Validation & Regularization

How to validate models?

Outline

1. Supervised Learning summarized
2. Linear Regression
3. Overfitting!
4. ML 101 validation
5. Regularization
6. Real world supervised learning

About me

Academic

- MSc in Computer Science student (IME-USP)
- Bachelor in Computer Engineering (Poli-USP)
- Bachelor in Economics (FEA-USP)

Work & activities

- Data Scientist at Nubank (2017 - Current)
- Teaching Machine Learning for MBA courses at FIA
- Udacity mentor and project reviewer for data related courses
- Nubank Machine Learning meet-up organizer
- Kaggler (competitions and datasets)
- Twitter and Blog: @lgmoneda and lgmoneda.github.io

How is it going to work?

1. Slides for intuition
2. Code and exercises in the notebook for experiments and hands on
3. Checkpoints after important topics: be honest if the concept isn't clear, I'm going to clarify, use further examples or other analogies.

Supervised Learning summarized

$$\mathbf{X} \xrightarrow{f} \mathrm{y}$$

- Empirical Risk Minimization
- Statistical Learning
- Independently identically distributed (iid)
- We want to predict things nicely, we don't care about what is the f

Checkpoint!

Linear Regression (notebook)

Overfitting

Source: Wikimedia commons

Overfitting

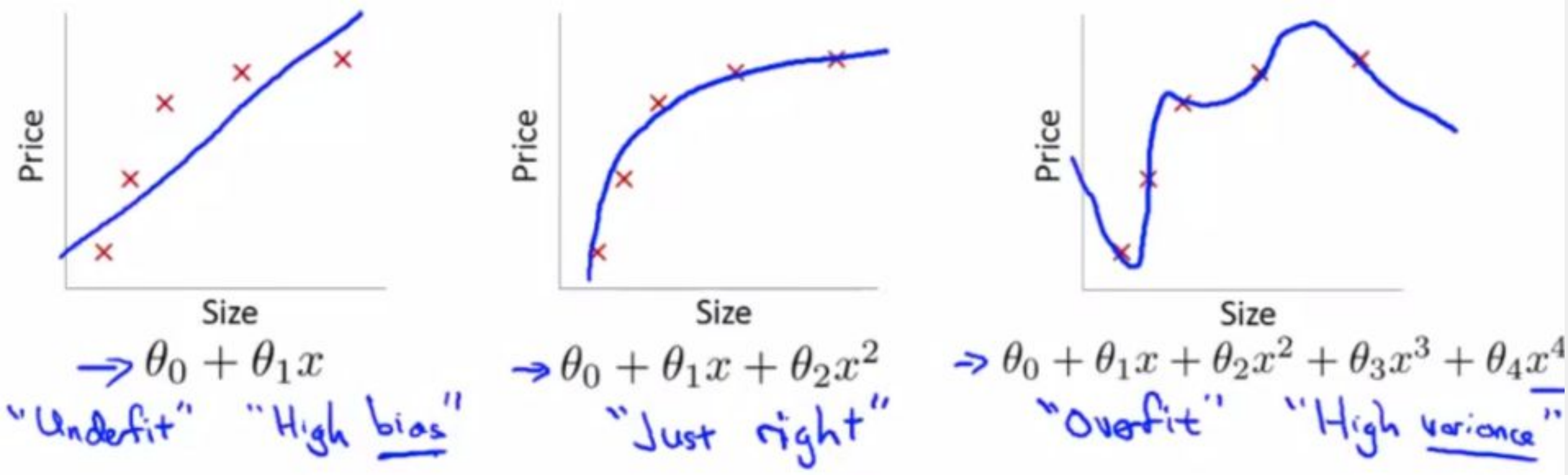
bitolado

adjetivo

1. que se bitolou.
2. figurado (sentido)•figuradamente

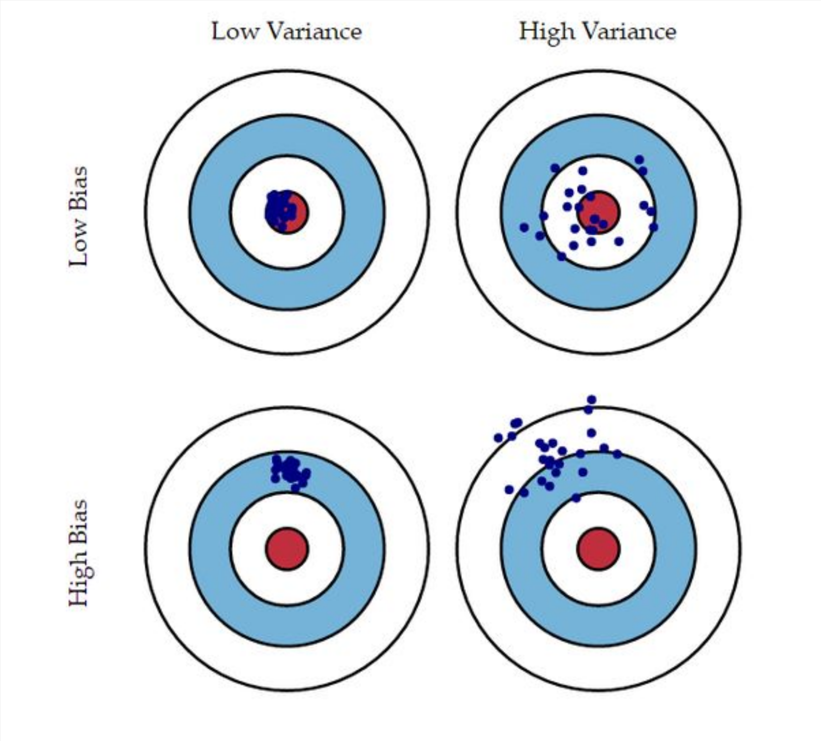
   que tem ideias, opiniões ou conhecimentos estreitos, rígidos, limitados, ultrapassados; quadrado, careta.

