Introduction to Machine Learning with Python

Andreas Müller
Columbia University, scikit-learn

https://github.com/amueller/quick-ml-intro

What is machine learning?

loan_amnt	term	int_rate	grade	home_ownership	annual_inc
5000.0	36 months	10.65%	B	RENT	24000.0
2500.0	60 months	15.27%	C	RENT	30000.0
2400.0	36 months	15.96%	C	RENT	12252.0
10000.0	36 months	13.49%	C	RENT	49200.0
3000.0	60 months	12.69%	B	RENT	80000.0
5000.0	36 months	7.90%	A	RENT	36000.0
7000.0	60 months	15.96%	C	RENT	47004.0
3000.0	36 months	18.64%	E	RENT	48000.0
5600.0	60 months	21.28%	F	OWN	40000.0
5375.0	60 months	12.69%	B	RENT	15000.0

loan_status
Fully Paid
Charged Off
Fully Paid
Fully Paid
Fully Paid
Fully Paid
Fully Paid
Fully Paid
Charged Off
Charged Off

Supervised Learning

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

$$f(x_i) \approx y_i$$

Most common applications:
- Automate a manual task
- Predict the future

Classification and Regression

Classification:

- y discrete

Will they subscribe?

Regression:

- y continuous

How much will the returns be?

Generalization

Not only $$f(x_i) \approx y_i$$

Also for new data: $$f(x) \approx y$$

Home Installation Documentation Examples

Google™ Custom Search
Search ×

Documentation of scikit-learn 0.17

Quick Start

A very short introduction into machine learning problems and how to solve them using scikit-learn. Introduced basic concepts and conventions.

User Guide

The main documentation. This contains an in-depth description of all algorithms and how to apply them.

Other Versions

- scikit-learn 0.18 (development)
- scikit-learn 0.17 (stable)
- scikit-learn 0.16
- scikit-learn 0.15

Tutorials

Useful tutorials for developing a feel for some of scikit-learn's applications in the machine learning field.

API

The exact API of all functions and classes, as given by the docstrings. The API documents expected types and allowed features for all functions, and all parameters available for the algorithms.

Additional Resources

Talks given, slide-sets and other information relevant to scikit-learn.

Contributing

Information on how to contribute. This also contains useful information for advanced users, for example how to build their own estimators.

Flow Chart

A graphical overview of basic areas of machine learning, and guidance which kind of algorithms to use in a given situation.

FAQ

Frequently asked questions about the project and contributing.

http://scikit-learn.org/

Representing Data

$$X = \begin{array}{l} \text{one sample} \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix} \qquad y = \begin{pmatrix} 1.6 \\ 2.7 \\ 4.4 \\ 0.5 \\ 0.2 \\ 5.6 \\ 6.7 \end{pmatrix}$$

one feature

outputs / labels

Training and Testing Data

training set

$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix} \qquad y = \begin{pmatrix} 1.6 \\ 2.7 \\ 4.4 \\ 0.5 \\ 0.2 \\ 5.6 \\ 6.7 \end{pmatrix}$$

