Credit Card Fraud Detection - dokumentacja wstępna projektu (K10)

Zaawansowane uczenie maszynowe

Rębacz Gabriel, Belniak Michał

22 listopada 2020

1. Szczegółowa interpretacja tematu projektu

Wybranym zadaniem jest analiza problemu klasyfikacji na zbiorze danych *Credit Card Fraud Detection*[2] dostępnym w serwisie Kaggle.com. Zbiór zawiera informacje o transakcjach kartami kredytowymi, w tym informacje o kwocie transakcji, czasie transakcji oraz 28 parametrach o nieznanym znaczeniu, które zostały poddane transformacji poprzez analizę głównych składowych.

Zakres projektu obejmuje 4 podstawowe sfery:

- wnikliwą analizę zbioru danych,
- przetworzenie zbioru danych do postaci odpowiednich dla poszczególnych modeli,
- przygotowanie, uczenie i strojenie modeli wybranych spośród dostępnych w języku R,
- ocenę i porównanie modeli wraz z adekwatnymi wnioskami.

Zbiór danych należy przeanalizować pod kątem czynników które mają lub mogą mieć wpływ na sposób ich przygotowania oraz dobór parametrów modeli. Należy także określić sposób porównywania jakości modeli.

2. Opis wykorzystywanych algorytmów

W ramach projektu zostaną wykorzystane trzy modele przedstawione poniżej.

2.1. Drzewo klasyfikacji

Drzewo klasyfikacji [5] to rodzaj drzewa decyzyjnego - narzędzia wspierającego decyzje, które wykorzystuje strukturę drzewiastą jako reprezentację. W węzłach

znajdują się warunki dla wartości atrybutów, które kierują do jednego z możliwych wyników - rozgałęzień. Liście natomiast reprezentują klasę lub prawdopodobieństwo klas. Ścieżka od korzenia do liścia stanowi proces predykcji. O warunku w węźle drzewa decyduje wartość entropii wzajemnej bądź indeksu Giniego dla podziału. Wysokie wartości tych wskaźników oznaczają bardziej równomierny, a więc i korzystny, podział zbioru treningowego między węzły potomne. Zakończenie tworzenia kolejnych rozgałęzień i umieszczenie liścia zachodzi w momencie osiągnięcia kryterium stopu, zarówno wynikającego 'naturalnie' (na przykład brak próbek treningowych lub wszystkie przykłady z jednej klasy) jak i określonego przez implementującego.

W naszym projekcie R wykorzystany zostanie pakiet 'tree'.

2.2. Regresja logistyczna

Regresja logistyczna [4] jest jedną z metod regresji która głównie znajduje zastosowanie jeśli przewidywana klasa może przyjąć tylko dwie wartości. Zgodnie z nazwą, model wykorzystuje funkcję logistyczną jako funkcję określającą wartość predykcji, a klasyfikacja odbywa się dzięki ustalonej wartości progowej, po przekroczeniu której wynikową klasą jest 1, a w przeciwnym przypadku 0.

$$h\Theta(x) = \sigma(Z) = \frac{1}{1 + e^{-Z}}; \tag{1}$$

Argumentem funkcji jest równanie liniowe wartości atrybutów próby postaci:

$$Z = \sum_{i=1}^{n} w_i a_i(x) + w_{n+1} \tag{2}$$

gdzie wartości $w_i, w_2...w_{n+1}$ są modyfikowane w procesie uczenia stanowiąc parametry modelu. W celu optymalizacji tych parametrów definiuje się funkcję kosztu.

$$L = Cost(h\Theta(x), c) = -c\log(h\Theta(x)) - (1-c)\log(1 - h\Theta(x)), \tag{3}$$

Funkcja kosztu jest wykorzystywana do obliczenia jej gradientu w celu zastosowania metody gradientu prostego. Przy metodzie tej parametry zmienia się według poniższej reguły:

$$w_i = w_i - \alpha\frac{\delta L}{\delta w_i} \tag{4}$$

Gdzie α stanowi parametr kroku, który może decydować o szybkości zbieganiа algorytmu do wartości optymalnych lub nawet utrudniać lub uniemożliwiać jej osiągnięcie.

