TEXAS
The University of Texas at Austin

I320D – Topics in Human Centered Data Science
Text Mining and NLP Essentials

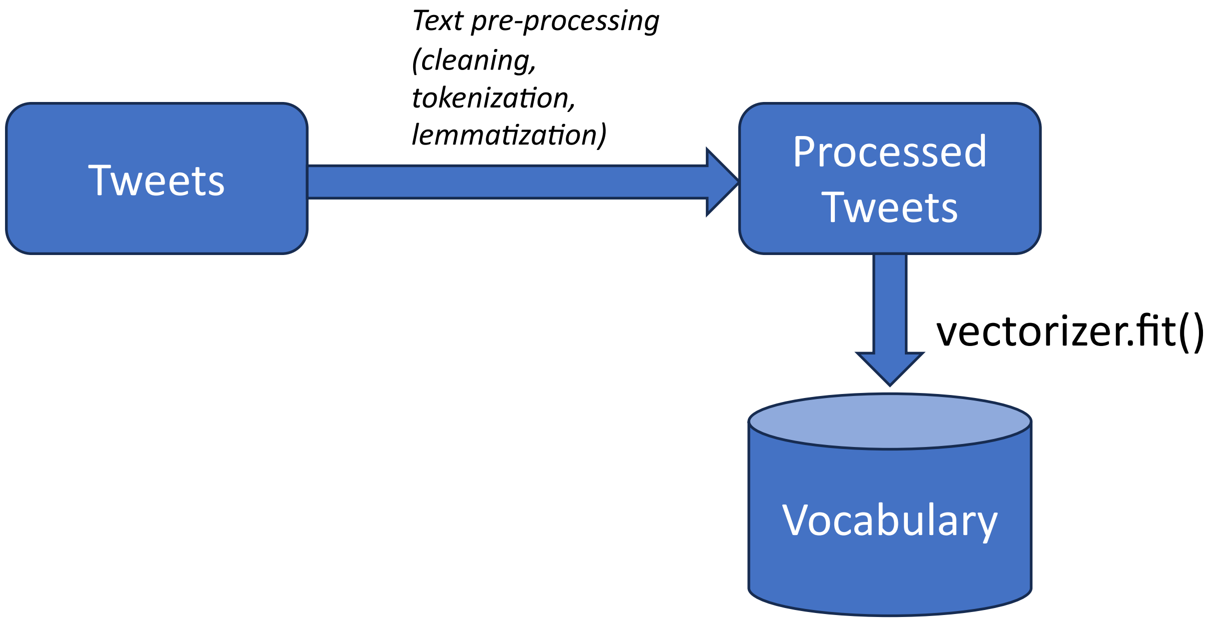
Week 7: Machine Learning for NLP -1

Dr. Abhijit Mishra

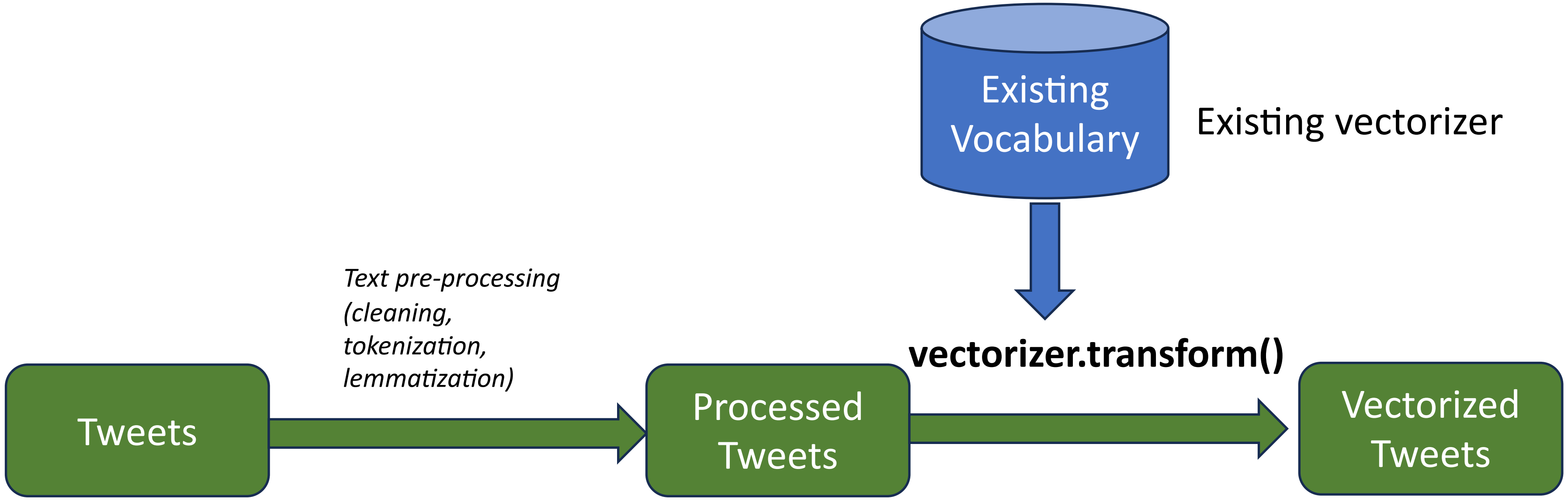
Before we start …
(Ongoing and upcoming assignments)

- Ongoing assignment :
  - Hashtag based tweet search (Deadline today, 02/27)
- Upcoming assignment:
  - Character assessment from stories
  - To be posted today (deadline 03/08/2024)
- Group Project formation:
  - Group size: Max 4, Min 3
  - Proposals to be solicited immediately after Spring break
  - Fill out the form here: https://forms.gle/H9akcB9PGLNEmmUU8

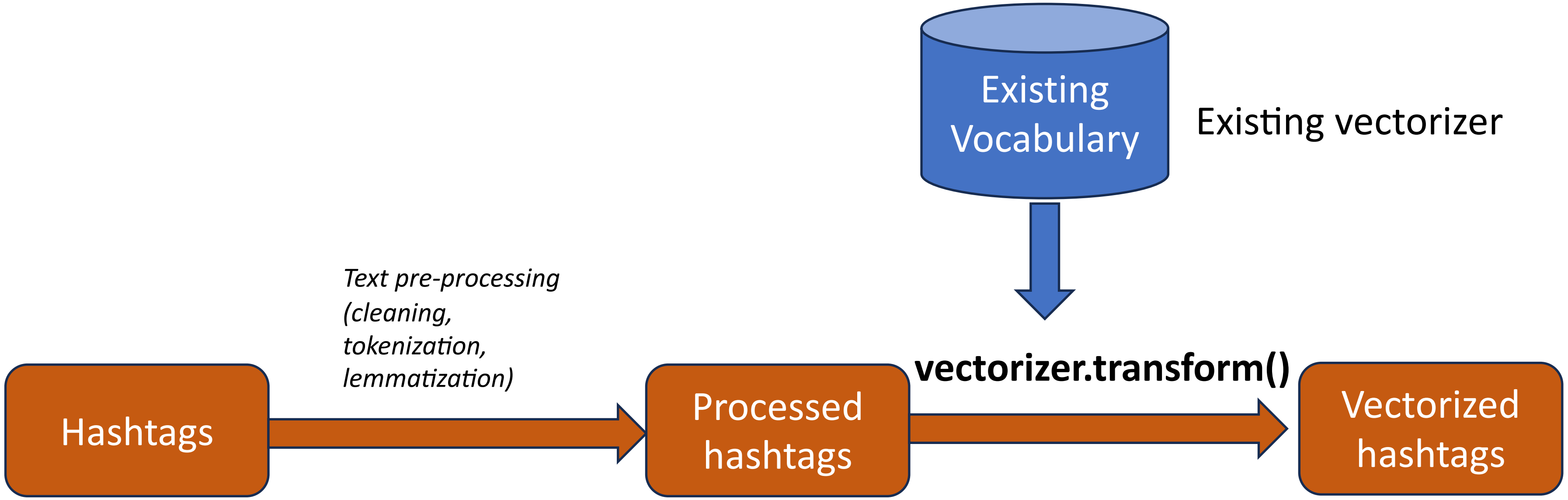
Before we start …
(Reg. Assignment 3)

