Linear Models for Supervised Learning

Weinan Zhang
Shanghai Jiao Tong University
http://wnzhang.net
Discriminative Model and Generative Model

• **Discriminative model**
  • modeling the dependence of unobserved variables on observed ones
  • also called conditional models.
  • Deterministic: \( y = f_\theta(x) \)
  • Probabilistic: \( p_\theta(y|x) \)

• **Generative model**
  • modeling the joint probabilistic distribution of data
  • given some hidden parameters or variables
    \[ p_\theta(x, y) \]
  • then do the conditional inference
    \[
    p_\theta(y|x) = \frac{p_\theta(x, y)}{p_\theta(x)} = \frac{p_\theta(x, y)}{\sum_{y'} p_\theta(x, y')}
    \]
Discriminative Model and Generative Model

• **Discriminative model**
  - modeling the dependence of unobserved variables on observed ones
  - also called conditional models.
  - Deterministic: \( y = f_\theta(x) \)
  - Probabilistic: \( p_\theta(y|x) \)

• Directly model the dependence for label prediction
• Easy to define dependence specific features and models
• Practically yielding higher prediction performance

• Linear regression, logistic regression, k nearest neighbor, SVMs, (multi-layer) perceptrons, decision trees, random forest etc.
Discriminative Model and Generative Model

• **Generative model**
  • modeling the joint probabilistic distribution of data
  • given some hidden parameters or variables
    \[ p_\theta(x, y) \]
  • then do the conditional inference
    \[ p_\theta(y|x) = \frac{p_\theta(x, y)}{p_\theta(x)} = \frac{p_\theta(x, y)}{\sum_{y'} p_\theta(x, y')} \]

• Recover the data distribution [essence of data science]
• Benefit from hidden variables modeling

• Naive Bayes, Hidden Markov Model, Mixture Gaussian, Markov Random Fields, Latent Dirichlet Allocation etc.
Linear Regression
Linear Discriminative Models

• Discriminative model
  • modeling the dependence of unobserved variables on observed ones
  • also called conditional models.
  • Deterministic: \( y = f_\theta(x) \)
  • Probabilistic: \( p_\theta(y|x) \)

• Focus of this course
  • Linear regression model
  • Linear classification model
Linear Discriminative Models

• Discriminative model
  • modeling the dependence of unobserved variables on observed ones
  • also called conditional models.
  • **Deterministic**: \( y = f_\theta(x) \)
  • Probabilistic: \( p_\theta(y|x) \)

• Linear regression model

\[
y = f_\theta(x) = \theta_0 + \sum_{j=1}^{d} \theta_j x_j = \theta^\top x
\]

\[
x = (1, x_1, x_2, \ldots, x_d)
\]
Linear Regression

- One-dimensional linear & quadratic regression

\[ f(x) = \theta_0 + \theta_1 x \]

Linear Regression

\[ f(x) = \theta_0 + \theta_1 x + \theta_2 x^2 \]

Quadratic Regression

(A kind of generalized linear model)
Linear Regression

- Two-dimensional linear regression

\[ f(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \]
Learning Objective

• Make the prediction close to the corresponding label

\[
\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i, f_\theta(x_i))
\]

• Loss function \( \mathcal{L}(y_i, f_\theta(x_i)) \) measures the error between the label and prediction

• The definition of loss function depends on the data and task

• Most popular loss function: squared loss

\[
\mathcal{L}(y_i, f_\theta(x_i)) = (y_i - f_\theta(x_i))^2
\]
Squared Loss

\[ \mathcal{L}(y_i, f_\theta(x_i)) = \frac{1}{2}(y_i - f_\theta(x_i))^2 \]

- Penalty much more on larger distances
- Accept small distance (error)
  - Observation noise etc.
  - Generalization
Least Square Linear Regression

