Math of Data Lecture 1 - Intro

Vlad Kobzar

APAM, Columbia

- Course overview
- A few motivating examples

- Combine theory and computation
  - Theory tells us about solutions and how to find them
  - Computation allows us to find solutions
  - They are related: understanding computational methods is a type of theory

Tools

- The main math tools for this course are linear algebra and probability/statistics
- The main computational tool is optimization
- Probability and statistics will come in two forms:
  - *Randomized models*: data is modeled by some unknown distribution; the problem would entail estimating that distribution
  - *Randomized algorithms*, e.g., stochastic gradient descent

Supervised learning and linear models

- Linear models
  - *regression, such as OLS*
  - classification, including perceptron, logistic regression and SVMs
  - *focus on geometric and probabilistic interpretations and computational solutions*
- Fitting models to training data, such as empirical risk minimization
- Performance on unseen (test) data and regularization techniques
- *Sparse models and compressed sensing*

Regression example

Data points $(a_1, b_1), \ldots, (a_n, b_n) \in \mathbb{R}^d \times \mathbb{R}$ organized as

- *Features* $A = \begin{bmatrix} -a_1- \\ -a_2- \\ \vdots \\ -a_n- \end{bmatrix}$ and *response* $b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$
- E.g., NOAA publishes hourly observation of temperature at various stations across the US
  - The Yellowstone sensor fails at time τ
  - Can we predict the temperature $\hat{b}_\tau$ there from the contemporaneous observations at other stations a_τ ?
  - Use the observations A and b from the other periods to "fit" to the data a simple linear model

$$f(a) = x^\top a.$$

- CONCEPT CHECK: Can you "solve" $Ax = b$ when $n >> d$ ("big data" regime)?

OLS

- The least-squares fit between the data and the model

$$\hat{x} = \arg\min_{x} R(x)$$

where

$$R(x) = \sum_{t=1}^{n} (b_t - a_t^\top x)^2 = \|b - Ax\|_2^2$$
$$= x^\top A^\top Ax - 2b^\top Ax + b^\top b$$

- If A is full rank and $n \geq d$, $A^\top A$ is positive definite
- By the 2nd derivative test

$$\hat{x} = \left(A^\top A\right)^{-1} A^\top b$$

Stats interpretation

- If the data is given by a probabilistic model

$$b \sim Ax + N(0, \sigma^2 I)$$

OLS provides weights that make the data most probable, i.e.

$$x_{MLE} = \arg\max_x L(x, \sigma^2) = \arg\min_x R(x) = \hat{x}$$

where the log of the likelihood function is

$$L(x, \sigma^2) = p(b|Ax, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\|b-Ax\|^2}{2\sigma^2}}$$

- *Pf:* Maximizing $L(x, \sigma^2)$ w.r.t. x is the same as minimizing

$$R(x) = \|b - Ax\|_2^2.$$

Geometric interpretation

$$\hat{x} = \arg\min_x R(x)$$

where

$$R(x) = \|b - Ax\|_2^2$$

- $A\hat{x} = UU^T b$ - orthogonal projection of b on the span of the columns of A
- Prove using the SVD: $A = U\Sigma V^T$.

Computational aspects

$$\hat{x} = \arg\min_x R(x) = \left(A^\top A\right)^{-1} A^\top b$$

- But if d is large, inverting $A^\top A$ is computationally expensive.
- The factorization $A = QR$ avoids inversion
- First solve for $\lambda = Rx = Q^{-1}b = Q^\top b$
- Then back-substitution to solve $\lambda = Rx$ for x
- For sparse data, use iterative optimization methods (e.g., gradient descent and conjugate gradients)
- Since R is convex, convergence is guaranteed; can study rates
- For very large data use randomized algorithms

Overfitting

- Small error on the training set, but high error on a test set because $\hat{x}$ will fit the features that may not be relevant (e.g., sensors very far from Yellowstone)
- Can we find a sparse linear model?
  - E.g., predict the Yellowstone temperatures based on observations from a small subset of the stations
  - This subset is “learned” from the training data

Sparse regression (LASSO)