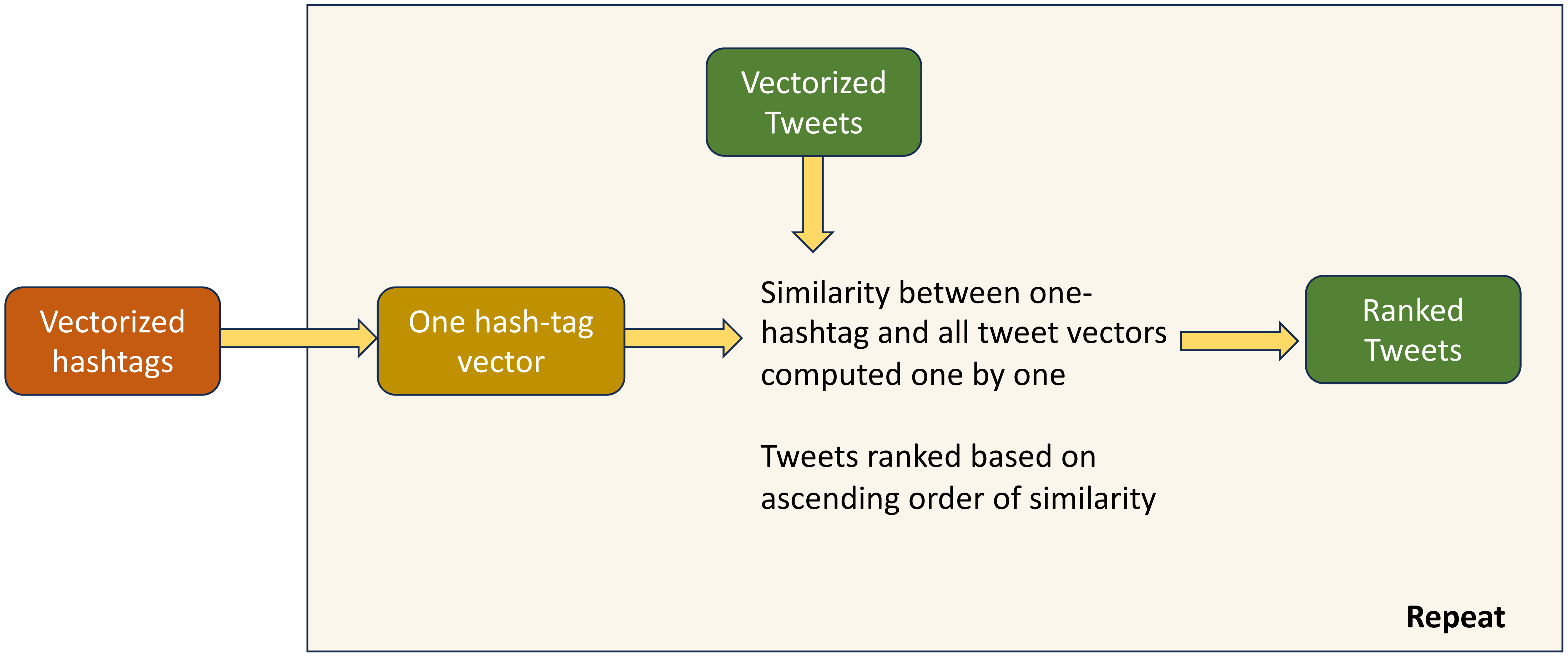


Before we start …
(Reg. Assignment 3)



Before we start …
(Reg. Assignment 3)




Vectorized hashtags
One hash-tag vector
Vectorized Tweets
Similarity between one-hashtag and all tweet vectors computed one by one
Tweets ranked based on ascending order of similarity
Ranked Tweets
Repeat

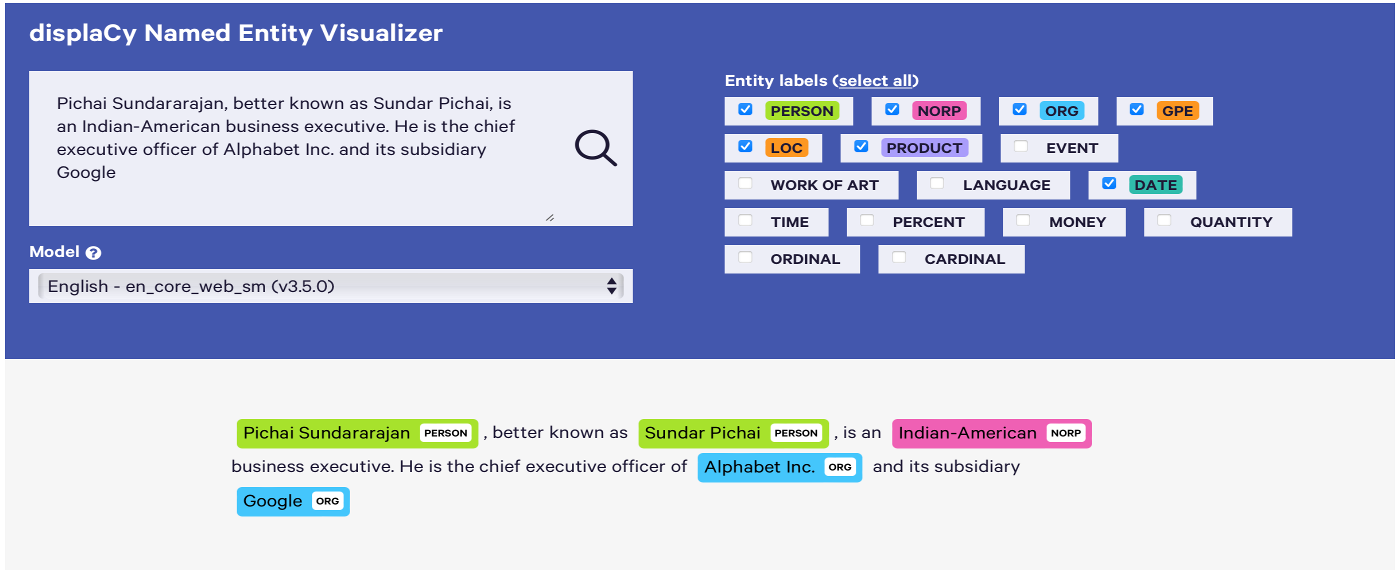
So far in I320D – Text Mining and NLP

- W1. Language and Ambiguity
- W2. Basics of Text Data and Linguistic Concepts
- W3. Text Preprocessing Techniques
- W4. Lexical Analysis
- W5. Syntax Analysis
- W6. Information Extraction

W7. Machine Learning Methods for NLP
W8. Unsupervised ML and Topic Modeling Basics
W10-W11. Deep learning for NLP
W12. NLP Applications
W13. Small and Large Language Models and Prompt Engineering Basics
W14. Knowledge Networks
W15. Evaluation Metrics

Recap: Shallow Parsing Tasks

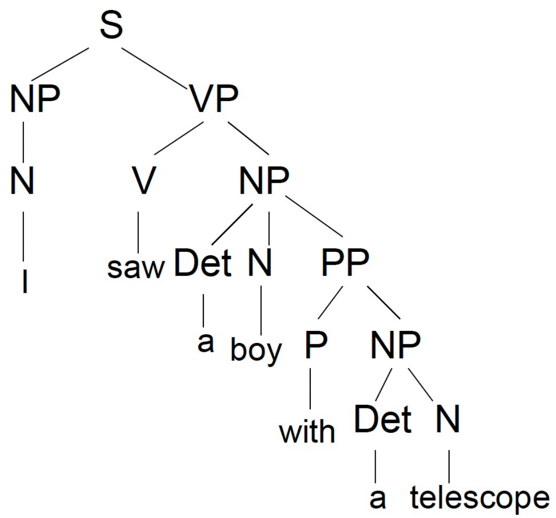
- Part of Speech Tagging
- Noun Phrases / Verb Phrases Chunking
- Named Entity Identification

