<table>
<thead>
<tr>
<th>Assignment</th>
<th>Date of assignment</th>
<th>Deadline (midnight 23:59)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW1</td>
<td>Sep 7</td>
<td>Sep 20</td>
</tr>
<tr>
<td>HW2</td>
<td>Sep 21</td>
<td>Oct 4</td>
</tr>
<tr>
<td>HW3</td>
<td>Oct 5</td>
<td>Oct 18</td>
</tr>
<tr>
<td>Paper summary</td>
<td>Oct 12</td>
<td>Oct 26</td>
</tr>
<tr>
<td>HW4</td>
<td>Oct 19</td>
<td>Nov 1</td>
</tr>
<tr>
<td>HW5</td>
<td>Nov 2</td>
<td>Nov 15</td>
</tr>
<tr>
<td>HW6</td>
<td>Nov 23</td>
<td>Dec 6</td>
</tr>
<tr>
<td>Project</td>
<td><strong>Oct 5 - 7</strong></td>
<td><strong>Dec 14 - 16</strong></td>
</tr>
</tbody>
</table>

Intermediate presentations are **Nov 16 - 18**
Ensemble Learning
“In previous episodes...”
There are two main ways of looking at this (A and B)

**way A:** something wrong with a model

**way B:** something wrong with a data

Dataset

Epoch #5

Model

Validation

Overfitting

batch #1

batch #2

batch #3
Regularisation methods

Way A methods

Way B methods
Way A methods
Explicit regularisation methods (Dropout, L1/L2 regularisation)
Despite some differences, these methods belong to so called **explicit regularisation methods**.

They **explicitly limit** predictive power of the model by introducing **additional constraints**.

**L2/L1 regularisation** VS **Dropout**
Way B methods
Implicit regularisation methods
These are the most basic transforms
More advanced things are available via github/aleju/imgaug
Explicit regularisation methods
(Dropout, L1/L2 regularisation)

Implicit regularisation methods
There are plenty of other cool ways to get **better performance** and **fight overfitting**!
There are plenty of other cool ways to get better performance and fight overfitting!

Weak learners, ENSEMBLE!
Multiple models are built on training data

Linear Regression

\[ y = 1.6 + 0.79x \]
Multiple models are built on training data.
Multiple models are built on training data
Multiple models are built on training data.

Ridge Regression

Linear Regression

Lasso Regression

Multiple models are built on training data.
Multiple models are built on training data
Multiple models are built on training data
Multiple models are built on training data

**Ridge Regression**

**Linear Regression**

**Lasso Regression**
Multiple models are built on training data

- **Ridge Regression**
- **Linear Regression**
- **Lasso Regression**

Multiple models are built on training data.
Multiple models are built on training data

Ridge Regression

Linear Regression

Lasso Regression
Multiple models are built on training data

Ridge Regression

Linear Regression

Lasso Regression

Multiple models are built on training data
Multiple models are built on training data

\[
\begin{align*}
\text{Ridge Regression} & : y = 1.6 + 0.79x \\
\text{Linear Regression} & : y = 3.28 + 0.14x \\
\text{Lasso Regression} & : y = 1.94 + 0.64x \\
\end{align*}
\]
Multiple models are built on training data.
Multiple models are built on training data.
Multiple models are built on training data.

\[ y = 1.6 + 0.79x_1 + 0.14x_2 + 0.64x_3 \]

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

- Linear Regression
- Ridge Regression
- Lasso Regression
Multiple models are built on training data

\[
\begin{align*}
1.6 + 0.79x^2 \\
3.28 + 0.14x^3.28 + 0.14x
\end{align*}
\]

\[
\begin{array}{c|c|c}
\text{X1} & \text{X2} \\
\hline
2 & 5 \\
\end{array}
\]
Multiple models are built on training data.

Linear Regression: $1.6 + 0.79x$

Ridge Regression: $1.94 + 0.64x$

Lasso Regression: $3.28 + 0.14x$

$X_1 = 2 \quad X_2 = 5$
Multiple models are built on training data.

Linear Regression

\[ y = 1.6 + 0.79x \]

Ridge Regression

\[ y = 3.18 + 0.14x \]

Lasso Regression

\[ y = 1.94 + 0.64x \]

Multiple models are built on training data.

\[ X_1 = 2 \quad X_2 = 5 \]

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td></td>
</tr>
</tbody>
</table>
Multiple models are built on training data.
Multiple models are built on training data

\[ y = 1.6 + 0.79^*x \]

Linear Regression

Ridge Regression

Lasso Regression

\[ y = 3.28 + 0.14^*x \]

\[ y = 1.94 + 0.64^*x \]

\[ X_1 = 2 \quad X_2 = 5 \]

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td></td>
</tr>
</tbody>
</table>
Multiple models are built on training data.
Multiple models are built on training data

