Third assignment

52041MAL6Y Machine Learning 1 23/24 (Period 1.1) · 3 exercises · 19.0 points

1 Principal component analysis

7.0 points · 11 questions

Suppose we have a dataset of N D-dimensional vectors $\{\mathbf{x}_n\}_{n=1}^N$, $\mathbf{x}_n \in \mathbb{R}^D$.
We can write the entire dataset as an $N \times D$ matrix $\mathbf{X}$ (row n is $\mathbf{x}_n^\top$). We may wish to perform PCA on this data in the original data space, or in kernel-space using kernel-PCA. In the latter case, the data are projected into feature-space, such that $\boldsymbol{\phi}_n = \boldsymbol{\phi}(\mathbf{x}_n)$ is an M-dimensional feature space representation of $\mathbf{x}_n$. Consider the procedure for PCA (which can be generalized to kernel-PCA):

- Step 1: Center $\mathbf{X}$ using its sample mean, producing a centered data matrix $\hat{\mathbf{X}}$.
- Step 2: Compute sample covariance $\mathbf{S}$ of the centered dataset.
- Step 3: Solve the eigenvalue problem as to find the decomposition $\mathbf{S} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$, where $\mathbf{U}$ is a square matrix whose k-th column is the eigenvector $\mathbf{u}_k$ of $\mathbf{S}$, and $\boldsymbol{\Lambda}$ is a diagonal matrix whose diagonal elements are the corresponding eigenvalues λ_k, *i.e.* $\Lambda_{kk} = \lambda_k$ and zero otherwise. The covariance matrix thus decomposes into a set of orthonormal (linearly independent) basis vectors and their corresponding eigenvalues. This allows us to order our new basis vectors into independent projections that preserve the largest variance of the data.
- Step 4: Pick the K eigenvectors with largest eigenvalues $\{\mathbf{u}_1, \ldots, \mathbf{u}_K\}$. These will correspond to the projections that preserve the largest variance in the data. Since these projections are now linearly independent, the eigenvectors with smaller eigenvalues will only account for smaller variations in the data, and hence we can discard them without the original data being altered too much.
- Step 5: Project data onto the new K-dimensional basis.

Answer the following questions:

Text

a Provide an expression for $\hat{\mathbf{x}}_n$ (with $\hat{\mathbf{x}}_n$ being a mean-centered vector from $\mathbf{X}$).

0.25 points · Open

+0.5 points

The sample mean is $\overline{\mathbf{x}} = \frac{1}{N}\sum_{m=1}^{N}\mathbf{x}_m$.
The expression for $\hat{\mathbf{x}}_n$ is

$$\begin{aligned}\hat{\mathbf{x}}_n &= \mathbf{x}_n - \overline{\mathbf{x}} \\ &= \mathbf{x}_n - \frac{1}{N}\sum_{m=1}^{N}\mathbf{x}_m\end{aligned}$$

-0.3 points

The expression $\hat{\mathbf{x}}_n = \mathbf{x}_n - \overline{\mathbf{x}}$
in and of itself does not provide a complete expression for
$\hat{\mathbf{x}}_n$. $\overline{\mathbf{x}}$ is not defined anywhere;
what is the value of $\overline{\mathbf{x}}$?

-0.25 points

The sample mean vector $\overline{\mathbf{x}}$ is a D-dimensional
vector whose d-th entry $\overline{\mathbf{x}}_d$ are defined as
$\overline{\mathbf{x}}_d = \frac{1}{N}\sum_{m=1}^{N}\mathbf{x}_{md}$. That is, the
average is taken oven all N data points, independently for each
dimension, and not over D dimensions.

-0.1 points

The n-th row of $\mathbf{X}$ is $\mathbf{x}_n^\top$,
not $\mathbf{x}_n$. This is because $\mathbf{x}_n$ is a column
vector, but $\mathbf{X}$ is organized as an $N \times D$ matrix.

0 points

Be careful when using the same indices for data points and for sums, like: $\hat{\mathbf{x}}_n = \mathbf{x}_n - \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$. This is correct assuming that the scope of the sum index only exists within the sum itself, but can be a source of trouble, especially when taking partial derivatives.

-0.1 points

$\mathbf{x}_n$ is the n-th data-point, but it is not the n-th column of $\mathbf{X}$. Rather, $\mathbf{x}_n^\top$ is the n-th row of $\mathbf{X}$.

