**Ques 1 Ensemble Techniques in Machine Learning**

Ensemble techniques are a powerful approach in machine learning that combine multiple models to improve predictive performance. By leveraging the strengths and weaknesses of various models, ensembles can often achieve higher accuracy, reduce overfitting, and enhance generalization.

**Ques 2 Bagging and How it Works**

Bagging (Bootstrap Aggregating) is a popular ensemble technique that involves training multiple models on different bootstrap samples of the training data. Bootstrap sampling creates new datasets by randomly sampling with replacement from the original dataset. Each model is then trained on its respective bootstrap sample.

The final prediction is made by aggregating the predictions of the individual models. This can be done through voting (for classification) or averaging (for regression). Bagging helps reduce variance and overfitting by creating diverse models.

**Ques 3 Purpose of Boosting in Bagging**

Boosting, another ensemble technique, is often used in conjunction with bagging to further improve performance. While bagging focuses on reducing variance, boosting aims to reduce bias by sequentially training models that focus on the errors made by previous models.

**Ques 4 The Random Forest Algorithm**

The Random Forest algorithm is a specific implementation of bagging that uses decision trees as the base models. It introduces additional randomness by randomly selecting a subset of features at each node during tree construction. This further diversifies the models and helps prevent overfitting.

**Ques 5 How Randomization Reduces Overfitting**

Randomization in Random Forests helps reduce overfitting by introducing diversity among the decision trees. By selecting different subsets of features at each node, the trees are less likely to become overly dependent on a small set of features. This helps prevent the models from fitting the training data too closely and improves their ability to generalize to new data.

**Ques 6 Feature Bagging in Random Forests**

Feature bagging is a technique used in Random Forests where only a random subset of features is considered at each node during tree construction. This further increases diversity among the trees and helps prevent overfitting.

**Ques 7 Role of Decision Trees in Gradient Boosting**

Gradient Boosting is another ensemble technique that sequentially trains models. Decision trees are often used as the base models in Gradient Boosting. Each new tree is trained to focus on the errors made by the previous trees, improving the overall performance.

**Ques 8 Difference Between Bagging and Boosting**

* **Bagging:** Creates parallel models using bootstrap samples, focusing on reducing variance.
* **Boosting:** Creates sequential models that focus on correcting the errors of previous models, focusing on reducing bias.

**Ques 9 The AdaBoost Algorithm**

AdaBoost (Adaptive Boosting) is a popular boosting algorithm. It assigns weights to training examples, giving more weight to misclassified examples. The weights are adjusted after each iteration to focus the next model on the difficult instances.

**Ques 10 Weighted Learning in Boosting Algorithms**

Weighted learning is a key concept in boosting algorithms. By assigning weights to training examples, the algorithm can focus on the instances that are more difficult to classify or predict. This helps improve the overall performance of the ensemble.

**Ques 11 Adaptive Boosting Process**

Adaptive Boosting involves the following steps:

1. Initialize weights for all training examples.
2. Train a base model on the weighted dataset.
3. Calculate the error rate of the model.
4. Update the weights of training examples based on their classification accuracy.
5. Repeat steps 2-4 for a specified number of iterations or until a stopping criterion is met.
6. Combine the predictions of all models using weighted voting.

**Ques 11 Adjusting Weights in AdaBoost**

AdaBoost adjusts the weights of misclassified examples to give them more importance in the next iteration. This helps the next model focus on the difficult instances and improve its performance.

**Ques 13 XGBoost Algorithm**

XGBoost (Extreme Gradient Boosting) is a powerful boosting algorithm that has gained popularity due to its efficiency and performance. It introduces several enhancements over traditional gradient boosting, such as regularization, column subsampling, and parallel processing.

**Ques 14 Regularization in XGBoost**

XGBoost incorporates regularization techniques to prevent overfitting. L1 and L2 regularization can be used to penalize large weights, which helps the model generalize better to new data.

**Ques 15 Types of Ensemble Techniques**

In addition to bagging and boosting, other ensemble techniques include:

* **Stacking:** Combining the predictions of multiple models using a meta-model.
* **Blending:** Similar to stacking, but using a fixed combination of base models.
* **Random Subspace:** Creating diverse models by using different subsets of features.

**Ques 16 Comparison of Bagging and Boosting**

**Ques 17 Ensemble Diversity**

Ensemble diversity refers to the extent to which the individual models in an ensemble are different from each other. Diverse models tend to make different errors, which can improve the overall performance of the ensemble.

