Politecnico di Milano

V Facoltà di Ingegneria

Corso di Laurea Specialistica in Ingegneria delle Telecomunicazioni

CLASSIFICAZIONE DEL TRAFFICO INTERNET CON ATTRIBUTI PER-SOURCE

Relatore: Chiar.mo Prof. Giacomo Verticale

Tesi di Laurea di: Cristina Rottondi

Matr. Nr. 731321

Anno Accademico 2009-2010

In ogni cosa ho voglia di arrivare
sino alla sostanza.
Nel lavoro, cercando la mia strada,
nel tumulto del cuore.

Sino all'essenza dei giorni passati,
sino alla loro ragione,
sino ai motivi, sino alle radici,
sino al midollo.

Eternamente aggrappandomi al filo
dei destini, degli avvenimenti,
sentire, amare, vivere, pensare,
effettuare scoperte.

Oh, se mi fosse dato, se potessi
almeno in parte,
mi piacerebbe scrivere otto versi
sulle proprietà della passione.

Sulle trasgressioni, sui peccati,
sulle fughe, sugli inseguimenti,
sulle inavvertenze frettolose,
sui gomiti, sui palmi.

Dedurrei la sua legge,
il suo cominciamento,
dei suoi nomi verrei ripetendo
le lettere iniziali.

I miei versi sarebbero un giardino.
Con tutto il brivido delle nervature
vi fiorirebbero i tigli a spalliera,
in fila indiana, l'uno dietro l'altro.

Introdurrei nei versi la fragranza
delle rose, un alito di menta,
ed il fieno tagliato, i prati, i biodi,
gli schianti della tempesta.

Così Chopin immise in altri tempi
un vivente prodigio
di ville, di avelli, di parchi, di selve
nei propri studi.

Giuoco e martirio
del trionfo raggiunto,
corda incoccata
di un arco teso.

(Boris Pasternak)

Abstract

Internet traffic classification techniques aim at associating to a packet sequence between two hosts and two corresponding transport ports (flow) the generating application. This is an important tool for traffic and Quality of Service management, network monitoring and security policies implementation. The most widely used classification techniques are based on packet payload inspection or assume the use of well known transport ports. However, nowadays a lot of protocols and applications use random port numbers and cipher packet payloads, making these techniques useless. Therefore, the recent literature proposes the use of observable attributes of traffic patterns, such as packet sizes and interarrival times. As the formulation of models based on such attributes is extremely complex, it is usually achieved by using Machine Learning algorithms. Classification can be performed online or offline: in the first case, classification delays are of primary importance, as it is typically required to obtain a result by only considering the first few packets of the flow, thus minimizing the time elapsed between the beginning of the flow and its identification. On the other hand, the more information is available, the more the classification performance is improved, resulting in a trade-off between delay and accuracy.

This work proposes a classification technique for the outgoing traffic from a server host on a specific port. After evaluating the characteristics of different classifiers (e.g. Support Vector Machines and Random Forests) and statistical attributes, we

focused on a single classification feature, the Index of Variability, which has been proposed in literature for characterizing the autocorrelation of a traffic process. This parameter has been evaluated for different time scales and the obtained measurements have been used to train a Parzen classifier, which has been proved to provide good performance. The classification error has been analytically computed and compared to results obtained with synthetic data.

Sommario

Le tecniche di classificazione del traffico Internet associano ad una sequenza di pacchetti scambiati tra due host e due rispettive porte di trasporto (*flusso*) la presunta applicazione generatrice, la cui identificazione riveste elevata importanza in innumerevoli ambiti, dalla gestione della Qualità del Servizio al monitoraggio di rete.

Le tecniche più comunemente sfruttate per il riconoscimento delle applicazioni si basano sull'ispezione del payload dei pacchetti o sull'utilizzo di porte note. Tuttavia, ad oggi molti protocolli ed applicazioni sfruttano numeri di porta casuali ed effettuano la cifratura del payload, rendendo queste tecniche inapplicabili. La letteratura più recente propone perciò l'utilizzo di attributi statistici, quali la lunghezza dei pacchetti o i tempi di interarrivo. Poichè la formulazione di modelli basati su tali grandezze si è rivelata di estrema complessità, essa viene effettuata servendosi di algoritmi di Machine Learning.

L'operazione di classificazione può essere realizzata online oppure offline: nel primo caso, le tempistiche rivestono un'importanza cruciale, giacchè tipicamente si richiede di pervenire ad un risultato considerando solo i primi pacchetti del flusso e dunque minimizzando il ritardo introdotto tra l'inizio del flusso e la sua identificazione. D'altro canto, le prestazioni dei classificatori migliorano al crescere delle informazioni disponibili, rendendo dunque necessario un trade-off tra ritardo e accuratezza.

Il presente lavoro propone una tecnica di classificazione basata sull'osservazio-

ne del traffico uscente dalla coppia host-porta lato server. Dopo aver valutato le caratteristiche di diversi classificatori e l'impatto di vari attributi di classificazione, è stata focalizzata l'attenzione sull'Indice di Variabilità, un parametro proposto in letteratura per caratterizzare l'autocorrelazione di un processo di traffico. Esso è stato valutato per diverse scale temporali e le misure ottenute sono state utilizzate per addestrare un classificatore di Parzen, il quale ha dato prova di fornire buone prestazioni. l'errore di classificazione è stato valutato analiticamente e confrontato con i risultati ottenuti con dati sintetici.

Capitolo 1

INTRODUZIONE

Le tecniche di classificazione del traffico IP appartengono ad un ampio insieme di strumenti che hanno come obiettivo la risoluzione dei complicati problemi di gestione della rete con cui devono confrontarsi gli *Internet Service Provider* (ISP), i quali necessitano di conoscere il tipo di applicazioni transitanti nella rete in modo da poter prontamente reagire in supporto ai vincoli di qualità e sicurezza stabiliti con gli utenti. Avvalendosi di tali tecniche, essi possono così allocare, controllare e gestire efficacemente le risorse nelle reti TCP/IP, nonchè migliorare l'affidabilità dei *Network Intrusion Detection Systems* (NIDS). In aggiunta, recentemente le autorità governative hanno stabilito nuovi obblighi degli ISP rispetto alla *lawful interception* (LI) del traffico IP: come già accade per le compagnie telefoniche, anche agli ISP possono essere richieste informazioni sull'uso della rete da parte di determinati utenti in alcuni periodi di tempo d'interesse.

Per la classificazione del traffico IP possono essere usate differenti metodologie: la più semplice consiste nell'identificare l'applicazione che genera un flusso di traffico mediante la porta sorgente e destinazione del livello di trasporto, considerando pacchetti IP consecutivi con la stessa 5-tupla (*protocol, source address, source port, destination address, destination port*) come appartenenti allo stesso flusso. Questo criterio di classificazione si basa quindi sull'uso di porte note, ma la costante crescita di nuove applicazioni Internet (tra cui ad esempio quelle peer-to-peer) che fanno uso di porte di livello di trasporto casuali o comunque non assegnate dallo IANA

(Internet Assigned Numbers Authority) o che sfruttano tunnel HTTP per superare firewall e NAPT box ne ha evidenziato le lacune intrinseche.

Un secondo tipo di approccio consiste invece nel riconoscere le applicazioni tramite l'ispezione del payload di ciascun pacchetto. Questa soluzione ha lo svantaggio di richiedere elaborazioni computazionalmente onerose, che devono per di più essere effettuate con velocità proporzionale alla capacità dei link, risultando quindi di limitata utilità nelle reti moderne a larga banda. Inoltre, l'approccio di tipo Deep Packet Inspection è per definizione fondato sulle seguenti due ipotesi:

1. terze parti devono essere in grado di ispezionare il payload di ogni pacchetto IP;

2. il classificatore deve conoscere la sintassi di ogni applicazione all'interno del payload del pacchetto.

Attualmente, la prima ipotesi è minata dal crescente utilizzo di algoritmi crittografici per la cifratura dei pacchetti (inclusi i numeri di porta TCP o UDP) e dalle eventuali regolamentazioni in difesa della privacy imposte dalle autorità competenti, che possono limitare la capacità di parti terze di ispezionare i payload. La seconda ipotesi impone inoltre un pesante carico operazionale, costringendo gli ISP ad aggiornare frequentemente gli strumenti di analisi per stare al passo con le modifiche apportate ai formati delle applicazioni all'interno del payload dei pacchetti.

Negli ultimi anni è dunque emersa la necessità di sfruttare nuove metodologie di classificazione, basate esclusivamente sull'analisi delle proprietà statistiche del flusso dei pacchetti, modellizzato come un processo casuale. Con questa tecnica ogni flusso di traffico è associato ad un insieme di attributi e, tramite l'analisi dei valori di tali attributi, il classificatore può attribuire al flusso in esame l'applicazione d'appartenenza più probabile.

Un classificatore basato su un approccio di tipo statistico può essere realizzato con algoritmi di apprendimento automatico (*Machine Learning*) che offrono una va-

lida alternativa alle tecniche di classificazione appena descritte, sebbene, nel caso di apprendimento di tipo supervisionato, sia necessario effettuare la raccolta preventiva di un rappresentativo numero di flussi delle applicazioni che si desidera vengano riconosciute dal classificatore. La principale difficoltà consiste proprio nel reperire un sufficiente numero di flussi di appartenenza certa alle classi di interesse, in modo da fornire una base di verità affidabile: una possibile soluzione è quella di registrare varie tracce di traffico ed usare tecniche di ispezione manuale dei payload *offline* per etichettare i dati di training.

Un altro problema derivante dall'uso di algoritmi di Machine Learning è rappresentato dal fatto che le proprietà statistiche dei flussi di traffico variano da link a link, il che comporta un degrado delle prestazioni del classificatore, se fatto operare su mix di traffico diversamente caratterizzati dal punto di vista statistico rispetto a quello di addestramento. Infine si segnala il problema relativo al tempo di raccolta dei pacchetti: infatti, per essere classificati, i flussi devono essere osservati per un intervallo di tempo tale da consentire la raccolta di una sufficiente quantità di informazioni. In particolare, la valutazione del tempo di classificazione riveste un'importanza cruciale nel caso si operi una classificazione *online*, in cui vengono presi in considerazione soltanto i primi pacchetti del flusso e viene tipicamente richiesto di minimizzare il tempo intercorrente tra l'inizio dell'osservazione e l'istante in cui è reso disponibile l'esito della classificazione.

Nell'ambito della classificazione statistica si colloca anche il presente lavoro di tesi, svolto presso il Dipartimento di Elettronica e Informazione del Politecnico di Milano durante l'anno accademico 2009/2010, oggetto del quale è la classificazione del traffico Internet sulla base di attributi statistici per-sorgente. Esso si propone di valutare l'impatto che la scelta della tipologia di classificatore e degli attributi di classificazione ha sulle prestazioni del classificatore stesso, provando l'utilità di sfruttare parametri statistici standard quali la lunghezza dei pacchetti o i tempi di interarrivo tra i medesimi, associati ad altri caratterizzanti l'attività di una sorgente

di traffico nel suo complesso. L'introduzione di questi nuovi attributi per-sorgente permette dunque di ottenere informazioni addizionali sui flussi di traffico e di ridurre potenzialmente il numero di parametri standard necessari per pervenire ad una classificazione sufficientemente accurata, il che si traduce in una diminuzione del numero di pacchetti osservati per ciascun flusso e conseguentemente del ritardo di classificazione. E' stata quindi focalizzata l'attenzione su un particolare parametro statistico per-sorgente, l'Indice di Variabilità, il quale può essere considerato una generalizzazione del parametro di Hurst valutato su diverse scale temporali e descrive il grado di Long Range Dependence di un processo di traffico. Il calcolo dell'Indice di Variabilità è stato specializzato al caso particolare di una sorgente di traffico iperesponenziale di ordine 2 (RPH2) e per esso è stato definito uno stimatore, del quale sono state calcolate analiticamente le espressioni di media e varianza. Infine, è stato definito un classificatore di Parzen binario atto a discriminare due classi di traffico in funzione dei soli valori assunti dall'Indice di Variabilità: per tale classificatore è stata calcolata analiticamente la probabilità d'errore e i risultati raggiunti sono stati confrontati con i dati sperimentali ottenuti con traffico generato sinteticamente.

La presentazione del lavoro svolto è organizzata come segue: nel capitolo 2 verrà fornita una breve panoramica dello stato dell'arte e i riferimenti necessari per una trattazione più approfondita dell'argomento, oltre all'esposizione dei concetti teorici che stanno alla base della teoria del riconoscimento applicata alla classificazione del traffico.

Nel capitolo 3 verrà invece descritto l'ambiente di sperimentazione, i dati di traffico e i software di elaborazione utilizzati ai fini sperimentali.

Il capitolo 4 introdurrà l'utilizzo di una tipologia ibrida di classificatore, in grado di sfruttare parametri statistici per-flusso e per-sorgente, e ne illustrerà le prestazioni tramite risultati sperimentali ottenuti con due diversi algoritmi di apprendimento (Random Forests e Support Vector Machines).

Sulla base di tali risultati, nel capitolo 5 verrà definito un classificatore di Parzen

binario, finalizzato al riconoscimento del traffico in uscita da un server su una data porta, che utilizza come attributo di classificazione l'Indice di Variabilità valutato per diverse scale temporali. Di tale classificatore verranno calcolate analiticamente le prestazioni e i risultati ottenuti saranno comprovati da test sperimentali effettuati con tracce di traffico sintetiche.

Il capitolo 6 infine presenterà le conclusioni generali sul lavoro svolto.

Capitolo 2

STATO DELL'ARTE

2.1 Articoli correlati

La letteratura scientifica nell'ambito della modellizzazione e della misurazione del traffico Internet ha ampiamente dimostrato che le caratteristiche statistiche di quest'ultimo sono strettamente dipendenti dall'insieme di applicazioni generatrici. Una panoramica esaustiva sulle problematiche relative alla classificazione del traffico può essere reperita in [1], che analizza 18 articoli pubblicati tra il 2004 e il 2007, fornendo una tassonomia delle tecniche basate su algoritmi di Machine Learning attualmente utilizzate. Una rassegna di tecniche per-flow per scopi di intrusion detection viene fornita in [2], mentre un confronto critico tra approcci di classificazone per porta, per flusso e comportamentali è presentato in [3] e [4], il quale evidenzia inoltre l'impatto della collocazione geografica e temporale sull'accuratezza di classificazione. In questa sezione viene presentato un breve sommario dei contributi scientifici più significativi, citando anche i risultati fondamentali riguardanti il legame intercorrente tra sorgenti di traffico di tipo Long Range Dependent e spettro della Legge di Potenza.

Roughan et al. [5] utilizzano gli algoritmi Nearest Neighbours (NN), Linear Discriminate Analysis (LDA) e Quadratic Discriminate Analysis (QDA) per identificare la classe QoS di varie applicazioni. Gli autori individuano una lista di possibili attributi calcolati considerando l'intera durata del flusso e, nei risultati riportati,

ottengono errori di classificazione compresi tra il 2.5% e il 12.6%, in funzione del numero di classi QoS.

Auld et al. [6] propongono l'applicazione di una rete neurale Bayesiana: l'accuratezza di classificazione di questa tecnica raggiunge il 99% se i dati di train e di test sono raccolti nella medesima giornata, mentre si attesta attorno al 95% quando la raccolta dei dati di test è posticipata di 8 mesi rispetto a quella dei dati di train.

Gli autori di [7] sfruttano un nuovo metodo di classificazione basato sui più recenti n pacchetti del flusso: gli attributi considerati sono le statistiche riguardanti lunghezze e tempi di interarrivo di tali pacchetti. L'accuratezza ottenuta è circa il 98%, ma le performances risultano scarse se i pacchetti iniziali del flusso di traffico non vengono forniti al classificatore. E' stato dunque proposto un metodo di training che sfrutti attributi statistici calcolati su più sotto-flussi estratti dal flusso originale, il che non ha comunque introdotto miglioramenti apprezzabili delle prestazioni di classificazione.

Park et al. [8] [9], si servono di un Algoritmo Genetico (GA) per selezionare gli attributi più promettenti, confrontando tre classificatori: Naive Bayes con Kernel Estimation (NBKE), l'albero decisionale C4.5 e il Reduced Error Pruning Tree (REPTree). I migliori risultati sono stati ottenuti con il classificatore C4.5 e calcolando gli attributi sulla base dei primi 10 pacchetti del flusso.

Zander et al. [10] utilizzano un algoritmo di apprendimento non supervisionato Bayesiano (Autoclass) per l'identificazione di 8 diverse applicazioni (FTP, Telnet, SMTP, DNS, HTTP, AOL Messenger, Napster, Half-Life) in tracce di traffico reale, ottenendo un'accuratezza media pari all'86.5%, con picchi che raggiungono il 95% per alcune classi.

Crotti et al. [11] propongono una tecnica chiamata Protocol Fingerprinting, che prende in considerazione lunghezze, tempi di interarrivo e ordine di arrivo dei pacchetti. Classificando flussi generati da 3 diverse applicazioni (HTTP, SMTP e POP3), gli autori ottengono un'accuratezza non inferiore al 91%.

Verticale e Giacomazzi [12] usano l'albero decisionale C4.5 per classificare traffico

WAN. Gli attributi considerati sono le lunghezze e i tempi di interarrivo dei primi 5 pacchetti in entrambe le direzioni, ottenendo un'accuratezza compresa tra il 92% e il 99%.

Soysal et al. [13] confrontano le prestazioni di tre algoritmi di apprendimento (Bayesian Networks, Decision Trees e Multilayer Perceptrons) effettuando una classificazione per-flow su 6 diverse classi di traffico (tra cui P2P e Akamai content delivery) e analizzano la dipendenza dell'accuratezza di classificazione dalla quantità e dalla composizione dei dati di training.

Huang et al. [14] propongono l'utilizzo dell'algoritmo di apprendimento supervisionato K-Nearest-Neighbor addestrato con attributi statistici di tipo header-derived, ottenendo un'accuratezza attorno al 90%.

Holanda Filho et. al [15] introducono diversi insiemi di attributi statistici che possono essere selezionati adattativamente in funzione delle classi di traffico da discriminare e sviluppano un classificatore binario di tipo K-Means in grado di identificare il traffico P2P.

Algoritmi di apprendimento supervisionato e non supervisionato possono essere combinati come proposto da Yuan et al. in [16], che utilizzano l'algoritmo Support Vector Machines per incrementare dal 93.8% al 97% l'accuratezza ottenuta tramite un processo di clustering.

Un metodo di classificazione innovativo, basato sul comportamento dell'host a livello di trasporto è proposto in [17]: vengono utilizzati attributi di classificazione a livello sociale, funzionale e applicativo ed i risultati mostrano un'accuratezza superiore al 95% per l'80-90% delle tracce di traffico considerato.

Per quanto riguarda la scelta degli attributi statistici di classificazione, Lazarou et al. [18] introducono il parametro Indice di Variabilità, in grado di evidenziare le variazioni di burstiness del traffico di rete in funzione di diverse scale temporali, definendone la forma analitica per due modelli di sorgente: il processo di Poisson modulato su una catena di Markov a due stati (MMPP) e l'iperesponenziale di ordine 2 (RPH2). Viene inoltre dimostrata l'indipendenza di tale parametro dal

numero di flussi costituenti il processo in esame, in caso essi siano indipendenti ed identicamente distribuiti. I risultati analitici sono supportati da stime estrapolate da dati sperimentali su scale temporali nel range $[10^{-1} : 10^{3}]$s.

Nell'ambito dell'indagine sulla relazione tra natura dell'applicazione e spettro della Legge di Potenza, Leland et al. [19] sono giunti per primi alla conclusione che l'andamento di tipo power-law dello spettro del traffico di una rete LAN è causato dalla natura stessa delle applicazioni per il trasferimento dati.

Paxson e Floyd [20] identificano uno spettro di tipo power-law a livello di pacchetto anche per il traffico WAN, concludendo inoltre che, a livello di connessione, le connessioni di controllo FTP e Telnet possono essere modellizzate come un processo di Poisson, mentre le connessioni dati FTP, NTP e SMTP presentano caratteristiche differenti.

Crovella e Bestavros [21] misurano il traffico web-browsing studiando la sequenza di richieste di file durante ciascuna sessione, ove per sessione si intende una singola esecuzione dell'applicazione di web-browsing, giungendo alla conclusione che il motivo dell'andamento di tipo power-law è dato dalla distribuzione long-tailed dei files richiesti e dai tempi morti introdotti dall'utente.

Nuzman et al. [22] analizzano l'attività di web-browsing dell'utente a livello di connessione e di sessione, considerando come sessione un insieme di connessioni generate da un dato indirizzo IP. Gli autori concludono che il processo degli arrivi di connessione è di Poisson, mentre a livello di connessione si presenta un comportamento di tipo power-law.

2.2 Nozioni basilari sugli algoritmi di Machine Learning

Il Machine Learning (o *apprendimento automatico*) è il settore della Computer Science che studia gli algoritmi capaci di emulare le modalità di ragionamento tipiche dell'uomo quali riconoscere, decidere, scegliere, ossia apprendere ed estrarre infor-

mazioni su un determinato problema esaminando una serie di esempi ad esso relativi. La messa a punto di dispositivi artificiali capaci di apprendere riveste un'importanza tale da costituire uno degli obiettivi primari di diversi settori scientifici, tra cui la statistica (studio del mercato e delle vendite), l'intelligenza artificiale e le scienze cognitive (motori di ricerca, diagnosi mediche, riconoscimento del testo).

Apprendere consiste nell'effettuare un ragionamento di tipo induttivo, che passa cioè dagli esempi alle regole. Questo passaggio può essere visto come un processo di estrazione di informazione da un insieme di esempi, individuando e descrivendo utili schemi strutturali a partire dai dati.

Un algoritmo di machine learning deve dunque avere l'abilità di imparare automaticamente dall'esperienza e migliorare così la sua conoscenza di base: in particolare, un programma viene considerato capace di imparare dall'esperienza (**E**) acquisita rispetto ad un dato obiettivo (**O**) ed a una data misura di prestazioni (**P**), se le sue prestazioni nel pervenire ad **O**, valutate tramite **P**, migliorano grazie ad **E**. Tipicamente, l'obiettivo **O** consiste nell'estrarre un modello di un fenomeno desiderato a partire da un insieme finito di dati di addestramento, al fine di poterlo utilizzare per future predizioni o come supporto decisionale, mentre **E** è costituito da un insieme di esempi del comportamento desiderato del modello, preprocessati da un esperto in modo da essere espressi in funzione di coppie input-output, e la performance **P** è la misura della distanza tra l'output desiderato e quello fornito dal modello, quando esso viene applicato a dati diversi da **E**.

