Introduction to Machine learning with scikit-learn

Important ideas

What is machine learning?

- extracting knowledge from data
- closely related to statistics and optimization.
- What distinguishes machine learning is that it is very focused on prediction.

Types of Machine Learning

- Supervised
- Unsupervised
- Reinforcement
-

Types of ML

- **Supervised** learning: Models that can predict labels based on **labeled data**
  - Classification: Models that predict labels as two or more discrete categories
  - Regression: Models that predict continuous labels
- **Unsupervised** learning: Models that identify structure in unlabeled data
  - Clustering: Models that detect and identify distinct groups in the data
  - Dimensionality reduction: Models that detect and identify lower-dimensional structure in higher-dimensional data.

Supervised Learning

$$(x_i, y_i) \propto p(x, y) \text{ i.i.d.}$$

$$x_i \in \mathbb{R}^p$$

$$y_i \in \mathbb{R}$$

$$f(x_i) \approx y_i$$

- Given an array of test results from a patient, does this patient have diabetes?

The x_i would be the different test results, and y_i would be diabetes or no diabetes.

- Given a piece of a satellite image, what is the terrain in this image?

Here x_i would be the pixels of the image, and y_i would be the terrain types.

Unsupervised Learning

$$x_i \propto p(x) \text{ i.i.d.}$$

Learn about p.

- discovering topics in news articles or on twitter, or grouping data into clusters for easier analysis.
- outlier detection, where you ask "does this data look normal" which is important for fraud detection and security systems.

Classification and Regression

Classification

- target y discrete
- Is this patient sick?

Regression

- target y continuous
- How long will it take for the patient to recover?

Generalization

- Not only $f(x_i) \approx y_i$ for the data seen/used
- also for new data: $f(x) \approx y$

Supervised learning

What is the relationship between input and output variables?

IQ	ClassesAttended	CatsYouOwn	YourFinalGrade
110	14	0	73
105	28	2	99
107	26	1	95

- Input variables
  - Independent variables, predictors, input, features
- Output variables
  - Dependent variables, response, output

The relationship between X and Y

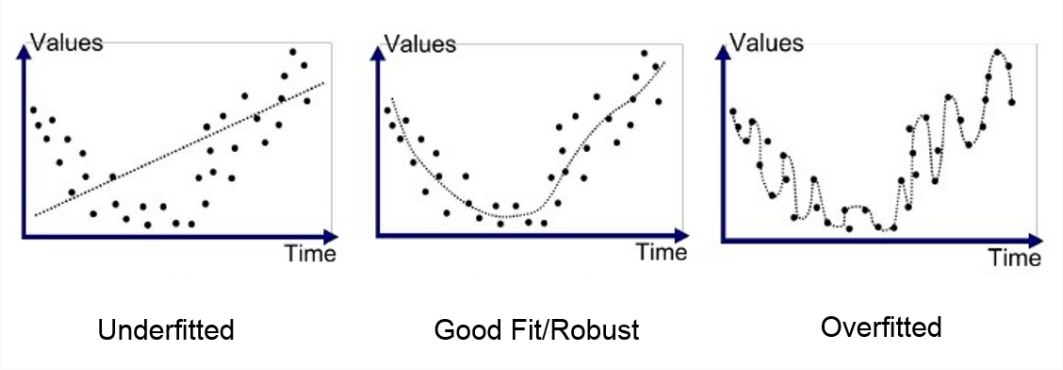
The true relationship is $Y = f(X)$

Our goal is to come up with an *estimate* $\hat{f}$ of the true f

$$\hat{Y} = \hat{f}(X)$$

- Why?
  - So we can plug in values of X and see what $\hat{Y}$ are produced
  - Think of it like a machine.

