CS 181 Spring 2020 Section 7
Mixture Models, EM, PCA

1 Mixture Models

1.1 Review of notation

Vectors are denoted using **bold letters**. In the statement,

"Assume you have observed data $\{\mathbf{x}_i\}_{i=1}^N$."

This means there are some constant N distinct observations. Each observation $\mathbf{x}_i$ is a vector, where each component of the vector represents each feature.

When used to define distributions, the **semicolon** means that you can plug in the deterministic value of the variable after the semicolon into your expression for the distribution. Typically the variables that appear after the semicolon are unknown parameters for which you have some fixed estimate of what they may be.

For example, say you are given that random variable $x \sim N(0, \sigma)$, where σ is unknown. You believe that $\sigma = 2$. Then

$$p(x; \sigma) \sim N(0, 2)$$

When reading mathematical expressions, pay close attention to which variables are *random* (random variables can be *observed* or *unobserved*), and which variables are *deterministic constants*.

1.2 Motivation

Textbook sections 9.1, 9.2.

A *mixture model* is a type of probabilistic model for unsupervised learning.

Assume you have observed data $\{\mathbf{x}_i\}_{i=1}^N$.

Mixture models are used when you have reason to believe that each individual has an unobserved discrete latent variable $\mathbf{z}_i$ that determines the data generating process. Say there are C possible values for each $\mathbf{z}_i$, denoted $\{C_k\}_{k=1}^c$ where each C_k is a one-hot encoded vector of length C.

Assume that each data point $\mathbf{x}_i$ is generated as follows:

- Sample latent class $\mathbf{z}_i$ from $\boldsymbol{\theta}$, the categorical distribution over $\{C_k\}_{k=1}^c$ s.t. $p(\mathbf{z} = C_k; \boldsymbol{\theta}) = \theta_k$. Call this sampled latent class C_S.

- Given that $\mathbf{z}_i = C_S$, sample $\mathbf{x}_i$ from the distribution

  $$p(\mathbf{x}|\mathbf{z} = C_S; \mathbf{w})$$

  This conditional distribution is a modeling assumption (which means we will give it to you in this class), and is specified using unknown parameters $\mathbf{w}$.

  For example, we may assume that $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z} = C_k) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ are the unknown mean and covariance of the kth cluster. (See Section 2.4 for more about Gaussian mixture models!)

Example: Say you have a dataset containing weights from a random sample of animals in a pet store. Each x_i is the animal's weight. The latent variables z_i represent what kind of animal is being weighed, so the possible values $\{C_1, C_2, ..., C_C\}$ may represent the categories cat, dog, bird, etc. In your model, you also use the assumption that $p(x|z = C_k; \mathbf{w}) \sim N(\mu_k, \sigma_k)$.

Exercise 1. *In this example, can you give an intuitive explanation of what vector $\boldsymbol{\theta}$ represents? What does it mean that $p(x|z = C_k; \mathbf{w}) \sim N(\mu_k, \sigma_k)$?*

Solution. $\boldsymbol{\theta}$ represents the proportion of each type of animal in the pet store. For example, if $\theta_k = 0.2$ and C_k represents dogs, then 20% of all animals in the pet stores are dogs (or, if an animal is chosen at random, there is a 20% chance it will be a dog).

This class-conditional distribution means that for each type of animal, the weights of the animal are distributed normally with a fixed mean and variance for that animal. □

2 Expectation Maximization

Textbook sections 9.3, 9.4.

Expectation maximization is a general technique for maximum-likelihood estimation used primarily for models with latent variables. Here we will show how to use EM to train a mixture model, but EM is also used for a variety of other models!

Consider a generative mixture model consisting of a latent variable $\mathbf{z}$ from a distribution $p(\mathbf{z}; \boldsymbol{\theta})$ and an observed variable $\mathbf{x}$, such that we draw $\mathbf{x}$ from a distribution $p(\mathbf{x}|\mathbf{z}; \mathbf{w})$.

