COMP 551 – Applied Machine Learning
Lecture 2: Linear Regression

Instructor: Herke van Hoof (herke.vanhoof@mail.mcgill.ca)

Slides mostly by: Joelle Pineau

Class web page: www.cs.mcgill.ca/~hvanho2/comp551

Unless otherwise noted, all material posted for this course are copyright of the Instructors, and cannot be reused or reposted without the instructor’s written permission.
Supervised learning

- Given a set of **training examples**: $x_i = <x_{i1}, x_{i2}, x_{i3}, \ldots, x_{in}, y_i>$
  
  $x_{ij}$ is the $j^{th}$ feature of the $i^{th}$ example
  
  $y_i$ is the desired **output** (or **target**) for the $i^{th}$ example.
  
  $X_j$ denotes the $j^{th}$ feature.

- We want to learn a function $f: X_1 \times X_2 \times \ldots \times X_n \rightarrow Y$
  
  which maps the input variables onto the output domain.

<table>
<thead>
<tr>
<th>tumor size</th>
<th>texture</th>
<th>perimeter</th>
<th>outcome</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.02</td>
<td>27.6</td>
<td>117.5</td>
<td>N</td>
<td>31</td>
</tr>
<tr>
<td>17.99</td>
<td>10.38</td>
<td>122.8</td>
<td>N</td>
<td>61</td>
</tr>
<tr>
<td>20.29</td>
<td>14.34</td>
<td>135.1</td>
<td>R</td>
<td>27</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
</tbody>
</table>
Supervised learning

- Given a dataset $X \times Y$, find a function: $f : X \rightarrow Y$ such that $f(x)$ is a good predictor for the value of $y$.

- Formally, $f$ is called the hypothesis.

- Output $Y$ can have many types:
  - If $Y = \mathbb{R}$, this problem is called regression.
  - If $Y$ is a finite discrete set, the problem is called classification.
  - If $Y$ has 2 elements, the problem is called binary classification.
Prediction problems

- The problem of predicting tumour recurrence is called: **classification**
- The problem of predicting the time of recurrence is called: **regression**
- Treat them as two separate supervised learning problems.

<table>
<thead>
<tr>
<th>tumor size</th>
<th>texture</th>
<th>perimeter</th>
<th>outcome</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.02</td>
<td>27.6</td>
<td>117.5</td>
<td>N</td>
<td>31</td>
</tr>
<tr>
<td>17.99</td>
<td>10.38</td>
<td>122.8</td>
<td>N</td>
<td>61</td>
</tr>
<tr>
<td>20.29</td>
<td>14.34</td>
<td>135.1</td>
<td>R</td>
<td>27</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Variable types

- **Quantitative**, often real number measurements.
  - Assumes that similar measurements are similar in nature.

- **Qualitative**, from a set (categorical, discrete).
  - E.g. \{Spam, Not-spam\}

- **Ordinal**, also from a discrete set, without metric relation, but that allows ranking.
  - E.g. \{first, second, third\}
The i.i.d. assumption

• In supervised learning, the examples $x_i$ in the training set are assumed to be independently and identically distributed.
The i.i.d. assumption

• In supervised learning, the examples $x_i$ in the training set are assumed to be independently and identically distributed.

  – Independently: Every $x_i$ is freshly sampled according to some probability distribution $D$ over the data domain $X$.

  – Identically: The distribution $D$ is the same for all examples.

• Why?
Empirical risk minimization

For a given function class $F$ and training sample $S$,

- Define a notion of error (left intentionally vague for now):
  \[ L_S(f) = \# \text{ mistakes made by function } f \text{ on the sample } S \]
Empirical risk minimization

For a given function class $F$ and training sample $S$,

- Define a notion of error (*left intentionally vague for now*): $L_S(f) = \#$ mistakes made by function $f$ on the sample $S$

- Define the Empirical Risk Minimization (ERM):
  $$\text{ERM}_F(S) = \text{argmin}_{f \in F} L_S(f)$$

  where $\text{argmin}$ returns the function $f$ (or set of functions) that achieves the minimum loss on the training sample.

