Lecture 09: Machine learning 4

Meelis Kull
meelis.kull@ut.ee

Autumn 2021
✓ Trade-off between true positives and false positives
✓ Scoring classifiers for TP/FP trade-off
✓ Constructing ROC curves
✓ Regression
✓ Simple linear regression
✓ Multi-variate linear regression
✓ Regularization
✓ Fitting non-linear regression curves
• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Lecture 09 – Machine learning 4

• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
## Classification on Lenses dataset

<table>
<thead>
<tr>
<th>Presbyopic</th>
<th>Young</th>
<th>Spectacle prescription</th>
<th>Astigmatic</th>
<th>Tear production rate</th>
<th>Can use contact lenses</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
</tbody>
</table>
### Classification on Lenses dataset

#### Features

<table>
<thead>
<tr>
<th>No</th>
<th>Yes</th>
<th>Myope</th>
<th>Hypermetrope</th>
<th>Categorical Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
</tr>
</tbody>
</table>

#### Training data

<table>
<thead>
<tr>
<th>No</th>
<th>No</th>
<th>Hypermetrope</th>
<th>Yes</th>
<th>Reduced</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>No</td>
</tr>
</tbody>
</table>

#### Test data

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
<th>Myope</th>
<th>Yes</th>
<th>Normal</th>
<th>???</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
</tbody>
</table>
Classification

Features

Categorical Label

Training data

Test data

From earlier lectures
Method 1: Majority class classifier

Test instance: \((P,Y,H,A,R)\)

Constant \(L=0\) classifier

Prediction: \(L\)

Machine learning terminology:
This is our *predictive model.*

Predictive model = function:
- Input: vector with feature values
- Output: value of the label

Majority class classifier is a constant predictive model (constant function)
Method 4: Decision tree (features H,A,R)

Test instance: (P,Y,H,A,R) → Decision tree (see below) → Prediction: L

<table>
<thead>
<tr>
<th>Features</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>H=0,A=0,R=0</td>
<td>L=1</td>
</tr>
<tr>
<td>H=0,A=0,R=1</td>
<td>L=0</td>
</tr>
<tr>
<td>H=0,A=1,R=0</td>
<td>L=1</td>
</tr>
<tr>
<td>H=0,A=1,R=1</td>
<td>L=0</td>
</tr>
<tr>
<td>H=1,A=0,R=0</td>
<td>L=1</td>
</tr>
<tr>
<td>H=1,A=0,R=1</td>
<td>L=0</td>
</tr>
<tr>
<td>H=1,A=1,R=0</td>
<td>L=0</td>
</tr>
<tr>
<td>H=1,A=1,R=1</td>
<td>L=0</td>
</tr>
</tbody>
</table>

From earlier lectures
Classification

Features

Categorical Label

Training data

Test data

From earlier lectures
Regression

Features

Numeric Label

Training data
(labels known)

Test data
(must predict)

