

Machine Learning Interview Questions and Answers

1) What is the difference between Parametric and Non-Parametric Algorithms?

Parametric models:

- ✓ These algorithms assume a fixed, predefined structure for the model, such as linear or polynomial relationships.
- ✓ They rely on a finite number of parameters (e.g., weights in Linear Regression).
- ✓ They are computationally efficient, making them faster to train.
- ✓ However, they are less flexible and may underperform on complex datasets if the assumptions about the data's structure are incorrect.
- ✓ **Examples:** Linear Regression, Logistic Regression, and SVM with a linear kernel.

Non-Parametric models:

- ✓ These algorithms do not assume any specific structure or distribution for the data.
- ✓ They are more flexible and capable of capturing complex patterns since the model adapts as more data becomes available.
- ✓ However, they require more data to generalize well and are computationally expensive because they often store the entire dataset.
- ✓ **Examples:** k-NN, Decision Trees, Random Forest, and SVM with RBF kernel.

2) Difference between convex and non-convex cost function; what does it mean when a cost function is non-convex?

Convex:

- ✓ A convex function has a bowl-shaped curve where any line drawn between two points on the curve lies above or on the curve.
- ✓ It has a single global minimum, which makes optimization straightforward since gradient descent will always converge to that minimum.
- ✓ Example: Mean Squared Error (MSE) in Linear Regression.

Non-Convex:

- ✓ A non-convex function has a curve with multiple local minima and maxima, meaning it is not bowl-shaped.
- ✓ The presence of multiple minima makes optimization challenging since gradient descent might get stuck in a local minimum instead of reaching the global minimum.
- ✓ Example: Cost functions in deep learning (e.g., neural networks), which are typically non-convex due to the high-dimensional and complex structure of the parameter space.

3) How do you decide when to go for deep learning for a project?

Deciding to use deep learning depends on the complexity of the problem, the data type, and available resources. Deep learning is ideal for solving complex tasks involving unstructured data like images, videos, or text because of its ability to automatically extract meaningful features. However, it requires large datasets to perform well and significant computational power, like GPUs, for training. For smaller datasets or structured data, traditional machine learning models are often more practical and efficient. Additionally, deep learning models are less interpretable, which might be a limitation for applications requiring transparency, such as healthcare or finance. If resources and data size permit, and the problem demands state-of-the-art performance, deep learning can be a strong choice.

4) Give an example of when False positive is more crucial than false negative and vice versa?

- ✓ **False Positive is More Crucial:** In fraud detection systems, flagging a legitimate transaction as fraudulent (false positive) can inconvenience customers and damage the user experience. This is more critical in high-value scenarios, such as large business transactions, where trust is paramount.
- ✓ **False Negative is More Crucial:** In medical diagnosis, missing a life-threatening condition like cancer (false negative) is far more severe because it delays treatment and can result in fatal consequences. It's better to flag a condition as positive, even if it requires additional tests to confirm.

5) Why is "Naive" Bayes naive?

Naive Bayes is considered 'naive' because it makes a strong assumption that all features in the dataset are independent of each other, given the class label. This assumption rarely holds true in real-world scenarios, as features often have some level of correlation. Despite this simplification, Naive Bayes performs surprisingly well in many applications, especially in text classification and spam detection, where feature independence is approximately valid. The 'naive' assumption simplifies the computation of probabilities, making the algorithm efficient and easy to implement.

6) Give an example where the median is a better measure than the mean?

The median is a better measure of central tendency than the mean when the data contains outliers or is skewed. For example, consider a dataset of house prices in a city where most houses cost around \$300,000, but a few luxury homes are priced at \$10 million. The mean would be heavily influenced by these extreme values and may give a misleading impression of the typical house price. In contrast, the median, which represents the middle value of the sorted data, is unaffected by outliers and provides a more accurate representation of the central tendency in such cases.

7) Why KNN is known as a lazy learning technique?

K-Nearest Neighbors (KNN) is called a lazy learning technique because it doesn't learn an explicit model during the training phase. Instead, it simply stores the training data and defers all computation until prediction time. When making predictions, KNN calculates the distance between the query point and all stored data points, identifies the k-nearest neighbors, and predicts based on their labels. This 'lazy' approach contrasts with algorithms like decision trees or logistic regression, which build a model during training and make predictions based on that model.

8) What do you mean by semi supervised learning?

Semi-supervised learning is a type of machine learning that lies between supervised and unsupervised learning. In semi-supervised learning, the model is trained on a small amount of labeled data and a large amount of unlabeled data. The labeled data provides guidance for the model, while the unlabeled data helps the model learn underlying patterns and generalize better. This approach is useful when labeling data is expensive or time-consuming, but there's an abundance of unlabeled data available. It leverages both to improve the model's performance without requiring a large amount of labeled data.

Example: Image classification.

9) What is an OOB error and how is it useful?

