Data Analytics

Ilaria Battiston

Anno scolastico 2018-2019

Contents

1 Introduzione **3**

2 Data analytics issues **3**
2.1 Istanze, classi e attributi 3
2.2 Data analytics tasks 4

3 Data preprocessing **5**

4 Unbalanced data **5**
4.1 SMOTE 6
4.2 Tomek Link Method 6

5 Subset selection and evaluation **6**
5.1 Feature reduction 6
5.2 Feature extraction 6
5.3 Feature selection 7
5.4 Wrappers 7

6 Evaluation issues **8**
6.1 Numerosità del dataset 9

1 Introduzione

L'analisi dei dati è una disciplina introdotta nel 1935 da *Fisher* (F-test, per capire se due campioni sono originari della stessa popolazione) e in seguito sfruttata per automazione e gestione di dati prevalentemente non strutturati.

Le prime analisi descrittive risalgono alla fine degli anni '70, insieme ai primi software e al linguaggio R, il quale è successivamente stato integrato con tecnologie big data. Le prime modalità di visualizzazione (grafici a torta, istogrammi) nascono per dare una forma ai fenomeni legati al business (**business intelligence**).

Alla fine degli anni '90 questi modelli convergono nel **machine learning**, che permette non solo la creazione di modelli descrittivi ma anche previsioni e prescrizioni in base al contesto.

Negli ultimi decenni Google ha sviluppato strumenti come TensorFlow per poi renderli accessibili al pubblico, e sono nate diverse discipline e nuove applicazioni di data analytics (Google Flu Trends).

Alcuni utilizzi di questo campo sono i database transazionali per i sistemi di raccomandazione, IoT tramite wireless sensor data, o le analisi in ambito medico. Il volume di dati da elaborare è immensa, essendo questi prodotti in tempo reale dagli utenti e dai dispositivi.

Per **data analytics** si intende la generazione di valore da dati per scopi decisionali, cioè la trasformazione dei dati in prodotti. L'intervallo di tempo considerato include presente, passato e futuro, ed è necessaria una profonda comprensione empirica dei modelli.

I big data hanno la caratteristica di essere molto economici e in grande quantità, quindi hanno bisogno di velocità e disponibilità, con query ottimizzate. I sistemi di gestione sono prevalentemente NoSQL.

Più il dato è complesso, più crescono velocità e volume a discapito della varietà (e viceversa).

- Dati strutturati: hanno forma **proposizionale** (tabellare), permettono query con range numerici e matching esatto di stringhe;
- Dati non strutturati: **grafi** (reti) e **testo** libero, i quali compongono circa l'80% del totale dei dati disponibili in un'azienda;
- Dati semi-strutturati: via di mezzo tra strutturati e non. La maggior parte dei dati è libero, ma esistono linee guida per le rappresentazioni (ad esempio HTML).

2 Data analytics issues

2.1 Istanze, classi e attributi

Istanza (oggetto, record): esempio descritto da un numero di attributi.
Attributo (campo, caratteristica): misura gli aspetti delle istanze.
Classe: gruppo di istanze.

Le istanze sono tipi specifici di esempi che devono essere classificati, associati ed eventualmente raggruppati. Possono essere *dipendenti* o *indipendenti* e sono caratterizzate da un numero predeterminato di attributi: l'input ai modelli di apprendimento sono istanze contenute in un dataset,

basandosi su assunzioni **IID** (indipendenti e identicamente distribuite, derivano dalla stessa distribuzione).

La rappresentazione in forma proposizionale (tabellare) implica la definizione degli attributi, rimuovendo le relazioni ed esprimendole eventualmente tramite campi e condizioni. Viene considerato solo l'insieme delle osservazioni disponibili ("mondo chiuso").

Il processo di flattening di relazioni per creare un'unica tabella si indica con **proposizionalizzazione**: è possibile con qualsiasi insieme finito, ma causa "biased models" con irregolarità o dati replicati rispetto al modello originale. Le relazioni $1 : n$ sono gestite associando un campo aggiuntivo nel lato 1 oppure con matrici booleane. L'operazione di *join* generalmente causa rappresentazioni non totalmente veritiere (distorsioni), ma che hanno minore complessità computazionale.

Gli attributi sono le *features* che costituiscono lo spazio di rappresentazione dell'input: devono sempre avere lunghezza predefinita, eventualmente si ricorre all'uso di flag. L'esistenza di alcuni attributi (derivabili) può dipendere da altri, e questo potrebbe aumentare la complessità del modello.

Le variabili **nominali** (simboliche, categoriche, discrete) rappresentano una quantità nominale e non hanno relazioni logico-matematiche tra loro.
Le variabili **ordinali** (numeriche) impongono un ordine (numeri, stringhe).
Spesso non è immediata la distinzione tra nominali e ordinali.