- Easier to minimize the error with i.i.d. assumption.
A regression problem

- What hypothesis class should we pick?

<table>
<thead>
<tr>
<th>Observe</th>
<th>Predict</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>0.86</td>
<td>2.49</td>
</tr>
<tr>
<td>0.09</td>
<td>0.83</td>
</tr>
<tr>
<td>-0.85</td>
<td>-0.25</td>
</tr>
<tr>
<td>0.87</td>
<td>3.10</td>
</tr>
<tr>
<td>-0.44</td>
<td>0.87</td>
</tr>
<tr>
<td>-0.43</td>
<td>0.02</td>
</tr>
<tr>
<td>-1.1</td>
<td>-0.12</td>
</tr>
<tr>
<td>0.40</td>
<td>1.81</td>
</tr>
<tr>
<td>-0.96</td>
<td>-0.83</td>
</tr>
<tr>
<td>0.17</td>
<td>0.43</td>
</tr>
</tbody>
</table>
Linear hypothesis

• Suppose $Y$ is a **linear function** of $X$:

$$f_W(X) = w_0 + w_1 x_1 + \ldots + w_m x_m$$

$$= w_0 + \sum_{j=1}^{m} w_j x_j$$

• The $w_j$ are called **parameters** or **weights**.

• To simplify notation, we add an attribute $x_0=1$ to the $m$ other attributes (also called **bias term** or **intercept**).

**How should we pick the weights?**
Least-squares solution method

• The linear regression problem: 
  \[ f_w(X) = w_0 + \sum_{j=1}^{m} w_j x_j \]
  where \( m \) = the dimension of observation space, i.e. number of features.

• **Goal:** Find the **best** linear model given the data.

• Many different possible **evaluation** criteria!

• Most common choice is to find the \( w \) that minimizes:

  \[ Err(w) = \sum_{i=1}^{n} (y_i - w^T x_i)^2 \]

  (A note on notation: Here \( w \) and \( x \) are column vectors of size \( m+1 \).)
Least-squares solution for $X \in \mathbb{R}^2$

FIGURE 3.1. Linear least squares fitting with $X \in \mathbb{R}^2$. We seek the linear function of $X$ that minimizes the sum of squared residuals from $Y$.

Note that (3.2) makes no assumptions about the validity of model (3.1); it simply finds the best linear fit to the data. Least squares fitting is intuitively satisfying no matter how the data arise; the criterion measures the average lack of fit.

How do we minimize (3.2)? Denote by $X$ the $N \times (p+1)$ matrix with each row an input vector (with a 1 in the first position), and similarly let $y$ be the $N$-vector of outputs in the training set. Then we can write the residual sum-of-squares as

$$RSS(\beta) = (y - X\beta)^T (y - X\beta).$$

(3.3)

This is a quadratic function in the $p+1$ parameters. Differentiating with respect to $\beta$ we obtain

$$\frac{\partial RSS}{\partial \beta} = -2X^T (y - X\beta),$$

$$\frac{\partial^2 RSS}{\partial \beta \partial \beta^T} = 2X^T X.$$  

(3.4)

Assuming (for the moment) that $X$ has full column rank, and hence $X^T X$ is positive definite, we set the first derivative to zero

$$X^T (y - X\beta) = 0 \quad (3.5)$$

to obtain the unique solution

$$\hat{\beta} = (X^T X)^{-1} X^T y.$$  

(3.6)
Least-squares solution method

- Re-write in matrix notation: 
  \[ f_w(X) = Xw \]
  \[ Err(w) = (Y - Xw)^T(Y - Xw) \]

  where 
  \( X \) is the \( n \times m \) matrix of input data, 
  \( Y \) is the \( n \times 1 \) vector of output data, 
  \( w \) is the \( m \times 1 \) vector of weights.

