**Guru Tegh Bahadur Institute of Technology, New Delhi**

# Foundations of Data Science (Question Bank)

**Course Name: B.Tech (AIML)     Semester: 3rd     SUB CODE: AIML203**

## UNIT-I

### Section I: Multiple Choice Questions (MCQs) with Answers

**1. What is the primary goal of data science?**

(a) To program complex software applications

(b) To extract insights and knowledge from data

(c) To design and maintain computer networks

(d) To create visually appealing presentations

Ans: (b) To extract insights and knowledge from data

**2. Which of the following is NOT a typical application of data science?**

(a) Fraud detection in financial transactions

(b) Recommending products to customers online

(c) Designing user interfaces for mobile apps

(d) Predicting weather patterns

Ans: (d) Predicting weather patterns

**3. A data scientist is responsible for:**

(a) Managing databases and servers

(b) Collecting and cleaning raw data

(c) Writing complex marketing copy

(d) Designing computer graphics

Ans: (b) Collecting and cleaning raw data

**4. Which skill is NOT essential for a data scientist?**

(a) Programming languages like Python

(b) Statistical analysis and modeling

(c) Excellent communication skills

(d) Web development experience

Ans: (d) Web development experience

**5. Why is Python a popular choice for data analysis?**

(a) It's specifically designed for financial modeling

(b) It has a large collection of data science libraries

(c) It's the most secure programming language

(d) It requires minimal coding experience

Ans: (b) It has a large collection of data science libraries

**6. What is the first step in the data science life cycle?**

(a) Model training and evaluation

(b) Data understanding and pre-processing

(c) Data visualization and communication

(d) Model deployment and monitoring

Ans: (b) Data understanding and pre-processing

**7. What is the role of domain knowledge in data science?**

(a)  Understanding the specific industry or problem context

(b) Knowing how to write complex algorithms

(c)  Expertise in database management systems

(d)  Excellent graphic design skills

Ans: (a) Understanding the specific industry or problem context

**8. How is structured data different from unstructured data?**

(a) Structured data is organized in tables, while unstructured data is not.

(b) Structured data is always numerical, while unstructured can be text or images.

(c) Structured data is more difficult to analyze than unstructured data.

(d) Structured data is created by humans, while unstructured data is generated by machines.

Ans: (a) Structured data is organized in tables, while unstructured data is not.

**9. What is a synthetic dataset?**

(a) A real-world dataset collected through surveys or experiments

(b) A dataset artificially created with specific characteristics

(c) A dataset containing errors and inconsistencies

(d) A dataset that is too large to be analyzed

Ans: (b) A dataset artificially created with specific characteristics

**10. Which software is most commonly used to create a synthetic dataset?**

(a) Python programming language

(b) Microsoft Excel

(c) Data visualization tools like Tableau

(d) Machine learning libraries like TensorFlow

Ans: (b) Microsoft Excel

**Section II: Short answer type questions**

**1. What is data science and how is it applied?**

**Answer:** Data science is a field that extracts knowledge and insights from data. It's used in many fields like finance (fraud detection), retail (product recommendations), healthcare (predictive medicine), and weather forecasting.

**2. What does a data scientist do and what skills are needed?**

**Answer:** Data scientists collect, clean, analyze data, build models, and communicate findings. They need skills in programming (Python), statistics, machine learning, and communication.

**3. Why is understanding data types like structured and unstructured data important?**

**Answer:** Structured data (organized in tables) and unstructured data (text, images) require different processing techniques. Understanding types helps choose the right approach for analysis.

**4. What is data pre-processing and why is it significant?**

**Answer:** Data pre-processing is cleaning and preparing raw data for analysis. It's crucial to handle missing values, inconsistencies, and formatting issues to ensure reliable results.

**5. How can Microsoft Excel be used to create a synthetic dataset?**

**Answer:** Excel can be used to generate artificial data with specific characteristics using formulas and functions. This is helpful for practicing data analysis techniques before working with real-world datasets.