- **Linear Regression**: $y = 1.6 + 0.79x$
- **Ridge Regression**: $y = 1.94 + 0.64x$
- **Lasso Regression**: $y = 3.28 + 0.14x$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td></td>
</tr>
</tbody>
</table>

$x_1 = 2$, $x_2 = 5$
Multiple models are built on training data.
Multiple models are built on training data

Linear Regression

Ridge Regression

Lasso Regression

\[ y = 1.6 + 0.79x \]
\[ y = 3.28 + 0.14x \]

\[ y = 1.94 + 0.64x \]

\[ y = 5.55 \]

\[ X_1 = 2 \quad X_2 = 5 \]

<table>
<thead>
<tr>
<th>LR</th>
<th>3.18</th>
</tr>
</thead>
</table>

Multiple models are built on training data.
Multiple models are built on training data.
Multiple models are built on training data

\[
\text{Linear Regression: } y = 1.6 + 0.79x
\]

\[
\text{Ridge Regression: } y = 3.28 + 0.14x
\]

\[
\text{Lasso Regression: } y = 1.94 + 0.64x
\]

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
</tbody>
</table>

\( X_1 = 2 \quad X_2 = 5 \)
Multiple models are built on training data
Multiple models are built on training data.

For the given data points, the equations for the models are:

1. Linear Regression: \( y = 1.6 + 0.79x \)
2. Ridge Regression: \( y = 3.28 + 0.14x \)
3. Lasso Regression: \( y = 1.94 + 0.64x \)

The table shows the predictions for two different inputs, \( X_1 = 2 \) and \( X_2 = 5 \):

<table>
<thead>
<tr>
<th>Model</th>
<th>Input X1</th>
<th>Predicted Y1</th>
<th>Input X2</th>
<th>Predicted Y2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The graph illustrates the fitted lines for each model and the data points.
Multiple models are built on training data

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ y = 1.6 + 0.79x \]
\[ y = 3.28 + 0.14x \]
Multiple models are built on training data

Multiple models are built on training data
Multiple models are built on training data.
Multiple models are built on training data.

- **Ridge Regression**
- **Linear Regression**
- **Lasso Regression**

Predictions:

- LR: 3.18, 5.55
- Ridge: 3.23, 5.17
- Lasso: 3.57, 4
- Average: 3.39, 4.83

Average the predictions.

\[ X_1 = 2 \quad X_2 = 5 \]
Multiple models are built on training data.

\[
\begin{align*}
Y &= 1.6 + 0.79x + 0.14x + 0.64x \\
X_1 &= 2 \quad X_2 = 5
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Average the predictions

\[
\text{mean}(3.18, 3.23, 3.57)
\]
Multiple models are built on training data

**Linear Regression**

\[ y = 1.6 + 0.79x + 3.28 + 0.14x \]

**Ridge Regression**

\[ y = 1.94 + 0.64x \]

**Lasso Regression**

\[ y = 3.28 + 0.14x \]

Multiple models are built on training data

\[ X_1 = 2 \quad X_2 = 5 \]

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4.00</td>
</tr>
<tr>
<td>Average</td>
<td>3.32</td>
<td></td>
</tr>
</tbody>
</table>

**Average the predictions**

\[ \text{mean}(3.18, 3.23, 3.57) \]
Multiple models are built on training data

\[
\begin{align*}
\text{Ridge Regression} & : \quad y = 1.6 + 0.79x \\
\text{Linear Regression} & : \quad y = 3.28 + 0.14x \\
\text{Lasso Regression} & : \quad y = 1.94 + 0.64x
\end{align*}
\]

\[
X_1 = 2 \quad X_2 = 5
\]

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td>3.32</td>
<td></td>
</tr>
</tbody>
</table>

\[
\text{Average} = \frac{3.18 + 5.55 + 5.17 + 4}{4} = 5.1875
\]

Average the predictions
Multiple models are built on training data

Average the predictions

\[
\begin{align*}
X_1 &= 2 & X_2 &= 5 \\
\begin{array}{|c|c|}
\hline
& 1 & 2 \\
\hline
LR & 3.18 & 5.55 \\
Ridge & 3.23 & 5.17 \\
Lasso & 3.57 & 4 \\
Average & 3.32 & 4.89 \\
\hline
\end{array}
\]

mean(5.55, 5.17, 4)
Multiple models are built on training data

Average the predictions

\begin{align*}
X_1 &= 2 \quad X_2 = 5
\end{align*}

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td>3.32</td>
<td>4.89</td>
</tr>
</tbody>
</table>
Multiple models are built on training data

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4.00</td>
</tr>
<tr>
<td>Average</td>
<td>3.32</td>
<td>4.89</td>
</tr>
</tbody>
</table>

TRUE | 4   | 5   |

$X_1 = 2$  $X_2 = 5$
Multiple models are built on training data.

Multiple models are built on training data.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td>3.32</td>
<td>4.89</td>
</tr>
<tr>
<td>TRUE</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Basic ensemble
In 1907, 787 villagers tried to guess the weight of an ox.