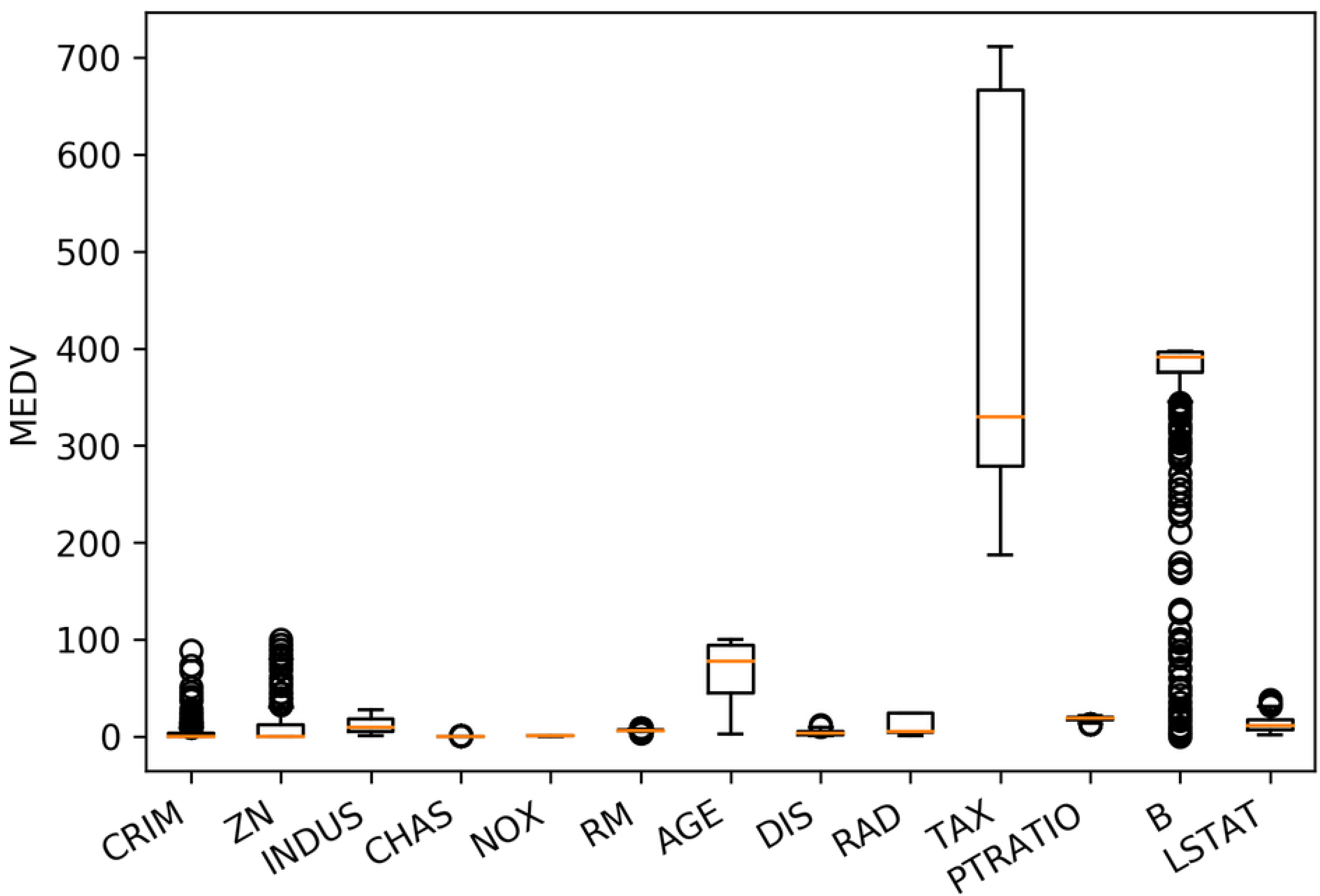
test set

IPython Notebook:
Part 1 – Data Loading

Preprocessing

```
plt.boxplot(X)
plt.xticks(np.arange(1, X.shape[1] + 1), boston.feature_names, rotation=30, ha="right")
plt.ylabel("MEDV")
```

```
<matplotlib.text.Text at 0x7f580303eac8>
```



Categorical Features

Categorical Features

$$\{\text{"red"}, \text{"green"}, \text{"blue"}\} \subset \mathbb{R}^p \quad ?$$

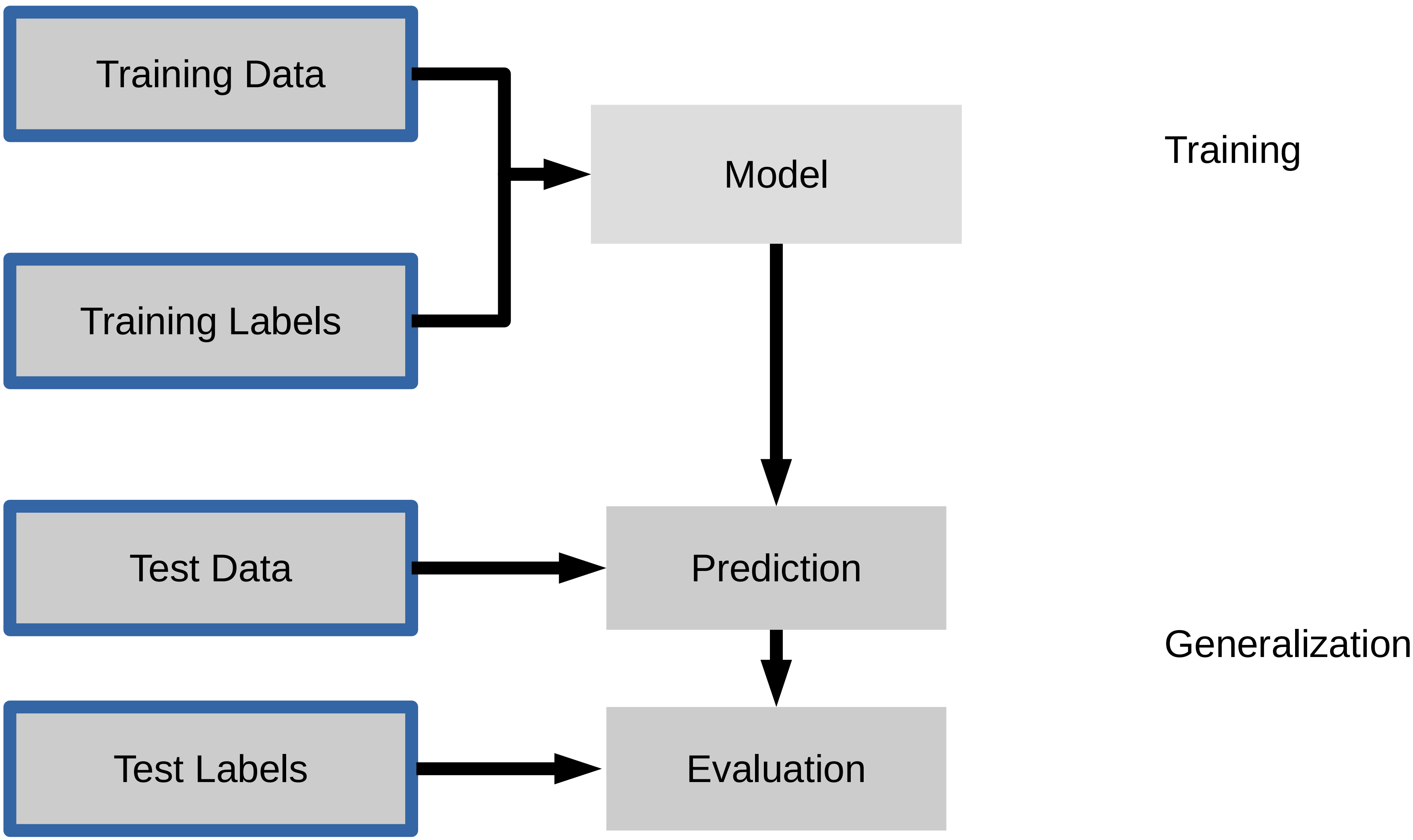
Categorical Variables

$$\begin{array}{ccc} \text{"red"} & \text{"green"} & \text{"blue"} \\ \end{array}$$