**Section III: Long answer type questions**

**1. Explain the concept of data science, its role in the modern world, and provide examples of its applications in different industries.**

**Answer:** Data science is an interdisciplinary field that uses scientific methods, statistics, programming, and domain knowledge to extract knowledge and insights from data. In today's data-driven world, data science plays a crucial role in solving complex problems, making informed decisions, and driving innovation. Here are some industry-specific applications:

- **Finance:** Fraud detection, risk assessment, personalized financial products.
- **Retail:** Product recommendations, customer segmentation, demand forecasting.
- **Healthcare:** Predictive medicine, personalized treatment plans, drug discovery.
- **Manufacturing:** Predictive maintenance, process optimization, quality control.
- **Media & Entertainment:** Content recommendation, targeted advertising, understanding user behavior.

**2. Describe the typical responsibilities of a data scientist and elaborate on the essential skills required to excel in this role.**

**Answer:** Data scientists are responsible for the entire data science life cycle, which can be broken down into these stages:

- **Data Acquisition:** Collecting data from various sources (databases, APIs, web scraping).
- **Data Understanding & Pre-processing:** Cleaning, exploring, and preparing data for analysis.
- **Exploratory Data Analysis (EDA) & Feature Engineering:** Identifying patterns, trends, and creating new features from existing data.
- **Model Building & Training:** Selecting appropriate algorithms, building and training models.
- **Model Evaluation & Testing:** Assessing model performance, identifying biases and errors.
- **Deployment & Communication:** Implementing models in production and communicating insights effectively.

**Essential skills for a data scientist include:**

- **Programming:** Python is dominant, with knowledge of R and SQL also valuable.
- **Statistics & Machine Learning:** Understanding statistical concepts, model training, and evaluation techniques.
- **Data Wrangling & Pre-processing:** Ability to clean, manipulate, and transform data for analysis.

- **Domain Knowledge:** Understanding the specific industry or problem context is crucial.
- **Communication & Visualization:** Presenting insights clearly and concisely, using data visualizations effectively.

**3. Explain the difference between structured and unstructured data. Discuss the challenges associated with analyzing unstructured data and how data science techniques address these challenges.**

**Answer:** Data can be categorized into structured and unstructured data based on its format and organization.

- **Structured Data:** Highly organized data stored in a predefined format, like rows and columns in a database table. It's easily usable with standard data analysis tools.
- **Unstructured Data:** Data lacking a consistent format, including text documents, images, audio, video, and social media posts. Analyzing this data requires specific techniques.

**Challenges with Unstructured Data:**

- **Heterogeneity:** Varied formats and lack of standardization.
- **Scalability:** Large volumes of unstructured data can be difficult to process.
- **Meaning Extraction:** Capturing the underlying meaning from text, images, or audio requires specialized techniques.

**Data Science Techniques for Unstructured Data:**

- **Natural Language Processing (NLP):** Analyzing text data, extracting sentiment, and identifying topics.
- **Computer Vision:** Extracting insights from images and videos.
- **Text Mining:** Identifying patterns and trends within textual data.

**4. Why is data pre-processing an essential step in the data science workflow? Elaborate on different data pre-processing techniques used to address common data quality issues.**

**Answer:** Data pre-processing is a crucial phase in data science as it prepares raw data for analysis, leading to more reliable and accurate results.

**Common Data Quality Issues Addressed by Pre-processing Techniques:**

- **Missing Values:** Techniques like imputation (filling in missing values) or deletion are employed.
- **Inconsistencies:** Formatting errors, typos, and outliers are addressed through standardization and normalization techniques.
- **Scaling:** Features are scaled to a common range to ensure all features have equal influence in models.
- **Encoding Categorical Variables:** Converting categorical variables (like text) into numerical representations suitable for analysis.

**5. Explain how Microsoft Excel can be used to create a synthetic dataset for data exploration purposes. Discuss the advantages and limitations of using Excel-generated datasets.**

**Answer:** While not a powerhouse data science tool, Microsoft Excel can be surprisingly useful for creating synthetic datasets for initial data exploration and practicing data analysis techniques. Here's a detailed breakdown of its capabilities and considerations:

**Techniques for Synthetic Data Generation in Excel:**

- **Random Number Functions:**
    - **RAND():** Generates a random number between 0 (inclusive) and 1 (exclusive). You can use this to create random values for a continuous variable.
    - **RANDBETWEEN(lower limit, upper limit):** Generates a random integer between a specified lower and upper limit (inclusive). This is useful for creating discrete variables with a defined range.
- **Data Validation:** This feature allows you to set restrictions on the type of data that can be entered into a cell. For instance, you can limit entries to a specific range of numbers or a predefined list of options. This helps control the characteristics of your synthetic data.
- **Formulas and Functions:** You can leverage Excel's extensive library of formulas to create more complex synthetic data. Here are some examples:
    - **VLOOKUP/INDEX MATCH:** These functions can be used to create datasets with relationships between variables. You can define a lookup table with specific values and then use these functions to populate another column with corresponding values based on a linking criteria.
    - **Conditional Formatting:** Used to highlight cells based on specific conditions. This can be helpful for visually identifying patterns or trends within your synthetic data.

**Advantages of Using Excel-Generated Datasets:**

- **Accessibility:** Most people already have access to Excel, making it a readily available tool.
- **Simplicity:** The basic functions and formulas are relatively easy to learn and use, allowing for quick creation of simple datasets.
- **Visualization:** Excel offers built-in data visualization tools (charts and graphs) to analyze and understand the synthetic data you generate.

**Limitations of Using Excel-Generated Datasets:**

- **Complexity:** Creating intricate datasets with complex relationships or distributions can be cumbersome and time-consuming in Excel.
- **Scalability:** Large datasets become unwieldy to manage and manipulate in Excel.
- **Limited Statistical Control:** While you can control some aspects of the data, it's difficult to achieve complex statistical distributions often required for advanced analysis.
- **Reproducibility:** Recreating the exact same dataset can be challenging due to reliance on manual steps and random number generation.

**In Conclusion:**

Excel can be a valuable starting point for data exploration and practicing basic data analysis skills. However, for professional data science work involving large datasets or complex statistical modeling, dedicated data science tools like Python with libraries like NumPy and Pandas are much more efficient and offer greater control.

## UNIT-II

### Section I: Multiple Choice Questions (MCQs) with Answers

**1. Which of the following is a valid variable name in Python?**

a) 1variable

b) variable-1

c) _variable1

d) variable 1

Answer: c) _variable1

**2. What is the correct syntax to print a value in Python?**

a) print "Hello, World!"

b) echo "Hello, World!"

c) printf("Hello, World!")

d) print("Hello, World!")

Answer: d) print("Hello, World!")

**3. Which of the following statements is used to check a condition in Python?**

a) if

b) for

c) while

d) def

Answer: a) if

**4. What is the output of the following code: print(2 + 3 * 4)?**

a) 20

b) 14

c) 24

d) 12

Answer: b) 14

**5. How do you create a loop that iterates 5 times in Python?**

a) for i in range(1, 5):

b) for i in range(5):

c) while i < 5:

d) do 5 times

Answer: b) for i in range(5):

**6. How can you import a dataset in Python using Pandas?**

a) import pandas as pd; df = pd.read_csv('file.csv')

b) import pandas as pd; df = pd.read('file.csv')

c) import pandas as pd; df.read_csv('file.csv')

d) import pandas as pd; df = read_csv('file.csv')

Answer: a) import pandas as pd; df = pd.read_csv('file.csv')

**7. Which function is used to display the first few rows of a DataFrame in Pandas?**

a) head()

b) tail()

c) first()

d) show()

Answer: a) head()

**8. How can missing values be handled in a dataset using Pandas?**

a) df.fillna(0)

b) df.dropna()

c) Both a and b

d) None of the above

Answer: c) Both a and b

**9. Which of the following is an arithmetic operation in Python?**

a) *

b) %

c) +

d) All of the above

Answer: d) All of the above

**10. What does the following code do: for i in range(3): print(i)?**

a) Prints 0, 1, 2

b) Prints 1, 2, 3

c) Prints 0, 1, 2, 3

d) Prints 1, 2, 3, 4

Answer: a) Prints 0, 1, 2

**Section II: Short answer type questions**

1. **What is a variable in Python?**

Answer: A variable in Python is a name given to a memory location that stores a value. It acts as a placeholder for storing data that can be used and manipulated within a program.

2. **How do you perform addition and multiplication in Python?**

Answer: Addition is performed using the + operator and multiplication using the * operator. For example, result = 2 + 3 * 4 computes to 14.

