DDA3020 Machine Learning
Lecture 06 Logistic Regression

Jicong Fan
School of Data Science, CUHK-SZ

09/25/2025

Outline

1. Review of last week
2. Classification and representation
3. Logistic regression
4. Regularized logistic regression
5. Probabilistic perspective of logistic regression
6. Summary: linear regression vs. logistic regression

Linear regression: deterministic perspective

- **Linear hypothesis function**: $f_{\mathbf{w},w_0}(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} + w_0$, or, simply $f_{\mathbf{w}}(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}$ by concatenating w_0 and $\mathbf{w}$ together and augmenting $\mathbf{x}$ to $[1; \mathbf{x}]$
- **Linear regression** by minimizing the residual sum of squares (RSS):

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} J(\mathbf{w}), \text{ where } J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}(\mathbf{x}_i^\top \mathbf{w} - y_i)^2 = \frac{1}{2}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$$

- **Two solutions**:
  - Closed-form solution: $\mathbf{w}^* = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}$
  - Gradient descent: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \mathbf{X}^\top(\mathbf{X}\mathbf{w} - \mathbf{y})$, for multiple iterations until convergence

Linear regression: probabilistic perspective

- We assume that: $y = \mathbf{w}^\top \mathbf{x} + e$, where $e \sim \mathcal{N}(0, \sigma^2)$ is called **observation noise** or **residual error**
- y is also a random variable, and its conditional probability is

$$p(y|\mathbf{x}, \mathbf{w}) = \mathcal{N}(\mathbf{w}^\top \mathbf{x}, \sigma^2)$$

- **Maximum log-likelihood estimation**:

$$\begin{aligned}
\mathbf{w}_{MLE} &= \arg\max_{\mathbf{w}} \log \mathcal{L}(\mathbf{w}; D) = \arg\max_{\mathbf{w}} \log \Big(\prod_i^m p(y_i|\mathbf{x}_i, \mathbf{w}) \Big) & (1) \\
&= \arg\max_{\mathbf{w}} \sum_i^m \log p(y_i|\mathbf{x}_i, \mathbf{w}) = \arg\max_{\mathbf{w}} \sum_i^m \log \mathcal{N}(\mathbf{w}^\top \mathbf{x}_i, \sigma^2) & (2) \\
&= \arg\max_{\mathbf{w}} - \log(\sigma^m (2\pi)^{\frac{m}{2}}) - \frac{1}{2\sigma^2} \sum_i^m (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 & (3) \\
&= \arg\min_{\mathbf{w}} \frac{1}{2} \sum_i^m (y_i - \mathbf{w}^\top \mathbf{x}_i)^2, & (4)
\end{aligned}$$

Variants of linear regression

- **Ridge regression** to avoid over-fitting, through MAP estimation:

$$\begin{aligned} \mathbf{w}_{MAP} &= \arg\max_{\mathbf{w}} \sum_{i=1}^{m} \log p(y_i|\mathbf{x}_i, \mathbf{w}) + \log p(\mathbf{w}) && (5) \\ &= \arg\max_{\mathbf{w}} \sum_{i=1}^{m} \log \mathcal{N}(\mathbf{w}^\top \mathbf{x}_i, \sigma^2) + \mathcal{N}(\mathbf{w}|\mathbf{0}, \tau^2\mathbf{I}) && (6) \\ &\equiv \arg\min_{\mathbf{w}} \sum_{i=1}^{m} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \lambda\|\mathbf{w}\|_2^2. && (7) \end{aligned}$$

- **Polynomial regression**: linear model with basis expansion $\phi(\mathbf{x})$

$$\begin{aligned} f_{\mathbf{w}}(\mathbf{x}) &= w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d}\sum_{j=1}^{d} w_{ij} x_i x_j + \sum_{i=1}^{d}\sum_{j=1}^{d}\sum_{k=1}^{d} w_{ijk} x_i x_j x_k + \dots \\ &= \phi(\mathbf{x})^\top \mathbf{w}, && (8) \\ \phi(\mathbf{x}) &= [1, x_1, \dots, x_d, \dots, x_i x_j, \dots, x_i x_j x_k, \dots]^\top, \\ \mathbf{w} &= [w_0, w_1, \dots, w_d, \dots, w_{ij}, \dots, w_{ijk}, \dots]^\top. \end{aligned}$$

