Lecture 2:
Feature Engineering for AI

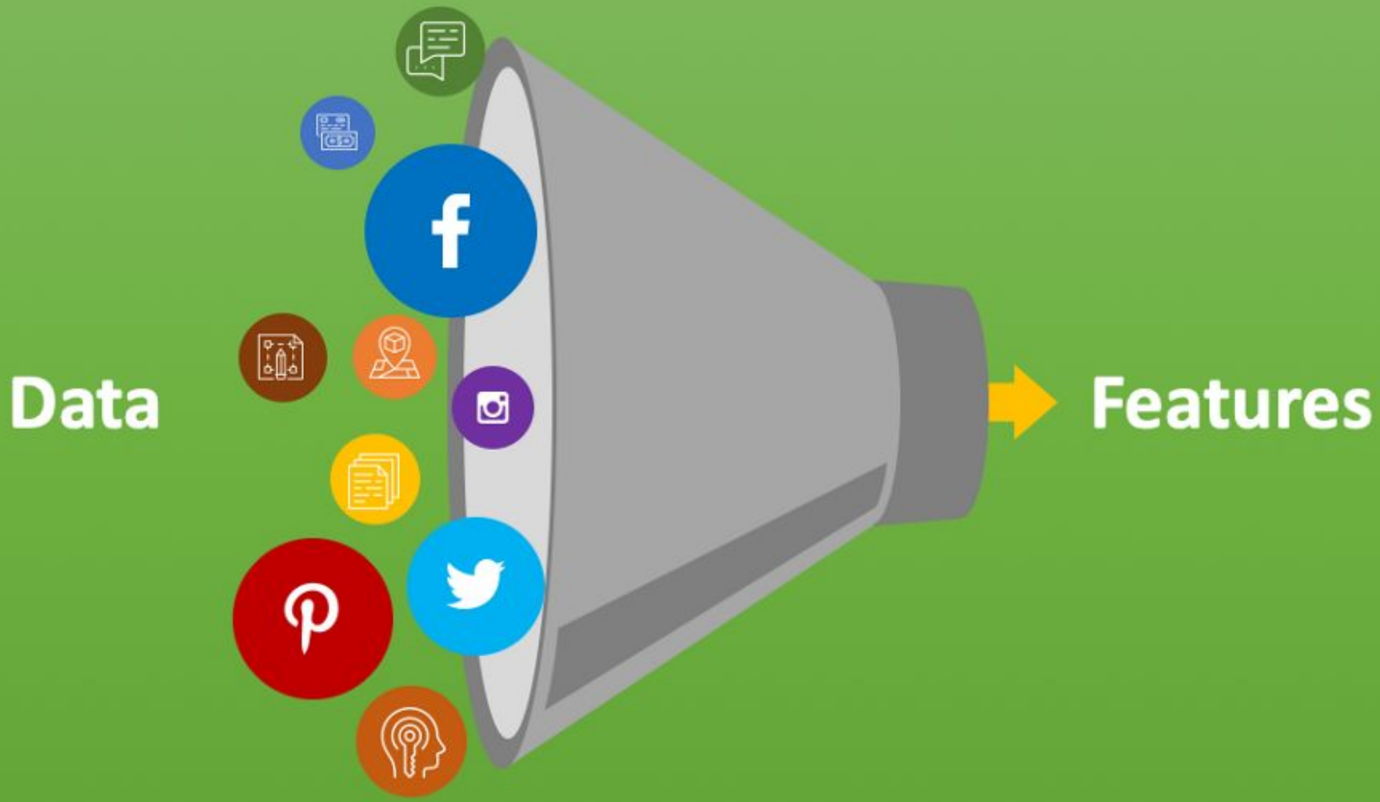
https://github.com/kaopanboonyuen/SC310005_ArtificialIntelligence_2025s1

