Intelligenza Artificiale

KDD99 e Classificatore Naive Bayes

Niccolò Guiducci

13 aprile 2020

1 Introduzione

Nella seguente relazione si andranno a confrontare le prestazioni di alcuni classificatori Naive Bayes con le prestazioni ottenute, nell'articolo del 2004, da Nahla Ben Amor, Salem Benferhat e Zied Elouedi nell'ambito del rilevamento di intrusioni in una rete. Il linguaggio di programmazione utilizzato è **Python**, appoggiandosi alle librerie **sklearn** e **pandas**. Sono stati studiati tre classificatori (Bernoulli, Multinomiale e Gaussiano) Naive Bayes tentando di eguagliare i risultati presenti nell'articolo; la scelta è ricaduta su questi ultimi essendo presenti nella libreria **sklearn**.

2 La Teoria

I classificatori Naive Bayes appartengono ad una famiglia di algoritmi di classificazione basati sul Teorema di Bayes. La caratteristica saliente di questi algoritmi è l'assunzione del principio secondo cui ogni caratteristica è indipendente l'una dall'altra, anche se, nella realtà dei fatti, questo potrebbe non essere vero. Con questa assunzione, date n variabili osservate $x_1 \cdots x_n$, la probabilità che l'evento y si verifichi è: $P(x_1 \cdots x_n|y) = \prod_{i=1}^{n} p(x_i|y)$

3 Il Dataset

Il dataset utilizzato per l'addestramento dei classificatori è il 10% del KDD99 reperibile direttamente dall'omonimo sito web: è composto da 494021 esempi con 41 features. I dati riportati sono sia simbolici che numerici(sia continui che discreti). Il dataset di training contiene in tutto 24 tipi di attacco appartenenti a 4 macro-categorie: DoS(Denial of Service), Probing(sorveglianza), U2R(accesso non autorizzato ad un superuser locale) e R2L(accesso remoto non autorizzato).
I dati richiesti per il test non sono stati presi dal dataset di training ma vengono forniti separatamente sul sito della KDD99 in un dataset di test; questo contiene anche 14 nuovi tipi di attacco che non sono presenti nel dataset di training.
In una nota del sito si viene avvertiti che il dataset è stato modificato nel 2006; non è esplicitamente scritto che siano state aggiunte o tolte determinate righe, ma si è osservato che il numero di esempi presenti nel dataset per ogni classe è diverso da quello riportato nell'articolo di riferimento.

Training	Articolo di Riferimento	Dati Riscontrati
Normal	19.65%	19.69%
Dos	79.07%	79.24%
Probing	0.83%	0.83%
R2L	0.23%	0.23%
U2R	0.22%	0.01%

Tabella 1: Distribuzione dei dati nel dataset di training.

4 Preparazione dei Dati

Il dataset si presenta in formato compresso .gz; si è quindi proceduto attraverso l'uso della libreria **pandas** all'estrazione e memorizzazione dei dati sotto forma di *Dataframe*(una particolare struttura dati di **pandas**). Fatto questo, il dataset, non si presenta comunque pronto all'uso, sono

Test	Articolo di Riferimento	Dati Riscontrati
Normal	19.48%	19.48%
Dos	73.90%	73.90%
Probing	1.34%	1.34%
R2L	5.21%	4.43%
U2R	0.07%	0.84%

Tabella 2: Distribuzione dei dati nel dataset di test.

