Summary Report: Fundamentals of Algorithm Design and Machine Learning

Name: Soham Mittal
Roll No: 24BM6JP52
Professor: Aritra Hazra

24 - 28th March 2025

1 SVM Optimization Problem

The Support Vector Machine (SVM) optimization problem is formulated as:

$$\min \frac{1}{2} W^T W$$

subject to:

$$y_i \left(W^T x_i + b \right) \geq 1, \quad \forall (x_i, y_i)$$

where W represents the hyperplane coefficients, b is the bias, and (x_i, y_i) are training data points.

Using the KKT theorem, the dual optimization problem becomes:

$$minL_p = \frac{1}{2} W^T W + \sum_{i=1}^{n} \alpha_i [1 - y_i (W^T x_i + b)]$$

After taking derivatives and setting to zero:

$$L = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (Z_i, Z_j)$$

2 Transformation

When linear classification is not possbile in input space X, data is transformed into a higher-dimensional space Z using a mapping function $\phi(x)$. This enables linear separability in Z-space. It can be transformed back again using $\phi^{-1}(x)$.

For example, transforming a 2D point $X = (1, x_1, x_2)^T$ to:

$$Z = \phi(X) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x_2)$$

3 Kernel Trick

The kernel trick allows computation of dot products in high-dimensional spaces without explicitly computing the transformation $\phi(x)$. This is achieved through kernel functions $K(X, X')$ that directly compute the dot product $\phi(X) \cdot \phi(X')$. For example:

$$K(X, X') = (1 + x_1 x_1' + x_2 x_2')^2$$

This corresponds to a 6-dimensional feature mapping:

$$\phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1 x_2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$$

The optimization problem is reformulated as:

$$L = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(X_i, X_j)$$

Generalization bounds depend on the number of support vectors:

$$E[E_{\text{out}}] \leq \frac{\text{No. of support vectors}}{(N-1)}$$

where N = number of points

4 Radial Basis Function (RBF)

The RBF kernel maps data into an infinite-dimensional space:

$$K(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}} = e^{-\gamma \|x-x'\|^2}$$

Where $\gamma = \frac{1}{2\sigma^2}$ controls the width of the Gaussian curve.

Using Taylor series expansion, the RBF kernel can be written as:

$$K(x, x') = e^{-\|x\|^2} e^{-\|x'\|^2} \sum_{k=0}^{\infty} \frac{(2)^k (x \cdot x')^k}{k!}$$

This kernel maps data into infinite-dimensional space and captures complex patterns effectively. The RBF kernel's flexibility makes it suitable for non-linear decision boundaries.

5 Conditions for a Valid Kernel

5.1 Structural Consistency

The feature mappings $\phi(x)$ and $\phi(x')$ must have the same structure for the inner product to be well-defined.

5.2 Mercer's Theorem

A symmetric function $K(x, x')$ is a valid kernel if:

- It is symmetric: $K(x, x') = K(x', x)$
- The kernel matrix is positive semi-definite: $\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j K(x_i, x_j) \geq 0$ for any vector $c \in \mathbb{R}^N$

6 Dimensionality reduction

Dimensionality reduction aims to reduce the number of features in a dataset while preserving essential information. It addresses computational efficiency and the curse of dimensionality in machine learning. The transformation maps data from D-dimensional space to d-dimensional space ($1 \leq d \leq D$).

7 Feature Selection

Feature selection identifies a subset of original features relevant to a task, using metrics like Kullback-Leibler (KL) Divergence to measure information gain.

7.1 Best Subset Selection: Forward

Forward selection starts with an empty feature set and iteratively adds features that improve model performance based on KL Divergence. The computational complexity is $O(D^d)$.

7.2 Best Subset Selection: Backward

Backward selection begins with all features and iteratively removes less useful ones using KL Divergence. Computational complexity is also $O(D^d)$.

8 Feature Extraction

Feature extraction transforms original features into new ones, capturing critical information.

8.1 Principal Component Analysis (PCA)

PCA is an unsupervised method that identifies principal components to maximize variance in data. Steps include:

1. Centering data by subtracting the mean.
2. Computing the covariance matrix.
3. Performing eigen decomposition to find eigenvalues and eigenvectors.
4. Selecting top eigenvectors as principal components.
5. Projecting data onto principal components.

Limitations:

- Assumes linear relationships; may not capture nonlinear structures.
- Interpretation of principal components can be challenging.

8.2 Linear Discriminant Analysis (LDA)

LDA is a supervised method for dimensionality reduction in classification problems. It maximizes separation between classes by projecting data onto a vector $\mathbf{w}$. Fisher's linear discriminant is $\mathbf{w}$ that maximises:

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$

where m_1, m_2 are class means after projection, and s_1^2, s_2^2 are scatter measures.

9 Performance Metrics

The evaluation of hypotheses involves understanding how accurately unknown test data is classified, estimating performance metrics, and comparing models based on these metrics.

9.1 Confusion Matrix

A confusion matrix is a tool used to evaluate the performance of a classification algorithm. For a 2-class classification, it helps in identifying true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

9.2 Metrics

- **Accuracy**: Measures the ratio of correctly predicted observations.

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

- **Precision**: Indicates the proportion of positive predictions that are correct.

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

- **Recall**: Also known as sensitivity or true positive rate.

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

- **F Score**: Harmonic mean of Precision and Recall.

