UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

VYUŽITIE ŠTATISTICKÉHO MODELOVANIA PRE PREDIKCIU SPRÁVANIA HRÁČOV MOBILNÝCH HIER

DIPLOMOVÁ PRÁCA

2020 Bc. Pavol Lisý

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

VYUŽITIE ŠTATISTICKÉHO MODELOVANIA PRE PREDIKCIU SPRÁVANIA HRÁČOV MOBILNÝCH HIER

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomicko-finančná matematika a modelovanie
Študijný odbor:	1114 Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	Mgr. Viktor Gregor

Bratislava 2020 Bc. Pavol Lisý

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Pavol Lisý
Študijný program: ekonomicko-finančná matematika a modelovanie (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Využitie štatistického modelovania pre predikciu správania hráčov mobilných hier
Application of statistical models for prediction of player behaviour in mobile games

Anotácia: Diplomová práca zahŕňa zadefinovanie problému, ktorý sa bude modelovať. Ďalšou časťou je výber vhodného algoritmu a vývoj prototypu modelu. V práci kladieme dôraz na možnosť aplikácie modelu v praxi.

Vedúci: Mgr. Viktor Gregor
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Marek Fila, DrSc.

Dátum zadania: 08.01.2019

Dátum schválenia: 08.01.2019 prof. RNDr. Daniel Ševčovič, DrSc.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie Touto formou sa chcem poďakovať vedúcemu mojej diplomovej práce Mgr. Viktorovi Gregorovi za ochotu, pomoc, rady a pripomienky, ktoré mi pomohli pri písaní tejto diplomovej práce. Tiež by som sa chcel poďakovať svojej manželke, ktorá ma podporovala a motivovala.

Abstrakt

LISÝ, Pavol: Využitie štatistického modelovania pre predikciu správania hráčov mobilných hier [Diplomová práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školitel': Mgr. Viktor Gregor, Bratislava, 2020, 61 s.

V tejto diplomovej práci spolupracujeme so slovenskou firmou Pixel Federation, s.r.o., ktorá je tvorcom mobilných hier. Pre firmu je z rôznych dôvodov dôležité vedieť predpovedať správanie sa hráčov v hre. Predikciu správania sa hráčov mobilných hier modelujeme pomocou metód štatistického modelovania, konkrétne pomocou strojového učenia konštruujeme vhodný klasifikátor. V prvej kapitole sa venujeme opisu teórie k použitým metódam. V druhej kapitole opisujeme dáta s ktorými sme pracovali a definujeme metodiku vyhodnocovania úspešnosti predikcie. Cieľom tejto práce je predovšetkým nájsť vhodný aparát, ktorý vie čo najpresnejšie predikovať správanie sa hráčov mobilných hier. V poslednej časti druhej kapitoly sa venujeme porovnaniu rôznych metód predikcie, pričom dôraz kladieme hlavne na dosiahnutú úspešnosť jednotlivých metód pri predpovedi správania sa hráčov.

Kľúčové slová: Predikcia. Klasifikátor. Klasifikačný strom. Logistická regresia. Bayesov naivný klasifikátor. Metóda oporného bodu. Hráči mobilných hier.

Abstract

LISÝ, Pavol: Application of statistical models for prediction of player behaviour in mobile games [Master thesis], Comenius university in Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics; Supervisor: Mgr. Viktor Gregor, Bratislava, 2020, 61 p.

In this master thesis we are cooperating with a Slovak company Pixel Federation s.r.o., which is a creator of mobile games. For various reasons, it is important for the company to be able to predict the behavior of players in the game. We model the prediction of mobile game players' behavior using statistical modeling methods, specifically using machine learning to construct a suitable classifier. In the first chapter we describe the theory of the methods used. In the second chapter we describe the data we worked with and define the methodology for evaluating the success of the prediction. The aim of this work is primarily to find a suitable apparatus that can accurately predict the behavior of mobile game players. In the last part of the second chapter, we focus on the comparison of different methods of prediction, with emphasis mainly on the achieved success of individual methods in predicting player behaviour.

Keywords: Prediction. Classifier. Classification trees. Logistic regression. Naive Bayes classifier. Support vector machine. Mobile game players.

Obsah

Úvod

Pixel Federation je slovenská firma, ktorá sa zaoberá vývojom mobilných, či počítačových hier. Zisk firmy tvorí v drvivej väčšine príjem z hier, pričom používa takzvaný Freemium model. V praxi to znamená že hráč si môže hru stiahnuť a hrať zadarmo. Ak sa hráčovi hra páči, prípadne chce mať výhodu oproti iným hráčom, alebo chce hru vyhrať rýchlejšie, prípadne iný dôvod má možnosť zakúpiť si za peniaze rôzne vylepšenia v hre.

Firma výšku svojho budúceho zisku nepozná a veľmi záleží na tom aké množstvo peňazí sú hráči ochotní do hry investovať. Pre firmu je z viacerých dôvodov dôležité predikovať koľko a ktorí hráči v budúcnosti zaplatia. Napríklad z hľadiska investícií, či už do nových hier alebo vylepšovania starších hier je dôležité poznať aspoň približne výšku budúceho príjmu. V závislosti od predpokladaného budúceho príjmu sa môže firma v danom momente lepšie rozhodnúť. Okrem toho je dôležité predikovať počet budúcich platičov kvôli vyhodnocovaniu marketingových kampaní. Firma sa musí neustále rozhodovať, či marketingovú kampaň zastaví, alebo v nej bude pokračovať. V tomto rozhodovaní je predpokladaný budúci počet platiacich hráčov z kampane kľúčový. Z hľadiska maximalizácie zisku je pre firmu dobré vedieť aj ktorí hráči majú potenciál v budúcnosti zaplatiť, aby firma mala informáciu, ktorým hráčom sa má v prípade problémov prednostne venovať. Budúce správanie sa hráčov mobilných hier sa najlepšie predpovedá pomocou štatistického modelovania.

V súčasnosti je strojové učenie, ktoré spadá pod štatistické modelovanie, účinnou metódou pri riešení viacerých matematických problémov. Jeho výhodou je, že dokáže pracovať s veľkým množstvom dát, pričom s pojmom veľké dáta sa v súčasnosti stretávame čoraz viac. Myslím si, že strojové učenie bude v budúcnosti využívané v čím ďalej tým viac odvetviach priemyslu, ale aj každodenného života. Strojové učenie je pre mňa fascinujúce, a aj preto som si vybral túto tému diplomovej práce.

Cieľom tejto diplomovej práce je, ako sme už spomínali, čo najpresnejšie predikovať koľko a ktorí hráči v budúcnosti zaplatia. V prvej kapitole sa zaoberáme teoretickým opisom klasifikácie a jej metódami ktoré v tejto práci sú : Logistická regresia, Klasifikačné stromy, Bayesov klasifikátor a Metóda oporného bodu. Okrem toho v tejto časti opisujeme vlastnosti klasifikátorov a spôsob určovania dôležitosti predikčných premenných. V druhej

kapitole definujeme klasifikačnú úlohu a spôsob vyhodnocovania úspešnosti klasifikátorov. Ďalej sa venujeme opisu dát a opisu spôsobu ako s nimi budeme pracovať. V zvyšnej časti druhej kapitoly sa venujeme predikcii. Zaoberáme sa binárnou klasifikačnou úlohou, a to tak že sa snažíme predpovedať, či daný hráč v budúcnosti zaplatí aspoň raz alebo nezaplatí vôbec. V závere praktickej časti medzi sebou porovnávame jednotlivé metódy a ich úspešnosť pri klasifikácii podľa nami zvolených kritérií.

1 Teoretická časť

1.1 Klasifikátor

Klasifikátor alebo klasifikačné pravidlo je funkcia c (x) definovaná na X tak, že pre každé x, c(x) sa rovná jednému z čísel 1, 2, ..., Y. Ďalším spôsobom, ako sa pozerať na klasifikátor, je definovať A_j ako podmnožinu X, na ktorej c (x) = y; to znamená,

$A_j = \{x; c(x) = y\}$. Množiny A_1, ..., A_y sú disjunktné a $X = \bigcup_y A_y$. Teda A_y tvorí rozdelenie X.

Klasifikátory nie sú konštruované rozmarne. Sú založené na skúsenostiach zminulosti. Lekári napríklad vedia, že starší pacienti so srdcovým infarktom s nízkym krvným tlakom sú vo všeobecnosti vysoko rizikoví. Pri systematickej konštrukcii klasifikátorov sú minulé skúsenosti zhrnuté v takzvanej učebnej (tréningovej) vzorke. Pozostáva z nameraných údajov o N prípadoch pozorovaných v minulosti a ich skutočnej klasifikácie.

Predpokladáme, že konštrukcia klasifikátora je založená na tréningovej vzorke, pričom tréningové dáta pozostávajú z údajov (x_1, y_1), ..., (x_N, y_N) o N prípadoch, kde $x_n \in X$ a $y_n \in \{1, ..., Y\}$, $n = 1, ..., N$ Vzorka vzdelávania je označená L; $L = \{(x_1, y_1) ..., (x_N, y_N)\}$. Rozlišujeme dva všeobecné typy premenných, ktoré sa môžu objaviť vo vektore prediktorov. Premenná sa nazýva usporiadaná alebo numerická, ak sú jej namerané hodnoty reálne čísla. Premenná je kategorická, ak berie hodnoty z konečnej množiny bez prirodzeného usporiadania. Napríklad kategorická premenná by mohla brať hodnoty z množiny {červená, modrá, zelená}.

Základným účelom klasifikačnej štúdie môže byť v závislosti od problému buď vytvorenie presného klasifikátora alebo odhalenie prediktívnej štruktúry problému. Ak sa zameriavame na to druhé, potom sa snažíme pochopiť, ktoré premenné alebo interakcie predikčných premenných ovplyvňujú výstupnú premennú - to znamená dať jednoduchú charakterizáciu podmienok, ktoré určujú kedy je objekt skôr v jednej triede ako v inej triede. Tento prístup budeme v tejto práci opisovať v kapitole 1.8. Ak sa zameriavame na vytvorenie presného klasifikátora, musíme si určiť na základe čoho budeme presnosť merať. V literatúre sa často stretávame s pojmom validácia modelu.

Validácia modelu sa vykonáva s cieľom zistiť, či model bude vedieť dobre predpovedať hodnoty výstupnej premennej pre budúce subjekty alebo subjekty, ktoré neboli použité pre natrénovanie modelu. 3 hlavné príčiny zlyhania validácie modelu sú:

1) Pretrénovanie modelu
2) Zmeny v spôsobe merania prediktorov
3) Zmeny v zahrnutí prediktorov v subjektoch

Existujú dva hlavné spôsoby validácie modelu, externý a interný. Interný spôsob validácie modelu sa pozerá ako dobre klasifikátor roztriedil tie dáta, na ktorých bol trénovaný, teda tréningové dáta. Externý spôsob validácie modelu je taký, kde sa výkonnosť modelu testuje na dátach, ktoré neboli súčasťou konštrukcie modelu. Externé spôsoby však môžeme ďalej deliť na prísne a menej prísne. Prísne externé spôsoby validácie môžu zahrňovať dáta zozbierané z inej oblasti, od iných ľudí iným spôsobom. V našom prípade by to mohli byť dáta od inej firmy vyvíjajúcej mobilné hry. Keďže v našej diplomovej práci máme prístup len k jednému datasetu budeme sa zaoberať najmenej prísnou formou externej validácie modelu, ktorá spočíva v rozdelení dát, ktoré máme k dispozícii na tréningovú a testovaciu vzorku. Týmto rozdelením zapríčiníme stratu informácie pre tréningové dáta, ktoré sú dôležité pri konštrukcii klasifikátora, čo môže viesť k zhoršeniu výkonu modelu, ale na druhej strane budeme vedieť na testovacích dátach otestovať, či sa model nepretrénoval. Ak sa nebude meniť štruktúra dát, budeme mať kontrolu nad hlavnými tromi príčinami zlyhania validácie modelu. Ak by sa nám zdalo, že rozdelenie dát na tréningovú a testovaciu vzorku je príliš veľká strata informácií, podľa [8] je možné použiť bootstrap metódu, ktorá vie efektívne zabezpečiť aby aj tréningová aj testovacia časť dát mali takú veľkosť ako pôvodné dáta. Ďalšia možnosť je použiť cross-validáciu (z angl. cross-validation), ktorá spočíva vo vytvorení viacerých náhodných tréningových a testovacích vzoriek. Výsledný model bude potom v istom zmysle priemerným modelom zo všetkých vytvorených podmodelov. V tejto kapitole sme čerpali z [3] a [8].

1.2 Logistická regresia

Logistická regresia je vhodná regresná analýza, ktorá sa vykonáva najmä v prípade, keď je závislá premenná dichotomická, teda binárna. Rovnako ako všetky regresné analýzy, aj logistická regresia je prediktívna analýza. Logistická regresia sa používa na opis údajov

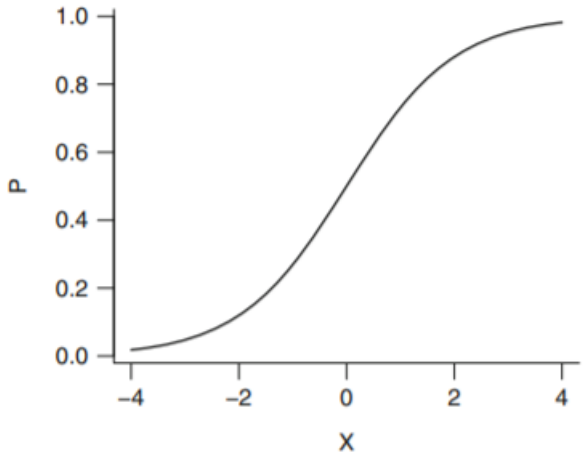
a na vysvetlenie vzťahu medzi jednou závislou binárnou premennou a jednou alebo viacerými nominálnymi, ordinálnymi, intervalovými alebo pomerovo nezávislými premennými. V prípade viacstavovej závislej, teda výstupnej premennej existujú algoritmy ako OVA (one-versus-all) alebo AVA (all-versus-all) pomocou ktorých sa viacstavový problém dá previesť do viacerých binárnych problémov. Pre jednoduchosť definujeme výstupnú premennú $Y = 0$ alebo $Y = 1$, pričom $Y = 1$ reprezentuje výskyt javu, ktorý pozorujeme. Často sa dá binárny výstup opísať pomocou pomerov niektorých veličín. Napríklad podľa pomeru javu vyskytujúceho sa u žien a u mužov. Zvyčajne však existuje viac, než jedna predikčná premenná, ktorá je často spojitá a používať pomery môže byť náročné. Označme $X = \{X_1, X_2, ..., X_n\}$ ako vector prediktorov. Prvý nápad ako modelovať správanie sa výstupnej premennej Y v závislosti od vektora prediktorov X by mohla byť lineárna regresia, ktorá je nasledovná:

$$E(Y|X) = X\beta$$

Tá však má viacero problémov, prvý, ako inidikuje názov je, že takýto vzťah je lineárny, čo vo všeobecnosti nemusí platiť. Ďalší problem je, že by sme chceli dostať pravdepodobnosť $P(Y=1)$, ale ak by sme $P(Y=1)$ položili rovné $E(Y=1|X)$ nedostali by sme hodnoty iba v intervale $[0,1]$. Štatistický model, ktorý sa pre analýzu binárnych výstupov vo všeobecnosti uprednostňuje je vyjadrený z hľadiska pravdepodobností, volá sa binárny logistický regresný model, ktorý podľa [8] vymysleli Cox, Walker a Duncan a je nasledovný:

$$P(Y = 1|X) = [1 + e^{(-X\beta)}]^{-1}$$

$X\beta = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$, pričom β sa odhadne buď pomocou metódy maximálnej vierohodnosti alebo metódou najmenších štvorcov. Častejšie používanou metódou je podľa [12] metóda maximálnej vierohodnosti. $p_1 = P(Y = 1|X)$ by sme mohli definovať aj iným spôsobom. Funkcia $f(x) = [1 + e^{(-x)}]^{-1}$ sa volá logistická funkcia (z angl. logistic function) a jej priebeh je znázornený na obrázku 1, ktorý sme prevzali z [8].



Obr. 1: Priebeh logistickej funkcie

Keďže máme len dve výstupné premenné a pridáme k tomu poznatok z teórie pravdepodobnosti, vieme že $p_1 + p_0 = 1$. Triedy 0 a 1 totiž môžeme ľubovoľne zamieňať. Štandardne sa však používa taká interpretácia, kde triedu 1 reprezentuje pozitívny jav. p_0, respektíve $1 - p_1$ vieme vypčítať nasledovne:

$$P(Y = 0|X) = 1 - [1 + e^{(-X\beta)}]^{-1}$$

$$P(Y = 0|X) = \frac{e^{(-X\beta)}}{[1 + e^{(-X\beta)}]}$$

Rovnako môžeme napísať rovnosť pre všeobecné x.

$$1 - p_1 = \frac{e^{(-X)}}{[1 + e^{(-X)}]}$$

Z tejto rovnice vieme vyjadriť x a dostávame:

$$x = \ln\left(\frac{p_1}{1 - p_1}\right)$$

Táto funkcia je inverznou logistickou funkciou a nazýva sa logit funkcia. Vyjadruje logaritmus pomeru pravdepodobností, že Y bude triedy 1 k tomu, že bude triedy 0, inými slovami vyjadruje logaritmus šancí (z angl. odds), že Y bude triedy 1.

Keďže logistický model je priamym pravdepodobnostným modelom, jeho jediné predpoklady súvisia s regresnou rovnicou. Predpoklady logistického modelu sú ľahko pochopiteľné transformáciou pravdepodobnosti p_1, tak aby bol model lineárny v $X\beta$.

$$logit(Y = 1|X) = logit(p_1) = \ln\left(\frac{p_1}{1 - p_1}\right) = X\beta$$

Tento model predpokladá, že pre každú predikčnú premennú X_i platí:

$$logit(Y = 1|X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n = \beta_i X_i + C$$

Za predpokladu, že ostatné predikčné premenné okrem X_i sú konštantné, je C konštanta. Parameter β_i je potom zmena v logaritme šancí na jednotku v X_i. Pre lepšiu interpretáciu vieme vyjadriť šance (odds), že Y bude triedy 1:

$$odds(Y = 1|X) = e^{\beta_i X_i + C}$$

Regresné parametre vieme vyjadriť aj pomocou pomeru šancí:

$$\frac{odds(Y = 1|X_1, X_2, \dots, X_i + d, \dots, X_n)}{odds(Y = 1|X_1, X_2, \dots, X_i, \dots, X_n)} = e^{\beta_i d}$$

Z hore uvedenej rovnice vyplýva, že ak sa hodnota X_i posunie o d jednotiek, pomer šancí sa zmení o $e^{\beta_i d}$. O koľko sa zmení pravdepodobnosť danej triedy záleží od pôvodnej pravdepodobnosti. Čím bližšie je pôvodná pravdepodobnosť k 0,5, tým viac sa potom zmenou jedného alebo viacerých z prediktorov mení pomer šancí. Takáto metóda má potom tendenciu vytvárať väčšinu pravdepodobností buď blízko k číslu 0 alebo blízko k číslu 1. Na základe takejto regresie môže vzniknúť klasifikátor. Pre klasifikátor je dobré, ak väčšina pravdepodobností je na "chvostoch", aby sa dali dobre oddeliť.

Podľa [14] je pri logistickej regresii dôležité pri výbere modelu dbať aj na to, aby predikčné premenné neboli medzi sebou závislé. V tejto kapitole sme čerpali najmä z [14] a [8].

1.3 Bayesov naivný klasifikátor

Bayesovský klasifikátor (z angl. naive Bayes) je algoritmus dohliadaného strojového učenia založený na Bayesovej vete, ktorý sa používa na riešenie problémov klasifikácie nasledovaním pravdepodobnostného prístupu. Je založený na myšlienke, že predikčné premenné v modeli strojového učenia sú navzájom nezávislé. Znamená to, že výsledok modelu závisí od súboru nezávislých premenných, ktoré spolu nemajú nič spoločné. V problémoch skutočného sveta nie sú predikčné premenné vždy na sebe nezávislé, vždy

medzi nimi existuje aspoň nejaká korelácia. Keďže algoritmus považuje každú predikčnú premennú za nezávislú od ktorejkoľvek inej premennej v modeli, nazýva sa naivný.

