**Naive Approach:**

**1. What is the Naive Approach in machine learning?**

A. The Naive Approach in machine learning refers to a simple and straightforward method that makes assumptions of independence or uniformity. It often lacks sophistication or consideration of complex relationships within the data, serving as a baseline or starting point for more advanced modeling techniques.

**2. Explain the assumptions of feature independence in the Naive Approach.?**

A. The Naive Approach assumes that the features (input variables) are independent of each other given the class variable. In other words, it assumes that the presence or value of one feature does not affect the presence or value of other features. This assumption allows for simplification in modeling and calculation, but it may not hold true in many real-world scenarios where feature dependencies exist.

**3. How does the Naive Approach handle missing values in the data?**

A. The Naive Approach typically handles missing values by simply ignoring them during training and inference. It assumes that the missing values are missing completely at random (MCAR) and do not carry any meaningful information. Therefore, the Naive Approach treats missing values as if they were not present in the data, and the calculations or predictions are performed based on the available non-missing values. This approach can lead to biased results if the missing values are not MCAR, and more advanced techniques like imputation or data augmentation may be necessary to handle missing data more effectively.

**4. What are the advantages and disadvantages of the Naive Approach?**

A. the Naive Approach has its simplicity and efficiency as advantages, its strong independence assumption and limited expressiveness can limit its applicability and accuracy in more complex tasks. It is often used as a baseline model or in situations where the independence assumption holds reasonably well.

**5. Can the Naive Approach be used for regression problems? If yes, how?**

A. Yes, the Naive Approach can be used for regression problems, although it may not be the most suitable choice in many cases. The Naive Approach for regression typically involves assuming independence between the input features and directly estimating the target variable without considering their relationships.

**6. How do you handle categorical features in the Naive Approach?**

A. In the Naive Approach, categorical features are handled by treating each category as a separate feature and assuming independence between the categories. Each category is represented as a binary variable (0 or 1) indicating the presence or absence of that category in the data. This allows the Naive Approach to consider the influence of each category independently in the modeling process, assuming that the presence or absence of one category does not affect the presence or absence of other categories.

**7. What is Laplace smoothing and why is it used in the Naive Approach?**

A. Laplace smoothing, also known as add-one smoothing or additive smoothing, is a technique used in the Naive Approach to handle the issue of zero probabilities or zero frequencies in categorical features.

In the Naive Approach, probabilities are calculated by counting the occurrences of different categories or values in the training data. However, if a category or value is not observed in the training data, the probability estimate becomes zero, which can cause problems during inference or prediction.

**8. How do you choose the appropriate probability threshold in the Naive Approach?**

A. Choosing the appropriate probability threshold in the Naive Approach depends on the specific task and the trade-off between precision and recall.

In simple terms, the probability threshold determines the minimum probability value that must be exceeded for a sample to be classified as belonging to a particular class. A higher threshold increases precision (fewer false positives) but may decrease recall (more false negatives), while a lower threshold increases recall (fewer false negatives) but may decrease precision (more false positives).

The choice of the threshold depends on the relative importance of precision and recall in the task at hand. For example, in a spam email classification task, precision may be more critical to avoid incorrectly classifying non-spam emails as spam, while recall is also important to minimize the number of spam emails that are missed.

**9. Give an example scenario where the Naive Approach can be applied.**

A. Document Classification: sentiment analysis, topic classification, or spam detection in text documents.

Customer Support Routing

Disease Diagnosis

Fraud Detection

**KNN:**

**10. What is the K-Nearest Neighbors (KNN) algorithm?**

A. The K-Nearest Neighbors (KNN) algorithm is a non-parametric classification or regression algorithm used for supervised learning tasks. In KNN, the class or value of a new data point is predicted based on the majority vote or average of the K nearest neighbors in the training dataset.

**11. How does the KNN algorithm work?**

A. the KNN algorithm works as follows:

Given a training dataset with labeled data points, each having a set of features and corresponding class or value.

When a new data point needs to be classified or predicted, the algorithm identifies the K closest neighbors in the training dataset based on a distance metric (e.g., Euclidean distance).

For classification, the majority class among the K neighbors is assigned to the new data point. For regression, the average or weighted average of the values among the K neighbors is used as the predicted value.

The algorithm returns the predicted class or value for the new data point.