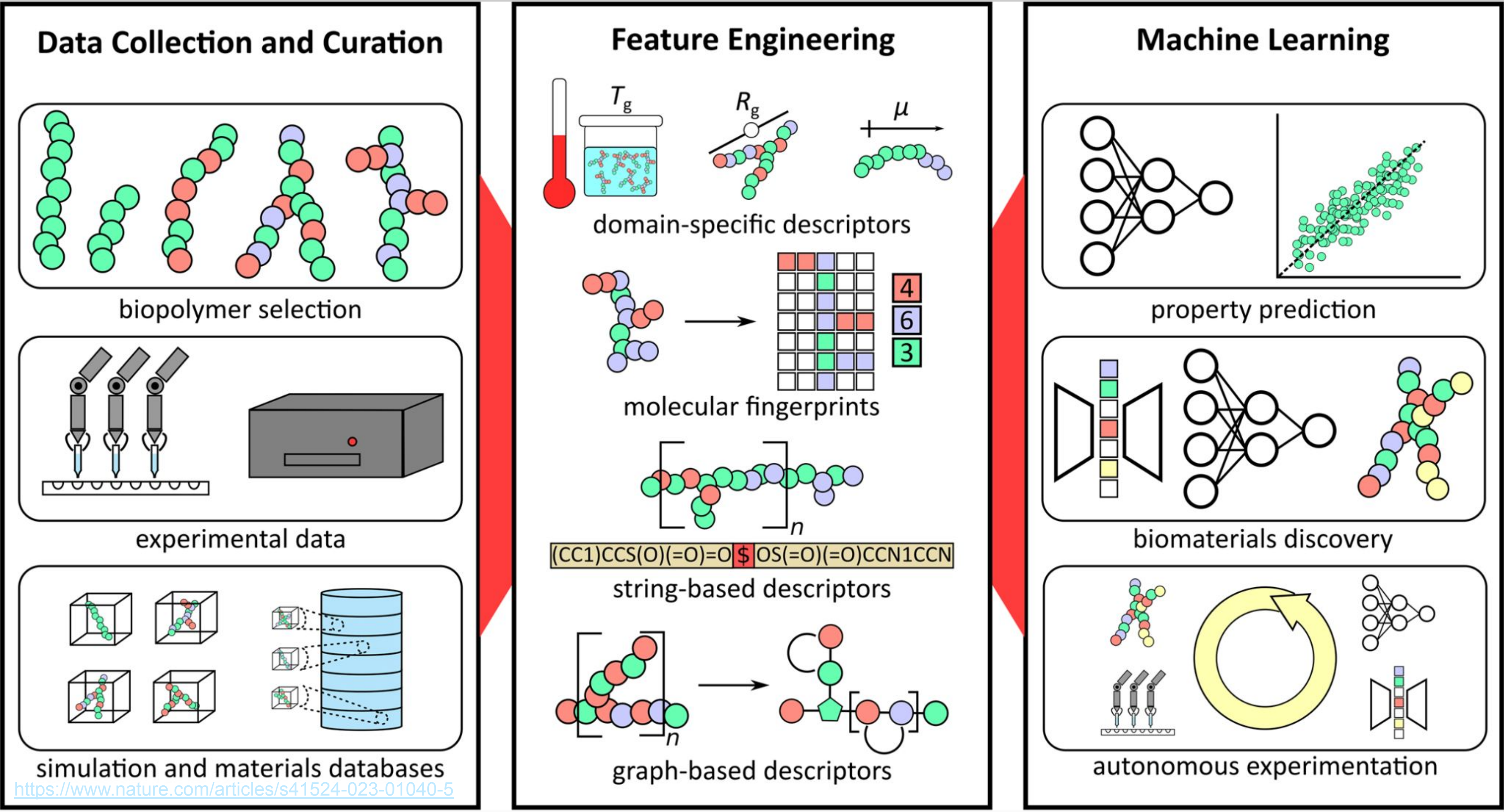
Teerapong Panboonyuen
https://kaopanboonyuen.github.io

🔍 What is Feature Engineering?

