

# Data Wrangling

**Vishal Patel**

Spring 2023

# Course Outline

1. Introduction

2. The Data Science Process

3. Supervised Learning

4. Unsupervised Learning

**5. The Grunt Work**

6. Closing Thoughts

# Model Building Process

1. **Assemble** a modeling sample.
2. **Create** the target variable.
3. **Create** attributes.
4. **Explore** data (EDA).
5. **Transform** attributes (trim, impute, etc.).
6. **Select** attributes (aka Feature Selection).
7. **Train** and **test** the model.
8. **Select** the best model.
9. **Validate** the model.
10. ...

DATA PREPARATION

DATA WRANGLING

DATA MUNGING

FEATURE ENGINEERING

# Summary Statistics

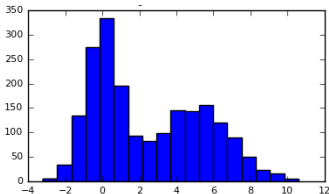
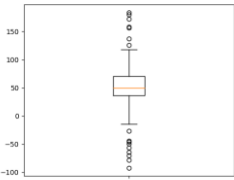
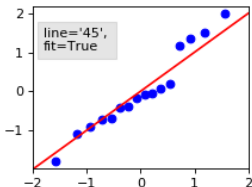
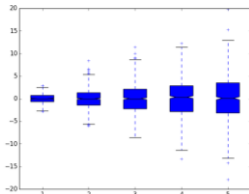
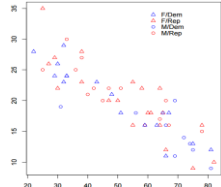
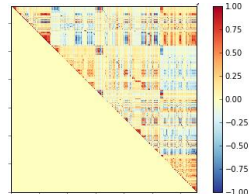
For each **numeric** variable in the training data set, calculate the following statistics:

1. Mean,
2. Median,
3. Mode,
4. Min and max,
5. Standard deviation,
6. Percentile values (1<sup>st</sup>, 2<sup>nd</sup>, 98<sup>th</sup>, 99<sup>th</sup>),
7. Cardinality (number of distinct values),
8. Kurtosis, and
9. Percent missing values (% of training data set that contain missing ).

For each **categorical** variable in the training data set, calculate the following:

1. Cardinality (number of distinct values)
2. Count of records (rows) within each level (value) of the categorical variable,
3. Percent of records within each level, and
4. Target even rate for each level.

# Exploratory Data Analysis (EDA)

	Univariate	Multivariate
Non-Graphical	<ul style="list-style-type: none"> <li>○ Categorical: Tabulated frequencies</li> <li>○ Quantitative: <ul style="list-style-type: none"> <li>○ Central tendency: mean, median, mode</li> <li>○ Spread: Standard deviation, inter-quartile range</li> <li>○ Skewness and kurtosis</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>○ Cross-tabulation</li> <li>○ Univariate statistics by category</li> <li>○ Correlation matrices</li> </ul>
Graphical	<ul style="list-style-type: none"> <li>○ Histograms</li> <li>○ Box plots, stem-and-leaf plots</li> <li>○ Quantile-normal plots</li> </ul> <div>    </div>	<ul style="list-style-type: none"> <li>○ Univariate graphs by category (e.g., side-by-side box-plots)</li> <li>○ Scatterplots</li> <li>○ Correlation matrix plots</li> </ul> <div>    </div>

# Trimming

- Variable trimming can help improve model stability by reducing the influence of extreme values the data.
- To decide which variables contain extreme values, you can use **kurtosis**. A high kurtosis value suggests **flat tail(s)** in the distribution.
- Steps for trimming:
  1. From the descriptive statistics of all numeric variables, identify variables that have a kurtosis value **higher than 10<sup>†</sup>**, and apply trimming and/or capping on those variables as explained in the next steps.
  2. **Capping**: Replace extremely high values – i.e., values that exceed the variable's **99<sup>th</sup> percentile<sup>†</sup>** value – with the **99<sup>th</sup> percentile<sup>†</sup>** value.
  3. **Flooring**: If the variable contains extreme negative values, and if a negative value exceeds the variable's **1<sup>st</sup> percentile<sup>†</sup>** value, then replace the extreme negative value with the **1<sup>st</sup> percentile<sup>†</sup>** value.

<sup>†</sup> These are suggested threshold values; change these values as needed.

# Missing Values

1. Some statistical methods and data mining algorithms (e.g., linear regression, Support Vector Machines) are **unable to handle** missing values.
2. Missing data introduces **complications** in analyzing and processing (imputing) missing values.
3. Improper handling of missing values may introduce **bias** in the analysis.

# Reasons for Missing data

## 1. Not applicable

- Data usage for telecom customers who are voice-only users
- Loan balance amount for a customer who never had a loan

## 2. Not reported or provided

- Information not provided in the application form (e.g., race, ethnicity)

## 3. Undefined

- Division by zero

## 4. Not available

- Unknown reason

Sometimes missing data are denoted by specific (usually extreme) values. Each value indicates a different reason for why the data is missing.

**Example:** Total number of trades with derogatory occurred in the last 12 months including collections:  
{0-90: valid values, 99: No trade including collections, 97: No trade reported within timeframe required}



# Missing Data Mechanisms

## Random

Missing Completely at Random (MCAR)

Missing at Random (MAR)

## Non-random

Missingness depends on unobserved predictors

Missingness depends on missing value itself

# Missing Completely at Random (MCAR)

The reason for missing values  
is **completely independent**  
of all other features.

In other words,  
the probability of missing value  
is the same for all records in the dataset.

**This is an unlikely scenarios for most real-world datasets.**

# Missing (Conditionally) at Random (MAR)

Missingness is **dependent** on  
other available features.

**In practice, this is hard to know.**

**If you know which features the missingness is dependent on<sup>†</sup>,  
then including those features in the model  
would eliminate the potential bias.**

<sup>†</sup> One could model this process as a logistic regression model, where the target variable is 1 for missing cases and 0 otherwise.

