Machine Learning - MT 2016
2. Linear Regression

Varun Kanade

University of Oxford
October 12, 2016

Announcements

- All students eligible to take the course for credit can sign-up for classes and practicals
- Attempt Problem Sheet 0 (contact your class tutor if you intend to attend class in Week 2)
- Problem Sheet 1 is posted (submit by noon 21 Oct at CS reception)

Announcement : Strachey Lecture

- Will finish 15-20 min early on Monday, October 31
- May run over by 5 minutes or so a few other days

Outline

Goals

- Review the supervised learning setting
- Describe the linear regression framework
- Apply the linear model to make predictions
- Derive the least squares estimate

Supervised Learning Setting

- Data consists of input and output pairs
- Inputs (also covariates, independent variables, predictors, features)
- Output (also variates, dependent variable, targets, labels)

Why study linear regression?

- Least squares is at least 200 years old going back to Legendre and Gauss
- Francis Galton (1886): ‘‘Regression to the mean’’

- Often real processes can be approximated by linear models
- More complex models require understanding linear regression

- Closed form analytic solutions can be obtained
- Many key notions of machine learning can be introduced

A toy example : Commute Times

Want to predict commute time into city centre

What variables would be useful?

- Distance to city centre
- Day of the week

Data

dist (km)	day	commute time (min)
2.7	fri	25
4.1	mon	33
1.0	sun	15
5.2	tue	45
2.8	sat	22

Linear Models

Suppose the input is a vector $\mathbf{x} \in \mathbb{R}^D$ and the output is $y \in \mathbb{R}$.

We have data $\langle \mathbf{x}_i, y_i \rangle_{i=1}^N$

<u>Notation:</u> data dimension D, size of dataset N, column vectors

Linear Model

$$y = \underbrace{w_0}_{\text{Bias/intercept}} + x_1 w_1 + \cdots + x_D w_D + \underbrace{\epsilon}_{\text{Noise/uncertainty}}$$

Linear Models : Commute Time

Linear Model

$$y = \underbrace{w_0}_{\text{Bias/intercept}} + x_1 w_1 + \cdots + x_D w_D + \underbrace{\epsilon}_{\text{Noise/uncertainty}}$$

Input encoding: mon-sun has to be converted to a number

- monday: 0, tuesday: 1, . . . , sunday: 6
- 0 if weekend, 1 if weekday

Using 0-6 is a bad encoding. Use seven 0-1 features instead called one-hot encoding

Say $x_1 \in \mathbb{R}$ (distance) and $x_2 \in \{0, 1\}$ (weekend/weekday)

Linear model for commute time

$$y = w_0 + w_1 x_1 + w_2 x_2 + \epsilon$$

Linear Model : Adding a feature for bias term

dist x_1	day x_2	commute time y
2.7	fri	25
4.1	mon	33
1.0	sun	15
5.2	tue	45
2.8	sat	22

$\Leftrightarrow$

one x_0	dist x_1	day x_2	commute time y
1	2.7	fri	25
1	4.1	mon	33
1	1.0	sun	15
1	5.2	tue	45
1	2.8	sat	22

Model

$$y = w_0 + w_1 x_1 + w_2 x_2 + \epsilon$$

Model

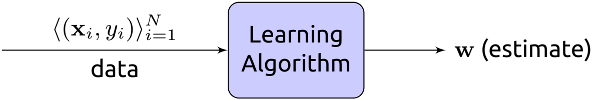
$$\begin{aligned} y &= w_0 x_0 + w_1 x_1 + w_2 x_2 + \epsilon \\ &= \mathbf{w} \cdot \mathbf{x} + \epsilon \end{aligned}$$

Learning Linear Models

Data: $\langle(\mathbf{x}_i, y_i)\rangle_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^D$ and $y_i \in \mathbb{R}$

Model parameter $\mathbf{w}$, where $\mathbf{w} \in \mathbb{R}^D$

Training phase: (learning/estimation $\mathbf{w}$ from data)



Testing/Deployment phase: (predict $\widehat{y}_{\text{new}} = \mathbf{x}_{\text{new}} \cdot \mathbf{w}$)