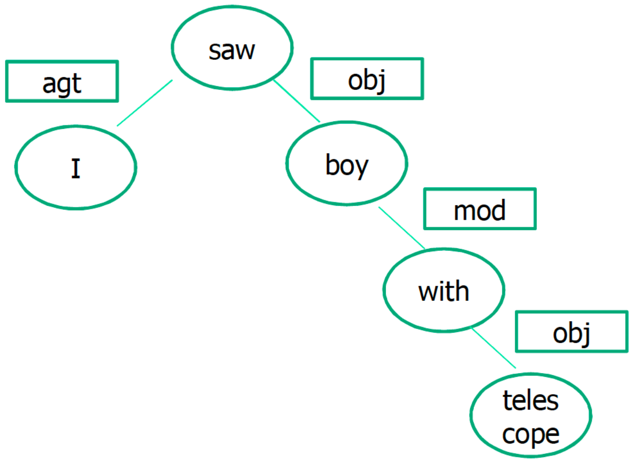


Recap: Deep Parsing Tasks

- Constituency Parsing

(grammar centric parsing)





- Dependency Parsing

(grammar+meaning centric parsing)

Week 7: Roadmap

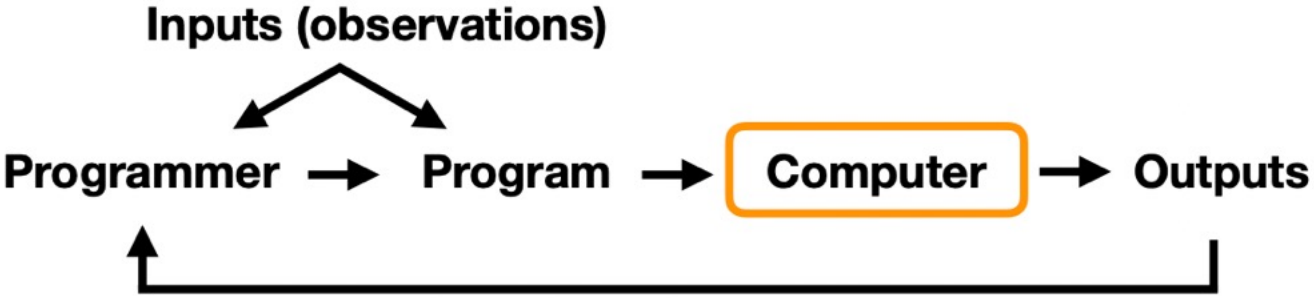
- Machine Learning for NLP
  - What is Machine Learning?
  - NLP tasks that require NLP
  - Text Classification and Sequence Tagging / Labeling tasks

What is Machine Learning?

The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience.

–Tom Mitchell, Professor, CMU and Popular ML author

The Traditional Programming Paradigm:



Requirement

x	y	Z
2	1	2
1	1	1
2	3	6

Human programmer understands and writes multiplication code

Machine Learning

Improves with experience

x	y	Z
2	1	2
1	1	1
2	3	6

Learned Program :May be just copy Col 1?

Learned Program : Definitely multiplication

Source:
https://sebastianraschka.com/pdf/lecture-notes/stat451fs20/01-ml-overview__notes.pdf

Formal Definition

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

–**Tom Mitchell, Professor, CMU**

Types of Learners

Supervised Learning

- Labeled data
- Direct feedback
- Predict outcome/future

Unsupervised Learning

- No labels/targets
- No feedback
- Find hidden structure in data

Reinforcement Learning

- Decision process
- Reward system
- Learn series of actions

Supervised Learning

- Learning from “Labeled training data”
- Labeled data = bunch of examples {Input, Expected Output}
- Classification or Regression depending on the outcome type
  - **Classification**: When the expected output type is categorical
  - **Regression**: When the expected output type is numeric (real number)
- In text world:
  - **Text classification –** Classify given text (snippets, sentences, documents) into predefined set of categories
    - **Sequence labeling –** Tagging of tokens
    - Sequence generation – Classifying inputs into one
  - **Regression** – Predicting a numeric output from textual input

Exercise

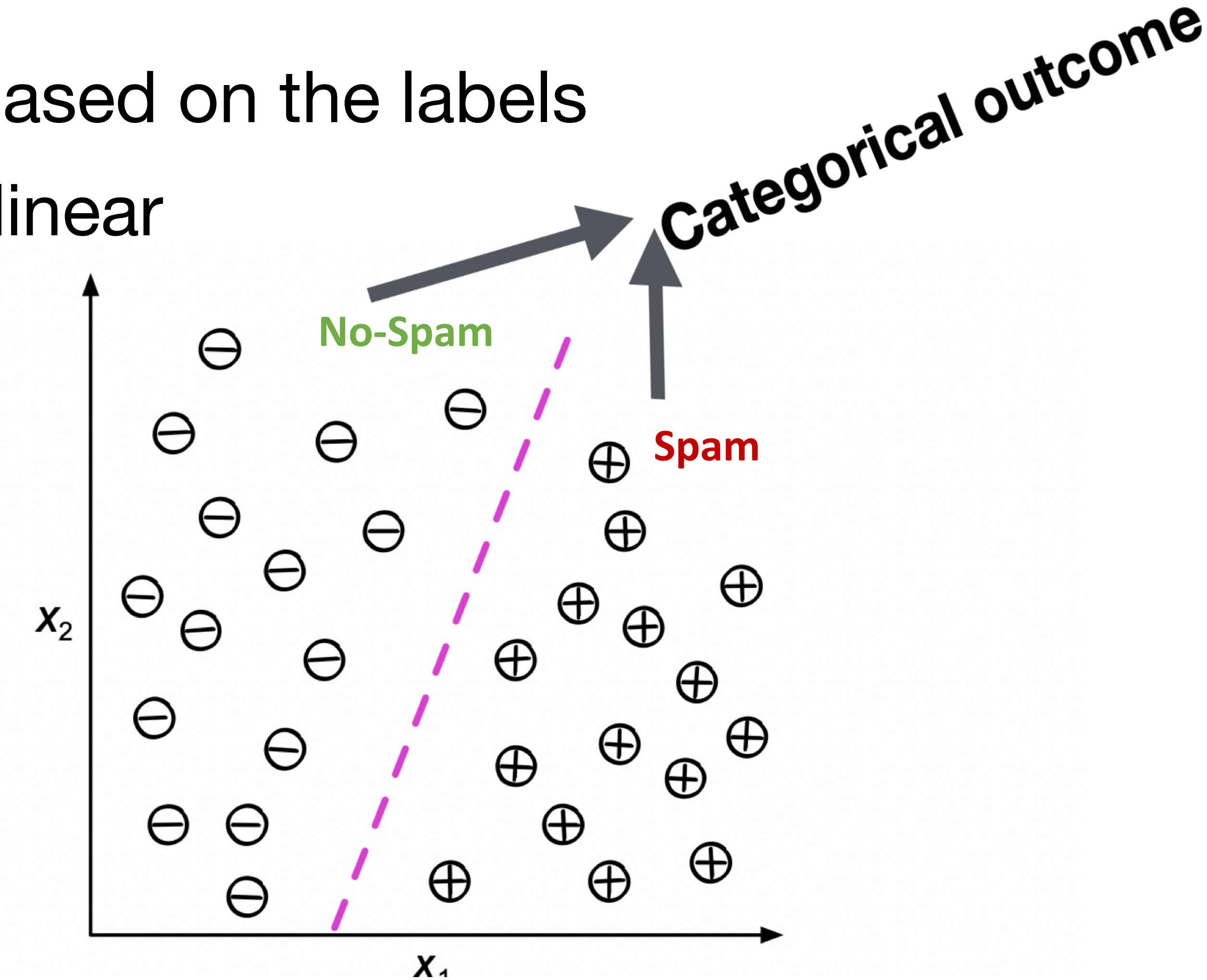
- Give an example of text classification
- What is the source of experience (i.e., Dataset)? How do you collect them?
- What is your performance measure P?

