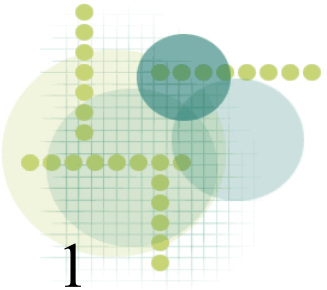
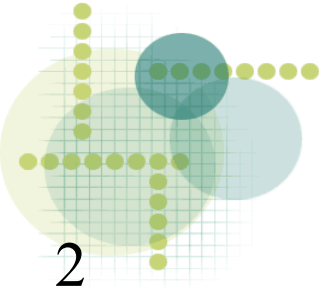
Klasyfikacja za pomocą kombinacji jądrowych estymatorów gęstości z wykorzystaniem informacji o gradiencie funkcji błędu

Mateusz Kobos, 13.05.2009
Seminarium Metody Inteligencji Obliczeniowej



Spis treści

- Działanie algorytmu
- Modyfikacje algorytmu
  - 2 niezależne estymatory
  - Funkcja błędu: uśrednianie z 10 powtórzeń
  - Funkcja błędu: information loss



Działanie algorytmu - klasyfikacja/odtwarzanie

- Dla punktu testowego:
  - Estymuj gęstość każdej z klas punkcie (korzystając z parametrów obliczonych podczas nauki)
    - Do estymacji gęstości używamy kombinacji estymatorów jądrowych
  - Zwróć etykietę klasy odpowiadającej największej gęstości (obliczamy korzystając ze wzoru Bayesa)

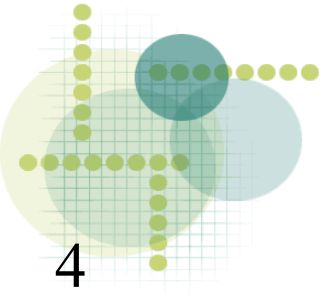
$$d_B(\mathbf{x}) = \arg\max_{w_i} \hat{P}(\omega_i|\mathbf{x}) = \arg\max_{w_i} \frac{\hat{p}(\mathbf{x}|\omega_i)\hat{P}(\omega_i)}{\hat{p}(\mathbf{x})}$$

Kombinacja

- Szerokości jąder każdego z estymatorów są połączone za pomocą funkcji wykładniczej:

$$h_j(a) = h_{\min} + a^j(h_{\max} - h_{\min})$$

- Gdzie:
  - *h_j* – szerokość jądra estymatora *j*
  - *[h_min, h_max]* – przedział sensownych szerokości jądra
  - *a* należy do *[0, 1]*



Uczenie algorytmu

- Minimalizacja funkcji błędu klasyfikacji ze względu na parametr estymatora gęstości

$$\mathrm{MSE}(\hat{P}(\cdot), \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \sum_{i=1}^{c} (\hat{P}(\omega_i|\mathbf{x}) - \mathbf{t}_i(\mathbf{x}))^2$$

- a - estimator's parameter
- $\mathcal{D}^v$ - validation set
- c - number of classes
- $\hat{p}(\omega_i|\mathbf{x}; a)$ - estimation of class ω_i probability in point $\mathbf{x}$, is equal to a
- $\mathbf{t}(\mathbf{x})[i]$ - actual value of point $\mathbf{x}$ probability of class ω_i

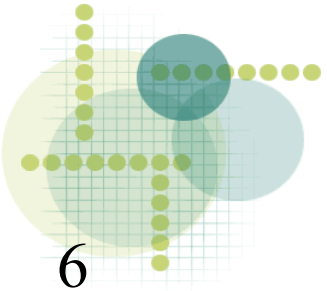
Uczenie - dokładniej

1)Losowa permutacja wszystkich instancji zbioru treningowego

2)Transformacja danych (np. standaryzacja, PCA)

3)Obliczenie *[h_min, h_max]*

4)minimalizacja funkcji błedu



Optymalizacja z wykorzystaniem pochodnej

- Algorytm optymalizacyjny: L-BFGS-B [Zhu97]
  - pseudo-Newtonowski, znajduje optimum lokalne
  - Przeznaczony do rozwiązywania dużych, nieliniowych problemów optymalizacyjnych, z ograniczoną pojemnością pamięci
  - Każda ze zmiennych może być ograniczona ($l \leq x_i \leq u$), tutaj: $0 \leq a \leq 1$
  - W każdym kroku wymaga jednoczesnego obliczenia wartości funkcji i pochodnej
- Co optymalizujemy: funkcję błędu klasyfikacji (MSE)

Zalety i wady optymalizacji z wykorzystaniem pochodnej

- Zalety
  - Poszukiwanie minimum trwa o ok. 75% krócej niż w metodzie brutalnej (dla 1 parametru)
- Wady
  - Trzeba ustalać punkt startowy
    - my przyjmujemy x0=(1, 1.1)
  - Znajdowane jest minimum lokalne

2 niezależne estymatory

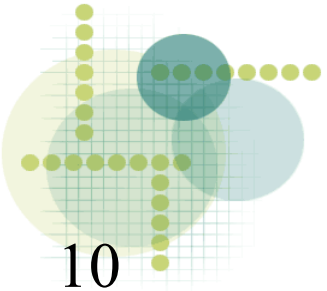
- Do tej pory szerokość badanych estymatorów była połączona za pomocą zależności funkcyjnej:

$$h_j(a) = h_{\min} + a^j(h_{\max} - h_{\min})$$

- Teraz rezygnujemy z tworzenia takiego arbitralnego ograniczenia: szerokość współczynnika wygładzania dla każdego estymatora dobieramy niezależnie
  - Badamy model z 2 estymatorami (i uśrednianiem) - model z 1 estymatorem jest jego szczególnym przypadkiem (pod względem doboru parametrów)
  - Minimum będziemy tutaj szukać za pomocą algorytmu optymalizacji gradientowej (L-BFGS-B)