The Out-of-Bag (OOB) error is an internal validation method used in ensemble learning algorithms like Random Forest. During training, each decision tree in a Random Forest is trained on a bootstrap sample, meaning a random subset of the data is selected with replacement. Some data points are not included in the bootstrap sample and are left out of the training of each tree; these are called 'out-of-

bag' points. The OOB error is calculated by using these out-of-bag points to test the model. This provides an unbiased estimate of the model's performance without needing a separate validation set. It's especially useful when you have limited data and want to make the most of it.

10) In what scenario decision tree should be preferred over random forest?

- ✓ **Interpretability:** When you need to easily explain the model's logic and decisions.
- ✓ **Speed and simplicity:** When computational efficiency is a concern and you don't need the complexity of an ensemble method.
- ✓ **Small datasets:** When you have a small dataset and overfitting is less of a concern, a decision tree might be sufficient.
- ✓ **Less computational resources:** When you need a less resource-intensive solution for prediction and training.

11) Why Logistic Regression is called regression?

Logistic Regression is called 'regression' because it models the relationship between the dependent variable and independent variables using a regression-based approach, even though its output is categorical (binary). The term 'regression' comes from its method of fitting a line (or curve) to data. In logistic regression, the model first calculates a linear combination of the input features (like in linear regression) and then applies a logistic (sigmoid) function to transform this into a probability between 0 and 1. This is used for binary classification. Despite being used for classification, it's called regression because it uses a linear equation to estimate the outcome before applying a transformation.

12) What is No Free Lunch Theorem?

The No Free Lunch (NFL) Theorem essentially states that no single machine learning algorithm works best for every problem. In other words, the performance of an algorithm is highly dependent on the specific dataset and problem it is being applied to. If an algorithm performs well on one type of problem, it doesn't guarantee that it will perform well on another. This means that, for any algorithm, there will always be some problems for which it is the best performer, and others where it will perform poorly. The theorem emphasizes the importance of experimenting with different algorithms and selecting the one that is most suited to the specific characteristics of your data and problem.

13) Imagine you are working with a laptop of 2GB RAM, how would you process a dataset of 10GB?

- ✓ **Use Data Streaming:** Instead of loading the entire dataset into memory, I would process it in chunks by streaming the data in small batches that fit into memory. Libraries like Pandas offer ways to load data in chunks using the chunksize parameter, allowing for efficient memory management.
- ✓ **Data Compression:** I would consider compressing the dataset into a format like CSV.gz or Parquet, which reduces the file size, and then load it in a more efficient, columnar format.
- ✓ **Out-of-Core Learning:** If applicable, I would use out-of-core learning techniques, which allow machine learning algorithms to train on data that doesn't fit in memory by processing it in chunks. Libraries like Scikit-learn and Dask provide out-of-core learning options.
- ✓ **Use a Database:** I would store the data in a database like SQLite or PostgreSQL and perform SQL queries to process the data in smaller, manageable chunks, only loading relevant data at any given time.

- ✓ **Distributed Computing:** If the dataset is large and the resources allow, I could use a distributed framework like Dask or Apache Spark to distribute the workload across multiple nodes and overcome the memory limitations of a single machine.
- ✓ **Increase Swap Space:** While this is not the most efficient option, increasing the swap space (virtual memory) could help when running out of physical RAM, though it can significantly slow down processing due to reliance on the hard disk.
- ✓ **Cloud or External Resources:** If none of the above strategies are sufficient, I would consider using cloud resources (e.g., AWS, Google Cloud) or external servers to handle larger datasets more effectively.

14) What are the main points of difference between Bagging and Boosting?

Bagging focuses on reducing variance and works with parallel models, while Boosting focuses on reducing bias and works with sequential models, adjusting the focus on difficult data points. *Random Forest* is a popular example of bagging. Algorithms like *AdaBoost*, *Gradient Boosting*, and *XGBoost* are examples of boosting.

15) How do you measure the accuracy of a Clustering Algorithm?

The choice of metric depends on whether ground-truth labels are available. External metrics like ARI (Adjusted Rand Index) and NMI (Normalized Mutual Information) are used with labels, while internal metrics like Silhouette Score and Inertia are used when labels are unavailable. Clustering accuracy is more about evaluating structure and separation rather than direct predictions.

- ✓ **Adjusted Rand Index (ARI):** Measures the similarity between the true labels and the clustering results, adjusted for chance.
- ✓ **Normalized Mutual Information (NMI):** Evaluates how much information the predicted clusters share with the true labels, normalized to prevent bias.
- ✓ **Silhouette Score:** Measures how well clusters are separated and how cohesive individual clusters are. A score close to 1 indicates well-separated clusters.
- ✓ **Inertia (within-cluster sum of squares):** Measures the compactness of clusters but does not directly account for separation between them.

16) What is Matrix Factorization and where is it used in Machine Learning?

Matrix Factorization is a way to break a big table of data into smaller pieces to understand it better. Imagine you have a table where rows represent users, columns represent movies, and the entries show how much a user liked a movie. But most of the table is empty because users haven't rated every movie.

Matrix Factorization splits this table into two smaller tables:

1. One table shows users and their preferences (like how much they like action or comedy).
2. The other table shows movies and their features (like how much action or comedy each movie has).

