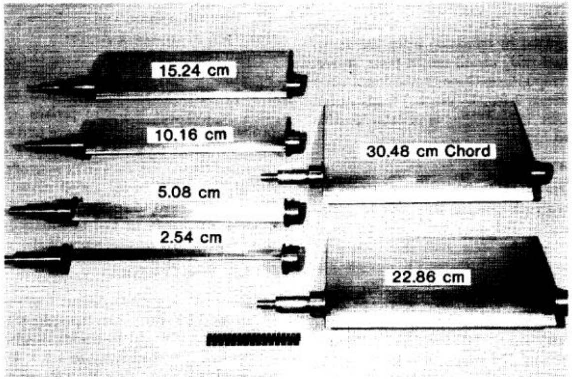
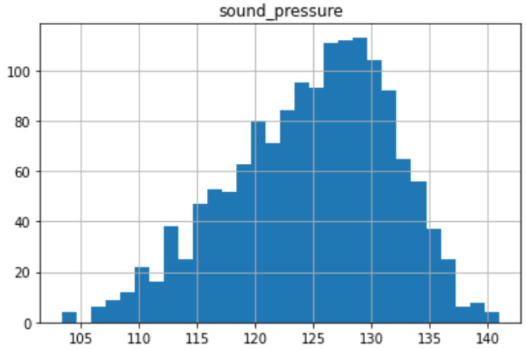
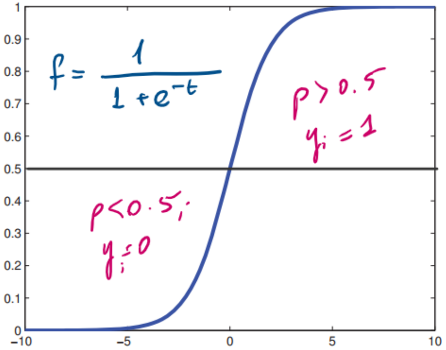
Data Driven Engineering I: Machine Learning for Dynamical Systems

Analysis of Static Datasets I: Classification

Institute of Thermal Turbomachinery
Prof. Dr.-Ing. Hans-Jörg Bauer







1 Page summary: Regression Problem

Karlsruhe Institute of Technology

* Regression := estimate noise from airfoils – NASA experiments
* Exploring data ⇨ histograms, corr. matrix
* Data Preparation ⇨ Test + Training ⟹ Cross Validation 8
* Linear Regression ⇨ how it works & why need regularization
* Learning Curves ⇒ how to interpret them.
* Support Vector Machines ⇨ how it works & data scaling

Today’s Agenda

Basic Steps to Follow:

0.) Understand the business/task.
1.) Understand the data.
2.) Explore & prepare the data.
3.) Shortlist candidate models.
4.) Training the model
5.) Evaluate the model predictions.
6.) "Serve" the model!

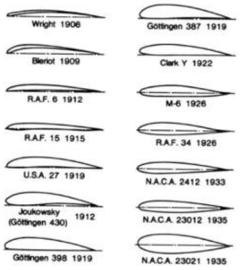
"Classification"

#0 Understanding the task

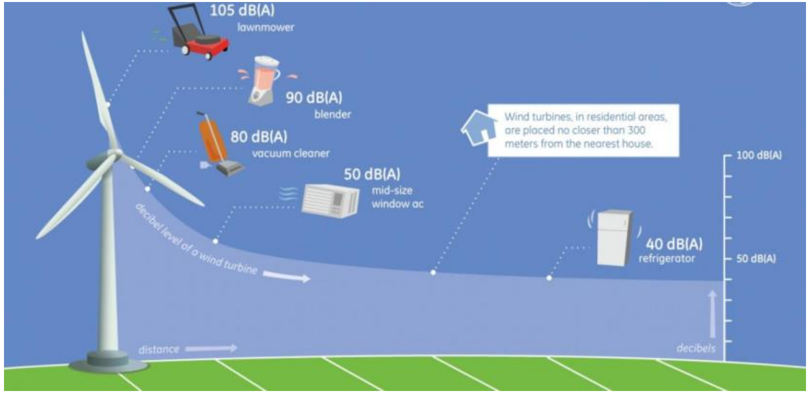
- **Problem**: NACA 0012 Airfoil Noise Prediction based on Wind Tunnel Testing

- **Noise** generated by an aircraft is an **economic** (efficiency) and **enviromental** issue.

