Consiglio Nazionale delle Ricerche

METODOLOGIA, TECNICHE E RISULTATI DI ESPERIMENTI DI DATA MINING NEL CONTRASTO DELL'EVASIONE FISCALE

F. Bonchi, F. Giannotti, G. Mainetto, D. Pedreschi

Rapporto CNUCE-B4-1999-004

CNUCE

Pisa

METODOLOGIA, TECNICHE E RISULTATI DI ESPERIMENTI DI DATA MINING NEL CONTRASTO ALL'EVASIONE FISCALE[1]

F. Bonchi[‡], F. Giannotti[*], G. Mainetto[*], D. Pedreschi[‡]

[*]CNUCE–CNR, Via S. Maria 36, 56126 Pisa
[‡]Dip. di Informatica, C.so Italia 40, 56125 Pisa

Abstract. *In this paper we present the metholodogy, the techniques we have used for dealing with problems of fiscal frauds in Italy. We describe in details the results of our experiments, the problems we have solved, which are a subset of the set of problems which constitutes the large body of the fiscal fraud detection problem in Italy.*

Abstract. *Questo articolo presenta la metodologia e le tecniche che abbiamo usato per trattare problemi di frode fiscale svoltesi in Italia. Descriviamo in dettaglio i risultati di vari esperimenti e i problemi che sono stati risolti, che sono un sottoinsieme dei problemi che costituiscono l'imponente problema del rilevamento delle frodi fiscali in Italia.*

Categories and Subject Descriptors: H, H2, H2.4, H2.8, H3.3

Keywords and Phrases: Knowledge Discovery, Data Mining, Classification, Decision Trees.

1. INTRODUZIONE: IL PROBLEMA AFFRONTATO

L'evasione fiscale in Italia è un problema di notevole rilevanza non solo da un punto di vista dell'etica e della possibilità di perseguire le politiche sociali eque, infatti questo fenomeno determina ingiustizie distributive e ha effetti distorsivi sull'allocazione delle risorse, ma anche da un punto di vista strettamente macroeconomico, nel senso che può ad esempio rendere difficile il conseguimento del pareggio nel bilancio di uno Stato. Stime del fenomeno evasione in Italia sono estremamente difficoltose, ma possono derivare da analogie con comportamenti evasivi in Paesi che al proposito dispongono metodologie di rilevamento e di dati più affidabili. Si stima che se, ad esempio, gli italiani dal 1970 in poi avessero evaso le imposte tanto quanto i cittadini statunitensi e lo Stato italiano fosse stato in grado di accertarsene, allora il debito pubblico in Italia sarebbe stato nel 1996 di poco superiore all'80%, anzichè il 120% del PIL. Se il loro comportamento nello stesso periodo fosse stato analogo a quello dei cittadini del Regno Unito e la Guardia di Finanza fosse stata altrettanto efficiente, allora il debito pubblico italiano nel 1996 sarebbe stato intorno al 60% del PIL [Alesina96].

L'opera di contrasto dell'evasione fiscale tocca moltissimi aspetti che vanno dalla necessità di far venire alla luce l'economia sommersa, a quello di modificare leggi e norme che permettono l'elusione fiscale, passando attraverso la cosidetta erosione fiscale [Tanzi93]. La forma di evasione fiscale che è indirettamente alla base di questo articolo è quella più diffusa che consiste nel dichiarare un ammontare di reddito inferiore a quello effettivamente guadagnato o nel reclamare un ammontare di esenzioni e

[1] Questo lavoro è stato in parte finanziato dal Progetto "Un sistema intelligente per la Individuazione della Evasione

deduzioni oltre il livello lecito. Sono questi i cosiddetti evasori parziali. Gli evasori parziali costituiscono comunque una quantità di evasione di tutto rispetto, che varia a seconda delle stime dal 3% al 10% del PIL (si veda il già citato [Tanzi93]).

In questo contesto evasivo, è noto che gli Enti dello Stato preposti al controllo e al contrasto di un fenomeno così rilevante sono fortemente interessati a tutte quelle tecniche anche automatiche che permettono di rendere più incisiva la loro azione. Detti Enti si trovano di fronte a questa situazione per quanto riguarda le sorgenti informative: dispongono di tutte le informazioni storiche sulle denunce dei redditi, organizzate per categorie di contribuenti; possono facilmente avere accesso a altre sorgenti informative che permettono di "incrociare" i dati segnalando eventuali potenziali anomalie; dispongono di tutte le informazioni storiche sugli accertamenti effettuati e, in particolare, sul loro esito. Inoltre, gli stessi Enti si trovano in assoluta carenza di personale da utilizzare per il controllo effettivo, cioè sul campo, delle denunce. Infine, è noto che il costo di un controllo su un soggetto potenzialmente evasore è molto oneroso dal punto di vista dell'impegno delle scarse risorse umane disponibili presso l'Ente.

Dal quadro complessivo sopra sommariamente ricordato, discendono molti problemi interessanti. Fra essi, un primo problema di ottimizzazione per l'Ente in questione è il seguente: come si può fare per individuare un numero ristretto di casi sui cui indirizzare gli accertamenti, così da ottimizzare l'utilizzo delle scarse risorse umane disponibili? Un secondo problema significativo, strettamente correlato al precedente, è questo: è possibile, sulla base della disponibilità delle predette sorgenti informative, predisporre un quadro oggettivo che permetta di stimare quale sarebbe il recupero di imposta evasa a fronte di un incremento/decremento del numero di addetti agli accertamenti?

Da queste ed altre domande che attengono principalmente la politica nel settore fiscale e nella gestione degli Enti dello Stato interessati, ne derivano alcune di carattere tecnico, le sole alle quali siamo qui interessati. Esse si incentrano sul fatto che la tecnologia del data mining sia applicabile in questo ambito articolato e complesso. Quali sono le tecniche di data mining utilizzabili per questi problemi? Quale è la metodologia che bisogna adottare per trattare i dati disponibili nelle sorgenti informative così da renderli utilizzabili con profitto dalle tecniche individuate? Queste sono alcune delle principali domande tecniche a cui tenteremo di dare una risposta esauriente in questo articolo.

L'articolo che qui presentiamo illustra parte dei risultati ottenuti dal progetto "Un sistema intelligente per la individuazione della evasione fiscale" che ha visto coinvolti da un lato l'Ente preposto al controllo e al contrasto della evasione fiscale e dall'altro Università ed Enti di Ricerca. L'articolo è così organizzato. Nella prossima Sezione si introduce il contesto metodologico e tecnologico utilizzato in questa esperienza e cioè la metodologia di scoperta della conoscenza e la tecnica di data mining detta albero delle decisioni. Si analizzano le caratteristiche principali degli algoritmi ID3 e C4.5, usati per costruire alberi di decisione, allo scopo di introdurre terminologia e concetti utili nel resto dell'articolo. La terza sezione, quella centrale di questo articolo, illustra la

Fiscale" con il contratto n. A07C981137.

metodologia e la tecnica all'opera sul caso concreto sperimentato. I risultati ottenuti vengono commentati. La quarta ed ultima sezione brevemente riassume il contributo di questo lavoro.

