PAT 463/563 (Fall 2025)

Music & AI

Lecture 11: Music Classification

Instructor: Hao-Wen Dong

UNIVERSITY OF MICHIGAN

Music Classification

Music Classification Tasks

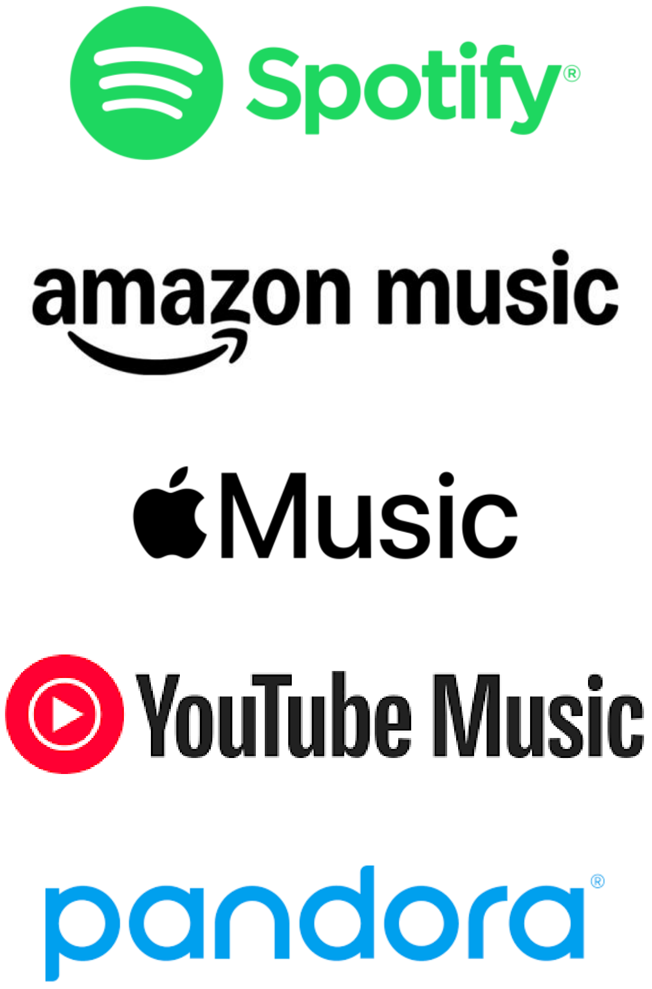
- **Genre** classification (pop, rock, r&b, jazz, hip-hop, classical, etc.)
- **Mood** classification (happy, sad, calm, aggressive, cheerful, etc.)
- **Instrument** recognition
- **Composer** identification
- **Key** detection
- **Chord** estimation
- **Music tagging → Can cover everything above!**

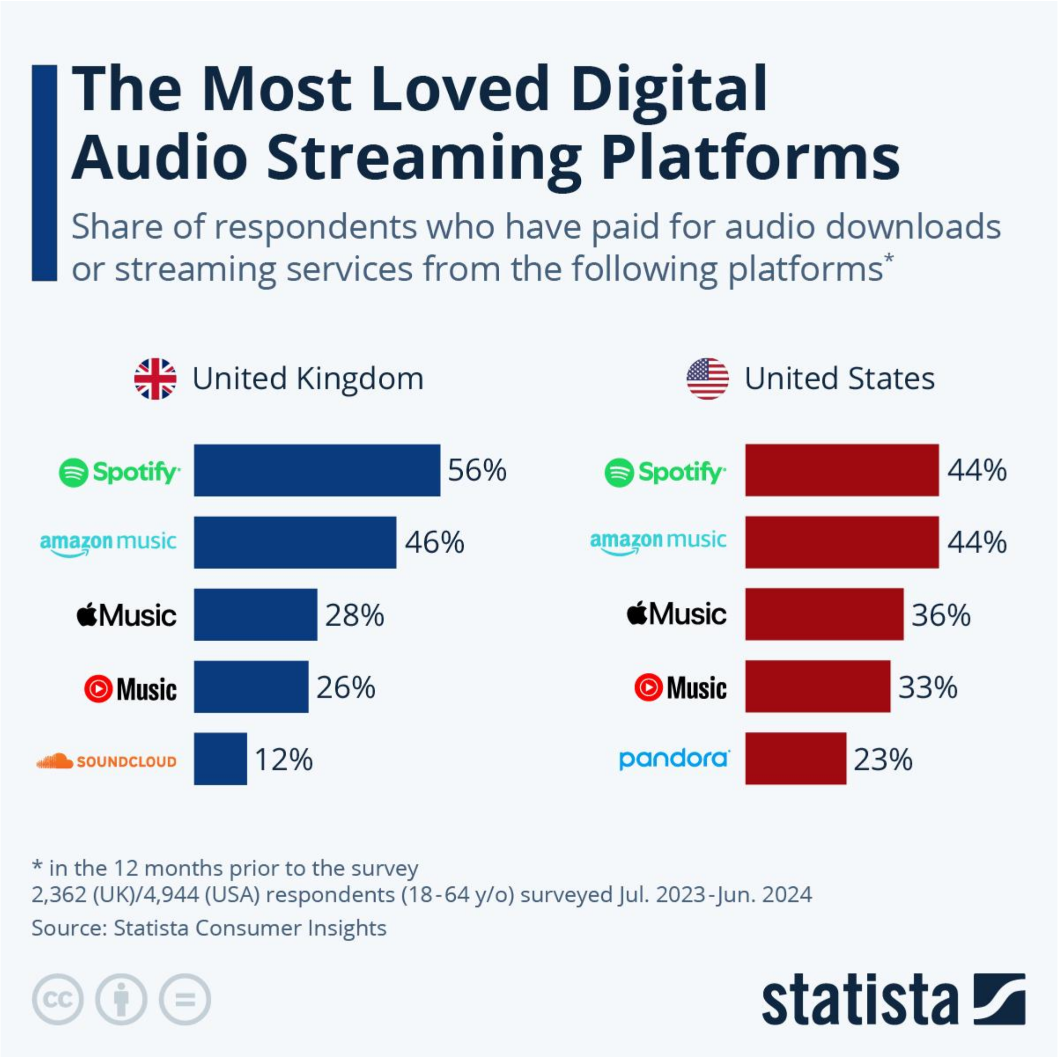
Applications of Music Classification Models

- Recommendation
- Curation
- Playlist generation
- Listening behavior analysis
- Musicology research

Minz Won, Janne Spijkervet, and Keunwoo Choi, "Music Classification: Beyond Supervised Learning, Towards Real-world Applications," *Tutorials of ISMIR*, 2021.

Music Classification for Recommendation

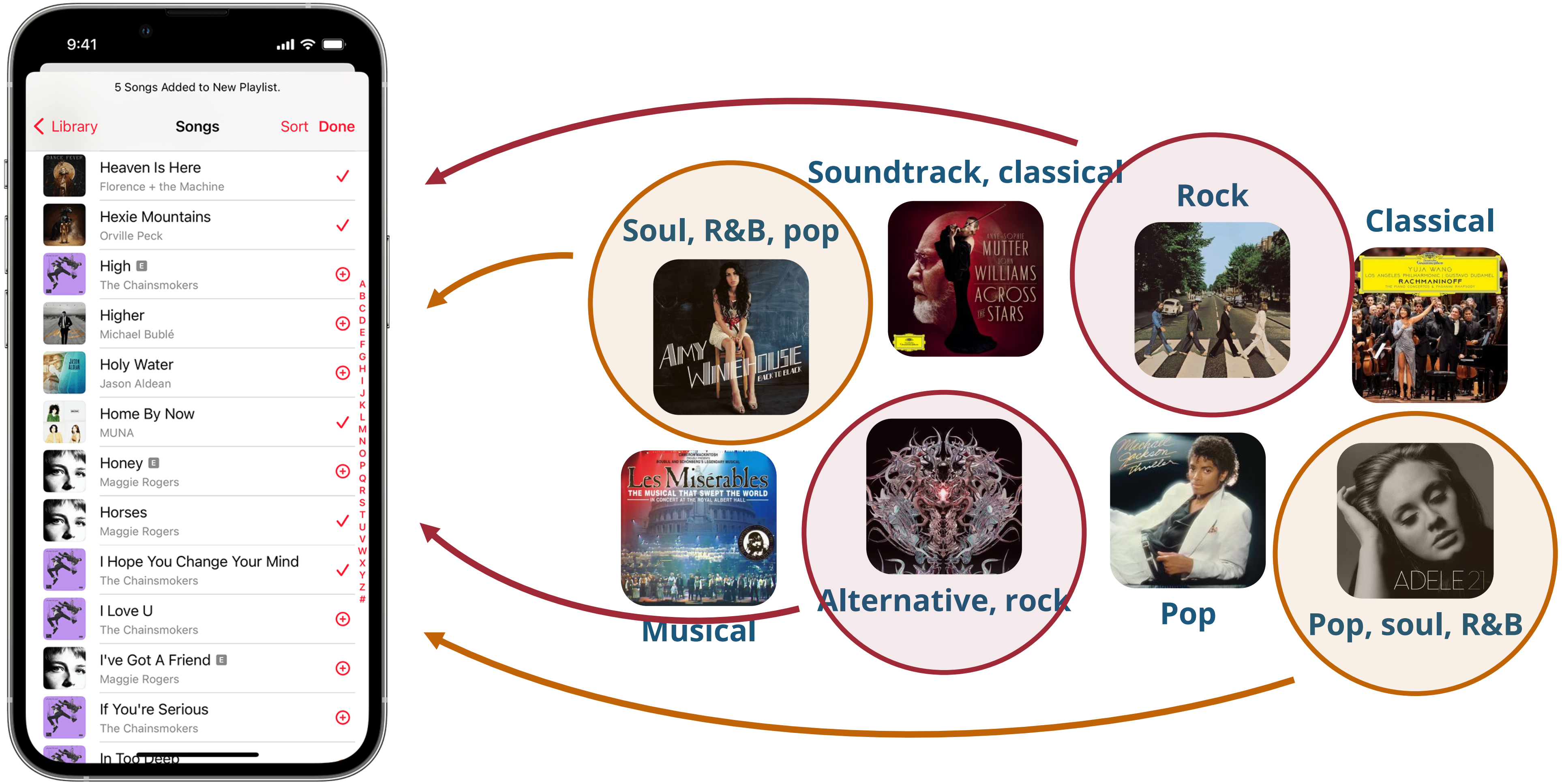




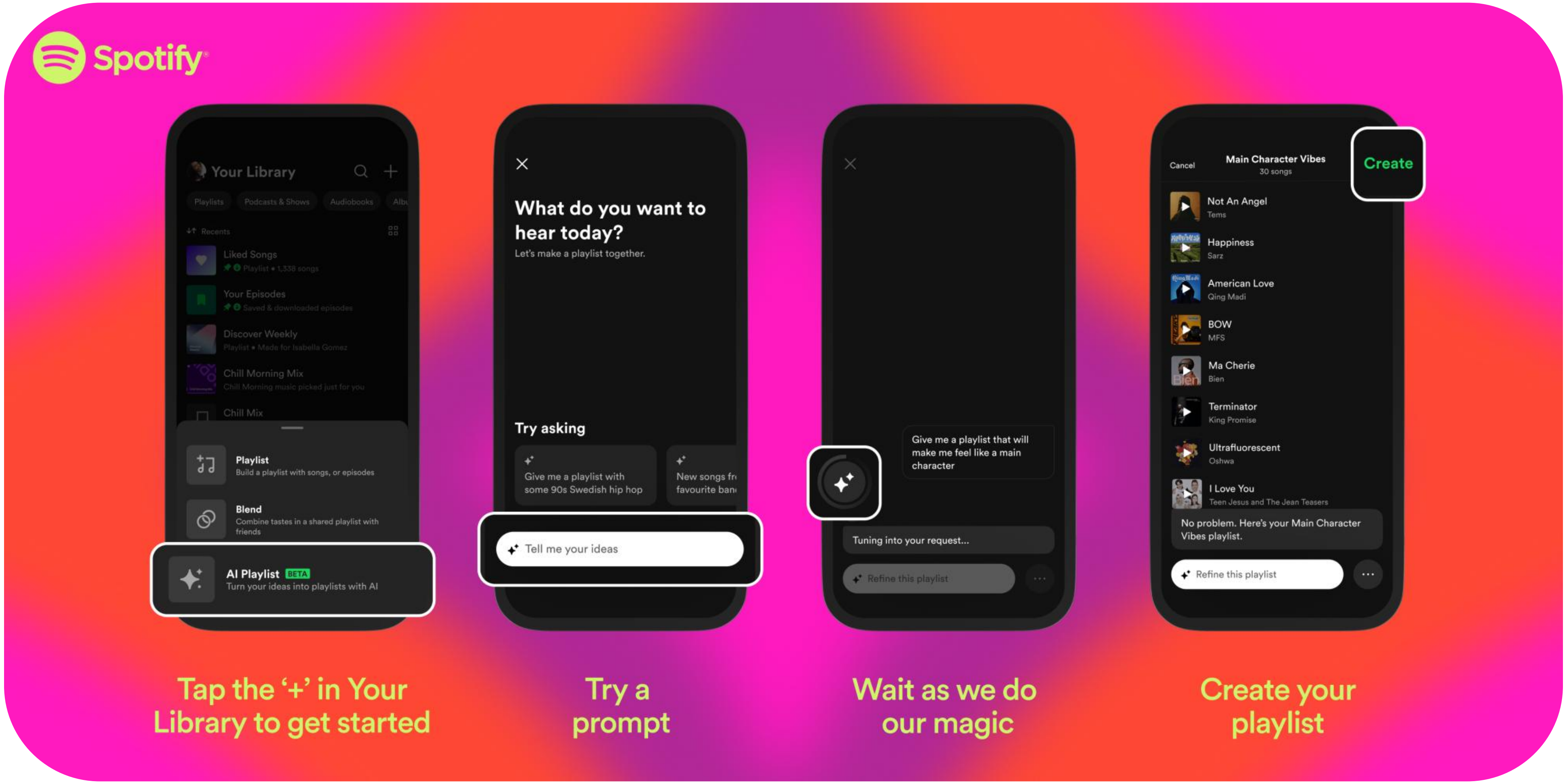
Anna Fleck, "The Most Loved Digital Audio Streaming Platforms," *Statista*, September 30, 2024.

Music Classification for Recommendation

Music Classification for Playlist Generation



Spotify's AI Playlist (2024)

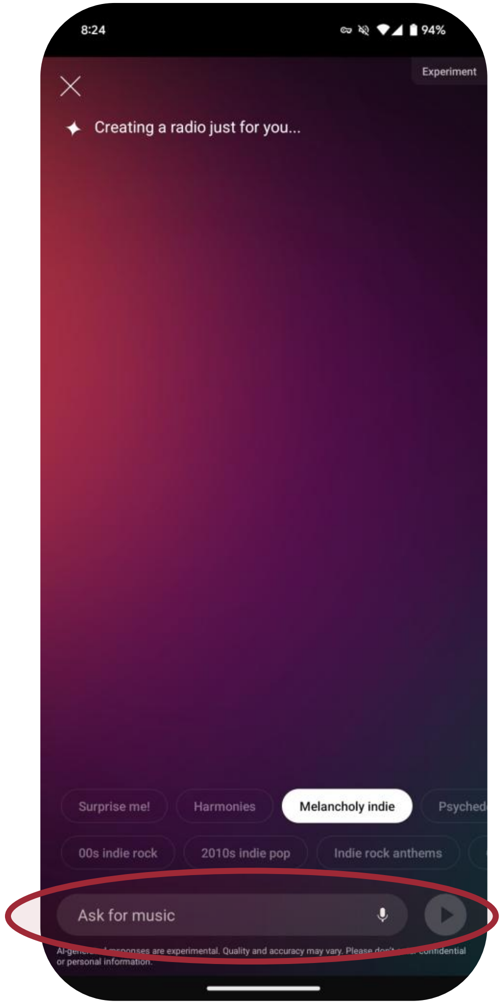


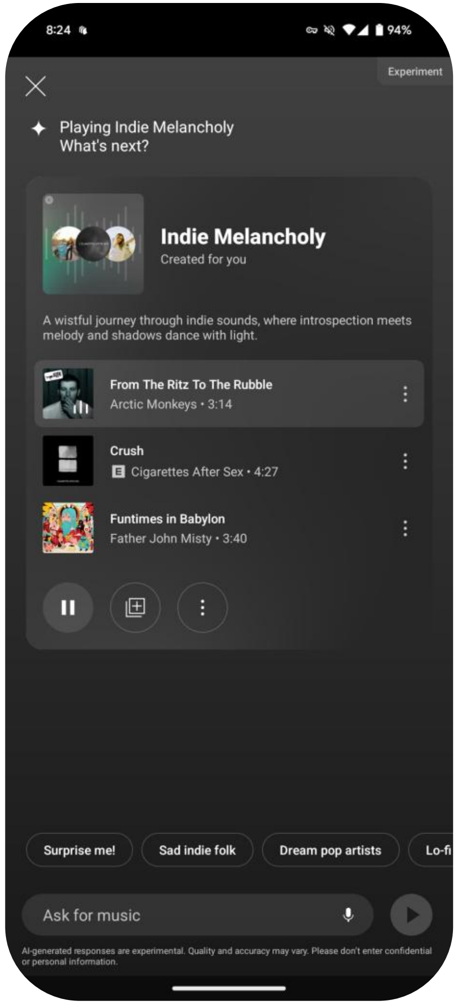
(Source: Spotify)

newsroom.spotify.com/2024-04-07/spotify-premium-users-can-now-turn-any-idea-into-a-personalized-playlist-with-ai-playlist-in-beta/

YouTube Music's Ask Music (2024)

Text prompt

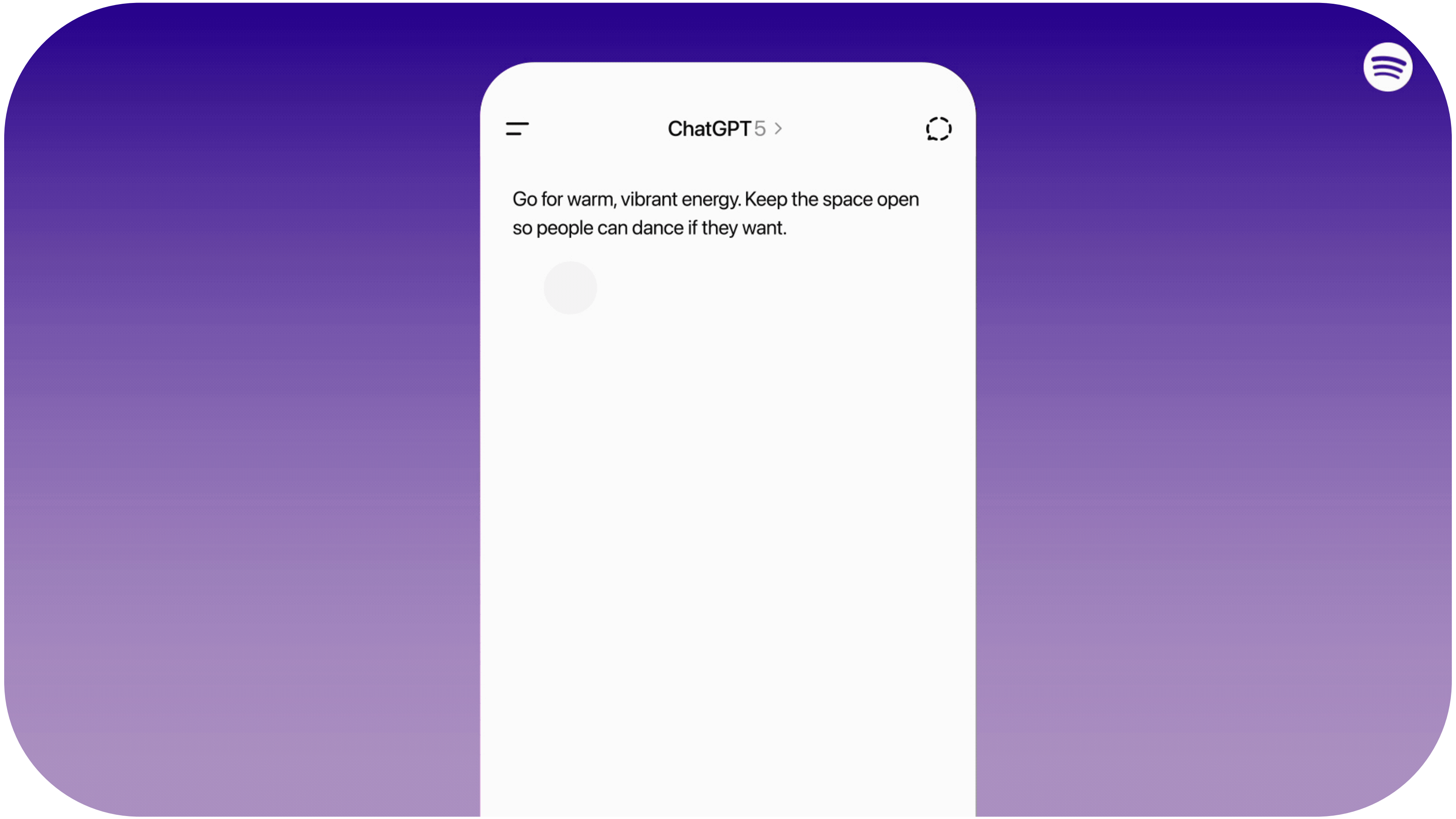




(Source: Android Police)

androidpolice.com/youtube-music-ask-music-ai-playlist-assistant/

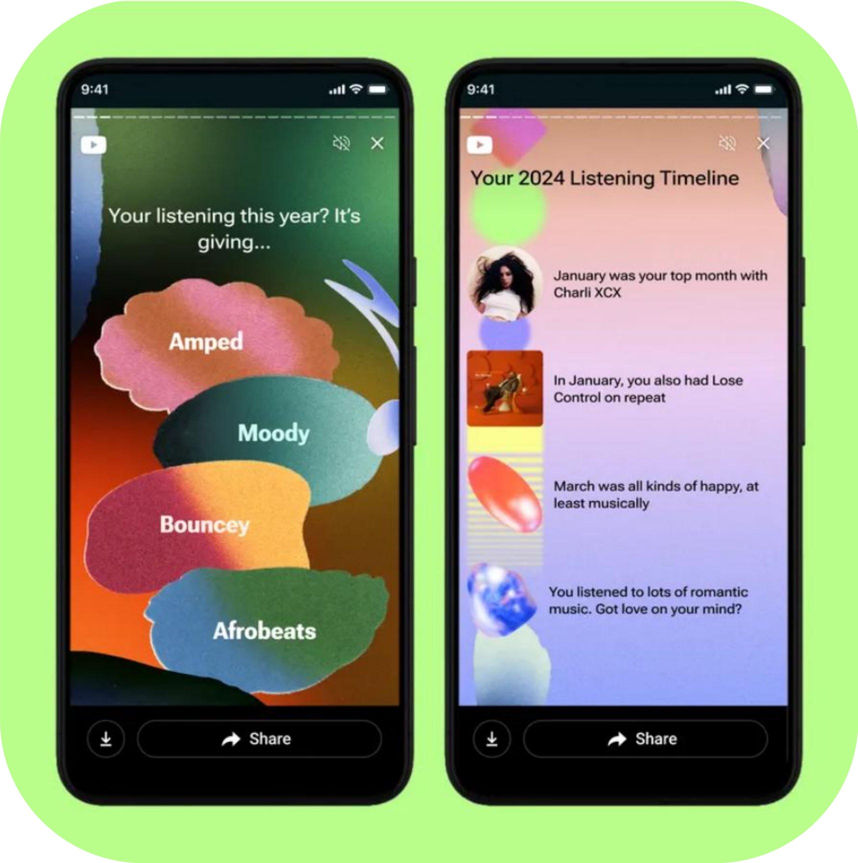
Spotify's Personalized Picks in ChatGPT (2025)



(Source: Spotify)

Music Classification for Listening Behavior Analysis

YouTube's Music Recap



(Source: YouTube)

Spotify's Listening Personality

(Source: Spotify)

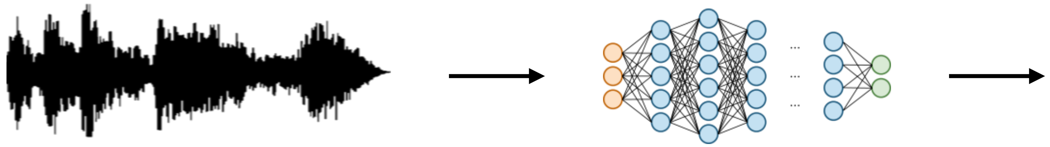
blog.youtube/news-and-events/2024-music-recap-youtube/
engineering.atspotify.com/2023/01/whats-a-listening-personality/

Types of Classification Tasks

Types of Classification Tasks

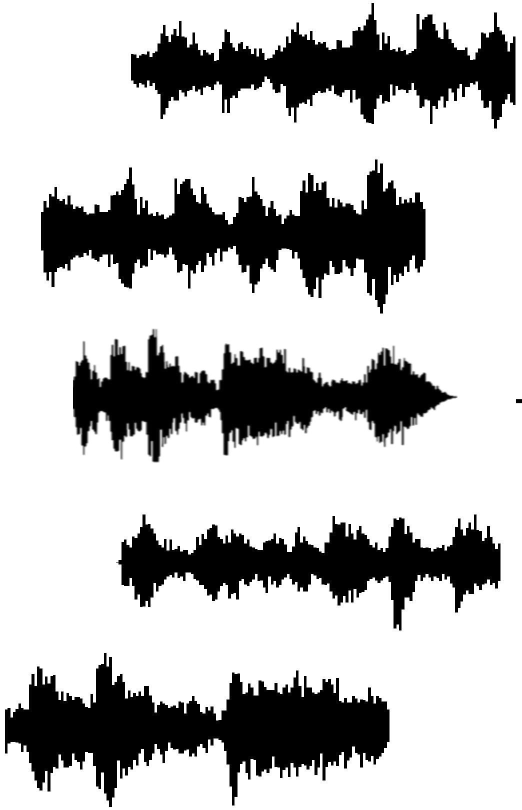
- **Binary** classification
- **Multiclass** classification
- **Multi-label** classification

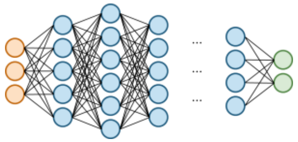
Binary Classification



Is Classical?

