**NAME: OLUMADE MODUPE (group 8)**

**Question:**

Find out 5 machine learning models asides from Logistic and Linear Regression with their documentation.

|  |  |  |
| --- | --- | --- |
| **S/N** | **MODEL** | **DOCUMENTATION** |
| **1** | **NAIVE BAYES** | Naive Bayes classifiers are collection of classification algorithms based on Bayes’ Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.  A naive Bayes classifier assumes that the presence of a feature in a class is independent of other features. It classifies new data based on the highest probability of it belonging to a particular class. This probability is determined by the probabilities of each feature.  It is a simple supervised machine learning algorithm that uses Bayes’ theorem with strong independence assumptions between the features. Here’s how it works:  **Assumption of Independence**: Naive Bayes assumes that each input variable is independent of the value of any other feature, given the class variable. For instance, when classifying fruits, it considers features like color, roundness, and diameter independently, regardless of their correlations.  **Classification Task**: Naive Bayes assigns class labels to problem instances based on feature vectors. It’s commonly used for tasks like text classification. For example, it can determine whether an email is spam or not based on the words it contains3.  **Generative Learning**: Naive Bayes is part of a family of generative learning algorithms. It models the distribution of inputs for a given class or category3.  Despite its seemingly oversimplified assumptions, Naive Bayes classifiers have proven effective in many real-world scenarios. They are highly scalable and computationally efficient, making them a popular choice for various applications |
| **2** | **DECISION TREE** | are a non-parametric supervised learning method used for [classification](https://scikit-learn.org/stable/modules/tree.html#tree-classification) and [regression](https://scikit-learn.org/stable/modules/tree.html#tree-regression). The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.  For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.  advantages of decision trees   * Simple to understand and to interpret. Trees can be visualized. * Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Some tree and algorithm combinations support [missing values](https://scikit-learn.org/stable/modules/tree.html#tree-missing-value-support). * The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree. * Able to handle both numerical and categorical data. However, the scikit-learn implementation does not support categorical variables for now. Other techniques are usually specialized in analyzing datasets that have only one type of variable. See [algorithms](https://scikit-learn.org/stable/modules/tree.html#tree-algorithms) for more information. * Able to handle multi-output problems. * Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret. * Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model. * Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.   disadvantages of decision trees   * Decision-tree learners can create over-complex trees that do not generalize the data well. This is called overfitting. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem. * Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble. * Predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations as seen in the above figure. Therefore, they are not good at extrapolation. |
| **3** | **RANDOM FOREST CLASSIFIER** | A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Trees in the forest use the best split strategy, i.e. equivalent to passing splitter="best" to the underlying Decision Tree Regressor. The sub-sample size is controlled with the max\_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree  For a comparison between tree-based ensemble models see the example Comparing Random Forests and Histogram Gradient Boosting models. |
| **4** | **K-NEAREST NEIGHBOURS OR K-NN** | This model can be used for either prediction or classification according to the variables in question. The model compares the closeness between observations that already exist in a data set and the ones that are newly formed. The machine does the math itself and selects the number of neighbours that need to be compared (k). It limits the occurrence of data underfitting and overfitting.  For classification, the closeness of most of the neighbours belonging to a specific class to the new observation determines the class of the new observation. In a prediction scenario, the value of the new observation is predicted by taking an average of the neighbours’ attributes that are being targeted. Learn more about |
| **5** | **CLASSIFICATION AND REGRESSION TREES (CART)** | Decision trees are a very effective way to divide different findings and then put into groups. CART is a preferred and useful type of a decision tree that is used both for regression and classification. A response variable is selected, and the predictor variables are divided into groups. The number of divisions required is typically chosen by the machine itself to avoid instances of overfitting and underfitting. CART is effective where other models like black-box ones are usually not found fit due to the lack of clarity or transparency they provide. |