2. METODOLOGIA E TECNICA UTILIZZATA

2.1 La Metodologia di Scoperta della Conoscenza

Le metodologie che fanno uso di tecniche di data mining sono classificate sulla base del fatto che la conoscenza sia fornita da un utente esperto o venga generata automaticamente [Berry97]. Nel primo caso, detto *Verifica di Ipotesi*, si tratta di un approccio top-down che cerca di sostanziare o confutare ipotesi sui dati precostituite, formulate cioè dall'utente esperto sulla base della sua conoscenza. Questa è una analisi passiva perchè verifica se i dati a disposizione sono consistenti con l'ipotesi formulata. Nel secondo caso, cioè nella *Scoperta di Conoscenza*, ci troviamo viceversa di fronte a un approccio bottom-up che parte dai dati e cerca di scoprirne caratteristiche non note a priori. Questa è una analisi attiva, in cui sono i dati stessi a suggerire possibili ipotesi sul significato del loro contenuto. È frequente il caso in cui entrambe le metodologie vengano utilizzate, passando da una all'altra, per affrontare un problema di analisi concreto.

Nel problema studiato si è fatto uso della *Scoperta di Conoscenza* poichè, data la complessità e la quantità dei dati disponibili, nessuno era in grado di formulare ipotesi relative alle caratteristiche comuni degli evasori fiscali. La *Scoperta di Conoscenza* può essere *diretta* o *indiretta*. In quella diretta, lo scopo di tutta l'attività è spiegare il valore di qualche attributo in termini di tutti gli altri attributi. In genere è presente un campo privilegiato in ogni dato, detto *target*, più interessante degli altri e si vuole stimarlo, o classificarlo, o predirlo. In quella indiretta non ci sono attributi privilegiati, si vuole cercare in modo cieco dei pattern sui dati che potrebbero essere significativi. La differenza fra i due tipi di scoperta di conoscenza risulta evidente quando si considera il loro utilizzo: si usa la indiretta quando si vuole individuare delle relazioni fra dati mentre la diretta serve allorchè si vuol spiegare perchè certe relazioni esistano una volta che sono state individuate.

Le metodologie utilizzate nella scoperta della conoscenza diretta sono suddivise nelle seguenti sei fasi.

1. Identificazione delle sorgenti di dati disponibili
2. Individuazione del modello di analisi
3. Preparazione dei dati per l'analisi
4. Riduzione dei dati
5. Costruzione ed allenamento del modello
6. Valutazione del modello

Delle predette fasi daremo una descrizione dettagliata nella Sez. 3, dove si descrivono gli esperimenti effettuati. Qui ricordiamo che l'obbiettivo tecnico generale nella scoperta di conoscenza diretta è quello di predire il valore di un attributo *target* sulla base dei valori assunti da altri attributi dello stesso dato,

e che per costruire il modello predittivo, detto anche classificatore, si usano i dati del passato in cui il valore dell'attributo target è noto.

Inizialmente si devono identificare i dati e prepararli per l'analisi. Successivamente, per la costruzione ed allenamento del modello, è necessario partizionare i dati in due insiemi. Una prima partizione, detta *training set* e cioè insieme per l'allenamento, è usata per costruire il modello iniziale; essa contiene dati del passato con valori noti dell'attributo target. La seconda partizione, il *test set*, è usata per verificare la correttezza del modello costruito. Ovviamente, anche questa seconda partizione contiene dati storici con valori noti per l'attributo target che vengono utilizzati per fare una analisi qualitativa del modello predittivo costruito. Nell'ultima fase, quella di valutazione del modello, si confrontano e si compongono le predizioni dei classificatori costruiti nelle precedenti fasi alla luce anche di considerazioni quantitative.

2.2 Le Tecniche di Data Mining: gli Alberi di Decisione

Una prima fase della sperimentazione è stata dedicata allo studio e alla applicazione di varie tecniche di data mining quali clustering, regole associative, regressione e alberi di decisione, nel tentativo di far emergere dai dati dei profili tipici, considerando poi sospetti i soggetti che si discostavano da questi profili tipici. L'obiettivo, quindi, era quello di costruire un sistema capace di separare i dati disponibili in due classi, distinguendo tra i soggetti a comportamento tipico e i soggetti a comportamento atipico (e quindi sospetti). Questo approccio non ha portato a buoni risultati, anche perché nel dominio in questione il comportamento evasivo è più frequente del comportamento non evasivo.

Pur non portando a risultati tangibili, questa prima fase della sperimentazione è stata molto utile per la comprensione del dominio e dei parametri in gioco. Dal punto di vista dei dati, si è definitivamente abbandonata l'idea di utilizzare tutti i dati, anche quelli senza gli attributi relativi all'accertamento. Dal punto di vista delle tecniche, la scelta è caduta sull'utilizzo degli alberi di decisione. Il motivo della scelta dipende da alcuni fattori. Innanzitutto, il modello predittivo che si ottiene è di facile interpretazione poichè assume la forma di regole esplicite. Ciò permette di valutare consistentemente i risultati, identificando gli attributi chiave del processo. Inoltre, l'uso degli alberi di decisione si rivela particolarmente efficace quando, come nel nostro caso, i dati disponibili sono talvolta di qualità incerta: risultati non attendibili sono facilmente identificabili allorchè le regole utilizzate sono esplicite. Infine, per quanto riguarda gli aspetti del problema connessi agli interessi della ricerca, le regole utilizzate negli alberi di decisione sono facilmente esprimibili in un linguaggio di programmazione logico.

Per gli esperimenti condotti sono stati utilizzati i seguenti strumenti che fanno uso di determinati algoritmi:

- See5/C5.0 utilizza l'algoritmo C5.0 che è una evoluzione di C4.5 [See5]
- KnowledgeSEEKER può usare uno degli algoritmi CHAID, CART, ID3 [KSEEKER]

Questi due strumenti rappresentano lo stato dell'arte tra i sistemi di supporto ad una metodologia di

Scoperta di Conoscenza Diretta che usano Alberi di Decisione. Si differenziano non solo per gli algoritmi utilizzati ma anche per le funzionalità messe a disposizione. Mentre il primo strumento è stato usato direttamente per costruire i modelli predittivi utilizzati negli esperimenti, il secondo è stato utilizzato per l'indagine interattiva del dominio applicativo. Nel seguito, analizzeremo le caratteristiche essenziali degli algoritmi ID3 e C4.5, gli algoritmi principali usati nella sperimentazione, alla luce di un esempio comune.

2.2.1 L'esempio e alcune definizioni generali

La base di dati di esempio mette a disposizione delle ennuple con valori relativi alle condizioni metereologiche. Il target è determinare se esistono le condizioni per giocare a golf.

