Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Adaptívne modely pre klasifikáciu nevybalansovaných dátových prúdov

Diplomová práca

2026 **Matúš Konopinský**

Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Adaptívne modely pre klasifikáciu nevybalansovaných dátových prúdov

Diplomová práca

Študijný program:	Inteligentné systémy
Študijný odbor:	Informatika
Školiace pracovisko:	Ústav umelej inteligencie (ÚUI)
Školiteľ:	doc. Ing. Martin Sarnovský, PhD.

Košice 2026 **Matúš Konopinský**

Abstrakt v SJ

Táto práca sa zaoberá klasifikáciou nevybalansovaných dátových prúdov s konceptovým driftom. Rozširujeme heterogénny ensemble model DDCW (Diversified Dynamic Class-Weighted) o mechanizmy na zvládanie nevyváženosti tried: adaptívne per-class váhovanie, replay menšinových vzoriek s augmentáciou regulovanou spätnou väzbou na senzitivita väčšinovej triedy, soft voting a Page-Hinkley detektor driftu. Model bol implementovaný v Pythone s využitím frameworku scikit-multiflow a vyhodnotený na 15 datasetoch vrátane syntetických prúdov s rôznymi typmi driftu, reálnych tabulárnych dát a textových dát predspracovaných pomocou GloVe embeddingov. Experimenty ukazujú, že navrhovaný prístup prináša zlepšenie na silno nevyvážených datasetoch, zatiaľ čo na mierne nevyvážených dátach zostávajú štandardné ensemble metódy konkurencieschopné.

Kľúčové slová v SJ

nevybalansované dátové prúdy, konceptový drift, ensemble učenie, online učenie, adaptívna klasifikácia

Abstrakt v AJ

This thesis addresses the classification of imbalanced data streams in the presence of concept drift. We extend the DDCW (Diversified Dynamic Class-Weighted) heterogeneous ensemble model with mechanisms targeting class imbalance: adaptive per-class weighting, minority class replay with augmentation regulated by majority recall feedback, soft voting, and a Page-Hinkley drift detector. The model was implemented in Python using the scikit-multiflow framework and evaluated on 15 datasets including synthetic streams with various drift types, real-world tabular data, and text data preprocessed with GloVe embeddings. Experiments show that the proposed approach brings notable improvements on highly imbalanced datasets, while on mildly imbalanced data standard ensemble methods remain competitive.

Kľúčové slová v AJ

imbalanced data streams, concept drift, ensemble learning, online learning, adaptive classification

Bibliografická citácia

KONOPINSKÝ, Matúš. *Adaptívne modely pre klasifikáciu nevybalansovaných dátových prúdov*. Košice: Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky, 2026. 64s. Vedúci práce: doc. Ing. Martin Sarnovský, PhD.

78747

TECHNICKÁ UNIVERZITA V KOŠICIACH

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Katedra kybernetiky a umelej inteligencie

ZADANIE
DIPLOMOVEJ PRÁCE

Študijný odbor: **Informatika**

Študijný program: **Inteligentné systémy**

Názov práce:

Adaptívne modely pre klasifikáciu nevybalansovaných dátových prúdov.

Adaptive models for classification on imbalanced data streams.

Študent: **Bc. Matúš Konopinský**

Školiteľ: **doc. Ing. Martin Sarnovský, PhD.**

Školiace pracovisko: **Katedra kybernetiky a umelej inteligencie**

Konzultant práce:

Pracovisko konzultanta:

Pokyny na vypracovanie diplomovej práce:

1. Podať teoretický prehľad oblasti dátových prúdov a konceptového driftu.
2. Podať prehľad súčasného stavu v oblasti zložených modelov používaných v klasifikácii na driftujúcich dátach.
3. Navrhnúť a implementovať zložený klasifikátor pozostávajúci z množiny rôznych základných modelov.
4. Experimentálne vyhodnotiť navrhnutý model na zvolených prúdoch dát.
5. Vypracovať dokumentáciu podľa pokynov vedúceho práce.

Jazyk, v ktorom sa práca vypracuje: slovenský

Termín pre odovzdanie práce: 24.04.2026

Dátum zadania diplomovej práce: 31.10.2025

TECHNICKÁ UNIVERZITA V KOŠICIACH FAKULTA ELEKTROTECHNIKY A INFORMATIKY -12-

v.z.

prof. Ing. Liberios Vokorokos, PhD.

dekan fakulty

Čestné vyhlásenie

Vyhlasujem, že som záverečnú prácu vypracoval samostatne s použitím uvedenej odbornej literatúry, ako aj s využitím nástrojov generatívnej umelej inteligencie v súlade s Metodickým pokynom o záverečných a kvalifikačných prácach a Etickým kódexom študenta Technickej univerzity v Košiciach. Umelá inteligencia bola použitá ako podporný nástroj bez porušenia zásad akademickej etiky a autorskej integrity.

Košice, 24.4.2026 .....

Vlastnoručný podpis

Poďakovanie

Na tomto mieste by som rád poďakoval svojmu vedúcemu práce, doc. Ing. Martinovi Sarnovskému, PhD., za jeho cenné rady, odborné vedenie a trpezlivosť, ktoré mi pomohli usmerniť a zlepšiť moju prácu.

Taktiež ďakujem všetkým, ktorí prispeli k tomu, aby táto diplomová práca mohla vzniknúť.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Objem dát, ktorý dnes generujú rôzne zariadenia, senzory či webové aplikácie, rastie nesmiernou rýchlosťou. Veľká časť týchto dát má charakter dátových prúdov — prichádzajú kontinuálne, nemožno ich celé uložiť a treba ich spracovať za behu [1]. Tradičné modely strojového učenia, natrénované raz na fixnom datasete, v takomto dynamickom prostredí postupne strácajú presnosť. Rozdelenie dát sa totiž mení — objavujú sa nové vzorce, staré zanikajú. Tomuto javu sa hovorí konceptový drift [2].

Jedným z osvedčených spôsobov, ako sa s driftom vysporiadať, sú adaptívne ensemble modely. Tie namiesto jedného klasifikátora udržiavajú celú skupinu modelov a priebežne rozhodujú, ktoré z nich sú ešte užitočné a ktoré treba nahradiť [1]. No drift nie je jedinou komplikáciou. Reálne dátové prúdy sú často aj výrazne nevyvážené — jedna trieda (napr. normálna sieťová prevádzka) dominuje a tá zaujímavá (útok, toxický komentár, meškanie letu) tvorí len malú časť dát. Keď sa oba problémy vyskytnú naraz, situácia sa komplikuje ešte viac. Model musí zároveň reagovať na meniace sa rozdelenie dát aj zachovať schopnosť identifikovať zriedkavé, ale dôležité prípady menšinových tried [3]. Väčšina súčasných metód rieši drift a nevyváženosť oddelene, čo v praxi nemusí stačiť [4].

Prečo je to dôležité vidíme, keď sa pozrieme na niektoré konkrétne aplikácie. Pri detekcii sieťových prienikov tvoria útoky zlomok bežnej prevádzky a ich charakter sa mení, ako vidíme napríklad na datasete KDD99. Pri moderácii online obsahu — napríklad detekcii toxických komentárov — je problémový obsah menšinou a jazyk na internete sa neustále vyvíja [5]. Aj pri predpovedaní letových meškaní sa distribúcia oneskorení mení podľa ročného obdobia. Vo všetkých týchto prípadoch je cena za prehliadnutie menšinovej triedy (nepodchytený útok, nezachytený toxický komentár) oveľa vyššia než cena za falošný poplach [3].

Formulácia úlohy

Cieľom diplomovej práce je navrhnúť a implementovať zložený ensemble klasifikátor pre klasifikáciu nevybalansovaných dátových prúdov s konceptovým driftom. Zadanie práce sme rozpracovali do nasledujúcich bodov:

1. Spracovať teoretický prehľad problematiky dátových prúdov, konceptového driftu a nevyvážených tried.
2. Zmapovať aktuálny stav výskumu zložených ensemble modelov pre klasifikáciu na driftujúcich a nevyvážených dátach.
3. Navrhnúť a naimplementovať nový ensemble model IDDCW (Imbalance-aware Dynamic Diversified Class-Weighted ensemble), ktorý nadväzuje na pôvodný DDCW model [6] a rozširuje ho o mechanizmy na zvládanie driftu aj nevyváženosti tried súčasne.
4. Experimentálne overiť navrhnutý model na syntetických aj reálnych datasetoch s rôznymi typmi driftu a stupňami nevyváženosti.
5. Vypracovať dokumentáciu podľa pokynov vedúceho práce.

Práca je rozdelená nasledovne. Prvá kapitola pokrýva teoretické základy — dátové prúdy, drift, nevyvážené triedy a metriky. Druhá kapitola mapuje existujúce ensemble prístupy. Tretia podrobne rozoberá algoritmy, na ktorých náš model stavia. Štvrtá kapitola predstavuje samotný návrh a implementáciu modelu IDDCW. Piata kapitola obsahuje experimenty a ich výsledky. Na záver zhŕňame dosiahnuté výsledky a naznačujeme smerovanie ďalšieho výskumu.

1 Problematika

Na začiatok je dôležité ujasniť základné pojmy. Čo presne sú dátové prúdy, čo je konceptový drift a prečo sťažuje klasifikáciu. A ako do toho vstupuje nevyváženosť tried. Táto kapitola sa venuje práve týmto otázkam, spolu s prehľadom metrík a evaluačných postupov.

1.1 Dátové prúdy a ich charakteristiky

Dátový prúd je potenciálne neobmedzená sekvencia dátových inštancií prichádzajúcich v čase. Na rozdiel od tradičného dávkového spracovania, kde sú všetky dáta dostupné naraz a model sa trénuje na celej množine, dátové prúdy vyžadujú spracovanie v reálnom čase s obmedzenými výpočtovými a pamäťovými zdrojmi [1]. Medzi hlavné charakteristiky dátových prúdov patria [7]:

- Veľký alebo neobmedzený objem dát – dátový prúd nemá pevne definovaný koniec a nové inštancie prichádzajú priebežne.
- Sekvenčný prístup – každá inštancia je spracovaná najviac raz, pretože uchovávanie celej histórie nie je realizovateľné.
- Obmedzená pamäť a výpočtový čas – model musí spracovať každú inštanciu v čo najkratšom čase.
- Potenciálna zmena rozdelenia dát v čase – štatistické vlastnosti dát sa môžu meniť, čo si vyžaduje adaptívne algoritmy.

Tieto vlastnosti vyžadujú inkrementálne algoritmy, ktoré sa aktualizujú s každou novou inštanciou bez nutnosti uchovávať celú históriu dát. Formálne, dátový prúd možno definovať ako postupnosť dátových dvojíc (x_t, y_t), kde $x_t \in R^d$ je vektor atribútov a $y_t \in \{1, \ldots, C\}$ je trieda v časovom kroku t [8]. Je potrebné zdôrazniť aj to, že tradičné predpoklady strojového učenia, najmä predpoklad stacionárnosti rozdelenia dát, v kontexte dátových prúdov často neplatia [2].

V praxi sa dátové prúdy vyskytujú v mnohých aplikáciách. Sieťová prevádzka generuje milióny paketov za sekundu, senzory v IoT zariadeniach produkujú nepretržité časové rady, sociálne médiá generujú obrovské množstvo textového obsahu a finančné transakcie tvoria neustály prúd dát vyžadujúcich klasifikáciu v reálnom čase. Vo všetkých týchto prípadoch je tradičné dávkové spracovanie buď nepraktické, alebo úplne nemožné [1].

1.2 Konceptový drift v dátových tokoch

1.2.1 Formálna definícia

Ak rozdelenie dát v čase t označíme ako $P_t(x, y)$, kde x predstavuje vektor vstupných atribútov a y cieľovú premennú, o konceptovom drifte hovoríme vtedy, ak existuje časové obdobie $t+\Delta$, pre ktoré platí $P_t(x, y) \neq P_{t+\Delta}(x, y)$ [9]. Tento zdanlivo jednoduchý formalizmus v sebe ukrýva viacero dôležitých aspektov, pretože zmena spoločného rozdelenia $P(x, y)$ môže vyplývať zo zmeny rozdelenia vstupov $P(x)$, zmeny posterior pravdepodobnosti $P(y|x)$, alebo oboch súčasne [2].

Konceptový drift je jednou z najvýznamnejších výziev pri učení z dátových prúdov, pretože spôsobuje degradáciu výkonnosti modelov trénovaných na historických dátach. Model, ktorý bol v určitom okamihu presný, môže po zmene v dátach produkovať chybné predikcie, čo v kritických aplikáciách, ako je detekcia podvodov alebo diagnostika, môže mať závažné následky [10].

1.2.2 Typy konceptového driftu

Jednotlivé formy driftu majú rozdielny dopad na modelovanie a vyhodnocovanie výkonu. Podľa rýchlosti a charakteru zmeny rozlišujeme tieto typy [11]:

- Náhly drift (abrupt): Zmena rozdelenia nastane v krátkom čase, prakticky okamžite. Starý koncept sa náhle nahradí novým. Príkladom je zmena politiky spoločnosti, ktorá okamžite ovplyvní správanie zákazníkov. Tento typ driftu je relatívne ľahko detekovateľný, no vyžaduje rýchlu adaptáciu modelu [2].

- Postupný drift (gradual): Staré a nové koncepty sa dočasne vyskytujú súčasne, pričom podiel nového konceptu postupne rastie. V prechodnom období sa inštancie generujú striedavo z oboch konceptov. Tento typ je náročnejší na detekciu, pretože zmena je rozložená v čase a môže byť zamieňaná so šumom [11].

- Inkrementálny drift: Zmena prebieha po malých krokoch v dlhšom horizonte, kde sa rozhodovacia hranica posúva veľmi pomaly. Jednotlivé kroky zmeny sú často nerozpoznateľné od šumu, no kumulatívny efekt je výrazný. Tento typ je najnáročnejší na detekciu [2].

- Rekurentný drift (recurring): Staršie koncepty sa periodicky vracajú, napríklad sezónne javy v maloobchode alebo cyklické zmeny v sieťovej prevádzke. Schopnosť rozpoznať a znovupoužiť staré koncepty je cennou vlastnosťou adaptívnych modelov [11].

Z hľadiska toho, čo sa mení, rozlišujeme aj reálny drift a virtuálny drift. Pri reálnom drifte sa mení rozhodovacia hranica, teda podmienená pravdepodobnosť $P(y|x)$, čo priamo ovplyvňuje správnosť predikcií modelu. Pri virtuálnom drifte sa mení rozdelenie vstupných atribútov $P(x)$ bez zmeny rozhodovacej hranice, čo nemusí nutne viesť k poklesu výkonnosti, no môže ovplyvniť prácu detektorov driftu a efektivitu niektorých algoritmov [2]. V praxi sa často vyskytujú oba typy súčasne, čo ďalej komplikuje detekciu a adaptáciu.

1.2.3 Príčiny konceptového driftu

V dátových prúdoch môže konceptový drift vzniknúť v dôsledku viacerých faktorov. Príčiny môžu byť následovné: zmeny v prostredí generujúcom dáta (napr. zmena senzora, degradácia zariadenia), zmeny v správaní sledovaných subjektov (napr. zmena nákupných preferencií zákazníkov), vonkajšie faktory (napr. ekonomické zmeny, legislatívne úpravy) a adaptácia na rušivé, zámerne škodlivé vstupy (napr. tvorcovia spamu prispôsobujú techniky na obchádzanie filtrov) [2].

V kontexte textových dátových prúdov, napríklad zo sociálnych sietí, je drift obzvlášť častý kvôli rýchlemu vývoju jazyka, zmene tém diskusií, vzniku nového slangu a aktívnemu prispôsobovaniu sa používateľov detekčným mechanizmom [9]. Čo sa týka vzniku, práve tematické zmeny v dátových prúdoch sociálnych médií môžu byť významným spúšťačom konceptového driftu, čo negatívne ovplyvňuje výkonnosť neadaptívnych klasifikátorov [9].

1.2.4 Metódy detekcie konceptového driftu

Detekcia konceptového driftu je kľúčovým krokom v adaptívnom učení. V štúdii od Lu et al. rozdeľujú existujúce metódy detekcie do nasledovných kategórií [10]:

- Metódy založené na chybovosti modelu sledujú štatistické vlastnosti chybovosti klasifikátora v čase. DDM (Drift Detection Method) monitoruje mieru chybovosti a jej štandardnú odchýlku. Keď sa chybovosť významne zvýši nad očakávanú úroveň, DDM signalizuje warning alebo drift [12]. EDDM (Early Drift Detection Method) rozšírenie DDM vylepšuje detekciu postupného driftu tým, že monitoruje vzdialenosť medzi po sebe idúcimi chybami klasifikátora [13]. Výhodou týchto metód je ich jednoduchosť a nízke pamäťové nároky, no vyžadujú prístup k okamžitej spätnej väzbe (labely) [14].

- Window metódy porovnávajú vlastnosti dát v dvoch oknách – typicky aktuálnom a referenčnom. ADWIN (Adaptive Windowing) automaticky prispôsobuje veľkosť okna podľa miery zmeny v dátach [15]. Udržiava okno premenlivej veľkosti, ktoré sa automaticky zväčšuje pri stacionárnych dátach (pre vyššiu presnosť odhadov) a zmenšuje pri detekcii zmeny (pre rýchlejšiu adaptáciu). ADWIN poskytuje matematické záruky na mieru falošne pozitívnych a falošne negatívnych detekcií a dosahuje pamäťovú zložitosť $O(\log W)$ vďaka použitiu exponenciálnych histogramov [15].

- Metódy založené na monitorovaní distribúcie dát priamo porovnávajú rozdelenie dát v rôznych časových obdobiach pomocou štatistických testov, napríklad Kolmogorov-Smirnovov test alebo test rovnosti proporcií (STEPD). Tieto metódy môžu detekovať drift bez prístupu k labelom, čo je výhodné v aplikáciách, kde okamžitá spätná väzba nie je dostupná [10].

V rozsiahlom porovnávacom experimente bolo vyhodnotených viacero detektorov driftu a ukázalo sa, že žiadna jednotlivá metóda nedominuje naprieč všetkými scenármi – výber vhodného detektora závisí od typu driftu a charakteru dát [16].

1.2.5 Stratégie adaptácie na konceptový drift

Po detekcii driftu je potrebné model adaptovať na nové podmienky. Gama et al. v ich prieskume rozlišujú tri základné stratégie adaptácie [2]:

- Aktívne prístupy explicitne detekujú drift a na základe detekcie vykonávajú adaptáciu – napríklad reset modelu, rekalibráciu alebo výmenu základných klasifikátorov v ensemble. Výhodou je, že adaptácia sa vykonáva len keď je to potrebné, čo šetrí výpočtové zdroje. Nevýhodou je závislosť na kvalite detekcie – oneskorená alebo falošná detekcia vedie k suboptimálnej adaptácii [14].

- Pasívne prístupy nepotrebujú explicitnú detekciu driftu, ale neustále aktualizujú model na základe najnovších dát. Typickým príkladom je použitie posuvného okna fixnej veľkosti, kde sa model trénuje len na najnovších inštanciách. Výhodou je jednoduchosť a robustnosť voči rôznym typom driftu. Nevýhodou je, že veľkosť okna je kritickým hyperparametrom – príliš malé okno vedie k nestabilite, príliš veľké k pomalej adaptácii [2].

- Hybridné prístupy kombinujú aktívnu detekciu s pasívnou adaptáciou. Napríklad model sa priebežne aktualizuje na nových dátach, ale pri detekcii výrazného driftu sa vykoná intenzívnejšia adaptácia, ako výmena modelov alebo reset váh. Tento prístup je v praxi najčastejší a je základom mnohých moderných ensemble metód pre dátové prúdy [17].

1.3 Nevyvážené triedy v dátových prúdoch

1.3.1 Definícia a dôsledky

Problém nevyvážených tried nastáva, keď distribúcia tried nie je rovnomerná a jedna alebo viac tried je výrazne menej zastúpených než ostatné. V binárnej klasifikácii sa väčšinová trieda označuje ako majority class a menšinová ako minority class. Pomer nevyváženosti sa definuje ako pomer počtu inštancií väčšinovej triedy k počtu inštancií menšinovej triedy [18].

Nevyváženosť tried spôsobuje, že klasifikátory trénované na takýchto dátach majú tendenciu preferovať väčšinovú triedu, čo vedie k nízkemu pokrytiu menšinových tried. Štandardné metriky ako accuracy sú v takejto situácii zavádzajúce – model, ktorý vždy predikuje väčšinovú triedu, môže dosiahnuť vysokú celkovú presnosť, no úplne ignoruje menšinové triedy. V mnohých aplikáciách je pritom práve správna identifikácia menšinovej triedy kriticky dôležitá – napríklad pri detekcii podvodov, diagnostike zriedkavých ochorení, identifikácii toxického obsahu alebo detekcii sieťových útokov [18].

1.3.2 Statická a dynamická nevyváženosť

V tradičnom dávkovom spracovaní je pomer nevyváženosti fixný a známy vopred. V dátových prúdoch sa problém ďalej komplikuje tým, že nevyváženosť môže byť dynamická [3]:

- Pomer tried sa môže meniť v čase – trieda, ktorá bola väčšinová, sa môže stať menšinovou a naopak.

- Nové triedy môžu pribudnúť alebo existujúce triedy môžu zmiznúť.
- Nevyváženosť sa môže kombinovať s konceptovým driftom, kde sa súčasne mení distribúcia tried aj rozhodovacia hranica.
- V niektorých prípadoch sa vyskytujú ultra-zriedkavé triedy s jednotkami inštancií, kde tradičné techniky vyvažovania zlyhávajú.

Aguiar, Krawczyk a Cano zdôrazňujú, že kombinácia dynamickej nevyváženosti a konceptového driftu je jednou z najaktuálnejších a najnáročnejších výziev v oblasti učenia z dátových prúdov [3].

