ML Assignment Questions

**1. What is a parameter?**

ANS: In Machine Learning (ML), a parameter refers to a variable the model learns from the training data. These parameters define how the model makes predictions and are updated during training to minimize the error.

Types of Parameters in ML:

1. Model Parameters:
   * Learned from the training data.
   * Define the structure and function of the model.
   * Examples:
     + Weights (W) and biases (b) in neural networks.
     + Coefficients in linear regression.
     + Decision boundaries in SVM.
2. Hyperparameters:
   * Set before training begins (not learned from data).
   * Control how the model is trained.
   * Examples:
     + Learning rate (α) in gradient descent.
     + Number of hidden layers in a neural network.
     + Number of clusters (k) in k-means clustering.

**2. What is correlation? What does negative correlation mean?**

ANS: Correlation is a statistical measure that describes the relationship between two variables. It indicates how changes in one variable are associated with changes in another. Correlation is measured using the correlation coefficient (r), which ranges from -1 to 1:

* r = 1 → Perfect positive correlation (Both variables increase together).
* r = 0 → No correlation (No relationship between variables).
* r = -1 → Perfect negative correlation (One variable increases while the other decreases).

A **negative correlation** means that as one variable increases, the other decreases. The correlation coefficient (**r**) is between **-1 and 0**.

**Examples of Negative Correlation:**

* **Temperature & Hot Coffee Sales** → As temperature increases, hot coffee sales decrease.
* **Work Experience & Training Time** → More experience leads to less training time required.
* **Speed & Travel Time** → As speed increases, travel time decreases.

**3. Define Machine Learning. What are the main components in Machine Learning?**

ANS: Machine Learning is a branch of Artificial Intelligence (AI) that enables computers to learn from data and make decisions or predictions without being explicitly programmed. Instead of using predefined rules, ML models recognize patterns in data and improve their performance over time.

### **Main Components of Machine Learning:**

1. **Data**
   * The foundation of ML, used to train and test models.
   * Types: Structured (tables, databases) & Unstructured (images, text, audio).
2. **Features (Input Variables)**
   * Characteristics or attributes used for making predictions.
   * Example: In predicting house prices, features could be **area, location, number of rooms, etc.**
3. **Model**
   * The mathematical representation that learns from data.
   * Examples: Linear Regression, Decision Trees, Neural Networks.
4. **Training Process**
   * The phase where the model learns from labeled data using an algorithm.
   * Adjusts model parameters to minimize errors.
5. **Loss Function**
   * Measures how well the model’s predictions match actual results.
   * Example: Mean Squared Error (MSE) for regression problems.
6. **Optimization Algorithm**
   * Adjusts model parameters to minimize loss.
   * Example: **Gradient Descent** updates weights in neural networks.
7. **Evaluation & Testing**
   * Measures model performance using a separate dataset.
   * Metrics: Accuracy, Precision, Recall, RMSE (Root Mean Squared Error).
8. **Deployment & Prediction**
   * Using the trained model to make real-world predictions.

**4. How does loss value help in determining whether the model is good or not?**

ANS: The **loss value** is a key metric that helps determine how well a machine learning model is performing. It measures the difference between the model's predictions and the actual values. The goal of training a model is to **minimize** the loss value.

### **How Loss Value Determines Model Performance:**

1. **Low Loss Value → Good Model**
   * If the loss is low, the model's predictions are close to the actual values.
   * Example: In a **house price prediction model**, if the predicted price is ₹50 lakh and the actual price is ₹51 lakh, the loss is small, meaning the model is accurate.
2. **High Loss Value → Poor Model**
   * If the loss is high, the model’s predictions are far from the actual values.
   * Example: If the predicted price is ₹30 lakh and the actual price is ₹51 lakh, the model is performing poorly.

**5. What are continuous and categorical variables?**

ANS: In Machine Learning, data features (or variables) are broadly classified into **continuous** and **categorical** variables. Understanding their differences helps in selecting the right preprocessing techniques and models.

### **1. Continuous Variables**

A **continuous variable** can take an infinite number of values within a given range. These values are usually numerical and measurable.

