ML Exam Answer Key

Absolutely, Sachin. Let's build a **unit-wise detailed theory answer key** for your **Machine Learning (ML)** subject based on previous university exam questions.

Please share one or more of the following so I can begin:

1. 📄 **List of previous exam questions** (unit-wise if possible), OR
2. 📘 **Syllabus of your ML subject** so I can identify probable topics and build the answer key accordingly, OR
3. 📸 **Images/PDFs of your question papers** — I can extract the questions from those.

Once I have the questions or syllabus, I'll generate clear, formal, and technical answers using:

- Definitions
- Mathematical notations and formulas
- Step-by-step derivations/algorithms
- Diagrams (if needed)
- Real-world examples

Let's get you a perfect answer key! Just upload or paste the data to get started.

Great! Below is a **formal and technical answer key** for the questions you've provided. These are structured for **maximum scoring potential in university exams**, with definitions, diagrams (described textually), examples, and comparisons.

---

1. Define Machine Learning. What are the types of Machine Learning? Explain with examples.

Definition:

Machine Learning (ML) is a subset of Artificial Intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.

> **Formal Definition:**
> *"A computer program is said to learn from experience E with respect to some task T and performance measure P, if its performance on T, as measured by P, improves with experience E."* — Tom Mitchell, 1997

Types of Machine Learning:

A. Supervised Learning:

- Learns a function from labeled training data (input-output pairs).
- Goal: Predict output `Y` from input `X`.

- **Examples:**
  - Regression: Predicting house prices.
  - Classification: Email spam detection.

B. Unsupervised Learning:

- Learns patterns or groupings from data without labeled responses.
- Goal: Find hidden structure in data.
- **Examples:**
  - Clustering: Customer segmentation.
  - Dimensionality Reduction: PCA in gene expression analysis.

C. Reinforcement Learning:

- Agent learns by interacting with an environment to maximize cumulative reward.
- Goal: Learn a policy π(s) that maps state `s` to action `a`.
- **Examples:**
  - Game AI (e.g., AlphaGo).
  - Robot navigation.

---

2. Explain the steps involved in building a Machine Learning model.

Steps in Building a Machine Learning Model:

1. **Problem Definition:**
   - Understand the business problem and define objectives clearly.
2. **Data Collection:**
   - Gather data from various sources (databases, APIs, sensors).
3. **Data Preprocessing:**
   - Handle missing values, outliers.
   - Feature scaling (e.g., normalization, standardization).
   - Encoding categorical variables.
4. **Feature Selection/Engineering:**
   - Select relevant features using techniques like correlation analysis or PCA.
   - Create new meaningful features.
5. **Splitting Dataset:**
   - Divide dataset into training, validation, and testing sets (e.g., 70:15:15).

6. **Model Selection:**
    - Choose appropriate algorithm (e.g., Linear Regression, Decision Trees, SVM).
7. **Model Training:**
    - Fit the model to training data by minimizing a cost/loss function.
8. **Model Evaluation:**
    - Evaluate performance using metrics:
        - Classification: Accuracy, Precision, Recall, F1-score.
        - Regression: RMSE, MAE, R^2.
9. **Hyperparameter Tuning:**
    - Use Grid Search or Random Search to find optimal model parameters.
10. **Model Deployment:**
    - Deploy the model into a production environment using Flask, FastAPI, or cloud services.
11. **Monitoring & Maintenance:**
    - Regularly monitor performance and retrain the model as required.

---

3. Explain Bias-Variance Tradeoff with suitable diagram.

Definitions:

- **Bias**: Error due to incorrect model assumptions. High bias leads to underfitting.
- **Variance**: Error due to model's sensitivity to small fluctuations in training data. High variance leads to overfitting.

Bias-Variance Tradeoff:

- There exists a tradeoff between bias and variance that affects the total prediction error:

$$\text{Total Error} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

Diagram Description (to be drawn in exams):

A **U-shaped curve**:

- X-axis: Model Complexity
- Y-axis: Error
- Plot three curves:
    - Training error (decreasing)
    - Validation error (U-shaped)
    - Bias and Variance components

Goal:

Find the optimal model complexity where the validation error is minimized — balance between bias and variance.

