Logistic Regression

Generative vs Discriminative Classifiers

- Classification task is to find a function h: X -> Y from training sample.
- h is an approximation to the real classification function f
- Generative Classifiers: approximate the function by P(Y|X) based on:
  $P(Y|\mathbf{X}) \propto P(\mathbf{X}|Y)\, P(Y)$

  - **X** is a vector of features

  - First term $P(\mathbf{X}|Y)$ is likelihood and
    Second term P(Y) is prior

  - How do we get their values -> from training data.

  - For likelihood term, we need joint probability distribution

Generative vs Discriminative Classifiers

- What is joint probability distribution?
  Suppose you have 3 Boolean attributes
  X = (X1, X2, X3)
  and two classes Y = 0 and Y = 1
- For each class, you need the complete table filled out.
  How many entries do you need? $2^3 - 1 = 7$

X1	X2	X3	P (Y=i)
0	0	0	?
0	0	1	?
..	..	..	?
..	..	..	?
1	1	1	?

- For two classes, total number of values (parameters) needed = $2 * (2^3 - 1)$
- For n Boolean variables, parameters = $2 * (2^n - 1)$
- That's too many parameters to estimate. Can we do better?

Generative vs Discriminative Classifiers

- How does conditional independence help
  P(**X** | Y) = P(X1 | Y) * P(X2 | Y) * P(X3 | Y)

  e.g. for class Y = 1, if I know P(X1 = 1),
  I know P(X1 = 0). So, one parameter for
  each attribute per class.

Θ_{11} Θ_{21} Θ_{31}

P(X1 =1 | Y=1) P(X2 =1 | Y=1) P(X3 =1 | Y=1)

- In this case, we just need 3 parameters for each class. We needed 7 without conditional independence (CI) assumption.
- For n Boolean attributes and 2 classes, we need 2n parameters, if we make the conditional independence assumption. We needed 2 * (2^n – 1) without conditional independence (CI) assumption.

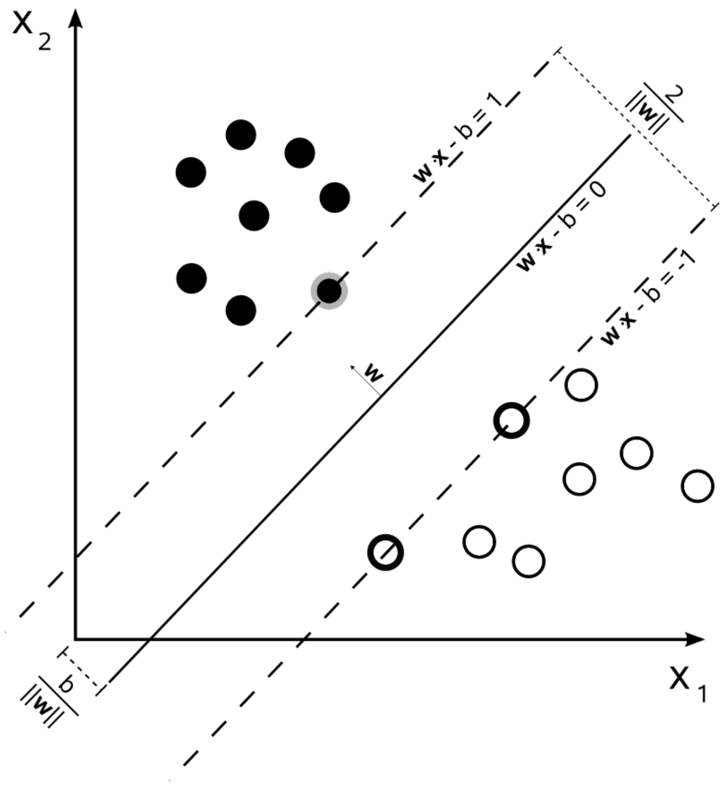
Generative vs Discriminative Classifiers

- Well, what if we make a functional model for probability:

$$P(Y=1 \mid \mathbf{X}) = h(\mathbf{X})$$

- Sounds like a good idea!
- What should be the form of h? How do we learn it from data?
- This is the focus of **discriminative classifiers.**
- Logistic regression is an example of discriminative classifiers.

