Introduction to Natural Language Processing and Computer Vision, Feature Representation, & Dimensionality Reduction

Danna Gurari

University of Texas at Austin

Spring 2021

Review

- Last week:
  - One-vs-all multiclass classification
  - Classifier confidence
  - Evaluation: ROC and PR-curves
  - Ensemble learning
- Assignments (Canvas)
  - Lab assignment 2 due yesterday
  - Problem set 5 due next week
  - Project pre-proposal due next week
  - Lab assignment 3 due in two weeks
- Questions?

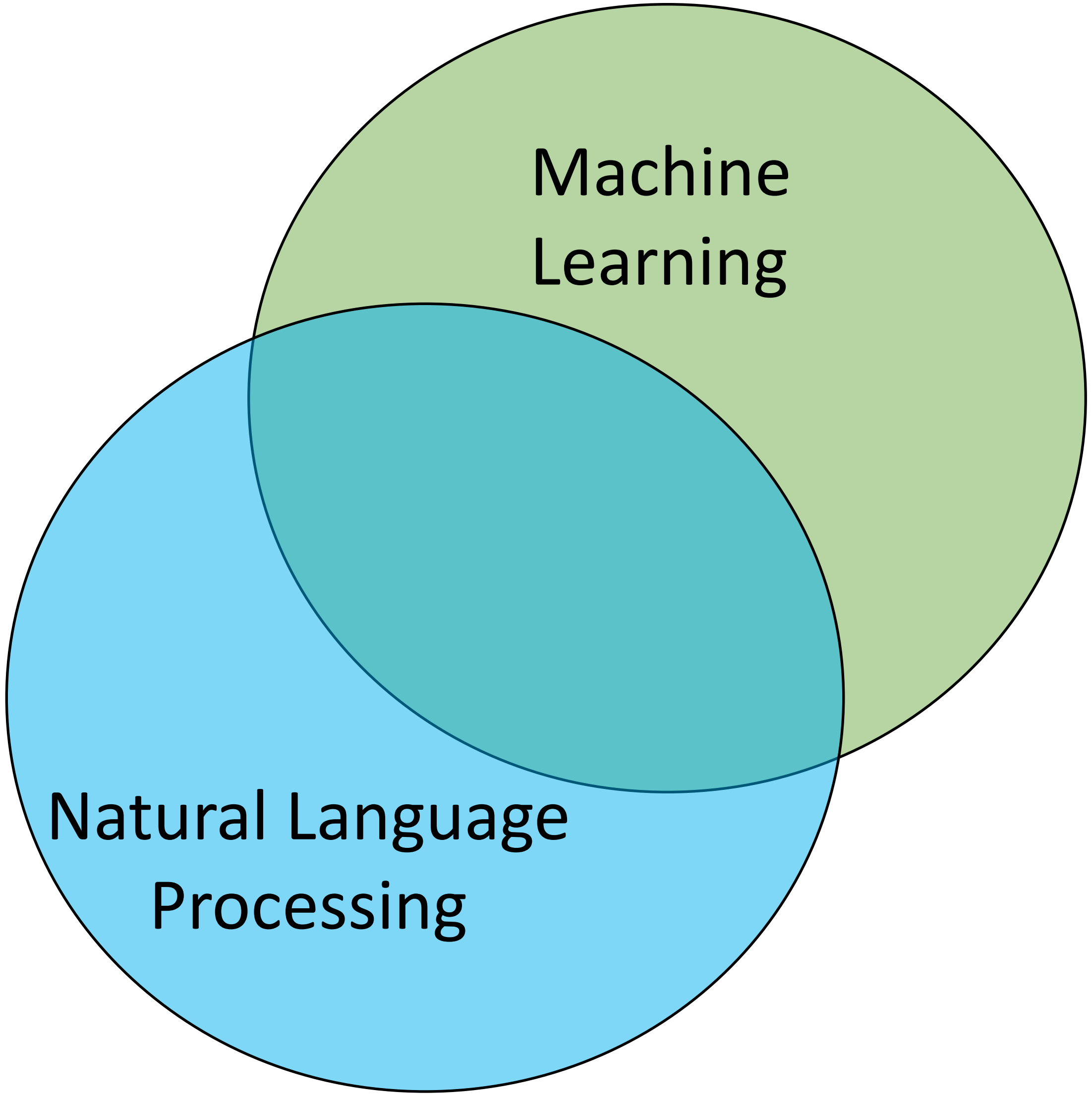
Today’s Topics

- Natural Language Processing
- Computer Vision
- Feature Representation
- Dimensionality Reduction
- Lab

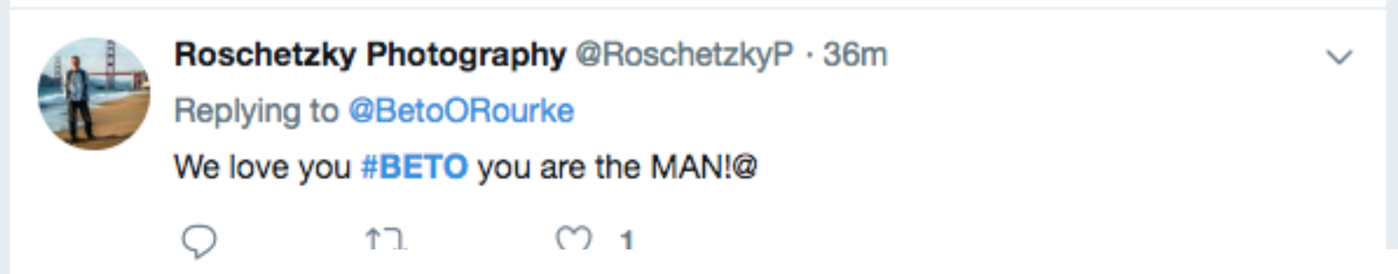
Today's Topics

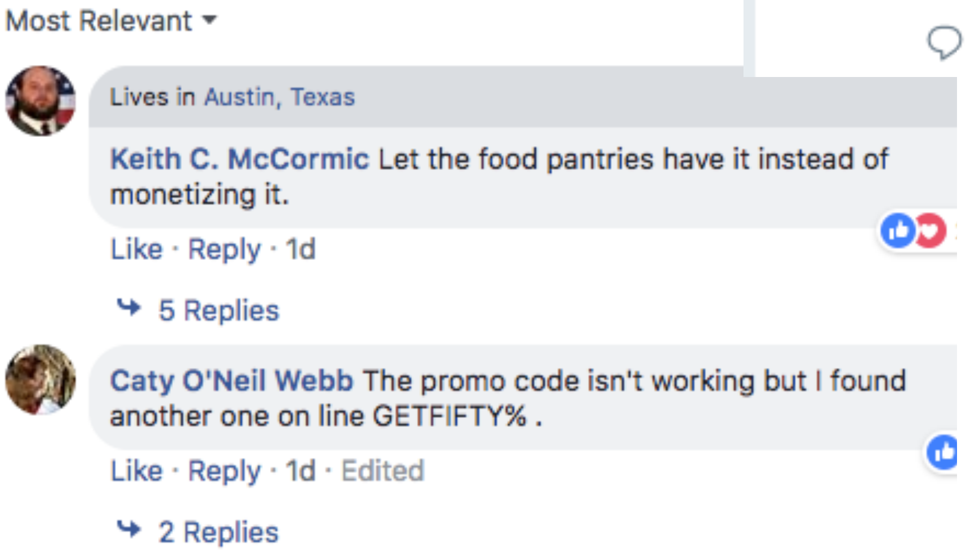
- Natural Language Processing
- Computer Vision
- Feature Representation
- Dimensionality Reduction
- Lab

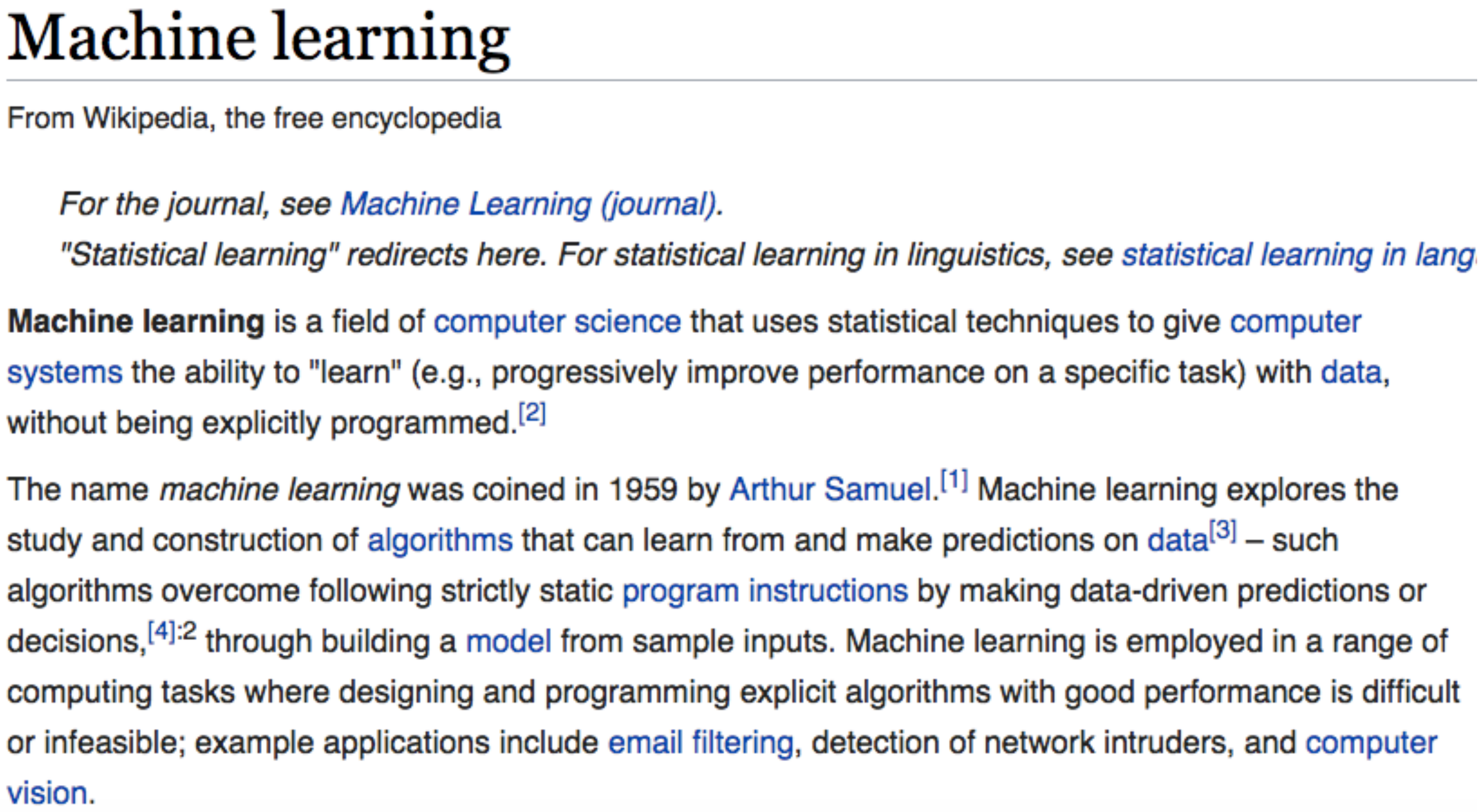
Relationship Between ML and NLP



Task Input: String (Collection of Characters)


Roschetzky Photography @RoschetzkyP · 36m
Replying to @BetoORourke
We love you #BETO you are the MAN!@


Most Relevant
Lives in Austin, Texas
Keith C. McCormic Let the food pantries have it instead of monetizing it.
Like · Reply · 1d
5 Replies
Caty O'Neil Webb The promo code isn't working but I found another one on line GETFIFTY% .
Like · Reply · 1d · Edited
2 Replies


Machine learning

From Wikipedia, the free encyclopedia

For the journal, see Machine Learning (journal).
"Statistical learning" redirects here. For statistical learning in linguistics, see statistical learning in lang

Machine learning is a field of computer science that uses statistical techniques to give computer systems the ability to "learn" (e.g., progressively improve performance on a specific task) with data, without being explicitly programmed.[2]

The name *machine learning* was coined in 1959 by Arthur Samuel.[1] Machine learning explores the study and construction of algorithms that can learn from and make predictions on data[3] – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions,[4]:2 through building a model from sample inputs. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or infeasible; example applications include email filtering, detection of network intruders, and computer vision.

- Common terms
  - **Corpus:** dataset
  - **Document:** example

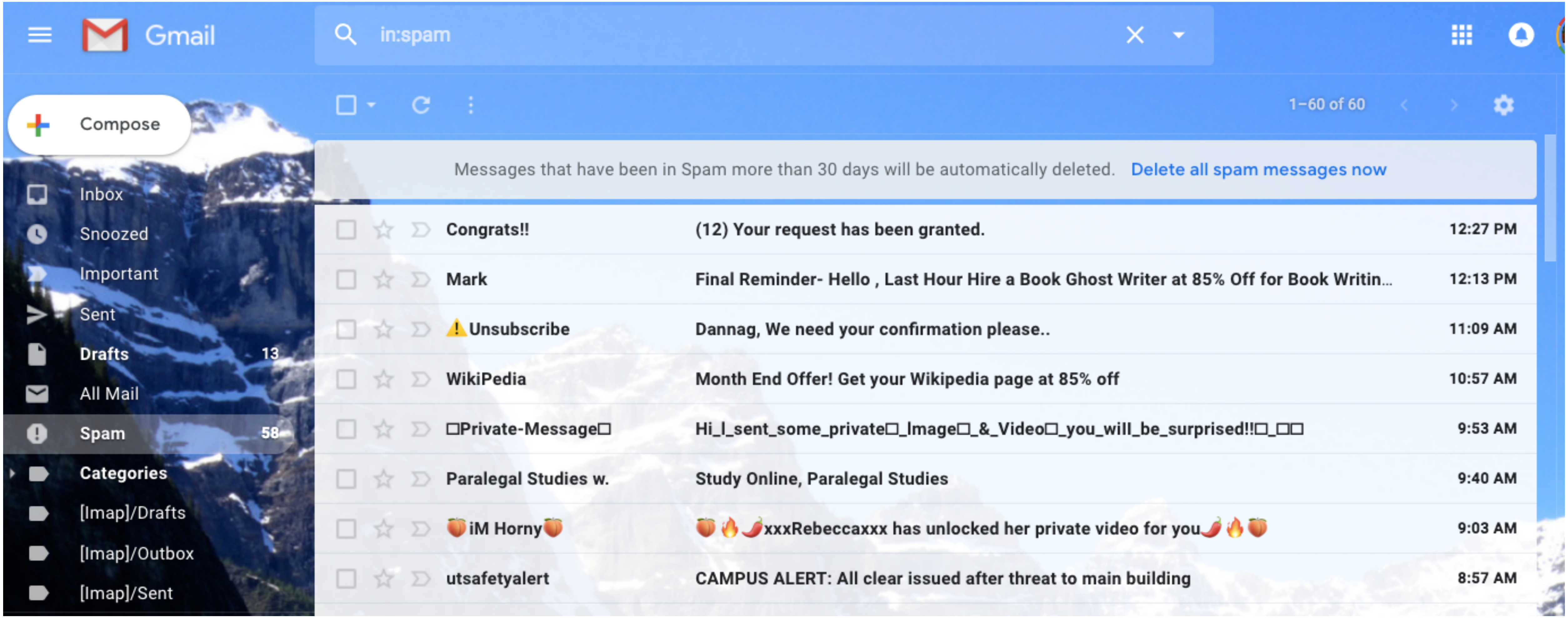
Task Input: String (Collection of Characters)

Which "String" Feature Types Apply?

