Solutions to CPSC 340 Assignment 5

Submitted by:

Armaan Kaur Bajwa
Student ID: 87921193

Sugun Machipeddy
Student ID: 65753337

Instructions

Rubric: {mechanics:5}

The above points are allocated for following the general homework instructions. In addition to the usual instructions: if you're embedding your answers in a document that also contains the questions, your answers should be in a colour that clearly stands out, such as green or red. This should hopefully make it much easier for the grader to find your answers. To make something green, you can use the LaTeX macro `\gre{my text}`.

Also, **READ THIS**: Like in a2, you'll need to grab the data from the course website. FYI: this happens because I'm using the GitHub API in a fairly silly way, which limits individual files to 1 MB each.

1 MAP Estimation

Rubric: {reasoning:10}

In class, we considered MAP estimation in a regression model where we assumed that:

- The likelihood $p(y_i|x_i, w)$ is a normal distribution with a mean of $w^T x_i$ and a variance of 1.
- The prior for each variable j, $p(w_j)$, is a normal distribution with a mean of zero and a variance of λ^{-1}.

Under these assumptions, we showed that this leads to the standard L2-regularized least squares objective function:

$$f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2.$$

For each of the alternate assumptions below, show how the loss function would change (simplifying as much as possible):

1. We use a zero-mean Laplace prior for each variable with a scale parameter of λ^{-1}, so that

$$p(w_j) = \frac{\lambda}{2}\exp(-\lambda|w_j|).$$

Solution:
$f(w) = \frac{1}{2}\|Xw - y\|^2 - \sum log(\exp(-\lambda|w_j|))$
$f(w) = \frac{1}{2}\|Xw - y\|^2 + \lambda\|w_j\|_1$

2. We use a Laplace likelihood with a mean of $w^T x_i$ and a scale of 1, so that

$$p(y_i|x_i, w) = \frac{1}{2}\exp(-|w^T x_i - y_i|).$$

Solution:
$f(w) = -\sum log(\frac{1}{2}\exp(-|w^T x_i - y_i|)) + \frac{\lambda}{2}\|w\|^2$
$f(w) = \sum |w^T x_i - y_i| + \frac{\lambda}{2}\|w\|^2$
$f(w) = \|Xw - y\|_1 + \frac{\lambda}{2}\|w\|^2$

3. We use a Gaussian likelihood where each datapoint has variance σ^2 instead of 1,

$$p(y_i|x_i, w) = \frac{1}{\sqrt{2\sigma^2\pi}}\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma^2}\right).$$

Solution:
$f(w) = -\sum \log(\frac{1}{\sqrt{2\sigma^2\pi}}\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma^2}\right)) + \frac{\lambda}{2}\|w\|^2$
$f(w) = \sum \frac{(w^T x_i - y_i)^2}{2\sigma^2} + \frac{\lambda}{2}\|w\|^2$
$f(w) = \frac{1}{2\sigma^2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$

4. We use a Gaussian likelihood where each datapoint has its own variance σ_i^2,

$$p(y_i|x_i, w) = \frac{1}{\sqrt{2\sigma_i^2\pi}}\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma_i^2}\right).$$

Solution:
$f(w) = -\sum \log(\frac{1}{\sqrt{2\sigma_i^2\pi}}\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma_i^2}\right)) + \frac{\lambda}{2}\|w\|^2$
$f(w) = \sum \frac{(w^T x_i - y_i)^2}{2\sigma_i^2} + \frac{\lambda}{2}\|w\|^2$
$f(w) = \frac{1}{2}(Xw - y)diag(\sigma_i^2)^{-1}(Xw - y) + \frac{\lambda}{2}\|w\|^2$

2 Principal Component Analysis

2.1 PCA by Hand

Rubric: {reasoning:3}

Consider the following dataset, containing 5 examples with 2 features each:

x_1	x_2
-2	-1
-1	0
0	1
1	2
2	3

Recall that with PCA we usually assume that the PCs are normalized ($\|w\| = 1$), we need to center the data before we apply PCA, and that the direction of the first PC is the one that minimizes the orthogonal distance to all data points.

1. What is the first principal component?
   Solution: $\mu_1 = 0$, $\mu_2 = 1$.
   Hence, we need to center x_2.
   The centered featured are as given below:

x_1	x_2
-2	-2
-1	-2
0	0
1	1
2	2

   As we can see, after being centered, $x_1 = x_2$. So the first principal component will be any 2-d vector with equal elements.

