DataVaders - Projektterv

Bajnóczi Bendegúz, Csiszár András, Mészáros Zsolt, Szakál Mátyás, Vass Máté
Gépi tanulási módszerek gyakorlat – 2022

Formai követelmények

Minden szekció maximum **1** oldal hosszú lehet. Betűméret: **12**. Ahol lehet, folyó szövegben fogalmazzunk listák és táblázatok helyett, hacsak nem tudunk azokkal sok helyet megspórolni. **Maximális pontszám** eléréséhez az összes felsorolt kérdésre válaszolni kell, valamint saját ötleteket is tartalmaznia kell a leírásnak, ezzel bizonyítva, hogy a csapat mélyen megvizsgálta a feladatot.

Dokumentáció beadása

Az elvárt formátum: pdf, ezt CooSpace-en kell beadni. Minden beadás tartalmazza a korábbi szekciókat is, például a 2. Mérföldkő minden előtte lévő Mérföldkövet és szekciót tartalmaz.

Mérföldkő neve	Pontszám	Mérföldkő határideje	Dokumentáció beadásának határideje
Feladatleírás	+5	-	2022-02-28
Adatfeldolgozás (MK - I)	20	2022-03-07	2022-02-28
Modellezés (MK - II)	20	2022-03-28	2022-03-21
További fejlesztések (MK - III)	20	2022-05-02	2022-04-25
Projekt prezentálása	5	2022-05-09	-
Egyéni feladatok	35	2022-05-02	2022-04-25

Feladat leírása

- Mi a feladat?
    - A projekt során egy felügyelt gépi tanulási problémát oldunk meg egy általunk tetszőlegesen választott adathalmazon. A munkához egy banki adatbázist választottunk, amiben a bank az ügyfeleiről tárol el különféle adatokat. Ezek alapján azt szeretnénk megjósolni, hogy egy új ügyfél a jövőben várhatóan fog-e betéti számlát nyitni.
- Feladat típusa
    - Osztályozás.
- Motiváció
    - A meglévő adatok alapján megpróbálhatjuk megjósolni, hogy egy új ügyfél tervez-e megtakarítást elhelyezni a bankban.
    - Miért hasznos ez a feladat?
        i. Így céltudatosan tud a bank hirdetéseket és telefonos ajánlatokat tenni bizonyos ügyfeleknek.
- Adathalmaz leírása
    - Adat formátuma
        i. .csv
    - Adatsorok jellemzői
        i. Rendezett adatsoraink vannak
        ii. Jellemzők:
            1. *Kliens adatai*: életkor, munka fajtája, kapcsolati státusz, tanulmányok, van-e hiteltartozása, átlagos éves egyenleg, van-e lakáshitele, van-e személyi hitele
            2. *Legutóbbi kapcsolatfelvétel jellemzői:* kapcsolatfelvétel típusa, kapcsolatfelvétel hónapja, kapcsolatfelvétel napja, kapcsolatfelvétel időtartama
            3. *Egyéb jellemzők:* kapcsolatfelvételek száma, utolsó kapcsolatfelvétel óta eltelt napok száma, eddigi összes kapcsolatfelvétel száma, előző kapcsolatfelvétel kimenetele
        iii. Címke:
            1. Az ügyfélnek van-e a banknál betéti számlája?
    - Prediktálási cél
        i. F-e az ügyfél a jövőben betéti számlát nyitni?
    - Adathalmaz mérete
        i. 45211 felcímkézett rekord
        ii. 16 jellemző vektor
        iii. 1 bináris (igen/nem) címke
- Használt környezet és eszközök:
    - Programozási nyelv
        i. Python (Google Colaboratory Notebook)
    - Gépi tanulást megvalósító könyvtárak
        i. NumPy, pandas, scikit-learn
    - Verziókövetés
        i. Github repository: https://github.com/material1999/gepitan_project

Adatfeldolgozás (MK - I)

