MOA

Agnieszka Lipska

https://moa.cms.waikato.ac.nz/

Agenda

- MOA – opis
- Klasyfikacja
- Klastrowanie
- Systemy rekomendacyjne
- Własny klasyfikator

MOA

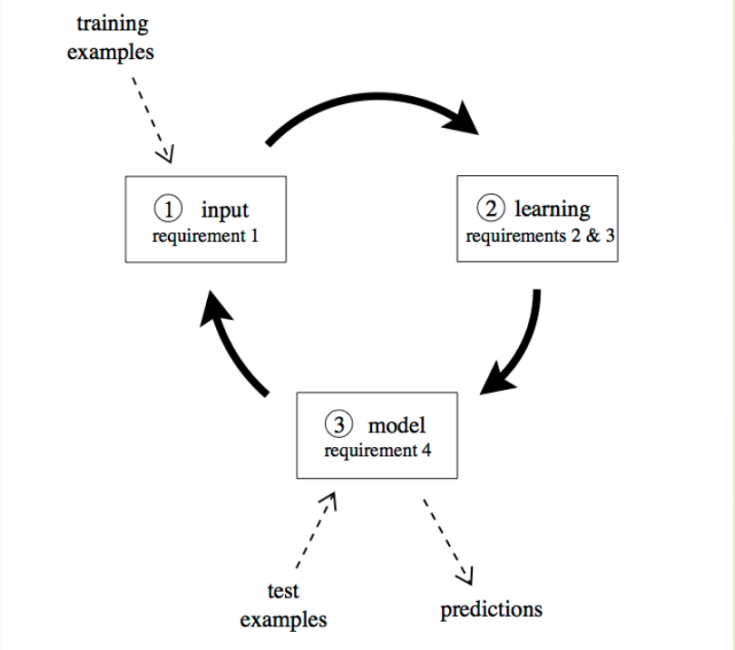
- **M**assive **O**n-line **A**nalysis
- Framework do przetwarzania strumieni danych
- Open source
- Napisany w Javie
- Powiązany z Weką
- Zawiera zbiór istniejących algorytmów i miar

Algorytmy uczenia maszynowego

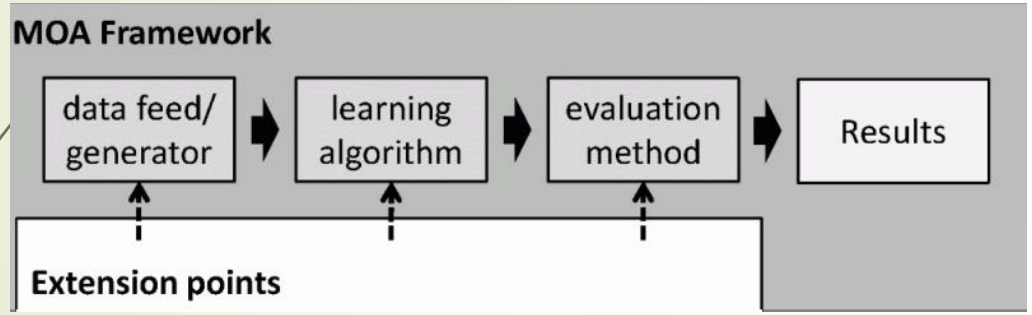
- Klasyfikacja
- Regresja
- Klastrowanie
- Detekcja anomalii
- Wykrywanie concept drift
- Systemy rekomendacyjne
- Active Learning

Wymagania dla przetwarzania strumieni danych

- Przetwarzanie przykładu po przykładzie, każdy przykład czytany tylko raz
- Ograniczona złożoność pamięciowa
- Ograniczona złożoność czasowa
- Zawsze przygotowany do klasyfikacji



