PAROLE CHIAVE:

Weka

Data Mining

Graingenes

Cerealab

INDICE

INDICE DELLE FIGURE

INDICE DELLE TABELLE

Introduzione

Con il termine Data Mining s'intende il processo di selezione, esplorazione e modellazione di grandi masse di dati al fine di scoprire regolarità o relazioni non note a priori e, allo scopo, di ottenere un risultato chiaro e utile a chi effettua analisi dei dati per scopi diversi: ricerca, previsione, statistica, ecc...
Un aspetto critico riguardo al data mining in genere è l'utilizzo di tecniche efficienti per estrarre conoscenza da un gran numero di dati sono complessi da rappresentare e trattare. I grandi progressi nel campo dell'hardware, della tecnologia dei database, della grafica e delle applicazioni open-source durante gli ultimi decenni rendono possibile la nascita di sistemi potenzialmente in grado di trattare grandi quantità d'informazioni complesse e multidimensionali presenti in svariate forme e tipologie a seconda del campo in cui vengono utilizzate: il range di applicazione è al giorno d'oggi molto vasto e in continua espansione. Letteralmente data mining significa estrazione di dati ed è a volte usato come sinonimo di "Knowledge Discovery in Database" (KDD, scoperta della conoscenza dei dati contenuti in un database). In realtà il termine KDD indica tutto il processo di scoperta della conoscenza mentre il termine Data Mining andrebbe usato quando si parla dell'applicazione ad alto livello di particolari metodi per l'estrazione dei dati.
Il processo di Knowledge Discovery, in sintesi, consiste in una sequenza di passi:

- "Data Cleaning", che tenta di ridurre i dati sbagliati, incompleti o rumorosi.
  Nella raccolta dati, infatti possono presentarsi valori inconsistenti causa la violazione di alcuni vincoli di integrità;
- "Data Integration", che integra le differenti sorgenti di dati in uno solo.
  I dati devono essere integrati risolvendo le possibili inconsistenze ed eliminando le ridondanze;

- “Data Selection”, che estrae dai dati sorgente solo dati rilevanti per le analisi;
- “Data Transformation”, che trasforma o consolida i dati in forme più appropriate per il mining;
- “Data Mining”, che identifica e caratterizza relazioni tra insiemi di dati senza richiedere necessariamente che l’utente ponga delle domande precise. Questa fase è quella in cui si opera l’applicazione di algoritmi specifici per estrarre modelli significativi dai dati;
- “Pattern Evaluation”, che identifica i patterns che rappresentano la conoscenza;
- “Knowledge Representation”, che mostra le tecniche di rappresentazione della conoscenza.

Attualmente un numero crescente di aziende ha sviluppato/adottato con successo applicazioni di data mining. Inizialmente, le prime ad adottare questa tecnologia furono principalmente le aziende operanti in settori come i servizi finanziari e direct mail marketing, oggi questa tecnologia è praticamente applicabile ad ogni azienda che intenda trarre vantaggi competitivi dal suo patrimonio informativo. Il tipo di lavoro svolto utilizza tecniche di Data Mining che, in minima parte, sconfinano in qualche aspetto della Knowledge Discovery, ovvero è presente una minima preparazione dei dati che precede effettiva l'elaborazione.

Infine, condizione necessaria per applicare con efficacia le tecniche di Data Mining è quella di poter accedere ad un vasta quantità di dati accumulati nel tempo (l’anno è l'ordine di grandezza più frequente) che possano essere richiamati efficacemente ad esempio tramite query formulate in un Dbms relazionale, oppure da database presenti sul Web utilizzando le funzionalità create da chi gestisce l'archivio on line. I dati utilizzati in questa tesi appartengono a due database che raccolgono le caratteristiche genetiche (genotipiche e fenotipiche) dei cereali come orzo, grano duro, grano tenero e avena.

Gli algoritmi di data mining sono stati applicati utilizzando Weka, un software open-source scritto interamente in linguaggio Java e sviluppato dal Progetto Pentaho. Weka fa parte di una suite di software open-source dedicati alla business

intelligence, essi sono liberamente scaricabili e dal sito ufficiale e permettono un ampio grado di personalizzazione per l'utente finale o per chi decide di modificarne il codice sorgente.

Il primo capitolo presenta una panoramica delle due basi di dati che forniscono i dati per l'elaborazione, nel secondo capitolo viene descritto il Progetto Pentaho e l'applicativo Weka per il data mining. Il terzo capitolo si dedica all'utilizzo di Weka tramite clustering sui dati del database fornito da C.R.A., il quarto capitolo utilizza in modo più approfondito gli algoritmi di data mining sul database Graingenes.

1. Il progetto Pentaho

Con il termine Business Intelligence si suole indicare l'insieme dei processi, delle tecniche e degli strumenti basati sulla tecnologia dell'informazione, che supportano i processi decisionali di carattere economico. Il problema fondamentale nel Business Intelligence è quello di disporre di sufficienti informazioni in modo tempestivo e fruibile e di analizzarle cosicché da poter avere un impatto positivo sulle strategie, le tattiche e le operazioni aziendali. Le informazioni possono riguardare la specifica impresa oppure situazioni più generali di mercato, o ancora la concorrenza (competitive intelligence).

1.1 Pentaho Business Intelligence

Il progetto Pentaho nasce dall'iniziativa della comunità open source di creare una classe di applicazioni dedicate alla Business Intelligence da proporre come valida alternativa alle soluzioni proprietarie in termini di costi, caratteristiche, funzioni e benefici. È la prima e la più vasta piattaforma per il business intelligence attualmente esistente rilasciata in forma open source interamente basata su Web Services e fornisce un insieme completo di applicazioni dedicate per il reporting, analisi dei dati, dashboard, data mining e integrazione di dati in una suite integrata e di facile impiego, continuamente aggiornata e che può essere utilizzata in termini modulari. La piattaforma è scritta interamente in linguaggio Java ed è disponibile per i sistemi operativi Linux, Mac OS X, Solaris e Windows XP/ME.
La gestione e la promozione del progetto, sono affidate alla compagnia Pentaho Corporation, ad essa fanno capo i software sviluppati dalla comunità Open Source, i building blocks java provenienti dagli sviluppatori da integrare ai software esistenti o per nuovi prodotti, fornisce inoltre supporto tecnico, training e diversi altri servizi alle aziende che ne fanno richiesta.

1.1.1 Componenti del progetto Pentaho

Le diverse aree di applicazione, accennate in precedenza, vengono gestite da software open source ben distinti, essi partecipano alla suite integrandosi fra loro, ma sono sviluppati come progetti separati:

- *Pentaho*, suite open source di business intelligence che prende nome dal progetto. La piattaforma è composta da un framework che fornisce procedure per logging, auditing, security, scheduling, ETL, webservices, attribute repository e rules engine;

- *Mondrian*, open source OLAP server diventato parte della suite Pentaho nel mese di novembre 2005. E' scritto in java e supporta query SQL e MDX, XML per l'analisi dei dati (XMLA) e Java OLAP.

- *JFreeReport*, open source reporting engine ovvero una collezione di progetti di sviluppo di applicazioni open source focalizzati alla creazione, gestione e distribuzione efficiente di report che coinvolgano diversi tipi di informazioni

- *Kettle*, open source data integration (ETL) per il data warehouse. Consiste in un insieme di tool che permettono di manipolare le informazioni provenienti da diverse basi di dati in un ambiente grafico, impostando il lavoro tramite metadati;

- *Weka*, open source data mining, software scritto interamente in java ed entrato a far parte del progetto Pentaho nel mese di settembre 2006, è costituito da una collezione di funzioni e diversi algoritmi di data mining per l'analisi dei dati, il tutto utilizzabile attraverso un'interfaccia grafica semplice.

Infine, è presente la Pentaho Community che si prefigge lo scopo di riunire gli sviluppatori per condividere qualsiasi informazione sullo sviluppo delle

applicazioni coinvolte nell'intero progetto: nuovo codice sorgente disponibile, segnalazione errori nelle applicazioni esistenti, test per nuove versioni in procinto di essere pubblicate, oppure qualsiasi altra informazione pertinente al progetto.

1.2 WEKA Data Mining

Weka è un ambiente software sviluppato nell'università di Waikato in Nuova Zelanda, è open source e viene rilasciato con licenza GNU. Scritto completamente in Java, consiste in una collezione di algoritmi dedicati all'analisi dei dati tramite il data mining: data preprocessing, classificazione, regressione, clustering, regole di associazione e visualizzazione e algoritmi di machine learning. Il progetto è nato nel 1993, Weka inizialmente fu scritto in linguaggio C ma la prima versione per il pubblico fu presentata nel 1996. Nel 1997 si decise di riscriverlo interamente in Java e nel corso degli anni l'interfaccia grafica è stata migliorata e resa più accessibile per l'utente. Il 19 settembre 2006 Pentaho annuncia l'acquisizione di Weka e il software è stato reso scaricabile alla pagina http://www.pentaho.org/download/ e da http://www.sourceforge.net sotto licenza GNU, è stato infine creato anche un forum per favorire l'interazione tra tutti gli utilizzatori dell'applicazione all'indirizzo http://community.pentaho.org.

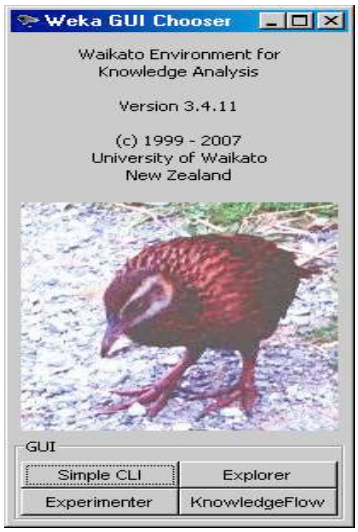

Figura 1 – Finestra di avvio di Weka

1.2.1 Modalità di utilizzo

Weka mette a disposizione all'avvio quattro diverse modalità di lavoro selezionabili con il mouse: Explorer, Knowledge Flow, Experimenter e Simple CLI; esse possono essere utilizzate contemporaneamente e gestiscono in modo differente l'approccio con cui si affronta il lavoro sui dati di interesse.

1.2.2 Interfaccia Explorer

Explorer è l'interfaccia grafica che permette in modo esaustivo di elaborare i dati e di visualizzarli con ampie possibilità di scelta. La figura 1 mostra la finestra che si presenta al primo avvio di Explorer. Le elaborazioni sui dati si impostano nelle sezioni rappresentate dalle voci in alto, esse sono:

- Preprocess: permette di caricare e modificare i dati da preparare per applicare elaborazioni varie
- Classify: permette di eseguire problemi riguardanti la classificazione di dati, regressione e valutarle
- Cluster: sezione dedicata all'applicazione degli algoritmi di clustering
- Associate: elaborazione e valutazione di regole di associazione
- Select attributes: tecniche di selezione degli aspetti più rilevanti di un insieme di dati
- Visualize: visualizzazione di diversi grafici bidimensionali sui dati in elaborazione

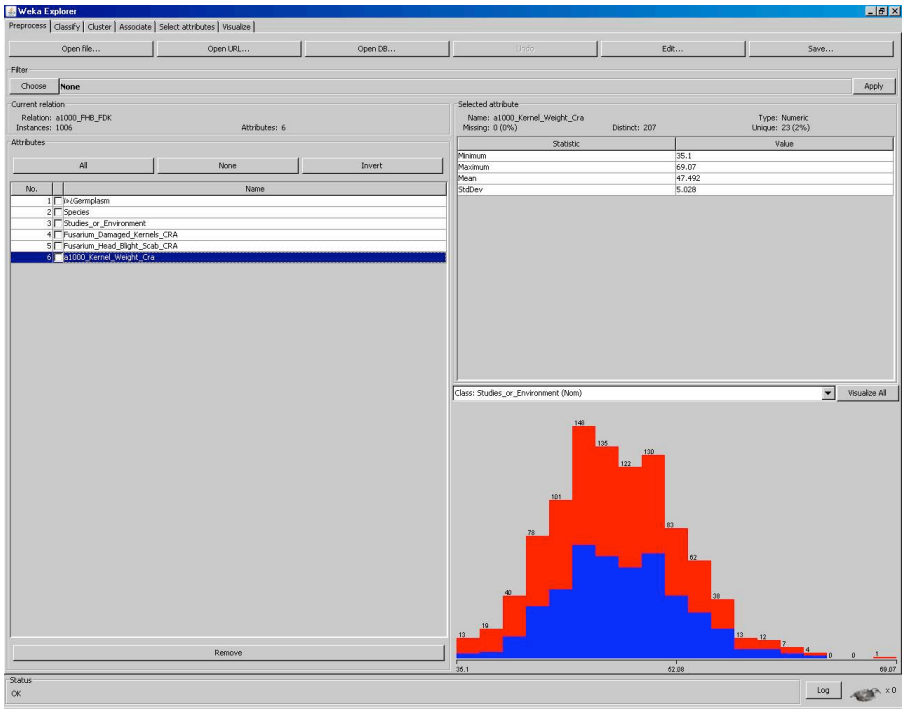

Figura 2 – Finestra principale di Explorer

I dati su cui effettuare le elaborazioni vengono caricati nella sezione Preprocess tramite i pulsanti in alto, le possibilità sono diverse: tramite file (preferibilmente in formato Csv o Arff), tramite URL o direttamente da un database utilizzando il driver Jdbc.

Una volta caricati i file, vengono mostrati tutti gli attributi e la quantità di tuple associate a tali attributi; selezionando il nome di un attributo numerico vengono mostrati il valore minimo, il massimo, la media e la deviazione standard riferita a tutti i valori presenti; viceversa si visualizza il numero di tuple associate all'etichetta in formato stringa. Tutti i valori possono essere modificati se ve ne è la necessità e possono essere nuovamente memorizzati in formato Arff, Csv o C45 in qualsiasi momento. Sotto a questi pulsanti, compare la voce Filter: essa permette di applicare filtri supervisionati e non supervisionati ai dati caricati, per esempio è possibile riordinare attributi, aggiungere attributi con valori generati casualmente, ricampionare valori esistenti o rimuovere quelli che raggiungono una certa soglia. Un grafico a barre, sobrio ma molto utile, riporta l'andamento dei valori per l'attributo scelto.

Nella parte bassa di ogni finestra di Explorer (e relativamente alla sezione in cui ci si trova) è presente uno status box, un bottone di log e un disegno rimpicciolito di

un animale weka: il primo mostra i messaggi su ciò che si sta elaborando, il secondo mostra, tramite doppio click con il mouse, informazioni sulle azioni che Weka ha eseguito nella sessione di lavoro corrente e il terzo si anima quando Weka è in attività e il numero di fianco al simbolo X indica quanti processi concorrenti sono in esecuzione nell'elaborazione corrente.

Le rimanenti sezioni Classify, Cluster, Associate e Select Attributes sono tutte analoghe in termini di utilizzo: si seleziona tramite il pulsante in alto uno specifico algoritmo, poi l'insieme dei dati da elaborare nell'area "Cluster mode", successivamente premendo sul pulsante Start si dà inizio all'elaborazione e l'output è mostrato nell'area a destra della figura 3; per ogni elaborazione si aggiorna la Result List che contiene tutte le elaborazioni fatte in ordine cronologico.

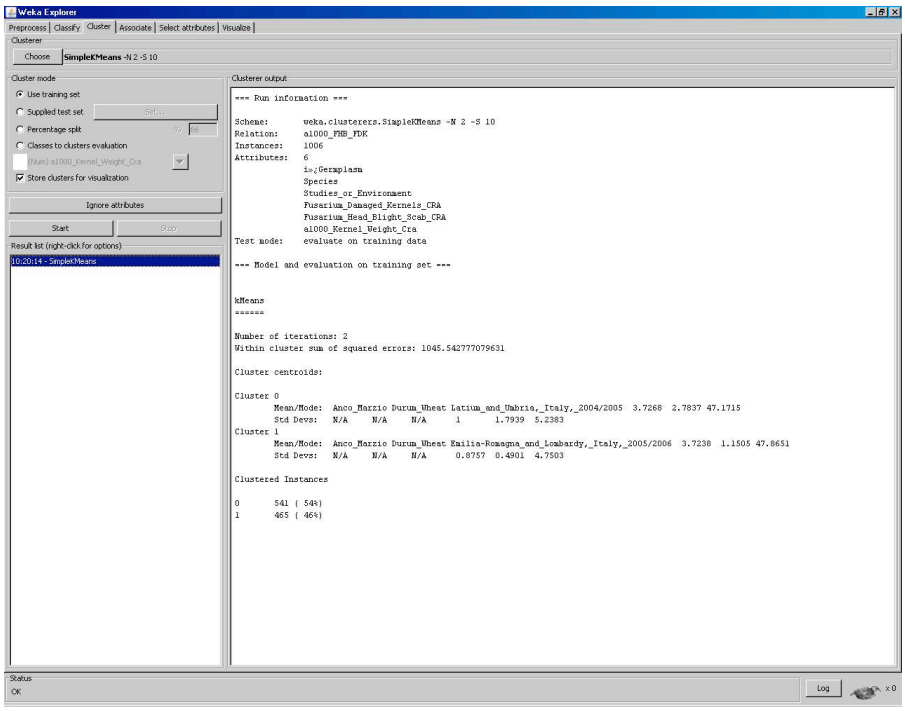

Figura 3 – Esempio di output in Explorer

Infine, la sezione Visualize permette di avere una visione d'insieme di tutti grafici disponibili ottenuti con tutte le combinazioni degli attributi caricati da file.

I grafici vengono visualizzati in piccolo come in un'anteprima, facendo doppio click con il mouse sul grafico scelto si apre una finestra separata che mostra il grafico nei dettagli: è possibile a questo punto intervenire cambiando gli attributi

degli assi X e Y rispetto a quelli scelti inizialmente, scegliere di colorare i punti in base ai valori di un terzo attributo, selezionare zone del grafico da poter vedere nei dettagli facendo una sorta di "zoom".

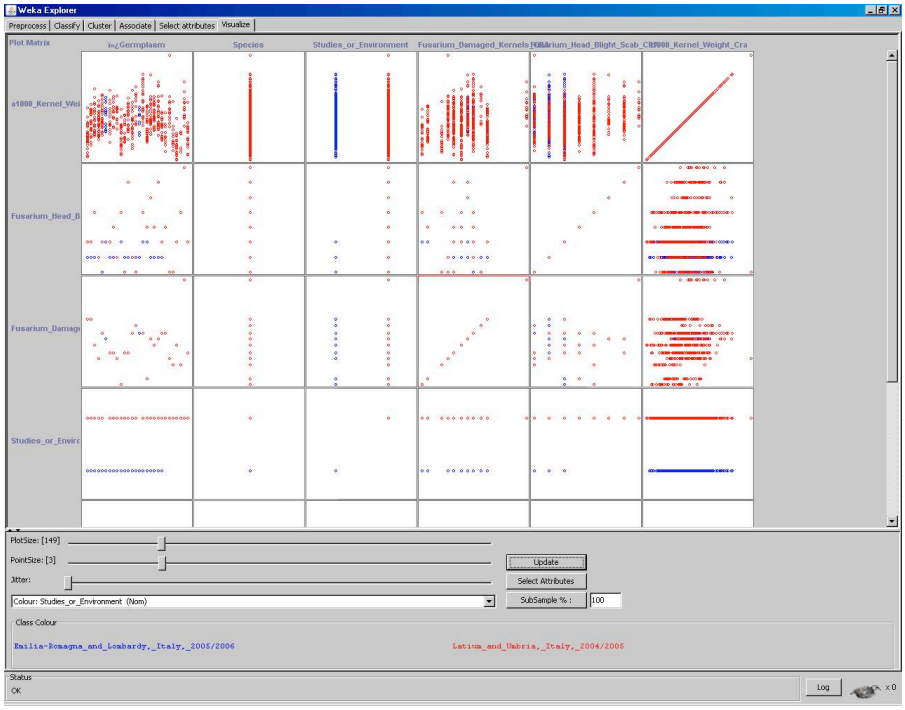

Figura 4 – Sezione Visualize di Explorer

1.2.3 Algoritmi implementati in Weka Explorer

Explorer è l'interfaccia che permette di accedere a tutti gli algoritmi presenti in Weka nell'intero processo di elaborazione dei dati, i filtri rappresentano la prima elaborazione per preparare i dati da elaborare e si impostano direttamente nella finestra Preprocess.

1.2.3.1 Filtraggio dei dati

Un filtro effettua elaborazioni più o meno complesse sui dati di ingresso (così come si presentano al momento del caricamento in Weka) in base alle scelte fatte dall'utente.

I filtri si distinguono in supervisionati e non supervisionati: nei supervisionati l'utente, attraverso i parametri impostati per l'algoritmo di filtraggio, imposta il

grado di elaborazione e modifica da compiere sui dati in ingresso, i non supervisionati invece effettuano una elaborazione automatica senza fare distinzione alcuna dei dati che si trovano ad elaborare.

All'interno di queste due categorie è possibile scegliere se applicare i filtri a livello di attributi o a livello di istanze: i primi operano su un singolo o più attributi selezionati, i secondi operano a livello di tuple prendendo in considerazione la totalità degli attributi. Una volta scelto il filtro, i parametri di configurazione si impostano in una finestra di dialogo che compare facendo doppio clic con il mouse sul nome del filtro stesso.

Nome	Descrizione sintetica
Add	Aggiunge un nuovo attributo a quelli esistenti con valori nulli
AddCluster	Aggiunge un attributo con una etichetta che rappresenta il cluster assegnato a ognuna tupla in base a un algoritmo di cluster scelto dall'utente
AddExpression	Crea un nuovo attributo con i valori risultanti da una funzione matematica basata sugli attributi già esistenti
AddNoise	Cambia una certa percentuale di valori aggiungendo rumore
ClusterMembership	Utilizza un algoritmo di clustering per generare valori appartenenti ai cluster trovati e che andranno a formare nuovi attributi
Copy	Copia un intervallo di attributi nel dataset
Discretize	Converte gli attributi numerici in etichette stringa
FirstOrder	Applica l'operatore di differenza del primo ordine su un intervallo di valori
MakeIndicator	Rimpiazza un attributo stringa con un attributo booleano
MergeTwoValues	Fonde due valori per un attributo specificato
NominalToBinary	Converte tutti gli attributi numerici dalla base dieci in base due.

Nome	Descrizione sintetica
NumerciTransform	Trasforma un attributo numerico utilizzando direttamente le funzioni Java
Obfuscate	"Offusca" il dataset rinominando le relazioni, tutti gli attributi e il loro tipo
PKIDiscretize	Discretizza attributi numerici
RandomProjection	Elabora i dati tramite un sottospazio con dimensioni minori di quello di partenza e genera una matrice di valori casuali
Remove	Rimuove gli attributi
RemoveType	Rimuove gli attributi di un tipo specifico
RemoveUseless	Rimuove attributi costanti, insieme ad attributi stringa che variano troppo
ReplaceMissingValues	Sostituisce i valori mancanti di attributi stringa o numerici con la moda e la media dei dati presenti
Standardize	Standardizza tutti gli attributi numerici in modo che abbiano media zero e varianza unitaria
StringToNominal	Converte un attributo stringa in etichetta
StringToWordVector	Converte un attributo stringa in un vettore che rappresenta la frequenza di parole
SwapValues	Scambia due valori di un attributo
TimeSeriesDelta	Sostituisce i valori di attributi nella istanza (tupla) corrente con la differenza tra il valore corrente e il valore predetto analizzando altre tuple
TimeSeriesTranslate	Sostituisce i valori di attributi nella tupla corrente con l'equivalente valore predetto analizzando altre tuple

Tabella 1 – Filtri non supervisionati per gli attributi

Nome	Descrizione sintetica
NonSparseToSparse	Converte le istanze in formato "sparse" ovvero con

Nome	Descrizione sintetica
	valori zero per gli attributi mancanti
Normalize	Considera gli attributi numerici come un vettore da normalizzare rispetto a una specifica lunghezza
Randomize	Mescola in modo casuale l'ordine delle tuple in un dataset
RemoveFolds	Riporta in output una specifica "fold" del dataset utilizzando la cross-validation
RemoveMissclassified	Rimuove le istanze classificate come incorrette in base a uno specifico classificatore
RemovePercentage	Rimuove una frazione del dataset espressa in percentuale
RemoveRange	Rimuove un determinato intervallo di istanze da un dataset
Resample	Produce un sottoinsieme di valori casuali del dataset originario
SparseToNonSparse	Converte tutte le istanze in formato "sparse" in formato "nonsparse"

Tabella 2 – Filtri non supervisionati per le istanze

Nome	Descrizione sintetica
AttributeSelection	Permette l'accesso alle funzioni di selezione di attributi così come nella sezione Select attributes di Explorer
ClassOrder	Randomizza o altera in altro modo l'ordine dei valori di una classe selezionata
Discretize	Converte gli attributi in formato binario, usando un metodo supervisionato se la classe è numerica

Tabella 3 – Filtri supervisionati per gli attributi

Nome	Descrizione sintetica
Resample	Produce un sottoinsieme di valori casuali per un dataset, sostituendo i valori originari del dataset
SpreadSubsample	Produce un sottoinsieme di valori casuali diffondendo i valori tra classi in base alla frequenza specificata
StratifiedRemoveFolds	Crea una cross-validation per il dataset da aggiungere ai dati originari

Tabella 4 – Filtri supervisionati per le istanze

1.2.3.2 Algoritmi di apprendimento

Si accede a questa tipologia di algoritmi nella sezione Classify di Explorer oppure in quella analoga di Knowledge Flow o in Experimenter. Esistono algoritmi di apprendimento supervisionato e non supervisionato dedicati alla classificazione dei dati nelle forme più varie: reti bayesiane, alberi di decisione, apprendimento di regole e funzioni matematiche per il calcolo di regressione, correlazione, eccetera. Alcuni algoritmi sono affiancati da versioni migliorate o leggermente modificate per aumentarne le prestazioni o estenderne le funzionalità.

	Nome	Funzione
Bayes	AODE	Averaged, one-dependence estimators
	BayesNet	Apprendimento di reti bayesiane
	ComplementNaiveBayes	Costruisce un classificatore bayesiano complementare
	NaiveBayes	Classificatore bayesiano probabilistico standard
	NaiveBayesMultinomial	Versione multinomiale del classificatore bayesiano
	NaiveBayesUpdateable	Classificatore bayesiano incrementale che apprende una istanza per volta
Alberi di	ADTree	Costruisce un albero di decisione di tipo

	Nome	Funzione
decisione		"alternating decision"
	DecisionStump	Costruisce un albero di decisione di primo livello
	ID3	Albero di decisione di tipo "divide-and-conquer" di base
	J48	Albero di decisione basato sull'algoritmo C4.5
	LMT	Costruisce alberi di decisione logistici
	M5P	Albero di apprendimento basato sull'algoritmo M5
	NBTree	Costruisce un albero di decisione basandosi sul classificatore bayesiano
	RandomForest	Costruisce un albero seguendo la procedura di "Random forest"
	RandomTree	Costruisce un albero basandosi su un dato numero di caratteristiche scelte casualmente
	REPTree	Albero con algoritmo di apprendimento veloce che usa il pruning
	UserClassifier	Permette all'utente di costruire un albero in base alle proprie scelte
Rules	ConjunctiveRule	Semplice algoritmo di apprendimento di regole
	DecisionTable	Costruisce una tabella di decisione semplice
	JRip	Algoritmo RIPPER per regole a induzione
	M5Rules	Ottiene regole basandosi su alberi di decisione di tipo M5P
	Nnge	Genera regole in base al metodo "nearest-neighbor"

	Nome	Funzione
	OneR	Classificatore 1R
	Part	Ottiene regole in base a porzioni di alberi creati con algoritmo J48
	Prism	Semplice algoritmo di copertura per generare regole
	Ridor	Algoritmo di apprendimento di tipo "Ripple-down"
	ZeroR	Predice i valori di maggior frequenza di una classe (se composta di etichette stringa) o il valore medio (se è composta da valori numerici)
Functions	LastMedSq	Regressione utilizzando la mediana invece della media
	LinearRegression	Regressione lineare standard
	Logistic	Crea modelli logistici lineari
	MultilayerPerceptron	Rete neurale a retropropagazione
	PaceRegression	Crea modelli di regressione lineare usando il metodo Pace
	RBFNetwork	Implementa una funzione di rete neurale radiale
	SimpleLinearRegression	Apprendimento tramite modello a regressione lineare basato su un singolo attributo
	SimpleLogistic	Calcola la regressione lineare logistica in base alla selezione di attributi
	SMO	Algoritmo di ottimizzazione sequenziale minima per classificazione fatta tramite vettori
	SMOreg	Algoritmo di ottimizzazione sequenziale minima con supporto per vettori a regressione

	Nome	Funzione
	VotedPerceptron	Algoritmo del Percettrone
	Winnow	Algoritmo del Percettrone guidato dagli errori
Lazy	IB1	Algoritmo di apprendimento di base basato sul metodo “nearest-neighbor”
	IBk	Classificatore k-nearest-neighbor
	KStar	Algoritmo di tipo nearest-neighbor con funzione di distanza
	LBR	Classificatore bayesiano di tipo lazy
	LWL	Algoritmo per l’apprendimento di dati valutati localmente tramite “peso”
Misc.	Hyperpipes	Algoritmo di apprendimento veloce e molto semplice basato su ipervolumi nello spazio delle istanze
	VFI	Algoritmo del metodo dei voti

Tabella 5 – Algoritmi di classificazione attualmente presenti in WEKA

1.2.3.3 Algoritmi di meta apprendimento

Gli algoritmi di meta apprendimento costituiscono un sottoinsieme degli algoritmi di apprendimento classici, essi si applicano ai meta dati ovvero alle informazioni che descrivono i dati stessi: si basano su un insieme di regole che descrivono la struttura dei dati e si applicano a un determinato caso noto a priori. Gli algoritmi presenti coprono diverse modalità elaborazione sui meta dati in modo piuttosto esaustivo.

