Viva Questions: K-Nearest Neighbors Classification

General Concepts

1. **What is the K-Nearest Neighbors (KNN) algorithm?**
   - **Answer:** KNN is a supervised machine learning algorithm used for classification and regression. It classifies new data points based on the majority class among its K nearest neighbors in the feature space.
2. **How does the KNN algorithm work?**
   - **Answer:** KNN calculates the distance between the new data point and all existing points in the dataset. It then selects the K nearest points and assigns the most frequent class (for classification) or the average (for regression) as the prediction.
3. **What distance metrics can be used in KNN?**
   - **Answer:** Common distance metrics include Euclidean distance, Manhattan distance, and Minkowski distance.
4. **What is the significance of the parameter 'K' in KNN?**
   - **Answer:** The parameter 'K' specifies the number of nearest neighbors to consider for making a prediction. A small K can lead to overfitting, while a large K can lead to underfitting.
5. **What is the impact of choosing a different value for `n_neighbors` on the KNN model?**
   - **Answer:** A smaller value of `n_neighbors` can make the model sensitive to noise and overfit the training data, while a larger value smooths out the predictions but may lead to underfitting. Choosing the optimal `n_neighbors` involves balancing bias and variance.
6. **How can you tune the hyperparameter `n_neighbors` to improve model performance?**
   - **Answer:** Hyperparameter tuning can be done using techniques like cross-validation to evaluate the model's performance for different values of `n_neighbors` and selecting the value that gives the best performance on validation data.
7. **What is cross-validation, and why is it important in evaluating KNN models?**
   - **Answer:** Cross-validation is a technique where the dataset is split into multiple folds to train and evaluate the model on different subsets. It helps in assessing the model's generalization performance and reduces the risk of overfitting.

Dataset-Specific Questions

1. **Why did you choose the Iris dataset for classification?**
   - **Answer:** The Iris dataset is a well-known dataset in machine learning that is commonly used for testing classification algorithms. It contains features of different iris species, making it suitable for demonstrating

KNN.

2. **What are the features of the Digits dataset used in the classification task?**
   - **Answer:** The Digits dataset consists of 8x8 pixel images of handwritten digits (0-9). Each image is flattened into a 64-dimensional vector representing pixel intensity values.
3. **What preprocessing steps are necessary for the Digits dataset before applying KNN?**
   - **Answer:** Preprocessing steps may include normalizing pixel values, handling missing values if any, and ensuring that the dataset is appropriately split into training and testing sets.
4. **How does the choice of distance metric affect the results in KNN classification?**
   - **Answer:** Different distance metrics (e.g., Euclidean, Manhattan) can affect the performance of KNN by changing how distances between points are calculated. The choice of metric may impact which neighbors are considered and thus the classification results.

Implementation and Evaluation

1. **How do you split the dataset for training and testing?**
   - **Answer:** The dataset is split into training and testing sets using `train_test_split` from `sklearn.model_selection`. Typically, a portion (e.g., 20%) of the data is used for testing, and the rest for training.
2. **What metrics are used to evaluate the performance of the KNN classifier?**
   - **Answer:** Common metrics include accuracy, confusion matrix, precision, recall, and F1-score. These metrics help assess the classifier's performance in terms of correctly predicted labels and classification errors.
3. **How is the confusion matrix interpreted?**
   - **Answer:** The confusion matrix shows the number of correct and incorrect predictions categorized by the true and predicted labels. It helps in understanding the performance of the classification model and in identifying where it might be making errors.
4. **What does the classification report provide?**
   - **Answer:** The classification report provides precision, recall, F1-score, and support for each class in the classification task. It gives a detailed summary of the classifier's performance.
5. **What are the advantages of using KNN for classification?**
   - **Answer:** Advantages include simplicity, no assumption about data distribution, and effectiveness for small datasets. KNN can also adapt easily to new data without retraining the model.
6. **What are the disadvantages or limitations of KNN?**
   - **Answer:** Disadvantages include high computational cost during pre-

diction, especially for large datasets, and sensitivity to irrelevant or redundant features. KNN can also suffer from the curse of dimensionality where performance degrades as the number of features increases.

7. **How can feature scaling affect the performance of KNN?**
   - **Answer:** KNN is sensitive to feature scaling because distance calculations are affected by the magnitude of features. Scaling features to the same range (e.g., using standardization or normalization) helps in improving the performance of KNN.

Code-Specific Questions

1. **What does the `KNeighborsClassifier` class do?**
   - **Answer:** `KNeighborsClassifier` is a class in `sklearn` that implements the K-Nearest Neighbors algorithm for classification tasks. It allows training with labeled data and predicting the class of new samples.
2. **Explain the purpose of the `fit` method in the KNN classifier.**
   - **Answer:** The `fit` method trains the KNN model using the provided training data and labels. It stores the training data to be used for making predictions.
3. **What is the purpose of `predict` method in the KNN classifier?**
   - **Answer:** The `predict` method is used to classify new data points based on the nearest neighbors from the training data.
4. **Why is `random_state` used in `train_test_split`?**
   - **Answer:** `random_state` ensures that the splitting of the dataset is reproducible. It allows for consistent results each time the code is run by controlling the randomness of the split.
5. **Why is the `accuracy_score` used as a metric in evaluating the KNN classifier?**
   - **Answer:** `accuracy_score` measures the proportion of correctly classified instances among the total instances. It provides a straightforward metric of overall performance.
6. **What does the `classification_report` provide, and how can it be useful?**
   - **Answer:** The `classification_report` provides detailed metrics including precision, recall, F1-score, and support for each class. It is useful for understanding how well the model performs on each class, especially in multi-class classification problems.
7. **What is the role of `plt.xlabel`, `plt.ylabel`, and `plt.title` in plotting the confusion matrix?**
   - **Answer:** These functions set the labels for the x-axis, y-axis, and the title of the plot, respectively. They help in making the plot more informative and easier to interpret.

Visualization

1. **How is the confusion matrix visualized using Seaborn?**
   - **Answer:** The confusion matrix is visualized using `seaborn.heatmap` which creates a heatmap where the color intensity represents the number of predictions for each class.
2. **What information is displayed in the confusion matrix plot?**
   - **Answer:** The plot shows the counts of true positives, true negatives, false positives, and false negatives, providing a visual representation of the classification performance.
3. **How can you interpret different colors in the heatmap of a confusion matrix?**
   - **Answer:** In a heatmap, different colors represent the count of predictions for each class. Darker or more intense colors usually indicate higher counts, making it easier to visually compare performance across different classes.