Overview of Feature Engineering

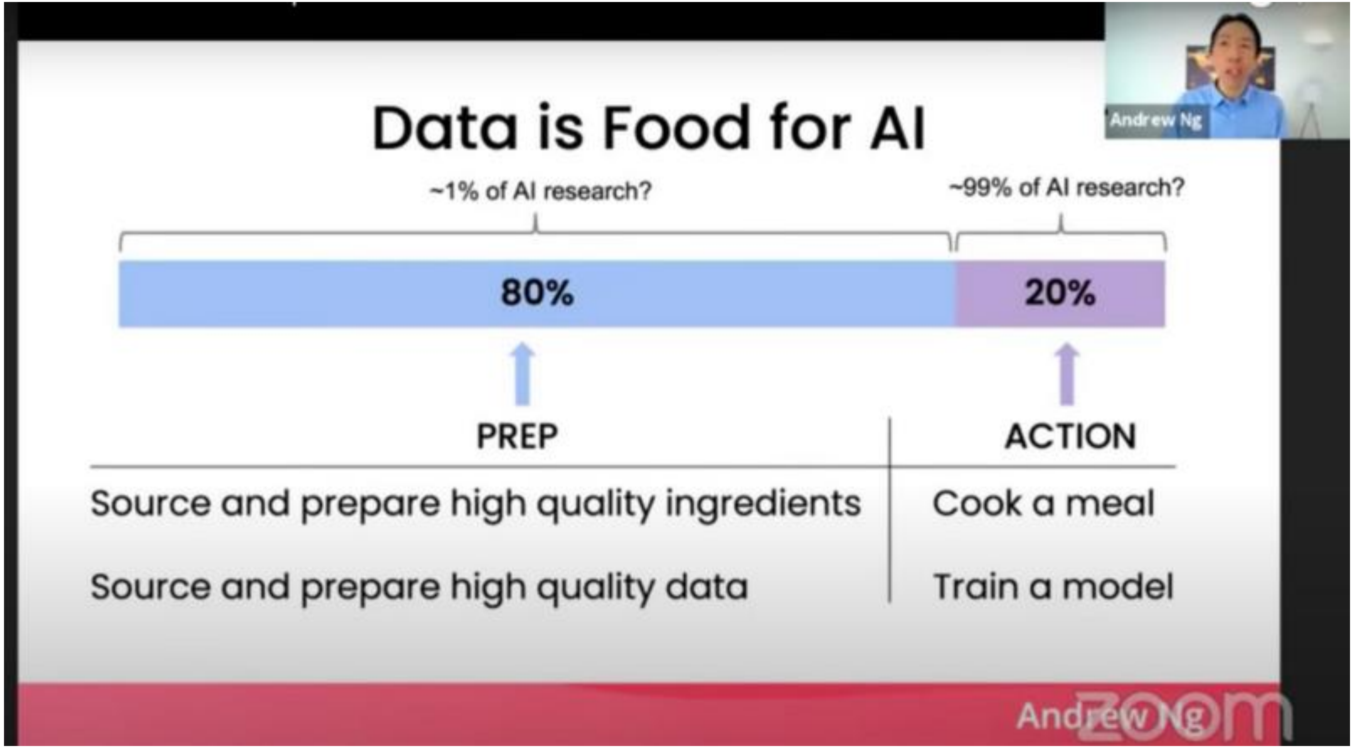
knowledgehut upGrad

Learning Objectives

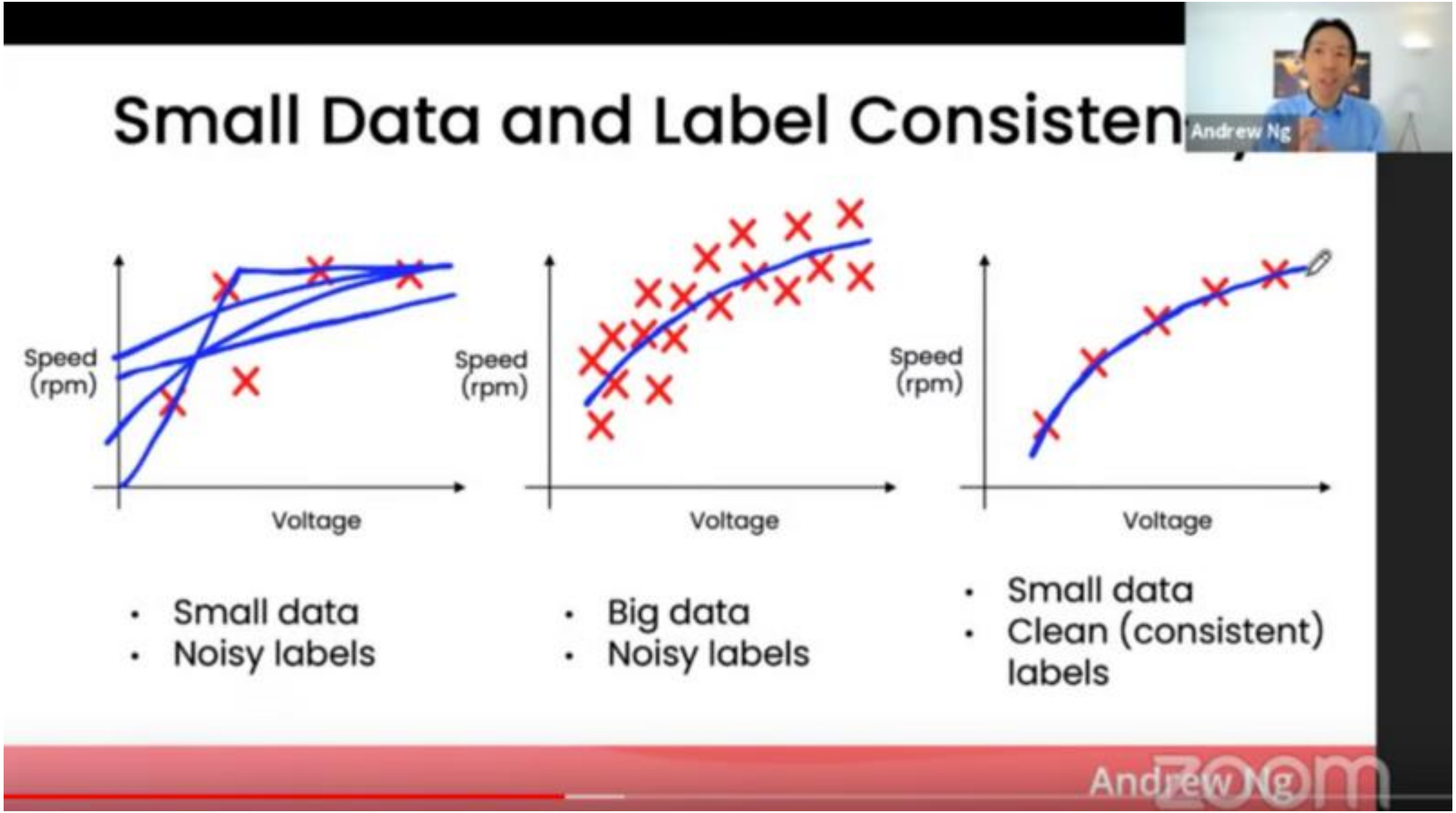
- Explain the basics of feature engineering
- Discuss the feature scaling
- Describe the interaction effect and the dummy variable
- Explain the feature selection

