University of Illinois Fall 2018

ECE 544NA: Pattern Recognition
Lecture 3: September 4

Lecturer: Alexander Schwing Scribe: Yuncheng Wu

1 Context And Motivation

1.1 Problem

Consider the simplest case that uses linear regression for classification, where $y^{(i)} \in \{-1, 1\}$. The model is $y = w_1 x + w_0$ and threshold is at $y = 0$, which means if a data point has a feature x, and $w_1 x + w_0 < 0$, the data point will be classified as -1, otherwise it will be classified as 1.

1.1.1 Perfect classification

In Figure 1, which is the example of perfect classification, the pink triangle, which is the decision boundary, is at 0. The blue data points on the left side of decision boundary are classified as -1. On the other hand, the black data points on the right side of decision boundary are classified as 1. Note that the feature of blue data points is $x < 0$ and the feature of black data points is $x > 0$.

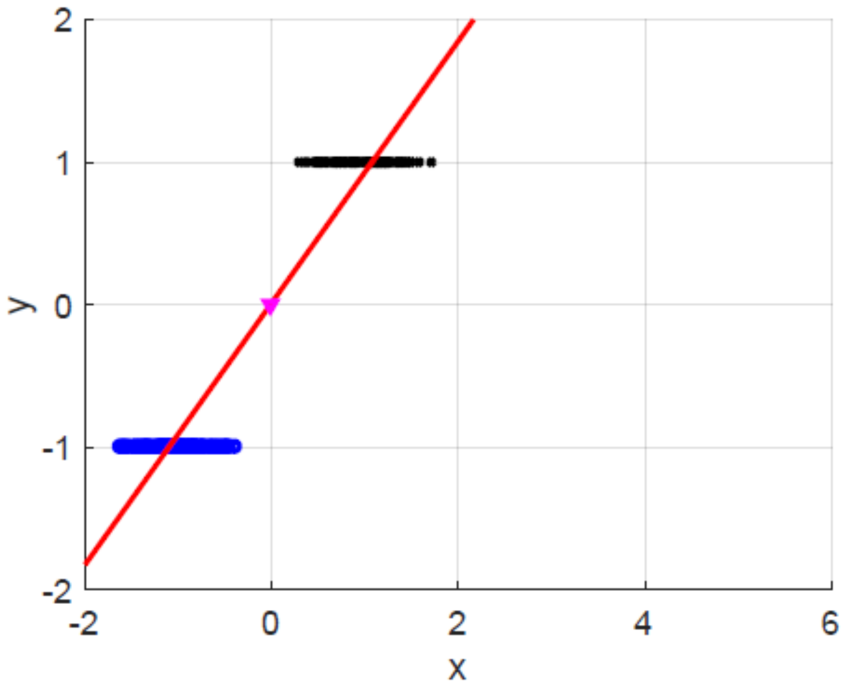

Figure 1: Perfect Classification

1.1.2 Imperfect classification

In Figure 2, we add another set of data points and fit the model with the new data set. In the plot, we can see that the pink triangle, representing the decision boundary, shifts right and thus larger than 0. Although the feature of black points does not change ($x > 0$), the shift of decision boundary makes some of black points on the left side, so they will be classified as -1. Such result shows that the new set of data points shifts the decision boundary, and leads to some incorrect classification results.

