



# ML WITH A LARGE SET OF VARIABLES: FEATURE SELECTION FOR REGRESSION

June 2nd, ML Prague 2023

Aneta Havlínová, Martin Koryták



**Aneta Havlínová**  
Data Scientist



**Martin Koryt'ák**  
Data Scientist









# AGENDA


- 01** Data with many variables: introduction
- 02** Feature selection algorithms overview
- 03** Practical use-case introduction
- 04** Part I: Linear models for feature selection
- 05** Part II: Tree-based models for feature selection
- 06** Feature selection algorithm: how to choose and validate

# Set-up

- A. Go to GitHub: [ml-prague-2023](#)
- B. Clone this repository locally

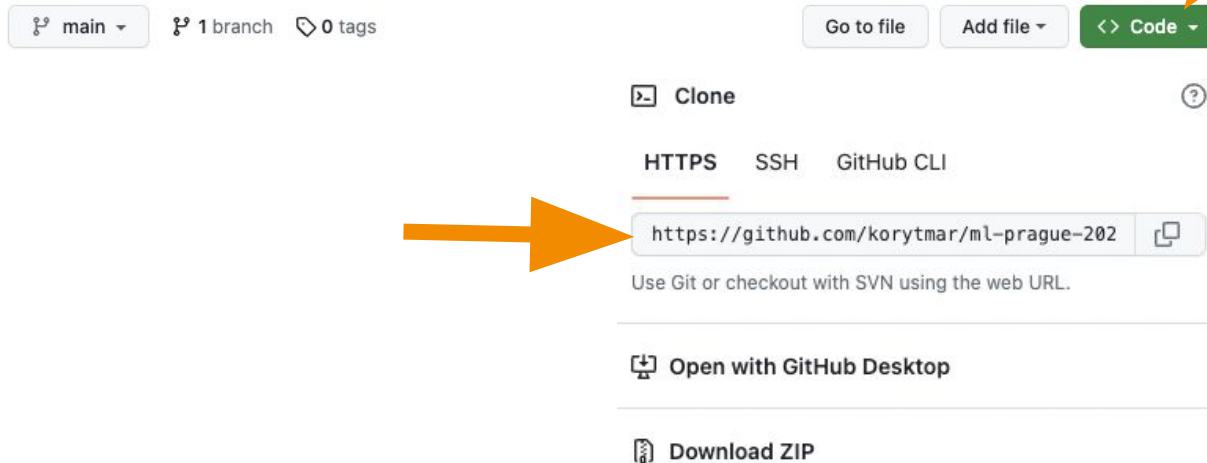
 main ▾  1 branch  0 tags

 Go to file  Add file ▾  Code ▾



# Set-up

- A. Go to GitHub: [ml-prague-2023](https://github.com/korytmr/ml-prague-2023)
- B. Clone this repository locally



# Know the audience

<https://t.ly/x9-V>



# DATA WITH MANY VARIABLES: INTRODUCTION

# Dataset with too many variables

## Example 1: HR data about employees

```
worker_data.head()
```

	compa_ratio	high_performer	high_potential	is_leader	is_manager	length_of_service	total_compensation	age	compa_ratio_range -> Above Compa-Ratio	compa_ratio_range -> At Compa-Ratio	...
0	1.088	False	False	False	False	1.573	4788.30	24.0	1	0	...
1	0.826	False	False	False	False	4.129	1238.43	28.0	0	0	...
2	0.998	False	False	False	False	7.463	2395.47	29.0	0	1	...
3	0.918	False	False	False	False	0.129	4133.05	22.0	0	0	...
4	1.343	True	False	False	False	1.307	1745.30	39.0	1	0	...

Goal: Predicting salaries using employee data such as experience, position, etc.



Anka





# Dataset with too many variables

## Example 1: HR data about employees

```
worker_data.head()
```

	compa_ratio	high_performer	high_potential	is_leader	is_manager	length_of_service	total_compensation	age	compa_ratio_range -> Above Compa-Ratio	compa_ratio_range -> At Compa-Ratio	...
0	1.088	False	False	False	False	1.573	4788.30	24.0	1	0	...
1	0.826	False	False	False	False	4.129	1238.43	28.0	0	0	...
2	0.998	False	False	False	False	7.463	2395.47	29.0	0	1	...
3	0.918	False	False	False	False	0.129	4133.05	22.0	0	0	...
4	1.343	True	False	False	False	1.307	1745.30	39.0	1	0	...

1000 rows x 404 columns

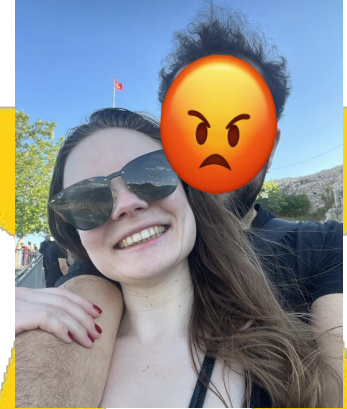


# Dataset with too many variables

## Example 1: HR data about employees

```
worker_data.head()
```

	compa_ratio	high_performer	high_potential	is_leader	is_manager	length_of_service	total_compensation	age	compa_ratio_range -> Above Compa-Ratio	compa_ratio_range -> At Compa-Ratio	...
0	1.088	False	False	False	False	1.573	4788.30	24.0	1	0	...
1	0.826	False	False	False	False	4.129	1238.43	28.0	0	0	...
2	0.998	False	False	False	False	7.463	2395.47	29.0	0	1	...
3	0.918	False	False	False	False	0.129	4133.05	22.0	0	0	...
4	1.343	True	False	False	False	1.307	1745.30	39.0	1	0	...



Angry boyfriend

1000 rows x 404 columns



Manual feature selection time: 2 days



# Dataset with too many variables

Example 2: Biology experiment with genetics data

	Gene 1	Gene 2	Gene 3	Gene 4	Gene 5
<b>Batch 1</b>	0.405	0.326	0.234	0.348	0.748
<b>Batch 2</b>	0.089	0.293	0.192	0.123	0.385
<b>Batch 3</b>	0.459	0.125	0.543	0.334	0.218
<b>Batch 4</b>	0.123	0.389	0.238	0.651	0.972

Goal: Predicting protein production of a cell using gene expressions data



Taha

# Dataset with too many variables

Example 2: Biology experiment with genetics data

	Gene 1	Gene 2	Gene 3	Gene 4	Gene 5
1	0.405	0.326	0.234	0.348	0.748
2	0.089	0.293	0.192	0.123	0.385
3	0.459	0.125	0.543	0.334	0.218
4	0.123	0.389	0.238	0.651	0.972

~200 rows x 10 000+ columns



# Dataset with too many variables

Example 2: Biology experiment with genetics data

	Gene 1	Gene 2	Gene 3	Gene 4	Gene 5
1	0.405	0.326	0.234	0.348	0.748
2	0.089	0.293	0.192	0.123	0.385
3	0.459	0.125	0.543	0.334	0.218
4	0.123	0.389	0.238	0.651	0.972

~200 rows x 10 000+ columns



Manual feature selection time: 2 years



Very angry wife



# Dataset with too many variables: So what can we do?

Go home  
and cry

Option 1

Do the  
manual  
selection  
anyway

Option 2

Use  
all  
features

Option 3

Use  
feature  
selection  
algorithm

Option 4

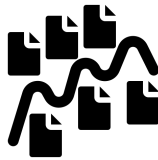
# Using all the features → not the best idea



A) Interpretability



B) Correlation among features



C) Overfitting



D) Inefficiency

## A) Interpretability

100 features	I can't	
1 000 features	I cannot	
10 000 features	I am unable to can	



## B) Correlated features

Too many features → likely many are highly correlated.

Some models such as those based on linear regression (Ridge, Lasso, OMP) have an **assumption of no multicollinearity**  
→ if high correlation is present, we get **biased coefficient estimates**.

## C) Efficiency

Less data → algorithms are trained faster.

Feature selection is especially useful for algorithms that are trained repeatedly.