Nome	Descrizione sintetica
AdaBoostM1	Algoritmo di boost che utiliza il metodo AdaBoostM1
AdditiveRegression	Aumenta la performance di un metodo di

Nome	Descrizione sintetica
	regressione riempiendo iterativamente i dati mancanti
AttributeSelectedClassifier	Riduce le dimensioni dei dati attraverso la selezione di attributi
Bagging	Classificatore di tipo bagging, funziona anche per la regressione
ClassificationViaRegression	Esegue la classificazione tramite un metodo di regressione
CostSensitiveClassifier	Classificatore in base al costo
CVParameterSelection	Seleziona parametri in base alla cross-validation
Decorate	Utilizza un insieme di classificatori basandosi su esempi costruiti artificialmente
FilteredClassifier	Esegue un classificatore sui dati filtrati
Grading	Algoritmo che accetta in input dati di previsione di baase che sono state precedentemente marcate come corrette o non corrette
LogitBoost	Esegue la regressione logistica additiva
MetaCost	Crea un classificatore sensibile al costo
MultiBoostAB	Combina il metodo di boosting e bagging usando il metodo Multiboosting
MultiClassClassifier	Usa un classificatore a due classi per i dataset a multiclasse
MultiScheme	Utilizza la cross-validation per selezionare un classificatore da diversi candidati
OrdinalClassClassifier	Applica gli algoritmi di classificazione standard a problemi con classi ordinate di valori
RacedIncrementalLogitBoost	Algoritmo di apprendimento funzionante sul principio dell'elaborazione batch
RandomCommittee	Crea un insieme di classificatore di base a caso
Stacking	Combina diversi classificatori usando il metodo stacking

Nome	Descrizione sintetica
StackingC	Versione più efficiente di Stacking
ThresholdSelector	Ottimizza le f-misure di un classificatore probabilistico
Vote	Combina diversi classificatori usando la media della probabilità stimata o le previsioni numeriche

Tabella 6 – Algoritmi di meta apprendimento presenti in WEKA

1.2.3.4 Algoritmi di clustering

Algoritmi di classificazione mediante clustering. Attualmente sono disponibili i metodi elencati in tabella, essi sono quelli stabili e perfettamente funzionanti. Nuovi metodi di clustering sono attualmente in corso di sviluppo e ancora non compaiono nelle versioni ufficiali di Weka.

Nome	Descrizione sintetica
EM	Clustering con algoritmo Expectation-Maximization
Cobweb	Clustering con algoritmo Cobweb e Classit
FarthestFirst	Clustering con algoritmo Farthest-first
SimpleKMeans	Clustering con algoritmo delle k-medie standard

Tabella 7 – Algoritmi di clustering attualmente presenti in Weka

1.2.3.5 Regole associative

Questi algoritmi, in base a parametri decisi dall'utente, cercano associazioni non note a priori di elementi che ricorrono frequentemente nell'insieme di dati di input, infine riportano in output le associazioni trovate mostrando il valore di confidenza e supporto calcolato per ogni regola associativa corrispondente

.

Nome	Descrizione sintetica
Apriori	Algoritmo Apriori per le regole
PredictiveApriori	Algoritmo Apriori che trova regole di associazione ordinate per accuratezza nella predizione
Tertius	Algoritmo a conferma guidata durante la scoperta di regole di associazione o classificazione

Tabella 8 – Algoritmi per regole di associazione attualmente presenti in Weka

1.2.3.6 Selezione e ricerca di attributi

Questi algoritmi eseguono una serie di operazioni che riguardano la ricerca di attributi in base a parametri impostati dall'utente e anche la valutazione su singoli attributi o alle tuple stesse prese nella loro interezza. Tramite output e valutazioni numeriche viene stimato il grado di importanza e di utilità dei dati di input, permettendo così di prendere decisioni sull'effettiva validità dei dati disponibili. Il lavoro può consistere nell'eliminazione di attributi interi o di singoli valori di uno o più colonne di una tabella, oppure si possono operare correzioni nei valori già esistenti in base a parametri specificati dall'utente e possono essere creati nuovi attributi da aggiungere a quelli originari in base alle esigenze dell'utente.

	Nome	Funzione
Valutazione di più attributi contemporaneamente	CfsSubsetEval	Considera il valore previsto di ogni attributo individualmente assieme al grado di ridondanza degli attributi stessi
	ClassifierSubsetEval	Usa un classificatore per valutare l'insieme di attributi
	ConsistencySubsetEval	Progetta il training set sul set di attributi e misura la

	Nome	Funzione
		consistenza in base ai valori assunti dalle classi
	WrapperSubsetEval	Usa un classificatore unito alla cross-validation
Valutazione di un singolo attributo	ChiSquaredAttributeEval	Calcola il valore di chi-quadro per ogni attributo rispetto alla classe
	GainRatioAttributeEval	Valuta un attributo in base al rapporto di guadagno
	InfoGainAttributeEval	Valuta un attributo inbase al guadagno di informazione
	OneRAttributeEval	Usa l'algoritmo OneR per valutare gli attributi
	PrincipalComponents	Analisi e trasformazione sui "principal components"
	ReliefAttributeEval	Valutazione attributi a livello di istanze
	SVMAttributeEval	Usa il metodo "support vector machine" per determinare il valore degli attributi
	SymmetricalUncertAttributeEval	Valuta un attributo in base alla incertezza simmetrica

Tabella 9 – Metodi per la selezione di attributi rilevanti presenti in Weka

	Nome	Funzione
Metodo di ricerca	BestFirst	Metodo di ricerca di tipo "greedy" con tracciatura all'indietro
	ExhaustiveSearch	Ricerca esaustiva

	Nome	Funzione
	GeneticSearch	Ricerca utilizzando un algoritmo di base di ricerca genetica
	GreedyStepwise	Algoritmo di ricerca di tipo "greedy" senza tracciatura all'indietro
	RaceSearch	Utilizza la metodologia della ricerca "race"
	RandomSearch	Cerca in modo casuale
	RankSearch	Ordina gli attributi e li classifica usanto un algoritmo di valutazione per subset di attributi
Metodo di classifica	Ranker	Classifica attributi singoli (non sottoinsiemi) in base alla loro rilevanza

Tabella 10 – Metodi ricerca per la selezione di attributi

1.2.4 Interfaccia Knowledge Flow

Un'alternativa a Explorer per elaborare i dati in Weka, consiste nell'utilizzare questa interfaccia che organizza ed elabora i dati impostati in maniera schematica e sintetica. Più in dettaglio l'utente seleziona un componente (rappresentato da un'icona) da una tool bar, lo posiziona nella finestra di lavoro e lo collega graficamente ad altri elementi già presenti nell'area di lavoro tramite frecce, ogni icona rappresenta una particolare elaborazione sui dati: apertura di file, salvataggio, applicazione di algoritmi di data mining e infine visualizzazione grafica. L'ordine con cui avvengono le operazioni viene stabilito al momento di collegare le frecce tra le icone, c'è sempre un'icona di partenza rappresentata dall'apertura di un file (arff o csv ad esempio) e da questa possono diramarsi una o più frecce che indicano una o più elaborazioni contemporanee di algoritmi.

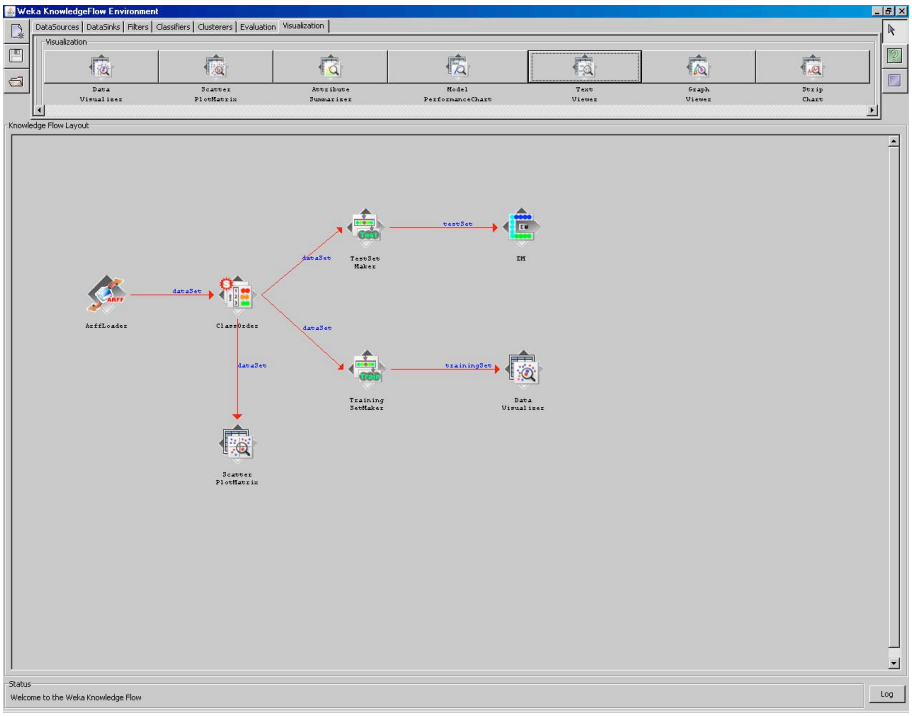

Figura 5 – Esempio di grafo di lavoro in Knowledge Flow

Questa modalità di lavoro permette quindi di rappresentare e successivamente di eseguire in termini di flow chart le stesse procedure che Explorer permette di fare, ma aggiunge un livello di descrizione del lavoro più chiaro e conciso. Ogni elemento dell'area di lavoro viene configurato individualmente tramite un menù che compare cliccando con il pulsante destro del mouse, tale menù ha tre voci: Edit, Connections e Actions. Con la voce Edit si cancellano i componenti o si apre la finestra di configurazione dell'elemento stesso. Gli algoritmi di classificazione o filtro si configurano come in Explorer, per il caricamento dati invece si sceglie un file da disco. La voce Actions comprende operazioni specifiche che riguardano il componente nell'area di lavoro che si vuole configurare al momento, infine tramite Connections si collegano i componenti tra di loro dall'icona sorgente e quella destinazione cliccando nei punti di connessione evidenziati da Weka. Diversamente da Explorer, i componenti per visualizzare e valutare i risultati mostrati nella tabella sono solo presenti in Knowledge Flow: vengono utilizzati prelevandoli dalla barra delle icone e connettendo con le frecce i flussi di dati interessati.

	Nome	Funzione
Visualization	*DataVisualizer*	Visualizza i dati in grafici a due dimensioni
	ScatterPlotMatrix	Visualizza il riepilogo di tutti i grafici
	AttributeSummarizer	Mostra istogrammi per ogni attributo
	ModelPerformanceChart	Disegna curve ROC e altre curve di soglia
	TextViewer	Visualizza i dati in formato testo
	GraphViewer	Visualizza i grafi ad albero
	StripChart	Mostra un grafico a scorrimento dei dati
Evalutation	*TrainingSetMaker*	Rende i dati di input come training set corrente
	TestSetMaker	Rende i dati di input come test set
	CrossValidationFoldMaker	Divide un dataset in fold
	TrainTestSplitMaker	Divide un dataset in training set e test set
	ClassAssigner	Imposta un attributo come classe di confronto
	ClassValuePicker	Sceglie un valore per la classe *positiva*
	ClassifierPerformanceEvaluator	Statistiche di valutazione sui risultati
	IncrementalClassifierEvaluator	Statistiche incrementali di valutazione sui risultati
	ClustererPerformanceEvaluator	Statistiche per il clustering
	PredictionAppender	Aggiunge al dataset i risultati delle predizioni di

Nome	Funzione
	un algoritmo classificatore

Tabella 11 – Componenti di visualizzazione e valutazione di Knowledge Flow

1.2.5 Interfaccia Experimenter

Questa interfaccia è dedicata alla effettiva "sperimentazione" di più algoritmi in serie che operano su una mole molto vasta di dati, pensata per quegli utenti Weka che richiedono elaborazioni complesse e articolate che necessitano di un tempo elevato di attesa, prima di fornire risultati, anche nell'ordine di diversi minuti. Si caricano più file corrispondenti a diversi insiemi di dati, si impostano gli algoritmi e le iterazioni necessarie e infine il file di output da creare con i risultati.

Una particolare e interessante caratteristica di Experimenter è quella di poter distribuire le elaborazioni su più processori ovvero su più postazioni che operano in parallelo, per fare ciò è necessario installare Java su tutti i computer, rendere l'accesso a tutti i dati che si stanno usando aperto a tutte le postazioni ed eseguire il processo server weka.experiment.RemoteEngine (tramite il file remoteExperimentServer.jar) su un computer che funge da host.

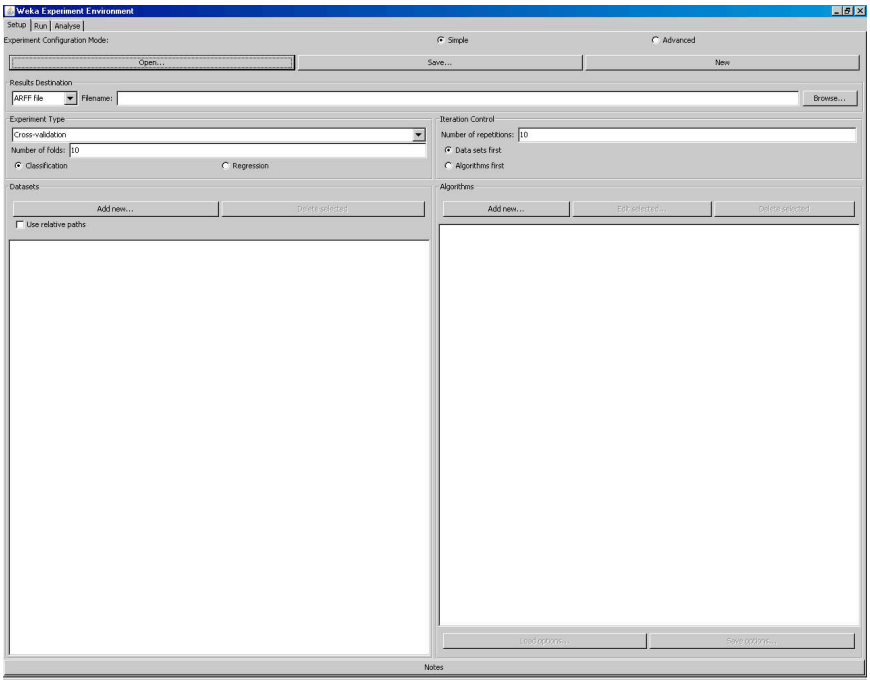

Figura 6 - Finestra principale di Experimenter

1.2.6 Interfaccia Simple CLI

Questa modalità permette, tramite una semplice shell, di utilizzare Weka con una interfaccia a linea di comando, l'utilizzo però presuppone la conoscenza della struttura di Weka in termini di classi, istanze e packages Java.

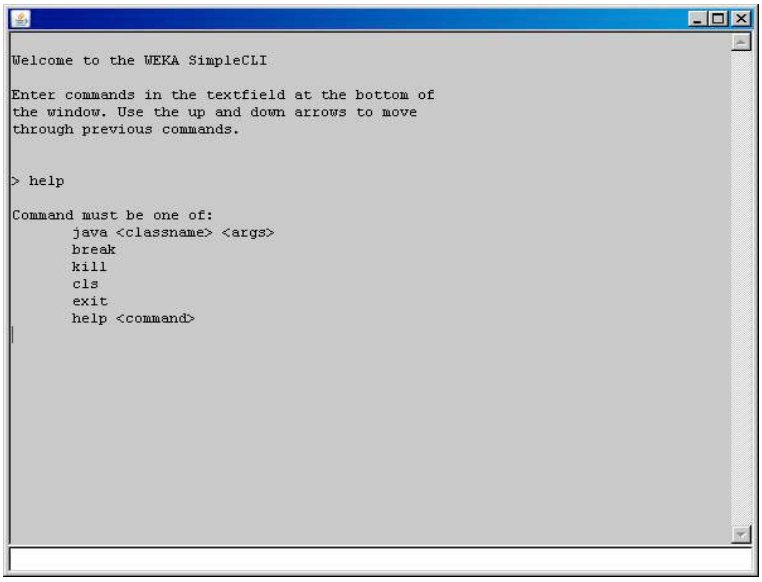

Figura 7 – Interfaccia Simple CLI

1.3 Il formato Arff

Un file Arff (Attribute-Relation File Format) è un file di testo in formato Ascii che descrive un insieme di tuple con valori e attributi, tale formato è stato sviluppato dal Machine Learning Project dell'Università di Waikato per essere utilizzato con il software Weka, è il formato di default utilizzato per caricare nuovi dati o salvare dati elaborati.

Un file Arff è composto da due sezioni, Header seguita da una sezione Data: l'Header contiene il nome della relazione, l'elenco degli attributi (ovvero le colonne di dati) e il loro formato; la sezione Data contiene l'elenco dei valori inseriti nello stesso ordine con cui sono stati definiti gli attributi nella sezione Header. Inoltre è possibile inserire commenti aggiungendo nella riga il simbolo "%".

1.3.1 Sezione Header

Il nome della relazione viene definito nella prima riga di un file Arff, il formato è @RELATION <nome-relazione>, dove <nome-relazione> è una stringa, se il nome include degli spazi è necessario racchiudere la stringa tra apici.
Il formato per gli attributi è: @attribute <nome-attributo> <tipo> dove il primo argomento deve iniziare con un carattere alfabetico e se è composto da spazi sono necessari gli apici.
L'argomento <tipo> può essere uno dei quattro tipi attualmente supportati da Weka:

- Numeric: l'intero insieme dei numeri reali
- String: attributi formati da qualsiasi stringa di caratteri
- Date [<formato data>]: data in vari formati
- <nominal-specification>: lista di etichette definite dall'utente da assegnare ad un attributo

1.3.2 Sezione Data

Questa sezione, contente i dati delle tuple, inizia con la riga che riporta l'etichetta "@data": ogni tupla è descritta in una singola riga che termina con un carattere di ritorno-carrello (carriage return) e gli attributi sono delimitati da virgole nello stesso ordine con cui sono dichiarati nella sezione Header, i valori null sono rappresentati da un singolo carattere di punto di domanda. I valori di tipo stringa e le liste di etichette sono case sensitive, e ognuna, se contiene spazi, deve essere racchiusa tra apici, le date possono essere inserite usando una stringa di rappresentazione specificata nella dichiarazione degli attributi.

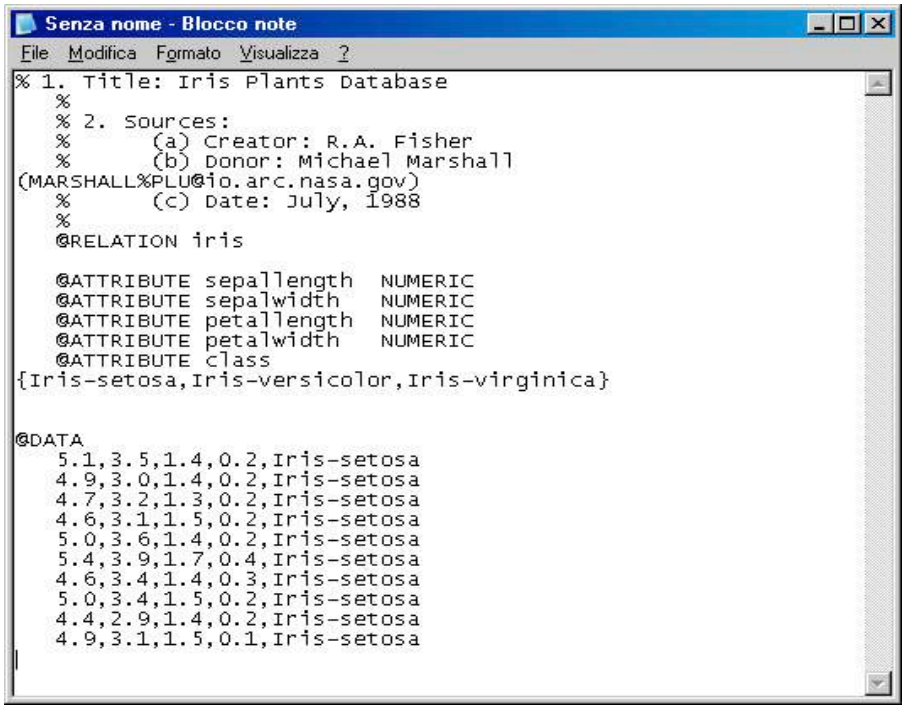

Figura 8 - Esempio di file Arff

1.4 Formato CSV (Comma-Separated Value)

Il comune formato CSV è completamente supportato a patto che il file di testo sia in formato testo Unix (nei sistemi Windows occorre pertanto una conversione per lasciare il solo carattere di line-feed), è necessario però racchiudere le stringhe di caratteri con doppie virgolette, nel caso queste contengano spazi o virgole.

2. Database Genetici

La disponibilità di enormi quantità di dati è di fondamentale importanza per poter applicare qualsiasi tecnica di data mining. Sono due le fonti rese disponibili per l'analisi, la prima comprende la raccolta di dati fenotipici del grano duro e tenero in Italia effettuata tramite il C.R.A., liberamente consultabile tramite il sito web ufficiale. La seconda fonte, Graingenes, è una raccolta di dati genetici complessi riguardanti vari tipi di cereali facente parte di una base di dati in continua evoluzione e anch'essa liberamente consultabile tramite sito web.

2.1 L'Istituto Sperimentale per la Cerealicoltura

Con il D.P.R. n. 1318 del 23 novembre 1967 venne istituito l'ISC, Istituto Sperimentale per la Cerealicoltura con sede centrale a Roma. L'ISC svolge ricerche in campo genetico, agronomico e tecnologico sui cereali, inoltre continua a mantenere e a moltiplicare le sementi delle proprie recenti costituzioni varietali, parte delle quali viene affidata a ditte sementiere che ne curano la moltiplicazione e la commercializzazione. In sintesi, l'attività dell'Istituto si riassume nei seguenti punti:

- Miglioramento delle rese produttive e della qualità finale dei prodotti;
- Disponibilità e impiego di nuova variabilità genetica;
- Nuovi prodotti, costituzioni varietali;
- Erogazione di servizi di carattere tecnico/scientifico, di sostegno ad attività di ricerca, sviluppo, diffusione di istituzioni pubbliche e private;
- Vantaggi tecnologici per l'industria di trasformazione;
- Formazione di nuove professionalità di tipologia qualificata trasferibile nell'industria locale;

- Sostegno alla innovazione e diffusione di tecnologie per valorizzare le produzioni cerealicole.

Attualmente l'istituto è in fase di riorganizzazione per essere annesso al Consiglio per la Ricerca e la Sperimentazione in Agricoltura.

2.1.1 Banca Dati I.S.C.

L'I.S.C. gestisce la banca dati che raccoglie le informazioni riguardanti la rete nazionale per il confronto varietale, il monitoraggio qualitativo, il controllo qualità e le patologie per il frumento duro e tenero distinto per anno.
La base di dati fornita dall'I.S.C. riguarda i dati fenotipici raccolti negli anni 2004/2005 e 2005/2006, tali dati sono stati utilizzati per il data mining tramite Weka e sono attualmente disponibili presso il sito dell'I.S.C. all'indirizzo http://www.cerealicoltura.it

2.2 C.R.A.

Il CRA, Consiglio per la Ricerca e la Sperimentazione in Agricoltura, raccoglie i dati delle prove e del confronto varietale a livello nazionale effettuato in numerose regioni italiane. Il Consiglio per la Ricerca e la sperimentazione in Agricoltura è un Ente nazionale di ricerca e sperimentazione con competenza scientifica generale nel settore agricolo, agroindustriale, ittico e forestale.
Il CRA ha personalità giuridica di diritto pubblico, posto sotto la vigilanza del Ministero delle Politiche Agricole Alimentari e Forestali ed ha autonomia scientifica, statuaria, organizzativa, amministrativa e finanziaria.
Il Consiglio opera sulla base di un piano triennale di attività, aggiornabile annualmente, con cui determina obiettivi, priorità e risorse umane e finanziarie per l'intero periodo, tenuto conto anche dei programmi di ricerca dell'Unione europea e delle esigenze di ricerca e sperimentazione per lo sviluppo delle regioni.
Istituito con D.L.vo 454/99, il CRA raccoglie le esperienze di 28 Strutture di ricerca e sperimentazione agraria e delle rispettive 54 sedi operative periferiche. Il 22 marzo 2006, ottenuto il parere favorevole della Conferenza Stato – Regioni e

Province autonome, è stato emesso il decreto di approvazione del "Piano di riorganizzazione e razionalizzazione" deliberato dal Consiglio di Amministrazione del CRA. Il Piano ha previsto l'attivazione di quattro Dipartimenti cui afferiscono 15 Centri di Ricerca (di cui uno interdipartimentale) e 32 Unità di ricerca. La distribuzione sul territorio nazionale consente al CRA di diffondere capillarmente le proprie competenze, operando sinergicamente con le Amministrazioni centrali, gli Enti locali, le Imprese e le Associazioni di categoria. Alla luce della recente riorganizzazione, le Regioni, in particolare, quali organi di raccordo con le realtà territoriali e l'agricoltura, assumono un ruolo di primo piano nella definizione degli orientamenti della ricerca del CRA. L'aggregazione in un unico Ente consente di perseguire il duplice obiettivo di consolidare l'esperienza di Istituti di ricerca storici e di adeguarsi alle crescenti necessità di innovazione del settore e all'evoluzione della tecnologia. Tali prospettive pongono il CRA nell'ottica di una rinnovata competitività della ricerca agraria sul piano europeo e internazionale e di una nuova operatività del sistema socio-economico nazionale.

2.2.1 Missione

L'attività del CRA è rivolta alla valorizzazione e al miglioramento della produzione agricola ispirata ai criteri della sostenibilità, della tracciabilità, della multifunzionalità. Da più di 100 anni gli Istituti del CRA sviluppano ricerca di base, ricerca applicata e sperimentazione in agricoltura, perseguendo obiettivi quali: la valorizzazione dei prodotti tipici dell'agroalimentare e la sicurezza alimentare, l'innovazione tecnico-scientifica di processi e prodotti per accrescere la competitività delle imprese, l'integrazione delle aree marginali e svantaggiate, la tutela dell'ambiente, del territorio e quella della biodiversità vegetale, animale e dei microrganismi. In alcuni settori – fragole, drupacee, pioppi, grano duro – il CRA è leader a livello mondiale. Non solo. Il CRA svolge attività di certificazione, prova e accreditamento, presta consulenze e collabora con alti Enti pubblici e privati del settore, ricoprendo un ruolo strategico e di relazione con il mondo degli operatori delle imprese agricole e farmaceutiche, quali destinatari

delle evidenze prodotte dalle ricerche. Per il perseguimento di tali obiettivi l'Ente agisce attraverso una struttura organizzata a rete che, unita alla collaborazione con gli operatori locali, consente l'applicazione di una politica di gestione integrata rivolta a favorire lo sviluppo rurale. Uno sviluppo fondato sulla crescita della competitività delle imprese e delle aziende del settore e teso alla valorizzazione e creazione di attività alternative che mitigano il fenomeno dell'esodo rurale. Fortemente legata al territorio, dunque, l'attività del CRA continua una tradizione di eccellenza scientifica, riconosciuta anche a livello internazionale, per orientarsi verso percorsi innovativi in linea con le emergenti esigenze socio-economiche del Paese.

2.3 Il Database Graingenes

Graingenes è un database genetico che raccoglie dati su frumento, avena e zucchero di canna che fa parte del National Agricultural Library's Plant Genome Program del Dipartimento per l'Agricoltura degli Stati Uniti d'America, le categorie di informazioni che raccoglie includono:

- Mappe genetiche e citogenetiche;
- Analisi genomica, sequenze di nucoleotidi;
- Geni, alleli e produzione di geni;
- Fenotipi, trait quantitativi e QTL (Quantitative Trait Locus);
- Genotipi e pedigree di cultivar e altri tipi di germplasm;
- Patologie e corrispondenti agenti patogeni, insetti e stress abiotici;
- Tassonomia del frumento e dell'avena;
- Indirizzi di persone che effettuano ricerche;
- Richiami bibliografici di interesse rilevante.

Il progetto è molto vasto e il lavoro di inserimento dei dati è in continuo progresso, la base di dati viene seguita da diversi soggetti coordinatori americani che si occupano di gestire una parte specifica delle informazioni citate precedentemente.

2.3.1 DMBS di Graingenes

A livello software la base di dati è mantenuta utilizzando il software AceDb, un Dbms genomico sviluppato nel 1989 da Jean Thierry-Mieg e Richard Durbin in linguaggio C, composto da una interfaccia grafica e diversi tool per la manipolazione dei dati. Per diverso tempo la base di dati è stata gestita tramite piattaforma AceDb, un Dbms genomico sviluppato nel 1989 da Jean Thierry-Mieg e Richard Durbin in linguaggio C, composto da una interfaccia grafica e diversi tool per la manipolazione dei dati. La piattaforma AceDb è stata abbandonata a favore di MySql, Dbms relazionale che permette costi contenuti, semplicità di utilizzo, velocità e facile reperimento di documentazione. Il processo di migrazione verso il nuovo sistema relazionale ha comportato quattro fasi di lavoro: traduzione del modello dati, esportazione dati in formato Acedb in nuove tabelle MySql, creazione della nuova interfaccia per consultazione web e in fine sviluppo di tool specifici da affiancare al sistema MySql di base. Fondamentalmente, la migrazione verso il nuovo Dbms ha comportato la creazione di diverse tabelle associate a un singolo oggetto in formato Acedb e il nuovo schema relazionale ottenuto è più complesso rispetto al sistema utilizzato in AceDb, tuttavia si è avuto un incremento di prestazioni soprattutto nei casi in cui la consultazione avviene tramite query complesse formulate direttamente nel sito web di Graingenes: il tempo di risposta è dell'ordine di qualche secondo (in precedenza si potevano avere anche attese dell'ordine dei minuti). Graingenes viene aggiornato con frequenza giornaliera ed è consultabile anche tramite una versione scaricabile per consultazione locale.

Esistono inoltre due basi di dati che si affiancano a Graingenes, esse sono Graingenes Webserver e Graingenes Gopher che contengono anch'esse una notevole quantità di informazioni strutturate in modo differente dal sistema AceDb e includono anche dati in testo puro e html, esse includono anche newsletter annuali sul grano, cataloghi di simboli genetici, cultivar, valutazioni di qualità.