When you multiply these two smaller tables, you can fill in the missing ratings in the big table. This helps recommend movies to users based on their preferences.

It's mostly used in recommendation systems, like Netflix or Spotify, to suggest things you might like, even if you haven't rated or seen them before.

17) What is an Imbalanced Dataset and how can one deal with this problem?

An imbalanced dataset is when one class in a classification problem has far more samples than the other, like 99% legitimate transactions and 1% fraud in fraud detection. This can lead the model to

favor the majority class, giving misleading results. To handle this, you can balance the data by oversampling the minority class (e.g., SMOTE), undersampling the majority class, or using algorithms that handle imbalance well by adjusting class weights. Additionally, you should evaluate the model using metrics like F1-score or ROC-AUC instead of accuracy to ensure fair assessment.

18) How do you measure the accuracy of a recommendation engine?

- ✓ **Precision:** Measures how many of the recommended items are relevant (e.g., out of the top 10 recommendations, how many are actually liked by the user).
- ✓ **Recall:** Measures how many of the relevant items are recommended (e.g., out of all items the user likes, how many were recommended).
- ✓ **F1-Score:** A balance between precision and recall.
- ✓ **Mean Average Precision (MAP):** Evaluates ranking by considering the relevance of recommendations and their order.
- ✓ **Root Mean Square Error (RMSE) or Mean Absolute Error (MAE):** Used when predicting ratings to measure the difference between predicted and actual ratings.
- ✓ **Hit Rate:** Measures how often a relevant item appears in the top-N recommendations.
- ✓ **Diversity and Novelty:** Evaluates how varied and new the recommendations are to avoid redundancy.

19) What are some ways to make your model more robust to outliers?

To make your model more robust to outliers, you can use techniques like scaling your features with robust methods (e.g., RobustScaler) that are less sensitive to extreme values. Consider using algorithms like tree-based models (e.g., Random Forest, XGBoost) or support vector machines (SVM) with appropriate kernels, as they are inherently less affected by outliers. For regression problems, switch to robust regression techniques like Huber regression or quantile regression. Additionally, detect and handle outliers through methods like the IQR (Interquartile Range) or z-score and either remove, cap, or transform them. Finally, applying log transformation or other nonlinear transformations can help mitigate the impact of outliers in skewed distributions.

20) How can you measure the performance of a dimensionality reduction algorithm on your dataset?

To measure the performance of a dimensionality reduction algorithm, you can evaluate how well the reduced data preserves the structure and information of the original dataset. Common methods include reconstructing the original data from the reduced dimensions and calculating the reconstruction error (e.g., using mean squared error). You can also assess how well the reduced data performs in downstream tasks, like classification or clustering, by comparing metrics such as accuracy or silhouette score before and after dimensionality reduction. Visualization techniques, like plotting 2D or 3D projections, can help analyze whether the reduced data retains separability between classes or clusters.

21) What is Data Leakage? List some ways using which you can overcome this problem.

Data Leakage occurs when information from outside the training dataset leaks into the model during training, giving it an unfair advantage and leading to overly optimistic performance during validation or testing. This results in a model that does not generalize well to unseen data.

Ways to Overcome Data Leakage:

1. **Separate Train-Test Data Properly:** Ensure that no information from the test set influences the training process. Always split the data before any preprocessing.

2. **Avoid Preprocessing on Entire Dataset:** Perform feature scaling, encoding, or transformations only on the training data and then apply the same transformations to the test set.
3. **Handle Temporal Data Carefully:** In time series data, avoid using future information to predict past or present events.
4. **Avoid Target Leakage:** Exclude features created from the target variable, e.g., including future sales data while predicting current sales.
5. **Cross-Validation:** Use techniques like k-fold cross-validation to ensure robust model evaluation without leakage.
6. **Feature Selection:** Perform feature selection only on the training data, not the entire dataset.

22) What is Multicollinearity? How to detect it? List some techniques to overcome Multicollinearity.

Multicollinearity occurs when two or more independent variables in a dataset are highly correlated, meaning they provide redundant information to the model. This can make it difficult for algorithms to determine the individual effect of each variable, leading to unstable coefficients in regression models and reducing interpretability.

How to Detect Multicollinearity:

1. **Variance Inflation Factor (VIF):** Calculate VIF for each variable. A VIF value greater than 5 or 10 indicates multicollinearity.
2. **Correlation Matrix:** Check the pairwise correlation coefficients. High correlations (e.g., above 0.8) may indicate multicollinearity.
3. **Eigenvalues of the Covariance Matrix:** Small eigenvalues indicate multicollinearity.

Techniques to Overcome Multicollinearity:

1. **Remove Highly Correlated Variables:** Drop one of the variables that are highly correlated.
2. **Principal Component Analysis (PCA):** Reduce the dimensionality of the dataset by creating uncorrelated components.
3. **Regularization Techniques:** Use methods like Ridge Regression (L2 regularization) or Lasso Regression (L1 regularization) to penalize large coefficients and reduce multicollinearity.
4. **Combine Variables:** Merge correlated variables into a single feature, e.g., by averaging or creating interaction terms.
5. **Increase Sample Size:** More data can help mitigate the impact of multicollinearity on model stability.

