**Assignment: Chapter 1**

**1)What is AI? Differences between AI, ML, DL & DS.**

**=>**AI, or artificial intelligence, refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions. It encompasses a wide range of technologies and disciplines, including machine learning (ML), deep learning (DL), and data science (DS).



The difference between AI,ML,DL and DS are:-

A diagram of data science

Description automatically generated

| **Artificial Intelligence (AI)** | **Machine Learning (ML)** | **Deep Learning (DL)** |
| --- | --- | --- |
| AI simulates human intelligence to perform tasks and make decisions. | ML is a subset of AI that uses algorithms to learn patterns from data. | DL is a subset of ML that employs artificial neural networks for complex tasks. |
| AI may or may not require large datasets; it can use predefined rules. | ML heavily relies on labeled data for training and making predictions. | DL requires extensive labeled data and performs exceptionally with big datasets. |
| AI can be rule-based, requiring human programming and intervention. | ML automates learning from data and requires less manual intervention. | DL automates feature extraction, reducing the need for manual engineering. |
| AI can handle various tasks, from simple to complex, across domains. | ML specializes in data-driven tasks like classification, regression, etc. | DL excels at complex tasks like image recognition, natural language processing, and more. |
| AI algorithms can be simple or complex, depending on the application. | ML employs various algorithms like decision trees, SVM, and random forests. | DL relies on deep neural networks, which can have numerous hidden layers for complex learning. |
| AI may require less training time and resources for rule-based systems. | ML training time varies with the algorithm complexity and dataset size. | DL training demands substantial computational resources and time for deep networks. |
| AI systems may offer interpretable results based on human rules. | ML models can be interpretable or less interpretable based on the algorithm. | DL models are often considered less interpretable due to complex network architectures. |
| AI is used in virtual assistants, recommendation systems, and more. | ML is applied in image recognition, spam filtering, and other data tasks. | DL is utilized in autonomous vehicles, speech recognition, and advanced AI applications. |

Machine Learning (ML): ML is a subset of AI, focusing on developing algorithms that allow computers to learn patterns from data without being explicitly programmed. Supervised, unsupervised, and reinforcement learning are prominent ML paradigms.

Deep Learning (DL): DL is a subfield of ML that utilizes artificial neural networks, inspired by the human brain's structure, to process and learn from vast amounts of data. It has gained significant attention due to its breakthroughs in image recognition, natural language processing, and more.

Data Science (DS): DS involves the extraction of knowledge and insights from structured and unstructured data. It combines elements of statistics, computer science, and domain expertise to interpret and draw meaningful conclusions from data.

**2) What is the Turing test? Explain.**

**=>** The Turing Test is a method of inquiry in artificial intelligence (AI) for determining whether or not a computer is capable of thinking like a human being. The Turing test is a test of a machine's ability to exhibit intelligent behavior equivalent to, or indistinguishable from, that of a human. It was proposed by the British mathematician and computer scientist Alan Turing in 1950 as a way to measure a machine's ability to exhibit intelligent behavior.

In the Turing test, a human judge engages in a natural language conversation with a human and a machine, both of which try to convince the judge that they are the human. If the judge is unable to reliably distinguish between the human and the machine based on the conversation, then the machine is said to have passed the Turing test.

The Turing test is not without criticism, as some argue that it is a limited measure of intelligence and does not capture all aspects of human cognition. Nonetheless, it remains a landmark concept in the field of artificial intelligence and has sparked ongoing debate about the nature of intelligence and the capabilities of machines.

**3) What are the goals and challenges of AI?**

=> Goals of AI:

**1.Replicating human intelligence**: One of the primary goals of AI is to develop machines that can replicate human intelligence, including the ability to learn, reason, and solve problems.

**2.Automation**: AI aims to automate repetitive tasks, freeing up human workers to focus on more creative and strategic endeavors.

**3.Enhancing decision-making:** AI systems can analyze large amounts of data quickly and accurately, helping humans make more informed decisions in various fields such as healthcare, finance, and transportation.

**4.Improving efficiency:** AI can optimize processes and systems to improve efficiency and productivity in various industries, leading to cost savings and better resource allocation.

**5.Innovation:** AI drives innovation by enabling the development of new technologies and solutions that were previously thought to be impossible.