**Examples:**

* **Height** (e.g., 5.6 ft, 6.1 ft)
* **Weight** (e.g., 70.5 kg, 85.3 kg)
* **Temperature** (e.g., 25.3°C, 30.8°C)
* **Salary** (e.g., ₹50,000, ₹75,250.50)

**How to Handle Continuous Variables in ML?**

* **Normalization (Min-Max Scaling)** → Scales values between 0 and 1.
* **Standardization (Z-Score Scaling)** → Converts values to have mean = 0 and standard deviation = 1.

### **2. Categorical Variables**

A **categorical variable** represents distinct groups or categories. These values do not have a meaningful numerical relationship.

**Types of Categorical Variables:**

1. **Nominal Variables** (No natural order)
   * **Example:**
     + Gender → Male, Female, Other
     + Colors → Red, Blue, Green
2. **Ordinal Variables** (Have a meaningful order/ranking)
   * **Example:**
     + Education Level → High School < Bachelor's < Master's < PhD
     + Customer Satisfaction → Poor < Average < Good < Excellent

**6. How do we handle categorical variables in Machine Learning? What are the common techniques?**

ANS: Handling categorical variables in machine learning is important because most machine learning models work with numerical data. There are several techniques to convert categorical data into a format suitable for modeling.

 Label Encoding

* Assigns a unique numeric value to each category.
* Example:
  + Colors: Red → 0, Blue → 1, Green → 2
* Works well when categories have an inherent order, but for unordered categories, it may introduce a false relationship.

 One-Hot Encoding

* Creates separate binary columns for each category.
* Example:
  + Colors: Red → (1,0,0), Blue → (0,1,0), Green → (0,0,1)
* Works well when the number of categories is small, but for high-cardinality features, it increases dimensionality.

 Ordinal Encoding

* Similar to label encoding but preserves order when present.
* Example:
  + Education Level: High School → 1, Bachelor's → 2, Master's → 3, PhD → 4
* Used when there is a meaningful ranking in the categories.

 Frequency Encoding

* Replaces categories with their occurrence count or frequency.
* Example:
  + If "Red" appears 50 times, "Blue" 30 times, and "Green" 20 times, they are replaced with 50, 30, and 20.
* Useful for high-cardinality categorical features.

 Target Encoding (Mean Encoding)

* Replaces each category with the mean of the target variable.
* Example:
  + If "Red" has an average sales value of 100, "Blue" has 150, and "Green" has 120, they are replaced accordingly.
* Used in supervised learning but can lead to data leakage.

 Binary Encoding

* Converts categories into binary numbers and represents them in multiple columns.
* Example:
  + Categories: A → 1 (001), B → 2 (010), C → 3 (011)
* Reduces dimensionality compared to one-hot encoding.

 Embedding Encoding

* Uses deep learning to map categories into continuous vector representations.
* Example:
  + Words in text data are converted into word embeddings.
* Useful for complex categorical data, such as NLP applications

**7. What do you mean by training and testing a dataset?**

ANS: Training and testing a dataset are essential steps in building and evaluating machine learning models.

1. Training a dataset
   * The training dataset is the portion of data used to train the machine learning model.
   * The model learns patterns, relationships, and features from this data to make predictions.
   * Example: If building a spam email classifier, the training dataset contains emails labeled as spam or not spam.
   * The model adjusts its parameters using algorithms like gradient descent to minimize errors.
2. Testing a dataset
   * The testing dataset is a separate portion of data used to evaluate the model's performance.
   * It helps determine how well the model generalizes to unseen data.
   * Example: If the spam classifier is tested on new emails, the model’s accuracy in predicting spam or not spam is measured.
   * Metrics like accuracy, precision, recall, and F1-score are used for evaluation.

A common practice is to split the dataset into:

* Training set (70-80%) → Used for model learning.
* Testing set (20-30%) → Used for evaluating the model’s performance

**8. What is sklearn.preprocessing?**

ANS: sklearn.preprocessing is a module in **Scikit-Learn** that provides tools to transform and standardize data for machine learning models. Many machine learning algorithms perform better when input data is properly scaled or encoded. The preprocessing module includes functions for feature scaling, encoding categorical variables, normalization, and handling missing values.

