text_explainability

Marcel Robeer

Sep 06, 2023

USING TEXT$_{EXPLAINABILITY}$

`text_explainability` provides a **generic architecture** from which well-known state-of-the-art explainability approaches for text can be composed. This modular architecture allows components to be swapped out and combined, to **quickly develop new types of explainability approaches** for (natural language) text, or to **improve a plethora of approaches by improving a single module**.

Several example methods are included, which provide **local explanations** (*explaining the prediction of a single instance*, e.g. `LIME` and `SHAP`) or **global explanations** (*explaining the dataset, or model behavior on the dataset*, e.g. `TokenFrequency` and `MMDCritic`). By replacing the default modules (e.g. local data generation, global data sampling or improved embedding methods), these methods can be improved upon or new methods can be introduced.

CHAPTER

ONE

QUICK TOUR

Local explanation: explain a models' prediction on a given sample, self-provided or from a dataset.

```
from text_explainability import LIME, LocalTree

# Define sample to explain
sample = 'Explain why this is positive and not negative!'

# LIME explanation (local feature importance)
LIME().explain(sample, model).scores

# List of local rules, extracted from tree
LocalTree().explain(sample, model).rules
```

Global explanation: explain the whole dataset (e.g. train set, test set), and what they look like for the ground-truth or predicted labels.

```
from text_explainability import import_data, TokenFrequency, MMDCritic

# Import dataset
env = import_data('./datasets/test.csv', data_cols=['fulltext'], label_cols=['label'])

# Top-k most frequent tokens per label
TokenFrequency(env.dataset).explain(labelprovider=env.labels, explain_model=False, k=3)

# 2 prototypes and 1 criticisms for the dataset
MMDCritic(env.dataset)(n_prototypes=2, n_criticisms=1)
```

CHAPTER

TWO

USING TEXT_EXPLAINABILITY

Installation
Installation guide, directly installing it via pip or through the git.

Example Usage
An extended usage example.

Explanation Methods Included
Overview of the explanation methods included in `text_explainability`.

text_explainability API reference
A reference to all classes and functions included in the `text_explainability`.

CHAPTER

THREE

DEVELOPMENT

text_explainability @ GIT
: The git includes the open-source code and the most recent development version.

Changelog
: Changes for each version are recorded in the changelog.

Contributing
: Contributors to the open-source project and contribution guidelines.

CHAPTER

FOUR

EXTENSIONS

`text_explainability` can be extended to also perform *sensitivity testing*, checking for machine learning model robustness and fairness. The `text_sensitivity` package is available through PyPI and fully documented at https://text-sensitivity.readthedocs.io/.

CHAPTER

FIVE

CITATION

```
@misc{text_explainability,
  title = {Python package text\_explainability},
  author = {Marcel Robeer},
  howpublished = {\url{https://git.science.uu.nl/m.j.robeer/text_explainability}},
  year = {2021}
}
```

5.1 Installation

Installation of `text_explainability` requires `Python 3.8` or higher.

5.1.1 1. Python installation

Install Python on your operating system using the Python Setup and Usage guide.

5.1.2 2. Installing `text_explainability`

`text_explainability` can be installed:

- *using pip*: *pip3 install* (released on [PyPI](https://pypi.org/project/text-explainability/))
- *locally*: cloning the repository and using `python3 setup.py install`

Using `pip`

1. Open up a `terminal` (Linux / macOS) or `cmd.exe/powershell.exe` (Windows)
2. Run the command:
    - `pip3 install text_explainability`, or
    - `pip install text_explainability`.

```
user@terminal:~$ pip3 install text_explainability
Collecting text_explainability
...
Installing collected packages: text-explainability
Successfully installed text-explainability
```

Speeding up the explanation-generation process can be done by using `pip3 install text_explainability[fast]` or having `fastcountvectorizer` installed.

Locally

1. Download the folder from `GitLab/GitHub`:
    - Clone this repository, or
    - Download it as a `.zip` file and extract it.
2. Open up a `terminal` (Linux / macOS) or `cmd.exe/powershell.exe` (Windows) and navigate to the folder you downloaded `text_explainability` in.
3. In the main folder (containing the `setup.py` file) run:
    - `python3 setup.py install`, or
    - `python setup.py install`.

```
user@terminal:~$ cd ~/text_explainability
user@terminal:~/text_explanability$ python3 setup.py install
running install
running bdist_egg
running egg_info
...
Finished processing dependencies for text-explainability
```

5.2 Example Usage

5.2.1 Dependencies

`text_explainability` uses instances and machine learning models wrapped with the InstanceLib library. For your convenience, we wrap some `instancelib` functions in `text_explainability.data` and `explainability.model`.

```
from text_explainability.data import import_data, train_test_split, from_string
from text_explainability.model import import_model
```

5.2.2 Dataset and model

As a dummy black-box model, we use the example dataset in `./datasets/test.csv` and train a machine learning model on it with `scikit-learn`.

```
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier

# Create train/test dataset
env = import_data('./datasets/test.csv', data_cols='fulltext', label_cols='label')
env = train_test_split(env, train_size=0.70)
```

(continues on next page)

(continued from previous page)

```
# Create sklearn model with pipeline
pipeline = Pipeline([('tfidf', TfidfVectorizer(use_idf=True)),
                     ('rf', RandomForestClassifier(random_state=0))])

# Build and fit (train) model
model = import_model(pipeline, environment=env)
```

5.2.3 Using Text Explainability

Text Explainability is used for *local explanations* (explaining a single prediction) or *global explanations* (explaining general dataset/model behavior).

Local explanations

Popular local explanations include `LIME`, `KernelSHAP`, local decion trees (`LocalTree`), local decision rules (`LocalRules`) and `FoilTree`. First, let us create a sample to explain:

```
from text_explainability.data import from_string

sample = from_string('Dit is zeer positieve of negatieve proef... Of toch negatief?')
```

Next, the prediction of `model` on `sample` can be explained by generating neighborhood data (`text_explainability.data.augmentation.TokenReplacement`), used by `LIME` (and its extension `BayLIME`), `LocalTree`, `FoilTree` and `KernelSHAP`:

```
from text_explainability import BayLIME, LIME, LocalTree, FoilTree, KernelSHAP

# LIME explainer for `sample` on `model`
explainer = LIME(env)
explainer(sample, model, labels=['neutraal', 'positief']).scores

# SHAP explanation for `sample` on `model`, limited to 4 features
KernelSHAP(label_names=labelprovider)(sample, model, n_samples=50, l1_reg=4)

# Bayesian extension of LIME with 1000 samples
BayLIME()(sample, model, n_samples=1000)

# Local tree explainer for `sample` on `model` (non-weighted neighborhood data)
LocalTree()(sample, model, weigh_samples=False)

# Contrastive local tree explainer for `sample` on `model` (why not 'positief'?)
FoilTree()(sample, model, foil_fn='positief').rules

# LocalRules on `model` (why 'positief'?)
LocalRules()(sample, model, foil_fn='negatief', n_samples=100).rules
```

Global explanations

Global explanations provide information on the dataset and its ground-truth labels, or the dataset and corresponding predictions by the `model`. Example global explanations are `TokenFrequency` (the frequency of each token per label/class/bucket) or `TokenInformation` (how informative each token is for predicting the various labels).

```
from text_explainability import TokenFrequency, TokenInformation

# Global word frequency explanation on ground-truth labels
tf = TokenFrequency(env.dataset)
tf(labelprovider=env.labels, explain_model=False, k=10).scores

# Global word frequency explanation on model predictions
tf(model=model, explain_model=True, k=3, filter_words=PUNCTUATION)

# Token information for dataset
ti = TokenInformation(env.dataset)
ti(labelprovider=env.labels, explain_model=False, k=50).scores

# Token information for model
ti(model=model, explain_model=True, k=50, filter_words=PUNCTUATION)
```

Global explanation: Explanation by example

Explanations by example provide information on a dataset (e.g. the test set) or subsets thereof (e.g. all training instances with label 0) by showing representative instances. Examples of representative instances are prototypes (`n` most representative instances, e.g. of a class) and criticsms (`n` instances not well represented by prototypes). Example explanations by example are `KMedoids` (using the *k-Medoids* algorithm to extract prototypes) and `MMDCritic` (extracting prototypes and corresponding criticisms). In addition, each of these can be performed labelwise (e.g. for the ground-truth labels in a `labelprovider` or for each models' predicted class).

```
from text_explainability import KMedoids, MMDCritic, LabelwiseMMDCritic

# Extract top-2 prototypes with KMedoids
KMedoids(env.dataset).prototypes(n=2)

# Extract top-2 prototypes and top-2 criticisms label with MMDCritic
MMDCritic(env.dataset)(n_prototypes=2, n_criticisms=2)

# Extract 1 prototype for each ground-truth label with MMDCritic
LabelwiseMMDCritic(env.dataset, labelprovider).prototypes(n=1)

# Extract 1 prototype and 2 criticisms for each predicted label with MMDCritic
LabelwiseMMDCritic(env.dataset, model)(n_prototypes=1, n_criticisms=2)
```

5.3 Explanation Methods Included

`text_explainability` includes methods for model-agnostic *local explanation* and *global explanation*. Each of these methods can be fully customized to fit the explainees' needs.