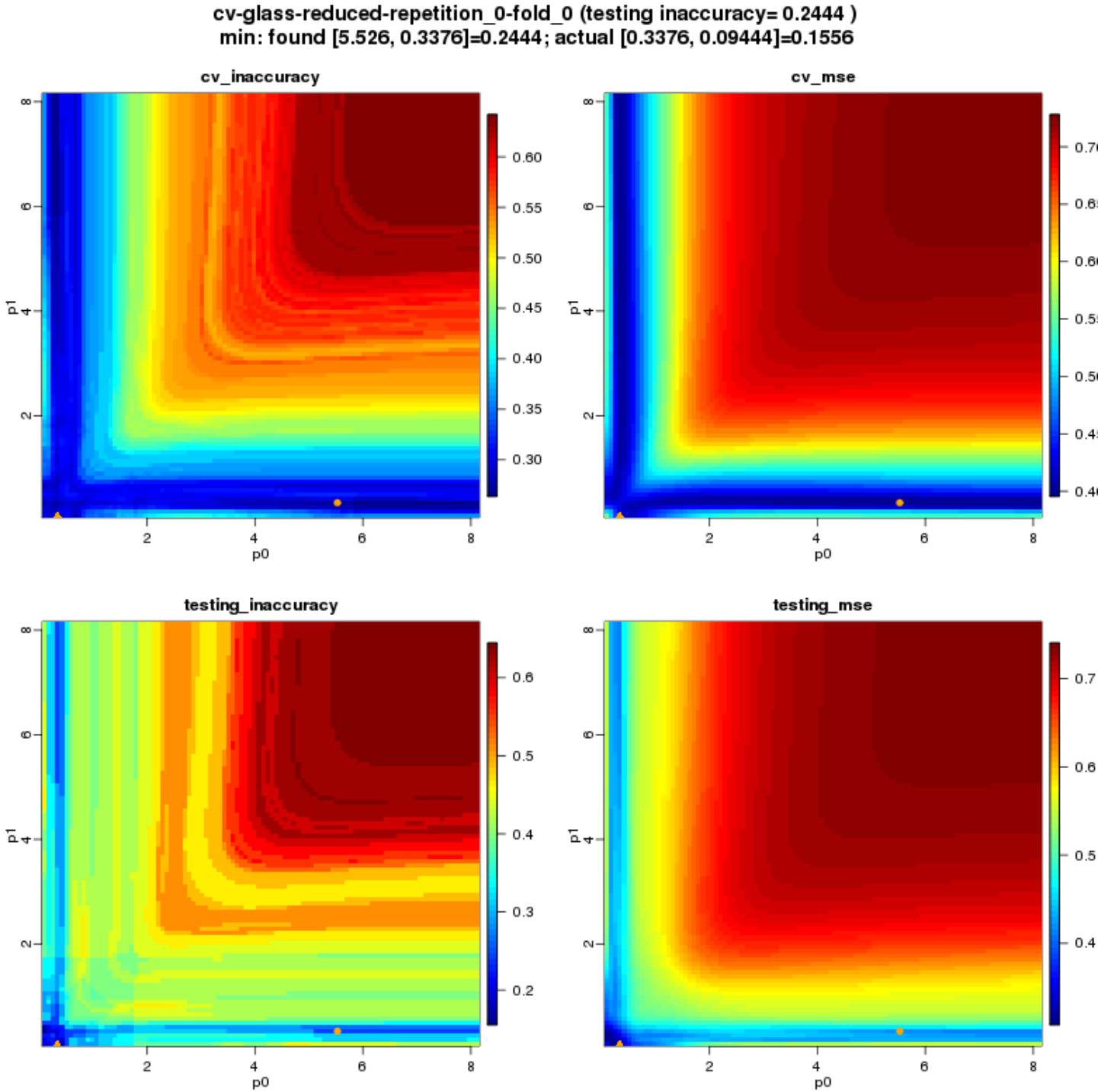
2 niezależne estymatory

- Podczas minimalizacji:
  - Minimalizujemy błąd MSE na zbiorze treningowym, który jest przybliżeniem
  - błędu klasyfikacji (errror rate) na zbiorze treningowym, który jest przybliżeniem
  - Błędu klasyfikacji (error rate) na zbiorze testowym