# Missing Not at Random: MNAR I

Missingness is **dependent** on  
unobserved (unavailable) features.

E.g., if a particular medical treatment causes discomfort,  
a patient is more likely to drop out of the study.

This missingness is not random,  
unless the level of “discomfort” is measured and observed for all patients.

**If missingness is not at random, it must be explicitly modeled,  
or else you must accept some bias in your inferences.**

# Missing Not at Random: MNAR II

Missingness is **dependent** on *itself*.

E.g., people with high income are less likely to report their income.

**If missingness is not random and is dependent on itself,  
it can be explicitly modeled to reduce some of the bias.**

# Complete-case analysis

	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	
1	5	3	4	4		✗
2	3	1	2	3	3	
3	4		4	3	5	✗
4	3	3	1	5	4	
5	1	5	5	2	4	

A direct and naïve approach is to exclude records that contain missing value(s).

**NOT RECOMMENDED!**

**Instead, impute missing values.**

# Mean (or Median) Imputation

	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>
1	5	3	4	4	4
2	3	1	2	3	3
3	4	3	4	3	5
4	3	3	1	5	4
5	1	5	5	2	4

Impute missing values with the **mean** (or **median**) of the observed values of that variable..

## POTENTIAL ISSUES?

- Distorts the distribution of the imputed variable
- Usually, reduces the correlation with the target variable

# Imputation Using Other Features

	<i>Square Footage</i>	<i># of Bedrooms</i>
1	1,019	3
2	3,273	4
3	1,162	3
4	1,106	3
5	1,806	4

Impute missing values with the observed value from another, similar record (“hot deck” imputation).

## POTENTIAL ISSUES?

- May propagate measurement errors
- May not be the optimal, “correct”, imputation value



# Multiple (Multivariate) Imputation

	<i>A</i>	<i>C</i>	<i>D</i>	<i>E</i>
1	5	4	4	?
2	3	2	3	3
3	4	4	3	5
4	3	1	5	4
5	1	5	2	4

## POTENTIAL ISSUES?

- Computationally intensive
- Dependent on how accurate the models are
- Difficult to put into production

$$X = \begin{matrix} & \begin{matrix} A & C & D \end{matrix} \\ \begin{pmatrix} 3 \\ 4 \\ 3 \\ 1 \end{pmatrix} & \begin{pmatrix} 2 \\ 4 \\ 1 \\ 5 \end{pmatrix} & \begin{pmatrix} 3 \\ 3 \\ 5 \\ 2 \end{pmatrix} \end{matrix} \quad y = \begin{matrix} & E \\ \begin{pmatrix} 3 \\ 5 \\ 4 \\ 4 \end{pmatrix} \end{matrix}$$

## Step 1

Build a model (e.g., regression) with all features that contain non-missing data as predictors and the feature that contain missing data as the target feature.

## Step 2

Use this model to predict the missing value(s) for the target feature and use those predictions for imputation.

# Missing Value Indicators

	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>B_Miss</i>	<i>E_Miss</i>
1	5	3	4	4	4	0	1
2	3	1	2	3	3	0	0
3	4	3	4	3	5	1	0
4	3	3	1	5	4	0	0
5	1	5	5	2	4	0	0

Create missing value indicators for each variable that contain missing values: 1 if missing, 0 otherwise.

# Autonomous Race Car Starts Test Lap, Immediately Slams Into Wall

And we mean immediately.

BY PETER HOLDERITH OCTOBER 29, 2020



[W]hile the team did code in many fail-safes in other areas of the application, it unfortunately only contained data validation on valid numbers—and as you recall, an **NaN** value is not a valid number, meaning that validation would not be performed on it.

# Standardize Features

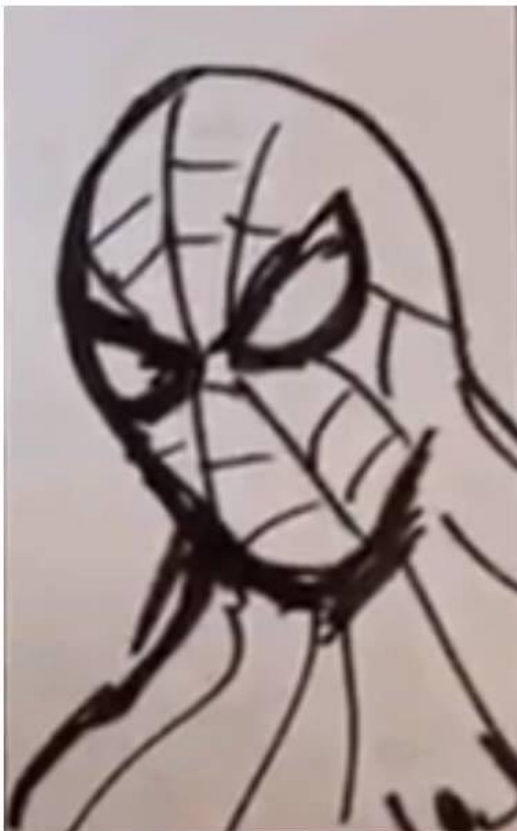
- Attribute values are adjusted for differing level and spread.
  - E.g., standard (z) score: subtract the mean and divide by standard deviation
- Standardization does not impact the shape of the distribution.
- Standardization makes the interpretation of coefficients easier in regression models.

