Bachelor of Science (Honours) in Data Science and Artificial Intelligence

DA261: Machine Learning Fundamentals

Instructor: Teena Sharma, Ph.D.
Mehta Family School of Data Science and Artificial Intelligence,
Indian Institute of Technology Guwahati, India

Learning Objectives

- Linear regression
- Overfitting
- Underfitting
- Regularization
- Differentiate between regression and classification tasks

Introduction to Supervised Learning

Given a set of input patterns X and a corresponding set of labels Y, the underlined mapping function $f: X \rightarrow Y$ is discovered. The classifier operates in two distinct phases, i.e., a training phase (model tuning), and an operating phase, in which the model is kept fixed and tested with different and new data.

Supervised learning is useful when the map from inputs to outputs is unknown, but we have a lot of input-to-output examples.

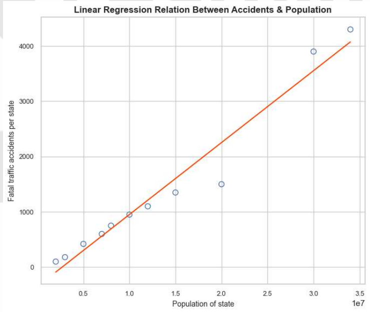
Introduction to Supervised Learning

$$\boxed{y_i = f(x_i)}$$

- When y_i is continuous, this is a regression problem.
- When y_i is discrete, this is a classification problem.

Linear Regression

- Regression analysis is a set of statistical methods used for the estimation of relationships between a dependent variable and one or more independent variables.
- Regression models are used to predict a continuous value.



Linear Regression

- Let's consider the linear regression problem.
- Feature vector: $\boldsymbol{x_i} = [x_{i_1}, x_{i_2}, \dots \dots, x_{i_d}]$
- Target: y_i
- Prediction: $\hat{y_i} = w_1 x_{i_1} + w_2 x_{i_2} + \cdots w_d x_{i_d}$ (weighted sum of features)

  Weight of feature 1 feature 1

Linear Regression

- Let's consider the linear regression problem.
- Feature vector: $\boldsymbol{x_i} = [x_{i_1}, x_{i_2}, \dots \dots, x_{i_d}]$
- Target: y_i
- Prediction: $\hat{y_i} = w_1 x_{i_1} + w_2 x_{i_2} + \cdots w_d x_{i_d}$ (weighted sum of features)

  Weight of feature 1 feature 1

If $w_1 > 0$, more populations means more murder rate

Linear Regression

- Let's consider the linear regression problem.
- Feature vector: $\boldsymbol{x_i} = [x_{i_1}, x_{i_2}, \dots \dots, x_{i_d}]$
- Target: y_i
- Prediction: $\hat{y_i} = w_1 x_{i_1} + w_2 x_{i_2} + \cdots w_d x_{i_d}$ (weighted sum of features)

  Weight of feature 1 feature 1

We want to make $y_i - \hat{y_i}$ small, i.e. we want prediction to be close to the target.

Linear Regression

Let's consider the linear regression problem.

- Residual: $y_i - \hat{y}_i = y_i - \sum_{j=1}^{d} w_j x_{ij}$

We need to tune the weights to **minimize the prediction error.**

Least Square Error

What is the classic way to minimize this error considering we have n training examples?

Minimizing Least Square Error!

We need to find $\boldsymbol{w}$ such that it minimizes: $\sum_{j=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{j=1}^{n} (y_i - \boldsymbol{w}^T x_i)^2$

where, $\boldsymbol{w} = [w_1, w_2, \dots, w_d]$ $\qquad = \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|^2$

Minimizing Least Square Error

$\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|^2$

$\boldsymbol{X} = [x_1, x_2, \dots, x_d]$ $\qquad \boldsymbol{w} = [w_1, w_2, \dots, w_d]^T$ $\qquad \boldsymbol{y} = [y_1, y_2, \dots, y_n]^T$

$n * d$ Feature matrix $\qquad d * 1$ Weight vector $\qquad n * 1$ Target vector

Minimizing Least Square Error

$\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|^2$

This is unknown

$\boldsymbol{X} = \boxed{[x_1, x_2, \dots, x_d]}$ $\qquad \boldsymbol{w} = \boxed{[w_1, w_2, \dots, w_d]^T}$ $\qquad \boldsymbol{y} = [\boxed{y_1}, y_2, \dots, y_n]^T$

