

**Uniwersytet Warszawski**  
Wydział Matematyki, Informatyki i Mechaniki

**Grzegorz Szpak**

Nr albumu: 319400

# **Klasyfikacja wielowymiarowych szeregów czasowych przy ewoluujących pojęciach**

Praca magisterska  
na kierunku INFORMATYKA

Praca wykonana pod kierunkiem  
**dra Andrzeja Janusza**  
Instytut Informatyki

Grudzień 2016

## **Oświadczenie kierującego pracą**

Potwierdzam, że niniejsza praca została przygotowana pod moim kierunkiem i kwalifikuje się do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

Data

Podpis kierującego pracą

## **Oświadczenie autora (autorów) pracy**

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Data

Podpis autora (autorów) pracy

## **Streszczenie**

W pracy przedstawiono sposoby wydajnej klasyfikacji szeregów czasowych dla danych pochodzących ze źródła o zmiennym rozkładzie. Opisane zostały metody ekstrakcji cech z wielowymiarowego szeregu czasowego. Autor opisuje także metody wyboru przestrzeni atrybutów odpornej na zmiany rozkładu źródła.

## **Słowa kluczowe**

Eksploracja danych, wielowymiarowy szereg czasowy, ewoluujące pojęcia, dopasowanie dziediny, ekstrakcja cech, selekcja cech, lasy losowe, regresja logistyczna, maszyna wektorów wspierających

## **Dziedzina pracy (kody wg programu Socrates-Erasmus)**

11.4 Sztuczna inteligencja TODO

## **Klasyfikacja tematyczna**

D. Software TODO

## **Tytuł pracy w języku angielskim**

Classification of multivariate time series in the presence of concept drift



# Spis treści

<b>1. Wprowadzenie</b>	5
1.1. Uczenie z nadzorem a <i>concept drift</i>	5
1.2. Formalizacja problemu	6
1.2.1. Paradygmaty uczenia się	6
1.2.2. Ewolucja pojęć a <i>overfitting</i>	8
<b>2. Opis przeprowadzanego eksperymentu</b>	9
2.1. Przedstawienie badanego problemu	9
2.1.1. Opis zbioru danych	9
2.1.2. Ewaluacja jakości klasyfikatora	11
2.2. Opis użytych klasyfikatorów	11
2.2.1. Klasyfikator liniowy	11
2.2.2. Lasy losowe	13
2.3. Ewolucja pojęć w analizowanym zbiorze	13
<b>3. Ekstrakcja cech z szeregów czasowych</b>	15
<b>Bibliografia</b>	17



# Rozdział 1

## Wprowadzenie

### 1.1. Uczenie z nadzorem a *concept drift*

Podstawowym problemem rozważanym w teorii uczenia maszynowego jest problem uczenia z nadzorem (ang. *supervised learning*). Niech dane będą:

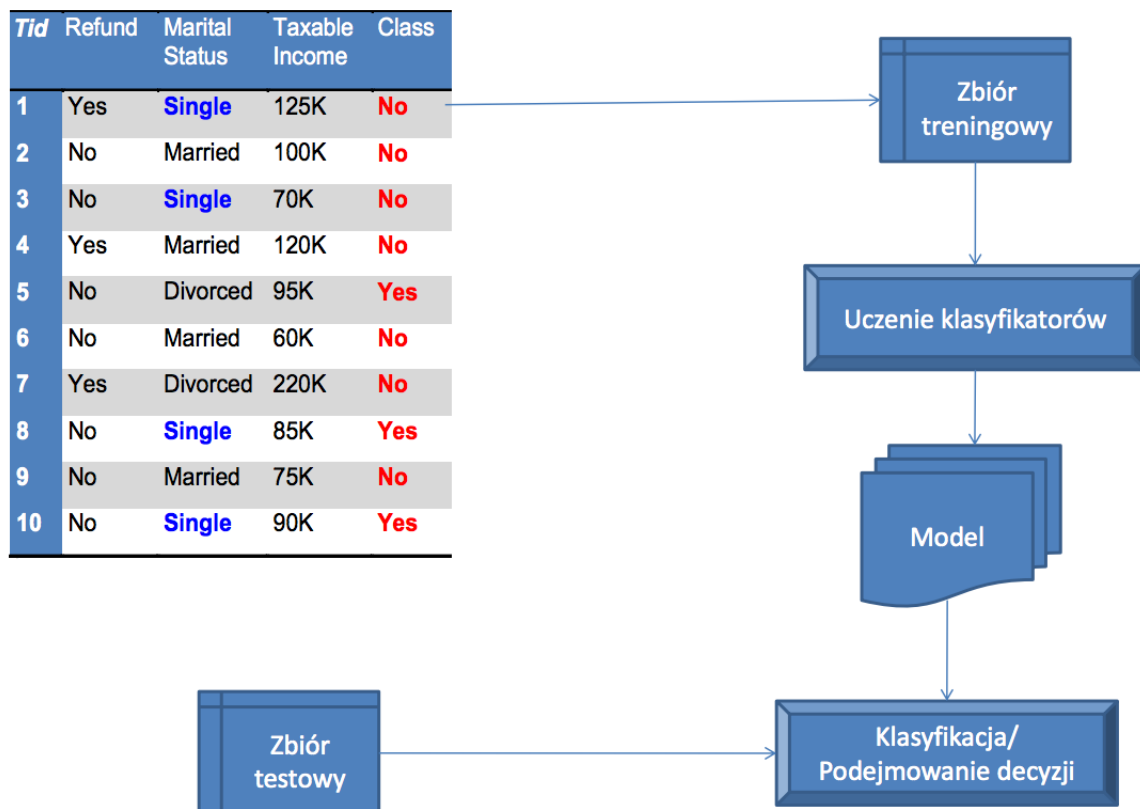
- zbiór  $X$  przykładów
- zbiór  $Y$  decyzji
- funkcja  $f : X \rightarrow Y$
- Parę zbiorów  $(X_{train} \subseteq X, Y_{train} \subseteq Y)$  instancji  $x_1, x_2, \dots, x_n$  oraz odpowiadających im decyzji  $f(x_1), f(x_2), \dots, f(x_n)$ , zwaną *zbiorem treningowym*

Zadanie uczenia z nadzorem polega na wyznaczeniu na podstawie zbioru treningowego oraz przy użyciu pewnego algorytmu uczącego (*klasyfikatora*) takiej funkcji  $c : X \rightarrow Y$  (zwanej *modelem*) będącej dobrą aproksymacją funkcji  $f$ . Jakość modelu  $c$  określa się, porównując jego wartości dla elementów skończonego zbioru testowego  $X_{test}$  z rzeczywistymi wartościami funkcji  $f$  dla tych elementów. Istotne jest przy tym założenie, że elementy zbiorów  $X_{train}$  oraz  $X_{test}$  losowane są ze zbioru  $X$  według tego samego rozkładu prawdopodobieństwa  $D$ .

Schemat rozwiązywania problemu uczenia z nadzorem przedstawia rys. 1.1.

Na przestrzeni ostatnich dziesięcioleci opracowanych zostało wiele algorytmów uczących wykazujących się dużą skutecznością w przeróżnych dziedzinach: od rozpoznawania obrazów, przez klasyfikację tekstów, rekomendację produktów, wykrywanie spamu, po przewidywanie zmian na giełdzie czy diagnostykę medyczną.

Sytuacja zmienia się diametralnie, gdy pominiemy założenie o równości rozkładów dla zbiorów treningowych i testowych. Problem ten nazywa się *ewolucją pojęć* (ang. *concept drift*) lub *dopasowaniem dziedziny* (ang. *domain adaptation*). Jest on szczególnie widoczny w zadaniach przetwarzania języka naturalnego (ang. *natural language processing*, w skrócie NLP). Rozpatrzmy dla przykładu problem rozpoznawania nazw własnych (ang. *named entity recognition*). Załóżmy, że klasyfikator uczony jest na podstawie danych encyklopedycznych oraz testowany na danych pochodzących z komunikatora internetowego. Obydwa zbiory, jakkolwiek powiązane, różnią się w znaczący sposób - przykładowo, szukanie wielkich liter może być bardzo pomocne w pierwszej dziedzinie, a nieść znacznie mniej informacji w wiadomościach z komunikatora.