Overfitting



Source: Coursera, Machine Learning, Andrew Ng, "Lecture 7.1 — Regularization | The Problem Of Overfitting"

Bias x Variance



Checkpoint!

Vapnik–Chervonenkis (VC) Dimension

To the notebook!

How overfitting, validation and regularization relate?

The relationship

Validation **identifies** overfitting, regularization help us **avoid** it!

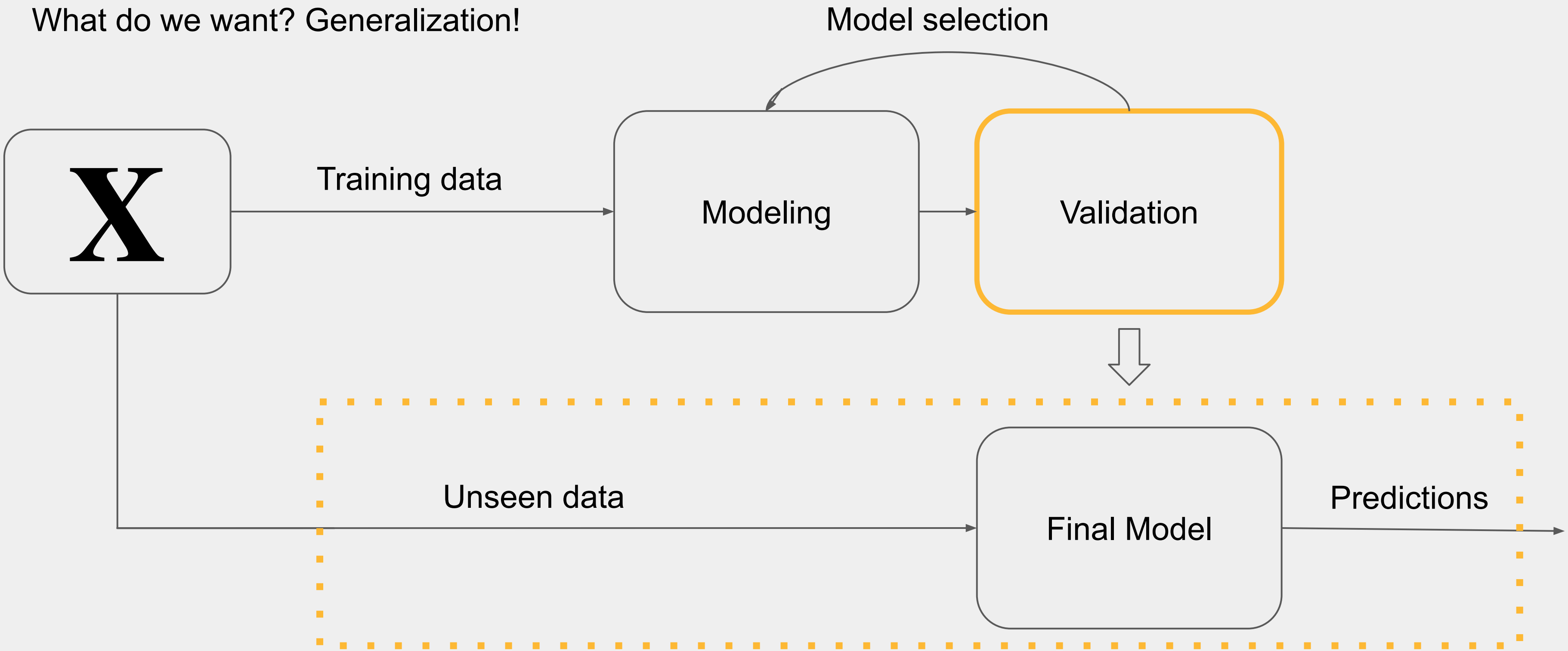
Checkpoint!