- ~~Categorical data~~
  - ~~Comes from a fixed list (e.g., education level)~~
- ~~Structured string data~~
  - ~~e.g., addresses, dates, telephone numbers,~~
- Text data

Applications: Spam Detection

Input: email; **Output**: yes/no

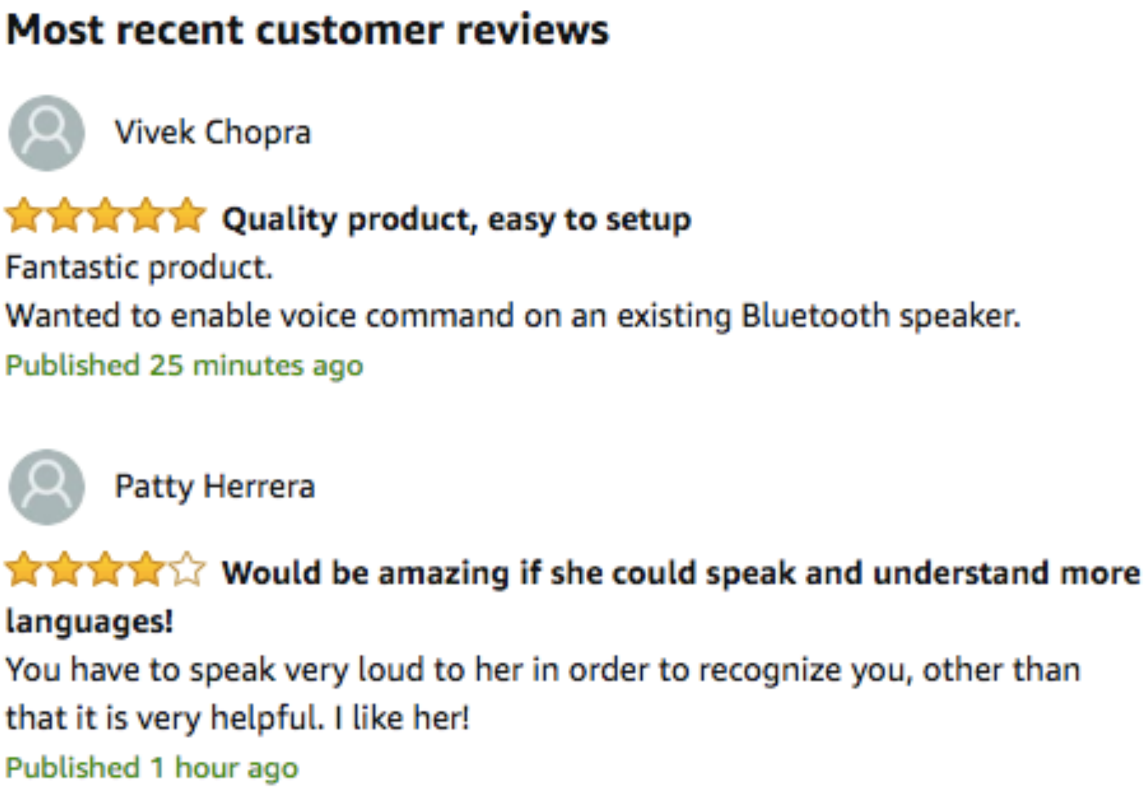


Applications: Opinion Mining

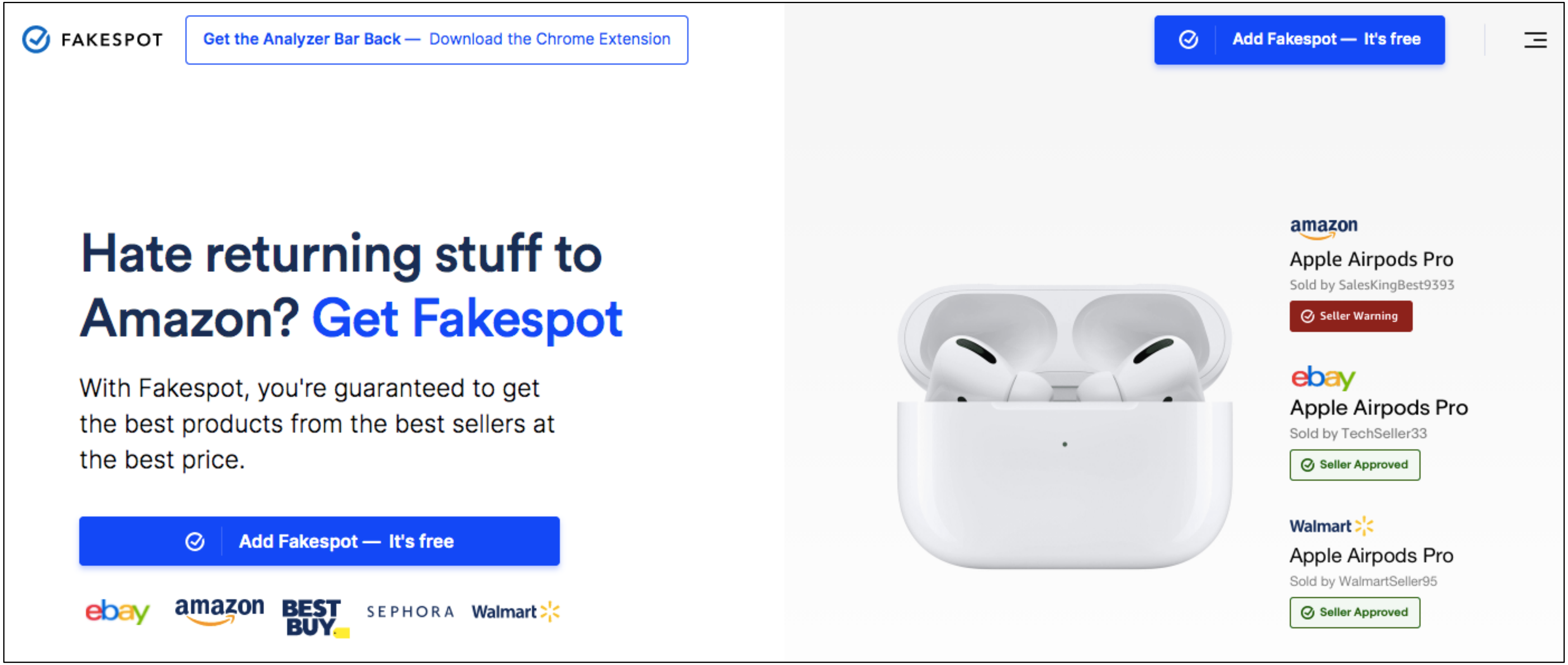
Input: script of speeches/tweets; **Output**: yes/no

e.g., Politics

e.g., Marketing



Applications: Detect Fraudulent Reviews and Third-Party Sellers



Applications: Machine Translation

Input: text; **Output**: text

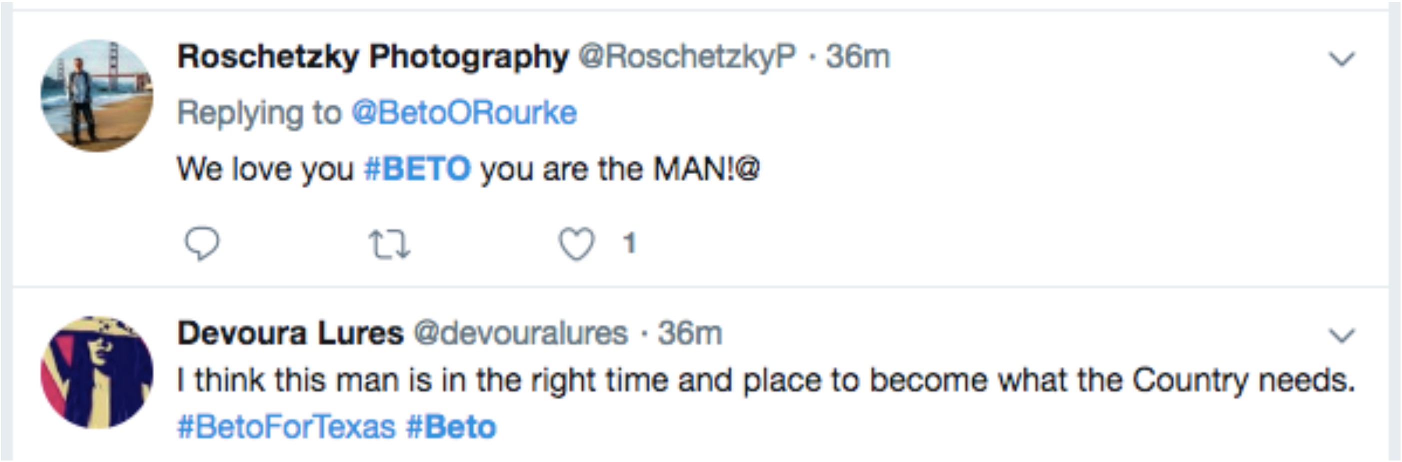
What does this say in English?

Applications

What are other natural language processing applications?

How to Describe Text to a Computer?

- Challenge: input often varies in length



- Solution: convert text to numeric format that ML algorithms can handle

How to Describe Text to a Computer?
- Pre-processing

- Tokenization: convert sequence of characters into sequence of tokens (typically, words)
  - e.g.,

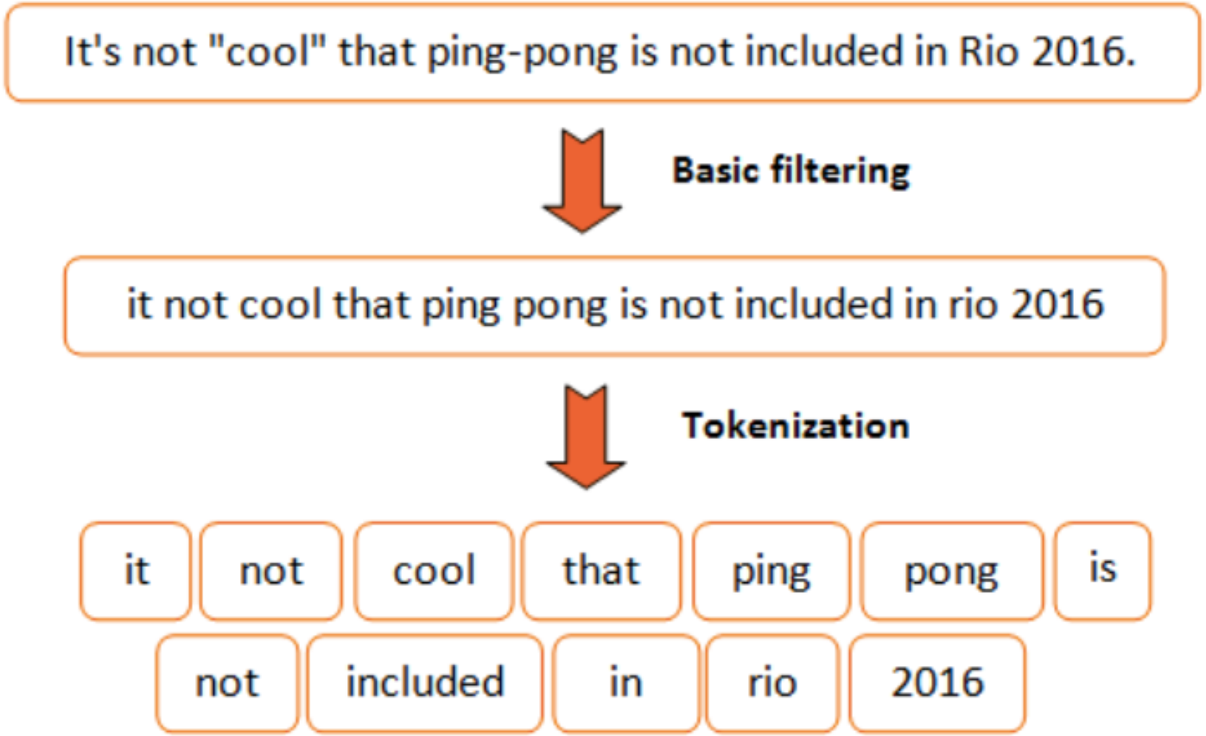
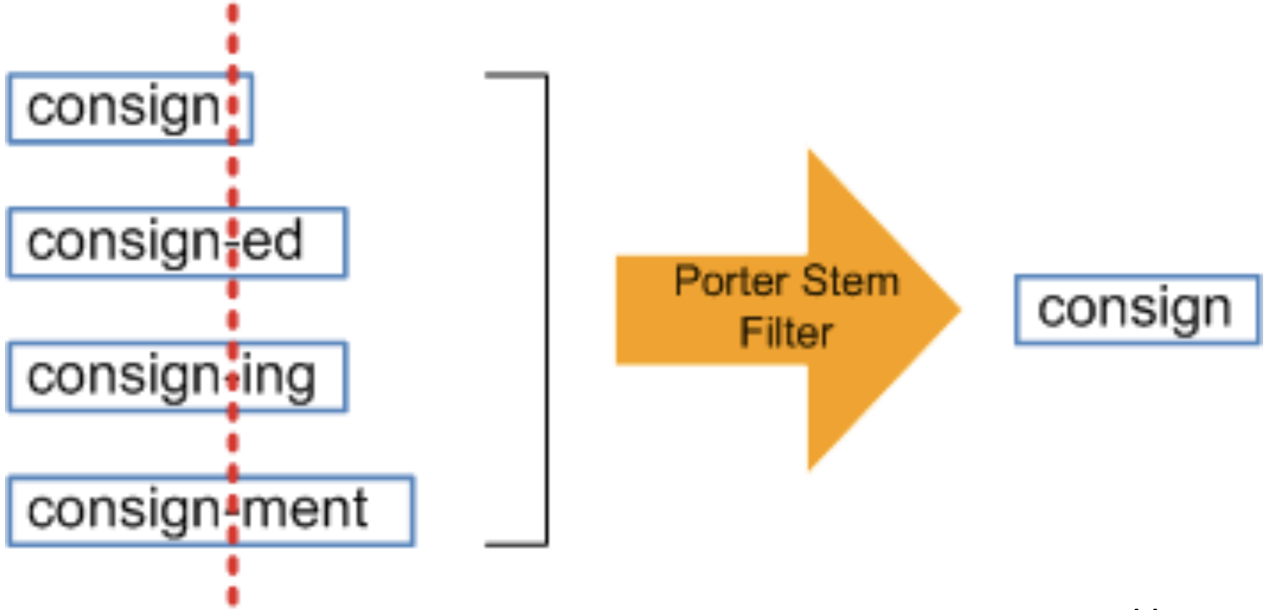


