# DSBDA Viva

Here is a well-structured Q&A bank from Question 1 to 41 with proper format (Definition, Example, Code snippet). I've also started adding basic Machine Learning (ML) concepts and algorithms from Q42 onward.

---

## 🔹 Q1) What is Pandas library in Python?

**Definition**:
Pandas is an open-source Python library used for data manipulation and analysis, particularly for structured data.

**Example**:
Useful for reading, processing, and analyzing CSV or Excel files.

**Code**:

```
python

import pandas as pd
df = pd.read_csv("data.csv")
print(df.head())
```

---

## 🔹 Q2) List some key features of Pandas.

**Definition**:

- Fast and efficient DataFrame object
- Tools for reading/writing data
- Handling of missing data
- Data alignment and integrated handling of time series

---

## 🔹 Q3) What is NumPy Library in Python?

**Definition**:
NumPy (Numerical Python) provides support for arrays, matrices, and many mathematical functions.

**Example**:
Used for numerical calculations and matrix operations.

**Code**:

```
python

import numpy as np
arr = np.array([1, 2, 3])
print(arr.mean())
```

---

## 🔹 Q4) What is Matplotlib library?

**Definition**:
Matplotlib is a 2D plotting library for creating static, animated, and interactive visualizations in Python.

**Example**:
Creating line graphs, bar charts, and histograms.

**Code**:

```
python

import matplotlib.pyplot as plt
plt.plot([1, 2, 3], [4, 5, 6])
plt.show()
```

---

## 🔹 Q5) Difference between Seaborn and Matplotlib?

**Definition**:

- **Seaborn**: Built on top of Matplotlib; provides high-level interface for statistical graphics.
- **Matplotlib**: More control, but more code needed for customization.

**Example**:

```
python

import seaborn as sns
sns.histplot([1,2,3,4,5])
```

---

## 🔹 Q6) Are Sklearn and Scikit-learn the same?

**Definition**:
Yes, `` `sklearn` `` is the import name for Scikit-learn library in Python.

**Use in DS**:
Used for ML tasks like classification, regression, and clustering.

**Code**:

```
python

from sklearn.linear_model import LinearRegression
```

---

## 🔹 Q7) Functions in Pandas and NumPy?

**Pandas**:

- `` `read_csv()` ``, `` `head()` ``, `` `describe()` ``, `` `dropna()` ``, `` `groupby()` ``

**NumPy**:

- `` `array()` ``, `` `mean()` ``, `` `median()` ``, `` `std()` ``, `` `reshape()` ``

---

## 🔹 Q8) What is DataFrame in Python?

**Definition**:
A 2D labeled data structure in Pandas (like a table with rows and columns).

**Code**:

```
python

df = pd.DataFrame({'a':[1,2], 'b':[3,4]})
```

---

## 🔹 Q9) How to find duplicates in Python?

**Code**:

```
python

df.duplicated()
df[df.duplicated()]
```

---

## 🔹 Q10) Use of `` `describe()` `` command?

**Definition**:
Provides statistical summary of numeric columns.

**Code**:

```
python

df.describe()
```

---

## 🔹 Q11) Naive Bayes classification algorithms in Python?

**Types**:

- GaussianNB
- MultinomialNB
- BernoulliNB

**Code**:

python

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
```

---

## 🔹 Q12) Significance of Confusion Matrix?

**Definition**:
Used to evaluate classification model performance.

---

## 🔹 Q13) TP, TN, FP, FN in Confusion Matrix?

- **TP**: True Positives
- **TN**: True Negatives
- **FP**: False Positives
- **FN**: False Negatives

---

## 🔹 Q14) What is Recall?

**Definition**:
Recall = TP / (TP + FN)

---

## 🔹 Q15) What is Precision?

**Definition**:
Precision = TP / (TP + FP)

---

## 🔹 Q16) What is F1 Score?

**Definition**:
F1 = 2 * (Precision * Recall) / (Precision + Recall)

---

## 🔹 Q17) Why is Data Visualization important?

**Definition**:
It simplifies complex data and highlights trends/patterns for better decisions.

---

## 🔹 Q18) What is an Outlier?

**Definition**:
An observation far from other values in a dataset.

---

## 🔹 Q19) Histogram vs Pie Chart?

- **Histogram**: Frequency distribution (use for continuous data)
- **Pie Chart**: Proportions of a whole (use for categorical data)

---

## 🔹 Q20) Challenges in Big Data Visualization?

- High volume
- High velocity
- Data variety
- Processing power
- Real-time rendering

---

## 🔹 Q21) What is Joint Plot, Dist Plot?

**Definition**:

- **jointplot**: Combines scatter plot and histogram.
- **distplot**: Shows distribution of a variable.

**Code**:

