Università degli Studi di Milano - Bicocca
Dipartimento di Economia e Statistica
Corso di Laurea Magistrale in Scienze Statistiche ed Economiche

Ottimizzazione Off-line basata su processi Gaussiani e DBSCAN

Relatore: Prof. Antonio Candelieri

Relazione della prova finale di:
Giacomo Morzenti
Matricola 837395

Anno Accademico 2023-2024

Indice

1 Introduzione

In questo elaborato viene proposto un nuovo metodo per svolgere ottimizzazione off-line attraverso l'utilizzo di **Processi Gaussiani** seguiti da un algoritmo di clustering chiamato **DBSCAN**.

In particolare i problemi di ottimizzazione off-line basati sul modello hanno come fine quello di trovare un input, ovvero un'osservazione, che massimizzi la funzione obiettivo ignota. Questo tipo di problema è molto comune in numerosi ambiti come la progettazione delle proteine, le sequenze di DNA e la robotica. Per risolvere questo tipo di problemi solitamente viene interrogata attivamente l'ignota funzione obiettivo nelle osservazioni proposte, ovvero viene costruito fisicamente il candidato, molecola o robot che sia, e vengono testati e memorizzati i risultati. Questo processo può essere oneroso sia a livello di tempo che di costi, per cui invece si può preferire un approccio che ottimizzi la funzione utilizzando soltanto i dati disponibili.

L'elaborato si sviluppa nella seguente maniera: nel primo capitolo viene presentato il problema di ottimizzaione off-line. Nel secondo capitolo viene proposto e descritto il nuovo metodo che si intende utilizzare. Nel capitolo seguente vengono proposti degli esempi con funzioni note per verificare il funzionamento dell'algoritmo attraverso delle simulazioni. Infine nell'ultimo capitolo viene riportato un esempio di un dataset reale a cui è applicato l'algoritmo e ne vengono discussi i risultati. L'algoritmo è stato implementato completamente utilzzando il software **R**. I dati e lo script sono consultabili all'indirizzo: https://github.com/GiacomoMorzenti .

2 Problema di ottimizzazione off-line

Nei problemi di ottimizzazione online, l'intento è di ottimizzare una funzione ignota $f(x)$ rispetto ai suoi input. L'obiettivo può essere scritto come $arg\ max_x f(x)$. I metodi per ottimizzazione online tipicamente ottimizzano la funzione obiettivo in maniera iterativa, proponendo un certo x_k alla k-esima iterazione e interrogando la funzione obiettivo ottenendo $f(x_k)$. Nel problema di ottimizzazione off-line, a differenza della sua controparte online, l'accesso alla funzione obiettivo non è disponibile. Invece, nel caso off-line, l'algoritmo ha accesso ad un insieme di dati statico $D = \{(x_i, y_i)\}$, in cui sono presenti un'insieme di osservazioni x_i con associati i rispettivi valori della funzione y_i. L'algoritmo utilizza questo insieme di dati e produce un candidato x^* che si ritiene ottimizzare la funzione ignota.
Una domanda che può sorgere è se risulta ragionevole aspettarsi che gli algoritmi di ottimizzazione off-line possano portare a un input che massimizzi la funzione ignota ottenendo un suo valore che sia maggiore di quello che assume nel miglior punto presente nel dataset. Per fornire una spiegazione di come questo sia possibile, consideriamo un problema semplice di ottimizzazione off-line dove la funzione obiettivo $f(x)$ può essere rappresentata come una somma di funzioni di partizioni indipendenti delle variabili in input, ad esempio $f(x) = f_1(x[1]) + f_2(x[2]) + ... + f_N(x[N])$, dove $x[1], ..., x[N]$ denotano sottoinsieme disgiunti delle variabili x. L'insieme di dati di partenza contiene le x ottime per ogni partizione, ma non la combinazione di queste. Se l'algoritmo di ottimizzazione off-line riesce a identificare tale struttura di partizioni indipendenti, allora permette di combinare i diversi ottimi e quindi a migliorare i risultati rispetto all'ottimo del dataset iniziale.

3 Processi gaussiani e DBSCAN

Un processo gaussiano è la generalizzazione della distribuzione di probabilità gaussiana. Mentre una distribuzione di probabilità descrive variabili aleatorie le quali sono scalari o vettori (in caso di distribuzioni multivariate), un processo stocastico controlla le proprietà di una funzione.

Una caratteristica fondamentale dei processi gaussiani che li distingue all'interno dei restanti modelli di machine learning, è che forniscono, non solo una previsione a livello puntuale, ma anche una stima della deviazione standard associata a questa. Proprio questa caratteristica, ovvero la presenza della deviazione standard associata alla stima puntuale, ci permette quindi di avere un'idea non solo della media della stima, bensì anche della sua accuratezza.

All'interno di questo lavoro la deviazione standard verrà sommata o sottratta alla stima puntuale a seconda che si voglia massimizzare o minimizzare la funzione ignota. In particolare avremo la presenza di un coefficiente β che moltiplicherà la deviazione standard e che andrà a sommarsi alla stima puntuale.

L'obiettivo del nostro algoritmo in un problema di massimizzazione della funzione ignota, sarà quindi quello di trovare un'osservazione (input) che abbia un valore più grande del massimo all'interno del dataset. Viceversa in presenza di un problema di minimizzazione andremo alla ricerca di un valore più piccolo del minimo del dataset.

3.1 Algoritmo di ottimizzazione

L'algoritmo qui proposto integra i processi gaussiani e il metodo DBSCAN secondo il seguente schema.

In primis viene allenato il processo gaussiano sul dataset in questione, fornendo quindi l'insieme delle variabili indipendenti come training e dando la variabile da massimizzare come risposta.

In seguito viene creata una griglia di valori relativa alle variabili indipendenti in modo da tracciare al meglio lo spazio delle osservazioni. Viene quindi fatta la previsione sulla griglia di valori appena creata utilizzando il modello GP precedentemente addestrato. In questo modo si ottiene la previsione sia puntuale sia a livello di deviazione standard su tutta la griglia di punti.

A questo punto entra in gioco il parametro β, il quale viene fissato pari a valori come $\{0, 1, 2, 3\}$. Questo parametro β viene moltiplicato per la deviazione standard e sommato alla media. Viene quindi calcolato il valore: $\mu + \beta\sigma$ per ogni punto della griglia e vengono ottenute

cosi le previsioni. In seguito vengono quindi selezionati i punti la cui previsione risulta essere maggiore del massimo del dataset.

Dal punto precedente otteniamo quindi delle zone di punti la cui previsione risulta essere maggiore del massimo del dataset. Tuttavia non è detto che tali zone siano facili da separare nel caso siano multiple, soprattutto in presenza di osservazioni multivariate; è questo il motivo per cui viene utilizzato il modello di Machine Learning per clustering DBSCAN.

DBSCAN permette di classificare i punti in questione in 3 possibili maniere: punti centrali, punti di contorno e outliers. Gli iperparametri del modello DBSCAN sono soltanto due: il primo riguarda la lunghezza del raggio del cerchio in cui controllare i punti vicini, il secondo invece indica il numero minimo di punti che devono essere inclusi nel cerchio per essere considerati punti centrali. Entrambi questi iperparametri vanno fissati in base al caso di applicazione.

Infine dopo aver applicato il DBSCAN e aver individuato eventuali zone, vi è bisogno di un metodo di sintesi per estrarre e proporre un singolo punto per zona. All'interno di questo elaborato si sono applicati due diversi metodi di sintesi: uno calcola la media delle osservazioni per la zona di punti. L'altro sceglie il massimo (o minimo in caso di minimizzazione) tra i valori riportati dalla previsione attraverso il GP.

4 Simulazioni ed esempi teorici

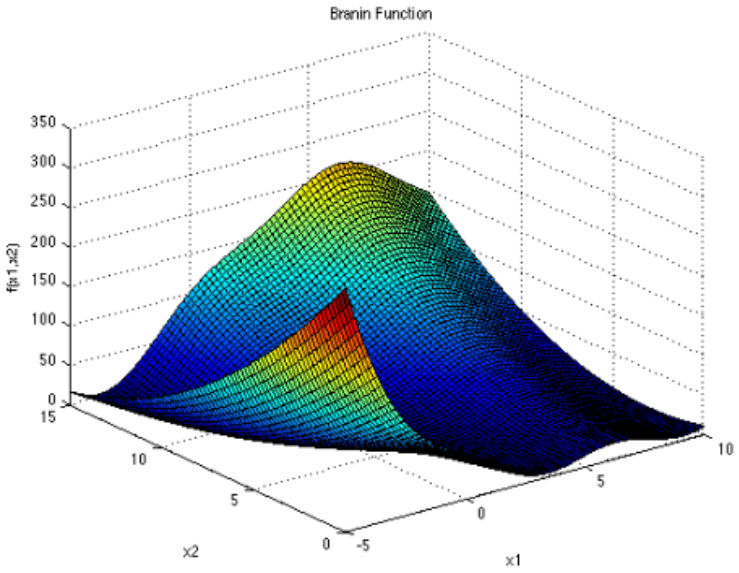
All'interno di questo capitolo verranno presentati due simulazioni dell'algoritmo riguardanti due funzioni note in letteratura. In particolare gli esempi riportati riguarderanno la funzione Branin, una funzione a due dimensioni, e la funzione Hartmann a 6 dimensioni. Entrambe queste funzioni possono essere consultate alla pagina https://www.sfu.ca/ ssurjano/optimization.html.

Per simulare un esempio pratico, verranno estratti dei dataset di punti casuali dalla funzione che verranno considerati come dataset di partenza.

4.1 Branin

La funzione Branin, o Branin-Hoo, ha due dimensioni e tre minimi globali. Con riferimento alla formulazione analitica della funzione in calce alla figura, i valori suggeriti per a, b, c, r, s e t sono: $a = 1, b = 5.1/(4\pi 2), c = 5/\pi, r = 6, s = 10$ e $t = 1/(8\pi)$.