Tu sa vynára zásadná otázka: ak je menšinová trieda extrémne zriedkavá — napríklad toxické komentáre tvoria len niekoľko percent obsahu sociálnej siete — má vôbec zmysel trénovať model na ich rozpoznávanie v reálnom čase? Odpoveď závisí od kontextu. V niektorých prípadoch (detekcia sieťových útokov, odhalenie podvodných transakcií) je aj jediný neodhalený prípad potenciálne katastrofický a náklady falošne negatívnej klasifikácie sú neporovnateľne vyššie než náklady falošného poplachu. V takýchto scenároch je nevyhnutné udržiavať model, ktorý menšinovú triedu aktívne vyhľadáva, aj keď sa vyskytuje zriedka.

No je tu aj iný pohľad. V situáciách, kde je menšinová trieda momentálne takmer neprítomná, ale v budúcnosti sa jej výskyt môže zvýšiť (napríklad nový typ kybernetického útoku, nová forma online obťažovania, alebo náhly nárast dezinformácií počas krízy), je dôležité mať model, ktorý sa dokáže rýchlo adaptovať na meniaci sa pomer tried. Model, ktorý menšinovú triedu úplne ignoruje, sa v momente nárastu jej výskytu nedokáže dostatočne rýchlo prispôsobiť. Naopak, model s mechanizmami, ktoré umožnia, aby si udržiaval "povedomie" o menšinovej triede aj v obdobiach jej nízkeho výskytu, je pripravený reagovať, keď sa jej frekvencia zvýši. Ide teda nielen o klasifikáciu momentálne zriedkavých udalostí, ale aj o prípravu modelu na budúce zmeny v distribúcii tried.

1.3.3 Interakcia medzi konceptovým driftom a nevyváženosťou

Konceptový drift a nevyvážené triedy sa v praxi vyskytujú často súčasne a ich interakcia prináša špecifické komplikácie, ktoré nie sú prítomné pri riešení každého problému zvlášť. Drift môže meniť samotný pomer nevyváženosti – trieda, ktorá bola pôvodne menšinová, sa môže po zmene konceptu stať väčšinovou. Detektory driftu založené na chybovosti modelu môžu byť ovplyvnené nevyváženosťou – ak model systematicky zlyháva na menšinovej triede, chybovosť môže byť stabilná (pretože menšinová trieda tvorí malý podiel celkových chýb), čo maskuje skutočný drift [3].

Zároveň techniky vyvažovania tried môžu interferovať s detekciou driftu. Agresívny oversampling menšinovej triedy môže vyvolať falošnú detekciu driftu, pretože mení distribúciu dát prezentovanú modelu. Naopak, undersampling väčšinovej triedy môže znížiť citlivosť detekcie driftu v oblastiach priestoru, kde sú primárne väčšinové inštancie [4].

Tieto interakcie zdôrazňujú potrebu integrovaných prístupov, ktoré riešia drift a nevyváženosť v jednom koherentnom systéme, namiesto ich oddelených riešení nasadených nezávisle.

1.3.4 Prístupy na zvládanie nevyváženosti

Na riešenie problému nevyvážených tried existujú tri základné kategórie prístupov [18]:

- Prístupy na úrovni dát: Upravujú distribúciu tréningových dát pomocou prevzorkovania. Oversampling zvyšuje počet menšinových inštancií, napríklad SMOTE (Synthetic Minority Over-sampling Technique) generuje syntetické inštancie interpoláciou medzi existujúcimi menšinovými vzorkami a ich najbližšími susedmi [19]. Undersampling redukuje počet väčšinových inštancií. V kontexte dátových prúdov boli navrhnuté online adaptácie týchto techník, ako napríklad C-SMOTE pre kontinuálne prevzorkovanie [3].

- Prístupy na úrovni algoritmov: Modifikujú samotný algoritmus učenia tak, aby bol citlivý na nevyváženosť. Cost-sensitive učenie priraďuje vyššie náklady za chybné klasifikovanie menšinových inštancií, čím motivuje model k väčšej pozornosti voči zriedkavým triedam. Modifikované stratégie rozdeľovania v rozhodovacích stromoch alebo úprava stratovej funkcie v gradientových metódach patria do tejto kategórie [18].

- Ensemble prístupy: Kombinujú resampling alebo cost-sensitive učenie s ensemble metódami. Táto kategória sa ukázala ako najúčinnejšia v kontexte dátových prúdov, pretože ensemble metódy prirodzene umožňujú dynamickú správu modelov a kombináciu rôznych stratégií vyvažovania [3].

1.4 Metriky pre hodnotenie na nevyvážených dátach

Voľba vhodných evaluačných metrík je pri nevyvážených dátach kriticky dôležitá. Štandardná accuracy je nedostatočná, pretože model ignorujúci menšinovú triedu môže dosiahnuť vysokú celkovú presnosť [3].

Confusion matrix – tvorí základ pre výpočet väčšiny metrík. Pre binárnu klasifikáciu definuje štyri kategórie: True Positives (TP), True Negatives (TN), False Positives (FP) a False Negatives (FN) [20].

Precision – vyjadruje podiel správne predikovaných pozitívnych prípadov z celkového počtu predikovaných pozitívnych: Precision $= \frac{TP}{TP+FP}$ [20].

Recall – vyjadruje podiel správne identifikovaných pozitívnych prípadov z celkového počtu skutočných pozitívnych: Recall $= \frac{TP}{TP+FN}$ [20].

F1-Score – je harmonický priemer precision a recall: F1 $= 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$ [20].

Pre multitriedne nevyvážené problémy sa používajú agregované metriky:

- Macro F1: Neváhovaný priemer F1-score naprieč všetkými triedami, ktorý dáva rovnakú váhu každej triede bez ohľadu na jej veľkosť. Táto metrika je citlivá na výkon na menšinových triedach.
- Weighted F1: Priemer F1-score vážený podľa počtu inštancií v každej triede, ktorý lepšie reflektuje celkový výkon.
- G-Mean: Geometrický priemer pokrytí jednotlivých tried. Pre binárny prípad G-Mean $= \sqrt{\text{Recall}_+ \cdot \text{Recall}_-}$. G-Mean penalizuje modely, ktoré dosahujú dobrý výkon len na niektorých triedach [3].
- AUC-ROC: Plocha pod krivkou ROC, ktorá vyjadruje schopnosť modelu rozlišovať medzi triedami nezávisle od zvoleného prahu [20].

1.5 Evaluácia v dátových prúdoch

Hodnotenie modelov v dátových prúdoch sa zásadne líši od hodnotenia v tradičnom dávkovom spracovaní, kde sa štandardne používa cross-validácia alebo rozdelenie na tréningovú a testovaciu množinu. Tieto prístupy predpokladajú, že všetky dáta sú dostupné naraz a ich rozdelenie je stacionárne, čo v dátových prúdoch neplatí [8].

1.5.1 Holdout evaluácia

Holdout evaluácia v kontexte dátových prúdov používa časovo oddelené bloky dát – model sa trénuje na jednom bloku a testuje na nasledujúcom. Výhodou je jednoduchosť a nezávislosť tréningovej a testovacej množiny. Nevýhodou je, že pri použití malých blokov sú odhady metrík šumové a pri veľkých blokoch sa stráca schopnosť zachytiť lokálne zmeny vo výkone [8].

1.5.2 Prequential evaluácia

Na hodnotenie streamových modelov sa bežne používa prequential (predictive sequential) evaluácia, kde sa každá inštancia najprv použije na testovanie a následne na aktualizáciu modelu [8]. Tento postup, známy aj ako test-then-train, vernejšie simuluje reálne podmienky nasadenia, pretože zachováva časové poradie inštancií a neporušuje predpoklad sekvenčného spracovania. Formálne, pre prúd inštancií $\{(x_1, y_1), (x_2, y_2), \ldots\}$ sa v čase t najprv vykoná predikcia $\hat{y}_t = f_t(x_t)$ aktuálnym modelom f_t, zaznamenajú sa metriky, a následne sa model aktualizuje: $f_{t+1} = \text{update}(f_t, x_t, y_t)$.

Gama, Sebastião a Rodrigues ukázali, že prequential chyba s mechanizmami zabúdania (sliding window alebo fading factors) konverguje k Bayesovej chybe pre konzistentné algoritmy a poskytuje spoľahlivý odhad výkonnosti v nestacionárnom prostredí [8]. Prequential evaluácia s posuvným oknom je preferovaná pred základnou prequential chybou, pretože kumulatívny priemer príliš silno závisí od počiatočného výkonu modelu a pomalšie odráža zmeny [8].

2 Prehľad aktuálneho stavu výskumu

V tejto kapitole sumarizujeme existujúce prístupy a techniky v oblasti klasifikácie dátových prúdov. Zameriavame sa na inkrementálne algoritmy, ensemble metódy, prístupy na zvládanie konceptového driftu a nevyvážených tried. Cieľom je poskytnúť ucelený prehľad aktuálneho stavu výskumu, z ktorého vychádza návrh vlastného riešenia.

2.1 Inkrementálne (online) klasifikačné algoritmy

Inkrementálne algoritmy tvoria základ učenia z dátových prúdov. Na rozdiel od tradičných dávkových modelov sa tieto algoritmy aktualizujú s každou novou inštanciou bez potreby prístupu k celej histórii dát.

2.1.1 Hoeffding Tree

Hoeffding Tree, známy aj ako Very Fast Decision Tree (VFDT), je jedným zo základných a najpoužívanejších algoritmov pre inkrementálnu klasifikáciu dátových prúdov. Bol predstavený Domingosom a Hultenom a využíva Hoeffdingovu nerovnosť na rozhodovanie o rozdelení uzlov so štatistickou zárukou, že výsledný strom je asymptoticky takmer identický so stromom trénovaným na všetkých dátach naraz [7].

Hoeffdingova nerovnosť hovorí, že pre ohraničenú náhodnú premennú s rozsahom R platí, že skutočný priemer sa od pozorovaného priemeru na n vzorkách líši najviac o $\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$ s pravdepodobnosťou aspoň $1-\delta$. Algoritmus využíva túto nerovnosť nasledovne: pre každý list stromu sleduje štatistiky informačného zisku jednotlivých atribútov. Keď je rozdiel informačného zisku medzi dvomi najlepšími atribútmi väčší než ϵ, uzol sa rozdelí podľa najlepšieho atribútu [7].

Kľúčové vlastnosti Hoeffding Tree zahŕňajú: spracovanie každej inštancie práve raz, konštantnú pamäťovú zložitosť na jeden uzol (uchovávajú sa len dostatočné štatistiky, nie samotné inštancie) a asymptotickú konvergenciu k optimálnemu

dávkovému rozhodnutiu. Tieto vlastnosti ho robia ideálnym základným klasifikátorom pre ensemble metódy v dátových prúdoch [17].

Existuje viacero rozšírení Hoeffding Tree. Hoeffding Adaptive Tree (HAT) integruje ADWIN detektor driftu do jednotlivých vetiev stromu, čo umožňuje lokálnu adaptáciu na drift. Extremely Fast Decision Tree (EFDT) zlepšuje kvalitu stromu tým, že umožňuje revíziu predchádzajúcich rozdelení, ak sa s príchodom nových dát ukáže lepšia voľba atribútu [1].

2.1.2 Naïve Bayes a ďalšie inkrementálne klasifikátory

Naïve Bayes je jedným z najjednoduchších a najrýchlejších inkrementálnych klasifikátorov. Jeho inkrementálna aktualizácia spočíva v aktualizácii odhadov podmienených pravdepodobností $P(x_i|y)$ pre každý atribút x_i a triedu y s každou novou inštanciou. Predpoklad podmienečnej nezávislosti atribútov je v praxi často porušený, no napriek tomu Naïve Bayes dosahuje konkurencieschopné výsledky v mnohých aplikáciách [1].

Medzi ďalšie inkrementálne klasifikátory patria online verzie k-NN (kde sa udržiava okno posledných inštancií), inkrementálna logistická regresia (aktualizovaná stochastickým gradientným zostupom) a Stochastic Gradient Trees, ktoré kombinujú výhody rozhodovacích stromov a gradientných metód [1].

2.1.3 Porovnanie inkrementálnych klasifikátorov

Voľba základného klasifikátora pre ensemble model závisí od viacerých faktorov: výpočtovej zložitosti aktualizácie, pamäťových nárokov, schopnosti pracovať s numerickými a kategorickými atribútmi, a citlivosti na nevyvážené triedy. Hoeffding Tree a jeho varianty sú v praxi najčastejšie používané ako základné klasifikátory v ensemble metódach pre dátové prúdy, a to vďaka ich vyváženosti medzi presnosťou, rýchlosťou a flexibilitou [1].

Naïve Bayes je rýchlejší na aktualizáciu a má menšie pamäťové nároky, no jeho predpoklad nezávislosti atribútov obmedzuje výkon na dátach so silnými závislosťami. Inkrementálna logistická regresia je efektívna pre lineárne separovateľné problémy, no jej výkon klesá pri nelineárnych rozhodovacích hraniciach. V kontexte heterogénnych ensemblov, kde sa kombinujú rôzne typy klasifikátorov, je výber diverznej sady základných modelov dôležitejší než výber jedného najlepšieho modelu [1].

2.2 Ensemble metódy pre dátové prúdy

Ensemble učenie kombinuje predikcie viacerých základných modelov s cieľom zvýšiť presnosť a robustnosť. V kontexte dátových prúdov sa ensemble prístupy ukázali ako obzvlášť efektívne, keďže umožňujú dynamickú správu množiny modelov – pridávanie, odstraňovanie a resetovanie jednotlivých členov ako reakciu na zmeny v dátach [1]. Gomes et al. v prehľadovej štúdii kategorizujú tieto metódy podľa spôsobu kombinácie modelov (hlasovanie, vážený priemer), manipulácie s tréningovými dátami (resampling, rozdelenie atribútov) a stratégie adaptácie na zmeny (pasívna periodická aktualizácia alebo aktívna reakcia na signál z detektora driftu) [1].

Historicky prvú skupinu tvoria online verzie klasických batch metód. Oza a Russell preniesli princíp baggingu do streamového prostredia tak, že každú prichádzajúcu inštanciu predložili každému zo M základných klasifikátorov k-krát, kde $k \sim \text{Poisson}(1)$ aproximuje bootstrap vzorkovanie z klasického baggingu [21]. Analogicky bol navrhnutý aj online boosting, v ktorom sa váhy inštancií upravujú podľa chýb predchádzajúcich klasifikátorov v reťazci. V praxi sa však ukázalo, že online bagging na driftujúcich prúdoch často prekonáva online boosting, pretože boosting má tendenciu prispôsobovať sa šumu a zastaraným konceptom [22].

Leveraging Bagging [22] rozširuje Ozov prístup zvýšením parametra λ v Poissonovom rozdelení (typicky na $\lambda = 6$) pre väčšiu diverzitu členov a integráciou detektora ADWIN pre každý strom – pri detekcii zmeny sa príslušný klasifikátor resetuje. Tento princíp – bagging doplnený o detekciu driftu na úrovni jednotlivých modelov – sa stal jedným z najúspešnejších návrhových vzorov v tejto oblasti a tvorí základ viacerých modernejších metód. Jeho najznámejším pokračovateľom je Adaptive Random Forest (ARF) [17], ktorý kombinuje online bagging, náhodný výber podmnožiny atribútov pri rozdeľovaní uzlov a duálny mechanizmus warning/drift detektorov s tzv. background stromami. ARF je v mnohých nezávislých benchmarkoch považovaný za state-of-the-art referenciu a v našej experimentálnej časti slúži ako hlavný baseline. Detailný opis ADWIN, online baggingu a ARF je uvedený v kapitole 3, keďže tieto tri metódy priamo súvisia s návrhom rozšíreného DDCW modelu.

Iná vetva ensemble metód využíva náhodné podpriestory atribútov. Streaming Random Patches [23] priraďuje každému základnému modelu fixnú náhodnú podmnožinu atribútov (na rozdiel od ARF, ktorý ju volí pri každom rozdelení), čo vedie k vyššej diverzite a v niektorých scenároch aj k lepšej výkonnosti

než ARF.

Odlišnú filozofiu adaptácie reprezentujú metódy založené na dynamickom vážení členov. Dynamic Weighted Majority (DWM) [24] reaguje na chyby jednotlivých modelov postupným znižovaním ich váh a odstraňovaním modelov pod prahom, pričom periodicky pridáva nové modely natrénované na najnovších dátach – drift teda adresuje implicitne, bez explicitného detektora. Príbuzné Accuracy Weighted Ensemble (AWE) [6] a skorší Streaming Ensemble Algorithm (SEA) [25] pracujú v chunk-based režime: po spracovaní každého bloku dát sa najslabší člen nahradí novým klasifikátorom trénovaným na najnovšom bloku. Princíp adaptácie klasického boostingu na nestacionárne prostredia s možnosťou znovupoužitia starých modelov pri rekurentnom drifte priniesol Learn++.NSE [26].

Rozsiahla porovnávacia štúdia De Barrosa a Santosa [16] ukázala, že naprieč rôznymi typmi driftu dosahujú konzistentne najlepšie výsledky metódy kombinujúce bagging s explicitnou detekciou driftu (ARF, Leveraging Bagging), čo motivuje aj náš výber baselinov v experimentálnej časti.

2.3 Ensemble prístupy pre nevyvážené dátové prúdy

Kombinácia konceptového driftu a nevyvážených tried je jednou z najaktuálnejších výziev učenia z dátových prúdov. Metódy zo sekcie 2.2 sú síce účinné pri zvládaní driftu, no nemajú zabudované mechanizmy na vyvažovanie tried, a pri silnej nevyváženosti často klesá najmä recall menšinových tried [3].

Prvú skupinu tvoria rozšírenia online baggingu o per-class úpravu Poissonovho parametra. Wang a kol. navrhli OOB (Oversampling-Based Online Bagging) a UOB (Undersampling-Based Online Bagging), ktoré používajú Poisson(λ) s rôznymi λ pre väčšinovú a menšinovú triedu – $\lambda > 1$ pre menšinu simuluje oversampling, $\lambda < 1$ pre väčšinu undersampling [3]. Tieto metódy sú jednoduché a výpočtovo lacné, no fixné hodnoty λ nezohľadňujú meniaci sa pomer tried v čase.

Dynamickejšiu adaptáciu ponúka ROSE (Robust Online Self-Adjusting Ensemble) [4], ktorý navrhli Cano a Krawczyk špeciálne pre nevyvážené driftujúce prúdy. Model kombinuje adaptívne vyvažovanie tried podľa aktuálneho imbalance ratio, online resampling integrovaný do baggingu a ADWIN-based detekciu driftu. V súčasnosti patrí medzi najrelevantnejšie referencie pre imbalanced streaming; detailnejšie je opísaný v kapitole 3.

Pre multitriedne prúdy so súbežným driftom navrhli Junaid et al. metódu SAEM (Smart Adaptive Ensemble Model) [27], ktorá monitoruje zmeny na úrovni

atribútov, pri detekcii zmeny vytvára background ensemble a uplatňuje per-class dynamické váhy podľa aktuálneho pomeru nevyváženosti. Iný smer reprezentuje Dynamic Ensemble Selection podľa Jiaa et al. [28], ktorá pre každú predikciu vyberá podmnožinu ensemble členov najvhodnejších pre lokálne susedstvo inštancie, čo pomáha najmä v oblastiach s prekryvom tried.

Najrozsiahlejšiu doterajšiu porovnávaciu štúdiu v tejto oblasti predstavili Aguiar, Krawczyk a Cano [3], ktorí vyhodnotili 24 state-of-the-art algoritmov na 515 dátových prúdoch kombinujúcich statickú aj dynamickú nevyváženosť, rôzne typy driftu a binárne aj multitriedne scenáre na reálnych aj semi-syntetických dátach. Ich hlavný záver je, že žiadna jednotlivá metóda nedominuje naprieč všetkými scenármi, a že najlepšie konzistentné výsledky dosahujú ensemble metódy v kombinácii s technikami vyvažovania tried. Štúdia zároveň poskytuje reprodukovateľný experimentálny framework, z ktorého čerpáme aj v tejto práci.

2.4 Súčasné výzvy a otvorené problémy

Z prehľadu literatúry vyplývajú viaceré otvorené problémy, ktoré motivujú výskum v tejto práci:

- Súčasný výskyt driftu a nevyváženosti: Väčšina existujúcich metód rieši tieto problémy oddelene alebo s obmedzenou integráciou. Metódy, ktoré ich rieši spoločne a dynamicky, sú stále predmetom aktívneho výskumu. Ako ukázali Aguiar, Krawczyk a Cano, interakcia medzi driftom a nevyváženosťou vytvára špecifické komplikácie, ktoré nie sú prítomné pri riešení každého problému zvlášť – napríklad drift v pomere tried, maskovaný drift v menšinových oblastiach priestoru alebo interferencia resamplingu s detekciou driftu [3].

- Multitriedna nevyváženosť: Prevažná časť výskumu sa zameriava na binárnu klasifikáciu. Rozšírenie na multitriedny prípad s rôznymi mierami nevyváženosti medzi jednotlivými triedami zostáva náročné, pretože vyžaduje per-class adaptáciu mechanizmov vyvažovania. SAEM model rieši tento problém čiastočne pomocou dynamických váh na menšinové triedy, no výsledky ukazujú priestor pre zlepšenie [27].

- Ultra-zriedkavé triedy: Triedy s veľmi malým počtom inštancií (desiatky z miliónov) sú obzvlášť problematické. Štandardné techniky vyvažovania (SMOTE, oversampling) nemajú dostatok vzoriek na efektívnu interpoláciu

alebo generovanie. V takýchto prípadoch je potrebné kombinovať data-level prístupy s algorithm-level adaptáciou [3].

- Textové dátové prúdy: Klasifikácia textových dátových prúdov prináša ďalšie výzvy – vysoká dimenzionalita, riedke príznaky, kontextová závislosť a rýchla evolúcia jazyka. Väčšina výskumu v oblasti streamových ensemblov sa zameriava na numerické dáta, zatiaľ čo textové prúdy zo sociálnych médií (detekcia toxicity, fake news) predstavujú rastúcu aplikačnú doménu s potenciálom pre adaptívne modely [5].

- Hodnotenie a porovnávanie: Chýbajú štandardizované benchmarky a evaluačné protokoly. Rôzne štúdie používajú rôzne datasety, metriky a evaluačné postupy, čo sťažuje porovnávanie výsledkov. Voľba metrík je obzvlášť dôležitá – accuracy je pre nevyvážené dáta neadekvátna, no aj F1 a G-Mean majú rôzne vlastnosti pri rôznych mierach nevyváženosti [3].