**Ques 18 Improving Predictive Performance with Ensembles**

Ensembles can improve predictive performance by:

* **Reducing overfitting:** By combining multiple models, ensembles can help prevent overfitting by averaging out the errors of individual models.
* **Improving accuracy:** Ensembles can often achieve higher accuracy than individual models, especially when the individual models are diverse and have complementary strengths.
* **Enhancing robustness:** Ensembles are less sensitive to noise and outliers in the data.

**Ques 19 Ensemble Variance and Bias**

Ensemble variance refers to the variability of the predictions made by the ensemble. High variance indicates that the ensemble is sensitive to small changes in the training data. Ensemble bias refers to the systematic error introduced by the ensemble.

**Ques 20 Trade-off Between Bias and Variance**

There is a trade-off between bias and variance in ensemble learning. Increasing diversity among the models can reduce bias but may increase variance. Finding the right balance between bias and variance is crucial for optimal performance.

**Ques 21 Applications of Ensemble Techniques**

Ensemble techniques are widely used in various domains, including:

* **Financial forecasting:** Predicting stock prices, credit risk, and fraud detection.
* **Medical diagnosis:** Predicting diseases and patient outcomes.
* **Natural language processing:** Sentiment analysis, text classification, and machine translation.
* **Image recognition:** Object detection, image classification, and image generation.

**Ques 22 Model Interpretability in Ensemble Learning**

Ensemble models can be challenging to interpret due to the combination of multiple models. However, techniques like feature importance analysis can help understand the relative importance of different features in the ensemble's predictions.

**Ques 23 Stacking in Ensemble Learning**

Stacking involves training a meta-model to combine the predictions of multiple base models. The meta-model learns to weigh the predictions of the base models to optimize overall performance.

**Ques 24 Role of Meta-Learners in Stacking**

Meta-learners are the models used in stacking to combine the predictions of the base models. They can be any type of machine learning model, such as linear regression, logistic regression, or decision trees.

**Ques 24 Challenges Associated with Ensemble Techniques**

Some challenges associated with ensemble techniques include:

* **Computational cost:** Training and combining multiple models can be computationally expensive, especially for large datasets.
* **Interpretability:** Ensemble models can be difficult to interpret due to the combination of multiple models.
* **Hyperparameter tuning:** Finding the optimal hyperparameters for each base model and the ensemble itself can be challenging.

**Ques 25 Boosting vs. Bagging**

* **Boosting:** Focuses on reducing bias by sequentially training models on the errors of previous models.
* **Bagging:** Focuses on reducing variance by training models on bootstrap samples of the data.

**Ques 26 Intuition Behind Boosting**

The intuition behind boosting is that by focusing on the errors made by previous models, the ensemble can learn from its mistakes and improve its performance over time.

**Ques 27 Sequential Training in Boosting**

Boosting involves training the models sequentially, with each model focusing on the errors made by the previous models. This allows the ensemble to learn from its mistakes and gradually improve its performance.

**Ques 28 Handling Misclassified Data Points in Boosting**

Boosting algorithms assign weights to training examples, giving more weight to misclassified examples. This helps the next model focus on the difficult instances and improve its performance.

**Ques 29 Role of Weights in Boosting Algorithms**

Weights in boosting algorithms are used to give more importance to misclassified examples. This helps the next model focus on the difficult instances and improve its performance.

**Ques 30 Difference Between Boosting and AdaBoost**

AdaBoost is a specific type of boosting algorithm. It uses exponential loss and adaptively updates the weights of training examples.

**Adjusting Weights for Misclassified Samples in AdaBoost**

AdaBoost adjusts the weights of misclassified samples exponentially, giving them more importance in the next iteration. This helps the next model focus on the difficult instances and improve its performance.

**Ques 31 XGBoost Algorithm and its Advantages**

XGBoost is a powerful boosting algorithm that offers several advantages over traditional gradient boosting, including:

* **Regularization:** XGBoost incorporates regularization techniques to prevent overfitting.
* **Column subsampling:** XGBoost randomly selects a subset of features at each split, which can help prevent overfitting and improve generalization.
* **Parallel processing:** XGBoost can be parallelized, making it efficient for large datasets.

**Ques 32 Regularization in XGBoost**

XGBoost uses regularization techniques to prevent overfitting. L1 and L2

regularization can be used to penalize large weights, which helps the

model generalize better to new data.