**6.Personalization:** AI can personalize experiences for users by analyzing their preferences and behaviors, leading to more tailored products and services.

**7.Advancing scientific research:** AI is used in scientific research to analyze complex data sets, simulate experiments, and make new discoveries in fields such as biology, astronomy, and chemistry.

**8.Assisting in dangerous tasks**: AI can be used in environments that are dangerous or inaccessible to humans, such as deep-sea exploration, space exploration, and disaster response.

**9.Improving healthcare:** AI has the potential to revolutionize healthcare by enabling early disease detection, personalized treatment plans, and better patient outcomes.

**10.Solving societal challenges:** AI can help address some of the world's most pressing challenges, such as climate change, poverty, and inequality, by providing insights and solutions that were previously unavailable.

Challenges of AI:

**1.Ethical concerns:** AI raises ethical concerns related to privacy, bias, job displacement, and the potential misuse of AI technologies.

**2.Bias and fairness:** AI systems can exhibit bias if they are trained on biased data, leading to unfair outcomes, particularly in areas such as hiring, lending, and law enforcement.

**3.Transparency and explainability:** AI systems can be complex and difficult to understand, making it challenging to explain their decisions and ensure accountability.

**4.Safety and reliability:** AI systems must be safe and reliable, especially in critical applications such as autonomous vehicles, healthcare, and finance.

**5.Regulatory challenges:** The rapid advancement of AI technology has outpaced regulatory frameworks, leading to challenges in ensuring that AI is developed and used responsibly.

**6.Job displacement:** AI has the potential to automate many jobs, leading to concerns about unemployment and the need for retraining and reskilling workers.

**7.Security risks**: AI systems can be vulnerable to attacks and manipulation, leading to concerns about data security and privacy.

**8.Over-reliance on AI:** There is a risk of over-reliance on AI systems, leading to a loss of human skills and judgment.

**9.Societal impact:** AI can have wide-ranging societal impacts, including changes in social dynamics, inequality, and the concentration of power.

**10.Technical limitations:** Despite advances, AI still faces technical limitations, such as the ability to understand context, common sense reasoning, and learning from limited data.

**4) List the AI approaches and the AI techniques and explain it.**

**=>**AI Approaches:

1. Symbolic AI: Symbolic AI, also known as classical AI or GOFAI (Good Old-Fashioned Artificial Intelligence), focuses on the manipulation of symbols and rules to simulate human reasoning. It involves the use of logic, knowledge representation, and symbolic reasoning to solve problems.
2. Connectionist AI: Connectionist AI, also known as neural networks or connectionism, is inspired by the structure and function of the human brain. It involves the use of artificial neural networks to simulate the way neurons in the brain process information and learn from data.
3. Evolutionary AI: Evolutionary AI is based on the principles of Darwinian evolution and natural selection. It involves the use of evolutionary algorithms to generate solutions to complex problems by simulating the process of evolution over multiple generations.
4. Bayesian AI: Bayesian AI is based on the principles of Bayesian probability theory. It involves the use of Bayesian inference to update beliefs and make decisions based on uncertain or incomplete information.
5. Analogical AI: Analogical AI is based on the use of analogies to solve problems. It involves the use of analogical reasoning to transfer knowledge from one domain to another and apply it to new situations.

AI Techniques:

1. Machine Learning (ML): Machine learning is a subset of AI that involves the development of algorithms and models that allow computers to learn from and make predictions or decisions based on data. It includes techniques such as supervised learning, unsupervised learning, and reinforcement learning.
2. Deep Learning (DL): Deep learning is a subset of machine learning that uses artificial neural networks to model and solve complex problems. It is particularly effective for tasks such as image and speech recognition, natural language processing, and autonomous driving.
3. Natural Language Processing (NLP): Natural language processing is a branch of AI that focuses on the interaction between computers and humans using natural language. It involves techniques such as text analysis, sentiment analysis, and machine translation.
4. Computer Vision: Computer vision is a branch of AI that focuses on enabling computers to interpret and understand visual information from the real world. It involves techniques such as image recognition, object detection, and image segmentation.
5. Reinforcement Learning: Reinforcement learning is a machine learning technique that involves training agents to make decisions in an environment to achieve a goal. It involves the use of rewards and penalties to encourage desirable behavior.