The choice of the value K determines the number of neighbors considered for classification or regression. A larger K value provides a smoother decision boundary but may introduce more bias, while a smaller K value can lead to a more detailed decision boundary but may be more sensitive to noise.

KNN is a simple and intuitive algorithm that does not require explicit training. It can handle multi-class classification and regression problems. However, its performance can be affected by the choice of K, the distance metric, and the dimensionality of the data. Additionally, KNN can be computationally expensive, especially with large training datasets, as it requires computing distances for each new data point.

**12. How do you choose the value of K in KNN?**

A. The value of K in KNN is typically chosen through experimentation and model evaluation. A smaller value of K can lead to more flexible and detailed decision boundaries but may be prone to overfitting, while a larger value of K can provide smoother decision boundaries but may introduce more bias. The optimal value of K depends on the specific dataset and problem at hand, and it is often determined by using techniques like cross-validation or grid search to find the value that maximizes the model's performance metrics.

**13. What are the advantages and disadvantages of the KNN algorithm?**

A. Simple and Intuitive: KNN is easy to understand and implement, making it accessible to beginners in machine learning.

No Training Phase: KNN does not require an explicit training phase, allowing for quick and straightforward deployment.

**Disadvantages of the KNN algorithm:**

Computational Complexity: KNN can be computationally expensive, especially with large datasets, as it requires computing distances for each new data point.

Sensitivity to Feature Scaling: KNN is sensitive to the scale and range of features, requiring proper feature scaling to ensure accurate results.

Memory Usage: KNN requires storing the entire training dataset, which can be memory-intensive for large datasets.

Choosing Optimal K: Selecting the right value of K can be challenging and may require experimentation and model evaluation.

Imbalanced Data: KNN can be biased towards the majority class in imbalanced datasets, requiring techniques like weighting or resampling to mitigate the issue.

**14. How does the choice of distance metric affect the performance of KNN?**

A. The choice of distance metric in KNN can significantly impact the performance of the algorithm. Different distance metrics (e.g., Euclidean, Manhattan, cosine similarity) measure the similarity or dissimilarity between data points differently. The selection of an appropriate distance metric depends on the nature of the data and the underlying relationships. A distance metric that aligns well with the data's characteristics can lead to more accurate and meaningful nearest neighbor selection, resulting in improved performance of the KNN algorithm.

**15. Can KNN handle imbalanced datasets? If yes, how?**

A. KNN can struggle with imbalanced datasets as it tends to be biased towards the majority class. To address this, techniques like weighted KNN, where the influence of each neighbor is weighted based on class frequencies, or resampling techniques like oversampling the minority class or undersampling the majority class can be applied. These methods help to mitigate the impact of class imbalance and improve the performance of KNN on imbalanced datasets.

**16. How do you handle categorical features in KNN?**

A. Categorical features in KNN can be handled by converting them into numerical representations. This can be done by using techniques like one-hot encoding or label encoding, where each category is represented as a binary vector or a numerical label, respectively. This transformation allows the KNN algorithm to calculate distances and compare categorical features effectively.

**17. What are some techniques for improving the efficiency of KNN?**

A. Indexing: Using data structures like KD-trees or ball trees to index the training data can speed up nearest neighbor search by reducing the number of distance computations.

Dimensionality Reduction: Applying dimensionality reduction techniques such as Principal Component Analysis (PCA) or t-SNE to reduce the number of features can lead to faster computations and improved efficiency.

Distance Metrics Approximation: Approximating distance metrics, such as using Locality-Sensitive Hashing (LSH), can provide faster search by reducing the number of exact distance computations required.

Nearest Neighbor Search Algorithms: Utilizing approximate nearest neighbor search algorithms, such as Approximate Nearest Neighbor (ANN) or Random Projection, can offer faster search times while maintaining reasonable accuracy.

**18. Give an example scenario where KNN can be applied.**

A. One example scenario where KNN can be applied is in recommendation systems. By utilizing KNN, similar users or items can be identified based on their features or preferences, and recommendations can be made based on the preferences of those nearest neighbors.

**Clustering:**

**19. What is clustering in machine learning?**

A. Clustering in machine learning is the process of grouping similar data points together based on their intrinsic properties or similarities. It aims to discover patterns or natural divisions within a dataset without any predefined class labels, enabling the identification of underlying structures or clusters in the data.