mescolati dati numerici e categoriali simbolici. Si è quindi reso necessario trasformare tutti i dati simbolici in numerici e, in particolare, per il modello BernoulliNB, si sono trasformate tutte le features in features binarie.
Queste trasformazioni sono state ottenute sia attraverso alcune funzioni presenti nella libreria **sklearn** sia attraverso funzioni programmate ad-hoc. Per quanto riguarda la trasformazione dei dati simbolici si è utilizzato: nel caso del modello BernoulliNB una trasformazione *OneHot*, che crea un nuova feature binaria per ogni valore presente nella feature da trasformare, mentre, per i modelli MultinomialNB e GaussianNB si è utilizzato un trasformatore auto-prodotto simile all'*OrdinalEncoder* di **sklearn** (non è stato possibile utilizzare quello della libreria in quanto non può gestire il fatto che nel dataset di test compaiano dati che, invece, non compaiono mai nel dataset di training). Per i modelli che non possono gestire dati numerici continui si è optato per una discretizzazione uniforme in 24 intervalli per ogni feature numerica continua; per questa trasformazione si è utilizzata la funzione *KBinsDiscretizer* di **sklearn**.
Nel caso del BernoulliNB si rende necessaria un'ulteriore trasformazione delle feature numeriche (adesso discrete) in feature binarie, sempre attuata attraverso un *OneHot* encoder.
A questo punto il dataset è pronto per essere utilizzato per l'addestramento ma, se procediamo e tentiamo di predire il dataset di test ci si accorge che i risultati non sono ottimali: è necessario continuare a processare i dati.
Gli algoritmi Naive Bayes risentono molto del numero di feature e della loro correlazione ed è per questo che, per prima cosa, sono state eliminate le features che presentavano una bassa correlazione con le classi da predire. Le features da scartare sono scelte grazie alla funzione *SelectPercentile* di **sklearn** a seconda di un indice ANOVA: viene mantenuta una certa percentuale di features. L'ultima trasformazione dei dati riguarda la correlazione tra le features rimaste, se due o più features sono strettamente correlate è possibile mantenerne anche solo una. Si calcola, dunque, la matrice di correlazione e si eliminano le features che hanno indice di correlazione maggiore di una certa soglia.
Si procede dunque all'addestramento e all'analisi dei risultati.

4.1 Scelta dei parametri delle trasformazioni

Per le trasformazioni che richiedono parametri, come la discretizzazione e la scelta delle features secondo la correlazione tra le stesse, questi sono stati scelti attraverso la riesecuzione dei test scegliendo quelli che massimizzano l'accuratezza di predizione. In particolare:

Percentuale di features mantenute I modelli sono stati testati tra percentuali che vanno dal 25% al 95% incrementando ad ogni iterazione del 5%, BernoulliNB e MultinomialNB sono risultati ottimali con il 60% delle features mantenute, mentre, GaussianNB con il 55%;

Coefficiente di Correlazione I modelli sono stati testati con coefficienti di correlazione a partire da 0.5 fino a 0.95, il BernoulliNB è risultato ottimale con un coefficiente di 0.85, mentre, il MultinomialNB con 0.90 e il GaussianNB con 0.65;

Numero di Intervalli di Discretizzazione Questo parametro, riguardante solo il modello BernoulliNB, è stato scelto tra valori che vanno da 2 a 50, l'accuratezza massimale è raggiunta con la divisione in 24 intervalli per ogni feature.

5 Risultati

Si ricorda che il numero totale di classi da predire è 39 ma quest'analisi non si concentra sul riconoscimento specifico di un attacco ma sulla classificazione di una connessione in 5 macro-classi: *normal, DoS, Probe, U2R* e *R2L*. I risultati di seguito riportati sono ottenuti da diverse strategie di addestramento dei modelli:

Correlazione si è provato ad addestrare i classificatori sia senza la selezione delle feature migliori, sia senza la diminuzione del numero di features;

Raggruppamento prima della classificazione le classi da predire sono raggruppate nelle 5 macro-classi prima di addestrare i classificatori;

Raggruppamento dopo la classificazione le 39 classi sono mantenute prima dell'addestramento per poi essere raggruppate dopo la predizione del dataset di test.

5.1 Bernoulli Naive Bayes

Bernoulli Train.	N. Feature	Acc_Prima	Acc_Dopo
Senza Correlazione	635	94.29%	99.262%
Solo Migliori Feature	451	97.31%	99.37%
Tutte le Trasformazioni	264	99.09%	99.72%

Tabella 3: Accuratezze di predizione, in percentuale, del classificatore BernoulliNB sui dati di training.

Bernoulli Test	Acc_Prima	Acc_Dopo
Senza Correlazione	79.22%	92.05%
Solo Migliori Feature	91.27%	92.34%
Tutte le Trasformazioni	91.95%	92.67%

Tabella 4: Accuratezze di predizione, in percentuale, del classificatore BernoulliNB sui dati di test.

