

ANALIZA KLASYFIKACYJNA WSTRZĄSÓW SEJSMICZNYCH

Projekt zaliczeniowy z przedmiotu Data Mining, 2013/14
Politechnika Warszawska
Wydział Matematyki i Nauk Informacyjnych

AUTORZY

MARCIN KOSIŃSKI, MARTA SOMMER

kosinskim@student.mini.pw.edu.pl, sommerm@student.mini.pw.edu.pl
Statystyka Matematyczna i Analiza Danych



SPIS TREŚCI

Od autorów	4
Motywacja	4
Opis zmiennych w zbiorze	5
Szczegóły techniczne pracy	6
1 Wstępna analiza danych	7
1.1 Podstawowe charakterystyki zmiennych	7
1.1.1 Zmienne ilościowe	7
1.1.2 Korelacja między zmiennymi ciągłymi	7
1.1.3 Zmienna objaśniana	7
1.2 Usunięcie braków danych, wstępna selekcja	7
2 Analiza dyskryminacyjna	9
2.1 Liniowa - LDA	9
2.1.1 Dopasowanie modelu i predykcja	9
2.1.2 Diagnostyka modelu	9
2.1.3 Krosvalidacja	9
2.1.4 Krzywa ROC i współczynnik AUC	10
2.2 Kwadratowa - QDA	10
2.2.1 Dopasowanie modelu i predykcja	10
2.2.2 Diagnostyka modelu	10
2.2.3 Krzywa ROC i współczynnik AUC	11
3 Regresja logistyczna i regularyzacja	12
3.1 Model pełny	12
3.2 Modele oparte o kryteria informacyjne	12
3.2.1 Kryterium AIC	12
3.2.2 Kryterium BIC	12
3.3 Porównanie z modelem pełnym	12
3.4 Podsumowanie	13
3.4.1 Predykcja dla modelu pełnego	13
3.4.2 Predykcja dla modelu opartego na AIC	13
3.4.3 Predykcja dla modelu opartego na BIC	14
3.5 Uwaga! na niską jakość dopasowania	14
3.6 Regularyzowana wersja regresji logistycznej	14
3.7 Krzywa ROC i współczynnik AUC dla modelu wybranego przez BIC	15
4 Naiwny Klasyfikator Bayesa	17
4.1 Dopasowanie modelu i predykcja	17
4.2 Diagnostyka modelu	17
4.3 Krzywa ROC i współczynnik AUC	17
5 Metoda k Najbliższych Sąsiadów	19
5.1 Dopasowanie modelu i predykcja	19
5.2 Diagnostyka modelu	19

6	Metoda SVM	20
6.1	Dopasowanie modelu i predykcja	20
6.2	Diagnostyka modelu	20
6.3	Krzywa ROC i współczynnik AUC	20
6.4	Wybór optymalnych parametrów modelu	21
6.5	Modele SVM z różnymi funkcjami jądrowymi	22
6.5.1	Jądro liniowe	22
6.5.2	Jądro wielomianowe	23
6.5.3	Jądro sigmoidalne	24
7	Drzewa decyzyjne	26
7.1	Graficzna prezentacja drzew decyzyjnych	26
7.2	Dopasowanie modelu i predykcja	27
7.2.1	Wybór drzewa optymalnego	27
7.2.2	Predykcja dla optymalnego drzewa	28
7.3	Diagnostyka modelu	28
7.4	Krzywa ROC i współczynnik AUC	29
8	Metody łączenia drzew decyzyjnych	30
8.1	Bagging	30
8.2	Boosting	31
8.3	Lasy losowe	32
	Zestawienie wyników dla klasyfikatorów użytych do analizy	35
9	Krzywe ROC	35
10	Tabela podsumowująca	36

OD AUTORÓW

Motywacja

Górnictwo od zawsze było powiązane z niebezpieczeństwami zwanymi zagrożeniami górniczymi. Szczególnym przypadkiem takich zagrożeń są wstrząsy sejsmiczne, często nawiedzające kopalnie głębinowe. Wstrząsy sejsmiczne są najtrudniejszym do wykrycia zagrożeniem naturalnym, co czyni je pod tym względem porównywalnymi do trzęsień ziemi. Coraz większy poziom zaawansowania systemów monitorowania wstrząsów i akustyki wstrząsów pozwala nam bardziej dogłębnie zrozumieć naturę ruchów tektonicznych i określić metody przewidywania wstrząsów.

Dokładność takich metod pozostawia jednak jak dotąd wiele do życzenia. Niedostateczna precyzja metod statystycznych spowodowana jest złożonością procesów tektonicznych, jak również dużą rozbieżnością między liczbą wstrząsów o niskiej magnitudzie a liczbą wstrząsów o wysokiej magnitudzie (np. wstrząsów o energii przekraczającej $10^4 J$). W rezultacie istniejące metody statystyczne bywają niewystarczające do dokładnego przewidywania wstrząsów.

Z uwagi na realne zagrożenie życia spowodowane wstrząsami sejsmicznymi istotne jest poszukiwanie nowych sposobów lepszego wykrywania niebezpieczeństw - między innymi za pomocą metod uczenia maszynowego. Uczenie maszynowe jest w sejsmologii dziedziną wciąż rozwijającą się, gdyż obecnie stosowane metody nie są dość czułe i nie pozwalają na przewidywanie zagrożeń sejsmicznych z wystarczającą dokładnością.

Precyzyjna prognoza wzrostu aktywności tektonicznej jest niezwykle istotna w praktycznym zapobieganiu zagrożeniom. Dlatego właśnie postanowiliśmy przyjrzeć się zestawowi danych dotyczących prawdziwych problemów sejsmologicznych, z którymi zmaga się współczesna statystyka stosowana.

Opis zmiennych w zbiorze

Zbiór wykorzystany w analizie cechuje się niezbalansowanym rozkładem przypadków występowania i braku występowania cechy, dla której ma być zadana predykcja. Na blisko ponad 2000 przypadków tylko 170 reprezentuje klasę 1. Będzie to główny problem, przewijający się przez cały proces analizy danych.

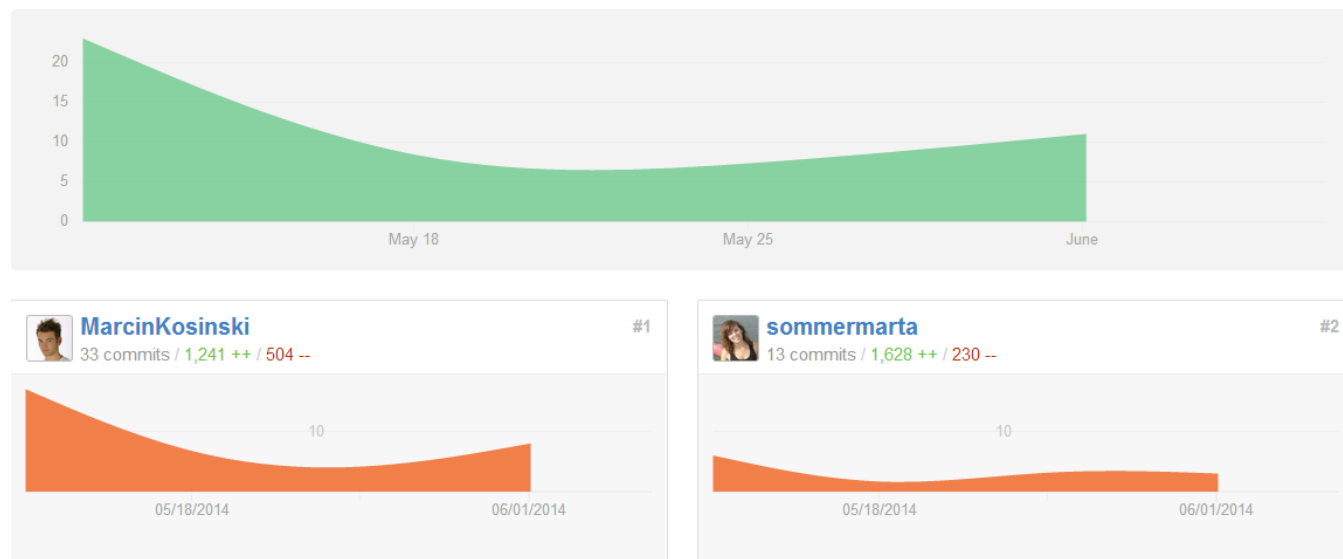
Tabela 1: Opis zmiennych ze zbioru *Seismic-bumps*.

Zmienna	Opis
seismic	wynik oceny zagrożenia sejsmicznego w czasie danej zmiany oszacowanego metodą pomiarów sejsmicznych (a - brak zagrożenia, b - niski poziom zagrożenia, c - wysoki poziom zagrożenia, d - niebezpieczeństwo wstrząsu)
seismoacoustic	wynik oceny zagrożenia sejsmicznego w czasie danej zmiany oszacowanego metodą pomiarów sejsmoakustycznych
shift	informacja na temat typu zmiany, (W - ekipa wydobywająca węgiel, N - zmiana przygotowawcza)
genergy	poziom energii sejsmicznej zarejestrowany przez najbardziej aktywny (GMax) spośród geofonów monitorujących tunel
gpuls	liczba drgań zarejestrowana ¹ przez Gmax
gdenergy	odchylenie poziomu energii zarejestrowanej ¹ przez Gmax od średniej zarejestrowanej podczas ostatnich 8 zmian
gdpuls	odchylenie liczby drgań zarejestrowanej ¹ przez Gmax od średniej zarejestrowanej podczas ostatnich 8 zmian
ghazard	wynik oceny zmiany zagrożenia sejsmicznego na zmianie oszacowany metodą <i>seismoacoustic</i> tylko przez geofonu Gmax
nbumps	liczba tąpnięć sejsmicznych ¹
nbumps2	liczba tąpnięć sejsmicznych ¹ w zakresie $[10^2, 10^3)$
nbumps3	liczba tąpnięć sejsmicznych ¹ w zakresie $[10^3, 10^4)$
nbumps4	liczba tąpnięć sejsmicznych ¹ w zakresie $[10^4, 10^5)$
nbumps5	liczba tąpnięć sejsmicznych ¹ w zakresie $[10^5, 10^6)$
nbumps6	liczba tąpnięć sejsmicznych ¹ w zakresie $[10^6, 10^7)$
nbumps7	liczba tąpnięć sejsmicznych ¹ w zakresie $[10^7, 10^8)$
nbumps89	liczba tąpnięć sejsmicznych ¹ w zakresie $[10^8, 10^{10})$
energy	całkowita energia tąpnięć sejsmicznych ¹
maxenergy	maksymalna energia tąpnięć sejsmicznych ¹
class	zmienna decyzyjna – czy na kolejnej zmianie wystąpi silne tąpnięcie

¹Zarejestrowana podczas poprzedniej zmiany

Szczegóły techniczne pracy

Analiza została przeprowadzona w pakiecie statystycznym R, wersja 3.1.0. Tekst raportu został złożony przy pomocy pakietu knitr, który umożliwia załączanie wyników i rezultatów z funkcji R w programie do składu publikacji LaTeX. Projekt został przeprowadzony przy wsparciu systemu kontroli wersji Git, który czuwał nad integracją plików i optymalizacją pracy zespołowej. Korzyść z zastosowania takiego rozwiązania technologicznego była taka, że w dwie osoby mogliśmy wspólnie pracować na wielu plikach .Rnw jednocześnie bez konfliktów i z możliwie szybkim przekazem plików. Do połączenia pakietu R z systemem kontroli wersji Git wykorzystano pakiet devtools, wspomagający integrację pakietu R z GitHubem (<https://github.com>). Ostatecznie dzięki systemowi kontroli wersji można śledzić postępy w pracach i zaangażowanie poszczególnych osób w sukces projektu. Poniżej screen z aktywności przy obecnym projekcie:



Repozytorium, w którym dostępne są wszystkie pliki jest dostępne pod adresem: <https://github.com/MarcinKosinski/DM1>.

```
Error in library("adabag"): there is no package called 'adabag'
Error in library("RcppEigen"): there is no package called 'RcppEigen'
Error in library("caret"): there is no package called 'caret'
Error in library("randomForest"): there is no package called 'randomForest'
```

ROZDZIAŁ 1

WSTĘPNA ANALIZA DANYCH

1.1 Podstawowe charakterystyki zmiennych

1.1.1 Zmienne ilościowe

	genergy	gpuls	gdenergy	gdpuls	nbumps	energy	maxenergy
Odchylenie St	229200.51	562.65	80.32	63.17	1.36	20450.83	19357.45
Wariancja	52532873277.49	316577.88	6451.15	3990.01	1.86	418236579.49	374711059.50
Mediana	25485.00	379.00	-6.00	-6.00	0.00	0.00	0.00
Średnia	90242.52	538.58	12.38	4.51	0.86	4975.27	4278.85

Tabela 1.1: Podstawowe statystyki agregujące dla zmiennych ciągłych.

