



**UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO**

MOE: Mixture Of Experts

Analisi della letteratura e clustering su CNEOS - Asteroids

Modellizzazione Statistica - Data Science

Ivan Diliso - 761053

Indice

1 Mixture of Experts	3
1.1 Introduzione	3
1.2 Experts	3
1.3 Gating	4
1.4 Hierarchical Mixture of Experts	4
2 Apprendimento e backpropagation	5
2.1 Modello MOE	5
2.2 Modello HMOE	5
2.3 Apprendimento	6
2.4 Funzioni di errore	6
2.4.1 Loss generiche	6
2.4.2 Loss per Gaussian MOE	6
2.4.3 Loss per Multi Layer Perceptron MOE	6
2.5 Model selection	7
2.5.1 Selezione su MOE	7
2.5.2 Selezione su HMOE	7
3 MOE e HMOE in R	7
3.1 mixtools::hmeEM	7
3.2 MEteorits	7
3.3 MoEClust	7
4 Clustering asteroidi CNEOS	8
4.1 Dataset - Asteroid NeoWs	8
4.2 Scelta e rimozione feature	9
4.3 Analisi dei dati	11
4.4 Clustering	12
4.5 Model selection	13
4.6 Valutazione del modello	14
4.7 Analisi dei risultati e conclusioni	16

1 Mixture of Experts

1.1 Introduzione

La Mixture of Experts (MOE) è una tecnica di ensemble learning di tipologia multi expert (diversi learner che lavorano in parallelo) con approccio locale (learner selection) applicabile in contesti di apprendimento supervisionato, come classificazione e regressione, e non supervisionato come clustering. Nascono nel campo delle reti neurali nell'ambito dei combining classifiers e ensemble of weak learners. I modelli MOE utilizzano la metodologia “**divide et impera**” per addestrare una serie di modelli parametrici e unire i loro output per fornire il risultato finale. Questa tipologia di modelli permette di dividere un task complesso in sottotask, addestrando i vari modelli dell'esemplice su un sottotask diverso. Nella terminologia delle MOE ogni modello addestrato nell'ensemble è chiamato **esperto**. La caratteristica fondamentale che distingue le MOE è l'utilizzo di un modello chiamato **gating** che permette di unire le soluzioni degli esperti ed assegnare ad ognuno di essi il peso che la loro soluzione avrà nel comporre la soluzione finale.

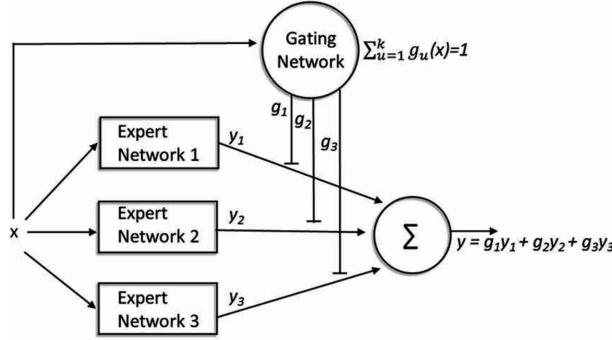


Figura 1: MOE con 3 esperti

A differenza di altre metodologie ensemble che producono esperti unbiased con stime di errore non correlate, MOE produce esperti biased con stime correlate negativamente. Con questa tipologia di modelli è necessaria una conoscenza pregressa di una divisione dei dati, è necessario quindi che il dataset sia divisibile.

1.2 Experts

Gli esperti sono i vari modelli addestrati nell'ensemble MOE, ogni esperto andrà a specializzarsi nel risolvere uno specifico sottoproblema, questo si concretizza nello specializzarsi su specifiche porzioni del feature space. I sottotask creati infatti possono essere sovrapponibili (più esperti condividono alcune feature) o non sovrapponibili (ogni esperto viene addestrato su un feature space diverso). Ogni esperto contribuisce quindi in modo diverso nella creazione dell'output finale. Nelle MOE infatti non è necessario rendere i vari learner dell'ensemble diversi in quanto naturalmente questi lavoreranno su task diversi. Il problema da risolvere è infatti trovare una divisione naturale dei dati, questo problema è risolto dall'utilizzo del modello di gating. Per il partizionamento del feature space, possono essere utilizzate due metodologie:

- **Partizionamento implicito:** Feature space viene diviso implicitamente in sottospazi tramite una funzione di errore. L'utilizzo dell'apprendimento tramite backpropagation e il partizionamento dei dati fornito dalla funzione di gating permette agli esperti di specializzarsi in diversi sottospazi del feature space. In questo caso si parla di approccio **MILE** (Mixture of the Implicitly Localised Experts)
- **Partizionamento esplicito:** Si utilizza un algoritmo di clustering per il partizionamento dei dati, questi vengono poi assegnati ad un esperto. In questo caso si parla di un approccio **MELE** (Mixture of Explicitly Localised Experts)

Un esperto può essere qualsiasi modello di apprendimento, nella formulazione originale gli esperti prendono la forma di modelli di **logistic regression**, ma possono essere anche implementati come percettroni, multi layer perceptron, gaussian mixture, reti neurali profonde (in questo caso si parla di Deep Mixture of Experts) etc. etc.

1.3 Gating

Il modello di gating può essere di diverse tipologie e assolve una funzione specifica in base alla sua tipologia. Nella formulazione originale il modello di gating è un modello che, dato in input un elemento del dataset, fornisce in output una distribuzione di probabilità sull'insieme di esperti, questa distribuzione può essere utilizzata in due modi:

- **Pooling:** L'esperto a cui è associata la probabilità più alta verrà selezionato come il modello per la predizione dei dati.
- **Combining:** Le soluzioni di tutti gli esperti vengono combinate pesate con la probabilità fornita dalla funzione di gating, ogni esperto quindi contribuisce proporzionalmente al peso generato dal gating.

Questo approccio associa quindi un diverso peso ad ogni esperto, questa tecnica può essere vista come una forma di **voting** dei modelli ensemble, dove però la capacità di voto può cambiare al variare dell'input. Dato che la distribuzione è appresa dinamicamente dal modello di gating sulla base dell'input questa apprende che input assegnare ad ogni esperto effettuando quindi un partizionamento dei dati. Questo partizionamento è definito **hard** se si utilizza una tecnica di pooling, **soft** se si utilizza il combining.

1.4 Hierarchical Mixture of Experts

Le Hierarchical Mixture of Experts (HMOE) sono una estensione dei modelli MOE in cui ogni esperto viene rimpiazzato con un sistema completo MOE in modo ricorsivo creando una **struttura ad albero**. Nell'albero HMOE i nodi interni saranno le varie funzioni di gating sviluppate in modo gerarchico, e i nodi foglia conterranno gli esperti. Può essere interpretato come un albero di decisione in cui i nodi interni (gating) fungono da nodi di decisione e la decisione finale è un singolo nodo foglia (in caso di pooling) o una combinazione pesata dei nodi foglia (in caso di combining). Se nei modelli MOE un parametro importante per l'ottimizzazione del modello è il numero di esperti da utilizzare in questo caso il parametro è la profondità della ricorsione, che determinerà il numero di esperti utilizzati.

