Section 10: Solutions

1. Kernels

Let $\phi : d \to k$ be a feature map, and define K to be the kernel matrix of ϕ.

(a) Prove that the kernel matrix is symmetric. That is, show $K_{i,j} = K_{j,i}$.

Solution:

Let $\phi(x_i)$ and $\phi(x_j)$ be the feature maps for x_i and x_j, respectively. Then $K_{i,j} = \phi(x_i)^T\phi(x_j) = \phi(x_j)^T\phi(x_i) = K_{j,i}$

(b) Show that K is positive semi-definite
Hint: consider the matrix B where the i^{th} column of B is $\phi(x_i)$.

Solution:

Recall that $K_{i,j} = \phi(x_i)^T\phi(x_j)$. Observe that $K = B^TB$, as $(B^TB)_{i,j} = \phi(x_i)^T\phi(x_j)$. Now consider an arbitrary vector y. To show K is PSD it suffices to show y^TKy is non-negative. We have:

$$y^TKy = y^TB^TBy = (By)^T(By) = ||By||_2^2 \geq 0$$

2. PCA

Consider the following data set, represented as three points in $\mathbb{R}^2$. Note: in this problem we will **not** demean the dataset. Perform all calculations as if the dataset were 0 mean.

$$\begin{bmatrix} 1 & 2 \\ 1.5 & 3 \\ 6 & 12 \end{bmatrix}$$

(a) What is the first principal component vector, v_1?

Solution:

Each point has second coordinate twice the first, so every point is on the line $y = 2x$, or equivalently is a multiple of the vector $[1, 2]$.

That direction, normalized, is the first principal component, so $v_1 = [1/\sqrt{5}, 2/\sqrt{5}]$.

(b) What is the second principal component, v_2?

Solution:

Since there is no data not in the span of the first principal component, any unit norm vector perpendicular to v_1 is an acceptable choice. One such vector is $[-2\sqrt{5}, 1/\sqrt{5}]$.

(c) If we use only the first principal component to compress the dataset, what will the representation of each point be?

Solution:

The first point is $\sqrt{5}v_1$, The second is $1.5 \cdot \sqrt{5}v_1$, The third is $6 \cdot \sqrt{5}v_1$.

(d) Will this representation be lossy, or perfectly preserve the data?

Solution:

On this particular dataset, we have lost no information (the points are all multiples of v_1).

Answer the same questions for the following, slightly larger dataset:

$$\begin{bmatrix} 1 & 1 \\ 1.5 & 1.5 \\ -2 & 2 \\ 4 & -4 \\ 6 & -6 \\ 2 & 2 \end{bmatrix}$$

(a) What is the first principal component vector, v_1? **Solution:**

Notice that every point is either a multiple of $[1, 1]$ or $[1, -1]$, so one of those must be our principal component. The norms of the multiples of $[1, -1]$ are much larger, so $[1/\sqrt{2}, -1/\sqrt{2}]$ is v_1.

(b) What is the second principal component, v_2?

Solution:

We need a vector perpendicular to v_1, that best describes our remaining data. Since we're in two dimensions, we don't have choices (beyond negating the vector) $[1/\sqrt{2}, 1/\sqrt{2}]$.

(c) If we use only the first principal component to compress the dataset, what will the representation of each point be?

Solution:

Data points 1, 2, and 6 are all perpendicular to v_1, so are represented as $[0, 0]$ (i.e. $0 \cdot v_1$). The other points are multiples of v_1: $-2\sqrt{2}v_1$, $4\sqrt{2}v_1$, and $6\sqrt{2}v_1$.

(d) Will this representation be lossy, or perfectly preserve the data?

Solution:

Lossy – points 1,2, and 6 have lost information.

In Lecture 17, we saw the following optimization problem in the context of autoencoders:

$$\min_{f,g} \sum_{i=1}^{n} ||x_i - g(f(x_i))||_2^2$$

Suppose we know that $f(x) = Ax$ for some matrix $A \in \mathbb{R}^{n \times d}$ and $g(y) = By$ for some matrix $B \in \mathbb{R}^{d \times n}$.

(a) How could you calculate A, B? **Solution:**

Our form is to minimize $\sum_{i=1}^{n} ||x_i - BAx_i||_2^2$

So we are looking for rank at most d matrices whose product preserves the x_i well (notice that since the x_i are in n dimensions, there may not be a way to preserve all the data perfectly). The best d directions to preserve the matrix A are exactly the top d directions in the SVD, or equivalently the top d principal components.

(b) In the special case that $d = n$ what happens?

Solution:

If $d = n$ then BA can be a full-rank (i.e. rank n) matrix. In particular, BA could be I_n. Any inverse matrices will create 0 error in the minimization (in particular we could take the full SVD here, though the solutions is no longer unique in this context.)

3. SVD

We will now explore SVD to transform matrices into another form. Recall that we can decompose any matrix by SVD into 3 components USV^T where matrices U and V^T are othornormal square matrices and S is a rectangular diagonal matrix.

Here we will use an example from lecture on kernels (Lecture 12). In that lecture we considered the optimization problem

$$\arg\min_w \sum_{i=1}^{n} (y_i - \phi(x_i)^T w)^2 + \lambda ||w||_2^2$$

, and we said that both of the following expressions were closed forms for the optimum $\hat{w}$:

$$\hat{w} = (\Phi^T\Phi + \lambda I_p)^{-1}\Phi^T y$$

and

$$\hat{w} = \Phi^T(\Phi\Phi^T + \lambda I_n)^{-1} y$$

Let USV^T be the SVD decomposition of Φ. Using the second of those two expressions, show $\hat{w} = VS^T(SS^T + \lambda I)^{-1}U^T y$. (This exercise is the first step in the proof that those two forms are equivalent – to complete the proof you would show the other expression can also be rewritten in this form).

