Machine Learning

February 2020

Valeria

Engineer
ITA

Senior Data Scientist
Nubank

For Fun
Investing
Ball Room
Video Games

Let's talk about today
ML principles based on my academic and professional experience...

Build foundations

Instigate curiosity

Best practices

Recommendations

Leading in **Data Science projects**, tools and tricks | Rafael Carrascosa

Challenge (business): What? Why?

Results and Impact: Business, Customers, Lives

Settings

Data Collection

Data Verification

Machine Learning

Machine Resource Management

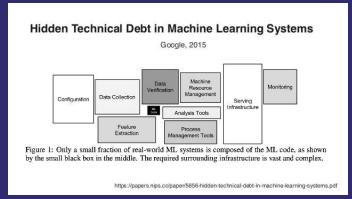
Serving

Feature extraction

Process management tools

Analysis Tools

Monitoring



Defining the target

What's being predicted?

How to convert a real-life problem into a solvable machine learning project?

Model Types

Learning

Evaluation

Metrics

Development

Techniques, strategies, drawbacks

LINEAR

VERSUS

ITERATIVE

Purpose is to entertain and teach.	Creative and experimental cycles.
Simplification	Real Life

Ready to predict and influence the **future?**!

Machine Learning is useful in mainly **2 types** of tasks

Scale tasks that humans are good at

- Is there a cat in the picture?
- Is this an angry tweet?
- What are the characters in this CAPTCHA?

Support decision making in tasks humans are not good at

- Which customers will pay the credit card bill?
- How sales will increase if I double marketing budget?
- What is the optimal credit limit for a given customer?

Model Types

Learning

SUPERVISED LEARNING

More than 80% of the cases

Unsupervised Learning
10%

Semi-Supervised

Recommender Systems (supervised)

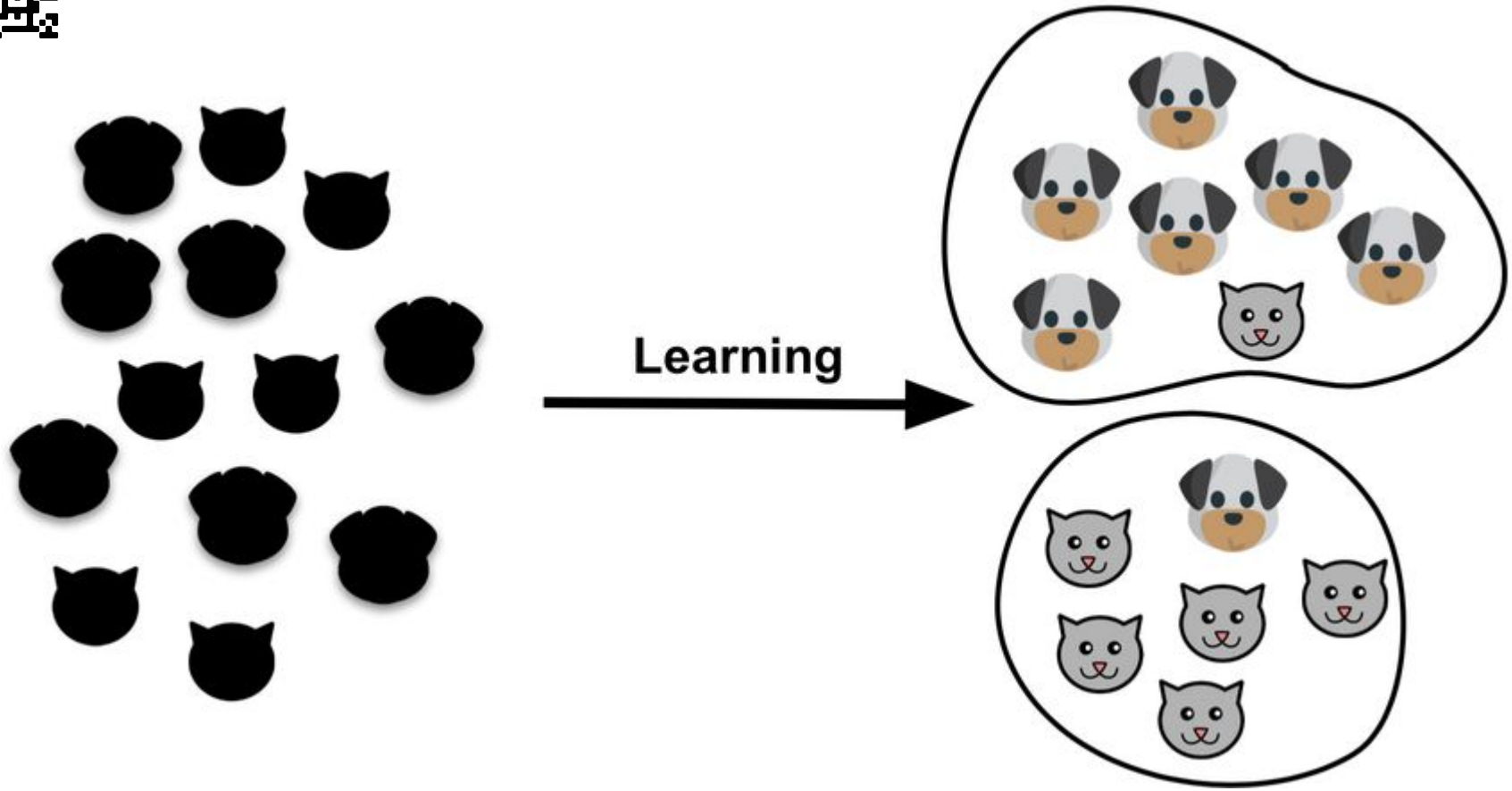
Reinforced Learning

Deep Learning

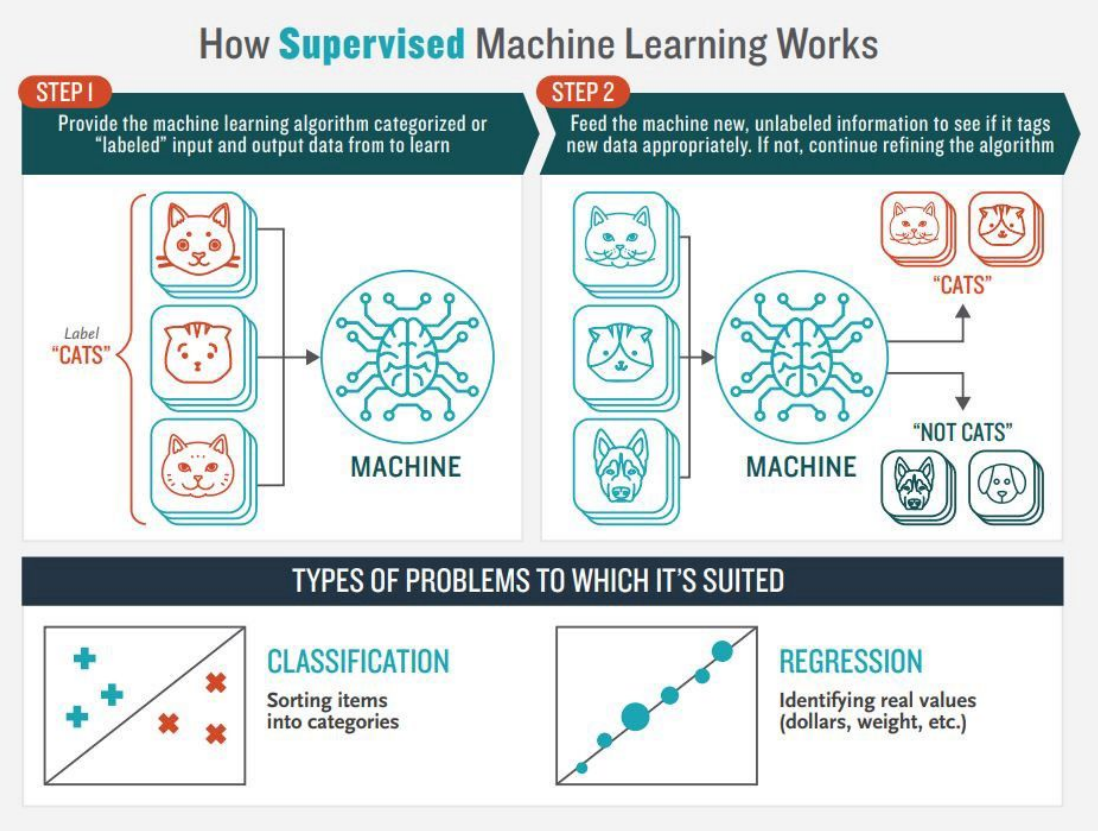
Survival (supervised)

Time Series (supervised)

UnSupervised Learning

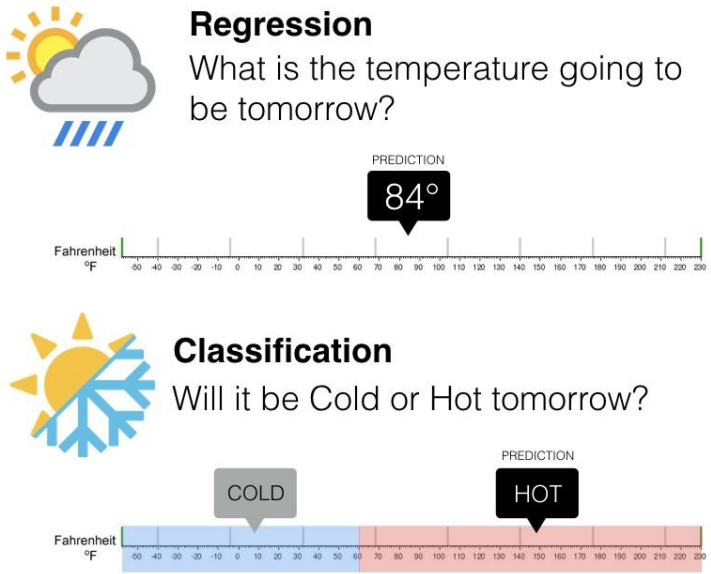


Supervised **Learning**



Supervised - Main types of models

Regression & Classification



Evaluation

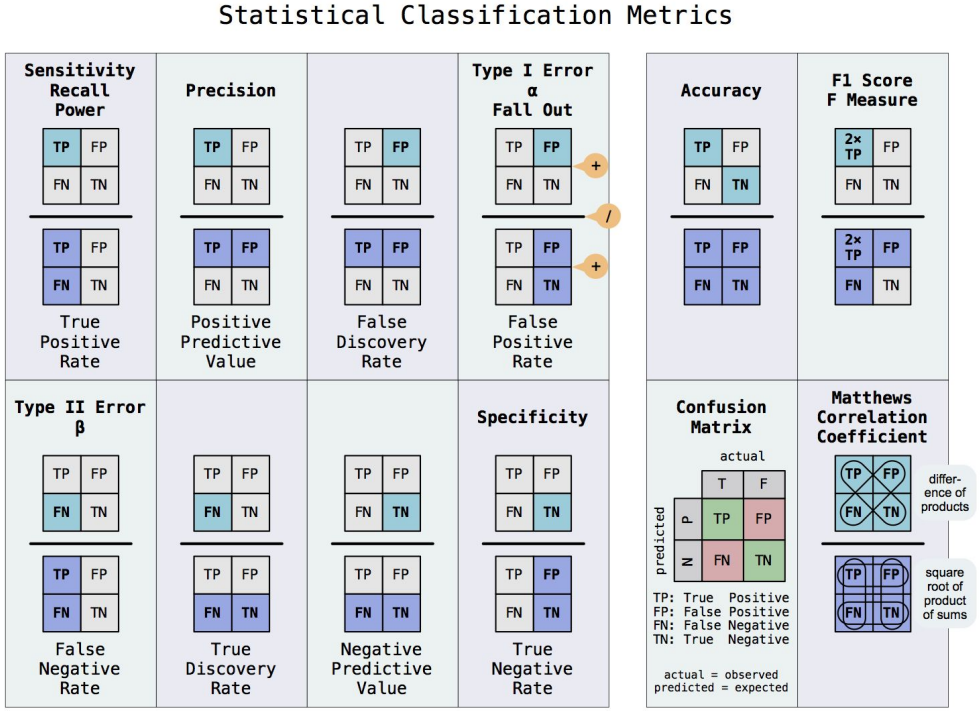
Metrics

Regression metrics

Mean squared error	$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$
Mean absolute percentage error	$\text{MAPE} = \frac{100\%}{n}\sum_{t=1}^{n} \left\lvert \frac{e_t}{y_t} \right\rvert$

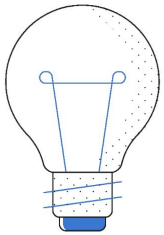
Regression
Even if your model is robust to outliers, the metric may not be.

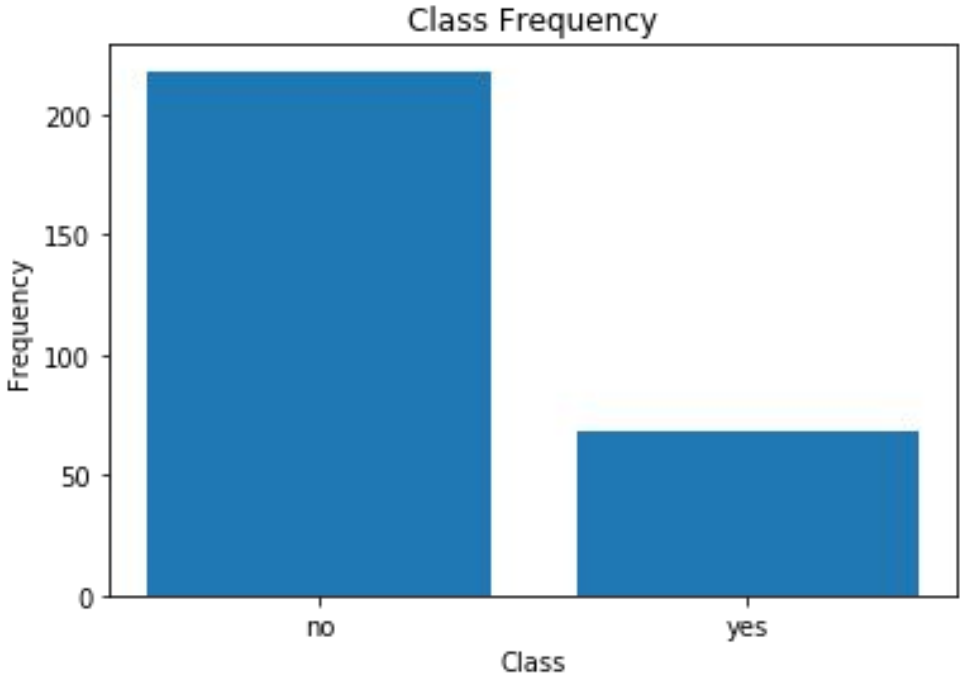
Classification metrics



Imbalanced data

Challenge





Accuracy

Classification

Metric

Accuracy

Confusion matrix — The confusion matrix is used to have a more complete picture when assessing the performance of a model. It is defined as follows:

		Predicted class +	**Predicted** class -
Actual class +		**TP** True Positives	**FN** False Negatives Type II error
Actual class -		**FP** False Positives Type I error	**TN** True Negatives

Main metrics — The following metrics are commonly used to assess the performance of classification models:

Metric	Formula	Interpretation
Accuracy	$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$	Overall performance of model

Unbalanced example:
- 99% >> 0
- 1% >> 1

Model
- Constant **0** prediction

Accuracy
- 99%

Stanford University

Accuracy

Classification

Metric

Confusion mat …he performance of a model. It is defi…

Main metrics — The following metrics are commonly used to assess the performance of classification models:

Metric	Formula	Interpretation
Accuracy	$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$	Overall performance of model

Unbalanced example:
- 99% >> 0
- 1% >> 1

Model
- Constant **0** prediction

Accuracy
- 99%

Stanford University

Precision-Recall

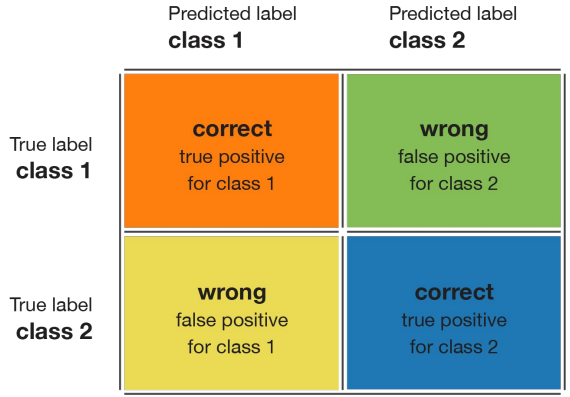
Classification

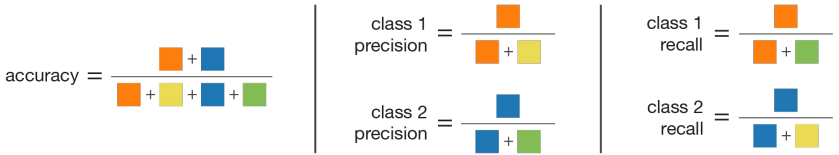
fb-score

Recall | Precision

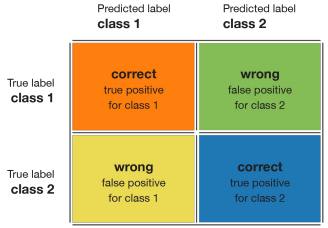
- high recall + high precision : the class is perfectly handled by the model
- low recall + high precision : the model can't detect the class well but is highly trustable when it does
- high recall + low precision : the class is well detected but the model also include points of other classes in it
- low recall + low precision : the class is poorly handled by the model

$$F_{Beta} = (\beta^2 + 1)\frac{Precision \cdot Recall}{(\beta^2 Precision) + Recall}$$





AUC
Classification



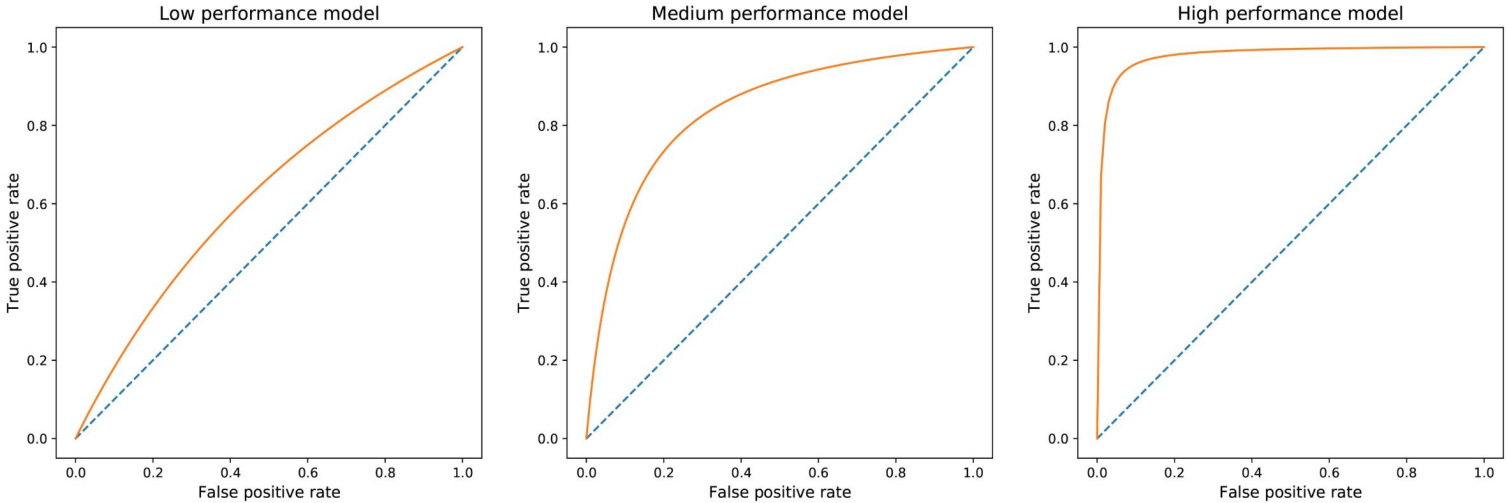


Illustration of possible ROC curves depending on the effectiveness of the model. On the left, the model has to sacrifice a lot of precision to get a high recall. On the right, the model is highly effective: it can reach a high recall while keeping a high precision.

