Regression: Linear Function

EE219: Large Scale Data Mining

Professor Roychowdhury

Jan 18, 2017

Summary

- Generate features
- Find linear function
- Evaluation
  - cross validation
- Regularization
  - L1 norm, L2 norm, pick λ

Introduction

To model a specific task in the real world, selecting features is difficult but a very important part.

Question

How to generate meaningful and useful features? In other words, which features should be selected, how to encode the important information we observed as features?

Scenario

In the image processing, without features, the original input dimension is too large. For a RGB image with 1000*800 pixels, its size is 3*1000*800 Bytes = 2.4 MB, we need to reduce it to features such as vectors for further processing.
e.g. swipe a small window through the image, use the color histogram or average brightness inside the window as features.

To track one person in the real world, we can use a vector, $X = \begin{bmatrix} x_1 & x_2 & \dots & x_d \end{bmatrix}^T \in \mathrm{R}^d$ as features to transform information into real numbers.
e.g x_1 maybe this person's age, x_2 maybe gender, x_3 maybe income and so on. We can also encode his location through longitude and latitude.

Encode information

Questions

Is it good way to encode the person's location in Zip code instead of longitude and latitude?

In the Euclidean space, the distance between two points $X_1, X_2 \in \mathrm{R}^d$ can be expressed as $\|X_1 - X_2\|_2$. Zip code won't have this property.

One-hot encoding

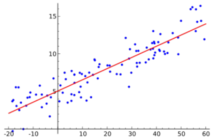
There will be single high bit(1) in the vector and the others are low bits(0).
e.g To encode person's current location among 100 possible cities, X $\in \mathrm{R}^{100}$, if X(i) represents Los Angeles, then a person is in LA can be expressed as X$= [0, ..1, ..0]^T$ with X(i)$= 1$. This vector encodes one city a time and is orthogonal to each other.

Prediction

There are companies such as Acxiom and Epsilon, using customs' data to make better decisions. For example, for a credit card company, it records 2000 dimensions vector for per costumer, (y_i, x_i). x_i is the input features, y_i represents number of times ith person's credit card was compromised in the past years or whether this person made purchases over \$3000.

Question

If given the data $(y_1, x_1), (y_2, x_2), (y_3, x_3), (y_4, x_4)$, can we predict y_i given x_i ?
How to solve it when X has dimension d $= 1$? when $d > 1$?



$$y_i = f(x_i) + \epsilon_i$$

ϵ_i is the ith error

Linear function

Question

What f can be used? How can we find f?

- linear function
  - easy to measure and deal with
  - nonlinear function can be approximated by linear function in certain conditions.
  - it has single minimum, which is the global minimum for MSE
  - $y_i = \sum_{j=1}^{d} a_j * x_i(j) + \epsilon_i = x_i^T \theta + \epsilon_i$, parameter θ is used to represent the parameter set of function f.
- Find best θ
  - Design the cost function:
  - solve mean squared error(MSE)/ least squares problem
  - Gauss developed the formula around 1800s.

Find best θ

Assume we have n observations, $(y_1, x_1), (y_2, x_2), ...(y_n, x_n)$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \text{——} & x_1^T & \text{——} \\ \text{——} & x_2^T & \text{——} \\ \vdots & & \\ \vdots & & \\ \text{——} & x_n^T & \text{——} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_d \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \vdots \\ \epsilon_n \end{bmatrix}$$

- It can be expressed as $y_{d\times 1} = B_{n\times d}\theta_{d\times 1} + \epsilon_{n\times 1}$.
- Define the cost function: $g(\theta) = \sum_{i=1}^{n} \epsilon_i^2$, then the optimal solution is $\hat{\theta} = argmin_{\theta}(\sum_{i=1}^{n} \epsilon_i^2) = argmin_{\theta}(\|y - B\theta\|_2^2)$
- it is an optimization problem and has single global minimum.

