2024 edition

Deep Learning for Music Analysis and Generation

Introduction

Course

Yi-Hsuan Yang Ph.D.
yhyangtw@ntu.edu.tw

Course Website

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

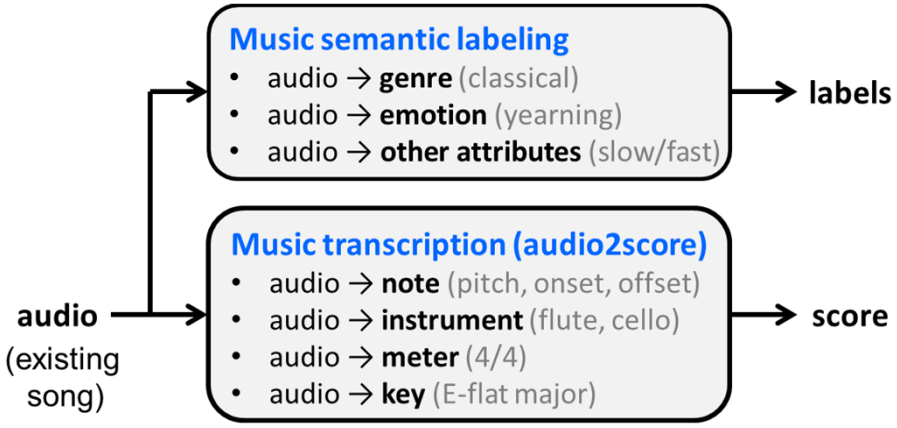
https://affige.github.io/teaching.html

Outline

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

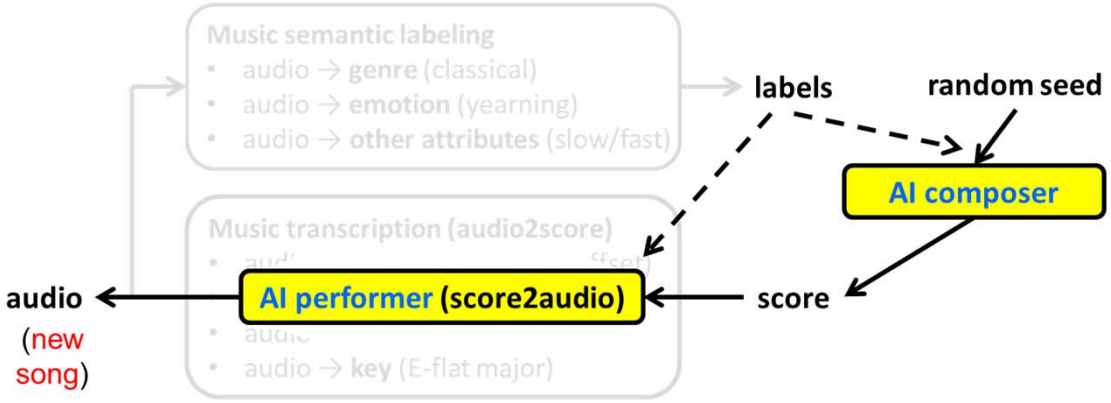
Music AI; or *Music Information Research* (MIR)

- **Music analysis**



  - music understanding
  - music search
  - music recommendation

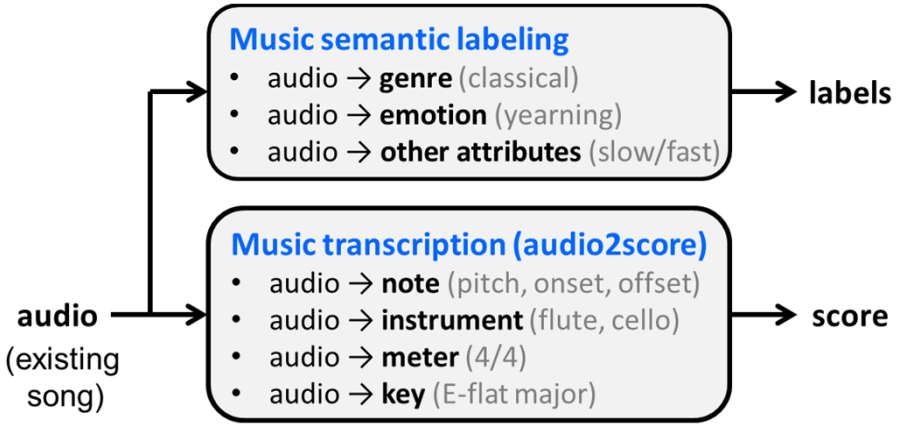
- **Music generation**



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

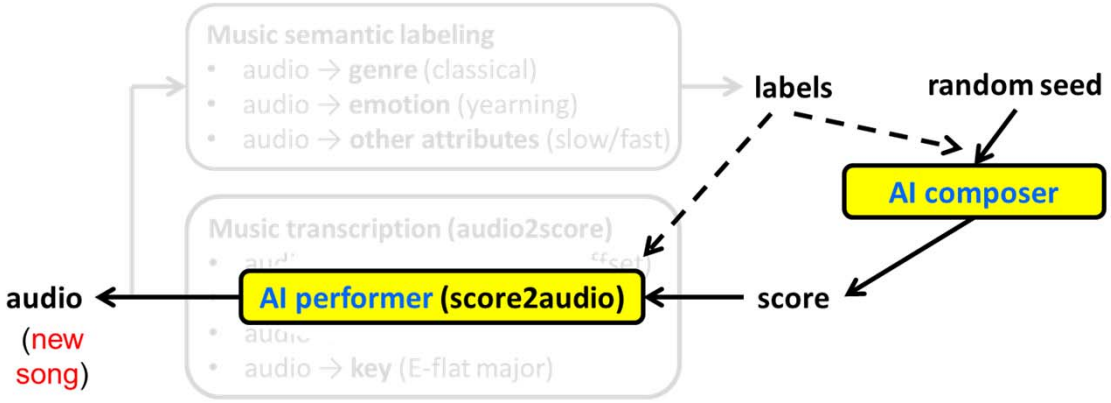
Music AI: {signal processing, machine learning} + music

