

Feature Extraction & Engineering

8

B

CS 3244
Machine Learning



NUS | Computing

What did we learn this week?

Data Issues

- 1. Linear Separability
- 2. Curse of Dimensionality
- 3. Imbalanced Data

Issue Template

- 1. **What** is the issue?
- 2. **Why** is it a problem?
- 3. **When** would it happen?
- 4. **How** to check for it?
- 5. **How** to mitigate it?

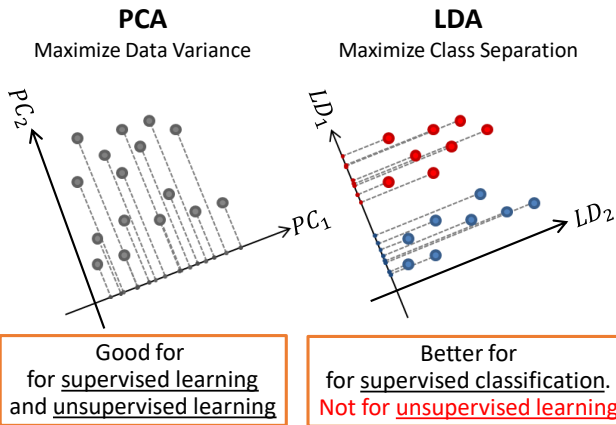
Checks

- 1. Linear SVM, PCA, LDA
- 2. Visualize Histograms

Mitigations

- 1. Dimensionality Reduction (PCA, LDA, Deep Auto-Encoders)
- 2. Feature Selection (Recursive Feature Elimination, Correlation, Mutual Information)
- 3. Resampling (Under/Oversampling, SMOTE)

PCA and LDA for Dimensionality Reduction



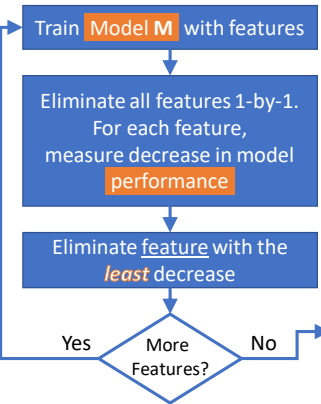
Steps

- All axes are orthogonal (independent)
- 1. **Identify** basis vectors
- 2. **Rank** basis vectors by importance
- 3. **Truncate** selection of basis vectors
 - Keeps more important features
 - Performs dimensionality reduction

Issue: Curse of Dimensionality

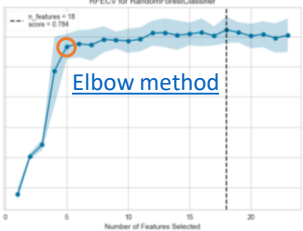
5. How to mitigate it?

- Feature Selection
 - Wrapper methods (e.g., RFE)
 - Filter methods



Recursive Feature Elimination (RFE)

Image Credit: https://www.scikit-yb.org/en/latest/api/model_selection/rfecv.html



Synthetic Minority Oversampling Technique (SMOTE)

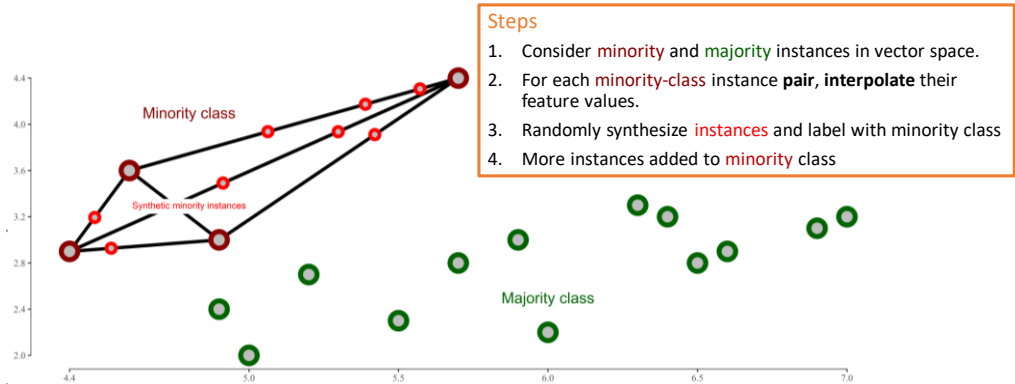


Image Credit: <https://www.quora.com/Can-you-explain-me-SMOTE-Synthetic-Minority-Over-sampling-Technique-in-simple-terms>

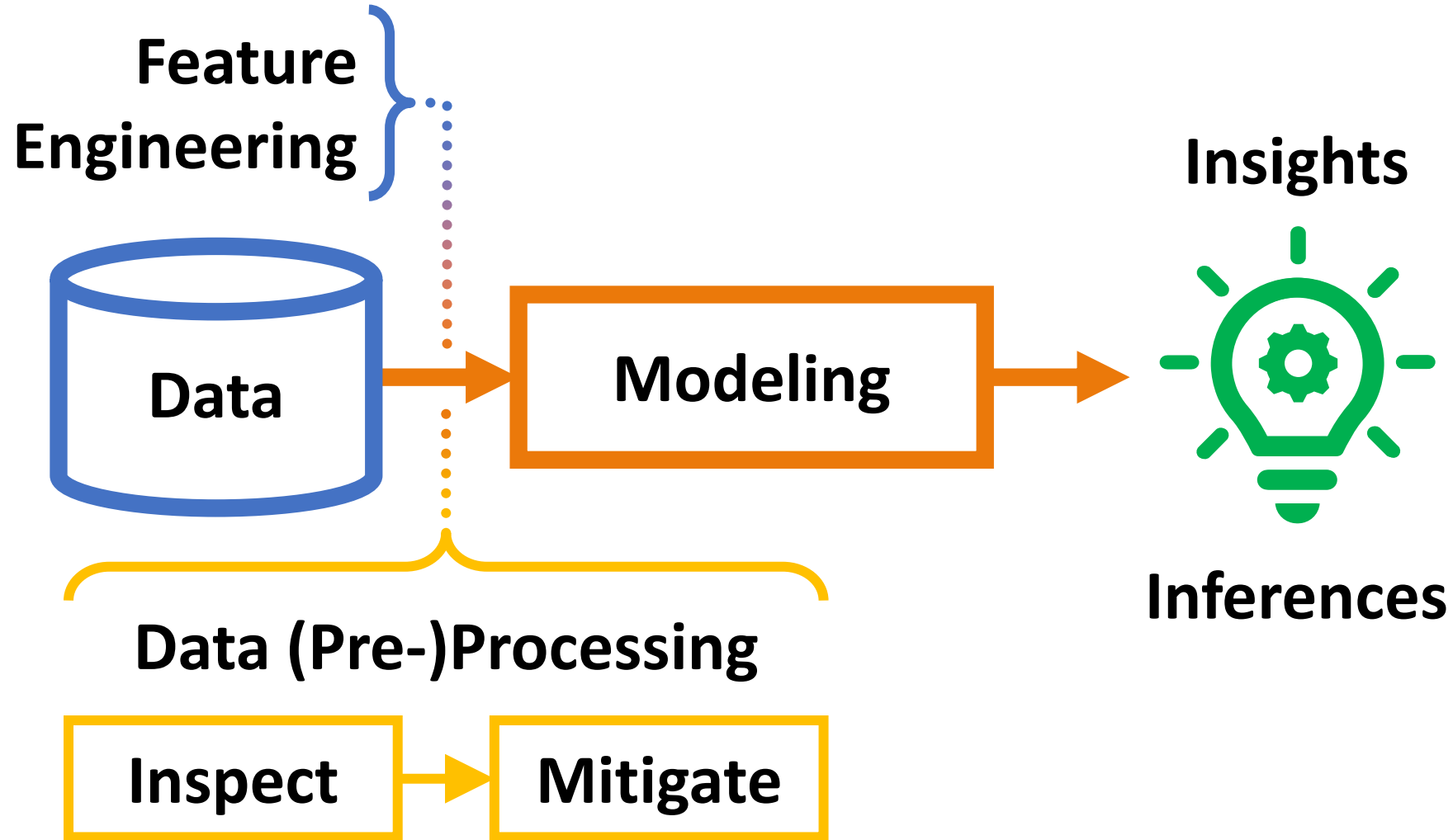
Week 08B: Learning Outcomes

1. Describe **techniques** of feature extraction/engineering for different data types
 - Tabular
 - Temporal (Time Series)
 - Image
 - Text
2. Describe **issues** when extracting features for various data types

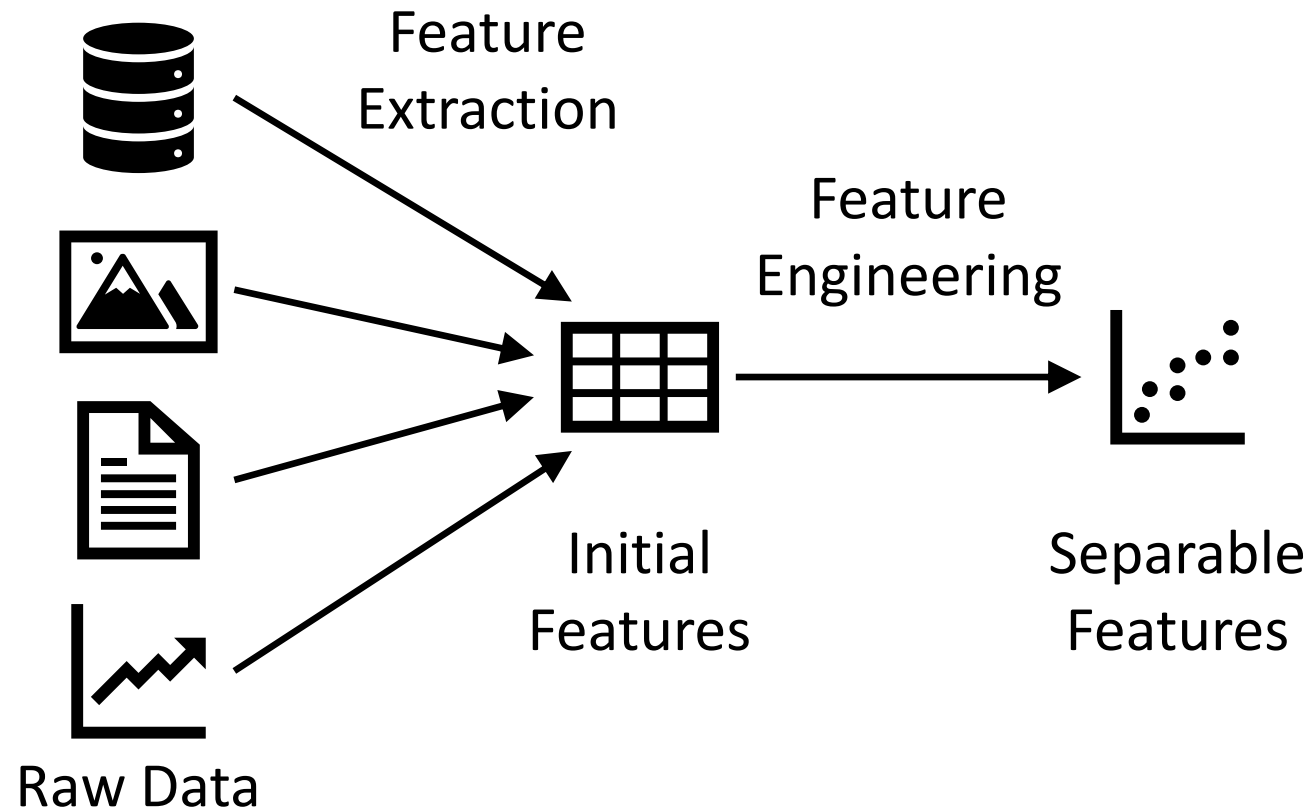
Week 08B: Lecture Outline

1. Overview: Feature Extraction and Engineering
2. Data Features
 1. Tabular Features
 2. Temporal (Time Series) Features
 3. Image Features
 4. Text Features

