



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE

CORSO DI LAUREA MAGISTRALE IN
INGEGNERIA GESTIONALE

DATA ANALYTICS IN AMBITO INDUSTRIALE: STUDIO COMPARATO DI METODI E MODELLI DI REGRESSIONE SU DATASET REALI

Tesi di laurea METODI E MODELLI DI DATA ANALYTICS M

Relatore

Prof. Andrea Borghesi

Presentata da

Matteo Biagio Montemitro

Sessione Febbraio 2025

Anno Accademico 2023/2024

ABSTRACT

L'Intelligenza Artificiale al giorno d'oggi sta prendendo piede sempre più. In questa tesi viene proposto un elaborato inerente al Machine Learning, una branca dell'Intelligenza Artificiale, per la previsione della durata dei jobs eseguiti dal Supercomputer giapponese Fugaku, trattando quindi dati reali tramite tecniche di Data Analytics. Sarà proposta una panoramica dello stato dell'arte, nonché dei tipi di dati trattati, accentuando l'attenzione sull'alta variabilità degli stessi. Verranno utilizzate strategie per il preprocessing, volte ad addestrare i modelli in maniera ottimale, sfoltendo le colonne del dataset di partenza da 45 a 17. Saranno proposti e selezionati diversi modelli e metodi in modo da ottenere la previsione migliore in relazione agli indici delle performance dei modelli, in particolare riferimento al MAPE (Mean Absolute Percentage Error) ed al MAE (Mean Absolute Error). Nello specifico, i modelli di Machine Learning proposti saranno 18 per la previsione del target 'duration', nonché durata dei jobs. Saranno mostrate, passo dopo passo, le scelte di diverse metodologie ottenendo così modelli con indici accettabili per i tipi di datasets analizzati. Vi sarà un'analisi puntuale e generale degli outliers, e in che misura questi impattano sulla bontà degli indici. Verranno mostrati gli indici MAPE e MAE e il loro intervallo di confidenza al 95% in modo da avere un parametro affidabilistico, con focus sui modelli migliori: Random Forest Regressor e Bagging Regressor. Tale elaborato ha lo scopo quindi di proporre le tecniche di machine learning migliori per la previsione delle durate inerenti ai jobs eseguiti dal Supercomputer Fugaku.

Keywords: Machine Learning, Random Forest, Bagging, Intervalli di confidenza, Mean Absolute Percentage Error.

INDICE

ABSTRACT	1
1) INTRODUZIONE.....	5
2) BACKGROUND E INFORMAZIONI PRELIMINARI.....	7
2.1) UN SUPERCOMPUTER FUGAKU: COS'È E DOVE SI TROVA	8
2.2) MACHINE LEARNING E INTELLIGENZA ARTIFICIALE	10
2.3) ML: MODELLI UTILIZZATI E INDICI DELLE PERFORMANCE	12
2.4) INFORMAZIONI PRELIMINARI: LINGUAGGIO DI PROGRAMMAZIONE UTILIZZATO E DETTAGLI DEI DATASETS.....	15
3) ANALISI PRELIMINARE E PREPROCESSING DEI DATI	19
3.1) ESPLORAZIONE E VISUALIZZAZIONE DEI DATI	19
3.2) IDENTIFICAZIONE DELLE FEATURE SIGNIFICATIVE	24
3.2.1) TRAINING PRELIMINARE DEL DECISION TREE REGRESSOR	24
3.2.2) SELEZIONE DELLE FEATURE	26
4) SELEZIONE DEI MODELLI E DELLE METODOLOGIE	29
4.1) SELEZIONE DEI MODELLI	30
4.2) TRAINING DEI MODELLI CON E SENZA KFOLD-CROSS VALIDATION	32
4.3) ANALISI PRELIMINARE DEI MODELLI SU DF AMPLIATI	33
4.4) ANALISI DELLA METODOLOGIA SELECTKBEST	35
5) GENERALIZZAZIONE DEI MODELLI E ANALISI DELLE PERFORMANCE.....	38
5.1) GENERALIZZAZIONE E IDENTIFICAZIONE DEI MODELLI MIGLIORI: RANDOM FOREST E BAGGING.....	38
5.2) ANALISI DEGLI INDICI PER BAGGING E RANDOM FOREST	40
6) ANALISI DEGLI OUTLIERS E INTERVALLI DI CONFIDENZA DEGLI INDICI MAPE E MAE	43
6.1) STATISTICA DEL DF 21_03 CON INDICI INACCETTABILI COMPARATA AL 21_04.....	43
6.2) ANALISI DEGLI ERRORI PERCENTUALE (APE) ED ASSOLUTO (AE)	46
6.3) FILTRAGGIO E INTERVALLI DI CONFIDENZA	52
7) CONCLUSIONE E SVILUPPI FUTURI	62
SITOGRAFIA.....	64

1)INTRODUZIONE

Il seguente elaborato ha come scopo quello di analizzare datasets inerenti ai jobs eseguiti dal Supercomputer Fugaku e proporre dei modelli di Machine Learning che possano prevedere la durata dei singoli jobs. Per lo scheduling di un HPC è molto utile riuscire a prevedere la durata dei jobs in modo da poter organizzare al meglio l'ordine di esecuzione degli stessi, avendo quindi una stima di quanto dureranno le singole esecuzioni. Saranno analizzate diverse tecniche di apprendimento automatico (machine learning) e verranno comparati gli indici per la scelta del modello di machine learning migliore in relazione ai tipi di datasets analizzati. Vi è una prima fase in cui verrà analizzato il background, utile ai fini della comprensione del lavoro svolto nel cuore dell'elaborato, mostrando tecniche e metodi di machine learning, analizzando inoltre gli indici dei modelli, necessari per valutarne la bontà.

Successivamente, saranno analizzate le features e le statistiche dei dati, notando l'alta variabilità degli stessi. Verrà quindi proposto il modello Decision Tree Regressor addestrato su un dataset casuale contenente 2.000 righe per la predizione della feature 'duration', impostata come target per tutto il percorso dell'elaborato. Saranno poi analizzati i risultati per questo modello, volti alla scelta delle feature significative, tenendone 17 su 45, considerandole quindi per tutto l'elaborato dato l'impatto positivo sulle performance dei modelli.

Verranno mostrati i modelli da utilizzare per le predizioni, in aggiunta al Decision Tree Regressor, per un totale di 18. Sarà posta attenzione sulla metodologia Kfold-Cross Validation, utile alla generalizzazione dei modelli, che verrà però scartata per l'intero elaborato. Inerente alla generalizzazione e alla robustezza, saranno ampliati i datasets da 2.000 a 10.000 jobs mostrando i migliori 5 modelli in base all'indice r2score. Una volta ampliati i datasets verrà mostrata la metodologia SelectKBest, considerando però che impatta negativamente sulla performance dei modelli, scartandola quindi per l'intero elaborato.

Il lavoro di apprendimento automatico sarà quindi applicato a 10 datasets casuali, contenenti 10.000 jobs ognuno, provenienti da 6 datasets mensili, per un totale di 60, in modo da poter dare un ulteriore elemento di robustezza ai modelli valutando la bontà degli indici al variare dei datasets utilizzati. Saranno mostrati i grafici inerenti all'indice MAPE al variare dei 60 datasets, evidenziando la bontà dei modelli Random Forest Regressor e Bagging Regressor in confronto agli altri 16 modelli, quindi procedendo all'analisi utilizzando solo questi ultimi.

Infine, vi sarà un'importante analisi degli outliers nei valori predetti dai modelli, notando come senza di essi, considerati come lo 0,5% del test set, i risultati migliorino in maniera sostanziale. Verranno per ultimi analizzati gli intervalli di confidenza al 95% degli indici MAPE e MAE per il complesso dei datasets analizzati, in modo da tenere a mente un parametro affidabilistico.

2) BACKGROUND E INFORMAZIONI PRELIMINARI

In questa sezione vi sarà una breve **panoramica** su:

- **Fugaku**, il supercomputer che elabora i jobs da cui provengono i dati trattati nell'elaborato
- Cos'è l'**intelligenza artificiale** e il **machine learning**
- **Modelli di Machine learning utilizzati** con focus su Decision Tree Regressor, Random Forest Regressor, Bagging Regressor
- **Indici utilizzati**: R2score, MSE, RMSE, MAE, MAPE
- **Linguaggio di programmazione** utilizzato, librerie utilizzate, tipi di dati.

2.1) UN SUPERCOMPUTER FUGAKU: COS'È E DOVE SI TROVA

HPC (High-Performance Computing) è il termine utilizzato per i supercomputer e infrastrutture di calcolo per risolvere problemi complessi e dispendiosi a livello computazionale¹. Sono utili nel calcolo delle previsioni metereologiche, simulazioni industriali nonché nel campo dell'**intelligenza artificiale**. Il supercomputer **Fugaku**, sviluppato congiuntamente da RIKEN e Fujitsu dal 2014, è stato completato il 9 marzo 2021 ed è situato a Kobe, presso il RIKEN Center for Computational Science, in Giappone.



Fig. 1: Supercomputer Fugaku sviluppato da RIKEN e Fujitsu.

Fugaku è al primo posto nella classifica mondiale HPCG (High Performance Conjugate Gradient) e Graph 500, possedendo una potenza di calcolo di 547 petaflops/s in modalità boost, con processori di tipo ARM A64FX per combinare prestazioni elevate ed efficienza energetica ottimale. Al momento ossiede un numero di nodi pari a 158.976².

¹ <https://www.ibm.com/it-it/topics/hpc>

² <https://www.fujitsu.com/global/about/innovation/fugaku/>



Fig. 2: Uno dei 158.976 nodi del Supercomputer Fugaku.

L'elaborato è volto alla manipolazione dei dati appartenenti alle check lists mensili dei jobs elaborati dal supercomputer Fugaku, in modo da poter prevedere la durata di questi ultimi tramite modelli di Machine Learning.

2.2) MACHINE LEARNING E INTELLIGENZA ARTIFICIALE

L'**intelligenza artificiale** (AI) è un settore scientifico volto allo sviluppo di macchine in grado di ragionare, imparare e agire in un modo quanto più simile all'intelligenza umana. Prevede l'analisi di dati che l'uomo non è in grado di elaborare.

L'AI è un campo ampio che **comprende molte materie diverse**, tra cui informatica, analisi dei dati e statistiche, ingegneria hardware e software, linguistica, neuroscienze e persino filosofia e psicologia. ³



Fig. 3: “La Rivoluzione dell'intelligenza artificiale” (rif.: <https://www.primapagina.sif.it/article/1670/la-rivoluzione-dell-intelligenza-artificiale>)

Il **Machine Learning** (ML) è un metodo di analisi dati che automatizza la costruzione di modelli analitici. È una branca dell'Intelligenza Artificiale e si basa sull'idea che i sistemi possono imparare dai dati, identificare modelli autonomamente e prendere decisioni con un intervento umano ridotto al minimo.⁴ Il ML è presente in diversi metodi sviluppati nel XX secolo quali⁵: statistica computazionale, riconoscimento dei pattern, reti neurali artificiali, elaborazione di immagini, data mining, algoritmi adattivi ecc. L'obiettivo è quello di rendere le macchine in

³ <https://cloud.google.com/learn/what-is-artificial-intelligence?hl=it>

⁴ https://www.sas.com/it_it/insights/analytics/machine-learning.html#:~:text=Il%20machine%20learning%20%C3%A8%20un,intervento%20umano%20ridotto%20al%20minimo

⁵ https://it.wikipedia.org/wiki/Apprendimento_automatico

grado di apprendere da diversi dati, scoprendo pattern che legano diversi tipi di features. I metodi di ML sono molto utilizzati nel filtraggio della posta elettronica, categorizzando quindi le mail, ad esempio, per spam. Il ML propone metodi utilizzati in molte materie, quali medicina, per il riconoscimento delle malattie a partire dalla presenza/assenza di diversi sintomi. Ha indubbiamente preso piede anche nel mondo legato al riconoscimento vocale e visione artificiale. I metodi proposti dal Machine Learning (ML) possono essere suddivisi in tre principali macrocategorie: **supervisionati, non supervisionati e metodi basati sul rinforzo**. Questi ultimi sono algoritmi in cui un agente interagisce con un ambiente per apprendere a massimizzare una ricompensa cumulativa. L'agente, osservando lo stato attuale, compie un'azione e riceve un feedback sotto forma di ricompensa, adattando la propria strategia per migliorare le prestazioni future. Nei modelli supervisionati, le previsioni di determinate variabili (output) vengono confrontate con i valori reali, grazie alla disponibilità di dati etichettati che permettono di misurare il grado di errore tramite un test set. Al contrario, gli algoritmi non supervisionati non dispongono di etichette nei dati e cercano di individuare pattern nascosti senza un riferimento diretto ai valori di output. I problemi supervisionati si possono differenziare quindi in problemi di **classificazione** e di **regressione**. Problemi di classificazione implicano una categorizzazione degli output dei modelli in classi discrete, a differenza dei modelli di regressione per cui gli output sono valori continui. Nelle sezioni successive saranno impiegati metodi supervisionati basati su algoritmi di regressione, poiché il target considerato è rappresentato dalla durata dei jobs in secondi.

2.3) ML: MODELLI UTILIZZATI E INDICI DELLE PERFORMANCE

I modelli di Machine Learning utilizzati nell'elaborato sono diversi, 18 nello specifico; pertanto, sarà proposta una breve sintesi dei soli modelli di ML utili a rendere questo elaborato autoesplicativo.

- **Decision Tree Regressor** ⁶:

Suddivide iterativamente i dati in sottogruppi basati su condizioni che minimizzano l'errore (ad esempio, la riduzione della varianza). Ogni nodo dell'albero rappresenta una condizione, mentre le foglie finali contengono le predizioni medie del target, "duration" nello specifico. È semplice da interpretare, ma incline all'overfitting.

- **Random Forest Regressor** ⁷:

Il Random Forest Regressor crea una foresta di alberi decisionali addestrati su sottocampioni casuali del dataset. Ogni albero fa una previsione, e la media delle predizioni fornisce il risultato finale. Questa tecnica riduce overfitting e migliora la stabilità rispetto a singoli alberi decisionali.

- **Bagging Regressor** ⁸:

Il Bagging Regressor combina modelli indipendenti, addestrati su campioni casuali del dataset. Nell'elaborato verrà utilizzato il Bagging che utilizza il Random Forest come modello base. Le predizioni vengono mediate per ottenere il risultato finale. Questo metodo riduce varianza e migliora robustezza, funzionando bene con modelli deboli come alberi decisionali semplici.

È utile tenere a mente gli **indici delle performance** che danno indicazioni sulla bontà dei modelli utilizzati. Gli indici discussi in questo elaborato sono i seguenti:

- **R2score** (coefficiente di determinazione) ⁹ : è un indice che misura il legame tra la variabilità dei dati e la correttezza del modello statistico utilizzato. Intuitivamente, esso è legato alla frazione della varianza non spiegata dal modello. Valori di r2score prossimi ad 1 implicano buona predizione del modello. Nel caso in cui il valore di r2score fosse 1 (valore massimo), ciò implica che il modello prevede con una precisione del 100%.

⁶ https://saedsayad.com/decision_tree_reg.htm

⁷ <https://towardsdatascience.com/random-forest-regression-5f605132d19d>

⁸ https://inria.github.io/scikit-learn-mooc/python_scripts/ensemble_bagging.html

⁹ https://it.wikipedia.org/wiki/Coefficiente_di_determinazione

La definizione generale di tale indice è la seguente:

$$R^2 = 1 - \frac{RSS}{TSS}$$

Fig. 4: Formula del coefficiente di determinazione

laddove RSS è la devianza residua:

$$RSS = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Fig. 5: Formula della devianza residua

mentre TSS è la devianza totale:

$$TSS = \sum_{i=1}^n (y_i - \bar{y})^2$$

Fig. 6: Formula della devianza totale

dove:

\hat{y}_i sono i dati stimati dal modello,

y_i sono i dati osservati,

$\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ è la media dei dati osservati.

- **MSE** (Mean Squared Error) ¹⁰: indica la discrepanza quadratica media fra i valori dei dati osservati ed i valori dei dati stimati. La formula è la seguente:

$$MSE = \frac{\sum_{i=1}^n (x_i - \hat{x}_i)^2}{n}$$

Fig. 7: Formula del Mean Squared Error

¹⁰ https://en.wikipedia.org/wiki/Mean_squared_error

dove le due x sono rispettivamente i valori osservati e stimati, n invece è la grandezza del campione.

- **RMSE** (Root Mean Squared Error): è un ulteriore indice calcolato come la radice quadrata di MSE, utile quando gli errori hanno diversi ordini di grandezza.
- **MAE** (Mean Absolute Error) ¹¹: calcola l'errore assoluto tra i valori predetti e i valori reali. La formula è la seguente:

$$\text{MAE} = \frac{\sum_{i=1}^n |y_i - x_i|}{n} = \frac{\sum_{i=1}^n |e_i|}{n}.$$

Fig. 8: Formula del Mean Absolute Error

dove le y sono i valori predetti, x i valori reali, e gli errori assoluti puntuali e n è la grandezza del campione. Questo indice è utile quando i dati hanno diversi ordini di grandezza, poiché calcolare l'errore assoluto medio può essere più utile rispetto all'errore quadratico medio (MSE).

- **MAPE** (Mean Absolute Percentage Error) ¹²: tale indice, **molto discusso nell'elaborato**, propone la media degli errori percentuali. La formula è la seguente:

$$\text{MAPE} = 100 \frac{1}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|$$

Fig. 9: Formula del Mean Absolute Percentage Error

dove A_t è il valore reale, F_t è il valore predetto, t è l'indice del campione, n è la grandezza del campione e il parametro 100 è utile per ottenere il valore percentuale (%).

Gli indici sopra elencati saranno utilizzati per l'intero elaborato, ponendo particolare **attenzione sul MAPE e sull'MAE**. L'importanza di questi ultimi è dovuta alla natura dei dati trattati in questo elaborato, come verrà mostrato nel prossimo capitolo.

¹¹ https://en.wikipedia.org/wiki/Mean_absolute_error

¹² https://en.wikipedia.org/wiki/Mean_absolute_percentage_error

2.4) INFORMAZIONI PRELIMINARI: LINGUAGGIO DI PROGRAMMAZIONE UTILIZZATO E DETTAGLI DEI DATASETS

L'elaborato è stato sviluppato in ambiente di programmazione **Python** tramite l'utilizzo di **Jupyter Notebook** per poter spaccettare il programma e porre in run pezzi di codice anziché il codice completo. Il lavoro di tirocinio per elaborare questa tesi vertono sull'acquisizione di skills in ambito di programmazione con focus su Python e l'utilizzo di strategie di Machine Learning. Sono stati presi in analisi diversi datasets pubblicati sul sito <https://zenodo.org/records/11467483>, le cui **informazioni** sono inerenti ai **jobs** eseguiti dal supercomputer **Fugaku** dal mese di marzo 2021 al mese di aprile 2024 compresi. I file trattati sono in formato **.parquet**. Per l'intero elaborato i dati mensili verranno chiamati nelle seguenti forme: **df** interi, file **parquet**, file mensili, **df** mensili oppure tramite la forma **YY_MM**, eg. **21_04**, in riferimento all'anno e al mese del dataset (aprile 2021). Poiché ogni file **parquet** ha una dimensione media di **500MB** circa, saranno proposti dei sottoinsiemi di questi file **parquet**. Per la **prima parte dell'elaborato**, verranno considerati datasets contenenti **2.000 righe** anziché il complesso mensile dei jobs, per la **seconda parte** dell'elaborato invece saranno considerati datasets di **10.000 jobs**. I dati mensili sono organizzati in tabelle, letti tramite la seguente istruzione.

```
# Importazione libreria pandas, utile alla lettura dei file .parquet
import pandas as pd

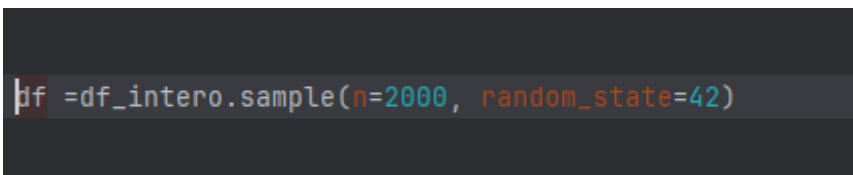
# Python Legge the 21_01.parquet file in formato dataframe
df = pd.read_parquet("21_01.parquet")
```

Fig. 10: Istruzione Python per la lettura dei file **.parquet** e la scrittura sulla variabile **df**

La **libreria** utilizzata per la lettura è **Pandas** che trasforma il file **.parquet** in **dataframe** (**df**), importata come in fig.10. Durante l'elaborato sarà proposto diverse volte il termine **dataframe** soprattutto nella sua forma abbreviata **df**; pertanto, si ritengono sinonimi i termini **dataframe**, **dataset** e **tabella**. Saranno necessarie diverse altre librerie, tra cui **Seaborn**, **Matplotlib**, **Scikit-learn**, **Numpy** ed altre, tutte utili al fine di poter manipolare i dati, analizzare statistiche, visualizzare i grafici, utilizzare metodi di machine learning (Scikit-learn) e molto altro.

