Classification learning II

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square
Classification

• **Data:** \( D = \{d_1, d_2, \ldots, d_n\} \)
  \[ d_i = \langle x_i, y_i \rangle \]
  – \( y_i \) represents a discrete class value

• **Goal:** learn \( f : X \rightarrow Y \)

• **Binary classification**
  – A special case when \( Y \in \{0,1\} \)

• **First step:**
  – we need to devise a model of the function \( f \)
Discriminant functions

• A common way to represent a classifier is by using
  – Discriminant functions
• Works for both the binary and multi-way classification
• Idea:
  – For every class $i = 0, 1, \ldots, k$ define a function $g_i(x)$
    mapping $X \rightarrow \mathbb{R}$
  – When the decision on input $x$ should be made choose the class with the highest value of $g_i(x)$

$$y^* = \arg \max_i g_i(x)$$
Discriminant functions

- Discriminant functions depend on parameters
  \[ g_i(x) \sim g_i(x, w) \]

- Logistic regression model (for 2 classes)
  \[
  g_1(x, w) = g(w^T x) \quad g_0(x, w) = 1 - g(w^T x)
  \]
  \[
  g(z) = 1/(1 + e^{-z}) \quad \text{- is a logistic function}
  \]
  \[
  g_1(x, w) \sim p(y = 1 \mid x, w) \quad \text{- Models directly class posterior}
  \]

- Generative probabilistic model (QDA)
  \[
  g_1(x, \Theta) = p(y = 1 \mid x, \Theta) = \frac{p(x \mid \mu_1, \Sigma_1) p(y = 1)}{p(x \mid \mu_0, \Sigma_0) p(y = 0) + p(x \mid \mu_1, \Sigma_1) p(y = 1)}
  \]
  \[
  g_0(x, \Theta) = p(y = 0 \mid x, \Theta) = 1 - g_1(x, \Theta)
  \]
Discriminative vs generative models

• What is the difference in between discriminative and generative models and learning?

• **Discriminative**: learns directly the discriminant functions and their parameters

• Define the loss function using \( g_1(x, w) \) and \( y \)

\[
\mathbf{w}^* = \arg \max_{\mathbf{w}} \text{Loss}(g_1(x, \mathbf{w}), y)
\]

• **Generative probabilistic**: learns the joint probability distribution \( p(x, y) \) and its parameters, discriminant functions are derived from the joint.

E.g: ML estimate of the parameters of the joint model

\[
\Theta^* = \arg \max_{\Theta} p(D \mid \Theta) = \arg \max_{\Theta} \prod_{i=1}^{n} p(x_i, y_i \mid \Theta)
\]
Logistic regression model

- **Discriminant functions:**
  \[ g_1(x) = g(w^T x) \quad g_0(x) = 1 - g(w^T x) \]

- Values of discriminant functions vary in [0,1]
  - **Probabilistic interpretation**
    \[ g_1(x, w) = p(y = 1 \mid w, x) = g(w^T x) \]
Binary classification example
Logistic regression model. Decision boundary

- **LR defines a linear decision boundary**

**Example:** 2 classes (blue and red points)
Generative approach to classification

Idea:

1. Represent and learn the distribution \( p(x, y) \)
2. Use it to define probabilistic discriminant functions

E.g. \( g_o(x) = p(y = 0 \mid x) \quad g_1(x) = p(y = 1 \mid x) \)

Typical model \( p(x, y) = p(x \mid y) p(y) \)

- \( p(x \mid y) = \text{Class-conditional distributions (densities)} \)
  - binary classification: two class-conditional distributions
    \( p(x \mid y = 0) \quad p(x \mid y = 1) \)
- \( p(y) = \text{Priors on classes} - \) probability of class \( y \)
  - binary classification: Bernoulli distribution
    \[ p(y = 0) + p(y = 1) = 1 \]
Quadratic discriminant analysis (QDA)

Model:

- **Class-conditional distributions**
  - multivariate normal distributions
    \[
    \begin{align*}
    x & \sim N(\mu_0, \Sigma_0) \quad \text{for} \quad y = 0 \\
    x & \sim N(\mu_1, \Sigma_1) \quad \text{for} \quad y = 1
    \end{align*}
    \]
    Multivariate normal \( x \sim N(\mu, \Sigma) \)
    
    \[
    p(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu)\right]
    \]

