

# **DSC 102**

# **Systems for Scalable Analytics**



Haojian Jin

Topic 5: Model Building Systems

Chapter 8.1 and 8.3 of MLSys Book

# Tentative Course Schedule

- ❖ Submit your SET. Deadline. Saturday, June 8 at 8:00 AM.
- ❖ <https://academicaffairs.ucsd.edu/Modules/Evals>
- ❖ 85%+ SET response for class yields => 0.5% collective boost to final score for everyone.
- ❖ Mid term scores are out. Final exam would be harder. I am working on the design right now.

Spring 2024	Student Evaluation of Teaching (SET)	DSC 102 [A00]	22.45% (33/147)		
Spring 2024	Student Evaluation of Teaching (SET)	DSC 102 [A00]	29.25% (43/147)	Available 6/19/2024 8:00 AM	
Spring 2023	Graduate Course Evaluation	DSC 204A [A00]	89.80% (44/49)		

# Tentative Course Schedule

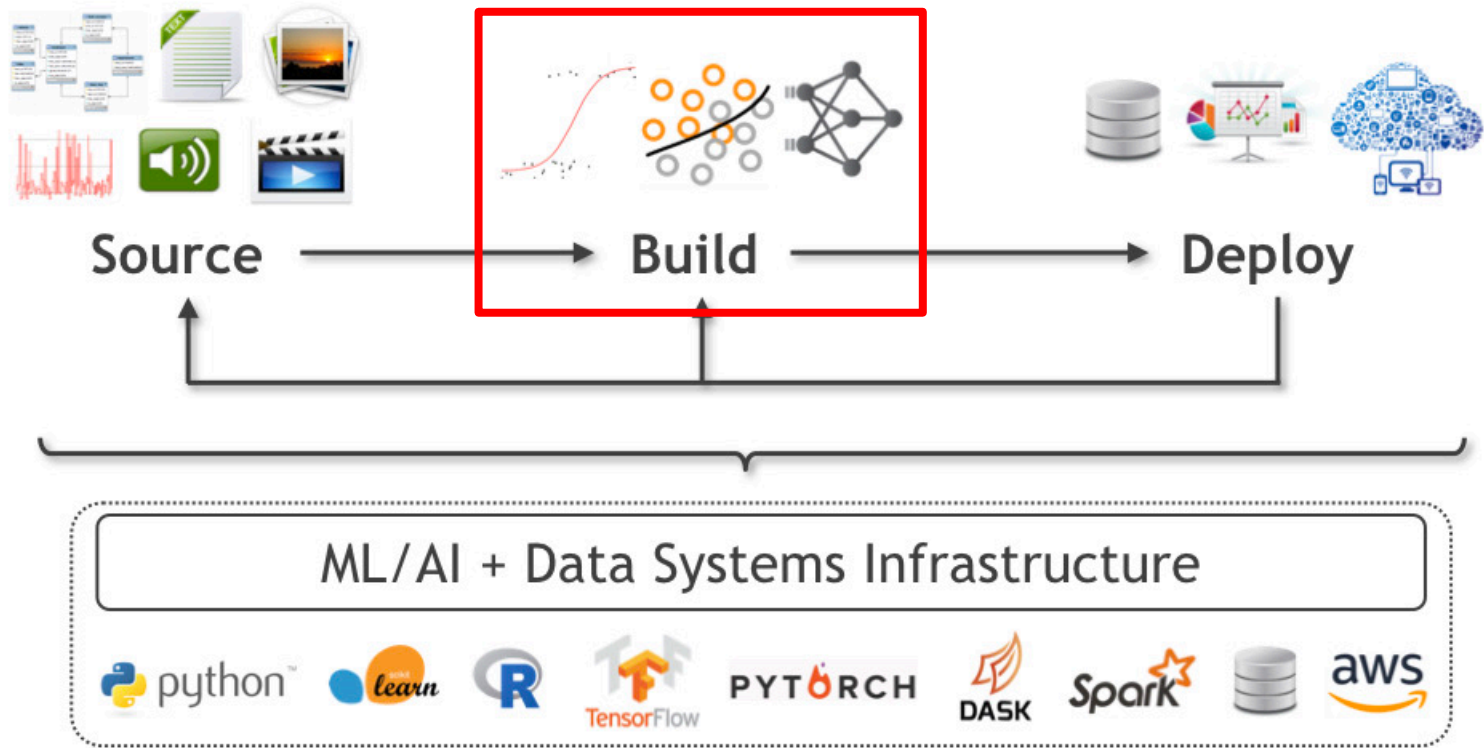
Week	Topic		
Systems Principles	Basics of Machine Resources: Computer Organization		
	Basics of Machine Resources: Operating Systems		
	4	Basics of Cloud Computing	
4-5	Parallel and Scalable Data Processing: Parallelism Basics		
Scalability Principles	Midterm Exam on TBD		
	6-7	Parallel and Scalable Data Processing: Scalable Data Access	
	7-8	Parallel and Scalable Data Processing: Data Parallelism	
9	Scalable Analytics Systems	Dataflow Systems	
10		ML Model Building Systems	
11		Final Exam on Dec 15	

There will be 2 industry guest lectures (maybe 3)