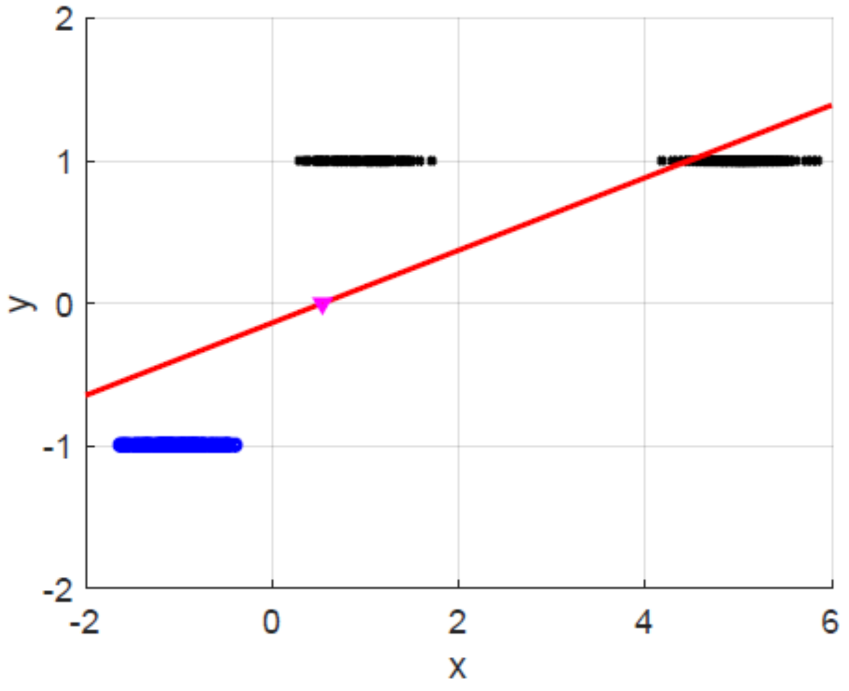

Figure 2: Imperfect Classification

1.2 Analysis

It is the quadratic loss we use in the linear regression results in decision boundary shifting. Recall $y^{(i)} \in \{-1, 1\}$. The loss function $\ell(y^{(i)}, \phi(x^{(i)})^\top \mathbf{w})$ is:

$$\begin{aligned}\ell(y^{(i)}, \phi(x^{(i)})^\top \mathbf{w}) &= \frac{1}{2}(y^{(i)} - \phi(x^{(i)})^\top \mathbf{w})^2 \\ &= \frac{1}{2}(y^{(i)} * y^{(i)} - y^{(i)} * \phi(x^{(i)})^\top \mathbf{w})^2 \\ &= \frac{1}{2}(1 - \underbrace{y^{(i)} \underbrace{\phi(x^{(i)})^\top \mathbf{w}}_{F(x^{(i)},\mathbf{w})}}_{F(x^{(i)},\mathbf{w},y^{(i)})})^2\end{aligned} \tag{1}$$

Note that in the above equation, $F(x^{(i)}, \mathbf{w}, y^{(i)})$ is the scoring function, and the classification result is computed by $F(x^{(i)}, \mathbf{w})$. If $F(x^{(i)}, \mathbf{w}) > 0$, the classification result is 1, otherwise the classification result is -1. In this way, if the classification result is the same as the actual result $y^{(i)}$, the result of scoring function is positive, otherwise, the result of scoring function is negative. Let's consider 3 cases:

- Case 1: $F(x^{(i)}) = 1$ and $y^{(i)} = 1$. In this case, the data point is classified correctly. The result of loss function is:

$$\frac{1}{2}(1 - y^{(i)}F(x^{(i)}, \mathbf{w}))^2 = \frac{1}{2}(1-1)^2 = 0$$

  In this way, correct classification does not pay any price.

- Case 2: $F(x^{(i)}) = 1$ and $y^{(i)} = -1$. In this case, the data point is classified incorrectly. The result of loss function is:

$$\frac{1}{2}(1 - y^{(i)}F(x^{(i)}, \mathbf{w}))^2 = \frac{1}{2}(1-(-1))^2 = 2$$

  In this way, incorrect classification pay some price.

- Case 3: $F(x^{(i)}) = 10$ and $y^{(i)} = 1$. In this case, the data point is classified correctly. The result of loss function is:

$$\frac{1}{2}(1 - y^{(i)}F(x^{(i)}, \mathbf{w}))^2 = \frac{1}{2}(1-10)^2 = \frac{81}{2}$$

  In this way, although classification result is correct, we pay a huge price.