These approaches and techniques represent different ways in which AI can be applied to solve problems and achieve intelligent behavior. They can be used individually or in combination to develop AI systems for various applications.

**5) Application of AI with 15 points and Explanations.**

**=>**Application of AI are given below:-

1) Healthcare: AI is used in healthcare for medical imaging analysis, personalized treatment plans, drug discovery, and virtual health assistants.

2) Finance: AI is used in finance for fraud detection, algorithmic trading, credit scoring, and customer service chatbots.

3) Automotive: AI is used in automotive for autonomous driving, predictive maintenance, driver behavior analysis, and smart navigation.

4) Retail: AI is used in retail for personalized marketing, demand forecasting, inventory management, and recommendation systems.

5) Manufacturing: AI is used in manufacturing for predictive maintenance, quality control, supply chain optimization, and process automation.

6) Education: AI is used in education for personalized learning, student performance analysis, automated grading, and virtual tutoring.

7) Agriculture: AI is used in agriculture for crop monitoring, yield prediction, pest detection, and smart irrigation.

8) Telecommunications: AI is used in telecommunications for network optimization, customer service chatbots, fraud detection, and predictive maintenance.

9) Energy: AI is used in energy for power grid optimization, predictive maintenance of infrastructure, energy consumption forecasting, and renewable energy integration.

10) Transportation: AI is used in transportation for route optimization, traffic management, predictive maintenance of vehicles, and autonomous vehicles.

11) Marketing: AI is used in marketing for customer segmentation, personalized advertising, sentiment analysis, and campaign optimization.

12) Entertainment: AI is used in entertainment for content recommendation, personalized playlists, video and audio analysis, and virtual reality experiences.

13) Security: AI is used in security for facial recognition, behavior analysis, threat detection, and anomaly detection.

14) Environmental Monitoring: AI is used in environmental monitoring for climate modeling, pollution detection, wildlife tracking, and disaster prediction.

15) Human Resources: AI is used in human resources for resume screening, candidate sourcing, employee performance analysis, and HR chatbots.

These are just a few examples of how AI is being applied across various industries to improve efficiency, decision-making, and customer experiences.**===>**

**6) Discuss one ethical consideration associated with the deployment of AI systems in society**.

**=>**One ethical consideration associated with the deployment of AI systems in society is the issue of bias. AI systems are trained on large datasets, and if these datasets contain biases, the AI system can learn and perpetuate those biases. This can lead to discriminatory outcomes, particularly in areas such as hiring, lending, and law enforcement.

For example, if an AI system is used to screen job applicants and is trained on historical data that reflects gender or racial biases in hiring practices, the AI system may inadvertently discriminate against certain groups of people. Similarly, in the criminal justice system, if an AI system is used to predict recidivism rates based on biased historical data, it may result in unfair sentencing outcomes for certain individuals.

Addressing bias in AI systems requires careful consideration of the data used to train these systems, as well as the algorithms and decision-making processes involved. Ethical AI practices involve ensuring that datasets are representative and unbiased, and that AI systems are designed to be transparent and accountable for their decisions.

**7) What distinguishes Artificial Intelligence from traditional computer programming?**

=>Artificial intelligence (AI) differs from traditional computer programming in several key ways:

1. **Flexibility and Adaptability:** Traditional programming involves writing explicit instructions for a computer to follow. In contrast, AI systems can adapt and learn from data, allowing them to improve their performance over time without being explicitly programmed.
2. **Complexity of Problem Solving:** AI is used to tackle complex problems that are difficult to solve using traditional programming techniques. AI systems can handle tasks such as natural language processing, image recognition, and decision-making in dynamic environments.
3. **Data Dependency:** AI systems rely heavily on data to learn and make decisions. Traditional programming may also use data, but AI algorithms are specifically designed to analyze and learn from large datasets.
4. **Decision Making:** AI systems can make decisions based on probabilities and uncertainties, while traditional programming typically follows a deterministic approach.
5. **Learning Capability:** AI systems have the ability to learn from experience and improve their performance over time, whereas traditional programs execute predefined instructions without learning from data.
6. **Human-like Intelligence:** AI aims to replicate human-like intelligence, such as reasoning, problem-solving, learning, and perception, which are not typically addressed in traditional programming.