We have 2 goals:

1. To compute the MLE for $\mathbf{w}$ and $\boldsymbol{\theta}$, i.e. the values of $\mathbf{w}, \boldsymbol{\theta}$ that maximize $p(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta})$.
2. To estimate the latent variable $\mathbf{z}_i$ corresponding to a particular $\mathbf{x}_i$, which in this case means maximize the distribution $p(\mathbf{z}_i|\mathbf{x}_i; \mathbf{w}, \boldsymbol{\theta})$.

Goal 2 is easy once we have an estimate of the MLE for $\mathbf{w}, \boldsymbol{\theta}$, because we can apply Bayes rule:

$$p(\mathbf{z}|\mathbf{x}; \mathbf{w}, \boldsymbol{\theta}) \propto p(\mathbf{x}|\mathbf{z}; \mathbf{w}, \boldsymbol{\theta}) p(\mathbf{z}; \mathbf{w}, \boldsymbol{\theta})$$

$$p(\mathbf{z}|\mathbf{x}; \mathbf{w}, \boldsymbol{\theta}) \propto p(\mathbf{x}|\mathbf{z}; \mathbf{w}) p(\mathbf{z}; \boldsymbol{\theta}) \tag{1}$$

2.1 Why EM?

The likelihood of the data can be written as

$$p(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta}) = \sum_{\mathbf{z} \in Z} p(\mathbf{x}, \mathbf{z}; \mathbf{w}, \boldsymbol{\theta})$$

Unfortunately calculating the MLE is often computationally intractable, because the log-likelihood is:

$$\log p(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta}) = \log \sum_{\mathbf{z} \in Z} p(\mathbf{x}, \mathbf{z}; \mathbf{w}, \boldsymbol{\theta}) \tag{2}$$

There is no closed form for the MLE of the log-likelihood because it is the log of a sum of expressions. We know the form of the model $p(\mathbf{x}, \mathbf{z}; \mathbf{w}, \boldsymbol{\theta})$, but in general we cannot solve for the $(\mathbf{w}, \boldsymbol{\theta})$ which maximize the likelihood $p(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta})$ in closed form.

2.2 The EM Algorithm

Since finding the MLE directly is difficult, we will use expectation maximization: an approximate iterative approach. The steps of the algorithm are:

1. Initialize $\mathbf{w}^{(0)}, \boldsymbol{\theta}^{(0)}$ randomly.
2. Calculate the distribution $\mathbf{q}$ over $\mathbf{z}$:

$$q_{i,k} := p(\mathbf{z}_i = C_k|\mathbf{x}_i; \mathbf{w}^{(t)}, \boldsymbol{\theta}^{(t)}) \propto p(\mathbf{x}_i|\mathbf{z}_i; \mathbf{w}^{(t)},) p(\mathbf{z}_i; \boldsymbol{\theta}^{(t)}) \tag{3}$$

3. Choose the value of $\mathbf{w}^{(t+1)}, \boldsymbol{\theta}^{(t+1)}$ that maximizes the expected complete data log likelihood (where the expectation is over the distribution calculated above):

$$\mathbf{w}^{(t+1)}, \boldsymbol{\theta}^{(t+1)} = \underset{\mathbf{w}, \boldsymbol{\theta}}{\operatorname{argmax}} \, \mathbb{E}_{\mathbf{q}} \left[\log p(\mathbf{x}, \mathbf{z}; \mathbf{w}, \boldsymbol{\theta}) \right] \tag{4}$$

4. Go back to step 2 until the log-likelihood estimate in step 3 converges.

2.3 Example: Coins

Consider a setup where we have 2 biased coins C_1 and C_2, where $Pr(C_1 = 1) = \pi_1$ and $Pr(C_1 = 1) = \pi_2$. Data points x_i are generated by:

- First, flip another biased coin C_z.
- If C_z is heads, then x_i is the outcome of flipping C_1.
- Otherwise, if C_z is tails, then x_i is the outcome of flipping C_2.

We wish to do inference to learn the unknown parameters of the coins (π_1, π_2), but the only data we're given is the outcomes of the flips (the x_is).

