Warm up: risk prediction with logistic regression

- Boss gives you a bunch of data on loans defaulting or not:

\[ \{(x_i, y_i)\}_{i=1}^{n} \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\} \]

- You model the data as:

\[ P(Y = y|x, w) = \frac{1}{1 + \exp(-y w^T x)} \]

- And compute the maximum likelihood estimator:

\[ \hat{w}_{MLE} = \arg \max_{w} \prod_{i=1}^{n} P(y_i|x_i, w) \]

For a new loan application \( x \), boss recommends to give loan if your model says they will repay it with probability at least .95 (i.e. low risk):

Give loan to \( x \) if

\[ \frac{1}{1 + \exp(-\hat{w}_{MLE}^T x)} \geq .95 \]

- One year later only half of loans are paid back and the bank folds. What might have happened? Model wrong, finite data, data shift, massive class imbalance (e.g. no 0 class)
Projects

Proposal due Thursday 10/25

Guiding principles (for evaluation of project)
- Keep asking yourself “why” something works or not. Dig deeper than just evaluating the method and reporting a test error.
- Must use **real-world data** available NOW
- Must report **metrics**
- Must reference papers and/or books

- Study a real-world dataset
  - Evaluate multiple machine learning methods
  - Why does one work better than another? Form a hypothesis and test the hypothesis with a subset of the real data or, if necessary, synthetic data
- Study a method
  - Evaluate on multiple real-world datasets
  - Why does the method work better on one dataset versus another? Form a hypothesis…
Perceptron

Machine Learning – CSE546
Kevin Jamieson
University of Washington

October 23, 2018
Binary Classification

- **Learn:** \( f : X \rightarrow Y \)
  - \( X \) – features
  - \( Y \) – target classes
  \( Y \in \{-1, 1\} \)

- **Expected loss of \( f \):**
  \[
  \mathbb{E}_{X,Y} \left[ 1 \{ f(X) \neq Y \} \right] = \mathbb{E}_X \left[ \mathbb{E}_{Y|X} \left[ 1 \{ f(x) \neq Y \} | X = x \right] \right]
  \]
  \[
  \mathbb{E}_{Y|X} \left[ 1 \{ f(x) \neq Y \} | X = x \right] = 1 - P(Y = f(x) | X = x)
  \]

- **Bayes optimal classifier:**
  \[
  f(x) = \arg \max_y \mathbb{P}(Y = y | X = x)
  \]

- **Loss function:**
  \[
  \ell(f(x), y) = 1 \{ f(x) \neq y \}
  \]
Binary Classification

- Learn: \( f : X \rightarrow Y \)
  - \( X \) – features
  - \( Y \) – target classes
    \[ Y \in \{-1, 1\} \]

- Expected loss of \( f \):
  \[
  \mathbb{E}_{XY}[\mathbf{1}\{f(X) \neq Y\}] = \mathbb{E}_X[\mathbb{E}_Y|X|\mathbf{1}\{f(x) \neq Y\}|X = x]]
  \]
  \[
  \mathbb{E}_Y|X|\mathbf{1}\{f(x) \neq Y\}|X = x] = 1 - P(Y = f(x)|X = x)
  \]

- Bayes optimal classifier:
  \[ f(x) = \arg\max_y \mathbb{P}(Y = y|X = x) \]

- Model of logistic regression:
  \[ P(Y = y|x, w) = \frac{1}{1 + \exp(-yw^Tx)} \]

- Loss function:
  \[ \ell(f(x), y) = \mathbf{1}\{f(x) \neq y\} \]

What if the model is wrong?
Can we do classification without a model of $\mathbb{P}(Y = y | X = x)$?
The Perceptron Algorithm

Classification setting: \( y \) in \(-1,+1\)

Linear model
- Prediction: \( \text{SIGN} (w^T x + b) \)

Training:
- Initialize weight vector: \( w = 0, \ b = 0 \)
- At each time step:
  - Observe features: \( x_t \)
  - Make prediction: \( \text{SIGN} (w^T x_t + b) = y_t \)
  - Observe true class: \( y_t \)
- Update model:
  - If prediction is not equal to truth

\[
\begin{bmatrix}
y_t \end{bmatrix} = \begin{bmatrix} w \\ b \end{bmatrix} + \begin{bmatrix} x_t \end{bmatrix} y_t
\]
The Perceptron Algorithm [Rosenblatt ‘58, ‘62]

- Classification setting: $y$ in $\{-1,+1\}$
- Linear model
  - Prediction: $\text{sign}(w^T x_i + b)$

- Training:
  - Initialize weight vector: $w_0 = 0, b_0 = 0$
  - At each time step:
    - Observe features: $x_k$
    - Make prediction: $\text{sign}(x_k^T w_k + b_k)$
    - Observe true class: $y_k$
  - Update model:
    - If prediction is not equal to truth
      $$
      \begin{bmatrix}
        w_{k+1} \\
        b_{k+1}
      \end{bmatrix} =
      \begin{bmatrix}
        w_k \\
        b_k
      \end{bmatrix} + y_k \begin{bmatrix}
        x_k \\
        1
      \end{bmatrix}
      $$
"the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence."