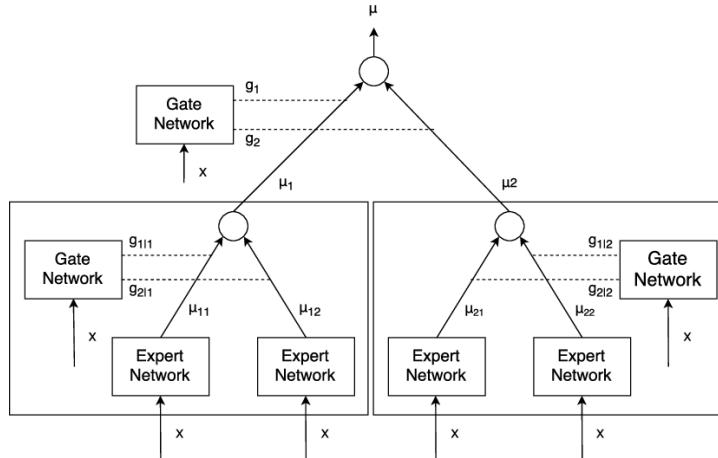


Figura 2: HMOE con profondità 2

Questa tipologia di albero, con gating di tipo combining, viene definita **soft decision tree** in quanto a differenza degli alberi di decisione hard, vengono utilizzate le distribuzioni di probabilità generate dai nodi gating per esplorare tutti i possibili percorsi dell'albero, per poi effettuare una somma pesata a livello di foglie, dove ogni peso è uguale al prodotto dei valori di gating di ogni path per arrivare alla foglia. A differenza dei modelli CART (Classification And Regression Trees) dove ogni nodo contiene un valore costante per la decisione, nei modelli HMO ogni nodo implementa un modello lineare (o di regressione logistica). L'uso di questa formulazione soft permette di catturare situazioni in cui la transizione da una risposta alta ad una bassa è graduale.

2 Apprendimento e backpropagation

2.1 Modello MOE

Sia $x \in \mathbb{R}^n$ vettore di input, sia T il numero di esperti modello, h_1, \dots, h_T gli esperti del modello e y variabile target. Dati W_i parametri dell'i-esimo esperto, l'obiettivo del modello è approssimare la distribuzione di y sulla base dei dati di training:

$$h_i(y|x; W_i)$$

La funzione di gating produce un set di coefficienti che pesano il contributo degli esperti, sia v_i vettore dei pesi della funzione di gating relativa all'i-esimo esperto e sia α insieme dei parametri del modello di gating (insieme dei pesi relativi ad ogni esperto). Essendo una distribuzione di probabilità, la somma dei coefficienti su tutti gli esperti dovrà essere uguale a 1. Definiamo quindi il set di coefficienti generati dalla funzione di gating, sull'i-esimo esperto, dato l'insieme di parametri α come:

$$\pi_i(x; \alpha) : \sum_{i=1}^T \pi_i(x; \alpha) = 1$$

Sulla base di queste probabilità partizioniamo lo spazio di input, diverse partizioni appartengono a diversi esperti. L'output del modello, dato ψ insieme dei parametri degli esperti e del modello di gating, sarà quindi:

$$H(y|x; \psi) = \sum_{i=1}^T \pi_i(x; \alpha) \cdot h_i(y|x; W_i)$$

Nella fase di training il valore $\pi_i(x; \alpha)$ indica la probabilità che l'istanza x appaia nel training set dell'i-esimo esperto. Mentre nella fase di testing definisce il contributo che il modello h_i fornisce alla predizione finale. L'output della funzione di gating può essere espresso tramite una softmax (utilizzata sia in casi di regressione che di classificazione):

$$\pi_i(x; \alpha) = \frac{e^{v_i} x}{\sum_{l=1}^k e^{v_l} x}$$

2.2 Modello HMOE

Seguendo il formalismo definito in precedenza definiamo un modello HMOE di profondità 2. Definito I numero di nodi connessi al nodo gating al livello di radice, J_i numero di nodi connessi all'i-esimo nodo gating, π_i output del nodo gating al livello di radice e $\pi_{j|i}$ output dell>j-esimo nodo gating connesso all'i-esimo nodo gating, l'output del modello HMOE sarà:

$$H(y|x; \psi) = \sum_{i=1}^I \pi_i(x; \alpha_{\pi_i}) \cdot \sum_{j=1}^{J_i} \pi_{j|i}(x; \alpha_{\pi_{j|i}}) h_{ij}(y|x; W_{ij})$$

Nella fase di apprendimento e testing i dati sono forniti in input agli esperti che producono dei vettori di output che procedono nell'albero verso l'alto seguendo la gerarchia fino al nodo gating radice, (in direzione opposta rispetto ai modelli CART). Gli output di ogni nodo vengono moltiplicati tra loro e sommati seguendo i vari livelli dell'albero.

2.3 Apprendimento

L'aggiornamento dei pesi avviene sulla base dell'output del modello di gating, questo infatti alloca in maniera hard o soft i dati di training a uno o più esperti del modello, e in base all'errore sull'output l'aggiornamento dei pesi viene localizzato solo sugli esperti utilizzati. Si parla di aggiornamento locale in quanto i pesi di degli esperti sono disaccoppiati. In base alla tipologia di esperto e gating scelta l'aggiornamento dei pesi può avvenire tramite:

- **Discesa del gradiente** Particolarmente utile con i **mixutre of multi layer perceptron experts**. Utilizzando questa funzione l'addestramento tende ad assegnare un dato di training ad ogni esperto.
- **Expectation Maximization** Modelli MOE cercano di risolvere due task: dato l'insieme di esperti trovare la funzione di gating ottimale per minimizzare l'errore, e, data la funzione di gating, addestrare ogni esperto a massimizzare le performance sulla distribuzione assegnata dalla funzione di gating. Questo rende naturale l'utilizzo di un algoritmo di expectation maximization.

2.4 Funzioni di errore

Per semplificare la lettura delle loss, definiamo g_i output della funzione di gating per il i -esimo esperto e con O_i output dell' i -esimo esperto:

$$g_i = \pi_i(x; \alpha) , \quad O_i = h_i(y|x; W_i)$$

2.4.1 Loss generiche

Una prima misura di errore utilizzabile è:

$$E_{\text{ensemble}} = \|y - \sum_i g_i O_i\|^2$$

In questo caso i pesi di ogni esperto sono aggiornati sulla base di un errore ensemble totale. Questo permette un alto livello di cooperazione e tende a sfruttare quasi tutti gli esperti del modello (nessun esperto non contribuisce al problema). In questa funzione di errore si assume che l'output del sistema sia una combinazione lineare degli output degli esperti locali, con il gating che determina la proporzione. Questo porta ad un forte accoppiamento dei pesi degli esperti. Una ulteriore misura di errore permette di aggiornare i pesi sulla base dell'errore del singolo esperto, non viene assicurata in questo caso però la localizzazione degli esperti

$$E_{\text{single}} = \sum_i g_i \|y - O_i\|^2$$

2.4.2 Loss per Gaussian MOE

Una misura di errore che tiene conto di entrambi i fattori è basata sulla negative log probability di generare l'output vector desiderato. Si assume una mixture di modelli gaussiani con Σ matrice di covarianza:

$$E_{GMOE} = -\log \sum_i g_i e^{-\frac{1}{2}(y-O_i)^T \Sigma^{-1}(y-O_i)}$$

L'apprendimento di ogni esperto avviene sull'errore individuale, ma l'aggiornamento dei pesi per ogni esperto è proporzionale al suo rateo di errore sull'errore totale. Questo permette la localizzazione degli esperti nei corrispondenti sottospazi delle feature.

2.4.3 Loss per Multi Layer Perceptron MOE

Per questa loss si considerano esperti in forma di multi layer perceptron con un hidden layer che produce un output O_i in funzione dell'input con funzione di attivazione sigmoidale. L'apprendimento avviene tramite backpropagation massimizzando la log likelihood dei dati di training dati i parametri del modello.

2.5 Model selection

2.5.1 Selezione su MOE

Nei modelli MOE definita la forma degli esperti e la tipologia di modello di gating l'iperparametro da ottimizzare per la model selection è il numero di esperti da generare, questo è particolarmente importante nell'applicazione del clustering in quanto il numero di esperti equivale al numero di cluster che verranno generati dal modello. Nella selezione del modello ottimale è possibile addestrare gli esperti o la funzione di gating esplicitamente su diversi sottospazi delle feature, in modo da trovare la combinazione che minimizzi l'errore.

