Detekcja oszustw z wykorzystaniem metod wrażliwych na koszt

Patryk Wielopolski

16 grudnia 2019

Rozdział 1

Wstęp

- Rosnąca ilość transakcji kartami kredytowymi - Rosnący poziom fraudów na świecie

- Flow transakcji oraz umiejscowienie systemu do detekcji fraudów - Aktualna sytuacja if-then + predicitve models

If-then: - więcej niż 4 wypłaty z bankomatu w ciągu godziny - więcej niż 2 transakcje w ciągu 5 minut - transakcja w sklepie karta a następna zaraz w internecie

Jeżeli więcej niż 1 reguła jest spełniona to odmowa transakcji. Pojawiają się problemy z niewykrywaniem nowych reguł oraz możliwe jest tworzenie tylko prostych reguł. Z drugiej strony są one proste do implementacji oraz interpretacji.

Rozdział 2

Wprowadzenie teoretyczne

W tej części zostaną wprowadzone wszelkie potrzebne miary skuteczności modeli oraz modele predykcyjne, które zostaną wykorzystane do przeprowadzenia eksperymentu.

Modele predykcyjne zostały podzielone na dwie kategorie: standardowe oraz wrażliwe na koszt. Pierwsze z nich są powszechnie wykorzystywane w standardowych aplikacjach. Drugie z nich dzielą się na dwie podkategorie - *Cost Sensitive Training* oraz *Cost Sensitive Classification*.

2.1 Miary skuteczności modeli

2.1.1 Macierz pomyłek

W tej sekcji zdefiniujemy macierz pomyłek.

		Predykcja	
		Oszustwo	Normalna
Prawda	Oszustwo	TP	FN
	Normalna	FP	TN

Tabela 2.1: Macierz pomyłek

Na podstawie podanej macierzy pomyłek w tabeli 2.1 definiujemy następujące miary skuteczności modeli:

$$\text{Skuteczność} = \frac{TP + TN}{TP + FP + FN + TN}$$

$$\text{Precyzja} = \frac{TP}{TP + FP}$$

$$\text{Czułość} = \frac{TP}{TP + FN}$$

$$\text{F1 Score} = 2 \cdot \frac{\text{Precyzja} \cdot \text{Czułość}}{\text{Precyzja} + \text{Czułość}}$$

2.1.2 Miary skuteczności modeli wrażliwe na koszt

Motywacją do powstania miar wrażliwych na koszt jest rzeczywista ewaluacja modeli. Podstawowe metryki nie uwzględniają różnicy w kosztach pomyłki dla fp i fn. Ponadto koszt fraudów znacząco różni się w zależności od przypadku.

W celu wprowadzenia potrzebnych metryk potrzebujemy wprowadzić tzw. macierz kosztu, która jest zaprezentowana w tabeli 2.2, gdzie poszczególne komórki oznacza odpowiadające wartość kosztu predykcji. Następnie definiujemy

		Predykcja	
		Oszustwo	Normalna
Prawda	Oszustwo	C_{TP_i}	C_{FN_i}
	Normalna	C_{FP_i}	C_{TN_i}

Tabela 2.2: Macierz kosztu

następującą wartość:

$$\text{Koszt}(f(\boldsymbol{x}_i^*)) = y_i(c_i C_{TP_i} + (1 - c_i)C_{FN_i}) + (1 - y_i)(c_i C_{FP_i} + (1 - c_i)C_{TN_i}),$$

gdzie

- $\boldsymbol{x}_i^* = [\boldsymbol{x}_i, C_{TP_i}, C_{FP_i}, C_{FN_i}, C_{TN_i}]$ - wektor zawierający wartości cech i-tej obserwacji rozszerzony o koszt klasyfikacji
- $C.$ - koszt klasyfikacji i-tej obserwacji
- $f(\cdot)$ - model predykcyjny
- y_i - prawdziwa wartość i-tej obserwacji
- c_i - predykcja dla i-tej obserwacji

Następnie wprowadzamy następujące miary skuteczności modeli:

$$\text{Koszt całkowity}(f(\boldsymbol{S})) = \sum_{i=1}^{N} \text{Cost}(f(\boldsymbol{x}_i^*)) \tag{2.1}$$

$$\text{Oszczędności} = \frac{\text{Koszt}_l(\boldsymbol{S}) - \text{Koszt}(f(\boldsymbol{S}))}{\text{Koszt}_l(\boldsymbol{S})} \tag{2.2}$$

gdzie

- $\boldsymbol{S}$ - data set
- $\text{Koszt}_l = min\{\text{Cost}(f_0(\boldsymbol{S}), \text{Koszt}(f_1(\boldsymbol{S})\}$
- $f_a(\boldsymbol{S}) = \boldsymbol{a}$ gdzie $a \in \{0, 1\}$

