ML Cheatsheet

Steve Young

Abstract: Everything I know about machine learning.

Contents

1 Conventions 1

2 Linear Regression 2

2.1 Basics 2

3 Logistic Regression 2

3.1 2 classes 2

3.2 2+ classes 3

4 Gaussian Discriminant Analysis (GDA) and Naive Bayes 3

4.1 GDA 3

4.2 Naive Bayes (for text classification) 3

5 Bias/Variance Tradeoff 4

6 Model Assessment 4

6.1 Classification 4

6.1.1 Precision/Recall 4

6.1.2 Receiver operating characteristic curve (ROC) 5

6.2 Misc 5

7 K-means 5

8 PCA 5

9 Time Series 6

9.1 Stationarity 6

9.2 ARCH 6

10 Gaussian Processes 6

11 Neural Networks 6

11.1 Hyperparameters 6

11.2 Bias/Variance 6

12 Variational Methods 6

A Math Stuff 8

A.1 Matrix Stuff 8

A.1.1 Partitioned Matrix Inversion 8

A.2 Gaussians 8
A.2.1 Partitioned Gaussians: Conditional and Marginal Distributions 8
A.2.2 Bayes' theorem for Gaussian variables 8
A.3 Matrix Factorization 9
A.3.1 Diagonalization (Eigendecomposition) 9
A.3.2 Singular value decomposition 9
A.4 Matrix derivatives 9

B The Linear Regression Hierarchy 9
B.1 Maximum likelihood 10
B.2 Maximum a posteriori 10
B.3 Full Bayesian (stationary prior) 10
B.4 Variational Bayes (non-stationary prior) 10

1 Conventions

Math Notation

- $x \in \{0,1\}^r$: x is a vector of form *e.g.* $(0,1,1,0,\ldots,1,0)$ of length r.
- $\sim$: random variable drawn from a distribution
  $\propto$: "proportional to"
- $\mathbf{1}(\cdot)$: indicator function — 1 when arg is true, 0 when arg is false.
- Boldface capital letters are matrices, *e.g.* $\mathbf{A} = \mathbf{UWV}^T$
- log is base e by default → entropy in nats.

Ng CS229 / DeepLearning.ai

- m : number of training examples in the dataset
- n : dimension of training examples
- $x^{(i)} \in \mathbb{R}^n$: i^{th} training example (*column* vector), $1 \leq i \leq m$
- $y^{(i)}$: i^{th} output (*column* vector), $1 \leq i \leq m$
- x_j : j^{th} component of a training example, $1 \leq j \leq n$.
- $\mathbf{X} \in \mathbb{R}^{n \times m}$: (input/design) matrix (training examples are *column* vectors)[1]
- $\mathbb{X} \in \mathbb{R}^{m \times n}$: (input/design) matrix (training examples are *row* vectors)

[1] Note our non-conventional definition of the design matrix $\mathbf{X}$. The more conventional version is denoted $\mathbb{X}$.

2 Linear Regression

2.1 Basics

- Hypothesis $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots = \sum_{j=0}^{n} \theta_j x_j \equiv \theta^T x$, where $x_0 = 1$.
- Cost function

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 \tag{2.1}$$

- Analytically solve via normal equations, where $\mathbf{X}$ is the augmented $(n+1) \times m$ design matrix

$$\vec{\theta} = \left(\mathbf{X}\mathbf{X}^T \right)^{-1} \mathbf{X} \vec{y} \tag{2.2}$$

  where $\vec{\theta}$ and $\vec{y}$ are column vectors of the $n+1$ weights and m outputs, respectively.

3 Logistic Regression

3.1 2 classes

With 2 labels $y \in \{0, 1\}$ and m training examples $x^{(i)}, \;\; 1 \leq i \leq m$, we have

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}} \tag{3.1}$$

$$p(y|x; \theta) = (h_\theta(x))^y (1 - h_\theta(x))^{1-y} \tag{3.2}$$

$$L(\theta) = \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1 - h_\theta(x^{(i)}))^{1-y^{(i)}} \tag{3.3}$$

$$l(\theta) = \log L(\theta) = \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \tag{3.4}$$

with deriv

$$\frac{\partial}{\partial \theta_j} l(\theta) = x_j^T (y - h_\theta(x)) \tag{3.5}$$

We can't minimize the log-likelihood analytically, so we must use numerical optimization.

