Teoria uczenia maszynowego: miniprojekt

Uwagi ogólne:

- Projekt może zostać wykonany *w parach*
- Format: *Jupyter Notebook* lub *PDF* z raportem zrobionym w innym narzędziu (nie ma tutaj ograniczeń, ale proszę nie przesyłać dokumentów typu .docx)
- Długość sprawozdania nie podlega ocenie, ale proszę nie przekraczać 4-5 stron!
- Wkład do zaliczenia: 40% oceny z ćwiczeń (pozostałe 60% to test końcowy)
- Nie ma potrzeby przesyłania kodu, można w raporcie umieścić link do np. githubu
- Termin oddania: **31 maja 2024**. Opóźnienie może skutkować karą w postaci obniżenia punktów. *Uwaga*: opóźnienie sięgające ostatniego tygodnia semestru (10-14 czerwca) może uniemożliwić zaliczenie przedmiotu w pierwszym terminie! (oceny muszą być wystawione i zatwierdzone w systemie do końca semestru).

Celem projektu jest wykonanie jednego eksperymentu obliczeniowego badającego zachowanie się algorytmu uczenia maszynowego względem zmian parametrów. Każdy zespół samemu wymyśla eksperyment, w tym dobiera wybrane przez siebie 1-2 zbiory danych. Poniżej kilka przykładowych tematów:

- Charakterystyka błędu na zbiorze treningowym i testowym w funkcji siły regularyzacji w regresji liniowej / regresji logistycznej / SVM (uwaga: współczynnik regularyzacji można dobierać na skali wykładniczej, np. $2^{-10}, 2^{-9}, \ldots, 2^{9}, 2^{10}$, itp.)
- Podobna charakterystyka dla innych algorytmów uczących względem ich złożoności, np. liczby klasyfikatorów w *Gradient Boosting*, głębokości drzewa dla pojedynczego drzewa decyzyjnego, stopnia wielomianu w regresji wielomianowej, horyzontu czasowego wstecz dla modeli autoregresywnych (szeregi czasowe), liczby lub szerokości warstw dla sieci neuronowej, itp.
- Podobna charakterystyka względem rosnącej wielkości zbioru uczącego n wraz z dopasowanie wykładnika α krzywej spadku błędu, error $\sim n^{-\alpha}$
- Charakterystyka spadku błędu zastępczego (na zbiorze treningowym i testowym), np. entropii skrośnej i jej relacji do błędu klasyfikacji, w zależności od rozmiaru zbioru treningowego n, np. dla regresji logistycznej lub sieci neuronowych
- Przebieg procesu uczenia dla sieci przy użyciu różnych algorytmów optymalizacji np. SGD, AdaGrad, Adam, itp.
- Przebieg procesu uczenia dla sieci w zależności od doboru szybkości uczenia np. w algorytmie SGD
- Próba odtworzenia zjawiska *double descent* – błąd testowy w funkcji złożoności modelu maleje, potem rośnie, potem znowu maleje
- Badanie charakterystyki zmian wag sieci w zależności od głębokości warstwy – czy wagi w głębszych warstwach faktycznie ulegają małym zmianom względem inicjalizacji?
- Badanie zjawiska *lottery ticket hypothesis* (pozostawienie tylko największych wag w nauczonej sieci, a następnie nauczenie od zera w ten sposób zmniejszonej sieci *przy tej samej inicjalizacji* skutkuje tylko niewielkim spadkiem jakości)
- Badanie rzadkości (zerowania) wartości post-aktywacji w architekturze transformer (wyjścia z warstwy atencji – już po zastosowaniu ReLU – cechują się dużą liczbą zer)
- Wpływ dodawania szumu na trafność algorytmów uczenia: np. dodanie dodatkowych cech wejściowych z czystym szumem (nawet wielu), zaszumienie istniejących cech, zaszumienie wartości na wyjściu (np. losowe zmiany etykiet klas w części obiektów).
- Wpływ tzw. wartości odstających (*outliers*, *out-of-distribution observations*) na trafność - można dodać takie obserwacje sztucznie do zbioru danych
- Czy LLM potrafią zrobić analizę statystyczną? Przygotowanie prompta (wraz z wynikami) np. do GPT-4, dzięki któremu może dostać gotowy kod umożliwiający przeprowadzenie np. jednego z powyższych eksperymentów. Alternatywnie: użycie LLM do analizy / interpretacji gotowych wyników / wykresów eksperymentu (wraz z własnym komentarzem odnośnie trafności tej analizy)