James Surowiecki, *The Wisdom of Crowds*
In 1907, 787 villagers tried to guess the weight of an ox. None of them guessed it correctly, but the average guess (542.9 kg) was very close to the actual weight of the ox (543.4 kg). 

James Surowiecki, *The Wisdom of Crowds*
Multiple models are built on training data.

Similar idea can be applied for **classification**.
Multiple models are built on training data

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td>3.32</td>
<td>4.89</td>
</tr>
<tr>
<td>TRUE</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Basic ensemble

$X_1 = 2 \quad X_2 = 5$

$1.94 + 0.64x$

$1.6 + 0.79x$

$3.28 + 0.14x$
Multiple models are built on training data.

Due to random weight initialisation, and data partition, trained models usually turn out to be slightly different.
Multiple models are built on training data

Training data

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>3.18</td>
<td>5.55</td>
</tr>
<tr>
<td>Ridge</td>
<td>3.23</td>
<td>5.17</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.57</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td>3.32</td>
<td>4.89</td>
</tr>
<tr>
<td>TRUE</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Basic ensemble

Basic ensemble

$X_1 = 2$  $X_2 = 5$
Multiple models are built on training data.

Training data

Test

<table>
<thead>
<tr>
<th>Model 1</th>
<th>Model 2</th>
<th>Model 3</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Dog</td>
</tr>
</tbody>
</table>

Basic ensemble
Multiple models are built on training data.

Training data

- Model 1
- Model 2
- Model 3

Basic ensemble

Ensemble

- TRUE

Test

<table>
<thead>
<tr>
<th>Model 1</th>
<th>Dog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 2</td>
<td></td>
</tr>
<tr>
<td>Model 3</td>
<td></td>
</tr>
</tbody>
</table>

Basic ensemble

- M1
- M2
- M3
Multiple models are built on training data

<table>
<thead>
<tr>
<th>Test</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>Dog</td>
</tr>
<tr>
<td>Model 2</td>
<td>Dog</td>
</tr>
<tr>
<td>Model 3</td>
<td></td>
</tr>
<tr>
<td>Ensemble</td>
<td></td>
</tr>
<tr>
<td>TRUE</td>
<td>Dog</td>
</tr>
</tbody>
</table>

Basic ensemble
Multiple models are built on training data

Training data

<table>
<thead>
<tr>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
</tr>
<tr>
<td>Model 2</td>
</tr>
<tr>
<td>Model 3</td>
</tr>
<tr>
<td>Ensemble</td>
</tr>
</tbody>
</table>

Basic ensemble
Multiple models are built on training data.

Training data

<table>
<thead>
<tr>
<th>Test</th>
<th>Model 1</th>
<th>Model 2</th>
<th>Model 3</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dog</td>
<td>Dog</td>
<td>Cat</td>
<td>Dog</td>
</tr>
</tbody>
</table>

Majority voting
Multiple models are built on training data.

<table>
<thead>
<tr>
<th>Test</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>Dog</td>
</tr>
<tr>
<td>Model 2</td>
<td>Dog</td>
</tr>
<tr>
<td>Model 3</td>
<td>Cat</td>
</tr>
<tr>
<td>Ensemble</td>
<td></td>
</tr>
<tr>
<td>TRUE</td>
<td>Dog</td>
</tr>
</tbody>
</table>

Dog - 2, Cat - 1

**Majority voting**
Multiple models are built on training data

Training data

<table>
<thead>
<tr>
<th>Model 1</th>
<th>Dog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 2</td>
<td>Dog</td>
</tr>
<tr>
<td>Model 3</td>
<td>Cat</td>
</tr>
<tr>
<td>Ensemble</td>
<td>Dog</td>
</tr>
<tr>
<td>Test</td>
<td>Dog</td>
</tr>
</tbody>
</table>

Majority voting

Dog wins!
Training data

<table>
<thead>
<tr>
<th>Test</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>Dog</td>
</tr>
<tr>
<td>Model 2</td>
<td>Dog</td>
</tr>
<tr>
<td>Model 3</td>
<td>Cat</td>
</tr>
<tr>
<td>Ensemble</td>
<td>Dog</td>
</tr>
<tr>
<td><strong>TRUE</strong></td>
<td><strong>Dog</strong></td>
</tr>
</tbody>
</table>

Majority voting
Majority voting implies equal weight of each model’s vote.
Training data

Majority voting implies **equal weight** of each model’s vote
Are these models actually equally useful?

Majority voting implies equal weight of each model's vote.
Are these models actually equally useful?

Let’s assume that we trust one model more than the other model.
Are these models actually equally useful?

Let’s assume that we trust one model more than the other model.