Exercise

- Give an example of text regression?
- What is the source of experience (i.e., Dataset)? How do you collect them?
- What is your performance measure P?

Supervised Learning - Classification

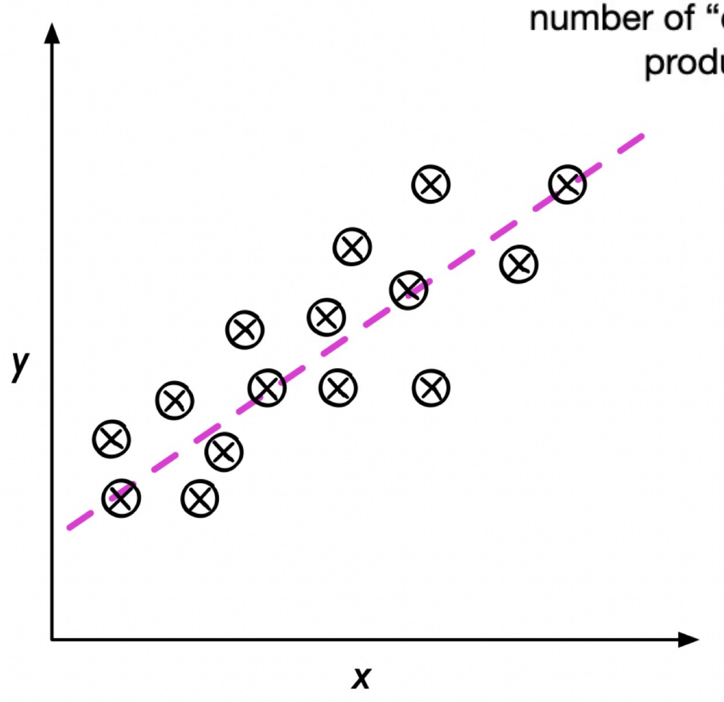
- Learn to separate input data based on the labels
- Separator may or may not be linear



Source:
https://sebastianraschka.com/pdf/lecture-notes/stat451fs20/01-ml-overview__notes.pdf

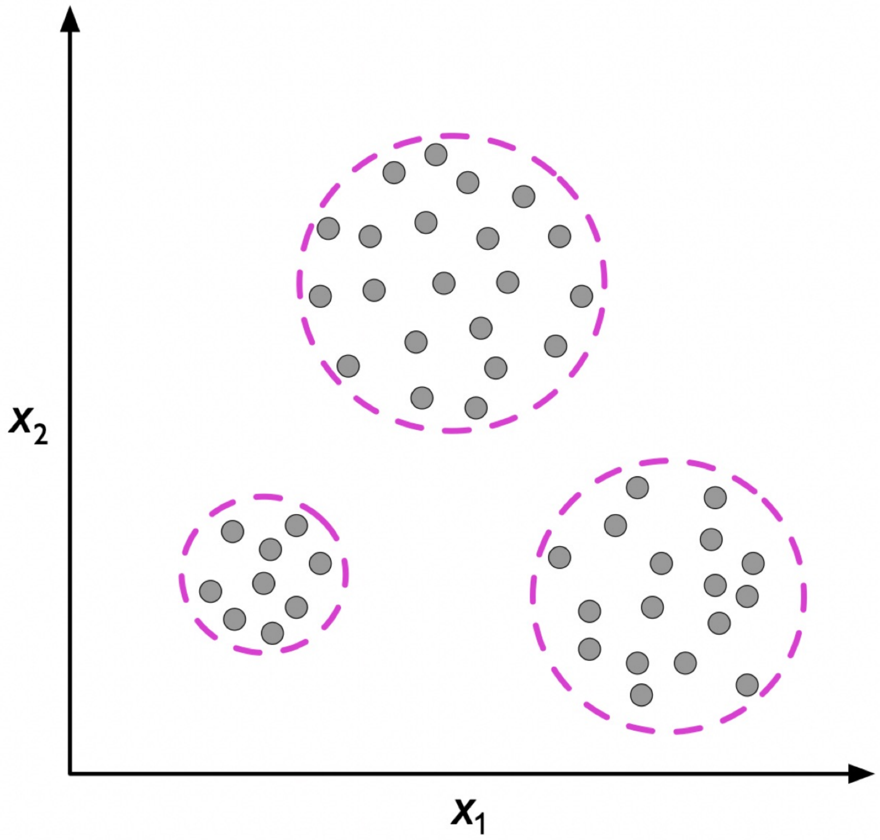
Supervised Learning - Regression

- Learn a “trend” (function of input)
- Output is real valued
- May or may not be linear



Unsupervised Learning

- No label present
- Learn to divide data into groups/clusters just from the input
- Example:
  - Grouping students based on <attendance, class-performance, homework completion rate>

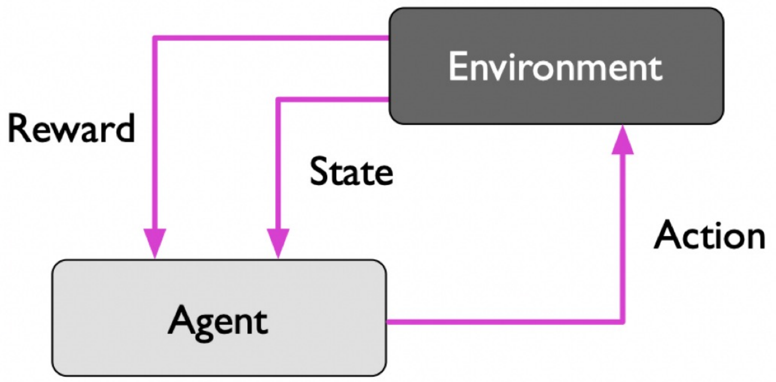


Exercise

- Give an example of unsupervised machine learning for text?
- What is the source of experience (i.e., Dataset)? How do you collect them?
- What is your performance measure P?