- One component of the noise the **self-noise of the airfoil**: interaction of the airfoil with its own boundary layer



1917, the NACA Technical Report No. 18 titled “Aerofoils and Aerofoil Structural Combinations,” was released.



#0 Understanding the task

- ❑ Engineering: semi-emprical models (Brooks)
- ❑ Five self-noise mechanisms due to specific boundary-layer phenomena have been identified
- ❑ The database is from seven NACA0012 airfoil blade sections of different sizes tested at wind tunnel speeds up to Mach 0.21 and at angles of attack from 0°to 25.2°.
  - ✓ Freq. of noise
  - ✓ Angle of attack
  - ✓ Free stream velocity
  - ✓ Geometry of the airfoil

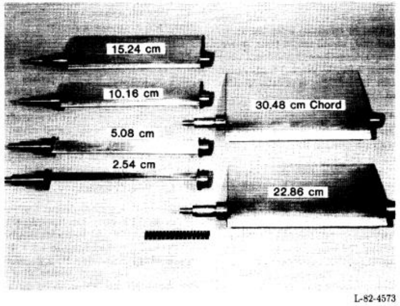
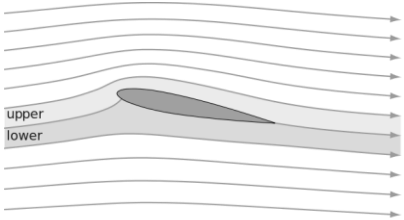
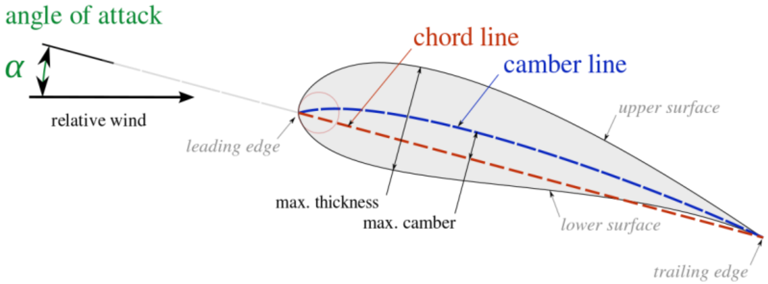


Figure 2. Two-dimensional NACA 0012 airfoil blade models.





#1 Understanding the data

- ❑ Check the data source: understand what the data refers to
- ❑ Objective: understand the characteristics of the data
- ❑ Look at the feature columns:
  - ❑ Any missing values?
  - ❑ Any features with NaN values?
  - ❑ Uniqueness of the dataset? (“cardinality”)

