UNIVERSIDAD DE GRANADA

Escuela de Posgrado
Universidad de Granada

Noise and Instance Selection in Apache Spark

Big Data II

Diego García (djgarcia@decsai.ugr.es) 19/04/2018

Outline

- **Noise**
- Instance Selection
- Spark Packages

What is Noise?

The *most* frequent problem present in data

Partial or complete alteration of an instance

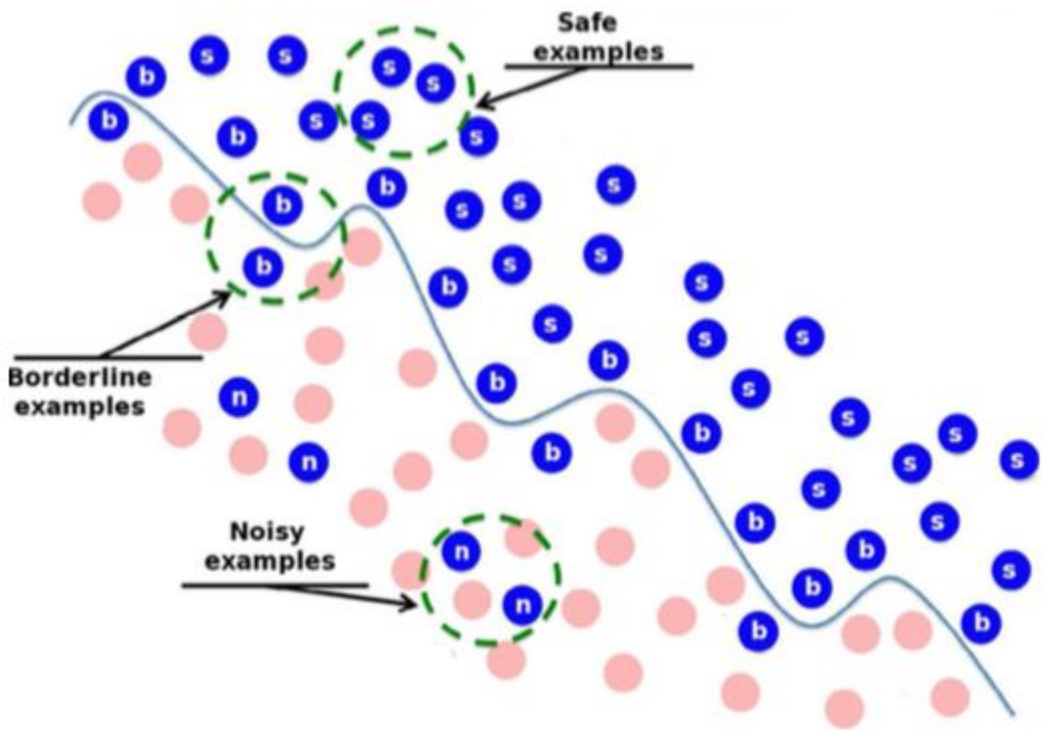
Excessively complex models with deteriorated performance

Class noise and *attribute noise*

What noise does

- Creates small clusters of instances of a particular class in the instance space corresponding to another class
- Displaces or removes instances located in key areas within a concrete class
- Disrupts the boundaries of the classes resulting in an increased boundaries overlap

Noise graphically



Class vs attribute noise

- Class noise takes place when an example is wrongly labeled
- Attribute noise refers to corruptions in the values of the input attributes (erroneous values and MVs)
- Class noise is considered more harmful to the learning process

Approaches

- Algorithm level:
  - Robust classification algorithms
  - Model noise, pruning strategies, dismiss importance of noisy instances
- Data level (*filters*):
  - Strategies to cleanse the dataset
  - Ensembles, partitioning, iteratively filtering noisy instances

Noise in Big Data

- In Big Data there is a special need for noise filtering methods
- The high dimensionality and example size generate accumulated noise
- Noise filters reduce the size of the datasets and improve its quality
- Most of the classic noise filters are not prepared for working in Big Data as they have an iterative approach