Table 1: Explanation methods in *text_explainability*

Type	Explanation method	Description	Paper/link
Local explanation	`LIME`	Calculate feature attribution with *Local Intepretable Model-Agnostic Explanations* (LIME).	[Ribeiro2016], interpretable-ml/lime
	`KernelSHAI`	Calculate feature attribution with *Shapley Additive Explanations* (SHAP).	[Lundberg2017], interpretable-ml/shap
	`LocalTree`	Fit a local decision tree around a single decision.	[Guidotti2018]
	`LocalRules`	Fit a local sparse set of label-specific rules using `SkopeRules`.	github/skope-rules
	`FoilTree`	Fit a local contrastive/counterfactual decision tree around a single decision.	[Robeer2018]
	`BayLIME`	Bayesian extension of LIME for include prior knowledge and more consistent explanations.	[Zhao201]
Global explanation	`TokenFrequ`	Show the top-*k* number of tokens for each ground-truth or predicted label.	
	`TokenInfo`	Show the top-*k* token mutual information for a dataset or model.	wikipedia/mutual_information
	`KMedoids`	Embed instances and find top-*n* prototypes (can also be performed for each label using `LabelwiseKMedoids`).	interpretable-ml/prototypes
	`MMDCritic`	Embed instances and find top-*n* prototypes and top-*n* criticisms (can also be performed for each label using `LabelwiseMMDCritic`).	[Kim2016], interpretable-ml/prototypes

5.3.1 Credits

- Florian Gardin, Ronan Gautier, Nicolas Goix, Bibi Ndiaye and Jean-Matthieu Schertzer. Skope-rules. 2020.
- Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini and Fosca Gianotti. Local Rule-Based Explanations of Black Box Decision Systems. 2018.
- Been Kim, Rajiv Khanna and Oluwasanmi O. Koyejo. Examples are not Enough, Learn to Criticize! Criticism for Interpretability. *Advances in Neural Information Processing Systems (NIPS 2016)*. 2016.
- Scott Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. *31st Conference on Neural Information Processing Systems (NIPS 2017)*. 2017.
- Christoph Molnar. Interpretable Machine Learning: A Guide for Making Black Box Models Explainable. 2021.
- Marco Tulio Ribeiro, Sameer Singh and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016)*. 2016.
- Marco Tulio Ribeiro, Sameer Singh and Carlos Guestrin. Anchors: High-Precision Model-Agnostic Explanations. *AAAI Conference on Artificial Intelligence (AAAI)*. 2018.

- Jasper van der Waa, Marcel Robeer, Jurriaan van Diggelen, Matthieu Brinkhuis and Mark Neerincx. "Contrastive Explanations with Local Foil Trees". *2018 Workshop on Human Interpretability in Machine Learning (WHI 2018)*. 2018.

5.4 text_explainability

Subpackages:

5.4.1 text_explainability.data

Data imports, sampling and generation.

`text_explainability.data.`**`from_list`**(*instances*, *labels*)

Create a TextEnvironment from a list of instances, and list of labels

Example

```
>>> from_list(instances=['A positive test.', 'A negative test.', 'Another positive␣
↪test'],
>>>           labels=['pos', 'neg', 'pos'])
```

Parameters

- **`instances`** (*`Sequence[str]`*) – List of instances.
- **`labels`** (*`Sequence[LT]`*) – List of corresponding labels.

Returns

Environment holding data (*.dataset*) and labelprovider (*.labels*).

Return type

TextEnvironment

`text_explainability.data.`**`from_string`**(*string*, *tokenizer=<function word_tokenizer>*)

Create a MemoryTextInstance from a string.

Example

```
>>> from_string('This is a test example.')
```

Parameters

- **`string`** (*`str`*) – Input string.
- **`tokenizer`** (*`Callable[[str], Sequence[str]], optional`*) – Tokenizer that converts string into list of tokens (e.g. words or characters). Defaults to default_tokenizer.

Returns

Holds information on the string, and its tokenized representation.

Return type

MemoryTextInstance

text_explainability.data.**import_data**(*dataset*, *data_cols*, *label_cols*, *label_map=None*, *method='infer'*, *_to_instancelib=True*, ***read_kwargs*)

Import data in an instancelib Environment.

Examples

Import from an online .csv file with data in the 'text' column and labels in 'category':

```
>>> from genbase import import_data
>>> ds = import_data('https://storage.googleapis.com/dataset-uploader/bbc/bbc-text.
↪csv',
                     data_cols='text', label_cols='category')
```

Convert a pandas DataFrame to instancelib Environment:

```
>>> from genbase import import_data
>>> import pandas as pd
>>> df = pd.read_csv('https://storage.googleapis.com/dataset-uploader/bbc/bbc-text.
↪csv')
>>> ds = import_data(df, data_cols='text', label_cols='category')
```

Download a .zip file and convert each file in the zip to an instancelib Environment:

```
>>> from genbase import import_data
>>> ds = import_data('https://archive.ics.uci.edu/ml/machine-learning-databases/
↪00462/drugsCom_raw.zip',
                     data_cols='review', label_cols='rating')
```

Convert a huggingface dataset (sst2) to an instancelib Environment:

```
>>> from genbase import import_data
>>> from datasets import load_dataset
>>> ds = import_data(load_dataset('glue', 'sst2'), data_cols='sentence', label_cols=
↪'label')
```

Parameters

- **dataset** (*_type_*) – Dataset to import.
- **data_cols** (*Union[KT, List[KT]]*) – Name of column(s) containing data.
- **label_cols** (*Union[KT, List[KT]]*) – Name of column(s) containing labels.
- **label_map** (*Optional[Union[Callable, dict]], optional*) – Label renaming dictionary/function. Defaults to None.
- **method** (*Method, optional*) – Method used to import data. Choose from 'infer', 'glob', 'pandas'. Defaults to 'infer'.
- **_to_instancelib** (*bool, optional*) – Whether to convert the final result to instancelib. Defaults to True.
- ****read_kwargs** – Optional arguments passed to reading call.

Raises

- **ImportError** – Unable to import file.

- **ValueError** – Invalid type of method.
- **NotImplementedError** – Import not yet implemented.

Returns
Environment for each file or dataset provided.

Return type
Union[il.Environment, pd.DataFrame]

text_explainability.data.**train_test_split**(*environment*, *train_size*, *train_name='train'*, *test_name='test'*)

Split an environment into training and test data, and save it to the original environment.

Parameters

- **environment** (*instancelib.Environment*) – Environment containing all data (*environment.dataset*), including labels (*environment.labels*).
- **train_size** (*Union[int, float]*) – Size of training data, as a proportion [0, 1] or number of instances > 1.
- **train_name** (*str, optional*) – Name of train split. Defaults to 'train'.
- **test_name** (*str, optional*) – Name of train split. Defaults to 'test'.

Returns

Environment with named splits *train_name* (containing training data) and *test_name* (containing test data)

Return type
instancelib.Environment

Submodules:

text_explainability.data.augmentation module

Augment a single instance to generate neighborhood data.

class text_explainability.data.augmentation.**LeaveOut**(*detokenizer=<function word_detokenizer>*, *seed=0*)

Bases: *TokenReplacement*

Leave tokens out of the tokenized sequence.

Parameters

- **detokenizer** (*Callable[[Iterable[str]], str]*) – Mapping back from a tokenized instance to a string used in a predictor.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

env: AbstractEnvironment[TypeVar(IT, bound= Instance[Any, Any, Any, Any]), Any, Any, Any, Any, Any]

class text_explainability.data.augmentation.**LocalTokenPertubator**(*detokenizer=<function word_detokenizer>*)

Bases: MultiplePertubator[TextInstance], ChildGenerator[TextInstance], Readable

Perturb a single instance into neighborhood samples.

Parameters
detokenizer (*Callable[[Iterable[str]], str]*) – Mapping back from a tokenized instance to a string used in a predictor.

static binary_inactive(*inactive*, *length*)

Return type
ndarray

env: AbstractEnvironment[TypeVar(IT, bound= Instance[Any, Any, Any, Any]), Any, Any, Any, Any, Any]

perturb(*tokenized_instance*, **args*, ***kwargs*)

Return type
Iterator[Tuple[Iterable[str], Iterable[int]]]

Parameters
- **tokenized_instance** (*Iterable[str]*) –
- **args** (*Any*) –
- **kwargs** (*Any*) –

class text_explainability.data.augmentation.**TokenReplacement**(*detokenizer=<function word_detokenizer>*, *replacement='UNKWRDZ'*, *seed=0*)

Bases: *LocalTokenPertubator*, SeedMixin

Perturb a tokenized instance by replacing with a set token (e.g. 'UNKWRDZ') or deleting it.