Overall, AI represents a shift from traditional programming paradigms by focusing on creating systems that can simulate human-like intelligence and behavior.

**Assignment1 : Titanic DataSets**

**1. Exploratory Data Analysis (EDA):**

**● Objective: Understand the basic characteristics of the dataset.**

**● Tasks:**

**○ Describe the dataset: number of rows, columns, data types.**

**○ Summarize numerical features: mean, median, mode, range, standard deviation.**

**○ Explore categorical features: frequency distribution, unique values.**

**○ Visualize distributions: histograms, box plots.**

**○ Identify outliers and handle them appropriately.**

**ANS:-**Exploratory Data Analysis (EDA) is a crucial step in understanding a dataset. It helps in uncovering patterns, relationships, and anomalies, which can guide further analysis and modeling. Here's a breakdown of the tasks involved in EDA:

1. **Describe the dataset**: Start by understanding the basic structure of the dataset. This includes the number of rows, columns, and data types (e.g., numerical, categorical).
2. **Summarize numerical features**: Calculate summary statistics for numerical features, such as mean, median, mode, range, and standard deviation. These statistics provide insights into the central tendency, spread, and shape of the data distribution.
3. **Explore categorical features**: For categorical features, analyze the frequency distribution to understand the distribution of unique values and their frequencies.
4. **Visualize distributions**: Use histograms and box plots to visualize the distributions of numerical features. Histograms show the frequency distribution of a continuous variable, while box plots provide a visual summary of the data distribution, including outliers.
5. **Identify outliers**: Outliers are data points that significantly deviate from the rest of the data. They can distort statistical analyses and machine learning models. Identify outliers using visualization techniques or statistical methods (e.g., z-score, IQR).
6. **Handle outliers**: Depending on the nature of the outliers, you can choose to remove them, transform them, or treat them as missing values. Be cautious when handling outliers, as their removal or manipulation can impact the analysis results.

Overall, EDA provides a comprehensive overview of the dataset, enabling data scientists to make informed decisions and derive meaningful insights.

Code:

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load the Titanic dataset

url = 'https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv'

titanic = pd.read\_csv(url)

# Describe the dataset

print("Number of rows and columns:", titanic.shape)

print("\nData types of columns:")

print(titanic.dtypes)

# Summarize numerical features

numerical\_summary = titanic.describe()

print("\nSummary of numerical features:")

print(numerical\_summary)

# Explore categorical features

categorical\_summary = titanic.describe(include=['O'])

print("\nSummary of categorical features:")

print(categorical\_summary)

# Visualize distributions

sns.set(style="whitegrid")

plt.figure(figsize=(12, 6))

plt.subplot(2, 2, 1)

sns.histplot(data=titanic, x='Age', bins=20, kde=True)

plt.title('Distribution of Age')

plt.subplot(2, 2, 2)

sns.boxplot(data=titanic, y='Fare')

plt.title('Boxplot of Fare')

plt.subplot(2, 2, 3)

sns.countplot(data=titanic, x='Survived', hue='Sex')

plt.title('Survival Count by Gender')

plt.subplot(2, 2, 4)

sns.countplot(data=titanic, x='Survived', hue='Pclass')

plt.title('Survival Count by Passenger Class')

plt.tight\_layout()

plt.show()

# Identify and handle outliers (for example, for 'Fare' column)

q1 = titanic['Fare'].quantile(0.25)

q3 = titanic['Fare'].quantile(0.75)

iqr = q3 - q1

upper\_bound = q3 + 1.5 \* iqr

outliers = titanic[titanic['Fare'] > upper\_bound]

print("\nNumber of outliers in 'Fare' column:", len(outliers))

**2. Handling Missing Values:**

**● Objective: Deal with missing data to ensure robust analysis.**

**● Tasks:**

**○ Identify missing values: count missing values per column.**

**○ Evaluate the impact of missing data: analyze patterns, reasons for missingness.**

**○ Choose appropriate strategies for handling missing values: imputation, deletion.**

**○ Implement chosen strategies: impute missing values using mean, median, mode,**

**or advanced methods like regression.**

**○ Validate the effectiveness of the chosen strategy: compare before and after results.**

**ANS:-**To handle missing values in the Titanic dataset, you can follow these steps:

1. Identify missing values: Count missing values per column using the **isnull()** method followed by **sum()**.
2. Evaluate the impact of missing data: Analyze patterns and reasons for missingness using visualizations or data exploration techniques.
3. Choose appropriate strategies for handling missing values: Decide whether to impute missing values or delete rows/columns with missing values based on the analysis.
4. Implement chosen strategies: Impute missing values using mean, median, mode, or advanced methods like regression. For simplicity, let's impute missing values in the 'Age' column with the median age.
5. Validate the effectiveness of the chosen strategy: Compare the dataset before and after imputation to ensure missing values have been properly handled.

