Introduction to Machine Learning

Lecture 5 - Neural Networks

Guang Bing Yang, PhD

yguangbing@gmail.com, Guang.B@chula.ac.th February 19th, 2021

Assignment2 — extended due date to Feb 26

- The classification problem includes decision making algorithm.
  - No decision making in regression problem although most of processes are the same in both problems. The classification problems need an activation function.
  - It can be a sigmoid or softmax or, just simple as a logical formula as:
    - $1, \text{ if } \hat{y} \geq 0.5$
    - $0, \text{ if } \hat{y} < 0.5$
  - Thus, the decision making expression or algorithm needs in the cost function, gradient, and prediction functions.
  - e.g., if the $\begin{matrix} 1, & \text{if } \hat{y} \geq 0.5 \\ 0, & \text{if } \hat{y} < 0.5 \end{matrix}$ is the decision making function, the cost function looks like this:
  - where n is the number of training data points (note that it is not the number of dimensions of the data)
  - The output of the cost function is a scalar number, divided by n is mainly to avoid the overflow of the numerical computation.

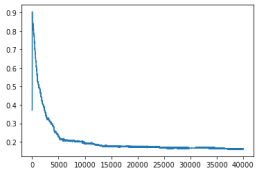
```
def costw(self, x, y):
    n = len(x)
    delta = np.dot(x, self.W)
    y_hat = np.array([1 if p >= 0.5 else 0 for p in delta])[:, np.newaxis]
    rss = np.sum((y-y_hat)**2)/n - 0.5*np.dot(self.W.T, self.W).flatten()[0]
    return rss
```

yguangbing@gmail.com, Guang.B@chula.ac.th February 19th, 2021

Assignment2 — extended due date to Feb 26

- In your gradient function, the same decision making algorithm has to be applied and a negative sign needs to be added, like this: return o-(np.dot(x.T,(y-y_hat))/n-self.W), and in the training function, the update process needs to change corresponding to: self.W -=lr*dw, Note that it is minus rather that plus.
- In addition, because the bias parameter was merged into the parameters W (self.W = (np.zeros(m+1)+1e-3)[:, np.newaxis]), you need to add a dummy ones in your data X
- You need tuning your training parameters, learning_rate and iteration of trainings
  - several good choices for learning_rate (lr) are [1e-5, 1e-6, and 1e-7], and training loops are [N=40000 to 100000].
  - A loss value plot looks like the figure 2, and overall testing accuracy is 0.67. Not that good because we have only 426 training data samples and there are 30 dimensions of the data, in addition, we haven't applied any pre-processing for the data.

```
def trainw(self, x, y, lr=1e-9, N=20000):
    ls=[]
    X = np.hstack((np.ones(x.shape[0])[:, np.newaxis], x))
    y = y[:, np.newaxis]
    for i in range(N):
```



yguangbing@gmail.com, Guang.B@chula.ac.th February 19th, 2021

Introduction to Neural Networks

- In this lecture, we will introduce the Neural Networks and focus on:
  - the forward propagation algorithm, and
  - cost function,
  - the network training, also
  - the backpropagation algorithm, and
  - briefly introduce the neural network model and architecture as well as.

yguangbing@gmail.com, Guang.B@chula.ac.th February 19th, 2021

Background and Goals

- The models for regression and classification consist of linear combinations of fixed basis functions.
- Linear models have useful analytical and computational properties, but not good for real data.
- The natural properties of the real data--which are non-linear and large-scale dimensionality--into the practices.
- SVMs, discussed in the previous lecture, address this problem by using kernel function and the support vectors.
- The advantages of SVMs are significant but shortcomings limit its applications in practice.

Background and Goals

- RVM overcomes some of disadvantages of SVMs, but its computational cost limits its applications in practice.
- There is a need for an alternative approach to overcome these disadvantages. Here is the Neural Network.
- The intuition comes out the feed-forward neural network, also known as the multilayer perceptron.
- The term 'neural network' came from biological systems. Machine learning focus on neural networks as efficient models for statistical pattern recognition.

Basic Concepts

- The purpose is to find useful representation of the target variables:
  - $\mathrm{t} = \mathrm{y}(\mathrm{x}, \mathrm{w}) + \epsilon(\mathrm{x})$, where
    - $\mathrm{t} = (t_1, \ldots, t_N)$ and $\mathrm{x} = (x_1, \ldots, x_N)^T$ are the observations.
    - $\epsilon(\mathrm{x})$ is the residual error.
- For example, to a linear model:
  - $y(\mathrm{x}, \mathrm{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathrm{x}) \right)$
    - $\phi = (\phi_0, \ldots, \phi_M)^T$ is the fixed model basis functions.
    - $\mathrm{w} = (w_0, \ldots, w_M)^T$ are the model parameters—also called coefficients.
    - For regression: f(.) is the identity function.
    - For classification: f(.) is an non-linear activate function.