Examples

Randomly replace at least two tokens with the replacement word 'UNK':

```
>>> from text_explainability.augmentation import TokenReplacement
>>> TokenReplacement(replacement='UNK').perturb(['perturb', 'this', 'into',
↪'multiple'],
>>>                                             n_samples=3,
>>>                                             min_changes=2)
```

Perturb each token with ['UNK', None]:

```
>>> from text_explainability.augmentation import TokenReplacement
>>> TokenReplacement(replacement=['UNK', None]).perturb(['perturb', 'this', 'into',
↪'multiple'], ...)
```

Perturb with synonyms:

```
>>> from text_explainability.augmentation import TokenReplacement
>>> replacement = {0: ['change', 'adjust'], 1: None, 2: 'to', 3: 'more'}
>>> TokenReplacement(replacement=replacement).perturb(['perturb', 'this', 'into',
↪'multiple'], ...)
```

Parameters

- **detokenizer** (*Callable[[Iterable[str]], str]*) – Mapping back from a tokenized instance to a string used in a predictor.
- **replacement** (*Optional[Union[str, List[str], Dict[int, Optional[Union[str, List[str]]]]]], optional*) – Replacement string, list or dictionary, or set to None if you want to delete the word entirely. Defaults to 'UNKWRDZ'.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

env: AbstractEnvironment[TypeVar(IT, bound= Instance[Any, Any, Any, Any]), Any, Any, Any, Any, Any]

perturb(*tokenized_instance*, *n_samples=50*, *sequential=True*, *contiguous=False*, *min_changes=1*, *max_changes=10000*, *add_background_instance=False*, *seed=None*, ***kwargs*)

Perturb a tokenized instance by replacing it with a single replacement token (e.g. 'UNKWRDZ'), which is assumed not to be part of the original tokens.

Example

Randomly replace at least two tokens with the replacement word 'UNK':

```
>>> from text_explainability.augmentation import TokenReplacement
>>> TokenReplacement(replacement='UNK').perturb(['perturb', 'this', 'into',
↪'multiple'],
>>>                                             n_samples=3,
>>>                                             min_changes=2)
```

Parameters

- **tokenized_instance** (*Iterable[str]*) – Tokenized instance to apply perturbations to.
- **n_samples** (*int, optional*) – Number of samples to return. Defaults to 50.
- **sequential** (*bool, optional*) – Whether to sample sequentially based on length (first length one, then two, etc.). Defaults to True.
- **contiguous** (*bool, optional*) – Whether to remove contiguous sequences of tokens (n-grams). Defaults to False.
- **min_changes** (*int, optional*) – Minimum number of tokens changes (1+). Defaults to 1.
- **max_changes** (*int, optional*) – Maximum number of tokens changed. Defaults to 10000.
- **add_background_instance** (*bool, optional*) – Add an additional instance with all tokens replaced. Defaults to False.
- **seed** (*Optional[int], optional*) – Seed for reproducibility, uses the init seed if None. Defaults to None.

Raises

ValueError – min_changes cannot be greater than max_changes.

Yields

Iterator[Sequence[Iterable[str], Iterable[int]]] –

Perturbed text instances and indices where
perturbation were applied.

Return type
`Iterator[Tuple[Iterable[str], Iterable[int]]]`

text_explainability.data.embedding module

Embed text instances into numerical vectors.

class text_explainability.data.embedding.**CountVectorizer**(***kwargs*)

Bases: *Embedder*

Embed sentences using **`sklearn.CountVectorizer`_**.

Parameters
****kwargs** – Optional arguments passed for *sklearn.CountVectorizer()* construction.

class text_explainability.data.embedding.**Embedder**(*model_fn*)

Bases: `Readable`

Embedding model base class to transform instances into vectors.

Parameters
model_fn (*Callable*) – Model that embeds instances (transforms into vectors).

embed(*instances*)

Embed instances (transform into numerical vectors).

Parameters
instances (*Union[np.ndarray, list, MemoryBucketProvider]*) – Sequence of instances.

Returns

Embedded instances (provided back into the BucketProvider if it
was originally passed as a BucketProvider).

Return type
Union[np.ndarray, MemoryBucketProvider]

class text_explainability.data.embedding.**SentenceTransformer**(*model_name='distiluse-base-multilingual-cased-v1'*, ***kwargs*)

Bases: *Embedder*

Embed sentences using the Sentence Transformers package.

By default requires and active internet connection, or provide the name of a local *model_name*.

Parameters

- **model_name** (*str, optional*) – Name of Sentence Transformer model. See https://www.sbert.net/docs/pretrained_models.html for model names. Defaults to 'distiluse-base-multilingual-cased-v1'.
- ****kwargs** – Optional arguments to be passed to *SentenceTransformer.encode()* function. See https://www.sbert.net/examples/applications/computing-embeddings/README.html

class text_explainability.data.embedding.**TfidfVectorizer**(***kwargs*)

Bases: *Embedder*

Embed sentences using `sklearn.TfidfVectorizer`_.

Parameters
****kwargs** – Optional arguments passed for *sklearn.TfidfVectorizer()* construction.

text_explainability.data.embedding.**as_2d**(*vectors*, *method='pca'*, ***kwargs*)

Summarize vectors in 2 dimensions.

Return type
ndarray

Parameters

- **vectors** (*ndarray | list | MemoryBucketProvider*) –
- **method** (*str*) –

text_explainability.data.embedding.**as_3d**(*vectors*, *method='pca'*, ***kwargs*)

Summarize vectors in 3 dimensions.

Return type
ndarray

Parameters

- **vectors** (*ndarray | list | MemoryBucketProvider*) –
- **method** (*str*) –

text_explainability.data.embedding.**as_n_dimensional**(*vectors*, *n=2*, *method='pca'*, ***kwargs*)

Summarize vectors into n dimensions.

Parameters

- **vectors** (*Union[np.ndarray, list, MemoryBucketProvider]*) – Vectors or BucketProvider with vectorized instances.
- **n** (*int, optional*) – Number of dimensions (should be low, e.g. 2 or 3). Defaults to 2.
- **method** (*str, optional*) – Method used for dimensionality reduction. Choose from ['pca', 'kernel_pca', 'incremental_pca', 'nmf', 'tsne']. Defaults to 'pca'.
- ****kwargs** – Optional arguments passed to method constructor.

Raises
ValueError – Unknown method selected.

Returns
Vectors summarized in n dimensions.

Return type
np.ndarray

text_explainability.data.sampling module

Sample an (informative) subset from the data.

class text_explainability.data.sampling.**KMedoids**(*instances*, *embedder=<class 'text_explainability.data.embedding.TfidfVectorizer'>*, *seed=0*)

Bases: *PrototypeSampler*, SeedMixin

Sampling prototypes (representative samples) based on embedding distances using k-Medoids.

Parameters

- **instances** (*MemoryBucketProvider*) – Instances to select from (e.g. training set, all instance from class 0).
- **embedder** (*Embedder, optional*) – Method to embed instances (if the *.vector* property is not yet set). Defaults to TfidfVectorizer.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

prototypes(*n=5*, *metric='cosine'*, ***kwargs*)

Select *n* prototypes (most representative samples) using k-Medoids.

Parameters

- **n** (*int, optional*) – Number of prototypes to select. Defaults to 5.
- **metrics** (*Union[str, Callable], optional*) – Distance metric used to calculate medoids (e.g. 'cosine', 'euclidean' or your own function). See *pairwise distances* for a full list. Defaults to 'cosine'.
- ****kwargs** – Optional arguments passed to k-Medoids constructor.
- **metric** (*str | Callable*) –

Returns
List of prototype instances.

Return type
Sequence[DataPoint]

class text_explainability.data.sampling.**LabelwiseKMedoids**(*instances*, *labels*, *embedder=<class 'text_explainability.data.embedding.TfidfVectorizer'>*, *seed=0*)

Bases: *LabelwisePrototypeSampler*

Select prototypes for each label based on embedding distances using k-Medoids.

Parameters

- **instances** (*MemoryBucketProvider*) – Instances to select from (e.g. training set, all instance from class 0).
- **labels** (*Union[Sequence[str], Sequence[int], LabelProvider]*) – Ground-truth or predicted labels, providing the groups (e.g. classes) in which to subdivide the instances.
- **embedder** (*Embedder, optional*) – Method to embed instances (if the *.vector* property is not yet set). Defaults to TfidfVectorizer.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

class text_explainability.data.sampling.**LabelwiseMMDCritic**(*instances*, *labels*, *embedder=<class 'text_explainability.data.embedding.TfidfVectorizer'>*, *kernel=<function rbf_kernel>*)

Bases: *LabelwisePrototypeSampler*

Select prototypes and criticisms for each label based on embedding distances using MMD-Critic.

Parameters

- **instances** (*MemoryBucketProvider*) – Instances to select from (e.g. training set, all instance from class 0).
- **labels** (*Union[Sequence[str], Sequence[int], LabelProvider]*) – Ground-truth or predicted labels, providing the groups (e.g. classes) in which to subdivide the instances.
- **embedder** (Embedder, *optional*) – Method to embed instances (if the *.vector* property is not yet set). Defaults to TfidfVectorizer.
- **kernel** (*Callable, optional*) – Kernel to calculate distances. Defaults to rbf_kernel.

criticisms(*n=5*, *regularizer=None*)

Select *n* criticisms (instances not well represented by prototypes).

Parameters

- **n** (*int, optional*) – Number of criticisms to select. Defaults to 5.
- **regularizer** (*Optional[str], optional*) – Regularization method. Choose from [None, 'logdet', 'iterative']. Defaults to None.

Raises

Exception – *MMDCritic.prototypes()* must first be run before being able to determine the criticisms.

Returns

Dictionary with labels and corresponding list of criticisms.