$n * d$ Feature matrix $\qquad d * 1$ Weight vector $\qquad n * 1$ Target vector

d

n

	A	B	C	D	E	F	G	H
1	mpg	cylinder	displacement	horsepower	weight	acceleration	modelyear	origin
2	18	8	307	130	3504	12	70	1
3	15	8	350	165	3693	11.5	70	1
4	18	8	318	150	3436	11	70	1
5	16	8	304	150	3433	12	70	1
6	17	8	302	140	3449	10.5	70	1
7	15	8	429	198	4341	10	70	1
8	14	8	454	220	4354	9	70	1
9	14	8	440	215	4312	8.5	70	1
10	14	8	455	225	4425	10	70	1
11	15	8	390	190	3850	8.5	70	1
12	15	8	383	170	3563	10	70	1
13	14	8	340	160	3609	8	70	1
14	15	8	400	150	3761	9.5	70	1
15	14	8	455	225	3086	10	70	1
16	24	4	113	95	2372	15	70	3
17	22	6	198	95	2833	15.5	70	1
18	18	6	199	97	2774	15.5	70	1
19	21	6	200	85	2587	16	70	1
20	27	4	97	88	2130	14.5	70	3
21	26	4	97	46	1835	20.5	70	2
22	25	4	110	87	2672	17.5	70	2
23	24	4	107	90	2430	14.5	70	2
24	25	4	104	95	2375	17.5	70	2
25	26	4	121	113	2234	12.5	70	2
26	21	6	199	90	2648	15	70	1

Minimizing Least Square Error

Finding solution to get $\boldsymbol{w}$: $\quad \text{minimize } \boxed{\|X\boldsymbol{w}-\boldsymbol{y}\|^2} \text{ over } \boldsymbol{w} \in R^d$

Set gradient equal to zero and solve for $\boldsymbol{w}$

Gradient is a vector of partial derivatives:

$$\nabla f(\boldsymbol{w}) = \left[\frac{\partial f}{\partial w_1}, \frac{\partial f}{\partial w_2}, \dots \frac{\partial f}{\partial w_d}\right]^T$$

Finding Gradient

Finding the gradient of linear functions:

Linear function: $\quad f(\boldsymbol{w}) = \boldsymbol{\alpha}^T\boldsymbol{w} + \beta$

Scalar

Vector

Finding gradient:

Step-1: $\quad f(\boldsymbol{w}) = \sum_{\{i=1\}}^{d} \alpha_i w_i + \beta$

Step-2: $\quad \frac{\partial f(\boldsymbol{w})}{\partial w_i} = \alpha_i$

Step-3: $\quad \nabla f(\boldsymbol{w}) = [\alpha_1, \alpha_2, \dots, \alpha_d]^T = \boldsymbol{\alpha}$

Finding Gradient

Finding the gradient of Quadratic functions:

Quadratic function: $\quad f(\boldsymbol{w}) = \boldsymbol{w}^T A \boldsymbol{w}$

Finding gradient:

Step-1: Convert to summation notation:

$$f(\boldsymbol{w}) = \sum_{\{i=1\}}^{d} \sum_{\{j=1\}}^{d} w_i a_{ij} w_j$$

where, a_{ij} is the element in row i and column j of $\boldsymbol{A}$. To help with computing the partial derivatives, it helps to re-write it in the form

$$f(\boldsymbol{w}) = \sum_{\{i=1\}}^{d} a_{ii} w_i^2 + \sum_{\{j \neq i\}} w_i a_{ij} w_j$$

Finding Gradient

Finding the gradient of Quadratic functions:

Quadratic function: $\quad f(\boldsymbol{w}) = \boldsymbol{w}^T A \boldsymbol{w}$

Finding gradient:

Step-2: Take the partial derivative with respect to an element k:

$$\frac{\partial}{\partial w_k}\left[\sum_{\{i=1\}}^{d} a_{ii} w_i^2 + \sum_{\{j \neq i\}} w_i a_{ij} w_j\right] = 2a_{kk} w_k + \sum_{\{j \neq k\}} w_j a_{jk} + \sum_{\{j \neq k\}} a_{kj} w_j$$

where, a_{ij} is the element in row i and column j of $\boldsymbol{A}$. To help with computing the partial derivatives, it helps to re-write it in the form

$$\frac{\partial}{\partial w_k}\left[\sum_{\{i=1\}}^{d} a_{ii} w_i^2 + \sum_{\{j \neq i\}} w_i a_{ij} w_j\right] = \sum_{\{j=1\}}^{d} w_j a_{jk} + \sum_{\{j=1\}}^{d} a_{kj} w_j$$

Finding Gradient

$$\frac{\partial}{\partial w_k}\left[\sum_{\{i=1\}}^{d} a_{ii}w_i^2 + \sum_{\{j\neq i\}} w_i a_{ij} w_j\right] = 2a_{kk}w_k + \sum_{\{j\neq k\}} w_j a_{jk} + \sum_{\{j\neq k\}} a_{kj} w_j$$

Finding Gradient

Finding the gradient of Quadratic functions:

Quadratic function: $f(\boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{A}\boldsymbol{w}$

$$\frac{\partial}{\partial w_k}\left[\sum_{\{i=1\}}^{d} a_{ii}w_i^2 + \sum_{\{j\neq i\}} w_i a_{ij} w_j\right] = \sum_{\{j=1\}}^{d} w_j a_{jk} + \sum_{\{j=1\}}^{d} a_{kj} w_j$$

Finding gradient:

Step-3: Assemble the partial derivatives into a vector:

$$\nabla f(w) = \begin{bmatrix} \frac{\partial}{\partial w_1} \\ \frac{\partial}{\partial w_2} \\ \vdots \\ \frac{\partial}{\partial w_d} \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^d w_j a_{j1} + \sum_{j=1}^d a_{1j} w_j \\ \sum_{j=1}^d w_j a_{j2} + \sum_{j=1}^d a_{2j} w_j \\ \vdots \\ \sum_{j=1}^d w_j a_{jd} + \sum_{j=1}^d a_{dj} w_j \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^d w_j a_{j1} \\ \sum_{j=1}^d w_j a_{j2} \\ \vdots \\ \sum_{j=1}^d w_j a_{jd} \end{bmatrix} + \begin{bmatrix} \sum_{j=1}^d a_{1j} w_j \\ \sum_{j=1}^d a_{2j} w_j \\ \vdots \\ \sum_{j=1}^d a_{dj} w_j \end{bmatrix}$$

Finding Gradient

Finding the gradient of Quadratic functions:

Quadratic function: $f(\boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{A}\boldsymbol{w}$

$$\frac{\partial}{\partial w_k}\left[\sum_{\{i=1\}}^{d} a_{ii}w_i^2 + \sum_{\{j\neq i\}} w_i a_{ij} w_j\right] = \sum_{\{j=1\}}^{d} w_j a_{jk} + \sum_{\{j=1\}}^{d} a_{kj} w_j$$

Finding gradient:

Step-4: Convert to matrix notation:

$$\nabla f(w) = \begin{bmatrix} \sum_{j=1}^d w_j a_{j1} \\ \sum_{j=1}^d w_j a_{j2} \\ \vdots \\ \sum_{j=1}^d w_j a_{jd} \end{bmatrix} + \begin{bmatrix} \sum_{j=1}^d a_{1j} w_j \\ \sum_{j=1}^d a_{2j} w_j \\ \vdots \\ \sum_{j=1}^d a_{dj} w_j \end{bmatrix} = A^T w + Aw = (A^T + A)w$$

Finding Gradient

Finding the gradient of Quadratic functions:

Quadratic function: $f(\boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{A}\boldsymbol{w}$

Finding gradient:

Final result: $\boxed{\boldsymbol{\nabla} f(\boldsymbol{w}) = (\boldsymbol{A}^T + \boldsymbol{A})\boldsymbol{w}}$

