67750: Advanced Practical Machine Learning 2018/9

Quiz - Solution

TA: Daniel Gissin

1 The EM Algorithm (33 Points)

1.1 MLE of the Exponential Distribution

In class we calculated the MLE of a sample drawn from a Bernoulli distribution or a Gaussian distribution.

You are now the manager of the Rothberg cafeteria, and you collect N samples of the time it took from the moment person i asked for a salad until he got his salad. You assume, as one does, that the service time is distributed exponentially ($p(x) = \lambda e^{-\lambda x}$ for $x \geq 0$). Calculate the optimal parameter λ using MLE and show that:

$$\hat{\lambda}_{MLE} = \frac{1}{\bar{x}}$$

Solution

We write the likelihood function of our sample:

$$\mathcal{L}(S; \lambda) = \prod_{i=1}^{N} \lambda e^{-\lambda x_i} = \lambda^N e^{-\lambda \sum_{i=1}^{N} x_i}$$

Now, we take the log of the likelihood function:

$$\ell(S, \lambda) = N log(\lambda) - \lambda \sum_{i=1}^{N} x_i$$

Finally, we take the derivative w.r.t. our parameter λ and set it equal to zero:

$$\frac{\partial \ell}{\partial \lambda} = \frac{N}{\lambda} - \sum_{i=1}^{N} x_i = 0 \rightarrow \hat{\lambda}_{MLE} = \frac{N}{\sum_{i=1}^{N} x_i} = \frac{1}{\bar{x}}$$

1.2 Mixture of Exponentials

There are k employees selling salads in the cafeteria, but your data doesn't contain who made each salad. Still, you want to try and estimate the service time distribution of each of your employees.

1. Write the probability distribution function of the "mixture of exponentials" model for a single service time, which assumes that first an employee is picked according to multinomial weights π and then the service time is sampled from that employee's exponential distribution.

2. What is the log likelihood function of the entire sample S under the mixture of exponentials model?

3. Formulate the EM update for this model.

Solution

1. $\mathbb{P}(x) = \sum_{y=1}^{k} \mathbb{P}(y)\mathbb{P}(x|y) = \sum_{y=1}^{k} \pi_y \lambda_y e^{-\lambda_y x}$

2. Our log likelihood function is:

$$\ell(S, \lambda) = \sum_{i=1}^{N} log(\sum_{y=1}^{k} \pi_y \lambda_y e^{-\lambda_y x})$$

3. We can calculate the probability of every service time coming from every one of the employees ($c_{i,y}$) by using Bayes' rule:

$$c_{i,y} = \mathbb{P}(y|x) = \frac{\mathbb{P}(x|y)\mathbb{P}(y)}{\mathbb{P}(x)} = \frac{\pi_y \lambda_y e^{-\lambda_y x}}{\sum_{l=1}^{k} \pi_l \lambda_l e^{-\lambda_l x}}$$

Next, we can write our expected log likelihood function, using the calculated probabilities:

$$\mathbb{E}[\ell(S, \lambda'); \lambda] = \sum_{i=1}^{N} \sum_{y=1}^{k} c_{i,y}\big(log(\pi_y) + log(\lambda_y e^{-\lambda_y x})\big) =$$

Optimizing for the parameters, we finally get (using a Lagrange multiplier for the π_y optimization):

$$\pi_y = \frac{1}{N} \sum_{i=1}^{n} c_{i,y}$$

$$\lambda_y = \frac{\sum_{i=1}^{N} c_{i,y}}{\sum_{i=1}^{N} c_{i,y} x_i}$$

2 Maximum Entropy Markov Models (33 Points)

2.1 Definition & Inference

For sequential data $(x_{1:T}, y_{1:T})$, we assume that the y variables are Markovian and hidden while the x variables are observed (as in the PoS tagging application from Ex2).

1. Define $\mathbb{P}(y_t|y_{t-1}, x_{1:T})$ according to the MEMM defined by the feature function ϕ and the weight vector w. Explicitly write the partition function which normalizes the distribution (Z).