Return type

Dict[str, Sequence[DataPoint]]

class text_explainability.data.sampling.**LabelwisePrototypeSampler**(*sampler*, *instances*, *labels*, *embedder=<class 'text_explainability.data.embedding.TfidfVectoriz*
***kwargs*)

Bases: Readable

Apply *PrototypeSampler()* for each label.

Parameters

- **sampler** (PrototypeSampler) – Prototype sampler to construct (e.g. *KMedoids*, *MMDCritic*)
- **instances** (*MemoryBucketProvider*) – Instances to select from (e.g. training set, all instance from class 0).
- **labels** (*Union[Sequence[str], Sequence[int], LabelProvider, AbstractClassifier]*) – Ground-truth or predicted labels, providing the groups (e.g. classes) in which to subdivide the instances.
- **embedder** (Embedder, *optional*) – Method to embed instances (if the *.vector* property is not yet set). Defaults to TfidfVectorizer.

- ****kwargs** – Additional arguments passed to *_setup_instances()* constructor.

prototypes(*n=5*)

Select *n* prototypes (most representatitve instances).

Parameters
n (*int, optional*) – Number of prototypes to select. Defaults to 5.

Returns
Dictionary with labels and corresponding list of prototypes.

Return type
Dict[str, Sequence[DataPoint]]

class text_explainability.data.sampling.**MMDCritic**(*instances*, *embedder=<class 'text_explainability.data.embedding.TfidfVectorizer'>*, *kernel=<function rbf_kernel>*)

Bases: *PrototypeSampler*

Select prototypes and criticisms based on embedding distances using MMD-Critic.

Parameters

- **instances** (*MemoryBucketProvider*) – Instances to select from (e.g. training set, all instance from class 0).
- **embedder** (*Embedder, optional*) – Method to embed instances (if the *.vector* property is not yet set). Defaults to TfidfVectorizer.
- **kernel** (*Callable, optional*) – Kernel to calculate distances. Defaults to rbf_kernel.

criticisms(*n=5*, *regularizer=None*)

Select *n* criticisms (instances not well represented by prototypes), using MMD-critic implementation.

Parameters

- **n** (*int, optional*) – Number of criticisms to select. Defaults to 5.
- **regularizer** (*Optional[str], optional*) – Regularization method. Choose from [None, 'logdet', 'iterative']. Defaults to None.

Raises

- **Exception** – *MMDCritic.prototypes()* must first be run before being able to determine the criticisms.
- **ValueError** – Unknown regularizer or requested more criticisms than there are samples left.

Returns
List of criticism instances.

Return type
Sequence[DataPoint]

prototypes(*n=5*)

Select *n* prototypes (most representative instances), using MMD-critic implementation.

Parameters
n (*int, optional*) – Number of prototypes to select. Defaults to 5.

Raises
ValueError – Cannot select more instances than the total number of instances.

Returns
List of prototype instances.

Return type
Sequence[DataPoint]

`to_config()`

`class text_explainability.data.sampling.PrototypeSampler(instances, embedder=<class 'text_explainability.data.embedding.TfidfVectorizer'>)`

Bases: `Readable`

Generic class for sampling prototypes (representative samples) based on embedding distances.

Parameters

- **instances** (*MemoryBucketProvider*) – Instances to select from (e.g. training set, all instance from class 0).
- **embedder** (*Embedder, optional*) – Method to embed instances (if the *.vector* property is not yet set). Defaults to TfidfVectorizer.

`property embedded: ndarray`

`prototypes(n=5)`

Select *n* prototypes.

Parameters
n (*int, optional*) – Number of prototypes to select. Defaults to 5.

Returns
List of prototype instances.

Return type
Sequence[DataPoint]

text_explainability.data.weights module

Functions for computing weights for training models (e.g. based on distance to original sample).

`text_explainability.data.weights.exponential_kernel(d, kw)`

Exponential kernel.

`text_explainability.data.weights.pairwise_distances(a, b, metric='cosine', multiply=100)`

Pairwise distancens between two vectors.

Parameters

- **a** – Vector A.
- **b** – Vector B.
- **metric** (*str, optional*) – Metric name (e.g. 'cosine', 'euclidean'). Defaults to 'cosine'.
- **multiply** (*int, optional*) – Multiply the final distance value by this constant. Defaults to 100.

Returns
Pairwise distances.

Return type
np.ndarray

text_explainability.data.weights.**rbf_kernel**(*X*, *gamma=None*)

Radial basis function (RBF) kernel.

5.4.2 text_explainability.generation

Feature selection and local/global surrogate model generation.

Submodules:

text_explainability.generation.feature_selection module

Feature selection methods for limiting explanation length.

class text_explainability.generation.feature_selection.**FeatureSelector**(*model=None*)

Bases: Readable

[summary]

Parameters

model (*Optional[LinearSurrogate], optional*) – Linear surrogate used to calculate feature importance scores. Defaults to None.

select(**args*, ***kwargs*)

Alias for *FeatureSelector().__call__()*

text_explainability.generation.return_types module

General return types for global/local explanations.

class text_explainability.generation.return_types.**BaseReturnType**(*labels=None*, *labelset=None*, *original_scores=None*, *type='base'*, *subtype=None*, *callargs=None*, ***kwargs*)

Bases: MetaInfo

Base return type.

Parameters

- **labels** (*Optional[Sequence[int]], optional*) – Label indices to include, if none provided defaults to 'all'. Defaults to None.
- **labelset** (*Optional[Sequence[str]], optional*) – Lookup for label names. Defaults to None.
- **original_scores** (*Optional[Sequence[float]], optional*) – Probability scores for each class. Defaults to None.
- **type** (*Optional[str]*) – Type description. Defaults to 'base'.
- **subtype** (*Optional[str], optional*) – Subtype description. Defaults to None.
- **callargs** (*Optional[dict], optional*) – Call arguments for reproducibility. Defaults to None.
- ****kwargs** – Optional meta descriptors.

label_by_index(*idx*)

Access label name by index, if *labelset* is set.

Parameters
idx (*int*) – Lookup index.

Raises
IndexError – *labelset* is set but the element index is not in *labelset* (index out of bounds).

Returns
Label name (if available) else index.

Return type
Union[str, int]

property labels

Get labels property.

property labelset

Get label names property.

property original_scores

class text_explainability.generation.return_types.**FeatureAttribution**(*provider*, *scores*, *used_features=None*, *scores_stddev=None*, *base_score=None*, *labels=None*, *labelset=None*, *original_scores=None*, *original_id=None*, *sampled=False*, *type='local_explanation'*, *sub-type='feature_attribution'*, *callargs=None*, ***kwargs*)

Bases: *ReadableDataMixin*, *FeatureList*, *LocalDataExplanation*

Create a *FeatureList* with additional information saved.

The additional information contains the possibility to add standard deviations, base scores, and the sampled or generated instances used to calculate these scores.

Parameters

- **provider** (*InstanceProvider*) – Sampled or generated data, including original instance.
- **scores** (*Sequence[float]*) – Scores corresponding to the selected features.
- **used_features** (*Optional[Union[Sequence[str], Sequence[int]]]*) – Selected features for the explanation label. Defaults to None.
- **scores_stddev** (*Sequence[float], optional*) – Standard deviation of each feature attribution score. Defaults to None.
- **base_score** (*float, optional*) – Base score, to which all scores are relative. Defaults to None.
- **labels** (*Optional[Sequence[int]], optional*) – Labels for outputs (e.g. classes). Defaults to None.

- **labelset** (*Optional[Sequence[str]], optional*) – Label names corresponding to labels. Defaults to None.
- **original_scores** (*Optional[Sequence[float]], optional*) – Probability scores for each class. Defaults to None.
- **original_id** (*Optional[LT], optional*) – ID of original instance; picks first if None. Defaults to None.
- **sampled** (*bool, optional*) – Whether the data in the provider was sampled (True) or generated (False). Defaults to False.
- **type** (*Optional[str]*) – Type description. Defaults to 'base'.
- **subtype** (*Optional[str], optional*) – Subtype description. Defaults to None.
- **callargs** (*Optional[dict], optional*) – Call arguments for reproducibility. Defaults to None.
- ****kwargs** – Optional meta descriptors.

property content

property scores

Saved feature attribution scores.

class text_explainability.generation.return_types.**FeatureList**(*used_features*, *scores*, *labels=None*, *labelset=None*, *original_scores=None*, *type='global_explanation'*, *subtype='feature_list'*, *callargs=None*, ***kwargs*)

Bases: *BaseReturnType*, *UsedFeaturesMixin*

Save scores per feature, grouped per label.

Examples of scores are feature importance scores, or counts of features in a dataset.

Parameters

- **used_features** (*Union[Sequence[str], Sequence[int]]*) – Used features per label.
- **scores** (*Union[Sequence[int], Sequence[float]]*) – Scores per label.
- **labels** (*Optional[Sequence[int]], optional*) – Label indices to include, if none provided defaults to 'all'. Defaults to None.
- **labelset** (*Optional[Sequence[str]], optional*) – Lookup for label names. Defaults to None.
- **original_scores** (*Optional[Sequence[float]], optional*) – Probability scores for each class. Defaults to None.
- **type** (*Optional[str]*) – Type description. Defaults to 'explanation'.
- **subtype** (*Optional[str], optional*) – Subtype description. Defaults to 'feature_list'.
- **callargs** (*Optional[dict], optional*) – Call arguments for reproducibility. Defaults to None.
- ****kwargs** – Optional meta descriptors.

property content

get_raw_scores(*normalize=False*)

Get saved scores per label as *np.ndarray*.