23) How do you approach a categorical feature with high cardinality?

A categorical feature with high cardinality refers to a feature in a dataset that is categorical (i.e., it contains discrete, non-numeric values like labels or categories) and has a large number of unique values or levels.

Example:

- A "Country" feature with 200 unique country names.
- A "User ID" or "Product ID" feature with thousands or even millions of unique values.

Why is it a challenge?

1. **Increased Model Complexity:** High cardinality can lead to a large number of features if one-hot encoding is used, which increases the dimensionality of the data.
2. **Overfitting:** Some categories may have very few samples, which can lead to overfitting to those rare categories.

3. **Memory and Computational Overhead:** Large numbers of categories can significantly increase memory usage and computational time.

When dealing with a **categorical feature with high cardinality** (i.e., many unique values), it is important to handle it carefully to avoid overfitting, increased model complexity, or memory issues. Here's how you can approach it:

1. **Frequency Encoding:** Replace categories with their frequency or count in the dataset. For example, assign a value based on how often each category appears.
2. **Target Encoding:** Replace each category with the mean (or other aggregation) of the target variable for that category. Be cautious of data leakage and use cross-validation.
3. **Hash Encoding:** Convert categories to hashed numerical values, reducing dimensionality while maintaining uniqueness.
4. **Clustering Similar Categories:** Group similar categories together based on business logic, domain knowledge, or statistical similarity (e.g., grouping rare categories as "others").
5. **Dimensionality Reduction:** Use techniques like PCA or embeddings (e.g., word embeddings for text-based categories) to represent high-cardinality categories in lower dimensions.
6. **Leave-One-Out Encoding:** Similar to target encoding, but it excludes the current row while calculating the mean for a category to reduce data leakage.
7. **One-Hot Encoding (Selective):** Use this only if the cardinality is moderate, as high cardinality will create a large number of sparse columns, which may increase computational overhead.

24) Explain Pruning in Decision Trees and how it is done?

Pruning in Decision Trees is the process of reducing the size of a decision tree by removing sections of the tree that provide little to no additional predictive power. It is done to prevent overfitting, improve generalization, and make the model simpler and more interpretable.

How Pruning is Done:

1. **Pre-Pruning (Early Stopping):** Stop the tree's growth early during training by setting constraints such as:
 - Maximum depth of the tree.
 - Minimum number of samples required to split a node.
 - Minimum gain in information (or reduction in impurity) required to make a split.
2. **Post-Pruning (Reduced Error Pruning):** Grow the tree fully and then prune back unnecessary branches:
 - Evaluate the impact of removing a subtree or branch on validation data.
 - Prune branches that do not reduce the overall error rate or provide marginal improvement.

25) What is ROC-AUC curve? List some of its benefits?

The ROC-AUC curve is a performance metric for classification models, where the ROC curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at different thresholds. The AUC (Area Under the Curve) measures the model's ability to distinguish between classes, with a value closer to 1 indicating better performance.

- **Threshold Selection:** It helps you decide the optimal probability threshold for classifying a prediction as positive or negative, depending on the trade-off between **True Positive Rate (TPR)** and **False Positive Rate (FPR)**.

- **Model Comparison:** The **Area Under the Curve (AUC)** provides a single scalar value to compare the performance of different models. A higher AUC indicates a better model in distinguishing between classes.
- **Class Imbalance:** Unlike accuracy, the ROC curve remains effective in assessing performance for imbalanced datasets, as it evaluates TPR and FPR independently of class distribution.
- **Diagnostic Tool:** It helps identify whether the model is biased towards a specific class (e.g., too many false positives or negatives), aiding in model tuning.

26) What are kernels in SVM? Can you list some popular SVM kernels.

In SVM (Support Vector Machines), kernels are techniques used to transform your data into a higher-dimensional space where it is easier to find a dividing line (or hyperplane) between different classes. This allows SVM to classify data that is not easily separable with a straight line in its original form.

Popular SVM Kernels:

1. **Linear Kernel:** This is used when the data can be separated by a straight line or hyperplane. It's the simplest and most efficient kernel when the data is linearly separable.
2. **Polynomial Kernel:** Used for data where the relationship between features is non-linear but can be modelled with polynomials (e.g., quadratic or cubic functions). It allows more flexibility in the decision boundary.
3. **Radial Basis Function (RBF) Kernel:** This is one of the most commonly used kernels because it can handle complex, non-linear data. It works well when there's no clear pattern in how data points are distributed.
4. **Sigmoid Kernel:** This kernel is inspired by the sigmoid function used in neural networks. It's less common, but may be useful in specific scenarios.

27) What is the difference between Gini Impurity and Entropy? Which one is better and why?

When we build a decision tree, we split the data at each step. To decide where to split, we use measures like Gini Impurity or Entropy.

