Topics

- The learning problem
  - hypothesis class, estimation algorithm
  - loss and estimation criterion
  - sampling, empirical and expected losses

- Regression, example

- Linear regression
  - estimation, errors, analysis
Review: the learning problem

- Recall the image (face) recognition problem

- **Hypothesis class**: we consider some *restricted* set $\mathcal{F}$ of mappings $f : \mathcal{X} \rightarrow \mathcal{L}$ from images to labels

- **Estimation**: on the basis of a training set of examples and labels, $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, we find an estimate $\hat{f} \in \mathcal{F}$

- **Evaluation**: we measure how well $\hat{f}$ *generalizes* to yet unseen examples, i.e., whether $\hat{f}(x_{new})$ agrees with $y_{new}$
Hypotheses and estimation

- We used a simple linear classifier, a parameterized mapping $f(x; \theta)$ from images $\mathcal{X}$ to labels $\mathcal{L}$, to solve a binary image classification problem (2’s vs 3’s):

$$\hat{y} = f(x; \theta) = \text{sign}(\theta \cdot x)$$

where $x$ is a pixel image and $\hat{y} \in \{-1, 1\}$. 
Hypotheses and estimation

- We used a simple linear classifier, a parameterized mapping $f(x; \theta)$ from images $\mathcal{X}$ to labels $\mathcal{L}$, to solve a binary image classification problem (2’s vs 3’s):

  $$\hat{y} = f(x; \theta) = \text{sign}(\theta \cdot x)$$

  where $x$ is a pixel image and $\hat{y} \in \{-1, 1\}$.

- The parameters $\theta$ were adjusted on the basis of the training examples and labels according to a simple mistake driven update rule (written here in a vector form)

  $$\theta \leftarrow \theta + y_i x_i, \text{ whenever } y_i \neq \text{sign}(\theta \cdot x_i)$$
Hypotheses and estimation

- We used a simple linear classifier, a parameterized mapping $f(x; \theta)$ from images $\mathcal{X}$ to labels $\mathcal{L}$, to solve a binary image classification problem (2’s vs 3’s):

  $$\hat{y} = f(x; \theta) = \text{sign}(\theta \cdot x)$$

  where $x$ is a pixel image and $\hat{y} \in \{-1, 1\}$.

- The parameters $\theta$ were adjusted on the basis of the training examples and labels according to a simple mistake driven update rule (written here in a vector form)

  $$\theta \leftarrow \theta + y_i x_i, \quad \text{whenever} \quad y_i \neq \text{sign}(\theta \cdot x_i)$$

- The update rule attempts to minimize the number of errors that the classifier makes on the training examples.
Estimation criterion

- We can formulate the estimation problem more explicitly by defining a zero-one loss:

\[
\text{Loss}(y, \hat{y}) = \begin{cases} 
0, & y = \hat{y} \\
1, & y \neq \hat{y}
\end{cases}
\]

so that

\[
\frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, \hat{y}_i) = \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, f(x_i; \theta))
\]

gives the fraction of prediction errors on the training set.

- This is a function of the parameters \( \theta \) and we can try to minimize it directly.
Estimation criterion cont’d

- We have reduced the estimation problem to a minimization problem

\[
\text{find } \theta \text{ that minimizes } \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, f(x_i; \theta))
\]
Estimation criterion cont’d

- We have reduced the estimation problem to a minimization problem

\[
\text{find } \theta \text{ that minimizes } \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, f(x_i; \theta))
\]

- valid for any parameterized class of mappings from examples to predictions
- valid when the predictions are discrete labels, real valued, or other provided that the loss is defined appropriately
- may be ill-posed (under-constrained) as stated
Estimation criterion cont’d

- We have reduced the estimation problem to a minimization problem

\[
\text{find } \theta \text{ that minimizes } \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, f(x_i; \theta))
\]

- valid for any parameterized class of mappings from examples to predictions
- valid when the predictions are discrete labels, real valued, or other provided that the loss is defined appropriately
- may be ill-posed (under-constrained) as stated

- But why is it sensible to minimize the empirical loss in the first place since we are only interested in the performance on new examples?
Training and test performance: sampling

- We assume that each training and test example-label pair, \((x, y)\), is drawn independently at random from the same but unknown population of examples and labels.