W naszym projekcie R wykorzystana zostanie funkcja **glm** oraz inne z pakietu 'ISLR'.

2.3. Sieć neuronowa

Siecią neuronową [1] nazywany jest aproksymator w postaci modelu grafowego kierunkowego, w którym węzeł, nazywany neuronem, przetwarza wartości przekazywane poprzez wchodzące do niego krawędzie i przekazuje obliczoną wartość do krawędzi wyjściowych. Neurony podzielone są na warstwy, w których żaden neuron nie jest ze sobą połączony, natomiast każdy neuron posiada tyle krawędzi wejściowych, ile jest neuronów w warstwie poprzedzającej i tyle krawędzie wyjściowych ile jest neuronów w warstwie następnej. Krawędzie sieci posiadają wagi, które stanowią parametry w procesie uczenia.

Sposób przetwarzania wartości w neuronie polega na obliczeniu sumy iloczynów wartości na krawędzi wejściowej i odpowiadających im wag, a następnie na obliczeniu wartości funkcji aktywacji, która ustalana jest dla wszystkich neuronów w warstwie. Wyjściem ostatniej warstwy jest aproksymowana wartość przybliżanej funkcji docelowej. Algorytm uczenia sieci opiera się na obliczaniu jakości aproksymacji sieci i wykorzystaniu algorytmu wstecznej propagacji gradientu do zaktualizowania wag sieci.

W naszym projekcie R wykorzystany zostanie pakiet 'neuralnet'.

3. Plan badań

3.1. Cel poszczególnych eksperymentów

Głównym celem będzie analiza porównawcza trzech modeli przy powyższym zadaniu pod kątem jakości predykcji klas i stopnia ich dopasowania do zbioru treningowego. Poszczególne eksperymenty będą miały początkowo na celu strojenie parametrów algorytmów, tak aby porównanie modeli było jak najmniej obarczone niedostosowaniem ich parametrów. Po znalezieniu odpowiednich parametrów porównane zostaną przebiegi i wyniki procesów treningowych na danych utworzonych poprzez *undersampling* oraz *oversampling* dla kolejnych wartości progowych klasyfikacji.

3.2. Charakterystyka zbioru danych

Dane są w postaci tabelarycznej i nie zawierają one pustych wartości. Liczba wszystkich rekordów to 284,807, więc można wnioskować, że zbiór jest odpowiednio liczny. Wszystkie kolumny zawierają wartości numeryczne (parametry V1-V28 oraz czas, wartość transakcji i klasę). Czas i klasa są liczbami całkowitymi, a pozostałe parametry liczbami rzeczywistymi. Zbiór klas jest dwuelementowy - 1 w przypadku oszustwa, a 0 w przypadku poprawnej transakcji. Parametry V1-V28 zostały poddane transformacji poprzez analizę głównych składowych, a ich interpretacja jest nieznana ze względu na poufność danych.

Zbiór jest mocno niezbalansowany - oszustwa stanowią jedynie 0.172% wszystkich transakcji. Konieczne jest przetworzenie zbioru w celu jego zbalansowania. Zostaną zastosowane dwie przeciwstawne metody - *oversampling*, czyli zwiększenie liczby rekordów z klasą 1 oraz *undersampling*, czyli wybranie tylko części

rekordów z klasą 0. Obydwie metody mają na celu stworzenie zestawu danych treningowych z równą liczbą rekordów z klasą 0 oraz 1. Dodatkowo, warto przeprowadzić normalizację wartości atrybutów czasu i kwoty, szczególnie w przypadku sieci neuronowej.

3.3. Parametry algorytmów, których wpływ na wyniki będzie badany

3.3.1. Drzewo klasyfikacji

W przypadku drzewa klasyfikacji głównymi parametrami wpływającymi na wynik modelu są kryterium stopu, sposób i agresywność przycinania oraz wartość progowa klasyfikacji, gdyż w naszym projekcie wartości liści będą reprezentowały prawdopodobieństwa przynależności do klasy 1, aby móc przeprowadzić analizę ROC. Dodatkowo, można zastosować różne techniki poprawiające jakość modelu, takie jak *boosting*. W takim wypadku należało będzie również określić parametry owych algorytmów.