Wartość $f_a(\boldsymbol{S})$ możemy rozumieć jako przypadek naiwnego modelu, który wszystkim obserwacjom przyznaje wartość a. Natomiast Koszt_l oznacza wybranie naiwnego klasyfikatora, który generuje mniejsze koszty. Zatem ostatecznie Oszczędności możemy rozumieć jako procentową wartość o ile testowany model jest lepszy od naiwnego klasyfikatora.

2.2 Standardowe modele

With judicious choices for yi, we may express a variety of tasks, such as regression, classification, and ranking. The task of training the model amounts to finding the best parameters that best fit the training data xi and labels yi

. In order to train the model, we need to define the objective function to measure how well the model fit the training data.

A salient characteristic of objective functions is that they consist two parts: training loss and regularization term:

where L is the training loss function, and is the regularization term. The training loss measures how predictive our model is with respect to the training data. A common choice of L is the mean squared error, which is given by

2.2.1 Regresja logistyczna

Regresja logistyczna należy do jednego z najbardziej podstawowych modeli statystycznych używanych do problemów klasyfikacyjnych.

$$\hat{p} = P(y = 1|\boldsymbol{x_i}) = h_\theta(\boldsymbol{x_i}) = g\left(\sum_{j=1}^{k} \theta^{(j)} x_i^{(j)}\right) \tag{2.3}$$

Standardowa funkcja straty przyjmuje następującą postać:

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{N} J_i(\theta)$$

gdzie funkcja $g(z)$ jest funkcją łączącą typu *logit* i przyjmuje postać

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

Natomiast

$$J_i(\theta) = -y_i log(h_\theta(\boldsymbol{x_i})) - (1 - y_i) log(1 - h_\theta(\boldsymbol{x_i})) \tag{2.4}$$

to standardowa entropia krzyżowa.

Standard costs:

$$J_i(\theta) \approx \begin{cases} 0, & \text{if } y_i \approx h_\theta(\boldsymbol{x_i}), \\ \infty, & \text{if } y_i \approx (1 - h_\theta(\boldsymbol{x_i})). \end{cases}$$

Thus

$$C_{TP_i} = C_{TN_i} \approx 0$$

$$C_{FP_i} = C_{FN_i} \approx \infty$$

Wytrenowany, aby minimalizować błąd klasyfikacji, a ewaluowany na metryce kosztu.

2.2.2 Drzewo decyzyjne

Drzewo klasyfikacyjne to przykład jednego z rodzaju drzew decyzyjnych, którego celem jest znalezienie najlepszego rozróżnienia pomiędzy klasami. W ogólności drzewo decyzyjne składa się z zestawu reguł.

Drzewo składa się z węzłów, które są reprezentowane przez parę $(\boldsymbol{x}^j, l^j)$, która oznacza podział zbioru obserwacji $\mathcal{S}$ na dwa zbiory: $\mathcal{S}^l$ oraz $\mathcal{S}^r$ względem wektora obserwacji $\boldsymbol{x}$ oraz progu decyzyjnego ^{j} w następujący sposób:

$$\mathcal{S}^l = \{\boldsymbol{x}^* : \boldsymbol{x}^* \in \mathcal{S} \wedge x_i^j \leqslant l^j\},$$

$$\mathcal{S}^r = \{\boldsymbol{x}^* : \boldsymbol{x}^* \in \mathcal{S} \wedge x_i^j > l^j\},$$

gdzie $\boldsymbol{x}^j$ reprezentuje j-ty atrybut wektora $\boldsymbol{x}$. Ponadto ^{j} jest wartością taką, że $\min \boldsymbol{x}^j \leqslant l^j < \max \boldsymbol{x}^j$. Ponadto warto zauważyć, że $\mathcal{S}^l \cup \mathcal{S}^r = \mathcal{S}$, co oznacza, że nasz podział rozdziela wektor obserwacji na dokładnie dwa rozłączne zbiory. Po znalezieniu optymalnego podziału zliczamy ilość pozytywnych próbek:

$$\mathcal{S}_1 = \{\boldsymbol{x}^* : \boldsymbol{x}^* \in \mathcal{S} \wedge y_i = 1\},$$

a następnie zliczamy procent pozytywnych próbek jako:

$$\pi_1 = \frac{|\mathcal{S}_1|}{|\mathcal{S}|}.$$

Następnie dla każdego z liści jest obliczana wielkość jego zanieczyszczenia.