Rysunek 1.1: Schemat uczenia z nadzorem

Stąd też właśnie w dziedzinie NLP powstało najwięcej metod mających rozwiązać problem ewoluujących pojęć. Przykładami takich metod są algorytm *structural correspondence learning* opisywany w [1] czy metoda odpowiedniego dopasowania przestrzeni parametrów zaproponowana przez Daume w [2].

Problem *domain adaptation* nie jest jednak często poruszany w przypadku klasyfikacji szeregów czasowych. W poniższej pracy autor przedstawia sposoby radzenia sobie z *concept drift* podczas klasyfikacji szeregów czasowych oraz wykonuje studium przypadku na wybranym zbiorze danych.

## 1.2. Formalizacja problemu

### 1.2.1. Paradygmaty uczenia się

Przyjmijmy definicje jak na początku sekcji 1.1. W zależności od rozkładów  $D_{train}$ ,  $D_{test}$  oraz od dostępności zbiorów  $Y_{train}$ ,  $X_{test}$ ,  $Y_{test}$ , można (za [?]) zdefiniować inne paradygmaty uczenia.

I tak, jeśli zbiór  $Y_{train}$  jest nieznany w momencie tworzenia modelu, mamy do czynienia z *uczeniem bez nadzoru* (ang. *unsupervised learning*).

Gdy zbiór  $X_{test}$  nie jest znany podczas uczenia, mowa o *uczeniu indukcyjnym* (ang. *inductive learning*). W przeciwnym razie takie uczenie nazywa się *uczeniem transdukcyjnym* (ang.



transductive learning).

W powyższym przykładach istotne jest założenie, iż zbiory  $X_{train}$ ,  $X_{test}$  pochodzą z tego samego rozkładu  $D$ . Odwrotna sytuacja rozpatrywana jest w paradygmacie *uczenia z przeniesieniem wiedzy* (ang. *transfer learning*). Przyjmuje się w nim, że dane są dwa różne rozkłady  $D^{source}$  i  $D^{target}$ . Model wyuczony na danych treningowych  $X_{train}^{source}, Y_{train}^{source}$  wykorzystywany jest zatem do klasyfikacji zbioru testowego  $X_{test}^{target}, Y_{test}^{target}$  pochodzących z rozkładu  $D^{target}$ . W poniższej pracy autor skupia się na problemie *dopasowania dziedziny*, który zakłada, że zbiór dostępnych klas  $Y$  jest ten sam dla  $D^{source}$  i  $D^{target}$ . Przeciwnieństwem dopasowania dziedziny jest zadanie *uczenia wielozadaniowego* (ang. *multi-task learning*, więcej między innymi w [4]), gdzie zbiory  $X_{train}$ ,  $X_{test}$  pochodzą z tego samego rozkładu, natomiast zbiory  $Y_{train}$ ,  $Y_{test}$  są różne.

Powyższe rozważania podsumowuje tabela 1.1.

Tabela 1.1: Paradymaty uczenia w teorii uczenia maszynowego. We wszystkich przypadkach zakładamy, że zbiór  $X_{train}$  jest dostępny podczas uczenia, podczas gdy zbiór  $Y_{test}$  nie jest znany.

Paradygmat	$Y_{train}$ dostępny?	$X_{test}$ dostępny?	Rozkład danych testowych
Indukcyjne uczenie bez nadzoru	Nie	Nie	$D^{source}$
Transdukcyjne uczenie bez nadzoru	Nie	Tak	$D^{source}$
Indukcyjne uczenie z nadzorem	Tak	Nie	$D^{source}$
Transdukcyjne uczenie z nadzorem	Tak	Tak	$D^{source}$
Indukcyjne uczenie bez nadzoru z przeniesieniem wiedzy	Nie	Nie	$D^{target}$
Transdukcyjne uczenie bez nadzoru z przeniesieniem wiedzy	Nie	Tak	$D^{target}$
Indukcyjne uczenie z nadzorem z przeniesieniem wiedzy	Tak	Nie	$D^{target}$
Transdukcyjne uczenie z nadzorem z przeniesieniem wiedzy	Tak	Tak	$D^{target}$

W poniższej pracy autor skupi się na problemie transdukcyjnego uczenia z nadzorem z przeniesieniem wiedzy. Przedstawione zostaną algorytmy, które wykorzystują dostępny zbiór  $X_{test}$  do znalezienia reprezentacji odpornej na zmiany rozkładu, co skutkować będzie zwiększoną jakością klasyfikacji w stosunku do standardowego podejścia opisanego w 1.1.