Case 3 shows that this cost function penalizes samples that are 'very easy to classify'. We can also plot the loss function shown as Figure 3.

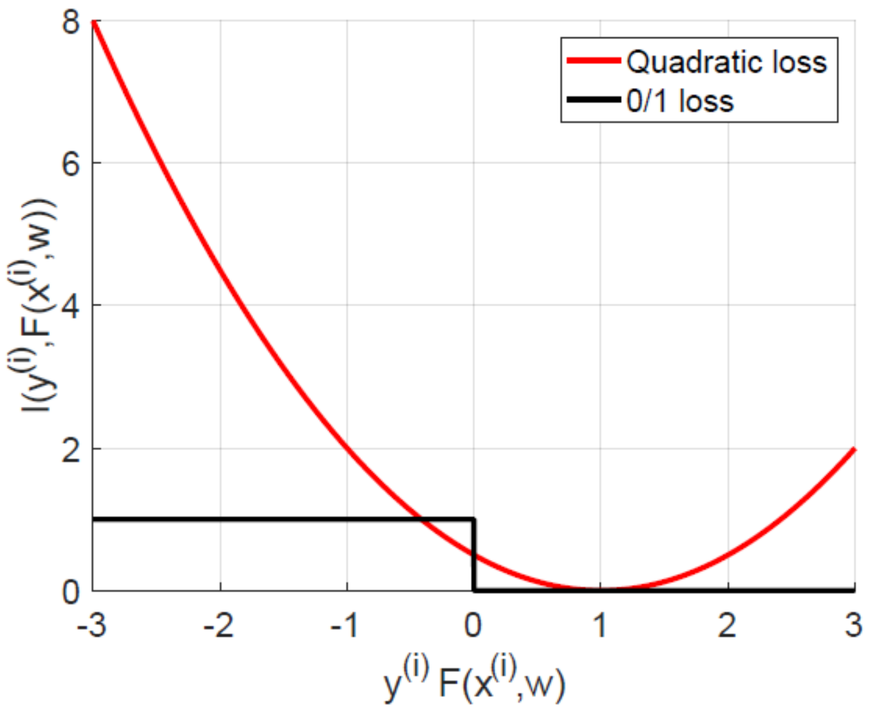

Figure 3: Quadratic Loss vs. 0/1 loss

Recall that one of the interpretations of linear regression is Gaussian distribution:

$$p(y^{(i)}|x^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\big(- \frac{1}{2\sigma^2}(y^{(i)} - \phi(x^{(i)})^\top \mathbf{w})^2\big) \tag{2}$$

We can change the distribution thus change our lost function.

2 Definition

We use another probabilistic formulation for classification ($y^{(i)} \in \{-1, 1\}$). Let's define the model:

$$p(y^{(i)} = 1|x^{(i)}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \phi(x^{(i)}))} \tag{3}$$

$$p(y^{(i)} = -1|x^{(i)}) = 1 - p(y^{(i)} = 1|x^{(i)}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \phi(x^{(i)}))} \tag{4}$$

We can combine (3) and (4) to get the model definition:

$$p(y^{(i)}|x^{(i)}) = \frac{1}{1 + \exp(-y^{(i)}\mathbf{w}^\top \phi(x^{(i)}))} \tag{5}$$

3 Loss function

3.1 Loss function of logistic regression

Suppose we are given a dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}$, and we assume data are drawn independently from an identical distribution (i.i.d assumption). We need to choose the model parameter $\mathbf{w}$ that maximizes the likelihood of this dataset. In mathematics form, we can write it as

$$\max_{\mathbf{w}} \prod_{(x^{(i)}, y^{(i)})} p(y^{(i)}|x^{(i)}) \tag{6}$$

Since $p(y^{(i)}|x^{(i)}) \leq 1$, the product of all probabilities is a very small number, leading to loss of precision. We can rewrite (6) into

$$\min_{\mathbf{w}} \sum_{(x^{(i)}, y^{(i)}) \in \mathcal{D}} -\log p(y^{(i)}|x^{(i)}) \tag{7}$$

