IAML DL - Study Guide - Week 5

Sambit Paul, Pavlos Andreadis, Nigel Goddard

January 2022

1 Introduction

Week 5 introduces the concepts of Optimisation and Regularisation of machine learning models and why they are essential. Optimisation deals with the *'how'* the machine learning model learns, while Regularisation aims to solve the over-fitting problem.

Along side that, we will also take a look at one of the most popular machine learning models - Support Vector Machines (SVMs). As stated by Andrew NG in this article, SVMs are among the best "off-the-shelf" supervised learning algorithm in use.

2 Optimisation and Regularisation

2.1 Optimisation

- Optimisation is the process of using an error or cost metric to find the most optimal solution to a given problem. The process of optimisation can either be arithmetic, or it can be computed stochastically.
- In Machine Learning, we typically use an optimisation process in order to solve the problem of which model parameter assignments make our model best fit the training data (indicated by which parameter assignment minimises our error metric). The problem is known as *model fitting*. Of course there is more nuance here, as we do not typically want the model to perfectly fit our training data (consider Generalisation, and Regularisation below). To read more about the different methods for model fitting, Murphy [2012] Section 8.3 gives a brief introduction to various methods.
- An error manifold is the space of all possible values to your objective function in the optimisation problem (assuming you have a criterion that needs to be minimised, such as a metric of how wrong your predictions with the model are). Given a specific set of samples on which we are comparing, the error manifold will depend on the different possible values

our model parameters can take. A brief explanation of error manifolds is provided in this Jupyter Notebook.

- *Cost function*, *loss function*, *error function*, and *objective function* are synonyms, though the first 3 imply that we are dealing with a minimisation problem, while *objective function* is more general. Typically we are looking to define a cost function which we need to minimise. Refer to Goodfellow et al. [2016] Section 4.3 to understand the basic idea behind convex optimisation.
- To understand and know more about different loss functions in brevity, you can refer to this article. Also, try understanding how each loss function is suited to classification and regression problems. Generally, you might want to introduce further terms to your loss function; this is an engineering problem in and of itself (for example, see Regularisation below).
- Gradient descent has been well explained in Goodfellow et al. [2016] Section 5.2.4 and the key equation behind gradient based optimisation is given in Equation 5.41.
- Learning rate is a hard parameter to tune, and Murphy [2012] Section 8.3.2 on Page 247 aims to explain how it works.
-
  - Batch gradient descent computes the gradient using the whole dataset.
  - Stochastic gradient descent (SGD) computes the gradient using a single sample randomly chosen over each epoch from the dataset.
  - Best of both worlds, is to use a mini-batch (set of samples chosen randomly from the dataset) and perform Batch descent over that mini-batch.

2.2 Regularisation

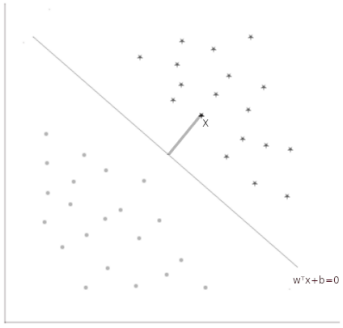
- The basic idea of regularisation is well articulated in Barber [2012] where he explains it as, "For most purposes, our interest is not just to find the function that best fits the train data but one that will generalise well. To control the complexity of the fitted function we may add an extra regularising term to the train error to penalise rapid changes in the output".
- To go through the method of finding weights when L2 regularisation is used in linear regression, you might want to use Bishop [2006] Pg 144. The equations is $w = (\Phi^T.\Phi + \lambda.I)^{-1}.\Phi^T.\mathbf{y}$. This is *Ridge Regression*.
- A clear explanation of L2 Regularisation (Ridge Regression) is given in Goodfellow et al. [2016] Section 7.1.1.
- Additionally, another important form of regularisation is L1 regularisation or what is known as *Lasso Regression*. It is useful to find sparse matrices

of weights. lease use Goodfellow et al. [2016] Section 7.1.2 to read how it works.

- To have a simplified understanding of L1 regularisation, this article is quite useful.