Attributo	*Tipo*	*Valori possibili*
Tempo	Discreto	sole, coperto, pioggia
Temperatura	Continuo	interi
Umidità	Continuo	interi
Ventoso	Discreto	vero, falso

L'insieme di ennuple che rappresenta il training set, compreso il valore per l'attributo target Gioca, è il seguente:

Tempo	*Temperatura*	*Umidità*	*Ventoso*	*Gioca*
sole	85	85	falso	no
sole	80	90	vero	no
coperto	83	78	falso	sì
pioggia	70	96	falso	sì
pioggia	68	80	falso	sì
pioggia	65	70	vero	no
coperto	64	65	vero	sì
sole	72	95	falso	no
sole	69	70	falso	sì
pioggia	75	80	falso	sì
sole	75	70	vero	sì
coperto	72	90	vero	sì
coperto	81	75	falso	sì
pioggia	71	80	vero	no

Negli alberi di decisione ciascun nodo non foglia corrisponde a un attributo diverso da quello target e ciascun arco corrisponde a un valore possibile per quell'attributo. Una foglia dell'albero specifica il valore atteso per l'attributo target per le ennuple descritte dal cammino che separa la radice da quella foglia. Per costruire un "buon" albero di decisione, dato un nodo N interno all'albero, ciascun ulteriore nodo interno dovrebbe essere associato a quell'attributo che è il più informativo fra gli attributi non ancora considerati nel cammino dalla radice a N. A questo fine, si usa una nozione di misura della quantità di informazione derivata dalla teoria dell'informazione quale l'*entropia*.

In generale, data una distribuzione di probabilità $P=(p_1, p_2, \ldots, p_n)$, l'*Informazione* convogliata da questa distribuzione, chiamata anche la Contrentropia di P, viene rappresentata dalla seguente formula:

$$\mathbf{I}(P) = -(p_1 * \log_2(p_1) + p_2 * \log_2(p_2) + \ldots + p_n * \log_2(p_n))$$

Se un insieme T di ennuple viene suddiviso tramite una partizione P composta da k classi disgiunte

C_1, C_2, ..., C_k sulla base del valore dell'attributo target, allora l'informazione necessaria per identificare la classe di un elemento di T è **Info**(T) = **I**(P), dove P è la probabilità di distribuzione della partizione e cioè:

$$\mathbf{I}(P) = (|C_1| / |T|, |C_2| / |T|, \ldots , |C_k| / |T|)$$

Nel caso dell'esempio, **Info**(T) = **I**(9/14, 5/14) = 0.94 poichè in 5 casi su 14 non si gioca a golf.

Se si partiziona l'insieme delle ennuple T sulla base del valore di un attributo A diverso da quello target in n classi disgiunte T_1, T_2, ..., T_n, allora l'informazione necessaria per identificare la classe di un elemento di T diventa la media pesata dell'informazione necessaria per identificare la classe di un elemento di T_i, cioè la media pesata di **Info**(T_i):

$$\mathbf{Info}(A,T) = \sum_{i=1,n} \frac{|T_i|}{|T|} * \mathbf{Info}(T_i)$$

Nel caso dell'esempio, per l'attributo *Tempo* abbiamo:

$$\mathbf{Info}(Tempo, T) = 5/14 * \mathbf{I}(2/5, 3/5) + 4/14 * \mathbf{I}(4/4,0) + 5/14 * \mathbf{I}(3/5, 2/5) = 0.694$$

poichè sui 5 casi di "sole", in 2 si gioca e in 3 no; in 4 su 4 di tempo "coperto" si gioca a golf e, infine, sui 5 di "pioggia" in 2 si gioca e in 3 no.

Si consideri ora la quantità **Gain**(A, T) così definita: **Gain**(A, T) = **Info**(T) - **Info**(A, T)

Questa quantità, che numericamente rappresenta la differenza fra l'informazione necessaria per identificare un elemento di T e l'informazione necessaria per identificare un elemento di dopo che il valore dell'attributo A è noto, rappresenta il guadagno nell'informazione dovuto all'attributo A.

Nel caso dell'esempio, per l'attributo *Tempo* abbiamo:

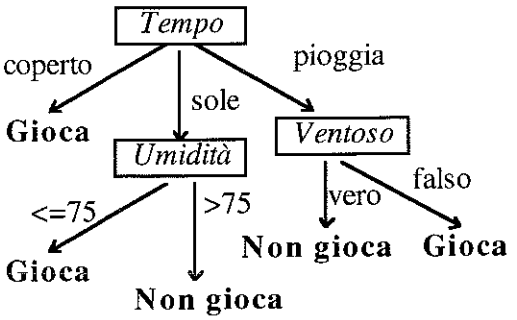
$$\mathbf{Gain}(Tempo, T) = \mathbf{Info}(T) - \mathbf{Info}(Tempo, T) = 0.94 - 0.694 = 0.246$$

Se invece si fosse usato l'attributo *Ventoso*, si sarebbe trovato che **Info**(*Ventoso*,T)=0.892 e **Gain**(*Ventoso*,T)=0.048. Così l'attributo *Tempo* offre un maggior guadagno di informazione di *Ventoso*.

La precedente nozione di guadagno nell'informazione viene usata per stabilira una graduatoria fra attributi e costruire alberi di decisione dove in un nuovo nodo sia posto l'attributo con maggior guadagno di informazione fra gli attributi non ancora presi in considerazione nel cammino che separa dalla radice.

2.2.2 L'algoritmo ID3

L'algoritmo ID3 usa la precedente nozione di attributo non target con maggior guadagno di informazione per stabilire un ordinamento fra gli attributi non target non ancora presi in considerazione [Winston92]. Lo scopo di questo ordinamento è duplice: creare piccoli alberi di decisione così che le ennuple possano essere classificate solo dopo aver visionato il valore di pochi attributi, cercare di individuare l'albero di costo minimo nel processo di classificazione. Nel caso in analisi, l'albero costruito risulta quello mostrato nella figura che segue.



La nozione di guadagno informativo introdotta precedentemente tende a favorire attributi che hanno un gran numero di valori. Se, ad esempio, un attributo *att* assume un valore distinto per ogni ennupla – è una chiave –, allora **Info**(*att*, T)=0, e **Gain**(*att*,T) è massimo. Per compensare questo aspetto, è stato suggerito di utilizzare il seguente rapporto al posto di **Gain** [Quinlan87]:

$$\textbf{GainRatio}(Att, T) = \textbf{Gain}(Att, T) \,/\, \textbf{SplitInfo}(Att, T)$$

dove **SplitInfo**(*Att*, T) è l'informazione dovuta alla suddivisione di T sulla base dei valori dell'attributo target *Att*.. Quindi, $\textbf{SplitInfo}(Att, T) = \textbf{I}(|T_1| / |T|, |T_2| / |T|, \ldots, |T_m| / |T|)$ dove $\{T_1, T_2, \ldots, T_m\}$ è la partizione di T indotta dai valori di *Att*.

Nell'esempio, facendo gli opportuni calcoli risulterà vantaggioso utilizzare l'attributo *Ventoso* come primo attributo al posto di *Tempo*.

2.2.2 L'algoritmo C4.5

C4.5 introduce un numero sostanziale di estensioni all'algoritmo ID3 [Quinlan93]. Nel costruire un albero di decisione, è possibile trattare con insiemi di training che hanno ennuple con valori di attributi non specificati (unknown), valutando il guadagno, o il rapporto di guadagno, per l'attributo considerando solo quelle ennuple dell'insieme di training in cui il valore dell'attributo è definito. Nell'usare un albero di decisione è possibile classificare le ennuple stimando la probabilità dei possibili risultati. Infine è possibile utilizzare domini continui sfruttando le sole informazioni disponibili nell'insieme di training. Attraverso complesse e costose – in termini di calcolo – costruzioni combinatorie, si partiziona il dominio in intervalli di crescente dimensione fino a selezionare quella partizione che massimizza il guadagno, o il rapporto di guadagno.