• Objective function to minimize

\[ J_\theta = \frac{1}{2N} \sum_{i=1}^{N} (y_i - f_\theta(x_i))^2 \]

\[ \min_\theta J_\theta \]
Minimize the Objective Function

• Let \( N=1 \) for a simple case, for \((x,y) = (2,1)\)

\[
J(\theta) = \frac{1}{2}(y - \theta_0 - \theta_1 x)^2 = \frac{1}{2}(1 - \theta_0 - 2\theta_1)^2
\]
Gradient Learning Methods

\[
\theta_{\text{new}} \leftarrow \theta_{\text{old}} - \eta \frac{\partial \mathcal{L}(\theta)}{\partial \theta}
\]
Batch Gradient Descent

\[ J(\theta) = \frac{1}{2N} \sum_{i=1}^{N} (y_i - f_\theta(x_i))^2 \]

\[ \min_{\theta} J(\theta) \]

- Update \( \theta_{\text{new}} \leftarrow \theta_{\text{old}} - \eta \frac{\partial J(\theta)}{\partial \theta} \) for the whole batch

\[
\frac{\partial J(\theta)}{\partial \theta} = -\frac{1}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i)) \frac{\partial f_\theta(x_i)}{\partial \theta}
\]

\[ = -\frac{1}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i))x_i \]

\[ \theta_{\text{new}} = \theta_{\text{old}} + \eta \frac{1}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i))x_i \]
Learning Linear Model - Curve

\[ f(x) = \theta_0 + \theta_1 x \]
Learning Linear Model - Weights
Stochastic Gradient Descent

\[ J^{(i)}(\theta) = \frac{1}{2} (y_i - f_\theta(x_i))^2 \quad \min_\theta \frac{1}{N} \sum_i J^{(i)}(\theta) \]

• Update \( \theta_{\text{new}} = \theta_{\text{old}} - \eta \frac{\partial J^{(i)}(\theta)}{\partial \theta} \) for every single instance

\[
\frac{\partial J^{(i)}(\theta)}{\partial \theta} = -(y_i - f_\theta(x_i)) \frac{\partial f_\theta(x_i)}{\partial \theta} \\
= -(y_i - f_\theta(x_i))x_i \\
\theta_{\text{new}} = \theta_{\text{old}} + \eta (y_i - f_\theta(x_i))x_i
\]

• Compare with BGD
  • Faster learning
  • Uncertainty or fluctuation in learning
Linear Classification Model
Mini-Batch Gradient Descent

• A combination of batch GD and stochastic GD

• Split the whole dataset into $K$ mini-batches

  \{1, 2, 3, \ldots, K\}

• For each mini-batch $k$, perform one-step BGD toward minimizing

  $$J^{(k)}(\theta) = \frac{1}{2N_k} \sum_{i=1}^{N_k} (y_i - f_\theta(x_i))^2$$

• Update $\theta_{\text{new}} = \theta_{\text{old}} - \eta \frac{\partial J^{(k)}(\theta)}{\partial \theta}$ for each mini-batch
Mini-Batch Gradient Descent

• Good learning stability (BGD)
• Good convergence rate (SGD)

• Easy to be parallelized
  • Parallelization within a mini-batch
Basic Search Procedure

• Choose an initial value for $\theta$
• Update $\theta$ iteratively with the data
• Until we research a minimum
Basic Search Procedure

• Choose a new initial value for $\theta$
• Update $\theta$ iteratively with the data
• Until we research a minimum

$J(\theta_0, \theta_1)$
Unique Minimum for Convex Objective

- Different initial parameters and different learning algorithm lead to the same optimum
Convex Set

• A convex set $S$ is a set of points such that, given any two points $A, B$ in that set, the line $AB$ joining them lies entirely within $S$.

\[ tx_1 + (1 - t)x_2 \in S \]
for all $x_1, x_2 \in S, 0 \leq t \leq 1$

Convex Function

A function \( f : \mathbb{R}^n \to \mathbb{R} \) is convex if \( \text{dom } f \) is a convex set and

\[
\forall x_1, x_2 \in \text{dom } f, 0 \leq t \leq 1, \quad f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)
\]
Choosing Learning Rate

\[ \theta_{\text{new}} = \theta_{\text{old}} - \eta \frac{\partial J(\theta)}{\partial \theta} \]

- \( \eta \) too small
  - slow convergence

- \( \eta \) too large
  - May overshoot the minimum
  - May fail to converge
  - May even diverge
  - Increasing value of \( J(\theta) \)

- The initial point may be too far away from the optimal solution, which takes much time to converge

