ann

**🔹 Basic Understanding**

1. **What is an Artificial Neural Network (ANN)?**  
   An ANN is a computational model inspired by the human brain, consisting of layers of interconnected neurons used for pattern recognition, classification, regression, etc.
2. **What are the main components of an ANN?**
   * Neurons (nodes)
   * Layers (input, hidden, output)
   * Weights and biases
   * Activation functions
3. **What is the difference between perceptron and multilayer perceptron?**
   * Perceptron: A single-layer neural network (only input and output).
   * Multilayer Perceptron (MLP): Has one or more hidden layers, allowing it to solve non-linear problems.

**🔹 Technical Concepts**

1. **What is forward propagation?**  
   It’s the process of passing inputs through the layers of the network to generate the output.
2. **What is backpropagation?**  
   A learning algorithm that adjusts the weights in the network by propagating the error backward from the output to the input layer.
3. **What is the activation function, and why is it used?**  
   It introduces non-linearity to the model, enabling the ANN to learn complex patterns. Examples: ReLU, Sigmoid, Tanh.
4. **What are weights and biases?**
   * Weights determine the importance of input features.
   * Biases allow the model to shift the activation function for better fitting.

**🔹 Training & Performance**

1. **What is overfitting in ANN?**  
   When the model learns the training data too well, including noise, and performs poorly on new/unseen data.
2. **How can you prevent overfitting in ANN?**
   * Use dropout
   * Early stopping
   * Regularization (L1/L2)
   * Cross-validation
3. **What is the role of the learning rate?**  
   It controls how much the weights are adjusted during training. A high learning rate might overshoot the minimum; a low one might be too slow.

**🔹 Practical and Code Related**

1. **Which libraries are commonly used to implement ANN in Python?**
   * TensorFlow
   * Keras
   * PyTorch
   * Scikit-learn (for simple MLP models)
2. **Can you explain the structure of the ANN you implemented in your practical?**  
   *(Be ready to describe the number of layers, neurons, activation functions used, and dataset.)*
3. **What loss function did you use and why?**
   * Classification: categorical\_crossentropy or binary\_crossentropy
   * Regression: mean\_squared\_error
4. **What optimizer did you use?**  
   Common ones: SGD, Adam, RMSprop.  
   *Adam is popular due to its efficiency and faster convergence.*
5. **What is an epoch and a batch?**
   * **Epoch:** One full pass through the entire training data.
   * **Batch:** A subset of the data used to train the model at one time.

### 2. Classifier and Regressor using ANN (CO II, III, IV)

**Q3.** What’s the difference between ANN for classification and regression?  
**A:** Classification predicts categories (e.g., spam or not), regression predicts continuous values (e.g., house prices).

**Q4.** What activation function is used in classification vs regression?  
**A:**

* Classification: ReLU in hidden layers, softmax or sigmoid in output.
* Regression: ReLU or linear in output.

**Q5.** What is a loss function and which ones are commonly used?  
**A:** Measures error.

* Classification: Binary/Categorical Crossentropy
* Regression: Mean Squared Error (MSE)

**Q6.** What is a confusion matrix?  
**A:** A table showing TP, TN, FP, FN used to evaluate classification accuracy.

**Q7.** Define precision, recall, F1-score.  
**A:**

* Precision = TP / (TP + FP)
* Recall = TP / (TP + FN)
* F1 = Harmonic mean of precision & recall

**Q8.** How do you visualize the training performance of an ANN?  
**A:** Using a **loss graph** or **accuracy graph** over epochs.

Navie bayes

Here are detailed **viva questions and answers** for the **Bayesian Classifier / Naive Bayes** topic in your ML practical:

**✅ Bayesian Classifier / Naive Bayes – Viva Questions**

**Q1. What is a Bayesian classifier?**  
**A:** It’s a probabilistic model based on Bayes’ Theorem that predicts the class of a sample using prior knowledge and evidence from the data.