### 1.2.2. Ewolucja pojęć a *overfitting*

Mówiąc o problemie ewoluujących pojęć, należy wspomnieć o zagadnieniu przeuczenia (ang. *overfitting*). Polega on na zbudowaniu nadmiernie skomplikowanego modelu, co skutkuje słabą jego jakością.

Obydwa pojęcia mogą być mylone przy niewłaściwym sposobie walidacji modelu. Jeśli jakość klasyfikacji sprawdzana jest wyłącznie na zbiorach treningowym i testowym, zarówno *concept drift*, jak i *overfitting* dają podobne objawy - wysoki wynik na zbiorze treningowym oraz niski na zbiorze testowym. W przypadku przeuczenia jest to spowodowane nadmiernym dopasowaniem modelu do danych treningowych i jego niską zdolnością do uogólniania. Jeśli mamy do czynienia z ewoluującymi pojęciami, słaba jakość modelu jest spowodowana innym rozkładem dla zbioru testowego.

W rozróżnieniu obydwu sytuacji pomagać może zastosowanie *walidacji krzyżowej* (ang. *cross-validation*) na zbiorze treningowym. Walidacja krzyżowa polega na podziale zbioru treningowego na  $n$  równolicznych części. Następnie budowane jest  $n$  modeli, przy czym  $n - 1$  części tworzy zbiór treningowy, natomiast pozostała część - zbiór testowy. Ostateczny wynik jest średnim wynikiem powstałych  $n$  modeli.

Nietrudno zauważyć, że w przypadku *concept drift* nie powinno się zauważyć znacznego spadku jakości modelu przy wykonaniu walidacji krzyżowej - w tym przypadku zbiór testowy pochodzi z tej samej dziedziny co treningowy. Inaczej będzie w przypadku przeuczenia - tu wynik walidacji krzyżowej będzie wyraźnie niższy niż wynik na zbiorze treningowym (tabela 1.2).

Tabela 1.2: *Concept drift* a *overfitting* - obniżony wynik modelu 1. przy walidacji krzyżowej świadczy o przeuczeniu, a nie występowaniu ewoluujących pojęć. W drugim przypadku model mimo dobrej umiejętności klasyfikacji elementów pochodzących z rozkładu  $D^{source}$ , cierpi na spadek jakości przy ewaluacji na zbiorze pochodzącym z rozkładu  $D^{target}$ .

	Wynik na $X_{train}$	Wynik CV	Wynik na $X_{test}$
Model 1	0.98	0.73	0.68
Model 2	0.97	0.92	0.76

## Rozdział 2

# Opis przeprowadzanego eksperymentu

### 2.1. Przedstawienie badanego problemu

#### 2.1.1. Opis zbioru danych

Dane, na których sprawdzana była jakość analizowanych algorytmów, pochodzą z konkursu *AAIA '15 Data Mining Competition: Tagging Firefighter Activities at a Fire Scene*<sup>1</sup> organizowanego przez Uniwersytet Warszawski oraz Szkołę Główną Służby Pożarniczej w Warszawie. Na potrzeby konkursu zebrano odczyty pochodzące z "inteligentnego kombinezonu", który monitoruje ruchy oraz funkcje życiowe strażaka. W skład kombinezonu wchodziło po siedem akcelerometrów i żyroskopów umieszczonych na: tułowi, ramionach, dłoniach i nogach (rys. 2.1).



