Interview preparation for Samsung

Q1: What is the Software Development Life Cycle (SDLC)?

Answer: SDLC is a structured process that guides software development through different phases, ensuring quality and efficiency.

The main phases are:

- Requirement Analysis – Understanding what the client needs.
- Design – Planning architecture and UI/UX.
- Implementation (Coding) – Writing the actual code.
- Testing – Identifying and fixing bugs.
- Deployment – Releasing the software for use.
- Maintenance – Updating and fixing issues post-release.

2. What are the different Git commands you use?

Answer: Some commonly used Git commands are:

git init – Initialize a new Git repository.

git clone <repo> – Clone an existing repository.

git add . – Add all changes to the staging area.

git commit -m "message" – Commit changes with a message.

git push origin main – Push local changes to a remote repository.

git pull origin main – Fetch the latest updates from the remote repository.

git branch <branch_name> – Create a new branch.

git checkout <branch_name> – Switch between branches.

git merge <branch_name> – Merge a branch into the main branch.

git log – View commit history.

3. What are APIs, and how do you work with them?

Answer: An API (Application Programming Interface) allows different software applications to communicate with each other.

REST API (Representational State Transfer) uses HTTP methods: GET, POST, PUT, DELETE.

Python Hands-On Questions

Q4: What are Python’s key features?

Answer:

- Easy to learn – Simple syntax and readability.
- Interpreted language – No need for compilation.
- Dynamically typed – No need to declare variable types.
- Extensive libraries – Includes NumPy, Pandas, TensorFlow, PyTorch, etc.
- Object-Oriented – Supports OOP principles.
- High scalability – Suitable for small scripts to large applications.

Q5: How does Python handle memory management?

Answer:

- Python uses automatic memory management via:
- Reference Counting – Objects are deallocated when their reference count reaches zero.
- Garbage Collection – Detects and removes unused objects.
- Memory Pools – Uses PyMalloc for efficient memory allocation.

3. Deep Learning & ML Frameworks

Q6: What are some commonly used ML/DL frameworks?

Answer:

- TensorFlow & Keras – Used for deep learning models.
- PyTorch – Popular in research for dynamic computation graphs.
- Scikit-Learn – Used for classical machine learning models.
- Hugging Face – Best for NLP tasks with Transformer models.
- NumPy & Pandas – Essential for data manipulation.

A. Deep Learning Fundamentals

Q1: What is Deep Learning, and how does it differ from Machine Learning?

Answer: Deep Learning is a subset of ML that uses neural networks with multiple layers to learn from data. Unlike ML, which relies on feature engineering, Deep Learning automatically extracts patterns using layers of neurons.

Q2: What are the different types of neural networks?

Answer:

- Feedforward Neural Networks (FNN) – Basic neural network with input, hidden, and output layers.
- Convolutional Neural Networks (CNN) – Best for image processing.
- Recurrent Neural Networks (RNN) – Best for sequential data like speech and text.
- Transformer Networks – Used in NLP (e.g., BERT, GPT).

Q3: What is the architecture of a basic artificial neural network (ANN)?

Answer: An ANN consists of an input layer, one or more hidden layers, and an output layer. Each layer contains neurons connected by weighted edges, which are updated using backpropagation.

Q4: What is Backpropagation, and how does it work?

Answer: Backpropagation is an optimization algorithm that helps neural networks learn by adjusting weights using gradient descent. It calculates the error in predictions, computes gradients, and updates weights to minimize loss.

Q5: What is the role of activation functions in neural networks?

Answer: Activation functions introduce non-linearity, allowing neural networks to learn complex patterns. Examples include ReLU, Sigmoid, and Tanh.

Q6: What is the difference between a shallow neural network and a deep neural network?

Answer: A shallow neural network has fewer layers (1-2 hidden layers), while a deep neural network has multiple layers (deep architectures improve feature extraction and learning).

Q7: Why do we use non-linear activation functions like ReLU, Sigmoid, or Tanh?