Parameters
normalize (*bool, optional*) – Normalize scores (ensure they sum to one). Defaults to False.

Returns
Scores.

Return type
np.ndarray

get_scores(*normalize=False*)

Get scores per label.

Parameters
normalize (*bool, optional*) – Whether to normalize the scores (sum to one). Defaults to False.

Returns

Scores per label, if no *labelset*
is not set, defaults to 'all'

Return type
Dict[Union[str, int], Tuple[Union[str, int], Union[float, int]]]

property scores

Saved scores (e.g. feature importance).

class text_explainability.generation.return_types.**Instances**(*instances*, *original_scores=None*, *type='global_explanation'*, *subtype='prototypes'*, *callargs=None*, ***kwargs*)

Bases: *BaseReturnType*

Parameters

- **original_scores** (*Sequence[float] | None*) –
- **type** (*str | None*) –
- **subtype** (*str | None*) –
- **callargs** (*dict | None*) –

property content

class text_explainability.generation.return_types.**LocalDataExplanation**(*provider*, *original_id=None*, *sampled=False*)

Bases: object

Save the sampled/generated instances used to determine an explanation.

Parameters

- **provider** (*InstanceProvider*) – Sampled or generated data, including original instance.
- **original_id** (*Optional[LT], optional*) – ID of original instance; picks first if None. Defaults to None.

- **sampled** (*bool, optional*) – Whether the data in the provider was sampled (True) or generated (False). Defaults to False.

property neighborhood_instances

Instances in the neighborhood (either sampled or perturbed).

property original_instance

The instance for which the feature attribution scores were calculated.

property perturbed_instances

Perturbed versions of the original instance, if *sampled=False* during initialization.

property sampled_instances

Sampled instances, if *sampled=True* during initialization.

class text_explainability.generation.return_types.**ReadableDataMixin**

Bases: object

property used_features

Names of features of the original instance.

class text_explainability.generation.return_types.**Rules**(*provider*, *rules*, *used_features=None*, *labels=None*, *labelset=None*, *original_scores=None*, *original_id=None*, *sampled=False*, *contrastive=False*, *type='local_explanation'*, *subtype='rules'*, *callargs=None*, ***kwargs*)

Bases: *ReadableDataMixin*, *UsedFeaturesMixin*, *BaseReturnType*, *LocalDataExplanation*

Rule-based return type.

Parameters

- **provider** (*InstanceProvider*) – Sampled or generated data, including original instance.
- **rules** (*Union[Sequence[str], TreeSurrogate, RuleSurrogate]*) – Rules applicable.
- **used_features** (*Optional[Union[Sequence[str], Sequence[int]]]*) – Used features per label. Defaults to None.
- **labels** (*Optional[Sequence[int]], optional*) – Label indices to include, if none provided defaults to 'all'. Defaults to None.
- **labelset** (*Optional[Sequence[str]], optional*) – Lookup for label names. Defaults to None.
- **original_scores** (*Optional[Sequence[float]], optional*) – Probability scores for each class. Defaults to None.
- **original_id** (*Optional[LT], optional*) – ID of original instance; picks first if None. Defaults to None.
- **sampled** (*bool, optional*) – Whether the data in the provider was sampled (True) or generated (False). Defaults to False.
- **contrastive** (*bool, optional*) – If the rules are contrastive. Defaults to False.
- **type** (*Optional[str]*) – Type description. Defaults to 'base'.
- **subtype** (*Optional[str], optional*) – Subtype description. Defaults to None.

- **callargs** (*Optional[dict], optional*) – Call arguments for reproducibility. Defaults to None.
- ****kwargs** – Optional meta descriptors.

property content

property rules

class text_explainability.generation.return_types.**UsedFeaturesMixin**

Bases: object

property used_features

Get used features property.

text_explainability.generation.surrogate module

Wrappers for surrogate models, used for local/global explanations.

class text_explainability.generation.surrogate.**BaseSurrogate**(*model*)

Bases: Readable

Base wrapper around a *sklearn* predictor.

Parameters
model – *sklearn* model to wrap.

property feature_importances

Surrogate model feature importances.

fit(*X*, *y*, *weights=None*)

Fit *sklearn* model.

Parameters

- **X** – Training data.
- **y** – Target labels corresponding to training data.
- **weights** (*optional*) – Relative weight of each instance. Defaults to None.

Returns
Fitted model.

Return type
BaseSurrogate

predict(*X*)

Predict a batch of instances.

Parameters
X – Instances.

Returns
Predicted instances.

Return type
np.ndarray

class text_explainability.generation.surrogate.**LinearSurrogate**(*model*)

Bases: *BaseSurrogate*

Wrapper around sklearn linear model for usage in local/global surrogate models.

alpha_reset()

Reset model alpha to the initial value.

alpha_zero()

Reset model alpha to zero.

property coef

Model coefficients.

property feature_importances

Model feature importances (same as *LinearSurrogate.coef*).

property fit_intercept

Model fit intercept.

property intercept

Model intercept.

score(*X*, *y*, *weights=None*)

Score instances.

property seed

Model seed.

class text_explainability.generation.surrogate.**RuleSurrogate**(*model*)

Bases: *BaseSurrogate*

Wrapper around **`SkopeRules`_** model for usage in local/global surrogate models.

_SkopeRules:
https://github.com/scikit-learn-contrib/skope-rules

Base wrapper around a *sklearn* predictor.

Parameters
model – *sklearn* model to wrap.

property feature_names

property rules

score_top_rules(*X*)

class text_explainability.generation.surrogate.**TreeSurrogate**(*model*)

Bases: *BaseSurrogate*

Wrapper around sklearn tree model for usage in local/global surrogate models.

Base wrapper around a *sklearn* predictor.

Parameters
model – *sklearn* model to wrap.

property classes

decision_path(*X*)

property feature_importances

features(*tokens_to_map=None*)

Parameters
tokens_to_map (*Sequence[str] | None*) –

leaf_classes()

property max_rule_size

property rules

to_rules(*classes=None*, *features=None*, *grouped=False*)

Parameters
grouped (*bool*) –

text_explainability.generation.target_encoding module

Encode targets into binary labels for contrastive explanation.

class text_explainability.generation.target_encoding.**FactFoilEncoder**(*foil*, *labelset=None*)

Bases: *TargetEncoder*

Encode target into foil (target class) fact (non-foil class).

Parameters

- **foil** (*int*) – Index of target class.
- **labelset** (*Optional[Sequence[str]], optional*) – Names of labels. Defaults to None.

encode(*y*)

Encode a single instance into foil (0) or not foil (1).

classmethod from_str(*label*, *labelset*)

Instantiate FactFoilEncoder with a string as foil.

Parameters

- **label** (*str*) – Foil (expected outcome) label.
- **labelset** (*Union[AbstractClassifier, Sequence[str]]*) – Labelset containing the foil.

Returns
Initialized FactFoilEncoder.

Return type
FactFoilEncoder

class text_explainability.generation.target_encoding.**TargetEncoder**(*labels=None*)

Bases: object

Encode model predictions based on encoding rule.

Parameters
labels (*Optional[Union[Sequence[str], AbstractClassifier]], optional*) – Labelset for mapping labels onto. Defaults to None.

encode(*y*)

Encode a single instance.

get_label(*y*, *proba_to_labels=True*, *label_to_index=True*)

Get prediction label as probability, string or class index.

Parameters

- **y** – Predictions with optional indices.
- **proba_to_labels** (*bool, optional*) – Whether to convert probability to highest scoring class. Defaults to True.
- **label_to_index** (*bool, optional*) – Convert string to index in labelset. Defaults to True.

Returns

Label names (if label_to_index is False) or label indices (otherwise).

Return type

Union[List[int], List[str]]

property labelset

Labels.

5.4.3 text_explainability.global_explanation

Global explanations explain the whole dataset or model behavior on that dataset.

class text_explainability.global_explanation.**GlobalExplanation**(*provider*, *seed=0*)

Bases: Readable, SeedMixin

Generic wrapper from global explanations (explain whole dataset or model).

Parameters

- **provider** (*InstanceProvider[TextInstance, Any, str, Any, str]*) – Dataset to perform explanation on.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

explain(**args*, ***kwargs*)

get_data()

Easy access to data.

Returns

Easily accessible dataset.

Return type

InstanceProvider

get_instances_labels(*model*, *labelprovider*, *explain_model=True*)

Get corresponding labels of dataset inputs, either from the original data or
according to the predict function.

Parameters

- **model** (*Optional[AbstractClassifier]*) – Model to perform predictions with.
- **labelprovider** (*Optional[LabelProvider]*) – Ground-truth labels.

- **explain_model** (*bool, optional*) – Whether to explain using the *model* labels (True) or *labelprovider* labels (False). Defaults to True.

Raises
ValueError – if explain_model = True provide a model, and if False provide a labelprovider.

Returns
Instances and corresponding labels

Return type
Tuple[InstanceProvider, np.ndarray]

predict(*model*)

Apply predict function of model to data.

Parameters
model (*AbstractClassifier*) – Model to apply predictions with.

Returns
Labels for dataset according to model.