Rysunek 2.1: Rozmieszczenie czujników na ciele strażaka

Każda instancja w zbiorze danych składała się z zebranych w krótkich odcinkach czasu (około 1.8 s) odczytów z zamontowanych sensorów. Dla każdego akcelerometru oraz żyroskopu zebrano po 400 odczytów wzdłuż osi  $x$ ,  $y$ ,  $z$ , wraz ze względny czas wykonania odczytu. Dla każdego wiersza daje to 42 - wymiarowy szereg czasowy o długości 400. Jak łatwo wywnioskować, średni odstęp między kolejnymi odczytami wynosił około 4.5 ms. Zestaw

<sup>1</sup><https://knowledgepit.fedcsis.org/contest/view.php?id=106>

atrybutów uzupełniały dodatkowo 42 agregaty ze wskazań urządzeń monitorujących funkcje życiowe strażaka (takie jak EKG, częstotliwość oddechu, temperatura skóry). Nieprzetworzone dane zawierały więc  $400 * 43 + 42 = 17242$  atrybuty. Zarówno zbiór treningowy, jak i testowy składały się z 20000 przykładów. Bardzo istotny dla dla analizy tego zbioru danych okazał się fakt, że dane treningowe i testowe pochodziły od różnych czteroosobowych grup strażaków.

Zadaniem uczestników konkursu było przypisanie każdej instancji w zbiorze postury strażaka w danym momencie oraz wykonywanej przez niego czynności. Zastosowane algorytmy miały pomóc w stworzeniu systemu monitorującego bezpieczeństwo strażaka podczas akcji.

Jako że problem klasyfikacji wieloetykietowej nie jest tematem niniejszej pracy, autor postanowił skupić się na problemie przewidywania czynności wykonywanej przez strażaka. Zbiór klas liczył więc 16 elementów. Postawiony problem utrudniał dodatkowo fakt, że klasy były wysoce niezbalansowane - najczęstsza klasa (*manipulating*) wystąpiła w zbiorze treningowym 6349 razy, podczas gdy najrzadsza (*signal\_hose\_pullback*) - jedynie 98 razy. Rozkład klas przedstawia rys. 2.2.



Rysunek 2.2: Rozkład klas w zbiorze treningowym

### 2.1.2. Ewaluacja jakości klasyfikatora

Standardową miarą jakości jest precyzja (ang. *accuracy*):

$$ACC(c) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}[c(\mathbf{x}_i) = y_i]$$

W opisywanym konkursie zastosowano modyfikację tej miary zwaną *balanced accuracy*, zdefiniowaną następująco:

$$ACC_i(c) = \frac{|j : c(\mathbf{x}_j) = y_j = i|}{|j : y_j = i|} \quad (2.1)$$

$$BAC(c) = \frac{\sum_{i=1}^l ACC_i(c)}{l} \quad (2.2)$$

Nietrudno zauważyć, że miara *BAC* jest bardziej wrażliwa na błędną klasyfikację rzadkich klas niż standardowa miara *ACC*.

## 2.2. Opis użytych klasyfikatorów

W kolejnych rozdziałach przedstawione zostaną metody mające na celu zwiększenie jakości modelu budowanego na danych, w których obecny jest problem ewoluujących pojęć. Użyteczność tych metod sprawdzana będzie dla trzech różnych algorytmów uczenia: klasyfikatorze opartym na regresji logistycznej, maszynie wektorów nośnych (ang. *support vector machine*, SVM) oraz drzewach decyzyjnych.

### 2.2.1. Klasyfikator liniowy

Regresja logistyczna i maszyna wektorów wspierających należą do grupy klasyfikatorów liniowych. Przyjmijmy oznaczenia jak w sekcji 1.1. Załóżmy, że zbiór klas  $Y$  jest dwuelementowy - dla uproszczenia niech  $Y = \{+1, -1\}$ . Niech dalej  $y \in Y$  będzie decyzją dla elementu  $\mathbf{x} = (x_1, x_2, \dots, x_m) \in X \subseteq \mathbb{R}^m$ . Klasyfikatorem liniowym nazywamy  $m - 1$  - wymiarową hiperpłaszczyznę rozdzielającą punkty z  $X$ . Niech  $\mathbf{w} = (w_1, w_2, \dots, w_m) \in \mathbb{R}^m$  będzie wektorem współczynników tej hiperpłaszczyzny. Uczenie klasyfikatora liniowego zwykle polega na minimalizacji wartości pewnej funkcji błędu  $l(\mathbf{w})$  na zbiorze treningowym.