# Box-Cox Transformations

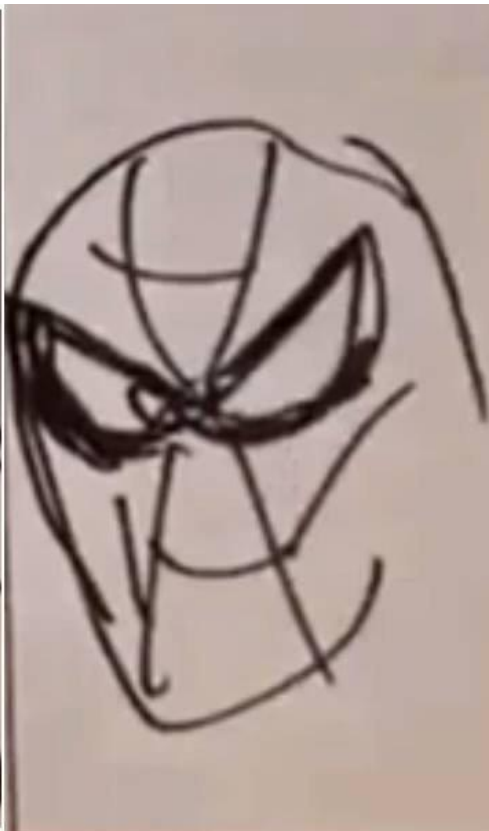
- Transformations may improve the linear correlations with the target
- **Examples:** Log, inverse, root, square root
  - **Log:** Reduces right-skewness. Can be useful for diminishing-return type relationships.
  - **Inverse:** Drastic impact on distribution shape. Sometimes it improves interpretation (e.g., 'purchase frequency per year' becomes 'days between purchases')
  - **Root:** Reduces right-skewness. Unlike 'log', roots can be applied to zero.
  - **Square:** Reduces left-skewness. Can be used to fit a model with quadratic relationship, e.g.,  $y = \alpha + \beta_1 x + \beta_2 x^2$ .



**Kaggle Local  
Cross-validation**



**Kaggle Public  
Leaderboard**



**Kaggle Private  
Leaderboard**



**Real World  
Unseen Data**

## Some reasons why models fail

1. Not standardizing the features or improper standardization
2. Improper handling of missing values
3. Insufficient handling of categorical data
4. **Lack of feature reduction (or selection)**
5. Wrong feature selection approach
6. Information leakage
7. Using the wrong training sample
8. Ignoring class imbalance
9. Flawed target definition
10. Incorrect usage (or lack of) hold-out sample
11. Using the wrong accuracy measure to select the best model
12. Lack of Exploratory Data Analysis (EDA)
13. Lack of consultative approach

# Feature Reduction

- **Feature Reduction:** The process of selecting a subset of features to be used in model construction
- Pre-processing of data in machine learning
- Useful for both supervised and unsupervised learning problems

**Art is the elimination of the unnecessary.**

– Pablo Picasso



# Why Not This?

1

```
import pandas as pd

X = pd.read_csv('C:\train.csv')
y = X['target']
```

2

```
from sklearn.linear_model import LogisticRegression

clf = LogisticRegression().fit(X.pop('target'), y)
```

**The kitchen-sink approach**

# Because...

- **True dimensionality** << **Observed dimensionality**

- Redundant and irrelevant features are in abundance.

- **Curse of dimensionality**

- With a fixed number of training samples, the predictive power reduces as the dimensionality increases. (Hughes phenomenon)

- **Goal of the Analysis**

- Descriptive → Diagnostic → Predictive → Prescriptive

Hindsight

Insight

Foresight

- **Law of Parsimony** (Occam's Razor)

- Other things being equal, simpler explanations are generally better than complex ones.

- **Overfitting** (The Bias-Variance trade-off)

- **Execution time** (Algorithm and data processing)

- **Points of failure**

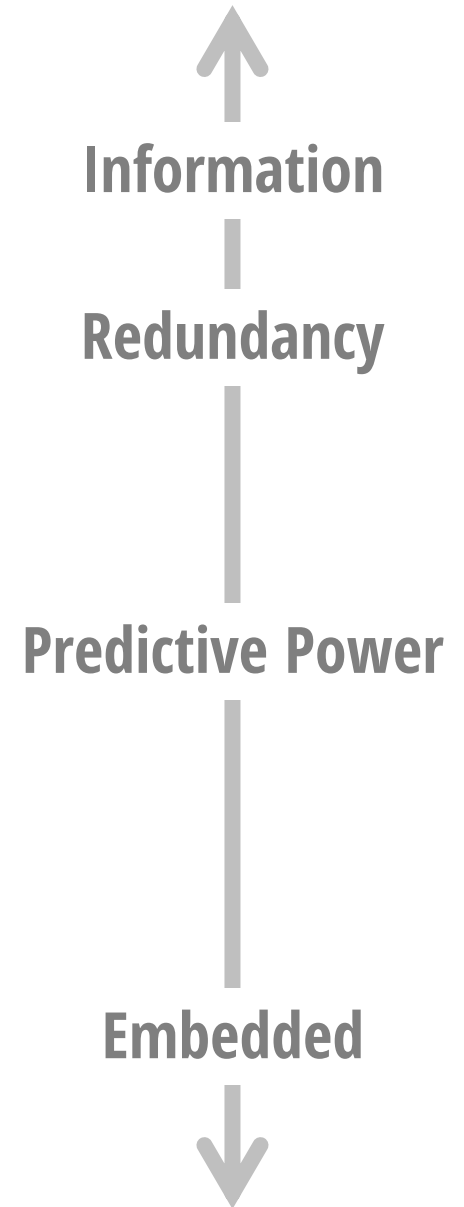
# Feature Reduction Techniques

1. Amount of variation
2. Percent missing values
3. Pairwise correlation
4. Multicollinearity
5. Principal Component Analysis (PCA)
6. Correlation (with the target)