# The Lifecycle of ML-based Analytics



Data Scientist/  
ML Engineer



Data acquisition  
Data preparation

Feature Engineering  
Training & Inference  
Model Selection

Serving  
Monitoring

# Building Stage of ML Lifecycle

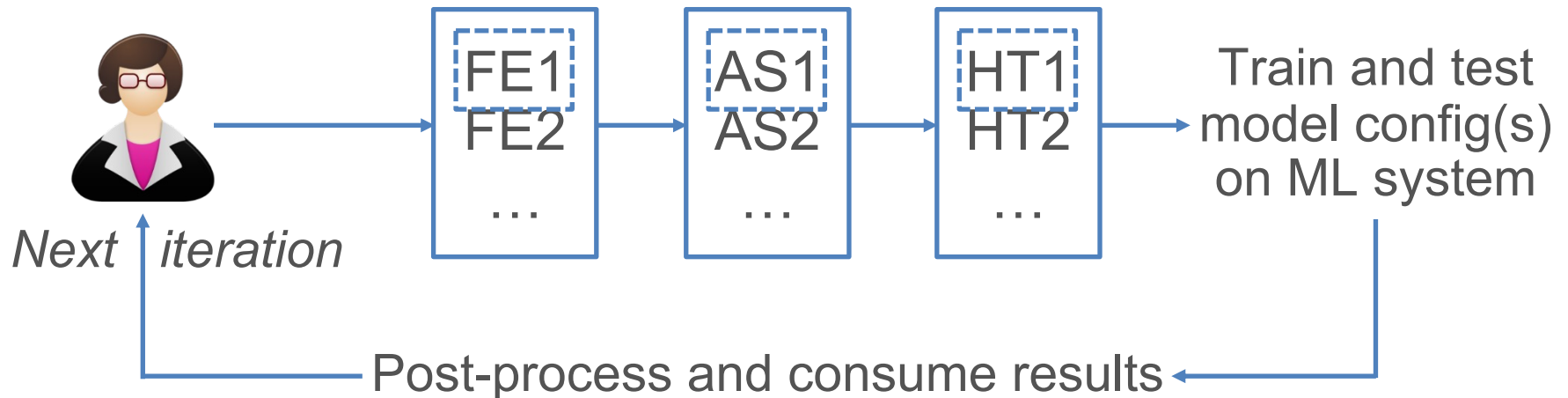
- ❖ Perform **model selection**, i.e., convert prepared ML-ready data to **prediction function(s)** and/or other analytics outputs
- ❖ What makes model building challenging/time-consuming?
  - ❖ **Heterogeneity** of data sources/formats/types
  - ❖ **Configuration complexity** of ML models
  - ❖ Large **scale** of data
  - ❖ Long **training runtimes** of some models
  - ❖ **Pareto optimization** on criteria for application
  - ❖ Data-generating process/application **keeps evolving**

# Building Stage of ML Lifecycle

- ❖ Perform **model selection**, i.e., convert prepared ML-ready data to **prediction function(s)** and/or other analytics outputs
- ❖ Data scientist / ML engineer must steer 3 key activities that invoke ML **training** and **inference** as sub-routines:
  1. **Feature Engineering (FE)**: How to represent signals appropriately for domain of prediction function?
  2. **Algorithm/Architecture Selection (AS)**: What class of prediction functions (incl. ANN architecture) to use?
  3. **Hyper-parameter Tuning (HT)**: How to improve accuracy/etc. by configuring ML “knobs” better?

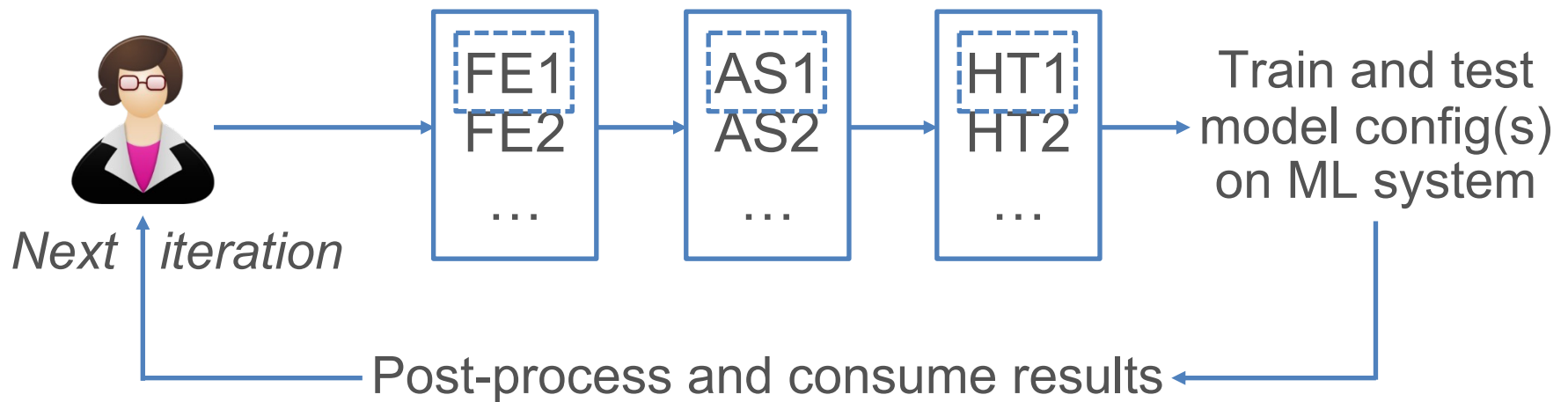
# Model Selection Process

- ❖ Model selection is usually an *iterative exploratory* process with human making decisions on FE, AS, and/or HT
- ❖ Increasingly, automation of some or all parts possible: **AutoML**