Workflow

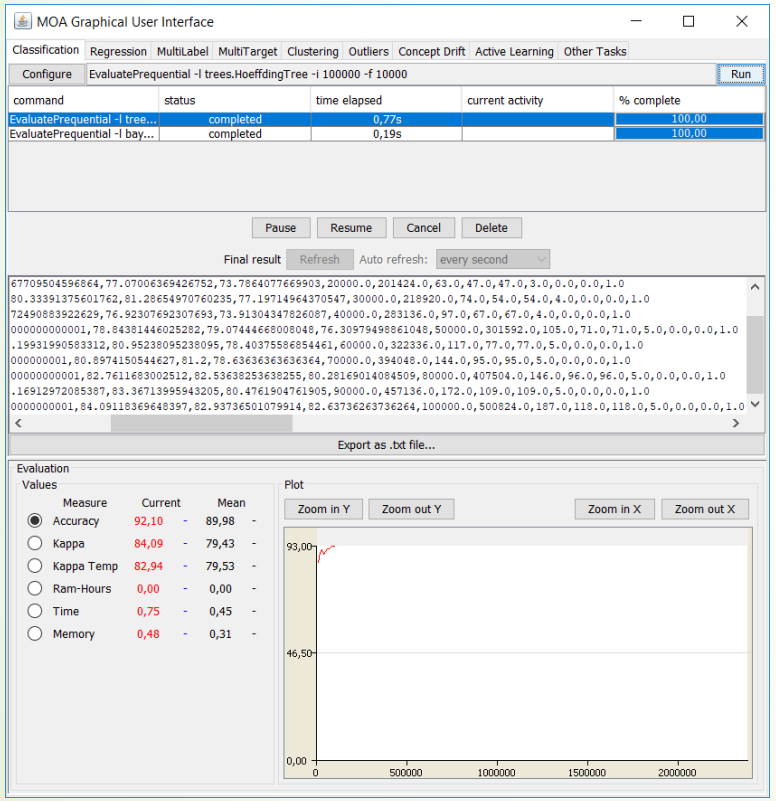


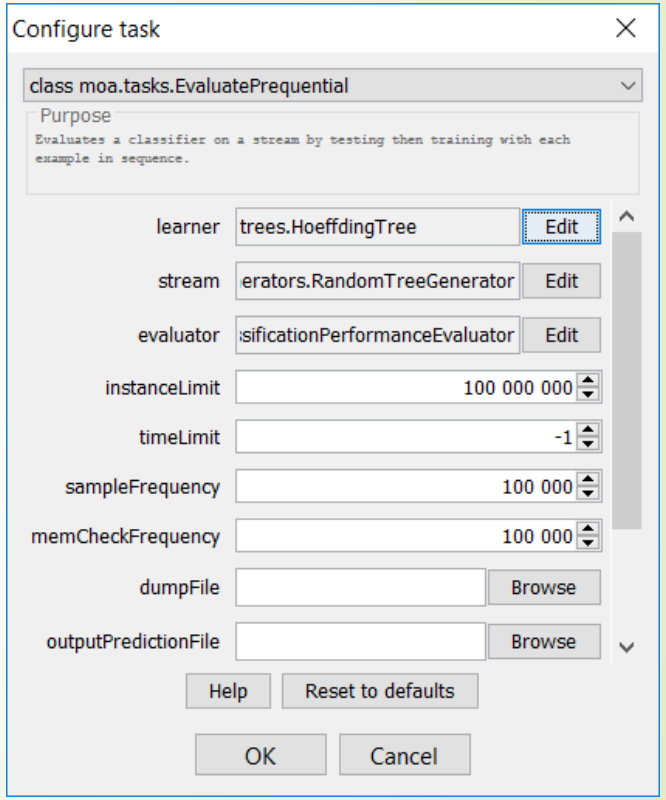
1. Wybór i konfiguracja generatora
2. Wybór algorytmu i ustawienie jego parametrów
3. Wybór miary
4. Wyniki

Klasyfikacja

- Naive Bayes
- Hoeffding Tree
- Bagging
- Boosting
- Stochastic Gradient Descent
- Single Classifier Drift
- …