- Misclassification: $I_m(\pi_1) = 1 - \max(\pi_1, 1 - \pi_1)$
- Entropy: $I_e(\pi_1) = -\pi_1 \log(\pi_1) - (1 - \pi_1) \log(1 - \pi_1)$
- Gini: $I_g(\pi_1) = 2\pi_1(1 - \pi_1)$

Następnie dla każdego proponowanego podziału dla danej reguły $(\boldsymbol{x}^j, l^j)$ liczony jest przyrost czystości w następujący sposób:

$$\text{Gain}(\boldsymbol{x}^j, l^j) = I(\pi_1) - \frac{|\mathcal{S}^l|}{|\mathcal{S}|} I(\pi_1^l) - \frac{|\mathcal{S}^r|}{|\mathcal{S}|} I(\pi_1^r),$$

gdzie $I(\pi_1)$ może być dowolną z zaproponowanych miar zanieczyszczenia. Ostatecznie wybiera się ten podział, który maksymalizuje przyrost czystości, a następnie dzieli się zbiór $\mathcal{S}$ na podzbiory $\mathcal{S}^l$ i $\mathcal{S}^r$. W taki sposób jest pokonywany pojedynczy podział zbioru dla konkretnego węzła. Całe drzewo tworzy się poprzez kolejne podziały węzłów aż do momentu dotarcia przez algorytm do kryterium stopu.

2.2.3 Las losowy

Kolejne modele są przedstawicielami szerokiej klasy metod *ensemble*, których celem jest złożenie predykcji wielu klasyfikatorów bazowych, aby poprawić generalizację pojedynczego estymatora. Wśród nich wyróżniamy dwie główne kategorie. Pierwsza z nich to metody uśredniania, które polegają na zbudowaniu wielu niezależnych klasyfikatorów, a następnie uśrednianie wyników predykcji. Przedstawicielem tej kategorii jest las losowy. Natomiast druga polega na iteracyjnym budowaniu kolejnych modeli, które próbują zredukować obciążenie poprzednika. Bardzo powszechnie znanym reprezentantem jest algorytm XGBoost, którym zajmiemy się w następnym podrozdziale.

Metody *ensemble* wykorzystują metody próbkowania w celu utworzenia różnych klasyfikatorów bazowych, aby następnie dokonać ostatecznej predykcji. Metody te są używane, aby zredukować wariancję klasyfikatora bazowego (zazwyczaj drzewa decyzyjnego) poprzez losowe dobieranie próbki i/lub zmiennych, na których model będzie uczony. W wielu przypadkach stworzenie modelu opartego o *bagging* jest znacznie prostsze, ponieważ wymaga jedynie zmiany próbkowania danych bez naruszania modelu bazowego, natomiast metody typu boosting wymagają zmiany całego algorytmu. Z licznych obserwacji wynika, że pierwsza z metod dużo lepiej radzi sobie mają jako bazowe klasyfikatory złożone modele, w przeciwieństwie do drugiej, która zazwyczaj najlepiej performuje wykorzystując proste modele (np. płytkie drzewa decyzyjne, tzn. o małej głębokości).

Przykładowe metody losowania próbek do modeli bazowych:

- *Pasting* - losowanie obserwacji bez powtórzeń
- *Bagging* - losowanie obserwacji z powtórzeniami
- *Random Subspaces* - losowanie podzbioru zmiennych
- *Random Patches* - losowanie podzbioru zmiennych oraz obserwacji

W przypadku lasu losowego proces losowania próbek jest podzielony na dwie fazy. Pierwsza z nich polega na próbkowaniu z powtórzeniami obserwacji ze

zbioru treningowego dla każdego z osobnych estymatorów bazowych. Następnie podczas fazy tworzenia kolejnych węzłów w drzewach wybierany jest losowy podzbiór zmiennych, które mogą być w tym kroku wykorzystane.