Nel contesto della classificazione del traffico Internet, gli algoritmi di Machine Learning vengono utilizzati per effettuare un mapping tra istanze, costituite dai singoli flussi di traffico, e differenti categorie di traffico di riferimento. Ciascun flusso può essere caratterizzato da un insieme di attributi per-flow (ad esempio le lunghezze dei primi pacchetti) o per-source (parametri statistici calcolati in funzione di uno o più pacchetti, che caratterizzano la singola sorgente). Ogni flusso sarà quindi associato allo stesso insieme di attributi, che tuttavia assumeranno valori differenti in funzione della classe di appartenenza.

Gli algoritmi ML utilizzati per la classificazione del traffico IP possono essere ricondotti a due tipologie fondamentali: metodi di apprendimento superivisionato o non supervisionato. La prima prevede che la classe di appartenenza di alcune istanze rappresentative sia nota a priori, in modo da poter effettuare l'addestramento del classificatore e costruire quindi un modello di classificazione in grado di etichettare correttamente nuovi flussi. La seconda raggruppa invece le varie istanze in clusters, in funzione della somiglianza tra i valori assunti dai vari attributi.

Durante il lavoro di tesi, è stata focalizzata l'attenzione su tre algoritmi di apprendimento supervisionato: Support Vector Machines (SVM), Random Forests (RF) e Parzen, dei quali si fornisce nelle sezioni seguenti una breve descrizione.

2.2.1 Support Vector Machines (SVM)

Le Support Vector Machines (SVM) definscono un'architettura per la classificazione di pattern non linearmente separabili in due classi con il minimo errore rispetto all'insieme di addestramento. Esse costituiscono una tecnica di apprendimento supervisionato molto diffusa, inizialmente concepita per classificazioni binarie [23] e successivamente adattata al caso multiclasse [24]. Una trattazione teorica esaustiva può essere reperita in [25] e [26] .

SVM lineare a due classi

Si consideri un classificatore binario per classi linearmente separabili con l'insieme di addestramento $F(\mathbf{x}_i, d_i)$ per $i = 1, ..., N$, dove $(\mathbf{x}_i, d_i)$ è la coppia campione-classe con $d_i = -1$ o $d_i = +1$. L'iperpiano discriminante è:

$$\mathbf{w}^T\mathbf{x} + b = 0$$

ove $\mathbf{x}$ è il vettore di ingresso, $\mathbf{w}$ il vettore di pesi e il parametro $\frac{b}{||\mathbf{w}||}$ definisce la distanza tra l'origine e l'iperpiano lungo la direzione del vettore $\mathbf{w}$. Di conseguenza, per l'i-esimo pattern di ingresso $\mathbf{x}_i$ risulta:

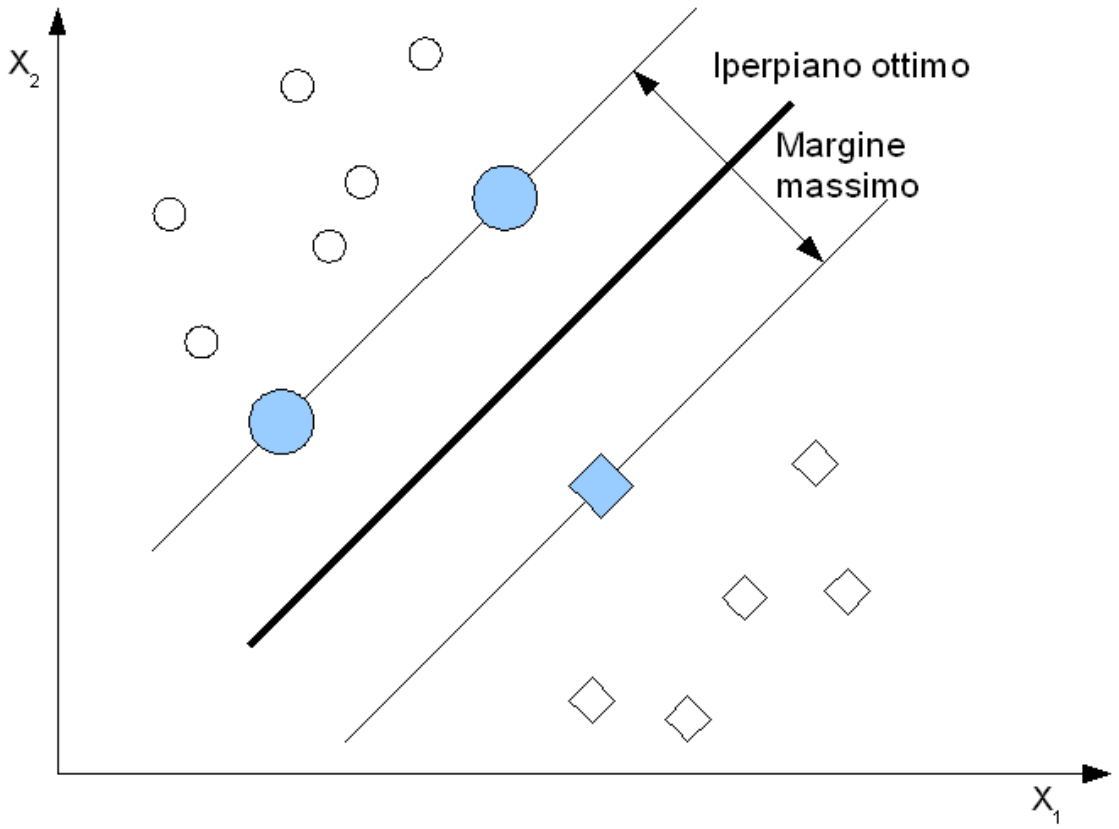

Figura 2.1: *Iperpiano ottimo di separazione tra due classi*

$$\mathbf{w}^T\mathbf{x}_i + b > 0, \text{ per } d_i = +1;$$
$$\mathbf{w}^T\mathbf{x}_i + b < 0, \text{ per } d_i = -1.$$
$$\mathbf{w}^T\mathbf{x}_i + b = 0 \text{ classificazione impossibile.}$$

Dati $\mathbf{w}$ e b, si definisce margine di separazione ρ_o la distanza tra l'iperpiano discriminante ed il punto $\mathbf{x}_i$ più vicino ad esso. L'obiettivo di una SVM è la determinazione dell'iperpiano con ρ_o massimo, individuato da:

$$\mathbf{w}_o^T\mathbf{x}_o + b = 0.$$

La distanza di un generico vettore $\mathbf{x}$ in ingresso dall'iperpiano ottimo è espressa dalla seguente funzione di discriminazione:

$$d(\mathbf{x}) = \mathbf{w}_o^T\mathbf{x} + b_o.$$

Una SVM deve quindi individuare i parametri w_o e b_o dell'iperpiano ottimo (figura 2.1), dato l'insieme di addestramento $F(\mathbf{x}_i, d_i)$ $i = 1, ..., N$, tali per cui:

$$\mathbf{w}_o^T \mathbf{x}_i + b_o \geq 1, \text{ per } d_i = +1;$$
$$\mathbf{w}_o^T \mathbf{x}_i + b_o \leq -1, \text{ per } d_i = -1.$$

Le particolari coppie $(\mathbf{x}_i, d_i)$ per cui l'una o l'altra delle precedenti relazioni sono soddisfatte con il segno di uguaglianza sono chiamati support vectors (cioè vettori di supporto per la classificazione), da cui il nome Support Vector Machine, ed hanno un ruolo rilevante nell'addestramento. Infatti, i support vectors sono i punti $\mathbf{x}_i$ più vicini alla superfcie di decisione e, pertanto, più diffcili da classificare e di più elevato impatto nella determinazione dell'iperpiano ottimo.

Nel caso in cui i dati di training siano linearmente separabili, il problema di ottimizzazione si riconduce con semplici considerazioni geometriche alla seguente formulazione:

$$\min_{\mathbf{w},b} ||\mathbf{w}||$$

con i vincoli:

$$d_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \quad \forall\, i \text{ in } 1, ..., N$$

Tale problema di ottimizzazione risulta piuttosto complesso da risolvere in quanto dipendente da $||\mathbf{w}||$, ossia dalla norma di $\mathbf{w}$. Tuttavia è possibile sostituire $||\mathbf{w}||$ con $\frac{||\mathbf{w}||^2}{2}$ senza alterare la soluzione (le due funzioni presentano infatti lo stesso minimo rispetto a $\mathbf{w}$ e b), ottenendo il seguente problema di ottimizzazione qudratica:

$$\min_{\mathbf{w},b} \frac{||\mathbf{w}||^2}{2}$$

soggetto agli stessi vincoli della formulazione precedente. Esso può essere espresso in funzione dei moltiplicatori di Lagrange non-negativi α_i come:

$$\min_{\mathbf{w},b,\alpha} \frac{||\mathbf{w}||^2}{2} - \sum_{i=1}^{N} \alpha_i [d_i(\mathbf{w}^T \mathbf{x}_i + b) - 1]$$

la cui soluzione risulta essere:

$$\mathbf{w_o} = \textstyle\sum_{i=1}^{N} \alpha_i d_i \mathbf{x}_i \qquad b_o = \sum_{i=1}^{N_{SV}} (\mathbf{w_i} \mathbf{x}_i - d_i)$$

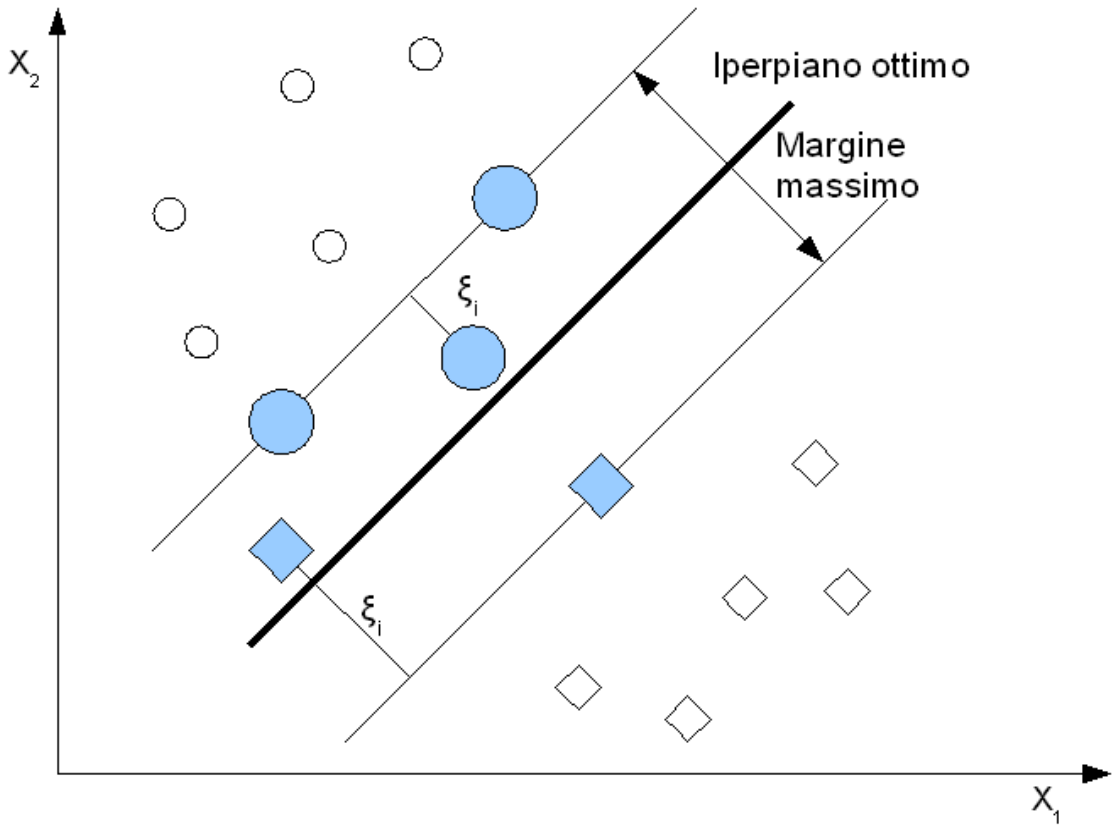

Figura 2.2: *Iperpiano ottimo per dati non linearmente separabili*

ove N_{SV} è il numero di support vectors, che possono essere facilmente individuati in quanto corrispondenti a coefficienti α_i positivi.

SVM non-lineare a due classi

In caso non esista un iperpiano in grado di separare perfettamente i campioni di training appartenenti alle due differenti classi, è possibile sfruttare una versione modificata dell'algoritmo SVM, finalizzata ad identificare l'iperpiano che minimizzi il numero di campioni erroneamente classificati, massimizzando invece la distanza rispetto ai campioni più vicini correttamente classificati (figura 2.2). Vengono pertanto introdotte delle variabili addizionali ξ_i che valutano il grado di misclassificazione del campione $\mathbf{x}_i$, ottenendo la seguente versione modificata dei vincoli:

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \quad \forall\, i \text{ in } 1, ..., N.$$

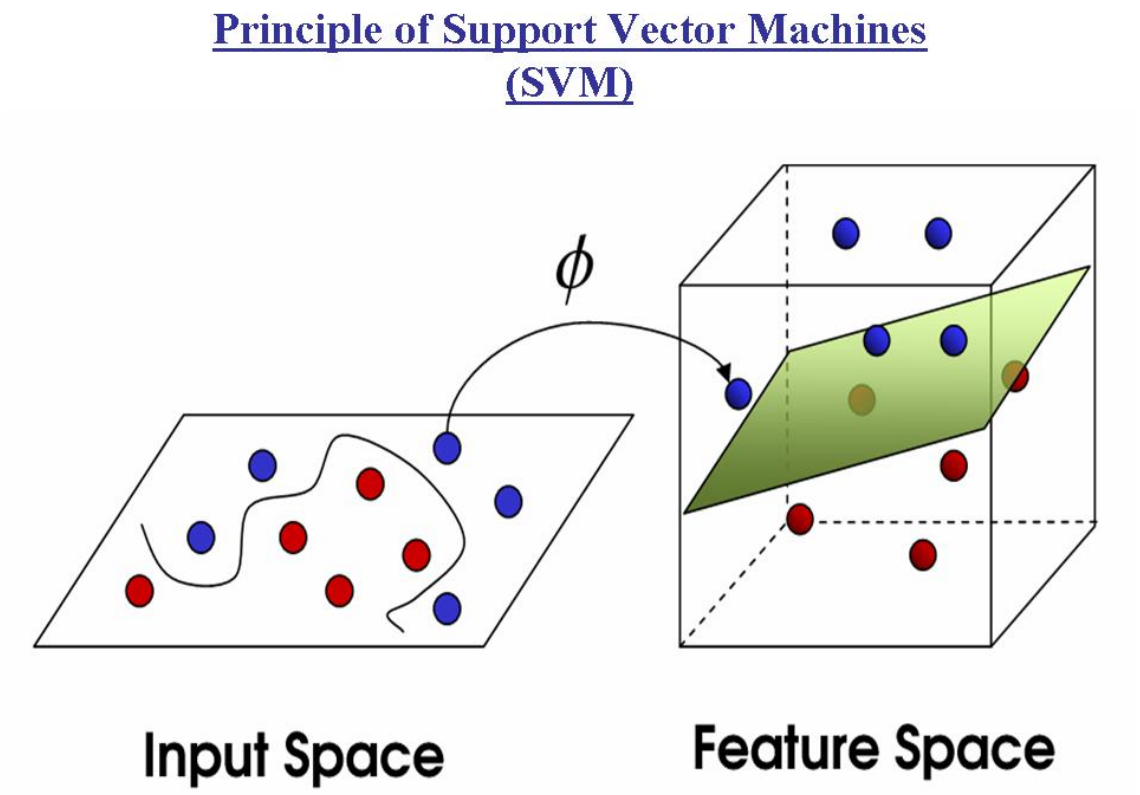

Figura 2.3: *Mappatura dei dati nello spazio degli attributi*

Coerentemente, alla funzione obiettivo viene sommata una funzione che penalizza valori non nulli di ξ_i ed il processo di ottimizzazione perverrà ad un trade-off tra ampiezza dei margini e penalità d'errore. Se la funzione di penalità è linare, si ottiene in particolare:

$$\min_{\mathbf{w},b} \frac{||\mathbf{w}||^2}{2} + C \sum_{1=1}^{N} \xi_i$$

Per realizzare un classificatore SVM non-lineare, Vapnik [23] propone anche di effettuare un mappatura dei dati dallo spazio originale ad uno spazio multidimensionale chiamato spazio degli attributi (vedi figura 2.3). L'algoritmo risultante è formalmente identico al precedente, pur di sostituire il prodotto scalare con una funzione di *kernel* non lineare, che permetta di individuare l'iperpiano discriminante ottimo nello spazio degli attributi.

SVM multiclasse

L'algoritmo SVM fin qui descritto è formulato per problemi di classificazione binaria e, dal momento che utilizza funzioni di decisione dirette, l'estensione al caso multiclasse non si presenta immediata. Vi sono due tipi di algoritmi SVM adatti a problemi multiclasse:

- SVM one-versus-all
- SVM one-versus-one

Secondo la formulazione di Vapnik, nell'SVM one-versus-all un problema con n classi è convertito in n problemi binari e per l'i-esimo problema binario, la classe i viene separata dalle restanti. La classificazione avviene secondo una strategia *winner-takes-all*: la classe viene assegnata dal classificatore binario che ottiene il più alto valore della funzione di decisione. E' dunque importante che i range di variazione delle funzioni di decisione siano calibrati correttamente, in modo da fornire risultati comparabili. Tuttavia, questo tipo di formulazione permette l'esistenza di regioni non classificabili nel caso si utilizzino funzioni di decisione discrete.

Per risolvere tale inconveniente, nell'SVM *one-versus-one*, Kreel [27] converte il problema a n classi in $n(n-1)/2$ problemi binari, riducendo in tal modo le regioni di non classificabilità. Le funzioni di decisione vengono definite per ciascuna delle possibili coppie di classi, sfruttando i dati di training corrispondenti alla coppia considerata. Ciò permette di ridurre notevolmente il numero di dati utilizzati nelle singole sessioni di training rispetto all'SVM *one-versus-all*. Per contro, il numero di funzioni di decisione è $n(n-1)/2$, rispetto alle n della tecnica precedente. La classificazione avviene per votazione: ciascun classificatore binario assegna il campione a una delle due classi e a procedura conclusa gli si attribuisce la classe più gettonata.

Una volta completata la procedura di training, si ottiene il classificatore mostrato in figura 2.4.

Per l'implementazione dell'algoritmo SVM è stato utilizzato il pacchetto `e1071` del software `R`, che fornisce un'interfaccia a `libsvm`, con possibilità di visualizzazione

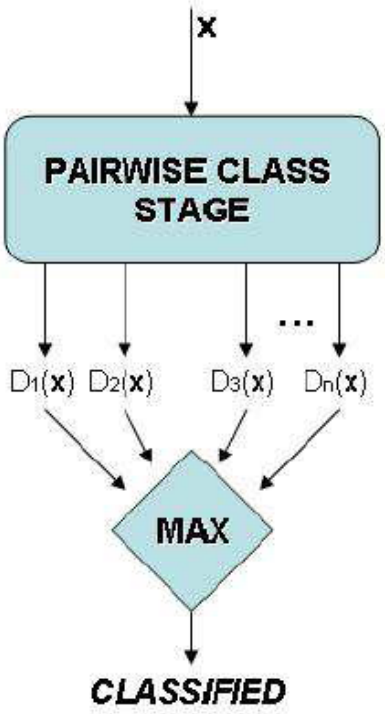

Figura 2.4: *Tecnica di classificazione*

dei risultati e di tuning dei parametri. Il kernel adottato nel caso specifico è una Radial Basis Function (RBF) data da:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\sigma||\mathbf{x_i} - \mathbf{x_j}||^2)$$

ove σ è un parametro positivo che ne controlla il raggio.

Poichè l'accuratezza di un algoritmo SVM risulta strettamente dipendente dalla scelta dei parametri del modello, è stato effettuato un preprocessing mirato all'individuazione del loro valore ottimale all'interno di dati range di valori: per evitare il rischio di overfitting, si è proceduto ad una cross-validazione che valutasse il fitting di ciascun valore attibuito ai parametri, il che ha incrementato il carico computazionale dell'algoritmo. Come compromesso tra complessità e prestazioni, per i parametri σ e C ci si è limitati a considerare 100 valori all'interno del range $[3^{-4} : 3^5]$ ed a 10 cross-validazioni.

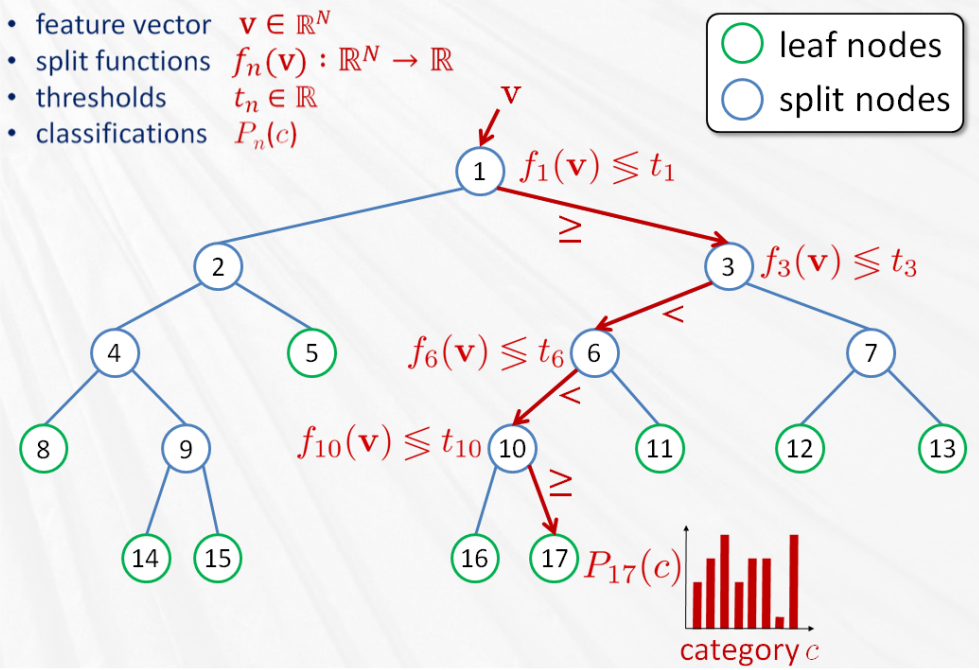

Figura 2.5: *Esempio di costruzione di un albero decisionale binario: le foglie evidenziano la probabilità di appartenenza alla classe c.*

2.2.2 Alberi decisionali e Random Forests (RF)

Gli algoritmi tradizionali per alberi decisionali creano modelli dotati di una struttura ad albero [28]: i nodi dell'albero rappresentano gli attributi, i rami indicano i possibili valori che li collegano. Una serie di nodi e rami è terminata da una foglia rappresentante una classe. Per determinare la classe di appartenenza di un'istanza è quindi necessario individuare un percorso lungo rami e nodi fino alla foglia corretta (vedi figura 2.5).