- Heterogénne ensemble modely: Väčšina existujúcich ensemble metód pre prúdy je homogénna, t.j. používa rovnaký typ základného klasifikátora (typicky Hoeffding Tree). Potenciál heterogénnych ensemblov, kombinujúcich rôzne typy inkrementálnych modelov, zostáva nedostatočne preskúmaný. Heterogénny prístup ale môže byť konkurencieschopný s homogénnymi metódami pri nižšej výpočtovej náročnosti [6].

- Výpočtová efektivita: Reálne nasadenie adaptívnych ensemble modelov vyžaduje kompromis medzi výkonom a výpočtovými nárokmi. Komplexné mechanizmy (replay, augmentácia, drift detekcia) zvyšujú presnosť, no zároveň spomaľujú spracovanie. Optimalizácia tohto kompromisu je prakticky dôležitá pre streaming aplikácie s vysokou rýchlosťou dát [1].

Práve kombinácia viacerých z týchto otvorených problémov – heterogénny ensemble s integrovaným zvládaním nevyváženosti a driftu, testovaný na binárnych aj multiclass numerických aj textových prúdoch – je hlavnou motiváciou pre návrh modelu IDDCW v tejto práci.

3 Detailná analýza vybraných prístupov

Táto kapitola sa venuje algoritmom a technikám, na ktoré priamo nadväzuje návrh rozšíreného DDCW modelu v kapitole 4. Najprv opíšeme pôvodný DDCW model [6], ktorý v tejto práci rozširujeme — popíšeme jeho architektúru, mechanizmus diverzity a váhovania, aby sa kapitola 4 mohla sústrediť už len na naše modifikácie. Ďalej opíšeme online bagging ako základ väčšiny streamových ensemble metód a Adaptive Random Forest, ktorý slúži ako hlavný baseline v experimentálnej časti. Napokon rozoberieme Page-Hinkley test, ktorý v rozšírenom DDCW používame ako detektor driftu. Záver kapitoly tvorí prehľad aplikačných domén a identifikácia výskumnej medzery, na ktorú táto práca reaguje.

3.1 ADWIN – adaptívne okno pre detekciu zmien

ADWIN (Adaptive Windowing) je algoritmus pre detekciu zmien v dátových prúdoch navrhnutý Bifetom a Gavaldàom [15]. Predstavuje jednu z najpoužívanejších a najdôkladnejšie teoreticky podložených metód detekcie driftu.

3.1.1 Princíp fungovania

ADWIN udržiava okno W posledných pozorovaní a pre každé možné rozdelenie okna na dve súvislé podokná W_0 (staršie dáta) a W_1 (novšie dáta) testuje, či sa ich priemery štatisticky významne líšia. Formálne, ADWIN detekuje zmenu, ak:

$$|\hat{\mu}_{W_0} - \hat{\mu}_{W_1}| \geq \epsilon_{\text{cut}}$$

kde $\hat{\mu}_{W_0}$ a $\hat{\mu}_{W_1}$ sú priemery pozorovaní v oboch podoknách a ϵ_{cut} je hranica odvodená z Hoeffdingovej nerovnosti závislá na veľkostiach okien a požadovanej úrovni spoľahlivosti δ. Ak je zmena detekovaná, staršia časť okna (W_0) sa zahodí [15].

3.1.2 Exponenciálne histogramy

Naivná implementácia ADWIN by vyžadovala uchovávanie všetkých individuálnych pozorovaní v okne, čo by malo lineárnu pamäťovú zložitosť $O(W)$. Bifet a Gavaldà navrhli efektívnu implementáciu využívajúcu exponenciálne histogramy, kde sa staršie pozorovanía agregujú do stále väčších skupín (buckets). Každý bucket uchováva len súčet a rozptyl pozorovaní, nie individuálne hodnoty. Vďaka tomu ADWIN dosahuje pamäťovú zložitosť $O(\log W)$ a amortizovanú časovú zložitosť $O(\log W)$ na jednu inštanciu, čo ho robí vhodným aj pre veľmi dlhé dátové prúdy [15].

3.1.3 Teoretické záruky

Na rozdiel od mnohých heuristických detektorov driftu, ADWIN poskytuje formálne záruky: pravdepodobnosť falošne pozitívnej detekcie je zhora ohraničená parametrom δ a pravdepodobnosť nedetekovanej zmeny je explicitne kontrolovateľná [15]. Tieto záruky robia ADWIN vhodným nielen pre samostatnú detekciu driftu, ale aj ako komponent v ensemble metódach, kde sa používa na monitorovanie výkonnosti jednotlivých základných modelov (napr. v ARF a Leveraging Bagging).

3.2 Pôvodný DDCW model

Diversified Dynamic Class-Weighted (DDCW) ensemble je chunk-based heterogénny adaptívny klasifikátor pre dátové prúdy s konceptovým driftom. Jeho hlavným príspevkom je kombinácia troch myšlienok, ktoré sa v predchádzajúcich metódach zvyčajne objavovali oddelene: heterogénneho zloženia ensemble, dynamického per-class váhovania členov a explicitného kritéria diverzity založeného na Q-štatistike pri správe expertov [6].

3.2.1 Architektúra a heterogénne zloženie ensemble

Model pracuje s množinou m ensemble členov $e_1, \ldots, e_m$ a veľkosť ensemble je dynamicky ohraničená dvoma parametrami — minimálnou veľkosťou k a maximálnou veľkosťou l. Pri vytváraní nového experta sa jeho typ volí náhodne zo zoznamu podporovaných základných klasifikátorov (v pôvodnej publikácii Naïve Bayes, Hoeffding Tree a k-NN). Práve heterogénne zloženie — kombinovanie rôznych typov base learnerov v jednom ensemble — je jedným z hlavných

motivačných prvkov DDCW, keďže väčšina streamových ensemblov (napríklad ARF) zostáva homogénna.

3.2.2 Matica váh a adaptívne per-class váhovanie

Váhy členov ensemble sú uložené v matici $W \in R^{m \times c}$, kde m je počet expertov a c počet cieľových tried. Prvok $w_{i,j}$ reprezentuje váhu experta e_i pri predikovaní triedy j. Na začiatku sú váhy inicializované rovnomerne. Pre každú prichádzajúcu inštanciu z aktuálneho chunku:

- každý expert e_i predikuje triedu;
- ak je predikcia správna, príslušná váha $w_{i,j}$ sa vynásobí koeficientom β (v pôvodnej práci implicitne $\beta = 3$), čím sa expert posilňuje pre danú triedu;
- globálna predikcia sa určí ako argmax súčtov váh naprieč predikovanými triedami.

Matica W teda zároveň slúži na dve veci: na hlasovanie pri predikcii a ako ukazovateľ sily jednotlivých expertov — súčet $\sum_j w_{i,j}$ vyjadruje celkové skóre experta e_i.

3.2.3 Q-štatistika ako miera párovej diverzity

Po spracovaní každého chunku model vypočíta Q-štatistiku párovej diverzity [6] medzi každou dvojicou expertov. Pre dva klasifikátory D_i a D_k je Q-štatistika definovaná ako:

$$Q_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}$$

kde N^{ab} je počet inštancií, pri ktorých klasifikátor D_i dosiahol výsledok a a klasifikátor D_k výsledok b (pričom 1 znamená správnu predikciu a 0 nesprávnu). Q-štatistika nadobúda hodnoty z intervalu $[-1, 1]$: pozitívne hodnoty znamenajú, že oba klasifikátory robia rovnaké chyby, záporné naznačujú, že sa navzájom dopĺňajú. Diverzita jedného člena sa potom získa ako priemer párových hodnôt voči ostatným:

$$\text{div}_{e_i} = \frac{1}{m-1} \sum_{k=1, k \neq i}^{m} Q_{i,k}$$

Táto hodnota je použitá pri aktualizácii váh — členovia s vyššou diverzitou (nižšou Q-štatistikou voči zvyšku ensemble) sú preferovaní. Takto model bráni konvergencii k množine podobne rozhodujúcich expertov.

3.2.4 Lifetime koeficient a postupné zabúdanie

Okrem výkonnosti a diverzity DDCW zohľadňuje aj vek expertov. Každému expertovi je priradený lifetime koeficient T_i, ktorý sa zvyšuje s každou periódou a exponenciálne (s parametrom α, implicitne $\alpha = 0{,}002$) znižuje váhy starších členov. Mechanizmus funguje ako postupné zabúdanie: na začiatku života experta je efekt minimálny, no pre dlho žijúcich členov sa váhy výraznejšie tlmia, čo uvoľňuje priestor pre nové modely trénované na aktuálnych konceptoch.

3.2.5 Správa expertov a parameter θ

Po každej perióde model:

1. aktualizuje váhy na základe správnosti predikcie, diverzity a lifetimu;
2. normalizuje váhy pre každú triedu;
3. odstráni expertov, ktorých celkové skóre klesne pod prah θ (implicitne $\theta = 0{,}02$);
4. ak je ensemble plný a globálna predikcia bola nesprávna, nahradí najslabšieho člena novo inicializovaným expertom;
5. ak je veľkosť ensemble pod k, dopĺňa nových členov.

3.2.6 Limity pôvodného modelu

Experimenty publikované v [6] ukázali, že DDCW dosahuje konkurencieschopné výsledky oproti state-of-the-art metódam s rozumnou spotrebou výpočtových zdrojov. Zároveň však autori sami konštatujú, že model má problémy na silno nevyvážených datasetoch ako KDD99 či Airlines, kde F1-skóre výrazne zaostáva za accuracy — symptóm charakteristický pre modely ignorujúce menšinové triedy. Pôvodný DDCW totiž rieši primárne konceptový drift a nemá mechanizmy cielene na imbalance: všetky triedy majú rovnakú odmenu β bez ohľadu na to, či sú majoritné alebo minoritné, chýba replay menšinových vzoriek, a chunk-based spracovanie je menej reaktívne než instance-based. Práve tieto tri limity tvoria hlavnú motiváciu pre rozšírenie modelu, ktoré predkladáme v kapitole 4.

3.3 Online Bagging a jeho varianty – detailná analýza

Online Bagging a jeho varianty tvoria základ väčšiny ensemble metód pre dátové prúdy. Pochopenie ich princípov je kľúčové pre návrh nových ensemble modelov.

3.3.1 Od batch baggingu k online baggingu

Klasický bagging navrhnutý Breimanom vytvára M bootstrap vzoriek z pôvodného datasetu veľkosti N a pre každú bootstrap vzorku trénuje jeden základný model. Predikcia ensemble je väčšinovým hlasovaním všetkých modelov. Bootstrap vzorkovanie zabezpečuje, že jednotlivé modely sú trénované na mierne odlišných dátach, čo generuje diverzitu – kľúčový faktor úspechu ensemble metód [21].

V bootstrap vzorke veľkosti N z datasetu veľkosti N sa každá inštancia vyskytne k-krát, kde k sleduje binomické rozdelenie $\text{Bin}(N, 1/N)$. Pre veľké N toto konverguje k $\text{Poisson}(1)$. Oza a Russell využili túto konvergenciu na návrh online verzie: pre každú novú inštanciu a každý model vygenerujú váhu $k \sim \text{Poisson}(1)$ a inštanciu poskytnú modelu k-krát [21].

3.3.2 Vplyv parametra λ na diverzitu

Leveraging Bagging ukázal, že použitie $\text{Poisson}(\lambda)$ s $\lambda > 1$ zvyšuje diverzitu ensemble [22]. Vyššia hodnota λ spôsobuje, že niektoré inštancie sú použité s výrazne vyššou váhou, čo simuluje agresívnejšie vzorkovanie s opakovaním. Bifet et al. experimentálne zistili, že $\lambda = 6$ je dobrý kompromis medzi diverzitou a stabilitou [22].

Tento princíp je relevantný aj pre nevyvážené dáta – použitie rôznych hodnôt λ pre rôzne triedy môže slúžiť ako forma online resamplingu, kde menšinové triedy dostávajú vyššiu váhu.

3.3.3 Integrácia detekcie driftu

OzaBaggingADWIN rozširuje štandardný online bagging o ADWIN detektor pre každý základný model. Keď ADWIN detekuje zmenu vo výkonnosti konkrétneho modelu, model sa resetuje. Tento prístup umožňuje lokálnu adaptáciu – ak drift ovplyvňuje len niektoré časti priestoru, len príslušné modely sa adaptujú, zatiaľ čo ostatné zostávajú stabilné [22].

3.4 Adaptive Random Forest

ARF je v súčasnosti jedným z najpoužívanejších a najcitovanejších ensemble algoritmov pre dátové prúdy. Jeho úspech vychádza z kombinácie niekoľkých osvedčených princípov [17].

3.4.1 Architektúra ARF

ARF udržiava ensemble M Hoeffding Tree klasifikátorov. Každý strom h_m je trénovaný na celom dátovom prúde, ale s dvoma zdrojmi randomizácie: (1) online bootstrap s Poissonovým vzorkovaním Poisson(λ) a (2) náhodný výber $\sqrt{d}$ atribútov z celkových d pri každom rozdelení uzla. Predikcia ensemble je väčšinovým hlasovaním všetkých stromov [17].

3.4.2 Mechanizmus background trees

Každý strom h_m v ARF má priradený warning detektor d_m^w a drift detektor d_m^d. Oba sú typicky implementované ako ADWIN s rôznymi hodnotami parametra δ – warning detektor má vyššiu citlivosť ($\delta_w > \delta_d$). Proces adaptácie prebieha v troch fázach [17]:

1. Normálna prevádzka: Strom h_m sa inkrementálne aktualizuje s každou novou inštanciou. Detektory monitorujú jeho chybovosť.
2. Warning fáza: Keď warning detektor d_m^w signalizuje varovanie, vytvorí sa background tree $\tilde{h}_m$, ktorý sa začne trénovať na nových dátach. Pôvodný strom h_m pokračuje v prevádzke.
3. Drift potvrdený: Keď drift detektor d_m^d potvrdí drift, background tree $\tilde{h}_m$ nahradí pôvodný strom h_m. Ak sa varovanie nepotvrdí, $\tilde{h}_m$ sa zahodí.

Výsledkom je, že nový strom má čas sa natrénovať pred nasadením, čo zabraňuje dočasnému poklesu výkonu, ktorý by nastal pri okamžitom resete.

3.4.3 Silné a slabé stránky ARF

ARF dosahuje vynikajúce výsledky na dátových prúdoch s rôznymi typmi driftu a je robustný voči šumu. Jeho hlavnou nevýhodou sú pomerne vysoké pamäťové nároky (každý strom + potenciálne background trees) a homogénnosť ensemble (všetky základné modely sú Hoeffding Trees). Gomes et al. navrhli aj paralelnú implementáciu ARF, kde sú stromy a adaptívne operátory nezávislé, čo umožňuje efektívnu paralelizáciu [17].

3.5 Page-Hinkley test

Page-Hinkley (PH) test [29] je klasický sekvenčný test zmien v strednej hodnote signálu, ktorý v rozšírenom DDCW modeli používame ako detektor koncepto-

vého driftu nad chybovosťou ensemble. Pôvodne bol navrhnutý v oblasti kontroly kvality a zmenil sa do štandardného nástroja aj v oblasti detekcie driftu v dátových prúdoch [2].

Pre pozorovaný signál $x_1, x_2, \ldots$ (v našom prípade binárny indikátor chyby predikcie: $x_t = 1$ pri nesprávnej predikcii, 0 pri správnej) PH test udržiava priebežný odhad priemeru $\bar{x}_t$ a kumulatívny súčet odchýlok S_t:

$$S_t = \sum_{i=1}^{t} (x_i - \bar{x}_i - \delta)$$

kde $\delta > 0$ je parameter, ktorý tolerovane posúva nulovú hladinu a chráni detektor pred reakciou na drobné fluktuácie. Zmena je detekovaná, keď rozdiel medzi aktuálnym S_t a jeho doterajším minimom presiahne prah λ:

$$S_t - \min_{1 \leq j \leq t} S_j > \lambda$$

Parameter δ tak reguluje citlivosť (menšie $\delta \rightarrow$ rýchlejšia reakcia, ale viac falošných pozitív), zatiaľ čo λ nastavuje úroveň istoty pri potvrdení zmeny. Pamäťové nároky testu sú konštantné, $O(1)$ — stačí udržiavať priebežný priemer, S_t a jeho minimum.

Hlavnou výhodou PH testu oproti ADWIN je jeho jednoduchosť a nízka réžia pri každej inštancii, čo je dôležité pri instance-based spracovaní s heterogénnym ensemble. Nevýhodou je absencia formálnych záruk na falošne pozitívnu mieru, ktoré ADWIN poskytuje cez Hoeffdingovu nerovnosť — voľba parametrov δ a λ je preto v praxi viac empirická. V kontexte rozšíreného DDCW nás však zaujíma chybovosť celého ensemble, nie jednotlivých expertov (tí sú riadení samostatným váhovým systémom), a konštantná pamäťová zložitosť PH testu sa hodí k tomu, že detektor sa aktualizuje pri každej vzorke. Detailné nastavenie parametrov a post-drift reakciu popisujeme v sekcii 4.2.5 v rámci návrhu rozšíreného modelu.

3.6 Aplikačné domény – detekcia toxického správania a falošných správ

Jednou z praktických domén, kde sa stretávajú všetky vyššie uvedené výzvy, je detekcia toxického správania v online prostredí. Dáta zo sociálnych médií tvoria prirodzené dátové prúdy s konceptovým driftom (jazyk sa vyvíja, témy sa menia) a výraznou nevyváženosťou tried (toxický obsah tvorí menšinu) [5].

Herodotou, Chatzakou a Kourtellis navrhli streamovací framework pre detekciu agresie na Twitteri, ktorý dosahuje vysokú presnosť (nad 90 %) a je škálovateľný na milióny tweetov denne. Ich práca zdôrazňuje praktické výzvy dynamického online prostredia a potrebu inkrementálnej adaptácie modelov na meniace sa vzorce agresívneho správania [30]. Výrazným problémom v tejto doméne je, že používatelia aktívne prispôsobujú svoj jazyk, aby obchádzali detekčné mechanizmy, čo vytvára adversariálny konceptový drift.

V oblasti ensemble prístupov pre textovú klasifikáciu Kamateri a Salampasis navrhli flexibilný doménovo-agnostický ensemble framework a experimentálne potvrdili, že diverzita základných modelov je kľúčovým faktorom úspechu ensemble systému [31]. Ďalšie práce sa venujú detekcii kyberšikany s využitím kombinácie deep learning modelov a bio-inšpirovaných optimalizačných metód [32], klasifikácii toxického textu pomocou kapsulových sietí s BiLSTM a GRU s využitím pomocných meta-príznakov [33], analýze toxických komentárov v prostredí online vzdelávania [34] a interpretovateľnosti modelov pre detekciu toxicity s využitím token-level predikcií [35].

Ďalšou relevantnou doménou je detekcia falošných správ (fake news), kde konceptový drift vzniká prirodzene zmenou diskutovaných tém. Napríklad prechod z volebnej tematiky na pandémiu COVID-19 predstavuje výrazný posun v jazyku, štýle aj obsahu príspevkov. Modely trénované na politických textoch nemusia byť účinné na zdravotníckej dezinformácii a naopak. Táto doména je obzvlášť vhodná na testovanie adaptívnych modelov, pretože zmeny tém sú často náhle a výrazné, čo umožňuje pozorovať reakciu modelov na reálny konceptový drift. Zároveň distribúcia tried (falošné vs. pravdivé správy) nemusí byť vyvážená a môže sa meniť v závislosti od aktuálnych udalostí.

3.7 Zhrnutie a výskumná medzera

Na základe analýzy vybraných prístupov vyplývajú tieto zistenia, ktoré priamo motivujú návrh rozšíreného DDCW modelu:

1. Ensemble metódy sú najúčinnejším prístupom pre klasifikáciu driftujúcich dátových prúdov, pričom bagging-based metódy s explicitnou detekciou driftu (ARF, Leveraging Bagging) dosahujú konzistentne najlepšie výsledky [16].

2. Pôvodný DDCW model priniesol originálnu kombináciu heterogénneho ensemble, per-class váhovania a Q-diverzity, no sám autori priznávajú jeho slabú výkonnosť na silno nevyvážených datasetoch (KDD99, Airlines) [6]. Explicitné mechanizmy na riešenie imbalance v modeli chýbajú.

3. Väčšina existujúcich ensemble metód pre prúdy je homogénna (typicky Hoeffding Trees). Potenciál heterogénnych ensemblov ako v DDCW zostáva v širšom meradle nedostatočne preskúmaný [1].

4. Metódy, ktoré integrujú zvládanie driftu a nevyváženosti v jednom konzistentnom systéme, dosahujú lepšie výsledky než oddelené riešenie oboch problémov [3].