**Q2. State Bayes’ Theorem.**  
**A:**

P(A∣B)=P(B∣A)⋅P(A)P(B)P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}

Where:

* P(A∣B)P(A|B): Posterior probability
* P(B∣A)P(B|A): Likelihood
* P(A)P(A): Prior
* P(B)P(B): Evidence

**Q3. What is Naive Bayes? Why is it “naive”?**  
**A:** Naive Bayes is a classification technique assuming all features are **independent** given the class label, which is rarely true—hence the term "naive".

**Q4. What are the types of Naive Bayes classifiers?**  
**A:**

* **Gaussian Naive Bayes**: For continuous data (assumes normal distribution)
* **Multinomial Naive Bayes**: For count data (e.g., text classification)
* **Bernoulli Naive Bayes**: For binary features (e.g., presence/absence)

**Q5. What are the advantages of Naive Bayes?**  
**A:**

* Fast and simple
* Performs well on high-dimensional data (e.g., spam detection)
* Works with small datasets

**Q6. What are the limitations?**  
**A:**

* Assumes feature independence (rare in real-world data)
* Poor performance if features are highly correlated

**Q7. How is probability calculated in Naive Bayes for a given class?**  
**A:**

P(C∣X)=P(X1∣C)⋅P(X2∣C)⋅...⋅P(Xn∣C)⋅P(C)P(X)P(C|X) = \frac{P(X\_1|C) \cdot P(X\_2|C) \cdot ... \cdot P(X\_n|C) \cdot P(C)}{P(X)}

**Q8. What is Laplace smoothing and why is it used?**  
**A:** It avoids zero probabilities for unseen features by adding a small constant (usually 1) to all counts.

**Q9. What metrics are used to evaluate a Naive Bayes classifier?**  
**A:**

* Accuracy
* Precision, Recall, F1-score
* Confusion Matrix

**Q10. Where is Naive Bayes commonly used?**  
**A:**

* Email spam detection
* Sentiment analysis
* Document classification
* Disease prediction

Let me know if you want **code + output** for a Naive Bayes classifier too!

Pac

Here are important **viva questions and answers** related to the **PAC (Probably Approximately Correct) learning** framework, which is a theoretical concept in machine learning.

**✅ PAC Learning – Viva Questions and Answers**

**Q1. What is PAC learning?**  
**A:** PAC stands for **Probably Approximately Correct** learning. It’s a theoretical framework introduced by **Leslie Valiant** to study the **learnability** of functions. A hypothesis class is PAC-learnable if, with high probability, it can learn a hypothesis that is approximately correct.

**Q2. What does "Probably Approximately Correct" mean?**  
**A:** It means:

* **Probably (1 - δ):** The learning algorithm returns a good hypothesis with high probability (confidence).
* **Approximately (ε):** The hypothesis has a small error (is close to the true function).

Formally, a learner returns a hypothesis with **error ≤ ε** with **probability ≥ 1 - δ**.

**Q3. What are ε (epsilon) and δ (delta) in PAC learning?**  
**A:**

* **ε (epsilon):** Error tolerance — how close the hypothesis should be to the target concept.
* **δ (delta):** Confidence — how likely the algorithm is to succeed. Smaller δ = higher confidence.

**Q4. What are the assumptions of PAC learning?**  
**A:**

* The training examples are drawn i.i.d. (independently and identically distributed).
* The hypothesis belongs to a known hypothesis class.
* There is a target concept to approximate.

**Q5. What is sample complexity in PAC learning?**  
**A:** It's the number of training examples needed to ensure the algorithm learns an approximately correct hypothesis with high probability.

For binary classification, sample complexity is:

m≥1ε(ln⁡∣H∣+ln⁡1δ)m ≥ \frac{1}{\varepsilon} \left( \ln |H| + \ln \frac{1}{\delta} \right)

Where:

* mm = number of samples
* ∣H∣|H| = size of hypothesis space
* ε\varepsilon, δ\delta as defined above

**Q6. Is Naive Bayes PAC-learnable?**  
**A:** Yes, under certain conditions (independence of features and finite hypothesis class), Naive Bayes can be PAC-learnable.