Reinforcement Learning

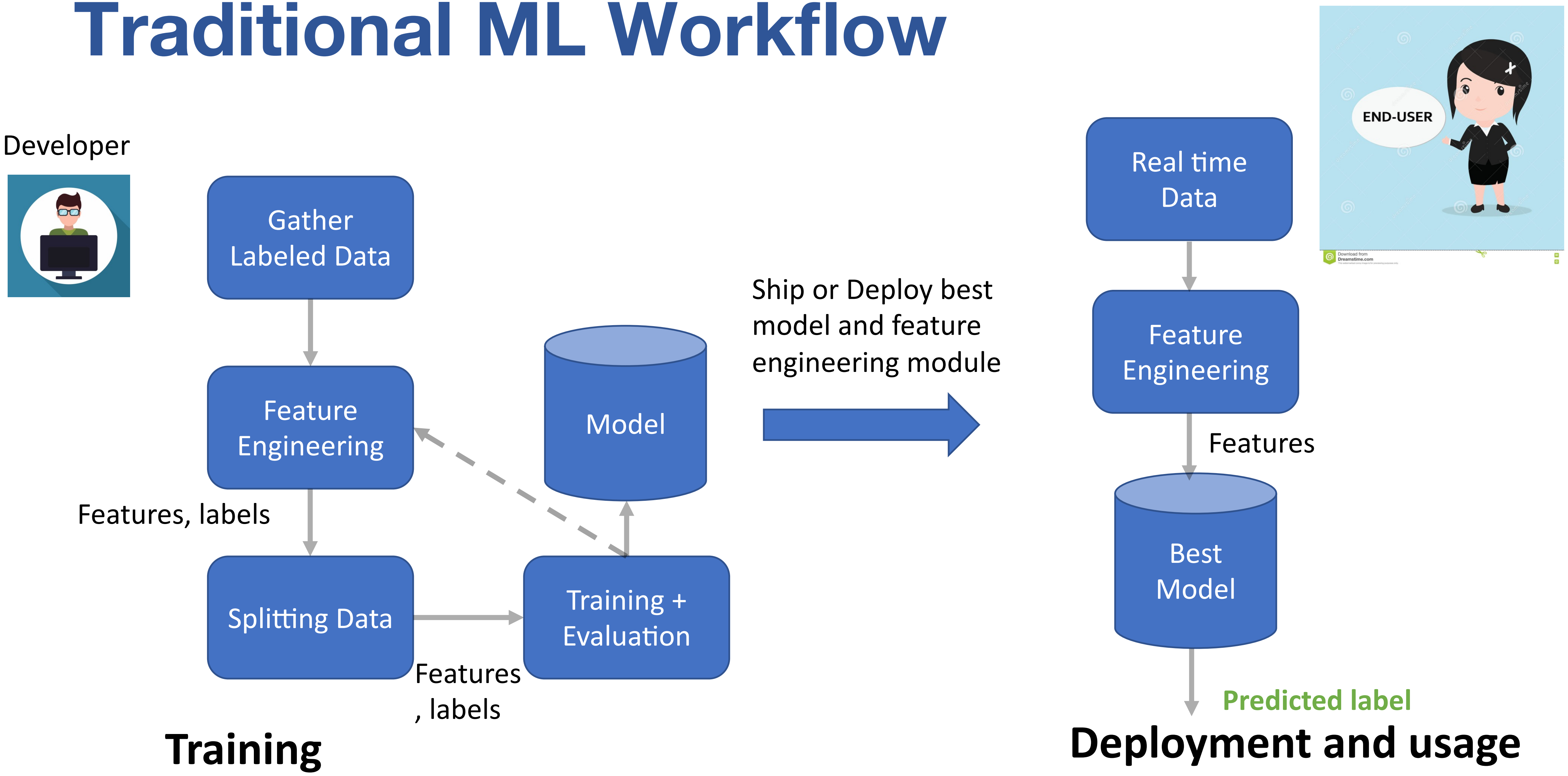
- Like supervised learning but in this case the exact outcome is not available during training.
- Instead, an indicator (or a reward scoring mechanism) shows how good or bad action is towards achieving the outcome
  - Example:
    - Playing chess
    - Automatic car parking



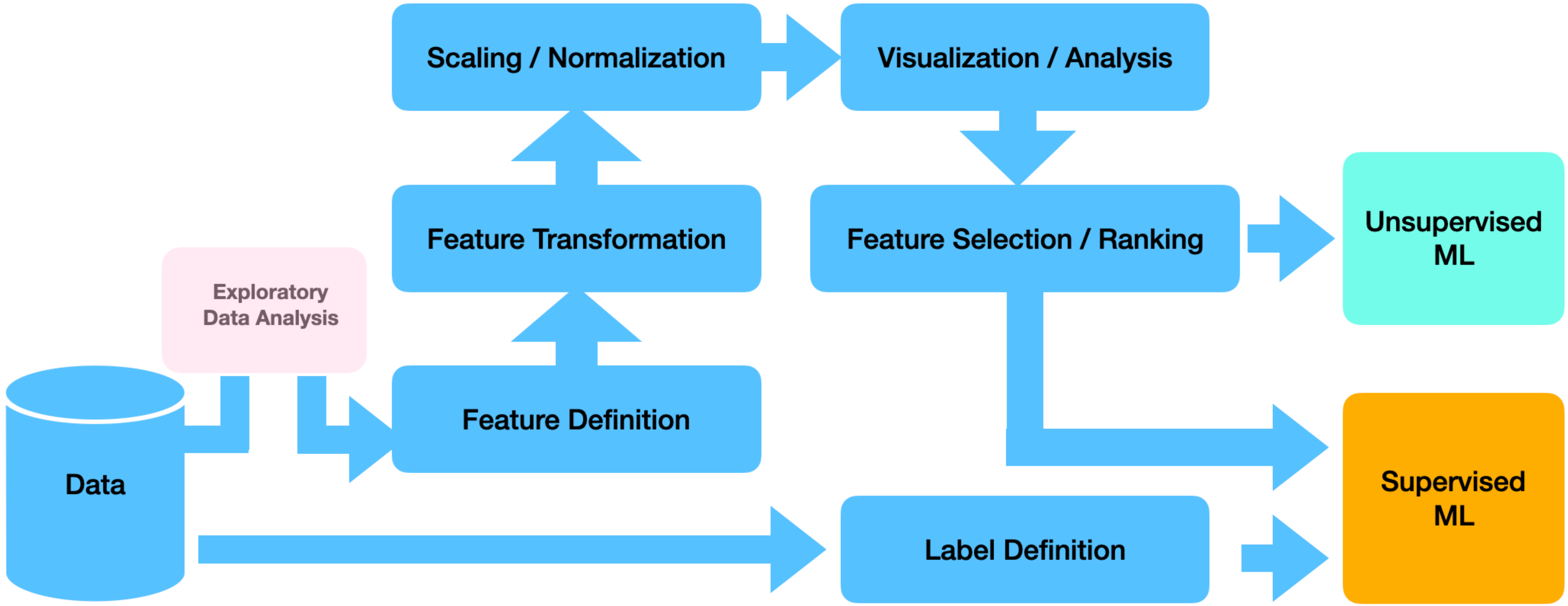
Exercise

- Give an example of reinforcement learning with text?
- What is the source of experience (i.e., Dataset)? How do you collect them?
- What is your performance measure P?

Traditional ML Workflow



Data Processing for Traditional ML



Consider a hypothetical data

- “Credit card spending habit” example
- Task: Predict the usage based on personal information

Index	City	Number of Cards Owned	Card type	Gender	Annual Income	Salaried	Usage
1	LA	1	Silver	F	250000	Yes	High
2	New York	1	Gold	M	220000	Yes	Low
3	LA	3	Platinum	F	455000	No	High
4	Chicago	1	Platinum	M	88000	No	Low
5	SF	1	Gold	F	295000	Yes	Low

- *Which columns are useful as input to an ML model?*
- *Which column can be considered as the label for Task?*
- *Can we any extra column based on our intuition?*

Feature Engineering

- Defining what should be the inputs to your program based on "domain knowledge"
  - Consider subset of columns
  - And / or define a set of new columns by combining the existing columns
- Convert all features into numeric form (i.e., floats / real numbers)

Feature Engineering

- Writing programs / implementing functions to transform every data type into float

```
def card_type_transformed(card_type):
    if card_type == "silver":
        return 1.0
    elif card_type == "gold":
        return 2.0
    elif card_type == "platinum":
        return 3.0
    else
        return 0.0
```

```
def income_category(annual_income, salaried):
    if annual_income > 100000 and salaried:
        return 1 #"high_salaried"
    elif annual_income > 100000 and not salaried:
        return 2 #"low_salaried"
    else:
        return 3 #"non_salaried"
```

Existing column

OR

Create an entirely new column

Feature Engineering

- Numeric feature: Use **as-is**
- String / Categorical Features:
  - Represent categories (e.g., academic_year)
  - Can be:
    - **Nominal**, i.e., not related to each other (e.g., city, country, university name)
    - **Ordinal**, i.e., certain order found between them (e.g., academic_year, letter grade)
- *Exercise: Example of nominal feature*
- *Exercise: Example of ordinal feature?*
- *Exercise: What kind of feature is "Month"?*

Feature Engineering

Index	City	Number of Cards Owned	Card type	Gender	Annual Income	Salaried	Usage
1	LA	1	Silver	F	250000	Yes	High
2	New York	1	Gold	M	220000	Yes	Low
3	LA	3	Platinum	F	455000	No	High
4	Chicago	1	Platinum	M	88000	No	Low
5	SF	1	Gold	F	295000	Yes	Low

