4. Regularization and Logistic regression

Shabana K M
PhD Research Scholar
Computer Science and Engineering

IIT Palakkad

28 August 2021

INDIAN INSTITUTE OF TECHNOLOGY PALAKKAD

IEEE STUDENT BRANCH GEC PALAKKAD | IEEE LINK | INTERNZONE | IEEE Kerala Section | IEEE

Recap

- Multiple regression
- Gradient descent
- Determining regression coefficients using:
  - Normal equation
  - Gradient descent

Polynomial regression

- the relationship between the independent variable x and the dependent variable y is modelled as an n^{th} degree polynomial in x

$$\hat{y} = w_0 + w_1 x + w_2 x^2 + \ldots + w_m x^m$$

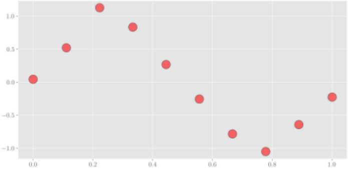
- fits a nonlinear `hypothesis` (model) to the data

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^m \\ 1 & x_2 & x_2^2 & \cdots & x_2^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N^1 & x_N^2 & \cdots & x_N^m \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_m \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix}$$

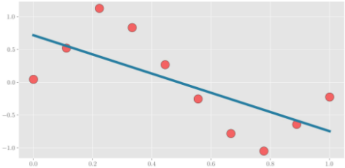
  - estimation problem is linear, as the regression function is linear in the unknown parameters
  - consider x^2, x^3, etc. as independent variables
- considered to be a special case of multiple linear regression
  - weights computed using gradient descent or normal equation

Polynomial regression

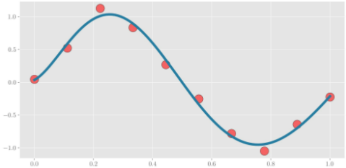
- used when the data distribution is more complex than a simple linear model



(a) Scatter plot of dataset



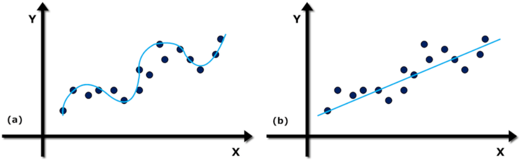
(b) Linear regression on data



(c) Polynomial regression of degree 6

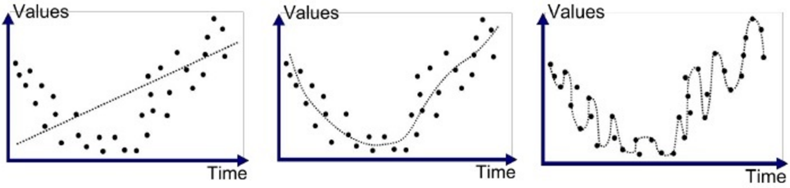
Overfitting

- An `overfitted` model performs very well on the training data but the performance drops significantly over test data
- model learns the detail and noise in the training data as concepts
- these concepts do not apply to new data - negatively impact the model's ability to generalize
- overfitting can be reduced by:
  - increasing training data
  - reducing model complexity



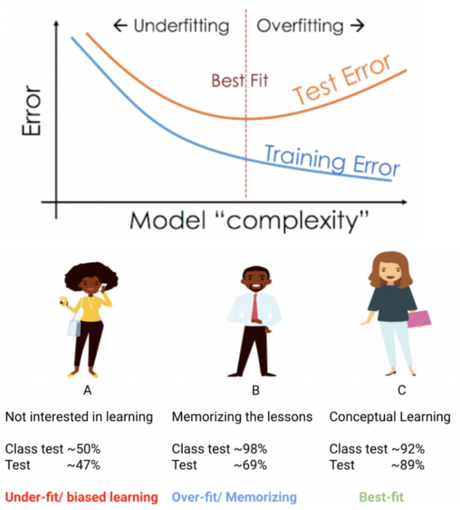
Underfitting

- An `underfitted` model performs poorly over the test and the training dataset
- neither models the training data nor generalize to new data
- underfitting can be reduced by:
  - increase model complexity by adding new features
  - try alternate learning algorithms



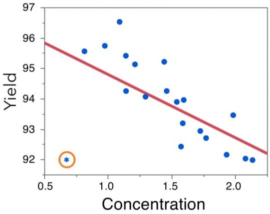
Underfitted   Good Fit/Robust   Overfitted

Overfitting vs. underfitting



A least squares regression model performs poorly when..

