RAPORT

Projekt nr 1

Piotr Olesiejuk

3 maja 2019

1 Cel projektu

Celem projektu było skonstruowanie klasyfikatora, który przewidzi, którzy klienci firmy telekomunikacyjnej skorzystają z oferty (klasa 1).

2 Miara oceny klasyfikatora

Miarą przyjętą do oceny jakości klasyfikacji jest precyzja 10%. Sposób jej obliczania jest następujący:

1. Po wytrenowaniu modelu wybiera się 10% największych prawdopodobieństw skorzystania z oferty
2. Zakłada się, że klasyfikator klasyfikuje wszystkie z wybranych rekordów jako te z klasy 1
3. Precyzję oblicza się jako procent poprawnych klasyfikacji elementów z klasy 1

3 Testowane algorytmy klasyfikacyjne

Przetestowano następujące cztery algorytmy klasyfikacyjne:

1. las losowy (randomForest)
2. drzewo klasyfikacyjne (rpart)
3. xgboost
4. kNN

4 Preprocessing

4.1 Wstępny preprocessing

W pierwszej fazie obróbki danych usunięte zostały kolumny z zasadniczą większością wartości NA. Przyjęto próg 95% wartości, jeżeli w kolumnie było więcej niż 95% NA, wtedy była ona usuwana.

Część kolumn numerycznych okazała się być kolumnami o ograniczonej liczbie wartości, zastosowano zatem do tych kolumn przekształcenie, traktując je jako zmienne kategoryczne. W projekcie, jeżeli zmienna

numeryczna miała poniżej 10 poziomów to traktowano ją jako kategoryczną, w innym przypadku pozostawiano ją jako zmienną numeryczną.

Dla zmiennych kategorycznych wartości puste oraz NA zostały zastąpione zmiennymi odpowiednio: "Empty" oraz "Nas".

Po tej fazie zredukowano ilość zmiennych objaśniających z 230 do 80.

4.2 Dalszy preprocessing

W kolejnej fazie została zastosowana indywidualna selekcja zmiennych. Obywała się ona na podstawie ręcznej analizy tabeli generowanej dla każdej zmiennej przez funkcję return_table. Tabelę analizowano pod kątem ilości elementów w poszczególnych kategoriach oraz na podstawie porównania: procenta rekordów w danej kategorii oraz procenta elementów z klasy 1 w danej kategorii spośród wszystkich elementów z klasy 1. Jeżeli te wartości były istotnie różne, uznawano, że zmienna jest istotna. W pozostałych przypadkach zmienna była odrzucana lub zostawiana w zależności od ilości poziomów kategorycznych. W przypadku dużej liczby poziomów była odrzucana. Modyfikacje polegały na redukcji ilości kategorii w istotnych zmiennych. Kategorie bardzo liczne nie były zmieniane, natomiast kategorie o mniejszych licznościach były grupowane do jednej kategorii "Others".

Zostało zmodyfikowanych: 17 zmiennych kategorycznych
Zostało usuniętych: 15 zmiennych kategorycznych
Pozostało bez zmian: 12 zmiennych kategorycznych

4.3 Encoding

Po zredukowaniu danych zmienne kategoryczne, które pozostały miały stosunkowo niewielką liczbę poziomów, ponieważ do 22. Z tego powodu w drzewach oraz w lasach losowych nie zastosowano specjalnego kodowania tych zmiennych. W przypadku xgboost przemianowano je na zmienne numeryczne. W kNN zastosowano one-hot encoding.

4.4 Normalizacja zmiennych numerycznych

Trening modeli odbywał się z wykorzystaniem kroswalidacji dziesięciokrokowej, w której zbiór treningowy był odpowiednio dzielony na część treningową i walidacyjną. Normalizacja odbywała się oddzielnie dla obu części.

Schemat normalizacji:

1. Brakujące zmienne uzupełnij losowo medianą lub średnią uciętą, gdzie poziom ucięcia jest losowany z przedziału (0.01, 0.03)
2. Przeskaluj wartości zmiennych do przedziału (0, 1) wg następującego wzoru: x = (x - min(var)) / (max(var) - min(var))

5 Wyniki

Każdy z wymienionych na początku algorytmów był testowany pod kątem precyzji 10%. Poniżej wymienione zostały najlepsze wyniki dla poszczególnych algorytmów.

Algorytm	Precyzja 10%
randomForest	39,08%*
xgboost	38,92%**
rpart	37,69%**
kNN	—

*Precyzja 10% obliczona jako średnia z wyników 5-krotnie powtórzonej dziesięciokrotnej kroswalidacji
**Precyzja 10% obliczona jako średnia z wyników 10-krotnie powtórzonej dziesięciokrotnej kroswalidacji

W poniższej tabeli opisano parametry, poszczególnych algorytmów, które pozwoliły uzyskać podane wyniki.

Algorytm	Ustawienia
randomForest	ntrees = 60
xgboost	objective = 'binary:logistic', nrounds = 10, eta = 0.04, base_score = 0.92
rpart	cp = 0.01
kNN	—

W projekcie skupiono się na maksymalizacji wyniku z wykorzystaniem xgboost oraz randomForest ze względu na optymistyczne wstępne wyniki.

W przypadku algorytmu randomForest wybór ilości drzew został wybrany na podstawie najlepszego wyniku dla pięciokrotnej kroswalidacji dla modeli o liczbie drzew od 30 do 100 zmienianych co dziesięć. Na tej podstawie wybrany został parametr ntrees = 60, który w tym eksperymencie dał precyzję 39,6%. Następnie wybrany model został przetestowany dziesięciokrotną kroswalidacją powtórzoną 5 razy.

Czas wykonywania algorytmu xgboost jest dużo krótszy niż randomForest, co pozwoliło na większą liczbę dokładniejszych testów. Na początku sprawdzano jak zmienia się precyzja przy zmianie parametru nrounds w dziesięciokrotnej kroswalidacji. Wybrano parametr z największym wynikiem i podobnie optymalizowano kolejno parametry: eta oraz base_score. Ostateczny rezultat, przedstawiony w pierwszej tabeli, jest wynikiem na dziesięciokrotnej kroswalidacji powtórzonej 10 razy.

Jako algorytm do ostatecznej aplikacji na zbiorze testowym rekomendowany jest randomForest. Pierwszym powodem jest najwyższy wynik precyzji, drugim natomiast jest mniejsza wariancja wyników po kroswalidacji niż w algorytmie xgboost. W randomForest wyniosła ona 0,15%, zaś w xgboost 0,22%.

6 Pliki programu

1. functions.R - zawiera wszystkie zdefiniowane funkcje wykorzystane w projekcie
2. preprocessing.R - zawiera kod, który przetwarza dane do postaci, która jest później wykorzystana w dopasowywanych modelach
3. rpart.R, randomForest.R, xgboost.R oraz kNN.R zawierają kody z dopasowaniem poszczególnych metod. Powinny być wywołane po wywołaniu kodu z pliku preprocessing.R