Advanced Data Analysis

DATA 71200

Class 6

Weekly Schedule

7-Jun	Inspecting Data
8-Jun	Representing Data
9-Jun	**Evaluation Methods**
10-Jun	Async

Splitting the Data

- **Cross-validation**
  - Splitting the data into folds and iteratively switching up which fold is used for testing (while the remainder are used for training)
  - The data can be split up in a variety of ways

Splitting the Data

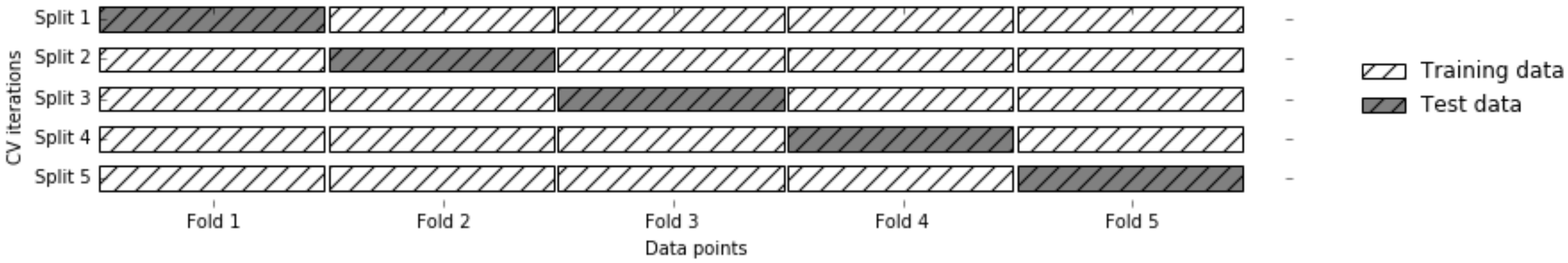


Figure 5-1. Data splitting in five-fold cross-validation

This will create issues if the data is ordered by class since some classes will appear disproportionately (or not at all) in either the testing or training sets

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Splitting the Data

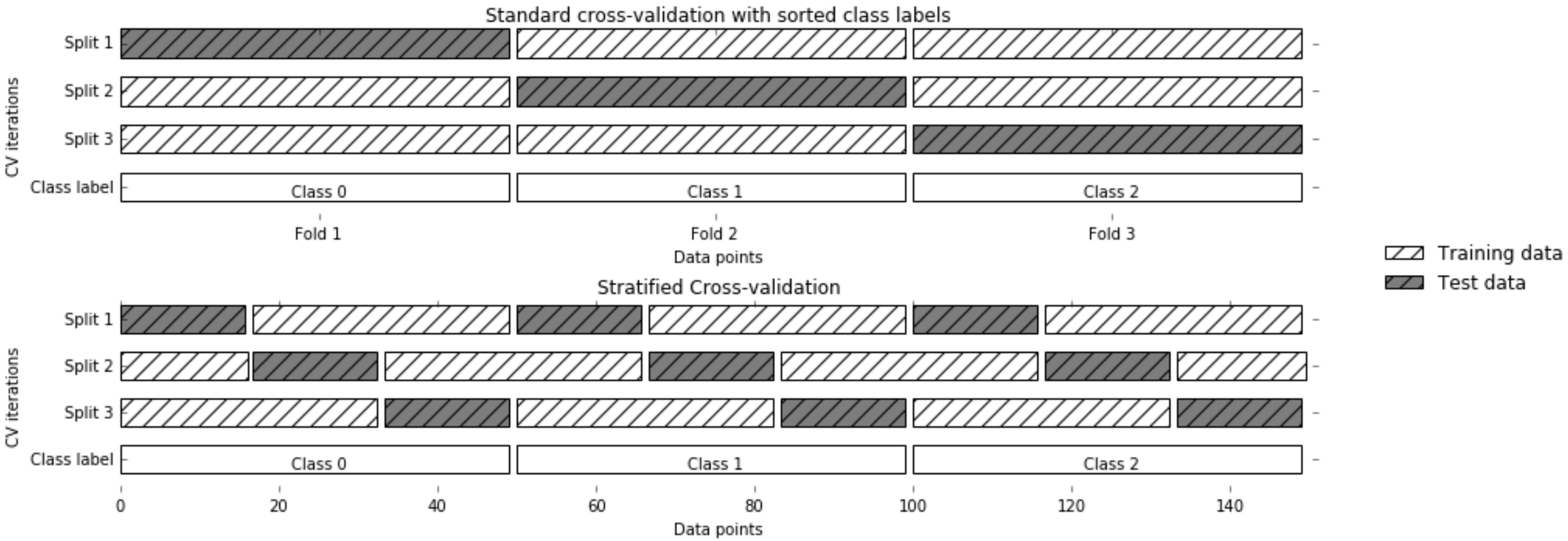


Figure 5-2. Comparison of standard cross-validation and stratified cross-validation when the data is ordered by class label

Stratified Cross-validation explicitly takes data from each class in order to count the issue of over-/under-sampling that can arise in standard cross-validation

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Splitting the Data

Small Data Sets

Leave-one out cross validation can be used where each fold is a single sample

Large Data Sets

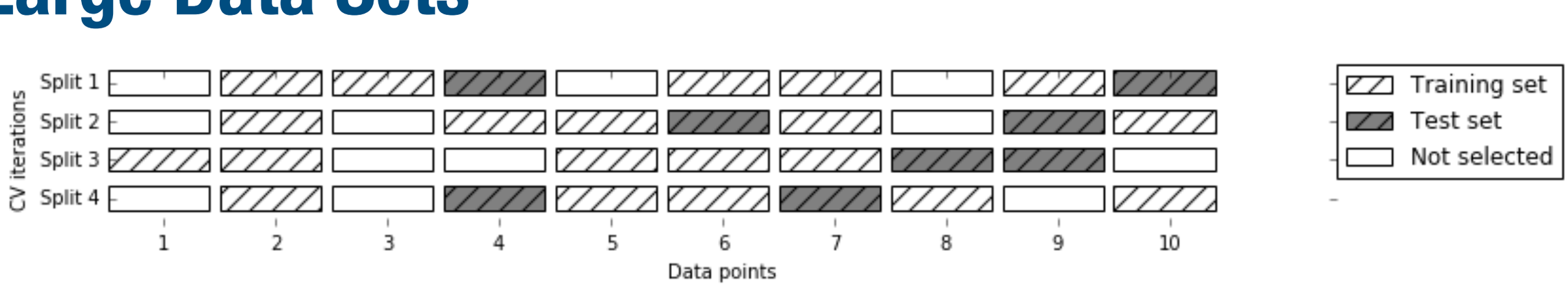


Figure 5-3. ShuffleSplit with 10 points, train_size=5, test_size=2, and n_splits=4