- **Gini Impurity:** Think of it as a measure of "mistakes." If we randomly pick a data point and classify it based on the split, Gini tells us the chances of getting it wrong. Lower Gini = better split.
- **Entropy:** It's a measure of "disorder" or uncertainty in the data. Higher Entropy = more mixed-up data. Lower Entropy = more pure groups (e.g., mostly 0s or 1s).

Both methods help us find splits that make the groups as "pure" as possible. Gini is faster to calculate, while Entropy can give deeper insights. But they often lead to similar results, so many people prefer Gini for its speed.

28) Why does L2 regularization give sparse coefficients?

In the context of machine learning and statistics, sparsity refers to a situation where most of the coefficients (or values) in a dataset or model are zero. It implies that only a few features or variables are actively contributing to the model or dataset, while the rest have no effect.

Actually, L2 regularization does not give sparse coefficients—that's a property of L1 regularization.

L2 Regularization (Ridge Regression)

- L2 regularization adds a penalty proportional to the square of the coefficients to the loss function.
- This shrinks the coefficients toward zero but **does not make them exactly zero**.

- As a result, all features remain in the model, though their impact is reduced.

Key takeaway: L2 regularization **does not produce sparsity**. It's useful when you want to keep all features but reduce their influence to avoid overfitting.

L1 Regularization (Lasso Regression)

- L1 regularization adds a penalty proportional to the absolute value of the coefficients to the loss function.
- This can shrink some coefficients exactly to **zero**, effectively **removing less important features** from the model.
- This happens because the penalty for L1 regularization creates a sharp, "pointy" constraint boundary (in contrast to the rounded boundary in L2).

Key takeaway: L1 regularization **produces sparse coefficients** and is often used for feature selection.

29) List some ways using which you can improve a model's performance?

- Engineer better features or remove irrelevant ones.
- Tune hyperparameters using Grid Search or Random Search.
- Use regularization (L1/L2) to prevent overfitting.
- Apply cross-validation for reliable evaluation.
- Increase training data or use data augmentation.
- Experiment with ensemble methods (e.g., Random Forest, XGBoost).
- Handle class imbalance with resampling or weighted loss functions.
- Optimize decision thresholds for better classification results.

30) Can PCA be used to reduce the dimensionality of a highly nonlinear dataset?

PCA is primarily a linear dimensionality reduction technique, meaning it works best when the dataset's variance is captured in linear combinations of features. For highly nonlinear datasets, PCA may not effectively capture complex structures or relationships.

In such cases, nonlinear techniques like t-SNE, UMAP, or Kernel PCA (which uses kernels to handle nonlinearity) are better suited. However, PCA can still be a good preprocessing step to remove noise or reduce dimensions before applying more complex methods.

31) What's the difference between probability and likelihood?

To summarize, probability is about predicting outcomes given parameters, while likelihood is about estimating parameters given observed outcomes.

32) What cross-validation technique would you use on a time series data set?

Time series data is special because the order of the data points (e.g., past, present, future) matters. You can't shuffle the data randomly, as you would in regular cross-validation, because it would mix up the time sequence. Instead, you use methods that respect the timeline:

1. Walk-Forward Validation

- Start with a small chunk of data for training (e.g., January) and test on the next chunk (e.g., February).
- Then expand the training set (e.g., January + February) and test on the next chunk (e.g., March).

- Repeat this process until you cover the whole dataset.
This way, you simulate how a model would make predictions in the real world—always using the past to predict the future.

2. Rolling Window Validation

- Instead of growing the training set, keep the training set size fixed.
- For example, train on January and test on February, then slide the window forward: train on February and test on March.

This helps test how the model performs across different time periods.

33) Why do we always need the intercept term in a regression model??

The intercept in a regression model is the starting point of the line (or curve) that the model fits. It shows what the outcome (target variable) would be if all the input features (independent variables) are zero. Here's why it's important:

- Why include it? If we don't include the intercept, the model will assume the line starts at zero, which may not match the real-world data. For example, predicting someone's salary based only on their years of experience—does someone with zero experience always earn \$0? Not necessarily.
- What happens without it? Without an intercept, the model can't shift up or down to better fit the data. This can lead to bad predictions.

In simple terms, the intercept lets the model start at the right place, making predictions more accurate.

34) Which Among These Is More Important Model Accuracy or Model Performance?

Model accuracy refers to how often the model makes correct predictions. It's a straightforward metric but may not always reflect the true effectiveness of a model, especially in cases of imbalanced datasets.

Model performance is a broader concept that includes metrics like precision, recall, F1 score, ROC-AUC, and more. It evaluates how well the model meets the specific goals of the task. For example, in a healthcare application, recall (sensitivity) might be more important than accuracy because identifying all positive cases (e.g., a disease) is critical.

35) What is active learning and where is it useful?

Active learning is a machine learning technique where the model actively selects the most informative data points from an unlabeled dataset for labeling by a human or another oracle. Instead of training on all available data, the model focuses on data that is likely to improve its performance the most.

Where it's useful:

Active learning is particularly useful when labeled data is scarce, expensive, or time-consuming to obtain. For example, in medical imaging, where expert annotations are required, active learning can reduce labeling efforts by focusing on ambiguous or high-value cases. Similarly, it's used in applications like fraud detection or sentiment analysis, where certain data points are more complex or uncertain.

