Nástroj na aplikovanie modelov generovaných prostriedkami automatického strojového učenia implementovaný ako webová aplikácia

A tool for applying models generated by means of automatic machine learning implemented as a web application

Pavol Sojka[1]

Abstrakt
V dnešnej dobe existuje množstvo dátových zdrojov, ktoré môžu svojim majiteľom priniesť dodatočné informácie. Ale len málo z nich je schopných vyhodnotiť tieto zdroje kvôli nedostatku vedomostí vo vedeckých oblastiach, ako je štatistika, matematika atď. Náš príspevok má za cieľ sčasti vyriešiť tento druh problému a vytvorili sme nástroj, ktorý implementuje kroky na uľahčenie tohto procesu. V našej predchádzajúcej práci sme implementovali nástroj na čistenie dát. Druhým krokom je výber vhodného modelu podľa daných údajov v procese automatického generovania vhodného modelu a tretím krokom je aplikovanie nájdeného modelu na dáta. V prvej kapitole popisujeme problém na celkovej úrovni, v druhej kapitole predstavujeme použitú metodiku, tretia kapitola popisuje samotný projekt a získané modely, použitie vygenerovaného modelu a posledná obsahuje záver.

Kľúčové slová
Python, web aplikácia, scikit-sklearn, dataset

Abstract
Nowadays, there are many data sources that can bring additional information to their owners. But few of them are able to evaluate these resources due to lack of knowledge in scientific fields such as statistics, mathematics, etc. Our paper aims to partially solve this kind of problem, and we have created a tool that implements steps to facilitate this process. In our previous work, we implemented a data cleaning tool. The second step is to select a suitable model according to the given data in the process of automatically generating a suitable model, and the third step is to apply generated model to the data. In the first chapter we describe the problem at the overall level, in the second chapter we present the methodology used, the third chapter describes the project itself and the obtained models, the use of the generated model and the last one contains the conclusion.

Key words
Python, web application, scikit-sklearn, dataset

JEL classification
M15

1 Úvod

V posledných rokoch sme svedkami rýchleho nástupu nových technológií a modernizácie existujúcich a tým aj obrovského množstva nimi generovaných dát. Na trhu je k dispozícii množstvo riešení, niektoré sú zadarmo a niektoré sú platené. Tieto aplikácie nebudeme v tomto

[1] Ekonomická univerzita v Bratislave, Fakulta hospodárskej informatiky, Katedra aplikovanej informatiky, Dolnozemská cesta 1, 852 35 Bratislava, pavol.sojka@euba.sk.

článku detailne popisovať z dôvodu širokého rozsahu a to nie je cieľom nášho príspevku. Viac o bezplatných a platených riešeniach nájdete napríklad na Mueller et al. 2019, Baumer a kol. 2017 alebo Gearheart 2020. Títo autori pokrývajú dva najpoužívanejšie jazyky v dátovej vede (R a Python) a posledný sa týka plateného softvérového riešenia s názvom SAS. Tieto tri sú veľmi dobre známe a používané, ale existuje mnoho ďalších ako SPSS od spoločnosti IBM a mnoho ďalších. Ďalej v texte predstavíme niektoré z najpoužívanejších technológií, no naším hlavným cieľom je použiť na tvorbu našej aplikácie jazyk *Python* so špecializovanými knižnicami. Naším cieľom bolo vytvoriť a implementovať takú aplikáciu, do ktorej si používateľ so štandardnými možnosťami (má internetový prehliadač a má k dispozícii dáta) môže nahrať svoj dataset a vykonávať na ňom všetky základné operácie ako čistenie dát a základnú analýza datasetu, následne uloženie datasetu so spracovanými úpravami, generovanie modelu a testovanie vygenerovaného modelu. Pri spracovaní údajov je dôležité pracovať s konzistentnými údajmi, ale v scenári reálneho sveta to nie je možné z dôvodu možnosti duplicity, chybných údajov, abnormálnych alebo nekonzistentných údajov (Wang et al., 2019).