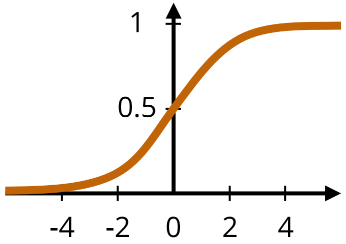
Binary Classification



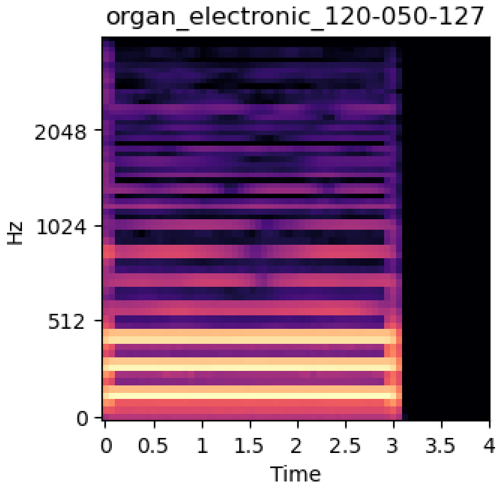


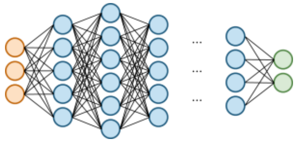
		Is Classical?
4.86		**0.99**
-1.87		**0.13**
2.10		**0.89**
-3.52		**0.03**
0.18		**0.54**

Sigmoid function



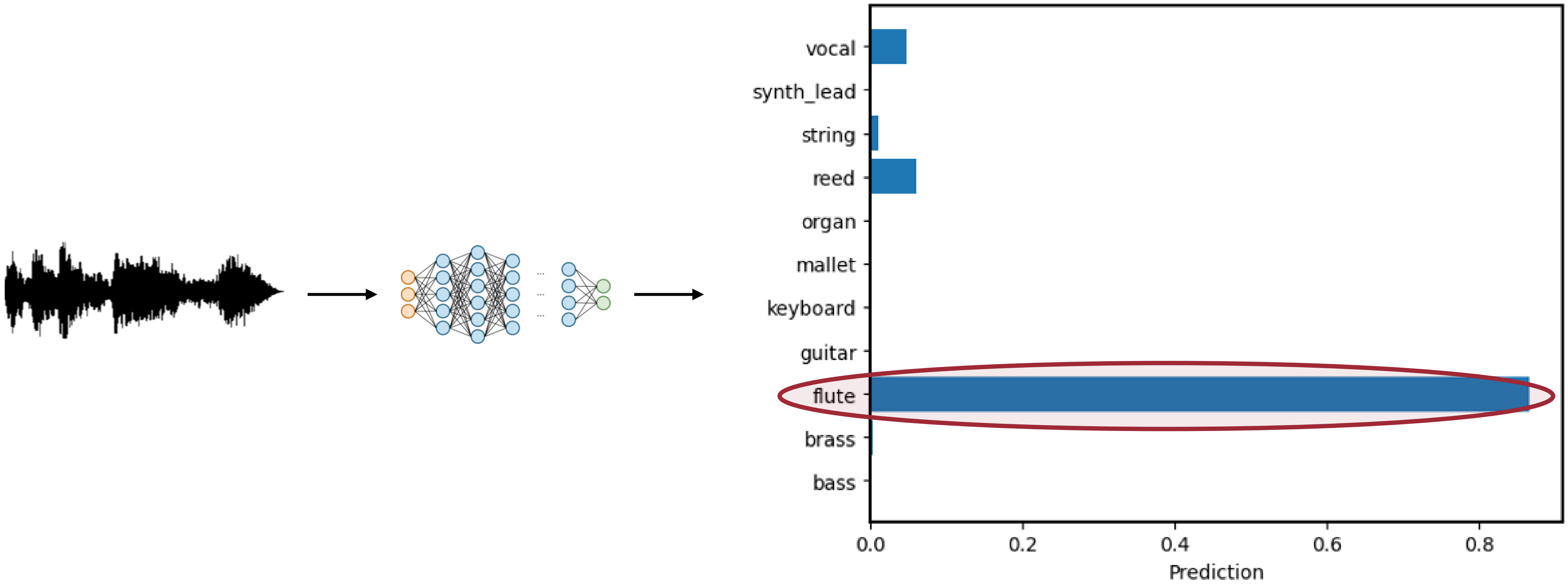
Multiclass Classification





Bass
Brass
Flute
Guitar
Keyboard
Mallet
Organ
Reed
String
Synth
Lead
Vocal

Multiclass Classification



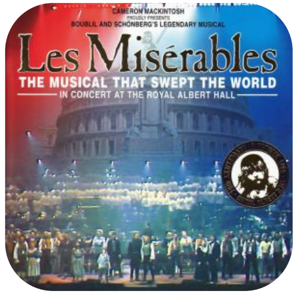
Multi-label Classification

Soul, R&B, pop

Soundtrack, classical

Rock

Classical



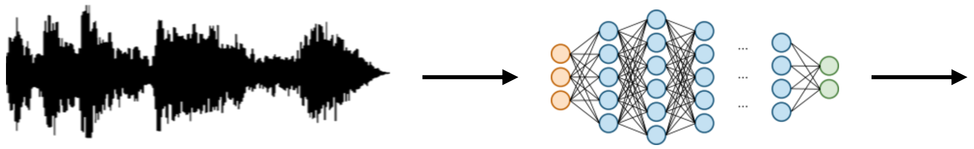
Musical

Alternative, rock

Pop

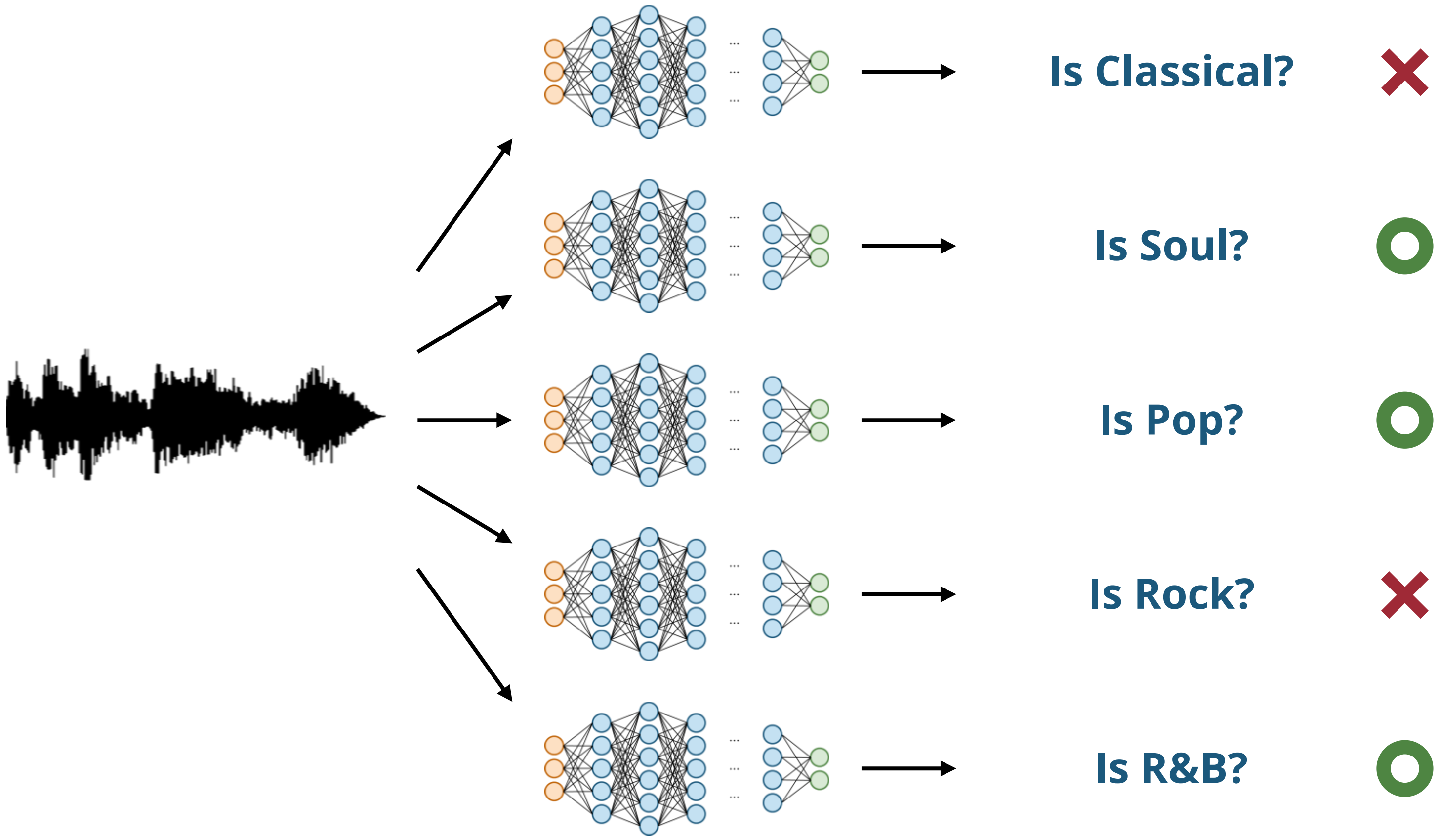
Pop, soul, R&B

Multi-label Classification

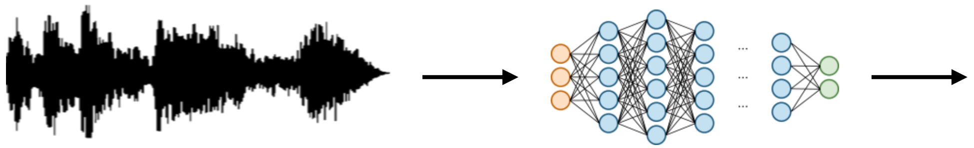


Soul, R&B, pop

Multi-label Classification as Binary Classification



Multi-label Classification as Multi-class Classification

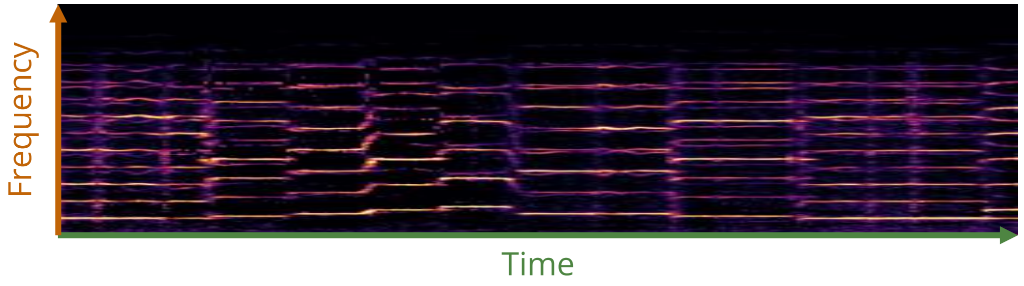


{Pop}
{R&B}
{Soul}
{Classical}
{Pop, R&B}
{Pop, Soul}
…
{Pop, Classical}
{Pop, R&B, Soul}
…
{R&B, Soul, Classical}
{Pop, R&B, Soul, Classical}

Label powerset

Input Features

- **Waveforms**
- Time-frequency representation (**spectrograms**)
- **Hand-crafted features** or **features provided in metadata**
  - **Acoustic**: loudness, pitch, timbre
  - **Rhythmic**: beat, tempo, time signature
  - **Tonal**: key, scale, chords
  - **Instrumentation**, **expressions**, **structures**, etc.



Common Datasets

- **GTZAN**: 1,000 30-sec songs, 10 genres
- **MagnaTagATune**: 5,405 29-sec songs, 188 tags, 230 artists
- **Million Song Dataset (MSD)**: 1M 30-sec songs, >500K tags, tricky to access
- **Free Music Archive (FMA)**: >10K full songs, 163 genres
- **MTG-Jamendo**: 55K full songs, 195 tags
- **AudioSet**: 1M songs, YouTube URLs, low-quality audio
- **NSynth**: ~306K 4-sec one-shot instrument sounds

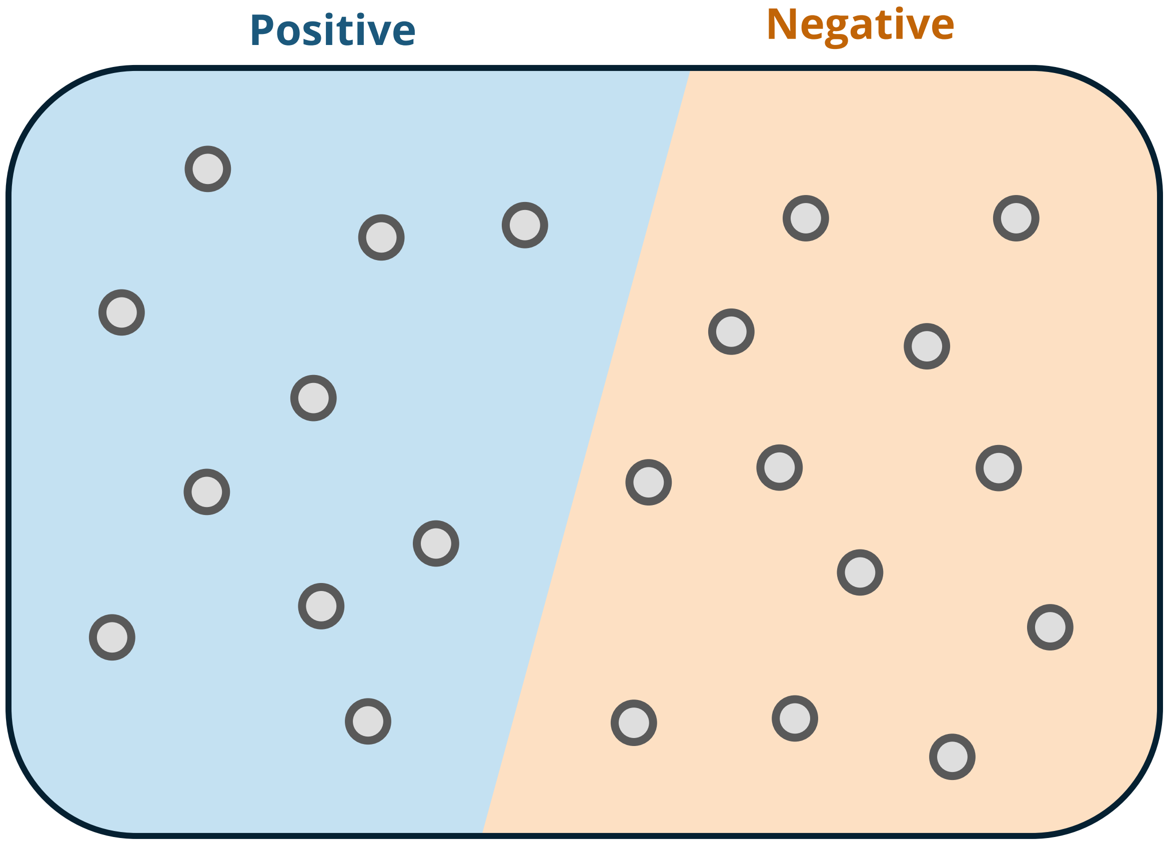
Minz Won, Janne Spijkervet, and Keunwoo Choi, "Music Classification: Beyond Supervised Learning, Towards Real-world Applications," *Tutorials of ISMIR*, 2021.

Evaluation Metrics

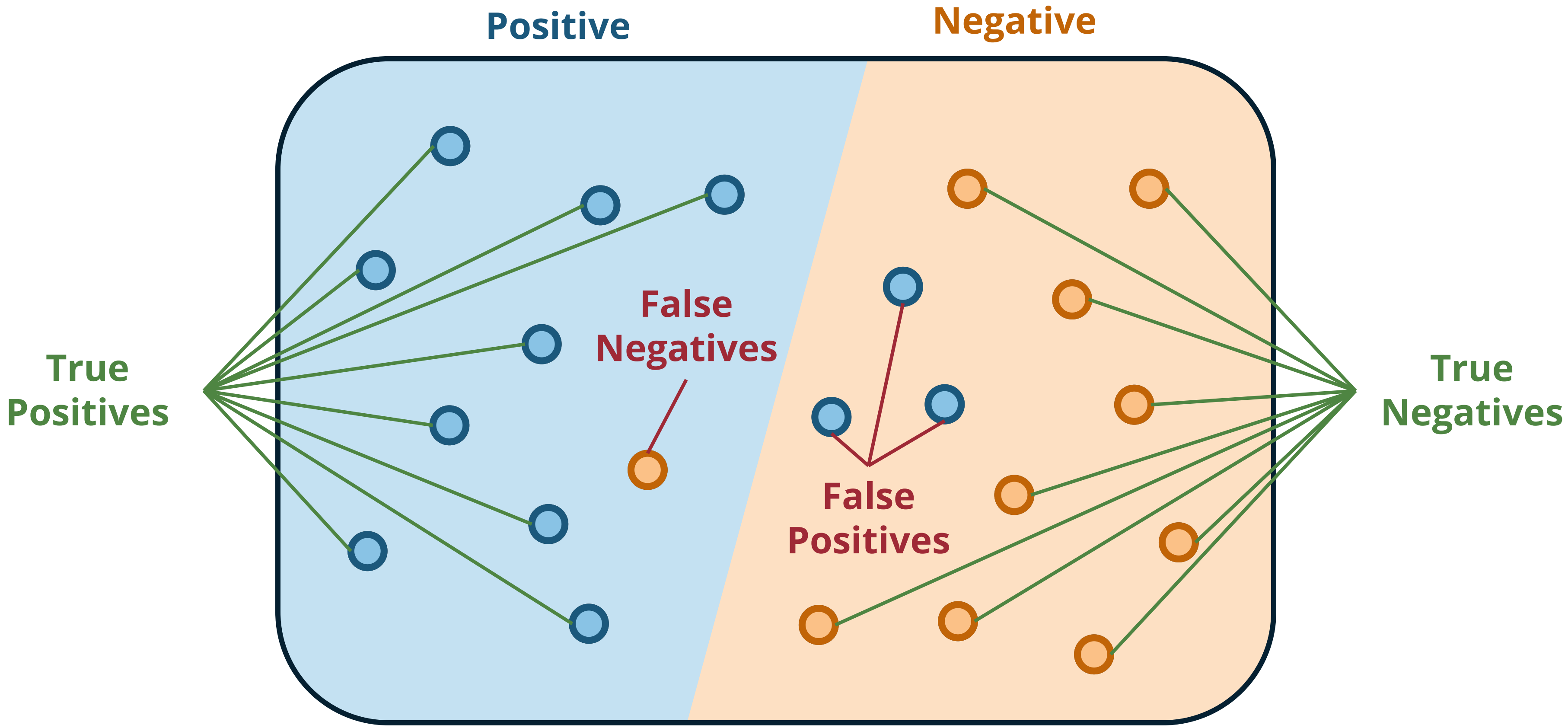
Evaluation Metrics

- **Key**: Capture what you care the most!
- The best evaluation metric depends on the actual use case
- Best to use several evaluation metrics to obtain a holistic view of your model's performance

Toy Example: Binary Classification

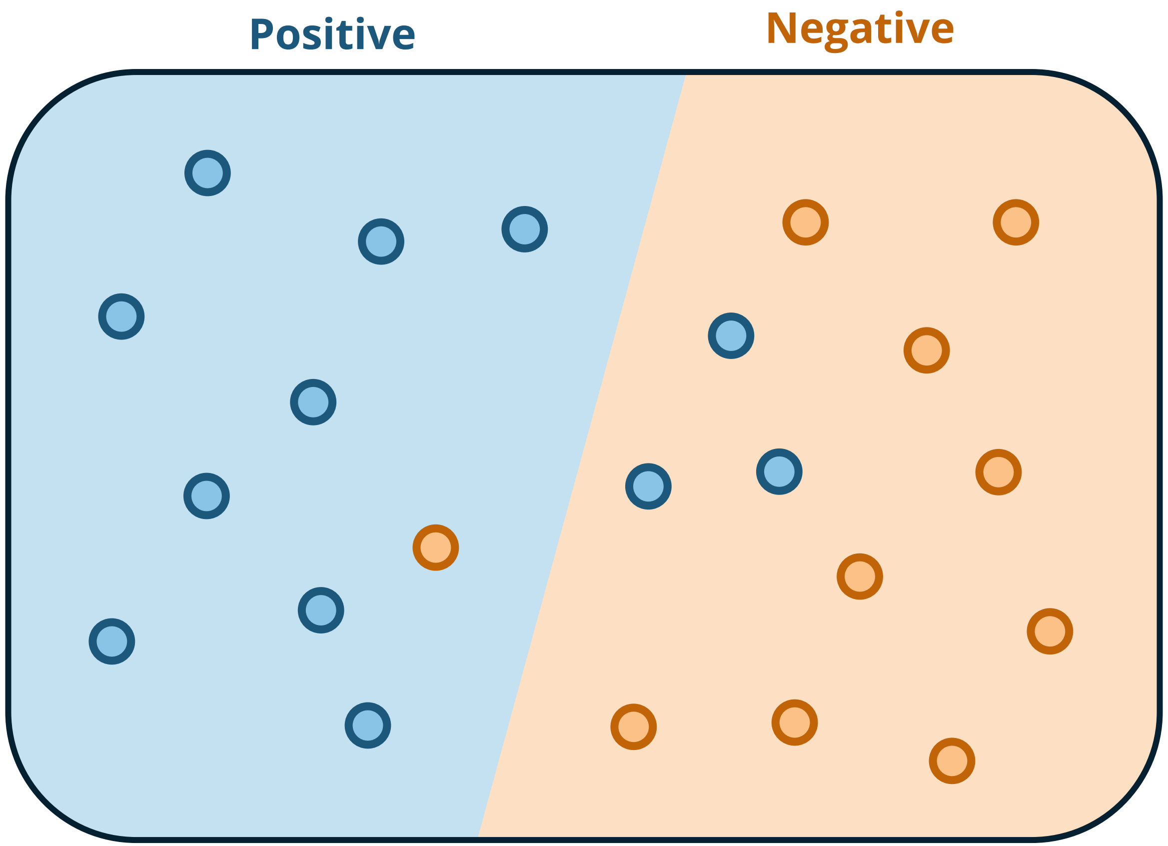


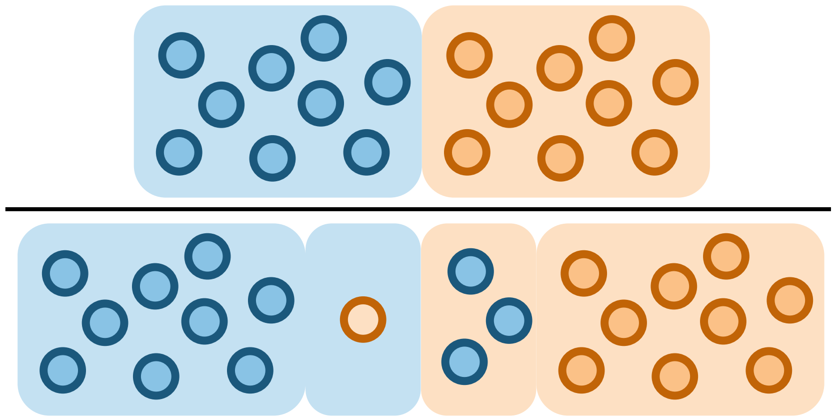
Toy Example: Binary Classification



Accuracy

- **Definition**: Percentage of correct predictions across all classes



Accuracy = 

= 0.82

Confusion Matrix for Binary Classification

		Prediction	
		Positive	Negative
Ground truth	Positive	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

Confusion Matrix for Binary Classification

		Prediction	
		Positive	Negative
Ground truth	Positive	True Positives (TP)  Cat	False Negatives (FN)  Not Cat
	Negative	False Positives (FP)  Cat	True Negatives (TN)  Not Cat

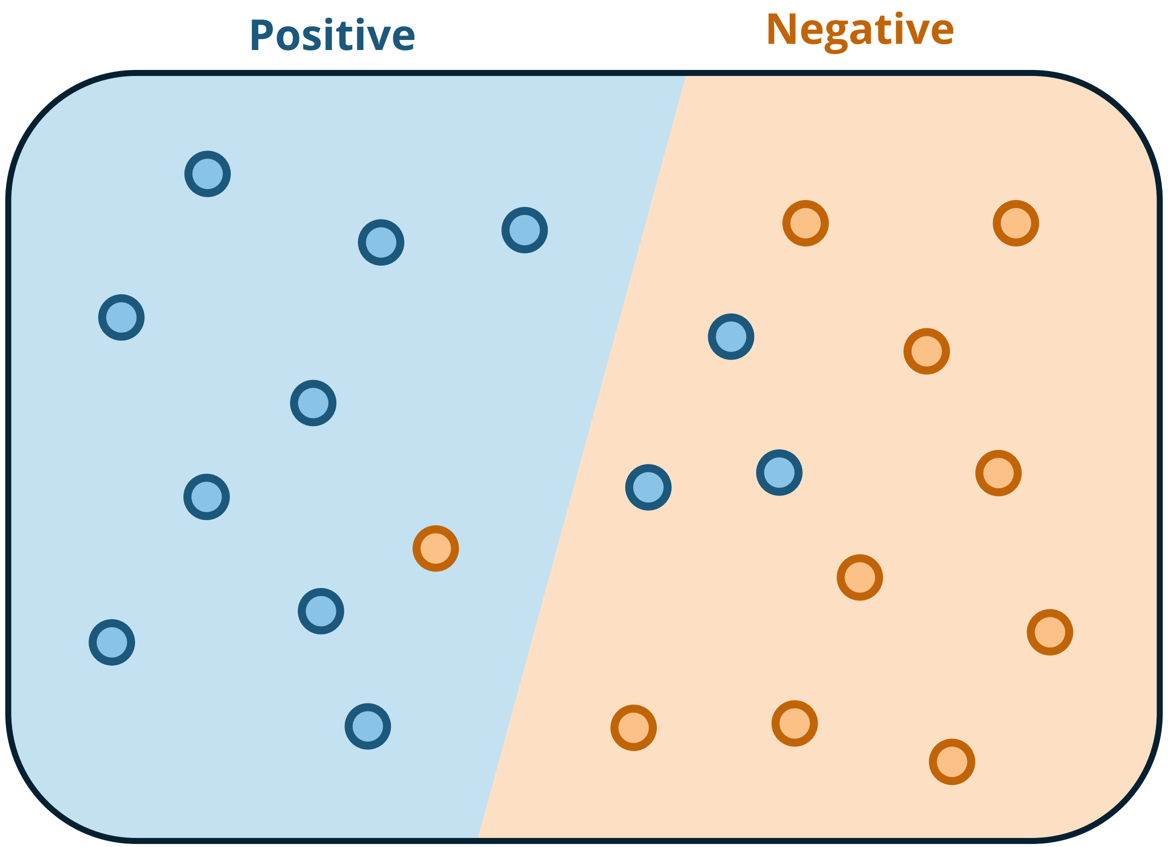
Accuracy on Imbalanced Datasets

- Accuracy does ***not*** work well on **imbalanced dataset**
- Take **a disease with a 1% prevalence** for example:
  - What if we simply say **negative to all diagnoses**?