Shuffle-Split provides a system way to sample from your dataset without using all of the data

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Splitting the Data

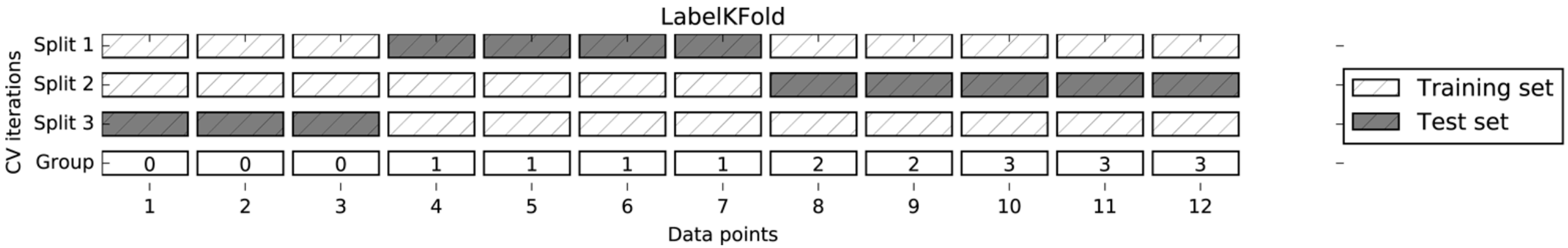


Figure 5-4. Label-dependent splitting with GroupKFold

When groups are highly related (e.g., individual speakers in speech recognition) you may not want to train and test on a single group

GroupKFold allows you select training and testing sets that either include or exclude an entire group

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Splitting the Data

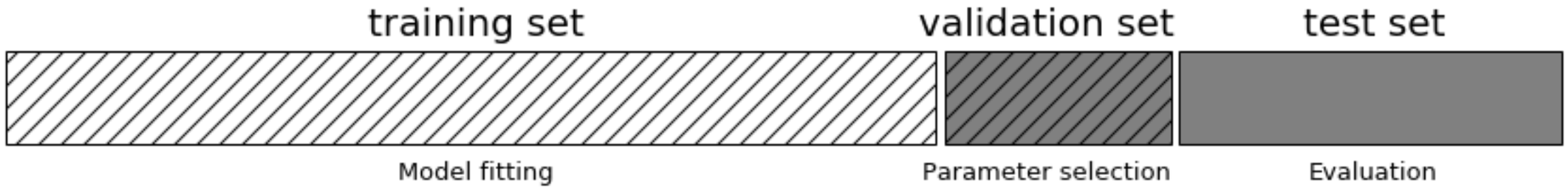


Figure 5-5. A threefold split of data into training set, validation set, and test set

If you are evaluating different models or tuning parameters (e.g., using Grid Search), you will want to have a reserved test set and then split the remaining data into training and validation sets

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Grid Search

Grid search can be performed simply (one split) or iteratively (cross-validation) - in the latter the optimal parameter values need to be summarized

