

2024 edition

Deep Learning for Music Analysis and Generation

# Introduction

## Course



**Yi-Hsuan Yang** Ph.D.  
[yhyangtw@ntu.edu.tw](mailto:yhyangtw@ntu.edu.tw)

# Course Website

<https://cool.ntu.edu.tw/courses/41158>

<https://affige.github.io/teaching.html>

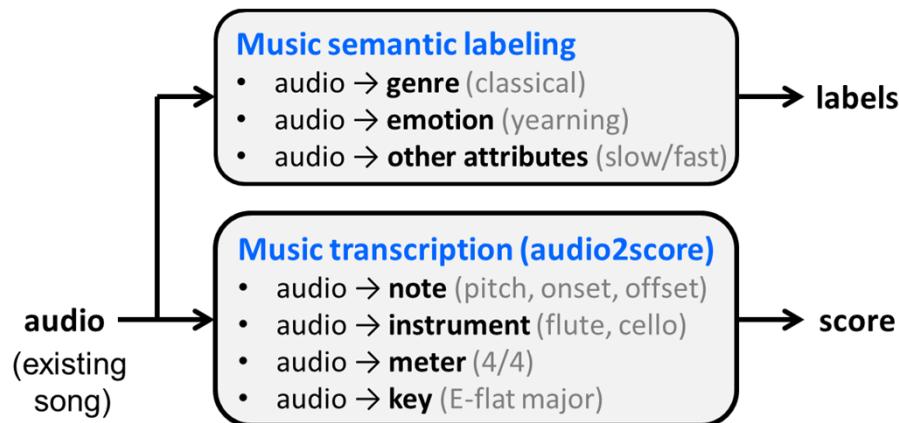


# **Outline**

- **Music & AI**
- The course
- ML 101

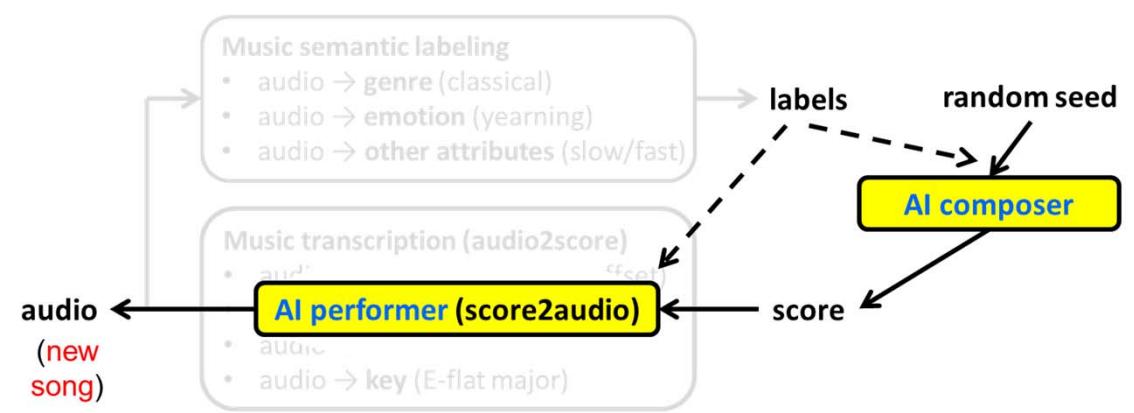
# Music AI; or *Music Information Research (MIR)*

- Music analysis



- music understanding
- music search
- music recommendation

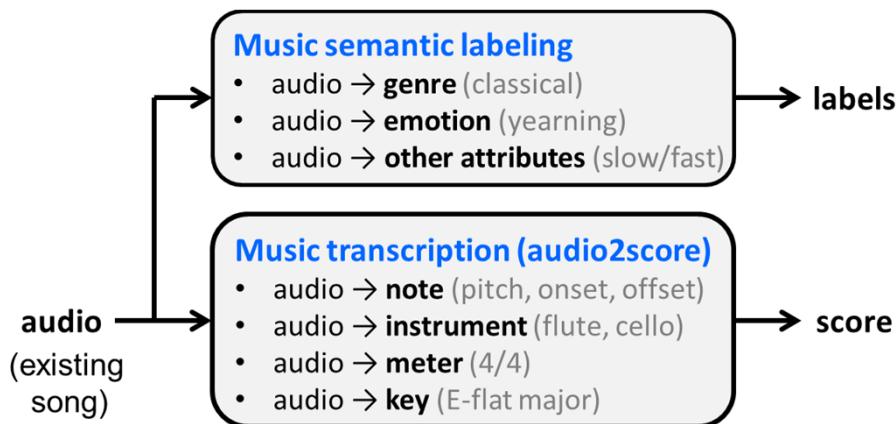
- Music generation



- MIDI generation
- audio generation
- MIDI-to-audio generation

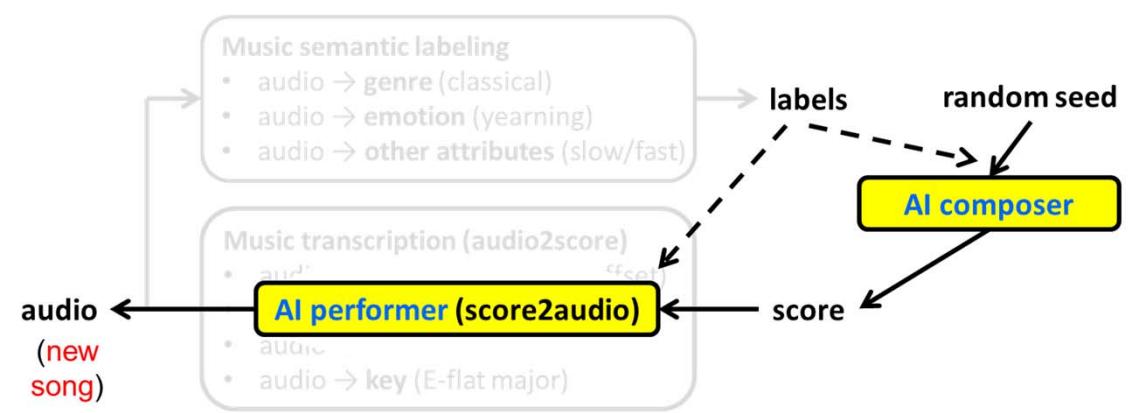
# Music AI: {signal processing, machine learning} + music

- Music analysis



- music understanding
- music search
- music recommendation

- Music generation



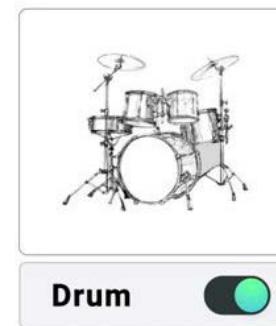
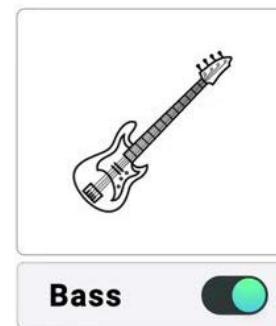
- MIDI generation
- audio generation
- MIDI-to-audio generation

# Music Analysis Demo: Source Separation

<https://www.gaudiolab.com/technology/source-separation>

<https://www.youtube.com/watch?v=r4NcZbgtlgw>

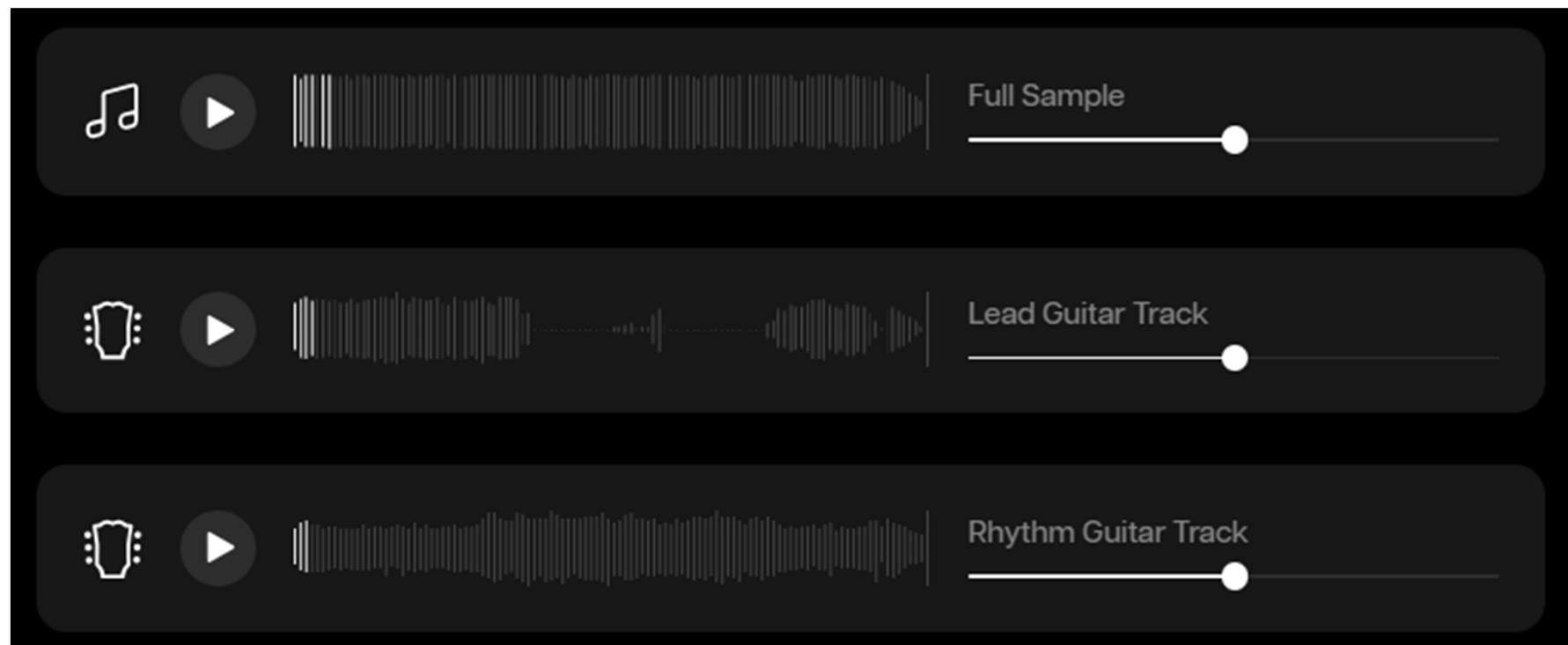
♪♪ Eagles 'Hotel California'