Exercise 2. *In this example, what is a reasonable choice for the latent variables z_i?*

Solution. The latent variable z_i should represent whether coin C_1 or C_2 was flipped to generate x_i. □

This choice is reasonable because we know the distribution $p(x_i|z_i;,\theta)$ when z_i is the outcome of the first coin flip of C_z. To be consistent with Textbook Example 9.4.5, which uses the same model for the mixture of multinomials, we'll let $\mathbf{x}_i$ be a one-hot vector s.t. $\mathbf{x}_{i1} = 1$ if the result of coin flip i was heads; $\mathbf{x}_{i2} = 1$ otherwise. $\mathbf{z}_i$ is a one-hot vector (of size 2) indicating which coin was flipped to generate $\mathbf{x}_i$.

We'll denote the vector of probabilities for C_z used to choose between coins as $\boldsymbol{\theta} \in \mathbb{R}^2$, where θ_1 is the probability we'll pick C_1, and θ_2 for C_2. Finally, we'll use $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 \in \mathbb{R}^2$ to denote the biases for each coin, where $\boldsymbol{\pi}_1$ is the vector of probabilities for C_1, etc. This is exactly the same setup as in class for a mixture of multinomials; here we only have two multinomials here (coins 1 and 2) and they have 2 outcomes (heads or tails). We let $\mathbf{w} := \{\boldsymbol{\theta}, \boldsymbol{\pi}\}$.

Let's use expectation maximization to learn parameters $\mathbf{w} := \{\boldsymbol{\theta}, \boldsymbol{\pi}\}$!

First we note that we can calculate $\mathbf{q}_i$ from $\mathbf{w}^{(t)}$ by writing:

$$\mathbf{q}_i = \begin{bmatrix} p(\mathbf{z}_i = C_1|\mathbf{x}_i; \mathbf{w}^{(t)}) \\ p(\mathbf{z}_i = C_2|\mathbf{x}_i; \mathbf{w}^{(t)}) \end{bmatrix} \tag{5}$$

$$\propto \begin{bmatrix} p(\mathbf{x}_i|\mathbf{z}_i = C_1; \mathbf{w}^{(t)})p(\mathbf{z}_i = C_1; \mathbf{w}^{(t)}) \\ p(\mathbf{x}_i|\mathbf{z}_i = C_2; \mathbf{w}^{(t)})p(\mathbf{z}_i = C_2; \mathbf{w}^{(t)}) \end{bmatrix} \tag{6}$$

$$\propto \begin{bmatrix} (\pi_{11})^{x_{i1}}(\pi_{12})^{x_{i2}}\theta_1 \\ (\pi_{21})^{x_{i1}}(\pi_{22})^{x_{i2}}\theta_2 \end{bmatrix} \tag{7}$$

We also have the data log-likelihood as

$$\begin{aligned}
\log p(\mathbf{x}_i, \mathbf{z}_i; \mathbf{w}) &= \log p(\mathbf{x}_i|\mathbf{z}; \mathbf{w}) p(\mathbf{z}; \mathbf{w}) && (8)\\
&= \log \prod_{k=1}^{2} \left(\theta_k \prod_{j=1}^{2} \pi_{kj}^{x_{ij}} \right)^{z_{ik}} && (9)\\
&= z_{i1} \left(\log \theta_1 + x_{i1} \log \pi_{11} + x_{i2} \log \pi_{12} \right) && (10)\\
&+ z_{i2} \left(\log \theta_2 + x_{i1} \log \pi_{21} + x_{i2} \log \pi_{22} \right) && (11)\\
\log p(\mathbf{x}, \mathbf{z}; \mathbf{w}) &= \sum_{i=1}^{n} \log p(\mathbf{x}_i, \mathbf{z}_i; \mathbf{w}) && (12)
\end{aligned}$$

Now expand the expected complete data log-likelihood:

$$\begin{aligned}
\mathcal{L}_c &= \mathbb{E}_{\mathbf{z}|\mathbf{x};\mathbf{w}} \left[\sum_{i=1}^{n} \log p(\mathbf{x}_i, \mathbf{z}_i; \mathbf{w}) \right] && (13)\\
&= \mathbb{E}_{\mathbf{z}|\mathbf{x};\mathbf{w}} \left[\sum_{i=1}^{n} \log p(\mathbf{z}_i; \mathbf{w}) + \log p(\mathbf{x}_i|\mathbf{z}_i; \mathbf{w}) \right] && (14)\\
&= \sum_{i=1}^{n} \mathbb{E}_{\mathbf{z}|\mathbf{x};\mathbf{w}} \left[\log p(\mathbf{z}_i; \mathbf{w}) + \log p(\mathbf{x}_i|\mathbf{z}_i; \mathbf{w}) \right] && (15)\\
&= \sum_{i=1}^{n} \sum_{k=1}^{c} q_{ik} \left(\log \theta_k + \sum_{j=1}^{2} x_{ij} \log \pi_{kj} \right) && (16)\\
&= \sum_{i=1}^{n} q_{i1} \left(\log \theta_1 + x_{i1} \log \pi_{11} + x_{i2} \log \pi_{12} \right) + q_{i2} \left(\log \theta_2 + x_{i1} \log \pi_{21} + x_{i2} \log \pi_{22} \right) && (17)
\end{aligned}$$

Now we can use these derivations to do expectation maximization!:

1. Initialize $\mathbf{w}^{(0)}$ randomly.
2. Use $\mathbf{w}^{(t)}$ to calculate the vector of probabilities $\mathbf{q}_i$ for the distribution of each $\mathbf{z}_i$ (eq. 7).
3. Calculate the approximate expected likelihood using $\mathbf{q}_i$ and $\mathbf{w}^{(t)}$ (eq. 11).

   This step is not strictly necessary for calculating updates, but can be helpful for a variety of purposes, including debugging and testing convergence. Note that we need *both* $\mathbf{q}$ and $\mathbf{w}^{(t)}$ to get a value here.
4. Use $\mathbf{q}$ to calculate an updated set of parameters $\mathbf{w}^{(t+1)}$ by maximizing the expected likelihood as a function of $\mathbf{w}$ (eq. 17). Note that here we do *not* use $\mathbf{w}^{(t)}$.

During optimization we need to enforce that $\sum_k \theta_k = 1$ and that $\sum_j \pi_{kj} = 1$, so that the distributions parameterized by $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ are valid.

This constraint can be enforced using Lagrange Multipliers, but in the 2-dimensional case we can substitute $\theta_2 = 1 - \theta_1$ and $\pi_{k2} = 1 - \pi_{k1}$:

$$\mathcal{L}_c = \sum_{i=1}^{n} q_{i1}\left(\log \theta_1 + x_{i1} \log \pi_{11} + x_{i2} \log(1 - \pi_{11})\right) \tag{18}$$

$$+q_{i2}\left(\log(1 - \theta_1) + x_{i1} \log \pi_{21} + x_{i2} \log(1 - \pi_{21})\right) \tag{19}$$

And then optimize w.r.t. $\theta_1, \pi_{11}, \pi_{21}$:

$$\frac{\partial \mathcal{L}_c}{\partial \theta_1} = \sum_{i=1}^{n} \left(\frac{q_{i1}}{\theta_1} - \frac{q_{i2}}{1 - \theta_1}\right) = 0 \tag{20}$$

$$\frac{\partial \mathcal{L}_c}{\partial \pi_{11}} = \sum_{i=1}^{n} q_{i1} \left(\frac{x_{i1}}{\pi_{11}} - \frac{x_{i2}}{1 - \pi_{11}}\right) = 0 \tag{21}$$

$$\frac{\partial \mathcal{L}_c}{\partial \pi_{21}} = \sum_{i=1}^{n} q_{i2} \left(\frac{x_{i1}}{\pi_{21}} - \frac{x_{i2}}{1 - \pi_{21}}\right) = 0 \tag{22}$$

From here we can solve for the optimal value of $\mathbf{w}$ (i.e. $\theta_1, \pi_{11}, \pi_{22}$), and set $\mathbf{w}^{(t+1)} = \operatorname{argmax}_{\mathbf{w}} \mathbb{E}_{\mathbf{z}|\mathbf{x};\mathbf{w}} \mathcal{L}_c$.

Note: Above we show the derivation of all steps of the algorithm, but once you know the closed form expression for $\mathbf{w}^{(t+1)}$, the steps of the algorithm are really just initialization, calculate the distribution $\mathbf{q}_i$ from $\mathbf{w}^{(t)}$, and then calculate $\mathbf{w}^{(t+1)}$ from $\mathbf{q}$. All the difficult work is in deriving the update equations.

