Università Degli Studi Di Milano

Metodi di Ensemble Gerarchici per la Predizione Strutturata della Funzione delle Proteine

Relatore

Prof. Giorgio Valentini

Correlatore

Dr. Marco Notaro

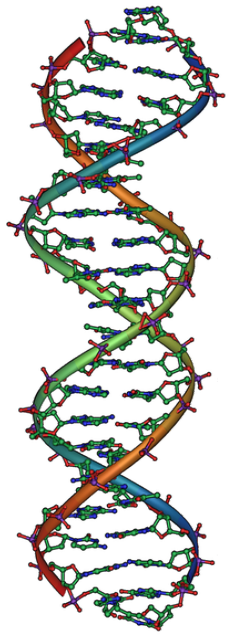
Candidato

Marco Odore

10 Luglio 2018

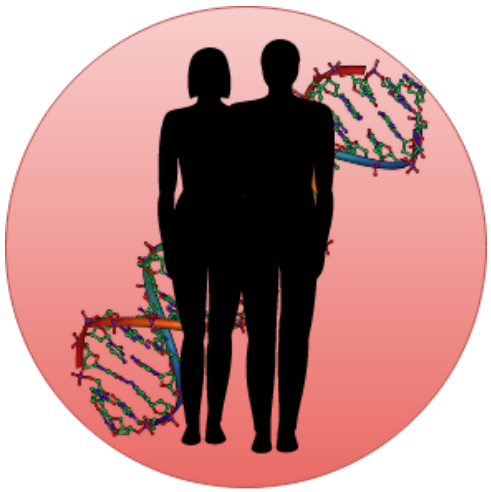
Central Dogma

- All'interno del DNA di ogni essere vivente esistono diverse migliaia di geni.



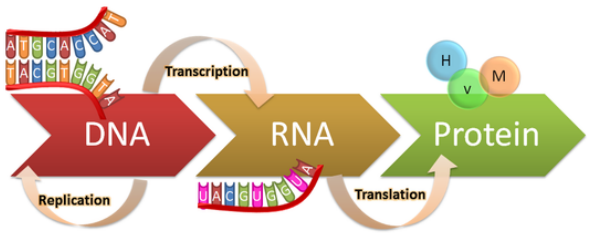
Central Dogma

- All'interno del DNA di ogni essere vivente esistono diverse migliaia di geni.
- Si stima che nel DNA dell'essere umano ci siano tra i 20.000 - 25.000 geni.



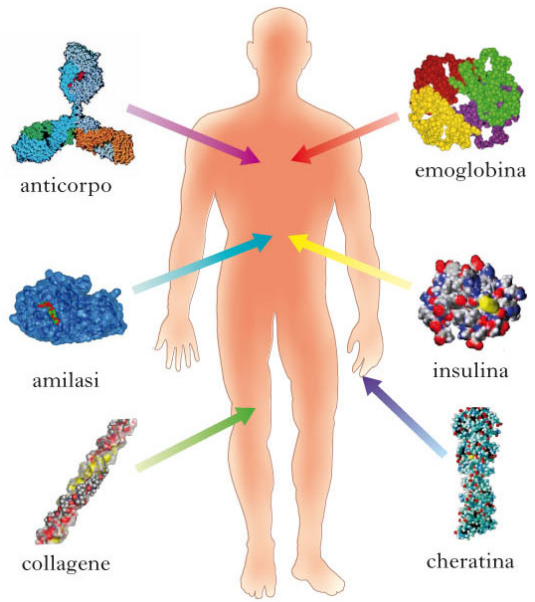
Central Dogma

- All'interno del DNA di ogni essere vivente esistono diverse migliaia di geni.
- Si stima che nel DNA dell'essere umano ci siano tra i 20.000 - 25.000 geni.
- Ogni gene all'interno del DNA è capace di codificare più proteine.



Central Dogma

- All'interno del DNA di ogni essere vivente esistono diverse migliaia di geni.
- Si stima che nel DNA dell'essere umano ci siano tra i 20.000 - 25.000 geni.
- Ogni gene all'interno del DNA è capace di codificare più proteine.
- Ogni proteina è responsabile di una o più funzioni all'interno delle cellule degli esseri viventi.

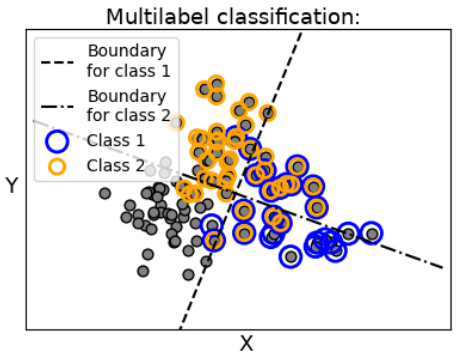


Central Dogma

- All'interno del DNA di ogni essere vivente esistono diverse migliaia di geni.
- Si stima che nel DNA dell'essere umano ci siano tra i 20.000 - 25.000 geni.
- Ogni gene all'interno del DNA è capace di codificare più proteine.
- Ogni proteina è responsabile di una o più funzioni all'interno delle cellule degli esseri viventi.
- Identificare la funzione delle proteine attraverso le analisi di laboratorio è **costosa** e richiede **molto tempo**

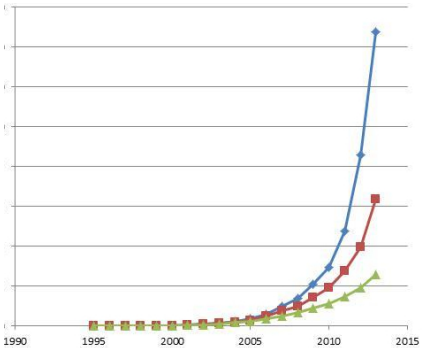
Central Dogma

- All'interno del DNA di ogni essere vivente esistono diverse migliaia di geni.
- Si stima che nel DNA dell'essere umano ci siano tra i 20.000 - 25.000 geni.
- Ogni gene all'interno del DNA è capace di codificare più proteine.
- Ogni proteina è responsabile di una o più funzioni all'interno delle cellule degli esseri viventi.
- Identificare la funzione delle proteine attraverso le analisi di laboratorio è **costosa** e richiede **molto tempo**
- Esistono centinaia di funzioni a cui poter associare un gene/proteina, anche contemporaneamente **(problema multiclasse e multietichetta)**



Central Dogma

- All'interno del DNA di ogni essere vivente esistono diverse migliaia di geni.
- Si stima che nel DNA dell'essere umano ci siano tra i 20.000 - 25.000 geni.
- Ogni gene all'interno del DNA è capace di codificare più proteine.
- Ogni proteina è responsabile di una o più funzioni all'interno delle cellule degli esseri viventi.
- Identificare la funzione delle proteine attraverso le analisi di laboratorio è **costosa** e richiede **molto tempo**
- Esistono centinaia di funzioni a cui poter associare un gene/proteina, anche contemporaneamente **(problema multiclasse e multietichetta)**
- Il quantitativo di dati genomici cresce molto rapidamente.