Answer: Non-linear activation functions enable networks to model complex relationships in data. ReLU avoids vanishing gradients, Sigmoid is useful for probability outputs, and Tanh centers data around zero.

Q8: What is the vanishing gradient problem, and how can it be resolved?

Answer: It occurs when gradients become too small during backpropagation, preventing deep networks from learning effectively. Solutions include using ReLU activation, batch normalization, and residual connections (ResNet).

Q9: What is the exploding gradient problem, and how do you fix it?

Answer: It happens when gradients become too large, leading to unstable training. Solutions include gradient clipping, proper weight initialization, and normalization techniques.

Q10: What are dropout and batch normalization, and why are they used?

Answer: Dropout prevents overfitting by randomly disabling neurons, while batch normalization stabilizes learning by normalizing activations.

Q11: What is weight initialization, and why is it important in Deep Learning?

Answer: Proper weight initialization helps prevent vanishing/exploding gradients. Methods include Xavier initialization and He initialization.

Q12: What are some common loss functions used in Deep Learning?

Answer:

- Mean Squared Error (MSE) – Regression tasks.
- Cross-Entropy Loss – Classification tasks.
- Huber Loss – Robust regression against outliers.

Q13: What is an autoencoder, and how does it work?

Answer: An autoencoder is an unsupervised neural network that compresses input into a lower-dimensional representation (encoder) and reconstructs it back (decoder).

Q14: What is the difference between training, validation, and test datasets?

Answer:

- Training Set: Used to train the model.
- Validation Set: Used for hyperparameter tuning.
- Test Set: Used to evaluate final performance.

Q15: What are attention mechanisms in deep learning, and why are they useful?

Answer: Attention mechanisms allow models to focus on important parts of input data, significantly improving NLP tasks like machine translation (e.g., Transformer models).

B. Convolutional Neural Networks (CNNs)

Q16: What is a Convolutional Neural Network (CNN)?

Answer: A CNN is a type of deep learning model specifically designed for processing structured grid data, such as images. It utilizes convolutional layers to extract hierarchical features from input images.

Q17: How do convolutional layers work in CNNs?

Answer: Convolutional layers apply filters (kernels) to input images, capturing spatial patterns such as edges, textures, and shapes. This reduces the need for manual feature extraction.

Q18: What is the role of pooling layers in CNNs?

Answer: Pooling layers reduce the spatial dimensions of feature maps, decreasing computation and improving model generalization. They help extract dominant features while reducing noise.

Q19: What are max pooling and average pooling?

Answer:

Max Pooling: Selects the maximum value from a region of the feature map, preserving prominent features.

Average Pooling: Computes the average value from a region, smoothing the feature map and reducing sensitivity to noise.

Q20: What is the difference between a fully connected layer and a convolutional layer?

Answer:

Fully Connected Layer: Every neuron is connected to all neurons in the previous layer, capturing global patterns but requiring many parameters.

Convolutional Layer: Uses localized filters to extract spatial features, making it more efficient for image tasks.

Q21: What are some popular CNN architectures?

Answer:

- LeNet-5 – Early CNN model for handwritten digit recognition.
- AlexNet – Improved CNN architecture with deeper layers, won ImageNet competition.
- VGGNet – Uses multiple 3x3 convolutional layers for deeper feature extraction.
- ResNet – Introduces residual connections to solve vanishing gradient issues.
- EfficientNet – Optimized for better performance with fewer parameters.

Q22: How does transfer learning work in CNNs?

Answer: Transfer learning leverages pre-trained CNN models on large datasets (e.g., ImageNet) and fine-tunes them on smaller datasets to improve performance while reducing training time.

Q23: What is data augmentation, and how does it help in CNN training?

Answer: Data augmentation artificially expands the training dataset by applying transformations such as rotation, flipping, cropping, and brightness adjustment, helping improve model robustness and prevent overfitting.

Q24: What are dilated convolutions, and how do they work?

Answer: Dilated convolutions expand the receptive field without increasing computation by introducing gaps between kernel pixels, making them useful for semantic segmentation tasks.