I **file parquet disponibili** sono nello specifico **38**, da 21_03 a 24_04 compresi, per la prima parte dell'elaborato verrà considerato il df_intero 21_04 e nello specifico un df casuale con **random state 42**, utilizzando poi anche altri df come 21_03, 21_05, 22_09, 23_01, 24_02 per la parte successiva di questo elaborato.

Random state è un parametro usato nei modelli, in particolare in quelli di machine learning, in modo da poter ottenere una sequenza casuale ripetibile. Tramite questo parametro, che spesso è proposto nella forma *random_state=m*, è possibile ripetere l'esperimento ottenendo gli stessi risultati. Cambiando il valore di *m*, che è un numero intero, è possibile ottenere diversi df casuali. Pertanto, per esempio, nell'istruzione seguente verrà tenuto in considerazione, un df di 2.000 righe scelte casualmente con parametro *random_state = 42*.



```
df = df_intero.sample(n=2000, random_state=42)
```

Fig. 11: Istruzione utile alla creazione di df casuali a partire dal df_intero

Cambiando quindi il valore di *random_state* è possibile ottenere diversi df casuali a partire da un df_intero. Nell'elaborato sarà considerata la forma YY_MM_RS quando si fa riferimento ad un df casuale, considerando ad esempio 21_04_42 il df casuale legato al mese di aprile 2021 con random state 42.

È necessario porre attenzione sulla **grande quantità di dati** contenuti nei 38 file parquet: tutti i file mensili possiedono un numero di feature (colonne) pari a 45, e un numero di jobs (righe) che varia da 100.000 a 800.000 ognuno, per un totale di circa 22 milioni di righe, ognuna delle quali ha 45 informazioni (colonne) al suo interno, per un totale stimato di circa **1 MLD di informazioni**. La dimensione totale dei file mensili da marzo 2021 ad aprile 2024 è di circa 27 GB. Pertanto, nell'elaborato saranno proposti dei metodi e addestrati su sottoinsiemi di questi.

Le 45 feature in questione sono:

- **jid**: L'ID assegnato al job al momento dell'invio.
- **usr**: Il nome utente dell'utente che ha inviato il job.
- **jnam**: Il nome del job.
- **cnumr**: Numero di core richiesti per il job.

- **cnumat**: Numero di core allocati al job.
- **cnumut**: Numero di core utilizzati dal job.
- **nnumr**: Numero di nodi richiesti per il job.
- **adt**: Data e ora di arrivo (momento dell'invio).
- **qdt**: Tempo di inserimento nella coda dei jobs.
- **Schedsdt**: Orario di completamento della schedulazione.
- **deldt**: Momento di eliminazione del job.
- **ec**: Codice di uscita del job.
- **elpl**: Limite di tempo trascorso (il limite di tempo impostato per la durata del job).
- **sdt**: Data e ora di inizio.
- **edt**: Data e ora di fine.
- **nnuma**: Numero di nodi allocati al lavoro.
- **idle_time_ave**: Tempo medio di inattività del lavoro.
- **nnumu**: Numero di nodi utilizzati dal job.
- **perf1**: Numero di cicli di esecuzione.
- **perf2**: Operazioni a virgola fissa specifiche.
- **perf3**: Operazioni a virgola mobile scalari specifiche.
- **perf4**: Letture totali della memoria.
- **perf5**: Scritture totali nella memoria.
- **perf6**: Numero di cicli di sospensione.
- **mszl**: Limite di dimensione della memoria per il job.
- **pri**: Priorità.
- **econ**: Consumo energetico.
- **avgpcon**: Consumo energetico medio dei nodi.
- **minpcon**: Consumo energetico minimo dei nodi.
- **maxpcon**: Consumo energetico massimo dei nodi.
- **msza**: Memoria allocata.
- **mmszu**: Memoria utilizzata.
- **uctmut**: Tempo totale di utilizzo della CPU da parte dell'utente.
- **sctmut**: Tempo totale di utilizzo della CPU di sistema.
- **usctmut**: Tempo totale di utilizzo della CPU.

- **jobenv_req**: Ambiente di lavoro richiesto.
- **freq_req**: Frequenza del nodo richiesta.
- **freq_alloc**: Frequenza del nodo allocata.
- **flops**: Numero di operazioni in virgola mobile al secondo.
- **mbwidth**: Larghezza di banda della memoria.
- **opint**: Intensità operativa.
- **pclass**: Classe di prestazioni (compute-bound o memory-bound).
- **embedding**: Codifica dei dati sensibili tramite Sentence-BERT.
- **exit state**: Stato di uscita dell'esecuzione del job (failed o completed).
- **duration**: Durata dell'esecuzione del job in secondi.

Durante l'elaborato, per i 18 modelli di ML utilizzati, i datasets saranno divisi in train set e test set con un rapporto 80%-20%, quindi se un df possiede 10.000 righe, il training sarà fatto su 8.000 righe e il test sulle restanti 2.000, scelte in maniera casuale tramite l'istruzione seguente:

```
# Divisione in train e test set
X_train, X_test, y_train, y_test = train_test_split(*arrays: X, y, test_size=0.2, random_state=42)
```

Sarà utilizzata la funzione *train_test_split* della libreria Scikit-learn, per poter dividere il dataset in due parti secondo la porzione posta dal parametro *test_size*. Nel caso specifico *test_size* è la porzione percentuale del dataset da considerare come test set, ovvero 20%, valutando di conseguenza come train set l'80% restante del dataset su cui verranno addestrati i modelli.

3) ANALISI PRELIMINARE E PREPROCESSING DEI DATI

Questo capitolo è volto alla **visualizzazione dei dati** con focus in particolare sull'alta variabilità degli stessi. Verranno mostrate delle **tecniche** per il **preprocessing** dei dati, necessario affinché i metodi di machine learning possano imparare senza che del rumore bianco interferisca nell'apprendimento, eliminando le features tra loro altamente correlate.

3.1) ESPLORAZIONE E VISUALIZZAZIONE DEI DATI

È stato preso in esame il df_intero 21_04, contenente 116.977 jobs (righe) e 45 features (colonne). Per poter manipolare questo tipo di dati su un PC commerciale è opportuno ridurre il df_intero (dimensione di circa 500MB). Quindi è stata analizzata la feature 'avgpcon' (consumo medio al nodo) proveniente dal df casuale 2.000 righe (21_04_42), la sua distribuzione risulta mostrata in fig. 12.

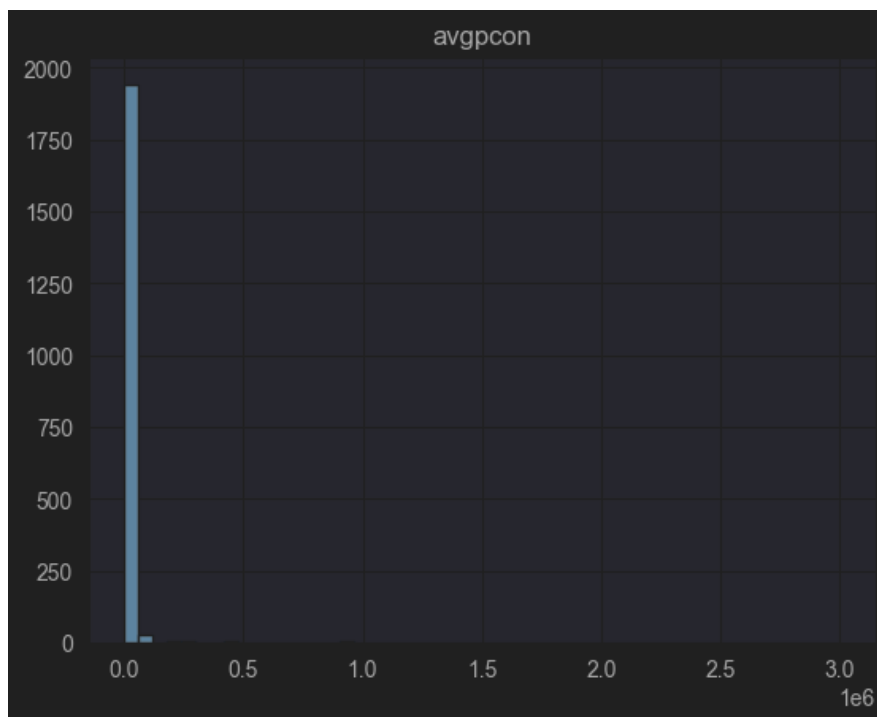


Fig. 12: Istogramma della feature 'avgpcons'

Considerando che il df di riferimento (21_04_42) contiene 2.000 jobs, la maggioranza di questi ha un valore di avgpcon molto basso (prossimo al valore 0.0×10^6) e solo per alcuni jobs i valori di avgpcon sono alti, arrivando fino a 3 milioni (3×10^6).

Tramite un boxplot si riesce a vedere l'alta variabilità di questa feature, come riportato nella fig.13.

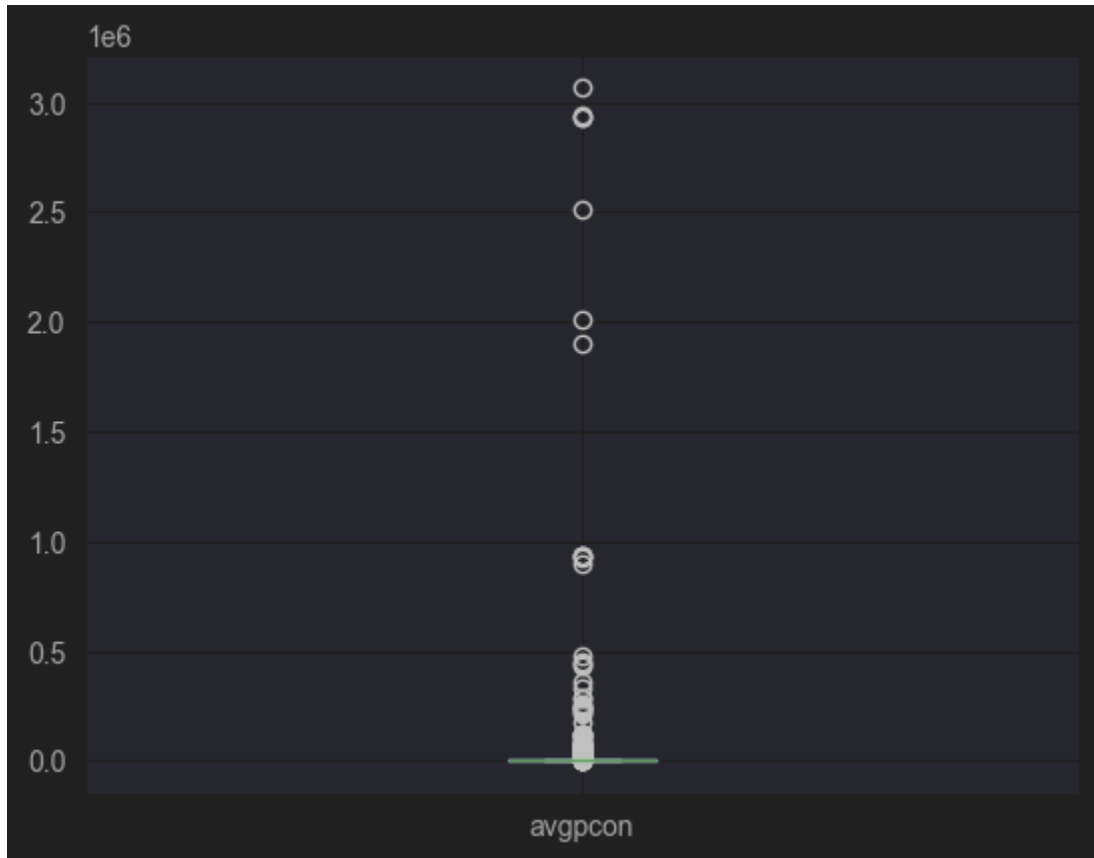


Fig. 13: Boxplot della statistica 'avgpcon', in linea verde la statistica

Un boxplot leggibile che contiene pochi outliers è fatto come in fig.14.

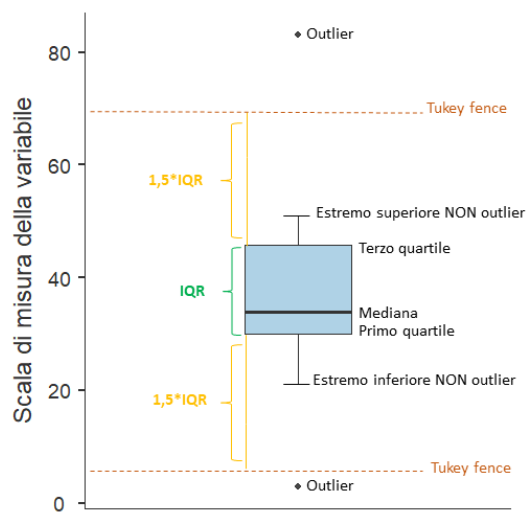


Fig. 14: Esempio di boxplot leggibile con due punti outliers

Ciò vuol dire che la statistica di avgpcon è estremamente concentrata nella linea verde (fig.13), con **tantissimi punti outliers**.

Si mostra l'analisi della feature **'duration'** (durata del job), nonché target dei modelli che verranno utilizzati, nella fig.15.

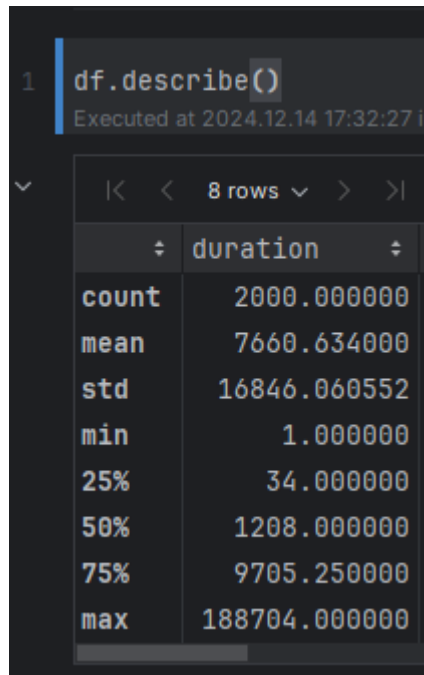


Fig. 15: descrizione statistica della feature 'duration'

È possibile notare che i valori della feature target variano da 1(min) a 188.704(max) per il df di 2.000 righe 21_04_42 (df casuale con random state 42 proveniente dal file mensile 21_04). In svariati dei 38 file mensili il valore massimo di 'duration' arriva ben oltre i 250.000 secondi. Si nota inoltre che il 75° percentile è 9.705 circa, mentre il massimo è posto a più di 188.704, sintomo dell'**altissima variabilità dei dati**. Tracciando un istogramma (fig.16) e un boxplot (fig.17) si nota, ancora una volta, che i dati sono molto concentrati nella prima parte del grafico e vi sono molti punti outliers.

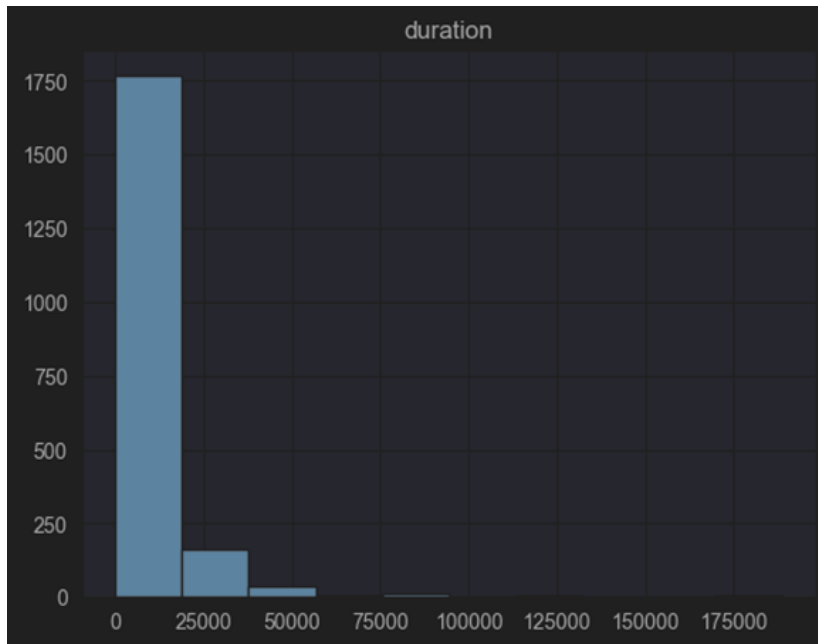


Fig. 16: Istogramma della feature 'duration'

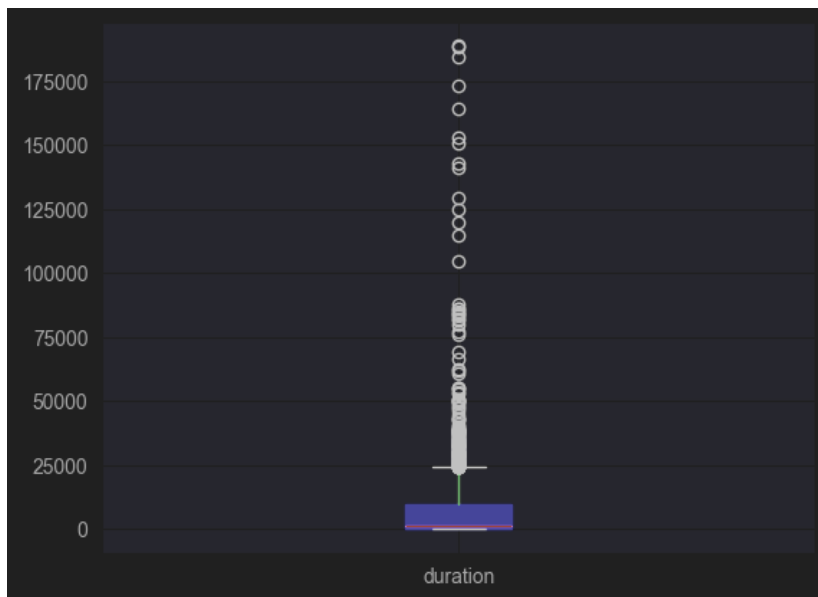


Fig. 17: Boxplot della feature 'duration'

Per vedere in maniera più efficace i dati delle features nominate in precedenza risulta necessario utilizzare la **funzione logaritmica** (fig.18 e 19).

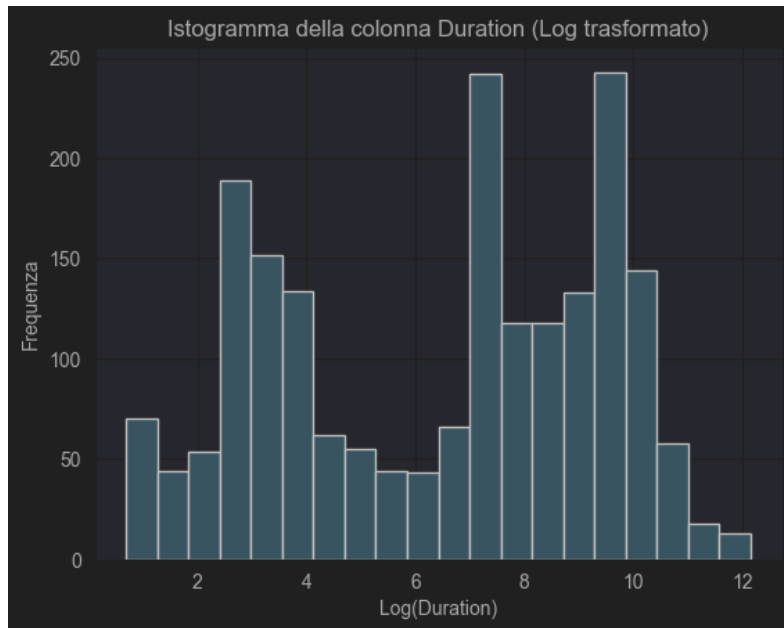


Fig. 18: Istogramma della Log(duration)

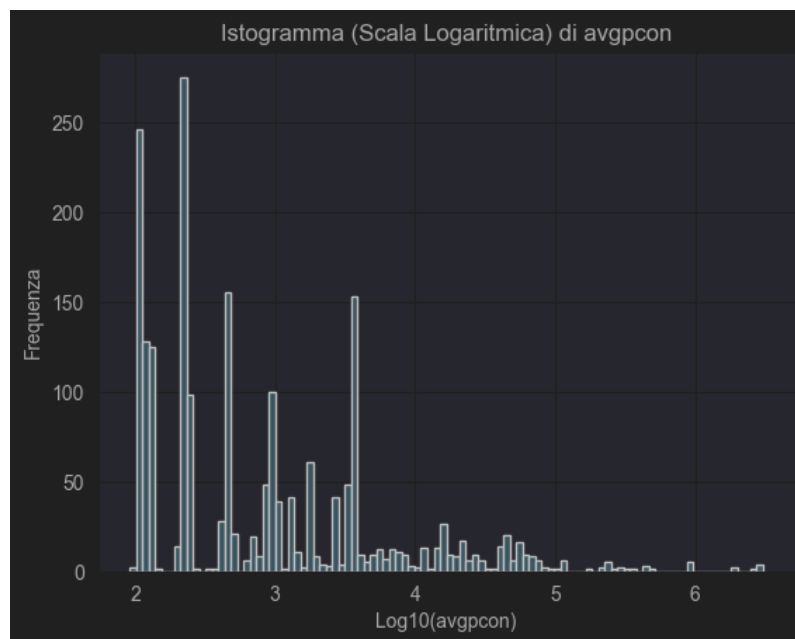


Fig. 19: Istogramma della Log(avgpcon)

Quindi i dati sono estremamente variabili, non solo per duration e avgpcon, ma per la maggior parte delle 45 features, di cui vengono omessi i grafici per semplicità, questo implica una **complessità** nel gestire tali informazioni e **nell'addestrare i modelli** affinché possano imparare da features che hanno una variabilità così accentuata.

