Statystyka wielowymiarowa
Laboratorium nr 2

Kamil Szkoła
Patryk Krukowski
Data Science, sem. 1

2 czerwca 2022

Spis treści

1 Wstęp

Naszym celem jest dokonanie analizy zbioru *titanic.csv* (dane dotyczące pasażerów statku Titanic podczas jego dziewiczego rejsu w 1912 roku) metodą regresji logistycznej w celu klasyfikacji zmiennej *Survived*, która mówi o tym, czy dany pasażer przeżył, czy też nie. Dodatkowo chcemy porównać wyniki klasyfikacji z następującymi metodami:

- LDA,
- QDA,
- kNN (z optymalnie wybraną liczbą sąsiadów k).

Zmienne obecne w danych:

- Survived (zmienna objaśniana),
- PClass,
- Sex,

- Age,
- Parch,
- Fare,
- Embarked,
- Has_cabin,
- FamilySize,
- IsAlone,
- Title.

Przejdziemy teraz do procesu odpowiedniego przygotowania danych.

2 Analiza

2.1 Przygotowanie danych

Przed analizą musimy odpowiednio przygotować dataset, ustawiając odpowiednie zmienne jako kategoryczne (funkcja *factor*). Dzielimy także dane na zbiór treningowy (75% dostępnych danych) oraz zbiór testowy (25% dostępnych danych).

```
> # Usuwamy wartosci zakodowane jako NA
> df <- read.csv('titanic.csv')
> df <- subset(df, select = -c(X))
> df <- na.omit(df)
> # Poprawiamy kodowanie zmiennej
> df <- df[df$Age <= 3,]
> df$Survived <- as.factor(df$Survived)
> # Dzielimy dane na zbiór testowy i treningowy
> smp_size <- floor(0.75 * nrow(df))
> set.seed(123)
> train_ind <- sample(seq_len(nrow(df)), size = smp_size)
> train <- df[train_ind, ]
> test <- df[-train_ind, ]
>
```

Przejdziemy teraz do właściwej analizy.

2.2 Dopasowanie modelu regresji logistycznej

Dopasowujemy model regresji logistycznej do części treningowej rozważanego zbioru danych (dla wszystkich zmiennych).

```
> # Dopasowanie modelu regresji logistycznej, istnieja istotne zmienne
> dir_logistic <- list()
> dir_logistic$fit <- glm(Survived ~ ., family = binomial, data = train)
> summary(dir_logistic$fit)

Call:
glm(formula = Survived ~ ., family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4706  -0.5959  -0.4272   0.6699   2.4205

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.37997    1.02133   3.309 0.000935 ***
Pclass      -0.77190    0.23230  -3.323 0.000891 ***
Sex         -1.95262    0.38504  -5.071 3.95e-07 ***
Age         -0.54023    0.14377  -3.758 0.000171 ***
Parch        0.13499    0.21666   0.623 0.533247
Fare         0.06527    0.17021   0.383 0.701385
Embarked     0.21798    0.16206   1.345 0.178603
Has_Cabin    0.66884    0.33918   1.972 0.048615 *
FamilySize  -0.41650    0.16707  -2.493 0.012671 *
IsAlone     -0.63046    0.33264  -1.895 0.058054 .
Title        0.23173    0.13361   1.734 0.082858 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 874.78  on 659  degrees of freedom
Residual deviance: 591.43  on 649  degrees of freedom
AIC: 613.43

Number of Fisher Scoring iterations: 5
```

Widzimy, że istnieją zmienne, które są istotne dla odpowiedzi modelu:

- PClass,
- Sex,
- Age,
- Has_Cabin,
- FamilySize.

Usuwamy z modelu zmienne, które są nieistotne (dopasowujemy nowy model do części treningowej z usuniętymi nieistotnymi zmiennymi).

```
> dir_logistic$fit <- glm(Survived ~ . -Parch -Fare -Embarked -IsAlone -Title,
+                         family = binomial, data = train)
> summary(dir_logistic$fit)

Call:
glm(formula = Survived ~ . - Parch - Fare - Embarked - IsAlone -
    Title, family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5010  -0.5993  -0.4257   0.6457   2.2970

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.90865    0.61826   6.322 2.58e-10 ***
Pclass      -0.86078    0.18412  -4.675 2.94e-06 ***
Sex         -2.61711    0.22191 -11.793  < 2e-16 ***
Age         -0.54783    0.14223  -3.852 0.000117 ***
Has_Cabin    0.74463    0.33888   2.197 0.027996 *
FamilySize  -0.17744    0.07488  -2.370 0.017801 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 874.78  on 659  degrees of freedom
Residual deviance: 601.36  on 654  degrees of freedom
AIC: 613.36

Number of Fisher Scoring iterations: 4
```