- l_0 penalty: $\|x\|_0 = \#$ of nonzero entries of x
- This regularization enforces sparsity: for $\lambda > 0$

$$x_0 = \arg\min_x \Big(R(x) + \lambda \|x\|_0 \Big)$$

- But is intractable (the objective not convex; l_0 not a norm)
- Would a "relaxation" to the l_1 norm also promote sparsity?
- LASSO
  - *Penalized form*

$$x_1 = \arg\min_x \Big(R(x) + \lambda \|x\|_1 \Big)$$

  - *Equivalent to constrained form*

$$x_1 = \arg \min_{\|x\|_1 \le r} R(x)$$

  - Pf. by a Lagrange multiplier-type calculation

LASSO

- Constrained form

$$x_1 = \arg\min_{\|x\|_1 \leq r} R(x)$$

- By completing the squares,

$$R(x) = (x - \hat{w})^\top AA^T (x - \hat{w}) + R(\hat{w})$$

where the OLS solution $\hat{w}$ of the unconstrained problem is the center of the ellipsoid OLS level sets

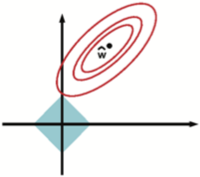

Figure: Level sets of $R(x)$ in red and the area satisfying $\|x\|_1 \leq r$ in blue (Fig 3.11 from [6]).

Solving LASSO numerically

- No general closed form solution
  - Even for OLS, the closed form solution is not used for large data sets due to computation cost of matrix inversion
- Use computational solutions
  - LASSO can be reduced to a convex optimization problem (quadratic program), can use standard iterative solvers
  - Can be more efficient to use other methods that exploit the structure of the lasso objective, e.g., the linear separability of the l^1 norm

Sparse inverse problems

- Now we have $n < d$ for $A \in \mathbb{R}^{n \times d}$ - "inverse problem" regime, e.g. MRI
- CONCEPT CHECK: If b in column space of A, can you solve $Ax = b$?
- With sparsity and other technical assumptions, l_1 minimization can exactly recover a sparse vector x.
- (Candes, Tao, Donoho) For

  $$x^* = \arg\min \|x\|_1$$
  $$s.t.\ b = Ax$$

  if (a) the row of A are not too localized so that they won't miss the entries of S-sparse x and (b) there is enough data $n \geq O(S \log d)$.
- key idea entails recovering the support of x (i.e., indices of nonzero entries) and therefore reducing it to a well-posed problem.

Course intro cont'd

- Unsupervised learning/dimensionality reduction
  - Principal component analysis (PCA)
  - *Matrix completion and the Netflix problem*
  - Clustering and graph-based learning
  - Ranking and the PageRank algorithm
- High-dimensional data: randomized LA and random projections
- Nonlinear models
  - Kernel methods and representer theorem
  - Mathematical aspects of deep learning, including universal approximation, backprop and vanishing gradients
  - CNNs and neural models for sequential data and graphs
- Generative models and semi-supervised learning
  - comparison to discriminative models in high dimensions
  - generalizations to diffusion models

Matrix completion

- Low rank models are common when only a few factors explain the variance in data organized in the matrix.
- Motivation: Netflix competition

$$
\begin{array}{l} \\ \text{The Dark Knight} \\ \text{Spiderman 3} \\ \text{Love Actually} \\ \text{Bridget Jones's Diary} \\ \text{Pretty Woman} \\ \text{Superman 2} \end{array}
\begin{array}{c} \begin{array}{cccc} \textit{Bob} & \textit{Molly} & \textit{Mary} & \textit{Larry} \end{array} \\ \begin{pmatrix} -10 & -10 & 10 & 5 \\ -7 & -10 & 8 & 10 \\ 8 & 10 & -5 & -9 \\ 10 & 4 & -6 & -10 \\ 8 & 9 & -9 & -4 \\ -9 & -8 & 9 & 10 \end{pmatrix} \end{array} := A,
$$

- To make a recommendation, estimate missing entries

$$
\begin{array}{lcccc} & \textit{Bob} & \textit{Molly} & \textit{Mary} & \textit{Larry} \\ \text{X-Men 7: Mutant Mosquito} & (-10 & ? & 8 & 10\ ) \end{array}
$$