- To minimize, take the derivative w.r.t. \( w \): 
  \[ \frac{\partial Err(w)}{\partial w} = -2X^T(Y - Xw) \]
  - You get a system of \( m \) equations with \( m \) unknowns.

- Set these equations to 0: 
  \[ X^T(Y - Xw) = 0 \]
  - Remember that derivative has to be 0 at a minimum of \( Err(w) \)
Least-squares solution method

- We want to solve for $\mathbf{w}$: $\mathbf{X}^T (\mathbf{Y} - \mathbf{X}\mathbf{w}) = 0$

- Try a little algebra: $\mathbf{X}^T \mathbf{Y} = \mathbf{X}^T \mathbf{X} \mathbf{w}$
  
  $\hat{\mathbf{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$

  ($\hat{\mathbf{w}}$ denotes the estimated weights)

- Train set predictions: $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{w}} = \mathbf{X} (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$

- Predict new data $\mathbf{X}' \rightarrow \mathbf{Y}'$: $\mathbf{Y}' = \mathbf{X}'\hat{\mathbf{w}} = \mathbf{X}' (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$
Example of linear regression

What is a plausible estimate of $w$?  

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.86</td>
<td>2.49</td>
</tr>
<tr>
<td>0.09</td>
<td>0.83</td>
</tr>
<tr>
<td>-0.85</td>
<td>-0.25</td>
</tr>
<tr>
<td>0.87</td>
<td>3.10</td>
</tr>
<tr>
<td>-0.44</td>
<td>0.87</td>
</tr>
<tr>
<td>-0.43</td>
<td>0.02</td>
</tr>
<tr>
<td>-1.10</td>
<td>-0.12</td>
</tr>
<tr>
<td>0.40</td>
<td>1.81</td>
</tr>
<tr>
<td>-0.96</td>
<td>-0.83</td>
</tr>
<tr>
<td>0.17</td>
<td>0.43</td>
</tr>
</tbody>
</table>

Try it!
Data matrices

\[ X^T X = \begin{bmatrix}
0.86 & 0.09 & -0.85 & 0.87 & -0.44 & -0.43 & -1.10 & 0.40 & -0.96 & 0.17 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix} \times \begin{bmatrix}
0.86 & 1 \\
0.09 & 1 \\
-0.85 & 1 \\
0.87 & 1 \\
-0.44 & 1 \\
-0.43 & 1 \\
-1.10 & 1 \\
0.40 & 1 \\
-0.96 & 1 \\
0.17 & 1
\end{bmatrix} = \begin{bmatrix}
4.95 & -1.39 \\
-1.39 & 10
\end{bmatrix} \]
Data matrices

\[ X^T Y = \]

\[
\begin{bmatrix}
0.86 & 0.09 & -0.85 & 0.87 & -0.44 & -0.43 & -1.10 & 0.40 & -0.96 & 0.17 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}
\times
\begin{bmatrix}
2.49 \\
0.83 \\
-0.25 \\
3.10 \\
0.87 \\
0.02 \\
-0.12 \\
1.81 \\
-0.83 \\
0.43
\end{bmatrix}
\]

\[ = \begin{bmatrix}
6.49 \\
8.34
\end{bmatrix} \]
Solving the problem

\[ w = (X^T X)^{-1} X^T Y = \begin{bmatrix} 4.95 & -1.39 \\ -1.39 & 10 \end{bmatrix}^{-1} \begin{bmatrix} 6.49 \\ 8.34 \end{bmatrix} = \begin{bmatrix} 1.60 \\ 1.05 \end{bmatrix} \]