Conoscere la natura dell'attributo è essenziale per poter effettuare confronti e avere un criterio per le operazioni, trattare i dati mancanti e gestire le problematiche legate alla qualità dei dati.

2.2 Data analytics tasks

2.2.1 Classification learning

Supervisionato, si occupa della classificazione di campioni predefiniti in classi, secondo approcci di machine learning (regressioni, alberi di decisione, reti bayesiane).

Il modello deve avere buone capacità di generalizzazione per input mai trattati prima, ed è possibile definire modelli di apprendimento in base a regole logiche su rappresentazioni proposizionali. Le regole logiche sono codificate usando *if*.

2.2.2 Clustering

Il clustering serve per identificare gruppi di istanze simili. Gli algoritmi sono non supervisionati: la classe di un esempio non è conosciuta in partenza. Vengono utilizzati per la segmentazione.

2.2.3 Associazione

Modello predittivo non supervisionato con l'obiettivo della comprensione di associazioni: dall'esistenza di un attributo prevedere l'esistenza di un altro.

2.2.4 Predizione numerica

Supervisionato, modelli con un valore target in input: cerca di individuare relazioni tra attributi numerici (regressione).

3 Data preprocessing

I dati solitamente contengono problematiche che devono essere affrontate prima di poterli dare in input al modello: il preprocessing è un'attività fondamentale per individuare rumore, inconsistenze e incorrettezze.

Il processo di **data cleaning** si occupa di rimpiazzamento di valori mancanti e smoothing dei dati rumorosi. Ci sono modelli in grado di gestire per natura i missing values, ma altri hanno necessariamente bisogno della completezza. Non tutti i dati incompleti possono essere sostituiti.

- MCAR (Missing Completely At Random): lla distribuzione di un esempio con valori mancanti non dipende da altri attributi;
- MAR (Missing At Random): la distribuzione di un esempio con valori mancanti dipende dagli attributi osservati, non necessariamente mancanti;
- NMAR (Not Missing At Random): la distribuzione di un esempio con valori mancanti dipende da attributi con valori mancanti.

I dati mancanti si possono ignorare, convertire a valori di default o rimpiazzare. Alcune tecniche di sostituzione implicano l'utilizzo della media (per valori continui con distribuzione normale) o della moda (discreti). Un altro modo è k-NN, che associa la classe sulla base della maggioranza degli oggetti vicini.

Un metodo di discretizzazione (smoothing) è il binning: divide il range in N intervalli in base alla media (distribuzioni normali) o alla frequenza (skewed).

4 Unbalanced data

Una delle problematiche che riguardano modelli di apprendimento predittivi e prescrittivi si verifica quando la distribuzione dei campioni di una determinata classe sono molto più frequenti rispetto a un'altra (es. contesto medico). La maggior parte dei campioni sono corretti, ma inutili: non è necessario fare analisi sulla maggioranza, sapendone già il comportamento.

Per bilanciare i dati si usano due tecniche:

- Oversampling: costruzione di un dataset con un numero desiderato di campioni dalla classe di minoranza e uno uguale dalle altre;
- Undersampling: eliminazione di un numero arbitrario di campioni dalla classe di maggioranza, in modo da avere lo stesso numero rispetto alla classe di minoranza.

Questi metodi sono applicabili con due classi, ma anche con un maggior numero. Ci sono algoritmi che si focalizzano su uno o sull'altro, e approcci ibridi che li combinano. Si assume che i dati siano corretti, rappresentativi e in assenza di rumore.

4.1 SMOTE

SMOTE (Synthetic Minority Oversampling Technique) è un algoritmo iterativo di oversampling:

1. Per ogni campione di minoranza, trova i suoi k elementi di minoranza più vicini;
2. Di queste ne sceglie n in modo casuale;
3. Calcola il punto medio tra quello iniziale e ciascuno degli n (Generated Synthetic Instance);
4. Il GSI viene aggiunto al dataset.

4.2 Tomek Link Method

Tomek Link è un algoritmo che si basa sulla frontiera della classe. Un Tomek Link è una coppia di istanze $< E_1, E_2 >$ tale che E_1, E_2 appartengano a una classe diversa e non esistano altri esempi E_k più vicini a ognuno di essi.

L'undersampling viene effettuato sui campioni della classe di maggioranza che non sono parte di Tomek Link.

5 Subset selection and evaluation

Questo processo consiste nella selezione di un sottoinsieme di attributi rilevanti per la costruzione del modello. Ciò comporta minore complessità e tempistiche èiù brevi.

5.1 Feature reduction

La riduzione e la modifica dello spazio di input è indispensabile, soprattutto per i modelli predittivi, per permettere una mappatura dall'input all'output (durante l'apprendimento): il target eredita delle caratteristiche, che non devono essere irrilevanti e ridondanti.