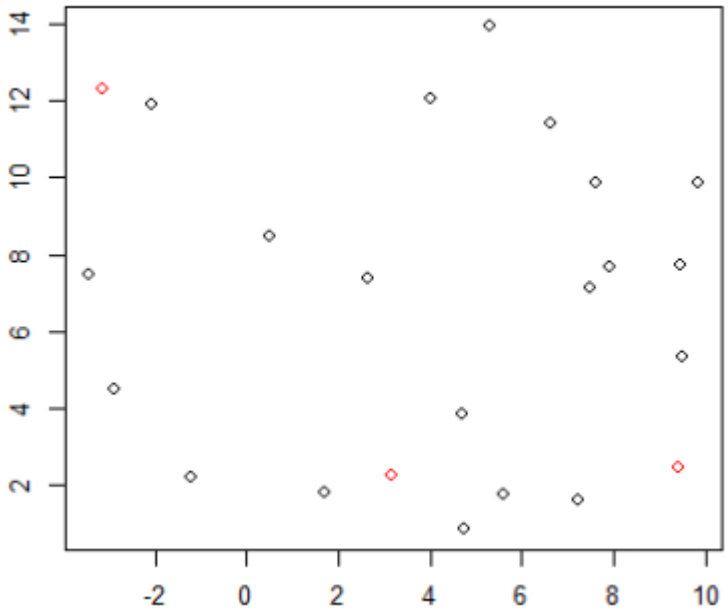
La funzione è solitamente calcolata nel dominio dell'input:
$x1 \in [-5, 10]$, $x2 \in [0, 15]$.



$$f(\mathbf{x}) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1-t)\cos(x_1) + s$$

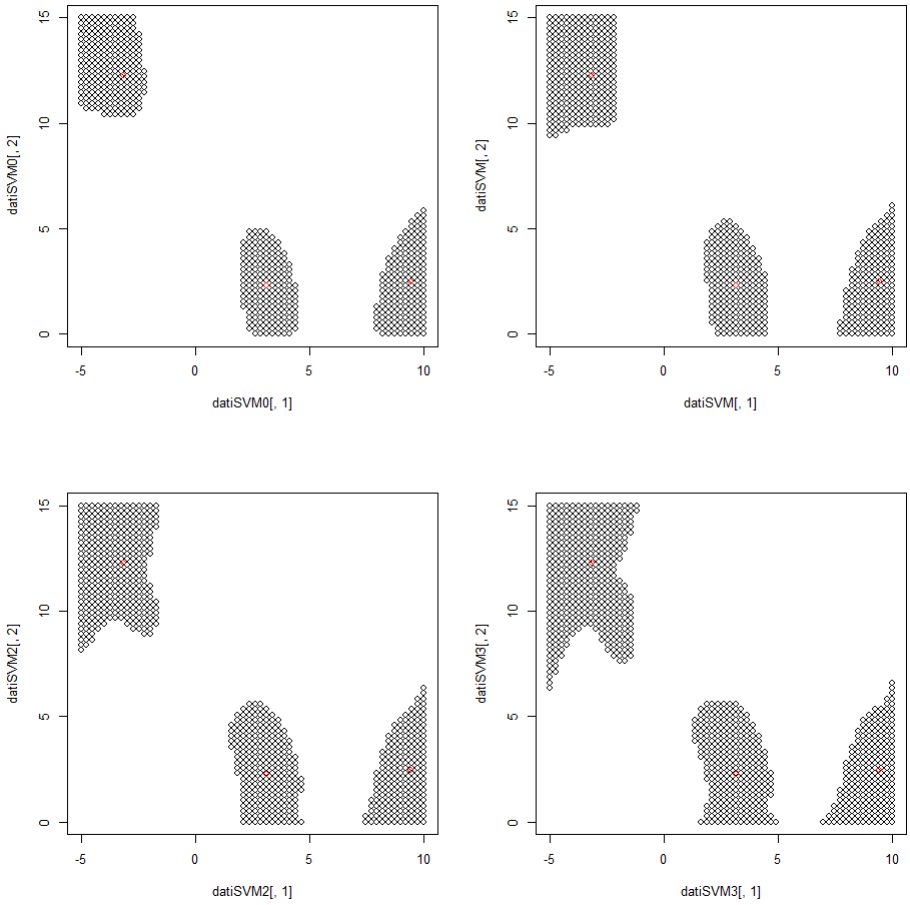
Si applica quindi l'algoritmo descritto precedentemente al caso della funzione branin. In primis si estraggono 20 punti in maniera casuale

dalla funzione e si crea così il dataset di partenza. Si calcola poi il valore dei 20 punti nella funzione branin e si individua il mimimo tra questi. Il valore minimo tra i dati risulta essere pari a 8.414, mentre il minimo globale "reale" è pari a 0.397. Si riporta il grafico dei punti estratti casualmente in nero e in rosso i 3 minimi globali della funzione. Questo è un caso abbastanza sfortunato in quanto i minimi globali sono relativamente isolati.

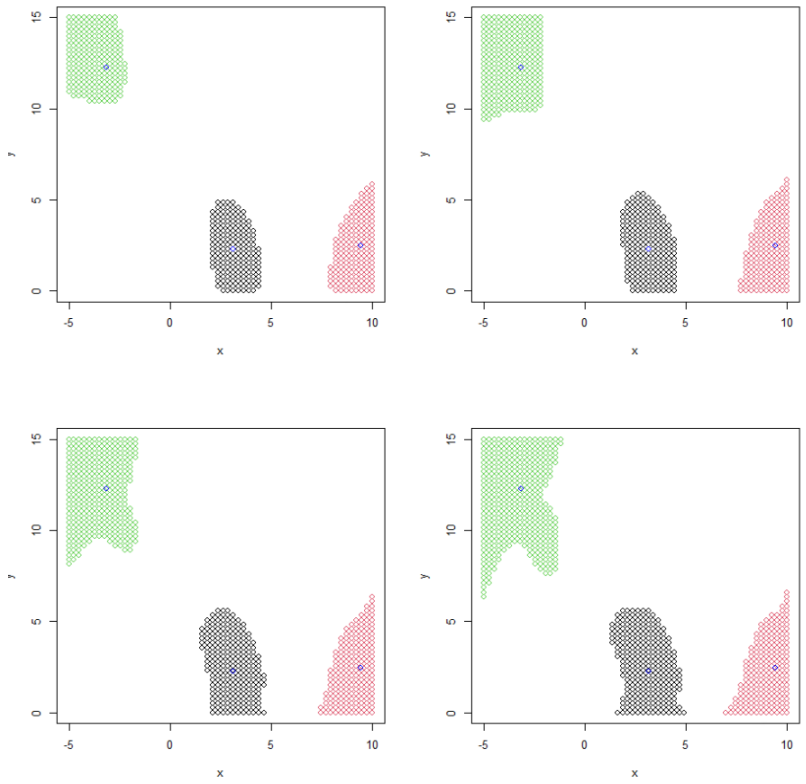


Si addestra il modello di machine learning GP utilizzando il dataset appena creato come training. In seguito si crea una griglia 60x60 sul dominio delle due dimensioni (x1,x2) e si fa previsione utilizzando il GP su questa griglia di punti. Si applica quindi il procedimento descritto nel capitolo precedente, che calcola il valore di $\mu - \beta\sigma$ e seleziona i punti che risultano essere minori del minimo del dataset. All'aumentare di β ci si aspetta di ottenere zone che gradualmente si allargano.

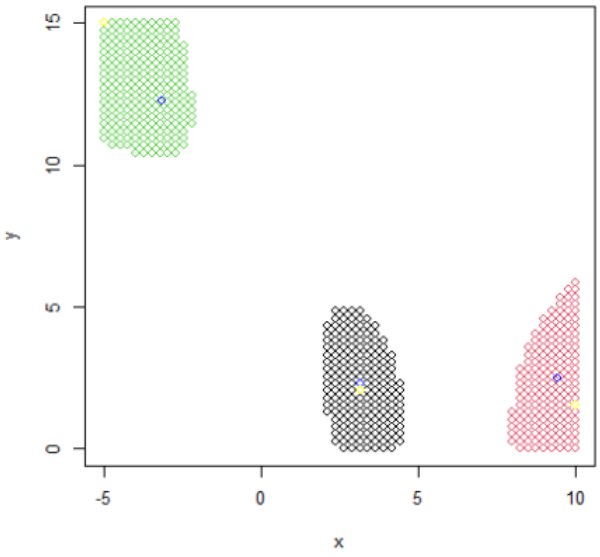
Si riportano i grafici per $\beta = 0, 1, 2, 3$, rispettivamente con anche i minimi globali riportati con il colore rosso:



Effettivamente con l'aumentare di β si ottengono zone gradualmente più larghe. Arrivati a questo punto quindi si può applicare il DBSCAN per suddividere le diverse zone. Si riporta quindi il grafico delle zone suddivise attraverso DBSCAN.

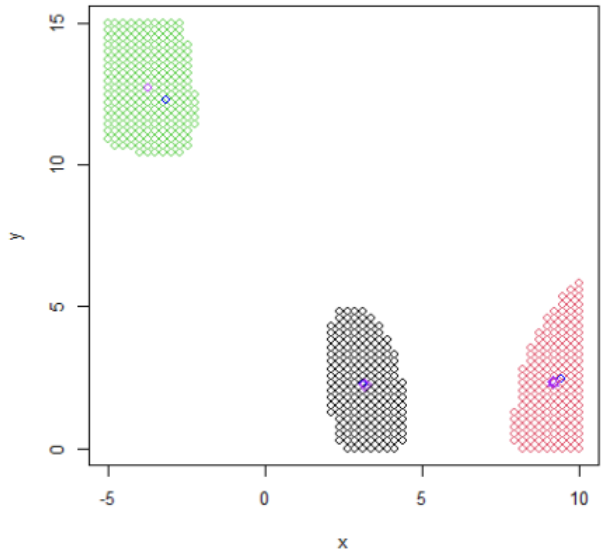


A questo punto occorre quindi sintetizzare ogni zona in un singolo punto e valutare quanto questo punto si avvicini al minimo reale. Si riporta quindi il caso con $\beta = 0$ ed si evidenzia in giallo i punti che risultano essere minimi per ogni gruppo secondo il GP.



Si riporta invece in seguito la media dei punti in viola, da confron-

tare con i minimi reali in blu.

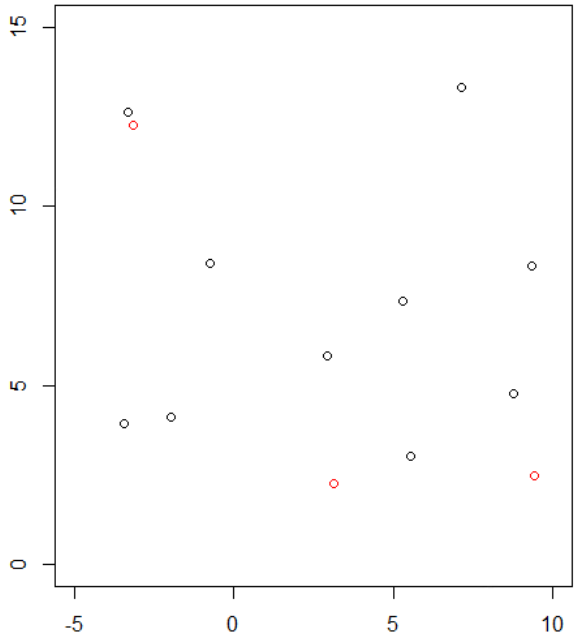


A livello visivo e qualitativo, sembra che la media dei punti sia una sintesi che si avvicina di più ai minimi reali del minimo secondo il processo gaussiano.

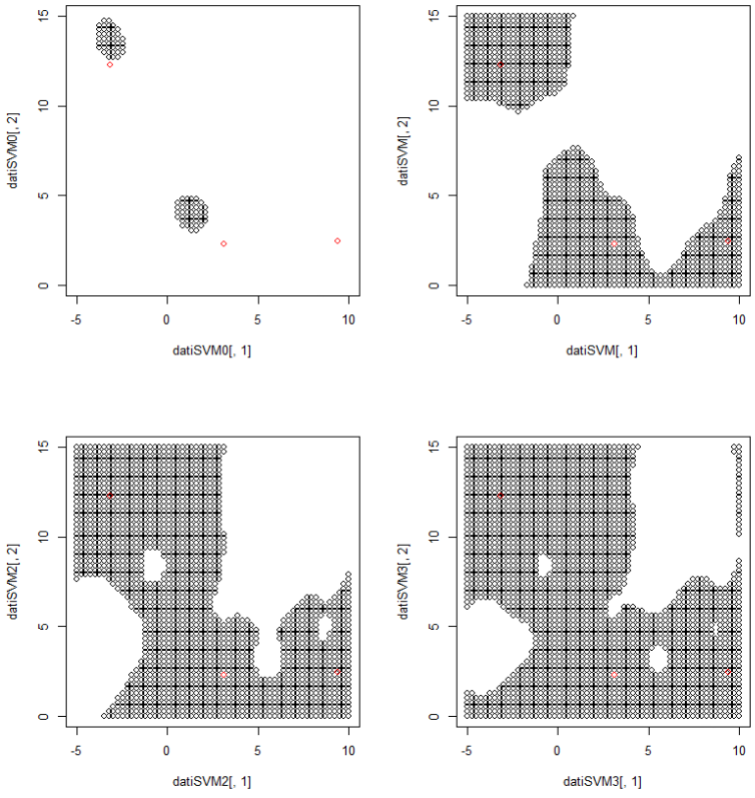
Un altro fattore da prendere in considerazione è il numero di punti estratti dalla funzione iniziale che costituiscono il dataset di partenza. Infatti diminuendo questo numero di punti si ottengono risultati ovviamente peggiori in quanto meno precisi, dal momento che c'è meno informazione sulla funzione all'interno del dataset. Viceversa al crescere del numero di punti iniziali si avranno risultati più precisi e zone più ristrette.