The log-likelihood is just the cross entropy

$$H(p, q) = -\sum_i p_i \log q_i \tag{3.6}$$

with p the actual outputs, and q the hypothesis $h_\theta(x)$

3.2 2+ classes

One-versus-all: For k classes, do k normal log regs. Each has two classes: the target class, and all the rest. New examples are classified by whichever of these k log regs. has highest score.

Softmax regression: Model classification cases as *multinomial* distribution. For k classes, hypothesis is k-dim vector

$$h_\theta(x) = \frac{1}{\sum_{l=1}^{k} \exp(\theta^{(l)T} x)} \times \left[\exp(\theta^{(1)T} x), \ldots, \exp(\theta^{(k)T} x)\right] \tag{3.7}$$

so that the final output layer has k units. The log-likelihood is

$$l(\theta) = -\sum_{i=1}^{m} \sum_{l=1}^{k} \mathbf{1}(y^{(i)} = l) \log \frac{\exp(\theta^{(l)T} x)}{\sum_{m=1}^{k} \exp(\theta^{(m)T} x)} \tag{3.8}$$

TODO: show how 2-class log reg cost func can be written this way. write down deriv of cost func. clean up notation.

4 Gaussian Discriminant Analysis (GDA) and Naive Bayes

Generative algorithm: Instead of modeling $p(y|x)$, model $p(y)$ and $p(x|y)$, then get posterior $p(y|x)$ via Bayes' theorem.

4.1 GDA

Can use GDA for classification problem when input features are continuous-val random vars. Models $p(x|y)$ as multivariate Gaussian. Model is

$$\begin{aligned} y &\sim \text{Bernoulli}(\phi) \\ x|y = 0 &\sim \mathcal{N}(\mu_0, \mathbf{\Sigma}) \\ x|y = 1 &\sim \mathcal{N}(\mu_1, \mathbf{\Sigma}) \end{aligned} \tag{4.1}$$

TODO: finish —

4.2 Naive Bayes (for text classification)

For training examples $x \in \{0, 1\}^n$ (a ***vocabulary***[2] of length n), assume the components of an example, x_j, are conditionally independent given y (***Naive Bayes assumption***).

$$p(x_1, \ldots, x_n|y) = \prod_{i=1}^{n} p(x_i|y) \tag{4.2}$$

[2] the j^{th} entry is 1 if the example contains the j^{th} word in the vocabulary. **Example**: If the vocab is $\{$cats, rats, bats$\}$, then $n = 3$, and a training example that contains “cats”, “rats”, but not “bats”, would be $x = (1, 1, 0)$.

Model is parameterized by

$$\phi_y = p(y=1), \qquad \phi_{j|y=1} = p(x_j = 1|y=1), \qquad \phi_{j|y=0} = p(x_j = 1|y=0) \tag{4.3}$$

where $j \in (1,n)$, for a total of $2n+1$ parameters. Likelihood is

$$L(\phi_y, \{\phi_{j|y=1}, \phi_{j|y=0}\}_{j=1}^n) = \prod_{i=1}^m p(x^{(i)}, y^{(i)}) \tag{4.4}$$

which has MLE values

$$\begin{aligned} \phi_{j|y=1} &= \text{fraction of spam (y=1) in which word j appears} \\ \phi_{j|y=0} &= \text{fraction of non-spam (y=0) in which word j appears} \\ \phi_y &= \text{ fraction of training examples that are spam} \end{aligned} \tag{4.5}$$

To make predictions, we don't need the evidence $p(x)$; we just need to compare

$$\begin{aligned} p(y=1|x) &\propto p(x|y=1)p(y=1) \\ p(y=0|x) &\propto p(x|y=0)p(y=0) \end{aligned} \tag{4.6}$$

and pick the class that has the higher value (un-normalized posterior).