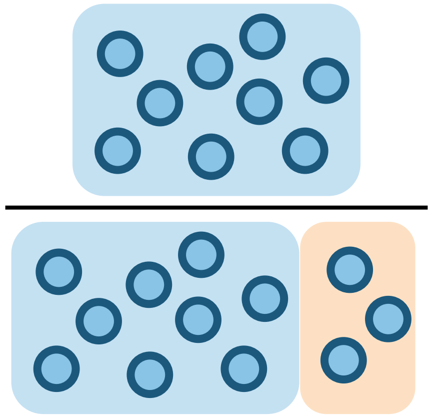
		Prediction	
		Positive	**Negative**
Ground truth	**Positive**	**True Positives (TP)** **0**	**False Negatives (FN)** **1**
	Negative	**False Positives (FP)** **0**	**True Negatives (TN)** **99**

Accuracy = 0.99

Precision



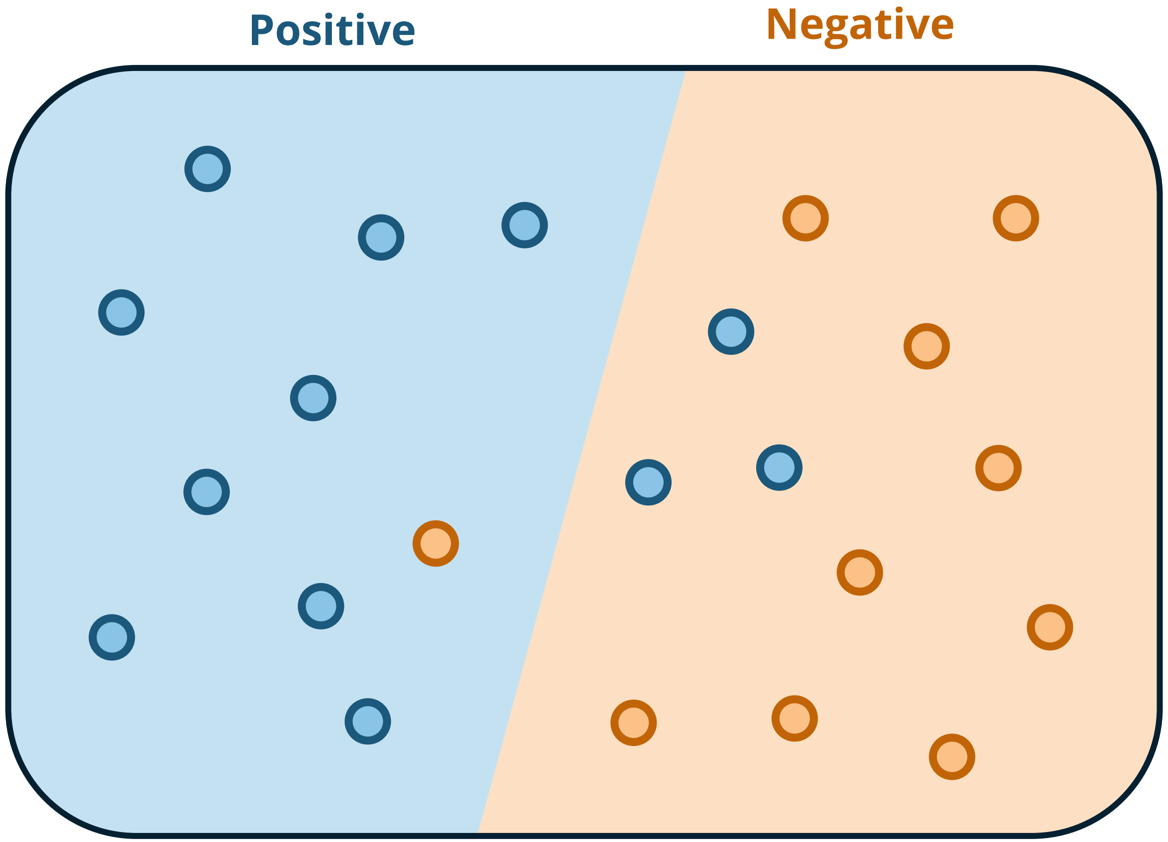
$$\text{Precision} = \frac{TP}{TP + FP} =$$



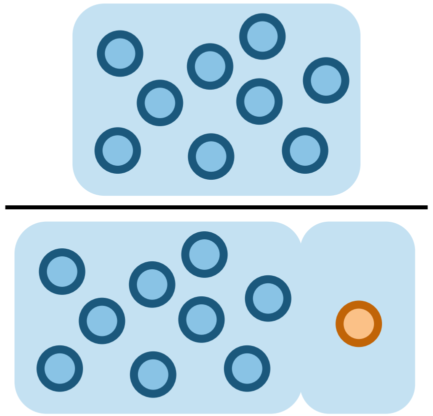
$$= 0.75$$

How often predictions for the positive are correct

Recall



$$\text{Recall} = \frac{TP}{TP + FN} = \frac{}{} = 0.9$$

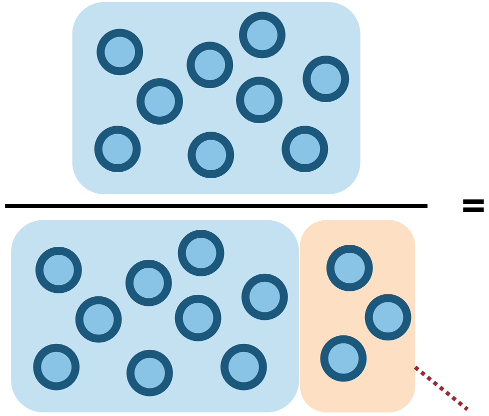


How well the model finds all positive instances in the dataset

Precision vs Recall

Precision

$$\frac{TP}{TP + FP} =$$



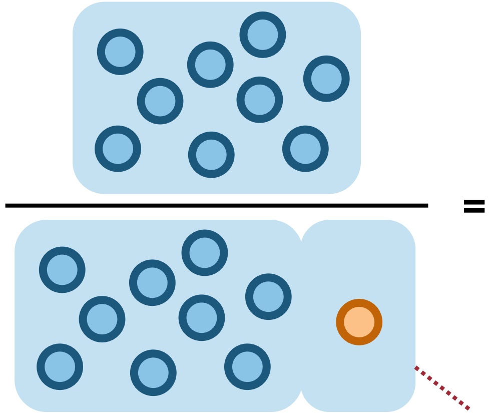
= 0.75

False alarms

How often predictions for the positive are correct

Recall

$$\frac{TP}{TP + FN} =$$



= 0.9

Missed detection

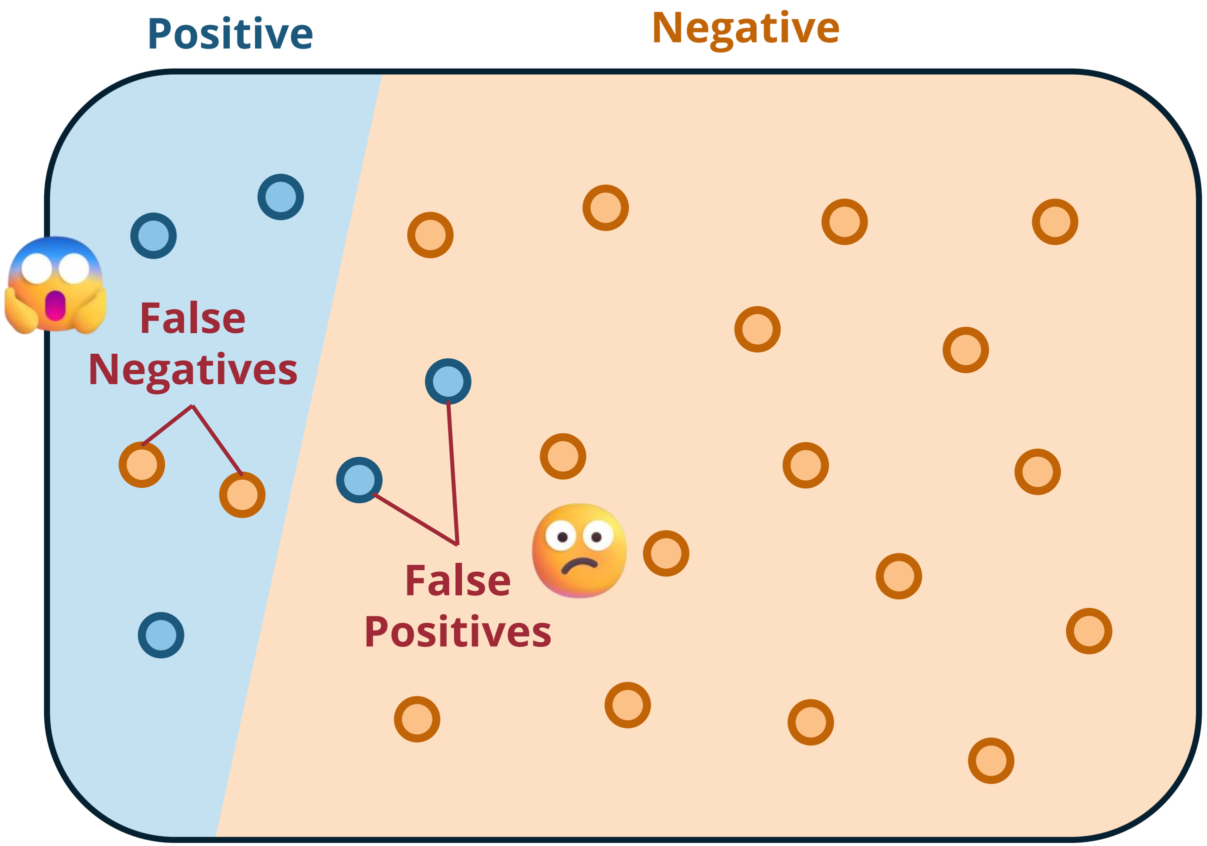
How well the model finds all positive instances in the dataset

When should we care about Precision & Recall?

Rare cancer detection

Aim for high precision or high recall?

High recall ensures most cancer cases are identified.



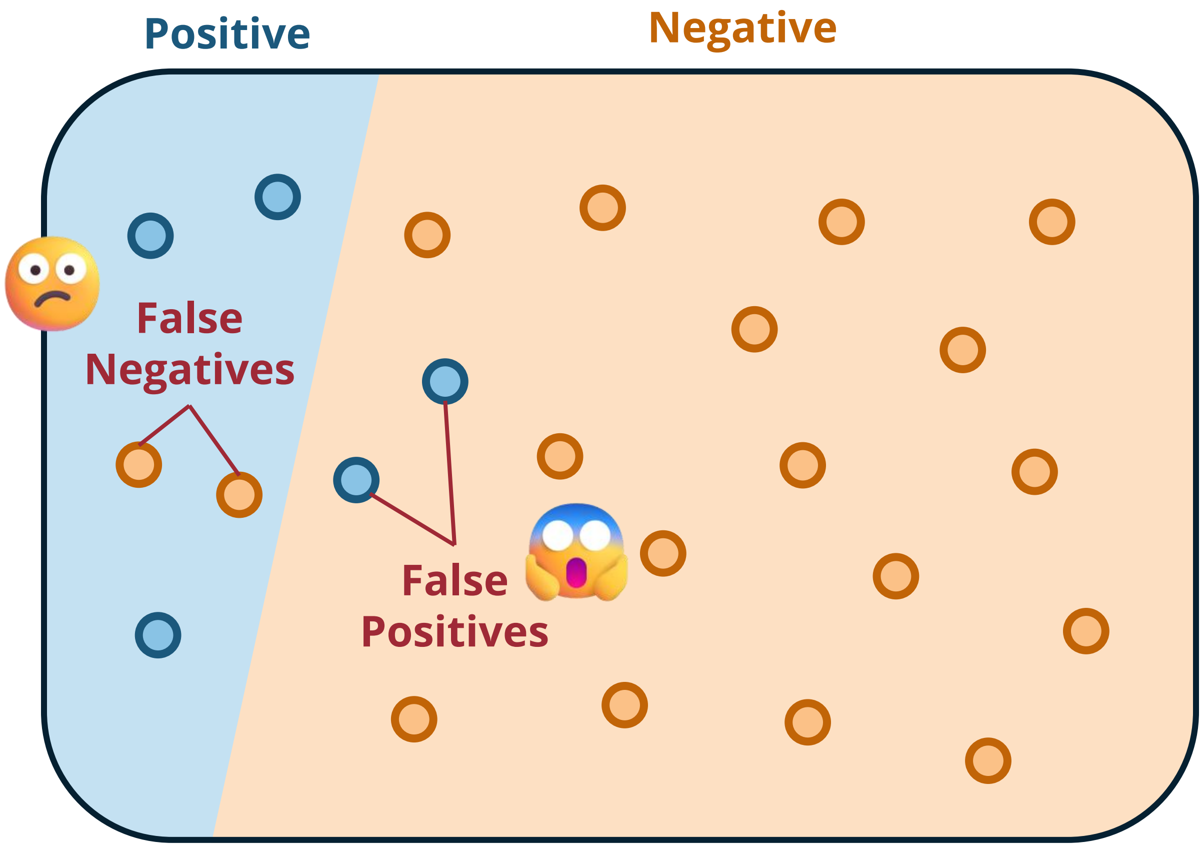
False alarms vs Missed detections

When should we care about Precision & Recall?

Music recommendation

Aim for high precision or high recall?

High precision ensures that the model won't recommend irrelevant items.



False alarms vs Missed recommendations

F1 Score: Considering both Precision & Recall

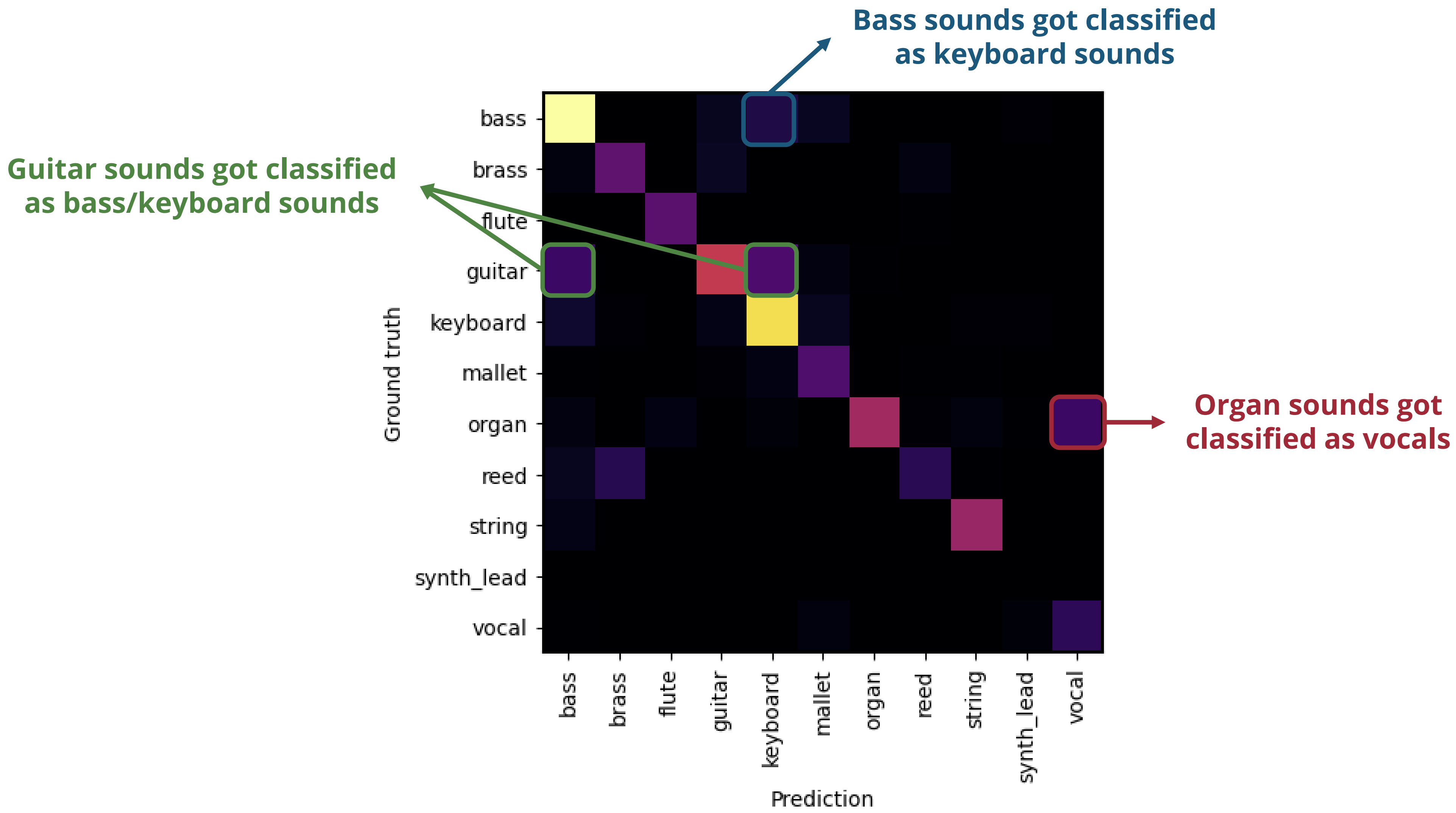
- Particularly useful for **imbalanced datasets**
  - Work better than accuracy when the dataset is imbalanced
  - For example, **music search**, **retrieval**, and **recommendation**

$$
\begin{aligned}
F_1 &= \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \\
&= \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}
\end{aligned}
$$

Confusion Matrix for Binary Classification

		Prediction	
		Positive	Negative
Ground truth	Positive	True Positives (TP)  Cat	False Negatives (FN)  Not Cat
	Negative	False Positives (FP)  Cat	True Negatives (TN)  Not Cat

Confusion Matrix for Multiclass Classification

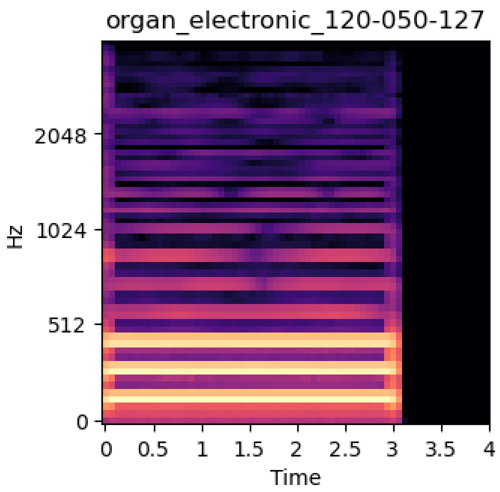


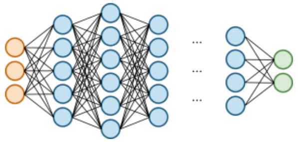
Resources on Music Classification

- Minz Won, Janne Spijkervet, and Keunwoo Choi, "Music Classification: Beyond Supervised Learning, Towards Real-world Applications," *Tutorials of ISMIR*, 2021.
- **Open source** music classification models
  - github.com/minzwon/sota-music-tagging-models
  - github.com/jordipons/musicnn

🔥 HW 3: Musical Note Classification using CNNs

- Instructions will be sent by **emails** and released on the **course website**
- Train a CNN that can classify audio files into their **instrument families**
  - **Input**: 64x64 mel spectrogram
  - **Output**: 11 instrument classes

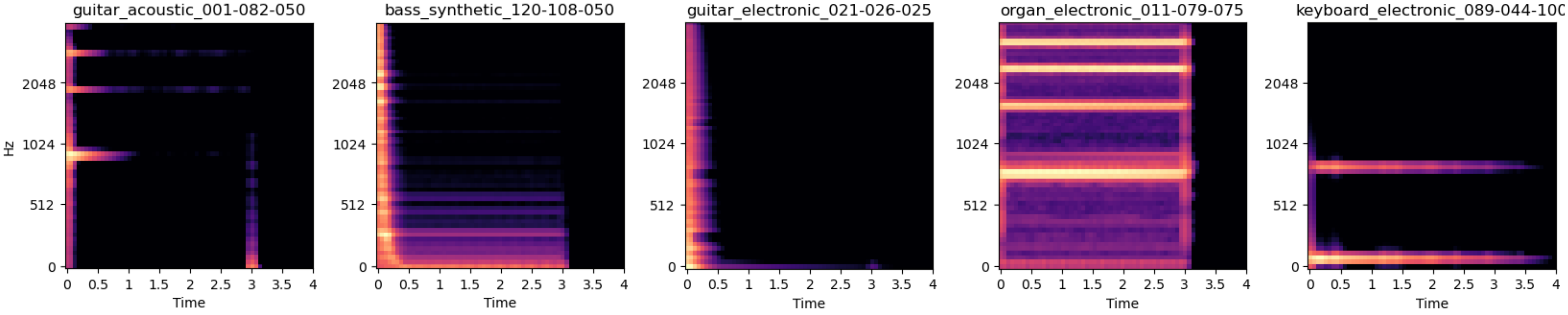




Bass
Brass
Flute
Guitar
Keyboard
Mallet
Organ
Reed
String
Synth Lead
Vocal

NSynth Dataset

- A collection of 305,979 **one-shot musical notes** (Engel et al., 2017)
  - Produced from 1,006 **commercial sample libraries**
  - With different **MIDI pitches** (21–108) and **velocities** (25, 50, 75, 100, 127)



Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi, “Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders,” *ICML*, 2017.