From earlier lectures
Example: Mtcars dataset

```r
> data(mtcars)
> mtcars

<table>
<thead>
<tr>
<th>mpg</th>
<th>cyl</th>
<th>disp</th>
<th>hp</th>
<th>drat</th>
<th>wt</th>
<th>qsec</th>
<th>vs</th>
<th>am</th>
<th>gear</th>
<th>carb</th>
</tr>
</thead>
<tbody>
<tr>
<td>21.0</td>
<td>6</td>
<td>160</td>
<td>110</td>
<td>3.90</td>
<td>2.620</td>
<td>16.46</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>21.0</td>
<td>6</td>
<td>160</td>
<td>110</td>
<td>3.90</td>
<td>2.620</td>
<td>16.46</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>22.8</td>
<td>4</td>
<td>108</td>
<td>93</td>
<td>3.85</td>
<td>2.320</td>
<td>18.61</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>21.4</td>
<td>6</td>
<td>258</td>
<td>110</td>
<td>3.08</td>
<td>3.215</td>
<td>19.44</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>18.7</td>
<td>8</td>
<td>360</td>
<td>175</td>
<td>3.15</td>
<td>3.440</td>
<td>17.02</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>18.1</td>
<td>6</td>
<td>225</td>
<td>105</td>
<td>2.76</td>
<td>3.460</td>
<td>20.22</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>14.3</td>
<td>8</td>
<td>360</td>
<td>245</td>
<td>3.21</td>
<td>3.570</td>
<td>15.84</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>16.4</td>
<td>4</td>
<td>146</td>
<td>62</td>
<td>3.69</td>
<td>3.190</td>
<td>20.00</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>22.8</td>
<td>4</td>
<td>140</td>
<td>95</td>
<td>3.92</td>
<td>3.150</td>
<td>22.90</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>19.2</td>
<td>6</td>
<td>167</td>
<td>123</td>
<td>3.92</td>
<td>3.440</td>
<td>18.30</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>17.8</td>
<td>6</td>
<td>167</td>
<td>123</td>
<td>3.92</td>
<td>3.440</td>
<td>18.90</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>16.4</td>
<td>8</td>
<td>275</td>
<td>180</td>
<td>3.07</td>
<td>4.070</td>
<td>17.40</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>17.3</td>
<td>8</td>
<td>275</td>
<td>180</td>
<td>3.07</td>
<td>3.730</td>
<td>17.60</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>15.2</td>
<td>8</td>
<td>275</td>
<td>180</td>
<td>3.07</td>
<td>3.780</td>
<td>18.00</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>10.4</td>
<td>8</td>
<td>472</td>
<td>205</td>
<td>2.93</td>
<td>5.250</td>
<td>17.98</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>10.4</td>
<td>8</td>
<td>460</td>
<td>215</td>
<td>3.00</td>
<td>5.424</td>
<td>17.82</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>14.7</td>
<td>8</td>
<td>440</td>
<td>230</td>
<td>3.23</td>
<td>5.345</td>
<td>17.42</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>32.4</td>
<td>4</td>
<td>78.7</td>
<td>66</td>
<td>4.08</td>
<td>2.200</td>
<td>19.47</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>30.4</td>
<td>4</td>
<td>75.7</td>
<td>52</td>
<td>4.93</td>
<td>1.615</td>
<td>18.52</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>33.9</td>
<td>4</td>
<td>71.1</td>
<td>65</td>
<td>4.22</td>
<td>1.835</td>
<td>19.90</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>21.5</td>
<td>4</td>
<td>120</td>
<td>97</td>
<td>3.70</td>
<td>2.465</td>
<td>20.01</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>15.5</td>
<td>8</td>
<td>318</td>
<td>150</td>
<td>2.76</td>
<td>3.520</td>
<td>16.87</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>15.2</td>
<td>8</td>
<td>304</td>
<td>150</td>
<td>3.15</td>
<td>3.435</td>
<td>17.30</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>13.3</td>
<td>8</td>
<td>350</td>
<td>245</td>
<td>3.73</td>
<td>3.840</td>
<td>15.41</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>19.2</td>
<td>8</td>
<td>400</td>
<td>175</td>
<td>3.08</td>
<td>3.845</td>
<td>17.05</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>27.3</td>
<td>4</td>
<td>79</td>
<td>66</td>
<td>4.08</td>
<td>1.935</td>
<td>18.90</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>26.0</td>
<td>4</td>
<td>120</td>
<td>91</td>
<td>4.43</td>
<td>2.140</td>
<td>16.70</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>30.4</td>
<td>4</td>
<td>95</td>
<td>113</td>
<td>3.77</td>
<td>1.513</td>
<td>16.90</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>15.8</td>
<td>8</td>
<td>351</td>
<td>264</td>
<td>4.22</td>
<td>3.170</td>
<td>14.50</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>19.7</td>
<td>6</td>
<td>145</td>
<td>175</td>
<td>3.62</td>
<td>2.770</td>
<td>15.50</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>15.0</td>
<td>8</td>
<td>301</td>
<td>335</td>
<td>3.54</td>
<td>3.570</td>
<td>14.60</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>21.4</td>
<td>4</td>
<td>121</td>
<td>109</td>
<td>4.11</td>
<td>2.780</td>
<td>18.60</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>21.4</td>
<td>6</td>
<td>160</td>
<td>110</td>
<td>3.90</td>
<td>2.620</td>
<td>16.46</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

mpg Miles/(US) gallon

cyl Number of cylinders

disp Displacement (cu.in.)

hp Gross horsepower

drat Rear axle ratio

wt Weight (1000 lbs)

qsec 1/4 mile time

vs V/S (0 = V-engine, 1 = straight engine)

am Transmission (0 = automatic, 1 = manual)

gear Number of forward gears

carb Number of carburetors

Can we predict 'mpg' from other features?
Regression

Instance: (cyl, disp, hp, ..., carb) → Regression model → Prediction: mpg
Linear regression

\[ \hat{y} = f(x; w) \]

\[ f(x; w) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_k x_k \]

**Linear regression finds the values of parameters** \( w_0, w_1, \ldots, w_k \) **such that the predictions would be as precise as possible**

\[ mpg = -127.773 + 10.298 \text{ cyl} - 0.019 \text{ disp} \]
\[ - 0.221 \text{ hp} + 24.966 \text{ drat} + 9.632 \text{ wt} \]
\[ + 0.276 \text{ qsec} + 3.839 \text{ vs} + 1.718 \text{ am} \]
\[ + 1.887 \text{ gear} - 6.138 \text{ carb} \]
Regression with deep learning

Instance
\((cyl, \text{disp}, \text{hp}, \ldots, \text{carb})\)

A deep regression model

Prediction
\(\text{mpg}\)

\[
\hat{y} = f(x; w)\\
\]

A complicated formula that is typically not even ever written down

\[
x = (x_1, x_2, \ldots, x_k)\]

The number of parameters \((m)\) is very large

Deep regression finds the values of parameters \(w_1, \ldots, w_m\) such that the predictions would be as precise as possible
Deep learning finds the values of parameters $w_1, \ldots, w_m$ such that the output would be as precise as possible.
Learning a neural network finds the values of parameters such that the output would be as precise as possible.
Deep learning vs neural networks

• There is almost no difference between the terms deep learning and artificial neural networks
  – What is the connection to biological neural networks? I will tell you later

• In deep learning, the learned neural networks are typically deep
  – I will explain soon what it means

• Sometimes shallow (i.e. non-deep) networks are used
  – In such case the term ‘deep learning’ is still sometimes used but is not well justified
Regression with deep learning

Instance

\((\text{cyl, disp, hp, \ldots, carb})\)

A deep regression model

Prediction

\(\text{mpg}\)

\(\begin{align*}
\mathbf{x} &= (x_1, x_2, \ldots, x_k) \\
\hat{y} &= f(\mathbf{x}; \mathbf{w})
\end{align*}\)

A complicated formula that is typically not even ever written down

