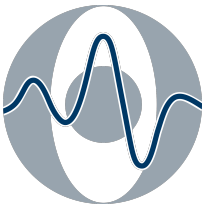
Exercises for
Pattern Analysis
Amir Davari, Dalia Rodriguez Salas, Johannes Bopp, Lina Felsner
Assignment 2, published on May 20, 2019



General Information:

Lecture:	Mon 16:15h – 17:45h and Wed 10:15h – 11:45h (H16)
Exercises:	Tue 14:00 – 15:00, 15:00 – 16:00 (02.151-113) and
	Thu 16:00 – 18:00 (0.01-142, 00.156-113, 02.151b-113)
Certificate:	Oral exam at the end of the semester
Contact:	amir.davari@fau.de & dalia.rodriguez@fau.de

Transforming an Unsupervised Task to a Supervised Task

Exercise 1

Preliminary Remarks: Density Estimation as a Regression Task

There are unsupervised models that operate on the empirical probability density distribution of the data, and there are supervised models to perform classification and regression. Supervised methods can be further split up into a generative probability model for the joint density $p(\mathbf{x}, \mathbf{y})$ of features $\mathbf{x}$ and labels $\mathbf{y}$, or a discriminative model that directly target at $p(\mathbf{y}|\mathbf{x})$.

There is a good reason to have specialized methods for different learning tasks: task-specific performance. However, from a theoretical perspective, it is quite interesting to note how closely related the seemingly disparate tasks of unsupervised learning, regression, and classification are.

This exercise is loosely inspired by Sec. 14.2.4 in "The Elements of Statistical Learning" by Hastie, Tibshirani, Friedman: let us convert the density estimation task into a regression task.

The key idea is to define a second density that models the background. Assign high values to the actual density samples (e.g., the ones from our raccoon), and low values to the background density. Then, the regression task is to fit one function that represents both distributions, with high values in regions where our actual samples dominate, and with low values in regions where the background dominates.

Technical Approach

Reuse the raccoon sampler from the previous exercise to create the actual density samples. We also need a background distribution: what might be a good distribution for representing the background? Hastie, Tibshirani, Friedman suggest that unless we have a better idea, we can certainly choose a uniform distribution, so let us work with a uniform distribution. Note that from a theoretical perspective, you will need at least as many samples in the background distribution as in the actual distribution.

Use a random forest to fit the regression model. That way, we can also earn some practical experience with parameterization of such a forest. Random forests are implemented in `scikit.learn.ensemble`, there are also plenty of tutorials where you can learn how to technically wrap our 2-D sample space and labels.

You can check the performance of your regressor visually, by making a regressor prediction for each pixel of the raccoon image: does the output look like a raccoon? A more quantitative approach that also allows to tune the forest parameters would be to reuse the cross-validation from the previous exercise (but this is no requirement, since the computation likely takes too for a "live" demo).

Tackling the Annoying Details

Allow me to direct your attention to a very common problem in the data science business: in order to "get things done" people use powerful frameworks such as `scikit.learn` with a lot of functionality. This is the right decision. However, oftentimes it is not clear what *exactly* a class does, but you desperately need a good understanding of what your algorithm does, otherwise you have no chance to identify limitations of your method.

We would like to avoid this problem. Browse through the whole list of classes in the reference documentation of `scikit.learn.ensemble`. Note that the `RandomForestRegressor` is not the only tree-related algorithm in this list: we also have an `ExtraTreeRegressor`, an `IsolationForest`, and a `RandomTreeEmbedding`. Ignore the `RandomTreeEmbedding` and `IsolationForest` for now (unless you are curious). However, what exactly is the difference between a `RandomForestRegressor` and an `ExtraTreeRegressor`? Which of both is closer to the Criminisi/Shotton/Konokoglu forest from the lecture? Experiment with both of them with different parameters, and also different numbers of samples. The most important parameters are the number of training samples, the tree depth, and the number of trees in a forst. Which parameter settings provide satisfying results? By tendency, are the requirements for the amount of training data larger or smaller than for our Parzen window estimator?

Exercise 2 A Second Dataset

Use the function `sklearn.datasets.make_moons` to create a synthetic density of two interwoven half moons. Repeat your experiments with the `RandomForestRegressor` and the `ExtraTreeRegressor`. Are your observations consistent with your observations on the previous dataset?

Demonstration of your Results: please book a date on your exercise slot in studOn, i.e., for Tuesday June 4 or Thursday June 6!