Year 2023

Group B

13 questions:

a) How can we handle outliers and missing values in a dataset?

Ans:

Why do we need to treat outliers?

Outliers can lead to vague or misleading predictions while using machine learning models. Specific models like linear regression, logistic regression, and support vector machines are susceptible to outliers. Outliers decrease the mathematical power of these models, and thus the output of the models becomes unreliable. However, outliers are highly subjective to the dataset. Some outliers may portray extreme changes in the data as well.

https://www.geeksforgeeks.org/ml-handling-missing-values/

b) What is Machine Learning? How it is related with Artificial Intelligence. What is the goal of Machine Learning?

Ans:

b)What is Machine Learning?

Machine Learning (ML) is a subset of **Artificial Intelligence (AI)** that focuses on developing algorithms and models that allow computers to learn patterns from data and make decisions or predictions **without being explicitly programmed** for each specific task.

Instead of following strict, hand-coded rules, a machine learning system identifies patterns and relationships within data and uses that understanding to generate outputs or improve its performance over time.

How is Machine Learning Related to Artificial Intelligence?

Machine Learning is a **core branch** of Artificial Intelligence.

- **Artificial Intelligence (AI)** is the broader concept of machines being able to carry out tasks in a way that we would consider “smart.”
- **Machine Learning** is a technique for achieving AI — by giving machines access to data and letting them learn for themselves.

Think of it this way:

AI is the umbrella, and ML is one of its most powerful tools.

Other components of AI include:

- Natural Language Processing (NLP)
- Robotics
- Expert Systems
- Computer Vision

What is the Goal of Machine Learning?

The **main goal of Machine Learning** is to enable machines to:

1. **Automatically learn from data**
2. **Improve their performance over time** without human intervention
3. **Make accurate predictions or decisions** when faced with new, unseen data

In simple terms:

The goal is to build systems that can generalize well from examples and perform intelligently in real-world tasks.

c) **Compare and contrast Supervised and Unsupervised learning algorithms with examples**.

Ans: https://www.geeksforgeeks.org/supervised-unsupervised-learning/

17. What is maximal margin classifier? Explain with diagram.

Ans:https://ywu120766.medium.com/support-vector-machine-in-r-hyperplane-and-maximal-margin-classifier-part-01-21c3f93f751d

Group c

18

a) **How does Reinforced Learning work in Machine Learning?**

Ans: https://www.geeksforgeeks.org/what-is-reinforcement-learning/

b) **Explain overfitting problem in regression analysis.**

Ans:

https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

19

a) What is the role of distance measures in machine learning? List various distance-measuring techniques and explain any one of them.

Ans:

1. https://www.geeksforgeeks.org/measures-of-distance-in-data-mining/
2. Distance-measuring techniques can be broadly classified into direct and indirect methods. Direct methods physically measure the distance using tools like tapes, chains, or odometers. Indirect methods use calculations based on angles, elevation differences, or time-of-flight measurements. One example is time-of-flight (ToF), which measures distance by calculating the time it takes for a signal (like light or sound) to travel to an object and back.
3. Direct Distance Measuring Techniques:
4. **Pacing:** Estimating distance by counting paces.
5. **Chaining:** Using a chain or steel tape to measure distance on the ground. This is a highly accurate method often used in surveying.
6. **Odometer:** A device that measures distance by tracking wheel revolutions.
7. **Measuring Tapes:** Flexible tapes used for measuring distances.
8. **Measuring Rods:** Specifically designed for measuring the widths of openings.
9. **Calipers and Micrometers:** Used for precise measurements of both external and internal dimensions.
10.

b) State and explain the steps of k-Nearest Neighbours (k-NN) algorithm

Ans: https://www.geeksforgeeks.org/k-nearest-neighbours/

21) a. State and explain k-mean clustering algorithm.

