Example: Decision tree on MNIST
Random forest
Example: DT, RF, KNN not very good
Basic linear classifier
  - Support vector machine
  - Underfitting and overfitting
  - Parameter tuning
  - Cross-validation
  - Machine learning pipeline
  - Learning on imbalanced data
Support Vector Machine (SVM)
Actual boiling point (based on physics)
How does SVM work?

• Let’s explain on a smaller dataset
  – 30 instances from our toy data
  – Same task
Smaller toy dataset

![Graph showing the relationship between temperature (Celsius) and pressure (hPa). The graph includes two states of water: gas (red dots) and liquid (blue triangles).](image)
SVM looks for a straight line that separates the classes

This line separates the classes:
- **gas** above the line
- **liquid** below the line
SVM looks for a straight line that separates the classes.

- This line does not separate the classes.
- 1 instance of liquid is above the line.
SVM looks for a straight line that separates the classes

Many separating lines! Which one is best?
SVM looks for a straight line that separates the classes.

Margin = distance from the line to the closest instance.

Margin of line 1

Margin of line 2

pressure (hPa) vs. temperature (Celsius)
SVM looks for a straight line that separates the classes

Margin = distance from the line to the closest instance

Objective of SVM:
Find the separating line with maximal margin

Margin of line 1
Margin of line 2

pressure (hPa)
SVM looks for a straight line that separates the classes with maximal margin.
SVM looks for a straight line that separates the classes with maximal margin.

Separating line with maximal margin.
SVM looks for a straight line that separates the classes with maximal margin.

**Alternative intuition:**
SVM finds the center of the widest strip that separates the classes.
SVM looks for a straight line that separates the classes with maximal margin.

Alternative intuition: SVM finds the center of the widest strip that separates the classes.

Cannot be made wider because of these 3 instances, called **support vectors** (hence the name Support Vector Machine).
SVM looks for a straight line that separates the classes with maximal margin.

Support vectors are all at the same distance from the line, equal to the margin.
SVM on the original dataset

![Graph showing the state of water based on temperature and pressure. The graph distinguishes between gas and liquid states using an SVM model.](image)
SVM

• We had a 2-dimensional space (2 features):
  – SVM finds the *separating straight line* with maximal margin

• In a 3-dimensional space (3 features):
  – SVM finds the *separating plane* with maximal margin

• In a high-dimensional space (many features)
  – SVM finds the *separating hyper-plane* with maximal margin
Mathematical notation

\( \mathbf{x} = (x_1, x_2) \)
features of an instance

\( \mathbf{w} = (w_1, w_2) \)
weights of the hyper-plane

\( b \) - bias
(shift from origin) of the hyper-plane

\[ \mathbf{w} \cdot \mathbf{x} = w_1 x_1 + w_2 x_2 \]
(dot product)

\[ \| \mathbf{w} \| \] - length of vector \( \mathbf{w} \)

\[ \frac{1}{\| \mathbf{w} \|} \] - margin
SVM mathematically

- Consider a training dataset with $n$ instances $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
  - $x_i$ is a vector of features for the $i$-th instance
  - $y_i$ is the class label ($+1$ or $-1$) of the $i$-th instance
- SVM algorithm must find a weight vector $w$:
  - Margin $1/||w||$ is maximized ($||w||$ is minimized)
  - Instances with $y_i = +1$ have $w \cdot x_i - b \geq 1$
  - Instances with $y_i = -1$ have $w \cdot x_i - b \leq -1$
- SVM must solve the optimization task:
  
  $$
  \text{minimize } ||w|| \\
  \text{such that } y_i(w \cdot x_i - b) \geq 1 \text{ for all } i = 1, \ldots, n
  $$
SVM algorithm

• SVM must solve the optimization task:

\[
\text{minimize } ||w|| \\
\text{such that } y_i(w \cdot x_i - b) \geq 1 \text{ for all } i = 1, \ldots, n
\]

• This task is solved by advanced optimization techniques
  (not covered in this course)
SVM

• Here classes were *linearly separable* (could be separated by a straight line)

• What if the classes are not linearly separable?
  – Then such w does not exist and we need to relax the requirements
  – Soft-margin SVM allows some points to violate the constraint
Hard-margin and soft-margin SVM