Za algoritmom stojí Bayesova veta, známa tiež ako Bayesov zákon. Bayesova veta sa používa na výpočet podmienenej pravdepodobnosti, čo nie je nič iné ako pravdepodobnosť výskytu udalosti na základe informácií o udalostiach v minulosti. Bayesov zákon vyzerá nasledovne:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

pričom:

P (A | B) - Podmienená pravdepodobnosť výskytu udalosti A za podmienky B

P (B | A) - Podmienená pravdepodobnosť výskytu udalosti B za podmienky A

Vyššie uvedená rovnica bola pre jednu predikčnú premennú, avšak v aplikáciách v reálnom svete existuje viac ako jedna predikčná premenná a pre klasifikačný problém existuje viac ako jedna výstupná trieda. Triedy môžu byť reprezentované ako, C_1, C_2, ..., C_k a predikčné premenné môžu byť reprezentované ako, x_1, x_2, ..., x_n. Potom platí:

$$P(c_i|x_1, x_2, \dots, x_n) = \frac{P(x_1, x_2, \dots, x_n|c_i)\,P(c_i)}{P(x_1, x_2, \dots, x_n)} \cdot \forall i = 1,2, \dots, k$$

Po úpravách dostávame:

$$P(c_i|x_1, x_2, \dots, x_n) = \prod_{j=1}^{n} P(x_j|c_i)\,\frac{P(c_i)}{P(x_1, x_2, \dots, x_n)} \cdot \forall i = 1,2, \dots, k$$

Stav konečnej klasifikácie je tá trieda pre ktorú prislúchajúci výraz je maximálny spomedzi všetkých ostatných pre dané predikčné premenné v danom prvku. V tejto kapitole sme čerpali z [7]

1.4 Metóda oporného bodu

Metóda oporného bodu je tiež algoritmus dohliadaného strojového učenia, ktorý slúži na klasifikáciu a regresnú analýzu, pričom v praxi je metóda využívaná najmä pri klasifikačnom probléme. V prípade binárnej klasifikácie táto metóda rozdeľuje priestor pomocou nadroviny na dve oblasti, z ktorých každá reprezentuje jednu klasifikačnú triedu.

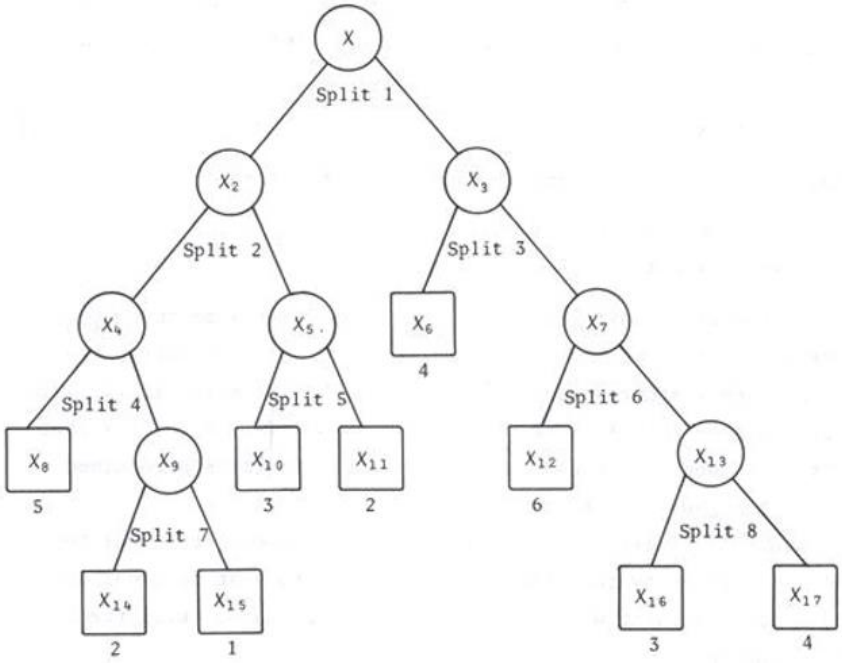
V prípade lineárne separovateľných dát je najlepšia tá nadrovina, ktorej vzdialenosť od najbližších bodov z oboch kategórií je najväčšia. Väčšina tréningových dát je však lineárne neseparovateľná, teda nedá sa rozdeliť pomocou jednej nadroviny na dve oblasti, ktoré by obsahovali jedine objekty z jednej a tej istej klasifikačnej triedy. Metóda oporného bodu v tomto prípade spočíva v tom, že sa istým správnym spôsobom dáta zobrazia v priestore s vyššou dimenziou pomocou takzvaného Kernel triku (z angl. Kernel trick), pričom v tomto viac-dimenzionálnom priestore už budú dáta lineárne separovateľné. Proces nájdenia správnej funkcie novej dimenzionálnej premennej však môže byť výpočtovo veľmi náročný, čo je asi hlavnou nevýhodou tohto algoritmu. Ďalšia nevýhoda je, že Kernel trick funkcie môžu byť veľmi zložité, čo spôsobí, že sa algoritmus stane „čiernou skrinkou“ do ktorej nevidíme. Nevýhody tejto metódy sú najviac pozorovateľné v prípade, že tréningové dáta sú rozsiahleho charakteru, čo sa bude v tejto diplomovej práci vyskytovať často a preto sme sa rozhodli túto metódu opísať len takto stručne. V tejto kapitole sme čerpali zo zdrojov [11] a [5].

1.5 Klasifikačné stromy

Klasifikačné (rozhodovacie) stromy patria medzi jednu z najpoužívanejších metód klasifikácie. Majú jednoduchú interpretáciu, pričom zostrojenie stromu nie je náročný proces. Ich veľkou výhodou je, že vieme klasifikátor vykresliť pomocou dobre interpretovateľného stromového grafu. Strom sa skladá z uzlov, ktoré delíme na tri typy:

- Koreňový uzol (z angl. root node)
    - Každý strom má práve jeden koreňový uzol. Špecifický je tým, že do neho (zhora) nesmerujú žiadne hrany, iba z neho smerujú smerom von (dole).
    - Koreňový uzol obsahuje všetky tréningové dáta
- Testovací uzol (z angl. test node)
    - Nazývaný aj rodičovský uzol (parent node)
    - Do každého testovacieho uzlu smeruje jedna hrana zhora a spravidla z neho smerujú dve hrany smerom dole.
- Terminálny uzol (z angl. terminal node)
    - Nazývaný aj rozhodovací uzol (decision node) alebo list (leaf)
    - Z terminálneho uzla už nevychádza žiadna hrana smerom dole.
    - Každý terminálny uzol vyhodnocuje stav konečnej klasifikácie

Okrem hore uvedeného rozdelenia môžeme uzol nazývať aj ako dcérsky uzol (z angl. daughter node). Takýto uzol je každý , ktorý má „rodiča“, teda všetky okrem koreňového.



Obr. 2: Príklad grafu klasifikačného stromu

Na obrázku 2 prevzatého z [3] vidíme príklad znázornenia klasifikačného stromu pomocou stromového grafu. Tento klasifikačný strom má 6 výstupných tried reprezentované číslami 1 až 6. X a X_i pre i=2,3,…,17 reprezentujú uzly, pričom tie, ktoré sú ohraničené kružnicou sú až na jednu výnimku testovacími uzlami. Uzol X je koreňovým uzlom. Uzly ohraničené štvorcom sú terminálne uzly, ktoré určujú konečnú klasifikáciu výstupnej triedy kde číslo pod týmito uzlami reprezentuje danú výstupnú triedu. Ďalej platí: Ak sa pozeráme len na jednu horizontálu, tak v prípade, že nad touto horizontálou sa nenachádza terminálny uzol, dáva zjednotenie uzlových bodov celé dáta. Napríklad $X_4 \cup X_5 \cup X_6 \cup X_7 = X$. Aj zjednotením množiny všetkých terminálnych uzlov, ktoré sa nazýva aj rozdelenie dát dostávame celý tréningový dataset X. Terminálnych uzlov môže byť pre jednu konkrétnu výstupnú premennú vo všeobecnosti ľubovoľne veľa, napríklad pre strom z obrázku 2 klasifikujeme ako triedu 2 všetky dáta, ktoré sa nachádzajú v uzloch X_{14} a X_{11}.

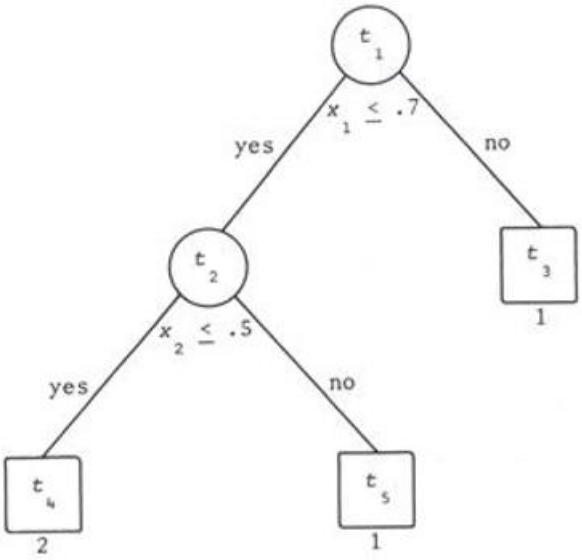
Spôsobov ako zostrojiť strom je niekoľko, pričom algoritmus známy ako rekurzívne delenie je kľúčom k neparametrickej štatistickej metóde klasifikačných a regresných stromov. Rekurzívne delenie je proces pomocou ktorého je budovaný rozhodovací strom a to buď spôsobom, že existujúci testovací uzol sa rozvetví na dva nové dcérske uzly alebo

sa nerozvetví. Klasifikačný a regresný strom sa líšia vo vlastnosti výstupnej premennej. Kým regresný strom pracuje so spojitou výstupnou premennou, klasifikačný strom pracuje s diskrétnou výstupnou premennou. Najčastejšie sa jedná o diskrétnu výstupnú premennú s dvomi rôznymi výstupmi, teda binárnu premennú, často z hodnotami 0/1 alebo áno/nie. V takom prípade hovoríme o binárnom klasifikačnom strome. Proces tvorby klasifikačného stromu je nasledovný:

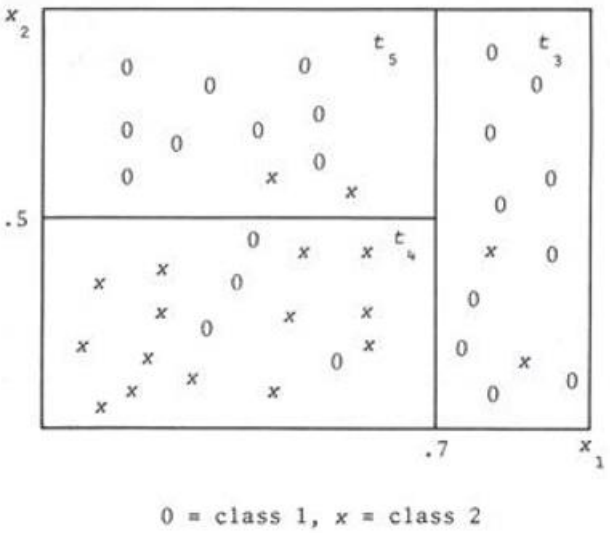
Nech $x_1, \dots, x_p$ je vektor prediktorov a θ_i je otázka, ktorú sa pýtame v nejakom testovacom uzli i. Otázka môže byť napríklad typu $x_j = 5$ alebo napríklad $x_7 \subset \{$červená, modrá, zelená$\}$. Otázka vždy musí roztriediť dáta v uzle na dve nové podmnožiny, teda dva nové dcérske uzly, pričom každý z nich musí byť neprázdny, inak otázka nemá zmysel. Potom postupujeme nasledovne:

1. Vytvoríme nový uzol, ktorý bude koreňovým uzlom alebo potomkom niektorého z doteraz terminálnych uzlov. Ak ide o druhú možnosť, vzniknú nám dva nové uzly.
2. Rozhodneme, či novo vytvorený uzol bude testovací alebo terminálny na základe toho aká je distribúcia tried jednotlivých prvkov v uzle. V ideálnom prípade terminálne uzly obsahujú prvky iba jednej triedy.
3. Ak je uzol terminálny, priradíme mu triedu. Ak uzol nie je terminálny pokračujeme rekurentne bodom 1.

Čo v skutočnosti robí klasifikačný strom si vysvetlíme na nasledujúcom príklade stromu, ktorý je znázornený na obrázku 3. Tréningové dáta na ktorých tento strom vznikol majú 2 predikčné premenné x_1 a x_2, ktoré pre jednoduchosť môžu nadobúdať jedine hodnoty v intervale $[0, 1]$ a dve výstupné triedy 1 a 2. Tento binárny klasifikačný strom má 3 terminálne uzly.



Obr. 3: Príklad binárneho klasifikačného stromu



Obr. 4: Rozdelenie časti roviny R^2 klasifikačným stromom.

Keďže spomínaný strom má 3 terminálne uzly, rozdelí nám priestor $[0, 1]x[0, 1] \subset R^2$ na 3 oblasti: t_3,t_4,t_5 ako je znázornené na obrázku 4. Body nachádzajúce sa v oblasti t_4 klasifikátor klasifikuje ako triedu 2 (značená znakom x) a ostatné oblasti ako triedu 1 (značená znakom 0). Môžeme si všimnúť, že niektoré body z dát sú zle klasifikované. Keby sme chceli, vedeli by sme zostrojiť taký strom, ktorý by žiaden z bodov neklasifikoval zle, bol by však omnoho zložitejší. Dôležité však je uvedomiť si, že klasifikačný strom vytvára v prípade R^2 obdĺžniky, ktorým priradí triedu. Vo vyšších rozmeroch vytvára klasifikačný strom kvádre, hyperkvádre a tak ďalej V prípade ordinálnych (diskrétnych) premenných

vytvára klasifikačný strom kvádre o toľko rozmerov nižšie koľko ordinálnych predikčných premenných v tréningových dátach je, ale robí to pre karteziánsky súčin množín všetkých hodnôt, ktoré v tréningových dátach jednotlivé diskrétne premenné nadobúdajú.

Aby sme mohli zostrojiť klasifikačný strom, musíme si zodpovedať na 3 základné otázky:

1) Na základe čoho sa rozhodneme rozdeliť testovací uzol na dva dcérske uzly? Respektíve, kedy sa prestaneme pýtať otázky a vyhlásime uzol za terminálny?
2) Ak rozhodneme, že uzol nebude terminálny akú otázku sa spýtame?
3) Ako priradíme terminálnemu uzlu triedu?

Pre zodpovedanie bodov 1) a 2) je najdôležitejšie určiť vhodný spôsob výberu otázok. Pri výbere otázok je najdôležitejšie to, aby dcérske uzly boli čo najčistejšie, pričom čistý uzol je taký, ktorý obsahuje prvky len jednej klasifikačnej triedy. V každom uzle sa musí algoritmus rozhodnúť cez ktorú predikčnú premennú bude uzol deliť na dva nové dcérske. Algoritmus môže spraviť všetky možné rozdelenia cez všetky premenné a na základe nejakého pravidla určiť ktoré rozdelenie je najlepšie. Pre ordinálne a diskrétne premenné je počet otázok konečný, avšak pre spojité premenné je počet možných otázok nekonečný. Preto sa musí algoritmus k spojitým premenným správať v istom zmysle ako k diskrétnym. Napríklad ak máme premennú x, ktorá nadobúda hodnoty v intervale $(0, 1)$ môžeme tento interval rozdeliť na dva podintervaly $(0; 0{,}5]$ a $(0{,}5; 1)$ a všetky hodnoty nachádzajúce sa v jednom intervale považovať za rovnaké. Potom v takom prípade dostaneme cez predikčnú premennú x len jedno možné rozdelenie. Analogicky ak máme spojitý prediktor v intervale (a,b), môžeme ho rozdeliť na c častí, pričom c môže byť ľubovoľne veľké. Keď podobné rozdelenia urobíme pre všetky prediktory dostávame konečný počet možných otázok z ktorých môžeme vybrať tú najlepšiu podľa nejakého kritéria. Existujú rôzne postupy ako hľadať najvhodnejšie otázky medzi ktoré patria napríklad: *miera chybovosti klasifikácie, informačná entropia* alebo *Giniho index*. V našej diplomovej práci sme použili metódu Giniho indexu, kde sa pri tvorbe každej otázky rieši nasledovná optimalizačná úloha:

$$\min \sum_{c=1}^{c} \hat{\pi}_{m_c}\left(1 - \hat{\pi}_{m_c}\right)$$

Pričom $\hat{\pi}_{m_c}$ predstavuje podiel prvkov v uzle m, ktoré prislúchajú do triedy c.

Teraz, keď už poznáme odpoveď na bod 2) a teda aký výber otázok je najvhodnejší, vieme jednoduchšie odpovedať na bod 1), teda sa rozhodneme či daný uzol bude terminálny alebo nebude. Napríklad ak nečistota podľa Giniho indexu, teda výsledok hore-uvedenej optimalizačnej úlohy je menší, než nejaké číslo t, rozhodneme sa tento uzol ďalej nevetviť a vyhlásime ho za terminálny. Ďalší spôsob podľa ktorého sa môžeme rozhodnúť je či sa rozvetvením zlepší celkový výkon modelu, čiže napríklad či sa priemerná hodnota Giniho indexov pre všetky terminálne uzly vytvorením nových dvoch uzlov zníži o číslo r. Odpoveď na otázku 3), teda ako rozhodnúť akú triedu priradíme terminálnemu uzlu je z týchto troch otázok asi najjednoduchšia. Najčastejší spôsob je priradiť uzlu tú triedu, ktorej prvkov obsahuje najviac zo všetkých možných tried.

V tejto kapitole sme čerpali hlavne zo zdrojov [9] a [3].

1.6 Vlastnosti klasifikátora

V práci sa budeme snažiť nájsť čo najlepší klasifikátor. To, či je klasifikátor dobrý budeme posudzovať podľa takzvanej Matice zámen (z angl. confusion matrix), ktorá má nasledovnú štruktúru:

Tabuľka 1: Matica zámen

	Realita: False	Realita: True
Predikcia: False	TN	FN
Predikcia: True	FP	TP

$TN, FN, FP, TP \in \mathbb{N}_0$

TN – Počet negatívnych pozorovaní, ktoré v skutočnosti nastali a zároveň klasifikátor predpovedal, že nastanú (po angl. true negatives)

FN – Počet negatívnych pozorovaní, ktoré v skutočnosti nastali, ale klasifikátor predpovedal, že nenastanú (po angl. false negatives) alebo aj chyba II. typu (z angl. type II error).

FP – Počet pozitívnych pozorovaní, ktoré v skutočnosti nastali, ale klasifikátor predpovedal, že nenastanú (po angl. false positives) alebo aj chyba I. typu (z angl. type I error)

TP – Počet pozitívnych pozorovaní, ktoré v skutočnosti nastali a zároveň klasifikátor predpovedal, že nastanú (po angl. true positives)

Zo štruktúry matice zámen jasne vyplývajú nasledovné tvrdenia:

TN+FP – Počet všetkých skutočných negatívnych pozorovaní

FN+TP – Počet všetkých skutočných pozitívnych pozorovaní

TN+FN – Počet všetkých predikovaných negatívnych pozorovaní

FP+TP – Počet všetkých predikovaných pozitívnych pozorovaní

TN+TP – Počet správne klasifikovaných pozorovaní

FN+FP – Počet nesprávne klasifikovaných pozorovaní

TN+FN+FP+TP – Počet všetkých pozorovaní

Ďalej definujme ďalšie 3 pojmy:

Presnosť (z angl. accuracy) klasifikátora je $\frac{TN+TP}{TN+FN+FP+TP}$,teda podiel správne klasifikovaných pozorovaní k všetkým pozorovaniam.

Senzitivita (z angl. sensitivity) klasifikátora je $\frac{TP}{FN+TP}$, teda podiel pozitívnych pozorovaní, ktoré v skutočnosti nastali, a zároveň klasifikátor správne predikoval, že nastanú ku všetkým skutočným pozitívnym pozorovaniam.

Precíznosť (z angl. precision) klasifikátora je $\frac{TP}{FP+TP}$, teda podiel pozitývnych pozorovaní, ktoré v skutočnosti nastali a zároveň klasifikátor správne predikoval, že nastanú ku všetkým, predikovaným pozitívnym pozorovaniam.

Môžeme si všimnúť, že ak chyba I. druhu klesá, precíznosť stúpa a naopak. Ak chyba I. druhu je rovná 0, precíznosť je rovná 1. Takisto ak chyba II. druhu klesá, stúpa senzitivita a naopak. Ak chyba II. druhu je rovná 0, senzitivita je rovná 1. Ak rastie senzitivita alebo precíznosť, rastie aj presnosť. Čím menšie sú chyby I. a II. druhu, tým je klasifikátor lepší. Ekvivalentne môžeme tvrdiť, že čím vyššie sú hodnoty pre senzitivitu a precíznosť, tým je klasifikátor lepší a teda budeme sa snažiť nájsť také klasifikátory, ktorých senzitivita a precíznosť sú čo najbližšie k číslu 1. Problém je, že ak sa snažíme zvyšovať jedno, nepriamo tým znižujeme to druhé.

