Sara Memon

Supervised & Unsupervised Learning Models

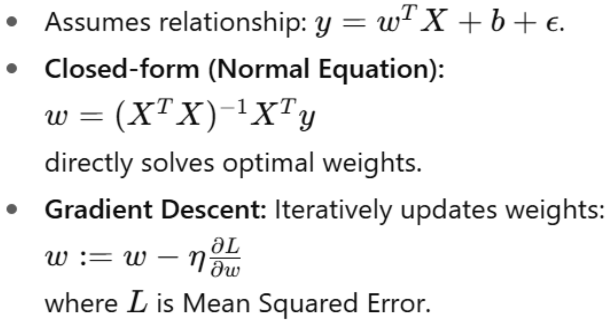
1. **Supervised Learning Models:**

Supervised learning models are a category of machine learning algorithms that learn from labeled datasets to predict outcomes or classify data.

1.1 Linear Regression (Closed-form + Gradient Descent)

Models a continuous target variable as a linear combination of input features. Closed-form solves weights analytically; gradient descent iteratively minimizes mean squared error.

Working Mechanism:



- Assumes relationship: $y = w^T X + b + \epsilon$.
- **Closed-form (Normal Equation):**
  $w = (X^T X)^{-1} X^T y$
  directly solves optimal weights.
- **Gradient Descent:** Iteratively updates weights:
  $w := w - \eta \frac{\partial L}{\partial w}$
  where L is Mean Squared Error.

Pros:

- Simple, interpretable, and fast to train.
- Works well when relationships are linear.
- Closed-form solution gives exact answer (no tuning needed).

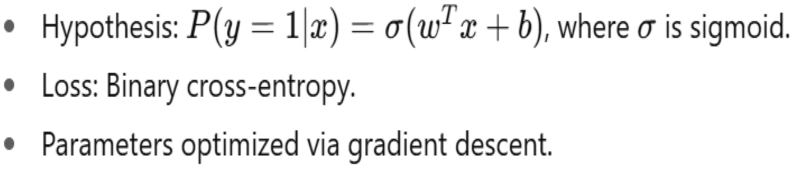
Cons:

- Poor performance on non-linear data.
- Sensitive to outliers and multicollinearity.
- Requires feature scaling for gradient descent.

1.2 Logistic Regression (Batch Gradient Descent)

A linear classifier for binary/multi-class problems that models the probability of class membership using the sigmoid (or SoftMax) function.

Working Mechanism:



- Hypothesis: $P(y=1|x) = \sigma(w^T x + b)$, where σ is sigmoid.
- Loss: Binary cross-entropy.
- Parameters optimized via gradient descent.

Pros:

- Probabilistic output (interpretable).
- Efficient and robust for linearly separable data.
- Works well as a baseline classifier.

Cons:

- Limited to linear decision boundaries.

- Requires feature engineering for non-linear problems.
- Can struggle with high-dimensional or highly correlated features.

1.3 k-Nearest Neighbors (k-NN)

Instance-based classifier: predicts a label based on the majority class among the *k* closest training points in feature space.

Working Mechanism:

- Compute distance (Euclidean, Manhattan, etc.) between query and all training samples.
- Select k closest points.
- Predict majority class (classification) or average (regression).

Pros:

- Simple, no training phase (lazy learning).
- Naturally handles multi-class classification.
- Works well with well-separated clusters.

Cons:

- Computationally expensive at prediction time (distance calculations).
- Sensitive to irrelevant features and scaling.
- Performance degrades with high-dimensional data (curse of dimensionality).

1.4 Decision Tree (CART, Classification)

A tree-structured model that splits data recursively using feature thresholds to minimize impurity (e.g., Gini index).

Working Mechanism:

- Start at root, choose best feature + split (using Gini index/entropy).
- Split dataset into child nodes.
- Repeat recursively until stopping criteria (depth, purity).
- Prediction = majority class in leaf node.

Pros:

- Easy to interpret and visualize.
- Handles non-linear relationships and mixed feature types.
- No feature scaling required.

Cons:

- Prone to overfitting (deep trees).
- Small changes in data can drastically change the tree (unstable).
- Biased toward features with many levels.

1.5 Random Forest (Ensemble of Decision Trees)

Ensemble method that builds many decision trees on bootstrap samples with feature randomness, then averages/votes predictions.

Working Mechanism:

- Bootstrap sampling: each tree trained on a random subset of data.

- At each split, only a random subset of features considered.
- Prediction: average (regression) or majority vote (classification).

Pros:

- Reduces overfitting vs. single tree.
- Handles high-dimensional, non-linear data well.
- Provides feature importance estimates.

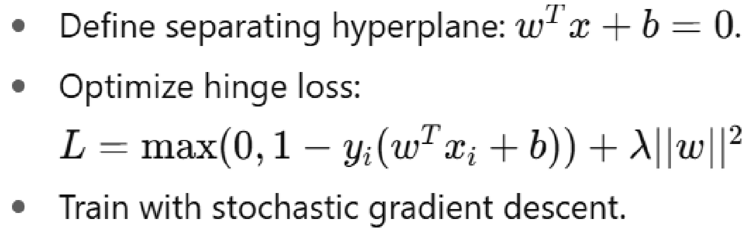
Cons:

- Less interpretable than single tree.
- Slower and more memory-intensive than a single model.
- Still struggles with extrapolation beyond training data range.