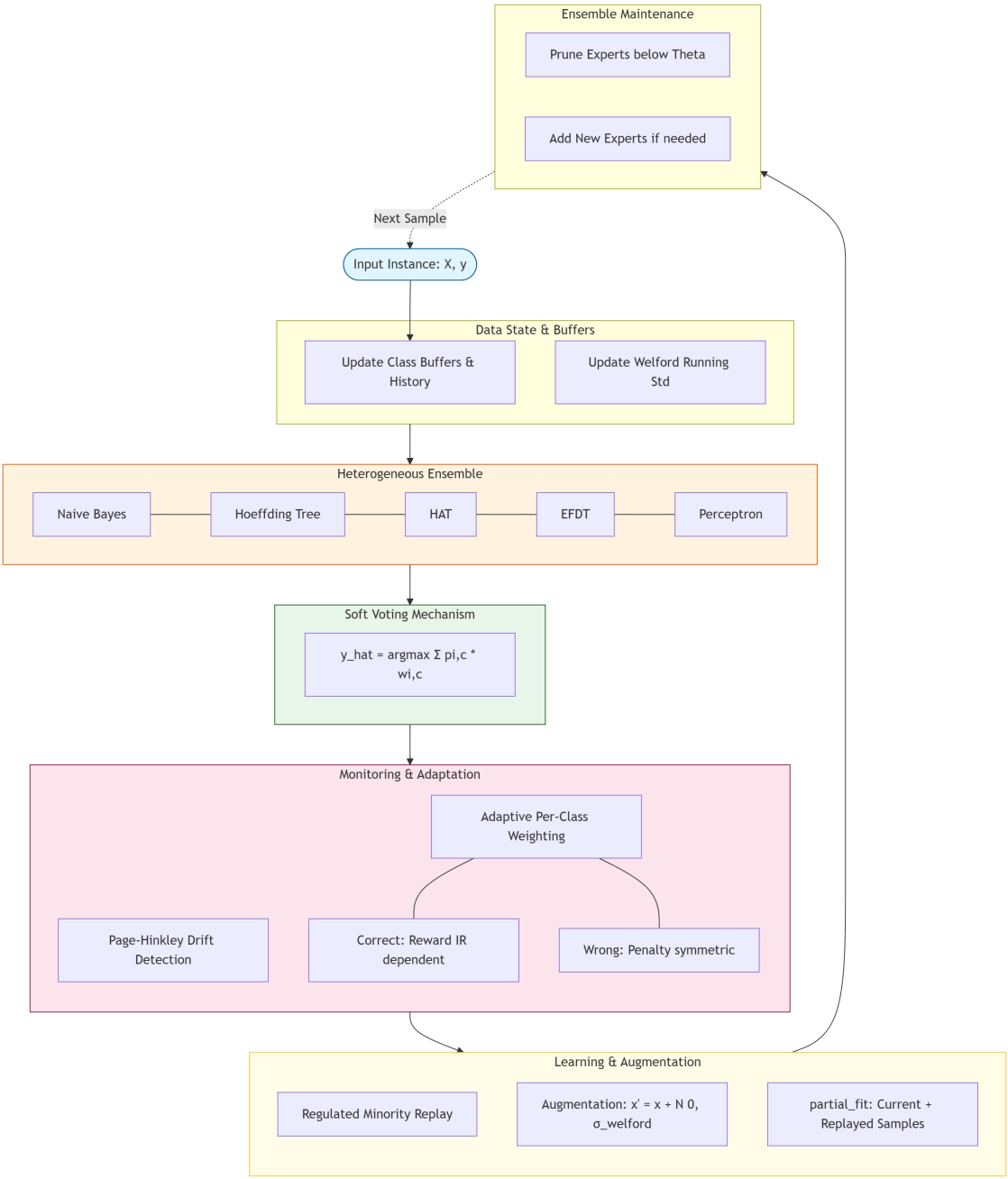
5. Page-Hinkley test ponúka alternatívu k štandardnému ADWIN detektoru pri nižšej výpočtovej réžii, čo je praktická výhoda pri instance-based spracovaní v heterogénnom ensemble.

Navrhovaný model IDDCW, ktorý predstavujeme v kapitole 4, vychádza z týchto zistení. Nadväzuje na pôvodný DDCW [6] a zapĺňa identifikovanú výskumnú medzeru: heterogénny ensemble kombinujúci rôzne typy základných klasifikátorov s integrovaným mechanizmom detekcie driftu a dynamickým váhovaním citlivým na aktuálny pomer tried a výkonnosť na menšinových triedach.

4 Návrh a implementácia modelu IDDCW

V tejto kapitole predstavujeme IDDCW (Imbalance-aware Dynamic Diversified Class-Weighted ensemble) — nový model, ktorý nadväzuje na pôvodný DDCW [6] a rozširuje ho o mechanizmy cielené na nevyvážené dátové prúdy. Pôvodný DDCW, detailne popísaný v sekcii 3.2, bol navrhnutý predovšetkým na zvládanie konceptového driftu — pracoval s heterogénnym ensemble, dynamicky váhoval členov podľa tried a využíval Q-štatistiku diverzity. Nevyváženosť tried v ňom ale explicitne riešená nebola. IDDCW si z DDCW ponecháva heterogénne zloženie ensemble a per-class váhovaciu schému, no prechádza z chunk-based na instance-based spracovanie a pridáva adaptívne per-class váhovanie závislé od aktuálneho pomeru tried, replay a augmentáciu menšinových vzoriek so spätnou väzbou na majority recall (aby sa nestalo, že v snahe zlepšiť minority recall model prestane rozoznávať väčšinovú triedu) a Page-Hinkley detektor driftu s cielenou post-drift reakciou.

Dostávame sa k jadru práce — samotnému návrhu modelu IDDCW. Celkovú architektúru modelu zobrazuje Obrázok 4.1.



Obr. 4.1: Schéma architektúry modelu IDDCW. Proces začína spracovaním vstupnej inštancie (X, y), ktorá aktualizuje dátové buffery a Welfordov priebežný odhad smerodajnej odchýlky pre potreby augmentácie. Heterogénny ensemble piatich expertov generuje predikcie, ktoré sú následne agregované mechanizmom soft votingu podľa váženého súčtu pravdepodobností. Modul monitorovania a adaptácie dynamicky upravuje per-class váhy expertov – udeľuje odmeny závislé od miery nerovnováhy (IR) alebo symetrické penalizácie – a zároveň využíva Page-Hinkley test na detekciu konceptuálneho driftu. Fáza učenia kombinuje štandardný `partial_fit` s regulovaným replayom minoritných vzoriek a Gaussovskou augmentáciou. Cyklus uzatvára modul údržby, ktorý redukuje expertov s váhou pod prahom θ a v prípade potreby dopĺňa ensemble o nových členov.

4.1 Motivácia pre nový model

Pôvodný DDCW bol primárne navrhnutý na zvládanie konceptového driftu, pričom nevyváženosť tried nebola explicitne riešená. V experimentoch na silno nevyvážených datasetoch (napr. KDD99, Shuttle) autori sami priznávajú [6], že model trpí poklesom F1-skóre — symptóm charakteristický pre modely ignorujúce menšinové triedy. V pôvodnom modeli sa prejavili nasledovné obmedzenia:

- Dynamické per-class váhovanie síce čiastočne zohľadňuje nevyváženosť, no neobsahuje explicitný mechanizmus na zvýšenie recall menšinových tried.
- Absencia replay alebo augmentácie menšinových vzoriek spôsobuje, že pri extrémnom imbalance (pomer > 10:1) model nemá dostatok menšinových vzoriek na efektívne učenie.
- Chunk-based spracovanie je menej reaktívne než instance-based spracovanie, čo spomaľuje reakciu na drift.

Model IDDCW rieši tieto obmedzenia prechodom na instance-based spracovanie a integráciou mechanizmov na zvládanie nevyváženosti tried.

4.2 Architektúra modelu IDDCW

IDDCW zachováva heterogénnu architektúru pôvodného DDCW, no prechádza na instance-based online spracovanie a pridáva viaceré nové mechanizmy. Celková schéma modelu je nasledovná: pre každú prichádzajúcu inštanciu (x_s, y_s) model najprv vykoná predikciu pomocou soft votingu, aktualizuje váhy expertov na základe správnosti predikcie, voliteľne vykoná replay menšinových vzoriek a natrénuje všetkých expertov na aktuálnej inštancii.

4.2.1 Základné klasifikátory

Model využíva heterogénnu sadu základných klasifikátorov implementovaných v scikit-multiflow:

- Naïve Bayes: Rýchly inkrementálny klasifikátor, vhodný pre dáta s menším počtom príznakov.
- Hoeffding Tree: Štandardný inkrementálny rozhodovací strom s konfigurovateľným grace periodom [7].

- Hoeffding Adaptive Tree: Rozšírenie HT s ADWIN detekciou driftu na úrovni vetiev stromu.
- Extremely Fast Decision Tree (EFDT): Vylepšený HT umožňujúci revíziu predchádzajúcich rozdelení.
- Perceptron: Lineárny klasifikátor s konfigurovateľnou rýchlosťou učenia.

Pri vytváraní nového experta sa náhodne vyberá typ z tejto sady a hyperparametre sa randomizujú (napr. grace period pre stromy v rozsahu 20–200), čím sa zvyšuje diverzita ensemble.

4.2.2 Soft voting mechanizmus

Na rozdiel od pôvodného DDCW, ktorý používal hard voting (argmax z predikcií expertov), IDDCW využíva soft voting. Pre každého experta e_i sa získa vektor pravdepodobností $\mathbf{p}_i = \text{predict\_proba}(x)$ a výsledná predikcia sa vypočíta ako:

$$\hat{y} = \arg\max_{c} \sum_{i=1}^{m} p_{i,c} \cdot w_{i,c}$$

kde $p_{i,c}$ je pravdepodobnosť triedy c predikovaná expertom e_i a $w_{i,c}$ je per-class váha experta e_i pre triedu c. Soft voting zabraňuje extrémnym rozhodnutiam, kde expert s 51% istotou hlasuje rovnako silno ako expert s 99% istotou.

4.2.3 Adaptívne váhovanie tried

Váhy expertov sú dynamicky aktualizované po každej inštancii. Pri správnej predikcii sa váha pre predikovanú triedu násobí koeficientom mult $= 1 + \beta \cdot r$, kde r je reward závisiacI na tom, či ide o menšinovú alebo väčšinovú triedu:

- Pri miernom imbalance (ratio < 3): symetrické odmeny $r_{\text{min}} = 0.12$, $r_{\text{maj}} = 0.10$.
- Pri extrémnom imbalance (ratio > 8): asymetrické odmeny $r_{\text{min}} = 0.20$, $r_{\text{maj}} = 0.08$.
- V pásme ratio 3–8: lineárna interpolácia medzi oboma režimami.

Penalty za nesprávnu predikciu je symetrická: mult $= 1 - \beta \cdot p$, kde $p = 0.04$ pre menšinovú vzorku a $p = 0.03$ pre väčšinovú. Táto adaptívna asymetria zabezpečuje, že pri extrémnom imbalance model viac odmeňuje správne predikcie menšinových tried, zatiaľ čo pri miernom imbalance zostáva váhovanie približne symetrické.

4.2.4 Replay a augmentácia menšinových vzoriek

Kľúčovým novým mechanizmom je replay menšinových vzoriek z class bufferov. Pre každú menšinovú triedu si model udržiava cyklický buffer posledných vzoriek tejto triedy. Po spracovaní každej inštancie model voliteľne vyberie k náhodných menšinových vzoriek z bufferov a natrénuje na nich všetkých expertov.

Intenzita replay (k) je regulovaná troma faktormi:

1. Logaritmické ohraničenie podľa imbalance ratio: $k_{\max} = \lfloor \log_2(\max(2, \text{ratio})) \rfloor$. Dôsledkom je, že replay je proporcionálny k závažnosti nevyváženosti.

2. Spätná väzba majority recall: Ak majority recall odhadovaný z posuvného okna posledných vzoriek klesne pod adaptívny prah, replay sa tlmí alebo úplne vypína. Prahy sú adaptívne podľa imbalance ratio – striktnejšie pri miernom imbalance, voľnejšie pri extrémnom.

3. Post-drift boost: Po detekcii driftu sa replay dočasne zosilňuje na urýchlenie adaptácie.

V režime augmentácie sa ku každej replay vzorke pridáva Gaussovský šum s rozptylom úmerným feature-wise štandardnej odchýlke dát, vypočítanej efektívne pomocou Welfordovho online algoritmu [36]. Sila augmentácie je regulovaná imbalance ratiom (vypnutá pri ratio < 2) a majority recall spätnou väzbou.

Triedy s menej ako min_replay_support vzorkami v bufferi sú vynechané z replay, pretože replay z príliš malého počtu vzoriek vytvára iba šumový oblak bez informačnej hodnoty.

4.2.5 Detekcia konceptového driftu

Model integruje Page-Hinkley drift detektor, ktorý monitoruje chybovosť ensemble. Princíp testu je detailne opísaný v sekcii 3.5; v IDDCW sa ako vstupný signál používa chybovosť ensemble predikcie: hodnota 1.0 pri nesprávnej predikcii a 0.0 pri správnej.

Voľba Page-Hinkley testu namiesto bežnejšieho ADWIN detektora bola motivovaná niekoľkými praktickými dôvodmi. Po prvé, ADWIN sa v existujúcich streaming modeloch typicky nasadzuje na úrovni jednotlivých base learnerov — napríklad v ARF má každý strom vlastný ADWIN detektor. V IDDCW máme heterogénny ensemble s per-class váhami a soft votingom, kde nás zaujíma chybovosť celého modelu, nie jednotlivých expertov. Page-Hinkley nad celkovým error streamom je pre tento účel jednoduchší a konzistentnejší s tým, ako IDDCW spravuje

expertov cez váhový systém. Po druhé, Page-Hinkley udržiava len kumulatívny súčet a priebežný priemer — pamäťový overhead je konštantný $O(1)$, zatiaľ čo ADWIN s exponenciálnymi histogramami vyžaduje $O(\log W)$. Pri instance-based spracovaní, kde sa detektor aktualizuje pri každej vzorke (nie raz za chunk), je tento rozdiel v réžii nezanedbateľný. Po tretie, parameter min_detection_interval sa s Page-Hinkley implementuje priamočiaro pridaním čítača od poslednej detekcie, čím sa jednoducho eliminujú kaskádové falošné detekcie na datasetoch s kontinuálnym driftom. Napokon, keďže baseline modely (ARF, OzaBaggingADWIN, LeveragingBagging) všetky interne používajú ADWIN, použitie odlišného detektora v IDDCW ukazuje, že výhody nášho modelu pochádzajú z replay a váhovacích mechanizmov, nie z konkrétnej voľby drift detektora.

Detektor udržiava kumulatívny súčet odchýlok od priebežného priemeru:

$$\bar{x}_t = \alpha \cdot \bar{x}_{t-1} + (1 - \alpha) \cdot x_t \tag{4.1}$$

$$S_t = S_{t-1} + (x_t - \bar{x}_t - \delta) \tag{4.2}$$

kde $\alpha = 0.999$ je faktor zabudania pre odhad priemeru, δ je parameter citlivosti (v implementácii $\delta = 0.005$) a x_t je chybovosť v čase t. Drift je detekovaný, keď rozdiel medzi aktuálnym kumulatívnym súčtom a jeho historickým minimom prekročí prah λ:

$$S_t - \min_{1 \leq j \leq t} S_j > \lambda$$

kde $\lambda = 100.0$ je detekčný prah. Nižšie hodnoty δ a λ zvyšujú citlivosť detektora, ale zároveň zvyšujú riziko falošne pozitívnych detekcií.

Implementácia obsahuje minimálny detekčný interval (parameter min_detection_interval = 2000 vzoriek) medzi dvoma po sebe idúcimi detekciami. Popísaný postup eliminuje kaskádové falošne pozitívne detekcie na datasetoch s kontinuálnym driftom, kde by opakované detekcie v krátkom čase viedli k neustálemu resetu modelu bez možnosti stabilizácie.

Pri detekcii driftu sa aktivujú nasledovné mechanizmy:

- History buffer sa vyčistí od väčšinových vzoriek – zachovajú sa len menšinové vzorky, čo urýchľuje adaptáciu na nový koncept bez straty cenných menšinových príkladov.
- Class buffery sa zmenšia na polovicu – zachová sa čiastočná história bez zastaraných vzoriek, pričom novšie vzorky majú prioritu.
- Aktivuje sa post-drift cooldown (parameter post_drift_cooldown = 300 vzoriek), počas ktorého sa replay intenzita dočasne zvyšuje (parameter post_drift_replay_boost = 1) a augmentácia sa znižuje na polovicu (parameter post_drift_aug_reduction = 0.5). Takto sa dosiahne, že po drifte model agresívnejšie opakuje menšinové vzorky z nového konceptu, zatiaľ čo šumová augmentácia sa tlmí, kým sa model nestabilizuje.

4.2.6 Správa expertov

Veľkosť ensemble je dynamicky riadená parametrami $k_{\min}$ (minimálna veľkosť, implicitne 5) a $k_{\max}$ (maximálna veľkosť, implicitne 20). Model vykonáva tri typy operácií so zložením ensemble:

Prvou je odstránenie slabých expertov. Po každej aktualizácii váh sú experti, ktorých celková váha $\sum_{j=1}^{c} w_{i,j}$ klesne pod prah $\theta \cdot c$ (kde c je počet tried a $\theta = 0.02$), automaticky odstránení z ensemble. Výsledkom je, že modely, ktoré konzistentne nesprávne klasifikujú, sú postupne eliminované.

Druhou je doplnenie na minimálnu veľkosť. Ak počet expertov klesne pod $k_{\min}$, pridajú sa noví náhodne inicializovaní experti. Pri vytváraní nového experta sa náhodne vyberie typ z definovanej sady základných klasifikátorov a hyperparametre sa randomizujú (napr. grace period pre stromy v rozsahu 20–200, learning rate pre Perceptron v rozsahu 0.001–0.05). Nový expert dostáva warmup periódu (implicitne 2 periódy $\times$ dĺžka okna), počas ktorej sa jeho váhy nestabilizujú.

Treťou je výmena najslabšieho. Ak je ensemble na maximálnej veľkosti $k_{\max}$ a celková predikcia je nesprávna, najslabší expert (s najnižším súčtom váh) sa nahradí novým. Toto zabezpečuje obnovu ensemble bez prekročenia pamäťového limitu.

4.3 Formálny zápis aktualizácie váh

Pre formálny popis uvádzame matematickú notáciu kľúčových mechanizmov modelu.

Nech $W \in R^{m \times c}$ je matica váh, kde $w_{i,j}$ je váha experta e_i pre triedu j. Nech $\text{IR}(t) = \frac{\max_j n_j(t)}{\min_{j:n_j>0} n_j(t)}$ je aktuálny pomer nevyváženosti v čase t, kde $n_j(t)$ je počet vzoriek triedy j v posuvnom okne.

Aktualizácia váh pri správnej predikcii ($\hat{y}_i = y$, kde $\hat{y}_i$ je predikcia experta e_i):

$$w_{i,\hat{y}_i} \leftarrow w_{i,\hat{y}_i} \cdot (1 + \beta \cdot r(y, \text{IR})) \tag{4.3}$$

kde funkcia odmeny $r(y, \text{IR})$ je definovaná adaptívne:

$$r(y, \text{IR}) = \begin{cases} r_{\text{min}}(\text{IR}) & \text{ak } y \neq y_{\text{maj}} \quad \text{(menšinová trieda)} \\ r_{\text{maj}}(\text{IR}) & \text{ak } y = y_{\text{maj}} \quad \text{(väčšinová trieda)} \end{cases} \tag{4.4}$$

s lineárnou interpoláciou:

$$r_{\text{min}}(\text{IR}) = \begin{cases} 0.12 & \text{ak IR} \leq 3 \\ 0.12 + \frac{\text{IR}-3}{5} \cdot 0.08 & \text{ak } 3 < \text{IR} < 8 \\ 0.20 & \text{ak IR} \geq 8 \end{cases} \tag{4.5}$$

Aktualizácia váh pri nesprávnej predikcii ($\hat{y}_i \neq y$):

$$w_{i,\hat{y}_i} \leftarrow w_{i,\hat{y}_i} \cdot (1 - \beta \cdot p) \tag{4.6}$$

kde $p = 0.04$ pre menšinovú vzorku a $p = 0.03$ pre väčšinovú, s orezaním na interval $[10^{-4}, 10^{4}]$.

Efektívna intenzita replay:

$$k_{\text{eff}} = \min\left(k, \lfloor \log_2(\max(2, \text{IR})) \rfloor\right) \cdot \gamma(\text{recall}_{\text{maj}}) \tag{4.7}$$

kde γ je regulačný faktor závislý na majority recall: $\gamma = 0$ ak $\text{recall}_{\text{maj}} < \tau_{\text{shutoff}}$, $\gamma = 0.5$ ak $\text{recall}_{\text{maj}} < \tau_{\text{critical}}$, inak $\gamma = 1$.

RWA metrika: Pre C tried s per-class accuracy a_c a per-class frekvenciou f_c:

$$\text{RWA} = \sum_{c=1}^{C} \frac{(1/f_c)}{\sum_{j=1}^{C}(1/f_j)} \cdot a_c \tag{4.8}$$

Táto metrika priraďuje vyššiu váhu správnej klasifikácii zriedkavejších tried, čím penalizuje modely ignorujúce menšinové triedy.

4.4 Pseudokód algoritmu

Algoritmus prezentuje pseudokód modelu IDDCW pre spracovanie jednej inštancie dátového prúdu.

Vstup: Inštancia (x_s, y_s), množina expertov $E = \{e_1, \ldots, e_m\}$, matica váh W
Výstup: Predikcia $\hat{y}_s$, aktualizované E, W

Pridaj (x_s, y_s) do history bufferu a class bufferov
Aktualizuj Welfordov running priemer a rozptyl
Urči majority triedu y_{maj} a minority triedy z posuvného okna
pre každého experta $e_i \in E$ **rob**
  $\mathbf{p}_i \leftarrow$ predict_proba(e_i, x_s)
koniec
$\hat{y}_s \leftarrow \arg\max_c \sum_{i=1}^{m} p_{i,c} \cdot w_{i,c}$ *// soft voting*
Aktualizuj posuvné okno (true, pred) pre majority recall
Aktualizuj drift detektor s $1[\hat{y}_s \neq y_s]$
ak drift detekovaný **potom**
  Vyčisti history buffer, zmenši class buffery
  Nastav post-drift cooldown
koniec
pre každého experta $e_i \in E$ **rob**
  ak $\hat{y}_i = y_s$ **potom** $w_{i,\hat{y}_i} \leftarrow w_{i,\hat{y}_i} \cdot (1 + \beta \cdot r(y_s, \text{IR}))$
  inak $w_{i,\hat{y}_i} \leftarrow w_{i,\hat{y}_i} \cdot (1 - \beta \cdot p)$
koniec
pre každého experta $e_i \in E$ **rob**
  e_i.partial_fit(x_s, y_s)
koniec
$k_{\text{eff}} \leftarrow$ effective_replay_k(IR, recall$_{\text{maj}}$)
ak $k_{\text{eff}} > 0$ **a** existujú validné minority buffery **potom**
  Vyber k_{eff} náhodných vzoriek z minority bufferov
  Voliteľne augmentuj Gaussovským šumom
  Natrénuj všetkých expertov na replay batch
koniec
Odstráň expertov s váhou pod prahom θ
Doplň ensemble na minimum k expertov
vráť $\hat{y}_s$

Na účely porovnania sú použité nasledovné baseline modely implementované v scikit-multiflow:

- ARF (Adaptive Random Forest) [17]: 10 stromov s ADWIN detekciou driftu a background tree mechanizmom.
- OzaBaggingADWIN [21, 15]: Online bagging s 10 Hoeffding Trees a ADWIN drift detekciou.
- LeveragingBagging [22]: Rozšírený online bagging s Poisson(6) a ADWIN.
- OnlineBoosting [21]: Online AdaBoost s 10 Hoeffding Trees.
- HoeffdingTree [7]: Jednotlivý inkrementálny strom ako najjednoduchší baseline.