1.1 Prvé kroky vo vede o údajoch (Data Science)

Ako je uvedené vyššie, Wang a kol. popísali celý proces prípravy dát pre ďalšie vyhodnotenie. Bez týchto dôležitých krokov by mohol byť konečný výsledok skreslený, a preto by prijaté rozhodnutie na základe týchto dát mohlo byť nesprávne a celý proces by bol stratou času. Prvé kroky podniknuté na poskytnutom súbore údajov (datasete) by sa mali týkať deduplikácie údajov, vymazania chybných údajov, hľadania chýbajúcich údajov a ich opravy podľa existujúcich údajov napríklad ich priemernými hodnotami. Mohli by existovať aj iné metódy, ale v našom článku používame tri formy nahradenia chýbajúcich údajov, prvá je vloženie nulových hodnôt, druhá vloženie priemerných hodnôt a tretia je vymazanie celých riadkov údajov, keď iné metódy nie sú vhodné. Používateľ si môže všetky metódy sám vyskúšať, aby zhodnotil, ktorá metóda je vhodná pre jeho potreby. Po spracovaní všetkých údajov sme pripravení vybrať vhodný model aplikovaný na náš súbor údajov.

1.2 Jazyk R

R je programovací jazyk a prostredie určené na štatistickú analýzu dát a ich grafické zobrazenie. Ide o implementáciu programovacieho jazyka S pod bezplatnou licenciou. Keďže je zadarmo, R už predbehlo komerčné S z hľadiska počtu používateľov a stalo sa de facto štandardom v mnohých oblastiach štatistiky. Funkcie prostredia R je možné rozšíriť pomocou knižníc nazývaných balíky. Pre verziu 3.6.2 ich bolo k januáru 2020 v centrálnom úložisku CRAN dostupných 15 325. Príkladom často používaného balíka je ggplot2 na vizualizáciu dát. R sa používa z príkazového riadku, ale existuje niekoľko rozhraní GUI, ako napríklad RKWard, RStudio a R Commander. R je tiež prepojené alebo používané v komerčnom softvéri, napr. v prostredí SPSS môžu používatelia priamo písať a spúšťať programy v jazyku R cez otvorené dáta.

R poskytuje širokú škálu štatistických a grafických techník vrátane lineárneho a nelineárneho modelovania, klasických štatistických testov, analýzy časových radov, klastrovania a ďalších. R je ľahko rozšíriteľný o funkcie a balíky a komunita R je známa svojimi aktívnymi aktualizáciami (Kaplan et al., 2017). Mnoho štandardných funkcií R je napísaných v samotnom jazyku R, čo používateľom uľahčuje sledovanie vykonaných zmien v algoritme. Pre výpočtovo náročné úlohy môže byť kód prepojený s C, C++ a Fortran a volaný za behu. Pokročilí používatelia môžu použiť C, C++, Java, .NET alebo Python na priamu manipuláciu s R objektmi. Ďalšou silnou stránkou R je statická grafika, ktorá dokáže generovať grafy vhodné pre vedecké publikácie vrátane napríklad matematických symbolov. Dynamická a interaktívna grafika je dostupná prostredníctvom dodatočných balíkov (Lewin-Koh, 2015).

1.3 Aplikácia SAS

SAS (predtým Statistical Analysis System) je integrovaný systém softvérových produktov vyrábaných spoločnosťou SAS Institute. Slúži ako databázový systém vo firmách, ako nástroj na analýzu a obchodné využitie dát a na druhej strane sa využíva aj na štatistickú analýzu dát vo vede a technike. Ide o modulárny softvér, takže zákazník môže využívať len časti, ktoré mu vyhovujú. SAS obsahuje vlastný programovací jazyk, označovaný aj ako SAS. Softvér SAS implementuje celý životný cyklus analýzy. Zahŕňa správu údajov, ktorá zahŕňa čistenie a prípravu údajov. Druhým krokom je analytická platforma, ktorá pozostáva z rôznych softvérových nástrojov a umelej inteligencie a posledným krokom je manažment kvality dát (Blokdyk, 2019). Táto platforma je platená a jednotlivci si ju nevyberajú často, ale v komerčnej oblasti je veľmi rozšírená.