In more complicated models, the optimal $\mathbf{w}^{(t+1)}$ may not have a closed form solution; in these cases, instead we can do gradient descent to calculate the optimal value.

Exercise 3. *Derive the closed form updates for $\boldsymbol{\theta}^{(t)}$, $\boldsymbol{\pi}^{(t)}$ from the steps above.*

Solution.

$$\theta_k^{(t)} \leftarrow \frac{\sum_{i=1}^{n} q_{ik}}{n}$$

$$\boldsymbol{\pi}_k^{(t)} \leftarrow \frac{\sum_{i=1}^{n} q_{ik} \mathbf{x}_i}{\sum_{i=1}^{n} \sum_{j=1}^{m} q_{ik} x_{ij}}$$

□

Once we have an estimate for the MLE $\mathbf{w}$, we can use it to do prediction of hidden states for a new incoming coin flip, using step 2 from above. So, given a new coin flip, we can predict whether it can from the first or the second coin.

Our model may not be very good, since in particular it is impossible to tell the difference between having one coin chosen with high probability with $\pi_1 = 0.5$ (and another picked almost never with $\pi_2 = 0.1$) and two equally likely coins with biases 0.4 and 0.6. This problem is due more to the data setup rather than the method, so let's try another problem:

Exercise 4. *Consider the following data generation process: the setup is the same as above, but instead of flipping the chosen coin once, we flip it 10 times before choosing a new coin.*

1. *Find an appropriate choice of latent variables $\mathbf{z}_i$ and calculate the distribution of $\mathbf{z}_i$ given the data $\mathbf{x}_{i,j}$ (where i iterates over the sets of 10 coin flips, and $j \in [1, 10]$) and an estimate for $\boldsymbol{\theta}$.*
2. *Find the expression for the expected complete data log-likelihood*
3. *Find the closed form update equations for $\boldsymbol{\theta}^{(t)}$, and compare them to the result from Exercise 1.*

Solution.

1. Here again we can use $\mathbf{z}_i$ to denote the 1-hot choice of the chosen coin, this time used for all 10 of the flips. The distribution of $\mathbf{z}_i$ is again proportional to the prior times the likelihood, which is:

$$p(\mathbf{z}_i|\mathbf{x}_i, \mathbf{w}) \propto p(\mathbf{x}_i|\mathbf{z}_i, \mathbf{w}) \cdot p(\mathbf{z}_i|\mathbf{w}) = \prod_{k=1}^{c=2} \left(\prod_{j=1}^{10} \prod_{l=1}^{2} \pi_{kl}^{x_{ijl}} \right)^{z_{ik}} \cdot \prod_{k=1}^{c=2} \theta_k^{z_{ik}}$$

   where x_{ijl} is the indicator variable for the jth coin in the ith set of 10 being in class l (in this case, heads or tails).

2. The expected complete data loglikelihood is similar to the above case:

$$\mathcal{L}_c = \sum_{i=1}^{n} \sum_{k=1}^{c=2} q_{ik} \left(\log \theta_k + \sum_{j=1}^{10} \sum_{l=1}^{2} x_{ijl} \log \pi_{kj} \right)$$

3. In fact, $\boldsymbol{\theta}$ satisfies the same expression as before, so the update is the same:

$$\theta_k^{(t)} \leftarrow \frac{\sum_{i=1}^{n} q_{ik}}{n}$$

   The only thing that has changed here is the matrix $\mathbf{q}$. Remember that q_{ik} is proportional to the prior probability of choosing the coin, times the likelihood. Here the likelihood term is stronger (because we see more evidence per latent variable) so we can expect this estimate to be less noisy, and thus our estimation of θ is less noisy.

□

2.4 Example: Gaussian Mixture Modeling

Textbook section 9.5.

The setup is that we have data $\mathbf{x}_i \in \mathbb{R}^m$ and a latent variable $\mathbf{z}_i$ (corresponding to the cluster that the point is drawn from) such that $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z} = C_k) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ are the mean and covariance of the kth cluster. The choice of cluster is drawn from a categorical distribution with probabilities $\boldsymbol{\theta} \in \mathbb{R}^c$. We are able to observe the data $\mathbf{x}_i$ and want to find the cluster centers and their covariances.