- **Music analysis**



  - music understanding
  - music search
  - music recommendation

- **Music generation**



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

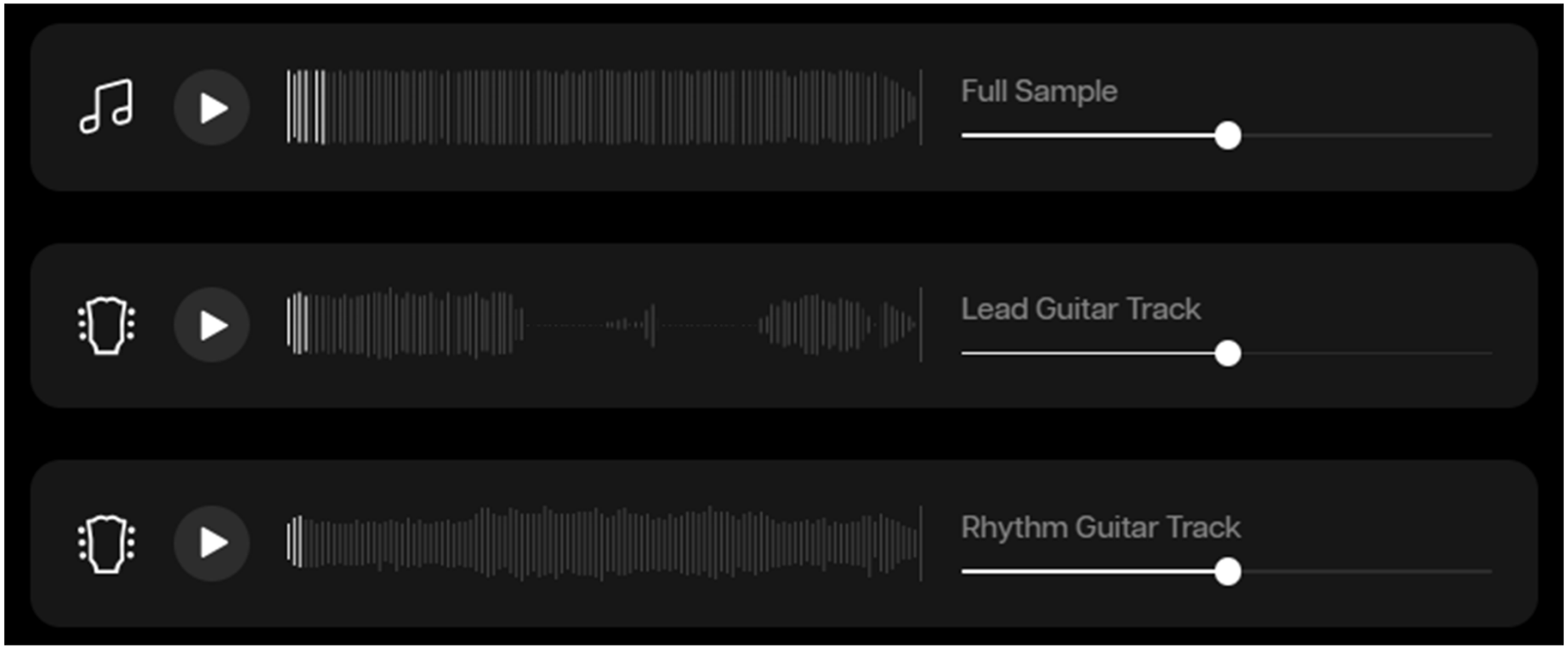
Music Analysis Demo: Source Separation

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

https://www.youtube.com/watch?v=r4NcZbgtIgw

Music Analysis Demo: Source Separation

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



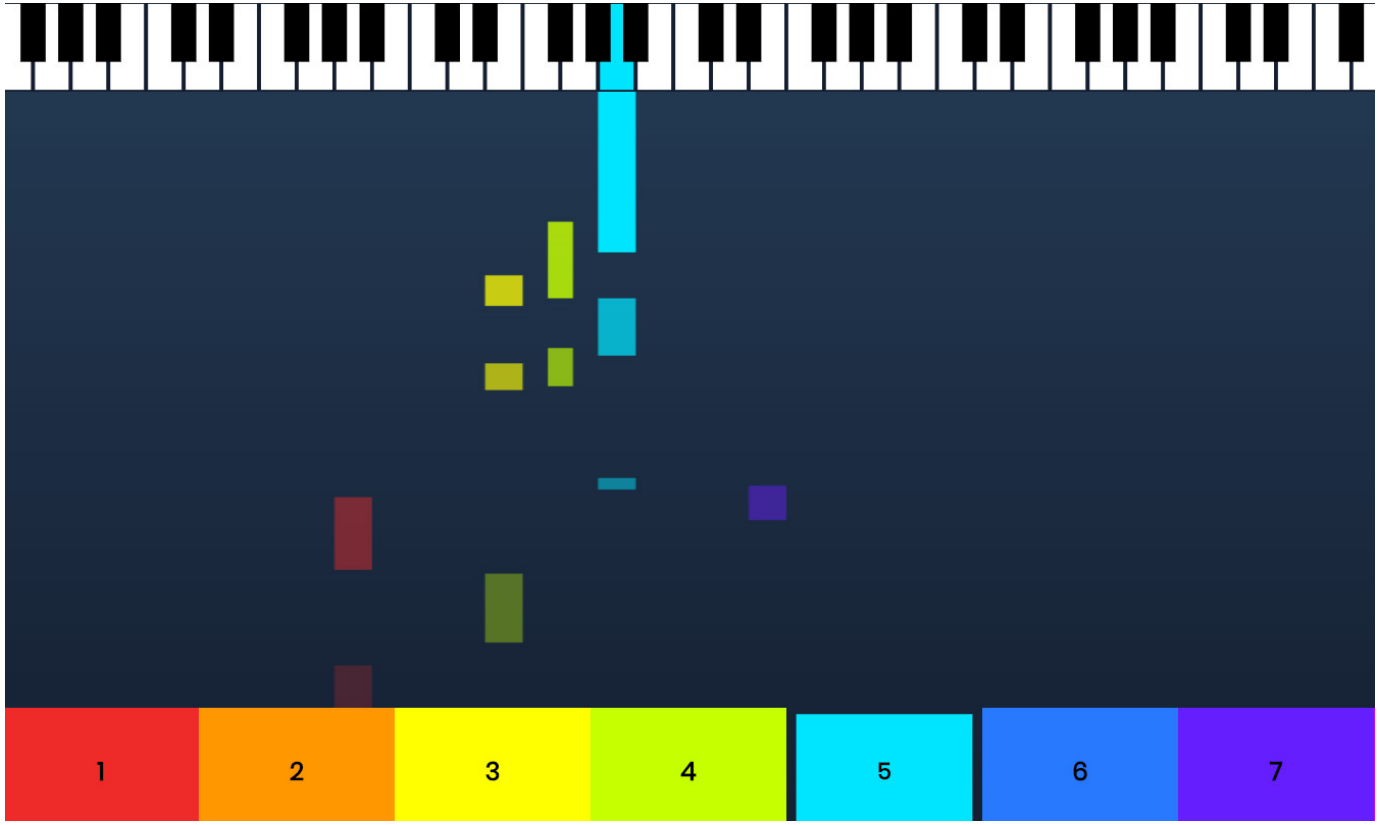
Positive Use Cases of Music Generation

- **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 昇華到情感面，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。

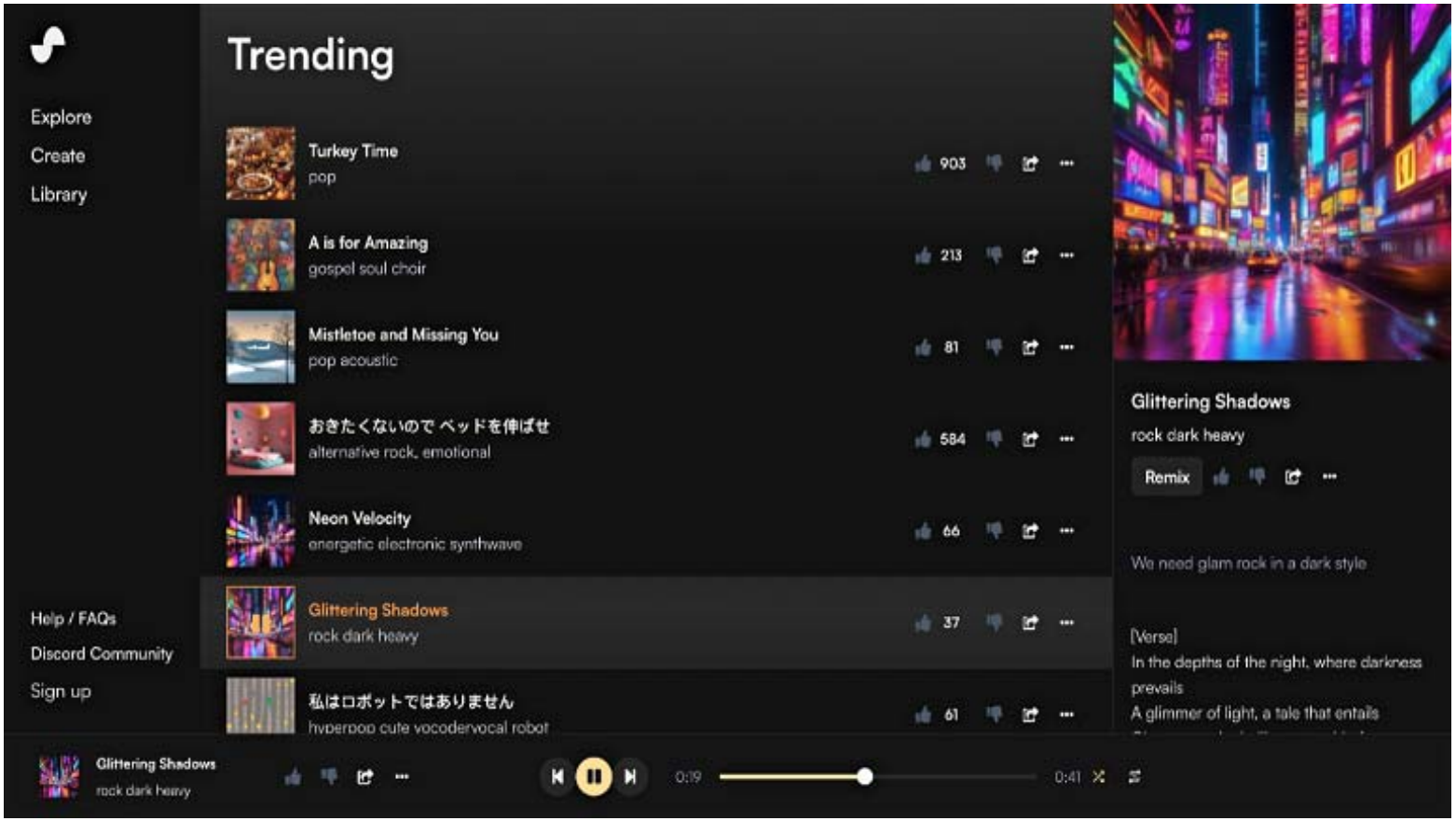
透過聆聽《教我如何做你的愛人》，試著探討：「若 AI 已經能模擬原唱的一切，那麼原唱歌手的價值會是什麼？」

