

Light Curves Classifier

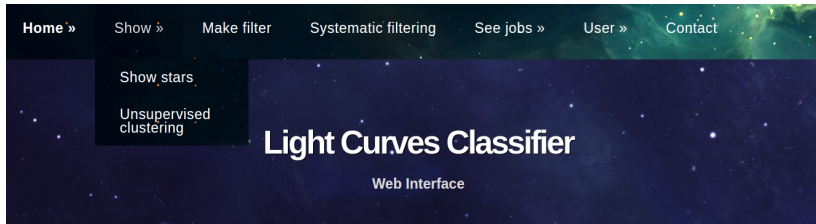
Martin Vo

Masaryk University
Department of Theoretical Physics and Astrophysics
Czech Republic

EWASS 2017
Prague, 30 June

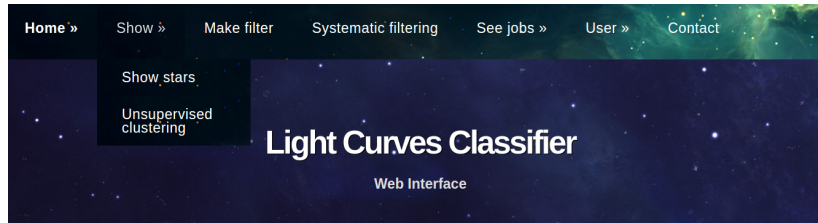
Light Curves Classifier

Package for tasks realated to classification of light curves



Light Curves Classifier

Package for tasks realated to classification of light curves



Using options:

- 1 Python package (github.com/mavrix93/LightCurvesClassifier)
- 2 Web Interface (vocloud-dev.asu.cas.cz/lcc)
- 3 Command line UI

Motivation

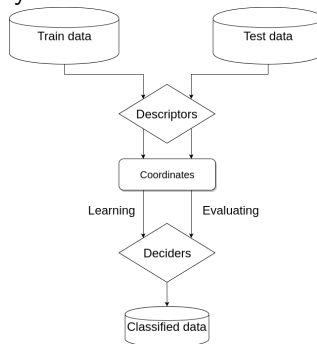
Workflow of classification tasks is very similar

- 1 Get data
- 2 Get features
- 3 Train classifiers
- 4 Evaluate methods
- 5 Find new objects

Motivation

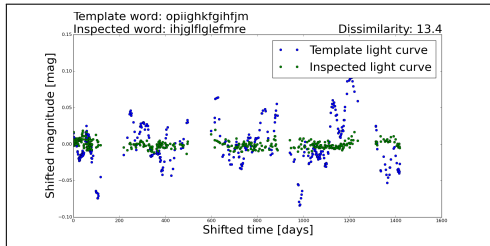
Workflow of classification tasks is very similar

- 1 Get data
- 2 Get features
- 3 Train classifiers
- 4 Evaluate methods
- 5 Find new objects



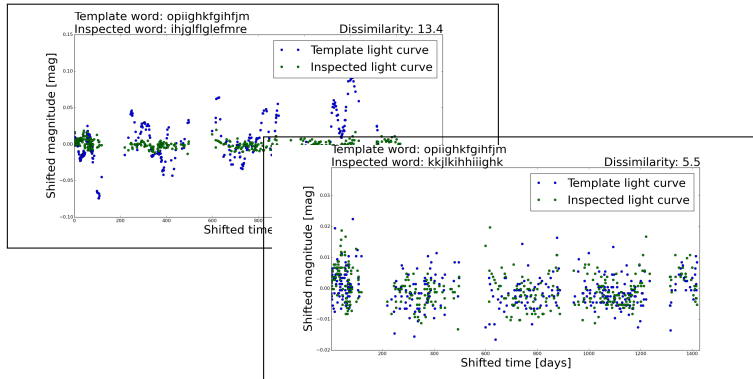
Light curves

How to describe light curves?



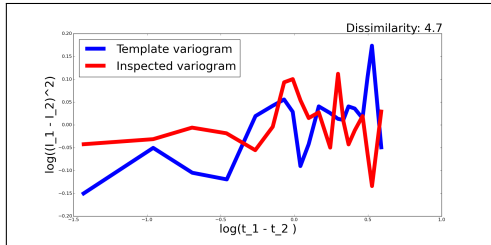
Light curves

How to describe light curves?



