Analisi dei test effettuati

Test di Stefano Biddau su F-MNIST con Foreste randomiche

Cenni teorici Random Forest

DEFINIZIONE:

Uno degli algoritmi **più utilizzati nel Machine Learning** è proprio quello denominato **Random Forest**. Esso è un **algoritmo di tipo supervisionato** che può essere usato sia per task di **classificazione** che di **regressione**.
Caratteristica di questo algoritmo è quella di essere un modello "**ensemble**" ovvero un modello che al proprio interno mette **insieme altri modelli** più semplici. In particolare la tipologia di ensemble rappresentata dal random forest è quella denominata "**bagging**" che sta per **Boostrap aggregating**.

FUNZIONAMENTO

Quando si decide di utilizzare un Random Forest, l' obiettivo sarà quello di **creare tanti classificatori deboli** che uniti insieme possono portare ad un risultato eccellente. La caratteristica di questo algoritmo è che **gli alberi sono indipendenti**, non si condizionano a vicenda. Questo permette di evitare alcuni degli errori più comuni degli alberi, infatti gli alberi rischiano di essere molto instabili a causa del loro modo intrinseco di fare prediction, ovvero il metodo verticale ad albero.
Altra importante peculiarità è quella di scegliere ogni volta un **campione casuale di variabili** da utilizzare per il train interno, il che rende ancora di più indipendenti e differenti gli alberi, permettendo ad ognuno di loro una diversa specializzazione. Il Bagging inoltre fa riferimento alla **tecnica boostrap**, o in italiano **campionamento casuale con rimpiazzo**. Questa tecnica implica che alcuni dati possono comparire contemporaneamente in più modelli mentre altri potrebbero non comparire mai.
Ogni albero vale per uno, e nessuno pesa più degli altri, questo come abbiamo detto è un punto di forza, ma bisogna comunque arrivare ad una qualche conclusione. Possiamo immaginare il **Random Forest come un parlamento**: il modo di decidere è quello del voto, nel caso della classificazione. Se si avranno 5 alberi nel modello e tre ottengono come risultato per una certa osservazione il target 1 e due il target 0, il risultato per maggioranza andrà al target 1.

PRINCIPALI IPER-PARAMETRI DEL MODELLO

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

Tra i parametri da tenere fortemente in considerazione c'è la metrica di splitting, che, misurando la qualità dello split, separerà i dati. Questo parametro si chiama **criterion** e dipende dal task che si vuole svolgere.
Per quanto riguarda il problema della classificazione, può avere "gini" e "entropy" come possibili valori. "Gini" fa riferimento all'impurità di Gini, mentre "entropy" all'informazione gain, basato sul concetto di entropia nella teoria dell'informazione.

Un altro parametro da controllare è quello che si chiama **max_depth**: questo indica quanto profondo sarà l'albero. Più profondo sarà l'albero, più split farà e più sarà preciso sui dati di addestramento. Questo però non vuol dire che se è più preciso sui dati di train il modello sarà in

grado di generalizzarli bene, anzi più sarà profondo più sarà profondo più preciso sarà sui dati di train e mono probabilmente sarà in grado di generalizzare su dati nuovi.
Una maggior profondità quindi rischia di far cadere in quel fenomeno chiamato overfitting ovvero un'ottima capacità sul train ma una pessima sul test che si traduce in un incapacità di generalizzare su nuovi dati.

Un altro parametro considerato è **min_samples_split**, ovvero il numero minimo di campioni necessari per dividere un nodo intero. Nel nostro caso sarà un valore intero.

