Sensie Documentation

Release 0.1

Colin Jacobs

Jun 19, 2020

API documentation:

CHAPTER 1

Introduction

Sensie's goal is to quickly interrogate a trained machine learning model, determining the sensitivity to a parameter or perturbation of the data.

Sensie probes the sensitivity of a network to inputs with a particular property, *p. This property can be a feature of the data; an otherwise known property of a test or training set that is not provided explicitly in training; or a function that can artificially vary this property for a supplied test set. The effect of a particular property is measured according to the variance it introduces to the correct output of the network, such as the score for the correct class . Quantitatively, we would like to know the function *mean_score = f(p)* for some property p; Sensie can calculate a linear approximation to this unknown function.

For more information and examples, see the GitHub repository at https://github.com/coljac/sensie.

CHAPTER 2

Installation

Check out the repository and install with:

```
pip install .
```

(or add the sensie directory to your PYTHONPATH.)

Dependencies are listed in `requirements.txt` included in the repository. Sensie requires python 3.6 or above.

Optionally, install pytest with `pip install pytest`, then run the tests with `pytest test` from the repository root.

2.1 sensie package

2.1.1 Module contents

class sensie.**Probe**(*model*, *predict_function=None*)

Bases: object

A class that wraps a pre-trained model and provides methods for testing its robustness and sensitivity to various properties.

plot_property(*test*, *label='property'*, *show_fit=False*, *fit='line'*, *save_to=None*, *ticklabels=None*, *errorbars=True*, *fitorder=2*)

Generates a plot from a SingleTest result.

test: SingleTest The test to visualize.

label: str Readable description for the property tested.

show_fit: bool If True, a fit to the data will be plotted.

fit: str "line" or "polynomial" - the fit to be shown.

fitorder: For a polynomial, the order of the fit.

save_to: str Filename to save the figure to.

ticklabels: list Labels for the x axis. Useful (for instance) when plotting class names.

errorbars: bool Plot error bars - one standard deviation from the mean score in the correct class.

predict_and_measure(*x_test*, *y_test*, *p_test*, *prop=None*, *continuous=False*, *bins=20*, *label=None*, *plot=False*, *propnames=None*, *batch_size=256*) → sensie.SensitivityMeasure

Scores the provided x_test and returns a SensitivityMeasure object with measured values and for plotting.

x_test: numpy.ndarray Tensor of examples for testing

y_test: numpy.ndarray Vector of ground-truth classes

p_test: numpy.ndarray or pandas.DataFrame Tensor or DataFrame containing the property/properties for testing.

prop: int or str (Optional) A numerical or string index into p_test, returning a vector or Series of the property in question. If this is None, will attempt for all columns in p_test

continuous: bool If true, assumes the p value is continues and needs to be binned.

bins: int Number of bins; used if continuous == True.

label: str (Optional) An string label for the property/properties in question; used for plotting.

plot: bool If True, produce and display a plot of the results.

propnames: list or array A list of property names, corresponding to p_test.

batch_size: int When calling the predict method, the batch size to use.

SensitivityMeasure An object containing summary information about the analysis.

predict_and_measure_perturbed(*x_test*, *y_test*, *perturber*, *p_values=None*, *p_min=0*, *p_max=1*, *steps=10*, *label=None*, *plot=False*, *ci=False*, *batch_size=1024*) → sensie.SensitivityMeasure

Scores the provided x_test as altered by the supplied perturber function, and returns a SensitivityMeasure object with measured values and for plotting.

x_test: numpy.ndarray Tensor of examples for testing

y_test: numpy.ndarray Vector of ground-truth classes

perturber: function A function, f(x_test, p), which alters (perturbs) the test set by an amount or scale p.

p_values: list or ndarray An iterable list of p_values to be passed to the perturber function and measured. If not supplied, numpy.linspace(p_low, p_high, steps) will be used instead.

p_min: int The minimum, and first, value for p to be passed to the perturber function.

p_max: int The maximum, and last, value for p to be passed to the perturber function.

steps: The number of steps from p_min to p_max to be passed to the perturber function.

label: str (Optional) An string label for the property/properties in question; used for plotting.

plot: bool If True, produce and display a plot of the results.

ci: bool If True, will conduct linear fit and generate credible intervals.

batch_size: int The x_test examples will be perturbed and scored in batches of this size.

SensitivityMeasure an object containing summary information about the analysis.

test_class_sensitivity(*x_test*, *y_test*, *plot=False*)
Same as predict_and_measure, except the property is the ground truth class itself. Useful to see if certain classes in the test set have markedly different performance to others.

x_test: numpy.ndarray Tensor of examples for testing

y_test: numpy.ndarray Vector of ground-truth classes

plot: bool If True, generates a plot of the results.

class sensie.**SensitivityMeasure**(*x_test*, *y_test*, *rightscores*)
Bases: object

This object wraps the individual tests performed on a model, and provides convience methods for setting credible intervals and displaying a summary.

set_credible_intervals()
Calculates credible intervals for each test performed so far (i.e. for each SingleTest instance).

summary()
Produces a summary table (as a pandas DataFrame) with the results, and significance of, tests performed.
Returns:

A pandas DataFrame with a row for each test performed.

class sensie.**SingleTest**(*property_name*, *p_vals*, *means*, *stds*, *p_points=None*, *y_vals=None*)
Bases: object

Encapsulates the results of a single signficance test.

get_gradient() → float
Returns the gradient of the test - the change in mean score by p.

Returns: float The gradient from a linear fit to xs, ys.

get_significance(*significance_floor=0.02*)
Returns a string indicating the significance of a sensitivity measure ("low", "medium", or "high")

set_credible_interval(*means_only=False*, *tune=None*, *samples=400*)
Runs pymc3 inference code to determine the slope of the relationship between p and accuracy, and saves 50% and 95% credible intervals in instance variables. The results are stored in this SingleTest instance.

sort_and_reorder(*labels=None*)
Reorders the test results by y-value, i.e. the mean correct-class score. Useful for testing of discrete, unordered properties such as class.

labels: list Labels for the classes/discrete values.

Returns: list The provided labels, in the re-ordered order (for plotting, etc).

summary()
Show the result (gradient) of score sensitivity to this property, optionally with credible intervals.

Returns: A pandas.DataFrame with the results of the test, including credible intervals if calculated.

sensie.**progbar**(*current*, *to*, *width=40*, *show=True*, *message=None*, *stderr=False*)
Displays a progress bar for use in certain testing operations.

CHAPTER 3

Issues, Questions and Contributions

Any problems or questions? Email colin@coljac.net, or open an issue on GitHub at https://github.com/coljac/sensie.

Contributions are welcome and encouraged. Fork the GitHub repository to your own machine, make some changes, and push your work back up to the fork and open a pull request so that I can review and incorporate the changes.

CHAPTER 4

Index

- genindex
- modindex
- search

Python Module Index

S

Index