Solution:

$$\begin{aligned}
\hat{w} &= \Phi^T(\Phi\Phi^T + \lambda I_n)^{-1} y && \\
&= \Phi^T(USV^TVS^TU^T + \lambda I_n)^{-1} y && \Phi = USV^T \\
&= \Phi^T(USV^TVS^TU^T + \lambda UU^T)^{-1} y && I_n = UU^T \\
&= \Phi^T(U(SV^TVS^T + \lambda U)U^T)^{-1} y && \text{Factorization} \\
&= VS^TU^TU(SS^T + \lambda I_n)^{-1}U^T y && U^{-1} = U^T \\
&= VS^T(SS^T + \lambda I_n)^{-1}U^T y && U^TU = I
\end{aligned}$$

4. More SVD, Gaussians

Let Σ be a 2×2 matrix with eigenvalues $v_1 = [1/2, \sqrt{3}/2]^T$ with eigenvalue 3 and $v_2 = [-\sqrt{3}/2, 1/2]^T$ with eigenvalue 2.

(a) Give an expression for Σ (Hint: look at the title of this section)

Solution:

We know what the SVD of Σ would be, so we can use that to find Σ:

$$\begin{bmatrix} 1/2 & -\sqrt{3}/2 \\ \sqrt{3}/2 & 1/2 \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1/2 & \sqrt{3}/2 \\ -\sqrt{3}/2 & 1/2 \end{bmatrix} = \begin{bmatrix} 9/4 & \sqrt{3}/4 \\ \sqrt{3}/4 & 11/4 \end{bmatrix}$$

(b) Let $\Sigma^{1/2}$ be the matrix such that $\Sigma^{1/2}\Sigma^{1/2} = \Sigma$. Give an expression for $\Sigma^{1/2}$. Do not try to simplify your expression.

Solution:

It's not too hard to see that $\Sigma^{1/2}$ must have the same eigenvectors as Σ.

Let v be an eigenvector of Σ with eigenvalue λ. Note that $\Sigma^{1/2}\Sigma^{1/2}v = \lambda v$ So if α is the eigenvalue associated with v for $\Sigma^{1/2}$, we have $\Sigma^{1/2}\Sigma^{1/2}v = \Sigma^{1/2}\alpha v = \alpha^2 v = \lambda v$. So the eigenvalues are just the squareroot of those for Σ.

Thus $\Sigma^{1/2}$ is:

$$\begin{bmatrix} 1/2 & -\sqrt{3}/2 \\ \sqrt{3}/2 & 1/2 \end{bmatrix} \begin{bmatrix} \sqrt{3} & 0 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 1/2 & \sqrt{3}/2 \\ -\sqrt{3}/2 & 1/2 \end{bmatrix}$$

(c) Recall that the density of a multivariate Gaussian with mean μ and covariance Σ is:

$$P(x) = \frac{1}{2\pi\sqrt{|\Sigma|}} \exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right]$$

Where $|.|$ is the determinant operator. Let $\mathcal{E}$ be the set of all points where the density is $e^{-3/2}/(2\sqrt{6}\pi)$, i.e. $\mathcal{E} = \{x|P(x) = e^{-3/2}/(2\sqrt{6}\pi)\}$ ($\mathcal{E}$ is an "isocontour"). Derive a simple formula for $\mathcal{E}$ (where μ is unknown, but Σ is the matrix from the first part). Leave your answer in terms of μ and Σ.

Solution:

Start by noticing that $|\Sigma|$ is 6 (you could figure this out by recalling that the determinant is always the product of the eigenvalues, by remembering that U only rotates so $|\Sigma| = |S|$ or by doing a full calculation)

Thus both the density and the right hand side have a denominator of $2\sqrt{6}\pi$ so we need to solve:

$$\exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right] = e^{-3/2}$$

So we have

$$(x-\mu)^T\Sigma^{-1}(x-\mu) = 3$$

(d) Consider the distribution $\mathcal{N}(\mu, \Sigma)$ with $\mu = [0, 1]^T$ and Σ being the matrix from part a. Draw the contour you described in the previous part. Hint: you should only need to think about the eigenvectors and eigenvalues of Σ to figure out what to draw.

The numbers defining $\mathcal{E}$ are not nice enough to graph by hand; it's enough to figure out its shape and the formulas for enough important points to make a plot.

Solution:

Since μ is just going to translate the drawing, let's first figure out the drawing when $\mu = [0,0]^T$ and translate it. We now need to find vectors x such that $x^T\Sigma^{-1}x = 3$. We know that the isocontours of the Gaussian distribution (the curves where the density has the same value) form ellipses, with the axes of the ellipse being the eigenvectors of the covariance. So it suffices to figure out what multiples of the eigenvectors are on the contour.

i.e. there's some α such that αv_1 is on the contour, and we want to find α.

$$\begin{aligned}(\alpha v_1)^T\Sigma^{-1}(\alpha v_1) &= (\alpha v_1)^T\left(\frac{\alpha}{3}v_1\right)\\ &= \frac{\alpha^2}{3}||v_1||^2\\ &= \frac{\alpha^2}{3}\end{aligned}$$

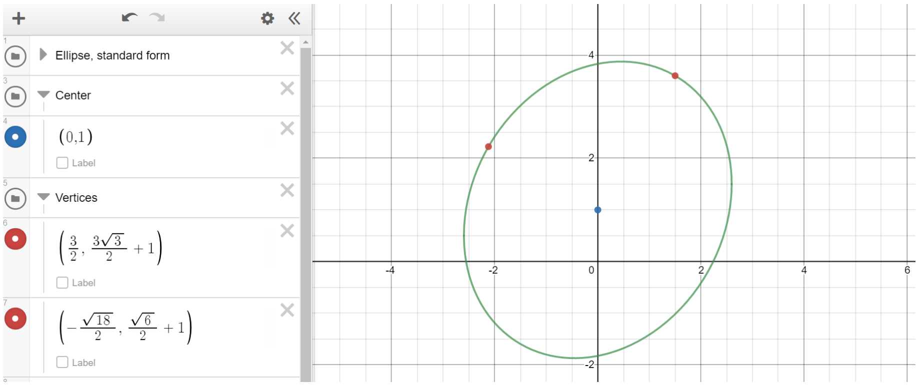
So we have $\alpha^2/3 = 3$ and $\alpha = 3$.