Il problema della predizione della funzione delle proteine 1/2

- A complicare ulteriormente il problema è il modo in cui sono *relazionate* tra loro le funzioni delle proteine.

Il problema della predizione della funzione delle proteine 1/2

- A complicare ulteriormente il problema è il modo in cui sono *relazionate* tra loro le funzioni delle proteine.
- Esistono infatti due tassonomie principali per l'organizzazione delle classi:

Il problema della predizione della funzione delle proteine 1/2

- A complicare ulteriormente il problema è il modo in cui sono *relazionate* tra loro le funzioni delle proteine.
- Esistono infatti due tassonomie principali per l'organizzazione delle classi:
  - **Gene Ontology** (GO): che organizza le funzioni come un grafo diretto aciclico (DAG), varia per ogni specie, e possiede tre ontologie differenti.

Il problema della predizione della funzione delle proteine 1/2

- A complicare ulteriormente il problema è il modo in cui sono *relazionate* tra loro le funzioni delle proteine.
- Esistono infatti due tassonomie principali per l'organizzazione delle classi:
  - **Gene Ontology** (GO): che organizza le funzioni come un grafo diretto aciclico (DAG), varia per ogni specie, e possiede tre ontologie differenti.
  - **Functional Catalogue** (FunCat): che è organizzato invece come un albero, non varia in base alle specie, e descrive le funzioni in maniera più sintetica rispetto alla Gene Ontology.

Il problema della predizione della funzione delle proteine 1/2

- A complicare ulteriormente il problema è il modo in cui sono *relazionate* tra loro le funzioni delle proteine.
- Esistono infatti due tassonomie principali per l'organizzazione delle classi:
  - **Gene Ontology** (GO): che organizza le funzioni come un grafo diretto aciclico (DAG), varia per ogni specie, e possiede tre ontologie differenti.
  - **Functional Catalogue** (FunCat): che è organizzato invece come un albero, non varia in base alle specie, e descrive le funzioni in maniera più sintetica rispetto alla Gene Ontology.

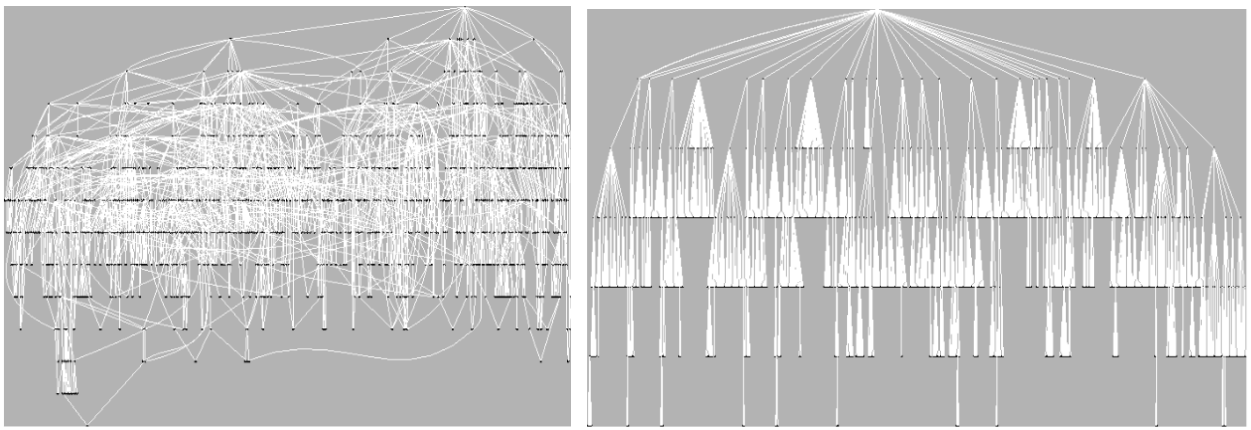


Figura: A sinistra un DAG della GO per la specie *S. cerevisiae*. A destra FunCat.

Il problema della predizione della funzione delle proteine (GO) 2/2

- Data la granularità e specificità superiori della GO e il suo largo utilizzo nella comunità scientifica, all'interno della tesi ci si è soffermati sulla predizione delle sue funzioni.

Il problema della predizione della funzione delle proteine (GO) 2/2

- Data la granularità e specificità superiori della GO e il suo largo utilizzo nella comunità scientifica, all'interno della tesi ci si è soffermati sulla predizione delle sue funzioni.
- Tale tassonomia presenta tre ontologie (e quindi tre DAG) principali:

Il problema della predizione della funzione delle proteine (GO) 2/2

- Data la granularità e specificità superiori della GO e il suo largo utilizzo nella comunità scientifica, all'interno della tesi ci si è soffermati sulla predizione delle sue funzioni.
- Tale tassonomia presenta tre ontologie (e quindi tre DAG) principali:
  - **Processo Biologico** (BP): descrive i processi ad alto livello, come insieme di diverse attività molecolari.

Il problema della predizione della funzione delle proteine (GO) 2/2

- Data la granularità e specificità superiori della GO e il suo largo utilizzo nella comunità scientifica, all'interno della tesi ci si è soffermati sulla predizione delle sue funzioni.
- Tale tassonomia presenta tre ontologie (e quindi tre DAG) principali:
  - **Processo Biologico** (BP): descrive i processi ad alto livello, come insieme di diverse attività molecolari.
  - **Funzione Molecolare** (MF): descrive le funzioni di specifici prodotti genici.

Il problema della predizione della funzione delle proteine (GO) 2/2

- Data la granularità e specificità superiori della GO e il suo largo utilizzo nella comunità scientifica, all'interno della tesi ci si è soffermati sulla predizione delle sue funzioni.
- Tale tassonomia presenta tre ontologie (e quindi tre DAG) principali:
  - **Processo Biologico** (BP): descrive i processi ad alto livello, come insieme di diverse attività molecolari.
  - **Funzione Molecolare** (MF): descrive le funzioni di specifici prodotti genici.
  - **Componente Cellulare** (CC): il luogo all'interno della cellula nelle quali avviene la funzione genica.

Il problema della predizione della funzione delle proteine (GO) 2/2

- Data la granularità e specificità superiori della GO e il suo largo utilizzo nella comunità scientifica, all'interno della tesi ci si è soffermati sulla predizione delle sue funzioni.
- Tale tassonomia presenta tre ontologie (e quindi tre DAG) principali:
  - **Processo Biologico** (BP): descrive i processi ad alto livello, come insieme di diverse attività molecolari.
  - **Funzione Molecolare** (MF): descrive le funzioni di specifici prodotti genici.
  - **Componente Cellulare** (CC): il luogo all'interno della cellula nelle quali avviene la funzione genica.