---

4. What are different applications of Machine Learning?

Major Applications:

1. **Healthcare:**
   - Disease prediction, radiology image analysis (e.g., cancer detection via CNNs).
2. **Finance:**
   - Fraud detection, credit scoring, algorithmic trading.
3. **Retail:**
   - Recommendation systems (e.g., Amazon, Netflix), customer segmentation.
4. **Manufacturing:**
   - Predictive maintenance, quality inspection.
5. **Transportation:**
   - Autonomous vehicles (self-driving cars using RL and DL).
6. **Natural Language Processing (NLP):**
   - Chatbots, sentiment analysis, translation systems.
7. **Cybersecurity:**
   - Intrusion detection, malware classification.

---

5. Compare Supervised, Unsupervised, and Reinforcement Learning.

Feature	Supervised Learning	Unsupervised Learning	Reinforcement Learning
Data	Labeled	Unlabeled	Sequential + Reward signal
Goal	Predict outcomes	Discover hidden patterns	Maximize cumulative reward
Examples	Classification, Regression	Clustering, Dim. Reduction	Game playing, Robotics
Algorithms	SVM, Decision Tree, KNN	K-means, PCA, Hierarchical	Q-learning, Deep Q-Network (DQN)
Feedback	Direct (via labels)	Indirect (via structure)	Delayed (via reward)
Nature	Passive Learning	Exploratory	Trial-and-error learning

---

6. Explain overfitting and underfitting with examples.

Underfitting:

- Model is too simple to learn the underlying data structure.

- High bias, low variance.
- Poor performance on both training and test data.
- **Example:** Using a linear model for non-linear data.

Overfitting:

- Model is too complex, fits noise in training data.
- Low bias, high variance.
- Excellent performance on training data but poor generalization.
- **Example:** Deep decision tree on small dataset.

Ideal Fit:

- Balanced model with good generalization.

Graph Description (to draw in exams):

- Plot three curves on training vs testing error:
  - Underfitting: High training and testing error.
  - Overfitting: Low training, high testing error.
  - Optimal Fit: Moderate training, low testing error.

Would you like me to continue with more units/questions in a similar format? Or would you like this compiled into a PDF/Word doc for easy printing or studying?

Excellent, Sachin! Here's the **Unit 2 theory answer key** on **Regression and Evaluation**, formatted formally and technically to suit **university exams**. Equations, concepts, formulas, and structured points are included for clarity and scoring.

✅ Unit 2: Regression and Evaluation – Theory Answer Key

---

1. Explain Linear Regression with mathematical equations.

Definition:

Linear Regression is a supervised learning algorithm used to model the relationship between a dependent variable y and one or more independent variables x, assuming a linear relationship.

Equation:

For **Simple Linear Regression**:

$$y = \beta_0 + \beta_1 x + \varepsilon$$

Where:

- y = Dependent variable (target)
- x = Independent variable (feature)
- β_0 = Intercept
- β_1 = Slope (coefficient)
- ε = Error term

Objective:

Minimize the **Cost Function** (typically Mean Squared Error):

$$J(\beta_0, \beta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$$

For **Multiple Linear Regression**:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$$

---

2. Explain Logistic Regression with Sigmoid function.

Definition:

Logistic Regression is a classification algorithm used to model the probability of a binary outcome using a logistic (sigmoid) function.

Hypothesis Function:

$$h_\theta(x) = \frac{1}{1 + e^{-(\theta^T x)}} = \sigma(\theta^T x)$$

Where:

- $\theta^T x$ = Linear combination of features
- $\sigma(z)$ = Sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Interpretation:

- Output $h_\theta(x) \in (0, 1)$ is the estimated probability:

$$P(y = 1 | x; \theta) = h_\theta(x)$$

Decision Boundary:

- If $h_\theta(x) \geq 0.5$, predict class 1.
- If $h_\theta(x) < 0.5$, predict class 0.