Ad esempio con un dataset di partenza costituito da soltanto 10 punti, le zone saranno decisamente molto meno delineate e più dettate dal caso.

Si riporta in seguito la distribuzione dei 10 punti iniziali rispetto ai minimi globali. In particolare il minimo del dataset risulta avere un valore di 0.61 rispetto al minimo globale di 0.39.

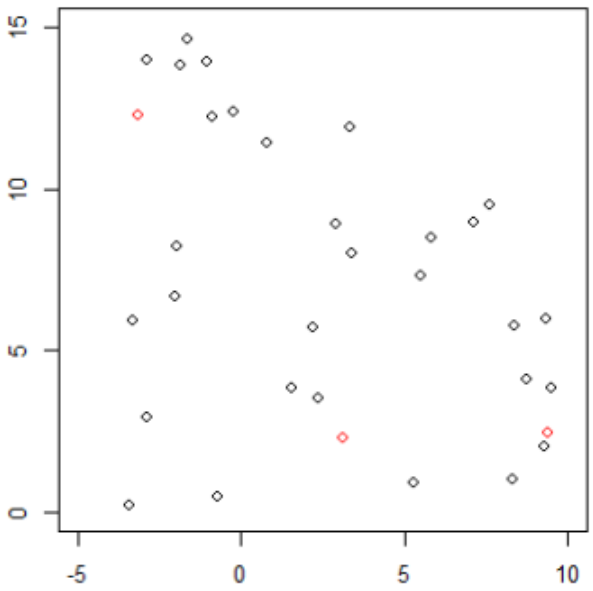


Vengono quindi riportati gli stessi grafici precedenti relativi al caso di 10 punti.

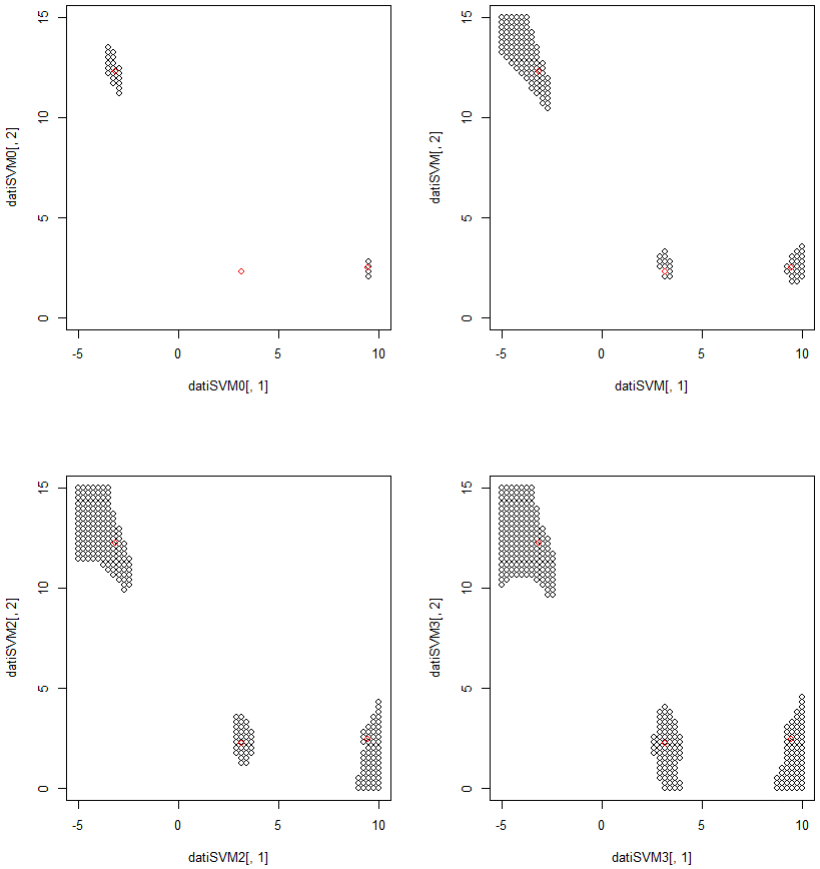


Come possiamo notare dal grafico precedente, nel caso di $\beta = 0$ vi sono soltanto due zone, le quali non contengono nemmeno i minimi globali. Inoltre all'aumentare di β la massa di punti selezionata diventa davvero vasta e poco informativa. Non si procede con la sintesi di punti in quanto è facile prevedere che i risultati sarebbero scarsi.

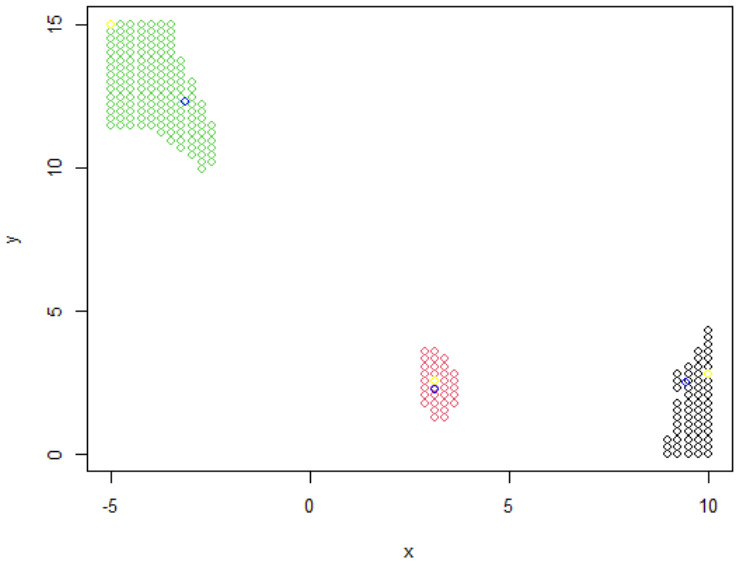
Si riportano invece i risultati relativi al caso di partenza di 30 punti, in cui si prevedono risultati migliori rispetto ai due precedenti. In questo caso il minimo del dataset risulta essere pari a 0.56, il minimo globale rimane 0.39. Ecco la nuvola di punti iniziali:



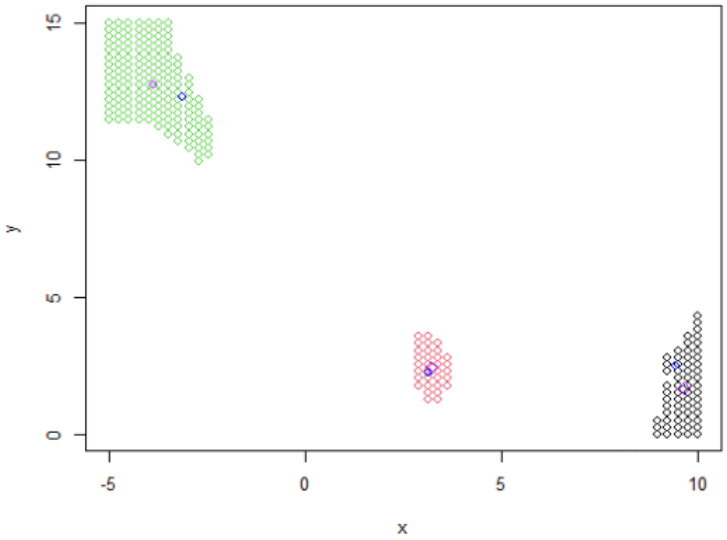
Vengono riportati qui sotto i risultati relativi al variare di β. Come previsto sono risultati decisamente più convincenti rispetto al caso precedente. In particolare, anche in questo caso per $\beta = 0$ esistono soltanto due zone, tuttavia questa volta sono ben centrate rispetto ai minimi globali reali.



Vengono quindi applicati i metodi di sintesi precedentemente descritti al caso relativo a $\beta = 2$. Come negli esempi precedenti i punti gialli rappresentano il minimo secondo il processo gaussiano, mentre i punti blu sono i minimi globali.



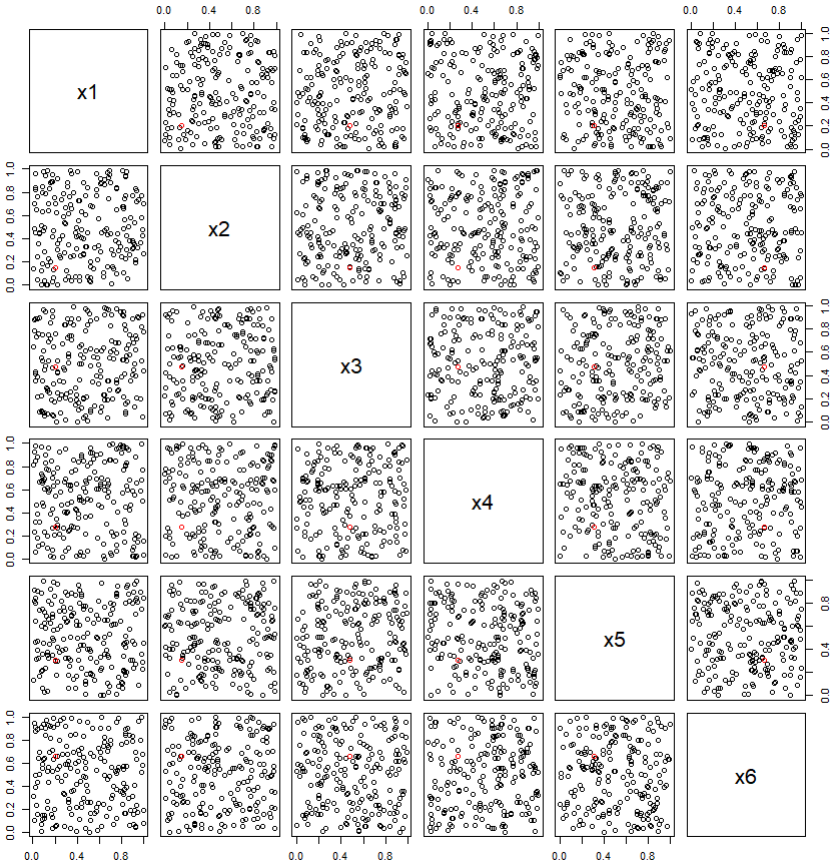
Infine si riporta il secondo metodo di sintesi, che usa la media. I punti di media sono colorati di viola.



I due metodi di sintesi sembrano abbastanza simili a livello di vicinanza con il minimo globale, e differiscono soltanto per il cluster verde dove il metodo della media risulta essere migliore.