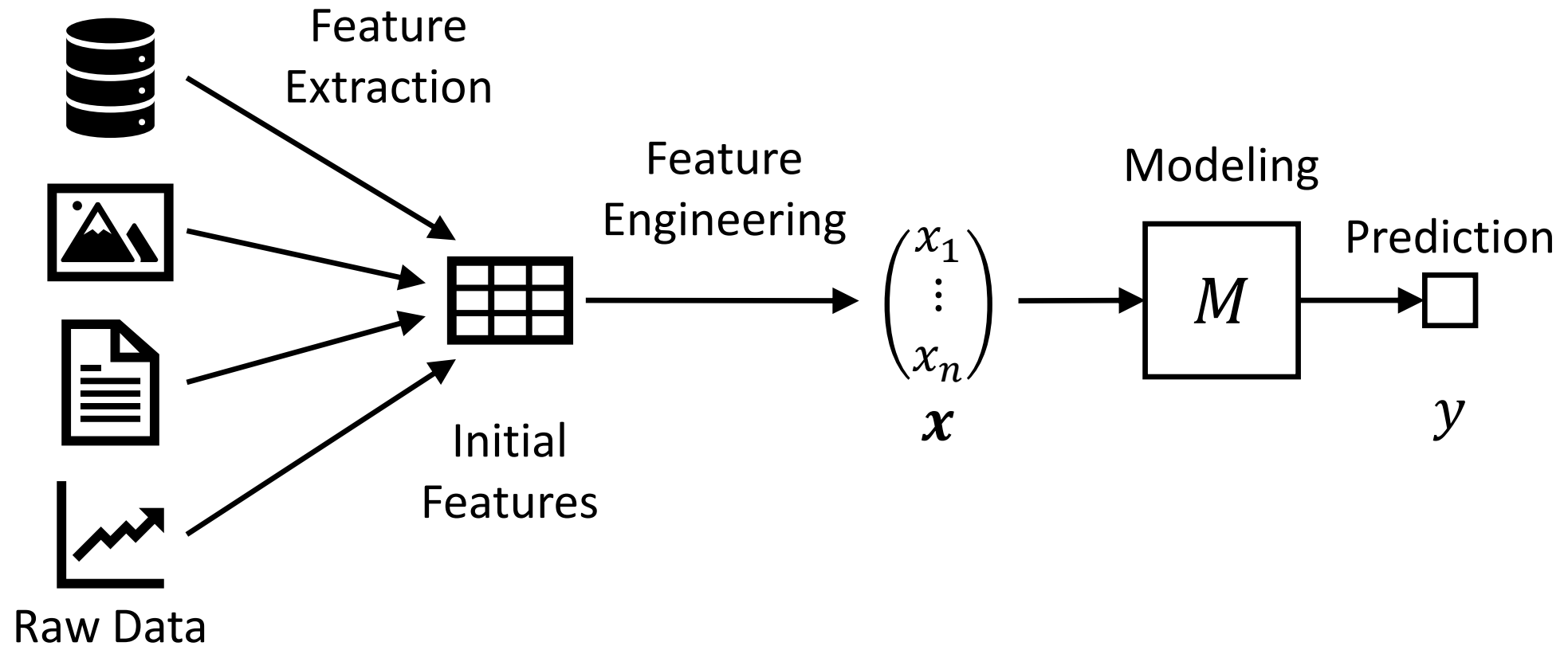
Machine Learning Pipeline



Feature Extraction and Engineering



Feature Extraction/Engineering \rightarrow Modeling



Feature Extraction and Engineering

- Process of ***transforming* raw data** to improve the accuracy of models
- Enables you to
 - **Capture domain knowledge** (e.g., periodicity or relationships between features)
 - **Express non-linear relationships** using linear models
 - **Encode non-numeric features** to be used as inputs to models

Adapted from: [Joseph Gonzalez, John DeNero, Josh Hug](#)



Tabular Features

Subscriptions → \$\$

Stopped Subscription = Churn

KKBOX

Every Voice Inspires!

Access over 70 million
songs ad-free

Listen to podcasts free,
anytime, anywhere

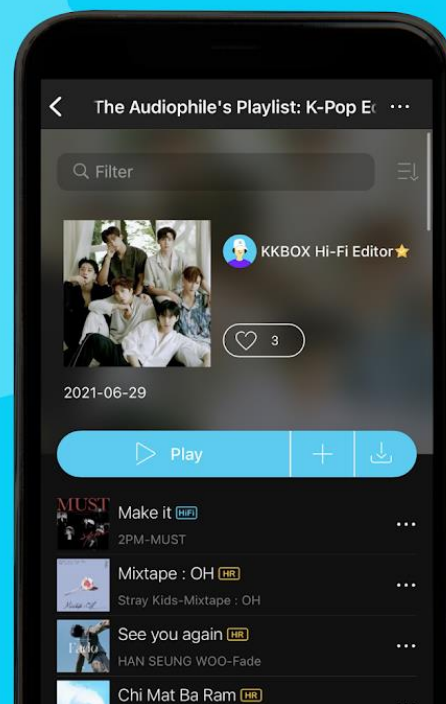
Subscribe and enjoy a
14-day Music Pass on us

Lossless Audio



Supports up to 24bit / 192kHz

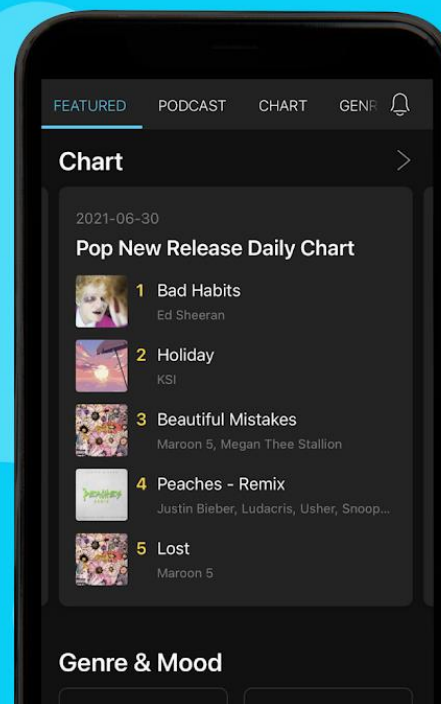
HR HiFi



Music Charts



Listen to what's trending daily



Listen With



Listen and interact with your
favourite artists

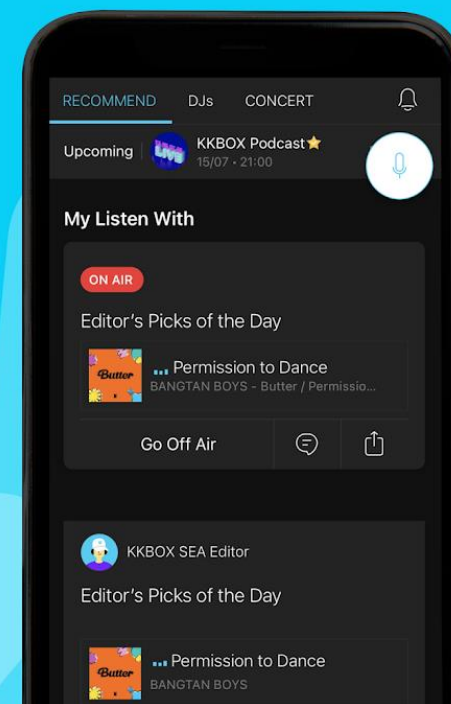


Image credit: https://play.google.com/store/apps/details?id=com.skysoft.kkbox.android&hl=en_NZ&gl=SG

Tabular Feature Extraction & Engineering

Prediction Task: Customer Churn Prediction

Database	Observations	# of Unique Users Covered	Dates	Data
members	6.7 mn KKBox users	6.7 mn users	Up to March 2017	<ul style="list-style-type: none">User's ageGenderCityInitial registration dateRegistration method (channel)
transactions	23 mn transactions	2.4 mn users (paying subscribers)	Jan. 2015 – March 2017	<ul style="list-style-type: none">Transaction dateTransaction amountSubscription duration of that txnPayment methodOn auto renew (yes/no)Cancelled subscriptions
user_logs	155 mn daily listening and user activity logs	2.5 mn users	Jan. 2015 – March 2017	<ul style="list-style-type: none">Date of activityTotal listening time that day (secs.)# of unique songs started that day% of song listened to: # of songs the user played for: 0-25% through, 25-50%, 50-75%, 75-98.5%, 98.5-100%
train	1 mn user churn results (yes/no)	1 mn users	Feb. 2017	<ul style="list-style-type: none">User's churn result for Feb. 2017 (yes/no)
test	1 mn user churn results (yes/no)	1 mn users	March 2017	<ul style="list-style-type: none">User's churn result for March 2017 (yes/no)

How to use?

- Data in multiple **separate tables**
- Need to **join** tables (e.g., with [SQL](#), [pandas](#))

How to use?