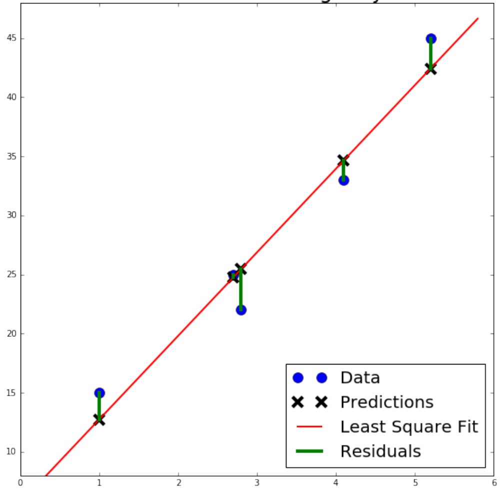
- How different is $\widehat{y}_{\text{new}}$ from y_{new} (actual observation)?
- We should keep some data aside for testing before deploying a model

$\langle (x_i, y_i) \rangle_{i=1}^N$, where $x_i \in \mathbb{R}$ and $y_i \in \mathbb{R}$

$\widehat{y}(x) = w_0 + x \cdot w_1$, (no noise term in $\widehat{y}$)

$$\mathcal{L}(\mathbf{w}) = \mathcal{L}(w_0, w_1) = \frac{1}{2N} \sum_{i=1}^{N} (\widehat{y_i} - y_i)^2 = \frac{1}{2N} \sum_{i=1}^{N} (w_0 + x_i \cdot w_1 - y_i)^2$$

Predict commute time using only distance



Loss function
Cost function
Objective Function
Energy Function
Notation - $\mathcal{L}, J, E, R$

This objective is known as the residual sum of squares or (RSS)

The estimate (w_0, w_1) is known as the least squares estimate

$\langle (x_i, y_i) \rangle_{i=1}^N$, where $x_i \in \mathbb{R}$ and $y_i \in \mathbb{R}$

$\widehat{y}(x) = w_0 + x \cdot w_1$, (no noise term in $\widehat{y}$)

$$\mathcal{L}(\mathbf{w}) = \mathcal{L}(w_0, w_1) = \frac{1}{2N} \sum_{i=1}^{N} (\widehat{y_i} - y_i)^2 = \frac{1}{2N} \sum_{i=1}^{N} (w_0 + x_i \cdot w_1 - y_i)^2$$

$$\frac{\partial \mathcal{L}}{\partial w_0} = \frac{1}{N} \sum_{i=1}^{N} (w_0 + w_1 \cdot x_i - y_i)$$

$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{1}{N} \sum_{i=1}^{N} (w_0 + w_1 \cdot x_i - y_i) x_i$$

We obtain the solution for (w_0, w_1) by setting the partial derivatives to 0 and solving the resulting system. (Normal Equations)

$$w_0 + w_1 \cdot \frac{\sum_i x_i}{N} = \frac{\sum_i y_i}{N} \tag{1}$$

$$w_0 \cdot \frac{\sum_i x_i}{N} + w_1 \cdot \frac{\sum_i x_i^2}{N} = \frac{\sum_i x_i y_i}{N} \tag{2}$$

$$\bar{x} = \frac{\sum_i x_i}{N}$$

$$\bar{y} = \frac{\sum_i y_i}{N}$$

$$\widehat{\mathrm{var}}(x) = \frac{\sum_i x_i^2}{N} - \bar{x}^2$$

$$\widehat{\mathrm{cov}}(x, y) = \frac{\sum_i x_i y_i}{N} - \bar{x} \cdot \bar{y}$$

$$w_1 = \frac{\widehat{\mathrm{cov}}(x, y)}{\widehat{\mathrm{var}}(x)}$$

$$w_0 = \bar{y} - w_1 \cdot \bar{x}$$

Linear Regression : General Case

Recall that the linear model is

$$\widehat{y}_i = \sum_{j=0}^{D} x_{ij} w_j$$

where we assume that $x_{i0} = 1$ for all $\mathbf{x}_i$, so that the bias term w_0 does not need to be treated separately.

