dennisbakhuis / **pigeonXT**

Type / to search

Code · Issues 1 · Pull requests · Actions · Projects · Security · Insights

## pigeonXT Public

forked from agermanidis/pigeon

master · nches · 0 Tags · Go to file t · Go · + · Add fil · Code

This branch is 53 commits ahead of `agermanidis/pigeon:master` .

**kallewesterling** A 7931b36 · 6 months ago · 55 Commits

| | | |
|---|---|---|
| .idea | moved to poetry and h... | 11 months ago |
| assets | rewrote internals to us... | 2 years ago |
| pigeonXT | Adding reference to ty... | 6 months ago |
| .gitignore | updated pyproject.toml | 11 months ago |
| .pre-commit-co... | minor pre-commit cha... | 11 months ago |
| LICENSE | moved to poetry and h... | 11 months ago |
| MANIFEST.in | replaced README.rst ... | 3 years ago |
| README.md | updated pyproject.toml | 11 months ago |
| environment.yml | moved to poetry and h... | 11 months ago |
| pigeonXT_Exam... | rewrote internals to us... | 2 years ago |
| poetry.lock | Import issues (#9) | last year |
| pyproject.toml | Import issues (#9) | last year |
| requirements.txt | moved to poetry and h... | 11 months ago |
| setup.py | Import issues (#9) | last year |

### About

🐦 Quickly annotate data from the comfort of your Jupyter notebook

Readme

Apache-2.0 license

Activity

252 stars

10 watching

124 forks

Report repository

### Releases

No releases published

### Packages

No packages published

### Languages

- ● **Python** 59.0%
- ● **Jupyter Notebook** 41.0%

# 🐦 pigeonXT - Quickly annotate data in Jupyter Lab

PigeonXT is an extention to the original Pigeon, created by Anastasis Germanidis. PigeonXT is a simple widget that lets you quickly annotate a dataset of unlabeled examples from the comfort of your Jupyter notebook.

PigeonXT currently support the following annotation tasks:

- binary / multi-class classification
- multi-label classification
- regression tasks
- captioning tasks

Anything that can be displayed on Jupyter (text, images, audio, graphs, etc.) can be displayed by pigeon by providing the appropriate `display_fn` argument.

Additionally, custom hooks can be attached to each row update ( `example_process_fn` ), or when the annotating task is complete( `final_process_fn` ).

There is a full blog post on the usage of PigeonXT on Towards Data Science.

### Contributors

- Anastasis Germanidis
- Dennis Bakhuis
- Ritesh Agrawal
- Deepak Tunuguntla
- Bram van Es

## Installation

PigeonXT obviously needs a Jupyter Lab environment. Futhermore, it requires ipywidgets. The widget itself can be installed using pip:

```
    pip install pigeonXT-jupyter
```

Currently, it is much easier to install due to Jupyterlab 3: To run the provided examples in a new environment using Conda:

```
    conda create --name pigeon python=3.9
    conda activate pigeon
    pip install numpy pandas jupyterlab ipywidgets pigeonXT-jupyter
```

For an older Jupyterlab or any other trouble, please try the old method:

```
    conda create --name pigeon python=3.7
    conda activate pigeon
    conda install nodejs
    pip install numpy pandas jupyterlab ipywidgets
    jupyter nbextension enable --py widgetsnbextension
    jupyter labextension install @jupyter-widgets/jupyterlab-manager

    pip install pigeonXT-jupyter
```

Starting Jupyter Lab environment:

```
    jupyter lab
```

### Development environment

I have moved the development environment to Poetry. To create an identical environment use:

```
conda env create -f environment.yml
conda activate pigeonxt
poetry install
pre-commit install
```