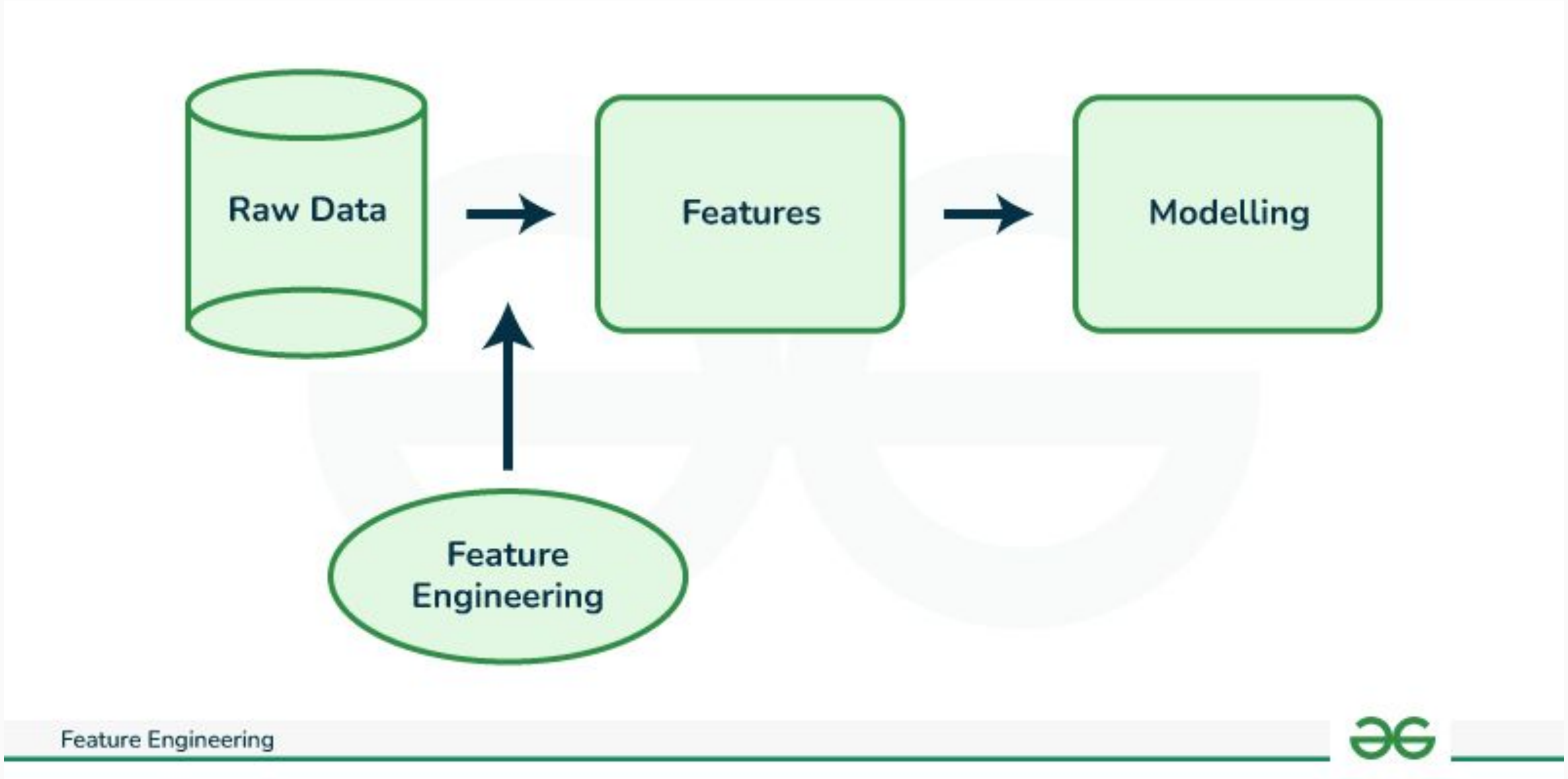
- Transform raw data into features usable by AI models
- Bridge between raw_input and predictive insight
- Boost model accuracy and interpretability

“Good features > complex models” – focus on understanding your data.