4.5 Evaluačná metodika

Používame prequential evaluáciu s blokovými metrikami – výkon modelu sa vyhodnocuje v blokoch po 500 vzorkách, čo umožňuje sledovať priebeh metrík v čase a vizuálne identifikovať momenty degradácie a zotavenia výkonu. Platí to pre všetky experimenty. Pred samotným hodnotením sa model inicializuje na prvých 2000 vzorkách prúdu (pretrain fáza), ktoré sa nezarátavajú do metrík. Počas prequential hodnotenia sa každá inštancia najprv použije na predikciu (test) a následne na aktualizáciu modelu (train).

Každý experiment sa opakuje 5-krát s rôznymi náhodnými seedmi na zabezpečenie štatistickej validity výsledkov. Výsledné metriky sú priemerom naprieč všetkými behmi.

5 Experimentálne vyhodnotenie

Táto kapitola popisuje metodológiu experimentov, použité datasety a dosiahnuté výsledky. Najprv zhrnieme technické prostredie, potom popíšeme evaluačný postup, metriky a porovnávané modely, následne prejdeme k datasetom a napokon k samotným výsledkom a ich interpretácii.

5.1 Technické prostredie

Všetky experimenty bežali v Pythone 3.7.5. Ako hlavný framework pre streamové učenie sme použili scikit-multiflow — z neho pochádzajú implementácie základných klasifikátorov (Hoeffding Tree varianty, NaiveBayes, Perceptron) aj baseline ensemble metód (ARF, OzaBagging, LeveragingBagging, OnlineBoosting). Metriky a predspracovanie zabezpečuje scikit-learn, generovanie nevyvážených datasetov imbalanced-learn a prácu s word embeddingami knižnica gensim. Samotný IDDCW model je implementovaný v súbore `configurable_ddcw.py` (názov súboru zachovávame pre konzistenciu s pôvodnou implementáciou). Experimenty bežia paralelne cez multiprocessing, pričom každá konfigurácia (dataset $\times$ model) sa opakuje 5-krát s rôznymi seedmi.

5.2 Metodológia experimentov

Hodnotenie prebieha formou prequential (test-then-train) evaluácie. Pre každú prichádzajúcu inštanciu model najprv vydá predikciu, zaznamenajú sa metriky, a až potom sa model na tejto inštancii natrénuje. Ide o štandard v oblasti streamového učenia — oproti klasickej cross-validácii zachováva časové poradie dát a vernejšie simuluje reálne nasadenie.

Pred samotným hodnotením sa model najprv inicializuje na prvých 2 000 vzorkách prúdu. Tieto vzorky sa do metrík nezapočítavajú — slúžia len na to, aby model nebol hodnotený od úplnej nuly, čo by umelo sťahovalo priemer. Hodnotu 2 000 sme zvolili na základe experimentov na kratších (50 000-vzorkových) synte-

tických datasetoch, kde sa ukázalo, že zlepšuje studený štart modelu na Agrawal a nezhoršuje výkon pri ostatných datasetoch. Pre dlhé prúdy (napr. RBF Drift Balanced s 999 999 vzorkami) je tento pretrain zanedbateľný podiel prúdu a na metriky nemá merateľný vplyv.

Pre objektívne porovnanie modelov na nevyvážených dátových prúdoch používame sadu metrík pokrývajúcich rôzne aspekty výkonnosti [3]:

- RWA (Rarity-Weighted Accuracy): Metrika priraďujúca vyššiu váhu správnej klasifikácii zriedkavejších tried. Pre každú triedu sa vypočíta per-class accuracy a výsledné RWA je vážený priemer, kde váhy sú inverzne proporcionálne frekvencii triedy.
- G-Mean: Geometrický priemer pokrytí všetkých tried.
- Macro F1 / Weighted F1: Neváhovaný a vážený priemer F1 naprieč triedami.
- Mean / Worst Minority Recall: Priemerne a najhoršie pokrytie menšinových tried.
- Majority Recall: Pokrytie väčšinovej triedy – dôležitá kontrolná metrika zabraňujúca kolapsu.
- Kappa, AUC: Doplnkové metriky pre celkový výkon.
- Drift Detections: Počet detekovaných driftov pre modely s drift detektorom.

Porovnávané modely

Výkonnosť navrhnutého IDDCW modelu porovnávame s piatimi baseline metódami dostupnými v knižnici scikit-multiflow, ktoré predstavujú zaužívané prístupy k streamovej klasifikácii:

- Adaptive Random Forest (ARF) [17] — state-of-the-art streamový ensemble, kde každý strom má vlastný ADWIN drift detektor a pri detekcii driftu sa nahrádza. Predstavuje najsilnejší baseline.
- OzaBagging + ADWIN [21] — online verzia baggingu, v ktorej sa každá vzorka trénuje k-krát podľa Poissonovho rozdelenia. Variant s ADWIN detektorom umožňuje porovnanie s ARF pri odlišnej voľbe základného learneru.

- Leveraging Bagging [22] — rozšírenie OzaBagging so zvýšenou diverzitou medzi členmi ensemblu prostredníctvom vyšších váh vzoriek.
- Online Boosting [21] — streamová verzia AdaBoost z rovnakej práce ako OzaBagging, kde sa vzorkám priraďujú váhy podľa chýb predošlých klasifikátorov.
- Hoeffding Tree [7] — jednoduchý single-model baseline, ktorý slúži ako referenčný bod pre ensemble metódy.

Všetky baseline ensemble metódy používame s 10 base learnermi a Hoeffding Tree ako základným klasifikátorom, aby bolo porovnanie férové. IDDCW sme konfigurovali s piatimi heterogénnymi base learnermi (Naïve Bayes, Hoeffding Tree, Hoeffding Adaptive Tree, Extremely Fast Decision Tree, Perceptron), pričom pre datasety s viac než 50 príznakmi Naïve Bayes z poolu vynechávame kvôli numerickým problémom s overflow. Podrobnú konfiguráciu hyperparametrov IDDCW uvádzame v kapitole 4.

Ako už bolo spomenuté, metriky zaznamenávame v blokoch po 500 vzorkách, čo nám dáva priebeh výkonu v čase (96 bodov pre 50 000 vzorkový dataset). Každý experiment sa opakuje 5-krát s rôznymi seedmi a výsledky uvádzame ako priemer cez tieto behy.

5.3 Datasety

Experimentovali sme so štyrmi skupinami datasetov: syntetické binárne (3 nevyvážené datasety po 50 000 vzoriek s pomerom tried 90/10 a tri vyvážené na overenie robustnosti modelu), syntetické multiclass (4 datasety, 3–4 triedy), reálne tabulárne (ELEC, KDD99, Airlines, Shuttle) a reálne textové (Jigsaw, ElectCovid, FakeNewsComb). Prehľad je v Tabuľke 5.1.

5.3.1 Generovanie syntetických datasetov

Syntetické binárne nevyvážené datasety vznikli tak, že sme z príslušných generátorov (SEA, Agrawal, Hyperplane) nazbierali dostatočný pool vzoriek a potom pomocou funkcie `make_imbalance` z knižnice imbalanced-learn vynútili požadovaný pomer tried 90/10. Výsledné vzorky sme náhodne premiešali, aby sme simulovali reálne podmienky, kde poradie inštancií nezávisí od triedy.

RBF Drift dataset sme použili v pôvodnej vyváženej podobe (pomer tried $\approx$50/50), ide o prúd 1 000 000 vzoriek s 10 príznakmi, v ktorom sa kontinuálne menia centroidy a tým aj rozhodovacia hranica. Tento dataset sme do experimentov zaradili cielene ako zástupcu čistého driftu bez nevyváženosti, aby sme overili, ako sa modely správajú v scenári, kde je jediným sťažujúcim faktorom drift. Vyvážené SEA a Agrawal (získané z rovnakých generátorov bez aplikácie `make_imbalance`) plnia rovnakú kontrolnú funkciu pre datasety s abrupt a ľahkým driftom.

Multiclass datasety sme vytvorili pomocou `make_classification` zo scikit-learn s rôznymi konfiguráciami rozdeliteľnosti tried, šumu a posunom príznakového priestoru. Drift je simulovaný tým, že prvá a druhá polovica datasetu majú rôzne parametre generátora. Pre graduálny drift sme použili prekrývajúce sa okno (8 000 vzoriek), kde sa postupne prechádza z jednej konfigurácie na druhú.

5.3.2 Predspracovanie textových datasetov

Textové datasety vyžadovali prevod do numerickej formy. Rozhodli sme sa pre prístup cez pretrained GloVe embeddingy, pretože poskytujú rozumne hustú reprezentáciu bez nutnosti trénovať vlastný embedding model na malom datasete.

Pre Jigsaw (neformálne internetové komentáre, slang, vulgarizmy) sme zvolili `glove-twitter-100` — 1.2 milióna slov, 100 dimenzií, trénovaný na Twitteri. Pre ElectCovid a FakeNewsComb (formálnejšie novinárske texty) sme použili

`glove-wiki-gigaword-100` s 400 000 slovami, ktorý lepšie pokrýva politický a žurnalistický jazyk.

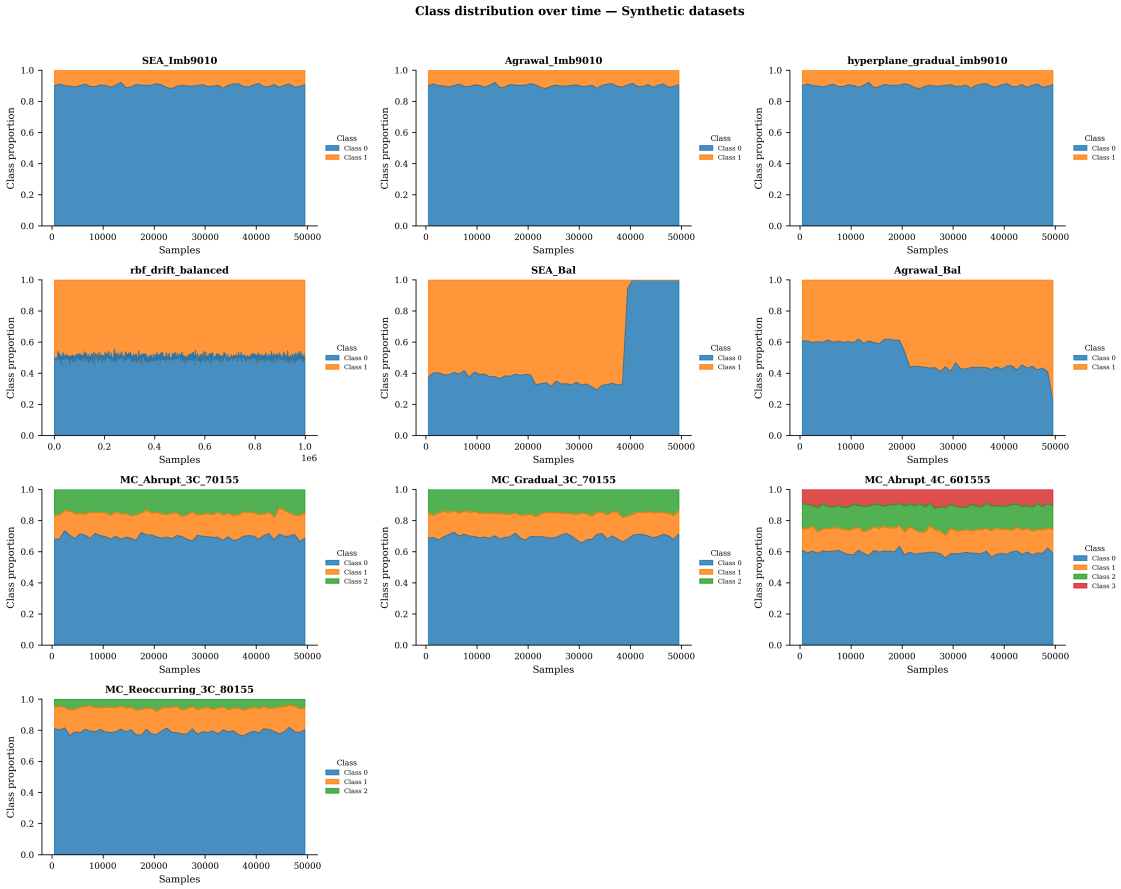
Samotný pipeline funguje v piatich krokoch. Najprv text vyčistíme — upravíme na lowercase, odstraníme URL adresy, HTML tagy a špeciálne znaky, ponecháme len slová dlhšie ako jeden znak. Potom spočítame TF-IDF váhy na celom korpuse ($\text{IDF}(w) = \log(N/(df(w)+1))$). Každý dokument potom reprezentujeme ako TF-IDF vážený priemer jeho GloVe vektorov. Na záver aplikujeme MinMax škálovanie do $[0, 1]$ a uložíme ako CSV pripravené pre streamový experiment. Poradie dokumentov zachovávame — simuluje priebeh prúdu.

Tabuľka 5.1: Prehľad datasetov použitých v experimentoch.

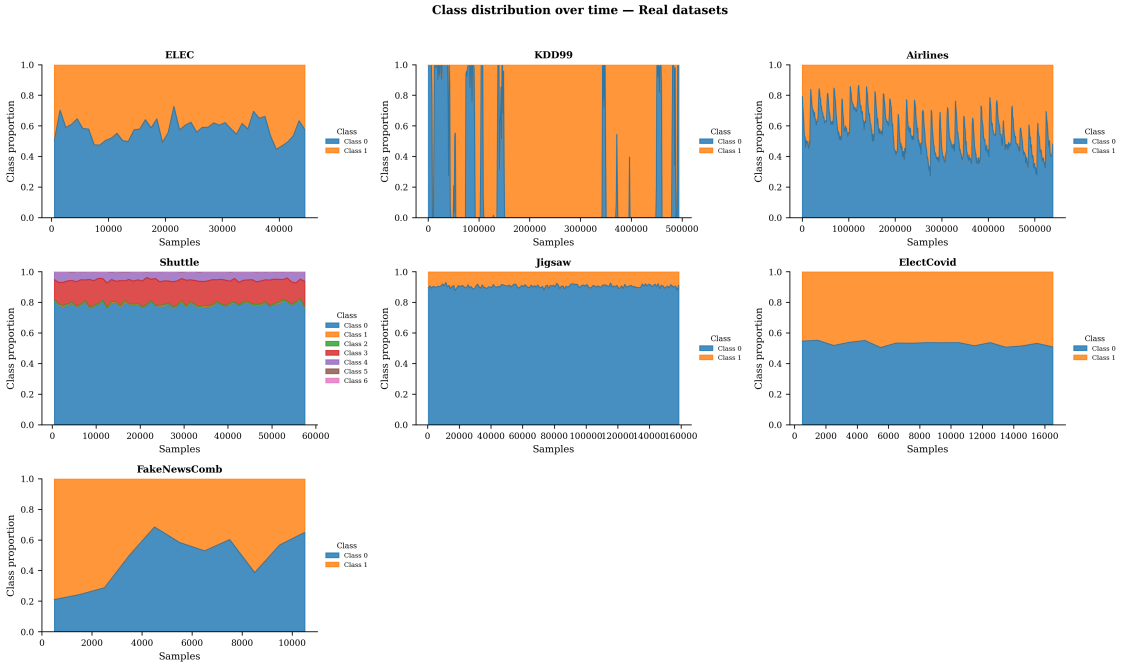
Dataset	Typ	Vzorky	Príznaky	Triedy	Drift
SEA 90/10	S-bin	50 000	3	2	žiadny
Agrawal 90/10	S-bin	50 000	9	2	abrupt
Hyperplane 90/10	S-bin	50 000	10	2	gradual
MC Abrupt 3C	S-mc	50 000	12	3	abrupt
MC Gradual 3C	S-mc	50 000	12	3	gradual
MC Abrupt 4C	S-mc	50 000	14	4	abrupt
MC Reoccurring 3C	S-mc	50 000	10	3	reoccurring
SEA Balanced	S-bin	50 000	3	2	abrupt
Agrawal Balanced	S-bin	50 000	9	2	abrupt
RBF Drift Balanced	S-bin	1 000 000	10	2	gradual
ELEC	R	45 312	6	2	neznámy
KDD99	R	494 021	41	2	neznámy
Airlines	R	539 383	7	2	neznámy
Shuttle	R	58 000	9	7	neznámy
Jigsaw	R-text	159 571	100	2	neznámy
ElectCovid	R-text	6 420	100	2	téma
FakeNewsComb	R-text	14 628	100	2	multitéma

5.3.3 Vizualizácia datasetov

Na vizualizáciu výskytu driftu sme použili dva prístupy: sledovanie feature importance v čase a distribúciu tried v čase. Obrázok 5.1 ukazuje distribúciu tried na syntetických datasetoch. Obrázok 5.2 zobrazuje to isté pre reálne datasety.



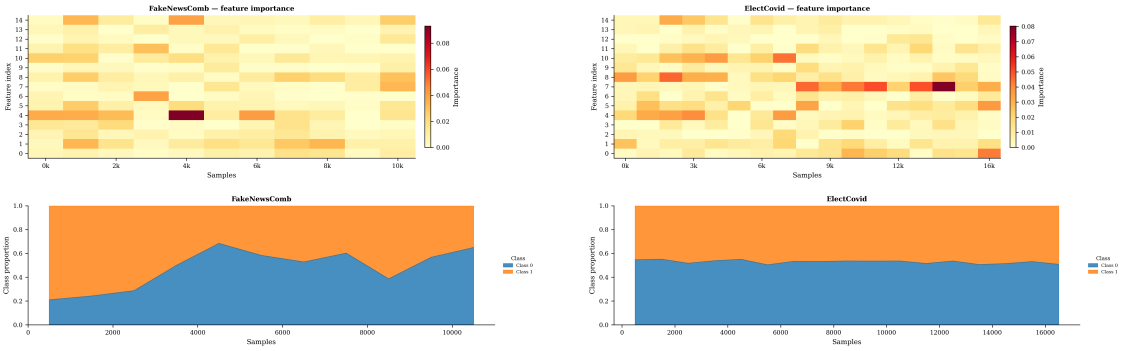
Obr. 5.1: Distribúcia tried v čase na syntetických datasetoch.



Obr. 5.2: Distribúcia tried v čase na reálnych datasetoch.

Na Obrázku 5.3 sú detailnejšie zobrazené dva vybrané reálne datasety — FakeNewsComb a ElectCovid. Na datasete FakeNewsComb je vo feature importance viditeľná ojedinelá výrazná anomália pri príznaku 4 okolo vzorky 4 000, pričom zvyšok streamu vykazuje relatívne rovnomernú dôležitosť príznakov. Distribúcia tried pritom výrazne kolíše — podiel majoritnej triedy rastie od približne 20 % až na 60 % ku koncu streamu, čo naznačuje postupnú zmenu v rozložení dát. Toto je klasický prior probability drift: mení sa, ako často sa jednotlivé triedy vyskytujú, ale samotná rozhodovacia hranica zostáva podobná.

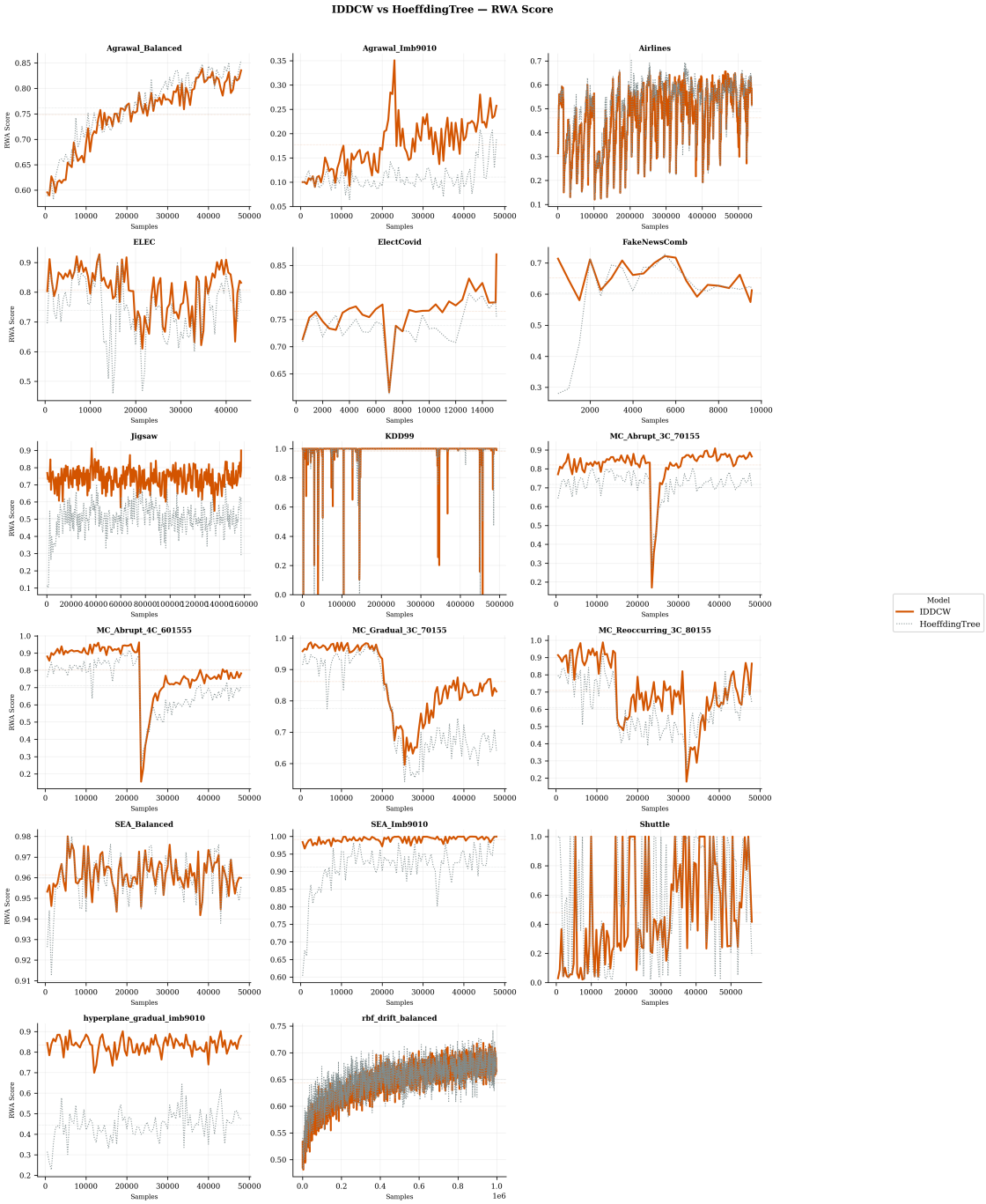
Na datasete ElectCovid sú naopak príznaky 7 a 8 konzistentne dominantné počas celého streamu, pričom okolo vzorky 13 000 dochádza k výraznému nárastu dôležitosti príznaku 7. Distribúcia tried zostáva počas celého streamu stabilná na pomere približne 55/45, čo naznačuje, že drift sa prejavuje najmä na úrovni príznakov, nie rozloženia tried. Toto naznačuje concept drift — mení sa, ktoré príznaky sú prediktívne, aj keď pomery tried ostávajú rovnaké. Pri ElectCovid to dáva zmysel — iné faktory začnú byť relevantné pre predikciu počas covidového obdobia.