Least Square Problem

Our least square problem:

$$\underset{w \in R^d}{\text{Minimize}} \; ||Xw - y||^2 \sim \underset{w \in R^d}{\text{Minimize}} \; \frac{1}{2}||Xw - y||^2$$

$$f(w) = \frac{1}{2}||Xw - y||^2$$

Least Square Problem

Our least square problem:

$$\underset{w \in R^d}{\text{Minimize}} \; ||Xw - y||^2 \sim \underset{w \in R^d}{\text{Minimize}} \; \frac{1}{2}||Xw - y||^2$$

$$f(w) = \frac{1}{2}||Xw - y||^2 = \frac{1}{2}(Xw - y)^T(Xw - y) \qquad [||a||^2 = a^T a]$$

$$= \frac{1}{2}(w^T X^T - y^T)(Xw - y) \qquad [(ab)^T = b^T a^T]$$

$$= \frac{1}{2}(w^T X^T (Xw - y) - y^T(Xw - y))$$

$$= \frac{1}{2}(w^T X^T Xw - w^T X^T y - y^T Xw + y^T y)$$

$w^T X^T y = (y^T Xw)^T = y^T Xw = (X^T y)^T w$
Since both are scalars, so they are equal

$$= \frac{1}{2}(w^T X^T Xw - (X^T y)^T w - (X^T y)^T w + y^T y) \qquad [w^T v = v^T w]$$

$$= \frac{1}{2} w^T X^T Xw - (X^T y)^T w + \frac{1}{2} y^T y$$

Least Square Problem

Objective function: $f(w) = \underbrace{\frac{1}{2} w^T X^T Xw}_{\text{Quadratic}} - \underbrace{(X^T y)^T w}_{\text{Linear}} + \underbrace{\frac{1}{2} y^T y}_{\text{Constant}}$

Gradient computation: $\nabla f(w) = X^T Xw - X^T y$

Setting gradient to zero:

Least Square Problem

Objective function: $f(w) = \underbrace{\frac{1}{2} w^T X^T Xw}_{\text{Quadratic}} - \underbrace{(X^T y)^T w}_{\text{Linear}} + \underbrace{\frac{1}{2} y^T y}_{\text{Constant}}$

Gradient computation: $\nabla f(w) = X^T Xw - X^T y$

Setting gradient to zero: $\nabla f(w) = X^T Xw - X^T y = 0 \rightarrow (X^T X)w = X^T y$

$(if\ (X^T X)\ is\ invertible \rightarrow (X^T X)^{-1}(X^T X)w = (X^T X)^{-1} X^T y$

$\rightarrow w = (X^T X)^{-1} X^T y \qquad [A^{-1}A = I]$

Least Square Solution

Cost of Computing Least Squares:

- Getting $\boldsymbol{X}^T\boldsymbol{y}$ costs $O(nd)$
- Getting $\boldsymbol{X}^T\boldsymbol{X}$ costs $O(nd^2)$
- Solving a $d * d$ linear system costs $O(d^3)$: Using Gaussian Elimination

Total cost = $O(nd^2 + d^3)$

Is Least Squares any good?

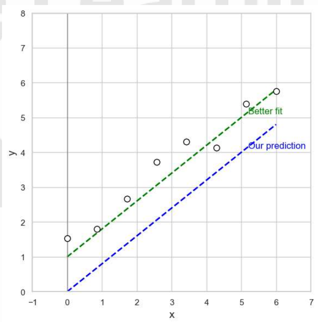
Issues with least squares model:

- It assumes a linear relationship between x and y.
- It might predict poorly for new values of x.
- $\boldsymbol{X}^T\boldsymbol{X}$ might not be invertible.
- It is sensitive to outliers.
- It might predict outside known range of y values.
- It always uses all features.
- Number of dimensions d might be so big we can't store $\boldsymbol{X}^T\boldsymbol{X}$.

Problem with Basic Linear Models?

This is our model: $\hat{y}_i = \sum_{j=1}^{d} w_j x_{ij} = \boldsymbol{w}^T\boldsymbol{x}_i$

- This is always satisfied: $\boldsymbol{x}_i = 0 \rightarrow \hat{y}_i = 0$
- The fitted line always pass through origin. *Undesirable!*



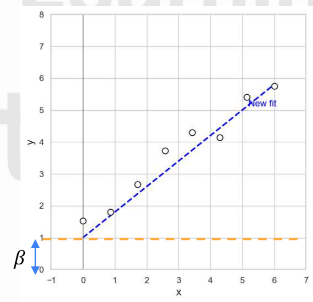
Problem with Basic Linear Models?