=> Colab

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1503 entries, 0 to 1502
Data columns (total 6 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   frequency               1503 non-null   int64
 1   angle_attack            1503 non-null   float64
 2   chord_length            1503 non-null   float64
 3   Free-stream_velocity    1503 non-null   float64
 4   displacement_thickness  1503 non-null   float64
 5   sound_pressure          1503 non-null   float64
dtypes: float64(5), int64(1)
memory usage: 70.6 KB
```

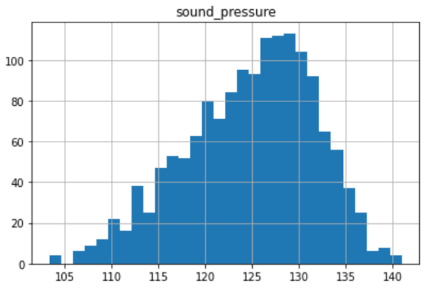
```
data.head(5)
```

	frequency	angle_attack	chord_length	Free-stream_velocity	displacement_thickness
0	800	0.0	0.3048	71.3	0.002663
1	1000	0.0	0.3048	71.3	0.002663
2	1250	0.0	0.3048	71.3	0.002663
3	1600	0.0	0.3048	71.3	0.002663
4	2000	0.0	0.3048	71.3	0.002663

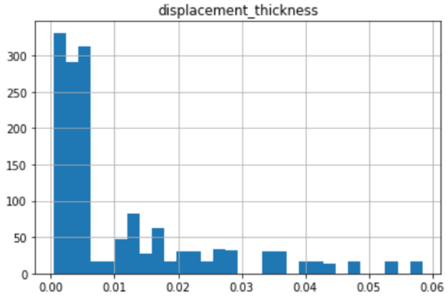
#2 Exploring the data

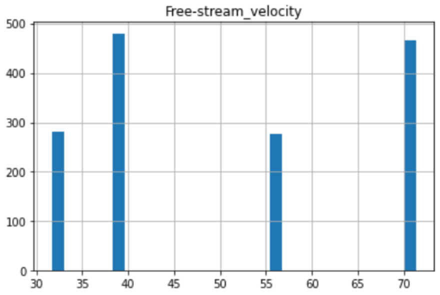
- ❑ **Objective**: generate a data quality report

- ❑ Using standard statistical measures of central tendency and variation
  - ❑ Tabular data and visual plots
  - ❑ mean, mode, and median
  - ❑ standard deviation and percentiles
  - ❑ Bars, histograms, box and violin plots



- ✓ Missing values,
- ✓ Irregular cardinality problems,
  - 1 or comparably small