Celem tych dwóch różnych źródeł losowości jest redukcja wariancji lasu. Pojedyncze drzewa mają tendencję do zbytniego dopasowywania się do danych, zatem losowanie poszczególnych zmiennych w każdym kolejnym węźle pomaga to zredukować. Natomiast losowanie różnych próbek do każdego z klasyfikatorów pozwala na stworzenie lekko odmiennych modeli bazowych.

2.2.4 XGBoost

Tak jak wspomnieliśmy w poprzednim rozdziale algorytm XGBoost jest przykładem klasyfikatora, który w iteracyjny sposób tworzy kolejne bazowe klasyfikatory. W przypadku tego algorytmu jako klasyfikator bazowy wykorzystujemy implementację drzew decyzyjnych typu CART, które nieznacznie różnią się od standardowych drzew decyzyjnych opisywanych w 2.2.2, ponieważ liść drzewa jest rozszerzony o wartość rzeczywistą, która reprezentuje decyzję modelu. Ponieważ jest to złożenie wielu klasyfikatorów, to możemy zapisać nasz model w następującej postaci:

$$\hat{y_i} = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F},$$

gdzie K oznacza liczbę drzew, f funkcje z przestrzeni $\mathcal{F}$ wszystkich możliwych drzew CART. Zatem funkcją, którą będziemy optymalizować przyjmuje następującą postać:

$$\text{Obj}(\theta) = \sum_{i=1}^{n} l(y_i, y_i^{(t)}) + \sum_{k=1}^{K} \Omega(f_k) \tag{2.5}$$

W przypadku klasyfikacji przyjmujemy tak samo jak poprzednio funkcję entropii krzyżowej (2.4). Patrząc na postać modelu nie różni się ona niczym od lasu losowego. Zatem różnica między tymi modelami polega na sposobie trenowania drzew, który pokrótce opiszemy.

Ponieważ naszym modelem bazowym jest drzewo, to nie jesteśmy w stanie wprost rozwiązać zagadnienia optymalizacyjnego poprzez obliczenie gradientu funkcji i iteracyjne znalezienie rozwiązania. W tym przypadku posłużymy się treningiem addytywnym, który polega na iteracyjnym poprawianiem błędów z poprzednich modeli poprzez odpowiednie przydzielanie wag próbkom w kolejnych krokach algorytmu. Oznaczmy wartość predykcji w kroku t jako $y_i^{(t)}$.

$$y_i^{(0)} = 0$$

$$y_i^{(1)} = f_1(x_i) = y_i^{(0)} + f_1(x_i)$$

$$y_i^{(2)} = f_1(x_i) + f_2(x_i) = y_i^{(1)} + f_2(x_i)$$

...

$$y_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = y_i^{(t-1)} + f_t(x_i)$$

Pozostaje zagadnienie jakie drzewo chcemy wybrać w każdym z kroków. Oczywiście takie, które optymalizuje naszą funkcję Obj. Korzystając z powyższych wzorów oraz 2.5 otrzymujemy następującą postać tej funkcji:

$$\sum_{i=1}^{n} l(y_i, y_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + \text{constant} \ .$$

Korzystając z rozwinięcia szeregu Taylora dla funkcji straty otrzymujemy ogólny wzór:

$$\text{Obj}^{(t)} = \sum_{i=1}^{n} [l(y_i, y_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) + \text{constant} \ ,$$

gdzie

$$g_i = \partial_{y_i^{(t-1)}} l(y_i, y_i^{(t-1)})$$

$$h_i = \partial^2_{y_i^{(t-1)}} l(y_i, y_i^{(t-1)})$$

Zatem po redukcji stałych, które są nieistotne z punktu widzenia optymalizacji otrzymujemy:

$$\sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)] + \Omega(f_t)$$

2.3 Cost Dependent Classification

2.3.1 Optymalizacja progu

2.3.2 Bayesian Minimum Risk

Risk associated with predictions:

$$R(p_f|x) = L(p_f|y_f)P(p_f|x) + L(p_f|y_l)P(y_l|x)$$

$$R(p_l|x) = L(p_l|y_l)P(p_l|x) + L(p_l|y_f)P(y_f|x)$$

Classification threshold:

$$R(p_f|x) \leqslant R(p_l|x)$$

Where:

- $P(p_f|x)$, $P(p_l|x)$ - estimated probability of fraud/legimate transaction
- $L(p_i|y_j)$ and $i, j \in \{l, f\}$ - loss function

Exact formula:

$$P(p_f|x) \geqslant \frac{L(p_f|y_l) - L(p_l|y_l)}{L(p_l|y_f) - L(p_f|y_f) - L(p_l|y_l) + L(p_f|y_l)}$$