Expressing everything in matrix notation

$$\widehat{\mathbf{y}} = \mathbf{X}\mathbf{w}$$

Here we have $\widehat{\mathbf{y}} \in \mathbb{R}^{N \times 1}$, $\mathbf{X} \in \mathbb{R}^{N \times (D+1)}$ and $\mathbf{w} \in \mathbb{R}^{(D+1) \times 1}$

$$\underbrace{\begin{bmatrix} \widehat{y}_1 \\ \widehat{y}_2 \\ \vdots \\ \widehat{y}_N \end{bmatrix}}^{\widehat{\mathbf{y}}_{N \times 1}} = \overbrace{\begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_N^\top \end{bmatrix}}^{\mathbf{X}_{N \times (D+1)}} \overbrace{\begin{bmatrix} w_0 \\ \vdots \\ w_D \end{bmatrix}}^{\mathbf{w}_{(D+1) \times 1}} = \overbrace{\begin{bmatrix} x_{10} & \cdots & x_{1D} \\ x_{20} & \cdots & x_{2D} \\ \vdots & \ddots & \vdots \\ x_{N0} & \cdots & x_{ND} \end{bmatrix}}^{\mathbf{X}_{N \times (D+1)}} \overbrace{\begin{bmatrix} w_0 \\ \vdots \\ w_D \end{bmatrix}}^{\mathbf{w}_{(D+1) \times 1}}$$

Back to toy example

one	dist (km)	weekday?	commute time (min)
1	2.7	1 (fri)	25
1	4.1	1 (mon)	33
1	1.0	0 (sun)	15
1	5.2	1 (tue)	45
1	2.8	0 (sat)	22

We have $N = 5$, $D + 1 = 3$ and so we get

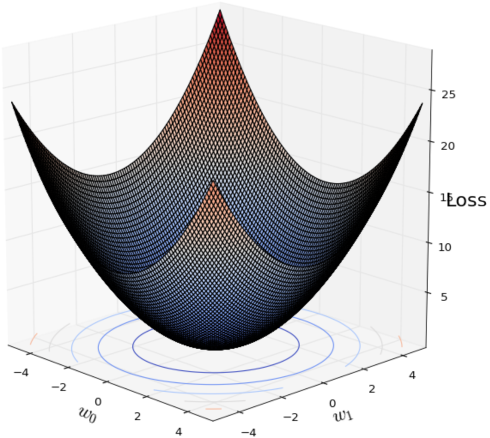
$$\mathbf{y} = \begin{bmatrix} 25 \\ 33 \\ 15 \\ 45 \\ 22 \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & 2.7 & 1 \\ 1 & 4.1 & 1 \\ 1 & 1.0 & 0 \\ 1 & 5.2 & 1 \\ 1 & 2.8 & 0 \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

Suppose we get $\mathbf{w} = [6.09, 6.53, 2.11]^\mathsf{T}$. Then our predictions would be

$$\widehat{\mathbf{y}} = \begin{bmatrix} 25.83 \\ 34.97 \\ 12.62 \\ 42.16 \\ 24.37 \end{bmatrix}$$

Least Squares Estimate : Minimise the Squared Error

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2N}\sum_{i=1}^{N}(\mathbf{x}_i^\mathsf{T}\mathbf{w} - y_i)^2 = (\mathbf{X}\mathbf{w} - \mathbf{y})^\mathsf{T}(\mathbf{X}\mathbf{w} - \mathbf{y})$$



Finding Optimal Solutions using Calculus

$$\begin{aligned}
\mathcal{L}(\mathbf{w}) &= \frac{1}{2N}\sum_{i=1}^{N}(\mathbf{x}_i^\top \mathbf{w} - y_i)^2 = \frac{1}{2N}\left(\mathbf{X}\mathbf{w}-\mathbf{y}\right)^\top\left(\mathbf{X}\mathbf{w}-\mathbf{y}\right) \\
&= \frac{1}{2N}\left(\mathbf{w}^\top\left(\mathbf{X}^\top\mathbf{X}\right)\mathbf{w} - \mathbf{w}^\top\mathbf{X}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{X}\mathbf{w} + \mathbf{y}^\top\mathbf{y}\right) \\
&= \frac{1}{2N}\left(\mathbf{w}^\top\left(\mathbf{X}^\top\mathbf{X}\right)\mathbf{w} - 2\cdot\mathbf{y}^\top\mathbf{X}\mathbf{w} + \mathbf{y}^\top\mathbf{y}\right) \\
&= \cdots
\end{aligned}$$