Enabling Smart Data: Noise filtering in Big Data classification

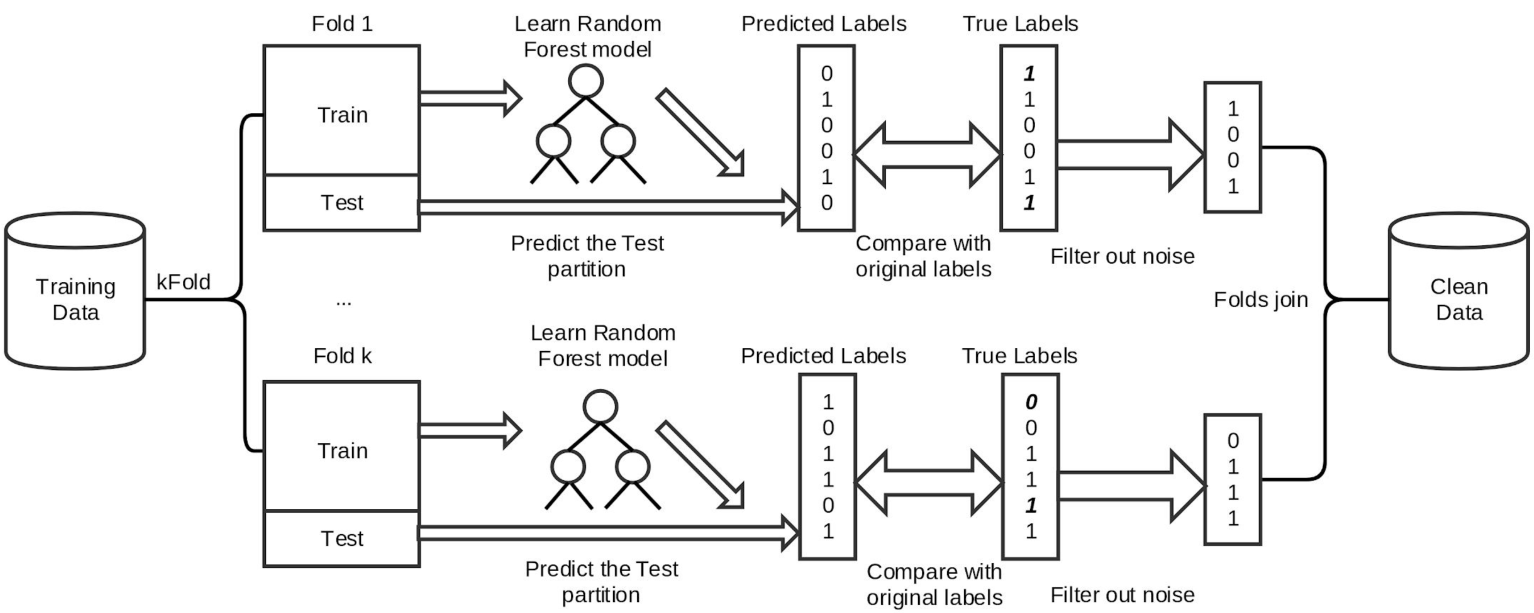
- We have tackled the problem of noise in Big Data classification
- Two noise filters proposed:
  - HME-BD: homogeneous ensemble
  - HTE-BD: heterogeneous ensemble
- First suitable noise filtering approaches in Big Data domains
- Integrated in Spark's MLlib

García-Gil, D., Luengo, J., García, S., & Herrera, F. (2017). Enabling smart data: Noise filtering in big data classification. CoRR, abs/1704.01770 . Retrieved from http://arxiv.org/abs/1704.01770

HME-BD

- Partitioning strategy (ensemble)
- *k*-Fold of the training data (typically 4 or 5)
- Random Forest as classifier
- Predicts only the *k* “test” partitions
- Wrong predicted instances are removed

