Lingue

Progetto di Programmazione dell'Appello di Settembre 2017

Introduzione

In questo esame vi chiederemo di scrivere un insieme di classi per il riconoscimento linguistico basato sull'analisi delle frequenze. L'*analisi delle frequenze* è una tecnica di crittanalisi che sfrutta il fatto che le lettere dell'alfabeto hanno frequenze diverse in lingue diverse.

	Italiano	***Inglese***
E	11.79%	12.31%
A	11.74%	9.59%
I	11.28%	8.05%
O	9.83%	7.94%
N	6.88%	7.19%

In questo esempio vedete a confronto la frequenza relativa di alcune lettere nelle lingue inglese e italiano.

Classi

Descriviamo ora nei dettagli come realizzare le classi necessarie al progetto. Se lo ritenete, potete apportare cambiamenti alle segnature dei metodi proposti e anche – se lo ritenete – alla struttura delle classi, restando ovviamente all'interno dei requisiti esposti nel paragrafo introduttivo.

N.B. (1): Oltre ai metodi qui indicati, aggiungete a ogni classe un valido metodo `toString()` e, dove lo ritenete, anche un metodo `equals` e un metodo `hashCode` (con le segnature appropriate e che soddisfino i contratti previsti).

N.B. (2): Nel seguito, useremo il termine *lista di X* per indicare un qualunque tipo che consenta di conservare delle sequenze di oggetti di tipo `X` : potete implementare una lista con un array (`X[]`) oppure usando ad esempio delle `ArrayList<X>` (pacchetto `java.util`). Nelle segnature useremo sempre `X[]` , ma resta inteso che potete modificarlo in `ArrayList<X>` , purché lo facciate in modo

consistente (sostituite *tutti* gli array con delle liste).

Classe ModelloLinguistico

Un modello linguistico viene addestrato con uno o più testi, e tiene traccia di quante volte ogni lettera dell'alfabeto è comparsa nei testi analizzati (la sua *frequenza assoluta*) e di quante lettere in tutto si sono viste. Dividendo la frequenza assoluta per il numero di lettere si ottiene un numero chiamato *frequenza relativa*: la frequenza relativa è un valore fra 0 e 1, e la somma delle frequenze relative di tutte le lettere è 1. L'insieme di tutte le frequenze relative viene chiamata *distribuzione* del modello linguistico.

Il modello tiene traccia delle frequenze assolute, del numero di lettere viste e del numero di testi su cui il modello è stato addestrato.

- `ModelloLinguistico()` : crea un modello linguistico non addestrato su nessun testo.
- `ModelloLinguistico(String testo)` : crea un modello linguistico addestrato sul testo dato (vedi il metodo `addestra` per i dettagli).
- `void addestra(String testo)` : addestra il modello sul testo dato. Per farlo, considera uno alla volta i caratteri del testo, ignorando quelli che non siano alfabetici. Ogni carattere alfabetico viene minuscolizzato (ad es., 'A' viene convertito in 'a') e il contatore corrispondente viene incrementato.
- `int quantiTesti()` : restituisce il numero di testi su cui questo modello è stato addestrato fino ad ora.
- `int quantiCaratteri()` : restituisce il numero di caratteri alfabetici visti fino ad ora.
- `int frequenzaAssoluta(char c)` : restituisce la frequenza assoluta del carattere `c` , se `c` è una lettera alfabetica minuscola ('a'...'z'). Se `c` è un qualunque altro carattere, questo metodo restituisce -1.
- `double frequenzaRelativa(char c)` : se è stato visto almeno un carattere (cioè se `quantiCaratteri()` restituisce un valore maggiore di zero), questo metodo restituisce la frequenza relativa del carattere (cioè, la sua frequenza assoluta divisa per il numero di caratteri alfabetici visti); in caso contrario, restituisce 1.0/26.
- `double differenza(ModelloLinguistico m)` : calcola la differenza in norma L2 fra questo modello linguistico e il modello linguistico `m` . Tale differenza si calcola come segue: per ogni carattere alfabetico, si calcola la differenza fra la frequenza relativa di quel carattere nei due modelli coinvolti (questo e `m`), si eleva al quadrato questa differenza, e tali quadrati si sommano. Alla fine si restituisce la radice quadrata della somma dei quadrati (metodo statico `sqrt` della classe `Math`).

Classe Istogramma

Per rendere più semplice la visualizzazione di un modello linguistico, si può usare questa classe. Essa permette di visualizzare il modello come una serie di istogrammi, come ad esempio:

```
a   ******
b   ***
c   ****
d   **
...
```

- `Istogramma(int larg)` : crea un istogramma in cui ogni riga ha la larghezza indicata (come numero di caratteri); notate che i primi due caratteri di ogni riga sono costituiti dal carattere ('a', 'b', 'c', ecc.) e da un TAB. I restanti `larg-2` al massimo sono asterischi.
- `String toString(ModelloLinguistico m)` : crea una stringa che rappresenta il modello linguistico `m` . Tale stringa è costituita da 26 righe (ognuna terminata con un a-capo), una per ogni carattere. La riga della lettera `a` è data da un numero di asterischi proporzionale alla frequenza relativa di `a` .

Classe Addestratore

Questa classe mantiene una lista di modelli linguistici di varie lingue, e una lista parallela di nomi di lingue. Se vi serve, potete assumere che al massimo ci siano 100 lingue.

- `Addestratore()` : crea un addestratore vuoto.
- `void addestra(String nomeLingua, String testo)` : se la lingua `nomeLinga` è sconosciuta, crea un nuovo modello linguistico per quella lingua addestrandolo con il testo fornito; se invece la lingua è già conosciuta, si usa il testo per addestrare ulteriormente il modello corrispondente.
- `String classifica(String testo)` : costruisce un modello linguistico per il testo passato come argomento, e lo confronta (usando il metodo `differenza`) con tutti i modelli di tutte le lingue che in questo momento l'addestratore conosce. Restituisce il nome della lingua che ha dato luogo alla differenza minima.
- `void stampa(int larg)` : stampa tutti gli istogrammi (di larghezza `larg`) di tutte le lingue conosciute, ognuno preceduto dal nome della lingua.

Esempi

Si forniscono anche vari file di esempio, chiamati `mix-K.txt` (per diversi valori di K). Il file `mix-K.txt` è un file di testo contenente K righe. Ogni riga inizia con il nome di una lingua (per esempio, `english`) seguito da un TAB e quindi da un testo (senza a-capi) in quella lingua. Potete usare questi file per sperimentare con le vostre classi.