-0.1 points

Given the sample mean $\overline{\mathbf{x}}$, you cannot just subtract it from the data matrix $\mathbf{X}$. The shapes do not match. Instead, if you wanted to use a matrix notation, you could use $\mathbf{1}_N$ that indicates a vector of one's of size N and write it as $\hat{\mathbf{X}} = \mathbf{X} - \mathbf{1}_N \cdot \overline{\mathbf{x}}^\top$.

-0.2 points

$\hat{\mathbf{x}}_n$ is centered, there is no scaling involved.

b Prove that the average of $\hat{\mathbf{x}}_n$ (over N data vectors) is the $\mathbf{0}$ vector.

0.25 points · Open

+0.5 points

$$\begin{aligned}\sum_{n=1}^{N}\hat{\mathbf{x}}_n &= \sum_{n=1}^{N}(\mathbf{x}_n - \overline{\mathbf{x}}) \\ &= \sum_{n=1}^{N}\mathbf{x}_n - N\overline{\mathbf{x}} \\ &= \sum_{n=1}^{N}\mathbf{x}_n - N\frac{1}{N}\sum_{m=1}^{N}\mathbf{x}_m \\ &= \sum_{n=1}^{N}\mathbf{x}_n - \sum_{m=1}^{N}\mathbf{x}_m \\ &= \mathbf{0} \\ \implies \frac{1}{N}\sum_{n=1}^{N}\hat{\mathbf{x}}_n &= \mathbf{0}\end{aligned}$$

-0.1 points

Your derivations skip a few steps; they should be more verbose. In particular, why is $\frac{1}{N}\sum_{n=1}^{N}\overline{\mathbf{x}} = \overline{\mathbf{x}}$? Or, equivalently, why is $\frac{1}{N}\sum_{n=1}^{N}\frac{1}{N}\sum_{m=1}^{N}\mathbf{x}_m = \frac{1}{N}\sum_{m=1}^{N}\mathbf{x}_m$? This should not be taken for granted. An extra derivation step or a verbal explanation is needed.

-0.2 points

Your derivations suddenly jump to the final answer, $\sum_{n=1}^{N}\mathbf{x}_n - \sum_{n=1}^{N}\mathbf{x}_n = \mathbf{0}$, or (almost) equivalently, $\overline{\mathbf{x}} - \overline{\mathbf{x}} = \mathbf{0}$. How did you get there?

0 points

The fact that the average of $\hat{\mathbf{x}}_n$ over N is $\mathbf{0}$ (which is the answer to this question) does not mean that all $\hat{\mathbf{x}}_n$ are $\mathbf{0}$.

c Provide an expression for $\mathbf{S}$ in terms of $\hat{\mathbf{X}}$.

0.25 points · Open

+0.5 points

Denoting the average of $\hat{\mathbf{x}}_n$ over N as $\bar{\hat{\mathbf{x}}}$, the sample covariance is computed as follows:

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{\mathbf{x}}_n - \bar{\hat{\mathbf{x}}}\right)\left(\hat{\mathbf{x}}_n - \bar{\hat{\mathbf{x}}}\right)^\top$$

From the previous question, we have $\bar{\hat{\mathbf{x}}} = \mathbf{0}$. Thus, the sample covariance is computed as:

$$\begin{aligned}\mathbf{S} &= \frac{1}{N}\sum_{n=1}^{N}\hat{\mathbf{x}}_n\hat{\mathbf{x}}_n^\top \\ &= \frac{1}{N}\hat{\mathbf{X}}^\top\hat{\mathbf{X}}\end{aligned}$$

-0.1 points

Covariance is an average, not a sum.

-0.2 points

You have to show your derivations, and not just provide the final answer.

-0.1 points

Covariance is a $D \times D$ matrix, $\mathbf{S} = \frac{1}{N}\hat{\mathbf{X}}^\top\hat{\mathbf{X}}$.

-0.1 points

The question asks for an expression in terms of $\hat{\mathbf{X}}$.

-0.1 points

The definition of the sample covariance takes the average into account. You, however, have omitted it.

-0.1 points

The sample covariance $\mathbf{S}$ is the covariance of the centered dataset $\hat{\mathbf{X}}$. Thus, the definition of it is

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{\mathbf{x}}_n - \bar{\hat{\mathbf{x}}}\right)\left(\hat{\mathbf{x}}_n - \bar{\hat{\mathbf{x}}}\right)^\top,$$

not

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{x}_n - \bar{\mathbf{x}}\right)\left(\mathbf{x}_n - \bar{\mathbf{x}}\right)^\top.$$

d What is the dimensionality of $\mathbf{S}$?

0.25 points · Open

+0.5 points

$\mathbf{S}$ is a $D \times D$ matrix.