**Q7. What are the limitations of PAC learning?**  
**A:**

* Assumes access to a perfect hypothesis class
* May not work well with noisy or real-world data without adjustments
* Doesn't handle adversarial inputs

**Q8. What are the variations of PAC learning?**  
**A:**

* **Agnostic PAC learning:** Does not assume the existence of a perfect hypothesis
* **Robust PAC learning:** Accounts for adversarial noise
* **Online PAC learning:** Learns from a stream of data points

**Q9. How is PAC learning related to generalization?**  
**A:** PAC learning guarantees that the learned hypothesis **generalizes well** to unseen data with high probability.

**Q10. Why is PAC learning important in ML theory?**  
**A:** It provides **mathematical guarantees** for learning, helps analyze **feasibility and efficiency** of algorithms, and forms the **foundation of computational learning theory**.

Let me know if you'd like a summary slide, PDF, or a diagram explaining the PAC learning framework!

Decision tree

**Q1.** What is a Decision Tree?  
**A:** A tree-like structure where internal nodes represent features, branches represent conditions, and leaves represent outcomes.

**Q2.** How is it used for classification vs regression?  
**A:**

* Classification: predicts a category
* Regression: predicts a numeric value

**Q3.** What are splitting criteria in classification?  
**A:**

* Gini Impurity
* Entropy (Information Gain)

**Q4.** What is overfitting in decision trees?  
**A:** When the model is too deep and learns noise in training data. Use pruning or max\_depth to reduce it.

Here are **detailed viva questions and answers** for **Decision Tree Classification and Regression**, useful for your Machine Learning practicals:

**✅ Decision Tree – Viva Questions and Answers**

**📌 BASICS**

**Q1. What is a Decision Tree?**  
**A:** A Decision Tree is a supervised learning algorithm used for both **classification** and **regression** tasks. It splits the data into branches based on feature values to make predictions.

**Q2. What is the basic idea behind a decision tree?**  
**A:** The algorithm splits the dataset into subsets based on feature values to form a **tree-like structure**. Each internal node represents a feature, each branch a decision, and each leaf a predicted outcome.

**📌 TYPES**

**Q3. What is the difference between Decision Tree Classification and Regression?**  
**A:**

| **Feature** | **Classification** | **Regression** |
| --- | --- | --- |
| Output | Class Label (categorical) | Continuous Value |
| Splitting Criteria | Gini, Entropy | MSE, MAE |
| Example Task | Spam detection, diagnosis | Price prediction, forecasting |

**📌 ALGORITHMS & CRITERIA**

**Q4. What are common algorithms to build decision trees?**  
**A:**

* **ID3** – Uses Entropy & Information Gain
* **CART** – Uses Gini Index (Classification) or MSE (Regression)
* **C4.5 / C5.0** – Improved versions of ID3

**Q5. What is Gini Index?**  
**A:** A measure of impurity or diversity used in CART.

Gini=1−∑i=1Cpi2Gini = 1 - \sum\_{i=1}^{C} p\_i^2

Where pip\_i is the probability of class ii.

**Q6. What is Entropy and Information Gain?**  
**A:**

* **Entropy** measures disorder in the data.

Entropy=−∑pilog⁡2piEntropy = - \sum p\_i \log\_2 p\_i

* **Information Gain** = Reduction in entropy after a split.

**Q7. What is the criterion used in regression trees?**  
**A:**

* **Mean Squared Error (MSE)**
* **Mean Absolute Error (MAE)**

**📌 ADVANTAGES & LIMITATIONS**

**Q8. What are the advantages of Decision Trees?**  
**A:**

* Easy to interpret
* Handles both numerical and categorical data
* No feature scaling needed
* Works well with missing values

**Q9. What are the limitations?**  
**A:**

* Prone to **overfitting**
* Sensitive to data variations
* Biased with unbalanced datasets

**Q10. How to avoid overfitting in decision trees?**  
**A:**

* **Pruning** (Pre-pruning or Post-pruning)
* **Limiting tree depth**
* **Setting min samples per split**