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

Music Generation Demo 5: Text-to-Music

(Create copyright free music for videos or games)

https://www.youtube.com/watch?v=R4oY1A9_Jss

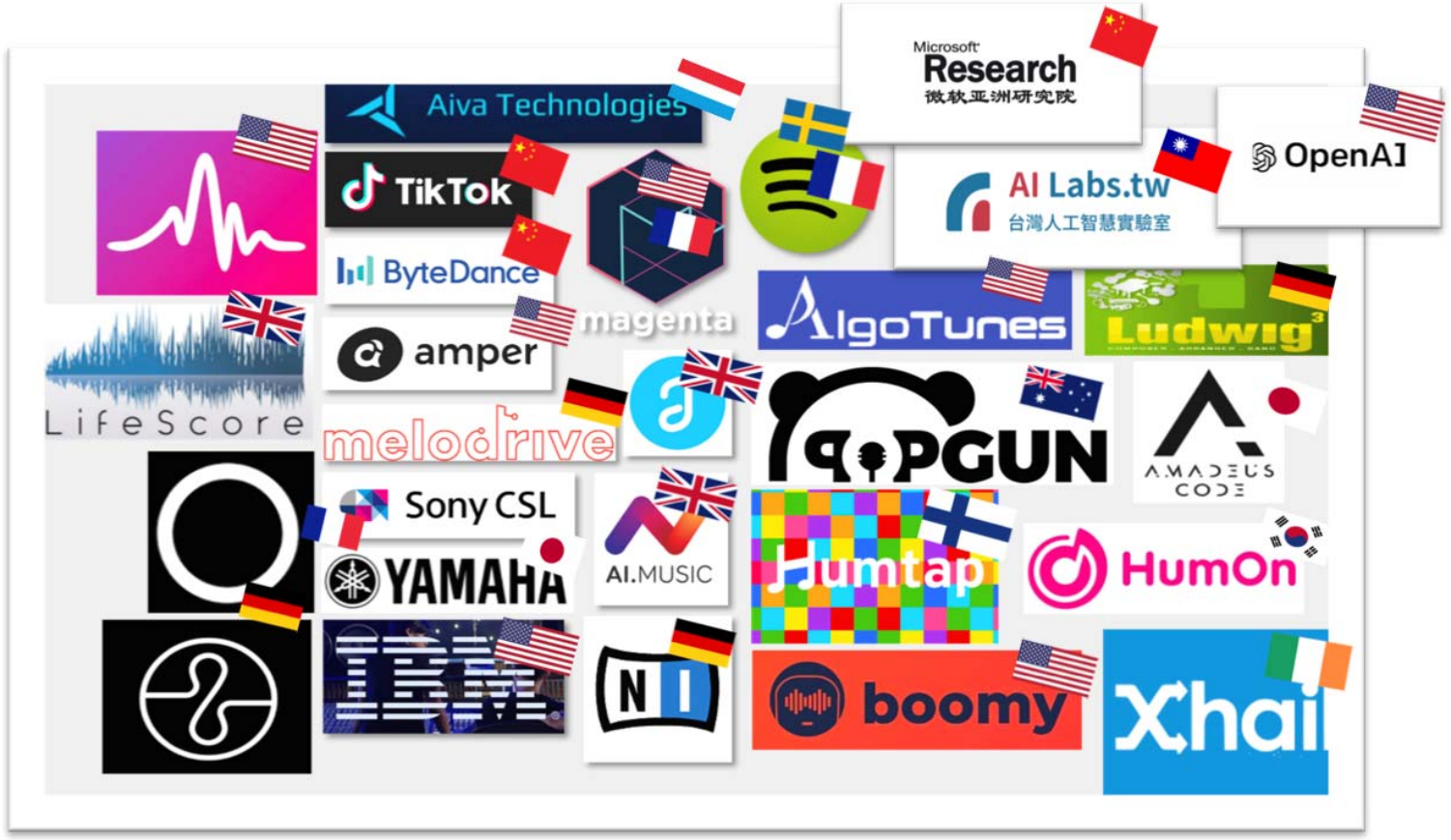


Music Generation Demo 5: Text-to-Music

(Create copyright free music for videos or games)

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

Global Interest in Music AI: Industry

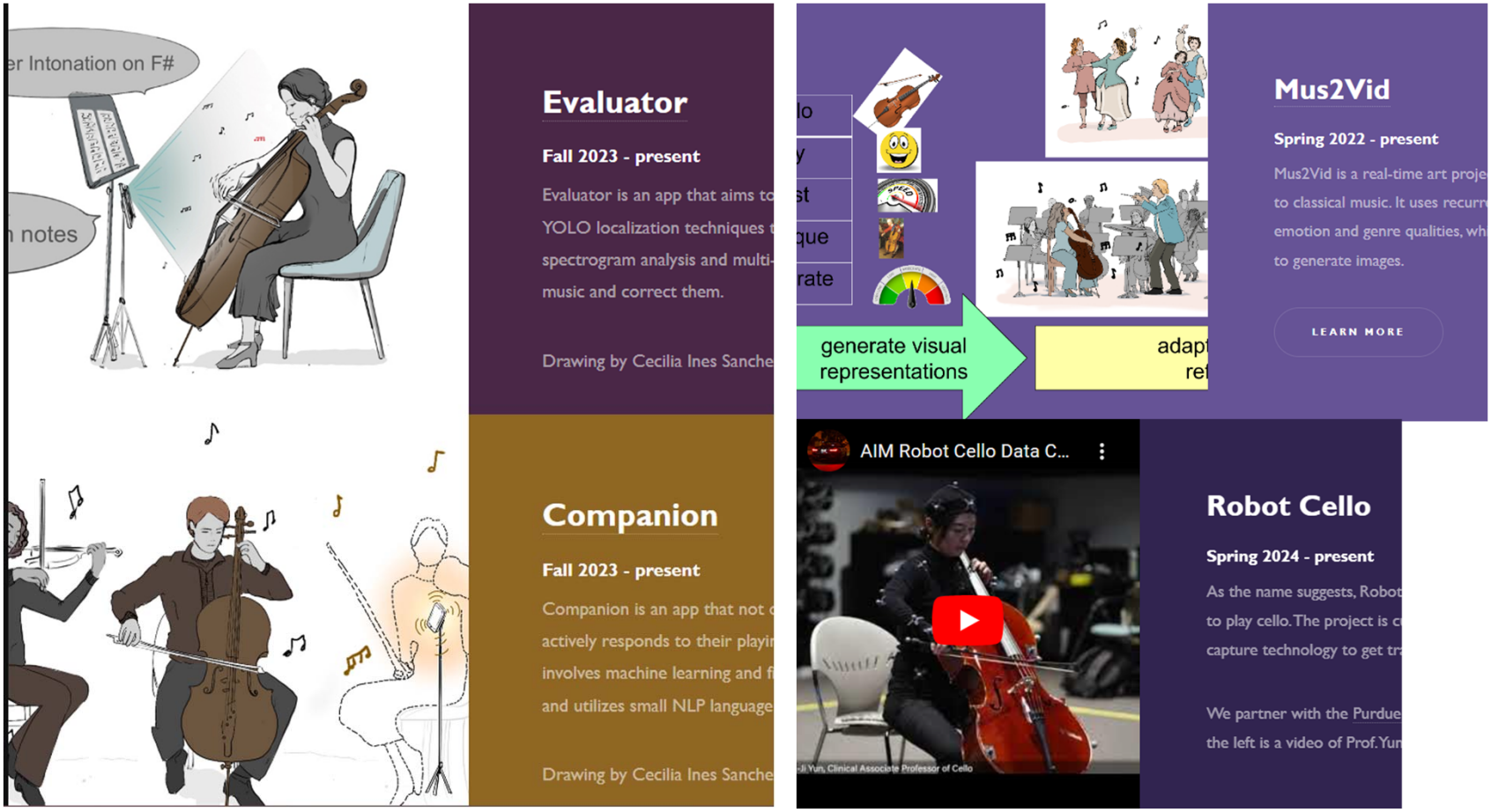


(**Slide from Rujing Huang, Bob L. T. Sturm, and Andre Holzapfel**, "De-centering the West: East Asian Philosophies and the Ethics of Applying Artificial Intelligence to Music," ISMIR 2021)

Global Interest in Music AI:

Purdue Music Technology Research Group

https://ai4musicians.org/



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

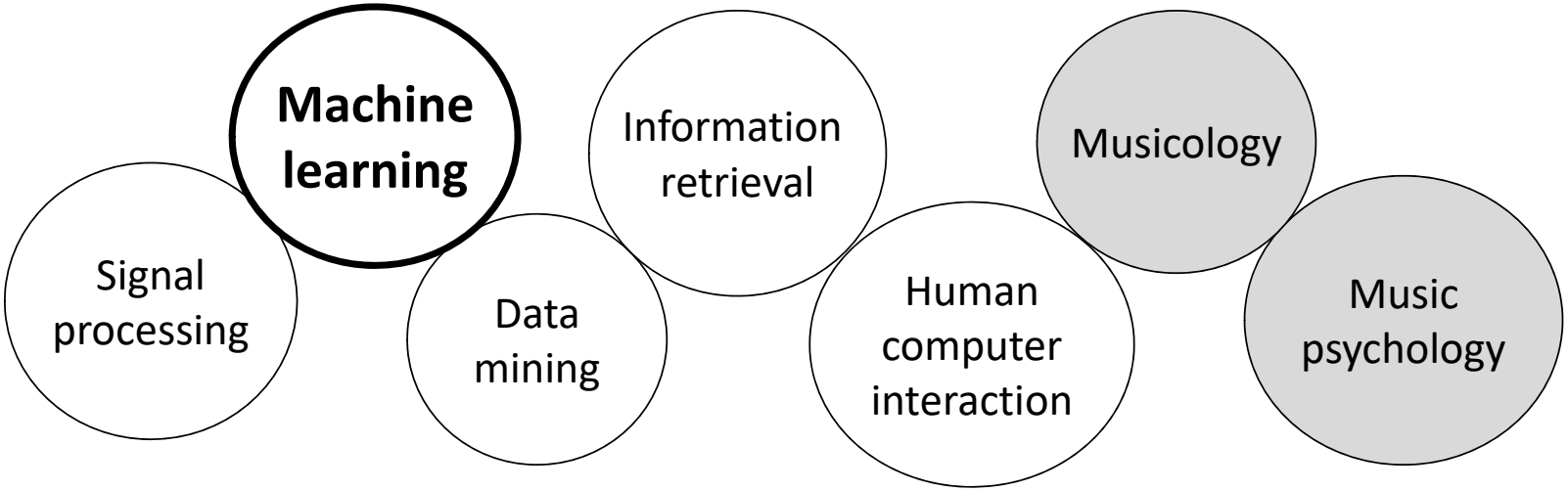
This Course: Wills and Won’ts

- 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

- 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)

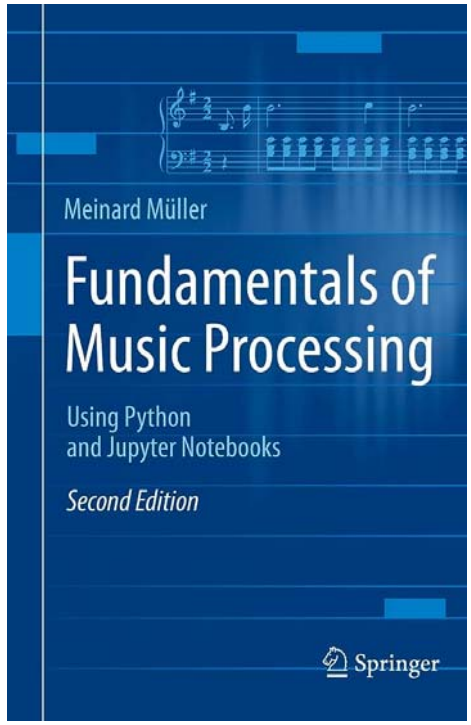
Textbook

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

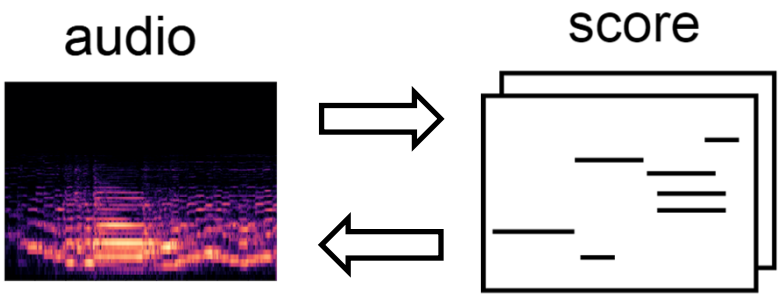
Part	Title	Notions, Techniques & Algorithms	HTML	IPYNB
4	Music Structure Analysis	Similarity matrix, repetition, thumbnail, homogeneity, novelty, evaluation, precision, recall, F-measure, visualization, scape plot	[html]	[ipynb]
5	Chord Recognition	Harmony, music theory, chords, scales, templates, hidden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

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

NO Cheating

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

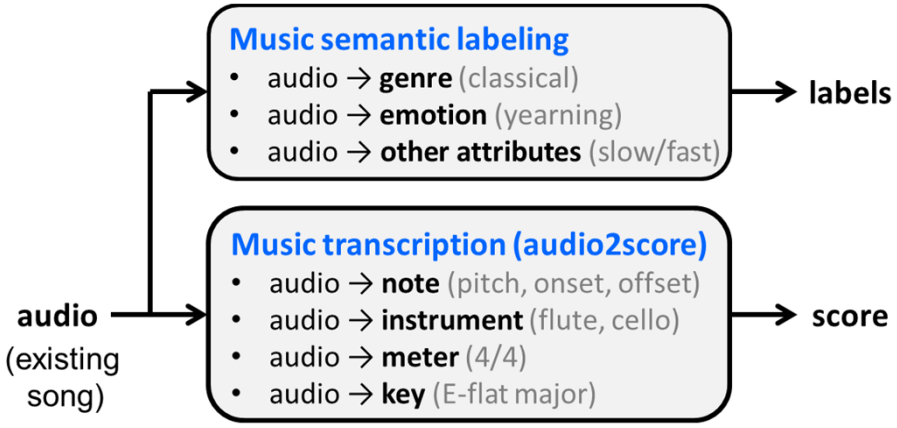
Final Project

- 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

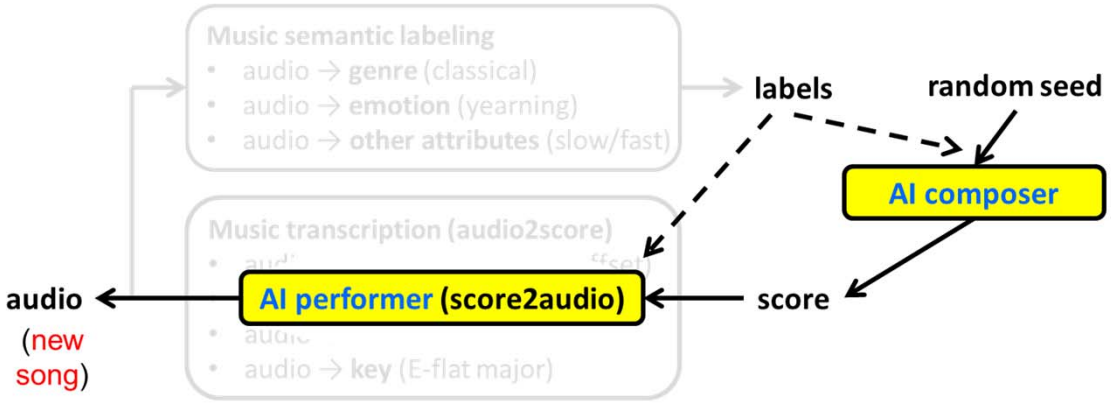
Final Project

- **Music analysis**



  - music semantic labeling
  - music transcription
  - source separation

- **Music generation**



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

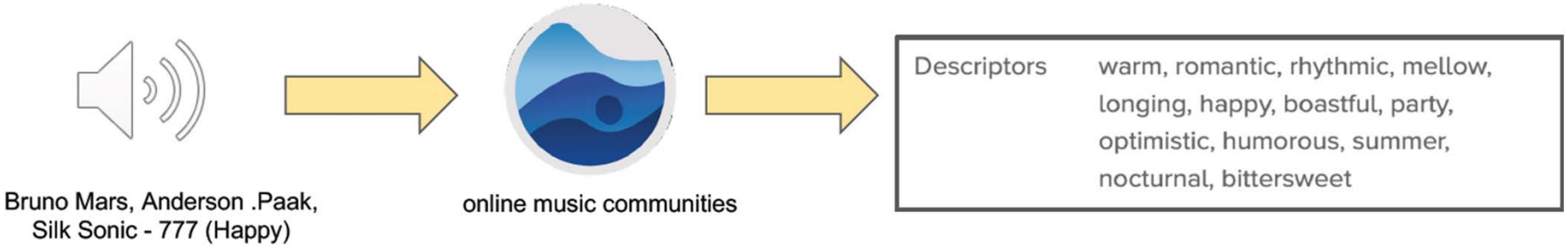
Final Project Showcase

https://affige.github.io/teaching_deepmir23.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

1

Final Project Showcase

- Guitar effect removal

  https://y10ab1.github.io/guitar_effect_removal/

Final Project Showcase

- Transfer music to failed recorder style

source

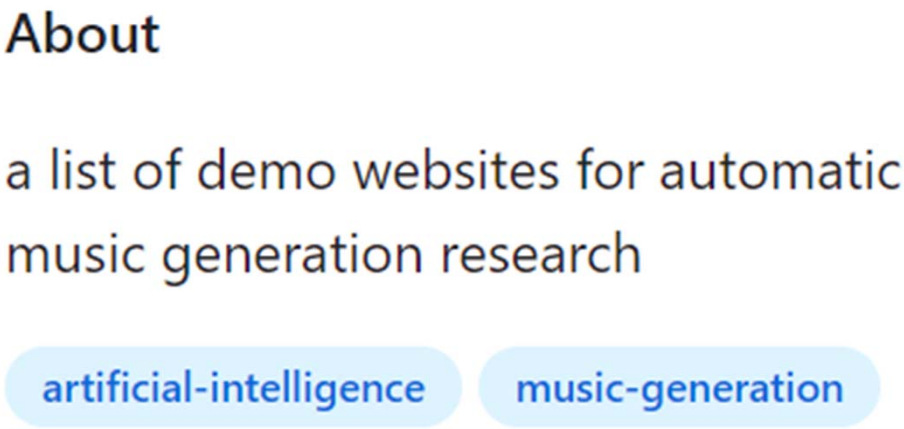
target style

stargan

vaegan

genmusic_demo_list

https://github.com/affige/genmusic_demo_list



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://musicinformationretrieval.com/
  - https://mac.kaist.ac.kr/~juhan/gct634/index.html
  - http://www.jordipons.me/apps/teaching-materials/
  - https://www.upf.edu/web/smc/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