Validation

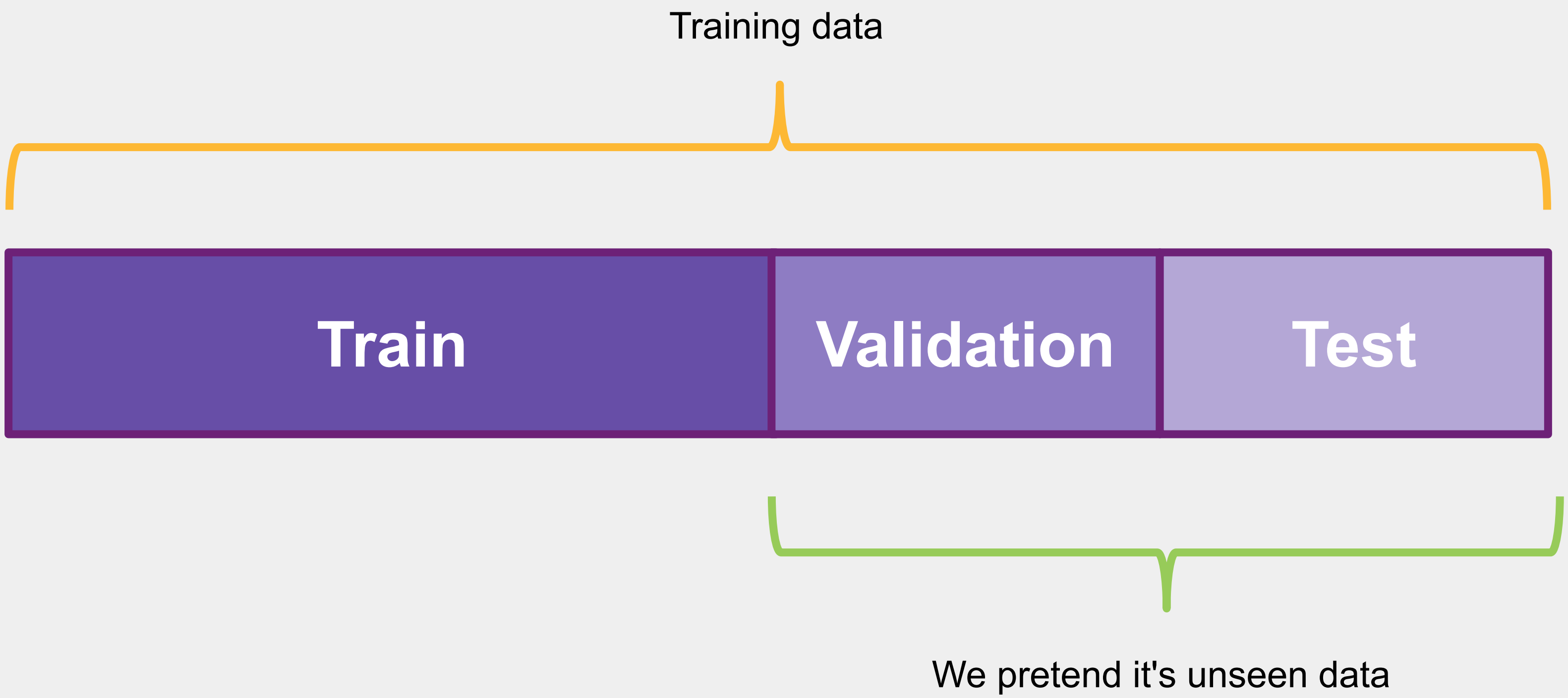
1. Model selection
2. Estimate generalization power

ML101 Validation

What do we want? Generalization!