More on Optimization

Training a Neural Network

Build a neural network
(which defines a set of functions)

↓

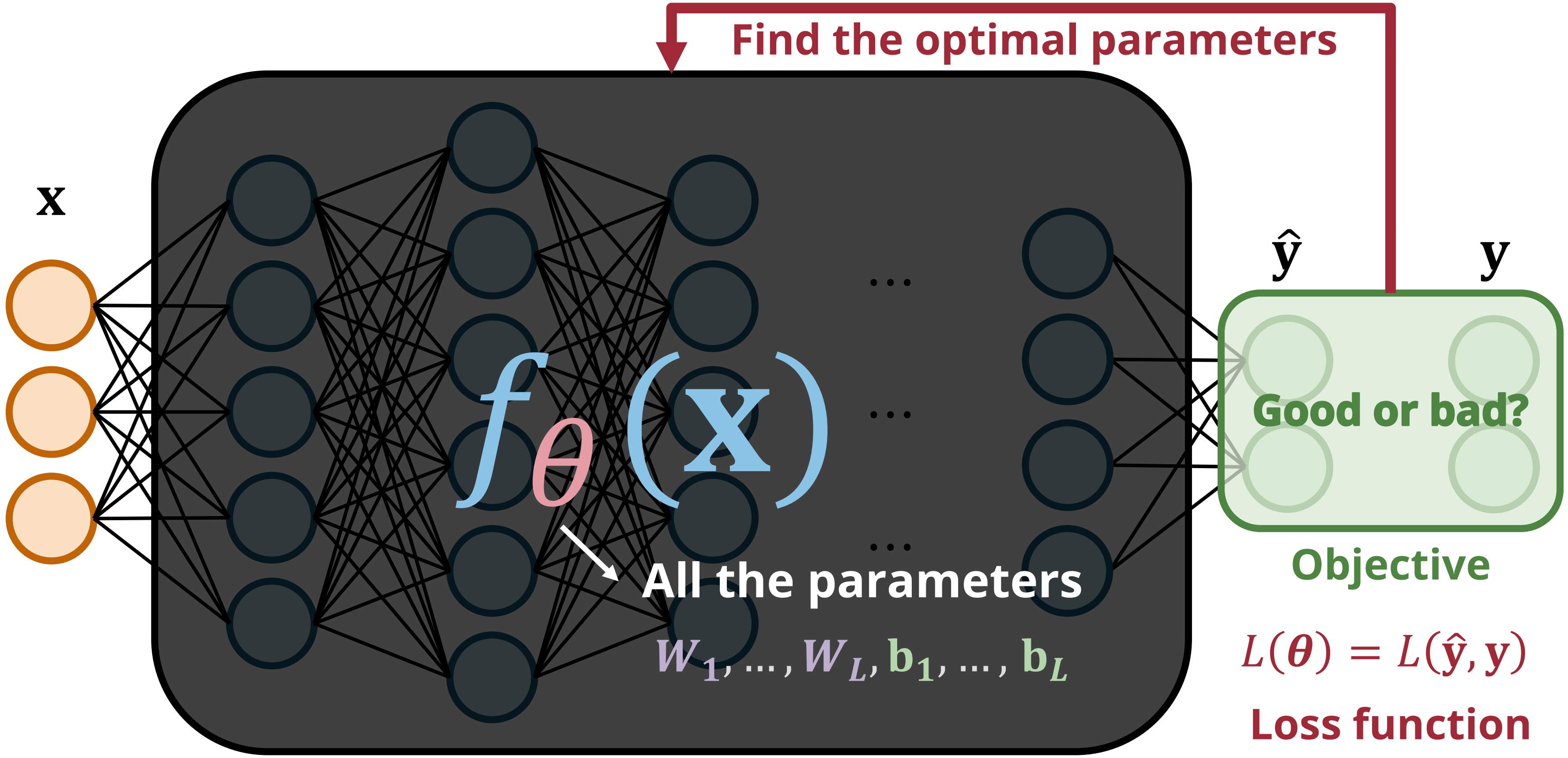
Define the objective
(i.e., what is good for a function)

↓

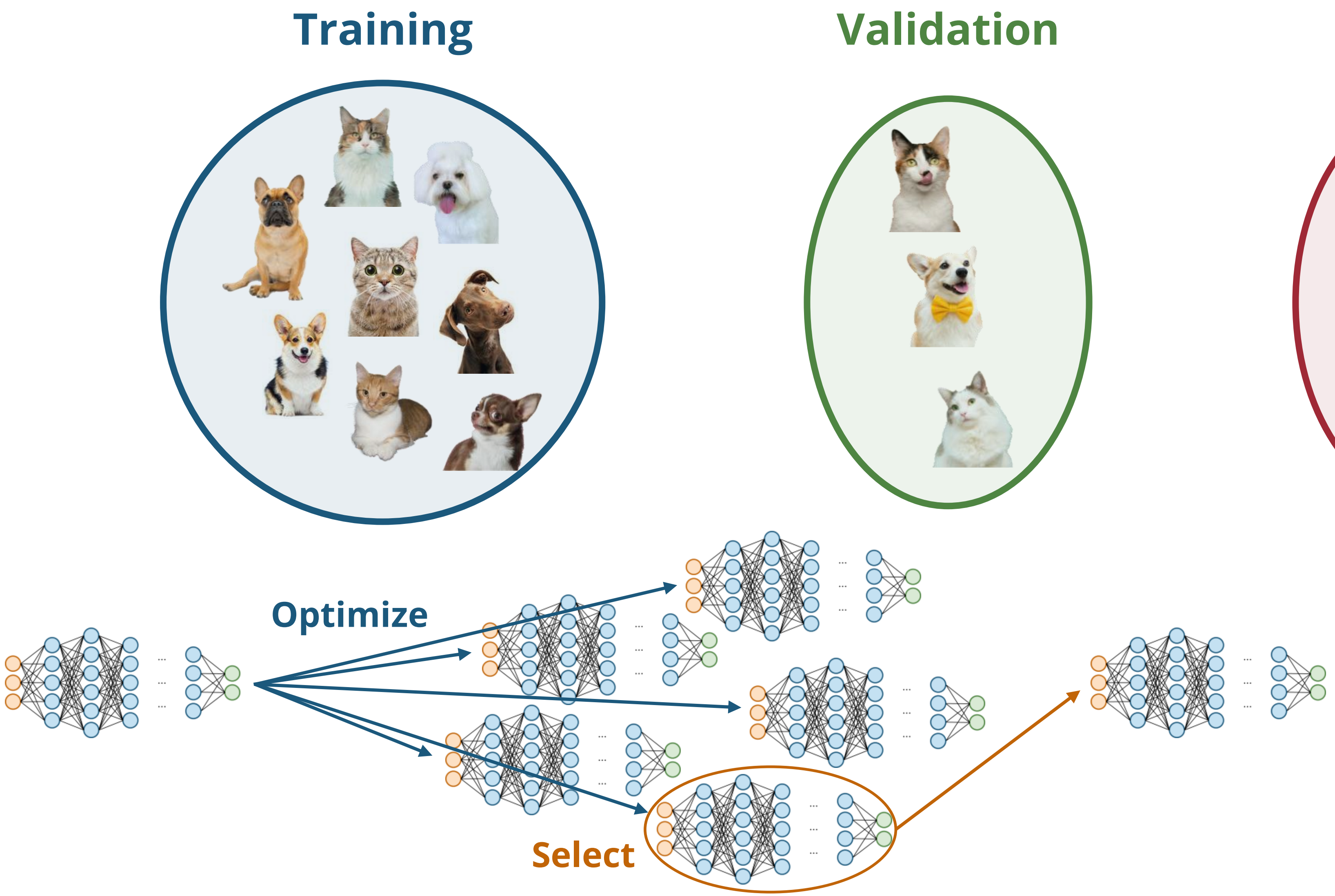
Find the optimal parameters
(which leads to the best function)

Neural Networks are Parameterized Functions

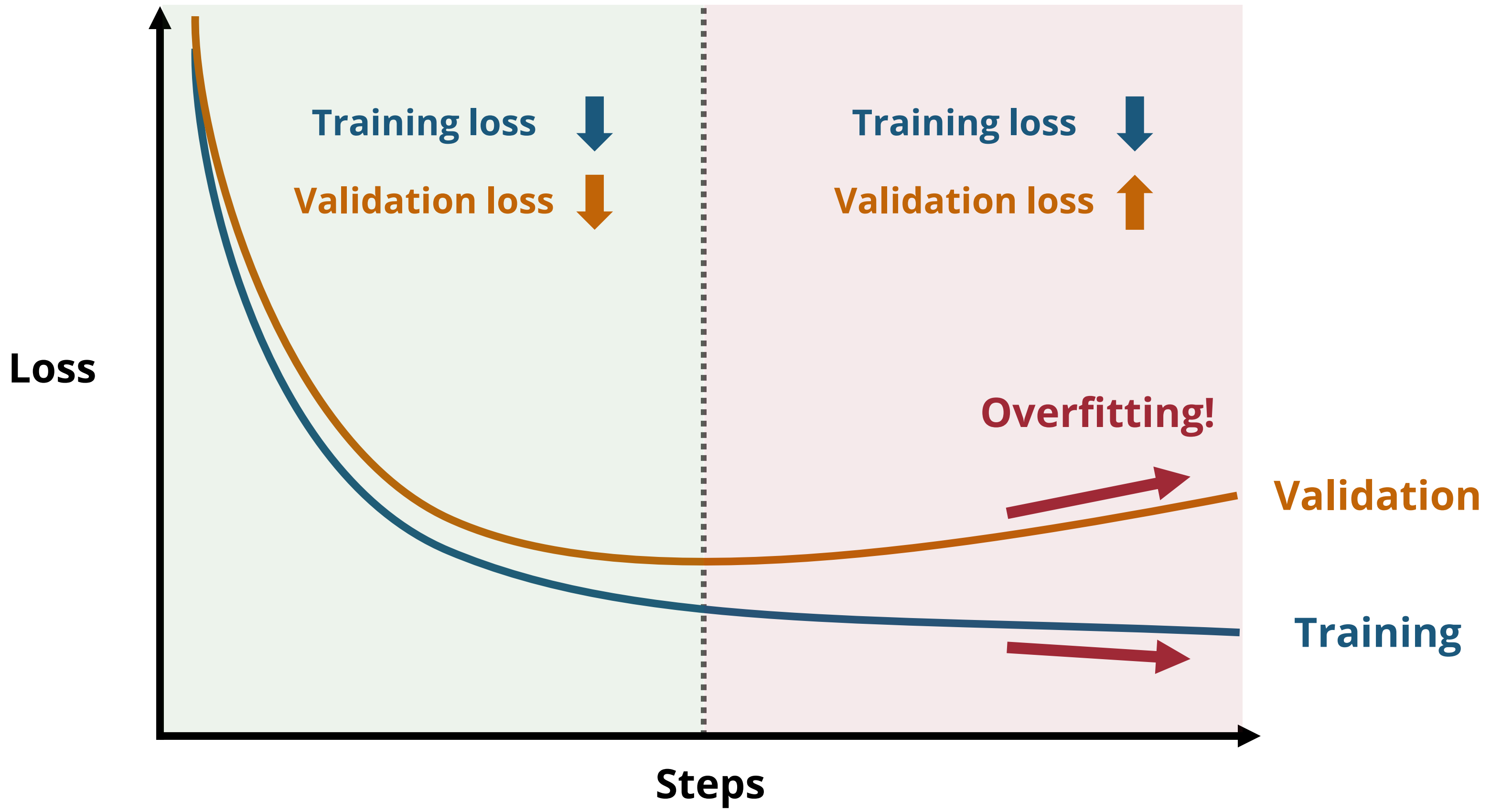
- A neural network represents **a set of functions**