```python
sns.jointplot(x='age', y='salary', data=df)
sns.histplot(df['age'])
```

---

## 🔹 Q22) Tools for Data Visualization?

- Matplotlib

- Seaborn
- Plotly
- Power BI
- Tableau

---

### 🔹 Q23) What is Data Wrangling?

**Definition**:
Cleaning and transforming raw data into usable format.

---

### 🔹 Q24) What is Data Transformation?

**Definition**:
Process of converting data into suitable format for analysis.

---

### 🔹 Q25) Use of `` `StandardScaler` `` in Python?

**Definition**:
Standardizes features by removing mean and scaling to unit variance.

**Code**:

```python
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df)
```

---

### 🔹 Q26) What is Hadoop?

**Definition**:
A framework for distributed storage and processing of large data sets.

---

### 🔹 Q27) What is HDFS and MapReduce?

- **HDFS**: Hadoop Distributed File System (stores data)
- **MapReduce**: Processes data in parallel.

---

### 🔹 Q28) Components of Hadoop Ecosystem?

- HDFS

- MapReduce
- YARN
- Hive
- Pig
- HBase
- Sqoop
- Flume

---

## 🔹 Q29) What is Scala?

**Definition**:
A general-purpose programming language combining functional and OOP.

---

## 🔹 Q30) Features of Scala?

- Type inference
- Immutability
- Interoperable with Java
- Functional constructs

---

## 🔹 Q31) How is Scala different from Java?

- Scala supports functional programming.
- Concise syntax.
- Better for concurrent applications.

---

## 🔹 Q32) Applications of Scala?

- Apache Spark
- Data science
- Web apps
- DSLs (Domain-Specific Languages)

---

## 🔹 Q33) What is Data Science?

**Definition**:
Field of extracting insights from data using tools like Python, ML, and statistics.

---

### 🔹 Q34) What is Big Data?

**Definition**:
Data that is too large or complex for traditional tools to process.

---

### 🔹 Q35) Characteristics of Big Data?

- Volume
- Velocity
- Variety
- Veracity
- Value

---

### 🔹 Q36) Phases in Data Science Life Cycle?

- Data collection
- Data wrangling
- Data analysis
- Model building
- Evaluation
- Deployment

---

### 🔹 Q37) What is Central Tendency?

**Definition**:
A central value (Mean, Median, Mode) representing a dataset.

---

### 🔹 Q38) What is Dispersion?

**Definition**:
Measure of spread of data (Range, Variance, SD).

---

### 🔹 Q39) Mean, Mode, Mid-range, Median?

**Data**: [10,22,13,10,21,43,77,21,10]

- **Mean**: (10+22+13+10+21+43+77+21+10)/9 = 25.22
- **Median**: 21
- **Mode**: 10
- **Mid-Range**: (min+max)/2 = (10+77)/2 = 43.5

---

## 🔹 Q40) What is Variance?

**Definition**:
Average of squared deviations from the mean.

---

## 🔹 Q41) What is Standard Deviation?

**Definition**:
Square root of variance.

---

Would you like me to continue with **ML Concepts from Q42 onward**?

Great! Let's continue from **Q42 onward**, covering **basic ML concepts**, starting with **Confusion Matrix** and **Feature Scaling** along with examples and code.

---

## 🔹 Q42) What is a Confusion Matrix? Explain with example.

**Definition**:
A confusion matrix is a table used to evaluate the performance of a classification model. It compares actual vs. predicted values.

**Structure**:

| | Predicted Positive | Predicted Negative |
|---|---|---|
| Actual Pos | True Positive (TP) | False Negative (FN) |
| Actual Neg | False Positive (FP) | True Negative (TN) |

**Example**: Suppose we have a binary classifier that predicts whether a patient has a disease:

- TP = 50 (Correctly predicted diseased)
- FP = 10 (Incorrectly predicted diseased)
- TN = 30 (Correctly predicted healthy)
- FN = 5 (Incorrectly predicted healthy)

**Code**:

```python
from sklearn.metrics import confusion_matrix
y_true = [1, 0, 1, 1, 0, 1, 0]
y_pred = [1, 0, 1, 0, 0, 1, 1]
cm = confusion_matrix(y_true, y_pred)
print(cm)
```

---

## 🔹 Q43) What is Feature Scaling? Why is it needed?

**Definition**:
Feature Scaling is a technique to normalize the range of independent variables or features of data.

**Why needed?**

- Algorithms like SVM, KNN, and gradient descent converge faster when data is scaled.
- Prevents features with large values from dominating.

---

## 🔹 Q44) Methods of Feature Scaling with example?

### 1. Standardization (Z-score Normalization)

Transforms data to have **mean = 0** and **std dev = 1**.

**Code**:

```python
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_data = scaler.fit_transform([[1, 2], [2, 4], [3, 6]])
print(scaled_data)
```

### 2. Min-Max Scaling (Normalization)

Scales data to a **range [0, 1]**.

**Code**:

```python
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform([[1, 2], [2, 4], [3, 6]])
print(scaled_data)
```

---

## 🔹 Q45) What is Train Test Split?

**Definition**:
Used to split data into training and testing sets to evaluate model performance.

**Code**:

```
python

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

---

## 🔹 Q46) What is Underfitting and Overfitting?

| Concept | Definition | Behavior |
|---|---|---|
| Underfitting | Model is too simple | Poor performance on both training and test data |
| Overfitting | Model is too complex | Excellent training performance, poor test performance |

---

## 🔹 Q47) What is Cross-Validation?

**Definition**:
Technique to evaluate models by splitting the data into multiple training and testing sets (folds).

**Code**:

```
python

from sklearn.model_selection import cross_val_score
scores = cross_val_score(model, X, y, cv=5)
print(scores)
```

---

Would you like me to continue with specific **ML algorithms** like Linear Regression, KNN, SVM next?

Great! Let's continue with **basic machine learning algorithms**. For each, I'll give:

- **Definition**
- **Example use-case**
- **Code snippet (with `scikit-learn`)**

---

## 🔹 Q48) Linear Regression

**Definition**:
Linear Regression is a supervised ML algorithm used for predicting a continuous target variable based on independent variables using a straight line (`**y = mx + c**`).

**Example**: Predicting house prices based on size.

**Code**:

```
python

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

# Sample Data
X = [[100], [200], [300], [400]]
y = [10, 20, 30, 40]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

model = LinearRegression()
model.fit(X_train, y_train)

print("Prediction:", model.predict([[250]]))
```

---

## 🔹 Q49) Logistic Regression

**Definition**:
Logistic Regression is a classification algorithm used to predict binary outcomes (0 or 1) using the logistic function (sigmoid).

**Example**: Predicting whether an email is spam or not.

**Code**:

```
python

from sklearn.linear_model import LogisticRegression

X = [[1], [2], [3], [4]]
y = [0, 0, 1, 1]

model = LogisticRegression()
model.fit(X, y)

print(model.predict([[2.5]]))  # Output: 0 or 1
```

---

## 🔹 Q50) K-Nearest Neighbors (KNN)

**Definition**:
KNN is a classification algorithm that classifies data points based on the labels of their nearest neighbors.

**Example**: Handwritten digit recognition.

**Code**:

```
python

from sklearn.neighbors import KNeighborsClassifier

X = [[1], [2], [3], [6], [7], [8]]
y = [0, 0, 0, 1, 1, 1]

model = KNeighborsClassifier(n_neighbors=3)
model.fit(X, y)

print(model.predict([[4]]))  # Expected: 0 or 1 based on neighbors
```

---

### 🔹 Q51) Support Vector Machine (SVM)

**Definition**:
SVM finds the best hyperplane that separates data points of different classes with the maximum margin.

**Example**: Classifying images as cats or dogs.

**Code**:

```
python

from sklearn.svm import SVC

X = [[1, 2], [2, 3], [3, 3], [6, 7], [7, 8]]
y = [0, 0, 0, 1, 1]

model = SVC(kernel='linear')
model.fit(X, y)

print(model.predict([[4, 4]]))
```

---

### 🔹 Q52) Naive Bayes Classifier

**Definition**:
Naive Bayes is a probabilistic classifier based on Bayes' Theorem assuming feature independence.

**Example**: Text classification like spam detection.

**Code**:

```
python

from sklearn.naive_bayes import GaussianNB

X = [[1, 20], [2, 21], [3, 22], [10, 100], [11, 101]]
y = [0, 0, 0, 1, 1]
```

```
model = GaussianNB()
model.fit(X, y)

print(model.predict([[5, 50]]))
```

---

Would you like me to include Decision Trees, Random Forest, and Clustering (like K-Means) next?

Perfect! Let's continue with **more machine learning algorithms** like **Decision Trees, Random Forest, and Clustering (K-Means)**.

---

## 🔹 Q53) Decision Tree

**Definition**:
A Decision Tree is a flowchart-like tree structure where each internal node represents a decision on a feature, each leaf node represents an outcome.

**Example**: Predicting whether a person buys a product based on age and income.

**Code**:

```
python

from sklearn.tree import DecisionTreeClassifier

X = [[25, 50000], [35, 60000], [45, 80000], [20, 20000]]
y = [0, 1, 1, 0]  # 0 = No, 1 = Yes

model = DecisionTreeClassifier()
model.fit(X, y)

print(model.predict([[30, 55000]]))
```

---

## 🔹 Q54) Random Forest

**Definition**:
Random Forest is an ensemble learning method that uses multiple decision trees to improve accuracy and avoid overfitting.