GAUDIO

# Music Analysis Demo: Source Separation

<https://moises.ai/blog/moises-news/new-guitar-separation-models/g>



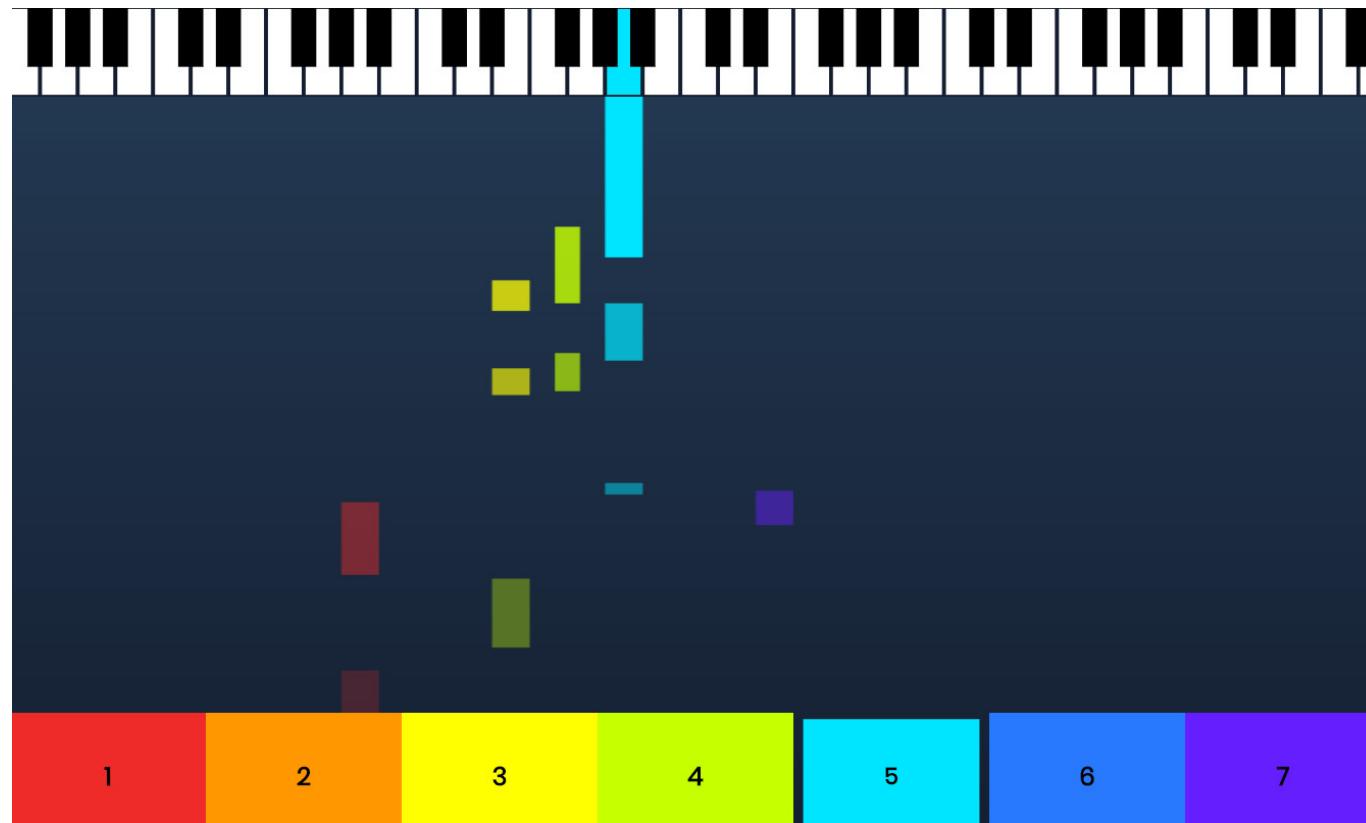
# **Positive Use Cases of Music Generation**

- **Make music easier to play with** (i.e., for common people)
  - “democratization” of music creation
- **Make musicians’ life easier** (i.e., for musicians)
  - inspire ideas
  - suggest continuations
  - suggest accompaniments
- **Create copyright free music for videos or games**
- **Music education / learning**
- **Digital archival**

# Music Generation Demo 1: Piano Genie

(Make music easier to play with)

<https://magenta.tensorflow.org/pianogenie>



# Music Generation Demo 2: Tone Transfer

(Make music easier to play with)

<https://sites.research.google/tonetransfer>



## Music Generation Demo 3: KaraSinger

<https://jerrygood0703.github.io/KaraSinger/>

Lyrics:

In this paper we propose  
a novel neural network model  
called Karaoke singer for a less studied  
singing voice synthesis task  
named score-free SVS  
in which the prosody and melody are  
spontaneously decided by machine.

# Music Generation Demo 4: AI Sandee



<https://www.youtube.com/watch?v=nWTuZIRU80A>

「音樂製作人的工作是無法被取代的」。AI vocal 要怎麼唱，能唱得多好，終究需要專業音樂製作人，以人類的美學和經驗去引導 AI，要如何將 AI 升華到情感面，終究還是需要製作人的能力，以及對音樂的想像力。

作為一個仍在線上的歌手與製作人，由我親自處理自己的AI vocal，讓這首歌傳達出「創作者、歌者不怕 AI 的挑戰」、「我們擁有自己的聲音的控制權」等訊息，同時也是「人類的思考和意志，才是人之所以為人」的巨大宣示。

透過聆聽《教我如何做你的愛人》，試著探討：「若 AI 已經能模擬原唱的一切，那麼原唱歌手的價值會是什麼？」

當 AI 真正學會唱歌之後，就是創作人與歌手，重新理解自身價值的時候了。....by公主

SandeeChan · 陳珊妮 公主粉絲團

1d ·

今天終於能夠揭示這個真相：《教我如何做你的愛人》是陳珊妮的 AI 模型演唱，以及我選擇在白色情人節上架的原因。

順帶一提，MV 今天上線了！（還不快去看）

在 AI 發展熱議的當下，希望透過這首歌，與所有關心創作的人一起思考——如果 AI 的時代必將到來，創作人該在意的或許不是「我們是否會被取代」，而是「我們還可以做些什麼」。[... See more](#)



# Music Generation Demo 5: Text-to-Music

(Create copyright free music for videos or games)

[https://www.youtube.com/watch?v=R4oY1A9\\_Jss](https://www.youtube.com/watch?v=R4oY1A9_Jss)

The screenshot shows a dark-themed user interface for a music generation service. On the left, a sidebar includes links for Explore, Create, Library, Help / FAQs, Discord Community, and Sign up. The main area is titled "Trending" and lists five tracks:

- Turkey Time (pop)
- A is for Amazing (gospel soul choir)
- Mistletoe and Missing You (pop acoustic)
- おきたくないのでベッドを伸ばせ (alternative rock, emotional)
- Neon Velocity (energetic electronic synthwave)

Below the trending list, there's a track titled "Glittering Shadows" (rock dark heavy) by "hyperloop cute vocoder vocal robot". The track has 37 likes and is 0:41 long. The lyrics displayed are:

[Verse]  
In the depths of the night, where darkness prevails  
A glimmer of light, a tale that entails

At the bottom of the screen, the currently playing track is "Glittering Shadows" (rock dark heavy), which is 0:19 long. The interface includes standard video controls like play/pause, volume, and a progress bar.

# Music Generation Demo 5: Text-to-Music

(Create copyright free music for videos or games)

<https://www.youtube.com/watch?v=8S95f1BR1Ls>



貓貓雨的形成是特殊大氣現象

# Global Interest in Music AI: Industry



(Slide from Rujing Huang, Bob L. T. Sturm, and Andre Holzapfel, "De-centering the West: East Asian Philosophies and the Ethics of Applying Artificial Intelligence to Music," ISMIR 2021)

# Global Interest in Music AI:

## Purdue Music Technology Research Group

<https://ai4musicians.org/>



**Evaluator**

**Fall 2023 - present**

Evaluator is an app that aims to automatically evaluate musicians' intonation. It uses YOLO localization techniques to detect the instrument and spectrogram analysis and multi-task learning to evaluate intonation and correct them.

Drawing by Cecilia Ines Sanchez

**Companion**

**Fall 2023 - present**

Companion is an app that not only records a musician's performance but also actively responds to their playing. This project involves machine learning and fine-grained audio processing and utilizes small NLP language models.

Drawing by Cecilia Ines Sanchez

**Mus2Vid**

**Spring 2022 - present**

Mus2Vid is a real-time art project that generates images in response to classical music. It uses recurrent neural networks to analyze emotion and genre qualities, which are then used to generate images.

[LEARN MORE](#)

generate visual representations → adapt references → refine



**Robot Cello**

**Spring 2024 - present**

As the name suggests, Robot Cello is a robot designed to play cello. The project is currently using motion capture technology to get training data.

We partner with the Purdue University Department of Music. On the left is a video of Prof. Yun-Jung Kim, Clinical Associate Professor of Cello.

# Global Interest in Music AI: Academia

- MIT EECS | Anna Huang (<https://czhuang.github.io/>)
- CMU CS | Chris Donahue (<https://chrisdonahue.com/>)
- U Michigan SMTD | Hao-Wen Dong (<https://hermandong.com/>)
- Stanford CCRMA | Ge Wang (<https://ccrma.stanford.edu/~ge/>)
- NYU MARL | Brian McFee (<https://brianmcfee.net/>)
- U Rochester | Zhiyao Duan (<https://hajim.rochester.edu/ece/sites/zduan/>)
- Georgia Tech | Alexander Lerch (<https://www.alexanderlerch.com/>)
- Academia Sinica | Li Su (<https://homepage.iis.sinica.edu.tw/pages/lisu/>)