- To see if gradient descent is working, print out \( J(\theta) \) for each or every several iterations. If \( J(\theta) \) does not drop properly, adjust \( \eta \)
Algebra Perspective

\[ X = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(n)} \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \cdots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \cdots & x_d^{(n)} \end{bmatrix} \]

\[ \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_d \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \]

- Prediction \( \hat{y} = X\theta = \begin{bmatrix} x^{(1)} \theta \\ x^{(2)} \theta \\ \vdots \\ x^{(n)} \theta \end{bmatrix} \)

- Objective \( J(\theta) = \frac{1}{2} (y - \hat{y})^\top (y - \hat{y}) = \frac{1}{2} (y - X\theta)^\top (y - X\theta) \)
Matrix Form

• Objective

\[ J(\theta) = \frac{1}{2}(y - X\theta)^{\top}(y - X\theta) \min_{\theta} J(\theta) \]

• Gradient

\[ \frac{\partial J(\theta)}{\partial \theta} = -X^{\top}(y - X\theta) \]

• Solution

\[ \frac{\partial J(\theta)}{\partial \theta} = 0 \Rightarrow X^{\top}(y - X\theta) = 0 \]

\[ \Rightarrow X^{\top}y = X^{\top}X\theta \]

\[ \Rightarrow \hat{\theta} = (X^{\top}X)^{-1}X^{\top}y \]
Matrix Form

• Then the predicted values are

\[ \hat{y} = X(X^\top X)^{-1}X^\top y \]

\[ = Hy \]

\( H: \) hat matrix

• Geometrical Explanation

• The column vectors \([x_1, x_2, \ldots, x_d]\) form a subspace of \(\mathbb{R}^n\)

• \(H\) is a least square projection

\[
X = \begin{bmatrix}
  x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \ldots & x_d^{(1)} \\
  x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \ldots & x_d^{(2)} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \ldots & x_d^{(n)} \\
\end{bmatrix}
\]

\[
y = \begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_n \\
\end{bmatrix}
\]

More details refer to Sec 3.2. Hastie et al. The elements of statistical learning.
$X^\top X$ Might be Singular

- When some column vectors are not independent
  - For example, $x_2 = 3x_1$
  
  then $X^\top X$ is singular, thus $\hat{\theta} = (X^\top X)^{-1} X^\top y$
  cannot be directly calculated.

- Solution: regularization

$$J(\theta) = \frac{1}{2} (y - X\theta)^\top (y - X\theta) + \frac{\lambda}{2} \|\theta\|^2_2$$
Matrix Form with Regularization

• Objective

\[ J(\theta) = \frac{1}{2}(y - X\theta)^\top(y - X\theta) + \frac{\lambda}{2}\|\theta\|_2^2 \quad \text{min}_{\theta} J(\theta) \]

• Gradient

\[ \frac{\partial J(\theta)}{\partial \theta} = -X^\top(y - X\theta) + \lambda\theta \]

• Solution

\[ \frac{\partial J(\theta)}{\partial \theta} = 0 \quad \rightarrow \quad -X^\top(y - X\theta) + \lambda\theta = 0 \]
\[ \rightarrow \quad X^\top y = (X^\top X + \lambda I)\theta \]
\[ \rightarrow \quad \hat{\theta} = (X^\top X + \lambda I)^{-1}X^\top y \]
Linear Discriminative Models

- Discriminative model
  - modeling the dependence of unobserved variables on observed ones
  - also called conditional models.
- Deterministic: \( y = f_\theta(x) \)
- Probabilistic: \( p_\theta(y|x) \)

- Linear regression with Gaussian noise model

\[
y = f_\theta(x) + \epsilon = \theta_0 + \sum_{j=1}^{d} \theta_j x_j + \epsilon = \theta^\top x + \epsilon
\]

\( \epsilon \sim \mathcal{N}(0, \sigma^2) \)

\( x = (1, x_1, x_2, \ldots, x_d) \)
Objective: Likelihood

\[ \epsilon \sim \mathcal{N}(0, \sigma^2) \]

\[ p(\epsilon) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\epsilon^2}{2\sigma}} \]

\[ y = \theta^\top x + \epsilon \]

• Data likelihood

\[ p(y|x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-\theta^\top x)^2}{2\sigma}} \]
Learning