What kind of functions should we seek to use here?
Choice of the parametric model

- In principle one could use any expression of \( w_1, \ldots, w_m, x_1, \ldots, x_k \) and mathematical operators where \( m \) (the number of parameters) is any number, not necessarily related to \( k \):

\[
\hat{y} = f(x; w) = \left( \frac{w_1 x_4 + w_2 (x_5 - 1)^2 x_6 / x_2}{w_4^{x_3-42/x_1} x_5} \right)^{w_3}
\]

- However:
  - Does it have a potential to match the true relationship between features and label?
Lecture 09 – Machine learning 4

• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Compositional models

• In reality, concepts are often compositional (consisting of many sub-components)

• E.g. dog:
  – Head
    • Eyes
    • Nose
    • Ears
    • …
  – Tail
  – 4 legs
  – …
Compositional models

• Composition is frequent in reality, so let’s use compositional models
• We should include some non-linearity into functions because linear function of linear functions is also linear
• E.g., we could use the logistic function

\[ \sigma(x) = \frac{1}{1 + e^{-x}} \]

to achieve non-linearity
Example compositional model

\[ f(x; w) = \sigma(w_1 x''_1 + w_2 x''_2 + w_3 x''_3 + w_4) \]

where:

\[ x''_1 = \sigma(w_5 x'_1 + w_6 x'_2 + w_7) \]
\[ x''_2 = \sigma(w_8 x'_1 + w_9 x'_2 + w_{10}) \]
\[ x''_3 = \sigma(w_{11} x'_1 + w_{12} x'_2 + w_{13}) \]

and:

\[ x'_1 = \sigma(w_{14} x_1 + w_{15} x_2 + w_{16}) \]
\[ x'_2 = \sigma(w_{17} x_1 + w_{18} x_2 + w_{19}) \]

Expanded form:

\[ f(x; w) = \sigma(w_1 \sigma(w_5 \sigma(w_{14} x_1 + w_{15} x_2 + w_{16}) + w_6 \sigma(w_{17} x_1 + w_{18} x_2 + w_{19}) + w_7) + w_2 \sigma(w_8 \sigma(w_{14} x_1 + w_{15} x_2 + w_{16}) + w_9 \sigma(w_{17} x_1 + w_{18} x_2 + w_{19}) + w_{10}) + w_3 \sigma(w_{11} \sigma(w_{14} x_1 + w_{15} x_2 + w_{16}) + w_{12} \sigma(w_{17} x_1 + w_{18} x_2 + w_{19}) + w_{13}) + w_4) \]
Example of a neural network

- This model is a **feed-forward neural network**

\[
f(x; w) = \sigma(w_1 \sigma(w_5 \sigma(w_{14} x_1 + w_{15} x_2 + w_6) + w_6 \sigma(w_{17} x_1 + w_{18} x_2 + w_9) + w_7)
+ w_2 \sigma(w_8 \sigma(w_{14} x_1 + w_{15} x_2 + w_6) + w_9 \sigma(w_{17} x_1 + w_{18} x_2 + w_9) + w_10)
+ w_3 \sigma(w_{11} \sigma(w_{14} x_1 + w_{15} x_2 + w_6) + w_{12} \sigma(w_{17} x_1 + w_{18} x_2 + w_9) + w_{13} + w_4)
\]

![Diagram of a feed-forward neural network](diagram.png)
• Deep learning
  • What is deep learning?
  • Compositional models
  • **Feed-forward neural network**
    • Learning a feed-forward neural network
    • Example task: learn a parabola
    • Back-propagation algorithm
    • From regression to classification
    • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Feed-forward neural network

- Consists of many computational units (artificial neurons)
  - Each unit has many inputs and single output
  - Output is calculated by applying a non-linear activation function on a linear combination of inputs

- Neurons are wired into a graph (neural network)

- There are no directed cycles in the graph (feed-forward neural network)
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas Immanent in Nervous Activity
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron

Rosenblatt (1957)
Perceptron

McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron
Rosenblatt (1957)
Perceptron

New York Times [1958]: “(The perceptron) is the embryo of an electronic computer that is expected to be able to walk, talk, see, write, reproduce itself and be conscious of its existence”
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron

Minsky & Papert (1969)
Perceptrons: an introduction to computational geometry
McCulloch & Pitts (1943)
A Logical Calculus of the Ideas
Immanent in Nervous Activity

Rosenblatt (1957)
Perceptron

Minsky & Papert (1969)
Perceptrons: an introduction to computational geometry

Rumelhart, Hinton & Williams (1986)
Learning representations by back-propagating errors
Multi-layer perceptron

• **Multi-layer perceptron (MLP)** can be considered a synonym for feedforward neural networks (according to the Deep learning book)

• Neurons in MLP can be organized into layers based on their connections
  – Input layer: contains feature values
  – Output layer: produces model output
  – Hidden layers: internal constructed features
Layers in our example network?

- This model is a **feed-forward neural network**

\[ f(x; w) = \sigma(w_1 \sigma(w_5 \sigma(w_{14} x_1 + w_{15} x_2 + w_{16}) + w_6 \sigma(w_{17} x_1 + w_{18} x_2 + w_{19}) + w_7) \\
+ w_2 \sigma(w_8 \sigma(w_{14} x_1 + w_{15} x_2 + w_{16}) + w_9 \sigma(w_{17} x_1 + w_{18} x_2 + w_{19}) + w_{10}) \\
+ w_3 \sigma(w_{11} \sigma(w_{14} x_1 + w_{15} x_2 + w_{16}) + w_{12} \sigma(w_{17} x_1 + w_{18} x_2 + w_{19}) + w_{13}) + w_4) \]
Layers in our example network

Our model is an example of a **feed-forward neural network** with 2 hidden layers and sigmoid activation functions.

Input layer

Hidden layer

Hidden layer

Output layer
Deep learning

• Deep learning stands for learning algorithms involving neural networks with several hidden layers (deep)

• How deep is deep?
  – At least 2 hidden layers
  – Usually 2-10, sometimes up to 100 or more

• Hidden layers can be thought of as automatically constructed features
Features constructed by a neural net

Image source: https://cs50x.mprog.nl/lectures/week-7
Origin: Lee et al, ICML 2009
What is deep learning?

Many layers of adaptive non-linear processing to model complex relationships among data

Space 1  →  Space 2
What is deep learning?

Many layers of adaptive non-linear processing to model complex relationships among data

Space 1

Space 2

0,1,2,3,…9
What is deep learning?

Many layers of adaptive non-linear processing to model complex relationships among data

Space 1 → species → Space 2
Many layers of adaptive non-linear processing to model complex relationships among data.

Space 1

"We love you"

Space 2
Evolution of ML methods

Rule-based

Input

Fixed set of rules

Output

Adopted from Y. Bengio [http://videolectures.net/deeplearning2015_bengio_theoretical_motivations/]
Evolution of ML methods

Rule-based

| Input | Fixed set of rules | Output |

Classic Machine Learning

| Input | Hand designed features | Learning | Output |

Evolution of ML methods

Rule-based

| Input | Fixed set of rules | Output |

Classic Machine Learning

| Input | Hand designed features | Learning | Output |

Representation Learning

| Input | Automated feature extraction | Learning | Output |
Evolution of ML methods

Rule-based  Classic Machine Learning  Representation Learning  (end-to-end) Deep Learning

Input → Fixed set of rules → Output

Input → Hand designed features → Learning → Output

Input → Automated feature extraction → Learning → Output

Input → Low level features → Learning → High level features → Output

Adopted from Y. Bengio [http://videolectures.net/deeplearning2015_bengio_theoretical_motivations/]
How many hidden layers does this feedforward network have:

A. 0
B. 1
C. 2
D. 3
E. 4
F. None of the above
G. I don’t know

Response Counter

1 1 1 1 1 1 1
Lecture 09 – Machine learning 4

• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • **Learning a feed-forward neural network**
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Learning a neural network

A neural network