Review of Types of Classifiers



3 types of classifiers:

1. Create a model for y (output) as a function of attributes. e.g. Perceptron, ANN, SVM
   $y = \text{sign}(w^T x)$

2. Probabilistic (Generative) classifiers
   $P(Y \mid X) \propto P(X \mid Y) * P(Y)$
   We estimate likelihood and prior from training data.

3. Discriminative classifiers –
   Create a model for P(Y|X) – Logistic Regression

Today's topic

Discriminative Classifiers

- Discriminative classifiers assume a functional form for P(Y|X).
- It can be shown that for discrete-valued Y and discrete or continuous X, naïve Bayes equation is equivalent to the following functional form for P(Y | X) :

$$P(Y = y_k \,|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

- Details of derivation are in the updated chapter of Tom Mitchell's book section 3.1.

Logistic Regression Learning Scenario for continuous attributes:

- Consider learning f: $X \rightarrow Y$, where
  - X is a vector of real-valued features, $< X_1 \ldots X_n >$
  - Y is boolean
  - assume all X_i are conditionally independent given Y
  - model $P(X_i \mid Y = y_k)$ as Gaussian $N(\mu_{ik},\sigma_i)$
  - model $P(Y)$ as Bernoulli (π)

- What does that imply about the form of $P(Y|X)$?

You can see the text for a complete derivation of this equation using above assumptions.

$$P(Y=1|X=<X_1,...X_n>) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

Logistic Regression

Note: Sometimes we may swap the equations for P(Y=1|X) and P(Y=0|X).

- Form of LR for Boolean Y:

$$P(Y = 1 \mid X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

- This is similar to the logistic function if we let: -z = $w_0 + \sum_{i=1}^{n} w_i X_i$

$$P(Y = 1 \mid z) = \frac{1}{1+\exp(-z)}$$

- Since P(Y=1 | X) + P(Y=0 |X) = 1

$$P(Y = 0 \mid X) = \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

Logit Function

- If we let P(Y=1|X) =p, and take the ratio of P(Y=0|X) and P(Y=1|X),

$$\frac{p}{1-p} = \exp(z)$$

- Take log of both sides:

$$\log\left(\frac{p}{1-p}\right) = z = \beta_0 + \sum_i \beta_i x_i$$

- The LHS of the function is also called the logit function

$$logit(p) = \beta_0 + \sum_i \beta_i x_i$$

Commonly used by most statistics textbooks

Logistic Regression

- We assume a functional form for learning P(Y|X)
  - logistic or sigmoid
    function
- Plot shown on the right.
- We have encountered this function before as an activation function for perceptrons.