3. **Write a basic if condition to check if a number is positive.**

Answer: if number > 0:

```
    print("The number is positive")
```

4. **How can you get basic insights like the number of rows and columns in a Pandas DataFrame?**

Answer: You can use the df.shape attribute or the df.info() method to get basic insights such as the number of rows and columns in a DataFrame.

5. **What method would you use to handle missing values by filling them with the mean in Pandas?**

Answer: You can use df.fillna(df.mean(), inplace=True) to fill missing values with the mean of the respective columns.

**Section III: Long answer type questions**

1. **Explain how loops work in Python with examples of a for loop and a while loop.**

Answer: **For loop:** A for loop in Python iterates over a sequence (such as a list, tuple, dictionary, set, or string) and executes a block of code for each element in the sequence.

```
for i in range(5):
   print(i)
```

This code prints the numbers 0 to 4.

**While loop**: A while loop repeatedly executes a block of code as long as a specified condition is true.

```
i = 0
while i < 5:
   print(i)
   i += 1
```

This code also prints the numbers 0 to 4. The condition i < 5 is checked before each iteration, and i is incremented in each iteration.

2. **Describe the process of importing and exporting data in Python using Pandas.**

Answer: **Importing data:** To import data, you can use functions like pd.read_csv(), pd.read_excel(), pd.read_sql(), etc. For example, to import a CSV file:

```
import pandas as pd
df = pd.read_csv('file.csv')
```

**Exporting data:** To export data, you can use functions like df.to_csv(), df.to_excel(), df.to_sql(), etc. For example, to export a DataFrame to a CSV file:

```
df.to_csv('output.csv', index=False)
```

These methods provide easy ways to read and write data between different formats and files.

3. **Explain how to clean and prepare a dataset by identifying and handling missing values.**

Answer: Cleaning and preparing a dataset often involves dealing with missing values, which can be identified using functions like df.isnull() or df.isna(). To handle missing values:

Remove rows/columns: Use df.dropna() to remove rows or columns with missing values.

Fill missing values: Use df.fillna(value) to fill missing values with a specified value, such as the mean, median, or a constant.

df['column'].fillna(df['column'].mean(), inplace=True)

This replaces missing values in 'column' with the column's mean. Handling missing values is crucial for maintaining data integrity and ensuring accurate analysis.

4. **Discuss the importance of getting basic insights from datasets and provide examples of methods used for this purpose.**

Answer: Getting basic insights from datasets is essential to understand the structure, contents, and quality of the data before performing any analysis. Key methods include:

**df.head():** Displays the first few rows of the DataFrame.

**df.tail():** Displays the last few rows of the DataFrame.

**df.info():** Provides a summary of the DataFrame, including data types and non-null counts.

**df.describe():** Generates descriptive statistics for numerical columns.

**df.head():** # Shows the first 5 rows

**df.info():** # Summary of the DataFrame

**df.describe():** # Descriptive statistics

These methods help in quickly assessing the dataset, identifying potential issues, and planning further analysis or cleaning steps.

5. **Write a Python program that reads a dataset, identifies missing values, fills them with the median of the column, and outputs the cleaned dataset.**

Answer:

import pandas as pd

**# Read the dataset**

df = pd.read_csv('file.csv')

**# Identify missing values**

missing_values = df.isnull().sum()

print("Missing values before cleaning:\n", missing_values)

```
# Fill missing values with the median of the column
df.fillna(df.median(), inplace=True)


# Output the cleaned dataset
cleaned_missing_values = df.isnull().sum()
print("Missing values after cleaning:\n", cleaned_missing_values)


# Export the cleaned dataset
df.to_csv('cleaned_file.csv', index=False)
```

This program demonstrates how to handle missing values by filling them with the median, ensuring that the dataset is clean and ready for analysis.

