Machine Learning

Short notes for B.Tech CSE / IT · BCA · MCA · M.Tech

Uncodemy Handwritten Notes · free for students

I. Introduction

- **Machine Learning (ML)** = giving computers the ability to learn from data without being explicitly programmed (Arthur Samuel).
- Tom Mitchell's definition: a program learns from experience **E** for task **T** with performance **P** if P on T improves with E.

EXAMPLE Spam filter. T = classify emails, E = emails labelled spam / not spam, P = percentage correctly classified.

➤ Types of learning

Type	Data	Goal	Examples
Supervised	Labelled (x, y)	Predict y	Regression, classification
Unsupervised	Only x	Find structure	Clustering, PCA
Semi-supervised	Few labels + many unlabelled	Use both	Text labelling
Reinforcement	Reward signal	Best policy	Games, robots

- **Regression** = output is a number (house price). **Classification** = output is a class (spam / not spam).

➤ ML workflow

1. Collect data
2. Clean and preprocess (missing values, encoding, scaling)
3. Split into train / validation / test
4. Choose and train a model
5. Evaluate
6. Tune hyperparameters
7. Deploy and monitor

2. Data Preprocessing

- **Missing values**: drop rows, or fill with mean / median / mode.
- **Encoding**: label encoding (0, 1, 2) for ordered data, one-hot encoding for unordered categories.
- **Feature scaling**:

FORMULA Min-max normalisation: $x' = (x - min) / (max - min)$. Standardisation (z-score): $z = (x - mean) / std$.

EXAMPLE Values 10, 20, 30. Min-max of 20 = (20 - 10) / (30 - 10) = 0.5.

- **Feature selection**: keep useful features. **Feature extraction**: create new ones (e.g. PCA).
- **Curse of dimensionality**: with too many features, data becomes sparse and models need far more data.

3. Bias, Variance and Evaluation

- **Underfitting** = model too simple, high bias, poor on train and test.
- **Overfitting** = model memorises training data, high variance, good on train, poor on test.
- **Bias-variance trade-off**: as model complexity grows, bias falls and variance rises. Aim for the middle.
- Fix overfitting: more data, regularisation, simpler model, cross-validation, pruning, dropout.

➤ k-fold cross-validation

- Split data into k parts. Train on k − 1 parts, test on the remaining one. Repeat k times and average.

➤ Confusion matrix (binary)

	Predicted +	Predicted −
Actual +	TP	FN
Actual −	FP	TN

FORMULA Accuracy = (TP + TN) / total. Precision = TP / (TP + FP). Recall = TP / (TP + FN). F1 = 2PR / (P + R).

EXAMPLE TP = 40, FP = 10, FN = 20, TN = 30. Accuracy = 70/100 = 0.70. Precision = 40/50 = 0.80. Recall = 40/60 = 0.667. F1 = 2(0.8)(0.667)/(1.467) = 0.727.

- Regression metrics: MAE, MSE, RMSE, R^2.
- **ROC curve**: TPR vs FPR. **AUC** = area under it (1 = perfect, 0.5 = random).

4. Regression

➤ Simple linear regression

- Model: $y = b0 + b1 \cdot x$. Find b0, b1 that minimise the sum of squared errors (least squares).

FORMULA $b1 = \Sigma(x - \bar{x})(y - \bar{y}) / \Sigma(x - \bar{x})^2$, $b0 = \bar{y} - b1 \cdot \bar{x}$.

EXAMPLE x = 1, 2, 3; y = 2, 4, 5. $\bar{x} = 2$, $\bar{y} = 11/3 = 3.667$. $\Sigma(x-\bar{x})(y-\bar{y}) = (-1)(-1.667) + 0 + (1)(1.333) = 3$. $\Sigma(x-\bar{x})^2 = 2$. b1 = 1.5, b0 = 3.667 - 3 = 0.667. So $y = 0.667 + 1.5x$.

- **Multiple linear regression**: many inputs, $y = b0 + b1x1 + \ldots + bnxn$.
- **Gradient descent**: update $w = w - \alpha \cdot \partial J/\partial w$, where α = learning rate.
- **Cost (MSE)**: $J = (1/n) \Sigma (y - \hat{y})^2$.

➤ Regularisation

Type	Penalty	Effect
Ridge (L2)	$\lambda \Sigma w^2$	Shrinks weights
Lasso (L1)	$\lambda \Sigma$ abs(w)	Can make weights exactly 0 (feature selection)
Elastic Net	L1 + L2	Mix of both

➤ Logistic regression

- Used for classification, despite its name.
- Output = sigmoid of a linear function: $\sigma(z) = 1 / (1 + e^{(-z)})$, which lies between 0 and 1.