**📌 EVALUATION**

**Q11. What metrics are used to evaluate a classification tree?**  
**A:**

* Accuracy
* Confusion Matrix
* Precision, Recall, F1-score

**Q12. What metrics are used for regression trees?**  
**A:**

* RMSE (Root Mean Squared Error)
* MAE (Mean Absolute Error)
* R² Score (Coefficient of determination)

**📌 APPLICATIONS**

**Q13. Where are decision trees used?**  
**A:**

* Credit scoring
* Medical diagnosis
* Price prediction
* Fraud detection
* Customer churn prediction

Svm

Here are **viva questions and answers** on **SVM (Support Vector Machine)** tailored for your Machine Learning practical:

**✅ SVM (Support Vector Machine) – Viva Questions and Answers**

**📌 BASICS**

**Q1. What is an SVM?**  
**A:** SVM (Support Vector Machine) is a supervised learning algorithm used for **classification** and **regression**. It finds the optimal hyperplane that separates data points of different classes with the **maximum margin**.

**Q2. What is a hyperplane in SVM?**  
**A:** A hyperplane is a decision boundary that separates different classes. In 2D, it’s a line; in 3D, it’s a plane; and in higher dimensions, it’s a generalization.

**Q3. What are support vectors?**  
**A:** Support vectors are the **data points closest to the hyperplane**. They determine the position and orientation of the hyperplane. The SVM aims to maximize the margin between these points and the hyperplane.

**📌 MARGINS & KERNELS**

**Q4. What is the margin in SVM?**  
**A:** The margin is the **distance between the hyperplane and the nearest support vectors** from both classes. SVM maximizes this margin to improve generalization.

**Q5. What is the difference between hard margin and soft margin SVM?**  
**A:**

* **Hard Margin:** No misclassifications allowed; only works if data is linearly separable.
* **Soft Margin:** Allows some misclassification; better for noisy or overlapping data.

**Q6. What is the kernel trick in SVM?**  
**A:** The kernel trick allows SVM to **transform data into higher dimensions** where it may become linearly separable. It avoids computing the transformation explicitly by using kernel functions.

**Q7. What are common kernel functions in SVM?**  
**A:**

* **Linear Kernel**: For linearly separable data
* **Polynomial Kernel**: For curved boundaries
* **RBF (Radial Basis Function)** / **Gaussian Kernel**: For complex boundaries
* **Sigmoid Kernel**: Similar to neural networks

**📌 APPLICATION & PERFORMANCE**

**Q8. What are the advantages of SVM?**  
**A:**

* Works well for high-dimensional data
* Effective when number of features > number of samples
* Robust to overfitting in many cases

**Q9. What are the limitations of SVM?**  
**A:**

* Not suitable for large datasets (high training time)
* Poor performance on overlapping classes
* Needs careful tuning of kernel parameters

**Q10. What are the important parameters in SVM?**  
**A:**

* **C (Regularization parameter)**: Controls trade-off between achieving low error and margin maximization
* **Gamma (for RBF kernel)**: Defines how far the influence of a single training example reaches
* **Kernel type**

**📌 EVALUATION**

**Q11. What metrics are used to evaluate SVM classification?**  
**A:**

* Accuracy
* Confusion Matrix
* Precision, Recall, F1 Score
* ROC-AUC curve

**📌 MISCELLANEOUS**

**Q12. Can SVM be used for regression?**  
**A:** Yes, **Support Vector Regression (SVR)** is a variant of SVM that predicts continuous values.

**Q13. What is the cost function of SVM?**  
**A:** It tries to minimize:

12∣∣w∣∣2+C∑i=1nξi\frac{1}{2} ||w||^2 + C \sum\_{i=1}^{n} \xi\_i

Where:

* ∣∣w∣∣||w||: margin
* ξi\xi\_i: slack variables (for soft margin)
* CC: regularization parameter

**Q14. What is overfitting in SVM and how do you prevent it?**  
**A:** Overfitting occurs when the model fits noise. Prevented by:

* Choosing the right **C** and **gamma** values
* Using **cross-validation**
* Reducing feature dimensionality

Let me know if you'd like a **Python example** using sklearn.svm.SVC or an **SVM loss graph** visualization!

