



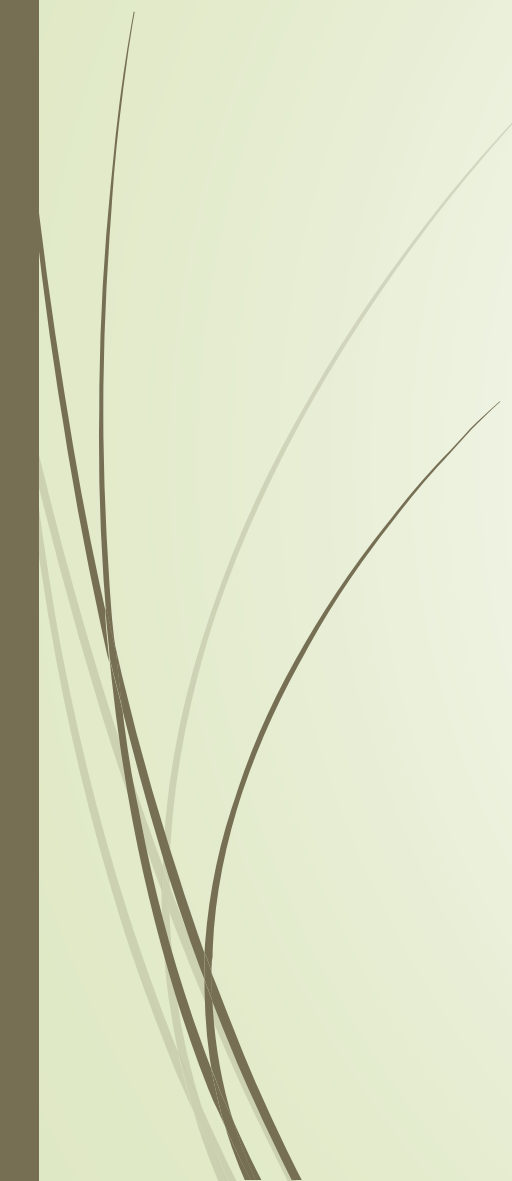
# MOA

Agnieszka Lipska

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



# Agenda

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

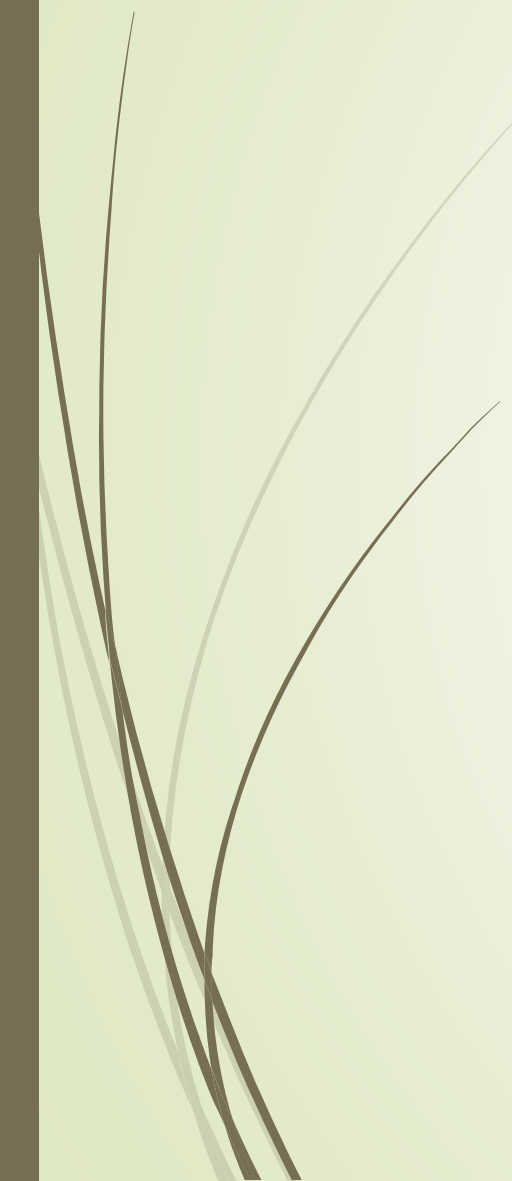


# MOA

- **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

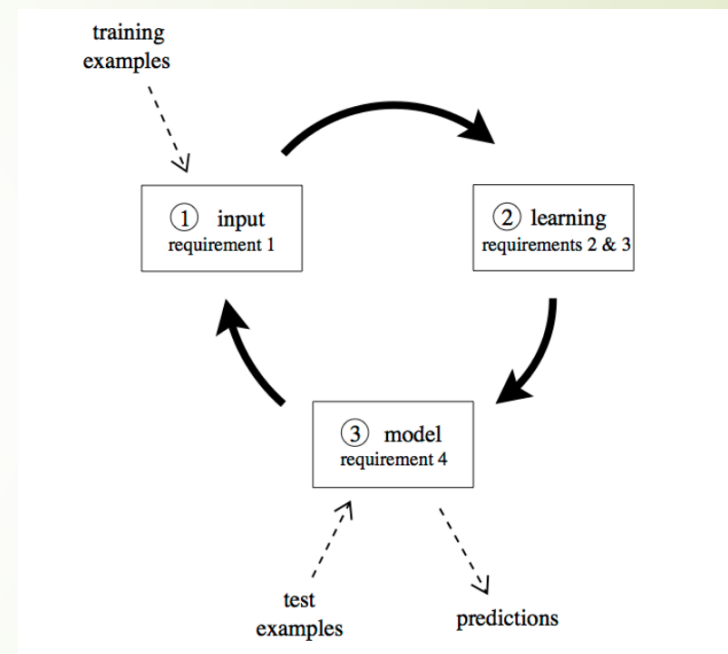


# Algorytmy uczenia maszynowego

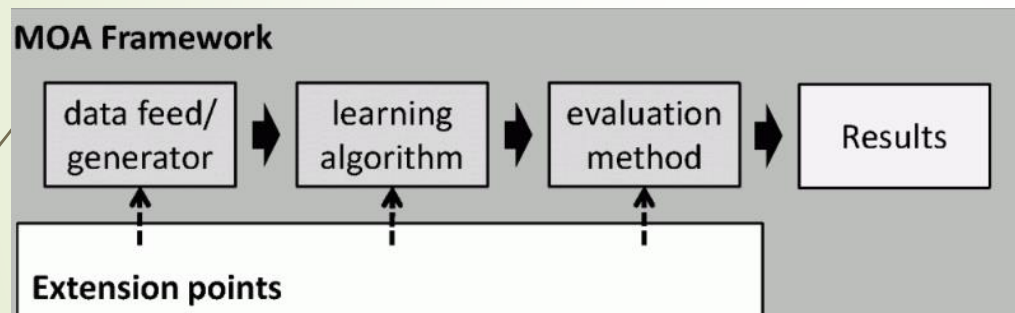
- Klasyfikacja
  - Regresja
  - Klastrowanie
  - Detekcja anomalii
  - Wykrywanie concept drift
  - Systemy rekomendacyjne
  - Active Learning
- 

# 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

MOA Graphical User Interface

Classification Regression MultiLabel MultiTarget Clustering Outliers Concept Drift Active Learning Other Tasks

Configure EvaluatePrequential -l trees.HoeffdingTree -i 100000 -f 10000 Run

command	status	time elapsed	current activity	% complete
EvaluatePrequential -l tree...	completed	0,77s		100,00
EvaluatePrequential -l bay...	completed	0,19s		100,00