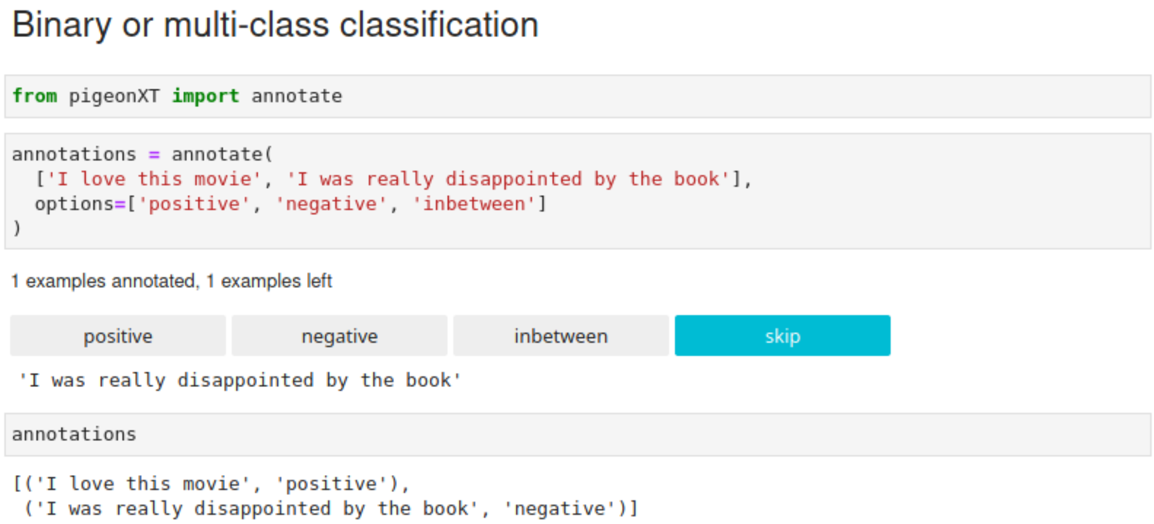
## Examples

Examples are also provided in the accompanying notebook.

## Binary or multi-class text classification

Code:

```
import pandas as pd
import pigeonXT as pixt

annotations = pixt.annotate(
    ['I love this movie', 'I was really disappointed by the book'],
    options=['positive', 'negative', 'inbetween']
)
```



Binary or multi-class classification

```
from pigeonXT import annotate
```

```
annotations = annotate(
  ['I love this movie', 'I was really disappointed by the book'],
  options=['positive', 'negative', 'inbetween']
)
```

1 examples annotated, 1 examples left

```
'I was really disappointed by the book'
```

```
annotations
```

```
[('I love this movie', 'positive'),
 ('I was really disappointed by the book', 'negative')]
```

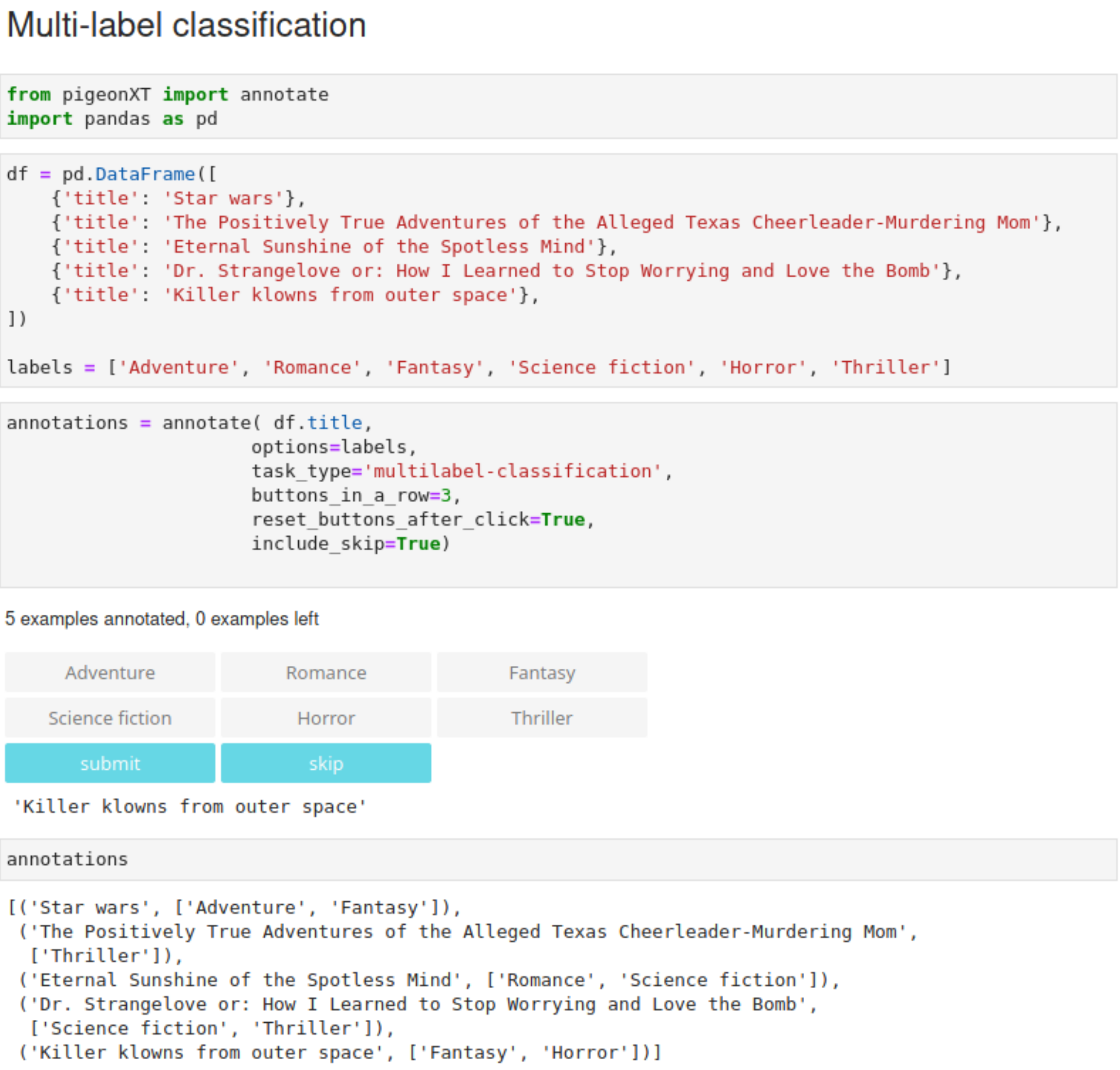
Preview:

## Multi-label text classification

Code:

```
import pandas as pd
import pigeonXT as pixt

df = pd.DataFrame([
    {'example': 'Star wars'},
    {'example': 'The Positively True Adventures of the Alleged Texas Cheerleader-Murdering Mom'},
    {'example': 'Eternal Sunshine of the Spotless Mind'},
    {'example': 'Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb'},
    {'example': 'Killer klowns from outer space'},
])

labels = ['Adventure', 'Romance', 'Fantasy', 'Science fiction', 'Horror', 'Thriller']

annotations = pixt.annotate(
    df,
    options=labels,
    task_type='multilabel-classification',
    buttons_in_a_row=3,
    reset_buttons_after_click=True,
    include_next=True,
    include_back=True,
)
```

README Apache-2.0 license



Multi-label classification

```
from pigeonXT import annotate
import pandas as pd
```

```
df = pd.DataFrame([
    {'title': 'Star wars'},
    {'title': 'The Positively True Adventures of the Alleged Texas Cheerleader-Murdering Mom'},
    {'title': 'Eternal Sunshine of the Spotless Mind'},
    {'title': 'Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb'},
    {'title': 'Killer klowns from outer space'},
])

labels = ['Adventure', 'Romance', 'Fantasy', 'Science fiction', 'Horror', 'Thriller']
```

```
annotations = annotate( df.title,
                        options=labels,
                        task_type='multilabel-classification',
                        buttons_in_a_row=3,
                        reset_buttons_after_click=True,
                        include_skip=True)
```

5 examples annotated, 0 examples left

```
'Killer klowns from outer space'
```

```
annotations
```

```
[('Star wars', ['Adventure', 'Fantasy']),
 ('The Positively True Adventures of the Alleged Texas Cheerleader-Murdering Mom',
  ['Thriller']),
 ('Eternal Sunshine of the Spotless Mind', ['Romance', 'Science fiction']),
 ('Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb',
  ['Science fiction', 'Thriller']),
 ('Killer klowns from outer space', ['Fantasy', 'Horror'])]
```

Preview:

## Image classification

Code:

```
import pandas as pd
import pigeonXT as pixt

from IPython.display import display, Image

annotations = pixt.annotate(
  ['assets/img_example1.jpg', 'assets/img_example2.jpg'],
  options=['cat', 'dog', 'horse'],
  display_fn=lambda filename: display(Image(filename))
)
```