## Filtering Approaches

7. Forward selection
8. Backward elimination
9. Stepwise selection
10. LASSO
11. Tree-based selection

## Wrapper Approaches



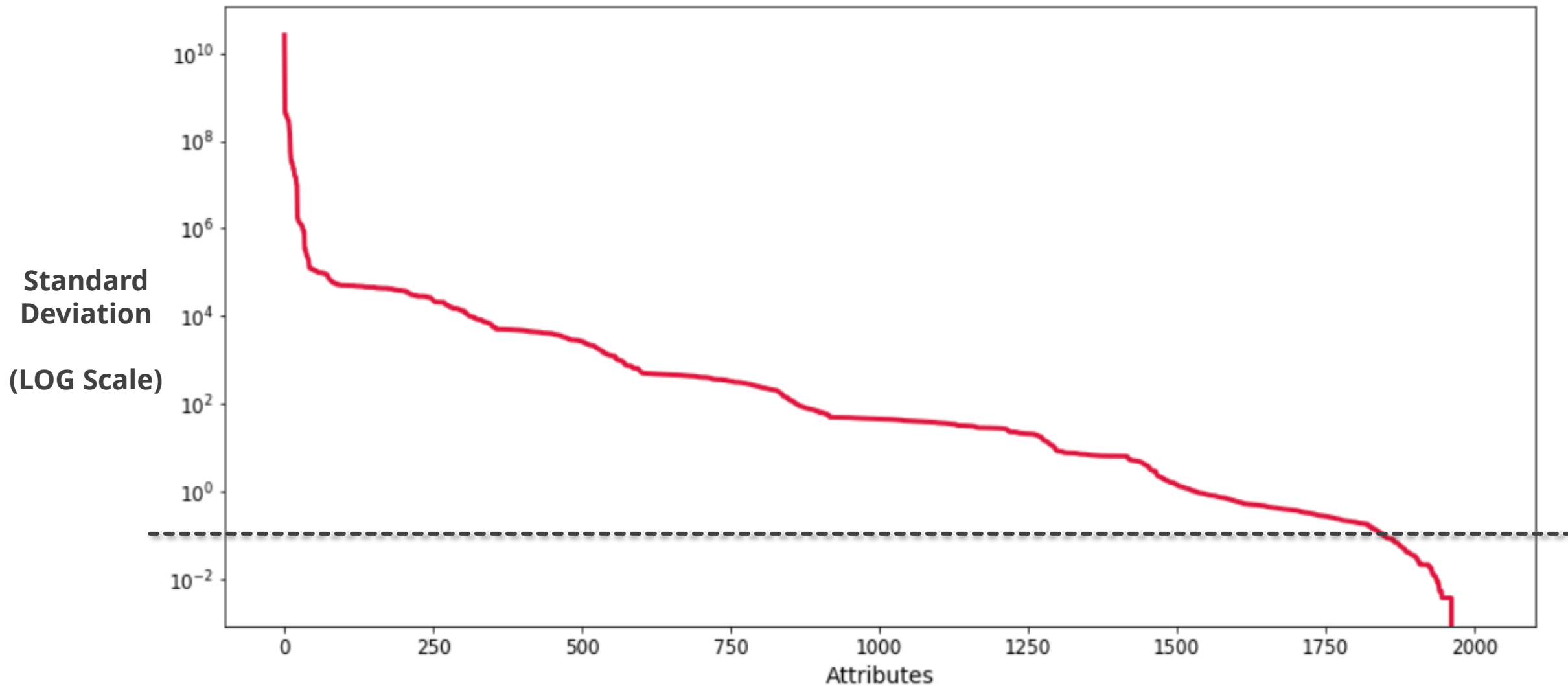
# 1

## Amount of Variation

- Drop or review variables that have a **very low standard deviation**.
  - Drop variables with zero variation (unary variables), if any.

