Name:
NetID:

COS 324 – Practice Final

Problem 1 (???pts)

There are many types of donuts. Jad and Farhan both like some donuts and not others. Suppose Farhan has a donut-eating utility function $U_F(\text{donut})$ and Jad has a donut-eating utility function $U_J(\text{donut})$. Say the relationship between their utilities is

$$U_F(\text{donut}) = 7 \times (U_J(\text{donut}))^2 - 42\,.$$

Explain whether or not Farhan and Jad have the same donut-eating preferences.

They do not have the same preferences. Consider a donut that Jad assigns a large positive utility u and a another donut that Jad strongly dislikes and assigns $-u$. Farhan is nevertheless ambivalent between these donuts, as the square eliminates the sign of Jad's utility.

Problem 2 (???pts)

One way to initialize Lloyd's algorithm for K-Means is to randomly select some of the data to be the first cluster centers. The easiest version of this would pick uniformly from among the data. K-Means++ biases this distribution so that it is not uniform. Explain in words how the distribution is non-uniform and why it should lead to better initializations.

The K-Means++ algorithm iteratively adds cluster centers, drawing them from a distribution over the data. This distribution is proportional to the squared distance of each datum from its nearest cluster center. Thus K-Means++ tends to favor points that are distant from the existing centers and produce a more diverse set of centers.

Problem 3 (???pts)

Standardizing data helps ensure that distances makes sense and that the different properties of the items are balanced. Give an example of a kind of data for which standardization might be necessary to get good results from K-Means clustering.

Any example with features that have different units:

- Clustering galaxies by size and brightness.
- Clustering fruit by color and weight.
- Clustering houses by square footage and price.

Problem 4 (???pts)

K-Medoids clustering is similar to K-Means, except that it requires the cluster centers to be data examples. Describe a situation in which this is desirable or necessary.

Sometimes we only have distances between points, and averages of points are not sensible or available. In such cases, we cannot describe a cluster by a mean of data and instead describe it by an "exemplar".

Problem 5 (???pts)

Suppose we have a grid-world where each state is represented as a point in $\{(x, y) : x \in \{0, \ldots, 4\}, y \in \{0, \ldots, 4\}\}$. Suppose you have a robot that starts in the lower left corner and is given a goal point that is needs to reach. At each state, the robot can move one point to the left, right, up, or down, and each action has a 90% chance of success (otherwise you stay at the current point).

A. What is the size of the state space in the resulting MDP?

B. Suppose we want to minimize the length of path to the goal, but we have no preference for which path the robot should take (and all points are reachable). What rewards could we assign to each state so that we recover a shortest path to the goal as our optimal policy?

A. Our grid is composed of each point in a 5×5 grid, so we have 25 possible states in the state-space.

B. We can set the reward of each state, new state pair (that does not end with the goal node) to be the same negative value. Then, when we try to maximize our reward we will choose the shortest path, and since each transition has the same reward, we do not preferentially choose any paths with the same length.

Problem 6 (???pts)

Consider the following generative process for generating $2N$ data points along the circumference of a unit circle. (Recall that a unit circle satisfies $x^2 + y^2 = 1$.)

- Draw a point x_n along the x-axis $\sim$ Unif$(-1, 1)$
- Given x_n, set $y_{n_1} = \sqrt{1 - x^2}$ and $y_{n_2} = -\sqrt{1 - x^2}$, and add both (x_n, y_{n_1}) and (x_n, y_{n_2}) to the dataset.

Suppose we generate N points x_n in this way, giving us a total of $2N$ datapoints.

A. If N is very large, what do you expect $\bar{\boldsymbol{x}}$, the empirical mean of the data, to (approximately) be?

B. Using the approximation of $\bar{\boldsymbol{x}}$ from A, write out the sample covariance matrix $\boldsymbol{S}$ of the data in terms of the x_n.

C. What (numerical) values will the entries of $\boldsymbol{S}$ approach as N gets large? (Hint: You may want to use a rearrangement of the identity $\mathbb{V}[x] = \mathbb{E}[x^2] - (\mathbb{E}[x])^2$, and the fact that $x_n \sim$ Unif$(-1, 1)$.)

D. If we wanted to use PCA to reduce the datapoints to a single dimension, given your answer to part (c), what will the first principal component (i.e., eigenvector) and its corresponding eigenvalue approximately be?