TODO: finish — show how last two eqs are actually calculated

5 Bias/Variance Tradeoff

- **High bias**: underfitting. high training error and test error.
- **High variance**: overfitting. *low* training error and *high* test error

Can't use the test set to decide on values of hypers (*e.g.* number of parameters in model) because "we could just tweak the values of the hypers until the estimator performs optimally [on the test set]".

Cross-validation allows you to do model scoring (on frequentist stats, versus *e.g.* info criteria for Bayesian stats.. **TODO: verify i know what i'm talking about**).

Can average together multiple models with high capacity (low bias, but high variance). The result of combining is to *reduce* the variance of the combined model.

6 Model Assessment

6.1 Classification

6.1.1 Precision/Recall

For 2 classes, with $\{\text{TN}, \text{FN}, \text{FP}, \text{TP}\} = \{\text{true negatives, false negatives, false positives, true positives}\}$, we have the ***loss*** or ***confusion matrix***:

$$\begin{pmatrix} \text{TN} & \text{FP} \\ \text{FN} & \text{TP} \end{pmatrix}, \tag{6.1}$$

where the rows/columns are actual/predicted numbers of examples not-in-class (negative), or in-class (positive). We define the ***precision*** and ***recall***:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{6.2}$$

Precision is ability of classifier to not classify actual negatives as positive (*i.e. low* false positives). So *e.g.* if positive is "person doesn't have cancer", we want *high precision*.

Recall is ability of classifer to find all positive samples.

6.1.2 Receiver operating characteristic curve (ROC)

True Positive Rate (TPR): same as recall
True Negative Rate (TNR): fraction of actual negatives that are classified as negative TN/(TN + FP)
False Positive Rate (FPR): 1 - TNR

ROC plots TPR versus FPR.

TODO: F score

6.2 Misc

k: number of model parameters

- **Akaike Information Criteria (AIC):** $2k - 2\ln L(\theta)$
  TODO: how to apply in practice

7 K-means

- Setup: Select any K points from data to serve as initial centroids of the clusters.
- Repeat until (stopping criteria: *e.g.* no points change clusters, sum of distances is minimized, some total number of iterations..):
  1. Assign each datapoint to the cluster with closest centroid.
  2. Compute K new centroids, each the mean of the datapoints in its cluster.

8 PCA

PCA is unsupervised technique for finding directions of most variance in dataset. The ***principal components*** are a set of n orthonormal n-dim basis vects along the directions of maximum variance, ordered by decreasing variance in their directions.

To calc:

- **via eigenvectors:** Principal components of *de-meaned* $n{\times}m$ design matrix $\mathbf{X}$ are the eigenvectors of the covariance matrix of $\mathbf{X}$, $\mathbf{C}_\mathbf{X} = \frac{1}{m}\mathbf{X}\mathbf{X}^T$.

- **via SVD:** Define $\mathbf{Y} = \frac{1}{\sqrt{m}}\mathbf{X}^T$, and take its SVD A.3.2, which is $\mathbf{Y} = \mathbf{U}\mathbf{W}\mathbf{V}^T$. Then $\mathbf{U}^T$ is $n \times m$ matrix that projects $\mathbf{X}$ onto its principal components, $\mathbf{W}$ is the matrix of principal comp values, and $\mathbf{V}$ is orthonormal matrix of the principal components. **TODO: Verify/fix statement on U projection.**

9 Time Series

9.1 Stationarity

Definition 1 (Second Order Stationarity) *is when correlation between sequential observations is only a function of the lag.*

- **Dickey-Fuller test:** tests for stationarity of AR model. Null hypothesis is "series is *non-stationary*".

9.2 ARCH

- **Box-Jenkins:** systematic methodology for identifying and estimating models that can incorporate both AR and MA.

10 Gaussian Processes

11 Neural Networks

11.1 Hyperparameters

Hypers: learning rate, num iteractions, num hidden layers, num units in each layer, choice of activation function, amout of momentum, minibatch size, regularization type and amount.

11.2 Bias/Variance

Bias/variance is less of a *tradeoff* in neural network training, since we have a number of methods to reduce both independently.