# Cards	Annual Incor	Salaried	Income Category_High	Income Category_Low	Card Type_Gold	Card Type_Platinum	Card Type_Silver	Gender_F	Gender_M	City_Chicago	City_LA	City_NY	City_SF	Usage_High	Usage_Low
1	250000	1	1	0	0	0	1	1	0	0	1	0	0	1	0
1	220000	1	1	0	1	0	0	0	1	0	0	1	0	0	1
3	455000	0	1	0	0	1	0	1	0	0	1	0	0	1	0
1	88000	0	0	1	0	1	0	0	1	1	0	0	0	0	1
1	295000	1	1	0	1	0	0	1	0	0	0	0	1	0	1

Given features

Additional Features

Given features

Labels

Feature Engineering

Transforming data into informative feature with “domain knowledge”

# Cards	Annual Incoı	Salaried	Income Category_High	Income Category_Low	Card Type_Gold	Card Type_Platinum	Card Type_Silver	Gender_F	Gender_M	City_Chicago	City_LA	City_NY	City_SF	Usage_High	Usage_Low
1	250000	1	1	0	0	0	1	1	0	0	1	0	0	1	0
1	220000	1	1	0	1	0	0	0	1	0	0	1	0	0	1
3	455000	0	1	0	0	1	0	1	0	0	1	0	0	1	0
1	88000	0	0	1	0	1	0	0	1	1	0	0	0	0	1
1	295000	1	1	0	1	0	0	1	0	0	0	0	1	0	1

$$X = \{x_1, x_2, x_3, \dots, x_N\} \text{ where } x_i \in \mathbb{R}$$

$$y_i^{actual} \in \mathbb{R}^2$$

$$\begin{pmatrix} 250000.0 & 1.0 & \dots & 1 & 0 & 0 \\ 220000.0 & 1.0 & \dots & 0 & 1 & 0 \\ 455000.0 & 0 & \dots & 1 & 0 & 1 \\ 88000.0 & 0 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}_{M \times N}$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ \dots & \dots \\ \dots & \dots \end{pmatrix}$$

M examples each with N features

Data Splitting

- After transforming, we (randomly) split the data into:
  - *Training set* – used repeatedly for learning (usually 80% of the data)
  - *Validation set* – used for checking “goodness” of model intermittently to decide which direction should the training go into (usually 10% of the data)
  - *Test set* – used once to evaluate the final model (usually 10%)

$$\begin{pmatrix} 250000.0 & 1.0 & \dots & 1 & 0 & 0 \\ 220000.0 & 1.0 & \dots & 0 & 1 & 0 \\ 455000.0 & 0 & \dots & 1 & 0 & 1 \\ 88000.0 & 0 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ \dots & \dots \\ \dots & \dots \end{pmatrix}$$

Training

Validation

Testing

What is Training?

- **Machine Learning:** automatically learning programs (i.e.,decision function) from experiences (data)

- **Training an ML model:** Define a decision function that **minimizes the error** on the training dataset

What is training?

- For a set of M training examples, each containing N features , minimize the average error on examples

$$\underset{f}{\text{minimize}} \quad \frac{1}{M}\sum_{\mathrm{i=1}}^{\mathrm{M}} \mathrm{Err}(\mathrm{y}^{\mathrm{i}}_{\mathrm{actual}}, \mathrm{y}^{\mathrm{i}}_{\mathrm{predicted}})$$

$$\Rightarrow \underset{f}{\text{minimize}} \quad \frac{1}{M}\sum_{\mathrm{i=1}}^{\mathrm{M}} \mathrm{Err}(\mathrm{y}^{\mathrm{i}}_{\mathrm{actual}}, \mathrm{f}(x_1^i, x_2^i, \dots, x_N^i))$$

What is Err( )?

- Mathematical measure that quantifies how well a machine learning model's predictions match the actual (true) values of the data it's trying to learn from.
- Often referred to as "**Loss**" function or "**Empirical Risk**" in ML
- Many possibilities
- Let's start with a simple (and elegant) error function

$$(y_{\text{predicted}} - y_{\text{actual}})^2$$

What is training – regression?

- For a set of M training examples, each containing N features

$$\underset{f}{\text{minimize}} \frac{1}{M} \sum_{i=1}^{M} (y^{i}_{\text{actual}} - y^{i}_{\text{predicted}})^2$$

Mean squared error

Parametric Function

$$\Rightarrow \underset{f}{\text{minimize}} \frac{1}{M} \sum_{i=1}^{M} (y^{i}_{\text{actual}} - f(x^{i}_{1}, x^{i}_{2}, \ldots, x^{i}_{N}))^2$$

Error for Classification

- Mathematical measure that quantifies how well a machine learning model's predictions match the actual (true) values of the data it's trying to learn from.
- Often referred to as “**Loss**” function or “**Empirical Risk**” in ML
- Many possibilities
- **Cross Entropy Error Example (Suitable for Classificaiton**

$$Err = -\sum_{c \in C} y^{c}_{actual} . log\ y^{c}_{predicted}$$

Where C is a collection of all possible classes

What is training – classification?

- For a set of M training examples, each containing N features

$$\underset{f}{\text{minimize}} \quad -\frac{1}{M}\sum_{i=1}^{M}\sum\nolimits_{c \in C} y_{actual}^{i,c} \cdot log\, y_{predicted}^{i,c}$$

$$= \underset{f}{\text{minimize}} -\frac{1}{M}\sum_{i=1}^{M}\sum\nolimits_{c \in C} y_{actual}^{i,c} \cdot \log(\mathrm{f}^{c}(x_1^i, x_2^i, \dots, x_N^i))$$

What is f( )

- Can be any mathematical function
- BUT we restrict it to a certain class of functions
- Example:
  - Naïve Bayes: Models based on Bayes' theorem
  - Makes the "naive" assumption that the features used to make predictions are conditionally independent

Naïve Bayes

- $f^{c}\left(x_1^i, x_2^i, \dots, x_N^i\right) = p(C \mid x_1^i, x_2^i, \dots, x_N^i)$

$$= \frac{P(C).P\left(x_1^i, x_2^i, \dots, x_N^i \middle| C\right)}{P\left(x_1^i, x_2^i, \dots, x_N^i\right)} \approx P(C).P\left(x_1^i, x_2^i, \dots, x_N^i \middle| C\right)$$

$$= \underbrace{P(C)}_{\text{Prior}}.\prod_{i=1}^{N} \underbrace{p(x_j^i \mid C)}_{\text{Likelihood}}$$

Logistic Regression

- $\mathrm{f}^{\mathrm{c}}\left(x_1^i, x_2^i, \ldots, x_N^i\right) = \mathrm{p}(\mathrm{C}\ |x_1^i, x_2^i, \ldots, x_N^i)$

$$= \frac{1}{1 + e^{-(w_1 x_1^i + w_2 x_2^i + \cdots + w_N x_N^i + \beta)}}$$

Other model types

- Support vector machines
  - Modeling to draw margins that separate data
- Decision Trees
- Feed Forward neural Networks
  - To be covered in week 7

Text as Data

Text as Data - Why is it important?

- Many ML applications
- Classification:
    - ***Given a piece of text, classify it***
    - Sentiment / Emotion Recognition from text
    - Topic classification
    - Fake / Real news identification
- Sequence classification
    - ***Given a piece of text, assign labels to sub-strings***
    - Named Entity Identification
    - Part-of-speech tagging
- Text Generation:
    - Text Summarization
    - Automatic Translation
    - Question Answering

Feature Engineering on Text

- **Objective**: Get fixed length vectors from variable length input

Feature Engineering on Text

- **Objective**: Get fixed length vectors from variable length input
- **How** (we have done this in the past)**?**

Feature Engineering on Text

- **Objective**: Get fixed length vectors from variable length input
- **How** (we have done this in the past)**?**
  - **N-hot vectorization**
  - **TF-IDF vectorization**
  - **Averaged word embeddings (such as GloVE)**
  - **...**
- **We can also add linguistic features based on text processing**
  - E.g., POS / Dependency parse information

Feature Engineering on Text

- **Example:**

1. *let us learn machine learning*
2. *machine learning emphasizes on learning programs from data*
3. *machine learning is a branch of AI*

Feature Engineering on Text (...)