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

- **Weighted Accuracy**: Averages out biases in metrics by using weights.

$$\text{WeightedAccuracy} = \frac{w_1 \cdot |TP| + w_4 \cdot |TN|}{w_1 \cdot |TP| + w_2 \cdot |FP| + w_3 \cdot |FN| + w_4 \cdot |TN|}$$

10 Methods of Estimation

The effectiveness of evaluation methods depends on: Class distribution in the dataset, Size of training and test sets, Cost of misclassification. Common approaches are -

- **Holdout Method**: Typically uses 2/3 of data for training and 1/3 for validation
- **Random Sub-Sampling**: Randomly selects instances for training and testing sets
- **Stratified Sampling**: Maintains class distribution proportions when sampling
- **K-fold Cross-Validation**: Divides the dataset into K equal parts, using K-1 parts for training and 1 part for validation, repeated K times

11 Model Comparison

The Receiver Operating Characteristic (ROC) curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) across different threshold settings. AUC (Area Under Curve) gives predictive power:

- AUC = 1.0: Perfect classifier
- AUC = 0.5: No discriminative power (equivalent to random guessing)
- Higher AUC indicates better model performance

The optimal threshold depends on the specific application context and the relative costs of false positives versus false negatives.

- Lower threshold: Increases TPR but also increases FPR (more sensitive, less specific)
- Higher threshold: Decreases FPR but also decreases TPR (more specific, less sensitive)

12 Learning from Data

Machine learning algorithms aim to derive a hypothesis set H from training data $[(x_1, y_1), \ldots, (x_n, y_n)]$. The final hypothesis $g(x) \to \hat{y}$ minimizes total error using methods like stochastic and batch gradient descent. Errors are categorized as:

- **In-sample error** (E_{in}): Error rate on training data.
- **Out-of-sample error** (E_{out}): Error rate on unseen data.

12.1 Learning Framework

The learning process involves:

1. Unknown target function $f(x) \to y$,
2. Training examples simulating $f(x)$,
3. Hypothesis set H minimizing error,
4. Final hypothesis $g(x) \to \hat{y}$,
5. Probability distributions for training/testing feasibility.

12.2 Error Estimation

There are 2 primary types of errors:

- In-sample error:

$$E_{in}(h) = \frac{1}{N} \sum_{n=1}^{N} e(h(x_n), f(x_n))$$

- Out-of-sample error:

$$E_{out}(h) = \text{Exp}_x[e(h(x_n), f(x_n))]$$

Error measures can be:

- Squared error: the square of the difference between predicted and actual values
- Binary error: for classification problems (1 for mismatch, 0 for match)

12.3 Noise in Data

Noise represents randomness or irrelevant information in datasets. It transforms deterministic targets into probabilistic distributions:

$$P(y|x) = p(x, y),$$

where noisy targets are modeled as:

$$f(x) = \mathbb{E}(y|x) + (y - f(x))$$

The modified learning diagram incorporates noise effects.

13 Theoretical Considerations

The inductive principle assumes $E_{\text{out}}(g) \approx E_{\text{in}}(g)$. The probability of deviation is bounded by:

$$P(|E_{\text{out}} - E_{\text{in}}| > \epsilon) \leq 2Me^{-2\epsilon^2 N},$$

where M is the number of hypotheses.

Model complexity affects generalization:

- Increased complexity reduces E_{in}.
- Overfitting occurs when generalization diminishes beyond a threshold (d_{vc}).

14 Goal of Learning

The goal of learning is to ensure that the output error $E_{out}(g)$ approximates the input error $E_{in}(g)$, i.e., $E_{out}(g) \approx E_{in}(g)$. This is achieved when:

1. $E_{out}(g) \approx E_{in}(g)$.
2. $E_{in}(g) \approx 0$.

As model complexity increases, $E_{in}(g)$ decreases, but the difference $E_{out}(g) - E_{in}(g)$ increases. A trade-off is necessary to optimize learning.

15 Growth Function

The growth function $m_H(N)$ is defined as the maximum number of dichotomies (ways to classify points) possible in a given training space when points are arranged in the worst possible configuration:

$$m_H(N) = \max_{X_1, X_2, ... X_N \in X} |H(X_1, X_2, ..., X_N)|$$

15.1 Growth Functions for Specific Cases

Examples of growth functions include:

1. **2D Perceptrons:** The growth function satisfies $m_H(N) \leq 2^N$.
2. **Positive Rays:** Dichotomies are given by $m_H(N) = N + 1$.
3. **Positive Intervals:** Dichotomies are given by $m_H(N) = \binom{N+1}{2} + 1$.
4. **Convex Sets:** Dichotomies are always 2^N.

15.2 Break Point

A break point k is the minimum size of a dataset for which $m_H(k) < 2^k$. In other words, no dataset of size k can be shattered by hypothesis set H.

- For 2D perceptrons: $k = 4$.
- For positive rays: $k = 2$.

- For positive intervals: $k = 3$.
- For convex sets: No finite break point ($k = \infty$).

15.3 Polynomial Bound on Growth Function

$m_H(N)$ is bounded by a polynomial when a finite break point exists:

$$m_H(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

This bound follows a pattern similar to Pascal's triangle and can be proven by mathematical induction.

Examples include:

1. Positive Rays (break point $k = 2$): $m_H(N) \leq 1 + N$
2. Positive Intervals (break point $k = 3$): $m_H(N) \leq 1 + N + \frac{N(N-1)}{2}$
3. 2D Perceptrons (break point $k = 4$): $m_H(N) \leq 1 + N + \frac{N(N-1)}{2} + \frac{N(N-1)(N-2)}{6}$