Instance → A neural network → Output

\[ x = (x_1, x_2, \ldots, x_k) \]

\[ \hat{y} = f(x; w) \]

Learning a neural network finds the values of parameters \( w_1, \ldots, w_m \) such that the output would be as precise as possible.

The number of parameters \( m \) is very large.

A complicated formula that is typically not even ever written down.
Learning a neural network

\[ \hat{y} = f(x; w) \]

But how can we find good values for the parameters?

Learning a neural network finds the values of parameters \( w_1, \ldots, w_m \) such that the output would be as precise as possible.

\[ x = (x_1, x_2, \ldots, x_k) \]

A complicated formula that is typically not even ever written down
Simple linear regression with ordinary least squares

We search for a function \( \hat{y} = f(x) \)

which minimizes mean squared error (MSE):

\[
\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 = \frac{1}{N} \sum_{i=1}^{N} (y_i - \beta x_i - \alpha)^2
\]

which means to find derivatives wrt \( \alpha \) and \( \beta \)

and solve the system of equations:

\[
\begin{align*}
\frac{\partial \text{MSE}}{\partial \alpha} &= 0 \\
\frac{\partial \text{MSE}}{\partial \beta} &= 0
\end{align*}
\]

Adapted from slides by Anna Leontjeva
Simple linear regression with ordinary least squares

\[
\begin{aligned}
\frac{\partial \text{MSE}}{\partial \alpha} &= 0 \\
\frac{\partial \text{MSE}}{\partial \beta} &= 0
\end{aligned}
\quad \Leftrightarrow 
\begin{aligned}
\alpha &= \bar{y} - \beta \bar{x} \\
\beta &= \frac{\sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N} (x_i - \bar{x})^2}
\end{aligned}
\]

where

\[
\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i \\
\bar{y} = \frac{1}{N} \sum_{i=1}^{N} y_i
\]

Adapted from slides by Anna Leontjeva
Multivariate linear regression

\[
\frac{\partial \text{MSE}}{\partial \beta} = -2y^T X + 2X^T X \beta
\]

\[
\frac{\partial \text{MSE}}{\partial \beta} = 0 \Rightarrow \beta = (X^T X)^{-1} X^T y
\]

complexity of matrix inverse is high: \(O(n^{2.373})\)
in practice iterative methods are used (e.g. gradient descent)

Adapted from slides by Anna Leontjeva
How to learn parameter values?

• Analytically solving the system of equations for the gradient to be zero is too hard for deep networks

• Let us solve it numerically, using gradient descent
  – I will explain soon
Multivariate linear regression

• Given:
  – Parametric model: \( \hat{y} = f(x; w) = w_0 + w_1 x_1 + \cdots + w_d x_d \)
  – Loss function: \( \ell(\hat{y}, y) = (\hat{y} - y)^2 \)
  – Training data: \((x_1, y_1), \ldots, (x_n, y_n)\)

• Denote:
  – Loss on instance \(i\): \( L_i(w) = \ell(f(x_i; w), y_i) \)

• Task:
  – Minimise average loss: \( \arg\min_w \frac{1}{n} \sum_{i=1}^{n} L_i(w) \)

• Typical method:
  – Stochastic gradient descent: \( w \leftarrow w - \eta \nabla L_i(w) \)
Neural network regression

• Given:
  – Parametric model: \( \hat{y} = f(x; w) \)
  – Loss function: \( \ell(\hat{y}, y) = (\hat{y} - y)^2 \)
  – Training data: \((x_1, y_1), \ldots, (x_n, y_n)\)

• Denote:
  – Loss on instance i: \( L_i(w) = \ell(f(x_i; w), y_i) \)

• Task:
  – Minimise average loss: \( \arg\min_w \frac{1}{n} \sum_{i=1}^{n} L_i(w) \)

• Typical method:
  – Stochastic gradient descent: \( w \leftarrow w - \eta \nabla L_i(w) \)
Neural networks

• Given:
  – Parametric model: $\hat{y} = f(x; w)$
  – Loss function: $\ell(\hat{y}, y)$
  – Training data: $(x_1, y_1), \ldots, (x_n, y_n)$

• Denote:
  – Loss on instance $i$: $L_i(w) = \ell(f(x_i; w), y_i)$

• Task:
  – Minimise average loss: $\arg\min_w \frac{1}{n} \sum_{i=1}^{n} L_i(w)$

• Typical method:
  – Stochastic gradient descent: $w \leftarrow w - \eta \nabla L_i(w)$
Deep learning

• Given:
  – Parametric model: \( \hat{y} = f(x; w) \)
  – Loss function: \( \ell(\hat{y}, y) \)
  – Training data: \((x_1, y_1), \ldots, (x_n, y_n)\)

• Denote:
  – Loss on instance \( i \): \( L_i(w) = \ell(f(x_i; w), y_i) \)

• Task:
  – Minimise average loss: \( \arg\min_w \frac{1}{n} \sum_{i=1}^{n} L_i(w) \)

• Typical method:
  – Stochastic gradient descent: \( w \leftarrow w - \eta \nabla L_i(w) \)
Learning algorithm

• Initialize the weights randomly
• Find a way to modify the weights by a tiny amount such that the predictions would get slightly better
• Keep doing it until the model is good
Gradient descent

• A general approach to minimization tasks where gradients can be calculated

• Task: minimize the loss $L(w)$

• Gradient descent algorithm:
  – Initialize $w$ (for example, randomly)
  – Repeat until convergence
    • $w \leftarrow w - \eta \nabla L(w)$

• Here $\eta > 0$ is the learning rate
Stochastic gradient descent (SGD)

- Suppose the function that is to be minimized can be written as a sum:

\[
L(w) = \sum_{i=1}^{n} L_i(w)
\]

- Stochastic gradient descent algorithm:
  - Initialize \( w \) (for example, randomly)
  - Repeat until convergence:
    - Randomly shuffle the indices \( 1, \ldots, n \)
    - For each index \( i \)
      \[
      w \leftarrow w - \eta \nabla L_i(w)
      \]
How to calculate the gradient?

• Gradient of the loss on one instance – how to calculate?

\[ \nabla \ell (\sigma (w_1 \sigma (w_5 \sigma (w_{14} x_1 + w_{15} x_2 + w_{16}) + w_6 \sigma (w_{17} x_1 + w_{18} x_2 + w_{19}) + w_7) \\
+ w_2 \sigma (w_8 \sigma (w_{14} x_1 + w_{15} x_2 + w_{16}) + w_9 \sigma (w_{17} x_1 + w_{18} x_2 + w_{19}) + w_{10}) \\
+ w_3 \sigma (w_{11} \sigma (w_{14} x_1 + w_{15} x_2 + w_{16}) + w_{12} \sigma (w_{17} x_1 + w_{18} x_2 + w_{19}) + w_{13}) + w_4), y) \\
= ??? \]

• Could do straightforward differentiation, but there are multiple shared expressions that should be exploited to speed it up

• Method: \textbf{back-propagation algorithm}

• Before explaining it, let us see an example
• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Example task: learn a parabola

• Task:
  – learn to predict \( y \) from \( x \) where \( y = x^2 \)

• Training data:
  – 1000 labelled instances \((x, x^2)\) where \( x \) is uniformly randomly drawn from \([-1, +1]\)
Network architecture

- Let us use MLP with:
  - 1 input neuron, 2 hidden layers, 1 output neuron
  - 2 neurons in each hidden layer
  - logistic activation in hidden layers
  - linear output neuron
Function approximation

• We want to learn to approximate $y = x^2$

• By this network:

• That is, by this function:

$$\hat{y} = w_{11} \sigma(w_5 \sigma(w_1 x + w_2) + w_6 \sigma(w_3 x + w_4) + w_7) + w_{12} \sigma(w_8 \sigma(w_1 x + w_2) + w_9 \sigma(w_3 x + w_4) + w_{10}) + w_{13}$$
Is it possible?

• Impossible exactly, possible approximately

• This (red) function has been obtained by using weights $w_1, \ldots, w_{13}$ equal to:

$3.07626963, 2.45446324, 3.06085944, -2.49138331, -2.56511831, 2.44493103, 0.64800489, 1.65763772, -2.29413271, -0.35358864, 3.14166284, -0.82448912, 0.04504132$
Extrapolating also?

• Is the fit as good outside of range \([-1,+1]\)?
Extrapolating also?

• Is the fit as good outside of range \([-1, +1]\) ?
• No, the model cannot extrapolate well!

Therefore, if we need predictions in a wider range, then need to extend training data also.
How to learn the weights?

- It is a parametric regression task
- We can use (stochastic) gradient descent on parameters to minimize mean squared error (MSE)
Parametric regression

- **Given:**
  - Parametric model: \( \hat{y} = f(x; w) \)