L'albero di decisione viene costruito usando l'insieme di training, e quindi tratta in genere molto bene i casi che si presentano in questo insieme. Nel far ciò, l'albero risultante può diventare molto lungo e con cammini sbilanciati. La *potatura* dell'albero di decisione si rende quindi spesso necessaria. La potatura viene effettuata rimpiazzando un intero sottoalbero con una sua foglia. la sostituzione ha luogo se una regola di decisione stabilisce che il tasso di errore atteso in un sottoalbero è maggiore di quello in una singola foglia. [Winston92] illustra come usare il *test esatto di Fisher* per decidere se l'attributo *target* è veramente dipendente da un attributo non *target*, e, conseguentemente, eliminare il nodo dell'attributo non *target* dall'albero. [Quinlan87] e [Breiman84] suggeriscono

euristiche ancor più sofisticate per la potatura.

C5.0 è una versione rivista di C4.5 che, a detta dei fornitori, è meno soggetta a errori, è più veloce e consuma un minor quantitativo di memoria per il suo funzionamento. See5 e C5.0 sono due versioni dello stesso prodotto su differente sistema operativo [See5].

3. L'ESPERIMENTO

3.1 Le sorgenti informative disponibili

I dati su cui si è costruito l'esperimento sono dati costituiti dalle dichiarazione dei redditi di soggetti che esplicano attività in uno specifico settore commerciale, integrati con i dati ricavati da altre fonti quale, ad esempio, i consumi elettrici e telefonici.

Senza entrare in dettagli non significativi ai fini di questo articolo, il dataset utilizzato consiste di 80643 soggetti (righe), ognuno dei quali presenta 165 attributi (colonne), tutti di tipo numerico, di cui molti continui e solo alcuni discreti. Di questo insieme di elementi solo 4103 sono stati soggetti ad un controllo per accertare la presenza di frode, il cui esito è stato archiviato in un altro dataset formato appunto da 4103 righe e 7 colonne: una di queste rappresenta il recupero ottenuto con il controllo (attributo *recupero*), ed è uguale a zero in assenza di frode.

3.2 L'obbiettivo e il modello dei costi

Attraverso la collaborazione con esperti del dominio in questione, è maturata la necessità di definire un modello di costi da includere nel modello predittivo. Nel dominio in analisi ogni controllo effettuato dall'Ente preposto comporta una certa spesa che è indipendente dalla quantità di evasione accertata. Risulta perciò importante non tanto individuare il soggetto evasore, che appare assai frequentemente nel dominio, quanto individuare il soggetto fortemente evasivo, quello su cui risulta più remunerativo effettuare un controllo. L'obbiettivo è stato perciò quello di costruire un classificatore capace di selezionare quell'insieme di soggetti su cui risulta fruttuoso fare un accertamento, cercando di massimizzare il recupero e minimizzare le spese.

Questo obbiettivo compare assai frequentemente nel dominio del marketing. Consideriamo il seguente esempio. Supponiamo di voler capire il profilo di un possibile acquirente in un servizio di vendita per corrispondenza di libri. Supponiamo di assegnare dei costi/profitti ai possibili comportamenti dei potenziali acquirenti di fronte ad un'offerta promozionale (a chiunque risponde sarà regalato un libro):

- se il cliente risponde acquistando i libri allora guadagniamo 30.000 lire;
- se il cliente non risponde allora si perdono 2000 lire (il costo della spedizione dell'offerta);
- se il cliente risponde affermando di non essere interessato all'acquisto, ma al regalo associato all'offerta, allora perdiamo 10.000 lire (ovvero: il costo della spedizione più il costo del libro oggetto dell'offerta).

La situazione è simile al caso che stiamo considerando. Nel nostro caso, è stato definito un attributo *costo_accertamento* in funzione di alcuni degli altri attributi di una ennupla. Questo attributo rappresenta una stima quanto più realistica del costo di un accertamento in base a dei fattori che indichino, per esempio, le dimensioni numeriche degli impiegati o del volume di affari o di consumi energetici in questione.

Siano $A_1(i)$, $A_2(i)$, ... , $A_k(i)$ gli attributi dell'ennupla i, che rappresenta un soggetto accertato. Sia $A_k(i)$ = *recupero(i)* l'attributo che rappresenta la quantità di evasione recuperata tramite un accertamento. Definiamo genericamente:

$$costo\_accertamento(i) = A_{k+1}(i) = f(A_1(i), A_2(i), \ldots, A_k(i))$$

L'attributo *recupero_effetivo* rappresenta il recupero ottenuto dal controllo al netto delle spese di accertamento:

$$recupero\_effettivo(i) = recupero(i) - costo\_accertamento\ (i)$$

$$A_{k+2}(i) = A_k(i) - A_{k+1}(i)$$

Questo attributo è chiaramente il *target* della nostra analisi: si vuol riuscire a discriminare i soggetti che hanno questo attributo maggiore di zero, da quelli che lo hanno minore.

Si noti come in questo modo il modello di costi non sia utilizzato solamente a posteriori per valutare la bontà degli esperimenti, ma sia bensì inserito nell'attributo *target* e quindi nel processo di apprendimento automatico. Crediamo che un tale approccio possa essere utilizzato nel dominio della fraud detection, ogniqualvolta non sia necessario individuare la frode in tempo reale, ma sia bensì richiesta un'analisi a posteriori in supporto alla pianificazione degli accertamenti.

3.2.1 Variabile target

Per riuscire a discriminare i soggetti che hanno un valore positivo per l'attributo *recupero_effettivo* da quelli che lo hanno negativo, si è deciso di utilizzare una classificazione binaria tramite alberi di decisione, come già detto precedentemente.

Per classificare le ennuple, occorre definire le due classi per l'attributo target della nostra analisi. Definiamo la *classe di recupero effettivo*:

$$c.r.e(i) = A_{k+3}(i) = \begin{cases} negativo & se\ A_{k+2}(i) = recupero\_effettivo(i) \leq 0 \\ positivo & se\ A_{k+2}(i) = recupero\_effettivo(i) > 0 \end{cases}$$

Quello che si vuole fare è sfruttare i dati a disposizione per costruire un sistema di predizione che per ogni nuovo soggetto predica la classe di appartenenza, intendendo ovviamente, successivamente, sottoporre a controllo solo i soggetti predetti positivi. Ossia si vuole costruire un classificatore che per ogni nuova ennupla i predica $c.r.e.(i) = A_{k+3}(i)$ in base a $A_1(i)$, $A_2(i)$, ... , $A_{k-1}(i)$ (ovviamente gli attributi $A_k(i)$, $A_{k+1}(i)$, $A_{k+2}(i)$ verranno esclusi dalla fase di apprendimento poiché l'attributo target $A_{k+3}(i)$ dipende da loro in modo diretto).

