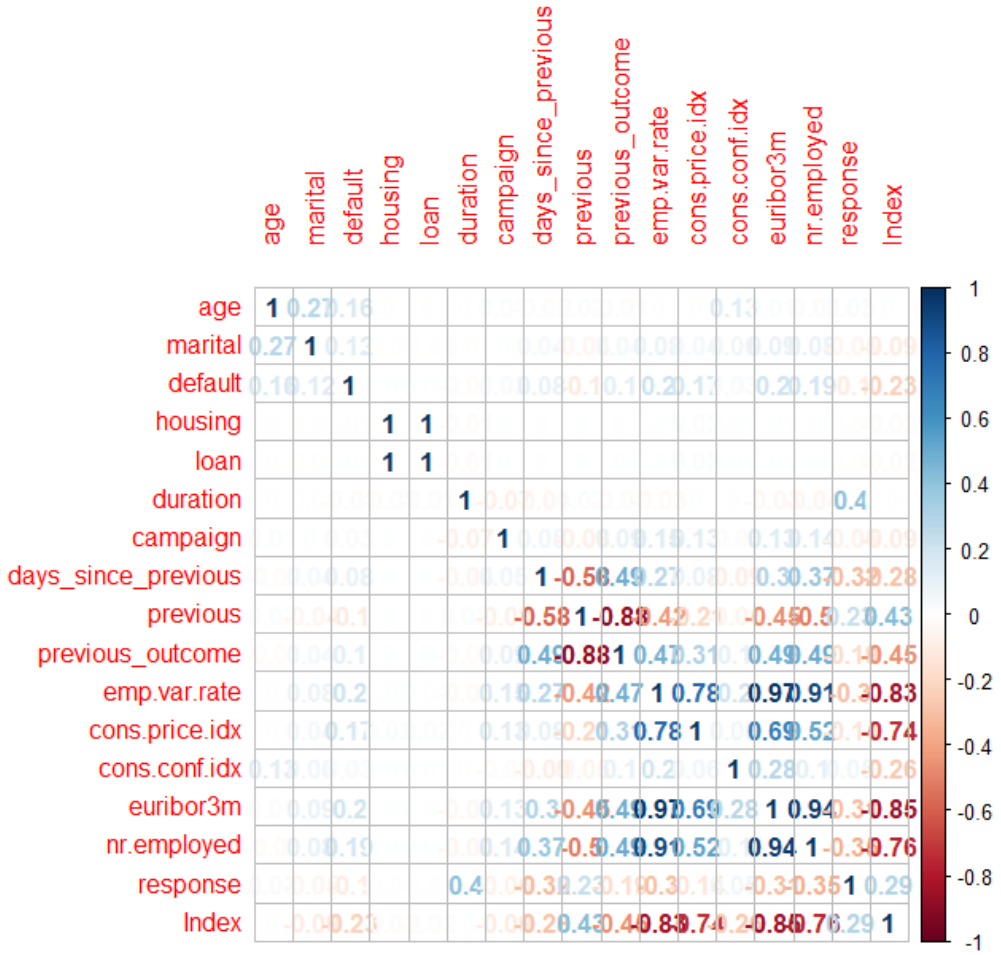
Raport 2A

Sebastian Krzosek 300136

Celem naszego zadania było przewidzenie, czy klienci banku z którymi się kontaktowano podczas kampanii telemarketingu zdecydowali się na założenie lokaty terminowej. Do realizacji tego zadania spośród podanych mi metod wykorzystałem algorytm K najbliższych sąsiadów (K-NN) oraz drzewo C5.0.

Prezentacja rozwiązania:

Zgodnie z poleceniem, na samym początku ustawiłem ziarno generatora liczb pseudolosowych na swój numer indeksu oraz stosując się do Pańskiej rady przygotowałem odpowiednio zmienne łącząc kategorie (marital), bądź przekształcając odpowiedzi odpowiednio z **yes/no/unknown** na **1/0/999**. (default, housing, loan, previous_outcome). Następnie na zmiennych numerycznych dokonałem podziału danych przy pomocy polecenia **createDataPartition** z pakietu **caret** w stosunku 70/30 (1). Kolejnym krokiem było stworzenie macierzy korelacji, dzięki której słusznie mogłem wybrać właściwe predyktory do stworzenia modelu. Jak niżej możemy zaobserwować, zmienne euribor3m i nr.employed są silnie skorelowane z emp.var.rate, oraz zmienna housing z loan, zatem postanowiłem je odrzucić(**euribor3m, nr.employed, housing**). (2)