2.3.2 Consultazione tramite bowser web

La home page del sito, accessibile all'indirizzo http://wheat.pw.usda.gov, riporta a sinistra l'elenco di voci dedicate alla consultazione distinte per tipologia. È possibile ottenere le informazioni che si cercano in modi differenti e con metodi di formulazione di query che vanno dal più semplice al più complesso.

Vi è la possibilità di formulare query con livelli di complessità crescente, accedere a risorse presenti sul web ovvero a link di siti affini, ottenere nominativi di persone che hanno effettuato ricerche, ricevere aggiornamenti del sito tramite newsletter e infine vi è la possibilità di collaborare apportando nuovi dati o contattare i responsabili per una eventuale richiesta di lavoro.

Si riporta di seguito una panoramica delle caratteristiche salienti offerte dal sito che riguardano la consultazione e il reperimento dei dati.

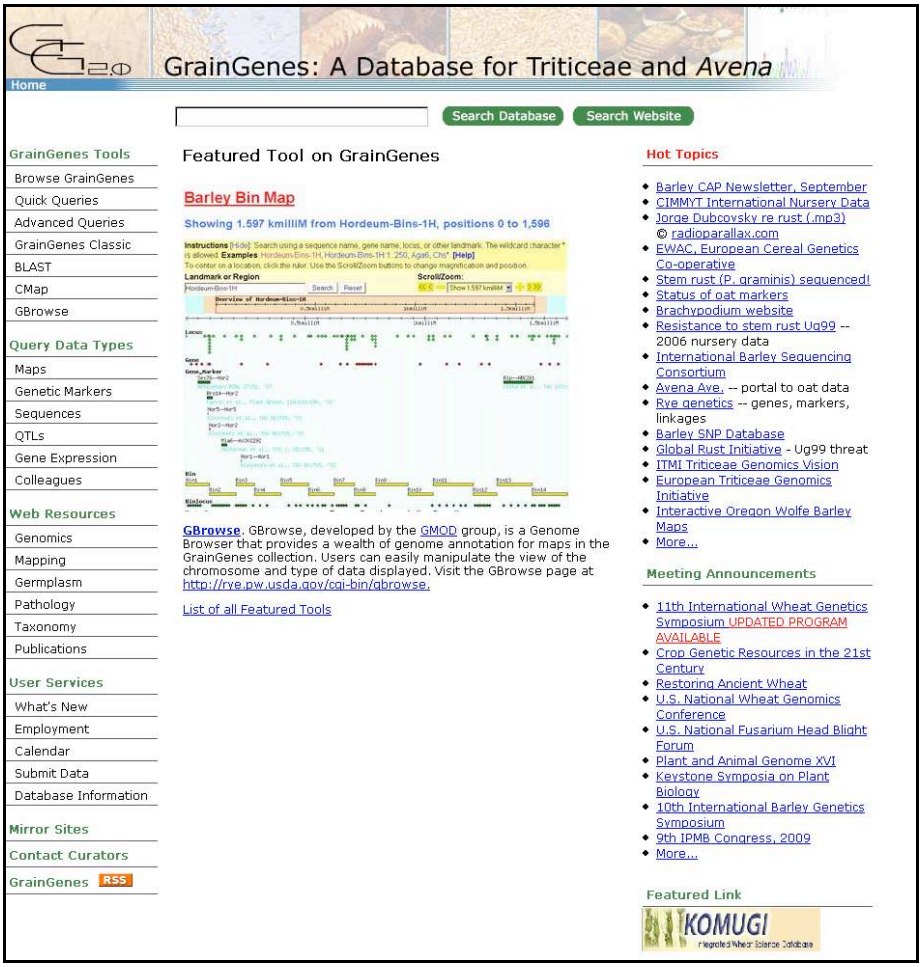

Figura 9 – Home page di Graingenes

2.3.3 GrainGenes Tools

Questa sezione, probabilmente la più importante per la consultazione, è dedicata alla estrapolazione dei dati di interesse. Tramite la voce Browse GrainGenes si ha a disposizione una panoramica di tutte le tipologie di voci archiviate, cliccando su una delle voci compare una lista associata di termini che a loro volta sono consultabili ulteriormente nei dettagli con un click di mouse.

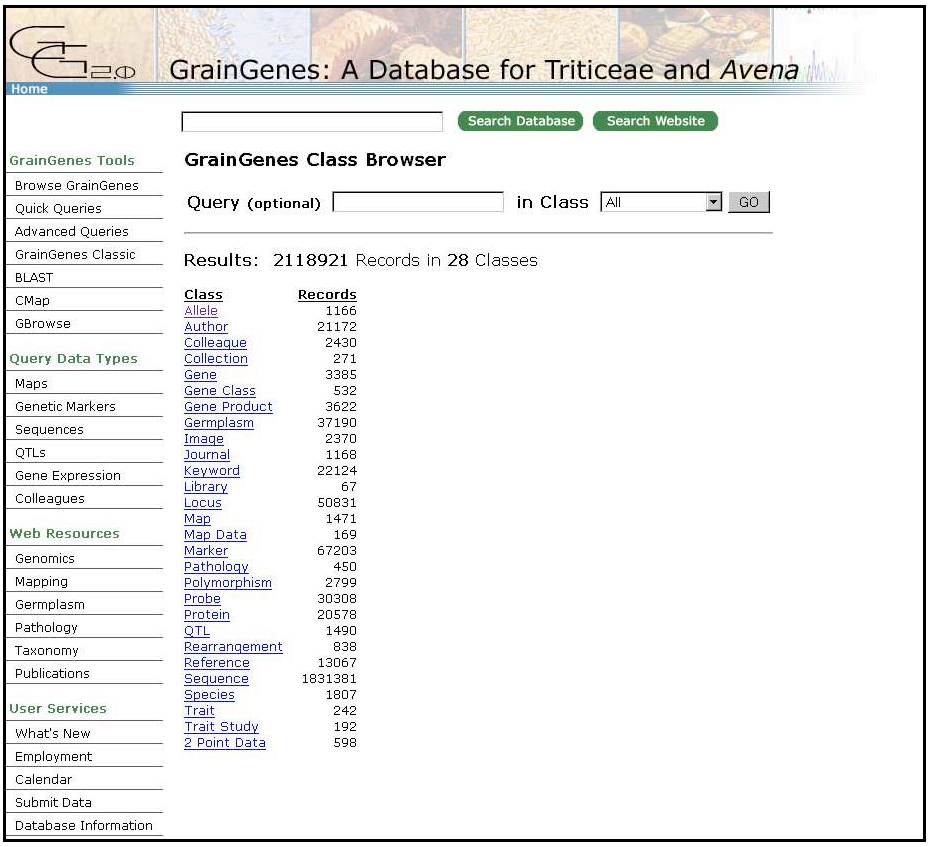

Figura 10 – Graingenes Class Browser

2.3.4 Quick Queries

Questa pagina permette di formulare query di ricerca rapide e con un grado di complessità non troppo elevato, vi sono dei campi da compilare già predisposti per ogni categoria di ricerca. Questa modalità permette un accesso guidato e veloce ai dati di interesse evitando che l'utente possa perdere tempo con lunghe query, inoltre viene specificato all'inizio che le query presenti sono quelle che si ritengono più frequenti.

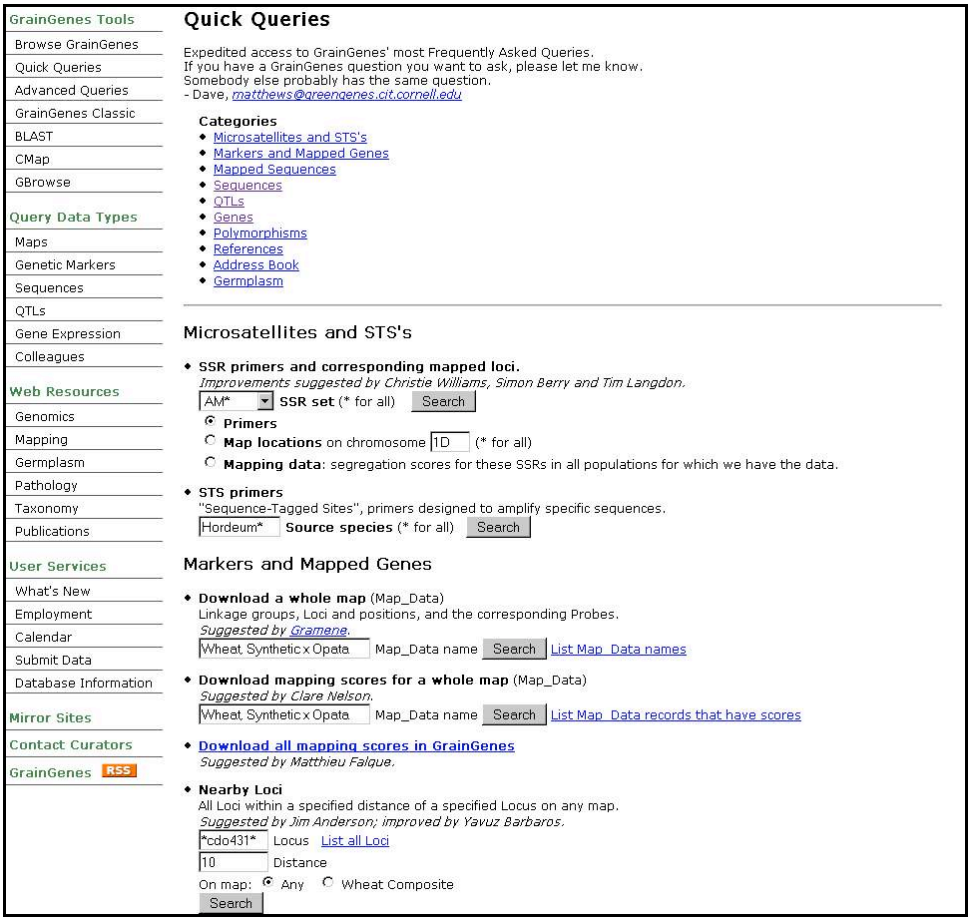

Figura 11 – Quick Queries di Graingenes

2.3.5 Advanced Queries

Query avanzate che permettono un alto grado di personalizzazione e anche di complessità possono essere formulate in questa pagina, le modalità con cui possono essere create sono diverse:

- Field-based Search permette di effettuare una ricerca orientata ai valori di singoli attributi con la possibilità di utilizzare wildcard, attualmente però tale ricerca è limitata solamente a geni e sequenze
- SQL Interface: creazione di query totalmente a carico dell'utente utilizzando il linguaggio SQL. E' possibile però selezionare una lista di query precostituite da modificare poi a piacere. Si possono consultare le definizioni SQL di tutte le tabelle del database e ottenere anche l'insieme delle tabelle appartenenti a una classe specifica (Gene, Allele, Personale di ricerca, ecc...) come file di immagine.

- Batch SQL Interface: uguale alla voce precedente ma in più permette di restringere ulteriormente gli attributi di ricerca in modo più comodo e dettagliato
- Batch Query to get Sequence in FASTA Format: creazione di query in formato testo per rappresentare sequenze di acidi nucleici o sequenze di peptidi in cui le coppie di basi o gli aminoacidi sono rappresentati utilizzando codici costituiti da singole lettere. Il formato permette anche di aggiungere commenti e nomi che precedono le sequenze. E' un formato proposto dal software omonimo dedicato all'elaborazione di sequenze di DNA e proteine.
- Batch Query to get all FASTA Sequences for a list of Probes: simile alla voce precedente ma invece delle sequenze ricerca dei nomi.

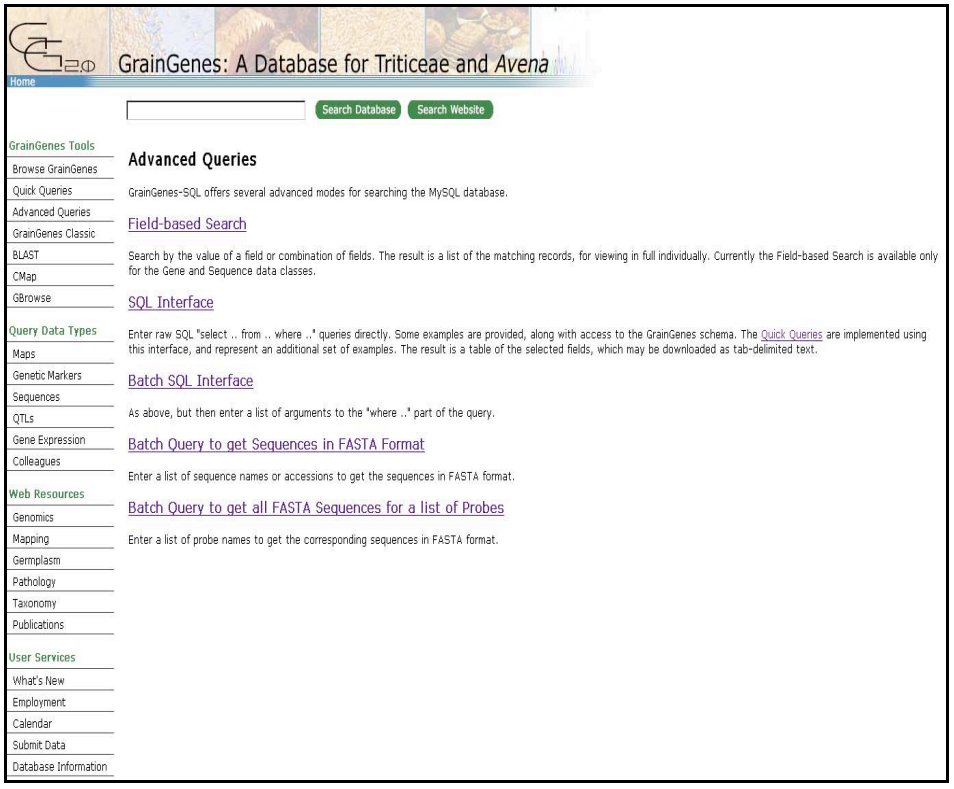

Figura 12 – Advanced Queries di Graingenes

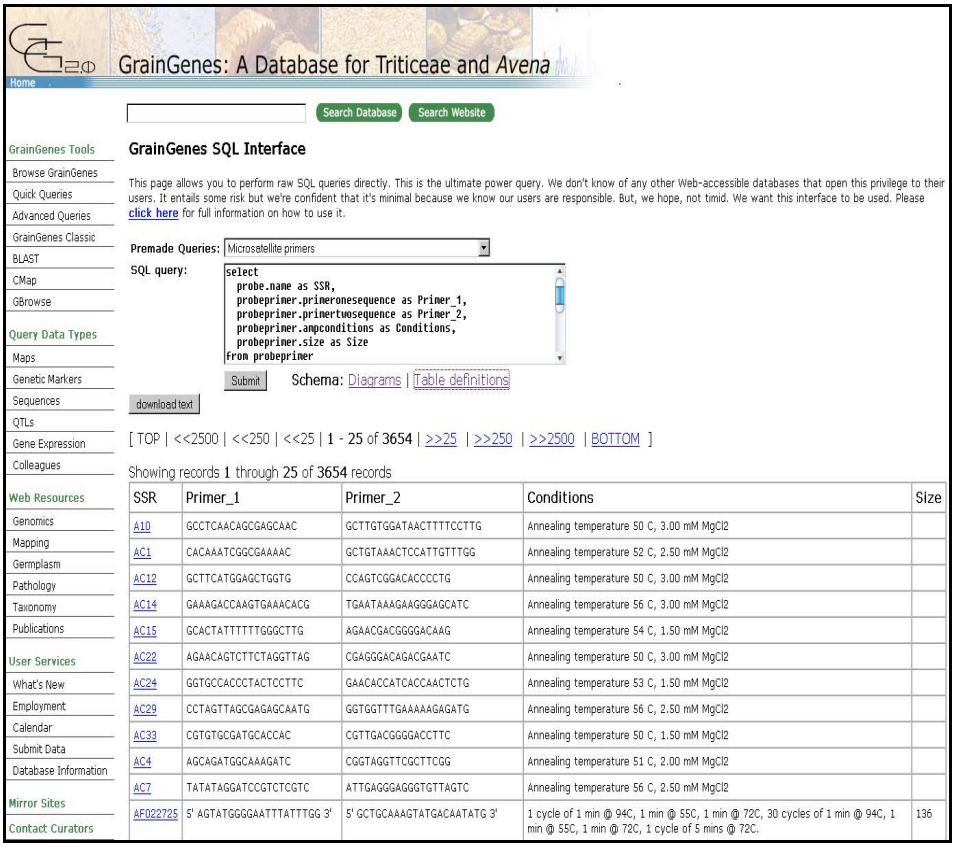

Figura 13 – Graingenes SQL Interface

2.3.6 GrainGenes Classic

E' la vecchia interfaccia utilizzata dal sito Graingenes che viene mantenuta per essere accessibile agli utenti che sono rimasti fedeli all'interfaccia originaria. Raccoglie in modo sobrio, sintetico ma efficace la formulazione di query e di consultazione.

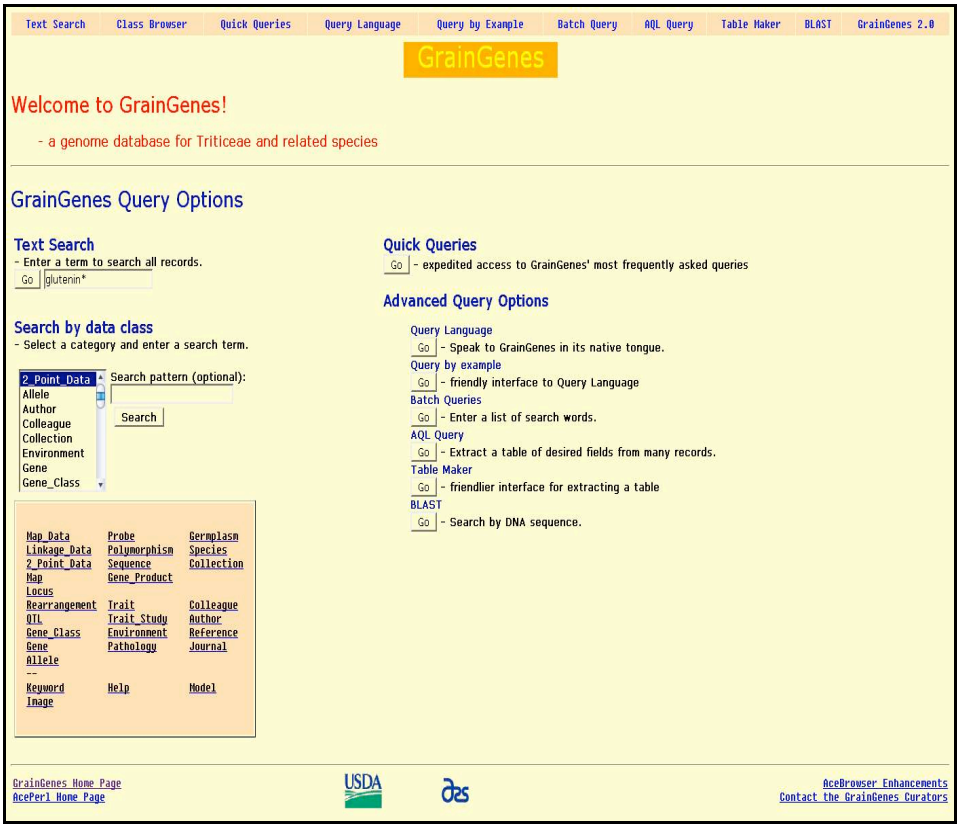

Figura 14 – Graingenes Classic

2.3.7 BLAST

Utilizzo di query basate su tool BLAST: un algoritmo per confrontare sequenze di aminoacidi, proteine, nucleotidi o sequenze di DNA, l'input e l'output viene impostato in formato FASTA.

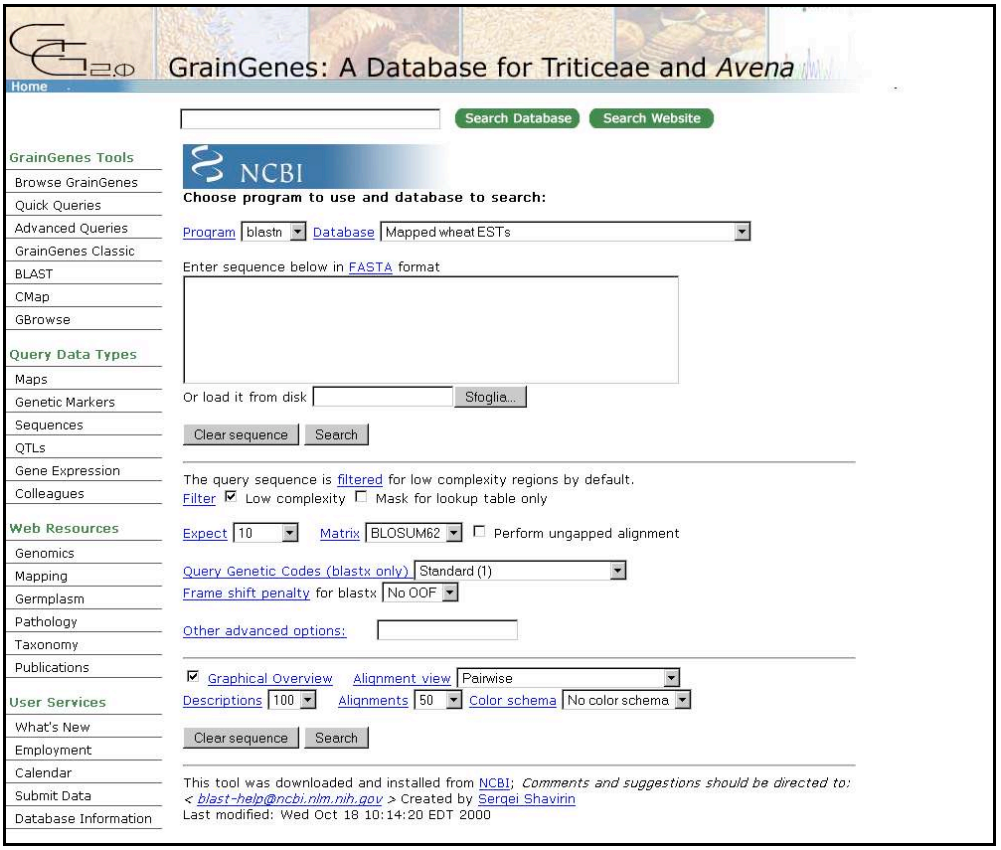

Figura 15 – Consultazione tramite tool BLAST

2.3.8 Cmap

Cmap è un servizio di visualizzazione grafica di mappe di GrainGenes, ovvero permette di ottenere le stesse mappe genetiche ottenibili tramite la pagina GrainGenes Classic e permette di confrontarle direttamente fra di loro in modo più semplice e soddisfacente, è una sorta di estensione delle funzionalità di GrainGenes Classic.

2.3.9 Gbrowse

Gbrowse è un tool per visualizzare i dati trattati con GMOD. Gmod (Generic Model Organism Database) è una collezione di tool software per creare e gestire basi di dati genomiche. Gbrowse permette di consultare in modo rapido e con output in formato grafico i dati di GrainGenes.

2.3.10 Query Data Types

Questa sezione è dedicata alla formulazione di query per consultazione di mappe, marcatori genetici, sequenze, qtl, espressioni geniche e informazioni riguardanti le persone che hanno effettuato ricerche. Le query impostabili possono avere un grado di complessità che dipende solo dalla scelta dell'utente. I campi per la compilazione sono analoghi a quelli della sezione GrainGenes Tools ma permettono solo l'utilizzo del linguaggio Sql.

2.3.11 Web Resources

Vasta collezione di link a siti affini o analoghi: progetti di raccolta dati genomici, mappature, articoli scientifici, ricerca sulle malattie, tabelle, nomenclature, pubblicazioni scientifiche.

2.3.12 User Services

Raccoglie informazioni riguardanti l'aggiornamento del sito e della base di dati GrainGenes, mostra gli incontri, le conferenze e altri eventi in genere del passato e in programmazione, permette di contattare i responsabili del progetto GrainGenes e di scaricare una versione locale del database

2.4 Formato e scelta dei dati da analizzare

I dati provenienti da Graingenes e CRA sono stati ottenuti in formato Mysql 5.0 e Sql Server 2000 rispettivamente e rappresentano la versione offline dei rispettivi archivi presenti sui siti web. Le query utilizzate per ricavare i dati di interesse sono state scelte in base alla quantità di tuple risultanti e attributi con il minor numero di valori nulli possibile. I risultati sono stati successivamente convertiti in formato testo CSV (Comma separated value) e infine aperti in Weka.
Sfortunatamente, per quanto riguarda Graingenes diverse query ritenute interessanti sono state scartate perchè non hanno dato alcuna tupla risultante: la

causa principale è da imputare alla notevole presenza di valori nulli sparsi ovunque nell'intero database.

3. Data Mining su database CRA

Il database CRA è costituito da diverse tabelle tutte simili fra di loro tranne per un attributo, quest'ultimo dà il nome alla tabella stessa e riporta le misure fenotipiche di interesse per l'analisi. I restanti attributi, ovvero tutte le altre colonne in comune tra le tabelle, hanno il seguente significato:

- `Germplasm`: nome associato alla pianta esaminata, mantenuta in purezza per il mantenimento e la conservazione del patrimonio genetico;
- `Species`: specie del cereale, tale attributo assume solo due valori riferiti al germplasm ovvero specifica se è "grano duro" (Durum Wheat) o "grano tenero" (Bread Wheat);
- `Taxon`: tipo di raggruppamento a cui appartiene un germplasm;
- `Germplasm type`: tipo di germplasm utilizzato;
- `Pedigree_CRA`: pianta o incrocio di piante da cui si è ottenuto un Germplasm;
- `Repository_CRA`: sede dell'istituto che ha reso disponibile un Germplasm;
- `Studies_or_Environment`: regione italiana in cui è stata compiuta la misurazione fenotipica di un Germplasm.

Non tutti gli attributi sopra elencati sono stati utilizzati: quelli significativi sono quelli che riguardano i germplasm, la species e la regione italiana che ha effettuato le misure. Gli attributi fenotipici invece saranno riportati nelle query presentate nei prossimi paragrafi.

3.1 Descrizione del lavoro

Le tabelle non presentano un ordinamento specifico in base a un valore di chiave, esse probabilmente sono state pensate per la semplice consultazione senza particolare riguardo ad eventuali manipolazioni o utilizzi più complessi. Allo scopo di formulare query utili per confrontare due o tre attributi fenotipici contemporaneamente si è imposto nel codice SQL l'uguaglianza tra i valori di germplasm, specie e regioni italiane: in questo modo è stato possibile ottenere tuple confrontabili ovvero ottenere una lista di nominativi di germplasm che avessero in comune la stessa regione di provenienza e la stessa specie, ottenendo così un elenco esauriente di valori.
Le fasi di lavoro per gli attributi fenotipici esaminati sono le seguenti:

1. Formulazione query in Microsoft Sql Server 2000;
2. Salvataggio delle tuple ottenute in un file di testo in formato CSV;
3. Apertura del file di testo in Weka Explorer;
4. Applicazione dell'algoritmo di clustering k-means;
5. Ricerca di corrispondenze o relazioni fra punti tramite i grafici ottenuti dall'algoritmo k-means.

Infine, seguiranno le considerazioni finali sulla ricerca effettuata.

3.1.1 Clustering tramite algoritmo k-means

Questo algoritmo di clustering fa parte della classe di metodi che basano il loro funzionamento sulla densità dei dati: l'idea di base è quella di costruire insiemi di dati fin tanto che la densità (intesa come numero di oggetti o valori numerici) nelle "vicinanze" superi una certa soglia prestabilita.
L'algoritmo richiede come valore di input il numero *k* di cluster da trovare: questa informazione dipende esclusivamente dall'utente. Valori troppo alti possono frammentare l'insieme di dati e renderlo non idoneo ad alcuna interpretazione, valori troppo bassi potrebbero invece dare risultati bizzarri e di scarso valore,

tuttavia è sempre consigliabile partire con bassi valori. L'algoritmo k-means è stato scelto per la sua semplicità di utilizzo e la rapidità di elaborazione, inoltre i dati da esaminare non sono molto articolati e i valori presi a confronto sono solo di tipo numerico, l'algoritmo quindi può rappresentare una soluzione di clustering prudente e ben proporzionata alla base di dati offerta per il lavoro di analisi.

3.1.1.1 Descrizione dell'algoritmo

Dato un insieme D che rappresenta il training set composto da n oggetti e k il numero di cluster da formare, un algoritmo di clustering riorganizza gli oggetti in k partizioni (con $k \leq n$) in cui ogni partizione rappresenta un cluster.

Vengono inizialmente scelti k oggetti, ognuno di essi rappresenta un centroide ovvero un punto da considerare il centro del cluster da trovare. I rimanenti oggetti che non sono centroidi vengono assegnati al cluster più simile, ovvero che ha un determinato valore basato sulla distanza tra il centroide e l'oggetto stesso che si sta valutando. Questo processo iterativo finisce quando la funzione-criterio converge. Tipicamente, la funzione-criterio è definita in base all'errore quadratico ovvero

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2,$$

Dove E è la somma dell'errore quadratico per tutti gli oggetti; p è il punto che rappresenta un determinato oggetto nello spazio di valori del training set, m_i è la media del cluster C_i (sia p che m_i sono multidimensionali). In altre parole, per ogni oggetto in ogni cluster, la distanza calcolata dall'oggetto al suo centroide è al quadrato e le distanze vengono sommate. Questo criterio tenta di ottenere k cluster compatti e nello stesso tempo separati tra di loro. Pseudocodice dell'algoritmo:

1) scelta arbitraria di *k* oggetti dall'insieme di dati di partenza *D* da considerare come centroidi;
2) **repeat**
   - assegna (o riassegna) un oggetto a un cluster in base al valore della media degli oggetti già presenti nel cluster stesso;

- aggiorna il valore della media di tutti i cluster, ovvero calcola le medie degli oggetti di ogni cluster.

until non vi è cambiamento nei valori delle medie

L'algoritmo produce risultati validi nel caso in cui dati sono distribuiti con densità concentrata in zone ben distinte, ovvero zone che presentano un andamento di punti che ricalca quella gaussiana o in genere che presentano una distanza non troppo elevata fra di loro.