Pause Resume Cancel Delete

Final result Refresh Auto refresh: every second

```
67709504596864,77.07006369426752,73.7864077669903,20000.0,201424.0,63.0,47.0,47.0,3.0,0.0,0.0,1.0
80.33391375601762,81.28654970760235,77.19714964370547,30000.0,218920.0,74.0,54.0,54.0,4.0,0.0,0.0,1.0
72490883922629,76.92307692307693,73.91304347826087,40000.0,283136.0,97.0,67.0,67.0,4.0,0.0,0.0,1.0
000000000001,78.84381446025282,79.07444668008048,76.30979498861048,50000.0,301592.0,105.0,71.0,71.0,5.0,0.0,0.0,1.0
.19931990583312,80.95238095238095,78.40375586854461,60000.0,322336.0,117.0,77.0,77.0,5.0,0.0,0.0,1.0
0000000001,80.8974150544627,81.2.78.63636363636364,70000.0,394048.0,144.0,95.0,95.0,5.0,0.0,0.0,1.0
00000000001,82.7611683002512,82.53638253638255,80.28169014084509,80000.0,407504.0,146.0,96.0,96.0,5.0,0.0,0.0,1.0
.16912972085387,83.36713995943205,80.4761904761905,90000.0,457136.0,172.0,109.0,109.0,5.0,0.0,0.0,1.0
0000000001,84.09118369648397,82.93736501079914,82.63736263736264,100000.0,500824.0,187.0,118.0,118.0,5.0,0.0,0.0,1.0
```

Export as .txt file...

Evaluation Values

Measure	Current	Mean
Accuracy	92,10	89,98
Kappa	84,09	79,43
Kappa Temp	82,94	79,53
Ram-Hours	0,00	0,00
Time	0,75	0,45
Memory	0,48	0,31

Plot

Configure task

class moa.tasks.EvaluatePrequential

Purpose  
Evaluates a classifier on a stream by testing then training with each example in sequence.

learner trees.HoeffdingTree Edit

stream erators.RandomTreeGenerator Edit

evaluator isificationPerformanceEvaluator Edit

instanceLimit 100 000 000

timeLimit -1

sampleFrequency 100 000

memCheckFrequency 100 000

dumpFile Browse

outputPredictionFile Browse

Help Reset to defaults

OK Cancel



# 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`

```
Administrator: Windows PowerShell
... 4 more
PS C:\Users\Agnieszka\Desktop\moa-release-2018.6.0-bin\moa-release-2018.6.0\lib> java -cp moa.jar -javaagent:sizeofag-1.0.4.jar moa.DoTask LearnModel -l trees.HoeffdingTree -s generators.WaveformGenerator -m 1000000 -O model1.moa

(M)assive (O)nline (A)nalysis
Version: 18.06 June 2018
Copyright: (C) 2007-2018 University of Waikato, Hamilton, New Zealand
Web: http://moa.cms.waikato.ac.nz/

Task completed in 7,55s (CPU time)

Model type: moa.classifiers.trees.HoeffdingTree
model training instances = 1 000 000
model serialized size (bytes) = 1 961 760
tree size (nodes) = 319
tree size (leaves) = 160
active learning leaves = 160
tree depth = 11
active leaf byte size estimate = 12 126,55
inactive leaf byte size estimate = 0.0
byte size estimate overhead = 1,011
Model description:
if [att 7:att7] <= 2.241316296084222:
  if [att 11:att11] <= 3.8331135321750933:
    if [att 12:att12] <= 3.2995765838666977:
      if [att 0:att0] <= 1.078613657804572:
        if [att 11:att11] <= 1.3765369306005536:
          if [att 13:att13] <= 3.2053598487204495:
            if [att 5:att5] <= 0.6500983735194632:
              Leaf [class:class] = <class 1:class1> weights: {3 004,418|11 392,91}
            if [att 5:att5] > 0.6500983735194632:
              Leaf [class:class] = <class 1:class1> weights: {3 730,582|11 490,09}
          if [att 13:att13] > 3.2053598487204495:
            if [att 5:att5] <= 0.5939137477893386:
              if [att 8:att8] <= 0.2340295746553323:
                Leaf [class:class] = <class 1:class1> weights: {2 854,053|0 2 777,143}
              if [att 8:att8] > 0.2340295746553323:
                Leaf [class:class] = <class 3:class3> weights: {512,947|0 920,857}
            if [att 5:att5] > 0.5939137477893386:
              Leaf [class:class] = <class 1:class1> weights: {4 011,245|0 11 887,392}
```