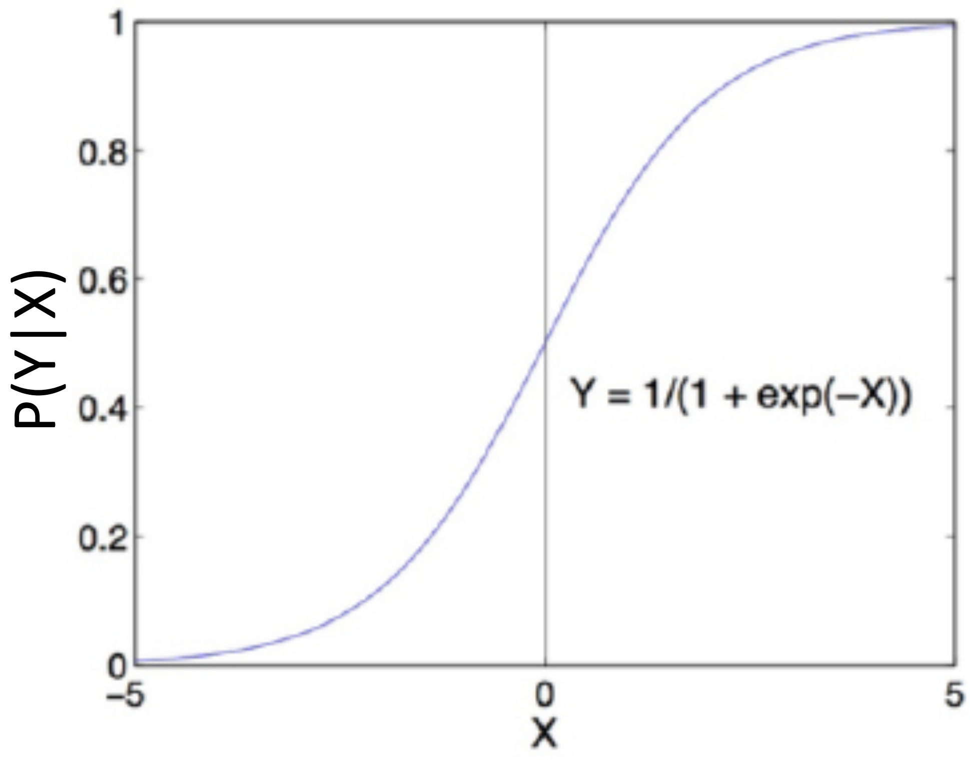


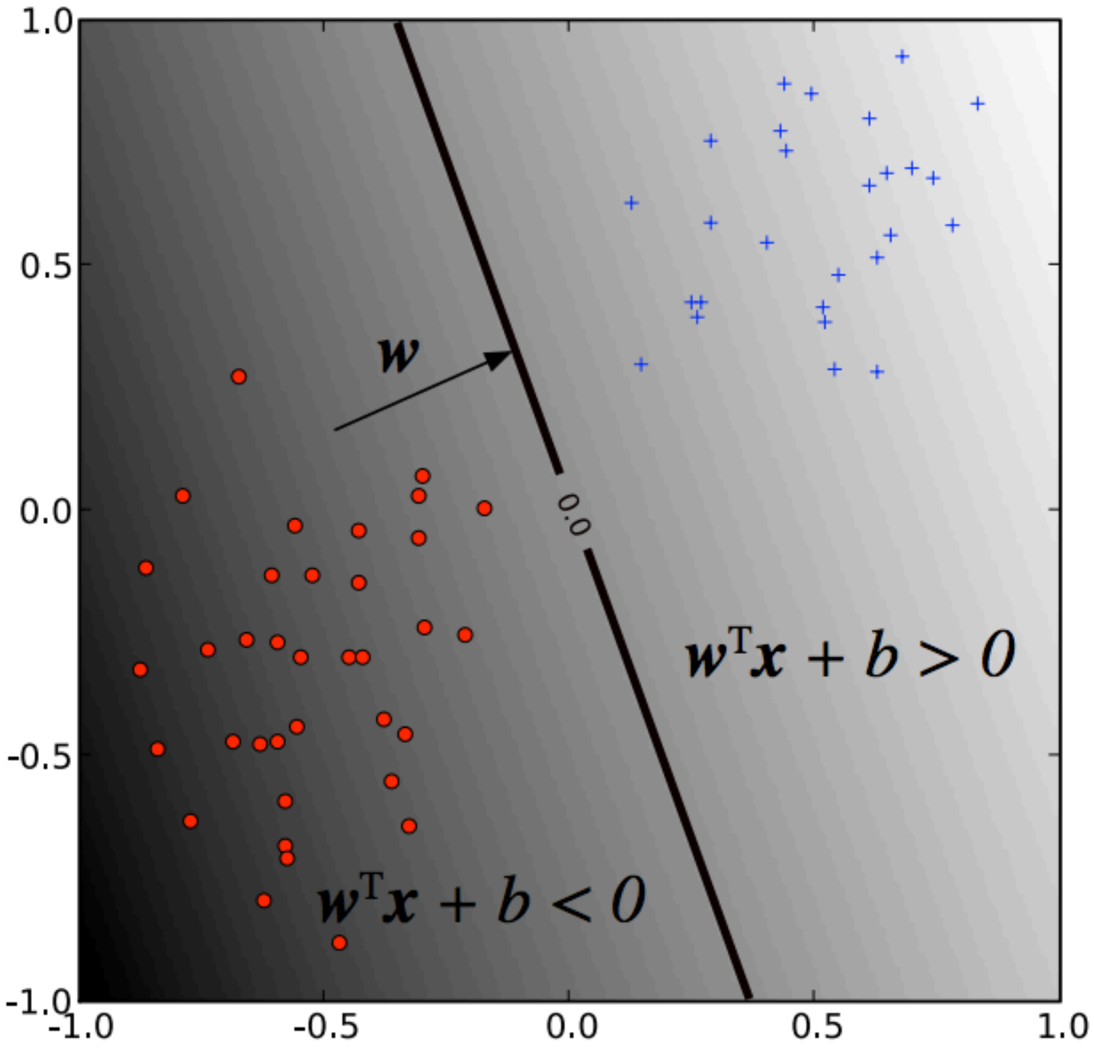
Figure 1: Form of the logistic function. In Logistic Regression, $P(Y|X)$ is assumed to follow this form.