A. By symmetry, we can see that $\bar{\boldsymbol{x}} \approx \mathbf{0}$, since the x_n are uniformly distributed around 0 and since there is a negative y value for every positive y value.

B. The sample covariance matrix is 2×2 and is computed as $\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^\mathsf{T}$. The diagonal entries are simply the sample variances:

$$S_{1,1} = \frac{1}{2N}\sum_{n=1}^{N} 2x_n^2 = \frac{1}{N}\sum_{n=1}^{N} x_n^2$$

$$S_{2,2} = \frac{1}{2N}\sum_{n=1}^{N}(\sqrt{1 - x_n^2})^2 + (-\sqrt{1 - x_n^2})^2 = \frac{1}{2N}\sum_{n=1}^{N} 2(1 - x_n^2) = \frac{1}{N}\sum_{n=1}^{N}(1 - x_n)^2$$

Then the off-diagonal term is the product between the two, but that turns out to cancel:

$$S_{1,2} = S_{2,1} = \frac{1}{2N}\sum_{n=1}^{N} x_n\sqrt{1 - x_n^2} - x_n\sqrt{1 - x_n^2} = 0$$

So:

$$\boldsymbol{S} = \begin{bmatrix} \frac{1}{N}\sum_{n=1}^{N} x_n^2 & 0 \\ 0 & \frac{1}{N}\sum_{n=1}^{N}(1 - x_n^2) \end{bmatrix}$$

C. The variance of x_n is $\frac{1}{2}\int_{-1}^{1} x^2\, dx = \frac{1}{6}x^3|_{-1}^{1} = \frac{1}{3}$. You can rearrange the term $S_{2,2}$ to save some work in computing the variance of the second term as N gets large:

$$S_{2,2} = \frac{1}{N}\sum_{n=1}^{N}(1 - x_n^2) = \frac{1}{N}\sum_{n=1}^{N} 1 - \frac{1}{N}\sum_{n=1}^{N} x_n^2 = 1 - \frac{1}{3} = \frac{2}{3}$$

So:

$$\boldsymbol{S} = \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & \frac{2}{3} \end{bmatrix}$$

D. Since the matrix is diagonal, the eigenvectors are the standard basis and the eigenvalues are the diagonal entries. So the principal eigenvector is $(0, 1)$ and its eigenvalue is $\frac{2}{3}$.

Problem 7 (???pts)

Your friend Bob has founded a startup based on his new top-secret unsupervised learning method. After signing an NDA, Bob reveals to you that his new idea is something he calls "double-PCA", in which he performs dimensionality reduction with PCA from N to $J < N$ dimensions, and then runs PCA *again* on the projected data to get it down to $K < J$ dimensions. Bob thinks he's going to be rich, but you're not so sure. Explain why double-PCA doesn't gain anything over regular PCA.

Without loss of generality, we could assume the data is already centered, i.e. the mean of the data points is the zero vector (otherwise we just subtract the mean from each data point).

Now we arrange all n data points into a $n \times d$ data matrix X. Suppose X has a SVD as

$$X = \sum_{i=1}^{d} \sigma_i u_i v_i^T$$

where $\sigma_1 \geq \sigma_2 \geq \cdots$.

Then the first round of reducing the data into a J dimension space would give the new data matrix

$$X' = [\sigma_1 u_1 \ \sigma_2 u_2 \ \cdots \ \sigma_J u_J] \in R^{n \times J}$$

(just take the top J left singular vectors scaled by the corresponding singular values if we want to use PCA to reduce the data to J dimension.)

We could then rewrite X' as

$$X' = \sum_{i=1}^{J} \sigma_i u_i e_i^T$$

where e_i is the ith standard basis. And it is easy to see that this is already in the form of SVD.

Therefore, the second round of reducing the data into K dimension would give the new data matrix

$$X'' = [\sigma_1 u_1 \ \sigma_2 u_2 \ \cdots \ \sigma_K u_K] \in R^{n \times K}$$

which is the same if we directly reduce the original X into K dimension.

Problem 8 (???pts)

In K-means we minimize the squared Euclidean (L^2 distance) from each datum to its prototype. Imagine if we wanted to construct a similar algorithm, but using L^1 distance instead. Recall that the L^1 norm for a vector in $\mathbb{R}^D$ is

$$||\boldsymbol{x}||_1 = \sum_{d=1}^{D} |x_d| \,.$$

Explain what would be different about a Lloyd-type algorithm to solve this problem.