2. What is the (L2-norm) reconstruction error of the point (3,3)? (Show your work.)
   Solution:
   Normalized $W_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$
   Now, $z = (x_1 - \mu_1)w_1 + (x_2 - \mu_2)w_2$

   $z = \frac{3-0}{\sqrt{2}} + \frac{3-1}{\sqrt{2}} = \frac{5}{\sqrt{2}}$

   Also, $\hat{x} = zW_1 + \mu = \frac{5}{\sqrt{2}}\left(\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}\right) + (0, 1)$

   So, $\hat{x} = \left(\frac{5}{2}, \frac{7}{2}\right)$

   Hence, Reconstruction error $= \sqrt{\left(3 - \frac{5}{2}\right)^2 + \left(3 - \frac{7}{2}\right)^2} = \frac{1}{\sqrt{2}}$

3. What is the (L2-norm) reconstruction error of the point (3,4)? (Show your work.)
   Solution:
   Normalized $W_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$
   Now, $z = (x_1 - \mu_1)w_1 + (x_2 - \mu_2)w_2$

   $z = \frac{3-0}{\sqrt{2}} + \frac{4-1}{\sqrt{2}} = 3\sqrt{2}$

   Also, $\hat{x} = zW_1 + \mu = 3\sqrt{2}\left(\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}\right) + (0, 1)$

   So, $\hat{x} = (3, 4)$

   Since $x = \hat{x}$, Reconstruction error $= 0$

2.2 Data Visualization

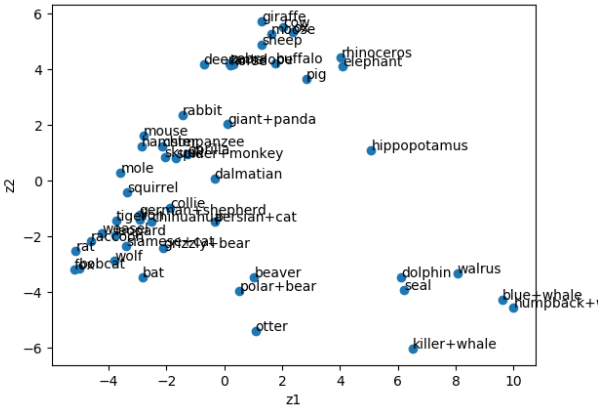
Rubric: {reasoning:2}

If you run `python main.py -q 2`, it will load the animals dataset and create a scatterplot based on two

randomly selected features. We label some random points, but because of the binary features the scatterplot shows us almost nothing about the data.

The class *pca.PCA* applies the classic PCA method (orthogonal bases via SVD) for a given k. Use this class so that the scatterplot uses the latent features z_i from the PCA model. Make a scatterplot of the two columns in Z, and label a bunch of the points in the scatterplot. Hand in your code and the scatterplot.

Solution: the code is available in main.py file.



2.3 Data Compression

Rubric: {reasoning:2}

1. How much of the variance is explained by our 2-dimensional representation from the previous question?

   Solution: The variance for k = 2 is 0.564159004282

2. How many PCs are required to explain 50% of the variance in the data?

   Solution: 3 PCs are required to explain50% of the variance
   variance for k = 1 0.617079080236
   variance for k = 2 0.564159004282
   variance for k = 3 0.523076456739
   variance for k = 4 0.491487753005
   variance for k = 5 0.462719415941

3 PCA Generalizations

3.1 Robust PCA

Rubric: {code:10}

If you run `python main -q 3.1` the code will load a dataset X where each row contains the pixels from a single frame of a video of a highway. The demo applies PCA to this dataset and then uses this to reconstruct

the original image. It then shows the following 3 images for each frame (pausing and waiting for input between each frame):

1. The original frame.
2. The reconstruction based on PCA.
3. A binary image showing locations where the reconstruction error is non-trivial.

Recently, latent-factor models have been proposed as a strategy for "background subtraction": trying to separate objects from their background. In this case, the background is the highway and the objects are the cars on the highway. In this demo, we see that PCA does an ok job of identifying the cars on the highway in that it does tend to identify the locations of cars. However, the results aren't great as it identifies quite a few irrelevant parts of the image as objects.

Robust PCA is a variation on PCA where we replace the L2-norm with the L1-norm,

$$f(Z, W) = \sum_{i=1}^{n} \sum_{j=1}^{d} |w_j^T z_i - x_{ij}|,$$

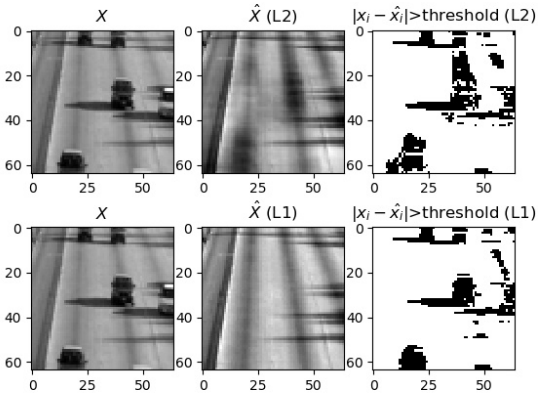
and it has recently been proposed as a more effective model for background subtraction. Complete the class *pca.RobustPCA*, that uses a smooth approximation to the absolute value to implement robust PCA. Comment on the quality of the results.