3.3.2. Regresja logistyczna

Wzięty pod uwagę będzie Współczynnik szybkości uczenia, który występuje we wzorze na aktualizację parametrów w_i jako α.

3.3.3. Gęsta sieć neuronowa

Przede wszystkim, właściwości modelu można modyfikować poprzez zmianę rodzaju i liczby warstw ukrytych oraz liczby neuronów w każdej z warstw. Ponadto, wpływ na działanie modelu ma dobór algorytmu optymalizującego średni błąd modelu wraz z jego parametrami, rozmiar pakietu (*batcha*) oraz liczba epok. W naszym zadaniu chcielibyśmy przyjąć stałą liczbę warstw ukrytych, to jest 1, lecz może się okazać, że potrzebna będzie nieco głębsza sieć i w takim przypadku zwiększymy liczbę warstw ukrytych do 2.

Użytym algorytmem optymalizacji będzie stochastyczny najszybszy spadek (*Stochastic Gradient Descent*), w którym jedynym parametrem jest współczynnik szybkości uczenia. Jeśli jednak model będzie wykazywał tendencję do bardzo powolnego spadku wartości funkcji straty zastosujemy algorytm ADAM, który pozwala na określenie trzech parametrów - współczynnik szybkości uczenia oraz β_1 i β_2, które jednak z zasady przyjmują niezmienne wartości odpowiednio 0.9 oraz 0.999. Sprawdzanymi funkcjami aktywacji w warstwie ukrytej będą funkcja logistyczna oraz *rectifier* (neurony typu ReLU). Wagi będą inicjalizowane metodą Xavier[3]. Rozmiar pakietu oraz liczba epok zostaną określone empirycznie.

3.4. Miary jakości i procedury oceny modeli

Głównym sposobem oceny jakości będzie porównanie analiz ROC do zadania wykrywania oszustwa dla poszczególnych modeli, jako jeden z najpopularniej-

szych sposobów oceny jakości predykcji binarnej. Uwzględnione zostaną także miary jakości: precyzja, odzysk oraz współczynnik F.

4. Otwarte kwestie

Do przeanalizowania pozostaje kwestia prawdopodobnego odrzucenia części atrybutów ze zbioru danych. Po zebraniu informacji o zbiorze okazuje się, że część atrybutów jest bardzo słabo skorelowana z wynikową klasą. Ponadto, wymiarowość dziedziny jest bardzo duża, co będzie z pewnością utrudniać osiągnięcie dobrych jakości predykcji, liczymy więc, że uzasadnionym będzie odrzucenie nawet połowy atrybutów. Ponadto, zasadne może okazać się modyfikowanie wybranych algorytmów, jeśli wyniki nie będą satysfakcjonujące i widoczne będzie pole do poprawy.

Bibliografia

[1] missinglink.ai Contributors. *The Complete Guide to Artificial Neural Networks: Concepts and Models*. [Online; accessed 16-November-2020]. URL: `https://missinglink.ai/guides/neural-network-concepts/complete-guide-artificial-neural-networks/`.

[2] *Credit Card Fraud Detection*. Dostęp zdalny (15.11.2020): `https://www.kaggle.com/mlg-ulb/creditcardfraud`. 2016.

[3] X. Glorot i Y. Bengio. „Understanding the difficulty of training deep feedforward neural networks". W: *Thirteenth International Conference on Artificial Intelligence and Statistics*. 2010.

[4] Saishruthi Swaminathan. *Logistic Regression — Detailed Overview*. [Online; accessed 16-November-2020]. 2018. URL: `https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc`.

[5] Wikipedia contributors. *Decision tree — Wikipedia, The Free Encyclopedia*. [Online; accessed 16-November-2020]. 2020. URL: `https://en.wikipedia.org/w/index.php?title=Decision_tree&oldid=983253586`.