3.2) IDENTIFICAZIONE DELLE FEATURE SIGNIFICATIVE

In questo paragrafo verranno mostrati i procedimenti volti a **soltire le features**. Si ricorda che i file mensili contengono in partenza 45 colonne. Le features verranno ridotte tenendo in conto solo di quelle che possiedono dati interi o numeri a virgola mobile (*int* e *float*).

Per applicare i modelli di regressione risulta necessario eliminare le feature di tipo *datetime*, tramite la funzione *.drop()* come mostrato in figura (fig.20).

```
df=df.drop(columns=['adt', 'qdt', 'schedsdt', 'deltdt', 'sdt', 'edt'])
```

Fig. 20: Istruzione per l'eliminazione delle colonne di tipo *datetime*

Successivamente sono state eliminate anche le colonne *'embedding'*, *'exit state'*, *'pclass'*, poiché non sono dati di tipo *float* o *int*. Per le colonne di tipo *string* come *'usr'* si è ritenuto opportuno in prima battuta tenere info di questo tipo trasformandole in tipo *int*: quindi, per alcune colonne di tipo *string*, prendendo in esempio la colonna *'usr'*, se il valore corrispondente risulta essere *usr_660407*, è stata eliminata la prima parte tenendo conto invece solo delle ultime sei cifre (*660407*), in modo da poter applicare i metodi di regressione.

In questa prima fase di preprocessing, le colonne risultano quindi essere 36, non più 45.

Si può quindi procedere al training preliminare di un modello di regressione.

3.2.1) TRAINING PRELIMINARE DEL DECISION TREE REGRESSOR

Come modello è stato preso in esame il **Decision Tree Regressor** (DTR). L'addestramento è stato svolto sul *df 21_04_42* contenete 2.000 jobs e 36 colonne utilizzando la funzione *StandardScaler()* di scikit-learn che permette la normalizzazione dei dati. Quest'ultima può essere utile per alcuni modelli, ma verrà evidenziato che nel dettaglio non aiuta il DTR a ridurre gli indici ritenuti significativi.

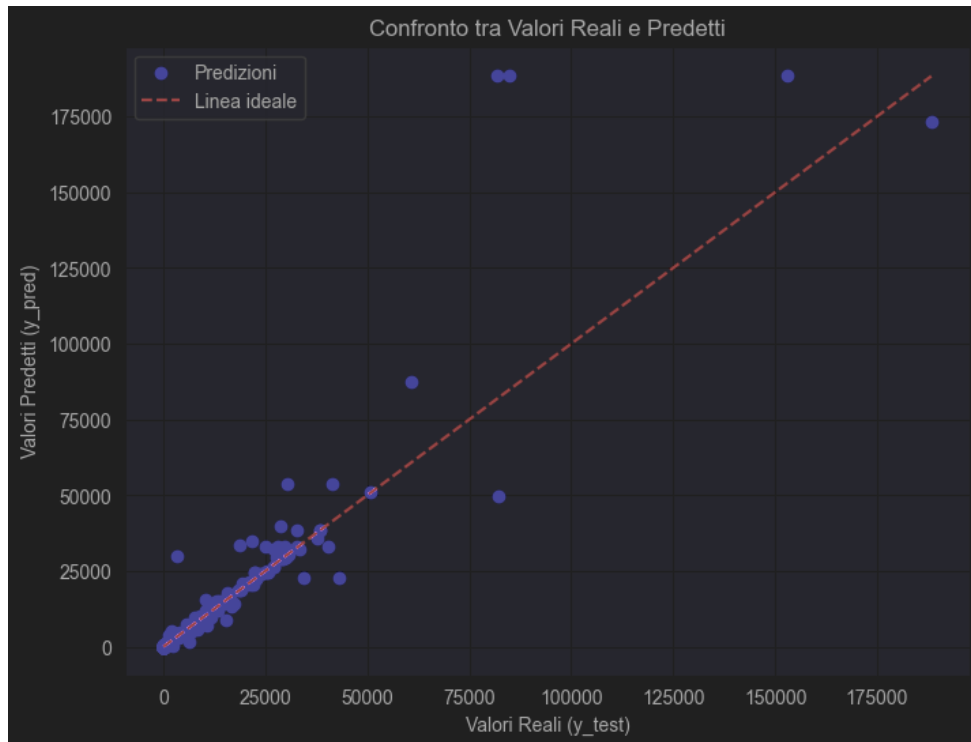


Fig. 21: Confronto tra valori predetti e valori reali del metodo Decision Tree Regressor con standardizzazione dei dati per il df 21_04_42 contenente 2.000 righe

Come mostrato in fig. 21 è possibile notare l'andamento dei punti y_{test} e y_{pred} per il DTR con standardizzazione dei dati.

Gli indici del DTR con e senza l'utilizzo di StandardScaler() sono mostrati nelle seguenti figure

```

Mean Squared Error: 70946460.83
R2 Score: 0.7404269602095908
Mean Absolute Error: 1492.67
MAPE: 18.38%

```

Fig. 22: Indici per il DTR con standardizzazione dei dati (36 colonne)

```

Mean Squared Error: 62380599.24
R2 Score: 0.7717670257368623
Mean Absolute Error: 1416.72
MAPE: 19.34%

```

Fig. 23: Indici per il DTR senza standardizzazione dei dati (36 colonne)

Si nota che l'utilizzo della standardizzazione migliora il valore di tutti gli indici in maniera significativa, eccetto che per il MAPE (media degli errori percentuali). Verrà posta particolare attenzione a questo indice poiché, considerando che l'obiettivo è quello effettuare previsioni sulla durata dei jobs, risulta necessario che il modello sbagli il meno possibile in riferimento all'errore percentuale. Per le prossime applicazioni i **dati verranno normalizzati**.

Il DTR è stato applicato anche a 6 colonne casuali anziché 36, ottenendo risultati molto peggiori; pertanto, è stato optato di continuare a sfortire le colonne tramite metodi più interessanti e basati sul buonsenso. Nel prossimo paragrafo verranno escluse diverse colonne in modo da ottenere risultati migliori.

3.2.2) SELEZIONE DELLE FEATURE

In questa sezione si approfondiranno le scelte che portano alla **riduzione delle features da 36 a 17** per l'esattezza.

In prima battuta sono state **escluse le colonne di tipo string** come *usr*, *jnam*, *jobenv_req*, *jid*, che in precedenza erano state prese in considerazione. Tale scelta è stata optata poiché, tenere un numero per indicare una classe potrebbe influire sul training del modello; infatti, risulta poco significativo tenere ad esempio 660407 a partire da *usr_660407* quando questo valore risulta un identificativo dello user nella realtà, ma il modello lo utilizza come un valore intero. Viene esclusa anche la colonna *pri* poiché poco utile ai nostri modelli di regressione considerando che contiene 127 come unico valore per il df 21_04_42 di 2.000 jobs preso in esame.

Tramite la funzione *.corr()* si possono visualizzare le relazioni tra le colonne tramite la **matrice delle correlazioni** in forma di tabella, come mostrato nella fig.24 in cui i valori più prossimi ad 1 hanno un colore più tendente al rosa.

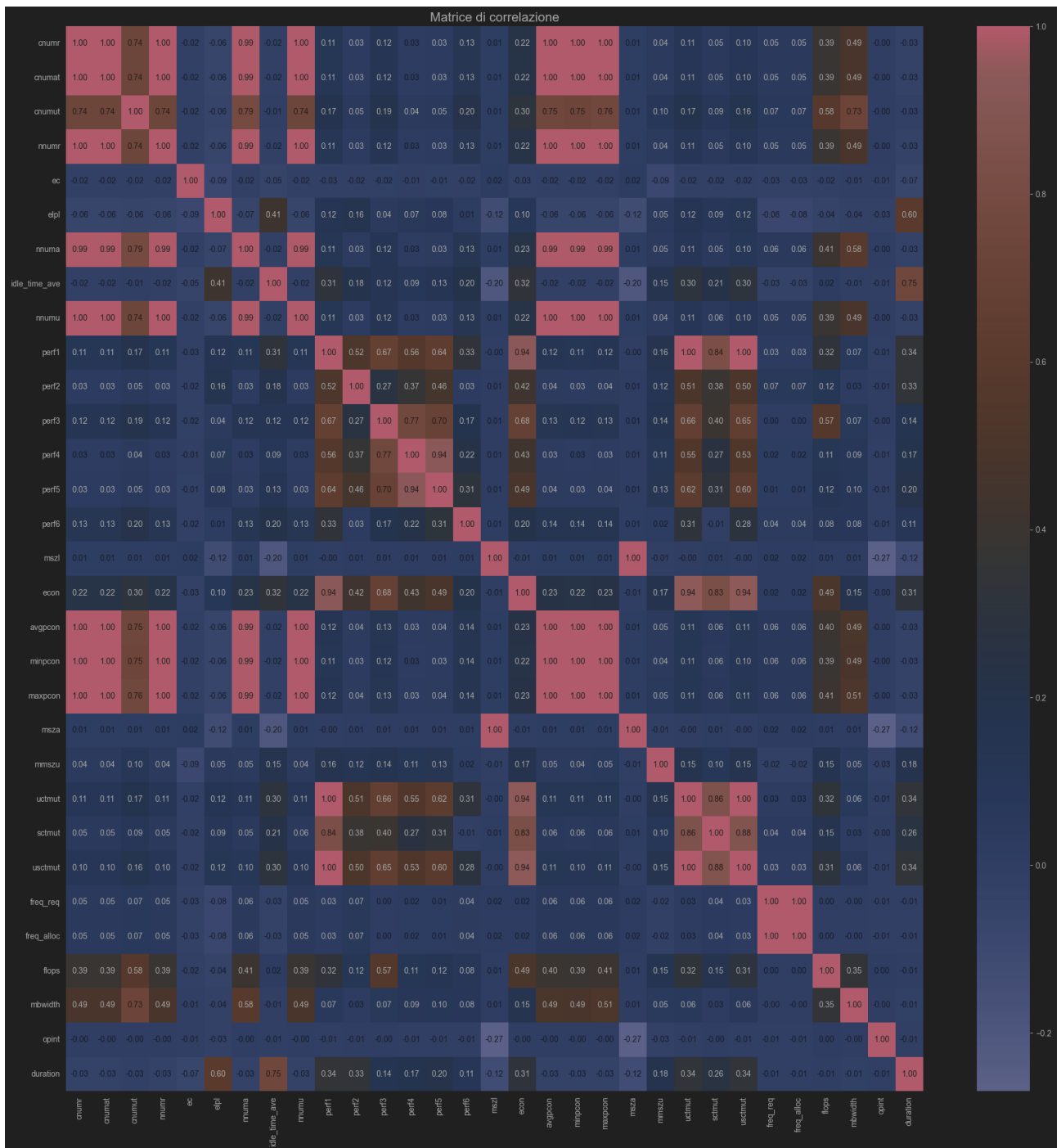


Fig. 24: Matrice di correlazione tra le features con scala a colori delle correlazioni.

Si è optato di **eliminare** tutte quelle **colonne** che sono **altamente correlate** tra loro, in modo da tenerne solo una.

Le colonne eliminate dal df sono racchiuse nella seguente tabella, tenendo a mente la fig.24, considerando un indice di correlazione $\geq 94\%$, passando così da **36 a 17 colonne**.

Feature eliminate	Feature Conservata	Valore di correlazione
cnumat, cnumut, nnumr nnuma, nnumu, minpcon, maxpcon, avgpcon	CNUMR	>0.99
econ, uctmut, usctmut	PERF1	>0.94
perf4	PERF5	>0.94
msza	MSZL	1
freq_req	FREQ_ALLOC	1

Dopo aver sfoltito quindi le features, applicando il DTR si ottengono i seguenti risultati

```

Mean Squared Error: 48872971.5525
R2 Score: 0.8211876161113839
Mean Absolute Error: 1206.0925
MAPE: 21.96%

```

Fig. 25: Indici del modello DTR per 17 colonne

Ciò significa che, escluso il MAPE, la scelta di sfoltire il dataset risulta una scelta che porta effettivamente ad un **miglioramento sostanziale degli indici**, portando R2 score da 0.77% a 0.82% circa; pertanto, nei prossimi capitoli verranno utilizzati datasets con 17 colonne.

4) SELEZIONE DEI MODELLI E DELLE METODOLOGIE

In questo capitolo verranno selezionati **18 modelli** da applicare, divisi in cinque categorie:

lineari, non lineari, ensemble, boosting avanzati e reti neurali.

Vedremo che i migliori modelli risultano essere gli ensemble e boosting avanzati.

Verranno prese in analisi le metodologie **Kfold-CrossValidation** e **SelectKBest**, entrambe però scartate per l'elaborato.

Kfold-CrossValidation¹³ è una tecnica usata per ridurre l'overfitting nei modelli e per renderli **generalizzabili**. Considerando che K è un numero intero, 10 nello specifico, viene preso un insieme di X samples, nello specifico 2.000, e successivamente viene diviso in K=10 folds. Per 10 volte vengono usati 10-1 folds per il training e il restante fold per il test. Ciò viene eseguito dieci volte, in modo che ogni fold possa essere usato da training e da test. Tenendo a mente che l'iterazione viene eseguita per 10 volte, può impattare significativamente sul tempo di addestramento dei modelli. Considerando inoltre la mole dei dati a disposizione, si è optato di non utilizzarlo perché come "tocco" di generalizzazione e robustezza, al posto di Kfold-CV, saranno considerate e mostrate altre strategie.

SelectKBest è una tecnica utilizzata per **sfoltire** le **colonne** del dataset tenendo solo quelle più significative, cosa che al momento già è stata fatta passando da 45 a 17 colonne. Saranno mostrati gli scarsi risultati derivanti da questa tecnica, che nello specifico utilizza 10 colonne anziché 17. Pertanto, non verrà considerata nel preprocessing dei dati da dare in pasto ai 18 modelli.

Per una maggiore **generalizzazione** e **robustezza**, come già accennato in precedenza, verrà ampliato il df su cui addestrare i modelli, passando da 2.000 samples a 10.000, verrà infine valutata la bontà dei modelli per diversi df, 6 per l'esattezza, notando i 5 migliori modelli per r2score e i 2 migliori per il MAPE.

¹³ https://scikit-learn.org/1.5/modules/cross_validation.html

4.1) SELEZIONE DEI MODELLI

Assodato il numero di colonne da utilizzare per addestrare i modelli, risulta necessario considerare, in relazione al tipo di dati che saranno manipolati, **modelli di diverse categorie**. Nello specifico saranno considerati modelli di tipo **lineare, non lineare, ensemble, boosting avanzati e reti neurali**.

Modelli Lineari:

Assumono una relazione diretta tra input (features) e output (target), rappresentabile con un'equazione lineare. Sono semplici, interpretabili e veloci da addestrare, ma limitati con dati complessi, nonché quelli trattati in questo elaborato. Funzionano bene in problemi con pattern lineari e pochi parametri rilevanti; infatti, per il df usato in questo elaborato non riescono a trovare delle relazioni tra input e output, data la natura complessa dei dati, tutt'altro che lineare.

Modelli Non Lineari¹⁴:

Catturano relazioni complesse tra variabili, non rappresentabili con una linea retta. Sono utili per dataset complessi ma possono essere più lenti e soggetti all'overfitting.

Ensemble¹⁵:

Combinano più predittori per migliorare precisione e robustezza. Bagging (es. Random Forest) riduce la varianza unendo predizioni indipendenti, mentre boosting (es. AdaBoost) corregge iterativamente errori. Questi approcci gestiscono dataset complessi, riducendo l'overfitting rispetto ai modelli singoli.

Boosting Avanzati¹⁶:

Tecniche di boosting avanzate come Gradient Boosting, XGBoost e LightGBM costruiscono modelli incrementali correggendo errori residui. Sono estremamente efficaci per problemi complessi e dataset rumorosi, con controlli avanzati di regolarizzazione. Necessitano di

¹⁴ <https://iartificial.blog/it/aprendizaje/diferencias-y-similitudes-entre-modelos-lineales-y-no-lineales-en-ml/#:~:text=nel%20Machine%20Learning%3F-,Definizione%20di%20modelli%20lineari,retta%20in%20uno%20spazio%20multidimensionale.>

¹⁵ <https://blog.alleantia.com/it/ensemble-modeling-come-migliorare-il-machine-learning>

¹⁶ <https://multinazionali.tech/algoritmi-gradient-boosting-guida-esempi-lavoro-nel-machine-learning>

ottimizzazione accurata per evitare overfitting e sono spesso usati in competizioni di data science.

Reti Neurali¹⁷:

Questi modelli simulano il cervello umano attraverso strati neuronali interconnessi. Apprendono pattern complessi usando tecniche come back-propagation. Utilizzate in deep learning, eccellono in visione artificiale e riconoscimento vocale. Possono richiedere risorse computazionali significative e grandi quantità di dati per allenarsi efficacemente.

Nello specifico, verranno utilizzati 18 modelli in totale, ovvero:

MODELLI LINEARI

- LINEAR REGRESSION
- RIDGE REGRESSION
- LASSO REGRESSION
- ELASTICNET REGRESSION
- STOCHASTIC GRADIENT DESCENT (SGD)
- KERNEL RIDGE
- PLS REGRESSION

MODELLI NON LINEARI

- SUPPORT VECTOR REGRESSION (SVR)
- K-NEAREST NEIGHBORS (KNN)
- GAUSSIAN PROCESS REGRESSOR
- DECISION TREE REGRESSOR

ENSAMBLE

- RANDOM FOREST REGRESSOR

¹⁷ <https://www.ibm.com/it-it/topics/neural-networks>

- GRADIENT BOOSTING REGRESSOR
- ADABOOST REGRESSOR
- BAGGING REGRESSOR

BOOSTING AVANZATI

- XGBOOST REGRESSOR
- LIGHTGBM REGRESSOR

RETI NEURALI

- MLP REGRESSOR

Si noterà come i migliori modelli risultano essere **Bagging Regressor** e **Random Forest Regressor** per l'indice MAPE, mentre in generale i modelli migliori per r2score risulteranno **ensemble** e **boosting avanzati**.

4.2) TRAINING DEI MODELLI CON E SENZA KFOLD-CROSS VALIDATION

Nella fig.26 è possibile notare i migliori 5 modelli per l'indice r2score **senza** l'utilizzo della tecnica **Kfold-CrossValidation**.

```
Top 5 modelli per R2:
LightGBM Regressor: R2 = 0.9768
Bagging Regressor: R2 = 0.9717
Gradient Boosting Regressor: R2 = 0.9709
Random Forest Regressor: R2 = 0.9705
XGBoost Regressor: R2 = 0.9657
```

Fig. 26: Migliori 5 modelli per l'indice r2score, senza l'utilizzo di Kfold-CrossValidation, addestrati sul df 21_04_42 di 2.000 jobs

Si nota già che i migliori modelli (r2score range: 96,57%-97,68%) sono **ensemble** e **boosting avanzati** (LightGBM, Bagging, Gradient Boosting, Random Forest, XGBoost), ciò è dovuto alla natura complessa dei dati.

Una volta **applicata** la tecnica **Kfold-CrossValidation**, si nota che i top 5 modelli rimangono gli stessi, in ordine diverso, portando però l'indice r2score a subire un decremento generale (range:96,45%-97,31%) (fig.27).

```
Top 5 modelli per R2 medio:  
Gradient Boosting Regressor: R2 medio = 0.9731  
LightGBM Regressor: R2 medio = 0.9710  
XGBoost Regressor: R2 medio = 0.9700  
Random Forest Regressor: R2 medio = 0.9688  
Bagging Regressor: R2 medio = 0.9645
```

Fig. 27: Migliori 5 modelli per l'indice r2score con l'utilizzo di Kfold-CrossValidation

Considerando che la tecnica applicata è utile a dare **generalità e robustezza** ai modelli, in questo elaborato non verrà presa in considerazione poiché vi è un'abbondanza di dati (38 df_interi che possiedono da 100.000 a 800.000 samples) su cui poter addestrare i modelli e tracciare l'andamento degli indici, considerata già quest'ultima come aggiunta alla robustezza e alla generalizzazione dei modelli.

