CPSC 540: Machine Learning

Matrix Notation

Mark Schmidt

University of British Columbia

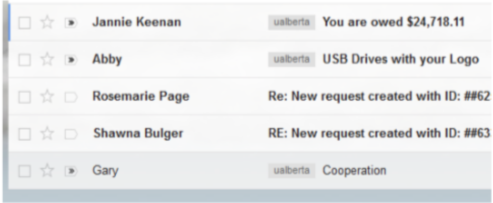
Winter 2017

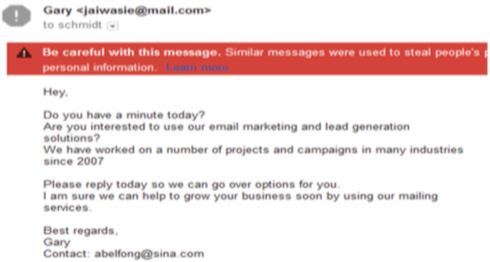
Admin

- Auditting/registration forms:
  - Submit them at end of class, pick them up end of next class.
  - I need your prereq form before I'll sign registration forms.
- Website/Piazza:
  - `https://www.cs.ubc.ca/~schmidtm/Courses/540-W17`.
  - `https://piazza.com/ubc.ca/winterterm22016/cpsc540`.
- No tutorial this week.
- Assignment 1 posted, due January 16.
- Viewing lecture slides:
  - Course slides will (mostly) be done in Beamer.
  - Beamer outputs PDFs, and simulates "transitions" by making new slides.
  - Don't freak out when you see 100+ slides lectures.
  - To review lectures, find PDF viewer that doesn't have transitions between slides.

Motivating Problem: E-Mail Spam Filtering

- We want to build a system that filters spam e-mails:





- We have a big collection of e-mails, labeled by users.
- We can formulate this as supervised learning.

Supervised Learning Notation

- Supervised learning input is a set of n training examples.
- Each training example usually consists of:
  - A set of features x^i.
  - A label y^i

- For e-mail spam filtering:
  - Features could indicate words, phrases, regular expressions, etc.
  - Label could be $(+1)$ for "spam" and (-1) for "not spam".
  - Supervised learning has been dominant approach for ~ 20 years.

- Supervised learning output is a model:
  - Given new inputs $\hat{x}^i$, model makes a prediction $\hat{y}^i$.
  - Goal is to maximize accuracy on new examples (test error).

Fundamental Trade-off of Learning Theory

- Learning theory says we must trade-off between two factors:
  - How low we can make the training error.
  - How well the training error approximates the test error.

- With complex models:
  - We can make the training error low, but it's a poor test error approximation.
- With simple models:
  - Training error is a good approximation of test error, but training error might be large.

Loss Plus Regularizer Framework

- We usually try to find the "best" model by solving an optimization problem.
- Typically this involves minimizing a function f of the form

$$f(w) = \underbrace{\sum_{i=1}^{n} f_i(w)}_{\text{data-fitting term}} + \underbrace{\lambda g(w)}_{\text{regularizer}}.$$

- Loss function f_i measures how well we fit example i with parameters w.
- Regularizer g measures how complicated the model is with parameters w.

- Regularization parameter $\lambda > 0$ controls strength of regularization:
  - Controls complexity of model, with large λ leading to less overfitting.
  - Usually set by optimizing error on a validation set or with cross-validation.

L2-Regularized Least Squares

- "Loss plus regularizer" framework:

$$f(w) = \underbrace{\sum_{i=1}^{n} f_i(w)}_{\text{data-fitting term}} + \underbrace{\lambda g(w)}_{\text{regularizer}}.$$

- We often consider linear models where

$$\hat{y}^i = w^T \hat{x}^i.$$

- A common choice for f is L2-regularized least squares where

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x^i - y^i)^2 + \frac{\lambda}{2}\sum_{j=1}^{d} w_j^2,$$

  so we have

$$f_i(w) = \frac{1}{2}(w^T x^i - y^i)^2, \quad g(w) = \frac{1}{2}\sum_{j=1}^{d} w_j^2.$$

Other Loss Functions and Regularizers

- "Loss plus regularizer" framework:

$$f(w) = \underbrace{\sum_{i=1}^{n} f_i(w)}_{\text{data-fitting term}} + \underbrace{\lambda g(w)}_{\text{regularizer}}.$$

- Other choices of loss function:
  - Absolute error $|w^T x^i - y^i|$ is more robust to outliers.
  - Hinge loss $\max\{0, 1 - y^i w^T x^i\}$ is better for binary y^i.
  - Logistic loss $\log(1 + \exp(-y^i w^T x^i))$ is better for binary and is smooth.
  - Softmax loss $-w_{y^i}^T x^i + \log(\sum_{c=1}^{k} \exp(w_c^T x^i))$ for discrete y^i.
- Another common regularizer is L1-regularization,

$$g(w) = \sum_{j=1}^{d} |w_j|,$$

which encourages sparsity in w (many w_j are set to zero for large λ).

