7. Implementing EM for matrix completion

We need to update our EM algorithm a bit to deal with the fact that the observations are no longer complete vectors. We use Bayes' rule to find an updated expression for the posterior probability $p\left(j|u\right) = P\left(y=j|x^{(u)}_{C_u}\right)$:

$$p\left(j \mid u\right) = \frac{p\left(u|j\right)\cdot p\left(j\right)}{p\left(u\right)} = \frac{p\left(u|j\right)\cdot p\left(j\right)}{\sum_{j=1}^{K} p\left(u|j\right)\cdot p\left(j\right)} = \frac{\pi_j N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)}{\sum_{j=1}^{K}\pi_j N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)}$$

This is the soft assignment of cluster j to data point u.

To minimize numerical instability, you will be re-implementing the E-step in the log-domain, so you should calculate the values for the log of the posterior probability, $\ell\left(j,u\right) = \log\left(p\left(j|u\right)\right)$ (though the actual output of your E-step should include the non-log posterior).

Let $f\left(u,i\right) = \log\left(\pi_i\right) + \log\left(N\left(x^{(u)}_{C_u};\mu^{(i)}_{C_u},\sigma_i^2 I_{C_u\times C_u}\right)\right)$. Then, in terms of f, the log posterior is:

$$\begin{aligned}
\ell\left(j|u\right) \quad &= \log\left(p\left(j \mid u\right)\right) = \log\left(\frac{\pi_j N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)}{\sum_{j=1}^{K}\pi_j N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)}\right)\\
&= \log\left(\pi_j N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)\right) - \log\left(\sum_{j=1}^{K}\pi_j N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)\right)\\
&= \log\left(\pi_j\right) + \log\left(N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)\right) - \log\left(\sum_{j=1}^{K}\exp\left(\log\left(\pi_j N\left(x^{(u)}_{C_u};\mu^{(j)}_{C_u},\sigma_j^2 I_{C_u\times C_u}\right)\right)\right)\right)\\
&= f\left(u,j\right) - \log\left(\sum_{j=1}^{K}\exp\left(f\left(u,j\right)\right)\right)
\end{aligned}$$

Once we have evaluated $p\left(j|u\right)$ in the E-step, we can proceed to the M-step. We wish to find the parameters π, μ, and σ that maximize $\ell\left(X;\theta\right)$,

Generating Speech Output

First, note that, by decomposing the multivariate spherical Gaussians into univariate spherical Gaussians as before, we can write, if $l \in C_u$:

$$\begin{aligned}\frac{\partial}{\partial \mu_l^{(k)}} N\left(x_{C_u}^{(u)} | \mu_{C_u}^{(k)}, \sigma_k^2 I_{|C_u| \times |C_u|}\right) &= N\left(\ldots\right) \frac{\frac{\partial}{\partial \mu_l^{(k)}}\left(\frac{1}{\sqrt{2\pi}\sigma_{l,(k)}} \exp\left(-\frac{1}{2\sigma_{l,(k)}^2}\left(x_l^{(u)} - \mu_l^{(k)}\right)^2\right)\right)}{\left(\frac{1}{\sqrt{2\pi}\sigma_{l,(k)}} \exp\left(-\frac{1}{2\sigma_{l,(k)}^2}\left(x_l^{(u)} - \mu_l^{(k)}\right)^2\right)\right)} \\ &= N\left(\ldots\right) \frac{x_l^{(u)} - \mu_l^{(k)}}{\sigma_{l,(k)}^2}\end{aligned}$$

where $N\left(\ldots\right) = N\left(x_{C_u}^{(u)} | \mu_{C_u}^{(k)}, \sigma_k^2 I_{|C_u| \times |C_u|}\right)$.

If $l \notin C_u$, that derivative is 0. To cover both cases, we can write:

$$\frac{\partial}{\partial \mu_l^{(k)}} N\left(x_{C_u}^{(u)} | \mu_{C_u}^{(k)}, \sigma_k^2 I_{|C_u| \times |C_u|}\right) = N\left(x_{C_u}^{(u)} | \mu_{C_u}^{(k)}, \sigma_k^2 I_{|C_u| \times |C_u|}\right) \delta\left(l, C_u\right) \frac{x_l^{(u)} - \mu_l^{(k)}}{\sigma_{l,(k)}^2}$$

where $\delta\left(i, C_u\right)$ is an indicator function: 1 if $i \in C_u$ and zero otherwise.