2.5.2 Selezione su HMOE

Nei modelli HMOE ci sono ulteriori parametri da ottimizzare quali la profondità e il numero di connessioni per ogni nodo. La model selection su questo tipo di modelli è simile alle metodologie applicate ai modelli CART, in cui viene modificata la funzione di valutazione di un branch dell'albero. Possibili metodologie sono:

- **Crescita:** Partendo da un albero con un solo layer si procede aumentando la profondità o il numero di connessioni.
- **Potatura:** Generato un albero ad alta profondità vengono analizzate le distribuzioni di probabilità sui vari percorsi in modo da eliminare (potatura) i percorsi meno utilizzati. L'obiettivo è la riduzione dei requisiti computazionali.

3 MOE e HMOE in R

3.1 mixtools::hmeEM

Il package mixtools permette di analizzare mixture model finiti in contesti parametrici e semiparametrici. Nello specifico, la classe `hmeEM` implementa un modello HMOE per la regressione addestrato tramite algoritmo EM. Ad oggi (Febbraio 2023) la libreria non permette di creare alberi di profondità maggiore di 2 ma verranno implementati in versioni successive. Non è possibile modificare la tipologia di esperti o di gating. Utilizza un modello di gating softmax tramite combining mentre gli esperti sono modelli lineari generalizzati (GLM, generalizzazione della regressione classica che permette di modellare la relazione tra le variabili anche quando la distribuzione della variabile dipendente non è gaussiana o in casi di non linearità). Non sono supportati in maniera efficiente dataset ad alta dimensionalità.

3.2 MEteorits

Il package MEteorits (Mixtures-of-ExperTs modEling for cOmplex and non-noRmal dIsTributions) contiene diverse implementazioni di mixture of experts per classificazione e clustering di dati in situazioni di distribuzioni normali, skewed, e con dati corrotti o osservazioni outlier. Tutti i modelli presenti sono implementati utilizzando regressione polinomiale per gli esperti e regressione logistica per il modello di gating, con possibilità di specificare il numero di esperti da utilizzare. Il principale svantaggio di questa libreria è che non permette di utilizzare dataset con dimensionalità maggiore di due, riducendo di molto l'applicabilità del modello a dataset reali. I modelli implementati sono:

- `emNMoE` / `emSNMoE`: Per distribuzioni normali con algoritmo EM o normali skewed con algoritmo ECM (Expectation Conditional Maximization).
- `emStMoE` / `emTMoE`: Per distribuzioni di student o student skewed con algoritmo ECM.

3.3 MoEClust

La libreria MoEClust(Gaussian Parsimonious Clustering Models with Gating and Expert Network Covariates and a Noise Component) permette di addestrare mixture of expert gaussiane tramite algoritmo di ottimizzazione expectation maximization, l'utilizzo principale della libreria è il clustering.

La particolarità di questa libreria è la possibilità di fornire diversi sottoinsiemi di feature ai modelli esperti e al modello di gating, implementando anche una funzione di model selection che permette di selezionare il modello migliore sulla base delle feature in input e del numero di esperti da generare. La libreria contiene anche tool per il clustering in contesti di dati molto rumorosi. Le principali funzioni sono:

- **MoE_clust**: Addestramento del modello tramite algoritmo EM/ECM. Il modello utilizzato è Gaussian Parsimonious Clustering Model (GPCM). Tramite le formule è possibile indicare i sottoinsiemi di feature da fornire a **gating** e **expert**.
- **MoE_stepwise**: Ricerca stepwise greedy per la model selection del modello migliore. Gli iperparametri ottimizzati sono il numero di esperti da generare e subset delle feature da utilizzare per esperti e gating.
- **MoE_compare**: Confronto di diversi modelli MoE_Clust
- **MoE_gpairs**: Visualizzazione della relazione pairwise delle variabili e dei risultati del clustering.
- **MoEClust::predict**: Predizione del cluster su nuovi dati

Maggiori dettagli su questa libreria sono forniti nel capitolo del clustering, dove questa viene utilizzata nell'esecuzione del progetto.

4 Clustering asteroidi CNEOS

L'obiettivo del progetto è effettuare clustering e analisi dei dati tramite modelli mixture of experts sui dati degli asteroidi potenzialmente pericolosi orbitanti la terra. Si cerca di ottenere informazioni su possibili raggruppamenti degli asteroidi, in modo da fornire un tool per aiutare a classificare gli asteroidi in base alle loro caratteristiche orbitali e pericolosità. L'assegnazione di un gruppo di riferimento può aiutare non solo a raggruppare asteroidi con caratteristiche simili ma anche a predire la loro possibile pericolosità.

4.1 Dataset - Asteroid NeoWs

Il dataset utilizzato è una raccolta di dati sugli asteroidi che si sono avvicinati o si avvicineranno all'orbita terrestre effettuata dal laboratorio NASA “Center for Near Earth Object Studies (**CNEOS**)”. Le informazioni vengono raccolte tramite **NeoWs** (NearEarthObject Web-Service) un servizio di API web che permette l'esecuzione di query per la ricerca e filtering degli asteroidi, il servizio fornisce anche un dataset con lo storico di tutti gli asteroidi che si sono avvicinati all'orbita terrestre o che lo faranno nel futuro.

```
# Caricamento dataset
nasa <- read_csv("data/nasa.csv")

# Feature del dataset
names(nasa)

## [1] "Neo Reference ID"                      "Name"
## [3] "Absolute Magnitude"                    "Est Dia in KM(min)"
## [5] "Est Dia in KM(max)"                   "Est Dia in M(min)"
## [7] "Est Dia in M(max)"                    "Est Dia in Miles(min)"
## [9] "Est Dia in Miles(max)"                 "Est Dia in Feet(min)"
## [11] "Est Dia in Feet(max)"                  "Close Approach Date"
## [13] "Epoch Date Close Approach"            "Relative Velocity km per sec"
## [15] "Relative Velocity km per hr"          "Miles per hour"
## [17] "Miss Dist.(Astronomical)"             "Miss Dist.(lunar)"
## [19] "Miss Dist.(kilometers)"                "Miss Dist.(miles)"
## [21] "Orbiting Body"                        "Orbit ID"
## [23] "Orbit Determination Date"             "Orbit Uncertainty"
## [25] "Minimum Orbit Intersection"           "Jupiter Tisserand Invariant"
## [27] "Epoch Osculation"                    "Eccentricity"
## [29] "Semi Major Axis"                     "Inclination"
```

```

## [31] "Asc Node Longitude"           "Orbital Period"
## [33] "Perihelion Distance"        "Perihelion Arg"
## [35] "Aphelion Dist"              "Perihelion Time"
## [37] "Mean Anomaly"                "Mean Motion"
## [39] "Equinox"                     "Hazardous"

```

Il dataset presenta 4687 datapoints ognuno descritto da 40 feature. Molte di queste feature sono coefficienti utilizzabili nelle equazioni di moto per predire il movimento e traiettoria dell'asteroide, alcune sono infatti facilmente ricostruibili a partire da altre feature presenti nel dataset. Sono presenti inoltre numerose feature ripetute con unità di misura diverse, risulta quindi necessaria una fase di rimozione e scelta delle feature più importanti.

