CS 383/613 – Machine Learning

Regression

Slides adapted from material created by E. Alpaydin
Prof. Mordohai, Prof. Greenstadt, Pattern Classification (2nd Ed.),
Pattern Recognition and Machine Learning

Objectives

- Linear Regression

Linear Regression

- Recall that in regression, we are attempting to predict a *continuous value.*
  - This is different than classification, which attempt to predict a class enumeration.
- The simplest regression function, and the one we will study, is called *linear regression,* since $g(\boldsymbol{x})$ is a linear combination of the features.

$$g(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_D x_D + b$$

  - We say that w_j is the *weight* of the j^{th} feature and b is the *bias*
  - You can think of the bias as the "offset". Something like the y-intercept of $y = mx + b$

Linear Regression

$$g(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_D x_D + b$$

- Via linear algebra, we can write this as:

$$g(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{w} + b$$

- Since our target value is continuous, we can attempt to find the weights (and bias) to minimize a ***mean squared error*** objective function, defined as:

$$J = \frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \widehat{Y_i}\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(Y_i - g(X_i)\right)^2$$

Objective Function Optimization

- To find the optimal weights for a least squares objective function we can use either a direct or iterative (gradient-based) solutions.
- Let’s look at both!

Optimization Approaches

Direct

- Take the derivate with respect to our weights, set it equal to zero, and solve for those weights.
- Cons: May not be feasible or even possible to solve.

Iterative

- Start with an initial “guess” for the values of the weights.
- Use the derivatives with respect to our weights to update the weights *iteratively*
- Pros: Flexible
- Cons: May take a while to get to a good solution.
- **We’ll get to this later.**

Direct Solution

- Let's start with the direct solution!

$$J = \frac{1}{N}\sum_{j=1}^{N}\left(Y_j - \widehat{Y}_j\right)^2 = \frac{1}{N}\sum_{j=1}^{N}\left(Y_j - \left(X_j\boldsymbol{w} + b\right)\right)^2$$

- First off, finding a direct solution would be easier if we can combine all the weights into one variable/vector.
- A trick to do this involves adding a "dummy" feature, $x_{D+1} = 1$, to our observations.
  - This extra feature is sometimes referred to as the *bias feature.*
- Now the last weight of $\boldsymbol{w}$, w_{D+1} is our bias b!
  - Technically we can put our bias feature anywhere!
- And now we can write our objective function as

$$J = \frac{1}{N}\sum_{j=1}^{N}\left(Y_j - X_j\boldsymbol{w}\right)^2$$

Direct Solution

$$J = \frac{1}{N}\sum_{j=1}^{N}\left(Y_j - X_j\boldsymbol{w}\right)^2$$

- Using linear algebra, we can "eliminate" (really compute) the sum of the squares by writing our objective function as:

$$J = \frac{1}{N}(Y - X\boldsymbol{w})^T(Y - X\boldsymbol{w})$$

- And finally, we just need to take the derivative of J with respect to $\boldsymbol{w}$, set it equal to zero, and solve for $\boldsymbol{w}$!
- Doing this will be easier if we expand on J

Direct Solution

$$J = \frac{1}{N}(Y - X\boldsymbol{w})^T(Y - X\boldsymbol{w})$$

- $J = \frac{1}{N}\left(Y^T - (X\boldsymbol{w})^T\right)(Y - X\boldsymbol{w})$ //distribute transpose
- $J = \frac{1}{N}\left(Y^T - \boldsymbol{w}^T X^T\right)(Y - X\boldsymbol{w})$ //distribute transpose
- $J = \frac{1}{N}\left(Y^T Y - Y^T X\boldsymbol{w} - \boldsymbol{w}^T X^T Y + \boldsymbol{w}^T X^T X\boldsymbol{w}\right)$ //distribution

Direct Solution

$$J = \frac{1}{N}\left(Y^T Y \ - Y^T X\boldsymbol{w} - \boldsymbol{w}^T X^T Y + \boldsymbol{w}^T X^T X\boldsymbol{w}\right)$$

- Taking the derivative of this with respect to $\boldsymbol{w}$…

$$\frac{dJ}{d\boldsymbol{w}} = \frac{1}{N}\left(0 - \left(Y^T X\right)^T - X^T Y + 2X^T X\boldsymbol{w}\right) = \frac{1}{N}\left(2X^T X\boldsymbol{w} - 2X^T Y\right)$$

- Setting this equal to zero and solving for $\boldsymbol{w}$ we get:

$$\boldsymbol{w} = \left(X^T X\right)^{-1} X^T Y$$

Example

- The dataset on the right contains test scores for 25 students
  - 25 rows
  - Three exams and a final exam.
- Our goal is to predict the final exam scores, given a students three exam scores.
- We'll use the first 8 observations for validation, and the remaining for training.