Preview:

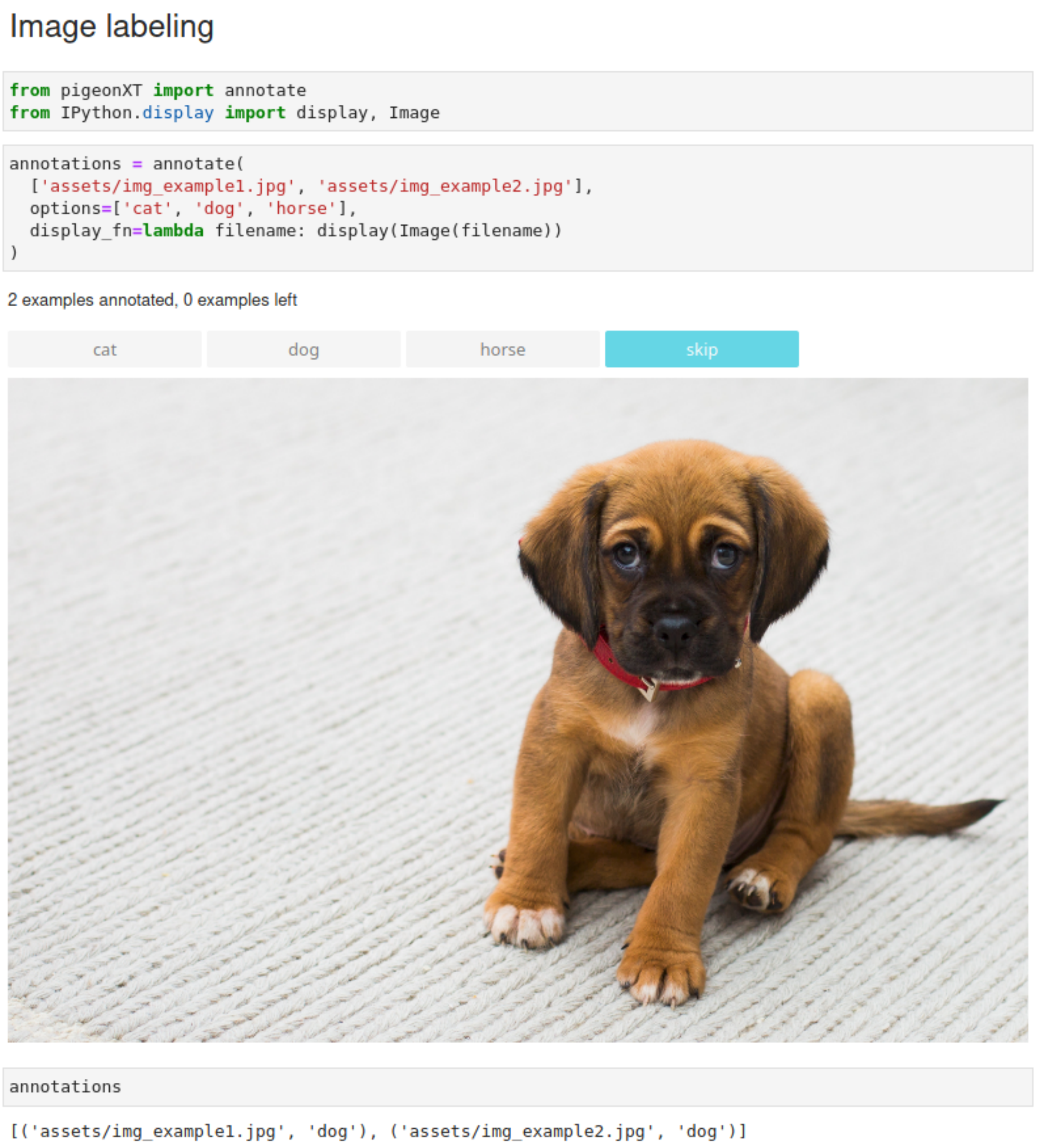


Image labeling

```
from pigeonXT import annotate
from IPython.display import display, Image
```

```
annotations = annotate(
  ['assets/img_example1.jpg', 'assets/img_example2.jpg'],
  options=['cat', 'dog', 'horse'],
  display_fn=lambda filename: display(Image(filename))
)
```