4.3) ANALISI PRELIMINARE DEI MODELLI SU DF AMPLIATI

Fino a questo momento è stato usato il df 21_04_42 di 2.000 jobs x 17 colonne. Per ottenere tale dataset è stata necessaria la seguente istruzione:

```
df_intero = pd.read_parquet("21_04.parquet")  
df =df_intero.sample(n=2000, random_state=42)
```

Fig. 28: Istruzione per scrivere sulla variabile df (dataframe) il dataset con 2.000 righe scelte casualmente dal df_intero 21_04.

Per dare generalizzazione ai modelli è stato **ampliato il df** ad un numero di jobs pari a **10.000** tramite la seguente istruzione.

```
df_intero= pd.read_parquet("21_04.parquet")
df =df_intero.sample(n=10000, random_state=42)
```

Fig. 29: Istruzione per scrivere sulla variabile *df* (dataframe) il dataset con 10.000 righe scelte casualmente dal *df_intero* 21_04.

È stato quindi cambiando il valore di *n* posto pari a 10.000 (fig.29).

D'ora in poi verranno utilizzati solo df di 10.000 jobs.

Si ricorda che l'argomento *random_state=42* permette di scegliere sottoinsiemi casuali in maniera ripetibile, così che, ogni volta che viene riproposto il codice in fig.29, il *df* di riferimento rimanga sempre lo stesso, in modo che l'operazione sia deterministica. Saranno considerati quindi diversi *df* casuali legati al parametro *random_state*.

Una volta ampliato il *df*, sono stati addestrati i 18 modelli per il *df* 21_04_42, ovvero il dataset casuale (*random_state=42*) contenente le informazioni di 10.000 jobs per il mese di aprile 2021.

I risultati dei migliori 5 modelli per *r2score* sono mostrati in figura sottostante

Top 5 modelli per R ² :						
	Model	R ²	MSE	RMSE	MAE	MAPE
0	Gradient Boosting Regressor	0.941576	1.525613e+07	3905.909157	1088.963238	1146.093806
1	XGBoost Regressor	0.933617	1.733443e+07	4163.464200	574.295106	85.754872
2	LightGBM Regressor	0.931056	1.800297e+07	4242.990110	625.659986	212.178806
3	Random Forest Regressor	0.928632	1.863613e+07	4316.958070	593.365270	9.002291
4	Bagging Regressor	0.918523	2.127567e+07	4612.555200	651.776000	9.051436

Fig. 30: Classifica dei migliori 5 modelli per *r2score* in riferimento al *df* 21_04_42 di 10.000 righe e 17 colonne

Stessa cosa è stata fatta per *df* casuale con *random_state=41* (21_04_41).

	Model	R ²	MSE	RMSE	MAE	MAPE
0	LightGBM Regressor	0.948845	14294289.114127	3780.778903	721.740672	412.279916
1	Gradient Boosting Regressor	0.948658	14346443.186249	3787.669889	1157.948396	1439.870549
2	Random Forest Regressor	0.946612	14918323.118746	3862.424513	559.978725	9.316890
3	Bagging Regressor	0.942697	16012271.982790	4001.533704	595.754300	9.188914
4	XGBoost Regressor	0.935150	18121025.768869	4256.879816	592.594782	74.757046

Fig. 31: Classifica dei migliori 5 modelli per r2score in riferimento al df 21_04_41 di 10.000 righe e 17 colonne

Si nota che i top 5 modelli per r2score, anche se in ordine diverso, rimangono sempre gli stessi: Gradient Boosting, XGBoost, LightGBM, Random Forest, Bagging. Si nota che i migliori valori di MAPE li ottengono i modelli di Random Forest e Bagging (9% circa per entrambe).

Pertanto, è stata effettuata un'analisi per i df con random state = (43, 44, 45, 46) derivanti dal file mensile 21_04, notando che i **top 5 modelli** per r2score **rimangono invariati**, se non per l'ordine, mentre i **top 2 modelli per MAPE** risultano sempre **Random Forest e Bagging**.

Nel prossimo capitolo verrà mostrato nel dettaglio l'andamento dei seguenti indici al variare dei df.

4.4) ANALISI DELLA METODOLOGIA SELECTKBEST

È stata applicata la tecnica **SelectKBest**¹⁸ per i df con random state da 41 a 46 per il file parquet 21_04. Questa è una tecnica utilizzata per ridurre le colonne in modo da poter effettuare un training con meno features significative.

SelectKBest viene introdotta nel codice secondo la seguente istruzione:

```
selector = SelectKBest(score_func=f_regression, k=10)
```

Fig. 32: istruzione utile al fine di utilizzare SelectKBest

laddove *score_funcion* è l'argomento che specifica la funzione del punteggio da utilizzare. Vengono quindi tenute le *k=10* migliori colonne secondo il punteggio di *score_funcion* più alto, che nello specifico è **f_regression**¹⁹, nonché regressione lineare. I risultati per i 6 df citati

¹⁸ https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.SelectKBest.html

¹⁹ https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.f_regression.html

all'inizio del paragrafo (21_04 random state (41-46)), risultano essere **molto peggiori** rispetto all'utilizzo di 17 colonne. Vi è un esempio nella seguente figura (fig.33).

Model	R ²	MSE	RMSE	MAE	MAPE
K-Nearest Neighbors (KNN)	0.924854	19622684.790880	4429.749969	940.711400	330.203268
LightGBM Regressor	0.922908	20130649.132212	4486.719195	815.721775	238.268128
Gradient Boosting Regressor	0.918590	21258201.863027	4610.661760	1404.143322	785.505032
Random Forest Regressor	0.915381	22096167.751122	4700.656098	678.593410	19.450913
XGBoost Regressor	0.902080	25569387.016145	5056.618140	690.632703	79.824735
Bagging Regressor	0.884507	30158223.960510	5491.650386	783.660500	17.563832

Fig. 33: Classifica dei migliori 5 modelli per r2score in riferimento al df 21_04_42 di 10.000 righe e 10 colonne scelte da SelectKBest con score_function=f_regression

Paragonando fig.30 (andamento degli indici utilizzando 17 colonne) alla fig.33 (andamento degli indici utilizzando 10 colonne), si nota che i migliori modelli addestrati su datasets con **10 colonne**, usando come score_funicion=f_regression, ottengono indice **r2score peggiore**; infatti, passa da un range di circa 0.92-0.94% a 0.90-0.92%, mentre il **MAPE** per Bagging e Random Forest subisce un **incremento di 8-10 punti percentuali**, passando da 9% a oltre 17% per entrambe i modelli.

Utilizzando invece come score_function la funzione **mutual_info_regression**²⁰ che misura la dipendenza non lineare tra le features, per i 6 df si notano risultati **lievemente peggiori** per tutti gli indici. Ne viene mostrato un esempio in fig.34.

Model	R ²	MSE	RMSE	MAE	MAPE
12 Gradient Boosting Regressor	0.956713	11303312.788244	3362.039974	1090.331419	803.831913
11 Random Forest Regressor	0.937189	16401642.948901	4049.894190	574.986720	9.517995
16 LightGBM Regressor	0.936160	16670205.723114	4082.916326	684.078096	191.056379
14 Bagging Regressor	0.928301	18722557.040610	4326.957019	648.232700	12.867747
15 XGBoost Regressor	0.924339	19756985.006480	4444.883014	579.055087	54.584225

Fig. 34: Classifica dei migliori 5 modelli per r2score in riferimento al df 21_04_42 di 10.000 righe e 10 colonne scelte da SelectKBest con score_function=mutual_info_regression

Confrontando la fig.34 con la fig.31, paragonando quindi gli indici ottenuti con 10 colonne (tramite l'utilizzo di **mutual_info_regression**) e con 17 colonne, si nota un peggioramento di pochi punti percentuali per r2score e un incremento del MAPE per Bagging e Random Forest rispettivamente di 0.5% e di 3.7% circa.

Negli altri 5 df non si nota un miglioramento in nessuno degli indici; pertanto, non verrà optato l'utilizzo di SelectKBest in alcun modo. Si procederà quindi con l'utilizzo delle 17 colonne,

²⁰ https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.mutual_info_regression.html

considerando che SelectKBest, tramite gli esperimenti eseguiti, non reca migliorie in alcun modo.

5) GENERALIZZAZIONE DEI MODELLI E ANALISI DELLE PERFORMANCE

Una volta scelti i modelli e le metodologie, risulta necessario monitorare **l'andamento degli indici** per i diversi file parquet, ovvero in diversi df_interi; pertanto, in questo capitolo verranno mostrati i grafici degli indici per diversi df. Verranno presi in esame 10 df casuali contenenti 10.000 jobs provenienti da 6 df mensili, ovvero: 21_03, 21_04, 21_05, 22_09, 23_01, 24_02 per un totale di 60 df casuali. Tale scelta è volta a dare **robustezza e generalizzazione** dei modelli.

Verrà mostrato secondo i grafici che i migliori due modelli secondo il MAPE risultano essere **Bagging e Random Forest**, questa volta in maniera generalizzata.

5.1) GENERALIZZAZIONE E IDENTIFICAZIONE DEI MODELLI MIGLIORI: RANDOM FOREST E BAGGING.

I modelli e le metodologie sono stati applicati ai **60 df casuali** citati nell'introduzione del capitolo (File parquet: 21_03, 21_04, 21_04, 22_09, 23_01, 24_02; Random State= da 41 a 50 compresi). Sono stati selezionati i file parquet di cui sopra per poter monitorare l'andamento degli indici, quindi dei modelli, e il loro andamento in relazione a diversi mesi dell'anno, per diversi anni. Quindi l'analisi è volta a visualizzare sotto forma di grafici il funzionamento dei modelli per diversi **jobs distanziati temporalmente** (marzo 2021, aprile 2021, maggio 2021, settembre 2022, gennaio 2023, febbraio 2024).

In prima battuta, è utile porre l'attenzione sull'indice MAPE, pertanto viene proposto il grafico di questo indice per i migliori modelli, per l'esattezza 9 sui 18 proposti (fig.35). Sono stati selezionati 9 modelli su 18 poiché, almeno una volta nei 60 df, ognuno di questi 9 ha sfiorato la classifica top 5 modelli per uno qualunque dei seguenti indici: MAE, MAPE, RMSE, r2score.

In ascissa vengono mostrati i valori ordinati di *file parquet+random state* nella forma YY_MM_RSXX, dove XX è il numero attribuito al parametro random_state, compreso tra 41 e 50, in modo da poter monitorare a prima vista l'andamento degli indici per i 6 file parquet mensili (zone da sinistra a destra: 21_03 prima zona, 21_04 seconda zona, 21_05 terza zona, 22_09 quarta zona, 23_01 quinta zona, 24_02 sesta e ultima zona).

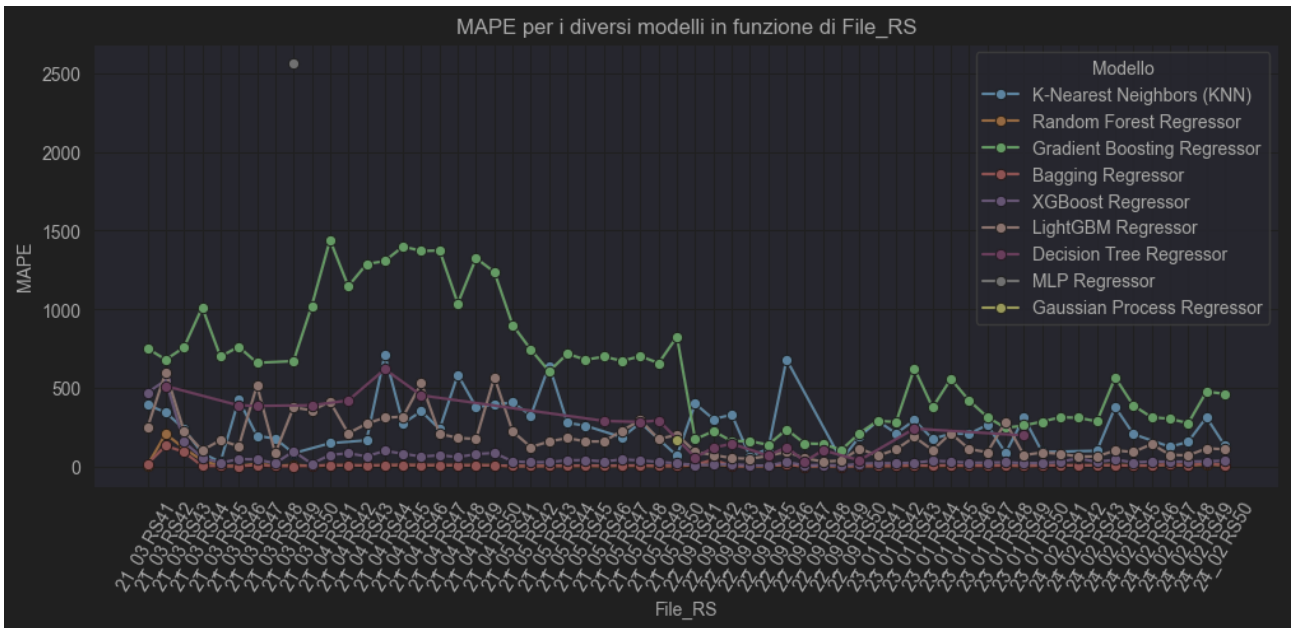


Fig. 35: Andamento del MAPE dei migliori modelli (9 modelli su 18) per i 60 df.

Si nota in prima battuta che i **migliori Modelli per il MAPE** risultano essere **Bagging e Random Forest**, ovvero quelli con valori più bassi.

Dopo un'analisi approfondita, emerge che in alcuni di questi 60 df, nello specifico per il file parquet 22_09 random state 41 e 42, risulta che **XGBoost** sfiora la classifica dei top2 modelli per il MAPE (fig.36).

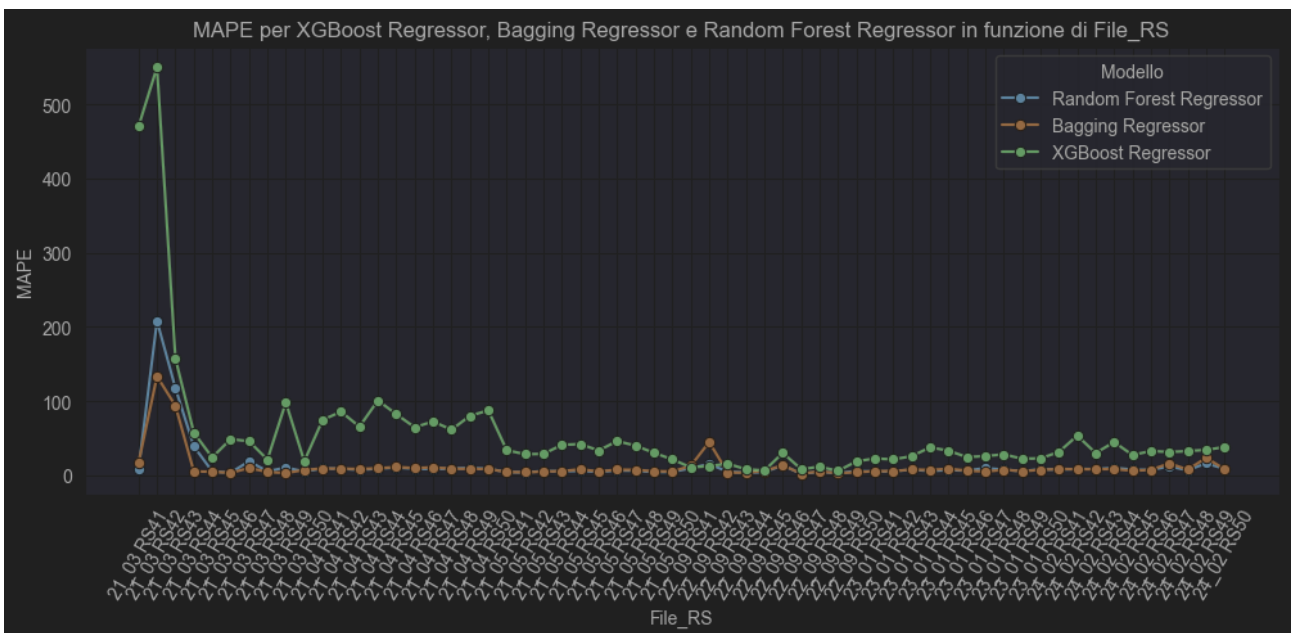


Fig. 36: Andamento del MAPE per i modelli Random Forest, Bagging e XGBoost.

Considerando che i valori di MAPE di XGBoost nei datasets 22_09 random state 41 e 42 sono molto simili ai valori di MAPE degli altri due modelli, **saranno considerati d'ora in poi solo i modelli Bagging Regressor e Random Forest Regressor**, ritenuti i migliori per indice MAPE, nonché il più significativo.

5.2) ANALISI DEGLI INDICI PER BAGGING E RANDOM FOREST

Assodato che i **migliori modelli per MAPE** sono **Bagging e Random Forest**, verranno monitorati gli **andamenti degli indici** r2score, RMSE, MAE e MAPE per i 60 df.

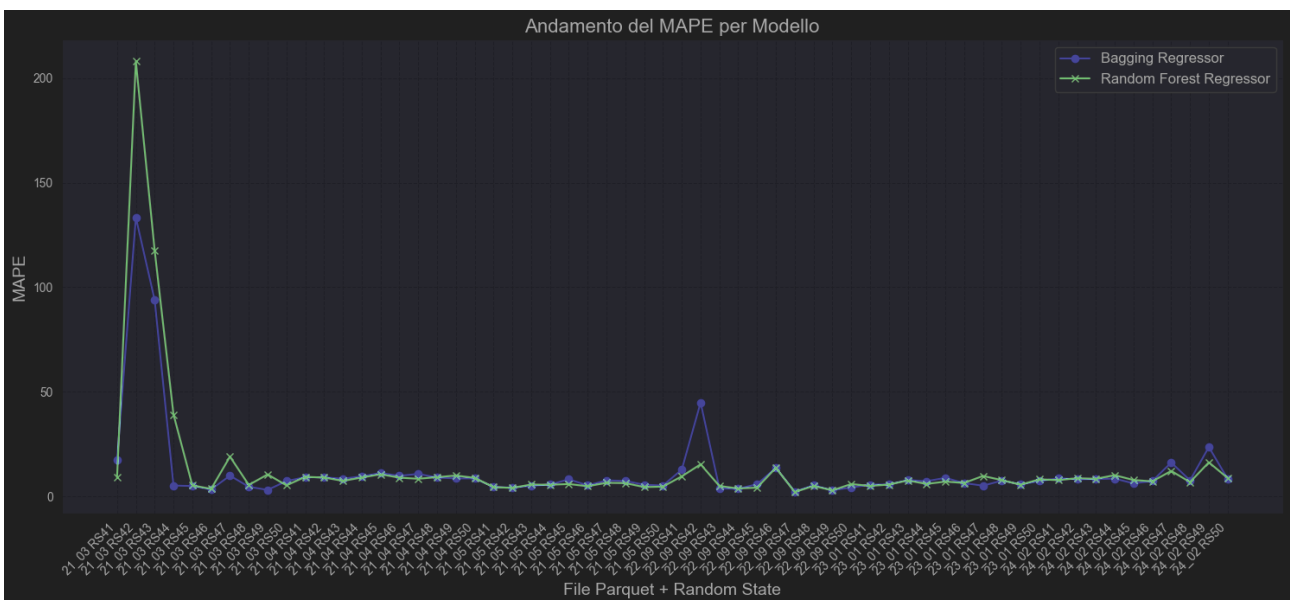


Fig. 37: Andamento del MAPE per i 60 df in riferimento a Bagging e Random Forest

Si nota come il MAPE (fig.37), nella sezione di sinistra, inerente ai df casuali provenienti da **21_03**, ha dei **valori estremamente elevati**. È possibile notare valori poco accettabili anche nei picchi di alcuni df casuali provenienti da 22_09 (zona centrale della fig.37), e 24_02 (zona a destra della fig.37). Al momento, ancora non è possibile definire quale tra Random Forest e Bagging Regressor sia il miglior modello, poiché i valori di MAPE di entrambi i modelli hanno andamenti molto simili, fatta eccezione dei picchi sopra citati.