#### Klasyfikacja oparta na regresji logistycznej (*logit*)

Niech

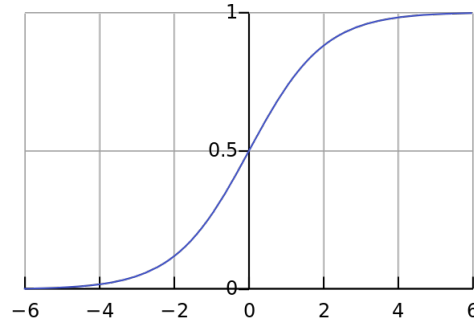
$$g(z) = \frac{1}{1 + e^{-z}}$$

zwana będzie funkcją logistyczną. Jej wykres przedstawia rysunek 2.3.

Funkcja  $g$  jest oczywiście ciągła. Jednocześnie  $\lim_{z \rightarrow -\infty} g(z) = 0$  oraz  $\lim_{z \rightarrow \infty} g(z) = 1$ . Dzięki tym właściwościom nadaje się ona do modelowania prawdopodobieństwa jakiegoś zjawiska. Klasyfikator bazujący na regresji logistycznej modeluje prawdopodobieństwo należenia do klasy pozytywnej przez funkcję:

$$p_w(\mathbf{x}) = g(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}}},$$

przy czym klasyfikacja dokonywana jest przez:



Rysunek 2.3: Wykres funkcji sigmoidalnej

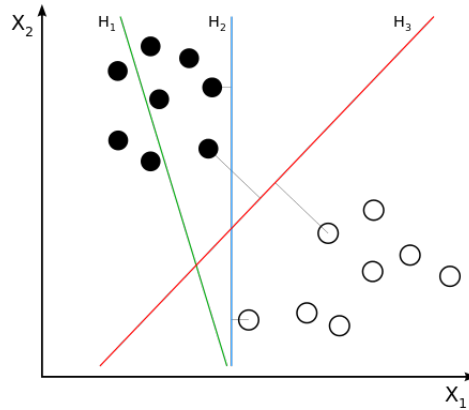
$$c_w(\mathbf{x}) = 1 \iff p_w(\mathbf{x}) \geq \frac{1}{2}.$$

Funkcja straty zdefiniowana jest następująco:

$$l_{log\_loss}(\mathbf{w}) = \sum_{i=1}^n \log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i}).$$

### Maszyna wektorów nośnych

Innym typem klasyfikatora liniowego jest (ang. *support vector machine*, SVM). Uczenie SVM polega na znalezieniu hiperpłaszczyzny o największej odległości od punktów obydwu klas (rys. 2.4) - z tego powodu SVM nazywany jest klasyfikatorem maksymalnego marginesu (ang. *max margin classifier*).



Rysunek 2.4: Płaszczyzna  $H_1$  nie rozdziela klas.  
Płaszczyzna  $H_2$  rozdziela je, ale z niewielkim marginesem.  
Płaszczyzna  $H_3$  rozdziela je z maksymalnym marginesem.

Założenie o liniowej separowalności klas jest jednak rzadko spełnione. Uczenie klasyfikatora SVM polega więc na znalezieniu hiperpłaszczyzny  $\mathbf{w}$ , która minimalizuje funkcję straty daną jako

$$l_{hinge\_loss}(\mathbf{w}) = \sum_{i=1}^n \max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i).$$

## Klasyfikator "jeden przeciwko wszystkim"

Jak wspomniano wcześniej, zarówno regresja logistyczna, jak i SVM są klasyfikatorami binarnymi - zakładają, że zbiór klas  $Y$  jest dwuelementowy. Aby użyć ich do klasyfikacji danego zbioru, potrzebna była metoda klasyfikacji wieloklasowej. Zastosowano podejście zwane "jeden przeciwko wszystkim" (ang. *one-vs.-all*, OvA, bądź *one-vs.-rest*, OvR).