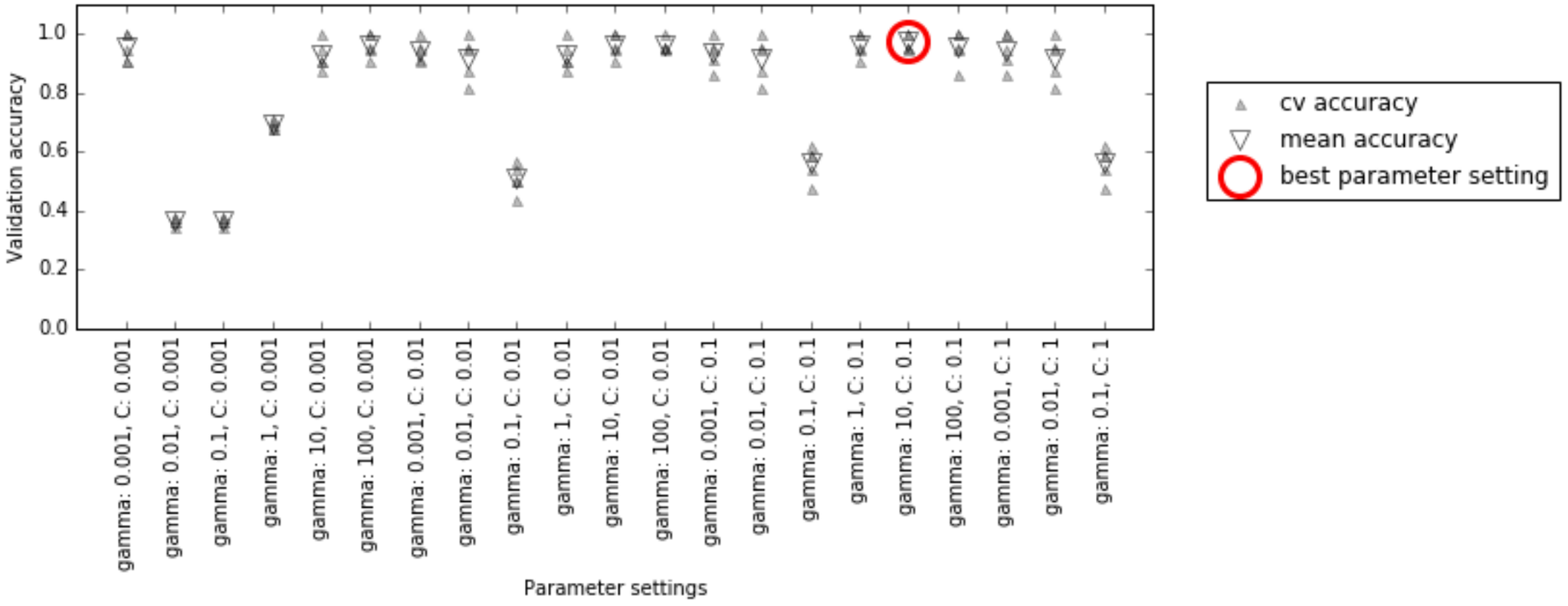


Figure 5-6. Results of grid search with cross-validation

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Grid Search

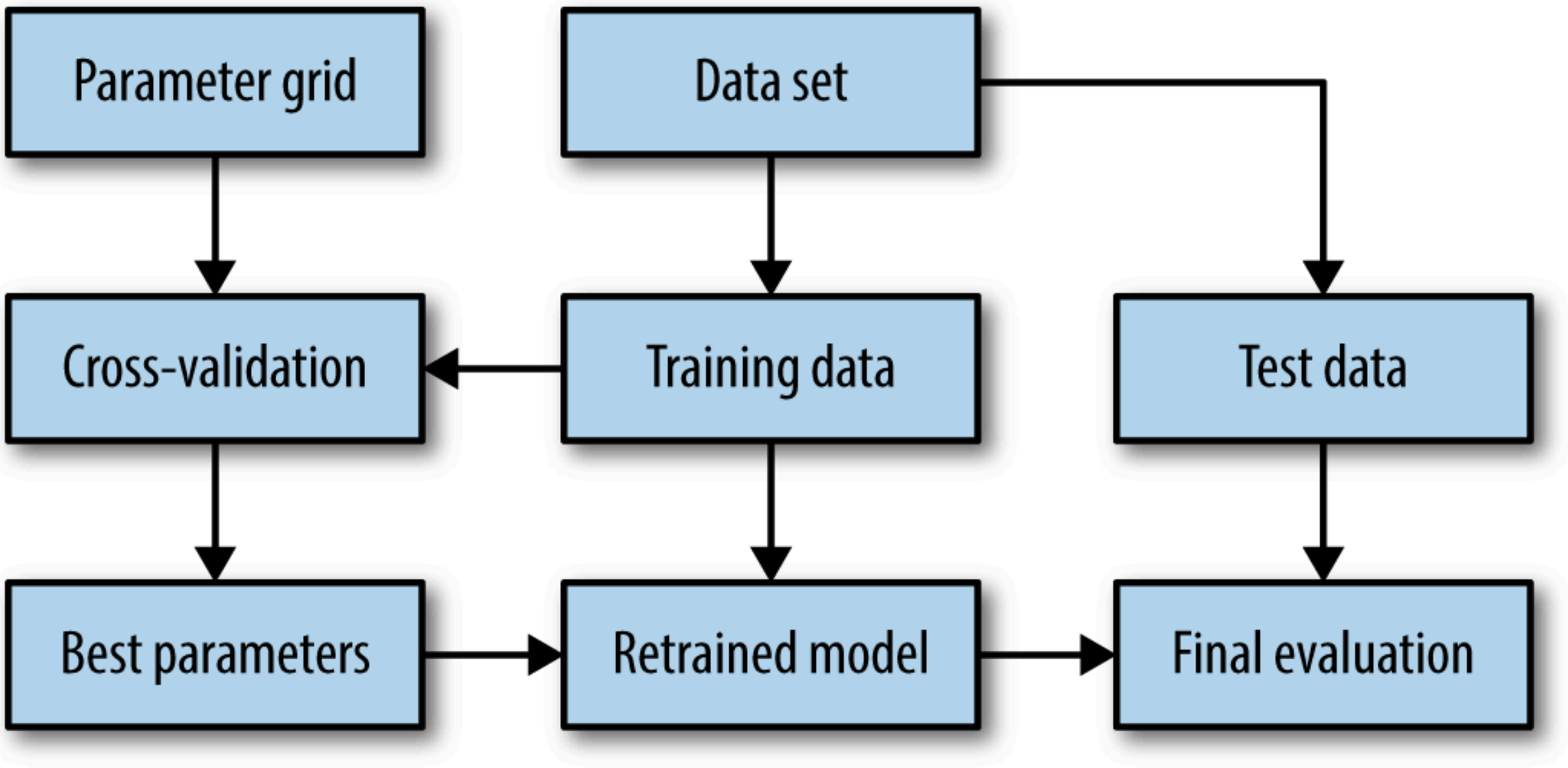


Figure 5-7. Overview of the process of parameter selection and model evaluation with GridSearchCV