**Example**: Credit scoring system.

**Code**:

```
python

from sklearn.ensemble import RandomForestClassifier

X = [[1, 2], [3, 4], [5, 6], [7, 8]]
y = [0, 0, 1, 1]

model = RandomForestClassifier(n_estimators=100)
model.fit(X, y)
```

```
print(model.predict([[4, 5]]))
```

## 🔹 Q55) K-Means Clustering

**Definition**:
K-Means is an unsupervised learning algorithm used to group data into **K clusters** based on similarity.

**Example**: Customer segmentation.

**Code**:

```
python

from sklearn.cluster import KMeans

X = [[1], [2], [3], [10], [11], [12]]
model = KMeans(n_clusters=2)
model.fit(X)

print(model.labels_)  # Shows which cluster each point belongs to
print(model.predict([[5]]))  # Predicts the cluster of a new point
```

## 🔹 Q56) What is Clustering in Machine Learning?

**Definition**:
Clustering is the task of grouping a set of objects so that objects in the same group (called a cluster) are more similar to each other than to those in other groups.

- **Types**: K-Means, Hierarchical, DBSCAN
- **Used in**: Market segmentation, Social network analysis, etc.

## 🔹 Q57) Supervised vs Unsupervised Learning

| Type | Input Data | Output | Example Algorithms |
|---|---|---|---|
| Supervised | Labeled | Predict label | Linear Regression, SVM, KNN |
| Unsupervised | Unlabeled | Grouping | K-Means, PCA, DBSCAN |

Would you like me to also include **evaluation metrics**, **PCA**, or continue with **deep learning basics**?

# ML Graphs and Plots, Concepts

In Machine Learning (ML), graphs, plots, and maps are essential tools for visualizing data, understanding patterns, and interpreting model performance. Below is a breakdown of the common types of graphs and plots used in ML, along with basic terminology and algorithms.

## 1. Graphs and Plots in ML

### a) Line Plot

- **Definition**: A line plot is used to visualize continuous data points, typically showing trends over time or sequential observations.
- **Example**: Plotting the loss vs. epoch during training to visualize the convergence of a model.

### b) Scatter Plot

- **Definition**: A scatter plot is used to display the relationship between two variables by plotting points on a 2D plane.
- **Example**: Visualizing the relationship between height and weight of a population.

### c) Histogram

- **Definition**: A histogram is used to show the distribution of a dataset. It divides the data into bins and counts the number of occurrences in each bin.
- **Example**: Displaying the distribution of age in a dataset.

### d) Box Plot

- **Definition**: A box plot (or box-and-whisker plot) is used to visualize the spread and skewness of data by showing the median, quartiles, and outliers.
- **Example**: Showing the distribution of salary data across different job titles.

### e) Heatmap

- **Definition**: A heatmap visualizes data using color gradients to represent values, often used to represent correlation matrices or the performance of a model.
- **Example**: A heatmap showing the correlation between various features in a dataset.

### f) Bar Plot

- **Definition**: A bar plot is used to represent categorical data with rectangular bars with lengths proportional to the values they represent.
- **Example**: Comparing the performance of different machine learning models.

#### g) Pair Plot

- **Definition**: A pair plot (also called a scatterplot matrix) is used to show pairwise relationships between variables in a dataset.
- **Example**: Pair plots to visualize relationships between features in a multi-dimensional dataset like Iris.

#### h) ROC Curve (Receiver Operating Characteristic Curve)

- **Definition**: A graphical plot used to evaluate the performance of binary classification models.
- **Example**: Analyzing the trade-off between true positive rate and false positive rate for a classification model.

#### i) Precision-Recall Curve

- **Definition**: This curve is used to evaluate classification models when classes are imbalanced, focusing on precision vs. recall.
- **Example**: A classification problem with rare disease detection.

#### j) Confusion Matrix

- **Definition**: A confusion matrix is used to assess the performance of a classification model by displaying the actual vs. predicted classifications.
- **Example**: Showing the results of a spam email classifier.

---

### 2. Basic Terminologies in ML

#### a) Feature

- **Definition**: An individual measurable property or characteristic of a phenomenon being observed (e.g., age, height, color).

#### b) Label

- **Definition**: The target or output variable that a model is trying to predict or classify.

#### c) Model

- **Definition**: A mathematical representation of a real-world process used to make predictions based on input data.

**d) Training Data**

- **Definition**: A set of data used to train a model.

**e) Test Data**

- **Definition**: A set of data used to evaluate the performance of a trained model.

**f) Overfitting**

- **Definition**: When a model performs well on training data but poorly on unseen test data due to learning the noise in the data.

**g) Underfitting**

- **Definition**: When a model is too simple and fails to capture the underlying patterns in the data.

---