4.2 Scelta e rimozione feature

Sulla base delle motivazioni descritte in precedenza, per migliorare la visualizzazione dei dati, dei risultati del clustering e velocizzare il processo di apprendimento vengono selezionate 13 feature dall'insieme di 40 feature del dataset origianale. Esempi di feature rimosse sono “Neo Reference ID” e “Name” che rappresentano l’ID dell’astroide, la stima del diametro massima e minima in miglia, metri e piedi, lasciando solo l’informazione in chilometri. Vengono inoltre rimosse le informazioni riguardanti la data, in quanto non utili al clustering (ci interessano informazioni su orbita e pericolosità) e rimosse tutte le feature che presentano lo stesso valore per tutti i datapoint, come “Equinox” e “Orbiting Planet”.

```

# Funzione per la visualizzazione dei dati
custom_plot_ggpairs <- function(dataset, columns){
  g <- ggpairs(data=dataset,
                columns = columns,
                mapping = ggplot2::aes(colour = as.character(Hazardous), alpha=0.3)) +
    scale_color_manual(values = c("cyan4", "darkorange1")) +
    scale_fill_manual(values = c("cyan4", "darkorange1"))
  g
}

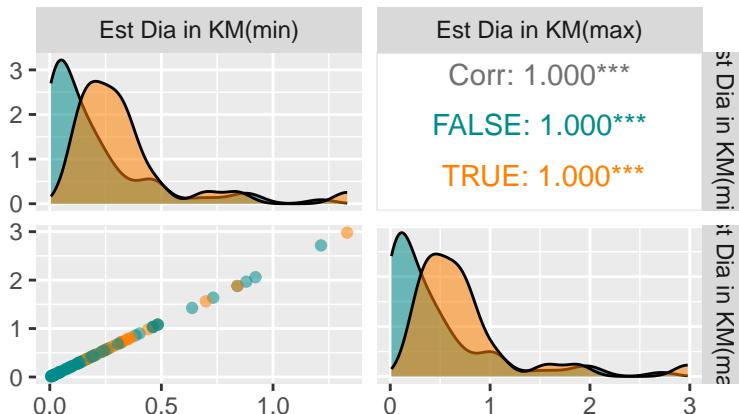
```

Da una prima analisi della correlazione delle feature (la matrice di correlazione è troppo grande per essere visualizzata su questo documento, può essere consultata nella cartella `images\corr_plot.png`) si nota che le variabili di stima del diametro minima e massima hanno indice di correlazione pari a 1. Si è deciso di mantenere solo una delle due feature, la stima di distanza massima. Nei grafici a seguire il valore false e il suo colore associato (cyan) indica un asteroide non pericoloso, il valore true (orange) ne indica invece uno pericoloso.

```

# Correlazione tra stima di distanza minima e massima
custom_plot_ggpairs(nasa[1:100,], c(4,5))

```



```

# Rimozione feature e creazione dataset finale
dataset <- nasa[, - (c(1, 2, 4, (6:11), (12:13), (15:16), (18:25),
27, 31, 34, 36, 38, 39)), drop=FALSE]
names(dataset) <- make.names(names(dataset))
dataset$Hazardous <- as.numeric(as.logical(dataset$Hazardous))

```

L'insieme delle feature scelte è il seguente:

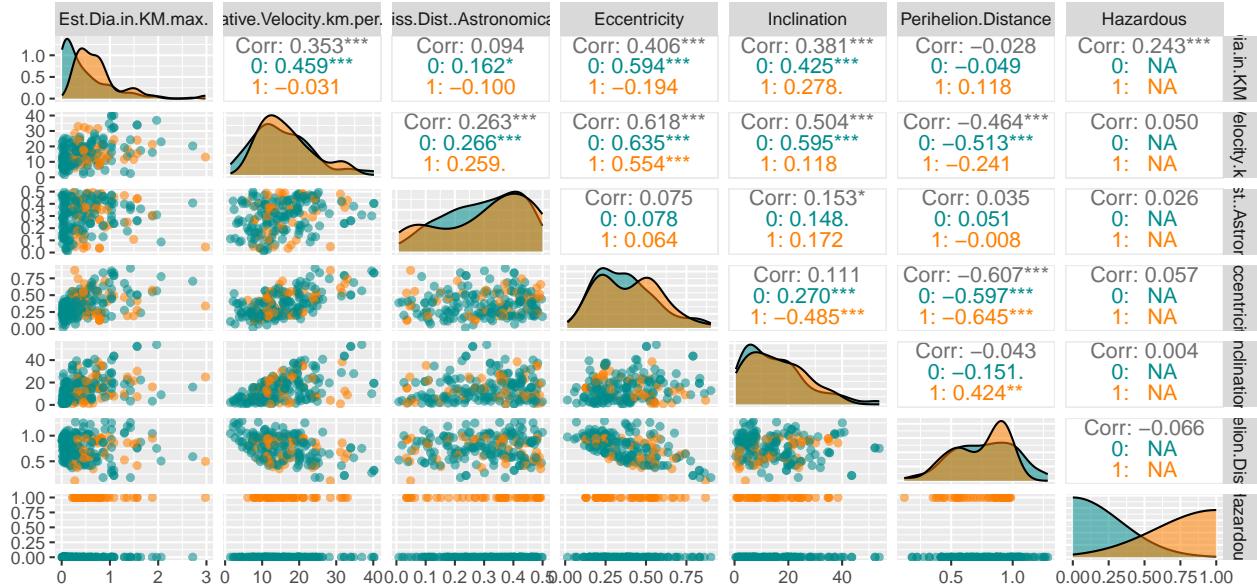
- **Absolute.Magnitude:** Misura della luminosità assoluta. Indica quanto luminoso sarebbe l'asteroide se fosse posizionato ad una distanza di 1 unità astronomica dal sole e dalla terra e riflettendo il 100% della luce proiettata su esso.
- **Est.Dia.in.KM.max:** Stima del diametro massimo in Km.
- **Relative.Velocity.km.per.sec:** Velocità relativa in Km/s.
- **Miss.Dist..Astronomical:** Distanza dalla terra in unità astronomica cioè la distanza media tra il sole e la Terra (circa 149.6 milioni di chilometri).
- **Jupiter.Tisserand.Invariant:** Unità per descrivere l'orbita dell'asteroide nel sistema solare rispetto all'orbita di Giove.
- **Eccentricity:** Unità per descrivere la forma dell'orbita dell'asteroide attorno al sole. Indica quanto è allungata o appiattita l'ellisse dell'orbita dell'asteroide.
- **Semi.Major.Axis:** Lunghezza del semi asse maggiore dell'asteroide
- **Inclination:** Angolo tra il piano dell'orbita e il piano dell'ellittica (orbita della Terra attorno al sole). Questa feature viene utilizzata per raggruppare gli asteroidi in famiglie orbitali in base a caratteristiche orbitali simili. Misura di come l'asteroide andrà ad interagire con altri corpi nello spazio in base alla sua orbita.
- **Orbital.Period:** Tempo necessario per completare un orbita completa attorno alla Terra.
- **Perihelion.Distance:** Punto dell'orbita in cui l'asteroide si avvicina di più al sole.
- **Aphelion.Dist:** Punto dell'orbita in cui l'asteroide si allontana di più dal sole.
- **Mean Anomaly::** Posizione dell'asteroide lungo la sua orbita, definita come l'angolo tra il perielio e la posizione dell'oggetto rispetto ad un punto di riferimento immaginario chiamato "punto vernal".
- **Hazardous:** Indica se l'asteroide è pericoloso o meno.