Per ridurre la sensibilità dell'algoritmo alla rumorosità dei dati e agli errori di etichettatura dei dati di training, in letteratura sono state proposte tecniche di apprendimento d'insieme che si basano su un set di classificatori deboli e ne combinano i risultati. I due metodi più comunemente usati a questo scopo sono chiamati boosting [29] e bagging [30]: il boosting prevede che si effettuino più classificazioni successive, attribuendo un peso addizionale ai punti erroneamente classificati dai

precedenti predittori, così da effettuare la predizione definitiva tramite un voto pesato; il bagging utilizza invece un insieme di alberi costruiti indipendentemente su sottoinsiemi differenti dei dati di training e la classificazione avviene sulla base di un semplice voto di maggioranza [31].

Breiman in [30] propone l'algoritmo denominato Random Forests (RF), che aggiunge un ulteriore fattore di randomizzazione al bagging: oltre a generare ogni albero a partire da un diverso dataset, modifica il modo in cui ciascun albero viene costruito, così da ottenere un insieme di alberi incorrelati e identicamente distribuiti. Il valore atteso della media di N alberi sarà pertanto uguale a quello di ciascuno di essi, mentre se il singolo albero ha varianza σ^2, la varianza di N alberi i.i.d. sarà pari a $\frac{\sigma^2}{N}$. Inoltre, mentre la procedura standard di splitting cerca lo split ottimo tra tutte le variabili per un determinato nodo, RF lo individua all'interno di un sottoinsieme di m attributi selezionati casualmente. Poichè la varianza di N alberi i.i.d. con correlazione positiva pari a ρ è [32]:

$$\rho\sigma^2 + \frac{\sigma^2(1-\rho)}{N^2}$$

la riduzione del numero di attributi causa un decremento della correlazione tra ciascuna coppia di alberi e una conseguente diminuzione della varianza.

Questa strategia ha mostrato ottime prestazioni e notevole robustezza rispetto all'overfitting. Inoltre, la procedura di training si rivela estremamente semplice, in quanto dipendente da due soli parametri (numero di alberi della foresta e numero di attributi tra i quali effettuare lo splitting a ciascun nodo) e tipicamente non molto sensibile ai loro valori.

L'algoritmo RF è di seguito riportato per completezza (cfr. algoritmo 1): dato un dataset di training T_r con n campioni $(x_1, y_1), (x_2, y_2), ...(x_n, y_n)$, ove x_i è il vettore degli attributi in uno spazio d-dimensionale R^d e $y_i \in \{-1, +1\}$ è l'etichetta della classe corrispondente, $1 \leq i \leq n$, lo scopo è individuare un classificatore con una funzione di decisione $f(x, \theta)$ tale che $y = f(x, \theta)$, ove y è l'etichetta della classe di appartenenza di x e θ è un vettore di parametri non noti della funzione. Il

metodo di bootstrap genera a partire dall'insieme originale un sottoinsieme In-Bag T_r^{IB} di n' elementi ($n' \leq n$) per campionamento uniforme con rimpiazzo di T_r e un sottoinsieme Out-Of-Bag T_r^{OB}. Tale procedura può essere ripetuta più volte, in modo da generare più sottoinsiemi T_r^{IB}.

Successivamente, per ciascun sottoinsieme In-Bag viene costruito un albero decisionale, selezionando ad ogni nodo un sottoinsieme di attributi rispetto ai quali effettuare lo splitting. Un campione di test viene quindi classificato in base al voto di maggioranza ottenuto dall'insieme degli alberi di decisione.

Algorithm 1 Algoritmo Random Forests

Draw N bootstrap samples from the original data.
for all N samples **do** {grow an unpruned classification tree}
  for each node **do**
    - randomly sample m of the predictors {rather than choosing the best split among all predictors}
    - choose the best split from among those variables {Bagging can be thought of as the special case of random forests obtained when $m = p$, the number of predictors.}
  end for
end for
Predict new data by aggregating the predictions of the N trees{i.e., majority votes for classification}

Per l'implementazione dell'algoritmo è stato utilizzato il pacchetto `Random Forest` del software `R`, che fornisce informazioni addizionali come la misura dell'importanza delle variabili di predizione attraverso il metodo della Differenza Media di Gini e della prossimità tra coppie di campioni. Nelle simulazioni il parametro m è stato fissato a $\sqrt{p}$, ove p è il numero totale degli attributi, mentre N è stato fissato a 500.

2.2.3 Classificatore di Parzen

Il metodo di Parzen è un metodo di riconoscimento statistico non parametrico con apprendimento supervisionato: dato un insieme di coppie campione-classe $(\mathbf{x}, C_h)$, esso effettua una stima per punti della densità di probabilità $\mathrm{P}(\mathbf{x})$.

Ricordando che questa può essere considerata un'approssimazione del rapporto incrementale, è possibile ricavarla da N campioni di apprendimento secondo la

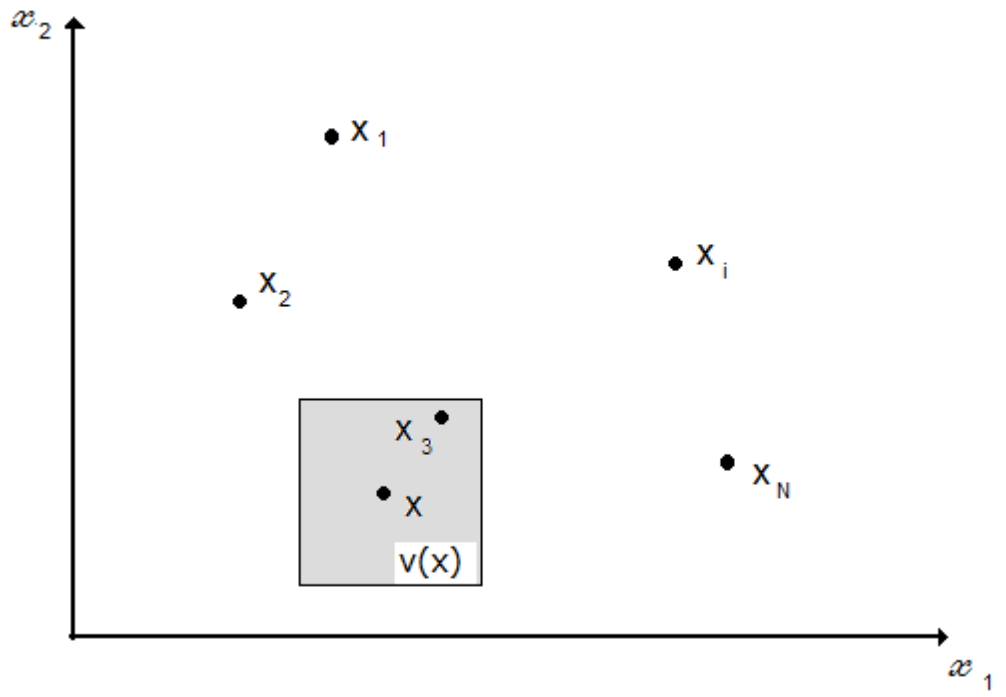

Figura 2.6: *Esempio di stima della densità di probabilità secondo il metodo di Parzen*

relazione:

$$P(\mathbf{x}) = \frac{k(\mathbf{x})}{Nv(\mathbf{x})}$$

ove $k(\mathbf{x})$ è il numero di campioni del volume $v(\mathbf{x})$ centrato in $\mathbf{x}$ (vedi figura 2.6). Supponendo $v(\mathbf{x})$ costante, il metodo può essere generalizzato definendo la funzione di *kernel* $k(\mathbf{x})$, da cui risulta:

$$P(\mathbf{x}) = \frac{\sum_{i=1}^{N} k(\mathbf{x} - \mathbf{x_i})}{N}$$

ove $\mathbf{x_i}$ sono i campioni di addestramento per cui il valore del *kernel* centrato in $\mathbf{x}$ è non nullo.

Comunemente si sceglie:

$$k(\mathbf{x} - \mathbf{x_i}) = \begin{cases} 1/v(\mathbf{x}) = 1/v = \text{cost. se } \mathbf{x_i} \in v(\mathbf{x}) \\ 0 \text{ se } \mathbf{x_i} \notin v(\mathbf{x}) \end{cases}$$

Tuttavia, a volte si preferisce un'espressione continua del *kernel* con il vincolo $\int k(\mathbf{x})d\mathbf{x} = 1$, come ad esempio una gaussiana, con il vantaggio di non dover definire il volume di appartenenza $v(\mathbf{x})$. P($\mathbf{x}$) risulta quindi essere una variabile casuale dipendente dal numero di campioni contenuti in $v(\mathbf{x})$ ed il cui valor medio è la stima cercata.

Il classificatore viene poi realizzato come segue: per ogni classe C_h $(h = 1...K)$ si hanno a disposizione N_h campioni di addestramento $\mathbf{x}_i^h$ $(i = 1...N_h)$. Per classificare il campione $\mathbf{x}$ si calcolano le stime delle densità di probabilità condizionate $P(\mathbf{x}/C_h)$:

$$P(\mathbf{x}/C_h) = \frac{\sum_{i=1}^{N_h} k_h(\mathbf{x} - \mathbf{x_i^h})}{N_h}$$

ove si è trascurato il valore (costante) $v(\mathbf{x})$, mentre il *kernel* k_h può dipendere dalla classe. La probabilità congiunta è quindi:

$$P(\mathbf{x}/C_h) = \frac{\sum_{i=1}^{N_h} k_h(\mathbf{x} - \mathbf{x_i^h})}{N_h} P(C_h).$$

Una volta calcolate le stime, la classificazione si effettua attribuendo al campione la classe h con la massima densità di probabilità. In particolare, se le probabilità a priori sono le medesime per ciascuna classe, è sufficiente la stima della probabilità condizionata, se invece le $P(C_h)$ non sono note, con questo metodo si ottiene una classificazione a massima verosimiglianza.

Per l'implementazione dell'agoritmo è stato utilizzato il software `R`, sebbene esso non fosse dotato di un package specifico relativo al classificatore di Parzen: la decisione è stata infatti dettata da motivi di coerenza con le precedenti scelte implementative. Si è quindi provveduto alla creazione di un tool dedicato sulla base di codici realizzati per il software `Matlab`, che sono stati convertiti nel linguaggio di programmazione proprio di `R`.

2.3 Attributi statistici

Oltre all'uso di attributi di traffico ricavati dall'analisi di ogni singolo flusso, sono stati considerati anche degli attributi di classificazione calcolati prendendo in

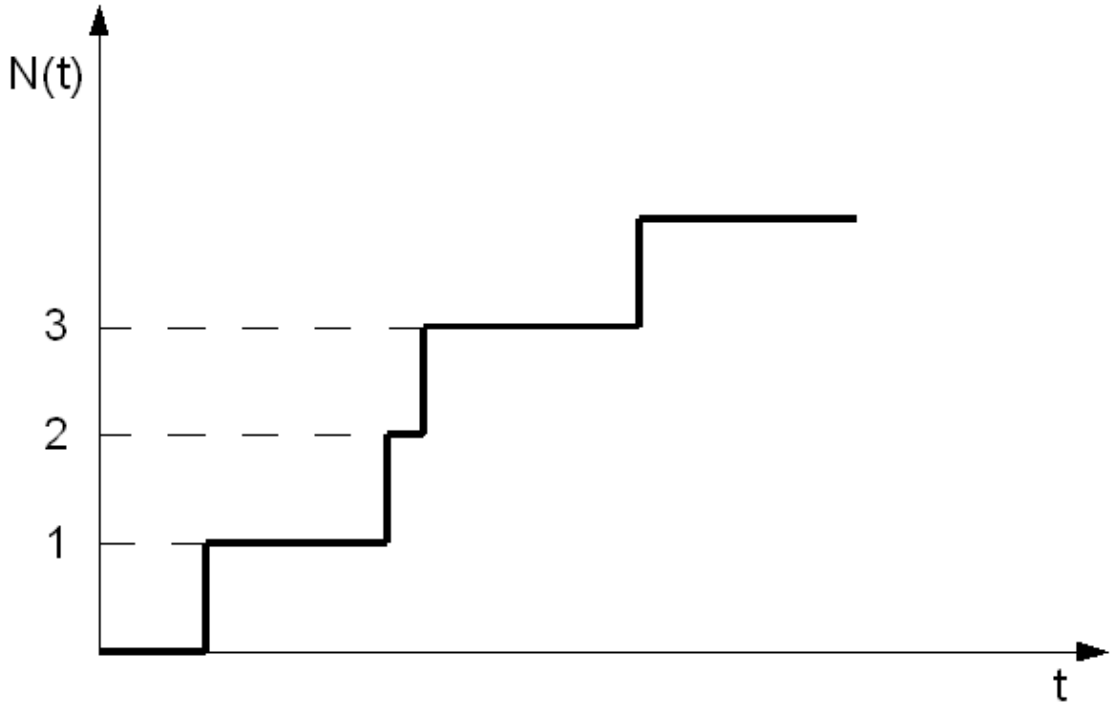

Figura 2.7: *Esempio di processo di conteggio*

considerazione un insieme di più flussi di traffico originati dallo stesso utente (attributi per-sorgente). In particolare, sono state estratte le statistiche del processo di conteggio delle connessioni che sarà introdotto nella seguente sezione.

2.3.1 Descrizione del modello

Sia $\mathcal{N}(t_1, t_2)$ una variabile casuale indicante il numero di eventi temporali che si susseguono nell'intervallo $[t_1; t_2]$, nel caso continuo chiuso a destra. Il processo $\mathcal{N}(0, t)$, mostrato in figura 2.7, si definisce processo di conteggio. Esso subisce al crescere del tempo degli incrementi unitari ogni volta che nell'intervallo si verifica un nuovo evento di arrivo. Nel caso specifico, si considerino gli insiemi dei flussi di traffico caratterizzati dallo stesso indirizzo IP sorgente (client), partizionati a loro volta in base alla porta di destinazione (classe di appartenenza). Dato un client i, un periodo di campionamento τ_0 e un intervallo di osservazione $n\tau_0$, è possibile contare per ogni τ_0 il numero di richieste di connessione da parte del client i per l'applicazione j. Si

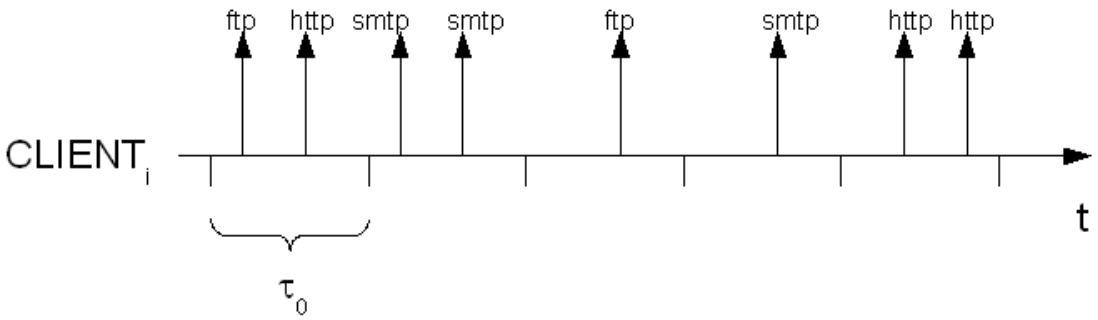

Figura 2.8: *Processo di conteggio delle connessioni*

definisce *processo di conteggio delle connessioni* la funzione $\mathcal{N}_i^j(n\tau_0)$, che indica per ogni τ_0 il numero di richieste di connessione da parte del client i per l'applicazione j nell'intervallo di tempo $[0, n\tau_0]$. In figura 2.8 sono raffigurate le richieste di connessione nel tempo da parte di un client, per il quale possono essere individuati i seguenti processi di conteggio:

$\mathcal{N}_i^{ftp}(n\tau_0) = \{1, 0, 1, 0, 0\}$

$\mathcal{N}_i^{http}(n\tau_0) = \{1, 0, 0, 0, 2\}$

$\mathcal{N}_i^{smtp}(n\tau_0) = \{0, 2, 0, 1, 0\}$

2.3.2 Definizione degli attributi statistici

per il processo $\mathcal{N}_i^j(n\tau_0)$ vengono poi calcolati i seguenti attributi statistici che forniscono, per confronto rispetto ad una distribuzione normale di media e varianza note, una descrizione più precisa dell'andamento di una curva:

- coefficiente di variazione;
- indice di dispersione;
- skewness;

Inoltre è stato calcolato tramite la tecnica della MAVAR l'attributo *alpha* (α), esponente della legge spettrale di potenza del processo delle connessioni, in funzione del

quale è stato definito l'*Indice di Variabilità* $H_v(\tau)$. Alternativamente, l'Indice di Variabilità piò essere stimato direttamente grazie alla tecnica della Varianza Aggregata. Viene di seguito fornita una breve descrizione di ciascuno dei parametri appena menzionati.

Coefficiente di variazione

Il coefficiente di variazione fornisce un'indicazione della distanza tra i valori della distribuzione statistica in esame ed un valore centrale della curva, solitamente la media. Dato il processo di conteggio $\mathcal{N} = \{n_1, n_2, \ldots, n_N\}$, si definiscono *valor medio* e *varianza* di $\mathcal{N}$ rispettivamente:

$$\mu_{\mathcal{N}} = \frac{1}{N}\sum_{i=1}^{N} n_i P(n_i)$$

$$\sigma^2_{\mathcal{N}} = \frac{1}{N}\sum_{i=1}^{N}(n_i - \mu_{\mathcal{N}})^2$$

Il coefficiente di variazione viene quindi calcolato come:

$$\mathrm{CV}_{\mathcal{N}} = \frac{\sigma_{\mathcal{N}}}{|\mu_{\mathcal{N}}|} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{n_i}{|\mu_{\mathcal{N}}|} - 1\right)^2}$$

ossia come rapporto tra la deviazione standard e la media (non nulla) di $\mathcal{N}$.

Indice di dispersione

Come il coefficiente di variazione, l'indice di dispersione è una misura normalizzata che quantifica il grado di aggregazione di un insieme di realizzazioni, rispetto ad un modello statistico di riferimento. L'IDC (Index of Dispersion for Counts) è definito come il rapporto tra varianza e media (non nulla) di $\mathcal{N}$:

$$\mathrm{IDC}_{\mathcal{N}} = \frac{\sigma^2_{\mathcal{N}}}{\mu_{\mathcal{N}}}$$

e viene di solito utilizzato per statistiche positive. Come riferimento si sceglie solitamente la distribuzione di Poisson, che ha IDC pari a 1. Quando L'IDC scende

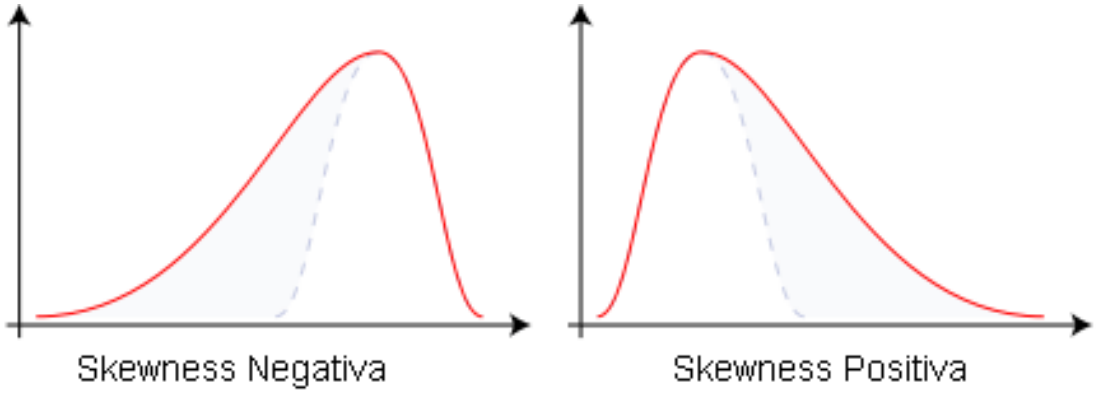

Figura 2.9: *Esempio di skewness negativa e di skewness positiva*

al di sotto del valore unitario, i dati vengono definiti "sottodispersi": ciò significa che, suddividendo l'asse delle ascisse in intervalli, vi sono più intervalli in cui cade un numero di campioni vicino alla media, rispetto ad una distribuzione di Poisson; viceversa, se l'IDC è maggiore di 1, i dati sono "sovradispersi", il che implica la presenza di un alto numero di intervalli con conteggi molto distanti dal valor medio.

Skewness

La skewness è un parametro che misura il grado di simmetria della distribuzione di probabilità di una variabile casuale. Data la variabile casuale $\mathcal{N}$, la skewness viene calcolata come:

$$\text{skewness}(\mathcal{N}) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{n_i - \mu_{\mathcal{N}}}{\sigma_{\mathcal{N}}}\right)^3$$

dove N è il numero dei campioni.

La skewness di una distribuzione normale è nulla, mentre assume valori diversi da zero in condizioni di asimmetria. In particolare (figura 2.9):

- **valori negativi** indicano una distribuzione con coda sinistra più lunga della destra (curva *left-skewed*). Ciò significa che i campioni sono concentrati sulla destra della curva;

- **valori positivi** indicano una distribuzione con coda destra più lunga della sinistra (curva *right-skewed*). Ciò significa che i campioni sono concentrati sulla sinistra della curva.

Modified Allan Variance

La MAVAR è stata originariamente concepita ai fini della caratterizzazione della stabilità in frequenza per oscillatori di precisione nel dominio del tempo [33] allo scopo di discriminare diverse tipologie di rumore caratterizzati da uno spettro power-law del tipo $f^{-\alpha}$. Recentemente, la MAVAR è stata proposta come strumento di analisi del traffico Internet [34] e ne è stata provata l'alta precisione nella stima dell'esponente α, insieme ad una buona robustezza alla non stazionarietà dei dati. Essa è stata applicata in [34] e [35] su tracce di traffico Internet, identificando rumore frazionario nei risultati sperimentali, e su traffico telefonico GSM, provando che esso si configura come un processo Poissoniano.

Risulta ormai noto [36] che l'alta variabilità del traffico Internet è dovuta alla proprietà di *Long Range Dependence* (LRD) dei processi. In generale un processo $Y = \{Y_n, n = 1, 2, ...\}$ è definito LRD se $\sum_{k=1}^{\infty} r(k) = \infty$, ove $r(k)$ è la correlazione tra campioni separati da k unità di tempo. Se invece $\sum_{k=1}^{\infty} r(k) \leq \infty$, il traffico è definito *Short Range Dependent* (SRD).