Code:

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load the Titanic dataset

url = 'https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv'

titanic = pd.read\_csv(url)

# Identify missing values

missing\_values = titanic.isnull().sum()

print("Missing values per column:")

print(missing\_values)

# Evaluate the impact of missing data

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

sns.heatmap(titanic.isnull(), cbar=False, cmap='viridis')

plt.title('Missing Values Heatmap')

plt.show()

# Choose appropriate strategies for handling missing values

# For simplicity, let's impute missing values in the 'Age' column with the median age

median\_age = titanic['Age'].median()

titanic['Age'] = titanic['Age'].fillna(median\_age)

# Validate the effectiveness of the chosen strategy

print("\nNumber of missing values in 'Age' column after imputation:", titanic['Age'].isnull().sum())

**3. Dealing with Duplicate Values:**

**● Objective: Identify and handle duplicate records.**

**● Tasks:**

**○ Detect duplicate rows: identify rows with identical values across all columns.**

**○ Analyze the impact of duplicated values: assess their frequency and distribution.**

**○ Decide on the treatment strategy: remove duplicates or keep one instance.**

**○ Implement the chosen strategy: remove duplicates based on specific criteria, such as ticket number or passenger name.**

**○ Validate the effectiveness of duplicate removal: assess the impact on data integrity.**

**ANS:-**To handle duplicate values in the Titanic dataset, you can follow these steps:

1. Detect duplicate rows: Identify rows with identical values across all columns using the **duplicated()** method.
2. Analyze the impact of duplicated values: Assess their frequency and distribution to understand the extent of duplication.
3. Decide on the treatment strategy: Choose whether to remove duplicates or keep one instance of each duplicate row.
4. Implement the chosen strategy: Remove duplicates based on specific criteria, such as ticket number or passenger name.
5. Validate the effectiveness of duplicate removal: Assess the impact on data integrity by comparing the dataset before and after duplicate removal.

Code:

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load the Titanic dataset

url = 'https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv'

titanic = pd.read\_csv(url)

# Exploratory Data Analysis (EDA)

print("Number of rows and columns:", titanic.shape)

print("\nData types of columns:")

print(titanic.dtypes)

# Summarize numerical features

numerical\_summary = titanic.describe()

print("\nSummary of numerical features:")

print(numerical\_summary)

# Explore categorical features

categorical\_summary = titanic.describe(include=['O'])

print("\nSummary of categorical features:")

print(categorical\_summary)

# Visualize distributions

sns.set(style="whitegrid")

plt.figure(figsize=(12, 6))

plt.subplot(2, 2, 1)

sns.histplot(data=titanic, x='Age', bins=20, kde=True)

plt.title('Distribution of Age')

plt.subplot(2, 2, 2)

sns.boxplot(data=titanic, y='Fare')

plt.title('Boxplot of Fare')

plt.subplot(2, 2, 3)

sns.countplot(data=titanic, x='Survived', hue='Sex')

plt.title('Survival Count by Gender')

plt.subplot(2, 2, 4)

sns.countplot(data=titanic, x='Survived', hue='Pclass')

plt.title('Survival Count by Passenger Class')

plt.tight\_layout()

plt.show()

# Handling Missing Values

missing\_values = titanic.isnull().sum()

print("Missing values per column:")

print(missing\_values)

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

sns.heatmap(titanic.isnull(), cbar=False, cmap='viridis')

plt.title('Missing Values Heatmap')

plt.show()

median\_age = titanic['Age'].median()

titanic['Age'] = titanic['Age'].fillna(median\_age)

print("\nNumber of missing values in 'Age' column after imputation:", titanic['Age'].isnull().sum())