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

```
PS C:\Users\Agnieszka\Desktop\moa-release-2018.6.0-bin\moa-release-2018.6.0\lib> java -cp moa.jar -javaagent:sizeofag-1.0.4.jar moa.DoTask "EvaluateModel -m file:model1.moa -s (generators.WaveformGenerator -i 2) -i 1000000"

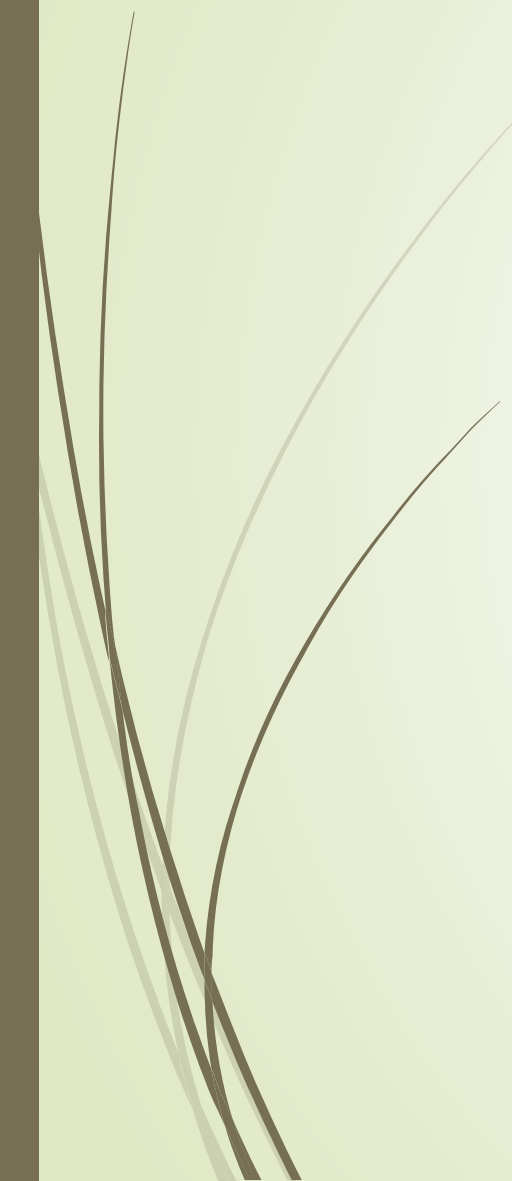
(M)assive (O)nline (A)nalysis
Version: 18.06 June 2018
Copyright: (C) 2007-2018 University of Waikato, Hamilton, New Zealand
Web: http://moa.cms.waikato.ac.nz/

Task completed in 4,38s (CPU time)

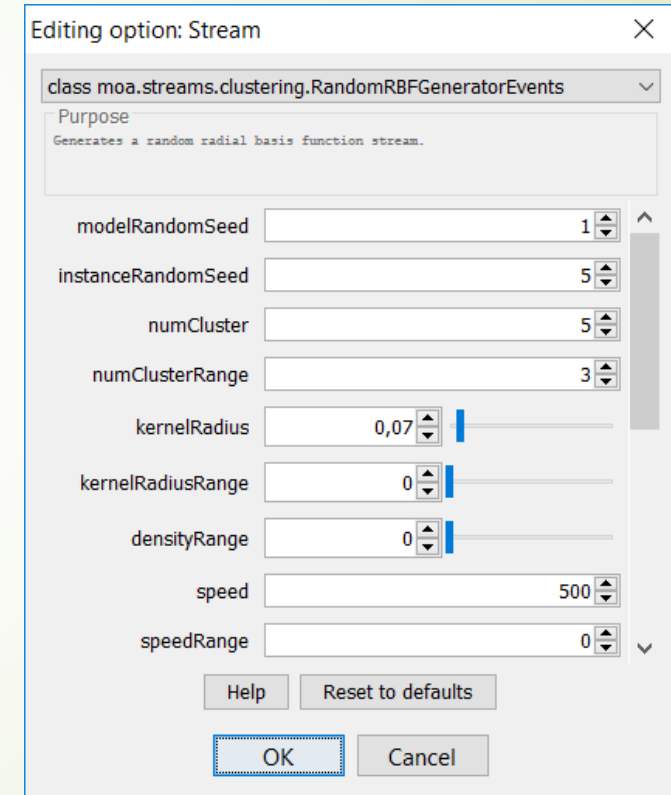
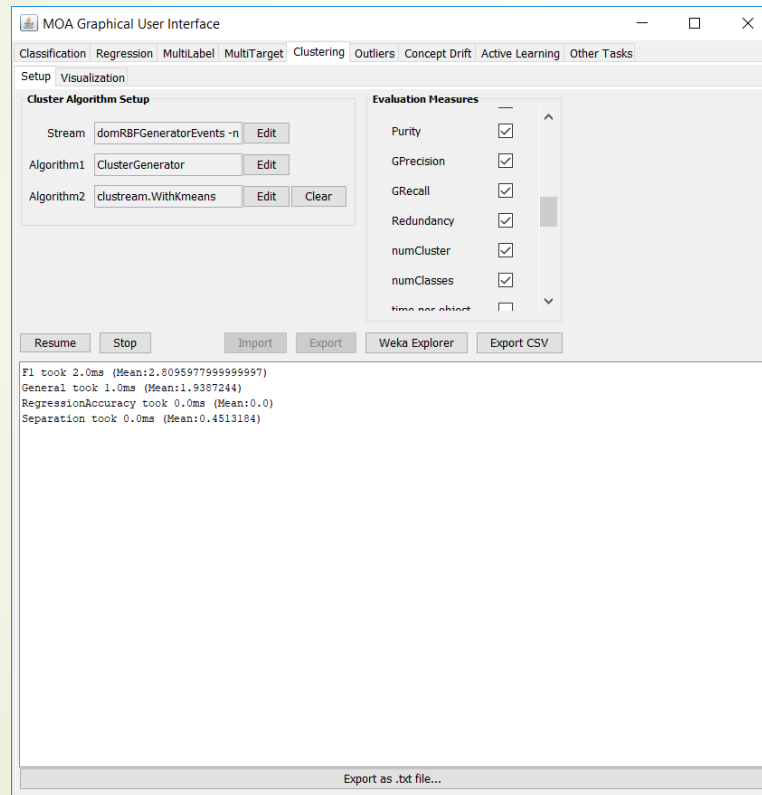
classified instances = 1 000 000
classifications correct (percent) = 84,474
Kappa Statistic (percent) = 76,711
Kappa Temporal Statistic (percent) = 76,708
Kappa M Statistic (percent) = 76,679
model training instances = 1 000 000
model serialized size (bytes) = 1 961 760
tree size (nodes) = 319
tree size (leaves) = 160
active learning leaves = 160
tree depth = 11
active leaf byte size estimate = 12 126,55
inactive leaf byte size estimate = 0.0
byte size estimate overhead = 1,011
PS C:\Users\Agnieszka\Desktop\moa-release-2018.6.0-bin\moa-release-2018.6.0\lib>
```



