Introduction to Computational Neuroscience
Practice IV: Data Analysis - Decoding

Ilya Kuzovkin, Raul Vicente

March 9, 2014

The main purpose of the field of *neural decoding* is to reconstruct stimuli that led to the neuronal response presented in the data. Usually we have a finite set of stimuli and we show them to the test subjects. And in the same time we record responses on the test subject's brain. The task is to create a *model*, which takes piece of recorded data as an input and *predicts* which stimulus is responsible for producing this piece of data.

In the previous session we attempted to manually create one such model: by looking at the firings of 72 neurons we identified the stimulus (orientation of the bar on the screen). Although such approach can be very fruitful, it is often impossible to embrace all of the data manually. The data can be too massive or too complex or it can be hard to find an intuition, which would explain features of the data.

And here the field of *machine learning* comes in. The main goal of machine learning can be summarized as providing an automatic way of finding dependencies between the data and the stimuli.

Before we continue let us go through vocabulary:

- **Dataset** is a structure which contains all the data we have.
- Dataset consist of **instances**. For example 4500ms of spiking data recorded while showing one stimulus is an instance.
- Instances consist of **features**. Those are parameters, which describe our data. For example average spiking rate is a feature, SPTH distribution is a feature, etc. Each instance has its own values of the features: for example one trial has spiking rate of 7 spikes/second, while another trial has spiking rate of 23 spikes/second.
- All features put together form a **feature vector**.
- One feature vector is a representation of one instance in the **feature space**. All instances live in there.
- Each instance belongs to a certain **class**. In previous dataset we had 12 different orientations, so each one represents different class in our dataset.
- The goal of a machine learning algorithm is to create a **model**, which can **classify** previously unseen data. For example we show one of the stimuli to the mouse, record new 4500ms of data, give this data to the model and the model tells us which stimulus was shown.
- The model is created from examples. Those are instances for which the class is known. Set of such examples is called **training set**, because we train our model on it.
- **Test set** is another set of instances, for which we also know the true class, but we do not give it to the model. We give the data (without the class) to the model and we ask the model to guess the class of the each instance.

- We can see how many instances from the test set model has identified correctly and the rate

$$\frac{\text{Number of correctly classified instances}}{\text{Total number of instances}}$$

is called **accuracy**[1] and it used to evaluate model's performance.

- **Principle Component Analysis** (PCA) is a method, which allows to replace existing features by principle components. The important property of the principle components is that the first component has the largest possible variance, the second one is orthogonal to the first one and again, has the second largest variance, and so on. As a result you usually can reduce the dimensionality of your feature space by using only N first components as they already describe all the variance in your data and you do not need the rest of them.

Exercise 1: Definitions (0.5pt)
Come up with an example from the field of neuroscience for the definitions we have above:

Q_1 : Start by saying what is your dataset (just think anything up).

Q_2 : What are the instances of this dataset?

Q_3 : What are the features of your data?

Q_4 : What is the dimensionality (size) of your feature vector?

Q_5 : How many classes your dataset has and what they are?

Q_6 : Why do you need to separate training and test datasets?

Q_7 : Let us say that you have 2 classes and your model predicts the class correctly in 50% of the cases. How useful this model is?

Q_8 : What does it mean if your model's accuracy on the training set is high, but on the test set it is very low?

Exercise 2: Where is the rat? (1pt)

In this exercise we will take a small part of real dataset HC-3[2]. The rat is running in a simple maze in which different visual markers are set up. The spiking activity of 311 neurons is recorded while rat is running in there. Among other things the (x, y) position of the rat is recorded. In this exercise we will attempt to estimate position of the rat from the neuron activity data.

The original dataset is quite massive, so we will use only the data from one shank (see Figure 1) and this data will be preprocessed by the authors of the dataset. In `data` folder you can find two files: `ec012ec.187.fet.1` and `ec012ec.187.whl`. The first file is a matrix with 29 features and 89148 instances. After the feature extraction authors ended up with 29 features:

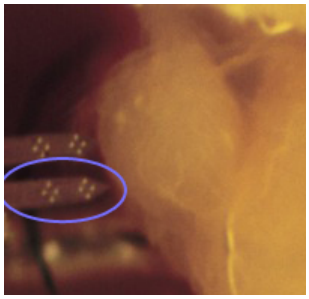

Figure 1: Shank with 8 electrodes (channels).

[1] Accuracy is a naive way to measure performance of the model. Read about *precision* and *recall* here `http://en.wikipedia.org/wiki/Precision_and_recall`

[2] https://crcns.org/data-sets/hc/hc-3/about-hc-3

`1-3` Three principal components, obtained using Principle Component Analysis (PCA)[3] of the data obtained form the channel 1.