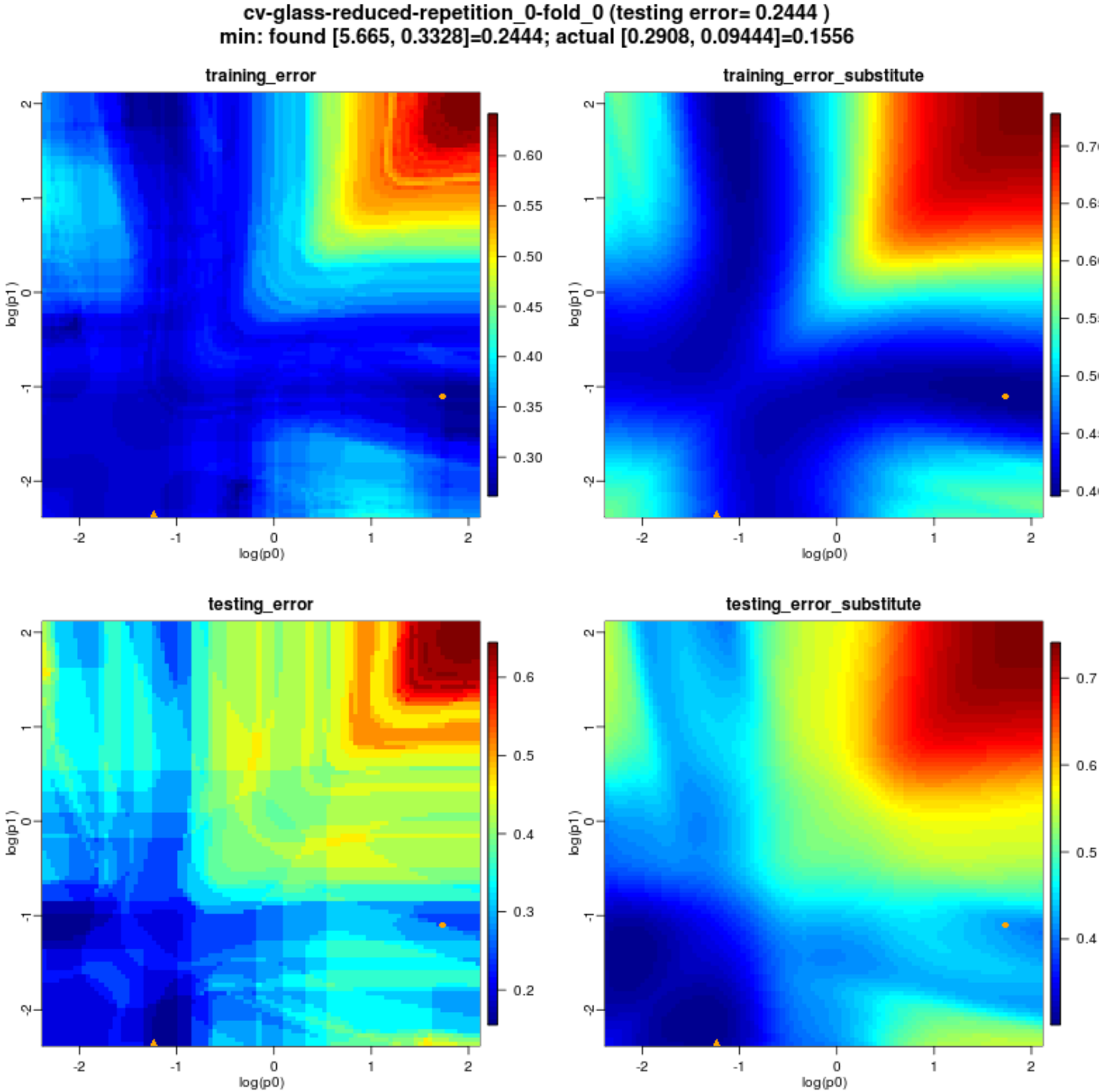
2 niezależne estymatory

- Zbiór glass-reduced: minimum na przekątnej – proponowany model nie poprawi wyników



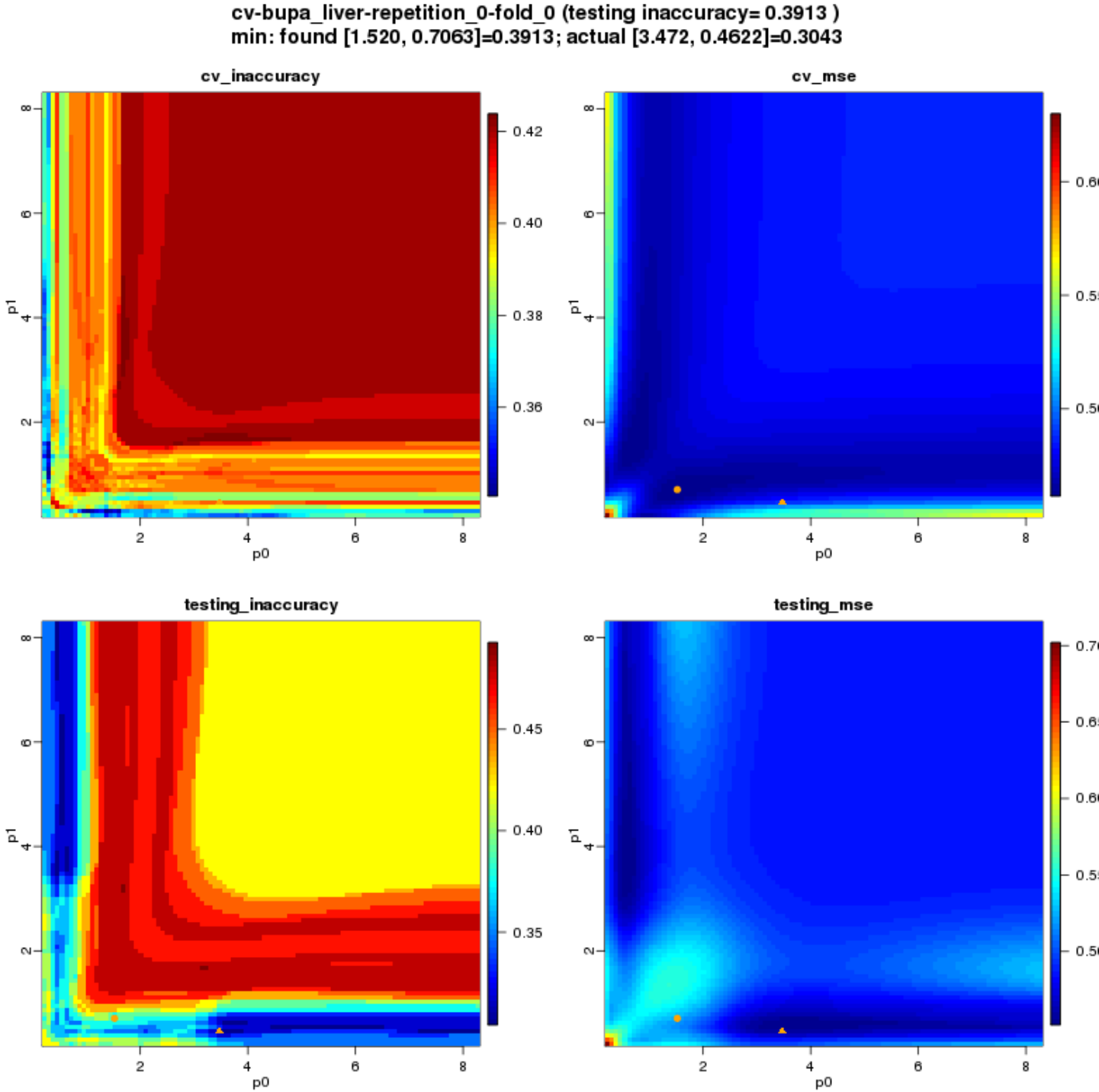
2 niezależne estymatory

- Zbiór glass-reduced: po zlogarytmowaniu - minimum nie jest na przekątnej, ale na zbiorze uczącym go nie widać