- We can represent this population as a joint probability distribution \(P(x, y)\) so that each training/test example is a sample from this distribution \((x_i, y_i) \sim P\).
Training and test performance: sampling

- We assume that each training and test example-label pair, \((x, y)\), is drawn independently at random from the same but unknown population of examples and labels.

- We can represent this population as a joint probability distribution \(P(x, y)\) so that each training/test example is a sample from this distribution \((x_i, y_i) \sim P\)

\[
\text{Empirical (training) loss} = \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, f(x_i; \theta))
\]

\[
\text{Expected (test) loss} = E_{(x,y)\sim P} \left\{ \text{Loss}(y, f(x; \theta)) \right\}
\]

- The training loss based on a few sampled examples and labels serves as a proxy for the test performance measured over the whole population.
Topics

- The learning problem
  - hypothesis class, estimation algorithm
  - loss and estimation criterion
  - sampling, empirical and expected losses

- Regression, example

- Linear regression
  - estimation, errors, analysis
Regression

- The goal is to make quantitative (real valued) predictions on the basis of a (vector of) features or attributes

- Example: predicting vehicle fuel efficiency (mpg) from 8 attributes

<table>
<thead>
<tr>
<th>y</th>
<th>cyls</th>
<th>disp</th>
<th>hp</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.0</td>
<td>8</td>
<td>307.0</td>
<td>130.00</td>
<td>3504</td>
</tr>
<tr>
<td>26.0</td>
<td>4</td>
<td>97.00</td>
<td>46.00</td>
<td>1835</td>
</tr>
<tr>
<td>33.5</td>
<td>4</td>
<td>98.00</td>
<td>83.00</td>
<td>2075</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Regression

- The goal is to make quantitative (real valued) predictions on the basis of a (vector of) features or attributes

- Example: predicting vehicle fuel efficiency (mpg) from 8 attributes

<table>
<thead>
<tr>
<th>y</th>
<th>cyls</th>
<th>disp</th>
<th>hp</th>
<th>weight</th>
<th>. . .</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.0</td>
<td>8</td>
<td>307.0</td>
<td>130.0</td>
<td>3504</td>
<td>. . .</td>
</tr>
<tr>
<td>26.0</td>
<td>4</td>
<td>97.00</td>
<td>46.00</td>
<td>1835</td>
<td>. . .</td>
</tr>
<tr>
<td>33.5</td>
<td>4</td>
<td>98.00</td>
<td>83.00</td>
<td>2075</td>
<td>. . .</td>
</tr>
<tr>
<td>. .</td>
<td>. .</td>
<td>. .</td>
<td>. .</td>
<td>. .</td>
<td>. .</td>
</tr>
</tbody>
</table>

- We need to
  - specify the class of functions (e.g., linear)
  - select how to measure prediction loss
  - solve the resulting minimization problem
• We begin by considering linear regression (easy to extend to more complex predictions later on)

\[ f : \mathcal{R} \rightarrow \mathcal{R} \quad f(x; \mathbf{w}) = w_0 + w_1 x \]

\[ f : \mathcal{R}^d \rightarrow \mathcal{R} \quad f(x; \mathbf{w}) = w_0 + w_1 x_1 + \ldots w_d x_d \]

where \( \mathbf{w} = [w_0, w_1, \ldots, w_d]^T \) are parameters we need to set.
Linear regression: squared loss

\[ f : \mathcal{R} \rightarrow \mathcal{R} \quad f(x; w) = w_0 + w_1 x \]

\[ f : \mathcal{R}^d \rightarrow \mathcal{R} \quad f(x; w) = w_0 + w_1 x_1 + \ldots w_d x_d \]