Sú dva spôsoby akými vieme zvyšovať ukazovatele dobrých klasifikátorov: cez klasifikátor – tunovaním modelu a cez maticu zámen – zmenou prahu, pričom pre konštrukciu dobrého klasifikátora je dôležité použiť oba spôsoby.

Najprv vysvetlíme ako vieme zmeniť niektoré ukazovatele ako precíznosť, citlivosť alebo presnosť pomocou zmeny prahu. Každý klasifikátor totiž každému subjektu, či už pri tréningu alebo pri predikcii nových pozorovaní, priraďuje pravdepodobnosť triedy. V prípade binárnej klasifikácie stačí brať do úvahy jednu pravdepodobnosť, v našom prípade nech je to pravdepodobnosť pozitívneho javu. Na základe týchto pravdepodobností sa klasifikátor rozhoduje, ktorým subjektom priradí triedu prislúchajúci pozitívnemu javu a to tak, že vezme všetky tie, ktorých pravdepodobnosť je nad nejakým prahom *t* (z angl. threshold). V závislosti od rôznych *t* dostávame rôzne matice zámen, z ktorých následne môžeme vypočítať rôzne hodnoty presnosti, precíznosti a citlivosti, ale aj iných metrík. Ak máme N subjektov, rôznych matíc zámen môže byť v prípade unikátnej pravdepodobnosti pre každý jeden subjekt až $N+1$. Často môže byť tento prah napríklad $\frac{1}{2}$, ale väčšina pokročilých algoritmov vyberá taký prah, pre ktorý je presnosť klasifikátora najvyššia.

Druhým spôsobom ako dokážeme docieliť zvýšenie cieľových charakteristík úspešnosti klasifikátorov je takzvané ladenie modelu (z angl. model tuning) alebo inak povedané optimalizácia hyperparametrov. Hyperparameter je parameter, ktorého hodnota sa používa na riadenie procesu učenia. Napríklad v prípade klasifikačného stromu je jedným z hyperparametrov hĺbka stromu. Alebo za hyperparameter môžeme považovať aj to, či predikčné premenné navzájom majú alebo nemajú rovnakú škálu. Rovnaký druh modelu strojového učenia môže vyžadovať rôzne hodnoty hyperparametrov pre rôzne tréningové dáta z hľadiska optimalizácie modelu. Tradičným spôsobom vykonávania optimalizácie hyperparametrov je vyhľadávanie cez mriežku (z angl. grid search), čo je jednoduché prehľadávanie ručne určenou mriežkovou podmnožinou hyperparametrového priestoru vzdelávacieho algoritmu. Napríklad v prípade klasifikačných stromov by sme mohli prehľadať priestor hyperparametrov *cp*, ktorý je pre konštrukciu klasifikačných stromov kľúčový z množiny $\{0{,}02;\ 0{,}04;\ 0{,}06;\ 0{,}08;\ 0{,}10\}$. Okrem vyhľadávania cez mriežku je známy aj algoritmus náhodných hodnôt. Príklad: vyskúšajme ako parameter cp 5 náhodných hodnôt z intervalu $[0{,}02;\ 0{,}10]$. Podľa [1] je z týchto dvoch metód efektívnejšia náhodná metóda. Pred použitím oboch metód je dobré urobiť prieskum priestoru hyperparametrov “ručne”, pretože vo všeobecnosti môžu hyperparametre nadobúdať

hodnoty z množiny všetkých reálnych čísel, ale aj iných množín, a my chceme nájsť ten najlepší parameter v konečnom čase. V konečnom dôsledku môže optimalizácia hyperparametrov zásadne zmeniť kľúčové charakteristiky výkonnosti klasifikátora.

1.7 Vlastnosti klasifikátora graficky

Existuje veľa možných spôsobov na meranie výkonnosti pravidiel klasifikácie. V ideálnom prípade by sme mali vybrať konkrétne kritérium výkonnosti, ktoré zodpovedá aspektu výkonu, ktorý považujeme za najdôležitejší pre konkrétne použitie. Často je však náročné identifikovať takýto ústredný aspekt a aj keď už nejaký identifikujeme, nemusí byť dlhodobo práve tento ten najdôležitejší, pretože podmienky sa môžu zmeniť. Z týchto dôvodov je často užitočné mať spôsob, ako zobraziť a zosumarizovať výkonnosť v širokom rozsahu. A presne to robí krivka ROC. Krivka ROC (z angl. receiver operating characteristics curve) je krivka, ktorá vyjadruje závislosť medzi senzitivitou (y-ová os) a veličinou 1 – špecificita (x-ová os), pričom špecificita (z angl. specificity) je definovaná ako $\frac{TN}{TN+FP}$, a teda veličina 1 – špecificita je rovná $\frac{FP}{TN+FP}$. Senzitivita je rovná $\frac{TP}{FN+TP}$. Táto závislosť môže byť rôzna závisiac od hodnoty prahu t (z angl. threshold). ROC krivka sa dá považovať za úplné znázornenie výkonu klasifikátora, vzhľadom na všetky možné prahy t. Pomocou krivky ROC môžeme merať úspešnosť klasifikátora rôznymi spôsobmi. Napríklad miera nesprávnej klasifikácie môže byť minimálna vzdialenosť medzi krivkou a ľavým horným rohom štvorec obsahujúci graf ROC, v prípade, že osi sú primerané veľkostiam klasifikačných tried. Pravdepodobne najpoužívanejšou mierou pre určovanie úspešnosti klasifikácie pomocou ROC krivky je plocha pod grafom, bežne označovaná ako AUC (z angl. area under the curve). V prípade perfektného klasifikátora je plocha pod krivkou ROC rovná obsahu štvorca 1x1, teda 1. zatiaľ čo v prípade klasifikátora, ktorý náhodne priradzuje triedy je AUC rovné obsahu trojuholníka pod uhlopriečkou štvorca 1x1, teda 0,5. Vo všetkých ostatných prípadoch je formálna definícia hodnoty nasledovná:

$$AUC = \int_0^1 y(x)dx.$$

Z tejto definície jasne vyplýva, že ak máme dva klasifikátory A a B, tak ak A je na celom grafe pod B, tak AUC pre B bude väčšie, než AUC pre A. Opačná implikácia žiaľ neplatí, pretože dve krivky sa vo všeobecnosti môžu pretínať. Práve pre pretínajúce sa krivky môže byť AUC dobrým meradlom výkonnosti klasifikátora, pričom čím väčšie AUC, v teórii tým lepší klasifikátor. V tejto časti sme sa inšpirovali zdrojom [4].

Precision-Recall krivky (ozn.P-R) sú podobne ako s nimi úzko súvisiace ROC krivky, hodnotiacim nástrojom pre binárnu klasifikáciu, ktoré pomáhajú vo vizualizácii výkonnosti klasifikátorov pre celkový rozsah prahov. P-R krivky sú stále viac používané pri strojovom učení najmä pri nevyvážených datasetoch, v ktorých sa jedna klasifikačná trieda vyskytuje častejšie ako tá druhá. V týchto nevyvážených alebo zošikmených dátach sú P-R krivky užitočnou alternatívou ku krivkám ROC, ktoré môžu zvýrazňovať rozdiely vo výkone nezachytené pomocou kriviek ROC. Aj v prípade P-R kriviek sa používa oblasť pod krivkou, označenie AUCPR (z angl. area under the curve, precision-recall) ako všeobecná miera výkonnosti klasifikátora. Samotná P-R krivka vyjadruje závislosť precíznosti (y-ová os) od senzitivity (x-ová os). P-R krivka sa konštruuje podobne ako ROC krivka a to vykreslením bodov pre páry senzitivít a precízností pre všetky rôzne matice zámen v závislosti od všetkých prahov. Pospájaním týchto bodov získavame obraz P-R krivky, ktorý sa rovnako ako ROC krivka nachádza v podmnožine R^2, v štvorci [0,1]x[0,1]. Avšak najlepšia metóda na zostavenie P-R krivky nie je jednoznačná. Pre jednotlivé body na x-ovej osi (reprezentujúce senzitivitu) môžeme vo všeobecnosti dostať veľa hodnôt precízností. Existujú najmenej štyri rôzne metódy s niekoľkými variáciami ako danú P-R krivku a jej obsah pod ňou vypočítať. Rozdiely vo výsledkoch využívajúcich rôzne prístupy sú najvýraznejšie, keď sú údaje veľmi skreslené, čo je presne ten prípad, keď sú P-R krivky výhodnejšie, než ROC krivky. Podľa [6] je pri nevyvážených dátach krivka P-R lepším meradlom dobrého klasifikátora, než častejšie používaná krivka ROC. Tá totiž najviac odzrkadluje presnosť modelu, ktorá je v nevyvážených dátach často vysoká a nereaguje príliš na malé výkyvy v precíznosti a senzitivite, ktoré naopak neovplyvňujú príliš presnosť, ale ovplyvňujú správnosť pri predikcii pozitívneho, respektíve toho málo pravdepodobného javu. A práve odhadovanie pozitívneho javu, čo je v našom prípade či hráč zaplatí je to podstatné a je to zároveň málo pravdepodobný jav. V tejto kapitole sme sa inšpirovali [6] a [2].

1.8 Určovanie dôležitosti premenných

Často chceme odmerať silu vzťahu medzi prediktormi a výsledkom. Keď sa počet predikčných premenných zvyšuje, analýza všetkých prediktorov môže byť nerealizovateľná a sústredenie sa na tie, ktoré majú silné vzťahy s výsledkom, môže byť efektívnou stratégiou pre natrénovanie dobrého a interpretovateľného modelu. Zoradenie prediktorov takýmto spôsobom môže byť veľmi užitočné najmä ak pracujeme s veľkými dátami, a to najmä čo sa týka počtu prediktorov. Jedným z hlavných dôvodov na meranie sily alebo relevantnosti

prediktorov je zostrojenie filtra, ktorý by sa mal použiť pre vstupy do modelu. Výsledky procesu filtrovania spolu s odbornými znalosťami môžu byť kritickým krokom pri vytváraní efektívneho klasifikátora. Mnoho prediktívnych modelov má zabudované vlastné špecifické merania dôležitosti prediktora. Z klasifikačných metód, ktorým sa v tejto práci venujeme je to klasifikačný strom, ktorý zachytáva zvýšenie výkonu, ku ktorému dochádza pri pridávaní každého prediktora do modelu. Logistická regresia zas používa kvantifikácie založené na modelových koeficientoch alebo štatistických mierach ako je napríklad t-štatistika.

Meranie dôležitosti prediktora sa v podstate odvodzuje pre každý prediktor jednotlivo a to hodnotením straty výkonu, keď daný prediktor pre natrénovanie modelu vynecháme. V tomto prípade výrazný pokles výkonnosti naznačuje dôležitý prediktor. Aj keď to môže byť efektívnym prístupom, nehovorí nám to nič o presnej formu vzťahu. Napriek tomu môžu byť tieto merania užitočné pri usmerňovaní, aby sme sa prostredníctvom vizualizácií a iných prostriedkov bližšie zamerali na konkrétne prediktory. Väčšina algoritmov na určovanie dôležitosti prediktorov je špecifická pre daný klasifikačný alebo regresný model.

Ak máme numerický výstup a numerické prediktory klasický prístup je taký, že na kvantifikáciu vzťahu predikčnej premennej s výsledkom sa použije korelačný koeficient. Táto metrika však meria len lineárnu závislosť medzi vstupom a výstupom. Ak je vzťah takmer lineárny alebo krivočiary, potom môže byť Spearmanov korelačný koeficient efektívnejší. Tieto metriky by sa však mali považovať za hrubé odhady vzťahu a nemusia byť účinné pri zložitejších vzťahoch. Alternatívou je použitie flexibilnejších metód, ktoré môžu byť schopné modelovať všeobecné nelineárne vzťahy. Jednou takouto technikou je lokálne vážený regresný model, známy ako LOESS. Ak sú prediktory kategorické, používajú sa iné metódy, ktoré v tejto práci nebudeme opisovať, pretože budeme pracovať s numerickými prediktormi. V prípade kategorického výstupu a numerických prediktorov existuje viacero prístupov na kvantifikáciu dôležitosti prediktorov. Jedným z nich je použiť oblasť pod krivkou ROC na určenie relevantnosti prediktora. Toto funguje za predpokladu existencie práve dvoch výstupných tried. Hodnoty prediktora použijeme ako vstupy do krivky ROC. Ak by prediktor vedel dokonale oddeliť triedy, AUC, teda oblasť pod krivkou ROC by sa rovnala 1-ke. Naopak, zlý prediktor by mal plochu pod krivkou v priemere približne 0,5. V tejto kapitole sme čerpali zo zdroja [10].

2 Praktická časť

2.1 Klasifikačná úloha

Hráčov mobilných hier môžeme vo všeobecnosti rozdeliť na dve triedy: platičov a neplatičov. Firma Pixel má zisk z hráčov, ktorí do hry investujú peniaze a je pre ňu dôležité vedieť týchto hráčov identifikovať. Je dôležité vedieť, ktorí hráči v budúcnosti zaplatia, pretože okrem toho, že budú mať výhody priamo v hre, by mali títo hráči mať výhodu oproti neplatičom aj mimo hry. Napríklad v prípade otázok alebo pri riešení nejakého problému je v záujme firmy prioritne riešiť tých hráčov, z ktorých majú potenciálny zisk. Ak by potenciálny budúci platič mal nejaký problém v hre, ktorý by vývojárom hry trvalo vyriešiť príliš dlho, tento hráč by si to s hrou mohol rozmyslieť, čo by v konečnom dôsledku pre firmu znamenalo stratu zisku. Ak by sa to isté stalo hráčovi, ktorí má veľmi nízky potenciál stať sa platičom firma by stratila jedného hráča a možno tento hráč by hru nikomu neodporučil, ale firma nepríde o zisk. Z hľadiska maximalizácie zisku je teda pre firmu veľmi dôležité vedieť identifikovať budúcich platičov. Okrem identifikácie potenciálnych platičov je pre firmu dôležité aj správne odhadnúť ich počet. Firma má k dispozícii informáciu odkiaľ hráč prišiel, často je to z marketingovej kampane. Pri vyhodnocovaní úspešnosti marketingovej kampane nezáleží na tom ktorí hráči zaplatia, ale záleží na tom koľko platiacich hráčov daná kampaň priláka. Reklamné kampane firmu stoja peniaze a musí sa v podstate v každom momente rozhodovať, či bude s kampaňou pokračovať alebo s ňou prestane. Odhadovaný počet platičov je v tomto prípade dôležitá informácia pri rozhodovaní. Okrem vyhodnocovaní kampaní je vhodné vedieť počet hráčov, ktorí zaplatia aj kvôli budúcemu zisku firmy. Môže pomôcť pri rozhodovaní sa koľko peňazí si firma môže dovoliť investovať do vylepšení starých alebo nových hier. Ak poznáme počet platičov, vieme jednoduchšie odhadnúť budúci výnos. K dispozícii máme dáta o nových hráčoch v hre a ak chceme vedieť, či v budúcnosti zaplatia alebo nezaplatia bude pre nás vhodnejšie venovať sa klasifikácii (a nie regresii) a keďže výstupné triedy sú dve, budeme v tejto práci konštruovať binárny klasifikátor.

2.2 Metodológia určovania úspešnosti klasifikátora

Chceme nájsť taký klasifikátor, ktorý bude vedieť dobre odhadnúť počet platičov a zároveň identifikovať tých, ktorí to budú. To sú však dve podmienky a z teórie optimalizácie vieme, že ak chceme niečo optimalizovať, účelová funkcia môže byť len jedna. Ako primárne kritérium úspešnosti klasifikátora sme sa preto rozhodli použiť takú metriku, ktorá bude odzrkadlovať koľko z budúcich platičov sa nám podarilo identifikovať.

Ako sekundárne budeme považovať to, či klasifikátor odhadol správny počet platičov. Pre určovanie primárneho kritéria úspešnosti nás bude najviac zaujímať matica zámen pre testovacie dáta. Z nej chceme zistiť, aké percento budúcich platičov sa nám podarilo predikovať ako platičov, z čoho prirodzene ako meradlo dostávame pomer: $\frac{\mathrm{TP}}{FN+TP}$, teda senzitivitu.

V prípade, že nás zaujíma len počet hráčov, ktorí zaplatia, a nezáleží by nám na tom, aby sme vedeli ktorí z daných hráčov to budú, nemusíme sa zameriavať na precíznosť, senzitivitu alebo presnosť, ale dobrý bude aj taký klasifikátor, pre ktorý bude chyba I. druhu približne rovná chybe II. druhu, respektíve FN∼FP, preto definujeme nasledovný pojem:

Nech relatívny rozdiel v počte, RCD, (z angl. relative count difference) je $\frac{FP+TP}{FN+TP}$, teda podiel celkového počtu platičov podľa predpovede k celkovému počtu platičov v skutočnosti. RCD môže nadobúdať hodnoty v intervale [0, ∞), pričom ideálny klasifikátor nadobúda hodnotu RCD=1.

Ďalej nech absolútna chyba klasifikátora je ξ, ktorú definujeme ako rozdiel celkového počtu platičov a celkového počtu platičov podľa predpovede:

$$\xi, = |\ (FP+TP) - (FN+TP)\ |$$

Po úprave dostaneme:

$$\xi, = |\ (FP+TP) - (FN+TP)\ | = |\ FP - FN\ |$$

Nech relatívna chyba klasifikátora je ε, ktorú definujeme nasledovne:

$$\varepsilon = |RCD\ -\ 1|$$

Po úprave dostávame:

$$\varepsilon = |RCD\ -\ 1|\ = |\frac{FP+TP}{FN+TP} - 1|\ = |\frac{FP+TP-FN-TP}{FN+TP}| = |\frac{FP-FN}{FN+TP}| = \frac{|FP-FN|}{FN+TP} = \frac{\xi}{FN+TP}$$

Relatívna chyba je absolútna chyba delená celkovým počtom hráčov, ktorí v skutočnosti zaplatili. Keďže dáta s ktorými pracujeme sú rôznej veľkosti, bude pre nás vhodné ako porovnávacie kritérium používať relatívnu chybu ε.

Pri tréningu budeme určovať optimálny prah tak, aby sme minimalizovali ε. Ak to bude možné ε bude pre tréningové dáta rovné 0. Inými slovami, budeme chcieť docieliť,

aby sa FN rovnalo FP, čo je ekvivalentné tomu, že sa precíznosť bude rovnať senzitivite. Keďže našou primárnou úlohou je identifikovať budúcich platičov, pričom dbáme aj na ich počet definujeme nasledovný pojem:

Nech Sε je ε-optimálna senzitivita, ktorá sa vypočíta ako senzitivita na základe ε-optimálneho prahu vypočítaného z tréningových dát. V prípade, že ε pre testovacie dáta nebude príliš veľké, napríklad nepresiahne hodnotu 10%, budeme kvalitu jednotlivých klasifikátorov posudzovať podľa metriky Sε pre testovacie dáta. Pre rekapituláciu, algoritmus podľa ktorého zistíme úspešnosť klasifikátora bude teda nasledovný:

1) Natrénujeme klasifikátor na tréningových dátach
2) Určíme optimálny prah. Optimálny prah je ten, pre ktorý je ε pre tréningové dáta čo najmenšie.
3) Klasifikátor z bodu 1) použijeme na testovacie dáta s optimálnym prahom z bodu 2).
4) Z matice zámen vzniknutej podľa bodu 3) vyrátame ε a Sε.
5) Ak ε z bodu 4) nepresiahne hodnotu 10%, Sε bude vyjadrovať úspešnosť klasifikátora

2.3 Dáta

Dáta potrebné k predikcii máme k dispozícii od firmy Pixel Federation.
K dispozícii je matica prediktorov *features* a matica *labels.*

2.3.1 Matica features

Matica prediktorov má približne 3,75 milióna riadkov, pričom riadky zodpovedajú údajom o jednotlivých hráčoch a má 25 stĺpcov, ktoré zodpovedajú jednotlivým prediktorom, ktoré sú nasledovné:

player_id – identifikátor hráča. Každému hráčovi je v jednej hre priradený práve 1 identifikátor.

days_in_game – premenná, ktorá určuje koľký deň od registrácie daného hráča sú k nemu zhromaždené údaje. Táto hodnota pre každého hráča nadobúda hodnoty 0,1,2,...,6. Teda ku každému hráčovi, respektíve ku každému identifikátoru prislúcha v našich dátach práve 7 riadkov.

register_platform – platforma, cez ktorú sa hráč zaregistroval.

register_month - mesiac, kedy sa hráč zaregistroval. V našich dátach vystupujú len dve hodnoty: „2019/07/01“ a „2019/08/01“

source – informácia kvôli čomu hráč prišiel, v našich dátach je táto premenná redundantná, pretože všetky hodnoty sú “marketing”

country – z ktorej krajiny hráč pochádza. Je to kategorická premenná a nadobúda 219 rôznych hodnôt.

tier – z akej skupiny krajín hráč pochádza. Možnosti sú 1,2,3,4.

today – v ktorý deň bol dataset spočítaný.

dx_pay_count – koľkokrát hráč zaplatil od svojej registrácie po days_in_game.

dx_revenue – akú sumu v peňažných jednotkách hráč zaplatil od svojej registrácie po days_in_game

d0_session_count – počet hraní (z angl. session) v deň registrácie.

d1x_session_count – počet hraní od prvého dňa od registrácia po days_in_game.

d0_session_time – kumulatívny súčet časov hraní v deň registrácie vyjadrený v časovej jednotke.

d1x_session_time – kumulatívny súčet časov hraní od prvého dňa od registrácie po days_in_game.

dx_session_days – počet dní počas ktorých sa vyskytlo aspoň jedno hranie od dňa registrácie po days_in_game.

dx_session_count – počet hraní odo dňa registrácie po days_in_game.

dx_session_time – kumulatívny súčet hraní od dňa registrácie po days_in_game.

d0_login_count – počet prihlásení v deň registrácie.

d1x_login_count – počet prihlásení od prvého dňa od registrácie po days_in_game.

dx_login_count – počet prihlásení od dňa registrácie po days_in_game.

dx_gems_count – počet transakcií s gemami od dňa registrácie po days_in_game

dx_gems_spent – počet použitých gemov od dňa registrácia po days_in_game.

version – verzia hry. V našich dátach nadobúda jedinú hodnotu.

project – názov projektu alebo hry. V našich dátach nadobúda jedinú hodnotu.

register_date – dátum, v ktorý sa hráč zaregistroval do hry.