- **Priors on classes (class 0,1)** \( y \sim Bernoulli \)
  - Bernoulli distribution
    \[
    p(y, \theta) = \theta^y (1 - \theta)^{1-y} \quad y \in \{0,1\}
    \]
Learning of parameters of the QDA model

Density estimation in statistics

• We see examples – we do not know the parameters of Gaussians (class-conditional densities)

\[ p(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp \left[ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right] \]

• ML estimate of parameters of a multivariate normal \( N(\mu, \Sigma) \) for a set of \( n \) examples of \( x \)

Optimize log-likelihood: \( l(D, \mu, \Sigma) = \log \prod_{i=1}^{n} p(x_i \mid \mu, \Sigma) \)

\[ \hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i \]
\[ \hat{\Sigma} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu})(x_i - \hat{\mu})^T \]

• How about class priors?
QDA
2 Gaussian class-conditional densities
QDA: Making class decision

Basically we need to design discriminant functions

Two possible choices:

- **Likelihood of data** – choose the class (Gaussian) that explains the input data ($x$) better (likelihood of the data)

\[
\frac{p(x | \mu_1, \Sigma_1)}{g_1(x)} > \frac{p(x | \mu_0, \Sigma_0)}{g_0(x)} \quad \text{then} \quad y = 1
\]

\[
\text{else} \quad y = 0
\]

- **Posterior of a class** – choose the class with better posterior probability

\[
p(y = 1 | x) > p(y = 0 | x) \quad \text{then} \quad y = 1
\]

\[
\text{else} \quad y = 0
\]

\[
p(y = 1 | x) = \frac{p(x | \mu_1, \Sigma_1) p(y = 1)}{p(x | \mu_0, \Sigma_0) p(y = 0) + p(x | \mu_1, \Sigma_1) p(y = 1)}
\]
QDA: Quadratic decision boundary

Contours of class-conditional densities
QDA: Quadratic decision boundary
Linear discriminant analysis (LDA)

- When covariances are the same
  \[ x \sim N(\mu_0, \Sigma), \ y = 0 \]
  \[ x \sim N(\mu_1, \Sigma), \ y = 1 \]
LDA: Linear decision boundary
LDA: linear decision boundary
Naïve Bayes classifier

• A generative classifier model with an additional simplifying assumption:
  – All input attributes are conditionally independent of each other given the class.

So we have:

\[ p(x, y) = p(x \mid y) p(y) \]

\[ p(x \mid y) = \prod_{i=1}^{d} p(x_i \mid y) \]
Learning parameters of the model

Much simpler density estimation problems

• We need to learn:
  \[ p(x \mid y = 0) \text{ and } p(x \mid y = 1) \text{ and } p(y) \]

• Because of the assumption of the conditional independence we need to learn:
  
  for every variable \( i \): \( p(x_i \mid y = 0) \text{ and } p(x_i \mid y = 1) \)

• Much easier if the number of input attributes is large

• Also, the model gives us a flexibility to represent input attributes different of different forms !!!

• E.g. one attribute can be modeled using the Bernoulli, the other as Gaussian density, or as a Poisson distribution
Making a class decision for the Naïve Bayes

**Discriminant functions**

- **Likelihood of data** – choose the class that explains the input data \((x)\) better (likelihood of the data)

\[
\prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) > \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \quad \text{then } y=1
\]

\[
g_1(x) > g_0(x)
\]

- **Posterior of a class** – choose the class with better posterior probability

\[
p(y = 1 \mid x) > p(y = 0 \mid x) \quad \text{then } y=1
\]

\[
p(y = 1 \mid x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 1)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \right) p(y = 1)}
\]

\[
p(y = 0 \mid x) = 1 - p(y = 1 \mid x)
\]

\[
g_0(x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 1)}
\]

\[
g_1(x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 1)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \right) p(y = 1)}
\]

\[
g_0(x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \right) p(y = 1)}
\]

\[
g_1(x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 1)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \right) p(y = 1)}
\]

\[
g_0(x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \right) p(y = 1)}
\]

\[
g_1(x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 1)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{1,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \right) p(y = 1)}
\]

\[
g_0(x) = \frac{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0)}{\left( \prod_{i=1}^{d} p(x_i \mid \Theta_{0,i}) \right) p(y = 0) + \left( \prod_{i=1}^{d} p(x_i \mid \Theta_{2,i}) \right) p(y = 1)}
\]
Back to logistic regression