3.3 Preparazione dei dati

3.3.1 Trasformazione dei dati

Il primo problema riscontrato è tecnico, e cioè la necessità di interagire con basi di dati operative di dimensioni notevoli e di tipo gerarchico. L'estrazione di un dataset di tipo relazionale è avvenuto tramite delle procedure interpretative costruite ad hoc con la collaborazione di un esperto della base di dati in questione.

Questa è stata solo la prima fase nella preparazione dei dati. I dati grezzi, infatti, possono richiedere delle modifiche sostanziali per permettere di evidenziare informative adeguate, calcolare sommari effettivi e effettuare analisi non necessariamente complicate. In molti casi, questo si è tradotto nella necessità di cambiare le unità di misura e la scala dei valori di alcuni attributi. Alcune difficoltà sono nate dal fatto che i dati presentavano forti asimmetrie, raggruppamenti a livelli differenti e devianze molto grandi rispetto ai modelli di distribuzione definibili.

Quando si è ottenuto un dataset gestibile, si è operato un *join* con il dataset più piccolo relativo ai soli accertamenti effettuati. Si è così giunti ad un dataset di sole 4103 ennuple, ognuna delle quali con 172 attributi. Si sono poi aggiunte le colonne $A_{k+1}(i)$, $A_{k+2}(i)$, ed infine l'attributo target $A_{k+3}(i)$ come definiti nel paragrafo precedente.

3.3.2 Pulizia dei dati (rimozione righe)

Nella fase di pulizia dei dati si sono eliminate dal dataset le ennuple troppo rumorose: in particolare sono state rimosse, tramite delle query SQL, quelle ennuple che presentavano degli attributi troppo devianti dalla media ed in definitiva poco realistici, probabilmente dovuti ad errori nella fase di acquisizione dati.

Inoltre sono state rimosse le ennuple con un numero di attributi con valore nullo superiore ad una certa soglia, anche queste probabilmente dovute a delle difficoltà nella fase di acquisizione dei dati.

Al termine di queste fasi di pulizia dei dati, il dataset risultava composto da 3880 ennuple. Di queste, 3183 fanno parte della classe negativa di *recupero effettivo* ($A_{k+3}(i)$ = *negativo*), le restanti 697 rientrano nella classe positiva ($A_{k+3}(i)$ = *positivo*). Ciò dimostra che, grazie al modello dei costi, si potrà focalizzare l'attenzione su un sottoinsieme ristretto di elementi del dominio, scartando la gran parte di elementi costuita da soggetti evasori ma per piccoli importi.

3.4 Riduzione dei dati

3.4.1 Selezione degli attributi (rimozione colonne)

L'individuazione e l'estrazione degli attributi ritenuti più significativi è stato un processo lungo e importante, durante il quale ci si è avvalsi della collaborazione di vari esperti del dominio. Dai 175 attributi sono stati rimossi quelli ridondanti o strettamente correlati, riducendone il numero effettivo a 20. In particolare, come già accennato precedentemente, sono stati eliminati gli attributi $A_k(i)$, $A_{k+1}(i)$, $A_{k+2}(i)$ perché strettamente correlati con l'attributo target.

La riduzione del numero degli attributi ha permesso di rendere le dimensioni del dataset iniziale trattabili dagli strumenti di datamining selezionati con prestazioni adeguate.

3.4.2 Scelta degli insiemi di training e test

La giusta dimensione del training set, ossia dell'insieme su cui si effettua l'allenamento del classificatore, è uno dei parametri più importanti in un esperimento di classificazione. Vi è una relazione fra la complessità delle associazioni che si vuole estrarre dai dati e il numero dei dati disponibili. Incrementando la dimensione del training set tipicamente aumenta anche la complessità del modello indotto e parallelamente diminuisce l'errore di allenamento (errore sulle ennuple del training set). Questo non significa che è sempre preferibile avere un training set quanto più grande possibile. Infatti una soluzione complessa, con un errore basso sul training set, può funzionare scorrettamente su nuove ennuple. Questo è il fenomeno dell'*over-fitting* sui dati: il classificatore più che imparare a classificare ha in realtà memorizzato perfettamente il comportamento da tenere sul training set e quindi si comporta bene sulle ennuple di allenamento e male sulle nuove ennuple (cattiva capacità di generalizzazione). Per evitare la trappola dell'*over-fitting* possiamo:

1. diminuire le dimensioni del training set,
2. aumentare il livello di pruning.

Ricordiamo comunque che una regola di validità generale utilizzabile anche in questo caso è il cosiddetto *Rasoio di Occam*: quando due soluzioni sono competitive è sempre preferibile la soluzione più semplice.

Nella sperimentazione, per individuare la giusta dimensione del training set si è utilizzato un approccio incrementale. Questo approccio consiste nell'allenare vari classificatori su sottoinsiemi del dataset scelti a caso e progressivamente più grandi. Un tipico modello può essere 10%, 20%, 33%, 50%, 66%, 90%, 100% del dataset disponibile. L'ultimo caso, quello in cui si utilizza l'intero dataset per l'allenamento, non è interessante in pratica, perché occorre lasciare sempre una porzione di dati per valutare le prestazioni del classificatore una volta completato l'allenamento. Questa porzione di dati è il cosiddetto test set.

Nel caso specifico, presto ci si è resi conto che i risultati erano migliori al crescere del training set. Questa ipotesi è stata confermata anche nella fase di aggiustamento del livello di pruning: applicando una tecnica incrementale, come quella sopra descritta, si ottengono risultati tanto migliori quanto più basso è il livello di pruning. In quest situazione si è quindi ben lontani dal raggiungere l'*over-fitting*: i dati a disposizione sono in realtà pochi rispetto alla complessità della conoscenza da estrarre.

In base a queste considerazioni le 3880 ennuple del dataset sono state suddivise nel seguente modo:

- training set: 3514 ennuple
- test set: 366 ennuple

3.5 Costruzione del modello

Ricordiamo che l'obiettivo dell'analisi è quello di costruire un classificatore che individui un insieme di ennuple su cui risulti "fruttuoso" un accertamento e che si utilizza una tecnica di classificazione binaria tramite alberi di decisione con target l'attributo *classe di recupero effettivo*. Gli alberi di

decisone sono quindi allenati a distinguere le ennuple in due classi: classe *negativa* (controlli non fruttuosi), classe *positiva* (controlli fruttuosi). Una volta terminata la fase di apprendimento, al classificatore viene dato in pasto il test set e si controlla se effettivamente riesce a distinguere le ennuple su cui risulta conveniente eseguire un controllo. In particolare, interessa non tanto all'errore di classificazione commesso da ogni albero (anche se questo rimane sempre un parametro importante) quanto il recupero al netto delle spese che si ottiene sottoponendo a controllo le ennuple classificate come positive. Questo valore è confrontato con il caso in cui si vadano a controllare tutte le 366 ennuple del test-set. Questo caso viene detto *Totale*:

- NumeroAccertamenti(*Totale*) = 366
- RecuperoEffettivo(*Totale*) = 309.058.283.000
- Spesa(*Totale*) = 48.220.000.000

Poichè si lavora solamente su ennuple che hanno subito un accertamento, è lecito ipotizzare che il caso *Totale* rappresenti il comportamento che sarebbe stato tenuto sulle 366 ennuple dai nostri committenti, tramite le tecniche attualmente in uso. Quindi, il confronto tra i risultati dei classificatori e il caso *Totale* rappresenta il miglioramento che le tecniche di datamining possono portare nel supporto al processo di pianificazione degli accertamenti.