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Grid Search

It is important than the parameter range reached is appropriate

Heat maps can help understand whether the range is appropriate

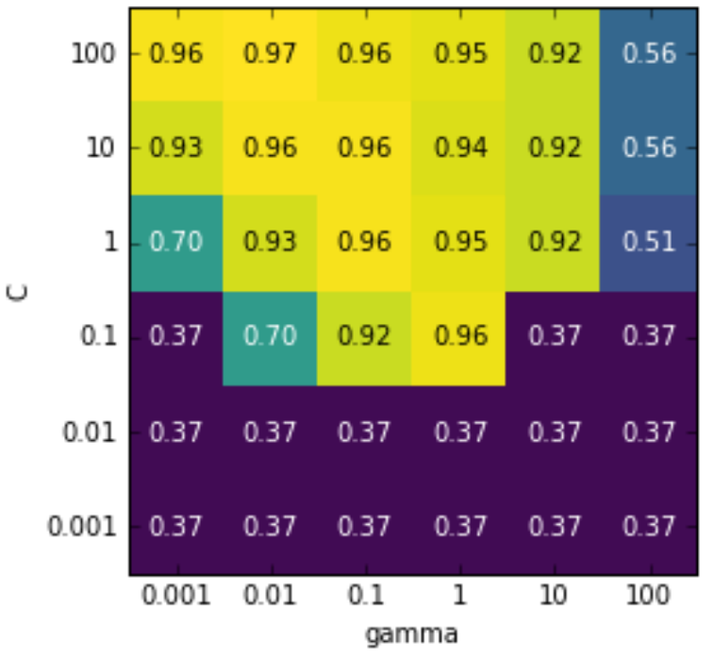


Figure 5-8. Heat map of mean cross-validation score as a function of C and gamma

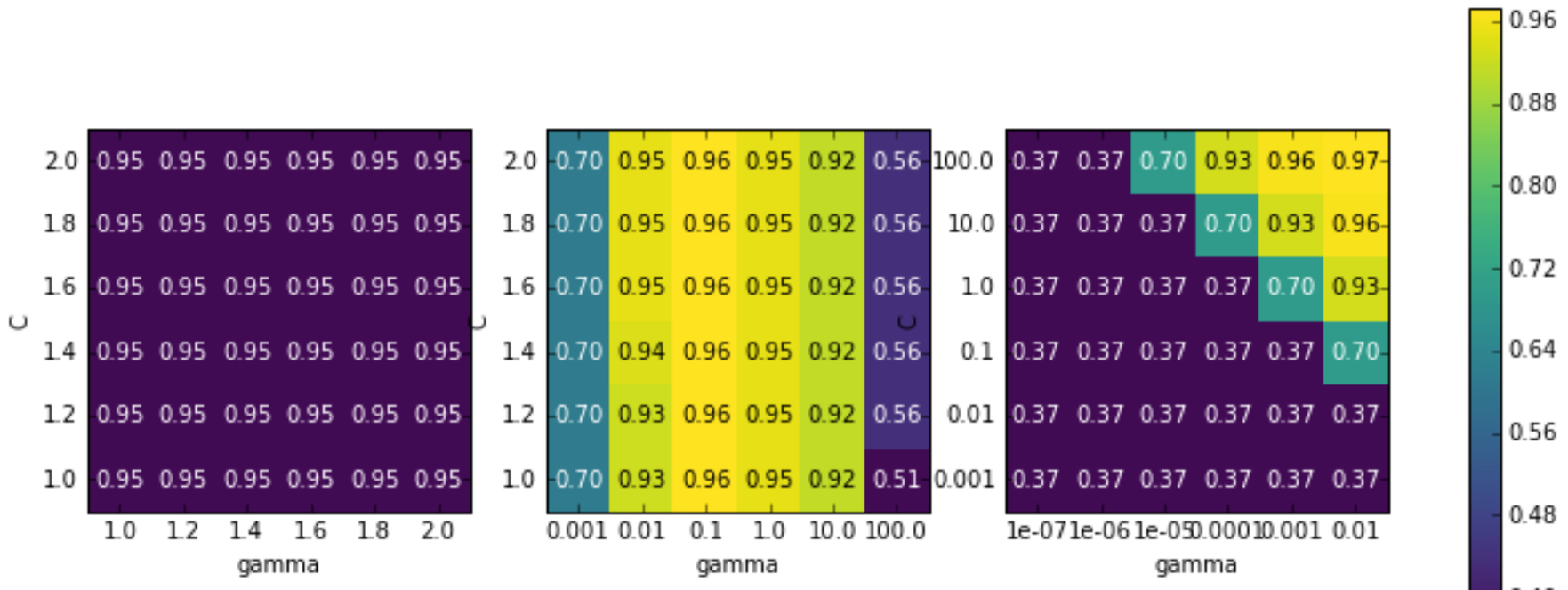


Figure 5-9. Heat map visualizations of misspecified search grids

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

TP - true positive

TN - true negative

FP - false positive

FN - false negative

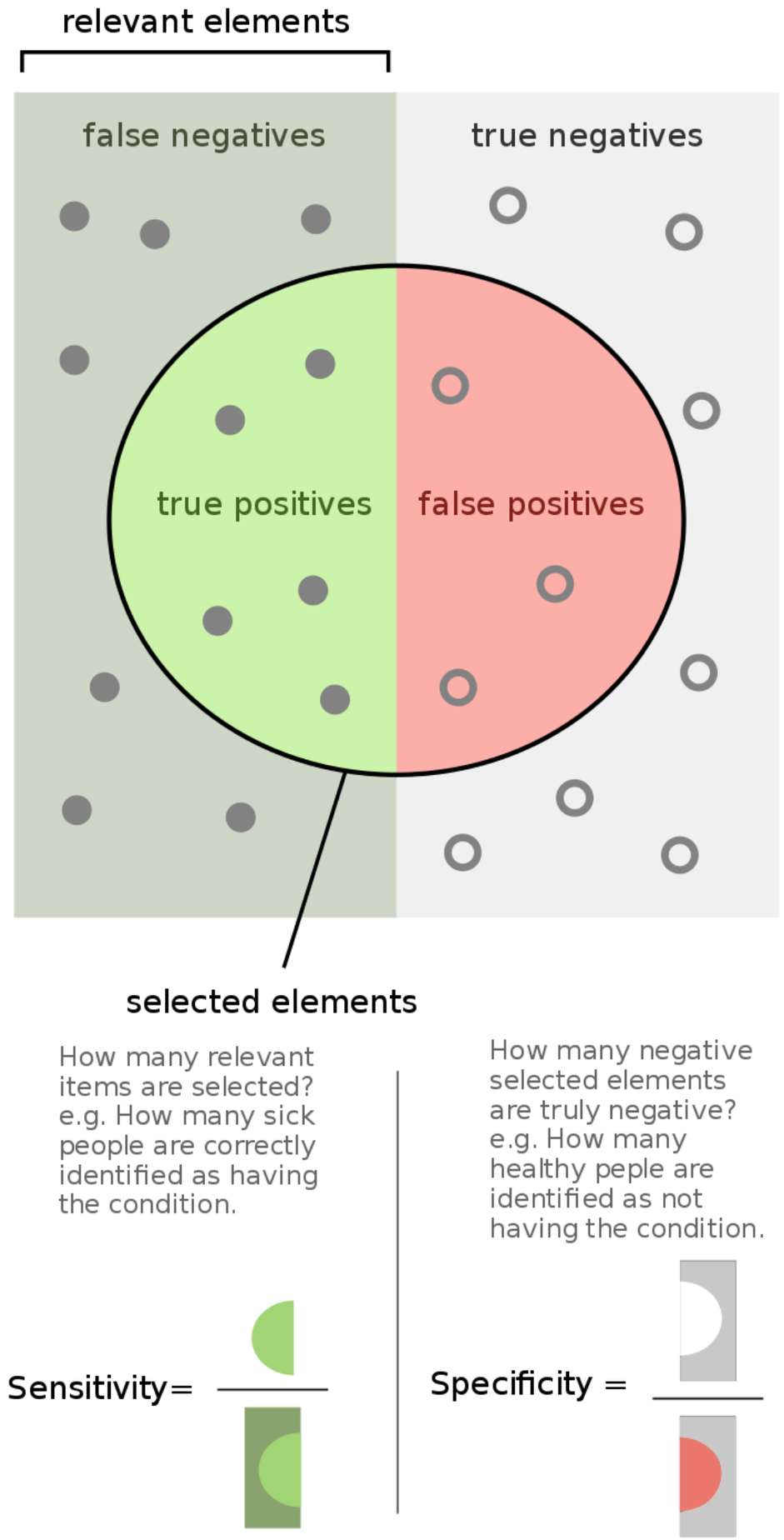
How many predictions were correct out of all the predictions made