Figure: https://www.meaningcloud.com/developer/resources/doc/models/models/text-tokenization-multiw

How to Describe Text to a Computer? - Pre-processing

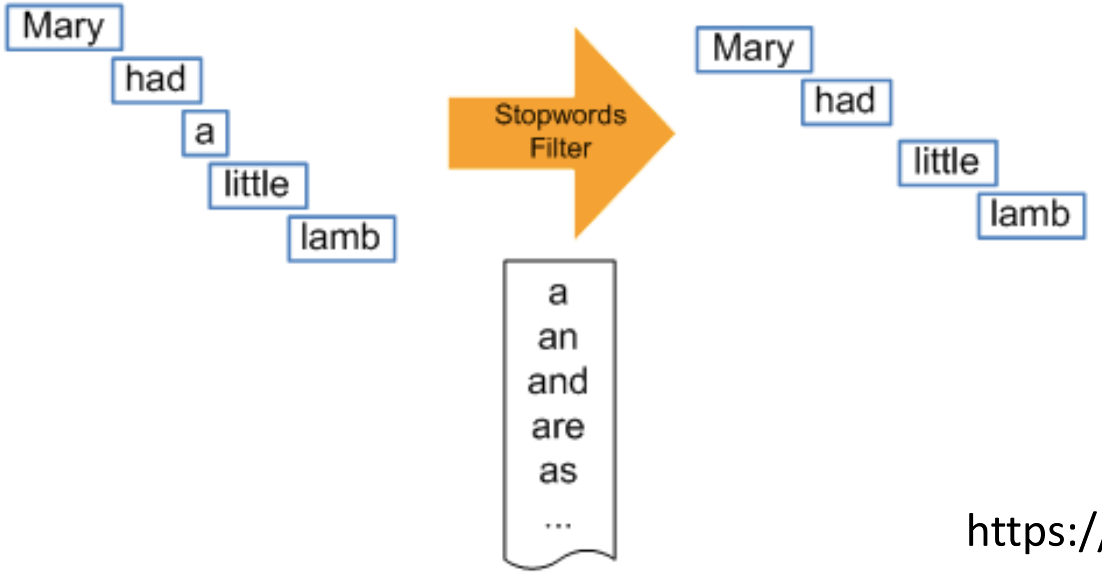
- Tokenization: convert sequence of characters into sequence of tokens (typically, words)
- Stemming: represent each word using its word stem such as by resolving singular versus plural, different verb forms, and more
  - e.g.,



https://dzone.com/articles/using-lucene-grails

How to Describe Text to a Computer? - Pre-processing

- Tokenization: convert sequence of characters into sequence of tokens (typically, words)
- Stemming: represent each word using its word stem such as by resolving singular versus plural, different verb forms, and more
- Stopword removal: discard words that are too frequent to be informative
  - e.g.,

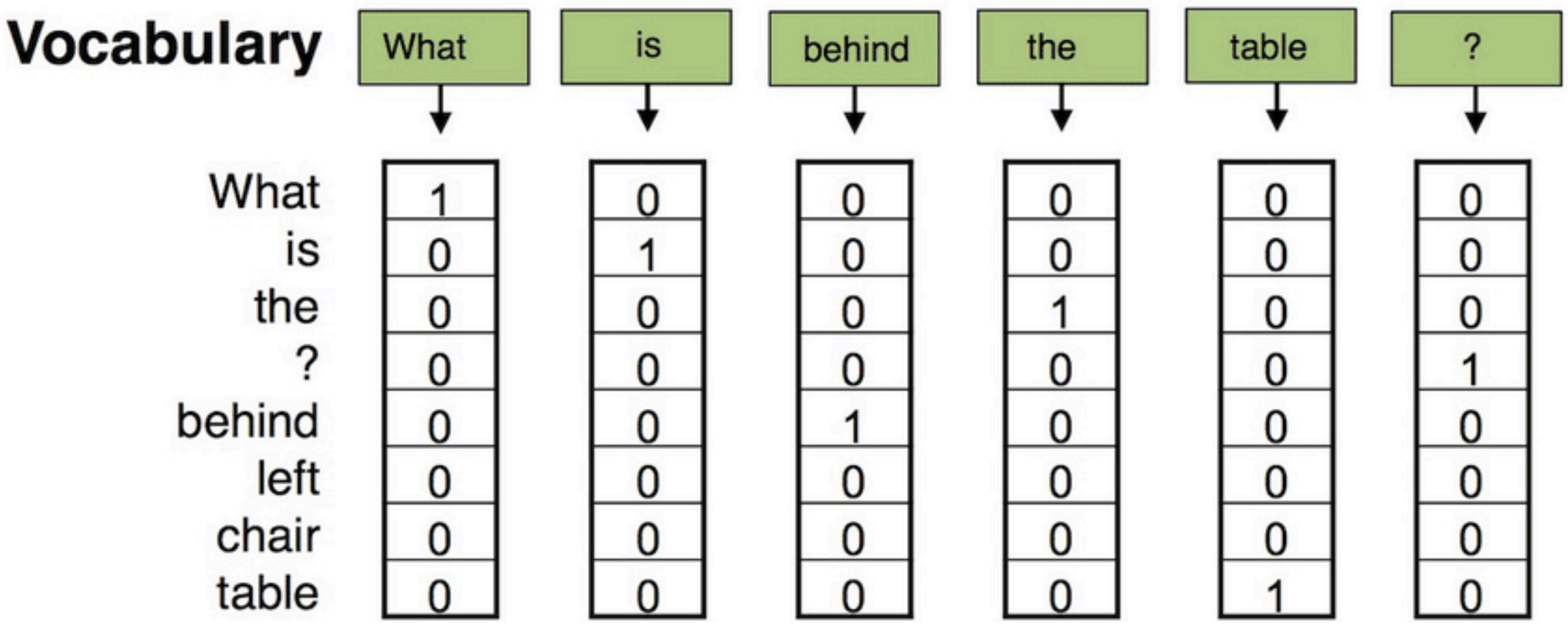


https://dzone.com/articles/using-lucene-grails

How to Describe Text to a Computer?
- Bag of Words

- Goal: convert each document into a fixed-length vector
- Algorithm:
    1. Learn vocabulary: all unique words in training data
    2. Encode vector: word counts for each document

e.g.,



How to Describe Text to a Computer?
- Bag of Words

- Goal: convert each document into a fixed-length vector
- Algorithm:
  1. Learn vocabulary: all unique words in training data

'All my cats in a row',
'When my cat sits down, she looks like a Furby toy!',

vocabulary
all
cat
cats
down
furby
in
like
looks
my
row
she
sits
toy
when

Adapted from: https://pythonprogramminglanguage.com/bag-of-words/

How to Describe Text to a Computer?
- Bag of Words

- Goal: convert each document into a fixed-length vector
- Algorithm:
  1. Learn vocabulary: all unique words in training data
  2. Encode vector: word counts for each document

e.g., “All my cats in a row”

[1 0 1 0 0 1 0 0 1 1 0 0 0 0]

e.g., “When my cat sits down, she looks like a Furby toy!

[0 1 0 1 1 0 1 1 1 0 1 1 1 1]

vocabulary
all
cat
cats
down
furby
in
like
looks
my
row
she
sits
toy
when

Adapted from: https://pythonprogramminglanguage.com/bag-of-words/

How to Describe Text to a Computer?
- TD-IDF (Term Frequency-Inverse Document Frequency)

- Motivation: avoid high frequency words with little useful content (e.g., “the”, “is”, “he”, “she”, “they”, etc)
- Idea: penalize frequent words that are frequent across all documents
- Algorithm:
  1. Compute term-frequency:
     - # of times a term *t* occurs in document *d*
  2. Compute inverse document frequency:

     Reduces weight on low document frequencies

     Total number of documents

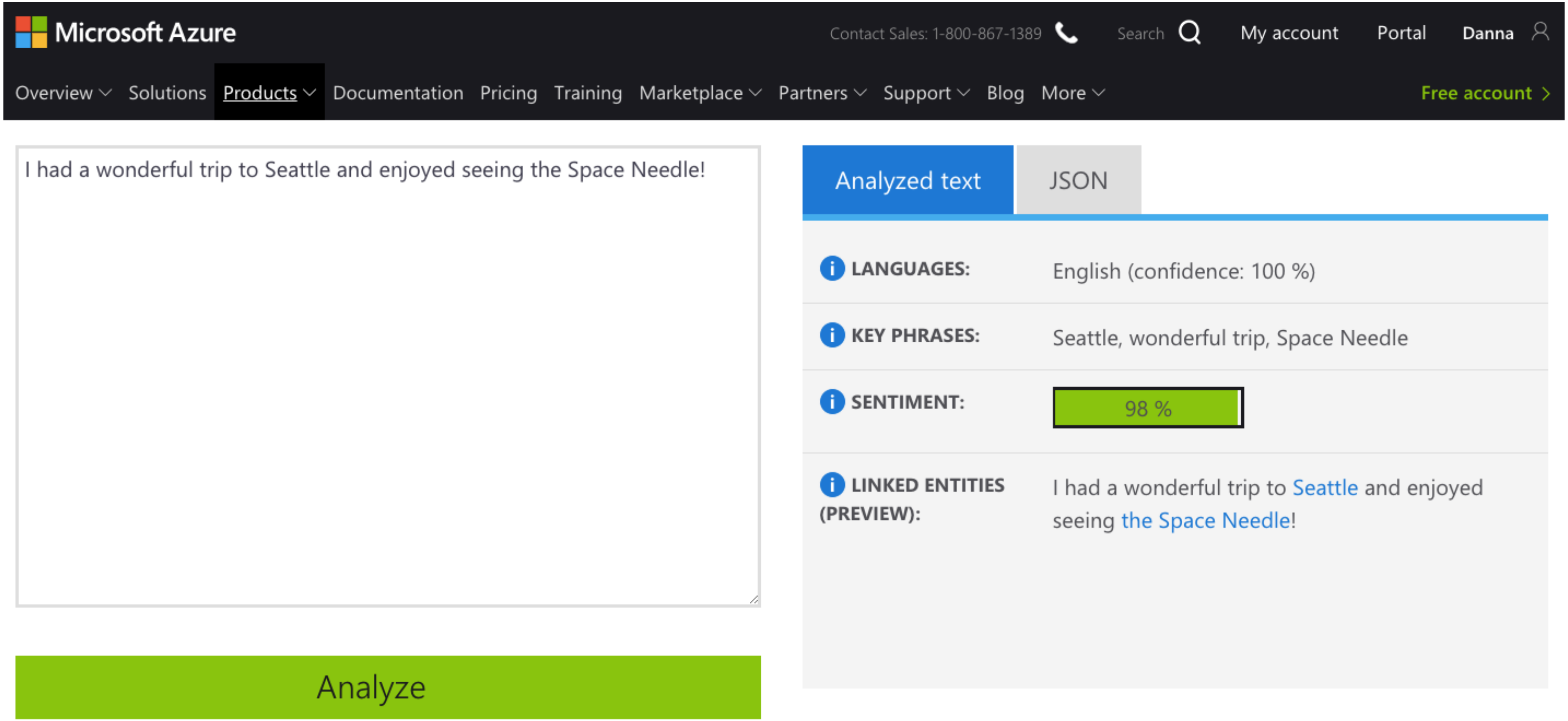
     $$\mathrm{idf}(\mathrm{t,d}) = \boxed{log}\,\frac{\boxed{n_d}}{\boxed{1} + \boxed{\mathrm{df}(\mathrm{d,t})}}$$

     To use non-zero value when term occurs in all documents

     Number of documents containing term t
  3. Compute TD-IDF:

$$\text{tf-idf}(\mathrm{t,d}) = tf(t,d) \times \mathrm{idf}(\mathrm{t,d})$$

How to Describe Text to a Computer? - Microsoft Azure: Text Analytics API



How to Describe Text Beyond English?

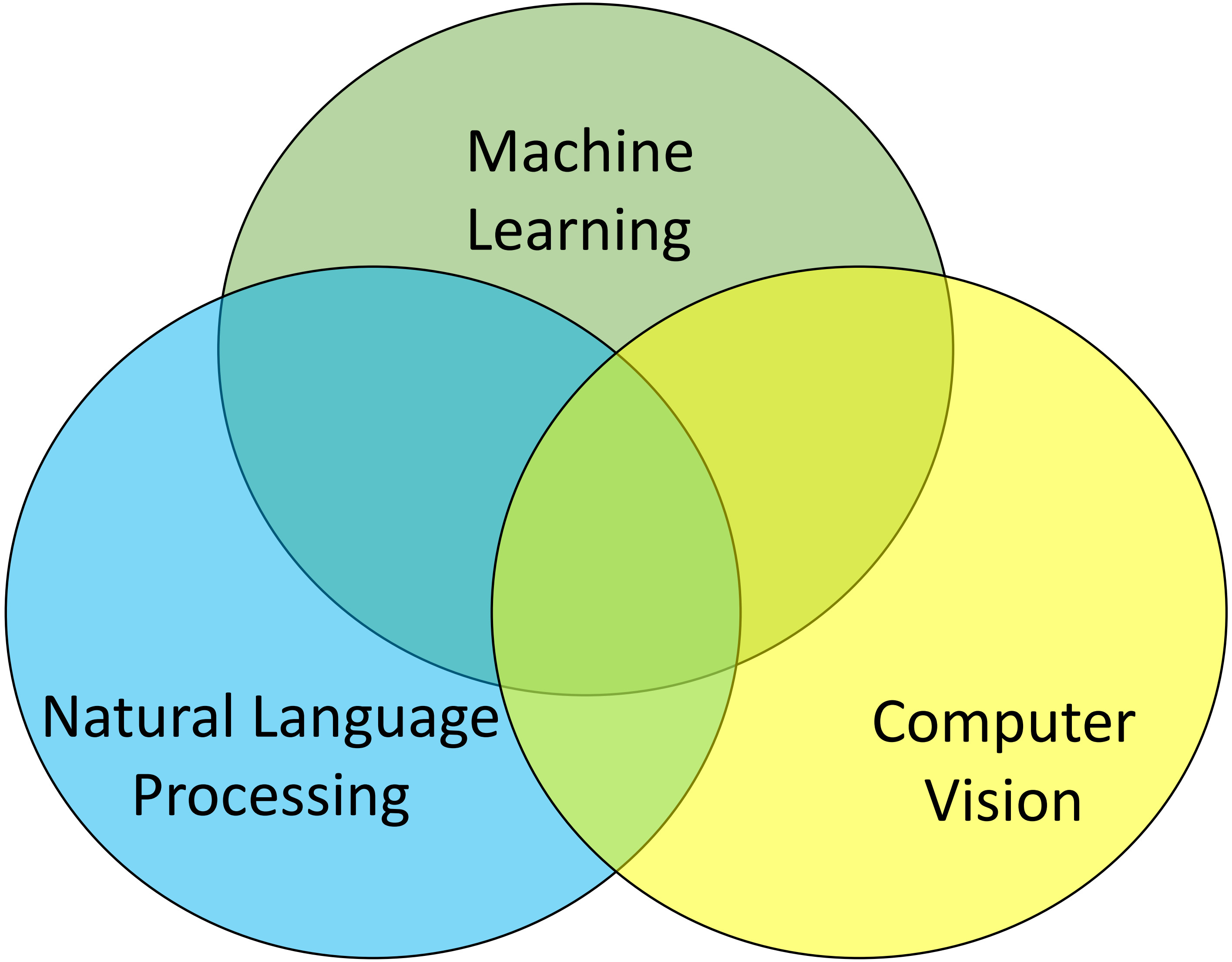
- 7000+ languages spoken around the world
- Limited NLP for languages beyond English
  - e.g., Stanford Word Segmenter for Arabic and Chinese