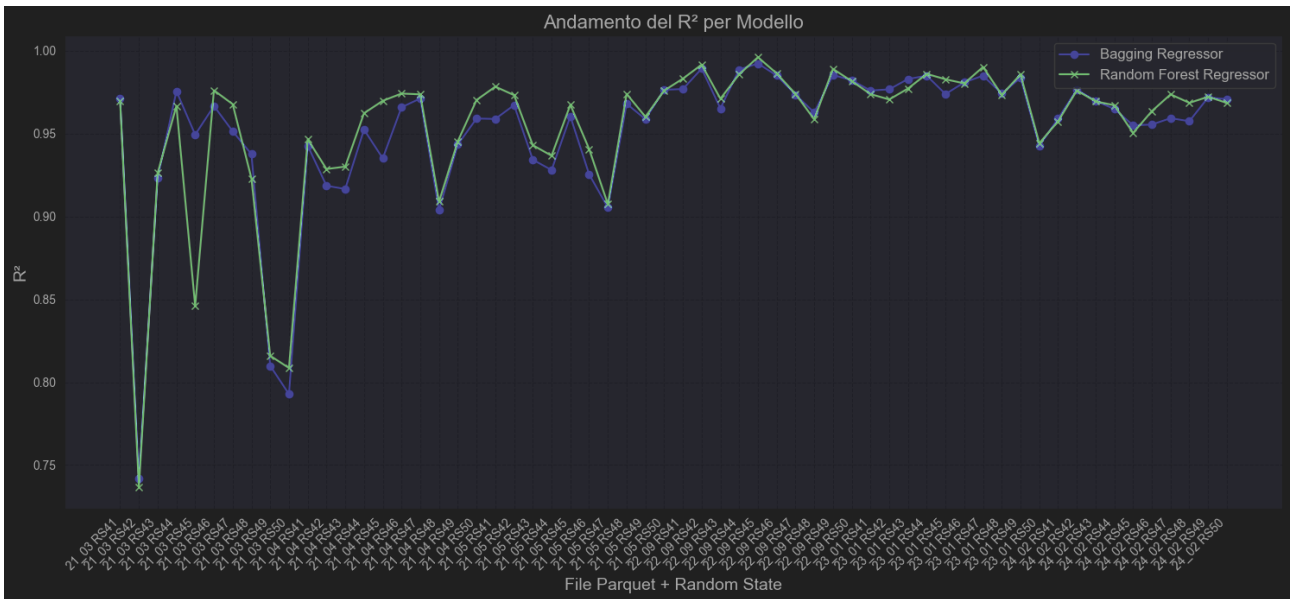


Fig. 38: Andamento dell'r2score per i 60 df in riferimento a Bagging e Random Forest

Per ciò che concerne l'indice **r2score**, si nota ancora una volta che, nella sezione di sinistra riferita al file mensile **21_03**, i valori hanno dei **valli importanti** (<90%) ritenuto **quindi non accettabile**. Si evince che ulteriori valli sono legate anche al file mensile 21_04 in cui il valore è prossimo al 90%, ritenuto quindi al limite dell'accettabilità. È possibile notare che per l'r2score un andamento del Random Forest leggermente migliore del Bagging Regressor escludendo le valli di inaccettabilità.

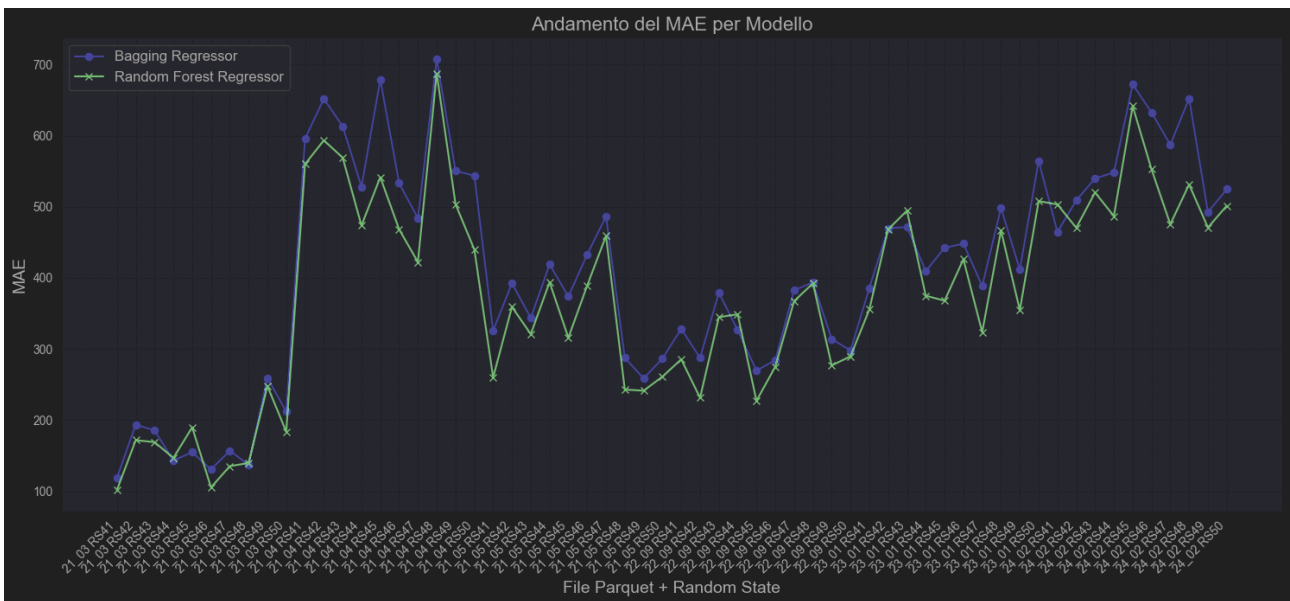


Fig. 39: Andamento del MAE per i 60 df in riferimento a Bagging e Random Forest

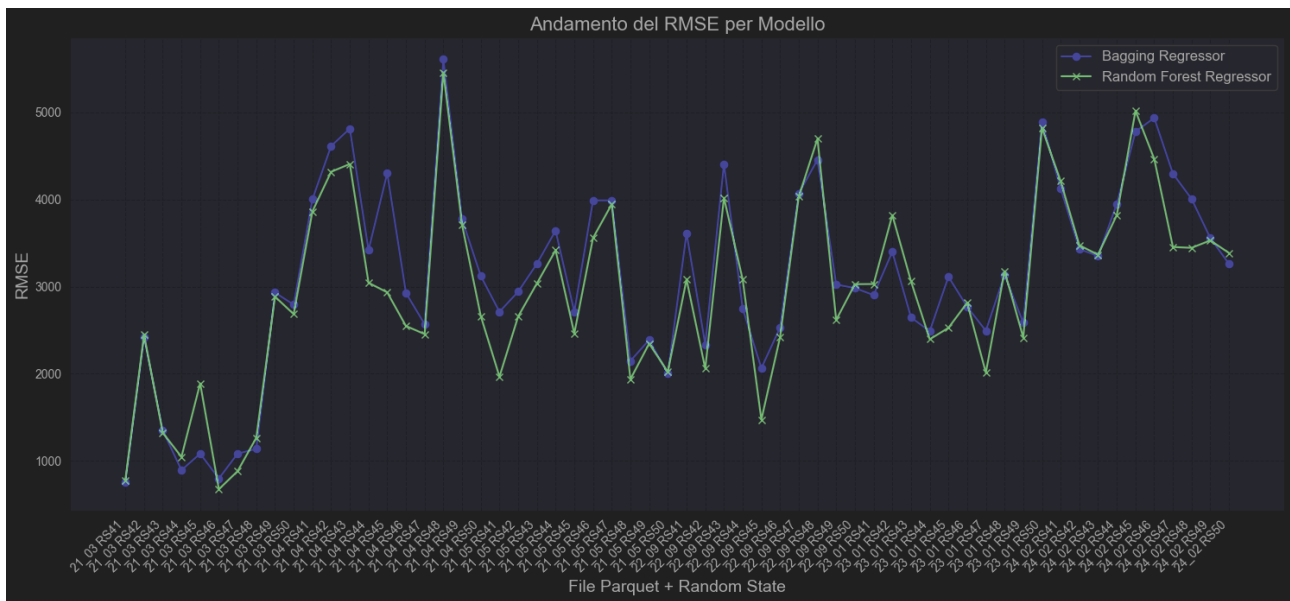


Fig. 40: Andamento dell'RMSE per i 60 df in riferimento a Bagging e Random Forest

Considerando la prima zona a sinistra degli ultimi grafici (21_03), in contrasto con MAPE e r2 score, si nota però che **MAE** e **RMSE** hanno valori **nettamente inferiori** (quindi migliori) per i df casuali provenienti dal file parquet 21_03 paragonati agli altri 5 file mensili. Si evince quindi che gli indici **MAE** e **RMSE** per la zona legata al **21_03** hanno valori migliori di 21_04, 21_05, 22_09, 23_01, 24_02. Si può notare che, per MAE a RMSE, il modello che possiede errori minori spesso risulta il **Random Forest**

Queste informazioni legate al file parquet 21_03 (zona di sinistra di fig. 37,38,39,40) sono in forte contrasto tra loro, perché, ci si aspetterebbe un peggioramento generale degli indici per lo stesso file mensile (21_03), cosa che non accade. Infatti, per **r2score** e **MAPE** i modelli addestrati sul **21_03** sono da **scartare**, mentre per **MAE** e **RMSE**, gli stessi modelli, in riferimento al medesimo file mensile, ottengono risultati **nettamente migliori** in confronto agli altri file mensili. Ciò induce a pensare che i risultati vengono considerati **inaccettabili** a causa di alcuni punti predetti sul test set, detti **outliers**. Verrà mostrato nel prossimo capitolo che il valore molto alto di alcuni indici dipende in gran parte da pochissimi punti predetti male dai modelli.

6) ANALISI DEGLI OUTLIERS E INTERVALLI DI CONFIDENZA DEGLI INDICI MAPE E MAE

In questo capitolo verranno analizzati nel dettaglio gli **outliers** e l'andamento comparato degli indici **MAE e MAPE** con e senza di essi. Sarà mostrata una prima parte di analisi evidenziando le **statistiche** del df intero 21_03, considerato inaccettabile a causa dei valori del MAPE e dell'r2score, paragonando tale file parquet al 21_04 che invece è considerato accettabile in base a tutti gli indici (r2score, MAPE, MAE, RMSE). Sarà inoltre mostrato **l'andamento degli indici escludendo dieci punti outliers delle predizioni**, corrispondenti allo 0,5% del test, notando un netto miglioramento delle performance dei modelli in base a MAPE e MAE. Infine, verranno mostrati gli indici **MAPE e MAE con gli intervalli di confidenza al 95%** comparati tra Bagging e Random Forest, evidenziando le ottime performance di quest'ultimo.

6.1) STATISTICA DEL DF 21_03 CON INDICI INACCETTABILI COMPARATA AL 21_04

Considerando l'andamento del **MAPE** (fig. 37) si nota che per il df 21_03, nello specifico i due df casuali con random state 42 e 43, i valori sono molto alti considerati **inaccettabili**, così come per il df casuale 22_09_42, poiché possiedono valori molto elevati (oltre di MAPE>50%). Per ciò che concerne **l'r2score** (fig.38), si nota che nel df 21_03, i valori di r2score sono **inaccettabili** poiché hanno punteggi < 90%, sintomo della cattiva predizione dei modelli.

Al contrario di quanto sopra, analizzando il **MAE** (fig.39) e **l'RMSE** (fig.40), risulta evidente che nella prima parte di grafici legata al df mensile 21_03 si hanno dei punteggi molto minori in confronto ai dati inerenti ai 5 file parquet restanti, evidenziando come lo stesso df_intero possieda i risultati migliori, quindi **accettabili**.

Queste due informazioni sono molto in contrasto perché secondo MAE e RMSE i modelli funzionano molto bene per il file parquet 21_03, mentre MAPE e r2score affermano l'esatto opposto. Pertanto, è stata eseguita un'analisi dettagliata della statistica dei dati inerenti al 21_03 per cui gli indici sono ritenuti inaccettabili, comparata alla statistica dei dati del file mensile 21_04 che invece ha indici accettabili.

Di seguito vediamo la descrizione delle statistiche dei dati del df 21_03_42 (df che comporta indici inaccettabili) contenente 10.000 righe.

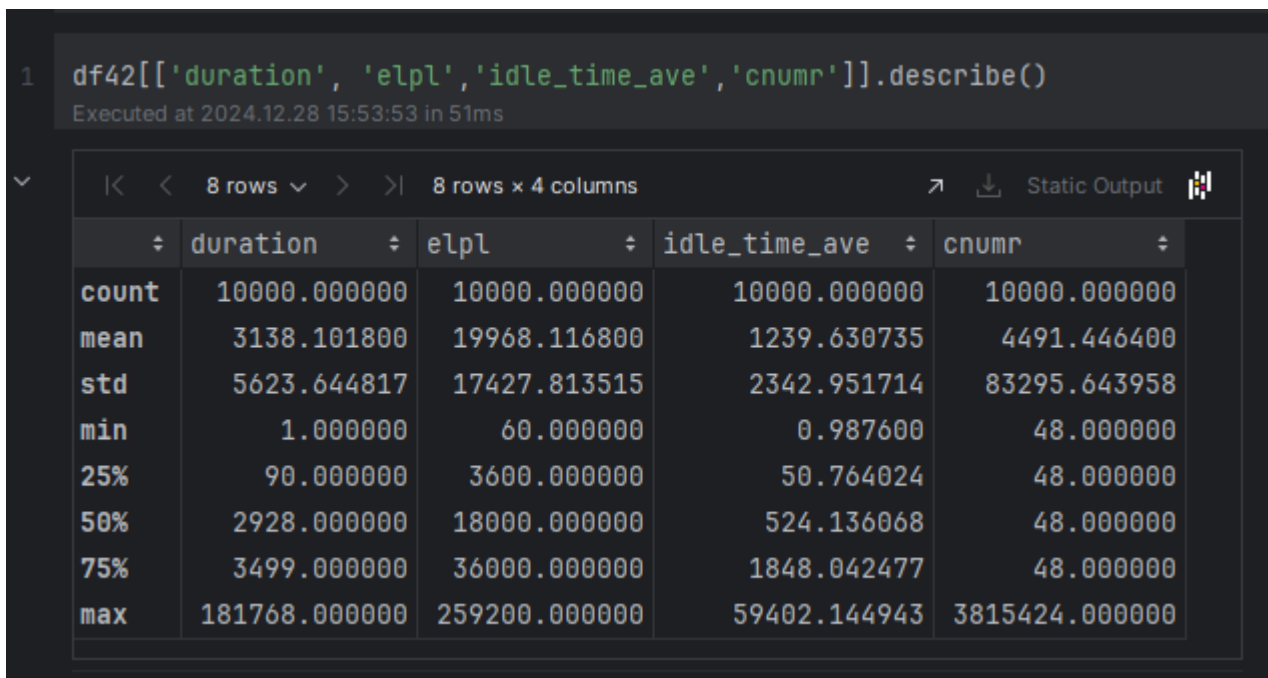


Fig. 41: Statistica delle features rilevanti (duration, elpl, idle_time_ave, cnumr) per il df 21_03_42 di 10.000 jobs (indici dei modelli inaccettabili)

Le statistiche precedenti sono state paragonate a quelle del 21_04_42, df per cui gli indici dei modelli sono ritenuti accettabili (fig.42).

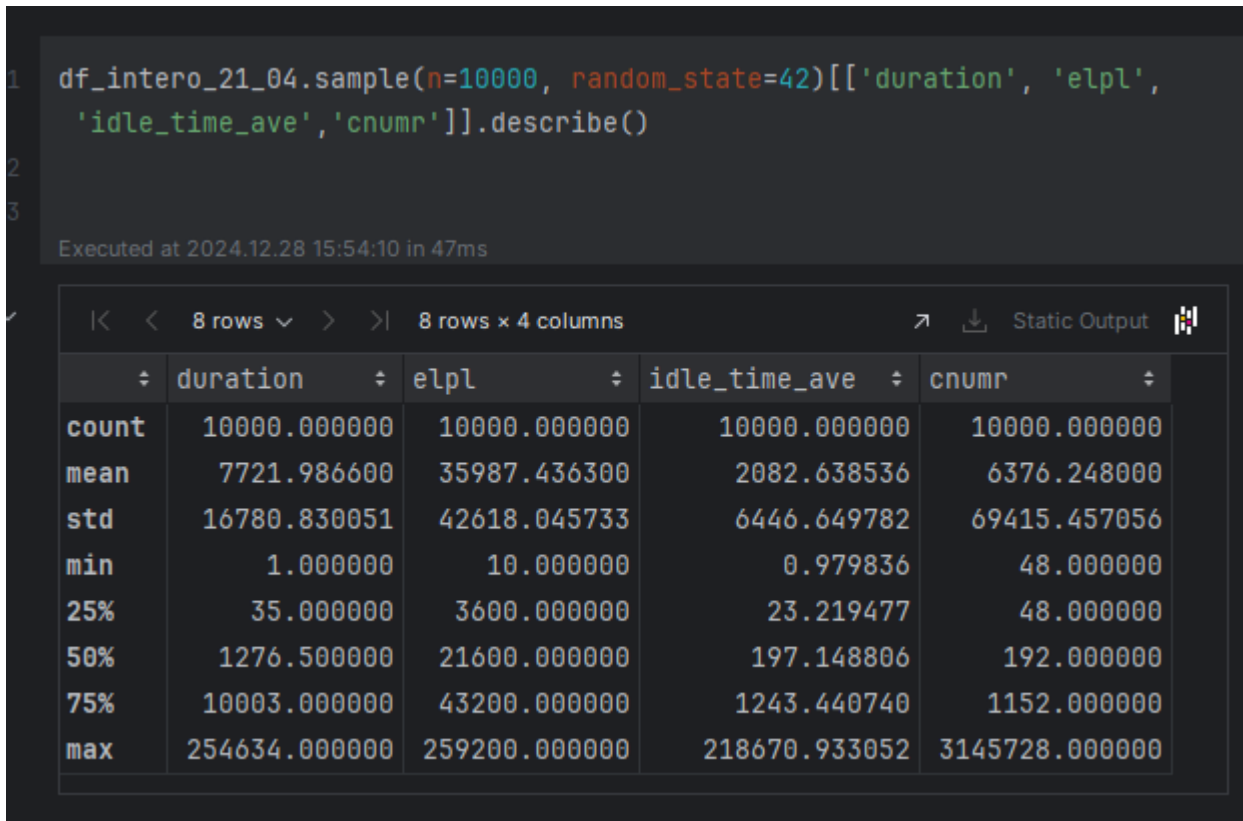


Fig. 42: Statistica delle features rilevanti (*duration*, *elpl*, *idle_time_ave*, *cnumr*) per il df 21_04_42 di 10.000 jobs (indici dei modelli accettabili)

Considerando che si ha a che fare con tabelle di 17 colonne, nel mostrare il paragone tra le statistiche di 21_03_42 e 21_04_42 sono state selezionate le **colonne** ritenute **più significative**: *duration*, *elpl*, *idle_time_ave*, *cnumr*.

Le statistiche del 21_03_42 paragonate al 21_04_42, per ciò che concerne la colonna *cnumr*, mostrano un'alta concentrazione dei valori di questa feature; infatti, posta l'attenzione sul 25°, 50° e 75° percentile, è evidente che tutti questi tre percentili hanno valore tutti pari a 48 (fig. 41), mentre in 21_04_42 le statistiche sono più sparse (25°percentile = 48, 50°percentile = 192, 75°percentile = 1152) (fig.42). Nonostante ciò, si nota una deviazione standard parecchio più alta in 21_03_42 (std di *cnumr* = 83.295 circa) in confronto a quella del df "accettabile" (std di *cnumr*=69.000 circa). È evidente, inoltre, che per tutte le colonne sopracitate, ad eccezione di *cnumr*, la deviazione standard (std) è molto più ristretta in 21_03_42 rispetto che in 21_04_42. Quest'analisi è stata fatta e comparata non solo per i df casuali, ma anche per i df interi (21_03, 21_04) e per tutte le 17 colonne, di cui si omettono le osservazioni tabulari. In generale si possiedono delle **statistiche**, per la maggior parte delle colonne, molto più **concentrate** in **21_03** che in 21_04, ciò forza i modelli a imparare da dati molto più concentrati e nel vedere

nuovi dati è molto più preciso per la maggior parte dei punti, ma **per pochissimi valori** del test set tende a compiere **errori estremamente accentuati** che rendono gli indici inaccettabili, come vedremo nel dettaglio nelle sezioni successive. Quindi nella prossima sezione sarà effettuata un'analisi ulteriore e più minuziosa degli errori dei modelli.

6.2) ANALISI DEGLI ERRORI PERCENTUALE (APE) ED ASSOLUTO (AE)

Considerando che MAE (Mean Absolute Error) e MAPE (Mean Absolute Percentage Error) sono rispettivamente indici degli errori **medio** assoluto e percentuale, è stato ritenuto opportuno calcolare anche il valore delle **deviazioni standard** per entrambi gli errori, così da avere ulteriori indici sulla variabilità di APE (Absolute Percentage Error) e AE (Absolute Error).

Una volta aggiunti gli indici **StdAPE** e **StdAE**, ovvero le rispettive deviazioni standard degli errori percentuale e assoluto, si è notato un alto valore di quest'ultimi, portando a categorizzarli come inaccettabili. Vi è un esempio della StdAPE in riferimento al Bagging in figura successiva.