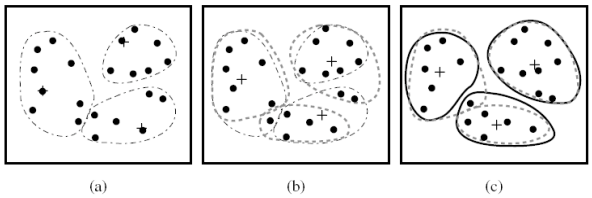

Figura 16 - Clustering k-means, la media è rappresentata da un "+"

L'algoritmo può essere applicato solo quando è possibile creare un centroide numerico che rappresenta la media del cluster corrente, ovvero solo in presenza di attributi numerici.

3.1.1.2 Implementazione in Weka

In Weka utilizzare k-means è molto semplice: è necessario fornire il numero di cluster da cercare e un numero a piacere che rappresenta il seme per la scelta casuale delle k partizioni in cui suddividere i dati in ingresso. Il valore del seme utilizzato nell'analisi è quello di default salvo quando viene specificato un valore diverso.

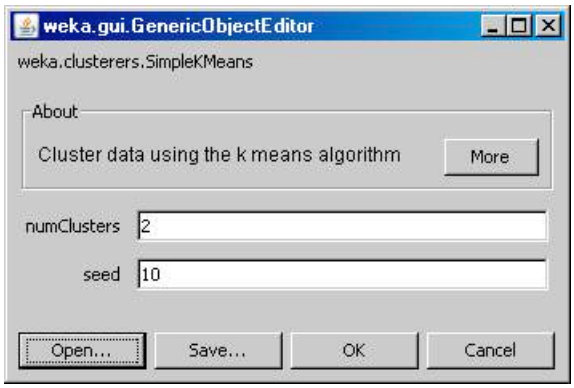

Figura 17- Finestra SimpleKMeans

3.1.1.3 Utilizzo del selettore Jitter

Nel mostrare i grafici rappresentativi di cluster di punti o di semplici valori di tuple verrà utilizzata, se ritenuta necessaria, (anche per quanto riguarda il data mining su database Graingenes) la funzione Jitter. In un qualsiasi grafico di punti, il Jitter (letteralmente "tremolio") è un cursore scorrevole che permette di riportare sugli assi uno spostamento casuale dato a tutti i punti nella rappresentazione corrente. Aumentare la quantità di jitter è utile per mettere in evidenza le concentrazioni di punti in tutto il grafico. Senza lo jitter, moltissime istanze non sembrerebbero differenti da una singola istanza isolata.

3.2 Query su attributi a1000_Kernel_Weight e Plant_Height

```
SELECT Germplasm,
       Species,
       P.Studies_or_Environment_CRA
       P.Plant_Height_Cra as Plant_Height_Cra,
       G.a1000_Kernel_Weight_CRA

FROM   Plant_Height_Cra P, a1000_Kernel_Weight_CRA G

WHERE  P.Germplasm=G.Germplasm
AND    P.Species=G.Species
```

```
AND     P.Studies_or_Environment_CRA =
        G.Studies_or_Environment_CRA
```

I due attributi da confrontare hanno il seguente significato:

- `a1000_Kernel_Weight_CRA`: misura in grammi di mille chicchi interi (seme con involucro) essiccati fino al 13% di umidità del loro contenuto e infine pesati su una bilancia di precisione;
- `Plant_Height_CRA`: misura in centimetri dell'altezza della pianta conteggiata dal suolo fino al punto più alto che essa raggiunge.

3.2.1 Apertura file in Weka Explorer

Nella sezione Preprocess di Explorer, una volta selezionato l'attributo da visualizzare è possibile avere una visione d'insieme delle distribuzioni distinte per Bread Wheat e Durum Wheat. Weka colora automaticamente i due grafici distinti ma non è possibile intervenire su alcun parametro di visualizzazione del grafico (scala, legenda, ecc…): in figura 18 e 19 vengono riportate le distribuzioni dei valori di entrambe le specie per i due attributi di interesse così come vengono visualizzate in Explorer nella sezione Preprocess, il colore blu fa riferimento ai valori di Bread Wheat, il colore rosso ai valori di Durum Wheat.

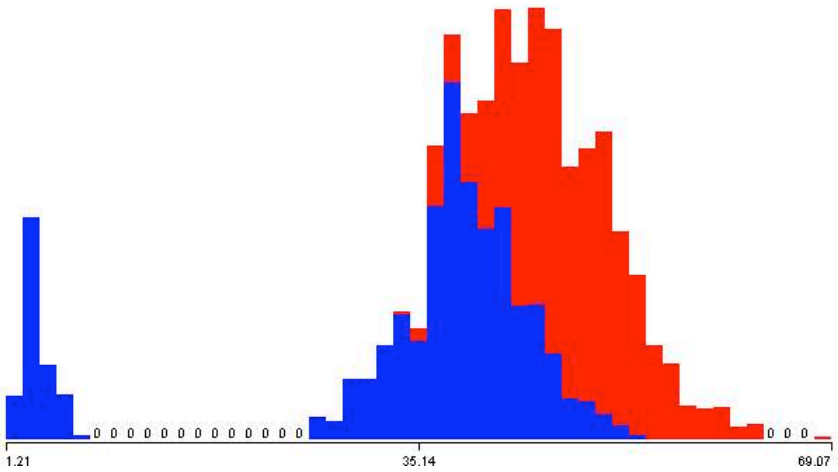

Figura 18 – Distribuzione valori attributo a1000_Kernel_Weight_Cra

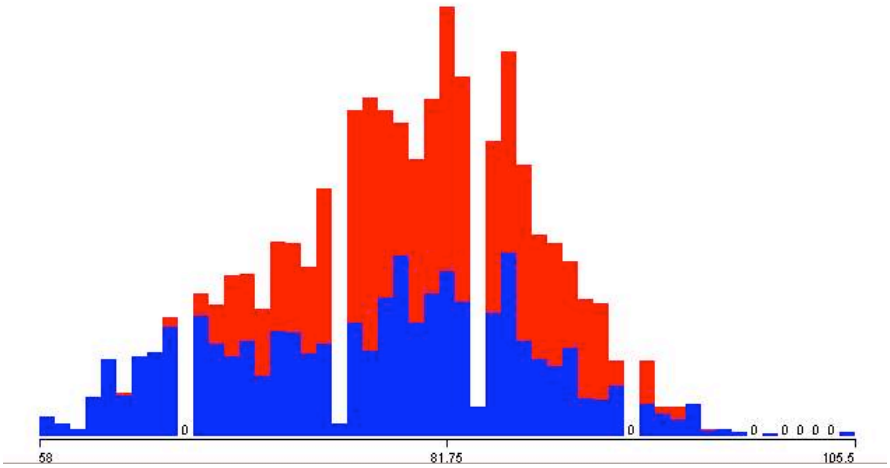

Figura 19 - Distribuzione valori attributo Plant_Height_CRA

È inoltre comodo poter visualizzare in modo più dettagliato gli andamenti dei valori su un grafico più grande, allo scopo si utilizza la sezione Visualize: Weka può visualizzare automaticamente una serie di grafici derivanti dalla combinazione di tutti gli attributi presenti ma spetta all'utente selezionare il grafico di interesse. È possibile in ogni momento impostare i valori per le ascisse e le ordinate e un colore in base alla "Classe" di attributi interessata. In questo caso si mostra in figura 20 e 21 i grafici riguardanti i la distribuzione dei punti dei due attributi numerici distinti per germplasm in ascissa e specie in ordinata, i punti in colore blu riguardano la specie Bread Wheat, quelli in colore rosso la specie Durum Wheat.

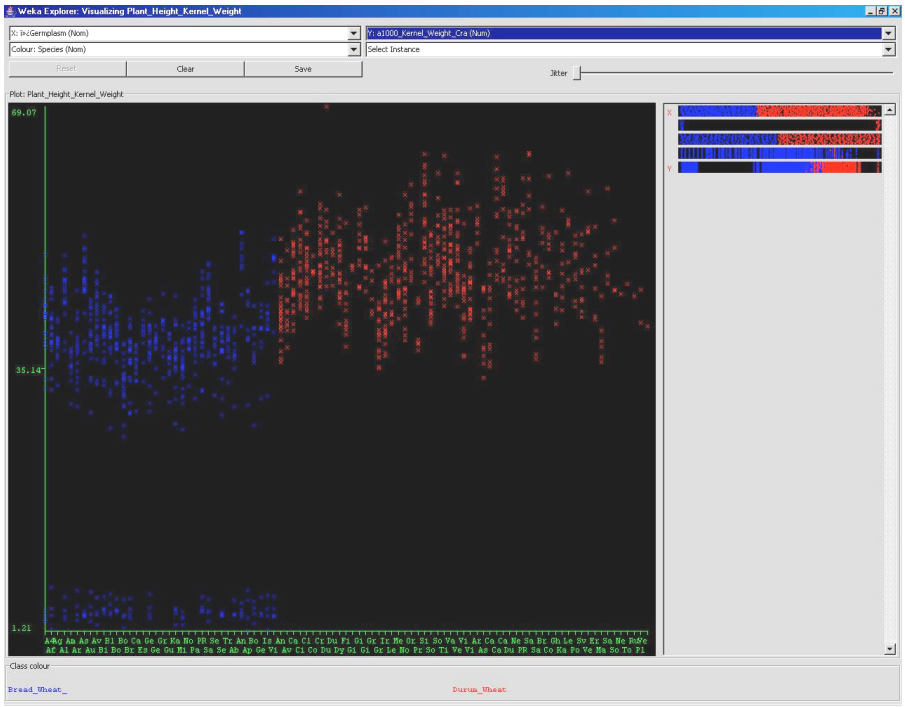

Figura 20 – Valori dell'attributo a1000_Kernel_Weight di entrambe le specie

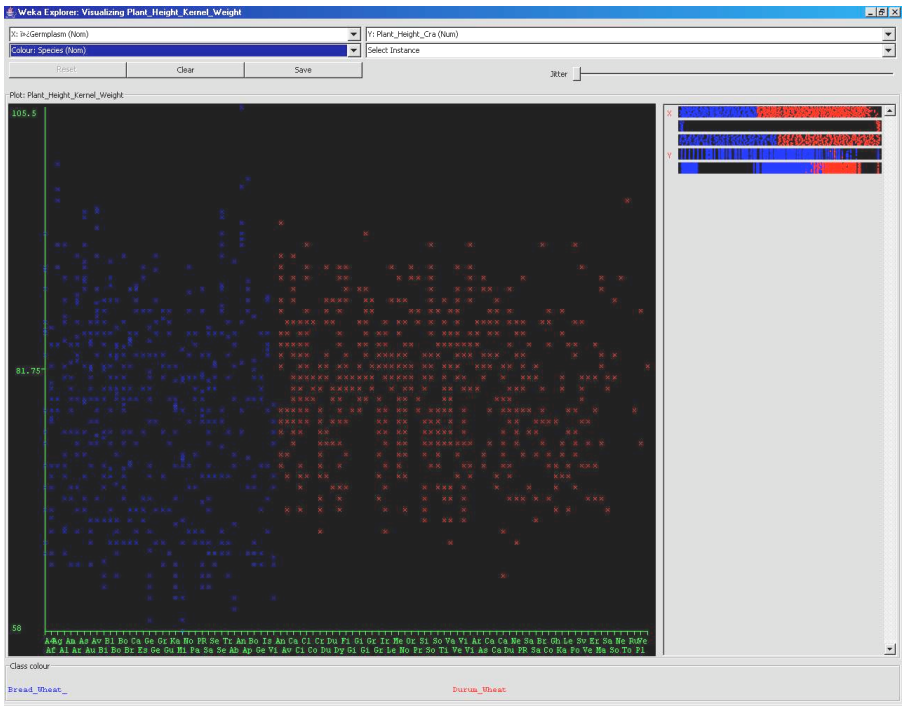

Figura 21 – Valori dell'attributo Plant_Height di entrambe le specie

3.2.2 Ricerca di cluster

Vediamo a questo punto se tramite l'algoritmo di clustering è possibile trovare qualche correlazione: i dati con specie Bread_Wheat vengono divisi dalla specie Durum_Wheat ed elaborati separatamente.
Weka permette in modo comodo ed efficace di effettuare questa divisione, non è necessario creare una nuova query SQL: basta selezionare i valori dell'attributo `Bread_Wheat` direttamente su un grafico della sezione Visualize tramite la modalità di selezione in alto a destra con etichetta "Select Instance".
In questo caso, essendo i punti di ogni specie ben distinti, verrà selezionata la modalità "Rectangle": con il mouse si traccia un rettangolo che copre i punti e si conferma l'insieme scelto remendo il pulsante "Submit". Si presenta così un nuovo grafico a cui sono associati i soli punti (e di conseguenza anche le tuple) visualizzati tramite la selezione. A questo punto si salvano i valori di interesse in un nuovo file di testo in formato Arff (Weka attualmente permette di salvare i dati di un grafico solamente in questo formato) premendo il pulsante "Save" in alto a sinistra. Riaperto il nuovo file nella sezione Preprocess, viene eliminato l'attributo Species perché ridondante e si impostano i parametri dell'algoritmo di clustering nella sezione Cluster.

3.2.2.1 Specie Bread Wheat, attributo `a1000_Kernel_Weight_CRA`

Eseguiamo una ricerca di due cluster inserendo il valore nel campo numClusters e valore 20 per Seed; il risultato che mostra Weka per l'output testuale è questo:

```
=== Run information ===

Scheme:       weka.clusterers.SimpleKMeans -N 2 -S 20
Relation:     Plant_Height_Kernel_Weight
weka.filters.unsupervised.attribute.Remove-R2
Instances:    13461
Attributes:   4
              Germplasm
              Studies_or_Environment
```

```
              Plant_Height_Cra
              a1000_Kernel_Weight_Cra
Test mode:    evaluate on training data

=== Model and evaluation on training set ===

kMeans
======

Number of iterations: 15
Within cluster sum of squared errors: 25529.75103505749

Cluster centroids:

Cluster 0
Mean/Mode:  Sagittario Friuli,_Italy,_2004/2005 77.4235
38.6806
Std Devs:   N/A     N/A      9.033   6.3842

Cluster 1
Mean/Mode:  Aubusson Marches,_Italy,_2005/2006 77.6718
15.5426
Std Devs:   N/A     N/A      8.3067 16.639

Clustered Instances

0      10670 ( 79%)
1       2791 ( 21%)
```

Weka mostra rispettivamente la moda e la media per gli attributi non numerici e numerici del file arff in esame, dispone le informazioni su una unica riga separate da spazio, nella riga sottostante invece viene riportata la deviazione standard degli

attributi numerici dei due distinti cluster. Infine, la quantità di istanze (tuple) associate ai cluster calcolati. Cliccando con il pulsante destro del mouse sulla voce appena creata in Result List si seleziona la visualizzazione grafica per vedere direttamente le assegnazioni dei punti ai cluster. In modo identico a come è stato descritto precedentemente, è possibile selezionare le ascisse e le ordinate: si è scelto di mantenere `Germplasm` per le ascisse e `a1000_Kernel_Weight` e `Plant_Height` per le ordinate: weka conta i cluster partendo da zero assegnando loro un colore. Per quanto riguarda l'attributo `a1000_Kernel_Weight`, l'algoritmo ha creato due cluster che si sono ben conformati alla distribuzione dei valori originari salvo una minoranza di punti che si notano subito sul grafico, in entrambi i cluster sono presenti una minoranza di elementi del cluster opposto: questi dati possono essere d'interesse e si possono osservare sul grafico con più dettaglio selezionando l'area come fatto precedentemente. Inoltre, cliccando con il mouse direttamente sui punti interessati si apre una finestra con i valori numerici che riguardano, oltre al punto di interesse, anche i valori che si trovano in prossimità di tale punto.

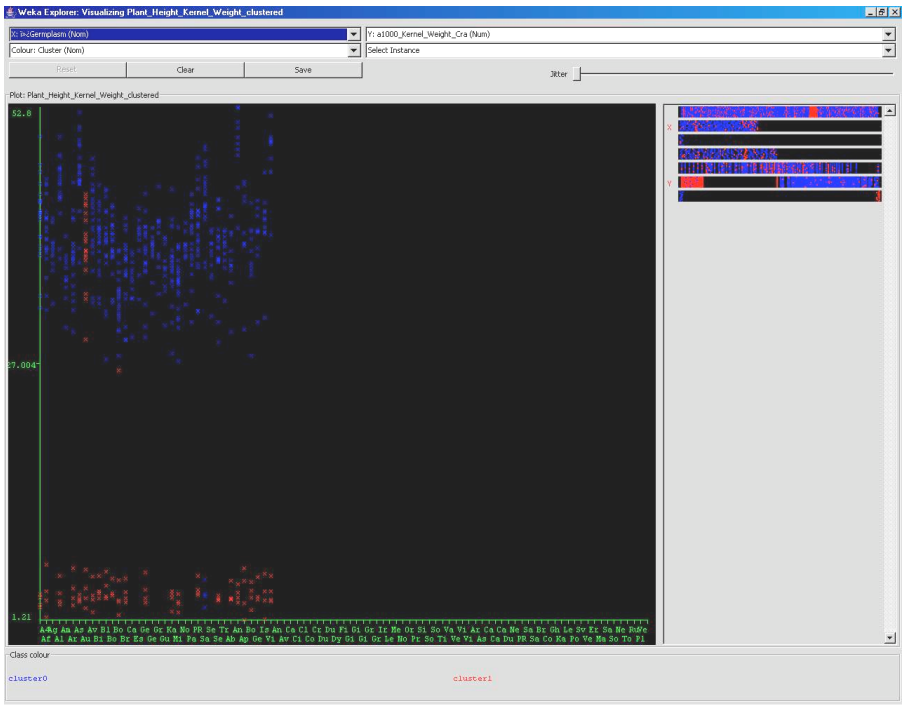

Figura 22 – Clustering sull'attributo a1000_Kernel_Weight

Per controllare i valori anomali, dopo aver selezionato l'area di interesse e salvato i dati, si può ottenere una visualizzazione immediata dei singoli valori numerici di ogni punto cliccando direttamente sul grafico in corrispondenza del punto colorato. La finestra che si apre riporta tutti gli attributi corrispondenti al punto interessato e in questo caso si vede che esiste un dato facente parte del cluster 1 tra tanti valori del cluster 0, tuttavia i valori non risultano diversi dai punti vicini e si può subito affermare che l'algoritmo non ha individuato nulla di particolare.
Un ulteriore controllo è stato fatto sulla provenienza dei dati, ovvero sull'attributo `Studies_or_Environment_CRA`: salvando il grafico in formato Arff con questi ultimi dati e riaprendolo nella sezione Preprocess di Explorer ci viene in aiuto la rappresentazione grafica generale delle distribuzioni dei dati distinti per colorazione di cluster. Come mostrato in figura 25, scegliendo `a1000_Kernel_Weight` nell'area attributi e come classe di visualizzazione l'attributo `Studies_or_Environment_CRA` si vede che le tuple distinte per provenienza hanno le distribuzioni di colore più o meno simili in tutte le colonne, tale distribuzione uniforme può essere vista a colpo d'occhio guardando il grafico impostato con ascissa `Studies_or_Environment_CRA` e ordinata il nome di `Germplasm`. I valori sono tutti simili e graficamente allineati fra di loro, è quindi evidente che non c'è alcuna osservazione particolare da fare per quanto riguarda l'attributo in questione.

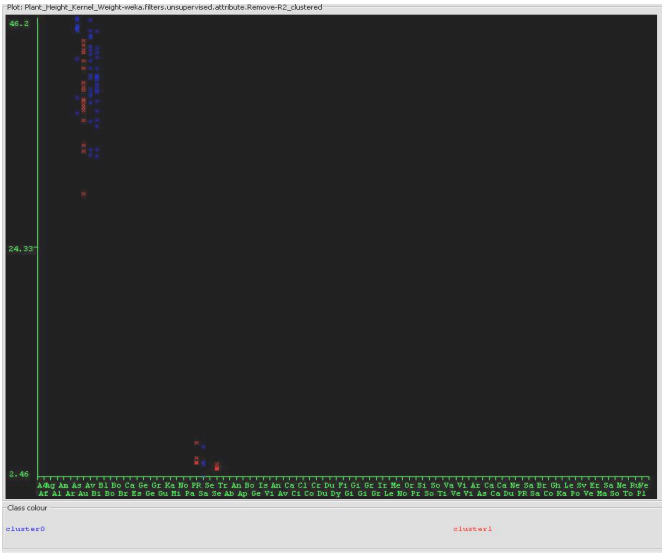

Figura 23 – Presunti punti anomali sul grafico dell'attributo a1000_Kernel_Weight

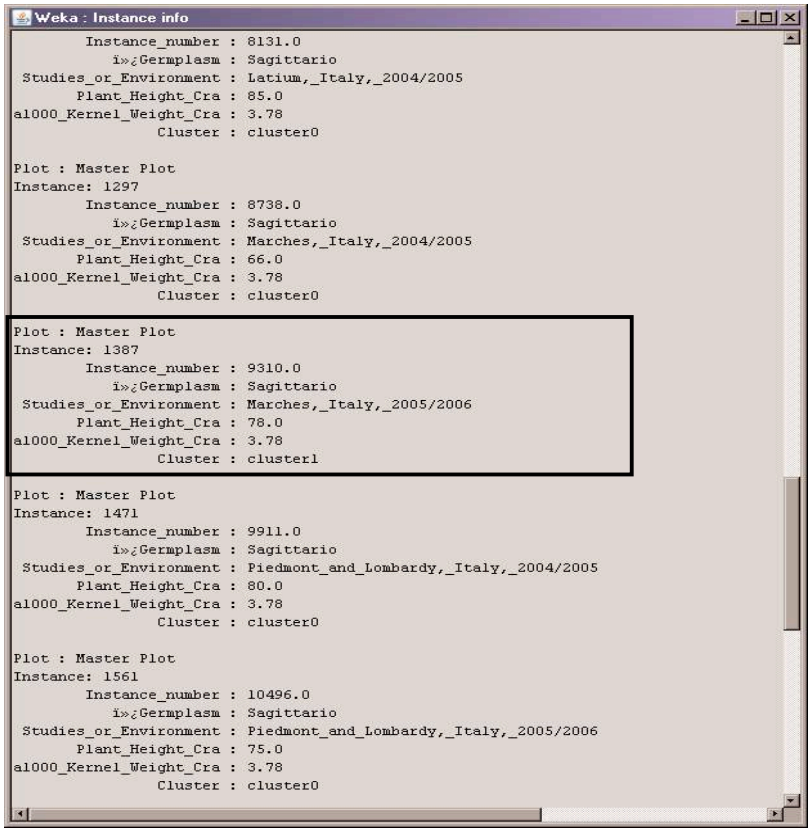

Figura 24 - Presunto valore anomalo del Germplasm Sagittario

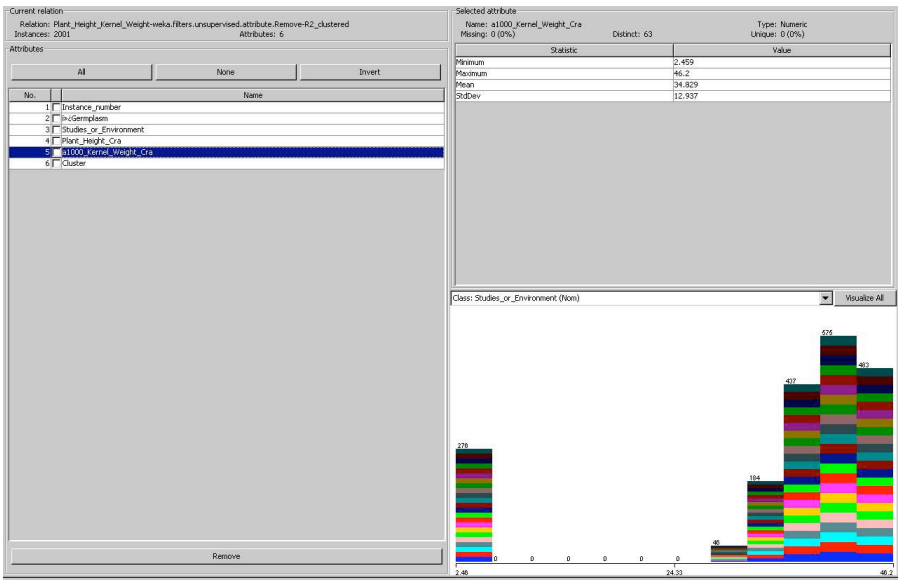

Figura 25 – Distribuzione della provenienza dei dati di a1000_Kernel_Weight evidenziata dai colori

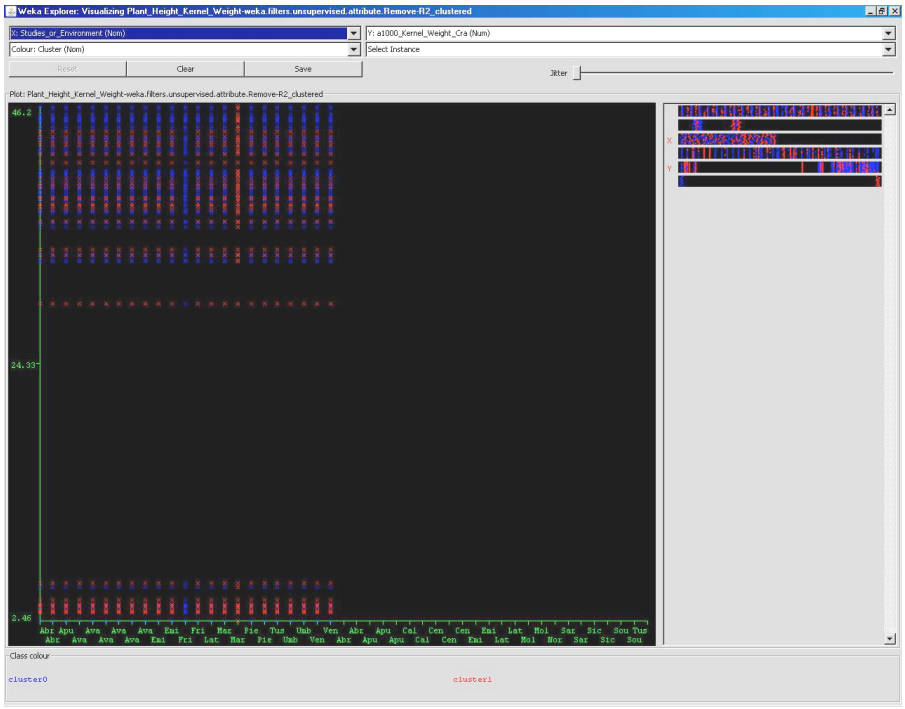

Figura 26 – Grafico a1000_kernel_weight-Studies_or_Environment

3.2.2.2 Specie Bread Wheat, attributo `PlantHeight`

Poiché il clustering ha coinvolto tutti i dati numerici, si è provveduto a salvare i risultati ottenuti (ovvero le partizioni effettuate dall'algoritmo) in un file Arff che successivamente è stato di nuovo aperto per concentrarsi sul nuovo attributo. In figura 27 è riportato il grafico della distribuzione dei valori dell'attributo `PlantHeight` con relativa colorazione dei due cluster. In questo caso la distribuzione dei punti del cluster 0 è piuttosto uniforme mentre quella dei punti del cluster 1 è localizzata maggiormente attorno a zone ben più precise, andando però a consultare i valori numerici non si nota alcun andamento degno di nota. Inoltre, impostando come asse delle ascisse l'attributo `Studies_or_Environment_CRA` si scopre che ancora l'algoritmo non ha evidenziato niente di particolarmente interessante: i dati però, per come risultano dalla query originaria, sono distribuiti in un modo più interessante, indipendentemente dalla scelta fatta dall'algoritmo: compaiono valori più alti e più bassi che selezionati appositamente in aree rettangolari con il mouse tramite l'opzione Select Instances ci danno contemporaneamente le provenienze dei

valori più alti e più bassi: Abruzzo, Puglia, Basilicata, Campania, Toscana e Marche hanno valori non molto elevati, Emilia Romagna, Veneto, Piemonte e Lombardia, Friuli con un valore elevatissimo di 105.5 per il germplasm Anapo invece presentano i valori più elevati.