The K-means algorithm alternately performs the following two steps: 1) assigning each data point to the nearest cluster; 2) update the new center for each cluster.

For step 1), now we are using the L^1 distance, we should assign each data point to the nearest cluster in terms of the L^1 distance instead of the L^2 distance. Therefore, suppose c_i is the center for cluster i. Then data point x would be assigned to cluster k that minimizes $\|x - c_k\|_1$.

For step 2), suppose $x_1, \ldots, x_m$ are assigned to cluster i in this round. Then we should update the center c_i for cluster i such that c_i minimizes

$$L = \|x_1 - c_i\|_1 + \cdots + \|x_m - c_i\|_1$$

And this could be written as

$$L = \sum_{j=1}^{m} \sum_{t=1}^{d} |x_{j,t} - c_{i,t}| = \sum_{t=1}^{d} \sum_{j=1}^{m} |x_{j,t} - c_{i,t}|$$

Therefore, minimizing L could be decomposed into d separate 1-dimensional problems where in each of them we need to find $c_{i,t}$ that minimizes $\sum_{j=1}^{m} |x_{j,t} - c_{i,t}|$, which should be the median of $x_{1,t}, \ldots, x_{m,t}$.

Problem 9 (???pts , Conjugate Prior)

Let $X_1, X_2, ..., X_N$ be N independent random variables with probability density $\mathcal{N}(\theta, \sigma^2)$, i.e., normal/Gaussian distributed with mean θ and variance σ^2. In this case, we the variance σ^2 is known and we want to estimate θ from the X_n. We further assume that we have a prior $\theta \sim N(\mu, \nu)$.

Show that the posterior distribution of θ is also Gaussian, compute its parameters, and show that the new variance is smaller than ν.

$$P(\theta \mid X_1, X_2, ..., X_N) \propto \exp\left[-\frac{\sum_{n=1}^{N}(X_n - \theta)^2}{2\sigma^2} - \frac{(\theta - \mu)^2}{2\nu}\right]$$

After expanding and rearranging by completing the square, we find that:

$$\theta \sim N(\mu', \sigma'^2)$$

with:

$$\sigma'^2 = \frac{1}{\frac{1}{\nu} + \frac{N}{\sigma^2}}$$

$$\mu' = \frac{1}{\sigma'^2}\left[\frac{\sum_{n=1}^{N} X_n}{\sigma^2} + \frac{\mu}{\nu}\right]$$

We have: $\frac{1}{\tilde{\sigma}^2} + \frac{N}{\sigma^2} > \frac{1}{\tilde{\sigma}^2}$ so $\sigma'^2 < \nu$.

Problem 10 (???pts , Maximum Entropy)

Let P be a probability distribution on the positive integers $\mathbb{N}^+$. The *entropy* $H(P)$ of such a probability distribution is defined by:

$$H(P) = -\sum_{i=1}^{+\infty} P(i)\log(P(i))$$

(We typically take $0 \log 0 = 0$ when computing entropies.) Out of all distributions P on the positive integers, with mean $\lambda > 1$, which one has the highest entropy ?

This is a constrained optimization problem, we want to maximize $H(P) = -\sum_{i=1}^{+\infty} P(i)\log(P(i))$ but we have two constraints:
1- $\sum_{i=1}^{+\infty} P(i) = 1$
2- $\sum_{i=1}^{+\infty} P(i) \times i = \lambda$
The Lagrangian for this problem is then :

$$L = -\sum_{i=1}^{+\infty} P(i)\log(P(i)) - \mu(\sum_{i=1}^{+\infty} P(i) - 1) - \nu(\sum_{i=1}^{+\infty} P(i) \times i - \lambda)$$

Differentiating with respect to $P(i)$ we find:

$$-1 - \log(P(i)) - \mu - i \times \nu = 0$$

Consequently:

$$P(i) \propto e^{-\nu i} = (e^{-\nu})^i$$

P is therefore a geometric distribution of mean λ,

$$P(i) = \frac{1}{\lambda}(1 - \frac{1}{\lambda})^{i-1}$$

Problem 11 (??? pts, Conditional Density Estimation)