What does it mean?
Training data

Representatives of each model
Ensemble parlement

Training data

Representatives of each model
Training data

Ensemble parlement

33 sits

Representatives of each model
Training data

Ensemble parlement

Representatives of each model

M1

M2

M3

0.33

0.37

0.3

37 sits
Training data

Ensemble parlement

Representatives of each model

30 sits
Training data

Representatives of each model

Ensemble parlement

Total (100 sits)
Training data

Ensemble parlement

Representatives of each model

Total (100 sits)
Training data

Each party (model) votes for the class as one team.

Ensemble parlement

Total (100 sits)
Previously each model was given only one vote.
Previously each model was given only one vote.
Previously each model was given only one vote.

M1 for Dog
M2 for Dog
M3 for Cat
Previously each model was given only one vote.

M1 for Dog
M2 for Dog
M3 for Cat

2 for Dog
1 for Cat
Training data

Previously each model was given only one vote.

M1 for Dog
M2 for Dog
M3 for Cat

0.33
0.37
0.3
Training data

Now each model has different number of votes

Previously each model was given only one vote.

Ensemble parlement

Total (100 sits)

M1 for Dog
M2 for Dog
M3 for Cat

2 for Dog
1 for Cat
We can estimate the weight of each model based on CV on training data.

Previously each model was given only one vote.

Now each model has different number of votes.

<table>
<thead>
<tr>
<th>Model</th>
<th>Vote Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>33</td>
</tr>
<tr>
<td>M2</td>
<td>2 for Dog</td>
</tr>
<tr>
<td>M3</td>
<td>1 for Cat</td>
</tr>
</tbody>
</table>
Previously each model was given only one vote.

Now each model has different number of votes:
- M1 for Dog: 33 votes
- M2 for Dog: 37 votes
- M3 for Cat: 0 votes

Ensemble parlement:
- Total (100 sits)
Previously each model was given only one vote.

Now each model has different number of votes:

- **M1** for Dog: 33
- **M2** for Dog: 37
- **M3** for Cat: 30

Ensemble parlement:

- Dog: Total 100
- Cat: 2

Training data:
Training data

Previously each model was given only one vote.

Now each model has different number of votes

<table>
<thead>
<tr>
<th>Model</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>33</td>
</tr>
<tr>
<td>M2</td>
<td>37</td>
</tr>
<tr>
<td>M3</td>
<td>30</td>
</tr>
</tbody>
</table>

Ensemble parlement

Total (100 sits)

<table>
<thead>
<tr>
<th>Class</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog</td>
<td>70</td>
</tr>
<tr>
<td>Cat</td>
<td>30</td>
</tr>
</tbody>
</table>
Previously each model was given only one vote. Now each model has different number of votes.

Training data

Ensemble parlement

Dog

Cat

Total (100 sits)

M1 for Dog
M2 for Dog
M3 for Cat

2 for Dog
1 for Cat

33 for Dog
37 for Dog
30 for Cat

70 for Dog
30 for Cat
In this case, the **outcome** (dog class wins) remains the same.
Things change if there would be **3 classes** (dog, cat, rabbit) instead of **2**

Previously each model was given only **one vote**.

Now each model has **different number of votes**.
Things change if there would be **3 classes** (dog, cat, rabbit) instead of **2**.

Previously each model was given only one vote.

Now each model has different number of votes.

<table>
<thead>
<tr>
<th></th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rabbit</td>
<td>33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dog</td>
<td>37</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cat</td>
<td>30</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Basic Ensemble

Weighted Ensemble

Training data

Ensemble parlement

Relative weight

Dog

Cat

Total (100 sits)

33 for Rabbit

37 for Dog

30 for Cat
Things change if there would be **3 classes** (dog, cat, rabbit) instead of **2**

- **Basic Ensemble**
  - M1 for Rabbit
  - M2 for Dog
  - M3 for Cat

- **Weighted Ensemble**
  - 33 for Rabbit
  - 37 for Dog
  - 30 for Cat

**Draw** (random choice)

- Dog class
  - (37 > 33 & 37 > 30)
If you know that your models are not equally useful, **how do you choose weights?**
We can estimate the weight of each model based on **CV** on training data.

If you know that your models are not equally useful, **how do you choose weights?**
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on **CV** on training data.

```
fold 1
fold 2
```

```
fold 1
fold 2
fold 3
```

```
Training data
```

```
fold 1
fold 2
fold 3
```

```
M2
```

```
M1
```

```
M3
```
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on **CV** on training data.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.

Let's choose some realistic values.
We can estimate the weight of each model based on CV on training data.

Let's choose some realistic values:

- Fold 1: 0.75
- Fold 2: 0.70
- Fold 3: 0.65

Validation data:
- Fold 1
- Fold 2
- Fold 3
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.

The weight of the model is proportional to its average score.
We can estimate the weight of each model based on CV on training data.

The weight of the model is proportional to its average score.

Average score: \[
\frac{0.75 + 0.7 + 0.65}{3}
\]
We can estimate the weight of each model based on **CV** on training data.

The weight of the model is proportional to its **average score**.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.

Training data

These are absolute values. How can we make sure they sum up to 1?
We can estimate the weight of each model based on CV on training data.