Budowa modelu klasyfikującego przy pomocy K-NN

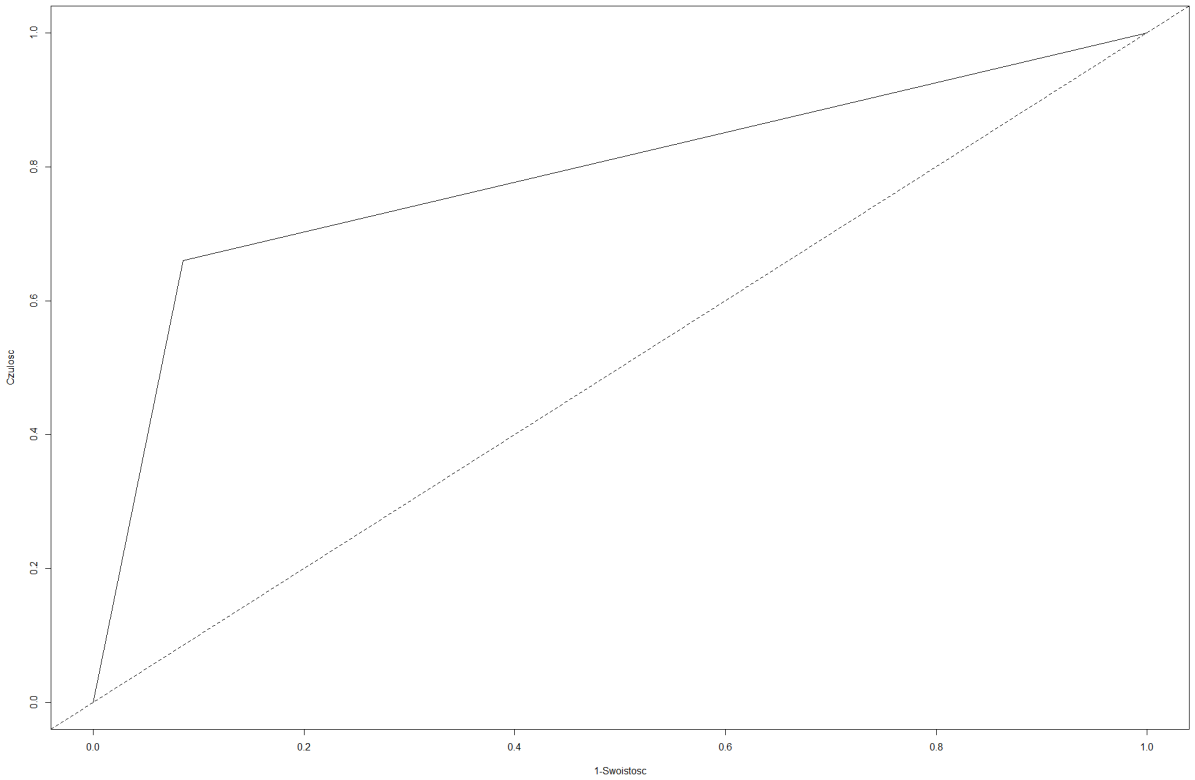
Zmienna celu: response
wykorzystane predyktory: age marital default loan duration campaign days_since_previous previous previous_outcome emp.var.rate cons.price.idx cons.conf.idx

Na samym początku przeprowadziłem standaryzację min-max danych, aby rola wartości (głównie małych) predyktorów nie została pominięta. Właśnie dzięki temu zabiegowi dysproporcja została zniwelowana. Następnie korzystając z wcześniej przygotowanego podziału uruchomiłem algorytm poleceniem **knn** z pakietu **class** zarówno na próbie uczącej jak i testowej dla **k = 29** (do k~ 15 występowało przeuczenie sieci, natomiast dalej idąc metodą prób i błędów dobrałem liczbę sąsiadów przy której miary jakości wychodziły najlepsze). Po zakończeniu algorytmy możemy zaobserwować zestawienie wyniku algorytmu z danymi rzeczywistymi oraz sprawdzić podstawowe miary jakości:

	Dane uczące:	**Dane testowe:**
Czułość:	68.41505 %	65.98465 %
Trafność:	90.67184 %	90.26055 %
Swoistość:	91.76005 %	91.49824 %

W tym miejscu możemy zaobserwować, że dane miary są do siebie dość zbliżone co jest oznaką tego, że nie wystąpiło przeuczenie.

