**KNN (K-Nearest Neighbors)**

*Abstract* – In the field of machine learning, prediction models can be broadly classified into supervised and unsupervised machine learning algorithms [5]. K-nearest-neighbor (KNN) is one of the most fundamental and simple classification methods [6] used when there is little or no prior knowledge about the distribution of the data. In this paper we will explore the details around the algorithm, hyper parameters and explore various optimization techniques.

*Keywords – KNN, non-parametric, normalization, Euclidean distance, Hamming distance, Manhattan distance, Minkowski distance, cross-validation, curse of dimensionality.*

1. **Introduction**

In an unpublished US Air Force School of Aviation Medicine report in 1951, Fix and Hodges introduced a non-parametric method for pattern classification that has since become known the k-nearest neighbor rule (Fix & Hodges, 1951) [14].

The KNN algorithm takes the idea of similarity and assumes that things that are similar tend to exist in close proximity. KNN captures this idea of similarity (sometimes called distance, proximity, or closeness) to predict the label of an unknown data point or a prediction target.

**KNN Algorithm**:

**Step 1**: The first step is to compute distances from the test point to all training points.

**Step 2**: The data points are then sorted according to distance in ascending order.

**Step 3**: We then apply a clever trick of summing the class labels for each of the K nearest neighbors and using the sign of this sum as our prediction.

## A screenshot of a cell phone Description automatically generated

Fig 1:Source: A course in machine learning Hal Daumé III

In case of a tie, below are the methods to resolve a tie for binary/multiclass classification:

1. **Using majority class label:** In case of a tie, wherein, there are equal number of nearest neighbors in each class, use the majority label in the training data.
2. **Ranking model**: Rank the K nearest neighbors based on their distance to the test data example. Choose the class label with minimum sum of rank of nearest neighbors.
3. **Avg. Distance model**: Choose the class label having the minimum Avg distance among the K nearest neighbors.
4. **Characteristics of KNN algorithm:**

*Supervised machine learning algorithm*as target variable is known.

*Non-parametric*as it does not make an assumption about the underlying data distribution pattern.

*Lazy algorithm* as KNN does not have a training step. All data points will be used only at the time of prediction. With no training step, prediction step is costly. An eager learner algorithm eagerly learns during the training step and discard the training examples after learning.

1. **Dependencies in choosing NN (nearest neighbors):**
2. Normalization of feature values:

In most cases the data contains multiple features with difference range of values. For example: Age: range (18, 80) and Salary: range (35,000, 500,000). In such scenarios, the features with larger range will become more important and have a significant effect on the prediction.

Normalizing the features by using techniques like min-max normalization to have the same range of values will result in even distribution of importance across all features in data.

1. Sensitivity to irrelevant inputs:

Irrelevant or noisy feature will add random perturbations to the distance metrics and easily result in bad performance of the model. Generally, irrelevant features can be obtained by analyzing the feature importance of the attribute against the target class label. Noisy features can be considered as the features having anomalous data or having significant burden of missing values.

Implementing techniques for detection of the outlier and discussing the potential outlier data points with subject matter experts to eliminate any irrelevant inputs will help reduce the variance in predictions caused in irrelevant inputs.

1. Type of distance metric:

Nearest neighbors critically depend on the distance metrics.

Distance between the points can be calculated using:

**Euclidean distance** is the square root of the sum of squared distance between two points. It is also known as L2 norm.

**Manhattan distance** is the sum of the absolute values of the differences between two points

**Hamming distance** is used for categorical variables. In simple terms it tells us if the two categorical variables are same or not.

* Hamming distance b/w categorical variable – color: Red and Green = 1
* Hamming distance b/w categorical variable – color: Green and Green = 0

**Minkowski distance** is the used to find distance similarity between two points. When p=1, it becomes Manhattan distance and when p=2, it becomes Euclidean distance

To calculate distance metric for categorical attributes, need to be first transformed by creating dummies or creating numerical attributes.

Depending on the distance metric used, the K nearest neighbors may vary given the training data resulting in variation in the test predictions. One can choose the best distance metric based on the properties of the training data. If you are unsure, you can experiment with different distance metrics and choose the distance metrics that results in the most accurate models.

1. **Methods to choose K:**

The big question, of course, is how to choose K? There is no structured method to find the best value for ‘K’. Having said this, below are some of the techniques and caveats around choosing the right ‘K’.

1. Trial and error: One can try using different K values and evaluate performance to choose the best K.
2. Choosing smaller values for K can be noisy and will have a higher influence on the result. With K = 1, we run the risk of overfitting[7].
3. Larger values of K will have smoother decision boundaries which mean lower variance but increased bias. If K is large (for instance, K = N), then KNN-Predict will always predict the majority class. Clearly that is underfitting[7]. Also, computationally expensive.
4. Elbow Method [8]: The elbow method considers the accuracy of predictions and determines the value of K to attain the best predictions.

Step 1: Run the KNN algorithm for different values of k.

Step 2: For every value of k, calculate accuracy metric on the test data.

Step 3: Plot the curve of accuracy vs the value of k.

Step4: Observe the plot, the point where the graph shows a curve is considered to be optimum value of k.