- We can measure the prediction loss in terms of squared error, \( \text{Loss}(y, \hat{y}) = (y - \hat{y})^2 \), so that the empirical loss on \( n \) training samples becomes mean squared error

\[ J_n(w) = \frac{1}{n} \sum_{i=1}^{n} \left( y_i - f(x_i; w) \right)^2 \]
Linear regression: estimation

- We have to minimize the *empirical* squared loss

\[
J_n(w) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i; w))^2
\]

\[
= \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2 \quad (1\text{-dim})
\]

By setting the derivatives with respect to \(w_1\) and \(w_0\) to zero, we get necessary conditions for the “optimal” parameter values

\[
\frac{\partial}{\partial w_1} J_n(w) = 0
\]

\[
\frac{\partial}{\partial w_0} J_n(w) = 0
\]
Optimality conditions: derivation

\[
\frac{\partial}{\partial w_1} J_n(w) = \frac{\partial}{\partial w_1} \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2
\]
Optimality conditions: derivation

\[
\frac{\partial}{\partial w_1} J_n(w) = \frac{\partial}{\partial w_1} \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2
\]

\[
= \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial w_1} (y_i - w_0 - w_1 x_i)^2
\]
Optimality conditions: derivation

\[
\frac{\partial}{\partial w_1} J_n(w) = \frac{\partial}{\partial w_1} \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2
\]

\[
= \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial w_1} (y_i - w_0 - w_1 x_i)^2
\]

\[
= \frac{2}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i) \frac{\partial}{\partial w_1} (y_i - w_0 - w_1 x_i)
\]
Optimality conditions: derivation

\[
\frac{\partial}{\partial w_1} J_n(w) = \frac{\partial}{\partial w_1} \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2 \\
= \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial w_1} (y_i - w_0 - w_1 x_i)^2 \\
= \frac{2}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i) \frac{\partial}{\partial w_1} (y_i - w_0 - w_1 x_i) \\
= \frac{2}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)(-x_i) = 0
\]
Optimality conditions: derivation

\[
\frac{\partial}{\partial w_1} J_n(w) = \frac{\partial}{\partial w_1} \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2
\]

\[
= \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial w_1} (y_i - w_0 - w_1 x_i)^2
\]

\[
= \frac{2}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i) \frac{\partial}{\partial w_1} (y_i - w_0 - w_1 x_i)
\]

\[
= \frac{2}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i) (-x_i) = 0
\]

\[
\frac{\partial}{\partial w_0} J_n(w) = \frac{2}{n} \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i) (-1) = 0
\]
Interpretation

- If we denote the prediction error as $\epsilon_i = (y_i - w_0 - w_1x_i)$ then the optimality conditions can be written as

$$\frac{1}{n} \sum_{i=1}^{n} \epsilon_i x_i = 0, \quad \frac{1}{n} \sum_{i=1}^{n} \epsilon_i = 0$$

Thus the prediction error is uncorrelated with any linear function of the inputs.
Interpretation

- If we denote the prediction error as $\epsilon_i = (y_i - w_0 - w_1 x_i)$ then the optimality conditions can be written as

$$\frac{1}{n} \sum_{i=1}^{n} \epsilon_i x_i = 0, \quad \frac{1}{n} \sum_{i=1}^{n} \epsilon_i = 0$$

Thus the prediction error is uncorrelated with any linear function of the inputs

but not with a quadratic function of the inputs

$$\frac{1}{n} \sum_{i=1}^{n} \epsilon_i x_i^2 \neq 0 \quad \text{(in general)}$$
Linear regression: matrix notation

- We can express the solution a bit more generally by resorting to a matrix notation

\[ y = \begin{bmatrix} y_1 \\ \cdots \\ y_n \end{bmatrix}, \quad X = \begin{bmatrix} 1 & x_1 \\ \cdots & \cdots & \cdots \\ 1 & x_n \end{bmatrix}, \quad w = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \]

so that

\[
\frac{1}{n} \sum_{t=1}^{n} (y_t - w_0 - w_1x_t)^2 = \frac{1}{n} \left\| \begin{bmatrix} y_1 \\ \cdots \\ y_n \end{bmatrix} - \begin{bmatrix} 1 & x_1 \\ \cdots & \cdots & \cdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \right\|^2 \\
= \frac{1}{n} \| y - Xw \|^2
\]
Linear regression: solution

