Variable Dropout Documentation

Release 1.0

group

Jun 27, 2018

Contents:

CHAPTER 1

Usage

Package can be used to establish the features importance for any classifiaction or regression model. You can present the result in text and graphical form. Additonally, you can present a few plots on the one graphics.

1.1 Method description

The functionality is based on the comparison of the loss function value for two models. The first model is the full one. In the second, values for the one variable are shuffled. In order to minimize the influence of randomness, the number of algorithm iteration was added. The final result is obtained as the mean of the results from the all iterations.

1.2 Installation

```
git clone https://github.com/Noctiphobia/variable-dropout/
cd variable-dropout
python setup.py install
```

If python resolves to Python 2, replace it with python3.

1.3 Requirements

- python 3.6
- scikit-learn 0.19.1
- numpy 1.11.3
- pandas 0.22.0
- matplotlib 2.2.0

1.4 Basic usage

1.4.1 Load dataset

Import necessary packages.

```
from sklearn import datasets
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
import sklearn as sk
import numpy as np
import pandas as pd
from xgboost import XGBClassifier
from variable_dropout.variable_dropout import variable_dropout
from variable_dropout.plot_variable_dropout import plot_variable_dropout
```

Import data, extract input dataset and a target vector.

```
dataset = datasets.load_breast_cancer()
data = pd.DataFrame(dataset.data)
target = dataset.target
data.columns = dataset.feature_names
```

1.4.2 Prepare models

Create a classification or a regression model.

```
model_rf = RandomForestClassifier(random_state=0)
model_lr = LogisticRegression(random_state=0)
model_xgb = XGBClassifier(random_state=0)
```

1.4.3 Train models

Train models on data.

```
model_rf.fit(X = data, y = target)
model_lr.fit(X = data, y = target)
model_xgb.fit(X = data, y = target)
```

1.4.4 Get features importance

Compute features importance for models.

```
importance_rf = variable_dropout(model_rf, data, target, loss_function=sk.metrics.
↪hinge_loss, random_state=0)
importance_lr = variable_dropout(model_lr, data, target, loss_function=sk.metrics.
↪hinge_loss, random_state=0)
importance_xgb = variable_dropout(model_xgb, data, target, loss_function=sk.metrics.
↪hinge_loss, random_state=0)
```

1.4.5 Text form of importance

Display computed importance for a model.

```
importance_rf
```

	variable	dropout_loss	label
0	_baseline_	0.84016	RandomForestClassifier
1	area error	0.39089	RandomForestClassifier
2	worst radius	0.38959	RandomForestClassifier
3	mean concave points	0.38006	RandomForestClassifier
4	mean texture	0.38068	RandomForestClassifier
5	worst texture	0.37935	RandomForestClassifier
6	worst concave points	0.38272	RandomForestClassifier
7	worst concavity	0.37685	RandomForestClassifier
8	worst perimeter	0.37748	RandomForestClassifier
9	mean perimeter	0.37744	RandomForestClassifier
10	mean concavity	0.37591	RandomForestClassifier
11	worst area	0.37906	RandomForestClassifier
12	worst smoothness	0.37760	RandomForestClassifier
13	mean smoothness	0.37539	RandomForestClassifier
14	mean radius	0.37618	RandomForestClassifier
15	mean symmetry	0.37454	RandomForestClassifier
16	concavity error	0.37489	RandomForestClassifier
17	mean fractal dimension	0.37485	RandomForestClassifier
18	radius error	0.37592	RandomForestClassifier
19	perimeter error	0.37539	RandomForestClassifier
20	concave points error	0.37482	RandomForestClassifier
21	mean compactness	0.37410	RandomForestClassifier
22	texture error	0.37410	RandomForestClassifier
23	smoothness error	0.37424	RandomForestClassifier
24	compactness error	0.37410	RandomForestClassifier
25	symmetry error	0.37426	RandomForestClassifier
26	fractal dimension error	0.37413	RandomForestClassifier
27	worst compactness	0.37467	RandomForestClassifier
28	worst symmetry	0.37426	RandomForestClassifier
29	worst fractal dimension	0.37434	RandomForestClassifier
30	mean area	0.37381	RandomForestClassifier
31	_full_model_	0.37410	RandomForestClassifier