HME-BD

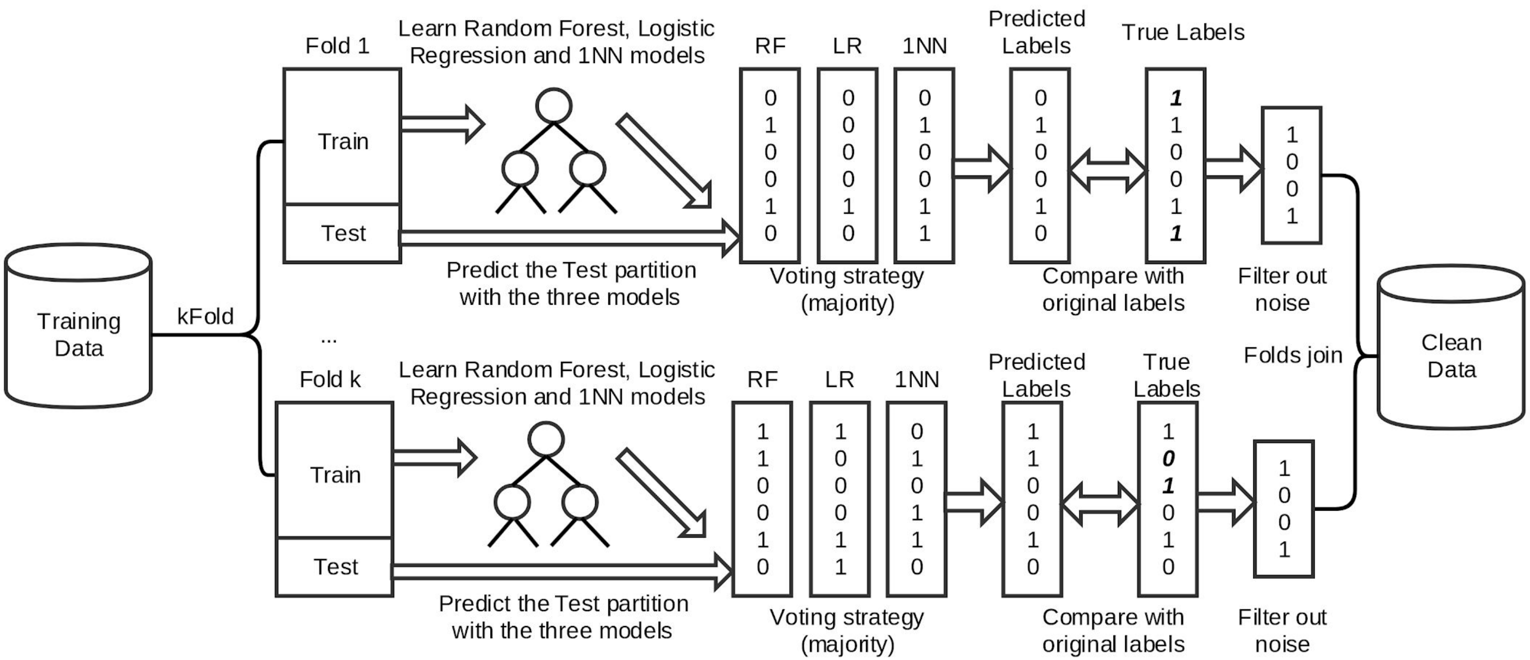


Training Data
kFold
Fold 1
Train
Test
Learn Random Forest model
Predict the Test partition
Predicted Labels
0 1 0 0 1 0
True Labels
1 1 0 0 1 1
Compare with original labels
Filter out noise
1 0 0 1
...
Fold k
Train
Test
Learn Random Forest model
Predict the Test partition
Predicted Labels
1 0 1 1 0 1
True Labels
0 0 1 1 1 1
Compare with original labels
Filter out noise
0 1 1 1
Folds join
Clean Data

HTE-BD

- Same workflow as HME-BD
- Random Forest, 1-NN and Logistic Regression as classifiers
- Two voting strategies: majority and consensus
- Noisy instances are removed according to the voting strategy

HTE-BD


Training Data
kFold
Fold 1
Train
Test
Learn Random Forest, Logistic Regression and 1NN models
Predict the Test partition with the three models
RF
LR
1NN
0 1 0 0 1 0
0 0 0 0 1 0
0 1 0 0 1 1
Voting strategy (majority)
Predicted Labels
0 1 0 0 1 0
True Labels
1 1 0 0 1 1
Compare with original labels
1 0 0 1
Filter out noise
...
Fold k
Train
Test
Learn Random Forest, Logistic Regression and 1NN models
Predict the Test partition with the three models
RF
LR
1NN
1 1 0 0 1 0
1 0 0 0 1 1
0 1 0 1 1 0
Voting strategy (majority)
Predicted Labels
1 1 0 0 1 0
True Labels
1 0 1 0 1 0
Compare with original labels
1 0 0 1
Filter out noise
Folds join
Clean Data