Kaggle Forum
Events and topics specific to our community

Log0

Precision-Recall AUC vs ROC AUC for class imbalance problems

20

posted in Kaggle Forum 6 years ago

Hi all,

I've been reading the paper "The Relationship Between Precision-Recall and ROC Curves" recently, which argues that at problems suffering from class imbalance problem, using an evaluation metric of Precision-Recall AUC (PR AUC) is better than Receiver-Operating-Characteristic AUC (ROC AUC).

The paper states that "A large number change in the number of false positives can lead to a small change in the false positive rate used in ROC analysis. Precision, on the other hand, by comparing false positives to true positives rather than true negatives, captures the effect of the large number of negative examples on the algorithm's performance."

My questions:

- For ROC, FP is captured in the False Positive Rate (FPR); For PR, FP is captured by the Precision. If it is a metric that is captured already in the values to be plotted, why would PR beats ROC?
- For the experienced pros, what would you recommend and why? How different are they in practice?

There wasn't really any mathematical proof to back the paper's claim up. I am a bit skeptical since there

Usual misconception

"**Classification** models output **probabilities**."

"**Regression** models output the **exact value we want to predict.**"

Truth

Normally an additional layer of data of treatment is necessary to convert model predictions into business decisions.

Solution

Business rules

and/or

Calibration

Usual misconception

"**Classi on** models output **p bilities**."

"**Regre io** models output th **xact value we wan to predict.**"

Truth

Normally an additional layer of data of treatment is necessary to convert model predictions into business decisions.

Solution

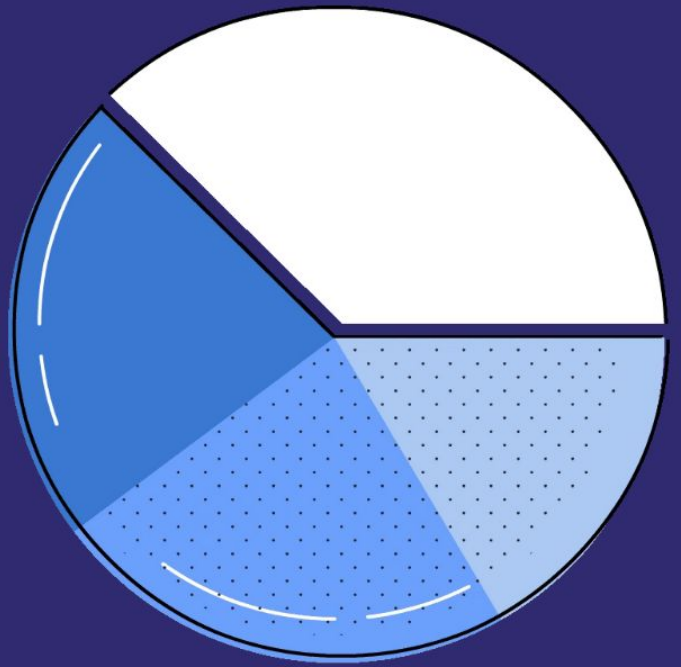
Business rules

and/or

Calibration

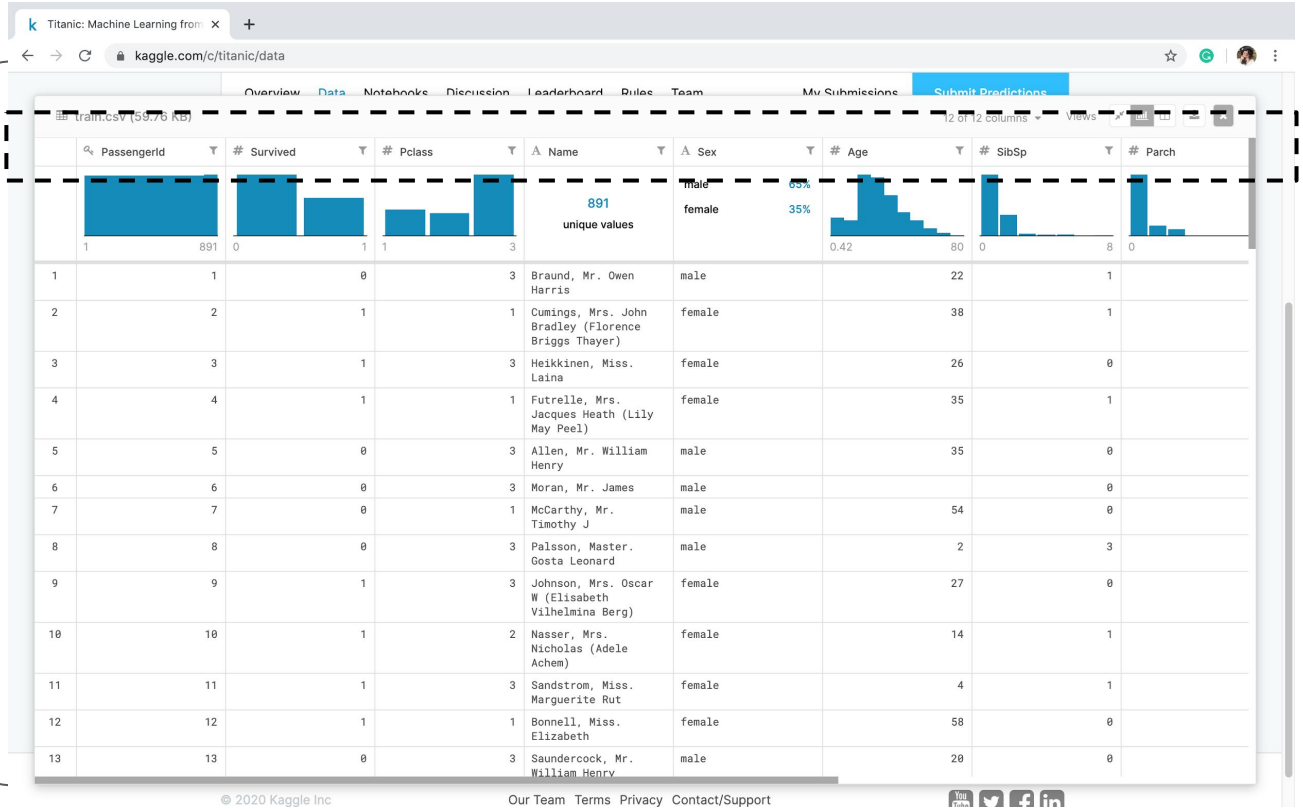
Data

Development



Structured Data

rows
-
sample



columns
-
features

Splits

Validation

Overfit and Underfit

Analogy - Doing tests at School

Overfit

- Memorized the exercises
- Can only reproduce what learned in the past
- Cannot generalize for unseen questions

Optimum

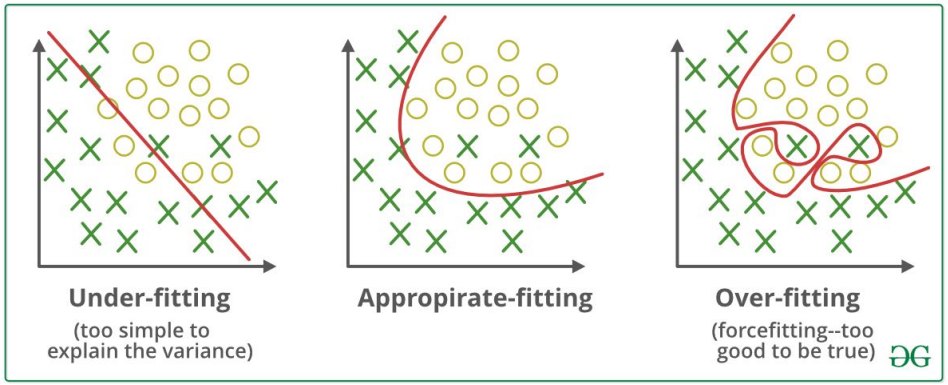
Learned.

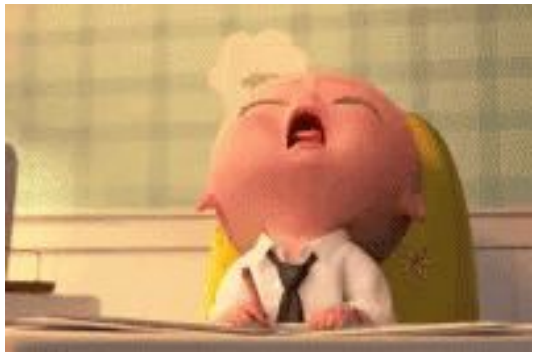
Underfit

- Did not prepare for the test
- Does not know the subject for the test
- Can not infer from previous experiences

Overfit and Underfit

Failure





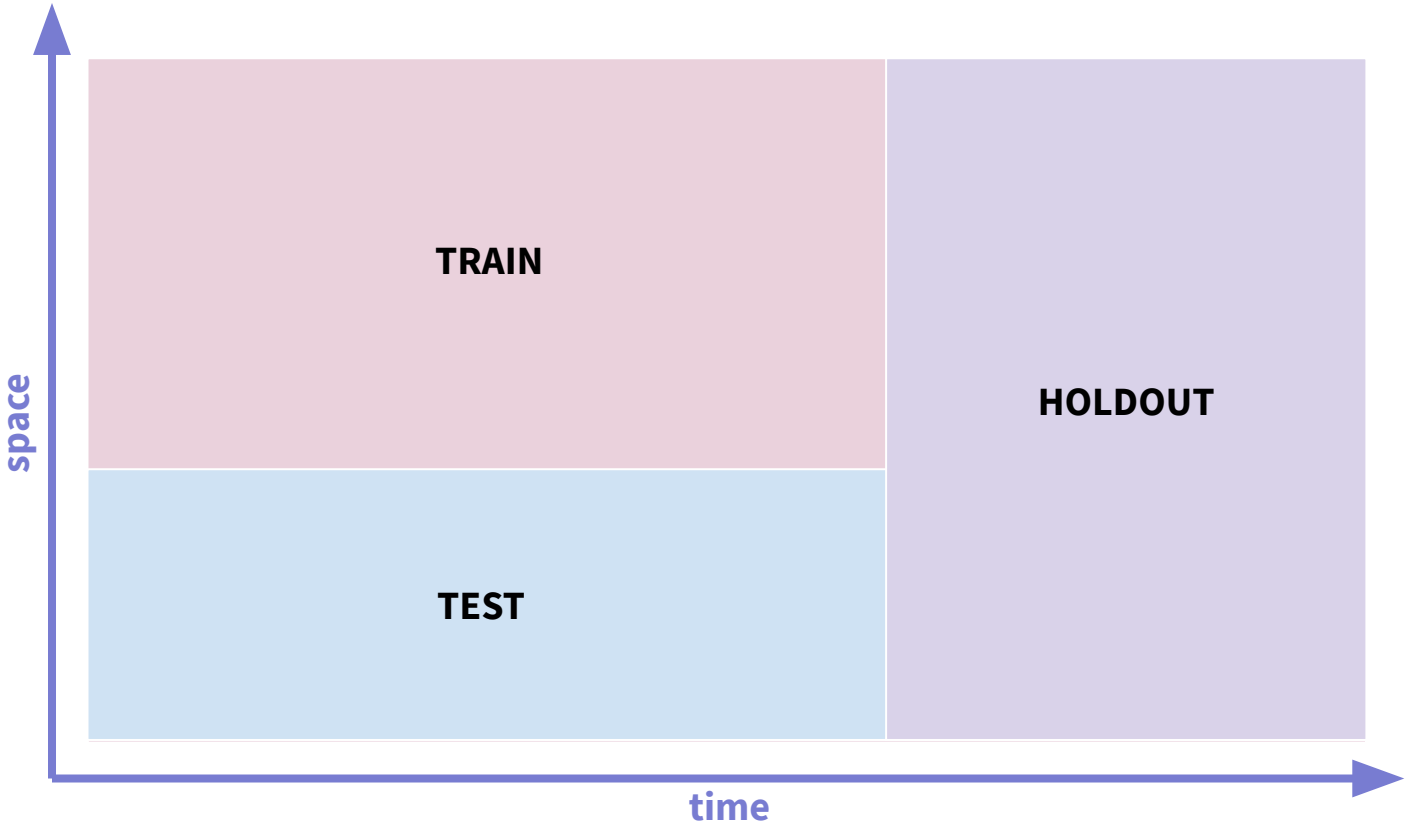
OVERFITTING:

- Memorized the train, but can reproduce it in new data
  - Models prone to:
    - Boosting
    - Random forest
    - Deep Learning
    - Neural nets

UNDERFITTING:

- Not doing a good job at learning in the training
  - Models prone to:
    - Decision tree
    - Logistic Regression

Time and Space Splits

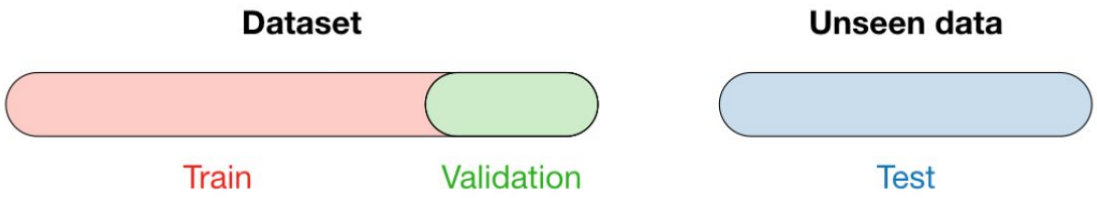


Model selection

Vocabulary — When selecting a model, we distinguish 3 different parts of the data that we have as follows:

Training set	Validation set	Testing set
• Model is trained • Usually 80% of the dataset	• Model is assessed • Usually 20% of the dataset • Also called hold-out or development set	• Model gives predictions • Unseen data

Once the model has been chosen, it is trained on the entire dataset and tested on the unseen test set. These are represented in the figure below:



But, why??? Leakage

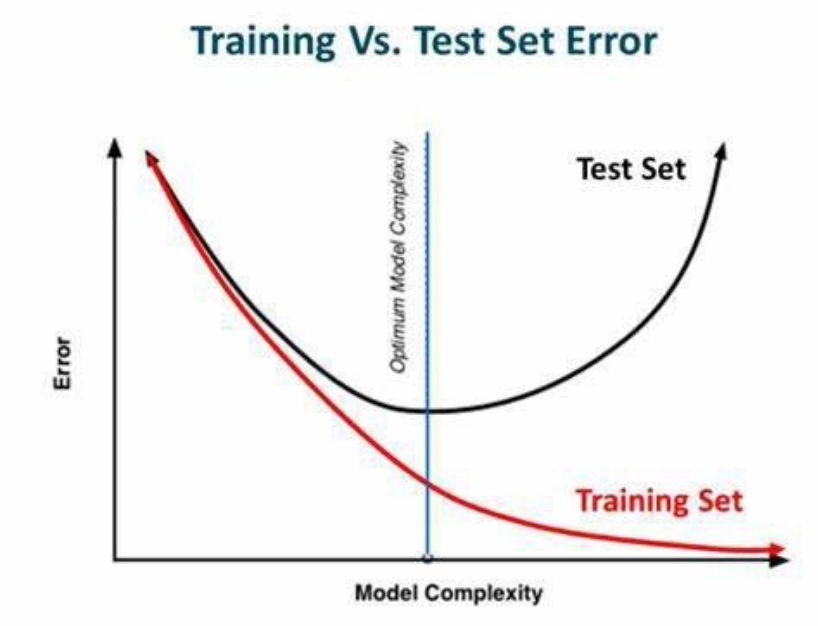
Huge Failure

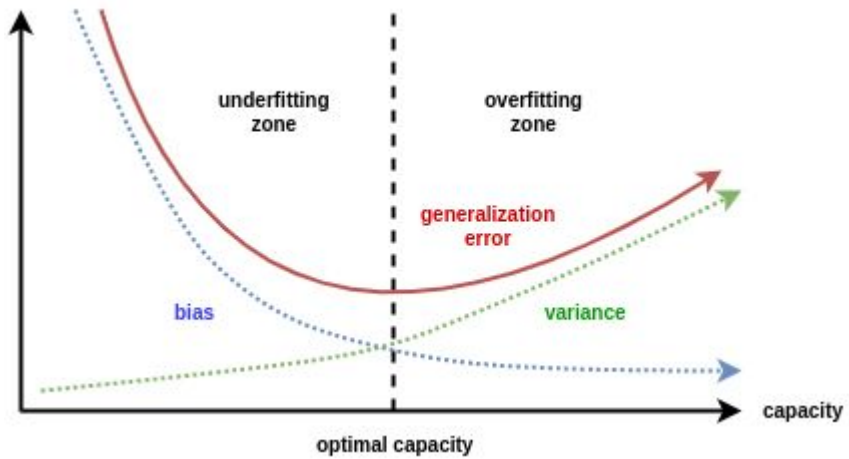
- Giving information that the model should not have for any reason. Usually: **SPACE** and/or **TIME.**
  - Space:
    - Data Duplicity
  - Time:
    - Data From the future

- **RISK:** When you do it, you are deceiving yourself because the increase in performance won't be seen in production.