## D) Overfitting

*"Garbage in,  
garbage out."*



→ Model works very well with training data, but does not generalise well with unseen data.

### Example: ML Doctor

Training observations



Features: green t-shirt, dark hair, cough → **label**: flu

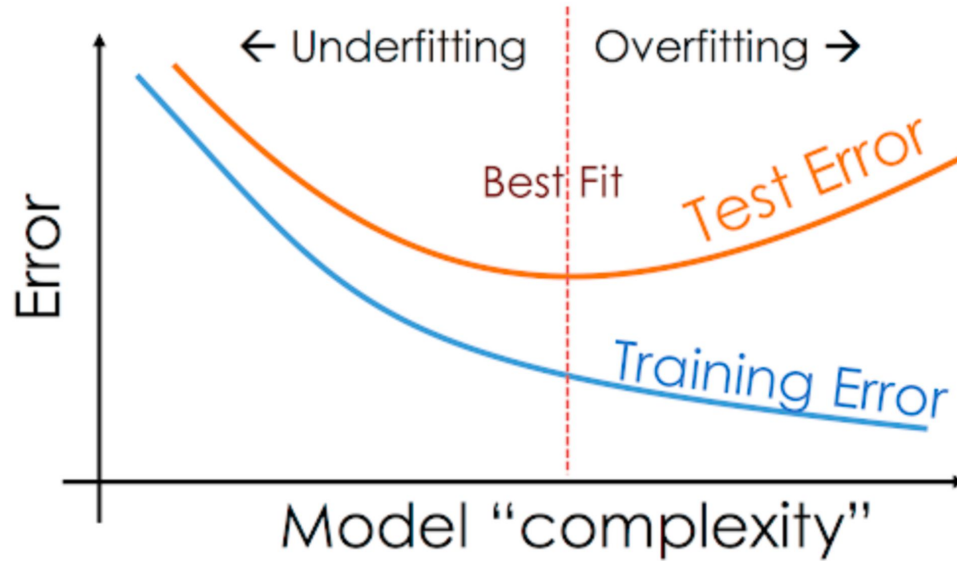
Testing observation



Features: green t-shirt, dark hair → **label**: flu

## D) Overfitting

→ Error on testing dataset is much higher than on training dataset



# Using all the features → not the best idea



A) Interpretability



B) Correlation among features



C) Overfitting



D) Inefficiency

# Dataset with too many variables: So what can we do?

Go home  
and cry

Option 1

Do the  
manual  
selection  
anyway

Option 2

Use  
all  
features

Option 3

Use  
feature  
selection  
algorithm

Option 4



# FEATURE SELECTION ALGORITHMS

# Feature selection: how many features are too many?

**Rule of thumb:** 10 samples to one feature

**Research:** *Hua, Jianping, et al. (2005)*

- **Uncorrelated features:** optimal number of features is  $N-1$  ( $N$  = sample size)
- As **feature correlation** increases, optimum is approximately  $\sqrt{N}$

**Other factors to consider:**

- the type of ML problem
- model complexity
- data quality and availability
- performance improvement with more features
- etc.



# Feature selection: methods

## Feature extraction

= Getting features typically by **transforming them to a new feature space**.

→ dimensionality reduction

Example: PCA creates new variables by linearly combining the original ones.

vs.

## Feature selection

= Choosing a **subset** of the original features set.

→ Feature selection is a special case of dimensionality reduction.

# Feature selection: methods

## 1. FILTER METHODS

Features are selected based on their **scores in various statistical tests**.

→ Correlation, ANOVA, Chi-Squared, ...

## 2. WRAPPER METHODS

**Many model specifications are tested** - based on one model's result, we decide about inclusion/exclusion of a variable in the next model.

→ Boruta, Orthogonal Matching Pursuit, Backward Elimination, ...

## 3. EMBEDDED METHODS

Algorithms with **built-in methods** for selecting features.

→ Random Forest, Ridge, LASSO, ...

# Feature selection: methods

## FILTER METHODS

- + Quick and less computationally expensive
- Ignore feature dependencies

## WRAPPER METHODS

- + Usually provide better model performance than filter methods
- Can have high computational cost

## EMBEDDED METHODS

- + Combine the best characteristics of filtering and wrapping
- Might be also computationally heavy

# Feature selection: methods we will cover

## 1. FILTER METHODS

Features are selected based on their scores in various statistical tests.

→ Correlation, ANOVA, Chi-Square, ...

## 2. WRAPPER METHODS

According to the conclusions drawn from the previous model, we decide whether to include/exclude certain features.

→ Boruta, Orthogonal Matching Pursuit, Backward Elimination, ...

## 3. EMBEDDED METHODS

Algorithms with built-in methods for selecting features.

→ Random Forest, Ridge, LASSO, ...

# PRACTICAL USE-CASE INTRODUCTION

HR Analytics: Predicting salaries

# HR Dataset

- Each row represents individual employee
- Each employee has some attributes:
  - Legal Name
  - Gender
  - Employee ID
  - Job Profile
  - Ethnicity
  - **Salary**
  - ...

# HR Dataset: Why predicting salaries?

- **Fair Compensation:** based on their skills, qualifications, experience, and contributions to the organization.
- **Transparency:** establishing a transparent and objective system for determining salaries, promoting equality and reducing biases.
- **Employee Retention and Satisfaction:** identifying any discrepancies, such as underpayment or overpayment.

# HR Dataset: Example

	compa_ratio	compa_ratio_range	cost_center	country	current_rating	employee_id	ethnicity	gender	generation	high_performer	...	org_level_2	org_level_3
0	70.767	Above Compa-Ratio	AMER - United States of America	United States of America	4 - Exceeds Expectations	1000	Asian	Female	Generation X (1965-1980)	False	...	NaN	NaN
1	25.560	Above Compa-Ratio	AMER - United States of America	United States of America	3 - Meets Expectations	1001	White	Male	Generation Z (1997 and onwards)	False	...	Board	Technology & IT
2	18.794	Above Compa-Ratio	Global Support - Asia/Pac	South Korea	5 - Outstanding Performance	1002	Asian	Male	Generation Y/Millenials (1981-1996)	True	...	Board	Technology & IT
3	13.968	Above Compa-Ratio	AMER - Canada	Canada	2 - Needs Improvement	1003	Asian	Male	Generation X (1965-1980)	False	...	NaN	NaN
4	21.618	Above Compa-Ratio	Global Support - Asia/Pac	South Korea	3 - Meets Expectations	1004	Asian	Female	Generation Y/Millenials (1981-1996)	False	...	Board	Finance