Molti modelli di traffico LRD sono basati su processi auto-similari, termine corrente con il quale si indicano processi *asintoticamente auto-similari del secondo ordine* o *monofrattali* [37], definiti come segue: si assuma che Y abbia una funzione di autocorrelazione del tipo $r(k) = k^{-\beta}L(k)$ per $k \to \infty$, con $0 \leq \beta \leq 1$ ed L tale per cui $\lim_{k\to\infty} \frac{L(kx)}{L(k)} = 1 \ \forall x \geq 0$. Per ogni $m = 1, 2, 3, ...$ sia $Y^{(m)} = \{Y_n^{(m)}, n = 1, 2, 3...\}$ una serie temporale aggregata ottenuta mediando la serie originale Y su blocchi non sovrapposti di durata m e rimpiazzando ogni blocco con la media dei suoi campioni. Quindi, per ogni $m = 1, 2, 3, ...$, risulta:

$$Y^{(m)} = \frac{Y_{nm-m+1} + ... + Y_{nm}}{m}, \quad n \geq 1.$$

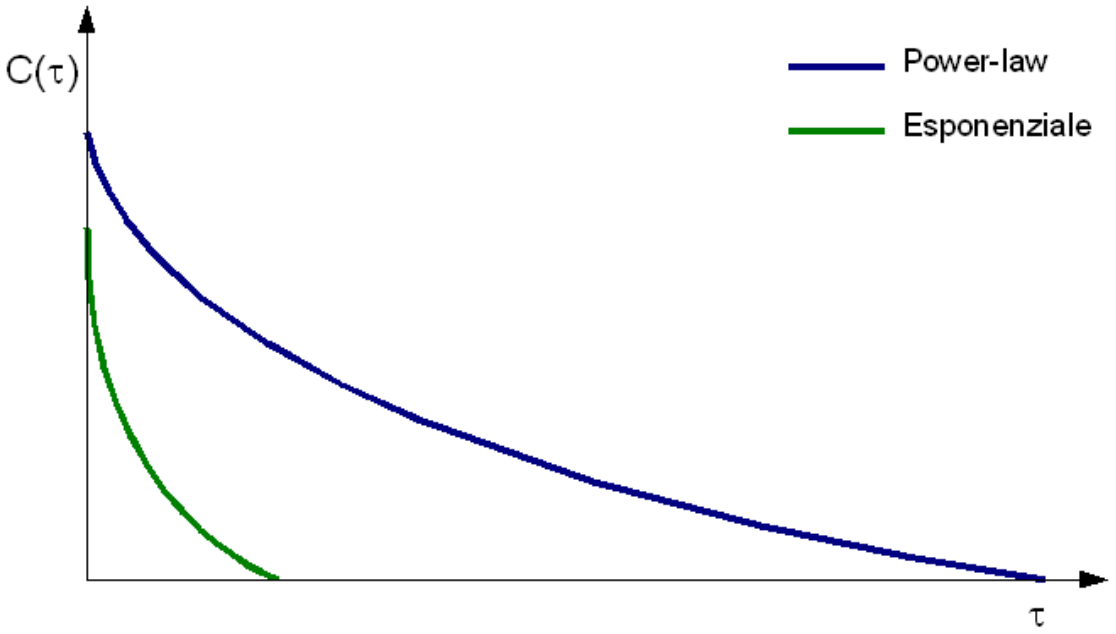

Figura 2.10: *Confronto tra l'autocorrelazione di tipo Power-law dei processi LRD e l'autocorrelazione di tipo esponenziale*

Questo processo tempodiscreto aggregato è anch'esso stazionario, con funzione di autocorrelazione $r^{(m)}(k)$. Allora, Y è definito asintoticamente auto-similare del secondo ordine con parametro di auto-similarietà $H = 1 - \frac{\beta}{2}$ se, per ogni k sufficientemente grande, $r^{(m)}(k) \to r(k)$ per $m \to \infty$, ossia se la funzione di correlazione del processo aggregato $Y_{(m)}$ diventa indistinguibile da quella del processo originale. Per definizione, l'autosimilarietà asintotica del secondo ordine implica Long Range Dependence.

IL parametro H è detto *parametro di Hurst* e misura il grado di auto-similarietà di un processo, ossia la velocità di decadimento della coda della funzione di autocorrelazione (figura 2.10). Esso assume valori compresi nell'intervallo $\left[\frac{1}{2}; 1\right]$ e può essere stimato a partire da α tramite la relazione:

$$\widehat{H} = \frac{\alpha + 1}{2}$$

In particolare, se $0,5 < H < 1$ il processo è LRD, mentre se $H = 0.5$ il processo è SRD.

Si richiamano ora brevemente alcuni concetti descritti in [38] riguardanti la tecnica della MAVAR e il calcolo dell'α della legge power-law. Data una sequenza infinita di campioni $\{x_k\}$ di un segnale in ingresso $x(t)$ campionato con un periodo di campionamento τ_0, la MAVAR è definita come:

$$Mod\sigma_y^2(\tau) = \frac{\left\langle \left[\frac{1}{n} \sum_{j=1}^{n} (x_{j+2n} - 2x_{j+n} + x_j) \right]^2 \right\rangle}{2n^2\tau_0^2} \tag{2.1}$$

dove $\tau = n\tau_0$ è l'intervallo di osservazione e l'operatore $\langle . \rangle$ indica la media su un intervallo infinito nel tempo.

In pratica, dato un insieme finito di N campioni in un intervallo di misura $T = (N-1)\tau_0$, la MAVAR può essere calcolata utilizzando lo stimatore standard ITU-T [39] come:

$$Mod\sigma_y^2(n\tau_0) = \frac{\sum_{j=1}^{N-3n+1} \left[\sum_{i=j}^{n+j-1} (x_{i+2n} - 2x_{i+n} + x_i) \right]^2}{2n^4\tau_0^2 (N - 3n + 1)}$$

con $n = 1, 2, \ldots, \lfloor N/3 \rfloor$.

Si consideri ora un processo casuale $x(t)$ con una densità spettrale di potenza modellizzata come:

$$S_x(f) = \begin{cases} \sum_{i=1}^{P} h_{\alpha_i} f^{\alpha_i} & \text{se } 0 < f \leq f_h, \\ 0 & \text{se } f > f_h \end{cases} \tag{2.2}$$

dove P è il numero di rumori presenti nel modello, α_i e h_{α_i} sono i parametri del modello e f_h la frequenza di cut-off. Per questi processi, comunemente chiamati processi *power-law*, la media infinita nel tempo in (2.1) converge per $\alpha_i > -5$. Quindi, considerando separatamente ogni termine della somma in (2.2) e ponendo $P = 1$ e $-1 < \alpha_i \leq 0$, la MAVAR obbedisce a una semplice legge di potenza dell'intervallo di osservazione τ (idealmente asintotico per $n \to \infty$, con $n\tau_0 = \tau$, in concreto per $n > 4$):

$$Mod_y^2(\tau) \simeq A_\mu \tau^\mu \tag{2.3}$$

dove $\mu = -3 - \alpha$ e A_μ è una costante. Se $P > 1$, l'equazione (2.3) può essere generalizzata come una sommatoria di termini del tipo power-law $A_{\mu_i}\tau^{\mu_i}$, dove $\alpha_i = -3 - \mu_i$. In [34] viene mostrata l'affidabilità di queste stime, per cui si è stabilito di avvalersi di questo strumento per analizzare la legge di potenza dei vari processi di conteggio delle connessioni ed estrarre il valore di α da utilizzare poi come attributo statistico per le tracce di traffico.

Aggregate Variance Method

Il metodo della varianza Aggregata [36] è uno dei più frequantemente utilizzati in letteratura per la stima del parametro di Hurst a partire da una traccia di traffico. Esso prevede, partendo dal processo $Y^{(m)}$ definito nella sezione precedente, di calcolare $\mathrm{Var}[Y^{(m)}]$ per valori di m e quidistanti su una scala logaritmica e di rappresentare $\mathrm{Var}[Y^{(m)}]$ in funzione di m in coordinate logaritmiche. L'andamento dei punti nel piano viene poi approssimato con una retta tramite il metodo dei minimi quadrati e di essa viene calcolata la pendenza, che risulta legata alla stima del parametro di Hurst dalla relazione:

$$\widehat{H} = 1 - \frac{|\text{pendenza}|}{2}$$

Il metodo può quindi essere riassunto dai seguenti punti:

1. Sia m un numero intero. Per diversi valori di m e un numero sufficiente (detto k_m) di sottosequenze di lunghezza m, calcolare la media campionaria $\overline{Y}_1^{(m)}, \overline{Y}_2^{(m)}, ..., \overline{Y}_{k_m}^{(m)}$ e la media totale:

$$\overline{Y} = \frac{1}{k_m}\sum_{j=1}^{k_m}\overline{Y}_j^{(m)}$$

2. per ogni m, calcolare la varianza delle medie campionarie $\overline{Y}_j^{(m)}$ $(j = 1, 2, ..., k_m)$:

$$s^2(m) = \frac{1}{k_m - 1}\sum_{j=1}^{k_m}(\overline{Y}_j^{(m)} - \overline{Y}^{(m)})^2$$

3. rappresentare $log(s^2(m))$ in funzione di $log(m)$.

E' inoltre interessante notare che nel caso di un processo auto-similare, il metodo della Varianza Aggregata è matematicamente equivalente al calcolo dell'IDC: infatti rappresentando $IDC(m\tau)$ in funzione di $log(m)$, asintoticamente si ottiene una retta con pendenza $2H-1$. Se Y è un processo LRD con $0<\beta<1$, esso presenterà un andamento asintotico della curva IDC in coordinate log-log secondo una retta con pendenza $1-\beta$, il che implica necessariamente $0<$ pendenza <1. Se l'andamento della curva IDC presenta un asintoto orizzontale per $\tau<\infty$, il processo è invece SRD.

Indice di variabilità

L'Indice di Variabilità $H_v(\tau)$ è stato introdotto da Lazarou et al. in [18] come misura del grado di burstiness del traffico di rete su diverse scale temporali. Esso è legato all'IDC dalla relazione:

$$H_v(\tau)=\frac{\frac{d(\log(IDC(\tau)))}{d(\log(\tau))}+1}{2}=\frac{1}{2}\left\{1+\tau\left(\frac{\frac{d(IDC(\tau))}{d\tau}}{IDC(\tau)}\right)\right\} \tag{2.4}$$

ove $\frac{d(\log(IDC(\tau)))}{d(\log(\tau))}$ è la pendenza della curva dell'IDC (supposta continua e differenziabile in $[0,\infty)$) rispetto ad una scala temporale τ, rappresentata in coordinate log-log. Si noti che nel caso di un processo asintoticamente auto-similare del secondo ordine $H_v(\tau)\to H\in(\frac{1}{2},1)$ per $\tau\to\infty$. Se invece il processo è esattamente auto-similare, allora $H_v(\tau)=H\in(\frac{1}{2},1)$ per ogni $\tau>0$. Ciò significa che, se $\log(IDC(\tau))$ è lineare rispetto a $(\log(\tau))$, allora $H_v(\tau)$ si riduce ad H e l'Indice di Variabilità può essere considerato come il paramentro di Hurst valutato per ogni scala temporale.

2.4 Caratterizzazione della sorgente di traffico RPH2

Per ciascuna applicazione considerata ai fini dell'attività sperimentale che sarà presentata nei prossimi capitoli, è stata definita una variabile aleatoria $X(t)$ indicante il tempo di interarrivo tra due pacchetti consecutivi generati da una dato server sulla

porta sorgente associata all'applicazione, con un modello *iperesponenziale* di ordine $K(K = 1, 2, 3...)$, il quale è risultato particolarmente adatto per caratterizzare traffico fortemente correlato [18]. La distribuzione della v.a. è quindi la somma pesata di K distribuzioni esponenziali:

$$F_K(x) = P[X \leq x] = \sum_{i=1}^{K} w_i(1 - e^{-\alpha_i x})$$

ove $w_i > 0$ sono dei pesi che soddisfano la condizione $\sum_{i=1}^{k} w_i = 1$ e $\alpha_i > 0$ sono i tassi delle distribuzioni esponenziali [40]. E' stato mostrato in [41] che se $w_i = w^i$ e $\alpha_i = \frac{\mu}{\eta^i}$ per $0 < w < 1$, $\eta > 0$ e $\mu > 0$, allora la coda della distribuzione esponenziale si allunga al crescere di K. Nel caso specifico è stato scelto $K = 2$, ottenendo il processo indicato come **RPH2**. Ponendo $a = \alpha_1$ e $b = \alpha_2$, la densità di probabilità dei tempi di interarrivo risulta essere:

$$f_2(x) = w_1 a e^{-ax} + w_2 b e^{-bx}. \tag{2.5}$$

Il tasso medio degli arrivi è dunque pari a:

$$\lambda = \frac{ab}{aw_2 + bw_1}$$

mentre il quadrato del coefficiente di variazione dei tempi di interarrivo è:

$$C^2(X) = 2\left[\frac{a^2 w_2 + b^2 w_1}{(aw_2 + bw_1)^2}\right] - 1$$

Si noti che se $a = b$, allora $a = b = \lambda$ e $C^2(X) = 1$ per ogni scelta di w_1 e w_2, quindi ci si riconduce ad un processo di Poisson. Il tempo medio di interarrivo è ovviamente legato al tasso degli arrivi dalla relazione:

$$\mu_X = \frac{1}{\lambda} = \frac{aw_2 + bw_1}{ab}$$

e la varianza può essere espressa in funzione di μ_x come:

$$\sigma_X^2 = \frac{(a^2 b^2 \mu_x^2 - 2a^2 b\mu_x + 2a^2)w_2 + (a^2 b^2 \mu_x^2 - 2ab^2 \mu_x + 2b^2)w_2}{a^2 b^2}.$$

Sia ora $N(t)$ il numero di arrivi avvenuti nell'intevallo $(0, t]$. Per ogni intervallo di lunghezza fissata $\tau > 0$, è possibile costruire il processo di conteggio $Y = Y_n(\tau), \tau > 0, n = 1, 2, ...,$ il quale definisce il numero di eventi verificatisi durante l'n-simo intervallo di durata τ, secondo la relazione:

$$Y_n(\tau) = N[n\tau] - N[(n-1)\tau]$$

In questo caso, $Y = Y_n(\tau)$ denota il numero di pacchetti osservati su un generico link, a partire da un istante di tempo arbitrario, durante l'n-simo intervallo temporale di durata τ (la variabile τ, rappresentante la durata di un elemento della sequenza Y, verrà d'ora in avanti considerata la scala temporale della traccia di traffico).

Il numero medio di arrivi verificatisi nell'intevallo $(0, t]$ è dato da:

$$E[N(t)] = \frac{t}{E[X]} = \lambda t$$

ove $E[N(t)]$ è il tempo medio di interarrivo, mentre il parametro $IDC(t)$ definito in sezione 2.3.2 risulta essere:

$$IDC(t) = \frac{Var[N(t)]}{E[N(t)]} = \frac{Var[N(t)]}{\lambda t}$$

Si noti che, dal momento che il processo è stazionario, $IDC(t)$ ha lo stesso valore per ogni intervallo di lunghezza t, pertanto t può essere visto come la scala temporale τ del processo Y.

Nel caso della sorgente di traffico iperesponenziale appena definita, in [18] è stata dimostrata la validità delle seguenti espressioni:

$$E[N(\tau)] = \frac{ab\tau}{aw_2 + bw_1}$$

$$Var[N(\tau)] = \frac{2\lambda[(aw_1 + bw_2)^2 - (a^2w_1 + b^2w_2)]}{(aw_2 + bw_1)^2} \times (1 - e^{-[aw_2+bw_1]\tau}) + \lambda C^2(X)\tau$$

$$IDC(\tau) = \frac{2[(aw_1 + bw_2)^2 - (a^2w_1 + b^2w_2)]}{(aw_2 + bw_1)^2} \times \frac{(1 - e^{-[aw_2+bw_1]\tau})}{\tau} + C^2(X)$$

e poichè l'indice di variabilità può essere calcolato in funzione dell'IDC (come mostrato nella sezione 2.3.2), si ottiene la relazione:

$$H_v(\tau) = \frac{1}{2}\left\{1 + \frac{2e^{-k\tau}(1 - e^{k\tau} + b\tau w_1 + a\tau w_2)[a^2(-1+w_1)w_1 + 2abw_1w_2 + b^2(-1+w_2)w_2]}{k^3\tau[-1 + \frac{2(b^2w_1+a^2w_2)}{k^2} - \frac{2(-1+e^{-k\tau})[a^2(-1+w_1)w_1+2abw_1w_2+b^2(-1+w_2)w_2]}{k^3\tau}]}\right\}$$

ove $k = bw_1 + aw_2$.

2.5 Metriche di valutazione

Il dato alla base della valutazione delle prestazioni di un classificatore è la matrice di confusione, definita come:

$$\mathcal{A} = \begin{bmatrix} a_{11} & a_{12} & ... & a_{1N} \\ a_{21} & a_{22} & ... & a_{2N} \\ ... & ... & ... & ... \\ a_{N1} & a_{N2} & ... & a_{NN} \end{bmatrix}$$

dove N è il numero di classi e l'elemento a_{ij} rappresenta il numero di flussi appartenenti alla classe i e classificati come classe j. Sommando gli elementi della riga i-esima si ottiene il numero totale di istanze del dataset di classe i, mentre la somma degli elementi della colonna j-esima formisce il numero di istanze classificate, correttamente o meno, nella classe j. Gli elementi della diagonale principale rappresentano quindi i flussi correttamente classificati, mentre gli altri elementi della matrice rappresentano i flussi erroneamente classificati.

Tramite la matrice di confusione si possono ricavare le seguenti metriche sintetiche:

- **Error-Rate**: rapporto tra il numero totale di istanze erroneamente classificate e il numero di istanze presenti nel dataset.

  $$\text{Error-rate } = 1 - \frac{\sum_{i=1}^{N} a_{ii}}{\sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij}}$$

  Questo valore fornisce una prima indicazione generale sulle prestazioni dell'algoritmo di classificazione, mentre per un'analisi più dettagliata si calcolano per ciascuna classe i seguenti indici:

- **True Positive Rate** per la classe i (TPR_i): rapporto tra il numero di istanze correttamente classificate come classe i e il numero totale di istanze appartenenti alla classe i.

  $$TPR_i = \frac{a_{ii}}{\sum_{j=1}^{N} a_{ij}}$$

- **False Positive Rate** per la classe i (FPR_i): rapporto tra il numero di istanze erroneamente classificate come classe i e il numero totale di istanze del dataset effettivamente non appartenenti alla classe i.

$$FPR_i = \frac{\sum_{i=1, i \neq j}^{N} a_{ij}}{\sum_{i=1, i \neq j} \sum_{j=1}^{N} a_{ij}}$$

- **False Discovery Rate** per la classe i (FDR_i): rapporto tra il numero di istanze erroneamente classificate come classe i e il numero totale di istanze calssificate erroneamente o meno nella classe i.

$$FPR_i = \frac{\sum_{i=1, i \neq j}^{N} a_{ij}}{\sum_{i=1}^{N} a_{ij}}$$

Capitolo 3

Ambiente di sperimentazione

L'attività sperimentale prevede l'utilizzo e l'elaborazione di file di dati contenenti traffico IP catturato dalla rete in tempo reale tramite il software `NetMate` il quale, seguendo le regole definite tramite NetAI [42], crea un file in formato `arff` con gli attributi calcolati su ogni singolo flusso di traffico IP, dopodichè si procede ad un'ulteriore elaborazione nell'ambiente R [43] tramite il quale si effettua un'analisi statistica dei dati al fine di estrarre nuovi attributi e si passa alla fase finale di classificazione.

3.0.1 Le tracce di traffico IP

Le tracce di traffico IP su cui sono state effettuate tutte le elaborazioni ed analisi presentate in questo capitolo rappresentano raccolte complete di header di livello IP e TCP dei pacchetti appartenenti ai flussi di traffico registrati e sono disponibili in vari formati liberamente reperibili. Le tracce usate sono state raccolte durante il progetto PMA (Passive Measurement and Analysis) [44], nato nel luglio del 2006 e sostenuto del gruppo di misurazione ed analisi della rete NLANR (The National Laboratory for Applied Network Research) [45] operante presso l'Università della California di San Diego. Lo scopo di tale progetto è quello di effettuare uno studio approfondito del comportamento della rete Internet per migliorarne la fruibilità da parte degli utenti. Il gruppo mette gratuitamente a disposizione, in aggiunta ai rilevamenti di traffico IP effettuati giornalmente, un certo numero di tracce speciali

registrate in punti diversi della rete in date antecedenti l'inizio del progetto PMA, usate da molti enti per studi analoghi. Le tracce sono state tutte anonimizzate prima della loro pubblicazione per garantire riservatezza agli utenti monitorati. Il processo di anonimizzazione consiste nella sostituzione degli indirizzi IP reali di sorgente e destinazione con indirizzi IP privati, riservati di solito alle reti LAN, prestando attenzione a mantenere la corretta differenziazione anche dopo il processo di anonimizzazione. In particolare, tutti gli indirizzi sono stati sostituiti con indirizzi IP privati di classe A del tipo 10.x.x.x.

La raccolta del traffico è stata effettuata tramite schede DAG installate in vari punti della rete, che hanno il compito di registrare il traffico IP in transito sui link monitorati. Queste schede sono composte da hardware e software progettati per la misurazione e la cattura ad alta velocità e senza alcuna perdita degli header dei pacchetti, indipendentemente dal tipo di interfaccia del link, dalla dimensione dei pacchetti e dal carico della rete. La tecnologia è stata sviluppata tra il 1995 e il 2001 presso l'Università di Waikato in Nuova Zelanda, con una successiva commercializzazione delle schede hardware, dei driver e dei tool di modifica delle tracce catturate, tra cui le librerie open-source `libtrace` per l'elaborazione delle tracce. I formati `tcpdump` e `pcap` sono i formati standard usati per creare i file che serviranno per la classificazione ed hanno solitamente una dimensione considerevole, pertanto si preferisce rendere disponibile per il download le tracce compresse e in formato `erf` o `Legacy Ethernet`. Nel campo dell'amministrazione di rete, il termine `pcap`, acronimo di Packet CAPture, indica una serie di interfacce di programmazione dell'applicazione (API) per la cattura del traffico dati in rete. Le librerie `libpcap`, che permettono la conversione delle tracce dai formati di cattura al formato `pcap`, furono originariamente create dagli sviluppatori del programma TCPdump, il più classico degli sniffer/monitor di rete, nel Network Research Group presso il Laboratorio Lawrence Berkeley.