Kinds of Errors

Type I: false positive

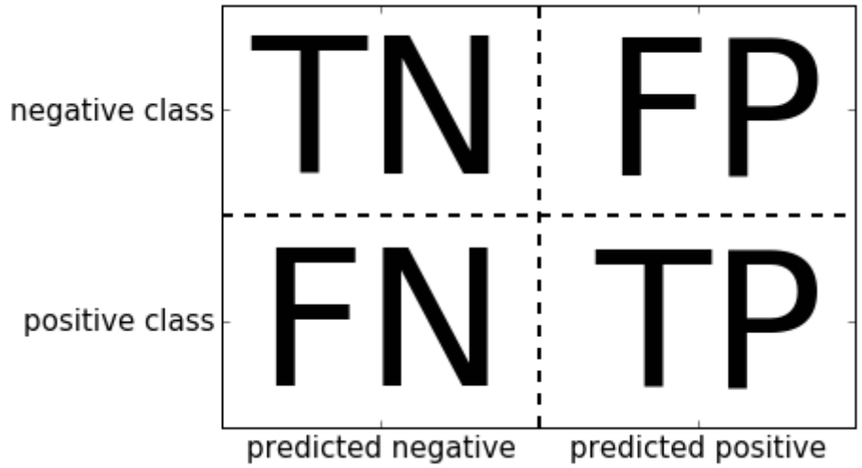
Type II: false negative

Balance of the classes in the dataset will influence how you interpret performance errors



https://commons.wikimedia.org/wiki/File:Precisionrecall.svg

Confusion Matrices



		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error
	Predicted condition negative	**False negative**, Type II error	**True negative**

Confusion matrices visualize not only how many correct estimates a model made but also what and how it mis-classified

https://en.wikipedia.org/wiki/Precision_and_recall

Precision

$$\text{Precision} = \frac{tp}{tp + fp}$$

tp - true positive

fp - false positive

Positive predictive value

Number of the positive predicted values that are actually positive

Recall

$$\text{Recall} = \frac{tp}{tp + fn}$$

tp - true positive

fn - false negative

Sensitivity

Proportion of the actually positives identified

F-1 Score (F-score, F-measure)

$$F_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}.$$

Harmonic mean of precision and recall

Summary measurement of the classifier's accuracy

Putting it All Together

		True condition			
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$	F_1 score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

https://en.wikipedia.org/wiki/Precision_and_recall

In-Class Activity

Calculate each of these metrics for the confusion matrix on the right

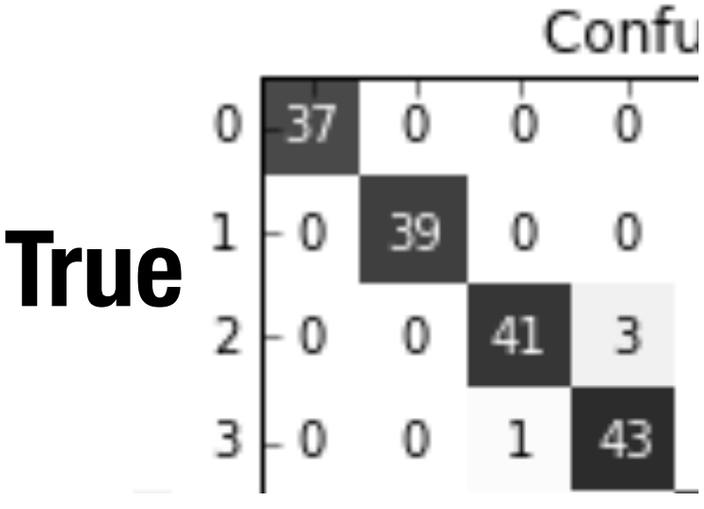


Figure 5-18. Confusion matrix for the 10-digit classification task

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Precision Recall Curves

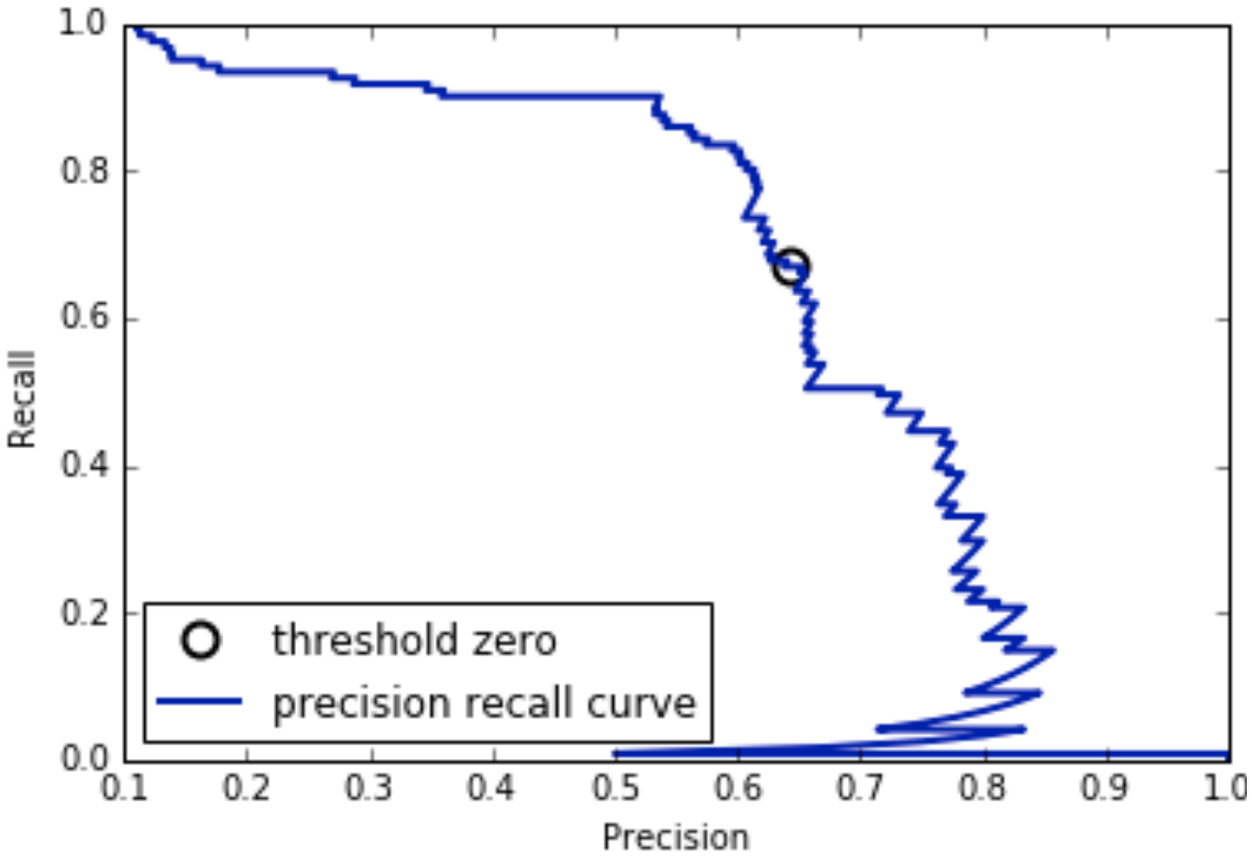


Figure 5-13. Precision recall curve for SVC(gamma=0.05)