Basic Concepts — Feed-forward Neural Networks

- Feed-forward Neural Networks generalize the linear model:
  - $y(\mathrm{x}, \mathrm{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathrm{x}) \right)$, where
    - the goal of feed-forward is to let the basis itself, as well as the coefficients w_j, will be adapted.
    - In other words, make the basis functions depend on the parameters.
    - The network uses the same form of the basis function
    - The basis function is a non-linear function of a linear combination of the inputs.

Basic Concepts — Feed-forward Neural Networks

- First, construct M linear combinations of the input variables $x_1, \ldots, x_D$ in the form:
  - $a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}$, where
    - a_j are the activations, $j = 1,...,M$.
    - $w_{ji}^{(1)}$ are weights for layer 1, where $i = 1,...,D$.
    - $w_{j0}^{(1)}$ are the biases for the layer 1.
    - Each linear combination a_j is transformed by a (nonlinear differentiable) activation function:
      - $z_j = h(a_j)$

Basic Concepts — Feed-forward Neural Networks

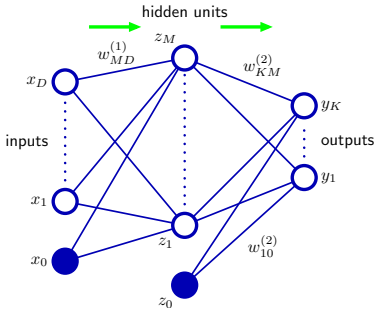
- The output activations of the hidden layer $z_j = h(a_j)$ are linearly combined in layer two:
  - $a_k = \sum_{j=0}^{M} w_{kj}^{(2)} z_j$, where
    - a_k are the output activations, $k = 1,...,K$.
    - $w_{kj}^{(2)}$ are weights for layer 2, where $j = 1,...,D$.
    - $w_{k0}^{(2)}$ are the biases for the layer 2.
    - The output activations a_k are transformed by output activation function:
      - $y_k = \sigma(a_k)$
        - y_k are the final outputs.
        - $\sigma(a)$ is a sigmoidal function (for binary classification)

Basic Concepts — Feed-forward Neural Networks

- The complete two layer model:
  - $y_k(\mathrm{x}, \mathrm{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \right) + w_{k0}^{(2)} \right)$, where
    - $h(\,.\,)$ is the basis or activation function and $\sigma(a)$ are sigmoidal functions, e.g., the logistic function.
    - Again, for regression, the $\sigma(a)$ becomes to the identity.
- Absorb the biases $w_{k0}^{(2)}$ and $w_{j0}^{(1)}$ into the weight sets, we get the compact form:
  - $y_k(\mathrm{x}, \mathrm{w}) = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i \right) \right)$
- Evaluation of the above model (network) is called forward propagation.

Basic Concepts — Feed-forward Neural Networks

- This two-layer network diagram is given as right Figure.
- The approximation process can be represented by a network:
  - Nodes are input, hidden and output units. Links are corresponding weights.
  - Information propagates 'forwards' from the explanatory variable x to the estimated response $y_k(\mathrm{x}, \mathrm{w})$.



Basic Concepts — Feed-forward Neural Networks

- The Properties and generalizations:
  - Normally $K \le D \le M$, which means that the network is redundant if all $h(\cdot)$ are linear.
  - There may be more than one layer of hidden units.
  - Individual units need not be fully connected to the next layer.
  - Individual links may skip over one or more subsequent layers.
  - Networks with two or more layers are universal approximations.
  - Any continuous function can be uniformly approximated to arbitrary accuracy, given enough hidden units.
  - This is true for many definitions of $h(\cdot)$, but excluding polynomials.
  - There may be symmetries in the weight space, meaning that different choices of w may define the same mapping from input to output.

Basic Concepts — Feed-forward Neural Networks

- Maximum Likelihood Parameters:
  - Maximum likelihood is the same as minimizing the residual error between $y_k(\mathrm{x}, \mathrm{w})$ and t_n.
  - Let the target be a scalar-valued function, which is Normally distributed around the estimate:
    - $p(t \mid \mathrm{x}, \mathrm{w}) = \mathbb{N}(t \mid y(\mathrm{x}, \mathrm{w}), \beta^{-1})$
  - Consider the sum of squared-errors: $E(\mathrm{w}) = \frac{1}{2}\sum_{n=1}^{N} (y(\mathrm{x}_n, \mathrm{w}) - t_n)^2$
  - The maximum-likelihood estimate of w can be obtained by (numerical) minimization:
    - $\mathrm{w}_{ML} = \min_w E(\mathrm{w})$
    - After get the w_{ML}, the precision, β can also be estimated. E.g. if the N observations are i.i.d. (Independent and identically distributed random variables), then their joint probability is:

      $p(\mathrm{t} \mid \mathrm{x}, \mathrm{w}, \beta) = \prod_{n=1}^{N} p(t_n \mid \mathrm{x}_n, \mathrm{w}, \beta)$

Basic Concepts — Feed-forward Neural Networks

- Maximum Likelihood Parameters:
  - The negative log-likelihood, in this case, is:
    - $-\log p(\mathrm{t} \mid \mathrm{x}, \mathrm{w}, \beta) = \beta E(\mathrm{w}_{ML}) - \frac{N}{2}\log\beta + \frac{N}{2}\log 2\pi$
  - By obtain the derivative $d/d\beta = E(\mathrm{w}_{ML}) - \frac{N}{2\beta}$, we have:
    - $\frac{1}{\beta_{ML}} = \frac{2}{N}E(\mathrm{w}_{ML})$
  - For $K > 2$ target variables, $\frac{1}{\beta_{ML}} = \frac{2}{NK}E(\mathrm{w}_{ML})$

Basic Concepts — Feed-forward Neural Networks

- Parameter Optimization:
  - Iterative search for a local minimum of the error:
    - $\mathrm{w}^{(\tau+1)} = \mathrm{w}^{(\tau)} + \Delta\mathrm{w}^{(\tau)}$
  - The local minimum is based on $\nabla E = 0$ at a minimum of the error.
  - τ is the time-step or iteration step.
  - $\Delta\mathrm{w}^{(\tau)}$ is the weight update.

Basic Concepts — Feed-forward Neural Networks

- To optimize the parameters, an approximation approach, namely local quadratic approximation is applied here:
  - $E(\mathrm{w}) \approx E(\hat{\mathrm{w}}) + (\mathrm{w}-\hat{\mathrm{w}})^T\mathrm{b} + \frac{1}{2}(\mathrm{w}-\hat{\mathrm{w}})^T\mathrm{H}(\mathrm{w}-\hat{\mathrm{w}})$
  - $\mathrm{b} = \nabla E|_{w=\hat{w}}$ is the gradient at $\hat{w}$.
  - $(\mathrm{H})_{ij} = \frac{\partial E}{\partial w_i \partial w_j}|_{w=\hat{w}}$ is the Hessian $\nabla\nabla E$ at $\hat{w}$
  - if $\mathrm{w} \approx \hat{\mathrm{w}}$ then $\nabla E \approx \mathrm{b} + \mathrm{H}(\mathrm{w}-\hat{\mathrm{w}})$.
  - Let w^* is at the minimum of E. so $\mathrm{b} = \nabla E|_{w=w^*} = 0$. then
    - $E(\mathrm{w}) = E(\mathrm{w}^*) + \frac{1}{2}(\mathrm{w}-\mathrm{w}^*)^T\mathrm{H}(\mathrm{w}-\mathrm{w}^*)$

Basic Concepts — Feed-forward Neural Networks

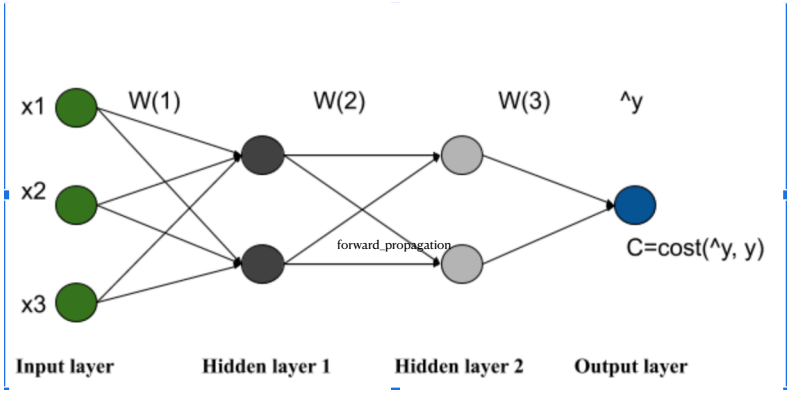
- Let w^* is at the minimum of E. so $\mathrm{b} = \nabla E|_{w=w^*} = 0$. then
  - $E(\mathrm{w}) = E(\mathrm{w}^*) + \frac{1}{2}(\mathrm{w}-\mathrm{w}^*)^T\mathrm{H}(\mathrm{w}-\mathrm{w}^*)$
  - where $\mathrm{H} = \nabla\nabla E|_{w=w^*}$ is the Hessian.
  - the eigenvectors $\mathrm{Hu}_i = \lambda \mathrm{u}_i$ are orthonormal.
  - $(\mathrm{w}-\mathrm{w}^*) = \sum_i \alpha_i \mathrm{u}_i$
  - Here we have: $\frac{1}{2}(\mathrm{w}-\mathrm{w}^*)^T\mathrm{H}(\mathrm{w}-\mathrm{w}^*) = \frac{1}{2}(\sum_i \lambda_i\alpha_i\mathrm{u}_i)^T(\sum_j \alpha_j\mathrm{u}_j)$
  - $E(\mathrm{w}) = E(\mathrm{w}^*) + \frac{1}{2}\sum_i \lambda_i\alpha_i^2$ because $\mathrm{u}_i^T\mathrm{u}_j = \mathrm{I}$, where I is identity matrix.