- **Hard-margin SVM (requires linear separable):**
  
  \[
  \text{minimize } \|w\| \\
  \text{such that } y_i(w \cdot x_i - b) \geq 1 \text{ for all } i = 1, \ldots, n
  \]

- **Soft-margin SVM (allows violations):**
  
  \[
  \text{minimize } \|w\| + C \sum_{i=1}^{n} \epsilon_i \\
  \text{such that } y_i(w \cdot x_i - b) \geq 1 - \epsilon_i \text{ and } \epsilon_i \geq 0 \text{ for all } i = 1, \ldots, n
  \]
  
  - \( C > 0 \) is the cost parameter of violations
  
  - Lower \( C \) means more violations are accepted
  
  - Both margin and sum of violations are minimized
Soft-margin SVM
Toy example extended

• Our toy example had:
  – Temperature range 99-101°C
  – Pressure range 990-1030 hPa

• Let’s now consider wider ranges:
  – Temperature range 50-150°C
  – Pressure range 0-3000 hPa
Toy dataset extended
Soft-margin SVM with $C=10000$
Actual class boundary (based on physics)
Linear and non-linear SVM

• Often the actual class boundary is non-linear
  – Then linear SVM has low accuracy
Kernel SVM

- Solution: map points non-linearly into a higher-dimensional space where the points become linearly separable.
Kernel SVM

• Examples of kernels (non-linear mappings)
  – Linear kernel:
    • New features = original features (no mapping)
  – Polynomial kernel:
    • New features = all polynomials of features up to some degree
  – Radial basis function (RBF) kernel:
    • New feature space is infinite-dimensional (not explained in this course)
    • Intuition: new features somehow relate to Euclidean distances to training instances
  – Kernels can be combined
SVM with polynomial kernel (degree = 2)
Actual class boundary (based on physics)

The diagram illustrates the phase transition of water between its gas (red dots) and liquid (blue triangles) states as a function of temperature in Celsius and pressure in hPa. The boundary curve separates the regions where water exists in gas phase from the regions where it is in liquid phase.
Linear separation in non-linear world

\[ \phi \]
Mapping Example

- map data points into feature space with some function $\phi$
- e.g.:
  - $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^2$
  - $(x_2, x_2) \rightarrow (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$

- hyperplane $\langle w \cdot z \rangle = 0$, as a function of $x$:

$$w_1x_1^2 + w_2\sqrt{2}x_1x_2 + w_3x_2^2 = 0$$
SVM in practice

• Usually in practice:
  – the actual class boundary is non-linear
  – there is extra noise

• Therefore the commonly used method is:
  – kernel soft-margin SVM

• Since it is so common, both ‘kernel’ and ‘soft’ are usually omitted (not written):
  – SVM
Example of what SVM can do
Example: Decision tree on MNIST
Random forest
Example: DT, RF, KNN not very good
Basic linear classifier
Support vector machine

- Underfitting and overfitting
- Parameter tuning
- Cross-validation
- Machine learning pipeline
- Learning on imbalanced data
Simple vs complicated

• We can now build very complicated models
• Is a complicated model with higher accuracy on the training data better than a simple model?
Training the model

Feature #1

Feature #2

Feature #1

Feature #2

Adapted from slides by Dmytro Fishman

Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
Training the model

Which model we should use?

Simple; not perfect fit

Complicated; ideal fit

Adapted from slides by Dmytro Fishman

Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
Training the model

Which model we should use?

Simple; not perfect fit  Complicated; ideal fit

Adapted from slides by Dmytro Fishman
Training the model

Which model we should use?

Simple; not perfect fit  
Complicated; ideal fit

Adapted from slides by Dmytro Fishman

Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
Training the model
Which model we should use?

Simple; not perfect fit
Complicated; ideal fit

Adapted from slides by Dmytro Fishman
Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
Overfitting

Too general model

Just right!

Overfitting

Adapted from slides by Dmytro Fishman
Underfitting and overfitting

- Underfitting
- Just right
- Overfitting

Prediction error vs. Model complexity

- Simpler models
- Training data
- Test data
- More complex models
Why test error higher than train error?

- Test error (prediction error on the test data) is almost always higher than training error (prediction error on the training data). Why?