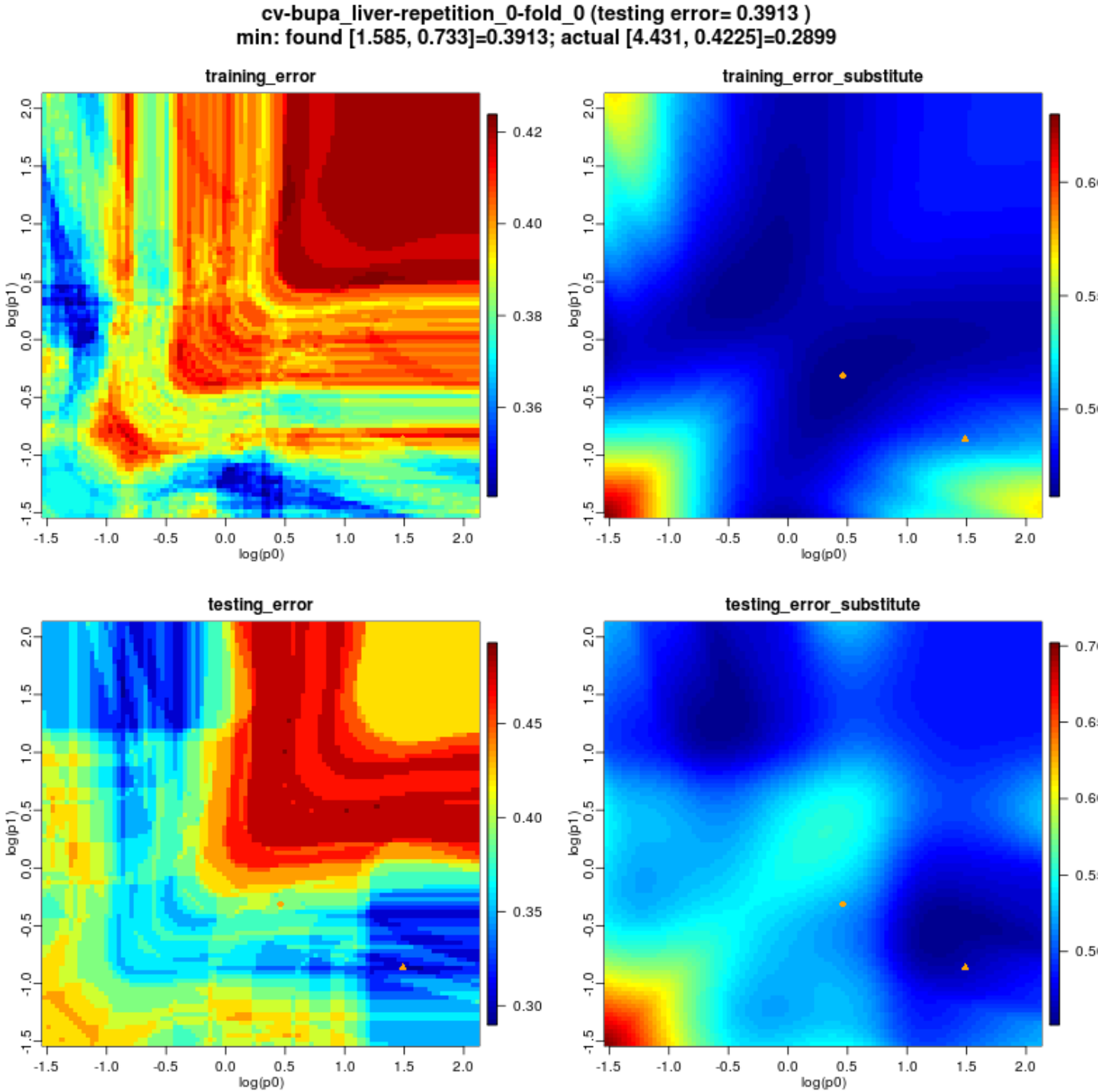
2 niezależne estymatory

- Zbiór bupa_liver: minimum nie jest na przekątnej – proponowany model może dać lepszy wynik



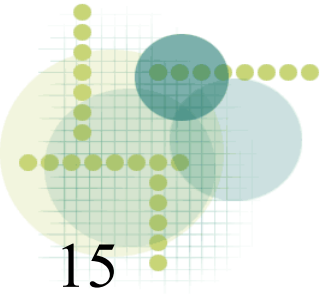
2 niezależne estymatory

- Zbiór bupa_liver: po zlogarytmowaniu



2 niezależne estymatory

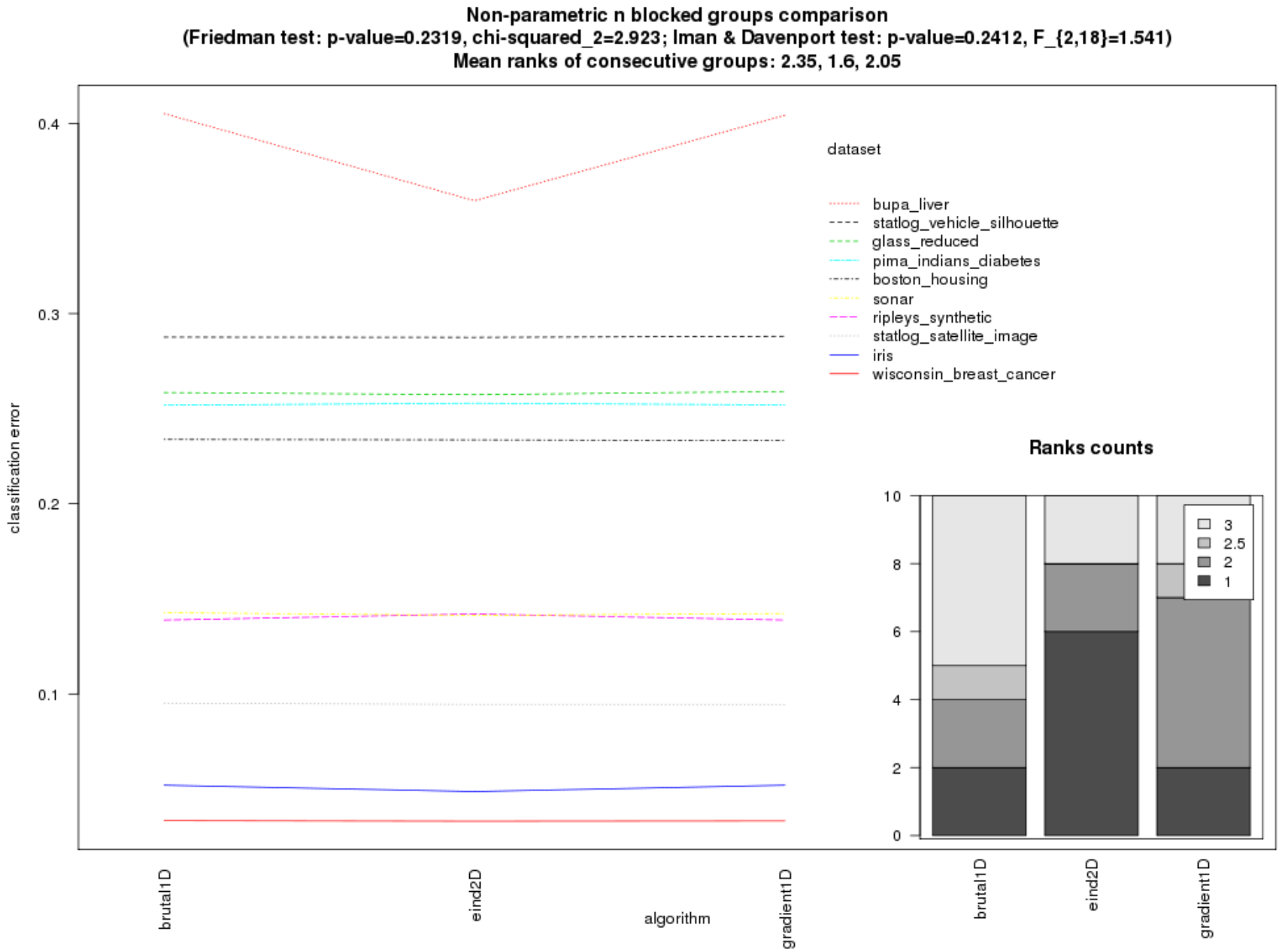
- Porównujemy algorytmy:
  - *eind2D* – 2 estymatory o niezależnie dobieranych parametrach, parametry dobierane za pomocą optymalizacji gradientowej
  - *brutal1D* – 1 estymator, parametr dobierany metodą brutalną
  - *gradient1D* – 1 estymator, parametr dobierany za pomocą optymalizacji gradientowej