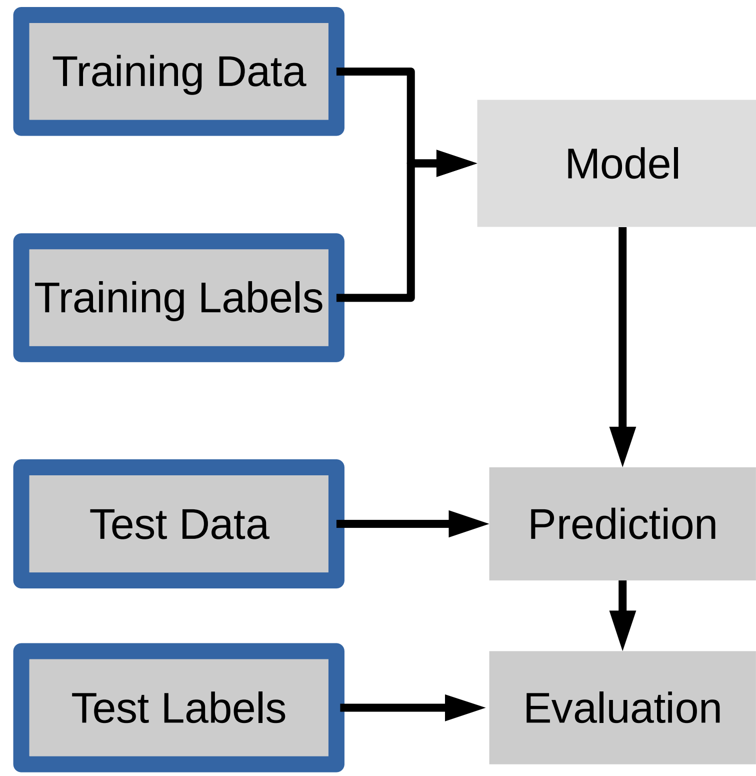
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

IPython Notebook:
Part 2 – Preprocessing

Supervised Machine Learning

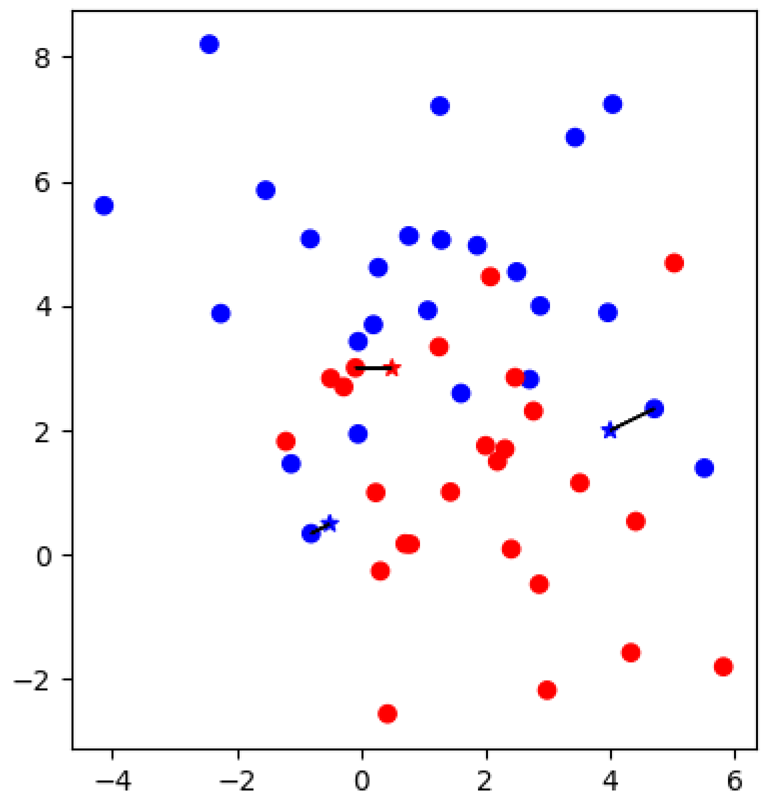


```
clf = RandomForestClassifier()

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)

clf.score(X_test, y_test)
```



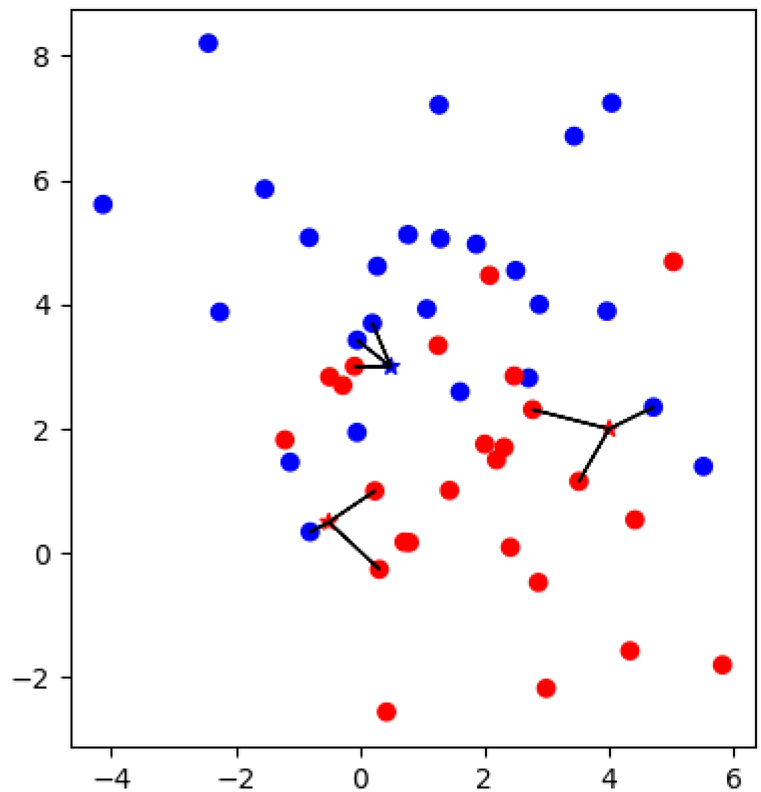
IPython Notebook:
Part 3 – Supervised Learning