One-hot vectorization example

“machine learning emphasizes on learning programs from data”

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

“let us learn machine learning”

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

Feature Vector Length = number of unique words = vocab length

Feature Engineering on Text (...)

Tf-Idf example:

1. *machine learning is a branch of AI*

TF-IDF("machine") =$1 * \log\left(\frac{3}{3}\right) = 0$

TF-IDF("AI") =$1 * \log\left(\frac{3}{1}\right) = \log 3 = 0.47$

TF-IDF("branch") = $1 * \log\left(\frac{3}{1}\right) = \log 3 = 0.47$

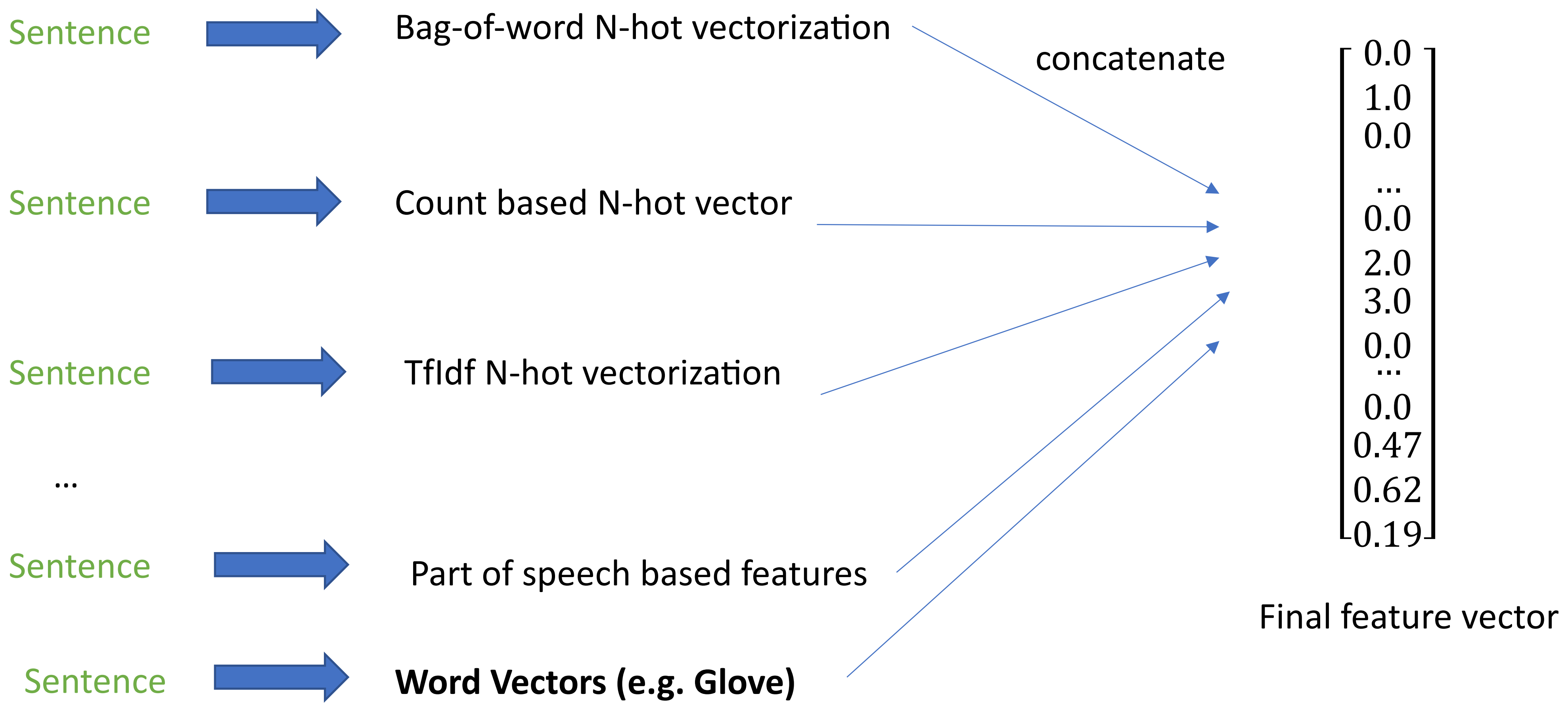
...

Feature vector = [0.47,0.47,0.47,0.47,0.47,0,0,0,0,0,0,0,0,0,0]

Other linguistic features

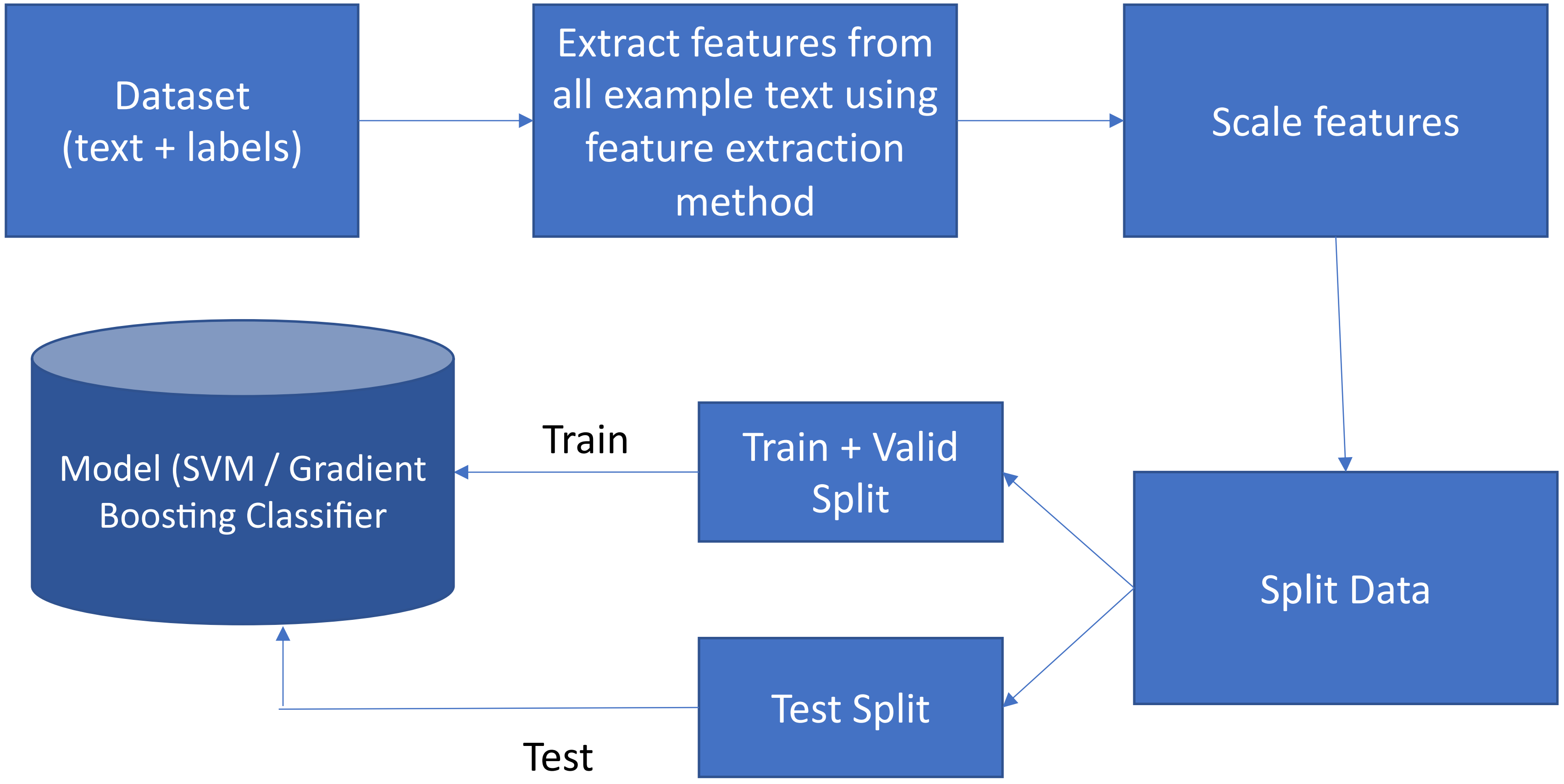
- Can be analyzed using linguistic properties
- Some examples:
  - How many nouns are present?
  - How many positive sentiment words are present?
  - How many negative sentiment words are present?
  - …
- Word embeddings (e.g., Glove)
- Sentence embeddings (e.g., BERT)