Return type
Union[Sequence[FrozenSet[str]], np.ndarray]

class text_explainability.global_explanation.**KMedoids**(**args*, ***kwargs*)

Bases: *PrototypeWrapper*

Get prototypes using method k-Medoids.

For arguments see *text_explainability.data.sampling.KMedoids*.

class text_explainability.global_explanation.**LabelwiseKMedoids**(**args*, ***kwargs*)

Bases: *PrototypeWrapper*

class text_explainability.global_explanation.**LabelwiseMMDCritic**(**args*, ***kwargs*)

Bases: *PrototypeCriticismWrapper*

class text_explainability.global_explanation.**MMDCritic**(**args*, ***kwargs*)

Bases: *PrototypeCriticismWrapper*

class text_explainability.global_explanation.**PrototypeCriticismWrapper**(*prototype_sampler*, **args*, *method=None*, *subtype='prototypes_&_criticisms'*, ***kwargs*)

Bases: *PrototypeWrapper*

Parameters

- **prototype_sampler** (PrototypeSampler) –
- **method** (*str | None*) –
- **subtype** (*str*) –

criticisms(**args*, ***kwargs*)

Return type
Instances

class text_explainability.global_explanation.**PrototypeWrapper**(*prototype_sampler*, **args*, *method=None*, *subtype='prototypes'*, ***kwargs*)

Bases: object

Parameters

- **prototype_sampler** (PrototypeSampler) –
- **method** (*str | None*) –
- **subtype** (*str*) –

prototypes(**args*, ***kwargs*)

Return type
Instances

class text_explainability.global_explanation.**TokenFrequency**(*provider*, *seed=0*)

Bases: *GlobalExplanation*

Generic wrapper from global explanations (explain whole dataset or model).

Parameters

- **provider** (*InstanceProvider[TextInstance, Any, str, Any, str]*) – Dataset to perform explanation on.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

class text_explainability.global_explanation.**TokenInformation**(*provider*, *seed=0*)

Bases: *GlobalExplanation*

Generic wrapper from global explanations (explain whole dataset or model).

Parameters

- **provider** (*InstanceProvider[TextInstance, Any, str, Any, str]*) – Dataset to perform explanation on.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

5.4.4 text_explainability.local_explanation

Local explanations explain why a model made a prediction for a single instance.

class text_explainability.local_explanation.**Anchor**(*env=None*, *labelset=None*, *augmenter=None*, *seed=0*)

Bases: *LocalExplanation*

Parameters

- **env** (*AbstractEnvironment | None*) –
- **labelset** (*Sequence[str] | LabelProvider | None*) –
- **augmenter** (LocalTokenPertubator | *None*) –
- **seed** (*int*) –

static beam_search(*provider*, *perturbed*, *model*, *beam_size=1*, *min_confidence=0.95*, *delta=0.05*, *epsilon=0.1*, *max_anchor_size=None*, *batch_size=20*)

Parameters

- **perturbed** (*ndarray*) –
- **beam_size** (*int*) –
- **min_confidence** (*float*) –
- **delta** (*float*) –
- **epsilon** (*float*) –
- **max_anchor_size** (*int | None*) –
- **batch_size** (*int*) –

best_candidate()

static dlow_bernoulli(*p*, *level*)

generate_candidates()

static kl_bernoulli(*p*, *q*)

class text_explainability.local_explanation.**BayLIME**(*env=None*, *local_model=None*, *kernel=None*, *kernel_width=25*, *augmenter=None*, *labelset=None*, *seed=0*)

Bases: *LIME*

Bayesian Local Interpretable Model-Agnostic Explanations (BayLIME).

Bayesian modification of LIME, which can exploit prior knowledge and Bayesian reasoning to improve the consistency in repeated explanations of a single prediction and the robustness to kernel settings.

Parameters

- **env** (*Optional[AbstractEnvironment]*) – Environment to save local perturbations in. Defaults to None.
- **local_model** (*Optional[LinearSurrogate], optional*) – Local Bayesian linear model. If None defaults to Bayesian Ridge regression. Defaults to None.
- **kernel** (*Optional[Callable], optional*) – Kernel to determine similarity of perturbed instances to original instance. Defaults to None.
- **kernel_width** (*Union[int, float], optional*) – Hyperparameter for similarity function of kernel. Defaults to 25.
- **augmenter** (*Optional[LocalTokenPertubator], optional*) – Function to augment data with perturbations, to generate neighborhood data. Defaults to None.
- **labelset** (*Optional[Union[Sequence[str], LabelProvider]], optional*) – Sequence of label names or LabelProvider containing named labels. When not supplied, it uses identifiers for labels. Defaults to None.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

class text_explainability.local_explanation.**FactFoilMixin**

Bases: object

to_fact_foil(*y*, *labelset*, *foil_fn*)

Parameters

foil_fn (FactFoilEncoder | *int* | *str*) –

class text_explainability.local_explanation.**FoilTree**(*env=None*, *labelset=None*, *augmenter=None*, *local_model=None*, *kernel=None*, *kernel_width=25*, *explanation_type='multiclass'*, *seed=0*)

Bases: *FactFoilMixin*, *LocalExplanation*, *WeightedExplanation*

Parameters

- **env** (*AbstractEnvironment* | *None*) –
- **labelset** (*Sequence[str]* | *LabelProvider* | *None*) –
- **augmenter** (LocalTokenPertubator | *None*) –
- **local_model** (TreeSurrogate | *None*) –
- **kernel** (*Callable* | *None*) –
- **kernel_width** (*int* | *float*) –
- **explanation_type** (*str*) –
- **seed** (*int*) –

class text_explainability.local_explanation.**KernelSHAP**(*env=None*, *labelset=None*, *augmenter=None*, *seed=0*)

Bases: *LocalExplanation*

Calculates Shapley values for an instance to explain, assuming the model is a black-box.

Calculates Shapley values (local, additive feature attribution scores) for an instance to explain, by calculating the average contribution of changing combinations of feature values.

Parameters

- **env** (*Optional[AbstractEnvironment], optional*) – Environment to save local perturbations in. Defaults to None.
- **augmenter** (*Optional[*LocalTokenPertubator*], optional*) – Function to augment data with perturbations, to generate neighborhood data. Defaults to None.
- **labelset** (*Optional[Union[Sequence[str], LabelProvider]], optional*) – Sequence of label names or LabelProvider containing named labels. When not supplied, it uses identifiers for labels. Defaults to None.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

static select_features(*X*, *y*, *default_features=1*, *l1_reg='auto'*)

Select features for data X and corresponding output y.

Parameters

- **X** (*np.ndarray*) – Input data.
- **y** (*np.ndarray*) – Prediction / ground-truth value for X.
- **default_features** (*int, optional*) – Default number of features, when returning all features. Defaults to 1.

- **l1_reg** (*Union[int, float, str], optional*) – Method for regularization, either *auto*, *n_features({int})*,
- **{int}** –
- **{float}** –
- **'auto'.** (*aic or bic. Defaults to*) –

Raises
Exception – Unknown value for *l1_reg*

Returns
Feature indices to include.

Return type
np.ndarray

class text_explainability.local_explanation.**LIME**(*env=None*, *local_model=None*, *kernel=None*, *kernel_width=25*, *augmenter=None*, *labelset=None*, *seed=0*)

Bases: *LocalExplanation*, *WeightedExplanation*

Local Interpretable Model-Agnostic Explanations (LIME).

Implementation of local linear surrogate model on (weighted) perturbed text data, to get feature attribution scores for an example instance.

Parameters

- **env** (*Optional[AbstractEnvironment]*) – Environment to save local perturbations in. Defaults to None.
- **local_model** (*Optional[LinearSurrogate], optional*) – Local linear model. If None defaults to Ridge regression with alpha 1.0. Defaults to None.
- **kernel** (*Optional[Callable], optional*) – Kernel to determine similarity of perturbed instances to original instance. Defaults to None.
- **kernel_width** (*Union[int, float], optional*) – Hyperparameter for similarity function of kernel. Defaults to 25.
- **augmenter** (*Optional[LocalTokenPertubator], optional*) – Function to augment data with perturbations, to generate neighborhood data. Defaults to None.
- **labelset** (*Optional[Union[Sequence[str], LabelProvider]], optional*) – Sequence of label names or LabelProvider containing named labels. When not supplied, it uses identifiers for labels. Defaults to None.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

class text_explainability.local_explanation.**LocalExplanation**(*env=None*, *augmenter=None*, *labelset=None*, *seed=0*)

Bases: `Readable`, `SeedMixin`

Generate explanation for a single decision.

Parameters

- **env** (*Optional[AbstractEnvironment], optional*) – Environment to save local perturbations in. Defaults to None.
- **augmenter** (*Optional[LocalTokenPertubator], optional*) – Function to augment data with perturbations, to generate neighborhood data. Defaults to None.

- **labelset** (*Optional[Union[Sequence[str], LabelProvider]], optional*) – Sequence of label names or LabelProvider containing named labels. When not supplied, it uses identifiers for labels. Defaults to None.
- **seed** (*int, optional*) – Seed for reproducibility. Defaults to 0.

augment_sample(*sample*, *model*, *sequential=False*, *contiguous=False*, *n_samples=50*, *add_background_instance=False*, *predict=True*, *avoid_proba=False*, *seed=None*, ***kwargs*)

Augment a single sample to generate neighborhood data.

