a higher-dimensional space. This allows for finding a hyperplane that can separate the classes in the transformed space.

SVM for Regression (Support Vector Regression - SVR):

* Linear SVR:
  + Similar to linear SVM for classification, linear SVR aims to find a hyperplane that fits the data points with the maximum margin. The goal is to minimize the deviation of data points from the hyperplane.
* Non-linear SVR:
  + Non-linear SVR uses the kernel trick to handle non-linear relationships between features and the target variable.

Training and Optimization:

* Objective Function:
  + The training of SVM involves solving an optimization problem that aims to maximize the margin while minimizing the classification error.
* Regularization:
  + SVM includes a regularization parameter (*C*) that controls the trade-off between achieving a smooth decision boundary and classifying training points correctly. A smaller *C* allows for a wider margin but may tolerate some misclassifications, while a larger *C* aims for a narrower margin with fewer misclassifications.

Advantages of SVM:

* Effective in High-Dimensional Spaces:
  + SVM performs well in datasets with a large number of features.
* Versatility:
  + SVM can be applied to both linearly and non-linearly separable data through the use of different kernels.
* Robustness to Overfitting:
  + SVM is less prone to overfitting, especially in high-dimensional spaces.
* Works Well with Small Datasets:
  + SVM can be effective even with small datasets.

Limitations of SVM:

* Computational Intensity:
  + Training an SVM on large datasets can be computationally intensive.
* Sensitivity to Noise:
  + SVM can be sensitive to noise in the dataset.
* Choice of Kernel:
  + The choice of the kernel and its parameters can impact the performance of SVM. It often requires tuning.