- Predict class 1 if σ(z) >= 0.5. Loss = log loss (binary cross-entropy).

REMEMBER Linear regression → number. Logistic regression → probability → class.

5. Classification Algorithms

➤ k-Nearest Neighbours (kNN)

- Find the k closest training points (usually Euclidean distance) and take a majority vote.
- Lazy learner: no training phase. Needs scaling. Choose odd k for two classes.

FORMULA Euclidean distance $d = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$. Manhattan = $|x1 - x2| + |y1 - y2|$.

➤ Naive Bayes

- Uses Bayes' theorem with the "naive" assumption that features are independent given the class.
- $P(C|x1..xn) \propto P(C) \times P(x1|C) \times ... \times P(xn|C)$. Pick the class with the highest value.
- Good for text classification. Types: Gaussian, Multinomial, Bernoulli.

➤ Decision tree

- Splits data on the feature that gives the most information.

FORMULA Entropy $H(S) = -\Sigma\, p \cdot \log2(p)$. Information gain = H(parent) − weighted average H(children). Gini = $1 - \Sigma\, p^2$.

EXAMPLE 9 Yes, 5 No. $H = -(9/14)\log2(9/14) - (5/14)\log2(5/14) = 0.410 + 0.530 = 0.940$.

- Algorithms: **ID3** (information gain), **C4.5** (gain ratio), **CART** (Gini index).
- Pruning reduces overfitting.

➤ Support Vector Machine (SVM)

- Finds the hyperplane with the maximum margin between classes.
- **Support vectors** = points nearest to the hyperplane.
- **Kernel trick** handles non-linear data: linear, polynomial, RBF (Gaussian).

- **Soft margin** (parameter C) allows some errors.

6. Ensemble Learning

- Combine many models to get a better one.

Method	Idea	Example
Bagging	Train models on random bootstrap samples, then vote / average	Random Forest
Boosting	Train models one after another, each fixing previous errors	AdaBoost, Gradient Boosting, XGBoost
Stacking	A meta-model learns from outputs of base models	Blending

- **Random Forest** = many decision trees + random feature subsets at each split. Reduces variance.
- **Boosting** mainly reduces bias.

7. Unsupervised Learning

➤ k-Means clustering

1. Choose k and pick k initial centroids.
2. Assign each point to the nearest centroid.
3. Recompute each centroid as the mean of its points.
4. Repeat steps 2–3 until centroids stop changing.

- Choose k with the **elbow method** (plot WCSS vs k) or silhouette score.

EXAMPLE Points 1, 2, 10, 11 with k = 2, start centroids 1 and 10. Clusters {1, 2} and {10, 11}. New centroids 1.5 and 10.5. No change next round, so stop.

➤ Other clustering

- **Hierarchical**: agglomerative (bottom-up merge) or divisive (top-down split). Shown as a **dendrogram**.
- Linkage: single (min distance), complete (max), average.
- **DBSCAN**: density-based, finds clusters of any shape, marks outliers as noise.

➤ Dimensionality reduction

- **PCA**: finds new axes (principal components) along directions of maximum variance. Uses eigenvectors of the covariance matrix.
- **LDA**: supervised; maximises separation between classes.

➤ Association rules

> **FORMULA** Support(A→B) = P(A and B). Confidence = Support(A and B) / Support(A). Lift = Confidence / Support(B).

- **Apriori** algorithm: every subset of a frequent itemset is also frequent.

8. Reinforcement Learning and Neural Network Basics

- RL parts: agent, environment, state, action, reward, policy.
- **Exploration vs exploitation**: try new actions vs use the best known one.
- **Q-learning** update:

> **FORMULA** $Q(s,a) = Q(s,a) + \alpha\,[\, r + \gamma \cdot \max Q(s',a') - Q(s,a)\,]$, γ = discount factor.

- **Perceptron**: $y = \text{step}(w \cdot x + b)$. Can learn only linearly separable data (cannot learn XOR alone).
- Multi-layer networks with backpropagation are covered in Deep Learning.

9. Most Asked Questions (practice)

1. Define ML. Supervised vs unsupervised vs reinforcement learning.
2. Explain overfitting, underfitting and the bias-variance trade-off.
3. Find accuracy, precision, recall and F1 from a confusion matrix.
4. Fit a simple linear regression line to given data.
5. Linear vs logistic regression. Explain the sigmoid function.
6. Build a decision tree using entropy and information gain (ID3).
7. Explain SVM with margin, support vectors and kernels.
8. Solve a k-means clustering example by hand.

9. Bagging vs boosting. Explain Random Forest.

10. Explain PCA. What is the curse of dimensionality?

> **TIP** Numericals come from confusion matrix, linear regression, entropy / information gain, Naive Bayes and k-means. Show every step for full marks.