These are absolute values. How can we make sure they sum up to 1?

Total = 0.61 + 0.7 + 0.57
We can estimate the weight of each model based on CV on training data.

These are absolute values. How can we make sure they sum up to 1?

Total = 1.88
We can estimate the weight of each model based on **CV** on training data.
We can estimate the weight of each model based on CV on training data.
We can estimate the weight of each model based on CV on training data.

Training data

<table>
<thead>
<tr>
<th>Model</th>
<th>Absolute Weight</th>
<th>Relative Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>0.61</td>
<td>0.61/1.88</td>
</tr>
<tr>
<td>M2</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>M3</td>
<td>0.57</td>
<td></td>
</tr>
</tbody>
</table>

Total = 1.88
We can estimate the weight of each model based on CV on training data.

<table>
<thead>
<tr>
<th>Model</th>
<th>Absolute weight</th>
<th>Relative weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>0.61</td>
<td>0.61/1.88</td>
</tr>
<tr>
<td>M2</td>
<td>0.7</td>
<td>0.7/1.88</td>
</tr>
<tr>
<td>M3</td>
<td>0.57</td>
<td>0.57/1.88</td>
</tr>
</tbody>
</table>

Total = 1.88
We can estimate the weight of each model based on CV on training data.
Ensembles tend to yield **better results** when there is a **higher diversity** among the models

(similar reasoning is applied to enabling diversity in human teams)
How to achieve diversity of models with only one type of algorithm (e.g. decision tree)?
How to achieve diversity of models with only one type of algorithm (e.g. decision tree)?

Training data

We can play around with the data!
Training data
Training data

Random
70%
Training data

Random 70%
Training data

Random 70%
Training data

Random 70%
Training data

Bootstrapped data

Random 70%
Bootstrapping

Training data

Bootstrapped data

Random 70%
Bootstrapping

Training data

Bootstrapped data

Random 70%
Bootstrapping

Training data

Bootstrapped data

Random 70%

Training on different parts of data produces diverse models
Bootstrapping

Training data → Bootstrapped data

Random 70%
Bootstrapping
Bootstrapping

Training data

Validation data

Bootstrapped data

Random 70%
Bootstrapping

Training data

Bootstrapped data

Validation data

Random 70%

Dog

Dog

Cat

Dog
Bootstrapping

Bootstrapped data

Random 70%

Training data

Validation data

Aggregation (majority vote or averaging)

Dog

Cat
Bootstrapping + Aggregation = Bagging
Bootstrapping + Aggregation = Bagging

Bootstrapped data

Training data

Validation data

Random 70%
Bootstrapping + Aggregation = Bagging
Decision Tree Algorithm

By asking a simple question about value of the independent variable, it tries to predict a value of the dependent variable.
Decision Tree Algorithm

By asking a simple **question** about value of $X_1$ and $X_2$ it tries to predict a class $(1,0)$
Decision Tree Algorithm

By asking a simple **question** about value of $X_1$ and $X_2$ it tries to predict a class $(1,0)$
Decision Tree Algorithm

By asking a simple question about value of $X_1$ and $X_2$ it tries to predict a class $(1, 0)$.
Decision Tree Algorithm

By asking a simple question about value of $X_1$ and $X_2$ it tries to predict a class $(1, 0)$.
Let’s apply **bagging** to learn **three different** decision trees
Let’s apply **bagging** to learn **three different** decision trees.
Let’s apply **bagging** to learn **three different** decision trees.
Let’s apply **bagging** to learn **three different decision trees**
Building the **first** decision tree based on data in the **bag #1**
Building the first decision tree based on data in the bag #1

What we need to do in order to build a decision tree?
Building the **first** decision tree based on data in the **bag #1**

1. Need to **evaluate** all possible splits
Building the **first** decision tree based on data in the **bag #1**

1. Need to **evaluate** all possible splits

What are the **all possible splits**
Building the **first** decision tree based on data in the **bag #1**

1. Need to **evaluate** all possible splits

What are the **all possible splits**
Building the **first** decision tree based on data in the **bag #1**

1. Need to **evaluate** all possible splits

What are the **all possible splits**? 

Are these **are only possible splits**?
Building the first decision tree based on data in the bag #1

$X_1$ and $X_2$ are both features that we can use to divide data

1. Need to evaluate all possible splits

What are the all possible splits

Are these are only possible splits?
Building the first decision tree based on data in the bag #1

$X_1$ and $X_2$ are both features that we can use to divide data

1. Need to evaluate all possible splits

What are the all possible splits?

Are these are only possible splits?
Building the **first** decision tree based on data in the bag #1

Are **all possible splits** are also **reasonable** splits?

1. Need to **evaluate** all possible splits
Building the **first** decision tree based on data in the bag #1

1. Need to **evaluate** all possible splits

What are the **all possible splits**

Are **all possible splits** are also **reasonable** splits?