2 niezależne estymatory

- Pytanie: Czy przynajmniej jeden z algorytmów uzyskuje wyniki lepsze od innych?
- Eksperyment:
  - Eksperyment: dla każdego algorytmu i zbioru danych: 10x 10-fold cross-validation i obliczamy średnią
  - Test: test rangowy Iman&Davenport (modyfikacja testu Friedmana) – zgodnie z metodologią z [Demsar06]
- Odpowiedź: nie ma statystycznie istotnych różnic między badanymi algorytmami

2 niezależne estymatory



2 niezależne estymatory

- Pytanie: Skoro żaden z algorytmów nie jest ogólnie lepszy, to może jest lepszy chociaż na paru zbiorach danych?
- Eksperyment:
  - Eksperyment: dla każdego algorytmu i zbioru danych: 10x 10-fold cross-validation
  - Test: corrected resampled paired t-test ([Nadeau03])
- Oznaczenia:
  - p.value: p-value dla corrected resampled paired t-test
  - t.p.value: p-value dla zwykłego paired t-test
  - mean: różnica między wartościami średnimi

Testy wielokrotne - uwaga

- Przy jednoczesnym rozważaniu wielu hipotez, p-value też należy rozważać razem
  - Zgodnie z (b. konserwatywną) zasadą Bonferroni'ego: adjusted p-value = p-value * (liczba rozważanych hipotez)

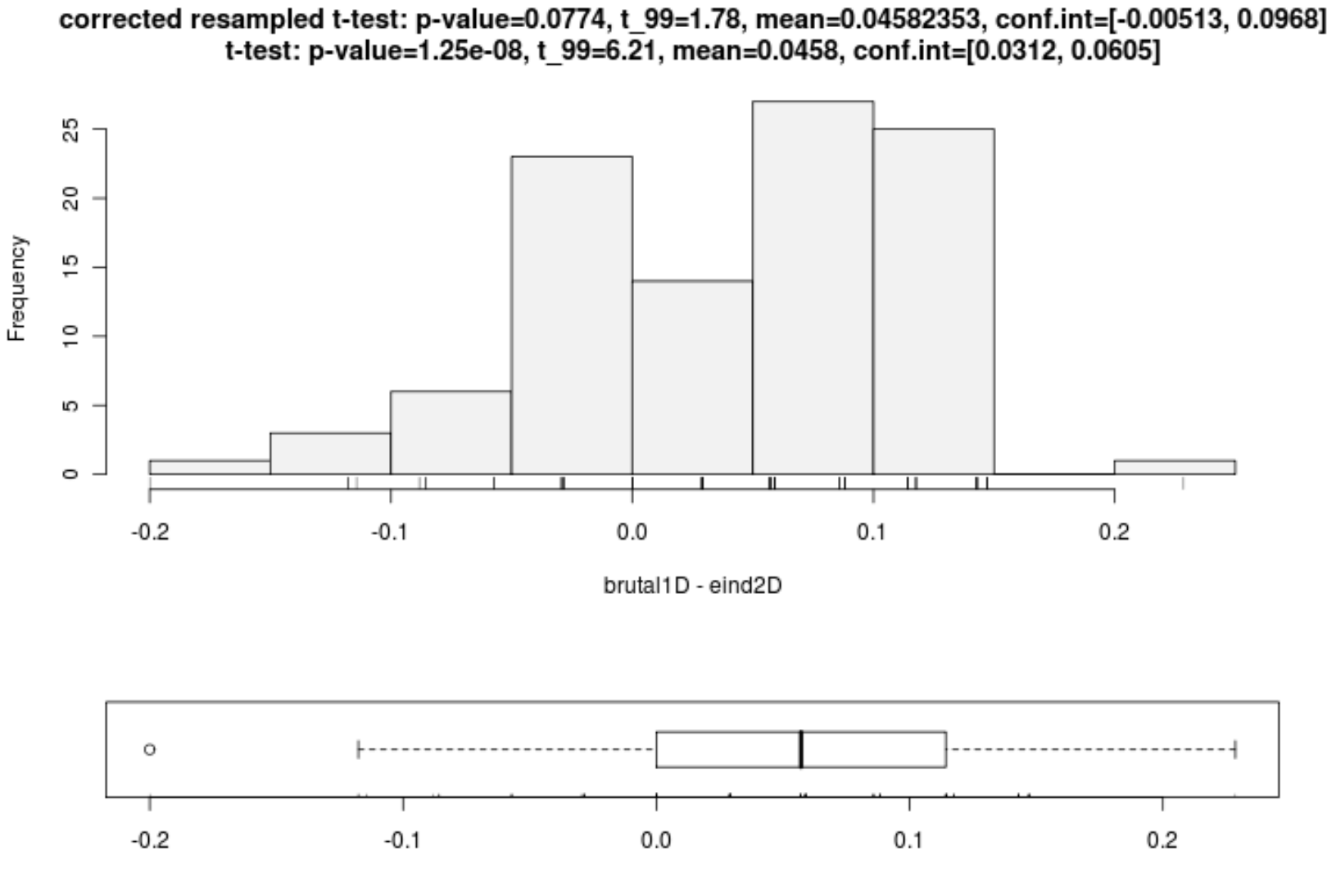
2 niezależne estymatory

- mean = brutal1D - eind2D

	t.p.value	p.value	mean
"boston_housing"	0.815	0.946	0.000
"bupa_liver"	0.000	0.077	0.046
"glass-reduced"	0.840	0.954	0.001
"iris"	0.058	0.583	0.003
"pima_indians_diabetes"	0.481	0.839	-0.001
"ripleys_synthetic"	0.282	0.757	-0.003
"sonar"	0.336	0.782	0.001
"statlog_satellite_image"	0.132	0.663	0.001
"statlog_vehicle_silhouette	0.888	0.968	0.000
"wisconsin_breast_cancer	0.419	0.816	0.000