- **Two models with linear decision boundaries:**
  - Logistic regression
  - Generative model with 2 Gaussians with the same covariance matrices

\[
x \sim N(\mu_0, \Sigma) \quad \text{for} \quad y = 0
\]

\[
x \sim N(\mu_1, \Sigma) \quad \text{for} \quad y = 1
\]

- **Two models are related !!!**
  - When we have 2 Gaussians with the same covariance matrix the probability of \( y \) given \( x \) has the form of a logistic regression model !!!

\[
p(y = 1|x, \mu_0, \mu_1, \Sigma) = g(w^T x)
\]
When is the logistic regression model correct?

• **Members of the exponential family can be often more naturally described as**

\[ f(x | \theta, \varphi) = h(x, \varphi) \exp \left\{ \frac{\theta^T x - A(\theta)}{a(\varphi)} \right\} \]

- **\( \theta \)** - A location parameter
- **\( \varphi \)** - A scale parameter

• **Claim:** A logistic regression is a correct model when class conditional densities are from the same distribution in the exponential family and have **the same scale factor** \( \varphi \)

• **Very powerful result !!!!**
  - We can represent posteriors of many distributions with the same small network
Linear units

**Linear regression**

\[ f(x) = w^T x \]

Gradient update:

\[ w \leftarrow w + \alpha \sum_{i=1}^{n} (y_i - f(x_i))x_i \]

Online: \[ w \leftarrow w + \alpha (y - f(x))x \]

**Logistic regression**

\[ f(x) = p(y = 1 | x, w) = g(w^T x) \]

Gradient update:

\[ w \leftarrow w + \alpha \sum_{i=1}^{n} (y_i - f(x_i))x_i \]

Online: \[ w \leftarrow w + \alpha (y - f(x))x \]
Gradient-based learning

- The **same simple gradient update rule** derived for both the linear and logistic regression models
- Where the magic comes from?
- Under the **log-likelihood** measure the function models and the models for the output selection fit together:
  - **Linear model + Gaussian noise**
    \[ y = w^T x + \varepsilon \quad \varepsilon \sim N(0, \sigma^2) \]
  - **Logistic + Bernoulli**
    \[ y = \text{Bernoulli}(\theta) \]
    \[ \theta = p(y = 1 \mid x) = g(w^T x) \]
Generalized linear models (GLIM)

Assumptions:
• The conditional mean (expectation) is:
  \[ \mu = f(w^T x) \]
  – Where \( f(.) \) is a **response function**
• Output \( y \) is characterized by an exponential family distribution with a conditional mean \( \mu \)

Examples:
  – Linear model + Gaussian noise
    \[ y = w^T x + \varepsilon \quad \varepsilon \sim N(0, \sigma^2) \]
  – Logistic + Bernoulli
    \[ y \approx \text{Bernoulli}(\theta) \]
    \[ \theta = g(w^T x) = \frac{1}{1 + e^{-w^T x}} \]
Generalized linear models

- A canonical response functions $f(\cdot)$:
  - encoded in the distribution

$$p(x \mid \theta, \phi) = h(x, \phi) \exp \left\{ \frac{\theta^T x - A(\theta)}{a(\phi)} \right\}$$

- Leads to a simple gradient form
- Example: Bernoulli distribution

$$p(x \mid \mu) = \mu^x (1 - \mu)^{1-x} = \exp \left\{ \log \left( \frac{\mu}{1 - \mu} \right)x + \log(1 - \mu) \right\}$$

$$\theta = \log \left( \frac{\mu}{1 - \mu} \right) \quad \mu = \frac{1}{1 + e^{-\theta}}$$

- Logistic function matches the Bernoulli
When does the logistic regression fail?