Q25: What are the challenges of using CNNs for large-scale image classification?

Answer:

- Computational Cost: CNNs require powerful hardware (GPUs/TPUs) for training.
- Large Datasets Needed: CNNs perform best with large labeled datasets.
- Overfitting: Can memorize training data if insufficient regularization is applied.
- Interpretability: Hard to understand decision-making processes in deep CNNs.

C. Recurrent Neural Networks (RNNs) and NLP

Q26: What is a Recurrent Neural Network (RNN)?

Answer: A Recurrent Neural Network (RNN) is a type of neural network designed for sequential data processing. Unlike feedforward networks, RNNs have loops that allow information to persist, making them useful for tasks like speech recognition and time series forecasting.

Q27: How do RNNs handle sequential data?

Answer: RNNs process sequential data by maintaining a hidden state that captures information from previous time steps. The hidden state is updated at each time step, allowing the network to retain memory of past inputs.

Q28: What is the problem of vanishing gradients in RNNs?

Answer: The vanishing gradient problem occurs when gradients shrink exponentially as they are propagated backward through time, making it difficult for RNNs to learn long-term dependencies. Solutions include using architectures like LSTMs and GRUs.

Q29: What are LSTMs and GRUs, and how do they improve RNNs?

Answer:

LSTMs (Long Short-Term Memory Networks): Introduce memory cells and gating mechanisms to store long-term dependencies.

GRUs (Gated Recurrent Units): Similar to LSTMs but with fewer parameters, making them computationally more efficient.

Q30: What is an embedding layer in NLP models?

Answer: An embedding layer maps words to dense vector representations, capturing semantic meanings. It helps transform categorical text data into continuous-valued vectors that neural networks can process efficiently.

Q31: How do transformer models like BERT and GPT differ from RNNs?

Answer:

RNNs: Process sequences sequentially, which can be slow for long texts.

Transformers (e.g., BERT, GPT): Use self-attention mechanisms to process entire sequences in parallel, making them more efficient and scalable.

Q32: What is the difference between sequence-to-sequence (Seq2Seq) models and attention-based models?

Answer:

- Seq2Seq Models: Use an encoder-decoder architecture, where the encoder processes the input, and the decoder generates output step by step.
- Attention-Based Models: Enhance Seq2Seq models by allowing the decoder to focus on relevant parts of the input sequence at each step, improving translation and summarization tasks.

Q33: What are pre-trained NLP models, and why are they important?

Answer: Pre-trained NLP models (e.g., BERT, GPT, T5) are trained on large datasets and can be fine-tuned for specific tasks, reducing training time and improving performance on NLP applications.

Q34: What is word embedding, and how do models like Word2Vec and GloVe work?

Answer:

- Word2Vec: Learns word representations by predicting word context (CBOW) or predicting words given a context (Skip-gram).
- GloVe: Generates word embeddings by analyzing word co-occurrence statistics in a corpus, capturing both local and global context.

E. ML Frameworks (TensorFlow, PyTorch, Scikit-Learn, Hugging Face)

Q42: What are some commonly used deep learning frameworks?

Answer: Popular frameworks include TensorFlow, PyTorch, Keras, Hugging Face Transformers, MXNet, and JAX.

Q43: What is the difference between TensorFlow and PyTorch?

Answer:

- TensorFlow: Uses static computation graphs (Graph execution), suitable for production deployment.
- PyTorch: Uses dynamic computation graphs (Eager execution), preferred for research due to flexibility.

Q44: How do you define a simple neural network in TensorFlow?

```
import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

model = Sequential([

  Dense(64, activation='relu', input_shape=(10,)),

  Dense(32, activation='relu'),

  Dense(1, activation='sigmoid')

])

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model.fit(X_train, y_train, epochs=10, batch_size=32)
```

Q45: What is eager execution in TensorFlow?

Answer: Eager execution allows operations to be evaluated immediately, making TensorFlow more intuitive and interactive compared to static computation graphs.

Q46: How does PyTorch's dynamic computation graph work?