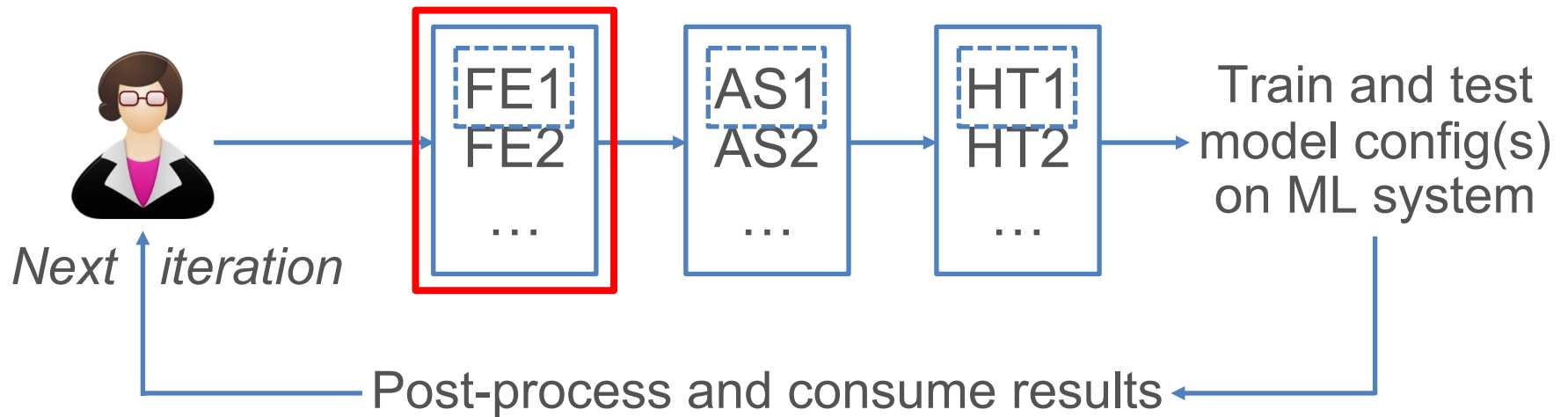
# Model Selection Process

- ❖ Decisions on FE, AS, HT guided by many constraints/metrics: prediction accuracy, data/feature types, interpretability, tool availability, scalability, runtimes, fairness, legal issues, etc.
- ❖ Decisions are typically application-specific and dataset-specific; recall Pareto surfaces and tradeoffs





# Feature Engineering



# Feature Engineering

- ❖ Converting prepared data into a *feature vector representation* for ML training and inference
  - ❖ Aka feature extraction, representation extraction, etc.
- ❖ Umbrella term for many tasks dep. on type of ML model trained:
  1. Recoding and value conversions
  2. Joins and/or aggregates
  3. Feature interactions
  4. Feature selection
  5. Dimensionality reduction
  6. Temporal feature extraction
  7. Textual feature extraction and embeddings
  8. Learned feature extraction in deep learning

# 1. Recoding and value conversions

- ❖ Common on relational/tabular data
- ❖ Typically needs some *global column stats* + code to *reconvert* each tuple (example's feature values)

UserID	State	Date	Upvotes	Comment	Label
143	CA	4/3/19	1539	"This restaurant is overrated"	-
337	NY	11/7/19	5020	"Not too bad!"	+
98	WI	2/8/20	402	"Pretty rad"	+
...	...	...	...	...	...

## Example:

Decision trees can use categorical features directly but GLMs support only numeric features; need **one-hot encoded** 0/1 vector

**Scaling global stats:** "SELECT DISTINCT State"?

**Reconversion:** Tuple-level function to look up domain hash table

# 1. Recoding and value conversions

- ❖ Common on relational/tabular data
- ❖ Typically needs some *global column stats* + code to *reconvert* each tuple (example's feature values)

UserID	State	Date	Upvotes	Comment	Label
143	CA	4/3/19	1539	"This restaurant is overrated"	-
337	NY	11/7/19	5020	"Not too bad!"	+
98	WI	2/8/20	402	"Pretty rad"	+
...	...	...	...	...	...

## Example:

GLMs and ANNs need **whitening** of numeric features; dense: subtract mean and divide by stdev; sparse: divide by max-min

**Scaling global stats:** How to scale mean/stdev/max/min?

**Reconversion:** Tuple-level function to modify number using stats

# 1. Recoding and value conversions

- ❖ Common on relational/tabular data
- ❖ Typically needs some *global column stats* + code to *reconvert* each tuple (example's feature values)

UserID	State	Date	Upvotes	Comment	Label
143	CA	4/3/19	1539	"This restaurant is overrated"	-
337	NY	11/7/19	5020	"Not too bad!"	+
98	WI	2/8/20	402	"Pretty rad"	+
...	...	...	...	...	...