Variants of linear regression

- **Lasso regression** to obtain sparse model,

$$\mathbf{w}_{MAP} = \arg\max_{\mathbf{w}} \sum_{i}^{m} \log \mathcal{N}(\mathbf{w}^\top \mathbf{x}_i, \sigma^2) + \mathrm{Lap}(\mathbf{w}|\mathbf{0}, b) \tag{9}$$

$$= \arg\min_{\mathbf{w}} \sum_{i=1}^{m} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \lambda \|\mathbf{w}\|_1. \tag{10}$$

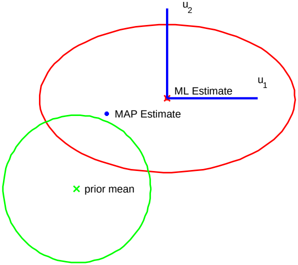
- **Robust regression** for data with outliers:

$$\mathbf{w}_{MLE} = \arg\min_{\mathbf{w}} \sum_{i=1}^{m} |\mathbf{w}^\top \mathbf{x}_i - y_i| \tag{11}$$

Summary of different linear regressions

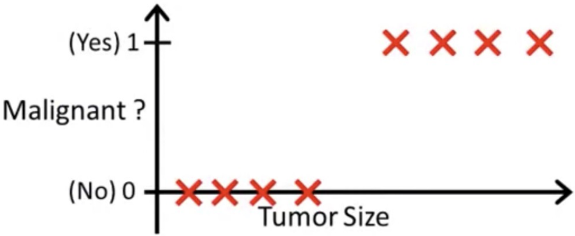
Note that the uniform distribution will not change the mode of the likelihood. Thus, MAP estimation with a uniform prior corresponds to MLE.

$p(y\|\mathbf{x}, \mathbf{w})$	$p(\mathbf{w})$	regression method
Gaussian	Uniform	Least squares
Gaussian	Gaussian	Ridge regression
Gaussian	Laplace	Lasso regression
Laplace	Uniform	Robust regression
Student	Uniform	Robust regression

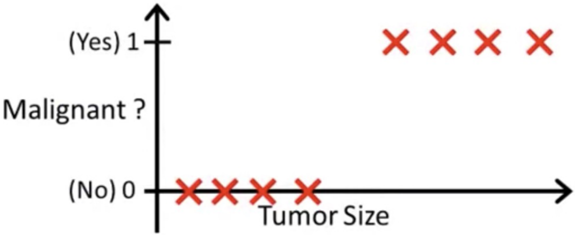


Classification

- Classification: classifying input data into discrete states
  - Email filtering: spam / not spam?
  - Weather forecast: sunny / not sunny?
  - Tumor: malignant / benign?
- The label $y \in \{0, 1\}$:
  - $y = 0$: negative class, *e.g.*, not spam, not sunny, benign
  - $y = 1$: positive class, *e.g.*, spam, sunny, malignant

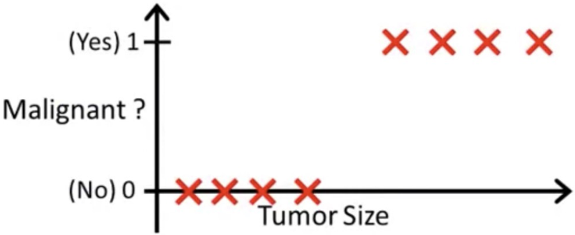


Threshold classifier with linear regression



- We assume a linear hypothesis function $f_{\mathbf{w}}(\boldsymbol{x}) = \mathbf{x}^\top \mathbf{w}$
- A simple threshold classifier with this hypothesis function is
  - If $f_{\mathbf{w}}(\boldsymbol{x}) > 0.5$, then $y = 1$, *i.e.*, malignant tumor
  - If $f_{\mathbf{w}}(\boldsymbol{x}) < 0.5$, then $y = 0$, *i.e.*, benign tumor

Threshold classifier with linear regression



- It seems that the simple threshold classifier with linear regression works well on this classification task
- However, if there is a positive sample with a very large tumor size (plot above), what will happen?
- The hypothesis function will be **significantly changed**, causing some positive samples to be misclassified as negative (not malignant). How to handle it? Adjusting the threshold value, or adopting robust linear regression.

Threshold classifier with linear regression

- But there is still something wired.
- Our goal is to predict $y \in \{0, 1\}$, but the prediction could be $f_{\mathbf{w}}(\mathbf{x}) > 1$ or $f_{\mathbf{w}}(\mathbf{x}) < 0$, which does not serve our purpose.
- A desired hypothesis function for this task should be $f_{\mathbf{w}}(\mathbf{x}) \in [0, 1]$.

Threshold classifier with linear regression

Exercise: Which statements are true?