So the best fit line is \( y = 1.60x + 1.05 \).
Solving the problem

\[
\mathbf{w} = (X^T X)^{-1} X^T \mathbf{Y} = \begin{bmatrix} 4.95 & -1.39 \\ -1.39 & 10 \end{bmatrix}^{-1} \begin{bmatrix} 6.49 \\ 8.34 \end{bmatrix} = \begin{bmatrix} 1.60 \\ 1.05 \end{bmatrix}
\]

So the best fit line is \( y = 1.60x + 1.05 \).
Interpreting the solution

- Linear fit for a prostate cancer dataset
  - Features $X = \{lcavol, lweight, age, lbph, svi, lcp, gleason, pgg45\}$
  - Output $y =$ level of PSA (an enzyme which is elevated with cancer).
  - High coefficient weight (in absolute value) = important for prediction.

<table>
<thead>
<tr>
<th>Term</th>
<th>Coefficient</th>
<th>Std. Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>$w_0 = 2.46$</td>
<td>0.09</td>
</tr>
<tr>
<td>lcavol</td>
<td>0.68</td>
<td>0.13</td>
</tr>
<tr>
<td>lweight</td>
<td>0.26</td>
<td>0.10</td>
</tr>
<tr>
<td>age</td>
<td>$-0.14$</td>
<td>0.10</td>
</tr>
<tr>
<td>lbph</td>
<td>0.21</td>
<td>0.10</td>
</tr>
<tr>
<td>svi</td>
<td>0.31</td>
<td>0.12</td>
</tr>
<tr>
<td>lcp</td>
<td>$-0.29$</td>
<td>0.15</td>
</tr>
<tr>
<td>gleason</td>
<td>$-0.02$</td>
<td>0.15</td>
</tr>
<tr>
<td>pgg45</td>
<td>0.27</td>
<td>0.15</td>
</tr>
</tbody>
</table>
Interpreting the solution

- Caveat: data should be in same range
- If we change unit for age from years to months, we expect the optimal weight to be 12x as low (so predictions don’t change)
- Doesn’t mean age became 12x less relevant!
- Can normalize data to make range similar
  - E.g. subtract average and divide by standard deviation
- More principled approach in next lecture
Example

Suppose we observe measurements at 11 equally spaced positions $x = -5, -4, \ldots, 4, 5$. The output for all measurements is $y=0$, except at $x=0$ where we observe $y=1$.

1. Using least-squares regression, what are the weights of the best line to fit this data?
2. What is the magnitude of the remaining least-squares error?
Example

Suppose we observe measurements at 11 equally spaced positions \( x = -5, -4, \ldots, 4, 5 \). The output for all measurements is \( y=0 \), except at \( x=0 \) where we observe \( y=1 \).

1. Using least-squares regression, what are the weights of the best line to fit this data?
   - Same outcomes for positive and negative \( x \), so slope is 0
   - Loss lowest if intercept is mean of outputs \( (1/11) \)

2. What is the magnitude of the remaining least-squares error?
   - \( (1/11)^2 \times 10 \) datapoints with \( y=0 + (10/11)^2 \) at \( x=0 \)
Computational cost of linear regression

• What operations are necessary?
Computational cost of linear regression

• What operations are necessary?
  – Overall: 1 matrix inversion + 3 matrix multiplications
  – $X^TX$ (other matrix multiplications require fewer operations.)
    • $X^T$ is $m \times n$ and $X$ is $n \times m$, so we need $nm^2$ operations.
  – $(X^TX)^{-1}$
    • $X^TX$ is $m \times m$, so we need $m^3$ operations.
Computational cost of linear regression

• What operations are necessary?
  – Overall: 1 matrix inversion + 3 matrix multiplications
  – $X^T X$ (other matrix multiplications require fewer operations.)
    • $X^T$ is $m x n$ and $X$ is $n x m$, so we need $n m^2$ operations.
  – $(X^T X)^{-1}$
    • $X^T X$ is $m x m$, so we need $m^3$ operations.