-0.2 points

$\hat{\mathbf{X}}$ is an $N \times D$ matrix, not $D \times N$.

e What is the expression for the linear projection $\mathbf{L}$ that maps data vectors $\hat{\mathbf{x}}$ onto a K-dimensional sub-space, $\mathbf{y}_n = \mathbf{L}\hat{\mathbf{x}}_n,$ such that it has zero mean and identity covariance.

- Write down the expression for $\mathbf{L}$. [.25 point]
- Prove that the average over N of y_n is 0. [.25 point]
- Prove that the covariance of y_n is the identity. [.25 point]
- What is the operator $\mathbf{L}$ called? [.25 point]

1.0 point · Open

+0.5 points

The full projection is

$$\mathbf{y}_n = \mathbf{L}\hat{\mathbf{x}}_n = \mathbf{\Lambda}_K^{-\frac{1}{2}}\mathbf{U}_K^\top\hat{\mathbf{x}}_n,$$

where $\mathbf{L} = \mathbf{\Lambda}_K^{-\frac{1}{2}}\mathbf{U}_K^\top$.

+1 point

The average $\mathbf{y}_n$ is

$$\begin{aligned}\sum_{n=1}^{N}\mathbf{y}_n &= \sum_{n=1}^{N}\mathbf{L}\hat{\mathbf{x}}_n \\ &= \mathbf{L}\sum_{n=1}^{N}\hat{\mathbf{x}}_n \\ &= \mathbf{L}\cdot\mathbf{0} \\ &= \mathbf{0} \\ \implies \overline{\mathbf{y}} &= \frac{1}{N}\sum_{n=1}^{N}\mathbf{y}_n \\ &= \mathbf{0}\end{aligned}$$

+1 point

Examine the covariance between the i-th and j-th projection.

We use y_{ni} to denote the i-th dimension of the n-th data point $\mathbf{y}_n$.

$$
\begin{aligned}
y_{ni} &= \frac{\mathbf{u}_i^\top}{\lambda_i^{\frac{1}{2}}} \hat{\mathbf{x}}_n \\
C_{ij} &= \frac{1}{N} \sum_{n=1}^{N} y_{ni}\, y_{nj} \\
&= \frac{1}{N} \sum_{n} \frac{\mathbf{u}_i^\top}{\lambda_i^{\frac{1}{2}}} \hat{\mathbf{x}}_n \hat{\mathbf{x}}_n^\top \frac{\mathbf{u}_j}{\lambda_j^{\frac{1}{2}}} \\
&= \frac{\mathbf{u}_i^\top}{\lambda_i^{\frac{1}{2}}} \left(\frac{1}{N} \sum_{n} \hat{\mathbf{x}}_n \hat{\mathbf{x}}_n^\top \right) \frac{\mathbf{u}_j}{\lambda_j^{\frac{1}{2}}} \\
&= \frac{\mathbf{u}_i^\top}{\lambda_i^{\frac{1}{2}}} \mathbf{S} \frac{\mathbf{u}_j}{\lambda_j^{\frac{1}{2}}} \\
&= \frac{\mathbf{u}_i^\top}{\lambda_i^{\frac{1}{2}}} \mathbf{U} \mathbf{\Lambda} \mathbf{U}^\top \frac{\mathbf{u}_j}{\lambda_j^{\frac{1}{2}}}
\end{aligned}
$$

Since $\mathbf{U}$ is orthonormal, we have $\mathbf{u}_i^\top \mathbf{U} = \mathbf{1}_i^\top$, where $\mathbf{1}_i$ is a one-hot column vector with zeroes everywhere except a value of 1 at index i.

Hence,

$$
C_{ij} = \frac{\mathbf{1}_i^\top}{\lambda_i^{\frac{1}{2}}} \mathbf{\Lambda} \frac{\mathbf{1}_j}{\lambda_j^{\frac{1}{2}}}
$$

Due to the dot product between the 2 one-hot encodings, the only non-zero entry in the covariance matrix occurs when $i = j$. In that case, we have:

$$
C_{ij} = \frac{\lambda_i}{\lambda_i^{\frac{1}{2}} \lambda_i^{\frac{1}{2}}} = 1
$$

In total, we have:

$$
C_{ij} = \delta_{ij} \implies \mathbf{C} = \mathbf{I}
$$

+0.5 points

This operation is called whitening or sphering.