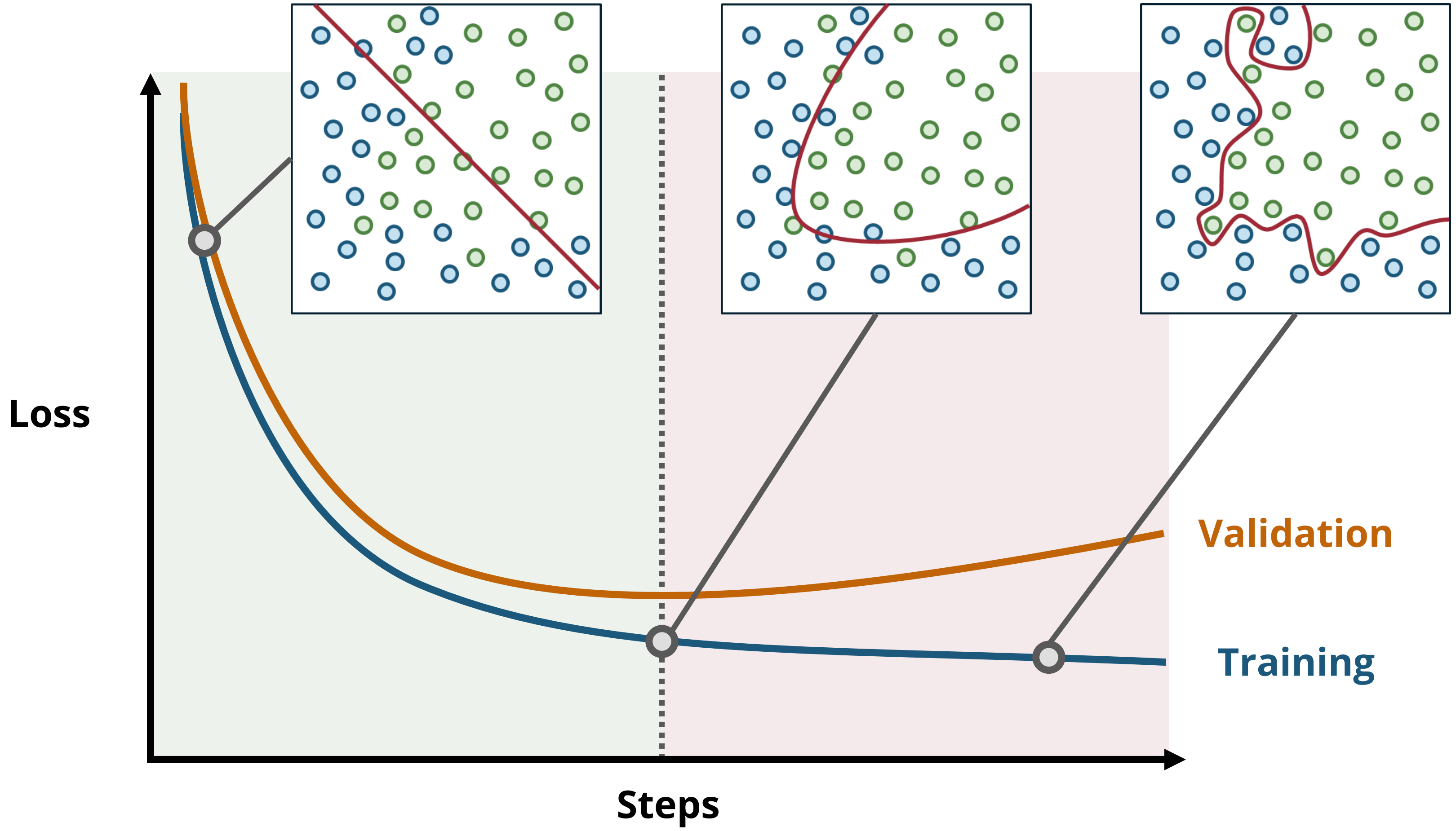
Training–Validation–Test Pipeline



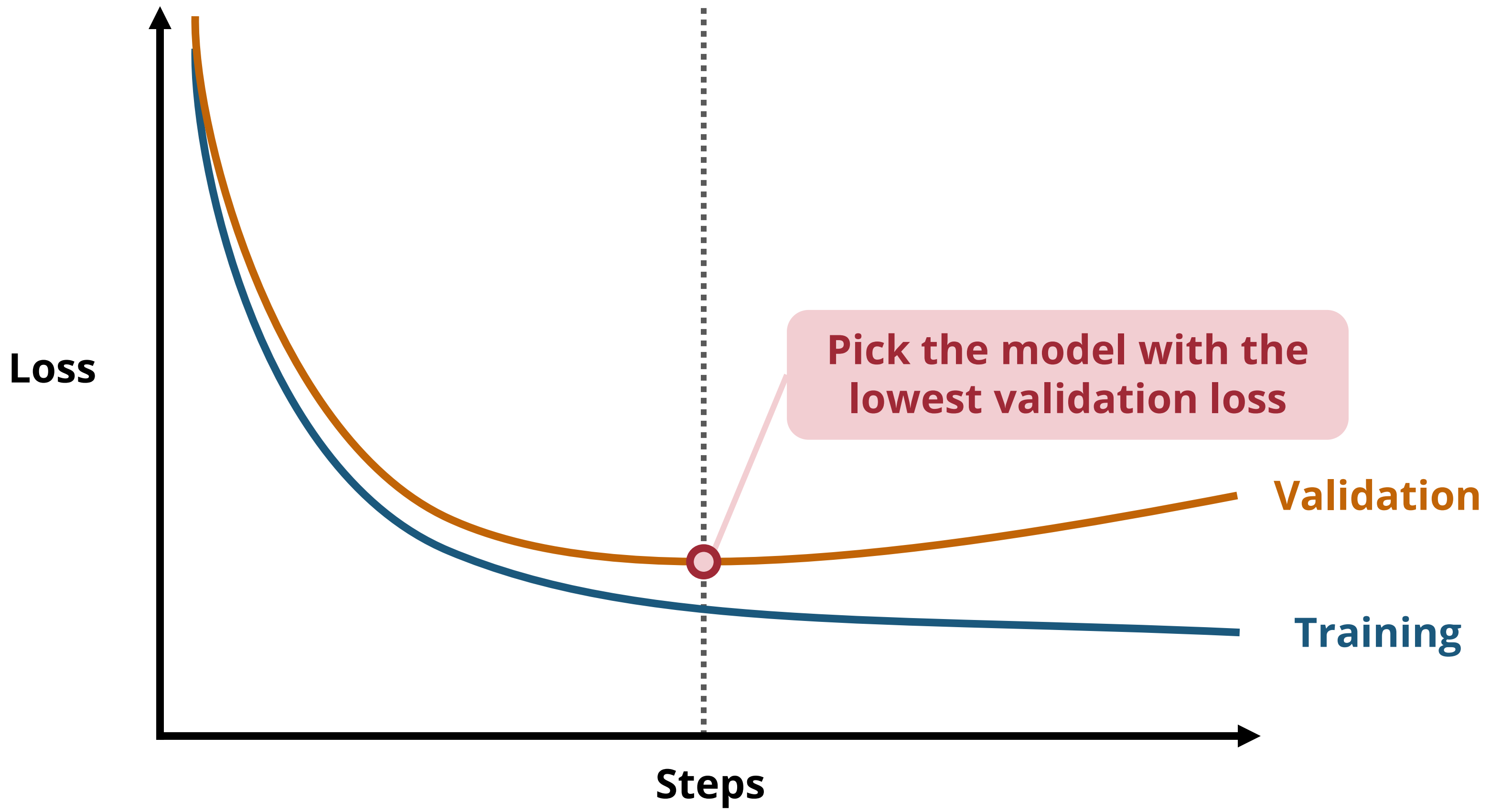
Training vs Validation Losses



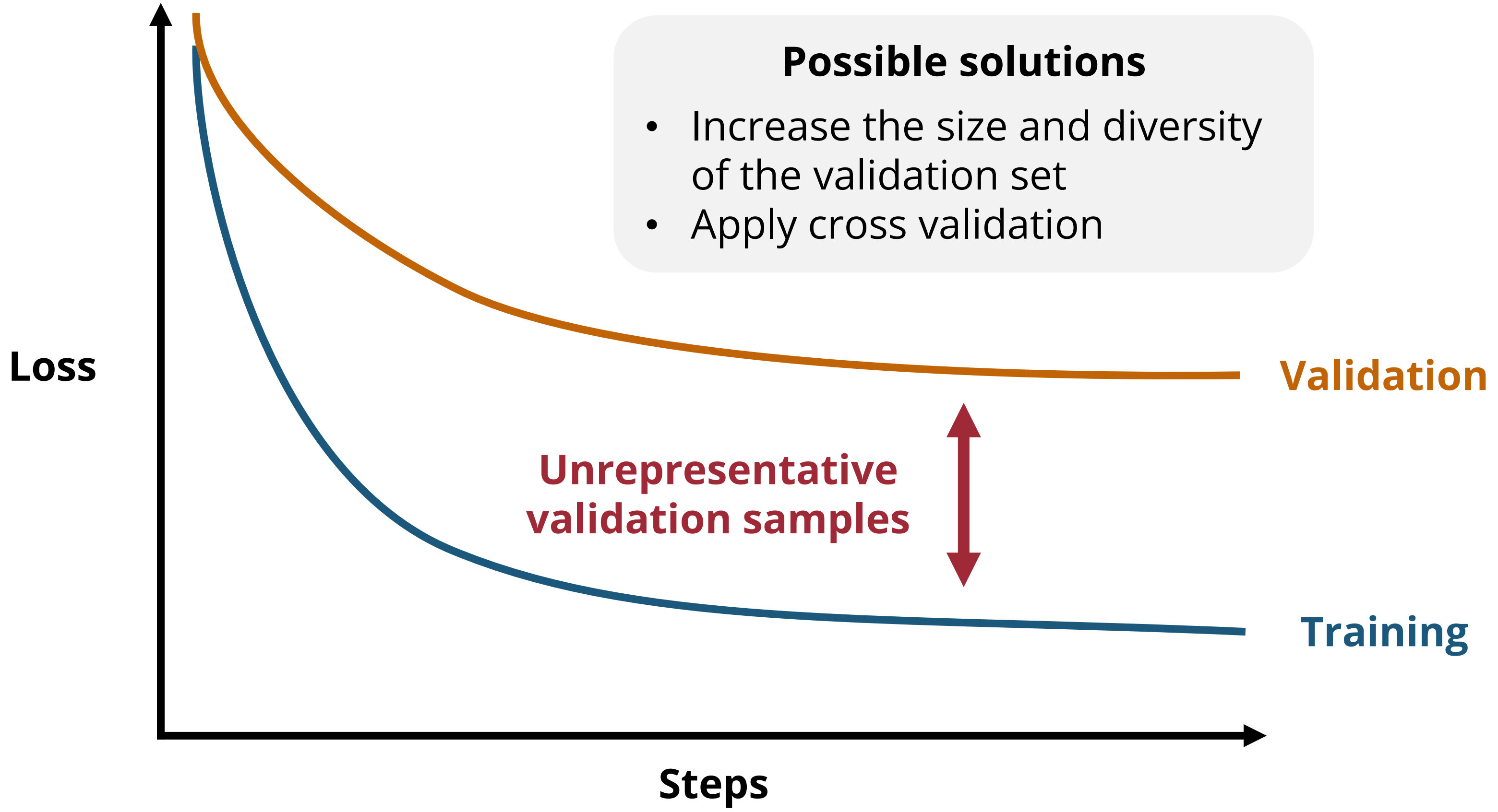
Training vs Validation Losses



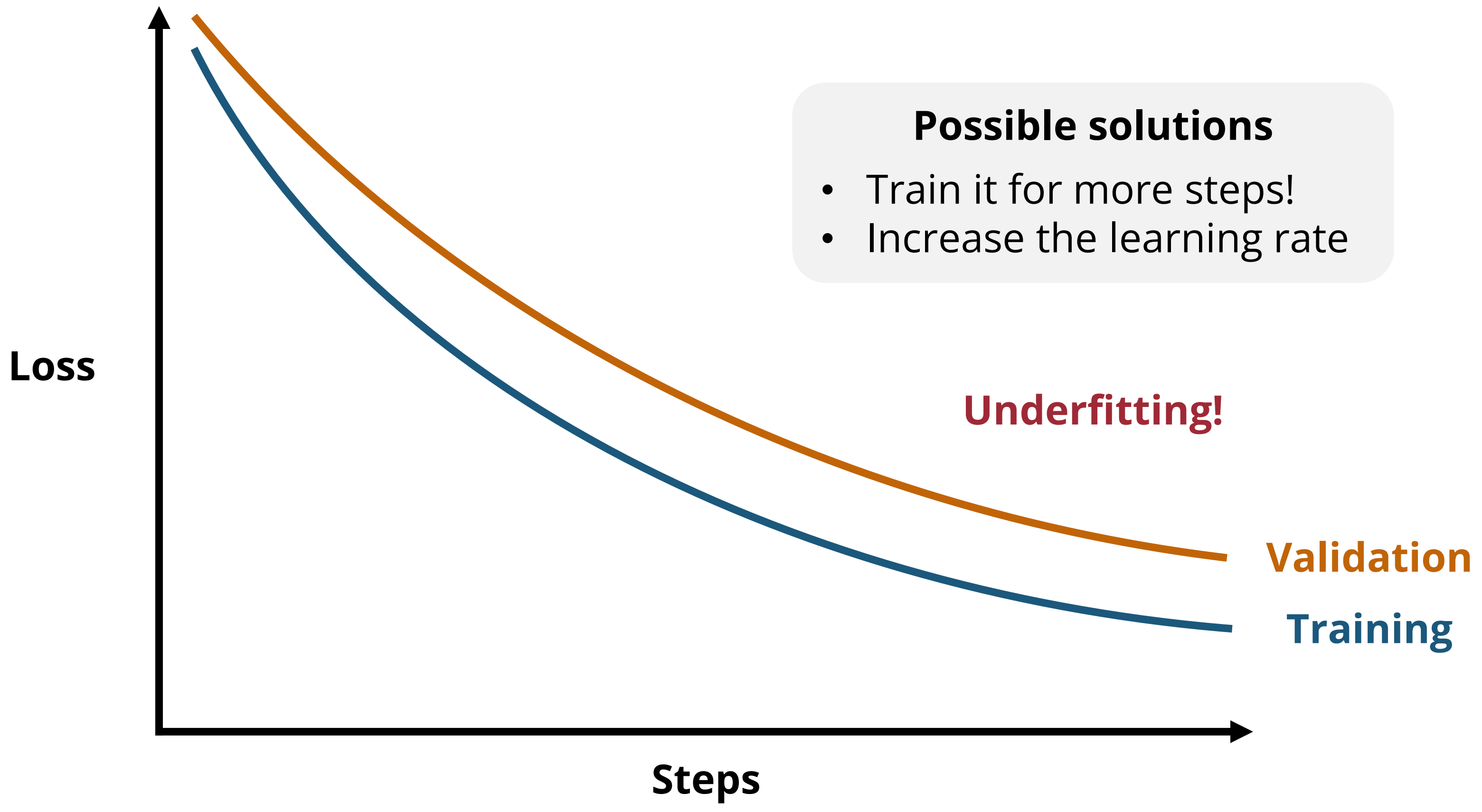
Training vs Validation Losses



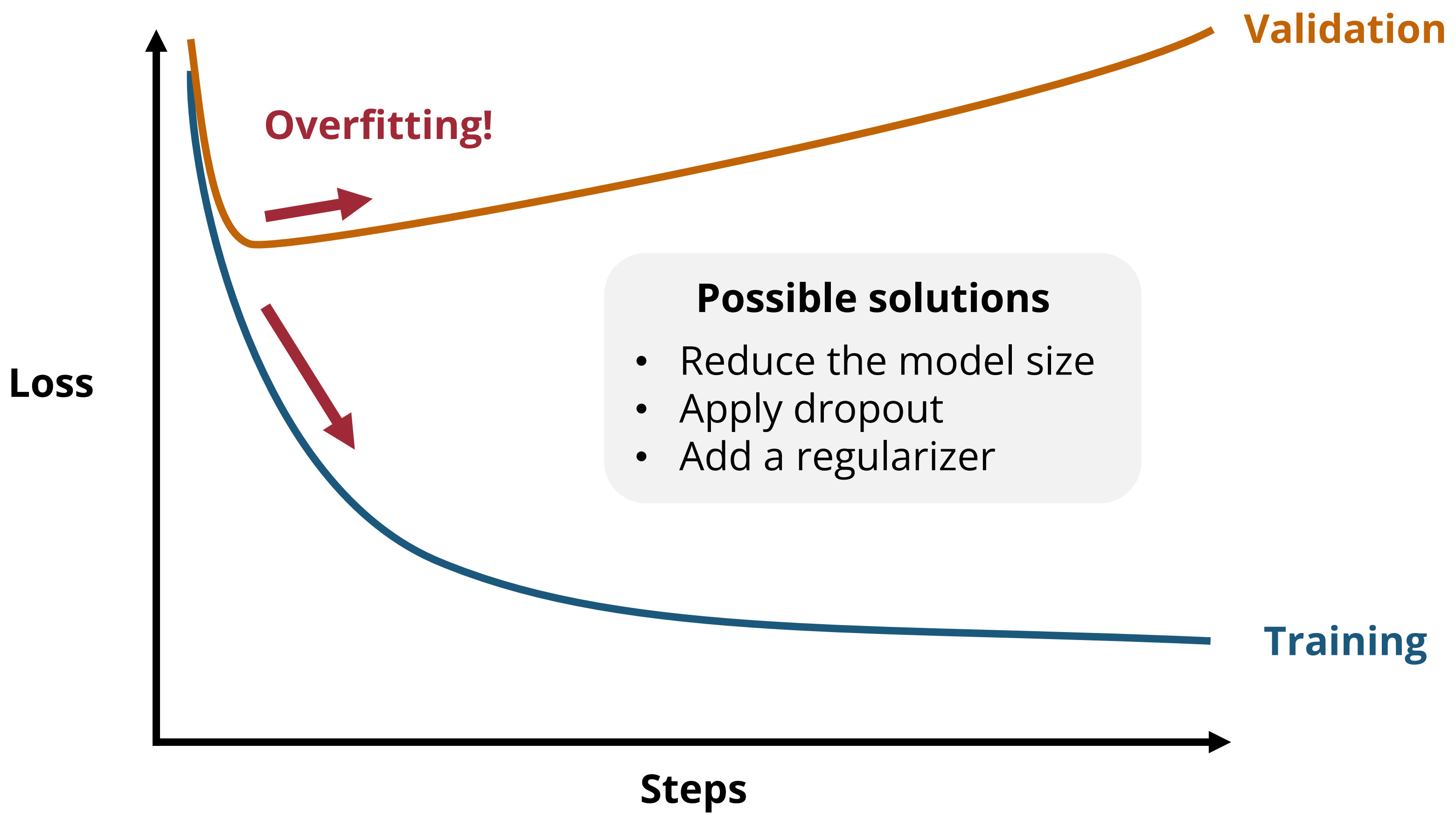
Training vs Validation Losses



Training vs Validation Losses



Training vs Validation Losses



Possible solutions

- Reduce the model size
- Apply dropout
- Add a regularizer

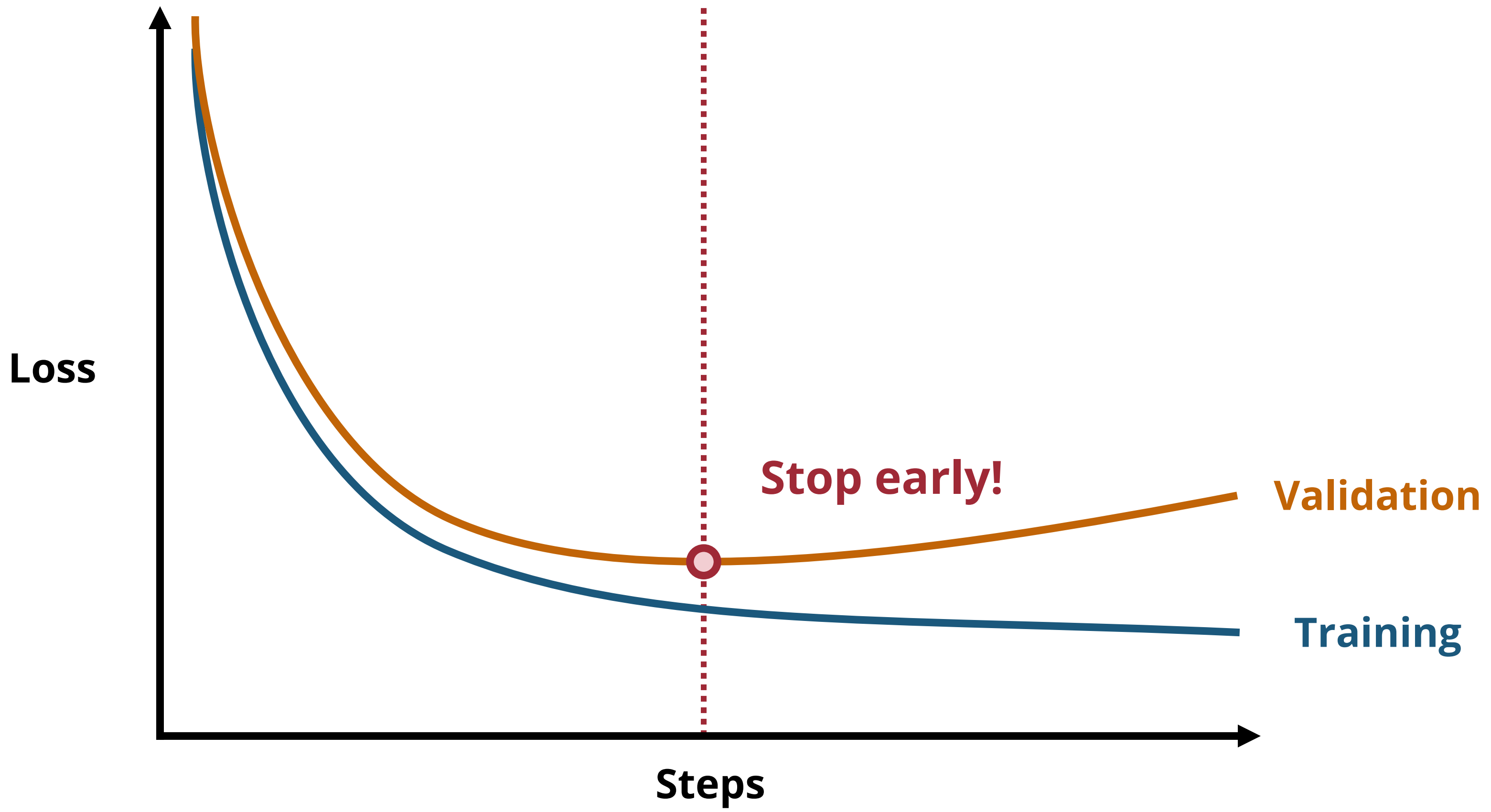
Train–Validation–Test Split

- **Keys**
  - **Never train or select your model on test samples!**
  - Don’t over-select your model on the validation set

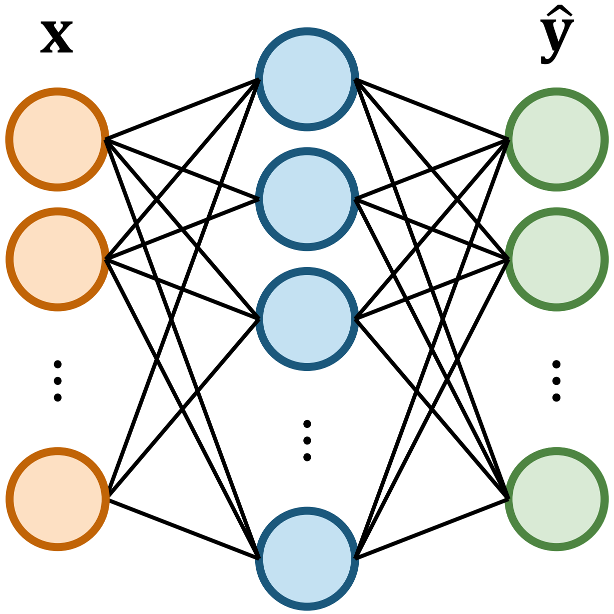
- What’s the **best ratio**?
  - Most common: **8:1:1** or 9:0.5:0.5
  - For smaller dataset, you might even want 6:2:2

Mitigating Overfitting

Early Stopping

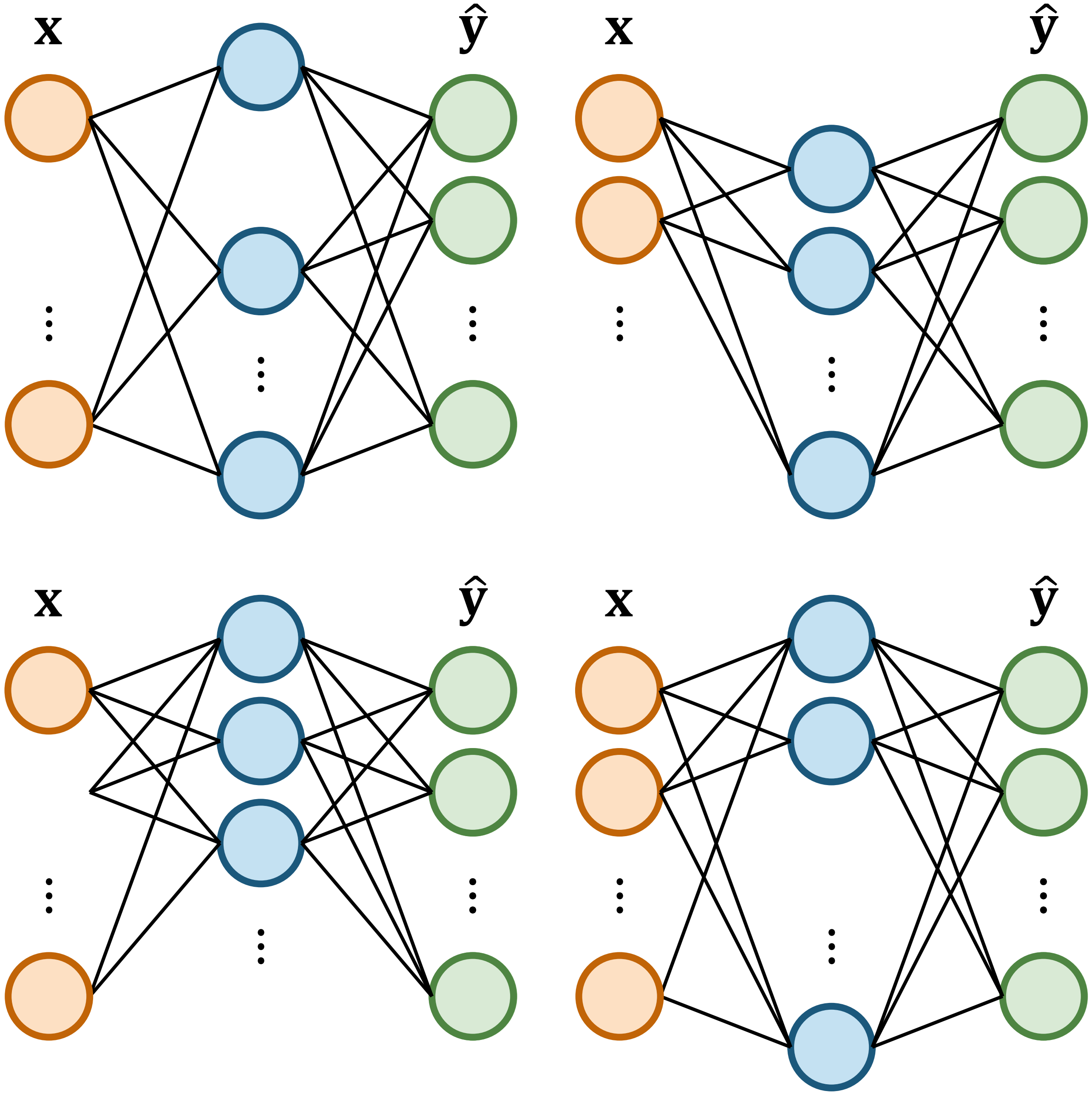


Dropout

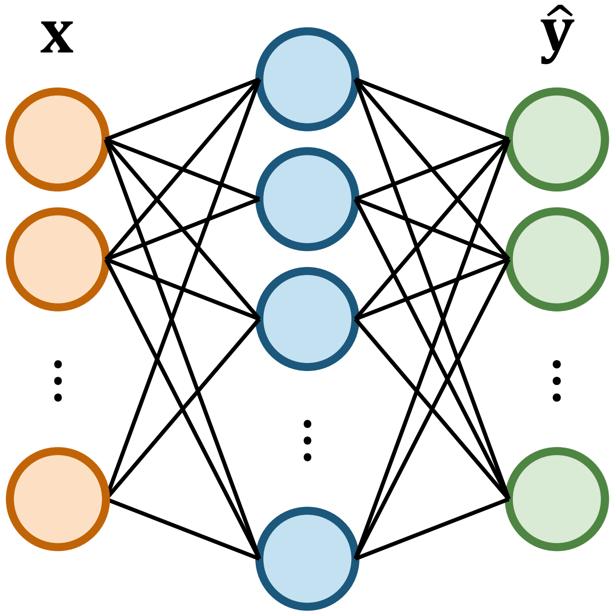


Each neuron may be removed with probability p during training

Dropout rate

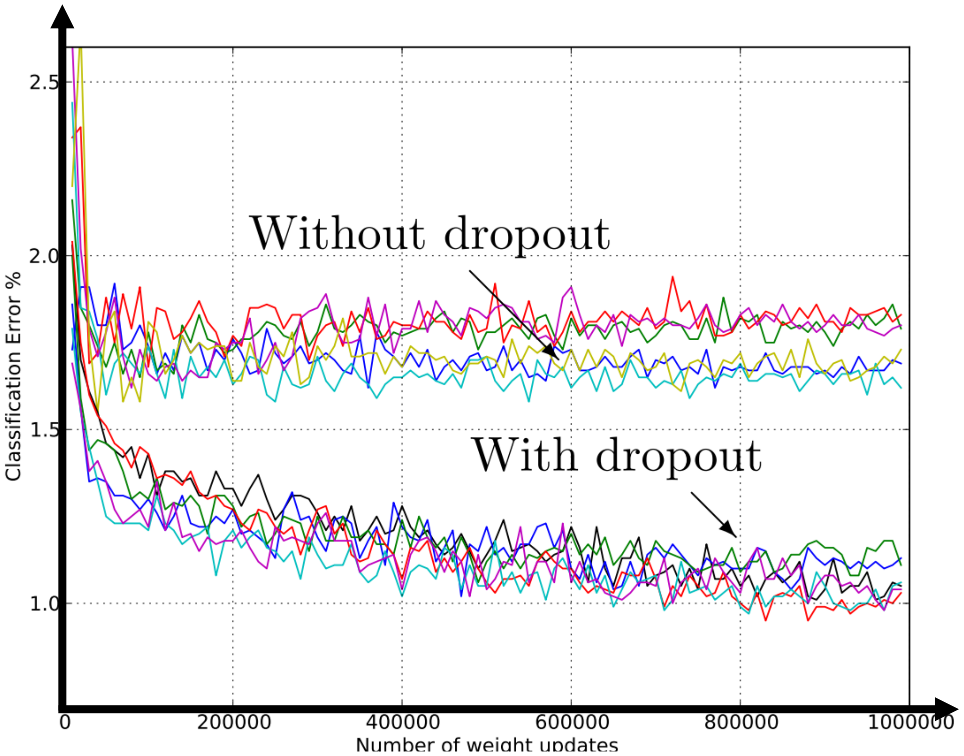


Dropout



Each neuron may be removed with probability p during training

Test error rate



Weight updates

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *JMLR*, 2014.\

Regularization Term

- A regularization term can help alleviate overfitting
  - **L1 regularization** (LASSO)

$$L' = L + \lambda(|w_1| + |w_2| + \cdots + |w_K|)$$

  - **L2 regularization** (ridge regression)

$$L' = L + \lambda\left(w_1^2 + w_2^2 + \cdots + w_K^2\right)$$

Both L1 and L2 regularizations encourage smaller weights

Adaptive Optimizers

Training a Neural Network

Build a neural network
(which defines a set of functions)

$$\hat{\mathbf{y}} = f_{\boldsymbol{\theta}}(\mathbf{x})$$

↓

Define the objective
(i.e., what is good for a function)

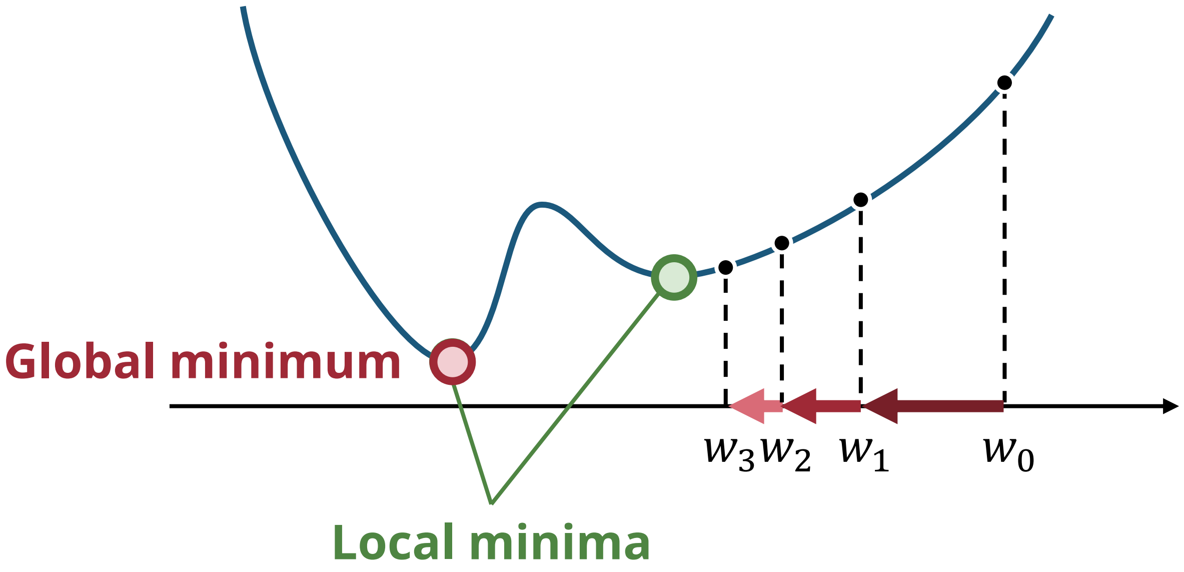
$$Loss(\boldsymbol{\theta}) = \sum_{k}^{N} L(\hat{\mathbf{y}}_k, \mathbf{y}_k)$$

↓

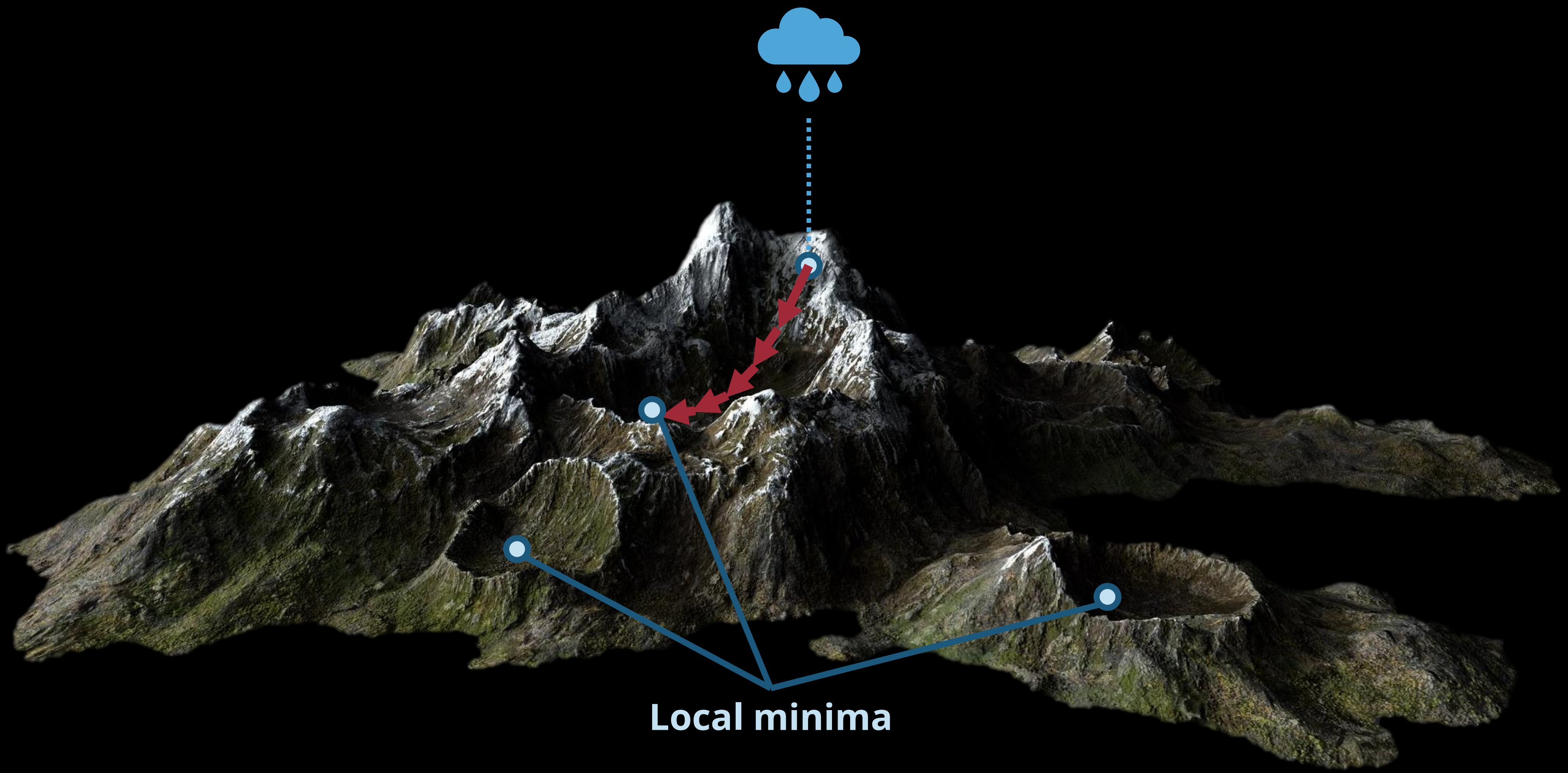
Find the optimal parameters
(which leads to the best function)

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$$

Gradient Descent Finds a Local Minimum

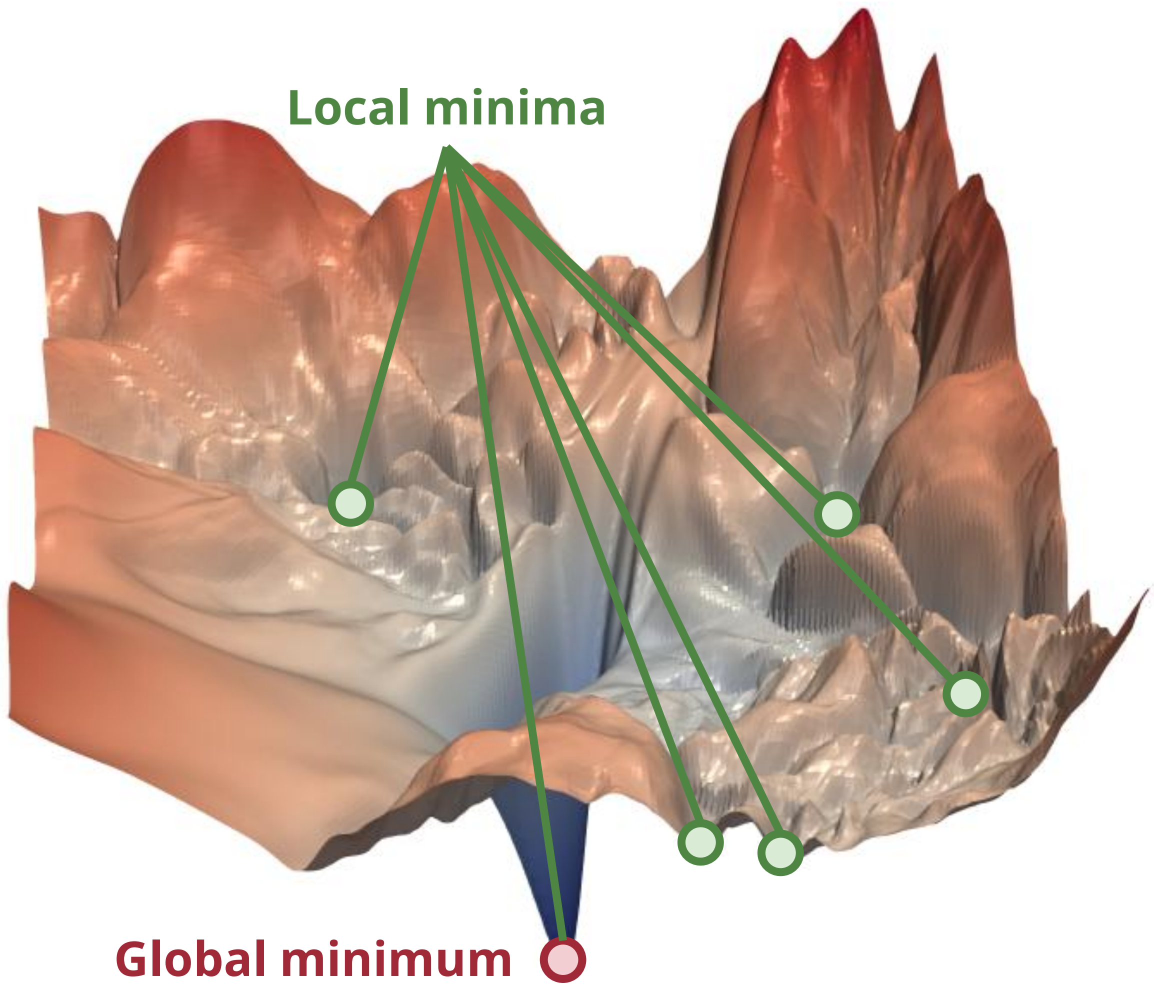


Gradient Descent Finds a Local Minimum



substance3d.adobe.com/community-assets/assets/9fe77d4279e12fe2fbd90b8f7ca18146a565f7ee

Local Minima in Complex Loss Landscape



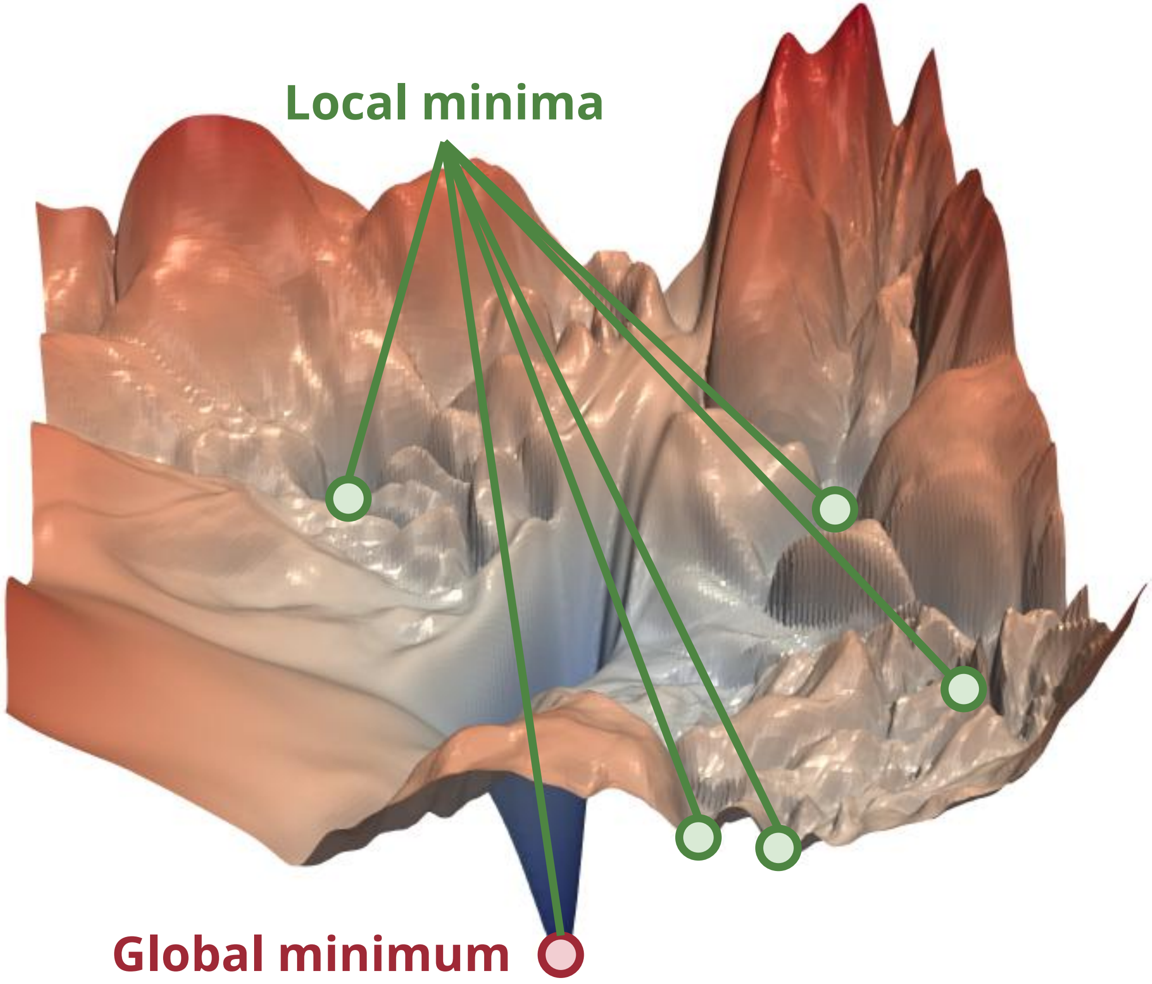
Solution 1
Use an optimizer with adaptive learning rate

Solution 2
Use a stochastic optimizer

Solution 3
Make the loss landscape smoother

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein, "Visualizing the Loss Landscape of Neural Nets," *NeurIPS*, 2018.