2.3.2 Matica labels

Táto má približne 7 miliónov riadkov a 7 stĺpcov, pričom opäť riadky prislúchajú jednotlivým hráčom a stĺpce vyjadrujú prediktory, ktoré sú nasledovné:

player_id, days_in_game, version, register_month – rovnaké prediktory ako v kapitole Matica features.

register_time – čas, kedy sa hráč zaregistroval.

dy_pay_count – počet platieb, ktoré hráč uskutočnil od dňa days_in_game+1 až 28.

dy_revenue – suma platieb v peňažných jednotkách, ktoré hráč uskutočnil od dňa days_in_game+1 až 28.

Matice *features* a *labels* spojíme podľa *player_id* a *days_in_game*. Pre jednoduchosť budeme k analýze brať do úvahy len tie riadky, v ktorých premenná *days_in_game*=6. Tento riadok obsahuje najviac informácií o danom hráčovi. Ďalej túto maticu očistíme o redundantné premenné, a to: *register_platform, register_month, source, today, version, project, register_date, register_time*. Premennú *country* vynecháme z dôvodu, že máme premennú *tier*, ktorá je jej zjednodušenou verziou. Tým pádom nebudeme mať ani jednu kategorickú premennú, ktorá by mohla robiť problémy vo výpočte. Ďalej odstránime ešte premenné *player_id* a *days_in_game*, *dy_revenue* ktoré nie sú pre výpočet potrebné.

Dostali sme jednu spoločnú maticu so všetkými potrebnými dátami s pracovným názvom *t6* (z angl. total 6), ktorá má 536 835 riadkov a 16 stĺpcov. Ďalej vytvoríme dve kópie matice *t6*, a to:

t6f (total 6 factorized) - premenná *dy_pay_count* bude nadobúdať len dve kategoriálne hodnoty: „True“ ak v *t6* je hodnota *dy_pay_count* >0 alebo „False“ ak v *t6* je *dy_pay_count*=0

t6s (total 6 signum) – premenná dy_pay_count bude nadobúdať len hodnoty 0, ak v *t6* hodnota *dy_pay_count*>0 alebo 1 inak.

Ďalej sme dáta rozdelili na tréningovú a testovaciu časť. Náhodne sme vybrali 75% pôvodných dát a uložili do matíc *trainingf* a *trainings.* Zvyšné dáta sme uložili do matíc *testingf* a *testings.* Po výbere sme skontrolovali, či je pomer súčtu „True“ hodnôt k všetkým hodnotám približne rovnaký v tréningovej a testovacej vzorke. Tieto dve matice sa líšia jedine v premennej *dy_pay_count,* a rozlišujeme ich jedine pri práci v R-ku, keďže niektoré metódy vyžadujú, aby modelovaná premenná bola numerická a niektoré, aby bola kategorická. V tejto práci budeme pracovať s jednotným názvom pre obe matice, s názvom *training*, respektíve *testing*.

Okrem prvého rozdelenia dát na tréningovú a testovaciu časť sme urobili ešte jedno rozdelenie. Intuitívne sme sa domnievali, že v niektorých prípadoch môže byť pre natrénovanie klasifikátora vhodné, ak v tréningovej vzorke bude približne rovnaký počet z oboch klasifikačných tried, pretože rozdiely medzi dvomi triedami by mohli byť klasifikátorom ľahšie rozpoznateľné. V dátach *t6f* existuje 5 292 takých riadkov, ktorých hodnota *dy_pay_count* je „True“. Rozhodli sme sa preto vytvoriť maticu *trainBalanced,* ktorá má 6 000 riadkov, pričom práve pri 3 000 riadkoch sa hodnota *dy_pay_count*=“True“ alebo 1, a práve v 3 000 prípadoch sa rovná “False“ alebo 0. Testovacie dáta *testBalanced* sú všetky ostatné dáta z matice *t6* okrem tých, ktoré sú už v *trainBalanced.*

2.4 Výsledky

2.4.1 Logistická regresia

Prvý binárny klasifikátor sme vyrobili pomocou logistickej regresie. Na trénovacích dátach sme pomocou funkcie glm() v softvéri R našli optimálne parametre logistickej regresie. Pomocou funkcie predict() sme odhadli pravdepodobnosť premennej *dy_pay_count* pre testovaciu vzorku. Keďže výsledkom tejto predikcie je číslo, tento výsledok sme upravili tak, aby sme dostali jedine hodnoty „True“ alebo „False“, respektíve 1 alebo 0. Takéto binárne rozdelenie sme dostali stanovením optimálneho prahu (z angl. Threshold). Prah je také číslo, ktoré predikcie rozdeľuje na True alebo False. Ak je predikcia nad týmto prahom, hráč sa klasifikuje ako platič, inými slovami premanná dy_pay_count bude predikovaná ako 1-ka alebo True, a ak je predikcia pod týmto prahom tak naopak. Optimálny prah je také číslo, ktoré minimalizuje ε.

Tabuľka 2: Výsledky logistickej regresie na trénovacích dátach *training* s optimálnym prahom

	Realita: False	Realita: True
Predikcia: False	396 245	2 413
Predikcia: True	2 413	1 556

Optimálny prah: 0,1724 ; $\varepsilon = 0$; Sε = 39,20%

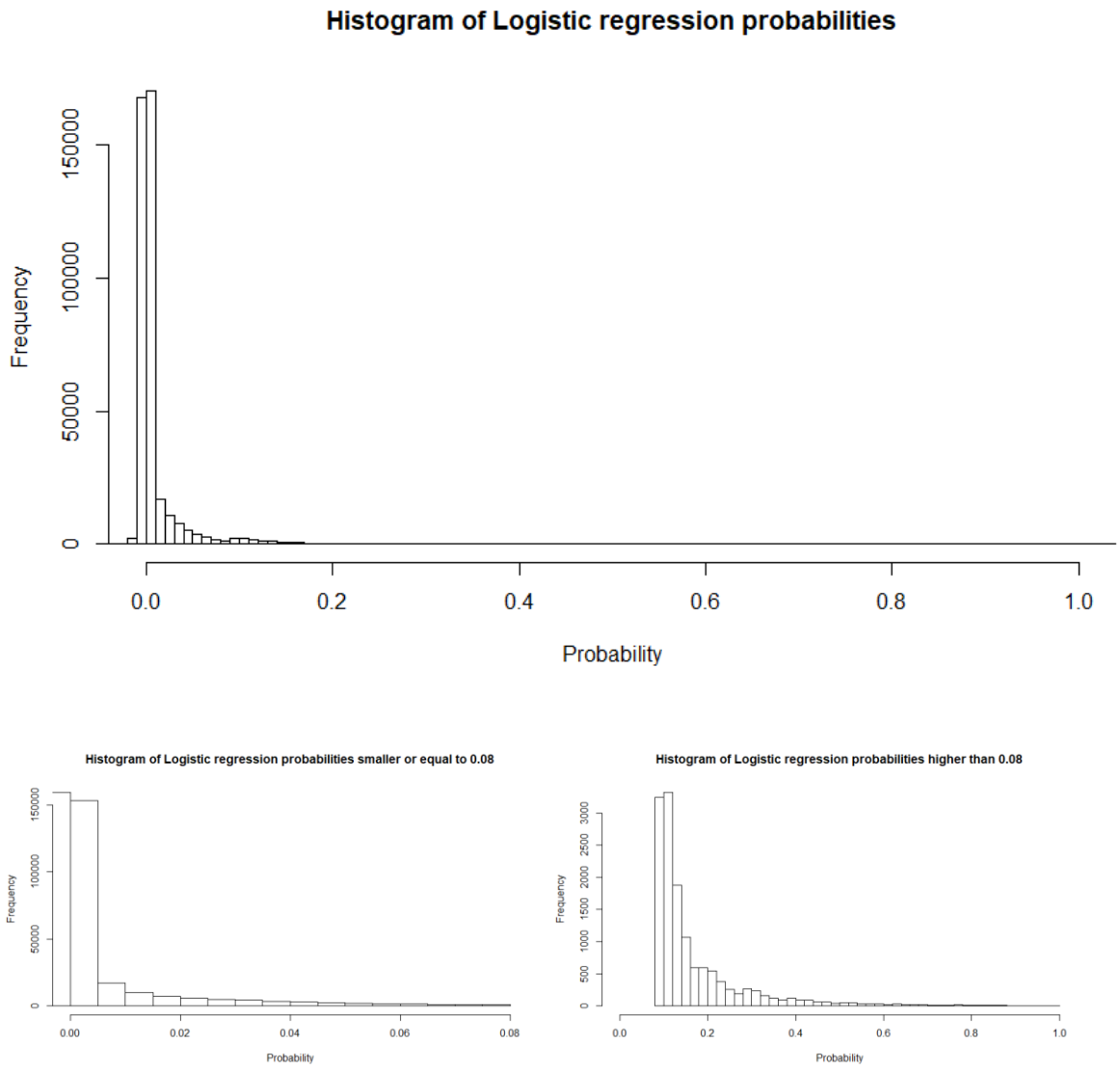
V tabuľke 2 môžeme vidieť, že chybu ε sa podarilo stlačiť na nulu, pretože hodnoty FN a FP sa rovnajú. Hodnota optimálneho prahu je 0,1724. Na základe tohto optimálneho prahu môžeme klasifikátor otestovať na testovacích dátach.

Tabuľka 3: Výsledky logistickej regresie na testovacích dátach *testing* s optimálnym prahom získaným z tréningových dát

	Realita: False	Realita: True
Predikcia: False	132 043	793
Predikcia: True	842	499

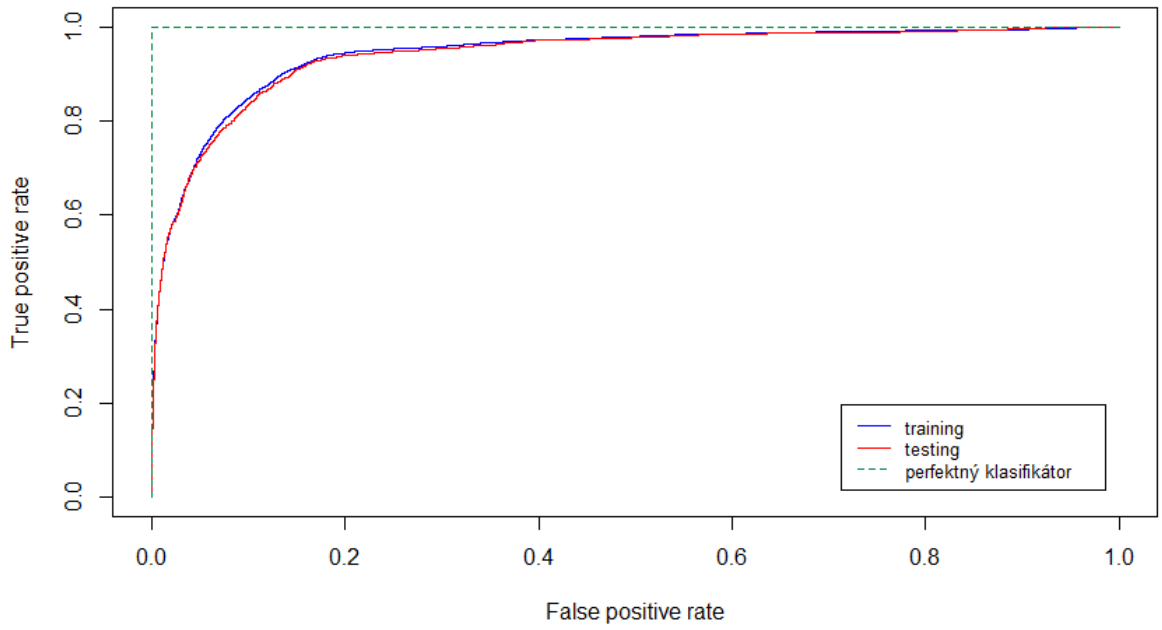
$\varepsilon = \frac{49}{1292} \doteq 3{,}8\%$; Sε=40,06%

V tabuľke 3 môžeme vidieť reálny výsledok prvého klasifikátora. Tu sme neočakávali nulovú chybu ε, pretože ide o testovacie dáta, na ktorých neprebiehal tréning. ε je veľmi blízko k číslu 0, čo je vyhovujúce z toho hľadiska, že klasifikátor relatívne presne odhadol počet platičov. Sε je na úrovni 40%, čo znamená, že z hráčov, ktorí v skutočnosti zaplatili klasifikátor odhalil 4-och z 10.



Obr. 5: Histogramy pravdepodobností pre logistickú regresiu

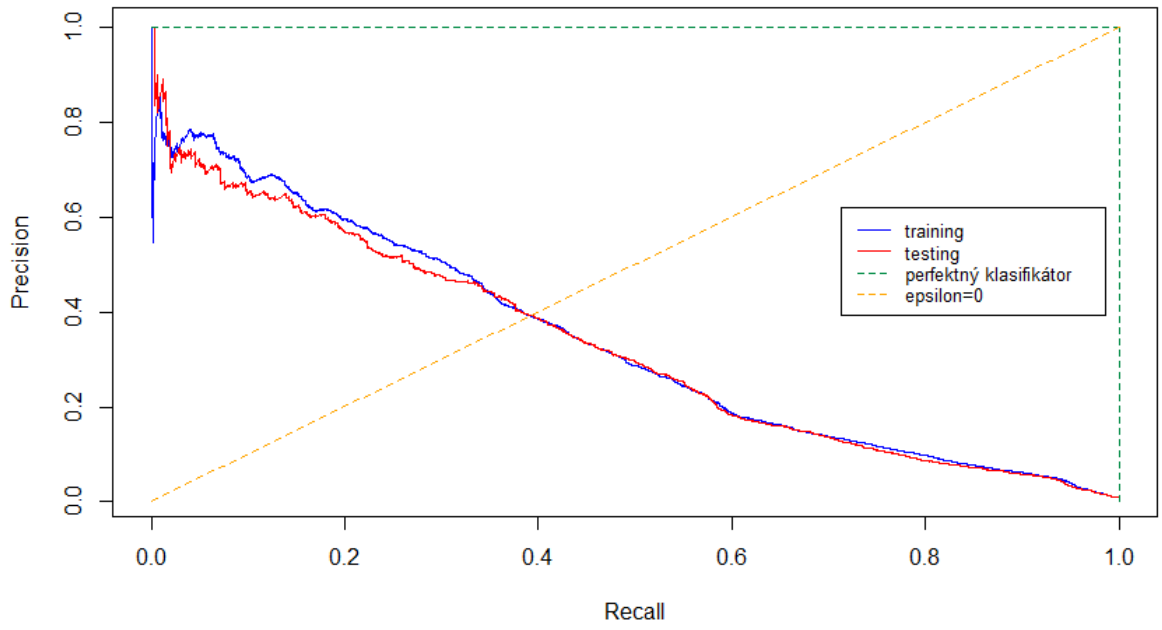
Na obrázku 5 môžeme vidieť histogram pre distribúciu pravdepodobností, ktoré sme pomocou logistickej regresie dostali pre tréningové dáta. V ideálnom prípade by platiči a neplatiči boli v kontexte obrázku dva oddeliteľné „kopčeky“. Najhoršia možná distribúcia by bol zrejme jeden veľký strmý kopec, pretože potom by malá zmena v optimálnom prahu znamenala veľké zmeny v k prahu prislúchajúcej matici zámen. Pozorujeme, že v našom prípade je drvivá väčšina pravdepodobností v blízkosti 0% a histogram má klesajúcu tendenciu, čo je niečo medzi optimálnym a najhorším scenárom.



Obr. 6: ROC krivky pre logistickú regresiu

Na obrázku 6 môžeme vidieť ROC krivky pre tréningové a testovacie dáta skonštruované na základe prvého klasifikátora. Jedna krivka reprezentuje všetky možné matice zámen klasifikátora na jedných dátach v závislosti od rôznych prahov. Čím vyššie sa krivka nachádza, tým je klasifikátor v teórii lepší.

Zelenou prerušovanou čiarou je znázornený perfektný klasifikátor, ktorý pre akýkoľvek prah predikuje všetkých hráčov správne. Modrá krivka reprezentuje tréningovú vzorku *training*, červená krivka testovaciu vzorku *testing*. Obe krivky sú zdanlivo blízko ku krivke perfektného klasifikátora, čo je dané konzistenciou dát. Vzhľadom na veľké množstvo neplatičov, ktoré klasfikátor často predikuje správne je väčšina hráčov správne klasifikovaná, čo sa v konečnom dôsledku preukáže na blízkosti ROC krivky k perfektnému klasifikátoru.



Obr. 7: P-R krivky pre logistickú regresiu

V prípade P-R kriviek platí podobný princíp ako v prípade kriviek ROC. Jeden bod krivky reprezentuje jednu maticu zámen v závislosti od použitého prahu a čím vyššie sa daná krivka nachádza, tým je v teórii klasifikátor reprezentovaný danou krivkou presnejší. Zelenou prerušovanou čiarou je znázornený perfektný klasifikátor, ktorý pre akýkoľvek prah predikuje všetkých hráčov správne. Modrá krivka reprezentuje tréningovú vzorku *training*, červená krivka testovaciu vzorku *testing.*

Oranžovou prerušovanou čiarou je znázornená časť pomyselnej priamky $\varepsilon = 0$. Tam kde táto čiara pretína modrú krivku je bod ktorý reprezentuje maticu zámen z tabuľky 2. Bod, kde oranžová čiara pretína červenú nie je znázornený ani v jednej z tabuliek, ale vzhľadom na to, že pre tabuľku 3 je $\varepsilon \sim 0$, táto tabuľka je reprezentovaná bodom, ktorý sa nachádza v blízkom okolí bodu pretnutia. Tento bod by sme na obrázku 7 vedeli nájsť pomocou nami vyrátaného $S\varepsilon$, pretože x-ová os reprezentuje práve senzitivitu (na obr. názov Recall).

Tiež si na obrázku 7 môžeme všimnúť, že červená krivka je na väčšine miest pod modrou, čo je prirodzené, keďže klasifikátor sme trénovali na tréningových dátach (modrá krivka) a teda by mal lepšie fungovať na dátach na ktorých bol trénovaný, než na akýchkoľvek iných dátach. Na druhú stranu krivky sú k sebe dosť blízko a takmer sa

kopírujú, čo značí, že tréningové aj testovacie dáta sú si veľmi podobné. Našim snom je nájsť taký klasifikátor, ktorý bude pre testovacie dáta čo najbližší k perfektnému klasifikátoru. Z obrázku 7 to vidno lepšie, než z obrázku 6, že tento klasifikátor má ešte ďaleko k nášmu snu.