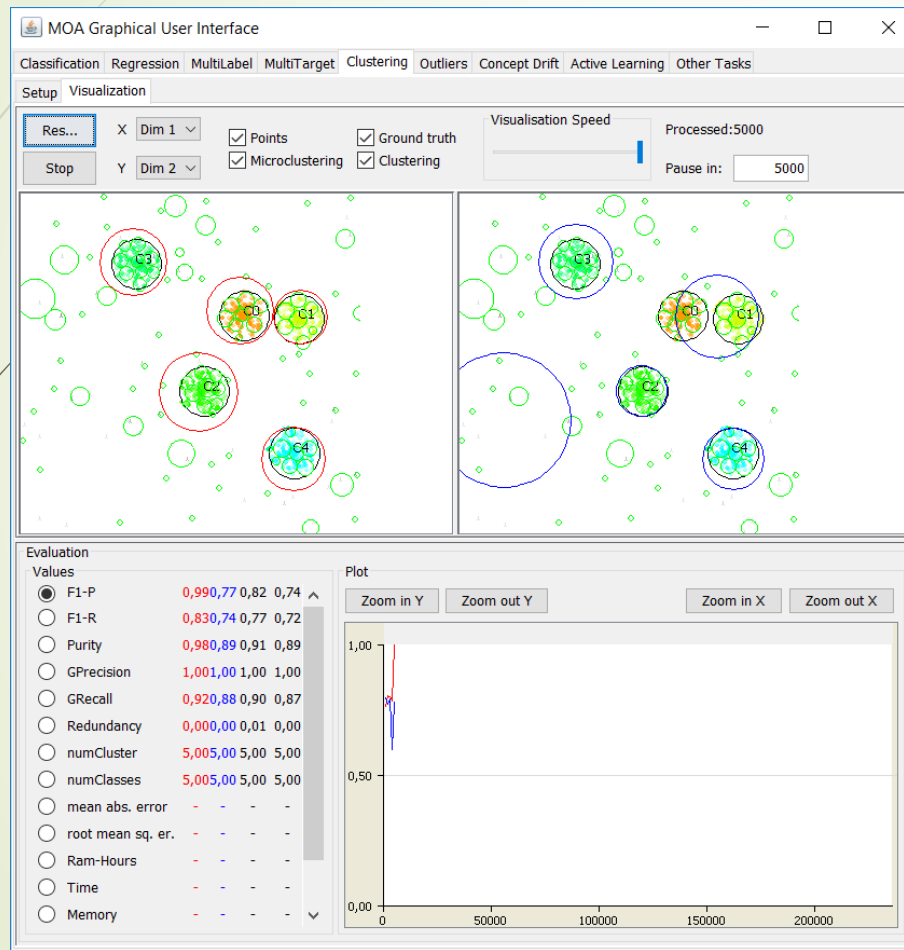
# Klastrowanie

- StreamKM++
  - CluStream
  - ClusTree
  - DenStream
  - CobWeb
  - ...
- 

# Klastrowanie



# Klastrowanie - wizualizacja



- Po lewej Clustream
- Po prawej Kmeans



# Systemy rekomendacyjne

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

# Systemy rekomendacyjne

MOA Graphical User Interface

Classification Regression MultiLabel MultiTarget Clustering Outliers Concept Drift Active Learning Other Tasks

Configure | pl.MovielensDataset -f C:\Users\Agnieszka\Desktop\ml-1m\ratings.dat -s (BRISMPredictor -d MemRecommenderData) | Run

command	status	time elapsed	current activity	% complete
EvaluateOnlineRecommen...	completed	11m24s		100,00
EvaluateOnlineRecommen...	failed	0,03s		100,00

Pause Resume Cancel Delete

Final result Refresh Auto refresh: every second

```
n, RMSE, trainingTime, evalTime
99.0, 1.0121832718163593, 0.0, 0.0
199.0, 1.0160718705991036, 0.0, 0.0
299.0, 1.0593976825083538, 0.0, 0.0
399.0, 1.1289690453400603, 0.0, 0.0
499.0, 1.1129426685813408, 0.0, 0.0
599.0, 1.075992253832316, 0.0, 0.0
699.0, 1.054029095518073, 0.0, 0.0
799.0, 1.0269342523876615, 0.0, 0.0
899.0, 1.0112153092868825, 0.0, 0.0
```

Export as .txt file...

Evaluation Values

Measure	Current	Mean
<input checked="" type="radio"/> Accuracy	-	-
<input type="radio"/> Kappa	-	-
<input type="radio"/> Kappa Temp	-	-
<input type="radio"/> Ram-Hours	-	-
<input type="radio"/> Time	-	-
<input type="radio"/> Memory	-	-

Plot

Zoom in Y Zoom out Y Zoom in X Zoom out X

Configure task

class moa.tasks.EvaluateOnlineRecommender

Purpose  
Evaluates a online recommender system.

dataset | \Agnieszka\Desktop\ml-1m\ratings.dat | Edit

ratingPredictor | MFpredictor -d MemRecommenderData | Edit

sampleFrequency | 100

taskResultFile | Browse

Help Reset to defaults

OK Cancel

# 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 ewolucji

# 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.toArrayCopy();
    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