Thus the vector $3v_1 = [3/2, 3\sqrt{3}/2]^T$ is on the ellipse centered at 0.

A similar calculation for v_2 gives $\sqrt{6}v_2 = [-\sqrt{18}/2, \sqrt{6}/2]^T$

We can now account for μ, by simply shifting the entire drawing by $[0,1]^T$, so we have: $[3/2, 3\sqrt{3}/2+1]^T$ and $[-\sqrt{18}/2, \sqrt{6}/2+1]^T$ on the ellipse, now centered at $[0,1]^T$.

The drawing looks like this:



Ellipse, standard form

Center

$(0,1)$

□ Label

Vertices

$\left(\frac{3}{2}, \frac{3\sqrt{3}}{2}+1\right)$

□ Label

$\left(-\frac{\sqrt{18}}{2}, \frac{\sqrt{6}}{2}+1\right)$

□ Label

Neither of these matrices can the be the covariance of a Gaussian. For each, give a reason why it's not.

(a) $\begin{bmatrix}3 & 4\\ 4 & -1\end{bmatrix}$

Solution:

The diagonal of the covariance matrix has the variances of the individual variables. Variances cannot be negative.

(b) $\begin{bmatrix}1 & 1\\ 2 & 0\end{bmatrix}$ **Solution:**

A covariance matrix is always symmetric (since entry i, j and entry j, i are both the covariance of dimensions i and j).

In section we proved directly that for any matrix A, AA^T is positive semi-definite. Use the SVD of A to show the same result. **Solution:**

If $A = USV^T$ then $AA^T = USV^TVSU^T = US^2U^T$. A symmetric matrix is PSD if and only if all of its eigenvalues are non-negative. The eigenvalues of AA^T are exactly the diagonal entries of S^2, and these are definitely non-negative (since they are squares of the entries of S).

5. EM

Suppose that Kevin and Anna assign grades as follows: With probability 0.5 you get a 4.0, with probability μ you get a 3.7, with probability 2μ you get a 3.4 and with probability $0.5 - 3\mu$ you get a 3.0. (μ is assumed to be between 0 and $1/6$.)

(a) You wish to find the maximum likelihood estimate of μ from data. You somehow managed to get ahold of some data and found out that a 4.0's were assigned, b 3.7's were assigned, c 3.4's were assigned and d 3.0's were assigned. What is the MLE of μ given a, b, c, d?

Solution:

The likelihood of the grades seen is: $(.5)^a \cdot \mu^b \cdot (2\mu)^c \cdot (.5 - 3\mu)^d$. Since we have exponents and multiplication, it will be easier to deal with the log-likelihood:

$$a \log(1/2) + b \log(\mu) + c \log(2\mu) + d \log(.5 - 3\mu)$$

Set the derivative equal to 0 and solve:

$$\begin{aligned}
\frac{b}{\mu} + \frac{c}{\mu} + \frac{-3d}{.5 - 3\mu} &= 0 \\
b + c\frac{-3d\mu}{1/2 - 3\mu} &= 0 \\
(1/2 - 3\mu)(b + c) - 3d\mu &= 0 \\
\frac{b}{2} + \frac{c}{2} - 3b\mu - 3c\mu - 3d\mu &= 0 \\
\frac{b+c}{2} &= 3\mu(b + c + d) \\
\frac{b+c}{6(b+c+d)} &= \mu
\end{aligned}$$

(b) Now suppose that instead some information is hidden. Specifically, you are told c and d (the number of 3.4's and 3.0's), but you only know h which is the combined number of 4.0's and 3.7s. Describe how you would use EM to solve for μ by filling in the following:

E step: if you knew the value of μ, you could compute the expected value of a and b.

M step: If you knew the expected values of a and b, you could compute the MLE of μ.

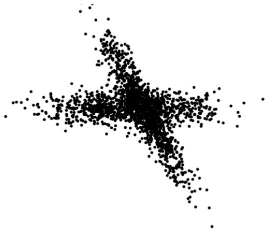
Solution:

Expectation step: We need to find the expectation of a and b given μ. We know that in expectation a has $1/2$ of the total mass and b has μ of it. Since we know a and b combine to h we can apply Baye's Rule to get: $\hat{a} = \frac{1/2}{1/2+\mu}h \quad \hat{b} = \frac{\mu}{1/2+\mu}h$

Maximization Step: We can just use the MLE we've already calculated. $\mu = \frac{b+c}{6(b+c+d)}$

6. Clustering Methods

You want to cluster your data into 2 clusters. For each of the following datasets, which of k-means and EM for Gaussian Mixture Models would provide a faithful representation? Answer k-means, GMM, Both, Neither.

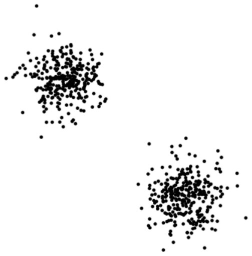


(a)

Solution:

GMM. k-means will fail badly here – since the clusters overlap heavily and are only differentiated by their orientation, k-means will be be able to find the center of each cluster, but since those centers are (essentially) on top of each other, we won't be able to differentiate between the clusters.

The mixture model can identify "directions" in the data so will be able to separate the extreme points accurately (though it won't be able to accurately classify the points near the center.



(b)

Solution:

Both will accurately classify the points.

7. Linear Regression

7.1. Regularization

You're running linear regression to predict from one-dimensional feature vectors. Your training set consists of the points $(2,2),(4,4),(2,1)$ (where the first entry is the feature, and the second entry is the value to be predicted).

You are going to learn w and b, and use the predictor $y = wx + b$ (note that we don't need a transpose on w, because it is actually a single number, not a vector like usual).