2.4.2 Bayesov klasifikátor

Naivný Bayesov klasifikátor sme natrénovali v softvéri R pomocou funkcie train(), pričom parameter method sme zvolili "nb". Výpočet bol značne dlhší, než v logistickej regresii, pre dáta *training* trvalo dlho najmä natrénovanie klasifikátora.

Tabuľka 4: Výsledky Bayesovho klasifikátora na tréningových dátach *training* s optimálnym prahom

	Realita: False	Realita: True
Predikcia: False	397 401	1 257
Predikcia: True	1 257	2 712

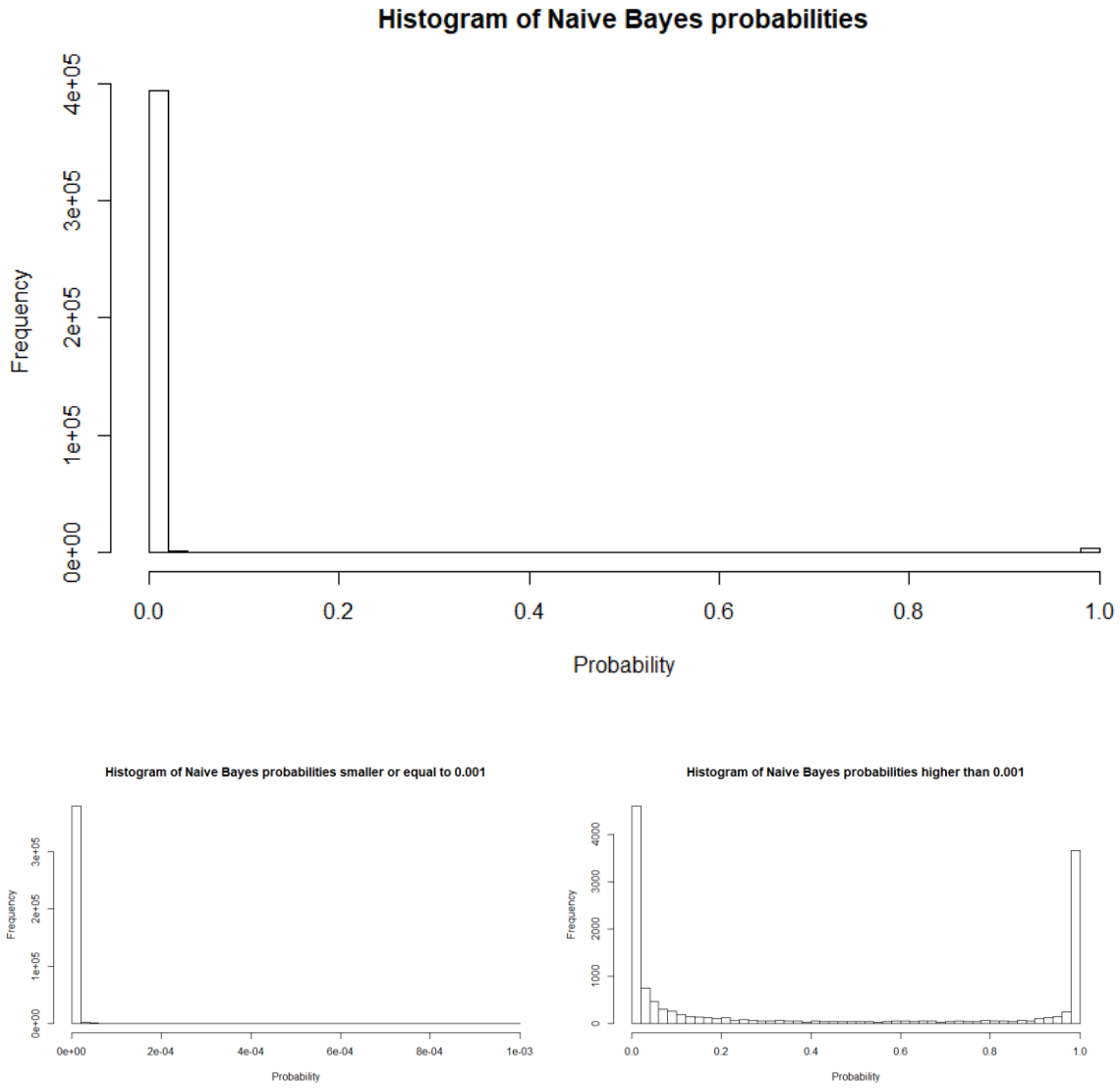
Optimálny prah: 0,953 ; $\varepsilon = 0$; $S\varepsilon = 68{,}32\%$

Tabuľka 5: Výsledky Bayesovho klasifikátora na testovacích dátach *testing* s optimálnym prahom získaným z tréningových dát

	Realita: False	Realita: True
Predikcia: False	132 457	422
Predikcia: True	458	901

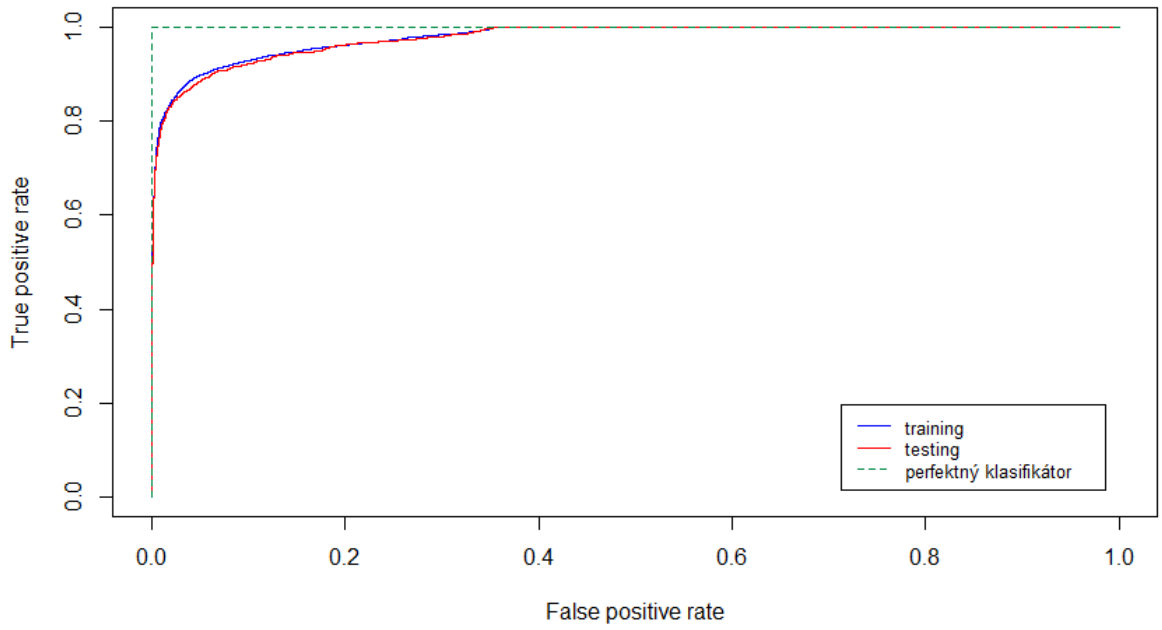
$\varepsilon = \frac{36}{1323} \doteq 2{,}7\%$; Sε=68,10%

V tabuľke 4 môžeme vidieť, že pre tréningové dáta *training* sa opäť podarilo stlačiť ε na nulu. Za platičov bude klasifikátor predpokladať tých hráčov, ktorým odhadne pravdepodobnosť zaplatenia vyššiu, než je optimálny prah, teda 95,3%. Po aplikácii tohto klasifikátora na testovacie dáta dostávame priaznivé výsledky, keďže ε je malé, rovné 2,7% a taktiež Sε je vyššie oproti logistickej regresii a má hodnotu 68,10%.

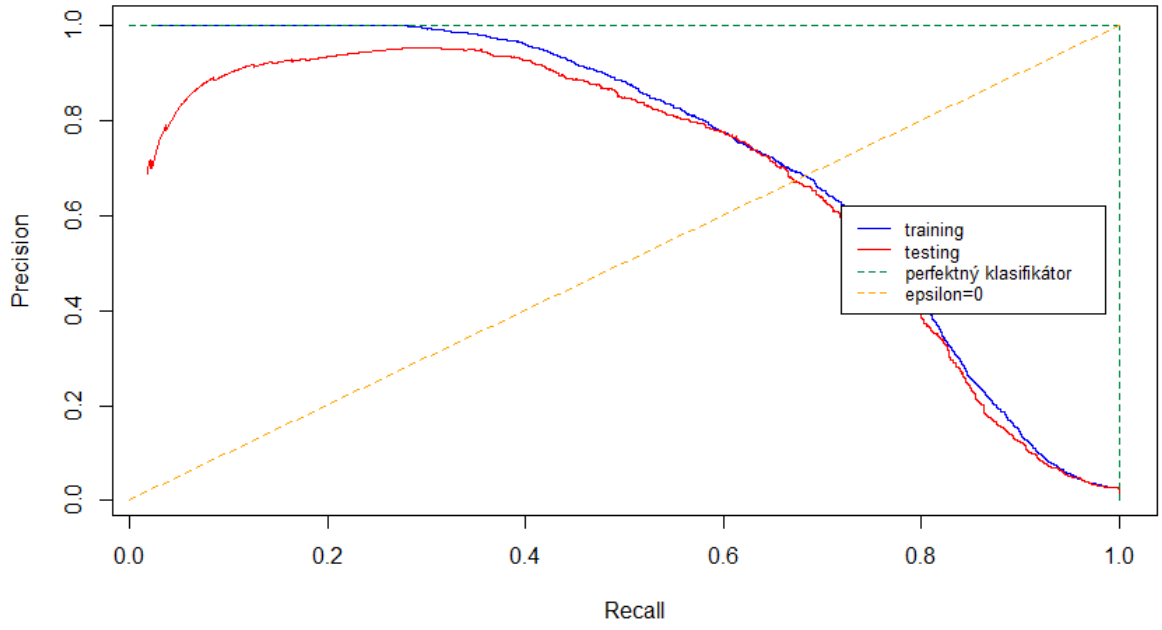


Obr. 8: Histogramy pravdepodobností pre Bayesov klasifikátor

Na histograme pravdepodobností pre Bayesov klasifikátor, znázornený na obrázku 8 pozorujeme dva kopce. Jeden veľký kopec v okolí 0 a jeden malý kopec v okolí 1. Hráčov, pre ktorých Bayesov klasifikátor vyhodnotil pravdepodobnosť, že zaplatia menšiu než 0,1% je 97,5%. Zvyšných 2,5% má túto pravdepodobnosť väčšiu, než 0,1%. Drvivej väčšine hráčov Bayesov klasifikátor odhadol pravdepodobnosť stať sa budúcim platičom veľmi blízku k číslu 0, niektorým priradil hodnoty rádu 10^{-30}. Na histograme, kde je pravdepodobnosť vyššia, než 0,1% pozorujeme dva oddeliteľné, približne rovnako veľké kopce. Približne 3000 hráčom predikuje Bayesov klasifikátor pravdepodobnosť zaplatenia veľmi blízku k číslu 1, aj preto bol optimálny prah pre tento klasifikátor (0,953) vyšší, než pre logistickú regresiu (0,17). Distribúcia pravdepodobností získaná z Bayesovho klasifikátora je z grafického hľadiska blízko k ideálnej distribúcii pravdepodobností.



Obr 9: ROC krivky pre Bayesov klasifikátor



Obr. 10: P-R krivky pre Bayesov klasifikátor

Na obrázkoch 9 a 10 si môžeme všimnúť, že ROC aj P-R krivky pre tréningové aj testovacie dáta sú bližšie k perfektnému klasifikátoru, než krivky pre logistickú regresiu.

Červená P-R krivka, reprezentujúca testovacie dáta sa opäť nachádza na takmer celom grafe pod modrou krivkou reprezentujúcou tréningové dáta.

2.4.3 Metóda oporného bodu

Klasifikátor podľa metódy oporného bodu sme v softvéri R natrénovali pomocou funkcie svm(). Tento algoritmus sme trénovali len na časti tréningových dát, konkrétne na 50 000 hráčoch, čo je približne $\frac{1}{8}$ celej tréningovej vzorky. Aj napriek tomu trval tréning najdlhšie zo všetkých metód, približne 2 hodiny. Tréningový proces pre kompletné tréningové dáta neskonvergoval ani po 10-kach hodín.

Tabuľka 6: Výsledky metódy oporného bodu na časti tréningových dátach *training* s optimálnym prahom

	Realita: False	Realita: True
Predikcia: False	49 102	216
Predikcia: True	216	466

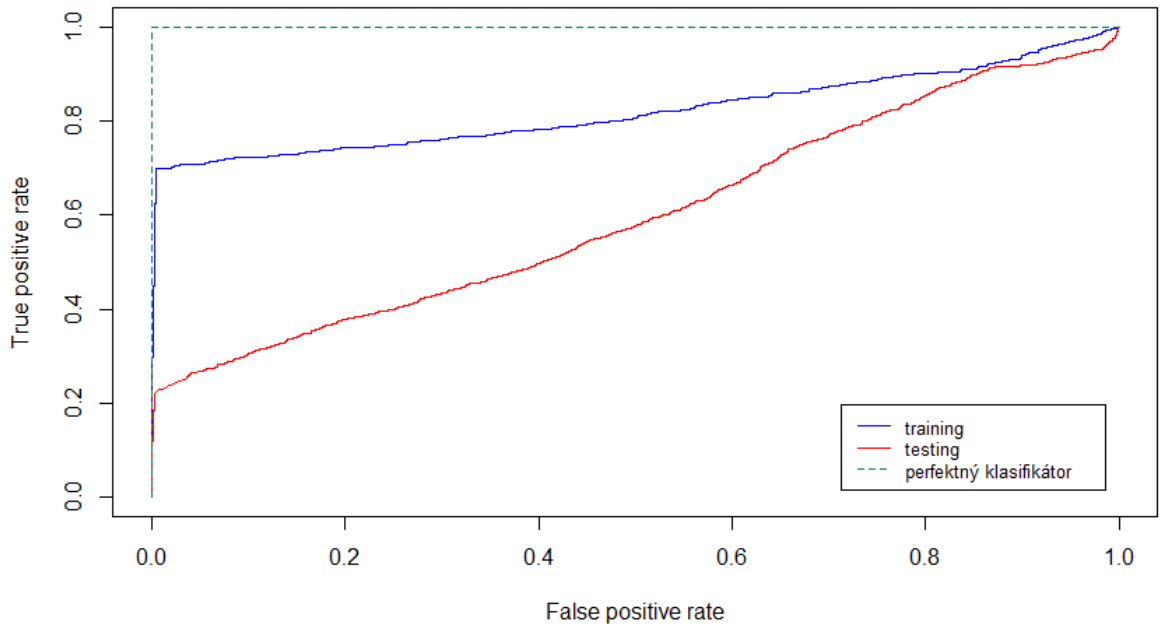
Optimálny prah: 0,0116 ; $\varepsilon = 0$; $S\varepsilon = 68{,}33\%$

Tabuľka 7: Výsledky metódy oporného bodu na testovacích dátach *testing* s optimálnym prahom získaným z tréningových dát

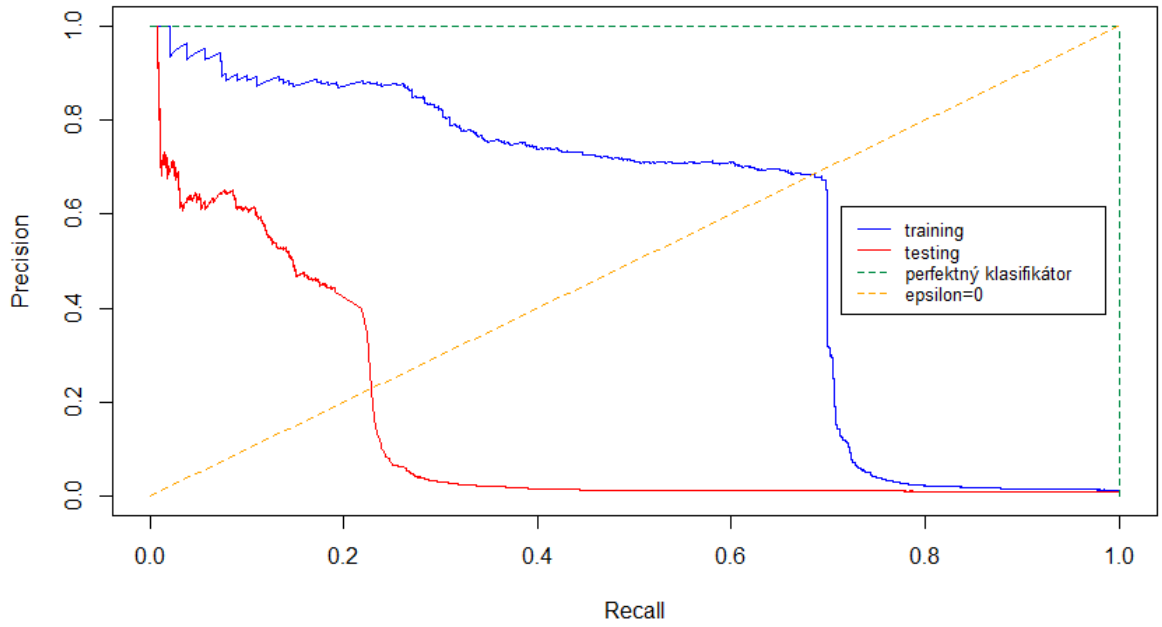
	Realita: False	Realita: True
Predikcia: False	18 328	124
Predikcia: True	114 557	1 199

$\varepsilon = \frac{114\ 433}{1323} \doteq 8650\%$; $S\varepsilon$=90,63%

V tabuľke 7 si môžeme všimnúť, že $S\varepsilon$=90,63% je zatiaľ najväčšia hodnota ε-optimálnej senzitivity akú sme doteraz dostali. Avšak z tabuľky tiež jasne vidno, že klasifikátor nemôžeme považovať za dostatočne dobrý, keďže jeho celková presnosť je len približne 17%. Pri definícii $S\varepsilon$ sme spomínali, že táto metrika je vhodná jedine v prípade, že ε je dostatočne malé, konkrétne menšie, než 10%, čo v tomto prípade nie je pravda a preto tento klasifikátor nepovažujeme za dobrý.



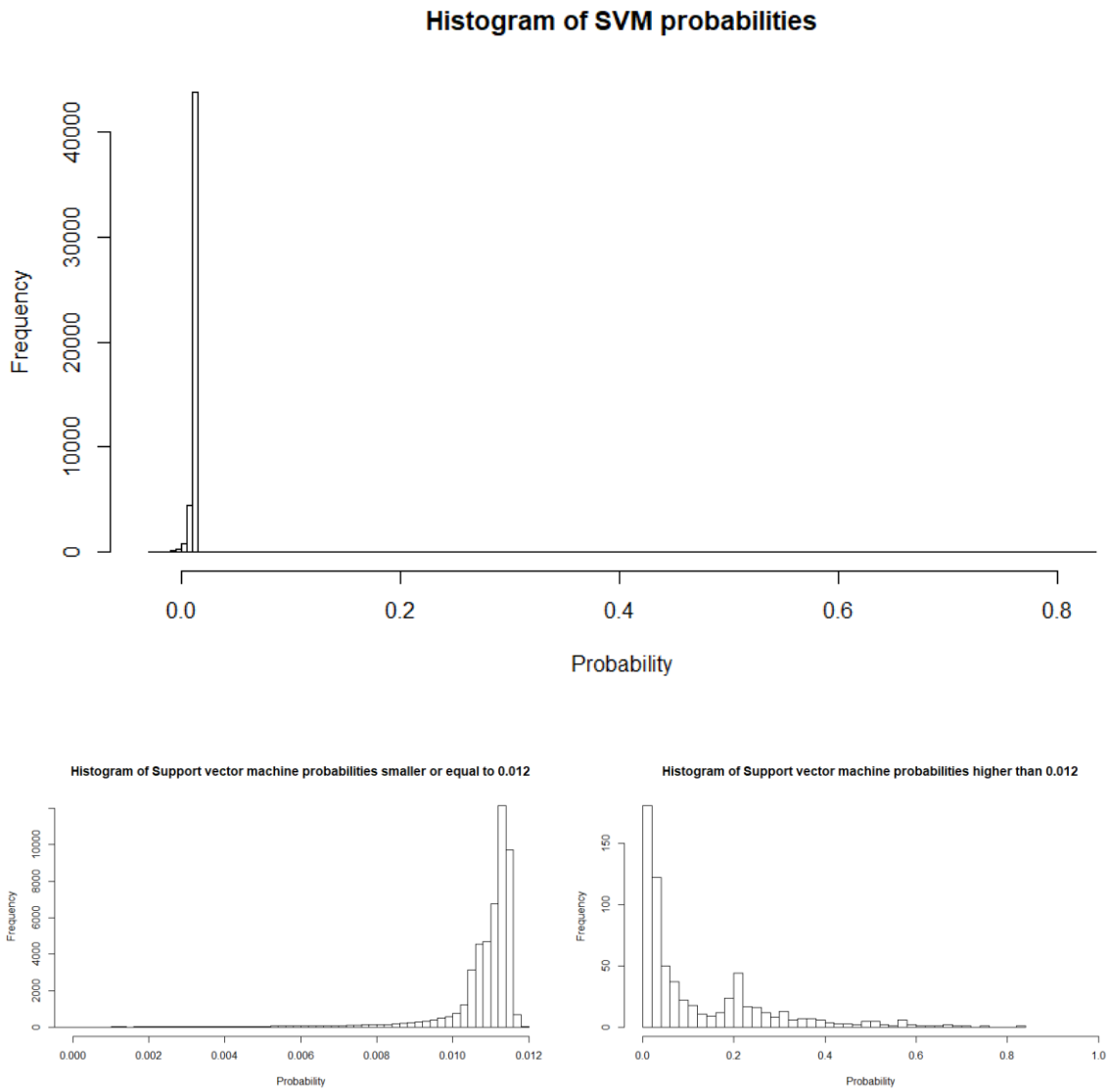
Obr. 11: ROC krivky pre metódu oporného bod



Obr. 12: P-R krivky pre metódu oporného bodu

Na obrázkoch 11 a 12 môžeme vidieť, že klasifikátor, ktorý sme na základe metódy oporného bodu skonštruovali funguje omnoho nepresnejšie na testovacích dátach, než na

tréningových, čo je znak pretrénovania modelu. „Veľké pády“ P-R kriviek sú sposobené tým, že veľa objektov (hráčov) má podľa modelu pravdepodobnosť byť platičom veľmi podobnú alebo rovnakú.