Nearest neighbors

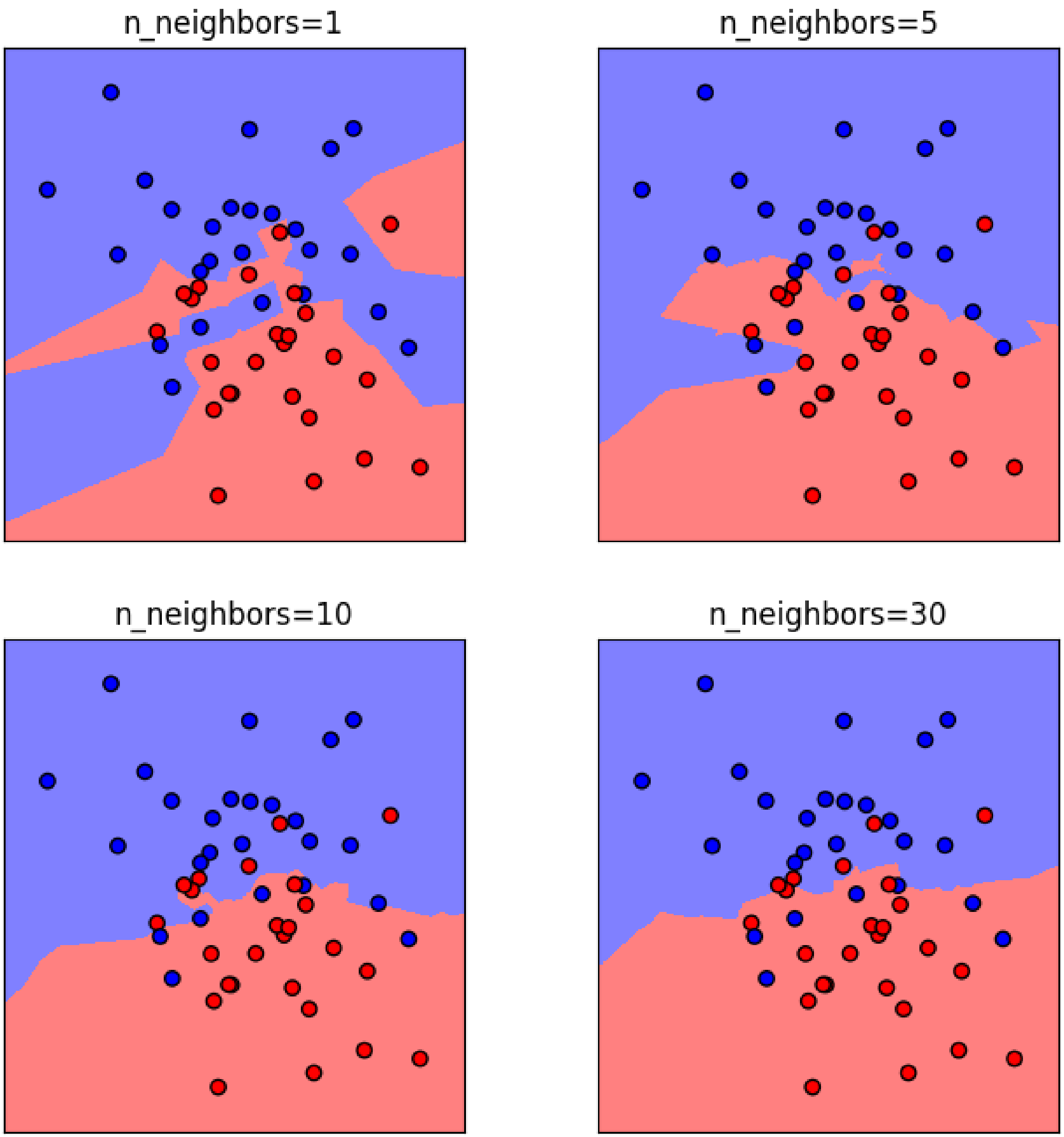


$$f(x) = y_i, i = \text{argmin}_j ||x_j - x||$$

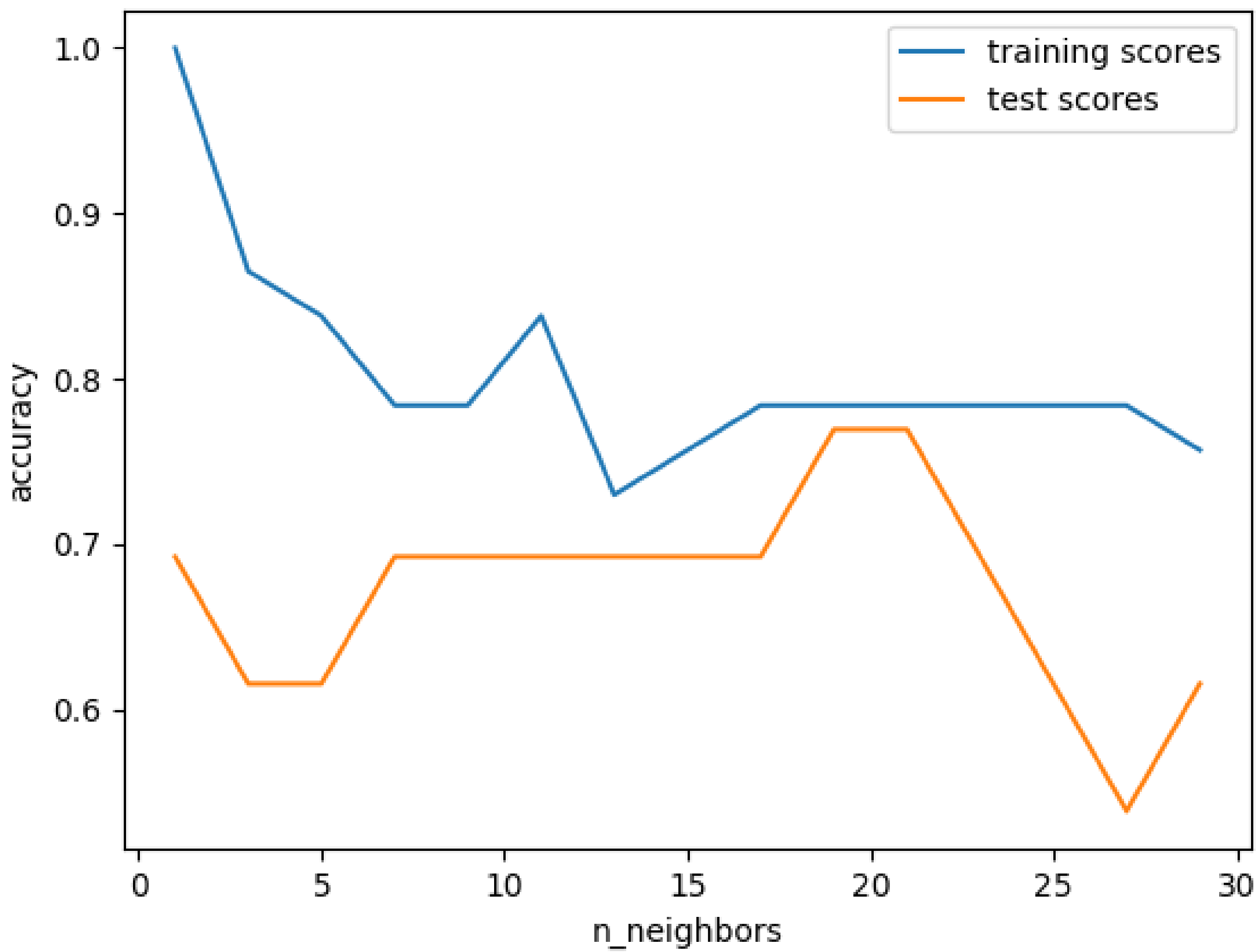
Nearest neighbors



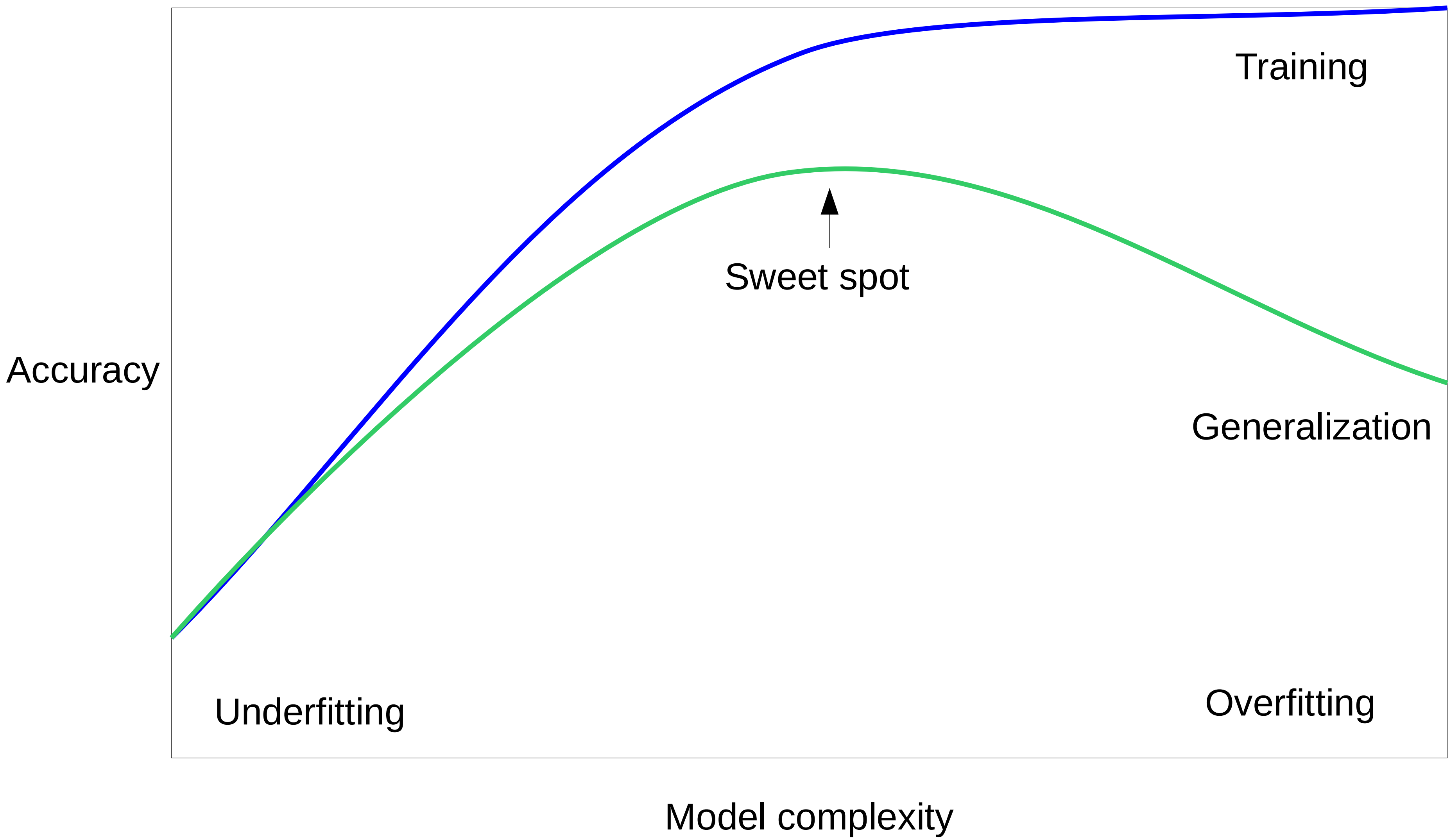
Influence of n_neighbors



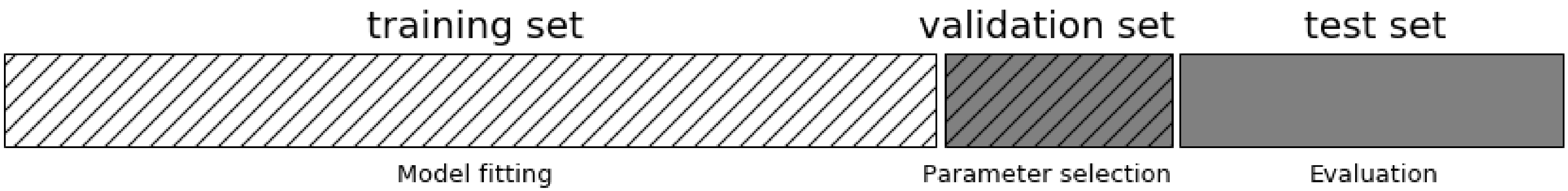