-0.2 points

The projection matrix is $\mathbf{L} = \mathbf{\Lambda}_K^{-\frac{1}{2}} \mathbf{U}_K^\top$, not $\mathbf{L} = \mathbf{\Lambda}_K^{-\frac{1}{2}} \mathbf{U}_K$.

f Given a projection $\mathbf{L}$ that maps data vectors onto a K-dimensional subspace:

- How does varying K affect the resulting variance, information loss, and computation? [.75 point]
- What trade-offs need to be considered when selecting K compared to D? [.25 point]

1.0 point · Bonus · Open

+0.25 points

Higher K retains more variance in the projection (Or inverse)

+0.25 points

Higher K retains more information in projection (Or inverse)

+0.25 points

Higher K implies a larger computation time (must mention that fewer eigenvalues have to be calculated)

+0.25 points

Explain that small K values reduce dimensionality and computation but lead to information loss, while larger K values retain more information, increase dimensionality and computational complexity.

g PCA provides a linear projection from a large M-dimensional feature space to a lower dimensional feature space D.

- Can you propose a non-linear adaption of PCA which can learn non-linear projections (without using pre-defined kernels)? What would this method look like? [.5 point]
- Give one pro and one con of this approach in comparison with PCA. [.5 point]

1.0 point · Open

+0.5 points

Using this architecture you can learn extremely complex functions instead of using pre-defined basis functions.
However, you cannot guarantee independence between the features of the latent space of the AE. Furthermore, the optimization routine might end up in local minima.

+0.5 points

We can make use of a non-linear autoencoder. This network comprises 2 sub-networks, an encoder and a decoder. The encoder is a neural network that maps the input to a lower-dimensional representation. The decoder is also a neural network takes this representation as input and produces an output of the same size as the input. The output is a reconstructed version of the input. This network should minimize the reconstruction error between target values (represented by the input values) and the output. An example of this network can be found on page 503 in the Bishop book (Figure 12.19).
Note that if we use a linear layer in both the encoder and the decoder, we have a linear autoencoder that is equivalent to PCA. Perhaps surprisingly, this also true even if we add a non-linearity at the hidden unit. A non-linear autoencoder's encoder and decoder both consist of (at least) a 2-layer MLP with a non-linearity in between. Such an architecture is strictly more powerful than PCA.

-0.2 points

For a non-linear autoencoder, both the encoder and the decoder should have at least 2 layers (and a non-linearity in between). An autoencoder with a linear encoder and a linear decoder is a linear model and is equivalent to PCA. The same applies even if we just add a non-linearity at the hidden unit.

-0.2 points

Even though the projection itself can be done using just the encoder, you would need a decoder too in order to train the model.

h Which two steps (1→5) are the slowest in PCA and what is the computational complexity of each of these? And for the whole algorithm? Here, suppose K=D. What is a possible problem you observe when inspecting this complexity and how could this problem be overcome?

1.0 point · Open

+0.25 points

Mentions the slowest steps are Step 2 (Calculation of Covariance matrix)and Step 3 (Eigen-decomposition.)

+0.25 points

Covariance Matrix Calculation: O(p^2 n)

Eigen-Value Decomposition: O(p^3)

+0.25 points

Total Complexity: (O(p^2 n | p^3))

+1 point

Problem: Complexity is too high to process all of the data in one go.

Solution: Batch Variant of PCA.

i PCA is an unsupervised method that does not take into account the given class information from your datasets. Could you name a dimensionality reduction method that can take into account the supervised information?

Think about approaches covered during the classification lectures.

1.0 point · Open

+0.5 points

Mentions correct technique: e.g LDA

+0.5 points

Gives correct explanation

j Given the annotated dataset (see the Figure in the pdf), which projection do you want to calculate to better separate the data? How would you calculate this projection?

In this figure (check the pdf), the data points with blue denote the first class, while red denotes the second class. The entities $\mathbf{m}_1$ and $\mathbf{m}_2$ denote the mean value for the initial space, while $\mu_1^{'}$ and $\mu_2^{'}$ are the means in the projected spaces. Similarly, you can find the variance for each class in the figure $s_1^{'}$ and $s_2^{'}$ in the projected space.

1.0 point · Open

k Can you formulate this projection in terms of $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{S}_1$ and $\mathbf{S}_2$?

Note: you do not need to perform any decomposition!

1.0 point · Open

2 Probabilistic PCA - A general latent space distribution

5.0 points · 4 questions

Principal Component Analysis (PCA) is an often used technique for dimensionality reduction. In this approach, we linearly project data onto the subspace of lower dimensionality. However, this approach can be extended to be more general by formulating a latent variable model named probabilistic PCA. In this approach, the PCA can be expressed as the maximum likelihood solution probabilistic PCA. There are multiple advantages of the probabilistic PCA over the conventional PCA. To name a few, we can associate a likelihood function to the probabilistic PCA which allows a direct comparison with other probabilistic density models, probabilistic PCA can be used to model class-conditional densities and can thus be used in classification problems, and also we can run the model generatively to provide samples from the modeled distribution.