- Quadratic decision boundary is needed
When does the logistic regression fail?

- Another example of a non-linear decision boundary
Non-linear extension of logistic regression

- use feature (basis) functions to model nonlinearities
- the same trick as used for the linear regression

**Linear regression**

\[ f(x) = w_0 + \sum_{j=1}^{m} w_j \phi_j(x) \]

**Logistic regression**

\[ f(x) = g(w_0 + \sum_{j=1}^{m} w_j \phi_j(x)) \]

\( \phi_j(x) \) - an arbitrary function of \( x \)
Evaluation of classifiers
Evaluation

For any data set we use to test the classification model on we can build a **confusion matrix:**

- Counts of examples with:
  - class label $\omega_j$ that are classified with a label $\alpha_i$

<table>
<thead>
<tr>
<th>predict</th>
<th>target</th>
<th>$\omega = 1$</th>
<th>$\omega = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha = 1$</td>
<td>140</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>$\alpha = 0$</td>
<td>20</td>
<td>54</td>
<td></td>
</tr>
</tbody>
</table>
Evaluation

For any data set we use to test the model we can build a **confusion matrix**:

<table>
<thead>
<tr>
<th>predict</th>
<th>target</th>
<th>$\omega = 1$</th>
<th>$\omega = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha = 1$</td>
<td>140</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>$\alpha = 0$</td>
<td>20</td>
<td>54</td>
<td></td>
</tr>
</tbody>
</table>

Error: ?
# Evaluation

For any data set we use to test the model we can build a confusion matrix:

<table>
<thead>
<tr>
<th>predict</th>
<th>$\omega = 1$</th>
<th>$\omega = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha = 1$</td>
<td>140</td>
<td>17</td>
</tr>
<tr>
<td>$\alpha = 0$</td>
<td>20</td>
<td>54</td>
</tr>
</tbody>
</table>

Error: $= \frac{37}{231}$

Accuracy $= 1 - \text{Error} = \frac{194}{231}$
**Evaluation for binary classification**

Entries in the confusion matrix for binary classification have names:

<table>
<thead>
<tr>
<th>predict</th>
<th>target $\omega = 1$</th>
<th>target $\omega = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha = 1$</td>
<td>$TP$</td>
<td>$FP$</td>
</tr>
<tr>
<td>$\alpha = 0$</td>
<td>$FN$</td>
<td>$TN$</td>
</tr>
</tbody>
</table>

*TP*: True positive (hit)

*FP*: False positive (false alarm)

*TN*: True negative (correct rejection)

*FN*: False negative (a miss)
Additional statistics

• Sensitivity (recall)

\[ SENS = \frac{TP}{TP + FN} \]

• Specificity

\[ SPEC = \frac{TN}{TN + FP} \]

• Positive predictive value (precision)

\[ PPT = \frac{TP}{TP + FP} \]

• Negative predictive value

\[ NPV = \frac{TN}{TN + FN} \]
### Binary classification: additional statistics

- **Confusion matrix**

<table>
<thead>
<tr>
<th>predict</th>
<th>target</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>140</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>20</td>
<td>180</td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{align*}
PPV &= \frac{140}{150} \\
NPV &= \frac{180}{200} \\
SENS &= \frac{140}{160} \\
SPEC &= \frac{180}{190}
\end{align*}
\]

**Row and column quantities:**
- Sensitivity (SENS)
- Specificity (SPEC)
- Positive predictive value (PPV)
- Negative predictive value (NPV)
Binary decisions: Receiver Operating Curves

- **Probabilities:**
  - **SENS**
  - **SPEC**

\[
p(x > x^* \mid \mathbf{x} \in \omega_2)\]

\[
p(x < x^* \mid \mathbf{x} \in \omega_1)\]
Receiver Operating Characteristic (ROC)

- ROC curve plots:
  \[ SN = p(x > x^* \mid x \in \omega_2) \]
  \[ 1-SP = p(x > x^* \mid x \in \omega_1) \]
  for different \( x^* \)

\[ SENS = p(x > x^* \mid x \in \omega_2) \]
ROC curve

Case 1

Case 2

Case 3

\[ p(x > x^* \mid x \in \omega_2) \]

\[ p(x > x^* \mid x \in \omega_1) \]
Receiver operating characteristic

- **ROC**
  - shows the discriminability between the two classes under different decision biases