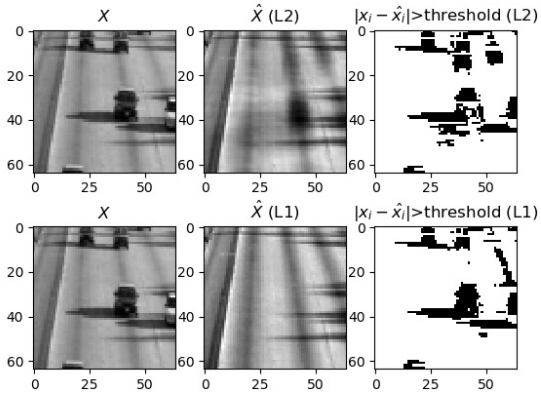
Hint: most of the work has been done for you in the class *pca.AlternativePCA*. This work implements an alternating minimization approach to minimizing the (L2) PCA objective (without enforcing orthogonality). This gradient-based approach to PCA can be modified to use a smooth approximation of the L1-norm. Note that the log-sum-exp approximation to the absolute value may be hard to get working due to numerical issues, and a numerically-nicer approach is to use the "multi-quadric" approximation:

$$|\alpha| \approx \sqrt{\alpha^2 + \epsilon},$$

where ϵ controls the accuracy of the approximation (a typical value of ϵ is 0.0001).

solutions: The following figure shows the difference between L1 regularization and L2 regularization in PCA. L1 regularization does a better job of identifying objects from their background. the cars are more accurately identified and removed from the background. the code is available in pca.py file in RobustPCA class.



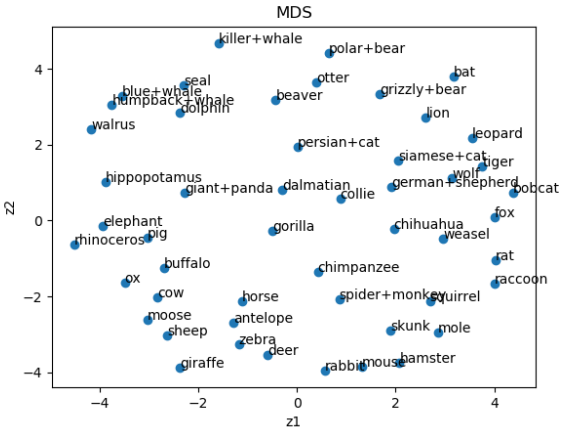


4 Multi-Dimensional Scaling

If you run `python main.py -q 4`, the code will load the animals dataset and then apply gradient descent to minimize the following multi-dimensional scaling (MDS) objective (starting from the PCA solution):

$$f(Z) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=i+1}^{n}(\|z_i - z_j\| - \|x_i - x_j\|)^2. \tag{1}$$

The result of applying MDS is shown below.



Although this visualization isn't perfect (with "gorilla" being placed close to the dogs and "otter" being placed close to two types of bears), this visualization does organize the animals in a mostly-logical way.

4.1 ISOMAP

Rubric: {code:10}

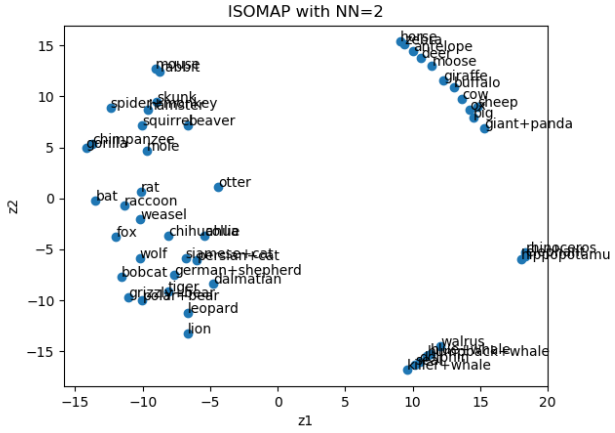
Euclidean distances between very different animals are unlikely to be particularly meaningful. However, since related animals tend to share similar traits we might expect the animals to live on a low-dimensional