- **Reduce bias (train set perf):** Bigger network, train longer, (NN arch search)
- **Reduce variance (dev set perf):** More data, regularization.. (when changed go back and retune bias)

12 Variational Methods

Preliminary section!

If our prior is not a conjugate prior to our likelihood, we can't compute posterior in closed form. Try:

- approximate intractable posterior $p(\vec{\theta}|\vec{y})$ with tractable $q(\vec{\theta}|\gamma)$.
- Adjust γ to minimize $KL\left[p(\vec{\theta}|\vec{y})||q(\vec{\theta}|\gamma)\right] \rightarrow$ complicated, since we don't know $p(\vec{\theta}|\vec{y})$. Eventually will use ***Expectation Propagation*** to solve.
- Adjust γ to minimize $KL\left[q(\vec{\theta}|\gamma)||p(\vec{\theta}|\vec{y})\right] \rightarrow$ simpler: ***Variational Bayes***.
- **Variational Bayes**: minimizing $KL\left[q(\vec{\theta}|\gamma)||p(\vec{\theta}|\vec{y})\right] \iff$ maximizing ELBO $\mathscr{L}$
- note the *latent/hidden variables* helping us are the $\vec{\theta}$

Appendix A Math Stuff

A.1 Matrix Stuff

A.1.1 Partitioned Matrix Inversion

Given block decomposition of matrix into submatrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$, the inverse is:

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{M} & -\mathbf{M}\mathbf{B}\mathbf{D}^{-1} \\ -\mathbf{D}^{-1}\mathbf{C}\mathbf{M} & \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\mathbf{M}\mathbf{B}\mathbf{D}^{-1} \end{pmatrix}, \tag{A.1}$$

where

$$\mathbf{M} = \left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} \tag{A.2}$$

(NB compare with Matt Headrick's compendium inversion formula)

A.2 Gaussians

A.2.1 Partitioned Gaussians: Conditional and Marginal Distributions

From BISHOP, sections 2.3.1 - 2.3.2.

Consider a joint Gaussian distribution, $\mathcal{N}(\vec{x}|\vec{\mu}, \mathbf{\Sigma})$ with $\mathbf{\Lambda} \equiv \mathbf{\Sigma}^{-1}$ and

$$\vec{x} = \begin{pmatrix} \vec{x}_a \\ \vec{x}_b \end{pmatrix}, \quad \vec{\mu} = \begin{pmatrix} \vec{\mu}_a \\ \vec{\mu}_b \end{pmatrix} \tag{A.3}$$

$$\mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_{aa} & \mathbf{\Sigma}_{ab} \\ \mathbf{\Sigma}_{ba} & \mathbf{\Sigma}_{bb} \end{pmatrix}, \quad \mathbf{\Lambda} = \begin{pmatrix} \mathbf{\Lambda}_{aa} & \mathbf{\Lambda}_{ab} \\ \mathbf{\Lambda}_{ba} & \mathbf{\Lambda}_{bb} \end{pmatrix}. \tag{A.4}$$

(That is, the vector $\vec{x}$ is partitioned into $\vec{x}_a$ and $\vec{x}_b$, which induces a block partitioning of the covariance matrix $\mathbf{\Sigma}$.)

The *conditional distribution* is:

$$p(\vec{x}_a|\vec{x}_b) = \mathcal{N}(\vec{x}|\vec{\mu}_{a|b}, \mathbf{\Lambda}_{aa}^{-1}) \tag{A.5}$$

$$\vec{\mu}_{a|b} = \vec{\mu}_a - \mathbf{\Lambda}_{aa}^{-1}\mathbf{\Lambda}_{ab}(\vec{x}_b - \vec{\mu}_b) \tag{A.6}$$

The *marginal distribution* is:

$$p(\vec{x}_a) = \mathcal{N}(\vec{x}_a|\vec{\mu}_a, \mathbf{\Sigma}_{aa}) \tag{A.7}$$

A.2.2 Bayes' theorem for Gaussian variables

From BISHOP, section 2.3.3. NB This is known as a *linear Gaussian model*.