• We can do linear regression in polynomial time, but handling large datasets (many examples, many features) can be problematic.
An alternative for minimizing mean-squared error (MSE)

- Recall the least-square solution: \( \hat{w} = (X^T X)^{-1} X^T Y \)
- What if \( X \) is too big to compute this explicitly (e.g. \( m \sim 10^6 \))?
An alternative for minimizing mean-squared error (MSE)

- Recall the least-square solution: \( \hat{w} = (X^TX)^{-1}X^T Y \)
- What if \( X \) is too big to compute this explicitly (e.g. \( m \sim 10^6 \))? 

- Go back to the gradient step: 
  \[
  \text{Err}(w) = (Y - Xw)^T(Y - Xw)
  \]
  \[
  \frac{\partial \text{Err}(w)}{\partial w} = -2X^T(Y - Xw)
  \]
  \[
  \frac{\partial \text{Err}(w)}{\partial w} = 2(X^TXw - X^TY)
  \]
Gradient-descent solution for MSE

- Consider the error function:
  
  - The gradient of the error is a vector indicating the direction to the minimum point.
  - Instead of directly finding that minimum (using the closed-form equation), we can take small steps towards the minimum.
Gradient-descent solution for MSE

- We want to produce a sequence of weight solutions, $w_0, w_1, w_2 \ldots$, such that: $\text{Err}(w_0) > \text{Err}(w_1) > \text{Err}(w_2) > \ldots$
Gradient-descent solution for MSE

• We want to produce a sequence of weight solutions, \( w_0, w_1, w_2, \ldots \), such that: \( \text{Err}(w_0) > \text{Err}(w_1) > \text{Err}(w_2) > \ldots \)

• The algorithm:

\[
\begin{align*}
\text{Given an initial weight vector } w_0, \\
\text{Do for } k=1, 2, \ldots \\
w_{k+1} &= w_k - \alpha_k \frac{\partial \text{Err}(w_k)}{\partial w_k} \\
\text{End when } |w_{k+1} - w_k| < \varepsilon
\end{align*}
\]

• Parameter \( \alpha_k > 0 \) is the step-size (or learning rate) for iteration \( k \).
Convergence

• Convergence depends in part on the $\alpha_k$.

• If steps are too large: the $w_k$ may oscillate forever.
  – This suggests that $\alpha_k \to 0$ as $k \to \infty$.

• If steps are too small: the $w_k$ may not move far enough to reach a local minimum.
Robbins-Monroe conditions

• The $\alpha_k$ are a Robbins-Monroe sequence if:

$$\sum_{k=0}^{\infty} \alpha_k = \infty$$

$$\sum_{k=0}^{\infty} \alpha_k^2 < \infty$$

• These conditions are sufficient to ensure convergence of the $w_k$ to a local minimum of the error function.
Robbins-Monroe conditions

- The $\alpha_k$ are a Robbins-Monroe sequence if:
  \[ \sum_{k=0}^{\infty} \alpha_k = \infty \]
  \[ \sum_{k=0}^{\infty} \alpha_k^2 < \infty \]

- These conditions are sufficient to ensure convergence of the $w_k$ to a \textbf{local minimum} of the error function.

  E.g. $\alpha_k = 1 / (k + 1)$ (averaging)
  
  E.g. $\alpha_k = 1/2$ for $k = 1, \ldots, T$
  
  $\alpha_k = 1/2^2$ for $k = T+1, \ldots, (T+1)+2T$
  
  etc.
Local minima

- Convergence is **NOT** to a global minimum, only to local minimum.

- The blue line represents the error function. There is *no guarantee* regarding the amount of error of the weight vector found by gradient descent, compared to the globally optimal solution.
Local minima

- Convergence is **NOT** to a global minimum, only to local minimum.

- For linear function approximations using Least-Mean Squares (LMS) error, this is not an issue: **only ONE** global minimum!
  - Local minima affects many other function approximators.
Local minima

- Convergence is **NOT** to a global minimum, only to local minimum.