By setting the derivatives of $\|y - Xw\|^2/n$ to zero, we get the same optimality conditions as before, now expressed in a matrix form

$$\frac{\partial}{\partial w} \frac{1}{n} \|y - Xw\|^2 = \frac{\partial}{\partial w} \frac{1}{n} (y - Xw)^T (y - Xw)$$
Linear regression: solution

By setting the derivatives of $\|y - Xw\|^2/n$ to zero, we get the same optimality conditions as before, now expressed in a matrix form

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

$$= \frac{2}{n} X^T (y - Xw)$$
Linear regression: solution

By setting the derivatives of $\|y - Xw\|^2/n$ to zero, we get the same optimality conditions as before, now expressed in a matrix form

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

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

$$= \frac{2}{n} (X^T y - X^T Xw) = 0$$

which gives

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

• The solution is a linear function of the outputs $y$
Linear regression: generalization

- As the number of training examples increases our solution gets “better”

We’d like to understand the error a bit better
Linear regression: types of errors

- **Structural error** measures the error introduced by the limited function class (infinite training data):

\[
\min_{w_1, w_0} E_{(x,y) \sim P} (y - w_0 - w_1 x)^2 = E_{(x,y) \sim P} (y - w_0^* - w_1^* x)^2
\]

where \((w_0^*, w_1^*)\) are the optimal linear regression parameters.
Linear regression: types of errors

- **Structural error** measures the error introduced by the limited function class (infinite training data):

  \[
  \min_{w_1, w_0} E_{(x,y) \sim P} (y - w_0 - w_1 x)^2 = E_{(x,y) \sim P} (y - w_0^* - w_1^* x)^2
  \]

  where \((w_0^*, w_1^*)\) are the optimal linear regression parameters.

- **Approximation error** measures how close we can get to the optimal linear predictions with limited training data:

  \[
  E_{(x,y) \sim P} (w_0^* + w_1^* x - \hat{w}_0 - \hat{w}_1 x)^2
  \]

  where \((\hat{w}_0, \hat{w}_1)\) are the parameter estimates based on a small training set (therefore themselves random variables).
Linear regression: error decomposition

- The expected error of our linear regression function decomposes into the sum of structural and approximation errors

$$
E_{(x,y) \sim P} (y - \hat{w}_0 - \hat{w}_1 x)^2 = \\
E_{(x,y) \sim P} (y - w^*_0 - w^*_1 x)^2 + \\
E_{(x,y) \sim P} (w^*_0 + w^*_1 x - \hat{w}_0 - \hat{w}_1 x)^2
$$

![Graph showing mean squared error vs. number of training examples](image-url)
Error decomposition: derivation

\[
E_{(x,y)\sim P} (y - \hat{w}_0 - \hat{w}_1 x)^2
= E_{(x,y)\sim P} ((y - w^*_0 - w^*_1 x) + (w^*_0 + w^*_1 x - \hat{w}_0 - \hat{w}_1 x))^2
= E_{(x,y)\sim P} (y - w^*_0 - w^*_1 x)^2
+ E_{(x,y)\sim P} 2(y - w^*_0 - w^*_1 x)(w^*_0 + w^*_1 x - \hat{w}_0 - \hat{w}_1 x)
+ E_{(x,y)\sim P} (w^*_0 + w^*_1 x - \hat{w}_0 - \hat{w}_1 x)^2
\]

The second term has to be zero since the error \((y - w^*_0 - w^*_1 x)\) of the best linear predictor is necessarily uncorrelated with any linear function of the input including \((w^*_0 + w^*_1 x - \hat{w}_0 - \hat{w}_1 x)\)