*The New York Times, 1958*
Linear Separability

- Perceptron guaranteed to converge if
  - Data linearly separable:

\[ y_t = \text{SIGN} \left( w^T x_t \right) \]

\[ w_{\text{new}} = w_{\text{old}} + x_t y_t \]
Perceptron Analysis: Linearly Separable Case

- Theorem [Block, Novikoff]:
  - Given a sequence of labeled examples: 
    \( (x_t, y_t) \)
  - Each feature vector has bounded norm: 
    \( \|x_t\| \leq R \)
  - If dataset is linearly separable: 
    \( w \) margin \( \gamma \)

- Then the number of mistakes made by the online perceptron on any such sequence is bounded by 
  \( \frac{R^2}{\gamma^2} \)
Beyond Linearly Separable Case

- Perceptron algorithm is super cool!
  - No assumption about data distribution!
    - Could be generated by an oblivious adversary, no need to be iid
  - Makes a fixed number of mistakes, and it’s done for ever!
    - Even if you see infinite data
Beyond Linearly Separable Case

- **Perceptron algorithm is super cool!**
  - No assumption about data distribution!
    - Could be generated by an oblivious adversary, no need to be iid
  - Makes a fixed number of mistakes, and it’s done for ever!
    - Even if you see infinite data

- **Perceptron is useless in practice!**
  - Real world not linearly separable
  - If data not separable, cycles forever and hard to detect
  - Even if separable may not give good generalization accuracy (small margin)
What is the Perceptron Doing???

- When we discussed logistic regression:
  - Started from maximizing conditional log-likelihood
    \[ P(Y = y | X = x) \]

- When we discussed the Perceptron:
  - Started from description of an algorithm
    Update \( w \) in some way.

- What is the Perceptron optimizing????
Linear classifiers – Which line is better?
Pick the one with the largest margin!
Pick the one with the largest margin!

Distance from $x_0$ to hyperplane defined by $x^T w + b = 0$?

$$\frac{w^T \hat{x}_0 + b = 0}{w^T x_0 = -b}$$

$$\|x_0 - \hat{x}_0\|_2 = \left| \frac{\omega^T (x_0 - \hat{x}_0)}{||w||_2} \right|$$

$$= \frac{1}{||w||_2} \left| w^T x_0 + b \right|$$
Pick the one with the largest margin!

Distance from $x_0$ to hyperplane defined by $x^T w + b = 0$?

If $\tilde{x}_0$ is the projection of $x_0$ onto the hyperplane then

$$||x_0 - \tilde{x}_0||_2 = |(x_0^T - \tilde{x}_0)^T \frac{w}{||w||_2}|$$

$$= \frac{1}{||w||_2} |x_0^T w - \tilde{x}_0^T w|$$

$$= \frac{1}{||w||_2} |x_0^T w + b|$$
Pick the one with the largest margin!

Distance of $x_0$ from hyperplane $x^T w + b$:
\[
\frac{1}{||w||_2} \{x_0^T w + b\}
\]

Optimal Hyperplane

\[
\max_{w,b} \gamma \\
\text{subject to } \frac{1}{||w||_2} y_i(x_i^T w + b) \geq \gamma \quad \forall i
\]
Pick the one with the largest margin!

Distance of $x_0$ from hyperplane $x^T w + b$:

$$\frac{1}{\|w\|_2} (x_0^T w + b)$$

Max Hyperplane

$$\max_{w,b} \gamma$$
$$\text{subject to } \frac{1}{\|w\|_2} y_i (x_i^T w + b) \geq \gamma \quad \forall i$$

Optimal Hyperplane (reparameterized)

$$\min_{w,b} \|w\|_2^2$$
$$\text{subject to } y_i (x_i^T w + b) \geq 1 \quad \forall i$$
Pick the one with the largest margin!

- Solve efficiently by many methods, e.g.,
  - quadratic programming (QP)
    - Well-studied solution algorithms
  - Stochastic gradient descent
  - Coordinate descent (in the dual)

\[
\begin{align*}
\min_{w, b} & \quad ||w||^2_2 \\
\text{subject to} & \quad y_i(x_i^T w + b) \geq 1 \quad \forall i
\end{align*}
\]
What if the data is still not linearly separable?

If data is linearly separable

\[
\min_{w,b} \frac{1}{\|w\|_2^2}
\]

\[
y_i(x_i^T w + b) \geq 1 \quad \forall i
\]
What if the data is still not linearly separable?

If data is linearly separable

\[
\min_{w,b} \|w\|_2^2 \\
y_i(x_i^T w + b) \geq 1 \quad \forall i
\]

If data is not linearly separable, some points don’t satisfy margin constraint:

\[
\min_{w,b} \|w\|_2^2 \\
y_i(x_i^T w + b) \geq 1 - \xi_i \quad \forall i \\
\xi_i \geq 0, \sum_{j=1}^{n} \xi_j \leq \nu
\]
What if the data is still not linearly separable?