1.4 Jazyk Python

Python je dynamicky interpretovaný jazyk. Niekedy sa zaraďuje medzi takzvané skriptovacie jazyky. Jeho možnosti sú však väčšie. *Python* bol navrhnutý tak, aby umožňoval vytváranie veľkých plnohodnotných aplikácií (vrátane grafického používateľského rozhrania – pozri napr. wxPython, ktorý využíva wxWidgets, alebo PySide a PyQT pre Qt, alebo PyGTK pre GTK+).

Python je hybridný (alebo multiparadigmatický) jazyk, to znamená, že umožňuje používať pri písaní programov nielen objektovo orientovanú paradigmu, ale aj procedurálnu a v obmedzenej miere aj funkčnú, podľa toho, čo vyhovuje. alebo je pre danú úlohu najlepšie. Z tohto dôvodu má *Python* vynikajúce vyjadrovacie schopnosti. Programový kód je krátky a ľahko čitateľný v porovnaní s inými jazykmi.

Charakteristickým rysom jazyka *Python* je produktivita z hľadiska rýchlosti písania programov. Platí to pre najjednoduchšie programy aj veľmi rozsiahle aplikácie. V prípade jednoduchých programov sa táto vlastnosť prejavuje najmä stručnosťou zápisu. Pri veľkých aplikáciách je produktivita podporovaná funkciami, ktoré sa využívajú pri rozsiahlom programovaní, ako je natívna podpora menných priestorov, používanie výnimiek, štandardné nástroje na písanie testov (testovanie jednotiek – unit testy) a ďalšie. Vysoká produktivita je spojená s dostupnosťou a jednoduchosťou používania širokej škály knižničných modulov, umožňujúcich jednoduché riešenia úloh z množstva oblastí (Jaworski et al., 2021).

Vzhľadom na všetky podmienky, ktoré musia byť splnené, sme sa rozhodli používať jazyk *Python* so všetkými poskytovanými balíkmi, tiež širokú podporu komunity vývojárov a posledným faktom bolo, že tento jazyk je rozšírený aj v komerčnej sfére.

1.5 Balík Streamlit

Streamlit je bezplatný, open-source, all-python rámec, ktorý umožňuje vedcom z údajov rýchlo vytvárať interaktívne dashboardy a webové aplikácie strojového učenia bez toho, aby boli potrebné skúsenosti s vývojom webu. "Ak poznáte *Python*, potom ste všetci pripravení používať *Streamlit"* (Li, 2022). Je to bezplatný produkt a pozostáva zo všetkého, čo je potrebné na vytvorenie interaktívnych webových stránok s grafmi, mapami, posuvníkmi a ďalšími užitočnými nástrojmi. Pomocou niekoľkých riadkov kódu môžete vytvárať informačné panely a ďalší interaktívny obsah. Využíva svoj vlastný webový server, takže aplikácia je okamžite pripravená na spustenie.

1.6 Balíky Pandas a matplotlib

V našej aplikácii sme okrem iného použili balíky s názvom *Pandas* a *Matplotlib* . *Pandas* je balík *Python* s otvoreným zdrojovým kódom, ktorý poskytuje množstvo nástrojov na analýzu

údajov. Balík sa dodáva s niekoľkými dátovými štruktúrami, ktoré možno použiť na mnoho rôznych úloh pri manipulácii s dátami. Má tiež rôzne metódy, ktoré možno použiť na analýzu údajov, čo je užitočné pri práci na problémoch vo vede o údajoch a strojového učenia v *Pythone*. Tento balík je široko používaný pri analýze údajov a tiež je určený na prácu s externými údajmi, ako sú excelovský pracovný hárok, súbory JSON (JavaScript Object Notation), súbory CSV (comma separated values - hodnoty oddelené čiarkou) a iné. Balík obsahuje rozšírené možnosti pre prácu s riadkami a stĺpcami bez toho, aby sme museli používať cykly (loops), namiesto nich sa používajú funkcie, takže tento typ programovacej paradigmy môžeme nazvať funkcionálnym programovaním. V *Pythone* ide o kombináciu procedurálneho, objektovo orientovaného a funkčného prístupu, ako už bolo spomenuté.