Model Complexity



Overfitting and Underfitting



Three-fold split



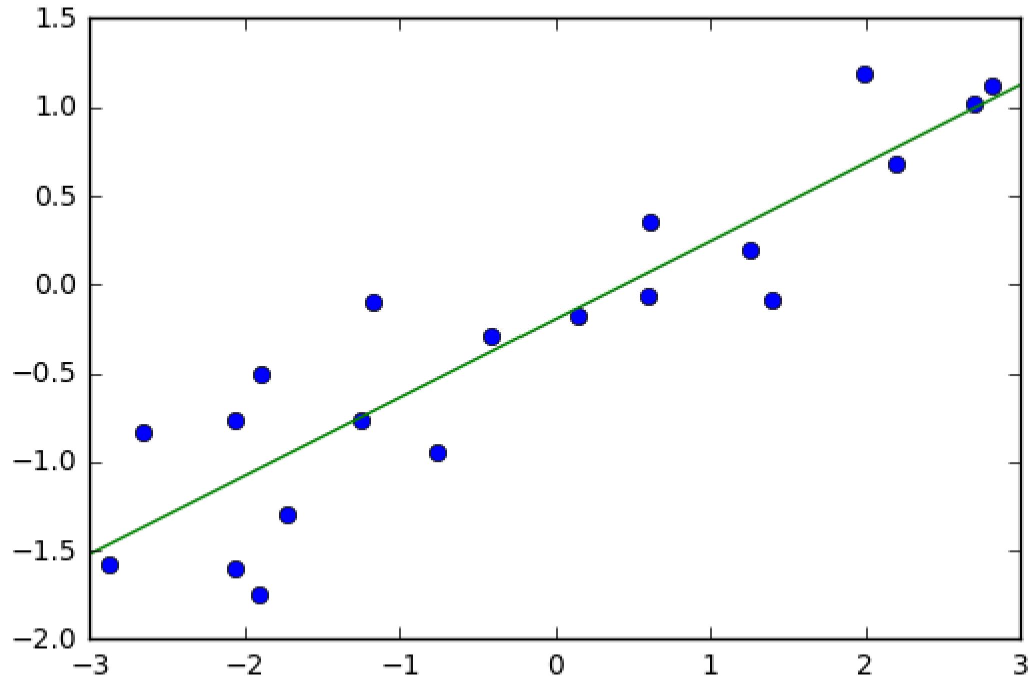
pro: fast, simple
con: high variance, bad use of data.

```python
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: {}".format(best_n_neighbors))

knn = KNeighborsClassifier(n_neighbors=best_n_neighbors)
knn.fit(X_trainval, y_trainval)
print("test-set score: {:.3f}".format(knn.score(X_test, y_test)))
```

```
best validation score: 0.972
best n_neighbors: 3
test-set score: 0.965
```