- ✓ Outliers
  - invalid outliers and valid outliers





#2 Exporing the data: Correlation Matrix

- Shows the correlation between each pair of features

$$Cov(a,b) = \frac{1}{n-1}\sum_{i=1}^{n}\left[(a_i-\bar{a})\times(b_i-\bar{b})\right]$$

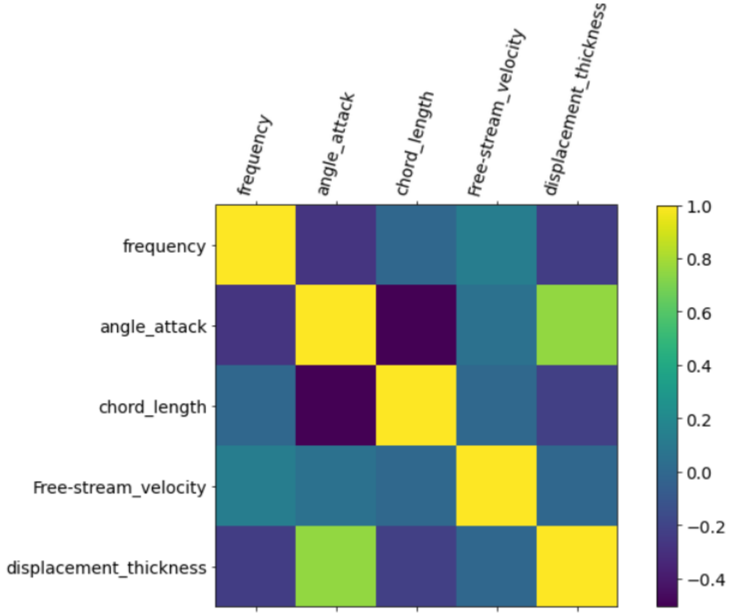
Features — instance — mean — mean

- Normalized form of "covariance"

$$Corr(a,b) = \frac{Cov(a,b)}{SD(a)\times SD(b)}$$

* Normalized
* Dimensionless
Easy to interpret

- Ranges between −1 and +1

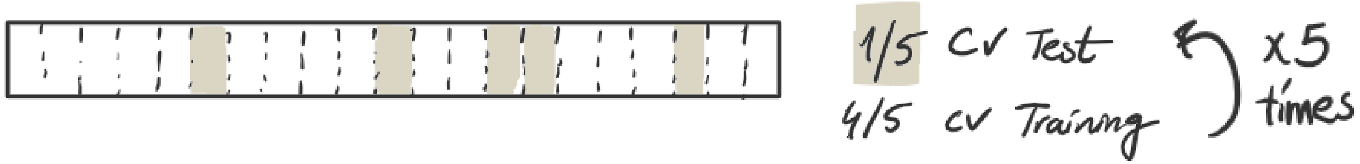


#2 Preparing the Data

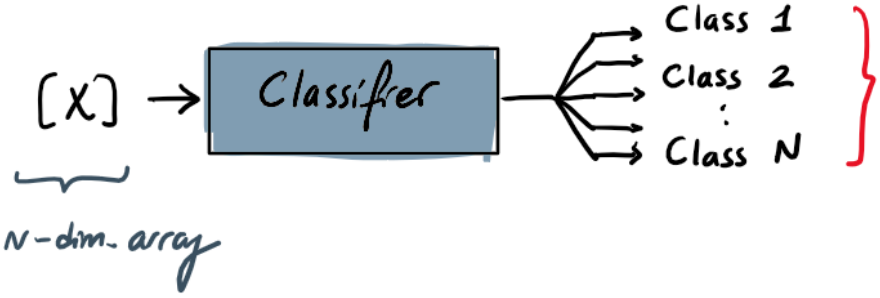
- Classification >> supervised >> **training & test split**

- Reducing overfitting via **cross-validation**: take **random portions** of the data to build a model

- **k-fold** method: k = 5; (typically 10)



#3 Candidate Model Selection 1



Data space is divided into decision regions

* Regression := $y(x,w) = (w \cdot x + b)$ } Cont'd "y"

learnt

* Classification;
  - $y \rightarrow \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ or $y := \{0, 0, 0, 1, 0\}$
  - $y = \phi(wx + b)$ "Activation function"

Probabilistic Discriminative Models : Logistic Regression

$\phi \rightarrow$ Sigmoid function;

- $p(C_1|x) = y(x) = \sigma(wx+b)$

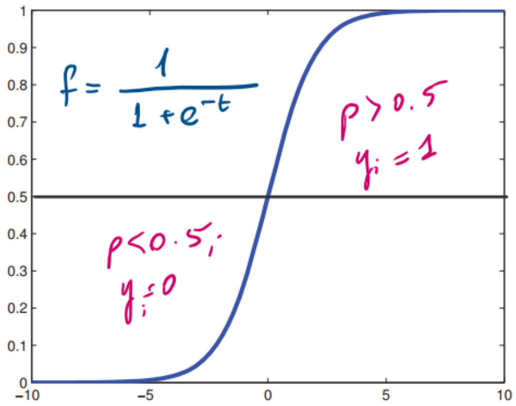
binary $\Rightarrow$ $p(C_2|x) = 1 - p(C_1|x)$