Altro iper-parametro importante è **n_estimators**. Questo parametro ci da la possibilità di scegliere in numero di alberi nella che faranno parte della foresta

PRO E CONTRO DELLA RANDOM FOREST

PRO
- Modello estremamente performante;
- Poca probabilità di incorrere in overfitting;
- Utile sia per classificazione che per regressione;
- Abbastanza facile da interpretare, essendo formato da algoritmi molto semplici;

CONTRO
- Tempo di esecuzione (solo se rapportato all'albero decisionale, stiamo parlando comunque di una manciata di secondi)

TEST EFFETTUATI

F-MNIST

Dai grafici di accuratezza generati dall'esecuzione dello script per l'ottimizzazione degli Iper parametri notiamo che l'indice di accuratezza si stabilizza al raggiungimento di queste soglie:
- Popolazione: almeno 100 alberi
- Profondità albero: almeno 13

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 una sola profondità ottimale degli alberi che è quella limite ossia 21, tutavia ciò che varia sembra essere il numero di alberi che compongono la foresta.

Testiamo perciò il comportamento del classificatore con profondità per gli alberi: (21) e popolazione (100,120,140,160,180,200). Inoltre per un controllo eventuale dell' overfitting teniamo in considerazione per alcuni blocchi di test i valori 25 e 220 che sforano i range rispettivamente per profondità e popolazione.

1° Blocco di test (Aumento popolazione, profondità: 21, c: Gini, mss: 2)

Indice test	Popolazione massima	Profondità massima	Criterio	Min samples split	Acc Train	Acc Test	Differenza MSE	Over-fit
1	100	21	Gini	2	99,11%	87,43%	1,60	NO
2	120	21	Gini	2	99,13%	87,45%	1,63	NO
3	140	21	Gini	2	99,13%	87,43%	1,62	NO
4	160	21	Gini	2	99,12%	87,44%	1,61	NO
5	180	21	Gini	2	99,13%	87,54%	1,58	NO
6	**200**	**21**	**Gini**	**2**	**99,14%**	**87,62%**	**1,57**	**NO**
7	220	25	Gini	2	99,56%	87,52%	1,65	SI

Da questi primi 6 test si evince che l'aumento della popolazione massima della foresta incide sul livello di accuratezza del set di test in modo molto lieve, mostrando un andamento di accuratezza in termini di percentuale altalenante ma che si mantiene in linea col valore del miglior test (**test 6**), ciò che cambia è il tempo di addestramento del modello:

- 100 -> 97 sec
- 120 -> 114 sec
- 140 -> 133 sec
- 160 -> 152 sec
- 180 -> 172 sec
- 200 -> 197 sec

Il **test 7** mostra come il modello inizia a comportarsi male andando oltre i parametri limite, infatti tale test mostra la differenza MSE più alta tra quelle registrate in questo blocco di test e l'accuratezza sul set di train si è avvicinata considerevolmente al 100% che ricordiamo renderebbe il modello inutile.

Adesso per i prossimi blocchi di test proviamo a cambiare il valore min samples split e vediamo il classificatore come si comporta.

2° Blocco di test (Aumento popolazione, profondità: 21, c: Gini, mss: 5)

Indice test	Popolazione massima	Profondità massima	Criterio	Min samples split	Acc Train	Acc Test	Differenza MSE	Over-fit
1	100	21	Gini	5	96,70%	87,27%	1,31	NO
2	120	21	Gini	5	96,76%	87,16%	1,34	NO
3	140	21	Gini	5	96,79%	87,29%	1,31	NO
4	160	21	Gini	5	96,77%	87,29%	1,31	NO
5	180	21	Gini	5	96,80%	87,20%	1,32	NO
6	**200**	**21**	**Gini**	**5**	**96,81%**	**87,37%**	**1,31**	**NO**
7	220	25	Gini	5	97,05%	87,21%	1,35	SI

Per questi 7 test effettuati raccogliamo dei risultati analoghi al blocco precedente soltanto con valori in percentuale di accuratezza, sul set di test e sul set di train, leggermente più bassi.
Il **test 6** anche in questo caso è quello che fornisce i migliori risultati sia in termini di accuratezza che differenza di MSE. Mentre il **test 7** mostra nuovamente che andare fuori i parametri del range scelto non è una scelta ottimale e oltre a far aumentare la differenza di MSE porterà ad overfitting se i parametri vengono ancora aumentati.
Un' altra considerazione che possiamo fare è quella relativa al tempo di addestramento del modello che come per il blocco precedente aumenta all'aumentare della popolazione di alberi della foresta, tuttavia il valore nuovo di **min_samples_split** lo rende più basso rispetto al blocco precendete:

- 100 -> 90 sec
- 120 -> 108 sec
- 140 -> 125 sec
- 160 -> 143 sec
- 180 -> 160 sec
- 200 -> 182 sec

3° Blocco di test (Aumento popolazione, profondità: 21, c: Gini, mss: 10)

Indice test	Popolazione massima	Profondità massima	Criterio	Min samples split	Acc Train	Acc Test	Differenza MSE	Over-fit
1	100	21	Gini	10	93,38%	86,58%	0,88	NO
2	120	21	Gini	10	93,41%	86,56%	0,90	NO
3	140	21	Gini	10	93,43%	86,62%	0,90	NO
4	160	21	Gini	10	93,46%	86,58%	0,92	NO
5	**180**	**21**	**Gini**	**10**	**93,52%**	**86,64%**	**0,92**	**NO**
6	200	21	Gini	10	93,51%	86,59%	0,92	NO

Per questi 6 test effettuati raccogliamo dei risultati analoghi ai due blocchi precedent con valori in percentuale di accuratezza, sul set di test e sul set di train, ancora più bassi.
Il **test 5** in questo caso è quello che fornisce i migliori risultati in termini di accuratezza. Un' altra considerazione che possiamo fare è quella relativa al tempo di addestramento del modello che come per il blocco precedente aumenta all'aumentare della popolazione di alberi della foresta, tuttavia il valore nuovo di **min_samples_split** lo rende più basso rispetto al blocco precendete:

- 100 -> 84 sec
- 120 -> 100 sec (tranne per questa eccezione dovuta alla macchina sul quale si eseguono i test)
- 140 -> 86 sec
- 160 -> 96 sec
- 180 -> 109 sec
- 200 -> 119 sec

*Ora ripetiamo questi 3 blocchi di test cambiando il **criterio** da **Gini** -> **Entropy** e vediamo se otteniamo migliori risultati*

4° Blocco di test (Aumento popolazione, profondità: 21, c: Entropy, mss: 2)

Indice test	Popolazione massima	Profondità massima	Criterio	Min samples split	Acc Train	Acc Test	Differenza MSE	Over-fit
1	100	21	Entropy	2	99,85%	87,53%	1,67	NO
2	120	21	Entropy	2	99,85%	87,45%	1,67	NO
3	140	21	Entropy	2	99,85%	87,72%	1,62	NO
4	160	21	Entropy	2	99,86%	87,61%	1,65	NO
5	180	21	Entropy	2	99,87%	87,66%	1,64	NO
6	**200**	**21**	**Entropy**	**2**	**99,86%**	**87,84%**	**1,62**	**NO**

Da questi 6 test si evince come il criterio entropia faccia lievitare il tasso di accuratezza sia sul set di train che su quello di test, che mantengono al aumentare della popolazione di alberi nella foresta, dei valori altalenanti che raggiungono il picco massimo con il **test 6**. Con questo test non solo otteniamo il valore più alto in termini di percentuale sull'accuratezza del set di test ma abbiamo anche il valore minimo relativo alla differenza di MSE, rispetto agli altri test.
Anche in questo caso il tempo di addestramento del modello è correlato alla popolazione della foresta:

- 100 -> 84 sec
- 120 -> 97 sec
- 140 -> 114 sec
- 160 -> 154 sec
- 180 -> 188 sec
- 200 -> 171 sec (tranne per questa eccezione dovuta alla macchina sul quale si eseguono i test)

Rispetto ai test del **1° blocco** i valori sono lievemente più bassi.