Then, write out all partial derivatives to form the gradient $\nabla_{\mathbf{w}}\mathcal{L}$

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial w_0} &= \cdots \\
\frac{\partial\mathcal{L}}{\partial w_1} &= \cdots \\
&\vdots \\
\frac{\partial\mathcal{L}}{\partial w_D} &= \cdots
\end{aligned}$$

Instead, we will develop tricks to differentiate using matrix notation directly

Differentiating Matrix Expressions

Rules (Tricks)

(i) Linear Form Expressions: $\nabla_{\mathbf{w}} \left(\mathbf{c}^{\mathsf{T}} \mathbf{w} \right) = \mathbf{c}$

$$\mathbf{c}^{\mathsf{T}} \mathbf{w} = \sum_{j=0}^{D} c_j w_j$$

$$\frac{\partial \left(\mathbf{c}^{\mathsf{T}} \mathbf{w} \right)}{\partial w_j} = c_j, \qquad \text{and so} \quad \nabla_{\mathbf{w}} \left(\mathbf{c}^{\mathsf{T}} \mathbf{w} \right) = \mathbf{c} \tag{3}$$

(ii) Quadratic Form Expressions:

$\nabla_{\mathbf{w}} \left(\mathbf{w}^{\mathsf{T}} \mathbf{A} \mathbf{w} \right) = \mathbf{A}\mathbf{w} + \mathbf{A}^{\mathsf{T}} \mathbf{w}$ ($= 2\mathbf{A}\mathbf{w}$ for symmetric $\mathbf{A}$)

$$\mathbf{w}^{\mathsf{T}} \mathbf{A} \mathbf{w} = \sum_{i=0}^{D} \sum_{j=0}^{D} w_i w_j A_{ij}$$

$$\frac{\partial \left(\mathbf{w}^{\mathsf{T}} \mathbf{A} \mathbf{w} \right)}{\partial w_k} = \sum_{i=0}^{D} w_i A_{ik} + \sum_{j=0}^{D} A_{kj} w_j = \mathbf{A}_{[:,k]}^{\mathsf{T}} \mathbf{w} + \mathbf{A}_{[k,:]} \mathbf{w}$$

$$\nabla_{\mathbf{w}} \left(\mathbf{w}^{\mathsf{T}} \mathbf{A} \mathbf{w} \right) = \mathbf{A}^{\mathsf{T}} \mathbf{w} + \mathbf{A} \mathbf{w} \tag{4}$$

Deriving the Least Squares Estimate

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2N}\sum_{i=1}^{N}(\mathbf{x}_i^\top \mathbf{w} - y_i)^2 = \frac{1}{2N}\left(\mathbf{w}^\top\left(\mathbf{X}^\top\mathbf{X}\right)\mathbf{w} - 2\cdot\mathbf{y}^\top\mathbf{X}\mathbf{w} + \mathbf{y}^\top\mathbf{y}\right)$$

We compute the gradient $\nabla_{\mathbf{w}}\mathcal{L} = \mathbf{0}$ using the matrix differentiation rules,

$$\nabla_{\mathbf{w}}\mathcal{L} = \frac{1}{N}\left(\left(\mathbf{X}^\top\mathbf{X}\right)\mathbf{w} - \mathbf{X}^\top\mathbf{y}\right)$$

By setting $\nabla_{\mathbf{w}}\mathcal{L} = \mathbf{0}$ and solving we get,

$$\left(\mathbf{X}^\top\mathbf{X}\right)\mathbf{w} = \mathbf{X}^\top\mathbf{y}$$

$$\mathbf{w} = \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{y} \qquad \text{(Assuming inverse exists)}$$

The predictions made by the model on the data $\mathbf{X}$ are given by

$$\widehat{\mathbf{y}} = \mathbf{X}\mathbf{w} = \mathbf{X}\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{y}$$

For this reason the matrix $\mathbf{X}\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top$ is called the "hat" matrix

Least Squares Estimate

$$\mathbf{w} = \left(\mathbf{X}^{\mathsf{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{y}$$

- When do we expect $\mathbf{X}^{\mathsf{T}}\mathbf{X}$ to be invertible?