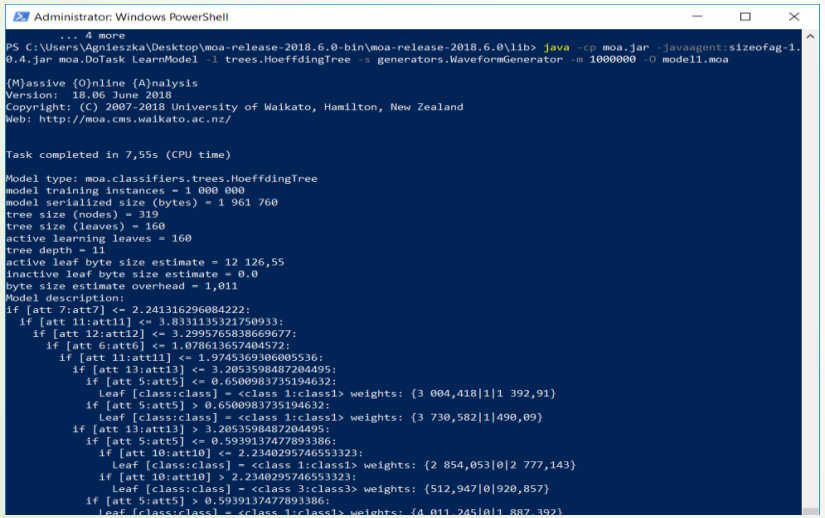
Klasyfikacja



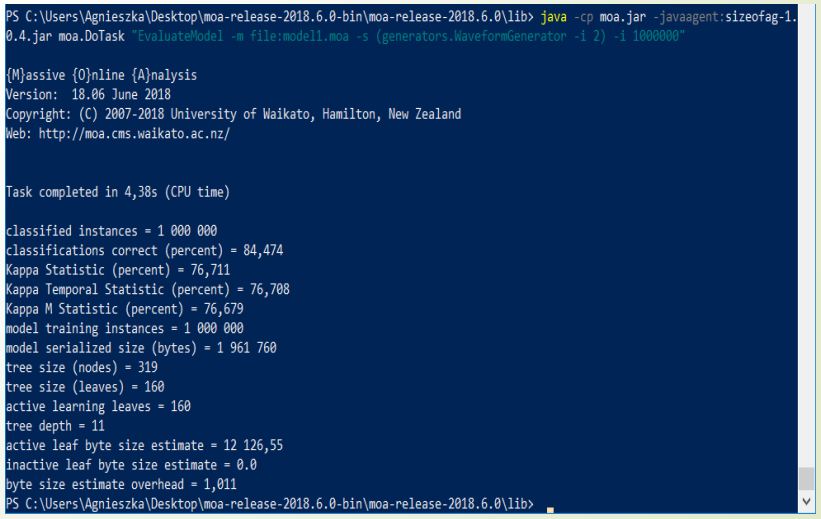


Klasyfikacja

- java -cp moa.jar -javaagent:sizeofag-1.0.4.jar moa.DoTask LearnModel –l trees.HoeffdingTree -s generators.WaveformGenerator -m 1000000 -O model1.moa



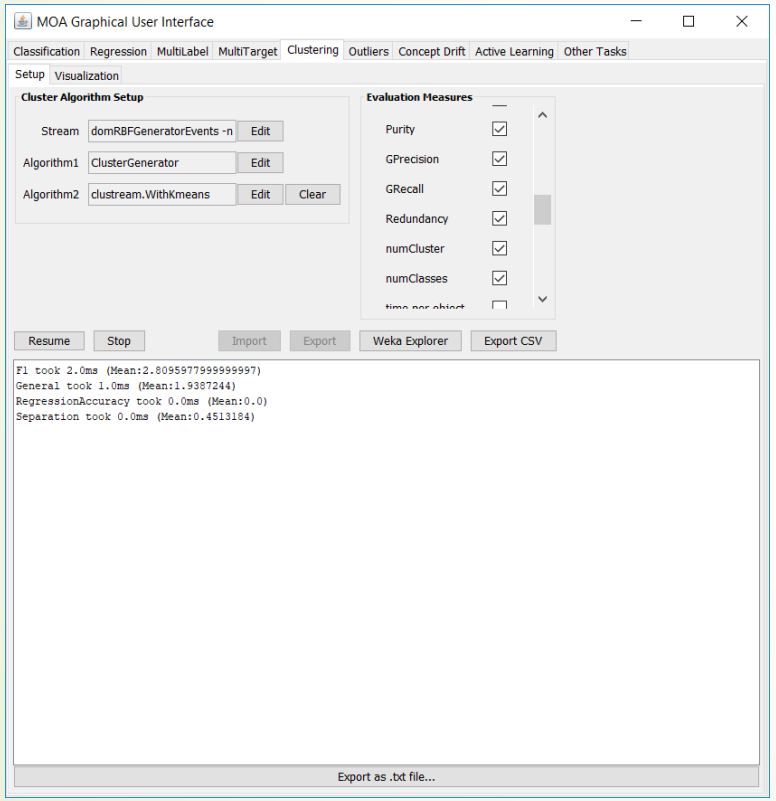
- java -cp moa.jar -javaagent:sizeofag-1.0.4.jar moa.DoTask "EvaluateModel -m file:model1.moa -s (generators.WaveformGenerator -i 2) -i 1000000"