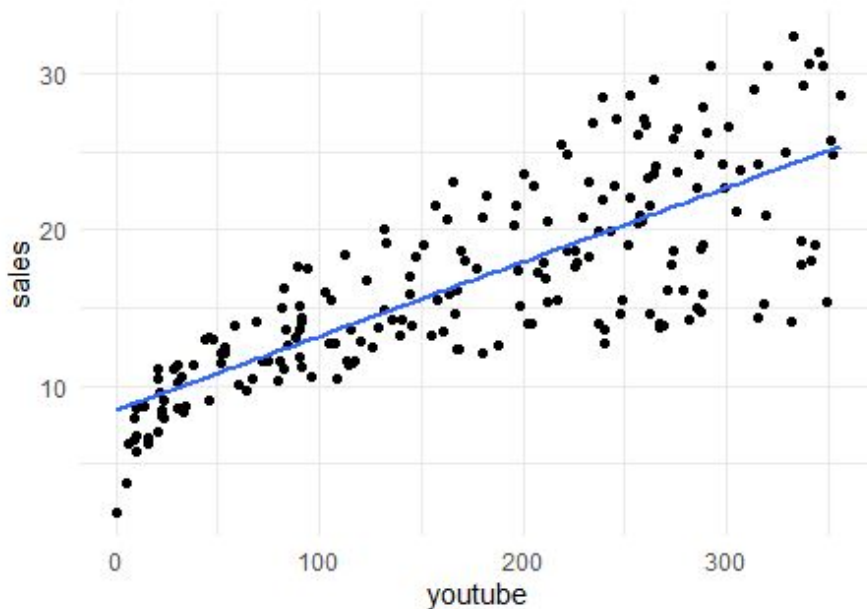
# DATA PREPROCESSING PRACTICAL PART

# PART I: LINEAR REGRESSION MODELS

# Linear Regression: The simplest example

$$y = \beta_0 + \beta_1 x + \varepsilon$$

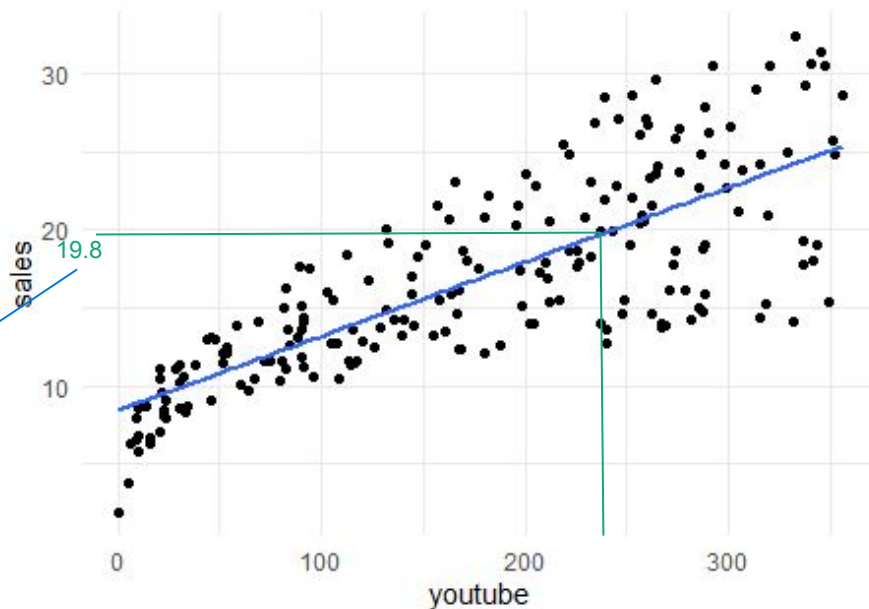
youtube	sales
276.12	26.52
53.40	12.48
20.64	11.16
181.80	22.20
216.96	15.48
10.44	8.64
69.00	14.16
144.24	15.84
10.32	5.76
239.76	?
79.32	?
257.64	?



# Linear Regression: The simplest example

$$y = \beta_0 + \beta_1 x + \varepsilon$$

youtube	sales
276.12	26.52
53.40	12.48
20.64	11.16
181.80	22.20
216.96	15.48
10.44	8.64
69.00	14.16
144.24	15.84
10.32	5.76
239.76	?
79.32	?
257.64	?



# Assumptions

We need to check that model assumptions hold.

- Linear relationship → biased betas
  - No multicollinearity → high variance of beta estimates
  - Random sample → biased betas
  - No omitted variable → biased betas
- 

Impact on  
values of  
betas

- Homoskedasticity → invalid inference (use robust errors)
- Normality → invalid inference (large sample desired)

Impact on  
validity of  
inference



# LINEAR REGRESSION PRACTICAL PART

# Orthogonal Matching Pursuit (OMP)

- Iterative greedy algorithm for sparse recovery
- OMP uses Linear Regression for estimation of coefficients in each iteration
- Improved version of Matching Pursuit algorithm

## **Properties**

- Convergence guaranteed for any target (signal) that is in the space spanned by the dictionary
- The error decreases monotonically
- At each step, the residuals are orthogonal to the selected features (atoms)

# Orthogonal Matching Pursuit (OMP)

**Input:**  $X$  - input dataset,  $y$  - target variable

**Output:**  $z$  - selected features,  $\beta$  - weights corresponding to selected Top Drivers

normalize  $X$  s.t. L2 norm of each column equals 1 (optional step) and standardize  $X$

residuals  $\leftarrow y$

$z \leftarrow$  empty set

while  $y$  is **not sufficiently explained** do

1. compute a **pseudo correlation score** of residuals for each column in  $X$  as a dot product
2. select column from  $X$  which has the **highest absolute score** in step 1 and save its index to  $z$
3. form a **new input dataset** consisting of columns in  $z$  and name it  $X'$
4.  $w \leftarrow$  **solve Least square problem** using  $X'$  and  $y$
6. **calculate new predictions** for all rows in  $X$  using coefficients  $\beta$
7. **update residuals** as a difference between  $y$  and new predictions obtained in step 6



# OMP: Iteration Explained

## Inputs

- Feature set (Dictionary)  $D$ : [  
     $[-0.707, 0.8, 0],$   
     $[0.707, 0.6, -1]$   
     $]$
- Target (Signal)  $y$ :  $[1.65, -0.25]$

## Goal

- Estimate coefficients for each vector in dictionary such that signal can be recovered, i.e.  $D * x \approx y$

# OMP: Iteration Explained

- Dictionary:  $\begin{bmatrix} -0.707, 0.8, 0, \\ 0.707, 0.6, -1 \end{bmatrix}$
- Signal:  $[1.65, -0.25]$

Dictionary D consists of 3 atoms (vectors)  $b_1$ ,  $b_2$  and  $b_3$ :

- $b_1 = [-0.707, 0.707]$
- $b_2 = [0.8, 0.6]$
- $b_3 = [0, -1]$

We try to estimate coefficients in vector  $x$  using dictionary D and Signal  $y$  such that:

$$b_1 * x_1 + b_2 * x_2 + b_3 * x_3 = [-0.707, 0.707] * x_1 + [0.8, 0.6] * x_2 + [0, -1] * x_3 \approx y$$

# OMP: Iteration Explained

- Dictionary:  $\begin{bmatrix} -0.707, 0.8, 0, \\ 0.707, 0.6, -1 \end{bmatrix}$
- Signal:  $[1.65, -0.25]$

Let's find the contribution of atoms to Signal  $y$  using  $b_i^T y$ :

- $b_1^T y = [-0.707, 0.707] * [1.65, -0.25] = -0.707 * 1.65 - 0.707 * 0.25 = \mathbf{-1.34}$
- $b_2^T y = [0.8, 0.6] * [1.65, -0.25] = 1.17$
- $b_3^T y = [0, -1] * [1.65, -0.25] = 0.25$

Atom  $b_1$  has the highest contribution while neglecting the negative value.  
Furthermore, atom  $b_1$  forms basis A.

# OMP: Iteration Explained

- Dictionary:  $\begin{bmatrix} -0.707, 0.8, 0 \\ 0.707, 0.6, -1 \end{bmatrix}$
- Signal:  $[1.65, -0.25]$

Next, we calculate the contribution of basis A to signal y. This leads to the Least Square Problem which is formulated as follows:

$$\min \|A * x - y\|_2$$

which can be solved as:

$$A^+ = (A^T * A)^{-1} * A^T$$

# OMP: Iteration Explained

- Dictionary:  $\begin{bmatrix} -0.707, 0.8, 0, \\ 0.707, 0.6, -1 \end{bmatrix}$
- Signal:  $[1.65, -0.25]$

$$A^+ = (A^T * A)^{-1} * A^T$$

- $x_1 = A^+ * y = [b_1] * y = [-0.707, 0.707] * [1.65, -0.25] = -1.34$
- $x_2 = 0$
- $x_3 = 0$

# OMP: Iteration Explained

- Dictionary:  $\begin{bmatrix} -0.707, 0.8, 0, \\ 0.707, 0.6, -1 \end{bmatrix}$
- Signal:  $[1.65, -0.25]$
- Residuals = Signal - current predictions
- $r = y - A * x = y - b_1 * x_1 = [0.7, 0.7]$

# OMP: Iteration Explained

- Dictionary:  $\begin{bmatrix} -0.707, 0.8, 0 \\ 0.707, 0.6, -1 \end{bmatrix}$
- Signal:  $[1.65, -0.25]$

Second iteration and so on:

- $b_2$  contributes better than  $b_3$
- Coefficients are updated as follows  $x = [-1.2, 1, 0]$
- Residuals are zeros after update, i.e.  $[0, 0, 0]$
- The algorithm **stops** after this iteration as residuals are already vanished

# OMP: Summary

---

```
 $x, r, \Lambda = \text{OMP}(\Phi, y);$ 
 $x^0 \leftarrow 0;$ 
 $r^0 \leftarrow y;$                                      //  $r = y - \Phi x$ 
 $\Lambda^0 = \emptyset;$                                // Index set of chosen atoms
 $k \leftarrow 0;$                                      // Iteration counter
repeat
     $h^{k+1} \leftarrow \Phi^T r^k;$                      // Match
     $\lambda^{k+1} = \arg \max_{j \notin \Lambda^k} |h_j^{k+1}|;$       // Identify
     $\Lambda^{k+1} \leftarrow \Lambda^k \cup \{\lambda^{k+1}\};$     // Update support
     $x^{k+1} \leftarrow 0;$ 
     $x_{\Lambda^{k+1}}^{k+1} \leftarrow \Phi_{\Lambda^{k+1}}^\dagger y;$  // Update representation LS
     $y_{\Lambda^{k+1}}^{k+1} = \Phi x_{\Lambda^{k+1}}^{k+1};$            // Update approximation
     $r^{k+1} \leftarrow y - y_{\Lambda^{k+1}}^{k+1};$            // Update residual
     $k \leftarrow k + 1;$                              // Update iteration counter
until halting criteria is satisfied;
 $x \leftarrow x^k;$  ;  $\Lambda \leftarrow \Lambda^k;$  ;  $r \leftarrow r^k;$ 
```

---



# Quiz



Are these statements true or false?

- OMP sequentially selects atoms from the dictionary that best match the residual signal.
- OMP guarantees exact recovery of the sparse signal in a finite number of iterations.
- OMP assumes that the signal can be represented as a linear combination of a few dictionary atoms.



OMP PRACTICAL PART



# **PART II:** **TREE-BASED METHODS FOR FEATURE SELECTION**



# TREE-BASED METHODS

- 
- 01** Decision Trees: the intro algorithm
  - 02** Random Forest explained
  - 03** Random Forest implementation
  - 03** Boruta explained
  - 04** Boruta implementation



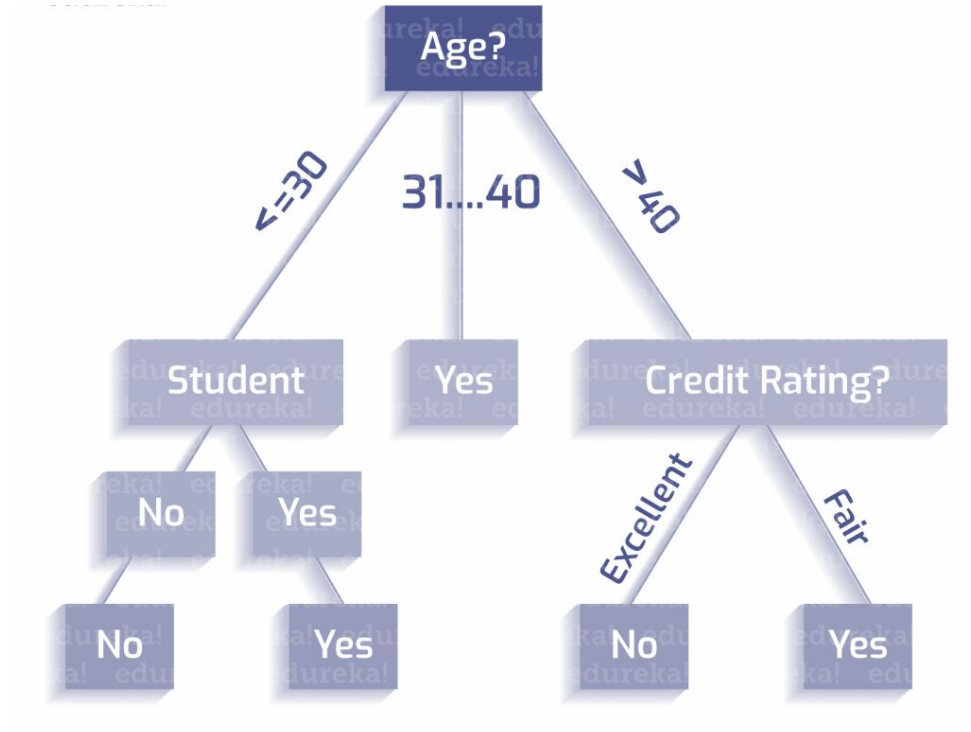
# DECISION TREES



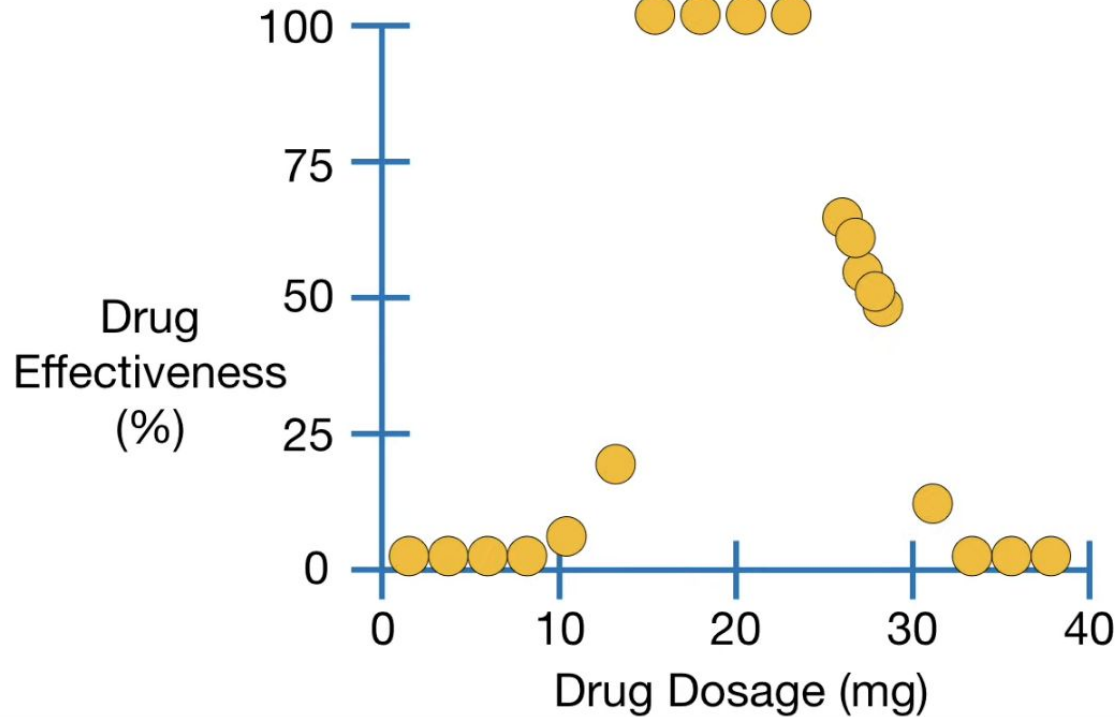
# Decision Tree: the starting algorithm

EXAMPLE:

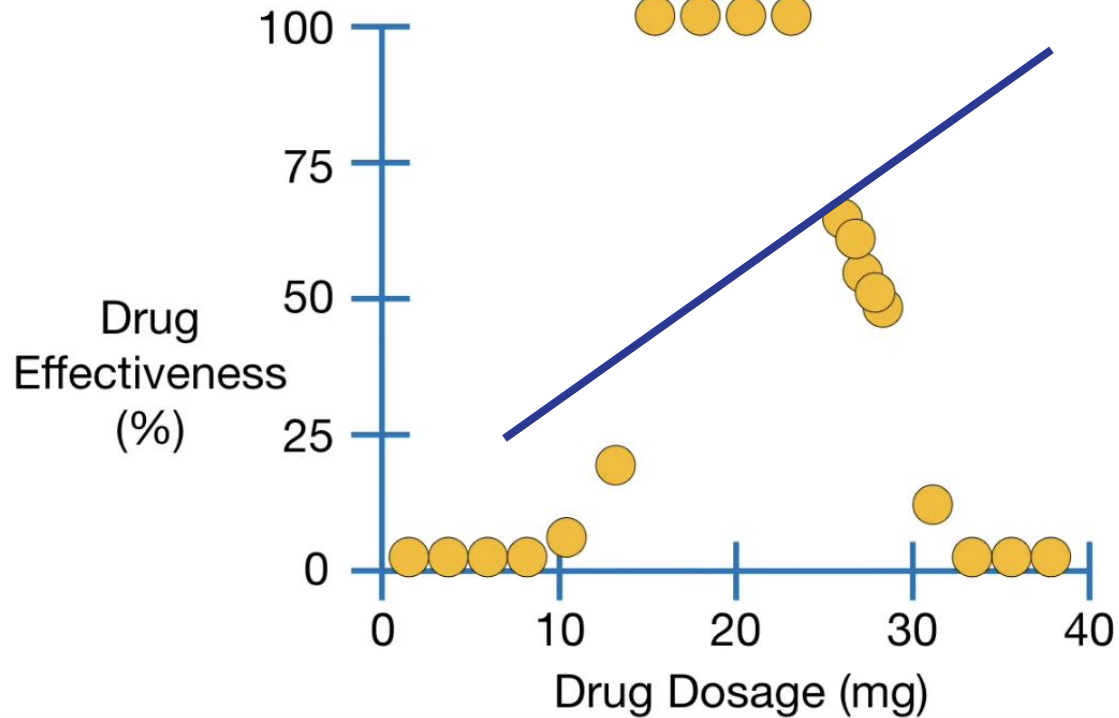
Predicting if a person will  
buy given product or not



# Decision Tree for regression: non-linearity

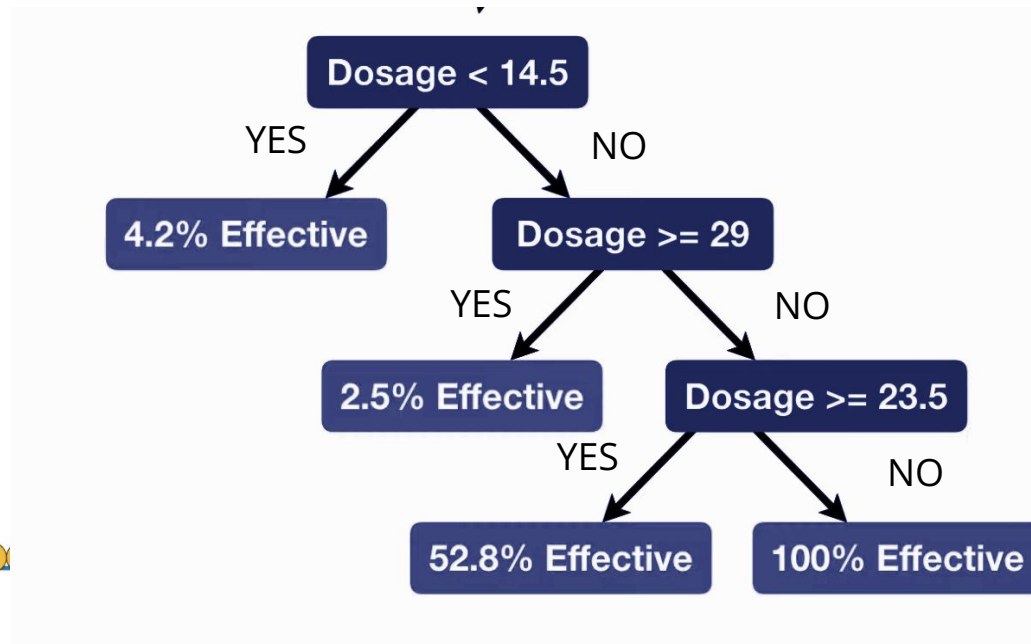
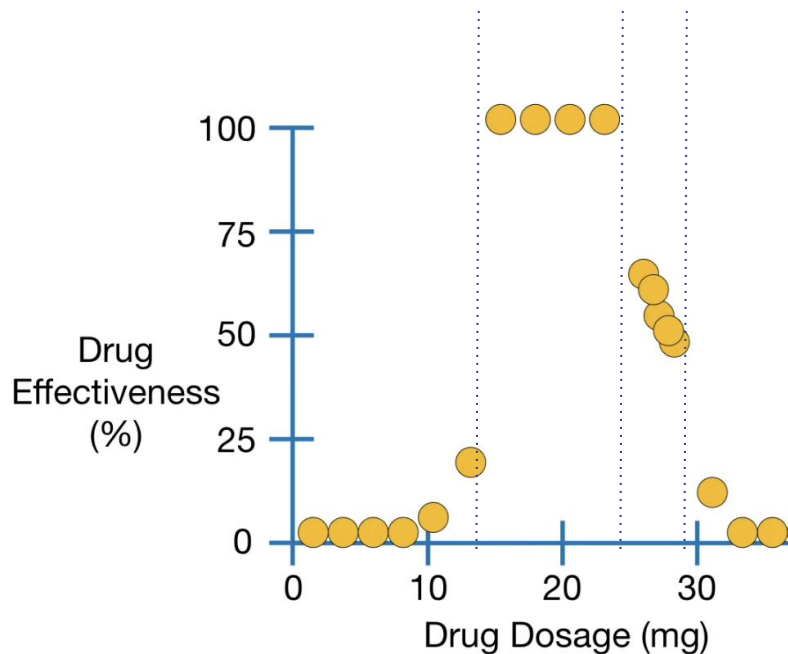


# Decision Tree for regression: non-linearity

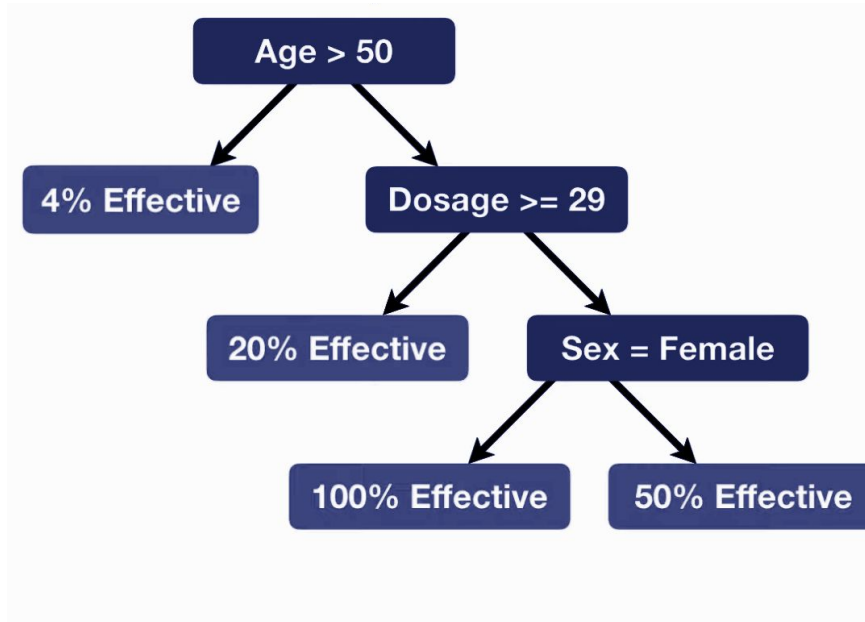




# Regression tree example



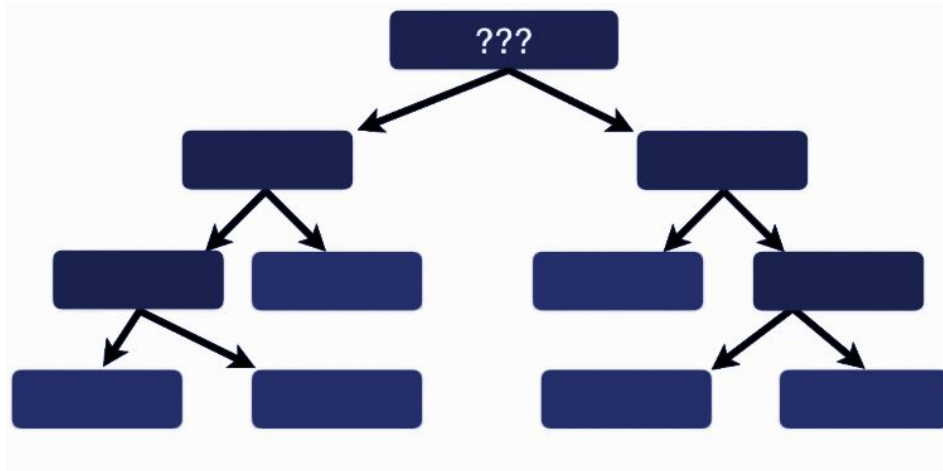
# Regression tree example - multiple variables



