Assignment 2, CSCI 471/571 Fall 2020

Kameron Decker Harris

Due: October 12, 11:59 p.m.

Group work following the guidelines in the Syllabus is okay. List your collaborators.
Total points: 78

The following sections should be computed by hand. Show your work.

Linear algebra

1. (Rank) Let $A = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 0 & 4 \\ 1 & 1 & 3 \end{bmatrix}$ and $B = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 0 & 1 \\ 1 & 1 & 2 \end{bmatrix}$. For each matrix A and B,

   1. *[2 points]* What is its rank?

   2. *[2 points]* What is a (minimal size) basis for its column span?

   [3 points] Compute an orthonormal basis for the column space of A.

2. (Matrix multiplication) Let $A = \begin{bmatrix} 0 & 2 & 4 \\ 2 & 4 & 2 \\ 3 & 3 & 1 \end{bmatrix}$, $\vec{b} = \begin{bmatrix} -2 & -2 & -4 \end{bmatrix}^T$, and $\vec{c} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$. Compute:

   1. *[1 points]* $A\vec{c}$

   2. *[1 points]* $A\vec{b}$

3. (Hyperplanes) Assume $\vec{w}$ is an d-dimensional vector and b is a scalar. A hyperplane in $\mathbb{R}^d$ is the set $\{\vec{x} \in \mathbb{R}^d : \vec{w}^T\vec{x} + b = 0\}$.

   1. *[1 points]* ($d = 2$ example) Draw the hyperplane for $\vec{w} = [-1, 2]^T$, $b = 2$. Label your axes.

   2. *[1 points]* ($d = 3$ example) Draw the hyperplane for $\vec{w} = [1, 1, 1]^T$, $b = 0$. Label your axes.

Gradients

4. For possibly non-symmetric matrices $A \in \mathbb{R}^{n\times n}$ $B \in \mathbb{R}^{m\times n}$, and $c \in \mathbb{R}$, let $f(\vec{u}, \vec{v}) = \vec{u}^T A\vec{u} + \vec{v}^T B\vec{u} + c$. Define the gradient $\nabla_{\vec{z}} f(\vec{u}, \vec{v}) = \begin{bmatrix} \frac{\partial f(\vec{u},\vec{v})}{\partial z_1} & \frac{\partial f(\vec{u},\vec{v})}{\partial z_2} & \dots & \frac{\partial f(\vec{u},\vec{v})}{\partial z_p} \end{bmatrix}^T$ for $\vec{z} \in \mathbb{R}^p$.

   1. *[1 points]* What dimensions must $\vec{u}$ and $\vec{v}$ have for the definition of f to make sense?

   2. *[2 points]* Explicitly write out the function $f(\vec{u}, \vec{v})$ in terms of the components $A_{i,j}$ and $B_{i,j}$ using appropriate summations over the indices.

   3. *[2 points]* What is $\nabla_{\vec{u}} f(\vec{u}, \vec{v})$ in terms of the summations over indices *and* vector notation?

   4. *[2 points]* What is $\nabla_{\vec{v}} f(\vec{u}, \vec{v})$ in terms of the summations over indices *and* vector notation?

Linear regression

5. In linear regression, we fit a linear function of the input $\vec{x} = [x_1, \ldots, x_d]^T$, a d-dimensional vector. These functions are of the form $f(\vec{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d$.

1. *[1 points]* Show that we can write $f(\vec{x}) = \vec{z}^T \vec{\beta}$ for a suitable vector $\vec{z}$ and $\vec{\beta} = [\beta_0, \beta_1, \ldots, \beta_d]^T$.

2. *[2 points]* Explain the role of the term β_0. If we don't include this term, what changes about the function $f(\vec{x})$?

6. *[2 points]* Write $\|X\vec{\beta} - \vec{y}\|^2$ in the same form as $f(\vec{u}, \vec{v})$ in Problem 4, identifying $\vec{u}, \vec{v}, A, B, c$. Note that c will depend on one of the vectors, but you can still apply the result from Problem 4 to compute the gradient. Use your result to obtain the normal equations for ordinary least squares.

7. Instead of just minimizing the squared error, as in ordinary least squares, suppose we decide to minimize the following *cost function*:

$$C(\vec{\beta}) = \|X\vec{\beta} - \vec{y}\|^2 + \alpha\|D(\vec{\beta} - \vec{\beta}')\|^2.$$

Here $X \in \mathbb{R}^{n \times d}$ is the usual data matrix, $\alpha > 0$, $D = \text{diag}(\sigma_1, \ldots, \sigma_d)$ is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \ldots > 0$, and $\vec{\beta}'$ is a vector with the same shape as $\vec{\beta}$. Consider the modified least-squares problem $\min_{\vec{\beta} \in \mathbb{R}^d} C(\vec{\beta})$:

1. *[4 points]* Compute $\nabla_{\vec{\beta}} C$. (Hint: Use Problem 4 again.)

2. *[4 points]* Set $\nabla_{\vec{\beta}} C = 0$ and derive a linear system of equations that an optimizer $\vec{\beta}^*$ must satisfy. Highlight the sizes of the matrices and which vector we are solving for.

3. *[2 points]* Take $D = I$ (the identity matrix). Explain what the "extra term" might be good for. You may wish to explain using your formula for $\vec{\beta}^*$ or using your intuition about norms and vectors. (Hint: What vectors $\vec{\beta}$ make the extra term small?)