Labels

Source: <https://medium.com/@chrishuskey/using-machine-learning-to-improve-subscriber-retention-at-kkbox-taiwans-spotify-before-a8c7702f8bd3>

Tabular Feature Engineering:

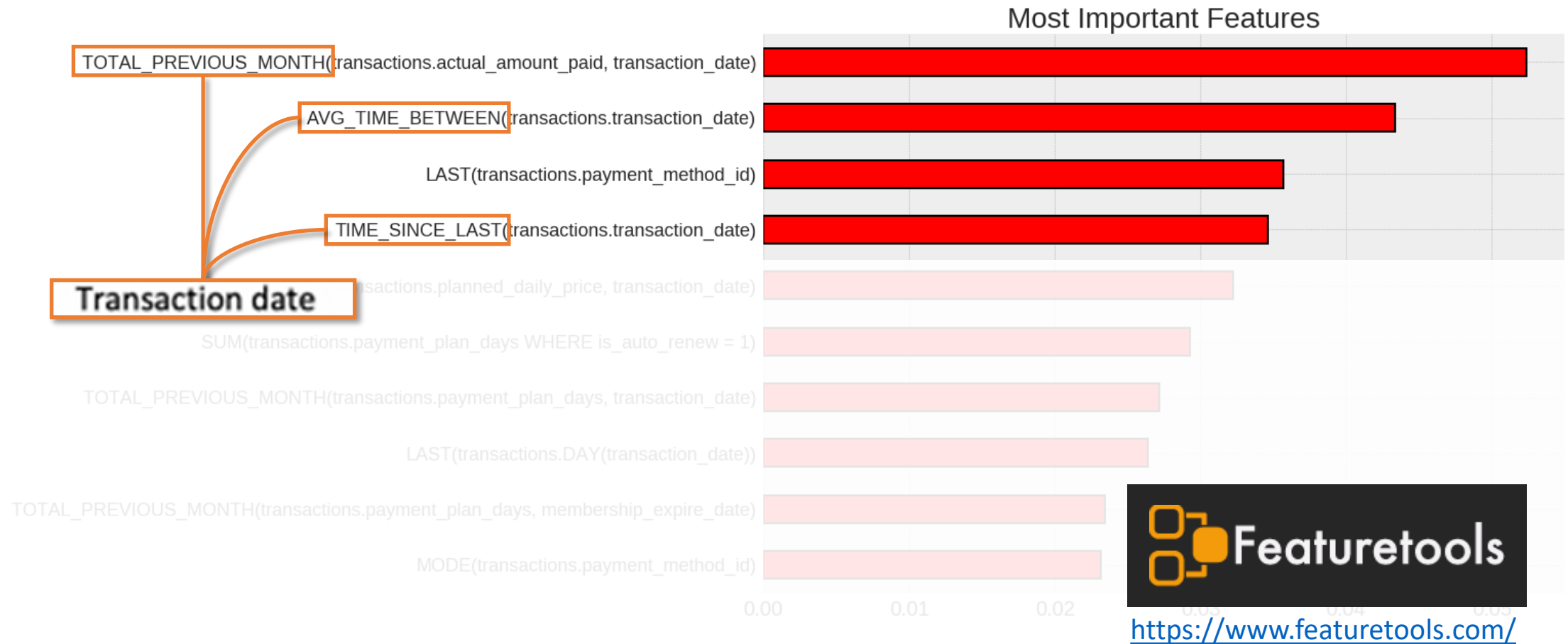
Custom Equations based on Domain Knowledge

% of song listened to: # of songs the user played for: 0-25% through, 25-50%, 50-75%, 75-98.5%, 98.5-100%

$$\text{Song affinity score} = \frac{\text{\# of songs listened } \geq 98.5\% \text{ through (last 30 days)} - \text{\# of songs listened } \leq 50\% \text{ through (last 30 days)}}{\text{total \# of songs started (last 30 days)}}$$

Source: <https://medium.com/@chrishuskey/using-machine-learning-to-improve-subscriber-retention-at-kkbox-taiwans-spotify-before-a8c7702f8bd3>

Tabular Feature Engineering: Counting, Aggregation, Difference, Min, Max



Source: <https://github.com/Featuretools/predict-customer-churn>

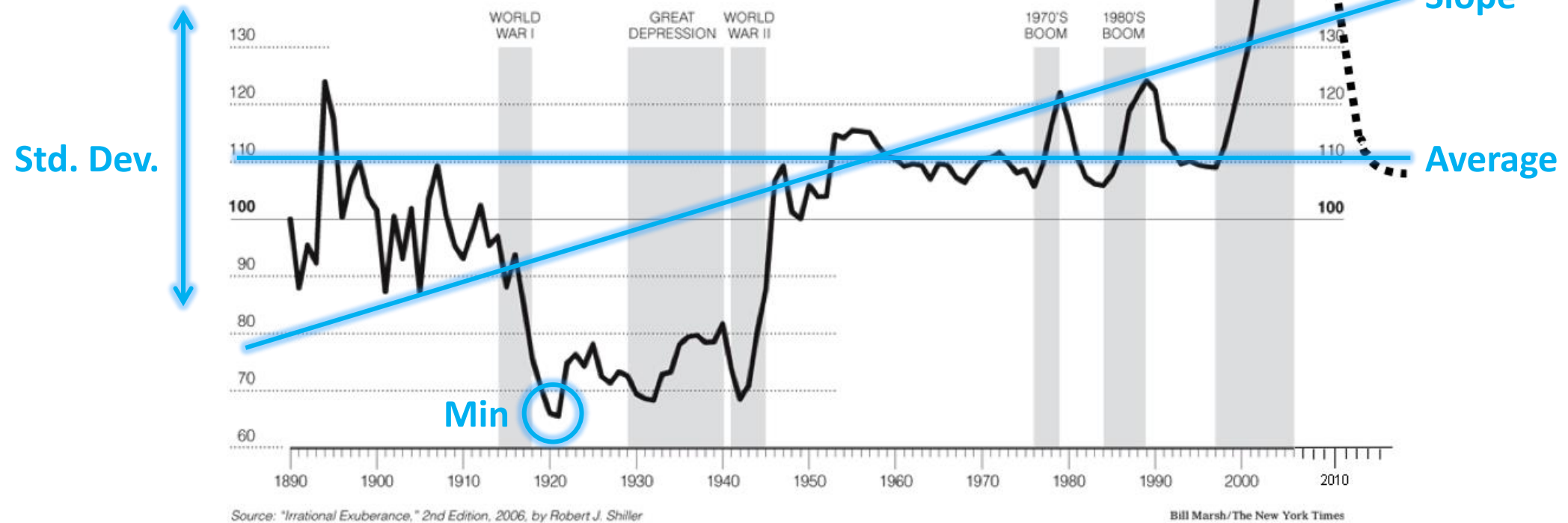


Temporal (Time) Features

A History of Home Values

Image credit:

<https://ritholtz.com/2008/12/classic-case-shiller-housing-price-chart-updated/>

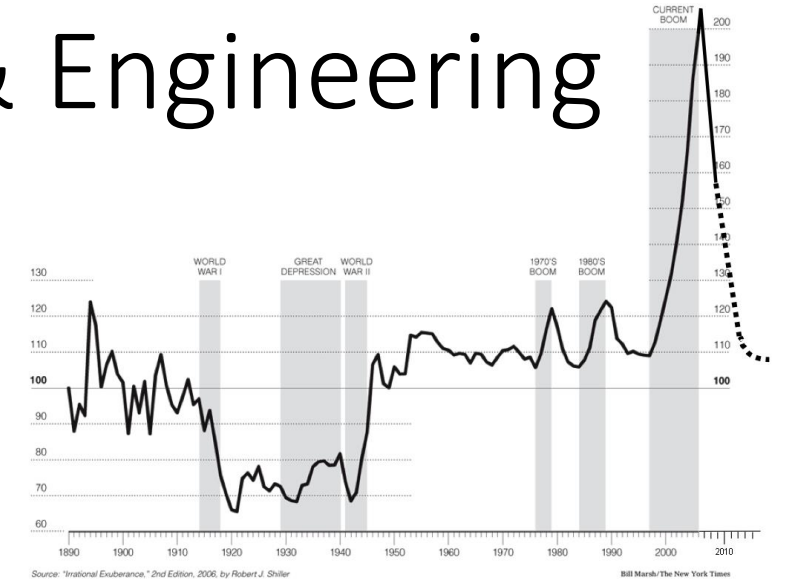


Temporal Feature Extraction & Engineering

- Prediction Task: **Home Price Prediction**
- Features

- Previous value

- Average
- Variation: Standard Deviation
- Range: Min, Max
- Trend: Slope of linear fit



Aggregate Statistics
(encode history)

Linear regression

Total?

Small Time Window?

Sliding Time Window

- Prediction Task: **Price Prediction**
- Features
 - Moving Average
 - Moving Standard Deviation
 - Moving Range (Min, Max)
 - Moving Trend (Slope of linear fit)