Na samym końcu wykonałem krzywą ROC która prezentuje się następująco:



Algorytm C5.0

Zmienna celu: response
wykorzystane predyktory: age marital default loan duration campaign days_since_previous previous previous_outcome emp.var.rate cons.price.idx cons.conf.idx

Pracę przy tym algorytmie rozpocząłem od obliczenia trafności sposobem naiwnym. Skorzystanie z algorytmu C5.0 powinno zwrócić lepsze wyniki niż te przedstawione tutaj:

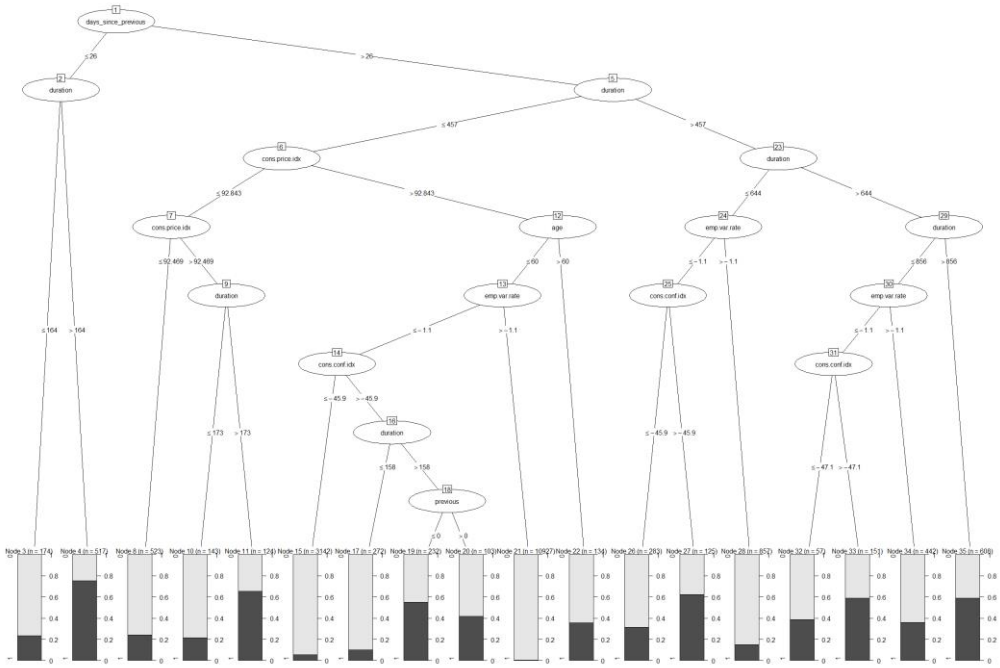
```
> proporcja
   0    1
0.89 0.11
```

Do stworzenia tego modelu wykorzystałem dokładnie te same predyktory co KNN oraz użyłem tego samego podziału danych na zbiory uczące i testowe. Przy użyciu metody **C5.0** z pakietu **C50** po wpisaniu zmiennej celu, predyktorów oraz ustawieniu **minCases** na 50 (tutaj również doszedłem do takiej liczby metodą prób i błędów – dokładnie przy tej opcji otrzymałem najbardziej zadowalające miary jakości). Następnie przy pomocy polecenia summary sprawdziłem, które predyktory okazały się najbardziej znaczące:

```
Attribute usage:

100.00% duration
100.00% days_since_previous
 88.18% emp.var.rate
 82.92% cons.price.idx
 78.72% age
 23.20% cons.conf.idx
  1.78% previous
```

Oraz zbudowałem model drzewa (Załączam je w osobnym pliku). Możemy na nim zaobserwować, że mamy jeden liść czysty, który swoją drogą zawiera w sobie największą liczbę obserwacji (10927). Przy budowaniu drzew R jest dość specyficzny, ponieważ łączy kategorie, aby z każdego węzła wychodziły jedynie 2 galęzie.