## UNIT-III

**Section I: Multiple Choice Questions (MCQs) with Answers**

1. **Which library is primarily used for numerical computations in Python?**

a) Pandas

b) Matplotlib

c) NumPy

d) Seaborn

**Answer: c) NumPy**

2. **What is the main data structure used in Pandas?**

a) array

b) DataFrame

c) list

d) dictionary

**Answer: b) DataFrame**

3. **Which function in Matplotlib is used to create a basic plot?**

a) plt.show()

b) plt.plot()

c) plt.draw()

d) plt.figure()

**Answer: b) plt.plot()**

**4. What is the primary purpose of the SciPy library?**

a) Data visualization

b) Statistical analysis

c) Data manipulation

d) Scientific computing

**Answer: d) Scientific computing**

**5. How do you import the Seaborn library in Python?**

a) import seaborn as sns

b) import seaborn as sb

c) import seaborn as sn

d) import seaborn

**Answer: a) import seaborn as sns**

**6. Which method in Pandas is used to read a CSV file into a DataFrame?**

a) pd.read_csv()

b) pd.load_csv()

c) pd.open_csv()

d) pd.read_file()

**Answer: a) pd.read_csv()**

**7. Which Matplotlib function is used to create a scatter plot?**

a) plt.scatter()

b) plt.plot()

c) plt.bar()

d) plt.hist()

**Answer: a) plt.scatter()**

**8. What is the purpose of the Seaborn library?**

a) Numerical computations

b) Data manipulation

c) Data visualization

d) Machine learning

**Answer: c) Data visualization**

**9. Which library provides functions for optimization, integration, and statistics?**

a) NumPy

b) Pandas

c) SciPy

d) Seaborn

**Answer: c) SciPy**

**10. What command in Matplotlib displays the plot to the screen?**

a) plt.plot()

b) plt.display()

c) plt.show()

d) plt.render()

**Answer: c) plt.show()**

**Section II: Short Answer Questions**

**1. What is a NumPy array and how is it different from a Python list?**

**Answer:** A NumPy array is a powerful n-dimensional array object which is faster and more efficient for numerical computations compared to Python lists. It supports element-wise operations and broadcasting.

**2. How can you create a DataFrame in Pandas?**

**Answer:** A DataFrame in Pandas can be created using the pd.DataFrame() constructor, by passing a dictionary, list of lists, or another DataFrame.

**3. What is the purpose of the plt.hist() function in Matplotlib?**

**Answer:** The plt.hist() function in Matplotlib is used to create a histogram, which is a graphical representation of the distribution of a dataset.

4. **Describe one method to handle missing data in a Pandas DataFrame.**

**Answer:** One method to handle missing data in a Pandas DataFrame is to use the df.fillna(value) function to replace missing values with a specified value, such as the mean or median of the column.

5. **What is the use of the sns.pairplot() function in Seaborn?**

**Answer:** The sns.pairplot() function in Seaborn creates a grid of scatter plots for pairwise relationships in a dataset, allowing for quick visualization of correlations and distributions.

## Section III: Long Answer Questions

**1. Explain the basic operations that can be performed using NumPy arrays.**

**Answer: Creation:** NumPy arrays can be created using functions like np.array(), np.zeros(), np.ones(), np.arange(), and np.linspace().

**Indexing and Slicing:** NumPy supports slicing similar to Python lists but is more powerful as it can handle multi-dimensional arrays.

**Element-wise Operations:** NumPy allows for efficient element-wise operations such as addition, subtraction, multiplication, and division.

**Aggregation Functions:** Functions like np.sum(), np.mean(), np.std(), np.min(), and np.max() perform aggregation operations over arrays.

**Broadcasting:** NumPy supports broadcasting, which allows arithmetic operations between arrays of different shapes.

**Reshaping:** Arrays can be reshaped using np.reshape() and np.flatten() to change the shape of the array without changing its data.

2. **Describe the process of data manipulation using Pandas, including examples of key functions.**

**Answer: Data manipulation using Pandas involves various steps and functions:**

**Loading Data:** Use pd.read_csv(), pd.read_excel(), etc., to load data into a DataFrame.

**Exploring Data:** Functions like df.head(), df.tail(), df.info(), and df.describe() help in understanding the structure and summary statistics of the data.

