Kriti Yadav

Summary

The provided content offers an in-depth exploration of the k-NN (k-

Frequent Interview Questions on k-NN Algorithm

Image-Pexels

Q.1 What is k-NN Algorithm?

Ans. k-NN is the simplest supervised learning algorithm. It assumes the similarity between the new data and available cases and puts new data into the category that is most similar. It stores all the available data and classifies a new data point based on similarity. k-NN can also be used for regression problems, but it is mostly used for classification problems. It is also known as a “lazy learner” because it does not learn from the training set immediately. But at the time of classification, it performs the action.

Ans. The ‘k’ in k-NN refers to the number of nearest neighbors that are considered for making a prediction for a new data point. The value of ‘k’ in k-NN is a hyperparameter that you can tune in order to optimize the performance of the algorithm. It is important to choose an appropriate value for K, as it can greatly affect the accuracy of the algorithm.

If you choose a small value of k, the algorithm will consider fewer neighbours, and the prediction will be more sensitive to outliers and noisy data. This can lead to overfitting, where the algorithm performs well on the training data but poorly on the test data.

On the other hand, if you choose a large value of k, the algorithm will consider more neighbours, and the prediction will be more stable and less sensitive to outliers and noisy data. However, this may result in underfitting, where the algorithm fails to capture the underlying patterns in the data.

Therefore, the choice of k should be made carefully, and typically involves trying different values of k and evaluating the performance of the algorithm on a validation set. Cross-validation can also be used to find an optimal value of k that balances between overfitting and underfitting.

Q.3 Why is k-NN a non-parametric algorithm?

not have a fixed number of parameters. Instead, the number of parameters grows with the size of the data.

The advantage of this approach is that the model can capture the complex relationship between features and target variables, without imposing any constraint on the form of the underlying function that describes those relationships.

Q.4 Why is the odd value of 'k' preferred over an even value in the k-NN algorithm?

Ans. Using an odd value of 'k' is preferred over an even value because it can prevent ties in the class prediction where k is even, and there is a possibility of having an equal number of neighbours for each class, resulting in a tie.

Q.5 How does the k-NN algorithm make predictions on unseen datasets?

Ans. In k-NN, to make a prediction for a new, unseen data point, the algorithm identifies the k-closest points (i.e., the k-nearest neighbour) in the training dataset based on some distance metrics, and then assigns the class label of the majority of these k neighbours to the new data point.

Manhattan distance and Minkowski distance.

Q.6 Is Feature scaling required for the k-NN?

Ans. Yes, feature scaling is generally recommended for k-NN algorithm. Since k-NN is a distance-based algorithm, the distance metric used in k-NN is typically Euclidean distance, which is sensitive to the difference in the magnitude of the different features.

If the range of the values for the different features in the dataset is not normalized, features on larger scales will dominate the distance calculation, and smaller-scaled features may not contribute as much to the final result.

Q.7 Describe the method used for feature scaling in k-NN algorithm?

Ans. In k-NN, the distances between neighbouring points are used to determine the classification of a new data point. If the features in the data set have different scales, the distance calculations can be dominated by the features with the largest scales, and features with smaller scales may be overlooked. This can lead to inaccurate predictions, especially if some features are more important than others for classification.

k-NN are:

(2) Min-Max scaling (also called normalization): This method scales the features so that they all have a range between 0 and 1. To perform Min-Max scaling, the following formula is applied to each feature:

X_scaled = (X — X_min) / (X_max — X_min)

where X is the original value of the feature, X_min is the minimum value of the feature in the data set, and X_max is the maximum value of the feature in the data set.

(2) Standardization: This method scales the features so that they have a mean of 0 and a standard deviation of 1. To perform standardization, the following formula is applied to each feature:

X_scaled = (X — X_mean) / X_std

where X is the original value of the feature, X_mean is the mean value of the feature in the data set, and X_std is the standard deviation of the feature in the data set.

Both Min-Max scaling and standardization are effective methods for feature scaling in k-NN. The choice of which method to use depends on the specific

Q.8 What is the space and time complexity of the k-NN Algorithm?