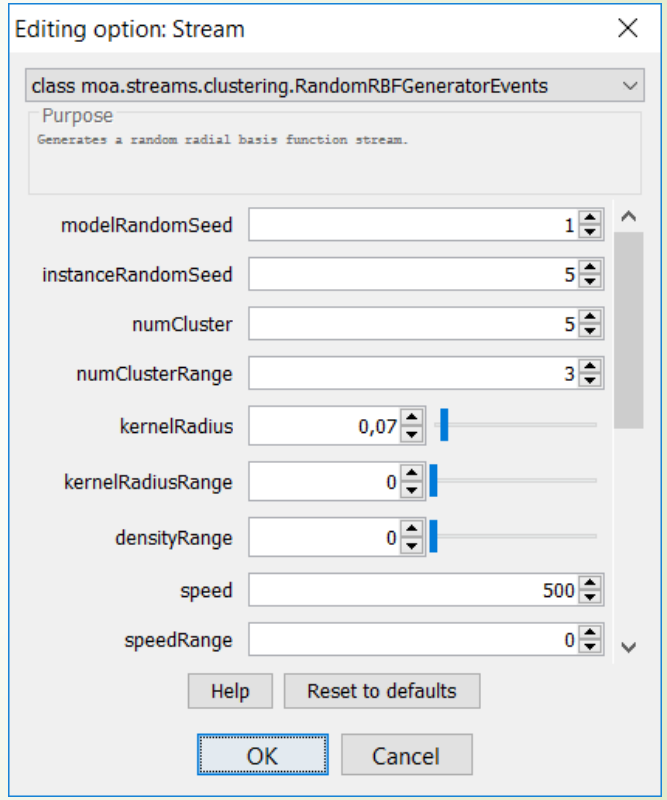


Klastrowanie

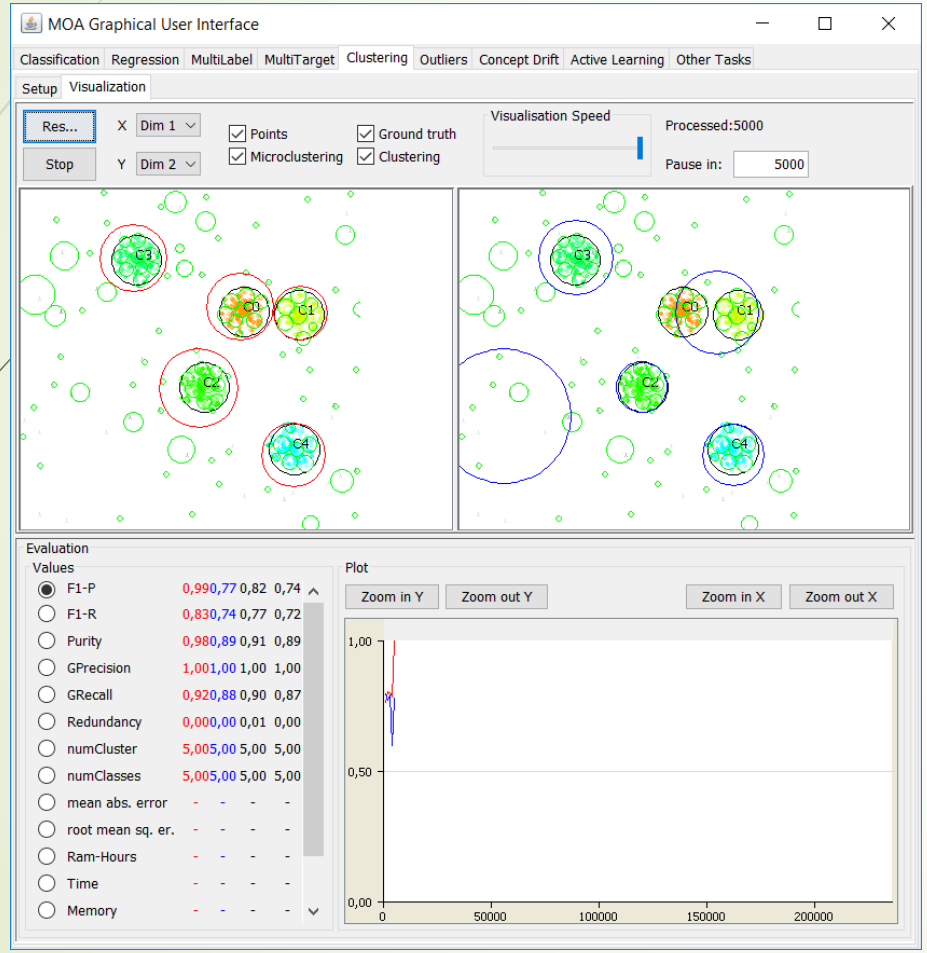
- StreamKM++
- CluStream
- ClusTree
- DenStream
- CobWeb
- …