Ans : https://www.geeksforgeeks.org/k-means-clustering-introduction/

22. Write short notes (any three) of the following

Ans:

(a) MLP: https://www.geeksforgeeks.org/multi-layer-perceptron-learning-in-tensorflow/
(b) SVM: https://www.geeksforgeeks.org/support-vector-machine-algorithm/
(c) HMM: https://www.geeksforgeeks.org/hidden-markov-model-in-machine-learning/
(d) Text representations in NLP: https://www.scaler.com/topics/nlp/text-representation-in-nlp/
(e) Agglomerative Hierarchical Clustering : https://www.geeksforgeeks.org/agglomerative-methods-in-machine-learning/

Year 2024

Group B

2.Compare and contrast Supervised and Unsupervised learning algorithms with examples.

Ans: https://www.geeksforgeeks.org/supervised-unsupervised-learning/

3.Explain bias-variance tradeoff.What is the purpose of using cross validation.

Ans: i). https://www.geeksforgeeks.org/ml-bias-variance-trade-off/

ii). https://www.geeksforgeeks.org/cross-validation-machine-learning/

4.explain logistic regression with example(trough sigmoid curve and maximum likihood function).is it a type of binary classification?

Ans:

i)https://www.geeksforgeeks.org/understanding-logistic-regression/

ii) Yes, **logistic regression is a type of binary classification** algorithm. It is used to predict the probability of an instance belonging to one of two classes (e.g., 0 or 1, yes or no, true or false). The output of the model is a probability value between 0 and 1, which is converted into a binary class using a threshold (usually 0.5).

4. What is maximum entropy?Describe its utility in mechine learning?

Ans: **Maximum Entropy** is a principle from information theory that suggests selecting the probability distribution with the **highest entropy** (i.e., the most uncertainty or least bias) among all distributions that satisfy certain constraints. Entropy, in this context, is a measure of randomness or unpredictability.

Definition:

The **Maximum Entropy Principle** states:

"Of all the probability distributions that satisfy given constraints (like known averages), choose the one with the maximum entropy."

Mathematically, for a discrete probability distribution P(x)P(x)P(x), entropy is defined as:

H(P)=−∑xP(x)logP(x)H(P) = -\sum_x P(x) \log P(x)H(P)=−x∑P(x)logP(x)

Utility in Machine Learning:

i)Classification (Maximum Entropy Classifier / Logistic Regression):

- The most direct application is in **Maximum Entropy Models**, which are essentially the same as **multinomial logistic regression**.
- It models the conditional probability P(y|x)P(y∣x)P(y|x) by maximizing entropy subject to the constraint that the expected values of features match their empirical values.

- This ensures the model makes the **least assumptive prediction**, consistent with observed data.

ii)Natural Language Processing (NLP):

- Widely used for **part-of-speech tagging**, **parsing**, **text classification**, and **information extraction**.
- Allows the model to learn from text data while avoiding unwarranted assumptions about the data distribution.

iii)Regularization and Generalization:

- Encourages models to be **less confident** in the absence of strong evidence (i.e., not overfit).
- Particularly useful when dealing with **sparse or incomplete data**.

iV)Probabilistic Reasoning:

- In scenarios where only partial knowledge (e.g., expected values of features) is known, maximum entropy provides the **most unbiased distribution** consistent with that knowledge.

5)**Mention How can you choose classifier base in training set size?**

Ans: https://www.tutorialspoint.com/choosing-a-classifier-based-on-a-training-set-data-size

Group C

7.b)**Advantages of tree models over linear model**.

Ans: Tree-based models (like Decision Trees, Random Forests, and Gradient Boosted Trees) have several advantages over linear models (like Linear Regression or Logistic Regression), especially when dealing with complex, real-world data. Here are the key advantages:

---

i. Capture Non-linear Relationships

- **Tree models** can model non-linear and complex interactions between features naturally.

- **Linear models** assume a linear relationship between input features and the output, which often oversimplifies real-world patterns.