You are deciding on a regularization strategy. You try each of the following regularizers:

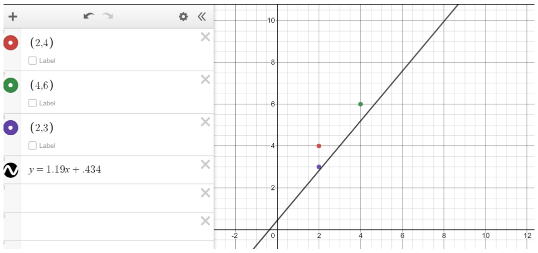
(a) no regularization

(b) $w^2 + b^2$

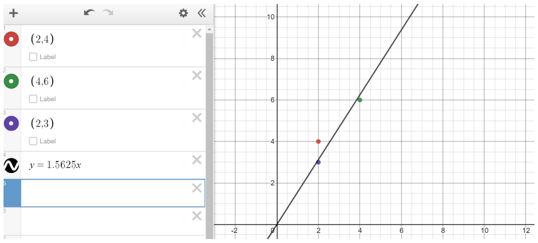
(c) $5(w^2 + b^2)$

(d) $|w| + |b|$

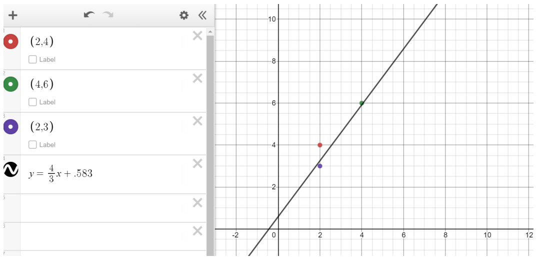
You solve all four regression problems and get the following set of predictors:


(2,4)
□ Label
(4,6)
□ Label
(2,3)
□ Label
$y = 1.19x + .434$

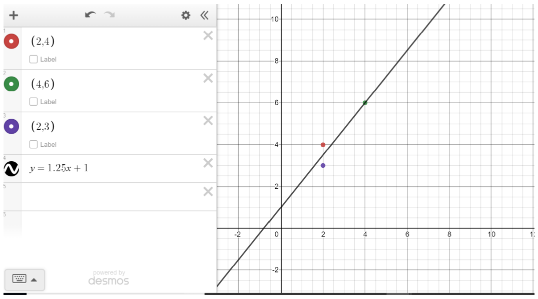
(I) $w = 1.190, b = 0.434$


(2,4)
□ Label
(4,6)
□ Label
(2,3)
□ Label
$y = 1.5625x$

(II) $w = 1.5625, b = 0.000$


(2,4)
□ Label
(4,6)
□ Label
(2,3)
□ Label
$y = \frac{4}{3}x + .583$

(III)$w = 1.333, b = 0.583$


(2,4)
□ Label
(4,6)
□ Label
(2,3)
□ Label
$y = 1.25x + 1$

(IV) $w = 1.250, b = 1.000$

Answer the following questions:

(a) Match each of your regularization strategies to the predictor it created.

Solution:

We'll use process of elimination to match these up. The easiest to identify is L1 regularization. We expect L1 to lead to sparse solutions, and there is one sparse image: image II.

The remaining three strategies can be thought of as all doing L2 regularization, with λ being 0, 1, or 5. We can order the predictors in terms of how much of a regularization penalty they incur to choose these:

image IV has the largest parameters, so involves no regularization.

image III is in the middle, so corresponds to $\lambda = 1$ L2-regularization.

image I has the smallest parameters, so we expect it to correspond to strategy 3 ($\lambda = 5$, L2-regularization).

(b) Usually we do not regularize by b. Explain why this is usually not a good idea.

Solution:

b is just an offset – it corresponds to where the mean of our data is. The goal of regularization is to decide keep our model simple (by either ensuring our parameters don't get too large or by selecting the features we think are more important), but that goal is not met when we force our predictor not to go through the mean of our data.

7.2. Midterm Redux: Closed Form

We will revisit the midterm problem of deriving a closed form for least squares when the errors for each sample have different (known) variances. Recall from the midterm that we are trying to minimize the negative log likelihood function of

$$\sum_{i=1}^{n} \frac{(y_i - x_i^T w)^2}{2\sigma_i^2}$$

where $x_i, w \in \mathbb{R}^{d\times 1}$ and $\mathbf{X} \in \mathbb{R}^{n\times d}$, $\mathbf{Y} \in \mathbb{R}^{n\times 1}$

Write the gradient of this function with respect to w. **Solution:**

$$\nabla_w = -\sum_{i=1}^{n} \frac{x_i(y_i - x_i^T w)}{\sigma_i^2}$$

Find a closed form of the best w (you will need to convert to matrix form to do this easily). **Solution:**

Now to convert this intro matrix form, let us try to think in numpy land. Since we know that ∇w will be formed from the summations of each x_i with its corresponding $\frac{(y_i - x_i^T w)}{\sigma_i^2}$, therefore, we can try to write that out first. Note that we can denote that as $(\mathbf{Y} - \mathbf{X}w)$, where each element of the resulting vector corresponds to $(y_i - x_i^T w)$. To get each element into $\frac{(y_i - x_i^T w)}{\sigma_i^2}$, we can then do $(\mathbf{Y} - \mathbf{X}w)^T S^{-1}$, now each element in the vector corresponds to $\frac{(y_i - x_i^T w)}{\sigma_i^2}$. Recall that ∇w is formed by multipying each x_i with the corresponding $\frac{(y_i - x_i^T w)}{\sigma_i^2}$ and summing them up together. Therefore, we need a way to multiply each element in the vector $(\mathbf{Y} - \mathbf{X}w)^T S^{-1}$ with the corresponding x_i, and it turns out we can do that by $(\mathbf{Y} - \mathbf{X}w)^T S^{-1} X$. However this resulting vector doesn't fit the same shape as the ∇w we want, therefore we can simply transpose it. $-((\mathbf{Y} - \mathbf{X}w)^T S^{-1} X)^T = -\mathbf{X}^T S^{-1}(\mathbf{Y} - \mathbf{X}w) = 0$. Now we can simply solve by algebra,

$$\begin{aligned}
-\mathbf{X}^T S^{-1}(\mathbf{Y} - \mathbf{X}w) &= 0 \\
-\mathbf{X}^T S^{-1}\mathbf{Y} + \mathbf{X}^T S^{-1}\mathbf{X}w &= 0 \\
\mathbf{X}^T S^{-1}\mathbf{X}w &= \mathbf{X}^T S^{-1}\mathbf{Y} \\
\hat{w} &= (\mathbf{X}^T S^{-1}\mathbf{X})^{-1}\mathbf{X}^T S^{-1}\mathbf{Y}
\end{aligned}$$