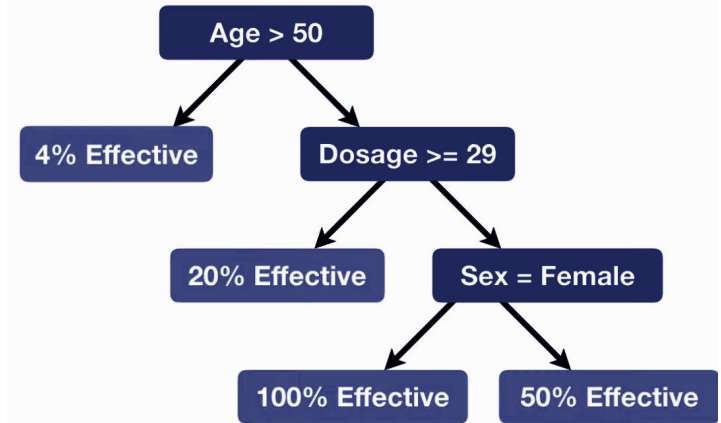
Dosage	Age	Sex	Etc.	Drug Effect.
10	25	Female	...	98
20	73	Male	...	0
35	54	Female	...	100
5	12	Male	...	44
etc...	etc...	etc...	etc...	etc...

# Building a tree - predicting effectiveness by dosage

What condition do we start with?



Is this the best tree?



# Building a tree - predicting effectiveness by dosage

What condition do we start with?

We try all possible thresholds, and **see which threshold gives us the lowest error.**

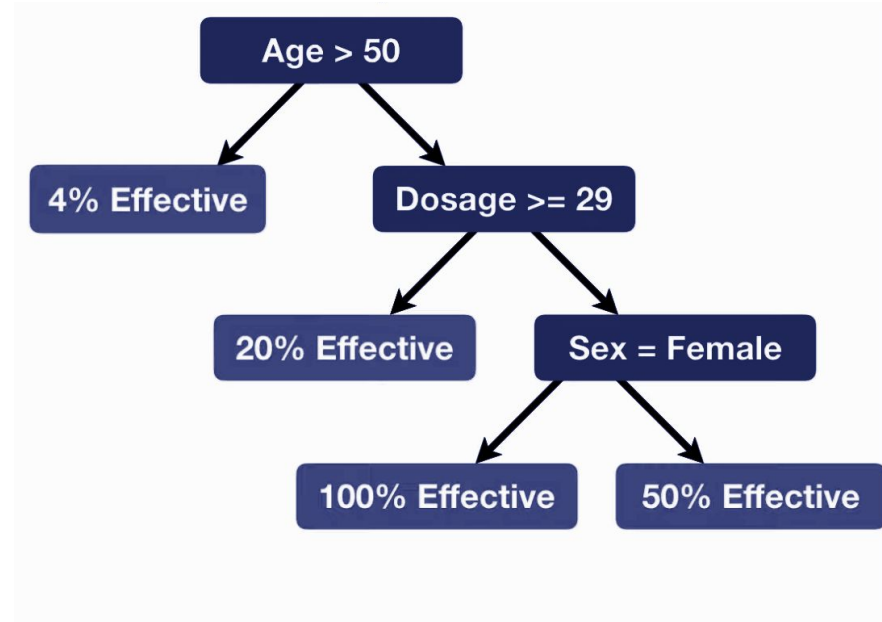
$$\text{Sum of squared residuals} = \sum (\text{Predicted value} - \text{Observed value})^2$$

Patient ID	Gender F	Weight	Diabetes	Effectiveness - observed	Effectiveness - predicted
111	1	55	1	65	67
112	1	68	0	72	74
113	0	80	0	33	32
114	1	59	0	83	79

# Predicting effectiveness using multiple variables

We calculate sum of squared residuals for every threshold of every variable.

→ We choose the one with the lowest error value.



# Decision Tree: Pros and Cons

## Advantages

- easy to visualize
- non-linear patterns can be captured easily
- no special data processing needed

## Disadvantages

- overfitting (imprecise predictions on new data) → large variance



# RANDOM FOREST



# Random Forest

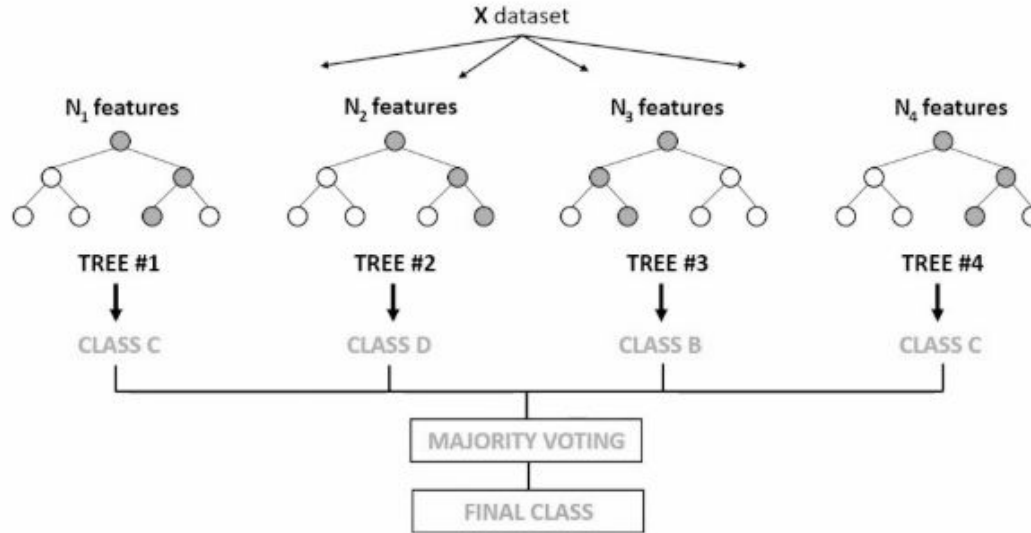
- Ensemble method that builds multiple decision trees
- It can have 50, 100, 200, 500, ... trees → depends on the data size