4.2 Hartmann 6d

La funzione Harmann a sei dimensioni ha 6 minimi locali. La funzione è solitamente calcolata nel dominio: $x_i \in (0, 1)$ per ogni $i = 1, \ldots, 6$. In questo caso la funzione ha un solo minimo globale, il quale ha valore pari a -3.32. Si applica quindi l'algoritmo alla funzione Hartman 6d. Vengono estratti casualmente 200 punti dalla funzione che rappresentano il dataset di partenza. Come fatto precedentemente viene calcolata la funzione in questi 200 punti e viene calcolato il minimo tra questi punti. Nel nostro caso il minimo del dataset risulta essere pari a -2.66. Si riporta il grafico del dataset di partenza assieme al minimo globale in rosso. Non essendo possibile riportare un grafico a sei dimensioni, vengono riportati gli scatter plot per ogni coppia di dimensioni.



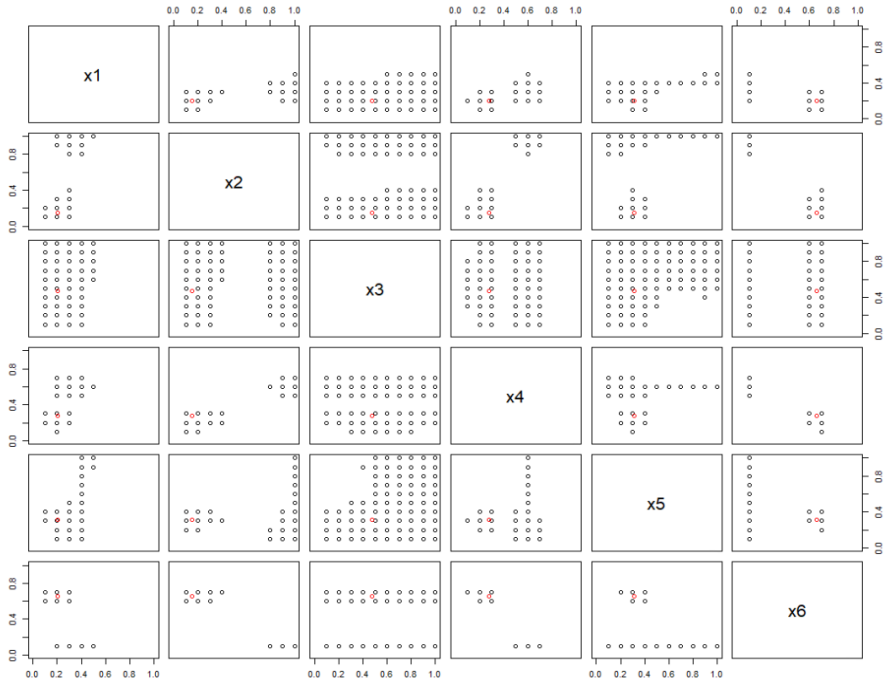
Viene quindi addestrato il modello GP sul dataset appena estratto. Successivamente viene creata una griglia costituita da 10 punti per dimensione sul dominio delle osservazioni. Essendo necessario mappare l'intero spazio delle osservazioni, a questo punto risulta evidente un limite di questo algoritmo. Infatti per mappare 6 dimensioni, utiliz-

zando soltanto 10 punti per dimensione, abbiamo già una griglia di un milione di punti. Più precisamente il numero di punti della griglia risulta crescere esponenzialmente con il numero delle dimensioni.

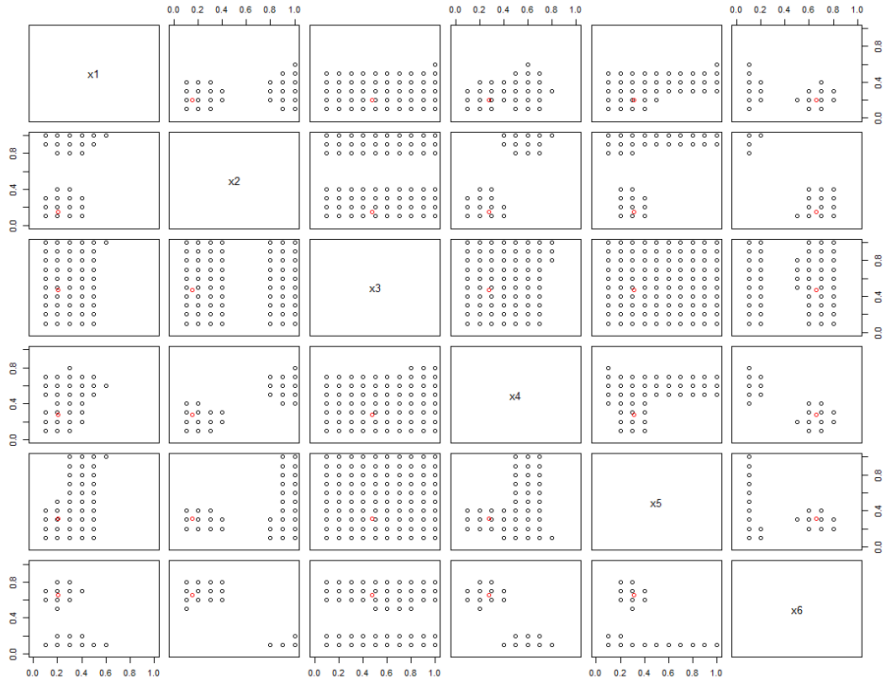
A questo punto vengono fatte le previsioni sui punti della griglia secondo il GP, ottenendo quindi sia la previsione puntuale sia la sua deviazione standard. Come fatto precedentemente, vengono calcolati i valori di $\mu - \beta\sigma$ per ogni punto della griglia e vengono selezionati i punti minori del minimo del dataset. Vengono quindi riportati i grafici rispettivi per $\beta = 0, 1, 2, 3$, con il minimo globale evidenziato in rosso.

$\beta = 0$

$\beta = 1$



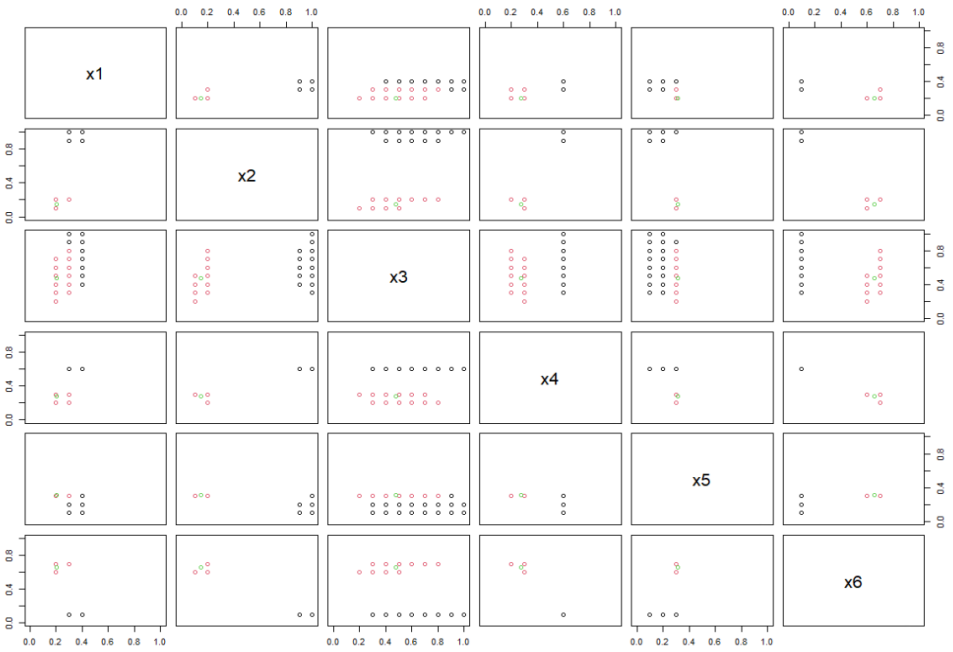
$\beta = 2$



$\beta = 3$

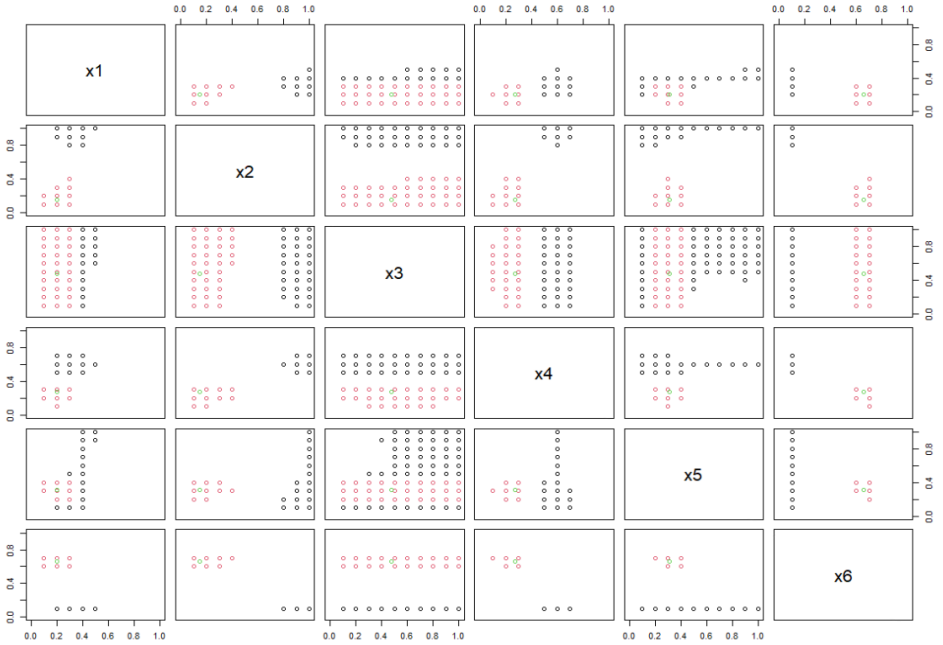
I precedenti grafici mostrano che all'aumentare di β le zone di punti si allargano.

Viene ora applicato l'algoritmo di machine learning DBSCAN per ogni β in modo da separare le diverse zone e quindi poter applicare i metodi di sintesi per ogni zona. Vengono riportati i grafici per ogni β raggruppando i punti attraverso DBSCAN. Il singolo punto evidenziato di colore diverso è il minimo globale.

