02/29/2016

Weighted Least-Squares Regression

- Assign each sample a weight ω_i; collect them in an nxn diagonal matrix Ω.
- Greater $\omega_i \Rightarrow$ work harder to minimize $|\hat{y}_i - y_i|$. Recall: $\hat{y} = Xw$.
- Optimization problem:

$$\boxed{\text{Find } w \text{ that minimizes } (Xw-y)^T\Omega(Xw-y)}$$

- Note,

$$(Xw-y)^T\Omega(Xw-y) = \sum_{i=1}^{n} \omega_i((Xw)_i - y_i)^2$$

- Sove for w in normal equations:

$$X^T\Omega Xw = X^T\Omega y$$

- Note: $\Omega^{\frac{1}{2}}\hat{y}$ is orthogonal projections of $\Omega^{\frac{1}{2}}y$ onto subspace spanned by columns of $\Omega^{\frac{1}{2}}X$.

Newton's Method

- Iterative optimization method for smooth functions $J(w)$ often much faster than gradient descent.
- Idea: You're at point v in a space, where you're trying to optimize w. Approximate $J(w)$ near v by quadratic function. Jump to its unique critical point. Repeat until bored.
- Taylor series about v:

$$\nabla J(w) = \nabla J(v) + (\nabla^2 J(v))(w-v) + \mathcal{O}(|w-v|^2)$$

- $\nabla^2 J(v)(w-v)$ is the Hessian matrix of J at v.
- Find critical point w by setting $\nabla J(w) = 0$:

$$w = v - (\nabla^2 J(v))^{-1}\nabla J(v)$$

```
Newton's Method:
    while not "converged":
        e ← solution to linear system (∇²J(w))e = −∇J(w)
        w ← w + e
```

- Warning: Doesn't know difference between min, max or saddle points. Starting point must be "close enough" to desired solution.
- Facts:
  - If objective function J is actually a quadratic function, you'll jump to the minimum in one iteration. The closer J is to quadratic the faster it is, and vice versa.
  - Superior to blind gradient descent: 1) Doesn't just take a blind step length downhill it actually tries to figure out the bottom of the curve. 2) It doesn't just pick one direction of steepest descent it tries to optimize all at once.
  - Biggest disadvantage: Computing the Hessian is very expensive.

Logistic Regression

- Recall: $s'(\gamma) = s(\gamma)(1 - s(\gamma))$, $s_i = s(X_i \cdot w)$.
- $\nabla_w J(w) = \sum_{i=1}^{n}(y_i - s_i)X_i = X^T(y - s)$, where s is n-vector with components s_i.

$$\nabla^2 J(w) = -\sum_{i=1}^{n} s_i(1 - s_i)X_i X_i^T = X^T \Omega X$$

  where Ω is a diagonal matrix with components $s_i(1 - s_i)$.

- Ω is positive definite for all w,so $X^T\Omega X$ is positive semidefinite, so $-J(w)$ is convex.

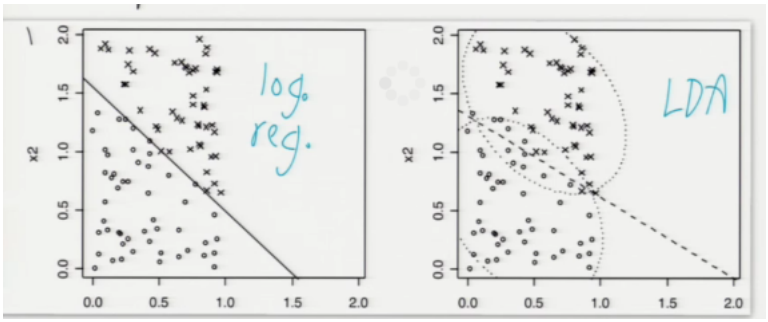
```
Newton's Method:
    while not "converged":
        Solve for e in normal equations: (X^TΩX)e = X^T(y − s)
        w ← w + e
```

Remember Ω, s are functions of w.

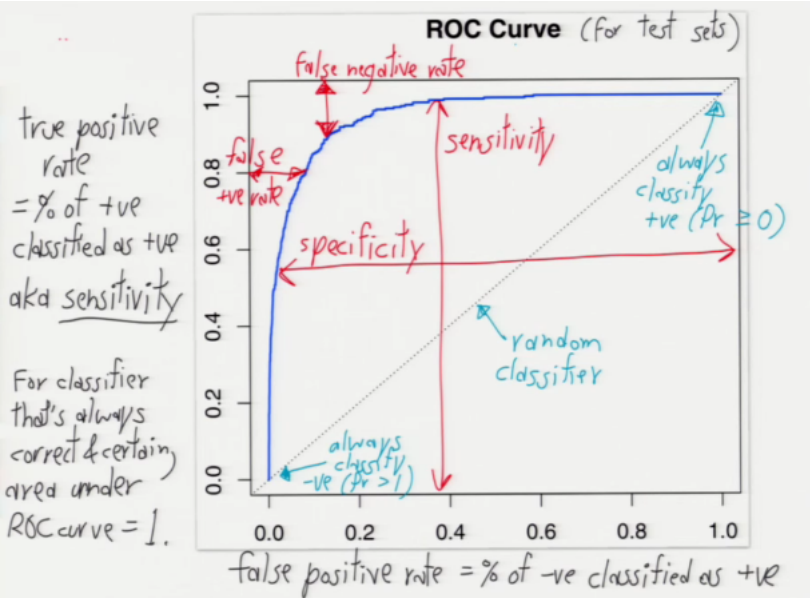
- An example of "iteratively re-weighted least squares."
- Ω prioritizes samples with s_i near 0.5; tunes out samples that are near 0/1.
- Idea: If n very large, save time by using a random subsample of the samples per iteration. Increase sample size as you go.

LDA vs. Logistic regression

- Advantages of LDA:
  - For well-separated classes, LDA stable; logistic regression surprisingly unstable.
  - For more than 2 classes easy & elegant, logistic regression needs modifying ("softmax regression").
  - Slightly more accurate when classes nearly normal, especially if n is small.
- Advantages of Logistic regression:
  - More emphasis on decision boundary.



  - Hence less sensitive to outliers.
  - Easy and elegant treatment of "partial" class membership; LDA is all-or-nothing.
  - More robust on some non-gaussian distributions (e.g. large skew).
- ROC Curve



- Receiver Operating Characteristics
- A way of telling you information about the test set, not part of the learning algorithm.
- Run classifier on test set, and it shows the rate of false positives vs. true positives for a range of settings.
- x-axis false positive rate = % of negative samples accidentally classified as positive.
- y-axis true positive rate = % of positive that actually got classified as positive (aka sensitivity).

Least-Squares Polynomial Regression

- Replace each X_i with feature vector $\Phi(X_i)$ with all terms of degree $0 \dots p$.
- e.g. $\phi(X_i) = \begin{bmatrix} X_{i1}^2 & X_{i1}X_{i2} & X_{i2}^2 & X_{i1} & X_{i2} & 1 \end{bmatrix}^T$
- Can also use non-polynomial features (e.g. edge detectors).
- Otherwise just like linear or logistic regression.
- Logistic regression plus quadratic features = same logistic posteriors as QDA. Very easy to overfit!