**9. What is a Test set?**

ANS: A **test set** is a portion of the dataset used to evaluate the performance of a trained machine learning model. It helps determine how well the model generalizes to new, unseen data.

### Characteristics of a Test Set

* It is **not used** during training.
* It simulates real-world data that the model will encounter in deployment.
* It helps measure accuracy, precision, recall, F1-score, and other performance metrics.

**10. How do we split data for model fitting (training and testing) in Python? How do you approach a Machine Learning problem?**

ANS: To split data into **training and testing sets**, we use train\_test\_split from **Scikit-Learn**. This ensures that we have separate data for model training and evaluation.

### How do you approach a Machine Learning problem?

A structured approach ensures an effective and accurate machine learning model.

**Step 1: Define the Problem**

* Identify the goal: Classification, Regression, Clustering, etc.
* Understand the business problem and expected outcome.

**Step 2: Collect and Explore Data**

* Gather data from sources (CSV, databases, APIs, etc.).
* Perform Exploratory Data Analysis (EDA):
  + Check missing values.
  + Identify data distribution, correlations, and patterns.

**Step 3: Preprocess and Prepare Data**

* Handle missing values using **imputation** (mean, median, mode).
* Encode categorical variables using **One-Hot Encoding** or **Label Encoding**.
* Normalize or standardize numerical features using **MinMaxScaler** or **StandardScaler**.
* Split data into **training** and **testing** sets (e.g., 80-20 split).

**Step 4: Select and Train a Model**

* Choose a model based on the problem type:
  + **Classification** → Logistic Regression, Random Forest, SVM
  + **Regression** → Linear Regression, Decision Tree, XGBoost
  + **Clustering** → K-Means, DBSCAN
* Train the model using fit() method.

**Step 5: Evaluate Model Performance**

* Use a test set to evaluate accuracy and performance using metrics like:
  + **Classification**: Accuracy, Precision, Recall, F1-score, ROC-AUC
  + **Regression**: Mean Squared Error (MSE), R² Score
* Perform **cross-validation** to check model consistency.

**11. Why do we have to perform EDA before fitting a model to the data?**

ANS: Exploratory Data Analysis (EDA) is a crucial step in the machine learning pipeline because it helps us **understand** and **prepare** the data before training a model. Skipping this step can lead to inaccurate models, poor performance, and misleading results.

### **Key Reasons for Performing EDA**

1. **Detecting Missing Values**
   * Helps identify if any data is missing and decide how to handle it (e.g., removal, imputation).
   * Missing values can negatively impact model training and accuracy.
2. **Identifying Outliers**
   * Outliers can **distort** model predictions, especially in regression problems.
   * EDA helps detect and decide whether to remove or transform them.
3. **Understanding Data Distribution**
   * Visualizing distributions (e.g., histograms, box plots) helps determine if features are skewed or normally distributed.
   * This affects scaling techniques and model selection.
4. **Detecting Data Imbalance**
   * In classification problems, imbalance in target classes can lead to biased models.
   * EDA helps decide if techniques like **oversampling**, **undersampling**, or **weighted loss functions** are needed.
5. **Feature Correlation Analysis**
   * Checking for **highly correlated features** prevents redundancy (multicollinearity).
   * Helps in feature selection and dimensionality reduction.
6. **Choosing the Right Encoding for Categorical Variables**
   * Determines whether **label encoding, one-hot encoding, or another method** is best suited.
7. **Checking Relationships Between Variables**
   * Scatter plots, correlation matrices, and statistical tests help understand relationships between features and the target variable.
8. **Validating Data Consistency**
   * Helps detect **incorrect or inconsistent values** (e.g., negative age values, duplicates).
   * Ensures data integrity before training.

**12. What is correlation?**

ANS: Correlation is a statistical measure that shows the relationship between two variables. It indicates how one variable changes in response to another.

### **Types of Correlation:**

1. **Positive Correlation** → Both variables increase or decrease together.
   * Example: More study hours → Higher exam scores.
2. **Negative Correlation** → One variable increases while the other decreases.
   * Example: More exercise → Lower body weight.
3. **No Correlation** → No relationship between variables.
   * Example: Shoe size and intelligence.