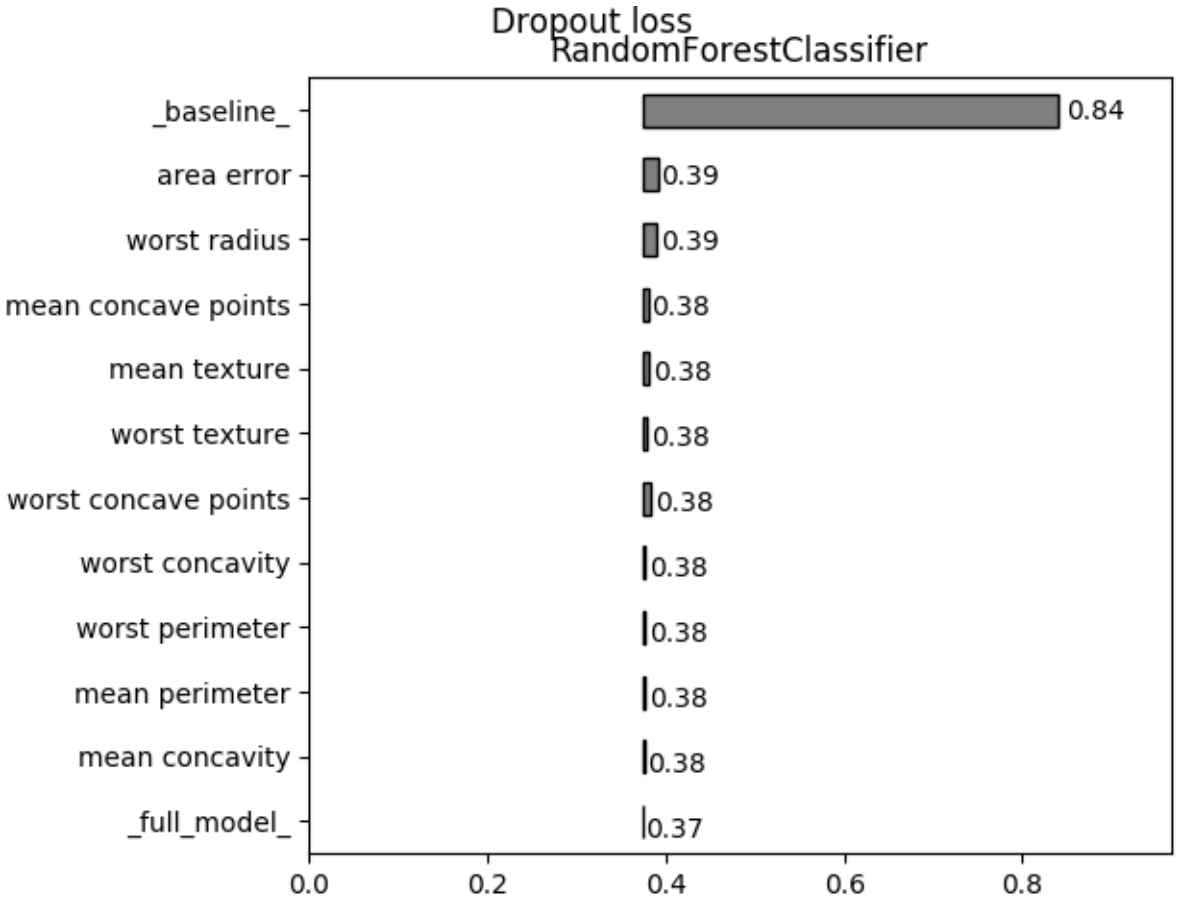
1.4.6 Visual form of importance

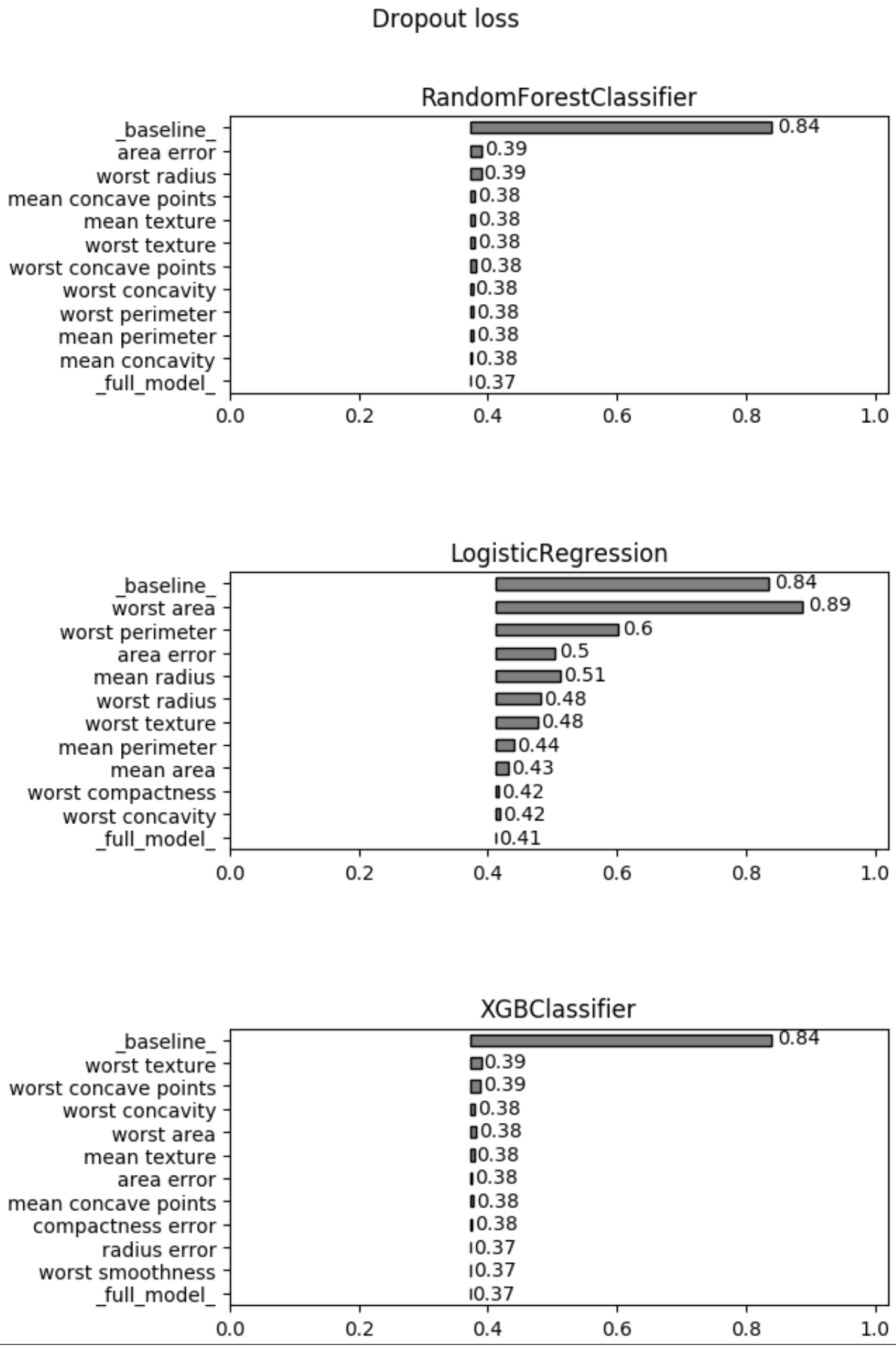
Visualize importance for one model.

```
plot_variable_dropout(importance_rf)
```

