Analisi dei test effettuati

Test di Stefano Biddau su MNIST con K-Nearest Neighbors

Cenni teorici K-Nearest Neighbors

DEFINIZIONE:

L'algoritmo k-nearest neighbors noto anche come KNN o k-NN è un classificatore di apprendimento supervisionato non parametrico che utilizza la prossimità per effettuare classificazioni o previsioni sul raggruppamento di un singolo punto di dati. Sebbene possa essere utilizzato per problemi di classificazione o regressione viene principalmente utilizzato per le problematiche di prima tipologia, basandosi sul presupposto che punti simili possono essere trovati l'uni vicino all'altro.

FUNZIONAMENTO

Per problemi di classificazione un' etichetta di classe viene assegnata sulla base di un voto a maggioranza, ad esempio viene utilizzata l'etichetta più frequente rappresentata attorno ad un determinato punto di dati. Ad esempio se ci sono solo due categorie la scelta ricadrà sull'etichetta che per maggioranza supererà il 50%. Quando ci sono più classi è logico pensare che la soglia si abbassi.

PRINCIPALI IPER-PARAMETRI DEL MODELLO

Vediamo quali sono (per la mia esperienza) i principali parametri o iper-parametri di un modello KNN, prendendo in considerazione l'implementazione della libreria scikit-learn in Python.

Per prima cosa si considera la **metrica di distanza** che aiutano a formare confini decisionali tra il punto di interrogazione e gli altri punti dati. Noi consideriamo diverse metriche per i nostri test:
- **euclidea**: misura una linea retta tra il punto di query e l'altro punto che si sta misurando;
- **manhattan**: misura il valore assoluto tra due punti;
- **minkowski**: questa misura della distanza è la forma generalizzata delle metriche di distanza euclidea e di manhattan;

Altro parametro molto importante che incide sulle prestazioni del classificatore è il **valore k** che è associato al numero di vicini che verranno controllati per determinare la classificazione di un punto specifico di query. Ad esempio se k = 1, l'istanza verrà assegnata alla stessa classe del suo singolo vicino più vicino. Definire k può essere un atto di bilanciamento in quanto valori diversi possono portare ad overfitting o underfitting. La determinazione di questo parametro dipende solo ed esclusivamente dai dati in input del dataset. Per convenzione si utilizzano principalmente dei valori dispari per evitare pareggi nella classificazione.

Infine l'ultimo parametro preso in considerazione è il **peso (weights)**, esso è associato alla funzione di peso utilizzata nella previsione dei possibili valori.
Può essere di due tipi:
- **uniforme**: tutti i punti in ogni vicinato sono ponderati allo stesso modo;
- **distanza**: in questo caso i vicini più vicini avranno un'influenza maggiore dei vicini più lontani;

PRO E CONTRO DELLA K-NEAREST NEIGHBORS

PRO

- Facile da implementare data la sua semplicità e allo stesso tempo forte accuratezza;
- Si adatta facilmente quando vengono aggiunti nuovi campioni
- Pochi iper parametri rispetto ad altri algoritmi di Machine Learning

CONTRO

- Non ha una buona scalabilità essendo un algoritmo pigro, occupa più memoria e spazio di Storage rispetto agli altri classificatori;
- Non funziona molto bene con input di dati ad alta dimensione;

TEST EFFETTUATI

MNIST

Eseguendo lo script per l'ottimizzazione degli iper parametri salta all'occhio che nelle prime 10 posizioni delle simulazioni effettuate, con i migliori responsi in termini di accuratezza troviamo principalmente il numero di vicini settato con questi valori: (3, 5, 7, 9, 11, 13)
Testiamo anche con il numero di vicini limite scelto che è 21 e poi testiamo con un numero di vicini che sfora il range prestabilito ossia 25.

1° Blocco di test (Aumento vicini, w: uniform, m: minkowski)

Indice test	Numero Vicini	Weights	Metric	Acc Train	Acc Test	Differenza MSE	Under-fit
1	**3**	**Uniform**	**Minkowski**	**98,67%**	**97,05%**	**0,32**	**NO**
2	5	Uniform	Minkowski	98,19%	96,88%	0,29	NO
3	7	Uniform	Minkowski	97,91%	96,94%	0,24	NO
4	9	Uniform	Minkowski	97,63%	96,59%	0,25	NO
5	11	Uniform	Minkowski	97,42%	96,68%	0,17	SI
6	13	Uniform	Minkowski	97,24%	96,53%	0,18	SI
7	21	Uniform	Minkowski	96,68%	96,30%	0,11	SI
8	25	Uniform	Minkowski	96,47%	96,09%	0,11	SI