$\sigma(x) = \dfrac{1}{1+\exp(-x)}$ } inverse; $x = \ln\left(\dfrac{\sigma}{1-\sigma}\right)$ } Logit Function

$x = \dfrac{p(C_1|x)}{p(C_2|x)}$

! M-dimensional X $\Rightarrow$ M parameters

? How to determine M parameters ...



Probabilistic Discriminative Models : Logistic Regression

Maximum likelihood

* Likelihood $\Rightarrow$ $p(y_t|w) = \prod_{n=1}^{N} \left\{ [p(C_1|x_n)]^{y_{t_n}} [1 - p(C_1|x_n)]^{1-y_{t_n}} \right\}$

* Error $\Rightarrow$ $E_p(w) = -\ln(p(y_t|w))$

$$= -\sum_n \left\{ y_{t_n} \ln(\underbrace{\sigma(wx_n+b)}_{p(y_n)}) + (1-y_t)\ln(1-\underbrace{\sigma(wx_n+b)}_{p(y_n)}) \right\}$$

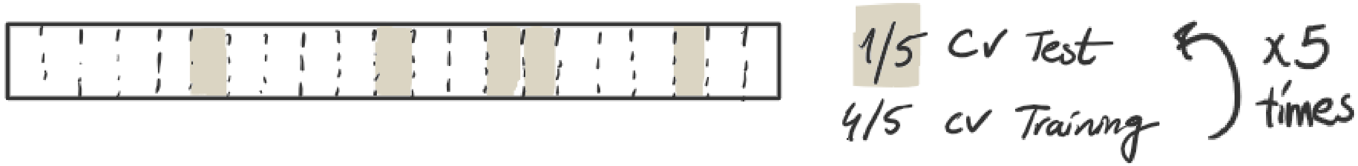
Scikit $\Rightarrow$ $\min_{w,c} \frac{1-\rho}{2} w^T w + \rho \|w\|_1 + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1)$

#4 Training the model

- Classification >> supervised >> **training & test split**

- Reducing overfitting via **cross-validation**: take **random portions** of the data to build a model

- **k-fold** method: k = 5; (typically 10)



colab

#5 Evaluation of the Predictions

Log loss for binary classification

* Cross-entropy between true labels & model predictions

* Average loss for Class A & Class B; considering N examples:

$$\underset{[A,B]}{\text{Log Loss}} = -\frac{1}{N}\sum_{i=1}^{N} \overbrace{y_i}^{\text{A Class}} \log(p(y_i)) + \overbrace{(1-y_i)}^{\text{B Class}} \log(1-p(y_i))$$

y_i → label; $p(y_i)$ → prob. predicted

Eg.

$$\underset{(\text{lin.})}{\text{cost}} = \begin{cases} -\log(p) & ; \; y=1 \\ -\log(1-p) & ; \; y=0 \end{cases}$$

$y_i = 1 \Rightarrow p_i = 10^{-4} \Rightarrow \left.\begin{matrix} -\log(p_i) = 4 \\ -\log(1-p_i) \cong 10^{-7} \end{matrix}\right\}$ large error !

$\searrow p_i = 0.95 \Rightarrow -\log(p_i) = 0.02 \}$ small error !

* Cost function ⇒ Convex ⇒ Global minimum exists !

--→ optimization algorithm needed !

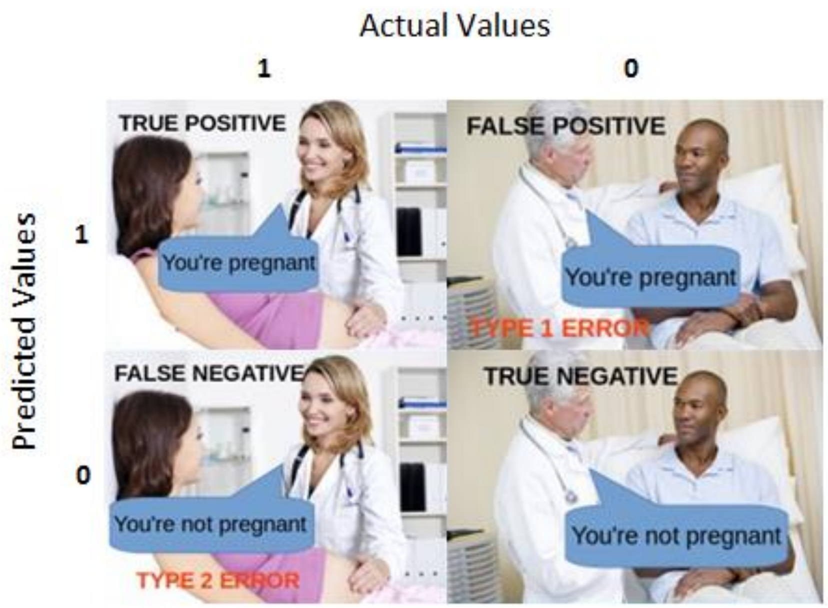
#5 Evaluation of the Predictions

Karlsruhe Institute of Technology

Confusion Matrix

- ☑ Convenient way to describe the performance.
- ☑ Basis for different measures
- ☑ Good for balanced classes

! Imbalanced dataset $\begin{pmatrix} C_1 \rightarrow 95\% \\ C_2 \rightarrow 5\% \end{pmatrix} \Rightarrow$ overpredict the performance



#5 Evaluation of the Predictions

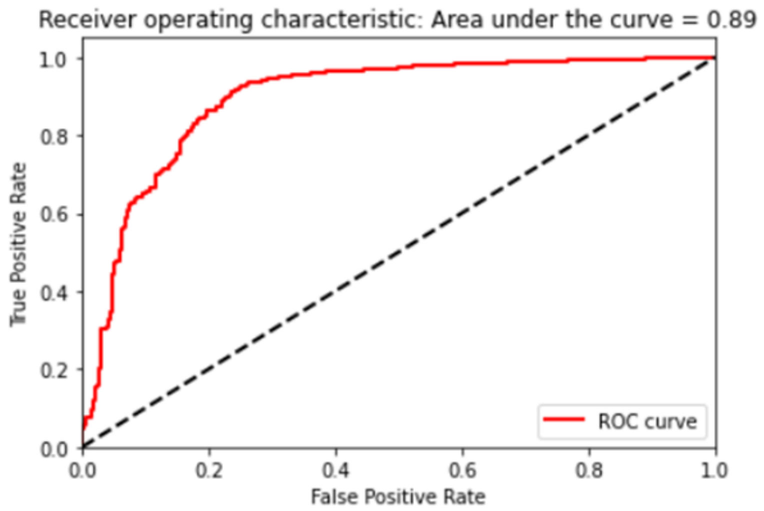
ROC Curve

Receiver Operating Characteristic

(i) Confusion matrix ⇒ based on a score threshold of 50%

(ii) 0% – 100% Threshold ⇒ TP & TN values would change.

(iii) ROC ⇒ Plot for every threshold value



! Closer to top-left, better it is

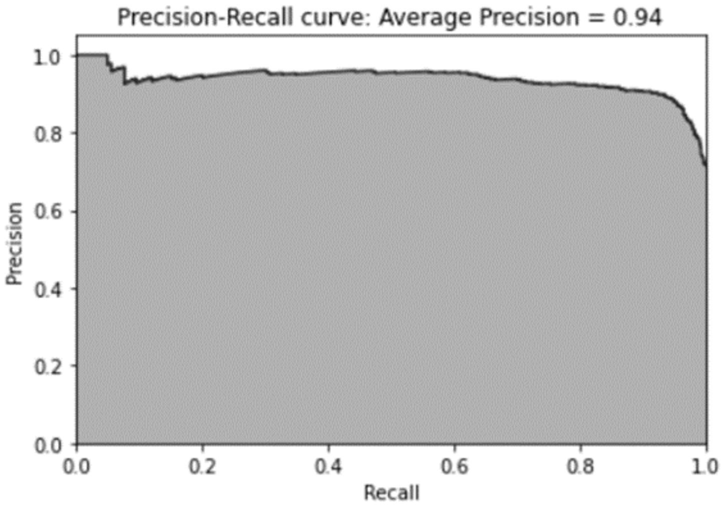