Given a marginal Gaussian distribution for $\vec{x}$ and a conditional Gaussian distribution for $\vec{y}$ given $\vec{x}$ in the form:

$$p(\vec{x}) = \mathcal{N}(\vec{x}|\vec{\mu}, \mathbf{\Lambda}^{-1}) \tag{A.8}$$

$$p(\vec{y}|\vec{x}) = \mathcal{N}(\vec{y}|\mathbf{A}\vec{x} + \vec{b}, \mathbf{L}^{-1}), \tag{A.9}$$

we have

$$p(\vec{y}) = \mathcal{N}(\vec{y}|\mathbf{A}\vec{\mu} + \vec{b}, \mathbf{L}^{-1} + \mathbf{A}\mathbf{\Lambda}^{-1}\mathbf{A}^T) \tag{A.10}$$

$$p(\vec{x}|\vec{y}) = \mathcal{N}(\vec{x}|\mathbf{\Sigma}\{\mathbf{A}^T\mathbf{L}(\vec{y} - \vec{b}) + \mathbf{\Lambda}\vec{\mu}\}, \mathbf{\Sigma}) \tag{A.11}$$

where

$$\mathbf{\Sigma} \equiv \left(\mathbf{\Lambda} + \mathbf{A}^T\mathbf{L}\mathbf{A}\right)^{-1} \tag{A.12}$$

A.3 Matrix Factorization

A.3.1 Diagonalization (Eigendecomposition)

A square $N \times N$ symmetric matrix $\mathbf{A}$ has real eigenvalues from its symmetricity, and N linearly independent eigenvectors. We can write $\mathbf{A} = \mathbf{ODO}^{-1} = \mathbf{ODO}^T$, with $\mathbf{O}$ an orthogonal[3] matrix with columns equal to $\mathbf{A}$'s eigenvectors, and $\mathbf{D}$ a diagonal matrix of $\mathbf{A}$'s eigenvalues.

A.3.2 Singular value decomposition

An $M \times N$ matrix $\mathbf{A}$ can be written as $\mathbf{UDV}^T$, where $\mathbf{U}$ is $M \times N$, $\mathbf{D}$ is $N \times N$ and diagonal, and $\mathbf{V}$ is $N \times N$ and orthogonal.

If $\mathbf{A}$ is square, the columns of $\mathbf{V}$ are its eigenvectors, and $\mathbf{D}$ are its eigenvalues.

Uses:

- If $M \geq N$, cols of $\mathbf{U}$ are an orthonormal basis for space spanned by cols of $\mathbf{A}$

TODO: finish —

A.4 Matrix derivatives

$$\frac{\partial}{\partial \vec{x}}\left(\vec{x}^T\vec{a}\right) = \frac{\partial}{\partial \vec{x}}\left(\vec{a}^T\vec{x}\right) = \vec{a} \tag{A.13}$$

TODO: review understanding of this (I recall this expression was not enough to answer my questions on a formula derivation) and give examples.

Appendix B The Linear Regression Hierarchy

Linear regression starting from maximum likelihood, through variational Bayes.

[3] An *orthogonal* matrix should really be called an *orthonormal* matrix, as its columns are linearly indepdendent, but also normalized.

B.1 Maximum likelihood

Maximize likelihood $p(y|\theta;\beta)$ wrt θ, β.

B.2 Maximum a posteriori

Introduce prior $p(\theta;\alpha) = \exp(-\alpha\theta^T\theta)$. Find θ by maximizing $p(y|\theta;\beta)p(\theta;\alpha)$ wrt θ.
Q: what is proper way to estimate β and α here?

B.3 Full Bayesian (stationary prior)

Calc posterior $p(\theta|y) = p(y|\theta;\beta)p(\theta;\alpha)/p(y;\alpha,\beta)$, where $p(y;\alpha,\beta) = \int_\theta p(y|\theta;\beta)p(\theta;\alpha)$ is the evidence. Determine values of hypers α, β by maximizing evidence. This is difficult to do directly, as derivs wrt α, β are hard to compute, so use EM algorithm.

B.4 Variational Bayes (non-stationary prior)