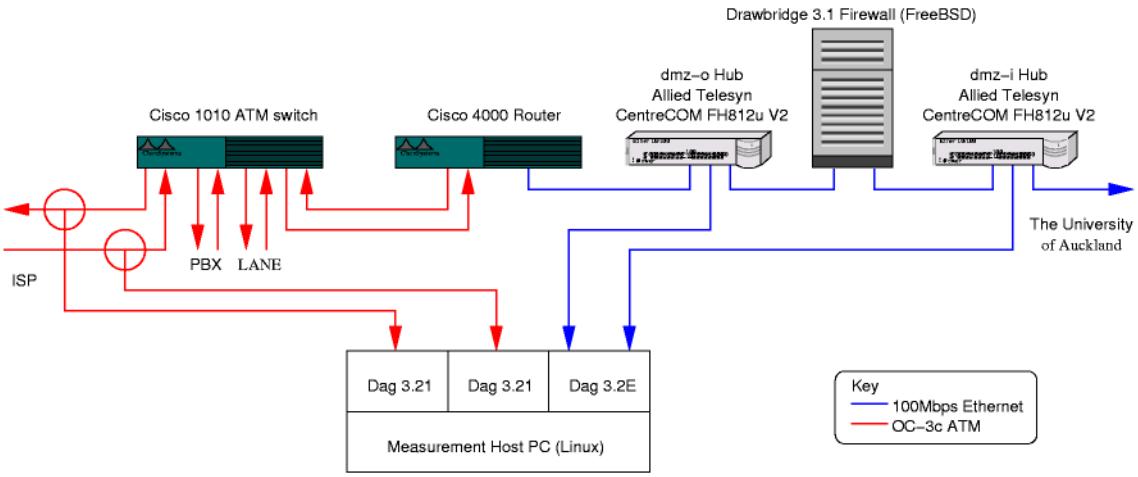

Figura 3.1: *Struttura di rete Auckland [46]*

Le tracce Auckland VI

Le tracce Auckland VI [47] sono una raccolta di header IP registrata senza interruzioni dall'8 al 13 giugno 2001 dal gruppo di ricerca WAND simultaneamente in tre punti differenti dell'infrastruttura Internet dell'Università di Auckland in Nuova Zelanda, mostrata in figura 3.1.

Un router collega uno switch ATM a un hub ethernet da 100Mbps, che è a sua volta collegato via firewall ad un secondo hub ad esso identico. Una coppia di schede DAG3 ATM intercettano il traffico sul link tra lo switch e l'ISP, mentre una terza scheda ethernet DAG3 è connessa tramite due sue porte ai due hub dmz, fornendo una copertura totale della rete e del traffico IP al suo interno. I dati catturati sono stati poi divisi in quattro tracce giornaliere da 6 ore ciascuna. Per questo lavoro di tesi sono state usate le tracce registrate tra il router ATM e l'ISP e, a causa della loro divisione basata sulla direzione del traffico, sono state riunite tramite il comando `tracemerge` della libreria `libtrace` in un unico file contenente i flussi appartenenti ad entrambe le direzioni. In particolare, sono state usate la traccia dell'11 giugno (Auck-I) e del 12 giugno (Auck-II).

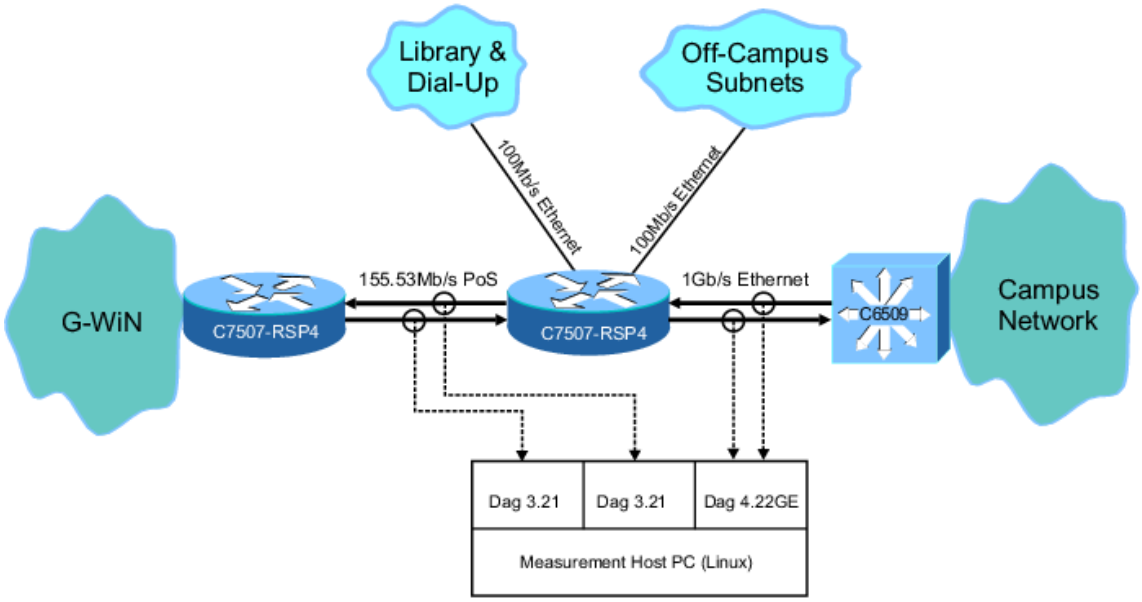

Figura 3.2: *Struttura di rete Lipsia [49]*

Le tracce NewZeland II

Le NewZeland II [48] sono tracce della durata di 5 giorni raccolte dal gruppo di ricerca WAND nel periodo compreso tra il 5 e il 10 luglio 2000 e contenenti il traffico IP tra il dipartimento ITS (Information e Technology Services Division) presso l'Università di Waikato in Nuova Zelanda e diversi ISP. La rete NZIX (New Zeland Internet Exchange), al momento della registrazione delle tracce, era composta da due router Cisco 2926, collegati ai maggiori ISP neozelandesi. La scheda di monitoraggio DAG era collegata a uno SPAN (Switched Port Analyzer) attraverso una porta ethernet a 100Mbps. Anche questi dati sono divisi in quattro tracce giornaliere da 6 ore ognuna ma, a differenza delle tracce Auckland VI, non vi sono due direzioni di flusso. Le tracce usate in questa tesi sono quelle registrate il 7 luglio del 2000 e chiamate Nzix-II.

Le tracce Leipzig II

Le tracce Leipzig II [50] sono state registrate il 21 febbraio 2003 simultaneamente ad entrambi i lati del router d'accesso ad Internet dell'Università di Lipsia in Germania,

la cui struttura di rete viene mostrata in figura 3.2. Due schede DAG3 sono usate per registrare il traffico nei i due canali Sonet OC3 che collegano il router centrale e la rete di ricerca tedesca G-WiN. Nella parte interna della rete una scheda DAG4 intercetta il collegamento ethernet da 1Gbps tra la rete del campus e il router centrale. In questa tesi sono state utilizzate le tracce prelevate dalla scheda DAG4, in particolare la Leip-II delle 12:14.

Trace	date	duration
Auckland-VI	11th June 2001	from 12:00 to 18:00
	12th June 2001	from 12:00 to 18:00
Nzix-II	7th July 2000	from 06:00 to 12:00
	7th July 2000	from 12:00 to 18:00
Leipzig-II	21st February 2003	from 12:14 to 15:00
	21st February 2003	from 15:00 to 21:00

Tabella 3.1: *Data di raccolta e durata delle tracce utilizzate durante le fasi di addestramento e di validazione.*

In tabella 3.1 vengono riassunte le caratteristiche delle tracce di traffico usate per effettuare le analisi descritte in questo capitolo.

3.0.2 NetMate

`NetMate` (Network Measurement and Accounting System) [51] è uno strumento di analisi del traffico nelle reti, tipicamente utilizzato o per la raccolta di traffico in tempo reale da un'interfaccia di rete (ad esempio eth0 su linux) oppure per la lettura dei dati di una traccia di traffico raccolti in un file (in formato `tcpdump` o `pcap`). `NetMate` necessita di un insieme di regole che spieghino come elaborare l'input e può fornire il risultato delle sue elaborazioni su un file di output.

Per quanto concerne le presenti analisi sperimentali, `NetMate` riceve in ingresso le tracce di traffico in formato `pcap`, esegue un'aggregazione del traffico in flussi secondo la quintupla base $\langle sourceIP, destinationIP, sourcePort, destinationPort, protocol \rangle$ ed estrae gli attributi del flusso in un file in formato `arff`.

Nel caso di connessione TCP, `NetMate` definisce un *flusso* come l'insieme di pacchetti appartenenti alla stessa connessione (aventi quindi tutti la stessa quintupla base). Se per 600 secondi non arriva nessun pacchetto, allora la connessione si considera terminata. Nel caso in cui il protocollo di trasporto sia UDP, viene considerato come flusso l'insieme di pacchetti con la stessa quintupla base e che arrivano entro un timeout di 600 secondi.

Tramite il tool NetAI (Network Traffic based Application Identification) [42] si è in grado di interfacciarsi a `NetMate`, fornendogli l'insieme di regole per le elaborazioni dei file `pcap` in ingresso e stabilendo gli attributi che si andranno ad estrarre. Modificando il file `netai_flowstat.c`, presente nella cartella sorgente di NetAI, è possibile modificare gli attributi esistenti, crearne di nuovi od estrarre solo quelli a cui si è interessati, mentre modificando il file `netAI_rules-arff.xml`, è possibile variare le regole che disciplinano l'aggregazione dei flussi e la creazione degli attributi.

Il formato `arff`

Il formato `arff` (Attribute-Relation File Format)è un formato di file di testo usato per memorizzare i dati in un database. Il file ha questa semplice struttura, organizzata in due parti:

- Intestazione (definizione degli attributi):

  @relation - nome del dataset -

  @attribute - nome attributo - tipologia attributo -

- Dati:

  @data - valore primo attributo - valore secondo atributo...

In questa seconda parte ogni istanza viene descritta dalla lista dei valori per ciascun attributo e ogni valore corrisponde all'attributo che si trova nella corrispondente posizione nell'intestazione. Gli attributi possono essere di quattro tipologie:

```
@relation Nzix II 20000706

@attribute dstport {ftp,telnet,smtp,domain,http}
@attribute fpktl1 numeric
@attribute fpktl2 numeric
@attribute fpktl3 numeric

@data
ftp,44,40,59
ftp,44,40,59
ftp,48,40,56
smtp,60,60,52
smtp,44,40,40
smtp,44,40,57
```

Figura 3.3: *Esempio di formato* `arff`

- numerici;
- nominali;
- stringhe;
- date.

nel caso presente sono stati usati attributi di tipo numerico per definire i valori, di tipo nominale per indicare la classe di appartenenza dei flussi e di tipo stringa per definire gli indirizzi IP sorgente e destinazione. In figura 3.3 si mostra un esempio della struttura `arff`, ove il primo attributo indica la classe.

3.0.3 L'ambiente R

Il sistema `R` [43] è considerato un'implementazione di S, linguaggio di programmazione statistico sviluppato nel 1980 nei Bell Laboratories da R. Becker, J. Chambers e A. Wilks. L'ambiente `R` è tuttora sostenuto e sviluppato da un gruppo di lavoro denominato Comprehensive R Archive Network (CRAN), che rende disponibile gratuitamente il software e la documentazione sotto i vincoli della General Public

Licence (GPL). `R` non è semplicemente un software statistico, ma un ambiente interattivo integrato, ovvero un insieme di funzioni, librerie ed oggetti che possono essere usati per la gestione e l'analisi dei dati e la realizzazione di grafici. Esso si compone di un efficiente elaboratore di dati, di un insieme di operatori per calcoli su array e matrici, di un ampio insieme di strumenti per analisi intermedie dei dati con la possibilità di creare grafici e di un semplice ed efficiente linguaggio interpretato object-oriented (il linguaggio S) che include istruzioni per la verifica di condizioni sulle variabili (`if`), cicli `for` e funzioni definite dall'utente che possono anche essere ricorsive. Un'importante differenza tra `R` e gli altri principali ambienti statistici, quali ad esempio SAS e SPSS, è rappresentata dal modo di procedere nell'analisi dei dati. In particolare, mentre SAS o SPSS forniscono soltanto una serie di risultati riguardanti un'analisi statistica dei dati, in `R` l'analisi è normalmente eseguita per passi, con risultati intermedi immagazzinati in oggetti su cui l'utente può operare attraverso l'uso di operatori aritmetici, logici e di comparazione e di funzioni. Nell'ambiente `R` sono state implementate tecniche statistiche classiche e moderne, alcune delle quali costruite nel pacchetto *base* che rappresenta, insieme a circa altri 25 pacchetti, il cuore di `R` e contiene inoltre funzioni basilari necessarie per leggere ed elaborare i dati ed alcune funzioni grafiche.

Capitolo 4

Prestazioni del classificatore ibrido N1-N2+

4.1 Tecnica di classificazione

In questa sezione verrà fornita una descrizione della tipologia di classificatore adottato e dei criteri di classificazione scelti.

4.1.1 Caratterizzazione statistica del processo degli arrivi delle connessioni

Per ciascun flusso, sono state misurate le lunghezze dei primi n pacchetti nella direzione client-server (che verranno utilizzate come attributi per-flow per la successiva classificazione) e il timestamp del primo pacchetto, considerato come indicatore dell'istante di richiesta di connessione. A partire dai timestamp, è possibile ricostruire la sequenza tempodiscreta $x_i^p(k)$, che conta le richieste di connessione da parte del client i-esimo associate alla p-esima porta durante l'intervallo di tempo k-esimo (la durata di un intervallo è stata considerata pari a 1 s). Ogni volta che si verifica una nuova richiesta di connessione, la sequenza $x_i^p(k)$ viene aggiornata e vengono calcolati i parametri statistici definiti in tabella 4.1 e descritti nel capitolo 2.

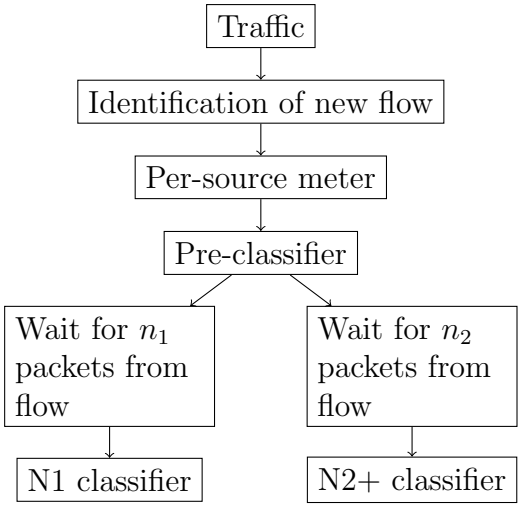

Figura 4.1: *Schema a blocchi del classificatore ibrido*

Metrica	Definizione
Indice di Dispersione	Rapporto tra varianza e media di $x_i^p(k)$
Skewness	Terzo momento standardizzato di $x_i^p(k)$
Esponente power-law	Esponente della Legge di Potenza di $x_i^p(k)$

Tabella 4.1: *Attributi per-sorgente.*

4.1.2 Il classificatore

L'introduzione di una tipologia di classificatore ibrida [52] mira a sfruttare i parametri statistici precedentemente definiti, minimizzando il numero di pacchetti necessari e conseguentemente il ritardo di classificazione. Lo schema a blocchi che ne riassume il funzionamento è mostrato in figura 4.1. Non appena si verifica l'arrivo del pacchetto iniziale di una nuova connessione, il primo stadio (che comprende per-source meter e preclassificatore) estrae l'indirizzo IP sorgente e la porta di destinazione del pacchetto, aggiorna gli attributi per-source ed effettua una decisione preliminare: se la sorgente di traffico in esame ha generato meno di ξ richieste di connessione, l'accuratezza di stima degli attributi per-source viene giudicata insufficiente. Pertanto, il flusso viene classificato non appena si giunge ad osservare n_1 pacchetti nella direzione client-server e gli attributi di classificazione saranno costituiti dalle

lunghezze $l_1, l_2, ..., l_{n_1}$, di tali pacchetti. Questa tipologia di classificatore è stata indicata in figura con N1. Se il flusso si conclude prima di aver generato n_1 pacchetti, la classificazione *post-mortem* viene effettuata attribuendo lunghezza -1 ai pacchetti mancanti.

Se invece la sorgente del flusso ha generato almeno ξ richieste di connessione, i parametri statistici per-source sono considerati sufficientemente affidabili per essere utilizzati durante la classificazione. Il flusso viene classificato non appena si giunge ad osservare n_2 pacchetti nella direzione client-server e gli attributi di classificazione saranno costituiti dalle lunghezze $l_1, l_2, ..., l_{n_2}$, di tali pacchetti e dai parametri statistici IDC, Skewness e α calcolati all'istante della classificazione preliminare del flusso. In seguito si farà riferimento a questa seconda tipologia di classificazione con N2+. Ancora una volta, se il flusso si conclude prima di aver generato n_2 pacchetti, la classificazione *post-mortem* viene effettuata attribuendo lunghezza -1 ai pacchetti mancanti. Il training del classificatore ibrido viene effettuato addestrando separatamente i classificatori N1 e N2+. In particolare, sono stati estratti 1000 flussi per classe (nel caso in cui il numero di flussi di una data applicazione non raggiungesse tale quantità, sono stati considerati tutti i flussi disponibili) e, per ciascun flusso, è stato calcolato il numero di richieste di connessione precedentemente effettuate dalla medesima sorgente IP. Se esso risulta inferiore alla soglia ξ, il flusso viene utilizzato per addestrare il classificatore N1, altrimenti vengono calcolati i relativi parametri per-source e il flusso concorre al training del classificatore N2+. Pe entrambi i classificatori sono stati sfruttati gli algoritmi Support Vector Machines e Random Forests.

Classificatori della tipologia di N1 sono ampiamente usati in letteratura, mentre il classificatore N2+ costituisce un elemento di innovazione: l'utilizzo esclusivo di quest'ultimo porterebbe tuttavia a prestazioni modeste, giacchè i parametri statistici per-source risulterebbero poco affidabili per sorgenti molto giovani. D'altro canto, il classificatore ibrido può sfruttare gli attributi statistici soltanto quando essi forniscono una certa garanzia di affidabilità, ripiegando sugli attributi standard

quando la confidenza sulle informazioni per-source è bassa. Assumendo $n_2 < n_1$, il classificatore ibrido effettua le decisioni osservando un numero di pacchetti mediamente più basso rispetto al solo classificatore N1, introducendo dunque un ritardo di classificazione inferiore e mantenendo un minor numero di flussi in uno stato di indecisione. In particolare, quando ξ è basso, il numero di pacchetti richiesti è prossimo a n_2, ma l'errore di classificazione si presuppone maggiore, poichè verranno utilizzati valori poco accurati. Viceversa, al crescere di ξ, il numero di pacchetti richiesti si avvicina a n_1, ma l'errore di classificazione sarà meno elevato.

4.2 Prestazioni della tecnica proposta

Allo scopo di valutare le prestazioni del classificatore N2+ appena descritto, verrà confrontato l'errore di classificazione da esso fornito per $n_1 = 5$ e $n_2 = 3$ con quello del classificatore N1 rispettivamente nei due casi $n_1 = 3$ e $n_1 = 5$. In questa sezione verrà mostrato che, con un'opportuna scelta del parametro ξ, il classificatore ibrido supera le prestazioni del solo classificatore N1 nel caso $n_1 = 3$ e risulta comparabile con il classificatore N1 nel caso $n_1 = 5$, presentando tuttavia un minor ritardo di classificazione.

4.2.1 Dati di validazione

Data una traccia di traffico, sono stati usati i tools `NetMate Meter` [51] e `NetAI, (Network Traffic based Application Identification)` [42] per ragruppare i pacchetti in flussi e calcolare gli attributi per-flow.

Per effettuare l'addestramento del classificatore è stato necessario identificare anche il numero di porta di destinazione per ciascun flusso, in modo da utilizzarlo come etichetta di riferimento durante la fase di validazione. Naturalmente, questo procedimento è subottimo, giacchè non è sempre possibile assumere l'utilizzo di porte note. Tuttavia nel caso si utilizzino tracce pubbliche e rese anonime tramite eliminazione del payload, questo approccio si rivela essere l'unico possibile. Di con-

Tracce Auckland, Leipzig, NZIX	
Applicazione	Porta/Protocollo
FTP	21/tcp
Telnet	23/tcp
SMTP	25/tcp
DNS	53/tcp
DNS	53/udp
HTTP	80/tcp
AOL	5190/tcp
Half-Life	27015/udp

Tabella 4.2: *Numeri di porta e protocolli associati a ciascuna applicazione.*

seguenza, il numero di porta è sempre stato considerato come identificatore veritiero dell'applicazione generatrice del flusso.

I dati raccolti sono poi stati elaborati tramite il software `R` [43] per effettuare il calcolo degli attributi per-source, il training del classificatore e la validazione.

Per poter effettuare un confronto con i risultati ottenuti da Zander et al.[10] si è scelto di realizzare la classificazione è rispetto ad 8 differenti applicazioni elencate in tabella 4.2.1. A differenza di [10], si è però deciso di escludere Napster dalle valutazioni, poichè i flussi appartenenti a tale applicazione non erano presenti nelle tracce considerate in numero sufficiente da permettere il calcolo di parametri statistici affidabili. E' stato inoltre stabilito di suddividere il DNS in due classi distinte, giacchè il comportamento dell'applicazione risulta fortemente influenzato dal protocollo di trasporto (TCP o UDP).

In entrambe le fasi di training e test, sono stati selezionati in maniera casuale 1000 flussi per ciascuna applicazione (nel caso in cui nelle tracce originali il numero di flussi di una data applicazione non raggiungesse tale quantità, sono stati considerati tutti i flussi disponibili). Per ciascuna coppia di tracce originali sono state così generate 10 coppie di sottotracce, ciascuna delle quali contenente 1000 campioni per ogni classe considerata. Per ogni coppia di sottotracce sono state effettuate due distinte validazioni, utilizzando la prima sottotraccia per l'addestramento e la seconda per il test del classificatore. Infine i risultati sono stati mediati. In tutti

gli esperimenti l'intervallo di confidenza è risultato trascurabile ed è stato quindi omesso.

4.2.2 Valutazione delle prestazioni del classificatore ibrido

Il primo set di validazioni è stato efettuato sulle tracce *Auckland*. In figura 4.2 viene mostrata la percentuale di errore in funzione della soglia ξ. Con questi dati entrambi i classificatori RF e SVM N1 con $n_1 = 3$ ottengono scarse prestazioni, con un tasso di errore attorno al 47%. Il classificatore N1 non utilizza gli attributi per-source, quindi i risultati sono indipendenti dalla soglia ξ e sono indicati in figura 4.2 con una linea orizzontale. Per contro, ponendo $n_1 = 5$, i classificatori RF e SVM riducono la percentuale di errore al 12,5%.

E' necessario tuttavia associare le prestazioni ottenute ai ritardi introdotti tra l'inizio del flusso e gli istanti in cui è reso disponibile il risultato della classificazione. In figura 4.3 sono rappresentati i tempi medi necessari per ottenere il numero di pacchetti richiesti dall'algoritmo di classificazione: per $n_1 = 5$ il tempo medio di attesa è pari a 222 ms, mentre per $n_1 = 3$ esso giunge ad abbassarsi in media a 190 ms.