2 examples annotated, 0 examples left

```
annotations
```

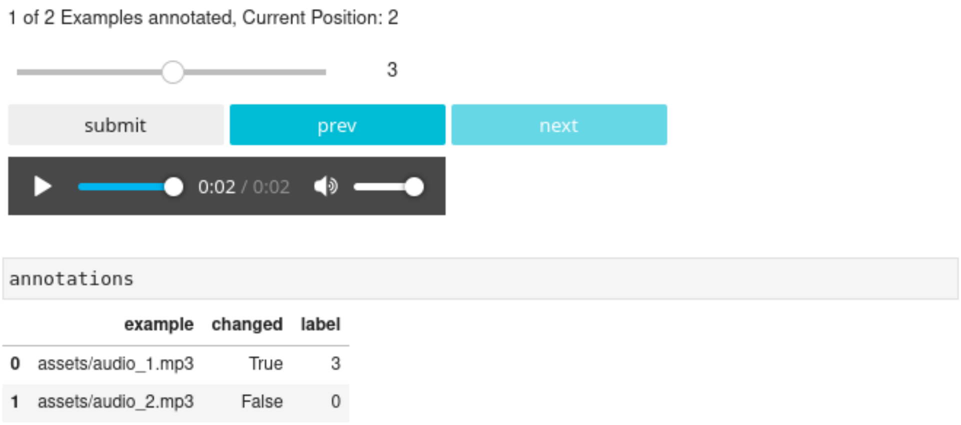
```
[('assets/img_example1.jpg', 'dog'), ('assets/img_example2.jpg', 'dog')]
```

## Audio classification

Code:

```
import pandas as pd
import pigeonXT as pixt

from IPython.display import Audio

annotations = pixt.annotate(
    ['assets/audio_1.mp3', 'assets/audio_2.mp3'],
    task_type='regression',
    options=(1,5,1),
    display_fn=lambda filename: display(Audio(filename, autoplay=True))
)

annotations
```

Preview:



1 of 2 Examples annotated, Current Position: 2

```
annotations
```

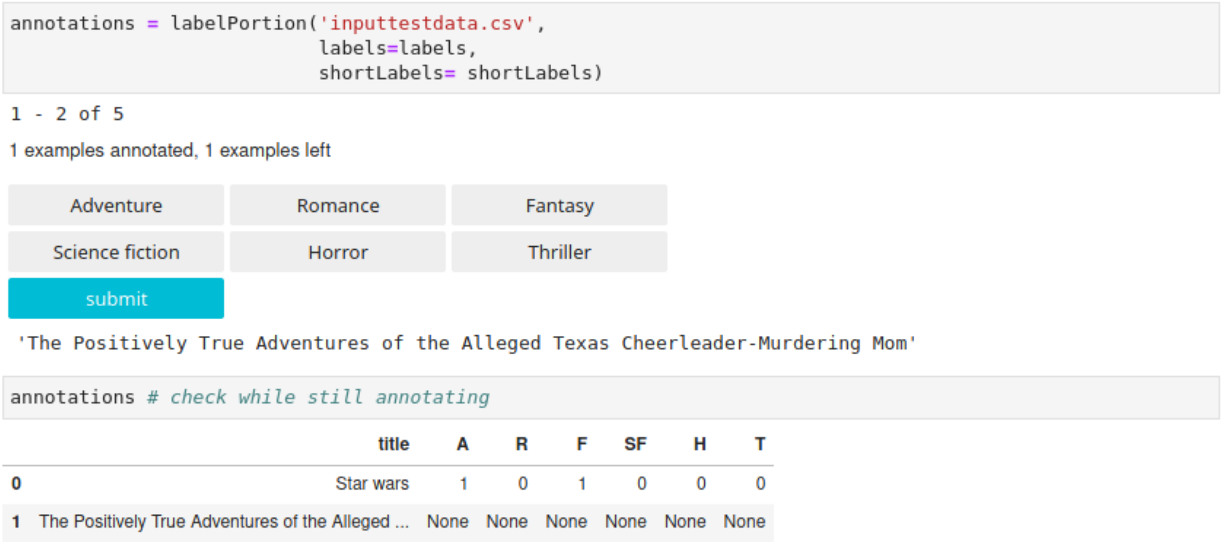
| | example | changed | label |
|---|---|---|---|
| 0 | assets/audio_1.mp3 | True | 3 |
| 1 | assets/audio_2.mp3 | False | 0 |

