CS 474/574 Machine Learning
5. Regression

Prof. Dr. Forrest Sheng Bao
Dept. of Computer Science
Iowa State University
Ames, IA, USA

February 28, 2023

Agenda

- Linear Regression
- Logistic Regression (that can be used for classification)
- SVM Regression
- Loss functions
- Something about regularization

Regression vs. Classification

- The very first demo ($h = \frac{1}{2}gt^2$) is regression.
- Regression is also supervised ML, thus given an $f = f(x)$, we want to contruct another $\hat{f}$ such that $\hat{y}$ and y is very close.
- The only difference is that in Classification, the y is usually discrete.
- While in Regression, the y is in a range - could be as large as the entire real number domain.
- Hence in regression, the output is usually not called **labels** but **targets**.
- And thus, the model is not called a **classifier** but a **regressor**.

Linear Regression

- We assume a linear relationship between two sets of variables $\mathbf{x}$ and $\mathbf{y}$
- Thus, the prediction is the same as in classification $\hat{y} = \mathbf{w}^T\mathbf{x}$.
- How do we count the loss? We could use mean squared error (MSE) again:

$$\sum_i (\mathbf{w}^T\mathbf{x}_i - y)^2$$

- A criterion often used to judge a regression model is **correlation coefficient**.

Logistic Regression I

- Logistic regression is nonlinear. It was not originally proposed for ML, but as a way to model the probability of a random events.
- It is frequently used for classification but its nature is regression.
- The log odds, or logit (**log**istic un**it**) for an event A of probability $P(A)$ is $\log\left(\frac{P(A)}{1-P(A)}\right)$.
- Use a linear model to fit the log odds: $\log\left(\frac{P(A)}{1-P(A)}\right) = b_0 + b_1x_1 + b_2x_2 + \cdots$.
- Solve it, we can express the probability as $P(A) = 1/\left(1 + e^{-(b_0+b_1x_1+b_2x_2+\cdots)}\right)$
- If we set a threshold on $P(A)$, e.g., $P(\text{the fruit is a banana}) > 0$, then we can use this function as a classifier.
- The fraction part is often called the **sigmoid** or **logistic** function

$$\sigma(z) = \frac{1}{1+e^{-z}} = \frac{e^z}{1+e^z},$$

  where the z can be a result of a linear transform $\mathbf{w}^T\mathbf{x}$ ($\mathbf{x}$ is augmented to include the bias).
- Properties of the sigmoid function: Range is $(0, 1)$.

Logistic Regression II

- Note that in some context, the word “sigmoid” is used to describe any S-shape functions. And the funtion symbol $\sigma()$ could be used for other functions. ..And, the logarithm can be of any base.
- To use logistic regression for classification, the class labels should be 0 and 1 instead of any arbitrary number, such as $+1$ and -1 we have been doing in class. In cross-entropy loss, only one term will be activated depending on the label.
- The loss function used for using logistic regression for classification is usually **cross-entropy**, also called **log loss**:

$$J = \sum_i \left[-y_i \log(\hat{y}_i) - (1 - y_i) \log(1 - \hat{y}_i) \right],$$

  where $\hat{y}_i = \sigma(\mathbf{w}^T \mathbf{x_i})$ is the prediction and y_i is the target for the i-th sample.
- That is

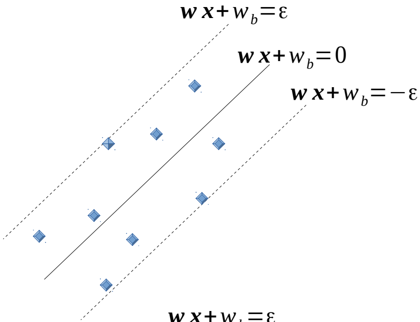
$$\begin{cases} -\log(\sigma(\mathbf{w}^T \mathbf{x_i})) & \text{if } y = 1, \\ -\log(1 - \sigma(\mathbf{w}^T \mathbf{x_i})) & \text{if } y = 0, \end{cases}$$

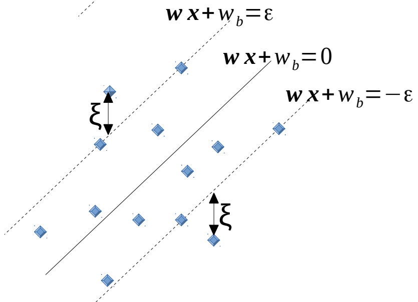
Logistic Regression III

- $$\frac{\partial J}{\partial \mathbf{w}} = \frac{1}{m} \sum_i (\sigma(\mathbf{w}^T \mathbf{x}_i) - y)\mathbf{x}_i$$

SVM-based regression

- In (hard-margin) SVMs, samples need to be out of the margin.
- A inverse problem: Find a strip zone along the hyperplane, such that all samples are in the zone.
- Hence we can use SVM for regression but samples must be inside the "margin". $\begin{cases} \min & \frac{1}{2}||\mathbf{w}||^2 \\ s.t. & |y_i - \mathbf{w}^T\mathbf{x}_i + w_b| \le \epsilon, \forall \mathbf{x}_i. \end{cases}$
- In regression a "hard margin" is often hard to achieve, so add the slack variables: $\begin{cases} \min & \frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i \\ s.t. & |y_i - \mathbf{w}^T\mathbf{x}_i + w_b| \le \epsilon + \xi_i, \forall \mathbf{x}_i \\ & \xi_i \ge 0. \end{cases}$
- ϵ is a predefined value – how accurately you want the regression to be.
- Actually, we don't have margin here. IT's called ϵ-insensitive zone.
- This kind of SVMs are called ϵ-SVMs.





SVM-regression (cond.)

- ϵ-insensitive loss (very similar to hinge loss): $L(\mathbf{w}) = \begin{cases} 0 & \text{if } |y - \mathbf{w}^T\mathbf{x}| \leq \epsilon, \\ |y - \mathbf{w}^T\mathbf{x}| - \epsilon & \text{o/w,} \end{cases}$
- What is the $\min \frac{1}{2}||\mathbf{w}||^2$ for? We don't have margins.
- It functions as an L2 regularizer.
- Good visuals:
  - http://kernelsvm.tripod.com/
  - https://www.saedsayad.com/support_vector_machine_reg.htm

Overfitting vs. Regularization

- **Overfitting**: A common problem in ML is that the model is very accurate on training data but not on test data
- A good example online
- In regression, this can be visualized as that the fitted curve matches training points very well, but misses test points.
- The cause is, for linear models, the magitudes of elements in $\mathbf{w}$ are too big.
- Dr. Chung's slides.
- A further extreme case is when the magnitudes of certain elements are substiantially bigger than others. The model relys on certain features or certain components of the data too much.
- How to avoid overfitting? **regularization**.

L1 and L2 regularization (for linear models)

- L1 regularization (Lasso regularization): $J = \text{Error}(\hat{y}, y) + \alpha||\mathbf{w}||$
- L2 (ridge): $J = \text{Error}(\hat{y}, y) + \alpha||\mathbf{w}||^2$
- α is a constant weighing the regularization term. It's also a hyperparameter.
- Why they work?
- The new gradients:

$$\frac{\partial J_{L1}}{\partial \mathbf{w}} = \frac{\partial \text{Error}}{\partial \mathbf{w}} \pm \alpha$$

or

$$\frac{\partial J_{L2}}{\partial \mathbf{w}} = \frac{\partial \text{Error}}{\partial \mathbf{w}} + 2\alpha\mathbf{w}$$

- When using the new gradients to update $\mathbf{w}$, $\mathbf{w}$ is not updated to what would be ideal.
- A good explanation online