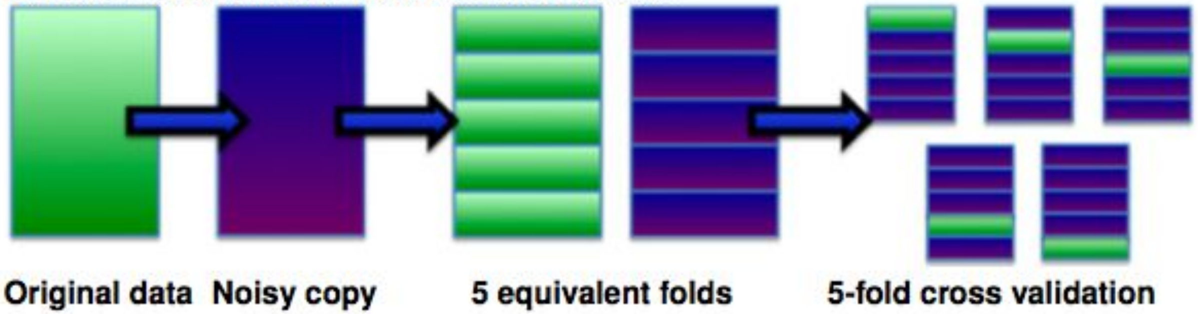
Datasets

0%, 5%, 10%, 15%, 20%

Table 1: Datasets used in the analysis

Dataset	Instances	Atts.	Total	CL
SUSY	5,000,000	18	90,000,000	2
HIGGS	11,000,000	28	308,000,000	2
Epsilon	500,000	2,000	1,000,000,000	2
ECBDL14	1,000,000	631	631,000,000	2

Noise introduction into the datasets:



Parameters

Table 3: Parameter setting for the classifiers

Classifier	Parameters
KNN	K = 1, distance = "euclidean"
Decision Tree	impurity = "gini", maxDepth = 20 and maxBins = 32

Results 1NN

Table 4: KNN test accuracy. The highest accuracy value per dataset and noise level is stressed in bold

Dataset P Vote	Noise (%)	Original	HME-BD 4	 5	HTE-BD 4 Majority	 4 Consensus	 5 Majority	 5 Consensus	ENN-BD
SUSY	0	71.79	**78.73**	78.72	77.86	74.64	77.88	74.65	72.02
	5	69.62	78.68	**78.69**	77.68	73.38	77.68	73.39	69.84
	10	67.44	**78.63**	78.62	77.44	72.01	77.46	72.00	67.66
	15	65.27	**78.62**	78.61	77.19	70.52	77.20	70.53	65.28
	20	63.10	78.56	**78.58**	76.93	69.10	76.93	69.04	63.25
HIGGS	0	61.21	**64.26**	64.25	63.94	62.30	63.93	62.23	60.65
	5	60.10	64.06	**64.07**	63.63	61.45	63.62	61.44	59.60
	10	58.97	63.83	**63.84**	63.29	60.65	63.24	60.66	58.56
	15	57.84	**63.65**	63.64	62.86	59.81	62.89	59.81	57.52
	20	56.69	**63.53**	63.40	62.55	58.89	62.55	58.85	56.45
Epsilon	0	56.55	**58.11**	58.06	57.43	55.19	57.39	55.40	56.21
	5	55.71	**58.64**	58.60	57.47	55.47	57.39	55.41	55.43
	10	55.20	58.51	**58.61**	57.26	55.25	57.26	55.25	54.79
	15	54.54	58.39	**58.41**	57.00	55.00	57.02	55.03	54.30
	20	54.05	58.02	**58.09**	56.75	54.72	56.71	54.72	53.68
ECBDL14	0	74.83	**76.06**	76.03	75.12	73.54	75.14	73.46	73.94
	5	72.36	**75.60**	75.59	74.59	72.89	74.59	72.84	72.77
	10	69.86	75.31	**75.32**	74.19	72.50	74.19	72.47	71.40
	15	67.39	75.11	**75.12**	73.99	72.11	74.01	72.06	69.68
	20	64.90	74.82	**74.83**	73.70	71.89	73.70	71.90	67.64