Other Loss Functions and Regularizers

- "Loss plus regularizer" framework:

$$f(w) = \underbrace{\sum_{i=1}^{n} f_i(w)}_{\text{data-fitting term}} + \underbrace{\lambda g(w)}_{\text{regularizer}}.$$

- To model non-linear effects we can use:
  - Non-linear features transformations ("change of basis" and kernel trick).
  - Unsupervised learning methods like sparse matrix factorization.
  - Neural networks which try to learn good features.

(pause)

Column-Vector Notation

- In this course we'll assume that all vectors are column-vectors,

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}, \quad y = \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^n \end{bmatrix}, \quad x^i = \begin{bmatrix} x_1^i \\ x_2^i \\ \vdots \\ x_d^i \end{bmatrix}.$$

- Note: in some cases we use superscripts to index based on training example.
  - I'm using w_j as the scalar parameter j.
  - I'm using y^i as the label of example i (currently a scalar).
  - I'm using x^i as the column-vector of features for example i.
  - I'm using x_j^i to denote feature j in training example i.

Matrix and Norm Notation

- Instead of writing L2-regularized least squares as

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^Tx^i - y^i)^2 + \frac{\lambda}{2}\sum_{j=1}^{d} w_j^2,$$

  in this course we'll use matrix and norm notation,

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

- Further, instead of just focusing on gradients,

$$\nabla f(x) = X^T(Xw - y) + \lambda w,$$

  we're going to also use Hessians

$$\nabla^2 f(x) = X^TXw + \lambda I,$$

  and use eigenvalues in our arguments.

Matrix and Norm Notation for L2-Regularization

- Let's first focus on the regularization term,

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^Tx^i - y^i)^2 + \frac{\lambda}{2}\sum_{j=1}^{d} w_j^2,$$

- Recall the definitions of inner product and L2-norm of vectors,

$$\|v\| = \sqrt{\sum_{j=1}^{d} v_j^2}, \quad u^Tv = \sum_{j=1}^{d} u_jv_j,$$

- Using this we can write regularizer in various forms using

$$\begin{aligned}\|w\|^2 &= \sum_{j=1}^{d} w_j^2 \\ &= \sum_{j=1}^{d} w_jw_j = w^Tw.\end{aligned}$$

Matrix and Norm Notation for Least Squares

- Let's next focus on the least squares term,

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x^i - y^i)^2 + \frac{\lambda}{2}\|w\|^2.$$

- Let's define the residual vector r with elements

$$r_i = w^T x^i - y^i.$$

- We can write the least squares term as squared L2-norm of residual,

$$\begin{aligned} \sum_{i=1}^{n}(w^T x^i - y^i)^2 &= \sum_{i=1}^{n} r_i^2 \\ &= r^T r \\ &= \|r\|^2. \end{aligned}$$

Matrix and Norm Notation for Least Squares

- Let's next focus on the least squares term,

$$f(w) = \frac{1}{2}\|r\|^2 + \frac{\lambda}{2}\|w\|^2, \quad \text{with} \quad r_i = w^T x^i - y^i$$

- We'll use X to denote the data matrix containing the x^i (transposed) in the rows:

$$X = \begin{bmatrix} \text{---} & (x^1)^T & \text{---} \\ \text{---} & (x^2)^T & \text{---} \\ & \vdots & \\ \text{---} & (x^n)^T & \text{---} \end{bmatrix}$$

- Using that $w^T x^i = (x^i)^T w$ and the definitions of r, y, and X we have

$$r = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix} = \begin{bmatrix} w^T x^1 - y^1 \\ w^T x^2 - y^2 \\ \vdots \\ w^T x^n - y^n \end{bmatrix} = \begin{bmatrix} (x^1)^T w \\ (x^2)^T w \\ \vdots \\ (x^n)^T w \end{bmatrix} - \underbrace{\begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^n \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} \text{---} & (x^1)^T & \text{---} \\ \text{---} & (x^2)^T & \text{---} \\ & \vdots & \\ \text{---} & (x^n)^T & \text{---} \end{bmatrix}}_{X} w - y = Xw - y.$$

Solving L2-Regularized Least Squares

- So we can write L2-regularized least squares objective as

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

- We want to minimize this function.
- Fortunately, minimizing "strictly-convex quadratic" functions is mechanical:
  - Solve for the unique w where $\nabla f(w) = 0$.
- This is easy to do by taking gradient in matrix notation.