This is our model: $\hat{y}_i = \sum_{j=1}^{d} w_j x_{ij} = \boldsymbol{w}^T\boldsymbol{x}_i$

- Now our updated model: $y_i = \boldsymbol{w}^T\boldsymbol{x}_i + \beta$

  Bias

- Now: $\boldsymbol{x}_i = 0 \rightarrow \hat{y}_i = \beta$



How to include the Bias variable?

To include the bias variable in the objective function of the least square problem:

1. Modify the weight Vector

$$\boldsymbol{w} = [w_1, w_2, \ldots, w_d]^T \rightarrow \overline{\boldsymbol{w}} = [w_0, w_1, w_2, \ldots, w_d]^T \quad , where\ w_0 = \beta$$

$d * 1$ $\qquad (d+1) * 1$

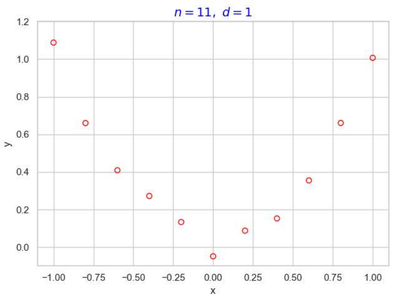
Solution with Bias variable

$$(\overline{X}^T\overline{X})\overline{\boldsymbol{w}} = \overline{X}^T\overline{y} \rightarrow \quad \overline{\boldsymbol{w}} = (\overline{X}^T\overline{X})^{-1}\overline{X}^T\overline{y}$$

Let's take simple example:

$\boldsymbol{X} \rightarrow 11 * 1$
$\boldsymbol{w} \rightarrow 1 * 1$
$\boldsymbol{y} \rightarrow 11 * 1$

x	y
-1.0000	1.0882
-0.8000	0.6600
-0.6000	0.4089
-0.4000	0.2720
-0.2000	0.1334
0.0000	-0.0489
0.2000	0.0875
0.4000	0.1524
0.6000	0.3548
0.8000	0.6605
1.0000	1.0072



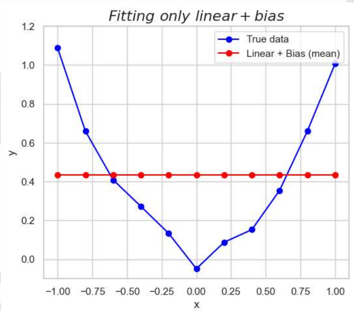
Solution with Bias variable

Linear Regression with bias: $\quad \hat{y}_i = \beta + w_1x_i$

We can clearly see that our data is not linear!

Now let's assume a quadratic basis:

$$\hat{y}_i = \beta + w_1x_i + w_2x_i^2$$



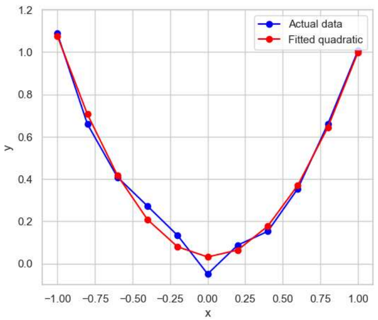
Solution with Quadratic variable

Model with Quadratic bias: $\hat{y}_i = \beta + w_1x_i + w_2x_i^2$

$$\boldsymbol{w} = [\beta, w_1, w_2]^T$$

$$\boldsymbol{y} = [y_1, y_2, \ldots, y_n]^T$$

Final solution: $\boldsymbol{w}^* = \left(\boldsymbol{X}_{poly}^T\,\boldsymbol{X}_{poly}\right)^{-1}\boldsymbol{X}_{ploy}^T\boldsymbol{y}$