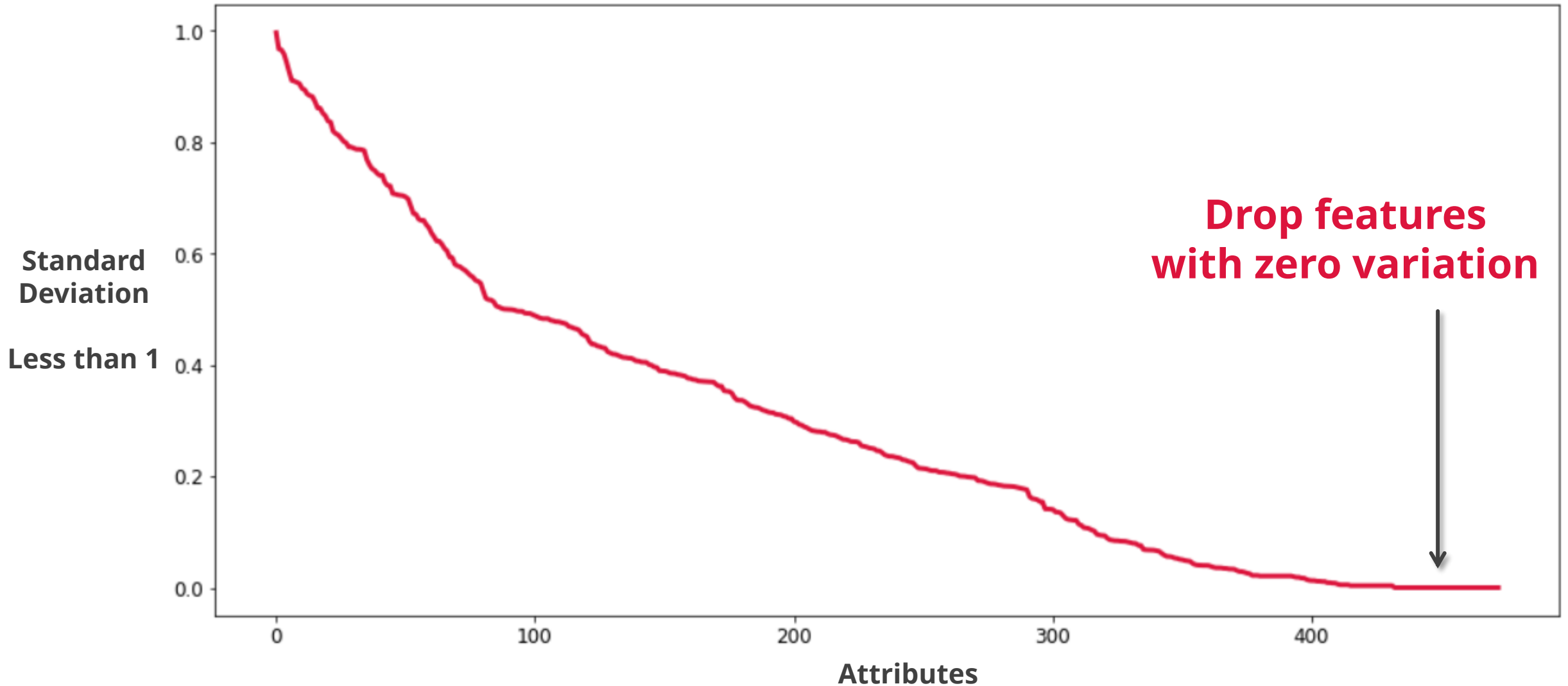
## 1

## Amount of Variation



# 1

## Amount of Variation



## 2

## Percent Missing Values

- Drop variables that have a very **high % of missing values**.
  - $\text{Number of records with missing values} / \text{Number of total records}$
- Create **binary indicators** to denote missing (or non-missing) values.
- Review or visualize variables with high % of missing values.

# 3

## Pairwise Correlations

- Many variables are often correlated with each other, and hence one of them is possibly redundant.
- If two variables are **highly correlated**, keeping only one will help reduce dimensionality without much loss of information.
- **Which variable to keep?**
  - The one that has a higher correlation coefficient **with the target**.



## 3

## Pairwise Correlations

1

Identify pairs of highly correlated variables.

	$x_1$	$x_2$	$x_3$	$x_4$
$x_1$	1.00	0.15	0.85	0.01
$x_2$		1.00	0.45	0.60
$x_3$			1.00	0.17
$x_4$				1.00

Correlation tolerance  $\approx 0.85$

## 3

## Pairwise Correlations

1

Identify pairs of highly correlated variables

	$x_1$	$x_2$	$x_3$	$x_4$
$x_1$	1.00	0.15	0.85	0.01
$x_2$		1.00	0.45	0.60
$x_3$			1.00	0.17
$x_4$				1.00

Correlation tolerance  $\approx 0.85$

2

Discard variable that has a weaker correlation with the target

	$y$
$x_1$	0.67
$x_2$	0.82
$x_3$	0.38
$x_4$	0.25

## 3

## Pairwise Correlations

1

Identify pairs of highly correlated variables.

	$x_1$	$x_2$	$x_3$	$x_4$
$x_1$	1.00	0.15	0.85	0.01
$x_2$		1.00	0.45	0.60
$x_3$			1.00	0.17
$x_4$				1.00

Correlation tolerance  $\approx 0.85$

2

Discard variable that has a weaker correlation with the target.

	$y$
$x_1$	0.67
$x_2$	0.82
$x_3$	0.38
$x_4$	0.25

Keep  $x_1$ , Discard  $x_3$

# 4

## Multicollinearity

- When two **or more** variables are highly correlated with each other.
- Dropping one or more variables should help reduce dimensionality without a substantial loss of information.
- Which variable(s) to drop?
  - Use **Condition Index**.

## 4

## Multicollinearity

## Eigenvalue Decomposition

	$x_1$	$x_2$	$x_3$	$x_4$
$x_1$	1.00	0.15	0.85	0.01
$x_2$		1.00	0.45	0.60
$x_3$			1.00	0.17
$x_4$				1.00

<i>Condition Index</i>	$x_1$	$x_2$	$x_3$	$x_4$
$u_1 = 1.0$	0.01	0.05	0.00	0.16
$u_2 = 16.5$	0.03	0.12	0.01	0.19
$u_3 = 28.7$	0.05	0.02	0.13	0.25
$u_4 = 97.1$	0.93	0.91	0.98	0.11

Condition Index tolerance  $\approx 30$

## 4

## Multicollinearity

	$x_1$	$x_2$	$x_3$	$x_4$
$x_1$	1.00	0.15	0.85	0.01
$x_2$		1.00	0.45	0.60
$x_3$			1.00	0.17
$x_4$				1.00

<i>Condition Index</i>	$x_1$	$x_2$	$x_3$	$x_4$
$u_1 = 1.0$	0.01	0.05	0.00	0.16
$u_2 = 16.5$	0.03	0.12	0.01	0.19
$u_3 = 28.7$	0.05	0.02	0.13	0.25
$u_4 = 97.1$	<u>0.93</u>	<u>0.91</u>	<u>0.98</u>	<del>0.11</del>