https://ruder.io/nlp-beyond-english/

Today’s Topics

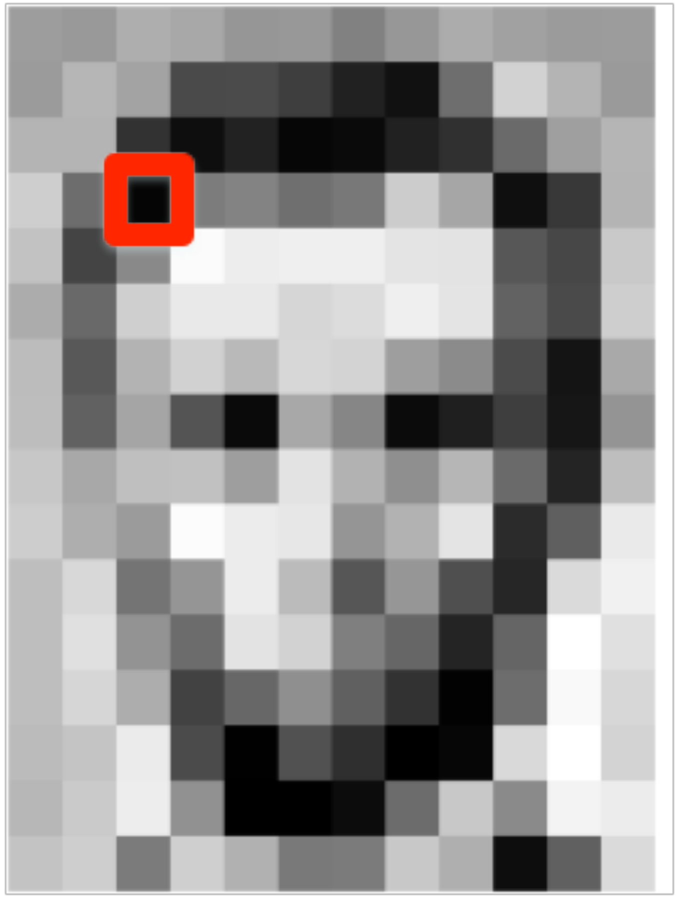
- Natural Language Processing
- Computer Vision
- Feature Representation
- Dimensionality Reduction
- Lab

Relationship Between ML, CV, and NLP

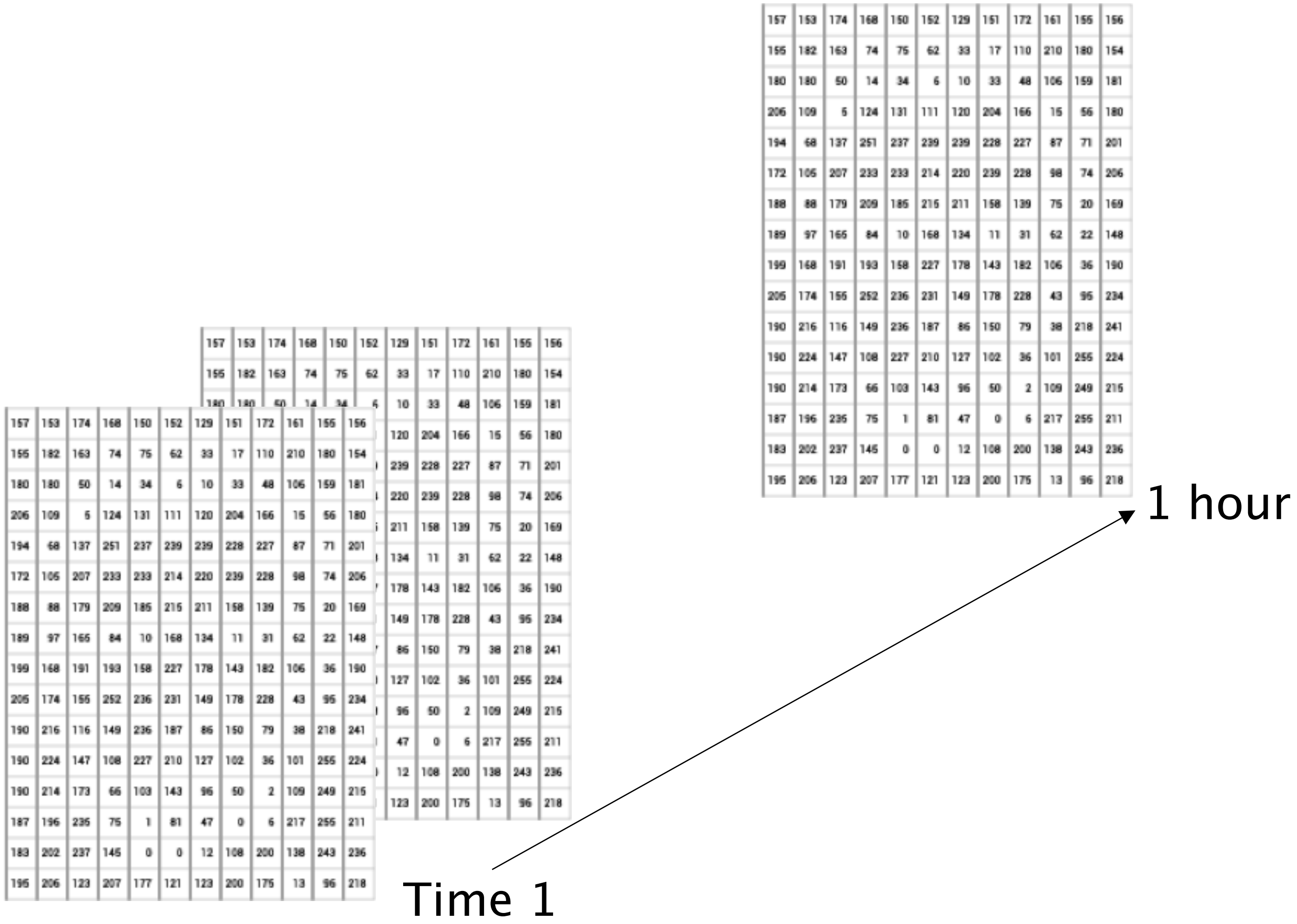


Task Input: Matrix

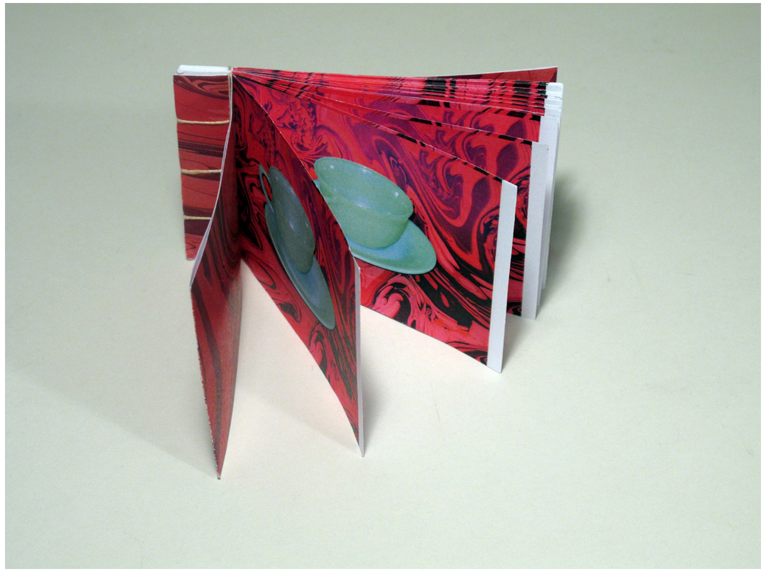
157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	[illegible]	5	[illegible]	131	111	120	204	166	15	56	180
194	68	[illegible]	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218



Task Input: Matrix (Video)

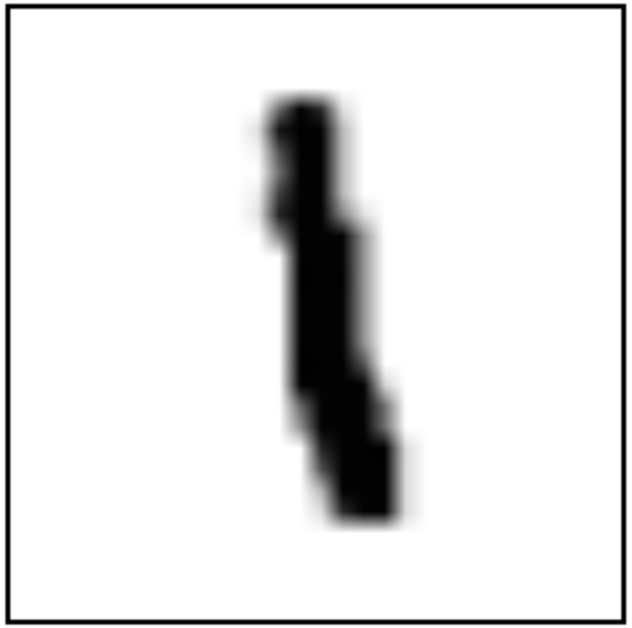


Analogous to:



How to Describe an Image to a Computer?

- Raw pixel values
  - e.g. MNIST: how many “features” would be in an image (28 x 28 pixels)



$$\approx \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&.6&.8&0&0&0&0&0&0\\ 0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&.5&1&.4&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&.7&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&.9&1&.1&0&0&0&0\\ 0&0&0&0&0&0&0&.3&1&.1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}$$

  - 784

http://colah.github.io/posts/2014-10-Visualizing-MNIST/

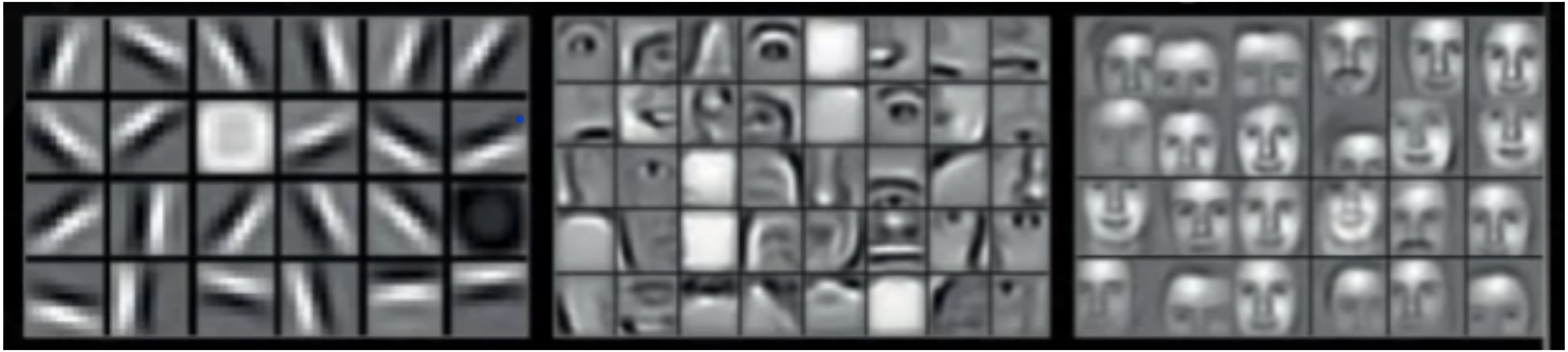
How to Describe an Image to a Computer?

- Raw pixel values
  - e.g. LFW: how many “features” would be in an image (50 x 37 pixels)

- 1,850

http://vis-www.cs.umass.edu/lfw/

How to Describe an Image to a Computer?
Low-Level to High-Level Representations



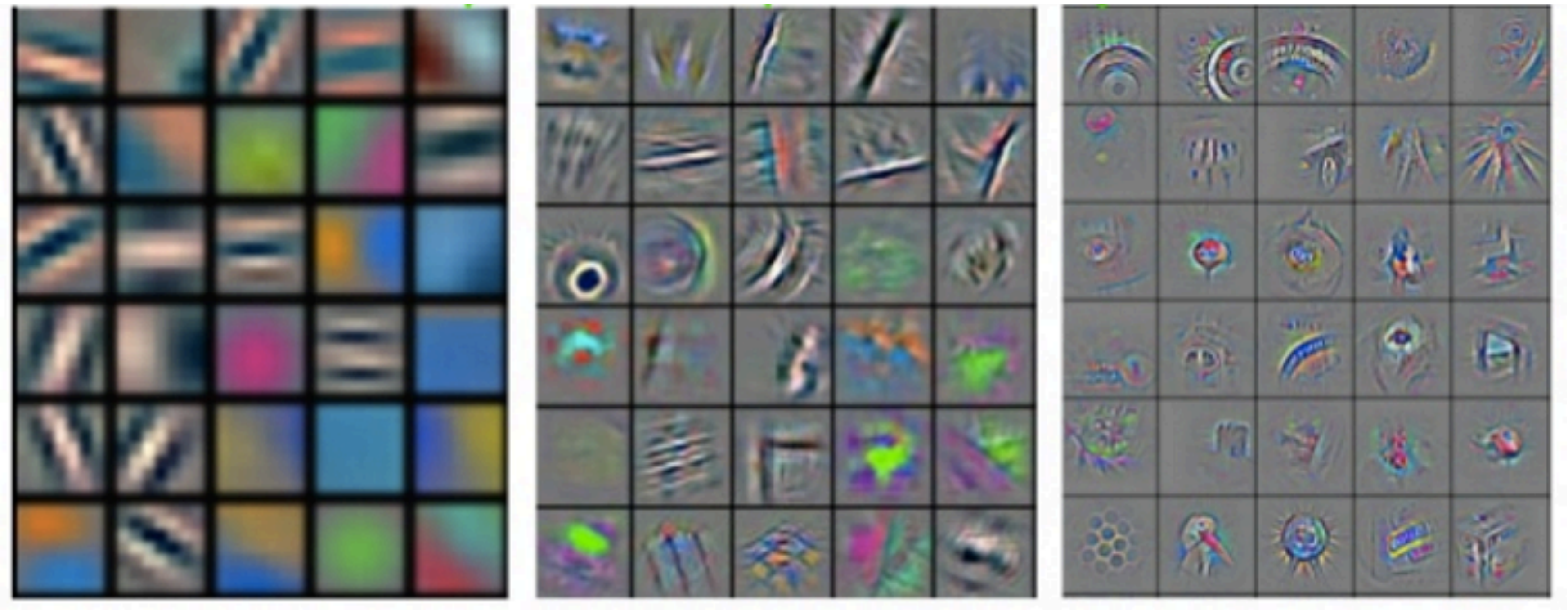
<u>Low-level features</u>
e.g., dots, edges, corners, lines, curves