- For linear function approximations using Least-Mean Squares (LMS) error, this is not an issue: **only ONE** global minimum!
  - Local minima affects many other function approximators.

- **Repeated random restarts** can help (in all cases of gradient search).
Example (cont’d)

Suppose we observe measurements at 11 equally spaced positions $x = -5, -4, \ldots, 4, 5$. The output for all measurements is $y=0$, except at $x=0$ where we observe $y=1$.

1. Using least-squares regression, what are the weights of the best line to fit this data?

2. What is the magnitude of the remaining least-squares error?

3. Perform 1 step of gradient descent on the weights found in (1) using step size $\alpha=0.05$. What are the new weights?
Example (cont’d)

Suppose we observe measurements at 11 equally spaced positions $x = -5, -4, \ldots, 4, 5$. The output for all measurements is $y=0$, except at $x=0$ where we observe $y=1$.

1. Using least-squares regression, what are the weights of the best line to fit this data?

2. What is the magnitude of the remaining least-squares error?

3. Perform 1 step of gradient descent on the weights found in (1) using step size $\alpha = 0.05$. What are the new weights?
   - We are at optimum already. Weights stay the same $(1/11, 0)$
Basic least-squares solution method

• Recall the least-square solution: \( \hat{w} = (X^T X)^{-1} X^T Y \)

• Assuming for now that \( X \) is reasonably small so computation and memory are not a problem. Can we always evaluate this?
Basic least-squares solution method

• Recall the least-square solution: \( \mathbf{\hat{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y} \)

• Assuming for now that \( \mathbf{X} \) is reasonably small so computation and memory are not a problem. **Can we always evaluate this?**

• To have a unique solution, we need \( \mathbf{X}^T \mathbf{X} \) to be **nonsingular**. That means \( \mathbf{X} \) must have full column rank (i.e. no features can be expressed using other features.)

Exercise: What if \( \mathbf{X} \) does not have full column rank? When would this happen? Design an example. Try to solve it.
Dealing with difficult cases of $(X^T X)^{-1}$

- **Case #1:** The weights are not uniquely defined.
  
  **Solution:** Re-code or drop some redundant columns of $X$.

- **Case #2:** The number of features/weights ($m$) exceeds the number of training examples ($n$).
  
  **Solution:** Reduce the number of features using various techniques (to be studied later.)
Predicting recurrence time from tumor size

This function looks complicated, and a linear hypothesis does not seem very good.

What should we do?
Predicting recurrence time from tumor size

This function looks complicated, and a linear hypothesis does not seem very good.

What should we do?

- *Pick a better function?*
- *Use more features?*
- *Get more data?*
Input variables for linear regression

- Original quantitative variables $X_1$, $\ldots$, $X_m$
- Transformations of variables, e.g. $X_{m+1} = \log(X_i)$
- Basis expansions, e.g. $X_{m+1} = X_i^2$, $X_{m+2} = X_i^3$, $\ldots$
- Interaction terms, e.g. $X_{m+1} = X_i X_j$
- Numeric coding of qualitative variables, e.g. $X_{m+1} = 1$ if $X_i$ is true and 0 otherwise.

In all cases, we can add $X_{m+1}$, $\ldots$, $X_{m+k}$ to the list of original variables and perform the linear regression.
Example of linear regression with polynomial terms

\[ f_w(x) = w_0 + w_1 x + w_2 x^2 \]

\[
\begin{bmatrix}
X^2 & x \\
0.75 & 0.86 & 1 \\
0.01 & 0.09 & 1 \\
0.73 & -0.85 & 1 \\
0.76 & 0.87 & 1 \\
0.19 & -0.44 & 1 \\
0.18 & -0.43 & 1 \\
1.22 & -1.10 & 1 \\
0.16 & 0.40 & 1 \\
0.93 & -0.96 & 1 \\
0.03 & 0.17 & 1
\end{bmatrix}
\begin{bmatrix}
Y \\
2.49 \\
0.83 \\
-0.25 \\
3.10 \\
0.87 \\
0.02 \\
-0.12 \\
1.81 \\
-0.83 \\
0.43
\end{bmatrix}
\]
Solving the problem