2. Inference for the MEMM is done using Viterbi. Write the definition of $\pi_t(i)$, the value which we calculate in the Viterbi dynamic programming algorithm.

3. We can efficiently calculate $\pi_t(i)$ with a recursive formula, by using the values of π_{t-1}. Write the recursive formula for calculating $\pi_t(i)$, given π_{t-1} and the model parameters.

Solution

1. $\mathbb{P}(y_t|y_{t-1}, x_{1:T}) = \frac{e^{w^T\phi(x_{1:T},y_{t-1},y_t,t)}}{\sum_{y'} e^{w^T\phi(x_{1:T},y_{t-1},y',t)}}$
2. $\pi_t(i) = \underset{\{y|y_t=i\}}{max}(\mathbb{P}(y_{1:t}|x_{1:T}))$
3. $\pi_t(i) = \underset{j}{max}\big(\pi_{t-1}(j)\frac{e^{w^T\phi(x_{1:T},y_j,y_i,t)}}{Z(x_{1:T},y_j,t)}\big)$

2.2 Learning

1. write the log probability of a single transition according to the MEMM ($log(\mathbb{P}(y_t|y_{t-1}, x_{1:T}))$).
2. Given a transition from your dataset $(y_{t-1}, y_t, x_{1:T})$, calculate the gradient of the log probability of the transition with respect to w. Show that it is the difference between the feature vector according to the real transition and the expected feature vector according to the model distribution (the expectation is over y_t).

Solution

1. $log(\mathbb{P}(y_t|y_{t-1}, x_{1:T})) = w^T\phi(x_{1:T}, y_{t-1}, y_t, t) - log(\sum_{y'} e^{w^T\phi(x_{1:T},y_{t-1},y',t)})$
2. Taking the derivative of the log probabilities gives us:

   $$\frac{\partial log(\mathbb{P}(y_t|y_{t-1}, x_{1:T}))}{\partial w} = \phi(x_{1:T}, y_{t-1}, y_t, t) - \frac{1}{\sum_{y'} e^{w^T\phi(x_{1:T},y_{t-1},y',t)}}\big(\sum_{y'}\phi(x_{1:T}, y_{t-1}, y', t)e^{w^T\phi(x_{1:T},y_{t-1},y',t)}\big)$$

   Rearranging the right hand term gives us:

   $$= \phi(x_{1:T}, y_{t-1}, y_t, t) - \sum_{y'}\frac{e^{w^T\phi(x_{1:T},y',y_t,t)}}{\sum_{y'} e^{w^T\phi(x_{1:T},y_{t-1},y',t)}}\phi(x_{1:T}, y', y_t, t)$$

   $$= \phi(x_{1:T}, y_{t-1}, y_t, t) - \mathbb{E}_{y'\sim MEMM}[\phi(x_{1:T}, y', y_t, t)|x_{1:T}, y_{t-1}]$$

3 Convnets (33 Points)

3.1 Neural Network Expressiveness

Consider two network layer architectures - the first is a fully connected layer followed by ReLU activation with an output of 100 neurons. The second is a convolutional layer of one weight filter followed by ReLU activation such that the output is 100 neurons (the image has 10×10 pixels).

Is one architecture more expressive than the other?

If the answer is no, show how any weight configuration of one layer can be expressed by the other.

If the answer is yes, show how one type of layer can express any configuration of the other and show a counter example for how the other layer can't express a specific configuration of the first layer.

Solution

The fully connected layer is more expressive.

A convolution is a linear operation and so a single neuron created by a inner product of a weight filter with a patch of the input can be expressed by a fully connected weight vector (by simply having zeros in weights not corresponding to the relevant input patch). Since the amount of output neurons is the same, this means the convolutional layer can be expressed by the fully connected layer.

On the other hand, we can easily think of a fully connected weight configuration which cannot be expressed using a convolutional layer. For instance, we can create a fully connected layer where the 100 weight vectors are all the same and have zeros everywhere but for the first coordinate. Clearly, a convolutional filter which doesn't get to see the first coordinate for most of the convolution operations can't express this function.