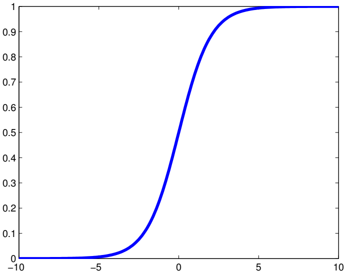
- If linear regression doesn't work well like the above example, feature scaling may help
- If the training set satisfies that all $y_i \in [0, 1]$ for all points $(\mathbf{x}_i, y_i)$, then the linear hypothesis function $f_{\mathbf{w}}(\mathbf{x}) \in [0, 1]$ for all values of $\mathbf{x}_i$
- None of the above is correct

Hypothesis representation

- A desired hypothesis function for this task should be $f_{\mathbf{w}}(\mathbf{x}) \in [0,1]$
- To this end, we introduce a novel function, as follows:

$$f_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w}^\top \mathbf{x}) \in [0,1],\ g(z) = \frac{1}{1+\exp(-z)},$$

where $g(\cdot)$ is called sigmoid function or logistic function (shown below)

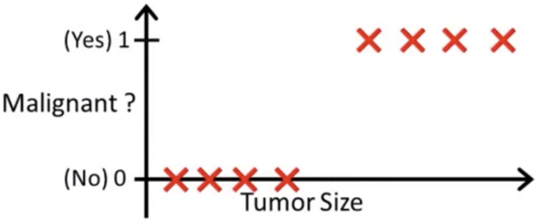


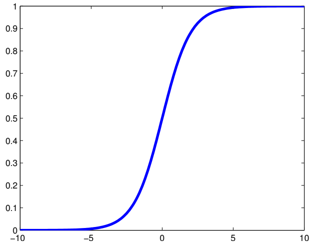
* Note that $\mathbf{w} = [w_0, w_1, \ldots, w_d]^\top$ and $\mathbf{x} = [1, x_1, \ldots, x_d]^\top$.

Hypothesis representation

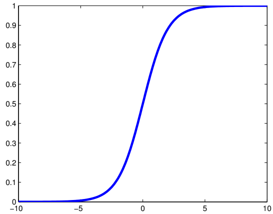
- **Interpretation of sigmoid/logistic function**
  - $f_{\mathbf{w}}(\mathbf{x})$ = estimated probability that $y = 1$ of input $\mathbf{x}$.
- For example (plot below), if $f_{\mathbf{w}}(\mathbf{x}) = 0.8$, then it means that a patient with tumor size $\mathbf{x}$ has 80% chance of tumor being malignant. In this task, a larger tumor size has a larger chance/probability of being a malignant tumor.
- Thus, we can say that

$$f_{\mathbf{w}}(\mathbf{x}) = P(y = 1|\mathbf{x}; \mathbf{w}).$$





Decision boundary

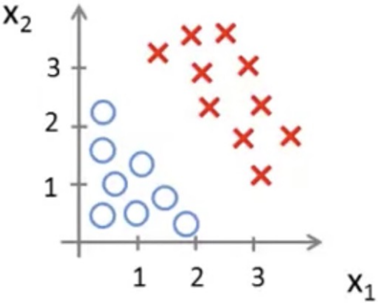


- In logistic regression, we have

$$f_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w}^\top \mathbf{x}) = P(y = 1|\mathbf{x}; \mathbf{w}) \in [0, 1], \quad g(z) = \frac{1}{1 + \exp(-z)}.$$

- Suppose that if $f_{\mathbf{w}}(\mathbf{x}) \geq 0.5$, then we predict $y = 1$; if $f_{\mathbf{w}}(\mathbf{x}) < 0.5$, then we predict $y = 0$
- Correspondingly, if $\mathbf{w}^\top \mathbf{x} \geq 0$, we predict $y = 1$; if $\mathbf{w}^\top \mathbf{x} < 0$, then we predict $y = 0$.
- It determines the decision boundary, which is the curve/hyper-plane corresponding to $f_{\mathbf{w}}(\mathbf{x}) = 0.5$, or $\mathbf{w}^\top \mathbf{x} = 0$

Decision boundary



- $f_{\mathbf{w}}(\mathbf{x}) = g(w_0 + w_1x_1 + w_2x_2) = g(-3 + x_1 + x_2)$
- Predict $y = 1$ if $-3 + x_1 + x_2 \geq 0$ (plot above)