I risultati degli esperimenti che seguono sono espressi mediante:

1. una percentuale di errore (percentuale di ennuple del test set misclassificate),
2. dei valori indicanti le spese e i recuperi degli accertamenti,
3. una *matrice di confusione* che riassume le predizioni sul test set del classificatore in questione.

Tale matrice ha la seguente forma:

negativi	positivi	← classificati come
TN	***FP***	classe effettiva negativa
FN	***TP***	classe effettiva positiva

Per ogni ennupla appartenente al training set, sia $pred_X(i)$ la classe di appartenenza predetta dal classificatore X per la ennupla i:

- ***TN*** = $\{i \mid pred_X(i) = c.r.e.(i) =$ negativa$\}$ sono le ennuple di classe negativa che vengono effettivamente classificate come tali: sono i soggetti negativi veri;
- ***FP*** = $\{i \mid pred_X(i) =$ positiva $\wedge\ c.r.e.(i) =$ negativa$\}$ sono le ennuple di classe negativa che vengono misclassificate di classe positiva. Queste ennuple subiranno degli accertamenti che saranno non fruttuosi ($recupero\_effettivo(i) \leq 0$) e costituiranno quindi una fonte di spesa inutile: sono i falsi positivi;
- ***FN*** = $\{i \mid pred_X(i) =$ negativa $\wedge\ c.r.e.(i) =$ positiva$\}$ sono le ennuple di classe positiva che vengono misclassificate di classe negativa. Queste ennuple non subiranno dei controlli anche se sarebbero in realtà fruttuosi. Sono i soggetti falsi negativi.

- ***TP*** = $\{i \mid pred_X(i) = c.r.e.(i) =$ positiva$\}$ sono le ennuple di classe positiva correttamente classificate come tali. Su queste ennuple si eseguono controlli fruttuosi. Sono i positivi veri.

L'errore di misclassificazione di ogni classificatore X è quindi dato dalla percentuale che ***FP+FN*** rappresenta sul numero totale di ennuple del test set. Questo valore verrà indicato come Errore(X).

Ai risultati di ogni classificatore X sono associati due indici:

$$\text{Redditività}(X) = \text{RecuperoEffettivo}(X)(\text{in miliardi}) / \text{NumeroAccertamenti}(X)$$

$$\text{Rilevanza}(X) = 10 * \text{Redditività}(X) / \text{Errore}(X)$$

3.5.1 Filosofie di sperimentazione

A questo punto sono possibili due diverse filosofie di esperimento. Da un lato, infatti, si può cercare di minimizzare il numero di ***FP***, in quanto sono la causa delle spese a vuoto; dall'altro, si può cercare di minimizzare i ***FN*** per ottenere un recupero maggiore.

Le due filosofie sono in contrasto. Infatti, se si minimizzano i ***FP***, si minimizza il numero di controlli a vuoto, ma inevitabilmente diminuiscono anche i ***TP*** (e quindi il numero di controlli totale) a favore dei ***FN*** che aumentano. In questo modo si ottiene anche un recupero minore, ma con spese e controlli minori. Se invece si cerca di minimizzare i ***FN***, si fanno più controlli e si recupera di più, ma si effettuano anche più controlli a vuoto.

In conclusione: la prima filosofia è migliore nei casi in cui si abbiano risorse limitate, la seconda nei casi in cui sono a disposizione infinite risorse per ottenere il massimo recupero. In ogni caso è possibile adattare i classificatori in base alle risorse messe a disposizione.

3.5.2 Tuning parametri: ovvero come minimizzare *FN* o *FP*

Una volta fissate le dimensioni del training set le variabili che restano da aggiustare sono:

- il livello di pruning,
- i pesi di misclassificazione.

Il livello di pruning, per le considerazione fatte precedentemente, è stato mantenuto molto basso in tutti gli esperimenti. Più precisamente gli alberi ottenuti dopo il pruning sono grandi almeno il 90% degli alberi originari.

I pesi di misclassificazione, invece, cambiano da esperimento ad esperimento: se si cerca di minimizzare i ***FP*** si fa pesare maggiormente questo tipo di errore, altrimenti si fa pesare di più gli errori di tipo ***FN***.

Un'altra tecnica utilizzata è la replicazione della classe di minoranza nel training set. La classificazione, infatti, è tipicamente sbilanciata sulla classe che è in maggioranza nel training set. Nel nostro caso specifico la classe di maggioranza è la *c.r.e. = negativa*. Se si vuol minimizzare i ***FN***, occorre spostare la classificazione sulla classe positiva, quindi si replicano le ennuple appartenenti a questa classe un numero di volte sufficiente a raggiungere una ripartizione almeno equilibrata delle due classi nel training set. Se invece si vuol minimizzare i ***FP***, si può lasciare il training set così com'è in quanto la classificazione è già sbilanciata sulla classe negativa.

Infine, si adotterà anche l'algoritmo di *boosting adattivo* [Freund95]. L'idea del boosting è quella di generare un certo numero di classificatori invece di uno solo. Inizialmente si genera un normale classificatore. Questo classificatore commetterà un certo numero di errori. Quando viene costruito il secondo classificatore viene posta particolare attenzione a quei casi su cui il primo classificatore commetteva gli errori. Ciò comporta che il secondo classificatore sia diverso dal primo, e tipicamente commetterà anche un numero di errori maggiore. Questi errori saranno al centro dell'attenzione del terzo classificatore. Questo processo continua per un numero predeterminato di iterazioni. Quando una nuova ennupla viene presenta, ogni classificatore vota per la sua classe di appartenenza e la ennupla verrà classificata secondo la maggioranza. I voti sono pesati rispetto alla bontà dei classificatori. Questa tecnica comporta una buona riduzione degli errori di classificazione.

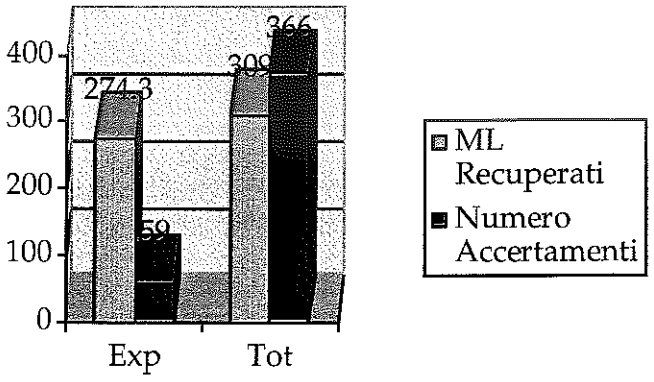
3.6 Valutazione del modello

Presentiamo adesso quattro classificatori, i primi due cercano di minimizzare i ***FP*** mentre gli altri due cercano di minimizzare i ***FN***.