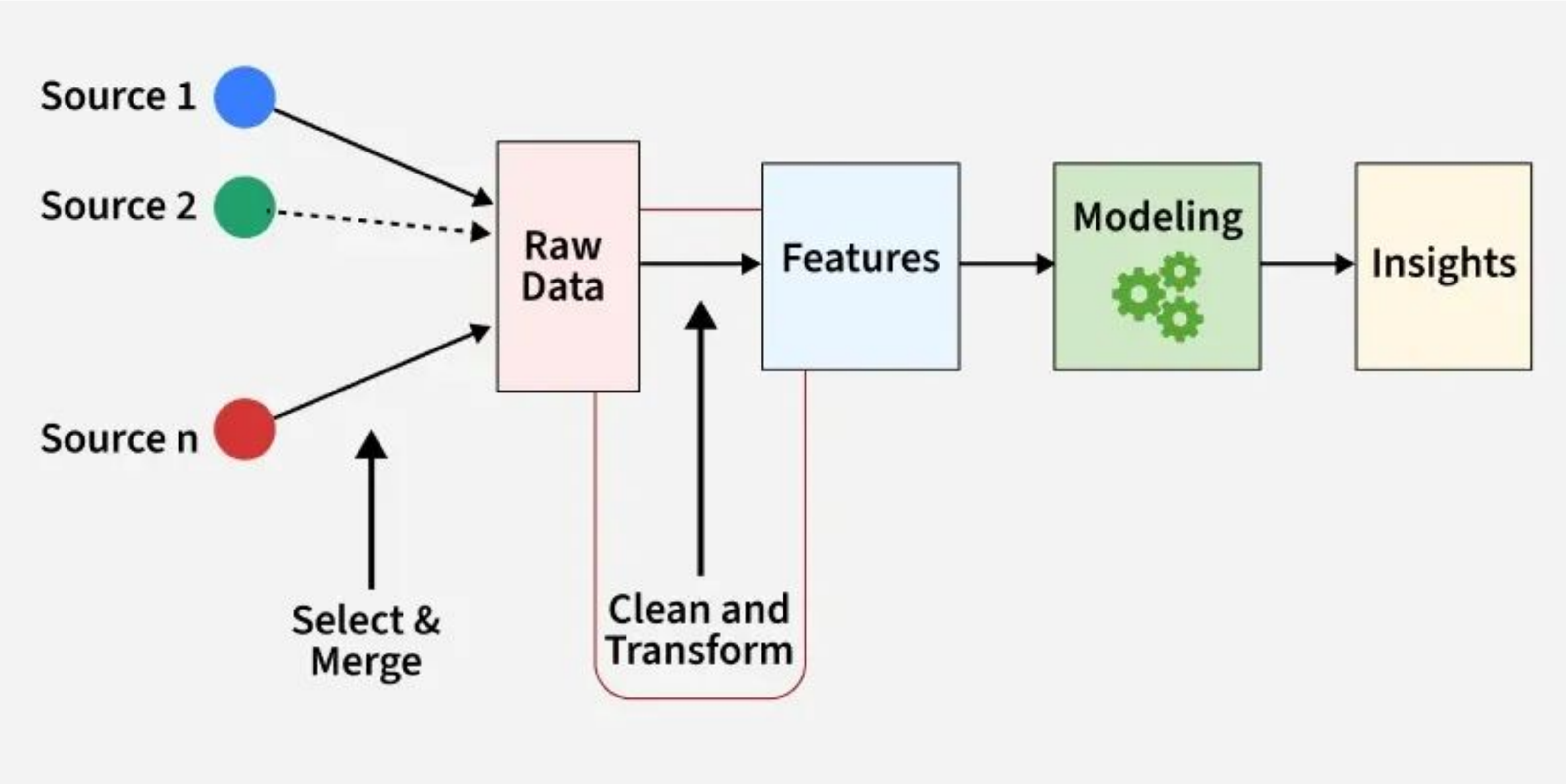

Data
Features



https://www.nature.com/articles/s41524-023-01040-5 https://www.geeksforgeeks.org/machine-learning/what-is-feature-engineering/