`4-24` PCA on channels 2 to 8, three main components per each.

`25` Peak-to-trough on the channel of largest amplitude.

`26` Peak-to-baseline on the channel of largest amplitude.

`27` Trough-to-baseline on the channel of largest amplitude.

`28` Spike duration.

`29` Time moment. You will have to study `https://crcns.org/files/data/hc2/crcns-hc2-data-description.pdf` to find out what the sampling rate is.

The second file `ec012ec.187.whl` contains locations of two LEDs[4] placed on the mouse. The file `data/ec013.939.mpg.tar.gz` has an example of the video from which the location were extracted (this is not the same trial we work with in the exercise, so the trails do not match). There are 4 columns in this file

1. X coordinate of the first LED.
2. Y coordinate of the first LED.
3. X coordinate of the second LED.
4. Y coordinate of the second LED.

We will use only the first LED.

Our goal is to to predict position of the rat only by looking by the features extracted from the neuronal activity. More details on how exactly we are going to do that are in the `codebase4.m` file. The result I got you can see on Figure 2

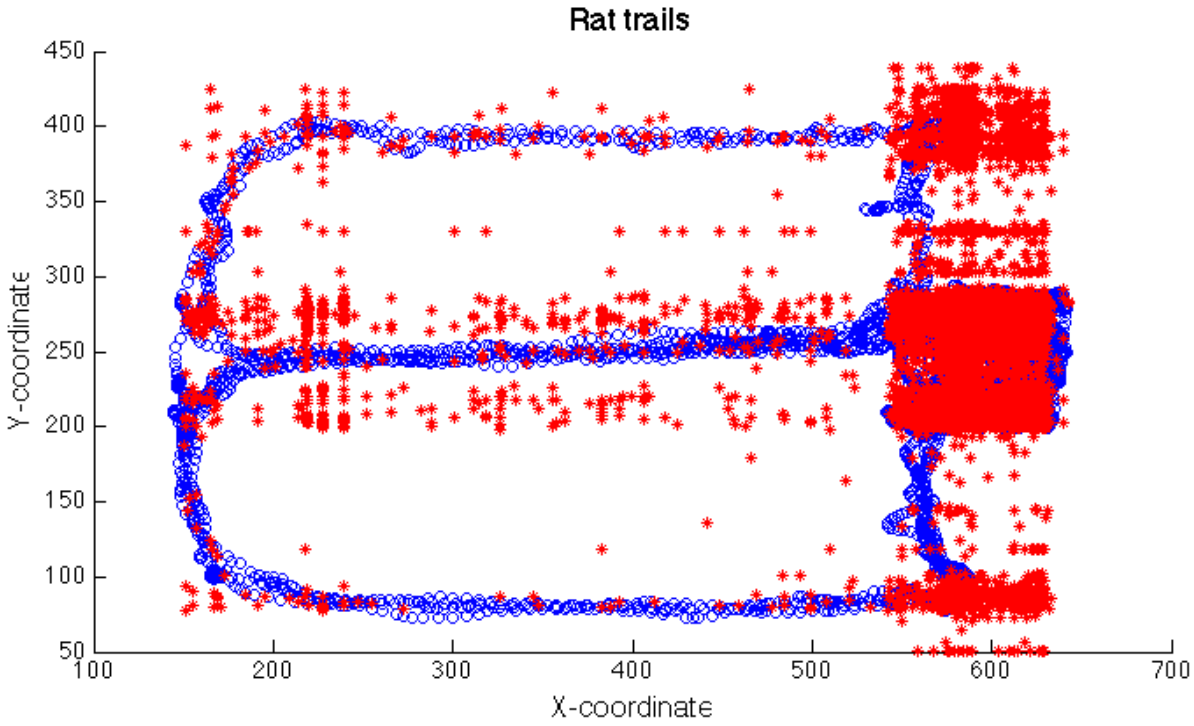

Figure 2: Blue circles indicate true positions of the rat, red crosses – predicted positions.

[3] `http://en.wikipedia.org/wiki/Principal_component_analysis`
[4] `http://en.wikipedia.org/wiki/Light-emitting_diode`

In your report describe the logic behind the operations you are performing, results, plots and interpretations.

Previous exercises from this and previous sessions might have left you with a feeling that it is quite easy to achieve a result in neuroscience: you just think a little bit about the representation of your data, then use bunch of tools and the result is in your hands. The next exercise will give you the real feeling and give an opportunity to tackle with a problem.

Exercise 3: PCA on fMRI data (1+1*pt)

The most spatially precise non-invasive way of measuring brain activity is the Functional Magnetic Resonance Imaging (fMRI)[5]. Things like reconstruction a video stream form the visual cortex has been attempted and successed[6].