  $\mathrm{rank}(\mathbf{X}^{\mathsf{T}}\mathbf{X}) = \mathrm{rank}(\mathbf{X}) \leq \min\{D+1, N\}$

  As $\mathbf{X}^{\mathsf{T}}\mathbf{X}$ is $D+1 \times D+1$, invertible is $\mathrm{rank}(\mathbf{X}) = D+1$

- What if we use one-hot encoding for a feature like day?

  Suppose $x_{\mathrm{mon}}, \ldots, x_{\mathrm{sun}}$ stand for 0-1 valued variables in the one-hot encoding

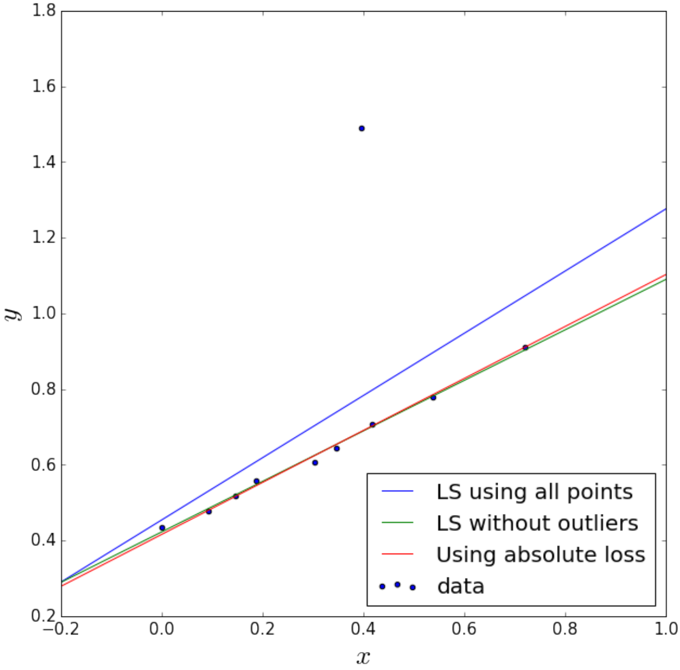
  We always have $x_{\mathrm{mon}} + \cdots + x_{\mathrm{sun}} = 1$

  This introduces a linear dependence in the columns of $\mathbf{X}$ reducing the rank

  In this case, we can drop some features to adjust rank. We'll see alternative approaches later in the course.

- What is the computational complexity of computing $\mathbf{w}$?

  Relatively easy to get $O(D^2 N)$ bound


1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
y
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
x
LS using all points
LS without outliers
Using absolute loss
data

Recap : Predicting Commute Time

Goal

- Predict the time taken for commute given distance and day of week
- Do we only wish to make predictions or also suggestions?

Model and Choice of Loss Function

- Use a linear model

$$y = w_0 + w_1 x_1 + \cdots + w_D x_D + \epsilon = \widehat{y} + \epsilon$$

- Minimise average squared error $\frac{1}{2N} \sum (y_i - \widehat{y_i})^2$

Algorithm to Fit Model

- Simple matrix operations using closed-form solution

Model and Loss Function Choice

"Optimisation" View of Machine Learning

- Pick model that you expect may fit the data well enough
- Pick a measure of performance that makes "sense" and can be optimised
- Run optimisation algorithm to obtain model parameters

Probabilistic View of Machine Learning

- Pick a model for data and explicitly formulate the deviation (or uncertainty) from the model using the language of probability
- Use notions from probability to define suitability of various models
- "Find" the parameters or make predictions on unseen data using these suitability criteria (Frequentist vs Bayesian viewpoints)

Next Time

- Probabilistic View of Machine Learning (Maximum Likelihood)
- Non-linearity using basis expansion
- What to do when you have more features than data?

- Make sure you're familiar with the the multi-variate Gaussian distribution