UNIT-III Machine Learning Techniques

INSTANCE-BASED LEARNING: k-Nearest Neighbor Learning.

DECISION TREE LEARNING - Decision tree learning algorithm, Inductive bias, Inductive inference with decision trees, Entropy and information theory, Information gain, ID-3 Algorithm, Issues in Decision tree learning.

SUPPORT VECTOR MACHINE: Introduction, Types of support vector kernel – (Linear kernel, polynomial kernel, and Gaussian kernel, Hyperplane – (Decision surface), Properties of SVM, and Issues in SVM.

BAYESIAN LEARNING - Bayes theorem, Concept learning, Bayes Optimal Classifier, Naïve Bayes classifier, Bayesian belief networks, EM Algorithm.

CLUSTERING AND ITS TYPES: k-means clustering, Hierarchical Clustering, partitioning clustering, Training and Evaluation of a model, Loss functions, Evaluation, Confusion Matrix, Dataset split and Crossvalidation, Underfitting and Overfitting, Feature Engineering.

INSTANCE-BASED LEARNING:

(also known as **memory-based learning**or**lazy-learning)**

In machine learning, instance-based learning (sometimes called memory-based learning) is **a family of learning algorithms that, instead of performing explicit generalization, compare new problem instances with instances seen in training, which have been stored in memory**.

* They are sometimes referred to as lazy learning methods because they delay processing until a new instance must be classified.
* It doesn’t attempt to build the models a-priori at all, but only queries the training data on demand during scoring for each specific case.
* The nearest neighbors of an instance are defined in terms of Euclidean distance.
* No model is learned.
* The stored training instances themselves represent the knowledge
* Training instances are searched for instance that most closely resembles new instance
* Examples of instance-based learning algorithms are the **k-nearest neighbors algorithm, kernel machines and RBF networks**.

What are the advantages and disadvantages of instance-based learning?

Answer:

**Advantages of instance-based learning:**

* It has the ability to adapt to previously unseen data, which means that one can store a new instance or drop the old instance.

**Disadvantages of instance-based learning:**

* Classification costs are high.
* Large amount of memory required to store the data, and each query involves starting the identification of a local model from scratch.

Difference between model- based and Instance-based Learning

|  |  |
| --- | --- |
| **Usual/Conventional Machine Learning/Model-based Learning** | **Instance Based Learning** |
| Prepare the data for model training | Prepare the data for model training. No difference here |
| Train model from training data to estimate model parameters i.e. discover patterns | Do not train model. Pattern discovery postponed until scoring query received |
| Store the model in suitable form | There is no model to store |
| Generalize the rules in form of model, even before scoring instance is seen | No generalization before scoring. Only generalize for each scoring instance individually as and when seen |
| Predict for unseen scoring instance using model | Predict for unseen scoring instance using training data directly |
| Can throw away input/training data after model training | Input/training data must be kept since each query uses part or full set of training observations |
| Requires a known model form | May not have explicit model form |
| Storing models generally requires less storage | Storing training data generally requires more storage |
| Scoring for new instance is generally fast | Storing for new instance may be slow |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

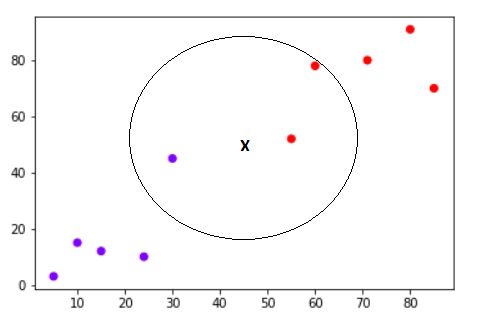
**K Nearest Neighbors (KNN) Algorithms:**

The K-nearest neighbors (KNN) algorithm is a type of supervised machine learning algorithms. KNN is extremely easy to implement in its most basic form, and yet performs quite complex classification tasks.

The intuition behind the KNN algorithm is one of the simplest of all the supervised machine learning algorithms. It simply calculates the distance of a new data point to all other training data points. The distance can be of any type e.g **Euclidean**or Manhattan etc. It then selects the K-nearest data points, where K can be any integer. Finally it assigns the data point to the class to which the majority of the K data points belong.

Example1 :

Your task is to classify a new data point with 'X' into "Blue" class or "Red" class. The coordinate values of the data point are x=45 and y=50. Suppose the value of K is 3. The KNN algorithm starts by calculating the distance of point X from all the points. It then finds the 3 nearest points with least distance to point X. This is shown in the figure below. The three nearest points have been encircled.



The final step of the KNN algorithm is to assign new point to the class to which majority of the three nearest points belong. From the figure above we can see that the two of the three nearest points belong to the class "Red" while one belongs to the class "Blue". Therefore the new data point will be classified as "Red".

Example 2:

Consider a dataset that contains two variables: height (cm) & weight (kg). Each point is classified as normal or underweight.

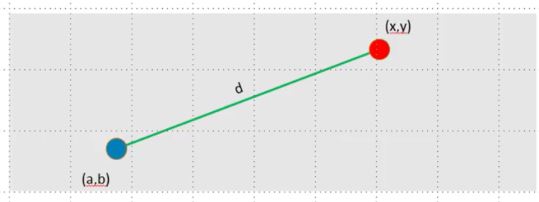


Based on the above data, you need to classify the following set as normal or underweight using the KNN algorithm.

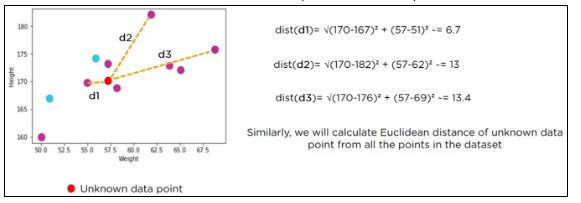
To find the nearest neighbors, we will calculate the Euclidean distance.

The Euclidean distance between two points in the plane with coordinates (x,y) and (a,b) is given by:

https://www.simplilearn.com/ice9/free_resources_article_thumb/dist.JPG



Let us calculate the Euclidean distance with the help of unknown data points.



The following table shows the calculated Euclidean distance of unknown data points from all points.



Now, we have a new data point (x1, y1), and we need to determine its class.

kg

Looking at the new data, we can consider the last three rows from the table—K=3.



Since the majority of neighbors are classified as normal as per the KNN algorithm, the data point (57, 170) should be normal.

**How do you decide the value of K (number of neighbors) in KNN?**

Now, you understand the KNN algorithm working mechanism. At this point, the question arises that How to choose the optimal number of neighbors? And what are its effects on the classifier? The number of neighbors(K) in KNN is a hyperparameter that you need choose at the time of model building. You can think of K as a controlling variable for the prediction model.

Research has shown that no optimal number of neighbors suits all kind of data sets. Each dataset has it's own requirements. In the case of a small number of neighbors, the noise will have a higher influence on the result, and a large number of neighbors make it computationally expensive. Research has also shown that a small amount of neighbors are most flexible fit which will have low bias but high variance and a large number of neighbors will have a smoother decision boundary which means lower variance but higher bias.

Generally, Data scientists choose as an odd number if the number of classes is even. You can also check by generating the model on different values of k and check their performance. You can also try Elbow method here.

**3 x 3 confusion matrix**