Visualize importance for multiple models.

```
plot_variable_dropout(importance_rf, importance_lr,importance_xgb)
```


Dropout loss
RandomForestClassifier
baseline
area error
worst radius
mean concave points
mean texture
worst texture
worst concave points
worst concavity
worst perimeter
mean perimeter
mean concavity
_full_model_
0.84
0.39
0.39
0.38
0.38
0.38
0.38
0.38
0.38
0.38
0.38
0.37
0.0
0.2
0.4
0.6
0.8


Dropout loss
RandomForestClassifier
baseline 0.84
area error 0.39
worst radius 0.39
mean concave points 0.38
mean texture 0.38
worst texture 0.38
worst concave points 0.38
worst concavity 0.38
worst perimeter 0.38
mean perimeter 0.38
mean concavity 0.38
_full_model_ 0.37
0.0 0.2 0.4 0.6 0.8 1.0
LogisticRegression
baseline 0.84
worst area 0.89
worst perimeter 0.6
area error 0.5
mean radius 0.51
worst radius 0.48
worst texture 0.48
mean perimeter 0.44
mean area 0.43
worst compactness 0.42
worst concavity 0.42
_full_model_ 0.41
0.0 0.2 0.4 0.6 0.8 1.0
XGBClassifier
baseline 0.84
worst texture 0.39
worst concave points 0.39
worst concavity 0.38
worst area 0.38
mean texture 0.38
area error 0.38
mean concave points 0.38
compactness error 0.38
radius error 0.37
worst smoothness 0.37
_full_model_ 0.37
0.0 0.2 0.4 0.6 0.8 1.0

CHAPTER 2

Code

2.1 Submodules

2.1.1 variable_dropout.variable_dropout

class variable_dropout.variable_dropout.**DropoutType**

Method of variable dropout loss representation. One of the following:

RAW - raw value of variable dropout loss,

RATIO - ratio of loss of variable dropout to loss for unperturbed model,

DIFFERENCE - difference between variable dropout loss and unperturbed model loss

variable_dropout.variable_dropout.**variable_dropout**(*estimator: Any, X: pandas.core.frame.DataFrame, y: Iterable[Any], loss_function: Callable[[Iterable[Any], Iterable[Any]], float] = <function mean_squared_error>, dropout_type: variable_dropout.variable_dropout.DropoutType = <DropoutType.RAW: (<function DropoutType.<lambda>>,)>, n_sample: int = 1000, n_iters: int = 100, random_state: Union[int, mtrand.RandomState, NoneType] = None, label=None*) → pandas.core.frame.DataFrame

Determines importance of variables in the model. Model trained on all variables is used to predict result variable for data with one variable randomly shuffled. The worse the result with a particular variable shuffled is, the more important the variable is.

Parameters

- **estimator** – any fitted classification or regression model with predict method.
- **X** – samples.
- **y** – result variable for samples.
- **loss_function** – a function taking vectors of real and predicted results. The better the prediction, the smaller the returned value.
- **dropout_type** – method of loss representation. One of values specified in DropoutType enumeration.
- **n_sample** – number of samples to predict for. Given number of samples. is randomly chosen from X with replacement.
- **n_iters** – number of iterations. Final result is mean of the results of iterations.
- **random_state** – ensures deterministic results if run twice with the same value.

Returns series of variable dropout loss sorted descending.

2.1.2 variable_dropout.plot_variable_dropout

variable_dropout.plot_variable_dropout.**plot_variable_dropout**(**args*, *max_vars: Union[int, NoneType] = 10*, *include_baseline_and_full: bool = True*) → None

Plots the results of variable_dropout.

Parameters

- **args** – any number of variable_dropout results.
- **max_vars** – maximum number of variables to plot per classifier, or None to plot all of them.
- **include_baseline_and_full** – whether to include _baseline_ and _full_model_ in the plot.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

v

Index