# **Outline**

- Music & AI
- **The course**
- ML 101

# This Course: Prerequisites

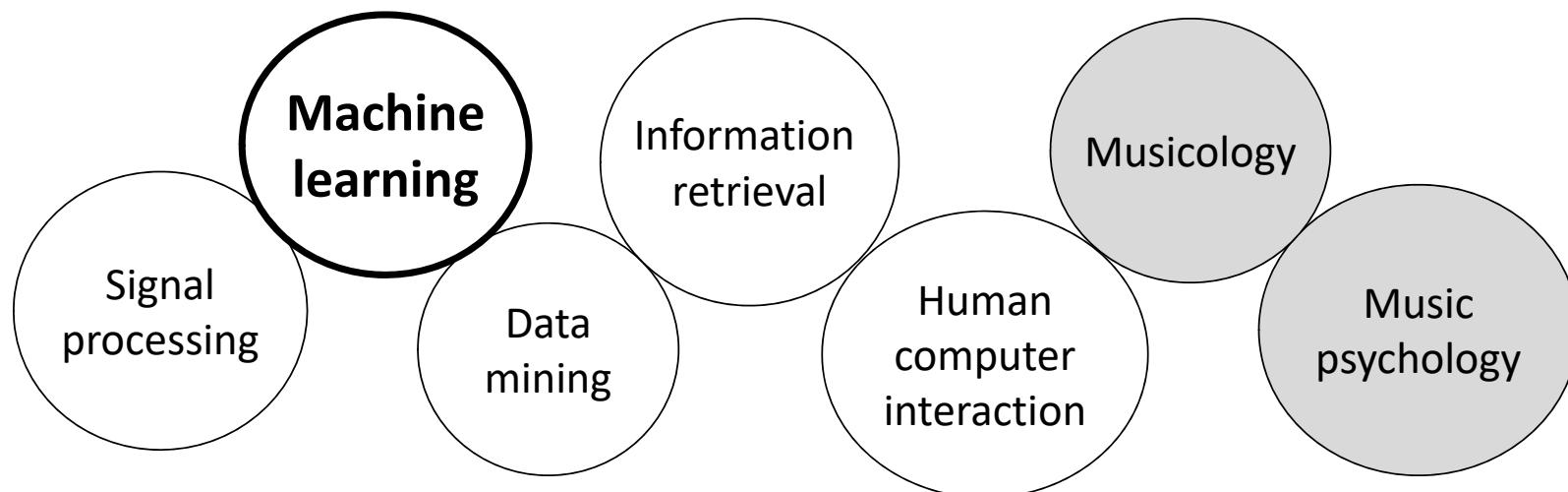
- **Graduate level** (CommE5070) @ NTU GICE
  - It's ***NOT*** a music course
  - It's an EE/CS graduate-level course working on music data/problems
- Suitable for people who have
  - Great interest in music
  - Good background in machine learning & deep learning
  - Good coding experience in python and a deep learning framework such as PyTorch
- Not a class designed for deep learning newbies

# This Course: Wills and Won'ts

- Will talk about
  - Domain knowledge in music data representation
  - Domain knowledge in music analysis: timbre, rhythm, pitch
  - Deep learning-based music analysis
  - Deep learning-based audio generation
  - Deep learning-based MIDI sequence generation
- **Won't** talk about (too much on)
  - Basics in machine learning and deep learning
  - Applications in other domains

# This Course: Objective

- Get you ready to do ML research on music AI
- Lots of hands-on



# Lecturer

- Lecturer
  - Yi-Hsuan Yang (楊奕軒)
    - <https://affige.github.io/>
    - [yhyangtw@ntu.edu.tw](mailto:yhyangtw@ntu.edu.tw)
- Office hour
  - Thursday 9:30-11:30, or by appointment
  - Office: EE2-337 (電二)

# Teaching Assistants

- TAs
  - Wei-Jaw Lee (李維釗)
    - weijaw2000@gmail.com
  - Hsin Ai (艾芯)
    - iivvyy0728777@gmail.com
- Office hour
  - Tuesday 14:00-15:30, or by appointment
  - Office: BL-505 (博理館)

## Location & Time

- Location: 學新 118
- Time: Thursday 6,7,8
  - 6: 13:20-14:10
  - 7: 14:20-15:10
  - 8: 15:20-16:10 (i.e., 10 mins earlier)

# Textbook

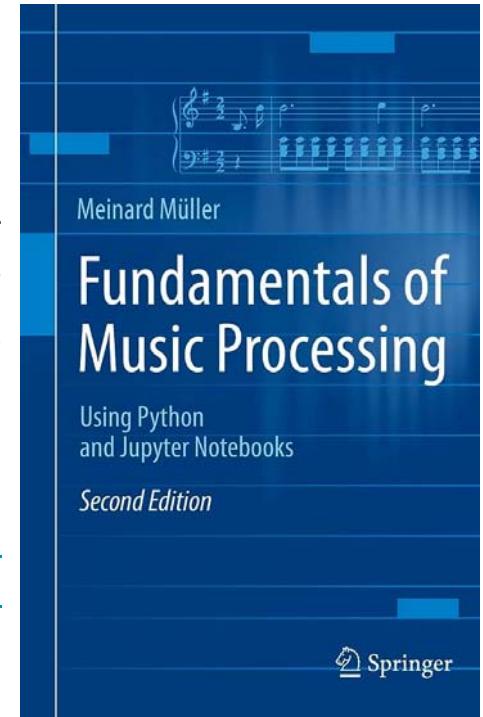
(for the music analysis part)

- Reference textbook

Meinard Müller  
*Fundamentals of Music Processing*  
Using Python and Jupyter Notebooks

ISBN: 978-3-030-69808-9  
Springer, April 2021

<https://www.audiolabs-erlangen.de/fau/professor/mueller/bookFMP>  
<https://www.audiolabs-erlangen.de/resources/MIR/FMP/C0/C0.html>

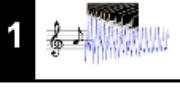
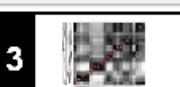


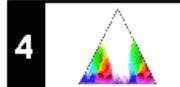
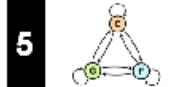
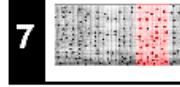
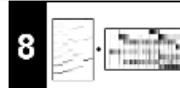
- Related book

– *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*, Wiley  
<https://github.com/alexanderlerch/pyACA>  
<https://github.com/alexanderlerch/ACA-Slides>

# FMP Notebook

<https://www.audiolabs-erlangen.de/resources/MIR/FMP/C0/C0.html>

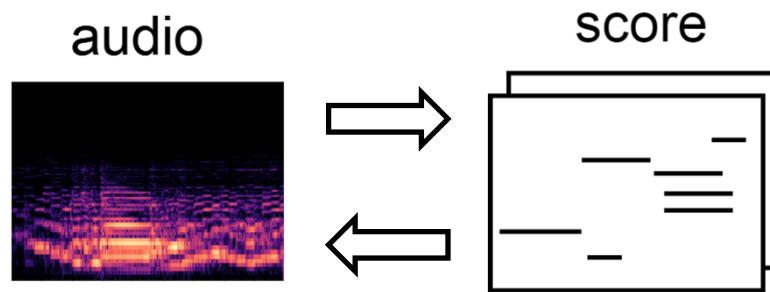
Part	Title	Notions, Techniques & Algorithms	HTML	IPYNB
B	 <a href="#">Basics</a>	Basic information on Python, Jupyter notebooks, Anaconda package management system, Python environments, visualizations, and other topics	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
0	 <a href="#">Overview</a>	Overview of the notebooks ( <a href="https://www.audiolabs-erlangen.de/FMP">https://www.audiolabs-erlangen.de/FMP</a> )	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
1	 <a href="#">Music Representations</a>	Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
2	 <a href="#">Fourier Analysis of Signals</a>	Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
3	 <a href="#">Music Synchronization</a>	Chroma feature, dynamic programming, dynamic time warping (DTW), alignment, user interface	<a href="#">[html]</a>	<a href="#">[ipynb]</a>

Part	Title	Notions, Techniques & Algorithms	HTML	IPYNB
4	 <a href="#">Music Structure Analysis</a>	Similarity matrix, repetition, thumbnail, homogeneity, novelty, evaluation, precision, recall, F-measure, visualization, scape plot	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
5	 <a href="#">Chord Recognition</a>	Harmony, music theory, chords, scales, templates, hidden Markov model (HMM), evaluation	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
6	 <a href="#">Tempo and Beat Tracking</a>	Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
7	 <a href="#">Content-Based Audio Retrieval</a>	Identification, fingerprint, indexing, inverted list, matching, version, cover song	<a href="#">[html]</a>	<a href="#">[ipynb]</a>
8	 <a href="#">Musically Informed Audio Decomposition</a>	Harmonic/percussive separation, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF)	<a href="#">[html]</a>	<a href="#">[ipynb]</a>

# Grading Policy

- Grading policy
  - **Assignments** (60%), 4 times, each 15%
  - **Final Project** (40%): for teams of 2 or 3 (recommended)
- **Work hard to get high score**
  - I don't plan to please the students

# Course Material



## Topics

- **audio → audio:** signal processing
- **audio → score:** transcription
- **score → score:** composition
- **score → audio:** synthesis
- **audio → knowledge:** audio analysis
- **score → knowledge:** symbolic-domain analysis

# Syllabus

- W1. Introduction to the course
  - HW0: Musical note classification
- W2. Fundamentals of musical audio
- W3. Music classification and transcription
  - HW1: Instrument activity detection
- W4. Source separation
- W5. GAN & Vocoders
  - HW2: Source separation
- (W6. National holiday)
- W7. Fundamentals of symbolic music
  - HW3: Lyrics-conditioned melody generation
- W8. Symbolic MIDI generation
- W9. Synthesis and style transfer
- W10. Singing voice generation
- W11. Text-to-music generation
- W12. Project pitch (proposals of final projects)
- W13. Differentiable DSP models and auto-mixing
- W14. Miscellaneous Topics
- (W15. Break)
- W16. Oral presentation of final projects

# Assignments

- **Programming (in python) + report (in English)**
  - We assume that you are comfortable with programming in python and deep learning frameworks such as PyTorch
  - One new assignment **every two weeks**; can be quite tough for DL beginners
  - Submit **code + model + report**
  - Will select around 3 people to share their work for HW2 and HW3
  - NO cheating: Will run *plagiarism detector*
- Topics
  - HW0: musical note classification
  - HW1: instrument activity detection
  - HW2: source separation
  - HW3: lyrics-conditioned melody generation

# Assignments

		HW0	HW1	HW2	HW3	Final project
		Classification	Detection	Separation	Melody generation	
W1	Intro	<i>announce on W1</i>				
W2	Fundamentals of audio signals					
W3	Music classification and transcription	<i>due before W4</i>	<i>announce on W3</i>			
W4	Source separation		<i>due before W5</i>			
W5	GAN and vocoders			<i>announce on W5</i>		
W6	(break)			<i>due before W8</i>	<i>announce on W7</i>	
W7	Fundamentals of symbolic music			<i>share on W9</i>	<i>due before W11</i>	
W8	MIDI generation				<i>share on W11</i>	
W9	Synthesis and style transfer					<i>oral report</i>
W10	Singing voice generation					
W11	Text-to-music generation					
W12	Project pitch					
W13	DDSP and auto-mixing					
W14	Miscellaneous Topics					
W15	(break)					
W16	Project presentation					<i>oral report</i>
W17						
W18						<i>paper report</i>