Klastrowanie





Klastrowanie - wizualizacja

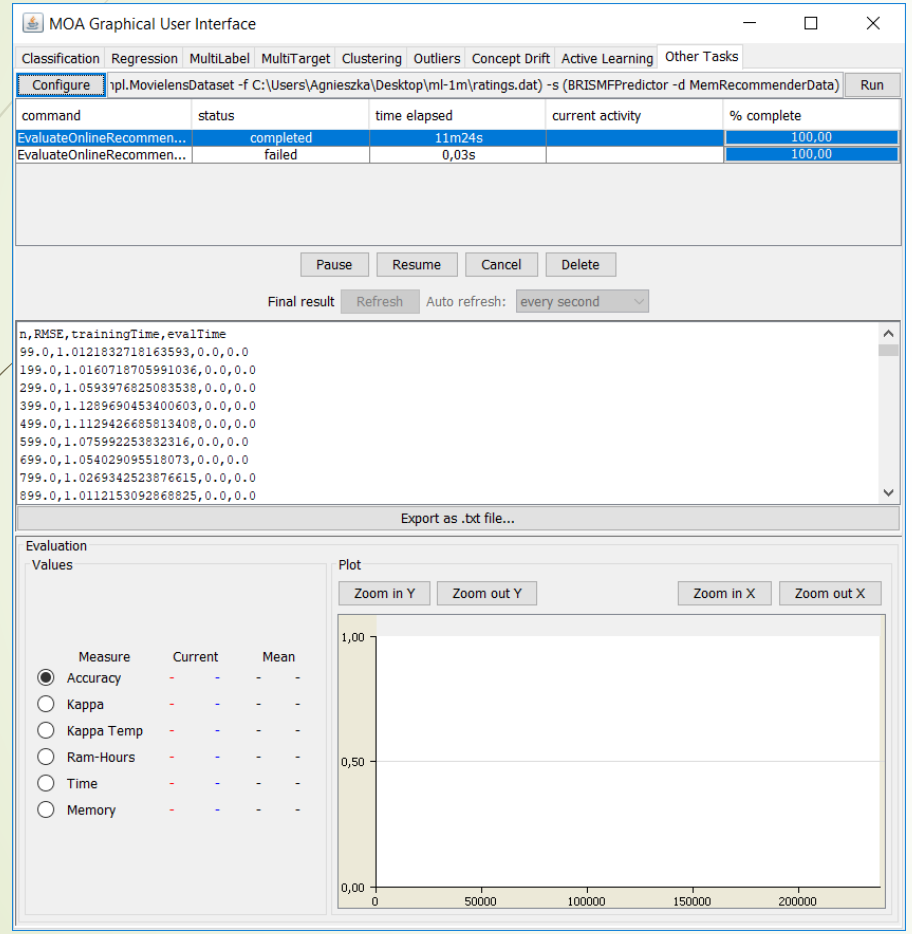


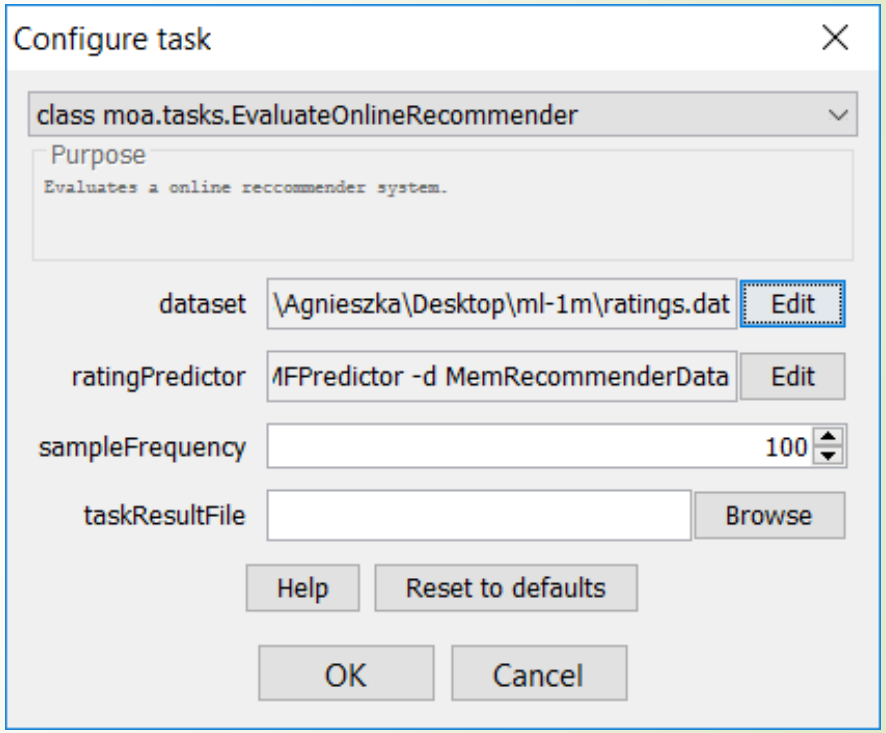
- Po lewej Clustream
- Po prawej Kmeans

Systemy rekomendacyjne

- W oknie „Inne zadania” wybieramy zadanie EvaluateOnilneReccomender
- Wybieramy rating predictor i zbiór danych
- Zbiór danych w postaci trójek [użytkownik, rzecz, ocena]
- BRISMPredictor i BaselinePredictor
- Wynik – nauczony predykator wraz z wartością RSME dla każdego punktu pomiarowego
- Przykładowy zbiór danych MovieLens: https://grouplens.org/datasets/movielens/

Systemy rekomendacyjne





Własny klasyfikator

Klasyfikacja

- Zaimplementowanie interfejsu Classifier.java
- void resetLearningImpl()
- void trainOnInstanceImpl(Instance)
- double[] getVotesForInstance(Instance)

Klastrowanie

- Zaimplementowanie interfejsu Clusterer.java
- void resetLearningImpl()
- void trainOnInstanceImpl(Instance)
- Clustering getClusteringResult()