  - Loss function: \( \ell(\hat{y}, y) = (\hat{y} - y)^2 \)
  - Training data: \((x_1, y_1), \ldots, (x_n, y_n)\)

- **Denote:**
  - Loss on instance \(i\): \(L_i(w) = \ell(f(x_i; w), y_i)\)

- **Task:**
  - Minimise average loss: \(\arg\min_w \frac{1}{n} \sum_{i=1}^{n} L_i(w)\)

- **Method:**
  - Stochastic gradient descent: \(w \leftarrow w - \eta \nabla L_i(w)\)
How to calculate the gradient?

• Gradient of the loss on one instance – how to calculate?

\[ \nabla \ell (w_{11}\sigma (w_5\sigma (w_1x + w_2) + w_6\sigma (w_3x + w_4) + w_7) \\
+ w_{12}\sigma (w_8\sigma (w_1x + w_2) + w_9\sigma (w_3x + w_4) + w_{10}) + w_{13}, y) \]
\[ = ??? \]

• Let us first introduce notation for its sub-expressions
\[ \hat{y} = w_{11} \sigma(w_{5} \sigma(w_{1} x + w_{2}) + w_{6} \sigma(w_{3} x + w_{4}) + w_{7}) + w_{12} \sigma(w_{8} \sigma(w_{1} x + w_{2}) + w_{9} \sigma(w_{3} x + w_{4}) + w_{10}) + w_{13} \]

\[ s_1 = w_1 x + w_2 \]
\[ s_2 = w_3 x + w_4 \]
\[ s_3 = w_5 a_1 + w_6 a_2 + w_7 \]
\[ s_4 = w_8 a_1 + w_9 a_2 + w_{10} \]
\[ a_1 = \sigma(s_1) \]
\[ a_2 = \sigma(s_2) \]
\[ a_3 = \sigma(s_3) \]
\[ a_4 = \sigma(s_4) \]
Gradient?

\[
\begin{align*}
    s_1 &= w_1 x + w_2 \\
    s_2 &= w_3 x + w_4 \\
    a_1 &= \sigma(s_1) \\
    a_2 &= \sigma(s_2) \\
    s_3 &= w_5 a_1 + w_6 a_2 + w_7 \\
    s_4 &= w_8 a_1 + w_9 a_2 + w_{10} \\
    a_3 &= \sigma(s_3) \\
    a_4 &= \sigma(s_4) \\
    \hat{y} &= w_{11} a_3 + w_{12} a_4 + w_{13} \\
\end{align*}
\]

For SGD we need to calculate the gradient of MSE:

\[
\nabla MSE = \left( \frac{\partial MSE}{\partial w_1}, \ldots, \frac{\partial MSE}{\partial w_{13}} \right) \\
\frac{\partial MSE}{\partial w_i} = \frac{\partial (\hat{y} - y)^2}{\partial w_i} = ???
\]
Chain-rule!

\[ s_1 = w_1 x + w_2 \]
\[ s_2 = w_3 x + w_4 \]
\[ s_3 = w_5 a_1 + w_6 a_2 + w_7 \]
\[ s_4 = w_8 a_1 + w_9 a_2 + w_{10} \]
\[ a_1 = \sigma(s_1) \]
\[ a_2 = \sigma(s_2) \]
\[ a_3 = \sigma(s_3) \]
\[ a_4 = \sigma(s_4) \]
\[ \hat{y} = w_{11} a_3 + w_{12} a_4 + w_{13} \]

\[
\frac{\partial \text{MSE}}{\partial w_{13}} = \frac{\partial (\hat{y} - y)^2}{\partial w_{13}} = \frac{\partial (\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial (\hat{y} - y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{13}} = 2(\hat{y} - y) \cdot 1 \cdot 1 = 2(\hat{y} - y)
\]
Chain-rule!

\[
\begin{align*}
s_1 &= w_1 x + w_2 \\
s_2 &= w_3 x + w_4 \\
a_1 &= \sigma(s_1) \\
a_2 &= \sigma(s_2) \\
s_3 &= w_5 a_1 + w_6 a_2 + w_7 \\
s_4 &= w_8 a_1 + w_9 a_2 + w_{10} \\
a_3 &= \sigma(s_3) \\
a_4 &= \sigma(s_4) \\
\hat{y} &= w_{11} a_3 + w_{12} a_4 + w_{13}
\end{align*}
\]

\[
\frac{\partial \text{MSE}}{\partial w_{12}} = \frac{\partial (\hat{y} - y)^2}{\partial w_{12}} = \frac{\partial (\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial (\hat{y} - y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{12}} = 2(\hat{y} - y) \cdot 1 \cdot a_4 = 2(\hat{y} - y)a_4
\]
Chain-rule!

\[ s_1 = w_1 x + w_2 \quad s_3 = w_5 a_1 + w_6 a_2 + w_7 \quad \hat{y} = w_{11} a_3 + w_{12} a_4 + w_{13} \]
\[ s_2 = w_3 x + w_4 \quad s_4 = w_8 a_1 + w_9 a_2 + w_{10} \]
\[ a_1 = \sigma(s_1) \quad a_3 = \sigma(s_3) \]
\[ a_2 = \sigma(s_2) \quad a_4 = \sigma(s_4) \]

**Derivative of the logistic activation:**

\[ \frac{\partial \sigma(x)}{\partial x} = \sigma(x)(1 - \sigma(x)) \quad \Rightarrow \quad \frac{\partial a_i}{\partial s_i} = a_i(1 - a_i) \]

\[
\frac{\partial MSE}{\partial w_5} = \frac{\partial(\hat{y} - y)^2}{\partial w_5} = \frac{\partial(\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial(\hat{y} - y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a_3} \frac{\partial a_3}{\partial s_3} \frac{\partial s_3}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_5} = 2(\hat{y} - y) \cdot 1 \cdot w_{11} \cdot a_3(1 - a_3) \cdot a_1
\]
Chain-rule!

\[ s_1 = w_1 x + w_2 \quad s_3 = w_5 a_1 + w_6 a_2 + w_7 \quad \hat{y} = w_{11} a_3 + w_{12} a_4 + w_{13} \]

\[ s_2 = w_3 x + w_4 \quad s_4 = w_8 a_1 + w_9 a_2 + w_{10} \]

\[ a_1 = \sigma(s_1) \quad a_3 = \sigma(s_3) \]

\[ a_2 = \sigma(s_2) \quad a_4 = \sigma(s_4) \]

\[
\frac{\partial \text{MSE}}{\partial w_1} = \frac{\partial (\hat{y} - y)^2}{\partial w_1} = \frac{\partial (\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial (\hat{y} - y)}{\partial \hat{y}} \left( \frac{\partial \hat{y}}{\partial a_3} \frac{\partial a_3}{\partial s_3} \frac{\partial s_3}{\partial a_1} + \frac{\partial \hat{y}}{\partial a_4} \frac{\partial a_4}{\partial s_4} \frac{\partial s_4}{\partial a_1} \right) \frac{\partial a_1}{\partial s_1} \frac{\partial s_1}{\partial w_1}
\]
How can we do less computation?

\[
\frac{\partial \text{MSE}}{\partial w_{13}} = \frac{\partial (\hat{y} - y)^2}{\partial w_{13}} = \frac{\partial (\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial (\hat{y} - y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{13}} = 2(\hat{y} - y) \cdot 1 \cdot 1 = 2(\hat{y} - y)
\]

\[
\frac{\partial \text{MSE}}{\partial w_{12}} = \frac{\partial (\hat{y} - y)^2}{\partial w_{12}} = \frac{\partial (\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial (\hat{y} - y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{12}} = 2(\hat{y} - y) \cdot 1 \cdot a_4 = 2(\hat{y} - y) a_4
\]

\[
\ldots
\]

\[
\frac{\partial \text{MSE}}{\partial w_5} = \frac{\partial (\hat{y} - y)^2}{\partial w_5} = \frac{\partial (\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial (\hat{y} - y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a_3} \frac{\partial a_3}{\partial s_3} \frac{\partial s_3}{\partial w_5} = 2(\hat{y} - y) \cdot 1 \cdot w_{11} \cdot a_3(1 - a_3) \cdot a_1
\]

\[
\ldots
\]

\[
\frac{\partial \text{MSE}}{\partial w_1} = \frac{\partial (\hat{y} - y)^2}{\partial w_1} = \frac{\partial (\hat{y} - y)^2}{\partial (\hat{y} - y)} \frac{\partial (\hat{y} - y)}{\partial \hat{y}} \left( \frac{\partial \hat{y}}{\partial a_3} \frac{\partial a_3}{\partial s_3} \frac{\partial s_3}{\partial a_1} + \frac{\partial \hat{y}}{\partial a_4} \frac{\partial a_4}{\partial s_4} \frac{\partial s_4}{\partial a_1} \right) \frac{\partial a_1}{\partial s_1} \frac{\partial s_1}{\partial w_1}
\]

- Let us store intermediate results!
Storing intermediate results

\[
\frac{\partial \text{MSE}}{\partial w_1} = \frac{\partial \text{MSE}}{\partial s_1} \frac{\partial s_1}{\partial w_1}, \quad \ldots
\]

\[
\frac{\partial \text{MSE}}{\partial w_5} = \frac{\partial \text{MSE}}{\partial s_3} \frac{\partial s_3}{\partial w_5}, \quad \ldots
\]

\[
\frac{\partial \text{MSE}}{\partial w_{13}} = \frac{\partial \text{MSE}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{13}}, \quad \frac{\partial \text{MSE}}{\partial s_1}, \quad \frac{\partial \text{MSE}}{\partial s_3}, \quad \frac{\partial \text{MSE}}{\partial \hat{y}}
\]
Storing intermediate results

\[
\frac{\partial \text{MSE}}{\partial w_1} = \frac{\partial \text{MSE}}{\partial s_1} \quad s_1
\]

\[
\frac{\partial \text{MSE}}{\partial w_5} = \frac{\partial \text{MSE}}{\partial s_3} \frac{\partial s_3}{\partial w_5} \quad s_3
\]

\[
\frac{\partial \text{MSE}}{\partial w_{13}} = \frac{\partial \text{MSE}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{13}} \quad \hat{y}
\]

This method is known as back-propagation.
Lecture 09 – Machine learning 4

• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Back-propagation algorithm