Thus, by combining our model (5) and our task (7), we can build our loss function:

$$\min_{\mathbf{w}} \sum_{(x^{(i)}, y^{(i)}) \in \mathcal{D}} \log(1 + \exp(-y^{(i)}\mathbf{w}^\top \phi(x^{(i)}))) \tag{8}$$

Note that compared with (7) we removed "-" before the "log" in (8), because we are minimizing the denominator of original model (5).

3.2 Loss function comparison

Recall risk minimization of linear regression (1):

$$\min_{\mathbf{w}} \sum_{(x^{(i)},y^{(i)})\in\mathcal{D}} \frac{1}{2}(1 - y^{(i)} \underbrace{\phi(x^{(i)})^\top \mathbf{w}}_{F(x^{(i)},\mathbf{w})})^2 \tag{9}$$

We can write risk minimization of logistic regression (8) in a similar form:

$$\min_{\mathbf{w}} \sum_{(x^{(i)},y^{(i)})\in\mathcal{D}} \log(1 + \exp(-y^{(i)} \underbrace{\mathbf{w}^\top \phi(x^{(i)})}_{F(x^{(i)},\mathbf{w})})) \tag{10}$$

Thus, we can see that linear regression and logistic regression share the same risk minimization:

$$\min_{\mathbf{w}} \sum_{(x^{(i)},y^{(i)})\in\mathcal{D}} \ell(y^{(i)}, F(x^{(i)}, w)) \tag{11}$$

In this way, we can compare loss function of linear regression and that of logistic regression in the same function plot.

In Figure 4, we can see that the loss function of logistic regression no longer penalize items that are easy to classify. Let's consider the Case 3 in Section 1.2 again with risk minimization of logistic regression (10).

- Case 3: $F(x^{(i)} = 10$ and $y^{(i)} = 1$. In this case, the data point is classified correctly. The result of loss function is:

$$\frac{1}{2}(1 - y^{(i)}F(x^{(i)}, \mathbf{w}))^2 = \log(1 + \exp(-1 * 10)) \approx 1.97 * 10^{-5}$$

, which is very small.

4 Optimization

4.1 Naive approach: set gradient to 0

In linear regression, we set gradient of loss function to 0 and solve for the parameter $\mathbf{w}$. We can also try the same approach for logistic regression. By setting gradient of loss function to 0, we have:

$$\nabla_{\mathbf{w}} f(\mathbf{w}) = \sum_{(x^{(i)},y^{(i)})\in\mathcal{D}} \frac{-y^{(i)} \exp(-y^{(i)}\mathbf{w}\top\phi(x^{(i)}))\phi(x^{(i)})}{1 + \exp(-y^{(i)}\mathbf{w}\top\phi(x^{(i)}))} = 0 \tag{12}$$

However, there are no analytic solution for $\mathbf{w}$ in general. The naive approach does not work.