# Dealing with Duplicate Values

duplicate\_rows = titanic[titanic.duplicated()]

print("Number of duplicate rows:", len(duplicate\_rows))

duplicate\_counts = titanic.duplicated().sum()

print("\nNumber of duplicated records:", duplicate\_counts)

titanic\_unique = titanic.drop\_duplicates()

print("\nNumber of rows before duplicate removal:", len(titanic))

print("Number of rows after duplicate removal:", len(titanic\_unique))

**Assignment 2: Iris Datasets**

**1. Exploratory Data Analysis (EDA):**

**● Objective: Understand the basic characteristics of the dataset.**

**● Tasks:**

**○ Describe the dataset: number of rows, columns, data types.**

**○ Summarize numerical features: mean, median, mode, range, standard deviation.**

**○ Explore categorical features: frequency distribution, unique values (though there**

**may not be categorical features in Iris).**

**○ Visualize distributions: histograms, box plots, scatter plots (especially useful for**

**comparing features).**

**○ Identify any patterns or relationships between features.**

**ANS:-** import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load the Iris dataset

url = 'https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/data/iris.csv'

iris = pd.read\_csv(url)

# Describe the dataset: number of rows, columns, data types

print("Number of rows and columns:", iris.shape)

print("\nData types of columns:")

print(iris.dtypes)

# Summarize numerical features: mean, median, mode, range, standard deviation

numerical\_summary = iris.describe()

print("\nSummary of numerical features:")

print(numerical\_summary)

# Explore categorical features: frequency distribution, unique values

# Note: Iris dataset does not have categorical features, so skipping this step

# Visualize distributions: histograms, box plots, scatter plots

plt.figure(figsize=(12, 6))

plt.subplot(2, 2, 1)

sns.histplot(data=iris, x='sepal\_length', bins=20, kde=True)

plt.title('Distribution of Sepal Length')

plt.subplot(2, 2, 2)

sns.boxplot(data=iris, y='petal\_width')

plt.title('Boxplot of Petal Width')

plt.subplot(2, 2, 3)

sns.scatterplot(data=iris, x='sepal\_length', y='sepal\_width', hue='species')

plt.title('Scatter Plot of Sepal Length vs Sepal Width')

plt.subplot(2, 2, 4)

sns.scatterplot(data=iris, x='petal\_length', y='petal\_width', hue='species')

plt.title('Scatter Plot of Petal Length vs Petal Width')

plt.tight\_layout()

plt.show()

# Identify any patterns or relationships between features

sns.pairplot(iris, hue='species')

plt.title('Pairplot of Iris Dataset')

plt.show()

**2. Handling Missing Values (if applicable):**

**● Objective: Check for missing data and handle them if present.**

**● Tasks:**

**○ Identify missing values: count missing values per column.**

**○ Decide on the treatment strategy: impute missing values or remove them.**

**○ Implement chosen strategy: impute missing values using mean, median, mode, or advanced methods.**

**○ Validate the effectiveness of the chosen strategy: compare before and after results.**

**ANS:-** The Iris dataset does not contain missing values, we can simulate this scenario by randomly introducing missing values into the dataset.

import pandas as pd

import numpy as np

# Load the Iris dataset

url = 'https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/data/iris.csv'

iris = pd.read\_csv(url)

# Introduce missing values

iris\_missing = iris.mask(np.random.random(iris.shape) < 0.1)

# Identify missing values

missing\_values = iris\_missing.isnull().sum()

print("Missing values per column:")

print(missing\_values)

# Decide on the treatment strategy: impute missing values with the mean

iris\_imputed = iris\_missing.fillna(iris\_missing.mean())

# Validate the effectiveness of the chosen strategy

missing\_values\_after\_imputation = iris\_imputed.isnull().sum()

print("\nMissing values per column after imputation:")

print(missing\_values\_after\_imputation**)**

**3. Dealing with Duplicate Values (if applicable):**

**● Objective: Identify and handle duplicate records.**

**● Tasks:**

**○ Detect duplicate rows: identify rows with identical values across all columns.**

**○ Decide on the treatment strategy: remove duplicates or keep one instance.**

**○ Implement the chosen strategy: remove duplicates based on specific criteria.**

**○ Validate the effectiveness of duplicate removal: assess the impact on data**

**integrity.**

**ANS:-** import pandas as pd

# Load the Iris dataset

url = 'https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/data/iris.csv'

iris = pd.read\_csv(url)