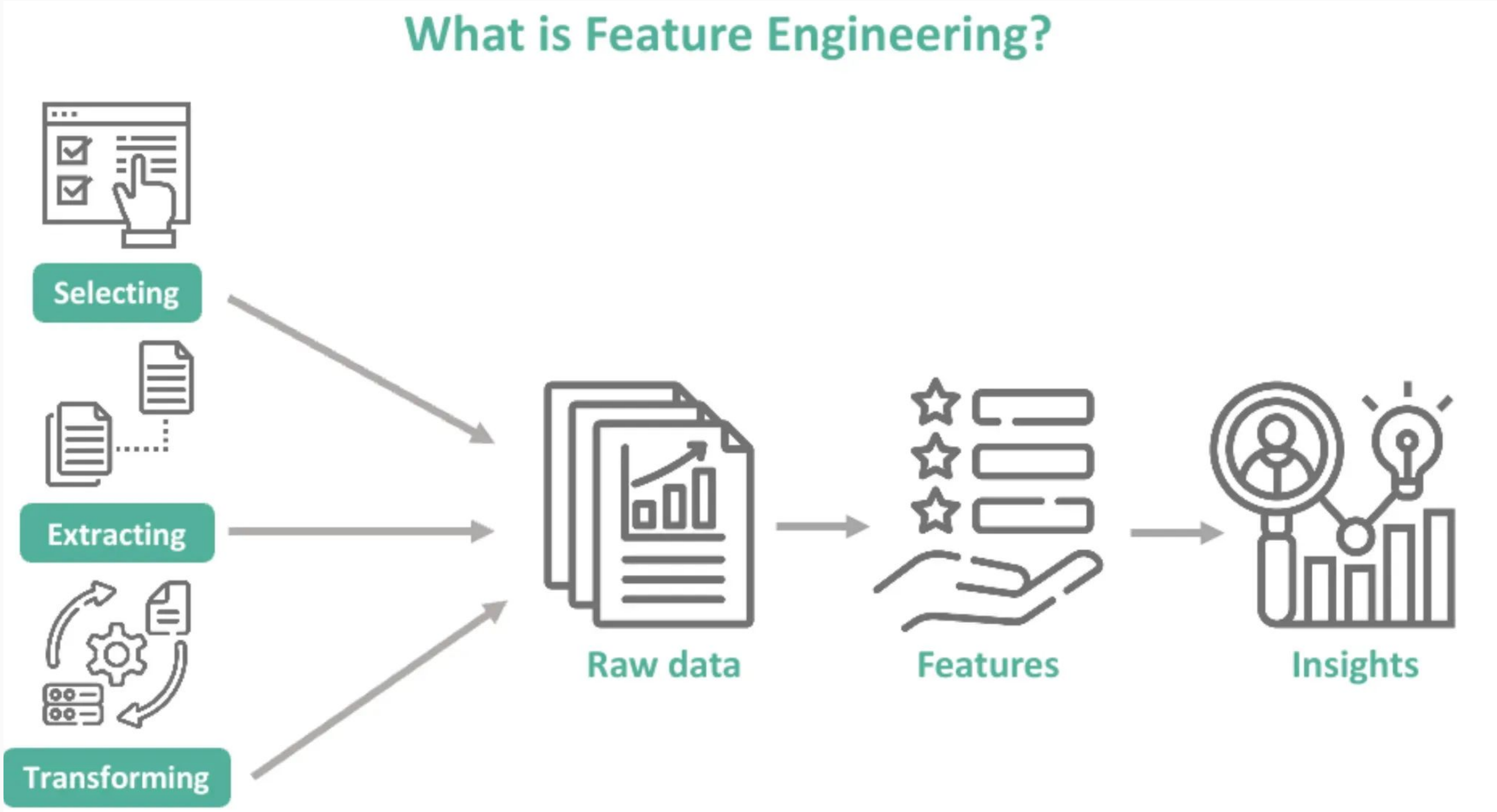
https://www.geeksforgeeks.org/machine-learning/what-is-feature-engineering/



🧠 Why It Matters in AI?