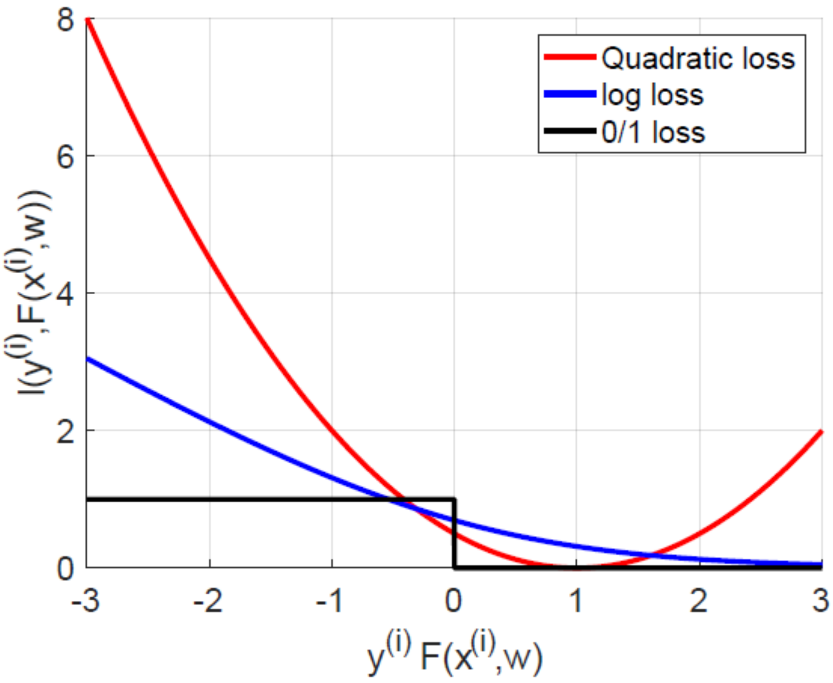

Figure 4: Quadratic Loss vs. log loss

4.2 Alternative approach: Gradient descent

Gradient descent is similar to walking down a mountain. At every step, make the function walk to the negative direction of gradient at that point. In the end, we can reach the minimum value of loss function and find out the parameter $\mathbf{w}$.

Figure 5 visualizes optimization process of loss function via gradient descent.
Here is the algorithm of gradient descent step by step:

1. Initialize step variable $t = 0$, step size α and initial value of parameter $\mathbf{w} = \mathbf{w}_t$
2. Calculate gradient at $\mathbf{w}_t$: $\mathbf{g}_t = \bigtriangledown_{\mathbf{w}} f(\mathbf{w}_t)$ (The same $f()$ in (12))
3. Update parameter $\mathbf{w_{t+1}} = \mathbf{w_t} - \alpha \mathbf{g_t}$
4. Update step variable: $t = t + 1$

Note the there are three options to set the step size α.

- Small fixed value: This is the simplest case and works very well.
- Line search to find out proper α, but line search is very expensive.
- Step size decay: Set α to some large number first, and decay it at each step.

5 Classification

Let's use logistic regression to classify datasets we used in Figure 1 and Figure 2. In Figure 6 we can see that the classification is still perfect when we add new data points.