Solution with Quadratic variable

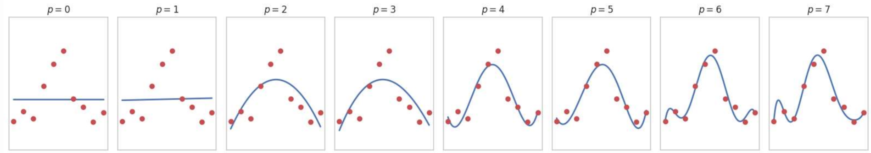
We can also have polynomial of degree p: $\hat{y}_i = \beta + w_1 x_i + w_2 x_i^2 + \cdots + w_p x_i^p$

$$\boldsymbol{w} = [\beta, w_1, w_2, \dots, w_p]^T$$

$$\boldsymbol{y} = [y_1, y_2, \dots, y_n]^T$$

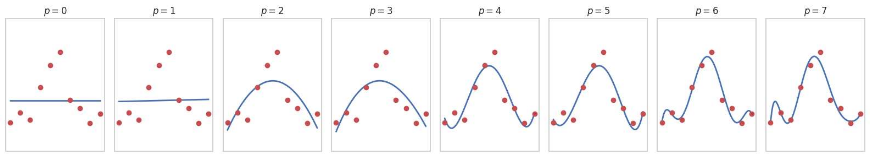
True or False?

Higher p means better model (Think and try to answer)?



True or False?

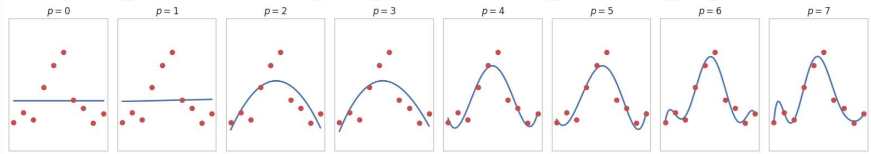
Higher p means better model (Think and try to answer)?



Low degree
- Less likely to fit data well
- Model does not change much with change in data

True or False?

Higher p means better model (Think and try to answer)?

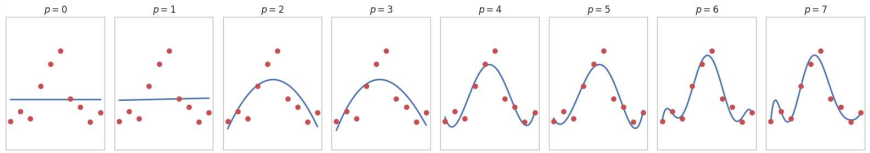


Low degree
- Less likely to fit data well
- Model does not change much with change in data

High degree
- Very likely to fit data well
- Model change a lot with change in data

True or False?

Higher p means better model (Think and try to answer)?



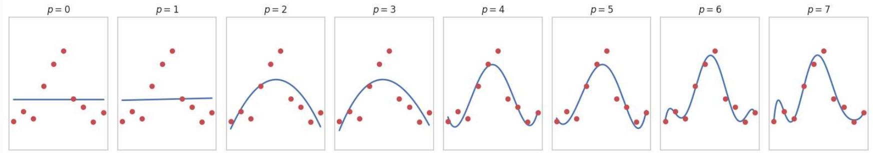
Low degree
- Less likely to fit data well
- Model does not change much with change in data

High degree
- Very likely to fit data well
- Model change a lot with change in data

High Bias, Low Variance
- High expected error due to wrong model

True or False?

Higher p means better model (Think and try to answer)?



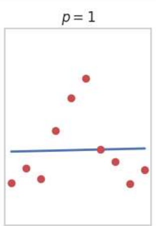
Low degree
- Less likely to fit data well
- Model does not change much with change in data

High degree
- Very likely to fit data well
- Model change a lot with change in data

High Bias, Low Variance
- High expected error due to wrong model

High Variance, Low Bias
- High sensitive the model is to particular training set

Underfitting

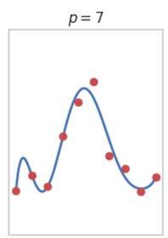


- Model does not fit the data well.
- The model fit is not very sensitive to the training data

High Bias, Low Variance

- High expected error due to wrong model

Overfitting



- Fit all the data, training error is very low
- The model fit is sensitive to the training data.
- Does not generalize well for new test data (not in training example).

High Variance, Low Bias
- High sensitive the model is to particular training set

Determining Model Overfitting or Underfitting

We can determine model underfitting or overfitting by observing the error during model training and testing.

For that we need to understand following things:

- What is model training?
- What is model testing?
- Can we use same dataset for both training and testing?

Training and Testing

First we have to divide our dataset into two parts:

- Training Data
- Testing Data

You can select the dataset in 70% train and 30% test data.

- Train model using training data
- Test model using testing data

Good training performance is useless, if it does not perform well with test data.