Early Stopping

A strategy for preventing overfit

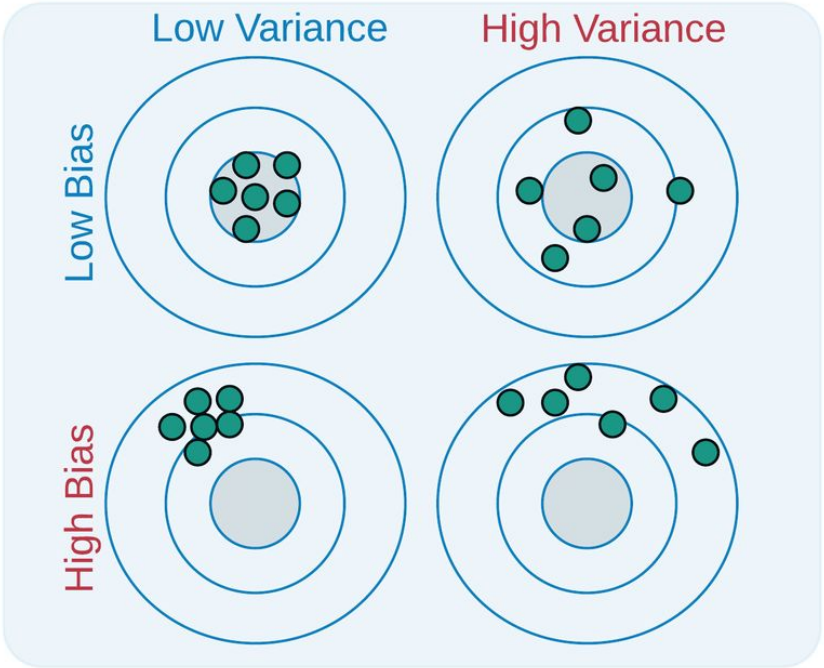




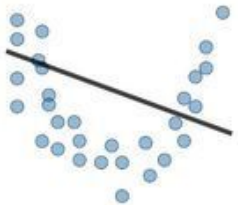
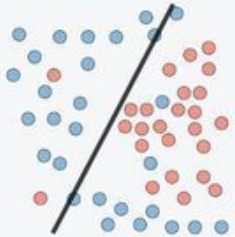
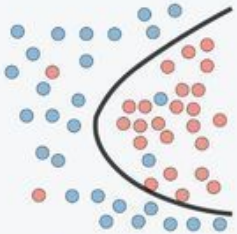
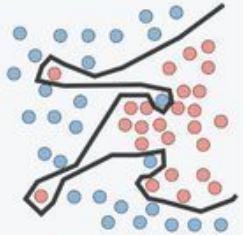
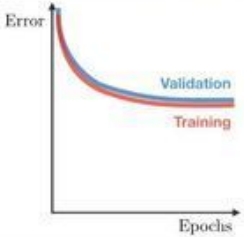
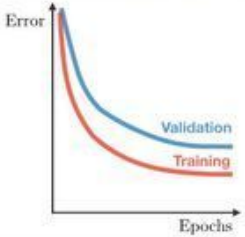
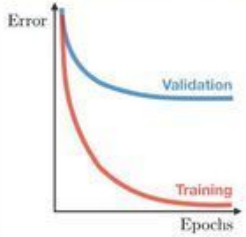
The simpler the model, the higher the bias, and the more complex the model, the higher the variance.

Bias Variance

Trade-off



The simpler the model, the higher the bias, and the more complex the model, the higher the variance.

	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

First Step: Baseline

What?

The simplest model you can build:

- Simple average
- Constant Value
- Decision tree
- Tabular data:
  - LightGBM, XGBoost, Catboost

Why?

Data Science is SCIENCE

- Experimentation, tests
- Uncertainty

So what?

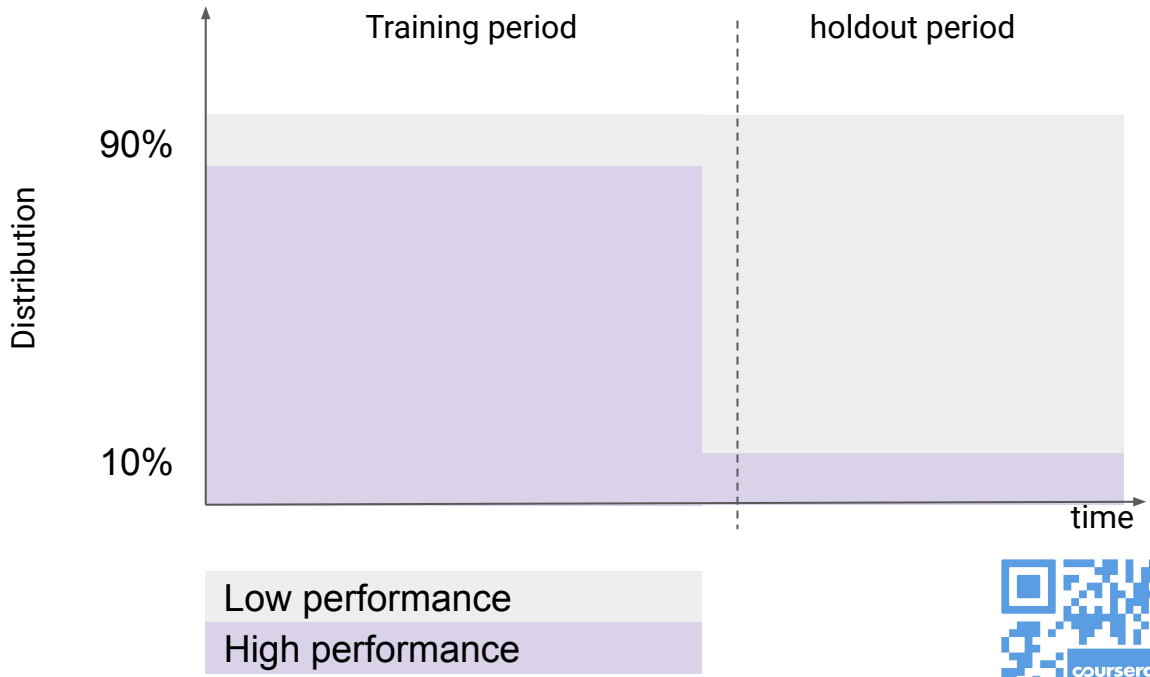
- Reduced Uncertainty
- Easier to prioritize
- Reasonable decisions about Result and Complexity

Exploratory Data Analysis

Why?