**20. Explain the difference between hierarchical clustering and k-means clustering.**

A. The main difference between hierarchical clustering and k-means clustering is that hierarchical clustering builds a tree-like structure of clusters, where the number of clusters is not predetermined, while k-means clustering aims to partition the data into a predefined number of clusters by iteratively optimizing cluster centroids based on distance measurements. Hierarchical clustering provides a hierarchy of nested clusters, while k-means clustering assigns each data point to a single cluster.

**21. How do you determine the optimal number of clusters in k-means clustering?**

A. The optimal number of clusters in k-means clustering can be determined using techniques such as the elbow method or silhouette analysis. The elbow method looks for the "elbow" point in the plot of the sum of squared distances between data points and their cluster centroids, indicating the point of diminishing returns in terms of adding more clusters. Silhouette analysis calculates a score for each data point, measuring how well it fits within its own cluster compared to other clusters, and the average silhouette score can help identify the optimal number of clusters with the highest overall cohesion and separation.

**22. What are some common distance metrics used in clustering?**

A. Euclidean Distance: Measures the straight-line distance between two points in Euclidean space.

Manhattan Distance: Calculates the sum of the absolute differences between the coordinates of two points.

Cosine Similarity: Measures the cosine of the angle between two vectors, capturing the direction and magnitude of similarity irrespective of the vector lengths.

Mahalanobis Distance: Accounts for correlations and variances among variables, considering the covariance structure of the data.

Jaccard Distance: Computes the dissimilarity between sets by dividing the size of the intersection by the size of the union of the sets.

**23. How do you handle categorical features in clustering?**

A. Categorical features in clustering can be handled by using appropriate distance metrics or similarity measures designed for categorical data, such as the Jaccard distance or the Hamming distance. Alternatively, one-hot encoding or binary encoding can be applied to convert categorical features into numerical representations, allowing them to be used with standard distance metrics.

**24. What are the advantages and disadvantages of hierarchical clustering?**

A. Advantages of hierarchical clustering:

Hierarchy of Clusters: Hierarchical clustering provides a hierarchical structure of clusters, allowing for a more detailed understanding of the relationships and subgroups within the data.

No Assumption of Cluster Number: Hierarchical clustering does not require a predetermined number of clusters, making it suitable for exploratory analysis or when the optimal number of clusters is unknown.

Disadvantages of hierarchical clustering:

Computational Complexity: Hierarchical clustering can be computationally expensive, especially for large datasets, as it requires pairwise distance calculations and merging of clusters at each level.

Lack of Scalability: The algorithm's computational requirements make it challenging to scale hierarchical clustering to large datasets, limiting its applicability in certain scenarios.

Difficulty in Interpreting Dendrograms: Interpreting dendrograms, the visual representations of hierarchical clustering results, can be subjective and challenging, especially when dealing with complex or large datasets.

**25. Explain the concept of silhouette score and its interpretation in clustering.**

A. The silhouette score is a measure of how well each data point fits within its own cluster compared to other clusters in clustering analysis. It ranges from -1 to 1, where a higher score indicates better clustering quality. A positive silhouette score suggests that the data point is well-matched to its cluster, while a negative score indicates that it may be more similar to neighboring clusters, highlighting potential clustering issues or misclassifications.

**26. Give an example scenario where clustering can be applied.**

A. One example scenario where clustering can be applied is customer segmentation in marketing. By clustering customers based on their purchasing behavior, demographics, or preferences, businesses can identify distinct customer segments and tailor marketing strategies or product offerings to each segment's specific needs and preferences.

**Anomaly Detection:**

**27. What is anomaly detection in machine learning?**

A. Anomaly detection in machine learning is the process of identifying rare, unusual, or abnormal patterns or instances in a dataset that deviate significantly from the expected or normal behavior. It involves building models to differentiate between normal and anomalous data points, helping to detect outliers or potential anomalies that may indicate fraudulent activities, system failures, or other irregularities.

**28. Explain the difference between supervised and unsupervised anomaly detection.**

A. The main difference between supervised and unsupervised anomaly detection lies in the availability of labeled data. In supervised anomaly detection, the algorithm is trained on labeled data that explicitly identifies anomalies, allowing it to learn the patterns of normal and abnormal behavior. In unsupervised anomaly detection, there are no labeled anomalies, and the algorithm learns the normal patterns from unlabeled data, identifying deviations from the learned patterns as potential anomalies.