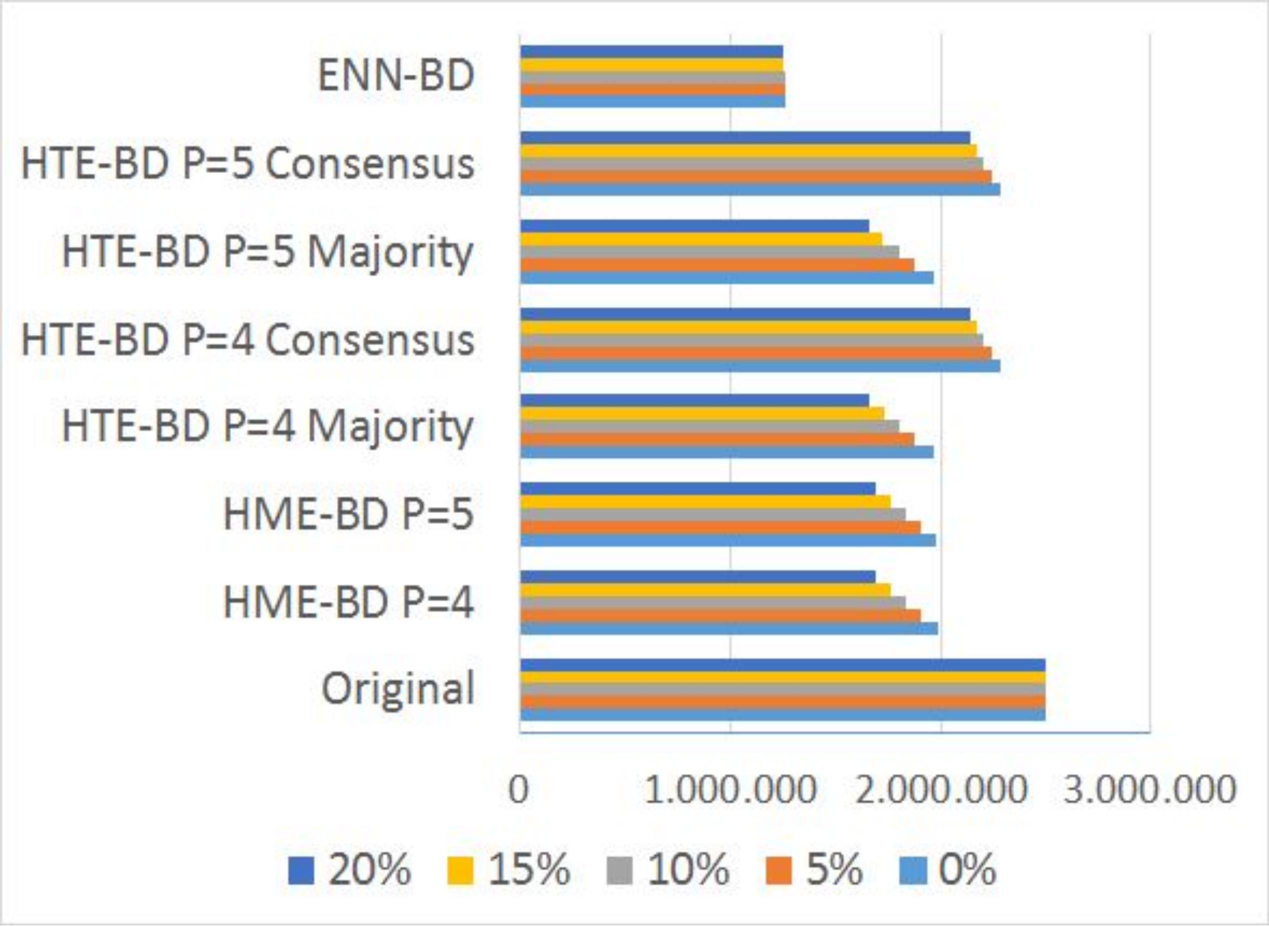
Results Decision Tree

Table 5: Decision tree test accuracy. The highest accuracy value per dataset and noise level is stressed in bold