K mean

Here are **detailed viva questions and answers** for **K-Means Clustering**, formatted for your **ML practicals**:

**✅ K-Means Clustering – Viva Questions and Answers**

**📌 BASICS**

**Q1. What is K-Means clustering?**  
**A:** K-Means is an **unsupervised learning algorithm** used to partition data into **K clusters**, where each data point belongs to the cluster with the nearest **centroid** (mean of the cluster).

**Q2. Is K-Means a supervised or unsupervised algorithm?**  
**A:** It is an **unsupervised algorithm**, because it works on unlabeled data and finds inherent patterns or groupings in the dataset.

**Q3. What is the goal of K-Means?**  
**A:** To minimize the **within-cluster sum of squares (WCSS)**, which is the sum of squared distances between each point and its assigned cluster centroid.

**📌 WORKING**

**Q4. What are the steps in the K-Means algorithm?**  
**A:**

1. Choose the number of clusters **K**
2. Randomly initialize **K centroids**
3. Assign each point to the **nearest centroid**
4. Recalculate the centroids of each cluster
5. Repeat steps 3–4 until centroids stop changing (convergence)

**Q5. How is the distance between points and centroids calculated?**  
**A:** Most commonly using **Euclidean distance**.

Distance=(x1−y1)2+(x2−y2)2+...\text{Distance} = \sqrt{(x\_1 - y\_1)^2 + (x\_2 - y\_2)^2 + ...}

**📌 PARAMETERS AND EVALUATION**

**Q6. How do you choose the value of K?**  
**A:**

* **Elbow Method**: Plot WCSS vs. K and look for the “elbow” point
* **Silhouette Score**: Measures how similar a point is to its own cluster vs. other clusters
* **Domain knowledge**

**Q7. What is inertia in K-Means?**  
**A:** Inertia is the **sum of squared distances** between data points and their cluster centroids. Lower inertia generally indicates better clustering.

**Q8. What is the cost function of K-Means?**  
**A:**

J=∑i=1k∑xj∈Si∣∣xj−μi∣∣2J = \sum\_{i=1}^{k} \sum\_{x\_j \in S\_i} ||x\_j - \mu\_i||^2

Where:

* SiS\_i = cluster
* μi\mu\_i = centroid of cluster
* xjx\_j = data points

**📌 LIMITATIONS**

**Q9. What are the disadvantages of K-Means?**  
**A:**

* Requires you to predefine **K**
* **Sensitive to initialization**
* Doesn’t work well with **non-spherical or overlapping clusters**
* Affected by **outliers**

**Q10. What are some ways to improve K-Means?**  
**A:**

* Use **K-Means++** for better centroid initialization
* Try **multiple runs** with different initial seeds
* Remove **outliers** beforehand
* Normalize/scale the data

**📌 APPLICATIONS**

**Q11. What are real-life applications of K-Means?**  
**A:**

* **Customer segmentation**
* **Image compression**
* **Market basket analysis**
* **Anomaly detection**

**Q12. What is K-Means++? How is it different?**  
**A:** K-Means++ improves centroid initialization by spreading out initial centroids. This reduces the chances of poor clustering and improves convergence speed.

**📌 MISCELLANEOUS**

**Q13. Is K-Means deterministic?**  
**A:** No, since centroid initialization is random (unless using K-Means++), results may vary across different runs.

**Q14. Can K-Means be used for non-numeric data?**  
**A:** Not directly. K-Means relies on **distance metrics**, so **categorical data must be encoded** (e.g., one-hot encoding or using K-Modes for categorical features).

**Q15. What is the time complexity of K-Means?**  
**A:**

* **O(n \* k \* i \* d)**  
  Where:
* **n** = number of data points
* **k** = number of clusters
* **i** = number of iterations
* **d** = number of features

Let me know if you’d like a **code example in Python**, an **Elbow Method plot**, or to cover **K-Medoids or DBSCAN** as well.