Le risorse computazionali sono ridotte (crescono esponenzialmente al numero delle variabili), e può essere complesso trovare le distribuzioni di probabilità dei campioni, quindi è meglio rimuovere alcuni attributi.

Riducendo le features, si riduce la dimensionalità dello spazio: gli attributi scelti devono essere sufficienti per distinguere i campioni tramite alberi di decisione (valori booleani).

5.2 Feature extraction

La feature selection trasforma attributi esistenti in uno spazio dimensionale minore. Dato un insieme di attributi $x = \{x_i | i = 1 \dots N\}$, si trova una mappatura $y = f(x) : R^N \to R^N$ con $M < N$ tale che il vettore trasformato preservi la maggior parte delle informazioni o della struttura.

Un mapping ottimale $y = f(x)$ risulterà in una minima probabilità di errori non incrementata, ma non esiste un modo sistematico per generare trasformazioni non lineari.

5.3 Feature selection

La feature selection seleziona un sottoinsieme degli attributi esistenti. Dato un insieme di attributi $x = \{x_i | i = 1 \dots N\}$, si trova un sottoinsieme $x_m = \{x_{i1}, x_{i2}, \dots, x_{iM}\}$ con $M < N$, che ottimizza una funzione obiettivo $J(Y)$.

Questo è necessario nei casi in cui le features siano difficili da ottenere o si voglia estrarre regole significative; è utile quando le unità di misura vengono perse o i dati non sono numerici.

Per trovare i candidati ottimali e i potenziali sottoinsiemi si ricorre al filtering: una ricerca esaustiva implica $\binom{n}{m}$ combinazioni, quindi si utilizzano metaeuristiche.

Criteri di ranking:

- Ranking variabile, ordinamento delle features in base a una funzione di scoring che misura la rilevanza e risulta in una permutazione ordinata;
- Correlazione, usando il coefficiente R di Pearson per m campioni, che misura la similarità e l'approssimazione a una funzione lineare;
  $R(f_i, y) = \frac{cov(f_i, y)}{\sqrt{var(f_i) var(y)}} \in [-1, 1]$
- Information Gain: classificazione in base all'informazione (entropia) guadagnata da ogni feature e alla sua contribuzione alla riduzione dell'incertezza della classe;
  $IG(C, A) = H(C) - H(C|A)$
- Subset evaluation tramite CFS (Correlation-based Feature Selection), selezione di g attributi altamente correlati tra loro ma incorrelati con le altre classi;
  $M_s = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k+1)\overline{r_{ff}}}}$ con k features, r_{cf} media della correlazione tra classi, r_{ff} la media dell'intercorrelazione tra features.

5.4 Wrappers

I wrappers sono metodologie per riconoscere le possibili interazioni tra le variabili, a discapito del tempo computazionale (al contrario dei filtri).

La subset evaluation viene effettuata tramite wrappers, i quali includono politiche per valutare le features rispetto al potere predittivo del modello di appendimento. Le performances devono soddisfare un determinato livello di qualità per poter accettare il sottoinsieme, e l'obiettivo è la massimizzazione della qualità.

I risultati possono variare a seconda della ricerca nello spazio delle possibili variabili, e l'output è strettamente dipendente dall'approccio utilizzato (paradigma di predizione). Questi parametri devono essere definiti e valutati attentamente, misurando man mano le performance e la robustezza del modello.

Fasi dell'algoritmo:

1. Selezione di un subset di attributi (uno o più);
2. Induzione di un algoritmo di apprendimento (albero di decisione, SVM, ...);
3. Valutazione dell'accuratezza o del risultato in generale;

4. Se essa non è sufficiente, ripetizione del processo.

Il problema della selezione del subset ottimale è NP-hard, quindi si ricorre a strategie metaeuristiche di ricerca. Le classi si dividono in forward e backward (aggiunta o rimozione di features nell'insieme), e la metodologia viene misurata tramite cross-evaluation, inducendo il wrapper sul training set e valutando sul test set.

La tecnica è comunque molto pesante computazionalmente, però permette un utilizzo semplice e universale dei wrappers.

6 Evaluation issues

Lo sviluppo di modelli per la valutazione delle features ha comportato la necessità di una misurazione accurata della predizione, introducendo errori e stime di performances.

Gli errori sul training data non sono buoni indicatori della qualità dell'apprendimento, perché non è possibile prevedere l'output con altri dati (la varianza è alta). I test set per questo sono utili a capire la variabilità dell'output con differenti input.

Problemi più comuni:

- Overfitting, quando il modello è molto accurato (troppo complesso) sui dati di training ma ha scarsi risultati con il testing e incapacità di generalizzazione;
- Underfitting, quando la varianza è nulla e tutti i dati sono interpretati allo stesso modo troppo generico (semplice), i risultati sono scarsi sia nel training che nel testing.

