Peak detection using machine learning

General idea: 1st we generate training data. Then we classify and predict. Finally, we visualize predictions.

1. Start by running the file peak\_detect\_dialog\_w\_gui. Useful tips are displayed when mouse is over different fields.

**Training data**

Some important parameters to adjust are listed below. Some trial and error is need for for each dataset. Parameters can be saved and loaded for future use.

* + 1. Filter (gaussian) – Window size and width. Adjust so that the noise is removed but signal is not affected much. Window size should be about 3 times the window width. Note: It is applied to the double derivative and not the original signal.
    2. Thresholds (for double derivative)– Preliminary peak detection is using double derivative. 2nd derivative has a minima when the original signal has a maxima (and vice versa). To visualize both the original signal and 2nd derivative, check the check box in the 'training data' panel. Adjust so that we have false positives but no false negatives at this stage. Smaller values of 1st threshold will detect more peaks in the 2nd derivative and consequently more peaks in the signal.
    3. Display range - Adjusting the display range using properly is crucial because this affects our perception of what is or isn’t a peak quite drastically. The y-range is relative to mean value of signal in x-range. If X-range is longer than full length of signal, full signal is displayed. This can be used if we need to view peaks in relation to full signal.
    4. Range – To select s sub-range of signal in which to look for peaks.
    5. Crop – Number of points to crop before and after the peak for training. Must be enough to include features in the feature list (like peak threshold) for each peak.

***Tips:***

1. It is important to be as consistent as possible when generating training data.
2. Cross validation accuracy assumes that the training data is correct. But it's possible that multiple labeling of the same training data leads to different labels. The cross-validation should be compared against this human accuracy.

**Estimation of accuracy of training data:**

1. Use generate\_training\_calib\_datasets.ipynb to generate multiple training data sets from combined file obtained by combining cropped peaks and peak\_vars. Choose how many peaks to relabel and how many times. generate\_training\_calib\_datasets.ipynb generates 'num\_repeats' data sets the first of which has original selection values but the rest have NaN. The selection value of other peaks can be assesed using plot\_peaks.py in which detect\_features is set to empty list. For now the 0 or 1 for selection are entered manually in the sample data sets.
2. Use calibrate\_training\_accuracy.ipynb to get the accuracy tested against the original training data set.

**Following steps are specific to analysis of synaptic currents**

1. Use merge\_genotype\_info.ipynb to merge mouse info. Merged file overwrites data file.
2. Use pt\_psc\_freq\_and\_amp.ipynb to calculate peak amplitude and peak freq averaged for each cell. Uses only selected peaks. Outputs 2 files

* peak\_amp\_cell\_avg.csv
* peak\_freq\_cell\_avg.csv

**Known issues and possible enhancements**

1. Double derivative peak detection can be replaced with template matching for more analyzing more complex peaks.

2. A pop-up should let users know that previously cropped peaks exist and let user choose whether to use previous analysis. Right now, the above happens by default, so the user needs to delete previous analysis to reanalyze from scratch.

3. Extra filtering (large value of sigma in Gaussian filter) leads to phase shift. Smaller values are fine.