- Reduces noise and improves generalization
- Encodes **domain knowledge** into ML-ready format
- Enables models to “see” patterns better

✅ More relevant features → faster training and better results https://botpenguin.com/glossary/feature-engineering

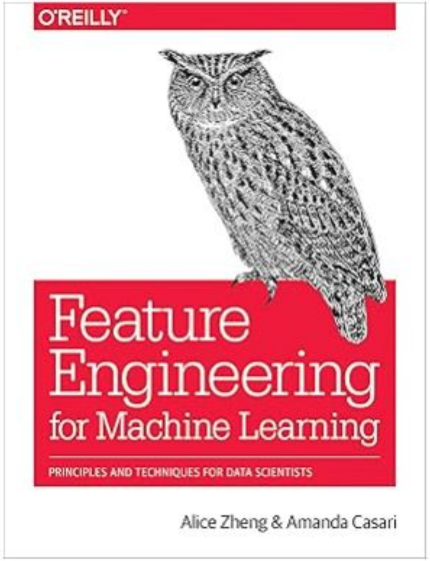


books Categories New & Trending Deals & Rewards Best Sellers & More Memberships Communities More Your Books

Dismiss Change Address

International Kindle Paperwhite Buy Now ›

Books › Computers & Technology › Databases & Big Data › Data Modeling & Design



Read sample

Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists 1st Edition

by Alice Zheng (Author), Amanda Casari (Author)

4.4 ★★★★☆ 81 ratings See all formats and editions

Feature engineering is a crucial step in the machine-learning pipeline, yet this topic is rarely examined on its own. With this practical book, you'll learn techniques for extracting and transforming features—the numeric representations of raw data—into formats for machine-learning models. Each chapter guides you through a single data problem, such as how to represent text or image data. Together, these examples illustrate the main principles of feature engineering.

Rather than simply teach these principles, authors Alice Zheng and Amanda Casari focus on practical application with exercises throughout the book. The closing chapter brings everything together by tackling a real-world, structured dataset with several feature-engineering techniques. Python packages including numpy, Pandas, Scikit-learn, and Matplotlib are used in code examples.

You'll examine:

- Feature engineering for numeric data: filtering, binning, scaling, log transforms, and power transforms
- Natural text techniques: bag-of-words, n-grams, and phrase detection

Read more

Report an issue with this product or seller

📦 Dataset for Today

Fictional Characters Dataset

Use fields like:

- Role, Genre, Personality Traits

- Description, Skills, Interests

Task: Engineer features from this semi-structured dataset for next week's classifier.

📚 Types of Features to Create

Numerical: Counts, lengths, scores

Categorical: Role, Alignment, Age Group

Text-based: Sentiment, Embedding, Named Entities

Combined: Sentiment + Similarity + Age group

🛠️ Example 1: Text Length Feature

python

Copy Edit

```python
df['Description Length'] = df['Description'].apply(lambda x: len(str(x)))
```

Why? Helps quantify verbosity or complexity of the character's profile.

Tip: Always start with basic stats like length and word count.

🛠️ Example 2: Word Count Feature

python

Copy Edit

```python
df['Word Count'] = df['Description'].apply(lambda x: len(str(x).split()))
```

Why? Reflects detail level of the character’s backstory.

Helps normalize or compare text-heavy columns.

🛠️ Example 3: Sentiment Analysis (Hugging Face)

python Copy Edit

```python
from transformers import pipeline
analyzer = pipeline("sentiment-analysis")
df['Sentiment'] = df['Description'].apply(lambda x: analyzer(x)[0]['label'])
```

Goal: Understand tone (positive/negative) of each character.

Use: Input for models that need emotional cues.

🛠️ Example 4: Named Entity Recognition (NER)

python Copy Edit

```python
ner_pipeline = pipeline("ner")
df['Named Entities'] = df['Description'].apply(lambda x: [e['word'] for e in ner_pipeline
```

Extracts named locations, people, etc.

Helps: Generate richer features like presence of "political", "military", "scientific" terms.

🛠️ Example 5: Role Popularity as Feature

python

Copy Edit

```
df['Role Popularity'] = df['Role'].map(df['Role'].value_counts())
```

Why? Helps encode how common a character role is in the dataset.

Can be useful in clustering and classification.

