Outline

- Model complexity, overfitting, bias and variance
- Regularization – Ridge regression, Lasso
  - Linear regression
  - Logistic regression
- Instance-based learning
Model complexity

No universal definition

Heading for the regularization . . .

model complexity is the number of hypothesis parameters

$$\Theta = \langle \theta_0, \ldots, \theta_m \rangle$$
Model complexity

Finding a model that minimizes generalization error

... is one of central goals of the machine learning process
Model complexity

Complexity of decision boundary for classification

![Graph showing decision boundaries](image)
Bias and variance

1. Select a machine learning algorithm
2. Get $k$ different training sets
3. Get $k$ predictors

- **Bias** measures error that originates from the learning algorithm
  - how far off in general the predictions by $k$ predictors are from the true output value

- **Variance** measures error that originates from the training data
  - how much the predictions for a test instance vary between $k$ predictors
Bias and variance

**Generalization error** $\text{error}_D(\hat{f})$ measures how well a hypothesis $\hat{f}$ ($f$ is a true target function) generalizes beyond the used training data set, to unseen data with distribution $D$. Usually it is defined as follows

- for **regression**: $\text{error}_D(\hat{f}) = E[(\hat{y}_i - y_i)^2]$
- for **classification**: $\text{error}_D(\hat{f}) = \Pr(\hat{y}_i \neq y_i)$

**Decomposition of** $\text{error}_D(\hat{f}) = \text{Bias}^2 + \text{Variance} + \text{IrreducibleError}$

For simplicity, ignore IrreducibleError.
Bias and variance

Regression

$$error_D(\hat{f}) = (E[\hat{f}(x)] - f(x))^2 + E[(\hat{f}(x) - E[\hat{f}(x)])^2]$$
Bias and variance

Classification

Zero-one (0-1) loss function $L(\hat{y}, y) = I(\hat{y}y \leq 0)$, indicator variable $I$ is 1 if $y\hat{y} \leq 0$, 0 otherwise

<table>
<thead>
<tr>
<th>Single loss</th>
<th>regression</th>
<th>Expected loss</th>
<th>classification</th>
<th>Main prediction $E[\hat{y}]$</th>
<th>Bias $^2$</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single loss</td>
<td>RSS</td>
<td>Expected loss</td>
<td>0-1</td>
<td>Mean</td>
<td>$(y - E[\hat{y}])^2$</td>
<td>$E[(E[\hat{y}] - \hat{y})^2]$</td>
</tr>
<tr>
<td>Expected loss</td>
<td>$E[(y - \hat{y})^2]$</td>
<td>$E[L(y, \hat{y})]$</td>
<td>$E[L(y, E[\hat{y}])$</td>
<td>Mean</td>
<td>$L(y, E[\hat{y}])$</td>
<td>$E[L(\hat{y}, E[\hat{y}])]$</td>
</tr>
</tbody>
</table>

For more details see


Bias and variance

• underfitting = high bias

• overfitting = high variance
Bias and variance

- **Overfitting**: high variance
- **Underfitting**: high bias
- **Good balance**: low variance, low bias
We want a model in between which is

- powerful enough to model the underlying structure of data
- not so powerful to model the structure of the training data

Let’s prevent overfitting by complexity regularization, a technique that regularizes the parameter estimates, or equivalently, shrinks the parameter estimates towards zero.
A machine learning algorithm estimates hypothesis parameters $\Theta = \langle \theta_0, \theta_1, \ldots, \theta_m \rangle$ using $\Theta^*$ that minimizes loss function $L$ for training data $Data = \{ \langle x_i, y_i \rangle, x_i = \langle x_{i1}, \ldots, x_{im} \rangle, y_i \in Y \}$

$$\Theta^* = \arg\min_{\Theta} L(\Theta)$$

**Regularization**

$$\Theta^*_R = \arg\min_{\Theta} L(\Theta) + \lambda \cdot \text{penalty}(\Theta), \text{ where } \lambda \geq 0 \text{ is a tuning parameter}$$