Local Minima in Complex Loss Landscape



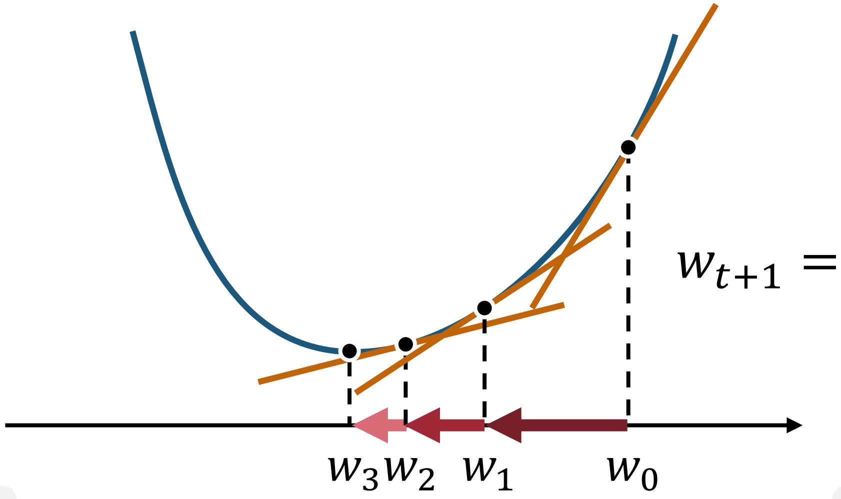
Solution 1
Use an optimizer with adaptive learning rate

Solution 2
Use a stochastic optimizer

Solution 3
Make the loss landscape smoother

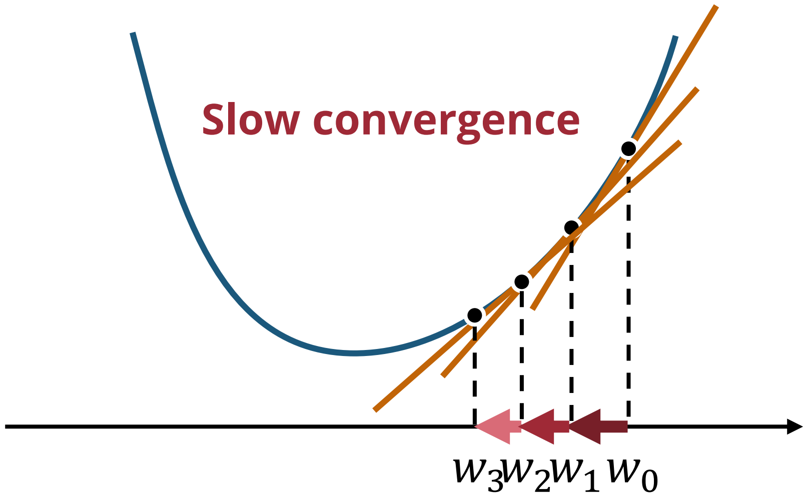
Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein, "Visualizing the Loss Landscape of Neural Nets," *NeurIPS*, 2018.

Learning Rate in Gradient Descent

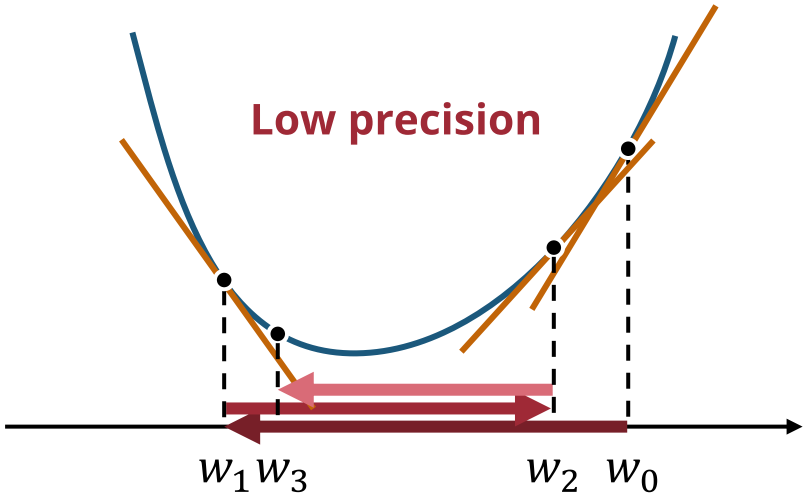


$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

Smaller learning rate



Larger learning rate

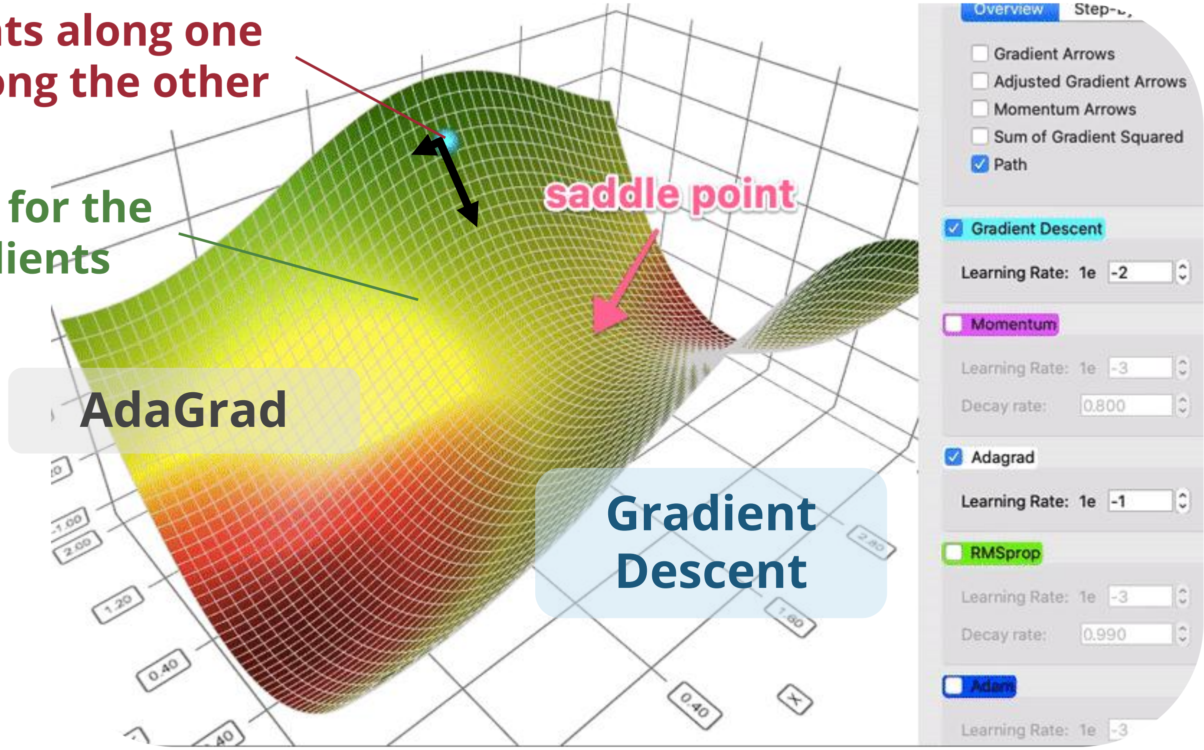


Gradient-based Adaptive Learning Rate

- **Intuition**: **Compensate axis that has little progress** by comparing the current gradients to the previous gradients

Large gradients along one axis, small along the other

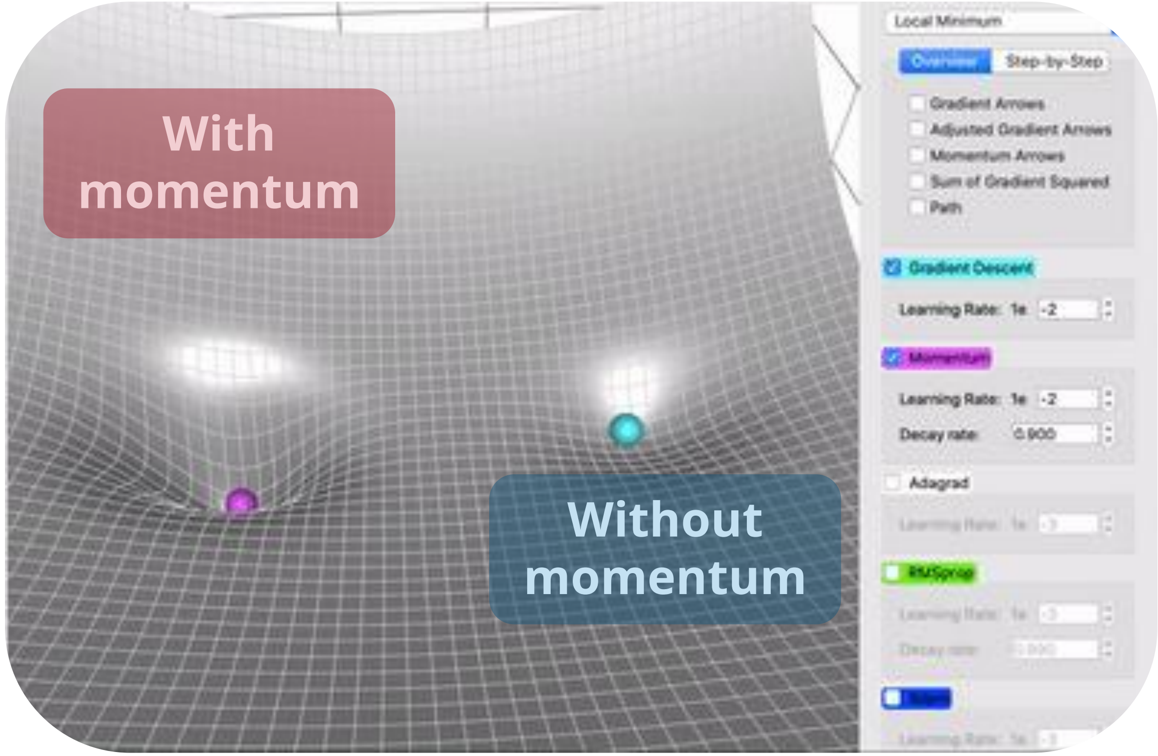
Use larger learning rate for the axis with smaller gradients



towardsdatascience.com/a-visual-explanation-of-gradient-descent-methods-momentum-adagrad-rmsprop-adam-f898b102325c

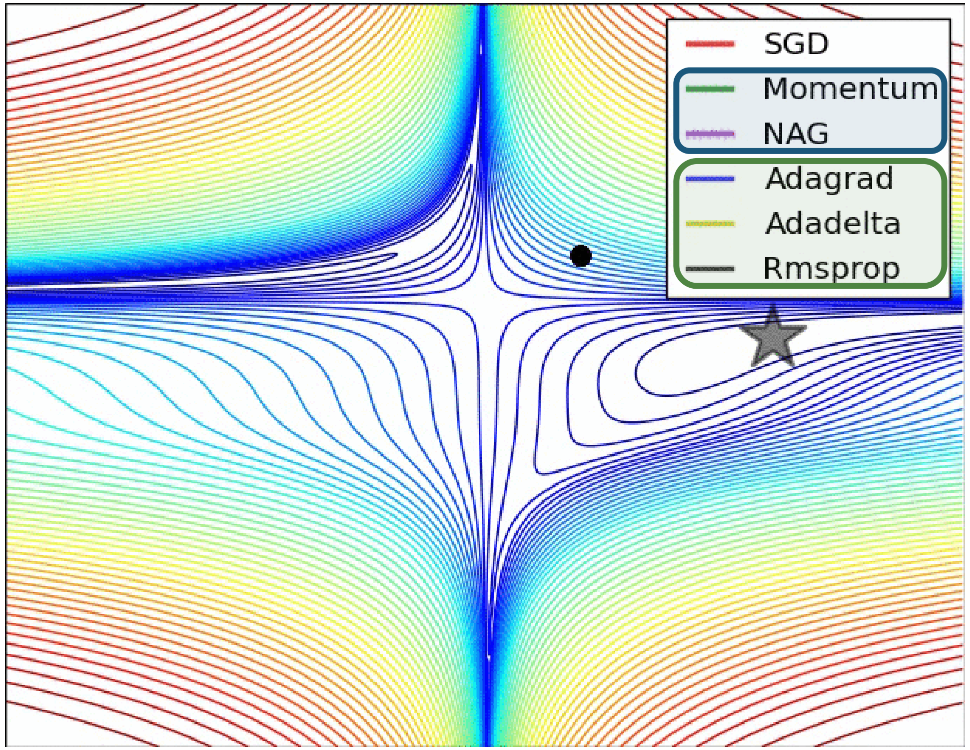
Momentum

- **Intuition**: Maintain the momentum to **escape from local minima**



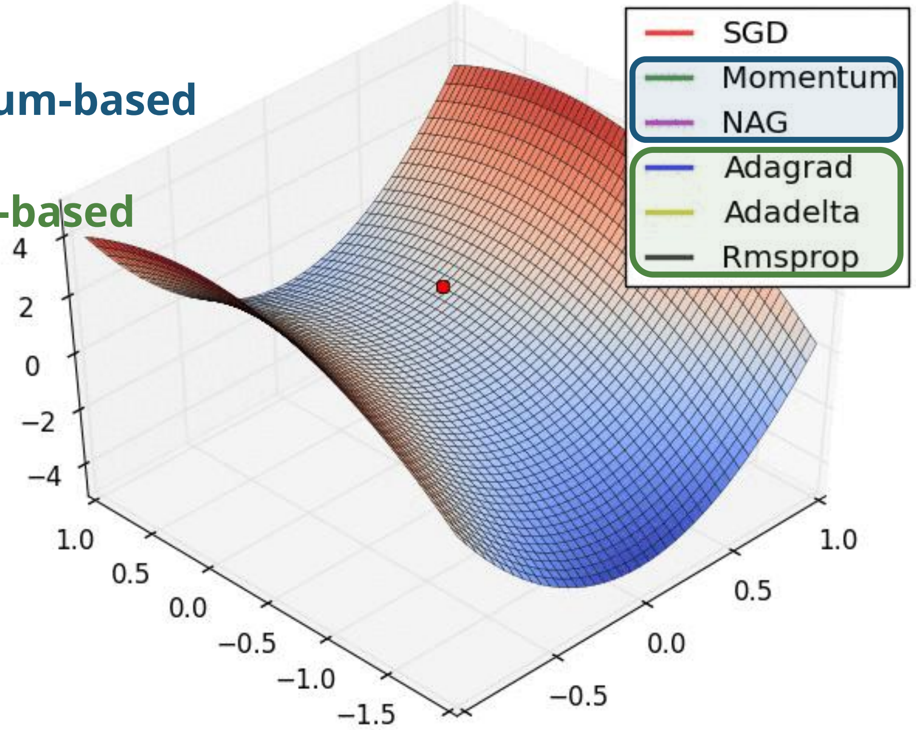
towardsdatascience.com/a-visual-explanation-of-gradient-descent-methods-momentum-adagrad-rmsprop-adam-f898b102325c

Comparison of Optimizers



Momentum-based

Gradient-based

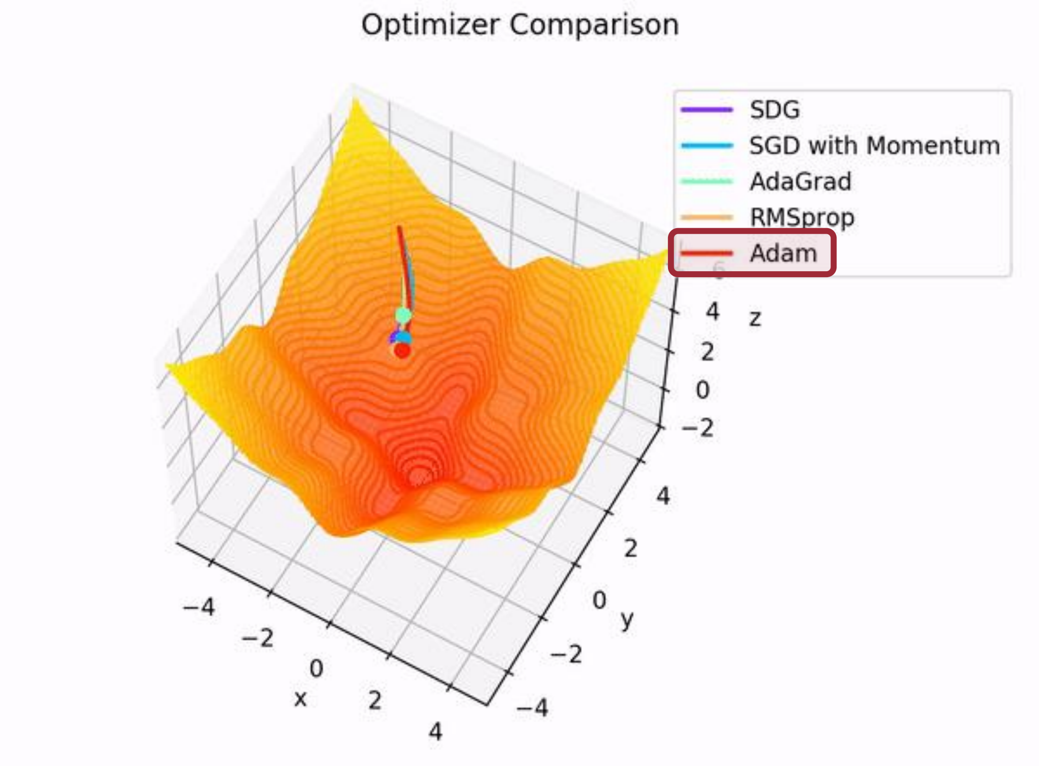


Can we combine them?

analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/

Adam Optimizer

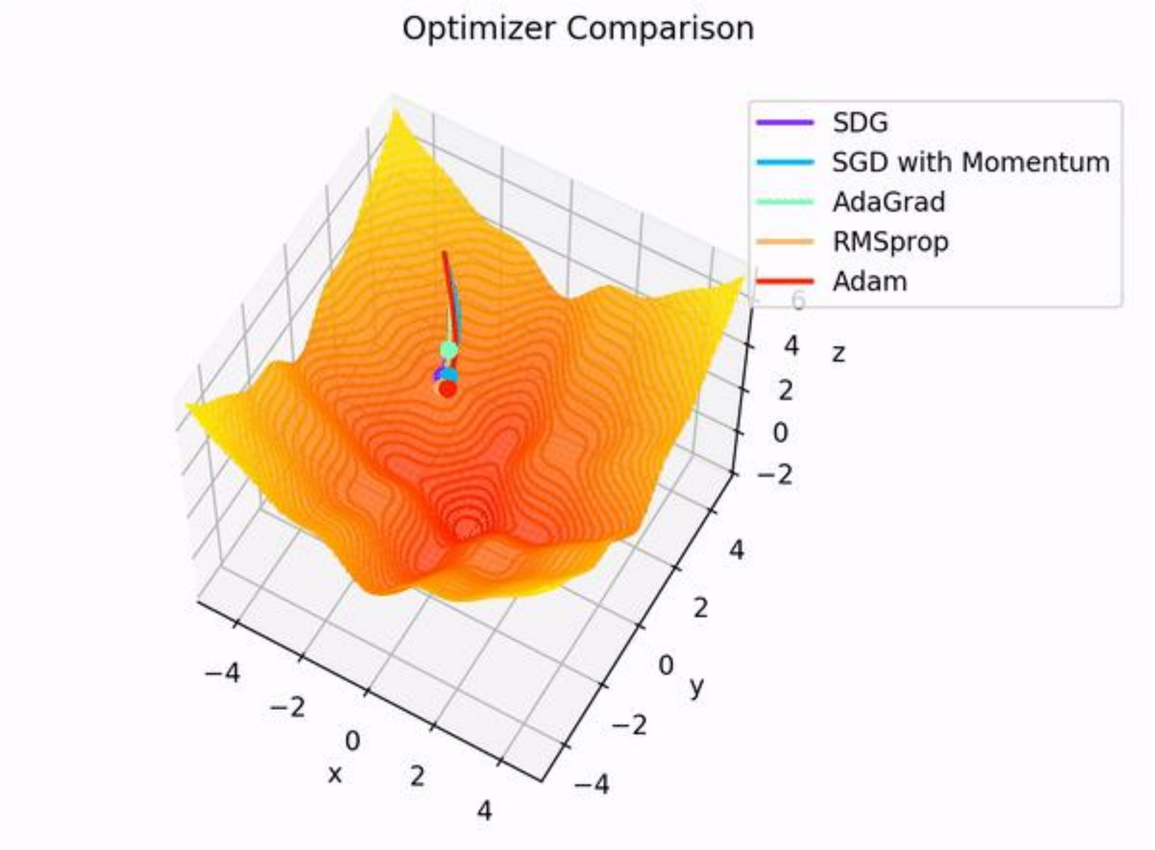
- Combine the idea of **adaptive learning rate** and **momentum**
- Work **empirically well** in complex neural network
- The **go-to choice** for most cases



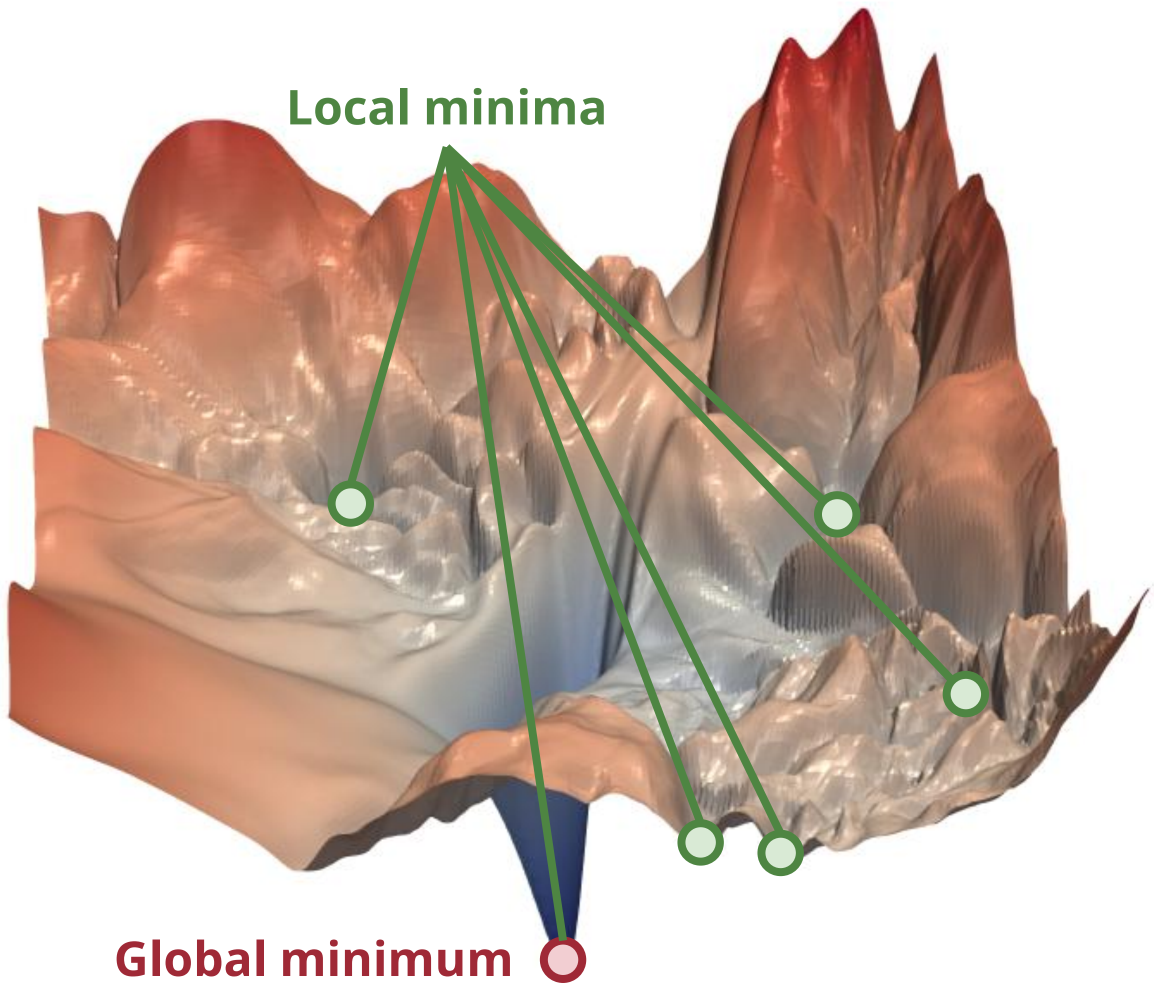
Comparison of Optimizers

- **Momentum**
  - Gets you out of spurious local minima
  - Allows the model to explore around

- **Gradient-based adaption**
  - Maintains steady improvement
  - Allows faster convergence



Local Minima in Complex Loss Landscape



Solution 1
Use an optimizer with adaptive learning rate

Solution 2
Use a stochastic optimizer

Solution 3
Make the loss landscape smoother

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein, "Visualizing the Loss Landscape of Neural Nets," *NeurIPS*, 2018.