Matplotlib je grafická knižnica dostupná pre programovací jazyk *Python* ako súčasť *NumPy*, balíka na spracovanie numerických údajov. *Matplotlib* používa objektovo orientované API (Application Programming Interface) na vkladanie grafov do *Python* aplikácií (Nelli , 2018). V našej aplikácii sa používa na zobrazenie grafu, tzv. histogramu.

2 Metodológia

Na základe prehľadu literatúry sme zvolili rámec už spomenutý v predchádzajúcom texte. Podľa Richardsa „Streamlit skracuje čas vývoja na vytváranie webových aplikácií zameraných na údaje, čo umožňuje vedcom z údajov vytvárať prototypy webových aplikácií pomocou *Pythonu* za hodiny namiesto dní.“ (Richards, 2021). Podľa Nokeri „Zabezpečte webové aplikácie a nasaďte ich na cloudové platformy“ (Nokeri , 2021) sme sa tiež rozhodli nasadiť našu aplikáciu do cloudovej infraštruktúry v cloude spoločnosti Oracle. Teraz si popíšme kroky k naplneniu nášho cieľa.

1. Rozhodli sme sa vyvinúť aplikáciu, ktorá umožní používateľom nahrávať svoje dátové súbory a vykonávať základné úlohy pri ich čistení, resp. doplnení o chýbajúce dáta. Tento prístup sa nazýva „čistenie údajov“ a podľa Naumana „Čistenie údajov je komplexný súbor úloh, ktorý berie ako vstup jeden alebo viacero súborov s údajmi a vytvára ako výstup jeden čistý súbor údajov“ (Nauman et al., 2022). Tento proces si zjavne vyžaduje veľké množstvo času, preto sme sa rozhodli tento čas skrátiť znížením niektorých úloh spojených s čistením údajov. Po vyčistení údajov sa pokúsime nájsť vhodný klasifikačný model aplikovaný na náš súbor údajov.
2. Po vykonaní prieskumu sme sa rozhodli použiť pre aplikáciu webovú technológiu kvôli jej rozšírenosti a jednoduchému použitiu s webovými prehliadačmi.
3. Preto sme sa rozhodli použiť webový framework *Streamlit* založený na technológii jazyka *Python* .
4. Vývoj zdrojového kódu na implementáciu aplikácie.
5. Testovanie a nasadenie na lokálnom serveri.
6. Finálne nasadenie aplikácie na cloudovej infraštruktúre Oracle na virtuálnom stroji Linux.

3 Príprava prostredia

3.1 Predpoklady

Na dosiahnutie konečného cieľa sme potrebovali urobiť povinné kroky. V prvom rade si musíme stiahnuť knižnicu (rámec - framework) *Streamlit* z domovskej stránky projektu. Na stránke sú inštalátory vhodné pre rôzne operačné systémy a architektúry. Po výbere vhodného sa odporúča postupovať podľa pokynov na úplné dokončenie procesu inštalácie. Na dosiahnutie

tejto úlohy by si implementátori mali byť vedomí správy operačných systémov *Windows*, *Linux* alebo *MacOS* . Po úspešnej inštalácii môžeme spustiť *Streamlit* framework. Ak všetko funguje dobre, dostaneme webový odkaz a číslo portu, na ktorom webový server počúva. Používateľ je potom presmerovaný do webového prehliadača, kde sa aplikácia po dokončení nachádza. Vyššie popísané kroky je potrebné vykonať, inak nebude implementácia možná.