Visti i costi e la complessità del problema, per indirizzare la ricerca di laboratorio si rende necessaria la predizione della funzione delle proteine tramite metodi **automatici**. Tra questi ricordiamo: i metodi basati sulla **comparazione di biosequenze**, i metodi **basati su reti**, i metodi **Kernel per spazi di output strutturato** e i metodi utilizzati in questa tesi, cioè i metodi **Ensemble Gerarchici**.

Metodi Ensemble Gerarchici 1/2

I Metodi di Ensemble Gerarchici sono metodi caratterizzati da due step principali:

Metodi Ensemble Gerarchici 1/2

I Metodi di Ensemble Gerarchici sono metodi caratterizzati da due step principali:

1. *Predizione flat* delle diverse classi dell'ontologia, generando diversi predittori *indipendenti*.

Metodi Ensemble Gerarchici 1/2

I Metodi di Ensemble Gerarchici sono metodi caratterizzati da due step principali:

1. *Predizione flat* delle diverse classi dell'ontologia, generando diversi predittori *indipendenti*.
2. *Combinazione e correzione gerarchica delle predizioni* sfruttando il DAG dei termini della GO.

Metodi Ensemble Gerarchici 1/2

I Metodi di Ensemble Gerarchici sono metodi caratterizzati da due step principali:

1. *Predizione flat* delle diverse classi dell'ontologia, generando diversi predittori *indipendenti*.
2. *Combinazione e correzione gerarchica delle predizioni* sfruttando il DAG dei termini della GO.

Il secondo step rappresenta la componente *ensemble* del metodo. Tale step si rende necessario in quanto le predizioni flat non tengono in considerazione la struttura gerarchica dei DAG della GO, portando a risultati *inconsistenti*.

Metodi Ensemble Gerarchici 1/2

I Metodi di Ensemble Gerarchici sono metodi caratterizzati da due step principali:

1. *Predizione flat* delle diverse classi dell'ontologia, generando diversi predittori *indipendenti*.
2. *Combinazione e correzione gerarchica delle predizioni* sfruttando il DAG dei termini della GO.

Il secondo step rappresenta la componente *ensemble* del metodo. Tale step si rende necessario in quanto le predizioni flat non tengono in considerazione la struttura gerarchica dei DAG della GO, portando a risultati *inconsistenti*.

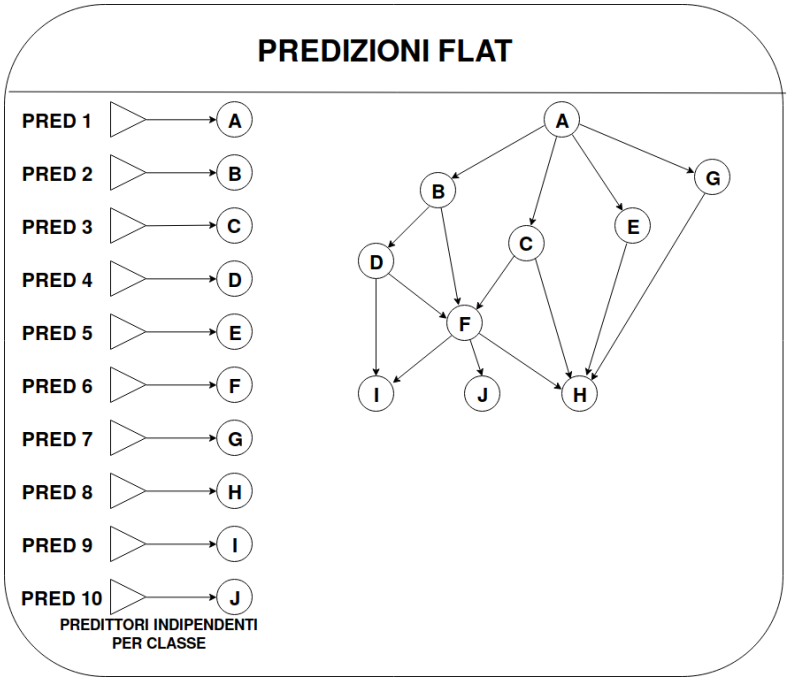
Consistenza & True Path Rule

Un insieme di predizioni $\hat{y} =< \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|N|} >$, dove $|N|$ è la cardinalità dei termini della gerarchia, è definito *consistente*, se rispetta la *True Path Rule*, e cioè:

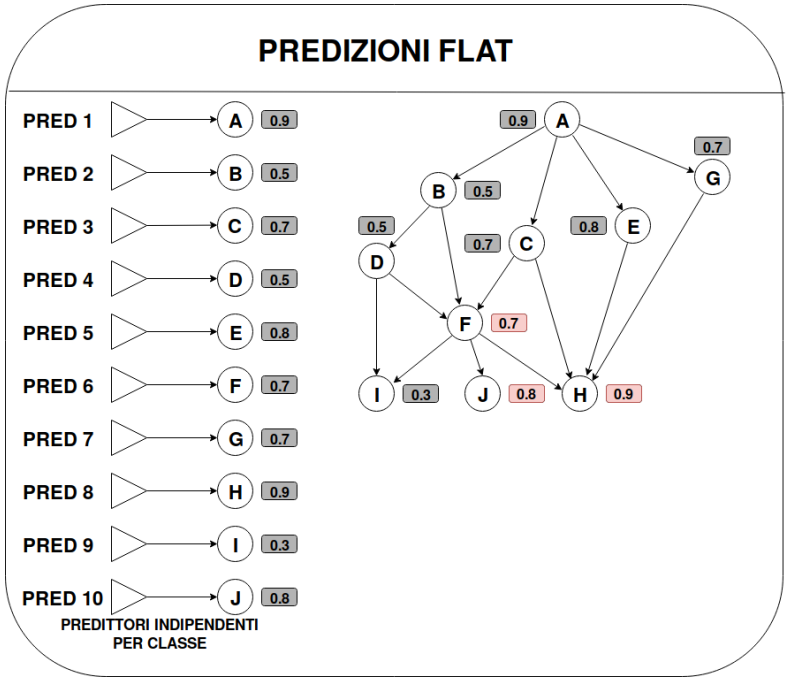
$$y \quad consistente \quad \leftrightarrow \forall i \in N, j \in par(i) \rightarrow y_j \geq y_i$$

Dove $par(i)$ indica l'insieme dei termini genitori del nodo i nella gerarchia.