Digression: Linear Functions and their Derivatives

- A linear function is a function of the form
$$f(w) = a^T w + \beta,$$
for a vector a and a scalar β.
- Computing gradient of linear function in matrix notation:
  1. Convert to summation notation:
  $$f(w) = \sum_{k=1}^{d} a_k w_k + \beta.$$
  2. Take partial derivative of generic i:
  $$\frac{\partial}{\partial w_i} f(w) = a_i.$$
  3. Assemble into a vector and convert to matrix notation:
  $$\nabla f(w) = \begin{bmatrix} \frac{\partial}{\partial w_1} f(w) \\ \frac{\partial}{\partial w_2} f(w) \\ \vdots \\ \frac{\partial}{\partial w_d} f(w) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_d \end{bmatrix} = a.$$

Digression: Linear Functions and their Derivatives

- A linear function is a function of the form

$$f(w) = a^T w + \beta,$$

  for a vector a and a scalar β.
- We can do the same sequence to get the Hessian matrix $\nabla^2 f(w)$:
  1. Convert to summation notation: $$\frac{\partial}{\partial w_i} f(w) = a_i.$$
  2. Take partial derivative of generic j: $$\frac{\partial}{\partial w_i \partial w_j} f(w) = 0.$$
  3. Assemble into a matrix and convert to matrix notation:

$$\nabla^2 f(w) = \begin{bmatrix} \frac{\partial}{\partial w_1 \partial w_1} f(w) & \frac{\partial}{\partial w_1 \partial w_2} f(w) & \cdots & \frac{\partial}{\partial w_1 \partial w_d} f(w) \\ \frac{\partial}{\partial w_2 \partial w_1} f(w) & \frac{\partial}{\partial w_2 \partial w_2} f(w) & \cdots & \frac{\partial}{\partial w_2 \partial w_d} f(w) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial w_d \partial w_1} f(w) & \frac{\partial}{\partial w_d \partial w_2} f(w) & \cdots & \frac{\partial}{\partial w_d \partial w_d} f(w) \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} = 0.$$

Digression: Quadratic Functions and their Derivatives

- A quadratic function is a function of the form

$$f(w) = \frac{1}{2}w^T A w + b^T w + \gamma,$$

for a square matrix A, vector b, and scalar γ.

- An example is the L2-regularizer:

$$f(w) = \frac{\lambda}{2}\|w\|^2,$$

with $A = \lambda I$, $b = 0$, $\gamma = 0$.

Digression: Quadratic Functions and their Derivatives

- Another quadratic function is the squared error:

$$
\begin{aligned}
f(w) &= \frac{1}{2}\|Xw - y\|^2 && \text{least squares objective} \\
&= \frac{1}{2}(Xw - y)^T(Xw - y) && \|v\|^2 = v^T v \\
&= \frac{1}{2}((Xw)^T - y^T)(Xw - y) && (A - B)^T = A^T - B^T \\
&= \frac{1}{2}(w^T X^T - y^T)(Xw - y) && (AB)^T = B^T A^T \\
&= \frac{1}{2}(w^T X^T Xw - w^T X^T y - y^T Xw + y^T y) && \text{distributive rule} \\
&= \frac{1}{2}(w^T X^T Xw - 2w^T X^T y + y^T y) && w^T X^T y = y^T Xw \text{ (scalar)} \\
&= \frac{1}{2}w^T X^T Xw - w^T X^T y + \frac{1}{2}y^T y,
\end{aligned}
$$

with $A = X^T X$, $b = X^T y$, $\gamma = \frac{1}{2}y^T y$.

Digression: Quadratic Functions and their Derivatives

- Let's compute gradient of a simple quadratic,
$$f(w) = w^T A w.$$
- In summation notation:
$$w^T A w = w^T \underbrace{\begin{bmatrix} \sum_{k=1}^d a_{1k} w_k \\ \sum_{k=1}^d a_{2k} w_k \\ \vdots \\ \sum_{k=1}^d a_{dk} w_k \end{bmatrix}}_{Aw} = \sum_{l=1}^d \sum_{k=1}^d w_k a_{kl} w_l.$$
- Generic partial derivative:
$$\frac{\partial}{\partial w_i} f(w) = 2a_{ii} w_i + \sum_{k \neq i} w_k a_{ki} + \sum_{l \neq i} a_{il} w_l = \sum_{k=1}^d w_k a_{ki} + \sum_{l=1}^d a_{il} w_l = w^T A_i + A_i^T w,$$
where A_i is column i and A_i^T is row i.