3.2 Výber vhodného súboru údajov

Naša aplikácia je teraz vhodná pre prácu s rôznymi množinami údajov (datasety). Využitie tejto aplikácie sme zamerali na datasety, ktoré väčšinou obsahujú číselné hodnoty usporiadané do riadkov a stĺpcov. Preto uprednostňujeme štandardnú formu súboru údajov, ktorá je všeobecne známa ako napríklad hárok Excel alebo hárok LibreOffice, ale exportovaný do formátu CSV. Iné typy datasetov, ktoré väčšinou obsahujú textové hodnoty, nie sú vhodné. Pre kategoriálne hodnoty implementujeme rôzne kódovače (enkódery), ktoré budú využívať algoritmy, ktoré konvertujú kategoriálne hodnoty na skupiny čísel, čo im umožní správne fungovať. Použili sme súbor údajov *sonar.csv* používaný pre tento typ výpočtov strojového učenia získaných z repozitára *Github*. Dataset *sonar.csv* obsahuje údaje, ktoré popisujú, či odrazené akustické signály zo sonaru patria k odrazovému vzoru *míny* (M) alebo *horniny (R).*

3.3 Výber vhodného modelu

Náš projekt je rozdelený do niekoľkých častí, prvá je čistenie dát a jednoduchá stručná grafická analýza na základe histogramových grafov, druhá časť je výber modelu a záverečná časť zahŕňa testovanie modelu.

Výber vhodného modelu je veľmi zložitý a náročný proces, ktorý zahŕňa výber vhodného typu estimátora, vhodného algoritmu, výber hyperparametrov a ich hodnôt a tiež ich rôznych kombinácií a pod. Odvetvie strojového učenia je teda veľmi zložité na to, aby ho pochopil jeden človek bez nutnosti dlhodobého štúdia problematiky. Keď sa pozriete na algoritmy strojového učenia, neexistuje žiadne jedinečné riešenie ani jeden prístup, ktorý by vyhovoval všetkým scenárom použitia. Existuje niekoľko faktorov, ktoré môžu ovplyvniť rozhodnutie zvoliť si algoritmus strojového učenia (Harlalka, 2018). Strojové učenie je metóda analýzy údajov, ktorá automatizuje vytváranie analytických modelov. Ide o odvetvie umelej inteligencie založené na myšlienke, že systémy sa môžu učiť z údajov, identifikovať vzory a robiť rozhodnutia s minimálnym zásahom človeka (Gradillas, 2021).

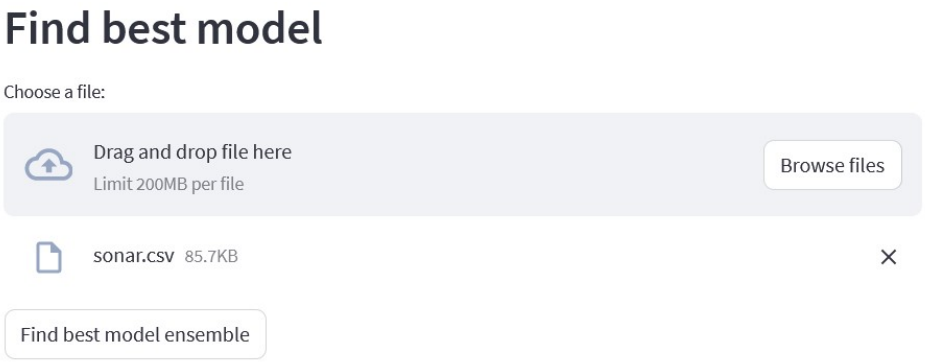
Naša aplikácia je napísaná ako webová aplikácia s veľmi jednoduchým a intuitívnym rozhraním, takže každý používateľ s minimálnymi znalosťami štatistických metód alebo algoritmov strojového učenia ju môže použiť pre svoje vlastné dáta. Rozhodli sme sa implementovať klasifikátory s predpripravenými klasifikačnými značkami v tréningovom dátovom súbore, toto sa nazýva supervízne učenie (s učiteľom), ďalšie typy učenia, ktoré sme v našej práci nepoužili, sú učenie bez učiteľa a posilňovacie učenie.