Infact, the penalty is applied to $\theta_1, \ldots, \theta_m$, but not to $\theta_0$ since the goal is to regularize the estimated association between each feature and the target value.
Regularized linear regression

\[ \theta_0 + \theta_{\text{age}} x \]

- training examples
Regularized linear regression

\[ \theta_0 + \theta_{\text{age}} x \]

- training examples
- test examples
Regularized linear regression

\[ \theta_0 + \theta_{age} x \]

- **training examples**
- **test examples**

regularize \( \theta_{age} \)
Regularized linear regression

\[ f(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_m x_m \]

\[ L(\Theta) = RSS = \sum_{i=1}^{n} (f(x_i) - y_i)^2 \]

\[ \Theta^*_R = \text{argmin}_\Theta [RSS + \lambda \cdot \text{penalty}(\Theta)] \]
Ridge regression

$$\Theta^*_R = \arg\min_{\Theta} [RSS + \lambda \cdot (\theta_1^2 + \cdots + \theta_m^2)]$$

The larger $\lambda$, $\theta_{age}$ gets asymptotically closer to 0 and salary is less sensitive to age.
Ridge regression

- Let $\theta^*_1, \ldots, \theta^*_m$ be ridge regression parameter estimates for a particular value of $\lambda$
- Let $\theta^*_1, \ldots, \theta^*_m$ be unregularized parameter estimates
- $0 \leq \frac{\theta^*_1^2 + \cdots + \theta^*_m^2}{\theta^*_1^2 + \cdots + \theta^*_m^2} \leq 1$ ... the amount that the ridge regression parameter estimates have been shrunken towards; a small value indicates that they have been shrunken very close to zero
- **When** $\lambda = 0$, **then** $\theta^*_{\lambda_i} = \theta^*_i$ for $i = 1, \ldots, m$
- **When** $\lambda$ is extremely large, **then** $\theta^*_{\lambda_i}$ is very small for $i = 1, \ldots, m$
- **When** $\lambda$ between, we are fitting a model and shrinking the parameters
Ridge regression

-4 -2 0 2 4

Log Lambda
Coefficients

NPFL054, 2023 Hladká & Holub Lecture 10, page 21/49
Lasso

\[
\text{penalty}(\Theta) = |\theta_1| + \cdots + |\theta_m|
\]

- Let \( \theta^*_{\lambda_1}, \ldots, \theta^*_{\lambda_m} \) be lasso regression parameter estimates
- Let \( \theta^*_1, \ldots, \theta^*_m \) be unregularized parameter estimates
- \textbf{When} \( \lambda = 0 \), \textbf{then} \( \theta^*_{\lambda_i} = \theta^*_i \) for \( i = 1, \ldots, m \)
- \textbf{When} \( \lambda \) grows, \textbf{then} the impact of penalty grows
- \textbf{When} \( \lambda \) is extremely large, \textbf{then} \( \theta^*_{\lambda_i} = 0 \) for \( i = 1, \ldots, m \)
Lasso

Log Lambda

Coefficients

98 96 91 58 ...

121
122
123
127
128
129 130 133
134
135 136
139
149
150
151
152
153
154
156 157
160
161
162
163
165
168

Log Lambda
Ridge regression shrinks all the parameters but eliminates none, while the Lasso can shrink some parameters to zero.
Elastic net

\[
\text{penalty}(\Theta) = \lambda_1 \cdot (|\theta_1| + \cdots + |\theta_m|) + \lambda_2 \cdot (\theta_1^2 + \cdots + \theta_m^2) \]

\[0 \leq \lambda_1, \lambda_2\] are tuning parameters
Ridge regression

Alternative formulation

\[ \Theta^*_R = \arg\min_{\Theta} \sum_{i=1}^{n} (f(x_i) - y_i)^2 \]

subject to \( \theta_1^2 + \cdots + \theta_m^2 \leq s \)