7.3. Alternative Noise Model

Recall that if we assume Gaussian noise on the labels then the MLE of our estimator w is one that minimizes the residual sum of squares $\sum_{i=1}^{n}(y_i - x_i^T w)^2$.
Instead, let's model the noise as a Laplace distribution. The PDF of Laplace(μ, b) is

$$P(y|\mu, b) = \frac{1}{2b}\exp\left(-\frac{|y-\mu|}{b}\right)$$

(a) Write the likelihood function for the dataset given training data $\{(x_i, y_i)\}_{i=1}^{n}$

Solution:

$$\prod_{i=1}^{n} P(y_i|x_i) = \prod_{i=1}^{n} \frac{1}{2b}\exp\left(-\frac{|y_i - w^T x_i|}{b}\right)$$

(b) Write the negative log-likelihood and show the MLE is the w that minimizes the *sum of absolute residuals*

$$\sum_{i=1}^{n} |y_i - x_i^T w|$$

Solution:

The negative log likelihood can be written as:

$$-\log\left(\prod_{i=1}^{n}\frac{1}{2b}\exp\left(-\frac{|y_i - w^Tx_i|}{b}\right)\right) = -\sum_{i=1}^{n}\log\left(\frac{1}{2b}\exp\left(-\frac{|y_i - w^Tx_i|}{b}\right)\right)$$
$$= n\log(2b) + \frac{1}{b}\sum_{i=1}^{n}|y_i - w^Tx_i|$$

Dropping constants, minimizing the negative log likelihood is equivilent to minimizing $\sum_{i=1}^{n}|y_i - w^Tx_i|$.

(c) Write the gradient descent update rule for minimizing the sum of absolute residuals.

Solution:

The gradient of the nll function can be taken as:

$$\nabla_w \sum_{i=1}^{n}|y_i - w^Tx_i| = \sum_{i=1}^{n}\begin{cases}-x_i & y_i - w^Tx_i \geq 0\\ x_i & y_i - w^Tx_i < 0\end{cases}$$

so the gradient descent update rule with step size η is

$$w \leftarrow w - \eta\sum_{i=1}^{n}\begin{cases}-x_i & y_i - w^Tx_i \geq 0\\ x_i & y_i - w^Tx_i < 0\end{cases}$$

(d) Why might we want to use this noise model instead of Gaussian noise? **Solution:**

The sum of absolute residuals is less sensitive to outliers than the residual sum of squares.

8. SVMs

This problem relies on a very specific definition of "support vector" that we have not discussed in class, so you do not need to worry about it.

Consider the dataset consisting of 7 data points, 4 with positive labels $\{0, 1, 2, 3\}$, and 3 with negative labels $\{-3, -2, -1\}$. Suppose we want to learn a linear SVM with slack variables for this dataset. Recall we can formalize this as a constrained optimization problem:

$$\min_{w,b,\xi} ||w||^2 + C\sum_i \xi_i$$
$$\text{subject to}$$
$$y_i(w \cdot x_i + b) \geq 1 - \xi_i \;\forall i$$
$$\xi_i \geq 0 \;\forall i$$

where C is a regularization parameter that balances the size of the margin (smaller $||w||^2$) vs. the violation of the margin (smaller $\sum_i \xi_i$).

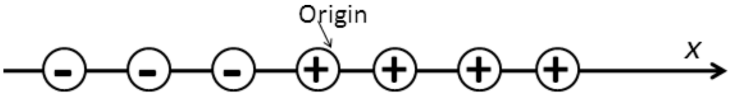


Figure 1: Dataset

~~Also recall that a support vector is a datapoint that lies on a margin.~~

(a) If $C = 0$, which means we only care about the size of the margin, how many support vectors do we have?
Solution:

7, if we just want to minimize $||w||^2$, then we will vary b and ξ_i so all of the constraints are satisfied. As $||w|| \to 0$, we only have to satisfy $y_i b \geq 1 - \xi_i$ and $\xi_i \geq 0 \; \forall i$. These constraints can all be satisfied with $\xi_i \in \{0, 2\}$ and $b = 1$, letting all of the vectors be support vectors.

(b) If $C \to \infty$, which means we only care about the violation of the margin, how many support vectors do we have? **Solution:**

2, if $C \to \infty$, then we just care about minimizing $\sum_i \xi_i$. That is done when we have only one margin between the two classes at -0.5 with two defining support vectors -1 and 0.

9. Optimization Basics

(a) Why do we use the word "convex" to refer to both convex sets and convex functions (i.e. why are they related?)

Solution:

Two reasons: First a function can only be convex if its domain is a convex set (one of the ways we defined "convex function" was that if we draw a secant line between two points on the function, the line is always above the function. That process doesn't make sense if the function isn't convex.)

Second there are close connections between the two objects, for example the set of all points above a convex function are a convex set.

(b) We've talked about convex sets of points in $\mathbb{R}^n$, but we can talk about convex subsets of other spaces. Instead of considering subsets of $\mathbb{R}^n$, let's consider subsets of $\mathbb{R}^{n \times n}$. I.e. instead of sets of points, we now have sets of matrices. In this ambient space, is the set of symmetric matrices a convex set?

Solution:

It is! Let's first understand what we mean. By definition, we need to be able to take any two symmetric matrices, A, B, and ensure that any "point" (i.e. matrix) on the "line" (i.e. convex combination) connecting them is still symmetric. Let's see if that's true.