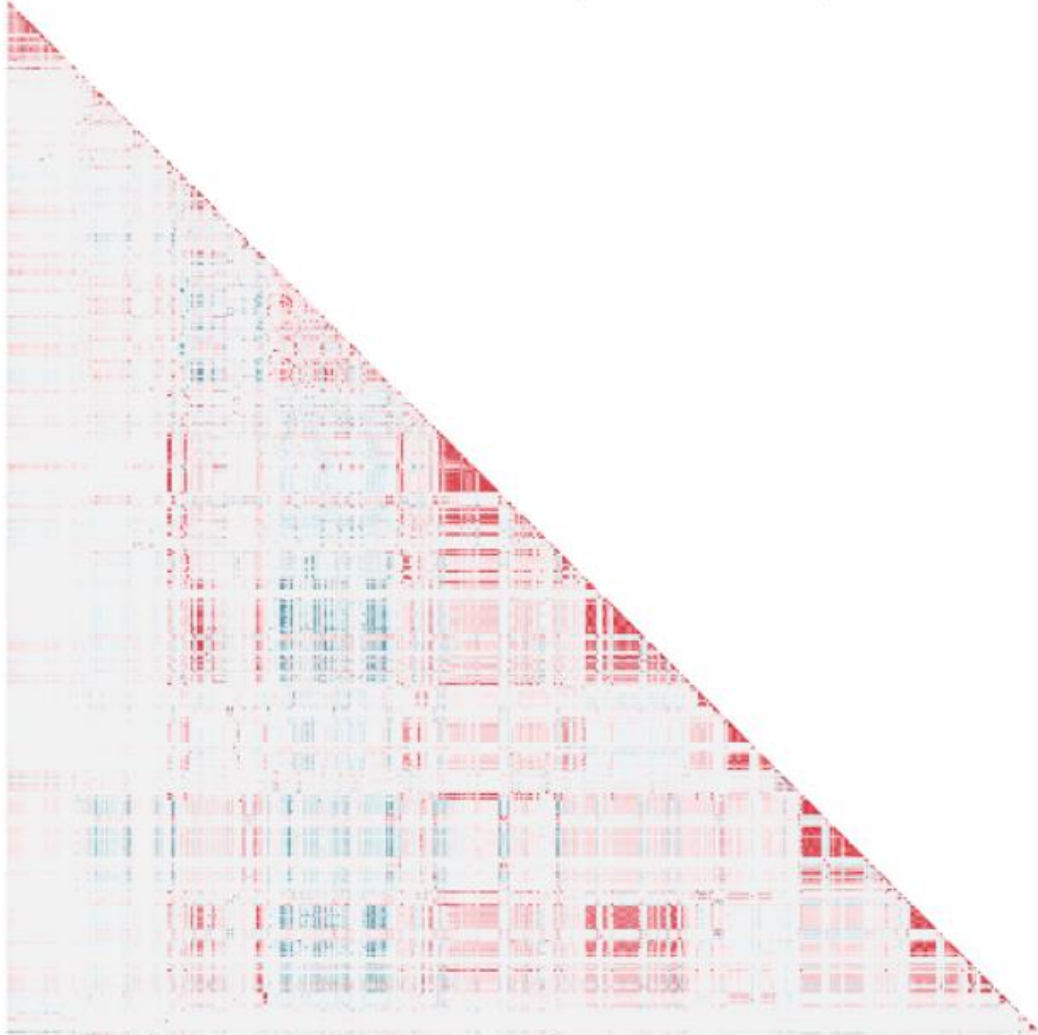
Discard  $x_3$ , Iterate

Condition Index tolerance  $\approx 30$

# 4

## Multicollinearity

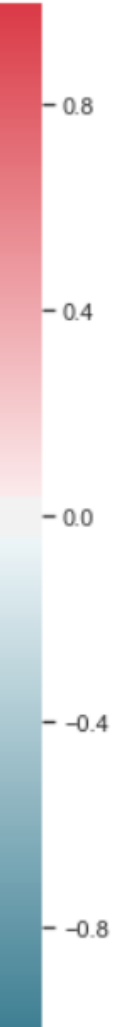
CORRELATION MATRIX (1667 FEATURES)



CORRELATION MATRIX (309 FEATURES)



**81% reduction in  
dimensionality!**



# 5

## Principal Component Analysis (PCA)

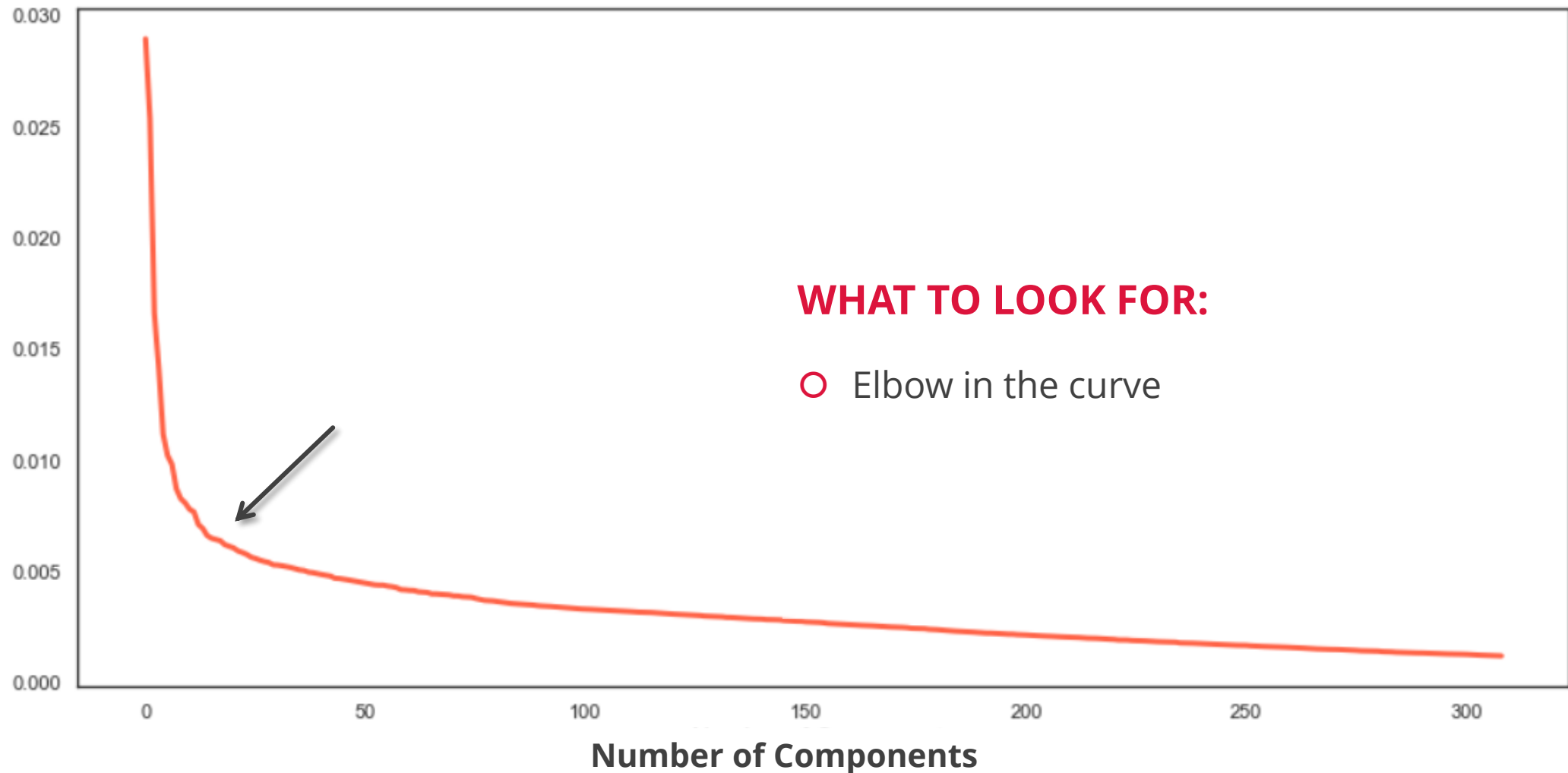
- A dimensionality reduction technique which emphasizes **variation**
- **Eliminates** multicollinearity – but **explicability** is compromised.
- Works independently from the (supervised) learning task.
- When to use:
  - Excessive multicollinearity
  - Explanation of the predictors is not important.
  - A slight overhead in implementation is okay.