## Example:

Some models like Bayesian Networks or Markov Logic Networks benefit from (or even need) **binning/discretization** of numerics

**Scaling global stats:** How to scale histogram computations?

**Reconversion:** Tuple-level function to convert number to bin ID

## 2. Joins and Aggregates

- ❖ Common on relational/tabular data
- ❖ Most real-world relational datasets are multi-table; require key-foreign key joins, aggregation-and-key-key-joins, etc.

UserID	Age	Name
304	40	...
23	25	...
143	33	...
...	...	...

UserID	State	Date	Upvotes	Comment	Label
143	CA	...	...	...	-
337	NY	...	...	...	+
143	CA	...	...	...	+
...	...	...	...	...	...

### Example:

Join tables on UserID; concatenate user's info. as extra features!

What kind of join is this? How to scale this computation?

## 2. Joins and Aggregates

- ❖ Common on relational/tabular data
- ❖ Most real-world relational datasets are multi-table; require key-foreign key joins, aggregation-and-key-key-joins, etc.

UserID	State	Date	Upvotes	Comment	Label
143	CA	...	...	...	-
337	NY	...	...	...	+
143	CA	...	...	...	+
...	...	...	...	...	...

### Example:

Join table with itself on UserID to count #reviews and avg #upvotes for each user in a new temp. table and join that to get more features!

What kind of computation is this? How to scale it?

# 3. Feature Interactions

- ❖ Sometimes used on relational/tabular data, especially for high-bias models like GLMs
- ❖ Pairwise is common; ternary is not unheard of

F1	F2	F3	Label
3	2	...	-
4	20	...	+
5	10	...	+
...	...	...	...

F1	F2	F3	F11	F12	F13	F22	F23	F33	Label
3	2	...	9	6	...	4	...	...	-
4	20	...	16	80	...	400	...	...	+
5	10	...	25	50	...	100	...	...	+
...	...	...	...	...	...	...	...	...	...

- ❖ No global stats, just a tuple-level function
- ❖ **NB:** Popularity of this has reduced due to kernel SVMs; but so-called “factorization machines” still need this



# 4. Feature Selection

- ❖ Sometimes used on relational/tabular data
- ❖ **Basic Idea:** Instead of using whole feature set, use a subset

UserID	State	Date	Upvotes	Comment	Label
...	...	...	...	...	...

State	Upvotes	Comment	Label
...	...	...	...

Upvotes	Comment	Label
...	...	...

...

- ❖ Formulated as a *discrete optimization* problem
  - ❖ NP-Hard in #features in general
  - ❖ Many heuristics exist in ML/data mining; typically rely on some *information theoretic criteria*
  - ❖ Typically scaled as “outer loops” over training/inference
- ❖ Some ML users also prefer human-in-the-loop approach

# 5. Dimensionality Reduction

- ❖ Often used on relational/structured/tabular data
- ❖ **Basic Idea:** Transforms features to a different latent space
- ❖ **Examples:** PCA, SVD, LDA, Matrix factorization

UserID	State	Date	Upvotes	Comment	Label
...	...	...	...	...	...

F1	F2	F3	Label
0.3	4.2	-29.2	...

**Q:** How is this different from “feature selection”?

- ❖ Feat. sel. *preserves* semantics of each feature but dim. red. typically does not—combines features in “nonsensical” ways
- ❖ Scaling this is non-trivial! Similar to scaling individual ML training algorithms (later)

# 6. Temporal Feature Extraction

- ❖ Many relational/tabular data have time/date
- ❖ Per-example reconversion to extract numerics/categoricals
- ❖ Sometimes global stats needed to calibrate time
- ❖ Complex temporal features studied in *time series mining*

UserID	State	Date	Upvotes	Comment	Label
143	CA	4/3/19	1539	"This restaurant is overrated"	-
337	NY	11/7/19	5020	"Not too bad!"	+
98	WI	2/8/20	402	"Pretty rad"	+
...	...	...	...	...	...

## Example:

Most classifiers cannot use Date directly; extract month (categorical), year (categorical?), day? (categorical), etc.

**Reconversion:** Tuple-level function to extract numbers/categories

# 7. Textual Feature Extraction

- ❖ Many relational/tabular data have text columns; in NLP, whole example is often just text
- ❖ Most classifiers cannot process text/strings directly
- ❖ Extracting numerics from text studied in *text mining*

...	Comment	Label
...	"This restaurant is sucks"	-
...	"Good good!"	+
...	"Pretty rad"	+
...	...	...

...	sucks	good	...	Label
...	1	0	...	-
...	0	2	...	+
...	0	0	...	+
...	...	...	...	...