Il classificatore ibrido cerca di attuare un trade-off tra tasso d'errore e ritardo di classificazione: in figura 4.2 è rappresentato l'errore di classificazione in funzione della soglia: quando $\xi = 0$, la maggioranza dei flussi viene classificata considerando $n_2 = 3$ pacchetti e gli attributi per-source. Bisogna infatti tenere in considerazione il fatto che, indipendentemente dal valore assunto dalla soglia, per alcuni flussi non possono essere calcolati i relativi attributi per-source (ad esempio nel caso in cui la sorgente non sia rimasta attiva per un periodo di tempo sufficiente per poter effettuare la stima dell'esponente α della legge di potenza). Al crescere della soglia, sempre meno flussi sono classificati con l'ausilio degli attributi statistici per-sorgente e di essi vengono dunque considerati soltanto i primi $n_1 = 5$ pacchetti. D'altro canto, i parametri per-source risultano più affidabili, a vantaggio delle prestazioni di classificazione. La figura 4.2 mostra che la percentuale d'errore scende al 14.9% e al

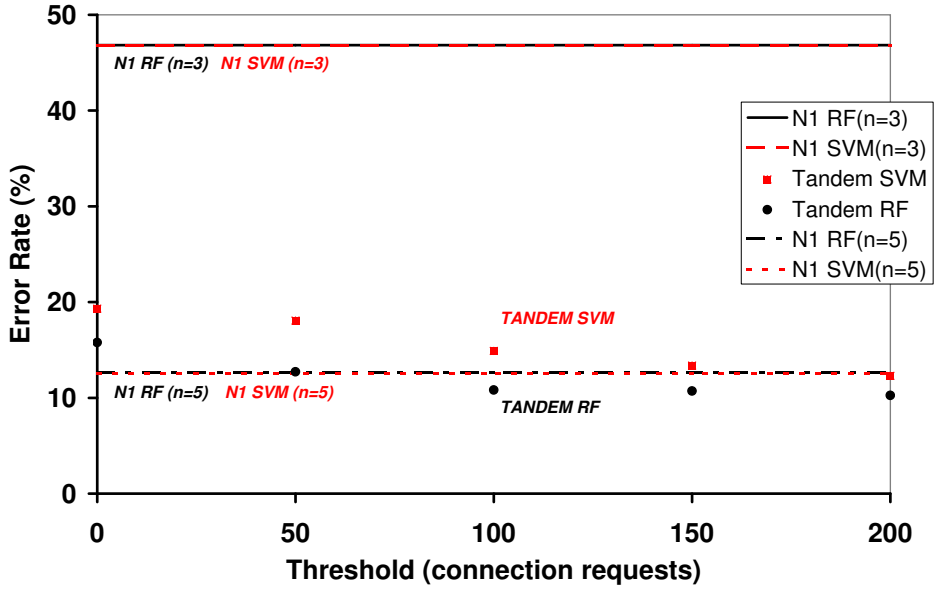

Figura 4.2: *Prestazioni dei classificatori SVM e RF per le tracce* Auckland.

10.8% per gli algoritmi SVM e RF rispettivamente, con una soglia di 100 richieste di connessione.

Tuttavia, dal momento che all'aumentare della soglia cresce il numero di flussi classificati in base ai primi n_2 pacchetti, il ritardo medio di classificazione s'innalza conseguentemente. In particolare, per $\xi = 100$ esso si attesta sui 195 ms (come mostrato in figura 4.3), ossia soltanto di 5 ms maggiore rispetto a quello ottenuto con il classificatore N1 per $n_1 = 3$ - il quale però ha prestazioni molto inferiori - e di 27 ms inferiore al ritardo registrato con il classificatore N1 per $n_1 = 5$. Inoltre, il classisficatore ibrido SVM mostra un errore di classificazione maggiore solo del 2% rispetto al classificatore N1 ($n_1 = 5$), quando $\xi > 100$, mentre il corrispettivo RF raggiunge prestazione addirittura migliore rispetto all'approccio del classificatore N1 ($n_1 = 5$).

Le tabelle 4.3 e 4.4 mostrano rispettivamente i valori di TPR e FDR per ciascuna applicazione e per diversi valori della soglia ξ. Le buone prestazioni del classificatore ibrido sono motivate dall'alto TPR e dal basso FDR ottenuti dalla maggior parte delle applicazioni. Tali valori evidenziano inoltre che il traffico generato dalle appli-

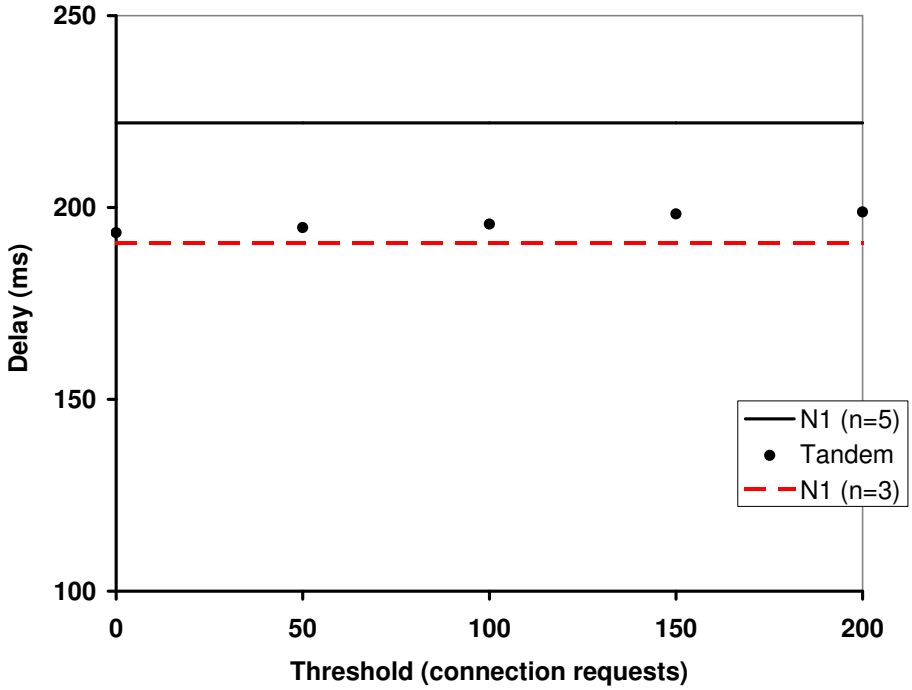

Figura 4.3: *Ritardo di classificazione per le tracce* Auckland.

cazioni Telnet, AOL e SMTP sia di più difficile identificazione, mentre al contrario il traffico Half-Life è facilmente riconosciuto dal classificatore ibrido.

Per motivi di sintesi i risultati ottenuti sulla tracce Nzix e Leipzig, che presentano un trend analogo a quello riscontrato nelle tracce Auckland, sono riassunti nelle tabelle 4.5 e 4.6.

4.3 Conclusioni

In questo capitolo sono stati esposti i risultati sperimentali riguardanti la classificazione del traffico Internet esaminando il flusso di pacchetti nella direzione client-server. Ci si è indirizzati verso una classificazione di tipo online, la quale richiede un compromesso tra accuratezza di classificazione e numero di pacchetti richiesti dal classificatore.

Sono state confrontate le prestazioni di due tecniche di classificazioni note: Support Vector Machines e Random Forests, effettuando test di validazione su 3 tracce

Classificazione	Applicazione	Soglia, ξ				
		0	50	100	150	200
RF	FTP	0.72	0.95	0.98	0.98	0.97
	Telnet	0.17	0.27	0.27	0.27	0.31
	SMTP	0.78	0.75	0.73	0.71	0.68
	Half-Life	1.00	1.00	1.00	1.00	1.00
	AOL	0.77	0.88	0.55	0.55	0.55
	DNS:udp	0.99	1.00	0.99	0.99	1.00
	DNS:tcp	0.76	0.72	0.9	0.92	0.96
	HTTP	0.92	0.94	0.94	0.94	0.95
SVM	FTP	0.68	0.91	0.94	0.94	0.94
	Telnet	0.27	0.41	0.41	0.41	0.41
	SMTP	0.7	0.66	0.66	0.66	0.65
	Half-Life	0.98	0.99	1.00	1.00	1.00
	AOL	0.67	0.83	0.83	0.83	0.82
	DNS:udp	0.55	0.54	0.91	0.94	0.94
	DNS:tcp	0.75	0.9	0.95	0.94	0.96
	HTTP	0.74	0.93	0.94	0.93	0.94

Tabella 4.3: *TPR dei classificatori ibridi RF e SVM sulle tracce Auckland.*

contenenti traffico generato da differenti applicazioni e concludendo che RF ottiene migliori prestazioni.

Inoltre è stato introdotto un nuove metodo di classificazione, basandosi sull'osservazione che il processo di generazione delle connessioni da una data sorgente di traffico è fortemente influenzato dall'applicazione che genera tali richieste. In particolare si evince dai dati sperimentali che la densità spettrale di potenza di tali processi evidenzia spesso un andamento di tipo power-law. Perciò si è scelto di utilizzare l'esponente della legge di potenza delle sorgenti di traffico come attributo addizionale per la classificazione dei flussi: il calcolo di tale attributo non introduce alcun ritardo aggiuntivo, giacchè si basa sui timestamps dei pacchetti iniziali. Sfruttando questo nuovo attributo è stato possibile ridurre il tasso di errore di classificazione per tutte le tracce considerate.

In aggiunta è stato valutato un metodo di classificazione ibrido che utilizza i primi 5 pacchetti del flusso nel caso di sorgenti a bassa attività, limitandosi invece ai primi 3 pacchetti per sorgenti ad alta attività, così da ridurre i ritardi di classificazione.

Classification	Application	Threshold, ξ				
		0	50	100	150	200
RF	FTP	0.29	0.27	0.3	0.31	0.32
	Telnet	0.35	0.06	0.06	0.06	0.05
	SMTP	0.22	0.16	0.13	0.1	0.06
	Half-Life	0.00	0.00	0.00	0.00	0.00
	AOL	0.51	0.32	0.14	0.14	0.05
	DNS:udp	0.02	0.01	0.01	0.00	0.01
	DNS:tcp	0.09	0.04	0.11	0.11	0.11
	HTTP	0.26	0.13	0.03	0.03	0.03
SVM	FTP	0.42	0.23	0.24	0.25	0.27
	Telnet	0.43	0.08	0.08	0.08	0.08
	SMTP	0.26	0.11	0.1	0.1	0.07
	Half-Life	0.00	0.00	0.00	0.00	0.00
	AOL	0.47	0.18	0.18	0.19	0.19
	DNS:udp	0.21	0.03	0.02	0.03	0.03
	DNS:tcp	0.11	0.07	0.05	0.04	0.03
	HTTP	0.44	0.41	0.27	0.22	0.2

Tabella 4.4: *FDR dei classificatori ibridi RF e SVM sulle tracce Auckland.*

Classificatore	Performance		Ritardo
	SVM	RF	
N1 (n=3)	12.29%	12.89%	229 ms
Tandem	6.22%	2.59%	344 ms
N1 (n=5)	3.14%	2.25%	529 ms

Tabella 4.5: *Prestazioni dei classificatori SVM e RF per le tracce* NZIX *per* $\xi = 100$.

Classificatore	Performance		Ritardo
	SVM	RF	
N1 (n=3)	15.89%	19.35%	143 ms
Tandem	9.13%	7.14%	202 ms
N1 (n=5)	5.76%	5.05%	346 ms

Tabella 4.6: *Prestazioni dei classificatori SVM e RF per le tracce* leipzig *per* $\xi = 100$.

Capitolo 5

Classificatore binario basato sull'Indice di Variabilità

In questo capitolo verrà definito un classificatore binario che utilizza come attributo di classificazione l'Indice di Variabilità descritto in [18], finalizzato al riconoscimento dell'applicazione generante i flussi di traffico in uscita da un host-server su una data porta. Le prestazioni del classificatore verranno indagate da un punto di vista analitico, calcolando l'errore teorico minimo di classificazione, e si fornirà una stima empirica dell'errore di classificazione atteso. Si procederà poi ad una valutazione delle prestazioni su tracce sintetiche. In particolare, sarà evidenziata l'influenza sulle prestazioni del classificatore della scelta della scala temporale per il calcolo dell'Indice di Variabilità.

5.1 Definizione del classificatore binario

5.1.1 Stima dei parametri del modello

Per effettuare la valutazione delle prestazioni del classificatore binario sono state considerate due tipologie distinte di sorgenti per l'applicazione HTTP: le sorgenti di traffico sono state modellizzate secondo la formula 2.5 e la scelta dei tassi a e b e dei pesi w_1 e w_2 è stata effettuata con riferimento alla letteratura. In particolare, per quanto riguarda il traffico HTTP, la normativa ETSI [53] prevede un modello di traffico a burst che definisce il tempo di interarrivo tra i pacchetti appartenenti allo

μ_B	0.0104 s
μ_I	60 s
μ_P	25 pacchetti

Tabella 5.1: *Parametri del modello di traffico web browsing.*

stesso burst come una variabile aleatoria con distribuzione geometrica e media μ_B e il tempo di idle intercorrente tra due burst come una variabile aleatoria geometrica di media μ_I. Il numero di pacchetti facenti parte di un singolo burst viene anch'esso caratterizzato con una distribuzione geometrica di media μ_P. Pertanto, identificando con a la frequenza di arrivo dei pacchetti all'interno di un burst, con b la frequenza delle fasi di idle e con w_1 la probabilità che la sorgente sia attiva (ossia che stia trasmettendo pacchetti) in un dato istante temporale, risulterà essere:

$$a = \frac{1}{\mu_B}$$

$$b = \frac{1}{\mu_I}$$

$$w_1 = \frac{\mu_P}{\mu_P + 1}$$

In tabella 5.1 sono riportati i valori di μ_B e μ_P proposti in [53] per un flusso WWW surfing UDD a 384 kbit/s, mentre per il parametro μ_I ci si è riferiti a quanto proposto da Bianco et al. in [54] per la generazione di tracce sintetiche di traffico web.

Le due tipologie di traffico HTTP considerate (che nel seguito verranno indicate come classe **A** e classe **B**) sono caratterizzate dalla stessa velocità media e di picco (in termini di pacchetti per secondo), in modo da non essere distinguibili utilizzando le sole statistiche del processo dei tempi di interarrivo. Per ottenere la stessa velocità di picco, deve essere necessariamente $a_A = a_B = a$. Si è invece scelto di differenziare il tempo medio di idle, ponendo $b_A = 0.017s^{-1}$ e $b_B = 0.04s^{-1}$.

Ricordando che la frequenza media degli arrivi per un processo **RPH2** è data da:

$$\lambda = \frac{ab}{aw_2 + bw_1}$$

	classe A	classe B
a	96 pacchetti/s	96 pacchetti/s
b	0.017 s^{-1}	0.04 s^{-1}
w_1	0.96	0.9

Tabella 5.2: *Parametri delle classi di traffico A e B.*

e considerando w_{1A} fissato, w_{1B} può essere calcolato risolvendo la seguente equazione lineare:

$$\frac{ab_A}{a(1-w_{1A})+b_A w_{1A}} = \frac{ab_B}{a(1-w_{1B})+b_B w_{1B}}$$

che fornisce la soluzione:

$$w_{1B} = \frac{ab_B(1-w_{1A})+b_A b_B w_{1A} - ab_A}{b_A(b_B-a)}$$

I valori numerici dei parametri a, b e w_1 per le due classi considerate sono riassunti in tabella 5.2.

Sulla base di questi, è stato valutato l'Indice di Variabilità per valori di τ compresi tra $10^{-3}s$ e $1s$ ed equispaziati su scala logaritmica: scale temporali maggiori non possono essere accuratamente stimate con intervalli di osservazione dell'ordine di $1h$, mentre scale temporali inferiori sono difficilmente misurabili in pratica. Il suo andamento per un'istanza rappresentativa di ciascuna classe è mostrato in figura 5.1.

5.1.2 Prestazioni dello stimatore dell'Indice di Variabilità

La classificazione deve avvenire utilizzando una stima dell'Indice di Variabilità, ottenuta a partire dalle tracce contententi il mix di traffico che si desidera identificare. Il valore stimato $\widehat{H_v(\tau)}$ sarà quindi una variabile aleatoria, avente una distribuzione normale di cui è necessario calcolare media e varianza.

Teorema 1 *Se il processo dei tempi di interarrivo è di tipo **RPH2**, la stima dell'indice di Variabilità $\widehat{H_v(\tau)}$ è caratterizzata dalle seguenti espressioni di media e*

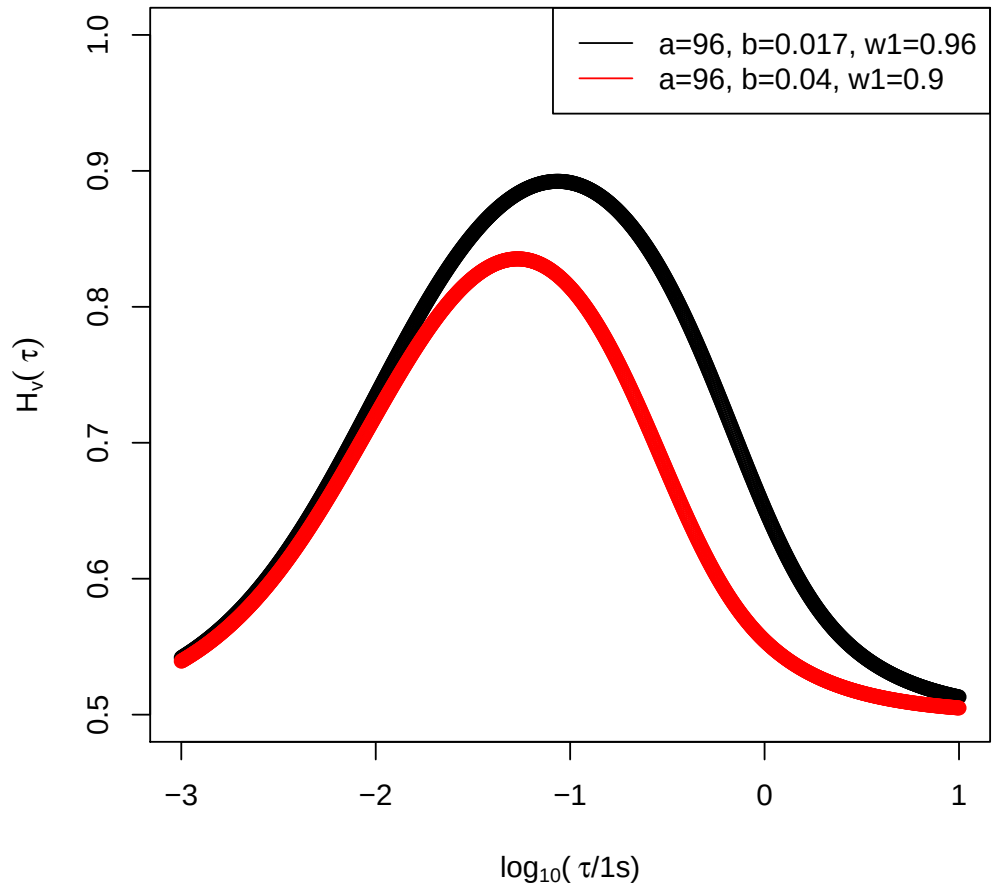

Figura 5.1: *Andamento dell'indice di variabilità al variare di τ.*

varianza:

$$E[\widehat{H_v(\tau)}] = \frac{1}{2}\left\{1 - \frac{1}{2\tau}\left(\frac{e^{-s\tau}(2ab(e^{s\tau}-2)w_1w_2 + a^2(2w_1 - 2w_1^2 + e^{s\tau})(w_2-2)w_2)}{absT_{tot}} + \frac{b^2(e^{s\tau}(w_1-2)w_1 - 2(w_2-1)w_2))\tau}{absT_{tot}} + \frac{2T_{tot}\tau^2}{(T_{tot}-\tau)^2} + \frac{2(1-e^{-s\tau})(r^2 - a^2w_1 - b^2w_2) + s^3(\frac{2(b^2w_1+a^2w_2)}{s^2} - 1)\tau}{abs^2T_{tot}} + \frac{4(a^2(w_1-1)w_1 + 2abw_1w_2 + b^2(w_2-1)w_2)\tau(1 - e^{s\tau} + bw_1\tau + aw_2\tau)}{A+B}\right)\right\} \tag{5.1}$$

$$Var[\widehat{H_v(\tau)}] = -\frac{1}{4T_{tot}}\left\{\frac{e^{-s\tau}(2ab(e^{s\tau}-2)w_1w_2 + a^2(2w_1 - 2w_1^2 + e^{s\tau}(w_2-2)w_2)}{abs} + \frac{b^2(e^{s\tau}(w_1-2)w_1 - 2(w_2-1)w_2))\tau}{abs} - \frac{2T_{tot}^2\tau^2}{(T_{tot}-\tau)^2} + \frac{2(1-e^{-s\tau})(r^2 - a^2w_1 - b^2w_2) + s^3(\frac{2(b^2w_1+a^2w_2)-1}{s^2})\tau}{abs^2}\right\} \tag{5.2}$$

ove :

- $r = aw_1 + bw_2$,
- $s = bw_1 + aw_2$,
- $A = a^3e^{s\tau}(w_2-2)w_2^2\tau + b^2[2(e^{s\tau}-1)w_2 - 2(e^{s\tau}-1)w_2^2 + be^{s\tau}(w_1-2)w_1^2\tau]$,
- $B = abw_1w_2\{4 + e^{s\tau}[b(3w_1-2)\tau - 4]\} - a^2w_1\{2 - 2w_1 + e^{s\tau}[2w_1 - 2 + b(2 - 3w_2)w_2\tau]\}$.