Decision boundary

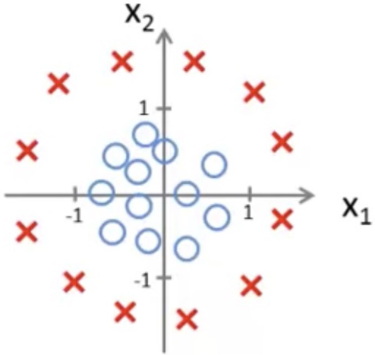


Figure: Non-linear decision boundary

- $f_{\mathbf{w}}(\mathbf{x}) = g(w_0 + w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_2^2) = g(-1 + x_1^2 + x_2^2)$
- Predict $y = 1$ if $-1 + x_1^2 + x_2^2 \geq 0$ (plot above)

Cost function

- Training set: m training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^m$
- Hypothesis function: $f_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}^\top \mathbf{x})}$
- Cost function:
  - **Linear regression**: $J(\mathbf{w}) = \frac{1}{2m}\sum_{i=1}^m (f_{\mathbf{w}}(\mathbf{x}_i) - y_i)^2 = \frac{1}{2m}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$, which is called ℓ_2 **loss** or **residual sum of squares**
  - It is convex *w.r.t.* $\mathbf{w}$ for linear regression
  - **Logistic regression**: If we adopt the same cost function for logistic regression, we have

$$J(\mathbf{w}) = \frac{1}{2m}\sum_{i}^{m}(g(\mathbf{w}^\top \boldsymbol{x}_i) - y_i)^2.$$

    However, it is non-convex *w.r.t.* $\mathbf{w}$.

`Exercise 1:` Prove the ℓ_2 loss is convex *w.r.t.* $\mathbf{w}$ for linear regression.
`Exercise 2:` Prove the ℓ_2 loss is non-convex *w.r.t.* $\mathbf{w}$ for logistic regression.

Cost function

`Exercise 1:` Prove the ℓ_2 loss is convex *w.r.t.* $\mathbf{w}$ for linear regression.
`Exercise 2:` Prove the ℓ_2 loss is non-convex *w.r.t.* $\mathbf{w}$ for logistic regression.

Cost function

- Cross-entropy:

$$H(p,q) = -\int_x p(x)\log(q(x))dx \quad \text{or} \quad -\sum_x p(x)\log(q(x)),$$

  where $p(x), q(x)$ are **probability density functions** (PDF) of x if x is a continuous random variable, or, **probability mass functions** if x is a discrete random variable

- We set

$$\text{ground-truth posterior probability}: \; y(\mathbf{x}) = P(y=1|\mathbf{x}),$$
$$\text{predicted posterior probability}: \; f_{\mathbf{w}}(\mathbf{x}) = P(y=1|\mathbf{x};\mathbf{w}).$$

- Cross-entropy loss:

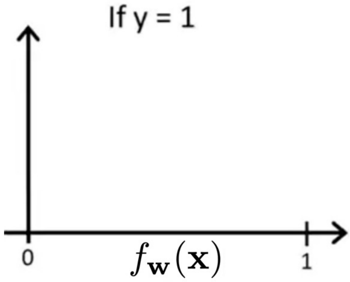
$$\begin{aligned}
&\text{cost}\big(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})\big) = H\big(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})\big) \\
=& -P(y=1|\mathbf{x})\cdot\log P(y=1|\mathbf{x};\mathbf{w}) - P(y=0|\mathbf{x})\cdot\log P(y=0|\mathbf{x};\mathbf{w}) \\
=& \begin{cases} -\log(f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 1 \\ -\log(1-f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 0 \end{cases}
\end{aligned}$$

Cost function for logistic regression

- Cross-entropy loss:

$$\text{cost}\big(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})\big) = \begin{cases} -\log(f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 1 \\ -\log(1 - f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 0 \end{cases}$$

- For $y = 1$, if $f_{\mathbf{w}}(\mathbf{x}) = 1$, *i.e.*, $P(y = 1|\mathbf{x}; \mathbf{w}) = 1$, then the prediction equals to the ground-truth label, the cost is 0.
- For $y = 1$, if $f_{\mathbf{w}}(\mathbf{x}) \to 0$, *i.e.*, $P(y = 1|\mathbf{x}; \mathbf{w}) \to 0$, then it should be penalized with a very large cost. Here we have $\text{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) \to \infty$.