**Example:**

**Bag-of-words** features: count number of times each word in a given *vocabulary* arises; need to know vocabulary first

**Scaling global stats:** How to get vocabulary?

**Reconversion:** Tuple-level function to count words; look up index

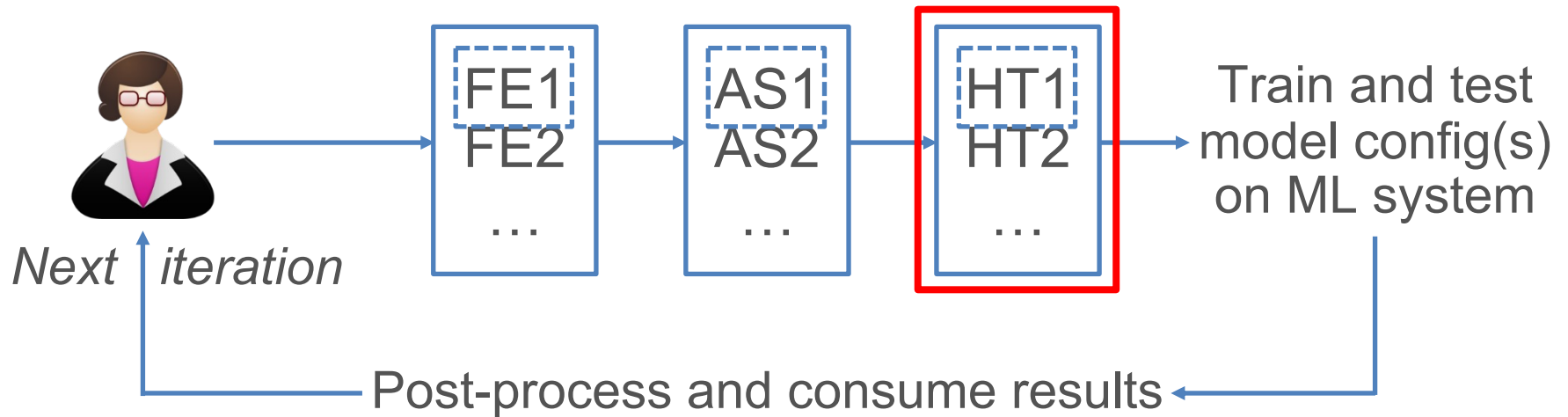
# 7. Textual Feature Extraction

- ❖ **Knowledge Base-based:** Domain-specific knowledge bases like entity dictionaries (e.g., celebrity or chemical names) help extract domain-specific features
- ❖ **Embedding-based:**
  - ❖ Numeric vector for a text token; popular in NLP
  - ❖ Offline training of function from string to numeric vector in self-supervised way on large text corpus (e.g., Wikipedia); embedding dimensionality is a hyper-parameter
  - ❖ *Pre-trained* word embeddings (Word2Vec and GloVe) and sentence embeddings (Doc2Vec) available off-the-shelf; to scale, just use a tuple-level conversion function

# 8. Learned Feature Extraction in DL

- ❖ A big win of DL is no manual feature eng. on *unstructured* data
  - ❖ **NB:** DL is *not* common on structured/tabular/relational data!
- ❖ DL is very *versatile*: almost *any data type* as input and/or output:
  - ❖ Convolutional NNs (CNNs) over images, video, time series data
  - ❖ Transformers and Recurrent NNs (RNNs) over text, sequence data
  - ❖ Graph NNs (GNNs) over graph-structured data
- ❖ Neural architecture specifies how to extract and transform features internally with weights that are learned
- ❖ **Software 2.0:** Buzzword for such “learned feature extraction” programs vs old hand-crafted feature engineering

# Hyper-Parameter Tuning

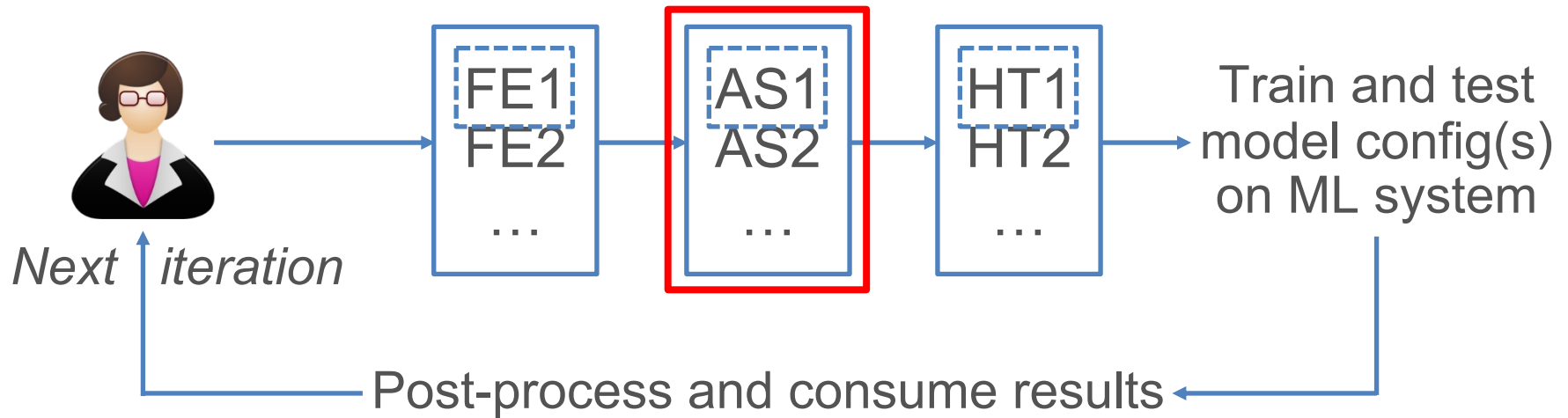


# Hyper-Parameter Tuning

- ❖ **Hyper-parameters:** Knobs for an ML model or training algorithm to control *bias-variance* tradeoff in a dataset-specific manner to make learning effective
- ❖ **Examples:**
  - ❖ GLMs: L1 or L2 *regularizer* to constrain weights
  - ❖ All gradient methods: *learning rate*
  - ❖ SGD: *batch size*
- ❖ HT is an “outer loop” around training/inference
- ❖ Most common approach: **grid search**; pick set of values for each hyper-parameter and take cartesian product
- ❖ Also common: **random search** to subsample from grid
- ❖ Complex AutoML heuristics exist too for HT, e.g., HyperOpt

