

ML UNIT – 4 (Supervised Learning: Classification) – END-SEM PYQ Answers➤ **MAY / JUN 2023****Q3) a) Explain with example the variant of SVM, the support vector regression. [5]**

Support Vector Regression (SVR): Support Vector Regression (SVR) is a variant of Support Vector Machine (SVM) that is used for **regression tasks** instead of classification.

Unlike classification SVM—which tries to find a maximum-margin separating hyperplane—SVR tries to find a **best-fit line (or curve)** that predicts continuous numerical values while keeping the prediction error within a certain margin.

Key Concepts of SVR**1. ϵ -insensitive loss function**

SVR introduces a margin of tolerance (ϵ).

Errors within $\pm\epsilon$ are ignored. Only points that lie **outside** this margin contribute to the loss.

2. Support Vectors

These are the data points that lie **outside the ϵ -tube** and influence the regression line.

They determine the final model.

3. Kernel Trick

SVR can model complex nonlinear relationships using kernels like:

- Linear
- Polynomial
- RBF (Gaussian)

4. Objective

SVR tries to keep the regression function as flat as possible while minimizing errors beyond ϵ .

Example: Suppose you want to predict house prices based on size:

Size (sq ft)	Price (₹ in lakhs)
600	30
750	40
900	48
1100	60

A normal regression line tries to minimize squared error.

But **SVR fits a line (or curve) with a margin $\pm\epsilon$** , such that:

Points within the ϵ -tube are ignored

Only points outside affect the model

If $\epsilon = 2$ lakhs, and a predicted price is 47 lakhs:

- Actual = 48
- Error = 1 lakh \rightarrow *within margin \rightarrow ignored*

If actual = 60 and predicted = 55:

- Error = 5 lakhs \rightarrow *outside margin \rightarrow becomes a support vector*

b) What do you mean by ensemble learning ? Differentiate between bagging& boosting. [6]

- **Ensemble learning** is a technique in machine learning where multiple models (called weak learners) are combined together to form a more accurate and stronger prediction model.
- The main idea is that a group of models working together can perform better than a single model, similar to a “committee decision.”
- Ensemble learning helps improve accuracy, reduce variance, and avoid overfitting. Common ensemble methods include Bagging, Boosting, and Stacking.
- It is widely used in classification and regression tasks where a single model may fail to generalize well.

Feature	Bagging	Boosting
Meaning	Uses multiple models trained independently on random subsets of data.	Trains models sequentially where each new model focuses on correcting errors of the previous one.
Objective	Reduce variance and avoid overfitting.	Reduce bias and improve overall accuracy.
Data Sampling	Uses bootstrapping (sampling with replacement).	Uses full dataset but assigns higher weights to misclassified samples.
Dependency Between Models	Models are independent of each other.	Models are dependent and trained one after another.
Common Algorithms	Random Forest, Bagged Decision Trees.	AdaBoost, Gradient Boosting, XGBoost.
Performance on Noisy Data	Performs well on noisy data.	Sensitive to noise and outliers.

c) What are different variants of multi-class classification? Explain them with suitable example.[6]

Multi-class classification is a type of supervised learning where a model is trained to classify input data into more than two distinct classes. Unlike binary classification, multi-class classification deals with problems having three or more outcome labels. It is commonly used in image recognition, document categorization, and medical diagnosis. The learning model predicts one class label for each input instance based on patterns identified during training.

Variant	Explanation	Example
One-vs-Rest (OvR)	In this approach, one classifier is trained per class. Each classifier separates one class	Identifying handwritten digits (0 –9) where separate classifiers

One-vs-All (OvA)	from all remaining classes. During prediction, the classifier with the highest score determines the final class.	are trained: digit 0 vs others, digit 1 vs others, etc.
One-vs-One (OvO)	This approach trains a separate classifier for every pair of classes. For n classes, $n(n-1)/2$ classifiers are built. A voting mechanism decides the final output class.	For a 4-class fruit classifier (Apple, Mango, Banana, Orange), classifiers like Apple vs Mango, Mango vs Banana, etc., are created.
Multinomial / Native Multi-Class	Here, a single model handles all classes directly without splitting into binary problems. Algorithms like Softmax Regression or Decision Trees support this format inherently.	Softmax regression predicting whether an animal in an image is a cat, dog, or horse in one model.

Q4) a) Calculate macro average precision, macro average recall and macro average F-score for the following given confusion matrix of multi-class classification. **[6]**

Predictions →

	A	B	C	D
A	100	80	10	10
B	0	9	0	1
C	0	1	8	1
D	0	1	0	9

Actual values ↓

Given confusion matrix:

Actual \ Predicted	A	B	C	D
A	100	80	10	10
B	0	9	0	1
C	0	1	8	1
D	0	1	0	9

Step-1: Calculate Precision, Recall & F-Score for each class**Class A**

- TP = 100
- FP = 0
- FN = (80+10+10) = 100

$$\text{Precision} = 100/(100+0) = \mathbf{1.00}$$

$$\text{Recall} = 100/(100+100) = \mathbf{0.50}$$

$$F1 = 2 \times (1 \times 0.5) / (1 + 0.5) = 0.6667$$

Class B

- TP = 9
- FP = (80+1+1) = 82
- FN = 1

$$\text{Precision} = 9/(9+82) = \mathbf{0.0989}$$

$$\text{Recall} = 9/10 = \mathbf{0.90}$$

$$F1 = 2 \times (0.0989 \times 0.9) / (0.9989) = 0.1782$$

Class C

- TP = 8
- FP = 10
- FN = 2

$$\text{Precision} = 8/(8+10) = \mathbf{0.4444}$$

$$\text{Recall} = 8/10 = \mathbf{0.80}$$

$$F1 = 2 \times (0.4444 \times 0.8) / (1.2444) = 0.5715$$

Class D

- TP = 9
- FP = 12
- FN = 1

$$\text{Precision} = 9/(9+12) = \mathbf{0.4286}$$

$$\text{Recall} = 9/10 = \mathbf{0.90}$$

$$F1 = 2 \times (0.4286 \times 0.9) / (1.3286) = 0.5805$$

Step-2: Compute Macro Averages

$$\text{Macro Precision} = \frac{1 + 0.0989 + 0.4444 + 0.4286}{4} = 0.493$$

$$\text{Macro Recall} = \frac{0.5 + 0.9 + 0.8 + 0.9}{4} = 0.775$$

$$\text{Macro F1-Score} = \frac{0.6667 + 0.1782 + 0.5715 + 0.5805}{4} = 0.499$$

Final Answer

- **Macro Average Precision = 0.493**
- **Macro Average Recall = 0.775**
- **Macro Average F-Score = 0.499**

b) Write a short note on : [6]**i) Random forest****ii) Adaboost**

i) Random Forest: Random Forest is an ensemble machine learning algorithm that builds multiple decision trees using random subsets of data and features, and combines their results to improve accuracy and reduce overfitting.

Key Points:

- Based on **bagging (Bootstrap Aggregation)** technique.
- Uses multiple **decision trees** trained independently.
- Final output is given by **majority voting (classification)** or **average prediction (regression)**.
- Handles **large datasets and high-dimensional data** effectively.
- Reduces **overfitting** and improves model stability and performance.
- Provides **feature importance ranking**, useful in data analysis.

ii) AdaBoost (Adaptive Boosting): AdaBoost is a boosting algorithm that combines multiple weak learners (usually decision stumps) trained sequentially, where each learner focuses more on errors made by the previous one.

Key Points:

- Based on the concept of **boosting**.
- Models are trained **sequentially**, not independently.
- Misclassified samples are given **higher weight** in the next round.
- Final output is obtained by **weighted voting** of all learners.
- Improves **accuracy and reduces model bias**.
- Works well for tasks like **face detection, spam filtering, and classification problems**, but is sensitive to noise and outliers.

c) Discuss K-Nearest Neighbour (KNN) algorithm with suitable example. [5]

K-Nearest Neighbour (KNN) is a supervised learning algorithm used for classification and regression, where prediction is made based on the class of the **K closest data points**. It assumes that similar samples exist close to each other in feature space.

How KNN Works (Points):

- KNN is a **lazy learner**, meaning it does not build a model during training but stores the dataset.
- When a new data point arrives, the algorithm calculates the **distance** between the new point and all training points (commonly using **Euclidean distance**).
- The algorithm selects the **K nearest neighbours** based on the smallest distances.
- For **classification**, prediction is done by **majority voting** among the neighbors.
- For **regression**, the predicted value is the **average** of the K nearest data points.
- Choice of **K plays an important role**:
 - Small K → may lead to overfitting
 - Large K → may oversmooth the result

Example: Suppose we want to identify whether a fruit is **Apple or Orange** using two features: sweetness and crunchiness.

If the new fruit is compared with the dataset and among its **5 nearest neighbors**,

- **3 are Apples**
- **2 are Oranges**

Then with **K = 5**, the fruit will be classified as **Apple (majority vote)**.

➤ **MAY / JUN 2024**

Q3) a) Why Ensemble Learning is used in Machine Learning? [5]

Ensemble learning is a technique in machine learning where multiple models (called weak learners) are combined to create a single stronger and more accurate model. The goal is to improve prediction performance by reducing errors that individual models might make.

Reasons Why Ensemble Learning is Used:

- **Improves Accuracy:**
Combining many weak or average models produces a stronger model that performs better than any individual model.
- **Reduces Overfitting:**
Ensemble techniques like bagging help in reducing variance, making the model more stable and improving generalization.

- **Handles Complex Problems:**
Single models may fail to capture complex patterns, but multiple models working together can learn better and give more reliable predictions.
- **Reduces Error (Bias & Variance):**
Techniques like boosting reduce bias, while bagging reduces variance, leading to lower overall prediction error.
- **More Robust and Reliable:**
Ensemble models are less sensitive to noise and outliers because decisions are based on multiple learners rather than one.

Example: Methods like Random Forest, AdaBoost, and Gradient Boosting are widely used in real-world applications such as fraud detection, medical diagnosis, and text classification, as they consistently outperform single models.

b) Advantages and Disadvantages of K-NN [6]

Advantages of K-NN:

- **Simple and Easy to Implement:** The algorithm is straightforward and does not require complex mathematical training steps.
- **No Training Phase Required:** Since it is a **lazy learner**, it stores the dataset and performs computation only when predicting.
- **Works Well with Small Datasets:** K-NN performs effectively when the dataset size is small and features are meaningful.
- **Flexible for Both Tasks:** Can be used for **classification and regression**, making it versatile in machine learning applications.
- **Adapts to Non-linear Decision Boundaries:** Because decisions are based on neighbors, K-NN can naturally handle complex decision surfaces.