**Types of Ensemble Techniques**

In addition to bagging and boosting, other ensemble techniques include:

* **Stacking:** Combining the predictions of multiple models using a meta-model.
* **Blending:** Similar to stacking, but using a fixed combination of base models.
* **Random Subspace:** Creating diverse models by using different subsets of features.

**Ques 33 Weak Learners in Boosting**

Weak learners are simple models that perform only slightly better than random guessing. They are typically used as the base models in boosting algorithms. The idea is that by combining many weak learners, we can create a strong ensemble model.

**Ques 34 The Process of Gradient Boosting**

Gradient boosting is a type of boosting algorithm that works by iteratively training weak learners to correct the errors of the previous models. Here's a basic outline of the process:

1. **Initialize:** Start with a base model (often a decision tree) and initialize its predictions.
2. **Calculate Residuals:** Calculate the residuals, which are the differences between the true target values and the predicted values.
3. **Train a New Model:** Train a new weak learner to predict the residuals.
4. **Update Predictions:** Update the overall predictions by adding the predictions of the new model to the existing predictions.
5. **Repeat:** Repeat steps 2-4 for a specified number of iterations.

**Ques 35 Purpose of Gradient Descent in Gradient Boosting**

Gradient descent is an optimization algorithm used in gradient boosting to find the optimal weights for the weak learners. It helps minimize the loss function, which measures the error between the predicted and true values.

**Ques 36 Role of Learning Rate in Gradient Boosting**

The learning rate controls the step size in gradient descent. A smaller learning rate can lead to slower convergence but can also help prevent overfitting. A larger learning rate can accelerate convergence but may increase the risk of overfitting.

**Ques 37 How Gradient Boosting Handles Overfitting**

Gradient boosting can help prevent overfitting by:

* **Using weak learners:** Weak learners are less likely to overfit compared to complex models.
* **Adaptive boosting:** The algorithm focuses on the errors made by previous models, reducing bias and preventing overfitting.
* **Regularization:** Techniques like L1 or L2 regularization can be used to prevent overfitting by penalizing large weights.

**Ques 38 Differences Between Gradient Boosting and XGBoost**

XGBoost is an advanced version of gradient boosting that incorporates several improvements:

* **Regularization:** XGBoost uses regularization techniques to prevent overfitting.
* **Column subsampling:** XGBoost randomly selects a subset of features at each split, which can help prevent overfitting and improve generalization.
* **Parallel processing:** XGBoost can be parallelized, making it efficient for large datasets.
* **Second-order derivatives:** XGBoost uses second-order derivatives to approximate the loss function, which can lead to faster convergence.

**Ques 39 Regularized Boosting**

Regularized boosting is a technique that incorporates regularization into boosting algorithms. This helps prevent overfitting by penalizing complex models. L1 and L2 regularization are commonly used in boosting.

**Ques 40 Advantages of Using XGBoost Over Traditional Gradient Boosting**

XGBoost offers several advantages over traditional gradient boosting, including:

* **Improved performance:** XGBoost often achieves better performance due to its regularization techniques, column subsampling, and efficient implementation.
* **Faster training:** XGBoost can be significantly faster than traditional gradient boosting, especially for large datasets.
* **Regularization:** XGBoost's built-in regularization helps prevent overfitting.

**Ques 41 Early Stopping in Boosting Algorithms**

Early stopping is a technique used in boosting algorithms to prevent overfitting. It involves monitoring the performance of the ensemble on a validation set and stopping the training process when the performance starts to degrade.

**Ques 42 How Early Stopping Prevents Overfitting in Boosting**

Early stopping helps prevent overfitting by stopping the training process before the ensemble starts to fit the training data too closely. This can improve the model's ability to generalize to new data.

**Ques 43 Role of Hyperparameters in Boosting Algorithms**

Hyperparameters in boosting algorithms control the behavior of the ensemble. Some important hyperparameters include:

* **Number of iterations:** The number of weak learners to train.
* **Learning rate:** Controls the step size in gradient descent.
* **Regularization parameters:** L1 and L2 regularization parameters.
* **Maximum depth:** The maximum depth of the decision trees used as base models.

**Ques 44 Common Challenges Associated with Boosting**

Some common challenges associated with boosting include:

* **Computational complexity:** Boosting can be computationally expensive for large datasets, especially with a large number of iterations.
* **Hyperparameter tuning:** Finding the optimal hyperparameters for boosting algorithms can be challenging.
* **Interpretability:** Boosting ensembles can be difficult to interpret due to the combination of multiple weak learners.

