Sparsity

- Vector $w$ is sparse, if many entries are zero:

- Very useful for many tasks, e.g.,
  - **Efficiency**: If size($w$) = 100B, each prediction is expensive:
    - If part of an online system, too slow
    - If $w$ is sparse, prediction computation only depends on number of non-zeros
  - **Interpretability**: What are the relevant dimension to make a prediction?
    - E.g., what are the parts of the brain associated with particular words?

![Image](image.png)
Regularization in Linear Regression

- Overfitting usually leads to very large parameter choices, e.g.:
  \[-2.2 + 3.1 X - 0.30 X^2\]
  \[-1.1 + 4,700,910.7 X - 8,585,638.4 X^2 + \ldots\]

- Regularized or penalized regression aims to impose a "complexity" penalty by penalizing large weights
  - "Shrinkage" method
    - L2 regularization

LASSO Regression

- LASSO: least absolute shrinkage and selection operator

- New objective:
  \[
  \min_w \sum_{j=1}^{N} \left( f(x_j) - \left( w_0 + \sum_{i} w_i h_i(x_j) \right) \right)^2 + \lambda \sum_{i=1}^{K} |w_i|
  \]

  - Don't regularize \( w_0 \)
Coordinate Descent for LASSO (aka Shooting Algorithm)

- Repeat until convergence
  - Pick a coordinate $j$ at (random or sequentially)
    - Set: $\tilde{w}_\ell = \begin{cases} 
    \frac{(c_\ell + \lambda)}{a_\ell} & c_\ell < -\lambda \\
    0 & c_\ell \in [-\lambda, \lambda] \\
    \frac{(c_\ell - \lambda)}{a_\ell} & c_\ell > \lambda 
  \end{cases}$
    - Where: $a_\ell = 2 \sum_{j=1}^{N} (w_j(x_j))^2$
      $c_\ell = 2 \sum_{j=1}^{N} h_j(x_j) \left(t(x_j) - (w_0 + \sum_{i=1}^{\ell} w_i h_i(x_j))\right)$

- For convergence rates, see Shalev-Shwartz and Tewari 2009

- Other common technique = LARS
  - Least angle regression and shrinkage, Efron et al. 2004

Recall: Ridge Coefficient Path

- Typical approach: select $\lambda$ using cross validation

From Kevin Murphy textbook
Now: **LASSO Coefficient Path**

![Graph showing LASSO Coefficient Path](image)

From Kevin Murphy textbook

---

**LASSO Example**

<table>
<thead>
<tr>
<th>Term</th>
<th>Least Squares</th>
<th>Ridge</th>
<th>Lasso</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>2.465</td>
<td>2.452</td>
<td>2.468</td>
</tr>
<tr>
<td>lcavol</td>
<td>0.680</td>
<td>0.420</td>
<td>0.533</td>
</tr>
<tr>
<td>lweight</td>
<td>0.263</td>
<td>0.238</td>
<td>0.169</td>
</tr>
<tr>
<td>age</td>
<td>-0.141</td>
<td>-0.046</td>
<td></td>
</tr>
<tr>
<td>lbph</td>
<td>0.210</td>
<td>0.162</td>
<td>0.002</td>
</tr>
<tr>
<td>svi</td>
<td>0.305</td>
<td>0.227</td>
<td>0.094</td>
</tr>
<tr>
<td>lcp</td>
<td>-0.288</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>gleason</td>
<td>-0.021</td>
<td>0.040</td>
<td></td>
</tr>
<tr>
<td>pgg45</td>
<td>0.267</td>
<td>0.133</td>
<td></td>
</tr>
</tbody>
</table>

From Rob Tibshirani slides
What you need to know

- Variable Selection: find a sparse solution to learning problem
- $L_1$ regularization is one way to do variable selection
  - Applies beyond regressions
  - Hundreds of other approaches out there
- LASSO objective non-differentiable, but convex ➔ Use subgradient
- No closed-form solution for minimization ➔ Use coordinate descent
- Shooting algorithm is very simple approach for solving LASSO