3. What are different error metrics for regression? Explain with formula.

A. Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

B. Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

C. Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

D. R-squared (R^2) Score:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where:

- y_i = Actual value
- $\hat{y}_i$ = Predicted value
- $\bar{y}$ = Mean of actual values

---

4. Differentiate between Linear and Logistic Regression.

Criteria	Linear Regression	Logistic Regression
Type	Regression	Classification
Output Range	Continuous (e.g., any real value)	Probability (0 to 1)
Hypothesis	$h(x) = \theta^T x$	$h(x) = \frac{1}{1+e^{-\theta^T x}}$
Error Function	MSE (Mean Squared Error)	Log Loss / Cross Entropy
Use Case	Predict sales, temperature, price, etc.	Predict binary class (spam or not)

---

5. Explain gradient descent algorithm.

Definition:

Gradient Descent is an optimization algorithm used to minimize the cost function by iteratively updating the model parameters in the direction of steepest descent.

Formula:

Update rule for each parameter θ_j:

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$$

Where:

- α = Learning rate (step size)
- $J(\theta)$ = Cost function
- $\frac{\partial J}{\partial \theta_j}$ = Partial derivative (gradient)

Steps:

1. Initialize θ randomly.
2. Compute cost $J(\theta)$.
3. Update θ using gradient.
4. Repeat until convergence (cost is minimized).

Variants:

- **Batch Gradient Descent** – Uses all data points.
- **Stochastic Gradient Descent (SGD)** – Uses one data point at a time.
- **Mini-batch Gradient Descent** – Uses small random batches.

---

6. Discuss the assumptions of linear regression.

1. **Linearity:**
   Relationship between independent and dependent variables is linear.
2. **Independence:**
   Observations are independent of each other.
3. **Homoscedasticity:**
   Constant variance of error terms.
4. **Normality of Errors:**
   Errors (residuals) are normally distributed.
5. **No Multicollinearity (for multiple regression):**
   Independent variables are not highly correlated.
6. **No Autocorrelation:**
   Errors are not correlated across observations (especially in time series).

---

7. Explain cost function in regression models.

Definition:

A cost function measures the error between predicted values $\hat{y}$ and actual values y. The goal of learning is to minimize this function.

Linear Regression Cost Function (MSE):

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (h_\theta(x_i) - y_i)^2$$

Where:

- $h_\theta(x_i)$ = Predicted value
- y_i = Actual value
- n = Number of samples

Logistic Regression Cost Function (Log Loss):

$$J(\theta) = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log(h_\theta(x_i)) + (1 - y_i) \log(1 - h_\theta(x_i))]$$

Purpose:

- Used in optimization (e.g., Gradient Descent).
- Helps quantify model performance.
- Guides parameter updates.

Would you like me to continue with **Unit 3 (Classification, Decision Trees, KNN, etc.)** next? I can keep the same detailed format. Let me know!

Great! Here's your **Unit 3: Classification Algorithms and Model Evaluation** theory answer key, written in a **formal and technical tone** suitable for university exams. Each question is answered with formulas, diagrams (described), examples, and comparisons.

✅ Unit 3: Classification Algorithms and Model Evaluation – Theory Answer Key

---

1. Explain Decision Tree algorithm with example and diagram.

Definition:

A Decision Tree is a flowchart-like structure used for classification and regression tasks. It splits the dataset into subsets based on feature values using decision rules.

Key Concepts:

- **Nodes**: Test conditions on features.
- **Branches**: Outcome of test conditions.
- **Leaf Nodes**: Class labels (decisions).

Splitting Criteria:

- **Gini Index** (for CART):

$$Gini(t) = 1 - \sum_{i=1}^{c} p_i^2$$

- **Information Gain** (for ID3):

$$IG(D, A) = Entropy(D) - \sum_{v \in Values(A)} \frac{|D_v|}{|D|} \cdot Entropy(D_v)$$

Entropy:

$$Entropy(D) = -\sum_{i=1}^{c} p_i \log_2(p_i)$$

Example:

Predicting whether a person will play tennis based on **Outlook**, **Humidity**, etc.

Outlook	Humidity	Play
Sunny	High	No
Overcast	Normal	Yes

- First split on “Outlook”, then based on “Humidity”.

Diagram Description:

Draw a tree:

- Root: Outlook
- Child Nodes: Sunny, Overcast, Rain
- Leaf Nodes: "Play = Yes" or "Play = No"

---

2. What is K-Nearest Neighbors (KNN)? How does it work?

Definition:

KNN is a **lazy learning algorithm** that classifies a data point based on the majority label among its k-nearest neighbors in the feature space.