Disadvantages of K-NN:

- **Computationally Expensive:** Prediction is slow as distance must be calculated from the query point to **every stored sample**.
- **Requires Feature Scaling:** Performance depends on normalized or standardized data because K-NN relies on distance measures.
- **Sensitive to Noise and Outliers:** A few incorrect or noisy points can affect prediction accuracy, especially with small K.
- **High Memory Usage:** Since it stores the entire dataset, it requires large memory for big datasets.
- **Choice of K Impacts Accuracy:** A poor selection of K may lead to underfitting (large K) or overfitting (small K).

c) What are different distance metrics used in K-NN? [6]

K-Nearest Neighbour (K-NN) uses distance metrics to measure how close or similar two data points are in the feature space. The choice of distance metric affects the prediction accuracy because K-NN relies on comparing distances between points.

Common Distance Metrics Used in K-NN:

1. Euclidean Distance

- Most widely used metric in K-NN.
- Measures the straight-line distance between two points.
- Formula: $d = \sqrt{\sum (x_i - y_i)^2}$
- Used when features are continuous.

2. Manhattan Distance (City Block Distance)

- Calculates distance as the sum of absolute differences between features.
- Formula: $d = \sum |x_i - y_i|$
- Useful when data lies on a grid or features are independent.

3. Minkowski Distance

- A generalized form of Euclidean and Manhattan distance.
- Formula: $d = (\sum |x_i - y_i|^p)^{1/p}$
- If $p=1 \rightarrow$ Manhattan, $p=2 \rightarrow$ Euclidean.

4. Chebyshev Distance

- Measures the maximum difference along any dimension.
- Formula: $d = \max(|x_i - y_i|)$
- Useful when direction independence or movement in all dimensions matters.

5. Hamming Distance

- Used for **categorical or binary data**.
- Counts number of positions where values differ.
- Example: "1101" vs "1000" \rightarrow Distance = 2

6. Cosine Similarity Distance

- Measures angle between two vectors instead of magnitude.
- Good for text data or high-dimensional datasets.
- Formula: $d = 1 - \frac{A \cdot B}{||A|| ||B||}$

Q4) a) What is Multi-Class Classification? Explain the variants of multi-class classification. [5]

Multi-class classification is a supervised learning approach where a model predicts one label from **three or more possible classes**. Unlike binary classification (which has only two classes), multi-class classification assigns each input to exactly **one class** among multiple categories. It is used in applications like handwritten digit recognition, sentiment analysis, and object classification.

Variants of Multi-Class Classification:**1. One-vs-Rest (OvR) / One-vs-All (OvA)**

- A separate classifier is trained for each class.
- Each classifier distinguishes **one class from all remaining classes**.
- During prediction, the classifier with the highest score decides the final output.
- Example: For classifying digits 0–9, **10 separate models** are built.

2. One-vs-One (OvO)

- A classifier is trained for **every pair of classes**.
- For **N classes**, number of models = $\frac{N(N-1)}{2}$.
- Final class is selected using a **majority voting system** among classifiers.
- Example: For 4 classes, $\frac{4 \times 3}{2} = 6$ models are created.

3. Multiclass Native (Direct Multiclass Models)

- A **single model** handles all classes together without splitting.
- Algorithms like **Decision Trees, Naive Bayes, and Softmax Regression** support multiclass learning naturally.
- Example: Softmax regression directly predicts if a picture is a **cat, dog, or horse**.

Multi-class classification allows prediction among multiple categories, and the main approaches include **OvR, OvO, and native multiclass methods**, chosen based on dataset size, complexity, and model requirements.

b) Explain kernel methods suitable for SVM. [6]

Kernel methods in Support Vector Machines (SVM) allow the algorithm to transform data from a lower-dimensional space to a higher-dimensional feature space. This helps SVM separate data that is **not linearly separable** using a linear boundary. The kernel function computes similarity between data points without explicitly transforming the data — a concept known as the **kernel trick**.

Common Kernel Functions Used in SVM:

1. Linear Kernel

- Used for linearly separable data.
- Performs no transformation; keeps the original feature space.
- Formula: $K(x, y) = x^T y$
- Works well when features are already meaningful and high-dimensional (e.g., text classification).

2. Polynomial Kernel

- Maps data into a higher dimension by creating polynomial combinations of features.
- Suitable for cases where relationships are more complex than linear separation.
- Formula: $K(x, y) = (x^T y + c)^d$
- Parameters c and degree d control curve flexibility.

3. RBF (Radial Basis Function) / Gaussian Kernel

- One of the most widely used kernels.
- Handles highly non-linear relationships by mapping data into infinite-dimensional space.
- Formula: $K(x, y) = e^{-\frac{\|x - y\|^2}{2\sigma^2}}$
- Works well when there is no prior knowledge about data structure.

4. Sigmoid Kernel

- Based on the activation function used in neural networks.
- Formula: $K(x, y) = \tanh(\alpha x^T y + c)$
- Mimics behavior of multilayer perceptrons (MLP).

Why Kernels are Useful:

- Allow SVM to handle **complex and non-linear boundaries**.
- Reduce computation using the **kernel trick** instead of computing high-dimensional transformations explicitly.
- Improve classification accuracy for datasets with complex structures.

c) What are different techniques used for outlier handling? [6]

Outlier handling refers to the process of detecting and treating extreme or unusual values in a dataset that differ significantly from other observations. Outliers can negatively affect machine learning model performance, especially in algorithms sensitive to numerical values.