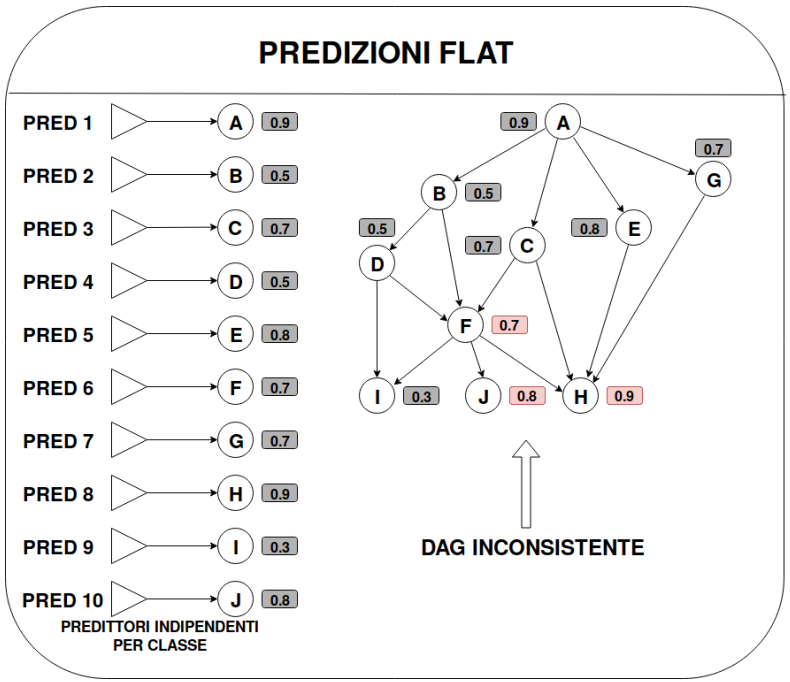
Metodi Ensemble Gerarchici (Esempio) 2/2



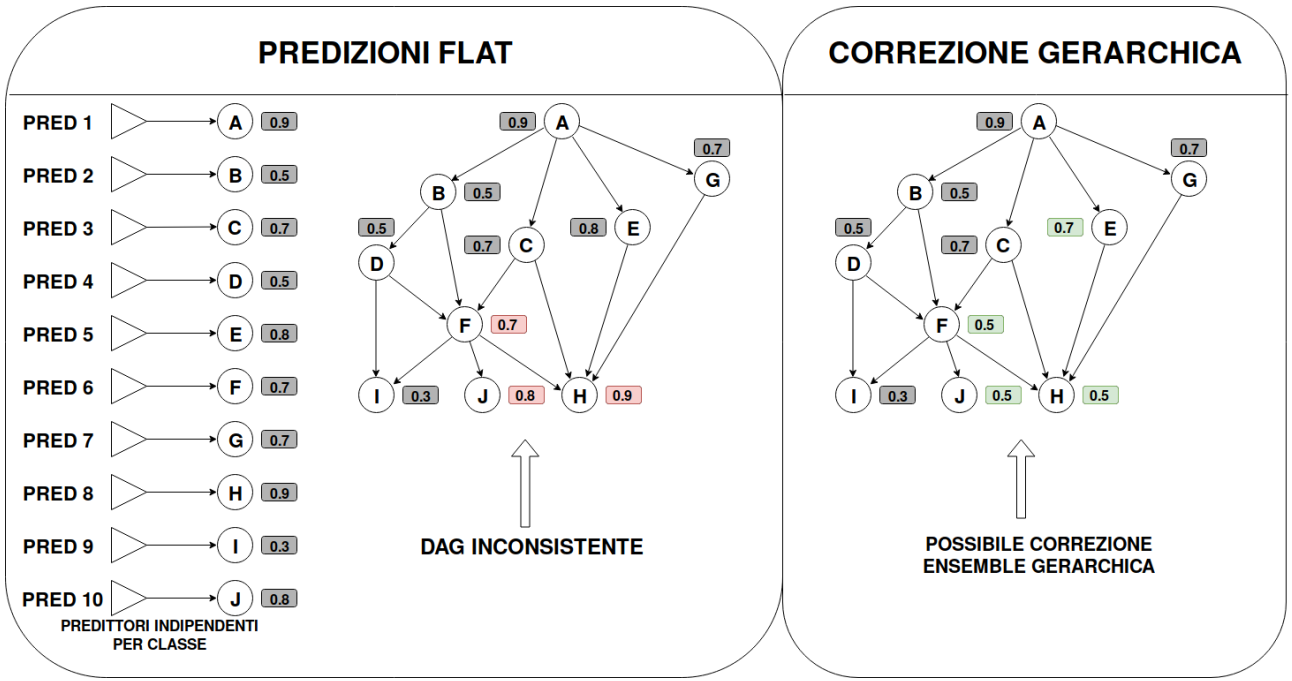
Metodi Ensemble Gerarchici (Esempio) 2/2



Metodi Ensemble Gerarchici (Esempio) 2/2



Metodi Ensemble Gerarchici (Esempio) 2/2



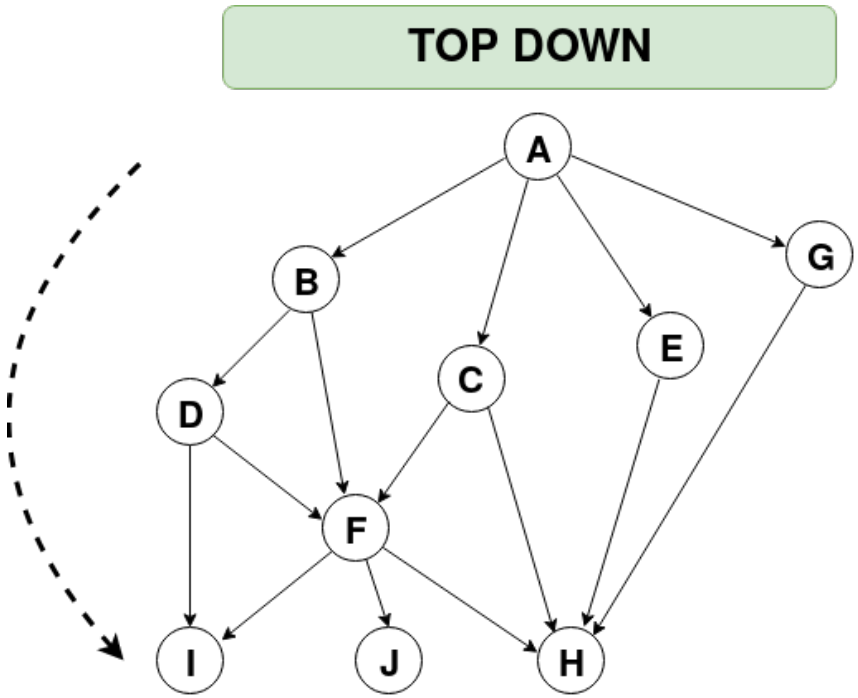
Metodi Ensemble Gerarchici: Approcci

Esistono fondamentalmente due approcci per la correzione:

Metodi Ensemble Gerarchici: Approcci

Esistono fondamentalmente due approcci per la correzione:

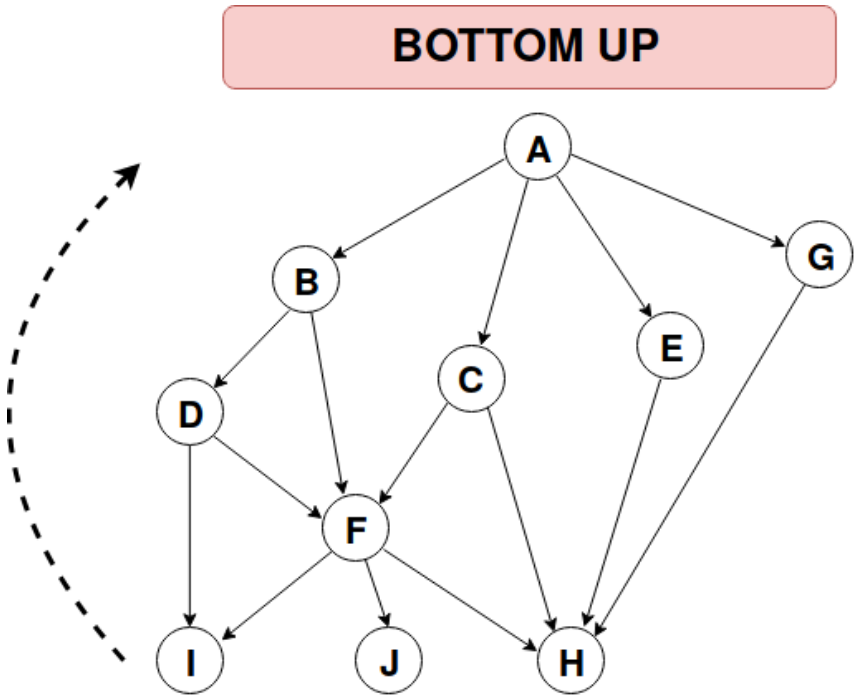
- *Top-down*: le predizioni vengono corrette dai nodi più generali a quelli più specifici.



Metodi Ensemble Gerarchici: Approcci

Esistono fondamentalmente due approcci per la correzione:

- *Top-down*: le predizioni vengono corrette dai nodi più generali a quelli più specifici.
- *Bottom-up*: Le predizioni vengono corrette dai nodi più specifici verso quelli più generali.



Metodo Top-Down Gerarchico (HTD-DAG)

- È un metodo che utilizza l'approccio *Top-Down*.

[1] **Dove il livello è quello del cammino massimo dalla radice**

Metodo Top-Down Gerarchico (HTD-DAG)

- È un metodo che utilizza l'approccio *Top-Down*.
- La correzione avviene ricorsivamente, percorrendo il grafo per *livelli*[1]. Più precisamente, dato il grafo $G = (N, E)$, gli score flat $f(x) = \hat{y}$ sono corretti gerarchicamente a $\bar{y}$, applicando la seguente regola:

Aggiornamento con HTD-DAG

$$\bar{y}_i := \begin{cases} \hat{y}_i & \text{if } \ i \in root(G) \\ min_{j \in par(i)} \bar{y}_j & \text{if } \ min_{j \in par(i)} \bar{y}_j < \hat{y}_i \\ \hat{y}_i & \text{altrimenti} \end{cases}$$

Dove $par(i)$ specifica i genitori del nodo i.

[1] **Dove il livello è quello del cammino massimo dalla radice**

Metodo True Path Rule per DAG (TPR-DAG)

- È un metodo che combina gli approcci top-down e bottom-up per la correzione delle predizioni flat.

Metodo True Path Rule per DAG (TPR-DAG)

- È un metodo che combina gli approcci top-down e bottom-up per la correzione delle predizioni flat.
- È suddiviso in due step sequenziali:

Metodo True Path Rule per DAG (TPR-DAG)

- È un metodo che combina gli approcci top-down e bottom-up per la correzione delle predizioni flat.
- È suddiviso in due step sequenziali:
  1. **Step bottom-up**: che partendo dai nodi più specifici del DAG, propaga quelle predizioni flat che sono considerate *positive*.

Metodo True Path Rule per DAG (TPR-DAG)

- È un metodo che combina gli approcci top-down e bottom-up per la correzione delle predizioni flat.
- È suddiviso in due step sequenziali:
  1. **Step bottom-up**: che partendo dai nodi più specifici del DAG, propaga quelle predizioni flat che sono considerate *positive*.
  2. **Step top-down**: È il medesimo step utilizzato dal metodo HTD-DAG.

Metodo True Path Rule per DAG (TPR-DAG)

- È un metodo che combina gli approcci top-down e bottom-up per la correzione delle predizioni flat.
- È suddiviso in due step sequenziali:
  1. **Step bottom-up**: che partendo dai nodi più specifici del DAG, propaga quelle predizioni flat che sono considerate *positive*.
  2. **Step top-down**: È il medesimo step utilizzato dal metodo HTD-DAG.
- Lo step top down si rende necessario in quanto la propagazione delle predizioni positive dal basso verso l'alto non garantisce la consistenza delle predizioni necessarie alla True Path Rule.

Metodo True Path Rule per DAG (TPR-DAG)

- È un metodo che combina gli approcci top-down e bottom-up per la correzione delle predizioni flat.
- È suddiviso in due step sequenziali:
  1. **Step bottom-up**: che partendo dai nodi più specifici del DAG, propaga quelle predizioni flat che sono considerate *positive*.
  2. **Step top-down**: È il medesimo step utilizzato dal metodo HTD-DAG.
- Lo step top down si rende necessario in quanto la propagazione delle predizioni positive dal basso verso l'alto non garantisce la consistenza delle predizioni necessarie alla True Path Rule.
- La selezione dei nodi considerati *positivi* può avvenire in diverse maniere: con *soglia adattiva*, *senza soglia* e con *soglia fissa*.

Generalized Pool Adjacent Violator (GPAV) (1/4)

- Per il passo Top-Down dell'algoritmo TPR-DAG (e HTD) è possibile sfruttare gli algoritmi per la risoluzione dei problemi di **Isotonic Regression**:

Isotonic Regression (caso generale con ordinamento parziale)