# **NO Cheating**

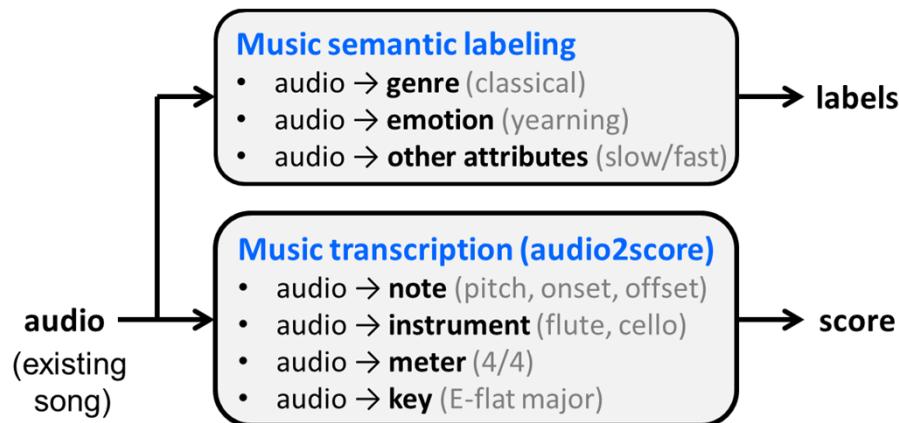
- Once caught: **failure** of the course

# Final Project

- For teams of 2 or **3** (recommended)
- Start earlier & form teams
- Deadline for **team-up**: W10
- **Project pitch**: W12
- **Final presentation**: W16
- Deadline for **final report**: W16+2

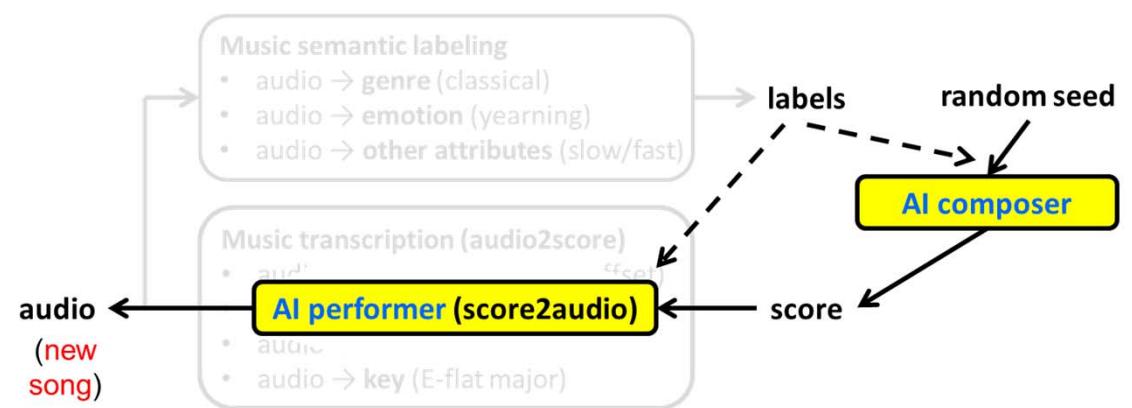
# Final Project

- Music analysis



- music semantic labeling
- music transcription
- source separation

- Music generation



- MIDI generation
- audio generation
- MIDI-to-audio generation

# Final Project Showcase

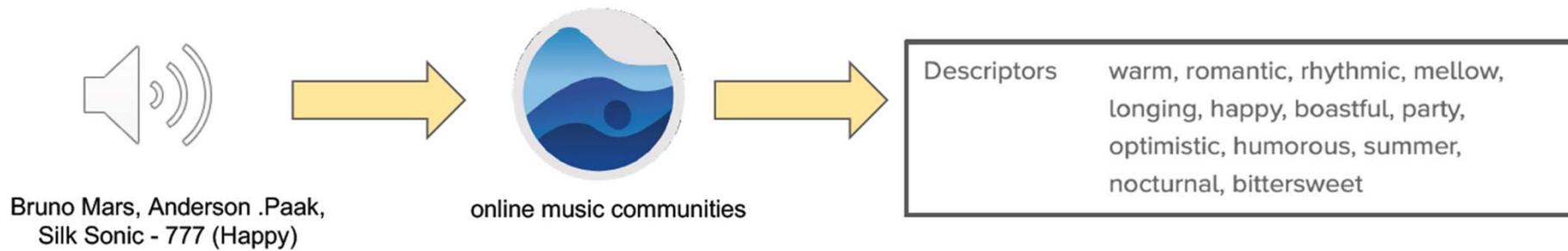
[https://affige.github.io/teaching\\_depmir23.html](https://affige.github.io/teaching_depmir23.html)

- **List of final projects** presented by the class students

- *Positive Grid guitar effect removal*
  - extensions of this project has been published at [DAFx'24](#)
- *Personalization on music generation*
  - extensions of this project has been published at [ISMIR'24](#)
- *Music2fail: transfer music to fail recorder style*
  - extensions of this project has been submitted to [APSIPA ASC'24](#)
- *Music generation from image and text*
  - extensions still ongoing
- *Neural screaming voice conversion*
  - extensions still ongoing
- *Acapella accompaniment generation*
  - extensions still ongoing
- *Toward cross-lingual singing voice conversion*
- *Personalized singing voice beautifier*
- *Multimodal music emotion recognition*
- *Music emotion recognition using contrastive language aud*
- *Evaluation toolkit for controllable text-to-music models*
- *Muse2bach*
- *GuitarPedal simulation*
- *OSU! Taiko auto mapper*
- *AI Nice Chord progression*
- *Timbre transfer by diffusion models*

# Final Project Showcase

- Multimodal music emotion recognition: audio + lyrics + MIDI



Song	Model	Prediction
Bruno Mars, Anderson .Paak, Silk Sonic - 777 (Happy)	CRNN	Happy
	MERT	Happy
	Whisper + BERT	Angry
	Late fusion	Angry
	LSTM-Attn + Remi	Angry

# Final Project Showcase

- Guitar effect removal

[https://y10ab1.github.io/guitar\\_effect\\_removal/](https://y10ab1.github.io/guitar_effect_removal/)



# Final Project Showcase

- Transfer music to failed recorder style



source



target  
style



stargan



vaegan

# **genmusic\_demo\_list**

[https://github.com/affige/genmusic\\_demo\\_list](https://github.com/affige/genmusic_demo_list)

## About

a list of demo websites for automatic  
music generation research

artificial-intelligence

music-generation

# Resources

- ML/DL
  - <http://speech.ee.ntu.edu.tw/~tlkagk/courses.html>
  - <https://www.csie.ntu.edu.tw/~htlin/course/>
  - <https://www.csie.ntu.edu.tw/~yvchen/teaching>
  - <https://courses.cs.washington.edu/courses/cse599i/20au/> (generative models)
- Music information research
  - [https://www.audiolabs-erlangen.de/fau/professor/mueller/teaching/2023w\\_mpa](https://www.audiolabs-erlangen.de/fau/professor/mueller/teaching/2023w_mpa)
  - <https://musicinformationretrieval.com/>
  - <https://mac.kaist.ac.kr/~juhan/gct634/index.html>
  - <http://www.jordipons.me/apps/teaching-materials/>
  - <https://www.upf.edu/web/smrc/audio-signal-processing-for-music-applications>

# Resources

- Conference proceedings
  - Int'l Soc. Music Information Retrieval Conf. (ISMIR)
  - Int'l Conf. Acoustic, Speech, and Signal Processing (ICASSP)
  - ACM MM, ACM ICMR, ACM SIGIR, IEEE ICME
- Transactions
  - IEEE Trans. Audio, Speech and Language Processing (TASLP)
  - IEEE Trans. Multimedia (TMM)
  - IEEE Trans. Signal Processing (TSP)

# Additional Enrollment

<https://forms.gle/p6ro7tE9ibMb5S9VA>

- **Sign up at NTU Cool (選課意願登記)**
- **AND, Fill the form before 23:59, September 6 (Friday)**
  - DL background
  - Music background
  - Ideas for final project
- Will announce the result before next **Monday**
  - Will only send a mail to those selected
  - Will also post the result online at <https://affige.github.io/teaching.html>

# Outline

- Music & AI
- The course
- **ML 101**
  - Will talk about DL 101 in the next lecture

# Machine Learning & Deep Learning

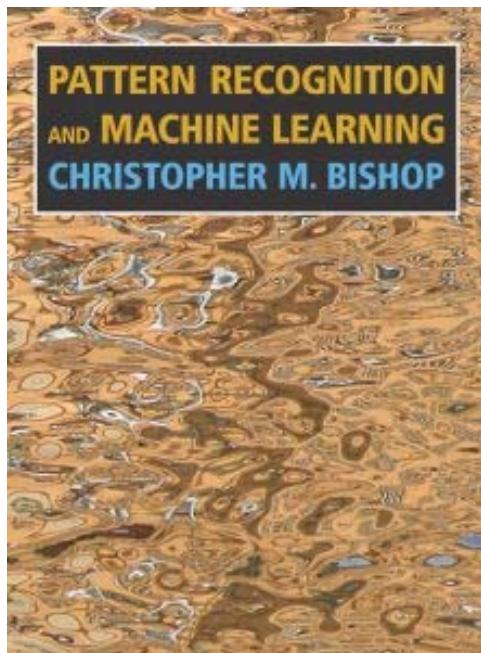
[https://nol2.aca.ntu.edu.tw/nol/coursesearch/print\\_table.php?course\\_id=921%20U2620&class=02&dpt\\_code=9210&ser\\_no=40152&semester=113-1&lang=CH](https://nol2.aca.ntu.edu.tw/nol/coursesearch/print_table.php?course_id=921%20U2620&class=02&dpt_code=9210&ser_no=40152&semester=113-1&lang=CH)

課程名稱	機器學習 Machine Learning
開課學期	113-1
授課對象	電機工程學研究所
授課教師	劉子毓
課號	EE5184
課程識別碼	921EU2620
班次	02
學分	4.0
全/半年	半年
必/選修	選修
上課時間	星期二2,3,4,5(9:10~13:10)
上課地點	電二106
備註	本課程以英語授課。 總人數上限：60人

週次	日期	單元主題
第1週	9/03	Introduction and rubric review; What is AI? What is ML?
第2週	9/10	Regression; Cross validation; Hyperparameter optimization
第3週	9/17	Support vector machine (SVM); the kernel trick
第4週	9/24	Decision tree, naive Bayes classifier
第5週	10/01	Ensemble methods, random forest and AdaBoost
第6週	10/08	Dimension reduction, PCA, MDS, tSNE
第7週	10/15	GMM as an example of Bayesian modeling
第8週	10/22	Midterm
第9週	10/29	Multilayer perceptron (MLP)
第10週	11/05	Convolutional neural network (CNN)
第11週	11/12	Autoencoder (AE)
第12週	11/19	Variational autoencoder (VAE)
第13週	11/26	Generative adversarial network (GAN)
第14週	12/03	Transformer
第15週	12/10	Invited talk from industry
第16週	12/17	Final exam