Dimostrazione

Ricordando che:

$$\widehat{H_v(\tau)} = \frac{\frac{d(\log(IDC(\tau)))}{d(\log(\tau))} + 1}{2} = \frac{1}{2} + \frac{\tau}{2}\frac{d(\log(IDC(\tau)))}{d\tau}$$

si ottiene una prima espressione di $E[\widehat{H_v(\tau)}]$ e $Var[\widehat{H_v(\tau)}]$:

$$E[\widehat{H_v(\tau)}] = \frac{1}{2} + \frac{\tau}{2}E\left[\frac{d(\log(\widehat{IDC_N}(\tau)))}{d\tau}\right]$$

$$Var[\widehat{H_v(\tau)}] = \frac{\tau^2}{4}E\left[\frac{d(\log(\widehat{IDC_N}(\tau)))}{d\tau}\right]$$

ove N è il numero totale di intervalli di durata τ considerati, legato al tempo totale di osservazione del traffico T_{tot} dalla relazione $N = \frac{T_{tot}}{\tau}$, mentre $\widehat{IDC}(\tau) = \frac{\widehat{s_N^2(\tau)}}{\widehat{\mu_N(\tau)}}$, indicando con $\widehat{\mu_N(\tau)}$ e $\widehat{s_N^2(\tau)}$ la media e la varianza stimate del processo di conteggio,

anch'esse variabili aleatorie gaussiane caratterizzate dalle seguenti medie e varianze [55]:

$$E[\widehat{\mu_N(\tau)}] = \mu_N(\tau) \qquad Var[\widehat{\mu_N(\tau)}] = \frac{\sigma_N^2(\tau)}{N}$$
$$E[\widehat{s_N^2(\tau)}] = \sigma_N^2(\tau) \qquad Var[\widehat{s_N^2(\tau)}] = \frac{2\sigma_N^4(\tau)}{N-1}$$

Applicando ora le formule per il calcolo di media e varianza di una generica funzione di due variabili aleatorie, date da:

$$\begin{aligned} E[g(X,Y)] \approx & g(\mu_X,\mu_Y) + \frac{1}{2}Var[X]\frac{d^2}{dx^2}g(x,y)|_{\mu_X,\mu_Y} + \frac{1}{2}Var[Y]\frac{d^2}{dy^2}g(x,y)|_{\mu_X,\mu_Y} \\ & + Cov[X,Y]\frac{d^2}{dx,dy}g(x,y)|_{\mu_X,\mu_Y} \end{aligned} \tag{5.3}$$

$$\begin{aligned} Var[g(X,Y)] \approx & Var[X]\left\{\frac{d}{dx}g(x,y)|_{\mu_X,\mu_Y}\right\}^2 + Var[Y]\left\{\frac{d}{dy}g(x,y)|_{\mu_X,\mu_Y}\right\}^2 \\ & + 2Cov[X,Y]\left\{\frac{d}{dx}g(x,y)|_{\mu_X,\mu_Y}\frac{d}{dy}g(x,y)|_{\mu_X,\mu_Y}\right\} \end{aligned} \tag{5.4}$$

e tenendo conto che nel caso specifico $Cov[\widehat{\mu_N(\tau)}, \widehat{s_N^2(\tau)}] = 0$ in virtù del teorema di Cochran [56], effettuando le opportune sostituzioni si ottiene:

$$E[\widehat{H_v(\tau)}] = \frac{1}{2} + \frac{\tau}{2}\left\{-\frac{\sigma_N^2(\tau)\mu'_N(\tau)}{N\mu_N^3(\tau)} + \frac{\sigma'^2_N(\tau)}{2N\mu_N^2(\tau)} + \frac{\mu_N(\tau)(-\frac{\sigma_N^2(\tau)\mu'_N(\tau)}{\mu_N^2(\tau)} + \frac{\sigma'^2_N(\tau)}{\mu_N(\tau)})}{\sigma_N^2(\tau)}\right\}$$

$$Var[\widehat{H_v(\tau)}] = \frac{\tau^2}{4}\left\{-\frac{2\sigma_N^2(\tau)\mu'_N(\tau)}{N\mu_N^3(\tau)} + \frac{\sigma'^2_N(\tau)}{N\mu_N^2(\tau)}\right\}$$

Considerando ora i valori di $\mu_N(\tau)$ e $\sigma_N^2(\tau)$ definiti in sezione 2.4 e sostituendo nuovamente, si ottengono le espressioni definitive 5.1 e 5.2, valide nel caso di una distribuzione dei tempi interarrivo di tipo **RPH2**.

5.1.3 Indipendenza dell'Indice di Variabilità dal numero di sorgenti di traffico

Se il processo dei tempi di interarrivo è costituito da M processi i.i.d. **RPH2**, indicando il tempo di interarrivo tra pacchetti generati da una singola sorgente di

traffico con Y_i e con Z quello del processo risultante, poichè $Z = min_i Y_i$, si ottiene:

$$P[Z > y] = \prod_{i=1}^{M} P[Y_i > y] = P[Y > y]^M$$

Utilizzando la funzione di ripartizione, che per un processo **RPH2** è data da:

$$F_Y(y) = 1 - w_1 e^{-ay} - w_2 e^{-by}$$

risulta:

$$1 - F_Z(y) = [1 - F_Y(y)]^M = [w_1 e^{-ay} + w_2 e^{-by}]^M = \sum_{i=1}^{M} \binom{M}{i} w_1^i w_2^{M-i} e^{-aiy-(M-i)by}$$

dalla quale appare evidente che il processo risultante non è **RPH2**. In [18] è stato però dimostrato che l'Indice di Variabilità di un processo aggregato, ottenuto sovrapponendo più sorgenti di traffico indipendenti, è dato da:

$$H_v(\tau) = \frac{1}{2}\left\{1 + \tau\left(\frac{\sum_{i=1}^{M} \frac{d(IDC_i(\tau))}{d\tau}(\frac{1}{\Lambda_i})}{\sum_{i=1}^{M} \frac{IDC_i(\tau)}{\Lambda_i}}\right)\right\}$$

ove $\Lambda_i = \frac{\sum_{j=1}^{M}}{\lambda_j}$. Se le sorgenti sono identiche, l'andamento di $H_v(\tau)$ si riduce pertanto alla 2.4.

Questo interessante risultato giustifica la scelta di $H_v(\tau)$ come attributo di classificazione: poichè il numero di sorgenti di traffico generanti un flusso è solitamente ignoto, è importante che la procedura di classificazione non sia influenzata dalla mancanza di tale informazione.

5.1.4 Errore di classificazione

Dal momento che la caratterizzazione statistica dei processi generatori dei flussi di traffico è in genere non nota, l'Indice di Variabilità deve essere stimato a partire dai dati.

Se il numero di campioni a disposizione è sufficientemente elevato, in virtù del teorema centrale del limite $H_v(\tau)$ può essere considerato una variabile aleatoria con distribuzione gaussiana.

Gli attributi di classificazione di ciascun campione sono pertanto costituiti dalla stima dell'Indice di Variabilità $\widehat{H_v(\tau)}$ valutato su diverse scale temporali τ_i ($i = 1, 2, ..., N$), quindi ad ogni istanza può essere associato un insieme di N variabili aleatorie $\widehat{H(\tau_1)}, \widehat{H(\tau_2)}, ..., \widehat{H(\tau_N)}$. Assumendo che, al variare di τ_i, $\widehat{H_v(\tau_i)}$ siano variabili aleatorie indipendenti, la densità di probabilità congiunta sarà data dal prodotto delle marginali e risulterà dunque pari a:

$$f_{H_1,H_2,...,H_N}(\mathbf{x}) = \prod_{i=1}^{N} \frac{1}{\sqrt{(2\pi)}\widehat{\sigma(\tau_i)}} e^{-\frac{1}{2}(\frac{x_i - \widehat{\mu_i(\tau_i)}}{\widehat{\sigma_i(\tau_i)}})^2}$$

Considerando ora differenti classi di traffico, ciascuna caratterizzata dal rispettivo Indice di Variabilità $\widehat{H_{vj}(\tau)}$, l'errore teorico di classificazione per la classe j (definito come la probabilità di attribuire un campione appartenente alla classe j ad una classe diversa da j) può essere calcolato come:

$$Err_j = \int_{\varphi_j} f_{Hv_j}(\mathbf{x})dx$$

ove: $\varphi_j = \{\mathbf{x} \in R^M : \exists \quad k \in 1, ..., M \quad k \neq j$ tale che $f_{Hv_k}(x) > f_{Hv_j}(x)\}$. Nel caso in cui le stime $\widehat{H_v(\tau_i)}$ non siano indipendenti, Err_j piò essere considerato un lower bound dell'errore di classificazione effettivo.

E' inoltre possibile effettuare una stima dell'errore di classificazione supponendo invece le stime $\widehat{H_v(\tau_i)}$ massimamente correlate: una volta nota la stima $\widehat{H_v(\tau_i)}$ per $i = \bar{i}$, le altre stime non saranno dunque portatrici di informazioni addizionali e il calcolo dell'errore di classificazione si riduce al caso monodimensionale.

5.2 Prestazioni del classificatore binario

In questa sezione viene effettuato un confronto tra le prestazioni del classificatore di Parzen e l'errore teorico calcolato analiticamente nella sezione precedente.

5.2.1 Generazione delle tracce sintetiche

Basandosi sui modelli statistici delle sorgenti di traffico descritti in sezione 5.1.1, sono state realizzate delle tracce sintetiche per effettuare l'addestramento e il test

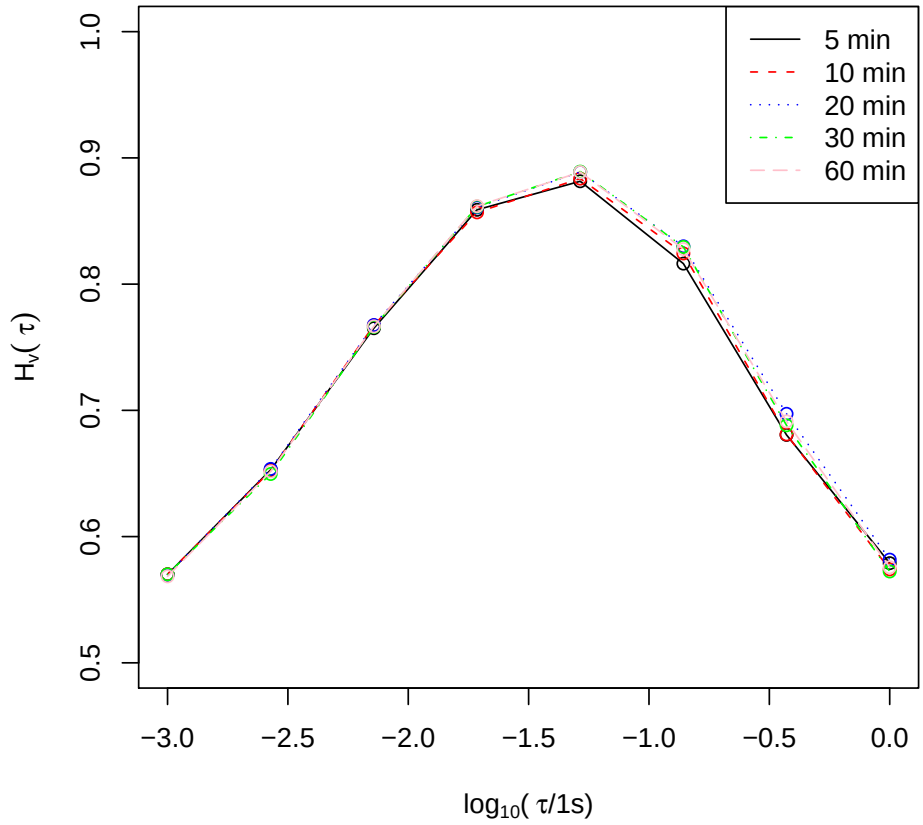

Figura 5.2: *Andamento dell'indice di variabilità stimato dai dati per le classe A.*

del classificatore binario. Per ognuna delle 2 classi (A e B) sono stati generati sinteticamente 100000 campioni di train della durata di 60 min e 100000 campioni di test per ciascuna delle seguenti durate: 5, 10, 20, 30 e 60 min (è importante notare che la stima di $H_v(\tau)$ effettuata su osservazioni di durata limitata non introduce errori sistematici, perciò il training può essere effettuato con tracce di durata maggiore, per le quali $\widehat{H_v(\tau)}$ presenta varianze più basse, mentre per il test possono essere scelte tracce di durata arbitraria). La procedura seguita è la seguente:

1. è stata generata una sequenza di tempi di interarrivo di durata complessiva pari a quella desiderata, estraendo i valori della variabile aleatoria $X(t)$ secondo la densità di probabilità corrispondente alla classe considerata.

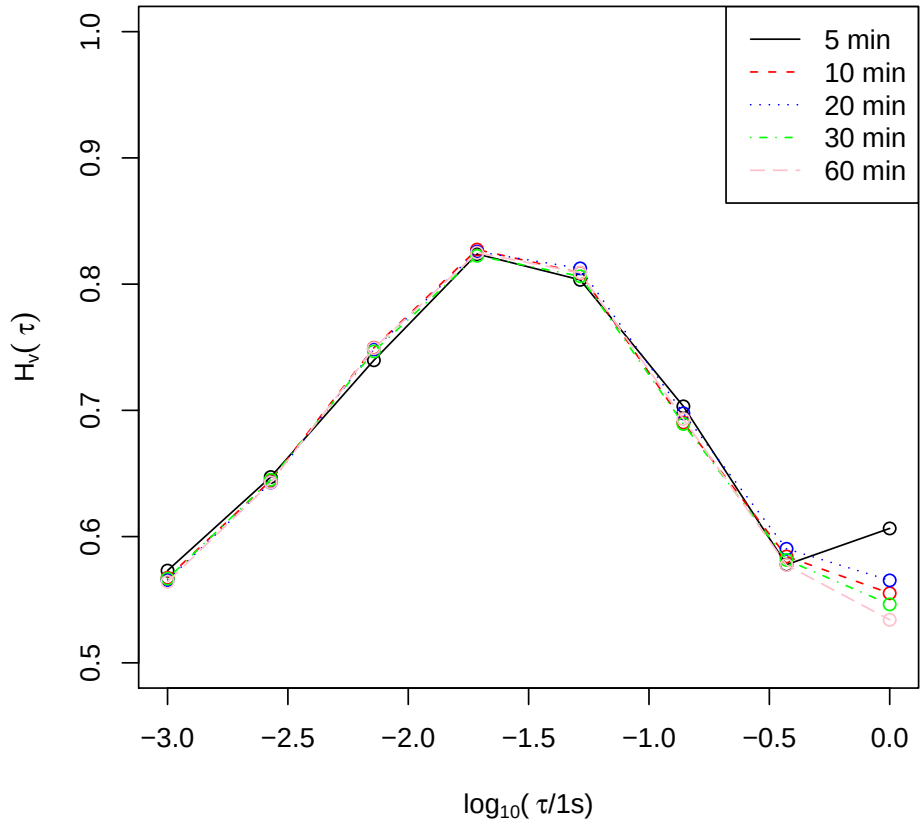

Figura 5.3: *Andamento dell'indice di variabilità stimato dai dati per le classe B.*

2. a partire dalla sequenza dei tempi di interarrivo, è stato calcolato il corrispondente processo degli arrivi.

3. in funzione del processo degli arrivi, è stato valutato l'Indice di Variabilità su 8 diverse scale temporali all'interno del range $(10^{-3}s, 1s)$ utilizzando il metodo della Varianza Aggregata proposto in [18] e descritto in sezione 2.3.2.

4. al vettore delle stime dell'Indice di Variabilità viene associata un'etichetta indicante la classe di appartenenza.

Un esempio dell'andamento della curva $\widehat{H_v(\tau)}$ per le classi A e B è mostrato nelle figure 5.2 e 5.3, che ne evidenziano la sostanziale indipendenza dalla durata della

traccia.

5.2.2 Confronto tra stime analitiche e statistiche dei dati sperimentali

Per entrambe le classi e ciascuna delle durate scelte, sono state calcolate le stime di media e varianza di ogni attributo utilizzando le espressioni ottenute grazie al teorema precedentemente enunciato e i risultati numerici sono stati confrontati con le statistiche calcolate sui dati sperimentali. I grafici 5.4, 5.5, 5.6 e 5.7 riportano un esempio rappresentativo per tracce di $300s$ di durata.

Risulta evidente lo scostamento tra i valori di varianza stimati e quelli effettivi, soprattuto per valori di τ compresi nel range $(10^{-3}{:}10^{-2}s)$. Ciò suggerisce che l'ipotesi iniziale di indipendenza tra $\widehat{H_v(\tau_i)}$ per differenti τ_i debba essere riconsiderata e che sussista una correlazione non trascurabile tra i vari attributi.

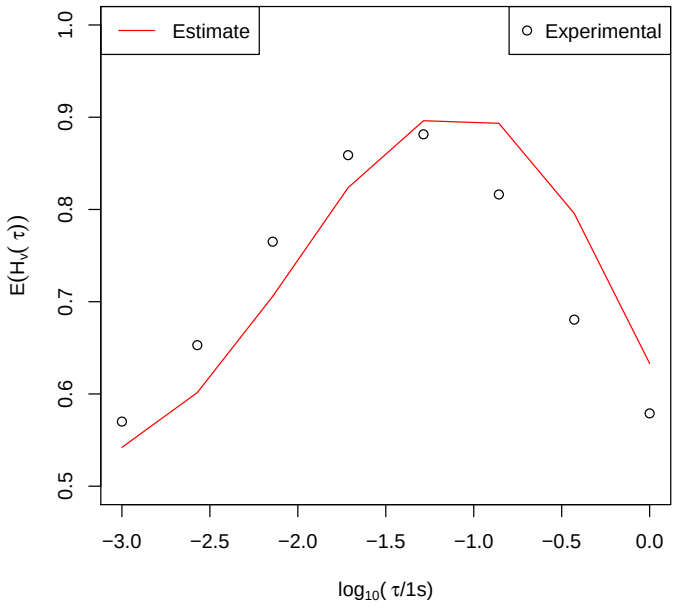

Figura 5.4: *Confronto tra andamento stimato e sperimentale del valor medio dell'Indice di Variabilità per la classe A.*

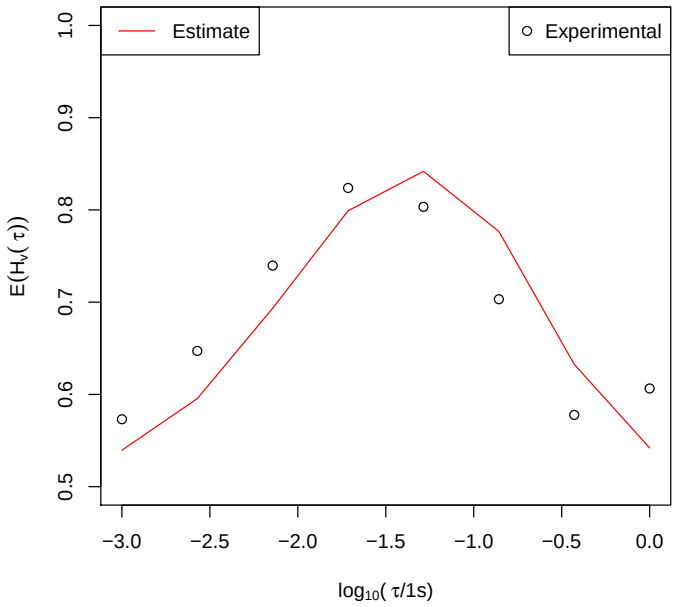

Figura 5.5: *Confronto tra andamento stimato e sperimentale del valor medio dell'Indice di Variabilità per la classe B.*

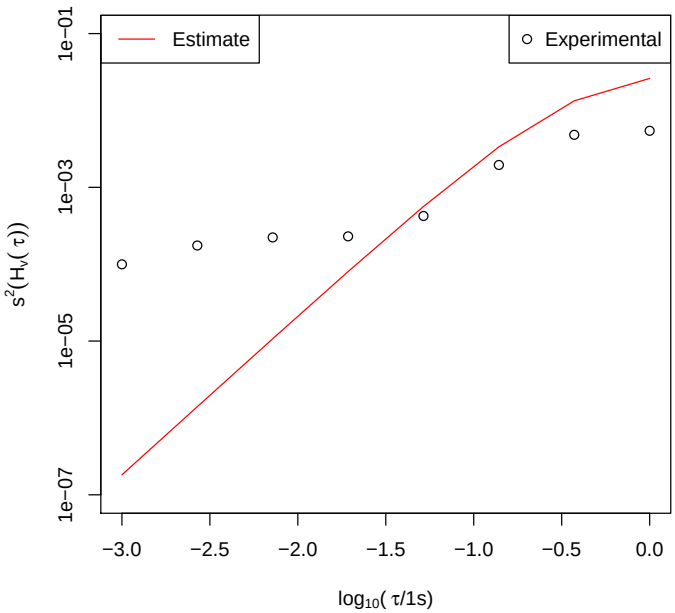

Figura 5.6: *Confronto tra andamento stimato e sperimentale della varianza dell'Indice di Variabilità per la classe A.*

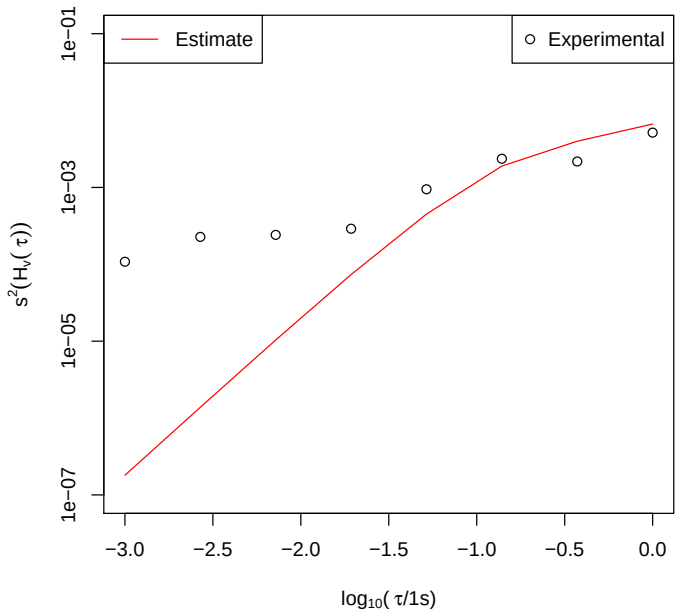

Figura 5.7: *Confronto tra andamento stimato e sperimentale della varianza dell'Indice di Variabilità per la classe B.*

5.2.3 Risultati sperimentali

Le prime 6 delle 8 stime dell'Indice di Variabilità ottenute secondo la procedura illustrata in sezione 5.2.1 (corrispondenti a scale temporali nel range $(10^{-3}s, 10^{-1}s)$) sono state utilizzate come attributi di classificazione, mentre le ultime 2 sono state escluse in quanto sono risultate poco accurate, specialmente per tracce di breve durata, inficiando le prestazioni del classificatore. Ciò è confermato dall'andamento crescente della varianza di $\widehat{H_v(\tau)}$, mostrato in figura 5.8 e 5.9.