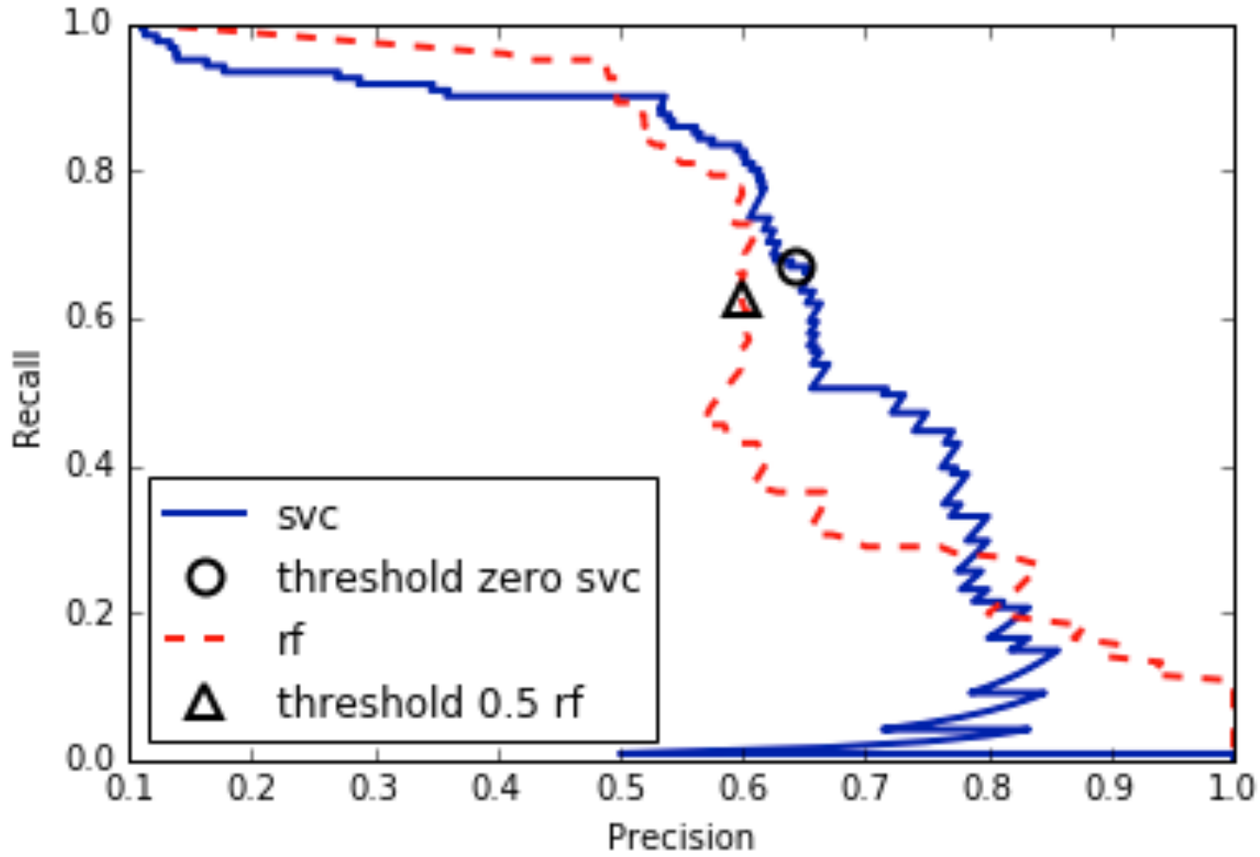


Figure 5-14. Comparing precision recall curves of SVM and random forest

Visualizes trade-off between positive predictive rate and true positive rate (sensitivity)