Da questi primi 8 test si evince che all'aumentare dei vicini il modello perde in accuratezza sia sul set di test ma sopratutto sul set di train, il cui valore dopo il test 4 scende vertiginosamente. Infatti possiamo notare di come la differenza di MSE diventi sempre più piccola. Nonostante ciò non ci troviamo davanti ad un caso di overfitting perché come possa notare la differenza di MSE non sale con l'aumentare dei vicini tuttavia l'errore sul set di trai e sul set di test aumenta sempre più quindi è bene parlare di underfitting. I migliori risultati si riscontrano con il **test 1 col la percentuale di accuratezza sul set di test più alta.**

Adesso proviamo a cambiare metrica quindi passiamo da minkowski -> euclidean

2° Blocco di test (Aumento vicini, w: uniform, m: euclidean)

Indice test	Numero Vicini	Weights	Metric	Acc Train	Acc Test	Differenza MSE	Under-fit
1	**3**	**Uniform**	**Euclidean**	**98,67%**	**97,05%**	**0,32**	**NO**
2	5	Uniform	Euclidean	98,19%	96,88%	0,29	NO
3	7	Uniform	Euclidean	97,91%	96,94%	0,24	NO
4	9	Uniform	Euclidean	97,63%	96,59%	0,25	NO
5	11	Uniform	Euclidean	97,42%	96,68%	0,17	SI
6	13	Uniform	Euclidean	97,24%	96,53%	0,18	SI
7	21	Uniform	Euclidean	96,68%	96,30%	0,11	SI
8	25	Uniform	Euclidean	96,47%	96,09%	0,11	SI

Al cambio della metrica il modello non subisce variazioni in termini di accuratezza ne in meglio ne in peggio ma mantiene lo stesso andamento del blocco di test precedente con gli stessi valori percentuali sia sul set di test che sul set di train, nonché la stessa differenza MSE.

Adesso proviamo a cambiare metrica quindi passiamo da euclidean -> manhattan e vediamo se otteniamo ancora gli stessi risultati

3° Blocco di test (Aumento vicini, w: uniform, m: manhattan)

Indice test	Numero Vicini	Weights	Metric	Acc Train	Acc Test	Differenza MSE	Under-fit
1	**3**	**Uniform**	**Manhattan**	**98,28%**	**96,33%**	**0,36**	**NO**
2	5	Uniform	Manhattan	97,74%	96,18%	0,32	NO
3	7	Uniform	Manhattan	97,39%	96,15%	0,28	NO
4	9	Uniform	Manhattan	97,10%	95,97%	0,24	NO
5	11	Uniform	Manhattan	96,87%	95,85%	0,20	SI
6	13	Uniform	Manhattan	96,66%	95,80%	0,18	SI
7	21	Uniform	Manhattan	95,96%	95,44%	0,10	SI

Per questi 7 test effettuati notiamo un comportamento analogo ai due blocchi precedenti, dopo aver raggiunto i migliori risultati con il **test 1** in termini di accuratezza sul set di test, all'aumentare dei vicini abbiamo una vertiginosa discesa in termini di punti percentuali sia sul set di train che sul set di test. Mentre all'aumentare dei vicini notiamo che la differenza di MSE si assottiglia sempre più. Anche in questo caso ci troviamo davanti ad un fenomeno di underfitting.

Diversamente dai due blocchi precedenti **la metrica manhattan fa lievitare considerevolmente il tempo di addestramento del modello** che passa da una manciata di minuti, a più di 30 minuti di addestramento, cosa che poi risulta oltretutto controproducente perché i tassi di accuratezza sui, in termini di percentuale, sui set di trai e di test sono addirittura più bassi rispetto agli altri dei blocchi precedenti.

Adesso ripetiamo i test effettuati cambiando il peso da uniform -> distance

4° Blocco di test (Aumento vicini, w: distance, m: minkowski)

Indice test	Numero Vicini	Weights	Metric	Acc Train	Acc Test	Differenza MSE	Under-fit
1	3	Distance	Minkowski	100%	97,17%	0,52	-
2	5	Distance	Minkowski	100%	96,91%	0,59	-
3	7	Distance	Minkowski	100%	97,00%	0,60	-
4	9	Distance	Minkowski	100%	96,73%	0,65	-
5	11	Distance	Minkowski	100%	96,78%	0,63	-
6	13	Distance	Minkowski	100%	96,65%	0,66	-
7	21	Distance	Minkowski	100%	96,32%	0,72	-

Da questi altri 7 test si evince che il cambio di peso influisce nettamente sul modello rendendolo in utile in quanto l'accuratezza sul set di train raggiunge il 100% che è sintomo di un modello inefficiente che non predice ma impara meccanicamente.