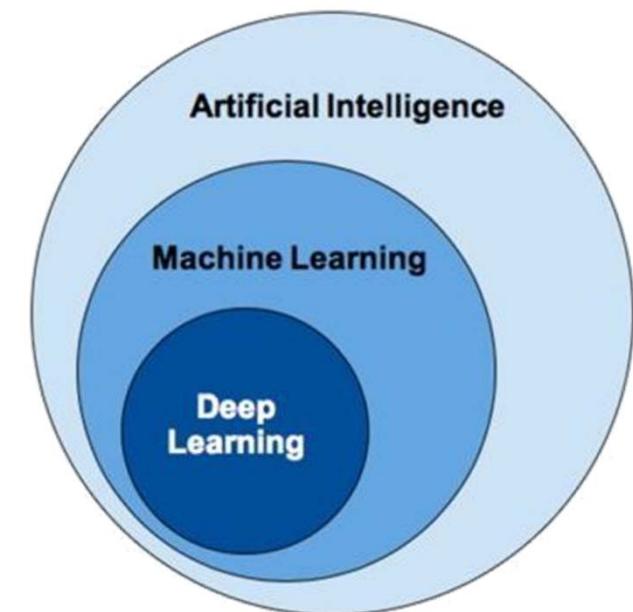
# Machine Learning

<https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/>

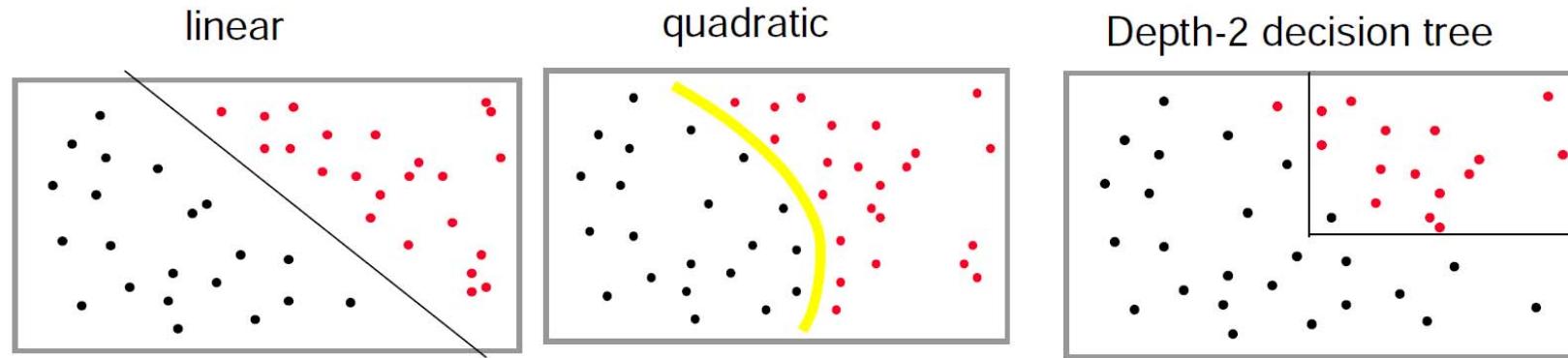


*Freely available*

1. Introduction
2. Probability Distributions
3. Linear Models for Regression
4. Linear Models for Classification
5. Neural Networks
6. ...



# Classification: Overview (1/2)

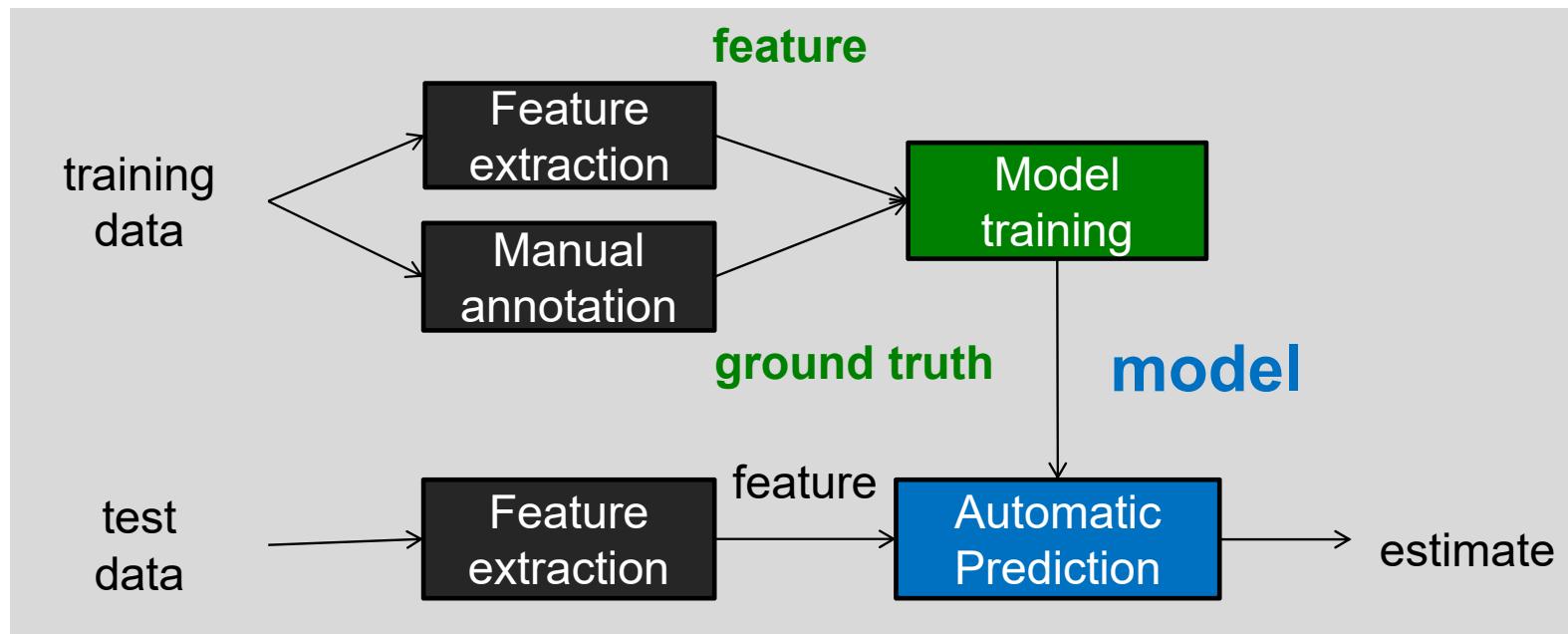
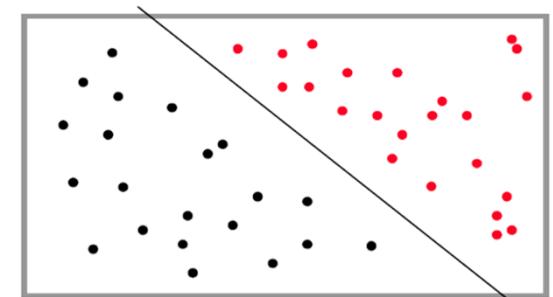


- Given:  $\{(x_1, y_1), \dots, (x_N, y_N)\}$ 
  - $x_i \in \mathbb{R}^M$ : feature representation of a data instance
  - $y_i \in \{-1, +1\}$  : class label of that instance
- Goal
  - learn a separating hyperplane

# Classification: Overview (2/2)

- Given:  $\{(x_1, y_1), \dots, (x_N, y_N)\}$ 
  - $x_i \in \mathbb{R}^M$ : feature representation
  - $y_i \in \{-1, +1\}$  : class label

linear



## Model Training (1/3)

1. Collect data ( $y$ ) and extract features ( $x$ )
2. Build model: choose hypothesis class  $\mathcal{H}$  and loss function  $l$ 
  - hypothesis class
    - linear:  $\hat{y} = f(x, w, b) = \text{sgn}(w^T x + b)$
    - quadratic:  $\hat{y} = f(x, A, w, b) = \text{sgn}(x^T A x + w^T x + b)$

$$\text{sgn}(x) = \begin{cases} +1, & x \geq 0; \\ -1, & x < 0. \end{cases}$$

□ “parameters” to be *learned*:  $\theta = \{A, w, b\}$

## Model Training (2/3)

1. Collect data ( $y$ ) and extract features ( $x$ )
2. Build model: choose hypothesis class  $\mathcal{H}$  and loss function  $l$ 
  - loss function
    - **indicator function:**  $l(y, \hat{y}) = 0$  if  $y = \hat{y}$ ;  
 $l(y, \hat{y}) = 1$  otherwise
    - **square loss:**  $l(y, \hat{y}) = (1 - y\hat{y})^2$
    - **hinge loss:**  $l(y, \hat{y}) = \max(0, 1 - y\hat{y})$
  - while  $y \in \{-1, +1\}$ ,  $\hat{y}$  may not
  - hinge loss is used in support vector machine (SVM)

## Model Training (3/3)

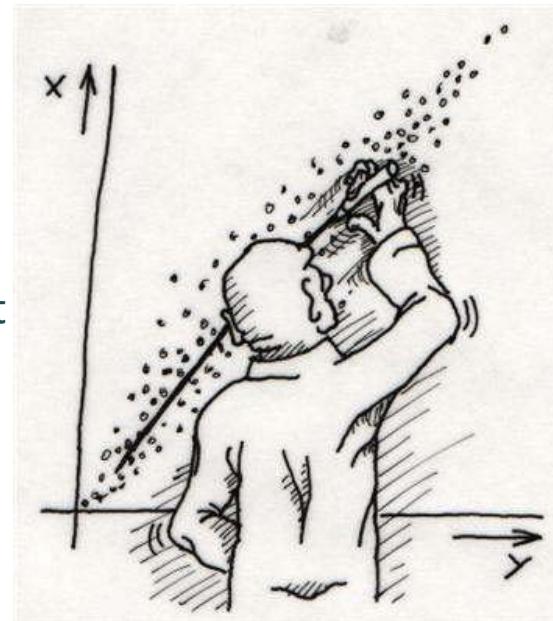
1. Collect data ( $y$ ) and extract features ( $x$ )
2. Build model: choose hypothesis class  $\mathcal{H}$  and loss function  $l$
3. **Optimization**: minimize the empirical loss
  - empirical loss:  $\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N l(y_i, f(x_i, \theta))$
  - optimization:  $\min_{\theta} \mathcal{L}(\theta) \Rightarrow$  by computing  $\frac{\partial \mathcal{L}(\theta)}{\partial \theta}$