Obr. 5.3: Feature importance a distribúcia tried. Vľavo: FakeNewsComb. Vpravo: ElectCovid.

5.4 IDDCW oproti jednoduchému baseline

Na Obrázku 5.4 je pre prehľadnosť porovnanie len IDDCW a HoeffdingTree naprieč všetkými datasetmi. Vidno, kde IDDCW výrazne vylepšuje výkon oproti základnému stromu a kde je rozdiel menší.



Obr. 5.4: IDDCW vs HoeffdingTree na všetkých dátových prúdoch (RWA v čase).

5.5 Výsledky na syntetických datasetoch

Tabuľka 5.2 zhŕňa výsledky na všetkých syntetických datasetoch pre všetkých šesť porovnávaných modelov.

Na SEA (bez driftu, 90/10) dosahuje IDDCW RWA 0.991 — tu replay funguje ideálne, lebo distribúcia sa nemení a model si stále dopĺňa menšinové vzorky. ARF a LeveragingBagging sú tesne za ním s RWA okolo 0.965, no v Macro F1 ho mierne prekonávajú. OzaBaggingADWIN tu zaostáva výraznejšie (0.828), čo naznačuje, že samotný ADWIN detektor bez mechanizmu na vyváženie tried nestačí.

Na Hyperplane má IDDCW 0.835 oproti 0.482 pre LeveragingBagging. Hyperplane má kontinuálny graduálny drift, kde sa rozhodovacia hranica pomaly posúva — a práve v takejto situácii kombinácia replay + adaptívne váhovanie funguje najlepšie. Všetky ostatné modely sa tu pohybujú medzi 0.24 a 0.48, čo ukazuje, že bez špeciálneho mechanizmu na nevyváženosť tento typ driftu výrazne znižuje schopnosť rozpoznať menšinovú triedu.

Zaujímavý je Agrawal, kde IDDCW (0.177) zaostáva za LeveragingBagging (0.282) aj OnlineBoosting (0.244). Pri prudkom abrupt drifte sa zdá, že jednoduchší prístup — rýchly reset a nový tréning — funguje lepšie než replay, ktorý krátko po drifte ešte prehráva staré vzorky. ARF (0.125) a OzaBaggingADWIN (0.103) sú na tom ešte horšie, čo naznačuje, že abrupt drift na silno nevyvážených dátach je problematický pre väčšinu metód.

Tabuľka 5.2: Výsledky na syntetických binárnych datasetoch (priemer z 5 behov).

Dataset	Model	RWA	G-Mean	Macro F1	Min. Rec.
SEA	DDCW	0.991	0.990	0.968	0.992
	ARF	0.966	0.981	0.989	0.962
	LevBagging	0.965	0.980	0.989	0.961
	OnlineBoosting	0.926	0.957	0.973	0.918
	OzaBagADWIN	0.828	0.899	0.942	0.809
	HoeffdTree	0.914	0.950	0.970	0.904
Agrawal	DDCW	0.177	0.256	0.540	0.087
	ARF	0.125	0.154	0.500	0.027
	LevBagging	0.282	0.448	0.645	0.202
	OnlineBoosting	0.244	0.399	0.604	0.161
	OzaBagADWIN	0.103	0.051	0.477	0.003
	HoeffdTree	0.110	0.106	0.484	0.011
Hyperplane	DDCW	0.835	0.831	0.703	0.835
	ARF	0.241	0.391	0.603	0.157
	LevBagging	0.482	0.646	0.745	0.426
	OnlineBoosting	0.391	0.564	0.693	0.325
	OzaBagADWIN	0.429	0.601	0.722	0.368
	HoeffdTree	0.445	0.615	0.729	0.385

Overenie na vyvážených datasetoch

Na SEA Balanced dosahuje IDDCW RWA 0.961, čo je v podstate identická hodnota ako LeveragingBagging a ARF. Všetky modely sa tu pohybujú v úzkom pásme 0.957–0.961, čo potvrdzuje, že pri vyváženom probléme bez extrémnej imbalance je IDDCW konkurencieschopný a jeho adaptívne mechanizmy sa správajú neutrálne — minority replay sa pri pomere tried blízkom 1:1 efektívne neaktivuje, model sa správa ako štandardný streamový ensemble.

Na Agrawal Balanced je IDDCW tesne za HoeffdingTree a viditeľne za LeveragingBagging. Zaostávanie nie je spôsobené replay mechanizmom, ale tým istým dôvodom ako pri Agrawal Imbalanced — pri abrupt drifte vo funkcii Agrawal generátora IDDCW potrebuje dlhšie na zotavenie ako jednoduchšie modely, ktoré post-drift hneď začnú učiť nový koncept bez záťaže histórie. Page-Hinkley detektor drift síce zachytil (priemerne 2 detekcie za beh), no post-drift cooldown a

dočistenie historického bufferu stoja IDDCW pár tisíc vzoriek výkonnosti. Tento výsledok je konzistentný s pozorovaním na Agrawal Imbalanced a naznačuje, že Agrawal-typ abrupt driftu je systematickou slabinou kombinácie replay + Page-Hinkley, nie anomáliou nevyváženosti.

Na RBF Drift Balanced vyniká LeveragingBagging s RWA 0.759, výrazne pred zvyškom poľa. Druhá skupina — ARF, HoeffdingTree a IDDCW — dosahuje prakticky zhodné výsledky s rozdielmi rádovo v jednotkách tisícin. Zaujímavé je, že samotný HoeffdingTree sa drží na úrovni ARF aj IDDCW, čo naznačuje, že pri tomto type kontinuálneho driftu bez nevyváženosti nepridávajú ensemble mechanizmy nič nad rámec toho, čo dokáže jediný adaptívny strom. Výnimkou je LeveragingBagging, ktorý vďaka agresívnejšiemu Poisson($\lambda = 6$) samplingu a ADWIN-riadenému resetu členov lepšie sleduje posun rozhodovacej hranice. IDDCW Page-Hinkley detektor priemerne zachytil 2 drifty na beh, no na tomto type driftu (pomalé plynulé presúvanie centroidov bez skokov) explicitná detekcia neprináša výhodu oproti pasívnej adaptácii baseline metód. OnlineBoosting a OzaBaggingADWIN zaostávajú v oboch prípadoch — OnlineBoosting kvôli citlivosti na šum v tomto type drift prúdu, OzaBaggingADWIN pravdepodobne kvôli častým false-positive resetom ADWIN detektorov v jednotlivých členoch. Tento výsledok potvrdzuje, že IDDCW je najsilnejší v scenároch, kde sa imbalance a drift kombinujú — v čisto drift-only scenári bez nevyváženosti sa mechanizmy na ochranu menšinových tried efektívne neaktivujú a model sa správa ako solídny priemer.

Tabuľka 5.3: Výsledky na syntetických vyvážených datasetoch (priemer z 5 behov).

Dataset	Model	RWA	G-Mean	Macro F1	Min. Rec.
SEA Balanced	DDCW	0.961	0.961	0.961	0.957
	ARF	0.961	0.961	0.961	0.959
	LevBagging	0.961	0.961	0.961	0.959
	OnlineBoosting	0.957	0.957	0.957	0.959
	OzaBagADWIN	0.958	0.958	0.958	0.946
	HoeffdTree	0.960	0.960	0.960	0.955
Agrawal Balanced	DDCW	0.749	0.748	0.749	0.769
	ARF	0.776	0.776	0.776	0.797
	LevBagging	0.832	0.831	0.832	0.884
	OnlineBoosting	0.713	0.713	0.713	0.732
	OzaBagADWIN	0.654	0.644	0.650	0.543
	HoeffdTree	0.762	0.762	0.762	0.738
RBF Drift Balanced	DDCW	0.645	0.645	0.645	0.645
	ARF	0.653	0.653	0.653	0.624
	LevBagging	0.759	0.758	0.759	0.734
	OnlineBoosting	0.579	0.579	0.579	0.574
	OzaBagADWIN	0.583	0.579	0.580	0.647
	HoeffdTree	0.650	0.650	0.650	0.638

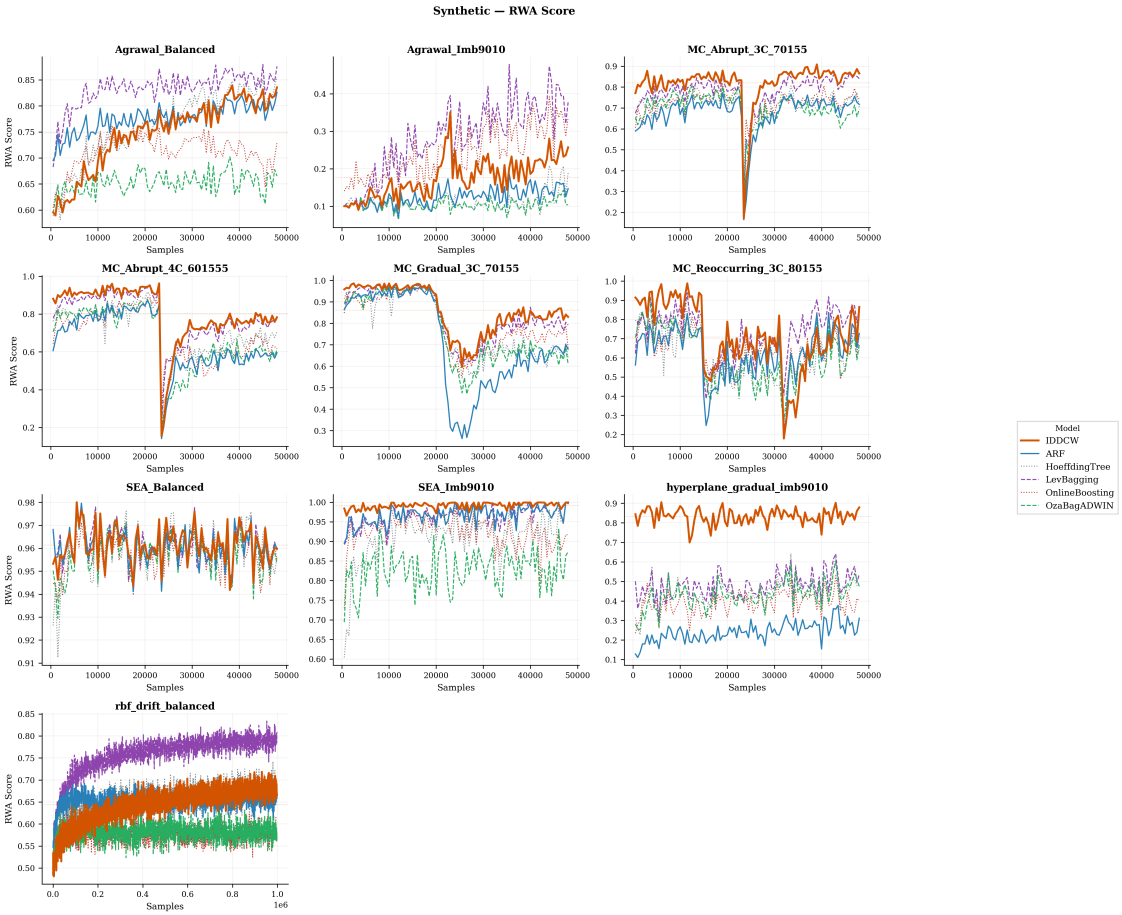
Celkovo tri vyvážené datasety potvrdzujú, že IDDCW nezhoršuje výkonnosť na vyvážených dátach oproti baseline metódam — pri ľahšom SEA je na úrovni špičky, pri RBF Drift Balanced drží krok s ARF a HoeffdingTree v strednej výkonnostnej skupine, a pri náročnejšom abrupt-drift scenári typu Agrawal má rovnakú slabinu ako má na nevyváženej verzii, nezávisle od class ratio. Jediným výraznejším víťazom naprieč vyváženými datasetmi je LeveragingBagging, čo zodpovedá jeho známym silným stránkam v čistých drift scenároch bez potreby ochrany menšinových tried.

V tabuľke 5.4 vidíme, že na multiclass datasetoch je IDDCW spoľahlivo prvý alebo druhý. Na MC Abrupt 3C dosahuje 0.822, na MC Gradual 0.862, na MC Abrupt 4C 0.804. Na všetkých troch výrazne vedie v Minority Recall. Za zmienku stojí, že HoeffdingTree — najjednoduchší model — tu nie je úplne zlý, čo naznačuje, že multiclass syntetické datasety sú pre single-model baseline prístupnejšie než binárne s extrémnym imbalance. Na MC Reoccurring je IDDCW v RWA mierne za LeveragingBagging, ale v Minority Recall vedie — adaptívne váhovanie chráni menšinové triedy aj pri opakovaných zmenách konceptu.

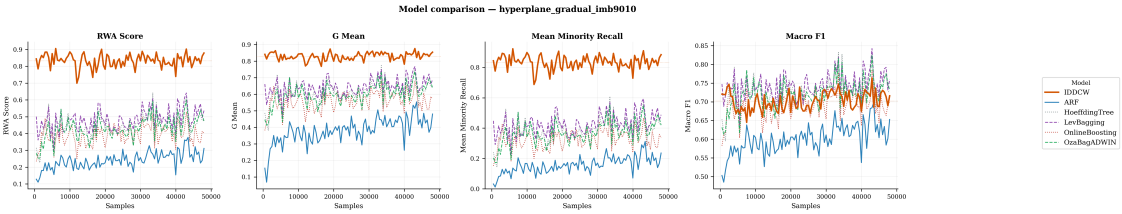
Tabuľka 5.4: Výsledky na syntetických multiclass datasetoch (priemer z 5 behov).

Dataset	Model	RWA	G-Mean	Macro F1	Min. Rec.
MC Abrupt 3C	DDCW	0.822	0.844	0.824	0.812
	ARF	0.679	0.744	0.808	0.645
	LevBagging	0.782	0.829	0.861	0.760
	OnlineBoosting	0.724	0.780	0.819	0.696
	OzaBagADWIN	0.699	0.757	0.802	0.668
	HoeffdTree	0.717	0.765	0.778	0.694
MC Gradual 3C	DDCW	0.862	0.863	0.824	0.861
	ARF	0.708	0.765	0.831	0.678
	LevBagging	0.836	0.868	0.892	0.821
	OnlineBoosting	0.799	0.839	0.870	0.780
	OzaBagADWIN	0.768	0.807	0.832	0.747
	HoeffdTree	0.776	0.812	0.825	0.758
MC Abrupt 4C	DDCW	0.804	0.815	0.794	0.803
	ARF	0.655	0.707	0.776	0.633
	LevBagging	0.786	0.816	0.848	0.771
	OnlineBoosting	0.689	0.729	0.773	0.668
	OzaBagADWIN	0.666	0.703	0.741	0.644
	HoeffdTree	0.711	0.742	0.765	0.693
MC Reoccurring	DDCW	0.703	0.789	0.765	0.767
	ARF	0.616	0.744	0.827	0.660
	LevBagging	0.728	0.820	0.868	0.754
	OnlineBoosting	0.647	0.766	0.828	0.689
	OzaBagADWIN	0.597	0.727	0.786	0.654
	HoeffdTree	0.603	0.725	0.758	0.654

Priebeh RWA v čase je na Obrázku 5.5. IDDCW (označené oranžovou čiarou) je na väčšine grafov viditeľne nad ostatnými, najmä na Hyperplane.



Obr. 5.5: Priebeh RWA v čase, syntetické datasety. IDDCW = oranžová.



Obr. 5.6: Detailné porovnanie na Hyperplane datasete.

5.6 Výsledky na reálnych datasetoch

Tabuľky 5.5 a 5.6 zhŕňa výsledky na reálnych datasetoch.

Na KDD99 sa všetky modely dostali nad 0.99 RWA — po predspracovaní je binárna klasifikácia normal/attack pomerne priamočiara. Na ELEC a Airlines vedú LeveragingBagging a ARF. Oba datasety majú miernejšiu nevyváženosť (okolo 55/45), takže replay nepriniesol výrazný efekt a len zvýšil výpočtovú náročnosť.

Najzaujímavejší výsledok je Jigsaw. IDDCW tu dosiahol RWA 0.740 a Minority Recall 0.724. Oproti tomu ARF sa dostal len na 0.305 RWA a 0.232 Minority Recall — v podstate ignoroval menšinovú triedu. LeveragingBagging bol na 0.526. Ide o dataset s asi 10% toxických komentárov a 100 GloVe príznakmi — presne typ scenára, kde replay menšinových vzoriek robí najväčší rozdiel.

Tabuľka 5.5: Výsledky na reálnych datasetoch 1/2 (priemer z 5 behov).

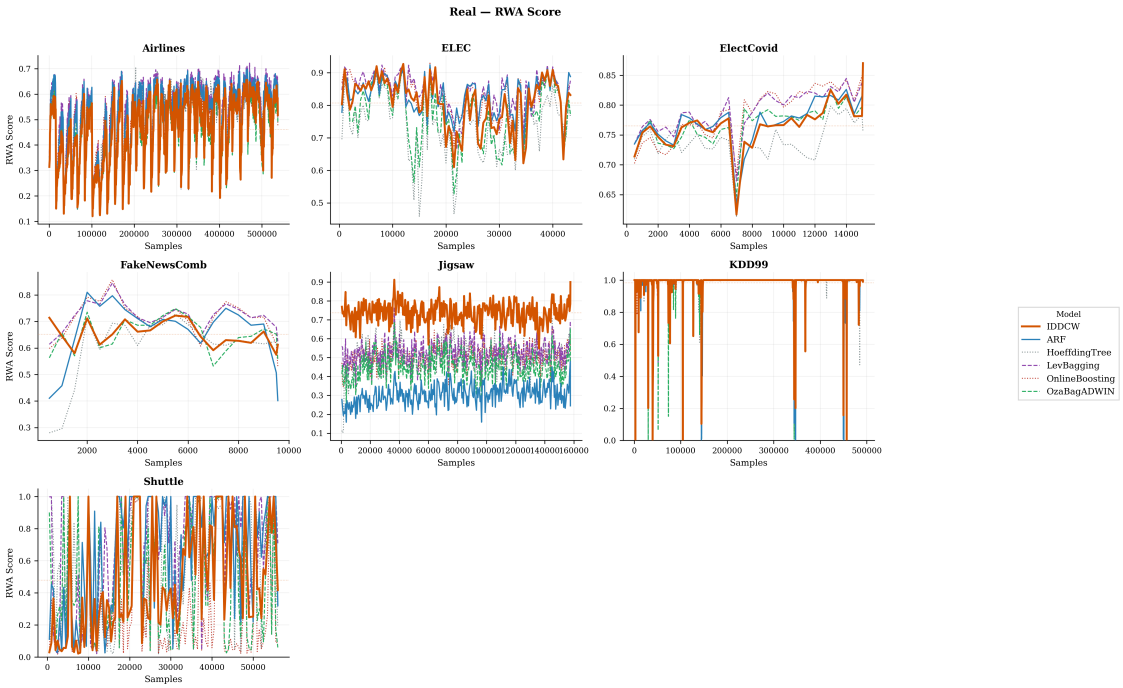
Dataset	Model	RWA	G-Mean	Macro F1	Min. Rec.
ELEC	DDCW	0.828	0.835	0.841	0.776
	ARF	0.842	0.848	0.852	0.798
	LevBagging	0.854	0.860	0.864	0.815
	OnlineBoosting	0.841	0.847	0.851	0.800
	OzaBagADWIN	0.784	0.793	0.804	0.701
	HoeffdTree	0.760	0.767	0.773	0.701
KDD99	DDCW	0.997	0.998	0.997	0.997
	ARF	0.999	0.999	0.999	1.000
	LevBagging	0.999	0.999	0.999	0.999
	OnlineBoosting	0.998	0.998	0.996	0.998
	OzaBagADWIN	0.998	0.998	0.997	0.999
	HoeffdTree	0.998	0.998	0.996	0.998
Jigsaw	DDCW	0.740	0.802	0.733	0.724
	ARF	0.305	0.481	0.666	0.232
	LevBagging	0.526	0.686	0.781	0.478
	OnlineBoosting	0.511	0.673	0.766	0.461
	OzaBagADWIN	0.456	0.629	0.751	0.399
	HoeffdTree	0.506	0.668	0.756	0.456

Tabuľka 5.6: Výsledky na reálnych datasetoch 2/2 (priemer z 5 behov).