**29. What are some common techniques used for anomaly detection?**

A. Statistical Methods: These include techniques such as z-score, percentile-based methods, or Gaussian distribution modeling to identify anomalies based on statistical properties.

Machine Learning Approaches: These include supervised methods like one-class SVM or isolation forests, and unsupervised methods like clustering-based approaches or autoencoders, which learn normal patterns and detect deviations as anomalies.

Time Series Analysis: Techniques like ARIMA, exponential smoothing, or change point detection methods are used to identify anomalies in time series data by detecting unusual patterns or sudden changes.

**30. How does the One-Class SVM algorithm work for anomaly detection?**

A. The One-Class SVM (Support Vector Machine) algorithm for anomaly detection works by finding a hyperplane that encloses the majority of the data points in a high-dimensional space, defining the region of normal behavior. Any data points lying outside this region are considered anomalies. It aims to maximize the margin around the normal data points while minimizing the influence of potential outliers or anomalies in the training set.

31. How do you choose the appropriate threshold for anomaly detection?

A. Choosing the appropriate threshold for anomaly detection depends on the desired trade-off between false positives and false negatives. It involves finding the balance between the detection of true anomalies and minimizing the inclusion of normal data points as anomalies. The threshold can be determined by analyzing the precision-recall trade-off, using evaluation metrics such as the receiver operating characteristic (ROC) curve or the F1-score to find the optimal threshold that maximizes performance based on the specific requirements of the application.

**32. How do you handle imbalanced datasets in anomaly detection?**

A. Handling imbalanced datasets in anomaly detection can involve techniques such as:

Resampling: Applying oversampling techniques to increase the representation of the minority class or undersampling techniques to reduce the dominance of the majority class.

Anomaly-specific Algorithms: Utilizing algorithms specifically designed for imbalanced datasets, such as anomaly detection methods that focus on detecting anomalies in the minority class while considering the majority class as normal.

These techniques help address the challenge of imbalanced data and ensure that the anomaly detection model is not biased towards the majority class, allowing for more accurate detection of anomalies.

**33. Give an example scenario where anomaly detection can be applied.**

A. . Anomaly detection can be applied in fraud detection for credit card transactions. By analyzing transaction patterns and identifying deviations from normal behavior, anomaly detection techniques can flag suspicious or fraudulent transactions that exhibit unusual spending patterns, large amounts, or transactions from unfamiliar locations, helping to prevent fraudulent activities and protect users from financial losses.

**Dimension Reduction:**

**34. What is dimension reduction in machine learning?**

A Dimension reduction in machine learning refers to the process of reducing the number of features or variables in a dataset while retaining the essential information. It aims to simplify the dataset's representation, remove irrelevant or redundant features, and alleviate the curse of dimensionality, improving computational efficiency and reducing the risk of overfitting by capturing the most relevant information in a lower-dimensional space.

**35. Explain the difference between feature selection and feature extraction.**

A. The main difference between feature selection and feature extraction lies in their approaches to reducing the dimensionality of a dataset.

Feature selection involves selecting a subset of the original features based on certain criteria, such as their importance or relevance to the target variable, discarding the irrelevant or redundant features.

Feature extraction, on the other hand, creates new transformed features by combining or projecting the original features into a lower-dimensional space, aiming to capture the most informative aspects of the data while discarding less relevant information.

**36. How does Principal Component Analysis (PCA) work for dimension reduction?**

A. Principal Component Analysis (PCA) for dimension reduction works by transforming the original high-dimensional data into a new lower-dimensional space. It identifies the principal components, which are orthogonal linear combinations of the original features, capturing the maximum variance in the data. By selecting a subset of the principal components that explain a significant portion of the variance, PCA reduces the dimensionality while preserving the most important information in the data.

**37. How do you choose the number of components in PCA?**

A. The number of components in Principal Component Analysis (PCA) is typically chosen by considering the cumulative explained variance ratio. By analyzing the proportion of variance explained by each component and plotting the cumulative explained variance, a suitable number of components can be selected that retains a significant portion of the variance. The elbow or knee point in the plot can indicate the optimal number of components, balancing the trade-off between dimensionality reduction and information preservation.