5° Blocco di test (Aumento vicini, w: distance, m: euclidean)

Indice test	Numero Vicini	Weights	Metric	Acc Train	Acc Test	Differenza MSE	Under-fit
1	3	Distance	Euclidean	100%	97,17%	0,52	-
2	5	Distance	Euclidean	100%	96,91%	0,59	-
3	7	Distance	Euclidean	100%	97,00%	0,60	-
4	9	Distance	Euclidean	100%	96,73%	0,65	-
5	11	Distance	Euclidean	100%	96,78%	0,63	-
6	13	Distance	Euclidean	100%	96,65%	0,66	-
7	21	Distance	Euclidean	100%	96,32%	0,72	-

I risultati ottenuti da questo blocco di 7 test evidenziano il rapporto che abbiamo riscontrato con i risultati tra il **Blocco 1 e il Blocco 2** ossia che le metriche euclidean e minkowsky fanno ottenere gli stessi risultati in termini di percentuale sia sul set di test che sul set di train. Chiaramente se il Blocco 4 ci mostra un modello che non apprende ma impara meccanicamente, lo stesso risultato si ottiene con questo blocco di test.

6° Blocco di test (Aumento vicini, w: distance, m: manhattan)

Indice test	Numero Vicini	Weights	Metric	Acc Train	Acc Test	Differenza MSE	Under-fit
1	3	Distance	Manhattan	100%	96,40%	0,67	-
2	5	Distance	Manhattan	100%	96,29%	0,71	-
3	7	Distance	Manhattan	100%	96,23%	0,73	-
4	9	Distance	Manhattan	100%	96,05%	0,76	-
5	11	Distance	Manhattan	100%	96,00%	0,77	-
6	13	Distance	Manhattan	100%	95,93%	0,79	-
7	21	Distance	Manhattan	100%	95,59%	0,85	-

I risultati di questi 7 test mostrano l'andamento dei due blocchi precedenti con peso distanze, ossia un modello che non predice ma offre una classificazione in modo meccanico.

Possiamo stilare una lista di 3 test, quelli con i migliori risultati ottenuti in termini di percentuale di accuratezza sul set di test:

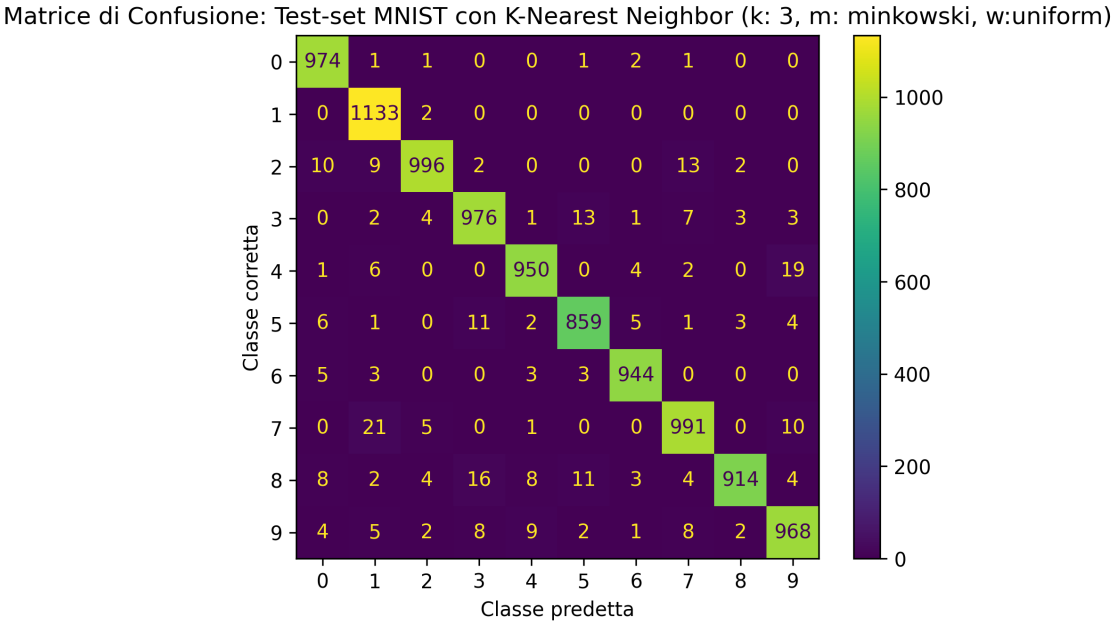
1) Blocco 1 - Test 1: Test del 2022-11-10 15:13:47
2) Blocco 2 - Test 1: Test del 2022-11-10 16:28:43
3) Blocco 3 - Test 1: Test del 2022-11-11 01:39:00