Area under curve summarizes this information

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

ROC Curves

$$FPR = \frac{FP}{FP + TN}$$

TP - true positive
TN - true negative
FP - false positive

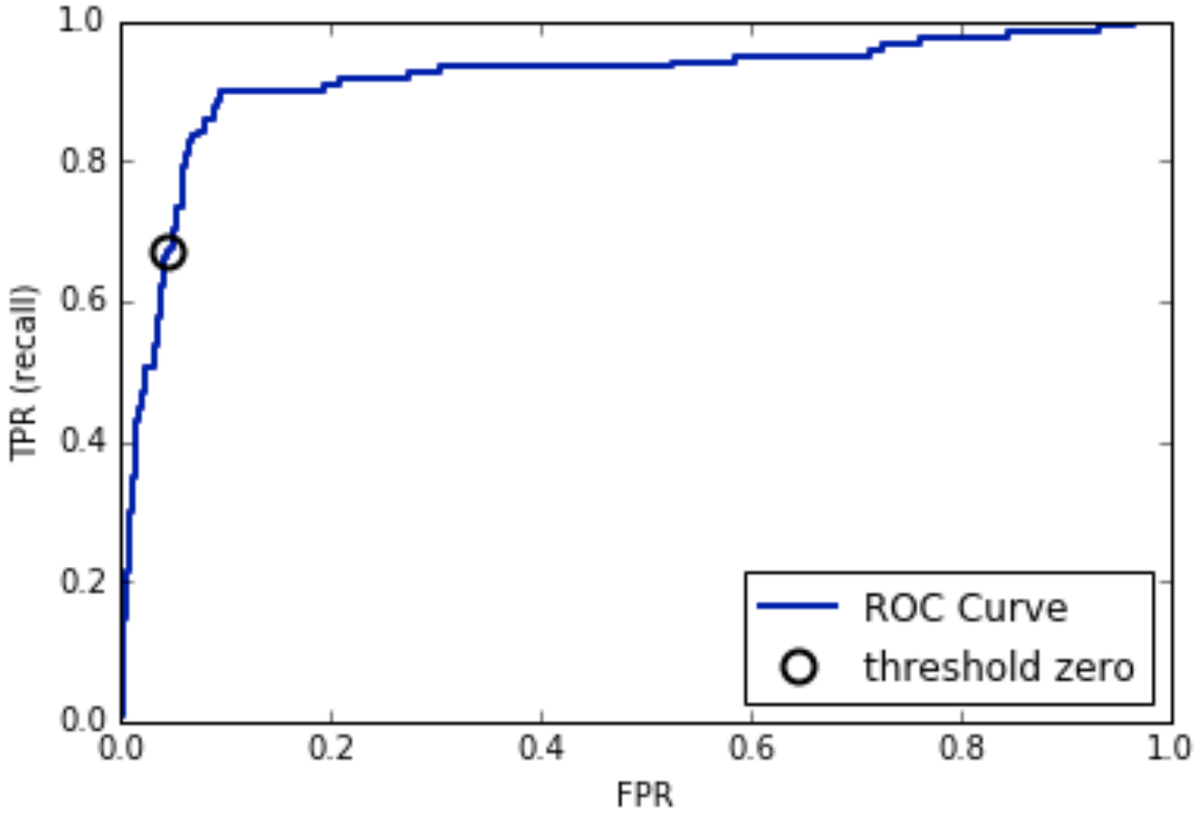


Figure 5-15. ROC curve for SVM

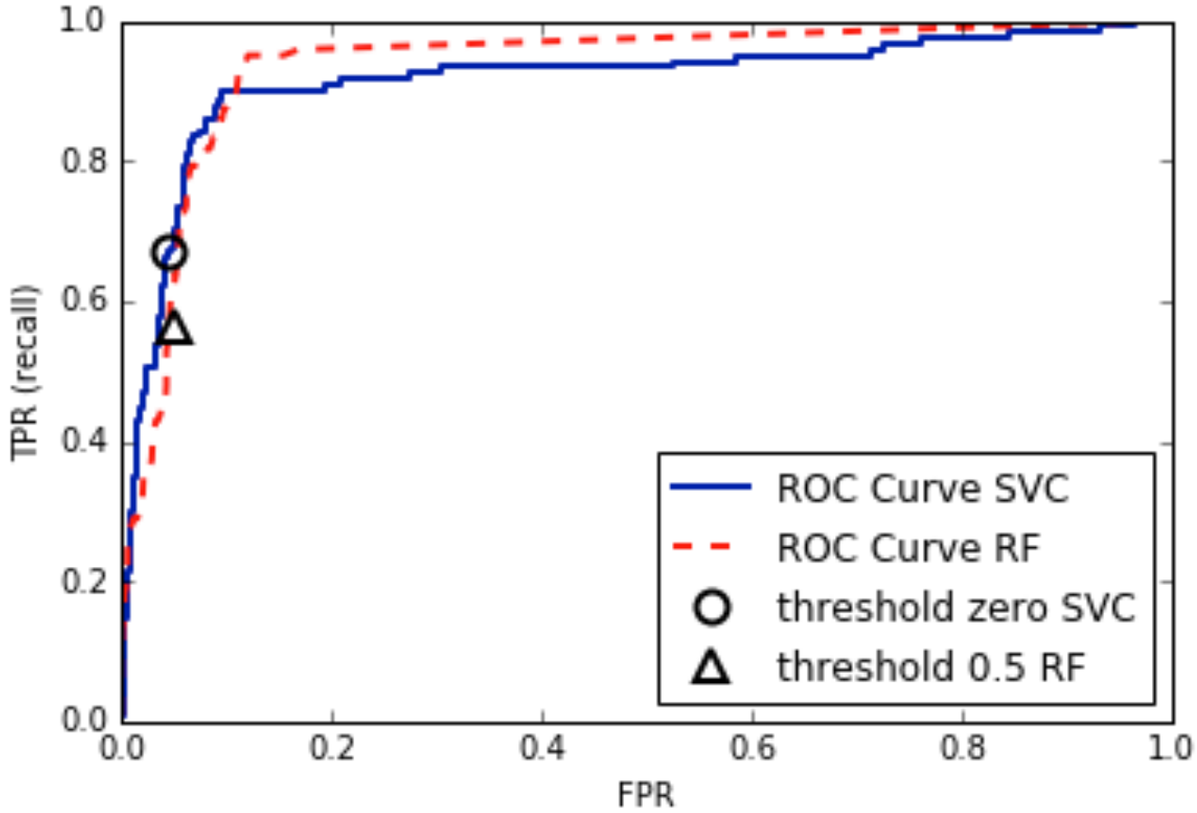


Figure 5-16. Comparing ROC curves for SVM and random forest

Receiver operating characteristics

Visualizes relationship between true positive rate (sensitivity) and false positive rate - the closer to the top right the better