Answer: PyTorch creates computation graphs dynamically at runtime, making it easier to debug and modify model architectures without explicitly defining the entire graph beforehand.

Q47: What are the advantages of using PyTorch over TensorFlow?

Answer:

- Easier debugging: Uses Pythonic execution.
- Dynamic computation graphs: More flexible for research.
- Better integration with NumPy: Seamless tensor operations.

Q50: What is the role of Scikit-learn in Machine Learning?

Answer: Scikit-learn is a library for traditional ML algorithms such as regression, classification, and clustering. It provides tools for preprocessing, model selection, and evaluation.

F. Training, Optimization, and Model Evaluation

Q53: What are the different optimization algorithms used in Deep Learning?

Answer: Common optimization algorithms include:

- Stochastic Gradient Descent (SGD) – Updates weights using small random batches.
- Adam (Adaptive Moment Estimation) – Combines momentum and adaptive learning rates.
- RMSprop (Root Mean Square Propagation) – Adjusts learning rates based on recent gradients.
- Adagrad & Adadelta – Scale learning rates dynamically per parameter.

Q54: How does the Adam optimizer differ from SGD?

Answer:

- SGD updates weights using a fixed learning rate, making it slower but more stable.
- Adam adapts learning rates for each parameter, converging faster and handling sparse gradients efficiently.

Q55: What is the learning rate, and how does it impact model training?

Answer: The learning rate determines how much model weights are updated in each step.

- High learning rate: Faster convergence but risks overshooting optimal values.
- Low learning rate: More stable but slow convergence.
- Adaptive learning rate: Adjusted dynamically to improve training efficiency

Q56: What is early stopping, and how does it prevent overfitting?

Answer: Early stopping monitors validation loss and stops training when the loss starts increasing, preventing the model from overfitting to training data.

Q57: What are some methods to improve model generalization?

Answer:

- Regularization techniques: L1/L2 (Ridge/Lasso) regularization, dropout.
- Data augmentation: Introduce variations in training data.
- Cross-validation: Improve validation set performance.
- Reducing model complexity: Avoid overly deep or wide networks.

Q58: What is gradient clipping, and how does it help in training stability?

Answer: Gradient clipping restricts the maximum gradient value to prevent exploding gradients in deep networks, improving stability during training.

Q59: How do you use k-fold cross-validation in deep learning?

Answer: K-fold cross-validation splits the dataset into K subsets, training on K-1 and testing on the remaining fold iteratively. This ensures robust evaluation.

Q60: What is hyperparameter tuning, and what are some common tuning techniques?

Answer: Hyperparameter tuning optimizes model parameters such as learning rate, batch size, and dropout rate. Techniques include:

- Grid Search: Exhaustive search over parameter combinations.
- Random Search: Randomly samples parameters within a range.
- Bayesian Optimization: Uses probabilistic models to find optimal parameters efficiently.

G. Working with Pre-trained Models & Transfer Learning

Q61: What is transfer learning, and why is it useful?

Answer: Transfer learning is a technique where a model trained on one task is adapted for a different but related task. It is useful because it reduces training time, requires fewer labeled data, and improves model performance.

Q62: How do you fine-tune a pre-trained model like ResNet?

Answer: Fine-tuning a ResNet model involves freezing the initial layers (to retain learned features) and retraining the last layers with new data.

Q63: What are some examples of pre-trained models used in Computer Vision?

Answer:

- ResNet (Residual Networks) – Used for image classification.
- VGGNet (Visual Geometry Group) – Deep CNN with uniform layer structures.
- EfficientNet – Optimized for accuracy and efficiency.
- YOLO (You Only Look Once) – Used for real-time object detection.
- Mask R-CNN – Used for instance segmentation.

Q65: How do you handle domain adaptation when using pre-trained models?

Answer: Domain adaptation strategies include:

- Feature extraction: Using the pre-trained model as a fixed feature extractor.
- Fine-tuning: Training only certain layers of the model on new domain data.
- Domain-specific pre-training: Adapting the model with additional training on a domain-specific dataset.