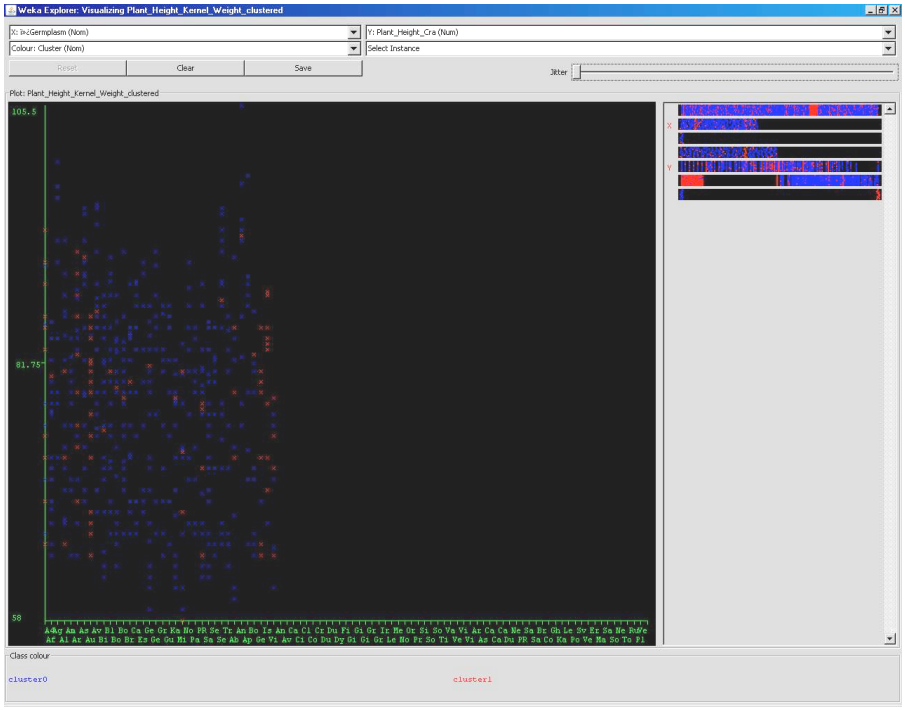

Figura 27 – Clustering sull'attributo Plant_Height

3.2.2.3 Specie Durum Wheat, attributo `a1000_Kernel_weight`

La ricerca di cluster rispetto all'attributo `a1000_Kernel_Weight_CRA` non ha dato alcun riscontro di interesse, i risultati sono analoghi a quelli appena descritti per la specie Bread Wheat. L'unico grafico che mostra scarse informazioni è quello con ascissa `Studies_or_Environment_CRA` e ordinata `a1000_Kernel_Weight`: la colorazione dei cluster è più diffusa con il colore rosso nei punti corrispondenti alle zone del sud Italia e, viceversa, ci sono colorazioni completamente blu che corrispondono alle zone d'Italia più a nord (Sardegna compresa), null'altro si può valutare perché la distribuzione dei punti rispetto è simile per tutte le zone d'Italia. Infine si nota solamente che,

indipendentemente dalla provenienza, un unico `Germplasm` ha valori elevati corrispondenti a 69.04 (ovvero il massimo valore presente nella base di dati): il Duetto.

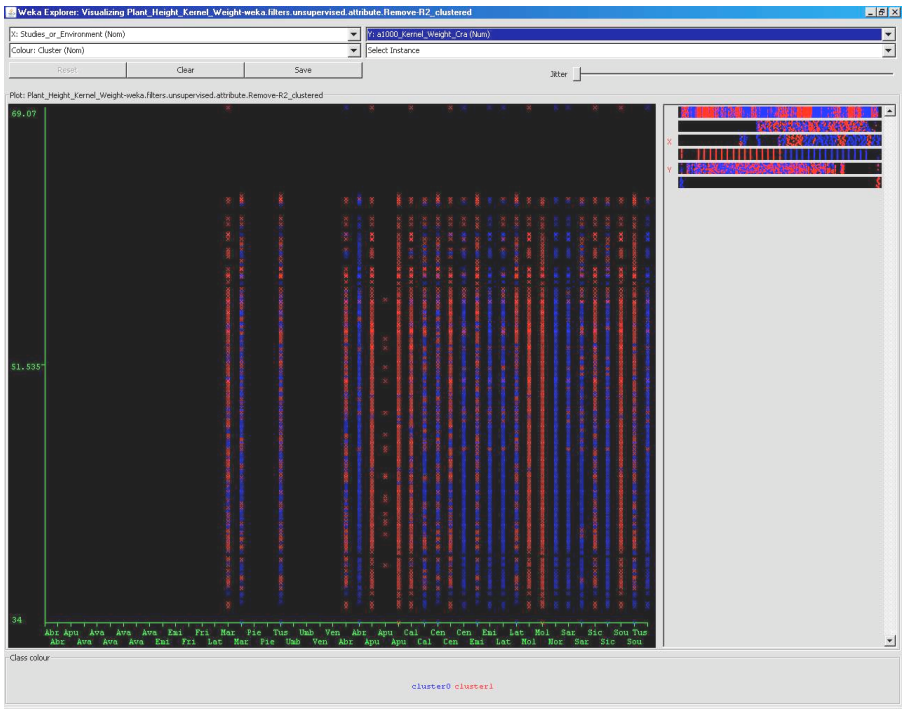

Figura 28 – Risultati di clustering per l'attributo a1000_Kernel_Weight

3.2.2.4 Specie Durum Wheat, attributo Plant_Height

K-means ha evidenziato un comportamento particolare per i valori di `Plant_Height`. Si è impostata inizialmente la ricerca di due cluster e un particolare è stato subito messo in evidenza: l'algoritmo divide sempre i punti in due partizioni e lo fa sempre allo stesso modo nonostante vengano scelti seed diversi. Nel caso venga impostata la ricerca di quattro cluster, i punti sono divisi in quattro partizioni uguali che altro non sono che la divisione a metà delle due partizioni trovate inizialmente. Anche visualizzando il grafico rispetto all'ascissa `Studies_or_Environment` si vede che la divisione è netta e analoga. In particolare, per quest'ultimo attributo si nota che l'algoritmo ha calcolato i cluster rispetto al valore medio dell'altezza, ovvero il valore 80 verificabile andando a consultare la sezione Preprocess di Explorer; attorno a tale valore pressoché tutti i germplasm sono presenti in tutte le zone d'Italia riportate nella base di dati. Infine

si nota che molti `Germplasm` provenienti dal centro e sud Italia hanno i valori tendenzialmente più bassi, mentre nel nord Italia (salvo eccezioni) vi sono piante con altezza maggiore; le eccezioni sono rappresentate da Abruzzo nell'anno 2005/2006, Sardegna, Sicilia 2005/2006, Italia centrale sponda tirrenica 2005/2006, Lazio e Umbria 2004/2005.

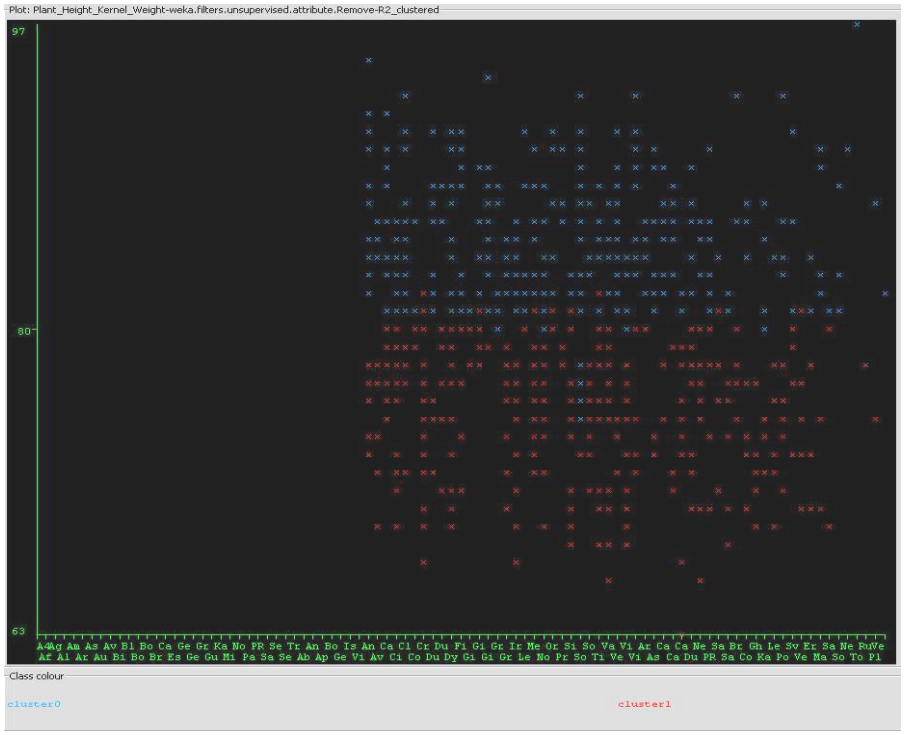

Figura 29 – Partizionamento di k-means nel grafico Plant_Height-Germplasm

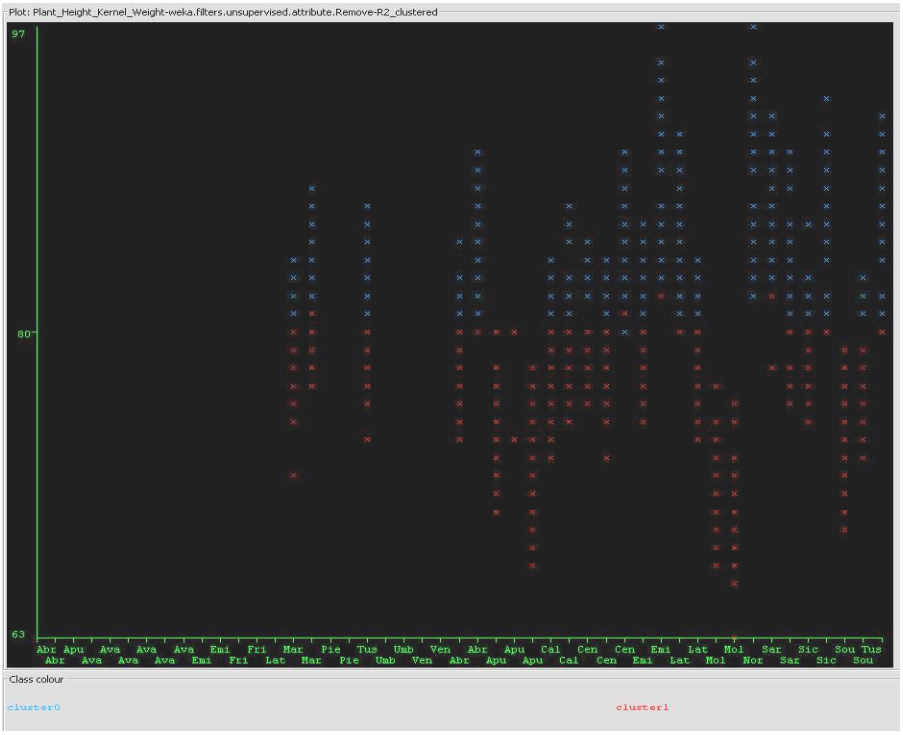

Figura 30 – Partizionamento di k-means nel grafico Studies_or_environment-Plant_Height

3.3 Query su attributi a1000_Kernel_Weight, Fusarium_Damaged_Kernels, Fusarium_Head_Blight_Scab

```
SELECT F1.Germplasm,
       F2.Species,
       F1.Studies_or_Environment_CRA,
       F2.Fusarium_Damaged_Kernels_CRA,
       F1.Fusarium_Head_Blight_Scab_CRA,
       a1000_Kernel_Weight_Cra

FROM   FHB_CRA F1, Fusarium_Damaged_Kernels_CRA F2,
       a1000_kernel_weight_Cra G

WHERE  F2.Germplasm=F1.Germplasm
  AND  F2.Germplasm=G.Germplasm
```

Gli attributi da confrontare hanno il seguente significato:

- `Fusarium_Head_Blight_Scab`: indice di resistenza all'attacco di malattie da funghi patogeni "Fusarium graminearum" appartenenti al genere Gibberella zeae,
- `Fusarium Damaged Kernels`: percentuale gusci danneggiati da attacco da funghi trovati in un campione di semi da un chilogrammo;
- `Plant Height`: misura in centimetri dell'altezza della pianta conteggiata dal suolo fino al punto più alto che essa raggiunge.

Per questa query i risultati disponibili sono solo quelli riguardanti la specie Durum Wheat, inoltre solo due sono i valori dell'attributo `Studies_or_Environment`: Emilia Romagna/Lombardia e Lazio/Umbria.

3.3.1 Apertura file in Weka Explorer

Il grafico più consono che viene mostrato distinguendo i punti per provenienza è quello con ascisse `Fusarium_Head_Blight` e ordinata `Fusarium_Damaged_Kernels`, inoltre non vi sono una quantità notevoli di punti distribuiti lungo gli assi. Conviene a questo proposito intervenire sul selettore di Jitter che mostra la quantità di punti che si trovano a un determinato valore, in tal modo è possibile avere una idea più evidente della distribuzione. Questa visualizzazione dei punti è solo per scopo visivo e serve per avere una idea generale dell'andamento dei valori.

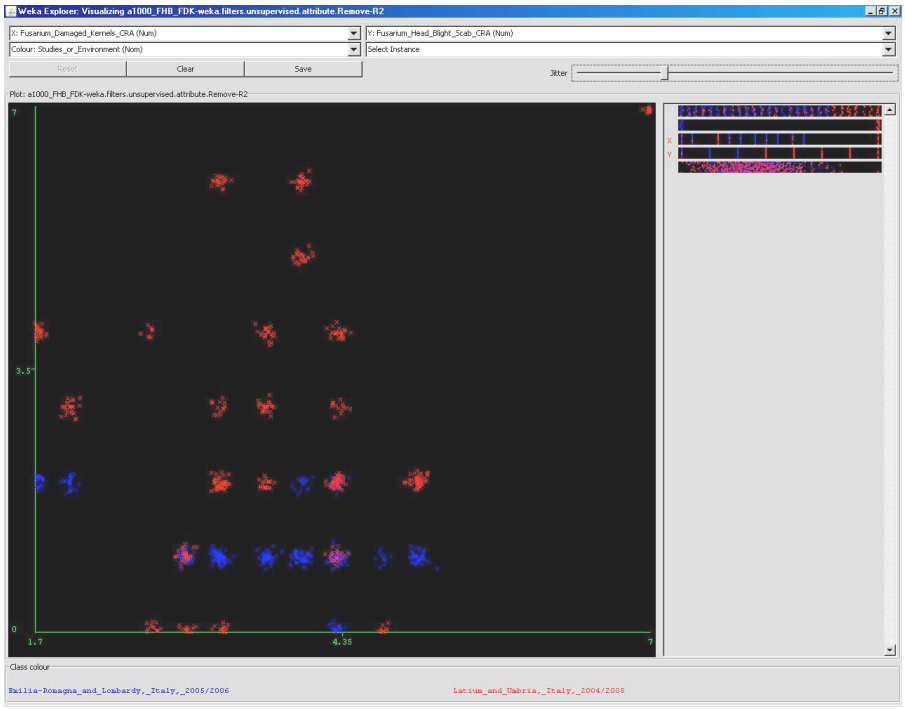

Figura 31 – Fusarium_Head_Blight e Fusarium_Damaged_Kernels distinti per provenienza con visualizzazione Jitter

Come mostrato in figura 31 la maggior parte dei punti colorati di rosso hanno valori tendenti verso l'alto e appartengono agli studi fatti in centro Italia (Lazio e Umbria 2004/2005), mentre quelli con valori tendenzialmente più bassi sono quelli del nord Italia (Emilia Romagna e Lombardia 2005/2006). In particolare, il germplasm Sorriso ha i valori più elevati per entrambi gli attributi corrispondenti a 7.0. Per quanto riguarda invece l'attributo `a1000_Kernel_Weight` il grafico mostra che la collocazione dei punti è ben distribuita e non ha un andamento degno di nota per quanto riguarda la provenienza, quest'ultima ha una distribuzione dei valori normale e si può vedere direttamente nella sezione Preprocess con l'usuale grafico a barre riassuntivo. Selezionando i valori del grafico minimi e massimi dell'attributo `a1000_Kernel_Weight` e impostando gli assi con gli attributi di tipo `Fusarium` non si nota nulla di particolarmente significativo, tranne per il germplasm Sorriso che ha valori molto alti per i due attributi `Fusarium` ma presenta un intervallo ampio di valori rispetto all'altezza raggiunta della pianta; tutti i restanti punti non hanno una distribuzione concentrata in particolari zone di interesse.

3.3.2 Ricerca di cluster

Con la ricerca di due cluster si è ottenuto un partizionamento dei punti che segue in modo fedele la provenienza, ovvero i due raggruppamenti ottenuti ricalcano quasi perfettamente la classificazione fatta rispetto all'attributo `Studies_or_Environment` di figura 31. Impostando invece a quattro la ricerca dei cluster si ottiene una distinzione solo per valori alti, medi e bassi per i due attributi di tipo Fusarium. Non è stato notato nulla di particolare nel cluster trovati, non vi sono concentrazioni di punti che possono essere interessanti e questo ha portato a concludere che tra questi tre attributi non vi sia una forte relazione: l'altezza della pianta non viene influenzata in particolar modo dagli indici di attacchi di funghi sebbene questi ultimi siano definiti in intervalli di valori precisi.

3.4 Query su attributi Plant_Height, Grain_Yield, Spike_Density

```
SELECT P.Germplasm
       P.Species,
       P.Studies_or_Environment_CRA,
       P.Plant_Height_Cra,
       S.[Spike_Density_(spike/m2)_CRA] as
         Spike_Density,
       G.Grain_Yield_Cra

FROM Plant_Height_Cra P, Spike_Density_Cra S,
     Grain_Yield_Cra G

WHERE P.Germplasm=S.Germplasm
  AND P.Germplasm=G.Germplasm
  AND P.Studies_or_Environment_CRA =
      S.Studies_or_Environment_CRA
```

```
  AND P.Studies_or_Environment_CRA =
      G.Studies_or_Environment_CRA
```

Significato degli attributi da confrontare:

- `Grain Yield`: indice di resa dei cereali, misurato in chilogrammi per ettaro al 14% di umidità.
- `Spike Density`: Numero di spighe per metro quadro
- `Plant Height`: Misura in centimetri dell'altezza della pianta conteggiata dal suolo fino al punto più alto che essa raggiunge

Anche per questa query le tuple ottenute fanno riferimento solamente alla specie Durum Wheat.

3.4.1 Ricerca di cluster

Con la ricerca di tre cluster, il risultato che si ottiene è abbastanza interessante e mostra una tendenza nei valori piuttosto semplice da descrivere. Si riporta di seguito il risultato testuale dell'algoritmo:

```
=== Run information ===

Scheme:       weka.clusterers.SimpleKMeans -N 3 -S 10
Relation:     PlantHeight_SpikeDensity_GrainYield
Instances:    442
Attributes:   6
              Plant_Height
              Spike_Density
              Grain_Yield
Ignored:
              Germplasm
              Species
              Studies_or_Environment
Test mode:    evaluate on training data
```

```
=== Model and evaluation on training set ===

kMeans
======

Number of iterations: 9
Within cluster sum of squared errors:
20.798786397696965

Cluster centroids:

Cluster 0
     Mean/Mode:   89.2121 650.9242   7.1414
     Std Devs:     3.6269  58.118    0.4343
Cluster 1
     Mean/Mode:   77.0171 337.9429   4.303
     Std Devs:     4.3727  44.1429   0.7194
Cluster 2
     Mean/Mode:   84.2488 416.3184   6.1187
     Std Devs:     4.3125  60.094    0.6254



Clustered Instances

0       66 ( 15%)
1      175 ( 40%)
2      201 ( 45%)
```

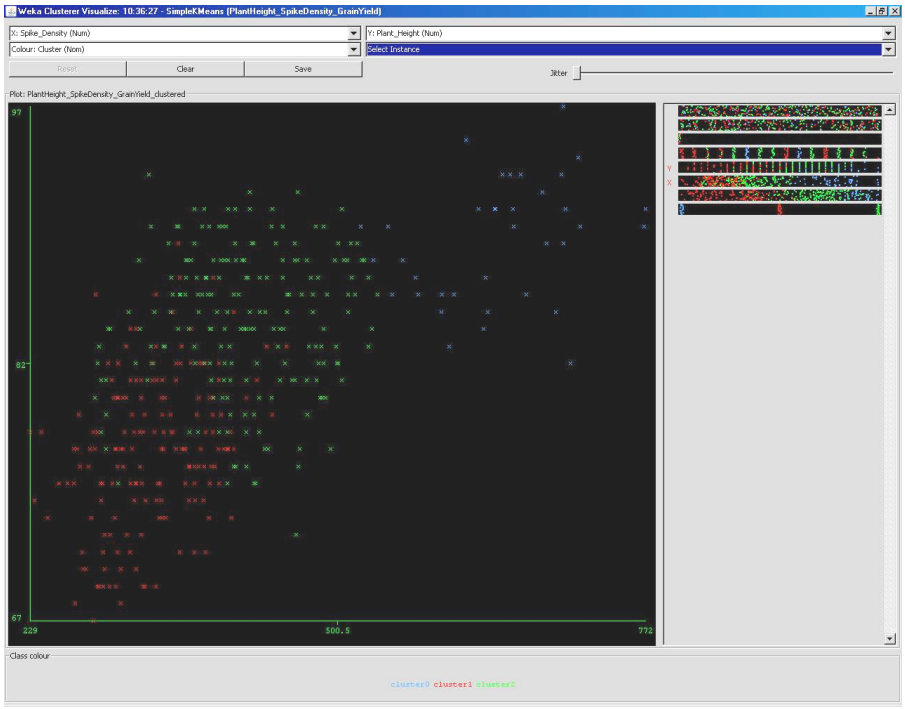

Figura 32 – Clustering su Spike Density e Plant Height

Nel grafico di figura 32 si nota a colpo d'occhio che non vi è una gran quantità di punti con i valori molto elevati (colorati in blu). Selezionando tale area corrispondente al cluster utilizzando la voce Polygon (ovvero delimitando i punti del grafico con un poligono che individua la zona interessata) e cambiando le ascisse l'attributo Studies_or_Environment il risultato si ottengono sia su `Plant_Height` che su `Spike_Density` solamente punti corrispondenti all'Emilia Romagna e Nord Italia, tranne qualche rarissimo valore per Lazio e Umbria e Italia Centrale.

Tornando ancora al grafico iniziale di figura 31 si è ripetuta la selezione delle altre due zone corrispondenti alla suddivisione dei cluster: il cluster con i punti colorati in giallo risultano quasi tutti assegnati a valori di `Studies_or_Environment` riguardanti le regioni Marche, Toscana, Sardegna, Italia Centrale costa tirrenica, Lazio e Umbria. Il cluster con colorazione rossa ha assegnato i punti alle regioni Abruzzo, Molise, Calabria, Sicilia, Puglia.

L'algoritmo, in sostanza, ha mostrato una suddivisione dei valori numerici in tre intervalli che possono essere associati fortemente alle regioni del nord, centro e sud Italia.

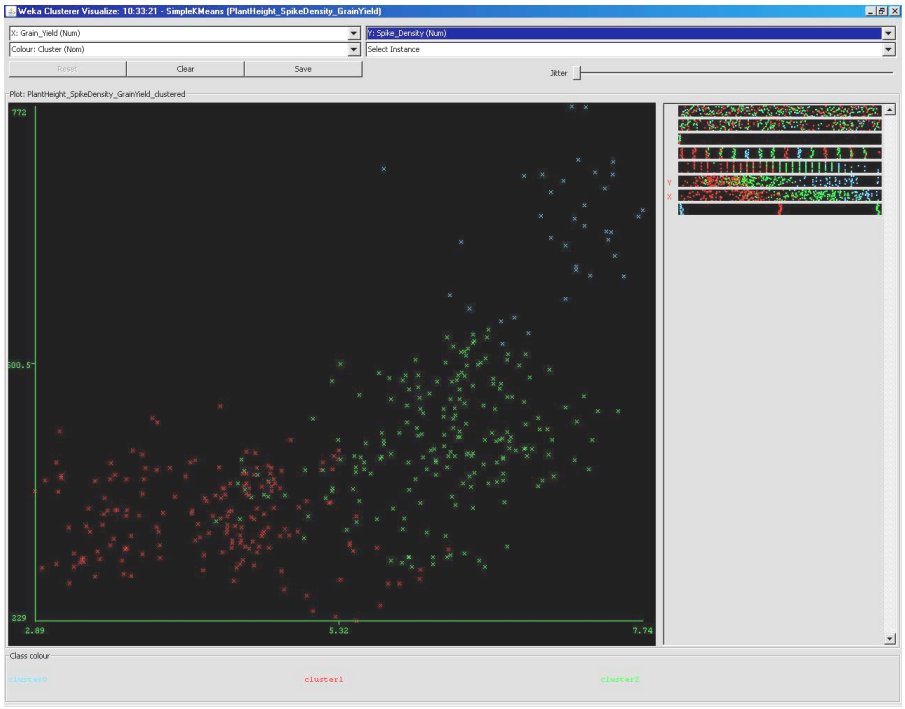

Figura 33 – Clustering su Grain Yield e Spike Density

Il partizionamento dei punti mostrato in figura 33 con ascissa e ordinata impostate rispettivamente sugli attributi `Grain_Yield` e `Spike_Density` ha mostrato ancora un andamento fortemente legato alle regioni di provenienza. Infatti i punti con valori più alti per entrambi gli attributi corrispondono anche questa volta a zone del nord Italia con qualche leggera presenza di Lazio e Umbria, a seguire i restanti due cluster danno risultati riguardanti centro Italia per i valori intermedi e sud Italia per i valori più bassi. Se si guarda infine il grafico con attributi `Plant_Height` e `Grain_Yield` ancora una volta si ottiene un comportamento analogo.

La scelta di trovare tre cluster si è rivelata giusta, l'algoritmo ha mostrato risultati da considerarsi soddisfacenti, la stessa distribuzione numerica dei punti ha favorito l'individuazione di cluster ben delimitati. Si è provato a impostare la ricerca su due cluster e poi su sei cluster ma si sono ottenuti partizionamenti senza alcuna particolarità da segnalare.

3.5 Considerazioni finali

Pochi sono stati i risultati interessanti, per gli attributi presi in esame si è visto che non vi sono particolari comportamenti se non per una leggera dipendenza dei valori rispetto alla provenienza geografica d'Italia: l'algoritmo di clustering ha fornito risultati soddisfacenti e significativi solo per questo caso specifico. I restanti risultati ottenuti dimostrano che non vi sono particolari correlazioni o concentrazioni di valori da tenere in considerazione.

3.6 Elaborazioni escluse dalla ricerca

Non è stato possibile creare una query valida che includesse tutti gli attributi esaminati per un trovare nuove corrispondenze più o meno nascoste tramite clustering k-means. Purtroppo la mancanza di valori di chiavi e la stessa mancanza di un numero elevato di tuple in alcune tabelle non ha reso possibile un confronto di questo tipo. Per alcune tabelle erano solo presenti solamente tuple con specie Durum Wheat, per altre si è visto che gli attributi `Germplasm` e `Studies_or_Environment` sono totalmente differenti e rendono improponibile, a parità di attributi, un confronto tra le caratteristiche fenotipiche presenti dalla base di dati. Non è nemmeno stato possibile applicare algoritmi diversi dal clustering, i tentativi fatti per trovare alberi di decisione o regole associative hanno dato rispettivamente risultati caotici e addirittura risultati nulli.

4. Data Mining su database Graingenes

4.1 Utilizzo dell'interfaccia Knowledge Flow

Allo scopo di mostrare una diversa impostazione del lavoro rispetto all'interfaccia Explorer, la ricerca sui dati di Graingenes è stata effettuata tramite la modalità di lavoro Knowledge Flow. È possibile fare tutto ciò che fa Explorer ma in un modo alternativo: il lavoro sui dati viene interamente preparato a priori tramite la creazione di un grafo che ha sempre un nodo di inizio e uno o più nodi finali: i nodi sono rappresentati da icone e il flusso dei dati da frecce. Le icone rappresentano il tipo di elaborazione da effettuare: per esempio apertura e/o salvataggio di un file di dati, algoritmi da applicare, rappresentazioni grafiche dei dati caricati o elaborati da uno specifico algoritmo, risultati in forma testuale ed eventuali filtri. Le freccie collegano le icone in base al tipo di elaborazione e al tipo di risultato che si vuole ottenere.

Un generico percorso di un grafo creato in Knowledge Flow comincia con il caricamento dei dati a cui segue l'elaborazione tramite gli algoritmi e infine termina con la visualizzazione testuale o grafica dei risultati. Una volta che il grafo è stato impostato si fa partire l'intero processo di elaborazione cliccando sull'icona che rappresenta il caricamento dei dati: Weka provvederà a elaborare in maniera sequenziale tutti i passi rappresentati dalle icone e per le eventuali doppie o triple diramazioni l'elaborazione sarà in parallelo.

Questo modalità di lavoro è una alternativa che viene incontro a chi predilige la rappresentazione schematica del lavoro da effettuare e che vuole avere sott'occhio in modo chiaro e sobrio tutti i passaggi. Oppure può essere utile nel caso si voglia effettuare una serie di elaborazioni standard da applicare a più dati simili. Lo svantaggio è dato dal "percorso obbligato" del grafo che necessita, per qualsiasi tipo di modifica, la riesecuzione dell'intero ciclo di elaborazioni (a differenza di

Explorer che permette piena libertà di scelta sul tipo di azioni da intraprendere in qualsiasi momento). Attualmente KnowledgeFlow è ancora in fase di sviluppo e non riporta tutti gli algoritmi presenti in Explorer come ad esempio Apriori, inoltre a volte non è possibile effettuare connessioni tra icone affini, tuttavia questi problemi non hanno ostacolato il lavoro sui dati di Graingenes.

4.2 Descrizione delle fasi di lavoro

- Formulazione query in Sql Server 2000 o MySQL e salvataggio risultati in file di testo con formato CSV;
- Apertura file di dati corrispondente in Weka Explorer;
- Selezione di attributi rilevanti tramite algoritmo Information Gain in Weka Explorer (attualmente non è previsto in Knowledge Flow);
- Impostazione in Knowledge Flow del grafo di lavoro;
- Applicazione dell'algoritmo di clustering Expectation-Maximization;
- Applicazione dell'algoritmo J48 (chiamato anche C45) per creare alberi di decisione;
- Conversione di tutti gli attributi in etichetta di testo;
- Applicazione dell'algoritmo Apriori in Weka Explorer;
- Considerazioni generali sulle elaborazioni.

Sebbene il database Graingenes sia molto vasto, è stato più volte riscontrata l'impossibilità di formulare query che coinvolgano 4 o più tabelle perché non esistevano tuple associate alle chiavi di ricerca. Altre volte le tuple risultanti erano in quantità inadeguata (alcune volte una o al massimo due sole tuple ottenute) e questa forte limitazione ha tolto la possibilità di creare query interessanti. I parametri utilizzati nei vari algoritmi sono in gran parte quelli di default e verranno modificati in base ai risultati ottenuti dalle query e alla importanza degli attributi rilevanti trovati. Nel caso di Apriori gli attributi con valori non numerici devono essere convertiti come etichette testo (ovvero considerati come stringhe di caratteri) per essere accettate dall'algoritmo.