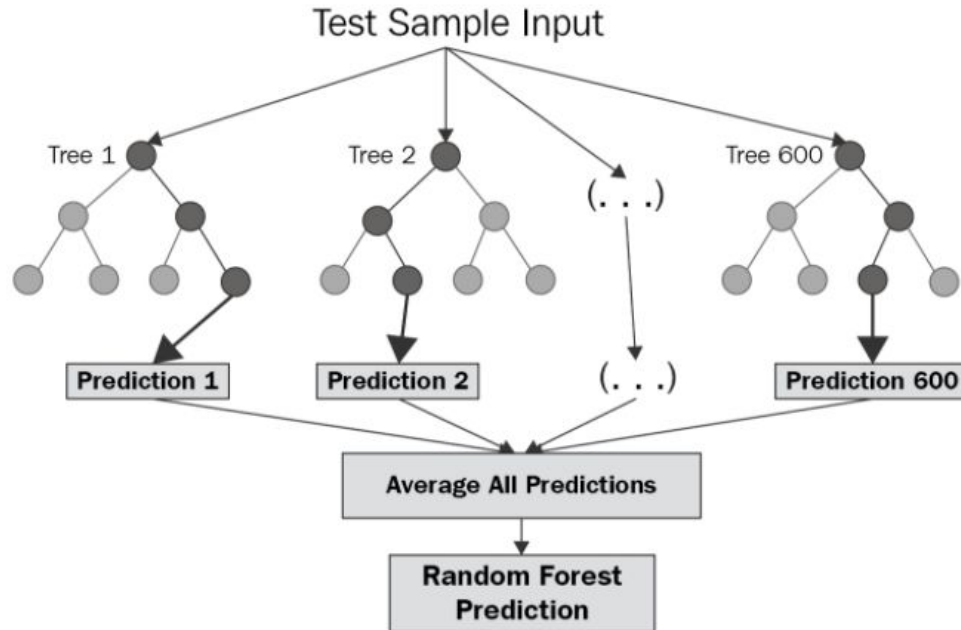
# Random Forest

In classification, the prediction is the majority vote of all decision trees' predictions.



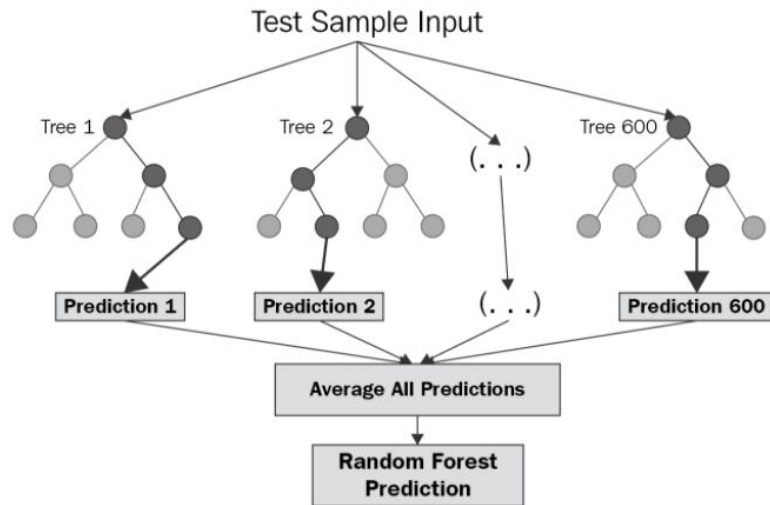
# Random Forest

In regression, the prediction is the average of all decision trees' predictions.



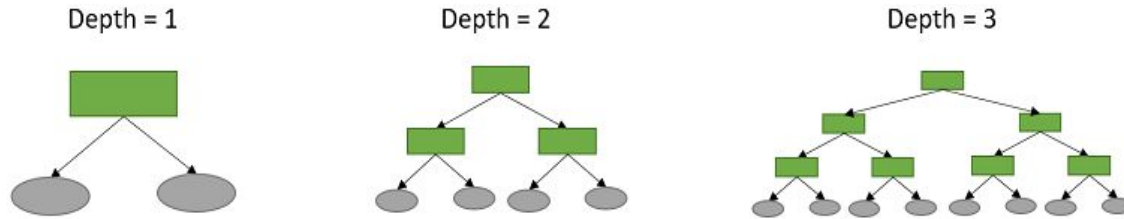
# Random Forest: How it works

- Each tree is build based on a **random sample** from training data.
- The algorithm **randomly selects a subset of explanatory variables** for each split in each decision tree.
- We use the **majority vote/average** as the final prediction.



# Random Forest: Hyperparameters tuning

a) Maximum depth of the trees



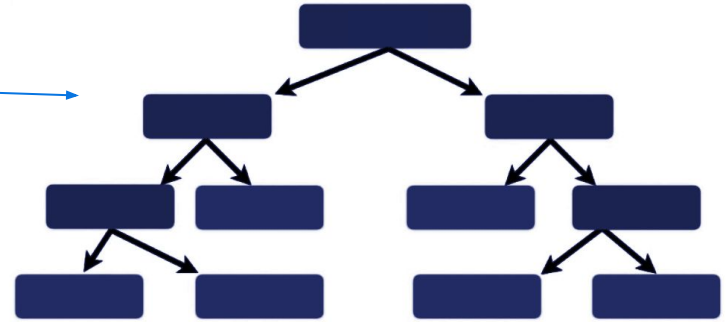
b) Number of trees in the forest

c) Maximum number of features to be considered at each node split

# Random Forest: Hyperparameters tuning

d) Minimum number of samples required to split a node

e) Minimum number of samples in a leaf node



# RF feature selection: Variable importance

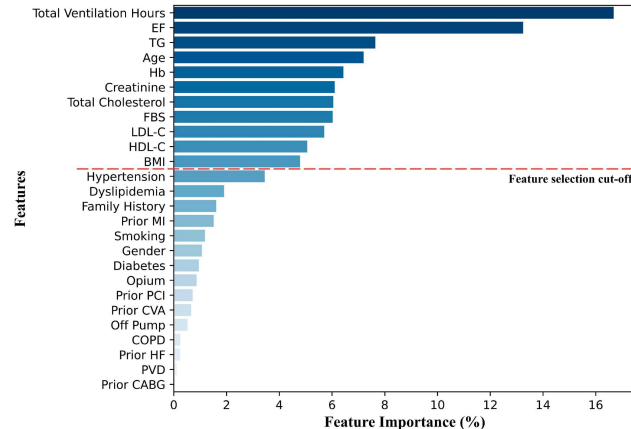
- Measure for ranking our explanatory variables
- Not interpreted as betas in linear regression

Variable Importance → increase in node purity: how much the sum of squared errors is reduced whenever a variable is chosen to split.

# RF feature selection: Variable importance

Final set of features:

- best **k** attributes
- best **x %** of attributes
- make **a cutoff** at the biggest difference in importance scores



# Random Forest for feature selection

- One of the **most accurate** algorithms available (even with a lot of missing data)
- **Efficient** with large datasets
- Variable importance as a **straightforward metrics**
- However, gives us just **relative ranking**, not the **magnitude** of feature impact



# Quiz



Are these statements true or false?

- Random forest randomly selects a subset of explanatory variables for each split in each decision tree.
- There is no generally applicable variable importance threshold.
- Decision tree maximizes the sum of squared residuals when it selects the decision conditions.



# RANDOM FOREST PRACTICAL PART



BORUTA



# Boruta

- An algorithm that builds up on random forest
- First, another data frame is created from original X data by randomly shuffling each feature
- The newly created data frame is attached to the original dataframe → now we have **twice as many columns**

# Boruta: shadow features

- Variables created by random shuffling of original values are called **shadow features**

	age	height	weight	shadow_age	shadow_height	shadow_weight
0	25	182	75	51	176	75
1	32	176	71	32	182	71
2	47	174	78	47	168	78
3	51	168	72	25	181	72
4	62	181	86	62	174	86

- Random Forest is run with all (old + shadow) features and variable importance is calculated



# Boruta: feature selection

**Threshold for variable selection:** a feature must have importance higher than the highest feature importance recorded among the shadow features

→ **The main idea:** a feature is useful only if it's capable of doing better than the best randomized feature.