Dataset P Vote	Noise (%)	Original	HME-BD 4	 5	HTE-BD 4 Majority	 4 Consensus	 5 Majority	 5 Consensus	ENN-BD
SUSY	0	80.24	79.78	79.79	79.69	80.27	79.17	**80.29**	78.56
	5	79.94	79.99	79.97	80.07	**80.36**	80.10	80.34	77.49
	10	79.15	79.85	79.84	79.81	80.04	79.81	**80.22**	77.00
	15	78.21	**79.81**	79.80	79.32	79.47	79.61	79.48	75.81
	20	77.09	79.71	**79.73**	79.35	78.95	79.31	79.41	74.21
HIGGS	0	70.17	71.16	**71.17**	69.61	70.41	69.68	70.33	68.85
	5	69.61	**71.14**	71.11	69.34	69.98	69.36	69.92	68.29
	10	69.22	**71.06**	71.04	68.95	69.56	68.97	69.58	67.52
	15	68.65	**71.03**	70.99	68.52	69.04	68.65	69.06	66.93
	20	67.82	**71.05**	71.02	68.18	68.38	68.35	68.39	66.05
Epsilon	0	62.39	**66.86**	66.19	65.13	66.07	65.11	66.02	61.54
	5	61.10	66.64	**66.83**	65.32	66.09	65.33	66.09	60.41
	10	60.09	66.87	**67.00**	65.46	66.11	65.47	66.10	59.20
	15	59.02	66.62	**66.85**	65.33	65.99	65.29	66.00	58.09
	20	57.73	66.46	**66.79**	65.08	65.69	64.98	65.65	56.71
ECBDL14	0	73.98	74.59	74.38	74.21	74.51	74.35	**74.62**	73.66
	5	72.87	74.64	74.40	74.16	74.54	74.25	**74.75**	73.48
	10	71.67	74.59	74.25	73.84	74.51	73.94	**74.63**	72.75
	15	70.28	**74.61**	74.22	73.82	73.91	73.98	74.10	71.68
	20	68.66	**74.83**	74.18	73.78	73.82	73.85	73.86	70.16