**Selecting Data:** Data can be selected using df['column'], df[['col1', 'col2']], df.loc[], and df.iloc[].

**Filtering Data:** Use boolean indexing, such as df[df['column'] > value], to filter rows.

**Adding/Modifying Columns: New columns can be added or modified using df['new_col'] = ....**

**Handling Missing Data:** Use df.dropna() to remove missing data and df.fillna() to fill missing values.

**Grouping and Aggregation:** Use df.groupby('column').agg() to perform group-wise aggregation.

**Merging Data:** Combine DataFrames using pd.merge() or pd.concat().

**Example source code:**

```
df = pd.read_csv('data.csv')

df['new_col'] = df['col1'] + df['col2']

df_grouped = df.groupby('category').sum()

df_cleaned = df.dropna()
```

3. **How can you create and customize a plot in Matplotlib? Provide an example with explanations.**

**Answer:**

**Creating a Plot:** Use plt.plot() to create a line plot.

**Customization:** Customize the plot with titles, labels, legends, and grid lines.

**Example source code:**

```
import matplotlib.pyplot as plt


# Data

x = [1, 2, 3, 4, 5]

y = [2, 3, 5, 7, 11]


# Creating the plot

plt.plot(x, y, label='Prime Numbers', color='blue', linestyle='--', marker='o')
```

```
# Adding titles and labels
plt.title('Prime Numbers Plot')
plt.xlabel('Index')
plt.ylabel('Value')

# Adding a legend
plt.legend()

# Adding grid lines
plt.grid(True)

# Displaying the plot
plt.show()
```

This code creates a line plot with customizations like line color, style, markers, titles, labels, legend, and grid lines.

4. **Discuss the capabilities of Seaborn for data visualization and provide an example of a specialized visualization.**

**Answer:** Seaborn is a Python library built on top of Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics. It simplifies complex visualizations and provides built-in themes.

**Capabilities:**

**Relational plots:** sns.scatterplot(), sns.lineplot()

**Distribution plots:** sns.histplot(), sns.kdeplot(), sns.boxplot()

**Categorical plots:** sns.barplot(), sns.countplot(), sns.violinplot()

**Matrix plots:** sns.heatmap(), sns.clustermap()

**Specialized Visualization Example:**

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Sample data
df = pd.DataFrame({
    'x': [1, 2, 3, 4, 5],
    'y': [10, 20, 30, 40, 50],
    'category': ['A', 'B', 'A', 'B', 'A']
})

# Creating a violin plot
sns.violinplot(x='category', y='y', data=df)

# Customizing the plot
plt.title('Violin Plot of Categories')
plt.xlabel('Category')
plt.ylabel('Values')

# Displaying the plot
plt.show()
```

This example creates a violin plot, which is a combination of box plot and KDE plot, showing the distribution of data across different categories.

**5. Describe the process of performing data processing and visualization using NumPy, Pandas, and Matplotlib. Include a detailed example with explanations for each step.**

Answer: Data processing and visualization involve multiple steps, from data loading and cleaning to analysis and plotting. Here's a detailed example illustrating this process:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

# Load data
df = pd.read_csv('data.csv')
# Inspect data
print(df.head())
print(df.info())
print(df.describe())
# Clean data
missing_values = df.isnull().sum()
print("Missing values:\n", missing_values)
df.fillna(df.median(), inplace=True)
print("Missing values after cleaning:\n", df.isnull().sum())
# Process data
df['total'] = df['A'] + df['B']
df_grouped = df.groupby('category').mean()
print(df_grouped)
# Basic visualization with Matplotlib
df_grouped.plot(kind='bar')
plt.title('Mean Values by Category')
plt.xlabel('Category')
plt.ylabel('Mean Values')
plt.show()
```

```
# Advanced visualization with Seaborn

sns.pairplot(df)

plt.title('Pair Plot of the Data')

plt.show()
```

**# Numerical computations with NumPy**

```
array_A = np.array(df['A'])

mean_A = np.mean(array_A)

std_A = np.std(array_A)

print("Mean of A:", mean_A)

print("Standard Deviation of A:", std_A)
```

This example demonstrates the complete process of loading, cleaning, processing, and visualizing data, integrating the use of NumPy, Pandas, Matplotlib, and Seaborn.

