HAZI05

Az órai feladat alapján készíts egy KNNClassifier osztályt, viszont az óraival ellentétben nem használhatsz numpy-t csak pandast.

- Mindenhol ahol numpy-t használtunk (adattárolás, broadcast, sort ...) helyettesítsd egy pandas megfelelővel.

- Ahol az órai feladatban list-et használtunk ott most is használhatsz list-et

- A funkcionalitás sehol ne változzon viszont a megvalósítás igen
- A confusion_matrix függvény-nél nem kell átkasztolni df-re ott maradjon a np.ndarray visszatérési típusnak

A mellékelt csv-ben egy másik adatset-et találtok mint az órán. A feladat nem változik, csak annyival, hogy több feature van. Ugyan úgy kezeljétek mint az órait is tettétek.

A load_csv-ben a shuffle-nél a random_state 42 legyen. (nem shuffle-nek hívják a függvényt amit a keveréshez kell használni :D)

Kötelező függvények amelyeknek szerepelnie kell:

- load_csv:

- train_test_split:

- euclidean:

- predict:

- accuracy:

- plot_confusion_matrix:

Ezek ugyan azok mint az órán, nem írok hozzá semmilyen plussz megjegyzést, a GYAK05.pdf -ben megtaláljátok.

További feladatok:

- Készíts egy függvényt ami visszaadja, hogy melyik k értékkel kaptuk a legjobb accuracy-t. A függvény visszatérési értéke egy Tuple legyen az első paramétere a k értéke a második pedig az accuracy két tizedesre kerekítve. Pl.: (3, 77.02) A k értékét 1-től 20-ig nézzétek.

- Készíts egy rövid szöveges leírást az egész workflow-ról, mit miért csináltál, pl. Miért split-elünk?, Hogyan működik az euclidean függvény?, Hogyan működik az accuracy függvény? Ha minden függvényhez írtok egy két mondatot az bőven elég. Ezt egy .txt file-ban a HAZI05 mappába tegyétek és pusholjátok a repoba.

NOTES:

-az óraihoz hasonló workflow gyorsíthat kicsit a fejlesztésen, tehát notebook-ban minden egyes függvényt külön-külön teszteltek és fejlesztetek és ha mindegyik működik, akkor egy class-ba össze szedhetitek és akkor már az egyes függvények funkcionalitásán nem kell aggódni, hogy működik-e

- A pandas-ba átíráshoz egy kis tipp, a legtöbb helyen ahol át kell írni, lesz rá pandas függvény, nem kell túl bonyolítani. Ha mondjuk egy shuffle kell az adatbetöltésnél akkor ha rákeresel, hogy (pandas df shuffle row wise) akkor lesz megoldás rá és beépített függvény. Ha egy probléma megfogalmazódik benned akkor keress rá angolul.
- Pandas-ban ha broadcast-ni akarsz vagy element wise összehasonlítani akkor fontos, hogy figyelj a df indexeire, ez majd az y_preds és y_test-nél lehet fontos ott ha elszáll a progi, akkor nézd meg mind a kettő df-nek az indexeit és ha eltérnek akkor a .reset_index(drop=True) segíthet. (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.reset_index.html)
  -