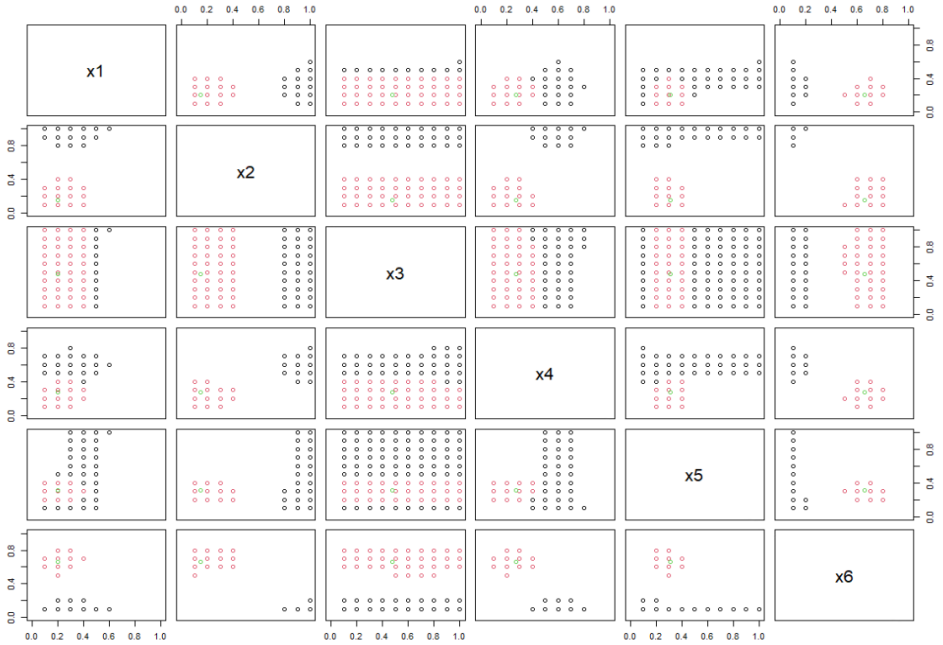


$\beta = 0$

$\beta = 1$



$\beta = 2$

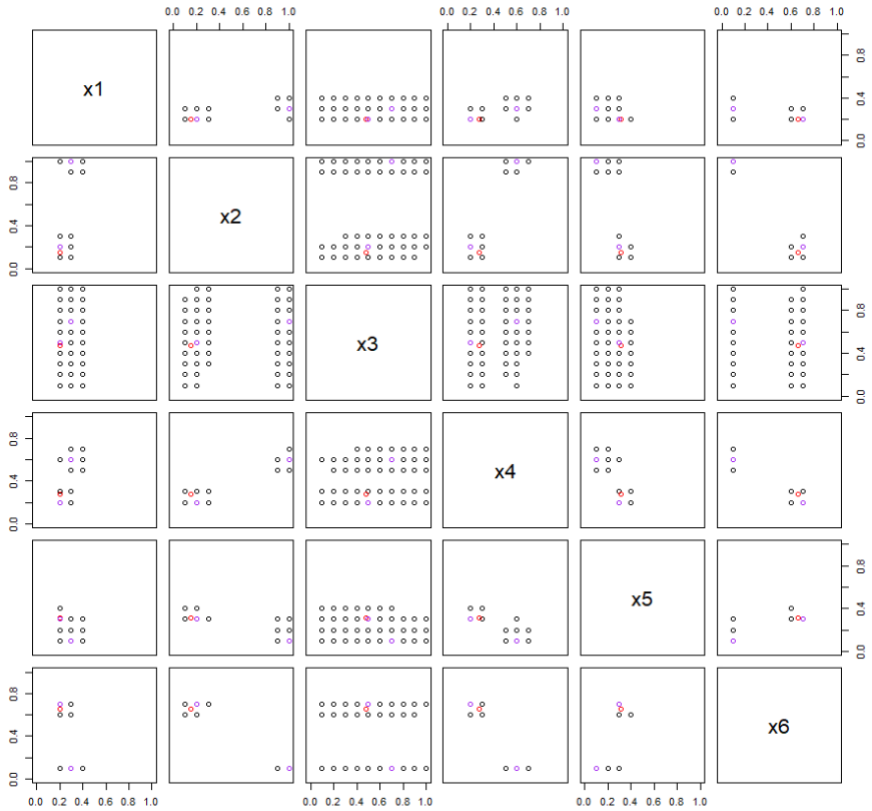


$\beta = 3$

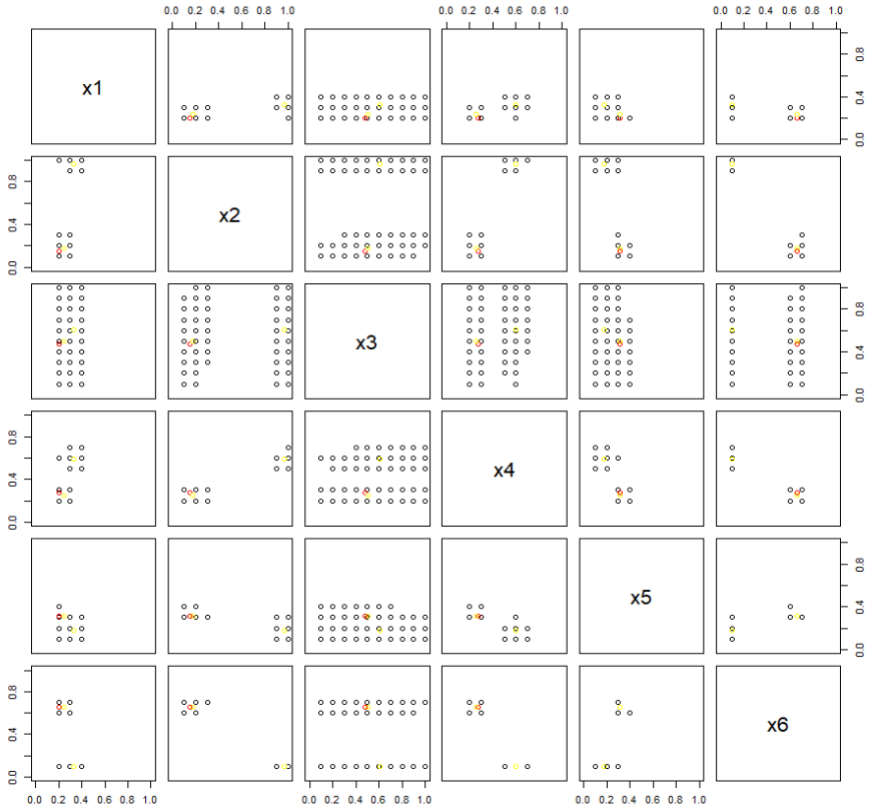
In tutti i grafici precedenti vengono evidenziate due zone, costituite rispettivamente dai punti rossi e dai punti neri. In particolare la zona

rossa contiene sempre il minimo globale, per cui l'algoritmo sembra individuare correttamente tale minimo.

Si prende in considerazione il caso di $\beta = 1$ per l'applicazione dei metodi di sintesi. Viene quindi riportato il grafico evidenziando in viola il punto di minimo secondo il GP per ogni zona.



Vengono riportati in giallo i punti relativi alla media.



I punti di minimo localizzati dall'algoritmo risultano essere molto vicini al minimo globale reale. Inoltre sono stati individuati in questo modo dei punti con valore al di sotto del minimo del dataset. Infatti il minimo del dataset risulta essere pari a -2.67, mentre i due punti di minimo secondo il GP risultano essere rispettivamente -2.48 e -2.92. Allo stesso modo la funzione calcolata nei minimi individuati con il metodo di sintesi della media risulta essere pari a -2.67 e -3.02.
Un altro fattore da valutare è modo in cui cambiano i punti di sintesi al variare del valore di β. Per semplicità di esposizione non verrano riportati i grafici relativi ai β rimanenti ma verrà riportata una semplice tabella riassuntiva.

	$\beta = 0$	$\beta = 1$	$\beta = 2$	$\beta = 3$
minimo GP	-2.48, 2.79	-2.48, 2.92	-2.48, -2.90	-2.48, -2.81
media	-2.68, -3.01	-2.67, -3.02	-2.65, -3.01	-2.65, -2.96

5 Analisi esplorativa

5.1 Introduzione

Il dataset su cui verrà applicato l'algoritmo precedentemente descritto riguarda il calcestruzzo e la sua capacità di resistenza ai carichi e alle tensioni. In particolare la variabile risposta, ovvero quella da massimizzare, risulta essere la resistenza calcolata in MPa, o megapascal, la quale è unità di misura della pressione o dello sforzo nel Sistema Internazionale di unità di misura(SI). Questa è comunemente usata per descrivere la resistenza alla compressione e alla trazione dei materiali. Nel contesto del cemento, l'MPa viene utilizzato per misurare la resistenza del cemento dopo che è indurito. Questa misura è fondamentale perché indica quanta forza per metro quadrato il cemento può sopportare prima di rompersi.
La valutazione in MPa del cemento o del calcestruzzo indica a costruttori e ingegneri l'idoneità del materiale per vari tipi di costruzione. Valori MPa più elevati indicano che il cemento può supportare carichi più pesanti e strutture più sostanziali, come edifici multipiano o grandi ponti.

Quindi, quando si discute di cemento e delle sue applicazioni nella costruzione, l'MPa è una metrica cruciale per determinare come il cemento si comporterà sotto vari carichi e tensioni. I costruttori devono scegliere un tipo di cemento con una valutazione MPa adeguata per garantire l'integrità strutturale e la sicurezza.

5.2 Descrizione e analisi delle variabili

Il dataset è composto da 1030 osservazioni e 9 variabili. Le osservazioni fanno riferimento alle possibili combinazioni di composti dei quali si è calcolata la resistenza in MPa. Quindi ogni osservazione è una lista di ingredienti (8 covariate) uniti in modo da creare un composto la cui resistenza risulta essere nota.
All'interno di questo elaborato si andrà quindi alla ricerca di un composto che abbia una resistenza maggiore di tutti quelli presenti nel dataset.

Su tutte le covariate numeriche si è deciso di applicare la normalizzazione, in modo da avere tutte le variabile su un intervallo (0,1) e poter applicare l'algoritmo DBSCAN su una griglia uniforme.

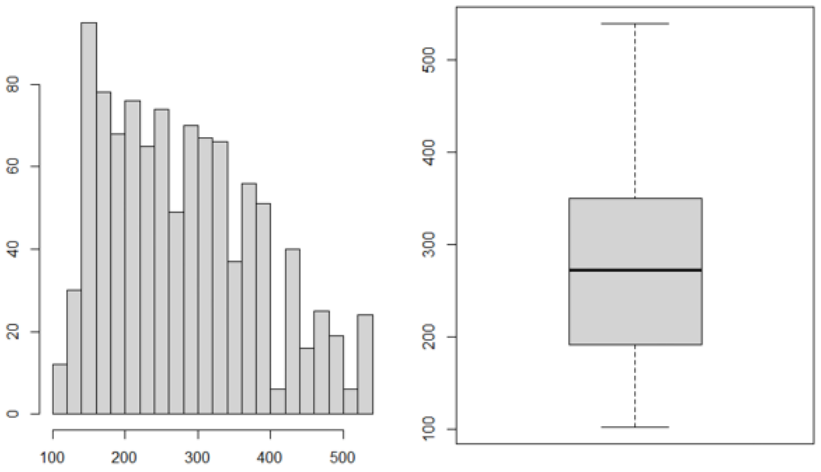
La trasformazione matematica da usare a questo scopo risulta essere

$$x_{norm} = \frac{x - min(x)}{max(x) - min(x)}$$

dove $min(x)$ e $max(x)$ risultano essere rispettivamente il minimo e il massimo della distribuzione dei valori di x all'interno del dataset.

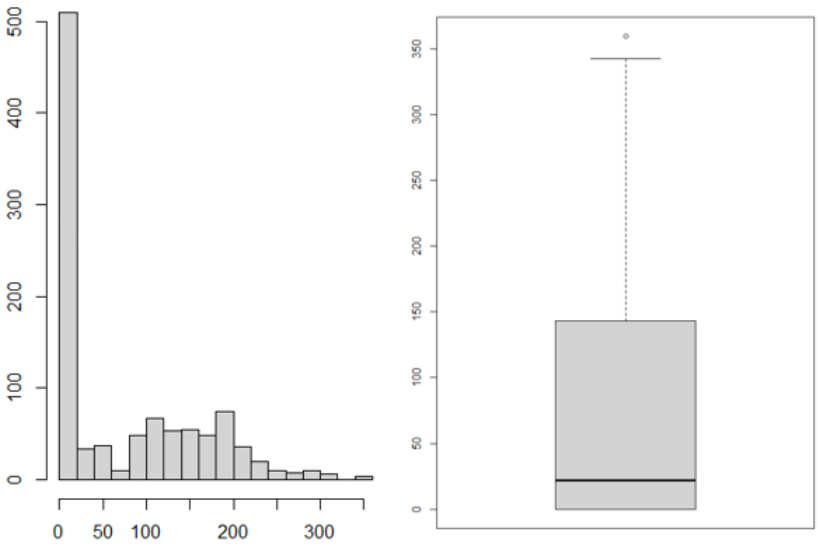
Le variabili sono descritte qui di seguito; per ogni variabile si riporta l'istogramma e il boxplot della sua distribuzione e una tabella riassuntiva.