Techniques for Outlier Handling:**1. Removal of Outliers**

- If outliers are due to incorrect data entry, noise, or measurement errors, they can be safely removed.
- Best used when dataset is large and the outlier does not contain valuable information.

2. Transformation Methods

- Apply transformations such as **log, square root, or Box-Cox** to reduce the effect of extreme values.
- Useful when outliers follow a skewed distribution.

3. Capping / Winsorization

- Replace extreme values with a defined upper and lower limit (e.g., 1st and 99th percentiles).
- Maintains dataset size while controlling the influence of outliers.

4. Imputation

- Replace outlier values with **mean, median, or mode** based on distribution.
- Median is preferred when data contains extreme skewness.

5. Statistical Methods (Z-Score / IQR Method)

- **Z-Score:** Values with $|Z| > 3$ are considered outliers.
- **IQR Rule:** Outliers fall below $Q_1 - 1.5 \times IQR$ or above $Q_3 + 1.5 \times IQR$.
- These are widely used during EDA (Exploratory Data Analysis).

6. Model-Based Approaches

- Use algorithms like **Isolation Forest, Local Outlier Factor (LOF), or DBSCAN** to automatically detect abnormal patterns.
- Useful in fraud detection, anomaly detection, and large datasets.

➤ MAY/ JUN 2025**Q3) a) Differentiate Binary classification with multiclass classification with example. [5]**

Binary Classification: Binary classification is a supervised learning task where the model predicts one of **two possible classes**. The output is limited to two labels such as **Yes/No, Spam/Not Spam, or 0/1**.

Multi-Class Classification: Multi-class classification is a supervised learning task where the model predicts one label from **three or more possible classes**. Each data point belongs to exactly one class among multiple categories.

Feature	Binary Classification	Multi-Class Classification
Number of Classes	Only two possible classes	Three or more classes
Output Type	Yes/No or 0/1	Multiple distinct labels
Model Complexity	Simple to train	More complex due to multiple classes
Evaluation Metrics	Accuracy, Precision, Recall, ROC-AUC	Macro/Micro Precision, Recall, Confusion Matrix
Algorithms Used	Logistic Regression, SVM, Naive Bayes	Softmax Regression, Random Forest, KNN, Multinomial Naive Bayes
Feature	Binary Classification	Multi-Class Classification
Decision Boundary	Usually simpler and linear	Often complex and non-linear
Model Output Format	Often uses sigmoid function	Often uses softmax function
Confusion Matrix Size	2×2 matrix	$N \times N$ matrix (where N = number of classes)
Training Time	Faster because fewer classes	Slower due to multiple class comparisons
Error Handling	Misclassification is straightforward	Misclassification may occur between multiple class pairs

b) Explain with example ensemble learning in ML [6]

Ensemble learning is a machine learning approach where multiple models (called weak learners) are combined to create a single, stronger and more accurate predictive model. The idea is that a group of models working together performs better than a single model, similar to how group decision-making is often more reliable.

Key Points:

- Ensemble learning helps to **improve accuracy and reduce prediction errors**.
- It helps overcome limitations of a single model by combining strengths of multiple learners.
- It reduces **overfitting** and increases model stability, especially in noisy datasets.
- It can reduce both **bias and variance**, depending on the technique used.
- Common ensemble strategies include **Bagging, Boosting, and Stacking**.
- Algorithms like **Random Forest and AdaBoost** are real-world examples of ensemble methods used in classification and regression.

Example: Suppose a classification problem predicts whether an email is **Spam or Not Spam**. Instead of using one model, we train:

- Logistic Regression
- Decision Tree
- SVM

These models may give slightly different predictions. Using ensemble learning, their outputs are combined through **majority voting** to make a final prediction.

If two models say "Spam" and one says "Not Spam," the final output is **Spam**.

c)Write short note on metrics for Evaluating classifier performance. [6]

Metrics for evaluating classifier performance are quantitative measures used to assess how accurately and effectively a classification model makes predictions. These metrics help compare models, identify errors, and improve the overall quality of the machine learning system.

Important Evaluation Metrics:

1. Accuracy

- Measures the proportion of correctly predicted instances out of total predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- Useful when classes are balanced.

2. Precision

- Measures how many predicted positive values are actually positive.

$$Precision = \frac{TP}{TP + FP}$$

- Important in applications like spam detection or fraud detection.

3. Recall (Sensitivity or True Positive Rate)

- Measures how many actual positive cases were correctly identified.

$$Recall = \frac{TP}{TP + FN}$$

- Useful when missing a positive case is costly (e.g., medical diagnosis).

4. F1-Score

- Harmonic mean of Precision and Recall; balances both especially for imbalanced datasets.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

5. Confusion Matrix

- A table that summarizes predicted vs actual classifications: TP, TN, FP, FN.
- Helps analyze classification errors in detail.

6. ROC Curve & AUC (Area Under Curve)

- ROC curve plots True Positive Rate vs False Positive Rate.
- **AUC score** indicates how well a classifier can distinguish between classes (closer to **1** is better).

Q4) a) Explain KNN with example. [5]

→ DONE

b) Write short note on handling imbalance data in multiclass classification.[6]

Imbalanced data refers to a situation in multi-class classification where the number of samples in each class is not evenly distributed. Some classes have many samples (majority classes), while others have very few (minority classes). This imbalance can cause the model to perform poorly on underrepresented classes.