Basic Concepts — Feed-forward Neural Networks

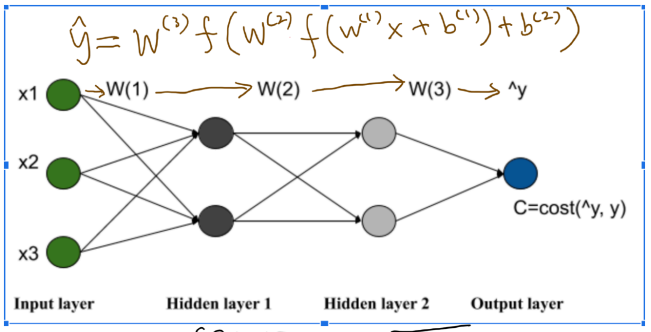
- Gradient Descent (GD)
  - The simplest approach is to update w by a displacement in the negative gradient direction.
    - $\mathrm{w}^{(\tau+1)} = \mathrm{w}^{(\tau)} - \eta\nabla E(\mathrm{w}^{(\tau)})$
  - This is a steepest descent algorithm.
  - η is the learning rate.
  - This is a batch method, as evaluation of ∇E involves the entire data set.
  - Conjugate gradient or quasi-Newton methods may, in practice, be preferred.
  - A range of starting points $\{w^{(0)}\}$ may be needed, in order to find a satisfactory minimum.

Basic Concepts — Feed-forward Neural Networks

- Optimization scheme:
  - Each iteration of the descent algorithm has two stages:
    1. Evaluate derivatives of error with respect to weights (involving backpropagation of error though the network).
    2. Use derivatives to compute adjustments of the weights (e.g. steepest descent).This is a batch method, as evaluation of ∇E involves the entire data set.
  - Backpropagation is a general principle, which can be applied to many types of network and error function.



Forward propagation



Basic Concepts —Backpropagation Neural Networks

- Backpropagation "repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the neural network and the desired output vector."[3]
- Chain rule: $\frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{jk}^l}$
- m - number of neurons in l-1 layer: $z_j^l = \sum_{k=1}^{m} w_{jk}^l a_k^{l-1} + b_j^1$
- by differentiation (calculating derivative): $\frac{\partial z_j^l}{\partial w_{jk}^l} = a_k^{l-1}$
- the final value, $\frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l} a_k^{l-1}$

- Reference: Hinton, G. & Williams, R. Learning representations by back-propagating errors. Nature 323, 533–536 (1986). https://doi.org/10.1038/323533a0

Recap

- The models for regression and classification consist of linear combinations of fixed basis functions.
- SVM comes out a solution as a kernel-based algorithm that has sparse solutions—which means the kernel function evaluated at a subset of the training data points.
- There is a need for an alternative approach to overcome these disadvantages. Here is the Neural Network.
- The intuition comes out the feed-forward neural network, also known as the multilayer perceptron.
- The term 'neural network' came from biological systems. Machine learning focus on neural networks as efficient models for statistical pattern recognition.

Recap

- Feed-forward Neural Networks generalize the linear model.
- construct M linear combinations of the input variables $x_1, \ldots, x_D$ in the form:
  $a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}$
- Each linear combination a_j is transformed by a (nonlinear differentiable) activation function: $z_j = h(a_j)$
- The output activations of the hidden layer $z_j = h(a_j)$ are linearly combined in layer two.
- The complete two layer model: $y_k(\mathrm{x}, \mathrm{w}) = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i \right) \right)$

Recap

- Evaluation of the feed-forward network is called forward propagation.
- Gradient Descent (GD) is the simplest approach is to update w by a displacement in the negative gradient direction.
- To optimize the parameters, evaluate derivatives of error with respect to weights (involving backpropagation of error though the network).
- Backpropagation is the key algorithm in neural network. It uses chain rules to compute gradients of cost function over weights and biases.
- There are three main processes in neural network: Forward propagation, cost function, and Backpropagation.
- The principals of the learning in neural network is about summation of information, non-linearly transformer the summation, re-allocation of weights of all neurones by adjusting the weights from the differentiate errors over weight parameters.

Questions?

Lab5