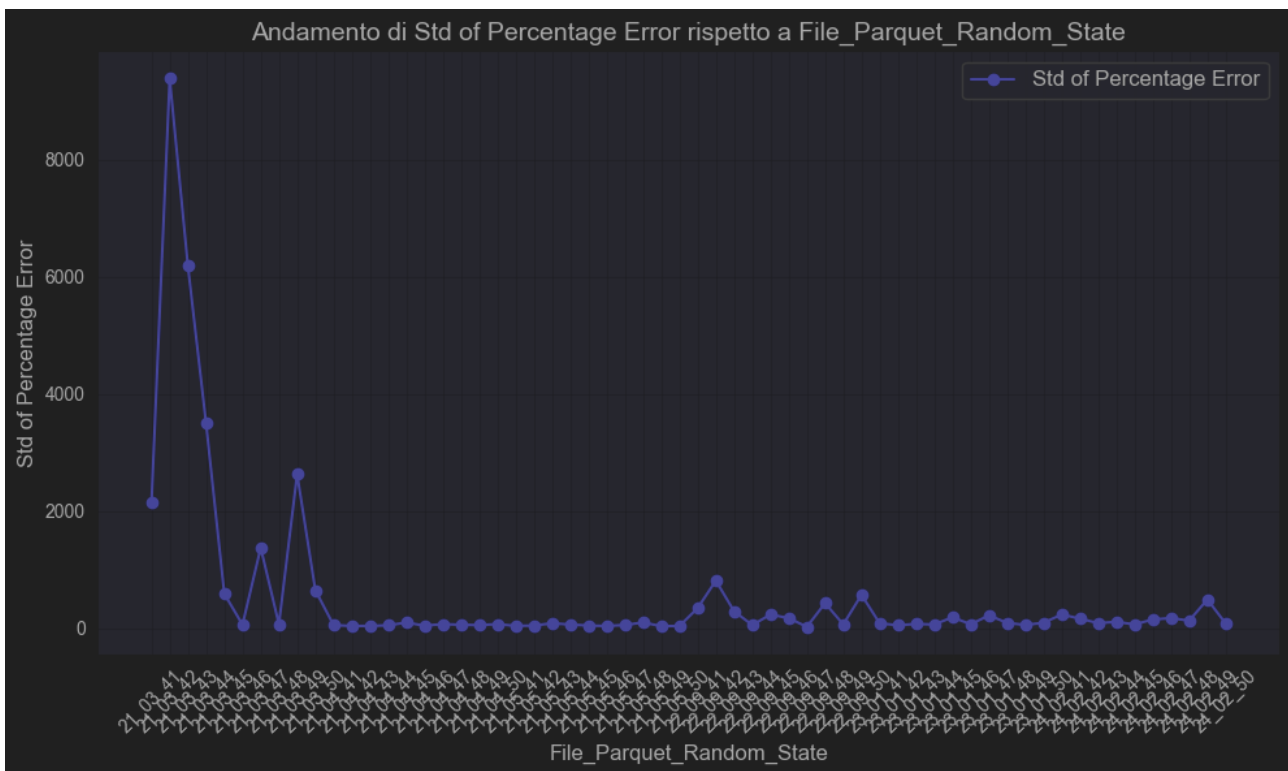


Fig. 43: Andamento di Std of Percentage Error per il modello Bagging Regressor

Affermare quindi che, per il df 21_03_42, l'errore percentuale abbia una media (MAPE) di oltre il 200% circa (fig.37) e una deviazione standard di oltre 9.000 (fig. 43) è assolutamente inaccettabile per un modello. Dalla fig.43 si nota inoltre che le deviazioni standard per l'errore percentuale hanno pochi **picchi** ma molto importanti, soprattutto in riferimento alla zona dei df casuali provenienti dal mese di marzo 2021 (**21_03**).

Ciò porta ad analizzare nel dettaglio la **quantità di punti**, sul totale del **test set**, che portano ad una condizione di **inaccettabilità** di tali indici. Una prima analisi grafica degli errori è stata fatta per il df 21_04_42 (che in precedenza è stato definito accettabile). I risultati sono i seguenti.

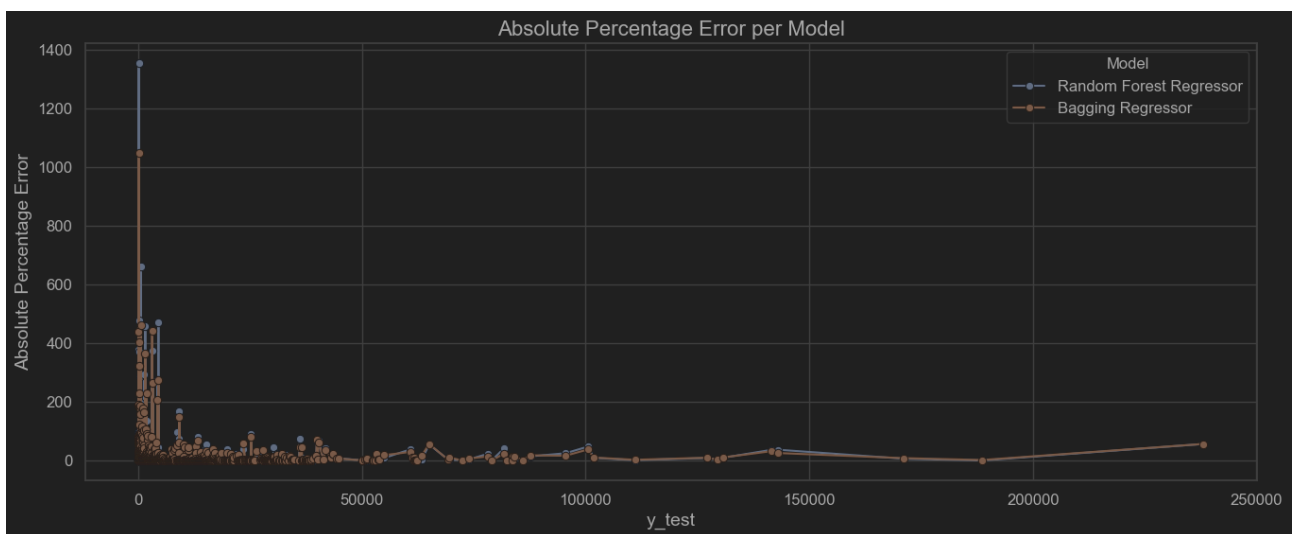


Fig. 44: Andamento dell'errore APE per il singolo df 21_04_42 contenente 10.000 jobs dei modelli Bagging e Random Forest

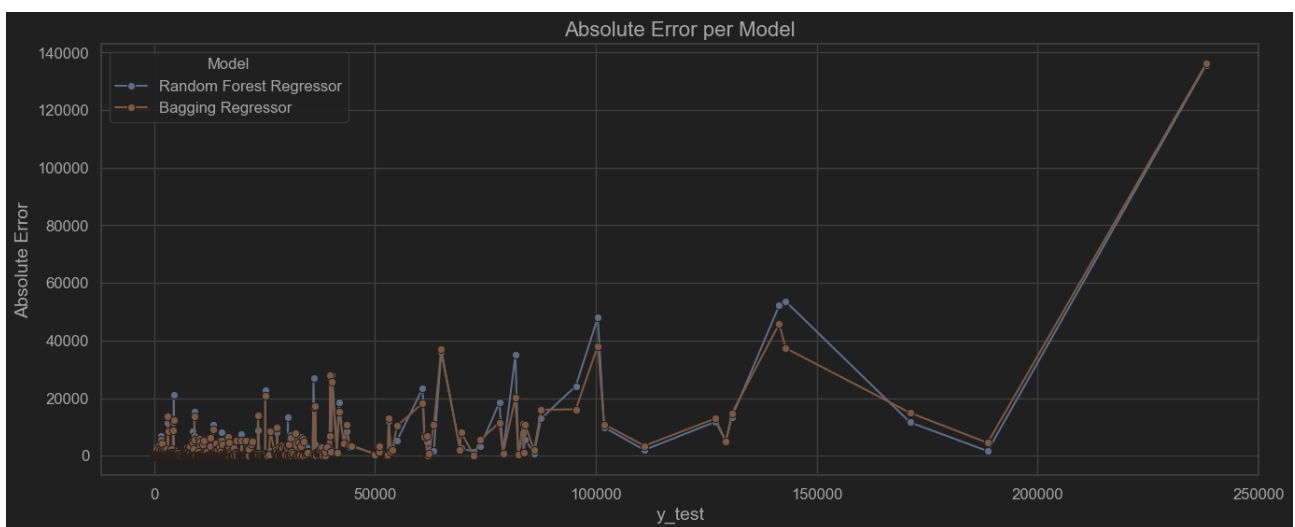


Fig. 45: Andamento dell'errore AE per il singolo df 21_04_42 contenente 10.000 jobs dei modelli Bagging e Random Forest.

Considerando che ogni test set è composto da 2.000 jobs (20% dei 10.000 jobs del dataset), nelle figure precedenti (fig.44 e fig. 45) vi sono 2.000 punti per ogni grafico. Risulta evidente che pochi di questi hanno valori di APE/AE estremamente elevati, portando di conseguenza ad un aumento del MAPE/MAE e della StdAPE/StdAE, nonostante gli indici per il df 21_04_42 siano stati ritenuti accettabili nelle sezioni precedenti.

Pertanto è stata effettuata un'ulteriore analisi per il Bagging e dei suoi errori percentuale e assoluto in riferimenti al df 21_04_42, mostrata nelle figura successiva.

```
res_bagging.describe()
Executed at 2025.01.05 14:56:25 in 31ms
```

	y_test	Absolute Error	Absolute Percentage Error
count	2000.000000	2000.000000	2000.000000
mean	7723.043500	605.506774	10.510517
std	16163.438193	4023.042153	43.408893
min	1.000000	0.000000	0.000000
25%	36.750000	0.408250	0.110074
50%	1269.000000	11.716000	0.859334
75%	10661.250000	93.499000	4.235639
max	238165.000000	136156.833000	1048.128916

Fig. 46: Statistiche complete dell'AE e dell'APE per il Bagging in riferimento al df 21_04_42

Nel dettaglio sono stati sottratti i punti con valore di APE >= 200, viene quindi mostrata la nuova statistica nella figura successiva (fig.47).

```
res_bagging[res_bagging['Absolute Percentage Error']<200].describe()
```

Executed at 2025.01.05 14:59:43 in 37ms

	y_test	Absolute Error	Absolute Percentage Error
count	1983.000000	1983.000000	1983.000000
mean	7779.129602	580.663984	7.356486
std	16220.562450	4010.311652	19.587823
min	1.000000	0.000000	0.000000
25%	36.000000	0.395000	0.109885
50%	1276.000000	11.140000	0.834135
75%	10892.500000	91.380000	4.086324
max	238165.000000	136156.833000	187.274202

Fig. 47: Statistiche dell'AE e dell'APE per il Bagging in riferimento al df 21_04_42 togliendo i valori con APE >=200 (17 punti tolti su 2.000)

Come si nota nella fig.47, sono stati **tolti 17 punti**, ovvero quelli con valore di **APE >=200**, lo si evince dalla riga “count” che passa da 2.000 a 1.983. Ponendo attenzione sulla colonna dell'errore percentuale, si passa dal valore di **std 43** (fig.46) al valore di **std di 19** (fig.47) togliendo pochi punti (17) sul set di test, mentre per il **MAPE** (riga “mean”) si passa da **10%** a **7%** circa.

Risultati analoghi sono stati ottenuti dalla sezione del Random Forest, che per semplicità non viene mostrata.

Visto ciò che accade sfoltoendo di pochi punti il test set derivante dal df 21_04_42, risulta utile vedere cosa succede nel df che ha i risultati di MAPE peggiori: 21_03_42. Vengono mostrati degli esempi nella figura seguente (fig. 48).

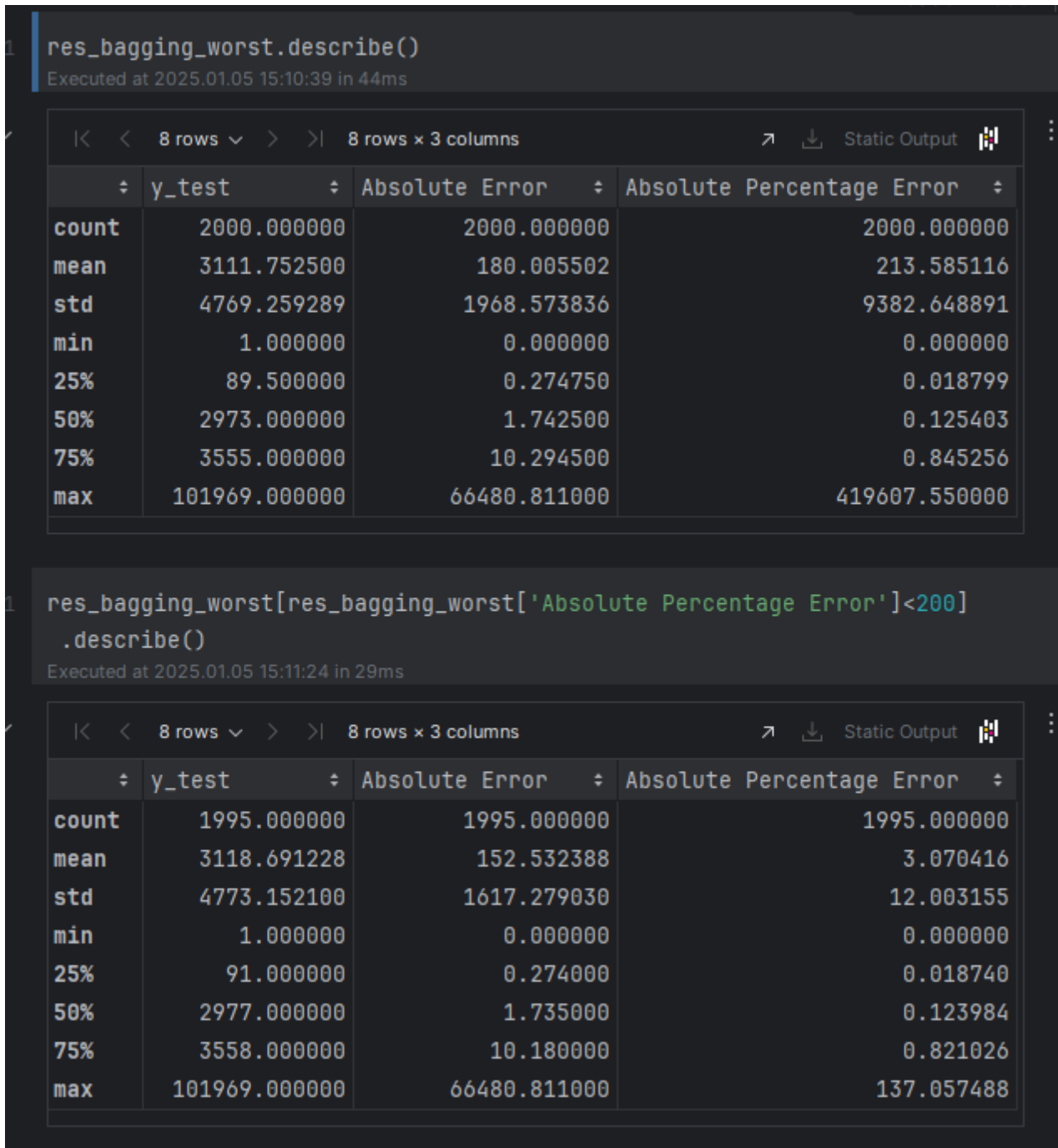


Fig. 48: Statistiche dell'AE e dell'APE per il Bagging in riferimento al df 21_03_42 con risultati di MAPE peggiori. La prima parte della figura fa riferimento alle statistiche del test set completo (2.000 punti), la seconda parte invece è priva dei punti con $APE \geq 200$ (1.995 punti).

La prima parte della fig.48 contiene tutti i 2.000 punti del test set (“count” = 2.000), la seconda contiene 1.995 punti del test set. Quindi sono stati sottratti i peggiori punti per il valore di APE, tenendo quindi quelli con errore percentuale < 200. Si nota un miglioramento del **MAPE** passando da **213%** a un valore di **3%** circa, e una **StdAPE** da **9382** a **12** circa, togliendo semplicemente 5 punti dal test set.

Di conseguenza, **dalla rimozione di pochissimi punti**, cinque nell'ultimo caso, si nota un netto **miglioramento** degli indici **MAPE** e **StdAPE**. Similmente, anche per l'errore assoluto sono state effettuate analisi analoghe, osservando un netto miglioramento degli indici MAE e StdAE dalla rimozione di pochi punti sul test set, nonché quelli con valore di AE più alti.

L'analisi puntuale degli outliers che distorcono le statistiche per 21_03_42 sono mostrate nella figura successiva per il Random Forest.

	Model	y_test	Absolute Error	Absolute Percentage ...
243	Random Forest...	12.000000	49244.350000	410369.583333
361	Random Forest...	302.000000	1306.410000	432.586093
1045	Random Forest...	45.000000	94.920000	210.933333
47	Random Forest...	41.000000	72.780000	177.512195
981	Random Forest...	56766.000000	94858.890000	167.105116
1098	Random Forest...	1316.000000	2146.950000	163.142097
724	Random Forest...	3197.000000	4421.820000	138.311542
984	Random Forest...	1823.000000	2176.520000	119.392211
894	Random Forest...	1323.000000	1566.350000	118.393802
963	Random Forest...	2961.000000	3095.330000	104.536643

Fig. 49: Statistiche dell'APE e dell'AE per il Random Forest addestrato sul df 21_03_42, ordinato per valori di APE decrescente

Nella fig. 49 sono stati elencati i dati per valore di **APE decrescente**, si vede che vi è un solo valore del test set molto diverso dagli altri, che possiede un errore percentuale di oltre **410.000%**, (per l'esattezza il valore y_test era 12 secondi ma sono stati predetti 4.924.447 secondi) mentre per il resto del set di test i valori sono compresi tra 0% e 432% circa, ciò implica una forte alterazione delle statistiche a causa della singola predizione con errore percentuale di oltre 410.000%.

Quindi risulta utile vedere come si comportano gli **indici senza** considerare i **punti peggiori** del test set sia per l'errore percentuale che assoluto. Tale osservazione verrà proposta nel prossimo paragrafo.

6.3) FILTRAGGIO E INTERVALLI DI CONFIDENZA

Come specificato precedentemente, i punti che rendono gli indici inaccettabili sono pochi rispetto al totale dei punti dei test sets. L'idea emersa, a fronte delle considerazioni fatte fino ad ora, comporta **l'eliminazione** sui 60 test set dei **10 punti peggiori per APE** per il calcolo delle statistiche legate all'errore percentuale e dei **10 punti peggiori per AE** per il calcolo delle statistiche legate all'errore assoluto. In questo modo saranno considerati gli indici inerenti all'errore percentuale e assoluto su un test set non più con 2.000 ma **1.990 punti**. D'ora in poi verrà chiamato "**Filtrati**" tutti i df e le statistiche legate a test set senza i 10 peggiori punti con errore percentuale o assoluto più alti.

Da questi nuovi test set filtrati sono state riproposte le statistiche del MAPE con e senza filtraggio, mostrate nelle figure successive, sia per Bagging che per Random Forest.