• Maximize the data likelihood

$$\max_{\theta} \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \theta^\top x_i)^2}{2\sigma}}$$

• Maximize the data log-likelihood

$$\log \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \theta^\top x_i)^2}{2\sigma}} = \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \theta^\top x_i)^2}{2\sigma}}$$

$$= - \sum_{i=1}^{N} \frac{(y_i - \theta^\top x_i)^2}{2\sigma} + \text{const}$$

$$\min_{\theta} \sum_{i=1}^{N} (y_i - \theta^\top x_i)^2 \quad \text{Equivalent to least square error learning}$$
Linear Classification
Classification Problem

• Given:
  • A description of an instance, $x \in \mathbb{X}$, where $\mathbb{X}$ is the instance space.
  • A fixed set of categories: $C = \{c_1, c_2, \ldots, c_m\}$

• Determine:
  • The category of $x: f(x) \in C$, where $f(x)$ is a categorization function whose domain is $\mathbb{X}$ and whose range is $C$
  • If the category set binary, i.e. $C = \{0, 1\}$ (\{false, true\}, \{negative, positive\}) then it is called binary classification.
Binary Classification

Linearly inseparable

Non-linearly inseparable
Linear Discriminative Models

• Discriminative model
  • modeling the dependence of unobserved variables on observed ones
  • also called conditional models.
• Deterministic: \( y = f_\theta(x) \)
  • Non-differentiable
• Probabilistic: \( p_\theta(y|x) \)
  • Differentiable

• For binary classification

\[
p_\theta(y = 1|x) \\
p_\theta(y = 0|x) = 1 - p_\theta(y = 1|x)
\]
Loss Function

• Cross entropy loss

Discrete case: \( H(p, q) = - \sum_x p(x) \log q(x) \)

Continuous case: \( H(p, q) = - \int_x p(x) \log q(x) dx \)

• For classification problem

\( \mathcal{L}(y, x, p_\theta) = - \sum_k \delta(y = c_k) \log p_\theta(y = c_k | x) \)

\( \delta(z) = \begin{cases} 
1, & z \text{ is true} \\
0, & \text{otherwise} 
\end{cases} \)
Cross Entropy for Binary Classification

- Loss function

\[ \mathcal{L}(y, x, p_{\theta}) = -\delta(y = 1) \log p_{\theta}(y = 1|x) - \delta(y = 0) \log p_{\theta}(y = 0|x) \]

\[ = -y \log p_{\theta}(y = 1|x) - (1 - y) \log(1 - p_{\theta}(y = 1|x)) \]
Logistic Regression

• Logistic regression is a binary classification model

\[ p_\theta(y = 1|x) = \sigma(\theta^\top x) = \frac{1}{1 + e^{-\theta^\top x}} \]
\[ p_\theta(y = 0|x) = \frac{e^{-\theta^\top x}}{1 + e^{-\theta^\top x}} \]

• Cross entropy loss function

\[ L(y, x, p_\theta) = -y \log \sigma(\theta^\top x) - (1 - y) \log(1 - \sigma(\theta^\top x)) \]

• Gradient

\[ \frac{\partial L(y, x, p_\theta)}{\partial \theta} = -y \frac{1}{\sigma(\theta^\top x)} \sigma(z)(1 - \sigma(z))x - (1 - y) \frac{-1}{1 - \sigma(\theta^\top x)} \sigma(z)(1 - \sigma(z))x \]
\[ = (\sigma(\theta^\top x) - y)x \]
\[ \theta \leftarrow \theta + \eta (y - \sigma(\theta^\top x))x \]

\[ \frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z)) \]
Label Decision

- Logistic regression provides the probability

\[ p_\theta(y = 1|x) = \sigma(\theta^\top x) = \frac{1}{1 + e^{-\theta^\top x}} \]

\[ p_\theta(y = 0|x) = \frac{e^{-\theta^\top x}}{1 + e^{-\theta^\top x}} \]

- The final label of an instance is decided by setting a threshold \( h \)

\[ \hat{y} = \begin{cases} 1, & p_\theta(y = 1|x) > h \\ 0, & \text{otherwise} \end{cases} \]
Evaluation Measures

- **True / False**
  - True: prediction = label
  - False: prediction ≠ label

- **Positive / Negative**
  - Positive: predict $y = 1$
  - Negative: predict $y = 0$