Logistic Regression

- We are creating a model for P(Y|X), but what does the decision boundary in the attribute (e.g. x1, x2) plane look like.

Remember for linear classification:

- The separating hyperplane has the equation: $w^T x + b = 0$
- The two classes are decided by whether $w^T x + b > 0$ or < 0
- Can we get something similar in logistic regression?



Functional Form: Two classes

$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

implies

$$P(Y=0|X) = \frac{\exp(w_0 + \sum_{i=1}^n w_i X_i)}{1+\exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

So, logistic regression is a linear classifier after all ☺

Classification Rule: Assign the label Y=0 if

$$1 < \frac{P(Y=0|X)}{P(Y=1|X)}$$

Take logs and simplify:

$$0 < w_0 + \sum_{i=1}^{n} w_i X_i$$

linear classification rule!

Y=0 if the RHS>0

Another way to express it

Logistic Regression can also be expressed as:

$$h_{\theta(x)} = P(Y = 1|X) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

$$P(Y = 0|X) = 1 - h_{\theta(x)} = 1 - g(\theta^T x) = \frac{e^{-\theta^T x}}{1 + e^{-\theta^T x}}$$

To predict class = 1, $\frac{P(Y=1|X)}{P(Y=0|X)} > 1$ or $e^{\theta^T x} > 1\ or\ \theta^T x > 0$

Logistic Regression

- The ratio of probability of success and failure is called odds ratio in statistics.
- In our case, it would be ratio of classes i.e. P(Y=1 |X) divided by P(Y=0 | X).
- Let's do some probability,

Statistical Insight – Odds Ratio

What is the odds (or odds ratio)?

Suppose you have following data:

What is the probability of choosing a red ball P(F): $\frac{3}{12}$

What is the probability of NOT choosing a red ball P(¬F): $\frac{9}{12}$

We define **odds for (of) red** as: $\frac{\text{Favorable Outcomes}}{\text{Unfavorable Outcomes}} = \frac{\mathrm{P}(F)}{P(\neg F)} = \frac{3}{9}$

Statistical Insight – Odds Ratio

What is the odds (or odds ratio)?

Suppose you have following data:

What is the probability of choosing a red ball : $\frac{3}{12}$

What is the probability of NOT choosing a red ball : $\frac{9}{12}$

We define **odds against red** as: $\frac{\text{Unfavorable Outcomes}}{\text{Favorable Outcomes}} = \frac{\mathrm{P}(\neg F)}{P(F)} = \frac{9}{3}$

Statistical Insight – Odds Ratio

What are the odds of winning a game of roulette?

Choices are numbers:

$$1 - 36 + 0 + 00 = \text{Total } 38$$

Only 1 of them wins:

Odds of winning: $\frac{1}{37}$

What if I play on 1st /2nd /3rd 12:

Odds of winning: $\frac{12}{26}$

Odd / Even Numbers: Odds of winning: $\frac{18}{20}$

Statistical Insight – Expected Value

What is the probability of winning a game of roulette?

Choices are numbers:

$$1 – 35 + 0 + 00 = \text{Total } 38$$

Only 1 of them wins:

Probability of winning: $\frac{1}{38}$ Probability of losing: $\frac{37}{38}$

If I bet $1 and my number wins, I get $35, else I lose $1.

What's expected value of win/loss?