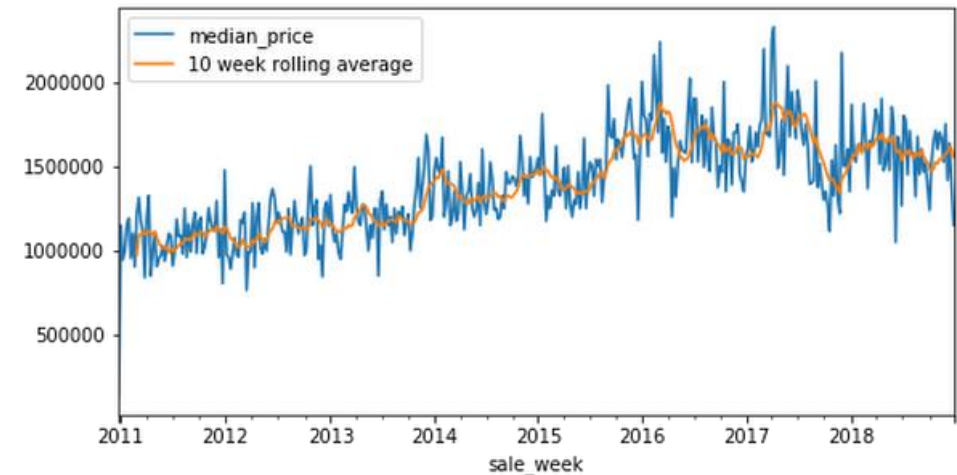
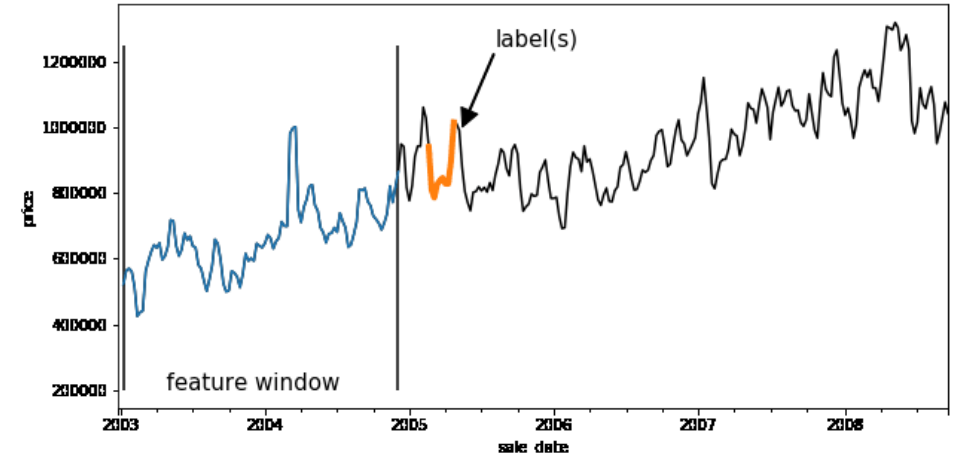
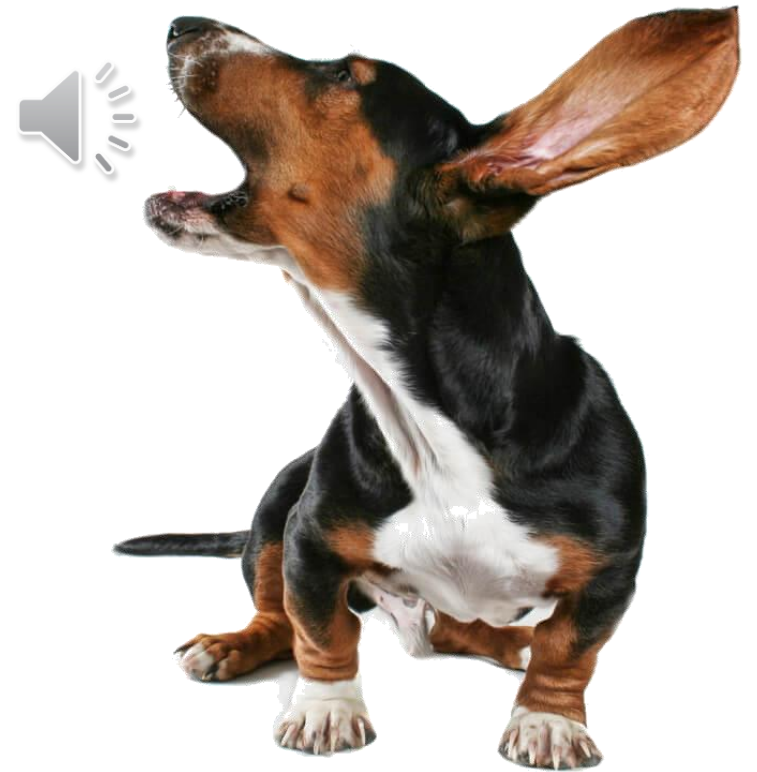
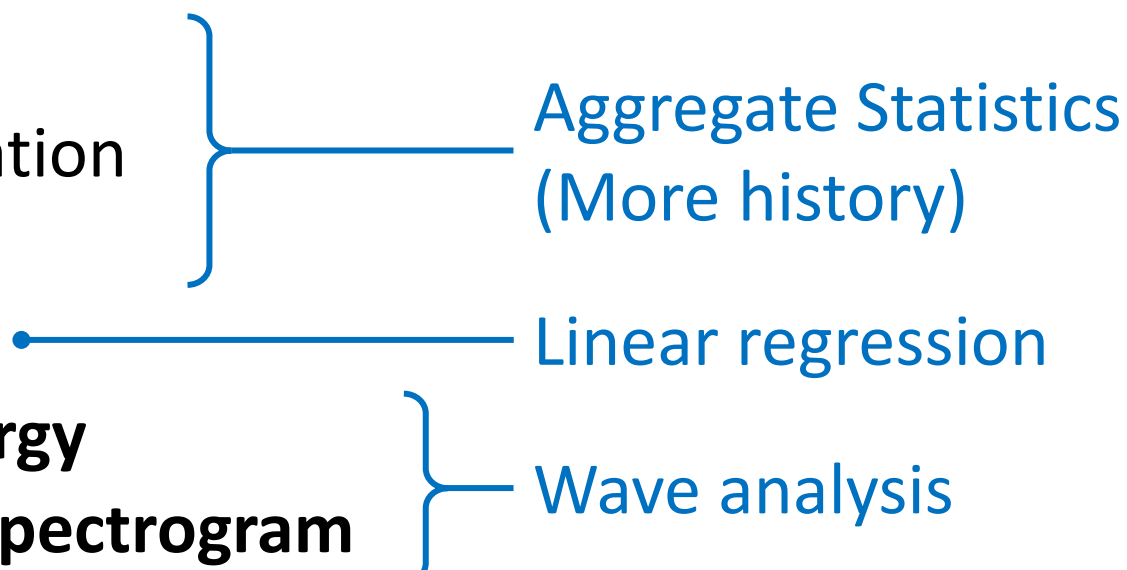


Image credit: <https://cloud.google.com/blog/products/ai-machine-learning/how-to-quickly-solve-machine-learning-forecasting-problems-using-pandas-and-bigquery>

Sound Classification

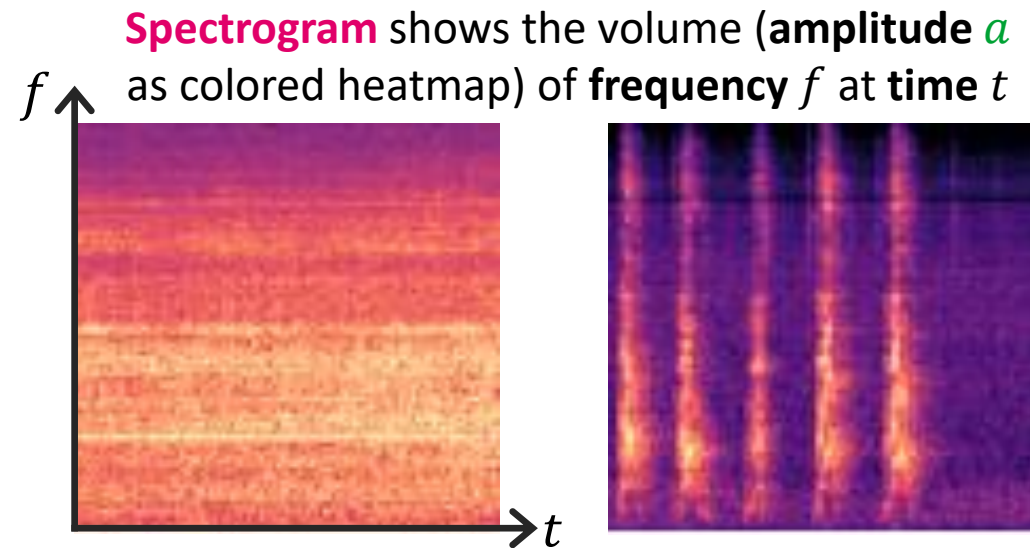
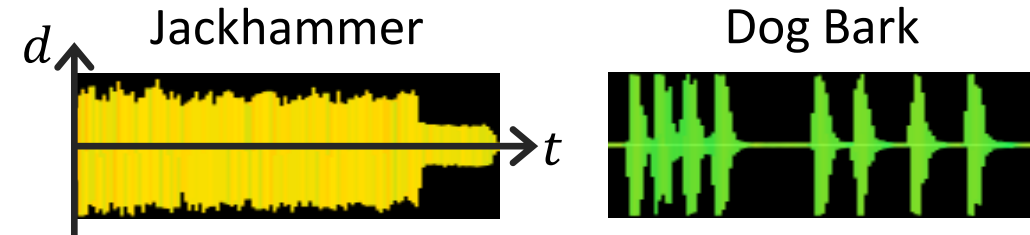
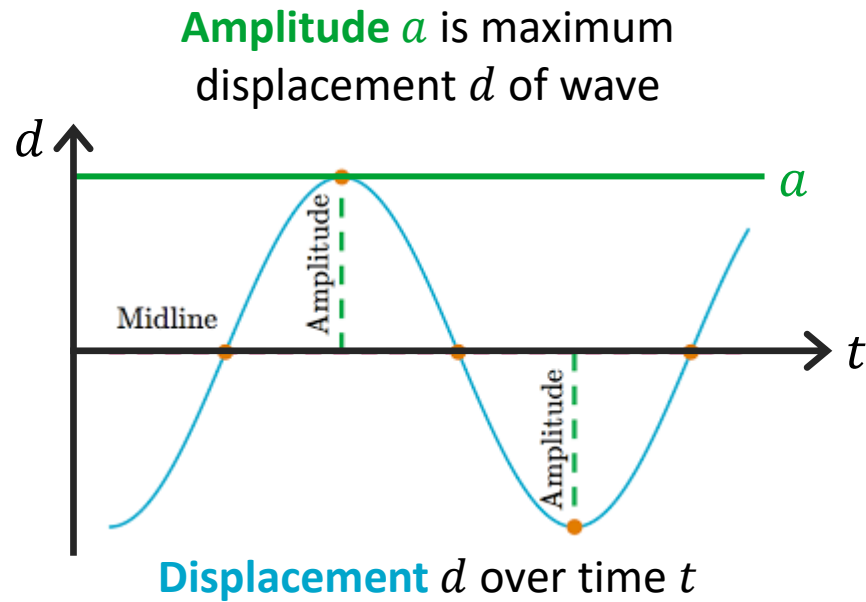


Temporal Feature Extraction & Engineering

- Prediction Task: **Sound Classification** (e.g., jackhammer, dog bark)
 - Features
 - Previous value
 - Average
 - Variation: Standard Deviation
 - Range: Min, Max
 - Trend: Slope of linear fit
 - **Volume: Amplitude, Energy**
 - **Frequency: Periodicity, Spectrogram**
- 
- The diagram uses blue lines and brackets to group features into three categories:
- Aggregate Statistics (More history)**: A vertical bracket on the right groups the features "Average", "Variation: Standard Deviation", and "Range: Min, Max".
 - Linear regression**: A horizontal line with a dot at its left end connects the feature "Trend: Slope of linear fit" to the label.
 - Wave analysis**: A vertical bracket on the right groups the features "**Volume: Amplitude, Energy**" and "**Frequency: Periodicity, Spectrogram**".

Audio Domain-Specific Features: Amplitude, Spectrogram

Not in exam



Notice: spectrograms are **images**
Can extract image features!

Further reading: <https://www.kdnuggets.com/2020/02/audio-data-analysis-deep-learning-python-part-1.html>
Programming library: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.spectrogram.html>
Spectrogram source: <https://etown.medium.com/great-results-on-audio-classification-with-fastai-library-ccaf906c5f52>



Questions!