- **Multicollinearity:** one (or more) of the independent variable(s) can be expressed as the linear combination of other independent variables - causes overfitting
  - coefficients change erratically in response to small changes in data
- When the number of independent variables is larger than the number of observations - causes overfitting
- Presence of `outliers` - data points that differ significantly from other observations



Regularization

- designed to address the problem of overfitting
- seeks to minimize the sum of the squared error as well as the model complexity
- discourages learning a more complex model by shrinking the coefficients towards zero
- Regularized Loss = Loss Function + Constraint
- different forms of constraints can be added
- popular ones are `Ridge regression`, `LASSO` and `Elastic Net`

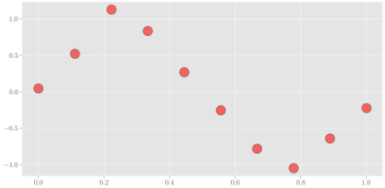
Ridge regression

- also called L2 regularization
- adds a constraint that is a linear function of the squared coefficients
- limits the $L2$ norm of the weights being learned
  - $L2$ norm of a vector v: $||v||_2 = (\sum_{i=1}^{n} v_i^2)^{\frac{1}{2}}$
- the loss function is given by:

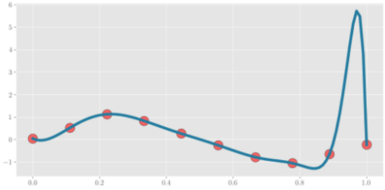
$$l(w) = \sum_{i=1}^{N} (y_i - w^T x_i)^2 + \lambda \sum_{j=1}^{D} w_j^2$$

- λ: **regularization parameter** - controls the trade-off between model complexity and the fit to the data
  - $\lambda > 0$
  - small λ: leads to overfitting
  - large λ: causes underfitting

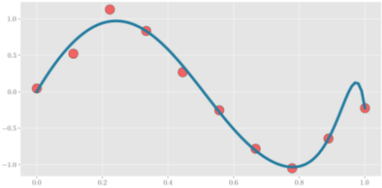
Ridge regression:Example



(a) Scatter plot of data



(b) Polynomial regression of degree 25



(c) Polynomial ridge regression of degree 25

LASSO (Least Absolute Shrinkage and Selection Operator)

- also known as L1 regularization
- penalizes the model by the sum of absolute values of weight coefficients, or the $L1$ norm
  - $L1$ norm of a vector v: $||v||_1 = \sum_{i=1}^{n} |v_i|$
- the loss function is given by:

$$l(w) = \sum_{i=1}^{N} (y_i - w^T x_i)^2 + \lambda \sum_{j=1}^{D} |w_j|$$

- most of the weights will be non-zero in ridge regression
- LASSO tries to find a set of weights such that most of them are almost zero
  - enforces sparsity on the learned weights
  - helps in feature selection