You are given a set of images $\{\boldsymbol{x}_n\}_{n=1}^N$ and corresponding labels $\{y_n\}_{n=1}^N$. For each image $\boldsymbol{x}_n$, let $\{x_{nj}\}_{j=1}^J$ be the color of the jth pixel in image n and takes on a value from 0 to 100. y_n corresponds to one of K image categories. For example, we might have a set of images of animals and we want to identify what animal (the label) is in which image. We tackle this problem using class-conditional density estimation. That is, first the label $y_n \in \{1, \ldots, K\}$ is drawn from a discrete distribution $\Pr(y_n)$ with parameters $\boldsymbol{\pi}$ and then the image pixels are drawn from a distribution specific to that label. In this case, we'll say that each of the K label-specific distributions over pixels are Gaussian and that they all share the same diagonal covariance. That is, for pixel j in image n, $\Pr(y_n)$ comes from a discrete distribution and then $\Pr(x_{nj} \mid y_n)$ is Gaussian with mean μ_{j,y_n} and variance σ_j^2.

A. Write the likelihood.

B. What is the total dimensionality of the parameter space to be learned?

C. Are there problems with the model's assumptions?

A. Likelihood

$$\Pr(\{\boldsymbol{x}_n, y_n\}_{n=1}^N \mid \boldsymbol{\pi}, \{\{\mu_{j,k}\}_{k=1}^K, \sigma_j^2\}_{j=1}^J) = \prod_{n=1}^N \pi_k^{\delta(k,y_n)} \prod_{j=1}^J \mathcal{N}(x_{nj} \mid \mu_{j,y_n}, \sigma_j^2)$$

It is also fine to set things up for a one-hot coding of y_n.

B. K dimensions in $\boldsymbol{\pi}$ for $\Pr(y_n)$, although reasonable to say $K - 1$ as the parameter is on the simplex. J dimensions for the σ_j^2 and $J \times K$ parameters for the $\mu_{j,k}$

C. The model assumes each image's pixels are conditionally independent given an image's label. This is a poor assumption given the obvious correlation structure with neighboring pixels in an image. Moreover, the Gaussian distribution is a poor model for images.

Problem 12 (??? pts, Softmax Regression)

In multi-class logistic regression, we are typically given data in pairs $\{\boldsymbol{x}_n, y_n\}_{i=1}^N$ where $\boldsymbol{x}_n \in \mathbb{R}^M$ is a feature vector and $y_n \in \{1, ..., K\}$ is the label for 1 of K classes. We are seeking to learn a matrix of parameters $\boldsymbol{W} \in \mathbb{R}^{K \times M}$ where W_{km} is the weight for feature m class k.

A. For datum n, what is the class conditional probability $\Pr(y_n = k \mid \boldsymbol{x}_n, \boldsymbol{W})$?

B. Show part A reduces to binary logistic regression when $K = 2$.

A. Taking $\boldsymbol{w}_k$ to be the kth row of $\boldsymbol{W}$.

$$\Pr(y_n = k \mid \boldsymbol{x}_n, \boldsymbol{W}) = \frac{e^{\boldsymbol{w}_k^T \boldsymbol{x}_n}}{\sum_{k'=1}^K e^{\boldsymbol{w}_{k'}^T \boldsymbol{x}_n}}$$

B. Write the softmax update with $K = 2$.

$$\Pr(y_n = 1 \mid \boldsymbol{x}_n, \boldsymbol{W}) = \frac{e^{\boldsymbol{w}_1^T \boldsymbol{x}_n}}{e^{\boldsymbol{w}_1^T \boldsymbol{x}_n} + e^{\boldsymbol{w}_2^T \boldsymbol{x}_n}}$$
$$\Pr(y_n = 2 \mid \boldsymbol{x}_n, \boldsymbol{W}) = \frac{e^{\boldsymbol{w}_2^T \boldsymbol{x}_n}}{e^{\boldsymbol{w}_1^T \boldsymbol{x}_n} + e^{\boldsymbol{w}_2^T \boldsymbol{x}_n}}$$

$Pr(y = k \mid \boldsymbol{X}\, \boldsymbol{W})$ is overparameterized. We do not need a separate vector of weights for class 1 and class 2. Because the softmax is overparameterized, we can subtract a constant vector ψ from each class-specific weight vector $\boldsymbol{w}_k$, and we will still have the same hypothesis! We can conveniently set our constant to one of the weights $\psi = \boldsymbol{w}_1$ and subtract ψ from each of the two parameters.