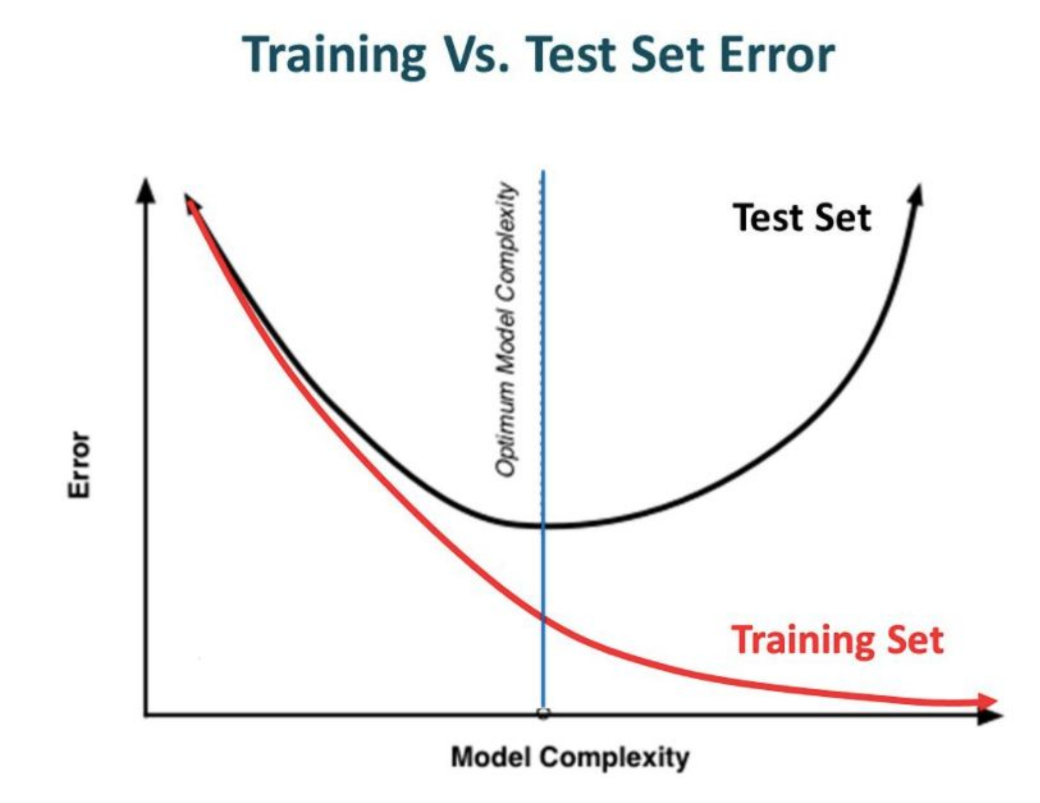


ML101 Validation: Simple split

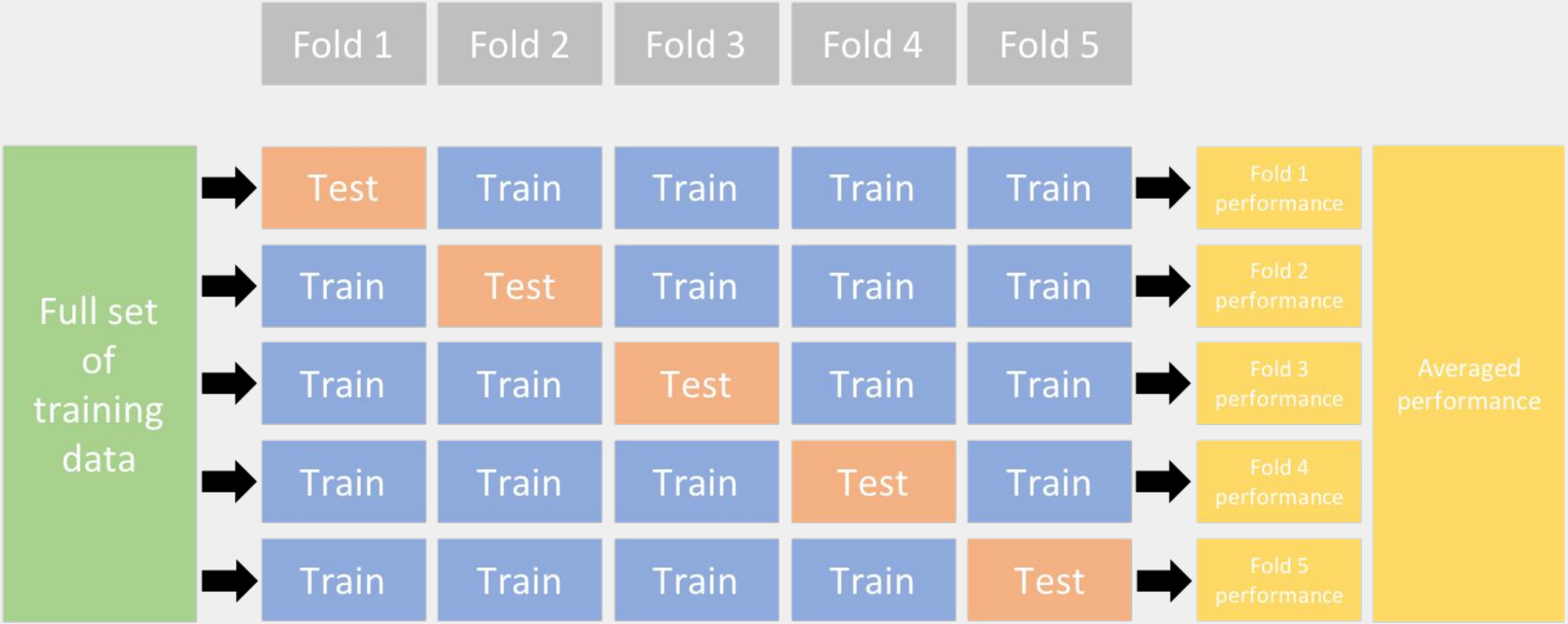


To the notebook!

Validation and overfitting



ML101 Validation: K-Fold

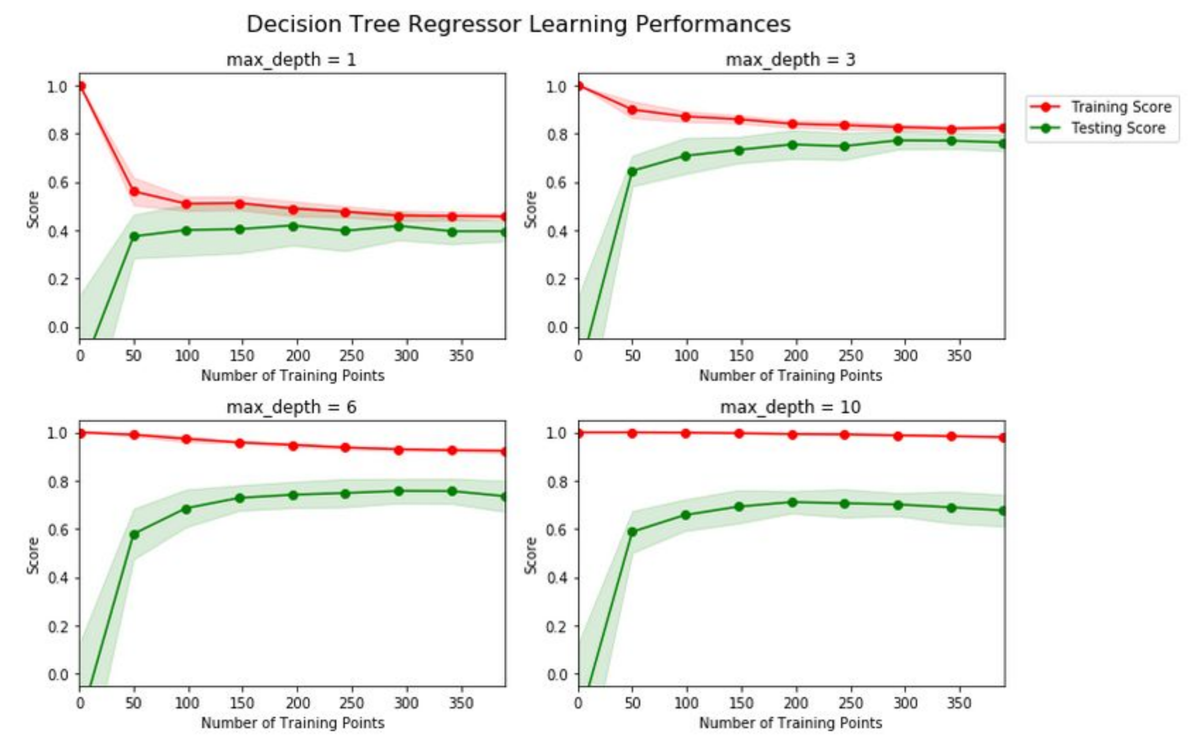


Source: **UC Business Analytics R Programming Guide**

Checkpoint!