- Wniosek: brak istotnych różnic (prawie istotna różnica dla „bupa_liver”)

2 niezależne estymatory - porównanie na bupa_liver



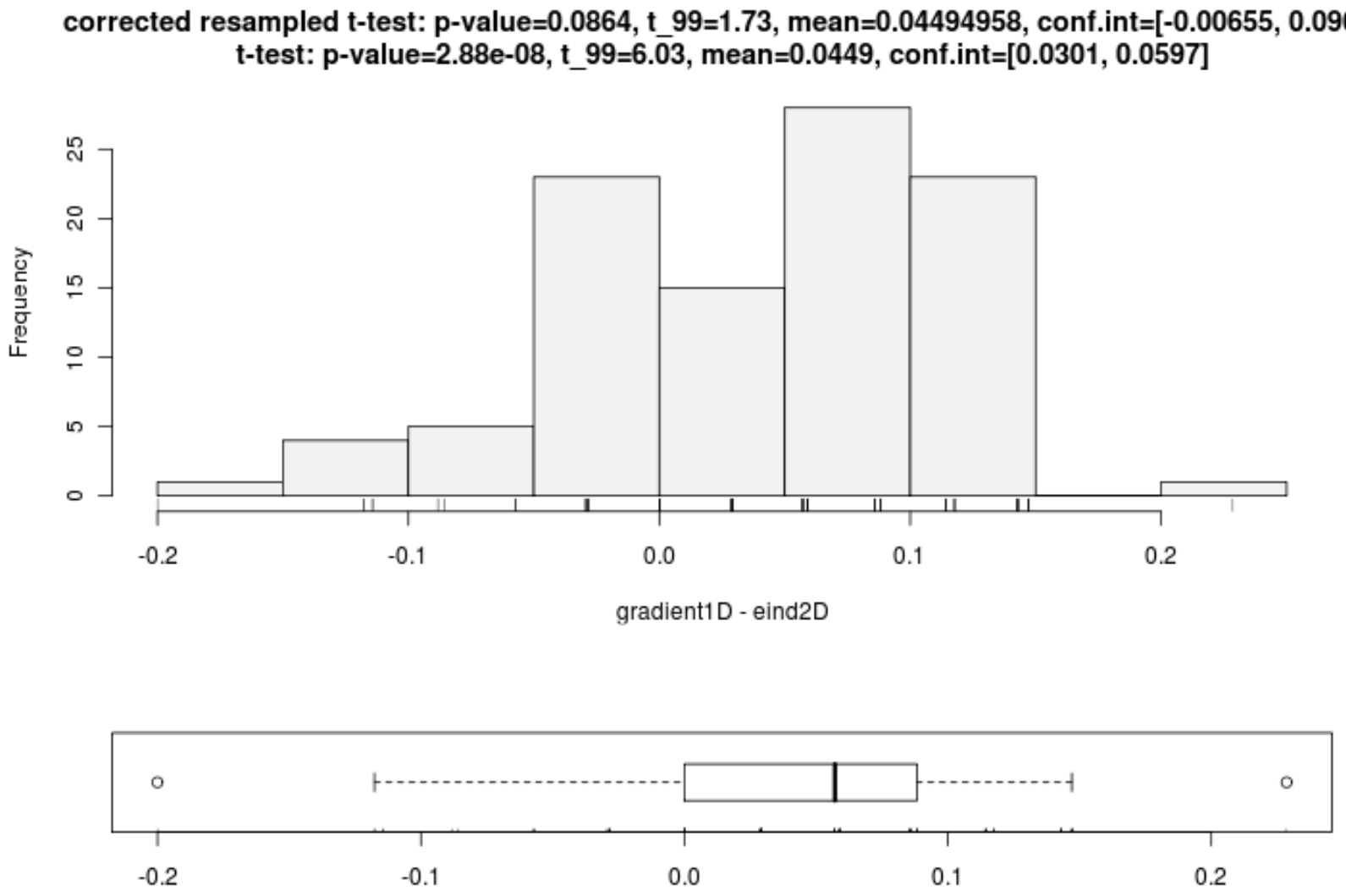
2 niezależne estymatory

- mean = gradient1D - eind2D

	t.p.value	p.value	mean
boston_housing	0.606	0.882	0.000
bupa_liver	0.000	0.086	0.045
glass-reduced	0.765	0.932	0.002
iris	0.058	0.583	0.003
pima_indians_diabetes	0.513	0.851	-0.001
ripleys_synthetic	0.260	0.745	-0.003
sonar	0.320	0.774	0.001
statlog_satellite_image	0.847	0.956	0.000
statlog_vehicle_silhouette	0.163	0.687	0.001
wisconsin_breast_cancer	0.566	0.869	0.000

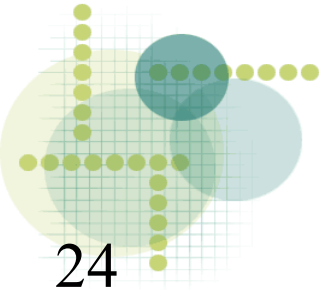
- Wniosek: brak istotnych różnic (prawie istotna różnica dla „bupa_liver”)

2 niezależne estymatory - porównanie na bupa_liver



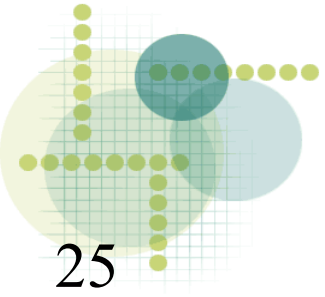
2 niezależne estymatory - wnioski

- Ogólny wniosek: nie można stwierdzić, że zaproponowana modyfikacja jest lepsza od wersji z jednym estymatorem
  - Potencjalna przyczyna: brak odpowiedniości między funkcją błędu na zb. treningowym a testowym
  - Można zbadać dokładniej zbiór bupa_liver (na którym wyniki są „prawie” istotne)