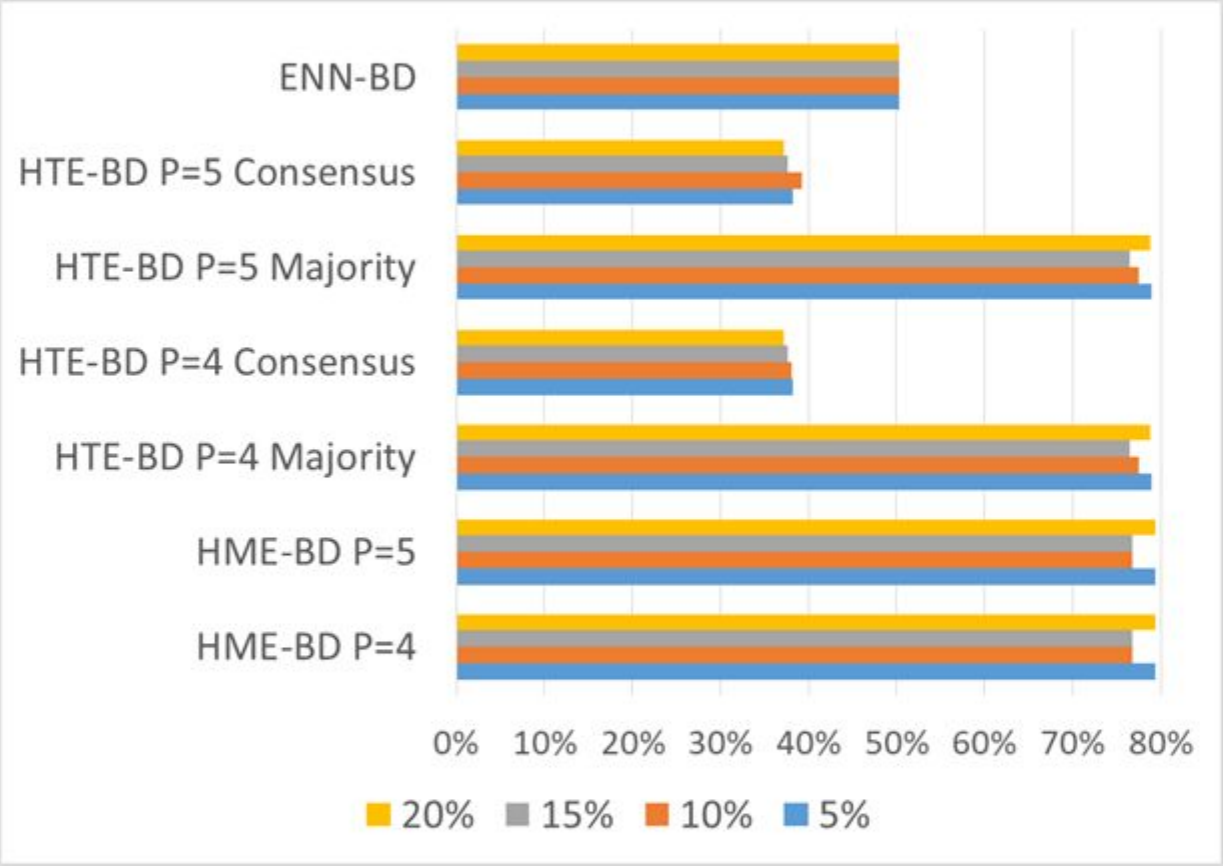
Instances Removed

Susy

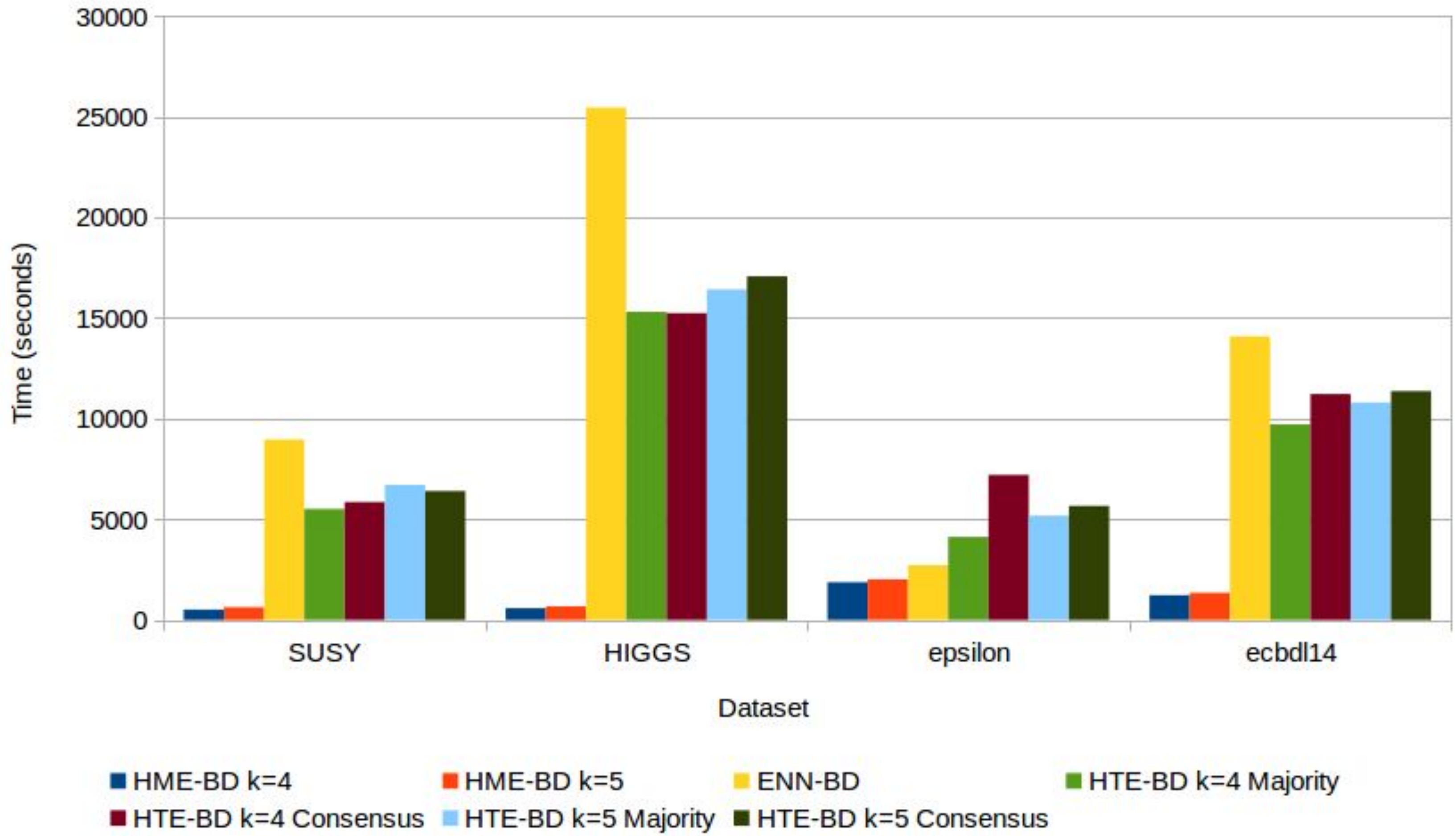


Correctly Removed Instances

Susy



Runtime



30000
25000
20000
15000
10000
5000
0
Time (seconds)
SUSY
HIGGS
epsilon
ecbdl14
Dataset
HME-BD k=4
HME-BD k=5
ENN-BD
HTE-BD k=4 Majority
HTE-BD k=4 Consensus
HTE-BD k=5 Majority
HTE-BD k=5 Consensus

Outline

- Noise
- **Instance Selection**
- Spark Packages

What is Data Reduction?

- Data Reduction is the set of techniques devoted to reducing the size of the original data whilst retaining as much information as possible
- Two main approaches:
  - Reduce attributes (features selection/extraction)
  - **Reduce instances** (instance selection/generation)

Objective of IS and IG

- Obtain a subset $SS \subset TR$ such that SS does not contain redundant or noisy examples and Acc(SS) ~= Acc(TR)

- IG methods may generate artificial data points if needed for a better representation of the training set

Prototype Selection

- Prototype Selection (PS) methods are IS methods that use an instance-based classifier with a distance measure, commonly k-NN, for finding a representing subset of the training set

- Examples: FCNN, RMHC, MR-DIS

Evolutionary PS

- The IS problem can be seen as a binary optimisation problem which consists of whether or not to select a training example
- The fitness function usually consists of classifying the whole training set using the k-NN algorithm
- Example: SSMA

Prototype Generation

- Another approach to perform instance reduction is IG, also called Prototype Generation (PG) in the case of instance-based classifiers
- Most popular strategy is to use merging of nearest examples to set the new artificial samples

Hybrid Approaches

- Combination of IS and IG (or PS and PG)