📏 Similarity to a Reference Text

python Copy Edit

```python
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

vectorizer = TfidfVectorizer()
tfidf = vectorizer.fit_transform(df['Description'])
ref_vec = vectorizer.transform(["Hero"])
df['Similarity to Hero'] = cosine_similarity(tfidf, ref_vec).flatten()
```

Use Case: How "heroic" is a description compared to a reference point?

Valuable in semantic analysis.

The Core Idea of TF-IDF

TF-IDF tells us how **important** a word is in a document, compared to **how common** it is in all documents.

- **TF** = how often a word appears in *one* document
- **IDF** = how rare that word is across *all* documents
- Together, they highlight **keywords** that are *unique & meaningful*

📦 Why Use TF-IDF?

Raw text is not usable in ML models

TF-IDF turns text into **numerical features**

Better than just counting words (Bag-of-Words), because it **downweights common words** (like "the", "and")

Helps models **understand important terms**

💬 Simple Example from Our Dataset

Let's take the `Description` of 3 characters:

Character	Description
Jeffrey	"Manager movie last speak rock."
Amanda	"Then truth raise think from seek."
Ryan	"Set wide land within medical."

🧮 Step 1: Term Frequency (TF)

Example for Ryan:

- "Set": 1
- "wide": 1
- "medical": 1
- TF for each = **1/5**

Step 2: Inverse Document Frequency (IDF)

Penalize words that appear in *many* documents

If "rock" appears only in Jeffrey's description → **high IDF**

If "think" appears in all 3 descriptions → **low IDF**

🧠 TF-IDF Score

TF-IDF = TF × IDF

- High if the word is **frequent in this document**, but **rare in others**

- Low if the word is **common across many documents**

📐 Feature: Average Word Length

python Copy Edit

```python
df['Avg Word Length'] = df['Description'].apply(
    lambda x: sum(len(w) for w in str(x).split()) / len(str(x).split()) if len(str(x).spl
```

Why? Sophisticated characters may use longer or more complex language.

Add depth to NLP analysis.

✅ Tips & Tricks for Better Features

- Don’t create too many features—focus on **relevance**
- Avoid data leakage (e.g., using post-event info)
- Combine domain knowledge with exploratory analysis
- Always visualize → correlate → engineer

🚀 Winning with Feature Engineering

Secrets to Standing Out in Kaggle, Hackathons, and Challenges


NETFLIX
2009
DATE: 09·21·09
PAY TO THE ORDER OF: BellKor's Pragmatic Chaos
$ 1,000,000 00
AMOUNT ONE MILLION 00/100
FOR: The Netflix Prize
Reed Hastings
NETFLIX

NETFLIX

Netflix Prize

Home | Rules | Leaderboard | Register | Update | Submit | Download

Leaderboard

Display top 40 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
--	No Grand Prize candidates yet	--	--	--
Grand Prize - RMSE <= 0.8563				
1	PragmaticTheory	0.8584	9.78	2009-06-16 01:04:47
2	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
3	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
4	Dace	0.8604	9.56	2009-04-22 05:57:03

https://www.youtube.com/watch?app=desktop&v=ft77eXtn30Q&pp=ygULI3dpdGhrYWdnbGU%3D

https://towardsdatascience.com/what-does-it-take-to-win-a-kaggle-competition-lets-hear-it-from-the-winner-himself-1d292a9f5b2d/

What are some of the best things you have learned via Kaggle that you apply in your professional work at H2O.ai?

Dmitry: Nobody knows what the best way to solve a problem in the beginning is. Not even halfway through. It is an iterative process of testing, failing, learning from the failure, and repeating.

> "The common approaches and state-of-the-art models usually suffice to achieve impressive results. But if you want to do better, you have to think outside the box. And it is a field with endless opportunities to be creative."

Research Prediction Competition

Indoor Location & Navigation

Identify the position of a smartphone in a shopping mall

$10,000

Prize Money

Microsoft Research · 1,170 teams · 21 days ago

Overview Data Code Discussion Leaderboard Rules

Late Submission

Public Leaderboard Private Leaderboard

The private leaderboard is calculated with approximately 85% of the test data.

This competition has completed. This leaderboard reflects the final standings.