**13. What does negative correlation mean?**

ANS: Negative correlation means that as one variable increases, the other decreases. It indicates an **inverse relationship** between two variables. The correlation coefficient (r) for negative correlation lies between **-1 and 0**.

### **Example:**

* **More exercise → Lower body weight**
* **Higher product prices → Lower sales**
* **Increase in social media usage → Decrease in study time**

**14. How can you find correlation between variables in Python?**

ANS: You can find the correlation between variables in Python using statistical methods. The most common way is to use the correlation coefficient, which measures the strength and direction of the relationship between two variables.

The main types of correlation methods used are:

* Pearson correlation – Measures linear relationships between variables.
* Spearman correlation – Used for ranked (ordinal) data.
* Kendall correlation – Measures the strength of association between variables.

A correlation matrix can also be used to check relationships between multiple variables in a dataset. This helps in understanding how different features interact with each other.

For visual representation, a heatmap can be used to display correlation values in a structured format.

**15. What is causation? Explain difference between correlation and causation with an example.**

### ANS: What is Causation?

Causation means that one event **directly influences** another. If **A causes B**, it means that changes in A lead to changes in B.

### Difference Between Correlation and Causation

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Correlation** | **Causation** |
| **Definition** | A statistical relationship between two variables. | A cause-and-effect relationship where one variable directly affects another. |
| **Direction** | Does not imply one variable causes the other to change. | One variable directly influences the other. |
| **Example** | Ice cream sales and drowning rates increase in summer. | Smoking leads to lung cancer. |

### **Example**

* **Correlation (No Causation)**: People who carry lighters are more likely to develop lung cancer. But carrying a lighter **does not cause** lung cancer; smoking (a hidden factor) does.
* **Causation**: Smoking directly **causes** lung cancer, as proven by scientific research.

**16. What is an Optimizer? What are different types of optimizers? Explain each with an example**

ANS: An optimizer is an algorithm used in **machine learning** and **deep learning** to update model parameters (weights and biases) to minimize the **loss function** and improve model performance.

### Types of Optimizers

1. **Gradient Descent (GD)**
   * It calculates the gradient of the loss function and updates weights accordingly.
   * **Types:**
     + **Batch Gradient Descent** – Uses the entire dataset.
     + **Stochastic Gradient Descent (SGD)** – Uses one sample at a time.
     + **Mini-batch Gradient Descent** – Uses small batches of data.
   * **Example:** Used in **linear regression** and **logistic regression**.
2. **Momentum-based Optimizer**
   * Helps speed up training by considering past gradients.
   * **Example:** Used in **deep neural networks** to avoid local minima.
3. **Adaptive Gradient (AdaGrad)**
   * Adjusts the learning rate for each parameter individually.
   * Works well for sparse data but slows down over time.
   * **Example:** Used in **text data processing**.
4. **RMSprop (Root Mean Square Propagation)**
   * Uses moving averages of squared gradients to normalize updates.
   * Helps in stabilizing training and avoiding large updates.
   * **Example:** Works well in **recurrent neural networks (RNNs)**.
5. **Adam (Adaptive Moment Estimation)**
   * Combines **momentum** and **RMSprop** for adaptive learning rates.
   * One of the most popular optimizers in deep learning.
   * **Example:** Used in **image classification (CNNs) and NLP models**.
6. **AdamW (Adam with Weight Decay)**
   * Improves generalization by adding weight decay to Adam optimizer.
   * **Example:** Used in **transformer-based models** like **BERT**.

### Choosing the Right Optimizer

* **SGD** → Works well for simple models.
* **Adam** → Best for deep learning and general-purpose applications.
* **RMSprop** → Suitable for recurrent models (RNNs).

**17. What is sklearn.linear\_model ?**

ANS: sklearn.linear\_model is a module in **scikit-learn** that provides various linear models for **regression** and **classification** tasks. These models assume a **linear relationship** between input features and the target variable.