<u>Mid-level features</u>
e.g., forms, colors

<u>High-level features</u>
e.g., objects, scenes, emotions

Adapted from https://dzone.com/articles/deep-learning-vs-machine-learning-the-hottest-topi

How to Describe an Image to a Computer? Low-Level to High-Level Representations



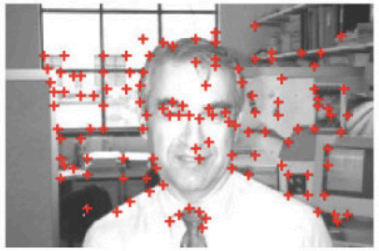
Low-level features Mid-level features High-level features

Adapted from https://www.slideshare.net/embeddedvision/01-am-keynotelecun

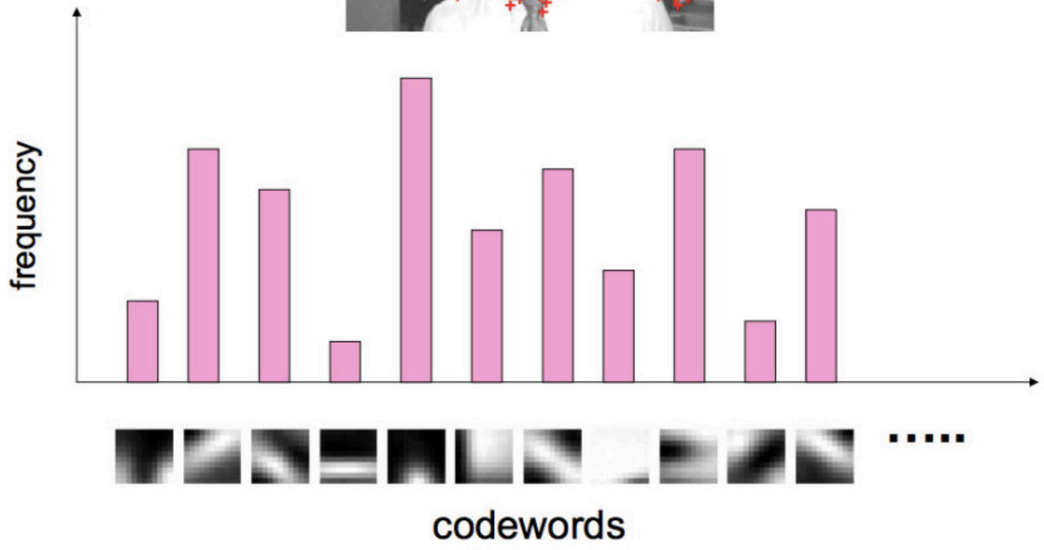
How to Describe an Image to a Computer?
- Bag of (Visual) Words

- Goal: convert each image into a fixed-length vector
- Algorithm:
  1. Learn vocabulary
     e.g., using HOG descriptors

  2. Encode vector



e.g.,



Good tutorial and image credit: https://gurus.pyimagesearch.com/the-bag-of-visual-words-model/
Good tutorial: https://jacobgil.github.io/machinelearning/bag-of-words

How to Describe an Image to a Computer? - Microsoft Azure Face API


```
Detection result:
JSON:
 [
  {
    "faceId": "83c0f042-8c96-4b00-97dd-66bdeba3b6bc",
    "faceRectangle": {
      "top": 128,
      "left": 459,
      "width": 224,
      "height": 224
    },
    "faceAttributes": {
      "hair": {
        "bald": 0.0,
        "invisible": false,
        "hairColor": [
          {
            "color": "brown",
            "confidence": 1.0
          },
          {
            "color": "blond",
            "confidence": 0.69
```

How to Describe an Image to a Computer? - Microsoft Azure Computer Vision API

FEATURE NAME:	VALUE
Description	{ "tags": ["train", "platform", "station", "building", "indoor", "subway", "track", "walking", "waiting", "pulling", "board", "people", "man", "luggage", "standing", "holding", "large", "woman", "yellow", "suitcase"], "captions": [{ "text": "people waiting at a train station", "confidence": 0.833099365 }] }
Tags	[{ "name": "train", "confidence": 0.9975446 }, { "name": "platform", "confidence": 0.995543063 }, { "name": "station", "confidence": 0.9798007 }, { "name": "indoor", "confidence": 0.927719653 }, { "name": "subway", "confidence": 0.838939846 }, { "name": "pulling", "confidence": 0.431715637 }]
Image format	"Jpeg"

Today’s Topics

- Natural Language Processing
- Computer Vision
- Feature Representation
- Dimensionality Reduction
- Lab

Real World Data Challenges

- Different data representations
- Missing data
- Different numerical scales

e.g.,

Dataset	Class Length (Categorical)	Rain (cm) (Numerical)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long		Yes
Test	Medium	0.1	Yes
Test	Short		Yes
Test		0.5	No

Categorical Variables

- Categorical
  - Nominal (2 or more categories with no ordering)
    - e.g., gender
  - Ordinal (categories with clear ordering)
    - e.g., t-shirt size, education level
- How to convert categorical to numerical variable?
  - Bad idea to map each category to a number

e.g.,

Categorical

Dataset	Class Length	Rain (cm)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long		Yes
Test	Medium	0.1	Yes
Test	Short		Yes
Test		0.5	No

Categorical Variables: One-Hot Encoding

- One-hot encoding: add one new feature per category
- How many features will be made for "Type"?
  - 2
- How many features will be made for "Length"?
  - 3
- How many features would the example dataset have with a one-hot encoding?
  - 6

e.g.,

Type	Length	IMDb_Rating	Liked
Comedy	Short	7.2	Yes
Drama	Medium	9.3	Yes
Comedy	Medium	5.1	No
Drama	Long	6.9	No
Drama	Medium	8.3	Yes
Drama	Short	4.5	No
Comedy	Short	8.0	Yes
Drama	Medium	7.5	Yes

Categorical Variables: One-Hot Encoding

IMDb_Rating	Type_Comedy	Type_Drama	Length_Long	Length_Medium	Length_Short
7.2	1	0	0	0	1
9.3	0	1	0	1	0
5.1	1	0	0	1	0
6.9	0	1	1	0	0
8.3	0	1	0	1	0
4.5	0	1	0	0	1
8.0	1	0	0	0	1
7.5	0	1	0	1	0

- What new challenges arise?
  - Large, sparse matrices
  - Test set may have value not observed in training

Missing Data

- How to replace missing values?
  - Ignore the tuple
  - Manually insert missing values
  - Insert global constant (e.g., 0)
  - Attribute mean
  - Attribute mean for all samples belonging to same class
  - And more…
- Algorithm
  1. Learn on training data
  2. Transform training data
  3. Transform test data

e.g.,

Dataset	Class Length	Rain (cm)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long		Yes
Test	Medium	0.1	Yes
Test	Short		Yes
Test		0.5	No

Missing Data: Impute mean values for rain

- Algorithm
  1. **Learn on training data**
  2. Transform training data
  3. Transform test data
- What is the value to impute?
  - 8.7/10 = 0.87

e.g.,

Dataset	Class Length	Rain (cm)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long		Yes
Test	Medium	0.1	Yes
Test	Short		Yes
Test		0.5	No

Missing Data: Impute mean values for rain

- Algorithm
  1. Learn on training data
  2. **Transform training data**
  3. **Transform test data**
- What is the value to impute?
  - 8.7/10 = 0.87

e.g.,

Dataset	Class Length	Rain (cm)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long	$\boxed{0.87}$	Yes
Test	Medium	0.1	Yes
Test	Short	$\boxed{0.87}$	Yes
Test		0.5	No

Missing Data: Impute mean values for rain for all samples belonging to same class

- Algorithm
  1. **Learn on training data**
  2. Transform training data
  3. Transform test data
- What is the value to impute?
  - "Yes"
    - 6.5/7 = 0.93
  - "No"
    - 2.2/3 = 0.73

e.g.,

Dataset	Class Length	Rain (cm)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long		Yes
Test	Medium	0.1	Yes
Test	Short		Yes
Test		0.5	No

Missing Data: Impute mean values for rain for all samples belonging to same class

- Algorithm
  1. Learn on training data
  2. **Transform training data**
  3. **Transform test data**
- What is the value to impute?
  - "Yes"
    - 6.5/7 = 0.93
  - "No"
    - 2.2/3 = 0.73

e.g.,

Dataset	Class Length	Rain (cm)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long	0.93	Yes
Test	Medium	0.1	Yes
Test	Short	0.93	Yes
Test		0.5	No

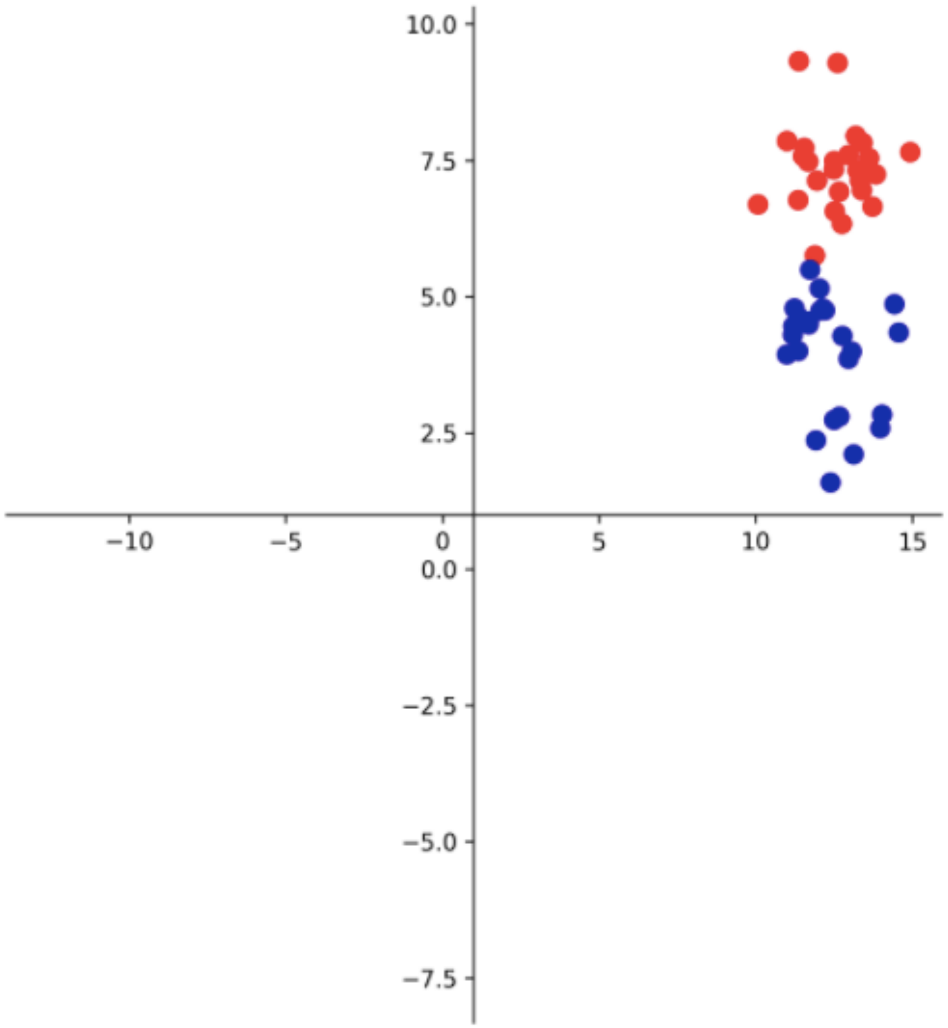
Different numerical scales

e.g.,

Numerical

Dataset	Class Length	Rain (cm)	Attend Class?
Train	Short	1.1	Yes
Train	Medium	2.3	Yes
Train	Medium	0	No
Train	Long	0.7	No
Train	Medium	0.3	Yes
Train	Short	1.5	No
Train	Short	0	Yes
Train	Medium	1.5	Yes
Train	Medium	0.7	Yes
Train		0.6	Yes
Train	Long		Yes
Test	Medium	0.1	Yes
Test	Short		Yes
Test		0.5	No

Different numerical scales



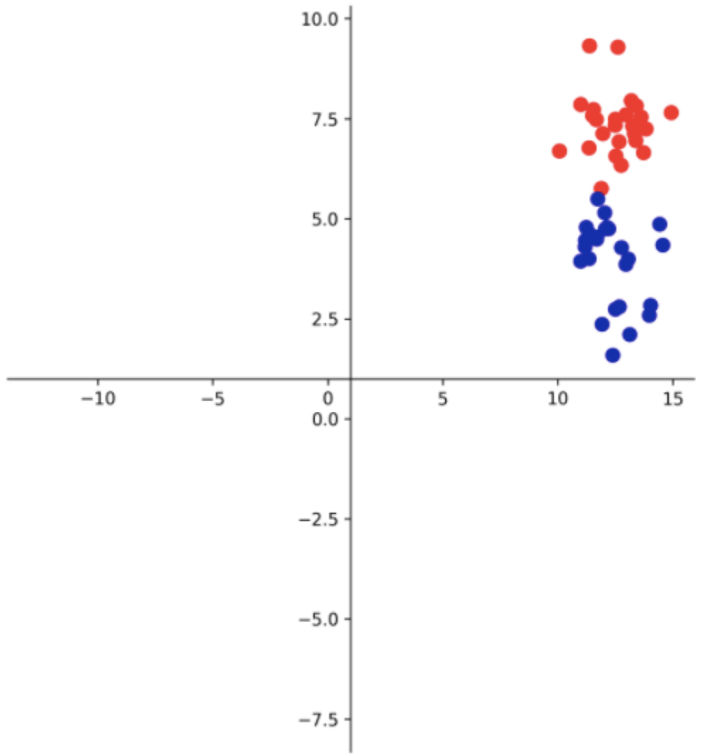
What is range of feature 1 values?

What is range of feature 2 values?