Obr. 13: Histogramy pravdepodobností pre metódu oporného bodu

Na obrázku 13 vidíme histogram pravdepodobností pre tréningové dáta určený metódou oporného bodu. Tento histogram vysvetľuje, prečo bol tento klasifikátor neúspešný na testovacích dátach. Drvivá väčšina hodnôt pravdepodobností sa totiž koncentruje v okolí hodnoty optimálneho prahu, čiže malá zmena optimálneho prahu alebo malá zmena dát bude znamenať veľkú zmenu v matici zámen, dôsledkom čoho je nepredvídateľné $S\varepsilon$ a ε pre akékoľvek testovacie dáta, ktoré nie sú totožné alebo veľmi podobné s tréningovými dátami. Hráčov, ktorým klasifikátor priradil pravdepodobnosti

výrazne väčšie, než je optimálny prah je príliš málo. Na histograme znázorňujúcom pravdepodobnosti väčšie, než 0,12 vidíme kopček okolo hodnoty 0,2. Ten je však príliš malý, pretože celý histogram obsahuje len pár stovák hráčov, čo je približne 0,1% všetkých hráčov. My však zo štruktúry tréningových dát vieme, že platiacich hráčov je približne 1%. Keď má klasifikátor takúto distribúciu pravdepodobností, tušíme, že sa niečo muselo pokaziť a takýto klasifikátor by sme prvoplánovo nepovažovali za dobrý ani v prípade, keby hodnoty ε a $S\varepsilon$ pre testovacie dáta boli dobré.

2.4.4 Klasifikačné stromy

Pre konštrukciu klasifikačných stromov sme používali funkciu rpart() v softvéri R, ktorá vie automaticky robiť binárnu klasifikáciu. Pre reguláciu hĺbky stromu sme menili parameter zložitosti *cp* (z angl. complexity parameter), ktorý zabezpečí to, že strom sa nebude ďalej vetviť, pokiaľ sa týmto vetvením nezabezpečí zníženie nedostatku modelu o danú hodnotu *cp*. Ako tréningové dáta sme použili dáta *training*, ktoré sme testovali na dátach *testing*.

Tabuľka 8: Prehľad vybraných metrík klasifikačných stromov v závislosti od parametra cp

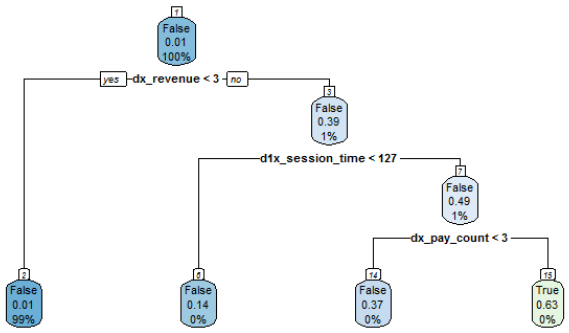
Č.stromu	cp	Sε-test	ε-test	Sε-train	ε-train	Opt. prah
1	10^{-4}	35,14%	1,4%	50,69%	0,3%	[0,158;0,158]
2	3.10^{-4}	37,87%	1,7%	39,58%	0,2%	[0,007;0,062]
3	5.10^{-4}	37,71%	1,4%	39,53%	0,02%	[0,067;0,076]
4	7.10^{-4}	37,79%	2,5%	39,48%	0,8%	[0,143;0,144]
5	10^{-3}	38,25%	3,1%	39,50%	1,3%	[0,143;0,144]
6	2.10^{-3}	38,62%	3,6%	39,53%	1,8%	[0,077;0,144]
7	3.10^{-3}	38,78%	3,9%	39,56%	2,1%	[0,067;0,144]
8	5.10^{-3}	38,93%	4,2%	39,58%	2,5%	[0,007;0,144]
9	10^{-2}	38,93%	4,2%	39,58%	2,5%	[0,007;0,144]
10	3.10^{-2}	0%	100%	0%	100%	[0,010;1]

V tabuľke 8 sa nachádza niekoľko vybraných charakteristík pre 9 rôznych stromov skonštruovaných na základe 10 rôznych hodnôt parametra cp. Najdôležitejšou charakteristikou je Sε-test teda senzitivita pre testovacie dáta na základe ktorej určujeme úspešnosť klasifikátora. ε-test je relatívna chyba pre testovacie dáta a hovorí nám ako veľmi sa klasifikátor pomýlil v predikcii počtu hráčov. Sε-train je senzitivita pre tréningové dáta,

ε-train je relatívna chyba pre tréningové dáta. Optimálny prah je vyrátaný pre tréningové dáta. Posledné 3 premenné nijako neurčujú úspešnosť, môžu nám však pomôcť pri porozumení celkového fungovania modelu, preto sme sa ich rozhodli do tabuľky zahrnúť.

ε-test je menšie, než 10% pre prvých 9 stromov, preto ich môžeme považovať za dobré. Strom č. 10 má však túto hodnotu väčšiu, než 10%, čo indikuje, že s ním niečo nie je v poriadku. Tento strom sa skladá z jediného uzla, ktorý je zároveň koreovým aj terminálnym, ktorý teda obsahuje aj všetkých hráčov. Klasifikátor má v tomto prípade len 2 možnosti, buď to všetkých hráčov klasifikovať ako platičov alebo ako neplatičov. Taký klasifikátor je absolútne zlý. Z tabuľky 8 si ďalej môžeme všimnúť spojitosť medzi cp a Sε-test . Zdá sa, že čím väčšie cp tým väčšia je aj senzitivita. Toto nám naznačuje, že najlepšie klasifikujú tie stromy, pre ktoré je parameter cp veľký, teda najjednoduchšie stromy. Premenná $S\varepsilon - \text{train}$ sa pohybuje okolo konštantnej hodnoty, okrem stromu č. 1, pre ktorý je táto hodnota vyššia, na druhú stranu je práve pri tomto strome Sε-test nižšia, než pre ostatné stromy. Pre tento zložitý strom to indikuje pretrénovanie sa. Ešte si môžeme si všimnúť, že pre strom číslo 8 a 9 dostávame rovnaké hodnoty ε a Sε , či už pre tréningové alebo testovacie dáta a takisto rovnaký optimálny prah. Dôvodom je, že ide o jeden a ten istý strom. V ostatných prípadoch bola zmena parametra cp voči susedným stromom dostatočne veľká na to, aby sme nedostali ďalšie dva identické klasifikačné stromy.

Nasledujúca časť tejto kapitoly obsahuje podrobné výsledky pre 3 rôzne klasifikačné stromy: Z tabuľky 8 sú to stromy č. 1, č.5 a č.9, ktoré pre lepšiu prehľadnosť budeme nazývať jednoduchý, stredne zložitý a zložitý strom.



Obr. 14: Jednoduchý klasifikačný strom trénovaný na dátach *training* s parametrom *cp*=10

Tabuľka 9: Výsledky jednoduchého klasifikačného stromu na tréningových dátach *training* s optimálnym prahom

	Realita: False	Realita: True
Predikcia: False	396 159	2 398
Predikcia: True	2 499	1 571

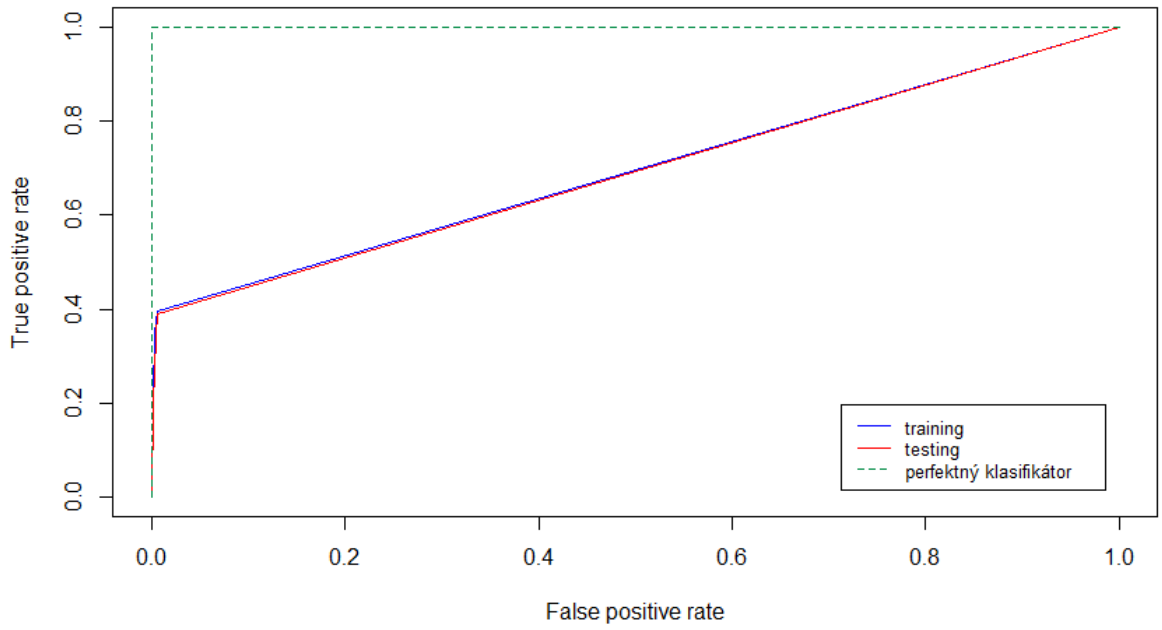
Optimálny prah $\in [0{,}007\ ;\ 0{,}144]$; $\varepsilon \doteq 2{,}5\%$; $S\varepsilon = 39{,}58\%$

Pri klasifikačných stromoch sa nám nepodarilo dostať $\varepsilon = 0$ pre tréningové dáta, čoho dôvodom je ich konštrukcia. Napríklad v prípade jednoduchého klasifikačného stromu existujú len 4 rôzne pravdepodobnosti pre hráčov byť platičom, ktoré sú dané v závislosti od toho v ktorom terminálnom uzli sa daný hráč nachádza, znázornené na Obrázku 14. V prvom terminálnom uzli sa nachádza 99% hráčov a klasifikátor im predpovedá pravdepodobnosť byť platičom približne 0,007. Hráči, ktorí spadajú do druhého uzla majú pravdepodobnosť byť platičom približne 0,144. V treťom uzle je to približne 0,36 a v štvrtom 0,63. Z existencie iba 4 rôznych pravdepodobností vyplýva existencia len 5-tich rôznych matíc zámen pre tento model. Matica zámen s najmenším ε je práve Tabuľka 9. Ďalším dôsledkom je veľká voľnosť optimálneho prahu, ktorá však nijako neovplyvní výsledky na testovacích dátach.

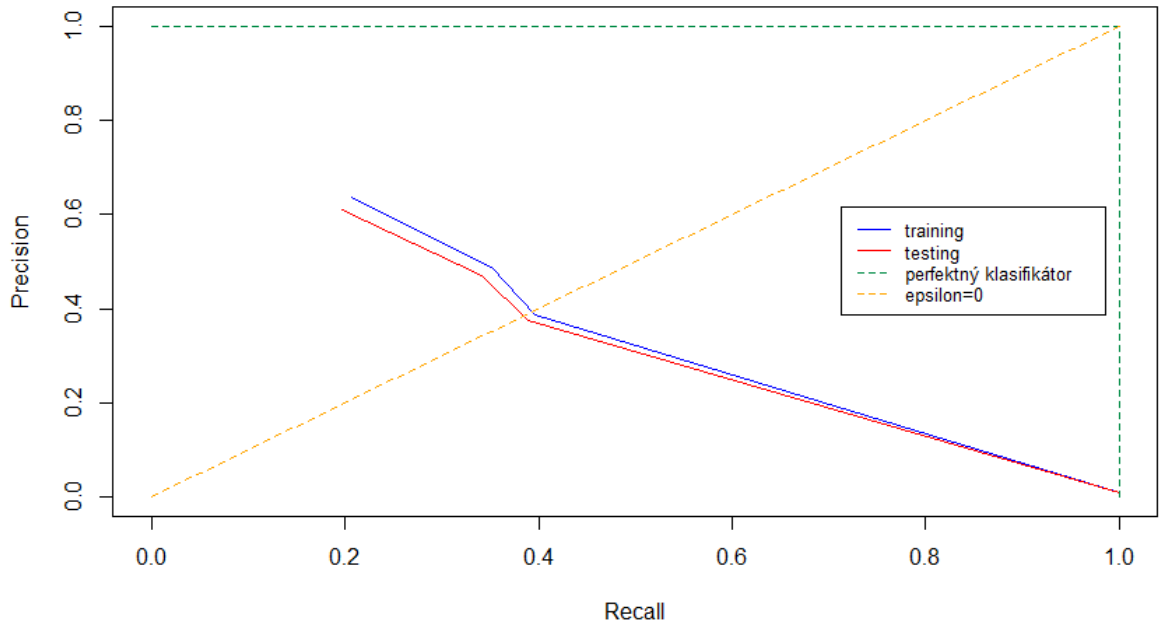
Tabuľka 10: Výsledky jednoduchého klasifikačného stromu na testovacích dátach *testing* s optimálnym prahom získaným z tréningových dát

	Realita: False	Realita: True
Predikcia: False	132 021	808
Predikcia: True	864	515

$\varepsilon = \frac{56}{2292} \doteq 4{,}2\%$; $S\varepsilon$=38,93%

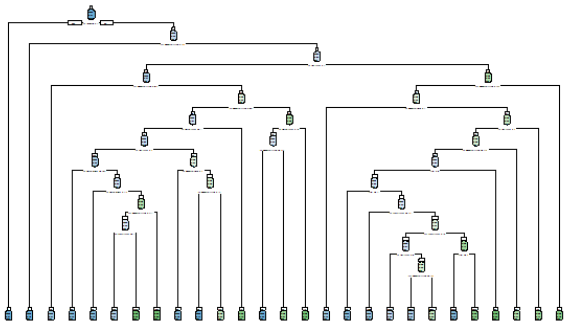


Obr. 15: ROC krivky pre jednoduchý klasifikačný strom



Obr. 16: P-R krivky pre jednoduchý klasifikačný strom

Na obrázkoch 15 a 16 vidíme, že bodov zlomu všetkých kriviek je málo, práve kvôli tomu, že možných matíc zámen je málo.



Obr. 17: Stredne zložitý klasifikačný strom trénovaný na dátach *training* s parametrom $cp=10^{-3}$

Tabuľka 11: Výsledky stredne zložitého klasifikačného stromu na tréningových dátach *training* s optimálnym prahom

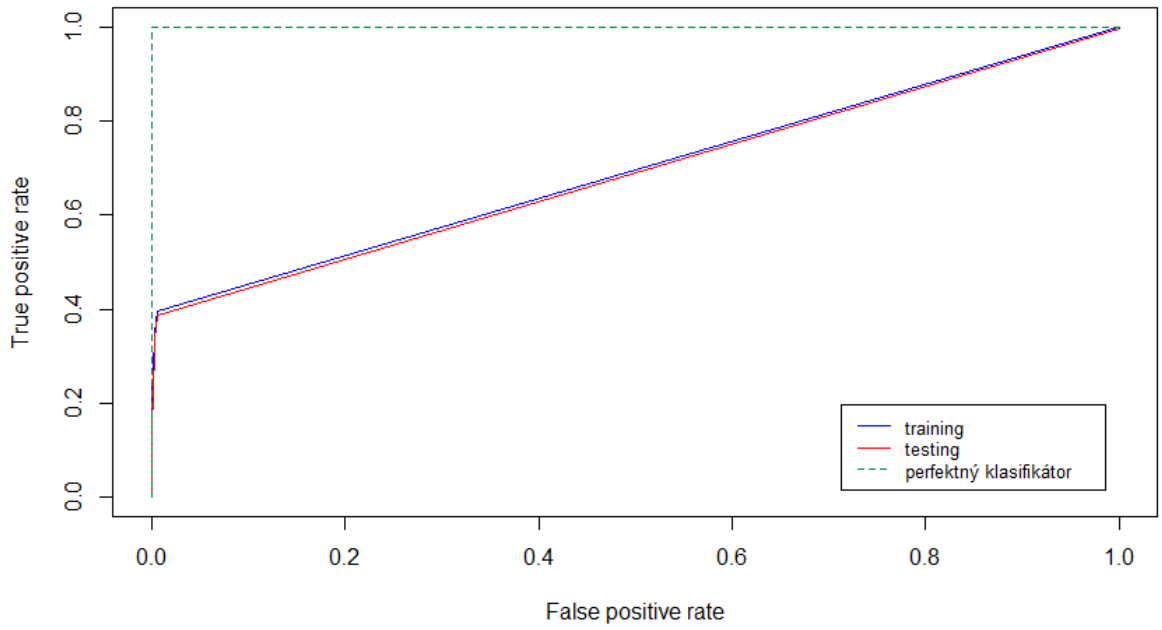
	Realita: False	Realita: True
Predikcia: False	396 207	2 401
Predikcia: True	2 451	1 568

Optimálny prah $\in [0{,}143 ; 0{,}144]$; $\varepsilon \doteq 1{,}3\%$; $S\varepsilon = 39{,}50\%$

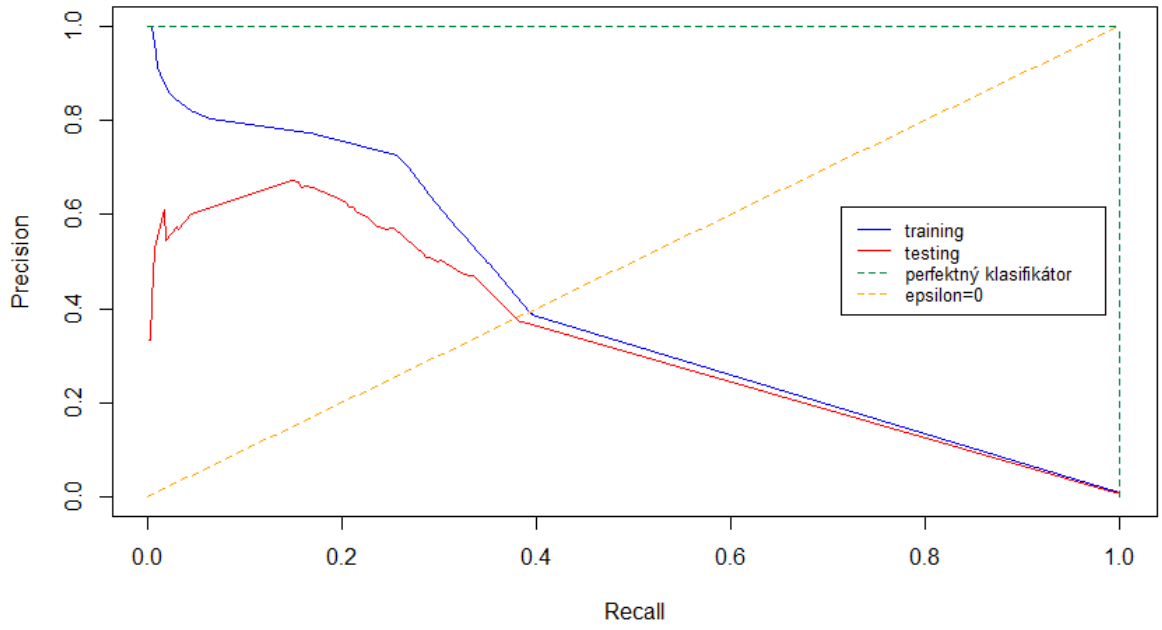
Tabuľka 12: Výsledky stredne zložitého klasifikačného stromu na testovacích dátach *testing* s optimálnym prahom získaným z tréningových dát

	Realita: False	Realita: True
Predikcia: False	132 027	817
Predikcia: True	858	506

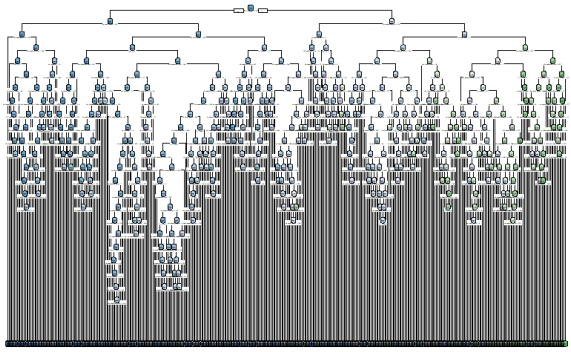
$\varepsilon = \frac{56}{2292} \doteq 3{,}1\%$; $S\varepsilon = 38{,}25\%$



Obr. 18: ROC krivky pre stredne zložitý klasifikačný strom



Obr.19: P-R krivky pre stredne zložitý klasifikačný strom



Obr. 20: Zložitý klasifikačný strom trénovaný na dátach *training* s parametrom *cp*=10^{-4}

Zložitý klasifikačný strom má príliš veľa terminálnych uzlov na to, aby ho R-ko dokázalo celý vykresliť (obrázok 20).