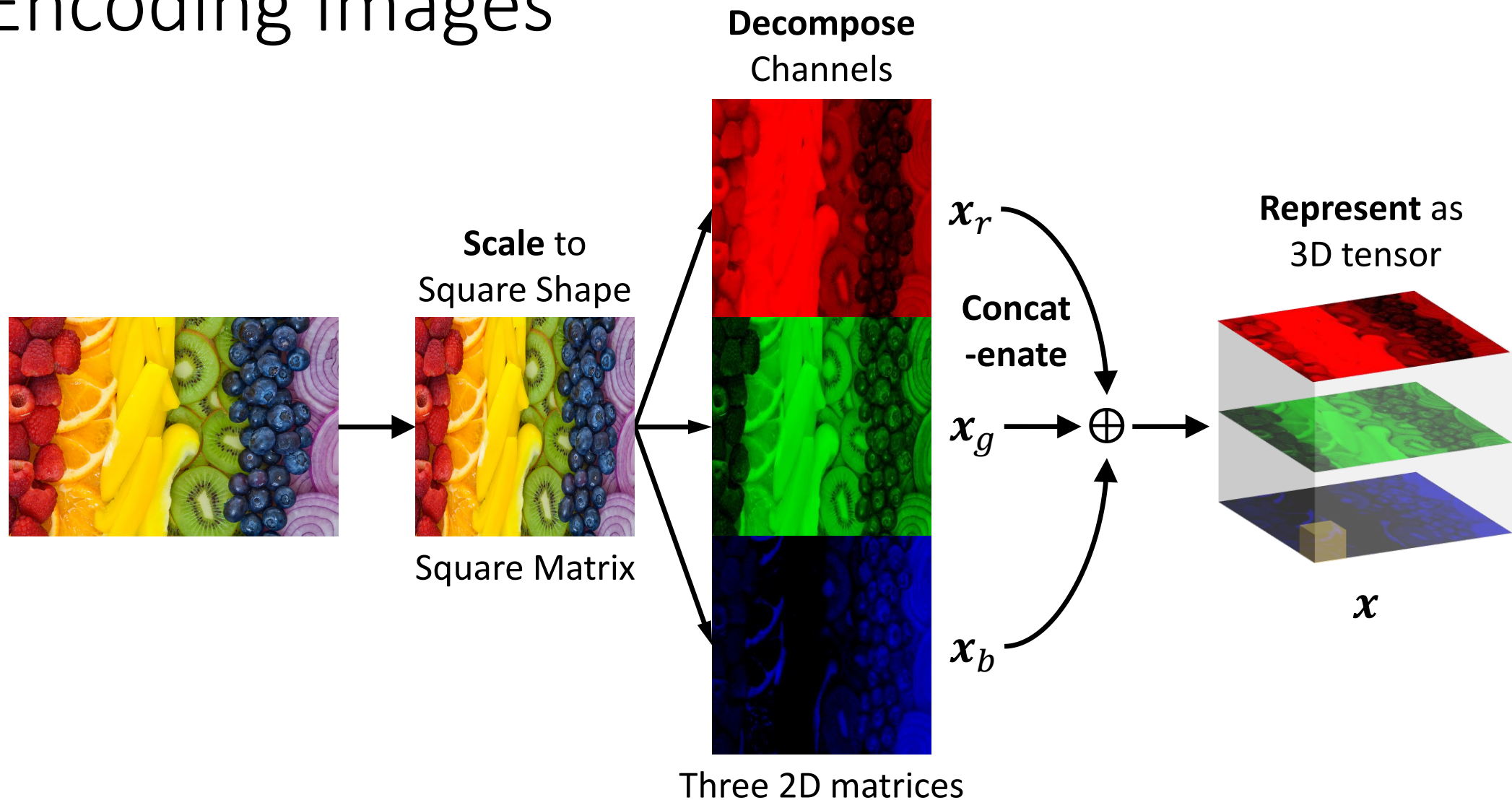




Image Features



Encoding Images



What **features** would you use to...
Classify the sports ball in the image?

In Slack [#lecture](#)

1. Respond to thread (write) to suggest feature
2. Emote (👍 :+1:) to vote for feature



Basketball



American Football



Bowling



Tennis

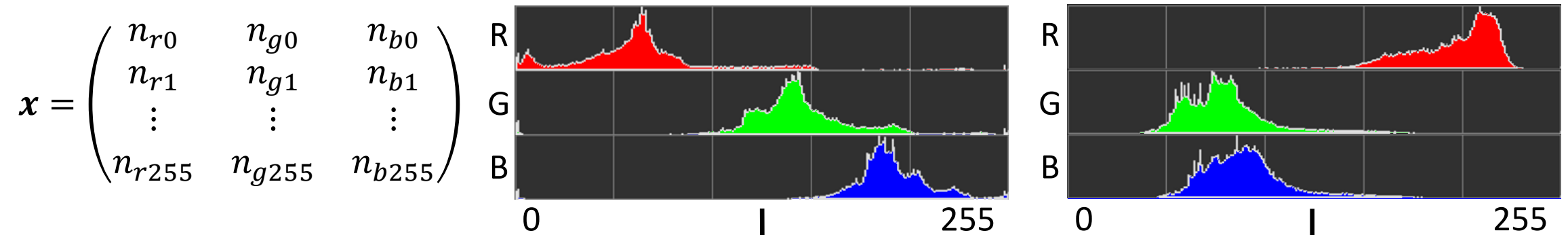


Golf

Image Feature Extraction & Engineering

- Prediction Task: **Sports ball classification**
- Features
 - Size? Photos may have **different** zoom levels
 - Color? Color Histogram → Vector
 - Shape? Edge detection → PCA
 - Texture? Convolutional filters

Feature: Color Distribution (Histogram)



Basketball



American
Football



Bowling



Tennis



Golf

Feature: Color Distribution

Color may **not** be good, since balls can have the **same colors** even for different sports



Basketball



American
Football



Bowling

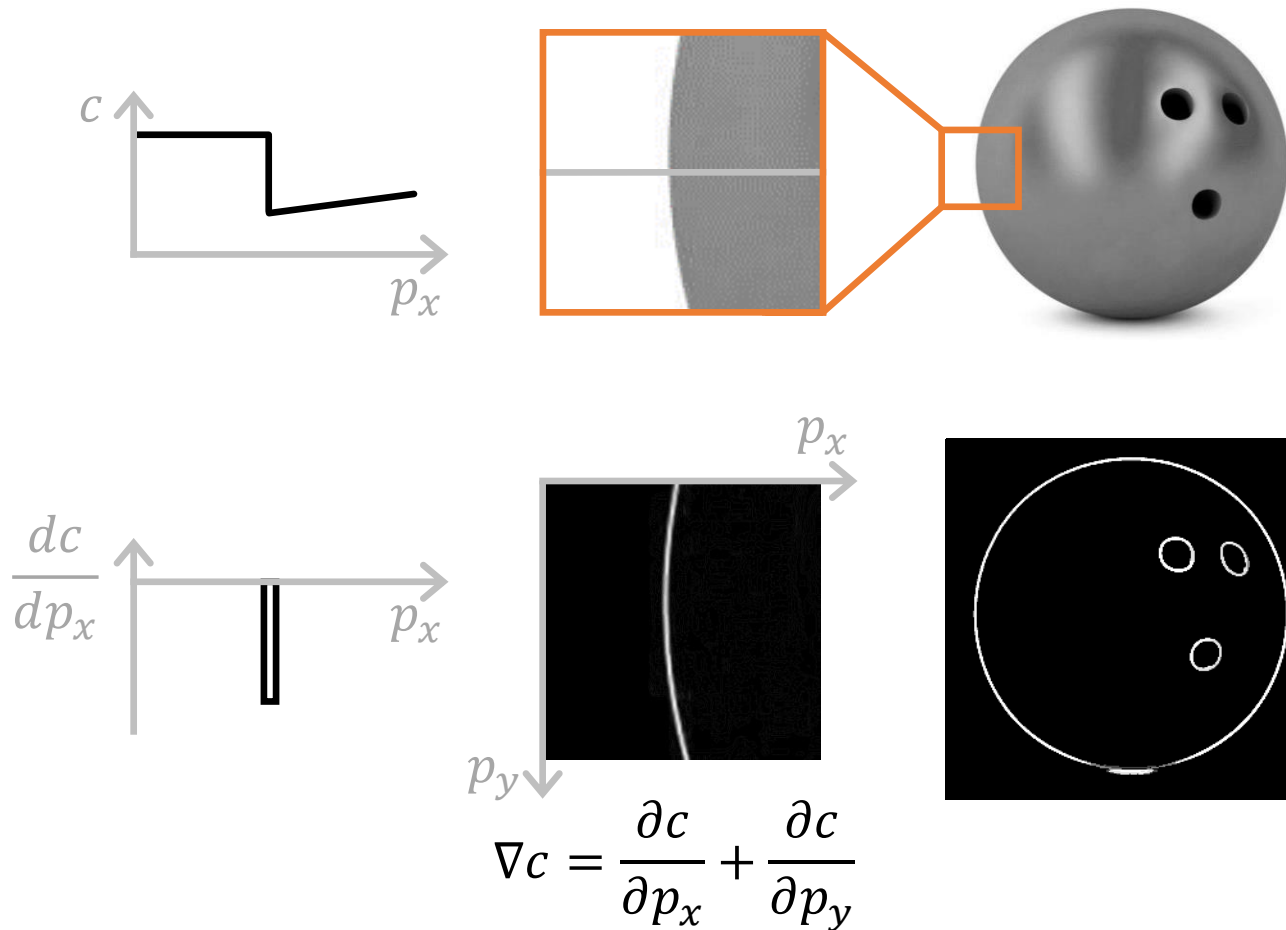


Tennis



Golf

Feature: Edge Detection for Shape Features



1. Convert to grayscale
2. Measure color c at each pixel position p_x
3. Compute 1st derivative $\frac{dc}{dp_x}$
 - High magnitude indicates edge
4. In 2D, calculate gradient ∇c
5. Threshold: $[\nabla c > c_{th}]$

Feature: Edge Detection Kernels

$$\frac{\partial c}{\partial p_x} \approx \frac{c_{(x+1,y)} - c_{(x-1,y)}}{p_{(x+1,y)} - p_{(x-1,y)}}$$

$$I_{p_x} = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} c_{(-1,-1)} & c_{(0,-1)} & c_{(1,-1)} \\ c_{(-1,0)} & c_{(0,0)} & c_{(1,0)} \\ c_{(-1,1)} & c_{(0,1)} & c_{(1,1)} \end{pmatrix}$$