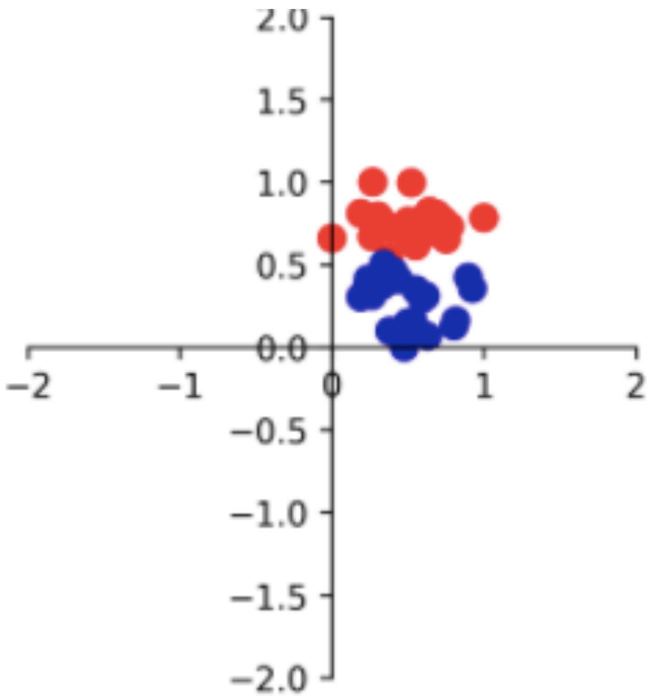
https://github.com/amueller/introduction_to_ml_with_python/blob/master/03-unsupervised-learning.ipynb

Different numerical scales: Solutions

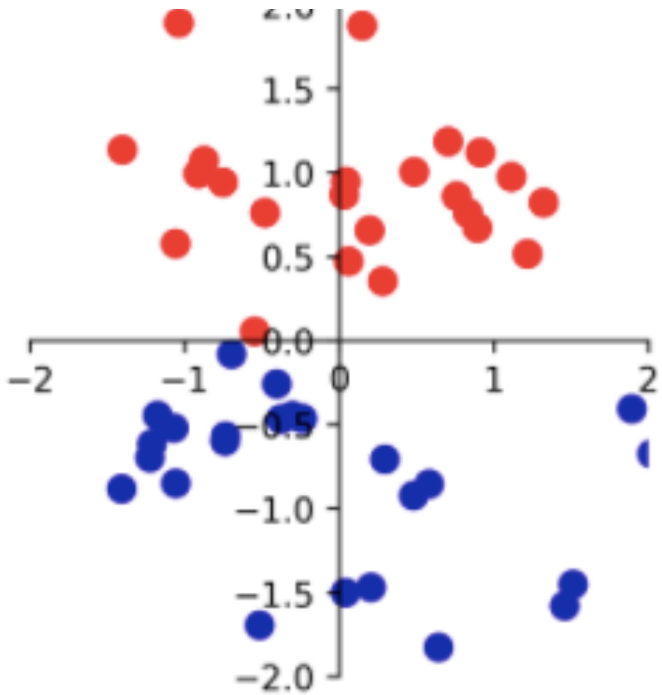
Original Data



Min-Max Scaling



Standardization



https://github.com/amueller/introduction_to_ml_with_python/blob/master/03-unsupervised-learning.ipynb

Different numerical scales: Solutions

- Scaling: puts numerical attributes onto same scale
  - Min-max scaling: shifts and rescales to range from 0 to1 -> $x_{norm}^{(i)} = \frac{x^{(i)} - x_{\min}}{x_{\max} - x_{\min}}$
    - Subtract min value and then divide by the max – min
    - Strength: Bounds values to a specific range
  - Standardization: ensures mean is zero and standard deviation is 1 -> $x_{std}^{(i)} = \frac{x^{(i)} - \mu_x}{\sigma_x}$
    - Subtract mean and then divide by the standard deviation
    - Strength: Less affected by outliers
- Algorithm
  1. Learn on training data
  2. Transform training data
  3. Transform test data

Which Scaling Solution When?

- Scaling: puts numerical attributes onto same scale
  - Min-max scaling: shifts and rescales to range from 0 to1
    - When bounded interval is needed
  - Standardization: ensures mean is zero and standard deviation is 1
    - When model weights are initialized to 0 or small values close to 0 (makes learning easier)
    - When the algorithm is sensitive to outliers

Group Discussion

- Why might datasets have incomplete data (i.e., missing values)? For example, think about examples in your daily lives where people collect information about you.

- Which of these algorithms are scale invariant (i.e., we do NOT need to worry about bringing features to the same scale)?
  - Linear regression
  - Decision trees
  - k-nearest neighbors

Today’s Topics

- Natural Language Processing
- Computer Vision
- Feature Representation
- Dimensionality Reduction
- Lab

Problems with High Dimensional Data?

- What are problems of having many features for machine learning?
  - Slower training
  - Slower testing
  - Can be harder to find a good solution, due to greater risk of overfitting
  - Requires lots of memory

Feature Selection Approaches

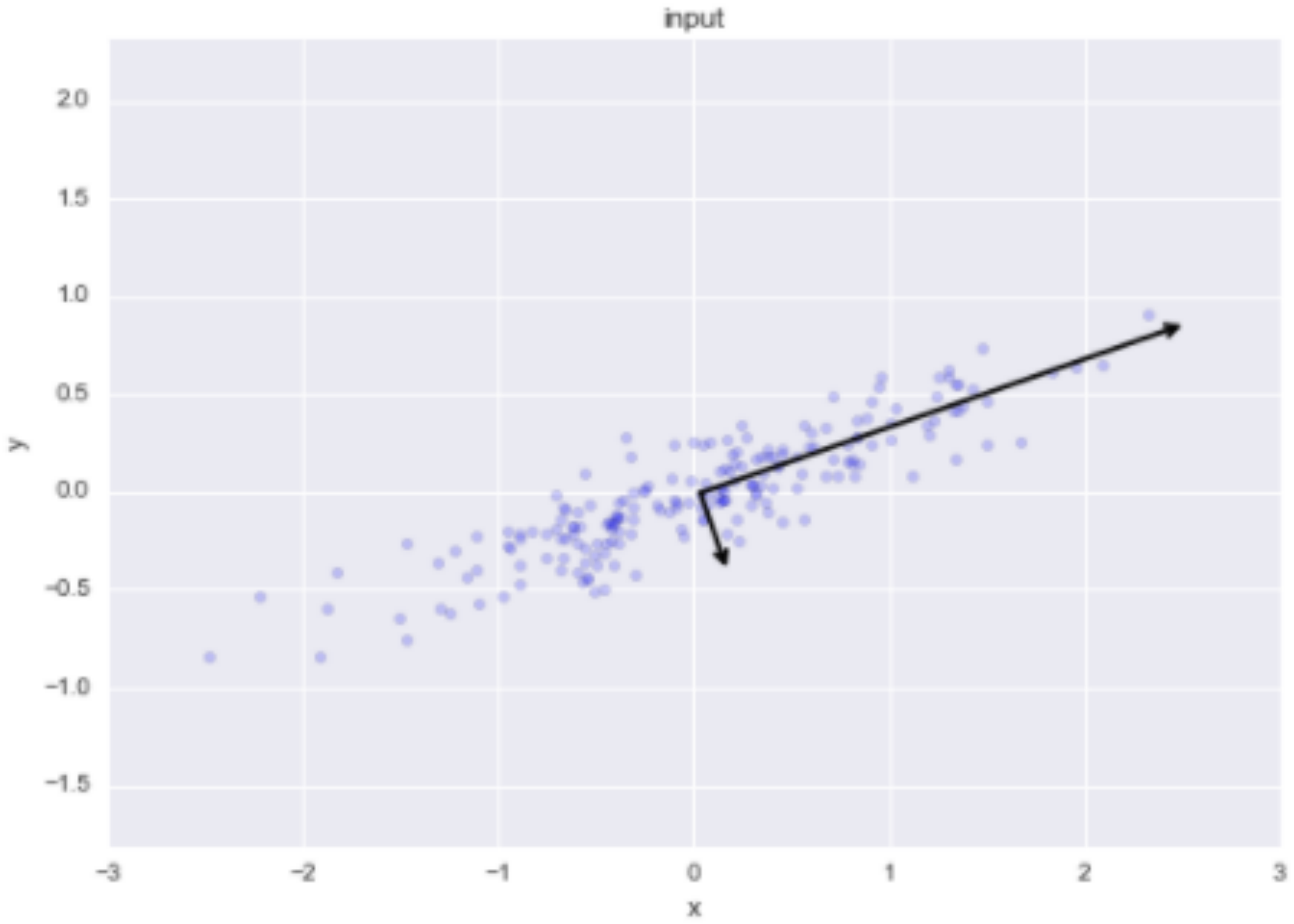
- Goal: remove irrelevant or redundant features
- Possible approaches?
  - Stepwise forward selection:
    - Iteratively add the feature among those remaining that leads to the greatest performance gain (greedy approach)
  - Stepwise backward elimination:
    - Iteratively remove the feature among those remaining that leads to the least performance loss (greedy approach)
  - Decision tree induction:
    - Use information gain when building decision trees; any features not included in the learned tree are deemed irrelevant

Projection Approaches

- Premise:
  - Many features are almost constant
  - Many features are highly correlated; e.g., age and height; degree and job title
- Idea:
  - Training instances actually lie within (or close to) a much lower-dimensional subspace of the high-dimensional space
- Approach:
  - Capture as much information about the features in fewer dimension(s)

Projection: PCA (Unsupervised)

- Idea: find principle axes and keep most important ones
- Vectors: *principal axes* of data
- Vector length: variance of the data described when its projected onto that axis.



https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html

Projection: PCA (Unsupervised)

- Assumption:
  - Data is linearly separable

- Algorithm
  1. Standardize data (i.e., center data around origin)
  2. Construct covariance matrix: how random variable pairs relate to each other

$$\text{Cov}\underbrace{\boxed{(X,Y)}}_{\text{Random variables}} = \Sigma E((X-\underbrace{\boxed{\mu}}_{\text{Mean of X}})(Y-\underbrace{\boxed{v}}_{\text{Mean of Y}})) \;/\; \underbrace{\boxed{n}}_{\text{\# items in dataset}}-1$$

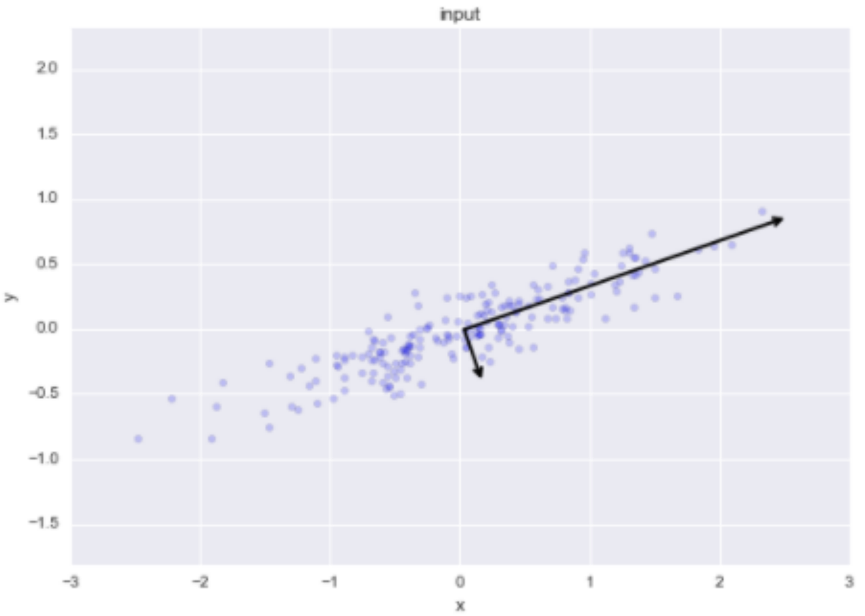
Positive when **large** values of X often occur with **large** values of Y; e.g., weight & height
Negative when **large** values of X often occur with **small** values of Y; e.g., grade and missed classes

Great tutorial: https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/covariance/

Projection: PCA (Unsupervised)

- Assumption:
  - Data is linearly separable

- Algorithm
  1. Standardize data (i.e., center data around origin)
  2. Construct covariance matrix
  3. Obtain eigenvalues and eigenvectors
     - Eigenvector: represents principal components (directions of maximum variance) of the covariance matrix
     - Eigenvalues: indicates corresponding magnitude of eigenvectors with larger values indicating direction of larger variance



https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html

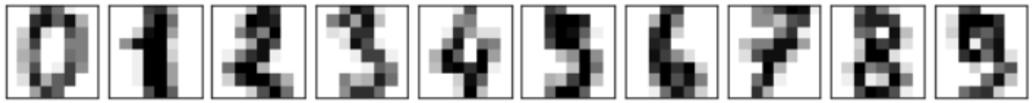
Projection: PCA (Unsupervised)

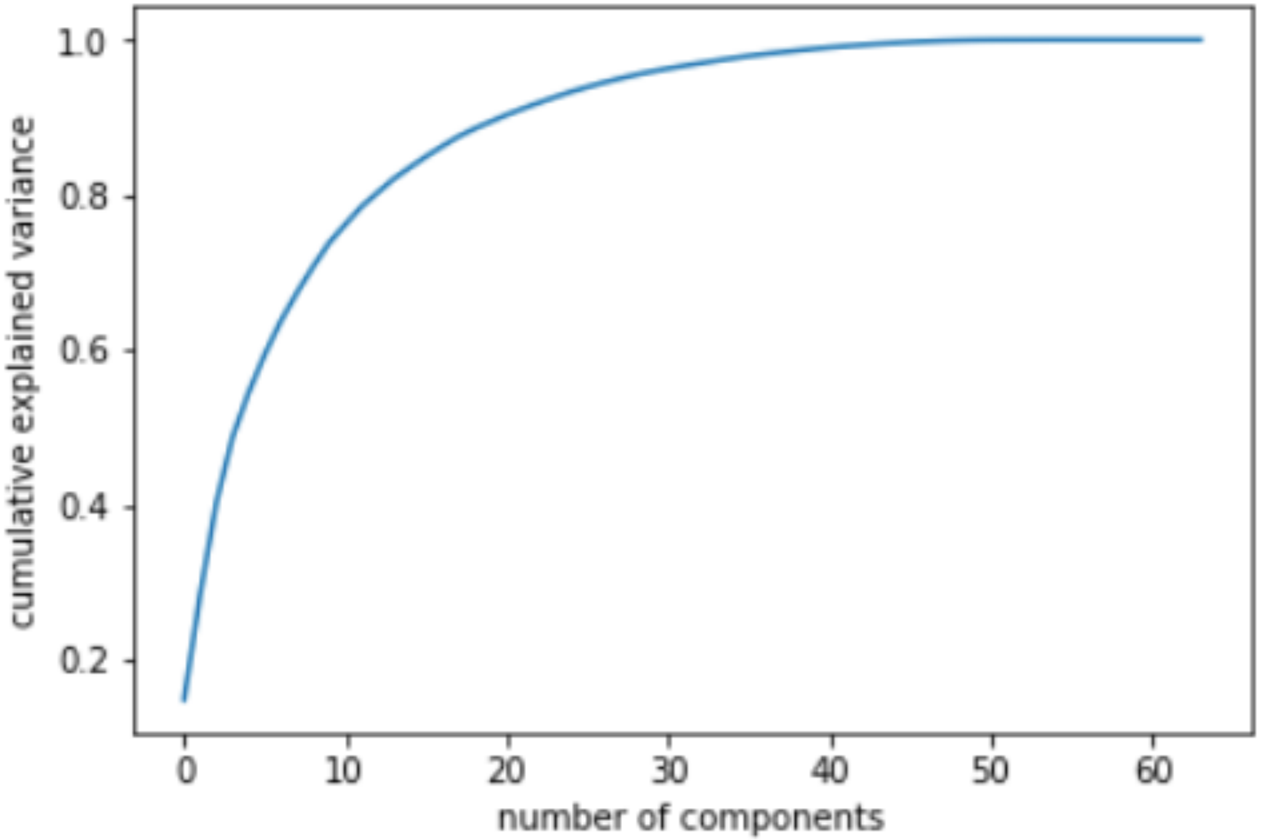
- Assumption:
  - Data is linearly separable