Classification
Logistic Regression

Machine Learning – CSE446
Carlos Guestrin
University of Washington

April 15, 2013
THUS FAR, REGRESSION: PREDICT A CONTINUOUS VALUE GIVEN SOME INPUTS

Weather prediction revisited

Temperature 63°F
Reading Your Brain, Simple Example

Pairwise classification accuracy: 85%

Person

Animal

Classification

- **Learn**: $h : \mathbf{X} \mapsto \mathbf{Y}$
  - $\mathbf{X}$ – features
  - $\mathbf{Y}$ – target classes

- Conditional probability: $P(\mathbf{Y}|\mathbf{X})$

- Suppose you know $P(\mathbf{Y}|\mathbf{X})$ exactly, how should you classify?
  - Bayes optimal classifier:

- **How do we estimate $P(\mathbf{Y}|\mathbf{X})$?**
Link Functions

- Estimating $P(Y|X)$: Why not use standard linear regression?

- Combing regression and probability?
  - Need a mapping from real values to $[0,1]$
  - A link function!

Logistic Regression

- Learn $P(Y|X)$ directly
  - Assume a particular functional form for link function
  - Sigmoid applied to a linear function of the input features:

$$P(Y = 0|X, W) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)}$$

Features can be discrete or continuous!
Understanding the sigmoid

\[ g(w_0 + \sum_i w_i x_i) = \frac{1}{1 + e^{w_0 + \sum_i w_i x_i}} \]

- \( w_0 = -2, w_i = -1 \)
- \( w_0 = 0, w_1 = -1 \)
- \( w_0 = 0, w_1 = -0.5 \)

Logistic Regression – a Linear classifier

\[ g(w_0 + \sum_i w_i x_i) = \frac{1}{1 + e^{w_0 + \sum_i w_i x_i}} \]
Very convenient!

\[ P(Y = 0 \mid X = < X_1, \ldots, X_n >) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)} \]
implies
\[ P(Y = 1 \mid X = < X_1, \ldots, X_n >) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)} \]
implies
\[ \frac{P(Y = 1 \mid X)}{P(Y = 0 \mid X)} = \exp(w_0 + \sum_i w_i X_i) \]
implies
\[ \ln \frac{P(Y = 1 \mid X)}{P(Y = 0 \mid X)} = w_0 + \sum_i w_i X_i \]

linear classification rule!

Loss function: Conditional Likelihood

- Have a bunch of iid data of the form:

- Discriminative (logistic regression) loss function:
  Conditional Data Likelihood

\[ \ln P(D_Y \mid D_X, w) = \sum_{j=1}^{N} \ln P(y^j \mid x^j, w) \]
Expressing Conditional Log Likelihood

\[ l(w) \equiv \sum_j \ln P(y^j | x^j, w) \]

\[ P(Y = 0|X, w) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)} \]

\[ P(Y = 1|X, w) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)} \]

\[ \ell(w) = \sum_j y^j \ln P(Y = 1|x^j, w) + (1 - y^j) \ln P(Y = 0|x^j, w) \]

Maximizing Conditional Log Likelihood

\[ l(w) \equiv \ln \prod_j P(y^j | x^j, w) \]

\[ P(Y = 0|X, W) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)} \]

\[ P(Y = 1|X, W) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)} \]

\[ l(w) = \sum_j y^j (w_0 + \sum_i w_i x_i^j) - \ln(1 + \exp(w_0 + \sum_i w_i x_i^j)) \]

**Good news:** \( l(w) \) is concave function of \( w \), no local optima problems

**Bad news:** no closed-form solution to maximize \( l(w) \)

**Good news:** concave functions easy to optimize
Optimizing concave function – Gradient ascent

Conditional likelihood for Logistic Regression is concave. Find optimum with gradient ascent

\[ \nabla_w l(w) = \left[ \frac{\partial l(w)}{\partial w_0}, \ldots, \frac{\partial l(w)}{\partial w_n} \right]^T \]

Gradient ascent is simplest of optimization approaches
- e.g., Conjugate gradient ascent can be much better