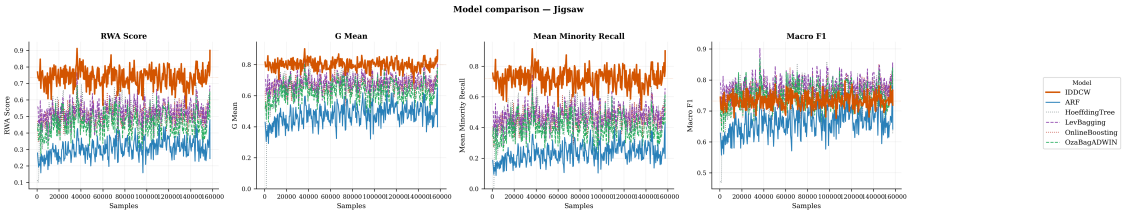
Dataset	Model	RWA	G-Mean	Macro F1	Min. Rec.
ElectCovid	DDCW	0.763	0.763	0.762	0.767
	ARF	0.771	0.772	0.773	0.747
	LevBagging	0.790	0.789	0.788	0.804
	OnlineBoosting	0.785	0.786	0.786	0.781
	OzaBagADWIN	0.767	0.767	0.766	0.770
	HoeffdTree	0.738	0.739	0.740	0.715
FakeNewsComb	DDCW	0.705	0.707	0.713	0.631
	ARF	0.754	0.755	0.754	0.743
	LevBagging	0.769	0.771	0.771	0.749
	OnlineBoosting	0.761	0.762	0.763	0.739
	OzaBagADWIN	0.693	0.695	0.700	0.632
	HoeffdTree	0.685	0.683	0.681	0.705
Airlines	DDCW	0.600	0.592	0.615	0.438
	ARF	0.636	0.636	0.651	0.510
	LevBagging	0.642	0.641	0.657	0.511
	OnlineBoosting	0.618	0.620	0.629	0.524
	OzaBagADWIN	0.607	0.602	0.622	0.456
	HoeffdTree	0.598	0.589	0.614	0.431
Shuttle	DDCW	0.163	0.226	0.599	0.487
	ARF	0.217	0.230	0.714	0.594
	LevBagging	0.311	0.352	0.722	0.640
	OnlineBoosting	0.032	0.000	0.473	0.362
	OzaBagADWIN	0.070	0.072	0.497	0.454
	HoeffdTree	0.261	0.000	0.592	0.551

Na Shuttle (7 tried, niektoré pod 0.1%) IDDCW zaostáva. Problém je v tom, že pre ultra-zriedkavé triedy sa v bufferi nenazbieral dostatočný počet vzoriek na aktiváciu replay (minimum je 10).

ElectCovid a FakeNewsComb — textové datasety so zmenou témy — sú zaujímavé iným spôsobom. IDDCW je tu konkurencieschopný, ale za LeveragingBagging zaostáva. Nevyváženosť na týchto datasetoch nie je extrémna (pomer tried okolo 2–3:1), takže sa replay tlmí podľa dizajnu. Drift tu spôsobuje zmena témy, nie zmena rozhodovacej hranice — Page-Hinkley ho detekuje, ale replay z predchádzajúcej témy logicky príliš nepomáha.



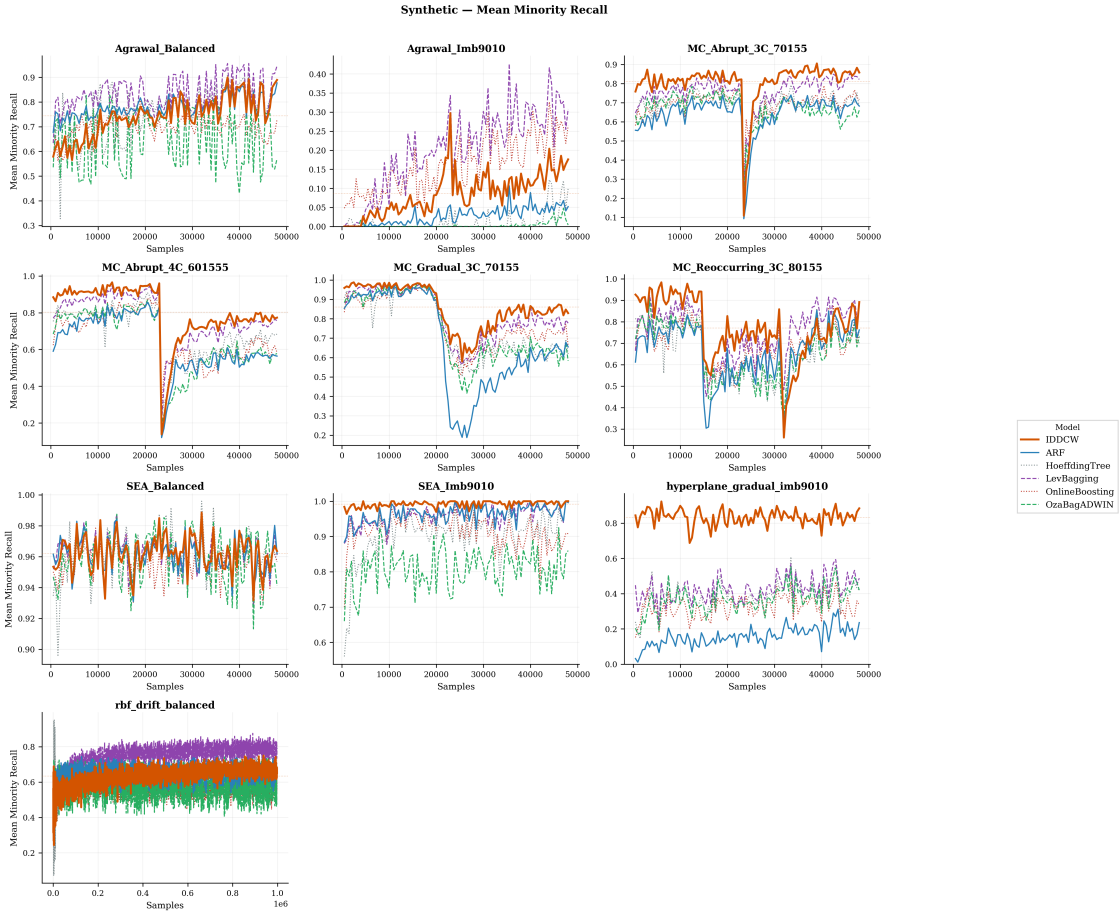
Obr. 5.7: Priebeh RWA v čase, reálne datasety.



Obr. 5.8: Detailné porovnanie na Jigsaw datasete.

5.7 Minority Recall v čase

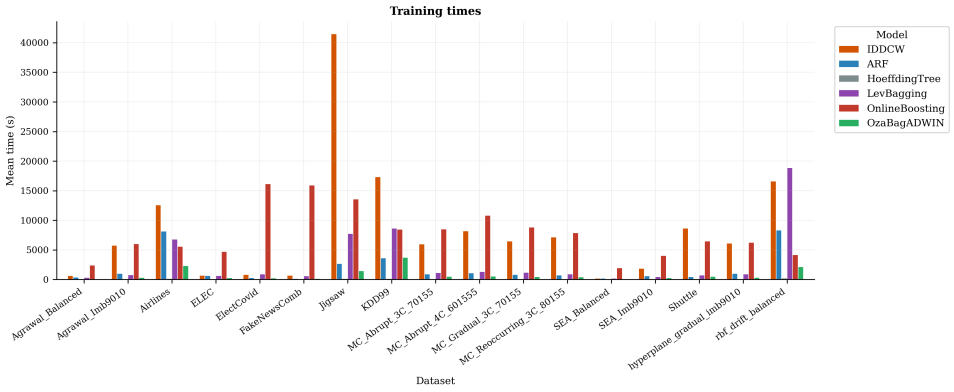
Pre nevyvážené dáta je kľúčové sledovať, či model dokáže identifikovať menšinovú triedu. Obrázok 5.9 ukazuje priebeh Mean Minority Recall na syntetických datasetoch — na Hyperplane je vidno, ako IDDCW stabilne udržiava recall nad 0.8, zatiaľ čo ostatné modely kolíšu výrazne nižšie.



Obr. 5.9: Mean Minority Recall v čase, syntetické datasety.

5.8 Tréningové časy

IDDCW je výpočtovo náročnejší než väčšina porovnávaných modelov (Obr. 5.10). Na KDD99 to bolo 17 345s oproti 3 634s pre ARF — asi 5-násobný overhead spôsobený replay mechanizmom na veľkom datasete. Na menších syntetických datasetoch je rozdiel menší (3–4×). Za zmienku stojí, že OnlineBoosting je na niektorých datasetoch ešte pomalší než IDDCW, pričom dosahuje nižší výkon na nevyvážených dátach. HoeffdingTree je najrýchlejší (desiatky sekúnd), ale jeho presnosť je zvyčajne najslabšia.



Obr. 5.10: Porovnanie tréningových časov naprieč datasetmi.

5.9 Zhrnutie experimentov

Ak by sme mali výsledky zhrnúť do jednej vety: IDDCW sa oplatí tam, kde je nevyváženosť naozaj výrazná (imbalance ratio nad 5). Na Hyperplane a Jigsaw priniesol dramatické zlepšenie oproti všetkým baseline modelom. Na multiclass scenároch bol spoľahlivo medzi najlepšími dvoma. Naopak, na mierne nevyvážených dátach (Airlines, ELEC) alebo pri ultra-zriedkavých triedach (Shuttle) sú jednoduchšie metódy ako LeveragingBagging lepšou voľbou — menej výpočtových nárokov, porovnateľný alebo lepší výkon.

Keď sa na výsledky pozrieme podrobnejšie, vynára sa niekoľko vzorcov. IDDCW jednoznačne dominuje v dvoch typoch scenárov: binárne datasety s výrazným imbalance a graduálnym alebo žiadnym driftom (SEA, Hyperplane), a silno nevyvážené textové dáta (Jigsaw). V oboch prípadoch je spoločným znakom to, že menšinová trieda je stabilne prítomná, ale výrazne menej zastúpená — presne situácia, na ktorú je replay z class bufferov navrhnutý. Na multiclass datasetoch (MC Abrupt, MC Gradual, MC Abrupt 4C) IDDCW konzistentne vedie v RWA a Minority Recall, hoci v Macro F1 ho zvyčajne predbieha LeveragingBagging. To naznačuje, že IDDCW efektívnejšie chráni menšinové triedy, no za cenu mierne nižšieho celkového F1 — čo je pri nevyvážených dátach prijateľný kompromis, pretože Macro F1 dáva rovnakú váhu všetkým triedam vrátane väčšinovej, kde IDDCW nemusí byť najsilnejší.

Tam, kde IDDCW zaostáva, sú dôvody pomerne jasné. Na Airlines a ELEC (pomer tried okolo 55/45) sa replay aktivuje len minimálne, lebo adaptívne prahy ho pri nízkom imbalance ratio tlmia — tak ako je navrhnutý. Model tu vlastne funguje ako trochu pomalší ensemble bez výrazného benefitu. Na Shuttle problém nie je v samotnom prístupe, ale v tom, že niektoré triedy majú tak málo vzoriek, že sa v bufferi nenazbieral ani minimálny počet na aktiváciu replay (10 vzoriek).

Na Agrawal s abrupt driftom je problém iný — replay krátko po drifte ešte predkladá vzorky z predchádzajúceho konceptu, čo spomalí adaptáciu. Na textových datasetoch s concept driftom (ElectCovid, FakeNewsComb) je IDDCW konkurencieschopný, no nevyniká — nevyváženosť na týchto datasetoch nie je dostatočne výrazná na to, aby sa replay naplno prejavil.

Z pohľadu výpočtovej náročnosti je IDDCW výrazne drahší než väčšina baseline modelov. Na veľkých datasetoch (KDD99, Airlines) je 3 až 8-krát pomalší než ARF. Hlavná cena pochádza z replay mechanizmu — pri každej vzorke sa vyberajú a augmentujú menšinové vzorky z bufferov a znovu sa predkladajú všetkým expertom. Na menších syntetických datasetoch je overhead menší (2 až 4-násobný). Treba ale povedať, že OnlineBoosting je na niektorých datasetoch ešte pomalší a dosahuje nižší výkon na nevyvážených dátach, takže samotná rýchlosť nie je zárukou efektivity.

Celkovo experimenty potvrdili, že integrovaný prístup IDDCW k nevyváženosti — replay s adaptívnym váhovaním a spätnou väzbou cez majority recall — prináša preukázateľné zlepšenie tam, kde ho je naozaj treba. Otázkou pre ďalší výskum zostáva, či sa dá nájsť spôsob, ako znížiť výpočtovú réžiu bez straty efektivity, a ako rozšíriť replay aj na scenáre s ultra-zriedkavými triedami.

Záver

Cieľom tejto práce bolo navrhnúť ensemble klasifikátor, ktorý dokáže pracovať s nevyváženými dátovými prúdmi a zároveň sa prispôsobovať konceptovému driftu. Výsledkom je model IDDCW (Imbalance-aware Dynamic Diversified Class-Weighted ensemble), ktorý nadväzuje na pôvodný DDCW [6] a rozširuje ho o niekoľko mechanizmov — adaptívne per-class váhovanie citlivé na aktuálny pomer tried, replay menšinových vzoriek z class bufferov s logaritmickým obmedzením podľa imbalance ratio, spätnú väzbu cez majority recall (čo bráni tomu, aby model v snahe zlepšiť minority úplne stratil schopnosť rozoznať väčšinovú triedu), soft voting namiesto hard voting, a Page-Hinkley detektor driftu s cielenou post-drift reakciou. Oproti pôvodnému DDCW sme tiež prešli z chunk-based na instance-based spracovanie.

Dôležité poznatky

Počas vývoja a ladenia modelu sme narazili na niekoľko poznatkov, ktoré stojí za to explicitne pomenovať. Prvým je, že samotný replay menšinových vzoriek nestačí — bez spätnej väzby cez majority recall model ľahko skĺzne do stavu, kde síce dobre rozoznáva menšinovú triedu, ale začne väčšinovú klasifikovať zle. Toto sme pozorovali v raných verziách modelu na datasetoch Airlines a ELEC, kde agresívny replay bez regulácie spôsobil pokles majority recall pod 40%. Zavedenie adaptívnych prahov podľa imbalance ratio tento problém vyriešilo.

Druhým poznatkom je, že prechod z chunk-based na instance-based spracovanie priniesol lepšiu reaktivitu na drift, ale za cenu výrazne vyšších výpočtových nárokov. Pri chunk-based spracovaní sa váhy a diverzita aktualizujú raz za chunk (napr. 500 vzoriek), kým pri instance-based sa aktualizujú pri každej vzorke. V praxi to znamená, že na veľkých datasetoch (KDD99 s takmer pol miliónom vzoriek) je IDDCW niekoľkonásobne pomalší než ARF.

Tretím poznatkom je dôležitosť voľby metriky. Accuracy na nevyvážených dátach pôsobí zavádzajúco — model, ktorý vždy predikuje väčšinovú triedu, do-

siahne accuracy 90% na datasete s pomerom 90/10, hoci menšinovú triedu úplne ignoruje. RWA a G-Mean sú výrazne informativnejšie, pretože penalizujú práve takéto správanie.

Z praktického pohľadu naše výsledky naznačujú, že pred výberom modelu sa oplatí najprv zhodnotiť mieru nevyváženosti v dátach. Pokiaľ je imbalance ratio nad 5, IDDCW dokáže priniesť merateľné zlepšenie. Pri miernejšom pomere tried sú jednoduchšie metódy efektívnejšie a zároveň rýchlejšie. Zdrojové kódy modelu aj experimentálneho frameworku sú dostupné pre prípadnú reprodukciu výsledkov.

Budúca práca

Vidíme niekoľko smerov pre ďalšiu prácu:

- Výpočtovú náročnosť replay mechanizmu by šlo znížiť batched spracovaním namiesto per-instance — nazbierať replay vzorky za celý blok a naraz ich predložiť expertom.
- Pre ultra-zriedkavé triedy (ako na Shuttle) by stálo za to experimentovať s generatívnymi augmentačnými technikami alebo s dynamickým znižovaním minimálneho počtu vzoriek pre replay — napríklad povoliť replay aj z 3–5 vzoriek, no s výraznejšou augmentáciou.
- Namiesto jednoduchého Gaussovského šumu by sa dala skúsiť online SMOTE varianta, ktorá by generovala syntetické menšinové vzorky interpoláciou medzi existujúcimi.
- Zaujímavým smerom by bolo preskúmať adaptívnu voľbu base learnerov — namiesto fixnej sady piatich typov modelov by model mohol dynamicky meniť zloženie podľa charakteristík aktuálnych dát.
- Na záver by bolo zaujímavé otestovať model na väčšom počte benchmarkových datasetov a doplniť štatistické testy pre rigoróznejšie porovnanie naprieč datasetmi.

Literatúra

1. GOMES, Heitor Murilo; BARDDAL, Jean Paul; ENEMBRECK, Fabrício; BIFET, Albert. A survey on ensemble learning for data stream classification. *ACM Computing Surveys*. 2017, roč. 50, č. 2, s. 1–36. Dostupné z DOI: 10.1145/3054925.
2. GAMA, Jo ao; ŽLIOBAITĖ, Indrė; BIFET, Albert; PECHENIZKIY, Mykola; BOUCHACHIA, Abdelhamid. A survey on concept drift adaptation. *ACM Computing Surveys*. 2014, roč. 46, č. 4, s. 1–37. Dostupné z DOI: 10.1145/2523813.
3. AGUIAR, Gabriel; KRAWCZYK, Bartosz; CANO, Alberto. A survey on learning from imbalanced data streams: taxonomy, challenges, empirical study, and reproducible experimental framework. *Machine Learning*. 2023, roč. 112, s. 4765–4827. Dostupné z DOI: 10.1007/s10994-023-06353-6.
4. CANO, Alberto; KRAWCZYK, Bartosz. ROSE: Robust Online Self-Adjusting Ensemble for Continual Learning on Imbalanced Drifting Data Streams. *Machine Learning*. 2022, roč. 111, č. 7, s. 2561–2599. Dostupné z DOI: 10.1007/s10994-022-06168-x.
5. WADUD, Md. Anwar Hussen; KABIR, Muhammad Mohsin; MRIDHA, M.F.; ALI, M. Ameer; HAMID, Md. Abdul; MONOWAR, Muhammad Mostafa. How can we manage Offensive Text in Social Media - A Text Classification Approach using LSTM-BOOST. *International Journal of Information Management Data Insights*. 2022, roč. 2, č. 2, s. 100095. ISSN 2667-0968. Dostupné z DOI: https://doi.org/10.1016/j.jjimei.2022.100095.
6. SARNOVSKY, Martin; KOLARIK, Michal. Classification of the drifting data streams using heterogeneous diversified dynamic class-weighted ensemble. *PeerJ Computer Science*. 2021, roč. 7, e459. Dostupné z DOI: 10.7717/peerj-cs.459.

7. DOMINGOS, Pedro; HULTEN, Geoff. Mining high-speed data streams. In: *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD*). ACM, 2000, s. 71–80. Dostupné z DOI: 10.1145/347090.347107.

8. GAMA, Jo ao; SEBASTI AO, Raquel; RODRIGUES, Pedro Pereira. On evaluating stream learning algorithms. *Machine Learning*. 2013, roč. 90, č. 3, s. 317–346. Dostupné z DOI: 10.1007/s10994-012-5320-9.

9. SARNOVSKÝ, Martin; BABIČ, František. Concept Drift Influenced by Topic Change in Data Streams from Social Media. In: *2025 IEEE 23rd World Symposium on Applied Machine Intelligence and Informatics* (*SAMI*). 2025, s. 000337–000340. Dostupné z DOI: 10.1109/SAMI63904.2025.10883281.

10. LU, Jie; LIU, Anjin; DONG, Fan; GU, Feng; GAMA, Jo ao; ZHANG, Guangquan. Learning under Concept Drift: A Review. *IEEE Transactions on Knowledge and Data Engineering*. 2019, roč. 31, č. 12, s. 2346–2363. Dostupné z DOI: 10.1109/TKDE.2018.2876857.

11. AGRAHARI, Supriya; SINGH, Anil Kumar. Concept Drift Detection in Data Stream Mining : A literature review. *Journal of King Saud University - Computer and Information Sciences*. 2022, roč. 34, č. 10, Part B, s. 9523–9540. ISSN 1319-1578. Dostupné z DOI: https://doi.org/10.1016/j.jksuci.2021.11.006.

12. GAMA, Jo ao; MEDAS, Pedro; CASTILLO, Gladys; RODRIGUES, Pedro. Learning with drift detection. *Lecture Notes in Computer Science*. 2004, roč. 3171, s. 286–295. Dostupné z DOI: 10.1007/978-3-540-28645-5_29.

13. BAENA-GARCÍA, Manuel; CAMPO-ÁVILA, José del; FIDALGO, Raúl; BIFET, Albert; GAVALDÀ, Ricard; MORALES-BUENO, Rafael. Early drift detection method. In: *Fourth International Workshop on Knowledge Discovery from Data Streams*. 2006, zv. 6, s. 77–86.

14. ARORA, Shruti; RANI, Rinkle; SAXENA, Nitin. A systematic review on detection and adaptation of concept drift in streaming data using machine learning techniques. *WIREs Data Mining and Knowledge Discovery*. 2024, roč. 14, č. 4, e1536. Dostupné z DOI: https://doi.org/10.1002/widm.1536.

15. BIFET, Albert; GAVALDÀ, Ricard. Learning from Time-Changing Data with Adaptive Windowing. In: *Proceedings of the 2007 SIAM International Conference on Data Mining* (*SDM*). SIAM, 2007, s. 443–448. Dostupné z DOI: 10.1137/1.9781611972771.42.

16. BARROS, Roberto S. M. de; CARVALHO SANTOS, Silas G. T. de. An overview and comprehensive comparison of ensembles for concept drift. *Information Fusion*. 2019, roč. 52, s. 213–244. Dostupné z DOI: 10.1016/j.inffus.2019.03.006.

17. GOMES, Heitor Murilo; BIFET, Albert; READ, Jesse; BARDDAL, Jean Paul; ENEMBRECK, Fabrício; PFAHRINGER, Bernhard; HOLMES, Geoff; ABDESSALEM, Talel. Adaptive random forests for evolving data stream classification. *Machine Learning*. 2017, roč. 106, č. 9-10, s. 1469–1495. Dostupné z DOI: 10.1007/s10994-017-5642-8.