Following the EM algorithm's approach of maximizing a proxy likelihood function $\hat{\ell}\left(X;\theta\right)$ during the M step, consider the following function:

$$\begin{aligned}\hat{\ell}\left(X;\theta\right) = &\sum_{u=1}^{n} \sum_{j=1}^{K} p\left(j \mid u\right) \log\left(\frac{p\left(x^{(u)} \text{ generated by cluster } j;\theta\right)}{p\left(j \mid u\right)}\right) \\ = &\sum_{u=1}^{n} \sum_{j=1}^{K} p\left(j \mid u\right) \log\left(\frac{\pi_j \mathcal{N}\left(x_{C_u}^{(u)} \mid \mu_{C_u}^{(j)}, \sigma_j^2 I_{|C_u| \times |C_u|}\right)}{p\left(j \mid u\right)}\right),\end{aligned}$$

where $p\left(x^{(u)} \text{ generated by cluster } j;\theta\right)$ is the likelihood of $x^{(u)}$ generated by cluster j and the parameter set is θ. The values $p\left(j \mid u\right)$ are the ones as we computed in the E step and they are constants for the M step.

We now take the derivative of $\hat{\ell}\left(X;\theta\right)$ with respect to $\mu_l^{(k)}$ to find the optimal value of $\mu_l^{(k)}$ that maximizes $\hat{\ell}\left(X;\theta\right)$.

$$\begin{aligned}\frac{\partial \hat{\ell}\left(X;\theta\right)}{\partial \mu_l^{(k)}} = &-\frac{\partial}{\partial \mu_l^{(k)}}\left[\sum_{u=1}^{n} \sum_{j=1}^{K} p\left(j \mid u\right) \cdot \frac{1}{2} \cdot \frac{\|x_{C_u}^{(u)} - \mu_{C_u}^{(j)}\|^2}{\sigma_j^2}\right] \\ = &\sum_{u=1}^{n} p\left(k \mid u\right) \delta\left(l, C_u\right) \frac{x_l^{(u)} - \mu_l^{(k)}}{\sigma_k^2},\end{aligned}$$

where $\delta\left(i, C_u\right) = 1$ if $i \in C_u$ and $\delta\left(i, C_u\right) = 0$ if $i \notin C_u$.

Generating Speech Output

Setting the partial derivative equal to zero, we obtain that

$$\widehat{\mu_l^{(k)}} = \frac{\sum_{u=1}^{n} p\left(k \mid u\right) \delta\left(l, C_u\right) x_l^{(u)}}{\sum_{u=1}^{n} p\left(k \mid u\right) \delta\left(l, C_u\right)}.$$

We leave it as an exercise to the reader to obtain the estimates of σ_k^2 and π_k for $k = 1, \ldots, K$. Verify that

$$\begin{aligned} \widehat{\sigma_k^2} &= \frac{1}{\sum_{u=1}^{n} |C_u| p\left(k \mid u\right)} \sum_{u=1}^{n} p\left(k \mid u\right) \left\| x_{C_u}^{(u)} - \widehat{\mu_{C_u}^{(k)}} \right\|^2, \\ \widehat{\pi_k} &= \frac{1}{n} \sum_{u=1}^{n} p\left(k \mid u\right). \end{aligned}$$

Implementation guidelines:

- You may find LogSumExp useful. But remember that your M-step should return the new $P = \hat{\pi}$, not the log of $\hat{\pi}$.
- The following will not affect the update equation above, but will affect your implementation: since we are dealing with incomplete data, we might have a case where most of the points in cluster j are missing the i-th coordinate. If we are not careful, the value of this coordinate in the mean will be determined by a small number of points, which leads to erratic results. Instead, we should only update the mean when $\sum_{u=1}^{n} p\left(j|u\right) \delta\left(i, C_u\right) \geq 1$. Since $p\left(j|u\right)$ is a soft probability assignment, this corresponds to the case when at least one full point supports the mean.
- To also avoid the variances of clusters going to zero due to a small number of points being assigned to them, in the M-step you will need to implement a minimum variance for your clusters. We recommend a value of 0.25, though you are free to experiment with it if you wish. Note that this issue, as well as the thresholded mean update in the point above, are better dealt with through regularization; however, to keep things simple, we do not do regularization here.
- To debug your EM implementation, you may use the data files `test_incomplete.txt` and `test_complete.txt`. Compare your results to ours from `test_solutions.txt`.

Implementing E-step (2)

0.0/1.0 point (graded)

In `em.py`, fill in the `estep` function so that it works with partially observed vectors where missing values are indicated with zeros, and perform the computations in the log domain to help with numerical stability.

