Weekly report of lessons

Name: Suryam Arnav Kalra
Roll No: 19CS30050
The week: 18th October 2021 to 22th October 2021

The topics covered:

- Discriminant functions
- Interpretation of $W^T X$ and Linearly separable classes
- Gradient descent method for iterative optimization
- Other forms of the error function
- Batch relaxation with margin algorithm
- Support Vector Machine (SVM) and SVM-Testing
- Convex quadratic optimization problem and Dual optimization problem
- Slack variable and soft margin hyperplane and optimization problem
- Projecting to higher dimensional space
- Kernel machines and Vectorial kernel functions
- Parametric discrimination and The logit function
- Learning weights algorithm
- Artificial Neural Network and Mathematical description of ANN
- Optimization problem, ANN training and Improving convergence

Summary topic wise:

- Discriminant functions: We define a set of functions (linear / quadratic) in which we choose C_i if $g_i(x) = max_j\ g_j(x)$. A linear function has $O(d)$ storage whereas a quadratic function has $O(d^2)$ storage. For two classes, we assign C_1 if g(x) > 0 else C_2.
- Interpretation of $W^T X$: Consider the hyperplane $W^T X$ = 0 separating two samples of classes 1 and 2 and then we assign class 1 if $W^T X$ > 0 else we assign class 2.
- Linearly separable classes: Classes are linearly separable if for classes each class C_i, there exists a hyperplane H_i such that all $x \in C_i$ lie on its positive side and all $x \in Cj, j != i$ lie on its negative side.
- Gradient descent method: This method is used to iteratively find the global minimum of a convex function. We start with an initial vector W, and compute its gradient and move closer to the minimum by updating W. In this way, we obtain the weight vector W which minimizes the error J(W).
- Other forms of the error function: There could be other forms of the error function
  - $J_p(W) = \sum -W^T Y$, it is not a continuous function.
  - $J_q(W) = \sum (W^T Y)^2$, it a continuous function with a very smooth boundary.
- Batch relaxation with margin algorithm: Linear SVM maximize the margin of separation between two separable data points of classes by initializing W and using a slack variable and iterating till convergence of W (very little change in updates).
- SVM: It is a linear discriminant classifier and uses the Vapnik's principle (never solve a more complex problem as a first step before the actual problem). It provides maximum margin based linear discrimination for two linearly separable classes.

- Convex quadratic optimization problem: In this we tend to have a convex objective function which has a global minima along with linear constraints. We use the lagrange's method of multipliers to find this global minima.
- Dual optimization: To be maximized w.r.t Lagrange multipliers (> 0) subject to that gradients w.r.t $w\ and\ w_0$ should be 0. We can use the dual problem as well to get the solution for the original problem. Most of the α^t(multipliers) will be zero.
- SVM-Testing: We check only the sign of discriminant value and let only support vectors decide class boundaries (other samples do not influence the classifier).
- Slack variable and soft margin hyperplane: Since the classes may not be linearly separable we use a slack variable and tend to minimize $w^2/2$ subject to $r^t(w^t x + w_0) >= 1 + s^t$ for all t with the constraint $0 < s^t < 1$ (x^tcorrectly classified), $s^t >= 1$ (x^t wrongly classified).
- Optimization problem: Add penalty term for soft error to define the objective function for minimization $L_p = \frac{w^2}{2} + C \sum s^t$ where C is the penalty factor.
- Projecting to higher dimensional space: We use basis function $z = \varphi(x)$ where g(z) = $w^t z$ for projecting the data points to higher dimensional place which may make them linearly separable.
- Kernel machines: Discriminant function $g(x) = \sum \alpha^t r^t K(x^t, x)$ where $K(x^t, x)$ is the kernel. Gram matrix is the matrix of kernel values K, where $K_{t,s} = K(x^t, x^s)$ and it should be symmetric and +ve semidefinite.
- Vectorial kernel functions: Polynomial of degree q, Radial basis function, Mahalanobis kernel function, and distance based function can be used as some kernel functions.
- Parametric discrimination: We know that if the class densities, $p(x|C_i)$, are Gaussian and share a common covariance matrix, the discriminant function is linear $g_i(x) = {w_i}^t x + w_{i0}$ and the parameters can be analytically calculated.
- logit function: . $Log(y/(1 - y))$ is known as the logit function or log(odds y). In the case of two normal classes sharing a common covariance matrix, the logit is linear.
- Learning weights algorithm: Assume $initial\ W\ and\ w_0$ and compute $y = sigmoid(w^t x + w_0)$ and compute the gradients. Using them, update $W\ and\ w_0$ and continue these steps till convergence.
- Artificial neural networks: It is a network of perceptrons which provides a powerful model describing input/ output relations and learns non-linear relations in data.
- Mathematical description of the model: Let j^{th} neuron of i^{th} layer by $ne_j^{(i)}$, and the corresponding weights $w_j(i) = (w_{j1}(i), w_{j2}(i), \ldots w_{j(n-1)}(i))$ and the bias $w_{j0}(i)$, where n_i, is the dimension of output at i^{th} layer. Output of the neuron: $y_j(i) = f(w_j(i)^T X^{i-1} + w_{j0}(i))$.
- Optimization problem: We have the error function $J(W) = \frac{1}{N} \sum (O_i - F(X_i; W))^2$ and we can apply the same gradient descent algorithm to obtain the solution.
- ANN training: Initialize W(0) and for each training sample (X_i, O_i) compute functional values doing a forward pass of the network and update weights of each line using a pass of back propagation from output layer towards the input.
- Improving convergence: We can use an adaptive learning rate and along with it we can use running average of weight updates to be added with the gradient to avoid abrupt changes in consecutive iterations.

Concepts challenging to comprehend:

Gradient descent method and SVM are a little bit challenging to comprehend.

Interesting and exciting concepts:

Artificial neural networks are quite interesting and exciting to learn.

Concepts not understood:

After going through the book and the video lectures the concepts are understood.

Any novel idea of yours out of the lessons:

I think that not only we can make data linearly separable by projecting to higher dimensions but we can change our coordinate system as well which may make the data separable. Instead of the common Cartesian system, we can use polar coordinates (r, θ, φ) or even cylindrical coordinates which may cover represent data differently and help avoiding the computational overload of doing the PCA. I think this technique may also be used in SVM to get the maximum margin of separable classes.