Per ciascuna dei 5 gruppi di tracce di diverse durate, sono state effettuate 100 classificazioni sorteggiando ad ogni iterazione 1000 campioni di train e suddividendo invece i campioni di test in 100 sottoinsiemi da 1000 elementi ciascuno. L'errore di classificazione per le classi A e B è presentato nelle figure 5.10 e 5.11 e confrontato con le stime calcolate analiticamente in sezione 5.1.4. I lower bound sono risultati prossimi allo 0% per entrambe le classi e tutte le durate, pertanto sono stati omessi,

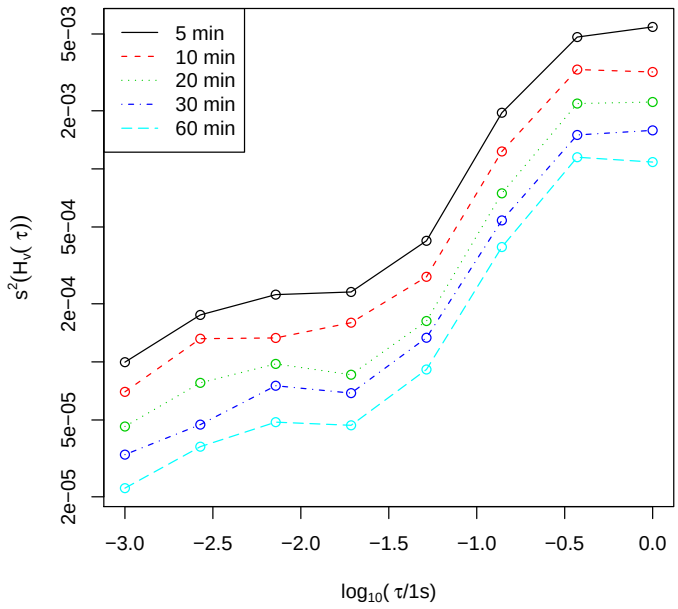

Figura 5.8: *Andamento della varianza dell'Indice di Variabilità per la classe A.*

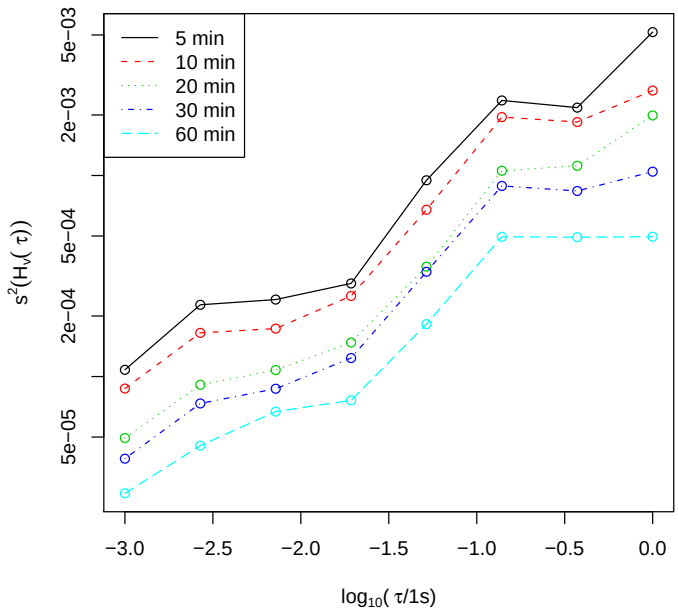

Figura 5.9: *Andamento della varianza dell'Indice di Variabilità per la classe B.*

mentre sono stati riportati gli intervalli di confidenza al 95%. Le stime degli errori sono state effettuate considerando separatamente ciascuno degli attributi, calcolando l'errore di classificazione relativo e scegliendo il valore numerico inferiore tra quelli ottenuti. Risulta evidente per entrambe le classi la coerenza delle stime degli errori rispetto ai dati sperimentali, che anzi per la maggior parte delle classificazioni si sono mantenuti al di sotto dei valori prospettati dalle stime stesse.

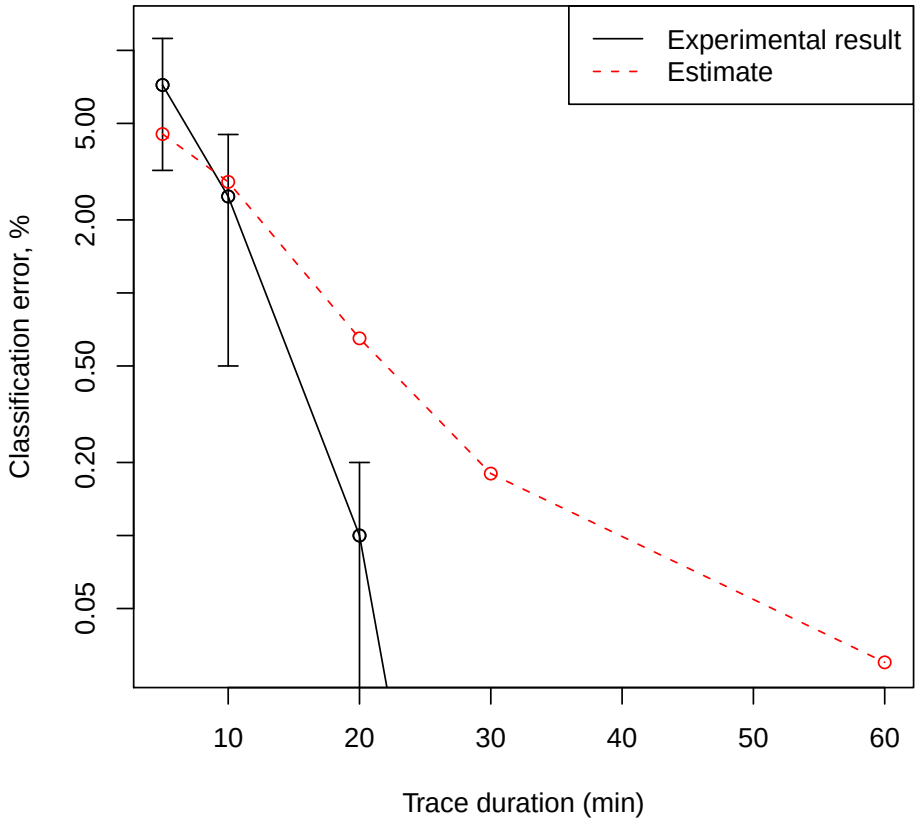

Figura 5.10: *Errore di classificazione per la classe A.*

L'esperimento è stato poi ripetuto con 2 ulteriori classi di traffico (**C** e **D**) caratterizzate dai valori di a e b riportati in tabella 5.3 e scelti con riferimento a [18]. I risultati ottenuti sono riassunti in tabella 5.4: in questo caso si è evidenziato un maggior sbilanciamento degli errori di classificazione a favore di una delle due

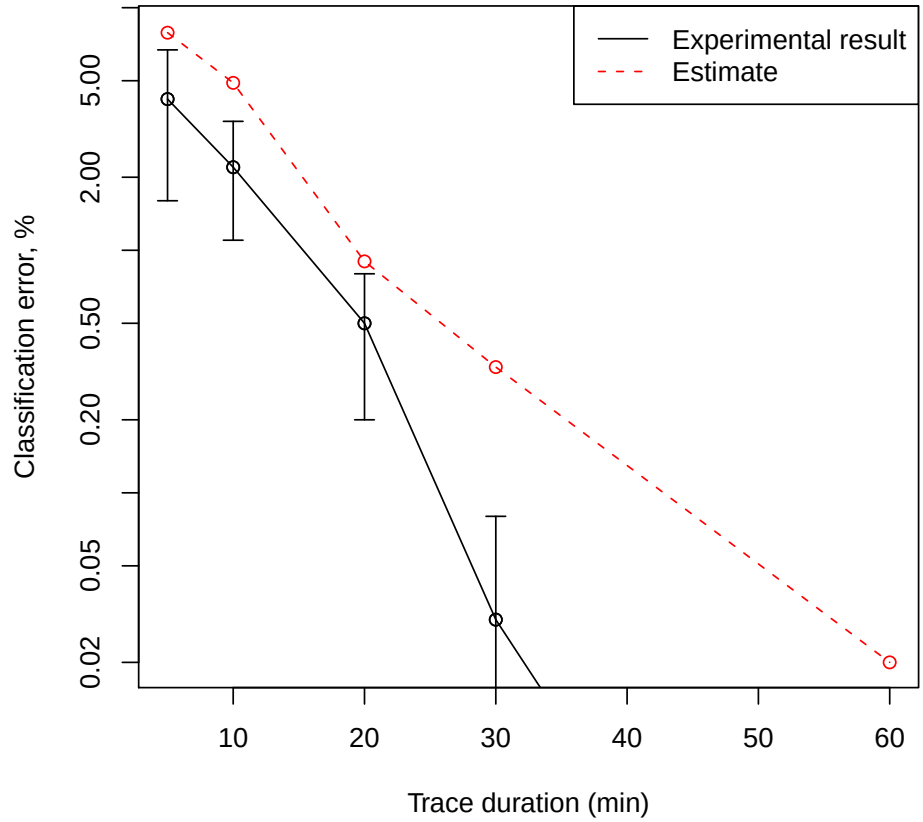

Figura 5.11: *Errore di classificazione per la classe B.*

	classe C	classe D
a	100 pacchetti/s	100 pacchetti/s
b	0.0001 s^{-1}	0.0001 s^{-1}
w_1	0.9	0.999

Tabella 5.3: *Parametri delle classi di traffico C e D.*

	5 min	10 min	20 min	30 min	60 min
Errore sperimentale classe C (%)	5.1	3.3	0.9	0.32	0.07
Stima errore classe C (%)	13.7 ± 1.6	7.8 ± 1.1	2.3 ± 0.6	1 ± 0.3	0.05 ± 0.04
Errore sperimentale classe D (%)	0.6	0.4	0.15	0.05	0.01
Stima errore classe D (%)	0.93 ± 0.35	$< 10^{-3}$	$< 10^{-3}$	$< 10^{-3}$	$< 10^{-3}$

Tabella 5.4: *Stima dell'errore di classificazione e risultati sperimentali per le classi C e D.*

classi, che è stata correttamente riconosciuta pressochè nella totalità delle istanze considerate. Ciò è dovuto alla notevole diversità nelle varianze degli attributi di classificazione, che differiscono in questo caso di diversi ordini di grandezza.

5.3 Conclusioni

I risultati sperimentali hanno confermato la fondatezza dell'indagine teorica e della valutazione analitica delle prestazioni del classificatore proposto. Tuttavia, i valori numerici ottenuti non consentono di affermare che sia possibile effettuare una classificazione accurata utilizzando il solo Indice di Variabilità come attributo. Esso dovrà essere pertanto affiancato da altri attributi statistici di classificazione, grazie ai quali sarà possibile ottenere un miglioramento delle prestazioni della tecnica descritta nel presente capitolo.

Capitolo 6

Conclusioni

Il presente lavoro di tesi è stato dedicato alla classificazione del traffico Internet sulla base di attributi statistici per-flusso (caratterizzanti una singola sequenza di pacchetti scambiati tra 2 specifici host e 2 rispettive porte di trasporto) e per-sorgente (ossia relativi a tutti i flussi di traffico generati dal medesimo processo).

Sebbene l'utilizzo di parametri statistici come attributi di classificazione sia ampiamente documentata in letteratura, non è ad oggi ancora stata condotta un'indagine con un approccio di tipo analitico. Avendo verificato sperimentalmente che l'introduzione di parametri per-sorgente consente di migliorare l'accuratezza e di ridurre il ritardo di classificazione rispetto ai soli parametri per-flusso, si è scelto di valutare analiticamente le prestazioni di un classificatore di Parzen binario che utilizza come attributo di classificazione l'Indice di Variabilità calcolato per diverse scale temporali. Per tale classificatore è stato valutato l'errore di classificazione ed esso è stato confrontato con i risultati sperimentali ottenuti su tracce di traffico generate sinteticamente.

Nonostante le prestazioni ottenute siano da ritenersi non pienamente soddisfacenti, l'Indice di Variabilità può essere considerato un attributo promettente da affiancare ad altri parametri statistici per-flusso e per-sorgente. Ciò può essere fatto sfruttando una tipologia ibrida di classificatore, che si limiti all'utilizzo dei soli attributi per-flusso nel caso il numero di flussi generati da una specifica sorgente siano insufficienti per una valutazione accurata delle statistiche per-sorgente, e che

in caso contrario effettui la classificazione avvalendosi di entrambi i tipi di attributi. Sono state quindi definite l'architettura e le modalità di addestramento di tale classificatore e ne sono state valutate le prestazioni utilizzando due differenti algoritmi di apprendimento (Random Forest e Support Vector Machines): l'accuratezza è in entrambi i casi prossima al 90%, mentre si riduce il ritardo di classificazione rispetto a un classificatore addestrato con i soli attributi per-flusso che fornisca risultati confrontabili.

Elenco delle figure

Elenco delle tabelle

Indice

Bibliografia

[1] T. Nguyen and G. Armitage. A survey of techniques for internet traffic classification using machine learning. *Communications Surveys and Tutorials*, Vol. 10(Nr. 4):56–76, 2008.

[2] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller. An overview of ip flow-based intrusion detection. *Communications Surveys Tutorials, IEEE*, 12(3):343 –356, 2010.

[3] Hyunchul Kim, KC Claffy, Marina Fomenkov, Dhiman Barman, Michalis Faloutsos, and KiYoung Lee. Internet traffic classification demystified: myths, caveats, and the best practices. *CoNEXT '08: Proceedings of the 2008 ACM CoNEXT Conference*, pages 1–12, 2008.

[4] Wei Li, Marco Canini, Andrew W. Moore, and Raffaele Bolla. Efficient application identification and the temporal and spatial stability of classification schema. *Computer Networks*, 53(6):790 – 809, 2009.

[5] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield. Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification. *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement. New York, NY, USA*, page 135–148, 2004.

[6] T. Auld, A.W. Moore, and S.F. Gull. Bayesian neural networks for internet traffic classification. *Neural Networks, IEEE Transactions on*, 18(1):223 –239, jan. 2007.

[7] T. Nguyen and G. Armitage. Synthetic sub-flow pairs for timely and stable IP traffic identification. *Proc. Australian Telecommunication Networks and Application Conference*, December 2006.

[8] J. Park, H. R. Tyan, and C. C. Kuo. Internet traffic classification for scalable QoS provision. *Multimedia and Expo, 2006 IEEE International Conference on*, July 2006.

[9] J. Park, H. R. Tyan, and C. C. Kuo. GA-based internet traffic classification technique for QoS provisioning. *Intelligent Information Hiding and Multimedia Signal Processing, International Conference on*, pages 251–254, 2006.

[10] Sebastian Zander, Thuy Nguyen, and Grenville Armitage. Automated traffic classification and application identification using machine learning. *Local Computer Networks, Annual IEEE Conference on*, 0:250–257, 2005.

[11] Manuel Crotti, Maurizio Dusi, Francesco Gringoli, and Luca Salgarelli. Traffic classification through simple statistical fingerprinting. *SIGCOMM Comput. Commun. Rev.*, 37(1):5–16, 2007.

[12] G. Verticale and P. Giacomazzi. Performance evaluation of a machine learning algorithm for early application identification. *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, pages 845 –849, oct. 2008.

[13] Murat Soysal and Ece Guran Schmidt. Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison. *Performance Evaluation*, 67(6):451 – 467, 2010.

[14] Shijun Huang, Kai Chen, Chao Liu, A. Liang, and Haibing Guan. A statistical-feature-based approach to internet traffic classification using machine learning. *Ultra Modern Telecommunications Workshops, 2009. ICUMT '09. International Conference on*, pages 1 –6, oct. 2009.

[15] R. Holanda Filho, M.F. Fontenelle do Carmo, J. Maia, and G.P. Siqueira. An internet traffic classification methodology based on statistical discriminators. *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, pages 907 –910, apr. 2008.

[16] Jing Yuan, Zhu Li, and Ruixi Yuan. Information entropy based clustering method for unsupervised internet traffic classification. *Communications, 2008. ICC '08. IEEE International Conference on*, pages 1588 –1592, may. 2008.

[17] Thomas Karagiannis, Konstantina Papagiannaki, and Michalis Faloutsos. Blinc: multilevel traffic classification in the dark. *SIGCOMM Comput. Commun. Rev.*, 35(4):229–240, 2005.

[18] Georgios Y. Lazarou, Julie Baca, Victor S. Frost, and Joseph B. Evans. Describing network traffic using the index of variability. *IEEE/ACM Trans. Netw.*, 17(5):1672–1683, 2009.

[19] Will E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Trans. Netw.*, 2(1):1–15, 1994.

[20] Vern Paxson and Sally Floyd. Wide area traffic: the failure of poisson modeling. *IEEE/ACM Trans. Netw.*, 3(3):226–244, 1995.

[21] M.E. Crovella and A. Bestavros. Self-similarity in world wide web traffic: evidence and possible causes. *Networking, IEEE/ACM Transactions on*, 5(6):835 –846, dec. 1997.

[22] Carl Nuzman, Iraj Saniee, Wim Sweldens, and Alan Weiss. A compound model for tcp connection arrivals for lan and wan applications. *Computer Networks*, 40(3):319 – 337, 2002.

[23] V. Vapnik. *Statistical learning theory.* Wiley, 1998.

[24] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415 –425, mar. 2002.

[25] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[26] B. P. S. H. Chen and C. J. Lin. A tutorial on support vector machines. *Applied Sthocastic Models in Business and Industry*, Vol. 21(Nr. 2):111–136, 2005.

[27] U. Kreel. *Advances in Kernel Methods Support Vector Learning*, chapter Pairwise classification and Support Vector Machines, pages 255–268. M. MIT Press, Cambridge Ed., 1999.

[28] W. K. R. Kohavi, J. R. Quinlan, and J. M. Zytkow. *Handbook of Data Mining and Knowledge Discovery*, chapter Decision-tree discovery, pages 267–276. Oxford University Press, 2002.

[29] R. Shapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. *Annals of Statistics*, Vol. 26:1651–1686, 1998.

[30] L. Breiman. Random forests. *Machine Learning*, Vol. 45:5–32, 2001.

[31] J. F. T. Hastie, R. Tibshirani, and J. Franklin. Random forests. *Machine Learning*, Vol. 27:83–85, June 2005.

[32] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*, chapter Random Forests, pages 601–616. Springer-Verlag., second edition, 2009.

[33] D.W. Allan and J.A. Barnes. A modified allan variance with increased oscillator characterization ability. *Thirty Fifth Annual Frequency Control Symposium*, pages 470 – 475, 1981.

[34] S. Bregni and L. Jmoda. Accurate estimation of the hurst parameter of long-range dependent traffic using modified allan and hadamard variances. *Communications, IEEE Transactions on*, 56(11):1900 –1906, nov. 2008.

[35] S. Bregni, R. Cioffi, and M. Decina. An empirical study on time-correlation of gsm telephone traffic. *Wireless Communications, IEEE Transactions on*, 7(9):3428 –3435, sep. 2008.

[36] J. Beran. *Statistics for Long-Memory Processes*. Chapman & Hall, 1994.

[37] J. Park and W. Willinger. *Self-similar Network Traffic and Performance Evaluation*. Wiley, New York, 2000.

[38] Giacomo Verticale. An empirical study of self-similarity in the per-user-connection arrival process. *AICT '09: Proceedings of the 2009 Fifth Advanced International Conference on Telecommunications*, pages 101–106, 2009.

[39] S. Bregni. *Time and frequency measurement techniques in telecommunications*. Wiley, 2002.

[40] R. Gusella. Characterizing the variability of arrival processes with indexes of dispersion. *IEEE J. Sel. Areas Commun.*, Vol. 9(Nr. 2):203–211, February 1991.

[41] M. Greiner, M. Jobmann, and L. Lipsky. The importance of power-tail distribution for telecommunications traffic modeling. *Inst. fur Informatik, Tehcnische Univ. Munchen, Munchen, Germany, Tech. Rep.*, 1995.

[42] *NetAI*. `http://caida.swin.edu.au/urp/dstc/netai/`, versione 0.1.

[43] *Ambiente R*. `http://www.r-project.org/`, versione 2.8.0.

[44] *Progetto PMA*. `http://pma.nlanr.net`.

[45] *NLANR*. `http://www.nlanr.net`.

[46] *University of Auckland Internet infrastructure.* `http://wand.cs.waikato.ac.nz/wits/auck/6/auckland/infra.png`.

[47] *Tracce Auckland VI.* `http://pma.nlanr.net/Traces/long/auck6.html`.

[48] *Tracce New Zealand II.* `http://pma.nlanr.net/Traces/long/nzix2.html`.

[49] *University of Leipzig's central Internet access router.* `http://pma.nlanr.net/Special/gmwin.png`.

[50] *Tracce Leipzig II.* `http://pma.nlanr.net/Special/leip2.html`.

[51] *NetMate Meter.* `http://sourceforge.net/projects/netmate-meter/`,versione 0.9.4.

[52] S. Bregni, D. Lucerna, C. Rottondi, and G. Verticale. Using per-source measurements to improve performance of internet traffic classification. *Latincom, IEEE, Latin-American Conference on Communications*, sept. 2010.

[53] *Raccomandazione ETSI 3GPP TR 30.03U.*

[54] Andrea Bianco, Gianluca Mardente, Marco Mellia, Maurizio Munafò, and Luca Muscariello. Web user-session inference by means of clustering techniques. *IEEE/ACM Trans. Netw.*, 17(2):405–416, 2009.

[55] K. Fukunaga. *Introduction to Statistical Pattern Recognition*, chapter Random Vectors and their properties, pages 17–23. Werner Rheinboldt, Academic Press, second edition, 1990.

[56] R. B. Bapat. *Linear Algebra and Linear Models.* Springer, second edition, 2000.

Ringraziamenti

Desidero in primo luogo ringraziare il Prof. Maurizio Decina per avermi dato la possibilità di collaborare con il suo gruppo di ricerca e l'Ing. Giacomo Verticale per la dedizione e la pazienza con le quali ha quotidianamente supportato il mio lavoro di tesi.

Ringrazio inoltre i professori Achille Pattavina, Paolo Giacomazzi e Stefano Bregni, l'Ing. Guido Maier e l'Ing. Massimo Tornatore per il loro supporto, Diego Lucerna, con cui ho condiviso tante giornate di lavoro, i dottorandi e i colleghi tesisti del laboratorio Bonsai, con i quali ho piacevolmente trascorso il mio ultimo anno da studentessa.

Grazie infine a tutti i professori e i ricercatori del DEI, che tanto spesso mi hanno dato dimostrazione di stima e fiducia e mi hanno resa orgogliosa di potermi accostare al mondo della ricerca: negli ultimi cinque anni ho avuto modo di conoscere non solo validi professionisti, ma prima di tutto persone splendide dal punto di vista umano, che mi hanno dato modo di entrare in contatto con un ambiente cosmopolita, aperto e vivo, mostrando come l'amore per la conoscenza possa superare qualsiasi divario sociale e culturale.

Il ringraziamento più affettuoso va poi ai miei genitori, il cui appoggio è sempre stato costante ed amorevole, alla mia famiglia, ai miei amici e a tutti coloro che mi hanno sostenuta e incoraggiata in ogni momento del mio percorso di studi.