To the notebook!

Learning Curve



Source: https://github.com/udacity/machine-learning/blob/master/projects/boston_housing/boston_housing.ipynb

Lasso and Ridge

Wanna complexity? Pay for it!

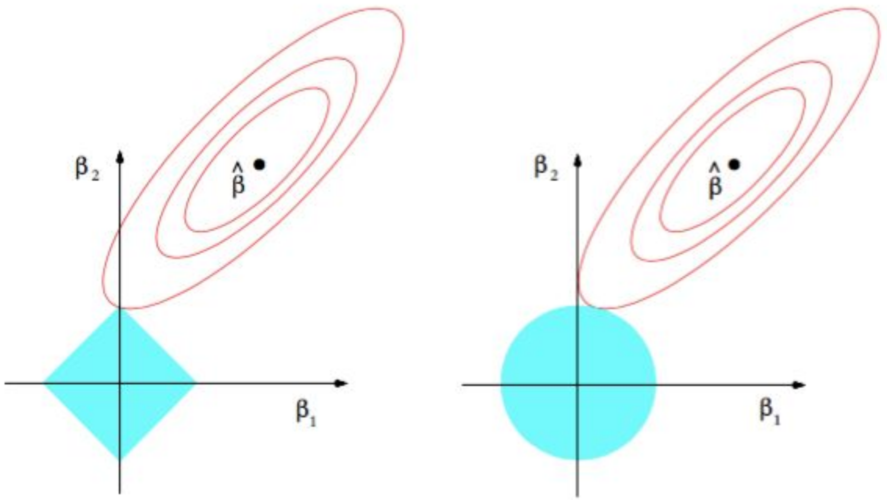


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \leq t$ *and* $\beta_1^2 + \beta_2^2 \leq t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

Source: 'The Elements of Statistic Learning'.

Checkpoint!

To the notebook!

ML101 Validation

So after your ML101 classes it may look very clear:

We want generalization, i.e. performing well on unseen data, so:

1) Leave some data out of the training process and pretend it's unseen;
2) Check if the learned model performs well on this unseen data;
3) If it performs reasonably, pick it!
4) Put in production!

What could possibly go wrong?

Then you go to the real world and...

Real World Supervised Learning

$$\mathbf{X} \xrightarrow{f} \mathrm{y}$$

Real World Supervised Learning

$$\xrightarrow{f} \mathrm{y}$$

Well, it turns out that in **most of the cases** the $\mathbf{X}$ is **mutant**!

Real World Supervised Learning

$\xrightarrow{f} \mathrm{y}$

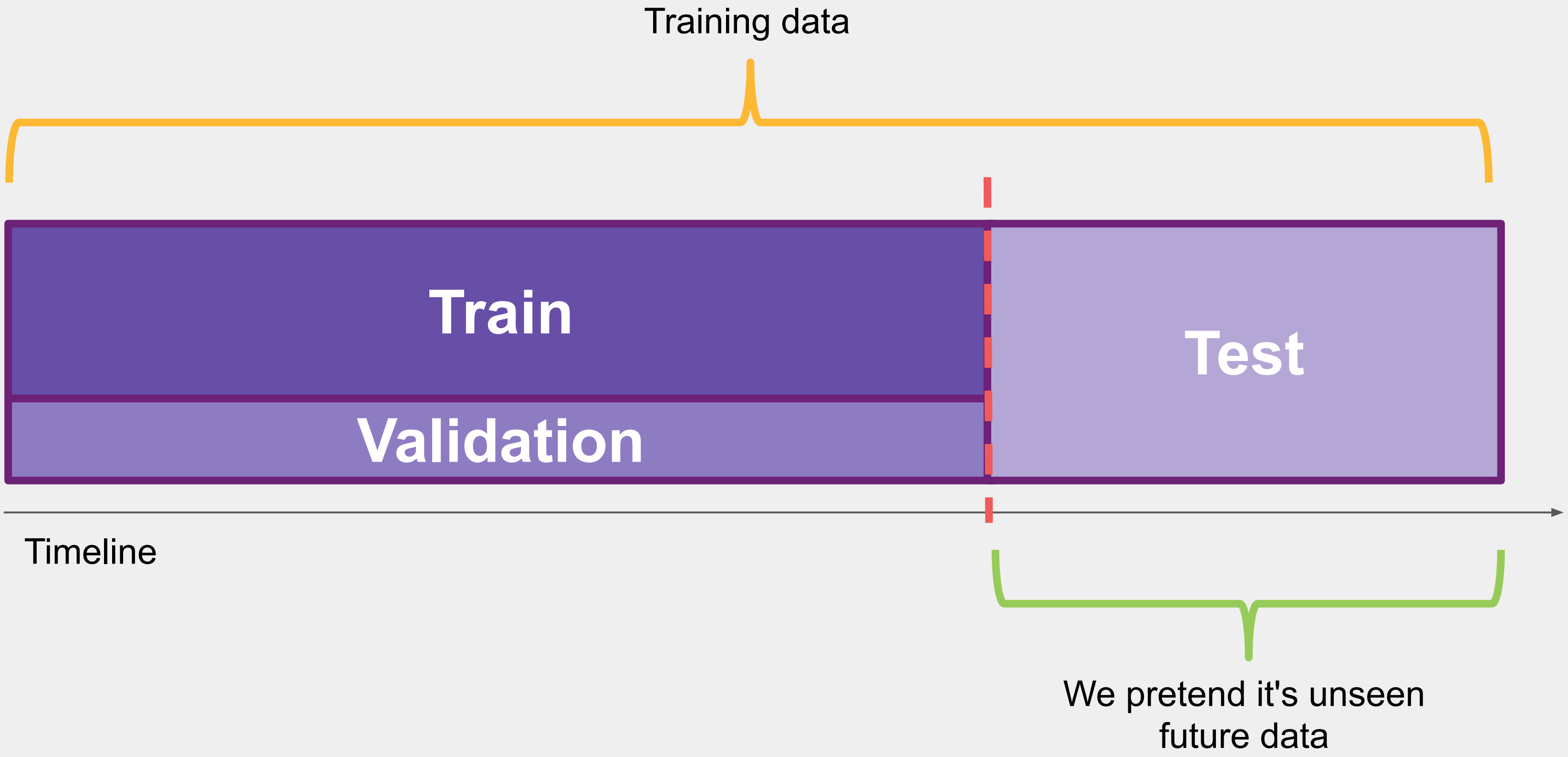
Well, it turns out that in **most of the cases** the $\mathbf{X}$ is **mutant**!