Following the same format as above, the steps of EM inference applied to this problem are:

1. Randomly initialize $\boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k$.
2. Next, calculate the new distribution of each $\mathbf{z}_i$:

$$q_{ik} = p(z_i = C_k|\mathbf{x}_i) \propto \theta_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{23}$$

   This is our new estimate of the distribution of $\mathbf{z}_i$ given the data and our estimate for $\boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k$.

3. Find the expected complete data log-likelihood:

$$\begin{aligned}\mathbb{E}_{\mathbf{Z}}\left[\mathcal{L}\right] &= \mathbb{E}_{\mathbf{Z}}[\sum_{i=1}^{n} \ln(p(\mathbf{x}_i, \mathbf{z}_i; \boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k))] &(24)\\ &= \sum_{i=1}^{n}\sum_{k=1}^{c} q_{ik} \ln \theta_k + q_{ik} \ln \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) &(25)\end{aligned}$$

   and then optimize it for each of the parameters $\boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k$. However, we need to be careful to remember constraints: since $\sum_k \theta_k = 1$, we must use Lagrange multipliers to optimize the parameters. We get the following update equations:

$$\begin{aligned}\theta_k^{(t+1)} &= \frac{\sum_{i=1}^{n} q_{ik}}{n} &(26)\\ \boldsymbol{\mu}_k^{(t+1)} &= \frac{\sum_{i=1}^{n} q_{ik}\mathbf{x}_i}{\sum_{i=1}^{n} q_{ik}} &(27)\\ \boldsymbol{\Sigma}_k^{(t+1)} &= \frac{\sum_{i=1}^{n} q_{ik}(\mathbf{x}_i - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_i - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_{i=1}^{n} q_{ik}} &(28)\end{aligned}$$

3 Principal Component Analysis

Textbook chapter 7.

3.1 Motivation

In many supervised learning problems, we try to find rich features that increase the expressivity of our model. In practice, this often involves using basis functions to transform model input into a higher dimensional space (eg. given data x, using x and x^2 as features, or using features learned by a neural network).

However, sometimes we want to reduce the dimensionality of our data.

Exercise 5. *Why would we want to reduce the dimensionality of our data? Can you think of example cases?*

Solution. Solution: There can be several reasons:

- Fewer features are easier to interpret: we might want to know why our model outputs a certain diagnosis, and only some of the patient record details will be relevant.
- Models with fewer features are easier to handle computationally.
- Our data might be arbitrarily high-dimensional because of noise, so we would like to access the lower-dimensional *signal* from the data.

□

One method for dimensionality reduction through **linear projections** of the original data is PCA. When reducing the dimensionality of our data from m to d where $d < m$, PCA can be interpreted as minimizing the reconstruction loss of projecting data onto d basis vectors, or as maximizing the variance in data that can be explained by d basis vectors.

3.2 Finding the lower dimensional representation

To perform PCA, or project each data point $\mathbf{x}$: $(m \times 1)$ to $\mathbf{z}$: $(d \times 1)$,

1. Standardize the data by subtracting the mean of each feature from each data point. Steps 2 - 5 will be performed on the standardized data $\mathbf{X}$: $(n \times m)$.
2. Calculated the normalized **feature covariance** matrix:

$$\mathbf{S} = (\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top) = \mathbf{X}^\top \mathbf{X}$$

3. Decide how many dimensions d out of the original m that we want to keep in the final representation. For visualizations, often this will be $d = 2$ or $d = 3$.

4. Find the d largest eigenvalues of $\mathbf{S}$. The $m \times 1$ eigenvectors $(\mathbf{u}_1, \ldots, \mathbf{u}_d)$ corresponding to these eigenvalues will be our lower-dimensional basis.