Nel momento in cui i risultati del training iniziano a divergere da quelli del testing, c'è la probabilità di overfitting. Il range ideale ha complessità ridotta del modello con poco scostamento delle curve.

Ci sono diversi metodi di valutazione degli errori:

- Misurazione delle performance, contando quante istanze sono classificate correttamente e in base a ciò calcolare error rate e accuracy;
- Falsi negativi/positivi, da minimizzare;
- Problemi multi-classe, osservando come il modello si comporta in base a ogni classe;
- Precisione, tenendo conto delle singole istanze con recall e self-measure;
- Curve ROC, grafici di veri positivi e falsi negativi al variare della soglia;
- Complessità computazionale.

Ognuno di questi è distinto dall'utilizzo di veri/falsi positivi e negativi, che contribuiscono al calcolo di alcuni indicatori di performance. Le misurazioni distinguono tra actual target values $a_1, a_2, \ldots, a_n$ e predicted target values $p_1, p_2, \ldots, p_n$.

$$\text{Mean-squared error: } \frac{(p_1 - a_1)^2 + \cdots + (p_n - a_n)^2}{n}$$

$$\text{Mean-absolute error (meno sensibile agli outliers): } \frac{|p_1 - a_1|^2 + \cdots + |p_n - a_n|^2}{n}$$

$$\text{Root ean-squared error: } \sqrt{\frac{(p_1 - a_1)^2 + \cdots + (p_n - a_n)^2}{n}}$$

Altre formule utilizzate:

1. Accuracy, $A = \frac{\text{numero di istanze correttamente classificate}}{\text{istanze totali}} = \frac{TP+TN}{TP+TN+FP+FN}$;
2. Precision, $P(k) = \frac{\text{numero di istanze correttamente classificate come } k}{\text{istanze totali classificate come } k} = \frac{TP+TN}{TP+FP}$;
3. Recall, $R(k) = \frac{\text{numero di istanze correttamente classificate come } k}{\text{istanze totali della classe come } k} = \frac{TP+TN}{TP+FN}$;
4. F-measure, $F(k) = \frac{2 \cdot P(k) \cdot R(k)}{P(k)+R(k)}$, distinta da:
   - Macro-average, $\frac{1}{N} \sum_{i=1}^{N} P(i)$, che consiste nella media di tutte le F-measures di ogni classe;
   - Micro-average, $\sum_{i=1}^{N} \frac{C_i}{N} P(i)$, per assegnare ad alcune classi una maggiore importanza (dove C è la cardinalità).

6.1 Numerosità del dataset

Generalmente il dataset, se sufficientemente grande, viene diviso in training e testing (solitamente 2/3 training, 1/3 testing) ed essi vengono utilizzati rispettivamente per costruire e valutare il classificatore.

Più aumenta di dimensioni il training set, più migliora il classificatore; più aumenta il testing set, più la stima degli errori è accurata. Dopo la valutazione, i dati vengono uniti per costruire la versione finale.

In mancanza di informazioni a riguardo, si assume che la distribuzione dei parametri sia uniforme, e poi si aggiusta la precisione in base alla performance.

Se il quantitativo di dati è piccolo, si usano strategie di repeated holdout: il processo di training viene ripetuto più volte con diversi sottoinsiemi di attributi, ma ciò causa overlap. A ogni istanza viene assegnato un numero casuale per l'assegnazione.

La cross-validation evita l'overlap dividendo i dati in k subsets (di solito 10) e dividendo ulteriormente essi in testing e training (k-fold), stratificando i gruppi e calcolando stime globali. La stratificazione permette il rispetto della cardinalità di ogni classe distribuendole comunque in modo relativamente uniforme.

Se un'istanza di minoranza viene valutata più volte in testing, la sua originale distribuzione marginale viene persa, e ci sono più previsioni per lo stesso dato.

Altre tecniche meno utilizzate sono quelle di bootstrap, con dati poco numerosi: ogni istanza uò far parte sia del set di training che di testing. Ogni elemento viene scelto più volte, il che è utile in presenza di classi di minoranza. La varianza è ridotta, ma i risultati tendono a essere pessimisti.

Una particolare istanza ha probabilità $1 - \frac{1}{n}$ di non essere selezionata, quindi la probabilità di essere selezionata è:

$$\left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0.368$$

Questo significa che il training data ha circa il 63.2% delle istanze.

Le curve di learning mostrano come l'accuratezza cambia al variare della dimensione del campione, stabilizzandosi dopo una certa quantità.

Quando ci sono davvero poche istanze si utilizza leave-one-out, una forma di cross-validation che crea tanti folds quante le istanze e ne usa uno solo come testing, iterativamente.