Przeanalizujemy teraz charakter wpływu każdego predyktora osobno na odpowiedź modelu:

- PClass - mały, ujemny,
- Sex - duży, ujemny,
- Age - mały, ujemny,
- Has_Cabin - mały, dodatni,
- FamilySize - mały, ujemny.

Największy wpływ na odpowiedź modelu ma zmienna *Sex*. Nie powinno nas to dziwić, gdyż to właśnie kobiety miały pierwszeństwo do szalup ratunkowych.

Dopasowanie modelu do danych jest sensowne, ponieważ z podsumowania wydrukowanego przez funkcję *summary* wynika, że null deviance (model stały) jest większe od residual deviance (model z predyktorami).

Ocenimy teraz skuteczność dopasowanego modelu. W tym celu policzymy miarę accuracy oraz skonstruujemy macierz pomyłek. W tym celu policzymy odpowiednie prawdopodobieństwa:

```
> dir_logistic$probs <- predict(dir_logistic$fit, test, type = "response")
```

Upewnijmy się, czy aby na pewno $Y = 1$ oznacza, że $Survived = 1$.

```
> contrasts(df$Survived)
```

```
  1
0 0
1 1
```

Widzimy, że wszystko jest w porządku. A zatem możemy wyznaczyć odpowiednie klasy, bazując na wyliczonych wcześniej prawdopodobieństwach.

```
> dir_logistic$predicted <- ifelse(dir_logistic$probs > 0.5, 1, 0)
```

Wyznaczamy accuracy i macierz pomyłek, korzystając ze zbioru testowego.

```
> table(dir_logistic$predicted, test$Survived)
```

```
    0   1
  0 114  28
  1  14  64
```

```
> mean(dir_logistic$predicted == test$Survived)
```

```
[1] 0.8090909
```

Accuracy dla modelu regresji logistycznej wynosi (w przybliżeniu) 0.81.

2.3 Dopasowanie modelu LDA

Dopasujemy teraz model LDA (*Linear Discriminant Analysis*). Aby zachować zgodność z modelem regresji logistycznej, dopasowujemy tylko te zmienne, które zostały przez nas także dopasowane do modelu regresji logistycznej.

```
> dir_lda <- list()
> dir_lda$fit <- lda(Survived ~ . -Parch -Fare -Embarked -IsAlone -Title,
+                    family = binomial, data = train)
> dir_lda$fit
```

```
Call:
lda(Survived ~ . - Parch - Fare - Embarked - IsAlone - Title,
    data = train, family = binomial)

Prior probabilities of groups:
        0         1
0.6227273 0.3772727

Group means:
    Pclass       Sex      Age Has_Cabin FamilySize
0 2.549878 0.8418491 1.335766 0.1216545   1.844282
1 1.979920 0.3253012 1.269076 0.3935743   1.907631

Coefficients of linear discriminants:
                 LD1
Pclass     -0.5771798
Sex        -2.1067844
Age        -0.3542817
Has_Cabin   0.5650650
FamilySize -0.1145228
```

Wyznaczamy accuracy i macierz pomyłek, korzystając ze zbioru testowego.

```
> dir_lda$predicted <- predict(dir_lda$fit, test)
> table(dir_lda$predicted$class, test$Survived)

     0   1
  0 112  25
  1  16  67

> mean(dir_lda$predicted$class == test$Survived)

[1] 0.8136364
```

Accuracy dla modelu LDA wynosi (w przybliżeniu) 0.814.