Techniques to Handle Imbalanced Data:

1. Resampling Techniques

- **Oversampling:** Increase minority class samples using duplication or synthetic data generation (e.g., **SMOTE**).
- **Undersampling:** Reduce samples from majority class to balance distribution.
- Useful when class imbalance is severe.

2. Class Weight Adjustment

- Assign higher weights to minority classes during training.
- Helps the model give more importance to rare classes.
- Supported in algorithms like Logistic Regression, SVM, and Neural Networks.

3. Synthetic Data Generation (SMOTE)

- SMOTE (Synthetic Minority Oversampling Technique) creates new artificial samples for minority classes instead of duplicating.
- Helps in improving representation of minority class without losing majority class data.

4. Using Ensemble Learning

- Techniques like **Bagging and Boosting** (e.g., Balanced Random Forest, AdaBoost) help improve minority class performance.
- Boosting focuses on correcting errors for underrepresented classes.

5. Evaluation Metric Selection

- Instead of using **accuracy** (which may mislead), use:
 - **Precision, Recall, F1-score**
 - **Macro-average metrics**
 - **Confusion Matrix**
- These provide better insight into minority class performance.

6. Data Augmentation

- Generating new training samples using transformations like rotation, scaling, or noise addition (mostly used in images and text).
- Useful when collecting more real data is difficult.

These techniques help improve model performance, fairness, and accuracy when working with imbalanced multiclass datasets.

c) Explain Bagging and Boosting used in Ensemble Learning. [6]

Bagging (Bootstrap Aggregating): Bagging is an ensemble learning technique that trains multiple models independently using **random subsets of the training data (with replacement)**. The final prediction is made using **majority voting** (classification) or **averaging** (regression). Bagging helps reduce **variance** and prevents overfitting.

Key Points:

- Uses **bootstrap sampling**, meaning each model gets a different random portion of data.
- Models are trained **in parallel**, not sequentially.
- Helps improve stability and accuracy, especially with high-variance models.
- Best suited for algorithms like **Decision Trees**.
- Final output is combined using **voting or averaging**.

Example: Random Forest is a well-known bagging method where multiple decision trees are trained, and the final prediction is based on majority voting.

Boosting: Boosting is an ensemble technique where multiple weak learners are trained **sequentially**, and each new model focuses more on correcting the errors made by the previous model. The final prediction is made using **weighted voting**.

Key Points:

- Models learn one after another (sequential learning).
- Misclassified samples are given **higher weights** so the next model learns better.

- Reduces **bias and improves accuracy**.
- Sensitive to noise and outliers because it focuses on difficult cases.
- Commonly used boosting algorithms include **AdaBoost, Gradient Boosting, and XGBoost**.

Example: In **AdaBoost**, early weak learners (like decision stumps) make predictions, and misclassified samples receive higher weights. The final model combines all learners through weighted voting.

Bagging reduces **variance**, while boosting reduces **bias**—both improve model performance and are widely used in real-world machine learning.

➤ NOV / DEC 2023

Q3) a) Differentiate between bagging and boosting. [4]

→ DONE

b) What is Ensemble Learning? Explain the concept of Random Forest Ensemble Learning. [9]

Ensemble learning is a machine learning technique where multiple models, known as **weak learners**, are combined to build a stronger and more accurate predictive model. The idea is that a group of models working together performs better than a single model alone.

Key Features:

- Improves **accuracy and generalization**.
- Reduces **errors, bias, and variance**.
- Works well for complex decision-making problems.

Common ensemble methods:

- **Bagging**
- **Boosting**
- **Stacking**

Random Forest Ensemble Learning

Definition: Random Forest is an ensemble learning technique based on **bagging**, where multiple decision trees are trained and combined to make a final prediction using **majority voting** (for classification) or **average prediction** (for regression).

How Random Forest Works:

1. **Bootstrap Sampling (Bagging):** Multiple subsets of the original training data are created **randomly with replacement**.
2. **Building Multiple Decision Trees:** Each tree is trained independently using a different data sample. At each split, only a **random subset of features** is considered, increasing diversity.
3. **Prediction Phase:** For classification: each tree votes for a class and the **majority class** becomes the final output. For regression: output is the **average of all tree predictions**.

Example: Suppose a dataset is used to predict whether a customer will **purchase a product**. Random Forest creates multiple decision trees using different subsets of customer data (age, salary, purchase history, etc.). If:

- Tree-1 → Yes
- Tree-2 → Yes
- Tree-3 → No
- Tree-4 → Yes

Then the final output = **Yes** (majority vote).

Advantages of Random Forest:

- Handles **large, high-dimensional datasets** efficiently.
- Reduces overfitting compared to a single decision tree.
- Can handle **missing values and noisy data**.
- Provides **feature importance**, useful for model interpretation.

Limitations:

- Requires more **computational resources** than a single tree.
- Less interpretable compared to a simple decision tree.

c) What is the relation between Precision and Recall? Explain with an example. [4]

Precision and Recall are performance metrics used to evaluate classification models, especially when dealing with imbalanced datasets.