- Back-propagation algorithm is also known as **automatic differentiation in reverse accumulation mode**
- First calculates the gradient of loss with respect to output layer
- Then uses the obtained gradient to calculate gradient in the last hidden layer
- Then uses that gradient to obtain one before
- Etc
- Gradient over all weights is obtained from these
Training with back-propagation

• 3 phases:
  – Forward pass (obtaining the prediction \( \hat{y} \))
  – Backward pass (calculating the gradient)
  – Weight update (multiplying the gradient by learning rate and subtracting from old weights)
Example

- Let us work through the same example, with particular initial weights:

\[ \hat{y} = 3\sigma (-3\sigma (3x + 2) + 2\sigma (3x - 2) + 1) \]

\[ - \sigma (2\sigma (3x + 2) - 2\sigma (3x - 2)) \]
Example: Forward pass

Let us work through the same example, with particular initial weights:

\[ \hat{y} = 0.09 \]

\[ a_1 = 0.97 \]

\[ a_3 = 0.24 \]

\[ x = 0.5 \]

\[ \hat{y} = -0.05 \]

\[ s_1 = 3.5 \]

\[ s_3 = -1.16 \]

\[ s_2 = -0.5 \]

\[ s_4 = 1.19 \]

\[ a_2 = 0.38 \]

\[ a_4 = 0.77 \]
Example: Backward pass

- Let us work through the same example, with particular initial weights:

\[
\begin{align*}
\frac{\partial \text{MSE}}{\partial w_1} &= 0.02 \\
\frac{\partial \text{MSE}}{\partial w_5} &= -0.32 \\
\frac{\partial \text{MSE}}{\partial w_{13}} &= -0.60 \\
\frac{\partial \text{MSE}}{\partial s_1} &= 0.03 \\
\frac{\partial \text{MSE}}{\partial s_3} &= -0.33 \\
\frac{\partial \text{MSE}}{\partial s_2} &= -0.20 \\
\frac{\partial \text{MSE}}{\partial s_4} &= 0.11 \\
\frac{\partial \text{MSE}}{\partial \hat{y}} &= -0.60
\end{align*}
\]

Training instance:

- \(x=0.5\)
- \(y=x^2=0.25\)
Example: Weight update

- Let us work through the same example, with particular initial weights:

Training instance:
\[ x = 0.5 \]
\[ y = x^2 = 0.25 \]

\[
\frac{\partial \text{MSE}}{\partial w_1} = 0.02
\]
\[ \ldots \]
\[
\frac{\partial \text{MSE}}{\partial w_5} = -0.32
\]
\[ \ldots \]
\[
\frac{\partial \text{MSE}}{\partial w_{13}} = -0.60
\]

In this example using learning rate \( \eta = 1 \)
Example: Weight update

• Let us work through the same example, with particular initial weights:

\[
\begin{align*}
\hat{y} &= w_1 a_1 + w_2 a_1 + w_3 a_1 + w_4 a_2 + w_5 a_2 + w_6 a_3 + w_7 a_3 + w_8 a_3 + w_9 a_4 + w_{10} a_4 + w_{11}\,
\end{align*}
\]

Training instance:  
\(x = 0.5\)  
\(y = x^2 = 0.25\)

\[
\begin{align*}
\frac{\partial \text{MSE}}{\partial w_1} &= 0.02 \\
\cdots \\
\frac{\partial \text{MSE}}{\partial w_5} &= -0.32 \\
\cdots \\
\frac{\partial \text{MSE}}{\partial w_{13}} &= -0.60
\end{align*}
\]

In this example using learning rate \(\eta = 1\)
Do we have a better model now?

- Legend:
  - Actual target value (blue)
  - Old model (red)
  - New model (green)

- We have over-compensated for the error, now even worse to the opposite direction

- Reason:
  - Too high learning rate
The same with learning rate 0.1

- **Legend:**
  - Actual target value (blue)
  - Old model (red)
  - New model (green)

- Now the model has become better in range $[-0.5, +1]$ and worse in $[-1, -0.5]$
Deep learning
- What is deep learning?
- Compositional models
- Feed-forward neural network
- Learning a feed-forward neural network
- Example task: learn a parabola
- Back-propagation algorithm
- From regression to classification
- Convolutional neural network

Overview of the machine learning landscape
- Machine learning objectives
- Machine learning tasks and approaches
- Learning theory
- Success stories in machine learning
From regression to classification

• This was for regression

• We can also do classification:
  – Use softmax as the output layer’s activation
    • Next slide explains
  – Use one-hot vectors to represent the true labels
  – Use the categorical cross-entropy loss to be minimized
    • See next slides
Softmax

• Argmax does not provide probabilities and is hard to train in back-propagation
• Instead it is possible to use **softmax** - a generalization of the logistic function for $K$ classes:

$$\sigma : \mathbb{R}^K \rightarrow (0, 1)^K$$

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$
Cross-entropy for multiple classes

- Cross-entropy for two classes:
  \[
  CE = -y \log \hat{p}(x) - (1 - y) \log (1 - \hat{p}(x))
  \]

- Cross-entropy for multiple classes:
  \[
  CE = -\sum_{j=1}^{K} y_j \log \hat{p}_j(x)
  \]

where \( y_j \) is a binary indicator of whether instance belongs to class \( j \) and \( \hat{p}_j(x) \) is the predicted probability to belong to class \( j \).
Training Neural Networks

http://playground.tensorflow.org/
• **Deep learning**
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • **Convolutional neural network**

• **Overview of the machine learning landscape**
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
1.2 million images

1000 categories
Errors

2010 2011

28% 26%
AlexNet (A. Krizhevsky et al. 2012)
Meelis Kull - Autumn 2021 - LTAT.02.002 – Intro to Data Science - Lecture 09

Errors


28% 26% 16% 12% 7% 3% <3%

AlexNet (A. Krizhevsky et al. 2012)

http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/
Hypothetical super-dedicated fine-grained expert ensemble of human labelers

AlexNet (A. Krizhevsky et al. 2012)
Let’s consider the following image
Convolutional Neural Network

Let's consider the following image
Convolutional Neural Network

Convolutional layer works as a filter applied to the original image.
Convolutional Neural Network

Convolutional layer works as a filter applied to the original image

There are many filters in the convolutional layer, they detect different patterns
Convolutional Neural Network

Each filter applied to all possible 2x2 patches of the original image produces one output value.
Convolutional Neural Network

Each filter applied to all possible 2x2 patches of the original image produces one output value.
Convolutional Neural Network

Each filter applied to all possible 2x2 patches of the original image produces one output value.
Convolutional Neural Network

Each filter applied to all possible 2x2 patches of the original image produces one output value.
Convolutional Neural Network

Each filter applied to all possible 2x2 patches of the original image produces one output value.

Repeat this process for all filters in this layer.
Convolutional Neural Network

Each filter applied to all possible 2x2 patches of the original image produces one output value.

Repeat this process for all filters in this layer and the next.
The output of the last convolutional layer is flattened into a single vector (like we did with images)
This vector is fed into **fully connected layer** with as many neurons as **possible classes**.

The output of **the last** convolutional layer is flattened into a **single vector** (like we did with images).
Convolutional Neural Network

This vector is fed into **fully connected layer** with as many neurons as **possible classes**

The output of the **last** convolutional layer is flattened into a **single vector** (like we did with images)

Each neuron outputs probabilities
Training Neural Networks (part III)

http://scs.ryerson.ca/~aharley/vis/conv/
Keras

High-level wrapper on top of Theano or TensorFlow

In practice, it is a module for Python that lets you describe your network