**Ques 45 Boosting Convergence**

Boosting algorithms typically converge to a solution, meaning that the error on the training set will decrease with each iteration. However, the rate of convergence can vary depending on the specific boosting algorithm and the dataset.

**Ques 46 Improving the Performance of Weak Learners in Boosting**

To improve the performance of weak learners in boosting:

* **Choose appropriate base models:** Select base models that are suitable for the problem at hand.
* **Tune hyperparameters:** Experiment with different hyperparameters for the base models.
* **Increase diversity:** Use techniques like feature bagging or random subspace to introduce diversity among the weak learners.

**Ques 47 Impact of Data Imbalance on Boosting Algorithms**

Data imbalance can affect the performance of boosting algorithms. If one class is significantly overrepresented, the model may become biased towards that class. Techniques like class weighting or oversampling can help address data imbalance.

**Ques 48 Real-World Applications of Boosting**

Boosting algorithms are widely used in various domains, including:

* **Financial forecasting:** Predicting stock prices, credit risk, and fraud detection.
* **Medical diagnosis:** Predicting diseases and patient outcomes.
* **Natural language processing:** Sentiment analysis, text classification, and machine translation.
* **Image recognition:** Object detection, image classification, and image generation.

**Ques 49 Ensemble Selection in Boosting**

Ensemble selection is the process of selecting a subset of models from the ensemble to improve performance. This can be done based on various criteria, such as performance on a validation set or feature importance.

**Ques 50 Contribution of Boosting to Model Interpretability**

While boosting ensembles can be challenging to interpret due to the combination of multiple weak learners, techniques like feature importance analysis can help understand the relative importance of different features in the ensemble's predictions.

**Ques 51 Curse of Dimensionality and its Impact on KNN**

The curse of dimensionality refers to the challenges that arise when working with high-dimensional data. In KNN, as the number of features increases, the data becomes sparser, making it difficult to find meaningful neighbors. This can lead to decreased performance and increased computational cost.

**Ques 52 Applications of KNN in Real-World Scenarios**

KNN is widely used in various applications, including:

* **Recommendation systems:** Recommending products or services based on user preferences and item similarities.
* **Anomaly detection:** Identifying unusual data points that deviate from the norm.
* **Image classification:** Classifying images based on their visual features.
* **Pattern recognition:** Recognizing patterns in data, such as handwritten digits or speech recognition.

**Ques 53 Weighted KNN**

Weighted KNN assigns different weights to the neighbors based on their distance from the query point. This allows closer neighbors to have a greater influence on the prediction.

**Ques 54 Handling Missing Values in KNN**

There are several ways to handle missing values in KNN:

* **Imputation:** Replace missing values with estimated values (e.g., mean, median, mode).
* **Ignore instances with missing values:** Remove instances that contain missing values.
* **Treat missing values as a separate category:** Consider missing values as a separate category for categorical features.

**Ques 55 Difference Between Lazy Learning and Eager Learning**

Lazy learning algorithms, like KNN, defer the learning process until a new data point is presented. Eager learning algorithms, like decision trees and neural networks, learn a model from the entire training set before making predictions.

**Ques 56 Improving the Performance of KNN**

To improve the performance of KNN:

* **Choose the appropriate distance metric:** The choice of distance metric can significantly impact the performance of KNN.
* **Optimize the value of K:** Experiment with different values of K to find the optimal value.
* **Handle missing values effectively:** Use appropriate techniques to handle missing values.
* **Consider feature scaling:** Scaling features to a common range can improve the performance of KNN, especially in high-dimensional spaces.

**Ques 57 Can KNN be Used for Regression Tasks?**

Yes, KNN can be used for regression tasks. In this case, the predicted value for a new data point is the average of the target values of its K nearest neighbors.

**Ques 58 Decision Boundary of the KNN Algorithm**

The decision boundary of KNN is non-linear and depends on the distribution of the data. It is typically a complex shape that separates the different classes in the feature space.

**Ques 59 Choosing the Optimal Value of K in KNN**

The optimal value of K in KNN depends on the dataset and the specific problem. A small value of K can lead to overfitting, while a large value of K can lead to underfitting. Cross-validation can be used to find the optimal value of K.

**Ques 60 Trade-offs Between Using a Small or Large Value of K in KNN**

* **Small K:** More localized decisions, can be sensitive to noise and outliers.
* **Large K:** More robust to noise and outliers, but can be less sensitive to local patterns.