## UNIT-IV

**Section I: Multiple Choice Questions (MCQs) with Answers**

1. **Which Python library is commonly used for scientific computing and includes functions for optimization, integration, and statistics?**

a) NumPy

b) Pandas

c) SciPy

d) Matplotlib

Answer: c) SciPy

2. **What type of learning involves training a model on labeled data?**

a) Supervised Learning

b) Unsupervised Learning

c) Reinforcement Learning

d) Semi-supervised Learning

Answer: a) Supervised Learning

3. **In unsupervised learning, which algorithm is commonly used for clustering data?**

a) Linear Regression

b) Decision Tree

c) K-Means

d) Support Vector Machine

Answer: c) K-Means

4. **Which of the following is a supervised learning algorithm?**

a) K-Means

b) Principal Component Analysis

c) Linear Regression

d) K-Nearest Neighbors

Answer: c) Linear Regression

5. **What is a recommender system used for?**

a) Classification

b) Predicting future trends

c) Providing personalized recommendations

d) Clustering data

Answer: c) Providing personalized recommendations

6. **Which library in Python is used for creating recommender systems and performing collaborative filtering?**

a) Surprise

b) SciPy

c) NumPy

d) Pandas

Answer: a) Surprise

7. **What is the purpose of trend and predictive mining in data analysis?**

a) To visualize data

b) To find patterns and make future predictions

c) To clean and preprocess data

d) To store data efficiently

Answer: b) To find patterns and make future predictions

8. **Which method is commonly used to evaluate the performance of a supervised learning model?**

a) Clustering

b) Cross-validation

c) Data Augmentation

d) Dimensionality Reduction

Answer: b) Cross-validation

9. **In decision-making models, what is a decision tree used for?**

a) Predicting continuous values

b) Making decisions based on rules and conditions

c) Reducing dimensionality

d) Clustering data

Answer: b) Making decisions based on rules and conditions

10. **Which Python library is commonly used for building machine learning models and includes modules for both supervised and unsupervised learning?**

a) TensorFlow

b) Keras

c) Scikit-learn

d) PyTorch

Answer: c) Scikit-learn

**Section II: Short Answer Questions**

1. **What is supervised learning?**

Answer: Supervised learning is a type of machine learning where a model is trained on labeled data, meaning the input data comes with associated correct outputs. The model learns to map inputs to outputs and can make predictions on new, unseen data.

2. **Name two common applications of recommender systems.**

Answer: Recommender systems are commonly used in online retail for product recommendations (e.g., Amazon) and in streaming services for suggesting movies or music (e.g., Netflix, Spotify).

3. **What is the main difference between supervised and unsupervised learning?**

Answer: The main difference is that supervised learning uses labeled data to train the model, whereas unsupervised learning uses unlabeled data and aims to find hidden patterns or intrinsic structures in the input data.

4. **How does a decision tree make decisions?**

Answer: A decision tree makes decisions by splitting the data into subsets based on the value of input features. Each node in the tree represents a feature, and branches represent the decisions based on the feature values, leading to leaf nodes which represent the final outcomes.

5. **What is collaborative filtering in the context of recommender systems?**

Answer: Collaborative filtering is a technique used in recommender systems where the system makes recommendations based on the preferences and behavior of similar users. It assumes that users who agreed in the past will agree in the future.

**Section III: Long Answer Questions**

1. **Explain the process of trend analysis using time series data in Python. Provide an example using a suitable dataset.**

Answer: Trend analysis in time series data involves examining data points collected or recorded at specific time intervals to identify patterns or trends over time. The process typically includes:

**Loading the dataset:** Import the time series data into a Pandas DataFrame.

**Visualizing the data:** Use Matplotlib to plot the time series data.

Decomposing the time series: Use statistical methods to decompose the time series into trend, seasonality, and residuals.

Modeling the trend: Apply models like ARIMA to forecast future trends.

**Example:**

```
import pandas as pd

import matplotlib.pyplot as plt

from statsmodels.tsa.seasonal import seasonal_decompose

from statsmodels.tsa.arima.model import ARIMA

# Load dataset

df = pd.read_csv('time_series_data.csv', parse_dates=['Date'], index_col='Date')



# Visualize the data

plt.figure(figsize=(10, 6))

plt.plot(df['Value'])

plt.title('Time Series Data')

plt.show()



# Decompose the time series
```

```
decomposition = seasonal_decompose(df['Value'], model='additive')

decomposition.plot()

plt.show()


# Fit ARIMA model

model = ARIMA(df['Value'], order=(5, 1, 0))

model_fit = model.fit()

print(model_fit.summary())


# Forecast future values

forecast = model_fit.forecast(steps=10)

plt.figure(figsize=(10, 6))

plt.plot(df['Value'], label='Historical')

plt.plot(forecast, label='Forecast')

plt.legend()

plt.show()
```