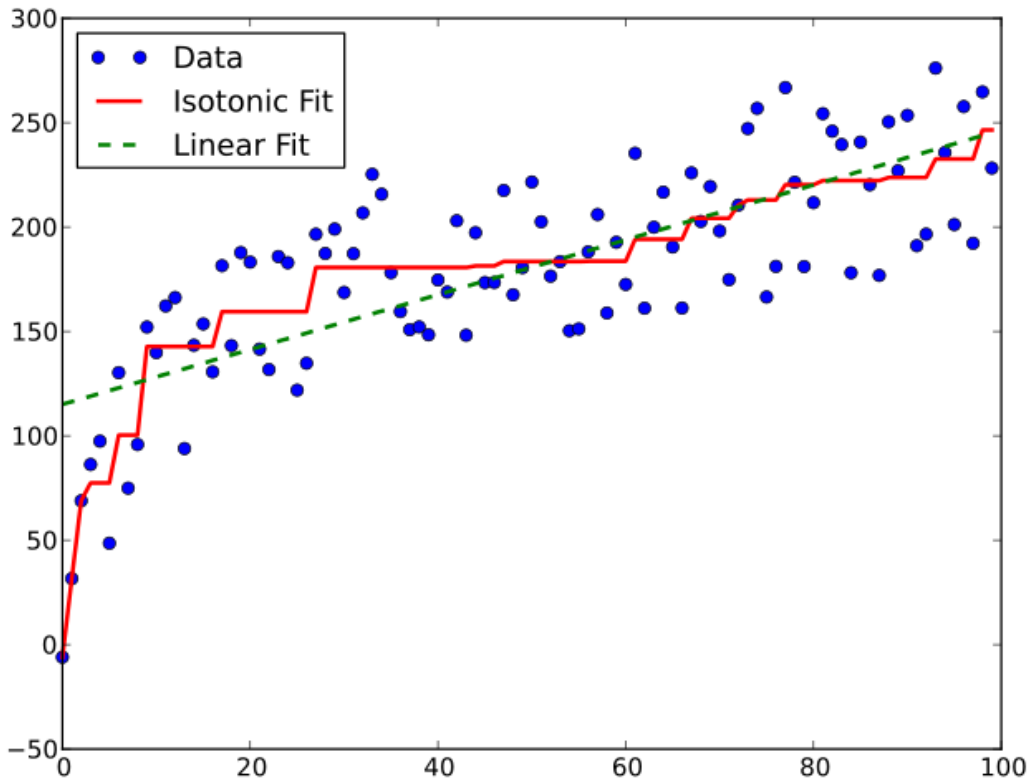
Dato un DAG, $G(N, E)$, con il set di nodi $N = \{1, 2, ..., n\}$, si deve trovare il vettore $x^* \in R^n$ tale che:

$$min \sum_{i=1}^{n} w_i (x_i - a_i)^2$$

$$\text{such that } x_i \leq x_j \ \forall (i, j) \in E$$

Generalized Pool Adjacent Violator (GPAV) (2/4)

Esempio di Isotonic Regression con *ordinamento totale*:



Generalized Pool Adjacent Violator (GPAV) (3/4)

Funzionamento Generale:

Generalized Pool Adjacent Violator (GPAV) (3/4)

Funzionamento Generale:

- Per far funzionare l'algoritmo è necessario effettuare un *ordinamento topologico* del grafo.

Generalized Pool Adjacent Violator (GPAV) (3/4)

Funzionamento Generale:

- Per far funzionare l'algoritmo è necessario effettuare un *ordinamento topologico* del grafo.
- L'algoritmo genera uno split del set di nodi N del DAG, in un insieme di *blocchi disgiunti definiti da H* (inizialmente $H = N$).

Generalized Pool Adjacent Violator (GPAV) (3/4)

Funzionamento Generale:

- Per far funzionare l'algoritmo è necessario effettuare un *ordinamento topologico* del grafo.
- L'algoritmo genera uno split del set di nodi N del DAG, in un insieme di *blocchi disgiunti definiti da* H (inizialmente $H = N$).
- Seguendo l'ordinamento topologico del grafo, un blocco *assorbe* un suo blocco *predecessore* se si verificano determinate condizioni.

Generalized Pool Adjacent Violator (GPAV) (3/4)

Funzionamento Generale:

- Per far funzionare l'algoritmo è necessario effettuare un *ordinamento topologico* del grafo.
- L'algoritmo genera uno split del set di nodi N del DAG, in un insieme di *blocchi disgiunti definiti da* H (inizialmente $H = N$).
- Seguendo l'ordinamento topologico del grafo, un blocco *assorbe* un suo blocco *predecessore* se si verificano determinate condizioni.
- I nodi presenti nel medesimo blocco B_i *condividono il medesimo valore* x_i, e quindi a seguito dell'assorbimento sarà necessario un aggiornamento di tale valore.

Generalized Pool Adjacent Violator (GPAV e ISO-TPR) (4/4)

Algorithm 1 GPAV

1: **procedure** GPAV
2: $H = N$
3: **for** (each $i \in N$) **do**
4: $B_i = \{i\}$
5: $B_i^- = i^-$
6: $x_i = \hat{y}_i$
7: $W_i = w_i$
8: **for** $k = 1, 2, ..., n$ **do**
9: *//finché esiste un predecessore di B_k che viola la monotonicità*
10: **while** $\{i \in B_k^- : x_i > x_k\} \neq 0$ **do**
11: *// Trova l'elemento che viola maggiormente il vincolo*
12: **Find** $j \in B_k^- : x_j = max\{x_i : i \in B_k^-\}$
13: **Absorb(k, j)** *// j viene assorbito da B_k*

Generalized Pool Adjacent Violator (GPAV e ISO-TPR) (4/4)

Algorithm 2 GPAV

procedure GPAV
  $H = N$
  for (each $i \in N$) **do**
    $B_i = \{i\}$
    $B_i^- = i^-$
    $x_i = \hat{y}_i$
    $W_i = w_i$
  for $k = 1, 2, ..., n$ **do**
    //finché esiste un predecessore di B_k che viola la monotonicità
    while $\{i \in B_k^- : x_i > x_k\} \neq 0$ **do**
      // Trova l'elemento che viola maggiormente il vincolo
      Find $j \in B_k^- : x_j = max\{x_i : i \in B_k^-\}$
      Absorb(k, j) *// j viene assorbito da B_k*

- Riassumendo, l'algoritmo effettua degli assorbimenti di blocchi adiacenti, finché questi violano i vincoli del problema quadratico, generando di fatto una partizione dei nodi, in cui le parti condividono lo stesso valore.

Generalized Pool Adjacent Violator (GPAV e ISO-TPR) (4/4)

Algorithm 3 GPAV

procedure GPAV
  $H = N$
  for (each $i \in N$) **do**
    $B_i = \{i\}$
    $B_i^- = i^-$
    $x_i = \hat{y}_i$
    $W_i = w_i$
  for $k = 1, 2, ..., n$ **do**
    //finché esiste un predecessore di B_k che viola la monotonicità
    while $\{i \in B_k^- : x_i > x_k\} \neq 0$ **do**
      // Trova l'elemento che viola maggiormente il vincolo
      Find $j \in B_k^- : x_j = max\{x_i : i \in B_k^-\}$
      Absorb(k, j) *// j viene assorbito da B_k*

- Riassumendo, l'algoritmo effettua degli assorbimenti di blocchi adiacenti, finché questi violano i vincoli del problema quadratico, generando di fatto una partizione dei nodi, in cui le parti condividono lo stesso valore.
- L'algoritmo GPAV ha una complessità pari a $O(n^2)$.