Maximize Conditional Log Likelihood: Gradient ascent

\[ l(w) = \sum_j y_j (w_0 + \sum_i w_i x_{ij}) - \ln(1 + \exp(w_0 + \sum_i w_i x_{ij})) \]
Gradient Ascent for LR

Gradient ascent algorithm: iterate until change < \( \varepsilon \)

\[
w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid x^j, w_0^0)]
\]

For \( i = 1, \ldots, n, \)

\[
w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid x^j, w_i^0)]
\]

repeat

---

Regularization in linear regression

- Overfitting usually leads to very large parameter choices, e.g.:

\[
-2.2 + 3.1 X - 0.30 X^2 \quad -1.1 + 4,700,910.7 X - 8,585,638.4 X^2 + ...
\]

- Regularized least-squares (a.k.a. ridge regression), for \( \lambda > 0 \):

\[
w^* = \arg \min_w \sum_j \left( t(x_j) - \sum_k w_k h_k(x_j) \right)^2 + \lambda \sum_{i=1}^k w_i^2
\]
Large parameters $\rightarrow$ Overfitting

- If data is linearly separable, weights go to infinity
- Leads to overfitting:

- Penalizing high weights can prevent overfitting...

Regularized Conditional Log Likelihood

- Add regularization penalty, e.g., $L_2$:

$$\ell(w) = \ln \prod_j P(y^j | x^j, w)) - \lambda ||w||^2_2$$

- Practical note about $w_0$:

- Gradient of regularized likelihood:
Standard v. Regularized Updates

- Maximum conditional likelihood estimate
  \[ w^* = \arg \max_w \ln \left[ \prod_{j=1}^N P(y^j | x^j, w) \right] \]
  \[ w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y_j - \hat{P}(Y^j = 1 | x^j, w)^{(0)}] \]

- Regularized maximum conditional likelihood estimate
  \[ w^* = \arg \max_w \ln \left[ \prod_j P(y^j | x^j, w) \right] - \lambda \sum_{i > 0} w_i^2 \]
  \[ w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\lambda w_i^{(t)} + \sum_j x_i^j [y_j - \hat{P}(Y^j = 1 | x^j, w)^{(0)}] \right\} \]

Please Stop!! Stopping criterion

\[ \ell(w) = \ln \prod_j P(y^j | x^j, w) - \lambda ||w||_2^2 \]

- When do we stop doing gradient descent?
  - Because \( l(w) \) is strongly concave:
    - i.e., because of some technical condition
      \[ \ell(w^*) - \ell(w) \leq \frac{1}{2\lambda} ||\nabla \ell(w)||_2^2 \]
    - Thus, stop when:
Stopping criterion

\[ \ell(w) = \ln \prod_j P(y^j | x^j, w) - \lambda \|w\|^2 \]

- Regularized logistic regression is strongly concave
  - Negative second derivative bounded away from zero:

- Strong concavity (convexity) is super helpful!!

- For example, for strongly concave \( l(w) \):
  \[ \ell(w^*) - \ell(w) \leq \frac{1}{2\lambda} \|\nabla \ell(w)\|^2 \]

Convergence rates for gradient descent/ascent

- Number of Iterations to get to accuracy
  \[ \ell(w^*) - \ell(w) \leq \epsilon \]

- If func Lipschitz: \( O(1/\epsilon^2) \)

- If gradient of func Lipschitz: \( O(1/\epsilon) \)

- If func is strongly convex: \( O(\ln(1/\epsilon)) \)
Digression: Logistic regression for more than 2 classes

- Logistic regression in more general case (k+1 classes), where \( Y \in \{y_1, \ldots, y_R\} \)

\[
P(Y = y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{n} w_{ki}X_i)}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}
\]

for \( k < R \)

\[
P(Y = y_R|X) = \frac{1}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}
\]

for \( k = R \) (normalization, so no weights for this class)

Learning procedure is basically the same as what we derived!
What you should know…

- Classification: predict discrete classes rather than real values
- Logistic regression model: Linear model
  - Logistic function maps real values to [0,1]
- Optimize conditional likelihood
- Gradient computation
- Overfitting
- Regularization
- Regularized optimization