Probabilistic PCA is an example of the linear-Gaussian framework, where both marginal and conditional distributions are Gaussian. We first define a latent variable $\mathbf{z}$, corresponding to the principal-component subspace. We can then define a prior distribution $p(\mathbf{z})$ over the latent variable $\mathbf{z}$, and also the conditional distribution $p(\mathbf{x}|\mathbf{z})$, which is given by

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I})\,.$$

The prior distribution over $\mathbf{z}$ is usually given by a zero-mean unit-covariance Gaussian:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})\,.$$

In this case, the marginal distribution $p(\mathbf{x})$ is also a Gaussian and is given by

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{W}\mathbf{W}^{\mathrm{T}} + \sigma^2\mathbf{I})\,.$$

However, suppose we replace the zero-mean and the unit-covariance latent space distribution with a general Gaussian distribution of the form

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{m}, \boldsymbol{\Sigma}).$$

In this problem, we wish to show that by redefining parameters of the model, this assumption on the prior leads to an identical model for the marginal distribution $p(\mathbf{x})$ over the observed variables for any valid choice of $\mathbf{m}$ and $\boldsymbol{\Sigma}$. We will derive this result in multiple steps.

Text

a We can express the random variable $\mathbf{z}$ as $\mathbf{z} = \mathbf{m} + \boldsymbol{\epsilon}_z$, with noise $\boldsymbol{\epsilon}_z \sim \mathcal{N}(0, \boldsymbol{\Sigma})$. Write out a similar expression for the variable $\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}_x$, and explain what distribution variable $\mathbf{x}$ follows. From which distribution is $\boldsymbol{\epsilon}_x$ sampled from?

1.0 point · Open

+0.5 points

Correct expression.

$$\mathbf{x} = \mathbf{W}(\mathbf{m} + \epsilon_z) + \mu + \epsilon_x$$

or

$$x = \mathbf{W}\mathbf{m} + \mu + \mathbf{W}\epsilon_z + \epsilon_x$$

+0.25 points

$\mathbf{x}$ follows a normal distribution.

+0.25 points

ϵ_x follows a normal distribution.

b Find the expectation value of the variable $\mathbf{x}$ using the linearity property of the expected value

1.0 point · Open

+0.75 points

Using linearity of expectation:

$$\begin{aligned}\mathrm{E}[\mathbf{x}] &= \mathrm{E}\left[\mathbf{W}\mathbf{m} + \boldsymbol{\mu} + \mathbf{W}\boldsymbol{\epsilon}_z + \boldsymbol{\epsilon}_x\right] \\ &= \mathrm{E}\left[\mathbf{W}\mathbf{m}\right] + \mathrm{E}\left[\boldsymbol{\mu}\right] + \mathrm{E}\left[\mathbf{W}\boldsymbol{\epsilon}_z\right] + \mathrm{E}\left[\boldsymbol{\epsilon}_x\right] \\ &= \mathbf{W}\mathrm{E}\left[\mathbf{m}\right] + \mathrm{E}\left[\boldsymbol{\mu}\right] + \mathbf{W}\mathrm{E}\left[\boldsymbol{\epsilon}_z\right] + \mathrm{E}\left[\boldsymbol{\epsilon}_x\right]\end{aligned}$$

+0.25 points

Filling in the known expectations:

$$E[\mathbf{x}] = \mathbf{W}\mathbf{m} + \boldsymbol{\mu}$$

c Find the covariance of the variable $\mathbf{x}$ using the definition of the covariance $\mathrm{cov}[\mathbf{x}, \mathbf{x}]$.

Hint: The following identity might be helpful: $\mathrm{Var}\left[\mathbf{AY}\right] = \mathbf{A}\mathrm{Var}\left[\mathbf{Y}\right]\mathbf{A}^{\mathrm{T}}$, where $\mathbf{A}$ is a matrix, and $\mathbf{Y}$ is a random variable.