These splits are **not** reasonable (data is not divided)
Building the **first** decision tree based on data in the bag #1

1. Need to **evaluate** all possible splits

What are the **all possible splits**
Building the **first** decision tree based on data in the **bag #1**

1. Need to **evaluate** all possible splits
   
   What are the **all possible splits** (but also **reasonable**)?
Building the **first** decision tree based on data in the **bag #1**

How do we evaluate splits?

1. Need to evaluate all possible splits

What are the all possible splits (but also reasonable)?
Building the first decision tree based on data in the bag #1

How do we evaluate splits?

1. Need to evaluate all possible splits (but also reasonable)?

We can compute accuracy of the split
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

1. Need to **evaluate** all possible splits
   - What are the all possible splits (but also **reasonable**)?

How do we **evaluate** splits?
Building the **first** decision tree based on data in the bag #1

**Accuracy** = (# of guessed correctly)/(# of all)

We can compute **accuracy** of the split

How do we **evaluate** splits?

1. Need to evaluate all possible splits

What are the all possible splits (but also **reasonable**)?
Building the first decision tree based on data in the bag #1

**Accuracy** = (# of guessed correctly)/(# of all)

We can compute **accuracy** of the split

How do we evaluate splits?

1. Need to evaluate all possible splits (but also reasonable)?
Building the **first** decision tree based on data in the bag #1.

**Accuracy** = \( \frac{3}{\text{# of all}} \)

How do we **evaluate** splits?

1. Need to **evaluate** all possible splits.

What are the **all possible splits** (but also **reasonable**)?

We can compute **accuracy** of the split.
Building the first decision tree based on data in the bag #1

Accuracy = \frac{3}{4}

How do we evaluate splits?

1. Need to evaluate all possible splits (but also reasonable)

What are the all possible splits (but also reasonable)?

We can compute **accuracy** of the split
Building the first decision tree based on data in the bag #1

Accuracy = 75%

How do we evaluate splits?

1. Need to evaluate all possible splits.

What are the all possible splits (but also reasonable)?

We can compute accuracy of the split.
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

1. Need to evaluate all possible splits
   (but also **reasonable**)?

How do we **evaluate** splits?
This split is **double sided**
This split is **double sided**
This split is **double sided**

What is the **accuracy** of the mirrored split?
This split is **double sided**

What is the **accuracy** of the mirrored split?
This split is **double sided**

Let’s differentiate them by the **arrow**
Building the first decision tree based on data in the bag #1

We can compute accuracy of the split

How do we evaluate splits?

1. Need to evaluate all possible splits
   (but also reasonable)

What are the all possible splits?
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

1. Need to **evaluate** all possible splits (but also **reasonable**)?

How do we **evaluate** splits?
Building the first decision tree based on data in the bag #1

We can compute accuracy of the split

It seems a lot more splits than we thought at first…

1. Need to evaluate all possible splits (but also reasonable)?

How do we evaluate splits?

What are the all possible splits (but also reasonable)?

We can compute accuracy of the split.
Building the first decision tree based on data in the bag #1

We can compute accuracy of the split

It seems a lot more splits than we thought at first…

Worry not, accuracy of the split defines the accuracy of the reverse split

1. Need to evaluate all possible splits (but also reasonable)?

How do we evaluate splits?
Building the first decision tree based on data in the bag #1

We can compute **accuracy** of the split.

1. Need to **evaluate** all possible splits.
   - What are the **all possible splits** (but also **reasonable**)?

How do we **evaluate** splits?
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

1. Need to **evaluate** all possible splits
   - What are the all possible splits (but also **reasonable**)?

How do we **evaluate** splits?
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

1. Need to evaluate all possible splits (but also reasonable)?

How do we **evaluate** splits?
Building the first decision tree based on data in the bag #1

How do we evaluate splits?

1. Need to evaluate all possible splits

What are the all possible splits (but also reasonable)?

We can compute accuracy of the split
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

How do we **evaluate** splits?

1. Need to **evaluate** all possible splits
   (but also **reasonable**)?
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

How do we **evaluate** splits?

1. Need to **evaluate** all possible splits

   What are the **all possible splits** (but also **reasonable**)?
Building the **first** decision tree based on data in the bag #1

How do we **evaluate** splits?

1. Need to **evaluate** all possible splits
   (but also **reasonable**)?

We can compute **accuracy** of the split
Building the first decision tree based on data in the bag #1

We can compute accuracy of the split

How do we evaluate splits?

1. Need to evaluate all possible splits (but also reasonable)?

What are the all possible splits (but also reasonable)?
Building the first decision tree based on data in the bag #1

We can compute accuracy of the split

1. Need to evaluate all possible splits (but also reasonable)

How do we evaluate splits?
Building the **first** decision tree based on data in the bag #1

We can compute **accuracy** of the split

How do we **evaluate** splits?

1. Need to **evaluate** all possible splits 
   (but also reasonable)

What are the all possible splits
Building the first decision tree based on data in the bag #1

We can compute accuracy of the split

How do we evaluate splits?