Linear Models for Regression

Linear Models for Regression



$$\hat{y} = w^T \mathbf{x} + b = \sum_{i=1}^{p} w_i x_i + b$$

Linear Regression & Ridge Regression

$$\hat{y} = w^T \mathbf{x} + b = \sum_{i=1}^{p} w_i x_i + b$$

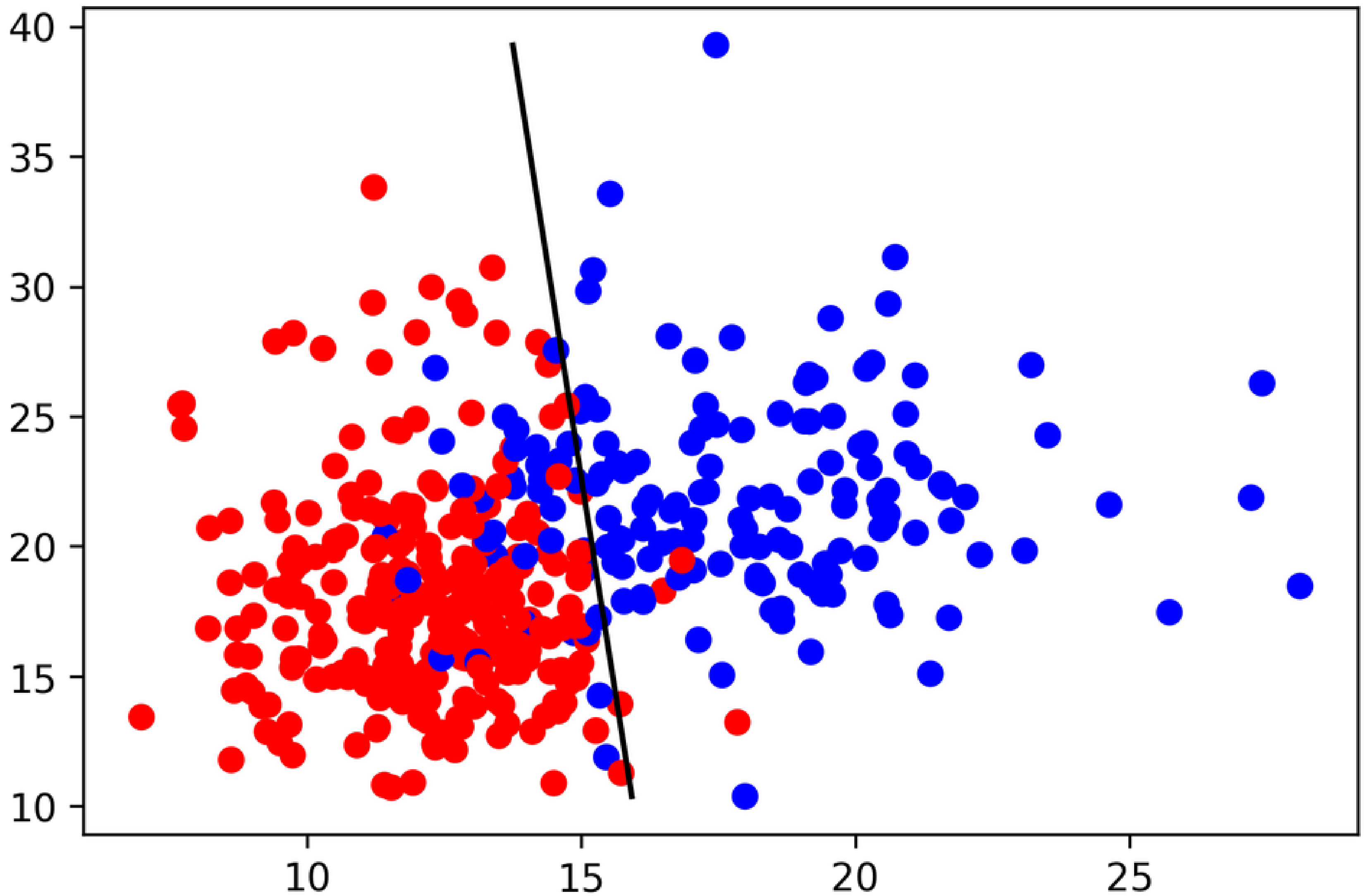
$$\min_{w \in \mathbb{R}^p} \sum_{i=1}^{p} ||w^T \mathbf{x}_i - y_i||^2$$

Unique solution if $\mathbf{X} = (\mathbf{x}_1, ...\mathbf{x}_n)^T$ has full column rank.

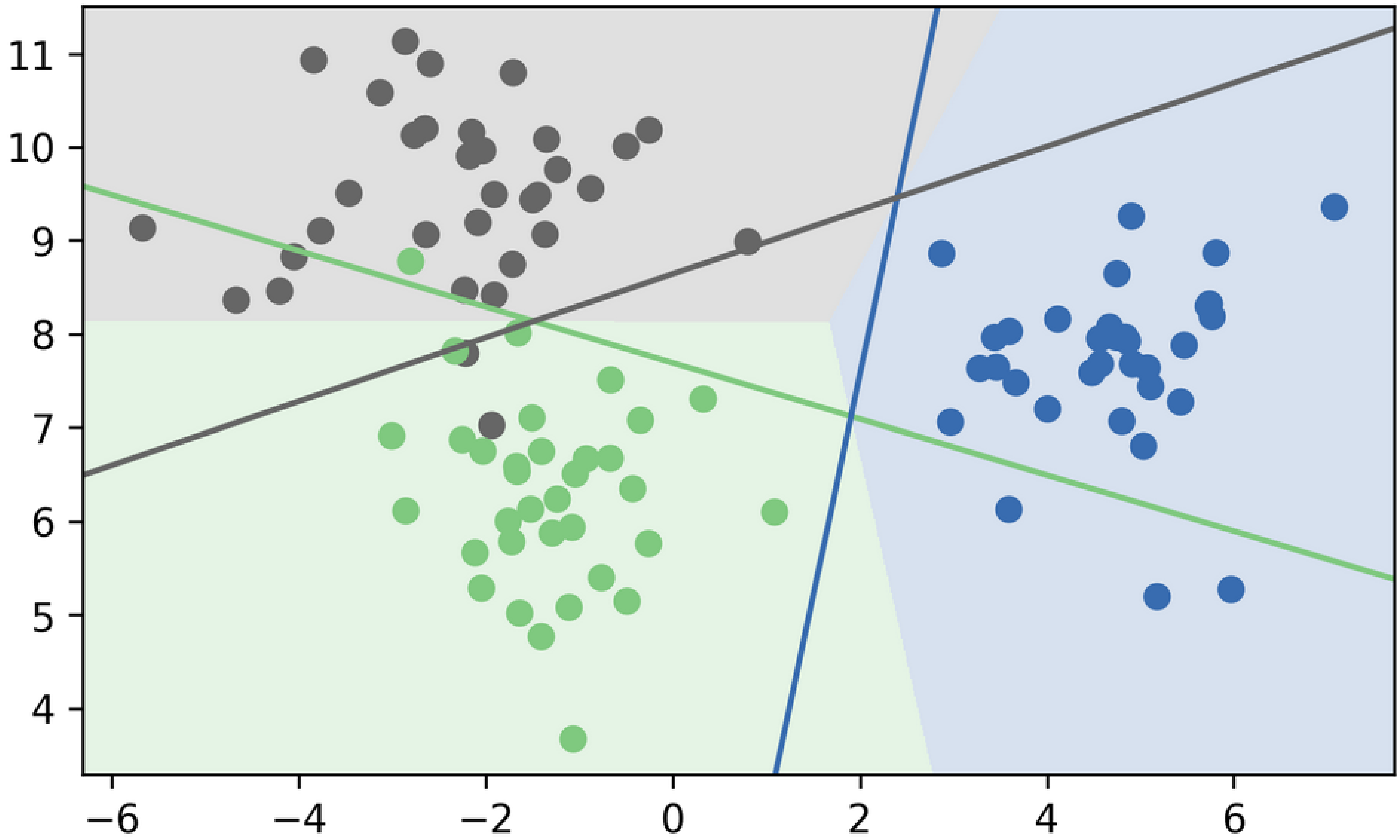
$$\min_{w \in \mathbb{R}^p} \sum_{i=1}^{n} ||w^T x_i - y_i||^2 + \alpha ||w||^2$$