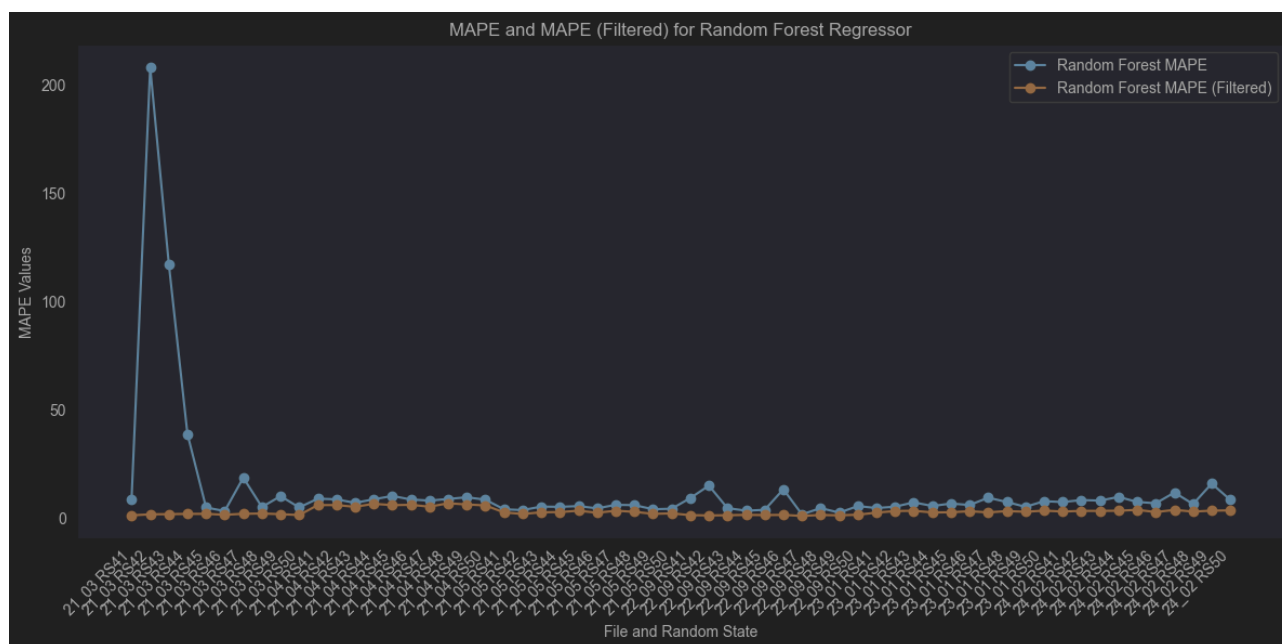


Fig. 50: MAPE e MAPE filtrato per Random Forest

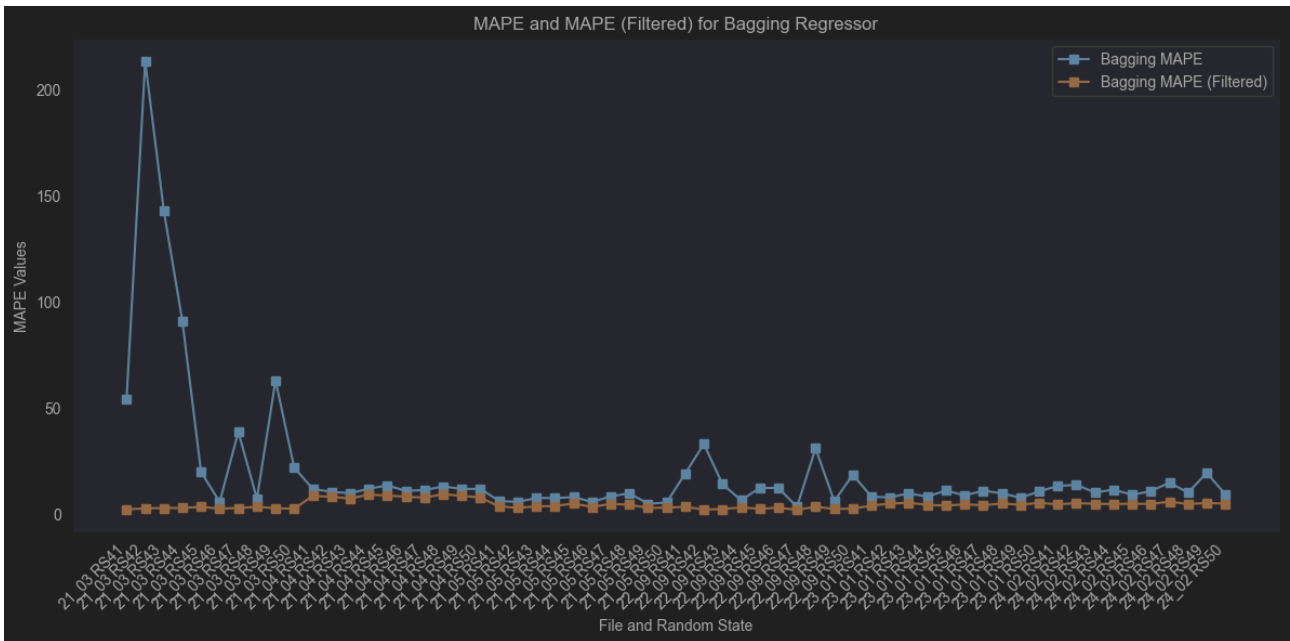


Fig. 51: MAPE e MAPE filtrato per Bagging.

È evidente una importante **riduzione delle statistiche** inerenti all'errore percentuale medio dopo aver tolto solo 10 punti su 2.000 del test set per entrambe i modelli Random Forest e Bagging (fig.50 e fig.51), si notano quindi netti miglioramenti soprattutto per la sezione iniziale legata al df mensile 21_03. Si passa quindi da un MAPE di oltre 200% a un MAPE molto ridotto (<5%). Pertanto, le osservazioni fatte riguardo gli outliers, nel paragrafo precedente, risultano qui mostrate e generalizzate a tutti i 60 df; infatti, si nota una forte riduzione del MAPE per il df 21_03 che in precedenza era considerato inaccettabile per entrambi i modelli. È possibile quindi confermare che per il 99.5% dei jobs (1.990 punti su 2.000) i modelli predicono bene, considerando il valore di MAPE filtrato molto buono. Vi sono ulteriori figure volte alla visualizzazione degli indici (MAE, StdAE, StdAPE) con e senza filtraggio per Bagging e Random Forest.

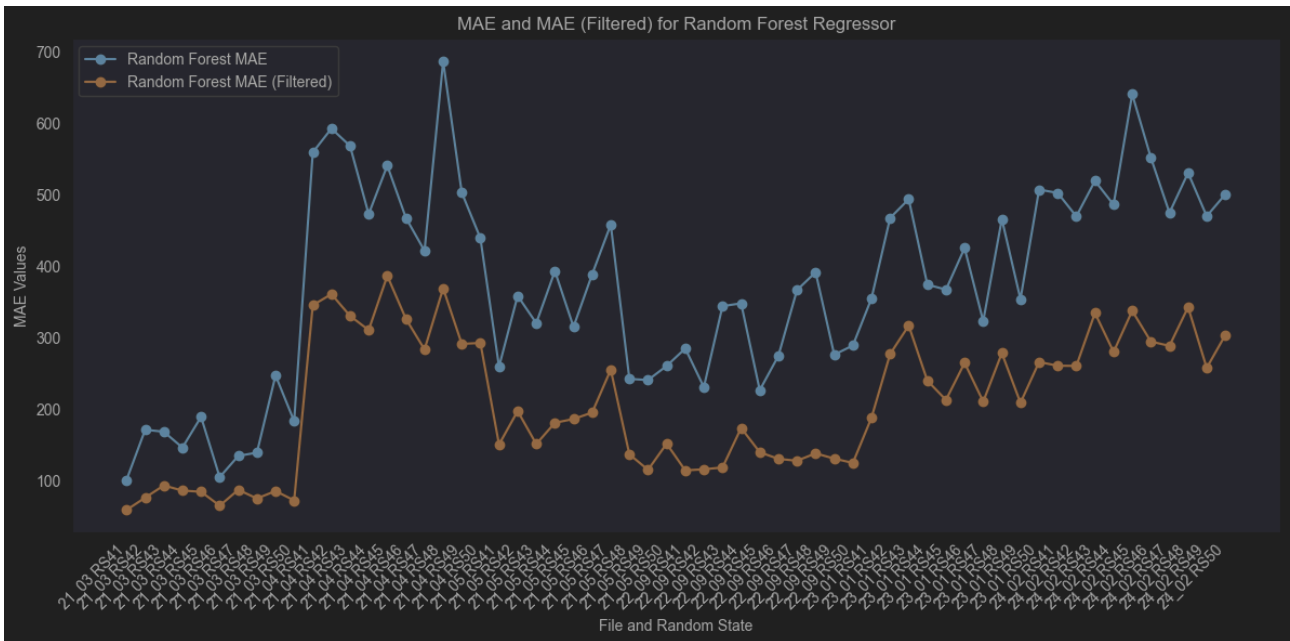


Fig. 52: MAE e MAE filtrato per Random Forest

Anche per il MAE (fig.52) è possibile notare un netto miglioramento. Si nota ancora una volta, nella sezione del 21_03, che i valori di MAE filtrato sono i migliori rispetto agli altri 5 file mensili. Per diversi df sui 60 analizzati si nota che il MAE è più che dimezzato, lo si evince in particolare nella zona del 21_03 e del 21_05.

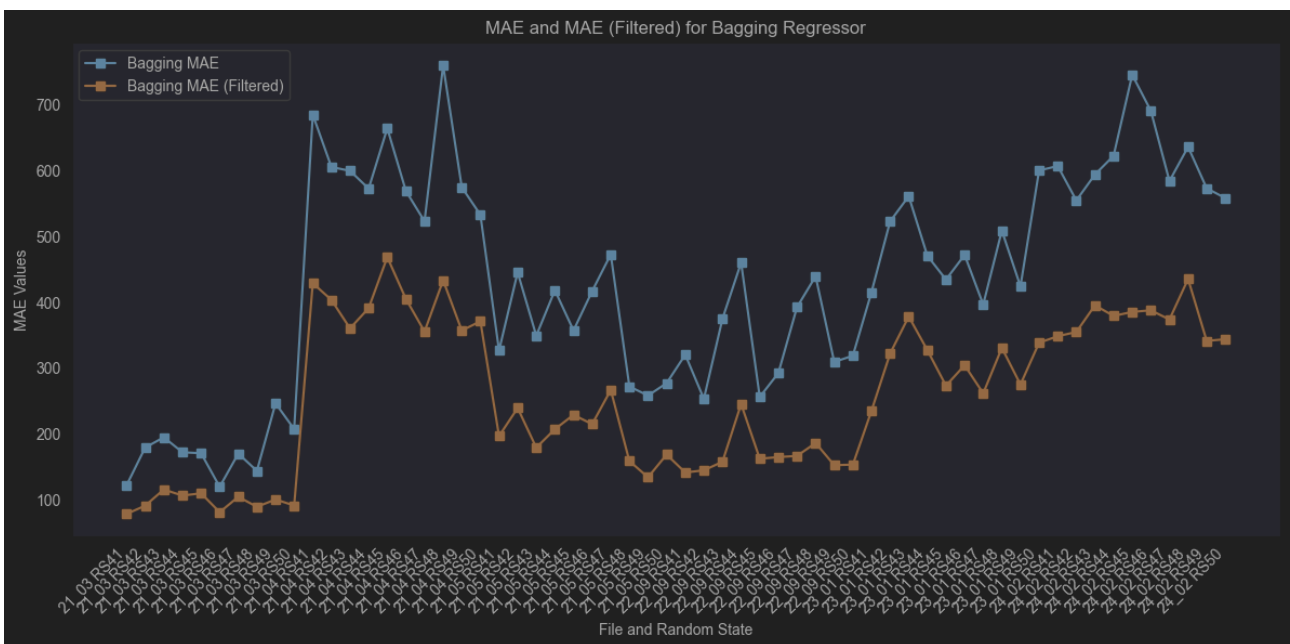


Fig. 53: MAE e MAE filtrato per Bagging

Graficamente si nota che, filtrando i test sets, gli indici ottengono valori decisamente migliori, portando anche i picchi del MAE (fig. 52 e fig.53) da circa 700 a circa 400. Così come per MAPE e MAE, anche la deviazione standard sia dell'errore percentuale che dell'errore assoluto ottengono miglioramenti sostanziali. (da fig.54 a fig. 57)

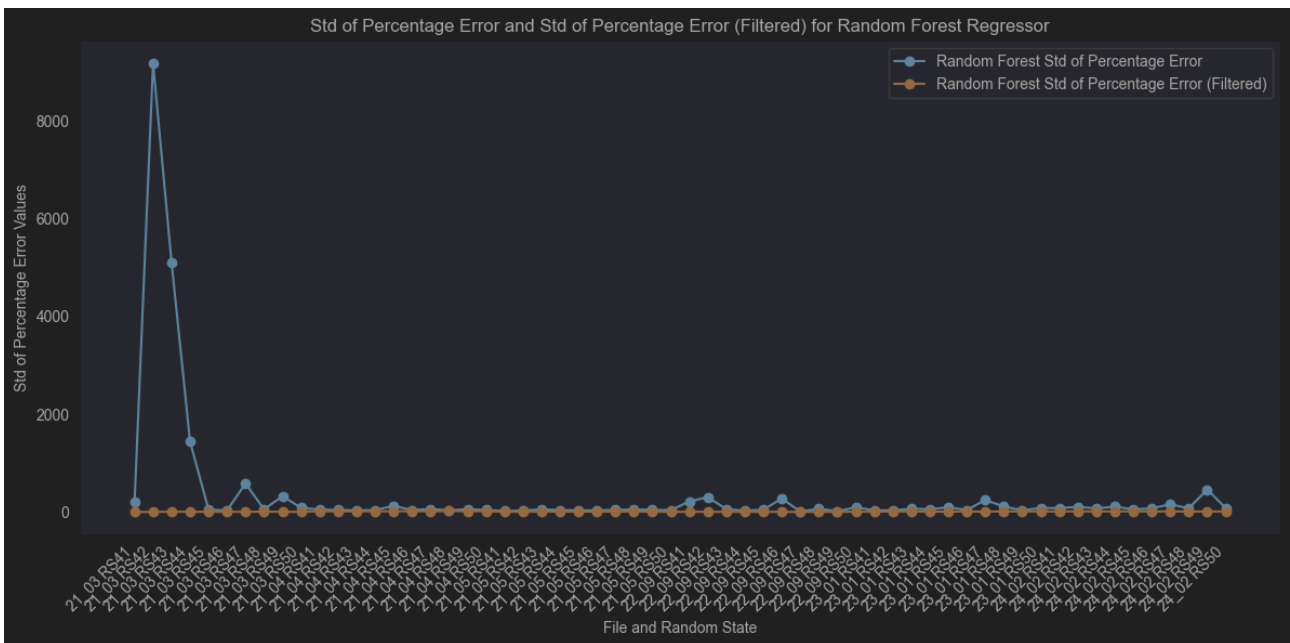


Fig. 54: StdAPE con e senza filtraggio per Random Forest

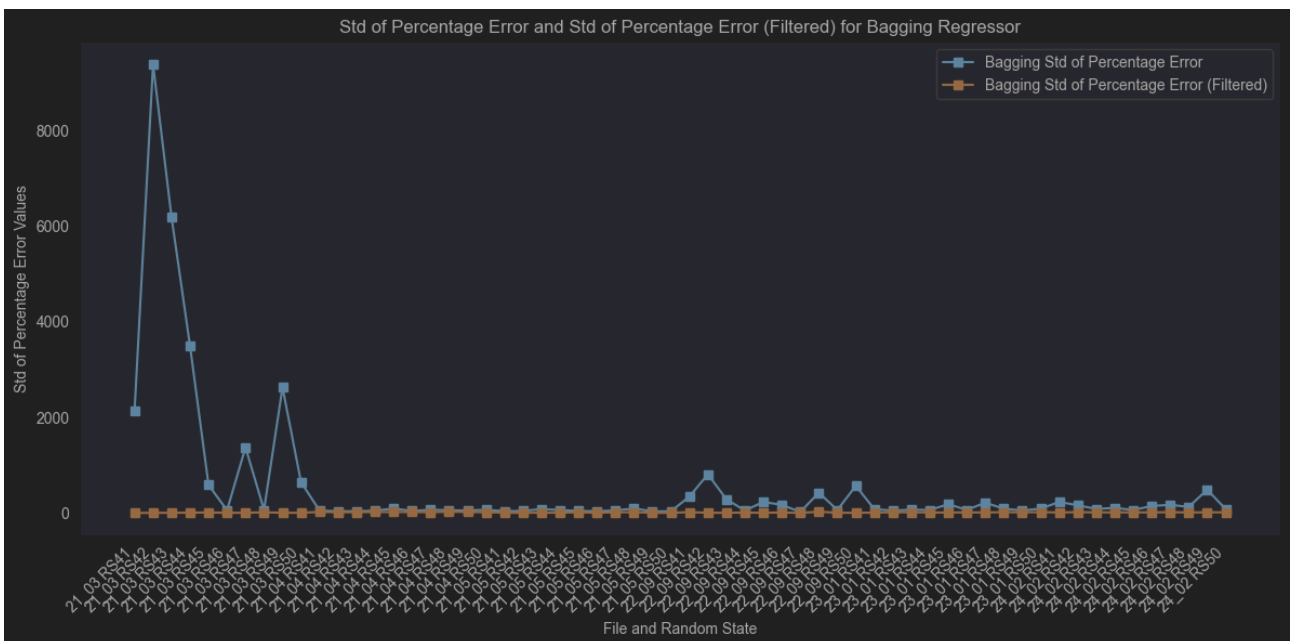


Fig. 55: StdAPE con e senza filtraggio per Bagging

Anche per la StdAE i miglioramenti sono non indifferenti a fronte del filtraggio sul test set. Anche in questo caso, si evince che spesso i valori della StdAE (fig56 e 57) vengono più che dimezzata, come ad esempio nei punti legati al df 21_04_48. Considerando quindi i grafici mostrati in precedenza, i **valori inaccettabili** degli indici sono dovuti a pochissimi punti, per un totale di **10 punti** sui test sets contenenti 2.000 punti.

Ora è quindi possibile proporre la statistica del MAPE e MAE per i due modelli con gli intervalli di confidenza al 95%, senza considerare gli outliers, quindi calcolati sul 99,5% dei test sets.

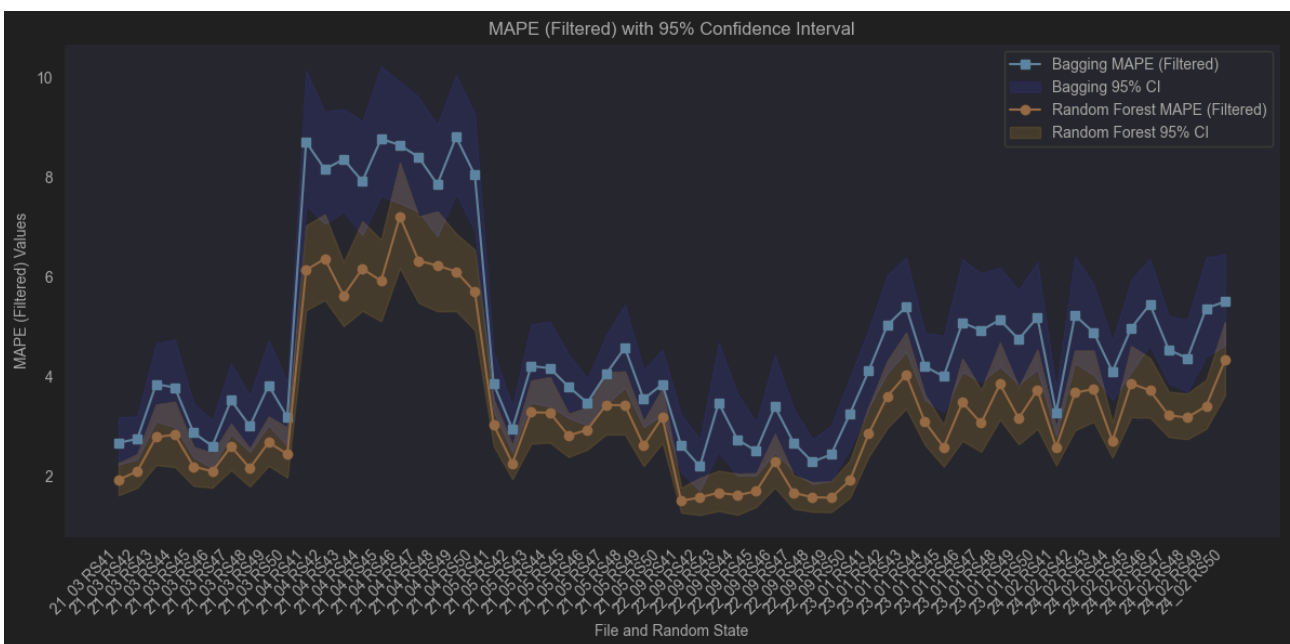


Fig. 58: MAPE filtrato dai 10 peggiori punti con APE più alto, con intervalli di confidenza al 95% calcolati con metodo bootstrap per Random Forest e Bagging.

Dalla fig.58 si evince che il valore del MAPE filtrato per Bagging oscilla tra 2,5% e poco meno di 9%, mentre il range per Random Forest è circa 2%-6,5%. I picchi sono adesso inerenti al df mensile 21_04 che in precedenza è stato ritenuto accettabile. Per ciò che concerne il 21_03 è quindi evidente che si passa da un MAPE con valori assolutamente non accettabili come specificato in precedenza, ad un MAPE con valori addirittura migliori di altri df mensili. Nello specifico si nota che l'intervallo di confidenza al 95% è molto contenuto; pertanto, è possibile affermare che il Random Forest possiede un MAPE filtrato che statisticamente, al 95% appartiene al range 1,5% circa-8%circa, mentre per il Bagging il range è 2% circa-10%circa, il tutto per i 60 df analizzati. In precedenza, non è stato possibile calcolare tale intervallo di

confidenza dovuto alla presenza di outliers. Ciò implica che usando il Random Forest Regressor su datasets di questo tipo l'errore medio percentuale è dell'8% massimo, con una confidenza del 95% (10% per il Bagging), valido per il 99,5% dei test sets.