## 5

## Principal Component Analysis (PCA)

% Variance  
Explained

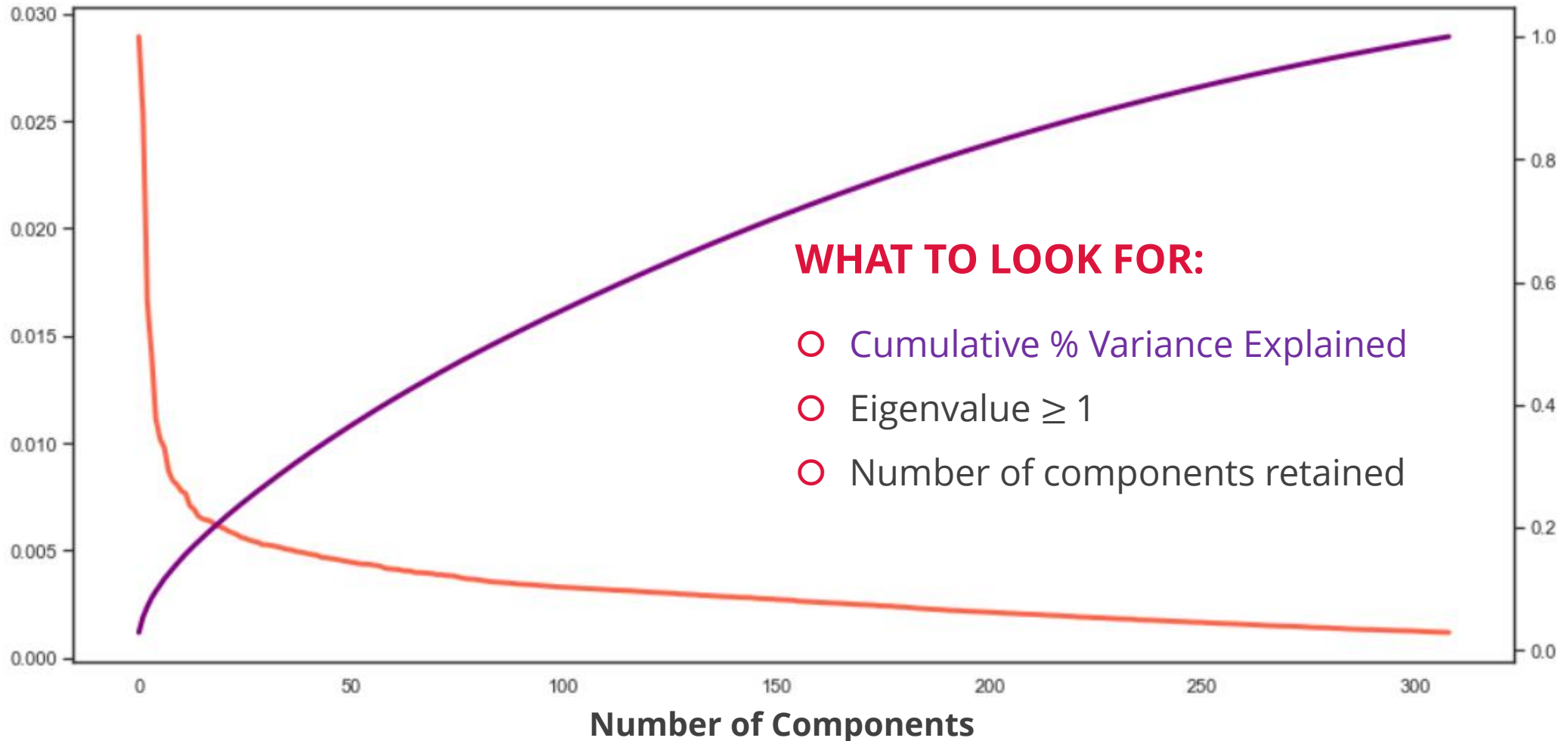


## 5

## Principal Component Analysis (PCA)

% Variance  
Explained

Cumulative  
% Variance  
Explained



# 6

## Variable Clustering

- A dimensionality reduction technique which emphasizes **correlation** (or, similarity).
  - Identify groups of variables that are as correlated as possible among themselves and as uncorrelated as possible with variables in other clusters.
- Reduces multicollinearity – and explicability is not compromised.

# 6

## Variable Clustering

```
from sklearn.cluster import FeatureAgglomeration  
  
var_clus = FeatureAgglomeration(n_clusters=50)  
  
X_clus = var_clus.fit_transform(X_scaled)
```

- This returns a pooled value for each cluster (i.e., the centroids), which can then be used to build a supervised learning model.
- You can choose one variable from each cluster that is the most representative of that cluster.
  - Use  $1 - R^2$  ratio to select the most representative variable from each cluster.