# Boruta: feature selection

→ Based on a high variable importance, we tag some features as 'important'

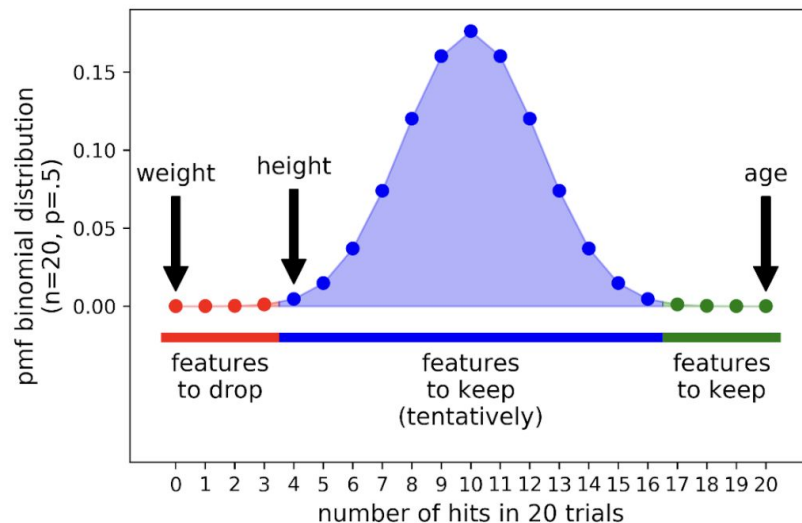
→ If **some features have significantly lower importance** than the best shadow feature, we tag them as 'unimportant' and **remove them**

# Boruta: multiple randomized runs

We use multiple boruta runs:

- a specified number of iterations, or
- until all features are tagged as 'important' or 'unimportant'

A series of  $n$  trials follows a binomial distribution.

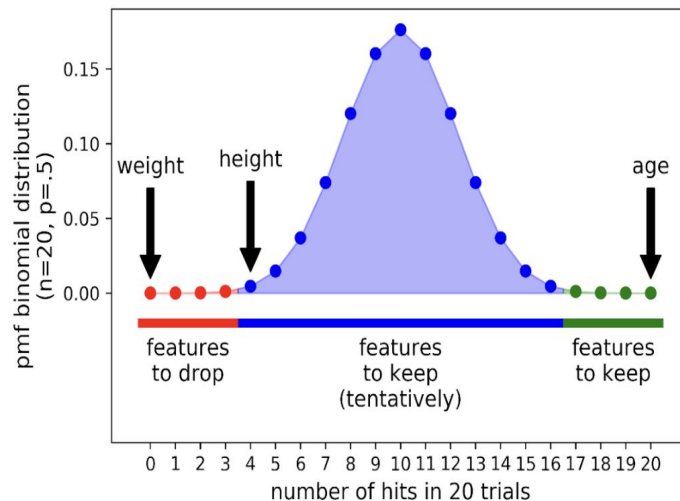




# Boruta: final decision for selection

Boruta has three areas:

- **area of refusal** (red): features are dropped
- **area of irresolution** (blue): Boruta is indecisive about these features → up to a data scientist, but these features are typically kept
- **area of acceptance** (green): features are kept



# Quiz



Are these statements true or false?

- For each variable, Boruta gives us a definite answer on whether to keep it in the model or not.
- Boruta uses original features and shadow features in one model all at the same time.
- Shadow features are created by randomly shuffling values of the original features.



BORUTA PRACTICAL PART



CONCLUSION



# How are the selected features similar?

Jaccard similarity:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Selected features by all models:

```
{'age', 'compa_ratio', 'cost_center ->  
Global Support - Asia/Pac', 'is_leader',  
'job_level -> M5', 'job_level -> M6',  
'job_level -> M7', 'job_level -> M8',  
'job_level -> P1', 'job_level -> P3',  
'job_level -> P5', 'region -> AMER'}
```

	OMP	RF	Boruta
OMP	1	0.43	0.48
RF	0.43	1	0.87
Boruta	0.48	0.87	1

# How to choose your algorithm?

Multiple metrics which are used for model evaluation on the testing dataset.

<b>Metric</b>	<b>OMP</b>	<b>RF</b>	<b>Boruta</b>
<b>RMSE</b>	24 917	6 821	6 700
<b>MAPE</b>	9.84 %	0.5 %	0.46 %

# How to evaluate/compare models?

- Compare **performance metrics** (consider multiple metrics)
- Check results of other models (**consistency** of outputs)
- Consider **model complexity** (architecture, number of parameters or training time – always prefer less complex model)
- Consider **domain-specific knowledge** and interpretability
- External **benchmarks** (SOTA models in the field)
- Consider also models not covered in this workshop such as Lasso, Ridge and Gradient boosted trees

# Conclusion

## What we learned today:

- Issues when dealing with too many **features**: interpretability, correlation, efficiency, overfitting
- **Algorithms**: linear & tree based methods
- Metrics (RMSE, MAPE) to **evaluate models**
- **Select the best model** based on: performance, consistency, complexity, or domain knowledge



# Conclusion

## What we learned today:

- Issues when dealing with too many **features**: interpretability, correlation, efficiency, overfitting
- **Algorithms**: linear & tree based methods
- Metrics (RMSE, MAPE) to **evaluate models**
- **Select the best model** based on: performance, consistency, complexity, or domain knowledge





**Thank You**





# Questions?



Your feedback is important!

<https://t.ly/u9R6>



# Sources

<https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f>

<https://towardsai.net/p/programming/decision-trees-explained-with-a-practical-example-fe47872d3b53>

Taboga, Marco (2021). "Decision tree", Lectures on machine learning. <https://www.statlect.com/machine-learning/decision-tree>.

Statquest: Decision Trees [https://www.youtube.com/watch?v=7VeUPuFGJHk&ab\\_channel=StatQuestwithJoshStarmer](https://www.youtube.com/watch?v=7VeUPuFGJHk&ab_channel=StatQuestwithJoshStarmer)

<https://towardsdatascience.com/decision-tree-ensembles-bagging-and-boosting-266a8ba60fd9>

<https://medium.com/swlh/random-forest-classification-and-its-implementation-d5d840d9bead0>

Hua, Jianping, et al. "Optimal number of features as a function of sample size for various classification rules." *Bioinformatics* 21.8 (2005): 1509-1515.

<https://datascience.stackexchange.com/questions/11390/any-rules-of-thumb-on-number-of-features-versus-number-of-instances-small-da>

<https://graphite-note.com/how-much-data-is-needed-for-machine-learning#:~:text=A%20rule%2Dof%2Dthumb%20approach,types%20of%20m>  
achine%2Dlearning%20problems.

<https://h2o.ai/wiki/feature-selection/#:~:text=In%20the%20machine%20learning%20process,why%20feature%20selection%20is%20important>.

<https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/>

<https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f#:~:text=Feature%20selection%20using%20Random%20f>  
orest%20comes%20under%20the%20category%20of,built%2Din%20feature%20selection%20methods.

<http://www.cs.man.ac.uk/~nogueirs/files/SLIDES-master.pdf>

[https://www.researchgate.net/figure/Merits-and-demerits-of-filter-wrapper-and-embedded-feature-selection-methods\\_tbl1\\_313098800](https://www.researchgate.net/figure/Merits-and-demerits-of-filter-wrapper-and-embedded-feature-selection-methods_tbl1_313098800)

<https://www.yourdatateacher.com/2021/10/11/feature-selection-with-random-forest/>

<https://www.kaggle.com/code/prashant111/random-forest-classifier-feature-importance>

<https://towardsdatascience.com/feature-selection-with-boruta-in-python-676e3877e596>

<https://www.listendata.com/2017/05/feature-selection-boruta-package.html>

# Sources cont.

<https://www.displayr.com/how-is-variable-importance-calculated-for-a-random-forest/>

<https://towardsdatascience.com/best-practice-to-calculate-and-interpret-model-feature-importance-14f0e11ee660>

<https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3>

<https://korediantousman.staff.telkomuniversity.ac.id/files/2017/08/main-1.pdf>

<https://www.statisticshowto.com/>

<https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/>

<https://datascience.stackexchange.com/questions/11390/any-rules-of-thumb-on-number-of-features-versus-number-of-instances-small-da>

[https://www.frontiersin.org/files/Articles/977747/fcvm-09-977747-HTML/image\\_m/fcvm-09-977747-g003.jpg](https://www.frontiersin.org/files/Articles/977747/fcvm-09-977747-HTML/image_m/fcvm-09-977747-g003.jpg)

<https://analyticsindiamag.com/hands-on-guide-to-automated-feature-selection-using-boruta/>