Feature Engineering (overall)



Final feature vector length should be same across all examples (both training and testing)

Performing classification



What is Evaluation (Testing)?

- Evaluate the performance of a model on a test dataet
- **Classification**
  - **Accuracy:** *how many times predicted class is equal to the actual class in test dataset*
  - *Precision, Recall , F1 scores*

- **Regression**
  - Mean Squared Error
  - Mean Absolute Error
  - Correlation between predicted and actual values

Examples from Literature

Linguistic Features – Example: Sarcasm Detection (Joshi et al, 2015)

- Objective – classify short sentences as sarcastic / not

Lexical	
Unigrams	Unigrams in the training corpus
Pragmatic	
Capitalization	Numeric feature indicating presence of capital letters
Emoticons & laughter expressions	Numeric feature indicating presence of emoticons and 'lol's
Punctuation marks	Numeric feature indicating presence of punctuation marks
Implicit Incongruity	
Implicit Sentiment Phrases	Boolean feature indicating phrases extracted from the implicit phrase extraction step
Explicit Incongruity	
#Explicit incongruity	Number of times a word is followed by a word of opposite polarity
Largest positive /negative subsequence	Length of largest series of words with polarity unchanged
#Positive words	Number of positive words
#Negative words	Number of negative words
Lexical Polarity	Polarity of a tweet based on words present

Bag of words

Table 1: Features of our sarcasm detection system

Joshi, A., Sharma, V., & Bhattacharyya, P. (2015, July). Harnessing context incongruity for sarcasm detection. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers) (pp. 757-762).

Classification Results

- Showing feature importance through ablation studies
- **Ablation studies:** Remove one of more features at a time and repeat training and testing

Features	P	R	F
Original Algorithm by Riloff et al. (2013)			
Ordered	0.774	0.098	0.173
Unordered	0.799	0.337	0.474
Our system			
Lexical (**Baseline**)	0.820	0.867	0.842
Lexical+Implicit	0.822	0.887	0.853
Lexical+Explicit	0.807	0.985	0.8871
All features	0.814	0.976	**0.8876**

Classifier = SVM

Joshi, A., Sharma, V., & Bhattacharyya, P. (2015, July). Harnessing context incongruity for sarcasm detection. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers) (pp. 757-762).

Another Example: Readability Assessment of Healthcare Text

Objective – Predict readability score (0-100) of a document with the help of the following features

Feature category	Name
Raw Text	Average number of words per sentence Average number of characters per word
Lexical	Type/Token Ratio Lexical density *Basic Italian Vocabulary (BIV)* (De Mauro, 2000) rate
Morpho–syntactic	Part-Of-Speech unigrams Mood, tense and person of verbs
Syntactic	Distribution of dependency types Depth of the whole parse tree Average depth of embedded complement 'chains' Distribution of embedded complement 'chains' by depth Number of verbal roots Arity of verbal predicates Distribution of verbal predicates by arity Distribution of subordinate vs main clauses Relative ordering with respect to the main clause the Average depth of 'chains' of embedded subordinate clauses Distribution of embedded subordinate clauses 'chains' by depth Length of dependency links feature

Venturi, G., Bellandi, T., Dell'Orletta, F., & Montemagni, S. (2015, September). NLP–based readability assessment of health–related texts: a case study on Italian informed consent forms. In Proceedings of the Sixth International Workshop on Health Text Mining and Information Analysis (pp. 131-141).

Results

Classifier = SVM

Medical Specialty	nº documents	nº tokens	READ–IT		
			Base	Lexical	Syntax
Anesthesiology	20	21,065	50	93.37	69.62
Colorectal surgery	2	1,997	75.18	100	93.81
Obesity surgery	3	8,091	51.63	93.42	59.20
General surgery	19	11,588	43.03	78.29	58
Plastic surgery	4	3,550	88.95	98.72	96.51
Thoracic surgery	9	5,608	94.98	99.94	95.55
Vascular surgery	16	22,739	88.64	98.13	97.62
Ophthalmology	7	10,496	49.21	98.89	61.29
Otorhinolaryngology	134	194,421	25.14	94.90	69.42
Orthopaedics	44	76,712	50.54	97.58	89.66
Obstetrics and gynecology	35	31,243	60.37	97.31	58.52
Urology	17	19,576	85.40	98.08	89.16
TOTAL: Surgery	**313**	**407,086**	**63.59**	**95.72**	**78.19**
Cardiology	54	39,887	66.20	94.50	78.99
Diabetology	1	297	23.05	100	45.68
Gastroenterology	9	9,856	41.12	87.90	59.82
Neurology	8	5,199	69.44	97.96	94.98
Oncology	3	1,692	46.34	99.73	96.07
Pulmonology	4	3,220	49.57	98.18	78.27
Senology	17	20,455	85.09	99.68	93.88
TOTAL: Internal Medicine	**96**	**80,309**	**54.26**	**96.85**	**78.24**
Psychology	13	11,651	80.44	96.25	98.32
Screening	8	2,007	53.13	65.14	50.60
Vaccine	1	2,852	33.72	100	71.76
TOTAL: Prevention	**22**	**16,510**	**55.76**	**87.13**	**73.56**
Genetics	11	6,416	56.26	95.65	81.45
Immunohematology and transfusion	43	45,962	56.84	93.39	83.47
Nuclear medicine	29	18,045	52.62	96.56	68.48
Radiology	24	17,358	63.78	98.61	78.68
TOTAL: Medical Services	**107**	**87,781**	**57.38**	**96.05**	**78.02**
General	33	8,928	51.59	87.81	88.27
Pediatrics	13	6,092	49.84	99.46	74.67
Rehabilitation	2	674	63.84	99.99	96.25

Summary

- **Open-Ended Text Analysis Often Demands ML-Based Solutions**
  - Leveraging machine learning is frequently essential for uncovering insights from open-ended text data.
- **Feature-Based Methods Remain Effective in Specialized Domains**
  - Feature-based approaches maintain relevance in specific domains where data characteristics differ significantly from the mainstream.
- **However, Deep Learning Dominates Broader Solution Landscapes**
  - Deep learning techniques have emerged as a dominant force in addressing general open-ended text analysis challenges.

Summary (1)

- **The Crucial Role of Labeled Datasets**:
  - Building labeled datasets is a foundational step to train accurate and effective machine learning models for text analysis.
- **The Pitfalls of Relying Solely on Accuracy**
  - Accuracy as a sole metric can be deceptive; consider broader evaluation concepts to capture model performance accurately.
- **Rethinking Evaluation Metrics: Precision, Recall, F-score**
  - Reevaluate your model's performance using precision, recall, and F-score to gain a more comprehensive understanding of its effectiveness.

Next class:

- **Tutorial:**
  - Building ML classifiers with text data
- **Fill out this form for group formation**
  - https://forms.gle/H9akcB9PGLNEmmUU8