Feature Engineering

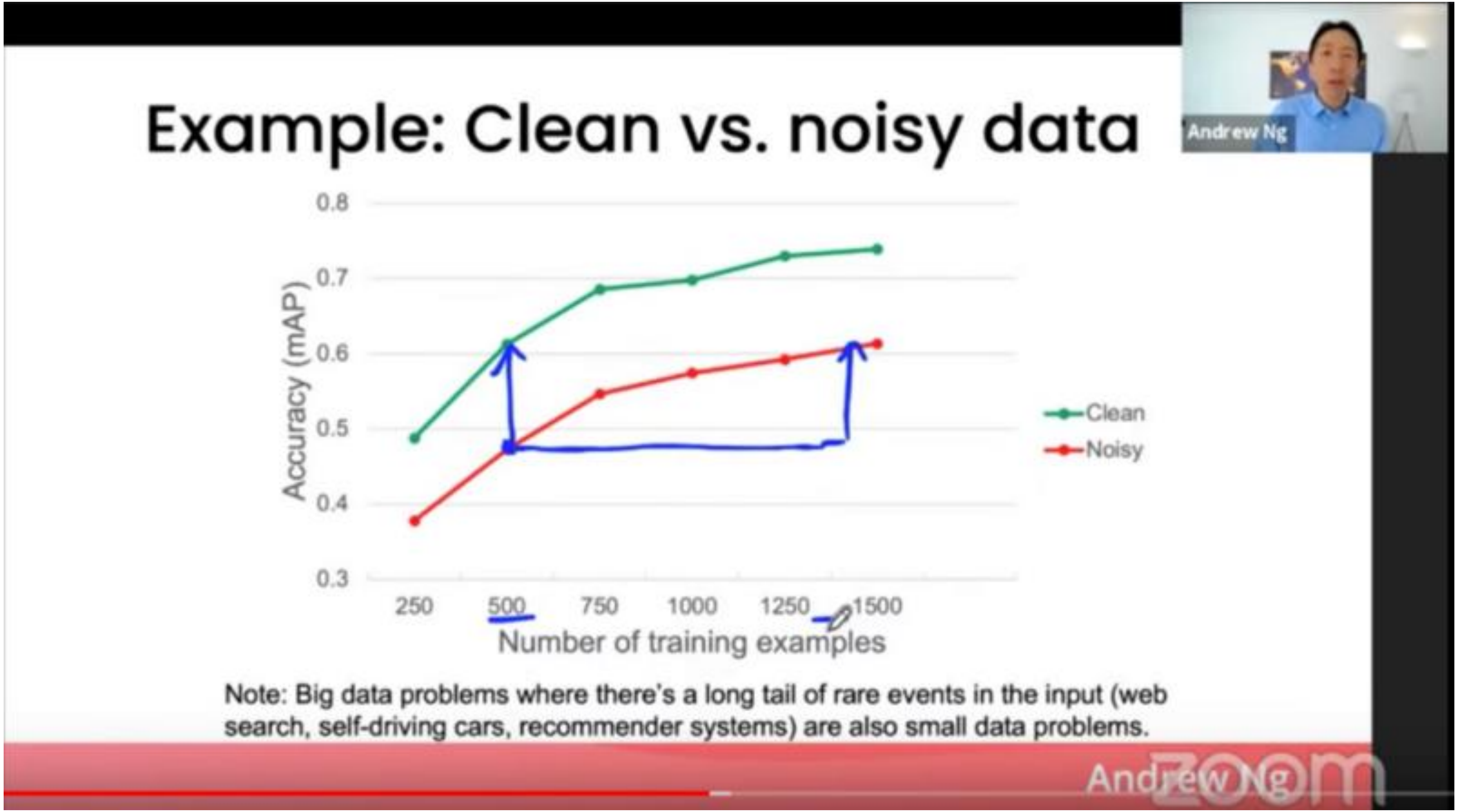
Why Feature Engineering ?