Regression coefficients: Ridge vs LASSO

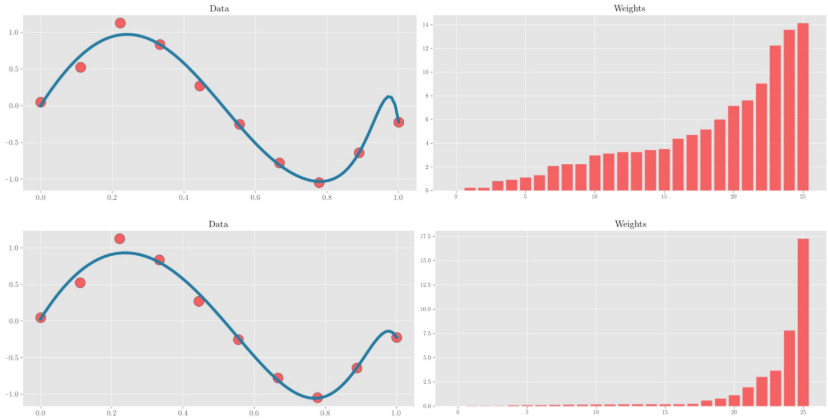


Figure: Ridge regression model (top) LASSO model (bottom)

Elastic Net

- combination of ridge and LASSO regression
- loss term includes both the L1 and L2 norm of the weights with their respective scaling constants

$$l(w) = \sum_{i=1}^{N}(y_i - w^T x_i)^2 + \lambda_1 \sum_{j=1}^{D} |w_j| + \lambda_2 \sum_{j=1}^{D} w_j^2$$

where $\lambda_1, \lambda_2 > 0$

- shrinks the coefficients as well as eliminates some of the insignificant ones

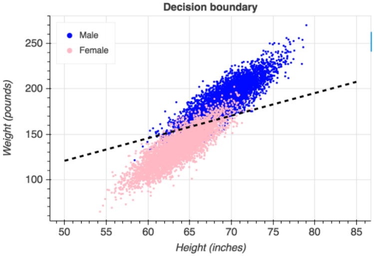
Logistic regression

- regression technique used when dependent variable takes binary values (eg:- yes/no, 0/1, etc.)
- a simple algorithm that performs very well on a wide range of problems
- consider the task of predicting a person's gender (Male/Female) based on their `weight` and `height`

```
// heights_weights.csv

"Height","Weight","Male"
73.847017017515,241.893563180437,1
68.7819040458903,162.3104725213,1
74.1101053917849,212.7408555565,1
71.7309784033377,220.042470303077,1
69.8817958611153,206.349800623871,1
67.2530156878065,152.212155757083,1
68.7850812516616,183.927888604031,1
68.3485155115879,167.971110489509,1
67.018949662883,175.92944039571,1
63.4564939783664,156.399676387112,1
...
63.1794982498071,141.266099582434,0
62.6366749337994,102.85356321483,0
62.0778316936514,138.691680275738,0
60.0304337715611,97.6874322554917,0
59.0982500313486,110.529685683049,0
66.1726521477708,136.777454183235,0
67.067154649054,170.867905890713,0
63.8679922137577,128.475318784122,0
69.0342431307346,163.852461346571,0
61.9442458795172,113.649102675312,0
```

(a) preview of dataset



(b) scatter plot of dataset with decision boundary

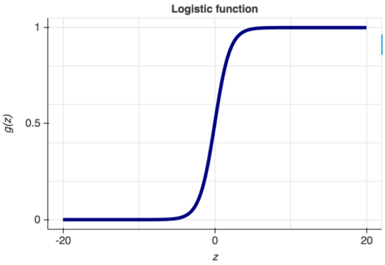
Logistic regression model

- for a binary dependent variable y and independent variables $x_1, x_2, ..., x_D$, a logistic regression model is defined as follows:

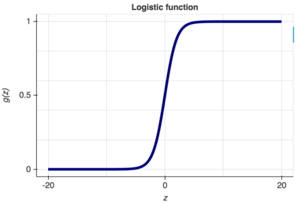
$$y = g(w_0 + w_1 x_1 + ... + w_D x_D)$$

where $g(z) = \frac{1}{1+e^{-z}}$

- $g(.)$ is called the `logistic` or `sigmoid` function
- logistic regression model: $y = h(w, x) = \frac{1}{1+e^{-w^T x}}$



Logistic regression



logistic function

- an S-shaped function
- squashes the value of z ($w^T x$ in our case) into the range $[0, 1]$

- $h(w, x)$ can therefore be interpreted as a probability value
- $P(y = 1|x) = h(w, x) = \frac{1}{1+e^{-w^T x}}$
- $P(y = 0|x) = 1 - h(w, x)$
- model predicts $y = 1$ when $h(w, x) > 0.5$

  $\Rightarrow \quad w^T x > 0$
- the decision boundary is given by $w_0 + \sum_{i=1}^{D} w_i x_i = 0$ linear!!