3.6.1 Classificatore *A*

Questo esperimento utilizza per l'apprendimento il training set originale con 3514 ennuple. Utilizzando il training-set originale, senza alcuna replicazione di ennuple, la classificazione si sbilancia sulla classe negativa (essendo questa la classe di maggioranza nel training set), così da minimizzare i ***FP*** senza ricorrere a pesi di misclassificazione diversi. Si utilizza inoltre una tecnica di boosting a 10 alberi. In questo modo si ottengono 10 alberi, la loro votazione fornisce la classificazione mostrata dalla seguente matrice di confusione:

negativi	positivi	← classificati come
237	11	classe effettiva negativa
70	48	classe effettiva positiva



Si commettono 81 errori (22,1%), e se si ipotizza di controllare le 59 ennuple predette di classe positiva, si eseguiranno solamente 11 controlli a vuoto, riportando i seguenti risultati:

- NumeroAccertamenti(*A*) = 59

- RecuperoEffettivo(A) = 274.383.972.000
- Spese(A) = 7.870.000.000
- Redditività(A) = 4,649
- Rilevanza(A) = 2,1

Si osservi la redditività: si hanno 274 miliardi in 59 controlli, ben 4,649 miliardi per controllo. Si osservi come con un numero molto più basso di controlli si riesca ugualmente a recuperare circa l'88% del recupero nel caso *Totale*.

Nel grafico sono messi a confronto il recupero e il numero di accertamenti di questo esperimento con il caso *Totale*.

3.6.2 Classificatore *B*

Questo esperimento porta al limite la filosofia di minimizzazione dei ***FP***, utilizzando, oltre al dataset originale anche dei pesi di misclassificazione diversi. In particolare, si fa pesare gli errori di tipo ***FP*** due volte gli errori di tipo ***FN***.

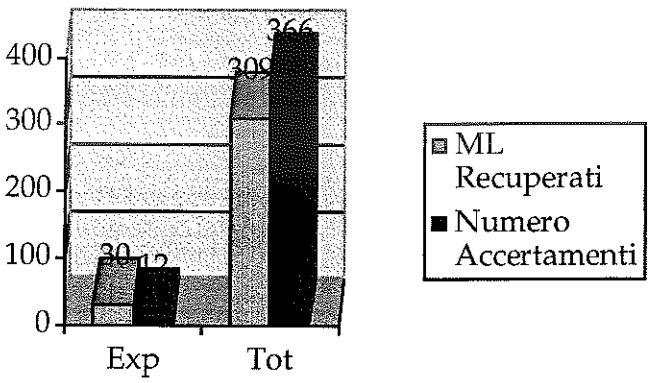
- Peso ***FP***: 2
- Peso ***FN***: 1

Inoltre si utilizza una tecnica di boosting a 3 alberi. Questa è la classificazione ottenuta:

negativi	positivi	← classificati come
246	***2***	classe effettiva negativa
108	***10***	classe effettiva positiva

Si commettono 110 errori (30,1%), riportando i seguenti risultati:

- NumeroAccertamenti(B) = 12
- RecuperoEffettivo(B) = 30.088.341.000
- Spese(B) = 2.160.000.000
- Redditività(B) = 2,5
- Rilevanza(B) = 0,83



Un recupero così modesto è dovuto al fatto che facendo solamente 12 controlli si perdono alcuni evasori ad altissimo recupero, che costituiscono la fetta maggiore del recupero totale. Questo esperimento, pur non ottenendo risultati eclatanti, può essere tenuto in considerazione allorchè si

vuole eseguire un numero molto piccolo di controlli.

Il prossimo grafico mette a confronto il recupero e il numero di accertamenti di questo esperimento con il caso *Totale*.

3.6.3 Classificatore *C*

Questo classificatore cerca di minimizzare i ***FN*** con l'obbiettivo di spostare la classificazione sulla seconda classe di recupero effettivo. A tal fine si utilizza un training set replicato: la porzione di ennuple di classe positiva viene replicata in modo da ottenere una distribuzione uniforme (50:50) tra le due classi. Il test set rimane invariato. Si utilizza un boosting a 3 alberi e i seguenti pesi di misclassificazione:

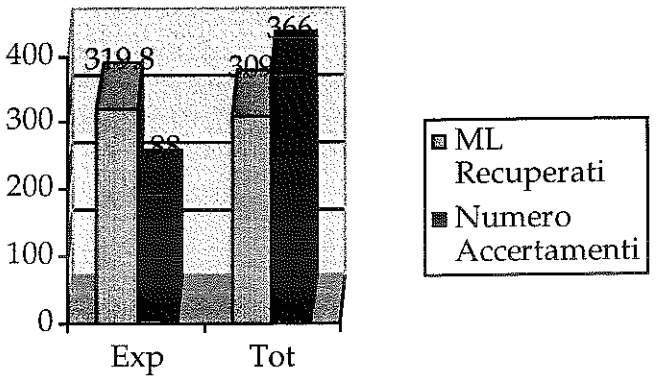
- Peso ***FP***: 1
- Peso ***FN***: 3

La classificazione risultante è la seguente:

negativi	positivi	← classificati come
150	98	classe effettiva negativa
28	90	classe effettiva positiva

L'albero commette 126 errori sul test set (34,4%), ma ottiene ugualmente un ottimo recupero:

- NumeroAccertamenti(*C*) = 188
- RecuperoEffettivo(*C*) = 319.862.135.000
- Spese(*C*) = 25.040.000.000
- Redditività(*C*) = 1,701
- Rilevanza(*C*) = 0,49



Il recupero al netto delle spese è addirittura più grande del recupero generale, facendo circa la metà dei controlli. Questo è dovuto ai 150 soggetti che giustamente vengono classificati come non fruttuosi (***TN***) e sui quali vi è un risparmio non essendo sottoposti a controlli inutili. La redditività è comunque assai minore di quella ottenuta negli esperimenti dell'altra filosofia.

3.6.4 Classificatore *D*

In questo esperimento si tenta di spostare ulteriormente la classificazione sulla classe positiva. Si

utilizza il training set replicato e i seguenti pesi di misclassificazione:

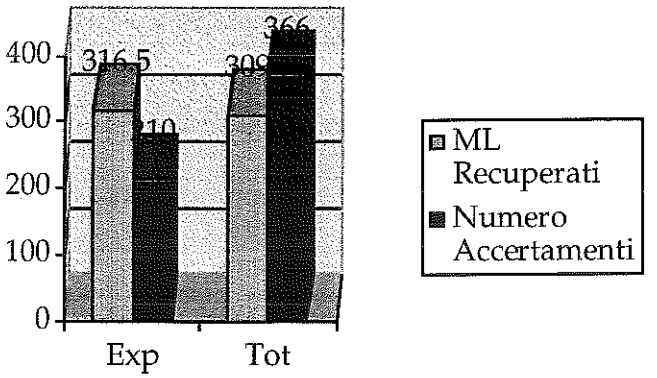
- Peso ***FP***: 1.
- Peso ***FN***: 4.