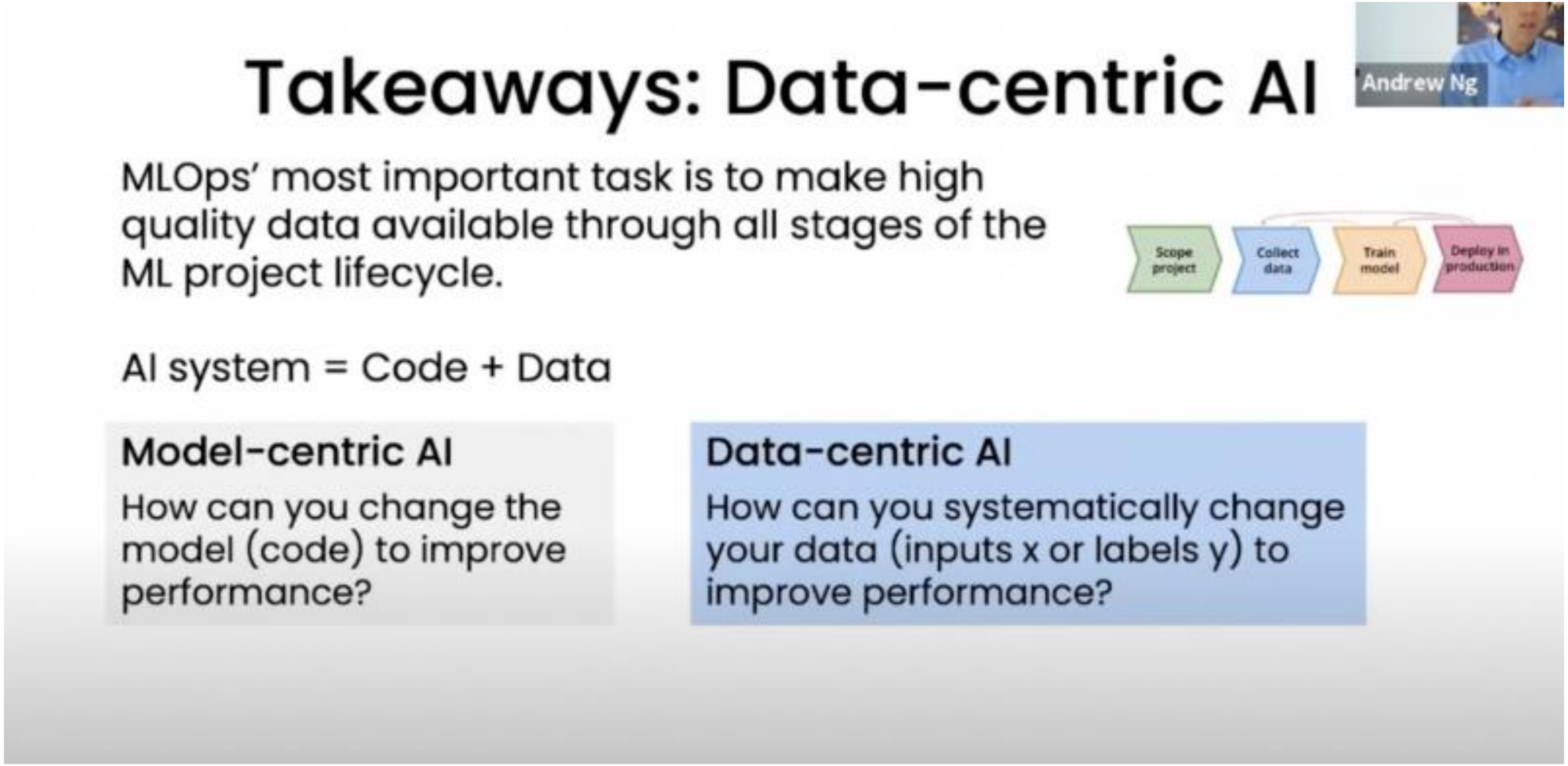
Why Feature Engineering ?



Why Feature Engineering ?

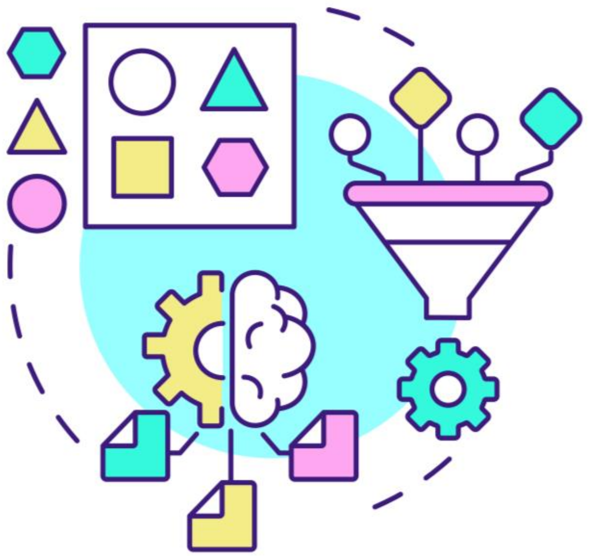


Why Feature Engineering ?