- Adatfelosztási módszer
    - Mivel nagyon nagy mennyiségű felcímkézett példa áll rendelkezésünkre, így terveink szerint 70%-15%-15% felosztást fogunk alkalmazni (70% train, 15% dev, 15% test)
    - Így nagyjából 30 ezer adaton tudunk majd betanítani, továbbá 7-7 ezer adaton fejleszteni, valamint tesztelni.
- Tanuláshoz felhasznált jellemzők
    - A tanuláshoz terveink szerint az adathalmazunk jellemzésénél felsoroltak közül az *“1. Kliensek jellemzői”* attribútumokat szeretnénk felhasználni. Ezeket gondoltuk a legfontosabb információknak a feladatunk szempontjából, a többi jellemző kihagyásával elkerülhetjük az esetleges túltanulást.
    - Összehasonlításképp tervezzük a modellünk tanítását egy bővített jellemző halmazon is elvégezni, és az így kapott eredményeinket különböző metrikák szerint összehasonlítani a szűkebb halmazon kapottakkal.
- Mi fog történni az üres cellákat tartalmazó rekordokkal, ha vannak?
    - Mivel azoknak a rekordoknak a száma, amikben van üres cella, nem jelentős, ezért egyszerűen el fogjuk dobni őket (körülbelül 2 ezer rekord a 45 ezerből). Azért is döntöttünk így, mert 1-1 hiányzó adatot nem egyértelmű, mely másik rekordok alapján lehet hatékonyan prediktálni.
- Előfeldolgozási lépések
    - A tanításhoz használt jellemző vektorok közül a szöveges információk csak adott, igen kis méretű halmazokból vehetnek fel értékeket. Így ezeket először egy OrdinalEncoder segítségével egyenletes eloszlású számsorrá alakíthatjuk, majd egy MinMaxScaler segítségével lenormálhatjuk a [0,1] intervallumba.
    - Ezek után a többi, számszerű adatot tartalmazó jellemző vektort egy StandardScaler segítségével fogjuk transzformálni, ezzel a kapott oszlopokban az átlagunk 0, a szórásunk pedig 1 lesz.
    - A két transzformáció eredményét összesítve készen is áll az adathalmazunk a később definiált modellünk betanítására.
- A futtatásokhoz jelenleg nem tervezünk külső szkript-eket írni, a futtatási paramétereket külön kód blokkokba szervezzük ki a egyes szekciók elején, és minden programkód a Google Colaboratory Notebook-ban lesz megtalálható.

Modellezés (MK - II)

- Az értékeléshez használt metrikák
    - Mivel a feladatunk osztályozás, ezért a scikit-learn “Metrics and scoring” oldalának tanulmányozásával (link) az alábbi metrikákat választottuk:
        i. **accuracy_score** (A többcímkés osztályozás esetén a függvény visszaadja a részhalmaz pontosságát.)
        ii. **balanced_accuracy_score** (mivel nem egyenletesen oszlanak el a címkéink, ezért a pontosságot ezek súlyozásával is megkaphatjuk.)
        iii. **recall_score** (Egy arányt ad vissza. (TP / (TP + FN)) , ahol TP a true-positive, FN pedig a fals pozitív eredmények száma. Segítségével meghatározhatjuk az összes pozitív mintát.)
        iv. **precision_score** (Egy arányt ad vissza. (TP / (TP + FP)), ahol TP a true-positive, FN pedig a fals negatív. Segítségével elkerülhetjük, hogy a negatív címkék pozitívnak legyenek kategorizálva.)
        v. **f1_score** (A recall_score és a precision_score-nak a harmónikus középértéke. A harmonikus középérték miatt ez az érték csak akkor lesz magas, ha mindkettő metrika magas.)
        vi. **brier_score_loss** (A Brier-pontszám az előrejelzett valószínűség és a tényleges eredmény közötti átlagos négyzetes különbséget méri.)
        vii. **confidence interval** (Valószínűségi intervallum. Azt adja meg, hogy valószínűleg minden érték tartományba fog esni az eredményünk. A konfidencia intervallum adott szignifikancia-szinten: a becsült változó alsó és felső korlátja.)
- Baseline metódus
    - Először egy **véletlenszerű felcímkézést** végzünk el a teszt halmaz elemein. Ekkor nem az 50%-os pontosság az elvárt, hiszen sokkal több “no” címkénk van az adatbázisunkban, mint “yes”.
    - Ehelyett csinálhatunk a címkék arányának ismeretében egy **adott eloszlású véletlen címkézést**, azaz pl. a dobott címkék 80%-a “no” lesz, 20%-a “yes”, ezek pedig véletlenszerűen lesznek elosztva.
    - Ezek után kipróbáljuk a **leggyakoribb címkével** való predikciót.
- Ötletek a modellekhez
    - **Gaussian Naive-Bayes** (A Bayes-tételt alapján az egyes változók között feltételes függetlenséget feltételezve, normál eloszlást felhasználva számolja ki az ismeretlen példányhoz tartozó legvalószínűbb címkét.)
    - **Gaussian Mixture Model** (A Gauss-keverék modell egy valószínűségi modell, amely feltételezi, hogy az összes adat pontot véges számú, ismeretlen paraméterű Gauss-eloszlás keverékéből állítják elő.)
    - **K Nearest Neighbour** (Adott számú legközelebbi szomszéd távolságával súlyozva állapítja meg az ismeretlen példány címkéjét.)
    - **Nearest Centroid** (A Nearest Centroid osztályozó egy egyszerű algoritmus, amely minden osztályt a tagok súlypontjával reprezentál.)
- Optilamizálandó hiperparaméterek
    - **GMM** esetén különböző függőségek kipróbálása
    - **KNN** esetében “legjobb” k megtalálása → Grid Search (Kipróbálja az általunk megadott lehetőségeket, és a legjobb megoldást választja.)