Refresh

In the money Gold Silver Bronze

#	△pub	Team Name	Notebook	Team Members	Score	Entries	Last
1	—	Track me if you can			1.49717	322	21d
2	—	MYRCJ			2.19892	513	22d
3	▲ 2	Zidmie			2.48923	476	21d

💡 Why Feature Engineering is Your Secret Weapon

- Everyone has access to the same dataset
- **Modeling gives diminishing returns** after a point
- Top solutions often win because of **creative feature ideas**
- Feature engineering adds **insight**, **context**, and **structure**

"Great features make average models look genius."

🏆 How to Win a Kaggle or Hackathon

1. **Understand the domain deeply**
   What are the real-world meanings behind each field?
2. **Create features no one else thought of**
   Go beyond the obvious
3. **Stack many weak features**
   Ensembling isn't just for models—also for features
4. **Automate exploration**
   Use `.corr()`, `.value_counts()`, SHAP, or permutation importance to find gold

🧠 Core Idea: What Makes a Feature "Genius"?

✅ It **extracts hidden patterns**
✅ It **simplifies complexity**
✅ It **captures relationships** others overlook
✅ It **aligns with real-world logic** or psychology
✅ It's **hard to discover** without creativity

🛠️ Examples of Genius Feature Ideas

Idea	Example	Why It Wins
Sentiment of Backstory	`pipeline("sentiment-analysis")` on `Backstory`	Emotion reflects role
Named Entity Count	Count of locations or people in `Description`	Complexity of narrative
Role Impact Score	Frequency of `Role` * Sentiment score	Balance of popularity and tone
Thematic Density	Count how often fantasy/science keywords appear	Matches to Genre
Similarity to Archetypes	TF-IDF or BERT similarity to "hero", "villain", etc.	Semantic relevance

🔥 Advanced Feature Engineering Moves

- **Word embeddings (BERT, FastText)** to convert text into dense vectors
- **Graph features** if relationships exist (e.g., who mentors who)
- **Polynomial or interaction terms** between important variables
- **Clustering (KMeans) + Label** = Group ID features
- **Text summarization → sentiment → scale**: multi-layer extraction

🧪 How to Create a Cool Feature from Scratch

1. Ask: “What **hidden pattern** is missing from this data?”

2. Think **like a detective**: what's behind the text?

3. Break complex data into **tokens, counts, scores, groups**

4. Use **pre-trained models** to inject external intelligence (e.g., GPT, transformers)

5. Visualize the feature's distribution → is it informative?

🔧 Technical Framework for Creating Cool Features

python

Copy Edit

```python
# General recipe
def create_feature(df):
    df['New Feature'] = (
        df['Field A'].apply(transform_logic)
        + df['Field B'].apply(some_score)
        * df['Sentiment'].map(sentiment_score_map)
    )
    return df
```

- Always test with `.groupby(target).mean()` or `.corr()`
- Plot it with seaborn or use feature importance tools (e.g., SHAP)

🎯 Tips to Outperform in Competitions

✅ Focus on **data understanding**, not just models
✅ Explore edge cases or rare patterns
✅ Use **target encoding**, **binarization**, **clustering**
✅ Ensemble both **models** and **feature sets**
✅ Track your experiments & **version** your data

Pro Tip: Train multiple models on **different feature views**

🧪 What 1st Place Solutions Often Do

Spend **80% of the time on feature engineering**

Use **feature importance** to drop noise

Create **custom loss functions** based on feature behavior

Build **meta-features**: features of other features

Document every transformation → reproducibility = bonus points

✨ What Makes You Stand Out

Don't repeat what everyone else does

Be the **first to discover a pattern**

Feature engineering is where **you bring yourself** into the model

Add **human sense** to machine learning

🧠 Takeaway Mindset

- Model performance isn't luck—it's **insight**
- Raw data is clay. Your features are the sculpture.
- Great ML starts with asking the right questions, not using the fanciest algorithm.

📘 Challenge for You This Week

✅ Choose 5 creative features

✅ Explain *why* they add value

✅ Visualize each one

🏆 Bonus: Try one external model (e.g., Hugging Face) in your feature pipeline