5. Thus, we reduce the dimensionality of a data point $\mathbf{x}$ by projecting it onto this basis - we combine the eigenvectors into the $m \times d$ matrix $\mathbf{U}$, and compute

$$\langle \mathbf{u}_1^\top \mathbf{x}, \mathbf{u}_2^\top \mathbf{x}, \ldots, \mathbf{u}_d^\top \mathbf{x}, 0 \rangle = \mathbf{U}^\top \mathbf{x} = \mathbf{z}$$

   $\mathbf{z}$ is called the reconstruction coefficients where $\mathbf{Uz}$ is the reconstruction of $\mathbf{x}$.

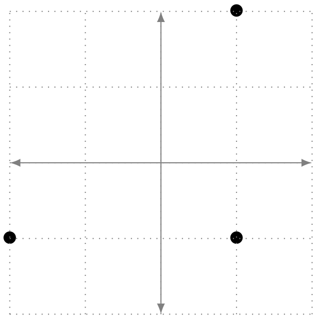
Exercise 6. *You are given the following data set:*

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} -2 \\ -1 \end{bmatrix}$$

You would like to use PCA to find a 1-dimensional representation of the data.

1. *Plot the data set.*
2. *Compute the feature covariance matrix* $\mathbf{S}$.
3. *You find that* $\mathbf{S}$ *has eigenvector* $[-1\ 1]^\top$ *with eigenvalue* 3 *and eigenvector* $[1\ 1]^\top$ *with eigenvalue* 9. *What is the (normalized) basis vector* $\mathbf{u}_1$ *of your 1-dimensional representation? Add the basis vector* $\mathbf{u}_1$ *to your plot.*
4. *Compute the coefficients* z_1, z_2, z_3. *Add the lower-dimensional representations* $z_1\mathbf{u}_1, z_2\mathbf{u}_1, z_3\mathbf{u}_1$ *to your plot. Based on your plot, what is the relationship between* $z_i\mathbf{u}_1$ *and* $\mathbf{x}_i$ *with respect to the new basis?*
5. *Based on your plot, what would happen if you chose the unused eigenvector to be your basis vector?*

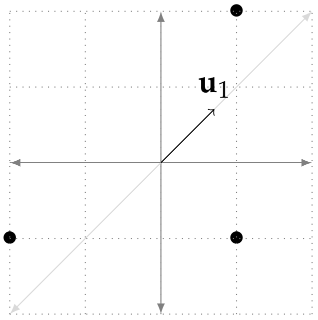
Solution. 1.



2.

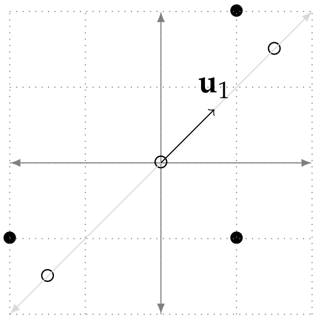
$$\mathbf{S} = \mathbf{X}^\top \mathbf{X} = \begin{bmatrix} 1 & -1 \\ 1 & 2 \\ -2 & -1 \end{bmatrix}^\top \begin{bmatrix} 1 & -1 \\ 1 & 2 \\ -2 & -1 \end{bmatrix} = \begin{bmatrix} 6 & 3 \\ 3 & 6 \end{bmatrix}$$

3. We select the eigenvectors with the largest eigenvalues for our basis, so our basis will contain a scalar multiple of $[1\ 1]^\top$. Normalizing $[1\ 1]^\top$ gives us that $\mathbf{u}_1 = [\frac{\sqrt{2}}{2}\ \frac{\sqrt{2}}{2}]^\top$.



4.
$$z_1 = \mathbf{x}_1^\top \mathbf{u}_1 = 0, \quad z_2 = \mathbf{x}_2^\top \mathbf{u}_1 = \frac{3\sqrt{2}}{2}, \quad z_3 = \mathbf{x}_3^\top \mathbf{u}_1 = -\frac{3\sqrt{2}}{2}$$

The open circles in the plot represent the lower-dimensional representation:



$z_i \mathbf{u}_1$ is the projection of $\mathbf{x}_i$ onto the basis vector.

5. If we chose $[-1\,1]^\top$ to be the basis of our new representation, then the representation would capture less of the variance in the data. For example, $\mathbf{x}_2$ and $\mathbf{x}_3$ would be represented by the same point.

□