5° Blocco di test (Aumento popolazione, profondità: 21, c: Entropy, mss: 5)

Indice test	Popolazione massima	Profondità massima	Criterio	Min samples split	Acc Train	Acc Test	Differenza MSE	Over-fit
1	100	21	Entropy	5	97,59%	87,43%	1,40	NO
2	120	21	Entropy	5	97,64%	87,37%	1,42	NO
3	140	21	Entropy	5	97,61%	87,42%	1,41	NO
4	**160**	**21**	**Entropy**	**5**	**97,64%**	**87,53%**	**1,39**	**NO**
5	180	21	Entropy	5	97,68%	87,53%	1,39	SI
6	200	21	Entropy	5	97,70%	87,51%	1,40	SI
7	220	25	Entropy	5	97,84%	87,36%	1,41	SI

Per questi 7 test effettuati raccogliamo dei risultati quasi analoghi al blocco corrispondente con **criterio Gini**, tuttavia in questo caso otteniamo dei valori di accuratezza sul set di test molto più alti, ma anche un avvicinamento all'overfitting più veloce. Infatti in questi test effettuati notiamo che l'accuratezza sul set di test dopo aver raggiunto i migliori risultati nel **test 4** da una popolazione di 180 alberi in poi rende il modello meno efficiente, cosa che avveniva con il **2° Blocco** di test soltanto quando si sforavano i parametri limite relativi alla **profondità degli alberi** e al **numero di esemplari**.
I tempi di addestramento del modello sono indicativamente gli stessi.