\[ \mathbf{w} = (X^T X)^{-1} X^T Y = \begin{bmatrix} 4.11 & -1.64 & 4.95 \\ -1.64 & 4.95 & -1.39 \\ 4.95 & -1.39 & 10 \end{bmatrix}^{-1} \begin{bmatrix} 3.60 \\ 6.49 \\ 8.34 \end{bmatrix} = \begin{bmatrix} 0.68 \\ 1.74 \\ 0.73 \end{bmatrix} \]

So the best order-2 polynomial is \( y = 0.68x^2 + 1.74x + 0.73 \).

Compared to \( y = 1.6x + 1.05 \) for the order-1 polynomial.
Input variables for linear regression

How to choose input variables?

• Propose different strategies, then perform model selection using cross validation (more details later)

• Add many transformation to the set of features, then perform feature selection or dimension reduction (more details later)

• Use problem specific insights:
  – Say, predict displacement of falling option as function of time
  – From physics, know that $s=gt^2$
  – In that case, use squared transformation of $t$ (input variable is $t^2$)
What you should know

• Definition and characteristics of a supervised learning problem.
• Linear regression (hypothesis class, cost function).
• Closed-form least-squares solution method (algorithm, computational complexity, stability issues).
• Gradient descent method (algorithm, properties).
To-do

• Reproduce the linear regression example (slides 17-21), solving it using the software of your choice.

• Suggested complementary readings (this lecture and next lecture):
  – Ch.2 (Sec. 2.1-2.4, 2.9) of Hastie et al.
  – Ch.3 of Bishop.
  – Ch.9 of Shalev-Schwartz et al.

• Write down midterm date in agenda: April 4th, 5:30pm.


• Office hours (confirmed): www.cs.mcgill.ca/~hvanho2/comp551/syllabus.html
Weight space view

FIGURE 3.1. Linear least squares fitting with $X \in \mathbb{R}^2$. We seek the linear function of $X$ that minimizes the sum of squared residuals from $Y$. Note that (3.2) makes no assumptions about the validity of model (3.1); it simply finds the best linear fit to the data. Least squares fitting is intuitively satisfying no matter how the data arise; the criterion measures the average lack of fit.

How do we minimize (3.2)? Denote by $X$ the $N \times (p+1)$ matrix with each row an input vector (with a 1 in the first position), and similarly let $y$ be the $N$-vector of outputs in the training set. Then we can write the residual sum-of-squares as

$$\text{RSS}(\beta) = (y - X\beta)^T(y - X\beta).$$

(3.3)

This is a quadratic function in the $p+1$ parameters. Differentiating with respect to $\beta$ we obtain

$$\frac{\partial \text{RSS}}{\partial \beta} = -2X^T(y - X\beta)$$

$$\frac{\partial^2 \text{RSS}}{\partial \beta^T} = 2X^TX.$$

(3.4)

Assuming (for the moment) that $X$ has full column rank, and hence $X^TX$ is positive definite, we set the first derivative to zero

$$X^T(y - X\beta) = 0$$

(3.5)

to obtain the unique solution

$$\hat{\beta} = (X^TX)^{-1}X^Ty.$$

(3.6)
Instance space view (Geometric view)

\[
X \approx Y
\]

\[
\begin{bmatrix}
1 & 1 \\
1 & 2 \\
1 & 3 \\
\end{bmatrix}
\begin{bmatrix}
w_1 \\
w_2 \\
\end{bmatrix}
\approx
\begin{bmatrix}
1.7 \\
1.7 \\
2.7 \\
\end{bmatrix}
\]