The fMRI data is rather massive: at each moment of time (which is 1-2 seconds) the machine records the activity of ≈100,000 voxels (see Figure 3). As you can imagine it is quite a challenge from computational point of view to analyze this kind of data.

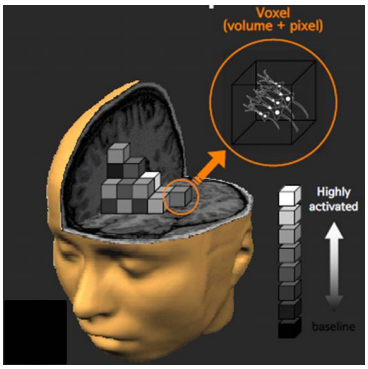

Figure 3: Voxel.

In this exercise we will use the dataset[7], where the region of the visual cortex (≈25,000 voxels) was recorded while the test subject was looking at different images (see Figure 4 for an example). The dataset consists of 1750 training instances and 120 test (validation) instances. We will be working with pre-processed data, which, nevertheless, has the size of 700MB (the orignial data is about 10GB).

In this exercise we apply PCA on this data to reduce it. The funny thing is, that you actually can throw more than half of it away without loosing any information at all.

The codebase in the file `codebase4.m` will get you started. Go through the code and understand the logic of what is happening. Describe this logic in your report. Finish the code and run it, add pictures, plots and results what you get to the report.

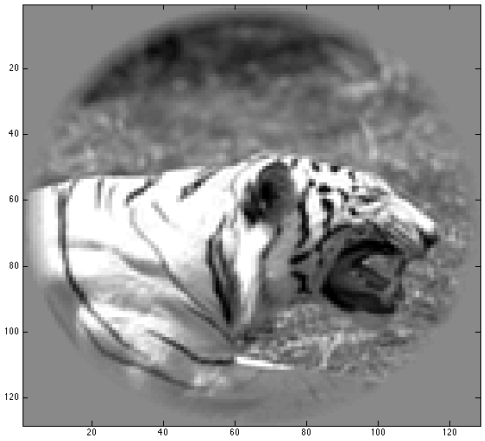

Figure 4: An example of a stimulus.

This is enough to get 1 point for this exercise. But since you are fascinated by the potential possibility of extracting visual information from the human brain you might want to continue

[5] `http://en.wikipedia.org/wiki/Functional_magnetic_resonance_imaging`
[6] See for youself: `http://www.youtube.com/watch?v=nsjDnYxJ0bo`
[7] `https://crcns.org/data-sets/vc/vim-1/about-vim-1`

and try to create an actual model, which will try to reconstruct images form the brain data. For that you will need to:

- Perform the PCA on images.
- Perform the PCA on features.
- Tranform test data into PC space using coefficients you obtained from PCA.
- Create a classifier.
- Evaluate the results.

I must warn you, that

1. On average laptop building such a model with naive approach might take one day or so (10 hours on my laptop with 500 features and full images).
2. If your reconstruction will show sky in light colors and land in dark color and everything else is not recognizable, this is already pretty good.

Sufficient effort will be awarded with 1 bonus point.

If you use Octave then you will need to install `statistics`[8] package to run some functions we use. To do that open your Octave environment and type

```
> pkg install -forge io
> pkg install -forge statistics
```

Exercise 4*: Extract orientation (1pt)
As you remember in the previous practice session we tried to identify orientation of the bar by looking at the neuronal activity of the rat. The dataset we used is `http://crcns.org/files/data/lgn-1/crcns_lgn-1_data_description.pdf`.

Your task is to take the data from the previous session and attempt to once more implement the whole pipeline of applying machine learning to the data:

1. Take the data and look at it. Read the description, study the values, etc. until you understand it.
2. Think how you can convert spiking data into features.
3. Extract the features you came up with and create your instances.
4. Create list of classes (12 orientations in this case).
5. Split dataset into training and test sets.
6. Create a model using training set.
7. Evaluate it's performance on the test set.
8. Try to change something to increase the performance.
9. Report what you did, what you got and interpret the results.

[8] `http://octave.sourceforge.net/statistics/index.html`

Exercise 5: Performance measures (0.5pt)
Read about *precision* and *recall* (`http://en.wikipedia.org/wiki/Precision_and_recall`). Talk about true/false positives/negatives and how to compute precision and recall. Explain what is the intuition behind these concepts. Come up with an example where simple accuracy does not give reasonable evaluation of the model. Find any other performance measure, explain it and demonstrate how it is better on the same example where accuracy fails.

Please submit a `pdf` report with answers to the questions and comments about your solutions. Also submit a code for the programming exercise(s). Pack those into `zip` archive and upload to the course web page.