Digital Technologies and Value Creation

Dr. Philippe Blaettchen
Bayes Business School (formerly Cass)

Learning objectives of today

Goals:

- Revisit linear regression from a machine learning perspective (prediction)
- Understand more general types of regression
- Understand some common pitfalls of supervised learning and how to overcome them

How will we do this?

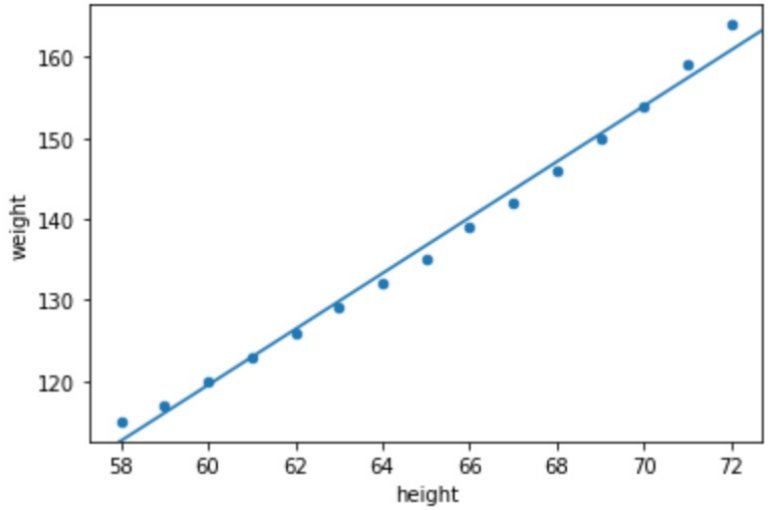
- Make use of our prior knowledge about regression
- Activity: M. Gelato's ice-creams
- Learn about training, testing, and validation sets

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Linear regression and prediction

Linear regression recap

- What do we do in linear regression?



- How do we add this line?
  - **Input:** datapoints (x_i, y_i) – 15 here
  - **Goal:** find numbers $(a = intercept, b = slope)$ such that sum of residuals squared
    - $(y_1 - a - b \cdot x_1)^2 + \cdots + (y_n - a - b \cdot x_n)^2$ is smallest possible
  - **Output:** (a, b) provide an explanation of the influence of height on weight (under some assumptions): $y_i = a + b \cdot x_i + \varepsilon_i$

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

A new context: supervised machine learning

Recall from the previous lecture:

What is the difference
between **supervised** and **unsupervised** learning?

Data with **labels** or not.

→ There is a notion of what the “right” answer is in supervised learning, and the objective is to correctly predict this answer

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Regression in the context of supervised machine learning

Why do we do linear regression (in a machine learning context)?

> The goal is to predict new values.

For example, if we know height we should be able to infer weight by using our regression line:

$$y_i^{pred} = a + b \cdot x_i$$

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Linear regression in Python

- Two methods: statsmodels (stats viewpoint) and scikit (ML viewpoint)
- Statsmodels and scikit both have pros and cons: need to know how to modify examples given to you

```
X=df[["height"]]
Y=df[["weight"]]

lm = LinearRegression().fit(X, Y) # Fit a

print("Intercept = ",lm.intercept_) # Pri

print("Model coefficients = ", lm.coef_)

print("R^2 =",lm.score(X,Y)) # Print the
```

- Define your independent variables in X, your dependent variable in Y
  - With scikit, you don't have to add the constant manually
- Define your model, linear regression here
- Fit the linear regression
- Print properties of the model (R^2 and coefficients)