**Ques 61 Feature Scaling in the Context of KNN**

Feature scaling is important in KNN because it ensures that all features contribute equally to the distance calculations. If features have different scales, features with larger magnitudes can dominate the distance calculations, leading to biased results.

**Ques 62 Comparing KNN with Other Classification Algorithms**

KNN is a simple and intuitive algorithm, but it can be computationally expensive for large datasets. Other classification algorithms like SVM and decision trees may be more.

**Ques 63 Impact of Distance Metric on KNN Performance**

The choice of distance metric significantly affects the performance of KNN. Different distance metrics measure similarity in different ways, leading to different results.

**Ques 63Common distance metrics:**

* **Euclidean distance:** Measures the straight-line distance between two points.
* **Manhattan distance:** Measures the distance along city blocks.
* **Minkowski distance:** A generalization of Euclidean and Manhattan distances.
* **Hamming distance:** Measures the number of positions where two binary vectors differ.
* **Cosine similarity:** Measures the angle between two vectors.

**Choosing the right distance metric:**

The choice of distance metric depends on the nature of the data and the problem at hand. For example:

* **Numerical data:** Euclidean, Manhattan, or Minkowski distances are often used.
* **Binary data:** Hamming distance is appropriate.
* **Text data:** Cosine similarity can be used to measure the similarity between documents.

**Ques 64 Handling Imbalanced Datasets in KNN**

Imbalanced datasets, where one class has significantly more examples than the other, can bias the KNN algorithm towards the majority class. To address this:

* **Oversampling:** Increase the number of examples from the minority class.
* **Undersampling:** Decrease the number of examples from the majority class.
* **Class weighting:** Assign higher weights to examples from the minority class during training.
* **SMOTE (Synthetic Minority Over-sampling Technique):** Generate synthetic examples for the minority class.

**Ques 65 Cross-Validation for Tuning KNN Parameters**

Cross-validation is a technique used to evaluate the performance of a machine learning model on unseen data. It involves splitting the data into multiple folds, training the model on a subset of the data, and evaluating it on the remaining fold. This process is repeated for all folds, and the average performance is reported.

Cross-validation can be used to tune the hyperparameters of KNN, such as the value of K and the choice of distance metric. By trying different combinations of hyperparameters and evaluating their performance using cross-validation, you can select the best set of parameters for your specific problem.

**Ques 66 Uniform vs. Distance-Weighted Voting in KNN**

* **Uniform voting:** All neighbors contribute equally to the prediction.
* **Distance-weighted voting:** Neighbors closer to the query point are given more weight in the prediction.

Distance-weighted voting can be beneficial when the data is not uniformly distributed or when there is a significant difference in the distances between neighbors.

**Ques 67 Computational Complexity of KNN**

The computational complexity of KNN is O(n*d*k), where n is the number of data points, d is the dimensionality of the data, and k is the number of neighbors. This makes KNN computationally expensive for large datasets and high-dimensional data.

**Ques 68 Impact of Distance Metric on Sensitivity to Outliers**

The choice of distance metric can impact the sensitivity of KNN to outliers. For example, Euclidean distance can be sensitive to outliers, as a single outlier can significantly affect the distance between two points. Manhattan distance and Minkowski distance with a higher p value can be less sensitive to outliers.

**Ques 69 Selecting the Optimal Value of K Using the Elbow Method**

The elbow method is a heuristic technique for selecting the optimal value of K in KNN. It involves plotting the average distance to the K nearest neighbors against different values of K. The optimal value of K is typically the point where the plot starts to flatten out, forming an elbow shape.

**Ques 70 Principal Component Analysis (PCA) and Its Applications**

PCA is a dimensionality reduction technique that transforms the data into a new coordinate system where the axes represent the principal components, which are linear combinations of the original features. The principal components capture the most variance in the data.

**Ques 71 Applications of PCA:**

* **Visualization:** PCA can be used to visualize high-dimensional data in a lower-dimensional space.
* **Feature extraction:** PCA can be used to extract the most important features from a dataset.
* **Noise reduction:** PCA can be used to remove noise from the data.

**Ques 72 Reconstruction Error in PCA**

The reconstruction error measures the loss of information when projecting the data onto a lower-dimensional subspace using PCA. A lower reconstruction error indicates that more variance is captured by the principal components.

