PW SKILLS
KNN ASSISGNMENT 1

1. KNN Algorithm: K-Nearest Neighbors (KNN) is a non-parametric algorithm used for both classification and regression tasks. It works on the principle of finding the K nearest data points in the feature space to a given query point and making predictions based on the labels or values of those neighbors. KNN doesn't involve any explicit training phase; instead, it memorizes the entire training dataset, making it a "lazy learner." When a prediction is needed, KNN calculates the distances between the query point and all training points, selects the K closest ones, and then predicts the output based on the majority class (for classification) or the average (for regression) of those neighbors.

2. Choosing the Value of K: Selecting the appropriate value of K is crucial in KNN, as it directly affects the model's performance. A small value of K can lead to overfitting, where the model becomes sensitive to noise in the data, resulting in a jagged decision boundary. Conversely, a large value of K can lead to underfitting, where the model oversmooths the decision boundary, potentially misclassifying data points from different classes. Techniques like cross-validation, grid search, or domain expertise are often employed to find the optimal value of K that balances bias and variance in the model.

3. Difference between KNN Classifier and Regressor: In KNN classification, the algorithm predicts the class membership of a query point by a majority vote among its K nearest neighbors. The class with the most representatives among the neighbors is assigned to the query point. In contrast, KNN regression predicts the continuous output value for a query point by averaging the values of its K nearest neighbors. The predicted value is the mean of the target values of those neighbors. While both methods rely on the concept of proximity, the output type (categorical or continuous) determines whether KNN is used for classification or regression.

4. Performance Measurement of KNN: Evaluating the performance of a KNN model requires appropriate metrics depending on the task. For classification, metrics such as accuracy, precision, recall, F1-score, ROC curve, and confusion matrix are commonly used to assess the model's

ability to correctly classify instances. For regression, metrics like Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared are used to measure the deviation between predicted and actual values. It's essential to choose evaluation metrics that align with the specific objectives of the task and consider the inherent characteristics of the data.

5. Curse of Dimensionality in KNN: The curse of dimensionality refers to the challenges and limitations that arise when dealing with high-dimensional data in machine learning algorithms like KNN. As the number of dimensions (features) increases, the volume of the feature space grows exponentially, causing the data points to become increasingly sparse. In high-dimensional spaces, the concept of proximity becomes less meaningful, as all data points appear to be roughly equidistant from each other. This makes it challenging for KNN to discern meaningful patterns and relationships in the data, leading to decreased predictive performance and increased computational complexity.

6. Handling Missing Values in KNN: Dealing with missing values is a critical preprocessing step in KNN, as the algorithm relies on the proximity between data points. One common approach is to impute missing values before applying KNN. This can be done using techniques such as mean, median, or mode imputation, where missing values are replaced by the mean, median, or mode of the respective feature. Another approach is to use more sophisticated imputation methods like KNN imputation, where missing values are estimated based on the values of the nearest neighbors. However, care must be taken to ensure that missing values are imputed appropriately to avoid biasing the results.

7. Comparison of KNN Classifier and Regressor: KNN classifier and regressor differ in their output types and the nature of the prediction task. KNN classifier is suitable for problems where the output variable is categorical, such as image classification, sentiment analysis, or fraud detection. It works by assigning the most frequently occurring class among the K nearest neighbors to the query point. On the other hand, KNN regressor is used for problems with continuous output variables, such as predicting house prices, stock prices, or temperature. It predicts the output value by averaging the values of the K nearest neighbors. The

choice between classifier and regressor depends on the nature of the problem and the type of output variable.

8. Strengths and Weaknesses of KNN: KNN has several strengths, including simplicity, versatility, and effectiveness in capturing complex decision boundaries. It doesn't make any assumptions about the underlying data distribution and can handle non-linear relationships between features and the target variable. However, KNN also has weaknesses, such as computational inefficiency with large datasets, sensitivity to irrelevant features, and the need for appropriate distance metrics selection. These weaknesses can be addressed by using dimensionality reduction techniques, feature selection methods, or optimizing the distance metric based on the characteristics of the data.

9. Difference between Euclidean and Manhattan Distance: Euclidean distance and Manhattan distance are two common distance metrics used in KNN to measure the proximity between data points. Euclidean distance is the straight-line distance between two points in Euclidean space, calculated as the square root of the sum of the squared differences along each dimension. It considers the magnitude of the differences between corresponding coordinates. In contrast, Manhattan distance (also known as city block distance or L1 norm) calculates the distance by summing the absolute differences along each dimension. It measures the distance traveled along the axes, resembling the distance a taxi would travel in a city grid. While Euclidean distance is sensitive to scale differences between features, Manhattan distance is more robust to such variations. The choice between Euclidean and Manhattan distance depends on the characteristics of the data and the specific requirements of the problem.

10. Role of Feature Scaling in KNN: Feature scaling is essential in KNN to ensure that all features contribute equally to the distance calculation. Since KNN relies on the concept of proximity, features with larger scales may dominate the distance calculation, leading to biased results. Feature scaling transforms the feature values to a similar scale, preventing any single feature from having a disproportionate influence on the distance metric. Common scaling techniques include min-max scaling (scaling features to a range between 0 and 1) and standardization (scaling features to have zero mean and unit variance). By scaling the

features appropriately, KNN can make more accurate predictions and avoid the distortion caused by differences in feature magnitudes.