- Answer:
  - On training data the model fits both **signal** (meaningful information) as well as **noise** (randomness)
  - On test data the **signal** is the same but **noise** is different and cannot be predicted
  - Model predicts **noise** well on training data but not on the test data, causing a gap in error rates
Machine learning methodology

• Therefore, when reporting accuracy of a model, then it is important to report it on the test set

• Splitting the dataset into training and test folds (sub-datasets) is usually the first step in the machine learning workflow

• Typical split sizes are:
  - 50% train 50% test
  - 67% train 33% test
  - 80% train 20% test
Underfitting and overfitting

![Graph showing prediction error vs. model complexity with Underfitting, Just right, and Overfitting regions.

- Underfitting: The model is too simple, leading to high prediction error on both training and test data.
- Just right: The model complexity is balanced, resulting in low prediction error on both training and test data.
- Overfitting: The model is too complex, fitting the training data well but failing to generalize to unseen data.](image-url)
Overfitting

• How can I recognize overfitting?
  – Test error is much worse than training error

• When does my model have the right complexity?
  – Its complexity is needed
    • Simpler models have higher train and test errors
  – Cannot make it more complex
    • Increase in complexity leads to higher test error
    • However, training error might decrease
How to recognize overfitting?

A. More errors on training data than on test data
B. Model is making too good predictions on the test data
C. The machine learning method is running too slowly
D. More errors on test data than on train data
E. Model is making too bad predictions on the training data
F. Not sure

![Response Counter](image)
Example: Decision tree on MNIST

Random forest

Example: DT, RF, KNN not very good

Basic linear classifier

Support vector machine

Underfitting and overfitting

- Parameter tuning
- Cross-validation
- Machine learning pipeline
- Learning on imbalanced data
# Parameter tuning

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Hyper-parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-nearest neighbour</td>
<td>$K$ - number of neighbours, $(1, \ldots, 100)$, Distance measure</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>Metric (‘gini’, ‘information gain’), Stopping criteria</td>
</tr>
<tr>
<td>Random Forest</td>
<td>Number of trees $(3, \ldots, 100, \text{more better})$, metric (‘gini’, information gain’), stopping criteria</td>
</tr>
<tr>
<td>SVM</td>
<td>$C (10^{-5}, \ldots, 10^5)$ and kernel parameters</td>
</tr>
</tbody>
</table>

Adapted from slides by Dmytro Fishman
Pitfalls in parameter tuning

• Typical **methodological mistake:**
  – Train a model with many parameter value combinations
  – Find the combination with the lowest test error
  – Report the best combination together with its test error
  
    • E.g. SVM with polynomial kernel of degree 4 and $C=42$, test error 3% (test accuracy 97%)

3% is now too optimistic because it was obtained on the same test set where parameters were tuned!
Parameter tuning done correctly

1. Split the training data into training data and validation data (for example, 80%+20%)
2. Train a model with many parameter value combinations
3. Find the combination with the lowest error on the validation data
4. Find the test error of the best combination
5. Report the best combination together with its test error
The whole dataset 100%
The whole dataset 100%

Training 60%
The whole dataset 100%

Training 60%
For fitting initial model
The whole dataset 100%

Training 60%  Validation 20%

For fitting initial model
The whole dataset 100%

Training 60%
For fitting initial model

Validation 20%
For parameter tuning & performance evaluation

Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
The whole dataset 100%

Training 60%
For fitting initial model

Validation 20%
For parameter tuning & performance evaluation

Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
The whole dataset 100%

Training 60%
For fitting initial model

Validation 20%
For parameter tuning & performance evaluation
The whole dataset 100%

Training 60%
For fitting initial model

Validation 20%
For parameter tuning & performance evaluation
The whole dataset 100%

Training 60%
For fitting initial model

Validation 20%
For parameter tuning & performance evaluation
The whole dataset 100%

Training 60%
For fitting initial model

Validation 20%
For parameter tuning
& performance evaluation

Testing 20%

Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
The whole dataset 100%

Training 60%
For fitting initial model

Validation 20%
For parameter tuning & performance evaluation

Testing 20%
For one shot evaluation of trained model

Meelis Kull - Autumn 2020 - LTAT.02.002 – Intro to Data Science - Lecture 07
Example: Decision tree on MNIST
Random forest
Example: DT, RF, KNN not very good
Basic linear classifier
Support vector machine
Underfitting and overfitting
Parameter tuning
- Cross-validation
- Machine learning pipeline
- Learning on imbalanced data