További fejlesztések (MK - III)

Választott irányok:

- **Feature selection használata**

  Jelentős betanulási idő javulást eredményezhet, ha a tanuláshoz használt jellemzők számát csökkentjük. Ha jól választjuk meg, mely jellemzőket hagyjuk el, a kiértékelés során hasonlóan jó eredményeket érhetünk el. Ehhez a scikit-learn beépített SelectKBest, valamint VarianceThreshold függvényeit fogjuk kipróbálni. Ezt akár a teljes adathalmazon megtehetjük, mindenféle kézi jellemző válogatás nélkül.

- **Train-dev-test halmazméretek variálása**

  Szintén javíthatunk a betanulási időn, minimális teljesítménycsökkenéssel, ha megfelelő méretű adathalmazt vágunk le a betanuláshoz a teljes adatbázisból. Túlzottan nagy train size esetén a túltanulás problémájába eshetünk, kevés adaton azonban nehezebb lehet bizonyos mintázatokat betanulni. Különböző méretekkel kísérletezve pontosabb képet kaphatunk arról, melyik megközelítés a leghatékonyabb az általunk használt adatbázisra.

Egyéni feladatok (Bajnóczi Bendegúz)

Feladat -1

Lineáris modellek kipróbálása

Az alábbi módszereket szeretném kipróbálni a már meglévő adathalmazunkon, a teljesítmény javítása érdekében.

- Logistic regression
- Support Vector Machine

Mindkét esetben egy optimális c paraméter keresése a cél.

Feladat -2

A fenti módszerekkel elért eredmények viselkedésének megfigyelése a feladat paramétereinek változása esetén

Továbbá az új fenti módszerek teljesítményét szeretném megvizsgálni a következő paraméterek változtatása mellett.

- *Jellemzők számának változtatása*
- *Tanító példányok számának változása*

Egyéni feladatok (Csiszár András)

Feladat -1

Ensembling módszerek kipróbálása

Az alábbi módszereket szeretném kipróbálni a már meglévő adathalmazunkon, a teljesítmény javítása érdekében.

- *Random Forest Classifier*

  Ennél a metódusnál az együttes minden fája a tanuló adatbázisból pótlással vett mintából épül fel. Továbbá az egyes csomópontok felosztása során a fa felépítésénél a legjobb felosztást megtaláljuk az összes bemeneti jellemzőből vagy a max méretű jellemzők véletlenszerű részhalmazából.E két véletlenszerűségi forrás célja az erdőbecslő varianciájának csökkentése.

- *Extra Tree Classifier* (Extremely Randomized Trees Classifier)

  A Extra Tree Classifier-nél még véletlenszerűbb lesz a felosztások kiszámítása. A Random Forest-hez hasonlóan a jelölt jellemzők véletlenszerű részhalmazát használják, de ahelyett, hogy a leginkább megkülönböztető küszöbértékeket keresnénk, a küszöbértékeket véletlenszerűen húzzuk meg minden egyes jelölt jellemzőhöz, és a véletlenszerűen generált küszöbértékek közül a legjobbat választjuk felosztási szabályként. Ez általában lehetővé teszi a modell szórásának további csökkentését, viszont a torzítás valamelyes nőni fog.

Feladat -2

Új módszerekkel elért eredmények javítása különböző módokon

Továbbá az új fenti módszereken szeretnék további két megoldást alkalmazni annak érdekében, hogy tovább javítsam a teljesítményt.