AB.MAG	EST.DIA	REL.VEL	MISS.DIST	TISS	ECC
21.6	0.2844723	6.115834	0.4194825	4.634	0.4255491
21.3	0.3266179	18.113985	0.3830145	5.457	0.3516743
20.3	0.5176545	7.590711	0.0509560	4.557	0.3482483
27.4	0.0196807	11.173875	0.2853223	5.093	0.2165783
21.6	0.2844723	9.840831	0.4078322	5.154	0.2104479
19.6	0.7145621	10.808844	0.3927848	4.724	0.5634411

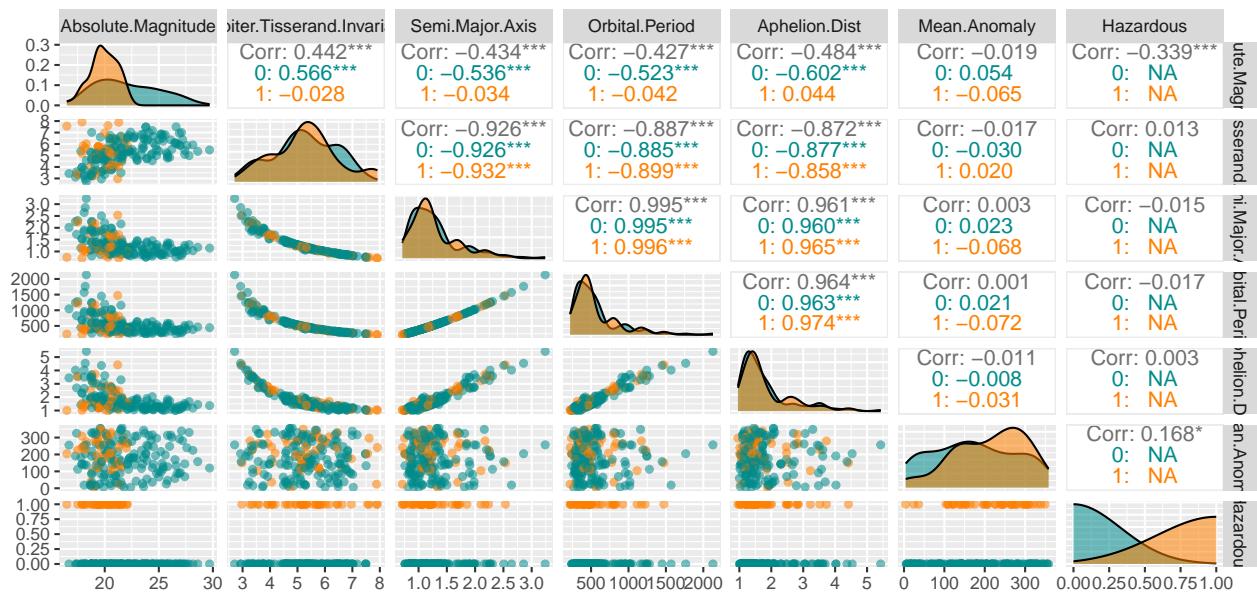
MAGAX	INC	ORB.PER	PER.DIS	APH.DIS	ANOM	HAZ
1.407011	6.025981	609.5998	0.8082589	2.005764	264.83753	1
1.107776	28.412996	425.8693	0.7181996	1.497352	173.74111	0
1.458824	4.237961	643.5802	0.9507910	1.966857	292.89365	1
1.255903	7.905894	514.0821	0.9839016	1.527904	68.74101	0
1.225615	16.793382	495.5978	0.9676866	1.483543	135.14213	1
1.323532	17.927751	556.1606	0.5777998	2.069265	354.23737	0

4.3 Analisi dei dati

Nella prima fase di sviluppo, prima di procedere alla fase di clustering eseguiamo un'analisi della correlazione delle feature, questa fase ha come obiettivo individuare quelle feature che fungono da indicatore per la pericolosità dell'asteroide. Per facilitare la visualizzazione dei dati vengono mostrate le correlazioni prima delle variabili debolmente correlate, e successivamente le variabili fortemente correlate tra loro. Le immagini complete contenenti le correlazioni per l'intero set di feature e sull'intero dataset sono presenti in `images\corr_plot_full.png`



Da questo primo grafico non è possibile notare nessuna feature potenzialmente utilizzabile come feature preferenziale per la funzione di gating o esperto. Nel secondo grafico invece, notiamo una tendenza negli asteroidi classificati come pericolosi ad avere valori bassi di grandezza assoluta e picchi più alti di anomalia media, gli asteroidi non pericolosi invece tendono a concentrarsi su valori più bassi di periodo orbitale e distanza dal perihelion e aphelion, con valori alti della costante di Tisserand.



4.4 Clustering

Nella fase di clustering verrà utilizzata la libreria MoEClust per eseguire il clustering degli asteroidi. Non è conosciuto il numero di cluster a priori quindi verrà utilizzata la funzione MoE_stepwise per effettuare una ottimizzazione di questo parametro. Sulla base della conoscenza acquisita nella fase di analisi vengono anche definiti 4 ulteriori modelli sulla base delle feature fornite al modello di gating:

- model_1 Hazardous, Absolute.Magnitude e Orbital.Period
- model_2 Hazardous, Absolute.Magnitude, e Jupiter.Tisserand.Invariant
- model_3 Absolute.Magnitude, Jupiter.Tisserand.Invariant, e Orbital.Period
- model_4 Hazardous e Absolute.Magnitude

Gli esperti verranno addestrati su tutte le varibili del modello, in quanto la divisione del feature space avverrà implicitamente tramite il modello di gating. Si fornisce alla maggior parte dei modelli custom la feature Hazardous in quanto si vuole suggerire una divisione con particolare attenzione alla pericolosità dell'asteroide. A causa delle limitazioni computazionali disponibili è stato selezionato un sample di 1500 datapoints a partire dal dataset per l'esecuzione dell'algoritmo.

```
# Definizione dei modelli e addestramento
sample <- sample(1:nrow(dataset), replace=TRUE, size=1500)
clustering_dataset <- dataset[sample,]

model_1 <- MoE_clust(data = clustering_dataset,
                      G=2:4,
                      equalPro = FALSE,
                      gating = ~ Hazardous + Absolute.Magnitude + Orbital.Period,
                      network.data = clustering_dataset)

model_2 <- MoE_clust(data = clustering_dataset,
                      G=2:4,
                      equalPro = FALSE,
                      gating = ~ Hazardous + Absolute.Magnitude + Jupiter.Tisserand.Invariant,
                      network.data = clustering_dataset)

model_3 <- MoE_clust(data = clustering_dataset,
                      G=2:4,
                      equalPro = FALSE,
                      gating = ~ Absolute.Magnitude + Orbital.Period +
                        Jupiter.Tisserand.Invariant,
                      network.data = clustering_dataset)

model_4 <- MoE_clust(data = clustering_dataset,
                      G=2:4,
                      equalPro = FALSE,
                      gating = ~ Hazardous + Absolute.Magnitude,
                      network.data = clustering_dataset)

model <- MoE_clust(data = clustering_dataset,
                     G=2:4,
                     equalPro = FALSE)

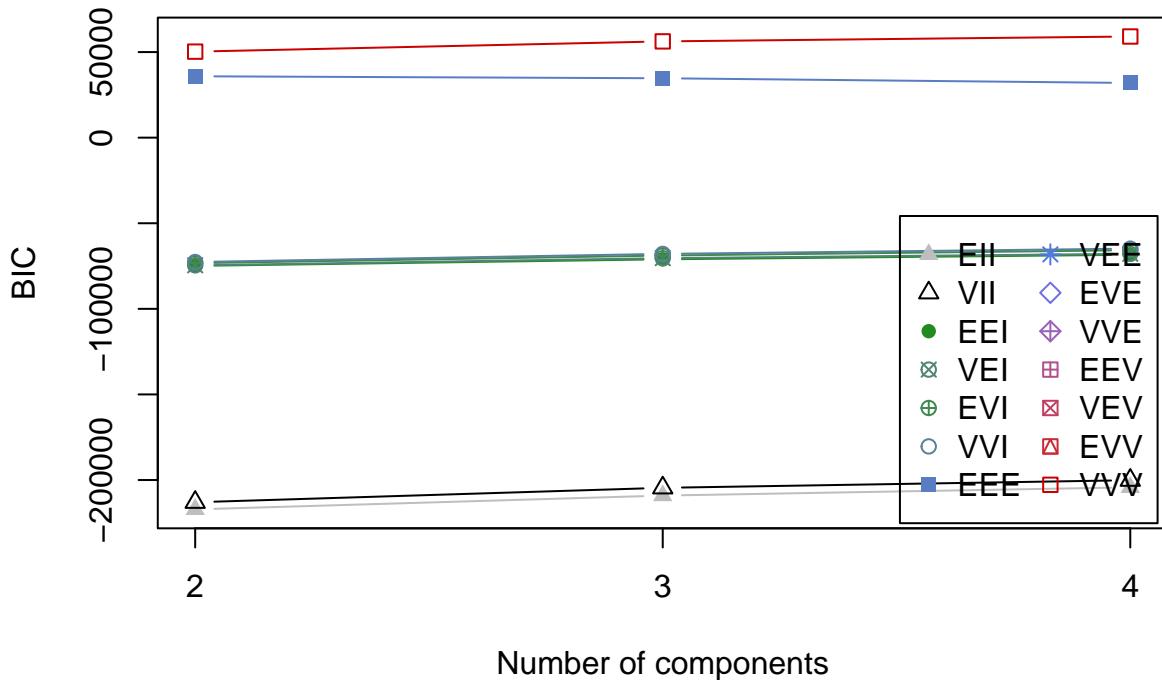
# Ricerca greedy numero di cluster migliore a partire da un modello basilare
(optimized_models <- MoE_stepwise(clustering_dataset, initialG = 2, initialModel = model))
```