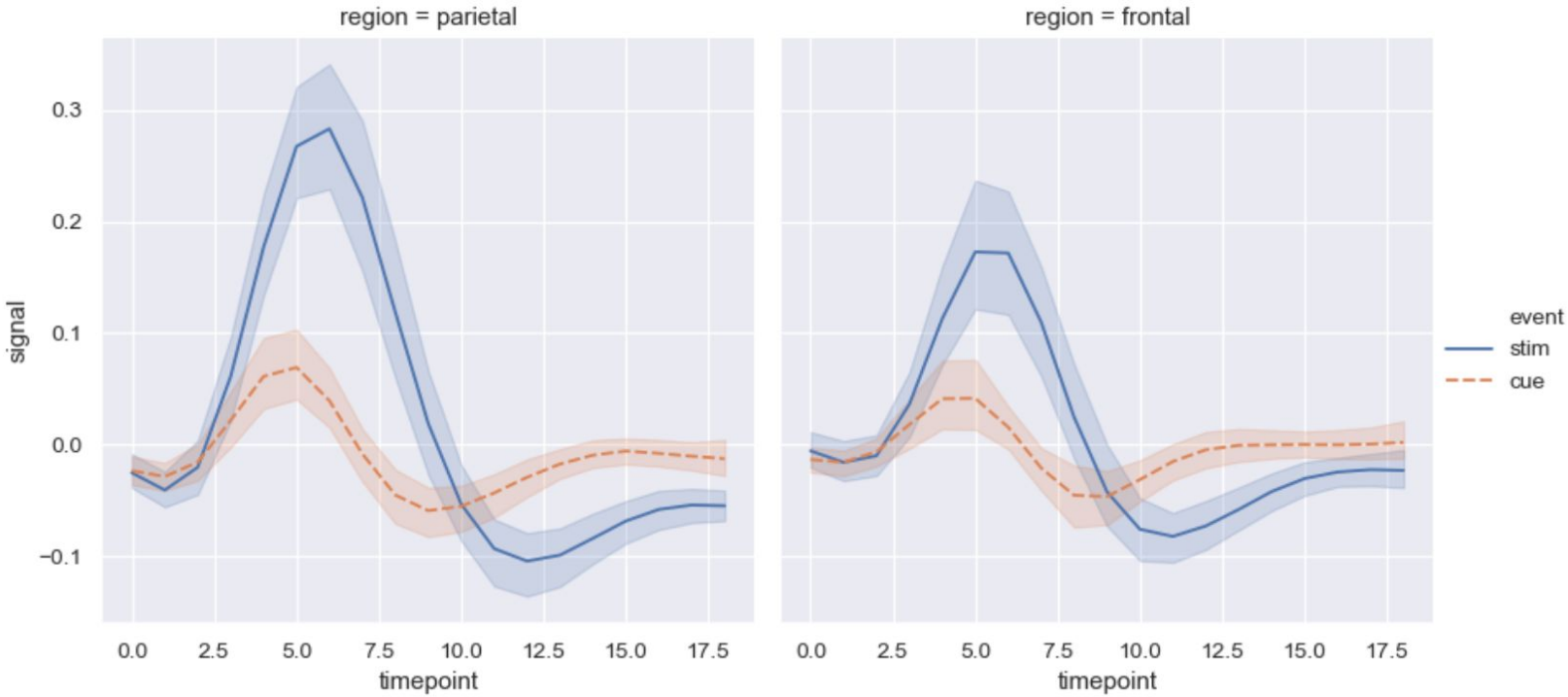
Exploratory Data Analysis

- Better understand data
- Build intuition
- Generate hypothesis
- Find Insights



Stability - Line plot

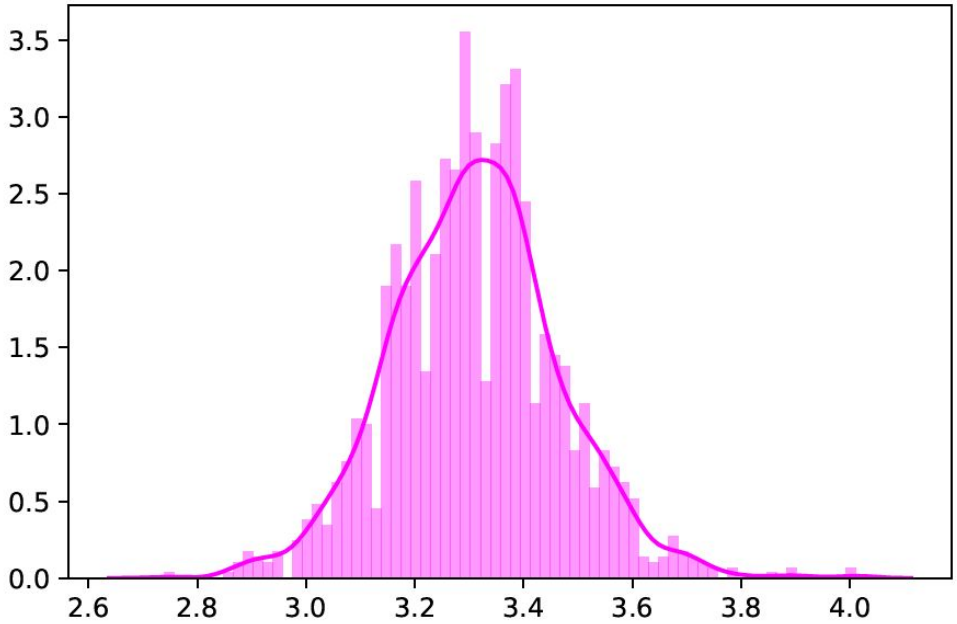
Exploratory Data Analysis



seaborn

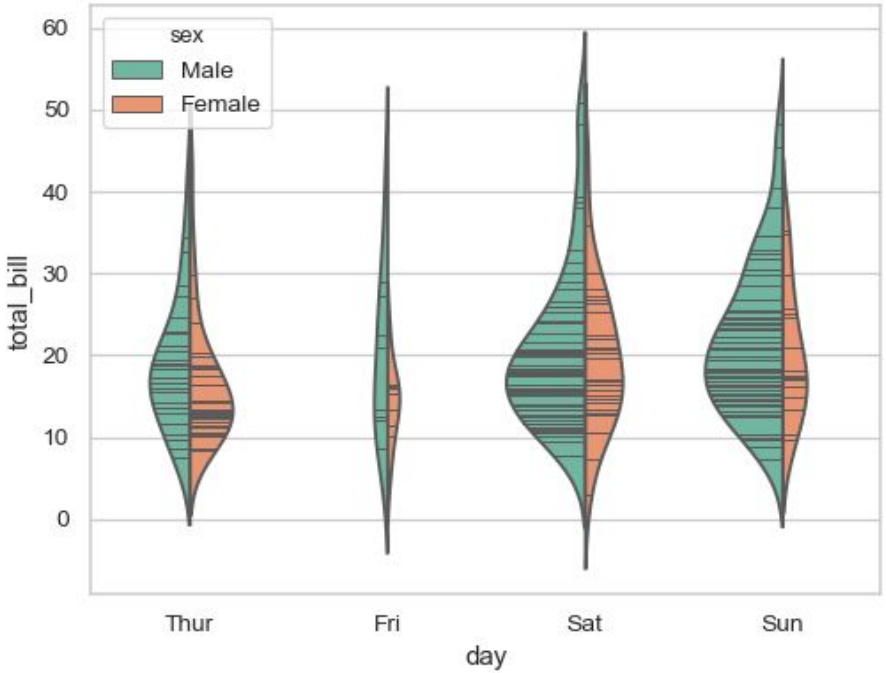
Histogram

Exploratory Data Analysis



Violin Plot

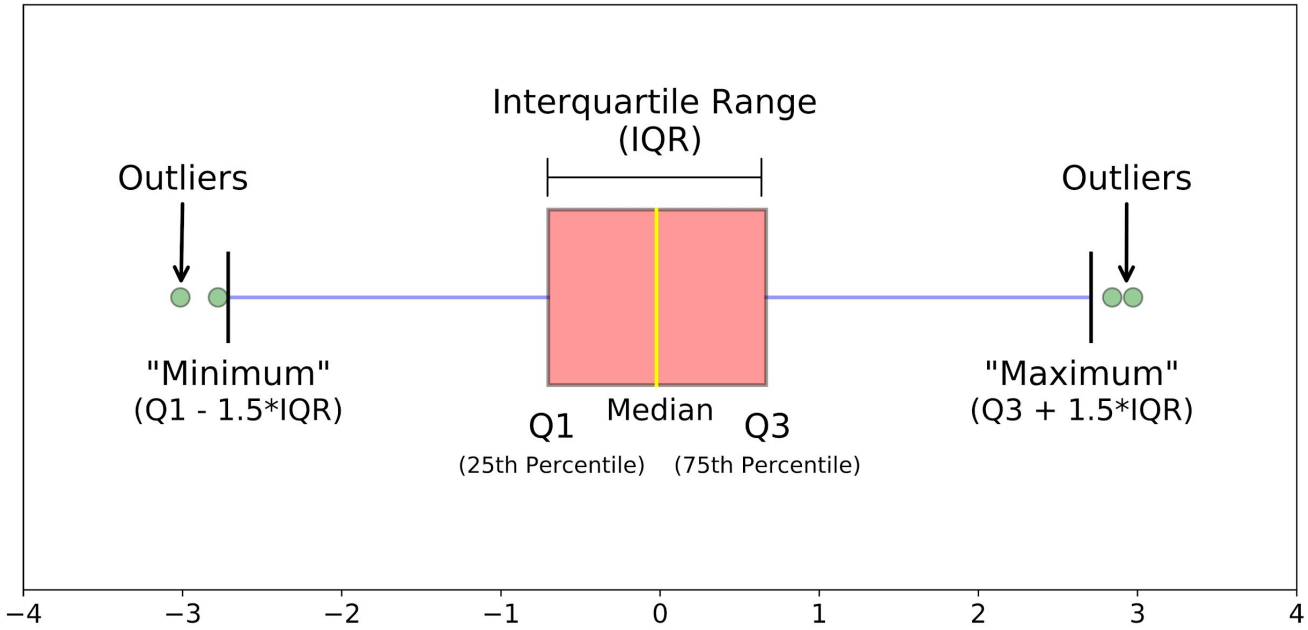
Exploratory Data Analysis



seaborn

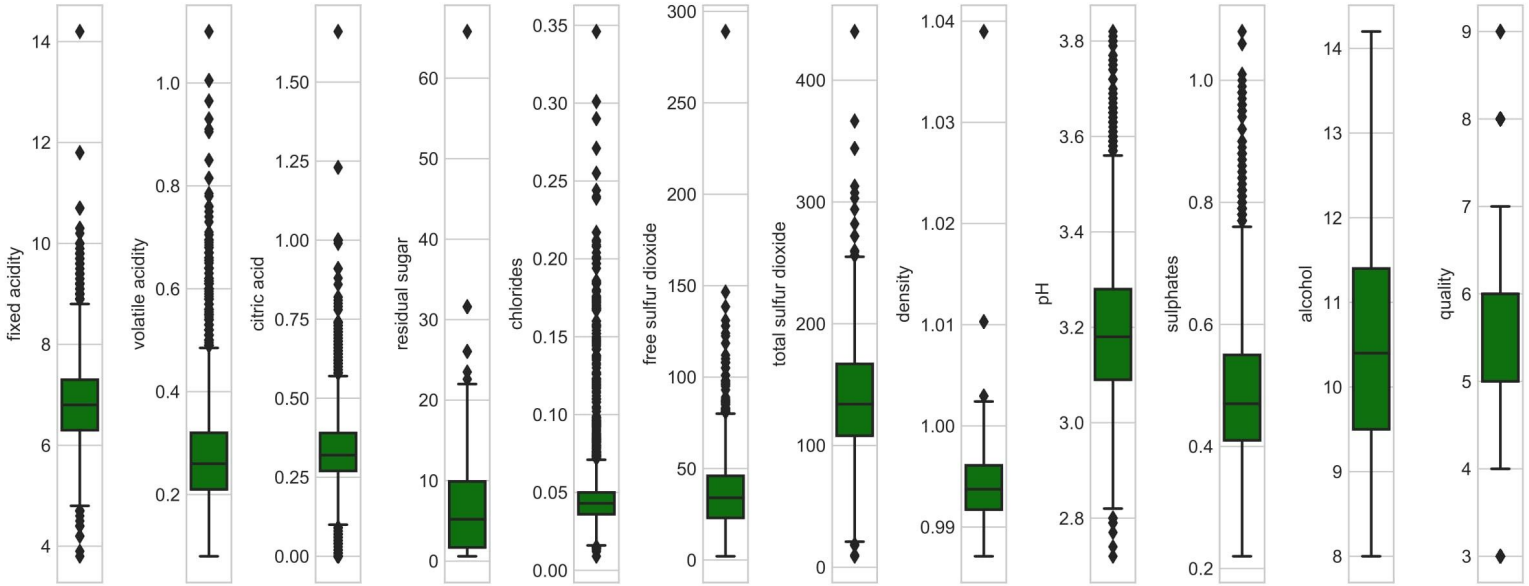
Boxplot

Exploratory Data Analysis



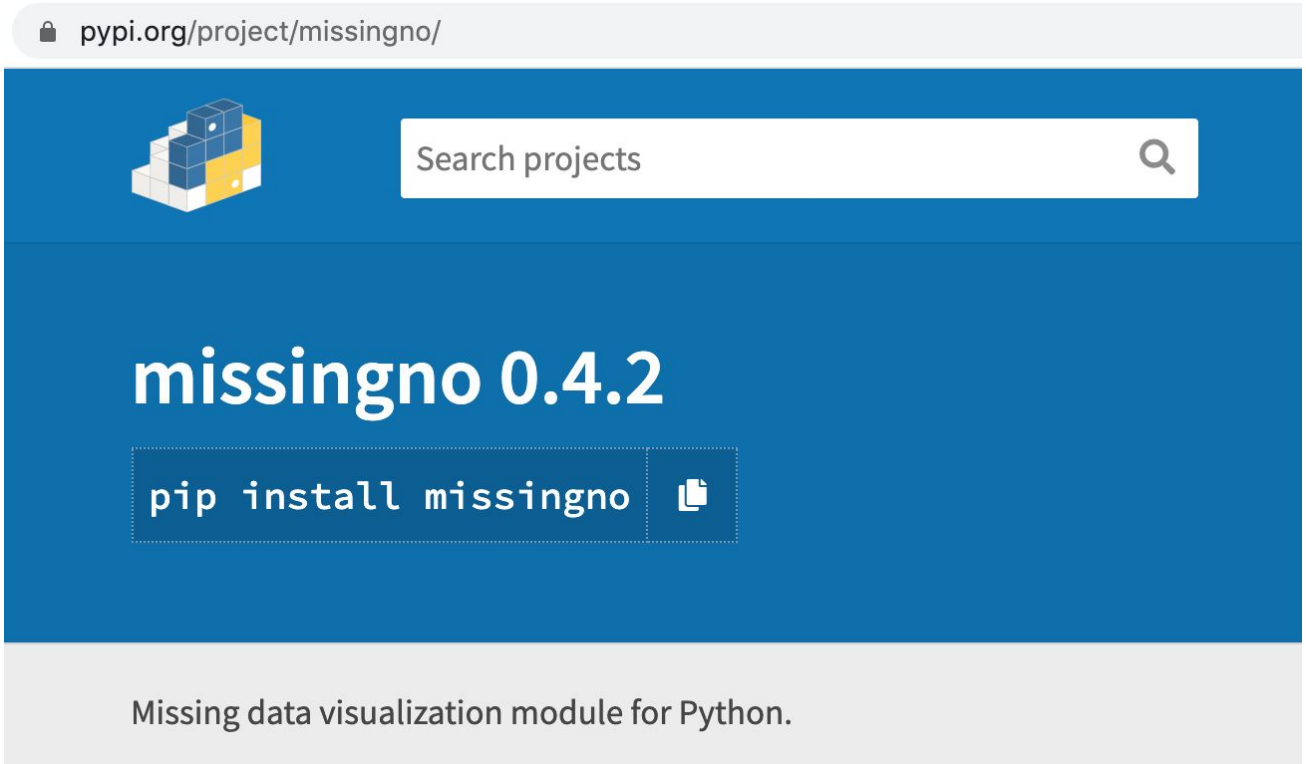
Boxplot

Exploratory Data Analysis



Missing

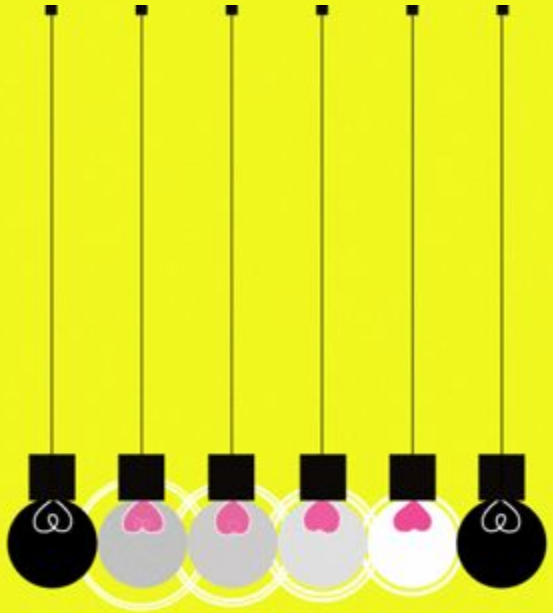
Exploratory Data Analysis



Feature Engineering

Feature engineering

Model choice



Appropriate feature engineering depends on the model choice.

Encoding

Categorical features

HJ Van Veen

Label encoding

TV show	Categorical#	Score
Dexter	1	10
Dark	2	9.5
La casa de Papel	3	9

One hot encoding

Dexter	Dark	La casa de papel	Score
1	0	0	10
0	1	0	9.5
0	0	1	9

HJ Van Veen

Count Encoding

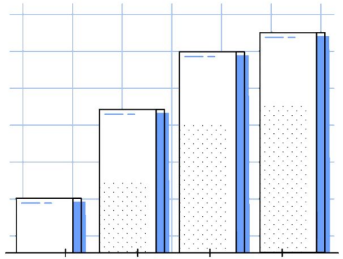
Categorical features

- Replace categorical variables with their count in the training set
- Useful for both linear and non-linear algorithms
- Can be sensitive to outliers
- May add log-transform, works well with counts
- Replace unseen variables with `1`
- May give collisions: same encoding, different variables

HJ Van Veen

Target encoding

Categorical features



- **Encode categorical variables by their ratio of target (binary classification or regression)**
- Form of stacking: single-variable model which outputs average target
- Do in cross-validation manner
- Add smoothing to avoid setting variable encodings to 0
- Add random noise to combat overfit
- When applied properly: Best encoding for both linear and nonlinear

Scaling

Numerical features

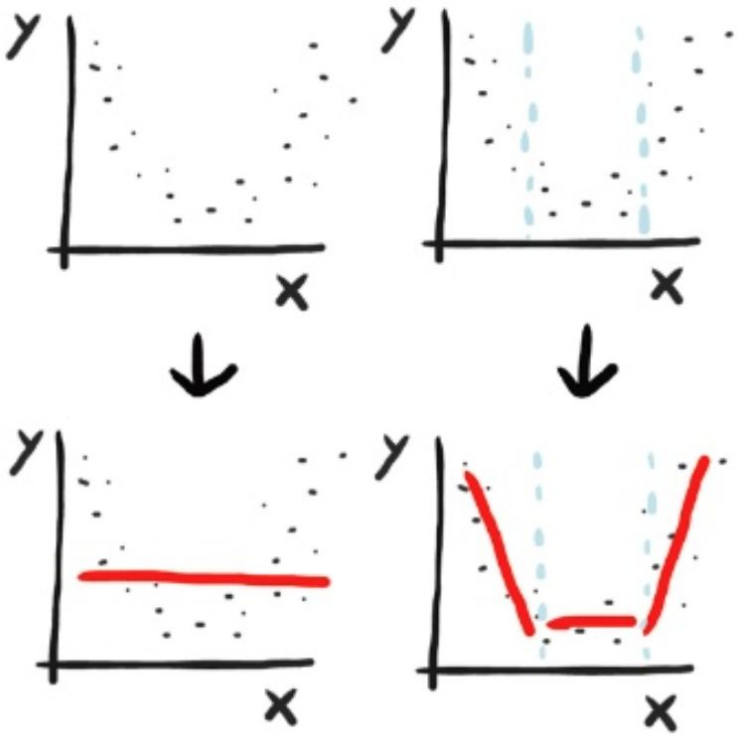
HJ Van Veen

- Scale to numerical variables into a certain range
- Standard (Z) Scaling
- MinMax Scaling
- Root scaling
- Log scaling

Binning

Numerical features

HJ Van Veen

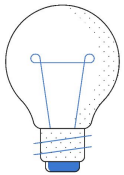


HJ Van Veen

Missing values imputation

Feature engineering

Missing values may be valuable info
>> Create dummy features is a viable option.



Mode

Median

Mean

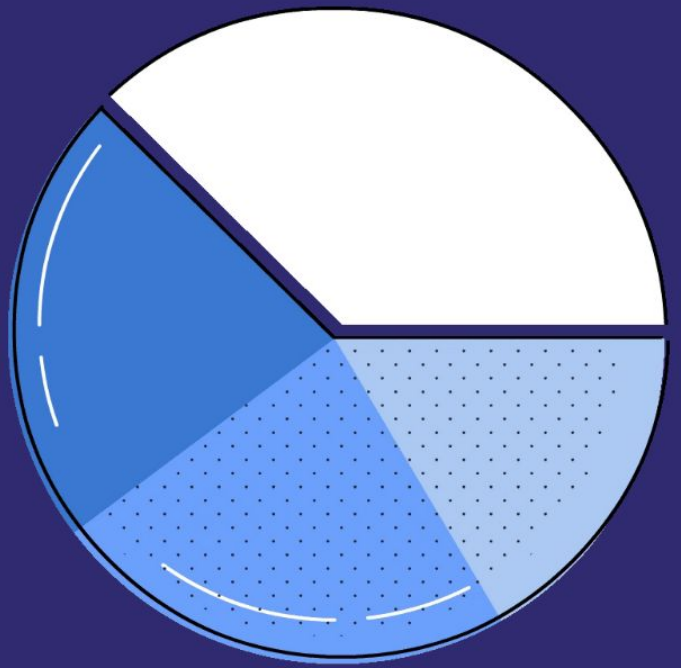
Avoid if outliers

Extreme values

Avoid if model or metric not robust to outliers

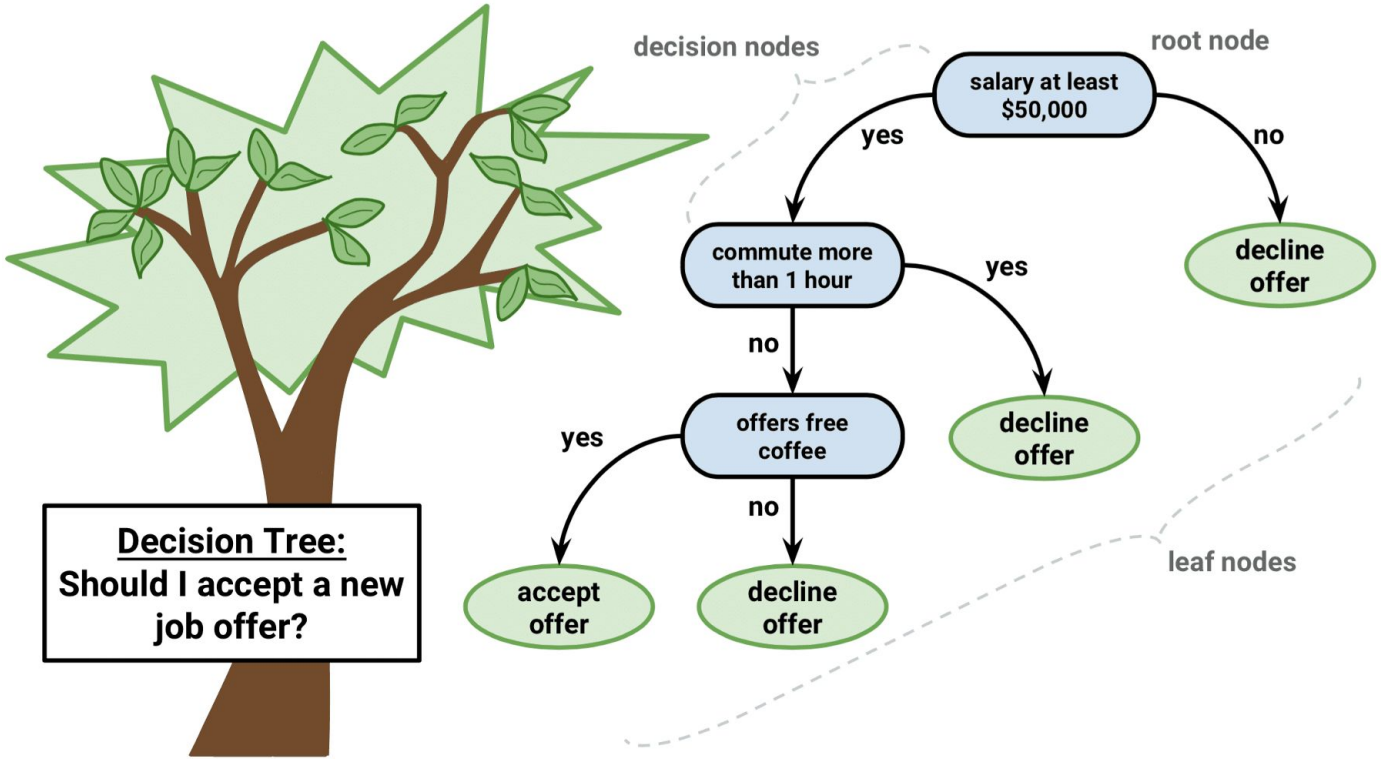
Model

Development



Decision trees

Model Development



Linear models

Model Development

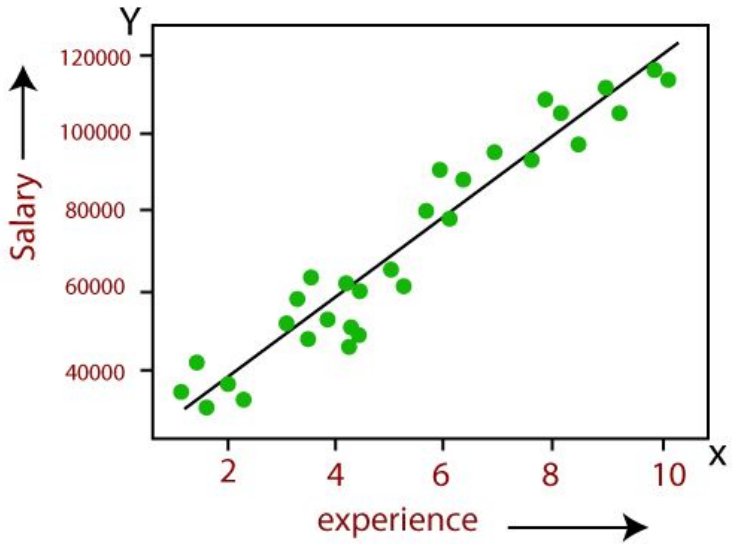
Linear Regression: Single Variable

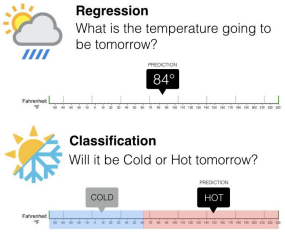
$$\hat{y} = \beta_0 + \beta_1 x + \epsilon$$

Predicted output — Coefficients — Input — Error

Linear Regression: Multiple Variables

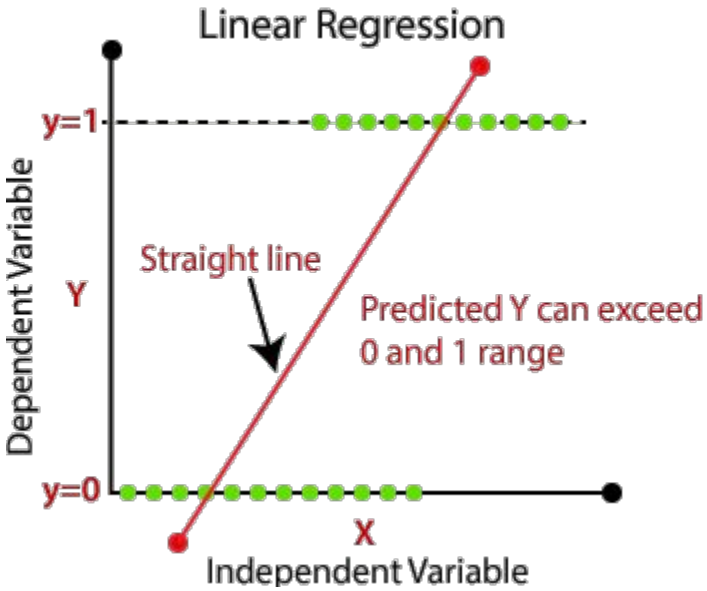
$$\hat{y} = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \epsilon$$