4.2.1 Grafo di lavoro in Knowledge Flow

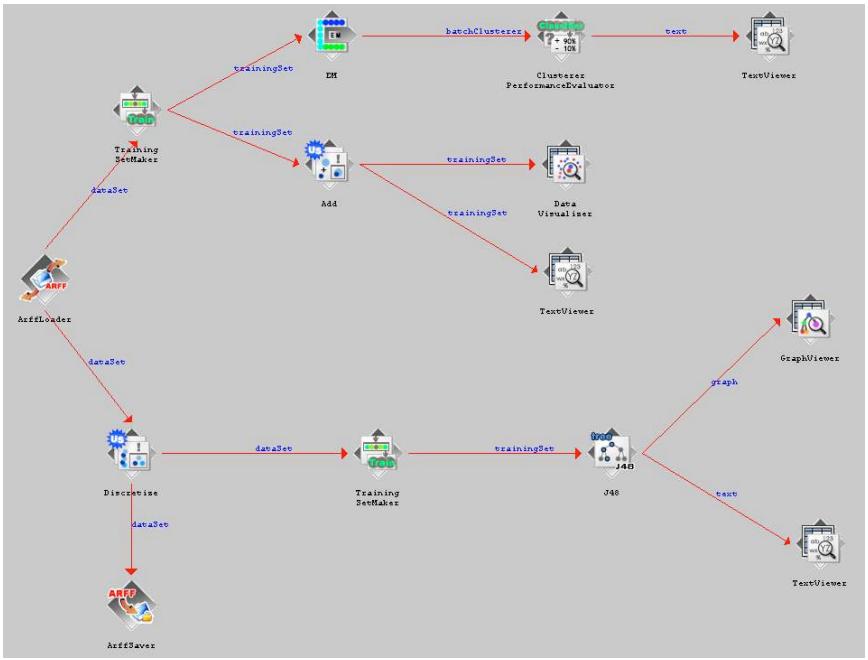

Figura 34 – Grafo di lavoro creato in Knowledge Flow

In Knowledge Flow è stato creato il grafo da applicare a tutte le query, l'elaborazione dei risultati delle query inizia con l'apertura del file (icona all'estrema sinistra di figura 34), successivamente il lavoro si svolge in "parallelo" e vengono applicati gli algoritmi di clustering e di albero di decisione.

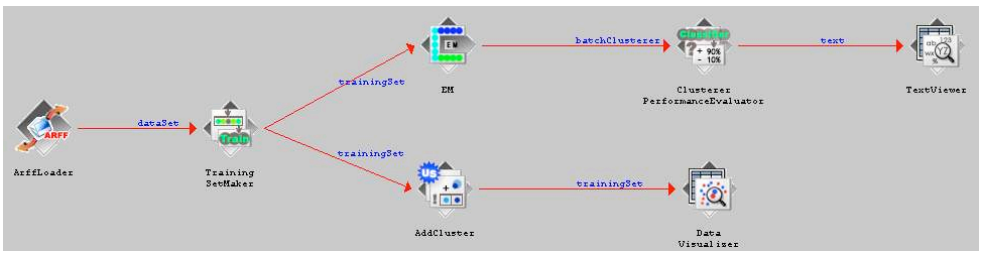

Figura 35 – Ramo del grafo riguardante il Clustering E-M

Il percorso fatto dal ramo riguardante il clustering E-M si svolge nel seguente modo (vedi figura 35):

1. Caricamento del file di query in formato csv;

2. I dati caricati vengono impostati come training set per le due successive le ramificazioni, corrisponde alla voce “Use training set” utilizzata in Explorer;
3. I dati di training set vengono instradati all’icona EM che rappresenta l’algoritmo di clustering E-M, i risultati vengono raccolti e preparati per l’output dall’icona Classifier e mostrati tramite TextViewer che riporta l’output in formato testo della elaborazione;
4. L’icona AddCluster crea un attributo da aggiungere ai dati originari di training set per poter rappresentare la classificazione di dati in base ai cluster creati dall’algoritmo;
5. I risultati del partizionamento vengono uniti ai dati originari e visualizzati graficamente tramite l’icona DataVisualizer, essa è l’equivalente della sezione Visualize di Explorer

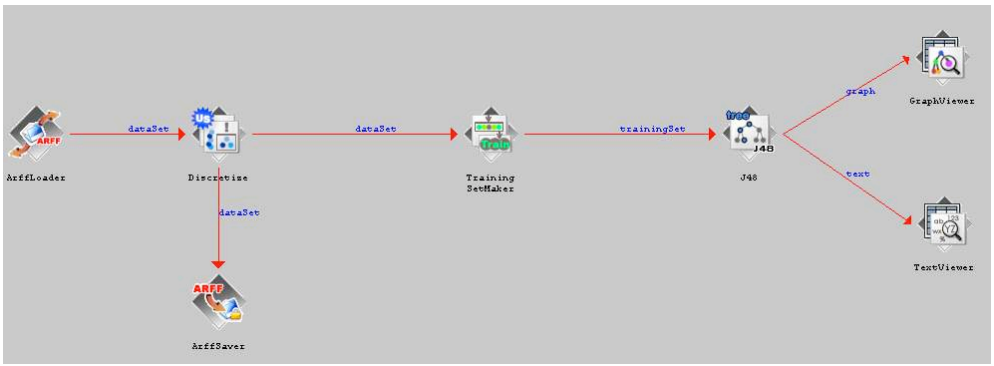

Figura 36 – Ramo del grafo riguardante l’albero di decisione J48

Il percorso fatto dal ramo riguardante il clustering E-M si svolge nel seguente modo (vedi figura 36):

1. Caricamento del file di query in formato csv;
2. L’icona Discretize prende i dati in input e li converte in etichette di testo;
3. Salvataggio del nuovo file in formato Arff con gli attributi convertiti, servirà successivamente per riutilizzare i dati per l’algoritmo Apriori in Explorer;
4. I dati vengono impostati come training set;

5. Creazione dell'albero di decisione simboleggiato dall'icona J48;
6. Icona TextViewer: raccoglie, tramite le frecce a cui fa capo, le informazioni in formato testo (numeri, etichette) dell'output instradato dalle elaborazioni del punto precedente;
7. GraphViewer: visualizza l'albero di decisione ottenuto.

4.3 Selezione di attributi rilevanti

I risultati delle query sul database Graingenes spesso compaiono con uno o più attributi che riportano in diverse tuple valori null, data però la quantità di tuple risultanti non è facile individuare a vista quali di queste presentano i valori: per ovviare è stata utilizzata una funzione di Weka per la selezione di attributi ovvero il metodo basato sul guadagno di informazione.

4.3.1 Entropia e guadagno di informazione

Sia D l'insieme di dati su cui effettuare la selezione di attributi e se ne prenda uno da confrontare con i restanti, l'attributo scelto ha una serie di m valori distinti che definiscono m classi distinte C_i (con i=1,…, m) e sia $C_{i,D}$ l'insieme di tuple di classe C_i in D; infine con |D| e $|C_{i,D}|$ indichiamo il numero di tuple di D e di $C_{i,D}$ rispettivamente.
L'entropia (chiamata anche semplicemente *informazione su D*) è data da:

$$E(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

dove p_i è la probabilità che una determinata tupla di D appartenga alla classe C_i ovvero $|C_{i,D}|/|D|$. E(D) è la quantità media di informazione che serve per identificare una classe di tuple di D. Si consideri poi un qualsiasi altro attributo A di D con v distinti valori $\{a_1, a_2,\ldots,a_v\}$ che divide l'insieme D in v partizioni o sottoinsiemi $\{D_1, D_2,\ldots,D_v\}$ contenenti le tuple di D che hanno valore a_j di A e si calcoli

$$E_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times E(D_j)$$

Il termine $|D_j|/|D|$ è il peso della partizione j-esima di D. $E_A(D)$ rappresenta l'informazione necessaria per classificare una tupla di D in base al partizionamento dell'attributo A. Più piccolo è il valore di E_A e più grande la partizione contiene valori uniformi.
Infine il guadagno di informazione si ottiene come differenza tra E(D) ed $E_A(D)$, ovvero

$$Guadagno(A) = E(D) - E_A(D)$$

In altre parole, per ogni attributo il guadagno corrispondente dà una valutazione sulla rilevanza dell'attributo stesso rispetto a quello scelto in partenza, più alto è il valore trovato più alta sarà la rilevanza.
L'algoritmo è supervisionato perché si definisce a priori il valore di information gain che serve per dividere gli attributi validi da quelli non soddisfacenti; si è scelto il valore 1 come discriminante e nei casi in cui vi siano molti attributi scartati (o con valori molto inferiori all'unità) si provvederà a una modifica di tale valore in base ai risultati del guadagno.

4.3.2 Implementazione in Weka

In Weka, l'impostazione di tale funzione di selezione avviene nella sezione Select Attributes scegliendo InfoGainAttributeEval alla voce Attribute Evaluator e Ranker alla voce Search Method.

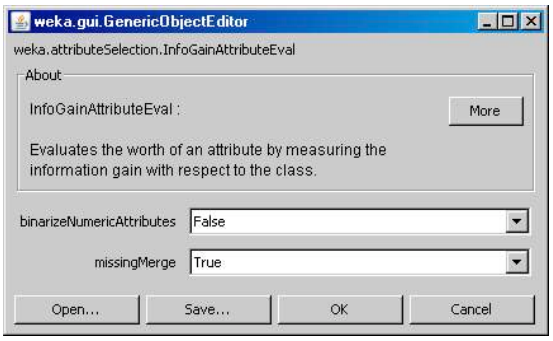

Figura 37 – Finestra Information Gain

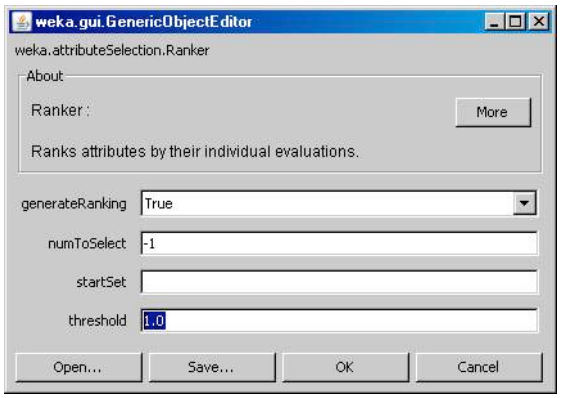

Figura 38 - Finestra Ranker

4.4 Algoritmo Apriori

Weka attualmente è in grado di utilizzare efficacemente soltanto regole interdimensionali, quelle che coinvolgono più attributi, che occorrono però una sola volta nella regola, l'algoritmo disponibile è il classico Apriori: per trovare regole di associazione dato un insieme di dati in ingresso, l'algoritmo cerca di trovare sottoinsiemi che siano presenti nell'insieme delle transazioni almeno S volte (con S supporto definito dall'utente). Apriori usa un approccio "bottom up" in cui i sottoinsiemi frequenti sono costruiti aggiungendo un item per volta (generazione dei candidati), i gruppi di candidati sono successivamente testati sui dati, infine l'algoritmo termina quando non ci sono più itemset frequenti. Gli itemset candidati di lunghezza k sono generati a partire da itemset di lunghezza k-1, successivamente si scartano i candidati non frequenti.

4.4.1 Formulazione dell'algoritmo

I passi fondamentali eseguiti dall'algoritmo sono:

1. Trova gli insiemi frequenti di oggetti (F.I., frequent itemset):
   - Devono soddisfare il vincolo sul supporto
   - Un sottoinsieme di un F.I. è a sua volta un F.I.: se l'item set {A,B} è frequente allora anche {A} e {B} lo sono, con supporto maggiore
   - Iterativamente trova gli F.I. con cardinalità da 1 a k
2. Usa gli F.I. per generare regole associative

Vengono riportati in forma di pseudocodice, i passi iterativi dell'algoritmo:

C_k: itemset candidato di dimensione k
L_k : itemset frequente di dimensione k

```
L1 = {item frequenti};
for (k = 1; Lk !=∅; k=k+1) do
        begin
      Ck+1 = candidati generati da Lk;
      for each transazione t in database do
           incrementa il conteggio di candidati in Ck+1
           che sono contenuti in t
      Lk+1 = candidati in Ck+1 con min_support
        end
return ⋃k Lk
```

4.4.2 Generazione dei candidati

Supponendo gli item in L_{k-1} ordinati:

Passo 1: Self-joining L_{k-1} tramite query

```
insert into Ck
select p.item1, p.item2, …, p.itemk-1, q.itemk-1
from Lk-1 p, Lk-1 q
where p.item1=q.item1, …, p.itemk-2=q.itemk-2,
      p.itemk-1 < q.itemk-1
```

Passo 2: pruning

```
For all itemsets c in Ck do
        For all (k-1)-subsets s of c do
         if (s is not in Lk-1) then delete c from Ck
```

4.4.3 Supporto e confidenza

Per l'algoritmo Apriori, i valori di confidenza e supporto sono fondamentali per determinare una soglia entro cui creare regole.
Dato un insieme di item I, un insieme di transazioni D e una regola associativa del tipo $X \Rightarrow Y$ (X *implica* Y) con gli attributi $X,Y \subset I$ e $X \cap Y = \varnothing$:

- si definisce supporto s di $X \Rightarrow Y$ se una frazione pari a delle transazioni contengono tutti gli item in $X \cup Y$;
- si definisce confidenza c di $X \Rightarrow Y$ se una frazione pari a c delle transazioni in cui compare X contiene Y.

4.4.4 Implementazione in Weka

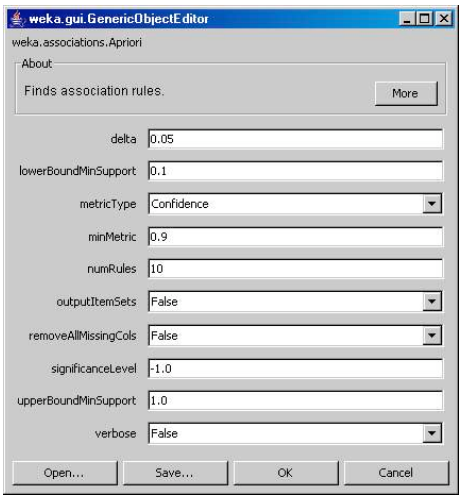

Figura 39 – Finestra Apriori

In Weka l'algoritmo si imposta in Explorer nella sezione Associate, la finestra che compare permette di selezionare diverse modalità per la ricerca di regole associative. Cruciali sono la soglia di confidenza (minMetric) e il numero di regole da daterminare (numRules): l'algoritmo infatti opera in modo da determinare numRules regole che superano la soglia di confidenza impostata. Le soglie per il valore minimo e massimo di supporto non sono fissate: il sistema parte da una certa soglia massima (upperBoundMinSupport) normalmente impostata a 1, e scende gradatamente verso una soglia minima (lowerBoundMinSupport) a passi impostati dal parametro delta. Per ogni valore di soglia per il supporto, il sistema tenta di determinare numRules regole che superino la confidenza minima: se ci riesce si ferma e visualizza le regole migliori trovate, altrimenti diminiusce il supporto minimo corrente del valore delta e ci riprova.

4.5 Clustering con algoritmo Expectation-Maximization

Tramite questo algoritmo iterativo (abbreviato semplicemente in E-M) ogni cluster viene identificato e rappresentato (nel set di dati in ingresso) in base alla sua distribuzione di probabilità. Può essere visto come una estensione della

versione k-means (che assegna un elemento a un cluster con cui vi è maggiore similarità) : E-M assegna un elemento a un cluster in base al peso ovvero alla probabilità di appartenere a uno o più cluster vicini (non ci sono confini definiti), il funzionamento è distinto in due parti, la prima di Expectation Step e la seconda di Maximization Step.

1. L'algoritmo inizia selezionando k punti casuali di partenza nello spazio dei dati (come in k-means)
2. Raffinamento dei valori ripartita in due fasi:
    a. Fase Expectation: assegna ogni elemento x_i al cluster C_k che ha probabilità

$$P(x_i \in C_k) = p(C_k \mid x_i) = \frac{p(C_k)p(x_i \mid C_k)}{p(x_i)},$$

    dove $p(C_k \mid x_i) = N(m_k, E_k(x_i))$ rappresenta la distribuzione normale (gaussiana) attorno alla media m_k con valore atteso E_k. In questa fase viene calcolata la prorabilità di appartenenza a un cluster di un elemento x_i per ognuno dei cluster, ovvero l'algoritmo "presume" che il punto appartenenga a quel cluster.
    b. Fase Maximization: massimizza le distribuzioni di probabilità calcolate nel punto precedente assumendo che queste rappresentino anche i dati mancanti nell'insieme di dati iniziale.

4.5.1 Implementazione in Weka

L'implementazione in Weka di questo algoritmo consente di selezionare il numero massimo di iterazioni e permette anche di non scegliere il numero di

cluster da trovare da parte dell'utente: impostando il valore "-1" nel campo numClusters essi verranno calcolati automaticamente tramite cross validation sui dati.

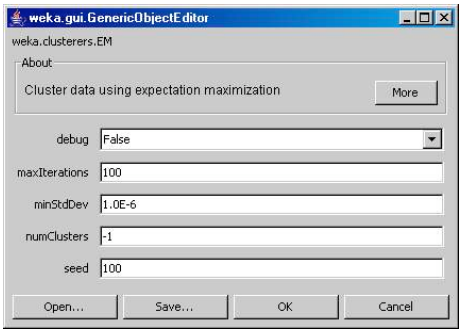

Figura 40 – Finestra per clustering Expectation-Maximization

4.6 Algoritmo J48 per gli alberi di decisione

L'algoritmo è strutturato nei seguenti passi:

1. si utilizza un sottoinsieme scelto casualmente dei pattern del training set (chiamato "finestra") come training set;
2. si utilizza la finestra corrente per la costruzione di un albero di decisione con tecniche che fanno uso della teoria dell'informazione;
3. si testa l'albero così costruito sui pattern rimanenti;
4. si incrementa la dimensione della finestra aggiungendo dei pattern che sono stati classificati erroneamente, fino a un numero massimo stabilito;
5. si ripetono iterativamente i passi 2, 3 e 4 fino a quando si ottiene un albero che

classifica correttamente tutti i pattern del training set, oppure quando il tasso d'errore dell'ultimo albero costruito risulta maggiore a o uguale a quello dell'albero precedente;

6. si può effettuare un'operazione di pruning sull'albero di decisione ottenuto.

Un parametro che regola (in maniera non nota deterministicamente) la capacità di generalizzazione è il livello di pruning (potatura dell'albero). Il pruning impedisce la divisione ricorsiva su attributi che non mostrano una chiara rilevanza e consiste nel "tagliare" tutti i rami dell'albero che si ritiene peggiorino la capacità di generalizzazione dell'albero e che incrementano la sua complessità.

4.6.1 Cenni sul pruning di J48

Ad ogni foglia i è possibile associare un errore ε_i dato dalla frazione di record classificati correttamente sul training set:

$$\varepsilon_i = \frac{n_i}{n_i^t}$$

dove n_i è il numero di record classificati correttamente e n_i^t è il numero totale di record del training set che terminano nella foglia i. La procedura di pruning è realizzata attraverso la sostituzione di un intero sottoalbero con una foglia. Tale rimpiazzamento ha luogo se il tasso di errore totale sul sottoalbero è superiore a quello associato alla singola foglia.

4.6.2 Implementazione in Weka

L'impostazione di un albero avviene tramite i seguenti parametri:

- binarySplits permette di creare o meno un albero di decisione binario;

- confidenceFactor: numero reale compreso tra i valori zero e uno (default 0.25) che regola il livello di pruning e rappresenta il livello di confidenza utilizzato per stimare la probabilità d'errore associata alle foglie dell'albero; serve per decidere se tagliare o meno un determinato sottoalbero; a valori maggiori del confidence-factor corrisponde un albero più complesso;
- minNumObj il numero minimo di esempi (oggetti) che devono essere classificati;
- da ogni foglia (in altri termini, impone il vincolo che ogni foglia deve determinare la classe di almeno minNumObj esempi);
- reducedErrorPruning abilita o meno la procedura di riduzione dell'errore di pruning, questa è una procedura alternativa rispetto l'algoritmo C4.5, che cerca di ottimizzare le prestazioni potando l'albero in maniera da ottimizzare le prestazioni su un holdout fold;
- numFolds se si abilita reducedErrorPruning, è possibile regolare l'ampiezza dell'holdout fold: il dataset di training può essere diviso in un numFolds sottoinsiemi di eguale cardinalità, identificandone uno come holdout fold;
- saveInstanceData per salvare o meno i dati di training per la visualizzazione;
- seed su quale base estrarre gli esempi in maniera casuale, quando è abilitato reducedErrorPruning;
- subtreeRaising decide se considerare l'operazione di sostituzione di un nodo con uno dei nodi successivi, piuttosto che sostituire un sottoalbero con una foglia;
- unpruned per decidere se generare o no la potatura
- useLaplace decide se i valori di probabilità devono essere approssimati attraverso il metodo di Laplace; dato un nodo avente occorrenze n_1, $n_2, \ldots, n_c$ per ogni classe $i \in [1,\ldots,c]$ (quindi con occorrenza complessiva $\sum_{k=1}^{c} n_k$), possiamo calcolare una stima semplice della probabilità che un pattern di classe i attraversi il nodo $p_i = n_i / \Sigma_k n_k$; purtroppo non conosciamo l'affidabilità della stima. Con la stima di Laplace

cerchiamo di porre rimedio:

$$p_i = \frac{n_i + 1}{\sum_k n_k + c}$$

dove c è il numero di classi.

Graficamente l'albero si presenta con foglie racchiuse in rettangoli, la radice e i nodi intermedi sono delimitati da un'ellisse al cui interno mostrano il valore dell'attributo.

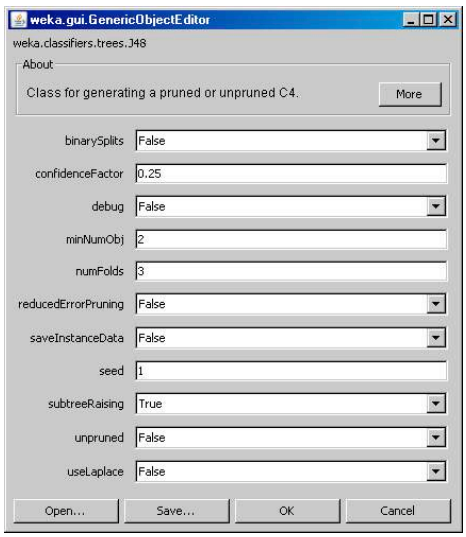

Figura 41 – Finestra J48

4.7 Prima Query

```
SELECT      traitstudy.name,
            traitstudy.populationsize,
            traitstudy.populationtype,
            traitstudy.markerstested,
            traitstudy.qtlsfound,
            traitstudygermplasmscore.germplasmscore,
            traitstudygermplasmscore.units
```

```
FROM        traitstudy

INNER JOIN  traitstudygermplasmscore
            ON traitstudy.id =
               traitstudygermplasmscore.traitstudyid
```

Il significato degli attributi è il seguente:

- `Traitstudy.name`: nome di un trait, ovvero la caratteristica ereditata geneticamente di un organismo, la possibile espressione che uno specifico carattere può assumere. Può anche essere un articolo o un tipo di studio fatto su un particolare trait;
- `Traitstudy.populationsize`: Quantità di piante utilizzate per le misurazioni
- `Traitstudy.populationtype`: tipo di piante esaminate
- `Traitstudy.markerstested`: quantità di marcatori genetici utilizzati per effettuare i test
- `Traitstudy.qtlsfound`: quantità di QTL trovati ;
- `Traitstudygermplasmscore.germplasmscore`: valore che indica la misura di un carattere;
- `Traitstudygermplasmscore.units`: unità di misura riferita all'attributo numerico germplasmscore;

4.7.1 Selezione attributi rilevanti

I risultati di Weka riguardano alla selezione di attributi, rispetto a `Traitstudy.name` sono i seguenti:

```
=== Run information ===
```

```
Evaluator:
weka.attributeSelection.InfoGainAttributeEval
Search:       weka.attributeSelection.Ranker -T
              -1.7976931348623157E308 -N -1
Relation:     Query1
Instances:    1126
Attributes:   7
              name
              populationsize
              populationtype
              markerstested
              qtlsfound
              germplasmscore
              units
Evaluation mode:    evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:
     Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 1
name): Information Gain Ranking Filter

Ranked attributes:
 3.0893  6 germplasmscore
 2.2207  7 units
 0       3 populationtype
 0       2 populationsize
 0       5 qtlsfound
 0       4 markerstested
```

```
Selected attributes: 6,7,3,2,5,4 : 6
```

Gli attributi rilevanti sono ovviamente quelli con valori non nulli del guadagno di informazione, in questo caso `Name`, `Germplasmscore`, `Units`.

4.7.2 Risultati sul Clustering

Non è stato evidenziato nulla di interessante tranne per un raggruppamento dei cluster basato sulle unità di misura e sui valori delle etichette di `Germplasmscore`. Sono stati evidenziati sedici cluster: sono molti ma non hanno portato a nessuna interpretazione valida. L'algoritmo ha solamente evidenziato la corrispondenza tra valori e unità di misura ovvero quelle già presenti nelle tuple risultanti dalla query. La scarsità di attributi da esaminare ha influenzato fortemente la classificazione in termini negativi.

4.7.3 Albero di decisione ottenuto

L'albero di decisione è stato impostato con un confidencefactor di 0.25 ed è stato creato di tipo binario. Si presenta con una struttura piuttosto elaborata con molti rami e risulta di scarsa leggibilità a causa delle ramificazioni basate sulle unità di misura e sui diversi valori null. Questo risultato non è assolutamente apprezzabile e risente della scarsità di attributi presenti e della presenza dei nulli.

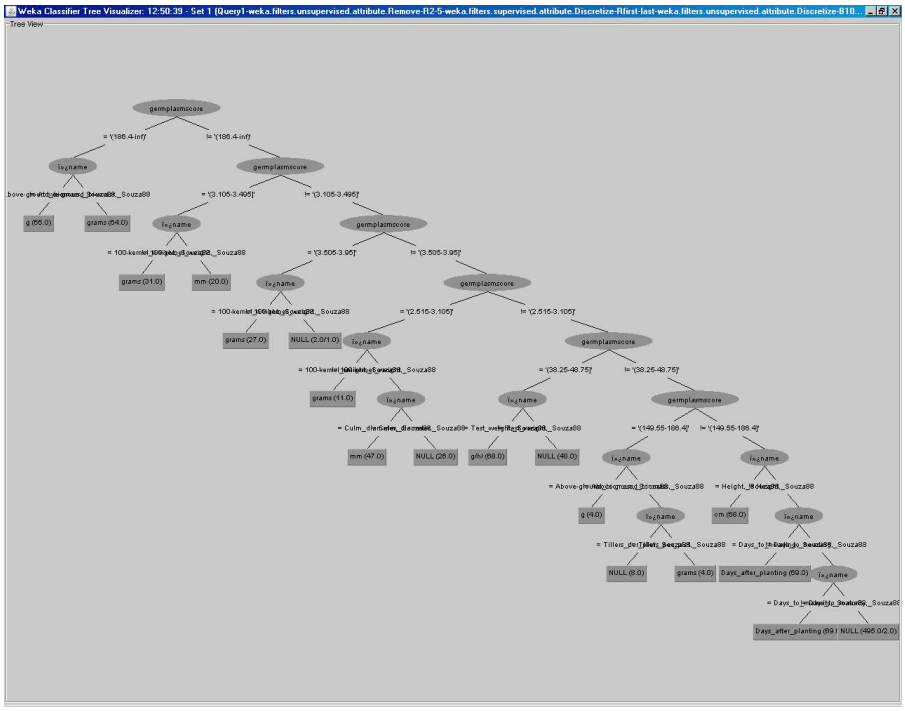

Figura 42 – Albero di decisione prima query Graingenes

4.7.4 Risultati di Apriori

Cercando quattro regole con confidenza minima 0.4 (40%) e supporto minimo 0.1 (10%) l'algoritmo riporta questo risultato:

```
Apriori
=======

Minimum support: 0.1 (113 instances)
Minimum metric <confidence>: 0.4
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Size of set of large itemsets L(2): 2
```

```
Best rules found:

1. germplasmscore='(-inf-42.17]' 503 ==> units=NULL 358
conf:(0.71)
2. units=NULL 576 ==> germplasmscore='(-inf-42.17]' 358
conf:(0.62)
3. germplasmscore='(83.44-124.71]' 239 ==> units=NULL
118    conf:(0.49)
```

Queste regole trovate non hanno alcun valore e risentono anch'esse della scarsa quantità di dati risultanti dalla query, i valori nulli in quantità elevata non hanno permesso di formulare alcuna associazione di valore.

4.7.5 Considerazioni

La query ha dimostrato la scarsità di informazioni riguardanti la porzione di Graingenes presa in esame e non si è rivelata utile per applicare gli algoritmi in modo soddisfacente.