Solve MSE problem

$$\begin{aligned} g(\theta) = \|y - B\theta\|_2^2 = & \quad (y^T - \theta^T B^T)(y - B\theta) \\ = & \quad y^T y - 2\theta^T B^T y + \theta^T B^T B\theta \end{aligned}$$

$$\frac{\partial g}{\partial \theta} = \begin{bmatrix} \frac{\partial g}{\partial \theta_1} \\ \frac{\partial g}{\partial \theta_2} \\ \vdots \\ \frac{\partial g}{\partial \theta_d} \end{bmatrix} = 2B^T B\theta - 2B^T y = 0$$

Then the optimal solution is $\theta^* = (B^T B)^{-1} B^T y$, $\epsilon_i = y_i - x_i^T \theta^*$
(if $B^T B$ is invertible)

Covariance matrix

$$B^T B = \begin{bmatrix} | & \vdots & | \\ x_1 & \vdots & x_n \\ | & \vdots & | \end{bmatrix} \begin{bmatrix} \text{——} & x_1^T & \text{——} \\ \text{——} & x_2^T & \text{——} \\ \vdots & & \\ \vdots & & \\ \text{——} & x_n^T & \text{——} \end{bmatrix} = \sum_{i=1}^{n} (x_i x_i^T)$$

When $\mathrm{E}[X] = 0, \mathrm{Cov}(X) = \frac{1}{n} B^T B$, $(B^T B)_{kj} = \sum_{i=1}^{n} (x_i(k) x_i(j))$

Evaluation

Training error : MSE $= \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i^T\theta^*)^2$
$(y_1, x_1)...(y_n, x_n)$ are n training data.

Problem

- even if $B^T B$ is invertible, it may be ill-conditioned?????
- θ^* is a function of the training data set, when it meets with quite different data set, it may fail to predict.
- introduce evaluation to test model's ability to generalize. This is typical overfitting problem, it arises because the model is learnt "too closely" from the training set.(New nonlinear model will be discussed later)

Cross Validation

- Random shuffle the training data,
- break up into k sets
- repeat k times
  - learn the model of k-1 sets
  - compute θ^* based on the $(\frac{k-1}{k})n$ data points
  - calculate mean square prediction error(MSPE) using the left $(\frac{1}{k})n$ data points : $\frac{1}{\frac{n}{k}} \sum_{i \in left\ out\ set} (y_i - x_i^T \theta^*)^2$
- calculate the average MSPE by averaging k MSPE in k operations.

Regularization - L2

$\theta^* = argmin_\theta L(\theta) = argmin_\theta \frac{1}{n}(y^T - \theta^T B^T)(y - B\theta) + \lambda \theta^T \theta,$

$\frac{\partial L(\theta)}{\partial \theta} = 2B^T B\theta - 2B^T y + 2\lambda I\theta = 0,$

the optimal solution $\theta^* = (B^T B + \lambda I_{d\times d})^{-1} B^T y$

- to make the MSE solution more stable
    - previous $\theta^* = (B^T B)^{-1} B^T y$, $B^T B$ might be close to being singular.
    - After L2 regularization, the optimal solution will be more stable.
- add a penalty term
  $\theta^T \theta$ is the magnitude/ L2 norm of vector θ.

Regularization - L1

L1 regularization is also called sparse regression.It is usually applied when dimension d is large.

- y_i dependent on only a subset of features.
- we want to pick the significant ones

$$\theta^* = argmin_\theta L(\theta) = argmin_\theta \frac{1}{n}(y^T - \theta^T B^T)(y - B\theta) + \lambda \sum_{i=1}^{d} |\theta_i|,$$

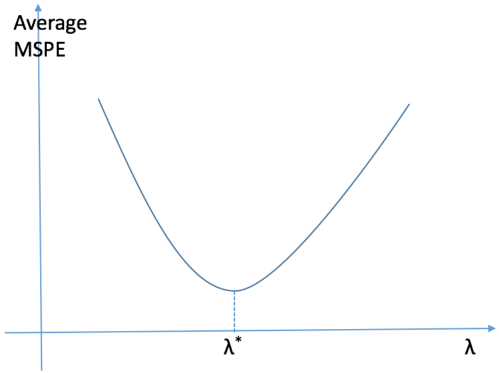
- there is no close form solution, when d is not very large, solution can also be found in explosive way.
- $L(\theta)$ is convex function, so it has global minimum.
- minimization can be done numerically efficiently, such as Newton Method.
- the optimal θ^* will have a lot of '0's.

Convex function

If L(θ) is a convex function of θ, then
$L(a\theta_1 + (1-a)\theta_2) \leq aL(\theta_1) + (1-a)L(\theta_2)$ for $0 \leq a \leq 1$

pick hyper parameter λ

In the case of L1 and L2 regularization, there is a hyper parameter λ, which makes trade off between penalty and training error.
How to pick?


Average MSPE
λ*
λ