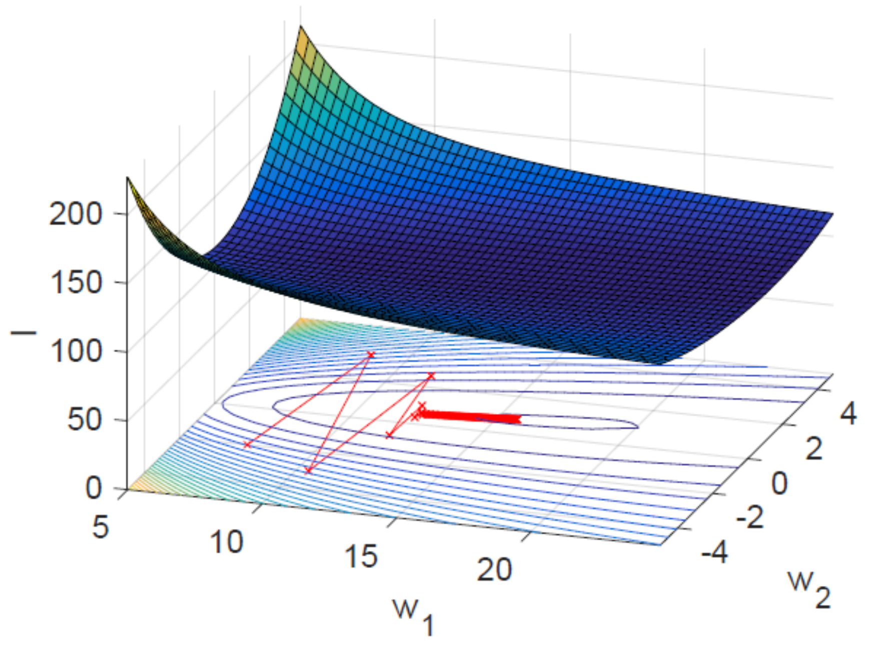

Figure 5: Optimization process of loss function via gradient descent

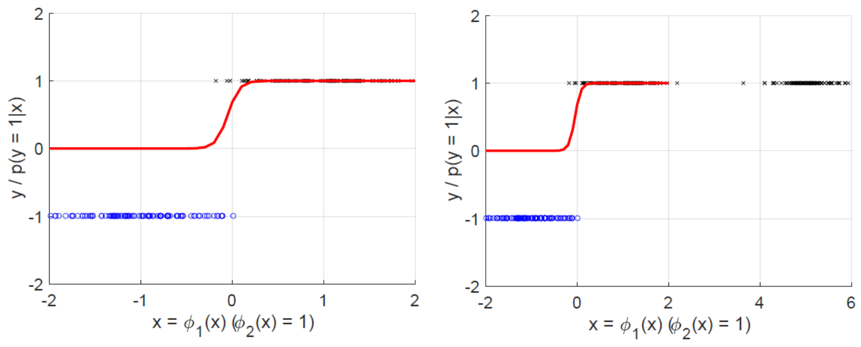

Figure 6: Classification via logistic regression

6 Comparison

Linear regression	**Logistic regression**
Closed form solution: we set derivative of loss function to 0 and find out model parameters	No closed form solution: we use gradient descent to find out model parameters that minimizes loss function
Gaussian probability model	Logistic probability model (binomial distribution used here)
Not too well suited for classification due to quadratic error	Well suited for binary classification

7 Application

7.1 Problem

One of the problems we met in edge detection is that intensity cue is not necessarily a good indicator for boundaries. For example, in Figure 7:

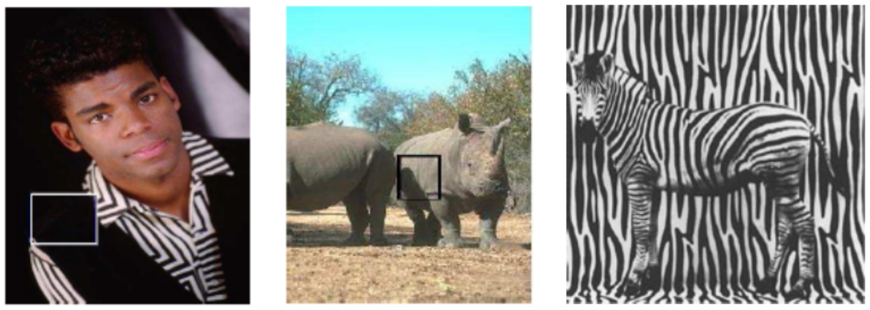

Figure 7: Edge Detection Issues

- The first picture does not have enough contrast between cloth and background, so it is hard to detect edge of the cloth.
- The second picture has shadow, which is an intensity change not on edge.
- The third picture has texture, which has strong intensity change, so it is hard to recognize if a intensity change is on texture or on edges.

7.2 Solution

We can use other image cues to detect edges. For example, we can use boundary gradient (BG), color gradient (CG) and texture gradient (TG). In Figure 8, we can see relationship between those cues and edges in the image.