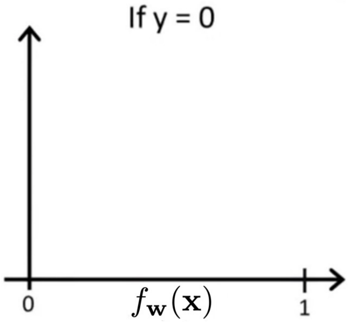


Cost function for logistic regression

- **Cross-entropy loss:**

$$\text{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) = \begin{cases} -\log(f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 1 \\ -\log(1 - f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 0 \end{cases}$$

- For $y = 0$, if $f_{\mathbf{w}}(\mathbf{x}) = 0$, *i.e.*, $P(y = 1|\mathbf{x}; \mathbf{w}) = 0$, then the prediction equals to the ground-truth label, the cost is 0
- For $y = 0$, if $f_{\mathbf{w}}(\mathbf{x}) \to 1$, *i.e.*, $P(y = 1|\mathbf{x}; \mathbf{w}) \to 1$, then it should be penalized with a very large cost. Here we have $\text{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) \to \infty$



Cost function for logistic regression

- **Cross-entropy loss**:

$$\mathrm{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) = \begin{cases} -\log(f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 1 \\ -\log(1 - f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 0 \end{cases}$$

Exercise: Which states are true?

- If $f_{\mathbf{w}}(\mathbf{x}) = y$, then $\mathrm{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) = 0$ for both $y = 0$ and $y = 1$
- If $y = 0$, then $\mathrm{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) \to \infty$ as $f_{\mathbf{w}}(\mathbf{x}) \to 1$
- If $y = 0$, then $\mathrm{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) \to \infty$ as $f_{\mathbf{w}}(\mathbf{x}) \to 0$
- Regardless whether $y = 0$ or $y = 1$, if $f_{\mathbf{w}}(\mathbf{x}) = 0.5$, then $\mathrm{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) > 0$

Cost function of logistic regression

- **Cost function of logistic regression**

$$J(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} \text{cost}(y_i, f_{\mathbf{w}}(\mathbf{x}_i)),$$

$$\text{cost}(y(\mathbf{x}), f_{\mathbf{w}}(\mathbf{x})) = \begin{cases} -\log(f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 1 \\ -\log(1 - f_{\mathbf{w}}(\mathbf{x})), & \text{if } y(\mathbf{x}) = 0 \end{cases}$$

- The above cost function can be simplified as follows

$$J(\mathbf{w}) = -\frac{1}{m}\sum_{i=1}^{m} \left[y_i \log(f_{\mathbf{w}}(\mathbf{x}_i)) + (1 - y_i) \log(1 - f_{\mathbf{w}}(\mathbf{x}_i)) \right].$$

`Exercise: Please prove that` $J(\mathbf{w})$ `is convex` *w.r.t.* $\mathbf{w}$.

Gradient descent for logistic regression

- Learning $\mathbf{w}$ by minimize $J(\mathbf{w})$, *i.e.*,

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} J(\mathbf{w}) = -\frac{1}{m}\sum_{i=1}^{m}\big[y_i \log(f_{\mathbf{w}}(\mathbf{x}_i)) + (1-y_i)\log(1-f_{\mathbf{w}}(\mathbf{x}_i))\big].$$

- **Gradient descent**: repeat the following update until convergence

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \nabla_{\mathbf{w}} J(\mathbf{w})$$

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}[f_{\mathbf{w}}(\mathbf{x}_i) - y_i]\mathbf{x}_i$$

- How to define convergence? Calculating the changes of $J(\mathbf{w})$ or $\mathbf{w}$ in the last K steps, if the change is lower than a threshold than it can be seen as convergence. Remember that choosing a suitable learning rate α is important to achieve a good converged solution.

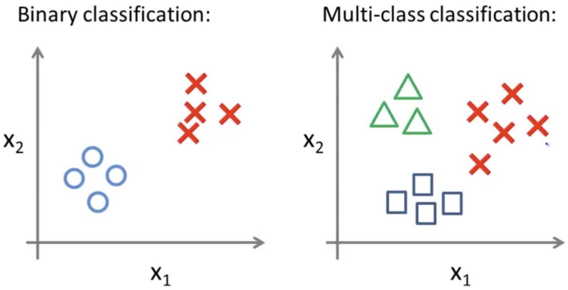
Gradient descent for logistic regression

Exercise: Suppose you are running a logistic regression model, and you should observe the learning procedure to find a suitable learning rate α. Which of the following is reasonable to make sure α is set properly and that the gradient descent is running correctly?