- PS is used for selecting the most representative subset of the training data, and PG is tasked to improve this subset by modifying the values of the instances
- Exmaple: SSMA-SFLSDE

Outline

- Noise
- Instance Selection
- **Spark Packages**

HME-BD & HTE-BD

Available in Spark Packages:

https://spark-packages.org/package/djgarcia/NoiseFramework

NoiseFramework (homepage)

Noise Framework for removing noisy instances with three algorithms: HME-BD, HTE-BD and ENN.

@djgarcia / ★★★★★ (2)

In this framework, two Big Data preprocessing approaches to remove noisy examples are proposed: an homogeneous ensemble (HME_BD) and an heterogeneous ensemble (HTE_BD) filter. A simple filtering approach based on similarities between instances (ENN_BD) is also implemented.

HME_BD, HTE_BD & ENN_BD

IS with kNN

Available in Spark Packages:

https://spark-packages.org/package/djgarcia/SmartReduction

SmartReduction (homepage)

Smart Reduction framework for Big Data

@djgarcia / ★★★★★ (2)

This framework implements four distance based Big Data preprocessing algorithms for prototype selection and generation: FCNN_MR, SSMASFLSDE_MR, RMHC_MR, MR_DIS, with special emphasis in their scalability and performance traits.

FCNN_MR, SSMASFLSDE_MR, RMHC_MR & MR_DIS

Noise Filtering with kNN

Available in Spark Packages:

https://spark-packages.org/package/djgarcia/SmartFiltering

SmartFiltering (homepage)

Smart Filtering framework for Big Data

@djgarcia / (2)

This framework implements four distance based Big Data preprocessing algorithms to remove noisy examples: ENN_BD, AllKNN_BD, NCNEdit_BD and RNG_BD filters, with special emphasis in their scalability and performance traits.

AllKNN_BD, NCNEdit_BD & RNG_BD

UNIVERSIDAD DE GRANADA

Escuela de Posgrado
Universidad de Granada

APACHE Spark™

Noise and Instance Selection in Apache Spark

Big Data II

Diego García (djgarcia@decsai.ugr.es) 19/04/2018