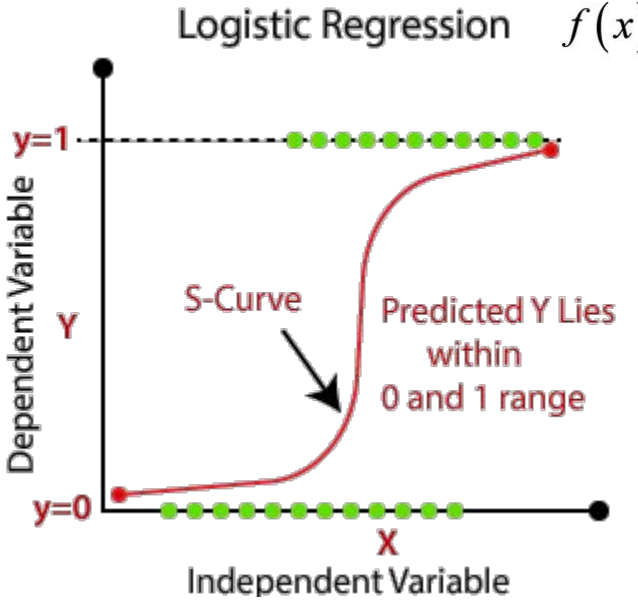


Linear models

Model Development



Regression model



Classification model

Tree based vs. Linear

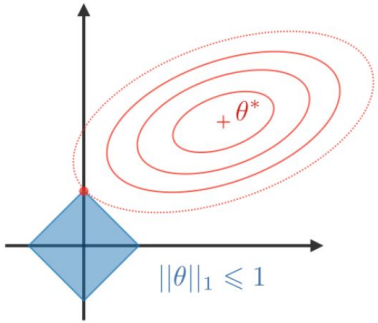
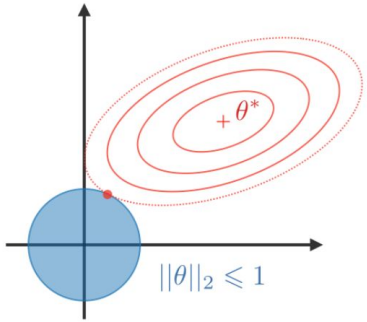
Model Development

	Tree	Linear
Is Robust to outliers?	Yes	No
Standard Scaling?	No	Yes

Regularization

Solutions

Regularization — The regularization procedure aims at avoiding the model to overfit the data and thus deals with high variance issues. The following table sums up the different types of commonly used regularization techniques:

LASSO	Ridge
• Shrinks coefficients to 0 • Good for variable selection	Makes coefficients smaller
	
$\ldots + \lambda \|\|\theta\|\|_1$ $\lambda \in \mathbb{R}$	$\ldots + \lambda \|\|\theta\|\|_2^2$ $\lambda \in \mathbb{R}$

Ensemble

Development

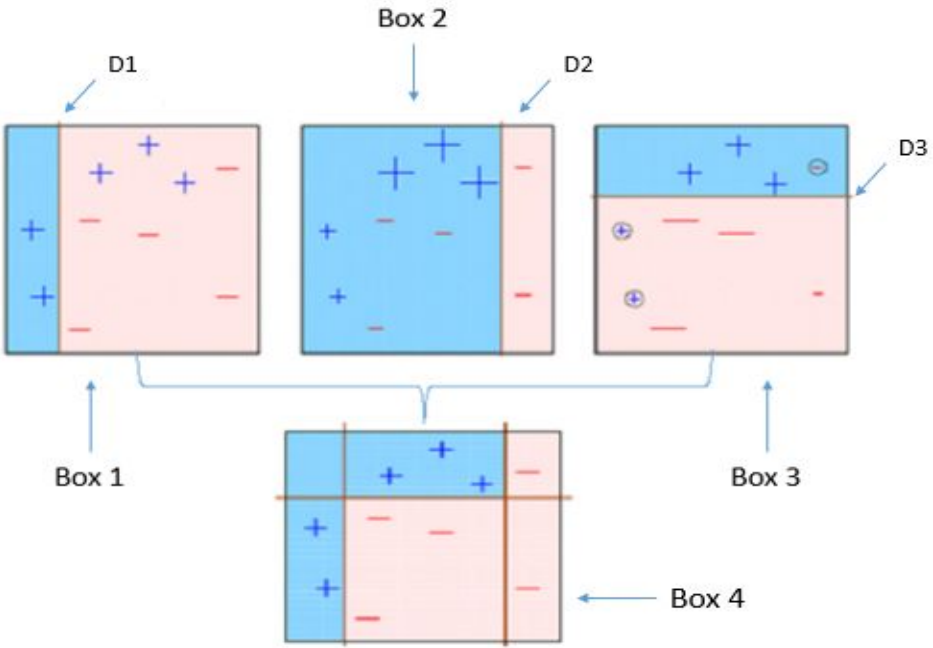
Ensembling

Analogy - College entrance exam

Average	7.1	7.3	7.5	7.8	7.4	7.0	7.1
Best Score	Math 9.5	Physics 9.9	English 9.8	History 9.3	Chem. 9.3	CS 10.0	Stats 9.5

AdaBoost (Adaptive Boosting)

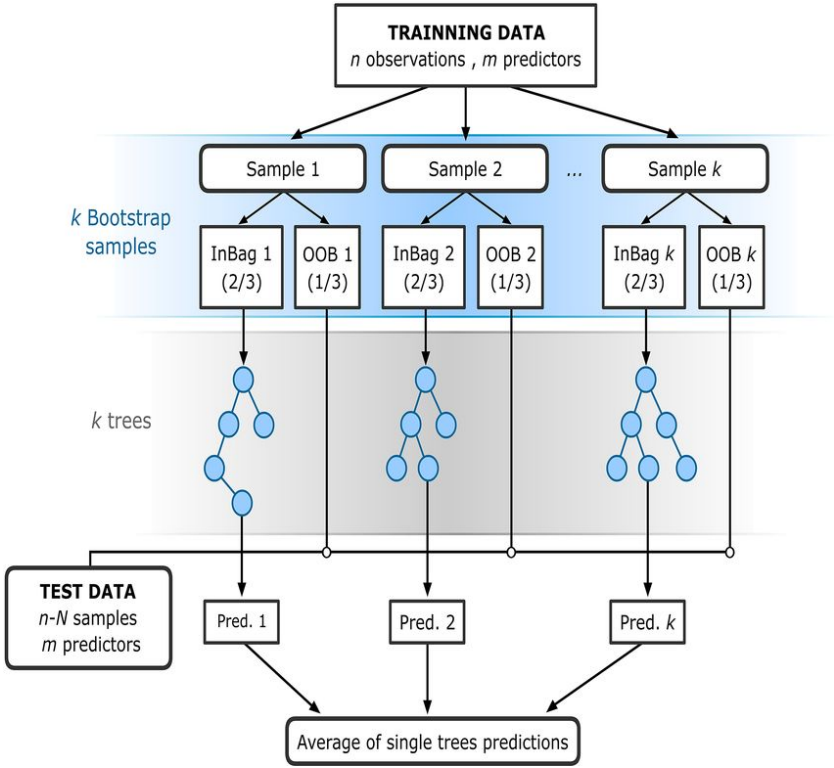
Ensembling



Prone to overfitting

Random Forest

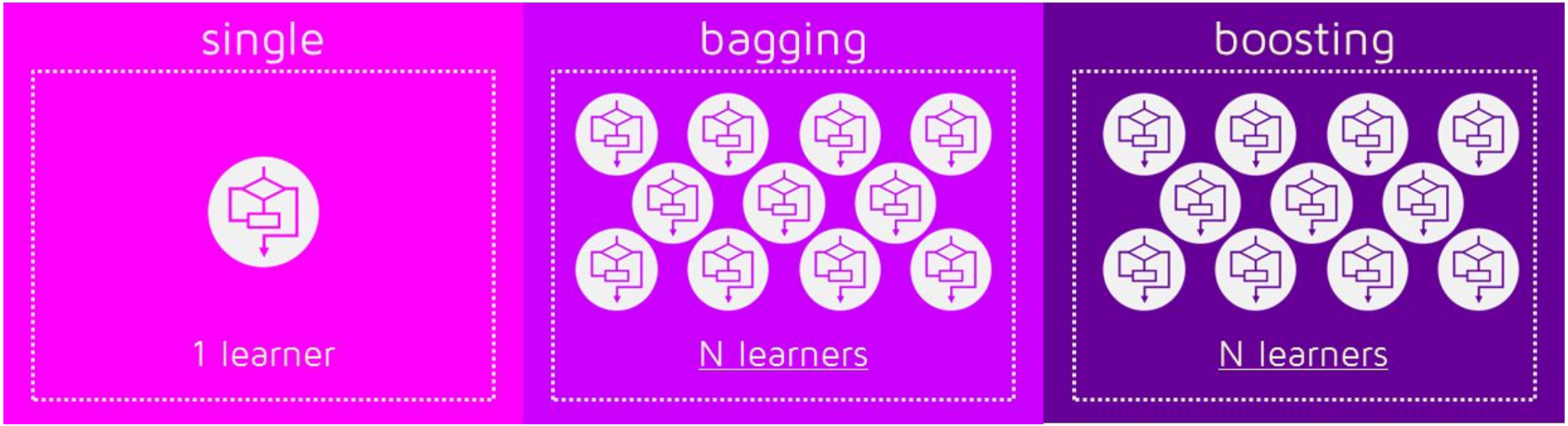
Ensembling



Random sample of features

Ensembling

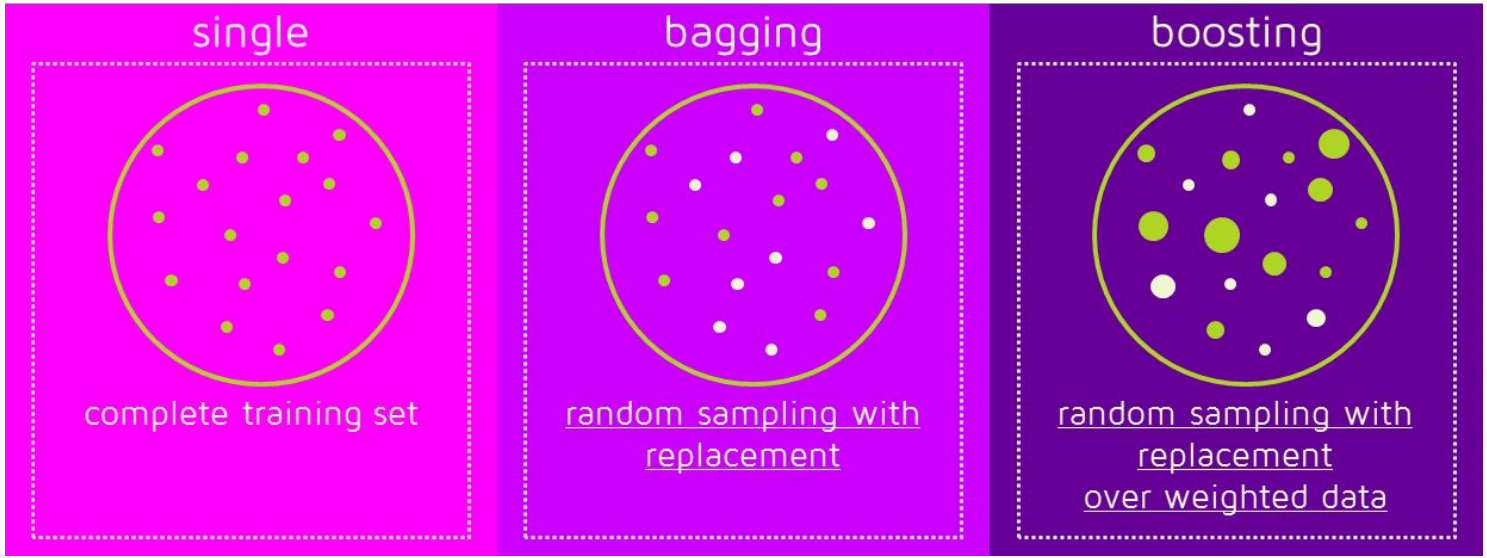
Model Development



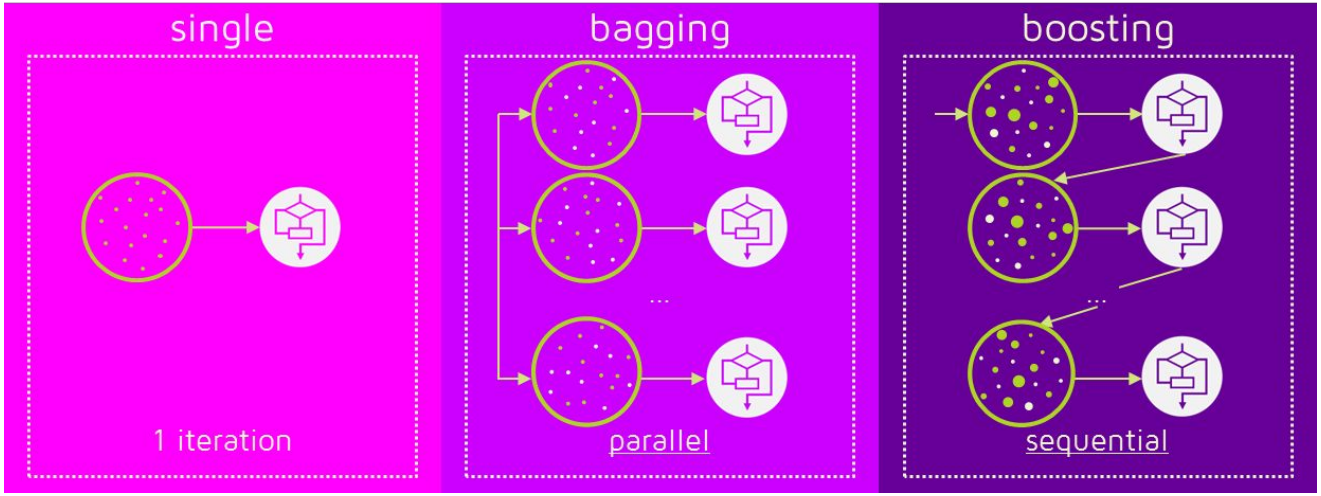
Source: Quantdare

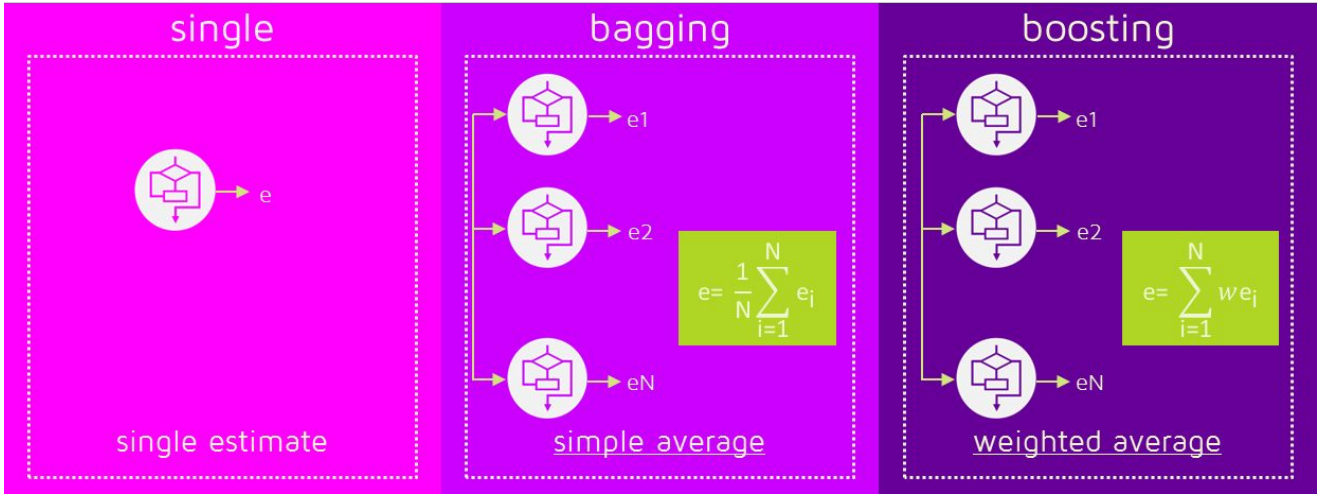
Ensembling

Model Development



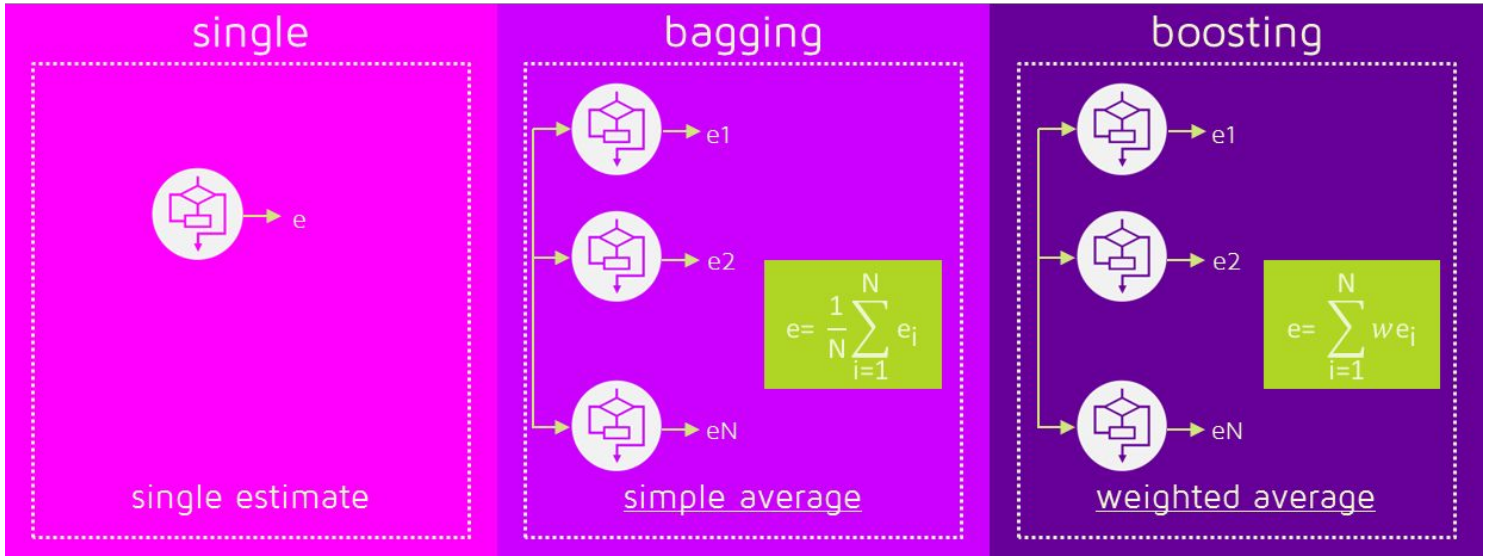
Source: Quantdare


single
1 iteration
bagging
parallel
boosting
sequential


single
e
single estimate
bagging
e1
e2
eN
$e = \frac{1}{N}\sum_{i=1}^{N} e_i$
simple average
boosting
e1
e2
eN
$e = \sum_{i=1}^{N} we_i$
weighted average