Formally what we need to show is that for any $\lambda \in [0, 1]$: $C = \lambda A + (1 - \lambda)B$ is still symmetric.

What's entry $C_{i,j}$? $\lambda A_{i,j} + (1 - \lambda)B_{i,j}$ and $C_{j,i}$? $\lambda A_{j,i} + (1 - \lambda)B_{j,i} = \lambda A_{i,j} + (1 - \lambda)B_{i,j} = C_{i,j}$ so it is symmetric! Thus the set is convex.

(c) For each of the following, either give a one-sentence explanation for why the statement is true, or a counter-example.

- If f and g are convex functions, then $f+g$ is a convex function.

  Solution:

  This is true, you proved it on your homework. An easy proof is to notice that the derivative of $f+g$ is $f'+g'$ so the derivative is still below the function.

- If $f+g$ is a convex function, then f and g are both convex functions.

  Solution:

  This is false, for example $f = x^2 - x^3$ is not convex, but for $g(x) = x^3$, $f+g = x^2$ is convex.

- If A is a convex set then $A \cap B$ is convex for any set B.

  Solution:

  This is false in general – take B to be any non-convex subset of A, then $A \cap B = B$, which is not convex by assumption. On the other hand, if B is convex, then $A \cap B$ is convex (we showed this in section).

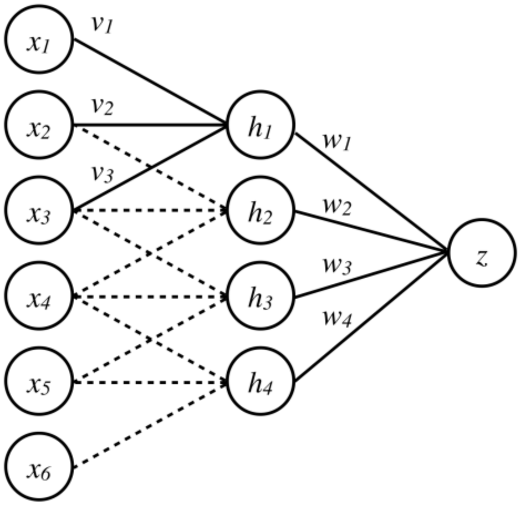
- If A and B are convex and $A \cap B \neq \varnothing$ then $A \cup B$ is a convex set.

  Solution:

  This is false, for example, take two circles of radius 1, with centers at $(0,0)$ and $(1/2,0)$. They definitely overlap, but (for example) the line connecting the points $(0,1)$ and $(1/2,1)$ leaves the union of the circles.

10. Neural Networks

Consider the convolutional neural network architecture:



In the first layer, we have a one-dimensional convolution with a single filter of size 3 such that $h_i = \sigma(\sum_{j=1}^{3} v_j x_{i+j-1})$. The second layer is fully connected, such that $z = \sum_{i=1}^{4} w_i h_i$. The hidden units' activation function $\sigma(x)$ is the sigmoid function. The output unit is linear. We perform gradient descent on the loss function $L = (y-z)^2$ where y is the training label for x.

(a) How many parameters does this network have? Recall that convolutional layers share weights. There are no bias terms.

Solution:

There are 7 parameters, 3 in the first layer and 4 in the second.

(b) Why might we use convolutions?

Solution:

We use convolutions to capture local structure in our data. Convolutions also allow us to use fewer parameters as they share weights across the entire input.

Consider the neural network defined as $g(x) = W_2 f(W_1 x)$. Where f is an activation function, $W_1 \in \mathbb{R}^{h \times d}$, $W_2 \in \mathbb{R}^{1 \times h}$. Let us consider minimizing the square loss for a regression task.

(c) Let $f(x) = \tanh(x)$. Is the loss function convex in the model parameters?

Solution:

No, the nonlinearities make it so the loss function is non-convex. This means that we are not guaranteed to find the global minimum, or a minimum at all (we could end up at a saddle point).

(d) Let $f(x) = x$. What types of functions can we represent using this model?

Solution:

If $f(x) = x$, then we can simplify our model to $g(x) = Ax$ where $A = W_2 W_1$. This means (no matter how big we make h) we can only represent linear functions.

11. Multiple Choice T/F and Short Answer

(a) Let X have the following Singular Value Decomposition: $X = USV^T$. For each of the follow matrices, state whether or not the columns of U are the eigenvectors of that matrix.

- $X^T X$

  Solution:

  No. $X^T X = (VSU^T)(USV^T) = VS^2V^T$ so V is the set of eigenvectors, not U.

- $X^T X X^T X$

  Solution:

  No. Repeating the calculation from the previous part, we get that this matrix is VS^4V^T.

- XX^T

  Solution:

$$\begin{aligned} XX^T &= USV^T VS^T U^T \\ &= USS^T U^T \\ &= US^2 U^T \end{aligned}$$

- $XX^T XX^T$

Solution:

$$\begin{aligned} XX^T XX^T &= USV^T (USV^T)^T (USV^T)(USV^T)^T \\ &= USV^T VS^T U^T USV^T VS^T U^T && VV^T = UU^T = I \\ &= USS^T SS^T U^T \\ &= US^4 U^T \end{aligned}$$

(b) Which of the following are reasons why PCA useful for doing preprocessing steps on data? Please provide justification
a) Reduce overfitting of data
b) Increase computational efficiency during training
c) Extract more features/information about your training data

Solution:

A and B. For A, we can reduce overfitting of data by extracting the k most important directions in our data, and forcing the algorithm to ignore the noise in the data. When we do PCA, our dimensions essentially reduces thereby increasing our training efficiency. PCA does not create extra features in our data. It only identifies important linear combinations of existing ones.

(c) Let us consider a matrix $X = \sum_{i=1}^{r} s_i u_i v_i^T$. What are the possible values for the rank for this matrix?