Q66: What are some challenges in working with pre-trained deep learning models?

Answer:

- Domain mismatch: Pre-trained models may not generalize well to new datasets.
- Computational cost: Fine-tuning large models requires high computing power.
- Overfitting: If not tuned properly, the model may overfit the new dataset.
- Bias in data: Pre-trained models inherit biases from the original training dataset.

TensorFlow

TensorFlow is an open-source machine learning framework developed by Google. It is widely used for deep learning applications, providing tools for building, training, and deploying neural networks. TensorFlow supports GPUs and TPUs for high-performance computations and includes TensorFlow Lite for mobile deployment.

Pandas

Pandas is a powerful Python library for data manipulation and analysis. It provides data structures like Series (1D) and DataFrame (2D) for handling structured data efficiently. It is commonly used for tasks like data cleaning, transformation, and analysis.

NumPy

NumPy (Numerical Python) is a fundamental library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions for performing operations on these arrays. It is heavily used in scientific computing and machine learning.

Keras

Keras is a high-level deep learning API built on top of TensorFlow. It allows for easy and rapid development of neural networks, supporting various architectures such as CNNs, RNNs, and Transformers. Keras is known for its simplicity and modularity.

Scikit-Learn

Scikit-Learn is a widely used machine learning library that provides efficient implementations of various ML algorithms, including classification, regression, clustering, and dimensionality reduction. It also offers tools for preprocessing, model selection, and evaluation.

PyTorch

PyTorch is an open-source deep learning framework developed by Facebook. It is popular for its dynamic computation graph and ease of use in research and production. PyTorch is widely used in NLP, computer vision, and reinforcement learning applications.

Hugging Face

Hugging Face provides a vast ecosystem for natural language processing (NLP) and deep learning, offering pre-trained transformer models such as BERT, GPT, and T5. The transformers library simplifies the process of training and fine-tuning models for various NLP tasks.

SQL

SQL (Structured Query Language) is used to manage and manipulate relational databases. It allows users to perform operations like querying, inserting, updating, and deleting data from databases. SQL is essential for data engineering and analytics.

Matplotlib

Matplotlib is a plotting library in Python used for data visualization. It provides functions to create static, animated, and interactive plots, including line graphs, bar charts, histograms, and scatter plots. It is often used with Pandas and NumPy for exploratory data analysis.

For project 1: wine quality

1. **Why are we using LabelEncoder for the target variable?**

   The LabelEncoder converts categorical labels into numerical values (e.g., "low", "high" salary → 0, 1), which is necessary for binary classification using a neural network.

2. **Why are categorical variables one-hot encoded?**
   One-hot encoding transforms categorical variables into a numerical format that neural networks can process. Each unique category becomes a separate column with binary values (0 or 1). This prevents the model from interpreting categorical values as ordinal (e.g., treating "Manager" > "Employee" based on label encoding).
3. **What does StandardScaler do?**
   StandardScaler normalizes numerical features by transforming them to have a mean of 0 and a standard deviation of 1, ensuring that all features contribute equally to the model's training.
4. **Why use ColumnTransformer?**
   ColumnTransformer applies transformations separately to different types of features. It standardizes numerical features while one-hot encoding categorical ones, making data preprocessing more efficient.
5. **What does Dropout do in the neural network?**
   Dropout helps prevent overfitting by randomly setting a fraction of neurons to zero during training. This forces the model to learn more generalizable patterns instead of memorizing training data.
6. **Why is binary_crossentropy used as the loss function?**
   Since this is a binary classification problem (salary prediction with two classes), binary_crossentropy is the appropriate loss function. It helps optimize the probability output from the sigmoid activation function.