18. ZHU, Wuxing et al. A survey on imbalanced learning: latest research, applications and future directions. *Artificial Intelligence Review*. 2024, roč. 57, s. 137. Dostupné z DOI: 10.1007/s10462-024-10759-6.

19. RUPAPARA, Vaibhav; RUSTAM, Furqan; SHAHZAD, Hina Fatima; MEHMOOD, Arif; ASHRAF, Imran; CHOI, Gyu Sang. Impact of SMOTE on Imbalanced Text Features for Toxic Comments Classification Using RVVC Model. *IEEE Access*. 2021, roč. 9, s. 78621–78634. Dostupné z DOI: 10.1109/ACCESS.2021.3083638.

20. HASSAN, Sayar Ul; AHAMED, Jameel; AHMAD, Khaleel. Analytics of machine learning-based algorithms for text classification. *Sustainable Operations and Computers*. 2022, roč. 3, s. 238–248. ISSN 2666-4127. Dostupné z DOI: https://doi.org/10.1016/j.susoc.2022.03.001.

21. OZA, Nikunj C.; RUSSELL, Stuart J. Online Bagging and Boosting. In: *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*. PMLR, 2001, zv. R3, s. 229–236. Proceedings of Machine Learning Research.

22. BIFET, Albert; HOLMES, Geoff; PFAHRINGER, Bernhard. Leveraging Bagging for Evolving Data Streams. In: *Machine Learning and Knowledge Discovery in Databases* (*ECML PKDD 2010*). Springer, 2010, zv. 6321, s. 135–150. LNAI. Dostupné z DOI: 10.1007/978-3-642-15880-3_15.

23. GOMES, Heitor Murilo; READ, Jesse; BIFET, Albert. Streaming Random Patches for Evolving Data Stream Classification. In: *IEEE International Conference on Data Mining* (*ICDM*). IEEE, 2019, s. 240–249. Dostupné z DOI: 10.1109/ICDM.2019.00034.

24. KOLTER, J. Zico; MALOOF, Marcus A. Dynamic Weighted Majority: An Ensemble Method for Drifting Concepts. *Journal of Machine Learning Research*. 2007, roč. 8, s. 2755–2790.

25. STREET, W. Nick; KIM, YongSeog. A streaming ensemble algorithm (SEA) for large-scale classification. In: *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2001, s. 377–382. Dostupné z DOI: 10.1145/502512.502568.

26. ELWELL, Ryan; POLIKAR, Robi. Incremental Learning of Concept Drift in Nonstationary Environments. *IEEE Transactions on Neural Networks*. 2011, roč. 22, č. 10, s. 1517–1531. Dostupné z DOI: 10.1109/TNN.2011.2160459.

27. JUNAID, Khan Abdul Muqeet; PAULRAJ, D.; SETHUKARASI, T. Smart adaptive ensemble model for multiclass imbalanced nonstationary data streams. *Scientific Reports*. 2025, roč. 15, s. 2264. Dostupné z DOI: 10.1038/s41598-025-05122-w.

28. JIAO, Baisong; GUO, Yuanfang; GONG, Dunwei; CHEN, Qiguang. Dynamic Ensemble Selection for Imbalanced Data Streams With Concept Drift. *IEEE Transactions on Neural Networks and Learning Systems*. 2024, roč. 35, č. 1, s. 1278–1291. Dostupné z DOI: 10.1109/TNNLS.2022.3183120.

29. FRANCIS, Navya; ALI, Anooja. Page-Hinkley Test: Approaches, Obstacles, and Resolutions. In: *Proceedings of International Conference on Recent Trends in Computing: ICRTC 2024, Volume 1*. Springer Nature, 2025, zv. 1, s. 33.

30. HERODOTOU, Herodotos; CHATZAKOU, Despoina; KOURTELLIS, Nicolas. A Streaming Machine Learning Framework for Online Aggression Detection on Twitter. In: *2020 IEEE International Conference on Big Data (Big Data)*. 2020, s. 5056–5067. Dostupné z DOI: 10.1109/BigData50022.2020.9377980.

31. KAMATERI, Eleni; SALAMPASIS, Michail. An Ensemble Framework for Text Classification. *Information*. 2025, roč. 16, č. 2. ISSN 2078-2489. Dostupné z DOI: 10.3390/info16020085.

32. PANDIARAJA, P.; MUTHUMANICKAM, K.; GEETHA, M.; MAHESH, P.C. Senthil; S K, Vedhashree; SELVARATHI, C. An Innovative Method to Detect Cyberbullying on Social Media. In: *2025 International Conference on Electronics and Renewable Systems (ICEARS)*. 2025, s. 855–858. Dostupné z DOI: 10.1109/ICEARS64219.2025.10940598.

33. GHOSH, Sreyan; KUMAR, Sonal; LEPCHA, Samden; S JAIN, Suraj. Toxic Text Classification. In: 2021, s. 251–260. ISBN 978-981-15-5308-0. Dostupné z DOI: 10.1007/978-981-15-5309-7_27.

34. VICHARE, Manaswi; THORAT, Sakshi; UBEROI, Cdt. Saiba; KHEDEKAR, Sheetal; JAIKAR, Sagar. Toxic Comment Analysis for Online Learning. In: *2021 2nd International Conference on Advances in Computing, Communication, Embedded and Secure Systems (ACCESS)*. 2021, s. 130–135. Dostupné z DOI: 10.1109/ACCESS51619.2021.9563344.

35. XIANG, Tong; MACAVANEY, Sean; YANG, Eugene; GOHARIAN, Nazli. ToxCCIn: Toxic Content Classification with Interpretability. *CoRR*. 2021, roč. abs/2103.01328. Dostupné z arXiv: 2103.01328.

36. WELFORD, B. P. Note on a method for calculating corrected sums of squares and products. *Technometrics*. 1962, roč. 4, č. 3, s. 419–420. Dostupné z DOI: 10.1080/00401706.1962.10490022.

Zoznam príloh

Príloha A CD médium – záverečná práca v elektronickej podobe

Príloha B Používateľská príručka

Príloha C Systémová príručka

Príloha B

Používateľská príručka

Táto príručka popisuje proces inštalácie prostredia, prípravu dát a spustenie experimentov pre model DDCW (Diversified Dynamic Class-Weighted ensemble).

Požiadavky na prostredie

Prostredie je navrhnuté pre jazyk Python. Vzhľadom na použitie frameworku `scikit-multiflow` sa odporúča verzia **Python 3.7.5** (novšie verzie môžu mať problémy s kompatibilitou niektorých starších knižníc).

Základné závislosti je možné nainštalovať pomocou balíčkovacieho systému pip:

```
pip install -r requirements.txt
```

Príprava datasetov

Pred spustením samotných experimentov je nutné vygenerovať syntetické dáta a predspracovať reálne dáta do jednotného formátu.

1. Syntetické datasety: Pre vygenerovanie binárnych a multiclass syntetických datasetov (SEA, Agrawal, Hyperplane, RBF a pod.) spustite skript:

   ```
   python generate_imbalanced_data.py
   ```

   Dáta sa uložia do adresárov data/synthetic_imbalanced/ a data/synthetic_multiclass/.

2. Tabulárne reálne datasety: Surové (raw) dáta (ELEC, KDD99, Airlines, Shuttle atď.) musia byť umiestnené v data/real/real_raw/. Predspracovanie (Label encoding, MinMax škálovanie) sa vykoná príkazom:

   ```
   python -m utils.data_preprocesing
   ```

3. Textové datasety (Fake News, ElectCovid a Jigsaw): Predspracovanie textu pomocou GloVe embeddingov a TF-IDF váhovania vyžaduje stiahnutie slovníkov (vykoná sa automaticky pri prvom spustení). Spustite skripty:

   ```
   python preprocess_fakenews.py
   python preprocess_jigsaw.py
   ```

Spustenie experimentov

Hlavným spúšťačom experimentov je skript `run_experiments_parallel.py`. Využíva paralelizáciu, čím výrazne urýchľuje beh.

Pred spustením je možné priamo v kóde upraviť premenné:

- `NUMBER_OF_RUNS`: Počet opakovaní experimentov s rôznymi seedmi (štandardne 5).
- `DATASET_MODE`: Určuje, aké datasety sa spustia (`"synthetic"`, `"real"`, alebo `äll"`).

Experimenty spustíte príkazom:

```
python run_experiments_parallel.py
```

V termináli sa zobrazí dashboard, ktorý indikuje priebeh jednotlivých prúdov a modelov. Výsledky sa priebežne ukladajú do adresára `results/`.

Generovanie grafov a analytiky

Po dokončení experimentov môžete výsledky vizualizovať:

- Hlavné grafy a tabuľky: Spustite `python generate_plots.py`. Skript vytvorí vo folderi `results/figures/` PNG grafy a tabuľky vo formáte LaTeX.
- Vizualizácia distribúcie tried: Skript `python visualize_datasets.py` vygeneruje grafy pre jednotlivé datasety.
- Analýza menšinových tried: Skript `python analyze_minority_performance.py` vygeneruje confusion matrice) a zhrnie F1-skóre do dedikovaného adresára.

Systémová príručka

Systémová príručka

Táto príručka poskytuje prehľad o softvérovej architektúre projektu, vnútornej štruktúre hlavných modulov a prepojeniach medzi nimi. Projekt je postavený tak, aby bol plne modulárny a pripravený na pridávanie ďalších metód a datasetov.

Dostupnosť zdrojového kódu a dát

Kompletný zdrojový kód projektu, spolu s dokumentáciou, inštrukciami a predspracovanými datasetmi, je verejne dostupný na:

`https://github.com/MatusKonopinsky/IDDCW`

`https://drive.google.com/file/d/1IEedzXqlprrxpA1ACXaPCUsNxVsRA61E/view?usp=sharing`

Architektúra riešenia

Celá platforma je štruktúrovaná do niekoľkých adresárov a koncepčných blokov:

- `model/` – Obsahuje samotnú implementáciu rozšíreného DDCW modelu a detektora driftu.
- `utils/` – Pomocné moduly pre logovanie, metriky, predspracovanie a definíciu baseline modelov.
- `data/` – Adresár so syntetickými a reálnymi prúdmi dát.
- `results/` – Automaticky generovaný adresár obsahujúci výstupy, predikcie a grafy.
- Root adresár – Spúšťacie skripty experimentov a generátory grafov.

Jadro modelu: `configurable_ddcw.py`

Zdrojový kód pre navrhovaný model sa nachádza v triede `Configurable_DDCW`, ktorá dedí od tried frameworku scikit-multiflow (`BaseSKMObject`, `ClassifierMixin`). Trieda implementuje štandardné scikit-multiflow rozhranie (`partial_fit`, `predict`, `predict_proba`, `reset`, `get_params`), čo znamená, že je zameniteľná za ktorýkoľvek iný klasifikátor v rámci knižnice.

Vnútorné komponenty

Hlavné stavebné prvky triedy sú:

- Správa expertov (`WeightedExpert`): Vnútorná trieda, kde každý expert drží referenciu na bázový klasifikátor (`estimator`), jeho *per-class* váhový vektor (`weight_class`), počítadlo životnosti (`lifetime`) a *warmup* počítadlo, ktoré obmedzuje hlasovaciu silu experta bezprostredne po jeho pridaní do ensemble.

- Trénovacia slučka (`fit_single_sample`): Kľúčová metóda volaná pri každej inštancii zo streamu. Postupne vykoná: soft-voting predikciu, aktualizáciu Page-Hinkley detektora nad prúdom chybovosti, odmenu/trest per-class váh s asymetriou adaptívnou k *imbalance ratio*, inkrementálne učenie všetkých expertov a voliteľný *replay* menšinových vzoriek z class bufferov (prípadne s Gaussovskou augmentáciou).

- Class buffere: Kruhový FIFO buffer pre každú neväčšinovú triedu, v ktorom sa držia posledné videné vzorky. Slúžia ako zdroj pre replay.

- Page-Hinkley detektor (`SimplePageHinkley`): Samostatná trieda implementujúca Page-Hinkley test nad prúdom chybovosti celého ensemble modelu. Reaguje na zmeny s konfigurovateľnou citlivosťou a minimálnym detekčným intervalom, ktorý zabraňuje zbytočnému spúšťaniu detekcie okamžite po predošlej.

Konfigurovateľné parametre

Všetky parametre sa nastavujú v konštruktore triedy `Configurable_DDCW`. Pre lepšiu orientáciu sú rozdelené do logických skupín. Pri každom parametri je uvedená východisková hodnota (default) a hodnota reálne použitá v experimentoch tejto práce, ak sa líši.

Skupina expertov a ensemble:

- `base_estimators` (zoznam, default: heterogénna pätica NaiveBayes, HoeffdingTreeClassifier, HoeffdingAdaptiveTreeClassifier, ExtremelyFastDecisionTreeClassifier, PerceptronMask) – zoznam bázových klasifikátorov, z ktorých sa ensemble skladá. Pre datasety s viac než 50 príznakmi sa NaiveBayes automaticky vynecháva kvôli numerickému overflow.
- `min_estimators` (int, default `5`) – minimálny počet expertov v ensemble. Pod túto hranicu sa experti neodstraňujú.
- `max_estimators` (int, default `20`) – maximálny počet expertov. Pri tomto hornom strope sa najslabší expert nahradí novým.
- `warmup_windows` (int, default `2`) – počet periód `period` po pridaní experta, počas ktorých má znížený hlasovací vplyv. Chráni pred tým, aby neskúsený nový člen kazil predikcie.

Váhovanie a aktualizácia:

- `period` (int, default `600`) – dĺžka periódy v počtoch vzoriek, po ktorej sa vyhodnocuje výkon expertov, prípadne sa dopĺňajú/odstraňujú členovia.
- `alpha` (float, default `0.002`) – koeficient, ktorý s rastúcou životnosťou experta postupne znižuje jeho hlasovaciu silu.
- `beta` (float, default `1.5`) – násobiteľ odmeny/trestu pri aktualizácii per-class váh. Vyššia hodnota = prudšia reakcia na správnu/nesprávnu predikciu.
- `theta` (float, default `0.02`) – minimálna prahová hodnota váhy experta pre konkrétnu triedu. Ak klesne pod `theta`, expert je pre danú triedu prakticky neaktívny.
- `enable_diversity` (bool, default `False`) – zapína výpočet Q-statistic diverzity medzi expertmi, ktorá sa potom zohľadňuje pri úprave váh.
- `rwa_strength` (float, default `1.0`) – sila rarity-weighted zložky v odmene. Pri hodnote 0 sa správa ako bežné class-weighting, pri 1 sa plne zohľadňuje frekvencia triedy v prúde.
- `use_lifetime_trend` (bool, default `True`) – povoľuje zohľadnenie životnosti experta pri úprave váh (plynulé vytlačenie zastaralých členov).

Buffere pre historické vzorky:

- `history_buffer_size` (int, default `600`) – veľkosť globálneho buffera posledných videných vzoriek. Používa sa na výpočet priebežného imbalance ratio a majority recall.

- `class_buffer_size` (int, default `300`) – veľkosť per-class buffera pre každú menšinovú triedu. Väčšie buffere umožňujú bohatší replay pool, ale zvyšujú pamäťovú náročnosť úmerne počtu tried.

Replay a augmentácia menšinových vzoriek:

- `replay_mode` (str, default replay, v experimentoch augment) – režim replay mechanizmu off ho úplne vypne, replay prehráva menšinové vzorky bez zmeny, augment ich prehráva s Gaussovskou augmentáciou podľa ďalšieho nastavenia.

- `replay_k` (int, default `3`, v experimentoch `5`) – koľko menšinových vzoriek sa prehrá na každú incoming inštanciu. Reálny počet je navyše logaritmicky limitovaný podľa aktuálneho imbalance ratio, aby sa nezahltil stream.

- `augmentation_mode` (str, default `"none"`, v experimentoch `"noise"`) – typ augmentácie. `"none"` prehráva originály, `"noise"` pridáva Gaussovský šum s online odhadom štandardných odchýlok (Welfordov algoritmus).

- `augmentation_strength` (float, default `0.02`, v experimentoch `0.01`) – relatívna sila šumu voči online odhadnutej smerodajnej odchýlke príznaku. Hodnota 0.01 znamená, že šum má $1\,\%$ variability reálnych dát – dostatočne na diverzifikáciu, nie však na deformáciu distribúcie.

- `imbalance_aware_augmentation` (bool, default `True`) – ak je zapnuté, sila augmentácie sa škáluje podľa imbalance ratio (čím väčšia nerovnováha, tým agresívnejšia augmentácia).

Regulácia replay cez majority recall:

- `majority_recall_critical` (float, default `0.55`) – ak priebežný majority recall klesne pod túto hranicu, replay sa začne tlmiť (nasleduje lineárna redukcia `replay_k`).

- `majority_recall_shutoff` (float, default `0.40`) – ak majority recall padne pod túto hranicu, replay sa v danej perióde úplne vypne, aby sa model stabilizoval na väčšinovej triede.

- `min_replay_support` (int, default `10`) – minimálny počet vzoriek, ktoré musia byť v buffere danej triedy, aby sa replay aktivoval. Tento práh sa prejavil ako zásadný napr. na datasete Shuttle, kde niektoré triedy nedosiahli ani toto minimum.

Detekcia driftu (Page-Hinkley):

- `enable_drift_detector` (bool, default `False`, v experimentoch `True`) – celkový prepínač Page-Hinkley detekcie. Ak je vypnutý, model je čisto pasívny a adaptuje sa len cez per-class váhy.
- `drift_delta` (float, default `0.005`) – parameter δ Page-Hinkley testu. Predstavuje minimálnu akceptovateľnú odchýlku od priemeru; nižšie hodnoty znamenajú citlivejšiu detekciu s vyšším rizikom falošných pozitív.
- `drift_threshold` (float, default `100.0`) – prah λ, pri ktorom sa drift vyhlási. Vyššie hodnoty = konzervatívnejšia detekcia.
- `drift_min_detection_interval` (int, default `2000`) – minimálny počet vzoriek medzi dvoma po sebe idúcimi detekciami. Zabraňuje kaskáde detekcií okolo jedinej skutočnej zmeny.
- `post_drift_cooldown` (int, default `300`) – počet vzoriek po detekovanom drifte, počas ktorých sa aplikuje zmäkčený režim učenia (znížená augmentácia, prípadne boost replay).
- `post_drift_replay_boost` (int, default `1`) – o koľko sa dočasne zvýši `replay_k` počas cooldownu.
- `post_drift_aug_reduction` (float, default `0.5`) – multiplikatívny faktor pre `augmentation_strength` počas cooldownu. Hodnota 0.5 znamená polovičnú augmentáciu, čo dáva modelu šancu lepšie zachytiť nový koncept z originálnych, nedeformovaných vzoriek.
- `reset_majority_history_on_drift` (bool, default `True`) – vymaže globálny history buffer po drifte, aby sa imbalance ratio počítalo z čerstvej distribúcie.
- `keep_class_buffers_on_drift` (bool, default `True`) – ak `True`, per-class buffere zostanú zachované aj po drifte, aby replay nestrácal pool menšinových vzoriek. Pri real-concept drift (kde sa menia aj príznakové distribúcie menšinových tried) môže byť výhodné prepnúť na `False`.

Reprodukovateľnosť:

- random_state (int alebo None, default None) – seed pre internú numpy.random.RandomState. V experimentoch sa používa schéma 400 + run_id, kde run_id $\in \{1, \ldots, 5\}$, pre 5 nezávislých behov s deterministickými seedmi.

Rozšíriteľnosť

Dizajn triedy je pripravený na pridávanie ďalších mechanizmov:

- Nový typ augmentácie – stačí pridať ďalší reťazec do `augmentation_mode` a príslušnú vetvu v metóde `fit_single_sample`.
- Iný drift detektor – namiesto `SimplePageHinkley` možno použiť ľubovoľný detektor, ktorý poskytuje metódu `update(error_value) -> bool`.
- Nový replay mode – pridaním novej vetvy do replay_mode sa dá experimentovať s rôznymi stratégiami prehrávania bez zásahu do zvyšku modelu.
- Heterogénny pool základných klasifikátorov – `base_estimators` akceptuje akékoľvek scikit-multiflow klasifikátory implementujúce partial_fit, predict_proba, takže pridanie novej metódy do poolu je jednoriadková zmena.

Moduly v adresári `utils/`

- `metrics.py` a `rwa_metric.py`: Poskytujú robustné výpočty metrík pre nevyvážené dáta, vrátane multiclass RWA (Rarity-Weighted Accuracy), G-Mean, F1 a Precision/Recall.
- `drift_metrics.py`: Zastrešuje *post-hoc* analytiku po zbehnutí experimentov. Vypočítava metriky ako oneskorenie detekcie (*Detection Lag*) a dobu zotavenia (*Recovery Time*) spárovaním známych bodov driftu a reálnych detekcií.
- `model_factory.py`: Definuje zloženie porovnávacích baseline modelov (ARF, OzaBaggingADWIN, LevBagging atď.). Centrálne priraďuje statické konfigurácie a dynamické scikit-multiflow inštancie.
- `logger.py`: Zabezpečuje multiprocesové logovanie do živého terminálového panelu. Využíva IPC (*Inter-Process Communication*) mechanizmus cez frontu (`multiprocessing.Queue`). Každý worker posiela tuply do fronty, ktoré vykresľuje jeden hlavný proces.

Spúšťanie a paralelizácia (run_experiments_parallel.py)

Skript inicializuje iterátor cez všetky dostupné datasety a bázové modely. Aby nedochádzalo k zahlteniu pamäte pri Windowse a spawn metóde vytvárania procesov, workeri ako parameter dostávajú cestu k súbor namiesto naítaných obrovských NumPy polí.

Jadro paralelného spracovania sa deje pomocou multiprocessing.Pool.imap_unordered, čo zaručuje maximálne vyťaženie procesora pri rozsiahlom grid-searchi. Metriky sú ukladané asynchrónne počas behu do pamäte a po úspešnom spracovaní modelu sú ihneď exportované do formátu CSV (napr. prequential_block_metrics.csv).