1.1.2 Korelacja między zmiennymi ciągłymi

	genergy	gpuls	gdenergy	gdpuls	nbumps	nbumps2	nbumps3	nbumps4	nbumps5	energy	maxenergy
genergy	1.00	0.76	0.39	0.39	0.49	0.34	0.39	0.23	0.04	0.48	0.47
gpuls	0.76	1.00	0.46	0.58	0.31	0.22	0.21	0.22	0.04	0.30	0.30
gdenergy	0.39	0.46	1.00	0.80	0.07	0.08	0.01	0.07	0.06	0.07	0.06
gdpuls	0.39	0.58	0.80	1.00	0.10	0.09	0.04	0.09	0.07	0.10	0.09
nbumps	0.49	0.31	0.07	0.10	1.00	0.76	0.76	0.34	0.09	0.96	0.94
nbumps2	0.34	0.22	0.08	0.09	0.76	1.00	0.31	0.15	0.00	0.59	0.56
nbumps3	0.39	0.21	0.01	0.04	0.76	0.31	1.00	0.16	0.06	0.79	0.78
nbumps4	0.23	0.22	0.07	0.09	0.34	0.15	0.16	1.00	-0.02	0.45	0.45
nbumps5	0.04	0.04	0.06	0.07	0.09	0.00	0.06	-0.02	1.00	0.13	0.13
energy	0.48	0.30	0.07	0.10	0.96	0.59	0.79	0.45	0.13	1.00	1.00
maxenergy	0.47	0.30	0.06	0.09	0.94	0.56	0.78	0.45	0.13	1.00	1.00

1.1.3 Zmienna objaśniana

Poniżej widoczny jest podział klas w naszym zbiorze danych:

```
summary(se[,16])

 0      1
2414 170
```

1.2 Usunięcie braków danych, wstępna selekcja

W naszym zbiorze nie występują braki danych.

Na pierwszy rzut oka widać, że niektóre zmienne są zupełnie nieistotne lub bardzo ze sobą skorelowane, dlatego chcielibyśmy od razu usunąć je z analizy. Przyjrzyjmy się naszym danym:

	seismic	seismoacoustic	shift	genergy	gpuls	gdenergy	gdpuls	ghazard	nbumps	nbumps2	nbumps3	nbumps4
1	a	a	N	15180	48	-72	-72	a	0	0	0	0
2	a	a	N	14720	33	-70	-79	a	1	0	1	0
3	a	a	N	8050	30	-81	-78	a	0	0	0	0
	nbumps5	nbumps6	nbumps7	nbumps89	energy	maxenergy	class					
1	0	0	0	0	0	0	0					
2	0	0	0	0	2000	2000	0					
3	0	0	0	0	0	0	0					

Zacznijmy od zmiennych *nbumps6*, *nbumps7* oraz *nbumps89*. Są to zmienne ilościowe, wszystkie jednak przyjmują wartość zero. Niczego więc nie wnoszą one do naszej analizy. Zastąpmy je więc przez zmienną *nbumps5* (a raczej rozszerzymy definicję zmiennej *nbumps5*), która będzie od tej pory definiować liczbę wstrząsów sejsmicznych w zakresie $[10^5, 10^{10}]$. Z analizy wyrzucimy również zmienną *maxenergy*, która wydaje się silnie zależna od zmiennej *energy*.

Kolejną zmienną, którą usuniemy z analizy będzie zmienna *nbumps*, która jest liniowo zależna od zmiennych *nbumps2*, *nbumps3*, *nbumps4* i *nbumps5*.

Zajmijmy się jeszcze zmiennymi nominalnymi. Niestety nie każda metoda, którą będziemy stosować do analizy (np. QDA) dobrze sobie z nimi radzi. Będziemy więc pracować na dwóch zestawach danych. Tam, gdzie się da, będziemy korzystać z danych oryginalnych. W reszcie metod zamienimy sobie dane nominalny na dane ilościowe w następujący sposób:

Zmienną *seismic* przyjmującą wartości ze zbioru $\{a, b, c, d\}$ zamienimy na zmienną *seismic2* przyjmującą wartości $\{0, 1, 2, 3\}$. W analogiczny sposób zamienimy zmienną *seismoacoustic* na zmienną *seismoacoustic2* oraz zmienną *ghazard* na zmienną *ghazard2*. Możemy tak zrobić, gdyż ich wartości reprezentują pewien porządek (a - małe, d - duże). Tak nie jest niestety w przypadku zmiennej *shift*. Dlatego dla niej stworzymy zmienną indykatorową *shift2* równą 1, gdy na zmianie byli górnicy, a 0, gdy na zmianie była ekipa przygotowawcza.

W ten sposób nasz zbiór danych w formie nominalnej przedstawia się następująco:

	seismic	seismoacoustic	shift	genergy	gpuls	gdenergy	gdpuls	ghazard	nbumps2	nbumps3	nbumps4	
1	a		a	N	15180	48	-72	-72	a	0	0	0
2	a		a	N	14720	33	-70	-79	a	0	1	0
3	a		a	N	8050	30	-81	-78	a	0	0	0
	nbumps5	energy	class									
1	0	0	0									
2	0	2000	0									
3	0	0	0									

A zbiór danych w formie liczbowej następująco:

	seismic2	seismoacoustic2	shift2	genergy	gpuls	gdenergy	gdpuls	ghazard2	nbumps2	nbumps3	nbumps4
1	0	0	0	15180	48	-72	-72	0	0	0	0
2	0	0	0	14720	33	-70	-79	0	0	1	0
3	0	0	0	8050	30	-81	-78	0	0	0	0
	nbumps5	energy	class								
1	0	0	0								
2	0	2000	0								
3	0	0	0								

ROZDZIAŁ 2

ANALIZA DYSKRYMINACYJNA

2.1 Liniowa - LDA

2.1.1 Dopasowanie modelu i predykcja

Dopasujemy model LDA do naszych danych na zbiorze treningowym:

```
mod_lda <- lda(class~., data=Train)
```

Zróbmy predykcję na zbiorze testowym:

```
pred_klas <- predict(mod_lda, newdata=Test)$class  
pred_praw <- predict(mod_lda, newdata=Test)$posterior[,2]
```

2.1.2 Diagnostyka modelu

Tabela rekasyfikacji na zbiorze testowym wygląda następująco:

	klasy_prawdziwe	
predykacja_klasy	0	1
0	793	45
1	18	6

A procent poprawnego dopasowania, czułość i precyzja następująco:

```
procent(tab)  
[1] 92.69142  
  
czulosc(tab)  
[1] 0.25  
  
precyzja(tab)  
[1] 0.1176471
```

Mimo że procent poprawnej klasyfikacji jest bardzo wysoki (ponad 92%), to nie możemy tu mówić o dobrym klasyfikatorze. Wyraźnie widać, że nasza LDA klasyfikuje prawie wszystkie obserwacje jako 0, a przez fakt, że próba nie jest symetryczna (zera stanowią znaczną większość próby) osiągamy wysoki procent poprawnej klasyfikacji. O tym, że rzeczywiście model nie jest najlepszy powiedzą nam również czułość (25%) i precyzja (12%).

2.1.3 Krosvalidacja

Sprawdźmy jeszcze jakość dopasowanego modelu przeprowadzając krosvalidację dziesiętnokrotną. Oto otrzymane rezultaty:

```
mean(pro) # procent poprawnej klasyfikacji  
[1] 92.02438
```

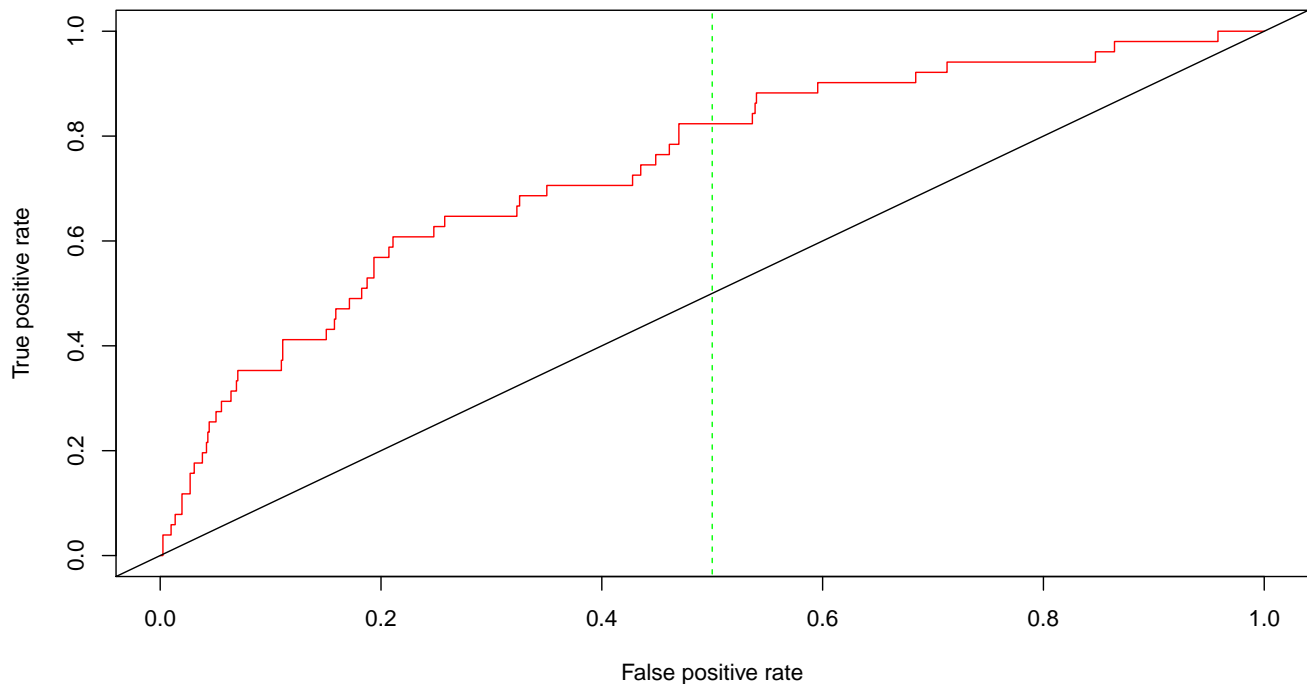
```
mean(czu) # czułość
[1] 0.3199206

mean(pre) # precyzja
[1] 0.1488991
```

Czułość i precyzja nieco się poprawiły.

2.1.4 Krzywa ROC i współczynnik AUC

I na koniec wyznaczmy krzywą ROC:



Współczynnik AUC wynosi:

```
[1] 0.7398032
```

2.2 Kwadratowa - QDA

2.2.1 Dopasowanie modelu i predykcja

Dopasujemy model QDA do naszych danych na zbiorze treningowym:

```
mod_qda <- qda(class~., data=Train2)
```

Zróbmy predykcję na zbiorze testowym:

```
pred_klas <- predict(mod_qda, newdata=Test2)$class
pred_praw <- predict(mod_qda, newdata=Test2)$posterior[,2]
```

2.2.2 Diagnostyka modelu

Tabela rekasyfikacji na zbiorze testowym wygląda następująco:

	klasy_prawdziwe	
predykcja_klasy	0	1
0	734	34
1	77	17

A procent poprawnego dopasowania, czułość i precyzja następująco:

```
procent(tab)
[1] 87.12297

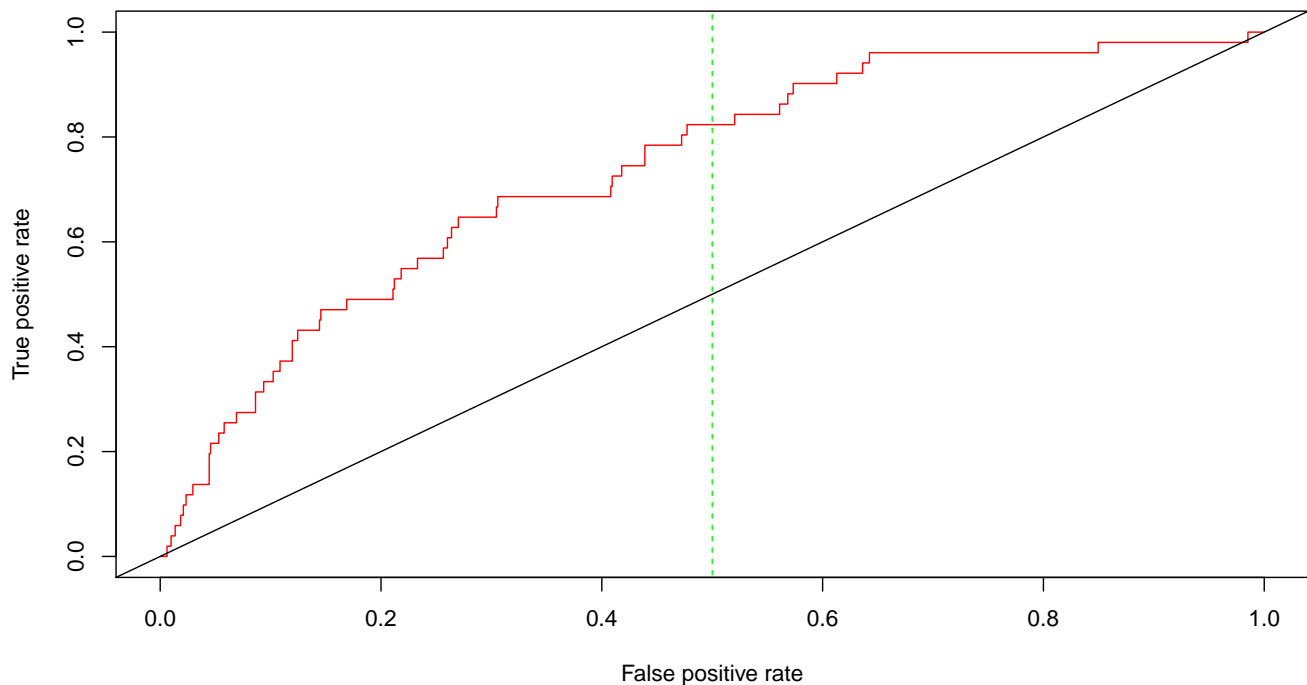
czulosc(tab)
[1] 0.1808511

precyzja(tab)
[1] 0.3333333
```

No i znów (podobnie jak dla LDA) model nie jest najlepiej dopasowany. Mimo to wyraźnie widać znaczną poprawę (szczególnie precyzji).