**38. What are some other dimension reduction techniques besides PCA?**

A. t-SNE (t-Distributed Stochastic Neighbor Embedding): A nonlinear technique that maps high-dimensional data into a lower-dimensional space, emphasizing the preservation of local structure and capturing complex relationships.

LDA (Linear Discriminant Analysis): A technique commonly used for supervised dimension reduction, where the objective is to maximize class separability by projecting the data onto a lower-dimensional space that maximizes between-class scatter and minimizes within-class scatter.

Autoencoders: Neural network-based models that aim to learn a compressed representation of the data by training an encoder and decoder network, effectively reducing the dimensionality while capturing important features of the data.

**39. Give an example scenario where dimension reduction can be applied.**

A. Dimension reduction can be applied in image processing, such as facial recognition. By reducing the dimensionality of facial images while preserving important features, dimension reduction techniques can extract essential facial characteristics and enable efficient face recognition algorithms by reducing computational complexity and improving classification accuracy.

**Feature Selection:**

**40. What is feature selection in machine learning?**

A. Feature selection in machine learning refers to the process of selecting a subset of the original features in a dataset that are most relevant or informative for building an accurate predictive model. It involves evaluating the importance or usefulness of each feature based on certain criteria, such as statistical measures, feature importance scores, or domain knowledge, and retaining only the most informative features while discarding irrelevant or redundant ones.

**41. Explain the difference between filter, wrapper, and embedded methods of feature selection.**

A. The main difference between filter, wrapper, and embedded methods of feature selection lies in how they incorporate the feature selection process into the machine learning workflow.

Filter methods select features based on their intrinsic characteristics and relevance to the target variable, without involving a specific learning algorithm.

Wrapper methods evaluate different subsets of features by using a specific learning algorithm and measuring their performance on a validation set, resulting in more accurate but computationally expensive feature selection.

Embedded methods perform feature selection during the model training process, where the selection of features is directly embedded within the learning algorithm, resulting in a more efficient and integrated approach.

**42. How does correlation-based feature selection work?**

A. Correlation-based feature selection works by calculating the correlation coefficient between each feature and the target variable, ranking the features based on their correlation values, and selecting the top-ranked features that exceed a predefined threshold. The selected features are considered the most relevant ones for modeling.

**43. How do you handle multicollinearity in feature selection?**

A. To handle multicollinearity in feature selection, you can analyze the pairwise correlations between the selected features and remove any redundant or highly correlated features. This helps ensure that the selected features provide unique and independent information, reducing the impact of multicollinearity on the model's performance.

**44. What are some common feature selection metrics?**

A. Some common feature selection metrics include:

Mutual Information: Measures the amount of information that one feature provides about the target variable.

Feature Importance: Assess the importance of features based on their contribution to a predictive model, such as decision trees or random forests.

Correlation Coefficient: Determines the linear relationship between each feature and the target variable, indicating their strength of association.

**45. Give an example scenario where feature selection can be applied.**

A. An example scenario where feature selection can be applied is in the field of healthcare for predicting disease outcomes. By selecting the most relevant features from medical records, such as age, gender, vital signs, and lab test results, feature selection can help identify the key factors that contribute to predicting disease progression or treatment response, enabling personalized and effective healthcare interventions.

**Data Drift Detection:**

**46. What is data drift in machine learning?**

A. Data drift refers to the phenomenon where the statistical properties of the input data used for training a machine learning model change over time, resulting in a mismatch between the training and deployment environments. This can lead to degraded model performance as the model is no longer effectively capturing the new patterns and relationships present in the evolving data distribution.

**47. Why is data drift detection important?**

A. Data drift detection is important because it helps identify when the underlying data distribution has changed, which can impact the performance and reliability of machine learning models. By detecting data drift, one can assess the need for model retraining, adaptation, or other corrective measures to ensure that the model remains accurate and effective in the face of evolving data patterns.

**48. Explain the difference between concept drift and feature drift.**

A. Concept drift refers to the change in the underlying relationship between input features and the target variable over time. It means that the target variable's distribution or the relationship between the input features and the target variable is different between the training and deployment environments. Feature drift, on the other hand, refers to the change in the distribution of the input features themselves over time, while the relationship with the target variable remains the same. Feature drift affects the input data, while concept drift affects the relationship between inputs and outputs.