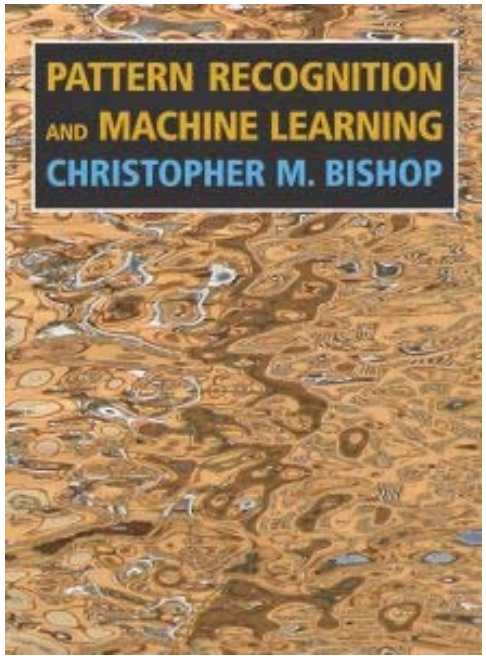
https://nol2.aca.ntu.edu.tw/nol/coursesearch/print_table.php?course_id=921%20U2620&class=02&dpt_code=9210&ser_no=40152&semester=113-1&lang=CH

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

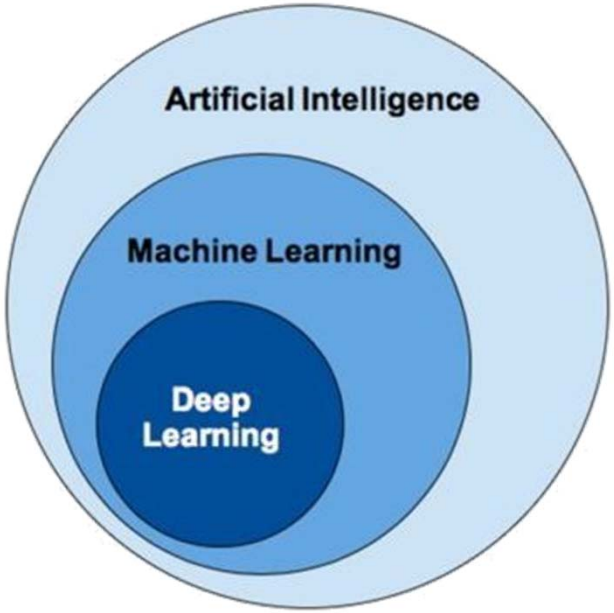
Machine Learning

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

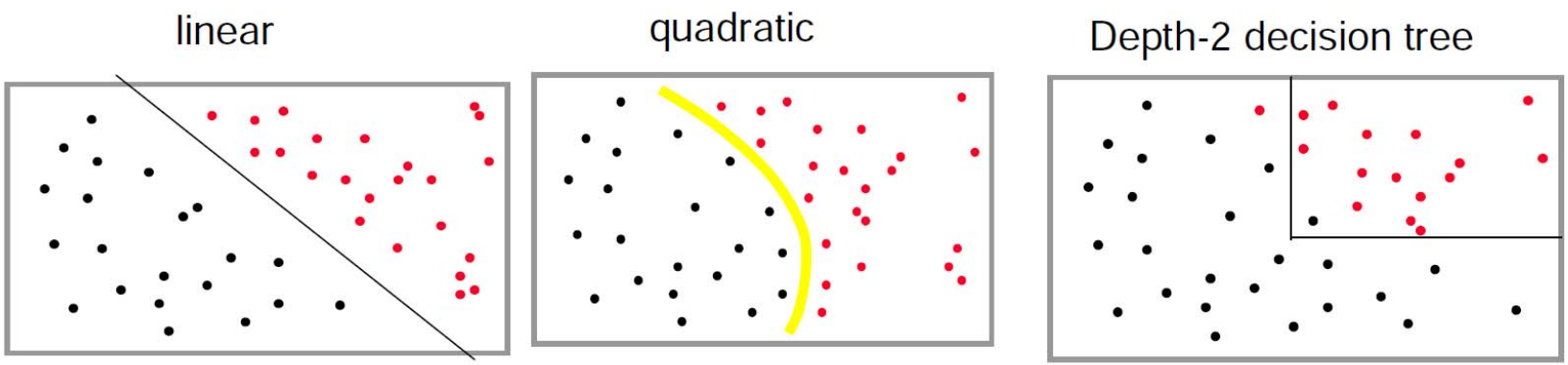


Freely available

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



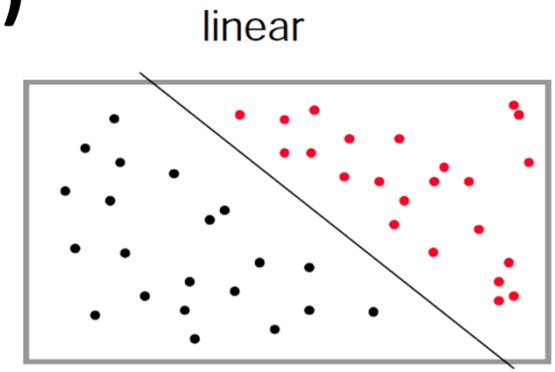
Classification: Overview (1/2)

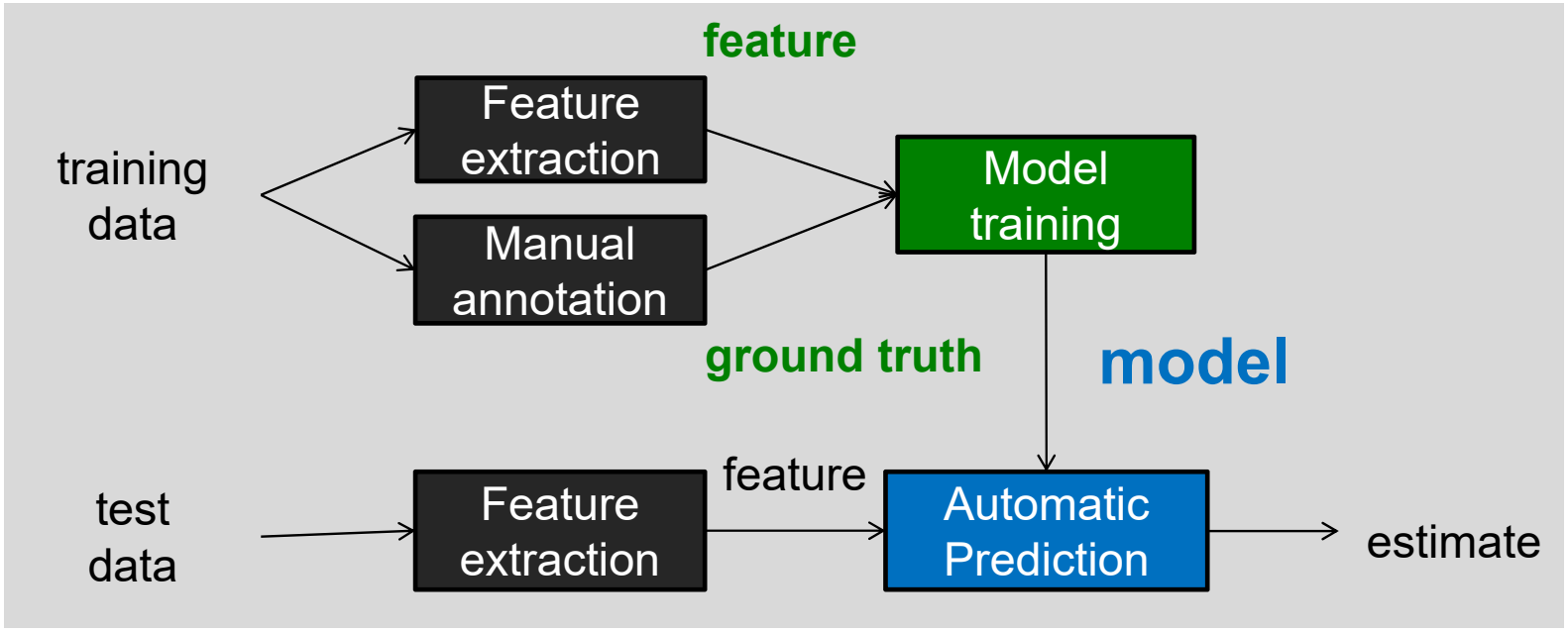


- Given: $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, yN)\}$
  - $\boldsymbol{x}_i \in \mathbb{R}^M$: feature representation of a data instance
  - $y_i \in \{-1, +1\}$: class label of that instance
- Goal
  - learn a separating hyperplane