Ensembling

Model Development



Source: Quantdare

Ensembling

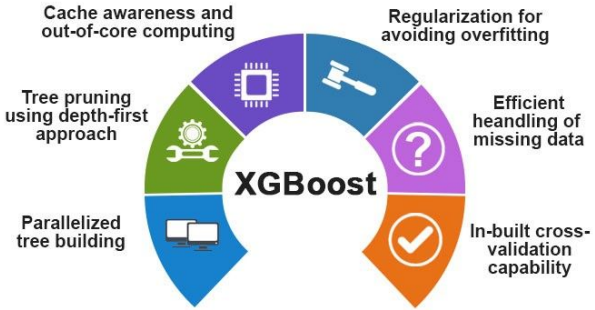
Model Development

Source: Quantdare

Most used boosting models

XGBoost

https://github.com/dmlc/xgboost



LightGBM

https://github.com/microsoft/LightGBM

Fast

Microsoft

CatBoost

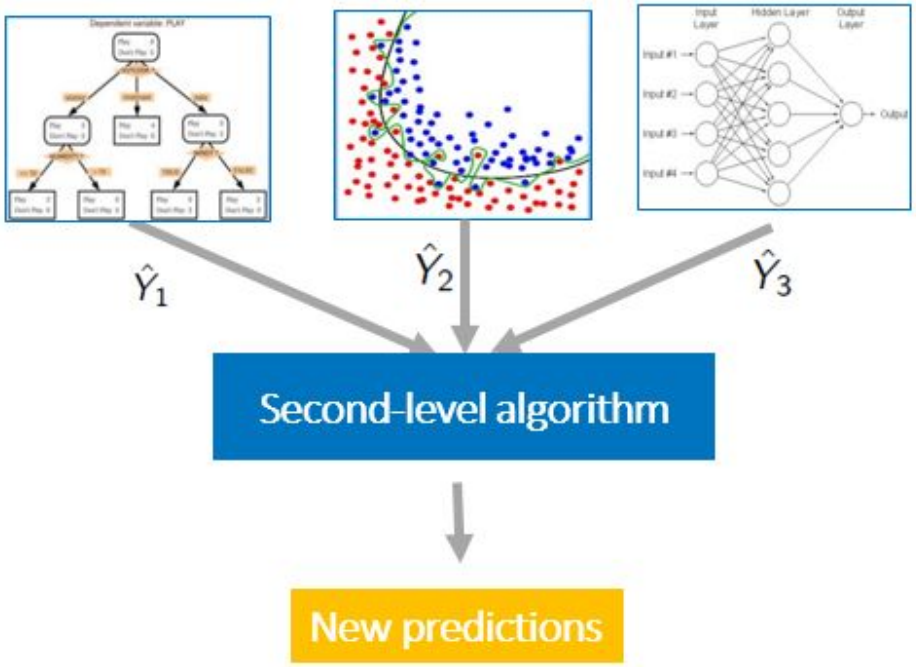
https://catboost.ai/

Categorical features & Overfit optimization

Yandex

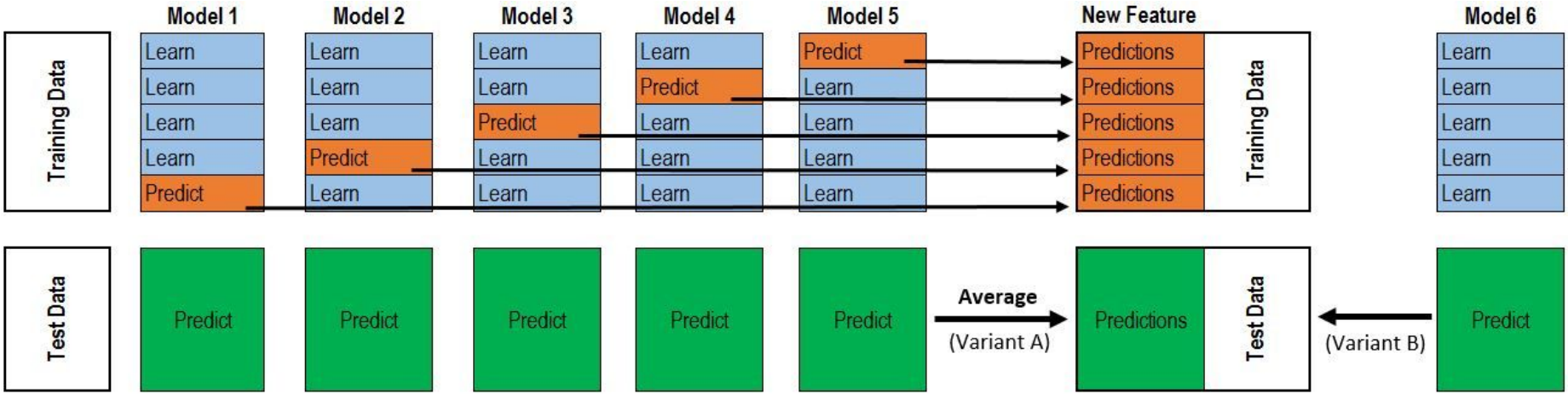
Stacking

Model Development



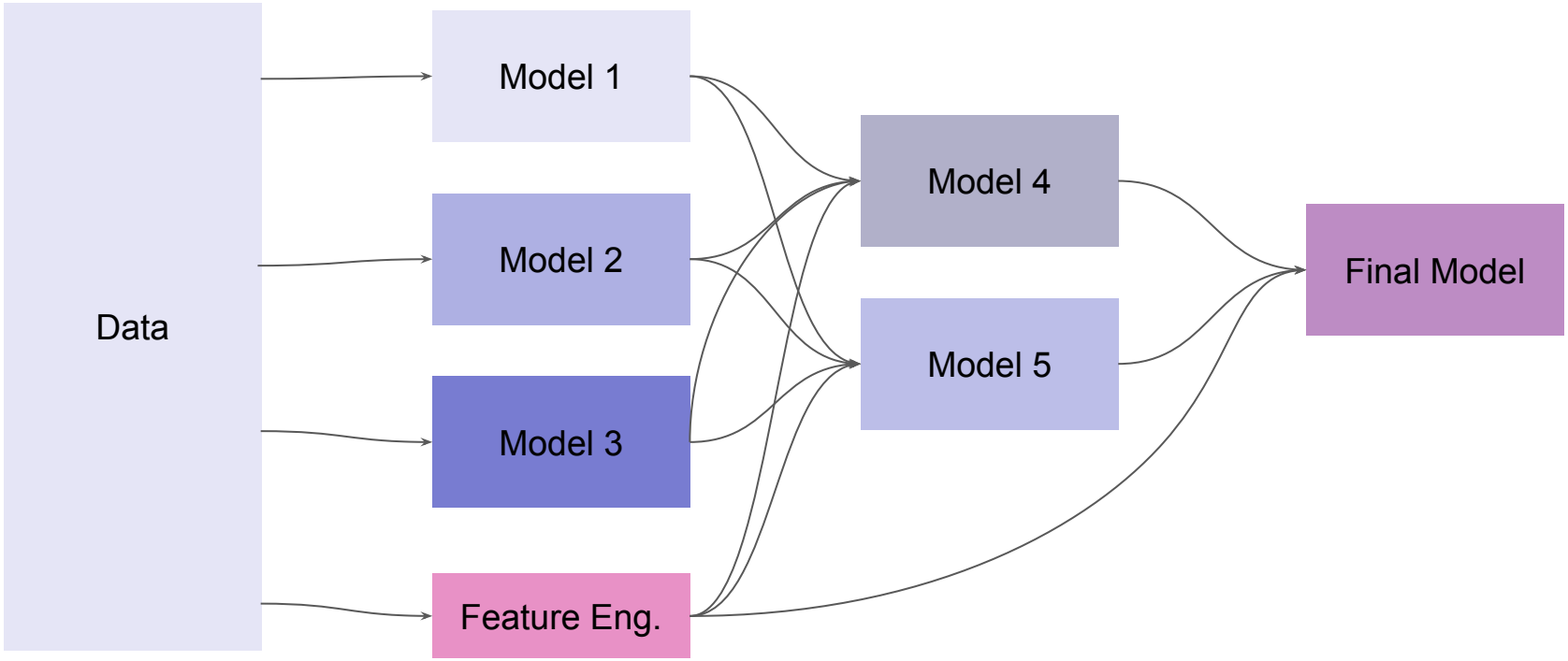
Stacking

Model Development



Stacking

Model Development

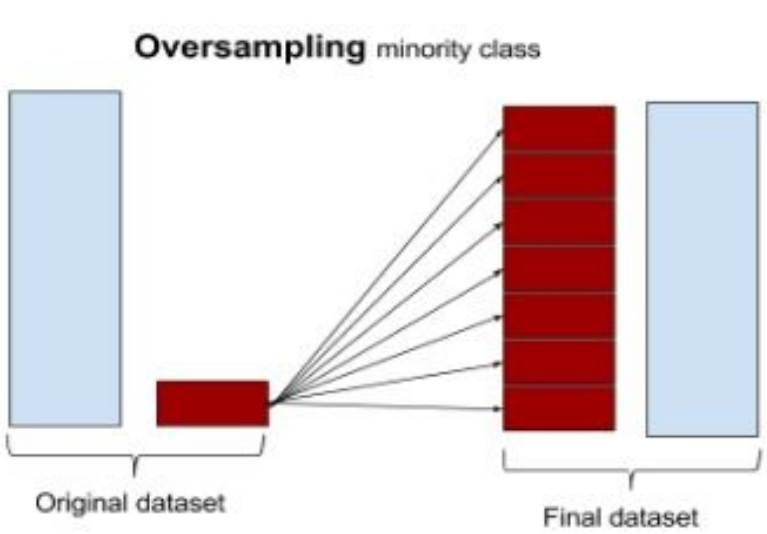


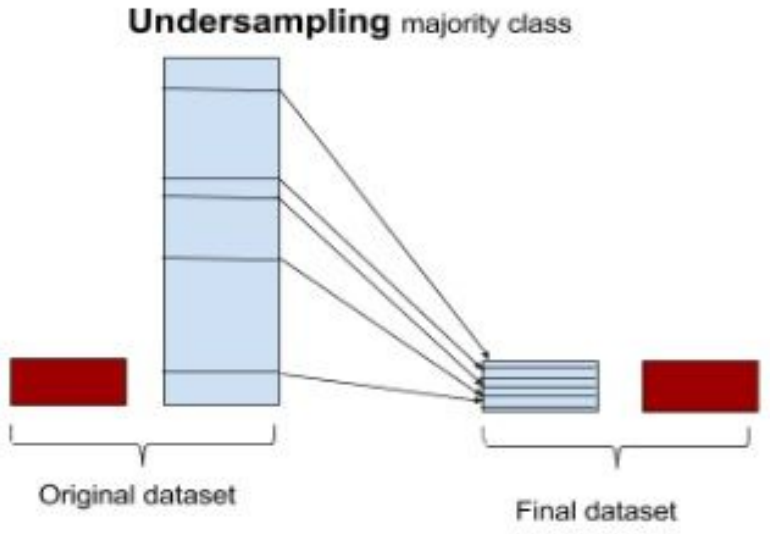
Strategies

Imbalanced data

Solutions

Over and Under Sampling



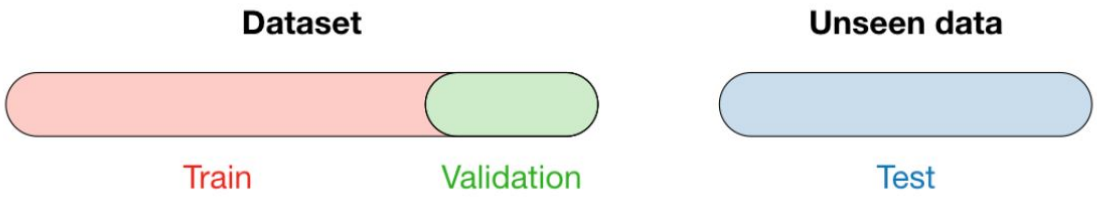


Model selection

Vocabulary — When selecting a model, we distinguish 3 different parts of the data that we have as follows:

Training set	Validation set	Testing set
• Model is trained • Usually 80% of the dataset	• Model is assessed • Usually 20% of the dataset • Also called hold-out or development set	• Model gives predictions • Unseen data

Once the model has been chosen, it is trained on the entire dataset and tested on the unseen test set. These are represented in the figure below:



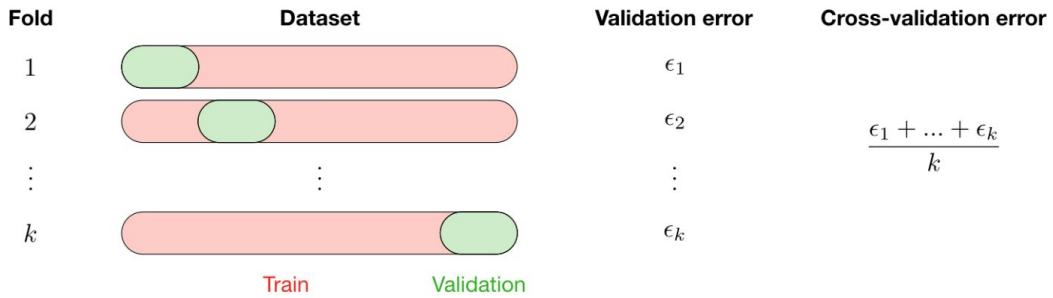
Cross-validation

Solutions

Cross-validation — Cross-validation, also noted CV, is a method that is used to select a model that does not rely too much on the initial training set. The different types are summed up in the table below:

k-fold	Leave-p-out
• Training on $k-1$ folds and assessment on the remaining one • Generally $k=5$ or 10	• Training on $n-p$ observations and assessment on the p remaining ones • Case $p=1$ is called leave-one-out

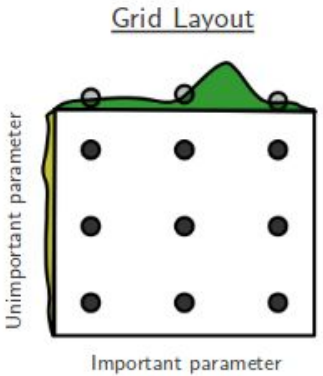
The most commonly used method is called k-fold cross-validation and splits the training data into k folds to validate the model on one fold while training the model on the $k-1$ other folds, all of this k times. The error is then averaged over the k folds and is named cross-validation error.



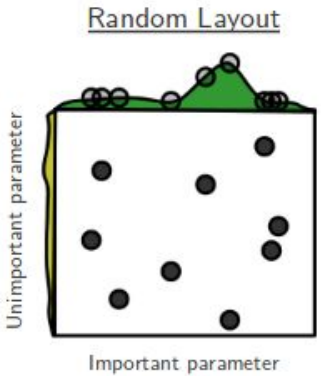
Hyper Parameter Tuning

Solutions

Grid Search



Random Search



Hyperopt

https://pypi.org/project/hyperopt/

https://blog.dominodatalab.com/hyperopt-bayesian-hyperparameter-optimization/

"easy-to-use implementation of a Bayesian hyperparameter optimization algorithm"

HYPEROPT

Optuna

https://arxiv.org/pdf/1907.10902.pdf

Framework	API Style	Pruning	Lightweight	Distributed	Dashboard	OSS
SMAC [3]	define-and-run	✗	✓	✗	✗	✓
GPyOpt	define-and-run	✗	✓	✗	✗	✓
Spearmint [2]	define-and-run	✗	✓	✓	✗	✓
Hyperopt [1]	define-and-run	✗	✓	✓	✗	✓
Autotune [4]	define-and-run	✓	✗	✓	✓	✗
Vizier [5]	define-and-run	✓	✗	✓	✓	✗
Katib	define-and-run	✓	✗	✓	✓	✓
Tune [7]	define-and-run	✓	✗	✓	✓	✓
Optuna (this work)	define-by-run	✓	✓	✓	✓	✓

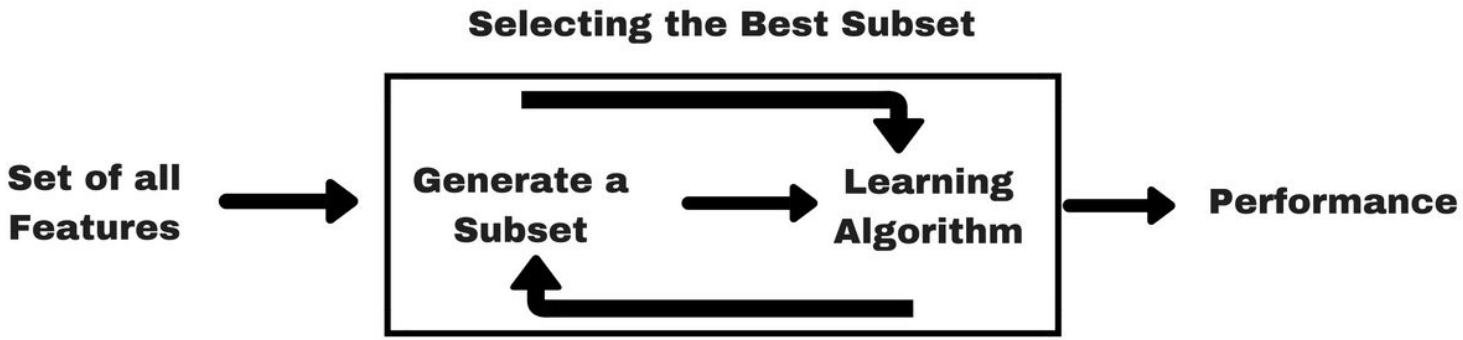
Feature Selection

Solutions

Why?