Generalized Pool Adjacent Violator (GPAV e ISO-TPR) (4/4)

Algorithm 4 GPAV

procedure GPAV
  $H = N$
  for (each $i \in N$) **do**
    $B_i = \{i\}$
    $B_i^- = i^-$
    $x_i = \hat{y}_i$
    $W_i = w_i$
  for $k = 1, 2, ..., n$ **do**
    //finché esiste un predecessore di B_k che viola la monotonicità
    while $\{i \in B_k^- : x_i > x_k\} \neq 0$ **do**
      // Trova l'elemento che viola maggiormente il vincolo
      Find $j \in B_k^- : x_j = max\{x_i : i \in B_k^-\}$
      Absorb(k, j) *// j viene assorbito da B_k*

- Riassumendo, l'algoritmo effettua degli assorbimenti di blocchi adiacenti, finché questi violano i vincoli del problema quadratico, generando di fatto una partizione dei nodi, in cui le parti condividono lo stesso valore.
- L'algoritmo GPAV ha una complessità pari a $O(n^2)$.
- Sostituendo GPAV allo step Top-Down dell'algoritmo TPR-DAG visto in precedenza (invece che HTD), si ottiene l'algoritmo **ISO-TPR**.

Predizione della funzione delle proteine della specie C.elegans (WORM) 1/2

- Si è eseguita la sperimentazione sul genoma della specie *Caenorhabditis elegans* (WORM), un organismo modello molto semplice, utilizzato frequentemente in biologia, soprattutto per studi di biologia dello sviluppo.

Predizione della funzione delle proteine della specie C.elegans (WORM) 1/2

- Si è eseguita la sperimentazione sul genoma della specie *Caenorhabditis elegans* (WORM), un organismo modello molto semplice, utilizzato frequentemente in biologia, soprattutto per studi di biologia dello sviluppo.
- L'insieme delle istanze utilizzato come input dai diversi predittori per il nostro problema, è dato dall'insieme dei geni della specie oggetto della sperimentazione, il quale è rappresentato da una matrice simmetrica generata dal network di interazione proteina-proteina estratto dal database *STRING* (Search Tool for the Retrieval of Interacting Genes/Proteins).

Predizione della funzione delle proteine della specie C.elegans (WORM) 1/2

- Si è eseguita la sperimentazione sul genoma della specie *Caenorhabditis elegans* (WORM), un organismo modello molto semplice, utilizzato frequentemente in biologia, soprattutto per studi di biologia dello sviluppo.
- L'insieme delle istanze utilizzato come input dai diversi predittori per il nostro problema, è dato dall'insieme dei geni della specie oggetto della sperimentazione, il quale è rappresentato da una matrice simmetrica generata dal network di interazione proteina-proteina estratto dal database *STRING* (Search Tool for the Retrieval of Interacting Genes/Proteins).
- Nel caso specifico della specie C.elegans, la matrice STRING ha dimensione 15752×15752. Il nostro problema ha quindi 15752 istanze.

Predizione della funzione delle proteine della specie C.elegans (WORM) 1/2

- Si è eseguita la sperimentazione sul genoma della specie *Caenorhabditis elegans* (WORM), un organismo modello molto semplice, utilizzato frequentemente in biologia, soprattutto per studi di biologia dello sviluppo.
- L'insieme delle istanze utilizzato come input dai diversi predittori per il nostro problema, è dato dall'insieme dei geni della specie oggetto della sperimentazione, il quale è rappresentato da una matrice simmetrica generata dal network di interazione proteina-proteina estratto dal database *STRING* (Search Tool for the Retrieval of Interacting Genes/Proteins).
- Nel caso specifico della specie C.elegans, la matrice STRING ha dimensione 15752×15752. Il nostro problema ha quindi 15752 istanze.
- Per le classi funzionali/termini e annotazioni, queste cambiano in base al DAG di riferimento:

ontologia	numero di termini	numero di archi
BP	4068	8066
MF	1163	1567
CC	578	1082

Predizione della funzione delle proteine della specie C.elegans (WORM) 2/2

Per evitare di avere problemi nella cross-validazione, i DAG sono stati ridotti a quei termini per cui si hanno almeno 10 annotazioni. Un po' di statistiche a seguito della selezione:

Onto	**numero di termini**	**media**	**d.std.**	**massimo**	**minimo**
BP	1335	71,33	151,68	2597	10
MF	186	61,23	191,84	1806	10
CC	221	131,9	302,25	1924	10

Tabella: La colonna *numero di termini* indica il numero di termini ottenuti dopo la selezione, *media* la media delle annotazioni per classe, per l'ontologia di riferimento, *d.std.* la deviazione standard delle annotazioni per l'ontologia di riferimento, *massimo* e *minimo* rispettivamente il massimo e minimo numero di annotazioni.

Algoritmi di ML per le predizioni flat

- Per quanto riguarda gli algoritmi di apprendmento automatico da utilizzare per le predizioni flat, si sono considerati i seguenti metodi:
  1. *K-Nearest Neighbors*
  2. *Logit Boost*
  3. *Linear Discriminant Analysis*
  4. *eXtreme Gradient Boosting*
  5. *C5.0* (Alberi di decisione)
  6. *Random Forest*
  7. *Multilayer Perceptron*
  8. *Support Vector Machine lineare*
  9. *Bagged CART* (Bagged ensemble di alberi di decisione)
  10. *AdaBoost.M1*
  11. *Naive Bayes*
  12. *Glmnet*

Algoritmi di ML per le predizioni flat

- Per quanto riguarda gli algoritmi di apprendmento automatico da utilizzare per le predizioni flat, si sono considerati i seguenti metodi:
  1. *K-Nearest Neighbors*
  2. *Logit Boost*
  3. *Linear Discriminant Analysis*
  4. *eXtreme Gradient Boosting*
  5. *C5.0* (Alberi di decisione)
  6. *Random Forest*
  7. *Multilayer Perceptron*
  8. *Support Vector Machine lineare*
  9. *Bagged CART* (Bagged ensemble di alberi di decisione)
  10. *AdaBoost.M1*
  11. *Naive Bayes*
  12. *Glmnet*
- Dato l'elevato numero di algoritmi selezionati per la sperimentazione, si è deciso di non effettuare il tuning dei parametri, questo per evitare di allungare ulteriormente i tempi dell'intero processo di valutazione e generazione degli score flat.