## More on Loss Functions (1/4)

- Different loss function leads to different machine learning problems
  - **classification**:  $l(y, \hat{y}) = (1 - y\hat{y})^2$
  - **regression**:  $l(y, \hat{y}) = (y - \hat{y})^2$

Given  $N$  inputs  $(x_i, y_i)$ ,  $1 \leq i \leq N$ , where  $x_i$  is feature and  $y_i$  is the numerical value to be predicted, train a regression model  $f(\cdot)$  such that the **mean squared error (MSE)** is minimized

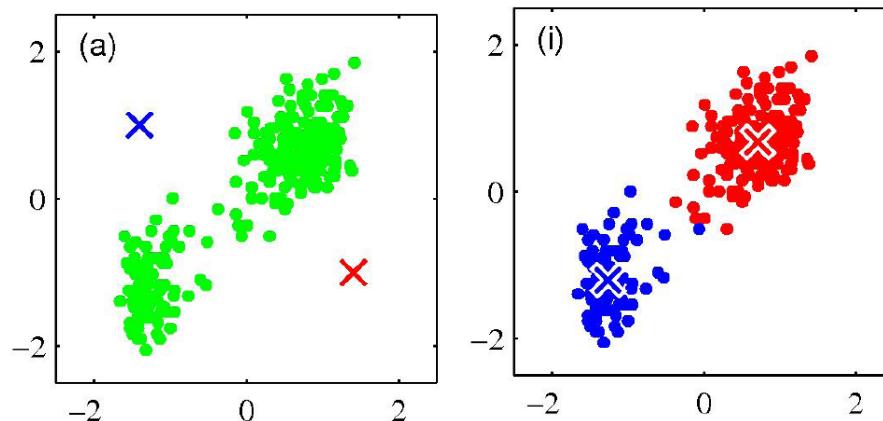
$$\min_f \frac{1}{N} \sum_{i=1}^N (y_i - f(x_i))^2$$



## More on Loss Functions (2/4)

- Different loss function leads to different machine learning problems
  - **classification**:  $l(y, \hat{y}) = (1 - y\hat{y})^2$
  - **regression**:  $l(y, \hat{y}) = (y - \hat{y})^2$
  - **clustering** (an unsupervised task)

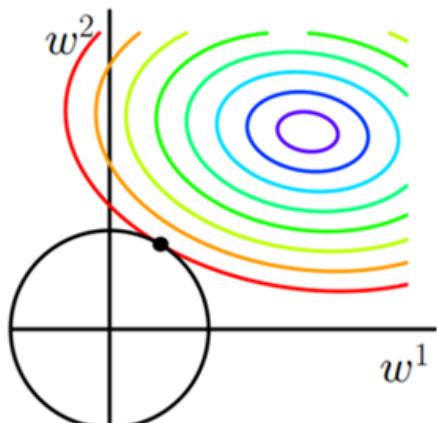
$$\mathcal{L}(\Delta) = \sum_{k=1}^K \sum_{i \in C_k} \|x_i - \mu_k\|^2$$



## More on Loss Functions (3/4)

- Adding regularizers

- original:  $\mathcal{L}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N (y_i - \mathbf{w}^T \mathbf{x}_i)^2$
- modified:  $\mathcal{L}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|_2^2$



(b)  $\ell_2$ -ball meets quadratic function.  
 $\ell_2$ -ball has no corner. It is very unlikely  
that the meet-point is on any of axes.

$$\|\mathbf{w}\|_2^2 = \mathbf{w}^T \mathbf{w} = \sum_{j=1}^M w_j^2$$

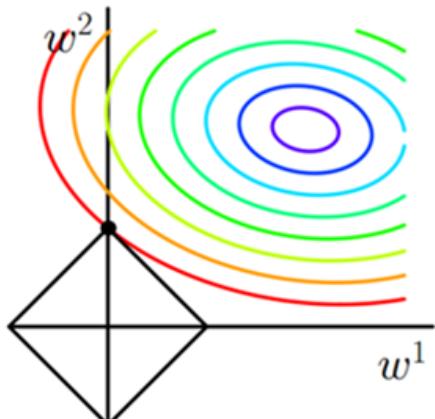
prefer  $\mathbf{w} = [1.5, -0.5, 0.8, -1.0]^T$   
over  $\mathbf{w} = [9.5, -8.0, 2.5, -3.0]^T$ ,  
despite that the latter has smaller MSE

- $\lambda \geq 0$  is a parameter to be tuned (i.e. empirically determined)

# More on Loss Functions (4/4)

- Adding regularizers

- modified:  $\mathcal{L}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|_2^2$
- modified:  $\mathcal{L}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \underline{\lambda \|\mathbf{w}\|_1}$



$$\|\mathbf{w}\|_1 = \sum_j |w_j|$$

prefer  $\mathbf{w} = [1.5, 0, 0, -2.0]^T$   
over  $\mathbf{w} = [1.5, -0.5, 0.8, -1.0]^T$ ,  
despite that the latter has smaller MSE

- L1 is the only norm that is both convex and sparsity promoting

## Example: Ridge Regression (for Regression)

- Loss function

$$\mathcal{L}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|_2^2$$

$$\text{or, } \mathcal{L}(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}) + \lambda \mathbf{w}^T \mathbf{w}$$

$\downarrow \quad \downarrow \quad \downarrow$   
 $N \times 1 \quad N \times M \quad M \times 1$

- Optimization

$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}} = 2\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + 2\lambda \mathbf{w} = \mathbf{0}$$

let  
 $M \times N \quad N \times 1 \quad M \times 1$

$$\therefore \mathbf{X}^T \mathbf{y} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}) \mathbf{w}$$

$$\therefore \mathbf{w} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$$

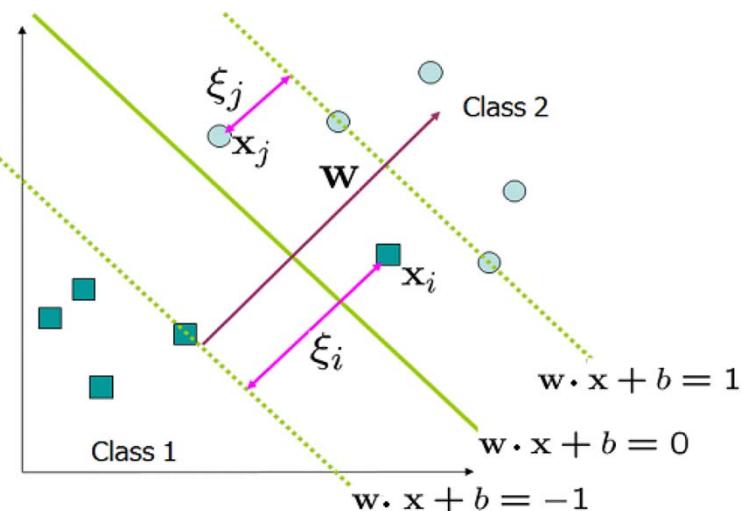
# Example: Support Vector Machine (1/3)

- Loss function

$$\min_{\mathbf{w}, \xi_i} \|\mathbf{w}\|_2 + C \sum_{i=1}^N \xi_i$$

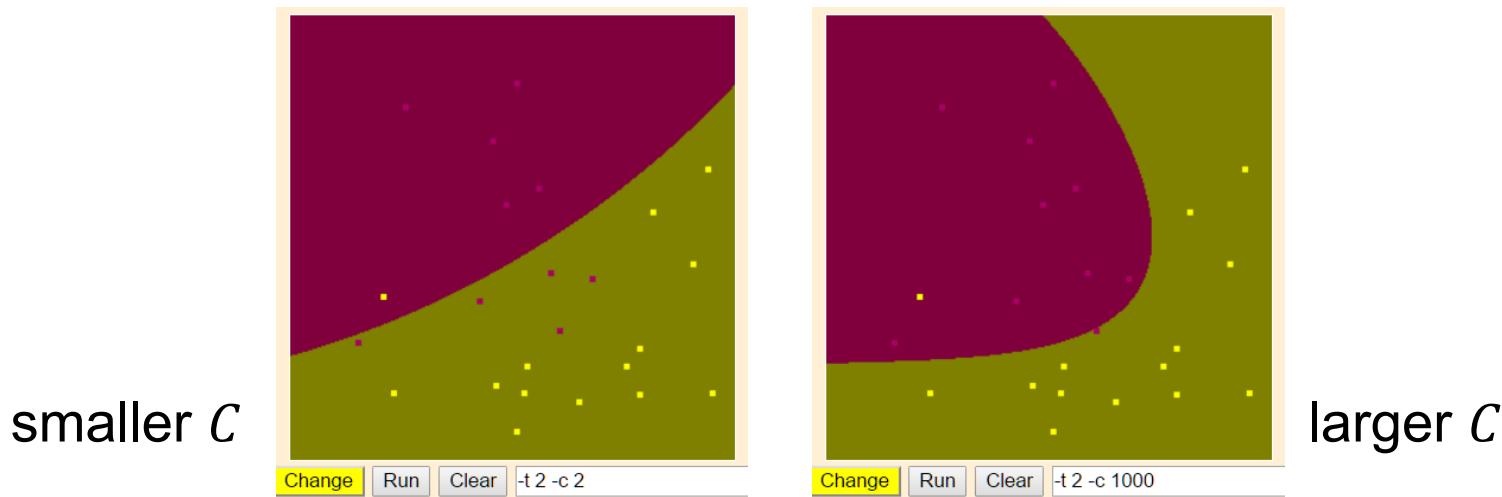
subject to  $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$  for  $i = 1 \dots N$

- the idea of hinge loss  
 $\max(0, 1 - y\hat{y})$  is also to have  $y\hat{y} \geq 1$
- the slack variables  $\xi_i \geq 0$  allow for errors



## Example: Support Vector Machine (2/3)

- Loss function:  $\min \|\mathbf{w}\|_2 + C \sum_{i=1}^N \xi_i$ 
  - smaller  $C$ : reduced model complexity
  - larger  $C$  : less misclassification
- Demo: <https://www.csie.ntu.edu.tw/~cjlin/libsvm/>