2.0 points · Open

+1 point

Simplifying the identity:

$$\begin{aligned}\mathrm{Cov}[\mathbf{x}, \mathbf{x}] &= \mathrm{Cov}[\mathbf{Wm} + \boldsymbol{\mu} + \mathbf{W}\boldsymbol{\epsilon}_z + \boldsymbol{\epsilon}_x, \mathbf{Wm} + \boldsymbol{\mu} + \mathbf{W}\boldsymbol{\epsilon}_z + \boldsymbol{\epsilon}_x] \\ &= \mathrm{Cov}\left[\mathbf{W}\boldsymbol{\epsilon}_z, \mathbf{W}\boldsymbol{\epsilon}_z\right] + \mathrm{Cov}\left[\boldsymbol{\epsilon}_x, \boldsymbol{\epsilon}_x\right] \\ &= \mathrm{Var}[\mathbf{W}\boldsymbol{\epsilon}_z] + \mathrm{Var}[\boldsymbol{\epsilon}_x]\end{aligned}$$

Here we have used that the covariance of independent variables is zero.

+1 point

Using hint and filling in known quantities:

$$\begin{aligned}\mathrm{Cov}[\mathbf{x}, \mathbf{x}] &= \mathbf{W}\mathrm{Var}[\boldsymbol{\epsilon}_z]\mathbf{W}^{\mathrm{T}} + \mathrm{Var}[\boldsymbol{\epsilon}_x] \\ &= \mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^{\mathrm{T}} + \sigma^2\mathbf{I}\end{aligned}$$

d To show that using the general Gaussian prior still leads to an identical model $p(\mathbf{x})$, we have to be able to write the distribution $p(\mathbf{x})$ in the form $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\tilde{\boldsymbol{\mu}}, \tilde{\mathbf{W}}\tilde{\mathbf{W}}^{\mathrm{T}} + \sigma^2\mathbf{I})$. Find the appropriate expressions for $\tilde{\boldsymbol{\mu}}$ and $\tilde{\mathbf{W}}$.

1.0 point · Open

+0.5 points

$\tilde{\mu} = \mathbf{W}\mu + \mathbf{m}$

+0.5 points

$\tilde{\mathbf{W}} = \mathbf{W}\boldsymbol{\Sigma}^{1/2}$

3 Mixtures of experts

7.0 points · 7 questions

In class, you discussed and were introduced to mixture models as a way to perform unsupervised learning tasks, e.g. clustering. Mixture models can be similarly used for supervised learning tasks. In this question, we will discuss and explore the Mixtures of Experts (MoEs), a model that softly partitions the input space and learns a supervised model for each area.

Consider that you have K expert models available in order to model a specific dataset of N data points $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where $\mathbf{x}_n \in \mathbb{R}^D$ and y_n corresponds to the ground truth label for $\mathbf{x}_n$. Let $\mathbf{z}_n \in \mathbb{R}^K$ correspond to a one-hot encoding of a categorical random variable for data point $\mathbf{x}_n$ that denotes which of the K expert models is 'active'. If expert model k is active for datapoint $\mathbf{x}_n$, it means that model k provides the prediction for datapoint $\mathbf{x}_n$. The strength of an MoE approach is that it trains expert models that each specialize in a specific portion of the data.

Then, let $\mathbf{\Theta}$ be a matrix in $\mathbb{R}^{D\times K}$ that contains the D-dimensional column vector of parameters foreach expert. We will assume that each y_i is a continuous random variable at the $[0, \infty)$ interval and is distributed according to an exponential distribution with a rate $\lambda > 0$. Given the aforementioned assumptions, each expert $k \in K$ has the following linear predictive model:

$$p\left(y_n|\mathbf{x}_n, \mathbf{z}_n, \mathbf{\Theta}\right) = p\left(y_n|\mathbf{x}_n, \boldsymbol{\theta}_k = \mathbf{\Theta}\mathbf{z}_n\right) = \text{Exponential}\left(y_n|\lambda = \exp\left(\boldsymbol{\theta}_k^T\mathbf{x}_n\right)\right)$$

where $\mathbf{z}_n$ is the one-hot-encoded vector representation of the categorical variable $\mathbf{z}_n$ and

$$\text{Exponential}(y|\lambda) = \lambda\exp(-\lambda y) \text{ for } y \geq 0.$$

The flexibility of MoEs stems from the fact that there is a ``routing''
mechanism which determines how relevant each of the K experts is for a specific datapoint $\mathbf{x}_n$. As in this case, we have a discrete set of Kexperts, a simple linear routing mechanism is the following:

$$p(\mathbf{z}_n = k|\mathbf{x}_n, \mathbf{\Phi}) = \pi_{nk} = \frac{\exp(\boldsymbol{\phi}_k^T\mathbf{x}_n)}{\sum_j \exp(\boldsymbol{\phi}_j^T\mathbf{x}_n)}$$

where $\mathbf{\Phi}$ is a matrix in $\mathbb{R}^{D\times K}$ that contains allof the parameters of the routing function, i.e. $\mathbf{\Phi} = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_K]$

Text

f Note that the output of the routing function falls between 0 and 1. Thus, we need to construct our vector $\mathbf{z}_n$ based on these values.