4.5 Model selection

Una volta addestrati i modelli viene scelto il modello migliore sulla base del parametro “Baesian Information Criterion” (BIC), un criterio di selezione che preferisce modelli con pochi parametri e più facilmente interpretabili. Questa misura infatti tiene conto sia della precisione del modello sia del numero di parametri utilizzati. In generale si cerca di minimizzare il valore di BIC, ma la libreria MoEClust utilizza la definizione di BIC della libreria mclust, che lo definisce come il negativo del valore BIC originale. Verrà selezionato quindi il modello con il valore BIC più alto.

```
# Confronto tra modelli custom e modelli addestrati tramite ricerca greedy
custom <- MoE_compare(model_1, model_2, model_3, model_4, optimal.only = FALSE)
stepwise <- MoE_compare(optimized_models, optimal.only = FALSE)

plot(MoE_compare(custom, stepwise, optimal.only = FALSE), what="criterion")
```



```
# Scelta del modello ottimale
best <- MoE_compare(custom, stepwise, optimal.only = FALSE)$optimal
summary(best)

## Best model occurs at the max of the number of components considered
## -----
## Gaussian Parsimonious Clustering Model with Covariates
## Data: clustering_dataset
## -----
## MoEClust: VVV (ellipsoidal, varying volume, shape, and orientation), with 4 components
## 
## Gating Network Covariates: ~Hazardous + Absolute.Magnitude + Jupiter.Tisserand.Invariant
## Expert Network Covariates: None
## Noise Component: FALSE
## 
## log.likelihood    n   d   df iters      BIC       ICL       AIC Algo
##          31109.51 1500 13 428      2 59088.95 59052.01 61363.01   EM
```

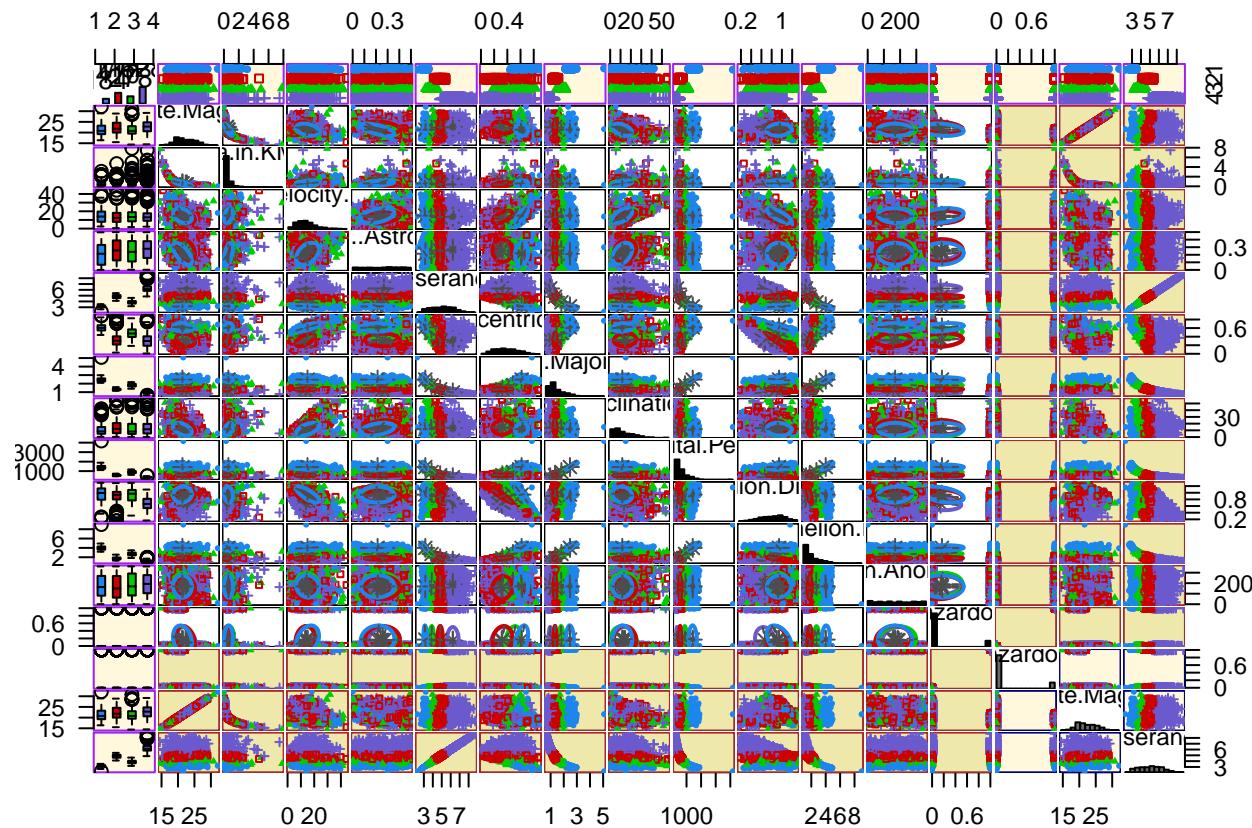
```
##  
## Clustering table :  
##   1   2   3   4  
## 184 410 278 628
```

Confrontati i modelli, scegliamo quello migliore che risulta essere un modello custom descritto nell'output del blocco precedente. Possiamo notare che il modello migliore utilizza le variabili `Hazardous`, `Absolute.Magnitude`, e `Jupyter.Tisserand.Invariant`. La fase di analisi è risultata quindi utile nel generare un modello più adeguato per il clustering dei dati.

4.6 Valutazione del modello

Visualizziamo quindi nuovamente tutte le coppie di dati, mostrando inoltre il cluster associato ad ognuno dei data point. L'immagine a più alta risoluzione è presente in `images\cluster_result.png`.