Goal: Highest performance with the smallest complexity

Reducing complexity: | Deploying | Monitoring | Stability

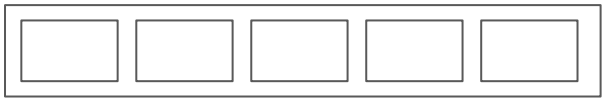


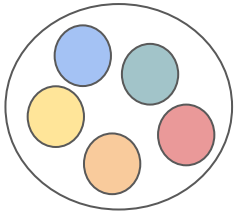
Forward and backward

Feature Selection

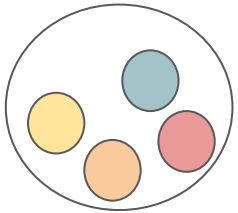
Forward

Start: model with no features

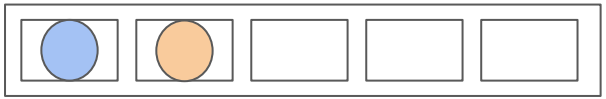


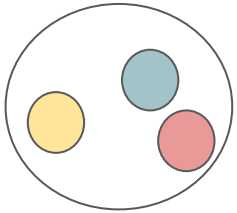


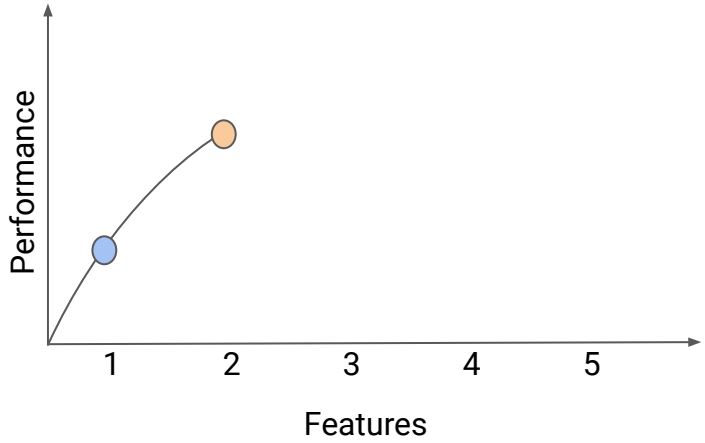
Add the most important feature



Keep adding the most important feature until stopping rule or running out of features





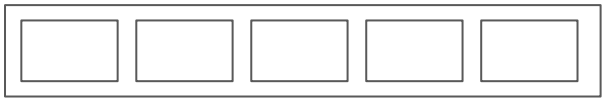


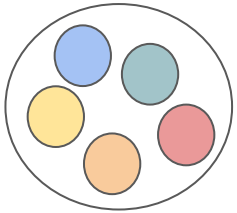
Forward and backward

Feature Selection

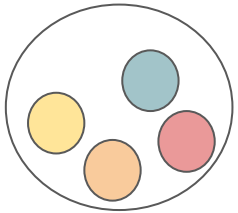
Forward

Start: model with no features

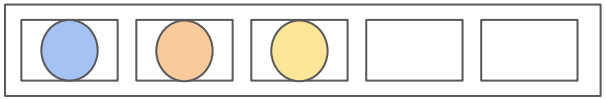


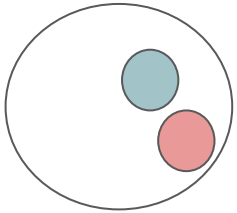


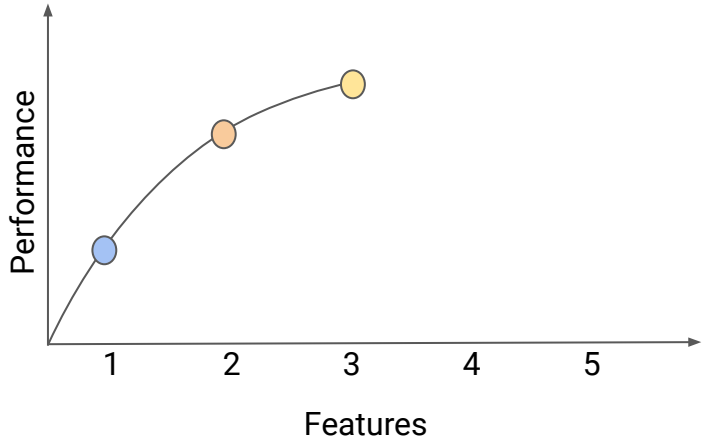
Add the most important feature



Keep adding the most important feature until stopping rule or running out of features





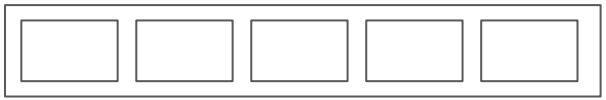


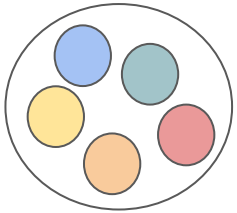
Forward and backward

Feature Selection

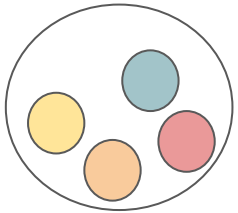
Forward

Start: model with no features

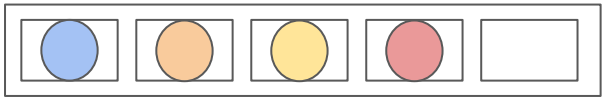


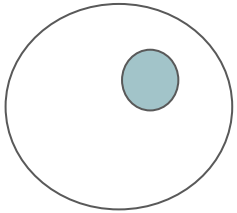


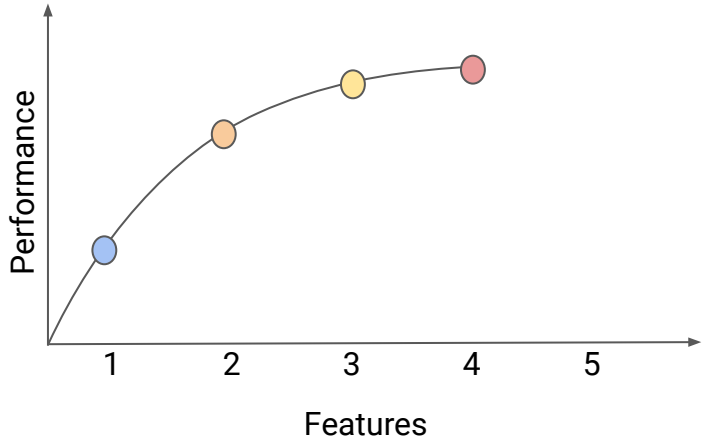
Add the most important feature



Keep adding the most important feature until stopping rule or running out of features





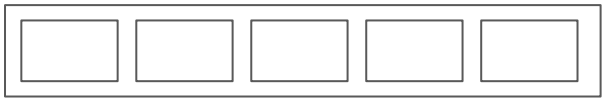


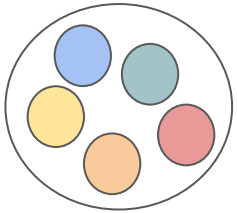
Forward and backward

Feature Selection

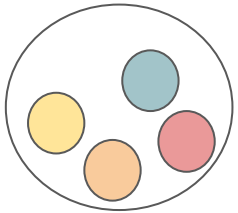
Forward

Start: model with no features

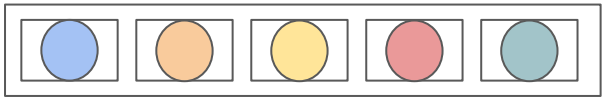


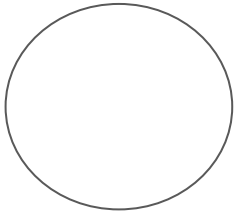


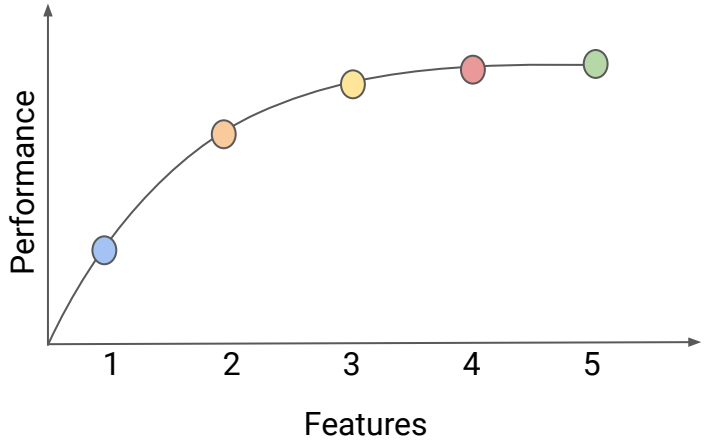
Add the most important feature



Keep adding the most important feature until stopping rule or running out of features





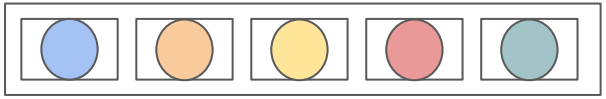


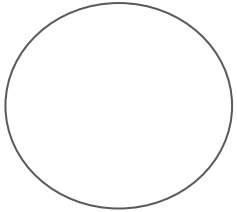
Forward and backward

Feature Selection

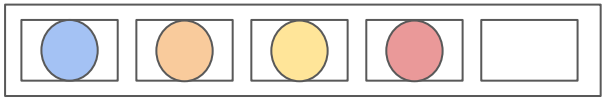
Backward

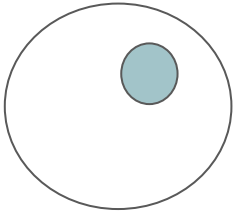
Start: model with all features



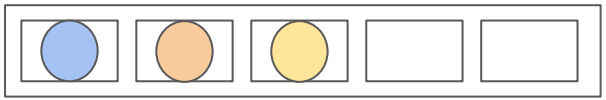


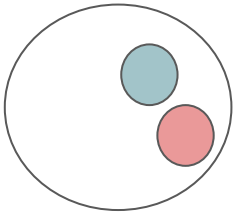
Remove the least important feature

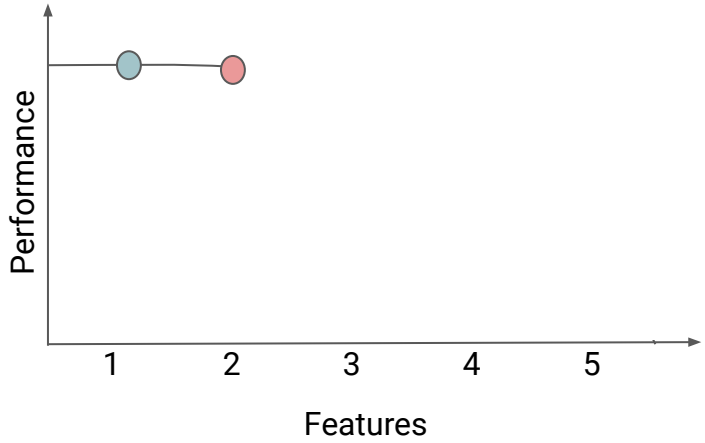




Keep removing the least important feature until stopping rule or running out of features





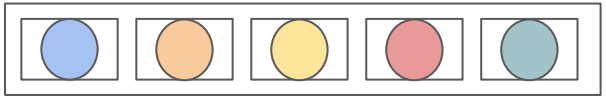


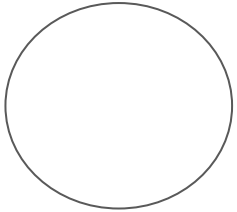
Forward and backward

Feature Selection

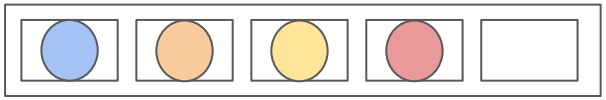
Backward

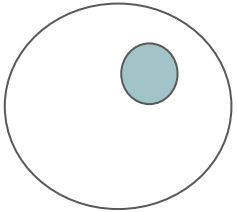
Start: model with all features



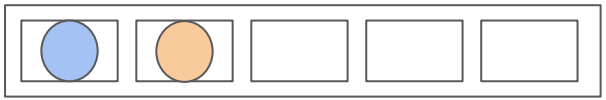


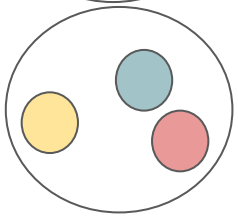
Remove the least important feature

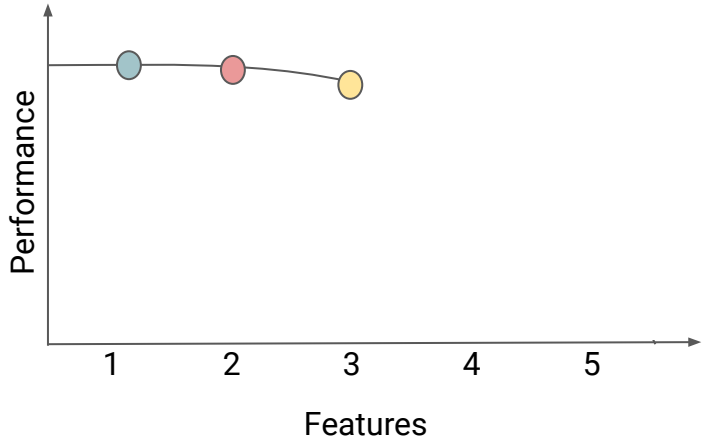




Keep removing the least important feature until stopping rule or running out of features





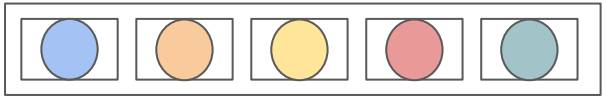


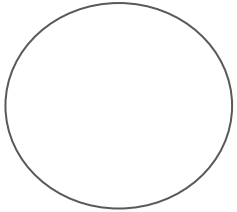
Forward and backward

Feature Selection

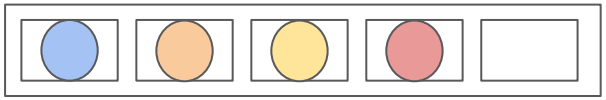
Backward

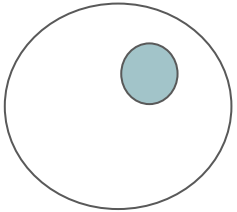
Start: model with all features



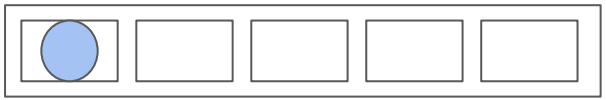


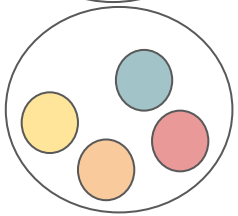
Remove the least important feature

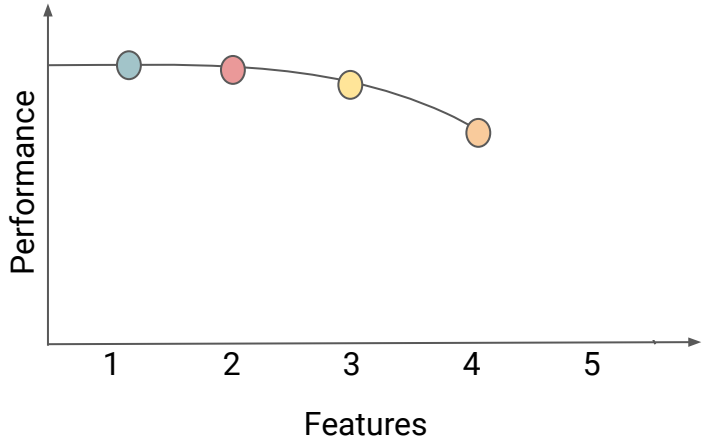




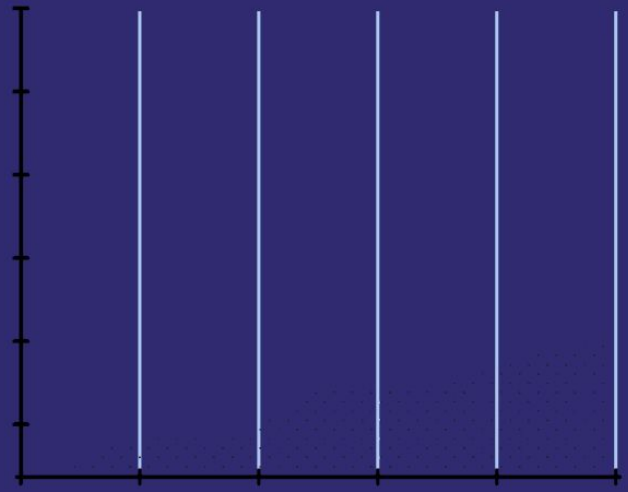
Keep removing the least important feature until stopping rule or running out of features