**49. What are some techniques used for detecting data drift?**

A. Statistical Measures: Utilizing statistical tests, such as Kolmogorov-Smirnov, to compare the distributions of the training and incoming data.

Monitoring Drift Indicators: Tracking and monitoring specific indicators or metrics, such as mean, variance, or entropy, to detect significant deviations over time and signal data drift.

**50. How can you handle data drift in a machine learning model?**

A. To handle data drift in a machine learning model, you can employ the following approaches:

Monitoring and Retraining: Continuously monitor the performance of the model and periodically retrain it using updated data to adapt to the changing data distribution.

Online Learning: Implement online learning techniques that allow the model to adapt and update itself in real-time as new data becomes available, maintaining model accuracy in dynamic environments affected by data drift.

**Data Leakage:**

**51. What is data leakage in machine learning?**

A. Data leakage in machine learning refers to the situation where information from outside the training data is inadvertently used during model development, leading to overly optimistic performance evaluations. It can occur when features or information that would not be available in a real-world setting are used to train or evaluate the model, resulting in misleading performance metrics and potentially ineffective or biased models.

**52. Why is data leakage a concern?**

Data leakage is a concern because it can lead to inflated performance metrics and unrealistic expectations of model performance in real-world scenarios. It can cause models to appear more accurate than they actually are, resulting in poor generalization and unreliable predictions. Data leakage can also introduce biases, undermine model interpretability, and compromise the integrity and fairness of the machine learning process.

**53. Explain the difference between target leakage and train-test contamination.**

A. Target leakage refers to a situation where information from the target variable is inadvertently included in the training data, leading to unrealistically high model performance. It occurs when data that would not be available during prediction is used to create features or make decisions. Train-test contamination, on the other hand, happens when information from the test set leaks into the training process, artificially inflating model performance during evaluation and leading to overfitting.

**54. How can you identify and prevent data leakage in a machine learning pipeline?**

A. To identify and prevent data leakage in a machine learning pipeline, you can follow these steps:

Careful Feature Engineering: Ensure that features are created using only information that would be available at the time of prediction, avoiding any use of future or target-related information.

Proper Train-Test Split: Establish a clear separation between the training and testing datasets, ensuring that no data from the test set is used during the model development process to prevent train-test contamination. Cross-validation techniques can also be used to prevent leakage within the training set.

**55. What are some common sources of data leakage?**

A. Some common sources of data leakage include:

Leaky Features: Features that contain information about the target variable that would not be available during prediction, such as future values or derived from the target variable itself.

Time-Based Data Leakage: Using future information or data from overlapping time periods between training and testing, which can occur in time series or temporal data analysis. Other sources may include:

Data Preprocessing: Applying preprocessing steps, such as scaling or normalization, based on information from the entire dataset, including the test set.

Data Collection Process: Including information in the dataset that is not available at the time of prediction, such as incorporating user-specific data or variables collected after the target event.

**56. Give an example scenario where data leakage can occur.**

**Cross Validation:**

**57. What is cross-validation in machine learning?**

A. Cross-validation is a technique used in machine learning to assess the performance and generalization ability of a model. It involves dividing the dataset into multiple subsets, training the model on some subsets, and evaluating it on the remaining subset, iteratively. This helps in estimating the model's performance on unseen data and detecting overfitting.

**58. Why is cross-validation important?**

A. Cross-validation is important in machine learning because it provides a more robust and reliable estimate of a model's performance. It helps to mitigate the risk of overfitting and allows for better comparison and selection of different models or algorithms. Additionally, it aids in identifying potential issues such as data bias and instability in model performance.

**59. Explain the difference between k-fold cross-validation and stratified k-fold cross-validation.**

A. K-fold cross-validation involves dividing the dataset into k equal-sized folds and using one fold as a validation set while training on the remaining k-1 folds. This process is repeated k times, with each fold serving as the validation set once. Stratified k-fold cross-validation maintains the class distribution in each fold, ensuring that the proportions of different classes are preserved across the folds. This is particularly useful when dealing with imbalanced datasets, where certain classes may be underrepresented.

**60. How do you interpret the cross-validation results?**

A. To interpret cross-validation results, look at the average performance metrics across all the folds. This provides an estimate of the model's generalization ability. Additionally, consider the variance or standard deviation of the performance metrics to assess the stability and consistency of the model's performance. Higher average scores and lower variance indicate better model performance.