Available Functions: You have access to the NumPy python library as `np`, to the `GaussianMixture` class and to typing annotation `typing.Tuple` as `Tuple`. You also have access to `scipy.special.logsumexp` as `logsumexp`

Hint: For this function, you will want to use `log(mixture.p[j] + 1e-16)` instead of `log(mixture.p[j])` to avoid numerical underflow

```
def estep(X: np.ndarray, mixture: GaussianMixture) -> Tuple[np.ndarray, float]:
    ...p: Softly assigns each datapoint to a gaussian component
```

Genera Speech Output

```

    Args:
        X: (n, d) array holding the data, with incomplete entries (set to 0)
        mixture: the current gaussian mixture

    Returns:
        np.ndarray: (n, K) array holding the soft counts
            for all components for all examples
        float: log-likelihood of the assignment

    """
    raise NotImplementedError

```

Press ESC then TAB or click outside of the code editor to exit

Unanswered

Generating Speech Output

```
def estep(X: np.ndarray, mixture: GaussianMixture) -> Tuple[np.ndarray, float]:
    """E-step: Softly assigns each datapoint to a gaussian component

    Args:
        X: (n, d) array holding the data, with incomplete entries (set to 0)
        mixture: the current gaussian mixture

    Returns:
        np.ndarray: (n, K) array holding the soft counts
            for all components for all examples
        float: log-likelihood of the assignment

    """
    n, _ = X.shape
    K, _ = mixture.mu.shape
    post = np.zeros((n, K))

    ll = 0
    for i in range(n):
        mask = (X[i, :] != 0)
        for j in range(K):
            log_likelihood = log_gaussian(X[i, mask], mixture.mu[j, mask],
                                          mixture.var[j])
            post[i, j] = np.log(mixture.p[j] + 1e-16) + log_likelihood
        total = logsumexp(post[i, :])
        post[i, :] = post[i, :] - total
        ll += total

    return np.exp(post), ll


def log_gaussian(x: np.ndarray, mean: np.ndarray, var: float) -> float:
    """Computes the log probablity of vector x under a normal distribution

    Args:
        x: (d, ) array holding the vector's coordinates
        mean: (d, ) mean of the gaussian
        var: variance of the gaussian

    Returns:
        float: the log probability
    """
    d = len(x)
    log_prob = -d / 2.0 * np.log(2 * np.pi * var)
    log_prob -= 0.5 * ((x - mean)**2).sum() / var
    return log_prob
```

Submit You have used 0 of 25 attempts

Answers are displayed within the problem

Implementing M-step (2)

0.0/1.0 point (graded)

In `em.py` , fill in the `mstep` function so that it works with partially observed vectors where missing values are

Generating Speech Output ros, and perform the computations in the log domain to help with numerical stability.

Available Functions: You have access to the NumPy python library as `np`, to the `GaussianMixture` class and to typing annotation `typing.Tuple` as `Tuple`.

```
def mstep(X: np.ndarray, post: np.ndarray, mixture: GaussianMixture,
          min_variance: float = .25) -> GaussianMixture:
    """M-step: Updates the gaussian mixture by maximizing the log-likelihood
    of the weighted dataset

    Args:
        X: (n, d) array holding the data, with incomplete entries (set to 0)
        post: (n, K) array holding the soft counts
            for all components for all examples
        mixture: the current gaussian mixture
        min_variance: the minimum variance for each gaussian

    Returns:
        GaussianMixture: the new gaussian mixture
    """
```

Press ESC then TAB or click outside of the code editor to exit

Unanswered

Generating Speech Output

```
def mstep(X: np.ndarray, post: np.ndarray, mixture: GaussianMixture,
          min_variance: float = .25) -> GaussianMixture:
    """M-step: Updates the gaussian mixture by maximizing the log-likelihood
    of the weighted dataset

    Args:
        X: (n, d) array holding the data, with incomplete entries (set to 0)
        post: (n, K) array holding the soft counts
            for all components for all examples
        mixture: the current gaussian mixture
        min_variance: the minimum variance for each gaussian

    Returns:
        GaussianMixture: the new gaussian mixture
    """
    n, d = X.shape
    _, K = post.shape

    n_hat = post.sum(axis=0)
    p = n_hat / n

    mu = mixture.mu.copy()
    var = np.zeros(K)

    for j in range(K):
        sse, weight = 0, 0
        for l in range(d):
            mask = (X[:, l] != 0)
            n_sum = post[mask, j].sum()
            if (n_sum >= 1):
                # Updating mean
                mu[j, l] = (X[mask, l] @ post[mask, j]) / n_sum
            # Computing variance
            sse += ((mu[j, l] - X[mask, l])**2) @ post[mask, j]
            weight += n_sum
        var[j] = sse / weight
        if var[j] < min_variance:
            var[j] = min_variance

    return GaussianMixture(mu, var, p)
```