- **Precision** measures how many of the predicted positive results are actually correct.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall (Sensitivity)** measures how many actual positive cases the model successfully identified.

$$Recall = \frac{TP}{TP + FN}$$

Relation Between Precision and Recall:

- Precision and Recall are **inversely related** in most cases. Increasing Precision may lower Recall and vice-versa, depending on the decision threshold.
- To balance both, the **F1-score** (harmonic mean) is used.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Example: Consider a medical test detecting a disease:

Output Type	Count
True Positive (TP)	8
False Positive (FP)	2
False Negative (FN)	4

- Precision = $\frac{8}{8+2} = 0.80$
- Recall = $\frac{8}{8+4} = 0.67$

This means the test is accurate in predicting positives (high precision), but it misses some real cases (lower recall).

Q4) a) What is K-Fold Cross-Validation? Comment on the value of K when it is too large or too small. How do you decide K value? [9]

K-Fold Cross-Validation is a model evaluation technique used to assess how well a machine learning model generalizes to unseen data. In this method, the dataset is divided into **K equal-sized subsets (folds)**. The model is trained on **K-1 folds** and tested on the remaining fold. This process is repeated **K times**, and the final evaluation metric is the **average performance** across all folds. It reduces bias and variance in model evaluation compared to a single train-test split.

Effect of Different Values of K

i) When the Value of K is Too Large

- When K is large (e.g., K = number of samples → Leave-One-Out CV), each fold contains only **one or very few samples** for testing.
- The training set becomes almost the entire dataset.
- This results in:
 - ✓ Very **low bias** (model trained nearly on full data)
 - × **High variance** in evaluation
 - × **High computational cost**
- **Summary:** The model may generalize poorly due to instability and takes longer to compute.

ii) When the Value of K is Too Small

- Small K values (e.g., K = 2 or 3) make each test fold large and training fold small.
- This results in:
 - × **High bias**, because the model gets less training data
 - ✓ **Low variance**, because averaging over fewer folds is stable
 - ✓ Faster computation
- **Summary:** The model performance may be underestimated due to insufficient training data.

How to Decide the Value of K

- The most commonly used and balanced choice is **K = 5 or K = 10**, as they provide a good trade-off between **bias, variance, and computation time**.
- For large datasets: **K = 5** is preferred.
- For small datasets: **K = 10** or **Leave-One-Out CV** can be used to avoid losing too much training data.

K Value	Bias	Variance	Computation	Recommendation
Too Small	High	Low	Fast	Not ideal
Too Large	Low	High	Slow	Not stable
Moderate (5–10)	Balanced	Balanced	Reasonable	Best choice

b) Explain: Accuracy, Precision, Recall and F-Score [8]

i) Accuracy: Accuracy measures the proportion of correctly predicted instances (both positive and negative) out of the total number of predictions.

$$\text{Formula: Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Use Case: Useful when the dataset has **balanced classes**.

ii) Precision: Precision measures how many of the predicted positive samples are actually positive. It focuses on the correctness of positive predictions.

$$\text{Formula: Precision} = \frac{TP}{TP + FP}$$

Use Case: Important when **false positives are costly**, e.g., spam detection, fraud alerts.

iii) Recall (Sensitivity or True Positive Rate): Recall measures how many actual positive samples were correctly identified by the model.

$$\text{Formula: Recall} = \frac{TP}{TP + FN}$$

Use Case: Useful when **missing positive cases is risky**, e.g., medical diagnosis or defect detection.

iv) F-Score (F1-Score): F-Score is the harmonic mean of Precision and Recall. It provides a balanced measure when both precision and recall are important.

$$\text{Formula: } F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Use Case: Useful for **imbalanced datasets**, where accuracy alone may give misleading results.

Example: If a model identifies emails as spam:

Prediction Type	Count
TP	40
FP	10
FN	5
TN	45

Then:

- Precision = $\frac{40}{40+10} = 0.80$
- Recall = $\frac{40}{40+5} = 0.89$
- F1-Score $\approx 2 \times (0.80 \times 0.89) / (0.80 + 0.89) \approx 0.84$

➤ NOV / DEC 2024

Q3) a) Explain kernel methods which are suitable for SVM.

→ DONE

b) What are advantages and disadvantages of K-NN? [6]

→ DONE

c) What are different distance metrics are used in K-NN? [5]

→ DONE

Q4) a) What is Multi Class Classification? Explain the variants of Multi Class Classification. [5]

→ DONE

b) What are different techniques used for outlier handling? [6]

→ DONE

c) With suitable diagram, explain Random forest Algorithm with example.[6]

1. Introduction: Random Forest is an **ensemble learning technique** used for **classification and regression**. It works by building a **large number of decision trees** and combining their outputs.

2. Key Concepts

- **Ensemble Method:** Combines results of multiple weak learners (decision trees) to form a strong model.
- **Bootstrap Sampling:** Each tree is trained on a random sample (with replacement) of the training data.
- **Feature Randomness:** At each split, the algorithm selects a **random subset of features**, increasing diversity among trees.
- **Majority Voting / Averaging:**

- For **classification** → final prediction = **majority vote** of all trees.
- For **regression** → final prediction = **average** of outputs.

3. Steps of the Random Forest Algorithm

- **Step 1:** Select multiple bootstrap samples from training data.
- **Step 2:** For each sample, build a **decision tree**:
 - Randomly choose a subset of features at every node.
 - Split the node based on the best feature among the selected subset.
- **Step 3:** Repeat to build many trees (**forest**).
- **Step 4:** For prediction:
 - Each tree gives its output.
 - Combine outputs using **majority vote** or **mean**.