2.2.3 Krzywa ROC i współczynnik AUC

I na koniec wyznaczmy krzywą ROC:



Współczynnik AUC wynosi:

```
[1] 0.7386669
```

ROZDZIAŁ 3

REGRESJA LOGISTYCZNA I REGULARYZACJA

3.1 Model pełny

Po wybraniu zbioru testowego i treningowego w proporcjach 1:2 przeprowadzono regresję logistyczną:

```
Se.logit <- glm(class~.,data=Train,family="binomial")
```

Przy tak ustawionym ziarnie losowania `set.seed(456)` istotnymi zmiennymi w modelu są zmienne `shift` w grupie `W` oraz zmienne `gpuls`, `nbumps2` i `nbump3`. Przy niektórych innych ziarnach również istotną zmienną w modelu był `genergy` - co widać po małej wartości krytycznej-p.

3.2 Modele oparte o kryteria informacyjne

Korzystając z funkcji `step()` wyliczono modele oparte o kryteria informacyjne AIC oraz BIC.

3.2.1 Kryterium AIC

Stosując metodę wsteczną otrzymano model:

```
class ~ seismic + shift + genergy + gpuls + gdenenergy + nbumps2 +  
nbumps3
```

Kryterium oparte o AIC wybrało zmienne, które były istotne w modelu pełnym oraz dodatkowo zmienne, których p-wartości były bliskie 0.05. Jako że kryterium AIC jest konserwatywne i nie odrzuca zmiennych, nie dziwne, że w modelu zostały uwzględnione również zmienne o p-wartości zbliżonej do 0.05.

3.2.2 Kryterium BIC

Stosując metodę wsteczną otrzymano model:

```
class ~ shift + gpuls + nbumps2 + nbumps3
```

Kryterium oparte o BIC wybrało tylko te zmienne, które były istotne w modelu pełnym, czyli model `shift + gpuls + nbumps2 + nbumps3`. Ponieważ prawdopodobieństwo wybrania przez kryterium BIC poprawnego modelu dąży do 1, a zbiór uczący miał ponad 1800 obserwacji, zakładam, że model wybrany przez BIC jest odpowiedni.

3.3 Porównanie z modelem pełnym

W celu przetestowania adekwatności wyboru modelu pełnego przeprowadzono test Chi-kwadrat, by porównać go z modelami mniejszymi:

Test: kryterium AIC vs model pełny

```
anova(Se.logit.aic,Se.logit,test="Chi")
```

Analysis of Deviance Table

```
Model 1: class ~ seismic + shift + genergy + gpuls + gdenenergy + nbumps2 +
nbumps3
Model 2: class ~ seismic + seismoacoustic + shift + genergy + gpuls +
gdenenergy + gdpuls + ghazard + nbumps2 + nbumps3 + nbumps4 +
nbumps5 + energy
Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      1714      731.07
2      1706      726.00  8    5.0772    0.7493
```

Ponieważ test nie odrzucił hipotezy zerowej (p-wartość jest większa o zakładanego poziomu istotności 0.05) to nie ma podstaw by nie zakładać, że model pełny może być uproszczony do modelu mniejszego wybranego przez kryterium informacyjne AIC.

Test: kryterium BIC vs model pełny

```
anova(Se.logit.bic,Se.logit,test="Chi")
```

Analysis of Deviance Table

```
Model 1: class ~ shift + gpuls + nbumps2 + nbumps3
Model 2: class ~ seismic + seismoacoustic + shift + genergy + gpuls +
gdenenergy + gdpuls + ghazard + nbumps2 + nbumps3 + nbumps4 +
nbumps5 + energy
Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      1717      740.54
2      1706      726.00 11   14.542    0.2044
```

Ponieważ test nie odrzucił hipotezy zerowej (p-wartość jest większa o zakładanego poziomu istotności 0.05) to nie ma podstaw by nie zakładać, że model pełny może być uproszczony do modelu mniejszego wybranego przez kryterium informacyjne BIC. **Test: kryterium AIC vs kryterium BIC**

```
anova(Se.logit.aic,Se.logit.bic,test="Chi")
```

Analysis of Deviance Table

```
Model 1: class ~ seismic + shift + genergy + gpuls + gdenenergy + nbumps2 +
nbumps3
Model 2: class ~ shift + gpuls + nbumps2 + nbumps3
Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      1714      731.07
2      1717      740.54 -3   -9.4646    0.02371 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Wartość krytyczna testu jest mniejsza od poziomu istotności zatem należy odrzucić hipotezę zerową i przyjąć model wybrany za pomocą kryterium BIC.

3.4 Podsumowanie

Regresja liniowa oparta o kryterium informacyjne BIC i regułę krokową wsteczną wybrało do modelu zmienne jako adekwatne i istotne: `shift + gpuls + nbumps2 + nbumps3`.

3.4.1 Predykcja dla modelu pełnego

Wywołanie i stworzenie klasyfikatora dla modelu pełnego wygląda następująco:

```
P <- predict(Se.logit,newdata=Test,type="response")
Pred <- ifelse(P>0.5,1,0)
```

Poprawność dopasowania wynosi:

```
Tab <- tabela(Pred,Test$class)
100*sum(diag(Tab))/sum(Tab)

[1] 93.96752
```

3.4.2 Predykcja dla modelu opartego na AIC

Wywołanie i stworzenie klasyfikatora dla modelu opartego na AIC wygląda następująco:

```
P.aic <- predict(Se.logit.aic,newdata=Test,type="response")
Pred.aic <- ifelse(P.aic>0.5,1,0)
```

Poprawność dopasowania wynosi:

```
Tab.aic <- tabela(Pred.aic,Test$class)
100*sum(diag(Tab.aic))/sum(Tab.aic)

[1] 93.96752
```

3.4.3 Predykcja dla modelu opartego na BIC

Wywołanie i stworzenie klasyfikatora dla modelu opartego na BIC wygląda następująco:

```
P.bic <- predict(Se.logit.bic,newdata=Test,type="response")
Pred.bic <- ifelse(P.bic>0.5,1,0)
```

Poprawność dopasowania wynosi:

```
Tab.bic <- tabela(Pred.bic,Test$class)
100*sum(diag(Tab.bic))/sum(Tab.bic)

[1] 93.96752
```

Wartości poprawności dopasowania w każdym z modeli wychodzą bardzo wysokie. Rzędu 94%. Nie jest to żaden błąd, gdyż np. 5 pierwszych predykcji wygląda różnie:

```
formatC(c(P[1:5],P.aic[1:5],P.bic[1:5]), digits=3, format="f")

      6      7      8      13      15      6      7      8      13      15      6      7
"0.035" "0.078" "0.021" "0.052" "0.018" "0.036" "0.080" "0.021" "0.053" "0.018" "0.038" "0.097"
      8      13      15
"0.022" "0.063" "0.021"
```

3.5 Uwaga! na niską jakość dopasowania

Modele dopasowane w poprzednim podrozdziale nie są poprawne pomimo, że błąd klasyfikacji w każdym z przypadków wyniósł po 6%. Proszę spojrzeć na tabelę klasyfikacyjną w każdym z tych modeli:

```
Tab

      klasy_prawdziwe
predycja_klasy  0  1
      0 808  49
      1   3   2

Tab.aic

      klasy_prawdziwe
predycja_klasy  0  1
      0 808  49
      1   3   2

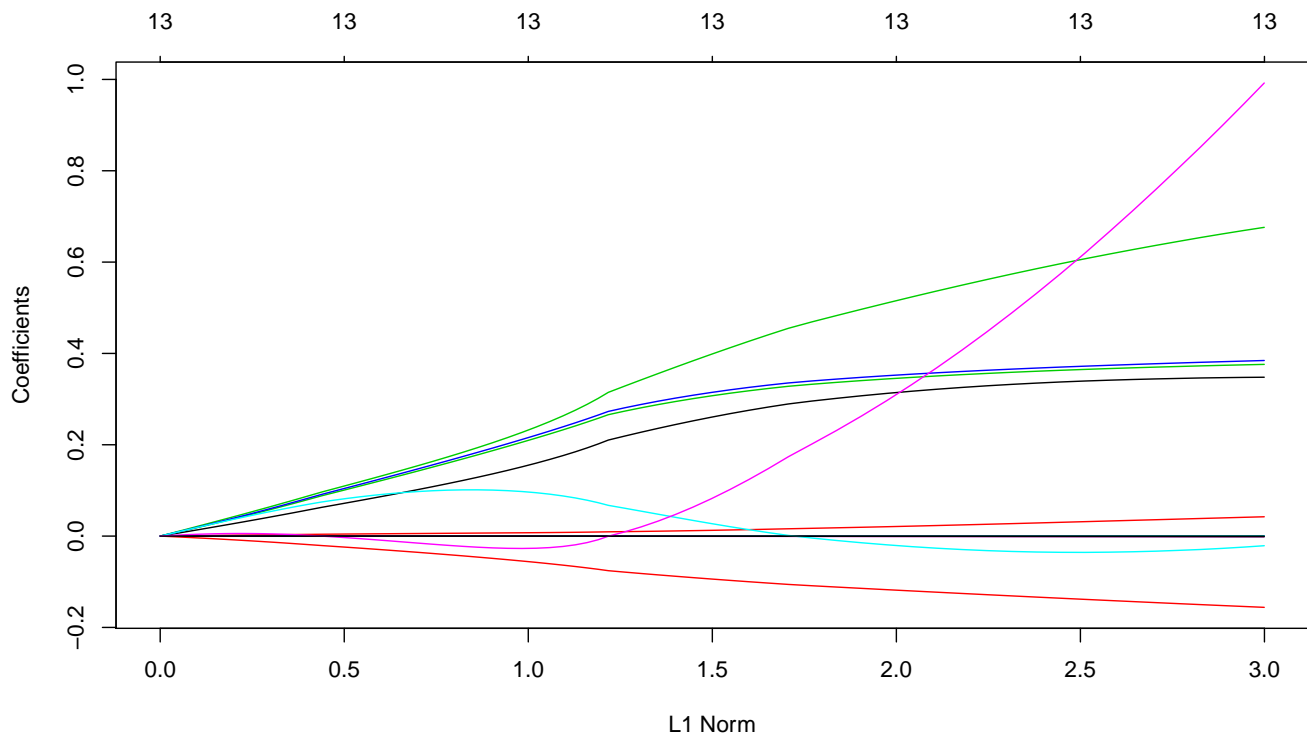
Tab.bic

      klasy_prawdziwe
predycja_klasy  0  1
      0 809  50
      1   2   1
```

W każdym z modelu około 50 przypadków wystąpień silnych wstrząsów przy następnej zmianie załogi górników nie zostało poprawnie zaklasyfikowanych. Oznacza to, że nasz klasyfikator jest błędny i klasyfikuje wszystkie przypadki do cechy 0, oznaczającej, że niebezpieczeństwa nie będzie i że silne wstrząsy nie nastąpią. Pomimo małego błędu predykcji klasyfikator oparty na modelu regresji liniowej jest nieadekwatny i błędny.

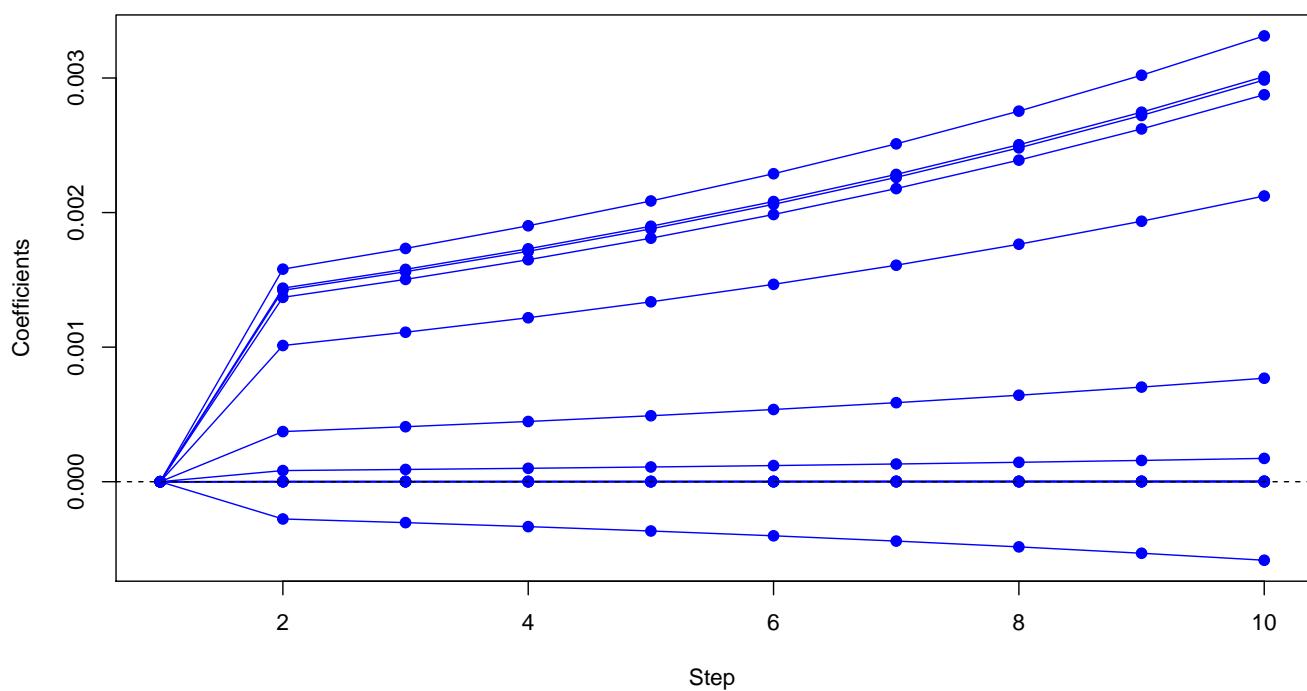
3.6 Regularyzowana wersja regresji logistycznej

```
rlass<-glmnet(as.matrix(se_r[,-14]), se_r[,14], family = "binomial", alpha = 0, lambda.min = 1e-4)
plot(rlass)
```



Powyżej widać, które zmienne dołączane są w którym kroku algorytmu.

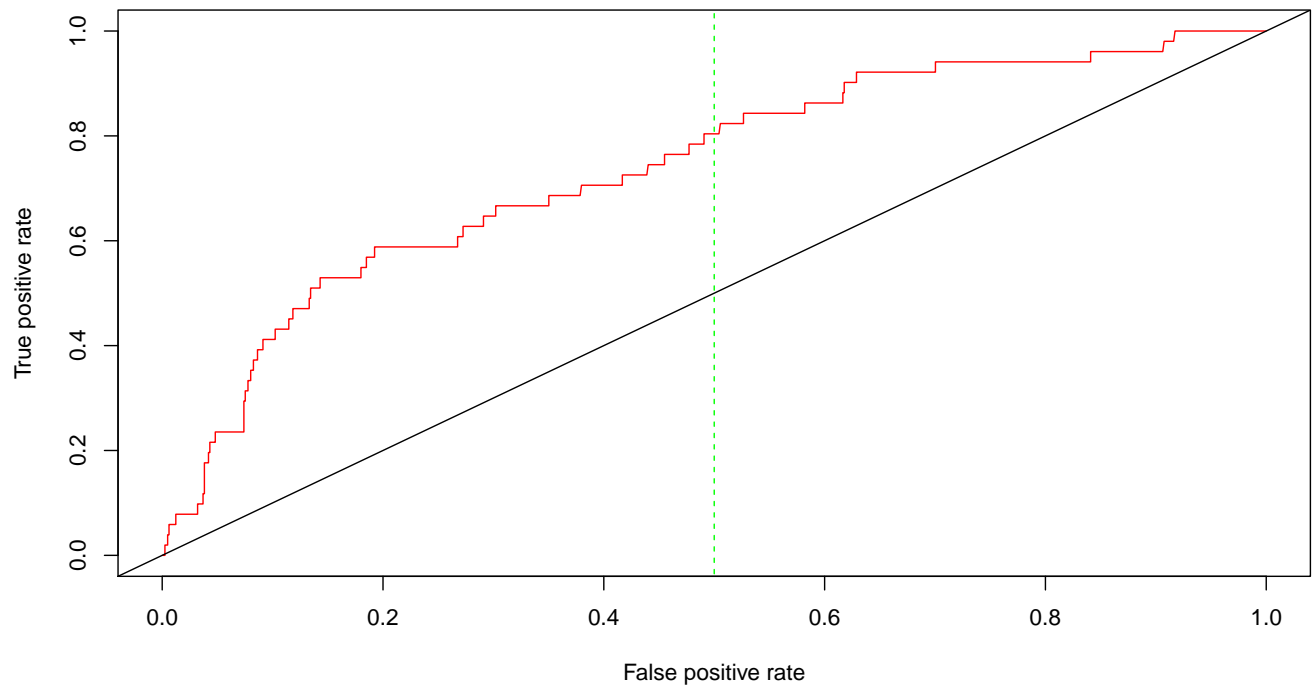
Lasso



Powyżej widać, które zmienne dołączane są w którym kroku algorytmu, jednak inna metodą narysowan i z inną skalą. Teraz widać więcej.

3.7 Krzywa ROC i współczynnik AUC dla modelu wybranego przez BIC

I na koniec wyznaczmy krzywą ROC:



Współczynnik AUC wynosi:

```
[1] 0.7383405
```

ROZDZIAŁ 4

NAIWNY KLASYFIKATOR BAYESA

4.1 Dopasowanie modelu i predykcja

Dopasowanie modelu na zbiorze treningowym:

```
bayes <- naiveBayes(class~.,data=Train, laplace=0.2)
```

Predykcja na zbiorze testowym, klasy i prawdopodobieństwo przynależności do klasy 1:

```
bayes_pred <- predict(bayes, newdata=Test)
bayes_prawd <- predict(bayes, newdata=Test, type="raw")[,2]
```

4.2 Diagnostyka modelu

Tabela rekasyfikacji na zbiorze testowym wygląda następująco:

	klasy_prawdziwe	
predykcia_klasy	0	1
0	738	31
1	73	20

A procent poprawnego dopasowania, czułość i precyzja następująco:

```
procent(tab)
[1] 87.93503

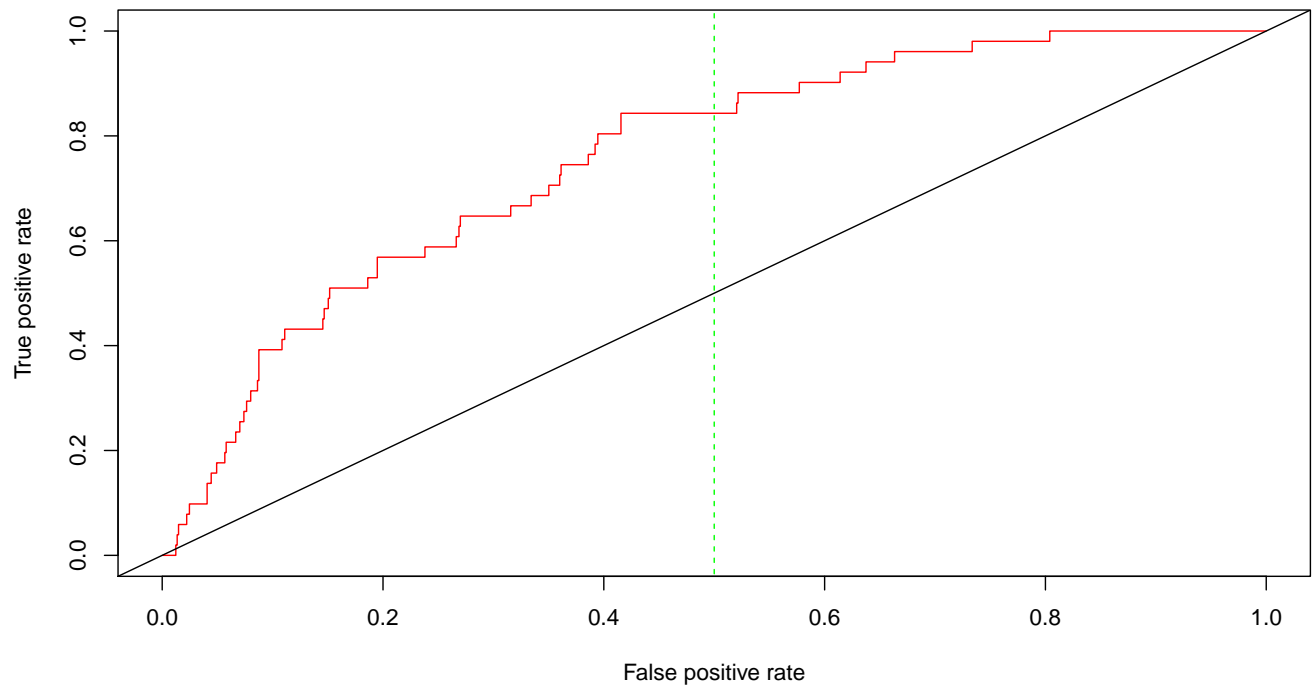
czulosc(tab)
[1] 0.2150538

precyzja(tab)
[1] 0.3921569
```

Pomimo wysokiej czułości i precyzji, naiwny klasyfikator bayesa ma bardzo mały, w porównaniu do innych klasyfikatorów użytych w tym raporcie, procent poprawnej klasyfikacji. Jednak czułość i precyzja są na bardzo wysokim poziomie, w porównaniu do innych klasyfikatorów!

4.3 Krzywa ROC i współczynnik AUC

I na koniec wyznaczmy krzywą ROC:



Współczynnik AUC wynosi:

```
[1] 0.7583714
```

Piękny wynik!

ROZDZIAŁ 5

METODA K NAJBLIŻSZYCH SĄSIADÓW

5.1 Dopasowanie modelu i predykcja

Dopasujemy metodę k Najbliższych Sąsiadów do naszych danych na zbiorze treningowym:

```
sei_knn <- knn(Train2[, -14], Test2[, -14], cl=Train$class, k=2)
```

5.2 Diagnostyka modelu

Tabela rekasyfikacji na zbiorze testowym wygląda następująco:

```
(tab <- tabela(knn(Train2[, -14], Test2[, -14], cl=Train$class, k=2), Test2$class))
```

	klasy_prawdziwe	
predykcja_klasy	0	1
0	773	46
1	38	5

A procent poprawnego dopasowania, czułość i precyzja następująco:

```
procent(tab)
[1] 90.25522

czulosc(tab)
[1] 0.1162791

precyzja(tab)
[1] 0.09803922
```

Pomimo dość dobrej poprawności procentowej dopasowania, czułość i precyzja dopasowania są na bardzo niskim poziomie.

ROZDZIAŁ 6

METODA SVM

6.1 Dopasowanie modelu i predykcja

Dopasujemy model SVM do naszych danych na zbiorze treningowym (przyjmując domyślne parametry i jądro radialne):

```
mod_svm1 <- svm(class~., data=Train, type="C", kernel="radial",  
                probability=TRUE)
```

Zróbmy predykcję na zbiorze testowym:

```
pred1 <- predict(mod_svm1, Test, probability=TRUE)  
pred_praw1 <- attr(pred1, "probabilities")[,2]  
pred_klas1 <- predict(mod_svm1, Test)
```

6.2 Diagnostyka modelu

Tabela rekasyfikacji na zbiorze testowym wygląda następująco:

	klasy_prawdziwe	
predykcia_klasy	0	1
0	811	51
1	0	0

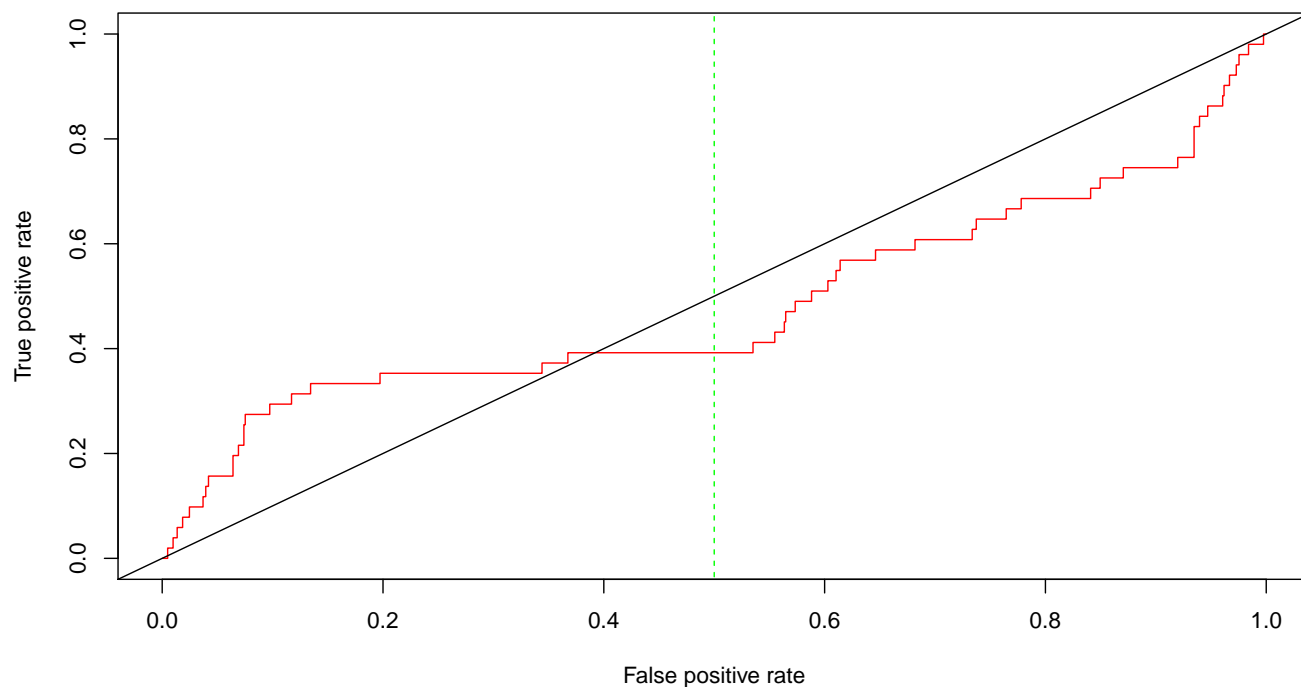
Nasz model zakwalifikował wszystkie obserwacje jako zera! Nie możemy więc tu mówić o dobrym klasyfikatorze. Mimo wszystko spójrzmy jeszcze, jak wygląda procent poprawnego dopasowania, czułość i precyzja:

```
procent(tab1)  
[1] 94.08353  
  
czulosc(tab1)  
[1] 0  
  
precyzja(tab1)  
[1] 0
```

Czułość i precyzja są równe zero, czyli rzeczywiście już gorzej być nie może.

6.3 Krzywa ROC i współczynnik AUC

I na koniec wyznaczmy krzywą ROC:



Współczynnik AUC wynosi:

```
[1] 0.4822175
```

Krzywa ROC i współczynnik AUC sugerują, że model jest gorszy, niż gdybyśmy robili klasyfikację zupełnie losowo!

6.4 Wybór optymalnych parametrów modelu

Spróbujmy więc jakoś poprawić jakość dopasowania. Korzystając z funkcji `tune()` wyznaczmy optymalne parametry modelu:

```
gamma cost
2      1  2
```

Dopasujmy teraz nowy model SVM, tym razem z optymalnymi parametrami i zrobmy predykcję:

```
mod_svm2 <- svm(class~, data=Train, type="C", kernel="radial",
                 gamma=obj$best.parameters[1], cost=obj$best.parameters[2],
                 probability=TRUE)
pred2 <- predict(mod_svm2, Test, probability=TRUE)
pred_praw2 <- attr(pred2, "probabilities")[,2]
pred_klas2 <- predict(mod_svm2, Test)
```

Tabela reklasifikacji wygląda następująco:

	klasy_prawdziwe	
predykcia_klasy	0	1
0	803	50
1	8	1

Widać, że jest już lepiej. Sprawdźmy, czy poprawiły się czułość i precyzja:

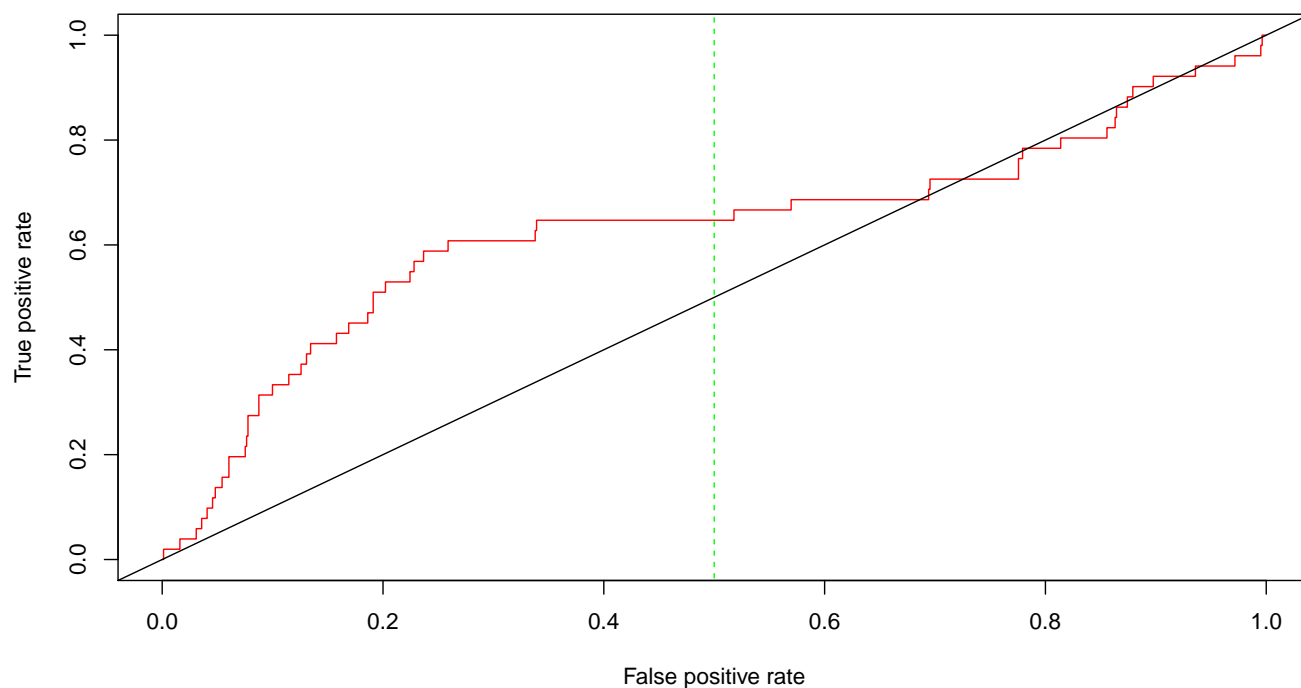
```
procent(tab2)
[1] 93.27146

czulosc(tab2)
[1] 0.1111111

precyzja(tab2)
[1] 0.01960784
```

Jest zdecydowanie lepiej, ale nadal źle. Spójrzmy na krzywą ROC i parametr AUC:

```
[1] 0.628273
```



Spróbujmy więc dopasować modele SVM używając różnych funkcji jądrowych.

6.5 Modele SVM z różnymi funkcjami jądrowymi

6.5.1 Jądro liniowe

Dopasowanie modelu i predykcja:

```
mod_svm3 <- svm(class~, data=Train, type="C", kernel="linear",
               gamma=obj$best.parameters[1], cost=obj$best.parameters[2],
               probability=TRUE)
pred3 <- predict(mod_svm3, Test, probability=TRUE)
pred_praw3 <- attr(pred3, "probabilities")[,2]
pred_klas3 <- predict(mod_svm3, Test)
```

Tabela rekasyfikacji:

	klasy_prawdziwe	
predykcia_klasy	0	1
0	811	51
1	0	0

I znów wszystko jest klasyfikowane jako zera... Popatrzmy na czułość, precyzję i procent poprawnego dopasowania:

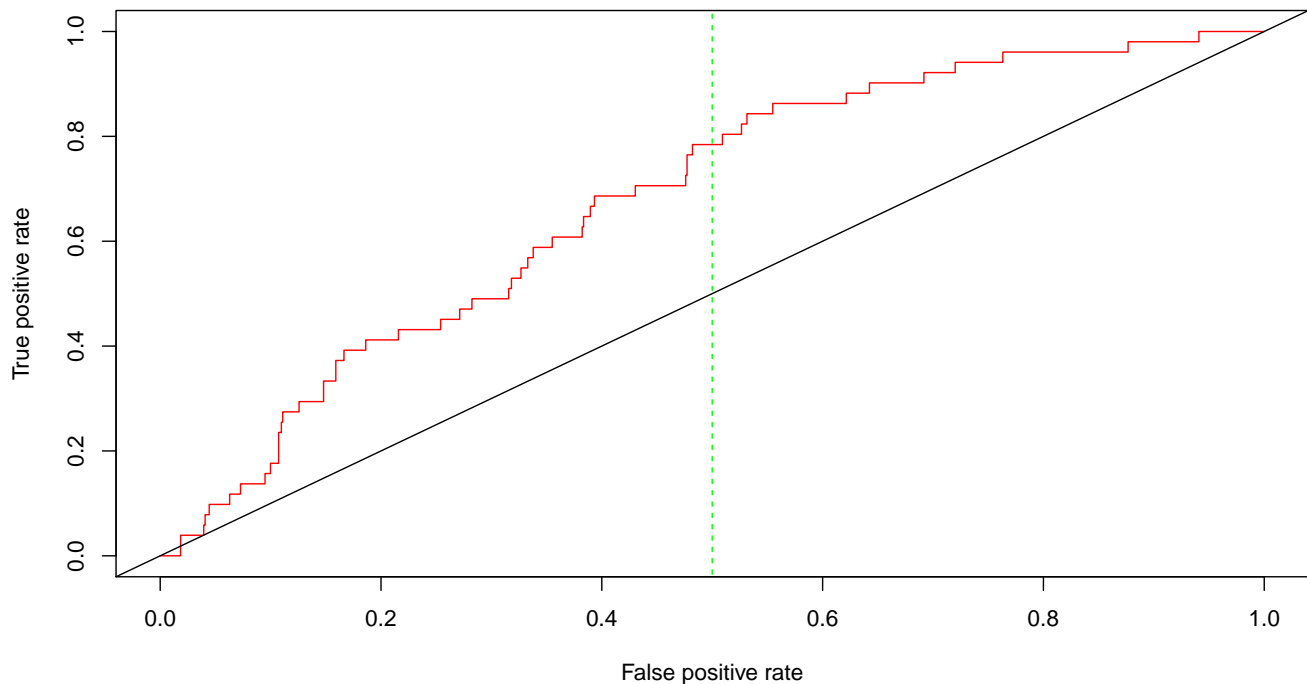
```
procent(tab3)
[1] 94.08353

czulosc(tab3)
[1] 0

precyzja(tab3)
[1] 0
```

Wychodzą równe zero, więc model jest fatalny. Mimo to krzywa ROC i parametr AUC tego nie potwierdzają:

[1] 0.678296



6.5.2 Jądro wielomianowe

Dopasujemy model i zrobmy predykcję:

```
mod_svm4 <- svm(class~., data=Train, type="C", kernel="polynomial",
               gamma=obj$best.parameters[1], cost=obj$best.parameters[2],
               probability=TRUE)
pred4 <- predict(mod_svm4, Test, probability=TRUE)
pred_praw4 <- attr(pred4,"probabilities")[,2]
pred_klas4 <- predict(mod_svm4, Test)
```

Przyjrzyjmy się tabeli rekasyfikacji:

	klasy_prawdziwe	
predykcja_klasy	0	1
0	759	44
1	52	7

A tak wyglądają czułość, precyzja i procent poprawnej klasyfikacji:

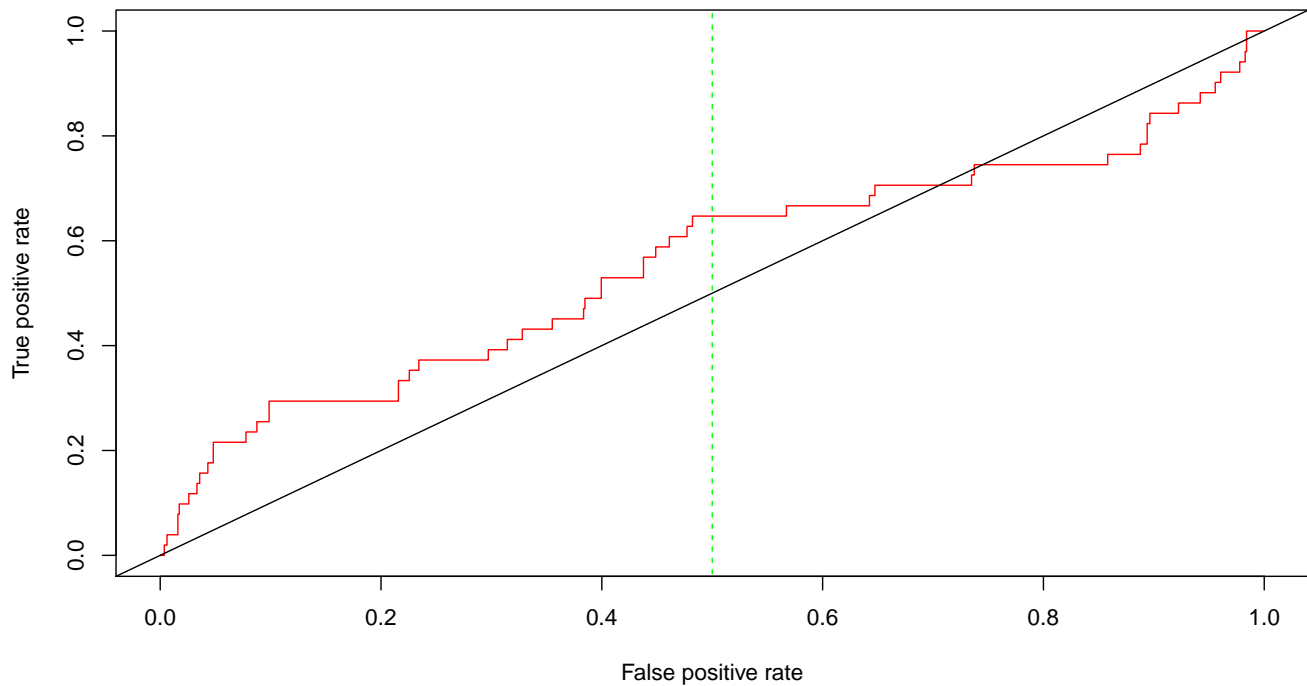
```
procent(tab4)
[1] 88.86311

czulosc(tab4)
[1] 0.1186441

precyzja(tab4)
[1] 0.1372549
```

Jest dużo lepiej. Spójrzmy jeszcze na krzywą ROC i AUC:

[1] 0.5564179



6.5.3 Jądro sigmoidalne

Dopasujemy model i zrobmy predykcję:

```
mod_svm5 <- svm(class~., data=Train, type="C", kernel="sigmoid",
  gamma=obj$best.parameters[1], cost=obj$best.parameters[2],
  probability=TRUE)
pred5 <- predict(mod_svm5, Test, probability=TRUE)
pred_praw5 <- attr(pred5,"probabilities")[,2]
pred_klas5 <- predict(mod_svm5, Test)
```

Przyjrzyjmy się tabeli rekasyfikacji:

	klasy_prawdziwe	
predykcia_klasy	0	1
0	771	44
1	40	7

A tak wyglądają czułość, precyzja i procent poprawnej klasyfikacji:

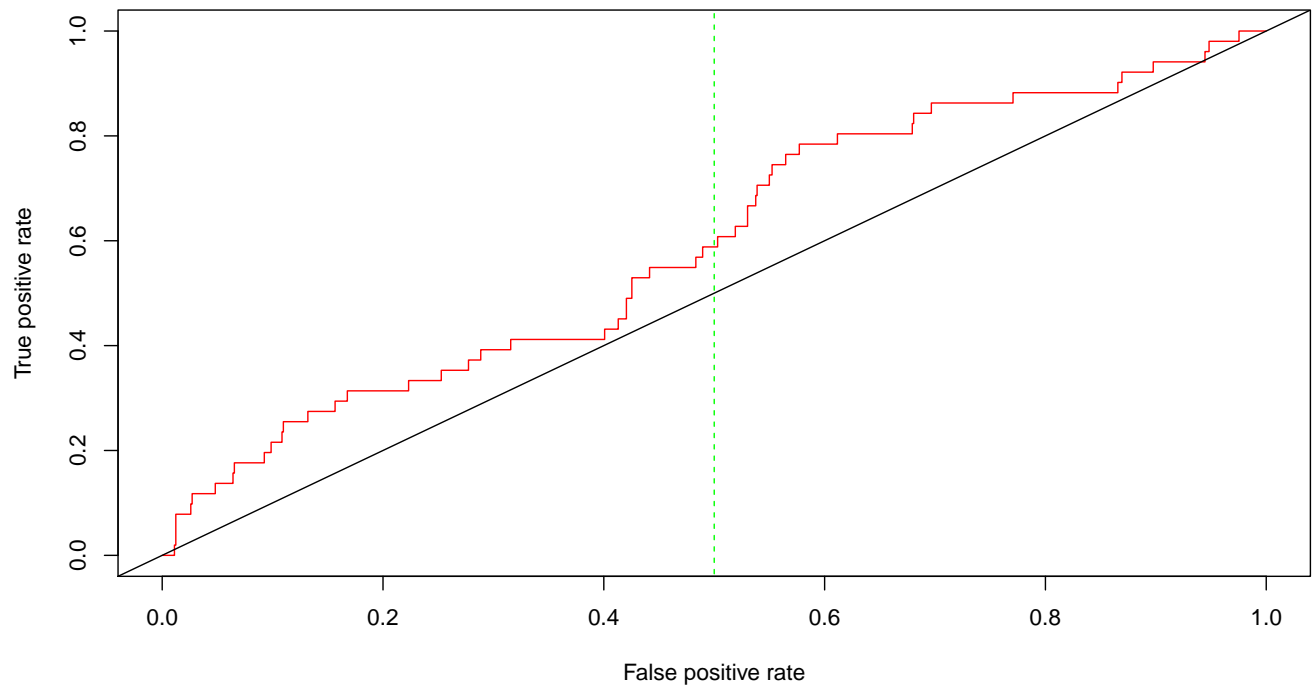
```
procent(tab5)
[1] 90.25522

czulosc(tab5)
[1] 0.1489362

precyzja(tab5)
[1] 0.1372549
```

Jest słabiej niż w wielomianowym. Spójrzmy jeszcze na krzywą ROC i AUC:

```
[1] 0.5928532
```

Wygląda na to, że najlepszy okazał się model SVM z jądrem wielomianowym. Ten też zatem będziemy analizować przy porównaniu wszystkich metod.

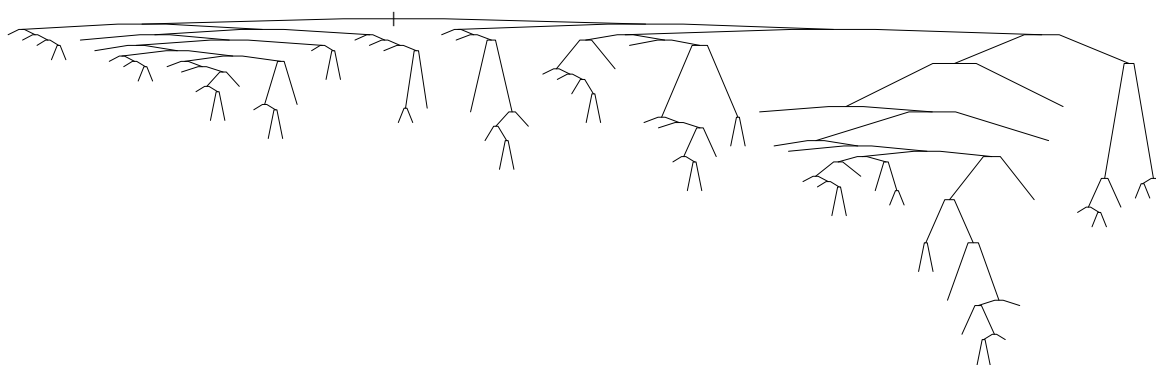
ROZDZIAŁ 7

DRZEWA DECYZYJNE

7.1 Graficzna prezentacja drzew decyzyjnych



Rysunek 7.1: Drzewo decyzyjne używające indeksu Giniego.



Rysunek 7.2: Drzewo decyzyjne używające entropii.

7.2 Dopasowanie modelu i predykcja

Dopasowanie klasyfikacji przy pomocy drzewa decyzyjnego używającego indeksu Giniego ma syntax jak poniżej:

```
Seismic.tree <- rpart(class~.,data=Train,cp=0.0001,minsplit=5)
```

Graficzne przedstawienie drzewa zaś ma postać 7.1.

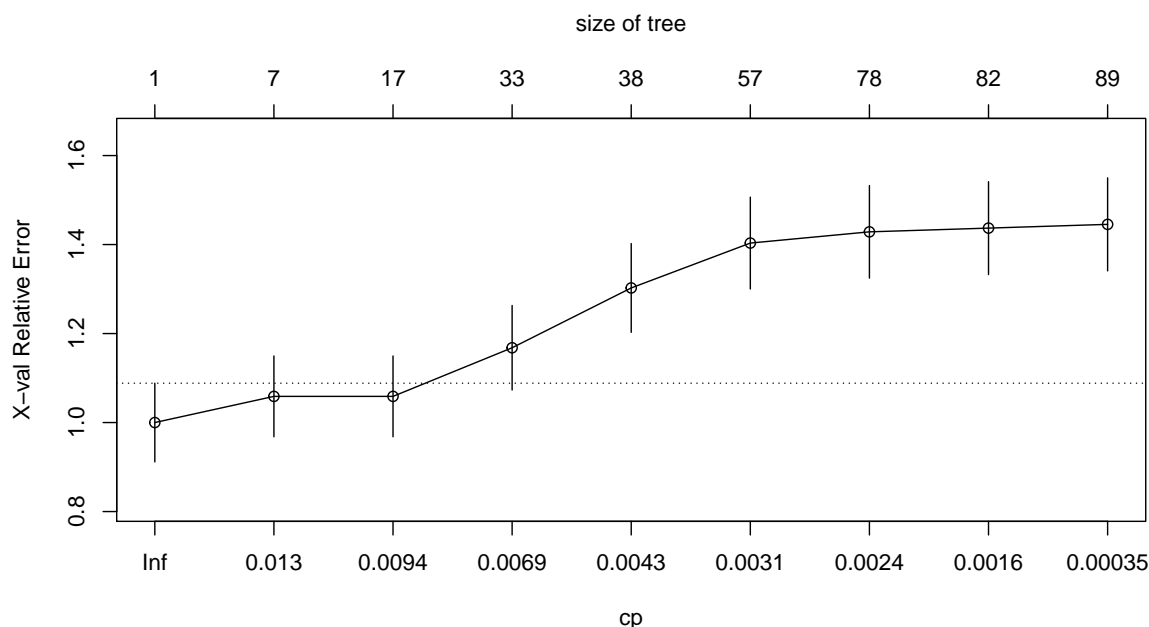
Dopasowanie klasyfikacji przy pomocy drzewa decyzyjnego używającego entropii ma syntax jak poniżej:

```
Seismic.tree.ent <- rpart(class~.,data=Train,cp=0.0001,minsplit=5,parms=list(split="information"))
```

Z kolei przedstawienie tego drzewa ma postać 7.2.

7.2.1 Wybór drzewa optymalnego

Wybieramy drzewo optymalne w oparciu o kryterium kosztu- złożoności, stosując regułę 1SE.



"A good choice of cp for pruning is often the leftmost value for which the mean lies below the horizontal line." - w takim wypadku nie trzeba przycinać drzewa.

Wykorzystajmy jeszcze funkcję `tune.rpart()` do wyliczenia optymalnego argumentu `minsplit`, który wyniósł 12.