---

ii. Handle Feature Interactions Automatically

- **Trees** can inherently capture interactions between features without manually adding interaction terms.
- **Linear models** require manual creation of interaction terms to capture such relationships.

---

iii. Less Need for Feature Scaling or Preprocessing

- **Tree models** are not sensitive to the scale or distribution of input features (no need for normalization/standardization).
- **Linear models** often require feature scaling and careful preprocessing.

---

iv. Handle Missing Values and Outliers Better

- **Trees** can handle missing values and are robust to outliers.
- **Linear models** can be significantly affected by outliers and missing data.

---

v. Work Well with Categorical Features

- **Decision Trees** and some tree-based models can handle categorical variables directly (depending on implementation).
- **Linear models** require encoding (like one-hot), which can increase dimensionality and reduce interpretability.

---

vi. Interpretability (for simple trees)

- **Small Decision Trees** can be visualized and interpreted easily, showing how decisions are made.
- **Linear models** are interpretable too, but only if the feature relationships are truly linear and non-interacting.

vii. No Assumptions About Data Distribution

- **Tree models** are non-parametric and make no assumptions about the underlying data distribution.
- **Linear models** often assume things like normality, homoscedasticity, and independence.

7)c.**what are the different impurity measures used in decision tree algorithm.**

Ans: https://www.geeksforgeeks.org/gini-impurity-and-entropy-in-decision-tree-ml/

8)b.**How can we evalute a regression model**?

Ans: https://www.geeksforgeeks.org/regression-metrics/

c.**compare and contrast L1 and L2 regularization techniques.**

Ans: https://www.tutorialspoint.com/difference-between-l1-and-l2-regularization

9)b.what is the multicollinneary in the context of multiple linear rgrassion?How can multicollinearity be detected,and what are the consequences of muticollinearity on regression analysis?

Ans**: https://www.geeksforgeeks.org/multicollinearity-in-regression-analysis/**

c.What is the difference between Type I and Type II errors in logistic regression?

Ans: https://www.geeksforgeeks.org/type-i-and-type-ii-errors/

10)b)What are the asuumption used in decision tree?

Ans: The assumptions used in a **Decision Tree** algorithm are:

1. **Feature-Target Relationship**:
   It assumes that the **attributes (features)** are sufficient to determine the output (target), meaning there is a relationship between input features and the target.

2. **Attribute Independence**:
   It assumes that **attributes are independent** of each other, i.e., there is no multicollinearity (although it can still work if attributes are correlated).

3. **Recursive Partitioning**:
   The data can be **split recursively** based on feature values to form a tree-like structure that classifies or predicts the outcome.

4. **Greedy Selection**:
   It assumes that using a **greedy approach** (e.g., selecting the best split at each node using metrics like Gini index or Information Gain) will lead to a good enough solution, even though it may not always be the globally optimal one.

5. **Feature Value Importance**:
   It assumes that the **most informative features** (based on entropy or Gini impurity) are placed near the top of the tree to improve classification or regression performance.

6. **Sufficient Data**:
   It assumes that there is **enough data** to learn meaningful patterns and avoid overfitting.

c)**How to control overfitting in decision tree?**

Ans: Overfitting in decision trees happens when the model learns not only the underlying patterns but also the noise in the training data, which leads to poor generalization on new data. Here are **key techniques to control overfitting in decision trees**:

---

1. Pruning the Tree

- **Pre-Pruning (Early Stopping):**
  - Set constraints during tree growth:
    - max_depth: Limits the depth of the tree.

        - min_samples_split: Minimum number of samples required to split a node.
        - min_samples_leaf: Minimum number of samples required in a leaf node.
        - max_leaf_nodes: Limits the number of leaf nodes.
- **Post-Pruning (Cost Complexity Pruning):**
    - First grow a large tree, then remove branches that have little importance using a pruning parameter (ccp_alpha in scikit-learn).