1. Another way to choose K is though cross-validation [9]. One way to select the cross-validation dataset from the training dataset is to take the small portion from the training dataset and call it a validation dataset, and then use the same to evaluate different possible values of K. This way we are going to predict the label for every instance in the validation set using with K equals to 1, K equals to 2, K equals to 3 and so on. Then we look at what value of K gives us the best performance on the validation set and then we can take that value and use that as the final setting of our algorithm, so we are minimizing the validation error.
2. In general, practice, choosing the value of k is k = sqrt(N) where N stands for the number of samples in your training dataset.
3. Particularly in classification, keep the value of k odd for binary classification to avoid confusion between two classes of data. Also, it is recommended to use a value of K > **# class labels** in train data.

Clearly, K is a **hyperparameter** [10]of the KNN algorithm that allows us to trade-off between overfitting (small value of K) and underfitting (large value of K).

1. **Using KNN in a classification and regression problem:**
2. Majority Voting (classification):

First, find the k closest point to test data point and then classify the point by majority vote of its k neighbors. Amongst, the K nearest neighbors in training data, each data point votes for their class and the class with the most votes is taken as the prediction.

The problem with this approach is that the data points that are farthest amongst the K nearest neighbors will contribute as much as the data points that are closest to the test data. As an improvement, researchers use a weighted nearest neighbors approach (explained below).

1. Average voting (regression):

Similarly, in a regression setting the average of the target variable of the K nearest neighbors is taken as the predicted value for the target variable for the test data point.

1. Weighted nearest neighbors[11]:

It makes sense to weight the contribution of each example according to the distance to the test data point.

* Weight: The weight varies inversely with the distance, such that the examples closer to the query point gets higher weight. [2]

When using the technique of weighted nearest neighbors, we could allow all the training example to contribute towards the prediction [3].

1. Weighted distance metric[12]**:**

One other approach is to weight each feature based on its mutual information to the target class. Then use the weighted square distance as the distance metric. Instead of using K examples, we could allow all training examples to contribute [2].

1. **Curse of dimensionality, pitfall to KNN:**

KNN breaks down in high dimensional space as the neighborhood becomes very large.

As the dimensionality of the data [13] increases the data becomes sparse in the high dimensional space, resulting in high variance.

One of the techniques to resolve this problem is to use dimensionality reduction by shrinking the unimportant dimensions of the feature space, bringing more relevant neighbors close to the target point. Generally, it is recommended to use the number of dimensions that explain 99% of the variance in data. For further learning on choosing the right number of dimensions see [15].

1. **Pros and Cons of KNN model:**
2. Advantages:

* Learning is very simple and intuitive
* Flexible decision boundaries
* Variable-sized hypothesis space

1. Disadvantages:

* Distance function must be carefully chosen or tuned
* Irrelevant or correlated features have high impact and must be eliminated
* Typically, cannot handle high dimensionality
* Computation cost in terms of memory and classification/prediction time: Adding training samples results in increase in time taken to predict the class of the test sample as the distance of the test sample with respect to the added training sample has to be calculated.
* Sensitive to class imbalanced dataset, where the majority class tends to dominate the voting of a new instance.

1. **Application of KNN:**

Used in recommendation systems, large entertainment service providers like Netflix and online stores like Amazon use in order to recommend different movies to watch or books to buy.

Credit ratings, collecting financial characteristics vs. comparing people with similar financial features to a database

In political science, classing a potential voter to a “will vote” or “will not vote”, or to “vote Democrat” or “vote Republican”.

Handwriting detection (like OCR), image recognition and even video recognition.

**References:**

1. A course in machine learning, Hal Daumé III.
2. T.M. Mitchell. “Machine Learning”. In: McGraw-Hill International Editions. McGraw-Hill, 1997. Chap. 8. isbn: 9780071154673. Available: https://books.google.com/books?id=EoYBngEACAAJ.
3. Donald Shepard. “A two-dimensional interpolation function for irregularly-spaced data”. In: Proceedings of the 1968 23rd ACM national conference. ACM. 1968, pp. 517–524.
4. Dudani, S.A. The distance-weighted k-nearest-neighbor rule. IEEE Trans. Syst. Man Cybern., SMC-6:325–327, 1976.
5. https://dataconomy.com/2015/01/whats-the-difference-between-supervised-and-unsupervised-learning/.
6. https://dataaspirant.com/2014/09/27/classification-and-prediction/https:/dataaspirant.com/2014/09/27/classification-and-prediction/.
7. https://www.analyticsvidhya.com/blog/2020/02/underfitting-overfitting-best-fitting-machine-learning/.
8. https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/.
9. https://towardsdatascience.com/why-and-how-to-cross-validate-a-model-d6424b45261f.
10. https://quantdare.com/what-is-the-difference-between-parameters-and-hyperparameters/.
11. https://en.wikipedia.org/wiki/Inverse\_distance\_weighting.
12. Bailey, T., Jain, A. A note on distance-weighted k-nearest neighbor rules. IEEE Trans. Systems, Man, Cybernetics, Vol. 8, pp. 311-313, 1978.
13. https://www.statisticshowto.com/dimensionality/#:~:text=Dimensionality%20in%20statistics%20refers%20to,one%20column%20representing%20each%20dimension.
14. http://www.scholarpedia.org/article/K-nearest\_neighbor.
15. https://www.sciencedirect.com/science/article/abs/pii/S0167947311004099