Digression: Quadratic Functions and their Derivatives

- Assemble into a vector and convert to matrix notation:

$$\nabla f(w) = \begin{bmatrix} w^T A_1 + A_1^T w \\ w^T A_2 + A_2^T w \\ \vdots \\ w^T A_d + A_d^T w \end{bmatrix} = A^T w + Aw.$$

- Giving the final result

$$\nabla[w^T A w] = (A^T + A)w \qquad \text{(general case)}$$

$$\nabla[w^T A w] = 2Aw \qquad \text{(symmetric } A\text{).}$$

- Note that this generalizes the scalar result that $\frac{d}{dw}[w\alpha w] = \frac{d}{dw}[\alpha w^2] = 2\alpha w$.
- By repeating the procedure we get that the Hessian is

$$\nabla^2[w^T A w] = A^T + A,$$

or $\nabla^2 f(w) = 2A$ for symmetric A.

Solving L2-Regularized Least Squares

- So we can write L2-regularized least squares objective as

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

  or as a quadratic function,

$$\begin{aligned} f(w) &= \frac{1}{2}w^T X^T X w - w^T X^T y + \frac{1}{2}y^T y + \frac{\lambda}{2}w^T I w \\ &= \frac{1}{2}w^T (X^T X + \lambda I) w - w^T X^T y + \frac{1}{2}y^T y, \end{aligned}$$

  where $A = X^T X + \lambda I$ (symmetric), $b = -X^T y$, and $\gamma = \frac{1}{2}y^T y$.
- Using our tedious matrix calculus exercises we have

$$\nabla f(w) = \underbrace{(X^T X + \lambda I)}_{A} w \underbrace{- X^T y}_{b}.$$

Solving L2-Regularized Least Squares

- Setting the gradient to 0 we have
$$(X^TX + \lambda I)w - X^Ty = 0,$$
or equivalently that
$$(X^TX + \lambda I)w = X^Ty.$$
- We'll show that A is invertible. We get solution by pre-multiplying by its inverse,
$$(X^TX + \lambda I)^{-1}(X^TX + \lambda I)w = (X^TX + \lambda I)^{-1}X^Ty$$
$$w = (X^TX + \lambda I)^{-1}X^Ty.$$
  - In Matlab: w=(X'*X-lambda*eye(d))\(X'*y).
- This is the unique w where $\nabla f(w) = 0$, but is it a minimizer?
  - Yes, because A is positive-definite.

Positive-Definite Matrices

- Equivalent definitions of a positive-definite matrix A:
  1. All eigenvalues of A are positive.
  2. The quadratic $v^T A v$ is positive for all non-zero v.
- Because eigenvalues are positive, positive definite matrices are invertible.
- To indicate that A is positive-definite we write $A \succ 0$.
- Sufficient condition for w to be a minimizer: $\nabla f(w) = 0$ and $\nabla^2 f(w) \succ 0$.

Positive-Definite Matrices

- The matrix $A = (X^TX + \lambda I)$ is positive-definite for any X for any $\lambda > 0$:
  $$v^TAv = v^T(X^TX+\lambda I)v = v^TX^TXv+v^T(\lambda I)v = (Xv)^TXv+\lambda v^Tv = \|Xv\|^2+\lambda\|v\|^2.$$
- Both terms are non-negative because they're norms.
- Second term $\|v\|$ is positive because $v \neq 0$ and $\lambda > 0$.

- If $\lambda = 0$ then it's only positive semi-definite:
  $$X^TX \succeq 0.$$
  - Replace "positive" with "non-negative" in definition of positive-definite.
  - X^TX may not be invertible and we many have multiple solutions to $\nabla f(w) = 0$.
  - The set of minimizers is the set of solutions to this linear system:
  $$\underbrace{(X^TX)}_{A}w = \underbrace{X^Ty}_{b}.$$

Summary

- Machine learning: automatically detecting patterns in data to help make predictions and/or decisions.
- CPSC 540: advanced/difficult graduate-level 2nd or 3rd course on this topic.
- Supervised learning: using data to learn input:output map.
- Loss plus regularizer optimization is most common machine learning framework.
- Matrix and norm notation are needed to describe several advanced topics.
- Linear and quadratic functions arise frequently in machine learning.
- L2-regularized least squares will be our "default" method that we'll improve on.

- Next time: solving non-quadratic problems.