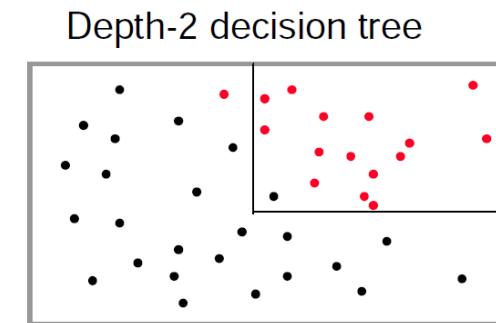
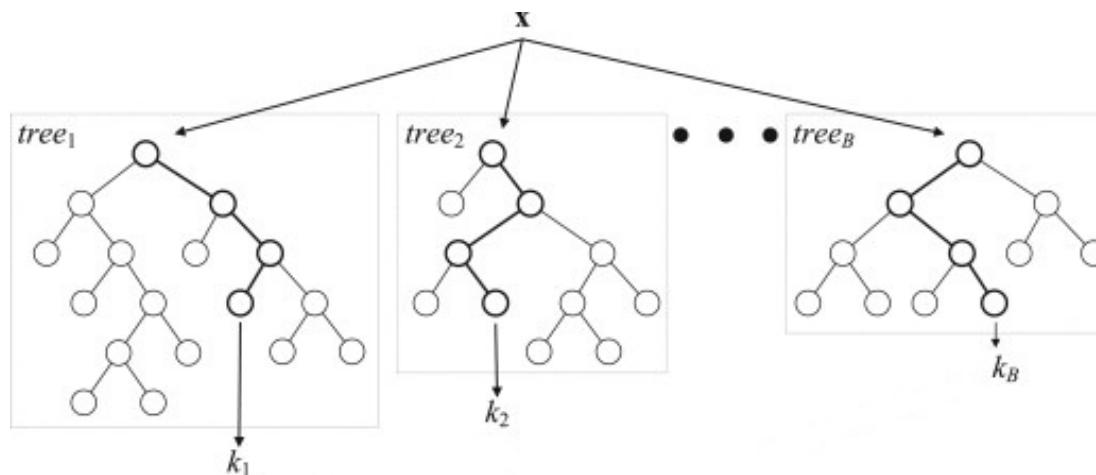
## Example: Support Vector Machine (3/3)

- Different kernel functions of SVM
  - the **RBF** kernel (-t 2) is often used for nonlinear SVM; important parameters: C and *gamma*

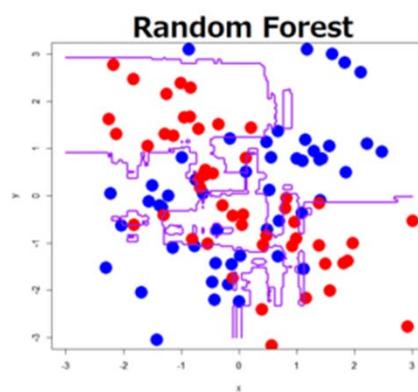
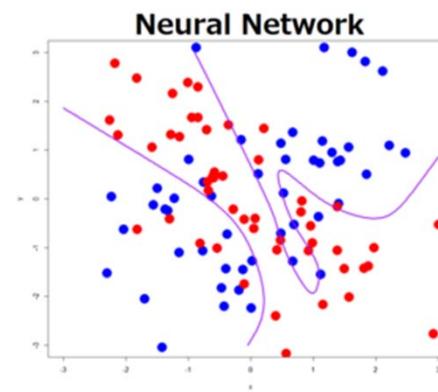
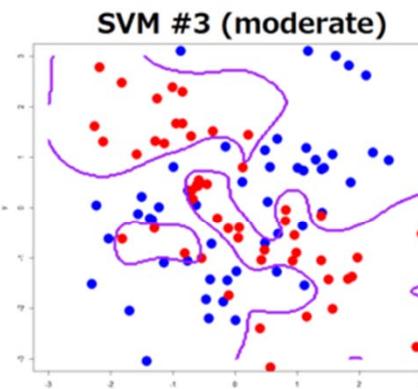
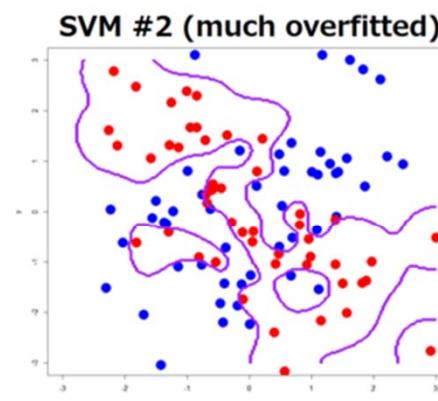
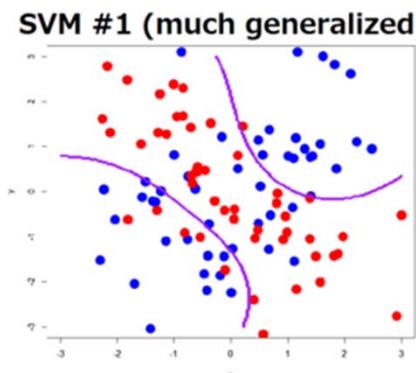
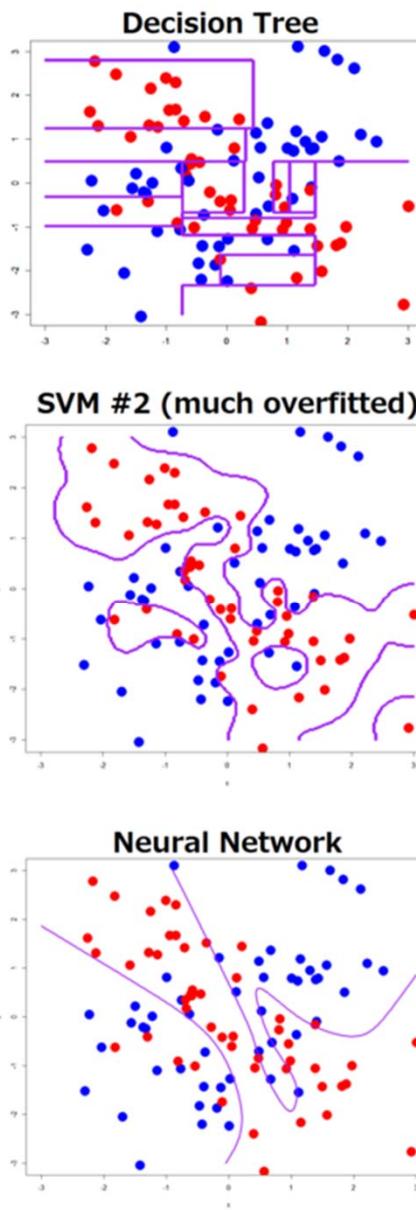
```
options:  
-s svm_type : set type of SVM (default 0)  
    0 -- C-SVC  
    1 -- nu-SVC  
    2 -- one-class SVM  
    3 -- epsilon-SVR  
    4 -- nu-SVR  
-t kernel_type : set type of kernel function (default 2)  
    0 -- linear: u'*v  
    1 -- polynomial: (gamma*u'*v + coef0)^degree  
    2 -- radial basis function: exp(-gamma*|u-v|^2)  
    3 -- sigmoid: tanh(gamma*u'*v + coef0)  
-d degree : set degree in kernel function (default 3)  
-g gamma : set gamma in kernel function (default 1/num_features)  
-r coef0 : set coef0 in kernel function (default 0)  
-c cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
```

# Example: Random Forest

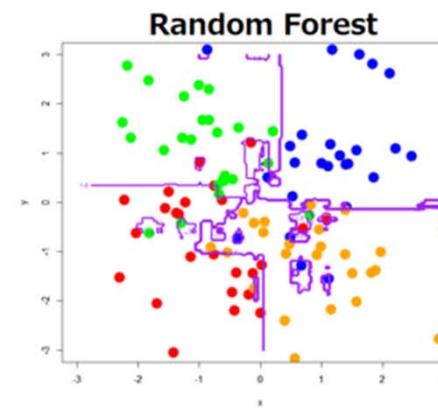
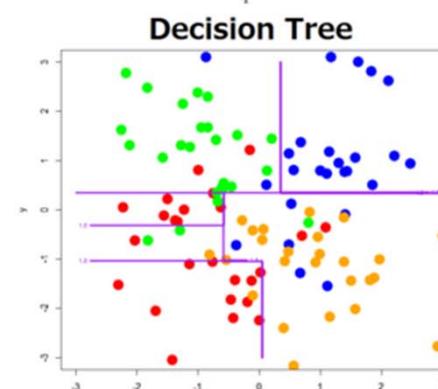
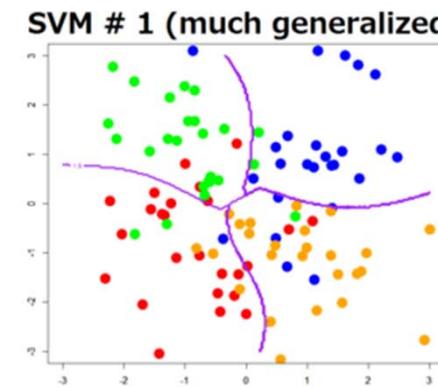
- Ensemble of decision trees
  - important parameters: number of trees, tree depth
  - strength: interpretability, non-linear



*binary classification*

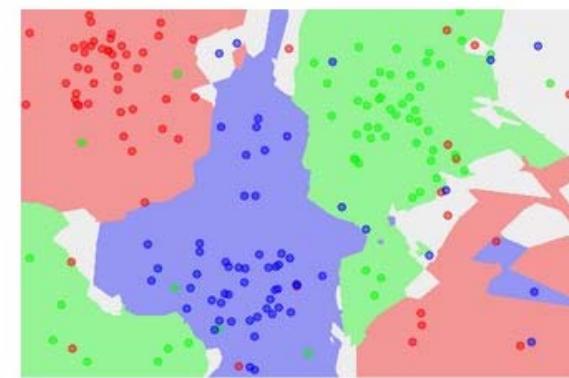
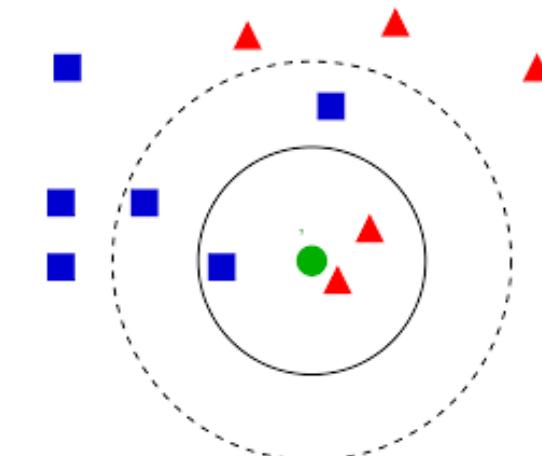


*multi-class classification*



## Example: k-Nearest Neighbors

- Classify an instance by a **majority vote** of the class membership of its  $k$  nearest neighbors
  - $k$  is a positive integer
  - performance depends on  $k$  and the features
  - lazy learning
  - nonlinear classifier