<table>
<thead>
<tr>
<th>Label</th>
<th>Prediction</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>True Positive</td>
<td>False Negative</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>False Positive</td>
<td>True Negative</td>
<td></td>
</tr>
</tbody>
</table>

Class 1
- TP: if predicting 1
- FN: if predicting 0

Class 0
- FP: if predicting 1
- TN: if predicting 0
Evaluation Measures

- **Accuracy**: the ratio of cases when prediction = label

<table>
<thead>
<tr>
<th>Label</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>True Positive</td>
</tr>
<tr>
<td>0</td>
<td>False Positive</td>
</tr>
</tbody>
</table>

\[
\text{Acc} = \frac{TP + TN}{TP + TN + FP + FN}
\]
**Evaluation Measures**

- **Precision**: the ratio of true class 1 cases in those with prediction 1

  \[
  \text{Prec} = \frac{TP}{TP + FP}
  \]

- **Recall**: the ratio of cases with prediction 1 in all true class 1 cases

  \[
  \text{Rec} = \frac{TP}{TP + FN}
  \]
Evaluation Measures

• Precision-recall tradeoff

\[ \hat{y} = \begin{cases} 
1, & \text{ } \quad p_\theta(y = 1|x) > h \\
0, & \text{otherwise} 
\end{cases} \]

  • Higher threshold, higher precision, lower recall
    • Extreme case: threshold = 0.99
  • Lower threshold, lower precision, higher recall
    • Extreme case: threshold = 0

• F1 Measure

\[ F1 = \frac{2 \times \text{Prec} \times \text{Recall}}{\text{Prec} + \text{Rec}} \]
Evaluation Measures

- Ranking-based measure: Area Under ROC Curve (AUC)
Evaluation Measures

- Ranking-based measure: Area Under ROC Curve (AUC)

Perfect Prediction
AUC = 1

Random Prediction
AUC = 0.5
Evaluation Measures

- A simple example of Area Under ROC Curve (AUC)

**AUC = 0.75**
Multi-Class Classification

- Still cross entropy loss

\[
\mathcal{L}(y, x, p_\theta) = -\sum_k \delta(y = c_k) \log p_\theta(y = c_k | x)
\]

\[
\delta(z) = \begin{cases} 
1, & z \text{ is true} \\
0, & \text{otherwise}
\end{cases}
\]
Multi-Class Logistic Regression

• Class set \( C = \{c_1, c_2, \ldots, c_m\} \)

• Predicting the probability of \( p_\theta(y = c_j|x) \)

\[
p_\theta(y = c_j|x) = \frac{e^{\theta_j^T x}}{\sum_{k=1}^{m} e^{\theta_k^T x}} \quad \text{for } j = 1, \ldots, m
\]

• Softmax
  • Parameters \( \theta = \{\theta_1, \theta_2, \ldots, \theta_m\} \)
  • Can be normalized with m-1 groups of parameters
Multi-Class Logistic Regression

• Learning on one instance \((x, y = c_j)\)
  • Maximize log-likelihood
    \[
    \max_{\theta} \log p_\theta(y = c_j | x)
    \]

• Gradient
  \[
  \frac{\partial \log p_\theta(y = c_j | x)}{\partial \theta_j} = \frac{\partial}{\partial \theta_j} \log \frac{e^{\theta_j^\top x}}{\sum_{k=1}^m e^{\theta_k^\top x}}
  = x - \frac{\partial}{\partial \theta_j} \log \sum_{k=1}^m e^{\theta_k^\top x}
  = x - \frac{e^{\theta_j^\top x} x}{\sum_{k=1}^m e^{\theta_k^\top x}}
  \]
Application Case Study
Click-Through Rate (CTR) Estimation in Online Advertising
Ad Click-Through Rate Estimation

Click or not?