Własny klasyfikator

- Decision Stump Classifier
- Jedno warstwowe drzewo
- Sprawdza podziały po atrybutach i wybiera ten najlepszy
- Extend moa.classifiers.AbstractClassifier
- Implement MultiClassClassifier

resetLearningImpl()

```
@Override
public void resetLearningImpl() {
    this.bestSplit = null;
    this.observedClassDistribution = new DoubleVector();
    this.attributeObservers = new AutoExpandVector<AttributeClassObserver>();
    this.weightSeenAtLastSplit = 0.0;
}
```

- Metod wywoływana przed rozpoczęciem uczenia, ustawia domyślne wartości
- bestSplit – obecnie najlepszy wybrany podział
- observedClassDistribution – rozkład klas zaobserwowany przez klasyfikator
- attributeObservers – trzyma informacje potrzebne do podjęcia decyzji po którym atrybucie najlepiej dokonywać podziału
- weightSeenLastSplit – czas wykonania ostatniej ewoluacji

trainOnInstanceImpl(Instance inst)

```
@Override
public void trainOnInstanceImpl(Instance inst) {
    this.observedClassDistribution.addToValue((int) inst.classValue(), inst
            .weight());
    for (int i = 0; i < inst.numAttributes() - 1; i++) {
        int instAttIndex = modelAttIndexToInstanceAttIndex(i, inst);
        AttributeClassObserver obs = this.attributeObservers.get(i);
        if (obs == null) {
            obs = inst.attribute(instAttIndex).isNominal() ?
                newNominalClassObserver() : newNumericClassObserver();
            this.attributeObservers.set(i, obs);
        }
        obs.observeAttributeClass(inst.value(instAttIndex), (int) inst
                .classValue(), inst.weight());
    }
    if (this.trainingWeightSeenByModel - this.weightSeenAtLastSplit >=
            this.gracePeriodOption.getValue()) {
        this.bestSplit = findBestSplit((SplitCriterion)
            getPreparedClassOption(this.splitCriterionOption));
        this.weightSeenAtLastSplit = this.trainingWeightSeenByModel;
    }
}
```

- Główna funkcja w algorytmie uczącym
- Wywoływana dla każdego przykładu
- Aktualizacja rozkładów klas
- Dla wszystkich argumentów jeśli nie było jeszcze obserwacji to tworzymy nowy observer
- Uaktualniamy obserwacje na podstawie nowego przykładu
- Sprawdzamy czy minął czas możliwy na znalezienie nowego podziału jeśli tak to uaktualniamy bestSplit

findBestSplit(SplitCriterion criterion)

```
protected AttributeSplitSuggestion findBestSplit(SplitCriterion criterion) {
    AttributeSplitSuggestion bestFound = null;
    double bestMerit = Double.NEGATIVE_INFINITY;
    double[] preSplitDist = this.observedClassDistribution.getArrayCopy();
    for (int i = 0; i < this.attributeObservers.size(); i++) {
        AttributeClassObserver obs = this.attributeObservers.get(i);
        if (obs != null) {
            AttributeSplitSuggestion suggestion =
                obs.getBestEvaluatedSplitSuggestion(
                    criterion,
                    preSplitDist,
                    i,
                    this.binarySplitsOption.isSet());
            if (suggestion.merit > bestMerit) {
                bestMerit = suggestion.merit;
                bestFound = suggestion;
            }
        }
    }
    return bestFound;
}
```

- Iteracja po możliwych podziałach i wybór najlepszego

getVotesForInstance(Instance inst)

```
public double[] getVotesForInstance(Instance inst) {
    if (this.bestSplit != null) {
        int branch = this.bestSplit.splitTest.branchForInstance(inst);
        if (branch >= 0) {
            return this.bestSplit
                    .resultingClassDistributionFromSplit(branch);
        }
    }
    return this.observedClassDistribution.getArrayCopy();
}
```

- Wykorzystanie wyuczonego modelu do predykcji
- Wywołanie metod zaimplementowanych w AttributeSplitSuggestionClass

Kompilacja

- javac -cp moa.jar DecisionStumpTutorial.java
- W folderze musi znajdować się moa.jar i sizeofag.jar
- Otrzymujemy DecisionStumpTutorial.class
- Należy ją umieścić w moa/classifiers/

Dziękuję za uwagę