# Evaluation (1/3)

- Model training
  1. Collect data and extract features
  2. Build model: choose hypothesis class and loss function
  3. Optimization: minimize the empirical loss
- Evaluation
  4. Evaluate on unseen, testing data

## Evaluation (2/3)

- Performance measures
  - classification accuracy (CA)
  - per-class precision (P), recall (R), F-score (F)
  - confusion table

		Predicted		
		Cat	Dog	Rabbit
Actual class	Cat	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

$$CA = (5+3+11)/(5+3+0+2+3+1+0+2+11)$$

$$P(cat) = 5/(5+2)$$

$$R(cat) = 5/(5+3)$$

# Evaluation (3/3)

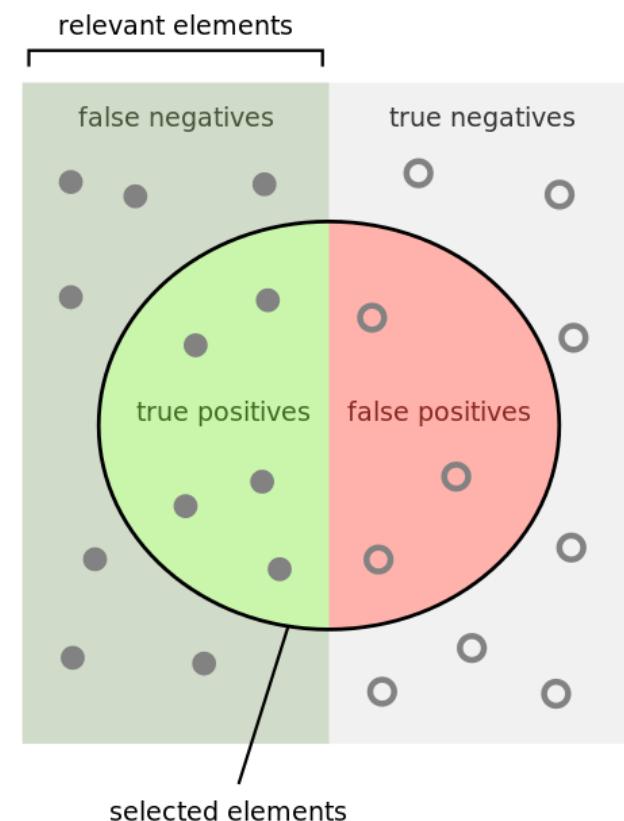
- Performance measures
  - precision (**P**):  $TP/(TP+FP)$
  - recall (**R**):  $TP/(TP+FN)$
  - F-score (**F**):  $2PR/(P+R)$

How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{selected elements}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$



# Train, Validation, Test (1/2)

- Test set is unseen, how to evaluate our classifier?
  - use part of the training set as the “**validation set**”
  - e.g. 80% for training, 20% for validation
  - use the validation set to: 1) tune the parameters, 2) select classification algorithms, and 3) select features
- **K-fold Cross validation**
  - divide the data into K folds, use one fold for validation and the rest of training, repeat K times and report the average result

## Train, Validation, Test (2/2)

- **Model overfitting**
  - training accuracy  $\neq$  validation accuracy
  - that's why we need a balance between model complexity and training error
- **Cross-dataset generalizability**
  - validation accuracy  $\neq$  test accuracy

# Data Labeling (1/2)

- Collect data and extract features
- How to get the labels?
  - **existing datasets**
    - due to copyright restrictions, it's not easy to get the audio files
  - **crawl the web**
    - unsure of the quality; do not know how the labeling was done
  - **manually annotate**
    - labor intensive and time consuming but can be worthwhile

## Data Labeling (2/2)

- Issues to be taken care of while building a dataset and while doing train/val/test splitting
  - 1. be aware of “confounds”**
    - Example 1: Jazz from the 60's VS contemporary Rock
    - Example 2: *song-level* split or chunk-level split?
  - 2. try to make the annotation process easier**
    - binary decision -> multiple choices -> rating
    - gamification
  - 3. clear instructions**
  - 4. check for reliability/consistency**

# Feature Extraction and Processing

- Collect data and extract features
- The usual pipeline
  - frame-level **feature extraction**
  - **pooling** (from frame-level to clip-level)
  - **feature normalization**
- Classification tasks in MIR
  - **clip-level prediction**: genre classification
  - **frame-level prediction**: onset detection, pitch detection
  - for clip-level prediction, pooling is often used

# Feature Pooling

- Rationale
  - thousands of frames per clip → thousands of feature vectors per clip
  - we can either train a frame-level classifier, but eventually still have to integrate the result to the clip-level
  - or, we can integrate the feature vectors before training the classifier
- Approach
  - take statistics such as **mean** and **standard deviation**

# Feature Normalization/Scaling

- For each individual feature dimension
  - **linear normalization:**  $x' = \frac{x - \min(x)}{\max(x) - \min(x)}$
  - **z-score normalization:**  $x' = \frac{x - \text{mean}(x)}{\text{std}(x)}$
  - the values  $\text{mean}(x)$  and  $\text{std}(x)$  are computed from all the *training* instances (i.e. *not* including the validation and the test sets)
- For each individual instance
  - **scaling to unit length:**  $x' = \frac{x}{|x|}$

## Early Fusion or Late Fusion

- You can extract features of different types to characterize different aspects of the signal
  - timbre, pitch, rhythm, lyrics, etc
- **Early fusion** (feature-level fusion)
  - concatenate them into a single feature vector then train a classifier
- **Late fusion** (decision-level fusion)
  - train a classifier for each feature type, and then aggregate the estimate of these classifiers (e.g., majority vote or average) to make the final decision

# Feature Selection

- Not all features would be relevant
- Feature selection according to
  - domain knowledge
  - empirical observation at the feature matrix (e.g., remove a feature if there is NaN or outlier [or replace the NaNs by the average])
  - statistical methods

# Library: Torchaudio

[https://pytorch.org/audio/0.11.0/tutorials/audio\\_feature\\_extractions\\_tutorial.html](https://pytorch.org/audio/0.11.0/tutorials/audio_feature_extractions_tutorial.html)

```
waveform, sample_rate = get_speech_sample()

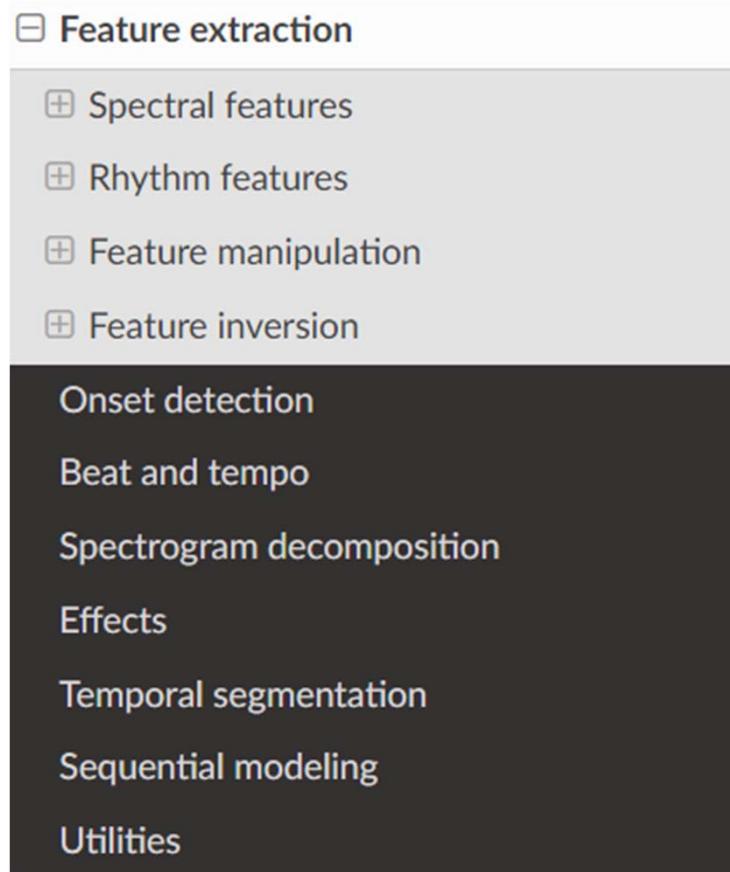
n_fft = 1024
win_length = None
hop_length = 512

# define transformation
spectrogram = T.Spectrogram(
    n_fft=n_fft,
    win_length=win_length,
    hop_length=hop_length,
    center=True,
    pad_mode="reflect",
    power=2.0,
)
# Perform transformation
spec = spectrogram(waveform)

print_stats(spec)
plot_spectrogram(spec[0], title="torchaudio")
```

# Python: Librosa

<https://librosa.org/doc/main/feature.html>



# Python: scikit-learn

<https://scikit-learn.org/1.4/tutorial/index.html>

The screenshot shows the official scikit-learn documentation website. At the top, there is a navigation bar with links for 'Install', 'User Guide', 'API', 'Examples', 'Community', and 'More'. Below the navigation bar, there is a sidebar with links for 'Prev', 'Up', and 'Next' (navigation), and 'scikit-learn 1.4.2' (the current version) and 'Other versions' (link to other releases). A yellow box in the sidebar encourages users to 'cite us' if they use the software. The main content area features a red box containing a message about the old release (version 1.4). The title 'scikit-learn Tutorials' is prominently displayed. Below it, a section titled 'An introduction to machine learning with scikit-learn' lists several topics: 'Machine learning: the problem setting', 'Loading an example dataset', 'Learning and predicting', and 'Conventions'.

This is documentation for an old release of Scikit-learn (version 1.4). Try the [latest stable release](#) (version 1.5) or [development](#) (unstable) versions.

## scikit-learn Tutorials

### An introduction to machine learning with scikit-learn

- Machine learning: the problem setting
- Loading an example dataset
- Learning and predicting
- Conventions

# Practical Issues 1: Multi-label Classification

- A data instance can be labeled with more than one classes
- *Multi-class* classification (one-hot; one out of many) vs. *multi-label* classification (multi-hot)
- Harder for ML (may need multiple binary classifiers) but simple for DL: it's just a matter of changing the loss function
  - Multi-class classification: **categorical cross entropy** (CE)
  - Multi-label classification: **binary cross entropy** (BCE)

## Practical Issues 2: Data Imbalance

- Simply predicting everything as the majority class would already give you good accuracy
- A naïve workaround: **subsampling the majority class**

## Practical ML Hints (not for DL)

- Look into the features
- Remember to normalize the features
- Use validation set to properly tune the parameters
- Nonlinear classifier usually performs better, but also requires longer training time
- Use a subset of data to fast prototype your ideas first, before using the full set for more complete experiments
- Study the classification result (e.g., confusion table)