Architetture e Programmazione dei Sistemi di Elaborazione
Progetto a.a. 2021/22

"Correlation Feature Selection"
in linguaggio assembly x86-32+SSE, x86-64+AVX e openMP

Fabrizio Angiulli Fabio Fassetti Simona Nisticò

1 Decrizione del problema.

La *feature selection* è il processo di selezione del sottoinsieme di caratteristiche più interessanti dei dati da utilizzare nell'analisi, questa semplificazione porta notevoli vantaggi per i modelli estratti, nella riduzione dei tempi di calcolo, nel migliorare l'interpretabilità dei risultati e nell'evitare i problemi legati all'alta dimensionalità. In alcuni contesti, inoltre, la feature selection è di per sé l'obiettivo dell'analisi.
Si consideri un dataset DS a valori reali definito su un insieme di features $\mathcal{F} = \{f_1, \ldots, f_d\}$ e composto quindi da n oggetti (righe) e d attributi (colonne). Ad un dataset è associata una variabile dicotomica c, detta variabile di *classe*, in accordo alla quale l'insieme di dati viene suddiviso in due gruppi. Quindi, ad un dataset DS è abbinato un vettore binario c composto da n elementi, dette *etichette*, tale che $c[i] = 0$ se l'i-esimo oggetto del dataset appartiene al gruppo 0 e $c[i] = 1$ se l'i-esimo oggetto del dataset appartiene al gruppo 1.
In molti scenari reali, lo scopo è scoprire quali caratteristiche dei dati hanno un'influenza maggiore sull'appartenenza di un individuo a un gruppo. In particolare, dato un valore k, l'obiettivo è quello individuare l'insieme $\mathcal{S} \subset \mathcal{F}$ delle $k < d$ features più influenti in accordo ad un criterio di qualità predefinito
L'assunzione alla base di una tecnica di feature selection è che i dati contengono alcune caratteristiche ridondanti o irrilevanti che possono essere rimossi senza perdere troppa informazione. *Ridondanza* e *irrilevanza* sono due nozioni distinte, la prima indica una caratteristica che porta con sé un contenuto informativo simile a quello associato ad un'altra caratteristica, la seconda indica invece una caratteristica non rilevante per l'analisi che si vuole svolgere. L'idea della feature selection è di selezionare un insieme di caratteristiche tutte rilevanti per l'analisi e non ridondanti tra di loro.
Esistono diversi meccanismi di selezione delle feature. Tra questi, notevole successo hanno quelli basati su algoritmi golosi che, tipicamente, seguono il seguente schema.
Una delle funzioni più usate per misurare ridondanza e rilevanza è la *correlazione* e la relativa tecnica di selezione è la *Correlation Feature Selection* (CFS). Questa valuta l'insieme di feature più interessanti sulla base della seguente ipotesi: "gli insiemi di feature più interessanti contengono caratteristiche altamente correlate con la variabile target, ma non correlate tra loro". Segue da questa definizione che il *merito* di un insieme di feature $\mathcal{S}$ costituito da k elementi

ALGORITMO 1: Greedy Feature Selection

Input: un dataset DS definito sull'insieme di feature $\mathcal{F}$, un vettore c contenente le etichette, il numero k di feature da estrarre

Output: l'insieme $\mathcal{S} \subseteq \mathcal{F}$ delle k feature selezionate

```
1 begin
2     S = ∅;
3     while |S| < k do
4         calcolare, per ogni feature f_i, il punteggio sc_i dell'insieme S ∪ {f_i};
5         aggiungere all'insieme S la feature che ha ottenuto il punteggio massimo;
```

può essere calcolato con la seguente equazione:

$$merit_{\mathcal{S}_k} = \frac{k \cdot |\overline{r_{cf}}|}{\sqrt{k + k \cdot (k-1)|\overline{r_{ff}}|}},$$

dove $\overline{r_{cf}}$ è il valore medio di tutte le correlazioni feature-classification e $\overline{r_{ff}}$ è il valore medio di tutte le correlazioni feature-feature.
In particolare, essendo la classe una variabile categorica dicotomica e la feature una variabile numerica, per calcolare la correlazione tra una feature f e la classe c si ricorre al *point biserial correlation coefficient*. Dividendo il dataset in due gruppi 0 e 1 sulla base del valore assunto dalla variabile di classe, il *point biserial correlation coefficient* è definito come:

$$r_{cf} = \frac{\mu_0 - \mu_1}{\sigma_f} \cdot \sqrt{\frac{n_0 \cdot n_1}{n^2}},$$

dove μ_0 è la media dei valori assunti dal gruppo 0 sulla feature f, μ_1 è la media dei valori assunti dal gruppo 1 sulla feature f, σ_f è la deviazione standard campionaria relativa alla feature f, n_0 è la numerosità del gruppo 0, n_1 è la numerosità del gruppo 1 e $n = n_0 + n_1$. La deviazione standard campionaria di una feature f si calcola come:

$$\sigma_f = \sqrt{\frac{1}{n-1} \cdot \sum_{i=1}^{n} \Big(x_i - \mu\Big)^2},$$

dove n è la numerosità dei dati, x_i sono i valori assunti sulla feature f e μ è il loro valore medio.
Riguardo la correlazione feature-feature $\overline{r_{ff}}$, per questa si ricorre al *Pearson's correlation coefficient*, definito come

$$r_{f_x f_y} = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^{n}(x_i - \mu_x)^2} \cdot \sqrt{\sum_{i=1}^{n}(y_i - \mu_y)^2}},$$

dove x_i sono i valori assunti sulla feature f_x, μ_x è il loro valore medio, y_i sono i valori assunti sulla feature f_y, μ_y è il loro valore medio e n è la numerosità dei dati.
L'algoritmo 2 riepiloga i passi previsti dalla tecnica descritta per la ricerca dell'insieme di features $\mathcal{S}$.