Area under curve also useful to calculate here

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Decision Thresholds

How much uncertainty the classifier will permit to make a classification

Can be adjusted to change the prioritization between precision and recall

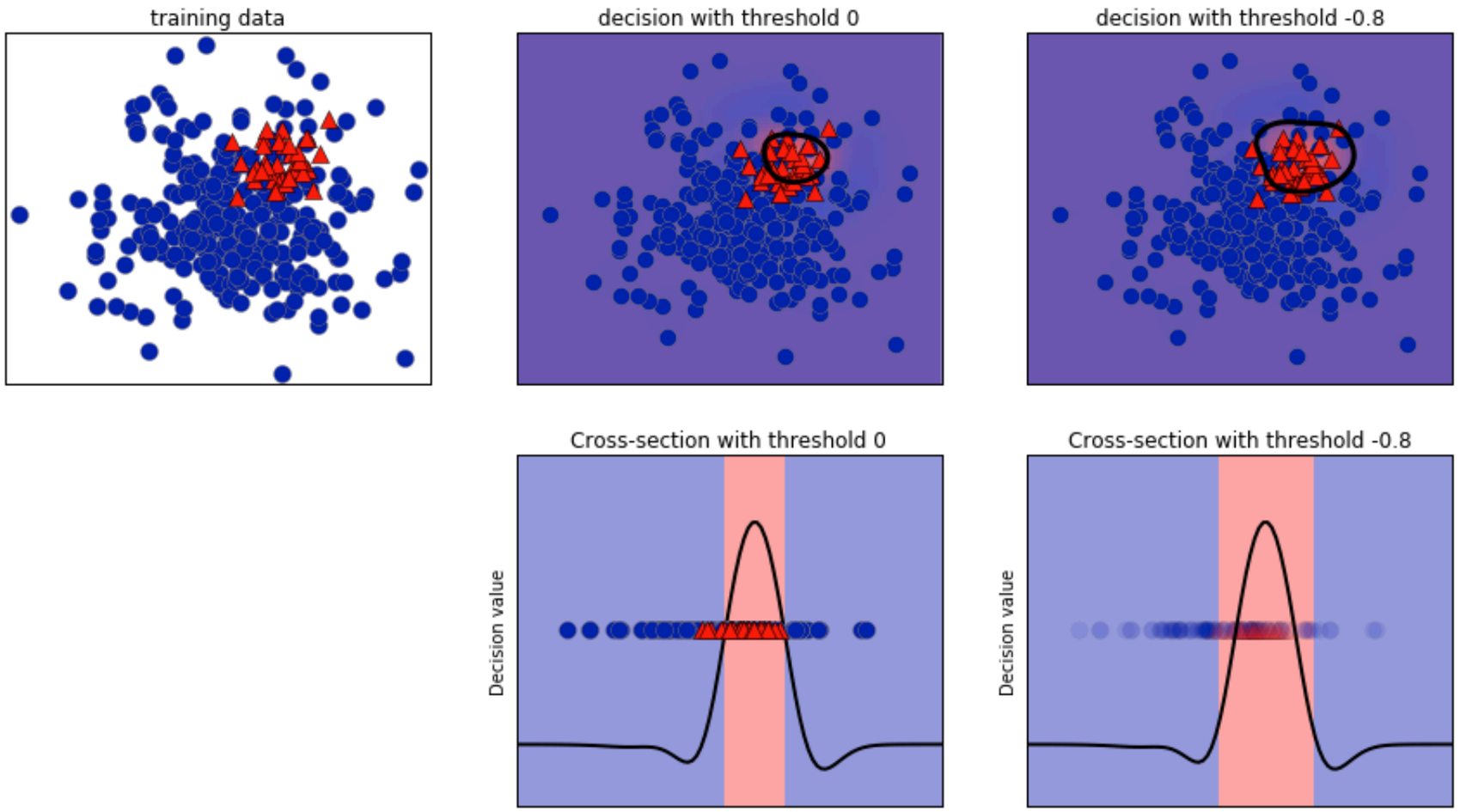


Figure 5-12. Heatmap of the decision function and the impact of changing the decision threshold

Guido, Sarah and Andreas C. Muller. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Regression Performance Measures

Equation 2-1. Root Mean Square Error (RMSE)

$$\text{RMSE}(\mathbf{X}, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right)^2}$$

m

- m is the number of instances in the dataset you are measuring the RMSE on.
  - For example, if you are evaluating the RMSE on a validation set of 2,000 districts, then m = 2,000.

$\mathbf{x}^{(i)}$ $y^{(i)}$

- $\mathbf{x}^{(i)}$ is a vector of all the feature values (excluding the label) of the i^{th} instance in the dataset, and $y^{(i)}$ is its label (the desired output value for that instance).

Géron, Aurélien. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly Media, Inc.

Regression Performance Measures

$\mathbf{X}^{(i)}$ $y^{(i)}$ — For example, if the first district in the dataset is located at longitude –118.29°, latitude 33.91°, and it has 1,416 inhabitants with a median income of $38,372, and the median house value is $156,400 (ignoring the other features for now), then:

$$\mathbf{x}^{(1)} = \begin{pmatrix} -118.29 \\ 33.91 \\ 1,416 \\ 38,372 \end{pmatrix}$$

and:

$$y^{(1)} = 156,400$$

Géron, Aurélien. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly Media, Inc.

Regression Performance Measures

X

- **X** is a matrix containing all the feature values (excluding labels) of all instances in the dataset. There is one row per instance and the i^{th} row is equal to the transpose of $\mathbf{x}^{(i)}$, noted $(\mathbf{x}^{(i)})^T$.[6]
  - For example, if the first district is as just described, then the matrix **X** looks like this:

$$\mathbf{X} = \begin{pmatrix} \left(\mathbf{x}^{(1)}\right)^T \\ \left(\mathbf{x}^{(2)}\right)^T \\ \vdots \\ \left(\mathbf{x}^{(1999)}\right)^T \\ \left(\mathbf{x}^{(2000)}\right)^T \end{pmatrix} = \begin{pmatrix} -118.29 & 33.91 & 1,416 & 38,372 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Géron, Aurélien. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly Media, Inc.

Regression Performance Measures

h

- h is your system's prediction function, also called a *hypothesis*. When your system is given an instance's feature vector $\mathbf{x}^{(i)}$, it outputs a predicted value $\hat{y}^{(i)} = h(\mathbf{x}^{(i)})$ for that instance ($\hat{y}$ is pronounced "y-hat").
  - For example, if your system predicts that the median housing price in the first district is \$158,400, then $\hat{y}^{(1)} = h(\mathbf{x}^{(1)}) = 158{,}400$. The prediction error for this district is $\hat{y}^{(1)} - y^{(1)} = 2{,}000$.

$\text{RMSE}(\mathbf{X},h)$

- RMSE($\mathbf{X}$,h) is the cost function measured on the set of examples using your hypothesis h.

- Computing the root of a sum of squares (RMSE) corresponds to the *Euclidian norm*: it is the notion of distance you are familiar with. It is also called the ℓ_2 *norm*, noted $\| \cdot \|_2$ (or just $\| \cdot \|$).

Géron, Aurélien. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly Media, Inc.

Regression Performance Measures

Equation 2-2. Mean Absolute Error

$$\mathrm{MAE}(\mathbf{X}, h) = \frac{1}{m}\sum_{i=1}^{m}\left|h\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right|$$

m

- m is the number of instances in the dataset you are measuring the RMSE on.

$\mathbf{x}^{(i)}$ $y^{(i)}$

- $\mathbf{x}^{(i)}$ is a vector of all the feature values (excluding the label) of the i^{th} instance in the dataset, and $y^{(i)}$ is its label (the desired output value for that instance).

- Computing the sum of absolutes (MAE) corresponds to the ℓ_1 *norm*, noted $\| \cdot \|_1$. It is sometimes called the *Manhattan norm* because it measures the distance between two points in a city if you can only travel along orthogonal city blocks.

Géron, Aurélien. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly Media, Inc.

Regression Performance Measures

RMSE is more sensitive to outliers than the MAE

- when outliers are exponentially rare (like in a bell-shaped curve), the RMSE performs very well and is generally preferred

Géron, Aurélien. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly Media, Inc.

Assignments for next week

- **DataCamp**
  - Pre-processing for Machine Learning in Python course
  - Model Validation in Python course

- **Project 1**

- **Videos to watch (password: 71200)**
  - K-nearest Neighbors: https://vimeo.com/400660692
  - Linear Models: https://vimeo.com/403004687