$E(W) = 35 * \frac{1}{38} - 1 * \frac{37}{38} = -0.0526$ or 5.26% House Edge

Back to Machine Learning

- Like in any other model, we need to find the parameters – weights in this case.
- How do we do that? What techniques do we know for parameter estimation?
  - Gradient Descent of error
  - Maximum Likelihood
  - MAP (naïve Bayes)

Learning the weights

- How do we learn the weights?
- Maximum likelihood to the rescue.
- Remember, we want to maximize the parameters given the training data.
  Example: Suppose you observe following observations:
  { Y = 1|**X1**, Y = 0|**X2**, Y = 0|**X3**, Y = 1|**X4**}
  We first evaluate how likely is this data in terms of parameters (ϴ)
  Likelihood (L) = P(Y=1|**X1, ϴ**) * P(Y=0|**X2, ϴ**) * P(Y=0|**X3, ϴ**) * P(Y=1|**X4, ϴ**)

  Log Likelihood (LL) = log[P(Y=1|**X1, ϴ**)] + log[P(Y=0|**X2, ϴ**)] + log[P(Y=0|**X3, ϴ**)] + P[(Y=1|**X4, ϴ**)]

  => Differentiate LL w.r.t. ϴ, set to 0 and solve for ϴ

We know how to get this value in Logistic Regression.

Learning the weights

- The value $P(Y^l|X^l, W)$ is known as conditional likelihood for a training example l.
- The optimization problem can be expressed as:

$$W \leftarrow \arg\max_{W} \prod_{l} P(Y^l|X^l, W)$$

W = <w0, w1, …., wn>
vector of parameters to be estimated
Y^l = observed value of Y in the l^{th} example
X^l = observed value of X in the l^{th} example

Learning the weights

- Remember the old trick:
  take the log of the likelihood function
- Want to maximize: $W \leftarrow \arg\max_W \sum_l \ln P(Y^l | X^l, W)$
- This is called the conditional data log likelihood, called $l(W)$
- Remember Y^l can take only 2 values -> 0 and 1

$$l(W) = \sum_l Y^l \ln P(Y^l = 1 | X^l, W) + (1 - Y^l) \ln P(Y^l = 0 | X^l, W)$$

Not convinced?
Set Y^l = 0 and check and then
Set Y^l = 1 and check.

Learning the weights

- To proceed, we use the following functional forms:

$$P(Y=0|X) = \frac{1}{1+\exp(w_0+\sum_{i=1}^{n} w_i X_i)}$$

I know we flipped from earlier definition ☺
But this is done for mathematical convenience

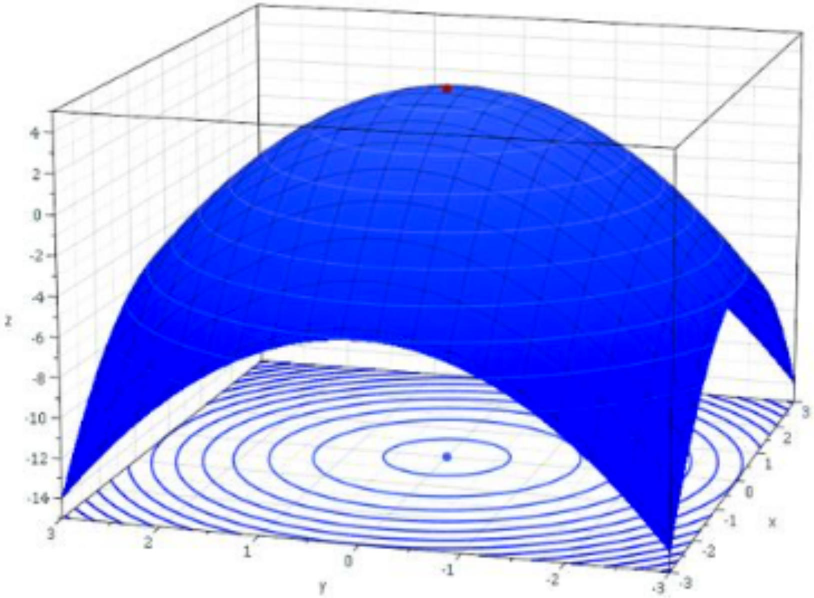
$$P(Y=1|X) = \frac{\exp(w_0+\sum_{i=1}^{n} w_i X_i)}{1+\exp(w_0+\sum_{i=1}^{n} w_i X_i)}$$

- Using in equation for $l(W)$:

$$\begin{aligned} l(W) &= \sum_l Y^l \ln P(Y^l=1|X^l,W) + (1-Y^l)\ln P(Y^l=0|X^l,W) \\ &= \sum_l Y^l \ln \frac{P(Y^l=1|X^l,W)}{P(Y^l=0|X^l,W)} + \ln P(Y^l=0|X^l,W) \\ &= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1+\exp(w_0+\sum_i^n w_i X_i^l)) \end{aligned}$$