10 x 10-fold cross-validation

- Pomysł na zwiększenie odpowiedniości między błędem na zb. treningowym a testowym: funkcję błędu obliczać dla 10 podziałów cross-walidacyjnych i uśredniać
- Pytanie: Czy modyfikacja jest lepsza na pewnych zbiorach danych?
- Eksperyment: tak jak poprzednio
- Oznaczenia: tak jak poprzednio



10 x 10-fold cross-validation

- mean = eind2D - 10x10-fold CV eind2D

	t.p.value	p.value	mean
boston_housing	0.605	0.882	0.001
bupa_liver	0.895	0.970	-0.001
glass-reduced	0.921	0.977	0.000
iris	0.158	0.684	-0.001
pima_indians_diabetes	1.000	1.000	0.000
ripleys_synthetic	0.533	0.858	-0.001
sonar	0.566	0.869	0.000
statlog_satellite_image	NA	NA	NA
statlog_vehicle_silhouette	0.019	0.495	-0.002
wisconsin_breast_cancer	0.412	0.814	0.000

- Wniosek: brak istotnych różnic

10 x 10-fold cross-validation

- mean = gradient1D - 10x10-fold CV gradient1D

	t.p.value	p.value	mean
boston_housing	0.514	0.851	0.001
bupa_liver	0.342	0.784	0.004
glass-reduced	0.544	0.861	0.001
iris	0.320	0.774	0.001
pima_indians_diabetes	0.493	0.844	-0.001
ripleys_synthetic	0.688	0.908	-0.001
sonar	0.985	0.996	0.000
statlog_satellite_image	NA	NA	NA
statlog_vehicle_silhouette	0.208	0.716	-0.001
wisconsin_breast_cancer	0.159	0.684	0.001

- Wniosek: brak istotnych różnic

10 x 10-fold cross-validation

- Ogólny wniosek: 10-krotne powtórzenie szacowania cross-validation nie zmienia znacząco wyników,
    - => nie należy stosować tej metody (bo jest czasochłonna)

Funkcja błędu: Information loss

- Pomysł na zwiększenie odpowiedniości między błędem na zb. treningowym a testowym: zamiast błędu MSE obliczać information loss:

Dla danego przykładu:

$$-log_2 p_i$$

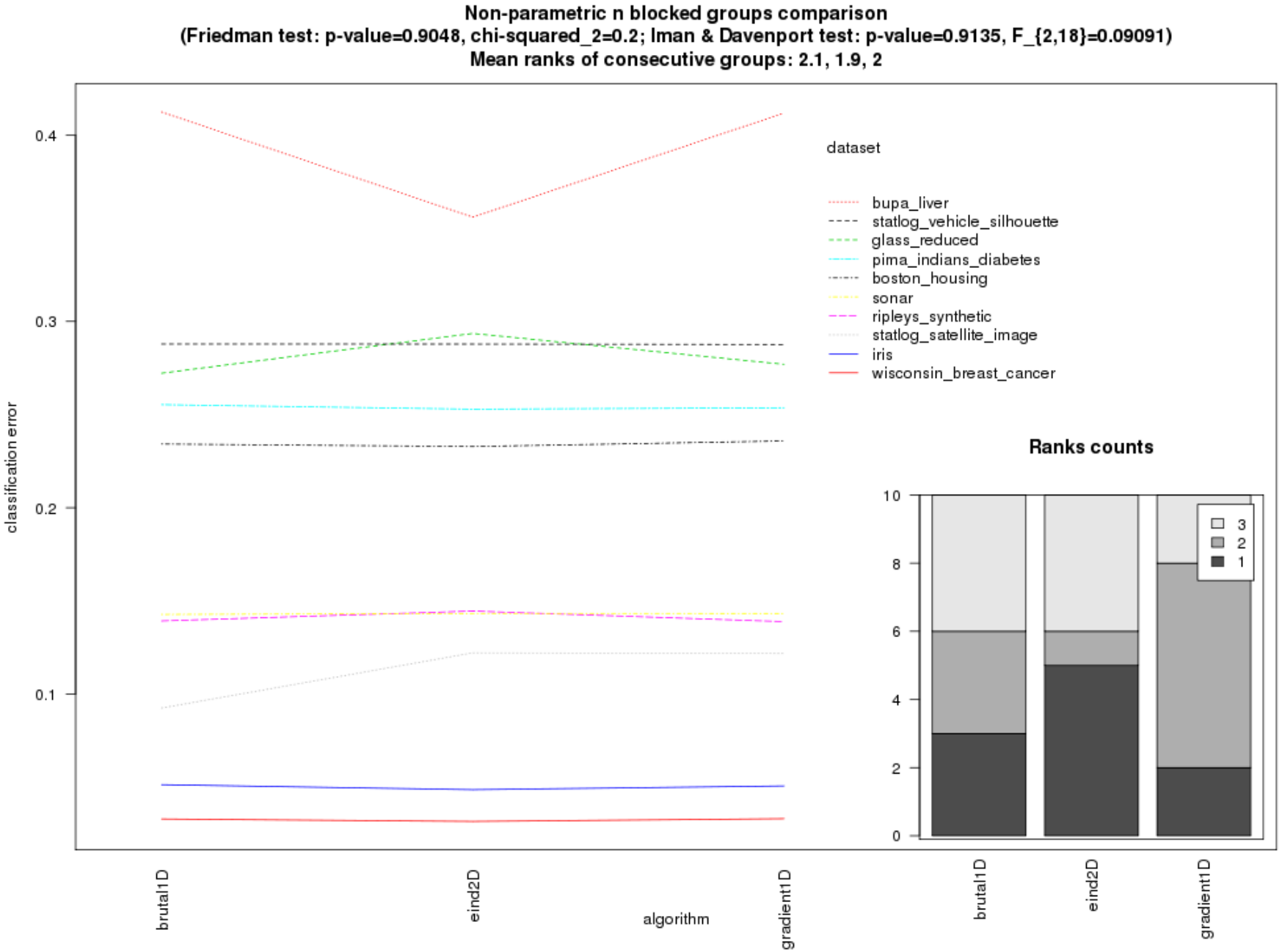
gdzie:

- p_i – przewidywane prawdopodieństwo wystąpienia i-tej klasy, gdzie i-ta klasa jest prawdziwą klasą dla danego przykładu