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

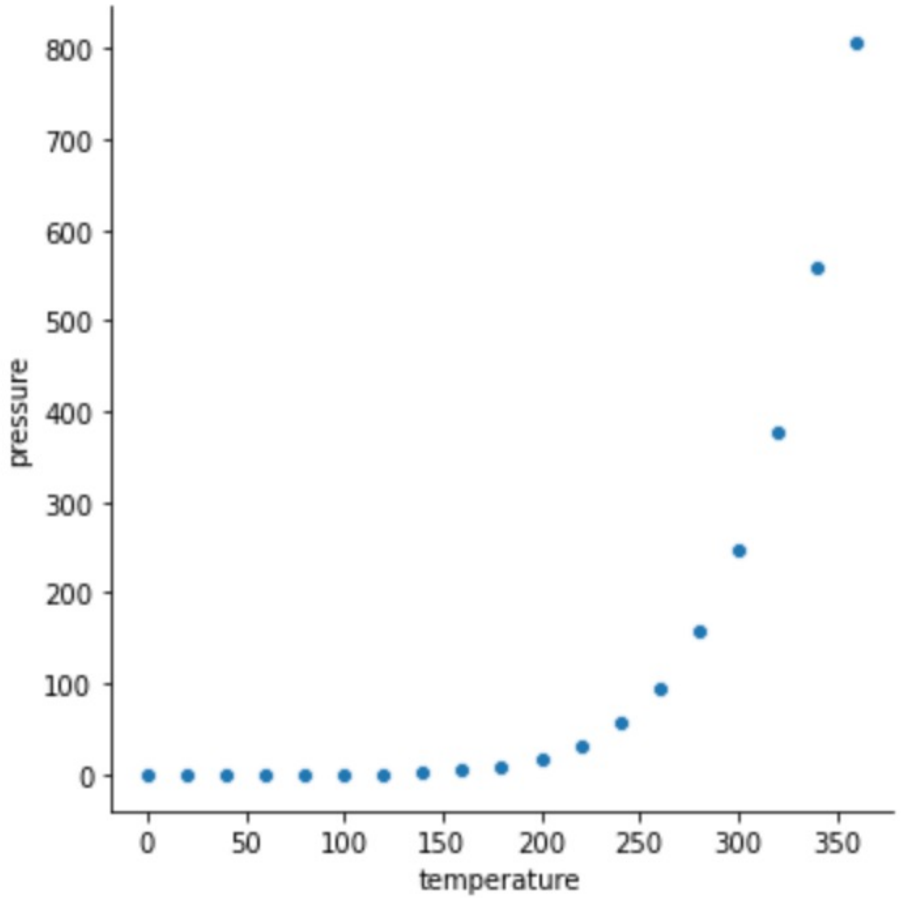
Let’s see linear regressions in scikit

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Regression more generally

Linear regression is not always good enough…

Many relationships are simply not linear:

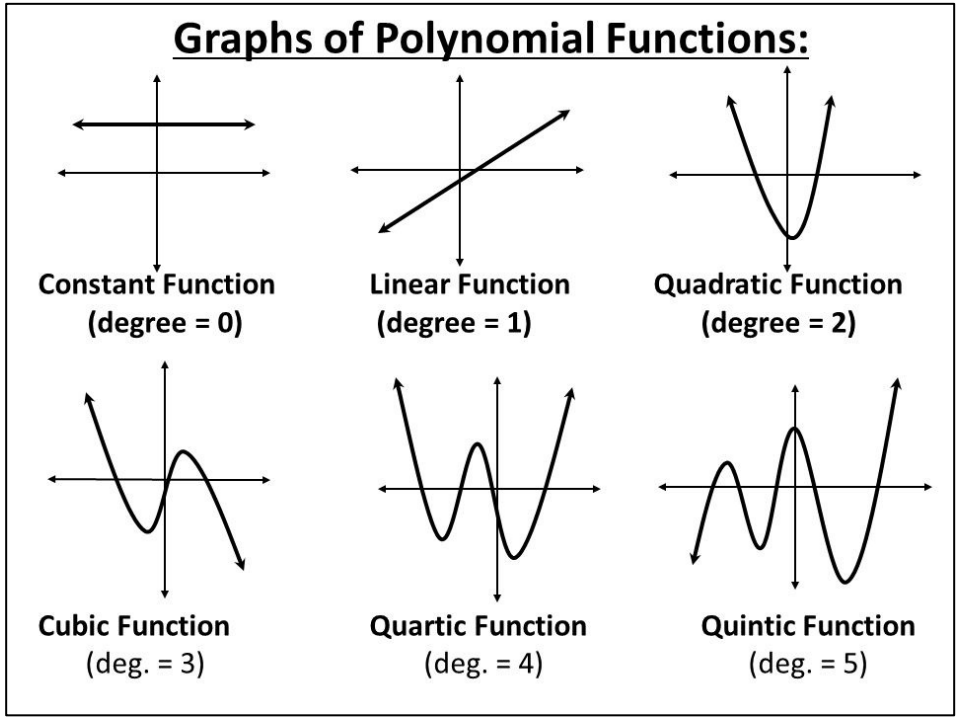


BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Regression

- In (general) regression, our goal is to fit a **curve** to data (not just a line as in linear regression).
- Of particular interest to us now is going to be **polynomial regression.**

Why? It is “easy” to do like linear regression but is more versatile.