```python
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout

model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```
• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Lecture 09 – Machine learning 4

• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
Main goal in machine learning

• GOAL:
  Find a (predictive) model which fits the data
  – Define fitness (evaluation measure)
  – How to search (optimization method)

• Optimization has a key role in machine learning
  – Gradient descent optimization methods are underlying many learning algorithms
Learning objectives (1)

- **Accuracy**
  - Small prediction errors
- **Time-efficiency**
  - Low training time, low prediction time
- **Memory-efficiency**
  - Low memory usage for training, predicting
- **Data-efficiency**
  - Few training data required
- **Imbalance-tolerance**
  - Tolerates imbalanced classes
- **Noise-tolerance**
  - Tolerates noise in features and labels
Learning objectives (2)

• Interpretability
  – Humans can understand why a particular prediction was made

• Fairness
  – Not discriminating for gender/race etc

• Generality
  – Applicable across several contexts

• Incrementality
  – Can be easily updated after small changes in data

• Calibratedness
  – Not over-confident and not under-confident
Computational aspects

• Train time vs test time:
  – Some algorithms are slow to train, fast to test (e.g. decision trees, random forests)
  – Some algorithms are fast to train, slow to test (e.g. K-nearest neighbours)

• Parallelizable vs not:
  – Some algorithms are easily parallelized (e.g. independently learn trees in a random forest)
  – Some not (e.g. SVMs)

• Small-scale vs large-scale:
  – Small-scale learning performed on CPUs
  – Large-scale learning usually performed on GPUs
Lecture 09 – Machine learning 4

- Deep learning
  - What is deep learning?
  - Compositional models
  - Feed-forward neural network
  - Learning a feed-forward neural network
  - Example task: learn a parabola
  - Back-propagation algorithm
  - From regression to classification
  - Convolutional neural network

- Overview of the machine learning landscape
  - Machine learning objectives
  - **Machine learning tasks and approaches**
  - Learning theory
  - Success stories in machine learning
### Classification on Lenses dataset

<table>
<thead>
<tr>
<th>Presbyopic</th>
<th>Young</th>
<th>Spectacle prescription</th>
<th>Astigmatic</th>
<th>Tear production rate</th>
<th>Can use contact lenses</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>???</td>
</tr>
</tbody>
</table>
### Classification on Lenses dataset

<table>
<thead>
<tr>
<th>Features</th>
<th>Categorical Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Yes Myope</td>
<td>No Normal</td>
</tr>
<tr>
<td>No Yes Myope</td>
<td>Yes Reduced</td>
</tr>
<tr>
<td>No Yes Hypermetrope</td>
<td>No Reduced</td>
</tr>
<tr>
<td>No Yes Hypermetrope</td>
<td>No Normal</td>
</tr>
</tbody>
</table>

#### Training data

<table>
<thead>
<tr>
<th>Features</th>
<th>Categorical Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>No No Hypermetrope</td>
<td>Yes Reduced</td>
</tr>
<tr>
<td>No No Hypermetrope</td>
<td>Yes Normal</td>
</tr>
<tr>
<td>Yes No Myope</td>
<td>No Normal</td>
</tr>
<tr>
<td>Yes No Hypermetrope</td>
<td>Yes Normal</td>
</tr>
</tbody>
</table>

#### Test data

<table>
<thead>
<tr>
<th>Features</th>
<th>Categorical Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes No Myope</td>
<td>Yes Normal</td>
</tr>
<tr>
<td>Yes No Hypermetrope</td>
<td>Yes Reduced</td>
</tr>
<tr>
<td>Yes No Hypermetrope</td>
<td>No Normal</td>
</tr>
<tr>
<td>Yes No Hypermetrope</td>
<td>Yes Reduced</td>
</tr>
</tbody>
</table>
Classification

- Features
- Categorical Label

Training data
(labels known)

Test data
(must predict)
Classification

Features

Categorical Label

Training data

Test data

(labels known)

(must predict)
Regression

<table>
<thead>
<tr>
<th>Features</th>
<th>Numeric Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training data</td>
<td>(labels known)</td>
</tr>
<tr>
<td>Test data</td>
<td>(must predict)</td>
</tr>
</tbody>
</table>
Supervised learning

<table>
<thead>
<tr>
<th>Features</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training data</td>
<td></td>
</tr>
<tr>
<td>Test data</td>
<td></td>
</tr>
</tbody>
</table>

Training data (labels known)

Test data (must predict)
Unsupervised learning

Features

Data

Label

?
Clustering

Data

Features

Categorical Label

?
Semi-supervised learning

<table>
<thead>
<tr>
<th>Features</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labelled training data</td>
<td>(labelled)</td>
</tr>
<tr>
<td>Unlabelled training data</td>
<td>N/A (unlabelled)</td>
</tr>
<tr>
<td>Test data</td>
<td>? (must predict)</td>
</tr>
</tbody>
</table>
Machine learning tasks (1)

• Supervised learning:
  – One or more features, one label variable
  – Subtypes:
    • Classification (categorical label)
    • Regression (numeric label)

• Unsupervised learning:
  – No labels or label is hidden/unknown
  – Subtypes:
    • Clustering (hidden categorical label)
    • Other methods to learn hidden structure of data
Machine learning tasks (2)

• Active learning:
  – Need to buy labels for the training instances from a limited budget

• Reinforcement learning:
  – Instead of a label there is an action variable
  – Feedback given with rewards and punishments depending on how good the chosen action was
Machine learning tasks (3)

• Online learning:
  – Data becomes available in a sequential order
  – Predictive model cannot be learned from scratch at every step due to time constraints
  – Instead, the model must be learned incrementally

• One-shot learning:
  – Learning to classify from just one training instance (or just a few)
  – For example, recognize a person after meeting once
Machine learning tasks (4)

• Zero-shot learning:
  – Learning to classify without training instances
  – Instead, there is some description of the class
  – For example, recognize the person from description

• Time-series prediction (forecasting):
  – Learn to predict the future
Machine learning tasks (5)

- **Learning to recommend:**
  - Learn to give recommendations based on known preferences of the same person and other similar people

- **Learning to learn / Lifelong learning**
  - The learning algorithm which is improving with experience

- **See and read more from:**
Cheat-sheet of Python sk-learn package

Learning paradigms

• Rule-based approaches
  – Including decision trees

• Similarity-based methods
  – Including K-nearest neighbours

• Kernel methods
  – Including support vector machines

• Neural networks
  – Including deep neural nets

• Probabilistic modelling
  – Including naïve Bayes method, Gaussian processes, Bayesian networks, conditional random fields
  – Covered in the machine learning course in spring
Deep learning

- What is deep learning?
- Compositional models
- Feed-forward neural network
- Learning a feed-forward neural network
- Example task: learn a parabola
- Back-propagation algorithm
- From regression to classification
- Convolutional neural network

Overview of the machine learning landscape

- Machine learning objectives
- Machine learning tasks and approaches
- Learning theory
- Success stories in machine learning
Under- and Overfitting

Regression:
- M = 1: Predictor too inflexible; cannot capture pattern
- M = 3
- M = 9: Predictor too flexible; fits noise in the data

Classification:

http://www.inf.ed.ac.uk/teaching/courses/iaml/slides/eval-2x2.pdf
The “No Free Lunch” Theorem

Learning **purely from data** is, in general, impossible

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>False</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>True</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>True</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>?</td>
</tr>
</tbody>
</table>

Slide adopted from Konstantin Tretyakov
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

• What should we do to enable learning?
  – Introduce assumptions about data (“inductive bias”):
    1. How do existing data relate to the future data?
    2. What is the system we are learning?

Slide adopted from Konstantin Tretyakov
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

Slide adopted from Konstantin Tretyakov
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

Rule 1: Generalization will only come through understanding of similarity!