6° Blocco di test (Aumento popolazione, profondità: 21, c: Entropy, mss: 10)

Indice test	Popolazione massima	Profondità massima	Criterio	Min samples split	Acc Train	Acc Test	Differenza MSE	Over-fit
1	100	21	Entropy	10	93,90%	86,80%	0,92	NO
2	120	21	Entropy	10	93,96%	86,91%	0,92	NO
3	140	21	Entropy	10	93,98%	87,00%	0,91	NO
4	**160**	**21**	**Entropy**	**10**	**94,00%**	**87,10%**	**0,89**	**NO**
5	180	21	Entropy	10	94,03%	86,98%	0,92	SI
6	200	21	Entropy	10	94,01%	86,94%	0,93	SI

Per questi 7 test effettuati raccogliamo dei risultati abbastanza differenti rispetto al **3° Blocco** di test effettuato con **criterio Gini**. Li avevamo un andamento altalenante dell' accuratezza sul set di test in questo caso invece, notiamo che all'aumentare della popolazione inizialmente l'accuratezza sul set di test cresce fino ad arrivare all' apice del suo valore al **test 4** nel quale registriamo anche la **miglior differenza di MSE**, dopo di ciò il valore inizia a scendere avviandosi verso l'overfitting.

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

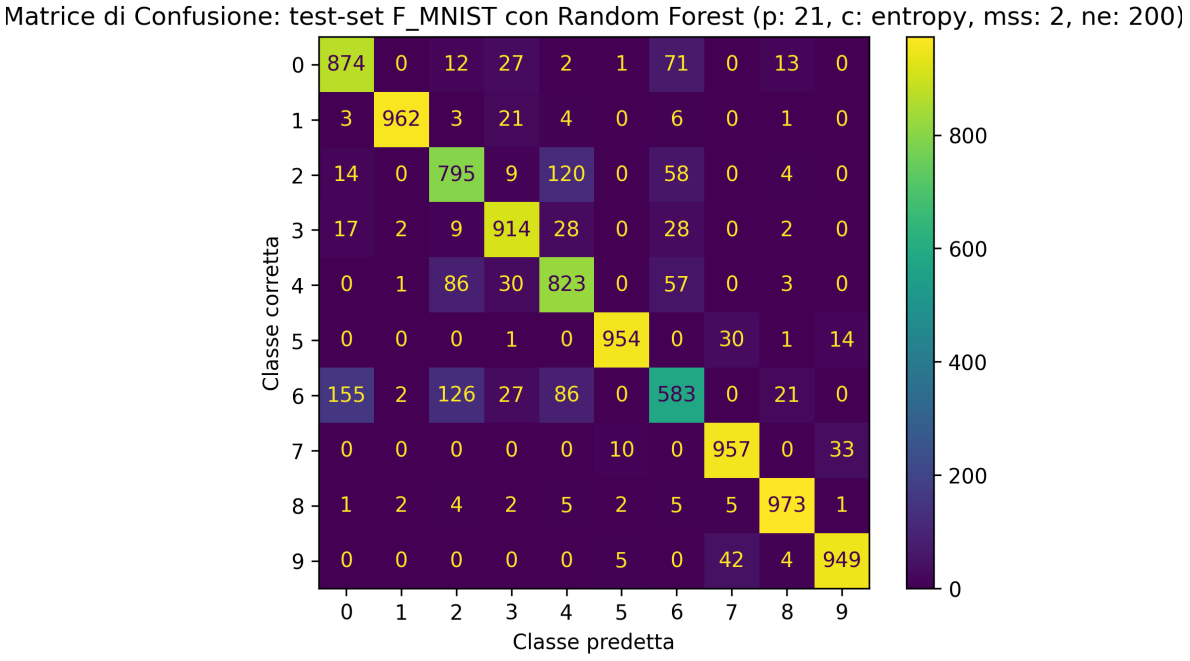
1) Blocco 4 - Test 6: Test del 2022-11-04 09:11:00
2) Blocco 1 - Test 6: Test del 2022-11-03 11:02:26
3) Blocco 5 - Test 4: Test del 2022-11-04 17:18:51