Funkcja błędu: Information loss

- Pytanie: Czy przynajmniej jeden z algorytmów uzyskuje wyniki lepsze od innych?
- Eksperyment:
  - Eksperyment: dla każdego algorytmu i zbioru danych: 10x 10-fold cross-validation i obliczamy średnią
  - Test: test rangowy Iman&Davenport (modyfikacja testu Friedmana) – zgodnie z metodologią z [Demsar06]
- Odpowiedź: nie ma statystycznie istotnych różnic między badanymi algorytmami

Funkcja błędu: Information loss



Funkcja błędu: MSE vs Info loss

- Pytanie: czy algorytm z funkcją MSE jest lepszy/gorszy niż z information loss na jednym ze zbiorów danych?
- Eksperyment:
  - Eksperyment: dla każdego algorytmu i zbioru danych: 10x 10-fold cross-validation
  - Test: corrected resampled paired t-test
- Oznaczenia takie jak poprzednio, oraz:
  - *.info.loss – wersja algorytmu z funkcją information loss
  - *.info.loss.laplacelike – wersja algorytmu z funkcją information loss ze zmodyfikowanym wzorem Bayesa

Funkcja błędu: MSE vs Info loss

- Oznaczenia
  - **+(-) zb.danych**: algorytm podany w wierszu jest silnie lepszy (gorszy) od algorytmu podanego w kolumnie na danym zbiorze danych, tj. $p<0.005$
  - +(-) zb.danych: algorytm podany w wierszu jest słabo lepszy (gorszy) od algorytmu podanego w kolumnie na danym zbiorze danych, tj. $0.005<p<0.05$
  - +(-) zb.danych: algorytm podany w wierszu jest prawie słabo lepszy (gorszy) od algorytmu podanego w kolumnie na danym zbiorze danych, tj. $0.05<p<0.1$

Funkcja błędu: MSE vs Info loss

- Porównanie różnych wersji eind2D

	eind2d.info.loss	eind2d.info.loss.laplacelike	eind2d-psuedoinfoloss
eind2D	+satellite (+glass_reduced)	**+satellite** (+glass_reduced)	+pima_indians
eind2D.info.loss		**+satellite**	+pima_indians -glass_reduced
eind.2D.info.loss .laplacelike			+pima_indians **-satellite** (-glass_reduced)

- Wniosek: eind2D jest najlepszy

Funkcja błędu: MSE vs Info loss

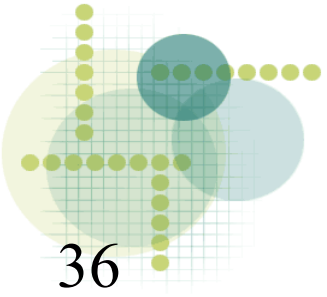
- Porównanie różnych wersji eind2D z wersjami z 1 estymatorem

	brutal1D	gradient1D	brutal1D.info.loss.laplacelike	gradient1d.info.loss.laplacelike
eind2D	(+bupa_liver)	(+bupa_liver)	+bupa_liver	**+satellite** +bupa_liver
eind2D.info.loss	+bupa_liver **-satellite** (-glass_reduced)	+bupa_liver -satellite (-glass_reduced)	+bupa_liver -satellite	**+satellite** +bupa_liver
eind.2D.info.loss.laplacelike	+bupa_liver **-satellite**	+bupa_liver **-satellite**	+bupa_liver **-satellite**	+bupa_liver

- Wniosek: eind2D jest najlepszy

Funkcja błędu: MSE vs Info loss

- Ogólne wnioski:
  - Wersja algorytmu z 2 niezależnymi estymatorami jest najlepsza (nie gorsza niż dowolna inna wersja)
  - Nie ma sensu stosować modyfikacji z funkcją błędu info loss

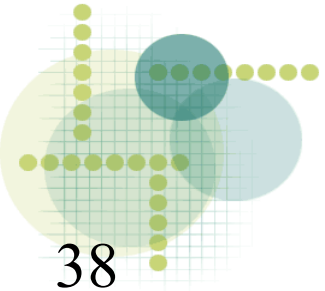


Pomysły

- Małe zmiany
  - Przetestować na 2 pominiętych zbiorach danych (image_segmentation, waveform)
  - Inne startowe punkty algorytmu optymalizacyjnego (dla wersji zlogartymowanej: (-1, -1), (0, -1))
  - Ustawić dolne ograniczenie na szerokość jądra na 0 (i przybliżać za pomocą 1-NN)
  - W nowym modelu powtórzyć eksperymenty z transformacją dla każdej klasy oddzielnie
  - Dobierać szerokość jądra dla każdej klasy oddzielnie

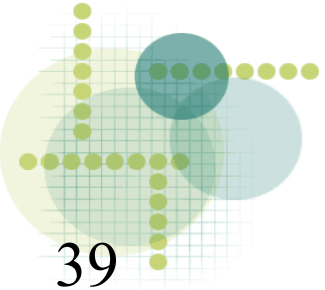
Pomysły

- Średnie zmiany
  - Używać innego jądra (p-Gaussian)
  - Startować alg. optymalizacyjny z punktów, które są sugerowane przez popularne metody szacowania optimum dla estymacji gęstości (np. Sheather-Jones)



Pomysły

- Duże zmiany:
  - Nowa funkcja błędu: położenie nacisku na punkty znajdujące się na granicach klas?
  - Dopasowywać rozmiar jądra do miejsca w przestrzeni?
  - Do estymacji gęstości używać Gaussian Mixture Model zamiast estymacji jądrowej



Literatura

[Demsar06] Demsar J. „Statistical Comparisons of Classifiers over Multiple Data Sets”, Journal of Machine Learning Research, 2006

[Lim00] Tjen-Sien Lim, Wei-Yin Loh, „A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms”, Machine Learning, 2000

[Ghosh06] Anil K. Ghosh, Probal Chaudhuri, and Debasis Sengupta, „Classification Using Kernel Density Estimates: Multi-scale Analysis and Visualization”, Technometrics, 2006

[Nadeau03] Nadeau, Bengio, „Inference for the Generalization Error", Machine Learning, 52, 2003

[Zhu97] C. Zhu, R. H. Byrd and J. Nocedal. L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization (1997), ACM Transactions on Mathematical Software, Vol 23, Num. 4, pp. 550 - 560.

Dziękuję za uwagę!