ALGORITMO 2: Correlation Feature Selection

Input: un dataset DS definito sull'insieme di feature $\mathcal{F}$, un vettore c contenente le etichette, il numero k di feature da estrarre

Output: l'insieme $\mathcal{S} \subseteq \mathcal{F}$ delle k feature selezionate

```
1 begin
2     S = ∅;
3     while |S| < k do
4         calcolare, per ogni feature f_i, il punteggio merit_{S∪{f_i}} dell'insieme S ∪ {f_i};
5         sia f_i^* la feature che ha ottenuto il punteggio massimo;
6         S = S ∪ f_i^* ;
7         aggiungere all'insieme S la feature f_i che ha ottenuto il punteggio massimo;
```

2 Descrizione dell'attività progettuale.

Obiettivo del progetto è mettere a punto un'implementazione dell'algoritmo Correlation Features Selection in linguaggio C e di migliorarne le prestazioni utilizzando le tecniche di ottimizzazione basate sull'organizzazione dell'hardware.
L'ambiente sw/hw di riferimento è costituito dal linguaggio di programmazione C (`gcc`), dal linguaggio assembly x86-32+SSE e dalla sua estensione x86-32+AVX (`nasm`) e dal sistema operativo Linux (`ubuntu`).
In particolare il codice deve consentire di trovare l'insieme di features $\mathcal{S}$ con l'algoritmo 2, dato il valore del parametro `-k <`k`>`.
Quindi la chiamata avrà la seguente struttura:

```
./cfs<arch> -ds <DS> -labels <LABELS> -k <k> [-s] [-d]
    <arch>: architettura associata all'eseguibile, {32c, 64c, 32ompc 64ompc}
    <DS>: file ds2 contenente una matrice di dimensione nxd riportante i dati
    <LABELS>: file ds2 contenente un vettore di dimensione n riportante le etichette dei dati
    <k>: numero di features da estrarre
    -s: opzionale, modo silenzioso, nessuna stampa, default 0 - false
    -d: opzionale, stampa a video i risultati, default 0 - false
```

e sorgenti ed eseguibili devono essere contenuti in una cartella il cui nome è l'id del gruppo. Qualora un valore di un parametro (sia esso di default o specificato dall'utente) non sia applicabile, il codice deve segnalarlo con un messaggio e terminare.

Di seguito si riportano ulteriori linee guida per lo svolgimento del progetto:

- Si consiglia di affrontare il progetto nel seguente modo:

  1. Codificare l'algoritmo interamente in linguaggio C, possibilmente come sequenza di chiamate a funzioni;
  2. Sostituire le funzioni scritte in linguaggio ad alto livello che necessitano di essere ottimizzate con corrispondenti funzioni scritte in linguaggo assembly.

  Ciò consentirà di verificare che l'algoritmo che si intende ottimizzare è corretto e di gestire più facilmente la complessità del progetto.

- Al fine di migliorare la valutazione dell'attività progettuale è preferibile presentare nella relazione un confronto tra le prestazioni delle versioni intermedie, ognuna delle quali introduce una particolare ottimizzazione, e finale del codice. Obiettivo del confronto è sostanziare la bontà delle ottimizzazioni effettuate.

- Occorre lavorare in autonomia e non collaborare con gli altri gruppi. Soluzioni troppo simili riceveranno una valutazione negativa. I progetti verranno messi in competizione.

- Sono richieste due soluzioni software, la prima per l'architettura x86-32+SSE e la seconda per l'architettura x86-64+AVX.

  Per i dettagli riguardanti la redazione del codice fare riferimento al file sorgente *c* `cfs32c.c`, al file sorgente *nasm* `cfs32.nasm`, allo script eseguibile `runcfs32` (versione `x86-32+SSE`) e al file sorgente *c* `cfs64c.c`, al sorgente *nasm* `cfs64.nasm`, allo script eseguibile `runcfs64` per la versione `x86-64+AVX` disponibili sulla piattaforma.

  Per l'interfacciamento tra programmi in linguaggio C e programmi in linguaggio assembly fare riferimento al documento allegato alla descrizione del progetto.

- È inoltre richiesta per ogni soluzione, una versione che faccia uso delle istruzioni `OpenMP`. I nomi dei relativi file dovranno contenere il suffisso "`_omp`" (es. `cfs32_omp.c`).

- Il software dovrà essere corredato da una relazione. Per la presentazione del progetto è possibile avvalersi di slide.

- Prima della data di consegna del progetto verranno pubblicate le convenzioni da rispettare riguardanti i nomi e la collocazione dei file/directory al fine della compilazione e l'esecuzione dei codici di programma mediante script appositamente predisposti. **Dato l'elevato numero di progetti, sarà cura del candidato accertarsi di aver rispettato pienamente le convenzioni di consegna.**

Buon lavoro!