Ans. Space Complexity: Assuming a training dataset of size n and a value of k, the space complexity of the k-NN algorithm is O(n), as the algorithm needs to store the entire training dataset in memory.

Time Complexity: Time Complexity depends on the size of the training dataset and the value of k. Assuming a training dataset of the size n and a value of k, the time complexity of the k-NN algorithm is O(nd), where d=dimension of feature space. The k-NN algorithm needs to calculate the distance between the query point and every point in the training datasets, which requires O(nd). Sorting the distance to find the k-nearest takes O(nlog n) time. Hence total time complexity is O(nd+nlog n+k).

Q.9 Can k-NN algorithm be used for a regression problem?

Ans. Yes, the k-NN algorithm can be used for a regression problem. In k-NN regression, the predicted value for a new data point is calculated by averaging the values of the k-NN.

Q.10 Why is it recommended not to use the k-NN Algorithm for large datasets?

needs to store all the data and then make a decision only at runtime. The k-NN algorithm can be computationally expensive, especially for large datasets. This is because the algorithm needs to compute the distance between each pair of data points in the dataset in order to identify the k-nearest neighbours. As the number of data points increases, the number of distance computations required grows quickly, and this can make the algorithm very slow.

In addition to the computational cost, there are also other issues that can arise when using the k-NN algorithm on large datasets. One of these is the "curse of dimensionality", which refers to the fact that as the number of dimensions (or features) in the data increases, the distance metric becomes less meaningful and the algorithm becomes less accurate. This can be especially problematic for high-dimensional data.

Overall, while the k-NN algorithm can be effective for smaller datasets or datasets with a small number of dimensions, it is not recommended for use with very large datasets or high-dimensional data. In these cases, other machine learning algorithms that are better suited to these types of data, such as decision trees, random forests, or neural networks, may be more appropriate.

Q.11 How to choose the optimal value of k in the k-NN Algorithm?

Ans. Choosing the optimal value of k in the k-NN (k-Nearest Neighbors) algorithm is an important task, as it can have a significant impact on the

determine the optimal value of k.

(1) Cross-validation: Cross-validation is a common method to evaluate the performance of a model. In k-fold cross-validation, the data is split into k parts, and the model is trained on k-1 parts and evaluated on the remaining part. This process is repeated k times, with each part used as the test set once. The optimal value of k is then selected based on the average performance across all k folds.

(2) Grid Search: Grid search is a technique to search for the best hyperparameters of a model. In the case of k-NN, grid search involves testing the model's performance with different values of k and selecting the value that yields the best performance.

(3) Rule of Thumb: A common rule of thumb for choosing k is to take the square root of the total number of data points in the training set. This is a quick and easy way to get a rough estimate of the optimal k value.

Q.12 How to handle categorical variables in the k-NN Algorithm?

Ans. To handle the categorical variables we have to create dummy variables out of a categorical variable and include them instead of the original categorical variable. Unlike regression, create k dummies instead of (k-1). Before applying the k-NN algorithm, it is also common practice to encode categorical variables as numerical variables using techniques such as one-hot

each category.

Q.13 How can you relate k-NN algorithm to the bias-variance Tradeoff?

Ans. The k-nearest neighbors (k-NN) algorithm is a non-parametric machine learning algorithm that is commonly used for classification and regression tasks. The algorithm works by finding the k closest training examples to a test example and making a prediction based on the labels or values of those k examples. The choice of the value of k can impact the algorithm's performance and can be related to the bias-variance tradeoff.

When **k is small**, the algorithm is more sensitive to the noise in the data, and it tends to have **low bias and high variance**. This means that the algorithm is likely to overfit the training data and perform poorly on new data. In contrast, when **k is large**, the algorithm is less sensitive to the noise in the data, and it tends to have **high bias and low variance**. This means that the algorithm is likely to underfit the training data and perform poorly on the training data.

Q.14 How do you handle missing data in KNN algorithm?