7. **What is the purpose of the Adam optimizer with a learning rate of 0.0001?**
   Adam is an adaptive optimizer that adjusts learning rates for each parameter dynamically. A learning rate of 0.0001 is chosen to ensure stable convergence without drastic weight updates.
8. **What does batch_size=200 mean?**
   This means that in each training iteration (epoch), the model processes 200 samples at a time before updating weights. A larger batch size can lead to more stable gradient updates.
9. **Why is validation_split=0.2 used in model.fit?**
   This reserves 20% of the training data for validation, allowing the model to be evaluated on unseen data during training. It helps monitor overfitting and generalization.
10. **What is Logistic Regression?**
    Logistic Regression is a supervised learning algorithm used for binary classification (0 or 1). Despite its name, it is a classification algorithm, not a regression algorithm. It predicts the probability that an instance belongs to a particular class.
11. **What is Linear Regression?**
    Linear Regression is a supervised learning algorithm used for predicting continuous values. It finds the best-fit line that minimizes the difference between actual and predicted values.

    Logistic Regression

    - Type: Classification (Binary)
    - Output Probability (0 or 1)
    - Activation: Sigmoid Function

    Linear Regression

    - Type: Regression (Continuous)
    - Output: Continuous Value (y)
    - Activation: No Activation (Linear)
    - Loss Function: Mean Squared Error (MSE)
12. **What is a Decision Tree Classifier?**
    A Decision Tree Classifier is a supervised learning algorithm used for classification tasks (categorizing data into distinct classes). It works by splitting the dataset into smaller subsets based on feature conditions, creating a tree-like structure where each internal node represents a decision rule, and each leaf node represents a class label.

    How It Works?

- The algorithm selects the best feature to split the dataset based on Gini Impurity or Entropy (Information Gain).
- It recursively splits the dataset until all leaf nodes are pure (i.e., contain only one class) or meet stopping criteria.

- The final decision is made based on the majority class in the leaf node.

13. What is a Decision Tree Regressor?

A Decision Tree Regressor is a supervised learning algorithm used for regression tasks (predicting continuous values). Instead of classifying data into categories, it predicts numerical values by splitting data into smaller regions and calculating the average of the target values in each leaf node.

How It Works?

- The algorithm selects the best feature to split the dataset based on Mean Squared Error (MSE) or Mean Absolute Error (MAE).
- It recursively splits the dataset, forming a tree structure where each leaf node represents the predicted value.
- The final prediction is the average of values in the corresponding leaf node.

14. What is a Random Forest Classifier?

A Random Forest Classifier is an ensemble learning method that builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting. It uses bagging (Bootstrap Aggregating) to train each tree on a different subset of the data.
How It Works?

- Randomly selects subsets of data and trains decision trees.
- Each tree gives a classification output (vote).
- The final prediction is determined by majority voting from all trees.

15. What is a Random Forest Regressor?

A Random Forest Regressor is similar to the classifier but is used for regression tasks (predicting continuous values). It averages the predictions from multiple decision trees.

16. What is Gradient Boosting?

Gradient Boosting Classifier is an ensemble method that builds decision trees sequentially, where each tree corrects the mistakes of the previous one using gradient descent. Unlike Random Forest, it focuses on improving weak learners rather than training independently.

17. What is Gradient Boosting Regressor?

A Gradient Boosting Regressor is used for continuous value predictions. It sequentially improves weak models by minimizing errors through gradient descent.

18. What is Support Vector Classifier (SVC)?

Support Vector Classifier (SVC) is a supervised learning model that finds the optimal hyperplane that maximizes the margin between different classes. It can use different kernels to handle non-linear data.

Algorithm	Type	Key Concept	Works Best For
RandomForestClassifier	Classification	Multiple decision trees (Bagging)	Large datasets with categorical features
RandomForestRegressor	Regression	Averaging multiple decision trees	Predicting continuous values
GradientBoostingClassifier	Classification	Boosting weak learners sequentially	Small to medium datasets with complex patterns
GradientBoostingRegressor	Regression	Boosting weak regressors sequentially	Predicting house prices, sales, etc
SVC (Support Vector Classifier)	Classification	Maximizing margin with hyperplanes	High-dimensional data with complex decision boundaries

✅ Random Forest: Works well with large datasets, less prone to overfitting.

✅ Gradient Boosting: More powerful but sensitive to hyperparameters and requires tuning.