Write down the formula to decide each element z_{nk} of $\mathbf{z}_n$, assuming you want the most relevant expert to be active for datapoint $\mathbf{x}_n$. Hint: remember that $\mathbf{z}_n$ is one-hot encoded.

0.5 points · Open

+0.5 points

$z_{nk'} = 1$ if $k' = \arg\max_k \pi_{nk}$, 0 otherwise

As a-priori we have no information about which of the experts is responsible for generating a particular prediction, we have to marginalize over all possible experts in order to compute the likelihood of an observed point. With this information answer the following questions:

Text

a Write down the likelihood of the entire dataset, $p(\mathbf{y}|\mathbf{X}, \mathbf{\Theta}, \mathbf{\Phi})$, and take its log under the i.i.d. assumption.

1.0 point · Open

+1 point

$$p(\mathbf{y}|\mathbf{X}, \mathbf{\Theta}, \mathbf{\Phi}) = \prod_{n=1}^{N} \sum_{k=1}^{K} p(z_n = k|\boldsymbol{x}_n, \mathbf{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = k, \boldsymbol{\theta}_k)$$

$$\log p(\mathbf{y}|\mathbf{X}, \mathbf{\Theta}, \mathbf{\Phi}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} p(z_n = k|\boldsymbol{x}_n, \mathbf{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = k, \boldsymbol{\theta}_k) \right).$$

b Write down the posterior probability r_{ni} of expert i producing the label y for datapoint n. We will also refer to this as the responsibility of expert i for datapoint n.

1.0 point · Open

+1 point

$$r_{ni} = \frac{p(z_n = i|\boldsymbol{x}_n, \mathbf{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = i, \boldsymbol{\theta}_i)}{\sum_{j=1}^{K} p(z_n = j|\boldsymbol{x}_n, \mathbf{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = j, \boldsymbol{\theta}_j)}$$

c Take the derivative of the log-likelihood w.r.t. the parameters of each expert $\boldsymbol{\theta}_i$ and the parameters of the routing mechanism for each expert $\boldsymbol{\phi}_i$. Do not substitute expressions for the probabilities but rather provide your answer in terms of $p(y_n|\mathbf{x}_n, z_n, \boldsymbol{\theta}_i)$, $p(z_n = k|\mathbf{x}_n, \boldsymbol{\Phi})$. Make sure to express the derivatives in terms of the responsibilities of each expert r_{ni}. (Hint: $\frac{\partial f(x)}{\partial x} = f(x)\frac{\partial \log f(x)}{\partial x}$), as that term will be present in the derivatives for both $\boldsymbol{\theta}_i, \boldsymbol{\phi}_i$.

1.5 points · Open

+1.5 points

Let $\mathcal{L} = \log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\Theta}, \boldsymbol{\Phi})$.

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta_i}} = \sum_{n=1}^{N} \frac{p(z_n = i|\boldsymbol{x}_n, \boldsymbol{\Phi})}{\sum_{j=1}^{K} p(z_n = j|\boldsymbol{x}_n, \boldsymbol{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = j, \boldsymbol{\theta}_j)} \frac{\partial p(y_n|\boldsymbol{x}_n, z_n = i, \boldsymbol{\theta}_i)}{\partial \boldsymbol{\theta}_i}$$

$$= \sum_{n=1}^{N} \frac{p(z_n = i|\boldsymbol{x}_n, \boldsymbol{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = i, \boldsymbol{\theta}_i)}{\sum_{j=1}^{K} p(z_n = j|\boldsymbol{x}_n, \boldsymbol{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = j, \boldsymbol{\theta}_j)} \frac{\partial \log p(y_n|\boldsymbol{x}_n, z_n = i, \boldsymbol{\theta}_i)}{\partial \boldsymbol{\theta}_i}$$