- **Decision bias**
  - can be changed using different loss function
Back to classification models
Discriminant functions
Discriminant functions

\[ g_1(x) \leq g_0(x) \]
Discriminant functions

\[ g_1(x) \geq g_0(x) \]

\[ g_1(x) \leq g_0(x) \]
Discriminant functions

- Define decision boundary

\[ g_1(x) \geq g_0(x) \]

\[ g_1(x) = g_0(x) \]

\[ g_1(x) \leq g_0(x) \]
Quadratic decision boundary

\[ g_1(x) \geq g_0(x) \]

\[ g_1(x) \leq g_0(x) \]

\[ g_1(x) = g_0(x) \]
Logistic regression model

- Defines a linear decision boundary
- Discriminant functions:
  \[ g_1(x) = g(w^T x) \quad \quad g_0(x) = 1 - g(w^T x) \]
- where \( g(z) = 1/(1 + e^{-z}) \) - is a logistic function

\[ f(x, w) = g_1(w^T x) = g(w^T x) \]
Logistic function

\[ g(z) = \frac{1}{1 + e^{-z}} \]

- Is also referred to as a **sigmoid function**
- Replaces the threshold function with smooth switching
- Takes a real number and outputs the number in the interval \([0,1]\)
Generative approach to classification

Idea:

1. Represent and learn the distribution \( p(x, y) \)
2. Use it to define probabilistic discriminant functions

E.g. \( g_o(x) = p(y = 0 \mid x) \quad g_1(x) = p(y = 1 \mid x) \)

Typical model \( p(x, y) = p(x \mid y) p(y) \)

- \( p(x \mid y) = \text{Class-conditional distributions (densities)} \)
  - binary classification: two class-conditional distributions
    \( p(x \mid y = 0) \quad p(x \mid y = 1) \)
- \( p(y) = \text{Priors on classes} - \) probability of class \( y \)
  - binary classification: Bernoulli distribution
    \( p(y = 0) + p(y = 1) = 1 \)
Quadratic discriminant analysis (QDA)

Model:

- **Class-conditional distributions**
  - **multivariate normal distributions**
    \[
    \mathbf{x} \sim N(\mu_0, \Sigma_0) \quad \text{for} \quad y = 0
    \]
    \[
    \mathbf{x} \sim N(\mu_1, \Sigma_1) \quad \text{for} \quad y = 1
    \]

  - Multivariate normal \( \mathbf{x} \sim N(\mu, \Sigma) \)
    \[
p(x | \mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp \left[-\frac{1}{2} (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)\right]
    \]

- **Priors on classes (class 0,1)** \( y \sim Bernoulli \)
  - **Bernoulli distribution**
    \[
p(y, \theta) = \theta^y (1-\theta)^{1-y} \quad y \in \{0,1\}
    \]
QDA
2 Gaussian class-conditional densities
QDA: Quadratic decision boundary
Linear discriminant analysis (LDA)

- When covariances are the same:
  \[ x \sim N(\mu_0, \Sigma), \ y = 0 \]
  \[ x \sim N(\mu_1, \Sigma), \ y = 1 \]
LDA: Linear decision boundary
LDA: linear decision boundary
Logistic regression vs LDA

• Two models with linear decision boundaries:
  – Logistic regression
  – Generative model with 2 Gaussians with the same covariance matrices

\[ x \sim N(\mu_0, \Sigma) \quad \text{for} \quad y = 0 \]
\[ x \sim N(\mu_1, \Sigma) \quad \text{for} \quad y = 1 \]

• Two models are related !!!
  – When we have 2 Gaussians with the same covariance matrix the probability of \( y \) given \( x \) has the form of a logistic regression model !!!

\[ p(y = 1 \mid x, \mu_0, \mu_1, \Sigma) = g(w^T x) \]
When is the logistic regression model correct?