Learning the weights

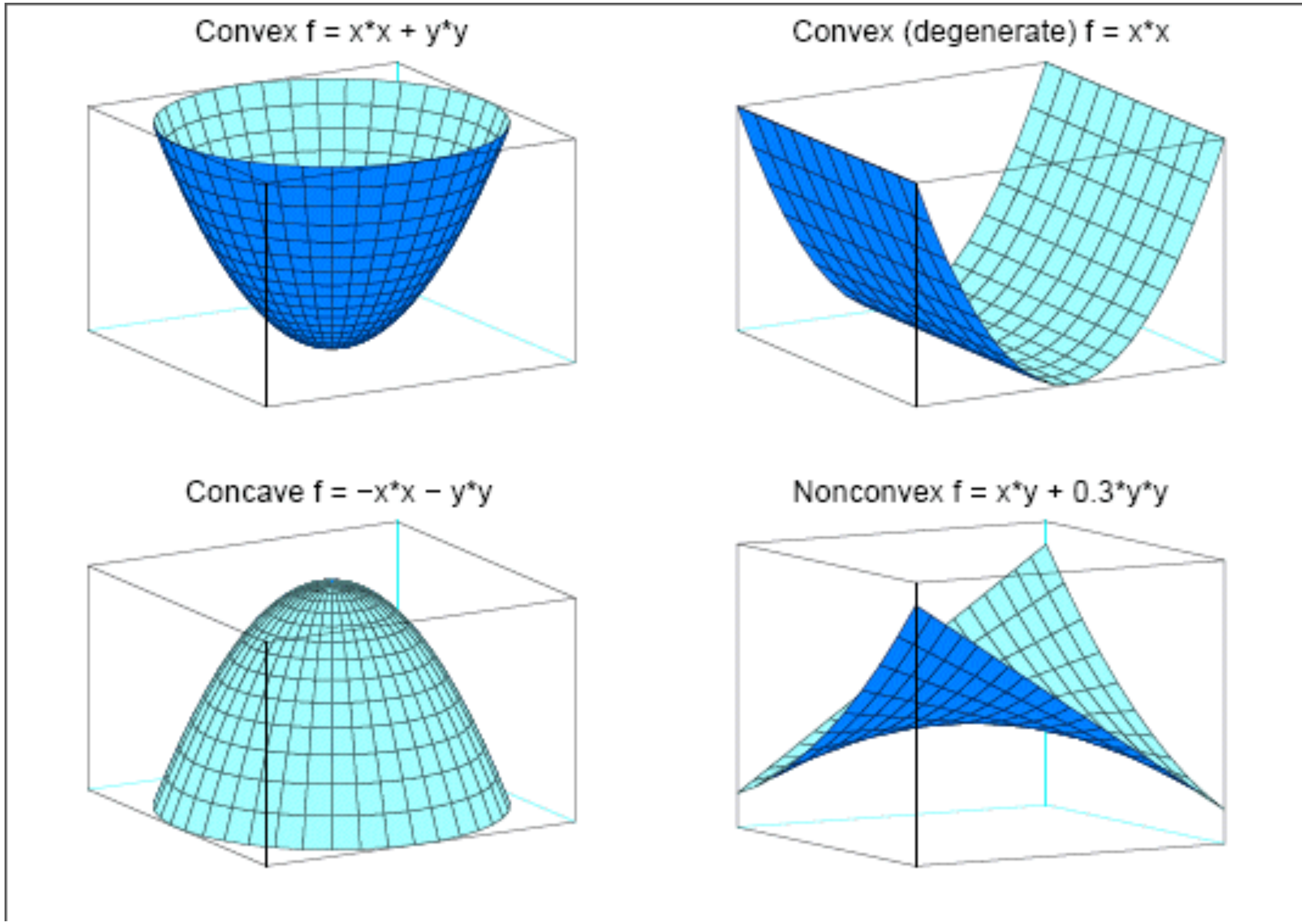
- There is no closed form solution for above equation.
- So how do we proceed? Gradient Ascent
- Why ascent? Because we want maximum value.
- When we wanted to minimize a function e.g. error, we used gradient descent



Convex and Concave functions

- For error, we want a convex function.
- For maxima evaluation, concave function.
- What type of function is the Entropy function
  $E = \sum - p * \log_2(p)$ [assume range of p to be between 0 and 1]?