I_{p_x} is a Convolution Matrix (**Kernel**)

* is **Convolution** operator

Means: element-wise multiply, then sum

$$I_{p_x} * \mathbf{x} = \begin{pmatrix} -c_{(-1,-1)} + 0 + c_{(1,-1)} \\ -c_{(-1,0)} + 0 + c_{(1,0)} \\ -c_{(-1,1)} + 0 + c_{(1,1)} \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} 9 & 9 & 3 & 3 & 4 \\ 9 & 3 & 3 & 4 & 5 \\ 9 & 3 & 3 & 5 & 5 \\ 9 & 3 & 3 & 4 & 5 \\ 9 & 9 & 3 & 3 & 4 \end{pmatrix}$$

$$I_{p_x} * \mathbf{x} = \begin{pmatrix} -6 - 6 - 6 & -6 + 1 + 2 & 1 + 2 + 2 \\ -6 - 6 - 6 & 1 + 2 + 1 & 2 + 2 + 2 \\ -6 - 6 - 6 & 2 + 1 - 6 & 2 + 2 + 1 \end{pmatrix}$$

Feature: Edge Detection Kernels

$$\frac{\partial c}{\partial p_x} \approx \frac{c_{(x+1,y)} - c_{(x-1,y)}}{p_{(x+1,y)} - p_{(x-1,y)}}$$

$$I_{p_x} = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} c_{(-1,-1)} & c_{(0,-1)} & c_{(1,-1)} \\ c_{(-1,0)} & c_{(0,0)} & c_{(1,0)} \\ c_{(-1,1)} & c_{(0,1)} & c_{(1,1)} \end{pmatrix}$$

I_{p_x} is a Convolution Matrix (**Kernel**)

Matrix convolutions
can do differentiation
in **parallel**
=> fast on GPU

$$\mathbf{x} = \begin{pmatrix} 9 & 9 & 3 & 3 & 4 \\ 9 & & & & 5 \\ 9 & & & & 5 \\ 9 & & & & 5 \\ 9 & 9 & 3 & 3 & 4 \end{pmatrix}$$

Convolutional Neural Network (CNN)
use kernels too! *More in W10*

* is **Convolution** operator

Means: element-wise multiply, then sum

$$I_{p_x} * \mathbf{x} = \begin{pmatrix} -c_{(-1,-1)} + 0 + c_{(1,-1)} \\ -c_{(-1,0)} + 0 + c_{(1,0)} \\ -c_{(-1,1)} + 0 + c_{(1,1)} \end{pmatrix}$$

$$I_{p_x} * \mathbf{x} = \begin{pmatrix} -6 - 6 - 6 & -6 + 1 + 2 & 1 + 2 + 2 \\ -6 - 6 - 6 & 1 + 2 + 1 & 2 + 2 + 2 \\ -6 - 6 - 6 & 2 + 1 - 6 & 2 + 2 + 1 \end{pmatrix}$$

$$= \begin{pmatrix} -18 & -3 & 5 \\ -18 & 4 & 6 \\ -18 & -3 & 5 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Feature: Edge Detection Kernels (2D)

$$\begin{aligned} \frac{\partial c}{\partial p_x} &\approx \frac{c(x+1,y) - c(x-1,y)}{p(x+1,y) - p(x-1,y)} \\ &\downarrow \\ I_{p_x} &= \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \\ \frac{\partial c}{\partial p_y} &\approx \frac{c(x,y+1) - c(x,y-1)}{p(x,y+1) - p(x,y-1)} \\ &\downarrow \\ I_{p_y} &= \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \end{aligned}$$

$\nabla c = \frac{\partial c}{\partial p_x} + \frac{\partial c}{\partial p_y} \approx (I_{p_x} + I_{p_y}) * \mathbf{x}$

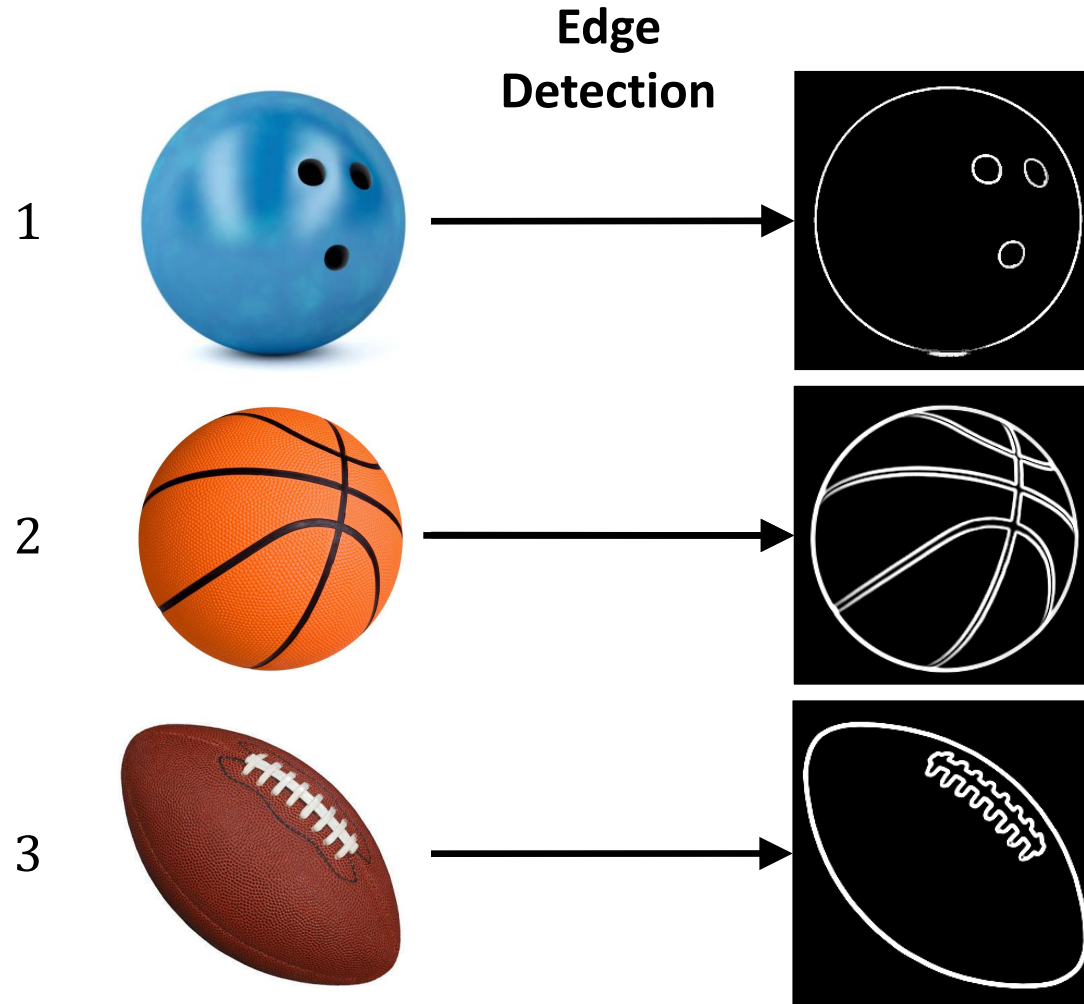
Feature: Edge Detection Kernels (2D)

$$\begin{aligned} \frac{\partial c}{\partial p_x} &\approx \frac{c(x+1,y) - c(x-1,y)}{p(x+1,y) - p(x-1,y)} \\ \frac{\partial c}{\partial p_y} &\approx \frac{c(x,y+1) - c(x,y-1)}{p(x,y+1) - p(x,y-1)} \end{aligned}$$

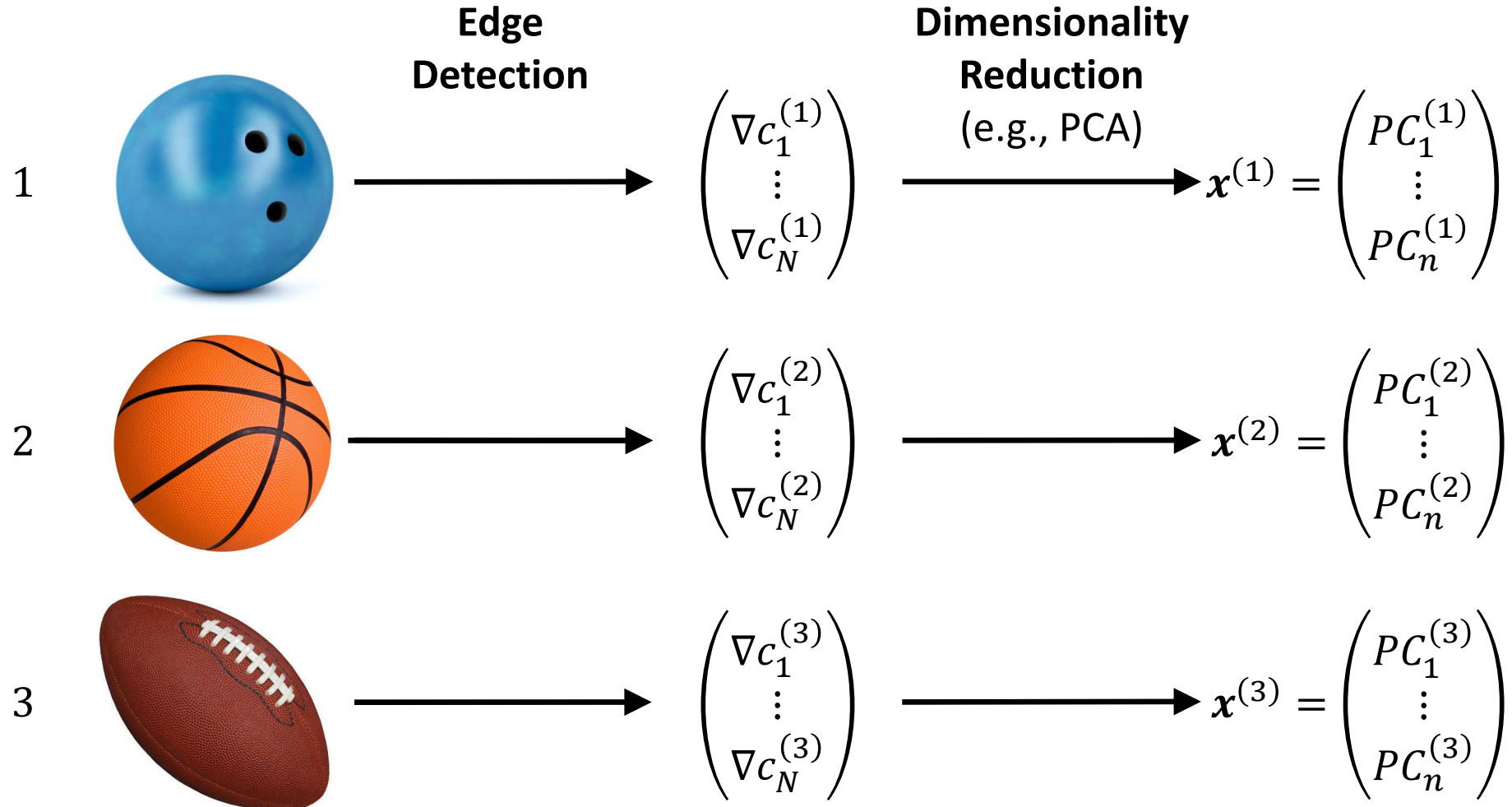
$$I_{p_x} = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad I_{p_y} = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\nabla c = \frac{\partial c}{\partial p_x} + \frac{\partial c}{\partial p_y} \approx \begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix} * \mathbf{x}$$