1) **Test del 2022-11-04 09:11:00 F_MNIST con Random Forest (profondità: 21, c: entropy, mss: 2, ne: 200):**

Report di classificazione

	precision	recall	f1-score	support
0	0.82	0.87	0.85	1000
1	0.99	0.96	0.98	1000
2	0.77	0.80	0.78	1000
3	0.89	0.91	0.90	1000
4	0.77	0.82	0.80	1000
5	0.98	0.95	0.97	1000
6	0.72	0.58	0.64	1000
7	0.93	0.96	0.94	1000
8	0.95	0.97	0.96	1000
9	0.95	0.95	0.95	1000
accuracy			0.88	10000
macro avg	0.88	0.88	0.88	10000
weighted avg	0.88	0.88	0.88	10000

Dal report di classificazione notiamo che il nostro classificatore ottiene i migliori risultati classificando la **classe 1-> Pantaloni** e i peggiori risultati di classificazione con la **classe 6 -> Camicia**



La matrice di confusione conferma ciò che viene mostrato nel report di classificazione la classe 1 che corrisponde ai **pantaloni** mostra su circa **1000** campioni **962** vengono classificati

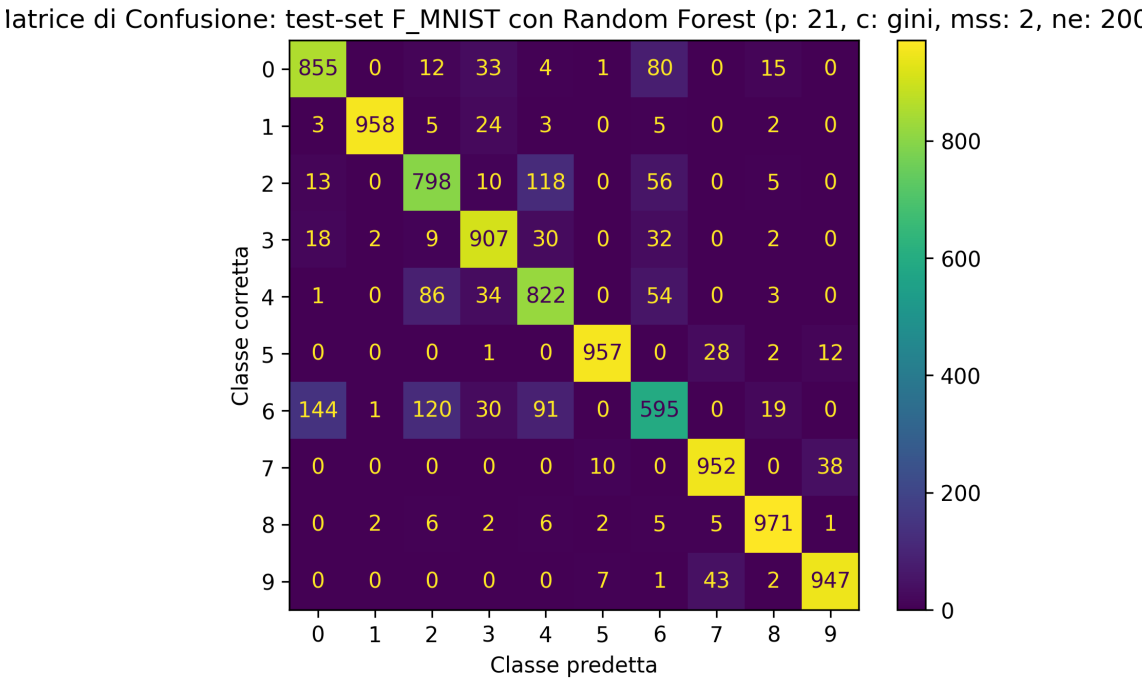
correttamente, da tenere conto anche dei risultati ottenuti dalla **classe 8** che corrisponde alla **borsa** dove su circa **1000** campioni **973** vengono classificati correttamente addirittura più della **classe 1** con l'unica differenza che la prima ottiene una migliore distribuzione degli errori. Analogamente la **classe 6** che corrisponde alle **camicie** è quella che ottiene i peggiori risultati infatti su **1000** campioni disponibili solo **583** vengono classificati correttamente. Risaltano all'occhio **155** campioni di **camice** classificati come **t-shirt/top,** i **126** campioni classificati come **maglioni**, e gli **86** come **cappotti**

2) **Test del 2022-11-03 11:02:26 F_MNIST con Random Forest (profondità: 21, c: gini, mss: 2, ne: 200)**

Report di classificazione

	precision	recall	f1-score	support
0	0.83	0.85	0.84	1000
1	0.99	0.96	0.98	1000
2	0.77	0.80	0.78	1000
3	0.87	0.91	0.89	1000
4	0.77	0.82	0.79	1000
5	0.98	0.96	0.97	1000
6	0.72	0.59	0.65	1000
7	0.93	0.95	0.94	1000
8	0.95	0.97	0.96	1000
9	0.95	0.95	0.95	1000
accuracy			0.88	10000
macro avg	0.88	0.88	0.87	10000
weighted avg	0.88	0.88	0.87	10000

Dal report di classificazione notiamo che il nostro classificatore ottiene i migliori risultati classificando la **classe 1-> Pantaloni** e i peggiori risultati di classificazione con la **classe 6 -> Camicia**