- Algorithm
  1. Standardize data (i.e., center data around origin)
  2. Construct covariance matrix
  3. Obtain eigenvalues and eigenvectors
  4. Sort eigenvalues by decreasing order to rank eigenvectors

Projection: PCA (Unsupervised)

e.g., data with 64 initial values



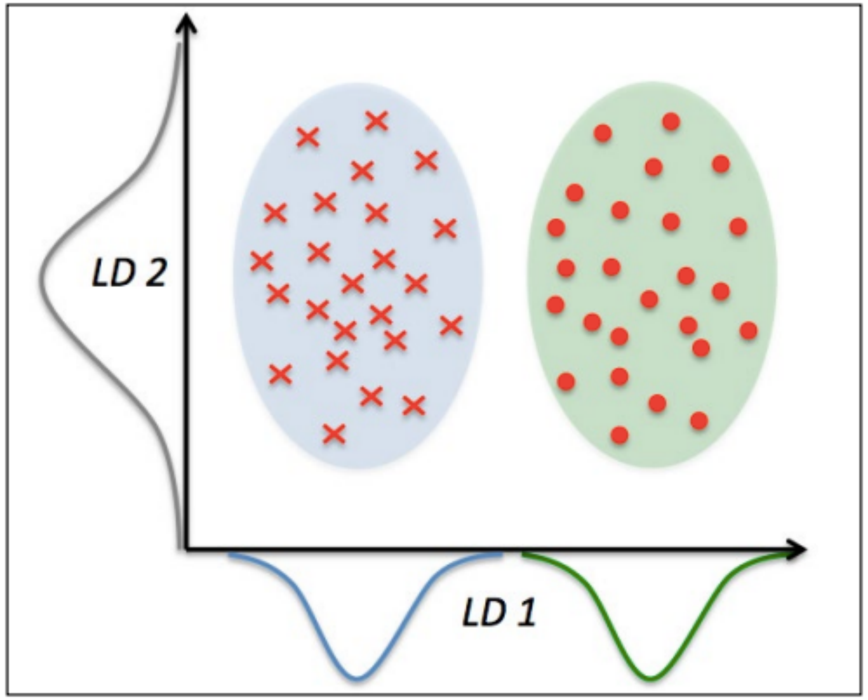


How many principal components are needed to preserve the information in the original data?

https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html

Projection: Linear Discriminant Analysis (Supervised); established 1936

- Assumptions:
  - Data is normally distributed
  - Data is linearly separable
  - e.g., x-axis would separate the two classes well
  - e.g., y-axis would not separate the two classes well
- Algorithm
  1. Standardize d-dimensional dataset
  2. For each class, compute d-dimensional mean vector
  3. Construct between-class scatter matrix and the within-class scatter matrix
  4. Compute eigenvectors and corresponding eigenvalues
  5. Sort eigenvalues by decreasing the order to rank the corresponding eigenvectors
  6. Choose k eigenvectors that correspond to the k largest eigenvalues
  7. Project samples onto the new feature space



Python Machine Learning; Raschkka & Mirjalili

Projection: Manifolds

- Manifold intuition:
  - e.g., Imagine a sheet of paper which is a 2-d object/manifold living/embedded in a 3-d world/space
  - Rotating, bending, or crumpling the paper does not change that it is 2d but it does mean that the embedding in 3d space is no longer linear
  - Algorithms seek to learn about the fundamental 2d nature of the paper even as it is contorted to fill the 3d space
- Algorithms:
  - Model the *manifold* on which the training instances lie; i.e., make an assumption or manifold hypothesis that most real-world high-dimensional datasets lie close to a much lower-dimensional manifold
  - e.g., Locally Linear Embedding

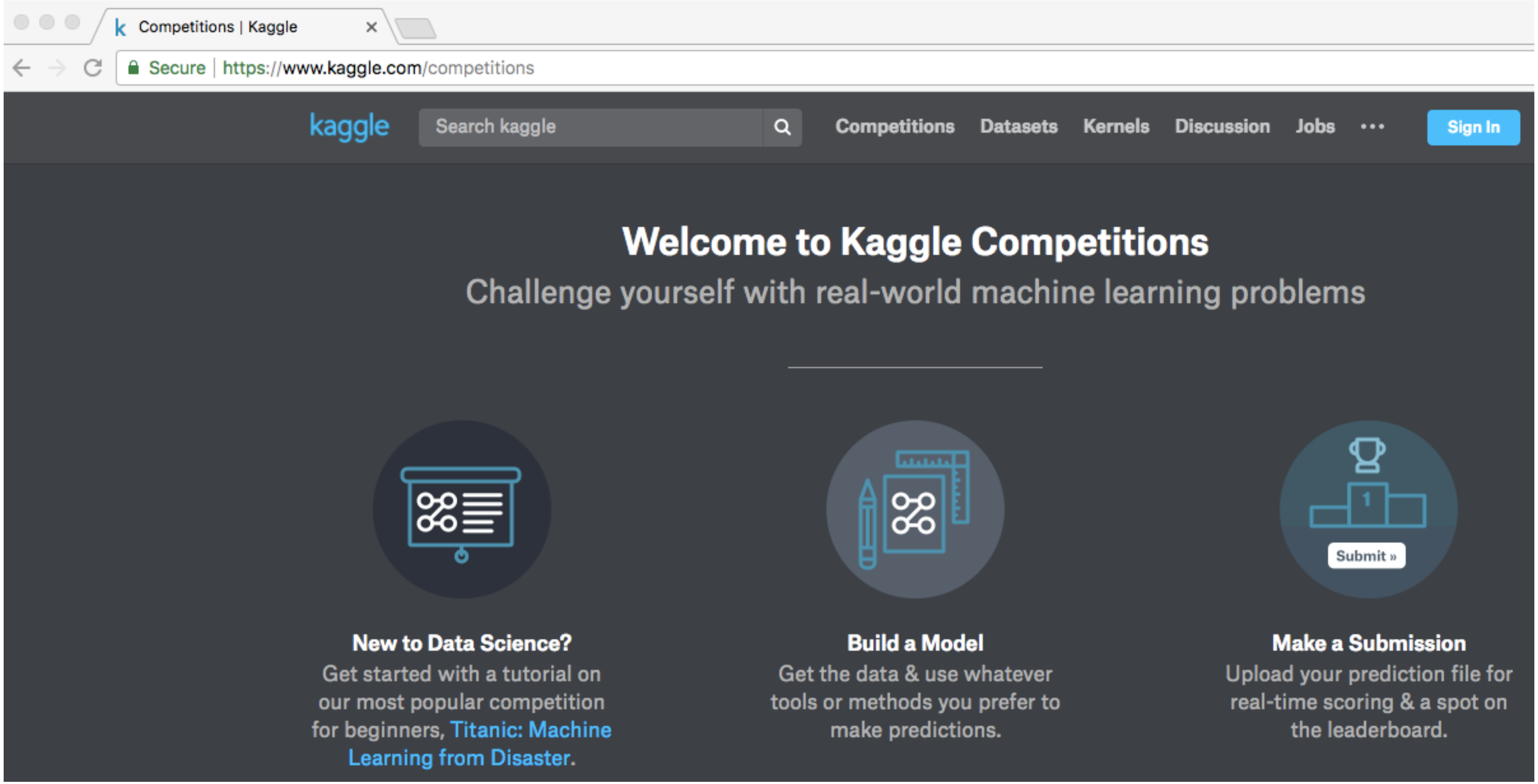
Why Use Data Reduction?

- Can lead to improved machine learning algorithm performance
- Visualization
- Data compression
- Noise removal

Today’s Topics

- Natural Language Processing
- Computer Vision
- Feature Representation
- Dimensionality Reduction
- Lab

Kaggle: Large-Scale Datasets + World-Wide Challenges Inspire Technological Innovation

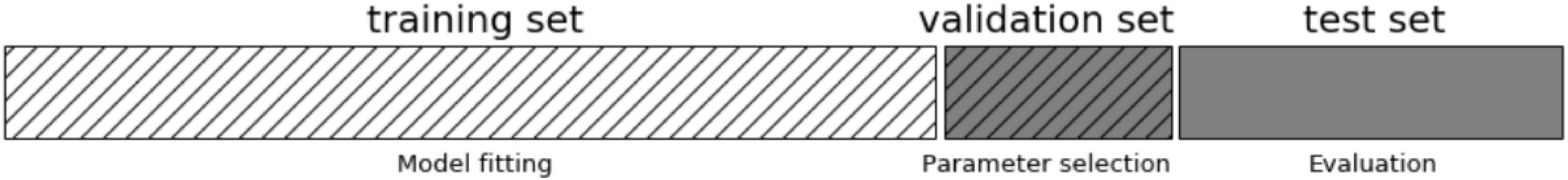


What Challenges Often Have in Common:

1. Publicly-shared train (and validation) dataset with “ground truth” labels

2. Publicly-shared test dataset (“ground truth” labels are hidden)

3. Metrics for evaluating algorithm-generated results on the test set



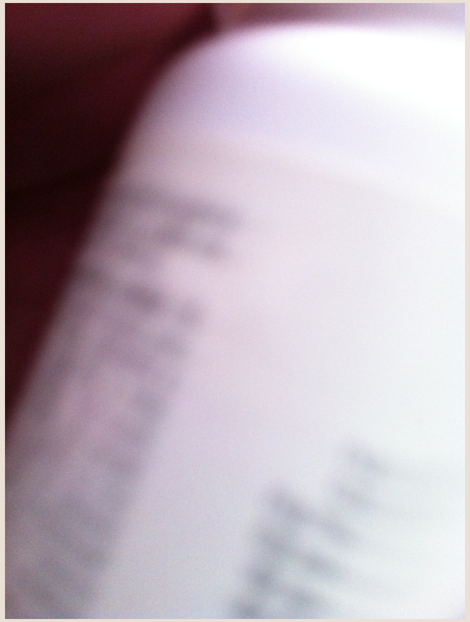
Why Have Challenges?

- Provide “fair” comparison between algorithms
- Create a community around a shared goal

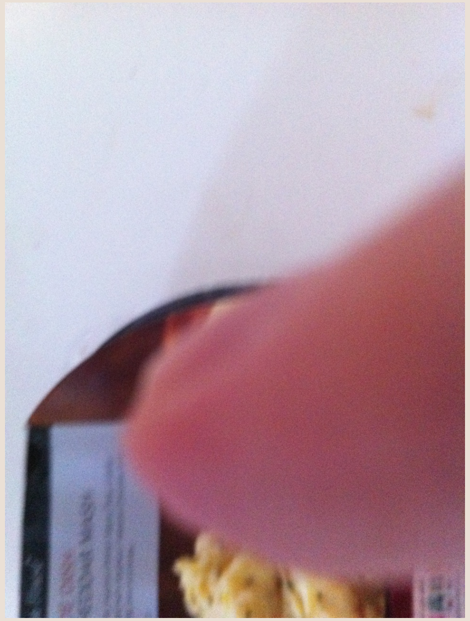
Task: Answer Blind People's Visual Questions

Is this shirt clean or dirty?

Hi there can you please tell me what flavor this is?

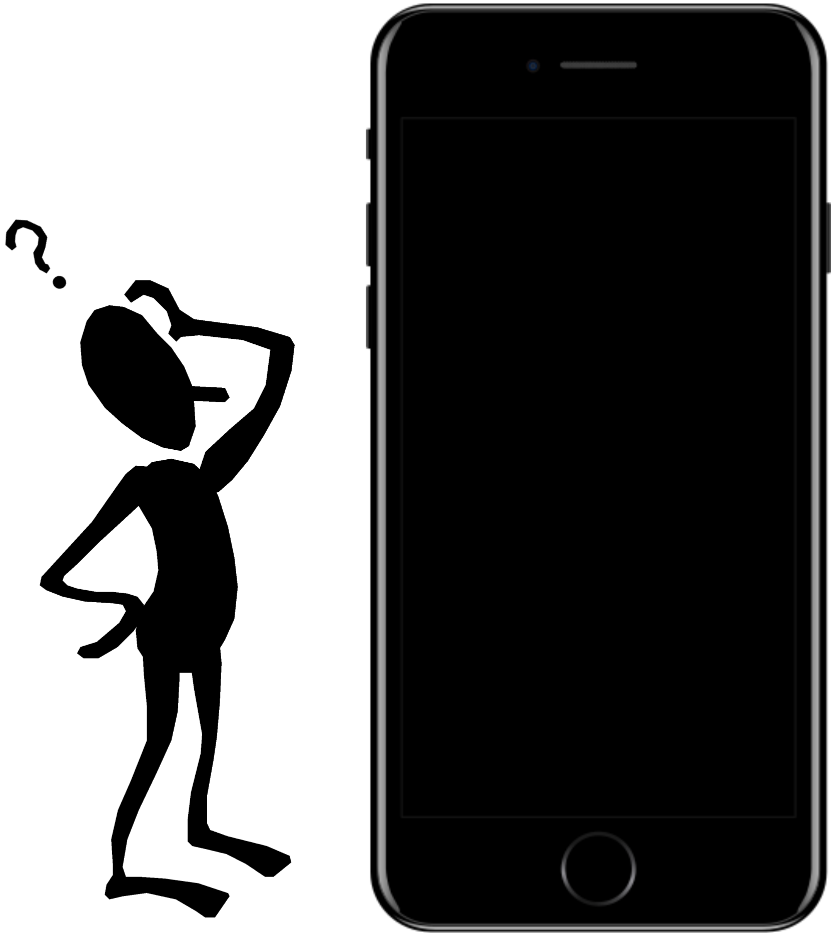


What type of pills are these?