Batch Gradient Descent

- How to aggregate the gradients obtained from different training samples?
- Batch gradient descent computes the mean gradients **over the whole training set**

MSE loss

$$Loss(\boldsymbol{\theta}) = \sum_{k}^{N} L(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{n} \sum_{k}^{N} \sum_{i}^{n} \left(\hat{y}_i^{(k)} - y_i^{(k)}\right)^2$$

Binary cross entropy

$$Loss(\boldsymbol{\theta}) = \sum_{k}^{N} L(\hat{y}, y) = \sum_{k}^{N} -y \log \hat{y} - (1 - y) \log(1 - \hat{y})$$

Cross entropy

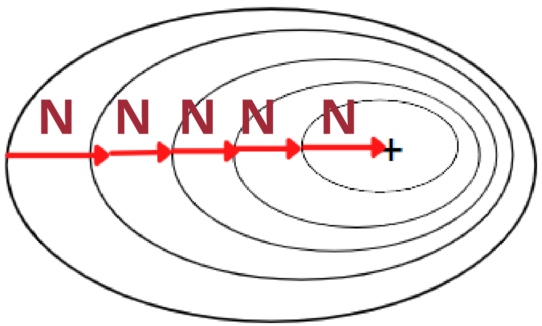
$$Loss(\boldsymbol{\theta}) = \sum_{k}^{N} L(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{k}^{N} \sum_{i}^{n} y_i \log \hat{y}_i$$

Stochastic Gradient Descent (SGD)

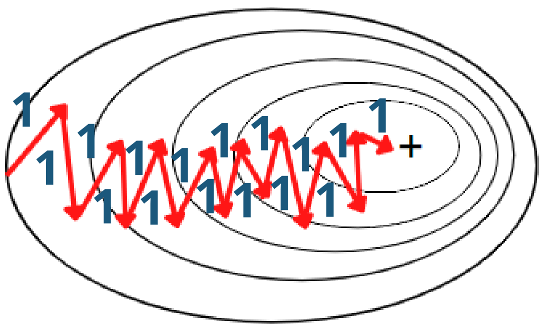
- **Intuition:** **Estimate** the gradient using **one random training sample**
- Benefits
  - **Speed up** the computation of the gradient **N computations → 1 computation**
  - Add some **randomness** to the gradient descent algorithm **Help escape spurious local minima**

Gradient descent

5N gradient computations



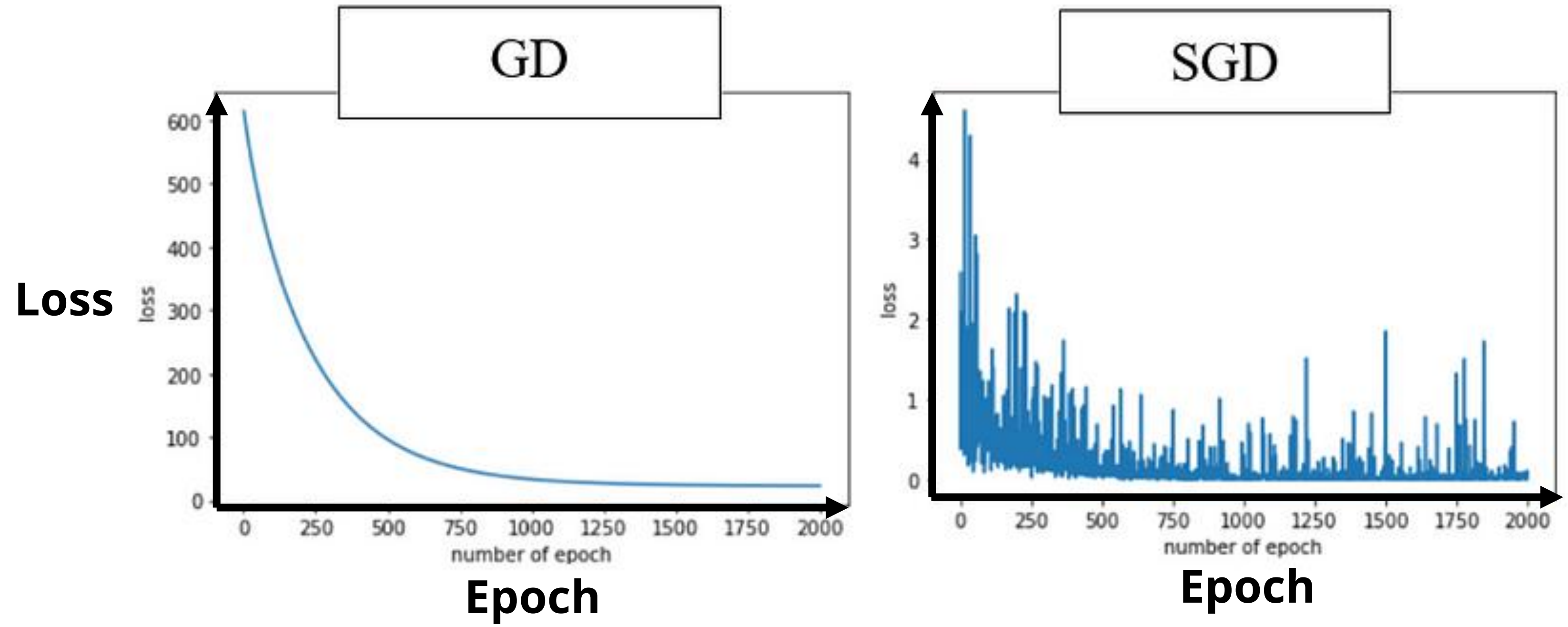
Stochastic gradient descent



16 gradient computations

Stochastic Gradient Descent is Noisy and Unstable

- Gradient estimate using one single sample can be unreliable

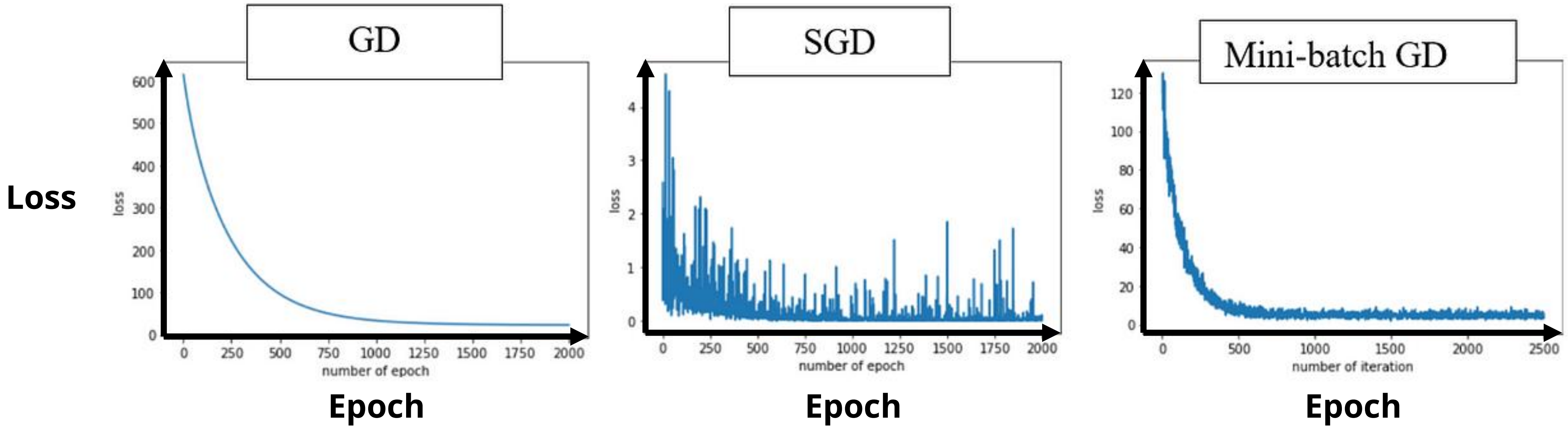


How about we use more samples to estimate the gradient?

towardsdatascience.com/deep-learning-optimizers-436171c9e23f

Mini-batch Gradient Descent

- **Intuition:** **Estimate** the gradient using **several random training samples**



batch size $= N$

batch size $= 1$

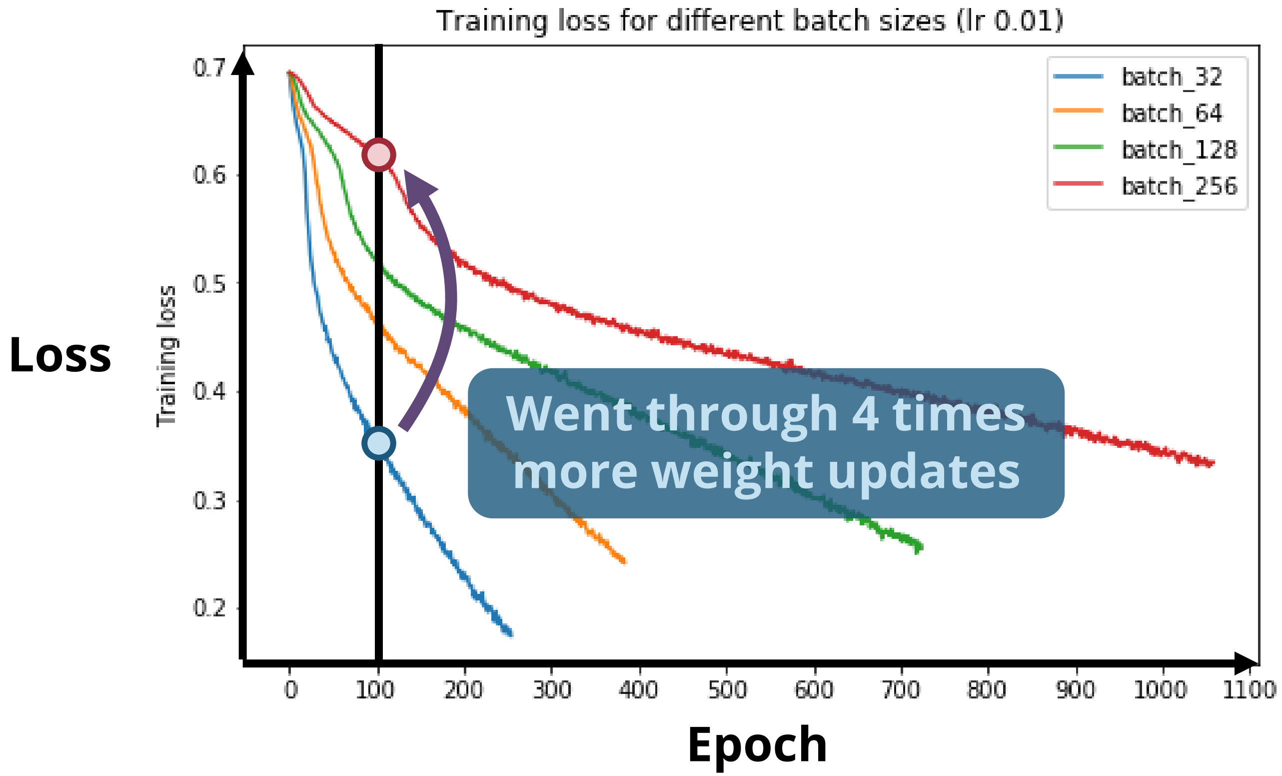
$1 <$ batch size $< N$

analyticsvidhya.com/blog/2022/07/gradient-descent-and-its-types/

Effects of Batch Size

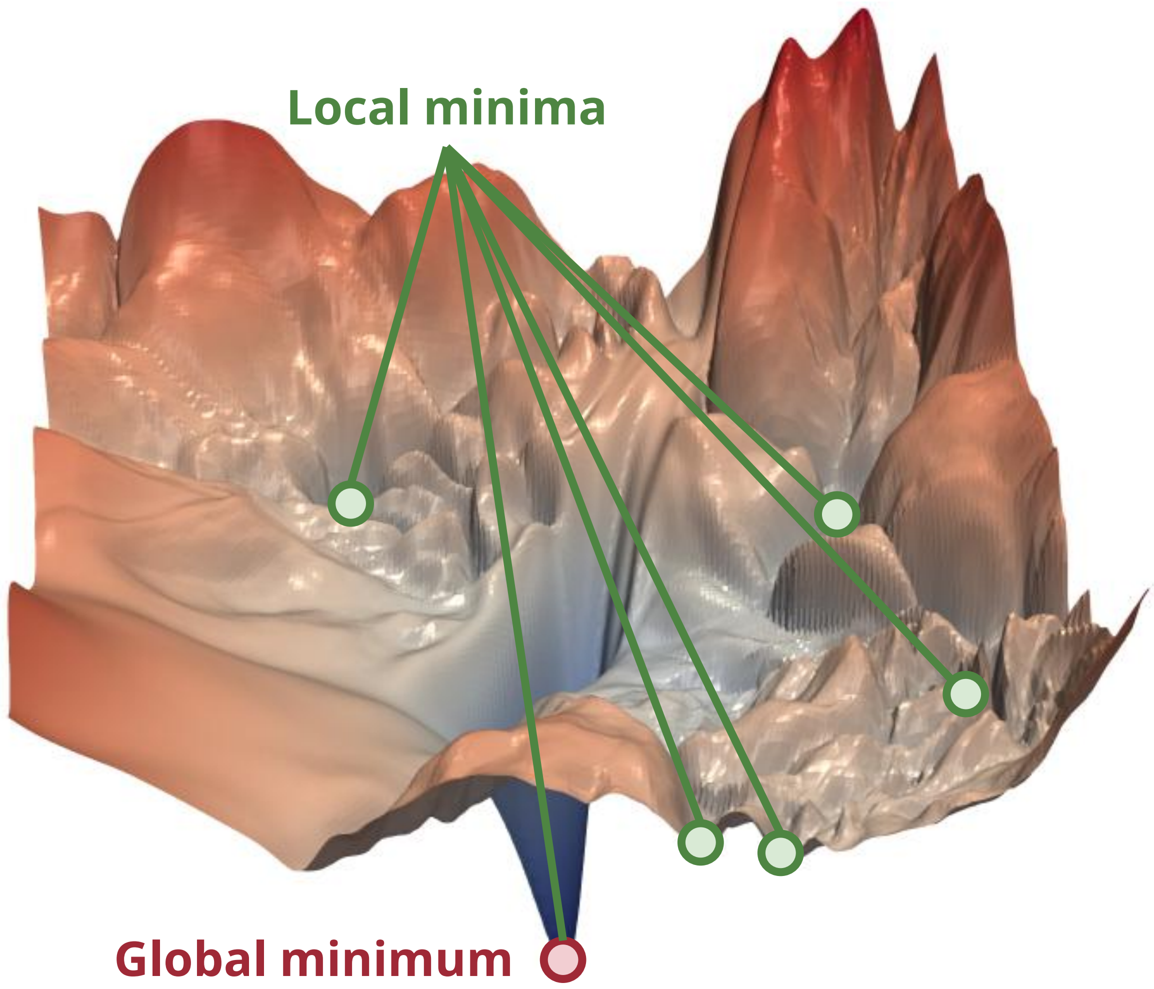
- An **epoch** is a full run of the whole dataset
- Steps per epoch depends on the batch size

$$\#(\mathbf{steps}) = \frac{\#(\mathbf{training\ samples})}{\mathbf{batch\ size}}$$



medium.com/deep-learning-experiments/effect-of-batch-size-on-neural-net-training-c5ae8516e57

Local Minima in Complex Loss Landscape



Solution 1
Use an optimizer with adaptive learning rate

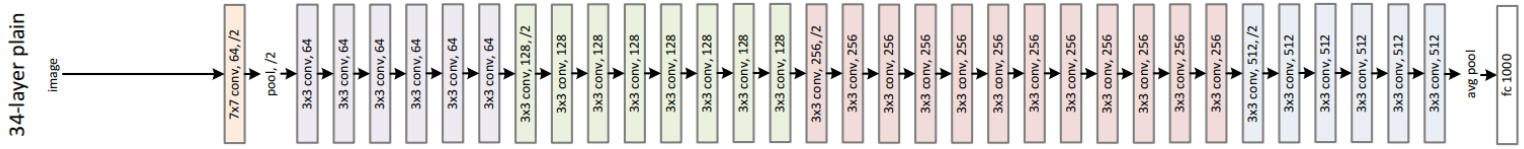
Solution 2
Use a stochastic optimizer

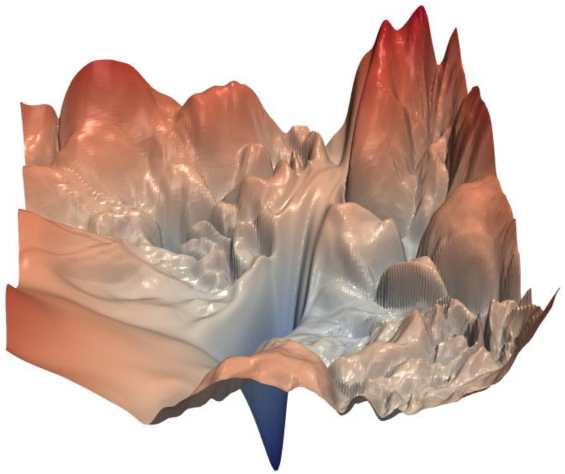
Solution 3
Make the loss landscape smoother

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein, "Visualizing the Loss Landscape of Neural Nets," *NeurIPS*, 2018.

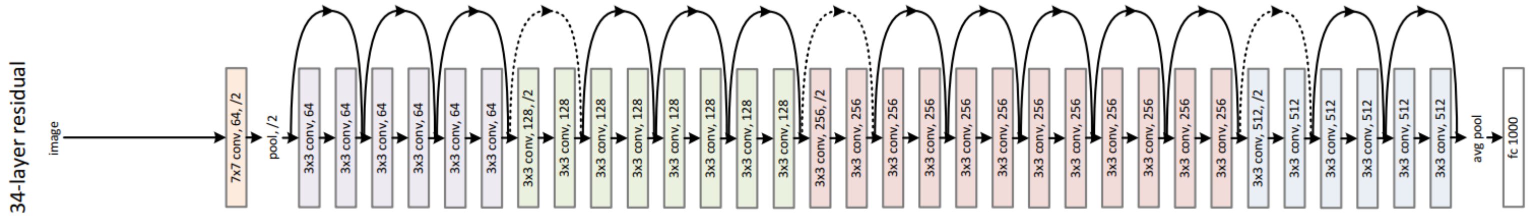
Skip Connections

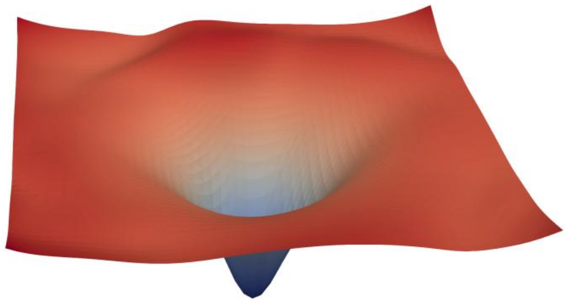
Without skip connections





With skip connections



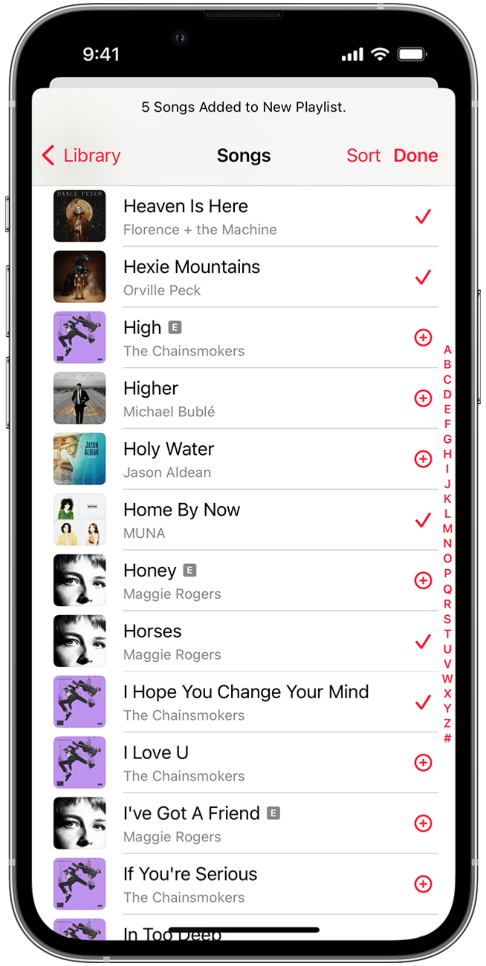


Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein, "Visualizing the Loss Landscape of Neural Nets," *NeurIPS*, 2018.

Recap

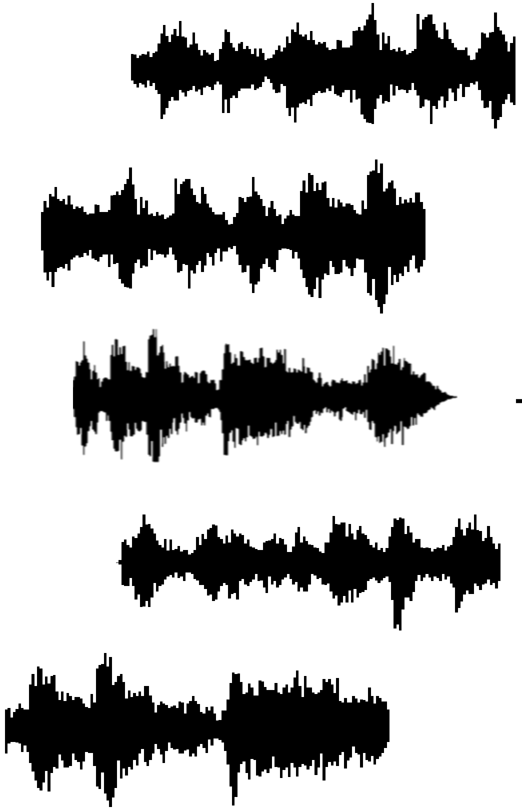
Music Classification for Recommendation

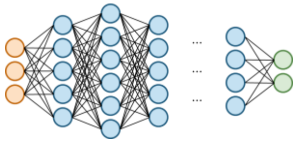
Music Classification for Playlist Generation



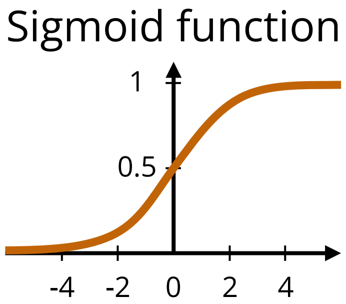
Binary Classification

Is Classical?