---

2. Use Cross-Validation

- Use **k-fold cross-validation** to tune hyperparameters and validate that your model generalizes well to unseen data.

---

3. Limit Feature Usage

- Use max_features to restrict the number of features considered when splitting. This reduces variance and helps generalization.

---

4. Use Ensemble Methods

- Instead of a single decision tree:
    - **Random Forest**: Reduces overfitting by averaging multiple de-correlated trees.
    - **Gradient Boosted Trees**: Builds trees sequentially while focusing on errors, with regularization.

---

5. Ensure Good Data Quality

- Clean and balanced datasets help prevent the tree from learning spurious patterns.

Example (scikit-learn):

python

Copy code

```
from sklearn.tree import DecisionTreeClassifier

tree = DecisionTreeClassifier(
    max_depth=5,
    min_samples_split=10,
    min_samples_leaf=5,
    max_leaf_nodes=20,
    ccp_alpha=0.01
)
```

11.Short Notes:-

a)K-N-N algorithm: https://www.geeksforgeeks.org/k-nearest-neighbours/

b)Random Forest algorithm: https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/

c)Support vector Mechine: https://www.geeksforgeeks.org/support-vector-machine-algorithm/

d)Over fitting problem: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

e)logistic Regrassion: https://www.geeksforgeeks.org/understanding-logistic-regression/

year 2025

Group B

2. **(i) What are the general steps of deploying classification model?**

   Ans: **(i) General Steps for Deploying a Classification Model**

Deploying a classification model involves several key steps that take a trained machine learning model from development to production, where it can serve real-time or batch predictions. Here's a general overview:

---

1. Data Preprocessing and Feature Engineering

- Clean and preprocess data (e.g., handle missing values, encode categorical variables).
- Normalize or scale features if needed.
- Select or extract relevant features.

---

2. Model Training

- Choose a classification algorithm (e.g., Logistic Regression, Random Forest, SVM).
- Split the dataset (training/test or cross-validation).
- Train the model on the training data.
- Evaluate performance using metrics like accuracy, precision, recall, F1-score.

---

3. Model Serialization

- Save the trained model to a file using tools like:
  - pickle or joblib (Python)
  - ONNX for interoperability across frameworks

Example (in Python):

python

Copy code

```
import joblib
joblib.dump(model, 'model.pkl')
```

---

4. Backend API Development

- Create a web server to load the model and serve predictions.
- Use frameworks like:
  - **FastAPI**, **Flask** (Python)
  - **Express.js** (Node.js)

Basic FastAPI example:

python

Copy code

```
from fastapi import FastAPI
```

```
import joblib
model = joblib.load("model.pkl")

@app.post("/predict")
def predict(input_data: dict):
   prediction = model.predict([list(input_data.values())])
   return {"prediction": prediction.tolist()}
```

---

5. Frontend Integration (Optional)

- Build a frontend (React, Angular, etc.) to collect user input.
- Connect frontend to backend using API calls (e.g., via fetch() or axios).

---

6. Model Hosting / Deployment

- Deploy the backend and model using:
  - **Cloud platforms** (e.g., AWS, Azure, GCP)
  - **Platform-as-a-Service** (e.g., Render, Heroku, Vercel for frontend)
  - **Docker** for containerization
- Use a domain or endpoint to access your service.

---

7. Testing and Monitoring

- Test the deployed model with real or sample inputs.
- Monitor API performance, error rates, and prediction accuracy.
- Implement logging and update models when needed.

ii)How can one apply ordinal along with one-hot encoding for the given scale of performance ratings: {poor, average, good, excellent, outstanding}?