- Temporally
- Spatially

Also:

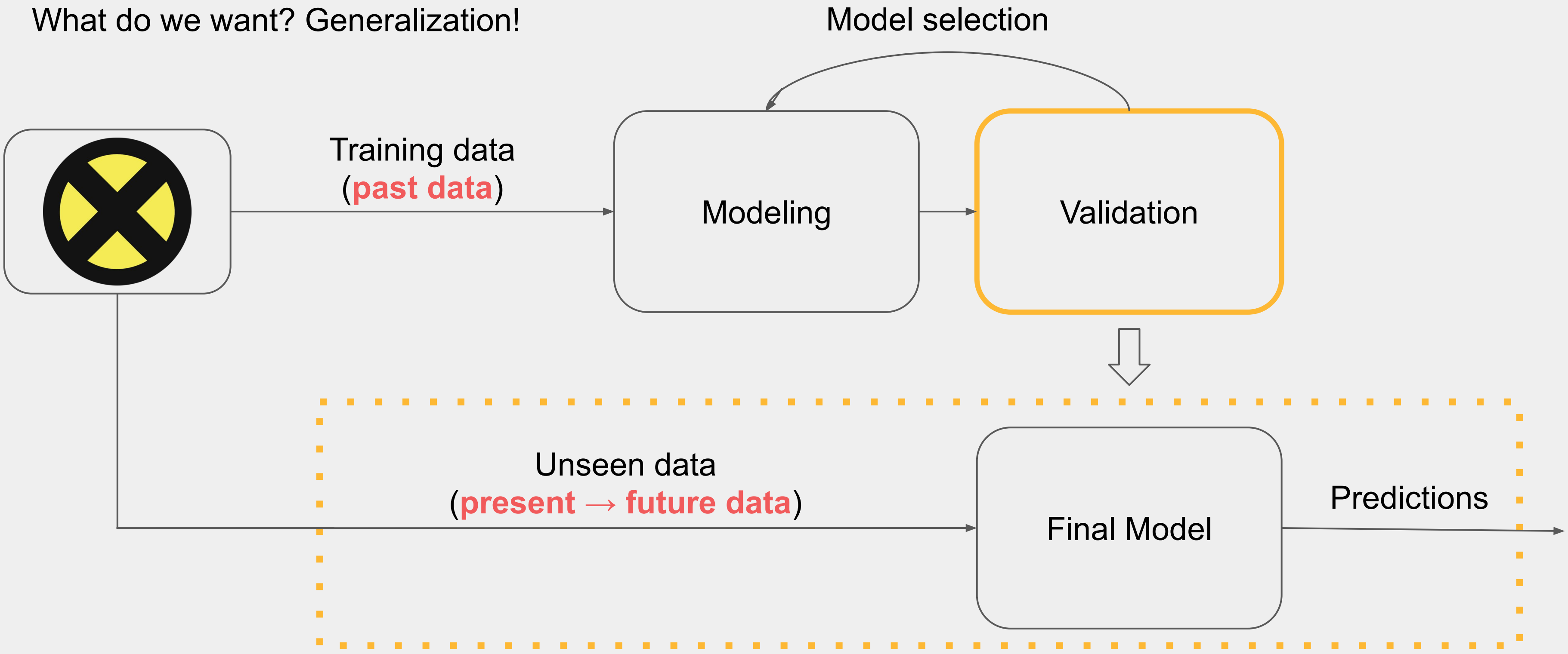
- The training data may be just a subset
- It's not perfectly distributed along its features

Real World Validation: Temporal split



Real World Validation

What do we want? Generalization!



When temporal validation can help us?

Basically, **always**!

All datasets have a temporal aspect because it is generated as the time passes by, but its effect depend on the problem.

Weak

- Images
- Text

Strong

- Time series
- Tabular data

To the notebook!

Ready to rock!

Ok, let's recapitulate:

- Now you know the **inherent role of time** in every dataset
- You can design a validation schema that **captures the time**

Now imagine an analyst come to you and say...