Parameters

- **sample** (*TextInstance*) – Instance to perturb.
- **model** (*AbstractClassifier*) – Model to provide predictions for neighborhood data.
- **sequential** (*bool, optional*) – Whether to sequentially sample based on length (first length 1, then 2, ...). Defaults to False.
- **contiguous** (*bool, optional*) – Whether to apply perturbations on contiguous stretches of text. Defaults to False.
- **n_samples** (*int, optional*) – Number of neighborhood samples to generate. Defaults to 50.
- **add_background_instance** (*bool, optional*) – Add an additional instance with all tokens replaced. Defaults to False.
- **predict** (*bool, optional*) – Defaults to True.
- **avoid_proba** (*bool, optional*) – Model predictions als labels (True) or probabilities when available (False). Defaults to False.
- **seed** (*Optional[int], optional*) – Seed for reproducibility, uses the init seed if None. Defaults to None.

Returns

Provider, how instances were perturbed and optionally the corresponding predictions for each instance.

Return type

Union[Tuple[InstanceProvider, np.ndarray], Tuple[InstanceProvider, np.ndarray, np.ndarray, np.ndarray]]

explain(**args*, ***kwargs*)

class text_explainability.local_explanation.**LocalRules**(*env=None*, *labelset=None*, *augmenter=None*, *local_model=None*, *kernel=None*, *kernel_width=25*, *explanation_type='multiclass'*, *seed=0*)

Bases: *FactFoilMixin*, *LocalExplanation*, *WeightedExplanation*

Parameters

- **env** (*AbstractEnvironment | None*) –
- **labelset** (*Sequence[str] | LabelProvider | None*) –
- **augmenter** (*LocalTokenPertubator | None*) –
- **local_model** (*RuleSurrogate | None*) –
- **kernel** (*Callable | None*) –

- **kernel_width** (*int | float*) –
- **explanation_type** (*str*) –
- **seed** (*int*) –

class text_explainability.local_explanation.**LocalTree**(*env=None*, *labelset=None*, *augmenter=None*, *local_model=None*, *kernel=None*, *kernel_width=25*, *explanation_type='multiclass'*, *seed=0*)

Bases: *LocalExplanation*, *WeightedExplanation*

Parameters

- **env** (*AbstractEnvironment | None*) –
- **labelset** (*Sequence[str] | LabelProvider | None*) –
- **augmenter** (*LocalTokenPertubator | None*) –
- **local_model** (*TreeSurrogate | None*) –
- **kernel** (*Callable | None*) –
- **kernel_width** (*int | float*) –
- **explanation_type** (*str*) –
- **seed** (*int*) –

class text_explainability.local_explanation.**WeightedExplanation**(*kernel=None*, *kernel_width=25*)

Bases: `object`

Add weights to neighborhood data.

Parameters

- **kernel** (*Optional[Callable], optional*) – Kernel to determine similarity of perturbed instances to original instance (if set to None defaults to *data.weights.exponential_kernel*). Defaults to None.
- **kernel_width** (*Union[int, float], optional*) – Hyperparameter for similarity function of kernel. Defaults to 25.

weigh_samples(*a*, *b=None*, *metric='cosine'*)

text_explainability.local_explanation.**default_env**(*env=None*)

If no environment is supplied, an empty Enviroment is created for text data.

Parameters
env (*Optional[AbstractEnvironment], optional*) – If a environment is supplied, it is used, otherwise.

Returns
The default/supplied environment.

Return type
AbstractEnvironment

5.4.5 text_explainability.ui

Extensions to *genbase.ui*.

Submodules:

text_explainability.ui.notebook module

Extension of *genbase.ui.notebook* for custom rendering of `text_explainability.

class text_explainability.ui.notebook.**Render**(**configs*)

Bases: Render

format_title(*title*, *h='h1'*, ***renderargs*)

Return type
str

Parameters

- **title** (*str*) –
- **h** (*str*) –

get_renderer(*meta*)

Parameters
meta (*dict*) –

render_subtitle(*meta*, *content*, ***renderargs*)

Return type
str

Parameters
meta (*dict*) –

text_explainability.ui.notebook.**default_renderer**(*meta*, *content*, ***renderargs*)

Default renderer fallback.

Return type
str

Parameters

- **meta** (*dict*) –
- **content** (*dict*) –

text_explainability.ui.notebook.**feature_attribution_renderer**(*meta*, *content*, ***renderargs*)

Render feature attribution return types.

Return type
str

Parameters
meta (*dict*) –

text_explainability.ui.notebook.**featurelist_renderer**(*meta*, *content*, *first_element='token'*, *second_element='frequency'*, *vertical=False*, *sorted=True*, ***renderargs*)

Render token information/frequency return types.

Return type
str

Parameters

- **meta** (*dict*) –
- **content** (*dict*) –
- **first_element** (*str*) –
- **second_element** (*str*) –
- **vertical** (*bool*) –
- **sorted** (*bool*) –

text_explainability.ui.notebook.**frequency_renderer**(*meta*, *content*, ***renderargs*)

Render token_frequency return type.

Return type
str

Parameters

- **meta** (*dict*) –
- **content** (*dict*) –

text_explainability.ui.notebook.**get_meta_descriptors**(*meta*)

Get type, subtype & method from *meta*.

Parameters
meta (*dict*) – [description]

Returns
type, subtype, method

Return type
Tuple[str]

text_explainability.ui.notebook.**information_renderer**(*meta*, *content*, ***renderargs*)

Render token_information return type.

Return type
str

Parameters

- **meta** (*dict*) –
- **content** (*dict*) –

text_explainability.ui.notebook.**original_scores_renderer**(*original_scores*, ***renderargs*)

Render predicted output scores of model on an instance.

Return type
str

Parameters
original_scores (*dict*) –

text_explainability.ui.notebook.**plotly_fallback**(*function*)

Return a graphics renderer, with fallback if plotly is not available.

text_explainability.ui.notebook.**prototype_renderer**(*meta*, *content*, ***renderargs*)

Render prototypes return type.

Return type
str

Parameters

- **meta** (*dict*) –
- **content** (*dict*) –

text_explainability.ui.notebook.**rules_renderer**(*meta*, *content*, ***renderargs*)

Render a set of rules from rule return types.

Return type
str

Parameters

- **meta** (*dict*) –
- **content** (*dict*) –

Submodules:

5.4.6 text_explainability.decorators module

Function decorators to ensure functions are fool-proof en readable.

text_explainability.decorators.**text_instance**(*func=None*, *, *tokenize=False*)

Decorator to convert an accidentally passed string to a TextInstance.

Parameters
tokenize (*bool*) –

5.4.7 text_explainability.model module

Model handling fuctions.

text_explainability.model.**import_model**(*model*, *environment=None*, *train='train'*, *label_map=None*)

Import a model from file or from a Python object.

Examples

Make a scikit-learn text classifier and train it on SST2

```
>>> from genbase import import_data, import_model
>>> from datasets import load_dataset
>>> ds = import_data(load_dataset('glue', 'sst2'), data_cols='sentence', label_cols=
↪'label')
>>> from sklearn.pipeline import Pipeline
>>> from sklearn.naive_bayes import MultinomialNB
>>> from sklearn.feature_extraction.text import TfidfVectorizer
>>> pipeline = Pipeline([('tfidf', TfidfVectorizer()),
...                      ('clf', MultinomialNB())])
>>> import_model(pipeline, ds, train='train')
```

Load a pretrained ONNX model downloaded from
https://github.com/mpbron/instancelib-onnx/blob/main/example_models/data-model.onnx

```
>>> from genbase import import_model
>>> import_model('data-model.onnx', label_map={0: 'Bedrijfsnieuws', 1: 'Games', 2:
↪'Smartphones'})
```

Parameters

- **model** – Model or path to model to import.
- **environment** (*Optional[Environment], optional*) – Environment corresponding to model (with dataset and ground-truth labels), used for importing models and/or training them.
- **train** (*Union[int, float, str, InstanceProvider], optional*) – Train split size, name in environment or provider. Defaults to 'train'.
- **label_map** (*Optional[Dict[LT, LT]], optional*) – Conversion of label IDs to named labels. Defaults to None.

Raises

- **ImportError** – Unable to import model or file.
- **NotImplementedError** – Type of model is not yet supported.

Returns
Instancelib wrapped model.

Return type
AbstractClassifier

5.4.8 text_explainability.utils module

Utility functions.

text_explainability.utils.**binarize**(*X*)

Turn an *np.ndarray* into 0s and 1s.

Return type
ndarray

Parameters
X (*ndarray*) –

text_explainability.utils.**character_detokenizer**(*input*)

Convert a list of characters into a string.

Return type
str

Parameters
input (*Iterable[str]*) –

text_explainability.utils.**character_tokenizer**(*input*)

Convert a string into a list of characters.

Return type
Sequence[str]

Parameters
input (*str*) –

text_explainability.utils.**default_detokenizer**(*input*)

Simple regex detokenizer, ideally resulting in *i = detokenizer(tokenizer(i))*.

Return type
str

Parameters
input (*Iterable[str]*) –

text_explainability.utils.**default_tokenizer**(*input*, *exclude_curly_brackets=False*)

Simple regex tokenizer.