### Common Models in sklearn.linear\_model

1. **LinearRegression**
   * Used for simple and multiple **linear regression** problems.
   * Assumes a straight-line relationship between features and the target.
   * Suitable for predicting **continuous values** (e.g., house prices).
2. **LogisticRegression**
   * Used for **binary and multi-class classification**.
   * Outputs probabilities for class labels.
   * Commonly used for tasks like **spam detection** and **medical diagnosis**.
3. **Ridge Regression**
   * A linear regression model with **L2 regularization** to prevent overfitting.
   * Helps when features are highly correlated (multicollinearity).
4. **Lasso Regression**
   * Uses **L1 regularization**, which can shrink some coefficients to zero.
   * Useful for **feature selection** in high-dimensional data.
5. **ElasticNet**
   * A combination of **Ridge (L2) and Lasso (L1) regression**.
   * Used when some features are important, but others should be eliminated.
6. **SGDClassifier & SGDRegressor**
   * Uses **Stochastic Gradient Descent (SGD)** for classification and regression.
   * Suitable for **large-scale datasets** where traditional solvers are slow.

**18 What does model.fit() do? What arguments must be given?**

ANS: The fit() function in machine learning is used to **train a model** on a given dataset. It finds the optimal values for the model parameters (weights and biases) by minimizing the **loss function** using optimization techniques.

Arguments Required in model.fit(X, y)

1. X (Features/Independent Variables) – A dataset in the form of a NumPy array, Pandas DataFrame, or SciPy sparse matrix.
2. y (Target/Dependent Variable) – The true labels (for classification) or continuous values (for regression).

**19. What does model.predict() do? What arguments must be given?**

ANS: The predict() function is used in machine learning to **make predictions** on new or unseen data after the model has been trained. It takes the learned parameters (weights and biases) and applies them to the input data to generate predictions.

### Arguments Required in model.predict(X)

1. **X (Feature Matrix / Independent Variables)** – The dataset for which predictions are required. It should have the same number of features as the training data.

**20. What are continuous and categorical variables?**

ANS: In machine learning and statistics, variables are classified into **continuous** and **categorical** types based on the type of values they hold.

### **Continuous Variables**

* These are **numeric** variables that can take an **infinite number of values** within a given range.
* They are **measurable** and can have **decimal points**.

#### Examples:

* Height (e.g., 5.6 feet)
* Weight (e.g., 70.5 kg)
* Temperature (e.g., 36.8°C)
* Income (e.g., ₹50,000 per month)

#### Use in Machine Learning:

* Typically used in **regression** problems.
* Requires **normalization** or **standardization** for better performance.

### **2. Categorical Variables**

* These represent **groups or categories** and do **not have numerical meaning**.
* They can be **nominal** (no order) or **ordinal** (ordered categories).

#### Examples:

* **Nominal:** Gender (Male, Female), Color (Red, Blue, Green)
* **Ordinal:** Education Level (High School < Bachelor < Master)

#### Use in Machine Learning:

* Often encoded using **One-Hot Encoding** or **Label Encoding** for use in models.
* Used mainly in **classification** tasks.

**21. What is feature scaling? How does it help in Machine Learning?**

ANS: Feature scaling is the process of **standardizing or normalizing** the range of independent variables (features) in a dataset. It ensures that all features contribute equally to the model by bringing them to a similar scale.

### **Why is Feature Scaling Important in Machine Learning?**

1. **Improves Model Performance** – Algorithms that rely on distance calculations (e.g., KNN, SVM, Linear Regression) perform better when features are on the same scale.
2. **Speeds Up Convergence** – In gradient-based optimization (e.g., Neural Networks, Logistic Regression), scaling helps the model converge faster.
3. **Prevents Features with Large Values from Dominating** – Models that use weight coefficients (e.g., Linear Regression) can give undue importance to features with larger magnitudes.
4. **Improves Accuracy** – Proper scaling helps prevent biased results due to feature disparities.

**22. How do we perform scaling in Python?**

ANS: Feature scaling in Python is done using preprocessing techniques from libraries like Scikit-Learn. The two most commonly used methods are Normalization (Min-Max Scaling) and Standardization (Z-Score Scaling).

### **Normalization (Min-Max Scaling)**

This method rescales the values of a feature so that they fall within a fixed range, typically **0 to 1** or **-1 to 1**. It is useful when the data has varying ranges and does not follow a normal distribution.