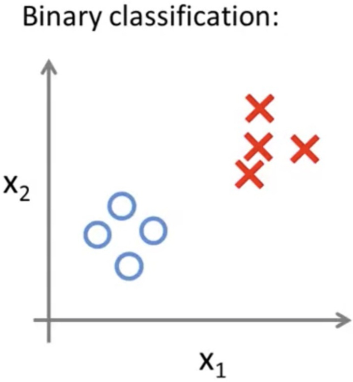
- Plot $J(\mathbf{w}) = -\frac{1}{m}\sum_i^m (y_i - f_{\mathbf{w}}(\mathbf{x}_i))^2$ as a function of the number of iterations (*i.e.*, the horizontal axis is the iteration number) and make sure $J(\mathbf{w})$ is decreasing on every iteration.
- Plot $J(\mathbf{w}) = -\frac{1}{m}\sum_i^m \left[y_i \log(f_{\mathbf{w}}(\mathbf{x}_i)) + (1-y_i)\log(1-f_{\mathbf{w}}(\mathbf{x}_i))\right]$ as a function of the number of iterations (*i.e.*, the horizontal axis is the iteration number) and make sure $J(\mathbf{w})$ is decreasing on every iteration.
- Plot $J(\mathbf{w})$ as a function of $\mathbf{w}$ and make sure it is decreasing on every iteration.
- Plot $J(\mathbf{w})$ as a function of $\mathbf{w}$ and make sure it is convex.

Multi-class classification

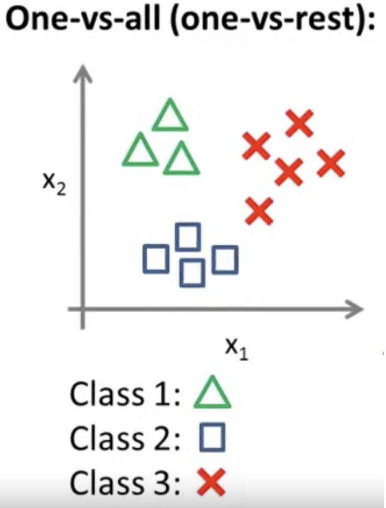
- Binary classification: in above examples and derivations, we only consider the binary classification problem, *i.e.*, $y \in \{0, 1\}$.
- Multi-class/multi-category classification: however, many practical problems involve with multi-category outputs, *i.e.*, $y \in \{1, \ldots, C\}$:
  - **Weather forecast**: sunny, cloudy, rain, snow
  - **Email tagging**: work, friends, families, hobby

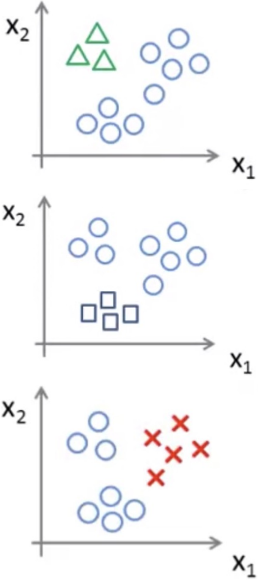


Multi-class classification



Multi-class classification: one-vs-all





Multi-class classification: one-vs-all

One-vs-all logistic regression:

- Train a binary logistic regression $f_{\mathbf{w}_j}(\cdot)$ for each class j, by setting all samples of other classes as negative class
- For a new testing sample $\mathbf{x}$, predict its class as $\arg\max_j f_{\mathbf{w}_j}(\mathbf{x})$.

Pros: Easy to implement

Cons: The training cost is too high and is difficult to scale to tasks with a large number of classes.

Multi-class classification: Softmax regression

- Softmax function:

$$f_{\mathbf{W}}^{(j)}(\mathbf{x}) = \frac{\exp(\mathbf{w}_j^\top \mathbf{x})}{\sum_{c=1}^{C} \exp(\mathbf{w}_c^\top \mathbf{x})} = P(y = j|\mathbf{x}; \mathbf{W}), \tag{12}$$

  where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_C] \in \mathbb{R}^{(d+1)\times C}$ with C being the number of classes and $\mathbf{x} = [1, x_1, \ldots, x_d]^\top$ is the augmented feature vector. For simplicity, in the following we write $f_{\mathbf{W}}^{(j)}(\cdot)$ as $f_{\mathbf{w}_j}(\cdot)$.

- Cost function:

$$J(\mathbf{W}) = -\frac{1}{m} \sum_{i}^{m} \sum_{j}^{C} \left[\mathbb{I}(y_i = j) \log(f_{\mathbf{w}_j}(\mathbf{x}_i))\right], \tag{13}$$

  where $\mathbb{I}(a) = 1$ if a is true, otherwise $\mathbb{I}(a) = 0$.