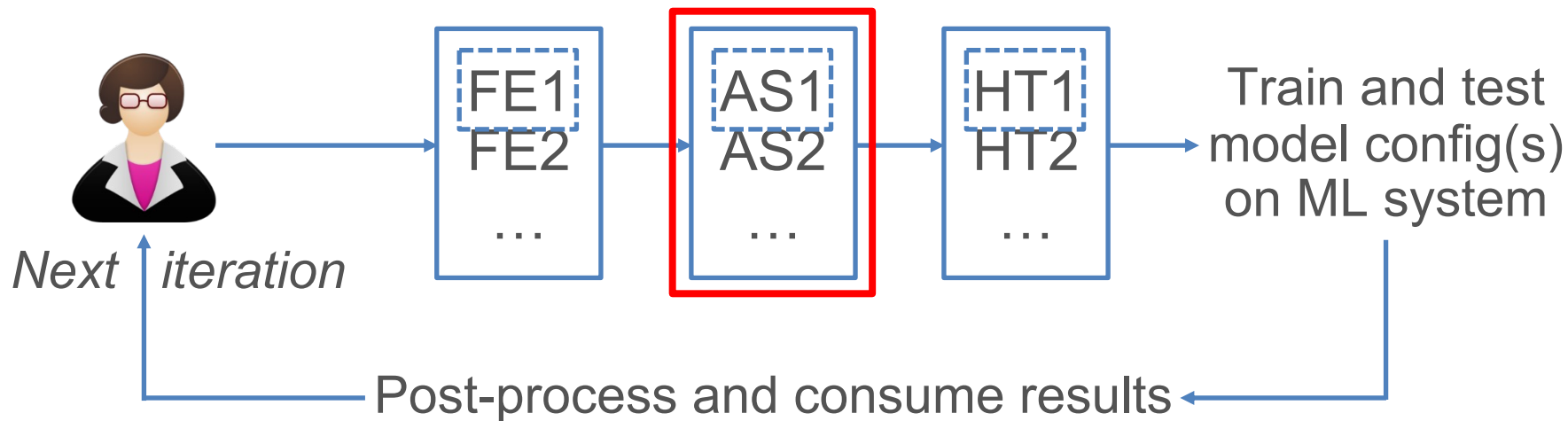


# Algorithm Selection



- ❖ Not much to say; ML user typically picks models/algorithms based on intuitions in “**classical**” ML (non-DL)
- ❖ Best practice: first train simple models (log. reg.) as baselines; then try complex models (XGBoost)
- ❖ **Ensembles**: Build diverse models and aggregate predictions

# Architecture Selection in DL



- ❖ More critical in DL; neural arch. is **inductive bias** in classical ML parlance; controls feature learning and bias-variance tradeoff
- ❖ Some applications: Many off-the-shelf pre-trained DL models to do “transfer learning,” e.g., [HuggingFace Models](#)
- ❖ Other applications: Swap pain of hand-crafted feature eng. for pain of neural arch. eng.! :)

# Pre-trained Models on HuggingFace

The screenshot displays the HuggingFace website interface. At the top, the navigation bar includes the HuggingFace logo, a search bar, and links for Models, Datasets, Spaces, Docs, Solutions, Pricing, Log In, and Sign Up. The main content area is divided into two columns. The left column features filters for Tasks, Libraries, Datasets, and Languages. The right column displays a list of pre-trained models, sorted by most downloads. The models listed include gpt2, bert-base-uncased, xlm-roberta-base, openai/clip-vit-large-patch14, roberta-base, allenai/specter, and Jean-Baptiste/camembert-ner. Each model entry shows its name, update date, download count, and heart count.

**Tasks**

- Image Classification
- Image Segmentation
- Automatic Speech Recognition
- Token Classification
- Audio Classification
- Summarization
- Translation
- Fill-Mask
- Sentence Similarity
- Question Answering
- Zero-Shot Classification

+ 22 Tasks

**Libraries**

- PyTorch
- TensorFlow
- JAX

+ 31

**Datasets**

- mozilla-foundation/common\_voice\_7\_0
- wikipedia
- emotion
- common\_voice
- bookcorpus
- squad
- glue
- xtreme