4.8 Seconda Query

```
SELECT Traitstudy.Name as Traitstudy_Name,
       Traitstudy.Populationsize,
       Traitstudy.Populationtype,
       Traitstudy.Markerstested,
       Traitstudy.Qtlsfound,
       Qtl.Name as Qtl_Name,
       Qtl.Chromosomearm,
       Qtl.SignificanceLevel,
       Qtl.Maplabel
```

```
FROM TraitStudy, Qtl, QtlTraitStudy,
     Qtlsignificantmarker

WHERE Traitid is not null
  AND Mapdataid is not null
  AND Qtlsfound is not null
  AND Traitstudy.id=QtlTraitStudy.Traitstudyid
  AND Qtl.id=QtlTraitStudy.Qtlid
  AND Qtl.id=Qtlsignificantmarker.Qtlid
```

- `Traitstudy.Name`: Nome di un trait, ovvero la caratteristica ereditata geneticamente di un organismo, la possibile espressione che uno specifico carattere può assumere. Può anche essere un articolo o un tipo di studio fatto su un particolare trait;
- `Traitstudy.Populationsize`: quantità di piante utilizzate per le misurazioni;
- `Traitstudy.Populationtype`: tipo di piante esaminate;
- `Traitstudy.Markerstested`: quantità di marcatori genetici utilizzati per effettuare i test;
- `Traitstudy.Qtlsfound`: quantità di QTL trovati;
- `Qtl.Name`: nome dato al QTL;
- `Qtl.Chromosomearm`: braccio cromosomico associato al QTL trovato;
- `Qtl.SignificanceLevel`: variabile numerica che misura il livello di importanza statistica per un determinato QTL;
- `Qtl.Maplabel`: nome sulla mappa genetica in cui è stato testato uno specifico QTL

4.8.1 Selezione attributi rilevanti

L'elaborazione di Weka riporta quanto segue:

```
=== Run information ===
Evaluator:
weka.attributeSelection.InfoGainAttributeEval
Search:       weka.attributeSelection.Ranker -T
-1.7976931348623157E308 -N -1
Relation:     Query2
Instances:    1176
Attributes:   9
              Traitstudy_Name
              Populationsize
              Populationtype
              Markerstested
              Qtlsfound
              Qtl_Name
              Chromosomearm
              SignificanceLevel
              MapLabel
Evaluation mode:    evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:
     Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 1
Traitstudy_Name):
     Information Gain Ranking Filter
```

```
Ranked attributes:
 4.169    6 Qtl_Name
 3.691    5 Qtlsfound
 2.861    9 MapLabel
 1.793    2 Populationsize
 1.644    4 Markerstested
 1.468    3 Populationtype
 1.15     7 Chromosomearm
 0.146    8 SignificanceLevel

Selected attributes: 6,5,9,2,4,3,7,8 : 8
```

La selezione degli attributi ha dato un buon esito tranne per una unica voce, `significancelevel` che è stata scartata per il valore troppo basso rispetto a tutti gli altri.

4.8.2 Risultati ottenuti tramite clustering

I clustering trovati sono sei, però secondo l'algoritmo (e anche secondo il seed impostato) uno di questi è completamente vuoto. Si guarderanno gli insiemi di dati trovati per capire questo risultato inaspettato.
Date anche le numerose combinazioni di attributi per i grafici, ne vengono mostrati solo alcuni giudicati di particolare interesse, essi mostrano oltre alla colorazione dei clusterla disposizione dei dati del database Graingenes in base agli attributi affini fra di loro.

4.8.2.1 Cluster del grafico `Populationtype-Populationsize`

Il grafico di figura 43 mostra le colorazioni per i cluster trovati con ascissa `Populationtype` e ordinata `Populationsize`. I valori sono tutti quasi tutti o elevati o bassi e l'area centrale è vuota, la colorazione dei cluster è ben concentrata in aree ben definite in base ai valori dei due assi, l'etichetta "Recombinat_inbred" di `Populationtype` presenta invece un misto di punti appartenenti a due cluster distinti.

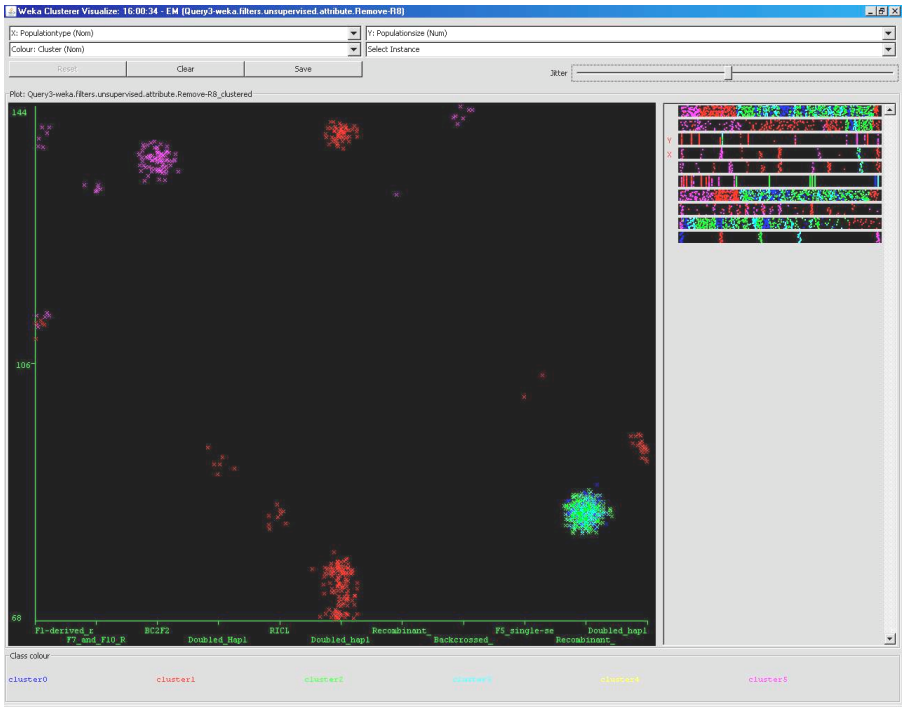

Figura 43 – Clustering seconda query Graingenes, attributi Populatintype-Populationsize

Selezionando tale area del grafico tramite un rettangolo, salvando questo sottoinsieme di dati e riaprendolo con Explorer si è potuto capire di più sui punti selezionati. Sono tre i cluster e non due come era stato notato all'inizio, i valori numerici corrispondenti a questi punti riguardano un insieme di tuple con lo stesso valore per l'attributo `Populationsize` pari a 84 e `Markertested` di 252, esse corrispondono ai `Traitstudyname` di:

- Days to heading, Siripoonwiwat 96;
- Plant Height, Siripoonwiwat 96;
- Grain Yield, Siripoonwiwat 96;
- Test Weight, Siripoonwiwat 96;
- Straw Yield, Siripoonwiwat 96;
- Groat Percentage, Siripoonwiwat 96;
- BYDV Resistance, Siripoonwiwat 96.

Guardando ancora una volta le suddivisioni e cambiando gli attributi delle ascisse e delle ordinate non si nota nient'altro di particolare.

4.8.2.2. Cluster del Grafico `Populationtype-Markedtested`

In questo grafico al cluster in basso a sinistra di figura 44 sono associati più valori distinti di `Populationtype` ovvero:

- F1-derived recombinant inbred;
- F7 and F10 RILs;
- BC2F2;
- Recombinant Inbred line (RIL): è presente una sola tupla di tale etichetta
- Backcrossed RILs

Osservando i punti dello stesso cluster interessato cambiando gli assi non è stata individuata alcuna associazione tra attributi da segnalare.

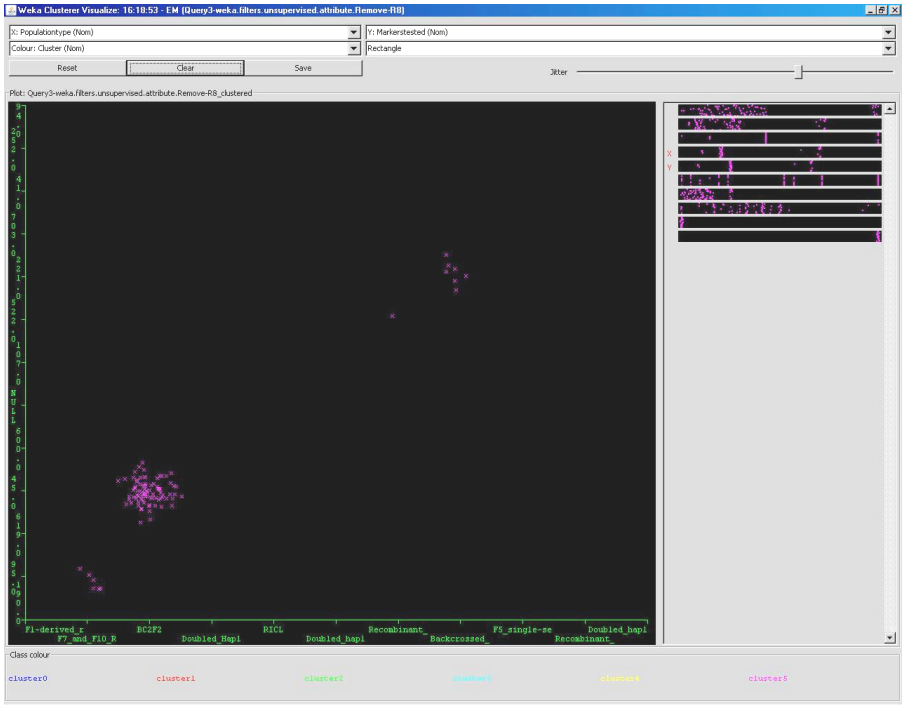

Figura 44 – Clustering seconda query Graingenes, attributi Populatintype-Markertested

4.8.2.3 Cluster del Grafico `Chromosomearm-Qtlsfound`

Questo grafico può essere interessante per vedere la distribuzione dei Qtl in base al tipo di `Chromosomearm`. Quest'ultimo attributo però presenta diversi valori

nulli: essi sono stati eliminati (selezionando le aree del grafico tramite il mouse) e ciò che ne risulta è una distribuzione di punti suddivisa in due cluster. I cluster che hanno evidenziato qualche dato rilevante sono quelli corrispondenti a cluster3 e cluster5. Il primo ha evidenziato che la maggior parte dei `Qtlsfound` ha piccoli valori sui `Chromosomearm` 2HS, 2HL, 4HS, 5HL; cambiando l'ordinata con l'attributo `Markertested` si vede che al valore 107.0 di quest'ultimo sono associati molti dei valori di `Chromosomearm` appena riportati; l'etichetta Populationtype corrispondente a questi `Chromosomearm` è "Double_Haploid"

L'altro cluster evidenzia in `Qtlsfound` valori superiori concentrati ancora una volta negli stessi `Chromosomearm` segnalati precedentemente, il valore di `Markertested` più frequente è 45.0 e `Populationtype` corrisponde a BC2F2.

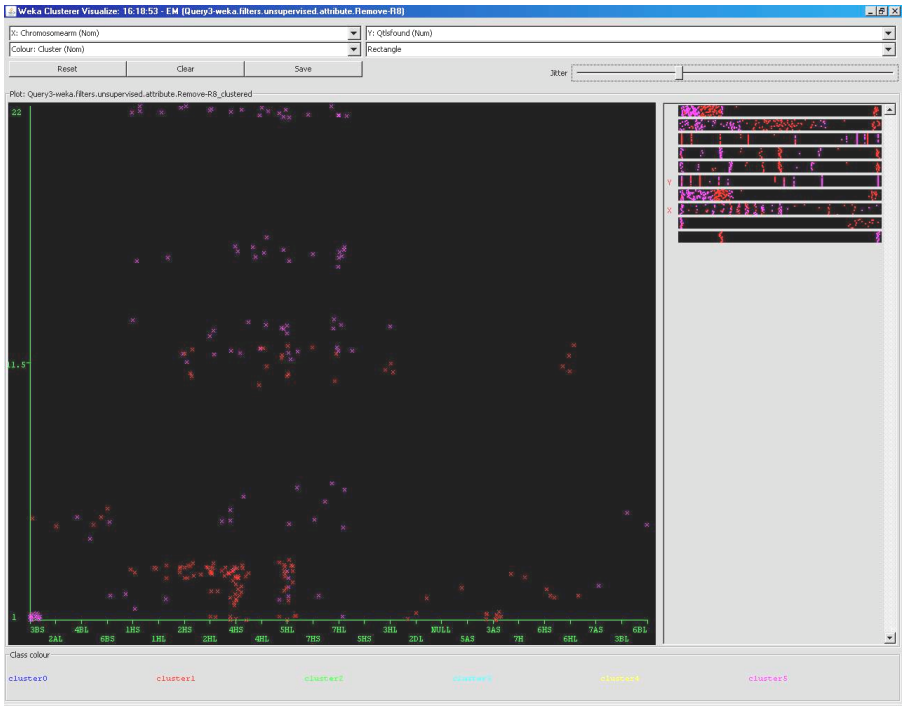

Figura 45 – Clustering seconda query Graingenes, attributi Chromosomearm-Qtlsfound senza valori nulli

4.8.3 Alberi di decisione ottenuti

Un primo albero di decisione trovato è quello che classifica l'attributo `Traitstudy_Name`. Weka riporta il seguente risultato:

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     Query2
weka.filters.unsupervised.attribute.Remove-R8-
weka.filters.supervised.attribute.Discretize-Rfirst-
last
Instances:    1176
Attributes:   8
              Traitstudy_Name
              Populationsize
              Populationtype
              Markerstested
              Qtlsfound
              Qtl_Name
              Chromosomearm
              MapLabel
Test mode:    evaluate on training data

=== Classifier model (full training set) ===

J48 pruned tree
------------------

Qtlsfound = '(-inf-22.5]': NULL (354.0/23.0)
Qtlsfound = '(22.5-29.5]': Straw_yield,_K,_NY3
(42.0/29.0)
```

```
Qtlsfound = '(29.5-44]': Groat_percentage,_K,_NY5
(63.0/45.0)
Qtlsfound = '(44-52.5]': BYDV,_K,_IL3 (95.0/79.0)
Qtlsfound = '(52.5-53.5]': Test_weight,_O,_NY5
(107.0/93.0)
Qtlsfound = '(53.5-66]': Days_to_heading,_K,_NY5
(141.0/126.0)
Qtlsfound = '(66-78.5]': Height,_K,_ID2 (200.0/174.0)
Qtlsfound = '(78.5-inf)': Yield,_K,_NY3 (174.0/152.0)

Number of Leaves  :     8

Size of the tree :  9

Time taken to build model: 0.08 seconds



=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances   455        38.6905 %
Incorrectly Classified Instances 721        61.3095 %
Kappa statistic                          0.3217
Mean absolute error                      0.0091
Root mean squared error                  0.0674
Relative absolute error                 72.4748 %
Root relative squared error             85.4964 %
Total Number of Instances             1176
```

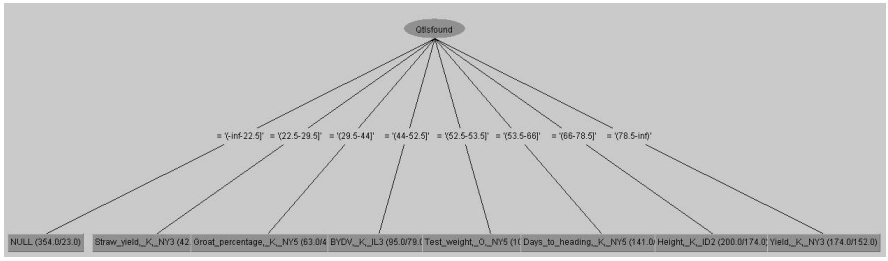

Figura 46 – Primo albero ottenuto, seconda query Graingenes

Questa prima classificazione, anche se può essere utile per una prima valutazione dei dati, non è di notevole rilevanza e ciò è dimostrato dal risultato riportato da Weka, le istanze non correttamente classificate sono molto elevate e quindi occorre cercare un nuovo albero che possa fornire una lettura più esaustiva dei dati. Forzando la creazione di un albero binario si è ottenuto un risultato più interessante e leggibile per quanto riguarda la classificazione dell'attributo `Populationtype`: il grafo ottenuto ha una percentuale di classificazione totale (salvo una singola tupla che possiamo considerare trascurabile). Weka riporta:

```
=== Summary ===

Correctly Classified Instances    1175     99.915  %
Incorrectly Classified Instances   1       0.085  %
Kappa statistic                          0.9982
Mean absolute error                      0.0003
Root mean squared error                  0.0121
Relative absolute error                  0.3337 %
Root relative squared error              5.8009 %
Total Number of Instances             1176
```

Graficamente è necessario scomporre e visualizzare l'albero in due parti per avere chiara la classificazione. La radice è rappresentata da Populationsize, la diramazione di sinistra è la più semplice ed è di lettura immediata, l'altra diramazione invece merita di essere guardata a parte.

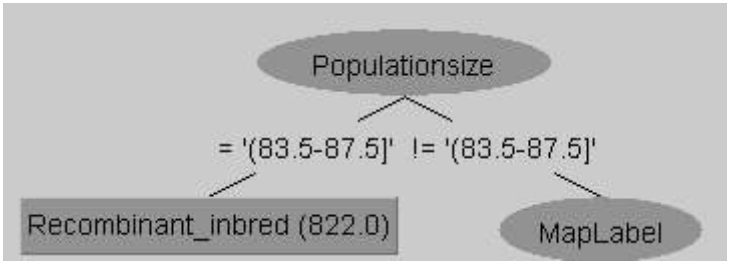

Figura 47 – Radice del secondo albero, seconda query Graingenes

L'attributo `Populationtype` corrispondente a "Recombinant_inbred (822.0)" di figura 47 è la foglia associata all'intervallo dei valori di `Populationsize` corrispondenti a "(83.5-87.5]". Per tutti gli altri valori è necessario percorrere più diramazioni. Dall'attributo `MapLabel` le diramazioni complesse sono quelle che partono dal ramo per valori uguali a null, mentre per le rimanenti si arriva a una foglia.

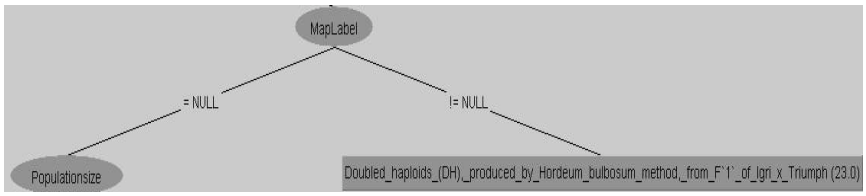

Figura 48 – Nodo Maplabel, seconda query Graingenes

L'attributo null di figura 48 è determinante nella composizione dell'albero e dimostra ancora una volta che i dati di Graingenes non sono ancora presenti in quantità ragionevole. Il ramo sinistro di `Populationsize` di figura 49 è poco annidato, mentre quello di destra è complesso e classifica la maggior parte delle tuple.

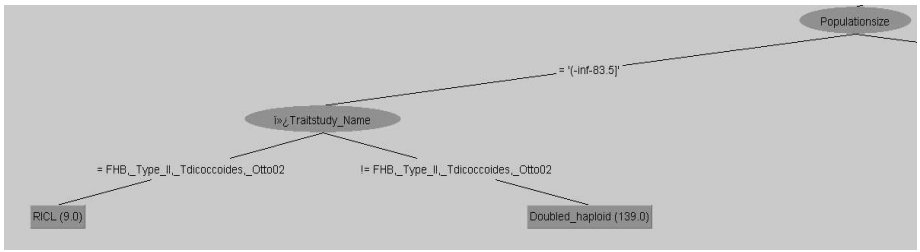

Figura 49 - Ramo sinistro di Populationsize, seconda query Graingenes

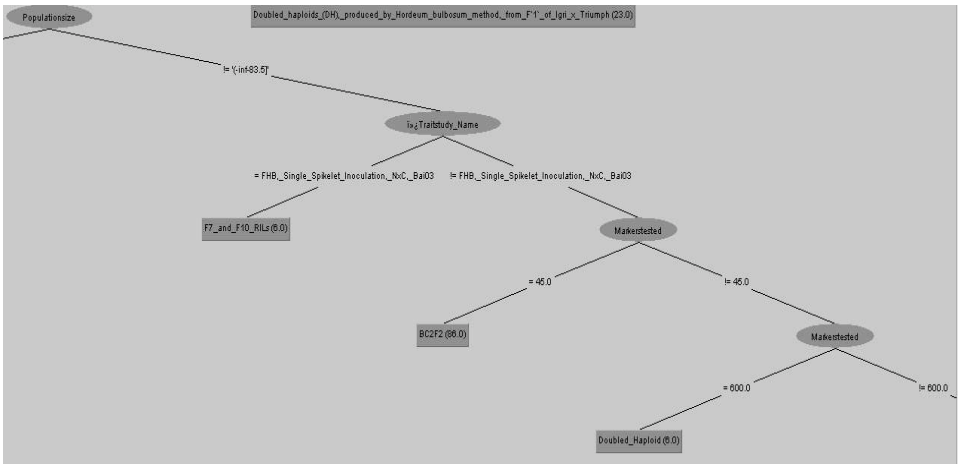

Figura 50 – Particolare del ramo destro, prima metà del percorso, seconda query Graingenes

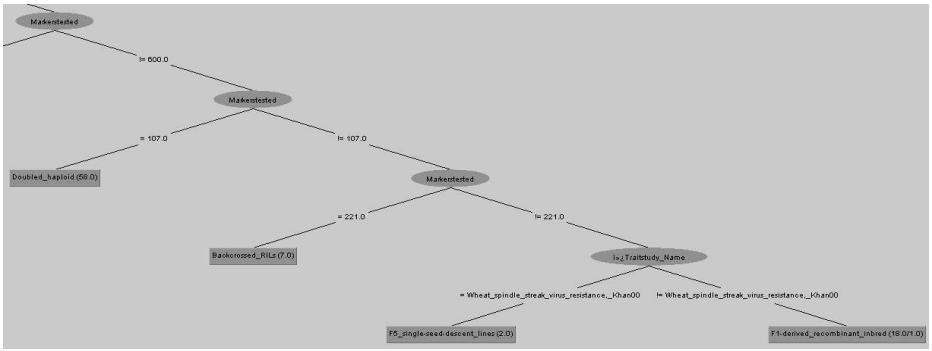

Figura 51 – Particolare del ramo destro, seconda metà del percorso, seconda query Graingenes

4.8.4 Risultati algoritmo Apriori

Con confidenza impostata al valore 0.3 e supporto 0.5 i risultati riportati sono i seguenti:

```
Apriori
=======

Minimum support: 0.65 (764 instances)
Minimum metric <confidence>: 0.3
Number of cycles performed: 7

Generated sets of large itemsets:
```

```
Size of set of large itemsets L(1): 4

Size of set of large itemsets L(2): 6

Size of set of large itemsets L(3): 4

Size of set of large itemsets L(4): 1

Best rules found:

1. Populationsize='(83.5-87.5]' 822 ==>
   Populationtype=Recombinant_inbred
   Markerstested=252.0 Chromosomearm=NULL 822
   conf:(1)
2. Populationtype=Recombinant_inbred 822 ==>
   Populationsize='(83.5-87.5]' Markerstested=252.0
   Chromosomearm=NULL 822    conf:(1)
3. Markerstested=252.0 822 ==>
   Populationsize='(83.5-87.5]'
   Populationtype=Recombinant_inbred Chromosomearm=NULL
   822    conf:(1)
4. Populationsize='(83.5-87.5]'
   Populationtype=Recombinant_inbred 822 ==>
   Markerstested=252.0 Chromosomearm=NULL 822
   conf:(1)
5. Populationsize='(83.5-87.5]' Markerstested=252.0 822
   ==> Populationtype=Recombinant_inbred
   Chromosomearm=NULL 822    conf:(1)
6. Populationsize='(83.5-87.5]' Chromosomearm=NULL 822
   ==> Populationtype=Recombinant_inbred
   Markerstested=252.0 822    conf:(1)
```

```
7. Populationtype=Recombinant_inbred
   Markerstested=252.0 822 ==>
   Populationsize='(83.5-87.5]' Chromosomearm=NULL 822
   conf:(1)
8. Populationtype=Recombinant_inbred Chromosomearm=NULL
   822 ==> Populationsize='(83.5-87.5]'
   Markerstested=252.0 822    conf:(1)
9. Markerstested=252.0 Chromosomearm=NULL 822 ==>
   Populationsize='(83.5-87.5]'
   Populationtype=Recombinant_inbred 822    conf:(1)
10.  Populationsize='(83.5-87.5]'
   Populationtype=Recombinant_inbred
   Markerstested=252.0 822 ==> Chromosomearm=NULL 822
   conf:(1)
```

Le regole perdono di significato a causa dell'attributo `Chromosomearm` che per molte tuple ha valore nullo, inoltre alcune si assomigliano molto ovvero descrivono associazioni identiche in cui cambia solo l'ordine di lettura degli attributi. Una volta rimosso l'attributo `Chromosomearm` è stato eseguito di nuovo l'algoritmo con gli stessi paramentri di supporto e confidenza ma con numero massimo di regole fissato a cinque. Il risultato ottenuto è soddisfacente:

```
Apriori
=======

Minimum support: 0.65 (764 instances)
Minimum metric <confidence>: 0.3
Number of cycles performed: 7

Generated sets of large itemsets:

Size of set of large itemsets L(1): 3
```

```
Size of set of large itemsets L(2): 3

Size of set of large itemsets L(3): 1

Best rules found:

1. Populationsize='(83.5-87.5]' 822 ==>
   Populationtype=Recombinant_inbred
   Markerstested=252.0 822    conf:(1)
2. Populationtype=Recombinant_inbred 822 ==>
   Populationsize='(83.5-87.5]' Markerstested=252.0 822
   conf:(1)
3. Markerstested=252.0 822 ==>
   Populationsize='(83.5-87.5]'
   Populationtype=Recombinant_inbred 822    conf:(1)
4. Populationsize='(83.5-87.5]'
   Populationtype=Recombinant_inbred 822 ==>
   Markerstested=252.0 822    conf:(1)
5. Populationsize='(83.5-87.5]' Markerstested=252.0 822
   ==> Populationtype=Recombinant_inbred 822
   conf:(1)
```

Le cinque regole hanno tutte confidenza unitaria e si sono focalizzate sul valore di Populationsize ovvero quello più frequente nelle tuple, si riportano nella seguente tabella i risultati in forma concisa:

	Attributo/i	Associato/i ad attributo/i	Conf.	Supp.
1	Populationsize= '(83.5-87.5]'	Populationtype= Recombinant_inbred e Markerstested=252.0	1	0.7
2	Populationtype= Recombinant_inbred	Populationsize= '(83.5-87.5]' e Markerstested=252.0	1	0.7

	Attributo/i	Associato/i ad attributo/i	Conf.	Supp.
3	Markerstested= 252.0	Populationsize= '(83.5-87.5]' e Populationtype= Recombinant_inbred	1	0.7
4	Populationsize= '(83.5-87.5]'	Populationtype= Recombinant_inbred	1	0.7
5	Populationsize= '(83.587.5]' e Markerstested= 252.0	Populationtype= Recombinant_inbred	1	0.7

Tabella 12 – Regole di associazione per la seconda query su Graingenes

Aumentando ulteriormente il valore di supporto non vengono individuate regole, lo stesso risultato si ottiene diminuendo molto il valore minimo di confidenza: le regole associative non variano. Non è possibile cercare altre regole.

4.8.5 Considerazioni

Questa query ha fornito risultati interessanti grazie alla quantità di dati disponibili e si è potuto testare in modo più concreto il funzionamento degli algoritmi. Sono stati trovati dati interessanti ma la presenza di valori nulli in quantità tutt'altro che modeste hanno comunque condizionato l'elaborazione di tutte le fasi di analisi. L'algoritmo E-M ha individuato qualche occorrenza riguardante l'attributo `Chromosomearm` e l'albero di decisione si è mostrato utile per una eventuale interpretazione dei dati disponibili. Le regole associative trovate, per quanto siano semplici e quasi ovvie possono bastare per avere informazioni immediate sulla maggioranza di valori riguardanti `Populationsize` e `Markedtested`.

4.9 Terza Query

```
SELECT locus.name as LocusName,
       locustype.type as LocusType,
       locuschromosomearm.chromosomearm as
       LocusChromosomearm,
       locuschromosome.chromosome as LocusChromosome,
       gene.name as GeneName,
       gene.fullname as GeneFullname,
       genelocus.howmapped as LocusHowmapped

FROM   locus
       INNER JOIN locuschromosome
       ON locus.id = locuschromosome.id
       INNER JOIN locuschromosomearm
       ON locus.id = locuschromosomearm.id
       INNER JOIN locustype
       ON locus.id = locustype.id
       INNER JOIN locusassociatedgene
       ON locus.id = locusassociatedgene.id
       INNER JOIN genelocus
       ON locus.id = genelocus.id
       INNER JOIN gene
       ON gene.id = locusassociatedgene.geneid
```

Il significato degli attributi è il seguente:

- `LocusName`: nome assegnato a un determinato locus individuato in un gene e in un cromosoma;
- `LocusType`: tecnica utilizzata per individuare un particolare locus nel cromosoma;
- `Locuschromosome`: cromosoma in cui è individuato un determinato locus;

- `Locuschromosomearm`: braccio cromosomico in cui è individuato un particolare locus, riferito al cromosoma riportato dall'attributo Locuschromosome;
- `GeneName`: nome abbrevviato del gene associato al locus;
- `GeneFullname`: nome esteso del gene associato al locus;
- `LocusHowmapped`: Posizione di uno specifico locus nella mappa genetica.

4.9.1 Selezione attributi rilevanti

I risultati riportati sono i seguenti:

```
=== Run information ===

Evaluator:
weka.attributeSelection.InfoGainAttributeEval
Search:       weka.attributeSelection.Ranker -T
              -1.7976931348623157E308 -N -1
Relation:     Query3
Instances:    1317
Attributes:   7
              LocusName
              LocusType
              LocusChromosomearm
              LocusChromosome
              GeneName
              GeneFullname
              LocusHowmapped
Evaluation mode:    evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:
```

```
     Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 1
LocusName):
     Information Gain Ranking Filter

Ranked attributes:
10.0433  5 GeneName
 5.4773  3 LocusChromosomearm
 4.4725  4 LocusChromosome
 3.2321  6 GeneFullname
 1.1118  2 LocusType
 0       7 LocusHowmapped

