Relazione Voted Perceptron

Per la risoluzione della richiesta a parte le funzioni e classi definite in utils.py e in Voted_Perceptron.py, è stato fatto utilizzo delle librerie:

- sklearn per l'implementazione delle metriche e della confusion matrix
- time per il calcolo del tempo di esecuzione
- random per la scelta casuale della parte di dataset da utilizzare per train e predict
- math per le funzioni sqrt e pow
- numpy per le funzione zeros e norm

I 3 Dataset differiscono per volume di dati:

- Dataset 1 (QSAR ANDROGEN RECEPTOR) presenta 1024 attributi per 1687 istanze

- Dataset 2 (QSAR ORAL TOXICITY) presenta 1024 attributi per 8992 istanze

- Dataset 3 (QSAR BIODEGRADATION), presenta 41 attributi per 1055 istanze

Come si può notare, sono stati scelti dataset seguendo una logica che permetta una varia sperimentazione, nel dettaglio abbiamo: un dataset con un basso numero di istanze e di attributi (3), un dataset con un basso numero di istanze ma un alto numero di attributi (1) ed un dataset con un alto numero di istanze e di attributi (2).

Tale distinzione è fondamentale per comprendere come effettivamente la complessità del programma, i cui tempi di esecuzione scalano a dismisura con il volume di dati da trattare.
Ecco alcuni tempi:

- Dataset 3, 80% train, 10 ripetizioni, 1 epoca: 10.075206995010376s
- Dataset 3, 80% train, 10 ripetizioni, 2 epoche: 19.486937761306763s
- Dataset 3, 80% train, 10 ripetizioni, 3 epoche: 27.325947761535645s

- Dataset 1, 80% train, 10 ripetizioni, 1 epoca: 366.2578971385956s
- Dataset 1, 80% train, 10 ripetizioni, 2 epoche: 634.2080867290497s
- Dataset 1, 80% train, 10 ripetizioni, 3 epoche: 1021.2671251296997s

- Dataset 2, 80% train, 10 ripetizioni, 1 epoca: 8702.229641199112s
- Dataset 2, 80% train, 10 ripetizioni, 2 epoche: 14857.505167007446s
- Dataset 2, 80% train, 10 ripetizioni, 3 epoche: non provato

Queste misure giustificano inoltre i seguenti esperimenti in quanto effettuati solo su Dataset 3, ma che al netto di differenze dovute alla randomizzazione ed alla conseguente distribuzione di positivi e negativi, si riflettono allo stesso modo anche negli altri due dataset.

Le seguenti tabella sono costruite per 10 ripetizioni di train/predict, con percentuale di addestramento sul dataset (3) del 80% e numero epoche massime da 1 a 3.

	Accuracy		Precision	
	Mean	Std. Deviation	Mean	Std. Deviation
1	0.6587677725118484	0.03071441089292826	0.20714285714285713	0.23112822629805635
2	0.6862559241706162	0.044052891141695714	0.57375	0.41125208731505264
3	0.7450236966824646	0.0569071332823731	0.7155984809997967	0.12597312182519468

	F1		Recall	
	Mean	Std. Deviation	Mean	Std. Deviation
1	0.025379576999353736	0.03451194462326357	0.013721323017097664	0.018618616769145 14
2	0.1545319211680768	0.2310074040524611	0.12058945389440745	0.21254937185042752
3	0.4628865544703 29	0.2660207640615485	0.4077474339225528	0.2830597793529659

NOTE FINALI: In alcuni casi (soprattutto nel caso di basso numero di addestramento/predizione) è possibile che che F1, Recall e Precision abbiano media e deviazione standard pari a 0, questo si verifica in casi in

cui la formula presenta una forma di indecisione del tipo $\frac{0}{0}$ in tali casi, mediante il parametro zero_division del metodo, viene imposto di restituire zero, di default il programma avrebbe anche lanciato un warning.

La randomizzazione dei dataset, viene operata prendendo un vettore casuale di indici compresi fra 0 e la lunghezza dei dataset, successivamente andiamo a costruire i dataset di train con le righe di dataset corrispondenti a indici contenuti nel vettore, qualora un indice non fosse nel vettore, la corrispondente riga viene utilizzata per il predict.