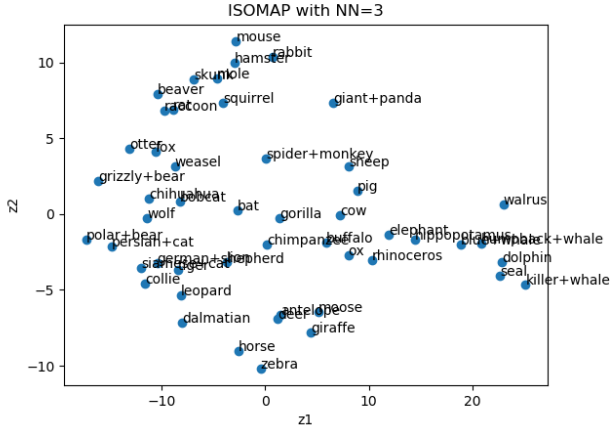
manifold. This suggests that ISOMAP may give a better visualization. Fill in the class *ISOMAP* so that it computes the approximate geodesic distance (shortest path through a graph where the edges are only between nodes that are k-nearest neighbours) between each pair of points, and then fits a standard MDS model (1) using gradient descent. Plot the results using 2 and using 3-nearest neighbours.

Note: when we say 2 nearest neighbours, we mean the two closest neigbours excluding the point itself. This is the opposite convention from what we used in KNN at the start of the course.

The function *utils.dijskstra* can be used to compute the shortest (weighted) distance between two points in a weighted graph. This function requires an $n \times n$ matrix giving the weights on each edge (use 0 as the weight for absent edges). Note that ISOMAP uses an undirected graph, while the k-nearest neighbour graph might be asymmetric. One of the usual heuristics to turn this into a undirected graph is to include an edge i to j if i is a KNN of j or if j is a KNN of i. (Another possibility is to include an edge only if i and j are mutually KNNs.)

Solution:





4.2 Reflection

Rubric: {reasoning:2}

Briefly comment on PCA vs. MDS vs. ISOMAP for dimensionality reduction on this particular data set. In your opinion, which method did the best job and why?

Solution: In Comaparison with PCA and MDS, I think ISOMAP did the best job of dimensionality reductionas it separates the animales the most. MDS and PCA produced an evenly distribution of the animals, where the separation is not that obvious and evident as in ISOMAP

5 Very-Short Answer Questions

Rubric: {reasoning:10}

1. Why is the kernel trick often better than explicitly transforming your features into a new space?
   Answer: Because in case of multi-dimensional polynomial basis, if we want to explicitly transform our features, the k-dimensional basis z_i that we use might be too huge to store, and this basis is only required to compute the Gram Matrix $K = ZZ^T$. So if we have a Kernel Function that computes $k(x_i, x_j)$, we don't need to compute z_i explicitly.

2. Why is the kernel trick more popular for SVMs than with logistic regression?
   Answer: Because in case of SVMs, if implemented properly, the cost of prediction can be reduced from $O(ndt)$ to $O(mdt)$ where 'm' is the number of support vectors, but in logistic regression, it's not so. So in case of a very large number of training examples, logistic regression is more expensive.

3. What is the key advantage of stochastic gradient methods over gradient descent methods?
   Answer: In case of Stochastic gradient method, iterations are 'n' times faster than gradient descent iterations, because instead of calculating the gradient for all training examples, we only calculate the gradient of one randomly chosen example.

4. Does stochastic gradient descent with a fixed α converge to the minimum of a convex function in general?
   Answer: No, because it has a tendency to show erratic behaviour when it gets too close to the solution, i.e. it bounces around the solution in a 'ball' of radius α.

5. What is the difference between multi-label and multi-class classification?
   Answer: In multi-class classification there is one true label, however, in multi-label classification, several (or none) of the labels can be applicable.

6. What is the difference between MLE and MAP?
   Answer: In MLE the objective functions are equivalent to maximizing $p(y|X, w)$, whereas MAP estimation directly models $p(w|X, y)$.

7. Linear regression with one feature and PCA with 2 features (and $k = 1$) both find a line in a two-dimensional space. Do they find the same line? Briefly justify your answer.
   Answer: No, because in case of Linear Regression, the vertical squared distance is minimized, whereas, for PCA, the orthogonal squared distance is minimized, so the two lines will not be same unless the vertical and the orthogonal distances are minimized by the same line.

8. Are the vectors minimizing the PCA objective unique? Briefly justify your answer
   Answer: No, because the minimizer is just like the 'span' of vectors, which is not unique.

9. Name two methods for promoting sparse solutions in a linear regression model that result in convex problems.
   Answer: Using non-negative constraints and L0/L1 regularization.

10. Can we use the normal equations to solve non-negative least squares problems?
    Answer: Not always, because normal equations don't follow the constraints of non-negativity.