Pre komplexnosť takejto aplikácie sme implementovali iba klasifikačný algoritmus, ale implementácia iných algoritmov je plánovaná v budúcom rozšírení aplikácie. Naša aplikácia ponúka čistenie dát, vyhľadávanie modelov a aplikáciu modelov nájdených automatizovaným strojovým učením. Ako je znázornené na obrázku 1, používateľ musí poskytnúť súbor vo formáte CSV (hodnoty oddelené čiarkou) s vopred klasifikovaným súborom údajov, na ktorom bude model trénovať. Následne sa súbor údajov nahrá na server a spustí sa vyhľadávanie modelu.

V tomto bode musíme mať na pamäti, že táto časť je veľmi citlivá, pretože musíme zvoliť aspoň približne správne množstvo primeraného času na vykonanie výpočtov. Automaticky vybrané modely sme testovali do limitu piatich minút určených na testovanie jednotlivých variácií modelov, pretože sme trénovali modely na malých súboroch údajov a boli sme obmedzení hardvérovými zdrojmi poskytnutými serveru v cloude. Tento typ výpočtu je veľmi

závislý od počtu procesorov počítača a veľkosti operačnej pamäte. Naše výpočty sme testovali aj na notebooku s ôsmimi jadrami a šestnástimi gigabajtmi pamäte, ale server, ktorý sme použili, má jedno jadro a šesť GB pamäte. Preto musíme zvoliť parametre, aby sme sa vyhli neprimeranému času na získanie výsledkov. Naše riešenie je v podstate pripravené na serverové použitie vzhľadom na typ služby - webovej aplikácie. Naše odporúčanie je používať našu aplikáciu v serverovom prostredí s minimálne štyrmi jadrami a ôsmimi gigabajtmi pamäte, aby sa dosiahli primerané výsledky v primeranom časovom rozpätí pre väčšie súbory údajov.

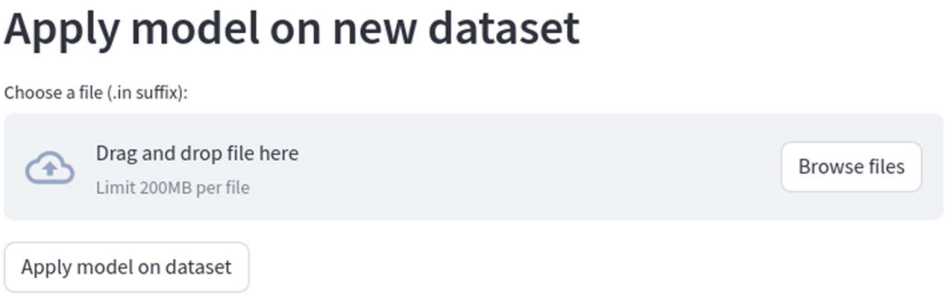
Obr. 1: Používateľské rozhranie výberu najlepšieho modelu



Zdroj: vlastné spracovanie

Po skončení hľadania najlepšieho modelu podľa zvolenej metriky, v našom prípade *accuracy* (presnosť) môžeme následne hneď v aplikácii otestovať aj presnosť vybudovaného modelu v časti určenej na nahratie testovacieho súboru s dátami (obr. 2).

Obr. 2: Používateľské rozhranie pre aplikovanie modelu



Zdroj: vlastné spracovanie

3.4 Krátky pohľad na kód

Pre väčšiu prehľadnosť sme zvolili možnosť, aby mal súbor koncovku *in,* aby aplikácia vedela jednoduchšie odlíšiť vstupy od výstupov, tento proces sa v ďalšej verzii aplikácie pravdepodobne zmení, aby nemusel používateľ premenovávať súbor - hoci stačí len pridať za názov súboru príponu *in*. Vo vstupnom súbore sa už nachádzajú dáta, ktoré nemajú klasifikované jednotlivé pozorovania (riadky) a práve na nich otestujeme náš model, a konkrétne to, či dokáže tieto dáta klasifikovať správne. My sme model otestovali aj na trénovacích dátach, ale klasifikačné značky sme predtým zmazali. Tým pádom sme overili presnosť modelu, kde bola úspešnosť klasifikácie 100%, čo sa dalo očakávať, keďže bol model na týchto dátach trénovaný pri použití klasifikátora *RandomForestClassifier*.