# Introduce duplicate rows

iris\_duplicates = pd.concat([iris, iris.head(10)], ignore\_index=True)

# Detect duplicate rows

duplicate\_rows = iris\_duplicates[iris\_duplicates.duplicated()]

print("Number of duplicate rows:", len(duplicate\_rows))

# Decide on the treatment strategy: remove duplicates

iris\_unique = iris\_duplicates.drop\_duplicates()

# Validate the effectiveness of the chosen strategy

print("\nNumber of rows before duplicate removal:", len(iris\_duplicates))

print("Number of rows after duplicate removal:", len(iris\_unique))

**Assignment 3: Dataset: Wine Quality**

1. Exploratory Data Analysis (EDA):

● Objective: Understand the basic characteristics of the dataset.

● Tasks:

○ Describe the dataset: number of rows, columns, data types.

○ Summarize numerical features: mean, median, mode, range, standard deviation.

○ Explore categorical features (if any): frequency distribution, unique values.

○ Visualize distributions: histograms, box plots, correlation matrix.

○ Identify any patterns or relationships between features and quality ratings.

ANS:- **Load the dataset and describe it**:

import pandas as pd

# Load the Wine Quality dataset

url\_red = 'https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv'

url\_white = 'https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv'

wine\_red = pd.read\_csv(url\_red, sep=';')

wine\_white = pd.read\_csv(url\_white, sep=';')

# Combine red and white wine datasets

wine = pd.concat([wine\_red, wine\_white])

# Describe the dataset

print("Number of rows and columns:", wine.shape)

print("\nData types of columns:")

print(wine.dtypes)

**Summarize numerical features**:

numerical\_summary = wine.describe()

print("\nSummary of numerical features:")

print(numerical\_summary)

**Explore categorical features (if any)**:

categorical\_summary = wine.describe(include=['O'])

print("\nSummary of categorical features:")

print(categorical\_summary)

**Visualize distributions**:

* Histograms:

import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(12, 6))

sns.histplot(data=wine, x='quality', bins=7, kde=True)

plt.title('Distribution of Wine Quality Ratings')

plt.show()

Box plots:

plt.figure(figsize=(12, 6))

sns.boxplot(data=wine[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar', 'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density', 'pH', 'sulphates', 'alcohol']])

plt.title('Boxplot of Chemical Properties')

plt.xticks(rotation=45)

plt.show()

Correlation matrix:

corr\_matrix = wine.corr()

plt.figure(figsize=(12, 6))

sns.heatmap(corr\_matrix, annot=True, cmap='coolwarm')

plt.title('Correlation Matrix')

plt.show()

**Identify patterns or relationships between features and quality ratings**:

* Scatter plots:

plt.figure(figsize=(12, 6))

sns.scatterplot(data=wine, x='alcohol', y='quality', hue='type')

plt.title('Scatter Plot of Alcohol Content vs Quality')

plt.show()

**2. Data Preprocessing:**

**● Objective: Prepare the data for analysis and modeling.**

**● Tasks:**

**○ Handle missing values: check for missing data and decide on the treatment**

**strategy (imputation or removal).**

**○ Encode categorical features (if any): convert categorical variables into numerical**

**format using techniques like one-hot encoding.**

**○ Scale numerical features: normalize or standardize numerical features to ensure**

**all features have the same scale.**

**○ Split the dataset into training and testing sets for modeling.**

**ANS:-**

**Handle Missing Values**:

Check for missing data:

missing\_values = wine.isnull().sum()

print("Missing values per column:")

print(missing\_values)

**Encode Categorical Features** (if any):

* Convert categorical variables into numerical format using one-hot encoding:

wine\_encoded = pd.get\_dummies(wine)

**Scale Numerical Features**:

* Normalize or standardize numerical features:

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

wine\_scaled = scaler.fit\_transform(wine\_encoded)

**Split the Dataset** into Training and Testing Sets:

* Split the dataset into input features (X) and target variable (y):

X = wine\_scaled.drop('quality', axis=1)

y = wine\_scaled['quality']

# Split the dataset into training and testing sets

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)