Source:http://brandon.ai/

BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

How to do polynomial regression in Python?

Selling ice-cream

M. Gelato is the owner of an ice-cream parlor.

- Over the course of the year, he has written down some of his **daily sales**.
- He knows that his sales are **periodic**: i.e., every year, he sells in approximately the same way
- He would like to fit a curve to his data so as to better predict what his sales are going to be next year.

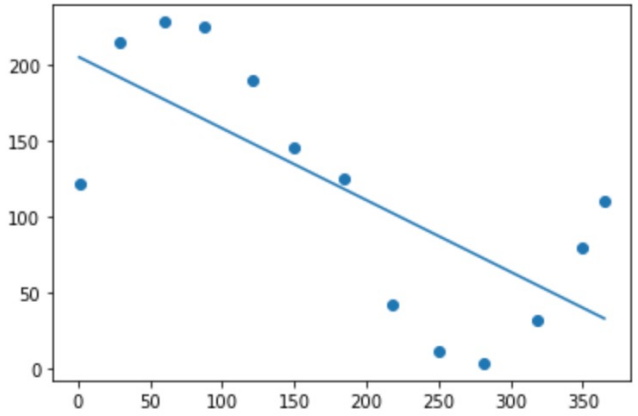
Goal: help M. Gelato!

Activity: open the Video-exercise Notebook and complete Part 3

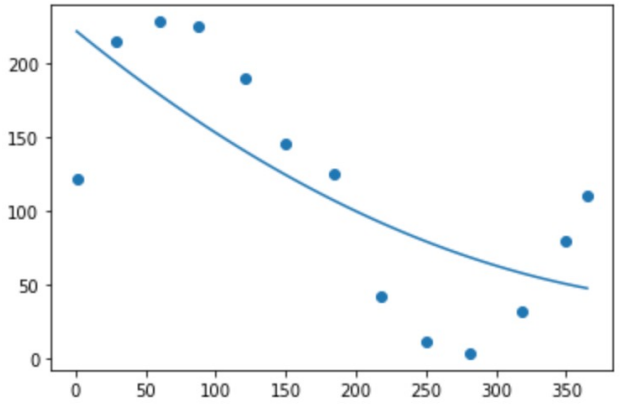
BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Overfitting and the basics of supervised learning