[http://news.ifeng.com]
User response estimation problem

• Problem definition

<table>
<thead>
<tr>
<th>One instance data</th>
<th>Corresponding label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date: 20160320</td>
<td>Click (1) or not (0)?</td>
</tr>
<tr>
<td>Hour: 14</td>
<td>Predicted CTR (0.15)</td>
</tr>
<tr>
<td>Weekday: 7</td>
<td></td>
</tr>
<tr>
<td>IP: 119.163.222.*</td>
<td></td>
</tr>
<tr>
<td>Region: England</td>
<td></td>
</tr>
<tr>
<td>City: London</td>
<td></td>
</tr>
<tr>
<td>Country: UK</td>
<td></td>
</tr>
<tr>
<td>Ad Exchange: Google</td>
<td></td>
</tr>
<tr>
<td>Domain: yahoo.co.uk</td>
<td></td>
</tr>
<tr>
<td>URL: <a href="http://www.yahoo.co.uk/abc/xyz.html">http://www.yahoo.co.uk/abc/xyz.html</a></td>
<td></td>
</tr>
<tr>
<td>OS: Windows</td>
<td></td>
</tr>
<tr>
<td>Browser: Chrome</td>
<td></td>
</tr>
<tr>
<td>Ad size: 300*250</td>
<td></td>
</tr>
<tr>
<td>Ad ID: a1890</td>
<td></td>
</tr>
<tr>
<td>User occupation: Student</td>
<td></td>
</tr>
<tr>
<td>User tags: Sports, Electronics</td>
<td></td>
</tr>
</tbody>
</table>

Date: 20160320
Hour: 14
Weekday: 7
IP: 119.163.222.*
Region: England
City: London
Country: UK
Ad Exchange: Google
Domain: yahoo.co.uk
URL: [http://www.yahoo.co.uk/abc/xyz.html](http://www.yahoo.co.uk/abc/xyz.html)
OS: Windows
Browser: Chrome
Ad size: 300*250
Ad ID: a1890
User occupation: Student
User tags: Sports, Electronics
One-Hot Binary Encoding

• A standard feature engineering paradigm

\[ x = [\text{Weekday=Friday, Gender=Male, City=Shanghai}] \]

\[ x = [0,0,0,0,1,0,0 0,1 0,0,1,0...0] \]

Sparse representation: \( x = [5:1 \ 9:1 \ 12:1] \)

• High dimensional sparse binary feature vector
  • Usually higher than 1M dimensions, even 1B dimensions
  • Extremely sparse
Training/Validation/Test Data

• Examples (in LibSVM format)

1 5:1 9:1 12:1 45:1 154:1 509:1 4089:1 45314:1 988576:1
0 2:1 7:1 18:1 34:1 176:1 510:1 3879:1 71310:1 818034:1
...

• Training/Validation/Test data split
  • Sort data by time
  • Train:validation:test = 8:1:1
  • Shuffle training data
Training Logistic Regression

• Logistic regression is a binary classification model

\[ p_\theta(y = 1|x) = \sigma(\theta^\top x) = \frac{1}{1 + e^{-\theta^\top x}} \]

• Cross entropy loss function with L2 regularization

\[ \mathcal{L}(y, x, p_\theta) = -y \log \sigma(\theta^\top x) - (1 - y) \log(1 - \sigma(\theta^\top x)) + \frac{\lambda}{2} \|\theta\|^2 \]

• Parameter learning

\[ \theta \leftarrow (1 - \lambda \eta)\theta + \eta(y - \sigma(\theta^\top x))x \]

• Only update non-zero entries
Experimental Results

• Datasets
  • Criteo Terabyte Dataset
    • 13 numerical fields, 26 categorical fields
    • 7 consecutive days out of 24 days in total (about 300 GB) during 2014
    • 79.4M impressions, 1.6M clicks after negative down sampling
  
  • iPinYou Dataset
    • 65 categorical fields
    • 10 consecutive days during 2013
    • 19.5M impressions, 937.7K clicks without negative down sampling
Performance

<table>
<thead>
<tr>
<th>Model</th>
<th>Linearity</th>
<th>AUC</th>
<th>Log Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Criteo</td>
<td>iPinYou</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>Linear</td>
<td>71.48%</td>
<td>73.43%</td>
</tr>
<tr>
<td>Factorization Machine</td>
<td>Bi-linear</td>
<td>72.20%</td>
<td>75.52%</td>
</tr>
<tr>
<td>Deep Neural Networks</td>
<td>Non-linear</td>
<td>75.66%</td>
<td>76.19%</td>
</tr>
</tbody>
</table>