$$\Pr(y_n = 1 \mid \boldsymbol{x}_n, \boldsymbol{W}) = \frac{e^{(\boldsymbol{w}_1 - \psi)^T \boldsymbol{x}_n}}{e^{(\boldsymbol{w}_1 - \psi)^T \boldsymbol{x}_n} + e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}} = \frac{e^{(0)^T \boldsymbol{x}_n}}{e^{(0)^T \boldsymbol{x}_n} + e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}} = \frac{1}{1 + e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}}$$
$$\Pr(y_n = 2 \mid \boldsymbol{x}_n, \boldsymbol{W}) = \frac{e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}}{e^{(\boldsymbol{w}_1 - \psi)^T \boldsymbol{x}_n} + e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}} = \frac{e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}}{1 + e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}} = 1 - \frac{1}{1 + e^{(\boldsymbol{w}_2 - \psi)^T \boldsymbol{x}_n}}.$$

Problem 13 (??? pts, Generalizing K-Means)

We can view K-means clustering as the minimization of a "distortion" in terms of squared error. That is, with N data $\{\boldsymbol{x}_n\}_{n=1}^N$ and K cluster centers $\{\boldsymbol{\mu}_k\}_{k=1}^K$, we imagine replacing each datum with a cluster center $\boldsymbol{\mu}_k$ and we have a good clustering when this works well on average across the data. We frame this as a minimization problem in terms of the centers and the binary responsibility vectors $\boldsymbol{r}_n \in \{0,1\}^K$, which one-hot encodings of the cluster assignments. What if instead of squared error $||\boldsymbol{x}_n - \boldsymbol{\mu}_k||_2^2$ we used a squared *Mahalanobis distance* $(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\mathsf{T}\boldsymbol{\Lambda}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)$ to measure distortion, where $\boldsymbol{\Lambda}$ is a square, symmetric, positive definite matrix.

A. Write an objective function that uses this distance measure.

B. Imagine adapting Lloyd's algorithm to this problem. What would the updates of the $\boldsymbol{\mu}_k$ be, for fixed responsibilities?

C. What would the updates for the responsibilities be, with fixed $\boldsymbol{\mu}_k$?

A.

$$J(\{\boldsymbol{r}_n\}_{n=1}^N, \{\boldsymbol{\mu}_k\}_{k=1}^K) = \frac{1}{N}\sum_{n=1}^N\sum_{k=1}^K r_{n,k}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\mathsf{T}\boldsymbol{\Lambda}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)$$

B.

$$\begin{aligned}
\nabla_{\boldsymbol{\mu}_{k'}} J(\{\boldsymbol{r}_n\}_{n=1}^N, \{\boldsymbol{\mu}_k\}_{k=1}^K) &= \nabla_{\boldsymbol{\mu}_{k'}} \frac{1}{N}\sum_{n=1}^N\sum_{k=1}^K r_{n,k}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\mathsf{T}\boldsymbol{\Lambda}(\boldsymbol{x}_n - \boldsymbol{\mu}_k) \\
&= \frac{2}{N}\sum_{n=1}^N r_{n,k'}\boldsymbol{\Lambda}(\boldsymbol{x}_n - \boldsymbol{\mu}_{k'}) = 0 \\
\sum_{n=1}^N r_{n,k'}\boldsymbol{\Lambda}\boldsymbol{x}_n &= \sum_{n=1}^N r_{n,k'}\boldsymbol{\Lambda}\boldsymbol{\mu}_{k'} \\
\frac{\sum_{n=1}^N r_{n,k'}\boldsymbol{\Lambda}\boldsymbol{x}_n}{\sum_{n=1}^N r_{n,k'}} &= \boldsymbol{\Lambda}\boldsymbol{\mu}_{k'} \\
\boldsymbol{\Lambda}^{-1}\frac{\sum_{n=1}^N r_{n,k'}\boldsymbol{\Lambda}\boldsymbol{x}_n}{\sum_{n=1}^N r_{n,k'}} &= \boldsymbol{\mu}_{k'} \\
\frac{\sum_{n=1}^N r_{n,k'}\boldsymbol{x}_n}{\sum_{n=1}^N r_{n,k'}} &= \boldsymbol{\mu}_{k'}
\end{aligned}$$

Doesn't depend on $\boldsymbol{\Lambda}$.
C. Set each one-hot encoded $\boldsymbol{r}_n$ to the minimum over $(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\mathsf{T}\boldsymbol{\Lambda}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)$.