1. Need to evaluate all possible splits

What are the all possible splits (but also reasonable)?
Building the first decision tree based on data in the bag #1

We can compute **accuracy** of the split

How do we **evaluate** splits?

1. Need to **evaluate** all possible splits
   (but also **reasonable**)?
Building the first decision tree based on data in the bag #1

We can compute accuracy of the split.

How do we evaluate splits?

1. Need to evaluate all possible splits (but also reasonable)?

What are the all possible splits (but also reasonable)?
1. Need to evaluate all possible splits

Building the first decision tree based on data in the bag #1

How do we evaluate splits?

What are the all possible splits (but also reasonable)?
Building the first decision tree based on data in the bag #1

1. Need to **evaluate** all possible splits
Building the first decision tree based on data in the bag #1

1. Need to **evaluate** all possible splits

2. Choose the **best** split
Building the first decision tree based on data in the bag #1

1. Need to evaluate all possible splits

2. Choose the best split
Building the **first** decision tree based on data in the bag #1

1. Need to **evaluate** all possible splits

2. Choose the **best** split

Which **splits** are we interested in?
Building the **first** decision tree based on data in the bag #1

1. Need to evaluate all possible splits

2. Choose the **best** split

**Which splits** are we interested in?
Building the first decision tree based on data in the bag #1

1. Need to evaluate all possible splits

2. Choose the best split

Which splits are we interested in?
Building the first decision tree based on data in the bag #1

Which splits are we interested in?

1. Need to evaluate all possible splits
2. Choose the best split

Which splits are we interested in?
Building the **first** decision tree based on data in the bag #1

1. Need to **evaluate** all possible splits

2. Choose the **best** split
Building the first decision tree based on data in the bag #1

Here is our tree so far:

\[ X_1 > 4.5 \]

- False
  - Class = 1
- True
  - Class = 0
Are we **done** with a tree for **bag #1**?

Here is our tree so far:

- **$X_1 > 4.5$**
  - **False** → **Class = 1**
  - **True** → **Class = 0**
Are we **done** with a tree for **bag #1**?

Here is our tree so far:

- **$X_1 > 4.5$**
  - **False** → **Class = 1**  
    - 66.6% correct
  - **True** → **Class = 0**  
    - 100% correct
Are we **done** with a tree for **bag #1**?

Here is our tree so far:

- \( X_1 > 4.5 \)
  - **False**
  - **True**

Class = 1
- 66.6% correct
Can we **improve** accuracy of this leaf?

Class = 0
- 100% correct
Let’s add another layer
Let’s add another layer

1. Need to **evaluate** all possible splits

2. Choose the **best** split
Let’s add another layer

1. Need to **evaluate** all possible splits

2. Choose the **best** split
Let’s add another layer

1. Need to **evaluate** all possible splits

2. Choose the **best** split
Let’s add another layer

1. Need to evaluate all possible splits

2. Choose the best split
Let's add another layer

1. Need to evaluate all possible splits

2. Choose the best split
Let’s add another layer

1. Need to evaluate all possible splits

2. Choose the best split
The final decision tree for **bag #1**

- $X_1 = 2.5$
- $X_1 = 4.5$

The decision tree is represented on a 2D plane with $X_1$ and $X_2$ axes. Points are indicated at specific coordinates, and the decision boundaries are marked at $X_1 = 2.5$ and $X_1 = 4.5$. The shaded area represents the decision region with a 66% probability.
The final decision tree for bag #1

Class = 1

Class = 0

The graph shows the decision tree with the following branches:
- **X₁ > 4.5**
  - False: **Class = 0**
  - True: **X₁ > 2.5**
    - False: **Class = 0**
    - True: **Class = 1**

The decision points are marked with vertical lines at **X₁ = 2.5** and **X₁ = 4.5**.
As we want to build three trees we need three independent bootstrapts.
As we want to build three trees we need three independent bootstraps.
As we want to build three trees we need three independent bootstrap samples.
As we want to build three trees we need three independent bootstrap samples.
As we want to build three trees we need three independent bootstrapts.
As we want to build three trees we need three independent bootstrapts.
As we want to build three trees we need three independent bootstrapts.
As we want to build three trees we need three independent bootstrapst.
As we want to build three trees we need three independent bootstrapts.
As we want to build three trees we need three independent bootstrap samples.
We can build the ensemble on the original data by combining trees.
We can build the ensemble on the original data by combining trees.
We can **build** the ensemble on the original data by **combining** trees.
We can **build** the **ensemble** on the original data by **combining** trees.
We can **build** the ensemble on the original data by **combining** trees.
We can **build** the **ensemble** on the original data by **combining** trees.
Perform the **majority vote** per segment to get the final decision boundaries.
Perform the **majority vote** per segment to get the final decision boundaries.
3 red vs 0 blue
3 red vs 0 blue
Perform the **majority vote** per segment to get the final decision boundaries.
We can build the ensemble on the original data.