Overfitting

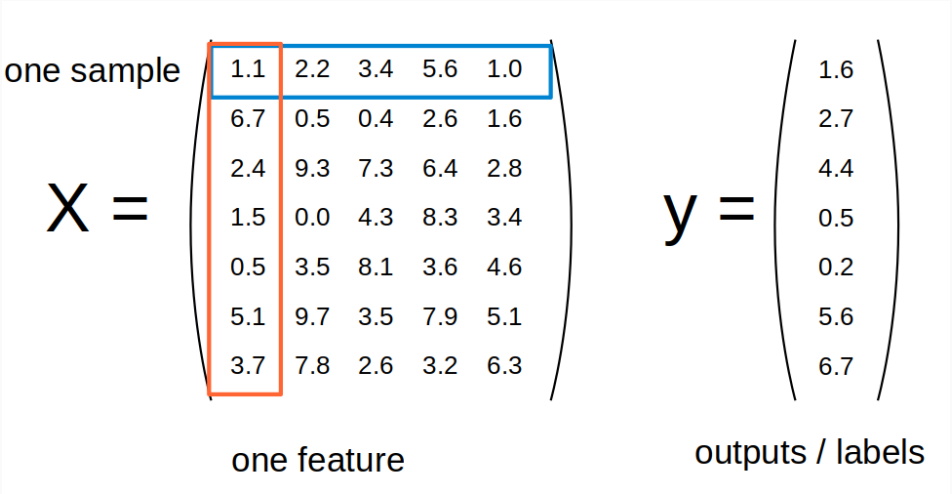


- Worst sin : overfitting

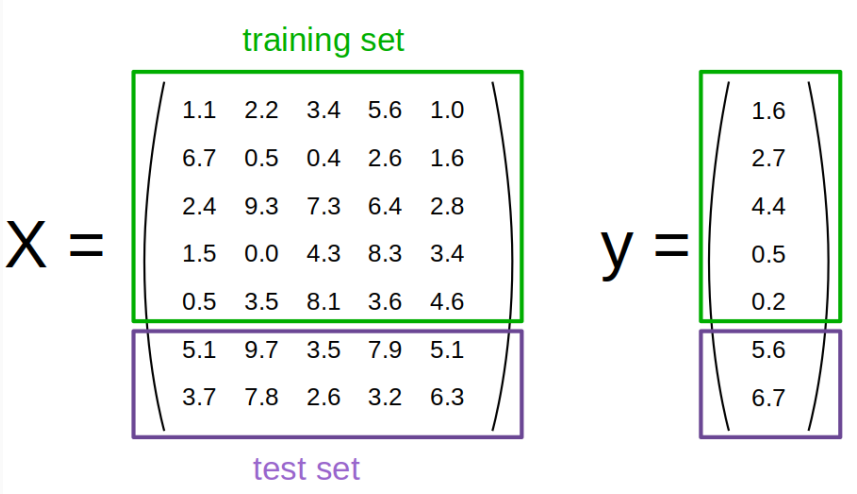
bcos the model works well for data that is seen but does poor job for unseen data

- Split data into Train-Test.

Representing Data



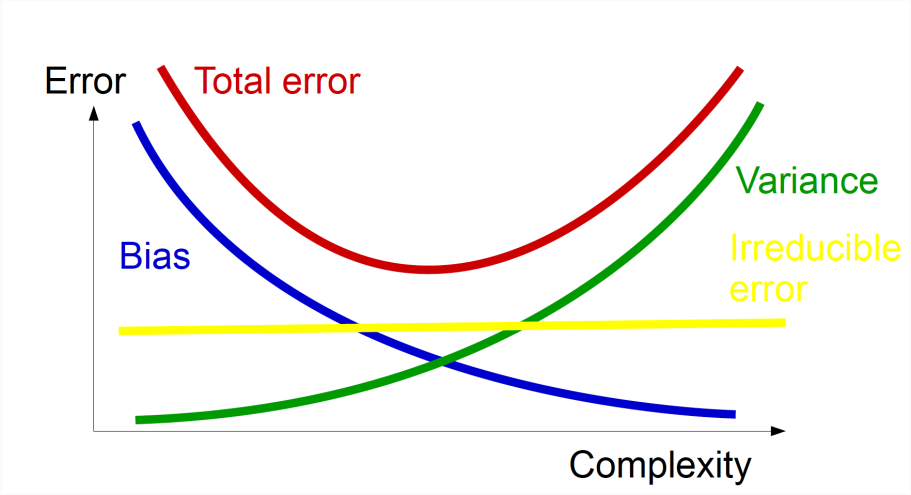
Training and Test Data



Model evaluation and selection

- train, test
- fit on train
- predict on test
- score the model by how well it does on test

- All data science in one slide: **bias variance tradeoff**



bias variance tradeoff

want the model with least error. As complexity increases, bias goes down but variance increases

- Every estimator has its advantages and drawbacks.
- Its generalization error can be decomposed in terms of bias, variance and noise.
  - The bias of an estimator is its average error for different training sets.
  - The variance of an estimator indicates how sensitive it is to varying training sets.
  - Noise is a property of the data.