After reformulation:

$$p \geqslant \frac{C_{FP} - C_{TN}}{C_{FN} - C_{TP} - C_{TN} + C_{FP}}$$

2.4 Cost Sensitive Training

Pierwszą podgrupą metod wrażliwych na koszt jest *Cost Sensitive Trainig*. Są to metody, które

2.4.1 Regresja logistyczna wrażliwa na koszt

Actual costs:

$$J_i^c(\theta) = \begin{cases} C_{TP_i}, & \text{if } y_i = 1 \text{ and } h_\theta(\boldsymbol{x_i}) \approx 1, \\ C_{TN_i}, & \text{if } y_i = 0 \text{ and } h_\theta(\boldsymbol{x_i}) \approx 0, \\ C_{FP_i}, & \text{if } y_i = 0 \text{ and } h_\theta(\boldsymbol{x_i}) \approx 1, \\ C_{FN_i}, & \text{if } y_i = 1 \text{ and } h_\theta(\boldsymbol{x_i}) \approx 0. \end{cases}$$

Cost sensitive loss function:

$$J^c(\theta) = \frac{1}{N}\sum_{i=1}^{N}\Big(y_i\Big(h_\theta(\boldsymbol{x_i})C_{TP_i} + (1 - h_\theta(\boldsymbol{x_i}))C_{FN_i}\Big) + (1 - y_i)\Big(h_\theta(\boldsymbol{x_i})C_{FP_i} + (1 - h_\theta(\boldsymbol{x_i}))C_{TN_i}\Big)\Big)$$

2.4.2 Drzewo decyzyjne wrażliwe na koszt

Cost Sensitive impurity measure: $I_c(\mathcal{S}) = min\left\{Cost(f_0(\mathcal{S})), Cost(f_1(\mathcal{S}))\right\}$

$$f(\mathcal{S}) = \begin{cases} 0, & \text{jeżeli Cost}(f_0(\mathcal{S})) \leqslant \text{Cost}(f_1(\mathcal{S})), \\ 1, & \text{w przeciwnym wypadku.} \end{cases}$$

Rozdział 3

Eksperyment

Celem eksperymentu jest zbadanie jaki wpływ mają na miarę F1 oraz oszczędności mają poszczególne algorytmy.

Do eksperymentu zostanie wykorzystany zbiór danych Credit Card Fraud Detection zawierający 284,807 transakcji w tym zaledwie 492 oszustw. Tabela składa się z 30 kolumn, w tym 28 z nich są to nienazwane, zanonimizowane zmienne, które były wcześniej poddane transformacji PCA (*ang. Principal Component Analysis*), dodatkowo posiadamy informacje dot. czasu transakcji oraz kwoty.

Mimo tego, że dane są zanonimizowane można mieć pewne intuicje na temat tego jakie zmienne zostały użyte z zbiorze danych. Raw data: - ID klienta, data transakcji, kwota, lokalizacja, typ transakcji (internet, płatność w sklepie, wypłata z bankomatu), rodzaj transakcji (linie lotnicze, hotel, wypożyczalnia samochodów), fraud, wiek klienta, kraj zamieszkania, kod pocztowy, typ karty Na podstawie ref zmienne, które wykorzystuje się do tego typu problemów to: - agregaty czasowe, np. ilość transakcji dla tego samego klienta w ciągu ostatnich 6 godzin, suma transakcji z ostatnich 7 dni itp. W ogólności są to agregaty klient/karta kredytowa x typ transakcji/sklep/kategorai sklepu/kraj x ostatnie godziny/dni/tygodnie/miesiące x ilość/średnia/suma

Rozkład kwoty...

Eksperyment został przeprowadzony w następujący sposób: 50-krotnie dzielimy zbiór danych w proporcjach 50:17:33 na zbiór treningowy, walidacyjny oraz testowy. Następnie uczymy wszystkie modele na zbiorze treningowym. Dla modelu XGBoost wykorzystujemy zbiór walidacyjny do procesu wczesnego zatrzymywania (*ang. Early stopping*), natomiast dla modeli BMR oraz TO korzystamy z tego zbioru jako zbiór treningowy. Następnie dla wszystkich modeli dokonujemy predykcyji na zbiorze testowym i mierzymy skuteczność typowań.

Rozdział 4

Rezultaty

Rozdział 5

Podsumowanie