3 Support Vector Machines 1

- The intuition is to learn a decision boundary (hyperplane) to separate classes of data such that margin between the training points on either classes is maximised.
- **Intuition behind SVM:**
  For a binary classification problem with 2 classes (say, +1 and -1), we need to:
  - Predict +1 iff $w^T x + b \geqslant 0$
    If $w^T x + b \gg 0$, confidence of class = +1 is very high.
  - Predict -1 iff $w^T x + b < 0$
    If $w^T x + b \ll 0$, confidence of class = -1 is very high.
- **Derivation of Geometric Margin of SVM:**
  Let us consider the distance between the training sample X and the decision boundary $w^T x + b = 0$ as γ. Since the *weight vector* **w** is orthogonal to the decision boundary, the projection of the point X on the decision boundary can be written as $(x - \gamma.\frac{w}{\|w\|})$. Let this point be called $\hat{X}$.
  Therefore, if $\hat{X}$ is put in the equation of the decision boundary, we should have $w^T \hat{X} + b = 0$. Therefore, we get the equation:



$$w^T(x - \gamma.\frac{w}{\|w\|}) + b = 0 \tag{1}$$

Solving Equation 1 for γ, we get:

$$\gamma = \frac{w}{\|w\|}\hat{X} + \frac{b}{\|w\|}$$

This is called the geometric margin of point X with respect to the decision boundary. The division of the hyperplane by the Euclidean norm of the weight vector is to ensure scaling the weights does not affect the margin.

- To understand why weights are always orthogonal to the decision boundary, please take a look at this article.
- To understand how to the math for maximising the margin works, please refer to Barber [2012] Section 17.5.1 and for an more in-depth explanation please refer to, Bishop [2006] Section 7.1.

4 Support Vector Machines 2

Part 2 continues with the concepts of Support Vector Machines introducing overlapping class distributions and how to modify the algorithm to deal with them and make them more robust. We will also deal with the usage of kernels to create non-linear SVM classifiers. This article provides a high level understanding of the importance of kernels in the field of machine learning

- Derivation of optimal parameters using Lagrange multipliers:

  $g(x) = |w|^2 \Rightarrow \frac{dg}{dw} = 2|w|$
  $f(x) = \sum\{y_i(w^T x_i + w_0) - 1\} \Rightarrow \frac{df}{dw} = \sum y_i x_i$

  Using Lagrange Multiplier, we can say:
  $g(x) = \lambda f(x)$
  $\Rightarrow 2|w| = \sum \lambda_i y_i x_i$
  $\Rightarrow |w| = \sum \alpha_i y_i x_i$ assuming $\alpha_i = \frac{\lambda_i}{2}$

  This concept has also been explained thoroughly in Bishop [2006]. Please refer to Section 7.1 Pg 328.

- For linearly non-separable data, creating a solution which gives an exact separation will not be generalisable. This requires the need to allow some data points to be misclassified. Please refer to Bishop [2006] Section 7.1.1 for more details on this.
- For a quicker introduction to the use of ξ_n for making SVMs more robust, please refer to Section 17.5.1 from Barber [2012].
- To understand the influence of the parameter C in SVM classification, this StackOverflow article provides a very good explanation.
- Following the 2-Norm Soft-margin subsection under Section 17.5.1 from Barber [2012], it will be clear that the optimisation problem requires only the inner product. A simpler derivation for the optimisation equation is provided here:
  $f(x) = \frac{1}{2} w^T w$ and $g(x) = y_n(w^T x_n + w_0) - 1$ Hence, using Lagrange multipliers, we can say that the optimisation problem is:
  $L(w, w_0) = f(x) + \sum \alpha_n g_n(w, w_0)$
  This implies,
  $\frac{dL}{dw} = w - \sum_n \alpha_n y_n x_n = 0$ and $\frac{dL}{dw_0} = 0 - \sum_n \alpha_n y_n = 0$

Filling in the values in the original optimisation equation, we get:
$L(w, w_0) = \sum_n \alpha_n - \frac{1}{2} w^T w \Rightarrow L(w, w_0) = \sum_n \alpha_n - \frac{1}{2} \sum_{n,m} \alpha_m \alpha_n y_n x_n^T x_m y_m$
Hence, the optimisation equation depends on the $x_n^T x_m$ which is basically an inner product. If x is replaces with the basis function, we get $\phi(x_n)^T \phi(x_m)$. This can be represented as $K(x_n, x_m)$ and are called kernel functions. Kernel function basically calculate the inner product in the transformed space.

- The conditions to determine which functions can be considered as Kernel functions is defined using Mercer's theorem.

References

David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

Christopher M Bishop. *Pattern recognition and machine learning*. Springer Science+ Business Media, 2006.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.