```
obj <- tune.rpart(class~., data = Train, minsplit = c(1:15))
summary(obj)
```

Parameter tuning of 'rpart.wrapper':

- sampling method: 10-fold cross validation

- best parameters:

```
minsplit
5
```

- best performance: 0.07550074

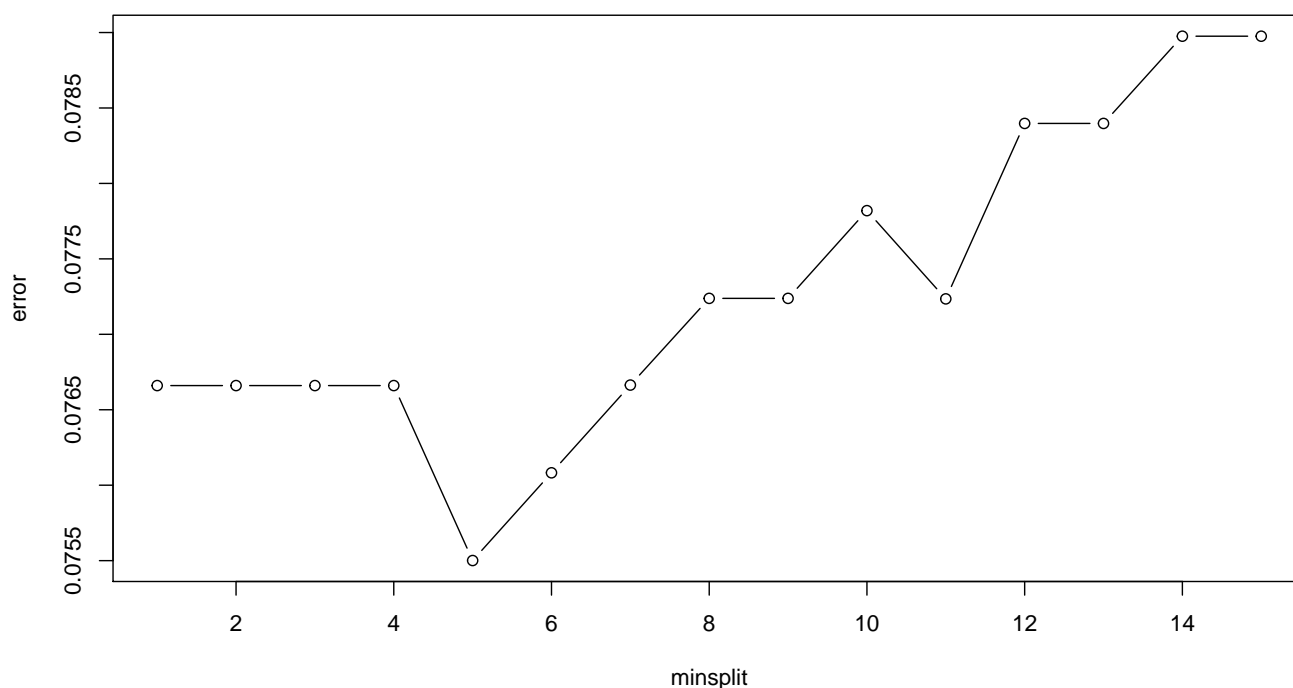
- Detailed performance results:

```
minsplit  error dispersion
1         1 0.07666017 0.02417954
```

```

2      2 0.07666017 0.02417954
3      3 0.07666017 0.02417954
4      4 0.07666017 0.02417954
5      5 0.07550074 0.02277257
6      6 0.07608213 0.02333686
7      7 0.07666353 0.02433951
8      8 0.07723820 0.01819095
9      9 0.07723820 0.01819095
10     10 0.07781960 0.01842992
11     11 0.07723484 0.02050928
12     12 0.07839763 0.02162029
13     13 0.07839763 0.02162029
14     14 0.07897567 0.02586128
15     15 0.07897567 0.02586128

```



Rysunek 7.3: Performance of rpart.wrapper !

Widać to również na wykresie Performance of rpart.wrapper !, dla którego minimum jest osiągnięte w punkcie 12. Zatem najoptymalniejszy model dla drzewa klasyfikacyjnego to:

```
Seismic.tree <- rpart(class~.,data=Train,cp=0.0001,minsplit=12)
```

7.2.2 Predykcja dla optymalnego drzewa

Predykcja dla tak dopasowanego drzewa wygląda następująco:

```

pred.tree.class <- predict(Seismic.tree, newdata=Test, type="class")
pred.tree.prob <- predict(Seismic.tree, newdata=Test, type="prob")[,2]

```

7.3 Diagnostyka modelu

Tabela rekasyfikacji na zbiorze testowym wygląda następująco:

	klasy_prawdziwe	
predykcja_klasy	0	1
0	784	45
1	27	6

A procent poprawnego dopasowania, czułość i precyzja następująco:

```
procent(tab)
[1] 91.64733

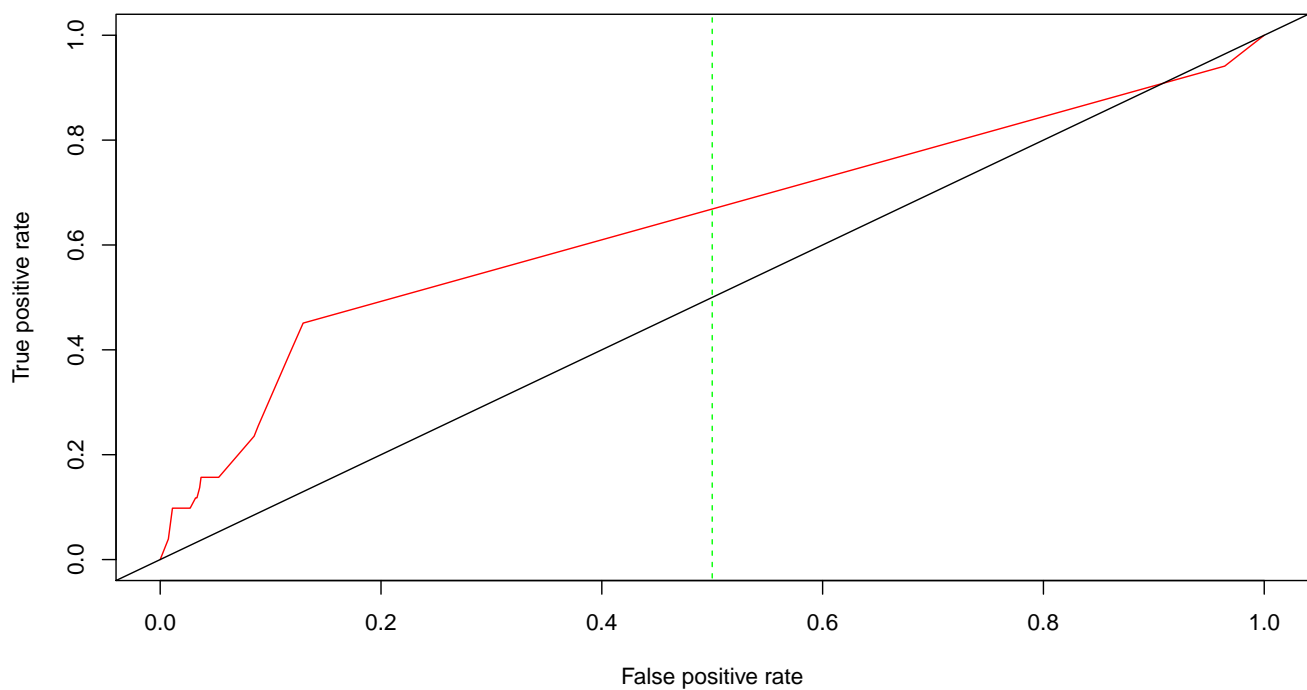
czulosc(tab)
[1] 0.1818182

precyzja(tab)
[1] 0.1176471
```

Pomimo w miarę dobrego poziomu procentowej poprawności klasyfikacji i dość dużej czułości w stosunku to poprzednich klasyfikatorów, optymalne drzewo regresyjne ma bardzo słabą precyzję.

7.4 Krzywa ROC i współczynnik AUC

I na koniec wyznaczmy krzywą ROC:



Współczynnik AUC wynosi:

```
[1] 0.642997
```

ROZDZIAŁ 8

METODY ŁĄCZENIA DRZEW DECYZYJNYCH

8.1 Bagging

Dopasujemy model opierając się na baggingu dla drzew decyzyjnych:

```
mod_bag <- bagging(class~,data=Train)
```

```
Error in eval(expr, envir, enclos): nie udało się znaleźć funkcji 'bagging'
```

Zróbmy predykcję:

```
predykacja <- predict.bagging(mod_bag,newdata=Test)
```

```
Error in eval(expr, envir, enclos): nie udało się znaleźć funkcji 'predict.bagging'
```

```
pred_klas <- predykacja$class
```

```
Error in eval(expr, envir, enclos): nie znaleziono obiektu 'predykacja'
```

```
pred_praw <- predykacja$prob[,2]
```

```
Error in eval(expr, envir, enclos): nie znaleziono obiektu 'predykacja'
```

Popatrzymy na tabelę klasyfikacji na zbiorze testowym:

```
      klasy_prawdziwe
predykacja_klasy  0   1
      0  734  34
      1   77  17
```

I na procent poprawnego dopasowania, czułość i precyzję:

```
procent(tab)
```

```
[1] 87.12297
```

```
czulosc(tab)
```

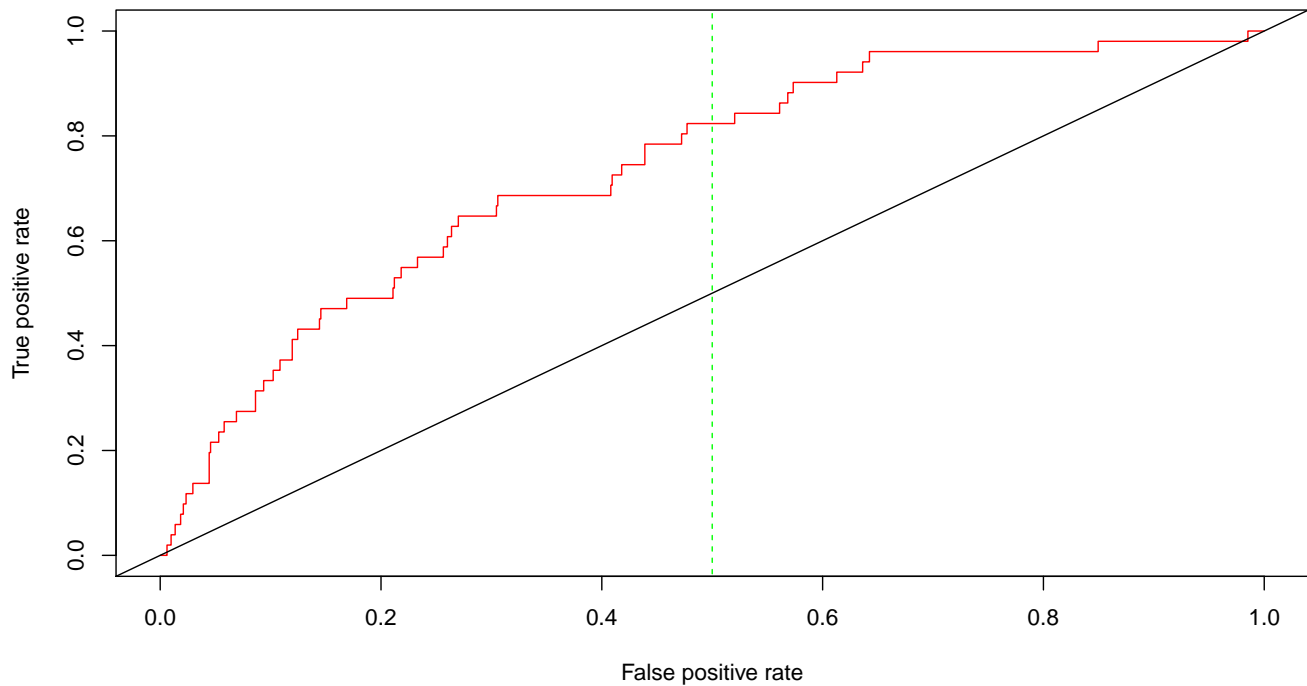
```
[1] 0.1808511
```

```
precyzja(tab)
```

```
[1] 0.3333333
```

Czułość jest nienajgorsza. Za to precyzja jest fatalna. Przyjrzyjmy się jeszcze krzywej ROC i współczynnikowi AUC:

```
[1] 0.7386669
```



Wygląda to całkiem dobrze.

8.2 Boosting

Dopasujemy model opierając się na boostingu dla drzew decyzyjnych:

```
mod_boo <- boosting(class~., data=Train, control = (minsplit = 10),
                    mfinal=20)
```

Error in eval(expr, envir, enclos): nie udało się znaleźć funkcji 'boosting'

Zróbmy predykcję na zbiorze testowym:

```
predykcja <- predict.boosting(mod_boo, newdata=Test)
```

Error in eval(expr, envir, enclos): nie udało się znaleźć funkcji 'predict.boosting'

```
pred_klas <- predykcja$class
```

Error in eval(expr, envir, enclos): nie znaleziono obiektu 'predykcja'