**Where it is used:**

* When working with machine learning models that rely on distances, such as **K-Nearest Neighbors (KNN)** and **Neural Networks**.
* When the dataset contains features with different scales.

### **2. Standardization (Z-Score Scaling)**

This method transforms the data so that it has a **mean of 0 and a standard deviation of 1**. It ensures that features have similar distributions, regardless of their original scale.

**Where it is used:**

* When data is normally distributed.
* In algorithms that assume normally distributed data, such as **Logistic Regression, Support Vector Machines (SVM), and Principal Component Analysis (PCA).**

**23. What is sklearn.preprocessing?**

### ANS: **What is** sklearn.preprocessing**?**

sklearn.preprocessing is a module in **Scikit-Learn** that provides various techniques for **preprocessing and transforming data** before training a machine learning model. It helps improve model performance by ensuring that data is properly scaled, encoded, and formatted.

### **Key Functionalities of** sklearn.preprocessing

1. **Feature Scaling** – Adjusts numerical features to a common scale
   * **Normalization (Min-Max Scaling)**: Rescales values between 0 and 1
   * **Standardization (Z-Score Scaling)**: Converts values to have a mean of 0 and variance of 1
2. **Encoding Categorical Variables** – Converts categorical data into numerical format
   * **Label Encoding**: Assigns a unique integer to each category
   * **One-Hot Encoding**: Creates binary columns for each category
3. **Handling Missing Values** – Fills in missing values with appropriate techniques
   * **Mean, Median, or Most Frequent Value Imputation**
4. **Polynomial Feature Generation** – Expands features by creating polynomial terms
   * Useful for **polynomial regression and feature interaction analysis**
5. **Binarization** – Converts numerical values into binary (0 or 1)
   * Useful when distinguishing between two states, such as "Has Loan" (1) or "No Loan" (0)

**24. How do we split data for model fitting (training and testing) in Python?**

ANS: When building a machine learning model, **splitting the dataset** into **training** and **testing** sets is a crucial step. This ensures that the model is trained on one portion of the data and evaluated on another to measure its performance.

### **Common Data Splitting Ratios**

* **80% Training - 20% Testing** (Standard practice)
* **70% Training - 30% Testing** (For larger datasets)
* **60% Training - 20% Validation - 20% Testing** (When using a validation set for hyperparameter tuning)

**25. Explain data encoding?**

ANS: Data encoding is the process of converting categorical data (text or labels) into numerical form so that machine learning models can process it. Since most ML algorithms work with numbers rather than text, encoding categorical variables is a crucial preprocessing step.

### **Types of Data Encoding**

1. **Label Encoding**
   * Assigns a unique integer to each category.
   * Example:
     + ["Red", "Blue", "Green"] → [0, 1, 2]
   * **When to use:** Suitable for **ordinal data** (where order matters, e.g., "Low", "Medium", "High").
   * **Limitation:** Models may assume a numerical relationship between categories.
2. **One-Hot Encoding**
   * Creates a separate binary column (0 or 1) for each category.
   * Example:
     + "Red", "Blue", "Green" → [1, 0, 0], [0, 1, 0], [0, 0, 1]
   * **When to use:** Suitable for **nominal data** (where order does not matter, e.g., "Apple", "Orange", "Banana").
   * **Limitation:** Increases dimensionality if there are too many unique categories.
3. **Ordinal Encoding**
   * Similar to label encoding but follows a specific ranking/order.
   * Example: "Low" → 0, "Medium" → 1, "High" → 2
   * **When to use:** When categorical data has an inherent order.
4. **Binary Encoding**
   * Converts categories into binary form and then represents them as columns.
   * Example:
     + "Apple" → 001, "Orange" → 010, "Banana" → 011
   * **When to use:** Useful for reducing high-dimensional categorical data while avoiding one-hot encoding drawbacks.
5. **Frequency Encoding**
   * Replaces categories with their frequency in the dataset.
   * Example: If "Apple" appears 50 times and "Orange" appears 30 times, they are encoded as 50 and 30.
   * **When to use:** When the frequency of a category carries significance.