Feature Engineering

- The process of turning raw data into useful features for better ML models.
- Having the right features is the most important thing in ML modeling.
- Feature engineering provides the best ROI of time (compared to algorithmic tweaking).
- It is an essential skill for every ML practitioner.



Feature Transformation

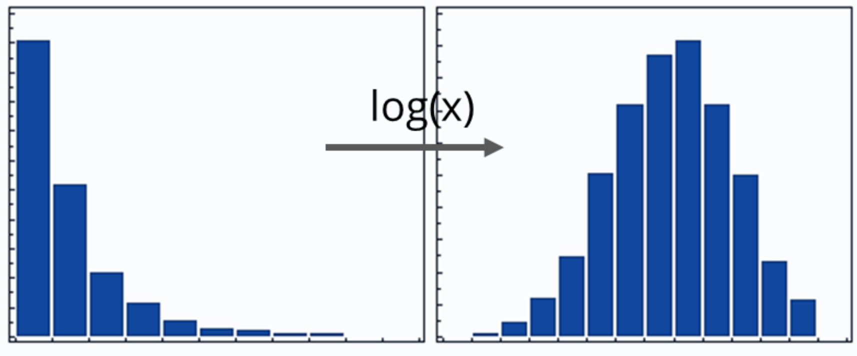
- There are various techniques for feature engineering, and one of the most common is a technique called feature transformation.

- Skewed data distributions can be problematic for ML algorithms. So transformation is applied to mitigate this.

- A typical example of this is the **Log transformation**.

- It helps to make uniform distribution and linear relationships between features and targets.

- Formula: log(x).



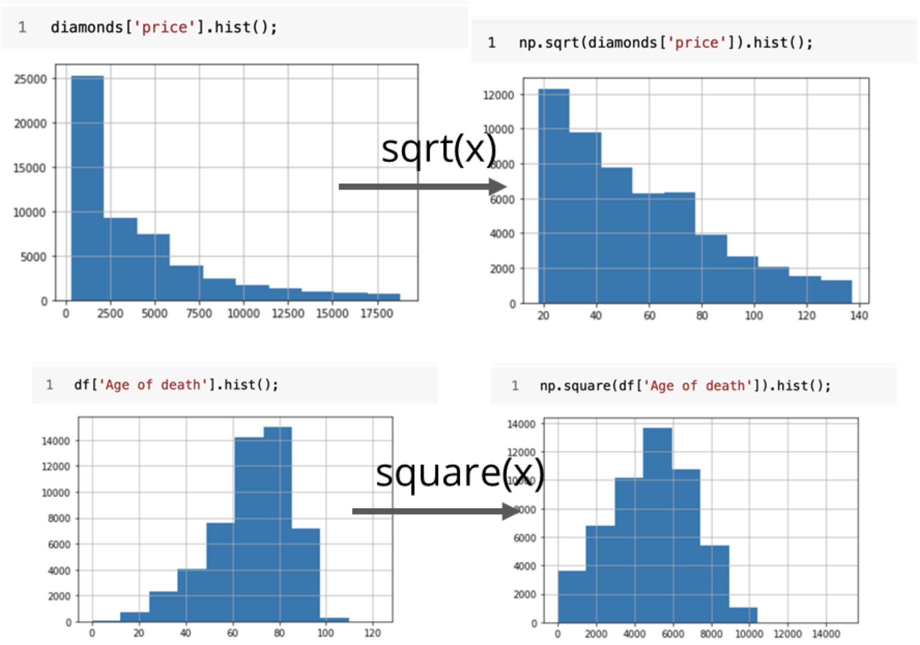
Other Feature Transformation

Square Root
- For right-skewed data, e.g., income distribution, age, height, and weight.

Reciprocal
- Reverses the order among values of the same sign (large values become smaller and vice-versa).