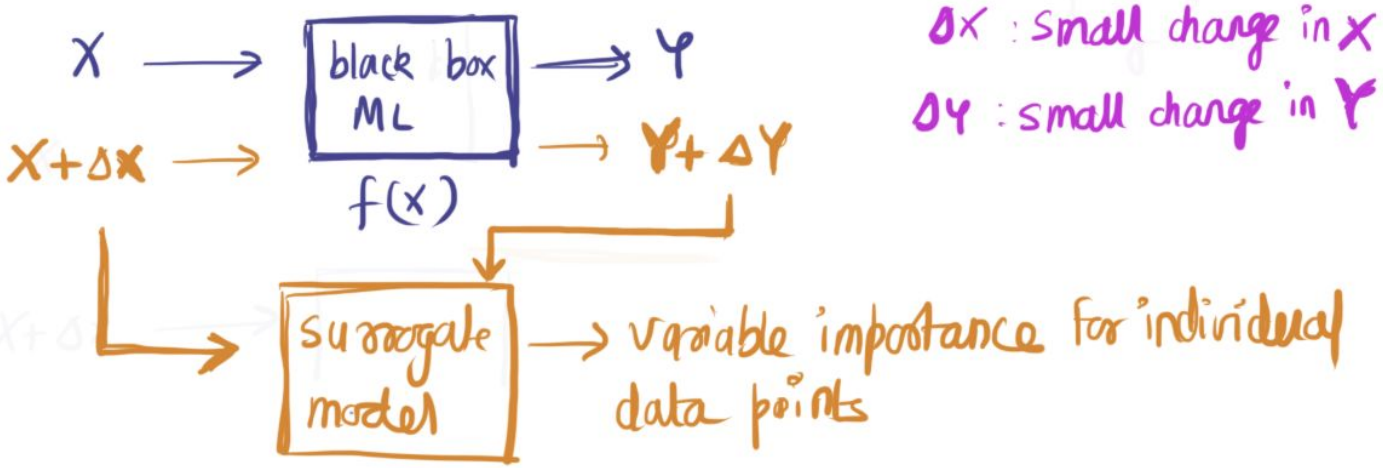


Feature Importance



SHAP

Interpretability

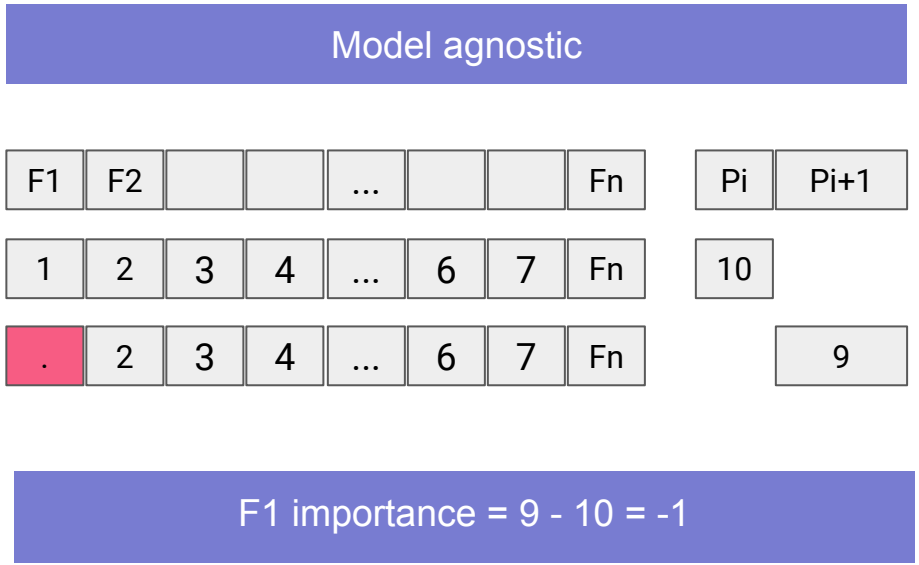


Shapley values calculate the importance of a feature by comparing what a model predicts with and without the feature. However, since the order in which a model sees features can affect its predictions, this is done in every possible order, so that the features are fairly compared.

SHAP

Interpretability

Shapley values calculate the importance of a feature by comparing what a model predicts with and without the feature. However, since the order in which a model sees features can affect its predictions, this is done in every possible order, so that the features are fairly compared.



https://github.com/slundberg/shap

SHAP

Interpretability

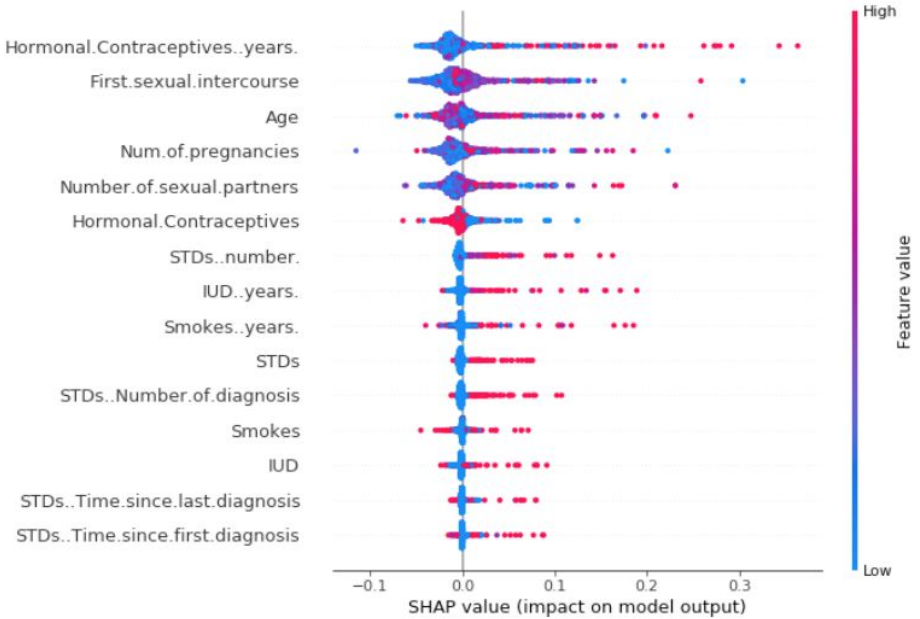
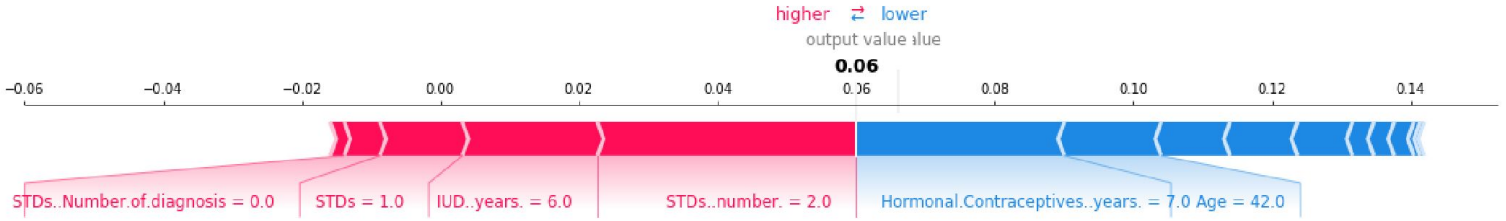
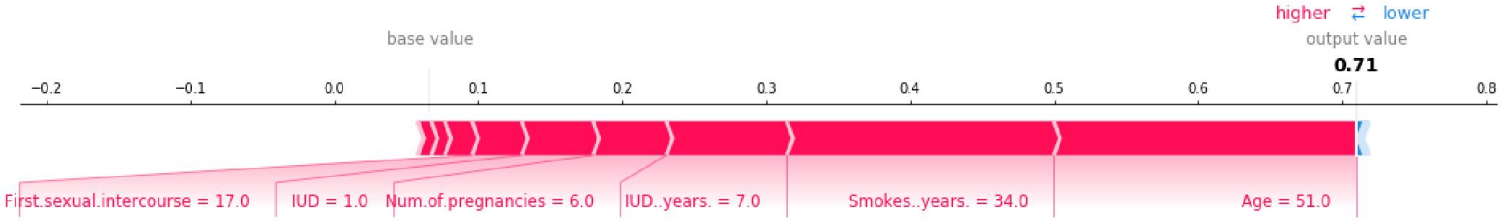


FIGURE 5.52: SHAP summary plot. Low number of years on hormonal contraceptives reduce the predicted cancer risk, a large number of years increases the risk. Your regular reminder: All effects describe the behavior of the model and are not necessarily causal in the real world.

SHAP

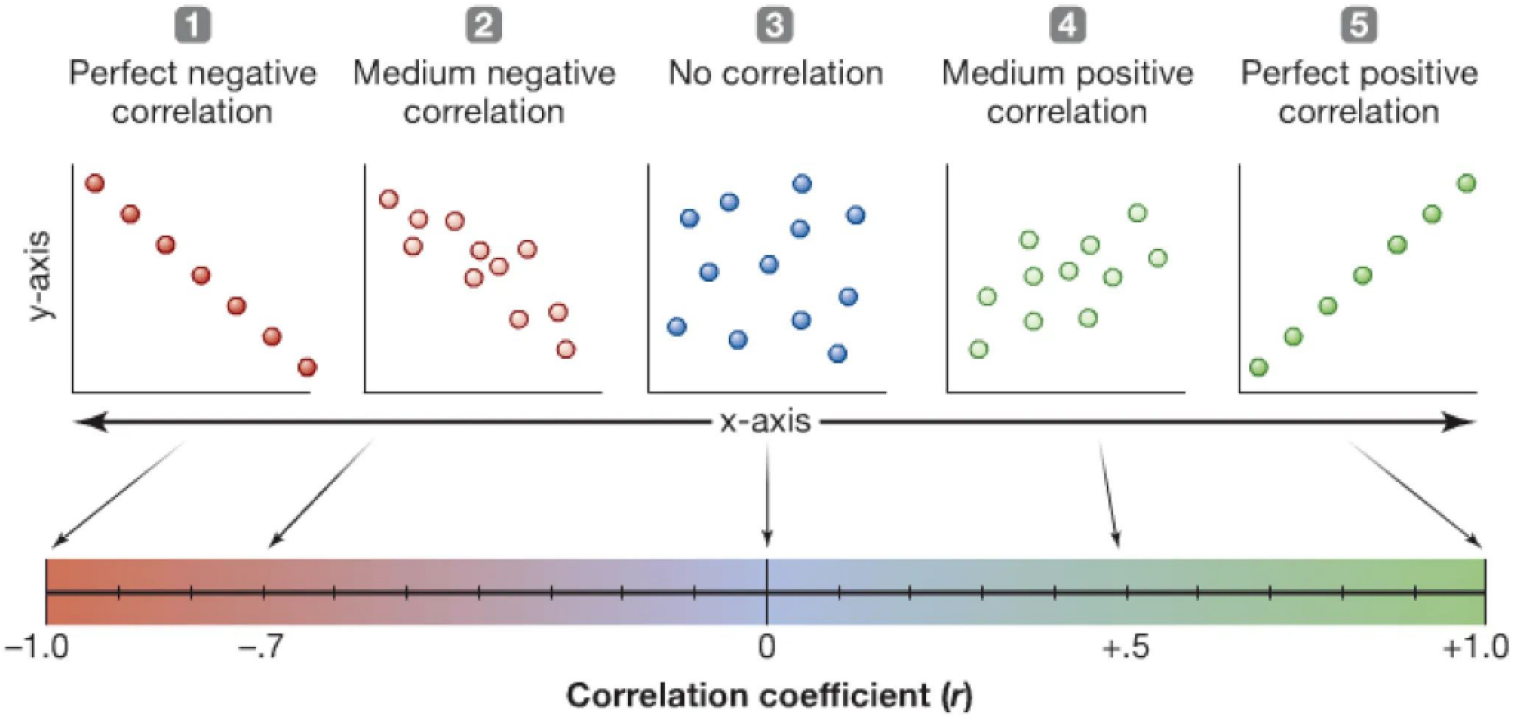
Interpretability





Correlations

Exploratory Data Analysis

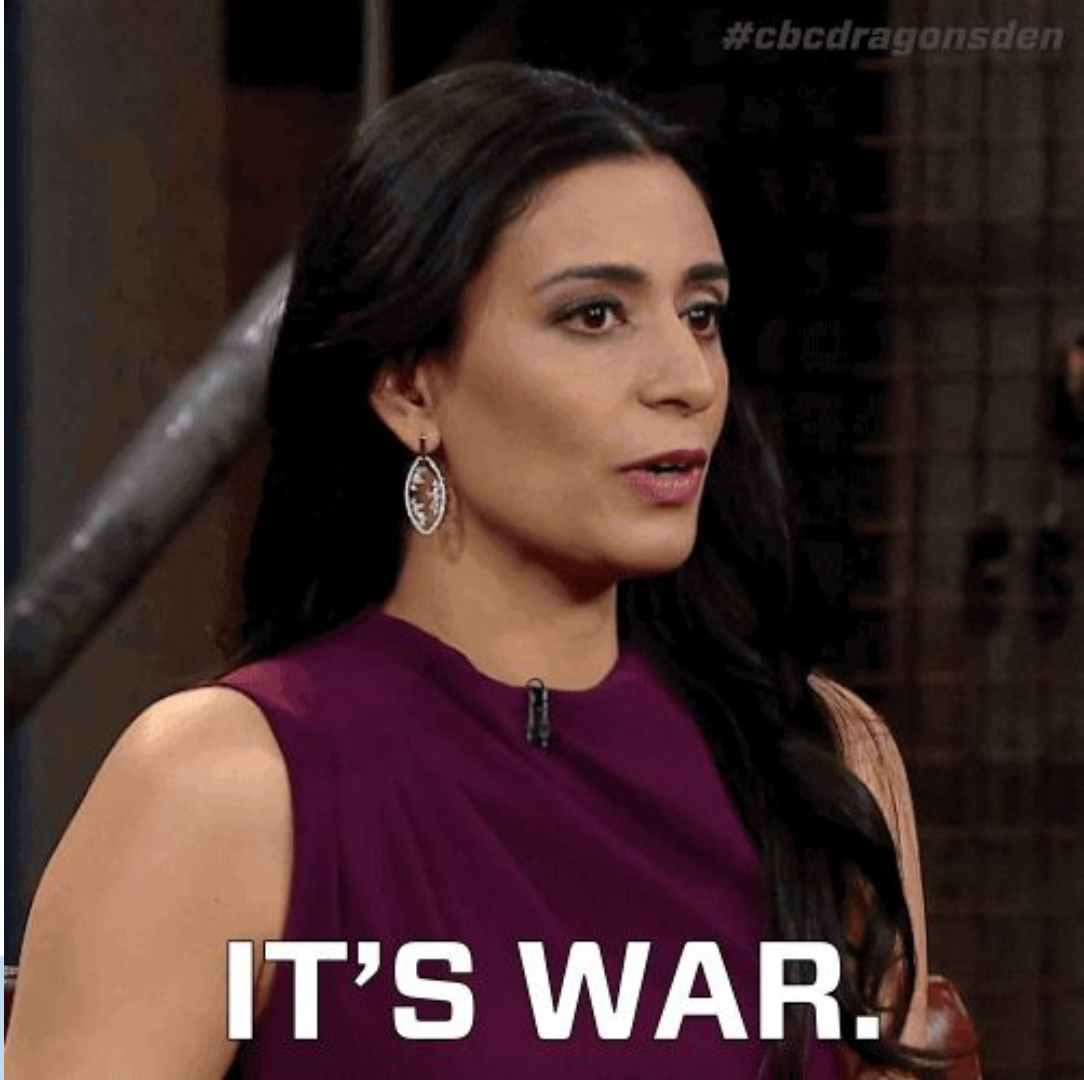


Bonus!

Do you want to practice?!

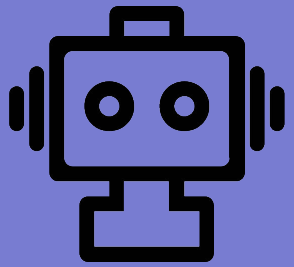
Best strategy: IMHO

kaggle™



AutoML & AutoDL

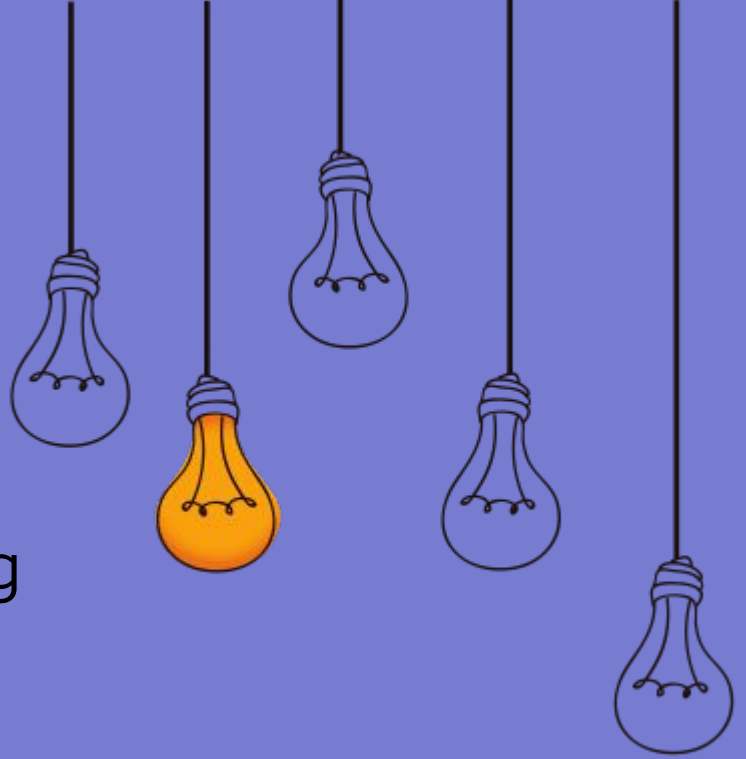
We are already living in the future!



- Auto-sklearn
- MLBox
- Ludwig (Uber)
- Fast.ai
- TPOT
- H2o AutoML
- Auto-Keras
- TransformgrifAI

- FeatureTools
- Nni
- Darts
- PocketFlow
- ...

BE INSPIRED



“

Creativity is intelligence having fun.

ALBERT EINSTEIN