Obr. 3: Ukážka kódu v Pythone

```
loaded_model = pickle.load(open('file.pickle', "rb"))
list_of_files = glob.glob('*.in')
latest_file = max(list_of_files, key=os.path.getctime)
print("IN (CSV) file currently using:", latest_file, "\n")
dataframe = read_csv(latest_file, header=None)
data = dataframe.values
X = data
ynew = loaded_model.predict(X)
```

Zdroj: vlastné spracovanie

Ako vidíme na obrázku 3, do objektu (premennej) *loaded_model sa* uloží (deserializuje) obsah binárneho súboru *file.pickle*, kde máme uložený predtrénovaný model, ktorý sme predtým hľadali (obr. 1). Ďalej systém hľadá všetky súbory s príponou *in* a následne vloží obsah najnovšieho súboru *in* do premennej *data*. Do premennej *ynew* sa uložia výsledky predikcie (obr. 4), ktoré sa uložia zároveň aj do súboru a ten je ponúknutý používateľovi na stiahnutie.

Obr. 4: Získaná predikcia podľa aplikovaného modelu

```
X=[0.02   0.0371 0.0428 0.0207 0.0954 0.0986 0.1539 0.1601 0.3109 0.2111
0.1609 0.1582 0.2238 0.0645 0.066  0.2273 0.31   0.2999 0.5078 0.4797
0.5783 0.5071 0.4328 0.555  0.6711 0.6415 0.7104 0.808  0.6791 0.3857
0.1307 0.2604 0.5121 0.7547 0.8537 0.8507 0.6692 0.6097 0.4943 0.2744
0.051  0.2834 0.2825 0.4256 0.2641 0.1386 0.1051 0.1343 0.0383 0.0324
0.0232 0.0027 0.0065 0.0159 0.0072 0.0167 0.018  0.0084 0.009  0.0032], Predicted=0
X=[0.0453 0.0523 0.0843 0.0689 0.1183 0.2583 0.2156 0.3481 0.3337 0.2872
0.4918 0.6552 0.6919 0.7797 0.7464 0.9444 1.     0.8874 0.8024 0.7818
0.5212 0.4052 0.3957 0.3914 0.325  0.32   0.3271 0.2767 0.4423 0.2028
0.3788 0.2947 0.1984 0.2341 0.1306 0.4182 0.3835 0.1057 0.184  0.197
0.1674 0.0583 0.1401 0.1628 0.0621 0.0203 0.053  0.0742 0.0409 0.0061
0.0125 0.0084 0.0089 0.0048 0.0094 0.0191 0.014  0.0049 0.0052 0.0044], Predicted=0
X=[0.0262 0.0582 0.1099 0.1083 0.0974 0.228  0.2431 0.3771 0.5598 0.6194
0.6333 0.706  0.5544 0.532  0.6479 0.6931 0.6759 0.7551 0.8929 0.8619
0.7974 0.6737 0.4293 0.3648 0.5331 0.2413 0.507  0.8533 0.6036 0.8514
0.8512 0.5045 0.1862 0.2709 0.4232 0.3043 0.6116 0.6756 0.5375 0.4719
0.4647 0.2587 0.2129 0.2222 0.2111 0.0176 0.1348 0.0744 0.013  0.0106
0.0033 0.0232 0.0166 0.0095 0.018  0.0244 0.0316 0.0164 0.0095 0.0078], Predicted=0
X=[0.01   0.0171 0.0623 0.0205 0.0205 0.0368 0.1098 0.1276 0.0598 0.1264
0.0881 0.1992 0.0184 0.2261 0.1729 0.2131 0.0693 0.2281 0.406  0.3973
0.2741 0.369  0.5556 0.4846 0.314  0.5334 0.5256 0.252  0.209  0.3559
0.626  0.734  0.612  0.3497 0.3953 0.3012 0.5408 0.8814 0.9857 0.9167
0.6121 0.5006 0.321  0.3202 0.4295 0.3654 0.2655 0.1576 0.0681 0.0294
0.0241 0.0121 0.0036 0.015  0.0085 0.0073 0.005  0.0044 0.004  0.0117], Predicted=1
```