Cost function

Goal: Find the weight w such that:

- the probability $P(y = 1|x) = h(w, x)$ is large when x belongs to the class 1, and
- small when x belongs to the class 0 i.e., $P(y = 0|x)$ is high

The `cost function` for a weight w is given by:

Cross-entropy loss function

$$l(w) = -\sum_{i=1}^{N}(y_i log(h(w, x_i)) + (1 - y_i) log(1 - h(w, x_i)))$$

Cross entropy loss function

Cross-entropy loss function

$$l(w) = -\sum_{i=1}^{N}(y_i log(h(w,x_i)) + (1-y_i)log(1-h(w,x_i)))$$

One of the two terms in the summation is non-zero for each x_i, depending on the value of y_i

Minimizing the loss function requires:

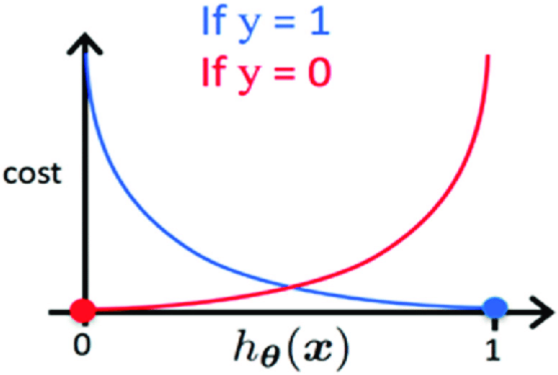
- making $h(w,x_i)$ large when $y_i = 1$
- making $1 - h(w,x_i)$ large or $h(w,x_i)$ small when $y_i = 0$

The cross entropy function is **convex**!

Minimizing the cross entropy loss function is equivalent to **maximizing** the `log-likelihood function` given by:

$$log\ L(w|x,y) = log\prod_{i=1}^{N} h(w,x_i)^{y_i}\ (1-h(w,x_i))^{(1-y_i)}$$

Cost function



Problem definition

Given: Training data set comprising N observations $(x_n, y_n)_{n=1}^{N}$, where $x_n = [x_{n1}, x_{n2}, ..., x_{nD}]$ is the input and $y_n \in \{0, 1\}$ is the corresponding output

Goal: Predict the y value for a new value of x

Estimate: The weights $w = [w_0, w_1, ..., w_D]$ such that:

Minimize: `cross-entropy`:

$$l(w) = -\frac{1}{N}\sum_{i=1}^{N}(y_i log(h(w, x_i)) + (1 - y_i) log(1 - h(w, x_i)))$$

where $h(w, x_i) = \frac{1}{1+e^{-w^T x_i}}$

References

1. https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/
2. https://www.analyticsvidhya.com/blog/2020/02/underfitting-overfitting-best-fitting-machine-learning/
3. https://medium.com/@zxr.nju/the-classical-linear-regression-model-is-good-why-do-we-need-regularizati
4. https://towardsdatascience.com/a-beginners-guide-to-regression-analysis-in-machine-learning-8a828b491bbf
5. https://www.coursera.org/learn/machine-learning/resources/Zi29t
6. https://towardsdatascience.com/understanding-logistic-regression-step-by-step-704a78be7e0a
7. https://www.coursera.org/learn/machine-learning/resources/Zi29t

Thanks Google for the pictures!