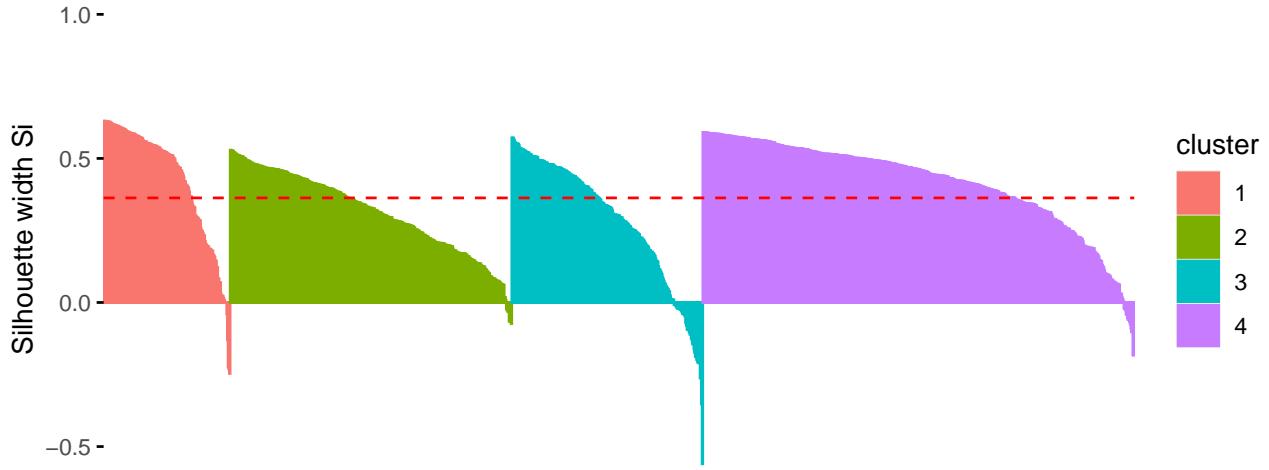
```
# Risultati del clustering su coppie di feature  
plot(best, what="gpairs")
```



Andiamo quindi a valutare la bontà del clustering mostrando il valore di Silhouette per ogni datapoint e per ogni cluster. Dalla visualizzazione della Silhouette possiamo notare dei valori superiori alla media per quasi tutti i data point del cluster 1 e del cluster 4, con risultati peggiori e valori negativi per i cluster 2 e 3.

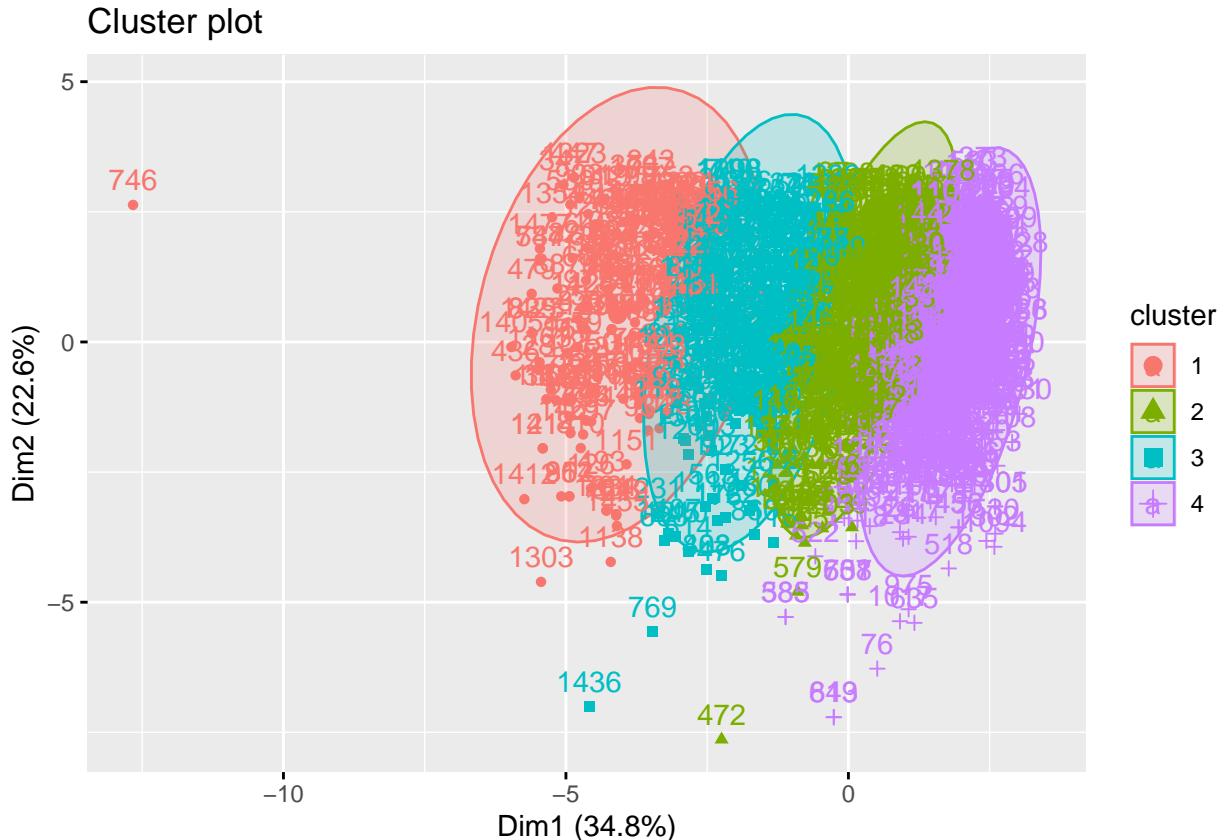
```
# Calcolo e visualizzazione Silhouette
sil <- silhouette(predict(best, clustering_dataset)$classification,
                    dist(clustering_dataset))
fviz_silhouette(sil)
```

Clusters silhouette plot
Average silhouette width: 0.36



Per avere una migliore idea della divisione fatta dall'algoritmo sui dati e per dare una più precisa interpretazione ai risultati dell'analisi tramite Silhouette viene applicata una riduzione di dimensionalità tramite PCA e visualizzati i cluster in due dimensioni. Possiamo infatti notare un maggiore mixing dei dati dei cluster tra i cluster 2 e 3, centrali nel grafico.

```
# Visualizzazione clustering
data <- list(data = clustering_dataset, cluster=predict(best, clustering_dataset)$classification)
fviz_cluster(data, ellipse.type = "norm")
```



4.7 Analisi dei risultati e conclusioni

Infine cerchiamo di illustrare le differenze principali tra i cluster, mostrando le disparità nella media del valore delle feature per ogni cluster. Andando ad analizzare i valori dei vari cluster scopriamo le seguenti particolarità:

- CLUSTER 1: Gli asteroidi appartenenti a questo cluster tendono ad essere di grandi dimensioni (alti valori del diametro), non provengono dalla cintura principale degli asteroidi ma da altre regioni del sistema solare (costante di tisserand mediamente bassa), hanno un'orbita molto allungata e lontana dal sole (eccentricità alta), e possono essere molto distanti e freddi (distanza da aphelion). Tutte queste variabili possono far pensare che l'asteroide provenga da un punto dello spazio chiamato "**nube di Oort**"
- CLUSTER 2: Asteroidi piccoli, provenienti dalla cintura principale, con un'orbita circolare intorno al sole e una distanza molto vicina alla Terra. Gli oggetti di questo tipo possono essere di tipologia **Apollo** o **Aten**, tipologia di classificazione degli asteroidi in base alla loro orbita.
- CLUSTER 4: Gli asteroidi di questo cluster hanno caratteristiche simili a quelle degli asteroidi del primo cluster, ma a differenza di quest'ultimo sono molto più vicini all'orbita terrestre e con un'orbita circolare e vicina al sole. Gli oggetti di questo tipo possono essere di tipologia **Atira**, sottocategoria di asteroidi Aten.