Ans: https://www.datacamp.com/tutorial/one-hot-encoding-python-tutorial

4)(ii**) How can we evaluate a regression model?**

Ans:https://www.geeksforgeeks.org/regression-metrics/

6)i) a)K-N-N algorithm: https://www.geeksforgeeks.org/k-nearest-neighbours/

GroupC

1. **(i) How the physiological behaviour of Biological Neuron can be mathematically modelled?**

Ans: **(i) How the physiological behaviour of a Biological Neuron can be mathematically modelled:**

The **physiological behavior** of a biological neuron can be **mathematically modeled** using simplified abstractions that replicate how neurons process and transmit information. One of the most foundational and widely used mathematical models is the **McCulloch-Pitts Neuron Model**, which captures key features of neural activity.

� Mathematical Modeling of a Neuron:

1. Inputs:

A biological neuron receives signals through **dendrites**. In the mathematical model, these are represented as numerical **inputs**:

x1,x2,...,xnx_1, x_2, ..., x_nx1,x2,...,xn

---

2. Weights:

Each input is associated with a **weight** that represents the strength or importance of that input:

w1,w2,...,wnw_1, w_2, ..., w_nw1,w2,...,wn

---

3. Weighted Sum (Net Input):

The total input to the neuron is computed as a **weighted sum**:

z=∑i=1nwixi+bz = \sum_{i=1}^{n} w_i x_i + bz=i=1∑nwixi+b

Here,

- bbb is a **bias** term (analogous to the neuron's threshold),

- zzz is the **net input** to the neuron.

---

4. Activation Function:

This net input is passed through an **activation function** that determines whether the neuron "fires" (produces an output):

Common activation functions include:

- **Step function**: f(z)=1f(z) = 1f(z)=1 if z≥θz \geq \thetaz≥θ, else 0
- **Sigmoid**: f(z)=11+e−zf(z) = \frac{1}{1 + e^{-z}}f(z)=1+e−z1
- **ReLU**: f(z)=max(0,z)f(z) = \max(0, z)f(z)=max(0,z)

So, the **output** of the neuron is:

y=f(z)y = f(z)y=f(z)

◆ Summary Equation:

The entire process can be summarized as:

y=f(∑i=1nwixi+b)y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)y=f(i=1∑nwixi+b)

◆ Biological Analog:

Biological Neuron	Mathematical Model Equivalent
Dendrites (input signals)	Inputs xix_ixi
Synaptic weights	Weights wiw_iwi
Soma (cell body)	Weighted sum ∑wixi+b\sum w_ix_i + b∑wixi+b
Axon	Output yyy
Action potential	Activation function output

ii) Discuss on the followings.

(a) Convolutional Neural network

(b) Recurrent Neural Network

Ans:a) https://www.geeksforgeeks.org/introduction-convolution-neural-network/

b) https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/

7. (i) Compare ANN model's functionalities with the physiological behaviour of Biological Neuron.

Ans: https://www.geeksforgeeks.org/difference-between-ann-and-bnn/
ii)What are the following Activation Functions?

(a) Sigmoid (b) ReLu (c) Binary Step
Ans: a)**Sigmoid:** https://www.geeksforgeeks.org/derivative-of-the-sigmoid-function/

(b) **ReLu:** https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

(c) **Binary Step:** The **Binary Step** function is a type of **activation function** used in neural networks and machine learning models. It's one of the simplest activation functions and is defined as:

(d) f(x)={1if x≥00if x<0f(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}f(x)={10if x≥0if x<0

(e) Key Points:

(f) **Output:** Only 0 or 1, depending on whether the input is negative or non-negative.

(g) **Usage:** It was originally used in perceptrons (early neural network models).

(h) **Limitation:** It is **not differentiable**, so it cannot be used in gradient-based optimization (like backpropagation), making it unsuitable for deep learning.

(i) Example:

(j) If the input x=−2x = -2x=−2, then f(x)=0f(x) = 0f(x)=0
If the input x=3x = 3x=3, then f(x)=1f(x) = 1f(x)=1

(iii) What is the advantage of K-Medoid clustering strategy over K Means clustering strategy?