4. Advantages

- Handles large datasets with higher dimensionality.
- Reduces overfitting due to randomness.
- Works well even when some data is missing.
- Stable and highly accurate compared to a single decision tree.

5. Example (Classification Example)

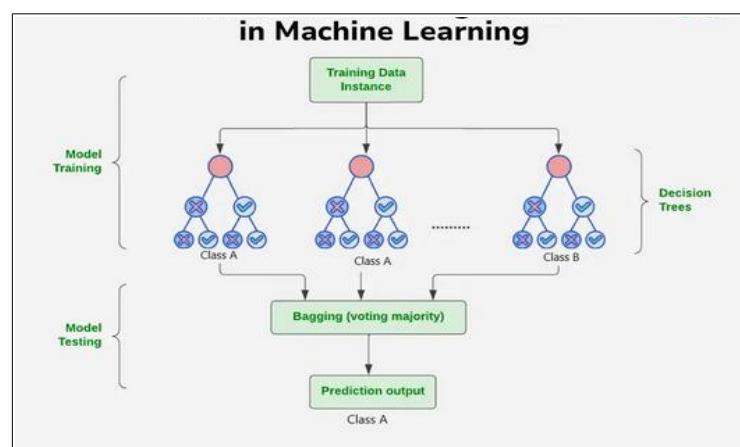
Suppose we want to classify whether a student will **Pass or Fail** based on:

- **Study Hours**
- **Attendance**
- **Internal Marks**

Process:

- Create multiple bootstrap samples from the dataset.
- Build many decision trees:
 - Each tree may select different features (e.g., one tree uses Attendance + Marks, another uses Hours + Attendance).
- For a new student:
 - Tree 1 → Pass
 - Tree 2 → Pass
 - Tree 3 → Fail
 - Tree 4 → Pass
 - Tree 5 → Pass

Final Prediction: → Majority vote = **Pass**



6. Conclusion: Random Forest is a **robust, accurate, and widely used ensemble algorithm**. It reduces overfitting, handles noisy data, and works well for both classification and regression.

➤ **Additional NOV / DEC 2022 Questions:**

Q3) a) Explain in brief methods used for Evaluating classification models. [5]

1. Confusion Matrix

- Tabular representation comparing **actual** vs **predicted** classes.
- Contains **TP (True Positive)**, **TN (True Negative)**, **FP (False Positive)**, **FN (False Negative)**.
- Forms the basis for most evaluation metrics.

2. Accuracy

- Measures the proportion of **correct predictions**.
- Formula:
Accuracy = (TP + TN) / (TP + TN + FP + FN)
- Useful when dataset is balanced.

3. Precision

- Shows how many predicted positives are actually positive.
- Important when **false positives are costly**.
- Formula:
Precision = TP / (TP + FP)

4. Recall (Sensitivity / True Positive Rate)

- Measures how many actual positives were correctly identified.
- Important when **false negatives are costly**.
- Formula:
Recall = TP / (TP + FN)

5. F1-Score

- Harmonic mean of **Precision** and **Recall**.
- Useful when dataset is **imbalanced**.
- Formula:
F1 Score = 2 × (Precision × Recall) / (Precision + Recall)

6. ROC Curve (Receiver Operating Characteristic)

- Plots **True Positive Rate (TPR)** vs **False Positive Rate (FPR)**.
- Shows model performance across different thresholds.

7. AUC (Area Under the Curve)

- Area under the ROC curve.

- Higher AUC indicates a **better classifier**.
- $AUC = 1 \rightarrow$ perfect model; $AUC = 0.5 \rightarrow$ random guessing.

8. K-Fold Cross-Validation

- Dataset is divided into **K subsets (folds)**.
- Model is trained on **K-1 folds** and tested on the remaining fold.
- Reduces overfitting and provides a reliable performance estimate.

9. Classification Report

- Provides a summary of **Precision, Recall, F1-Score** for each class.
- Useful for **multi-class** classification evaluation.

- b) Consider the following data to predict the student pass or fail using the K-Nearest Neighbor Algorithm (KNN) for the values physics = 6 marks, Chemistry = 8 marks with number of Neighbors $K = 3$. **[6]**

Physics (marks)	Chemistry (marks)	Results
4	3	Fail
6	7	Pass
7	8	Pass
5	5	Fail
8	8	Pass

1. Given Training Data

Physics	Chemistry	Result
4	3	Fail
6	7	Pass
6	8	Pass
5	5	Fail
8	8	Pass

2. Test Point: Physics = 6; Chemistry = 8

We must compute distances from the test point to all training samples.

3. Use Euclidean Distance Formula: $d = \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)}$

4. Calculate Distances

(i) To (4,3) – Fail

$$d = \sqrt{((6-4)^2 + (8-3)^2)} = 4+25 = 29 \approx 5.38$$

(ii) To (6,7) – Pass

$$d = \sqrt{((6-6)^2 + (8-7)^2)} = 1 = 1$$

(iii) To (6,8) – Pass

$$d = \sqrt{((6-6)^2 + (8-8)^2)} = 0$$

(iv) To (5,5) – Fail

$$d = \sqrt{((6-5)^2 + (8-5)^2)} = 1+9 = 10 \approx 3.16$$

(v) To (8,8) – Pass

$$d = \sqrt{((6-8)^2 + (8-8)^2)} = 4 = 2$$

5. Sort by Distance (Nearest First)

Distance	Point	Result
0	(6,8)	Pass
1	(6,7)	Pass
2	(8,8)	Pass
3.16	(5,5)	Fail
5.38	(4,3)	Fail

6. Select K = 3 Nearest Neighbors

1. (6,8) → **Pass**
2. (6,7) → **Pass**
3. (8,8) → **Pass**

7. Majority Voting

- Pass = **3**
- Fail = **0**

Final Prediction: The student will PASS.