Niech  $Y = \{1, 2, \dots, k\}$ . Klasyfikacja OvA polega na nauczaniu  $k$  binarnych klasyfikatorów  $c_1, c_2, \dots, c_k$ . Klasa dla  $j$ -tego klasyfikatora zdefiniowana jest następująco:

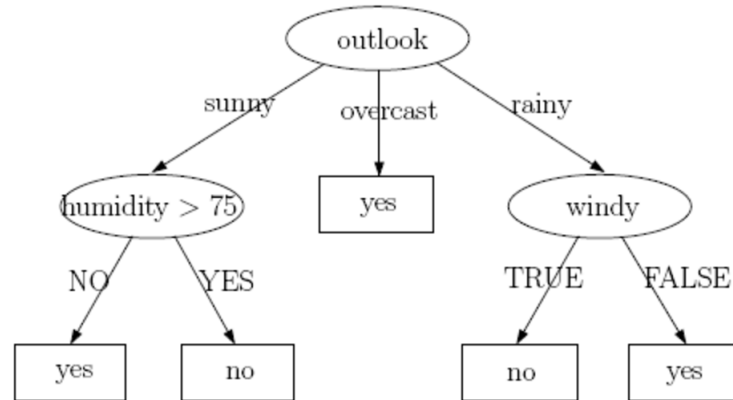
$$y_i^j = 1 \iff y_i = j$$

Ostatecznie elementowi  $x$  przypisywana jest klasa, której klasyfikator daje najmniejszą wartość funkcji błędu:

$$c(\mathbf{x}) = \arg \min_{j \in 1 \dots k} l_j(\mathbf{x})$$

### 2.2.2. Lasy losowe

Innym typem klasyfikatora jest las losowy. Jest to klasyfikator wykorzystujący prostszy klasyfikator - drzewo decyzyjne. Drzewo decyzyjne to etykietowane drzewo, którego każdy węzeł odpowiada przeprowadzeniu pewnego testu na wartościach atrybutów. Z węzła wewnętrznego wychodzi tyle gałęzi, ile jest możliwych wyników testu odpowiadającego temu węzłowi. Każdy liść zawiera decyzję o klasyfikacji obiektu. Przykładowe drzewo decyzyjne przedstawia rys. 2.5.



Rysunek 2.5: Przykładowe drzewo decyzyjne

Uczenie lasu losowego polega na uczeniu określonej liczby drzew decyzyjnych - każdego na losowej podprzestrzeni atrybutów. Finalna klasa jest modą klas wyznaczonych przez poszczególne drzewa. Lasy losowe, podobnie jak drzewa decyzyjne, wspierają klasyfikację wieloklasową.

## 2.3. Przykład Ewolucji pojęć w analizowanym zbiorze

Okazuje się, że głównym wyzwaniem w konkursie opisywanym w sekcji 2.1.1 są różnice w wykonywaniu poszczególnych czynności między strażakami. Aby się o tym przekonać, nauczono las losowy na 42 atrybutach opisujących funkcje życiowe, następnie zmierzono jego

jakość (używając miary  $BAC$ ) na zbiorach treningowym i testowym. Do pomiaru jakości na zbiorze treningowym zastosowano trójwarstwową walidację krzyżową. Problem *concept drift* spowodował drastyczne obniżenie jakości klasyfikatora, która spadła z 0.993 na 0.079.



## Rozdział 3

# Ekstrakcja cech z szeregów czasowych



# Bibliografia

- [1] John Blitzer, Ryan McDonald, Fernando Pereira, *Domain adaptation with structural correspondence learning*
- [2] Hall Daume, *Frustratingly easy domain adaptation*
- [3] Andrew Arnold, Ramesh Nallapati, William W. Cohen, *A comparative study of methods for transductive transfer learning*
- [4] R. K. Ando, T. Zhang, *A framework for learning predictive structures from multiple tasks and unlabeled data*
- [4] L. Breiman, *Random forests*, Machine Learning, 45(1), 5-32, 2001.