Steps:

1. Choose value of k.
2. Compute distance between the test point and all training points.
    - **Euclidean Distance**:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

3. Select the **k** closest data points.
4. Perform majority voting to assign a class label.

Characteristics:

- **Non-parametric** (no assumption about data distribution).
- **Distance-based**.

Example:

To classify a new flower based on petal length and width, find the 3 nearest labeled flowers and vote.

---

3. What is Naive Bayes classifier? Derive the classification formula.

Definition:

Naive Bayes is a **probabilistic classifier** based on Bayes' Theorem, assuming independence between features.

Bayes' Theorem:

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$$

Where:

- C = Class
- $X = (x_1, x_2, ..., x_n)$ = Feature vector

Naive Assumption:

$$P(X|C) = \prod_{i=1}^{n} P(x_i|C)$$

Final Classification Rule:

$$\hat{C} = \arg\max_{C} \left[P(C) \cdot \prod_{i=1}^{n} P(x_i|C) \right]$$

Types:

- **Gaussian Naive Bayes** (continuous features)
- **Multinomial Naive Bayes** (text data)
- **Bernoulli Naive Bayes** (binary features)

---

4. Compare Decision Tree and Random Forest.

Criteria	Decision Tree	Random Forest
Structure	Single tree	Ensemble of trees
Overfitting	Prone to overfitting	Reduces overfitting
Accuracy	Moderate	Higher (due to averaging)
Training Time	Fast	Slower (many trees)
Interpretability	Easy	Harder (black-box)
Robustness	Less robust to noise	More robust

---

5. What is confusion matrix? Explain with example.

Definition:

A confusion matrix is a performance evaluation table for classification models, showing actual vs predicted labels.

Structure (Binary Classification):

	Predicted: Positive	Predicted: Negative
Actual: Positive	True Positive (TP)	False Negative (FN)
Actual: Negative	False Positive (FP)	True Negative (TN)

Example:

For 100 emails:

- TP = 40 (spam correctly identified)
- FN = 10 (spam missed)
- FP = 5 (normal marked as spam)
- TN = 45

---

6. Define and differentiate precision, recall, accuracy, and F1-score.

Formulas:

- **Accuracy:**

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

- **Precision:**

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

- **Recall (Sensitivity):**

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

- **F1-Score:**

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

Comparison Table:

Metric	Focus	Best For
Accuracy	Overall correctness	Balanced datasets
Precision	Correct positive predictions	Minimizing false positives
Recall	Detecting actual positives	Minimizing false negatives
F1-Score	Balance of P and R	Imbalanced datasets

---

7. Explain ROC curve and AUC.

ROC (Receiver Operating Characteristic) Curve:

- A plot of **True Positive Rate (TPR)** vs **False Positive Rate (FPR)** at various thresholds.

TPR (Recall):

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

FPR:

$$FPR = \frac{FP}{FP + TN}$$

Interpretation:

- A model that performs well has a curve that reaches the top-left corner.

AUC (Area Under Curve):

- **AUC = 1.0** → Perfect classifier
- **AUC = 0.5** → Random guessing
- **Higher AUC** indicates better model performance.

Diagram Description (for exams):

- X-axis: FPR
- Y-axis: TPR
- Plot a curve rising towards top-left. Show the diagonal (random model) for reference.

Would you like me to move on to **Unit 4 (Clustering, Dimensionality Reduction, PCA, etc.)** next? Let me know if you'd prefer all units compiled into a PDF or formatted into notes.

Awesome! Here's the **Unit 4: Ensemble Learning and Clustering** theory answer key for your Machine Learning subject, presented in a formal and technical format — perfect for university exams. It includes structured definitions, equations, comparisons, examples, and conceptual clarity.

✅ Unit 4: Ensemble Learning and Clustering – Theory Answer Key

---

1. What is Ensemble Learning? Explain Bagging and Boosting techniques.

Definition:

Ensemble Learning is a technique where multiple base models (weak learners) are combined to produce a more accurate and robust prediction model.

✅ Bagging (Bootstrap Aggregating):

- **Idea:** Train multiple models on different random subsets of the training data (with replacement).
- **Purpose:** Reduce variance and prevent overfitting.