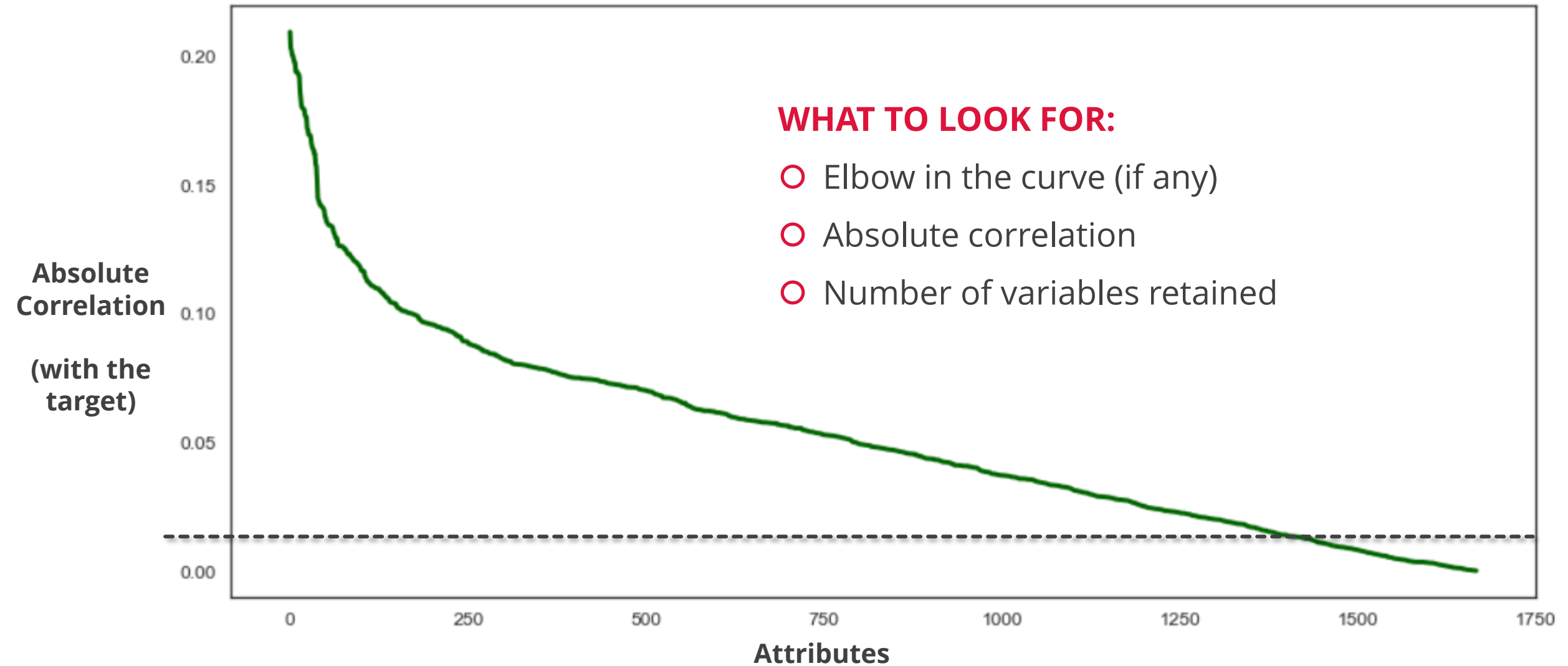
# 7

## Correlation (with the Target)

- Drop variables that have a very low correlation with the target.
  - If a variable has a very low correlation with the target, it's not going to be useful for the model (prediction).
- Caveat: **Linear** correlations

## 7

## Correlation (with the Target)



# 8

## Forward / Backward / Stepwise Selection

### ○ Forward Selection

1. Identify the **best** variable (e.g., based on model accuracy).
2. **Add** the next best variable into the model.
3. And so on until some predefined criterion is satisfied.

### ○ Backward Elimination

1. Start with **all** variables included in the model.
2. **Drop** the least useful variable (e.g., based on the smallest drop in model accuracy).
3. And so on until some predefined criterion is satisfied.

### ○ Stepwise Selection

- Similar to forward selection process, but a variable can also be dropped if it's deemed as not useful anymore after some steps.

- Least **A**bsolute **S**hrinkage and **S**election **O**perator
- Two birds, one stone: Variable Selection + Regularization

```
lasso = LogisticRegression(C=1.0, penalty='l1', random_state=314)
```



**OLS**

$$\arg \min_{\beta} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

**OLS**

$$\arg \min_{\beta} \sum_{i=1}^n (y_i - (\beta_0 + \sum_{j=1}^p \beta_j x_{ij}))^2$$

**LASSO**

$$\arg \min_{\beta} \sum_{i=1}^n (y_i - (\beta_0 + \sum_{j=1}^p \beta_j x_{ij}))^2 + \lambda \sum_{j=1}^p |\beta_j|^2$$

$\lambda$  controls the amount of ( $L_1$ ) regularization.

# LASSO

$$\arg \min_{\beta} \sum_{i=1}^n (y_i - (\beta_0 + \sum_{j=1}^p \beta_j x_{ij}))^2 + \lambda \sum_{j=1}^p |\beta_j|^2$$

- $\lambda$  controls the amount of ( $L_1$ ) **regularization**.
- If  $\lambda = 0$  then the LASSO model is identical to OLS.
- If  $\lambda > 1$  then a coefficient can increase only if it leads to a reduction in the Residual Sum of Squares (RSS for the OLS part).
- If  $\lambda = 1$  then a coefficient can increase by a certain amount only if it leads to the same amount of reduction in the RSS.
- Making  $\lambda$  sufficiently large will result in some of coefficients exactly equal to zero, aka feature reduction!

- Use a **forest** of trees to evaluate the importance of features.
- Fit a number of randomized decision trees on various sub-samples of the dataset and use `_feature_importances` to select features.