If data is linearly separable

\[
\min_{w,b} \|w\|_2^2 \\
y_i(x_i^T w + b) \geq 1 \quad \forall i
\]

If data is not linearly separable, some points don’t satisfy margin constraint:

\[
\min_{w,b} \|w\|_2^2 \\
y_i(x_i^T w + b) \geq 1 - \xi_i \quad \forall i \\
\xi_i \geq 0, \sum_{j=1}^{n} \xi_j \leq \nu
\]

What are “support vectors?”
SVM as penalization method

- Original quadratic program with linear constraints:

\[
\begin{align*}
\min_{w,b} & \quad \|w\|_2^2 \\
y_i (x_i^T w + b) & \geq 1 - \xi_i \quad \forall i \\
\xi_i & \geq 0, \sum_{j=1}^{n} \xi_j \leq \nu
\end{align*}
\]
SVM as penalization method

- Original quadratic program with linear constraints:
\[
\min_{w,b} \frac{1}{2} \|w\|^2 + \frac{1}{\lambda} \sum \xi_i \\
y_i(x^T_i w + b) \geq 1 - \xi_i \quad \forall i \\
\xi_i \geq 0, \sum_{i=1}^{n} \xi_i \leq \nu
\]

- Using same constrained convex optimization trick as for lasso:

For any \( \nu \geq 0 \) there exists a \( \lambda \geq 0 \) such that the solution the following solution is equivalent:

\[
\sum_{i=1}^{n} \max\{0, 1 - y_i(b + x^T_i w)\} + \lambda \|w\|^2
\]
Machine Learning Problems

- Have a bunch of iid data of the form:
  \[
  \{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d \quad y_i \in \mathbb{R}
  \]

- Learning a model’s parameters:
  Each \( \ell_i(w) \) is convex.

**Hinge Loss:**
\[
\ell_i(w) = \max \{0, 1 - y_i x_i^T w\}
\]

**Logistic Loss:**
\[
\ell_i(w) = \log(1 + \exp(-y_i x_i^T w))
\]

**Squared error Loss:**
\[
\ell_i(w) = (y_i - x_i^T w)^2
\]

How do we solve for \( w \)? The last two lectures!
Perceptron is optimizing what?

Perceptron update rule:

\[
\begin{bmatrix}
  w_{k+1} \\
  b_{k+1}
\end{bmatrix} = \begin{bmatrix}
  w_k \\
  b_k
\end{bmatrix} + y_k \begin{bmatrix}
  x_k \\
  1
\end{bmatrix} \mathbf{1}\{y_i (b + x_i^T w) < 0\}
\]

SVM objective:

\[
\sum_{i=1}^{n} \max \{0, 1 - y_i (b + x_i^T w)\} + \lambda \|w\|^2_2 = \sum_{i=1}^{n} \ell_i (w, b)
\]

\[
\nabla_w \ell_i (w, b) = \begin{cases} 
  -y_i x_i + \frac{2\lambda}{n} w & \text{if} \quad l - y_i (b + x_i^T w) > 0 \\
  \frac{2\lambda}{n} w & \text{otherwise}
\end{cases}
\]

It updates at random (n).

\[w_{k+1} = w_k - \nabla_w \ell_{i_k} (w_k, b_k) \]

©2018 Kevin Jamieson
Perceptron is optimizing what?

Perceptron update rule:

\[
\begin{bmatrix}
w_{k+1} \\
b_{k+1}
\end{bmatrix} = \begin{bmatrix}
w_k \\
b_k
\end{bmatrix} + y_k \begin{bmatrix}
x_k \\
1
\end{bmatrix} \mathbf{1}\{y_i(b + x_i^T w) < 0\}
\]

SVM objective:

\[
\sum_{i=1}^{n} \max\{0, 1 - y_i(b + x_i^T w)\} + \lambda||w||_2^2 = \sum_{i=1}^{n} \ell_i(w, b)
\]

\[
\nabla_w \ell_i(w, b) = \begin{cases} 
-x_i y_i + \frac{2\lambda}{n} w & \text{if } y_i(b + x_i^T w) < 1 \\
0 & \text{otherwise}
\end{cases}
\]

\[
\nabla_b \ell_i(w, b) = \begin{cases} 
-y_i & \text{if } y_i(b + x_i^T w) < 1 \\
0 & \text{otherwise}
\end{cases}
\]

Perceptron is just SGD on SVM with \( \lambda = 0, \eta = 1! \)
SVMs vs logistic regression

- We often want probabilities/confidences, logistic wins here?
SVMs vs logistic regression

- We often want probabilities/confidences, logistic wins here?
- No! Perform isotonic regression or non-parametric bootstrap for probability calibration. Predictor gives some score, how do we transform that score to a probability?
SVMs vs logistic regression

- We often want probabilities/confidences, logistic wins here?
- No! Perform isotonic regression or non-parametric bootstrap for probability calibration. Predictor gives some score, how do we transform that score to a probability?

- For classification loss, logistic and svm are comparable
- Multiclass setting:
  - Softmax naturally generalizes logistic regression
  - SVMs have
- What about good old least squares?
What about multiple classes?