Return type
Sequence[str]

Parameters

- **input** (*str*) –
- **exclude_curly_brackets** (*bool*) –

text_explainability.utils.**word_detokenizer**(*input*)

Simple regex detokenizer, ideally resulting in *i = detokenizer(tokenizer(i))*.

Return type
str

Parameters
input (*Iterable[str]*) –

text_explainability.utils.**word_tokenizer**(*input*, *exclude_curly_brackets=False*)

Simple regex tokenizer.

Return type
Sequence[str]

Parameters

- **input** (*str*) –
- **exclude_curly_brackets** (*bool*) –

5.5 Changelog

All notable changes to `text_explainability` will be documented in this file.

The format is based on Keep a Changelog, and this project adheres to Semantic Versioning.

5.5.1 Unreleased

5.5.2 0.7.0 - 2023-02-22

Added

- BayLIME for Bayesian local explanations (extension of LIME with more consistency across runs)

5.5.3 0.6.7 - 2023-02-21

Added

- Local model explanations now can be fully seeded

Changed

- Updated rendering of rule-based return type (tree surrogates and rule surrogates)

5.5.4 0.6.6 - 2023-02-02

Fixed

- Bugfix where tokens are not properly filtered in global explaanations (`TokenFrequency` and `TokenInformation`)

5.5.5 0.6.5 - 2022-07-19

Added

- Show predicted scores for each class in feature attribution
- First version of rule rendering

Fixed

- Rendering of labelwise prototypes

5.5.6 0.6.4 - 2022-07-08

Fixed

- Bugfix that returned generator for local neighborhood data generation (`explabox` issue #2)

5.5.7 0.6.3 - 2022-05-30

Added

- More complex neighborhood data augmentation
- Rule return type

Changed

- Non-duplicate generation of neighborhood data
- Replaced `skoperules` with `imodels` for future compatibility

Fixed

- Fallback to `default_tokenizer()` for `sklearn.CountVectorizer` and `sklearn.TfidfVectorizer`
- Bugfixes in feature selection when `n_features >= n_samples`

5.5.8 0.6.2 - 2022-04-06

Changed

- Requires `genbase>=0.2.8`
- Requires `scikit-learn>=1.0.2`

Fixed

- Bugfixes in `MMDCritic`

5.5.9 0.6.1 - 2022-03-16

Changed

- Requires `genbase>=0.2.4`
- Requires `instancelib>=0.4.3.1`

Fixed

- Typo fixes and small bugs

5.5.10 0.6.0 - 2022-03-04

Added

- More tests to increase test coverage

Changed

- Requires `genbase>=0.2.2`
- Renamed `pyproject.toml` to `.portray` to avoid build errors
- Made `fastcountvectorizer` optional

Fixed

- Bugfix when installing package, by moving `__version___` to `/_version.py`

5.5.11 0.5.8 - 2021-12-02

Added

- `get_meta_descriptors()` to get type/subtype/method from meta

Changed

- Requires `genbase>=0.1.13`

Fixed

- Bugfix in `MMDCritic` for prototype indices
- Bugfix in `TRANSLATION_DICT`

5.5.12 0.5.7 - 2021-12-01

Added

- Return type for `Instances`
- Rendering of `Instances`
- Rendering of `FeatureList`
- Extended rendering of `render_subtitle()`

Changed

- Ensure `MMDCritic/KMedoids` returns `Instances`
- Requires `genbase>=0.1.11`

Fixed

- Bugfix of instance identifier in `PrototypeSampler._select_from_provider()`

5.5.13 0.5.6 - 2021-11-30

Added

- Added meta information with `genbase.MetaInfo`
- Rendering with and extended `genbase.Render`

Changed

- Moved `Readable` to `genbase`
- Use `genbase.SeedMixin` for seeds
- Use `genbase.internationalization` for internationalization
- Requires `genbase>=0.1.10`

Fixed

- Selected features are in order in `FeatureList`

5.5.14 0.5.5 - 2021-11-17

Changed

- `TokenFrequency` and `TokenInformation` now use the faster `fastcountvectorizer` implementation

Fixed

- Bugfixes in return type of `TokenFrequency` and `TokenInformation`

5.5.15 0.5.4 - 2021-10-27

Fixed

- Bugfixes in local explanation return types

5.5.16 0.5.3 - 2021-10-19

Fixed

- Made `alpha` optional in `LinearSurrogate`
- Added `skope-rules` dependency to `setup.py`

5.5.17 0.5.2 - 2021-10-05

Fixed

- Hotfix in `FeatureSelector._information_criterion()`

5.5.18 0.5.1 - 2021-10-05

Added

- Added `text_explainability.data.from_list`

Changed

- Added example results in README.md

Fixed

- Added new methods and classes to `__init__.py`

5.5.19 0.5.0 - 2021-10-04

Added

- Security testing with bandit
- More locale translations
- Wrappers around `instancelib` in `text_explainability.data` and `text_explainability.model`

Changed

- Extended description in README.md
- Changed example usage to fit workflow changes
- Logo link in README.md

Fixed

- Bugfixes in MMDCritic
- Bugfixes in KernelSHAP

5.5.20 0.4.6 - 2021-10-02

Added

- External documentation
- Documentation styling
- Citation information

Changed

- Word tokenizer can now combine tokens in curly bracket when setting `exclude_curly_brackets=True`

5.5.21 0.4.5 - 2021-09-24

Added

- Decorator to allow strings to be converted into TextInstances
- Decorator to ensure TextInstances are tokenized when required

Fixed

- Typing fixes

5.5.22 0.4.4 - 2021-09-23

Added

- Character-level tokenizer/detokenizer

5.5.23 0.4.3 - 2021-09-20

Added

- New embeddings not requiring internet (`CountVectorizer`, `TfidfVectorizer`)
- `Rules` return type
- First version of local rules using `SkopeRules`
- More test cases

Changed

- New default embedding method for `MMDCritic` and `KMedoids`
- Version moved to `__init__.py`
- New README.md layout
- Updates to Anchor local explanations
- Added random state in example_usage to ensure reproducibility

5.5.24 0.4.2 - 2021-09-13

Fixed

- Hotfix to fix `predict_proba` usage

5.5.25 0.4.1 - 2021-09-13

Fixed

- Hotfix to make dependency on internet optional

5.5.26 0.4.0 - 2021-09-13

Added

- Initial support for embeddings/vectors
- Support for dimensionality reduction
- Initial implementation of MMD-Critic
- Initial implementation of labelwise MMD-Critic
- Initial implementation of prototype selection using k-Medoids

Changed

- Updated README.md

5.5.27 0.3.8 - 2021-09-07

Changed

- Support for dimensionality reduction

Fixed

- Bugfix in including `locale/*.json` files during setup

5.5.28 0.3.7 - 2021-09-07

Added

- Dependencies for package

5.5.29 0.3.6 - 2021-09-07

Added

- PyPI release script to .gitignore
- Badges to README.md
- Added dependencies to `setup.py`

5.5.30 0.3.5 - 2021-09-03

Changed

- Locale changed to .json format, to remove optional dependency

Fixed

- Bugfix for getting key in TokenFrequency
- Bugfixes in FeatureAttribution return type
- Bugfixes in `i18n`

5.5.31 0.3.4 - 2021-08-18

Changed

- External logo url

Fixed

- Hotfix in FeatureAttribution

5.5.32 0.3.3 - 2021-08-18

Added

- Updated to support `instancelib==0.3.1.2`
- `i18n` internationalization support
- CHANGELOG.md

Changed

- Additional samples in example dataset

Fixed

- Bugfixes for LIME and FeatureAttribution return type

5.5.33 0.3.2 - 2021-07-27

Added

- Initial support for ``*Foil Trees*` <https://github.com/MarcelRobeer/ContrastiveExplanation>`_
- Logo in documentation

Changed

- Improved documentation

5.5.34 0.3.1 - 2021-07-23

Added

- `flake8` linting
- CI/CD Pipeline
- Run test scripts

5.5.35 0.3.0 - 2021-07-20

Added

- Updated to support `instancelib==0.3.0.0`

Changed

- Improved documentation
- `global_explanation` classes have equal return types

5.5.36 0.2 - 2021-06-22

Added

- LICENSE.md
- Updated to support `instancelib==0.2.3.1`

Changed

- Module description

5.5.37 0.1 - 2021-05-28

Added

- README.md
- Example usage
- Local explanation classes (LIME, KernelSHAP)
- Global explanation classes
- Data augmentation/sampling
- Feature selection
- Local surrogates
- Tokenization
- `git` setup

5.6 Contributing

5.6.1 Maintenance

Contributors

- Marcel Robeer (`@m.j.robeer`)
- Michiel Bron (`@mpbron-phd`)

Todo

Tasks yet to be done:

- Implement local post-hoc explanations:
  - Implement Anchors
- Implement global post-hoc explanations:
  - Representative subset
- Add support for regression models
- More complex data augmentation
  - Top-k replacement (e.g. according to LM / WordNet)
  - Tokens to exclude from being changed
  - Bag-of-words style replacements
- Write more tests

5.7 Indices and tables

- genindex
- modindex
- search

PYTHON MODULE INDEX

t

INDEX

F

G

I

K

L

M

N

O

P

R

S

T

U

W