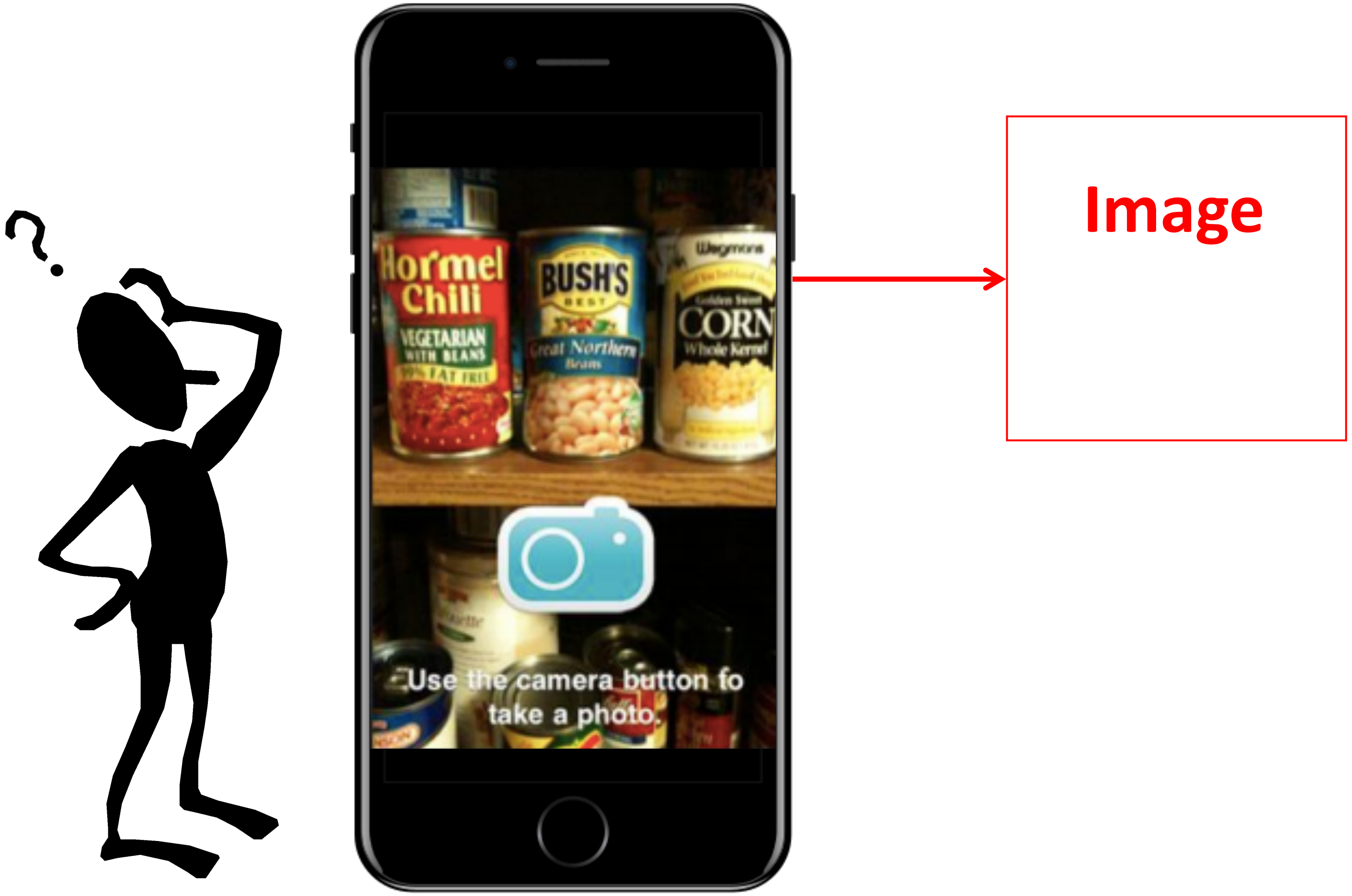
What is this?

Motivation: VizWiz Mobile Phone Application



Jeffrey Bigham et al. VizWiz: Nearly Real-time Answers to Visual Questions. UIST 2010

Motivation: VizWiz Mobile Phone Application

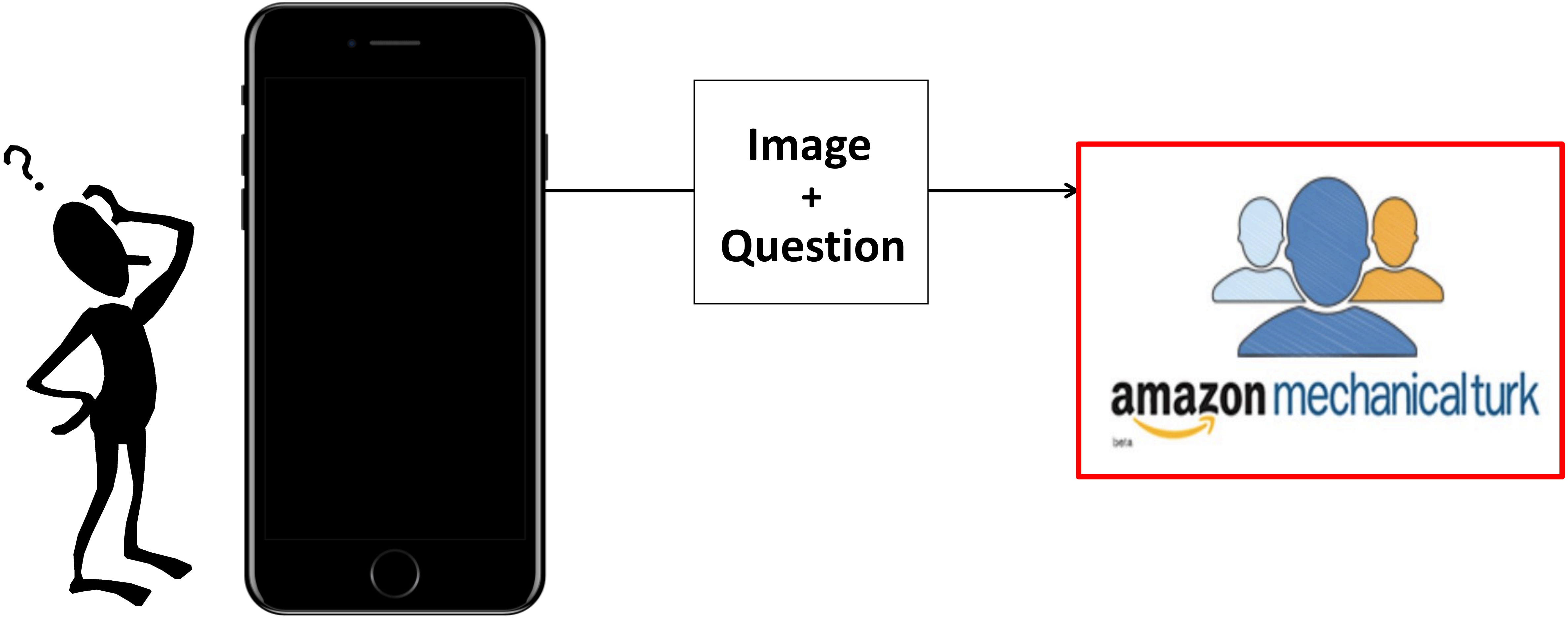


Jeffrey Bigham et al. VizWiz: Nearly Real-time Answers to Visual Questions. UIST 2010

Motivation: VizWiz Mobile Phone Application

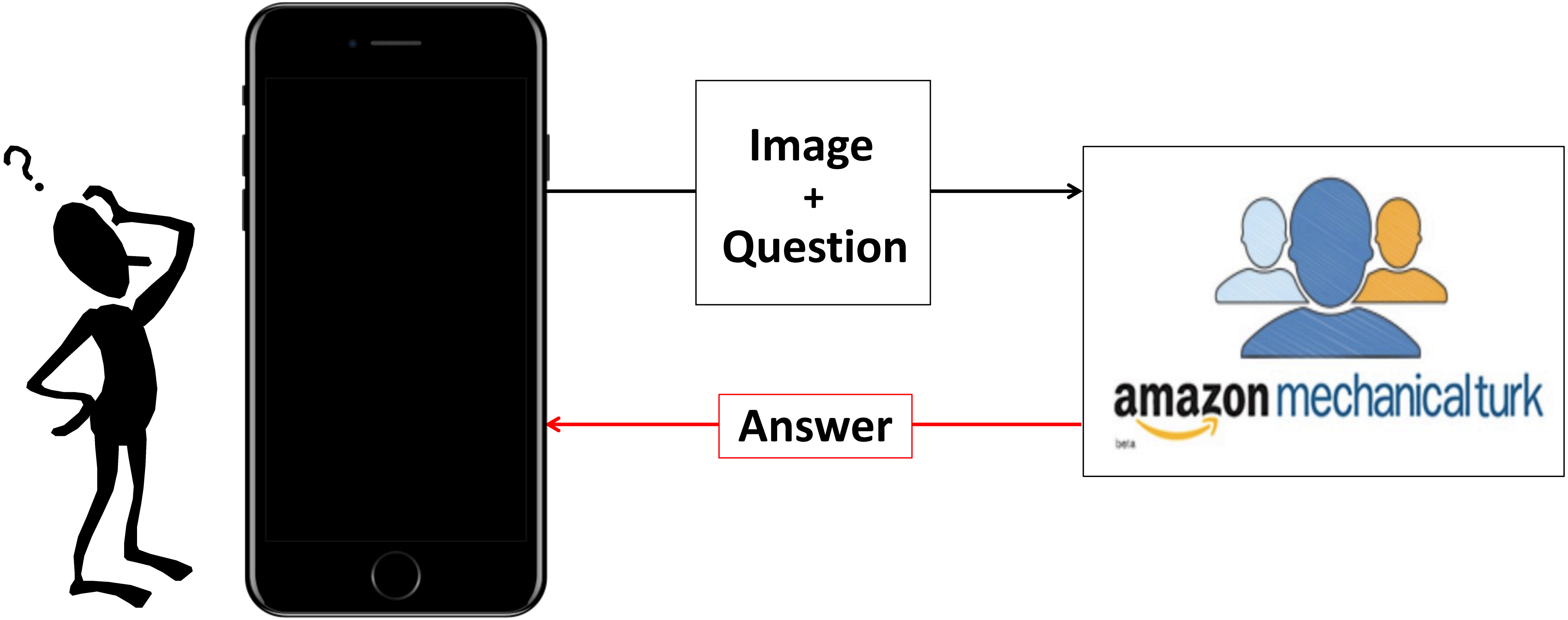
Jeffrey Bigham et al. VizWiz: Nearly Real-time Answers to Visual Questions. UIST 2010

Motivation: VizWiz Mobile Phone Application



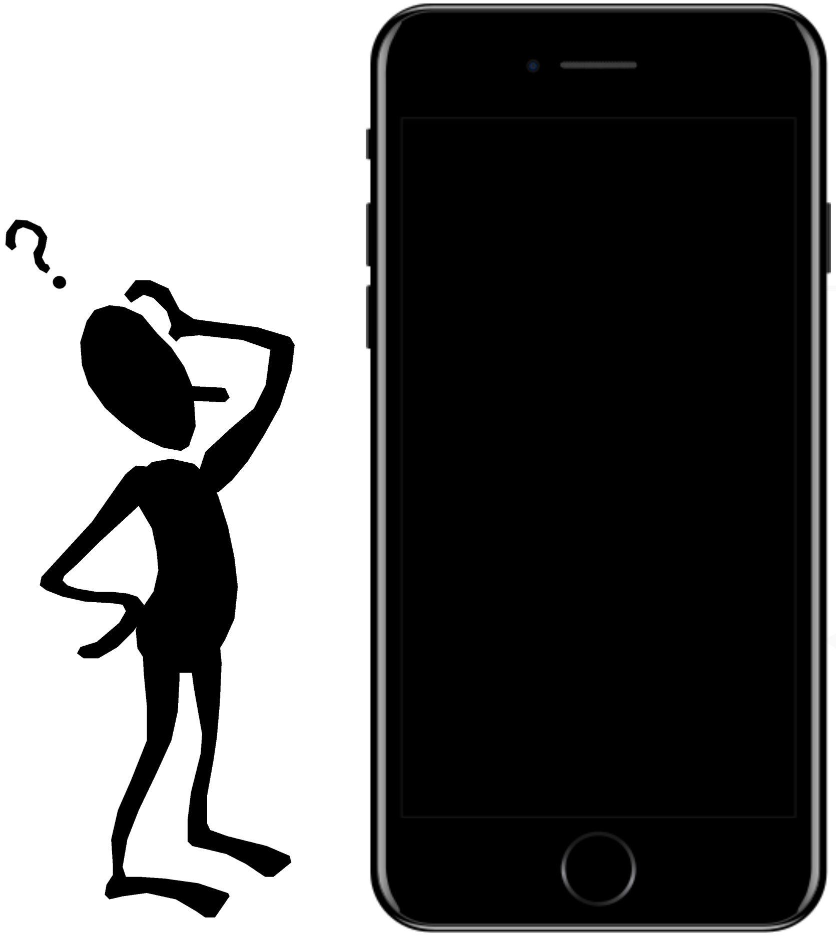
Jeffrey Bigham et al. VizWiz: Nearly Real-time Answers to Visual Questions. UIST 2010

Motivation: VizWiz Mobile Phone Application



Jeffrey Bigham et al. VizWiz: Nearly Real-time Answers to Visual Questions. UIST 2010

Motivation: VizWiz Mobile Phone Application



11,045 people asked 72,205 visual questions between 2011 and 2015

Jeffrey Bigham et al. VizWiz: Nearly Real-time Answers to Visual Questions. UIST 2010

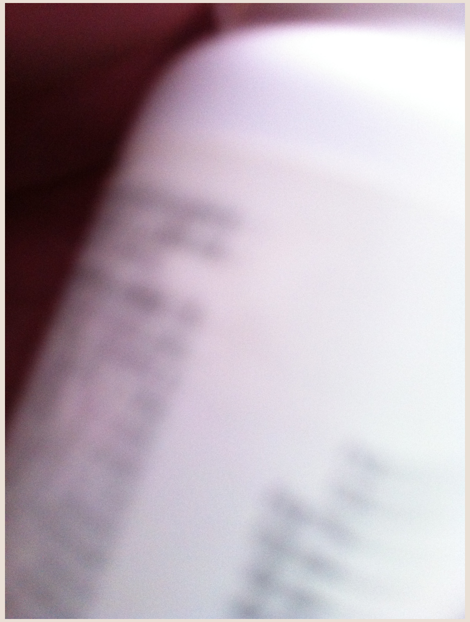
Your Lab Assignment Task: Predict from Visual Question Whether It Can Be Answered

Is this shirt clean or dirty?

answerable

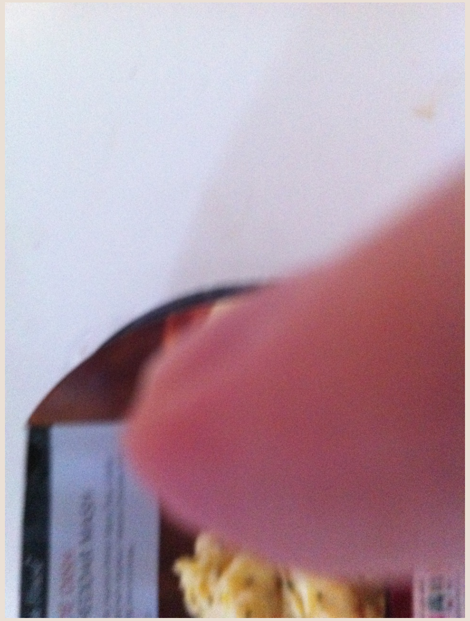
Hi there can you please tell me what flavor this is?

answerable



What type of pills are these?

unanswerable



What is this?

unanswerable