Solution:

Any integer from 0 to r (inclusive) is possible. Note that we cannot have rank more than r since there are only r many vectors. The lower bound is 0 if s_i is 0 for all i, and we can get any number in-between by repeating some of the u, v vectors (or setting some s_i to 0).

(d) Which of the following are true statements for k-nearest neighbors as we increase k?
a) The decision boundary becomes smoother
b) Variance decreases
c) As the number of sample points approaches infinity, error rate becomes less than twice the Bayes error for 1-nearest neighbors.
d) Bias decreases

Solution:

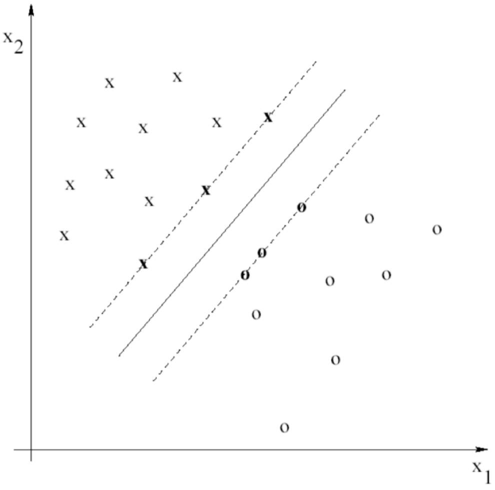
A is true – as k increases, we can “ignore” more and more points that are opposite of those nearby, and thereby smooth out the boundary (As an example, suppose you have a single positive point surrounded by negative points. For 1-NN, we will classify positively close to that single positive point. For 3-NN, we

will classify negatively everywhere around that positive point).
B is true the variance decreases and bias increases. This is easiest to see in extreme cases, for k the number of data points, we classify all points the same (low variance, but very high bias)
C is true (this is just a fact stated in lecture.
D is false (see the explanation in B).

(e) In lecture we have shown that low-rank approximation of matrix is helpful for many reasons. Which of the following are the reason to use a low-rank appoximation of your data matrix?
a) Denoising
b) Discover latent categories in data
c) Filling in unknown values
d) Storage Compression
Solution:

All of the above. We have shown in lecture all of the above uses of SVD. For denoising, we saw that we can reconstruct the data points with reduced noise by taking the first k components principal components (which we can find via SVD). We also saw an example of discovering latent categories in data through the DNA example we saw in Lecture 13. We also walked through an example of filling in unknown values, i.e. matrix completions in lecture (and you're doing it on HW4). Finally, we talked about eigenface in lecture, which is an example of compression (and you compressed data via SVD earlier in this handout).

(f) What is the leave one out cross-validation error when using an SVM for classification on the following example? (In this case, error will simply be the number of mistakes.) The SVM decision boundary and margin on either side are shown.



Solution:

0, removing any one point does not change the optimal hyperplane, so the decision boundary will not change and the LOOCV error will remain the same.

(g) What is the minimal number of points we can remove required to change the decision boundary?
Solution:

3 points. Since there are 3 data points that lie on each boundary, removing one point or two points

leaves two or one points, respectively that still define the margin. If we remove three points, the decision boundary changes.

(h) There are several ways to formulate the hard margin SVM. Consider a formulation where we try to directly maximize the margin γ. The training samples are $\mathbf{x}_1, \dots, \mathbf{x}_n$ and their labels are $y_1, \dots, y_n$. Circle all the constraints we should impose to get a correct SVM. Maximize γ subject to

(i) $y_i \mathbf{x}_i^T \mathbf{w} \leq \gamma \quad \forall i$

(ii) $y_i \mathbf{x}_i^T \mathbf{w} \geq \gamma \quad \forall i$

(iii) $||\mathbf{w}|| \geq 1$

(iv) $||\mathbf{w}|| = 1$

Solution:

We should maximize γ subject to (ii) and (iv).

12. Matchup

Here's a sample of some of the techniques we've seen in this course:

(a) Bootstrap

(b) EM Algorithm

(c) PCA/SVD

(d) Kernelization

(e) cross-validation

For each of the following scenarios, choose the most appropriate of the above techniques.

(a) You have a large number of features, and want to find the most important linear combinations of your features.

Solution:

We've seen a few ways to find important features, but **PCA/SVD** is the most appropriate one on this list, since we care about linear combinations of the features.

(b) You think your linear model would work best if you applied a feature map, but it would be infeasible to calculate all the entries of the map.

Solution:

When we only care about dot-products of the feature maps, we can use **Kernelization** to avoid calculating the full feature map.

(c) You're worried your model has too much variance, and want to estimate how much your predictions would change if you got a different training set.

Solution:

Bootstrap is a way of estimating a confidence interval around your predictions as a result of the randomness of the training set. This is exactly trying to understand the variance of your model.

(d) Your model uses α and β. The best α depends on the best β and the best β depends on the best α. You can't find a closed form that lets you find both α and β simultaneously, but given some $\hat{\beta}$ you can find the best $\hat{\alpha}$ and vice-versa. How do we find α, β?

Solution:

In an instance such as this, we can find good α and β using the **EM Algorithm**, though we usually don't have a guarantee of finding the best ones.

(e) Your model uses α and β. You aren't interested in finding the best α for the training set, but in finding one that will allow for better generalization. How do we find α?

Solution:

α is a hyperparameter. We would really like to choose an α by looking at what generalizes the best to new data, but we can't use our test set to find the best α. The closest we can get is **Cross-validation**.

13. Midterm Matchup Redux

This problem is copy-pasted from the midterm. The solutions will have a detailed breakdown of this question; you may want to try it again to see what you remember from the first half of the course.

Suppose you are doing regression and discover that your learning algorithm is not working as well as you think it should be. To fix the problem, some things you could try are the following:

(a) Get a hold of a larger training set.

(b) Add additional functions to your hypothesis class (e.g., switch from linear functions to cubic polynomials.)

(c) Try using a simpler hypothesis class

(d) Change the feature map

(e) Try changing the optimization algorithm used.