#5 Evaluation of the Predictions

Karlsruhe Institute of Technology

Precision - Recall Curve ⇨ for imbalanced data

$$\text{Precision} := \frac{\text{True Positive}}{\text{TP} + \text{False Positive}} \Rightarrow \frac{\text{It is positive}}{\text{“It is positive”}}$$

$$\text{Recall} := \frac{\text{True Positive}}{\text{TP} + \text{False Negative}} \Rightarrow \frac{\#\text{Correct Predict.}}{\#\text{True Cases}}$$



Precision: how often, when a model makes a positive prediction the prediction is correct.

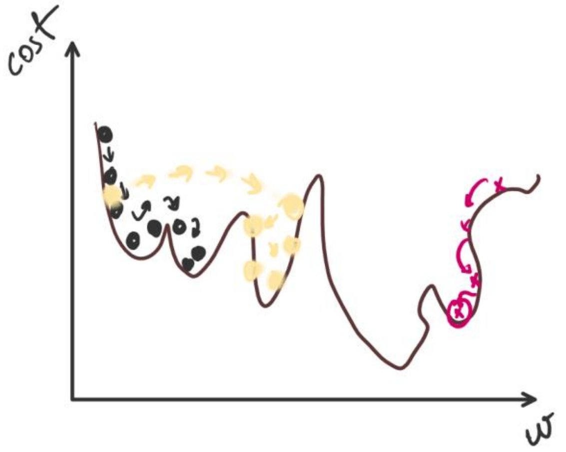
Recall: how much of the True Cases are correctly predicted by the model.

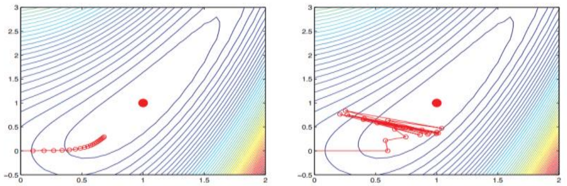
colab

#3 Candidate Model Selection 2

Gradient Decent ⇒ "scikit learn"

- GD ⇒ optimizer for a given loss function
- Measures the local gradient of the error function
  - ↳ goes in the direction of descending gradient ($\frac{\partial}{\partial w}$)
- Scikit-learn ⇒ interface for multiple models
  - Efficient & have many tuning options
  - Linear models





! Learn the details in neural networks

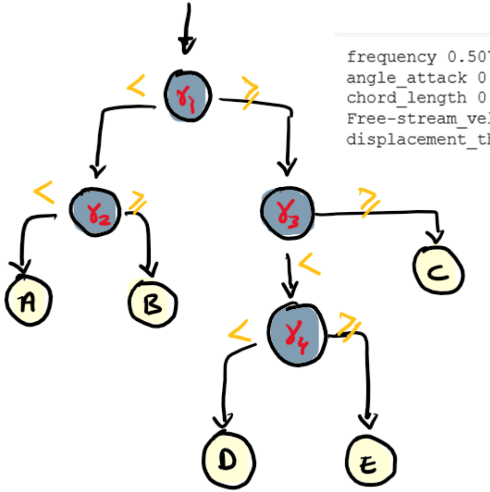
colab

#3 Candidate Model Selection 3

Decision trees ⇒ Random Forests

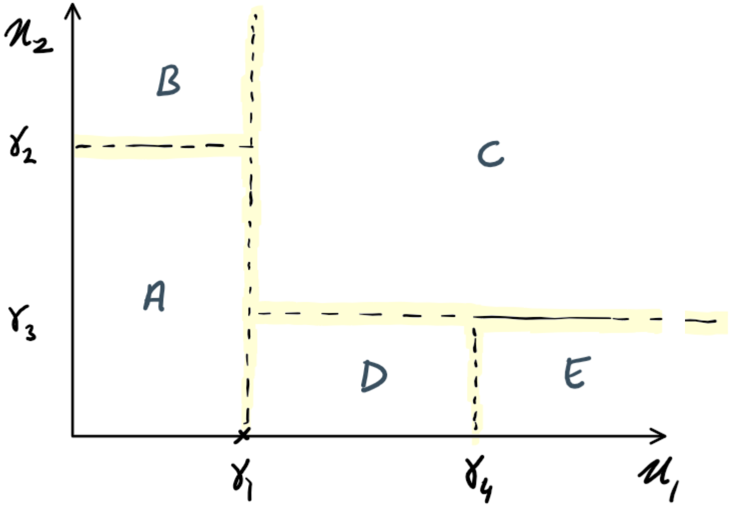
* Build a graph via sequential decisions.

$$[X] = (X_1, X_2)$$