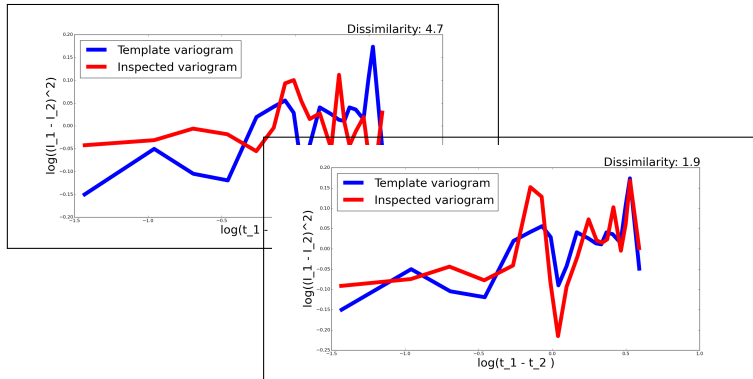
Variograms

Variograms



Variograms

Variograms



Classification

Available descriptors

- PositionDescriptor
- VariogramShapeDescr
- VariogramSlopeDescr
- CurveDescr
- HistShapeDescr
- CurveDensityDescr
- ColorIndexDescr
- PropertyDescr

ADD >>

Selected items

- KurtosisDescr
- SkewnessDescr
- CurvesShapeDescr
- AbbeValueDescr

<< REMOVE

Available deciders

- NeuronDecider
- ExtraTreesDec
- CustomDecider
- QDADec
- LDADec
- TreeDec
- AdaBoostDec
- GaussianNBDec