## multi-label text classification with custom hooks

Code:

```
import pandas as pd
import numpy as np

from pathlib import Path
from pigeonXT import annotate

df = pd.DataFrame([
    {'example': 'Star wars'},
    {'example': 'The Positively True Adventures of the Alleged Texas Cheerleader-Murdering Mom'},
    {'example': 'Eternal Sunshine of the Spotless Mind'},
    {'example': 'Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb'},
    {'example': 'Killer klowns from outer space'},
])

labels = ['Adventure', 'Romance', 'Fantasy', 'Science fiction', 'Horror', 'Thriller']
shortLabels = ['A', 'R', 'F', 'SF', 'H', 'T']

df.to_csv('inputtestdata.csv', index=False)


def setLabels(labels, numClasses):
    row = np.zeros([numClasses], dtype=np.uint8)
    row[labels] = 1
    return row

def labelPortion(
    inputFile,
    labels = ['yes', 'no'],
    outputFile='output.csv',
    portionSize=2,
    textColumn='example',
    shortLabels=None,
):
    if shortLabels == None:
        shortLabels = labels

    out = Path(outputFile)
    if out.exists():
        outdf = pd.read_csv(out)
        currentId = outdf.index.max() + 1
    else:
        currentId = 0

    indf = pd.read_csv(inputFile)
    examplesInFile = len(indf)
    indf = indf.loc[currentId:currentId + portionSize - 1]
    actualPortionSize = len(indf)
    print(f'{currentId + 1} - {currentId + actualPortionSize} of {examplesInFile}')
    sentences = indf[textColumn].tolist()

    for label in shortLabels:
        indf[label] = None

    def updateRow(example, selectedLabels):
        print(example, selectedLabels)
        labs = setLabels([labels.index(y) for y in selectedLabels], len(labels))
        indf.loc[indf[textColumn] == example, shortLabels] = labs

    def finalProcessing(annotations):
        if out.exists():
            prevdata = pd.read_csv(out)
            outdata = pd.concat([prevdata, indf]).reset_index(drop=True)
        else:
            outdata = indf.copy()
        outdata.to_csv(out, index=False)

    annotated = annotate(
        sentences,
        options=labels,
        task_type='multilabel-classification',
        buttons_in_a_row=3,
        reset_buttons_after_click=True,
        include_next=False,
        example_process_fn=updateRow,
        final_process_fn=finalProcessing
    )
    return indf

def getAnnotationsCountPerlabel(annotations, shortLabels):

    countPerLabel = pd.DataFrame(columns=shortLabels, index=['count'])

    for label in shortLabels:
        countPerLabel.loc['count', label] = len(annotations.loc[annotations[label] == 1.0])

    return countPerLabel

def getAnnotationsCountPerlabel(annotations, shortLabels):

    countPerLabel = pd.DataFrame(columns=shortLabels, index=['count'])

    for label in shortLabels:
        countPerLabel.loc['count', label] = len(annotations.loc[annotations[label] == 1.0])

    return countPerLabel


annotations = labelPortion('inputtestdata.csv',
                           labels=labels,
                           shortLabels= shortLabels)

# counts per label
getAnnotationsCountPerlabel(annotations, shortLabels)
```

Preview:



```
annotations = labelPortion('inputtestdata.csv',
                           labels=labels,
                           shortLabels= shortLabels)
```

```
1 - 2 of 5
```

1 examples annotated, 1 examples left

```
'The Positively True Adventures of the Alleged Texas Cheerleader-Murdering Mom'
```

```
annotations # check while still annotating
```

| | title | A | R | F | SF | H | T |
|---|---|---|---|---|---|---|---|
| 0 | Star wars | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | The Positively True Adventures of the Alleged ... | None | None | None | None | None | None |

The complete and runnable examples are available in the provided Notebook.