- Fit a low rank model using the SVD: $A = U\Sigma V^T$
  - a truncated rank-k SVD is the best rank-k approximation of A

Matrix completion

- Low rank structure implies correlation between entries
- *Netflix problem*: How do we exploit it to predict missing entries?
- E.g. where a user is going to like a new movie
- E.g., if the below matrix is rank 1, then we must have 1 in place of the missing entry.

$$\begin{pmatrix} 1 & ? & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \mathbb{1}\mathbb{1}^T$$

- This seems like an easy matrix to complete.

Matrix completion

- On the other hand, if a matrix is sparse or its rows correlate with the canonical basis, it seems much harder to complete

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & ? \end{pmatrix} = \begin{bmatrix} 0 \\ 0 \\ ? \end{bmatrix} \begin{bmatrix} 0 & 0 & ? \end{bmatrix}$$

  Therefore differences in the structure of a low rank matrix may determine how hard or difficult it is to complete.
- Coherence (or localization of non-zero values of rows and columns) is relevant here
- For example, if the left singular vectors (columns of U) correlate with the canonical basis vectors, matrix will hard to complete.

Nuclear norm minimization

- Since rank is not a convex function, minimization of the rank subject to known entries $A_0 = \{(i, j), a_{ij}\}$ is not computationally tractable.

$$
\begin{aligned}
(N) \qquad & \min \operatorname{rank}(A) \\
& A \in \mathbb{R}^{m \times n} \\
& A_{ij} = a_{ij} \text{ for } (i, j) \in A_0
\end{aligned}
$$

- Note that rank is an l_0 "norm" of Σ for $A = U \Sigma V^T$.

Nuclear norm minimization

- Instead use a "convex relaxation" based on minimization of the nuclear norm:

$$
\begin{aligned}
(N) \qquad & \min \|A\|_N \\
& A \in \mathbb{R}^{m \times n} \\
& A_{ij} = a_{ij} \text{ for } (i,j) \in A_0
\end{aligned}
$$

  where

$$\|A\|_N = \sum_{i=1}^{r} \sigma_i$$

  and σ_i are singular values and r is rank of A
- Note that rank is an l_1 "norm" of Σ.

From predictions to actions

- Sequential decision making and Kalman filtering
- Interaction with adversarial environments – online learning
- *Incomplete information and exploration-exploitation trade-offs*
  - *Bandit problems* and mathematical aspects of reinforcement learning

Modern example - movie ratings

- Let a feature vector x describe a user
- We choose 1 out of 5 hit movies and recommend it to x
- If we could figure out what movie this user would have liked the best (whether or not it's the one we chose)
  - "Full information" in the sense that we know what was the best action
  - Next time we have a similar user $x' \approx x$, we know what to do

Movie ratings - partial information

- Instead we only get the feedback on the movie recommended to x
  - Let's say 3 out of 5 stars
  - Next time we have a similar user $x' \approx x$, should we recommend the same movies
  - Or try a different one to get 5 stars??

Next steps

- Review of linear algebra, probability, statistics and optimization
- PCA, least squares

References I

- [1] Carlos Fernandez-Granda, *DS-GA 1013 / MATH-GA 2821 Mathematical Tools for Data Science, Lecture Notes*, 2020
- [2] Kevin P. Murphy, Machine Learning: a Probabilistic Perspective, MIT Press, 2012
- [3] David Rosenberg, *DS-GA 1003 Machine Learning and Computational Statistics, Lecture Notes*, 2017
- [4] David Rosenberg, *DS-GA 3001: Tools and Techniques for Machine Learning, Lecture Notes*, NYU Fall 2021, `https://github.com/davidrosenberg/ttml2021fall`
- [5] Hastie, Tibshirani, Wainwright, Statistical Learning with Sparsity: The Lasso and Generalizations, Chapman & Hall/CRC Monographs on Statistics and Applied Probability, 2015

References II

[6] Hastie, Tibshirani, Friedman, Elements of Statistical Learning & Springer, 2009