**3. Feature Engineering:**

**● Objective: Create new features or transform existing ones to improve model performance.**

**● Tasks:**

**○ Create new features based on domain knowledge or interaction between existing features.**

**○ Perform feature selection: identify the most relevant features using techniques like correlation analysis or feature importance.**

**ANS:-**

**Create New Features**:

* Example: Create a new feature representing the total amount of sulfur dioxide (sum of free and total sulfur dioxide):

wine['total\_sulfur\_dioxide'] = wine['free sulfur dioxide'] + wine['total sulfur dioxide']

**Perform Feature Selection**:

* Identify the most relevant features using correlation analysis:

corr\_matrix = wine.corr()

relevant\_features = corr\_matrix['quality'].sort\_values(ascending=False).index[1:4]

* Identify feature importance using a machine learning model (e.g., Random Forest):

from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor()

rf.fit(X\_train, y\_train)

feature\_importances = pd.Series(rf.feature\_importances\_, index=X.columns)

relevant\_features = feature\_importances.sort\_values(ascending=False).head(3).index

**4. Modeling:**

**● Objective: Build predictive models to predict wine quality.**

**● Tasks:**

**○ Choose appropriate machine learning algorithms for classification or regression**

**(e.g., Decision Trees, Random Forest, Support Vector Machines).**

**○ Train the models using the training dataset.**

**○ Evaluate model performance using appropriate metrics (e.g., accuracy, F1-score,RMSE).**

**○ Fine-tune hyperparameters to optimize model performance (optional).**

**ANS:-**

1. **Choose Machine Learning Algorithms**:
   * For regression (predicting wine quality), you can use algorithms like Decision Trees, Random Forest, or Support Vector Machines (SVM).
2. **Train the Models**:
   * Train the selected algorithms using the training dataset (X\_train, y\_train).
3. **Evaluate Model Performance**:
   * Evaluate the trained models using appropriate metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), or R-squared (R2) score for regression.
4. **Fine-Tune Hyperparameters** (Optional):
   * Use techniques like Grid Search or Random Search to fine-tune hyperparameters and optimize model performance.

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_squared\_error, r2\_score

# Initialize the model

rf = RandomForestRegressor()

# Train the model

rf.fit(X\_train, y\_train)

# Make predictions

y\_pred = rf.predict(X\_test)

# Evaluate model performance

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = np.sqrt(mse)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error:", mse)

print("Root Mean Squared Error:", rmse)

print("R-squared:", r2)

**5. Model Evaluation and Interpretation:**

**● Objective: Assess the performance of the trained models and interpret the results.**

**● Tasks:**

**○ Evaluate model performance on the test dataset.**

**○ Interpret the model predictions: analyze important features, identify strengths and**

**weaknesses of the models.**

**○ Visualize model predictions: plot predicted vs. actual quality ratings, ROC curves**

**(for classification models), etc.**

**○ Discuss the implications of the model results and potential real-world**

**applications.**

**ANS:-**

1. **Evaluate Model Performance**:
   * Use the test dataset to evaluate the performance of the trained models using metrics like Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2) score.
2. **Interpret Model Predictions**:
   * Analyze important features using feature importance scores from the trained models to understand which features have the most impact on predicting wine quality.
   * Identify strengths and weaknesses of the models based on their performance metrics and feature importance.
3. **Visualize Model Predictions**:
   * Plot predicted vs. actual quality ratings to visualize how well the models are predicting the wine quality.
   * For classification models, you can also plot ROC curves and calculate the Area Under the Curve (AUC) to visualize the model's performance.
4. **Discuss Implications**:
   * Discuss the implications of the model results in terms of their real-world applications. For example, how can the models be used by winemakers to improve wine quality or by consumers to choose wines based on their preferences?

# Evaluate model performance on the test dataset

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = np.sqrt(mse)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error:", mse)

print("Root Mean Squared Error:", rmse)

print("R-squared:", r2)

# Analyze important features

feature\_importances = pd.Series(rf.feature\_importances\_, index=X.columns).sort\_values(ascending=False)

print("\nTop 5 important features:")

print(feature\_importances.head(5))

# Visualize model predictions

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

plt.scatter(y\_test, y\_pred)

plt.xlabel('Actual Quality')

plt.ylabel('Predicted Quality')

plt.title('Actual vs. Predicted Wine Quality')

plt.show()