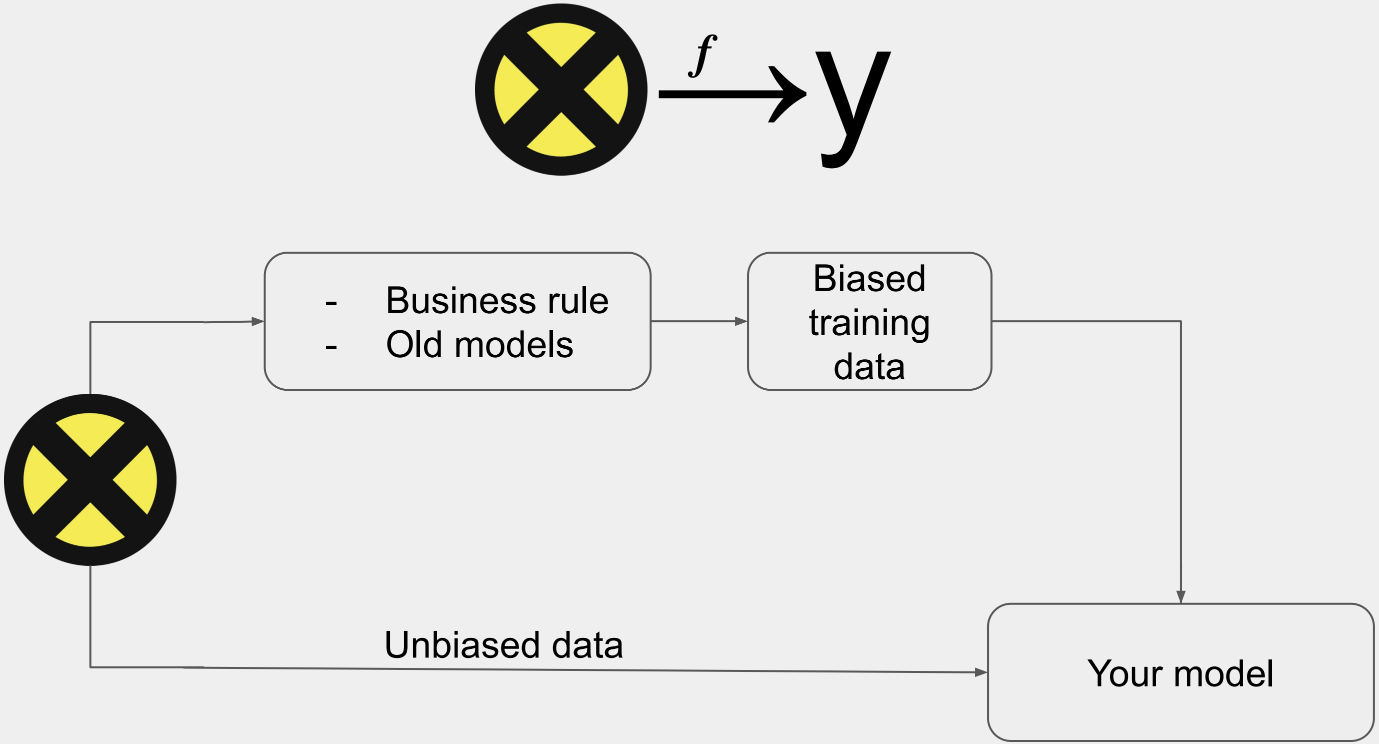
"We have some rules to decide what to do: we apply some IFs and..."

"Oh, do you think you can improve it?"

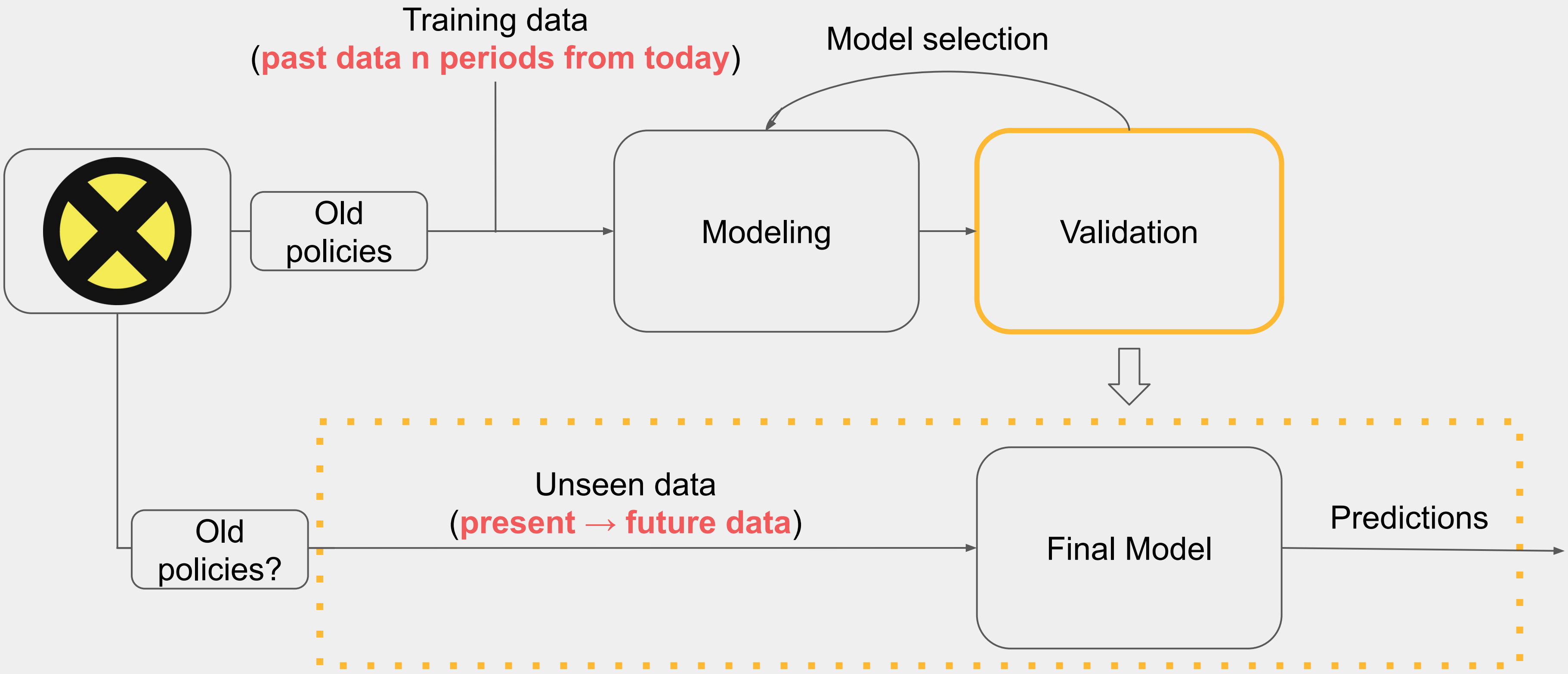
You develop a new model and replace all the old stuff!