- **cement** La quantità di cemento in kilogrammi all'interno della mistura.



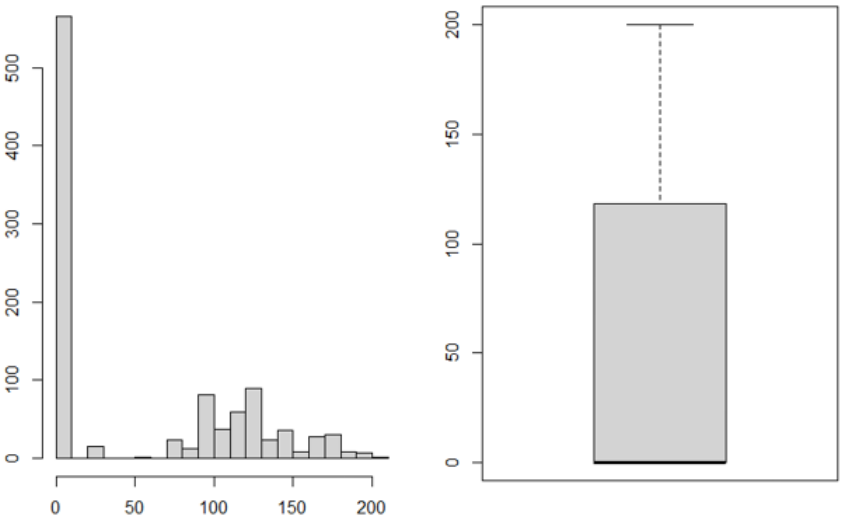
Min	Mean	Max	S.D.
102.0	281.2	540.0	104.5

- **blast furnace slag** La quantità della scoria da altoforno in kilogrammi all'interno della mistura. La scoria da altoforno, conosciuta anche come "blast furnace slag" in inglese, è un sottoprodotto della produzione di ferro nei forni alti. Quando il minerale di ferro viene ridotto per produrre ferro metallico, non solo il ferro ma anche vari altri componenti chimici presenti nel minerale vengono trasformati. La scoria si forma dalla fusione delle impurità nel minerale di ferro, come silice, allumina, calce, magnesia, e altri ossidi, con i fondenti (generalmente calce e/o dolomite) aggiunti per facilitare il processo. La scoria, in particolare quella granulata, è spesso utilizzata come materiale da costruzione, ad esempio come aggregato in cementi e calcestruzzi, grazie alle sue proprietà meccaniche e alla sua resistenza.



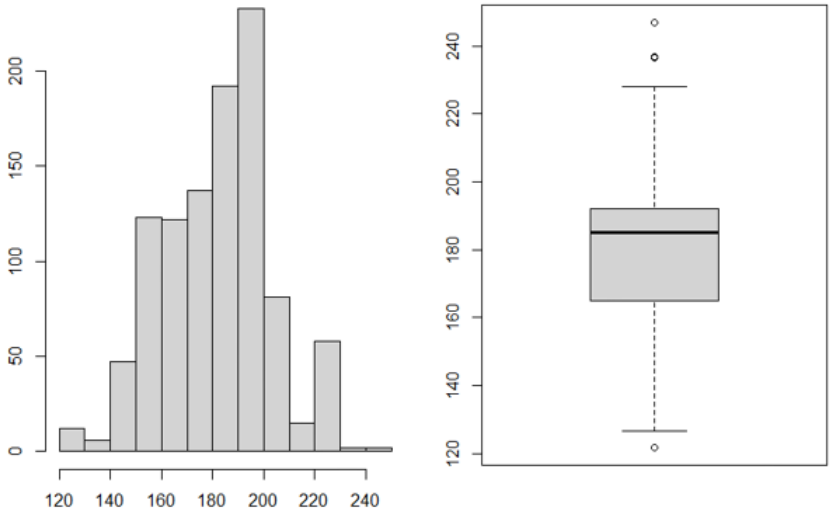
Min	Mean	Max	S.D.
0	73.9	359.4	86.3

- **fly ash** Il fly ash, o cenere volante, è un sottoprodotto della combustione del carbone nelle centrali termoelettriche. È composto principalmente da particelle finissime di silice, allumina e ossidi di ferro. Nel contesto del calcestruzzo, il fly ash viene utilizzato come additivo per migliorarne le proprietà e come sostitutivo parziale del cemento Portland. La variabile descrive la quantità di cenere voltante in kilogrammi all'interno della mistura.



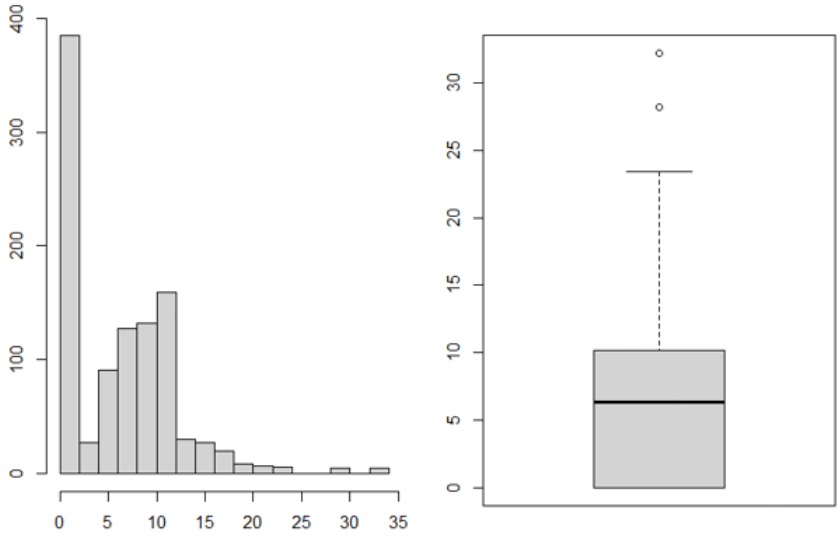
Min	Mean	Max	S.D.
0	54.2	200.1	63.99

- **water** La quantità di acqua in kilogrammi all'interno della mistura.



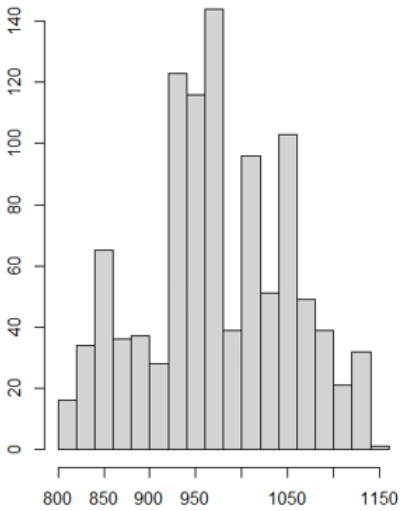
Min	Mean	Max	S.D.
121.8	181.6	247.0	21.3

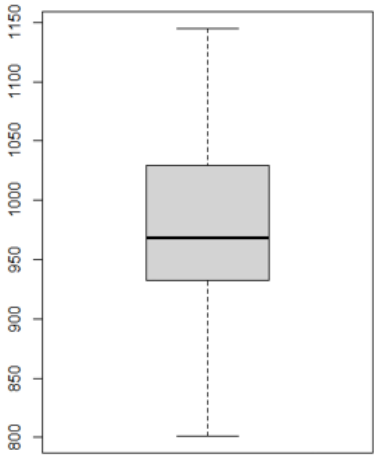
- **superplasticizer** I superplasticizzanti, conosciuti anche come riduttori di acqua ad alta gamma, sono additivi chimici utilizzati nel calcestruzzo per migliorarne notevolmente la lavorabilità senza compromettere la resistenza e la stabilità del materiale. Questi additivi sono particolarmente utili per ottenere miscele di calcestruzzo ad alta fluidità, come il calcestruzzo autocompattante, senza dover aggiungere acqua extra, che potrebbe indebolire la struttura finale. La variabile descrive la quantità in kilogrammi all'interno della mistura.



Min	Mean	Max	S.D.
0.0	6.2	32.2	5.9

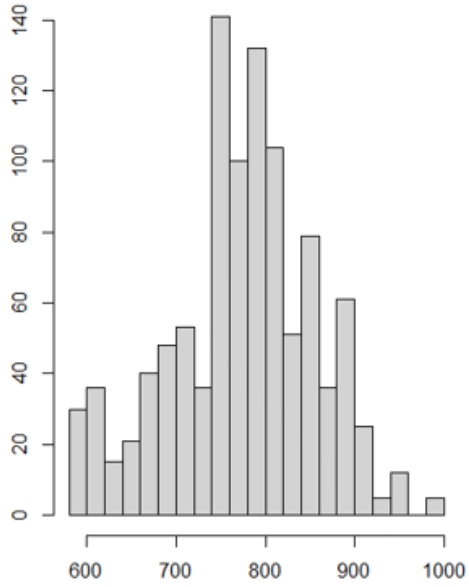
- **Coarse Aggregate** Nel calcestruzzo, l'aggregato grosso si riferisce ai componenti granulari con dimensioni maggiori, generalmente quelli che sono più grandi di circa 4.75 mm (dimensione passante attraverso un setaccio n° 4). Questi aggregati sono una parte fondamentale della composizione del calcestruzzo e sono solitamente costituiti da materiali come la pietra frantumata, la ghiaia, e, in alcuni casi, materiali riciclati come frammenti di calcestruzzo demolito. La variabile fa riferimento alla quantità di aggregato grosso misurata in kilogrammi all'interno della mistura.

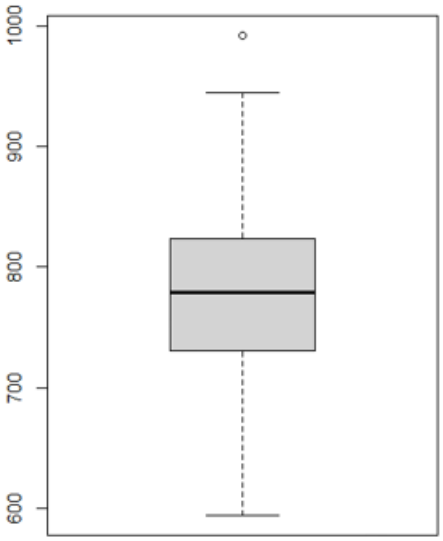




Min	Mean	Max	S.D.
801.0	972.9	1145.0	77.7

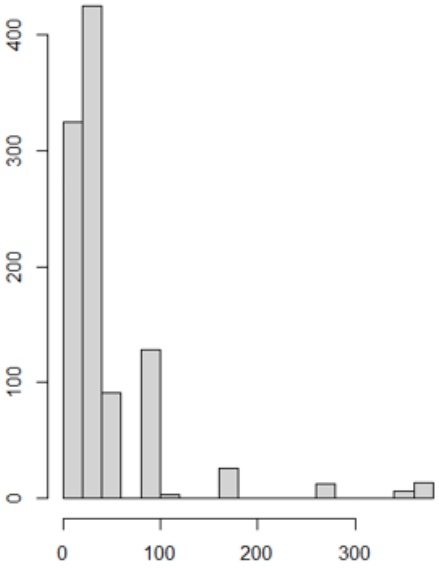
- **fine aggregate** Nel calcestruzzo, gli aggregati fini, spesso chiamati anche sabbia, sono particelle granulari che passano attraverso un setaccio di 4,75 mm di apertura e sono ritenute da un setaccio più fine, tipicamente di 75 micron. Questi aggregati sono cruciali per le proprietà e la performance del calcestruzzo e contribuiscono significativamente alla composizione e alla struttura del materiale finito. La variabile rappresenta la quantità di aggregati fini in kilogrammi all'interno della mistura.