- the gray circle represents the feasible region for Ridge regression
- the contours represent different RSS values for the unregularized model

\[ \Theta^*_R = \arg\min_{\Theta} L(\Theta) + \lambda(\theta_1^2 + \theta_2^2) \]
If $s$ is large enough, i.e. $\lambda = 0$, so that the minimum RSS value falls into the region of **ridge regression** parameter estimates then the alternative formulation yields the least square estimates.
Lasso
Alternative formulation

\[ \Theta^*_R = \arg\min_{\Theta} \sum_{i=1}^{n} (f(x_i) - y_i)^2 \]

subject to \(|\theta_1| + \cdots + |\theta_m| \leq s\)

- the grey square represents the feasible region of the Lasso
- the contours represent different RSS values for the unregularized model
• If $s$ is large enough, i.e. $\lambda = 0$, so that the minimum RSS value falls into the region of loss parameter estimates then the alternative formulation yields the primary solution.
Elastic net
Logistic regression

Sigmoid function $f(x) = \frac{1}{1 + e^{-\Theta^\top x}}$

Loss function $L(\Theta) = -\sum_{i=1}^{n} y_i \log P(y_i|\mathbf{x}_i; \Theta) + (1 - y_i) \log(1 - P(y_i|\mathbf{x}_i; \Theta))$
Regularized logistic regression

Ridge regression

$$\Theta^*_R = \arg\min_{\Theta} \left[ \sum_{i=1}^{n} y_i \log(f(x_i)) + (1 - y_i) \log(1 - f(x_i)) \right] + \lambda \sum_{j=1}^{m} \theta_j^2 ] =$$

$$= \arg\min_{\Theta} \left[ \sum_{i=1}^{n} y_i (-\log(f(x_i))) + (1 - y_i)(-\log(1 - f(x_i))) + \lambda \sum_{j=1}^{m} \theta_j^2 \right] =$$

$$= \arg\min_{\Theta} \left[ \sum_{i=1}^{n} y_i L_1(\Theta) + (1 - y_i)L_0(\Theta) + \lambda \sum_{j=1}^{m} \theta_j^2 \right]$$
Since

$$A + \lambda B \equiv CA + B, \quad C = \frac{1}{\lambda}$$

then

$$\Theta^*_R = \arg\min_\Theta \left[ \sum_{j=1}^{m} \theta_j^2 + C \left[ \sum_{i=1}^{n} y_i L_1(\Theta) + (1 - y_i) L_0(\Theta) \right] \right]$$

where

$$L_1(\Theta) = -\log \frac{1}{1 + e^{-\Theta^\top x}}$$

$$L_0(\Theta) = -\log(1 - \log \frac{1}{1 + e^{-\Theta^\top x}})$$
Regularized logistic regression
Ridge regression

\[ \Theta^*_R = \arg\min_\Theta \left[ \sum_{j=1}^m \theta_j^2 + C \sum_{i=1}^n \log(1 + e^{-\bar{y}_i \Theta^T x_i}) \right] \]

where

\[ \bar{y}_i = \begin{cases} 
-1 & \text{if } y_i = 0 \\
+1 & \text{if } y_i = 1
\end{cases} \]
SVM
Soft margin classifier

\[ \Theta^* = \arg\min_{\Theta} \sum_{j=1}^{m} \theta_j^2 + C \sum_{i=1}^{n} \xi_i \]

\( \xi_i \geq 0 \) is equivalent to \( \xi_i = \max(0, 1 - y_i \Theta^\top x_i) \), i.e.

\[ \Theta^* = \arg\min_{\Theta} \left[ \sum_{j=1}^{m} \theta_j^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i \Theta^\top x_i) \right] \]

s.t. \( \Theta^\top x_i \geq 1 - \xi_i \) if \( y_i = +1 \) and \( \Theta^\top x_i \leq -1 + \xi_i \) if \( y_i = -1 \)
**SVM**