But then your model fails miserably and you don't get what you're missing!

Old policies and models bias



Real World Validation

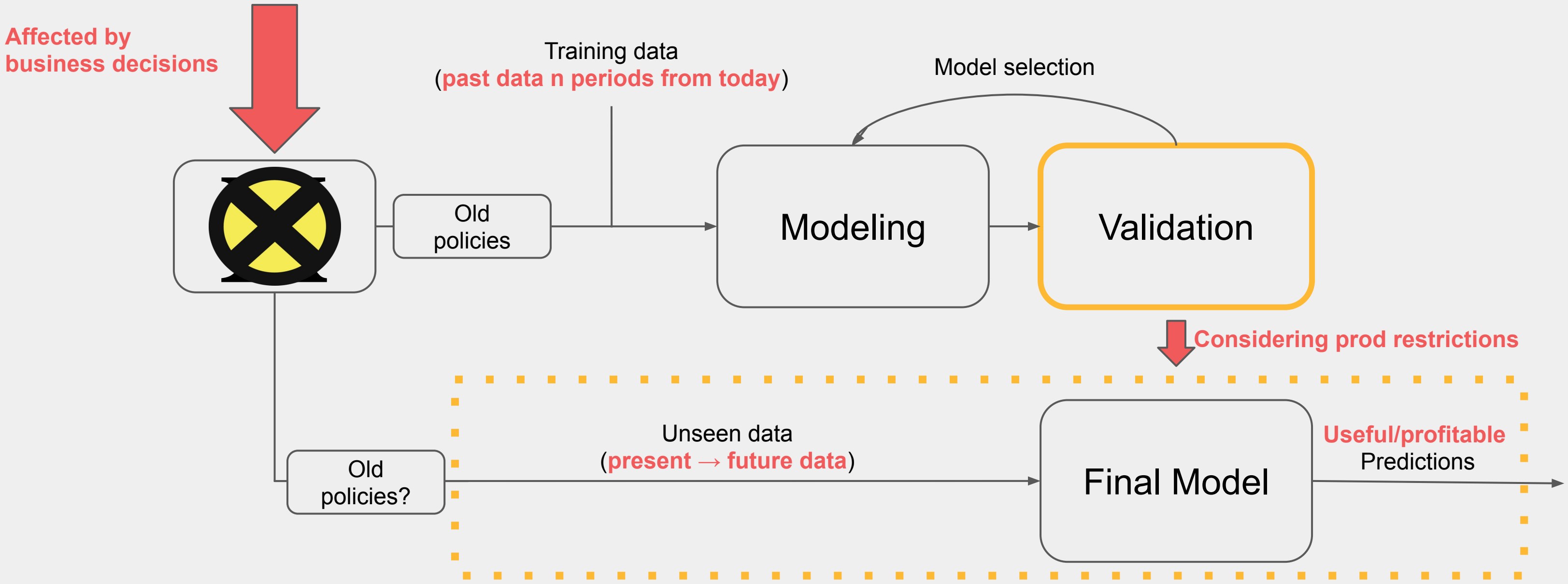


Business and Models

Business

- A lot of things can change the $\mathbf{X}$ distribution:
    - Marketing
    - New products
    - Communication
    - Growth/maturity

Real World Validation



Business

A lot of things can change the $\mathbf{X}$ distribution

You can't do anything at validation time, but **monitor**! You shipped something to score over X, but people won't care about, while you should.

So at the end...

X	
Train: A nice and invariant distribution I have a reasonable random sample. Apply: In an unseen random sample.	Train: Old, far from prediction time, biased by old policies and models, unequally distributed in the features you care about. Apply: In an unseen future data I'm not sure about how it's going to change accordingly to time and other business decisions.

Takeaways

It's hard to define a recipe for validation, but keep in mind the general idea of "**mimic the application case**":

- Use a **temporal split**
- Do a internal research about **how the data was collected** to be aware of all the old policies and **its bias**
- Know **how/when** your model is going to be applied
- Be aware in **population shifts** caused by business decisions

After class (notebook)

Twitter: @lgmoneda
E-mail: lgmoneda@gmail.com
Blog: http://lgmoneda.github.io/

Questions?