Example:

Train data

Inhabitants	BelowIncome	PercentageUnemployed	Murders
587000.0	16.5	6.2	11.2
643000.0	20.5	5.4	19.4
635000.0	20.3	5.3	40.7
692000.0	19.2	5.4	22.3
690000.0	19.6	5.2	12.8
684000.0	16.2	5.6	13.4
700000.0	21.3	5.7	26.9
713000.0	17.3	6.5	12.6
714000.0	14.3	6.4	9.6
758000.0	13.4	6.6	7.9
789000.0	13.3	6.5	9.8
718000.0	13.1	5.6	10.7
721000.0	12.7	5.8	12.7
727000.0	12.0	5.7	12.3
690000.0	16.4	6.3	8.3
854000.0	24.9	8.3	28.9

Test data

Inhabitants	BelowIncome	PercentageUnemployed	Murders
716000.0	17.9	6.7	14.8
952000.0	22.4	8.6	25.3
695000.0	20.2	8.4	22.7
3350000.0	16.9	6.7	25.7

Training and Testing

There are two phases of supervised learning:

- **Training Phase:** Model fitting based on the training data (X_{train}, y_{train}).
- **Testing Phase:** Model evaluation on test data (X_{test}, y_{test}), that was not used for training the model.

We also need to use evaluation metric to find the **testing error** e.g. $\|y_{pred} - y_{test}\|^2$

By observing only testing error we can say how **good** our model is.

Test data should never be used for training the model.

Managing Model Complexity

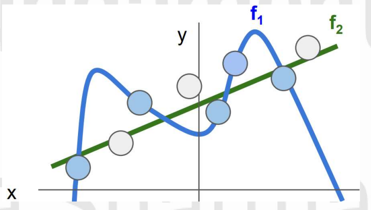
- In practical scenarios, the relationship between features and the target variable can be **quite complex**.
- However, **highly complex models tend to overfit the training data**.

So, how can we systematically control model complexity to achieve better generalization?

Regularization

One most popular technique is **Regularization**

- We can add the **penalty** on the complexity of the model by using **regularization.**



Regularization pushes against fitting the data too well so we don't fit noise in the data

Source: https://cs231n.stanford.edu/slides/2024/lecture_3.pdf

Regularization

The optimization problem for regression: $\underset{w \in R^d}{\text{Minimize}} \ \frac{1}{2}\|Xw - y\|^2$

Instead we need to solve this!

Standard L2-regularization strategy is to add a penalty on the L2-norm:

$$\underset{w \in R^d}{\text{Minimize}} \ \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$$

L2-norm: $\|X\|_2 = \left(\sum x_i^2\right)^{1/2} = \sqrt{\{x_1^2 + x_2^2 + \dots + x_N^2\}}$

Source: https://cs231n.stanford.edu/slides/2024/lecture_3.pdf

Regularization

The optimization problem for regression: $\underset{w \in R^d}{\text{Minimize}} \ \frac{1}{2}\|Xw - y\|^2$

Instead we need to solve this!

Standard L2-regularization strategy is to add a penalty on the L2-norm:

$$\underset{w \in R^d}{\text{Minimize}} \ \frac{1}{2}\|Xw - y\|^2 + \boxed{\frac{\lambda}{2}\|w\|^2}$$

Regularization term

L2-norm: $\|X\|_2 = \left(\sum x_i^2\right)^{1/2} = \sqrt{\{x_1^2 + x_2^2 + \dots + x_N^2\}}$

Source: https://cs231n.stanford.edu/slides/2024/lecture_3.pdf

Try this to find the solution

Objective function: $f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$

Find gradient: ?

Set gradient to zero: ?

Solution:?

Source: https://cs231n.stanford.edu/slides/2024/lecture_3.pdf

Try this to find the solution

Objective function: $f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$

Find gradient: ?

Set gradient to zero: ?

Solution: $\boxed{w^* = (X^T X + \lambda I)^{-1} X^T y}$

Source: https://cs231n.stanford.edu/slides/2024/lecture_3.pdf

DA261 Machine Learning Fundamentals Course Instructor: Teena Sharma, Ph.D.

Thank you!

DA261 Machine Learning Fundamentals Teena Sharma, Ph.D.