Knn

Here are **detailed viva questions and answers** for **K-Nearest Neighbors (KNN)** – perfect for your **Machine Learning practical preparation**:

**✅ KNN (K-Nearest Neighbors) – Viva Questions and Answers**

**📌 BASICS**

**Q1. What is KNN?**  
**A:** KNN (K-Nearest Neighbors) is a **supervised learning algorithm** used for **classification** and **regression**. It classifies a data point based on the **majority class among its 'K' nearest neighbors** in the training data.

**Q2. Is KNN supervised or unsupervised?**  
**A:** It is a **supervised algorithm** because it uses labeled data to make predictions.

**Q3. How does KNN work?**  
**A:**

1. Choose a value for **K** (number of neighbors).
2. Calculate the **distance** between the test point and all training points.
3. Select the **K nearest** training samples.
4. For classification: Assign the **most common class**.  
   For regression: Take the **mean/average value** of neighbors.

**📌 DISTANCE METRICS**

**Q4. What distance metrics are used in KNN?**  
**A:**

* **Euclidean distance** (most common)
* **Manhattan distance**
* **Minkowski distance**
* **Hamming distance** (for categorical variables)

**Q5. What is Euclidean distance?**  
**A:**

d(p,q)=∑i=1n(pi−qi)2d(p, q) = \sqrt{\sum\_{i=1}^{n} (p\_i - q\_i)^2}

Where p and q are two data points and n is the number of features.

**📌 HYPERPARAMETERS AND CHALLENGES**

**Q6. How do you choose the value of K?**  
**A:**

* Use **cross-validation**
* Plot **error rate vs. K**
* Avoid very small (overfitting) or very large K (underfitting)

**Q7. What happens when K = 1 or very high?**  
**A:**

* **K = 1**: Model becomes too sensitive (overfits).
* **K = large**: Model becomes too generalized (underfits).

**Q8. What are the limitations of KNN?**  
**A:**

* **Computationally expensive** (slow on large datasets)
* **Sensitive to irrelevant features and noise**
* Needs **feature scaling**
* Poor performance with **high dimensionality**

**📌 ADVANTAGES**

**Q9. What are the advantages of KNN?**  
**A:**

* Simple and intuitive
* No training phase (lazy learner)
* Works well with small datasets
* Non-parametric (no assumption about data distribution)

**Q10. Is KNN a parametric or non-parametric algorithm?**  
**A:** **Non-parametric**, as it does not make any assumptions about the underlying data distribution.

**📌 CLASSIFICATION VS REGRESSION**

**Q11. How does KNN work for regression?**  
**A:** For regression, KNN predicts the **average** of the values of the K nearest neighbors rather than voting on the class.

**Q12. Can KNN be used for both classification and regression?**  
**A:** Yes. It can solve both:

* **Classification**: Majority vote of neighbors
* **Regression**: Mean of neighbors' values

**📌 MODEL EVALUATION**

**Q13. What evaluation metrics are used for KNN?**

* **For classification:** Accuracy, Confusion Matrix, Precision, Recall, F1-Score
* **For regression:** MSE, RMSE, MAE, R² Score

**Q14. Why do we need to scale features in KNN?**  
**A:** Because distance metrics are **sensitive to the scale** of features. Without scaling, larger-scaled features can dominate the distance calculation.

**📌 MISCELLANEOUS**

**Q15. Is KNN a lazy learner or eager learner? Why?**  
**A:** **Lazy learner** – It **stores the entire training dataset** and performs computation **at prediction time**. It does not build a model beforehand.

**Q16. What are some real-world applications of KNN?**  
**A:**

* Recommender systems
* Handwriting recognition
* Fraud detection
* Medical diagnosis (e.g., classifying tumors)

Would you like a **Python implementation of KNN using sklearn**, or a **graph of error rate vs K**? I can also help you compare it with other classifiers like Decision Tree or SVM.

Cnn

Here are **detailed viva questions and answers** for **CNN (Convolutional Neural Networks)** – tailored for your **ML practical**:

## ✅ CNN (Convolutional Neural Network) – Viva Questions and Answers

### 📌 ****BASICS****

**Q1. What is a CNN?**  
**A:** A Convolutional Neural Network (CNN) is a type of **deep learning model** used mainly for **image classification, object detection, and computer vision tasks**. It uses **convolutional layers** to automatically learn spatial hierarchies of features from input images.