Classification: Overview (2/2)

- Given: $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, yN)\}$
  - $\boldsymbol{x}_i \in \mathbb{R}^M$: feature representation
  - $y_i \in \{-1, +1\}$: class label





Model Training (1/3)

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

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

   - "parameters" to be *learned*: $\theta = \{\boldsymbol{A}, \boldsymbol{w}, b\}$

Model Training (2/3)

1. Collect data (y) and extract features ($\boldsymbol{x}$)
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)

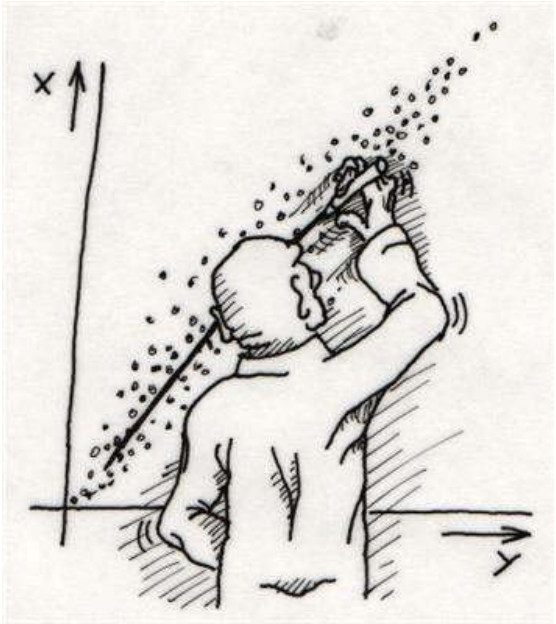
1. Collect data (y) and extract features ($\boldsymbol{x}$)
2. Build model: choose hypothesis class $\mathcal{H}$ and loss function l
3. Optimization: minimize the empirical loss
   - empirical loss: $\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} l(yi, f(\boldsymbol{x}_i, \theta))$
   - optimization: $\min_{\theta} \mathcal{L}(\theta) \Rightarrow$ 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 $(\boldsymbol{x}_i, y_i)$, $1 \leq i \leq N$, where $\boldsymbol{x}_i$ is feature and y_i is the numerical value to be predicted, train a regression model $f(.)$ such that the **mean squared error** (**MSE**) is minimized

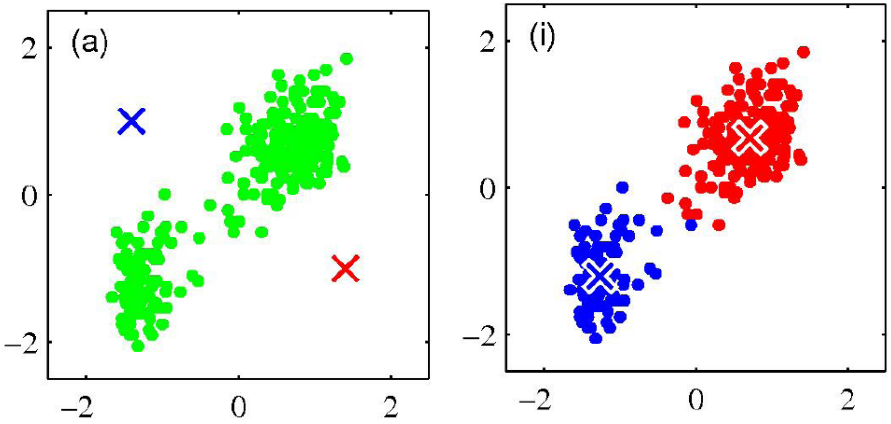
$$\min_f \frac{1}{N} \sum_{i=1}^{N} (y_i - f(\boldsymbol{x}_i))^2$$



More on Loss Functions (2/4)

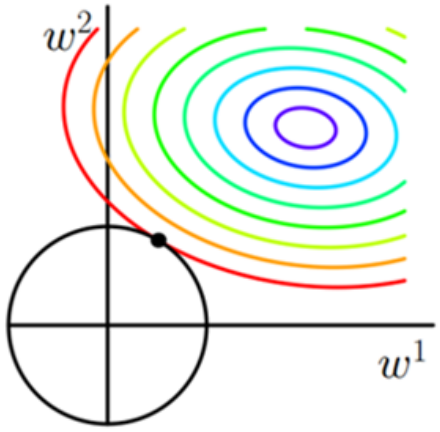
- Different loss function leads to different machine learning problems
  - **classification**: $l(y, \hat{y}) = (1 - y\hat{y})^2$
  - **regression**: $l(y, \hat{y}) = (y - \hat{y})^2$
  - **clustering** (an unsupervised task)

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



More on Loss Functions (3/4)

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



(b) ℓ_2-ball meets quadratic function. ℓ_2-ball has no corner. It is very unlikely that the meet-point is on any of axes."

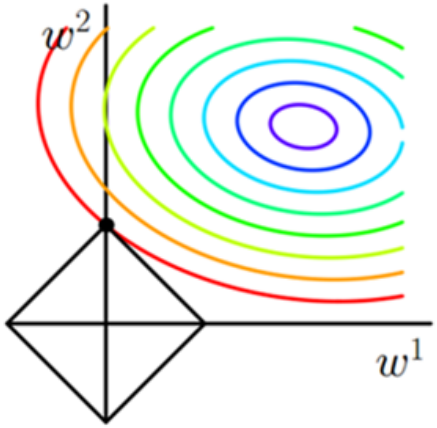
$$\|\boldsymbol{w}\|_2^2 = \boldsymbol{w}^T\boldsymbol{w} = \sum_{j=1}^{M} w_j^2$$

prefer $\boldsymbol{w}$ = [1.5, -0.5, 0.8, -1.0]^T
over $\boldsymbol{w}$ = [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: $\mathcal{L}(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \boldsymbol{w}^T\boldsymbol{x}_i)^2 + \lambda\|\boldsymbol{w}\|_2^2$
  - modified: $\mathcal{L}(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \boldsymbol{w}^T\boldsymbol{x}_i)^2 + \lambda\|\boldsymbol{w}\|_1$



(a) ℓ_1-ball meets quadratic function. ℓ_1-ball has corners. It's very likely that the meet-point is at one of the corners.

$$\|\boldsymbol{w}\|_1 = \sum_j |wj|$$

prefer $\boldsymbol{w}$ = [1.5, 0, 0, -2.0]^T
over $\boldsymbol{w}$ = [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

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

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

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

- Optimization

$$\frac{\partial \mathcal{L}(\boldsymbol{w})}{\partial \boldsymbol{w}} = 2\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}) + 2\lambda\,\boldsymbol{w} \overset{\text{let}}{=} \boldsymbol{0}$$

$M \times N \quad N \times 1 \qquad M \times 1$

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

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

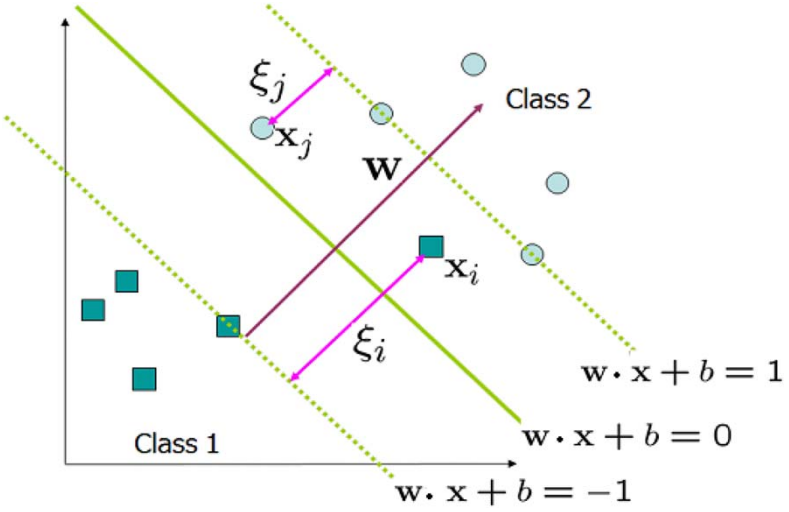
Example: Support Vector Machine (1/3)

- Loss function

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

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

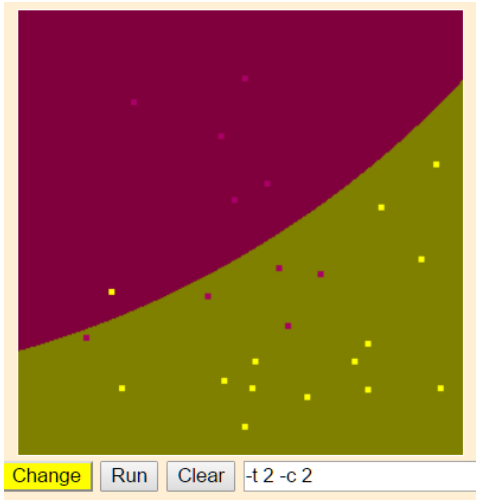
- the idea of hinge loss $\max(0, 1 - y\hat{y})$ is also to have $y\hat{y} \geq 1$
- the slack variables $\xi_i \geq 0$ allow for errors