- *Cross-validation*

  A Cross-Validation (Kereszt-validáció) a modell ellenőrzési technikák egy fajtája annak felmérésére, hogy a statisztikai elemzés eredményei hogyan általánosulnak egy független adathalmazra. Ez egy olyan újramintavételezési módszer, amely az adatok különböző részeit használja a modell különböző iterációkon történő teszteléséhez és betanításához.

- *Grid Search*

Egyéni feladatok (Mészáros Zsolt)

Feladat-1

A meglévő adatbázison az alábbi új módszereket próbálom ki, annak érdekében, hogy lehetőleg jobb eredményt érjek el a mostaninál.

- Lineáris regresszió
    - Akkor hatékony, ha a döntési határ lineáris, jól elkülönülnek az adott osztályok egymástól. Egy hiperparaméter finomhangolásával lehet a határon lévő pontokat meghatározni, itt ennek a megkeresése lesz a feladatom.
- Decision tree classifier
    - Felügyelt tanulási módszer, amelyet osztályozásra és regresszióra használnak. Amennyiben elkészült egy olyan modell, ahol az adat jellemzőkből kikövetkeztetett egyszerű döntési szabályok megtanulásával megjósolható egy célváltozó értéke.

Feladat-2

Előfeldolgozott adatbázison a már korábban elkészített és legjobb eredményeket hozó tanuló algoritmusokat fogom kipróbálni. Itt feladat a megtalált adatbázist a megfelelő formátumra hoznom, majd a különböző algoritmusokat kipróbálnom és az eredményeket összegeznem.

A kiválasztott adatbázis:
https://www.kaggle.com/datasets/uciml/german-credit

Egyéni feladatok (Szakál Mátyás)

Feladat -1

Keras classifier

- Egy Dense neuronhálós megvalósítás létrehozása a cél. N epoch számmal, illetve N réteggel. (Ezek számának és típusának megállapítása tesztelést igényel). A kapott adatok grafikus ábrázolása, következmények levonása.

Feladat -2

Keras + GridSearch

- Az előző kód kiegészítése egy GridSearch-ös megvalósítással, a két megoldási mód összehasonlításával.

Mindkettő megvalósítási mód egy-egy mély-tanuló algoritmus alkalmazása. A tanítás során a modell frissíti a paramétereit, hogy a legjobb eredményt érje el. A kapott eredményekből megállapítható lesz, hogy az algoritmus underfit vagy overfit állapotba került. Ha ezek előfordulnak, akkor a megfelelő lépésekkel ki kell küszöbölni.

Egyéni feladatok (Vass Máté)

Feladat 1:
GRP, PCA és LDA használata PipeLine felhasználásával

Teljesítmény, illetve betanulási idő javulást is eredményezhet, ha az adatbázisunkon dimenzió redukciót hajtunk végre. Ezzel kisebb dimenziószámú adathalmazt kapunk, mégpedig új jellemzők létrehozásával. Természetesen az új jellemzők létrehozásának a módját nem nekünk kell kitalálnunk, hanem használhatjuk a scikit-learn beépített függvényeit:

- Gaussian Random Projection (GRP)
- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)

Ezen módszereket az eddig megírt programkódunkhoz PipeLine-ok segítségével fogom hozzáadni.

Feladat 2:
Legjobb módszerek tesztelése kiegyensúlyozott adatbázison

Ebben a részben a korábbiakban megírt legjobb gépi tanuló algoritmusainkat fogom tesztelni egy kiegyensúlyozott adatbázison. Ehhez természetesen ezt az adathalmazt is meg kell tisztítanom, ami minimálisan különbözhet a korábban leírt módszereinktől.

Az eddigiekben használt banki adatbázisunkban két címke volt, körülbelül 16-84%-os eloszlásban. Így egy egyszerű DummyClassifier-rel leggyakoribb címkét adva is körülbelül 80%-os pontosságra számíthatunk. Egy kiegyensúlyozott adathalmazon (szintén két címke esetén) ez nagyjából csak 50%-ot eredményezne. Ekkor láthatóvá válna, hogy a bonyolultabb gépi tanuló módszerek valóban mennyivel is jobbak egy egyszerű címkézési megközelítésnél.

Választott adatbázis:
https://archive.ics.uci.edu/ml/datasets/banknote+authentication