Zdroj: vlastné spracovanie

4 Záver

Cieľom nášho príspevku bolo umožniť ľuďom s menšími znalosťami strojového učenia uľahčiť proces prípravy modelu. Naším sekundárnym cieľom bolo tiež vytvoriť aplikáciu, ktorá by bola nezávislá od množiny údajov (teda dataset bez špeciálnych názvov stĺpcov a riadkov), takže každý, kto má množinu údajov uloženú vo formáte CSV, bude môcť tento dataset nahrať na náš server. Tieto ciele boli naplnené a teda máme kompletný nástroj, ktorý zahŕňa transformáciu datasetu tak, aby neobsahoval chýbajúce alebo inak poškodené dáta, zahŕňa hľadanie príslušného modelu pomocou balíka *auto-sklearn* a nakoniec umožňuje model otestovať na dátach pomocou P*ython* knižnice *scikit-learn,* kde *sa* po dokončení úlohy ponúkne súbor na stiahnutie s výsledkami klasifikácie. Ďalšími možnosťami rozvoja aplikácie je rozšíriť možnosti hľadania vhodných modelov nielen pre klasifikačné estimátory ale aj iné estimátory, čo bude samé o sebe výzvou, lebo všetky oblasti, ktorých sa týka oblasť strojového učenia, sú pomerne obsiahle a náročné na spracovanie nových informácií.

Literatúra

1. Blokdyk, G. (2019). SAS Software a Complete Guide - 2019 Edition. Emereo Pty Limited.
2. Gearheart, J. (2020). End to end data science with Sas: A hands on programming guide. Sas Institute.
3. Gradillas, R. (2021). Machine Learning Algorithms: How to Choose the Right Kind of Machine Learning Model. Independently Published.
4. Harlalka, R. (2019, April 2). Choosing the right machine learning algorithm - HackerNoon.com - medium. Medium. https://medium.com/hackernoon/choosing-the-right-machine-learning-algorithm-68126944ce1f
5. Jaworski, M., & Ziadé, T. (2021). Expert python programming: Master python by learning the best coding practices and advanced programming concepts. Packt.
6. Kaplan, D., & Horton, N. J. (2017). Modern Data Science with R. CRC Press,Taylor & Francis Group.
7. Lewin-koh, Nicholas. (2015). CRAN Task View: Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization. cran.r-project.org.
8. Li, S. (2022, January 14). Streamlit Hands-On: From zero to your first awesome web app. Medium. https://towardsdatascience.com/streamlit-hands-on-from-zero-to-your-first-awesome-web-app-2c28f9f4e214
9. Mueller, J. P., & Massaron, L. (2019). Python for data science for dummies. Wiley.
10. Nauman, F., & Herschel, M. (2022). An introduction to duplicate detection. Springer Nature.
11. Nelli, F. (2018). Python data analytics: With pandas, NumPy, and Matplotlib. Apress.
12. Nokeri, T. (2022). Web App Development and Real-Time Web Analytics with Python: Develop and Inte-grate Machine Learning Algorithms into Web Apps. Apress.
13. Richards, T. (2021). Getting Started with Streamlit for Data Science: Create and deploy Streamlit web applications from scratch in Python. Packt Publishing Ltd.
14. SPRINGER Verlag, SINGAPORE. (2018). Data science: 4th International Conference of Pioneering Computer scientists.
15. Wang, D., Guo, Y., Dong, W., Wang, Z., Liu, H., & Li, S. (2019). Deep Code-Comment understanding and assessment. IEEE Access, 7, 174200–174209. https://doi.org/10.1109/access.2019.2957424