✅ SVC: Best for complex classification tasks with high-dimensional data.

19. what about heatmap

A heatmap is a data visualization technique that shows the magnitude of values using colors. It is often used for correlation matrices, feature importance, and data distribution analysis. A confusion matrix heatmap is useful for evaluating classification models.
A correlation matrix shows how features in a dataset are related. A heatmap helps visualize these relationships.
How to Interpret?

- Values close to +1 → Strong positive correlation
- Values close to -1 → Strong negative correlation
- Values near 0 → No correlation

20. What is GridSearchCV?

- **It exhaustively searches over a specified parameter grid.**
- **It uses cross-validation (CV) to evaluate each parameter combination.**
- **It helps prevent overfitting by selecting the best-performing hyperparameters.**
- **This helps find the best hyperparameters automatically!**

GridSearchCV: Hyperparameter Tuning in Machine Learning
GridSearchCV is a hyperparameter tuning technique in Scikit-Learn that systematically tests different combinations of hyperparameters to find the best model.

Why Use GridSearchCV?

✅ Automates hyperparameter tuning

✅ Prevents overfitting

✅ Improves model performance

Project 2 : Salary-Prediction-Classification

1. **Why does the model use sigmoid activation in the last layer?**
   Since this is a binary classification task, the sigmoid function ensures outputs are between 0 and 1, representing probabilities.
2. **How do I improve model performance?**
   - Increase data size
   - Try different architectures (e.g., deeper layers)
   - Use learning rate scheduling (e.g., ReduceLROnPlateau)
   - Use L2 regularization (e.g., kernel_regularizer=l2(0.01))
   - Adjust dropout rates to avoid underfitting/overfitting.
3. **How do I tune hyperparameters like learning rate and batch size?**
   Use GridSearchCV or Keras Tuner to automate hyperparameter tuning.
4. **What is OneHotEncoder?**
   OneHotEncoder is used to convert categorical variables into a binary (0s and 1s) format, where each unique category gets its own column.

5. **What is LabelEncoder?**
   LabelEncoder transforms categorical labels into numerical values (e.g., "Red" → 0, "Green" → 1). It is used when the categories have no ordinal relationship.
6. **What is ColumnTransformer?**
   ColumnTransformer allows applying different preprocessing steps (e.g., scaling numerical features and encoding categorical features) to different columns of a dataset.

7. **What is Sequential?**
   Sequential is a Keras model type where layers are stacked one after another in a straightforward manner, making it ideal for feedforward neural networks.
8. **What is Dense layer?**
   A Dense layer in Keras is a fully connected layer where each neuron is connected to all neurons in the previous layer. It is a fundamental layer in deep learning models.

9. **What is ReduceLROnPlateau?**
   ReduceLROnPlateau is a Keras callback that reduces the learning rate when the model's performance stops improving, helping to fine-tune training.

10. **What is EarlyStopping?**

EarlyStopping stops training when the model's validation performance stops improving, preventing overfitting and saving time.

11. **What is regularizers, L2?**

L2 regularization (also known as Ridge regularization) adds a penalty to the model's weights to prevent overfitting by discouraging large weight values.

12. **What happens if a dataset has both categorical and numerical features?**

We need to preprocess them separately using tools like Column Transformer to apply different transformations to each type.

ColumnTransformer: Detailed Explanation

The ColumnTransformer in sklearn.compose is a powerful tool that allows you to apply different transformations to different columns in a dataset. It is especially useful when working with datasets containing both numerical and categorical features.

Why Use ColumnTransformer?

- Handles multiple preprocessing steps at once – You can apply different transformations to different types of columns.
- Prevents data leakage – Ensures proper transformation within a pipeline.
- Makes code cleaner and more structured – Instead of manually encoding and scaling features, you can do it in one step.

How Does ColumnTransformer Work?

- It takes a list of tuples, where each tuple consists of:
- A name (string identifier for the transformation)
- A transformer (e.g., StandardScaler, OneHotEncoder)
- The columns to apply the transformation to