Do the majority vote per segment to get the final decision boundaries.
Perform the **majority vote** per segment to get the final decision boundaries.
Perform the **majority vote** per segment to get the final decision boundaries.
Bag of decision trees
Let’s add a very simple **twist** to this algorithm.
As last time we shall start with generating bootstraps.
As last time we shall start with generating bootstraps.
Something will change with respect to how we build trees.
Something will change with respect to how we build trees

Hint: we will have to toss coins, again :)}
Familiar algorithm of building tree had 2 steps:
Familiar algorithm of building tree had **2 steps**:

1. Need to evaluate all possible splits
2. Choose the **best** split
Here we add a new step:

1. Need to evaluate all possible splits

2. Choose the best split
Here we add **a new step**: 

0. Choose a **random set** of features 

1. Need to evaluate all possible splits 

2. Choose the **best** split
Here we add a new step:

0. Choose a random set of features

1. Need to evaluate all possible splits

2. Choose the best split

Here we have only 2 features ($X_1$ and $X_2$), so we will choose one, but normally you would keep about 80% of the original features.
0. Choose a random set of features

1. Need to evaluate all possible splits

2. Choose the best split

Here we add a new step:

- **Heads**, means we keep $X_2$
- **Tail**, means we keep $X_1$
Here we add a new step:

0. Choose a random set of features

1. Need to evaluate all possible splits

2. Choose the best split
Here we add a new step:

0. Choose a **random set** of features

1. Need to evaluate all possible splits

2. Choose the **best** split
Here we add a new step:

0. Choose a **random** set of features

1. Need to evaluate all possible splits

2. Choose the **best** split

Bag #1

We keep $X_2$
Here we add a new step:

0. Choose a random set of features

1. Need to evaluate all possible splits

2. Choose the best split

We keep $X_2$ and get rid of $X_1$
Here we add a new step:

0. Choose a random set of features

1. Need to evaluate all possible splits

2. Choose the best split

We keep $X_2$ and get rid of $X_1$
0. Choose a **random set** of features

1. Need to evaluate all possible splits

2. Choose the **best** split
0. Choose a **random set** of features

1. Need to evaluate all possible splits

2. Choose the **best** split
0. Choose a random set of features

1. Need to evaluate all possible splits

Thanks to step 0 we have only one option left…
0. Choose a **random set** of features

1. Need to evaluate **all possible splits**

2. Choose the **best split**

---

Bag #1
0. Choose a **random set** of features

1. Need to evaluate all possible splits

2. Choose the **best split**

Both *original* and *reverse* are equally bad, so we **toss a coin** again…
0. Choose a random set of features

1. Need to evaluate all possible splits

2. Choose the best split
This is what we have built so far:

- $X_2 > 4.5$
  - False: Class = 1
  - True: Class = 0
This is what we have built so far:

In principle you may go **deeper** into each one of the leaves.

Bag #1

Class = 1

Class = 0
This is what we have built so far:

In principle you may go deeper into each one of the leaves.

Every time you would need to toss a coin to select a new random set of features from initial features.
We kept $X_2$
We kept $X_2$
We kept $X_2$.

We keep $X_1$. 

80%
We keep $X_2$
We kept $X_2$
We kept $X_2$.

We keep $X_1$. 

We keep $X_1$. 

We keep $X_1$. 

We keep $X_1$. 

80%
We kept \( X_2 \)

80%

B1

We kept \( X_1 \)

80%

B2

We kept \( X_1 \)

80%

B3
We can build an \textbf{ensemble}
We can build an ensemble.
We can build an ensemble

We kept $X_2$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$

We kept $X_1$
We kept $X_2$.

This model is **imperfect**.
This model is **imperfect**

We used only **1 layer** trees

---

66%
This model is **imperfect**

We used only **1 layer** trees

Selecting features randomly for each node, has been shown to **produce great results** in practice
We kept $X_2$

This model is **imperfect**

We used only 1 layer trees

Selecting features randomly for each node, has been shown to **produce great results** in practice (acts as **regularisation**).
The **Random Forest** algorithm

We kept $X_2$

We kept $X_1$

We kept $X_1$
The Random Forest algorithm
Form several **bags** (using bootstrapping)

The **Random Forest** algorithm
Form several bags (using bootstrapping)

For each bag build a tree

The **Random Forest** algorithm
Form several bags (using bootstrapping)

For each bag build a tree

For each node in the tree choose random set of features

The Random Forest algorithm
The Random Forest algorithm

Form several bags (using bootstrapping)

For each bag build a tree

For each node in the tree choose random set of features

Merge predictions using majority vote or averaging
The **Random Forest** algorithm

- **Form several bags (using bootstrapping)**
  - For each bag build a tree
    - For each node in the tree choose random set of features
  - Merge predictions using majority vote or averaging

Plenty of **coin tossing**!
That's all Folks!