Datasets

- Breast Cancer
- Wine
- Iris
- Parkinsons

Team Exercise 1

Take a look at the data

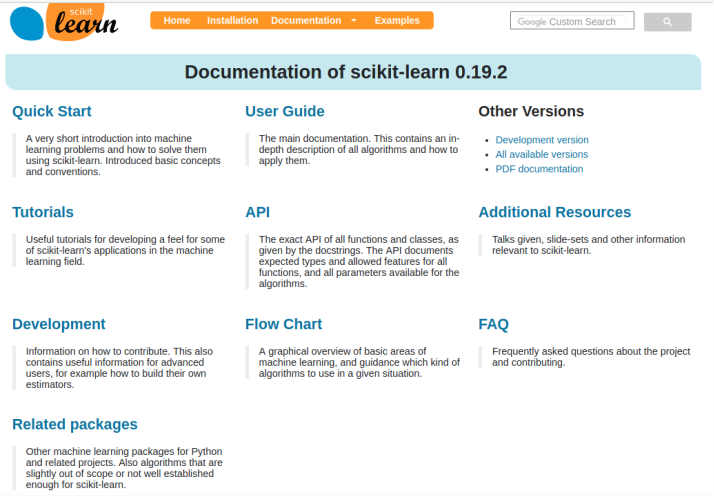
Team Exercise 2

Plot the data

Present : Explain your data set

- Explain the problem you are trying to solve
- Give some preliminary visualizations that are indicative of the difficulty of the problem and possible approaches

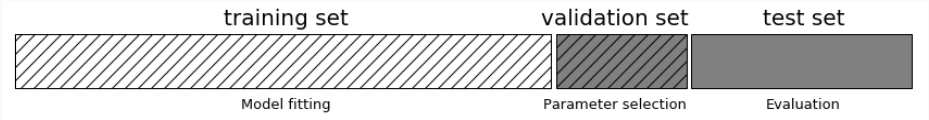
scikit-learn documentation



Team Exercise 3

Prepare Rmd slides and present one of the methods below

Threefold split



We use the training set for model building, the validation set for parameter selection and the test set for a final evaluation of the model.

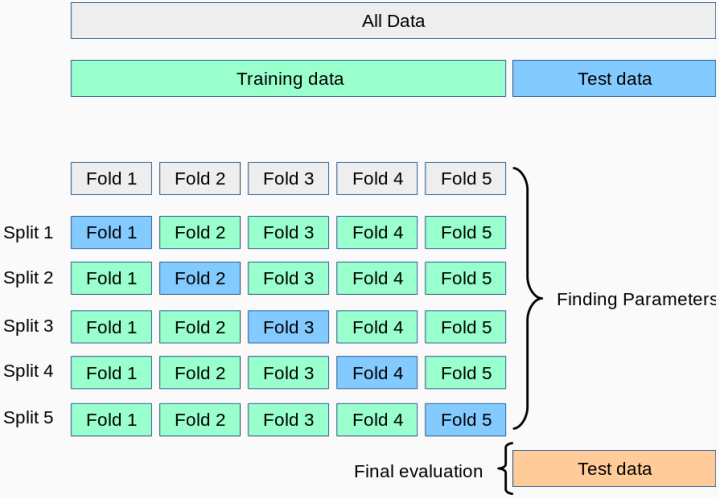
pro: fast, simple

con: high variance, bad use of data

Implementing threefold split

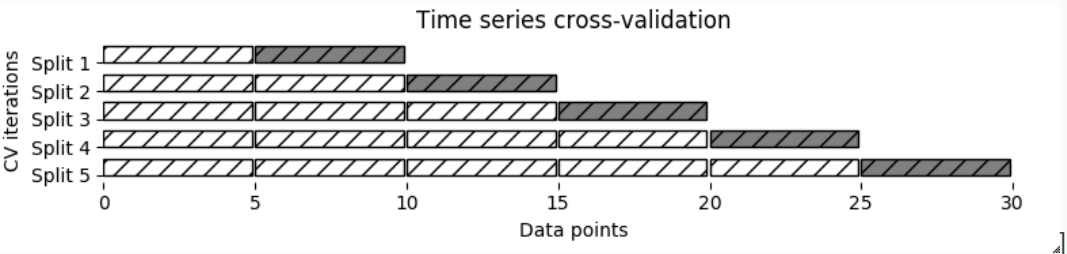
```
X_trainval, X_test, y_trainval, y_test = train_test_split(X, y)
X_train, X_val, y_train, y_val = train_test_split(X_trainval, y_trainval)

val_scores = []
neighbors = np.arange(1, 15, 2)
for i in neighbors:
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, y_train)
    val_scores.append(knn.score(X_val, y_val))
print("best validation score: {:.3f}".format(np.max(val_scores)))
best_n_neighbors = neighbors[np.argmax(val_scores)]
print("best n_neighbors:", best_n_neighbors)
```

Cross-validation + test set



TimeSeriesSplit

.padding-top[



]