$$= \sum_{n=1}^{N} r_{ni} \frac{\partial \log p(y_n|\boldsymbol{x}_n, z_n = i, \boldsymbol{\theta}_i)}{\partial \boldsymbol{\theta}_i}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi_i}} = \sum_{n=1}^{N} \sum_{k=1}^{K} \frac{p(y_n|\boldsymbol{x}_n, z_n = k, \boldsymbol{\theta}_i)}{\sum_{j=1}^{K} p(z_n = j|\boldsymbol{x}_n, \boldsymbol{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = j, \boldsymbol{\theta}_j)} \frac{\partial p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\Phi})}{\partial \boldsymbol{\phi}_i}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \frac{p(y_n|\boldsymbol{x}_n, z_n = k, \boldsymbol{\theta}_i) p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\Phi})}{\sum_{j=1}^{K} p(z_n = j|\boldsymbol{x}_n, \boldsymbol{\Phi}) p(y_n|\boldsymbol{x}_n, z_n = j, \boldsymbol{\theta}_j)} \frac{\partial \log p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\Phi})}{\partial \boldsymbol{\phi}_i}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \frac{\partial \log p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\Phi})}{\partial \boldsymbol{\phi}_i}$$

d Replace the expressions for each of the respective probability distributions and compute the final derivatives for $\boldsymbol{\theta}_i, \boldsymbol{\phi}_i$.

1.5 points · Open

+1.5 points

(+0.75)

$$\log p(y_n|\boldsymbol{x}_n, z_n = i, \boldsymbol{\theta}_i) = \boldsymbol{\theta}_i^T \boldsymbol{x}_n - y_n \exp(\boldsymbol{\theta}_i^T \boldsymbol{x}_n)$$

$$\frac{\partial \log p(y_n|\boldsymbol{x}_n, z_n = i, \boldsymbol{\theta}_i)}{\partial \boldsymbol{\theta}_i} = \boldsymbol{x}_n - y_n \exp(\boldsymbol{\theta}_i^T \boldsymbol{x}_n)\boldsymbol{x}_n = (1 - y_n \exp(\boldsymbol{\theta}_i^T \boldsymbol{x}_n))\boldsymbol{x}_n^T$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}_i} = \sum_{n=1}^{N} r_{ni}(1 - y_n \exp(\boldsymbol{\theta}_i^T \boldsymbol{x}_n))\boldsymbol{x}_n^T.$$

(+0.75) And:

$$\log p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\Phi}) = \sum_{k=1}^{K} \mathbb{I}[z_n = k] \left(\boldsymbol{\phi}_k^T \boldsymbol{x}_n - \log \sum_j \exp(\boldsymbol{\phi}_j^T \boldsymbol{x}_n) \right)$$

$$\frac{\partial \log p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\Phi})}{\partial \boldsymbol{\phi}_i} = \left(\mathbb{I}[k = i] - \pi_{ni}\right) \boldsymbol{x}_n^T$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_i} = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left(\mathbb{I}[k = i] - \pi_{ni}\right) \boldsymbol{x}_n^T = \sum_{n=1}^{N} (r_{ni} - \pi_{ni}) \boldsymbol{x}_n^T$$

where $\pi_{ni} = \frac{\exp(\boldsymbol{\phi}_i^T \boldsymbol{x}_n)}{\sum_j \exp(\boldsymbol{\phi}_j^T \boldsymbol{x}_n)}$

e Write down an iterative algorithm that maximizes the log-probability of the data by jointly optimizing the $\mathbf{\Theta}$ and $\mathbf{\Phi}$ parameters. Make use of appropriate convergence criteria.

1.0 point · Open

+1 point

Example solution:

- Randomly initialize $\mathbf{\Theta}, \mathbf{\Phi}$. Choose a learning rate η and tolerance ϵ. Set $\mathcal{L}_0 = \inf$
- for i in range(1, max_iter):
  - $\boldsymbol{\theta}_k = \boldsymbol{\theta}_k + \eta(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}_k})^T$
  - $\boldsymbol{\phi}_k = \boldsymbol{\phi}_k + \eta(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_k})^T$
  - if $|\mathcal{L}_i - \mathcal{L}_{i-1}| \leq \epsilon$: break
- $\mathbf{\Theta}^*, \mathbf{\Phi}^*$

g For this question assume that instead of having $\mathbf{z}_n$ as a one-hot encoding, we have $z_{nk} = \pi_{nk}$. Thus, each element of $\mathbf{z}_n$ falls between 0 and 1. To compute our final prediction $\hat{y_n}$ for datapoint $\mathbf{x}_n$, we will now weigh the prediction of each expert by its relevancy. Write down the formula for determining the final prediction $\hat{y_n}$. Denote the prediction of expert k with $\hat{y_{nk}}$.

0.5 points · Open

+0.5 points

$\hat{y_n} = \frac{1}{K} \sum_{k=1}^{K} \pi_{nk} \hat{y_{nk}}$