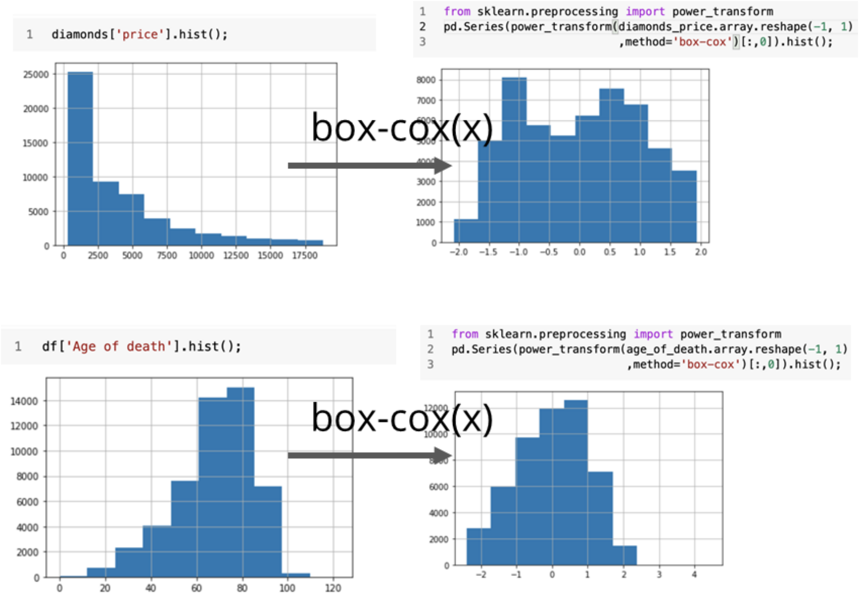
Exponential
- For left-skewed data, e.g., age of death.

Box-Cox



Box-Cox Transformation

- A box-cox transformation is a so-called power transformation.
- Finds the ideal exponent to make data look Gaussian-like.
- It makes distribution look uniform.
- It works well on both right-skewed and left-skewed data.



Feature Scaling

Importance of Feature Scaling

- ML model does not understand that different features represent different things.
- The model might give higher importance to larger values.
- Apply scaling before turning features into models.
- It mitigates risk, especially for classical models.

Side effect:

- It interprets coefficients as feature importance.

Common Approaches in Scaling (Normalization)

Fit the variables in the range of **[0, 1]**
- 0 is the lowest.
- 1 is the highest.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Fit variables in the custom range **[a, b]**
- In the range of [-1, 1].

$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)}$$

Common Approaches in Scaling (Standardization)

- Fit variables with zero mean and unit variance.
- It is for normally distributed data.

$$x' = \frac{x - \bar{x}}{\sigma}$$

Important Notes on Scaling

knowledgehut upGrad

- Global statistics, such as min, max, mean, etc., must be calculated on the training set and made available on the test set or during prediction, respectively (new data).

- If the data distribution changes significantly, the statistics must be recalculated on the training set, and the model must be retrained.

- Scale the data after splitting to avoid target leakage.

Interaction Effect and Dummy Variable

Why Encode Categorical Data?

- Most ML models essentially multiply weights with features, and numeric values are required for this.

- For the classical algorithm, encode categorical data into a matrix form, i.e., to numeric values.

There are essentially three approaches:

- Label encoding
- Dummy encoding
- One-Hot encoding

Label Encoding

knowledgehut upGrad

- It assigns a numeric value to each category of a variable.
- It becomes problematic for nominal data as ML algorithms use the resulting numbers on a ratio scale.
- Label encoding works well for ordinal and binary data.

Example:
Variable “Color” with categories [Red, Not Red]

Color
Red
Not Red
Not Red
Not Red
Red
…

Color
0
1
1
1
0
…

One-hot Encoding

- One-hot encoding gets the same number of features as the number of categories for a variable.
- So, there would be three columns in the red, green, and blue examples.

Example:
Variable "Color" with categories [Red, Blue, Green]

Color
Red
Blue
Green
Blue
Red
…

→

Red	Blue	Green
1	0	0
0	1	0
0	0	1
0	1	0
1	0	0
…	…	…

Dummy Variable Encoding

- Dummy variable encoding follows an approach that will give n-1 categories, so instead of three columns, there will be only two columns.
- Dummy encoding removes a duplicate category in each categorical variable. This avoids the dummy variable trap.
- Dummy encoding works well if the set of categories is naturally fixed.
- However, in practice, it gets trickier if categories can change – and this happens quite often.