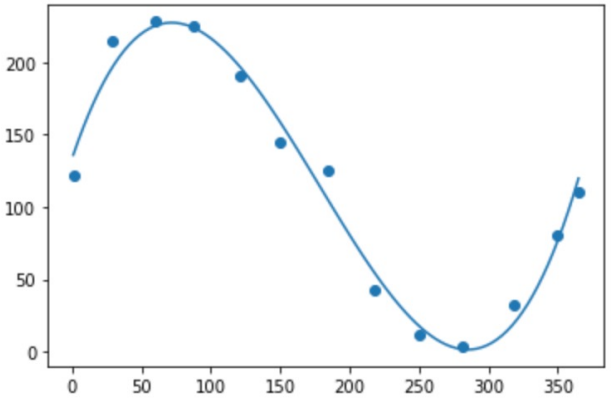
Activity wrap-up



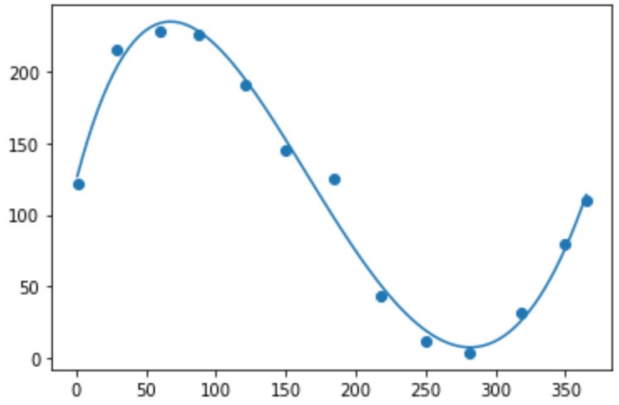
d=1, R^2=0.516



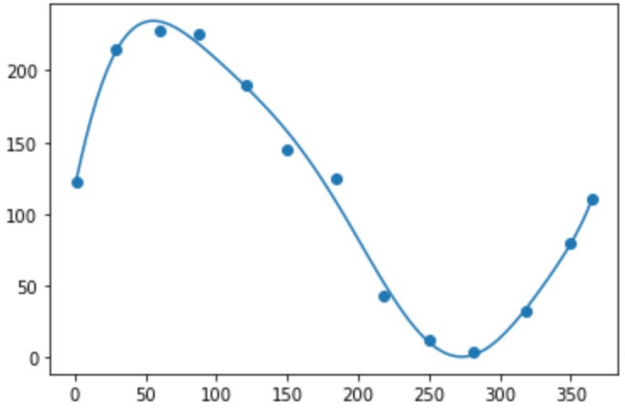
d=2, R^2=0.531



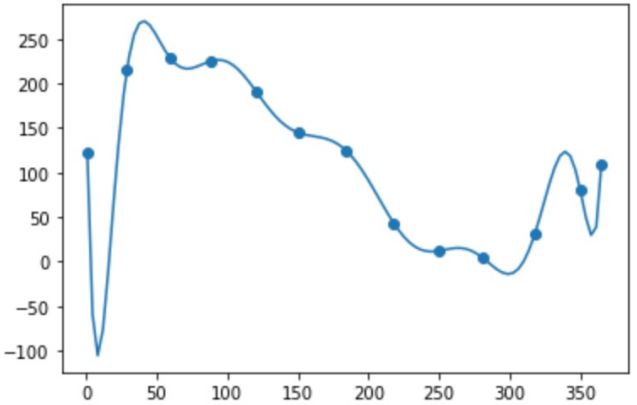
d=3, R^2=0.979



d=5, R^2=0.985



d=8, R^2=0.992



d=12, R^2=1

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Activity wrap-up

- **R^2 increases** as the degree increases as there is more and more “freedom” in how the polynomial curve can twist.
- **The fit gets stranger and stranger** (negative values when d=12…)
- **Conclusions:**
  - This is known as **overfitting:** regression curve fits “too closely” to existing datapoints.
  - Ends up **not reflecting reality** as too tailored to the dataset we have: poor prediction abilities
  - Need to find **new ways of measuring** what a “good fit” is!

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Testing versus training

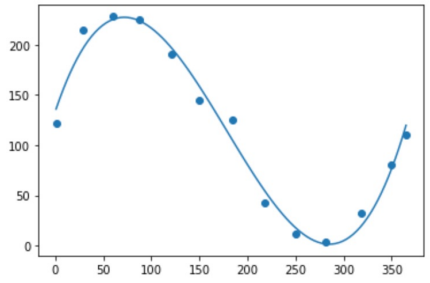
- Not good enough to have a regression that works well **just on the data we have at hand**
- Need to know whether it will perform well **for new data as well**
- What is a **good measure** of whether it will perform well on new data?

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

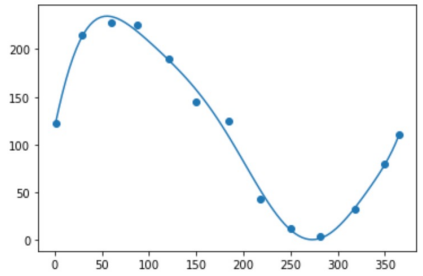
Training and testing set in practice

Example: Polynomial regression

1. Use the **training set** to fit the polynomial regression



d=3



d=8

2. Use the **testing set** to evaluate how well the model you picked would perform on new data.

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Testing versus training

- **Train** the model on the training set (e.g., run the regression)
- **Test** the model on the testing set to get an indication of its performance
- This is called the **"hold-out" method**
  - **What kind of metrics to use for testing?**

$$\text{Root Mean Squared Error (RMSE)} = \sqrt{\frac{1}{n} \cdot \sum_{i=1..n} (prediction_i - actual_i)^2}$$

n = number of datapoints in test set

$$\text{Mean Absolute Percentage Error (MAPE)} = \frac{1}{n} \cdot \sum_{i=1...n} \frac{|prediction_i - actual_i|}{|actual_i|}$$

BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

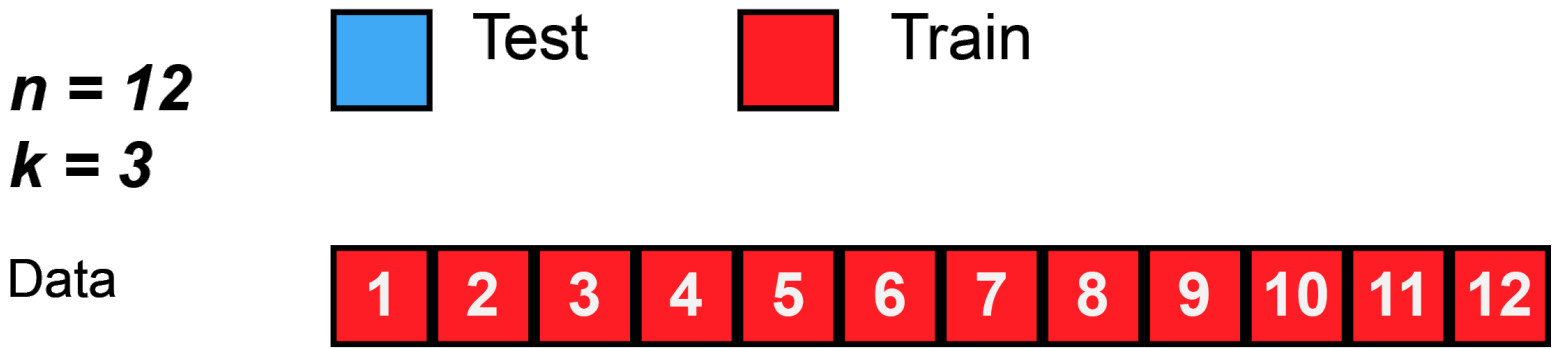
Let’s try it out

Cross-validation

- Dividing the data into one testing-training set gives you just **one measure** of the error of the method.
- It may be useful to have **more than one** measure of error.

How can we get this? **Cross-validation!**

1. Split your data into k sets.
2. Train on k-1 of them and test on the k^{th} one
3. Repeat for every subset of k-1 sets
4. Obtain k measures of error: can consider average, standard deviation.



Source: Wikipedia

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Validation set

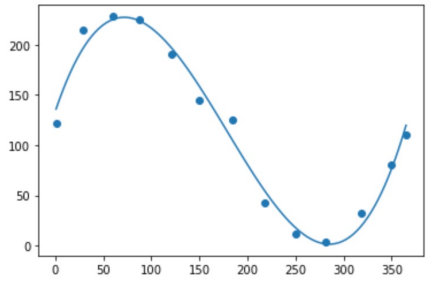
- The test set only gives you **a measure** of how good your model is.
- This is okay when you have **only one model** you want to try.
- But what if you want to **select** among many models?