```
frequency 0.5078899615845973
angle_attack 0.10084945043187156
chord_length 0.09809369981835218
Free-stream_velocity 0.05777680855346639
displacement_thickness 0.2353900796117127
```

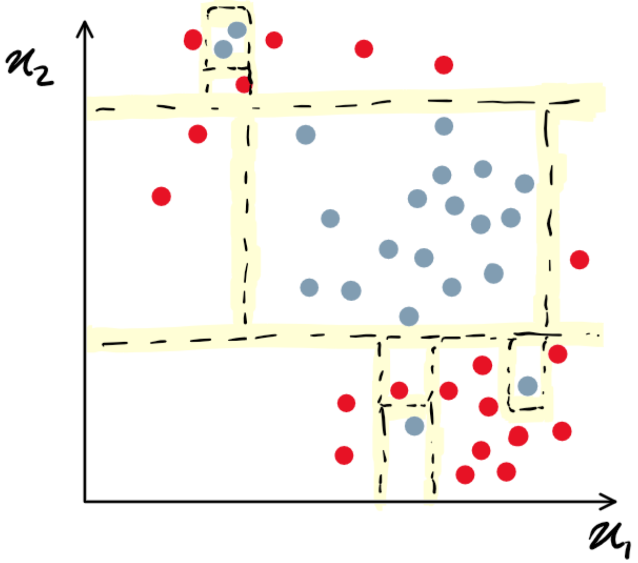
* Data space is divided into regions.



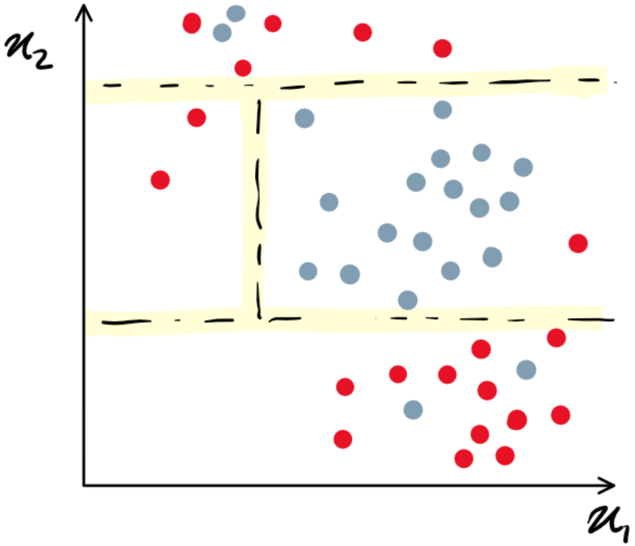
#3 Candidate Model Selection 3

Decision trees ⇒ Random Forests

* Build a graph via sequential decisions.



* Data space is divided into regions.



(i) Select a feature f_i

(ii) Select a value v_i

(iii) Create sub-trees

? when to stop

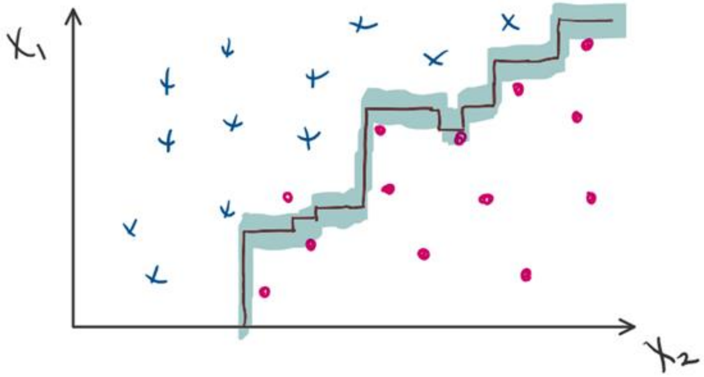
! Data sensitive

#3 Candidate Model Selection 3

Decision trees ⇒ Random Forests

Issues with DT:

* Predictive accuracy of DT ⇒ training data
* Need to restrict how DT grows
  * Max. depth
  * Pruning
* Decision boundaries ⇒ Orthogonal !



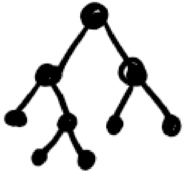
? Feature engineering

? Dim. Reduction methods

#3 Candidate Model Selection 3

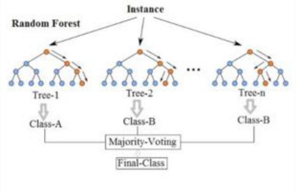
Decision trees ⇒ Random Forests

Issues with DT:

* 
  DT ⇒ Controlled ⇒ 'Simple, ~weak estimator ⇒ Fails to learn complex problem

* Tree ⇒ 'Forest': many weak learners
  - --> wisdom of the crowd
  - --> How to train individuals?

① Take average
② Domain experts
③ Iterative training := boosting



colab

#3 Candidate Model Selection 4

Karlsruhe Institute of Technology

LightGBM := Boosting

LightGBM

"Tree-based Gradient Boosting"

* Combine weak learners ⇒ ensemble learning
* Train predictors sequentially
  - -> each trying to correct its predecessor -> "learn from past mistakes"
    - -> focus on failed cases
* Gradient (G) ⇒ Second order derivatives
  ⇒ Regularization

colab

Additional Notes

Preparing the Data: Bootstrapping

- Bootstrapping approaches are preferred over CV in the case of very small datasets (< 300 instances).
- using slightly different training and test sets each time to evaluate the expected performance
- k is set to values greater than or equal to 200