Ans:

https://www.geeksforgeeks.org/k-means-vs-k-medoids-clustering/

11. (i) Compare Hierarchical Divisive Clustering strategy with Hierarchical Agglomerative Clustering strategy.
Ans: https://www.geeksforgeeks.org/difference-between-agglomerative-clustering-and-divisive-clustering/

10)iii) How Silhouette Coefficient is used to evaluate the quality of a clustering model?

Ans: The **Silhouette Coefficient** is a metric used to evaluate the **quality of clusters** in a clustering model. It measures how similar an object is to its **own cluster (cohesion)** compared to other clusters (**separation**).

Formula:

For a single sample iii, the **Silhouette Coefficient s(i)s(i)s(i)** is defined as:

s(i)=b(i)−a(i)max(a(i),b(i))s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}s(i)=max(a(i),b(i))b(i)−a(i)

Where:

- a(i)a(i)a(i) = average distance of sample iii to all other points in **the same cluster** (intra-cluster distance).
- b(i)b(i)b(i) = lowest average distance of sample iii to all points in **any other cluster**, i.e., the distance to the nearest cluster that iii is **not** a member of (nearest-cluster distance).

Interpretation:

- s(i)s(i)s(i) ranges from **-1 to +1**:
  - **+1**: Sample is well clustered (far from neighboring clusters).
  - **0**: Sample is on or very close to the decision boundary between two clusters.

  - **-1**: Sample is likely in the wrong cluster (closer to a neighboring cluster).

How it evaluates clustering quality:

1. **Compute the average Silhouette Coefficient** across all samples.
2. A higher **average silhouette score** indicates:
   - Better defined clusters (samples are close to their own cluster and far from others).
   - Appropriate number of clusters.

Use case:

You can use the silhouette score to:

- **Choose the optimal number of clusters** (e.g., in K-Means).
- **Compare different clustering algorithms** or hyperparameter settings.

12. (i) Propose a supervised learning strategy for predicting whether a mail is a spam or not.

Ans: To predict whether an email is **spam or not (ham)** using a **supervised learning strategy**, you can follow this structured approach:

✅ 1. Problem Definition

- **Type**: Binary classification (Spam = 1, Not Spam = 0)

✅ 2. Data Collection

- Use datasets like:
  - **Enron Email Dataset**
  - **SpamAssassin Public Corpus**
  - **UCI Spam SMS Dataset**

✅ 3. Data Preprocessing

- **Text Cleaning**:
  - Lowercasing

    - Removing punctuation, stopwords, HTML tags, and special characters

- **Tokenization**
- **Stemming or Lemmatization**

✅ 4. Feature Extraction

Convert text into numerical features:

- **Bag of Words (BoW)**
- **TF-IDF (Term Frequency-Inverse Document Frequency)**
- **Word embeddings** (optional, e.g., Word2Vec, GloVe)

✅ 5. Train-Test Split

- Split the data into **training and testing sets**, e.g., 80% train, 20% test.

✅ 6. Model Selection

Use any of the following classification algorithms:

- **Naive Bayes** (especially good for text)
- **Logistic Regression**
- **Support Vector Machines (SVM)**
- **Random Forest**
- **XGBoost**
- For deep learning: LSTM or BERT

✅ 7. Model Training

- Train the model on the training dataset using labeled data (spam = 1, ham = 0).

✅ 8. Model Evaluation

Use metrics suitable for classification:

- **Accuracy**
- **Precision, Recall, F1-score** (especially important to handle imbalanced data)
- **Confusion Matrix**
- **ROC-AUC Score**

✅ 9. Hyperparameter Tuning

- Use techniques like **Grid Search** or **Random Search** with **Cross-Validation**.

✅ 10. Deployment

- Integrate the trained model into an email service.
- Use **FastAPI** or **Flask** for an API layer.

Optional Enhancements:

- Use **ensemble models** for better performance.
- Implement **incremental learning** for adapting to new types of spam