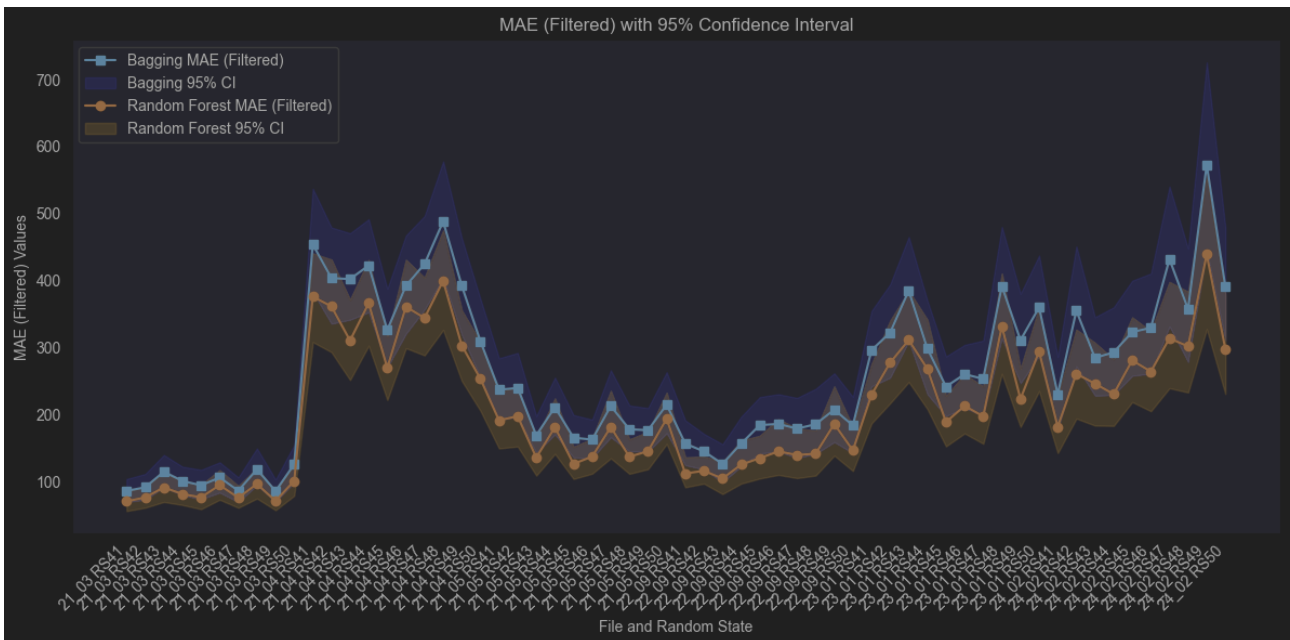


Fig. 59: MAE filtrato dai 10 peggiori punti con AE più alto, con intervalli di confidenza al 95% calcolati con metodo bootstrap per Random Forest e Bagging.

Considerando invece la fig.59 si nota un buon valore di MAE, compreso tra 60 circa e 400 circa per il Random forest, con intervalli di confidenza al 95% ristretti, anche in questo caso. Prendendo il caso peggiore del Random Forest, legato al 24_02_49, si può affermare quindi che al 95% il modello predice con un MAE tra 350 e 550 circa. Si ricorda che il MAE è la media degli errori assoluti; pertanto, affermare che l'errore medio assoluto è al massimo 550 secondi sul complesso delle durate predette risulta accettabile, considerando che la durata dei jobs può avere valori da 1 a ben oltre 250.000 secondi; quindi, il MAE di 550 secondi è accettabile.

L'intervallo di confidenza è uno strumento volto a definire la "fiducia" della statistica. Per calcolare l'intervallo di confidenza del MAPE e MAE, poiché che sono due medie, è stato necessario utilizzare il metodo bootstrap^{21 22}. Considerando un test set con X valori, il bootstrap esegue il resampling con sostituzione: pesca casualmente X valori dal test set in maniera casuale, ciò implica che alcuni valori possono essere pescati più di una volta mentre altri

²¹ <https://machinelearningmastery.com/confidence-intervals-for-machine-learning/>

²² <https://www.geeksforgeeks.org/confidence-intervals-for-xgboost/>

possono non essere pescati affatto per un campione bootstrap. Questa operazione viene ripetuta diverse volte, nello specifico è stata eseguita 1.000 volte. Un esempio del codice bootstrap è mostrato in fig.60

```
# Funzione per il bootstrap
def bootstrap_ci(data, stat_func, n_resamples=1000, alpha=0.05):
    stats = [stat_func(np.random.choice(data, size=len(data), replace=True)) for _ in range(n_resamples)]
    lower = np.percentile(stats, alpha / 2 * 100)
    upper = np.percentile(stats, (1 - alpha / 2) * 100)
    return lower, upper
```

Fig. 60: Istruzione per calcolare l'intervallo di confidenza al 95% tramite il bootstrap

Nel caso specifico `n_resamples`, ovvero il numero di campioni di bootstrap, è posto uguale a 1.000. Una volta calcolato il MAPE originale, e i diversi MAPE per 1.000 i campioni bootstrap, il metodo di selezione dell'estremo inferiore e superiore dell'intervallo di confidenza è quello del **percentile**. Vengono infatti presi i valori di MAPE del 2,5° percentile e il 97,5° percentile sui 1000 campioni di bootstrap per cui sono stati calcolati 1000 MAPE. Il tutto viene eseguito per ognuno dei 60 df in modo da ottenere l'andamento del MAPE con gli intervalli di confidenza al 95% come mostrato in (fig.58), analogamente è stato fatto per la sezione dell'errore assoluto (fig. 59)

Nello specifico, preso in esame il punto 21_03_42 in fig.58 per la zona del Random Forest, si può affermare che il valore del MAPE possiede una probabilità del 95% di appartenere al range 1.5%-2.5% per il 99,5% del test set, considerando che i test set sono stati filtrati.

A vista d'occhio si nota un intervallo di confidenza al 95% per MAPE e MAE molto più piccolo per la sezione del 21_03 alla sinistra che per gli altri 5 file mensili (fig.58 e fig.59), mentre nei paragrafi precedenti si notavano indici inaccettabili del df 21_03, ora si può confermare nuovamente che è il df con indici migliori. Ciò, quindi, è dovuto a pochissimi punti del test set predetti molto male.

È possibile notare che le curve dell'intervallo di confidenza per il **Random Forest**, quelle in giallo, **sono sempre minori** di quelle del **Bagging** in blu, ciò implica che tra i due modelli risulta preferibile effettuare predizioni con il **Random Forest** anziché il Bagging poiché possiede una predizione più accurata con un intervallo di confidenza del 95% per MAE e MAPE più ristretto, sul 99,5% dei test sets.

Di seguito vengono comparati gli indici per Random Forest e Bagging, considerando MAE MAPE filtrati.

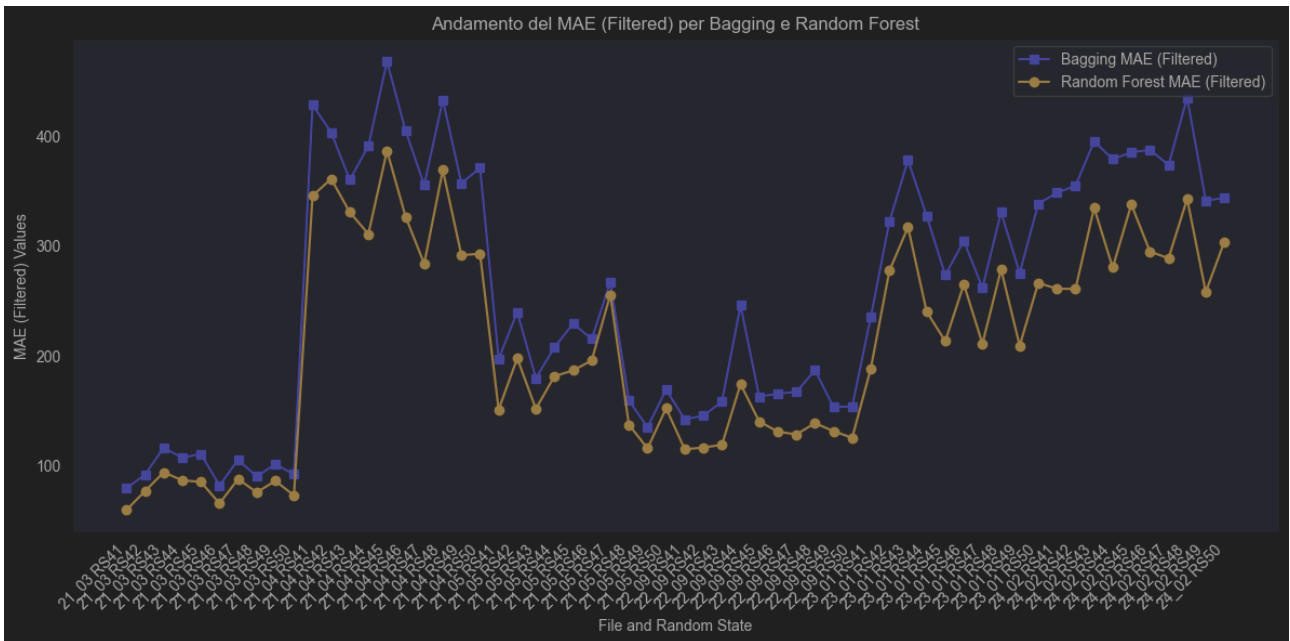


Fig. 6161: Comparazione del MAE filtrato tra Random Forest e Bagging

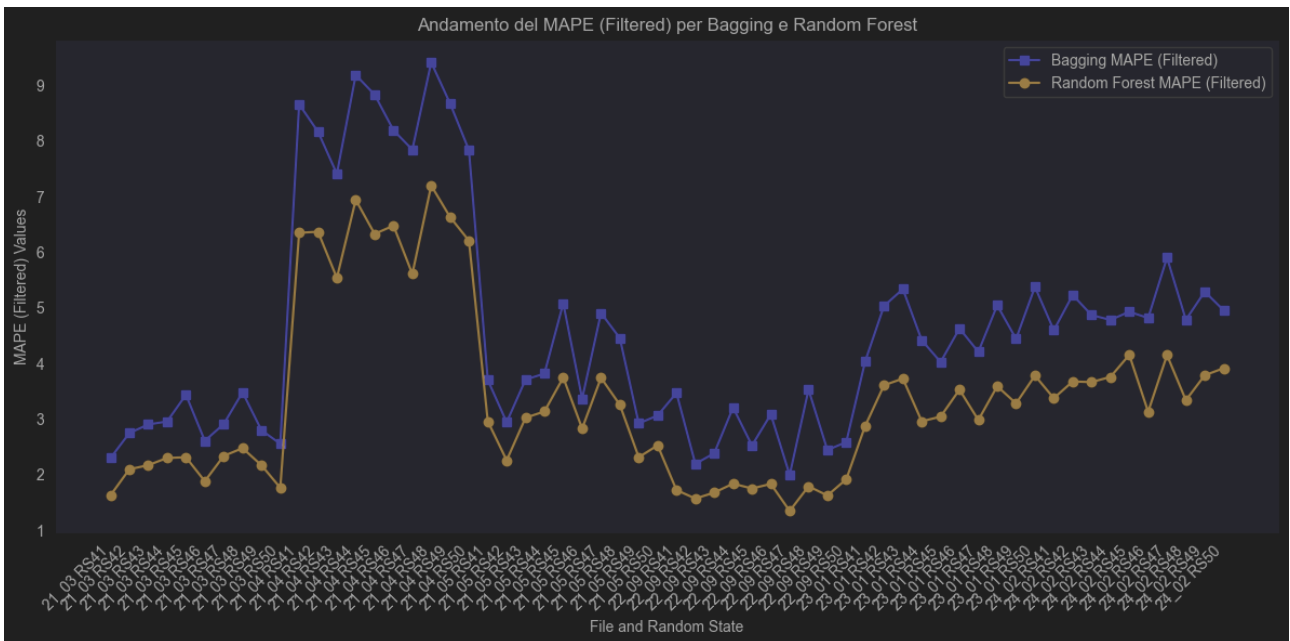


Fig. 62: Comparazione del MAPE filtrato tra Random Forest e Bagging

Nelle figure precedenti (fig.61 e fig.62) è possibile notare che il **Random Forest** per i test set filtrati prevede in maniera **sempre migliore** rispetto al Bagging, con errori minori, lo si evince dal fatto che le curve gialle del Random Forest sono sempre minori delle curve relative al Bagging. È utile osservare il paragone tra MAE e MAPE dei due modelli, notando come gli indici legati al Random Forest sono ridotti di parecchi punti percentuali, lo si evidenzia soprattutto dai picchi. Ad esempio, in fig. 62 il MAPE per il df 21_04_41, paragonando Random Forest (circa 6,5%) e Bagging (circa 9%) vi è una differenza di diverse decine di punti percentuali (6,5 vs 9).

È possibile quindi notare un importante miglioramento generale delle performance legate sia all'errore assoluto che percentuale non considerando 10 punti su 2.000 dei test sets. Pertanto, si può affermare che per il 99,5% del test set i modelli Random Forest Regressor e Bagging propongono delle buone predizioni, con intervalli di confidenza al 95% di MAE e MAPE accettabili. In conclusione, per datasets di questo tipo il Random Forest risulta essere il modello migliore in base agli indici MAPE e MAE, compresi di intervalli di confidenza al 95% sui test sets filtrati.

7) CONCLUSIONE E SVILUPPI FUTURI

In questo elaborato è stata proposta una vasta analisi dei datasets inerenti ai jobs eseguiti dal Supercomputer Fugaku. È stata analizzata in prima battuta la natura dei dati con le statistiche annesse, notando l'alta variabilità di quest'ultime. Sono state analizzate diverse metodologie, tra cui SelectKBest e Kfold-CrossValidation, successivamente scartate. Sono stati poi addestrati i modelli in base alle scelte precedentemente effettuate, una delle quali ha comportato il training dei modelli su 17 colonne anziché 45, tramite l'utilizzo della matrice di correlazione, con risultati nettamente migliori. Sono stati analizzati nel dettaglio gli indici inerenti ai modelli migliori, ovvero Random Forest Regressor e Bagging Regressor, migliori in base all'indice MAPE. È stato constatato che gli indici e il loro valore non accettabile possa dipendere da pochissimi punti del test set (outliers), alcuni punti per cui l'errore percentuale andava oltre il 410.000%. Infatti, rimuovendo i peggiori 10 punti sui test sets, analizzando quindi gli stessi indici sul 99,5% dei test sets, si è mostrato un netto miglioramento generale degli indici MAPE e MAE, importanti per la predizione della durata dei jobs. È stato infine mostrato l'andamento del MAE e MAPE con i rispettivi intervalli di confidenza al 95% sul 99,5% del test set, in modo da ottenere un parametro affidabilistico. Si è evidenziato infatti che il Random Forest Regressor risulta essere il modello migliore in base agli indici MAPE e MAE, ottenendo valori contenuti di entrambi gli indici paragonati a quelli di Bagging Regressor. Nello specifico, un'osservazione sul Random Forest è stata quella inerente all'intervallo di confidenza peggiore per il MAPE, con limite massimo di circa 8%, mentre per il MAE il massimo è circa 550 secondi, valori più che accettabili considerando l'alta variabilità del target (la durata dei jobs varia da 1 secondo a ben oltre 250.000 secondi). L'argomento trattato in questa tesi risulta di grande importanza in ambito HPC perché è volto alla predizione della durata dei jobs, in modo da poter schedare in maniera ottimale l'esecuzione degli stessi. Molti esperimenti non sono stati mostrati in questa tesi, pertanto alcuni sviluppi futuri inerenti ai lavori non completati potrebbero essere:

-analisi di ulteriori modelli ensemble o boosting avanzati, considerando che sono i migliori per questo tipo di dataset, provando ad addestrare ad esempio CatBoost sui datasets proposti

-analisi e regolazione degli iperparametri dei modelli, notando di volta in volta le migliori ottenute, al fine di trovare modelli che prevedano la durata dei jobs in maniera migliore del Random Forest, con particolare focus su LightGBM, GradientBoosting, XGBoost. Il focus su

questi ultimi tre è dovuto al fatto che, durante l'elaborato, hanno ottenuto risultati di r2score migliori rispetto a Random Forest e Bagging, ma scartati a causa del MAPE elevato.

-selezione delle feature di un numero compreso tra 10 e 17, analizzando se la predizione migliora utilizzando meno di 17 features, considerando che il training con 10 features tramite l'utilizzo di SelectKBest (mutual_info_regression) proponeva risultati paragonabili al training su 17 colonne per il Decision Tree

- training di modelli su df ampliati, utilizzando quindi non più 10.000 righe per il dataset ma un numero maggiorato, auspicabilmente il complesso dei file mensili.

-analisi della bontà dei modelli in base alla stagionalità dei jobs, con focus sui mesi in cui i modelli predicono meglio o peggio

-focus su altre features, come exit state (stato di uscita del job), avgpcon (consumo medio al nodo), pclass (classe di prestazione:compute bound-memory-bound) in modo da utilizzarle come target dei modelli.

SITOGRAFIA

Antici, F., Bartolini, A., Domke, J., Kiziltan, Z., & Yamamoto, K. (2024). *F-DATA: A Fugaku Workload Dataset for Job-centric Predictive Modelling in HPC Systems (1.0) [Data set]*. Zenodo. Tratto da <https://doi.org/10.5281/zenodo.11467483>

Barbieri, A. (2023, 08 10). *Cosa sono gli Algoritmi Gradient Boosting*. Tratto da Multinazionali Tech: <https://multinazionali.tech/algoritmi-gradient-boosting-guida-esempi-lavoro-nel-machine-learning>

Beheshti, N. (2022, 03 2). *Random Forest Regression*. Tratto da Towards Data Science: <https://towardsdatascience.com/random-forest-regression-5f605132d19d>

Brownlee, J. (2019, 08 8). *Confidence Intervals for Machine Learning*. Tratto da <https://machinelearningmastery.com/confidence-intervals-for-machine-learning/>

FUJITSU. (2024, 11 19). *Supercomputer Fugaku*. Tratto da <https://www.fujitsu.com/global/about/innovation/fugaku/>

Geeksforgeeks. (2024, 05). *Confidence Intervals for XGBoost*. Tratto da <https://www.geeksforgeeks.org/confidence-intervals-for-xgboost/>

Google Cloud. (s.d.). *Che cos'è l'intelligenza artificiale (AI)?* Tratto da <https://cloud.google.com/learn/what-is-artificial-intelligence?hl=it>

[https://en.wikipedia.org/wiki/Fugaku_\(supercomputer\)](https://en.wikipedia.org/wiki/Fugaku_(supercomputer)). (s.d.).

IArtificial. (s.d.). *Differenze e somiglianze tra modelli lineari e non lineari in ML*. Tratto da <https://iartificial.blog/it/aprendizaje/diferencias-y-similitudes-entre-modelos-lineales-y-no-lineales-en-ml/#:~:text=nel%20Machine%20Learning%3F-,Definizione%20di%20modelli%20lineari,retta%20in%20uno%20spazio%20multidimensional> e.

IBM. (s.d.). *Che cos'è l'high-performance computing (HPC)?* Tratto da <https://www.ibm.com/it-it/topics/hpc>

IBM. (s.d.). *Cos'è una rete neurale?* Tratto da <https://www.ibm.com/it-it/topics/neural-networks>

Inria Learning Lab, s.-l. @. (2024). *Bagging*. Tratto da https://inria.github.io/scikit-learn-mooc/python_scripts/ensemble_bagging.html

Pozzato, F. (2021, 08 13). *Ensemble Model: come migliorare il Machine Learning*. Tratto da Alleantia - NewsRoom: <https://blog.alleantia.com/it/ensemble-modeling-come-migliorare-il-machine-learning>

SAS. (s.d.). *Machine Learning, che cos'è e perché è importante*. Tratto da https://www.sas.com/it_it/insights/analytics/machine-learning.html#:~:text=Il%20machine%20learning%20%C3%A8%20un,intervento%20umano%20ridotto%20al%20minimo

Sayad, D. S. (s.d.). *Decision Tree - Regression*. Tratto da https://saedsayad.com/decision_tree_reg.htm

Scikit-learn developers. (s.d.). *Cross-validation: evaluating estimator performance*. Tratto da https://scikit-learn.org/1.5/modules/cross_validation.html

Scikit-learn. (s.d.). *f_regression*. Tratto da https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.f_regression.html

Scikit-learn. (s.d.). *mutual_info_regression*. Tratto da https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.mutual_info_regression.html

Scikit-learn. (s.d.). *SelectKBest*. Tratto da https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.SelectKBest.html

Wikipedia, L'Enciclopedia Libera. (s.d.). *Apprendimento automatico*. Tratto da https://it.wikipedia.org/wiki/Apprendimento_automatico

Wikipedia, L'Enciclopedia libera. (s.d.). *Coefficiente di determinazione*. Tratto da https://it.wikipedia.org/wiki/Coefficiente_di_determinazione

Wikipedia, L'Enciclopedia Libera. (s.d.). *Mean absolute error*. Tratto da https://en.wikipedia.org/wiki/Mean_absolute_error

Wikipedia, L'Enciclopedia Libera. (s.d.). *Mean absolute percentage error*. Tratto da https://en.wikipedia.org/wiki/Mean_absolute_percentage_error

Wikipedia, L'Enciclopedia Libera. (s.d.). *Mean squared error*. Tratto da https://en.wikipedia.org/wiki/Mean_squared_error