1.6 Support Vector Machine (Linear, SGD / Hinge Loss)

Finds the hyperplane that maximizes the margin between classes. Hinge loss penalizes misclassifications and violations of the margin.

Working Mechanism:



- Define separating hyperplane: $w^T x + b = 0$.
- Optimize hinge loss:
  $L = \max(0, 1 - y_i(w^T x_i + b)) + \lambda ||w||^2$
- Train with stochastic gradient descent.

Pros:

- Effective for high-dimensional data.
- Works well with small datasets and clear margins.
- Robust to overfitting (with proper regularization).

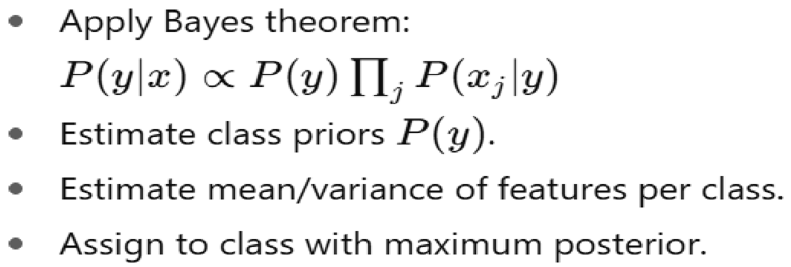
Cons:

- Training can be slow on large datasets.
- Linear version only models linear boundaries (non-linear requires kernels).
- Less interpretable than logistic regression.

1.7 Gaussian Naive Bayes

A probabilistic classifier based on Bayes' theorem, assuming independence between features. Each feature is modeled as a Gaussian within each class.

Working Mechanism:



- Apply Bayes theorem:
  $P(y|x) \propto P(y) \prod_j P(x_j|y)$
- Estimate class priors $P(y)$.
- Estimate mean/variance of features per class.
- Assign to class with maximum posterior.

Pros:

- Extremely fast to train and predict.
- Works well with high-dimensional and small datasets.
- Robust to irrelevant features.

Cons:

- Strong (often unrealistic) independence assumption.
- Poor handling of correlated features.
- Decision boundaries are less flexible than more complex models.

1.8 Multilayer Perceptron (1 Hidden Layer)

A feedforward neural network with one hidden layer, capable of learning non-linear mappings. Trained with backpropagation and gradient descent.

Working Mechanism:

- Input → Weighted Sum → Activation Function → Hidden Layer → Output Layer.
- For binary classification: sigmoid output; for multi-class: SoftMax.
- Loss = cross-entropy.
- Training = backpropagation + gradient descent.

Pros:

- Can approximate complex non-linear relationships.
- Flexible with feature types and scales.
- Extensible to deeper networks for more expressive power.

Cons:

- Requires careful tuning of hyperparameters (learning rate, hidden units).
- Less interpretable than simpler models.
- Prone to overfitting if not regularized (dropout, weight decay).

2. Unsupervised Learning Models:

Unsupervised learning models are a category of machine learning algorithms that analyze and find patterns in unlabeled data without human supervision or explicit instructions.

2.1 k-Means Clustering

Clustering algorithm that partitions data into k clusters by minimizing within-cluster variance (distance to centroids).

Working Mechanism:

- Initialize k centroids.
- Assign each point to nearest centroid.
- Update centroids = mean of assigned points.
- Repeat until convergence.

Pros:

- Simple, fast, and scalable.

- Works well when clusters are spherical and well-separated.
- Easy to implement and interpret.

Cons:

- Requires pre-specifying k.
- Sensitive to initialization (may converge to local minima).
- Struggles with non-spherical or imbalanced clusters.

2.2 Principal Component Analysis (PCA)

Linear dimensionality reduction technique that projects data onto orthogonal directions (principal components) that maximize variance.

Working Mechanism:

- Standardize data.
- Compute covariance matrix.
- Eigen-decompose covariance matrix.
- Select top k eigenvectors as new basis.

Pros:

- Reduces dimensionality and noise.
- Speeds up downstream models.
- Can reveal hidden structure in data.

Cons:

- Linear method — misses non-linear relationships.
- Principal components are hard to interpret.
- Sensitive to feature scaling.

2.3 Gaussian Mixture Model (GMM, EM Algorithm)

Probabilistic clustering model that represents data as a mixture of multiple Gaussian distributions, estimated via Expectation-Maximization.

Working Mechanism:

- Initialize Gaussian parameters (means, variances, weights).
- E-step: Compute probability of each point belonging to each Gaussian.
- M-step: Update parameters based on weighted averages.
- Repeat until convergence.

Pros:

- Soft clustering (probabilistic assignments).
- Captures elliptical/overlapping clusters better than k-means.
- Provides likelihood-based model selection (e.g., AIC, BIC).

Cons:

- Requires choosing number of components.
- Sensitive to initialization, may converge to poor solutions.

- Computationally heavier than k-means.

2.4 Autoencoder

A neural network trained to reconstruct its input by compressing it into a lower-dimensional latent representation and then decoding it back.

Working Mechanism:

- Encoder compresses input into latent representation.
- Decoder reconstructs input from latent representation.
- Loss = reconstruction error (MSE or cross-entropy).
- Can be regularized (denoising, sparsity).

Pros:

- Learns non-linear feature representations.
- Useful for dimensionality reduction, denoising, anomaly detection.
- Can be extended to variational autoencoders (probabilistic).

Cons:

- Requires large datasets and careful tuning.
- Black-box representations (less interpretable than PCA).
- May learn trivial identity mapping without proper regularization.