+ 308

**Languages**

- English
- French
- German
- Spanish
- Chinese
- Japanese

**Models** 85,177  Sort: Most Downloads

- gpt2**  
Updated 19 days ago • ↓ 32M • ♥ 282
- bert-base-uncased**  
Updated Oct 3 • ↓ 25.7M • ♥ 323
- xlm-roberta-base**  
Updated Jun 6 • ↓ 19.4M • ♥ 104
- openai/clip-vit-large-patch14**  
Updated Oct 4 • ↓ 10.3M • ♥ 66
- roberta-base**  
Updated Sep 29 • ↓ 7.33M • ♥ 77
- allenai/specter**  
Updated Jun 25 • ↓ 7.21M • ♥ 19
- Jean-Baptiste/camembert-ner**  
Updated 26 days ago • ↓ 7.17M • ♥ 38

# Automated Model Selection / AutoML

*Q: Can we automate the whole model selection process?*

- ❖ It depends. HT and most of FE already automated mostly in practice; (neural) AS is often application-dictated
- ❖ AutoML tools/systems now aim to reduce data scientist's work; or even replace them?! ;)



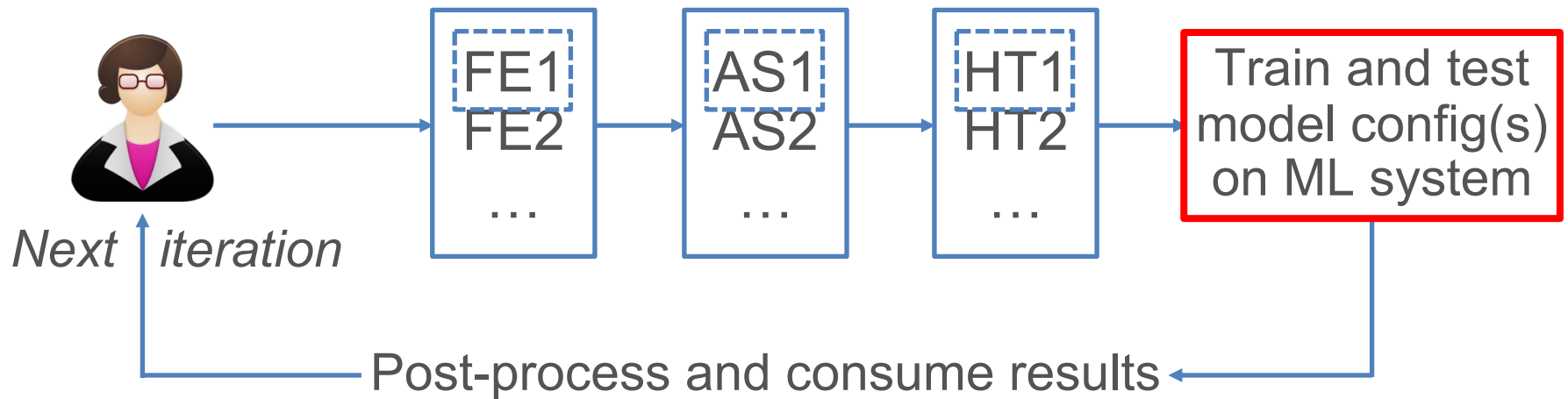
❖ **Pros:** Ease of use; lower human cost; easier to audit; improves ML accessibility