1) Test del 2022-11-10 15:13:47 MNIST con KNN (k: 3, w: uniform, m: minkowski)

Report di classificazione

	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.96	1.00	0.98	1135
2	0.98	0.97	0.97	1032
3	0.96	0.97	0.96	1010
4	0.98	0.97	0.97	982
5	0.97	0.96	0.96	892
6	0.98	0.99	0.98	958
7	0.96	0.96	0.96	1028
8	0.99	0.94	0.96	974
9	0.96	0.96	0.96	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

Dal report di classificazione si evince che la classe con la miglior accuratezza è la **classe 1 -> cifra 1**, mentre la peggiore corrisponde alla **classe 5- > cifra 5**.



La matrice di confusione conferma i dati mostrati dal report di classificazione con la **classe 1 -> cifra 1** su **1135 campioni** disponibili ne classifica correttamente **ben 1133**. Mentre in

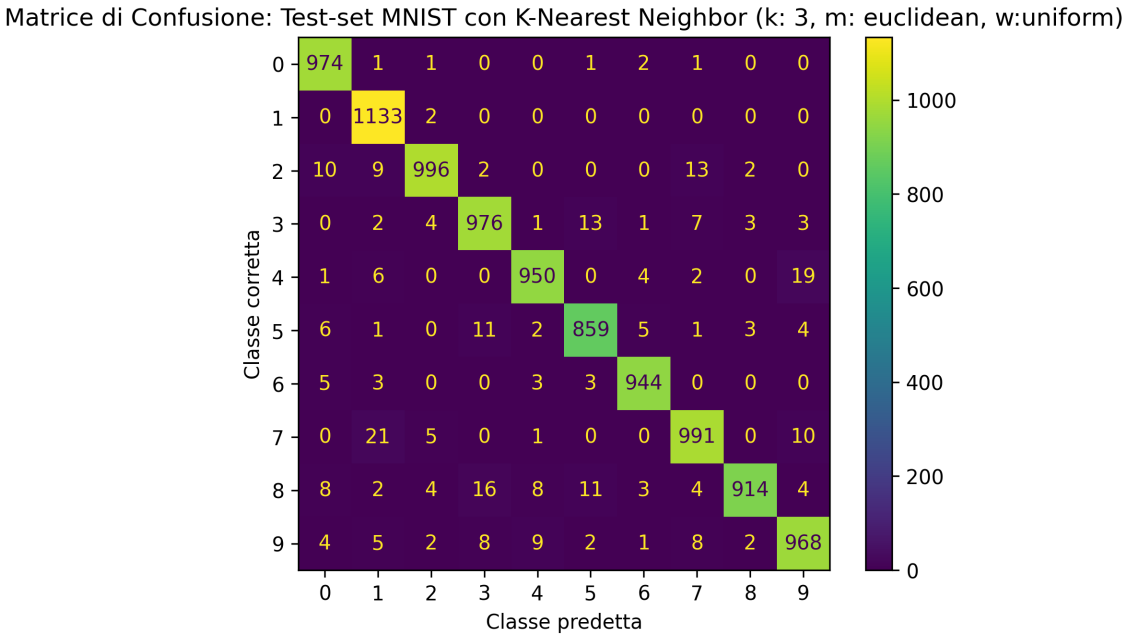
contrapposizione abbiamo la **classe 5 -> cifra 5** che ottiene i peggiori risultati possibili anche perchè ha il minor numero di campioni. Infatti su **892 disponibili** ne classifica correttamente **859**.

2) **Test del 2022-11-10 16:28:43 MNIST con KNN (k: 3, w: uniform, m: euclidean)**

Report di classificazione

	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.96	1.00	0.98	1135
2	0.98	0.97	0.97	1032
3	0.96	0.97	0.96	1010
4	0.98	0.97	0.97	982
5	0.97	0.96	0.96	892
6	0.98	0.99	0.98	958
7	0.96	0.96	0.96	1028
8	0.99	0.94	0.96	974
9	0.96	0.96	0.96	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

Dal report di classificazione si evince che la classe con la miglior accuratezza è la **classe 1 -> cifra 1**, mentre la peggiore corrisponde alla **classe 5- > cifra 5**.