3.2 Log Loss

A popular loss function for binary classification tasks is the log loss:

$$\ell(y,p) = -ylog(p) - (1-y)log(1-p)$$

Assume our networks final activation function is a sigmoid whose input comes from a fully connected layer:

$$p = \sigma(w^T x) = \frac{1}{1+e^{-w^T x}}$$

Calculate the derivative of the log loss with respect to w. Simplify your solution as much as you can. It may be helpful to use the chain rule. You do not have to derive expressions we saw in class.

Solution

We begin by writing out the chain rule:

$$\frac{\partial \ell}{\partial w} = \frac{\partial \ell}{\partial p}\frac{\partial \sigma(w^T x)}{\partial (w^T x)}\frac{\partial (w^T x)}{\partial w}$$

Going from right to left:

$$\frac{\partial (w^T x)}{\partial w} = x$$

$$\frac{\partial \sigma(w^T x)}{\partial (w^T x)} = \sigma(w^T x)(1-\sigma(w^T x)) = p(1-p)$$

$$\frac{\partial \ell}{\partial p} = -\frac{y}{p} + \frac{1-y}{1-p} = \frac{p-py-y+py}{p(1-p)} = \frac{p-y}{p(1-p)}$$

Putting it all together gives us the lovely expression:

$$\frac{\partial \ell}{\partial w} = (p-y)\cdot x$$

4 Clustering (33 Points)

4.1 k-means

1. Describe Lloyd's iterative algorithm for heuristically solving the k-means problem for the euclidean distance metric.
2. Prove that said algorithm converges in a finite number of steps.

Solution

1. Lloyd's algorithm is similar to EM. We begin by initializing k centroids and then iterate between the following two steps until convergence:
   Step 1 - we assign points to clusters according to the closest centroid:
   $$C(x_i) = \underset{C}{argmin}\big(||x_i - \mu_C||^2\big)$$
   Step 2 - we move the clusters so as to minimize the k-means cost function:
   $$\forall k,\, \mu_k = \underset{\mu}{argmin}\big(\sum_{x \in C_k} ||x - \mu||^2\big)$$
   Convergence is achieved when the clusters don't change from one iteration to the next.
2. The proof can be seen in the solution of the theoretical part of ex4.

4.2 Spectral Clustering

Spectral clustering performs k-means on an embedding of our original data. In the embedding, we move from the data matrix X to a distance matrix S and then to a non negative similarity matrix W. We then minimize the Laplacian of the graph whose adjacency matrix is described by W.

1. Define the graph Laplacian for a given adjacency matrix W.
2. Show that the vector of 1s is an eigenvector of L with an eigenvalue of 0.
3. Show that L is a PSD matrix (show that for any vector f, $f^T L f \geq 0$).

Solution

1. We first define the diagonal degree matrix D as being: $D_{i,i} = \sum_j W_{i,j}$
   Next, the Laplacian is simply $L = D - W$
2. $(L1)_i = (D1)_i - (W1)_i = D_{i,i} - \sum_j W_{i,j} = \sum_j W_{i,j} - \sum_j W_{i,j} = 0$
   This is true for every coordinate i, and so the resulting vector is the zero vector, which shows that 1 is an eigenvector of L with an eigenvalue of 0.

3. We will show that $f^T L f$ is a sum of quadratic functions multiplied by non negative scalars:

$$f^T L f = f^T D f - f^T W f = \sum_i D_{i,i} f_i^2 - \sum_{i,j} W_{i,j} f_i f_j = \sum_i \sum_j W_{i,j} f_i^2 - \sum_{i,j} W_{i,j} f_i f_j$$

This can be rewritten in the following way:

$$= \frac{1}{2}\Big(\sum_{i,j} W_{i,j} f_i^2 - 2\sum_{i,j} W_{i,j} f_i f_j + \sum_{i,j} W_{j,i} f_j^2 \Big) = \frac{1}{2} \sum_{i,j} W_{i,j} (f_i - f_j)^2 \geq 0$$

The expression is non negative because we designed W to contain values between 0 and 1 (which are all non negative values).