Steps:

1. Generate multiple bootstrap samples from training data.
2. Train a base learner (e.g., decision tree) on each sample.
3. Aggregate predictions:
    - Classification: **majority voting**
    - Regression: **average**

Example: Random Forest

✅ **Boosting:**

- **Idea:** Train models **sequentially**, where each new model focuses on correcting the errors of the previous model.
- **Purpose:** Reduce bias and improve accuracy.

Steps:

1. Initialize weights equally.
2. Train a model and calculate error.
3. Increase weights of misclassified points.
4. Train next model on updated weights.
5. Combine learners with weighted majority voting.

Examples: AdaBoost, Gradient Boosting, XGBoost

---

2. Explain Random Forest algorithm.

Definition:

Random Forest is an **ensemble learning** algorithm based on **bagging** using **multiple decision trees** to improve performance and reduce overfitting.

Working Steps:

1. **Bootstrap Sampling:** Draw random samples (with replacement).
2. **Feature Subset Selection:** At each tree split, a random subset of features is chosen.
3. **Tree Training:** Train decision trees independently on bootstrap samples.
4. **Prediction:**
    - Classification: **Majority vote**
    - Regression: **Average**

Advantages:

- Handles high-dimensional data well
- Reduces overfitting
- Supports both classification and regression

---

3. Differentiate between Bagging and Boosting.

Criteria	Bagging	Boosting
Model Training	Parallel	Sequential

Criteria	Bagging	Boosting
Focus	Reduces variance	Reduces bias
Error Handling	Treats all learners equally	Focuses on misclassified instances
Overfitting	Less prone	More prone (but controllable)
Examples	Random Forest	AdaBoost, Gradient Boosting, XGBoost

---

4. What is K-Means Clustering? Explain its working with example.

Definition:

K-Means is an **unsupervised clustering algorithm** that partitions the dataset into **K distinct non-overlapping clusters** by minimizing intra-cluster variance.

Objective Function:

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$$

Where:

- μ_i = centroid of cluster C_i

Algorithm Steps:

1. Select number of clusters k.
2. Initialize k centroids randomly.
3. Assign each point to the nearest centroid.
4. Recompute centroids as mean of assigned points.
5. Repeat steps 3–4 until convergence (no change in assignments).

Example:

For a dataset of coordinates representing customer locations:

- K = 3
- Clusters might represent 3 store regions.

---

5. What is Hierarchical Clustering? Explain.

Definition:

Hierarchical Clustering builds a **tree-like structure (dendrogram)** of clusters either by:

- **Agglomerative (bottom-up)**: Start with each point as a cluster and merge
- **Divisive (top-down)**: Start with all points and recursively split

---

Steps (Agglomerative):

1. Assign each data point to its own cluster.
2. Compute distance matrix.
3. Merge the two closest clusters.
4. Update distances and repeat until one cluster remains.

Distance Metrics:

- **Single Linkage:** Minimum pairwise distance
- **Complete Linkage:** Maximum pairwise distance
- **Average Linkage:** Average pairwise distance

Dendrogram:

A tree diagram that shows merging steps. You can cut it at a height to decide the number of clusters.

---

6. Compare K-Means and Hierarchical Clustering.

Criteria	K-Means	Hierarchical Clustering
Type	Partitional	Hierarchical
Need to specify k	Yes	No (can decide from dendrogram)
Scalability	Fast, efficient for large datasets	Slower (especially agglomerative)
Output	Centroids and cluster assignment	Dendrogram and cluster hierarchy
Deterministic	No (depends on initial centroids)	Yes (deterministic merging)
Use Case	Flat clustering	Hierarchical structure required

Would you like me to now prepare **Unit 5 (Dimensionality Reduction, PCA, SVD, Time Series Forecasting, etc.)** in a similar style? Let me know if you'd prefer this content in a Word, LaTeX, or PDF format for printing or submission too!

Here is your **Unit 5: Dimensionality Reduction and SVM** theory answer key. The content is written in a **formal, technical, and exam-ready** format with equations, step-by-step procedures, comparisons, examples, and use cases.

✅ Unit 5: Dimensionality Reduction and Support Vector Machines – Theory Answer Key

---

1. What is Dimensionality Reduction? Explain with example.

Definition:

Dimensionality Reduction refers to the process of reducing the number of input variables or features in a dataset while preserving as much relevant information as possible.