Multi-class classification: Softmax regression

- It can also be optimized by gradient descent:

$$\mathbf{w}_j \leftarrow \mathbf{w}_j - \alpha \frac{\partial J(\mathbf{W})}{\partial \mathbf{w}_j},$$

$$\text{hint: } \frac{\partial J(\mathbf{W})}{\partial \mathbf{w}_j} = -\frac{1}{m} \sum_i^m \left[\frac{\mathbb{I}(y_i = j)}{f_{\mathbf{w}_j}(\mathbf{x}_i)} \cdot \frac{\nabla f_{\mathbf{w}_j}(\mathbf{x}_i)}{\nabla \mathbf{w}_j} + \sum_{c \neq j}^{C} \frac{\mathbb{I}(y_i = c)}{f_{\mathbf{w}_c}(\mathbf{x}_i)} \cdot \frac{\nabla f_{\mathbf{w}_c}(\mathbf{x}_i)}{\nabla \mathbf{w}_j} \right]$$

$$\Longrightarrow \frac{\partial J(\mathbf{W})}{\partial \mathbf{w}_j} = \frac{1}{m} \sum_i^m \big(f_{\mathbf{w}_j}(\mathbf{x}_i) - \mathbb{I}(y_i = j) \big) \mathbf{x}_i \tag{14}$$

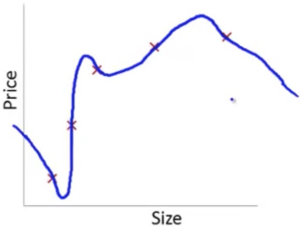
Note: $\{\mathbf{w}_c\}_{c=1}^C$ should be updated in parallel, rather than sequentially.

Overfitting in linear regression

Overfitting in linear regression

Addressing overfitting:

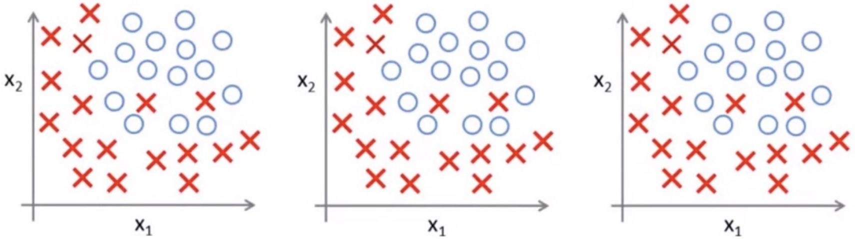
$x_1 =$ size of house
$x_2 =$ no. of bedrooms
$x_3 =$ no. of floors
$x_4 =$ age of house
$x_5 =$ average income in neighborhood
$x_6 =$ kitchen size
$\vdots$
x_{100}



Overfitting: If we have too many features, the learned hypothesis may fit the training data very well (low bias), but fail to generalize to new examples.

Overfitting in logistic regression

Example: Logistic regression



$f_{\mathbf{w}}(\mathbf{x}) = g(w_0 + w_1 x_1 + w_2 x_2)$

$f_{\mathbf{w}}(\mathbf{x}) = g(w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2$

$f_{\mathbf{w}}(\mathbf{x}) = g(w_0 + w_1 x_1 + w_2 x_1^2 + w_3 x_1^2 x_2 + w_4 x_1^2 x_2^2 + w_5 x_1^2 x_2^3 + w_6 x_1^3 x_2 + \ldots$

Under-fitting Good-fitting Over-fitting

Addressing Overfitting

Generally, there are two approaches to address the overfitting problem, including:

- Reducing the number of features:
  - Feature selection
  - Dimensionality reduction (introduced in later lectures)
- Regularization:
  - Keep all features, but reduce the magnitude/value of each parameter, such that each feature contributes a bit to predict y

In the following, we will focus on the regularization-based approach.

Regularized logistic regression

- The objective function of the regularized logistic regression is formulated as follows

$$\bar{J}(\mathbf{w}) = J(\mathbf{w}) + \frac{\lambda}{2m} \sum_{j=1}^{d} w_j^2$$

$$= -\frac{1}{m} \sum_{i}^{m} \left[y_i \log(f_{\mathbf{w}}(\mathbf{x}_i)) + (1 - y_i) \log(1 - f_{\mathbf{w}}(\mathbf{x}_i)) \right] + \frac{\lambda}{2m} \sum_{j=1}^{d} w_j^2.$$

  Note: the bias parameter w_0 is not regularized/penalized.

- The above objective function can also be solved by gradient descent, as follows

$$w_0 \leftarrow w_0 - \frac{\alpha}{m} \sum_{i=1}^{m} (f_{\mathbf{w}}(\mathbf{x}_i) - y_i) \cdot \mathbf{x}_i(0), \text{ where } \mathbf{x}_i(0) = 1, \forall i$$

$$w_j \leftarrow w_j - \frac{\alpha}{m} \Big[\sum_{i=1}^{m} (f_{\mathbf{w}}(\mathbf{x}_i) - y_i) \cdot \mathbf{x}_i(j) + \lambda \cdot w_j \Big],$$

  where $\mathbf{x}_i(j)$ denotes the j-th entry of $\mathbf{x}_i$, and $j = 0, \ldots, d$.