Feature: Shape Feature Vector

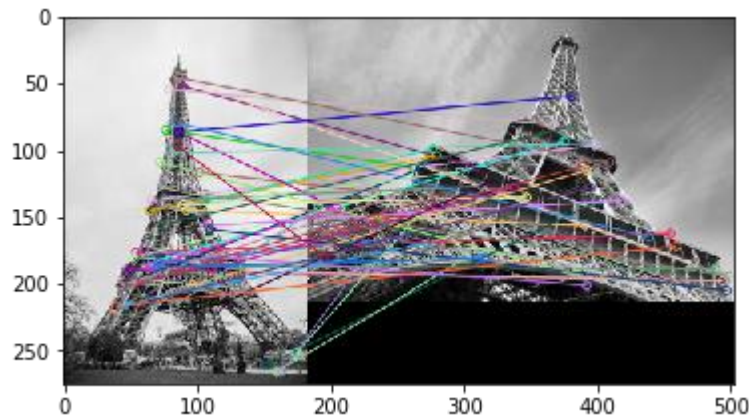


Feature: Shape Feature Vector



Advanced methods

- Scale-invariant feature transform (SIFT) – Not in exam
 - Find **keypoints** in image to match in other images



- Image \rightarrow SIFT feature vector \rightarrow Classify (e.g., kNN, logistic regression)
- Deep Learning (CNN) [W10]



Questions!



Text Features

What **features** would you use to...
Classify the sentiment of the restaurant review?

“Chicken wings were amazing honestly”

★★★ **Positive**

“Amazing wings, but waaaay to long to wait.”

★★☆ **Neutral**

“Not worth it! Too salty chicken and expensive!”

★☆☆ **Negative**

yelp Dataset

In Slack [#lecture](#)

1. Respond to thread (write) to suggest feature
2. Emote (👍 :+1:) to vote for feature

Text Feature Extraction & Engineering

- Prediction Task: **Review Sentiment Classification**
- Features
 - Text (String) Length?
 - Keywords? Which keywords?
 - Non-keywords? Stop words

Text Feature Extraction & Engineering



Problem / Objective	Approach
Extract words	Tokenization
Word variations	Stemming, Lemmatization
Uninformative words	Stop Word filtering
Identify informative words	Bag-of-Words (BOW)

Tokenization

- Split a single **string** of text into an **array** of substrings
- Split with **delimiters** (e.g., whitespaces ' ', newline '\n', punctuations ',.?')

Original Text	Tokenized into Array of Words
"Chicken wings were amazing honestly"	['chicken', 'wings', 'were', 'amazing', 'honestly']
"Amazing wings, but waaaay to long to wait."	['amazing', 'wings', 'but', 'waaaay', 'to', 'long', 'to', 'wait']
"Not worth it! Too salty chicken and expensive!"	['not', 'worth', 'it', 'too', 'salty', 'chicken', 'and', 'expensive']

Stemming and Lemmatization

 Stemming	 Lemmatization
Truncates words <i>without</i> contextual knowledge.	Groups <i>inflected</i> forms of a word as identified in a dictionary.
Base form of word is called stem	Base form of word is called lemma
Remove word endings: 'ed', 'ing', 'ly', 'ment', etc.	Looks up dictionary (e.g., WordNet) to find lemma, and replace word with lemma
Easier to implement	Harder to implement
Runs faster	Runs slower
E.g.: walking, walked → walk better → better having → hav	E.g.: walking, walked → walk better → good having → have

Further reading: <https://towardsdatascience.com/stemming-vs-lemmatization-2daddabcb221>

Programming library: <https://www.nltk.org/api/nltk.stem.html>

Stop Words

- Most **common** words for the language that have **little semantic meaning** (uninformative)
- Programming libraries maintain list of stop words (e.g., [NLTK](#))
- Problems? **May remove actually informative words** (e.g., 'not')

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now']

Original Text	After Stop Word filtering
"Chicken wings were amazing honestly"	"Chicken wings were amazing honestly"
"Amazing wings, but waaaay to long to wait."	"Amazing wings, but waaaay to long to wait."
"Not worth it! Too salty chicken and expensive!"	" Not worth it! Too salty chicken and expensive!"

Further reading: <https://medium.com/@saitejaponugoti/stop-words-in-nlp-5b248dadad47>



Bag of ~~Letters~~
Words

Image credit: <https://cdn-o.fishpond.co.nz/0218/542/604/1116441768/original.jpeg>

Bag-of-Words (BOW) Encoding

1. Preprocess string s to array of words \mathbf{w}
2. Array of words \rightarrow One-hot vector (fixed length)
3. $\text{BOW}(\mathbf{w}) \rightarrow \mathbf{x}$
4. Problem: **high dimensions** if many words

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_{13} \end{pmatrix}$$

$$\mathbf{x}^{(1)} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\mathbf{x}^{(2)} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

#	Original Text s	Pre-Processed Words \mathbf{w}	chicken	wings	amazing	honestly	but	way	too	long	wait	not	worth	salty	expensive
1	"Chicken wings were amazing honestly"	['chicken', 'wings', 'amazing', 'honestly']	1	1	1	1	0	0	0	0	0	0	0	0	0
2	"Amazing wings, but waaaay to long to wait."	['amazing', 'wings', 'but', 'way', 'too', 'long', 'wait']	0	1	1	0	1	1	1	1	1	0	0	0	0
3	"Not worth it! Too salty chicken and expensive!"	['not', 'worth', 'too', 'salty', 'chicken', 'expensive']	1	0	0	0	0	0	1	0	0	1	1	1	1

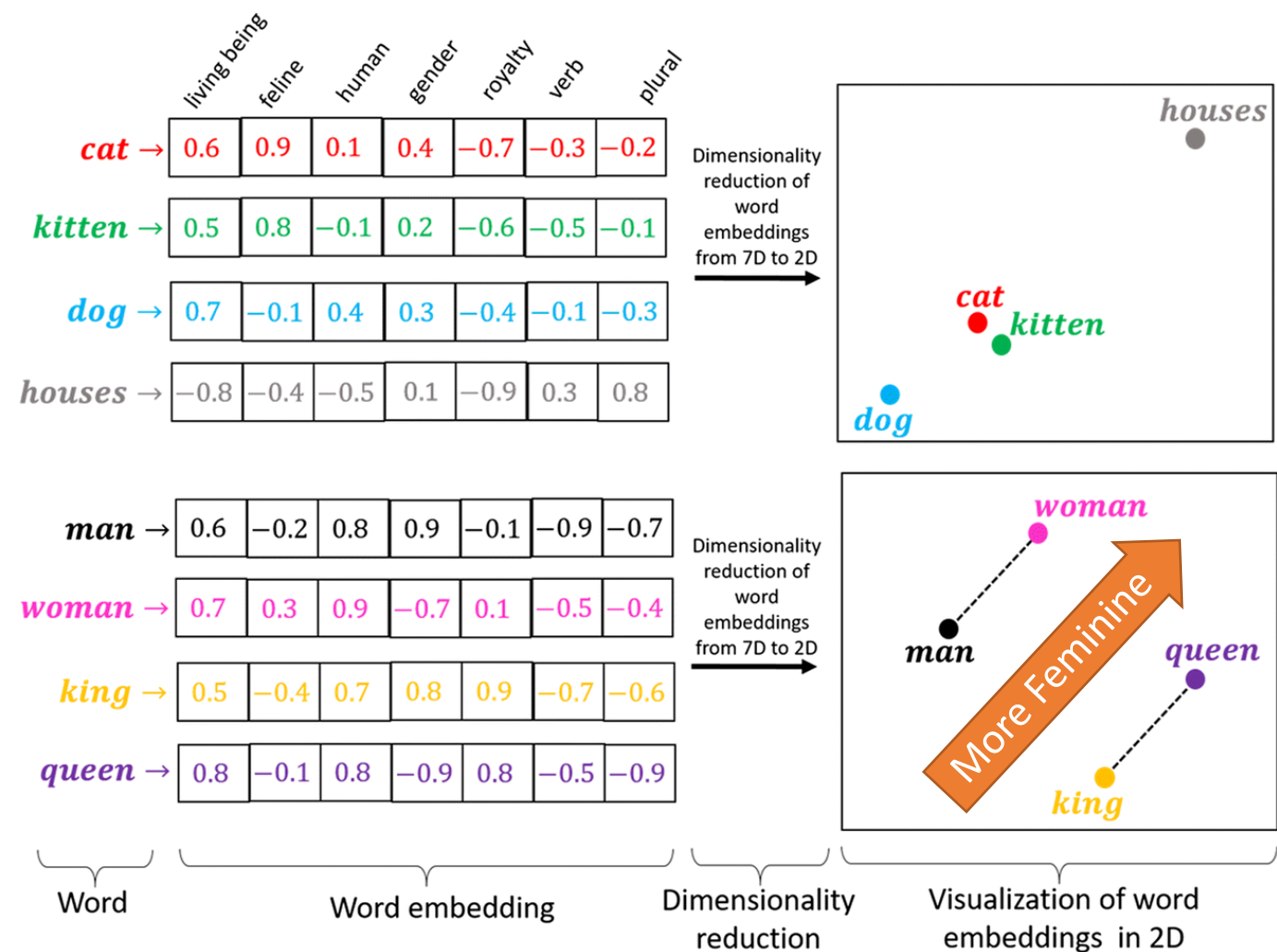
The word **"too"** could predict **negative** sentiment