Tabuľka 13: Výsledky zložitého klasifikačného stromu na tréningových dátach *training* s optimálnym prahom

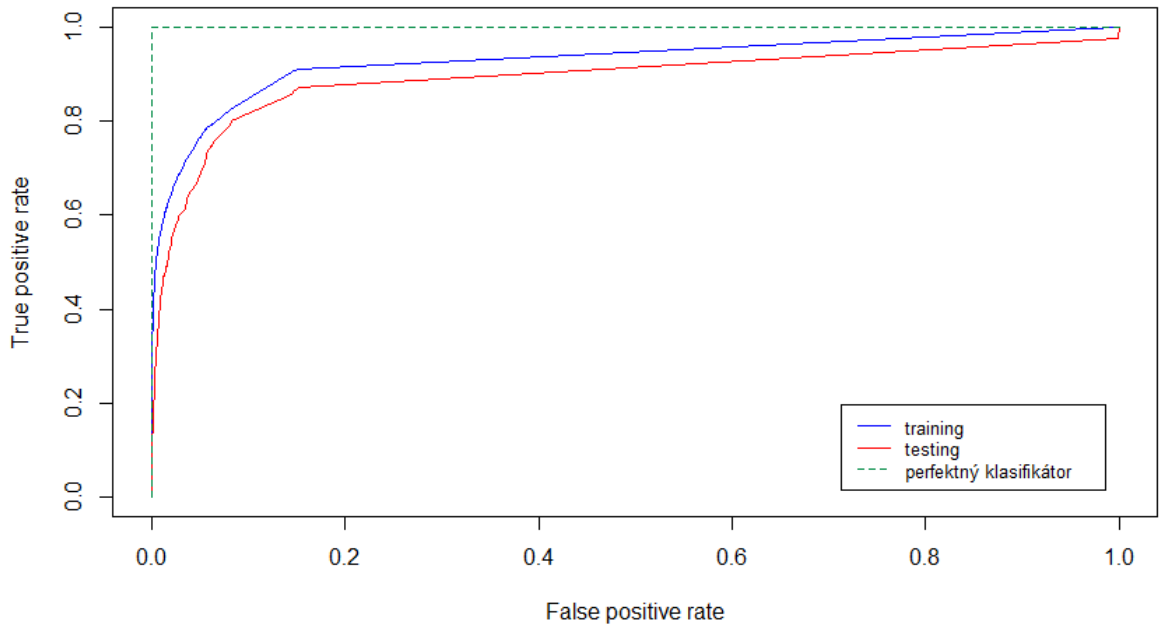
	Realita: False	Realita: True
Predikcia: False	396 713	1 957
Predikcia: True	1 945	2 012

Optimálny prah $\in [0{,}158 ; 0{,}158]$; $\varepsilon \doteq 0{,}3\%$; $S\varepsilon = 50{,}69\%$

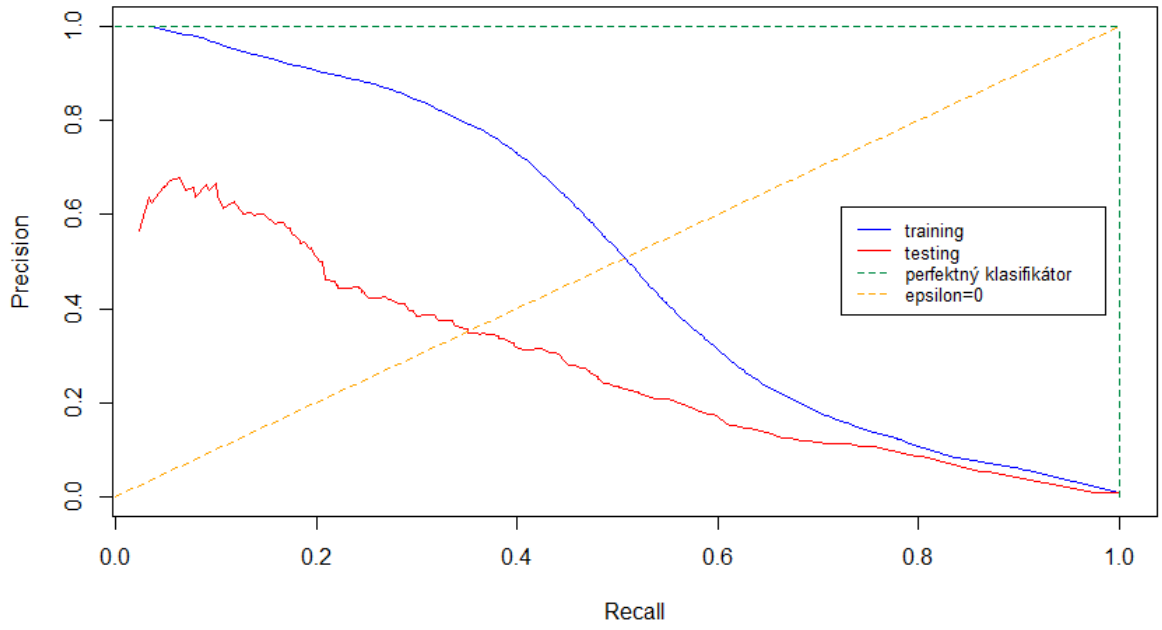
Tabuľka 14: Výsledky zložitého klasifikačného stromu na testovacích dátach *testing* s optimálnym prahom získaným z tréningových dát

	Realita: False	Realita: True
Predikcia: False	132 009	858
Predikcia: True	876	465

$\varepsilon = \frac{18}{1323} \doteq 1{,}4\%$; $S\varepsilon$=35,14%



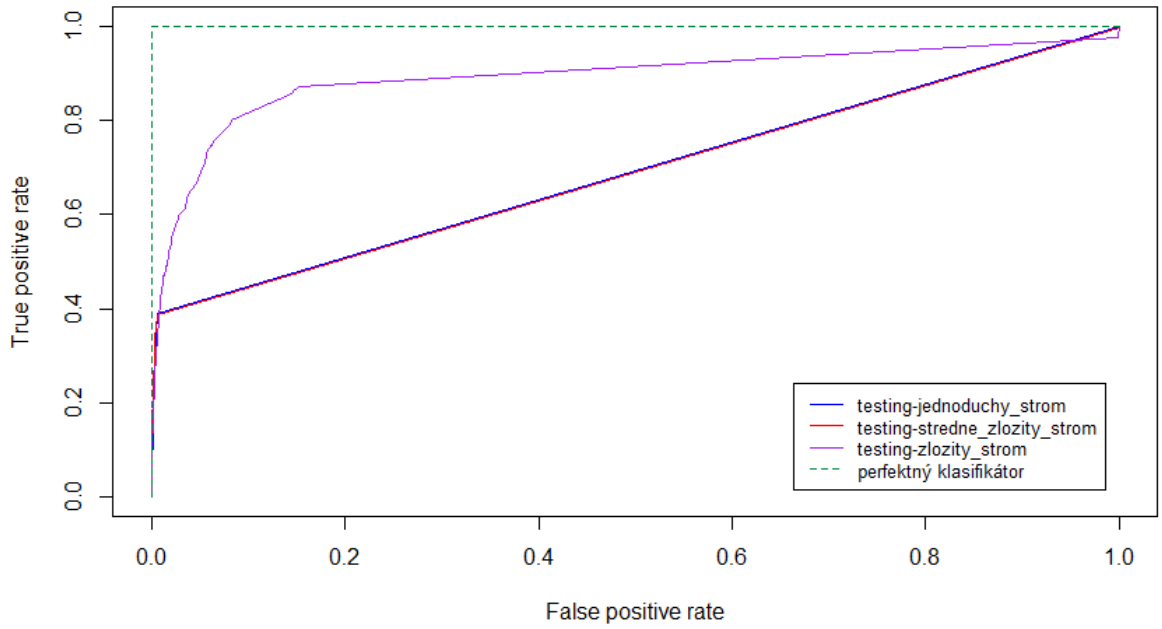
Obr. 21: ROC krivky pre zložitý klasifikačný strom



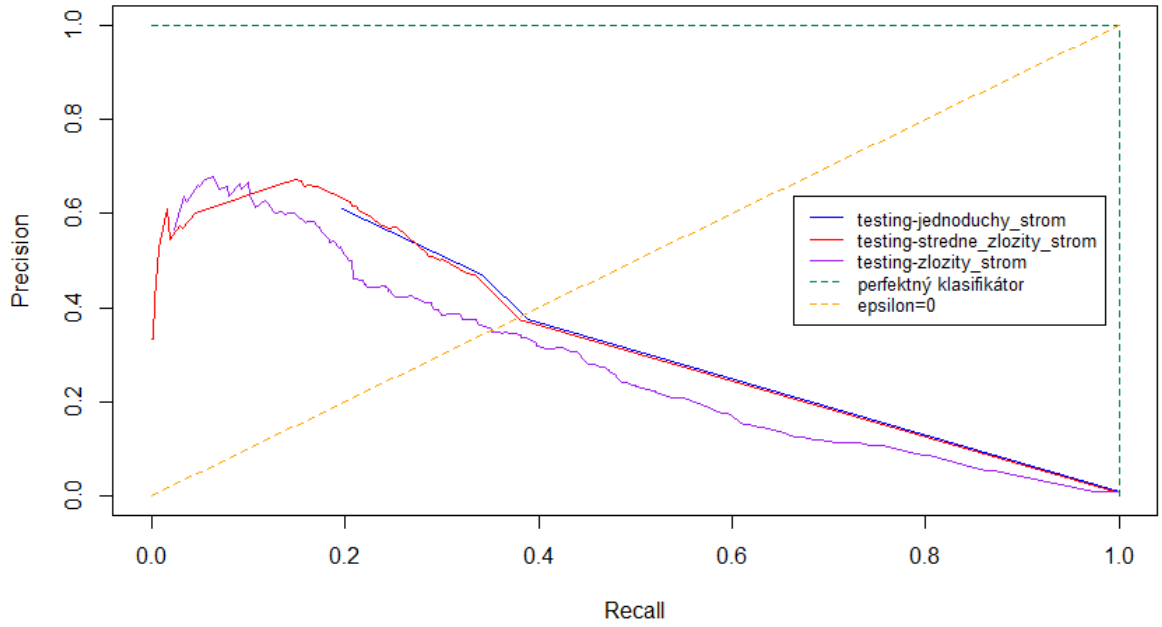
Obr. 22: P-R krivky pre zložitý klasifikačný strom

Na obrázku 22 si môžeme všimnúť, že červená P-R krivka pre testovacie dáta je značne nižšie ako P-R krivka pre tréningové dáta pre zložitý klasifikačný strom. Podobnú

situáciu sme videli pri metóde oporného bodu a značí to pretrénovanie modelu. Aj preto zrejme klesla Sε pre testovacie dáta pre zložitý strom oproti predošlým dvom stromom.



Obr. 23: Porovnanie ROC kriviek pre 3 klasifikačné stromy na dátach *testing*



Obr. 24: Porovnanie P-R kriviek pre 3 klasifikačné stromy na dátach *testing*

Ukázalo sa, že hĺbka klasifikačného stromu príliš neovplyvňuje úspešnosť modelu podľa nami zvolenej metriky. Z troch stromov bol aj podľa grafických ukazovateľov mierne menej úspešný zložitý klasifikačný strom, zrejme kvôli pretrénovaniu sa, ale výsledok aj napriek tomu nebol príliš odlišný od ostatných dvoch.

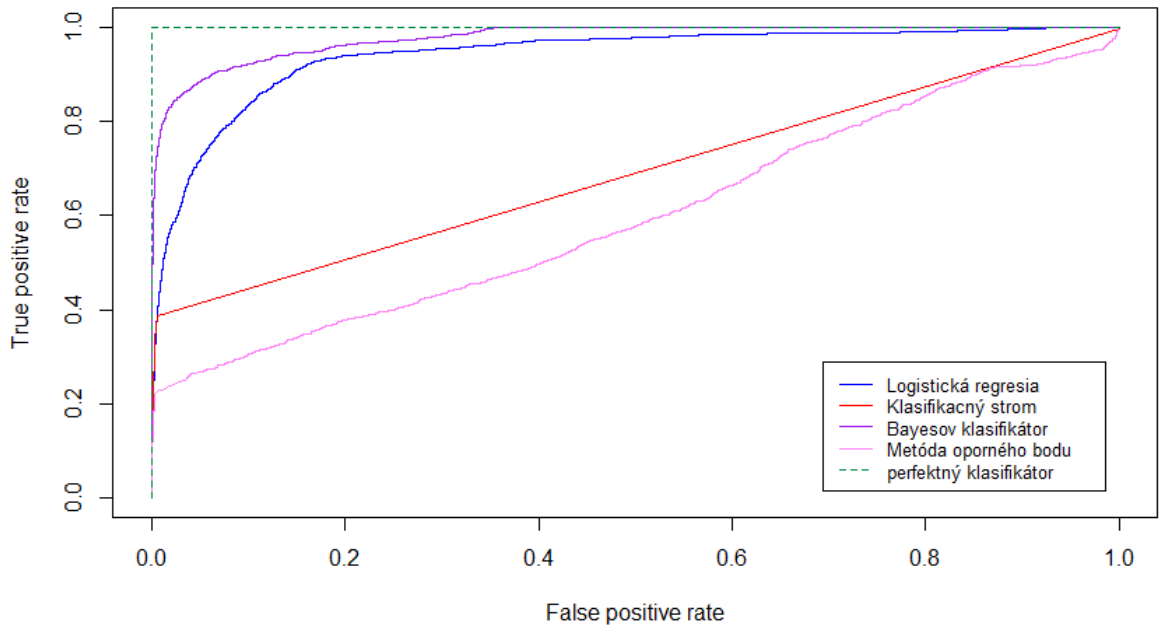
2.4.5 Porovnanie metód

Tabuľka 15: Zhrnutie doterajších výsledkov

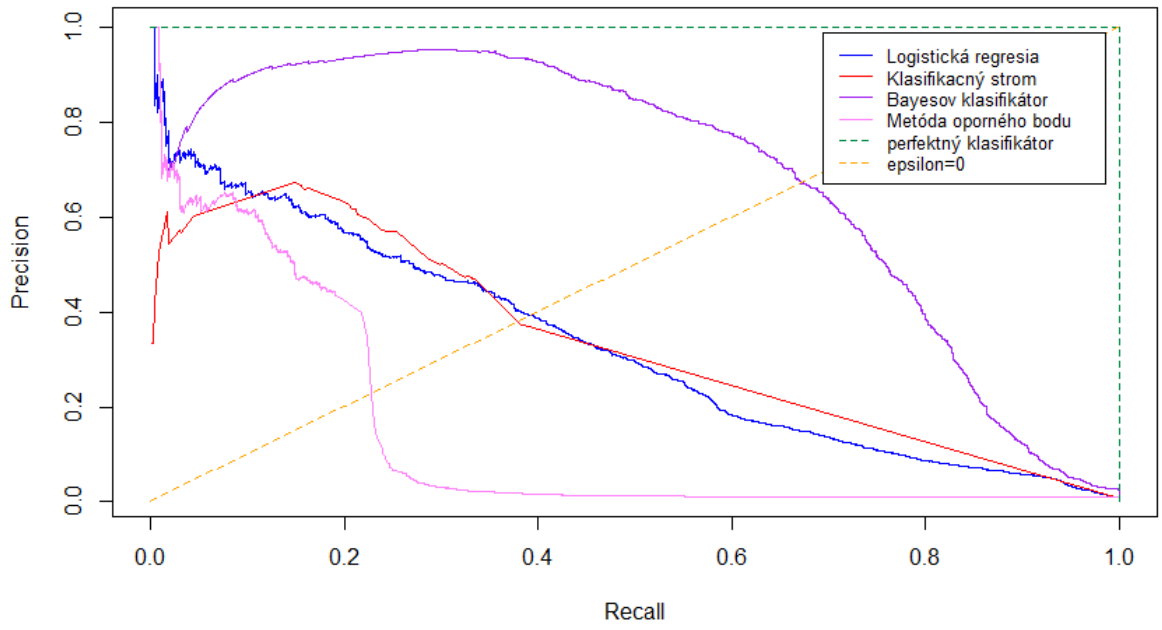
Názov	Sε-test	ε-test	Sε-train	ε-train	AUC-test	AUCPR-test
Bayesov klasifikátor	68,10%	2,7%	68,33%	0%	0,9724	0,6627
Logistická regresia	40,06%	3,8%	39,20%	0%	0,9433	0,3947
Klasifikačný strom	38,93%	4,2%	39,58%	2,5%	0,6919	0,3713
Metóda oporného bodu	90,63%	8650%	68,32%	0%	0,5991	0,1446

V tabuľke 15 uvádzame najlepšie klasifikátory zo všetkých 4 rôznych metód ktorým sme sa v tejto práci venovali. Sú zoradené od najúspešnejšej metódy po najmenej úspešnú. Ako kritérium úspešnej výkonnosti sme považovali senzitivitu pre testovacie dáta – Sε-test. Zároveň sme však dbali na to, aby ε pre testovacie dáta nebolo príliš ďaleko od nuly. Toto bolo pre nás dôležité, pretože za úspešný klasifikátor sme považovali taký, ktorý vedel aspoň približne odhadnúť počet platičov. Ako kritérium úspešného odhadnutia počtu platiacich hráčov sme si stanovili hranicu ε=10%, ktorú prekročila jedine metóda oporného bodu. Preto aj napriek vysokej senzitivite pre testovacie dáta sa nachádza na poslednom mieste v rebríčku. V tabuľke pozorujeme, že senzitivita pre tréningové dáta je pre úspešné klasifikátory približne rovnaká ako senzitivita pre testovacie dáta, v niektorých prípadoch nižšia, v niektorých vyššia. To je dobrý znak, pretože to značí vyvážený model. Ak by senzitivita pre tréningové dáta bola príliš vysoká oproti senzitivite pre testovacie dáta, značilo by to pretrénovanie modelu. Ak by bola príliš nízka, značilo by to nedostatočný model a vysokú senzitivitu pre testovacie dáta by sme mohli považovať za dielo náhody. Ďalej pozorujeme, že aj keď sa ε pre tréningové dáta v prípade klasifikačného stromu nepodarilo stlačiť na nulu v konečnom dôsledku to neviedlo k negatívnej výkonnosti modelu. Ďalšie pozorovanie z tabuľky 15 je možno najzaujímavejšie: S podľa nami vybranej metriky hodnotenia výkonnosti modelu sa zdanlivo zhodujú aj metriky AUC pre ROC krivku a AUC pre P-R krivku. Pre najúspešnejší Bayesov model spomedzi všetkých

klasifikátorov je AUC aj AUCPR pre testovacie dáta najvyššia hodnota a pre najmenej úspešný klasifikátor metódy oporného body sú tieto dve hodnoty najnižšie. Pre približne rovnako úspešné metódy logistickú regresiu a klasifikačný strom sa poradie pre AUC zachovalo, ale hodnota AUC pre logistickú regresiu je omnoho bližšie k hodnote AUC pre Bayesov klasifikátor, než k hodnote AUC pre klasifikačný strom. Zdá sa však, že úspešnosť modelu dobre opisuje premenná AUCPR, pretože jej hodnoty majú takmer 1-kovú koreláciu so senzitivitou pre testovacie dáta v prípade prvých troch metód. Pre štvrtú, metódu oporného bodu sa hodnota AUCPR s hodnotou senzitivity pre testovacie dáta líši, ale kvôli veľkému ε sme tento klasifikátor považovali za nekvalitný. Túto mieru nekvality AUCPR veľmi dobre zachytilo. Rozdiely v hodnotách AUC a AUCPR sú dobre viditeľné aj na obrázkoch 25 a 26 uvedených nižšie.