❖ **Cons:** Higher resource cost; less user control; may waste domain knowledge

❖ Pareto-optima; hybrids possible

**But:** The Data Sourcing stage is still very hard to automate!

# Scalable ML Training and Inference



# Major ML Model Families/Types

**Generalized Linear Models** (GLMs); from statistics

**Bayesian Networks**; inspired by causal reasoning

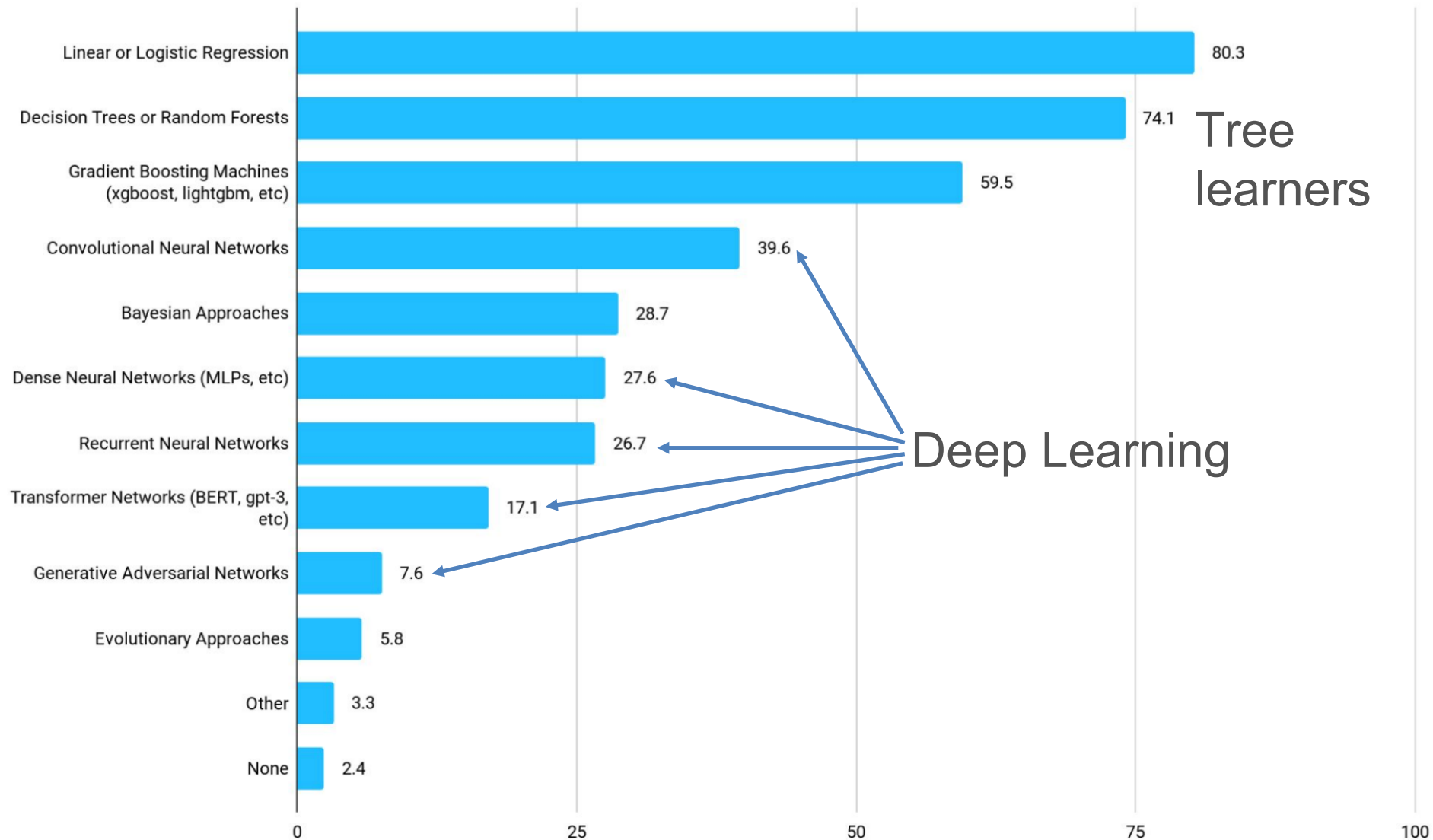
**Decision Tree-based**: CART, Random Forest, Gradient-Boosted Trees (GBT), etc.; inspired by symbolic logic

**Support Vector Machines** (SVMs); inspired by psychology

**Artificial Neural Networks** (ANNs): Multi-Layer Perceptrons (MLPs), Convolutional NNs (CNNs), Recurrent NNs (RNNs), Transformers, etc.; inspired by brain neuroscience

**Unsupervised**: Clustering (e.g., K-Means), Matrix Factorization, Latent Dirichlet Allocation (LDA), etc.

# ML Models in Kaggle 2021 Survey



# Scalable ML Training Systems

- ❖ Scaling ML training is involved and model type-dependent
- ❖ Orthogonal Dimensions of Categorization:
  - 1. Scalability:** In-memory libraries vs Scalable ML system (works on larger-than-memory datasets)
  - 2. Target Workloads:** General ML library vs Decision tree-oriented vs Deep learning, etc.
  - 3. Implementation Reuse:** Layered on top of scalable data system vs Custom from-scratch framework

**Ad: Take CSE 234 in for more on scalable ML systems**



# Major Existing ML Systems

## General ML libraries:

In-memory:



Disk-based files:



Layered on RDBMS/Spark:



Cloud-native:



“AutoML” platforms:



Decision tree-oriented:



Deep learning-oriented:



# Scalable ML Inference

- ❖ A trained/learned ML model is just a prediction function:

$$f : \mathcal{D}_X \rightarrow \mathcal{D}_Y$$

*Q: Given large dataset of examples, how to scale inference?*

- ❖ Assumption 1: An example fits entirely in DRAM
- ❖ Assumption 2:  $f$  fits entirely in DRAM
- ❖ If both hold, trivial access pattern: single filescan, apply per-tuple function  $f$ , write output. How to do this with MapReduce?
- ❖ If either fails, access pattern becomes more complex and dependent on breaking up internals of  $f$  to stage access to data for partial computations

## DSC 102 will get you thinking about the fundamentals of systems for scalable analytics

1. **“Systems”**: What resources does a computer have? How to store and efficiently compute over large data? What is cloud?
2. **“Scalability”**: How to scale and parallelize data-intensive computations?
3. **For “Analytics”**:
  1. **Source**: Data acquisition & preparation for ML
  2. **Build**: Feature eng. & model selection systems
  3. **Deploying** ML models
4. Hands-on experience with scalable analytics tools

# Tentative Course Schedule

Week	Topic	
Systems Principles	Basics of Machine Resources: Computer Organization	
	Basics of Machine Resources: Operating Systems	
	4	Basics of Cloud Computing
4-5	Parallel and Scalable Data Processing: Parallelism Basics	
Scalability Principles	Midterm Exam on TBD	
	Parallel and Scalable Data Processing: Scalable Data Access	
	6-7	Parallel and Scalable Data Processing: Data Parallelism
7-8		
9	Scalable Analytics Systems	Dataflow Systems
10		ML Model Building Systems
11	Final Exam on Dec 15	

Thank you for taking DSC 102.

Please make sure to submit your SET if you have not done so already.

All the best for final exams week!