Ans. In the k-nearest neighbors (KNN) algorithm, missing data can be handled in different ways depending on the specifics of the problem and the

handle missing data in KNN?

(1) Deletion: One simple approach is to remove any data points with missing values from the dataset before applying the KNN algorithm. This approach is straightforward, but it can lead to a loss of information if there are a large number of missing values.

(2) Imputation: Another approach is to fill in missing values with estimated values. There are different imputation techniques that can be used, such as mean imputation, median imputation, and regression imputation. Mean imputation involves replacing missing values with the mean of the available values for that feature. Median imputation involves replacing missing values with the median of the available values. Regression imputation involves using a regression model to predict missing values based on the available values.

(3) Distance weighting: In KNN, the distance between data points is used to determine their similarity. One approach to handling missing values is to modify the distance metric to give less weight to features with missing values. For example, you could use a modified Euclidean distance that adjusts the distance based on the number of missing values for each feature.

Q.15 How does the curse of dimensionality affect KNN algorithm?

feature space becomes increasingly sparse as the number of dimensions (features) increases. This can lead to a number of challenges for machine learning algorithms, including the k-nearest neighbor (KNN) algorithm. The curse of dimensionality can make this process difficult in several ways:

(1) Distance calculation: As the number of dimensions increases, the distance between any two points in the feature space also tends to increase. This can make it harder to identify truly "close" neighbors, since many points may be equidistant from the test example. As a result, the KNN algorithm may become less accurate as the number of dimensions increases.

(2) Increased computational complexity: With more dimensions, there are more combinations of features to consider, which can result in a larger search space. This can increase the computational complexity of KNN, making it slower and more memory-intensive.

(3) Overfitting: As the number of dimensions increases, the likelihood of overfitting the training data also increases. This is because, with more dimensions, there is a higher probability of finding spurious correlations between features and the target variable.

To mitigate the effects of the curse of dimensionality in KNN, one approach is to use feature selection or feature engineering techniques to reduce the number of dimensions in the feature space. Additionally, dimensionality reduction techniques like principal component analysis (PCA) can help to identify the most important features and reduce the computational

algorithms that are less susceptible to the curse of dimensionality, such as decision trees or neural networks.

Q.16 What are some of the applications of KNN algorithm?

Ans. K-Nearest Neighbors (KNN) is a popular machine learning algorithm that can be used for a variety of applications, including:

(1) Image recognition: KNN can be used to classify images by comparing their features to those of known images. For example, KNN can be used to classify images of animals based on their shape, size, color, etc.

(2) Recommendation systems: KNN can be used to create recommendation systems by finding the nearest neighbors to a given item or user, and recommending items that are similar.

(3) Text classification: KNN can be used to classify text documents, such as email spam detection, sentiment analysis, and topic classification.

Happy Learning!!

About the Author: I am Kriti Yadav, Data Scientist. My current work focuses on computer vision, deep learning, natural language processing, and machine learning. Please reach out to me via my **Linkedin** profile if you have any questions.

Recommended from ReadMedium

Mahendramedapati

Understanding Decision Tree Algorithm

An In-depth Exploration of Decision Trees: Theory, Implementation, and Practical Applications and using sklearn Library

17 min read

Be 10x Engineer

Staff Backend Engineer @ Google | Interview Experience

It was a crisp Tuesday morning when I first saw the email. My hands trembled slightly as I read the subject line: “Google Interview...

3 min read

Samuel Ozechi

Stochastic Gradient Descent For Deep Learning

Understanding the parameter optimization process for deep learning models.

Vyacheslav Efimov

Understanding Deep Learning Optimizers: Momentum, AdaGrad, RMSProp & Adam

Gain intuition behind acceleration training techniques in neural networks

8 min read

Thomas A Dorfer

Bagging vs. Boosting: The Power of Ensemble Methods in Machine Learning

How to maximize predictive performance by creating a strong learner from multiple weak ones

6 min read

Mikel

Pinterest ML Internship Summer 2025

Looking for a tech internship for the Summer 2025. I share my recent Interview experience, Questions, Solutions and tips.

6 min read