Purpose:

- Reduce computational complexity.
- Eliminate redundant or irrelevant features.
- Improve model generalization and visualization.

Types:

1. **Feature Selection**: Select a subset of original features.
2. **Feature Extraction**: Create new features by transforming original ones (e.g., PCA).

Example:

In a dataset with 100 features (e.g., pixels of an image), PCA can reduce it to 2 or 3 principal components capturing 95% of variance, useful for visualization or modeling.

---

2. Explain Principal Component Analysis (PCA) with steps.

Definition:

PCA is a statistical technique used for **unsupervised linear dimensionality reduction** by projecting data onto a lower-dimensional space that captures the maximum variance.

Mathematical Steps:

Given a dataset $X \in \mathbb{R}^{n \times d}$:

1. **Standardize the Data:**

$$X' = \frac{X - \mu}{\sigma}$$

2. **Compute Covariance Matrix:**

$$C = \frac{1}{n-1} X'^T X'$$

3. **Compute Eigenvalues and Eigenvectors** of the covariance matrix.
4. **Sort Eigenvectors** by decreasing eigenvalues (variance explained).
5. **Project Data** onto top k eigenvectors (principal components):

$$Z = X' \cdot W_k$$

where W_k contains top k eigenvectors.

Example:

Reduce a 3D dataset (height, weight, age) to 2D (first two principal components).

---

3. What is Support Vector Machine (SVM)? Explain with diagram.

Definition:

SVM is a **supervised learning** algorithm used for classification and regression that finds the **optimal hyperplane** that maximizes the margin between two classes.

Objective:

Maximize the margin $\frac{2}{\|\mathbf{w}\|}$ between the two classes.

Optimization Problem:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to: } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$$

Where:

- $\mathbf{w}$: weight vector
- b: bias
- $y_i \in \{-1, +1\}$

Diagram Description:

- Two classes separated by a **hyperplane**.
- **Support Vectors**: Points closest to the margin.
- Show margin width, separating hyperplane, and two parallel margin lines.

---

4. Differentiate between Linear and Non-linear SVM.

Feature	Linear SVM	Non-linear SVM
Data	Linearly separable	Non-linearly separable
Hyperplane	Straight line (or plane in higher dims)	Curved boundary via kernel transformation
Kernel	Not required	Required (e.g., RBF, polynomial)
Computation	Faster and simpler	Higher computational complexity

---

5. Explain kernel trick in SVM.

Definition:

The **kernel trick** allows SVM to operate in a high-dimensional feature space without explicitly computing the transformation.

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

Where:

- $\phi(x)$: mapping function
- $K(x_i, x_j)$: kernel function

Common Kernels:

1. **Linear Kernel**:

$$K(x, y) = x^T y$$

2. **Polynomial Kernel**:

$$K(x, y) = (x^T y + c)^d$$

3. **RBF (Gaussian) Kernel**:

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

Benefit:

Allows modeling of non-linear decision boundaries with efficient computation.

---

6. Applications of PCA and SVM in real-world ML tasks

Applications of PCA:

- **Face Recognition**: Eigenfaces using PCA to reduce dimensionality.
- **Image Compression**: Reduce size while retaining structure.
- **Data Visualization**: Reduce high-dimensional data to 2D or 3D.
- **Noise Reduction**: Remove low-variance features (assumed noise).

Applications of SVM:

- **Text Classification**: Spam detection, sentiment analysis.
- **Image Classification**: Handwritten digit recognition (MNIST).
- **Bioinformatics**: Cancer diagnosis using gene expression data.

- **Face Detection**: Using sliding window + SVM for face/non-face classification.

Would you like me to compile all **Units 1–5 into a clean PDF format** for revision or submission? I can also include a Table of Contents and diagrams.

Here's the **Unit 6: Artificial Neural Networks and Deep Learning** theory answer key for your Machine Learning subject. It's formatted with detailed, point-wise, and technically formal explanations — ideal for university exam preparation.

✅ Unit 6: Artificial Neural Networks (ANN) and Deep Learning – Theory Answer Key

---

1. Explain architecture of Artificial Neural Networks (ANN).

Definition:

An **Artificial Neural Network (ANN)** is a computational model inspired by the human brain, consisting of interconnected units (neurons) that process information using a connectionist approach.