Obr. 25: Porovnanie ROC kriviek pre 4 klasifikátory na dátach *testing*



Obr. 26: Porovnanie P-R kriviek pre 4 klasifikátory na dátach *testing*

Pre úspešné klasifikátory sme sa rozhodli zmerať dôležitosť prediktorov. Ako sme už spomínali v kapitole 1.8 spôsobov ako túto dôležitosť odmerať je pre každú metódu niekoľko. Tieto spôsoby sú pre rôzne klasifikátory rôzne a môžu ovplyvniť poradie dôležitosti. Našim cieľom však nie je jednoznačne určiť, ktorý prediktor je najdôležitejší, ale získať rozhľad o niektorých najdôležitejších. Preto sme sa rozhodli pre každý klasifikátor vybrať 6 najlepších. V prípade logistickej regresie a klasifikačného stromu sme použili v R-ku implementovanú funkciu varImp(), ktorá meria a porovnáva výkon modelu s použitím a bez použitia prediktora. Presný spôsob výpočtu je popísaný v [13]. Pre Bayesov klasifikátor nemá R-ko implementovanú funkciu na výpočet dôležitosti prediktorov, preto sme sa rozhodli použiť vlastný spôsob, pričom sme sa inšpirovali už spomínaným zdrojom. Rozhodli sme sa zmerať výkonnosť klasifikátora s a bez jednotlivých prediktorov a vzniknutý rozdiel vo výkonnosti sme považovali za dôležitosť prediktora. Ako výkonnosť modelu sme určili epsilon-optimálnu senzitivitu $S\varepsilon$ pre testovacie dáta. Výsledky môžeme vidieť v tabuľke 16.

Tabuľka 16: Dôležitosť prediktorov

Poradie	Bayesov klasifikátor	Logistická regresia	Klasifikačný strom
1.	*dx_revenue*	*dx_pay_count*	*dx_revenue*
2.	*dx_pay_count*	*d1x_session_time*	*dx_pay_count*
3.	*dx_gems_spent*	*dx_gems_count*	*dx_gems_spent*
4.	*tier*	*tier*	*d1x_session_time*
5.	*dx_gems_count*	*dx_revenue*	*d1x_session_count*
6.	*dx_session_days*	*d1x_session_count*	*dx_session_time*

Pozorujeme, že premenné *dx_pay_count* a *dx_revenue* sa vyskytujú v prvej 6-ke najdôležitejších prediktorov pre všetky klasifikátory. *dx_revenue* je 2-krát najdôležitejší prediktor, *dx_pay_count* je 1-krát najdôležitejší prediktor a 2-krát druhý najdôležitejší. Práve dva krát sa v tabuľke 16 nachádzajú prediktory: *tier, d1x_session_time, dx_gems_spent, dx_gems_count, d1x_session_count* a práve jeden krát prediktory: *dx_session_days, dx_session_time.* Z tejto analýzy sa zdá, že najviac dominantnou črtou hráčov, ktorí majú v budúcnosti zaplatiť je to, koľko krát a koľko už zaplatili. Dôležitými sú ale aj informácie z akej krajiny hráči pochádzajú a koľko času hre venovali.

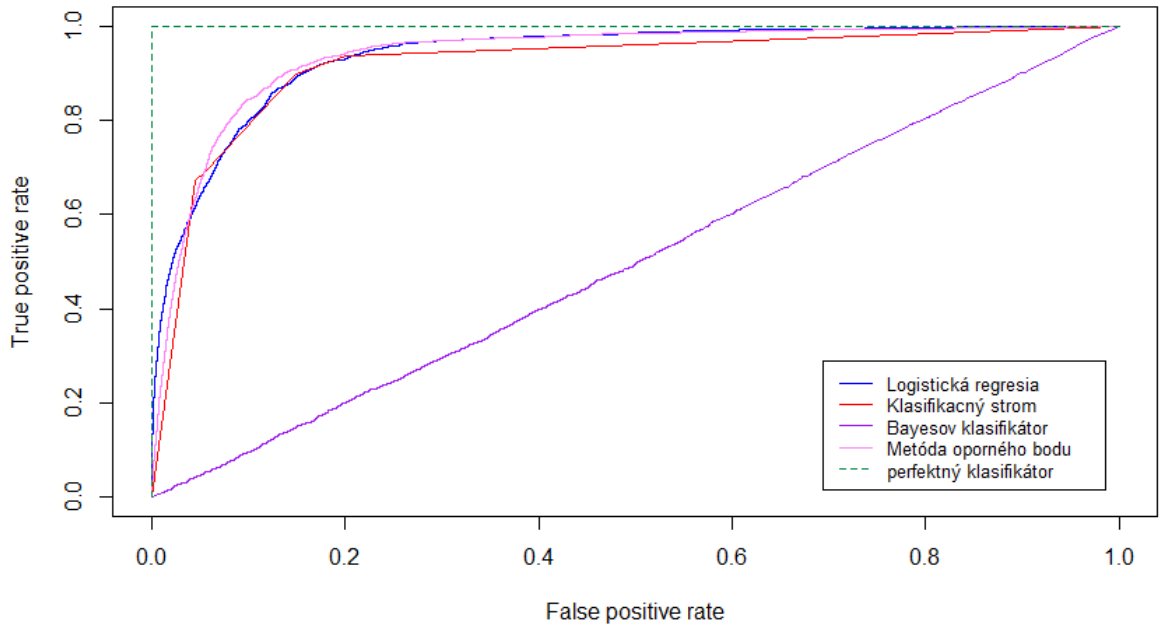
2.4.6 Vyvážené tréningové dáta

V tejto kapitole uvádzame výsledky klasifikátorov, ktoré sme vytvorili rovnakým spôsobom ako predošlé klasifikátory, ale ako tréningové dáta sme použili dáta *trainBalanced,* ktoré obsahujú informácie o 6 000 hráčoch, pričom práve polovica z nich má hodnoty *dy_pay_count* = 1 a druhá polovica hodnotu 0. Pre takéto zloženie tréningovej vzorky sme sa rozhodli preto, lebo sme sa domnievali, že ak budú obe triedy v dátach zastúpené rovnomerne, klasifikátor by mohol lepšie rozpoznať rozdiely medzi týmito triedami.

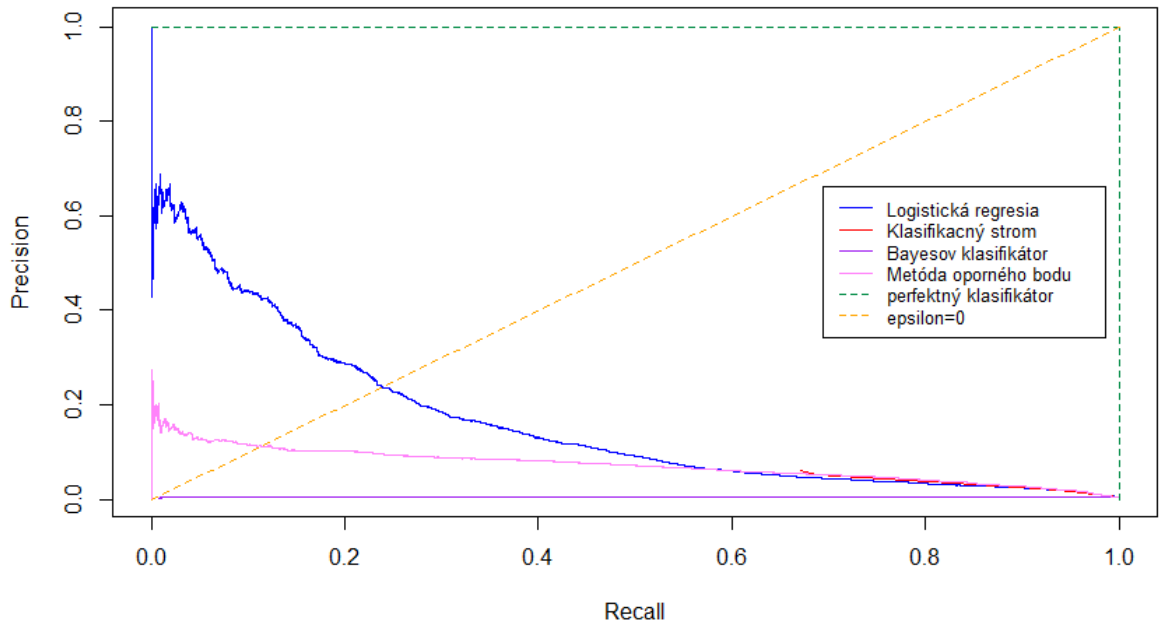
Tabuľka 17: Prehľad výsledkov klasifikátorov trénovaných na dátach *trainBalanced*, testovaných na *test*

Názov klasifikátora	Sε-test	ε-test	Sε-train	ε-train	AUC-test	AUCPR-test
Bayesov klasifikátor	14,40%	3249%	51,17%	0,03%	0,4992	0,0029
Logistická regresia	85,99%	3074%	86,77%	0%	0,9371	0,1755
Klasifikačný strom	89,79%	3435%	90,83%	5,07%	0,9181	0,3731
Metóda oporného bodu	87,30%	2817%	88,33%	0%	0,9378	0,1010

Ukázalo sa, že pre náš problém nebolo vhodné použiť vyvážené tréningové data. Hodnoty Sε sú pre 3 klasifikátory síce väčšie, než pre klasifikátory natrénované na nevyvážených dátach *training*, ale ε je pri testovaní vo všetkých prípadoch niekoľko tisícok percent. Všetky 4 klasifikátory mali väčšiu tendenciu hráčov klasifikovať ako platičov, než ako neplatičov, čo je pravdepodobne dôsledkom neprirodzene vysokej koncentrácie platičov v tréningovej vzorke. Hodnoty Sε sú vysoké čísla presne z toho dôvodu. Pri pohľade na obrázky 27 a 28 a hodnotu AUC z tabuľky 17 pozorujeme, že Bayesov klasifikátor je v tomto prípade približne taký dobrý ako náhodný klasifikátor.



Obr. 27: Porovnanie ROC kriviek pre 4 klasifikátory na dátach *testBalanced*, trénované na dátach *trainBalanced*



Obr. 28: Porovnanie P-R kriviek pre 4 klasifikátory na dátach *testBalanced*, trénované na dátach *trainBalanced*

Z obrázkov ROC kriviek pre testovacie dáta jednotlivých klasifikátorov nevidno, že by boli zlé vzhľadom na ich blízkosť k perfektnému klasifikátoru. Naopak z obrázku 28, kde sú znázornené krivky P-R je vidno, že klasifikátory nemajú dobrú výkonnosť, čo usudzujeme z ich blízkosti k ľavému dolnému rohu na grafe.

Záver

V tejto diplomovej práci sme sa venovali predikcii správania sa hráčov mobilných hier, konkrétne predikcii, či noví hráči v hre v blízkej budúcnosti investujú peniaze do hry. Našim cieľom bolo čo najlepšie odhadnúť počet hráčov, ktorí zaplatia a zároveň identifikovať ktorí to budú. Na splnenie tohto cieľa sme konštruovali klasifikátory s dôrazom na čo najväčšiu mieru úspešnosti klasifikácie, k čomu sme potrebovali nájsť vhodný aparát. Keďže nás zaujímal aj počet a aj ktorí hráči zaplatia, potrebovali sme zakomponovať obe kritériá do jedného spoločného kritéria. Najprv sme definovali ε ako relatívnu chybu klasifikátora, ktorá merala odchýlku od presnej predikcie počtu budúcich platiacich hráčov. Definovali sme optimálny prah tak, aby pre tréningové dáta minimalizoval ε. Nakoniec sme definovali Sε – senzitivitu pre ε -optimálny prah. Ako kritérium úspešnosti sme určili Sε pre testovacie dáta za predpokladu, že ε pre tieto dáta nepresiahne 10%. Určením tohto kritéria úspešnosti klasifikátorov sme splnili prvú časť cieľa, pretože táto miera berie do úvahy aj predikovaný počet aj predikciu pre konkrétnych hráčov.

Tréningové a testovacie dáta sme vytvorili ako dve disjunktné podmnožiny pôvodných dát, pričom do tréningovej časti sme vložili náhodne vybraných 75% a do testovacej časti zvyšných 25%. Podľa nami definovanej miery pre naše dáta boli pre klasifikačnú úlohu 3 zo 4 metód relatívne presné. Klasifikátor skonštruovaný podľa metódy oporného bodu nebol dostatočne úspešný. Už prvý problém pri metóde oporného bodu bol, že sme museli značne znížiť veľkosť tréningovej vzorky oproti iným metódam, kvôli jej výpočtovej náročnosti. Síce klasifikátor dosiahol veľkú senzitivitu pre testovacie dáta približne (Sε =90,63%) v predikovanom počte hráčov sa veľmi zmýlil (ε =8650%). Pomocou ostatných 3 metód sa nám podarilo skonštruovať úspešné klasifikátory, keďže pre všetky bolo ε pre testovacie dáta menšie, než 5%. Približne rovnako dobre klasifikovali klasifikačné stromy (Sε =38,93%) a logistická regresia (Sε =40,06%). Klasifikačných stromov sme skonštruovali viacero pomocou prehľadávania parametrického priestoru hyperparametra, ktorý nepriamo riadil hĺbku stromu. Ukázalo sa, že čím jednoduchší strom, tým lepšia senzitivita pre testovacie dáta. V prípade príliš komplikovaného stromu sme boli svedkami pretrénovania sa algoritmu, keďže senzitivitu pre tréningové dáta (50,69%) mal značne vyššiu, než senzitivitu pre testovacie dáta (35,14%). V prípade najjednoduchšieho klasifikačného stromu bol rozdiel pre tréningové dáta (39,58%) a testovacie dáta (38,93%) zanedbateľný. Jednoznačne najúspešnejšou metódou bol Bayesov klasifikátor, ktorý

dosiahol senzitivitu pre testovacie dáta (Sε =68,10%), čiže bol schopný odhadnúť presný počet budúcich platiacich hráčov s chybou presnosti do 5% a zároveň úspešne identifikoval približne 7 z 10-tich skutočných budúcich platičov.

Okrem porovnávania podľa Sε sme sledovali výkonnosť klasifikátorov aj graficky podľa ROC a P-R kriviek. Z výsledkov práce sa zdá, že P-R krivky sú vhodnejším grafickým porovnaním jednotlivých metód pre naše dáta. Najmä veličina AUCPR, teda plocha pod krivkou P-R sa javí ako dobrá alternatíva k nami definovanej metodológii určovania úspešnosti klasifikátora. Námetom pre budúce práce by mohlo byť skúmať, ktorá veličina, či AUCPR alebo Sε lepšie vystihuje spomínanú úspešnosť.

V práci sme sa ďalej pozreli na to, ktoré prediktory sú tie ktoré najviac ovplyvňujú, či hráč zaplatí. Po analýze dôležitosti prediktorov sme zistili, že najdôležitejšie sú pre jednotlivé klasifikátory rôzne. Napriek tomu sme podľa frekvencie výskytu a pozícii prediktorov pre 3 úspešné metódy z tabuľky 16 skonštatovali, že najdôležitejším faktorom je či hráč už v minulosti zaplatil a ak áno tak koľko. Ďalšími dôležitými prediktormi sa zdajú byť z akej skupiny krajín hráč pochádza, koľko-krát sa hráč prihlásil, koľko vylepšení v hre hráč využil a koľko času v hre strávil.

Ďalej sme sa v práci snažili výkonnosť klasifikátorov ešte zvýšiť. Domnievali sme sa, že sa nám to možno podarí zmenou tréningových dát, tak aby v nich boli rovnomerne distribuované výstupné triedy. Po vytvorení klasifikátorov na takejto vyváženej tréningovej vzorke, ktorá obsahovala približne 1% všetkých dát (testovacia vzorka obsahovala zvyšných 99%) sa ukázalo, že ani jeden takto skonštruovaný klasifikátor nemá dostatočnú výkonnosť. Všetky klasifikátory mali príliš vysokú tendenciu hráčov klasifikovať ako platičov, čoho dôsledkom bolo vysoké ε , približne 3000%. Existujú aj ďalšie spôsoby, ktoré majú potenciál nami sledovanú výkonnosť zvýšiť, ktorým by sme sa v prípade rozsiahlejšieho výskumu venovali. Napríklad odlišné vstupné dáta, transformácia vstupných dát alebo modifikované spôsoby trénovania klasifikátorov. My sme pracovali s dátami o hráčoch zo šiesteho dňa, ktorý zo všetkých dní obsahoval o hráčoch najviac informácií, ale možno by klasifikátor fungoval lepšie, keby mal k dispozícii aj dáta z ostatných dní. Ďalej dáta, ktoré sme mali k dispozícií sme použili priamo na tvorbu modelov. Spôsoby transformácie dát ako analýza hlavných komponentov, faktorová analýza, analýza zhlukov alebo iné by možno tiež prispeli k ich zlepšeniu. Ďalej by mohlo byť vhodné používať pri tvorbe klasifikátorov metódu cross-validácie, čiže rozdelenie na

tréningovú a testovaciu časť dát by sme urobili viackrát a možno aj v rôznych pomeroch, čoho výsledkom by bola väčšia istota v naše výsledky a lepšie rozhodovanie pri porovnávaní modelov. Okrem spomínaných vylepšení by sme ešte mohli modelovať pomocou iných metód, napríklad pomocou náhodných lesov alebo umelých neurónových sietí. Uvedomujeme si, že potenciál na zvýšenie úspešnosti predikcie je, napriek tomu si myslíme, že cieľ, ktorý sme si na začiatku práce stanovili sa nám podarilo splniť.

Zoznam použitej literatúry

[1] BERGSTRA, J., BENGIO, Y.: *Random search for Hyper-Parameter Optimization*, (2012). Dostupné na internete (8.5.2020): https://www.jmlr.org/papers/volume13/ bergstra12a /bergstra12a.pdf

[2] BLOCKEEL, H., KERSTING, K., et al.: *Machine Learning and Knowledge Discovery in Databases*, Springer (2013).

[3] BREIMAN, L.: *Classification and Regression Trees*-CRC Press, (2017).

[4] CHEN, C.H.: *ROC curves for continuous data*-CRC Press, (2009).

[5] DAUGMAN, J., ADLER, A., et al.: *Support Vector Machine.* (2009).

[6] EKELUND, S.: *Precision-recall curves – what are they and how are they used?* (2017), Dostupné na internete (7.3.2020): https://acutecaretesting.org/en/articles/precision-recall-curves-what-are-they-and-how-are-they-used

[7] HARMAN, R.: Multivariate Statistical Analysis (2018),

Dostupné na internete (13.1.2020) http://www.iam.fmph.uniba.sk/ospm/Harman/VSAp.pdf

[8] HARRELL, F.E. JR.: *Regression Modelling Strategies*, Springer (2015).

[9] IZENMAN, A. J.: *Modern multivariate statistical techniques*, Springer (2008).

[10] KUHN, M., JOHNSON, K.: *Applied Predictive Modeling*, Springer (2013).

[11] LE, J.: *Support vector machines in R* (2018), Dostupné na internete (17.2.2020) https://www.datacamp.com/community/tutorials/support-vector-machines-r

[12] PENG, C.-Y. J., LEE, K. L., INGERSOLL, G. M.: *An Introduction to Logistic Regression Analysis and Reporting. The Journal of Educational Research* (2002).

[13] RDOCUMENTATION, *Calculation Of Variable Importance For Regression And Classification Models,* Dostupné na internete (7.5.2020) https://www.rdocumentation.org/ packages/caret/versions/6.0-86/topics/varImp

[14] STOLTZFUS, J. C.: *Logistic Regression: A Brief Primer. Academic Emergency Medicine* (2011).