**Q2. How is CNN different from a regular ANN?**  
**A:**

* **ANN**: Fully connected, does not consider spatial structure.
* **CNN**: Uses local connections, parameter sharing, and pooling to **reduce parameters** and learn from **image pixels**.

**Q3. Why are CNNs good for image processing?**  
**A:**  
CNNs capture **spatial relationships** (e.g., edges, shapes) through convolution filters and **preserve spatial locality**, which makes them highly effective for image-related tasks.

### 📌 ****CNN LAYERS****

**Q4. What are the main layers in CNN?**  
**A:**

1. **Convolutional Layer**
2. **Activation Layer (usually ReLU)**
3. **Pooling Layer (Max or Average Pooling)**
4. **Fully Connected Layer**
5. **Output Layer (with Softmax or Sigmoid)**

**Q5. What is the convolution operation?**  
**A:** It is the process of sliding a **kernel/filter** over the input image and computing the **dot product** to extract features such as edges or textures.

**Q6. What is a kernel or filter in CNN?**  
**A:** A small matrix used in the convolution operation to detect features like **edges, corners, textures**, etc.

**Q7. What is the role of ReLU in CNN?**  
**A:** ReLU (Rectified Linear Unit) introduces **non-linearity** to the model and helps prevent vanishing gradients. It replaces all negative values with zero.

**Q8. What is pooling in CNN?**  
**A:** Pooling reduces the **spatial dimensions** (width, height) of the feature maps and makes the model **invariant to small translations**.  
Types:

* **Max pooling**: Takes the maximum value
* **Average pooling**: Takes the average value

### 📌 ****TRAINING AND PERFORMANCE****

**Q9. What is the loss function used in CNN classification?**  
**A:** Typically, **Categorical Crossentropy** for multi-class classification and **Binary Crossentropy** for binary classification.

**Q10. What optimizer is commonly used?**  
**A:**

* **Adam** (most common)
* **SGD** (Stochastic Gradient Descent)
* **RMSProp**

**Q11. What is overfitting in CNN and how to prevent it?**  
**A:** Overfitting happens when the model performs well on training data but poorly on test data.  
To prevent it:

* Use **Dropout layers**
* Apply **Data Augmentation**
* Use **Early stopping**
* **Reduce model complexity**

**Q12. What is dropout?**  
**A:** Dropout randomly deactivates some neurons during training to prevent overfitting and improve generalization.

**Q13. What does a fully connected (dense) layer do?**  
**A:** It connects every neuron in one layer to every neuron in the next. Used at the **end of CNN** to combine extracted features and perform classification.

### 📌 ****APPLICATIONS & MISC****

**Q14. Where are CNNs used?**  
**A:**

* Image classification (e.g., MNIST, CIFAR)
* Facial recognition
* Object detection
* Medical imaging
* Self-driving cars
* Gesture recognition

**Q15. Why do we use flattening in CNN?**  
**A:** Flattening converts the 2D feature maps into a **1D vector** so it can be fed into a **fully connected layer** for final classification.

**Q16. What dataset is commonly used for CNN practice?**  
**A:**

* **MNIST** (handwritten digits)
* **Fashion-MNIST**
* **CIFAR-10**
* **ImageNet**

**Q17. What is the difference between Softmax and Sigmoid?**  
**A:**

* **Sigmoid**: Used for binary classification; outputs a value between 0 and 1.
* **Softmax**: Used for multi-class classification; outputs probability distribution across multiple classes.

**Q18. How do you visualize CNN filters or outputs?**  
**A:** You can use tools like **matplotlib** or **TensorBoard** to visualize:

* Feature maps (activation maps)
* Convolution filters
* Class probabilities

Let me know if you want:

* **CNN code for MNIST using TensorFlow/Keras**
* **Loss & accuracy plot**
* **Confusion matrix visualization**

I can also generate a **ready-made Jupyter Notebook** for your practical.