2. **Describe how to implement a basic recommender system in Python. Provide a step-by-step explanation and code example.**

Answer: Implementing a basic recommender system in Python involves several steps, from loading and preparing the data to training the model and making recommendations. Here's a step-by-step explanation with a code example:

**Loading the Data:**

Load the dataset which contains user-item interactions. For this example, we will use the MovieLens dataset which is built into the Surprise library.

```
from surprise import Dataset, Reader
```

```
import pandas as pd

# Load MovieLens dataset

data = Dataset.load_builtin('ml-100k')
```

**Preparing the Data:**

Convert the data into a format suitable for the recommender system library. The Surprise library handles this internally when loading built-in datasets.

**Selecting an Algorithm:**

Choose an algorithm for making recommendations. Here, we use a basic algorithm like Singular Value Decomposition (SVD).

```
from surprise import SVD

from surprise.model_selection import cross_validate

# Select SVD algorithm

algo = SVD()
```

**Training the Model:**

Train the model using the training data. The Surprise library allows for easy cross-validation to evaluate the model's performance.

```
# Train and evaluate the algorithm using cross-validation

cross_validate(algo, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
```

**Making Predictions:**

Use the trained model to make recommendations for a specific user. Predict ratings for items the user has not yet interacted with.

```
# Build a full training set

trainset = data.build_full_trainset()

# Train the algorithm on the full dataset

algo.fit(trainset)
```

```
# Predict ratings for a specific user (e.g., user ID 196)

user_id = 196

user_ratings = []

for item_id in trainset.all_items():

    inner_id = trainset.to_inner_iid(item_id)

    predicted_rating = algo.predict(user_id, item_id).est

    user_ratings.append((item_id, predicted_rating))

# Sort and recommend the top 10 items

user_ratings.sort(key=lambda x: x[1], reverse=True)

top_10_recommendations = user_ratings[:10]

print("Top 10 recommendations for user 196:")

for item_id, rating in top_10_recommendations:

    print(f"Item ID: {item_id}, Predicted Rating: {rating}")
```

3. **Explain the difference between linear regression and logistic regression. Provide examples of situations where each would be used.**

Answer: Linear regression and logistic regression are both supervised learning algorithms but are used for different types of problems:

**Linear Regression:** Used for predicting continuous values. It models the relationship between the dependent variable and one or more independent variables using a linear equation.

Example: Predicting house prices based on features like size, location, and number of rooms.

```
from sklearn.linear_model import LinearRegression

X = df[['size', 'location', 'rooms']]

y = df['price']

model = LinearRegression()
```

```
model.fit(X, y)

predictions = model.predict(X)
```

**Logistic Regression:** Used for binary classification problems. It models the probability of a binary outcome using a logistic function.

Example: Predicting whether a student will pass or fail based on study hours and attendance.

```
from sklearn.linear_model import LogisticRegression

X = df[['study_hours', 'attendance']]

y = df['pass']

model = LogisticRegression()

model.fit(X, y)

predictions = model.predict(X)
```

4. **Describe how decision trees work and their application in decision-making processes. Include a detailed example of building a decision tree model using Python.**

Answer: Decision trees are used for both classification and regression tasks. They work by recursively splitting the dataset into subsets based on the value of input features, resulting in a tree-like model of decisions.

**How they work:** Each internal node represents a feature, each branch represents a decision rule, and each leaf node represents an outcome.

**Applications:** Decision trees can be used for various decision-making processes, such as diagnosing medical conditions, determining loan approvals, and more.

**Example:**

```
from sklearn.datasets import load_iris

from sklearn.tree import DecisionTreeClassifier

from sklearn.model_selection import train_test_split

from sklearn import metrics
```

```
# Load dataset

iris = load_iris()

X = iris.data

y = iris.target

# Split the dataset

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)

# Create decision tree classifier

clf = DecisionTreeClassifier()

# Train the model

clf.fit
```