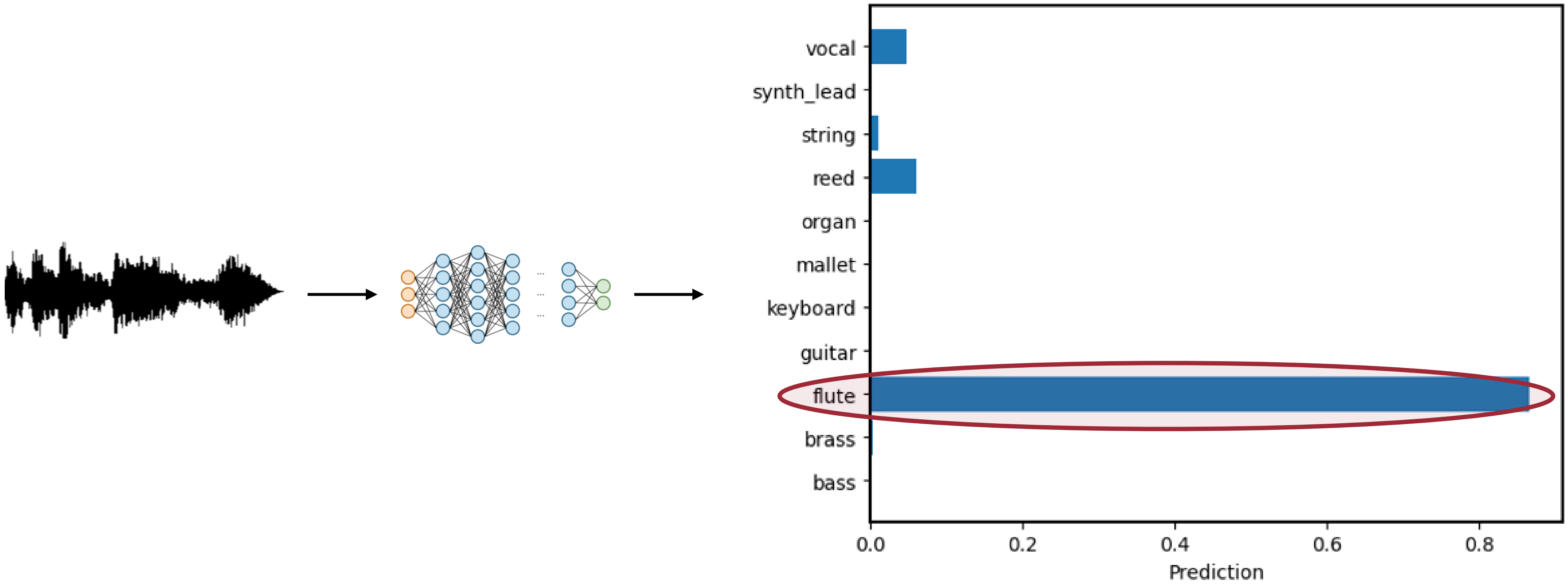




4.86	**0.99**	
-1.87	**0.13**	
2.10	**0.89**	
-3.52	**0.03**	
0.18	**0.54**	



Multiclass Classification



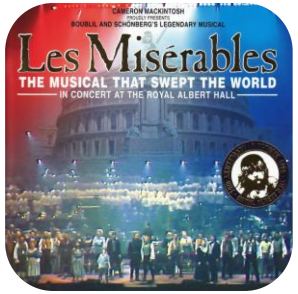
Multi-label Classification

Soul, R&B, pop

Soundtrack, classical

Rock

Classical



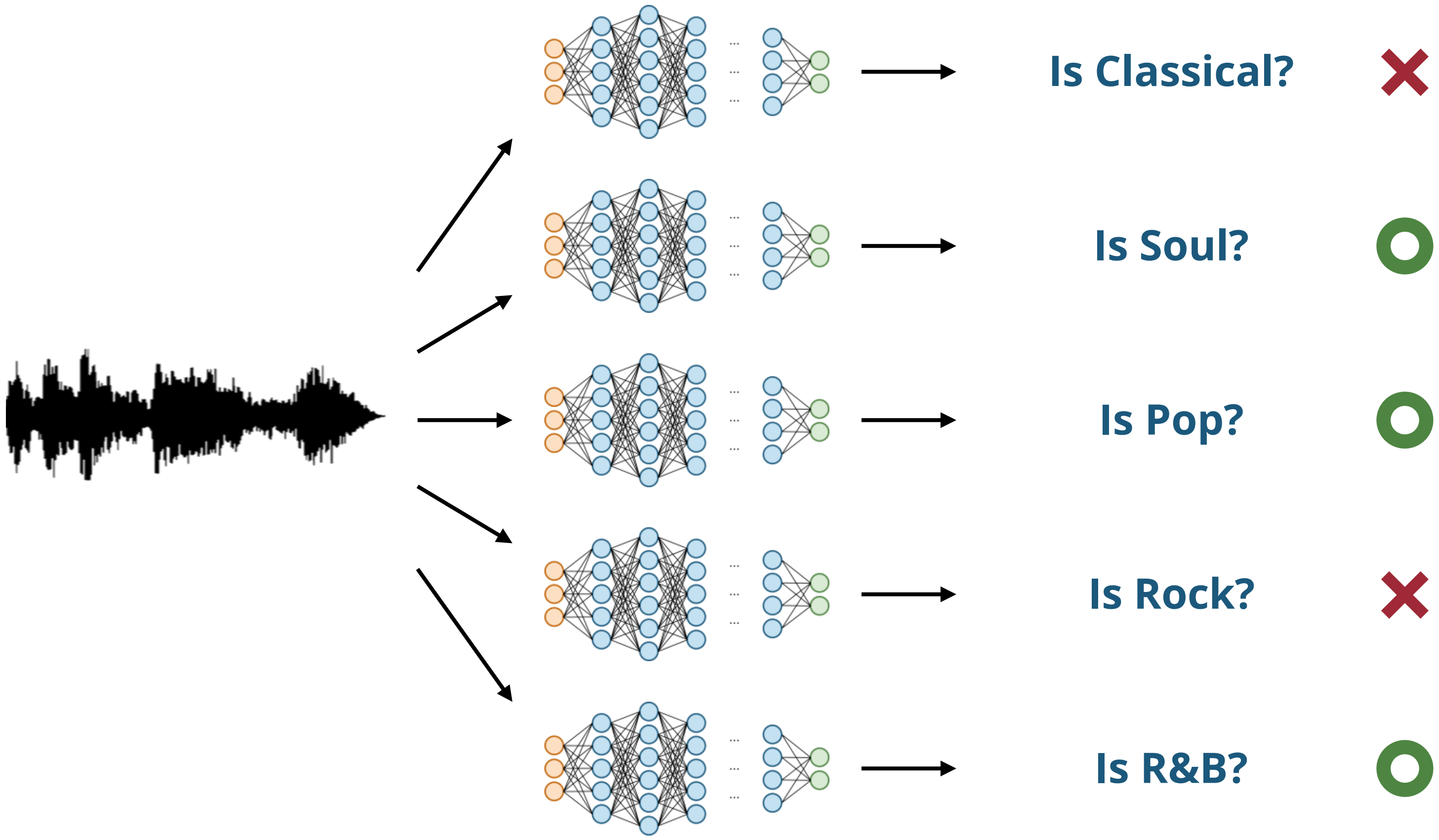
Musical

Alternative, rock

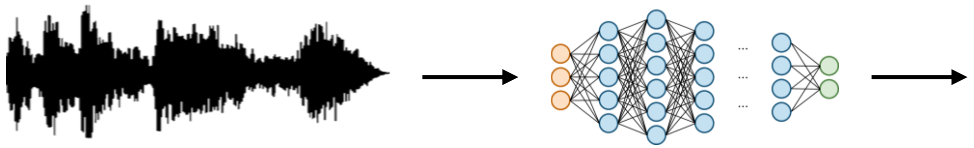
Pop

Pop, soul, R&B

Multi-label Classification as Binary Classification



Multi-label Classification as Multi-class Classification



{Pop}

{R&B}

{Soul}

{Classical}

{Pop, R&B}

{Pop, Soul}

…

{Pop, Classical}

{Pop, R&B, Soul}

…

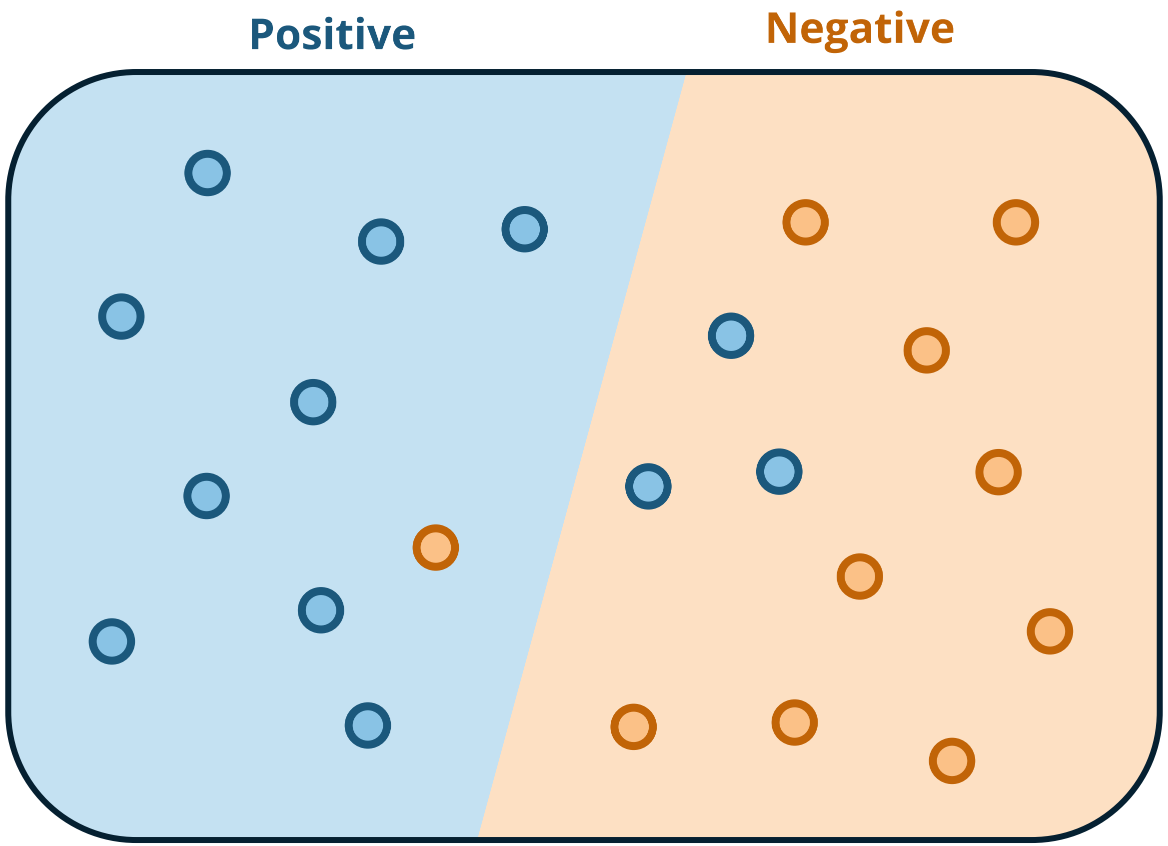
{R&B, Soul, Classical}

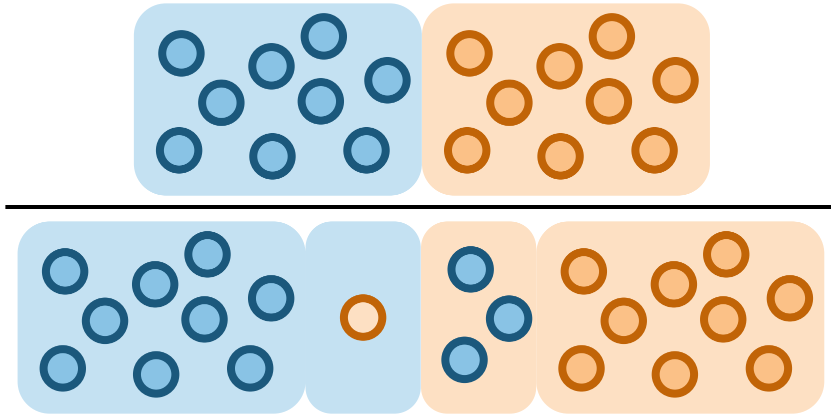
{Pop, R&B, Soul, Classical}

Label powerset

Accuracy

- **Definition**: Percentage of correct predictions across all classes



Accuracy = 

= 0.82

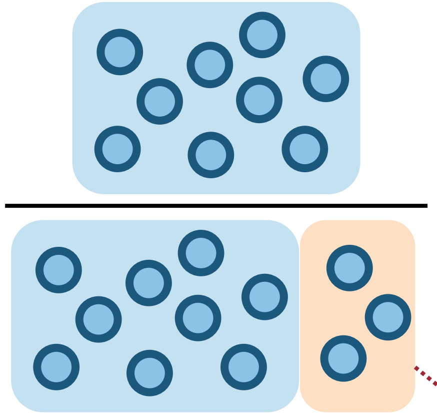
Confusion Matrix for Binary Classification

		Prediction	
		Positive	Negative
Ground truth	Positive	True Positives (TP)  Cat	False Negatives (FN)  Not Cat
	Negative	False Positives (FP)  Cat	True Negatives (TN)  Not Cat

Precision vs Recall

Precision

$$\frac{TP}{TP + FP} =$$



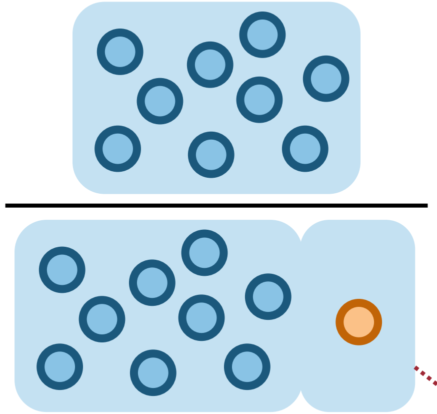
$$= 0.75$$

False alarms

How often predictions for the positive are correct

Recall

$$\frac{TP}{TP + FN} =$$



$$= 0.9$$

Missed detection

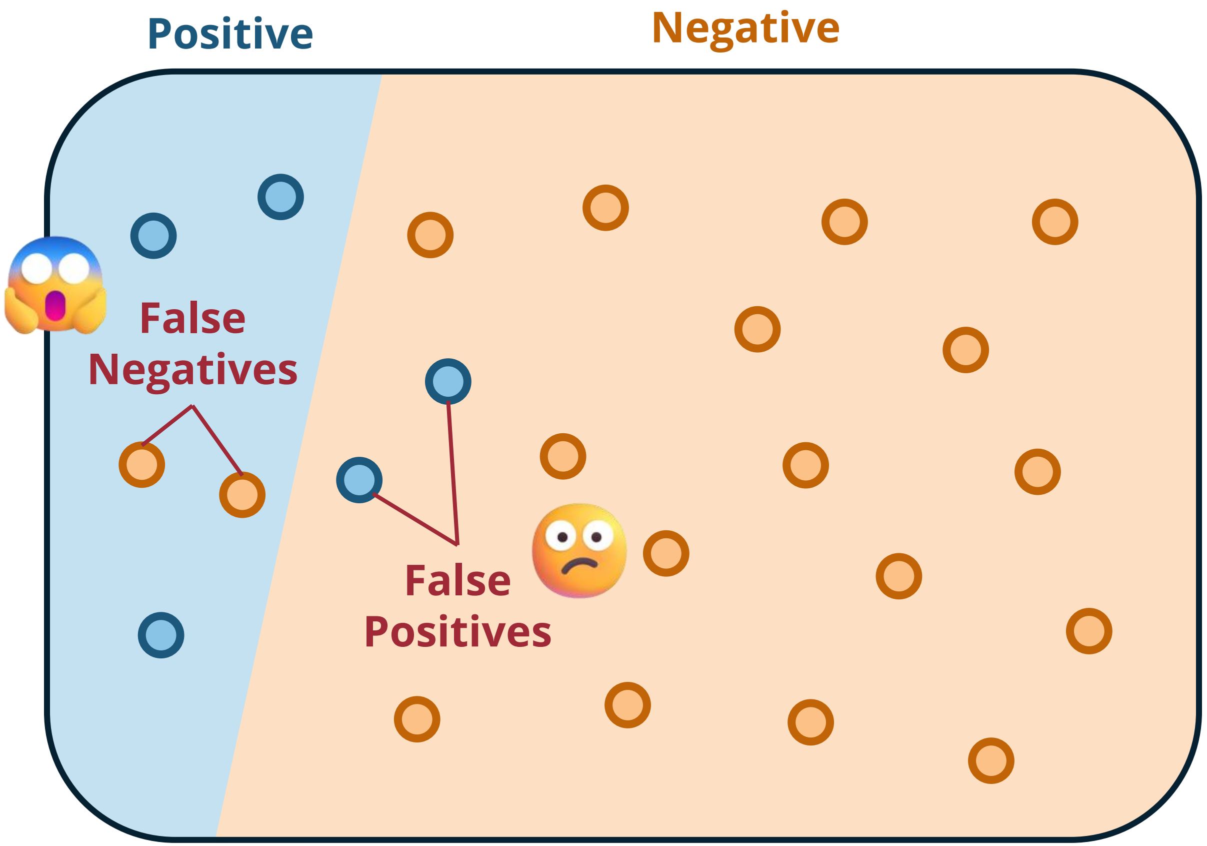
How well the model finds all positive instances in the dataset

When should we care about Precision & Recall?

Rare cancer detection

Aim for high precision or high recall?

High recall ensures most cancer cases are identified.



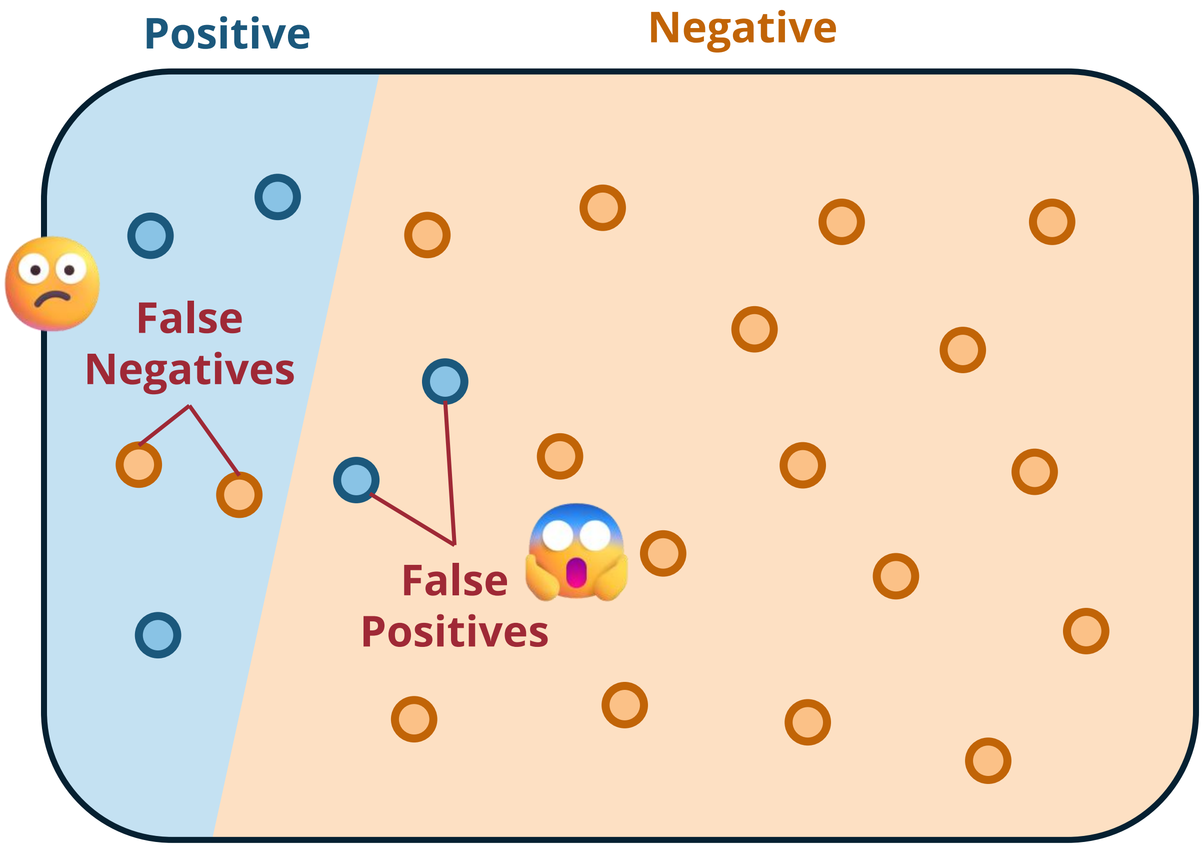
False alarms vs Missed detections

When should we care about Precision & Recall?

Music recommendation

Aim for high precision or high recall?

High precision ensures that the model won't recommend irrelevant items.



False alarms vs Missed recommendations

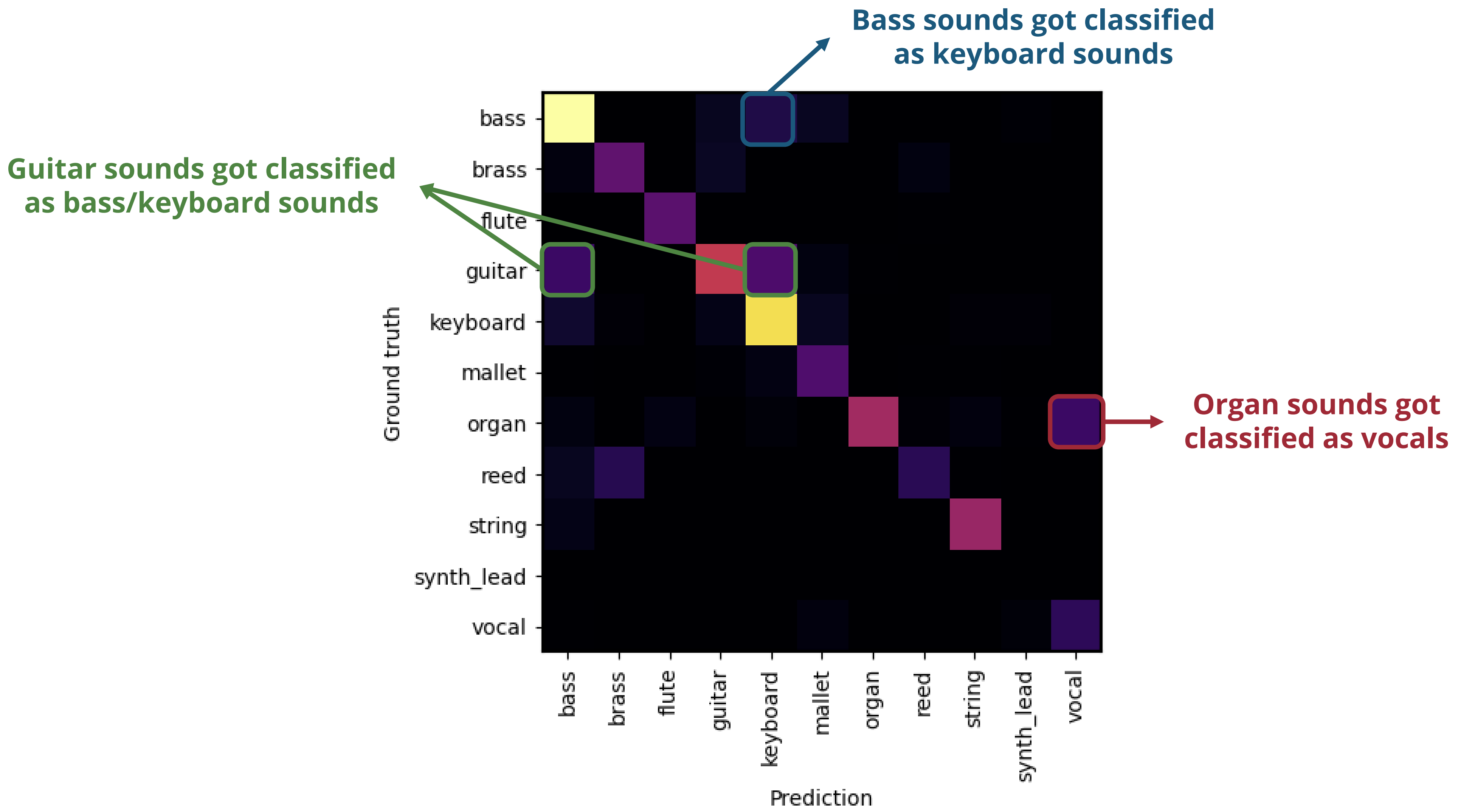
F1 Score: Considering both Precision & Recall

- Particularly useful for **imbalanced datasets**
  - Work better than accuracy when the dataset is imbalanced
  - For example, **music search**, **retrieval**, and **recommendation**

$$F_1 = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

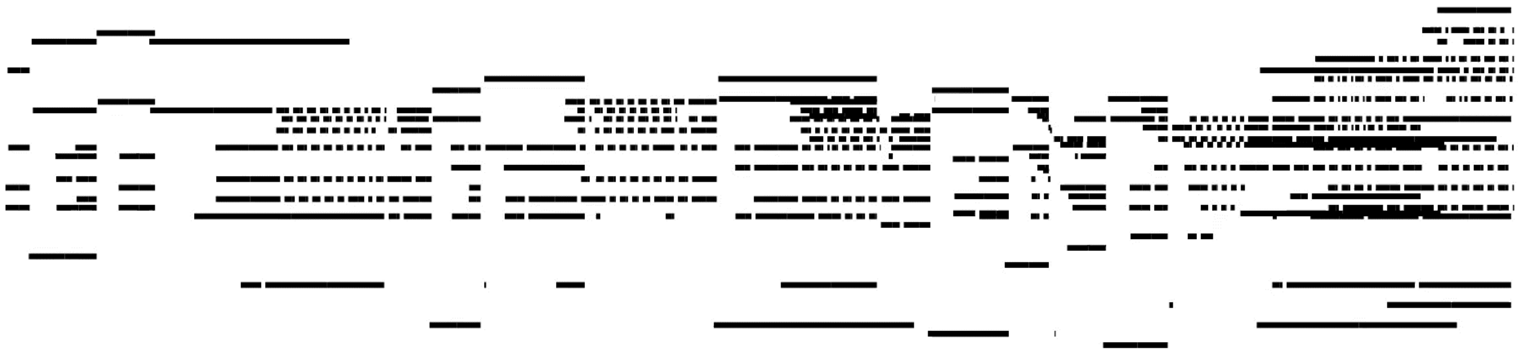
$$= \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

Confusion Matrix for Multiclass Classification



Next Lecture

Language-based Music Generation



(Source: Huang et al., 2018)