4. *[2 points]* Take $\vec{\beta}' = 0$. Explain what the extra term might be good for. You may wish to explain using your formula for β^* or using your intuition about norms and vectors.

5. *[1 points]* Now, for general D and $\vec{\beta}'$, what is the effect of the extra term?

6. *[1 points]* What is the effect of varying α?

Programming exercises

For the following, you should program in Python and may use the `numpy` package and `matplotlib` for plotting, but *you may not use any of the built-in least-squares solvers.* Any output from your program (displays of matrices or vectors and plots) must be included in your pdf submission. Your code must be submitted as described in the Syllabus. All plots should be legible with axes labeled and legends if there are multiple things plotted.

8. (Basic `numpy`) For the $A, \vec{b}, \vec{c}$ as defined in Problem 2, use NumPy to compute (take a screen shot of your answer):

1. *[2 points]* What is A^{-1}?

2. *[1 points]* What is $A^{-1}\vec{b}$? What is $A\vec{c}$?

9. (Efficiency of linear systems solve) Use the starter code to generate random test problems with $A \in \mathbb{R}^{n \times n}$ and $\vec{b} \in \mathbb{R}^n$. Write a function that solves $A\vec{x} = \vec{b}$ for $\vec{x} \in \mathbb{R}^n$ the following ways:

1. *[1 points]* Form the inverse A^{-1} and use it to compute the solution.

2. *[1 points]* Form the pseudoinverse $A^{\dagger}$ and compute the pseudoinverse solution.

3. *[1 points]* Using the optimized function `np.linalg.solve`.

Call the functions `solve_system_inv`, `solve_system_pinv`, `solve_system_numpy`.

- *[2 points]* Plot the runtime of all three functions on a single plot, log-log axes, for logarithmically spaced n from 10 to 1000 (or larger if your computer doesn't complain). For each n, call function 10 times and report the average runtime.

- *[2 points]* Discuss any differences in scaling that you observe. Can you think of a case where you might want to use one of the slower methods?

10. (Polynomial features) Here we will investigate the effect of adding polynomial features to a synthetic dataset with dimension $d = 2$.

1. *[1 points]* Compute the SVD of the X matrix using `np.linalg.svd`. Plot the singular value spectrum (the vector of singular values ordered from largest to smallest), with logarithmically scaled axis for the singular values. Include the plot here.

2. *[2 points]* Write a function that returns polynomial features of degree p. Degree p polynomials should include terms of degree $p-1$, etc. Include constant terms. **For simplicity, skip interaction terms.**

3. *[3 points]* On the same axes, plot the singular value spectrum for degree 0 through degree 10 polynomials using different colors or symbols to indicate degree.

4. *[2 points]* Describe any trends you see.

5. *[2 points]* Explain how the trends you see in the singular value spectrum will manifest in larger or smaller coefficients of $\vec{\beta}^*$ returned by ordinary least-squares.

11. (Polynomial regression) Using the same data as Problem 10, we now turn to fitting a regression model. Reuse your code from Problem 10 and again ignore interaction terms.

1. *[4 points]* Write a function that evaluates the test error numerically. Generate a uniformly spaced grid of n_{test} points over $[0,1]^2 = \{\vec{x} \in \mathbb{R}^2 : 0 \le x_1, x_2 \le 1\}$. For each point in the *test set*, evaluate the true function and the estimate returned by polynomial regression (the polynomial function fit using the *training set* of n_{samples} generated in the prototype code), and compute the squared error. The function should return the mean square error (MSE) over all n_{test} points, the *test MSE*. See the function prototype provided. It is ok if it only works with n_{test} equal to a square.

2. *[4 points]* Plot the training MSE (evaluated over the n_{samples} used to fit the polynomial) and test MSE (using your function for part 1) for polynomial degrees 0–10. Make separate plots for $n_{\text{test}} = 16$, 64, and 144. What is the optimal degree?

3. *[2 points]* Change the number of samples used to fit to $n_{\text{samples}} = 100$. Do the results change?

4. *[1 points]* Why could we get away with ignoring interaction terms for this target function?

12. (Interpretation and covariance) Start from the example code which loads a dataset of median housing prices (our targets y_i) and a number of predictors (the features $\vec{x}_i$). For a description of the predictors included, see `http://lib.stat.cmu.edu/datasets/boston`. We use the same ordering and MEDV is the target.

1. *[1 points]* Fit the entire dataset using ordinary least squares with an intercept term. Report your coefficients $\vec{\beta}^*$.

2. *[1 points]* Which coefficients are most important for the linear function? Least important?

3. *[2 points]* Compute the sample covariance matrix for the data. This is defined as

$$\frac{1}{n-1}\sum_{i=1}^{n}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T,$$

where $\vec{\mu} = \frac{1}{n}\sum_{i=1}^{n}\vec{x}_i$ is the sample mean. Print your result.

4. *[1 points]* Show that, when the mean is zero (i.e. the data are *centered*), that the covariance matrix appears in the normal equations. **Do this math by hand.**

5. *[4 points]* You have just started a data scientist job at the massive real estate website Zilfin. Your job is to use machine learning to set the housing prices that appear on Zilfin's house browser, an essential part of your business. You have been handed town-level housing data with similar attributes to these (read the description of the dataset linked above) along with attributes of the house in question.

   Do you have any qualms about estimating prices in this way? Should certain columns of these data be left out? Even if we leave out certain columns, what does the covariance matrix tell you about how the left out data might correlate with the response? Is it fair to price houses this way?

   Write 2–3 paragraphs (≈200–350 words).