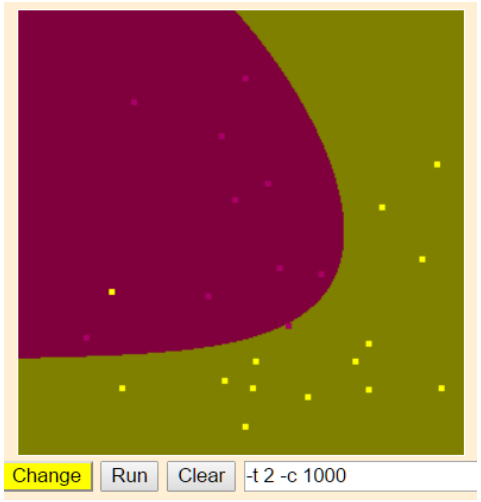


Example: Support Vector Machine (2/3)

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

smaller C





larger C

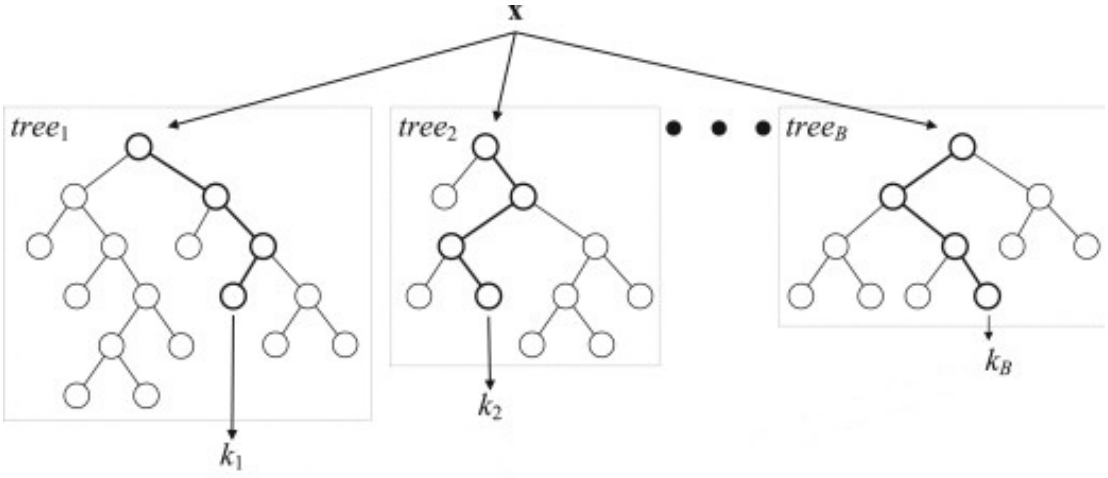
Example: Support Vector Machine (3/3)

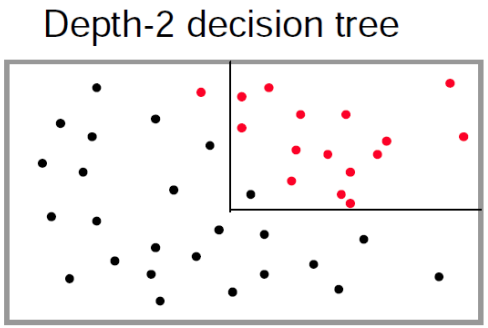
- Different kernel functions of SVM
  - the **RBF** kernel (-t 2) is often used for nonlinear SVM; important parameters: *C* and *gamma*