Always has a unique solution.
Tuning parameter alpha.

Linear Models for Classification



$$\hat{y} = \text{sign}(w^T \mathbf{x} + b) = \text{sign}(\sum_i w_i x_i + b)$$



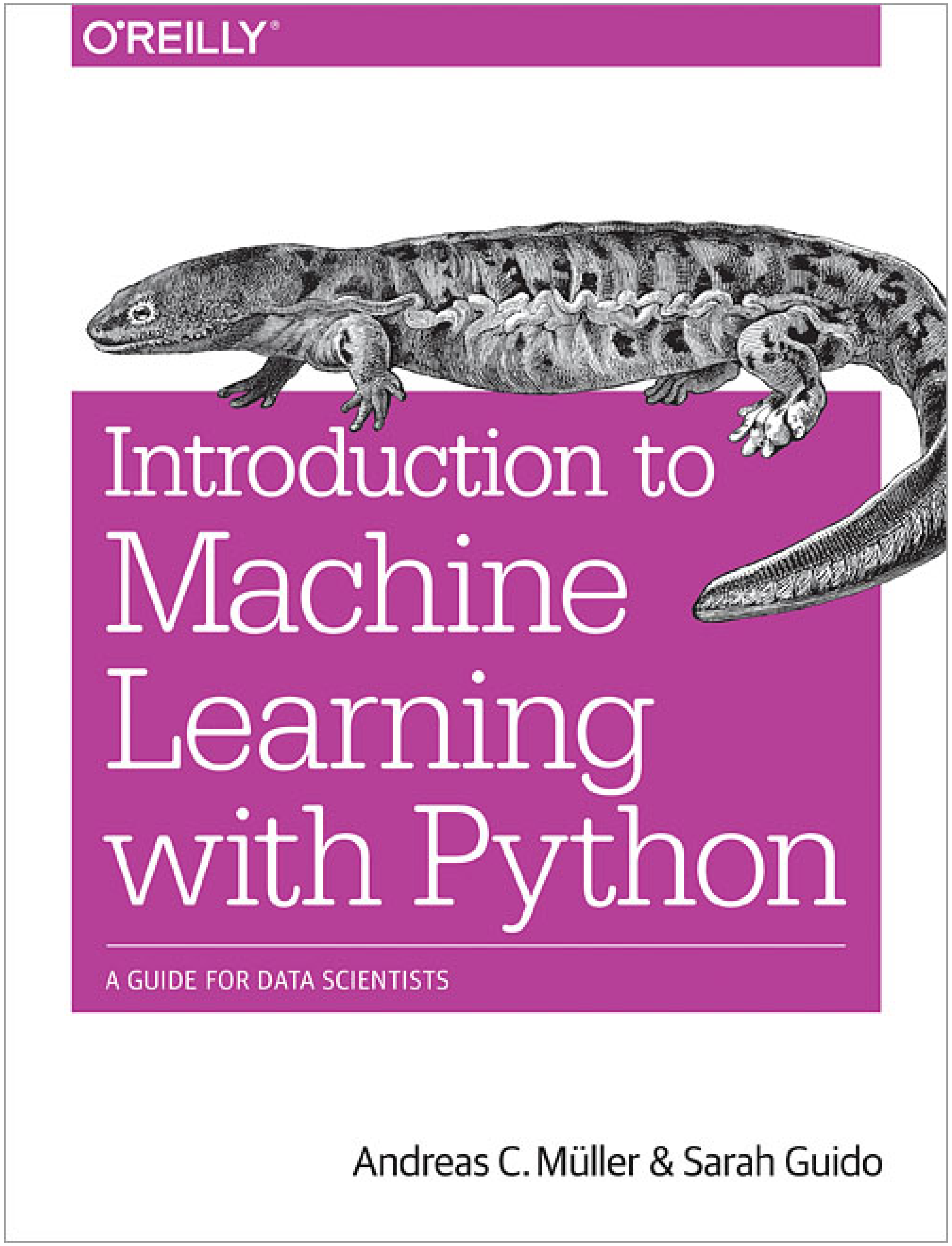
$$\hat{y} = \arg\max_{i \in Y} \mathbf{w}_i \mathbf{x} + b_i$$

IPython Notebook:
Part 4 – Linear Models for Regression

Basic API

`estimator.fit(X, [y])`

`estimator.predict`	**`estimator.transform`**
Classification	Preprocessing
Regression	Dimensionality reduction
Clustering	Feature selection
	Feature extraction


O'REILLY
Introduction to Machine Learning with Python
A GUIDE FOR DATA SCIENTISTS
Andreas C. Müller & Sarah Guido

Thank you for your attention.

@amuellerml

@amueller

importamueller@gmail.com

http://amueller.github.io