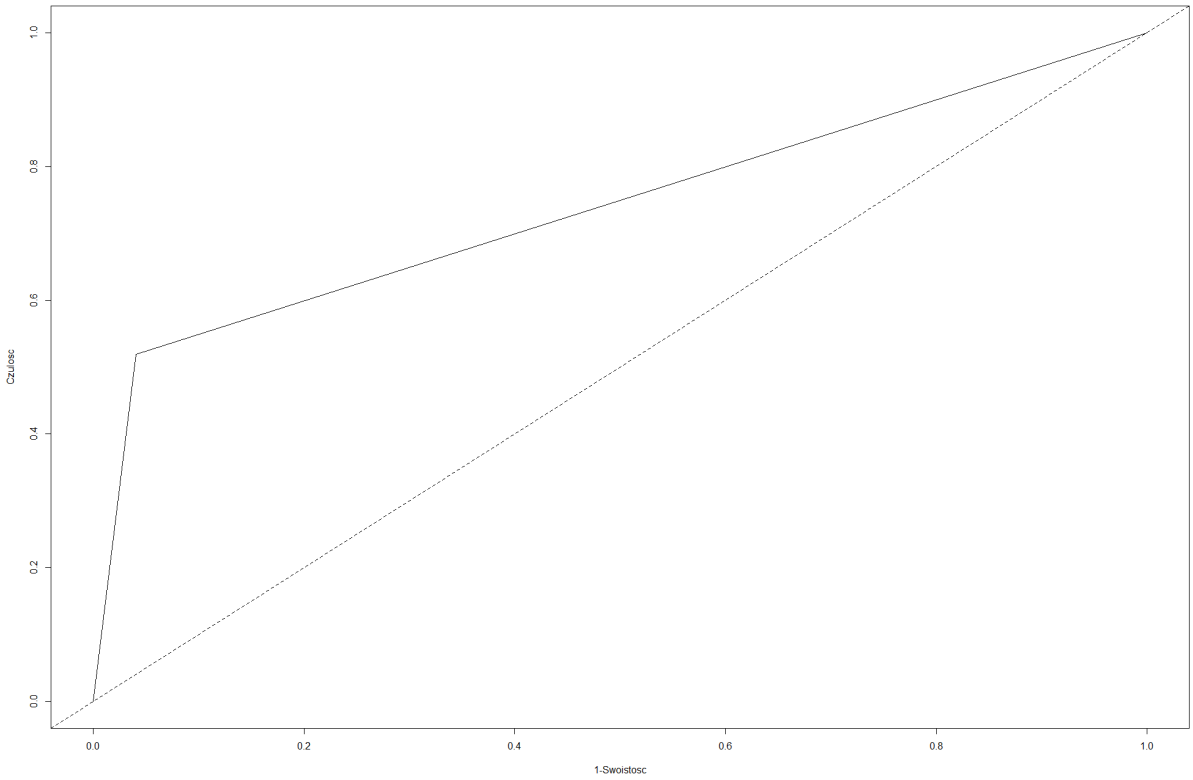


Przy wykorzystaniu algorytmu C5.0 otrzymałem następujące miary jakości:

	Dane uczące:	**Dane testowe:**
Czułość:	54.04235 %	51.86813 %
Trafność:	91.55416 %	90.95533 %
Swoistość:	96.21176 %	95.93007 %

Tutaj również możemy zaobserwować, że nie doszło do przeuczenia, ponieważ miary są do siebie zbliżone.

Dla algorytmu C5.0 również przygotowałem krzywą ROC:



Podsumowanie

Oba modele działają dobrze. Porównując krzywe ROC zbudowane na podstawie obu algorytmów możemy zauważyć zdecydowane różnice. Krzywa stworzona w oparciu o algorytm K-NN pokazuje nam, że dzięki niemu otrzymaliśmy większą czułość, natomiast mniejszą swoistość (wierzchołek znajduje się wyżej i na prawo względem krzywej ROC_C5.0). Moim zdaniem, to właśnie algorytm K Najbliższych Sąsiadów okazał się tym lepszym, ponieważ w przeprowadzanym eksperymencie większe znaczenie miała dla nas czułość niż swoistość, a wskazał on ją lepszą o około 14 punktów procentowych, natomiast duża zaletą drzewa C5.0 była lepsza wizualizacja modelu i wykorzystywanych danych.