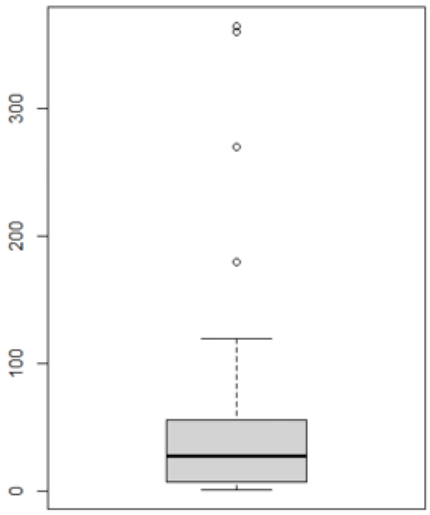




Min	Mean	Max	S.D.
594.0	773.6	992.6	80.2

- **age** Questa variabile rappresenta il numero di giorni in cui il composto è stato lasciato a maturare. L'età del calcestruzzo è un fattore cruciale per determinare la sua resistenza e durabilità.

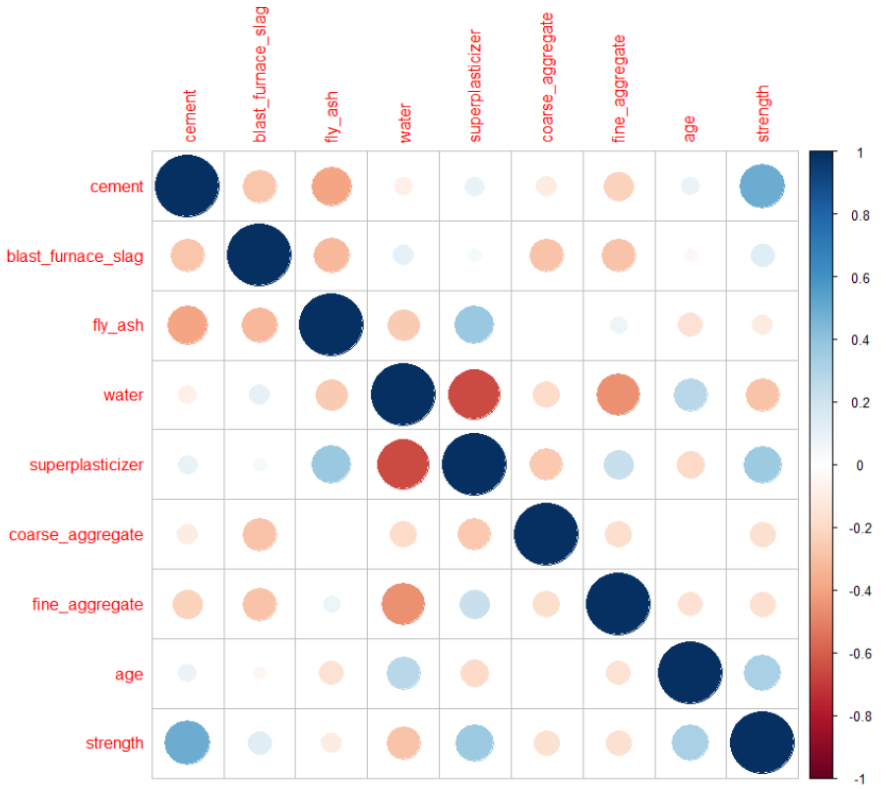




Min	Mean	Max	S.D.
1.0	45.6	365.0	63.2

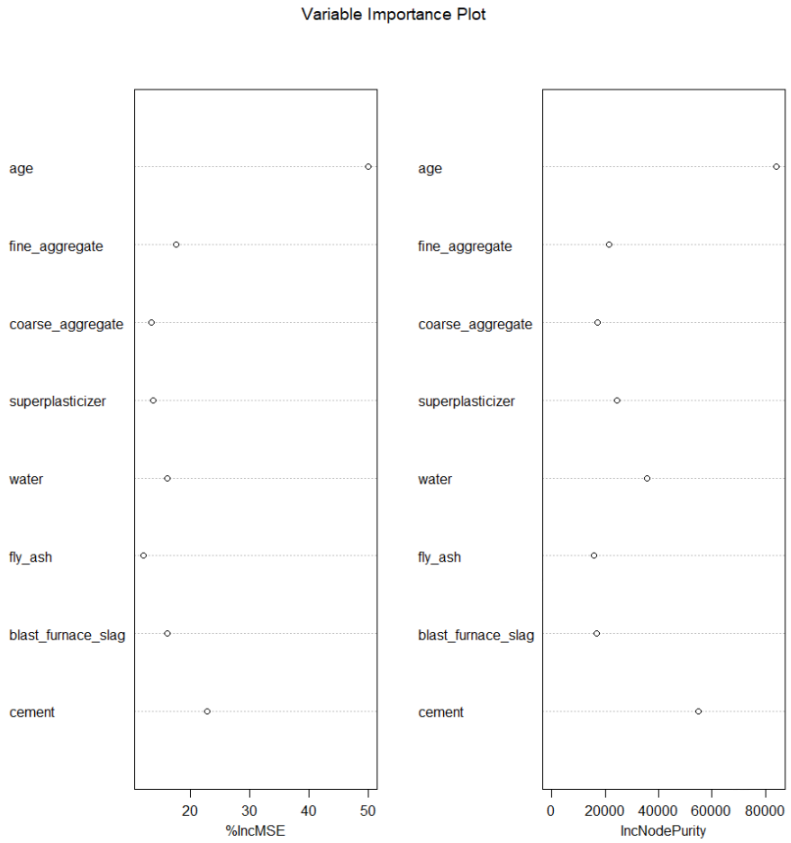
5.3 Analisi delle correlazioni

Attraverso la matrice di correlazione, si sono analizzate le correlazioni tra le variabili, compresa la variabile risposta. Si nota che la correlazione tra la variabile risposta *strength* e le variabili *fly ash* e *blast furnace slag* risulta essere molto bassa.



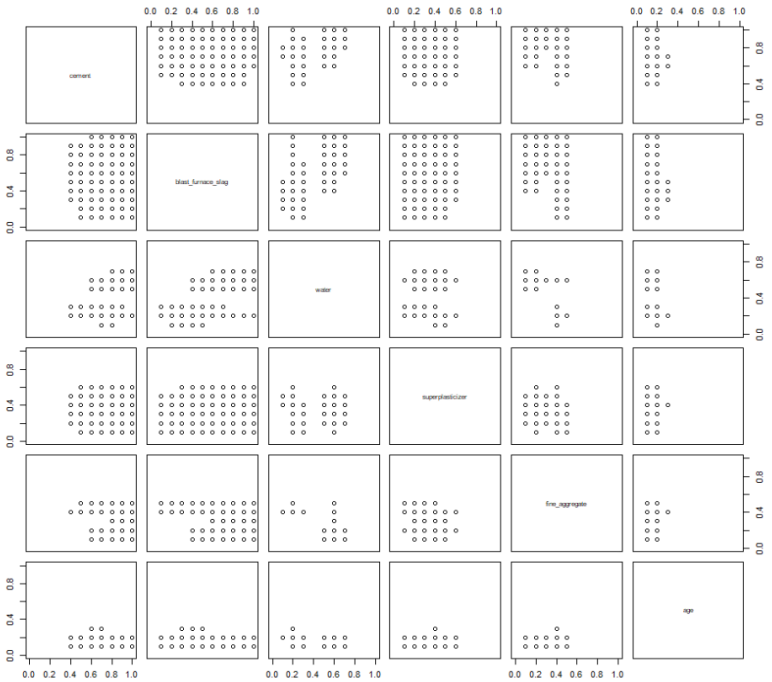
5.4 Selezioni delle variabili con foreste casuali

All'interno di questa sezione vengono riportati i due metodi principali per svolgere selezione delle variabili attraverso le foreste casuali. In particolare il primo metodo fa riferimento a quanto il modello risulti peggiorare a livello di MSE quando i valori della variabile in questione vengono permutati, e quindi viene meno la dipendenza tra la variabile in questione e la variabile risposta. Il secondo metodo invece descrive quanto i nodi risultano migliorare a livello di purità quando viene utilizzata la variabile in questione come split all'interno degli alberi che formano la foresta casuale. Entrambi i metodi concordano nello stabilire che *fly ash* sia la variabile meno importante. Inoltre sia *coarse aggregate* sia *blast furnace slag* hanno valori molto bassi di importanza per i due metodi. Quindi per rendere l'algoritmo meno oneroso a livello computazionale, si è deciso di escludere la variabile *fly ash* e la variabile *coarse aggregate*.