(f) Try L2 regularization

(g) Try L1 regularization

For each of the following specific issues, which of the above approaches has the potential to help?

For each of these problems, if there are two or more strategies that will help, list the two that you think are most likely to help. If there is only one strategy that will help, list it. If there are none, write "none".

To make the following discussion concrete, let's consider the bias-variance decomposition for squared loss. Specifically, consider a distribution P_{XY} over $\mathbb{R}^d \times \mathbb{R}$ and suppose we have a dataset $\mathcal{D}$ where each example pair $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$ is drawn iid from P_{XY}. Let $\mathbb{E}_{XY}[\cdot]$ denote the expectation with respect to a draw from P_{XY} and let $\mathbb{E}_{\mathcal{D}}[\cdot]$ denote the expectation with respect to the random draw of the dataset (i.e., $|\mathcal{D}|$ iid draws from P_{XY}). Assume we have access to a learning algorithm that takes a dataset $\mathcal{D}$ as input and outputs a function $\widehat{f}_{\mathcal{D}}$ that predicts Y from X. The learning algorithm is implicitly minimizing training error over a hypothesis class $\mathcal{F}$ of functions. Define $\eta(x) = \mathbb{E}_{Y|X}[Y|X=x]$. Then the bias-variance decomposition at any $x \in \mathbb{R}^d$ is equal to

$$\mathbb{E}_{Y|X,\mathcal{D}}[(Y-\widehat{f}_{\mathcal{D}}(x))^2|X=x] = \underbrace{\mathbb{E}_{Y|X}[(Y-\eta(x))^2|X=x]}_{\text{irrudicible error}} + \underbrace{(\eta(x)-\mathbb{E}_{\mathcal{D}}[\widehat{f}_{\mathcal{D}}(x)])^2}_{\text{bias-squared}} + \underbrace{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}'}[\widehat{f}_{\mathcal{D}'}(x)]-\widehat{f}_{\mathcal{D}}(x))^2]}_{\text{variance}}$$

(i) high bias

Solution:

[Accepted Answers: (b), (d)]

The bias term describes the difference between the best function one could hope for $\eta(x)$ and the expected function that will be learned by the learning algorithm $\mathbb{E}_{\mathcal{D}}[\widehat{f}_{\mathcal{D}}(x)]$ using the given training data $\mathcal{D}$. We often think of the bias term as describing the complexity of the hypothesis class, and in general, how well the hypothesis class can fit the provided training data. For example, the bias of 1-nearest-neighbor, a large depth tree, or large neural network is very low because they can perfectly learn just about any training set, and because the training set is an iid draw from P_{XY}, in expectation, the predictions should be close to $\eta(x)$. On the other hand, if the function class is not very complex like linear regression or a shallow tree, the amount of functions these classes can even learn is very constrained. Therefore the learning algorithm will output a function that will not be able to fit the training data well, and will not be able to match $\eta(x)$ in expectation. Increasing the size of your function class (b) or transforming your features (d) so that perhaps even a simple function class is better matched to $\eta(x)$ are the most appropriate answers. One could make a case that (a) getting a larger training set could reduce bias in edge cases (e.g., in linear regression, non-zero bias is inevitable until the number of examples in the dataset is at least the dimension) but we did not accept this answer as the problem only requested two answers, and (b,d) are far more appropriate to the spirit of reducing bias.

(ii) high variance

Solution:

[Accepted Answers:(a), (c), (d), (f), (g), (e)]

Variance describes how much the learned function is expected to deviate from dataset to dataset. That is, given two iid datasets, if the functions learned on each of them are wildly different, then the variance is very high. For instance, when we considered linear regression with high degree polynomial feature maps, that is a very high variance class since the functions could accurately fit just about any training set, but the functions learned deviated a lot from dataset to dataset. One nearest neighbor is also a high variance class as the particular examples seen directly result in a different function. Linear regression in the natural basis has d parameters and without regularization the variance scales like d/n if the dataset size is n, explaining (a). Adding more flexibility to your hypothesis class (b) only increases the opportunity for more variance, whereas using a simpler hypothesis class (c) only decreases the variance. Using a more appropriate feature map (d) may allow the algorithm to consider fewer or less noisy features, thereby decreasing variance. In linear regression when $n < d$, it is known that gradient methods will converge to a solution with small ℓ_2 norm, which is not a guarantee of other optimization methods; small ℓ_2 norm solutions are associated with smaller variance (e). Adding regularization decreases variance (f,g).

(iii) high irreducible noise

Solution:

[Accepted Answers: nothing]

The dataset, hypothesis class, and learning algorithm have no role in the irreducible noise term. Therefore changing these has no effect.

(iv) overfitting

Solution:

[Accepted Answers:(a), (c), (d), (e), (f), (g)]

Overfitting occurs when the the learning algorithm fits the training set very well but poorly generalizes to new, unseen test data. By reducing variance, perhaps at the expense of increasing bias, one can reduce overfitting. Thus, the same answers for reducing high variance apply.

(v) underfitting

Solution:

[Accepted Answers:(b), (d)]

Underfitting occurs when the learning algorithm has a poor fit to the training data. In otherwords, high bias, so the same answers for reducing bias apply.

(vi) difficulty understanding which features are most important for prediction.

Solution:

[Accepted Answers:(c), (g), (d), (e)]

Adding more hypotheses that have simple interpretations to your hypothesis class could potentially fit the data better, but typically increasing complexity only hurts interpretability because it potentially adds more ways to fit the data, which might actually mislead you, so (b) was not one of the most appropriate answers. A simpler hypothesis class (c) may reduce the number of possible explanations if the data is fit well, perhaps becoming more interpretable. Crazy non-linear features may allow you to fit the data well, but simple features with physical meaning may be easier to interpret (d). L1 regularization (g) encourages sparsity which can be much more interpretable than many small coefficients. In high dimensions and when $n < d$, many solutions may minimize the training error–the learning algorithm may result in a more or less interpretable model (e.g., the shooting algorithm versus gradient descent for solving lasso: the former will often result in a sparser solution).