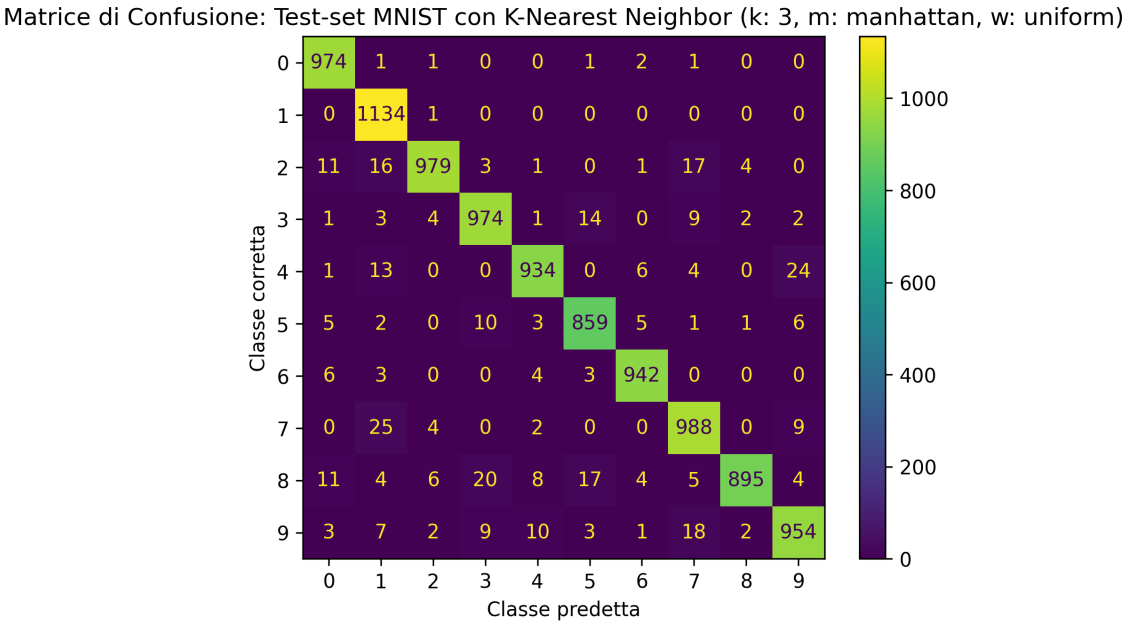
La matrice di confusione conferma i dati mostrati dal report di classificazione con la **classe 1 -> cifra 1** su **1135 campioni** disponibili ne classifica correttamente **ben 1133**. Mentre in contrapposizione abbiamo la **classe 5 -> cifra 5** che ottiene i peggiori risultati possibili anche perchè ha il minor numero di campioni. Infatti su **892 disponibili** ne classifica correttamente **859**.

3) Test del 2022-11-11 01:39:00 MNIST con KNN (k: 3, w: uniform, m: manhattan)

Report di classificazione

	precision	recall	f1-score	support
0	0.96	0.99	0.98	980
1	0.94	1.00	0.97	1135
2	0.98	0.95	0.97	1032
3	0.96	0.96	0.96	1010
4	0.97	0.95	0.96	982
5	0.96	0.96	0.96	892
6	0.98	0.98	0.98	958
7	0.95	0.96	0.95	1028
8	0.99	0.92	0.95	974
9	0.95	0.95	0.95	1009
accuracy			0.96	10000
macro avg	0.96	0.96	0.96	10000
weighted avg	0.96	0.96	0.96	10000

Dal report di classificazione si evince che le classi con la miglior accuratezza sono: **classe 0 -> cifra 0**, **classe 1 - > cifra 1**. Mentre la classe con la peggior accuratezza è la **classe 8 -> cifra 8**



La matrice di confusione conferma quanto mostrato dal report di classificazione, la classe che ottiene i migliori risultati è la **classe 0 -> cifra 0**, insieme alla **classe 1 - > cifra 1**, chiaramente si

da una leggera preferenza alla classe 0 dato che ottiene un accuratezza maggiore con un minor numero di campioni rispetto alla classe 1.
Analogamente la classe che si comporta meno bene è la **classe 8 -> cifra 8** che su **974** campioni disponibili ne classifica correttamente soltanto **895**.