Slide adopted from Konstantin Tretyakov
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

Slide adopted from Konstantin Tretyakov
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

Slide adopted from Konstantin Tretyakov
The “No Free Lunch” Theorem

Learning purely from data is, in general, impossible

Rule 2: Data can only partially substitute knowledge about the system!

Slide adopted from Konstantin Tretyakov
• Deep learning
  • What is deep learning?
  • Compositional models
  • Feed-forward neural network
  • Learning a feed-forward neural network
  • Example task: learn a parabola
  • Back-propagation algorithm
  • From regression to classification
  • Convolutional neural network

• Overview of the machine learning landscape
  • Machine learning objectives
  • Machine learning tasks and approaches
  • Learning theory
  • Success stories in machine learning
## AI has surpassed humans in... (1)

<table>
<thead>
<tr>
<th>Year</th>
<th>Domain</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1979</td>
<td>Backgammon</td>
<td>See <a href="#">Backgammon Computer Program Beats World Champion</a></td>
</tr>
<tr>
<td>1997</td>
<td>Chess</td>
<td>See <a href="#">Deep Blue versus Garry Kasparov</a></td>
</tr>
<tr>
<td>2005</td>
<td>Single character recognition</td>
<td>See <a href="#">Computers beat Humans at Single Character Recognition in Reading based Human Interaction Proofs (HIPs)</a>.</td>
</tr>
<tr>
<td>2007</td>
<td>Face recognition</td>
<td>See <a href="#">Face Recognition Algorithms Surpass Humans Matching Faces over Changes in Illumination</a>.</td>
</tr>
</tbody>
</table>

Source: [https://finnaarupnielsen.wordpress.com/2015/03/15/status-on-human-vs-machines](https://finnaarupnielsen.wordpress.com/2015/03/15/status-on-human-vs-machines) (updated: 2018/08/14)
<table>
<thead>
<tr>
<th>Year</th>
<th>Domain</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008</td>
<td>Poker</td>
<td>Michael Bowling, see the news report <a href="https://news.bbc.co.uk/2/hi/technology/7091057.stm">Battle of chips: Computer beats human experts at poker</a>. In 2015 heads-up limit hold’em poker was reported to be not just better than humans, but “essentially weakly solved”, see <a href="https://www.usatoday.com/story/sports/2015/01/18/computer-beats-human-experts-poker/21677197/">Heads-up limit hold’em poker is solved</a>.</td>
</tr>
<tr>
<td>2011</td>
<td>Jeopardy!</td>
<td>In January 2011 the IBM Watson system beat two human contestants in the open-domain question-answering television quiz show. An introduction to the technique in Watson is <a href="https://www.watson.com/">Introduction to “This is Watson”</a>.</td>
</tr>
<tr>
<td>2013</td>
<td>Smooth car driving</td>
<td>Google robotic car head Chris Urmson claimed that their self-driving cars “is driving more smoothly and more safely than our trained professional drivers.” For general car driving the Google car may as of 2014 not be better than humans, e.g., because of problems with road obstacles, see <a href="https://www.google.com/research/papers/selfdriving.pdf">Hidden Obstacles for Google’s Self-Driving Cars</a>.</td>
</tr>
<tr>
<td>Year</td>
<td>Domain</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2013</td>
<td>Age estimation</td>
<td>Estimation of a person’s age from a photo of the face. <a href="https://www.aeic.de/">Age Estimation from Face Images: Human vs. Machine Performance</a>. A considerable improvement with Winner of the ChaLearn LAP 2015 challenge: <a href="https://iacl.fr/">DEX: Deep EXpectation of apparent age from a single image</a></td>
</tr>
<tr>
<td>2014</td>
<td>Deceptive pain expression detection</td>
<td>See <a href="https://www.ncbi.nlm.nih.gov/pubmed/25620797">Automatic Decoding of Facial Movements Reveals Deceptive Pain Expressions</a>: “…and after training human observers, we improved accuracy to a modest 55%. However, a computer vision system that automatically measures facial movements and performs pattern recognition on those movements attained 85% accuracy.”</td>
</tr>
<tr>
<td>2015</td>
<td>Atari game playing</td>
<td>Google DeepMind deep neural network with reinforcement learning, see <a href="https://arxiv.org/abs/1509.06461">Human-level control through deep reinforcement learning</a>: “We demonstrate that the deep Q-network agent, receiving only the pixels and the game score as inputs, was able to surpass the performance of all previous algorithms and achieve a level comparable to that of a professional human games tester across a set of 49 games”. See also <a href="https://arxiv.org/abs/1312.5602">Playing Atari with Deep Reinforcement Learning</a></td>
</tr>
<tr>
<td>Year</td>
<td>Domain</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2015</td>
<td>Image classification</td>
<td>ImageNet classification by Microsoft Research researchers with deep neural network, see Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. Already in 2014 Google was close to human performance, see ImageNet Large Scale Visual Recognition Challenge. Human error rate in the ImageNet has been reported to be 5.1%, – and that was Andrej Karpathy, a dedicated human labeler. Microsoft reported in February 2015 4.94%. Google won one of the competitions in 2014 with “GoogLeNet” having a classification error on 6.66%. Baidu reported in January 2015 an error rate on 5.98% in January 2015 and later in February 5.33%</td>
</tr>
<tr>
<td>2016</td>
<td>Go</td>
<td>DeepMind’s AlphaGo beats best European Go player reported in January Mastering the game of Go with deep neural networks and tree search</td>
</tr>
<tr>
<td>2016</td>
<td>Geoguessing</td>
<td>Google’s PlaNet: “In total, PlaNet won 28 of the 50 rounds with a median localization error of 1131.7 km, while the median human localization error was 2320.75 km” according to Google Unveils Neural Network with “Superhuman” Ability to Determine the Location of Almost Any Image</td>
</tr>
<tr>
<td>2016</td>
<td>Conversational speech recognition</td>
<td>Microsoft Research reports past human performance on benchmark datasets in Achieving human parity in conversational speech recognition</td>
</tr>
</tbody>
</table>
## AI has surpassed humans in... (5)

<table>
<thead>
<tr>
<th>Year</th>
<th>Domain</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016</td>
<td>Lipreading</td>
<td><strong>Lip Reading Sentences in the Wild</strong> writes “… we demonstrate lip reading performance that beats a professional lip reader on videos from BBC television.”</td>
</tr>
<tr>
<td>2017</td>
<td>Poker (heads-up no-limits Texas Hold’em)</td>
<td><strong>According to Andrew Ng</strong> “AI beats top humans”, January 2017. Libratus, a reinforcement learning-based algorithm from Carnegie Mellon University, see <a href="#">Poker pros vs the machines</a>.</td>
</tr>
<tr>
<td>2017</td>
<td>Dota 2 1v1</td>
<td><strong>OpenAI reported</strong> “We’ve created a bot which beats the world’s top professionals at 1v1 matches of Dota 2 under standard tournament rules”, August 2017</td>
</tr>
<tr>
<td>2018</td>
<td>Stanford University reading and comprehension test</td>
<td>The model developed by Alibaba’s Institute of Data Science of Technologies scored 82.44, edging past the 82.304 that rival humans achieved in the Stanford University reading and comprehension test</td>
</tr>
</tbody>
</table>
Deep learning
- What is deep learning?
- Compositional models
- Feed-forward neural network
- Learning a feed-forward neural network
- Example task: learn a parabola
- Back-propagation algorithm
- From regression to classification
- Convolutional neural network

Overview of the machine learning landscape
- Machine learning objectives
- Machine learning tasks and approaches
- Learning theory
- Success stories in machine learning