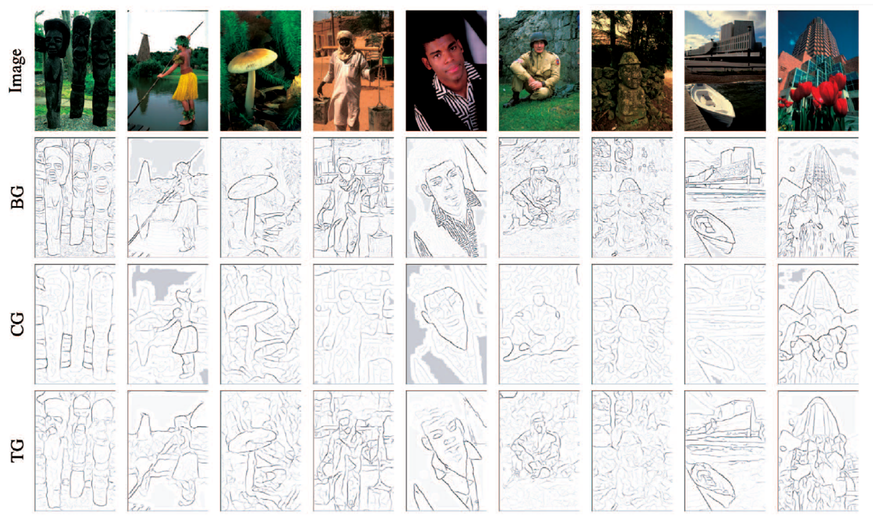

Figure 8: Image cues

To combine these cues to detect edges, we can learn a linear combination via logistic regression model. Here is the dataset we need to train our model:

1. $y^{(i)}$: Annotated pixel label, if a pixel is a pixel on edge
2. $x^{(i)}$: Image.
3. $\phi(x^{(i)})$: Vector of features computed in the nearby pixels. For example, BG, CG and TG that are mentioned above.

7.3 Result

As shown in Figure 9, Berkeley PB, '04 uses logistic regression to combine multiple image cues. The result shows that such approach works better than traditional Canny+Hysteresis edge detector.

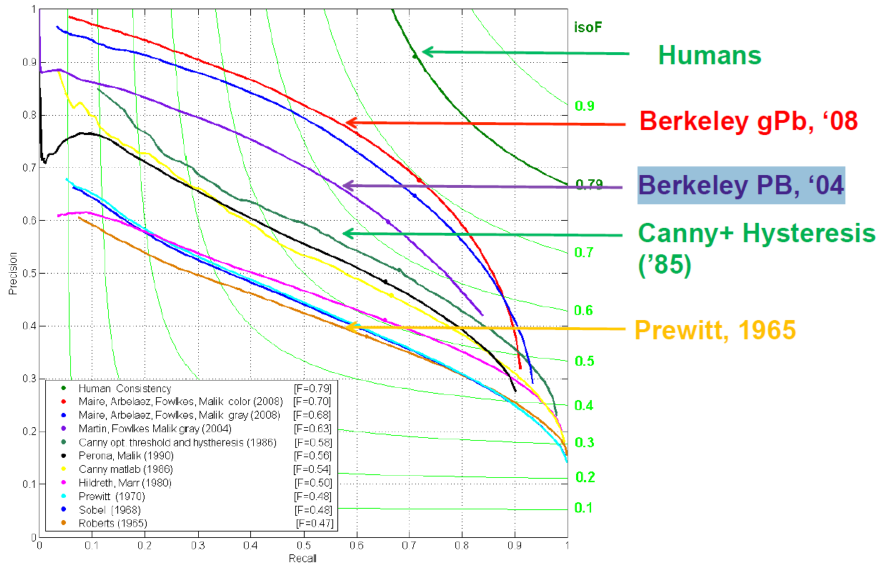

Figure 9: Edge detection results comparison

8 Quiz Questions

1. Which loss is used for logistic regression?
   - $\sum_{(x^{(i)},y^{(i)})\in\mathcal{D}} \log(1+\exp(-y^{(i)}\mathbf{w}^\top\phi(x^{(i)})))$
2. What is the difference between logistic and linear regression?
   - See Section 6 Comparison.
3. How to optimize linear and logistic regression?
   - Linear regression: Set gradient to 0 and solve
   - Logistic regression: Gradient descent

References