**Ques 73 Applications of PCA**

PCA has many applications, including:

* **Image processing:** Compressing images and reducing noise.
* **Natural language processing:** Topic modeling and document clustering.
* **Finance:** Analyzing stock market data and identifying patterns.
* **Machine learning:** Feature engineering and dimensionality reduction.

**Ques 75 Limitations of PCA**

* **Linearity assumption:** PCA assumes that the data is linearly related.
* **Loss of interpretability:** The principal components may not have a clear interpretation.
* **Sensitivity to outliers:** PCA can be sensitive to outliers in the data.

**Ques 79 t-Distributed Stochastic Neighbor Embedding (t-SNE)**

t-SNE is another dimensionality reduction technique that is particularly effective for visualizing high-dimensional data. It preserves local structure and non-linear relationships between data points.

**Ques 80 Advantages of t-SNE over PCA:**

* **Non-linearity:** t-SNE can capture non-linear relationships between data points.
* **Preservation of local structure:** t-SNE preserves the local structure of the data.
* **Visualization:** t-SNE is effective for visualizing high-dimensional data.

**Ques 81 Limitations of t-SNE**

* **Computational complexity:** t-SNE can be computationally expensive for large datasets.
* **Randomness:** The results of t-SNE can vary due to the random initialization of the algorithm.
* **Difficulty in interpreting the embedding space:** The embedding space produced by t-SNE may not have a clear interpretation.

**Ques 82 Manifold Learning and its Significance in Dimensionality Reduction**

Manifold learning is a set of techniques that aim to learn the underlying low-dimensional structure of high-dimensional data. It assumes that the data points lie on a low-dimensional manifold embedded in a high-dimensional space.

Manifold learning techniques can be used for dimensionality reduction and visualization. They can capture complex non-linear relationships between data points that cannot be captured by linear techniques like PCA.

**Ques 83 Autoencoders**

Autoencoders are neural networks that learn to reconstruct the input data. They can be used for dimensionality reduction by training the network to learn a compressed representation of the data.

**Ques 84 Advantages of autoencoders:**

* **Non-linearity:** Autoencoders can capture non-linear relationships between data points.
* **Flexibility:** Autoencoders can be customized for specific tasks.
* **Feature learning:** Autoencoders can learn meaningful features from the data.

**Ques 85 Challenges of Using Dimensionality Reduction Techniques**

Some challenges associated with using dimensionality reduction techniques include:

* **Loss of information:** Dimensionality reduction can lead to loss of information, especially if the data is inherently high-dimensional.
* **Interpretation:** The reduced dimensions may not have a clear interpretation.
* **Computational cost:** Some dimensionality reduction techniques, such as t-SNE, can be computationally expensive for large datasets.

**Ques 86 Impact of Distance Metric on Dimensionality Reduction Techniques**

The choice of distance metric can impact the performance of dimensionality reduction techniques. For example, Euclidean distance may not be suitable for data with non-linear relationships.

**Ques 87 Techniques to Visualize High-Dimensional Data After Dimensionality Reduction**

* **Scatter plots:** Visualize the data in two or three dimensions.
* **Parallel coordinate plots:** Visualize the data using parallel lines to represent each dimension.
* **t-SNE plots:** Visualize the data in a low-dimensional space using t-SNE.

**Ques 88 Feature Hashing and its Use in Dimensionality Reduction**

Feature hashing is a technique that maps high-dimensional categorical features to a lower-dimensional space using hash functions. It can be used for dimensionality reduction and feature engineering.

**Ques 89 Global vs. Local Feature Extraction Methods**

* **Global feature extraction methods:** Extract features from the entire dataset.
* **Local feature extraction methods:** Extract features from local regions of the data.

**Examples:**

* **Global feature extraction:** Bag-of-words, TF-IDF.
* **Local feature extraction:** SIFT, SURF.

**Ques 90 Impact of Feature Sparsity on Dimensionality Reduction Techniques**

Feature sparsity, where many features have zero or few non-zero values, can affect the performance of dimensionality reduction techniques. Some techniques, like PCA, may not be as effective for sparse data.

**Impact of Outliers on Dimensionality Reduction Algorithms**

Outliers can have a significant impact on dimensionality reduction algorithms, especially those that are sensitive to outliers, such as PCA. Outliers can distort the principal components and lead to inaccurate results.

I hope this comprehensive response addresses your questions and provides valuable insights into ensemble techniques, boosting, KNN, and dimensionality reduction.