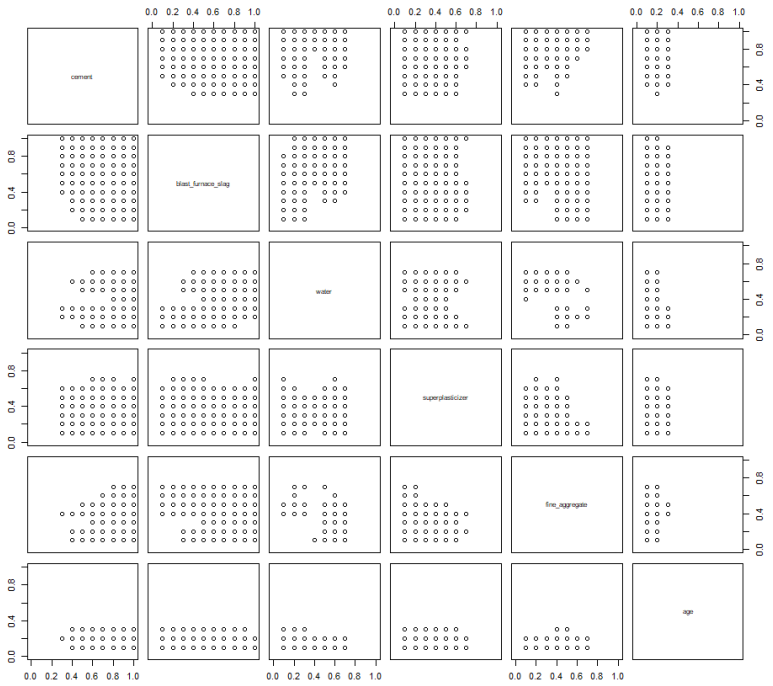


6 Applicazioni e risultati

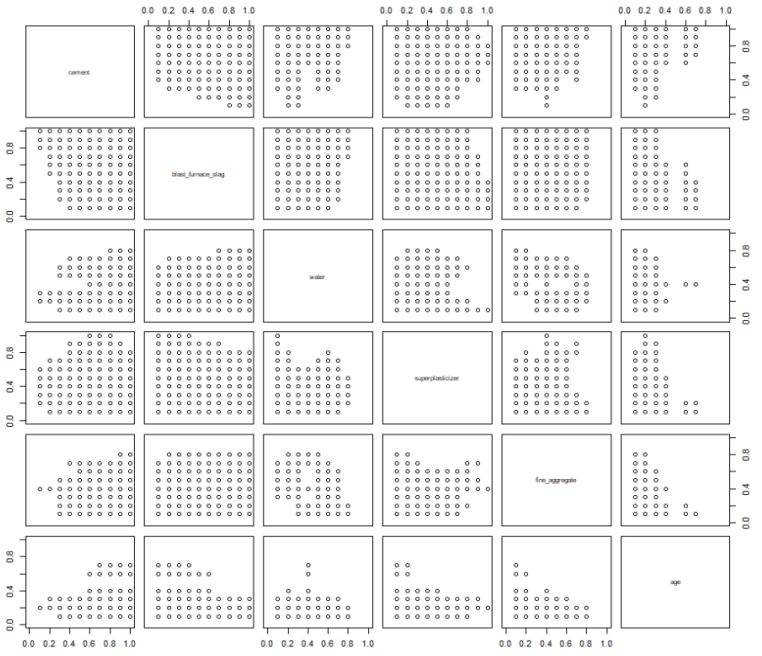
Viene quindi applicato l'algoritmo al dataset relativo al calcestruzzo avendo eliminato le variabili *fly ash* e *coarse aggregate*. Dopo aver normalizzato le covariate, si crea una griglia sullo spazio dei loro valori. In particolare avremo una griglia a sei dimensioni, con 10 punti per dimensione, per un totale di un milione di punti. A questo punto vengono fatte le previsioni nei punti della griglia secondo il GP addestrato sul dataset. Si calcola il valore di $\mu + \beta\sigma$ nei punti della griglia e si selezionano i punti che risultano essere maggiori del massimo del dataset. Si nota che per il valore di $\beta = 1$ vengono selezionati 11597 punti, mentre per $\beta = 2$ ne vengono selezionati addirittura 274735, ovvero più di un quarto dei punti della griglia. Per questa ragione non si procede con il calcolo di $\beta = 3$, bensì si opta per calcolare il caso in cui $\beta = 0.5$. Vengono quindi riportati i grafici relativi a $\beta = 0, 0.5, 1, 1.5$.



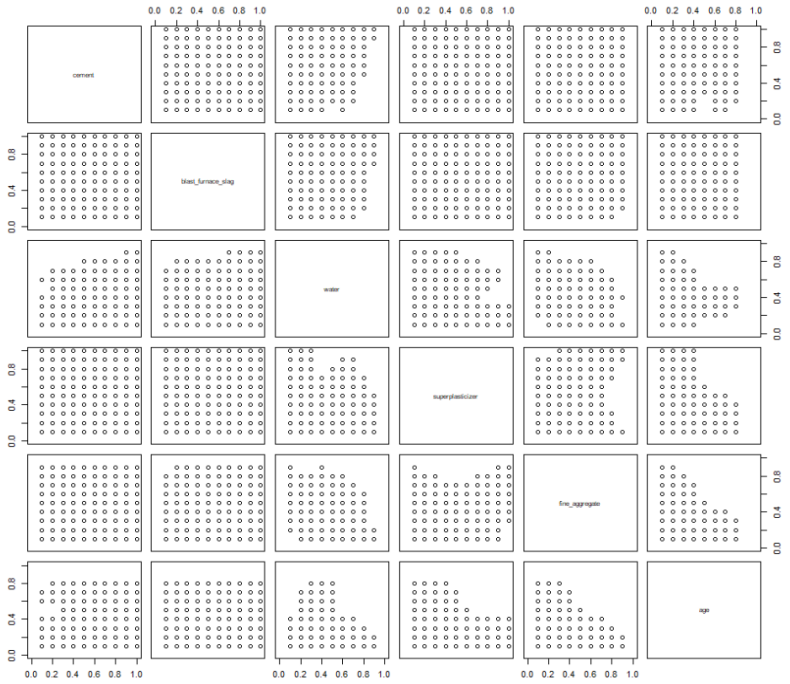
$\beta = 0$



$\beta = 0.5$



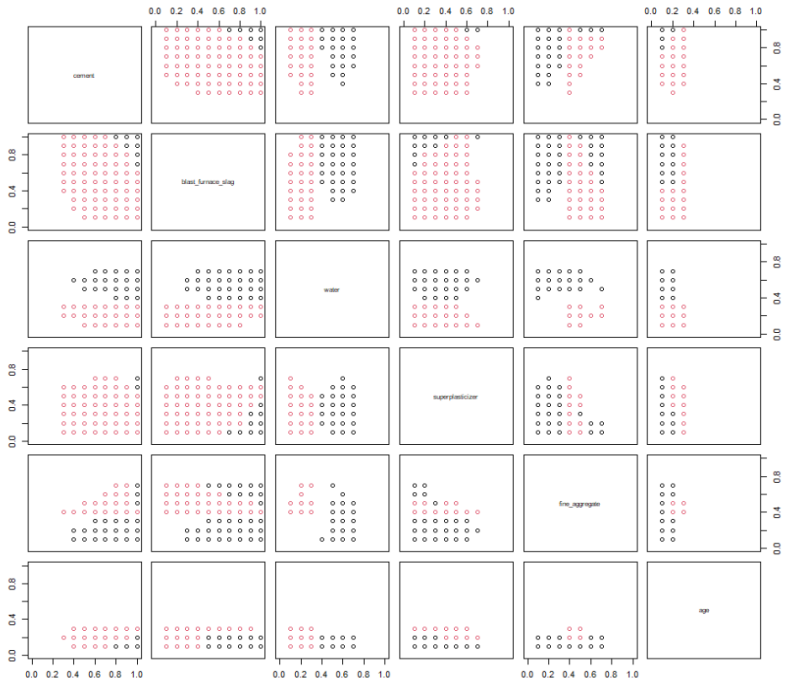
$\beta = 1$



$\beta = 1.5$

Si procede con l'applicazione del modello DBSCAN ai precedenti grafici e se ne riportano i risultati.

$\beta = 0$



$\beta = 0.5$

$\beta = 1$

Non viene riportato il grafico relativo a $\beta = 1.5$ in quanto DBSCAN individua un solo gruppo, e quindi il grafico sarebbe uguale a

quello precedente.
A livello interpretativo, notiamo che per tutti e 3 i valori di β analizzati, l'algoritmo DBSCAN individua due gruppi separati, uno costituito dai punti neri e l'altro costituito dai punti rossi. Tuttavia bisogna notare che nel grafico relativo a $\beta = 1$, in realtà vi è la presenza di un gruppo aggiuntivo rispetto a quelli predcedenti. Infatti in questo caso i due gruppi che prima erano stati identificati in maniera separata, essendo molto vicini, vengono a fondersi. Quindi quando si applicano i due metodi di sintesi, non si considera il primo gruppo identificato dal DBSCAN per $\beta = 1$.

Vengono quindi applicati i metodi di sintesi ai diversi casi al variare di β. I risultati sono riportati nelle seguenti tabelle riassuntive.

$\beta = 0$	cement	b_f_s	water	superplast	fine_aggr	age
GP1	540	323.46	196.9	12.88	673.72	37.40
media1	489.11	294.39	193.94	11.19	672.95	50.73
GP2	408.6	143.76	146.8	6.44	753.44	37.40
media2	411.58	162.6	148.5	9.08	758.81	60.50

$\beta = 0.5$	cement	b_f_s	water	superplast	fine_aggr	age
GP1	640.0	359.4	196.90	12.88	673.72	37.40
media1	484.47	285.48	194.88	11.02	684.92	54.41
GP2	452.40	107.82	146.8	6.44	753.44	73.80
media2	418.04	177.04	147.09	9.77	764.13	67.48

$\beta = 1$	cement	b_f_s	water	superplast	fine_aggr	age
GP2	540.0	35.94	171.85	3.22	633.86	219.40
media2	489.94	91.77	171.85	4.02	638.13	230.44

Sfortunatamente non possiamo verificare la resistenza relativa ai composti identificati come possibili massimi, in quanto la funzione risulta essere ignota e nell'ambito di questo lavoro non è possibile effettuare delle misure sperimentali.

7 Conclusioni

In questo elaborato è stato proposto un nuovo metodo per fare ottimizzazione off-line utilizzando i Processi Gaussiani per fare previsioni e l'algoritmo di clustering DBSCAN per raggruppare le osservazioni. In particolare l'obiettivo dell'algoritmo è quello di trovare un'osservazione, al di fuori del dataset di partenza, che massimizzi la funzione ignota.

L'algoritmo così sviluppato è stato applicato a due funzioni note, rispettivamente la funzione Branin a due dimensioni, e la funzione Hartmann a 6 dimensioni. In entrambi i casi l'algoritmo ha restituito risultati molto promettenti, riuscendo sempre a localizzare la posizione dei minimi globali delle funzioni, i quali non erano presenti all'interno del dataset utilizzato per allenare il processo gaussiano. In seguito si è mostrato come al variare del parametro β si possono ottenere risultati differenti. Inoltre nel caso della funzione Branin, abbiamo fatto variare il numero di osservazioni presenti all'interno del dataset di partenza su cui viene allenato il GP. Questo ha permesso di mostrare come una maggiore quantità di informazione sulla funzione ignota porti a delineare delle zone più ristrette. Viceversa un numero di punti ridotto, contenente quindi meno informazione sulla funzione obiettivo, porta ad avere risultati peggiori e difficili da analizzare.

In riferimento allo stato dell'arte riportato nel lavoro di rassegna [1], si sono valutati i seguenti pro e contro. Un numero elevato di variabili indipendenti risulta essere un problema per l'algoritmo. In particolare il numero di punti da analizzare risulta crescere esponenzialmente al crescere delle dimensioni delle variabili in input.
Invece un vantaggio di questo algoritmo, è che le previsioni vengono fatte solamente sui punti specificati nella griglia; per cui in questo caso non è presente il problema di ottenere ottimi al di fuori delle distribuzioni delle variabili indipendenti, problema classico in questo ambito.

Sulla base di questi risultati soddisfacenti si è deciso di applicare l'algoritmo a un dataset reale. È stato scelto un dataset riguardante il calcestruzzo e la sua resistenza. In particolare ogni osservazione rappresenta una diversa combinazione di ingredienti associata alla resistenza di tale composto. L'obiettivo dell'algoritmo è quindi trovare la combinazione di ingredienti (o composto) che massimizzi la sua resistenza. Si è svolta la selezione delle variabili attraverso le foreste casuali, le quali hanno permesso di scartare due covariate. In seguito si è applicato l'algoritmo al dataset reale e ne sono stati estratti i risultati.

Riferimenti bibliografici

[1] Brandon Trabucco & Xinyang Geng & Aviral Kumar & Sergey Levine *Design-Bench: Benchmarks for Data-Driven Offline Model-Based Optimization.* DOI: https://doi.org/10.48550/arXiv.2202.08450

[2] Aviral Kumar & Sergey Levine *Model Inversion Networks for Model-Based Optimization.* DOI: https://doi.org/10.48550/arXiv.1912.13464

[3] Williams, C. K., & Rasmussen, C. E. *Gaussian processes for machine learning* . DOI: https://doi.org/10.7551/mitpress/3206.001.0001