- **Members of the exponential family can be often more naturally described as**

\[
 f(x | \theta, \phi) = h(x, \phi) \exp \left\{ \frac{\theta^T x - A(\theta)}{a(\phi)} \right\}
\]

\[\theta\] - A location parameter \[\phi\] - A scale parameter

- **Claim**: A logistic regression is a correct model when class conditional densities are from the same distribution in the exponential family and have the same scale factor \[\phi\]

- **Very powerful result !!!!**
  - We can represent posteriors of many distributions with the same small network
Linear units

Linear regression

\[ f(x) = w^T x \]

Logistic regression

\[ f(x) = p(y = 1 | x, w) = g(w^T x) \]

\[ \text{Gradient update:} \]

\[ w \leftarrow w + \alpha \sum_{i=1}^{n} (y_i - f(x_i)) x_i \]

Online: \[ w \leftarrow w + \alpha (y - f(x)) x \]

The same

\[ w \leftarrow w + \alpha \sum_{i=1}^{n} (y_i - f(x_i)) x_i \]

Online: \[ w \leftarrow w + \alpha (y - f(x)) x \]
Gradient-based learning

- The **same simple gradient update rule** derived for both the linear and logistic regression models
- Where the magic comes from?
- Under the **log-likelihood** measure the function models and the models for the output selection fit together:
  - **Linear model + Gaussian noise**
    \[
    y = w^T x + \varepsilon \\
    \varepsilon \sim N(0, \sigma^2)
    \]
  - **Logistic + Bernoulli**
    \[
    y \sim \text{Bern}(\theta) \\
    \theta = p(y = 1 \mid x) = g(w^T x)
    \]
Generalized linear models (GLIM)

Assumptions:
- The conditional mean (expectation) is:  \( \mu = f(w^T x) \)
  - \( f(.) \) is a response (or a link) function
- Output \( y \) is characterized by an exponential family distribution with mean  \( \mu = f(w^T x) \)

Examples:
- Linear model + Gaussian noise
  \( y = w^T x + \varepsilon \quad \varepsilon \sim N(0, \sigma^2) \)
  \( y \sim N(w^T x, \sigma^2) \)
- Logistic + Bernoulli
  \( y \sim \text{Bern}(\theta) \sim \text{Bern}(g(w^T x)) \)
  \( \theta = g(w^T x) = \frac{1}{1 + e^{-w^T x}} \)
Generalized linear models (GLMs)

- A canonical response functions $f(.)$ :
  - encoded in the distribution

$$p(x \mid \theta, \phi) = h(x, \phi) \exp \left\{ \frac{\theta^T x - A(\theta)}{a(\phi)} \right\}$$

- Leads to a simple gradient form
- Example: Bernoulli distribution

$$p(x \mid \mu) = \mu^x (1 - \mu)^{1-x} = \exp \left\{ \log \left( \frac{\mu}{1 - \mu} \right) x + \log(1 - \mu) \right\}$$

$$\theta = \log \left( \frac{\mu}{1 - \mu} \right) \quad \mu = \frac{1}{1 + e^{-\theta}}$$

- Logistic function matches the Bernoulli
Non-linear extension of logistic regression

- use **feature (basis) functions** to model **nonlinearities**
  - the same trick as used for the linear regression

**Linear regression**

\[
f(x) = w_0 + \sum_{j=1}^{m} w_j \phi_j(x)
\]

**Logistic regression**

\[
f(x) = g(w_0 + \sum_{j=1}^{m} w_j \phi_j(x))
\]

\(\phi_j(x)\) - an arbitrary function of \(x\)
Similarly to the linear regression we can penalize the logistic regression or other GLM models for their complexity

- **L1 (lasso) regularization penalty**
- **L2 (ridge) regularization penalty**

- **Typically:** the optimization of weights \( w \) looks as follows

\[
\min_w \quad Loss(D, w) + Q(w)
\]

- **\( Loss(D, w) \) functions:**
  - Mean squared error
  - Negative log-likelihood

- **Regularization penalty** \( Q(w) \): L1, L2 or a combination
When does the logistic regression fail?

- Quadratic decision boundary is needed
When does the logistic regression fail?

- Another example of a non-linear decision boundary