I risultati del clustering sono quindi precisi nel raggruppare gli asteroidi in gruppi che corrispondono ad una classificazione effettivamente utilizzata nel loro campo di studio. Questo tipo di risultato può essere utile non solo per effettuare un filtering iniziale degli asteroidi ma anche per effettuare predizioni sulla probabile tipologia dell'asteroide anche in contesti di informazioni mancanti. La libreria MoEClust infatti permette di effettuare predizioni anche in caso di feature mancanti.

Un possibile sviluppo futuro potrebbe essere il confronto dei risultati ottenuti con algoritmi di clustering basati su distanza oppure effettuare classificazione della pericolosità degli asteroidi utilizzando come informazione anche la presenza dell'asteroide in uno dei cluster trovati.

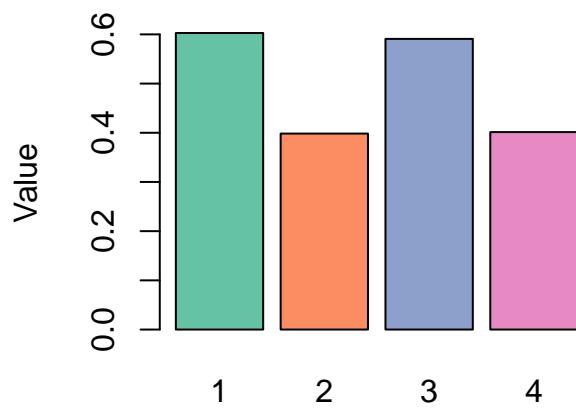
```
# Visualizzazione barplot delle medie per ogni feature e cluster
clusters <- factor(predict(best, clustering_dataset)$classification)
clusters_points <- split(clustering_dataset, clusters)
averages <- list()

for (i in clusters_points){
  averages <- rbind(averages, colMeans(i))
}

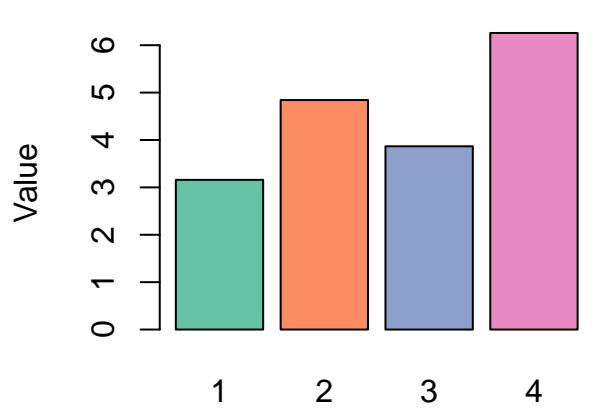
df <- data.frame(averages)
c <- 0

for (i in c(2,5,6,7,9,11)){
  barplot(as.numeric(averages[, i]), main = names(df)[i],
         names.arg = levels(clusters),
         xlab = "Cluster",
         ylab = "Value",
         col=brewer.pal(5, "Set2"),
         horiz=FALSE
        )
  c <- c+1
  if((c%%2) == 0){cat("<br /><br />")}
}
```

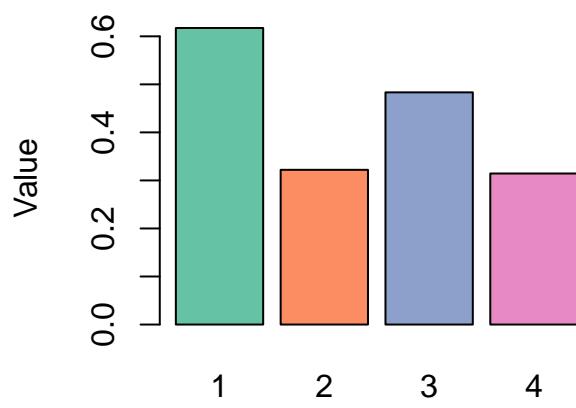
Est.Dia.in.KM.max.



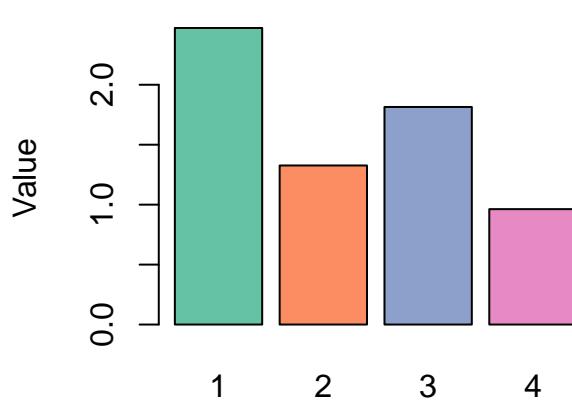
Jupiter.Tisserand.Invariant



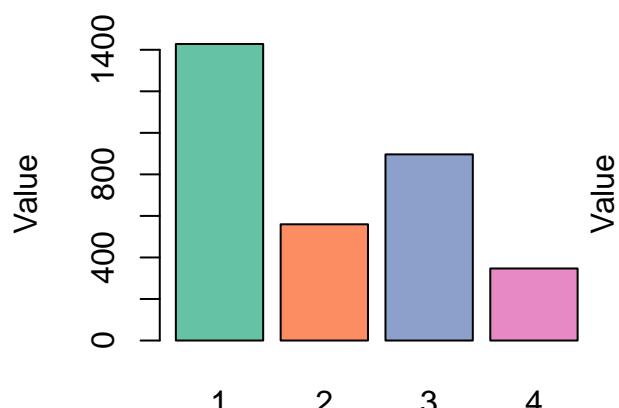
Cluster Eccentricity



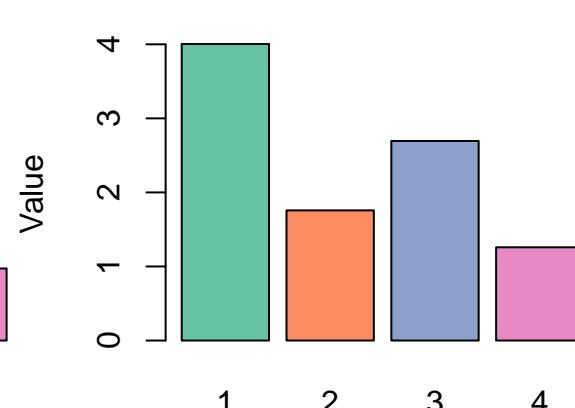
Cluster Semi.Major.Axis



Cluster Orbital.Period



Cluster Aphelion.Dist



Cluster

Cluster

Riferimenti bibliografici

- [1] Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.
- [2] Alpaydin, E. (2020). *Introduction to machine learning*. MIT press.
- [3] Zhou, Z. H. (2012). *Ensemble methods: foundations and algorithms*. CRC press.
- [4] Zhang, C., & Ma, Y. (Eds.). (2012). *Ensemble machine learning: methods and applications*. Springer Science & Business Media.
- [5] Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4, No. 4, p. 738). New York: springer.
- [6] Masoudnia, S., & Ebrahimpour, R. (2014). Mixture of experts: a literature survey. *The Artificial Intelligence Review*, 42(2), 275.
- [7] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural computation*, 3(1), 79-87.
- [8] Yuksel, S. E., Wilson, J. N., & Gader, P. D. (2012). Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems*, 23(8), 1177-1193.
- [9] Jordan, M. I., & Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural computation*, 6(2), 181-214.
- [10] Gormley, I. C., & Frühwirth-Schnatter, S. (2019). Mixture of experts models. In *Handbook of mixture analysis* (pp. 271-307). Chapman and Hall/CRC.