EXAM1	EXAM2	EXAM3	FINAL
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142
53	46	55	101
69	74	77	149
47	56	60	115
87	79	90	175
79	70	88	164
69	70	73	141
70	65	74	141
93	95	91	184
79	80	73	152
70	73	78	148
93	89	96	192
78	75	68	147
81	90	93	183
88	92	86	177
78	83	77	159
82	86	90	177
86	82	89	175
78	83	85	175
76	83	71	149
96	93	95	192

Example

- Learning our model as $\boldsymbol{w} = \left(X^T X\right)^{-1} X^T Y$ we get:

$$\boldsymbol{w} = \begin{bmatrix} 0.38 \\ 0.57 \\ 1.19 \\ -9.96 \end{bmatrix}$$

$$FinalExam = 0.38 * Exam1 + 0.57 * Exam2 + 1.19 * Exam3 - 9.96$$

- Evaluating:
  - Training RMSE: 2.39
  - Validation RMSE: 2.82
  - Training SMAPE: 0.0049
  - Validation SMAPE: 0.0076

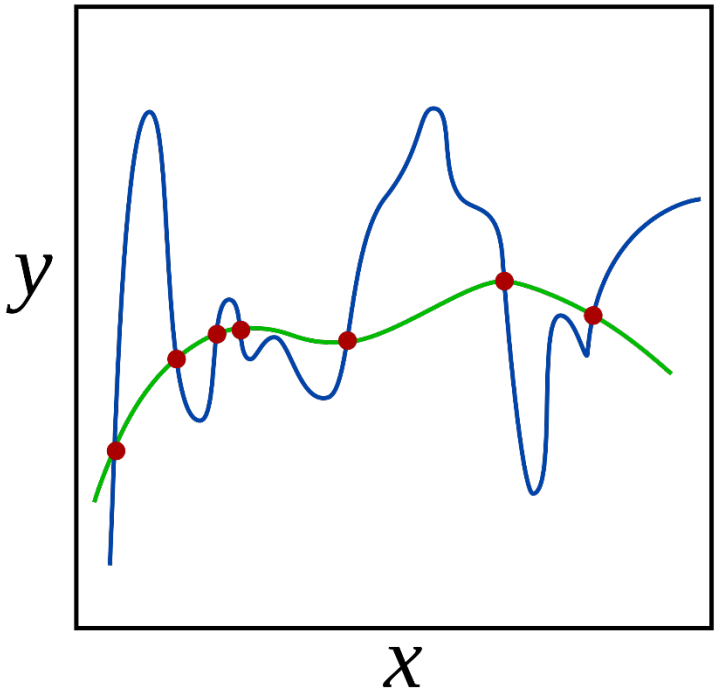
EXAM1	EXAM2	EXAM3	FINAL
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142
53	46	55	101
69	74	77	149
47	56	60	115
87	79	90	175
79	70	88	164
69	70	73	141
70	65	74	141
93	95	91	184
79	80	73	152
70	73	78	148
93	89	96	192
78	75	68	147
81	90	93	183
88	92	86	177
78	83	77	159
82	86	90	177
86	82	89	175
78	83	85	175
76	83	71	149
96	93	95	192

Dealing with Overfitting

- General approaches:
  - Try to get more data for training to generalize better
  - Or get more data for training via cross-validation.
  - Don’t use all the features.
- Algorithm-specific approaches:
  - Objective functions can add regularization terms to penalize complexity.
    - More on this now!

Regularization Term

- Overfit models tend to have a lot of large weights.
- So, one technique is to *regulate* how large these can become.



Regularization Term

- To do this, we augment our objective function to add in a penalization term (call a *regularization* term).
- Two simple ones are:
  - L^2 regularization – The sum (or average) of the squares of the weights.
  - L^1 regularization – The sum (or average) of the absolute values of the weights.
- These terms can then be blended in with the original objective function using some blending hyperparameter.

Regularization Term

- For instance, we could take our squared error objective function and add to it L^2 regularization of the weights $\boldsymbol{w}$
- We can compute the L^2 of the column vector $\boldsymbol{w}$ as:

$$\frac{1}{D+1}\sum_{d=1}^{D+1} w_d^2 = \frac{1}{D+1}\boldsymbol{w}^T\boldsymbol{w}$$

- So, given some blending factor $0 \leq \alpha \leq 1$, our regularized squared error objective function becomes:

$$J = (1-\alpha)(y-\hat{y})^2 + \alpha\frac{1}{D+1}\boldsymbol{w}^T\boldsymbol{w}$$

- Now for this we can compute our closed-form rules.

Beyond Linear Regression

- What if we want to find the weights for a quadratic equation like $y = ax^2 + bx + c$?
- Imagine our observation just has a single feature, ie. $\boldsymbol{x} = [x_1]$.
- We can write our quadratic equation as

$$y = w_1 x_1 + w_2 x_1^2 + b$$

- If we re-write $\boldsymbol{x}$ to be $x = [x_1 \quad x_1^2 \quad 1]$ then we can write this equation as $y = \boldsymbol{xw}$ and we're right back at linear regression!