Regularized logistic regression

Exercise: When using regularized logistic regression, which of these is the best way to monitor whether gradient descent is working correctly?

- Plot $J(\mathbf{w})$ as a function of the number of iterations and make sure it's decreasing
- Plot $J(\mathbf{w}) - \frac{\lambda}{2m}\sum_{j=1}^{d} w_j^2$ as a function of the number of iterations and make sure it's decreasing
- Plot $J(\mathbf{w}) + \frac{\lambda}{2m}\sum_{j=1}^{d} w_j^2$ as a function of the number of iterations and make sure it's decreasing
- Plot $\sum_{j=1}^{d} w_j^2$ as a function of the number of iterations and make sure it's decreasing

Logistic regression: probabilistic modeling

- Behind **logistic regression for binary classification**, we assume that both the feature $\mathbf{x}$ and and the label y are random variables, as follows

$$\mu(\mathbf{x}; \mathbf{w}) = \text{Sigmoid}(\mathbf{w}^\top \mathbf{x}),$$
$$y(\mathbf{x}; \mathbf{w}) \sim \text{Bernoulli}(\mu(\mathbf{x}; \mathbf{w})).$$

- Then, we have

$$P(y|\mathbf{x}; \mathbf{w}) = \begin{cases} \mu & \text{if } y = 1, \\ 1 - \mu & \text{if } y = 0. \end{cases}$$

- The log-likelihood function of $P(y|\mathbf{x}; \mathbf{w})$ is formulated as

$$\mathcal{L}(\mathbf{w}) = y \log(\mu) + (1 - y) \log(1 - \mu).$$

- Thus, we obtain

$$\max_{\mathbf{w}} \mathcal{L}(\mathbf{w}) \equiv \min_{\mathbf{w}} J(\mathbf{w}).$$

Logistic regression: probabilistic modeling

- Behind logistic regression, we assume that

$$\mu(\mathbf{x};\mathbf{w}) = \text{Sigmoid}(\mathbf{w}^\top \mathbf{x}),$$
$$y(\mathbf{x};\mathbf{w}) \sim \text{Bernoulli}(\mu(\mathbf{x};\mathbf{w})).$$

- ℓ_2-regularized logistic regression: we further assume $\mathbf{w} \sim \mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma^2\mathbf{I})$ (excluding the bias w_0), then we have

$$\max_{\mathbf{w}} \mathcal{L}(\mathbf{w}) + \log \mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma^2\mathbf{I}) \equiv \min_{\mathbf{w}} J(\mathbf{w}) + \frac{\lambda}{2m}\sum_{j=1}^{d} w_j^2.$$

- ℓ_1-regularized logistic regression: if we assume $\mathbf{w} \sim \text{Laplace}(\mathbf{w}|\mathbf{0}, b)$ (excluding the bias w_0), then we have

$$\max_{\mathbf{w}} \mathcal{L}(\mathbf{w}) + \log \text{Laplace}(\mathbf{w}|\mathbf{0}, b) \equiv \min_{\mathbf{w}} J(\mathbf{w}) + \frac{\lambda}{2m}\sum_{j=1}^{d} |w_j|.$$

Summary: linear regression vs. logistic regression

	Linear regression	Logistic regression
Task	regression	classification
Hypothesis $f_{\mathbf{w}}(\mathbf{x})$	$\mathbf{w}^\top \mathbf{x} \in (-\infty, \infty)$	$g(\mathbf{w}^\top \mathbf{x}) \in [0, 1]$
Objective $J(\mathbf{w})$	$\frac{1}{2m} \sum_i^m (y_i - \mathbf{w}^\top \mathbf{x}_i)^2$	$-\frac{1}{m} \sum_{i=1}^m \big[y_i \log(f_{\mathbf{w}}(\mathbf{x}_i))$ $+(1-y_i)\log(1-f_{\mathbf{w}}(\mathbf{x}_i))\big]$
Solution	closed-form or gradient descent	gradient descent

Note that: For each variant of linear/logistic regression, you can derive it from both the deterministic and the probabilistic perspectives.

Own reading: Both linear regression and logistic regression are special cases of generalized linear models. If interested, you can find more details in Section 4 of the book "Pattern Recognition and Machine Learning", Bishop, 2006.