**Soft margin classifier**

**Hinge loss** = \( \max(0, 1 - y_i \Theta^T x_i) \)

1. \( y_i \Theta^T x_i > 1 \): no contribution to loss
2. \( y_i \Theta^T x_i = 1 \): no contribution to loss
3. \( y_i \Theta^T x_i < 1 \): contribution to loss

![Hinge loss graph](image)
Soft-margin is equivalent to the regularization problem.
Instance-based learning

Key idea

- IBL methods = supervised ML methods
- IBL methods initially store training data, we call them lazy methods
- For a new instance, prediction is based on local similarity, i.e. a set of similar instances are retrieved and used for prediction
- IBL methods can construct a different approximation of a target function for each distinct test instance
- Both classification and regression
Instance-based learning

Key points

1. A distance metric
2. How many nearby neighbours look at?
3. A weighting function
4. How to fit with local points?
Instance-based learning

Distance metric

Recall distance used as dissimilarity metrics for clustering. The most common ones are:

- **Euclidean distance**

  \[ d(x_i, x_j) = \sqrt{\sum_{r=1}^{m} (x_{ir} - x_{jr})^2} \]

- **Manhattan distance**

  \[ d(x_i, x_j) = \sum_{r=1}^{m} |x_{ir} - x_{jr}| \]
Instance-based learning

$k$-Nearest Neighbour algorithm

1. A distance metric: Euclidian (most widely used)
2. How many nearby neighbours look at? \(k\) training instances closest to \(x\)
3. A weighting function: unused
4. How to fit with local points?
   - **k-NN classification**
     \[
     f(x) = \arg\max_{v \in Y} \sum_{i=1}^{k} \delta(v, y_i),
     \]
     where \(\delta(a, b) = 1\) if \(a = b\), otherwise 0
   - **k-NN regression**
     \[
     f(x) = \sum_{i=1}^{k} y_i / k
     \]
Instance-based learning

Distance-weighted $k$-NN algorithm

1. **A distance metric**: Euclidian (most widely used)

2. **How many nearby neighbours look at?** $k$ training instances closest to $x$

3. **A weighting function**: greater weight of closer neighbours, e.g.,

$$w_i(x) \equiv \frac{1}{d(x, x_i)^2}$$

4. **How to fit with local points?**

   - **Classification**

     $$f(x) = \arg\max_{v \in \mathcal{Y}} \sum_{i=1}^{k} w_i(x) \delta(v, y_i)$$ (3)

   - **Regression**

     $$f(x) = \sum_{i=1}^{k} w_i(x) y_i / \sum_{i=1}^{k} w_i(x)$$ (4)
Instance-based learning
Distance-weighted $k$-NN algorithm

Shepard’s method

- **Classification**

\[
 f(x) = \arg\max_{v \in Y} \sum_{i=1}^{n} w_i(x) \delta(v, y_i)
\]  

- **Regression**

\[
 f(x) = \frac{\sum_{i=1}^{n} w_i(x)y_i}{\sum_{i=1}^{n} w_i(x)}
\]
Instance-based learning
Locally weighted linear regression

1. A distance metric: Euclidian (most widely used)
2. How many nearby neighbours look at? $k$ training instances closest to $x$
3. A weighting function: $w_i(x)$
4. How to fit with local points?

$$\Theta^* = \arg\min_\Theta \sum_{i=1}^k w_i(x)(\Theta^T x_i - y_i)^2$$  \hspace{1cm} (7)
Instance-based learning
Locally weighted linear regression
Instance-based learning
LW linear regression vs. simple regression
Bias and variance

\( k \)-Nearest Neighbor

- \( \uparrow k \) → smoother decision boundary \( \rightarrow \downarrow \) variance and \( \uparrow \) bias
- \( \downarrow k \) \( \rightarrow \uparrow \) variance and \( \downarrow \) bias
Bias and variance

k-Nearest Neighbor

5-nearest neighbour

15-nearest neighbour
Summary of Examination Requirements

- Model complexity, generalization error, Bias and variance
- Lasso and Ridge regularization for linear and logistic regression
- Soft margin classifier and regularization
- k-NN algorithm