```
pred_praw <- predykcja$prob[,2]
```

Error in eval(expr, envir, enclos): nie znaleziono obiektu 'predykcja'

I przyjrzymy się tabeli klasyfikacji na zbiorze testowym:

```
      klasy_prawdziwe
predykcja_klasy  0    1
0      734   34
1      77   17
```

Inne współczynniki wyglądają następująco:

```
procent(tab)
```

```
[1] 87.12297
```

```
czulosc(tab)
```

```
[1] 0.1808511
```

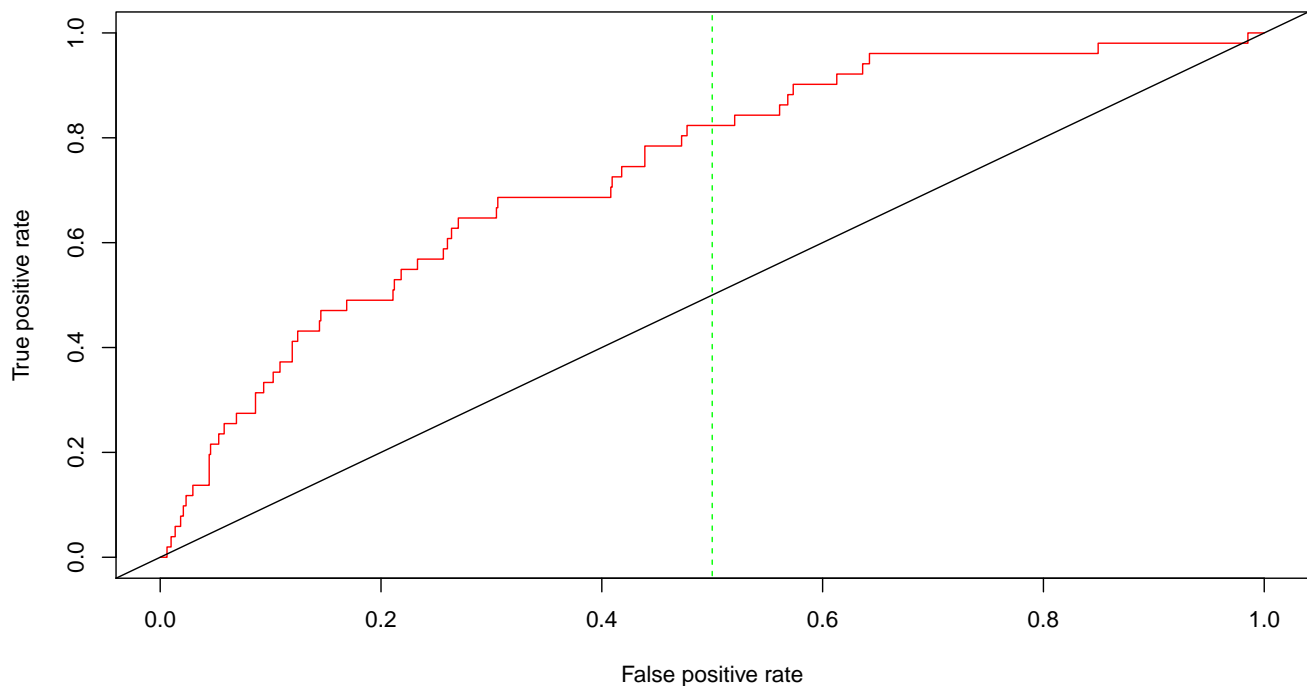
```
precyzja(tab)
```

```
[1] 0.3333333
```

Nie jest najlepiej...

Spójrzmy jeszcze na krzywą ROC i współczynnik AUC:

```
[1] 0.7386669
```



8.3 Lasy losowe

Dopasujemy do danych las losowy w następujący sposób:

```
mod_las <- randomForest(class~., data=Train)
```

```
Error in eval(expr, envir, enclos): nie udało się znaleźć funkcji 'randomForest'
```

Zróbmy predykcję:

```
pred_praw <- predict(mod_las,newdata=Test, type="prob")[,2]
```

```
Error in predict(mod_las, newdata = Test, type = "prob"): nie znaleziono obiektu 'mod_las'
```

```
pred_klas <- predict(mod_las,newdata=Test, type="response")
```

```
Error in predict(mod_las, newdata = Test, type = "response"): nie znaleziono obiektu 'mod_las'
```

Przyjrzyjmy się tabeli rekasyfikacji:

	klasy_prawdziwe	
predykcja_klasy	0	1
0	734	34
1	77	17

Oraz innym współczynnikiem:

```
procent(tab)
```

```
[1] 87.12297
```

```
czulosc(tab)
```

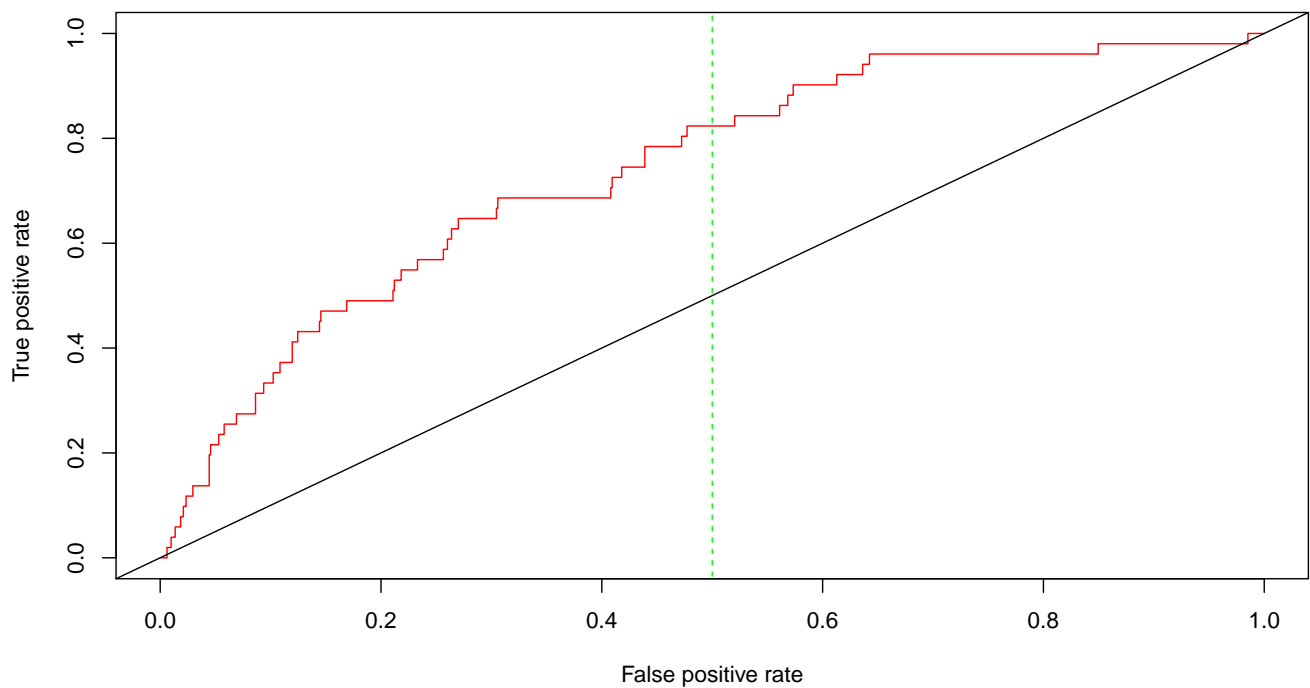
```
[1] 0.1808511
```

```
precyzja(tab)
```

```
[1] 0.3333333
```


Czułość jest na dobrym poziomie, natomiast precyzja w dalszym ciągu jest bardzo zła...
Zobaczmy jeszcze krzywą ROC i współczynnik AUC:

```
[1] 0.7386669
```



Krzywa ROC wygląda bardzo dobrze.

Na poniższym wykresie zobaczmy jeszcze, które zmienne są najbardziej istotne (te, co znajdują się na górze wykresu są najbardziej istotne).

```
Error in eval(expr, envir, enclos): nie udało się znaleźć funkcji 'varImpPlot'
```

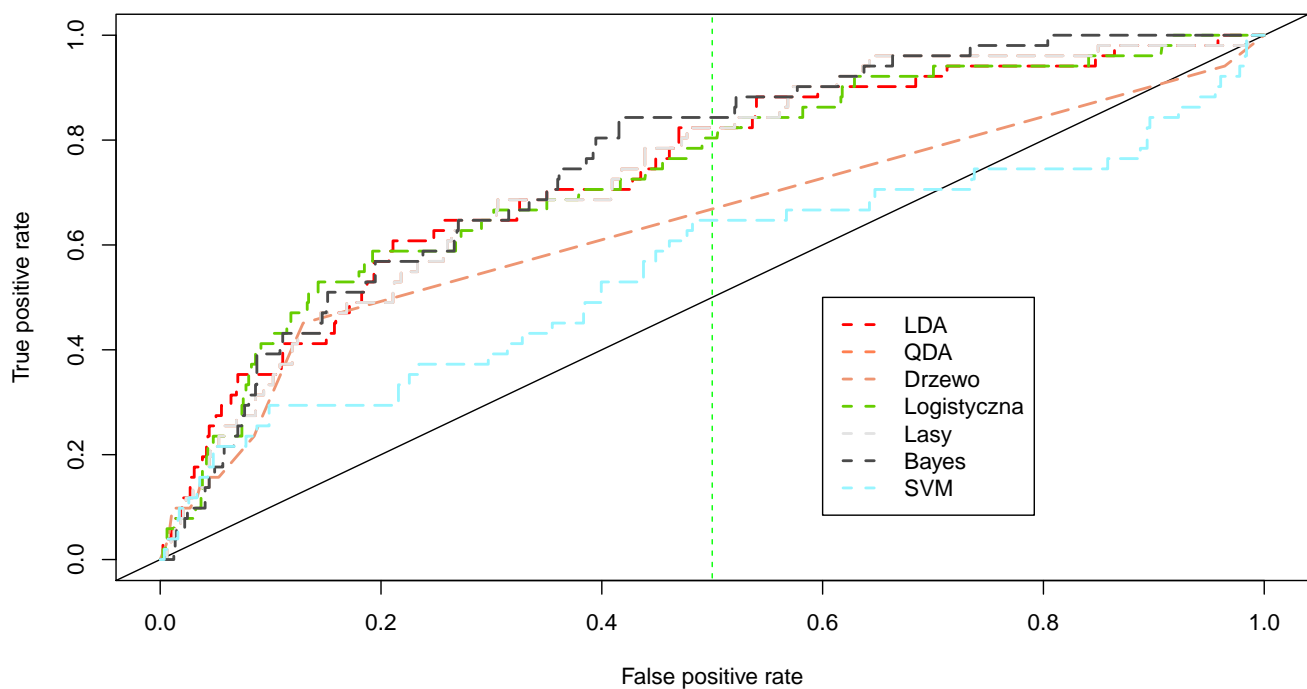
Zestawienie wyników dla klasyfikatorów użytych do analizy

ROZDZIAŁ 9

KRZYWE ROC

Wszystkie krzywe ROC dla klasyfikatorów wyglądają następująco. Najlepiej spisuje się Naiwny Klasyfikator Bayesa.

Porównanie krzywych ROC



ROZDZIAŁ 10

TABELA PODSUMOWUJĄCA

	Procent ¹	Czułość	Precyzja	AUC
lda	92.69	0.25	0.12	0.74
qda	87.12	0.18	0.33	0.74
logit	93.97	0.33	0.02	0.74
bayes	87.94	0.22	0.39	0.76
knn	90.26	0.12	0.10	-
svm	88.86	0.12	0.14	0.56
tree	91.65	0.18	0.12	0.64
bagging	87.12	0.18	0.33	0.74
boosting	87.12	0.18	0.33	0.74
las	87.12	0.18	0.33	0.74

Z powyższej tabeli widać wyraźnie, że najlepiej spisuje się klasyfikator naiwnego Bayesa. Ma on najwyższą precyzję (i to znacznie większą niż wszystkie pozostałe metody!) i najwyższą wartość parametru AUC. Jego czułość też należy raczej do tych lepszych. Niestety charakteryzuje się jednym z najniższych procentów poprawnej klasyfikacji. Patrząc jednak na precyzję i krzywą ROC możemy zdecydowanie stwierdzić, że jest on metodą, która najlepiej stosuje się do naszych danych. Precyzja jest tu ważna głównie ze względu na charakter naszych danych. Mianowicie zależy nam przede wszystkim na poprawnym wykryciu wstrząsów sejsmicznych. Czyli lepiej jest dla nas ogłosić fałszywy alarm, niż rzeczywistego trzęsienia nie wykryć.

Pozostałe metody sprawiają się niestety nie najlepiej (a niektóre wręcz fatalnie!).

¹tn. procent poprawnej klasyfikacji