Stima preliminare dei tempi di calcolo degli algoritmi flat

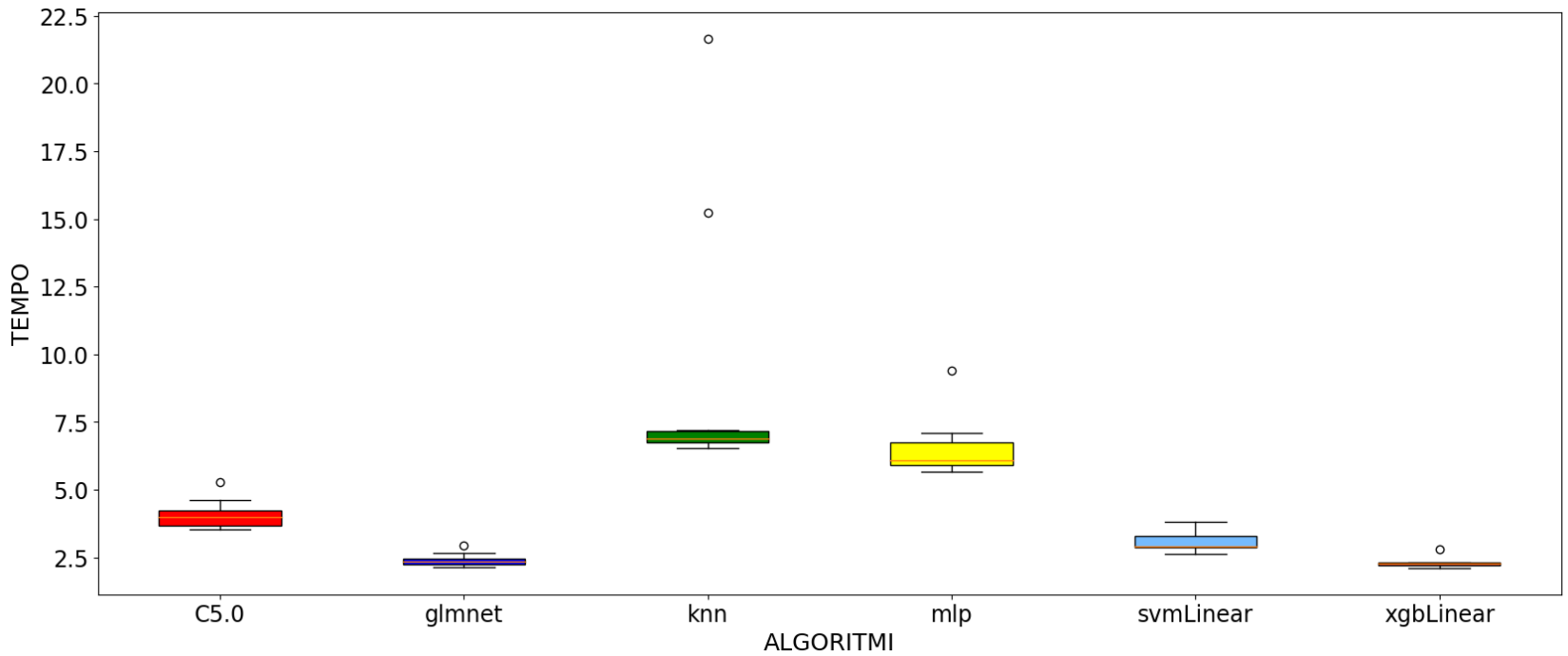


Figura: Il box plot dei tempi di esecuzione, con cross-validation a 10 fold per l' ontologia BP. I tempi sono da intendersi in ore e per classe, per un campione di 10 classi.

Riduzione della complessità del problema

- Oltre a ridurre il numero di fold della cross-validation a 5, per ridurre la complessità del problema si sono utilizzati due metodi:

Riduzione della complessità del problema

- Oltre a ridurre il numero di fold della cross-validation a 5, per ridurre la complessità del problema si sono utilizzati due metodi:
  1. *Selezione delle Feature con correlazione di Pearson* (FS).

Riduzione della complessità del problema

- Oltre a ridurre il numero di fold della cross-validation a 5, per ridurre la complessità del problema si sono utilizzati due metodi:
  1. *Selezione delle Feature con correlazione di Pearson* (FS).
  2. *Analisi delle componenti principali* (PCA) con selezione delle prime componenti che spiegano una determinata percentuale di varianza.

Riduzione della complessità del problema

- Oltre a ridurre il numero di fold della cross-validation a 5, per ridurre la complessità del problema si sono utilizzati due metodi:
  1. *Selezione delle Feature con correlazione di Pearson* (FS).
  2. *Analisi delle componenti principali* (PCA) con selezione delle prime componenti che spiegano una determinata percentuale di varianza.
- Per la FS con correlazione di Pearson, si sono effettuate stime dei tempi di calcolo e performance con le:
  - prime 1000 feature
  - prime 500 feature
  - e prime 100 feature

  nel ranking.

Riduzione della complessità del problema

- Oltre a ridurre il numero di fold della cross-validation a 5, per ridurre la complessità del problema si sono utilizzati due metodi:
  1. *Selezione delle Feature con correlazione di Pearson* (FS).
  2. *Analisi delle componenti principali* (PCA) con selezione delle prime componenti che spiegano una determinata percentuale di varianza.
- Per la FS con correlazione di Pearson, si sono effettuate stime dei tempi di calcolo e performance con le:
  - prime 1000 feature
  - prime 500 feature
  - e prime 100 feature

  nel ranking.
- Mentre per la PCA si sono selezionate le componenti che spiegavano rispettivamente:
  - il 90%
  - il 70%
  - e il 50%

  della varianza.

Organizzazione esperimenti

A seguito della precedente analisi, si è deciso di effettuare la fase di predizione flat iniziale con due configurazioni, che hanno rappresentato un buon compromesso tra tempi di esecuzione e performance, e cioè:

Organizzazione esperimenti

A seguito della precedente analisi, si è deciso di effettuare la fase di predizione flat iniziale con due configurazioni, che hanno rappresentato un buon compromesso tra tempi di esecuzione e performance, e cioè:

- cross-validation 5 fold + Selezione delle prime 100 feature con Correlazione di Pearson.

Organizzazione esperimenti

A seguito della precedente analisi, si è deciso di effettuare la fase di predizione flat iniziale con due configurazioni, che hanno rappresentato un buon compromesso tra tempi di esecuzione e performance, e cioè:

- cross-validation 5 fold + Selezione delle prime 100 feature con Correlazione di Pearson.
- cross-validation 5 fold + Selezione delle prime 15 componenti della PCA.

Organizzazione esperimenti

A seguito della precedente analisi, si è deciso di effettuare la fase di predizione flat iniziale con due configurazioni, che hanno rappresentato un buon compromesso tra tempi di esecuzione e performance, e cioè:

- cross-validation 5 fold + Selezione delle prime 100 feature con Correlazione di Pearson.
- cross-validation 5 fold + Selezione delle prime 15 componenti della PCA.

A seguito della generazione dei risultati flat, si è effettuata poi la correzione gerarchica tramite i seguenti metodi di ensemble gerarchico:

1. HTD-DAG
2. GPAV
3. TPR-DAG (senza soglia, con soglia adattiva, senza soglia e correzione con pesi)
4. ISO-TPR (senza soglia, con soglia adattiva)

Predizioni Flat

Predizioni Strutturate

Metodi Flat vs Metodi Ensemble Gerarchici

Confronto tra Metodi Ensemble Gerarchici

Conclusioni