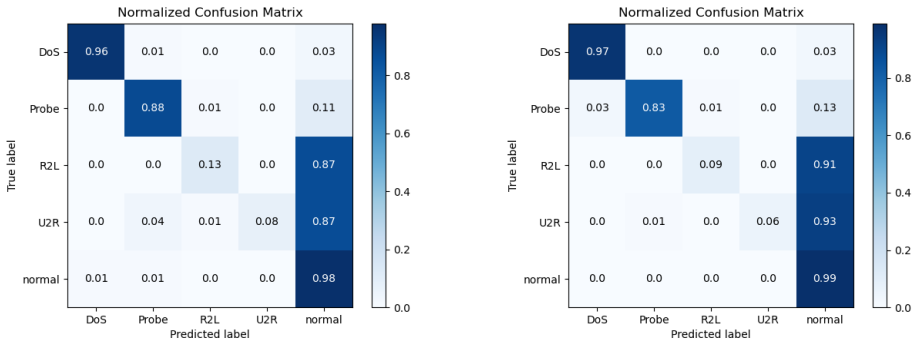

Figura 1: Matrice di confusione con raggruppamento PRIMA e DOPO l'addestramento rispettivamente.

5.2 Multinomial Naive Bayes

Multinomial Train.	N. Feature	Acc_Prima	Acc_Dopo
Senza Correlazione	41	91.53%	84.86%
Solo Migliori Feature	24	92.50%	86.50%
Tutte le Trasformazioni	16	96.63%	92.43%

Tabella 5: Accuratezze di predizione, in percentuale, del classificatore MultinomialNB sui dati di training.

Multinomial Test	Acc_Prima	Acc_Dopo
Senza Correlazione	78.27%	70.74%
Solo Migliori Feature	78.30%	72.13%
Tutte le Trasformazioni	91.08%	87.17%

Tabella 6: Accuratezze di predizione, in percentuale, del classificatore MultinomialNB sui dati di test.

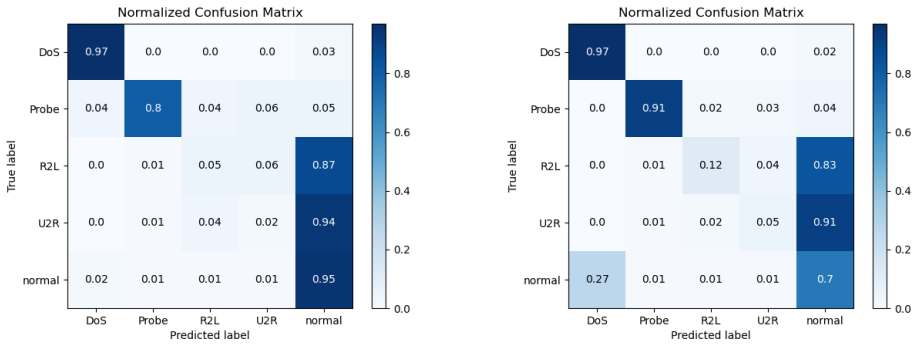

Figura 2: Matrice di confusione con raggruppamento PRIMA e DOPO l'addestramento rispettivamente.

5.3 Gaussian Naive Bayes

Per questioni di spazio non si riportano per intero i risultati ottenuti con GaussianNB essendo stati questi ultimi i peggiori sui dati di test: Acc_Prima del 85.2% e Acc_Dopo del 90.4%.
Forse unico merito di questo modello è la percentuale di predizione della classe *R2L* nel caso del raggruppamento post-predizione che ha raggiunto l' 88%.

5.4 Analisi dei Risultati

I risultati ottenuti sono in linea con quelli dell'articolo di riferimento. Il modello con le prestazioni migliori si è dimostrato essere il BernoulliNB che per quanto riguarda il PCC(Percent of Correct Classification) ha addirittura superato i risultati ottenuti nell'articolo. Se però osserviamo la matrice di confusione di BernoulliNB ci si accorge che la percentuale di predizione della classe *U2R* è più bassa di quella di riferimento: questo può essere dovuto sia alla struttura del modello stesso sia al fatto che il numero di esempi nel dataset di training per questa classe sia molto basso(Tabella 1), anche rispetto al numero di esempi dichiarati nell'articolo di riferimento.

6 References

1. Naive Bayes vs decision trees in intrusion detection systems. Nahla Ben Amor, Salem Benferhat e Zied Elouedi.
2. scikit-learn documentation: https://scikit-learn.org/stable/index.html
3. pandas documentation: https://pandas.pydata.org/pandas-docs/stable/index.html