ADD >>

Selected items

- SVCDec
- RandomForestDec
- GradBoostDec

<< REMOVE

Classification

KurtosisDescr

bins

absolute

SkewnessDescr

bins

absolute

CurvesShapeDescr

comp_stars

days_per_bin

alphabet_size

slide

meth

GradBoostDec

threshold

loss

learning_rate

n_estimators

subsample

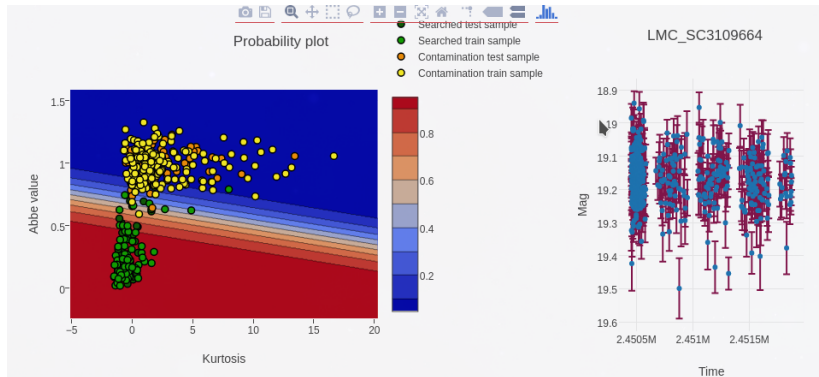
criterion

min_samples_split

min_samples_leaf

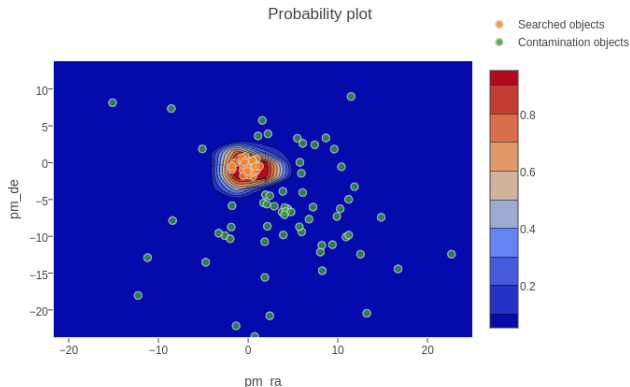
Classification

Qso vs non variable stars described by kurtosis and Abbe value



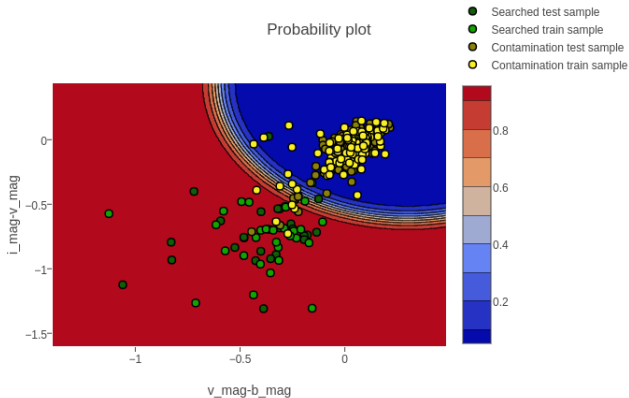
Classification

Members vs non-members of a star cluster described by proper motions



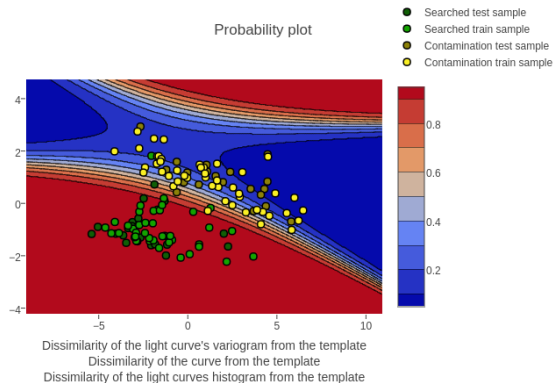
Classification

Qso vs Be Stars described by color indexes



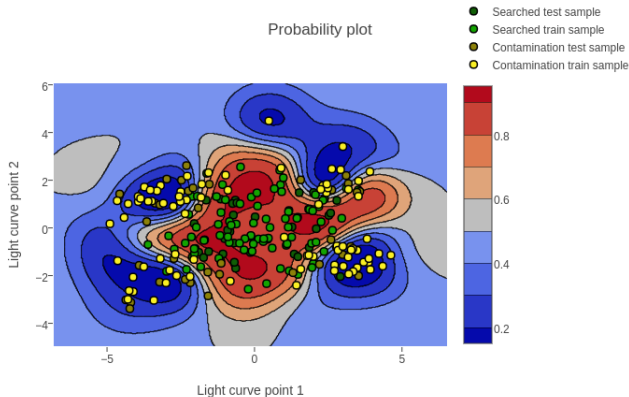
Classification

Qso vs periodic variable stars described by dissimilarity of light curves, histograms and variograms in symbolic space



Classification

Qso vs Be Stars described by reduced light curves by using PCA



Systematic searching

Available connectors

- Asas
- Oglell
- FileManager
- CorotBright
- Kepler
- Macho
- CorotFaint

ADD >>

Selected items

- Catalina
- Oglell

<< REMOVE

Catalina

Query

Load query file

```
ra: 170.8113  
dec: 34.1737  
delta: 2  
nearest: `True`
```

Oglell

Query

Load query file

```
field_num:5:7  
starid:1:5000  
target:lmc
```

Getting stars

```
queries = [{"ra": 297.8399, "dec": 46.57427,  
            "delta": 10},  
            {"kic_num": 9787239}]  
  
client = StarsProvider.getProvider("Kepler",  
                                   queries)  
  
stars = client.getStars()
```

Listing 1: Getting stars from Kepler

Filtering

```
filt = StarsFilter([AbbeValueDescr(bins=150),  
                  KurtosisDescr(bins=90)],  
                  [TreeDec(treshold=0.8)])  
filt.learn(searched_stars,  
           contam_stars)  
passed_stars = filt.filterStars(inspected_stars)
```

Listing 2: Filtering

Tuning parameters

```
estim = ParamsEstimator(searched_stars ,  
                        contamination_stars ,  
                        descriptors ,  
                        deciders ,  
                        tuned_params )  
  
estim.fit()
```

Listing 3: Tuning parameters

Implementing own modules

```
class StdDesc(BaseDescriptor):  
  
    def getFeatures(self, star):  
        """  
        Get standart deviation of magnitudes  
        """  
        return np.std(star.lightCurve.mag)
```

Listing 4: Custom descriptor

Uploading methods

```
PackageReader.appendModules("descriptors",  
                             "some_path/my_descriptors")
```

Listing 5: Uploading methods

Uploading methods

```
PackageReader.appendModules("descriptors",  
                             "some_path/my_descriptors")
```

Listing 6: Uploading methods

In Web Interface



Own connectors

```
class MachoDb(VizierTapBase, LightCurvesDb):
    """
    Client for MACHO database

    EXAMPLES:
    -----
        queries = [{"Field": 1, "Tile": 3441, "Seqn": 25}]
        client = StarsProvider.getProvider(obtain_method="Macho",
                                          obtain_params=queries)
        stars = client.getStars()
    """

    TABLE = "II/247/machovar"
    LC_URL = "http://cdsarc.u-strasbg.fr/viz-bin/nph-Plot/w/Vgraph/txt?II%2f247%2f.%2f{macho_name}&F=b%2f"

    NAME = "{Field}.{Tile}.{Seqn}"
    LC_FILE = ""

    LC_META = {"xlabel": "Time",
               "xlabel_unit": "MJD (JD-2400000.5)",
               "origin": "MACHO"}

    IDENT_MAP = {"MachoDb": ("Field", "Tile", "Seqn")}
    MORE_MAP = collections.OrderedDict(((("Class", "var_type"),
                                           ("Vmag", "v_mag"),
                                           ("Rmag", "r_mag"),
                                           ("rPer", "period_r"),
                                           ("bPer", "period_b"))))
```

Conclusion

- Many "ready to use" methods
- Connectors to most of the largest databases of light curves
- Possibility to easily implement new methods and connectors
- Executing jobs in cloud (for web service)
- Easy to tune parameters of methods
- Powerful library for development