Submit

You have used 0 of 25 attempts

Answers are displayed within the problem

Implementing run

0.0/1.0 point (graded)

In `em.py`, fill in the `run` function so that it runs the EM algorithm. As before, the convergence criteria that you should use is that the improvement in the log-likelihood is less than or equal to 10^{-6} multiplied by the absolute value of the new log-likelihood.

Available Functions: You have access to the NumPy python library as `np`, to the `GaussianMixture` class and to typing annotation `typing.Tuple` as `Tuple`. You also have access to the `estep` and `mstep` functions you have

Generating Speech Output

```
def run(X: np.ndarray, mixture: GaussianMixture,
        post: np.ndarray) -> Tuple[GaussianMixture, np.ndarray, float]:
    """Runs the mixture model

    Args:
        X: (n, d) array holding the data
        post: (n, K) array holding the soft counts
            for all components for all examples

    Returns:
        GaussianMixture: the new gaussian mixture
        np.ndarray: (n, K) array holding the soft counts
            for all components for all examples
        float: log-likelihood of the current assignment
    """
```

Press ESC then TAB or click outside of the code editor to exit

Unanswered

```
def run(X: np.ndarray, mixture: GaussianMixture,
        post: np.ndarray) -> Tuple[GaussianMixture, np.ndarray, float]:
    """Runs the mixture model

    Args:
        X: (n, d) array holding the data
        post: (n, K) array holding the soft counts
            for all components for all examples

    Returns:
        GaussianMixture: the new gaussian mixture
        np.ndarray: (n, K) array holding the soft counts
            for all components for all examples
        float: log-likelihood of the current assignment
    """

    prev_ll = None
    ll = None
    while (prev_ll is None or ll - prev_ll > 1e-6 * np.abs(ll)):
        prev_ll = ll
        post, ll = estep(X, mixture)
        mixture = mstep(X, post, mixture)

    return mixture, post, ll
```

Submit

You have used 0 of 25 attempts

Answers are displayed within the problem

Discussion

Hide Discussion

Topic: Unit 4 Unsupervised Learning (2 weeks) :Project 4: Collaborative Filtering via Gaussian Mixtures / 7. Implementing EM for matrix completion

Add a Post

Generating Speech Output

Show all posts by recent activity

Tip: use the numpy.ma module!
I found the numpy.ma module very helpful for this exercise. It enables the creation of masked arrays, which are normal numpy arrays ... 12
Pinned

My test files for project 4.7. Implementing EM for matrix completion
A bit late in the day but if you still need some test files, mine are [here][1]. Readme [here][2]. Praful [1]: https://github.com/Praful/MITx... 2
Pinned Community TA

[staff] help with M-step
Dear Staff, I want to continue to work on this project. I got the e-step and run function right but I having issues with M.step. could you ... 1

[Solved] [Help] - M-step troubles with variance
[edit]: the below problem was solved. I just scraped all the piece of code I used to have for the variance part and restarted with differe... 3

Posterior of an all zero row
How can I calculate the posterior of a row which is all zero. In my opinion, there should be no posterior for this since the point don't ex... 3

Is this vectrorizable?
May be a silly question, but has anyone managed a fully vectorized version of this? 16

I solved the E-step without using logs. Is this somehow wrong?
I passed the test without implementing the calculations with the logs. I wonder whether I'm missing something here. Anyone else solv... 6

Stuck in the derivation for estimates of π_k 3

E step
I stuck in e step. My result for nonzero values in X is right, but with zero values i have problem. Help me plesae, any hint. I am getting fr... 12

[Staff] 4

[STAFF] Please check my Implementing M-step (2)
I got all the answers correct using the same function "M Step" . But I managed to get only partial score by grader. My answer is wrong f... 1

[STAFF] EStep Log Likelihood Calculation
For the EStep function, I am able to calculate the posterior distribution correctly for all test cases, but for some reason, my log-likelihoo... 5

[SOLVED] M-Step - mu doesn't update when it should

Generating Speech Output