36) Why is Ridge Regression called Ridge?

Ridge Regression is called "Ridge" because it introduces a ridge (or a penalty) to the regression coefficients by adding an L2 regularization term to the cost function. This term is the square of the magnitude of coefficients, which helps shrink them toward zero without making them exactly zero, unlike Lasso regression.

The name "ridge" reflects the fact that this technique constrains the coefficients to lie within a certain boundary, forming a ridge-like structure in the optimization space. This helps prevent overfitting and makes the model more robust, especially when dealing with multicollinearity or high-dimensional datasets.

37) State the differences between causality and correlation?

Causality refers to a relationship where one variable directly influences another. For example, smoking causes an increased risk of lung cancer. **Correlation**, on the other hand, measures the statistical association between two variables but does not imply one causes the other. For instance, ice cream sales and drowning incidents may be correlated because both increase in summer, but one does not cause the other.

38) Does it make any sense to chain two different dimensionality reduction algorithms?

Yes, chaining two dimensionality reduction algorithms can make sense in certain scenarios. For instance, you might first use **Principal Component Analysis (PCA)** to reduce dimensionality by capturing most of the variance in the data, followed by a method like **t-SNE** or **UMAP** for further reducing dimensions while preserving local structures. This approach can help in visualizing high-dimensional data effectively or preparing data for clustering.

39) If a Decision Tree is underfitting the training set, is it a good idea to try scaling the input features?

Scaling input features is not effective for decision trees because they are not sensitive to the scale of the features. If a decision tree is underfitting, the issue is usually with the model's complexity, such as a shallow tree. In such cases, adjusting hyperparameters like increasing the tree depth or reducing the minimum sample size for splits would be more helpful.

40) How do we interpret weights in linear models?

In linear models, the weights (or coefficients) represent the relationship between the input features and the target variable. Specifically, they indicate how much change in the target variable is expected for a one-unit change in the corresponding feature, while holding other features constant.

Here's how you interpret them:

1. **Positive Weights:** A positive weight means that as the feature increases, the predicted value of the target variable also increases.
2. **Negative Weights:** A negative weight indicates that as the feature increases, the predicted value of the target variable decreases.
3. **Magnitude of the Weight:** The larger the magnitude of the weight (whether positive or negative), the stronger the influence that feature has on the prediction.
4. **Intercept (Bias Term):** The intercept term is the predicted value when all features are zero. It is a baseline prediction in the absence of any features.

For example, if you have a linear regression model predicting house prices, and the weight for the "size of the house" feature is 1000, this means that for every additional square foot of house size, the predicted house price increases by 1000 units, assuming all other features remain the same.

41) Why is it important to scale the inputs when using SVMs?

When using Support Vector Machines (SVMs), it is crucial to scale the inputs because SVMs are sensitive to the scale of the features. This is due to the fact that SVMs rely on calculating the distance between data points (e.g., using the dot product) to find the optimal hyperplane. If the features have

different scales, the SVM model may give disproportionate importance to features with larger scales, potentially leading to suboptimal performance.

Scaling the data ensures that each feature contributes equally to the model, preventing features with larger values from dominating the optimization process. Common scaling techniques include Standardization (subtracting the mean and dividing by the standard deviation) or Min-Max Scaling (rescaling the features to a fixed range, typically $[0, 1]$). Proper scaling leads to better convergence, faster training, and more accurate models in SVM classification or regression tasks.

42) How will you do feature selection using Lasso Regression?

Lasso Regression (Least Absolute Shrinkage and Selection Operator) is a regularization technique that helps in feature selection by adding a penalty to the loss function, proportional to the absolute values of the model coefficients. This penalty encourages the model to shrink less important feature coefficients toward zero, effectively eliminating them from the model.

Here's how Lasso helps in feature selection:

1. **Fit a Lasso Model:** You start by training a Lasso regression model on your dataset. The L1 penalty term in the Lasso algorithm forces the coefficients of some features to become exactly zero.
2. **Evaluate the Coefficients:** After training, the coefficients of the features that are not important for the model will be shrunk to zero. These features can be safely removed from the model.
3. **Select Features:** Features with non-zero coefficients are considered significant and are retained. Features with zero coefficients are excluded from the model.
4. **Tune the Regularization Parameter (Alpha):** The strength of the penalty is controlled by the regularization parameter alpha. A larger value of alpha leads to more features being shrunk to zero, while a smaller value allows more features to remain in the model. You can perform cross-validation to find the optimal alpha value.

43) What is the difference between loss function and cost function?

In machine learning, a loss function measures the error for a single training example, indicating how well the model's prediction matches the actual label for that data point. On the other hand, a cost function is an aggregate measure, typically the average (or sum) of the loss function applied across the entire training dataset, providing a global evaluation of the model's performance. Essentially, while the loss function evaluates individual data points, the cost function evaluates the model's overall performance across all examples.