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

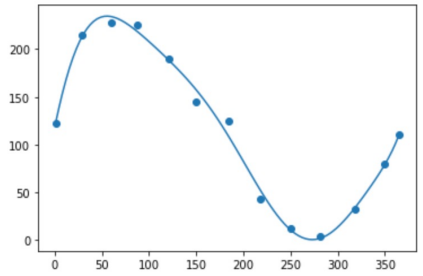
Validation set in practice

Example: Polynomial regression

1. Use the **training set** to come up with polynomial regressors with different degrees.



d=3



d=8

2. Use the **validation set** to pick which degree is the best, e.g., $d = 3$.

3. Use the **testing set** to evaluate how well the model you picked would perform on new data.

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Let’s try it out

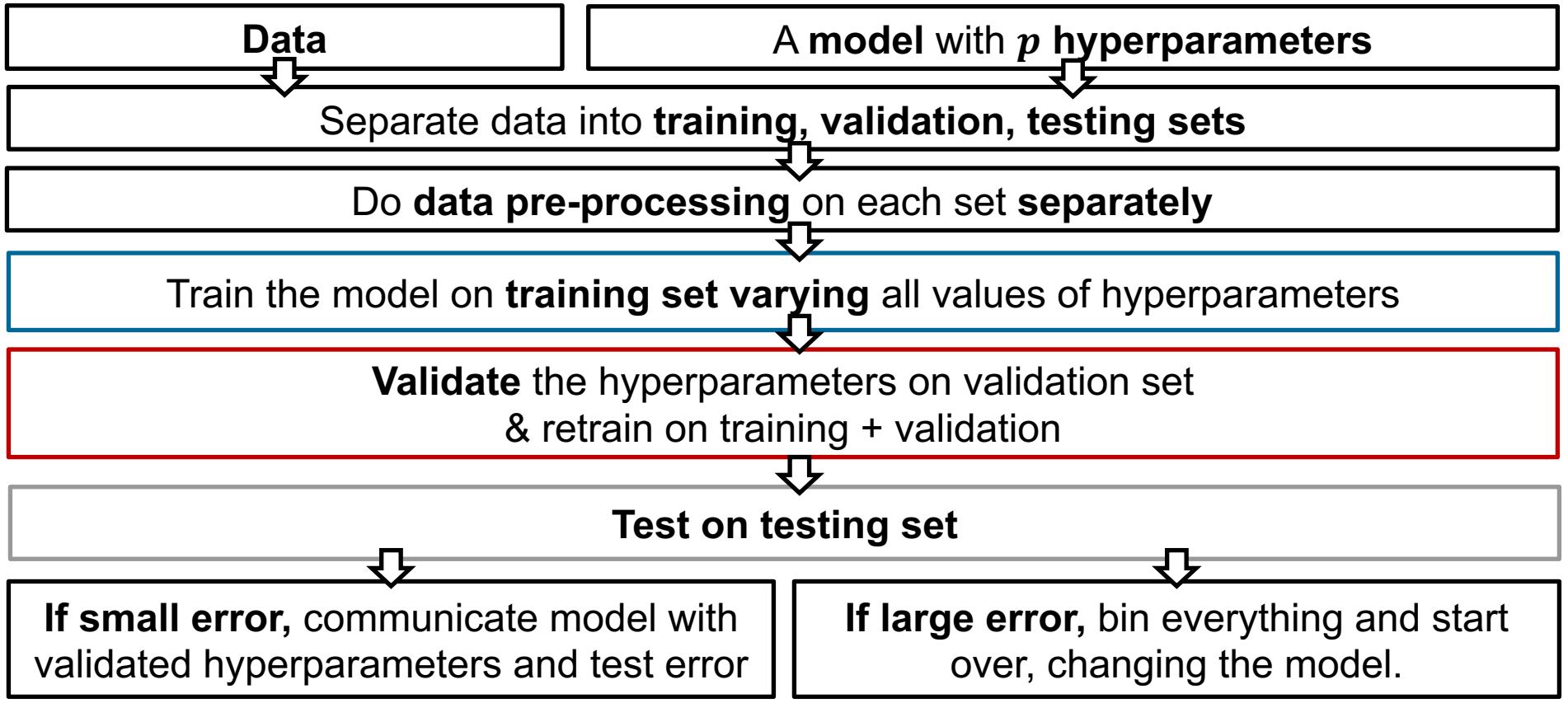
Using these concepts in Python

We use scikit here as it has automated ways to deal with these concepts.

What we want to do	In Python
Divide datasets X and Y into training and testing sets	from sklearn.model_selection import train_test_split X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)
Divide datasets X and Y into training, validation, testing sets	Use the previous function twice: • once to divide the set into testing and other • once to divide other into training and validation
Compute MSE	From sklearn.metrics import mean_squared_error mean_squared_error(y_true,y_pred)
Compute MAPE	Part of the unreleased version of scikit - TBD
Cross-validation	Use cross_val_score Complicated & Non-mandatory: see documentation

BAYES
USINESS SCHOOL
UNIVERSITY OF LONDON

Supervised learning process



- Ideally: have different validation sets for each parameter so as not to **overfit the validation set**
- **Requires a lot of data / computation power**

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Pitfalls in supervised ML

- Saw one already:

Overfitting

When the model is fitted too closely to the data

- How do we get round it?
  - This is the purpose of the test set
  - If your model is performing well on the training set but not on the test set, then you are overfitting
  - If this is happening, start over

Train-test contamination

When the testing set serves to fine tune the model when it should only be used to evaluate it.

- Examples of occurrence:
  - If **data pre-processing** (imputation with the mean, scaling/ normalizing) is done on training and testing sets at the same time
  - If **hyperparameters are tweaked** based on the testing set performance

- Model contains testing set information: the performance observed on the testing set **does not reflect reality.**

Target leakage

When the features include information that wouldn't be available at time of prediction (typically based on the actual prediction)

got_pneumonia	age	weight	male	took_antibiotic_medicine	...
False	65	100	False	False	...
False	72	130	True	False	...
True	58	100	False	True	...

Source: Kaggle

- How to avoid this?
  → Know your data and what your features represent!

- Other issues related to quality of input data: cherry-picking, training/testing/validation sets which are not homogeneous.


See you in class!