Q3) c) Write short note on Ensemble learning methods: [6]**i) Simple****ii) Advanced****i) Simple Ensemble Methods**

1. Definition: Combine predictions of multiple **independent base models** to improve accuracy and reduce errors. Also called **basic or classical ensemble methods**.

2. Types

- **Majority Voting (Hard Voting)**
 - Each classifier votes for a class.
 - Final output = **class with most votes**.
- **Averaging (Soft Voting)**
 - Used for regression or probability outputs.
 - Final output = **average of predictions**.
- **Weighted Voting**
 - Models with better performance get **higher weights**.
 - Final decision based on weighted sum of predictions.

3. Characteristics

- Easy to implement.
- No dependency between models.
- Works well with diverse classifiers.
- Reduces variance but may not reduce bias significantly.

ii) Advanced Ensemble Methods

1. Definition: More complex ensembles that use **meta-learning**, **randomization**, or **sequential model training**. Improve performance by reducing both **bias and variance**.

2. Techniques

- **Bagging (Bootstrap Aggregating)**
 - Trains multiple models on **different bootstrap samples**.
 - Reduces variance.
 - Example: **Random Forest**.
- **Boosting**
 - Models are trained **sequentially**.
 - Each new model focuses on **errors** of the previous one.

- Examples: **AdaBoost, Gradient Boosting, XGBoost.**
- **Stacking (Stacked Generalization)**
 - Combines predictions of base learners using a **meta-classifier**.
 - Example: Level-1 models' outputs → fed to another model for final prediction.

3. Characteristics

- More accurate than simple methods.
- Can handle complex data patterns.
- Used in high-performance ML applications.
- Requires more computation and tuning.

Summary

- **Simple Ensembles:** voting, averaging, weighted voting → easy & fast.
- **Advanced Ensembles:** bagging, boosting, stacking → more powerful & widely used.

Q4) b) Write short note on importance of confusion matrix.

Importance of Confusion Matrix:

1. Complete Performance Summary

- Provides a **detailed evaluation** of a classification model.
- Shows correct and incorrect predictions through **TP, TN, FP, FN** values.

2. Helps Identify Type of Errors

- Distinguishes between:
 - **False Positives (FP)** – Type I error
 - **False Negatives (FN)** – Type II error
- Useful for domains like **medical diagnosis, fraud detection**, etc.

3. Basis for All Major Evaluation Metrics: Confusion matrix is used to compute:

- **Accuracy**
- **Precision**
- **Recall (Sensitivity)**
- **Specificity**
- **F1-score**
- **Error rate**

Thus, it helps evaluate model performance from multiple perspectives.

4. Works for Both Binary and Multi-class Problems

- Can extend to **multi-class classification** by using larger matrices.
- Gives class-wise performance, useful for imbalanced datasets.

5. Reveals Class Imbalance Issues

- Shows how many samples from each class were correctly or incorrectly predicted.
- Helps detect bias toward majority class.

6. Improves Model Interpretation

- Easy to understand and visualize model behavior.
- Helps identify whether the model is **overfitting or underfitting** a specific class.

7. Useful for Model Comparison

- Allows comparison of different classifiers using the same dataset.
- Helps choose the best model based on error types and prediction patterns.

8. Supports Threshold Adjustment

- By analyzing FP and FN, decision thresholds can be corrected for:
 - Higher sensitivity
 - Higher specificity
 - Balanced performance

Conclusion: A confusion matrix is a **fundamental tool** in machine learning that provides a clear and comprehensive understanding of classification model performance. It helps measure accuracy, detect errors, handle imbalance, and guide model improvement.

Q4) c) Define following terms with reference to SVM.

i) Separating hyperplane

ii) Margin

i) Separating Hyperplane

- A **hyperplane** is a decision boundary that separates data points of different classes in SVM.
- In **2D**, it is a **line**; in **3D**, it is a **plane**; in higher dimensions, it is a **hyperplane**.
- The separating hyperplane divides the feature space such that:
 - Data points from one class lie on one side.
 - Points from the other class lie on the opposite side.
- Mathematically represented as: $w \cdot x + b = 0$
- SVM tries to find the **optimal hyperplane** that maximizes the margin.
- Only **support vectors** (critical data points) influence its position.

ii) Margin

- The **margin** in SVM is the distance between the separating hyperplane and the closest data points (support vectors) from each class.
- SVM aims to **maximize this margin**, leading to better generalization.
- Larger margin → **more robust classifier**, less overfitting.
- Margin boundaries are represented as: $w \cdot x + b = +1$; $w \cdot x + b = -1$
- Points lying on these boundaries are called **support vectors**.
- Optimal SVM chooses a hyperplane that gives the **maximum possible margin**.

Summary

- **Separating hyperplane**: Decision boundary that classifies data.
- **Margin**: Distance between hyperplane and nearest support vectors; SVM maximizes this margin.

NOTE: Please verify all answers before referring.