  Hint: Use R code:
  curve(-x*log2(x)-(1-x)*log2(1-x) ,0, 1)



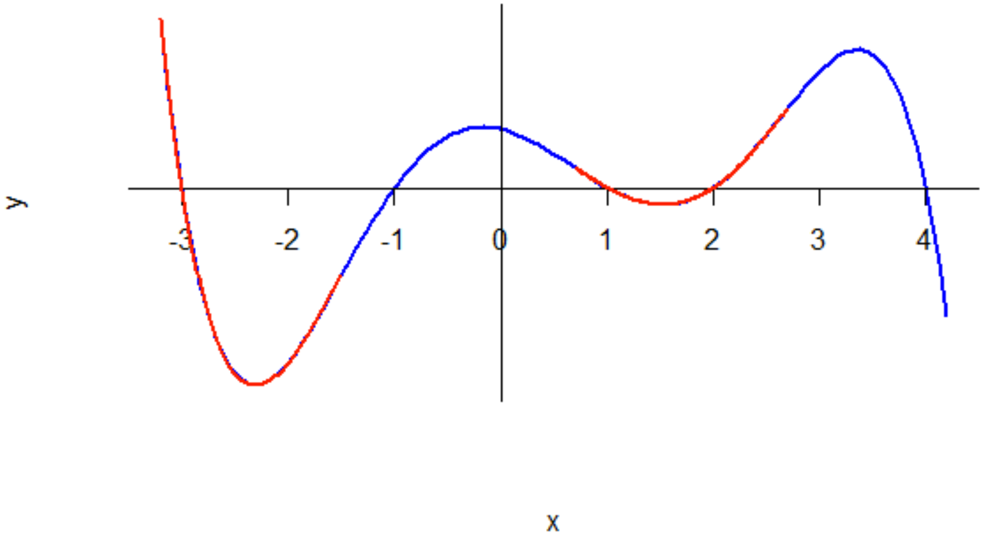
Learning the weights

- We need the slope of the curve: $$\frac{\partial l(W)}{\partial w_i} = \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))$$

where $\hat{P}(Y^l | X^l, W)$ is the LR prediction using current set of weights W

What's the update rule for gradient ascent?

$$w^{new} = w^{old} + \eta \frac{\partial l}{\partial w}$$



Note the + sign. We are climbing up the hill.

Learning the weights

- This leads to the gradient ascent weight update rule:

$$\boxed{w_i \leftarrow w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))}$$

- This is what most software packages use as a starting point for Logistic Regression.

Learning the weights - Regularization

- Large weights are a sign of overfitting. Let's penalize them:

$$W \leftarrow \arg\max_W \sum_l \ln P(Y^l | X^l, W) - \frac{\lambda}{2} ||W||^2$$

  This is called regularization.

- If we repeat the steps using this new objective function, we get the following update rule:

$$w_i \leftarrow w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W)) - \eta \lambda w_i$$

  where η a small constant that determines step size

Self Study Material:

- The textbook shows a nice derivation of Logistic Regression. Go over it and understand the steps.

  https://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf

Summary

- Logistic Regression directly learns the parameters of the model for P(Y|X).
- Naïve Bayes learns the parameters for P(X|Y) and P(Y) and then uses the naïve Bayes equation.
- If we assume that for each value y_k of Y, the distribution of each continuous X_i is Gaussian, then this is called Gaussian Naïve Bayes (GNB).
- The two approaches have been shown as converging towards each other in the textbook and the famous paper:

  Jordan, A. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, *14*, 841.
  Can be downloaded from:
  http://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf

R Resources for Logistic Regression

- Good resource with examples
  https://cran.r-project.org/web/packages/HSAUR/vignettes/Ch_logistic_regression_glm.pdf
- Another Tutorial
  https://ww2.coastal.edu/kingw/statistics/R-tutorials/logistic.html
- Tutorial that explains glm, family, and link
  http://data.princeton.edu/R/glms.html