Si ottengono i seguenti risultati:

negativi	positivi	← classificati come
135	113	classe effettiva negativa
21	97	classe effettiva positiva

L'albero commette 134 errori sul test-set (36,6%). Rispetto al classificatore precedente si è minimizzato ulteriormente i ***FN*** (21 adesso contro i 28 precedenti), quindi meno soggetti fraudolenti vanno persi. Nonostante ciò il recupero al netto delle spese è peggiorato:

- NumeroAccertamenti(D) = 210
- RecuperoEffettivo(D) = 316.583.392.000
- Spese(D) = 28.040.000.000
- Redditività(D) = 1,507
- Rilevanza(D) = 0,41



Con questo esperimento, in cui si è aumentato il peso dei ***FN*** rispetto al classificatore precedente, si effettuano un numero maggiore di controlli (210 contro 188), ma solamente 7 controlli in più fruttuosi. I 15 controlli non fruttuosi in più comportano spese che fanno abbassare il recupero effettivo.

3.6.5 Esperimenti composti

Si cerca ora di comporre le predizioni dei vari alberi generati. Per esempio, si può mettere in *and* le predizioni di due classificatori, considerando fraudolenti solamente quei soggetti che vengono predetti di classe positiva da entrambi i classificatori. Oppure si può metter in *or* le predizioni, considerando fraudolenti i soggetti che vengono predetti di classe negativa da almeno uno degli alberi. Questi sono i risultati ottenuti:

- NumeroAccertamenti($A \wedge C$) = 58
- RecuperoEffettivo($A \wedge C$) = 274.497.148.000
- Spese($A \wedge C$) = 7.710.000.000
- Redditività($A \wedge C$) = 4,73

- NumeroAccertamenti($A \vee C$) = 189
- RecuperoEffettivo($A \vee C$) = 319.748.959.000
- Spese($A \vee C$) = 25.200.000.000
- Redditività($A \vee C$) = 1,691

Chiaramente nel primo caso si segue la filosofia di minimizzazione dei ***FP***, nel secondo caso quella di minimizzazione dei ***FN***. Come gia' notato la prima filosofia risulta vincente. Si compongono quindi ulteriormente il classificatore $A \wedge C$ mettendolo prima in *and* e poi in *or* con il classificatore *E:*

- NumeroAccertamenti($A \wedge C \wedge E$) = 43
- RecuperoEffettivo($A \wedge C \wedge E$) = 108.434.046.000
- Spese($A \wedge C \wedge E$) = 6.280.000.000
- Redditività($A \wedge C \wedge E$) = 2,52

- NumeroAccertamenti($(A \wedge C) \vee E$) = 80
- RecuperoEffettivo($(A \wedge C) \vee E$) = 278.982.168.000
- Spese($(A \wedge C) \vee E$) = 10.100.000.000
- Redditività($(A \wedge C) \vee E$) = 3,48

Un altro modo di comporre i classificatori è la votazione. Nel prossimo esperimento votano i classificatori *A*, *B*, *C* e *D*. I casi di pareggio vengono arbitrati dal classificatore *E*:

- NumeroAccertamenti(*E_arbitra(A,B,C,D)*) = 80
- RecuperoEffettivo(*E_arbitra(A,B,C,D)*) = 281.534.705.000
- Spese(*E_arbitra(A,B,C,D)*) = 10.310.000.000
- Redditività(*E_arbitra(A,B,C,D)*) = 3,51

Infine nell'ultimo esempio si considerano fraudolenti solamente i soggetti identificati fraudolenti da almeno 3 classificatori su 4:

- NumeroAccertamenti(*Almeno3(A,B,C,D)*) = 61
- RecuperoEffettivo(*Almeno3(A,B,C,D)*) = 278.456.523.000
- Spese(*Almeno3(A,B,C,D)*) = 8.230.000.000
- Redditività(*Almeno3(A,B,C,D)*) = 4,56

4. CONCLUSIONI.

Al di questa fase di esperimentazione, possono essere tracciate delle valutazioni conclusive analizzando l'adeguatezza della tecnologia data mining e delle relative metodologie per trattare problemi di questo dominio, alla luce della disponibilità di

ambienti di sviluppo specifici.

Per quanto riguarda la tecnologia datamining, possiamo da un lato affermare che essa è sicuramente consolidata, in quanto si basa su algoritmi progettati 30 anni addietro e che ora riescono a trovare il supporto computazionale adeguato per essere utilizzati, e dall'altro constatare che però non è semplice l'uso di tale tecnologia poichè mancano metodologie di sviluppo supportate da calcolatore per guidare l'intero processo di KDD.

Metodologie attualmente non sono disponibili, nel senso che esistono alcune indicazioni metodologiche molto scarne, che vanno istanziate e raffinate sul dominio specifico. La progettazione di una analisi è strettamente dipendente dal dominio: quali attributi consentono di discriminare tra comportamento legittimo e fraudolento, come deve essere caratterizzato il profilo dell'utente, come devono essere rilevate le anomalie, quali sono le condizioni di eccezionalità sono tutti problemi da risolvere .

Tutto ciò pone la necessità di utilizzare un ambiente integrato per supportare varie metodologie di analisi. Un ambiente di sviluppo che contenga sia un kit di strumenti di analisi ma anche uno strato di software che permetta la composizione dei vari strumenti. Infatti, si può in generale pensare che gli strumenti di Data Mining siano componibili ed integrabili per effettuare analisi anche molto complesse. Procedendo in tal senso si auspica un aumento delle capacità di analisi di vari ordini di grandezza. In questa prospettiva un ambiente deduttivo di seconda generazione come LDL++ sembra essere un buon candidato [Arni93].

5. RIFERIMENTI

[Arni93] N.Arni, K.Ong, S.Tsur, C.Zaniolo. "LDL++: A Second Generation Deductive Databases System", *Technical Report, MCC Corporation*, 1993

[Alesina96] Alesina, A. and M. Marè, "Evasione e Debito", in La finanza pubblica italiana dopo la svolta del 1992, Monorchio A. (ed.), Il Mulino, Bologna, Italy, 1996, pp. 69-112 (in italian).

[Berry97] Berry, M. and G. Linoff, *Data Mining Techniques for Marketing, Sales and Customer Support*, Wiley Computer Publishing, New York, USA, 1997.

[Breiman84] Breiman, Friedman, olshen, Stone, *Classification and decision trees*, Wadsworth, 1984.

[Freund95] Freund, Y. "Boosting a Weak Learning Algorithm by Majority", *Information and Computation , 121(2)*, pp. 256-285, 1995.

[KSEEKER] http://www.angoss.com/

[Quinlan87] Quinlan, J. R. "Simplifying decision trees", *International Journal of Man-Machine Studies, No 27*, pp. 221-234, 1987.

[Quinlan93] Quinlan, J. R., *C4.5: Programs for Machine Learning*, Morgan Kauffman, 1993.

[See5] http://www.rulequest.com/

[Tanzi93] Tanzi, V. and P. Shome, "A Primer on Tax Evasion", in *IMF Staff Papers, No 4*, 1993.

[Winston92] Winston P. H., *Artificial Intelligence*, 3rd Edition, Addison Wesley, 1992.