Architecture Components:

1. **Input Layer:**
   - Takes in feature vectors.
   - Each neuron represents one input feature.
2. **Hidden Layer(s):**
   - Intermediate layers between input and output.
   - Neurons apply weighted sums followed by activation functions.
   - Can be multiple (deep networks).
3. **Output Layer:**
   - Provides final prediction.
   - One neuron for regression; one per class for classification.

Mathematical Representation:

Let input vector be $\mathbf{x} = [x_1, x_2, ..., x_n]$
Output of a neuron:

$$z = \sum_{i=1}^{n} w_i x_i + b, \quad a = \phi(z)$$

Where:

- w_i: weights
- b: bias
- ϕ: activation function
- a: activated output

---

2. What is activation function? List and explain different types.

Definition:

An **activation function** introduces non-linearity into the network, enabling it to learn complex patterns.

Types of Activation Functions:

Function	Formula	Characteristics
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	Output in (0,1); vanishing gradient
Tanh	$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	Output in (-1,1); zero-centered
ReLU	$f(x) = \max(0, x)$	Fast convergence; sparse activation
Leaky ReLU	$f(x) = \max(0.01x, x)$	Prevents dying ReLU problem
Softmax	$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$	Used in multiclass output layer

---

3. Explain Forward and Backward propagation in ANN.

Forward Propagation:

- Input passes layer by layer through the network.
- At each neuron:

$$z = \sum w_i x_i + b, \quad a = \phi(z)$$

- Final output is compared with ground truth using a **loss function**.

Backward Propagation:

- Computes gradients of loss w.r.t weights using **chain rule**.
- Propagates error backward from output to input.
- **Gradient Descent** is applied to update weights:

$$w = w - \eta \cdot \frac{\partial L}{\partial w}$$

Where:

- η: learning rate

- L: loss function

---

4. What is the role of learning rate and loss function?

Learning Rate (η):

- Controls the size of weight updates during training.
- **Too high:** may overshoot optimal value.
- **Too low:** slow convergence.

Loss Function:

- Quantifies the difference between predicted and actual outputs.
- Guides weight updates via backpropagation.

Examples:

- **MSE (Regression):**

$$L = \frac{1}{n}\sum(y - \hat{y})^2$$

- **Cross-Entropy (Classification):**

$$L = -\sum y \log(\hat{y})$$

---

5. Explain Convolutional Neural Networks (CNN) architecture.

Definition:

CNNs are specialized neural networks for processing **grid-like data** such as images. They exploit spatial hierarchies via convolution operations.

Components:

1. **Convolutional Layer:**
   - Applies filters (kernels) to extract features.
   - Output: **feature maps**.

$$(f * x)(i, j) = \sum_m \sum_n f(m, n) \cdot x(i + m, j + n)$$

2. **Activation Layer:**
   - Typically ReLU.
3. **Pooling Layer:**

- Reduces spatial size (e.g., Max Pooling).
- Reduces computation and overfitting.

4. **Flatten Layer:**
   - Converts 2D feature maps to 1D vector.
5. **Fully Connected Layer:**
   - Acts as classifier.
6. **Output Layer:**
   - Softmax for multi-class classification.

Diagram Suggestion:

- Input Image → Conv → ReLU → Pool → Conv → ReLU → Pool → Flatten → FC → Output

---

6. Compare traditional ML algorithms with Deep Learning models.

Feature	Traditional ML (e.g., SVM, DT)	Deep Learning (e.g., ANN, CNN)
Feature Engineering	Manual	Automated (learns features)
Data Requirement	Less data needed	Requires large datasets
Model Complexity	Lower	High (millions of parameters)
Performance	Good for structured data	Excellent for images, text, audio
Computation	Lower	Requires GPUs and large compute
Interpretability	Easier to interpret	Often a "black box"

Let me know if you want:

- A **PDF/LaTeX version** of this full answer key.
- Diagrams for CNNs, SVMs, etc.
- A **condensed 1-page revision sheet** for last-minute prep.

Also, I can generate **viva questions and answers**, or help with expected **university theory exam patterns** if needed.