- Compared with non-linear models, linear models
  - Pros: standardized, easily understood and implemented, efficient and scalable
  - Cons: modeling limit (feature independent assumption), cannot explore feature interactions

[Yanru Qu et al. Product-based Neural Networks for User Response Prediction. ICDM 2016.]
Generalized Linear Models
Review: Linear Regression

\[ X = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(n)} \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \ldots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \ldots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \ldots & x_d^{(n)} \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_d \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \]

- Prediction \( \hat{y} = X\theta = \begin{bmatrix} x^{(1)} \theta \\ x^{(2)} \theta \\ \vdots \\ x^{(n)} \theta \end{bmatrix} \)

- Objective \( J(\theta) = \frac{1}{2}(y - \hat{y})^\top (y - \hat{y}) = \frac{1}{2}(y - X\theta)^\top (y - X\theta) \)
Review: Matrix Form of Linear Reg.

• Objective

\[ J(\theta) = \frac{1}{2} (y - X\theta)^\top (y - X\theta) \quad \min_{\theta} J(\theta) \]

• Gradient

\[ \frac{\partial J(\theta)}{\partial \theta} = -X^\top (y - X\theta) \]

• Solution

\[ \frac{\partial J(\theta)}{\partial \theta} = 0 \quad \rightarrow \quad X^\top (y - X\theta) = 0 \]

\[ \rightarrow \quad X^\top y = X^\top X\theta \]

\[ \rightarrow \quad \hat{\theta} = (X^\top X)^{-1} X^\top y \]
Generalized Linear Models

• Dependence

\[ y = f(\theta^\top \phi(x)) \]

• Feature mapping function \( \phi(x) : \mathbb{R}^d \rightarrow \mathbb{R}^h \)

• Mapped feature matrix \( \Phi_{n \times h} \)

\[
\Phi = \begin{bmatrix}
\phi(x^{(1)}) \\
\phi(x^{(2)}) \\
\vdots \\
\phi(x^{(i)}) \\
\vdots \\
\phi(x^{(n)})
\end{bmatrix} = \begin{bmatrix}
\phi_1(x^{(1)}) & \phi_2(x^{(1)}) & \cdots & \phi_h(x^{(1)}) \\
\phi_1(x^{(2)}) & \phi_2(x^{(2)}) & \cdots & \phi_h(x^{(2)}) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(x^{(i)}) & \phi_2(x^{(i)}) & \cdots & \phi_h(x^{(i)}) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(x^{(n)}) & \phi_2(x^{(n)}) & \cdots & \phi_h(x^{(n)})
\end{bmatrix}
\]
Matrix Form of Kernel Linear Regression

- Objective
  \[ J(\theta) = \frac{1}{2} (y - \Phi \theta)^\top (y - \Phi \theta) \quad \min_{\theta} J(\theta) \]

- Gradient
  \[ \frac{\partial J(\theta)}{\partial \theta} = -\Phi^\top (y - \Phi \theta) \]

- Solution
  \[ \frac{\partial J(\theta)}{\partial \theta} = 0 \quad \Rightarrow \quad \Phi^\top (y - \Phi \theta) = 0 \]
  \[ \Rightarrow \quad \Phi^\top y = \Phi^\top \Phi \theta \]
  \[ \Rightarrow \quad \hat{\theta} = (\Phi^\top \Phi)^{-1} \Phi^\top y \]
Matrix Form of Kernel Linear Regression

• With the Algebra trick

\[(P^{-1} + B^\top R^{-1} B)^{-1} B^\top R^{-1} = PB^\top (BPB^\top + R)^{-1}\]

• The optimal parameters with L2 regularization

\[\hat{\theta} = (\Phi^\top \Phi + \lambda I_n)^{-1} \Phi^\top y\]
\[= \Phi^\top (\Phi \Phi^\top + \lambda I_n)^{-1} y\]

for prediction, we never actually need access \(\Phi\)

\[\hat{y} = \Phi \hat{\theta} = \Phi \Phi^\top (\Phi \Phi^\top + \lambda I_n)^{-1} y\]
\[= K(K + \lambda I_n)^{-1} y\]

where the kernel matrix \(K = \{K(x^{(i)}, x^{(j)})\}\)