Other methods: addressing weaknesses in BOW

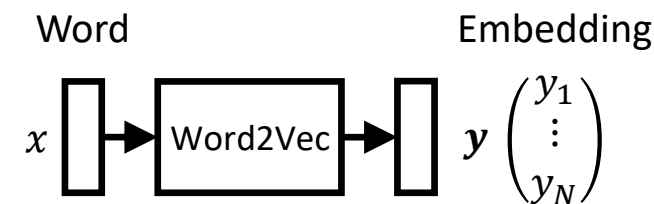
- Term frequency–inverse document frequency (TF-IDF)
 - Identifies if some words are generally **common** or **unique** to the instance
 - If **unique**, then word will be **more informative** to the class label
- N-grams
 - Adjacent words change meaning
 - e.g., “I am happy, not sad” vs. “I am sad, not happy”
 - But: BOW(“I am happy, not sad”) = BOW(“I am sad, not happy”)
 - With bigrams: BOW(“I am happy, not_sad”) \neq BOW(“I am sad, not_happy”)
- Parts of Speech (POS) and Grammar
 - Structure of the sentence matters

You need to **understand** w.r.t. Bag-of-Words (BOW)
 But you will **not** be tested about their technical approach in **exam**

Words → Concepts: Word Embeddings



Not in exam



<https://www.tensorflow.org/tutorials/text/word2vec>

Source: <https://medium.com/@hari4om/word-embedding-d816f643140>

DATA

5 tensors found
Word2Vec 10K

Label by word | Color by No color map

Edit by word | Tag selection as

Load | Publish | Download | Label

☒ Sphereize data

Checkpoint: Demo datasets

Metadata: oss_data/word2vec_10000_200d

UMAP | T-SNE | PCA | CUSTOM

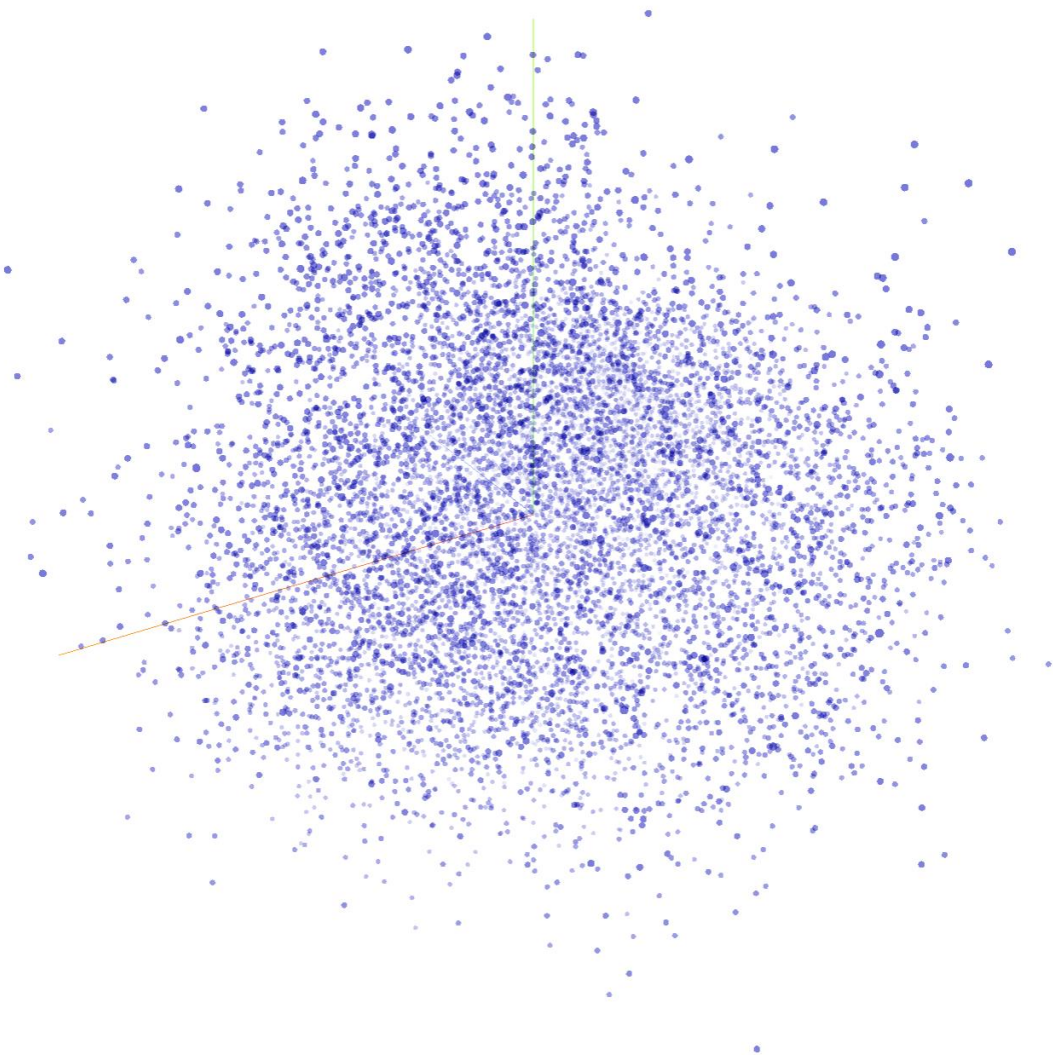
X Component #1 | Y Component #2

Z Component #3 ☒

PCA is approximate.

Total variance described: 8.5%.

Points: 10000 | Dimension: 200



Show All Data | Isolate 101 points | Clear selection

Search by word

BOOKMARKS (0)





Questions!



Wrapping Up

What did we learn?

1. Describe **techniques** of feature extraction/engineering for different data types

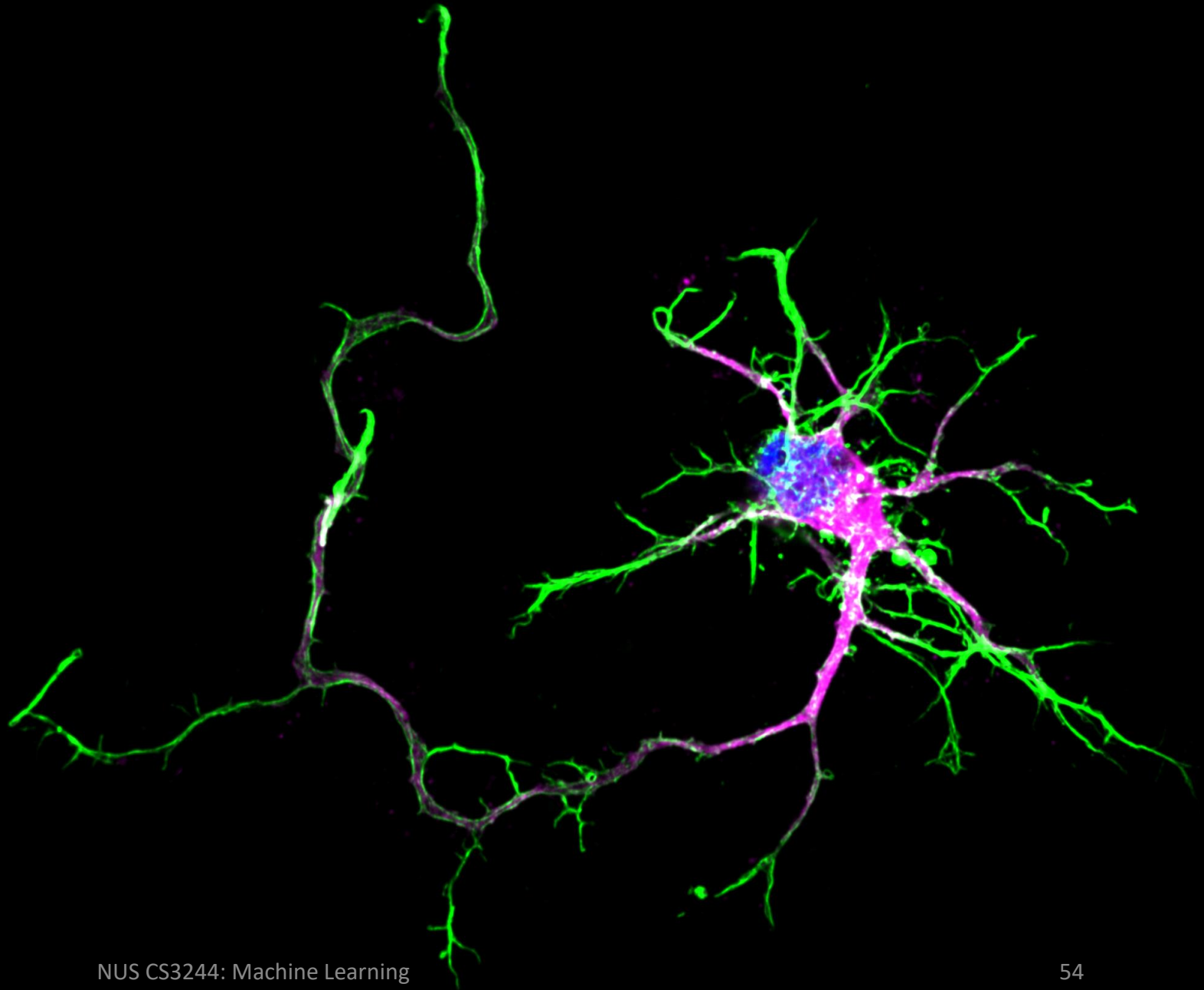
Tabular	Temporal	Image	Text
<ul style="list-style-type: none">• Domain-specific custom equations• Features from counting, aggregation, difference, min, max	<ul style="list-style-type: none">• Features from previous values, aggregate statistics, linear regression• Wave analysis features	<ul style="list-style-type: none">• RGB image as 3D tensor• Color features from RGB histogram• Shape features from edge detection• Edge detection via Convolution	<ul style="list-style-type: none">• Tokenization• Stemming, Lemmatization• Stop words• Bag-of-Words encoding

2. Describe **issues** when extracting features for various data types

Useful Programming Libraries for Feature Extraction

- General / Tabular ([pandas](#), [NumPy](#), [scikit-learn](#))
- Images / Computer Vision ([OpenCV](#), [Pillow/PIL](#), [scikit-image](#))
- Text / Natural Language ([NLTK](#), [CoreNLP](#), [spaCy](#))
- Temporal / Time Series ([tsfresh](#))

Next week: Perceptron and Neural Networks



W09 Pre-Lecture Task (due before next Mon)

Watch

1. [But what is a neural network? | Chapter 1, Deep learning](#) (~20 min) by [3Blue1Brown](#)
2. [The Nervous System, Part 1: Crash Course A&P #8](#) (~10 min) by [CrashCourse](#)

Discuss

1. Reflect on how artificial neural networks are different from human neural networks.
2. Identify **one** point (no need to write several).
3. Post a 1–2 sentence answer to the topic in your tutorial group: [#tg-xx](#)