44) What is the difference between standard scaler and minmax scaler? What you will do if there is a categorical variable?

The Standard Scaler standardizes features by removing the mean and scaling to unit variance, i.e., it transforms the data to have a mean of 0 and a standard deviation of 1. This is useful when the data follows a normal distribution or when you want to account for outliers.

The Min-Max Scaler scales the features to a specified range, typically $[0, 1]$, by subtracting the minimum value of the feature and dividing by the range ($\text{max} - \text{min}$). This is useful when the features have different units or when we need to ensure all features are on the same scale.

For categorical variables, they should be encoded into numerical values before scaling. This can be done using techniques such as One-Hot Encoding (for nominal categories) or Label Encoding (for ordinal categories), depending on the nature of the variable.

45) What are some advantages and Disadvantages of regression models and tree-based models?

Regression models are efficient for simple, linear relationships, while tree-based models are more flexible and capable of capturing complex patterns but may be more prone to overfitting and computationally intensive.

46) What are some important hyperparameters for XGBOOST?

- `n_estimators`: The number of boosting rounds (trees) to build. More trees generally lead to better performance but increase the risk of overfitting.
- `learning_rate` (eta): Controls the contribution of each tree to the final prediction. Lower values make the model more robust but require more trees to converge.
- `max_depth`: The maximum depth of each tree. Deeper trees can capture more complex patterns, but they can also lead to overfitting.

47) Difference between R and Adjusted R Squared values in ML?

1. R-squared (R^2):

- What it is: It measures the proportion of the variance in the dependent variable that is explained by the independent variables in the model. It ranges from 0 to 1, where 1 means the model perfectly explains the variance in the data.
- Issue: As you add more features to a model, R^2 will always increase, even if the additional features are irrelevant or don't improve the model. This means that a model with more predictors might look better simply because it has more variables, even if those variables aren't truly useful.

2. Adjusted R-squared:

- What it is: It adjusts the R^2 value by taking into account the number of predictors in the model. It penalizes the addition of irrelevant predictors. This means that if you add features that don't improve the model, Adjusted R^2 will decrease.
- Why it matters: Adjusted R^2 gives a more realistic measure of how well the model fits the data, especially when comparing models with different numbers of predictors. It helps prevent overfitting by discouraging the addition of unnecessary features.

In short:

- R^2 tells you how well your model explains the variance but can be misleading with many predictors.
- Adjusted R^2 provides a more reliable metric by adjusting for the number of predictors in the model, making it a better choice when comparing models with different numbers of features.

Example:

- R^2 could be 0.95 (indicating 95% of variance is explained), but after adding irrelevant features, it might still increase to 0.96 even though the model doesn't improve much.
- Adjusted R^2 might stay at 0.95 or drop slightly if those extra features don't improve the model's ability to explain the data.

48) What are the different evaluation metrics for a regression model?

1. Mean Absolute Error (MAE)

MAE measures the average of the absolute differences between predicted and actual values. It tells you how much, on average, the predictions are off by, without considering whether the prediction is over or under the actual value.

Example:

If your house price prediction model gives the following predictions and actual values:

- Predicted: [200K, 300K, 400K]
- Actual: [210K, 290K, 410K]

The absolute errors are:

- $|200K - 210K| = 10K$
- $|300K - 290K| = 10K$
- $|400K - 410K| = 10K$

$$MAE = (10K + 10K + 10K) / 3 = 10K$$

This means, on average, the model's predictions are off by 10K.

2. Mean Squared Error (MSE)

MSE squares the difference between actual and predicted values, which penalizes larger errors more than smaller ones. It's sensitive to outliers, meaning that if you make a large error, it will have a bigger impact on the MSE.

Example:

Using the same predictions and actual values:

- Predicted: [200K, 300K, 400K]
- Actual: [210K, 290K, 410K]

The squared errors are:

- $(200K - 210K)^2 = 100K^2$
- $(300K - 290K)^2 = 100K^2$
- $(400K - 410K)^2 = 100K^2$

$$MSE = (100K^2 + 100K^2 + 100K^2) / 3 = 100K^2$$

The MSE is $100K^2$, and since squaring magnifies larger errors, it's more sensitive to extreme values.

3. Root Mean Squared Error (RMSE)

RMSE is the square root of MSE. It gives the error value in the same units as the target variable (e.g., if you're predicting prices, the RMSE will be in dollars, not squared dollars). It helps interpret the error more easily.

Example:

If $MSE = 100K^2$, then

$$RMSE = \sqrt{100K^2} = 10K$$

This means the model's average error is 10K.

4. R-squared (R^2)

R^2 tells you how much of the variation in the actual values can be explained by the model. It ranges from 0 to 1, where 1 means the model perfectly explains the variation in the target variable.

Example:

Suppose the R^2 of a model is 0.9, which means the model explains 90% of the variation in the actual values. If your model's R^2 is low (e.g., 0.1), it means the model doesn't explain much of the data's variation, and it's probably a poor fit.

5. Adjusted R-squared

Adjusted R^2 adjusts the R^2 score by penalizing for adding more predictors to the model. It's useful when comparing models with a different number of predictors, as adding irrelevant features can artificially inflate R^2 .