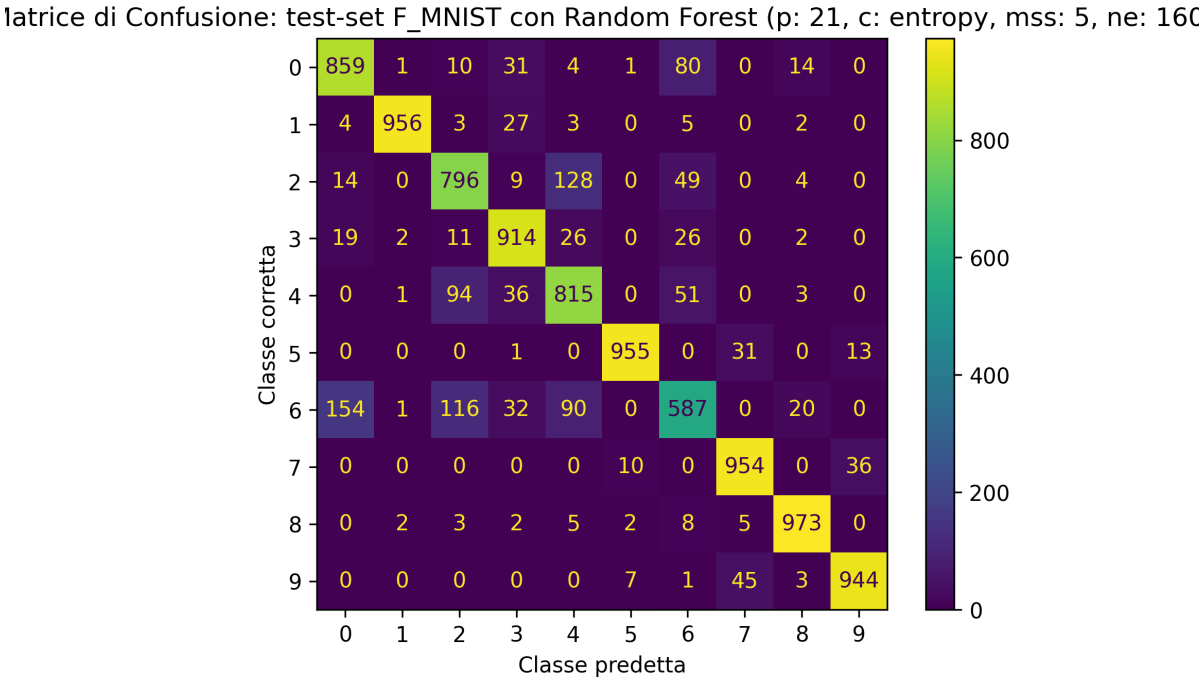
La matrice di confusione conferma ciò che viene mostrato nel report di classificazione la classe 1 che corrisponde ai **pantaloni** mostra su circa **1000** campioni **958** vengono classificati correttamente, da tenere conto anche dei risultati ottenuti dalla **classe 8** che corrisponde alla **borsa** dove su circa **1000** campioni **971** vengono classificati correttamente addirittura più della **classe 1** con l'unica differenza che la prima ottiene una migliore distribuzione degli errori. Analogamente la **classe 6** che corrisponde alle **camicie** è quella che ottiene i peggiori risultati infatti su **1000** campioni disponibili solo **595** vengono classificati correttamente. Risaltano all'occhio **144** campioni di **camice** classificati come **t-shirt/top,** i **120** campioni classificati come **maglioni**, e gli **91** come **cappotti**

3) **Test del 2022-11-04 17:18:51 F_MNIST con Random Forest (profondità: 21, c: entropy, mss: 5, ne: 160)**

Report di classificazione

	precision	recall	f1-score	support
0	0.82	0.86	0.84	1000
1	0.99	0.96	0.97	1000
2	0.77	0.80	0.78	1000
3	0.87	0.91	0.89	1000
4	0.76	0.81	0.79	1000
5	0.98	0.95	0.97	1000
6	0.73	0.59	0.65	1000
7	0.92	0.95	0.94	1000
8	0.95	0.97	0.96	1000
9	0.95	0.94	0.95	1000
accuracy			0.88	10000
macro avg	0.87	0.88	0.87	10000
weighted avg	0.87	0.88	0.87	10000

Dal report di classificazione notiamo che il nostro classificatore ottiene i migliori risultati classificando la **classe 1-> Pantaloni** e i peggiori risultati di classificazione con la **classe 6 -> Camicia**



La matrice di confusione conferma ciò che viene mostrato nel report di classificazione la classe 1 che corrisponde ai **pantaloni** mostra su circa **1000** campioni **956** vengono classificati correttamente, da tenere conto anche dei risultati ottenuti dalla **classe 8** che corrisponde alla **borsa** dove su circa **1000** campioni **973** vengono classificati correttamente addirittura più della **classe 1** con l'unica differenza che la prima ottiene una migliore distribuzione degli errori. Analogamente la **classe 6** che corrisponde alle **camicie** è quella che ottiene i peggiori risultati infatti su **1000** campioni disponibili solo **587** vengono classificati correttamente. Risaltano all'occhio **154** campioni di **camice** classificati come **t-shirt/top,** i **116** campioni classificati come **maglioni**, e gli **90** come **cappotti**