Example:
Variable "Color" with categories [Red, Blue, Green]

Color
Red
Blue
Green
Blue
Red
…

→

d1	d2
1	0
0	1
0	0
0	1
1	0
…	…

Dummy Encoding Examples (Approach 1)

Consider the example of ranking products for an e-commerce store, and you want to use the product brand as one feature.

Approach 1: Ignore

- The model will break as it cannot encode new categories.

Price	Brand	Rank
4.99	Acme	1
12.99	CoCo	2
5.99	Acme	3
5.99	BestB	4
9.99	DNB	5
...	...	...

Price: 6.99
Brand: NewB

Dummy Encoding Examples (Approach 2)

knowledgehut upGrad

Approach 2: Dummy 'unknown'

- The model will not break but will not predict anything for new brands.

Price	Brand	Rank
4.99	Acme	1
12.99	CoCo	2
5.99	Acme	3
5.99	BestB	4
9.99	DNB	5
...	...	...

Prediction

Price: 6.99
Brand: unknown

Dummy Encoding Examples (Approach 3)

Approach 3: Dummy ‘other’

- Brand has a chance to rank, but both new and small brands are treated the same.

Price	Brand	Rank
4.99	Acme	1
12.99	CoCo	2
5.99	Acme	3
5.99	Other	4
9.99	Other	5
...	...	...

Prediction

Price: 5.99
Brand: other

Curse of Dimensionality

- Encoding categorical data as additional features adds additional complexity in the form of additional dimensions to the model.
- **Curse of dimensionality:** As data becomes more sparse, calculations become more unstable, and model performance gets worse.

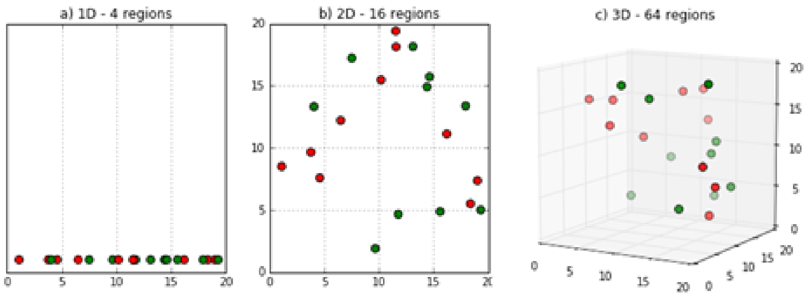


Image credits: DeepAI

Interaction Effect

- The effect of one feature on the target variable is dependent on the values of another feature.
- For example, to predict the probability of a person buying a house based on annual salary, marital status, and the number of kids.

Feature Crossing

- It helps capture the non-linear relationships in the data or put more domain knowledge into the training data.

Feature Selection

Features in an ML Model

Adding more features to the model leads to the following:

- Increase in model accuracy.
- Poor generalization or risk of overfitting.
- Technical debt on the data pipeline.

When the data pipeline changes, the affected features must be adjusted.

Multiple features make the data pipeline unnecessarily complex.

Features Selection Strategies

knowledgehut upGrad

Theory

Regularization can shrink feature importance to 0.

Practical

It’s better to remove unnecessary features directly.

The process of selecting a subset of features in machine learning is called **feature selection.**