Example:

If you have a model with two features and another with five, Adjusted R^2 helps to determine whether the additional features actually improve the model or not. If the model with five features has a much lower Adjusted R^2 , it may indicate that those extra features are unnecessary.

6. Mean Absolute Percentage Error (MAPE)

MAPE measures the percentage difference between the predicted and actual values. It's a good metric when you want to understand error in relative terms, i.e., how big is the error compared to the actual value.

Example:

For the same house prices:

- Predicted: [200K, 300K, 400K]
- Actual: [210K, 290K, 410K]

The percentage errors are:

- $|(200K - 210K) / 210K| \times 100 = 4.76\%$
- $|(300K - 290K) / 290K| \times 100 = 3.45\%$
- $|(400K - 410K) / 410K| \times 100 = 2.44\%$

$MAPE = (4.76\% + 3.45\% + 2.44\%) / 3 = 3.55\%$

This means that, on average, the model's predictions are off by about 3.55%.

49) What are the different evaluation metrics for a classification model?

For evaluating classification models, key metrics include Accuracy, which measures the proportion of correct predictions but is unreliable for imbalanced datasets. Precision focuses on correctly predicted positives, useful when false positives are costly, while Recall emphasizes detecting all actual positives, critical when false negatives are serious. The F1 Score balances Precision and Recall, ideal for imbalanced datasets. ROC-AUC evaluates the model's ability to distinguish classes across thresholds, and Log Loss penalizes incorrect probabilistic predictions, measuring confidence in predictions. Choose metrics based on the problem—e.g., Precision for spam detection or Recall for medical diagnoses.

Example Usage:

- Use Accuracy for balanced datasets.
- Use Precision or Recall for imbalanced datasets, depending on whether false positives or false negatives are more critical.
- Use F1 Score when balancing Precision and Recall is important.
- Use ROC-AUC for threshold-independent evaluation and comparison.

50) List some of the drawbacks of a Linear model?

Linear models, while simple and interpretable, have several drawbacks. They assume a linear relationship between input features and the target variable, which may not hold in real-world scenarios, limiting their ability to capture complex patterns. They are sensitive to multicollinearity and outliers, which can distort predictions. Linear models also struggle with high-dimensional data, as irrelevant or highly correlated features can negatively impact performance. Additionally, they require feature scaling for optimal performance and cannot handle non-linear relationships unless transformed or combined with other techniques like polynomial features.

51) What do you mean by Bias variance tradeoff?

The **Bias-Variance Tradeoff** is a fundamental concept in machine learning that describes the balance between two sources of error in a model:

1. **Bias:** The error due to overly simplistic assumptions in the model. High bias leads to underfitting, where the model fails to capture the underlying patterns in the data.
2. **Variance:** The error due to sensitivity to small fluctuations in the training data. High variance leads to overfitting, where the model captures noise instead of the actual signal.

The tradeoff lies in finding the right balance: reducing bias usually increases variance and vice versa. A well-performing model minimizes both bias and variance to achieve optimal generalization on unseen data.

52) What is the difference between Type 1 and Type 2 error?

- **Type 1 Error (False Positive):** Rejecting the null hypothesis when it is actually true. For example, concluding that a new drug is effective when it actually isn't. This error focuses on mistakenly identifying an effect or relationship that doesn't exist.
- **Type 2 Error (False Negative):** Failing to reject the null hypothesis when it is actually false. For example, concluding that a new drug is not effective when it actually works. This error overlooks a real effect or relationship.

In short, **Type 1 error relates to detecting a false effect**, while **Type 2 error relates to missing a true effect**.

53) How can you determine which features are the most important in your model?

- **Feature Importance (Tree-Based Models):** Algorithms like Random Forest, XGBoost, and Gradient Boosting provide feature importance scores based on how much each feature reduces impurity (Gini or entropy) or impacts model performance.
- **Coefficients (Linear Models):** In linear models, such as Linear Regression or Logistic Regression, the absolute values of coefficients indicate feature importance, assuming the features are scaled.
- **Permutation Importance:** Randomly shuffle the values of each feature and measure the drop in model performance to determine its importance.
- **SHAP Values:** SHAP (SHapley Additive exPlanations) provides a more interpretable method to measure the impact of each feature on the predictions.
- **Recursive Feature Elimination (RFE):** Iteratively removes less important features while evaluating model performance to identify key features.
- **Domain Knowledge:** Combining model-based methods with domain expertise ensures important features are correctly interpreted.

54) How would you differentiate between Multilabel and MultiClass classification?

Multilabel Classification: In multilabel classification, each instance can belong to multiple classes simultaneously. For example, in a movie genre prediction system, a single movie can be classified as both "Action" and "Comedy." The output is typically represented as a binary vector where each position corresponds to a label.

Multiclass Classification: In multiclass classification, each instance belongs to only one class out of multiple possible classes. For example, in handwritten digit recognition, each image represents one digit (0–9). The output is typically a single label indicating the class.