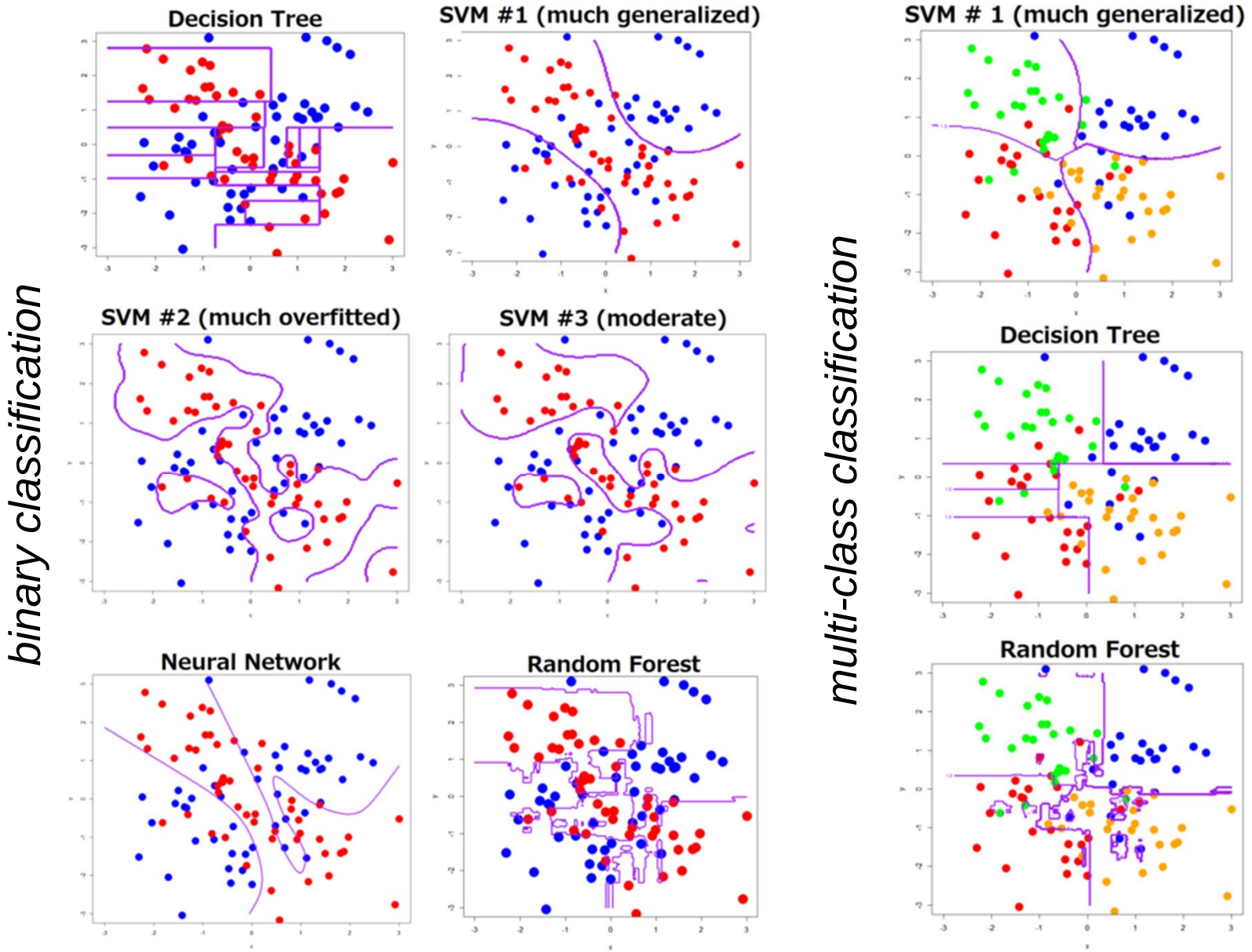
```
options:
-s svm_type : set type of SVM (default 0)
        0 -- C-SVC
        1 -- nu-SVC
        2 -- one-class SVM
        3 -- epsilon-SVR
        4 -- nu-SVR
-t kernel_type : set type of kernel function (default 2)
        0 -- linear: u'*v
        1 -- polynomial: (gamma*u'*v + coef0)^degree
        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

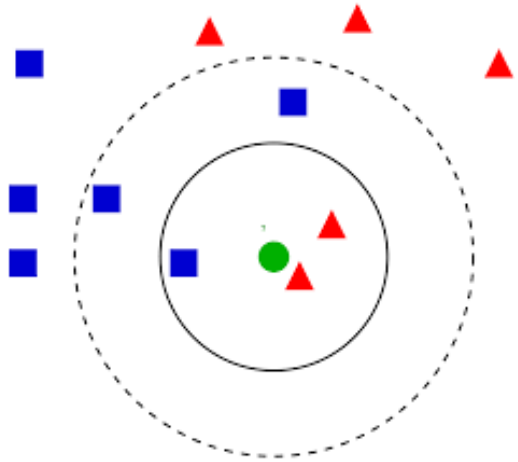


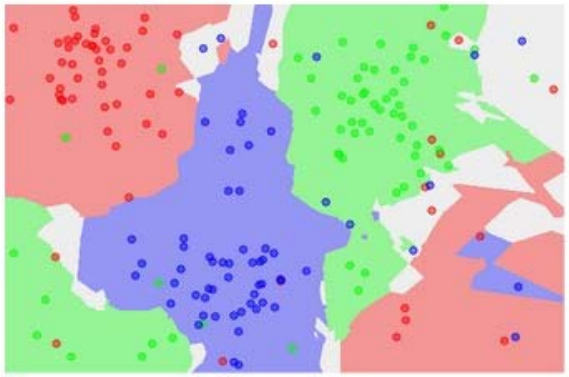



binary classification
Decision Tree
SVM #1 (much generalized)
SVM #2 (much overfitted)
SVM #3 (moderate)
Neural Network
Random Forest
multi-class classification
SVM # 1 (much generalized)
Decision Tree
Random Forest

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 the value of 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)

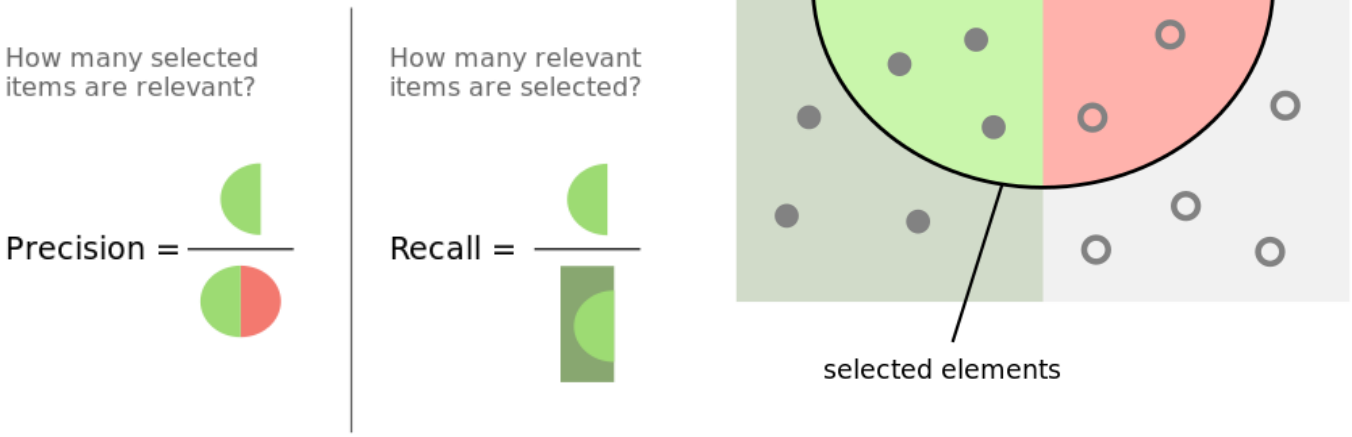
- Performance measures
  - classification accuracy (CA)
  - per-class precision (P), recall (R), F-score (F)
  - confusion table

		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)



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 ≠ validation accuracy
  - that's why we need a balance between model complexity and training error
- **Cross-dataset generalizability**
  - validation accuracy ≠ test accuracy
- **Cheating issue**
  - make sure there is **no overlaps** between the training and test sets
  - **song-level split** (preferred) vs. chunk-level split

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 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 new dataset
  1. **be aware of “confounds”**
     - Example 1: Jazz from the 60’s vs. contemporary Rock
     - Example 2: Happy songs from artist 1 vs. sad songs from artist 2
  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 mean(x) and 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

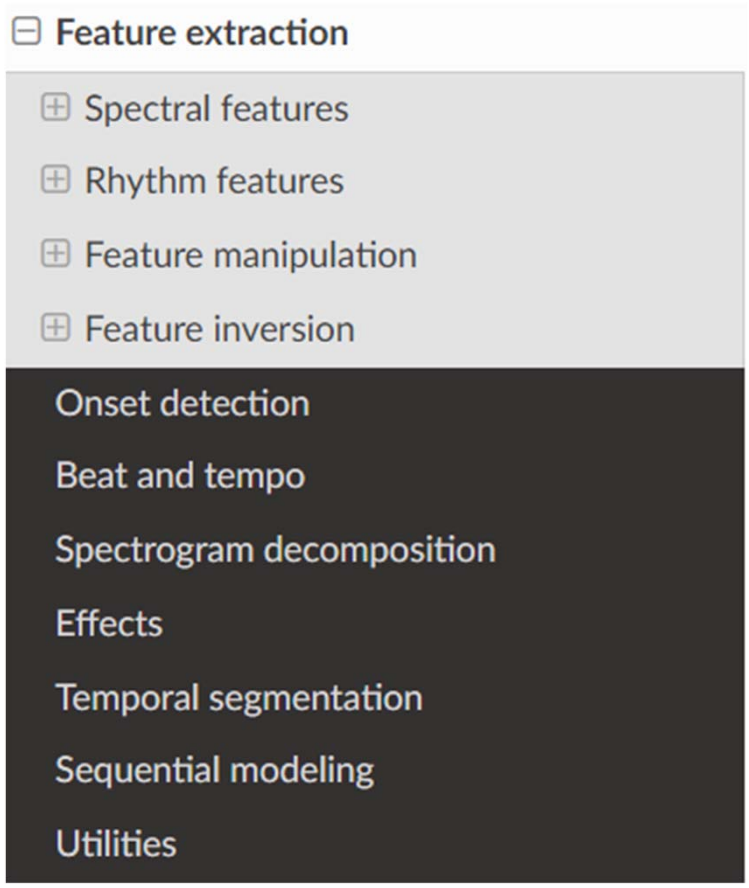
Library: Torchaudio

https://pytorch.org/audio/0.11.0/tutorials/audio_feature_extractions_tutorial.html

```
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")
```

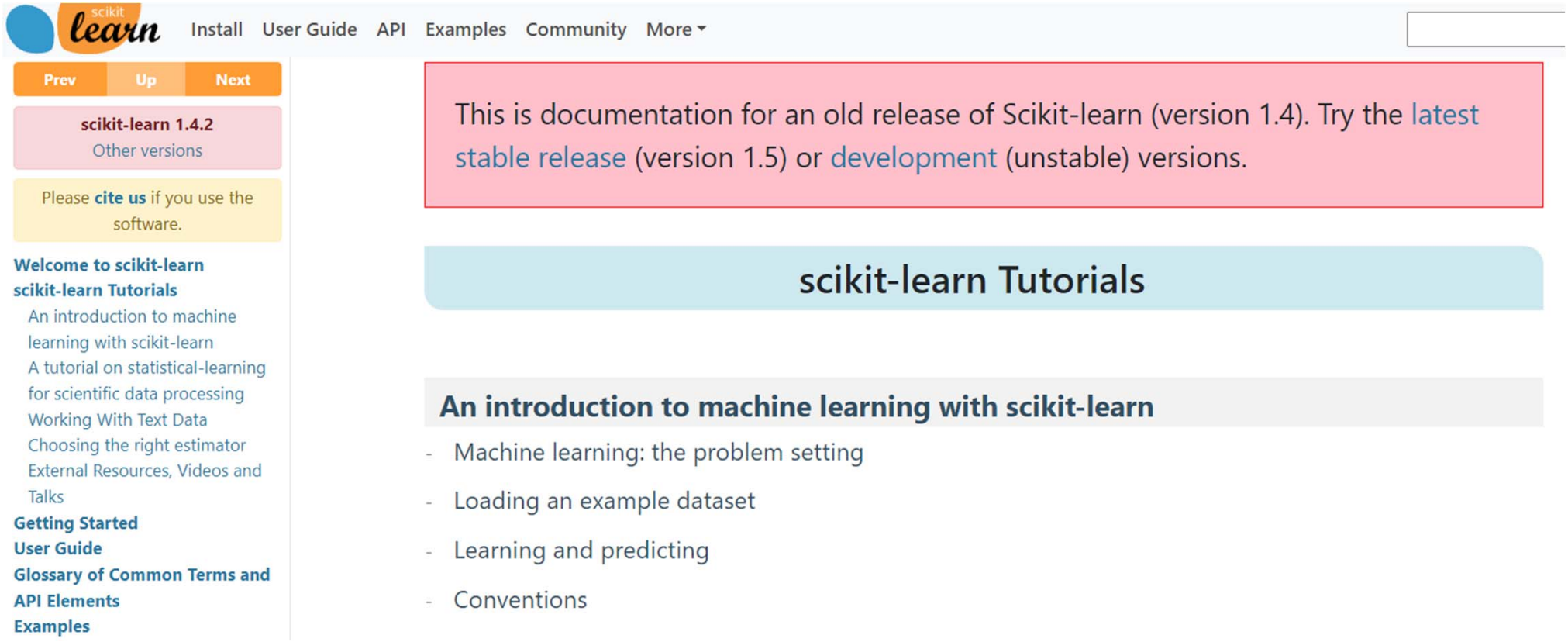
Python: Librosa

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



Python: scikit-learn

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



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)