Feature Selection Methods

knowledgehut upGrad

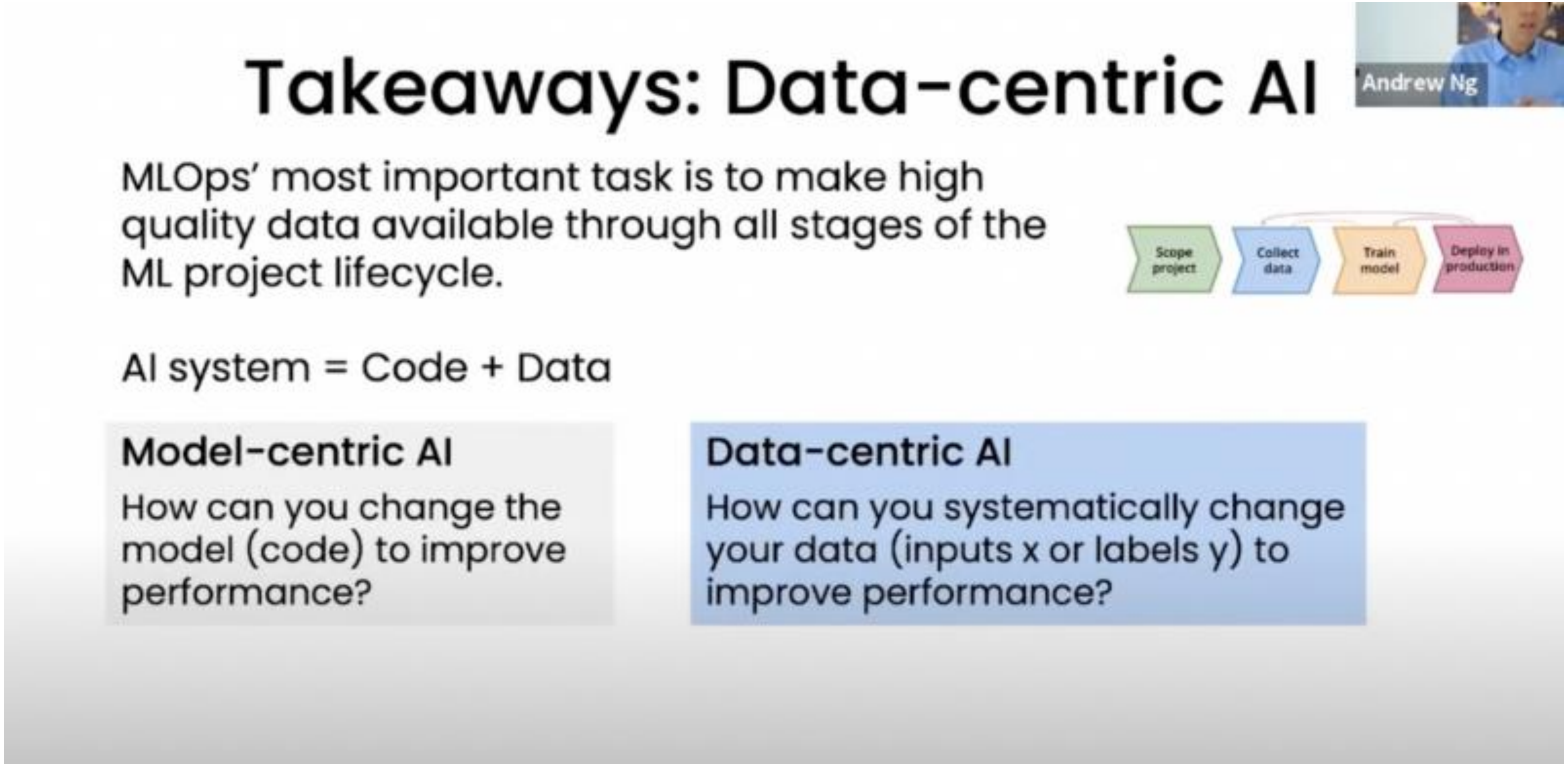
Forward Selection

- It starts with an empty feature set.
- It adds new features step-by-step.
- It checks the performance of the algorithm.
- Stops adding features when algorithm performance hits maximum.
- **Drawback:** It can get stuck in the local minimum.

Backward Elimination

- Backward elimination is a method of feature selection that starts with all features included and removes them one at a time until none feature meets the criterion or the desired number of features is reached.
- This process is also called Recursive Feature Elimination (RFE).

Key Message


Takeaways: Data-centric AI
Andrew Ng
MLOps' most important task is to make high quality data available through all stages of the ML project lifecycle.
Scope project
Collect data
Train model
Deploy in production
AI system = Code + Data
Model-centric AI
How can you change the model (code) to improve performance?
Data-centric AI
How can you systematically change your data (inputs x or labels y) to improve performance?

Thank you