2.4 Dopasowanie modelu QDA

Dopasujemy teraz model QDA (*Quadratic Discriminant Analysis*). Aby zachować zgodność z modelem regresji logistycznej, dopasowujemy tylko te zmienne, które zostały przez nas także dopasowane do modelu regresji logistycznej.

```
> dir_qda <- list()
> dir_qda$fit <- qda(Survived ~ . -Parch -Fare -Embarked -IsAlone -Title,
+                    family = binomial, data = train)
> dir_qda$fit
```

```
Call:
qda(Survived ~ . - Parch - Fare - Embarked - IsAlone - Title,
    data = train, family = binomial)

Prior probabilities of groups:
        0         1
0.6227273 0.3772727

Group means:
    Pclass       Sex      Age Has_Cabin FamilySize
0 2.549878 0.8418491 1.335766 0.1216545   1.844282
1 1.979920 0.3253012 1.269076 0.3935743   1.907631
```

Wyznaczamy accuracy i macierz pomyłek, korzystając ze zbioru testowego.

```
> dir_qda$predicted <- predict(dir_qda$fit, test)
> table(dir_qda$predicted$class, test$Survived)

     0   1
  0 108  19
  1  20  73

> mean(dir_qda$predicted$class == test$Survived)

[1] 0.8227273
```

Accuracy dla modelu QDA wynosi (w przybliżeniu) 0.823. Wynik jest wyższy, niż dla wcześniejszych modeli z uwagi na to, że QDA używa kwadratowej funkcji dyskryminacyjnej.

2.5 Dopasowanie modelu kNN

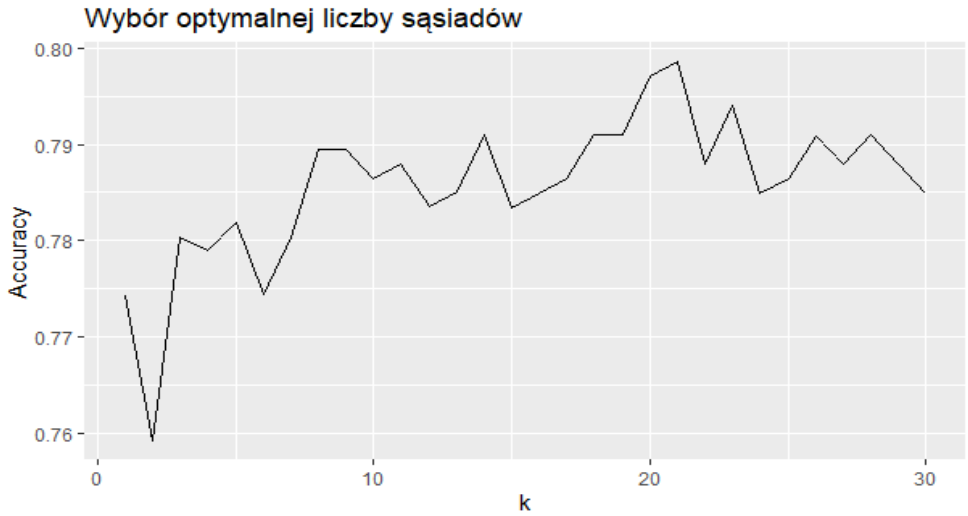
Dopasujemy teraz model kNN, dobierając optymalną liczbę sąsiadów przy użyciu 10-fold Cross Validation.

```
> tr_control <- trainControl(method  = "cv", number  = 10)
> # U■ywamy zbioru treningowego
> knn_fit <- train(Survived ~ . -Parch -Fare -Embarked -IsAlone -Title,
+              method     = "knn",
+              tuneGrid   = expand.grid(k = 1:30),
+              trControl  = tr_control,
+              metric     = "Accuracy",
+              data       = train)
```

Narysujemy teraz wykres zależności miary accuracy w zależności od liczby sąsiadów dla algorytmu kNN.

Optymalna liczba sąsiadów wynosi 21, co możemy zauważyć z rysunku nr 1.

```
> best_k <- 21
```



Rysunek 1: Wybór optymalnej liczby sąsiadów dla algorytmu kNN dopasowanego do części treningowej zbioru *titanic.csv*.

Dopasowujemy model kNN z optymalnym k do danych i generujemy macierz pomyłek i liczymy miarę accuracy, używając zbioru testowego.

```
> dir_knn <- knn(train, test, train$Survived, k = best_k)
> table(dir_knn, test$Survived)

dir_knn   0   1
      0 122  15
      1   6  77

> mean(dir_knn == test$Survived)

[1] 0.9045455
```

Accuracy wyniosło 0.91 (w przybliżeniu).

Wniosek 1 *Największą skutecznością cechuje się model kNN z liczbą sąsiadów równą* 21. *Następnie drugim najlepszym modelem okazał się być model QDA, a następnie LDA i regresja logistyczna.*