Selected attributes: 5,3,4,6,2,7 : 6
```

L'attributo `LocusHowMapped` viene scartato perché contiene solo valori nulli. L'attributo `LocusType` ha ottenuto il punteggio più basso a causa della presenza massiccia di un valore che è presente in gran parte delle tuple: ai fini dell'analisi, indipendentemente dai risultati dell'algoritmo di selezione, viene tenuto in considerazione con la stessa importanza degli altri.

4.9.2 Risultati ottenuti tramite Clustering

Sono stati trovati sei cluster. Passando alla visualizzazione grafica vi sono diverse combinazioni di attributi che presentano una assegnazione dei colori associati ai cluster caotica e di scarso rilievo. Aumentando il Jitter si scopre che i grafici di due coppie di attributi presentano insiemi di punti colorati che formano raggruppamenti omogenei e che sono quindi adeguate per una ispezione: esse sono `LocusName-LocusChromosome` e `LocusType-LocusChromosome`.

4.9.2.1 Grafico LocusName-LocusChromosome

Vi sono due gruppi di punti sul grafico che si prestano a essere osservati più da vicino e la figura 52 li mette in evidenza, questi due gruppi raccolgono a loro volta tre piccoli insiemi di punti molto vicini fra di loro.

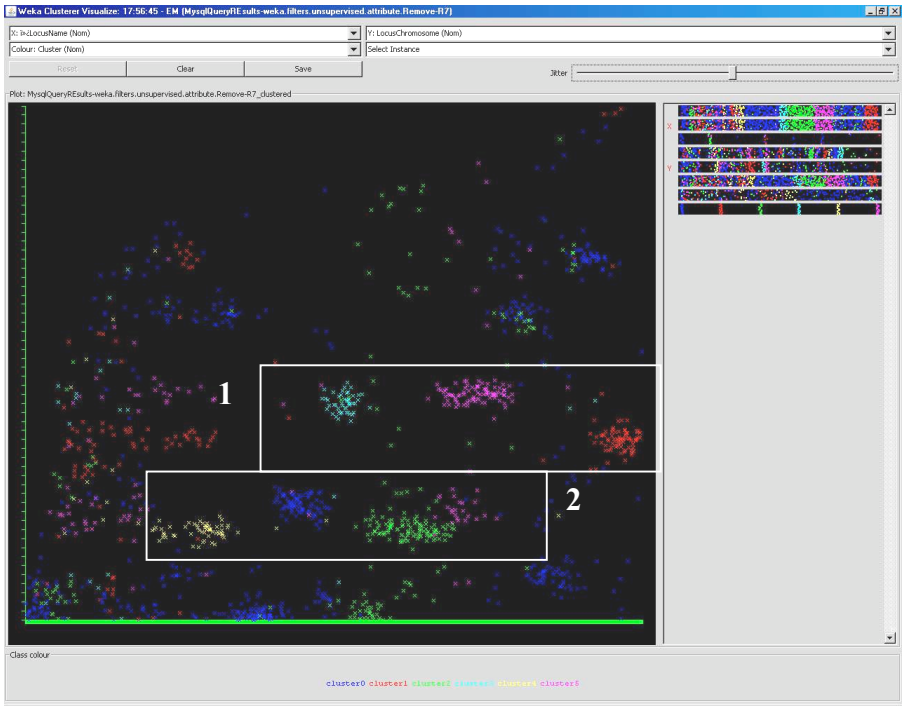

Figura 52 – I due raggruppamenti di cluster sul grafico LocusName-LocusChromosome

4.9.2.2 Primo raggruppamento

Sul grafico ottenuto, si procede a selezionare l'area corrispondente al primo gruppo tramite la voce Select Instance e poi a salvare l'area ottenuta in formato Arff per riaprirla in Explorer. Ricaricando il file Arff appena salvato nella sezione Preprocess si può subito avere una visione d'insieme delle varie quantità di valori presenti nei tre cluster. Poiché le tuple sono molte non è possibile guardare nel dettaglio ogni singolo valore, si presentano sinteticamente i valori più evidenti che risultano dai grafici corrispondenti:

- `GeneName`: molti nomi, con prevalenza di quelli con prefisso "Est", "Gli", "Glu", "Gpi", "Sr", "alpha" e "beta"
- `GeneFullname`: prevalenza di valori nulli

- `LocusName`: prevalenza di nomi con suffisso “AWBMA”, “BCD”
- `LocusType`: prevalenza di valori di tipo “RFLP” seguiti da valori di tipo “Gene”
- `LocusChromosome`: elevata frequenza di di valori corrispondenti a “7H”, “3H” e “6.0”
- `LocusChromosomeArm`: presenti vari valori, i più frequenti sono “3HL”, “7HL”, “6S”, “6L”

4.9.2.3 Secondo raggruppamento

Procedendo in modo analogo nella selezione dei punti, nel salvataggio e nella riapertura in Explorer si hanno le seguenti evidenze:

- `GeneName`: tra i molti nomi, vi è prevalenza di quelli con prefisso “Est”, “Ert”, “Stb”, “Xnt”
- `GeneFullname`: prevalenza di valori nulli
- `LocusName`: tra i molti nomi presenti, vi è prevalenza di nomi con prefisso “ABG”, “ACO”, “ABM” e “APR”
- `LocusType`: valori di tipo “RFLP” e di tipo “Gene” presenti in uguale quantità
- `LocusChromosome`: elevata presenza di valori di tipo “2H”, “3B” e “3A”
- `LocusChromosomeArm`: valori molto frequenti corrispondenti a “3BL”, “2HS”, “2HL”, “2S” e “2L”

4.9.2.4 `LocusType` “Gene” e “RFLP”

L’elemento in comune per entrambi i cluster è la prevalenza di valori `LocusType` di tipo “RFLP” e “Gene”. RFLP è l’acronimo di Restriction Fragment Length Polymorphism ovvero polimorfismo da lunghezza dei frammenti di restrizione: è una tecnica utilizzata per creare marcatori genetici sezionando parti di DNA con particolari enzimi detti endonucleasi, tali enzimi che attuano il taglio unicamente in corrispondenza di particolari sequenze

nucleotidiche, specifiche per ogni enzima. I frammenti di restrizione vengono quindi separati per lunghezza mediante elettroforesi su gel d'agarosio. La distanza tra le posizioni di taglio causate dagli enzimi di restrizione (i cosiddetti siti di restrizione) è variabile tra un individuo e l'altro, dando quindi luogo a variazioni nella lunghezza dei frammenti. Ciò si riflette in una diversa posizione di alcune bande sul gel, da cui il termine polimorfismo.
Il valore "Gene" indica che il particolare locus è stato individuato direttamente sul gene appartenente al cromosoma indicato nella tupla stessa, senza utilizzare la tecnica appena sopra accennata.
I geni e i bracci cromosomici associati con maggior frequenza al valore RFLP e Gene sono i seguenti:

- Per la tecnica RFLP, `LocusChromosome` 6.0, 2H, 3A, 3B, 3H, 7H; `LocusChromosomearm` 3AL, 3BS, 2S, 2L, 6S, 6L, 7HL
- Per la tecnica Gene, `LocusChromosome` 2H, 3H; `LocusChromosomearm` 2HL, 2HS, 3HL, 4HS

Salvo una modesta quantità di valori comuni per entrambe le modalità di `LocusType` trovate, è possibile affermare che i loci sui cromosomi con nomenclatura iniziale "3" (e che comprende numeri superiori a due) sono stati individuati con la tecnica RFLP, mentre quelli con nomenclatura iniziale "2" sono in prevalenza individuati direttamente sul gene.

4.9.2.5 Grafico `Locustype-LocusChromosome`

Vi sono due zone del grafico in cui si concentra la maggior parte dei punti, ovvero i valori di `LocusType` corrispondenti a "RFLP" e "Gene", questo andamento dei punti è simile a quello trovato nei due raggruppamenti precedenti. Si riportano di seguito i valori con maggior frequenza per entrambi i raggruppamenti.

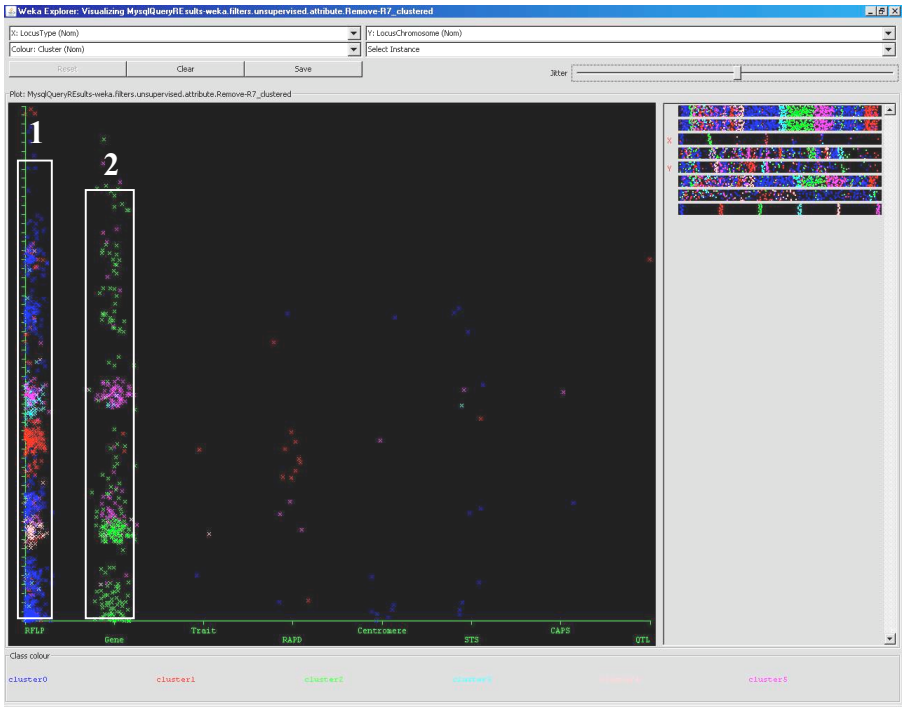

Figura 53 – I due raggruppamenti di cluster sul grafico LocusType-LocusChromosome

4.9.2.6 Primo raggruppamento

- `GeneName`: tra i molti nomi, vi è prevalenza di quelli con prefisso ”Ant”, “Brh”, “Cer”, “Est”, “Ert”, “Gli”, “Glu”, “Lr”, “Msg”, “Raw”, “Rph”, “Seg” e “Sr”;
- `GeneFullname`: prevalenza di valori nulli;
- `LocusName`: prevalenza di nomi con prefisso “Abc”, “Abg”, “Aco”, “Acor”, “Adh”, “Bagy”, “Bare”, “Bb” e “Bc”;
- `LocusType`: solamente valori di tipo “RFLP”;
- `LocusChromosome`: elevata presenza di valori di tipo “1H”, “5H”, “2H”, “3B”, “7H”, “4H”, “6H”;
- `LocusChromosomeArm`: valori molto frequenti corrispondenti a “3BL”, “1S”, “7HS”, “1L”, “7HL”, “2S”;

4.9.2.7 Secondo raggruppamento

- `GeneName`: tra i molti nomi, vi è prevalenza di quelli con prefisso “Abo”, “Alpha” “Est”, “Lr”, “Sr”, “Tel”;
- `GeneFullname`: prevalenza di valori nulli;

- `LocusName`: tra i molti nomi presenti, vi è prevalenza di nomi con prefisso “Amy”, “Ant”, “Amp”, “Apr”, “Awbma”, “Bare”;
- `LocusType`: solamente valori di tipo “Gene”;
- `LocusChromosome`: elevata presenza di valori di tipo “1H”, “2H”, “3H”;
- `LocusChromosomeArm`: valori molto frequenti corrispondenti a “3HL”, “2HS”, “2HL”;

4.9.3 Alberi di decisione ottenuti

Un albero di decisione cercato utilizzando tutte le tuple di partenza ha prodotto output visivi completamente illeggibili, si è quindi provato a creare alberi con gli stessi raggruppamenti trovati per il clustering. L’attributo `GeneFullname` viene rimosso perché assume spesso il valore nullo e può rovinare la valutazione fatta dall’algoritmo C4.5.
Utilizzando le sole impostazioni di default per tutti i valori e forzando la creazione di alberi binari, si sono ottenuti grafi chiari e leggibili, si riportano quelli più interessanti che hanno ottenuto una alta percentuale di istanze classificate correttamente, prendendo come riferiti ai due grafici utilizzati in precedenza per il clustering.

4.9.3.1 Alberi ricavati dai raggruppamenti di clustering

Questo albero è stato ricavato dalle tuple del primo raggruppamento selezionato nel grafico `LocusName-LocusChromosome`. L’albero con maggior numero di istanze classificate correttamente è quello riguardante l’attributo `LocusType`, l’albero è di facile lettura e interpretazione.

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -B -M 2
Relation:
Query3weka.filters.unsupervised.attribute.Remove-
R7_clustered-
weka.filters.unsupervised.attribute.Remove-R1,7-8
```

```
Instances:    242
Attributes:   5
              LocusName
              LocusType
              LocusChromosomearm
              LocusChromosome
              GeneName
Test mode:    evaluate on training data

=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances         195               80.5785 %
Incorrectly Classified Instances        47               19.4215 %
Kappa statistic                          0.5869
Mean absolute error                      0.0724
Root mean squared error                  0.1903
Relative absolute error                 62.1083 %
Root relative squared error             79.9363 %
Total Number of Instances              242
```

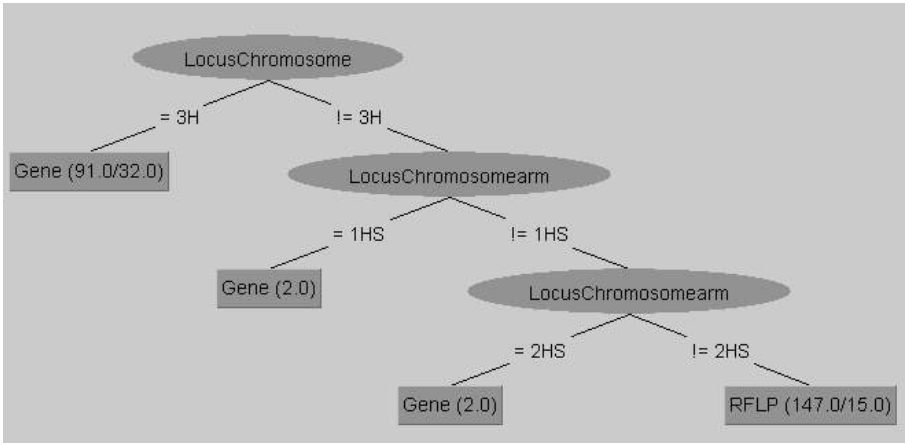

Figura 54 - Albero dell'attributo LocusType, terza query Graingenes

Un secondo albero di decisione che merita di essere tenuto in considerazione è quello che classifica l'attributo `LocusChromosome`. Anch'esso è stato ricavato dalle tuple del grafico `LocusName-LocusChromosome`, utilizzando il secondo raggruppamento.

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -B -M
2
Relation:     Query4-
weka.filters.unsupervised.attribute.Remove-
R7_clustered-
weka.filters.unsupervised.attribute.Remove-R1,7-8
Instances:    276
Attributes:   5
              LocusName
              LocusType
              LocusChromosomearm
              LocusChromosome
              GeneName
Test mode:    evaluate on training data


=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances      241      87.3188 %
Incorrectly Classified Instances    35       12.6812 %
Kappa statistic                          0.7461
Mean absolute error                      0.0071
Root mean squared error                  0.0597
Relative absolute error                 33.2275 %
Root relative squared error             60.7955 %
```

```
Total Number of Instances              276
```

Non sono stati ottenuti altri alberi interessanti, i risultati si limitavano a classificare con successo solo il 50% delle tuple presenti, inoltre gli alberi risultavano molto grandi e di difficile consultazione già nella visualizzazione a schermo intero.

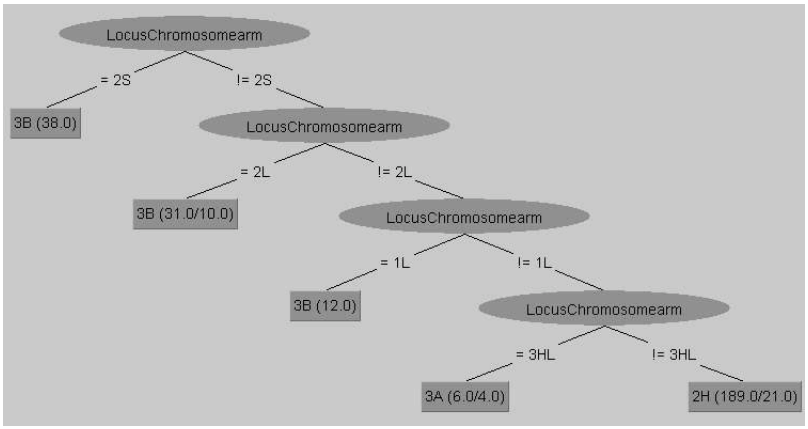

Figura 55 – Albero dell'attributo LocusChromosome, terza query Graingenes

4.9.4 Risultati algoritmo Apriori

Anche per la ricerca di regole associative, l'attributo `GeneFullname` è stato rimosso nonostante fosse stato considerato come valido all'inizio, esso infatti può falsare i risultati dell'algoritmo, oltre a essere inutile per qualsiasi valutazione.
I risultati effettuati su tutte le tuple non hanno portato ad alcuna formulazione di regole, supporto e confidenza sono stati impostati entrambi al valore 0.1 ma si sono rivelati insufficienti. Si è passati quindi a esaminare i quattro gruppi distinti di cluster esaminati in precedenza e si sono riportati i risultati più interessanti con livello di supporto e confidenza impostati entrambi sul valore 0.5 e nel caso in cui non vengano trovate regole, tali valori vengono scalati verso il basso e l'algoritmo eseguito di nuovo. I risultati, raccolti nelle tabelle riassuntive, riportano per ogni raggruppamento (in totale sono quattro) le associazioni calcolate e i valori di supporto e confidenza corrispondenti.

	Attributo/i	Associato/i attributo/i ad	Conf.	Supp.
1	LocusChromosome=7H	LocusType=RFLP	0.93	0.29
2	LocusType=Gene	LocusChromosome=3H	0.78	0.24
3	LocusChromosome=3H	LocusType=Gene	0.65	0.24
4	LocusType=RFLP	LocusChromosome=7H	0.43	0.29

Tabella 13 – Regole trovate nel primo raggruppamento impostando confidenza e supporto al minimi valore 0.2

	Attributo/i	Associato/i attributo/i ad	Conf.	Supp.
1	LocusType=Gene	LocusChromosome=2H	0.85	0.39
2	LocusChromosome=2H	LocusType=Gene	0.64	0.44

Tabella 14 – Regole trovate nel secondo raggruppamento impostando confidenza e supporto minimi al valore 0.3

	Attributo/i	Associato/i attributo/i ad	Conf.	Supp.
1	LocusType=Gene	LocusChromosome=2H	0.85	0.39
2	LocusChromosome=2H	LocusType=Gene	0.64	0.44

Tabella 15 – Regole trovate nel terzo raggruppamento impostando confidenza e supporto minimi al valore 0.3

Non sono stati trovate ulteriori regole. Tuttavia, gli scarsi risultati ottenuti non sono molto confortanti. Il supporto e la confidenza minimi che si è dovuto impostare è molto basso e rivela scarsa affinità tra gli elementi presenti nei vari raggruppamenti.

4.9.5 Considerazioni

La query ha fornito qualche risultato in più rispetto a quelle precedenti, ma a differenza delle altre è stato necessario considerare un insieme ristretto dei

risultati originari perché nella loro totalità le tuple non hanno fornito nulla di considerevole. Gli alberi ottenuti e le regole associative basate sui cluster trovati forniscono scarse informazioni e suggeriscono l'idea che i dati genetici siano per loro natura troppo variegati e distinti per poter cercare di ridurli in insiemi distinti. Seppur in questo caso in modo più leggero, i valori nulli continuano a essere presenti in modo determinante e a influenzare i risultati ottenuti.
Circa un quarto di tutte le tuple sono contrassegnate di tipo "Gene" nell'attributo `LocusType` e circa tre quarti delle rimanenti sono contrassegnate di tipo "RFLP". La tecnica RFLP, nata nei primi anni ottanta, è stata applicata all'allevamento delle piante solo in un secondo momento con ottimi risultati ed quindi è diventata la modalità più utilizzata per creare marcatori genetici, è ragionevole quindi affermare che la massiccia presenza di tuple di con il valore RFLP sia da imputare a tale ragione.

4.10 Query non utilizzate per l'analisi

Le seguenti query, sebbene ritenute molto interessanti per l'applicazione delle tecniche di data mining, non hanno fornito alcuna tupla come risultato e sono state scartate.

4.10.1 Qtls e Geni

```
SELECT      qtlsynonym.type as QtlSynonymType,
            qtlsynonym.name as QtlSynonymName,
            qtl.name AS QtlName,
            qtl.chromosomearm as QtlChromosomeArm,
            qtl.significancelevel as QtlSignificanceLevel,
            qtl.maplabel,
            geneclassremark.type AS GeneClassRemarkType,
            geneclassremark.remark as GeneClassRemark,
            geneclass.name AS GeneClassName
FROM        qtl
INNER JOIN qtlsynonym ON qtl.id = qtlsynonym.qtlid
INNER JOIN qtlassociatedgene ON qtl.id = qtlassociatedgene.qtlid
INNER JOIN qtlgeneclass ON qtl.id = qtlgeneclass.qtlid
```

```
INNER JOIN geneclassremark ON qtl.id = geneclassremark.id
INNER JOIN gene ON qtlassociatedgene.geneid = gene.id
INNER JOIN geneclass ON qtlgeneclass.geneclassid = geneclass.id
       AND geneclassremark.geneclassid = geneclass.id
```

4.10.2 Traits e Qtls

```
SELECT     traitstudy.name AS TraitstudyName,
           traitstudy.populationsize,
           traitstudy.populationtype,
           traitstudy.markerstested,
           traitstudy.qtlsfound,
           traitstudyheritability.heritability,
           traitstudyheritability.description,
           qtl.name AS QtlName,
           qtl.chromosomearm,
           traitstudyparentaldescription.description AS
           ParentalDescription

FROM       traitstudy

INNER JOIN  traitstudyparentaldescription ON traitstudy.id =
            traitstudyparentaldescription.traitstudyid
INNER JOIN  traitstudyheritability ON traitstudy.id =
            traitstudyheritability.traitstudyid
INNER JOIN  qtltraitstudy ON traitstudy.id =
            qtltraitstudy.traitstudyid
INNER JOIN  qtl ON qtltraitstudy.qtlid = qtl.id
```

4.10.3 Locus e Cromosomi

```
SELECT     gene.fullname,
           gene.name,
           genedescription.description,
           genechromosome.chromosome,
           genelocus.howmapped,
           genechromosomearm.chromosomearm
```

```
FROM        gene

INNER JOIN   genedescription ON gene.id = genedescription.geneid
INNER JOIN   genechromosome ON gene.id = genechromosome.geneid
INNER JOIN   genechromosomearm ON gene.id =
             genechromosomearm.geneid
INNER JOIN   genelocus ON gene.id = genelocus.geneid
```

4.10.4 Alleli e geni

```
SELECT     allele.name,
           allelegermplasm.type,
           gene.name AS GeneName,
           alleleproperty.property,
           allelephenotype.phenotype

FROM       allele

INNER JOIN  allelegermplasm ON allele.id = allelegermplasm.id
INNER JOIN  allelegeneproduct ON allele.id = allelegeneproduct.id
INNER JOIN  alleleproperty ON allele.id = alleleproperty.id
INNER JOIN  allelephenotype ON allele.id = allelephenotype.id
INNER JOIN  allelepathology ON allele.id = allelepathology.id
INNER JOIN  gene ON allele.id = gene.id
INNER JOIN  allelegene ON allele.id = allelegene.id
```

Conclusioni e sviluppi futuri

Alla luce del lavoro svolto con Weka è possibile elencare con precisione vantaggi e svantaggi percepiti durante l'utilizzo dell'applicazione.
I punti a favore sono i seguenti:

- È open source: la disponibilità gratuita di Weka e gli stessi aggiornamenti disponibili in breve tempo (spesso nell'ordine di qualche mese) rappresentano una forte attrattiva per chi deve scegliere uno strumento potente rispetto alle applicazioni a pagamento. La completa riscrittura dell'applicazione in linguaggio Java e la possibilità di modificare il codice sorgente permette l'utilizzo di Weka su sistemi con piattaforma Mac OS, Unix/Linux e Microsoft Windows e nello stesso tempo offre un elevato livello di personalizzazione per chiunque intenda modificare il codice alle proprie specifiche esigenze, inoltre entrare a far parte del gruppo di sviluppatori permette di condividere esperienze e punti di vista che in una applicazione a pagamento sono totalmente precluse.
- Potenza di calcolo. Weka si è dimostrata rapida e affidabile nei calcoli e nella presentazione dei risultati. In particolare, l'ambiente Explorer è multithreading e permette di eseguire più elaborazioni contemporanee senza appesantire il funzionamento della macchina ospitante la Java Virtual Machine. La possibilità di utilizzare, tramite lo strumento Experimenter, più macchine in rete che lavorano in parallelo è tale da soddisfare anche gli utilizzatori più esperti ed esigenti.
- La quantità di funzioni e algoritmi presenti, unitamente a quelli che saranno disponibili a breve termine tramite i continui aggiornamenti, estendono il potenziale campo di applicazione delle tecniche di data mining a nuove tipologie di dati o raffinano quelle già esistenti.

- Output esauriente. I risultati numerici ottenuti dagli algoritmi sono precisi e ricchi di dettagli, i grafici sono chiari e modificabili a proprio piacimento utilizzando il mouse in modo intuitivo.
- Modalità di impiego. Con ben quattro interfacce differenti utilizzabili contemporaneamente, Weka può essere liberamente utilizzato in modo più o meno complesso e articolato in base alle proprie esigenze.

Gli svantaggi riguardano caratteristiche intrinseche dell'applicazione e sono i seguenti:

- Weka è una applicazione che può rivelarsi di non facile utilizzo per l'utente poco esperto o per colui che si avvicina per la prima volta a una applicazione dedicata al data mining.
  Le modalità con cui si impostano gli algoritmi e si ottengono i risultati delle varie elaborazioni suggeriscono l'idea di un prodotto dedicato a chi di data mining ha già molta esperienza o sia data miner per professione.
- Scarsa documentazione. Come molte applicazioni open source, Weka soffre di una elevata mancanza di documentazione riguardante l'utilizzo delle interfacce e soprattutto degli algoritmi presenti. Attualmente tramite il sito ufficiale è possibile consultare alcune pagine che fungono da tutorial, per chi predilige la forma cartacea è possibile acquistare il libro **[8]** che tratta in modo più esauriente l'utilizzo di Weka e le tecniche di data mining che si possono applicare. Tuttavia tutto questo non è sufficiente, soprattutto per chi non è un "addetto ai lavori".
- Applicazione in continuo sviluppo. Sebbene metta a disposizione un vasto insieme di strumenti, Weka è sempre in continuo sviluppo ed è possibile incontrare, durante l'elaborazione, qualche malfunzionamento o notare la mancanza di alcune funzionalità. Un esempio è dato da Knowledge Flow: attualmente esso non presenta alcune funzioni che in Explorer esistono già da tempo e raramente si

sono avuti dei blocchi dell'applicazione per mancanza di memoria, causati probabilmente dal notevole numero di tuple da elaborare.

Infine alcune considerazioni generali sugli archivi CRA e Graingenes: il database CRA si è mostrato poco interessante per le tecniche di data mining puro, i risultati non sono stati molto significativi e suggeriscono l'idea di orientarsi verso altri tipi di ricerca, esso infatti è più adatto a un'analisi statistica che coinvolga i parametri numerici classici (correlazione, regressione, ecc…) e l'unica scelta ritenuta adeguata è stata quella di applicare un algoritmo di clustering. Non è stato nemmeno possibile ottenere alcuna regola associativa o alberi di decisione: l'estrema variabilità dei dati unita alla mancanza di valori di chiave non ha permesso di ottenere risultati nemmeno con valori di supporto e confidenza minimi.

Il database Graingenes, essendo molto vasto, ha favorito meglio l'analisi e ha permesso di creare un percorso di ricerca più articolato. Similmente al database CRA, Graingenes ha mostrato un insieme di dati genetici molto variegato che ha messo in difficoltà la ricerca di associazioni e classificazioni: i risultati di supporto e confidenza molto bassi evidenziano in modo inequivocabile l'elevata varietà dei dati e mostrano che non vi sono correlazioni degne di nota. Scoraggianti e da non sottovalutare i problemi dati dall'elevata presenza di attributi con valori nulli e la mancanza di tuple in alcune query ritenute interessanti: l'assenza di informazioni non solo si traduce in una scarsità di risultati nell'ambito della ricerca tramite Weka, ma rappresenta un lacuna anche per chi accede ai dati per applicazioni legate all'agricoltura e alla ricerca genetica.

Questa scarsità di risultati ottenuti non rappresenta un fallimento delle tecniche di data mining o di Weka ma va intesa come spunto per nuove ricerche che prendano in considerazione punti di vista differenti da quelli presi in esame in questa tesi.

Weka si è mostrato utile, efficiente e indispensabile e può essere utilizzato per continuare un eventuale lavoro di analisi sui due database genetici utilizzati in questa tesi.

Bibliografia

[1] Agrawal R., Imielinski T., Swami An., *Mining Association Rules between Sets of Items in Large Databases SIGMOD*. June 1993

[2] P. Dempster, N. M. Laird, D. B. Rubin. *Maximum Likelihood from Incomplete Data via the EM Algorithm*. Journal of the Royal Statistical Society. Series B (Methodological), Vol. 39, No. 1 (1977), pp. 1-38

[3] H.S. Chawla. *Introduction to Plant Biotechnology.* Science Publishers, U.S, Second Edition, 28 Jun 2002

[4] Consiglio per la Ricerca e la Sperimentazione in Agricoltura http://www.entecra.it/

[5] T. M. Cover, J. A. Thomas. *Elements of Information Theory*. Wiley, 1991

[6] Graingenes: A Database for Triticeae and Avena http://wheat.pw.usda.gov/GG2/index.shtml

[7] Gramene: A Resource for Comparative Grass Genomics http://www.gramene.org

[8] Ian H. Witten and Eibe Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[9] Jiawei Han, Micheline Kamber, *Data Mining: Concepts and Techniques, 2nd edition*. Morgan Kaufmann, Marzo 2006

[10] J. B. MacQueen (1967). *Some Methods for classification and Analysis of Multivariate Observations*, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1:281-297

[11] Pentaho Commercial Open Source Business Intelligence http://www.pentaho.com

[12] Pentaho Data Mining http://www.pentaho.com/products/data_mining/

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

[14] Richard J. Roiger, Michael W. Geatz, *Introduzione al data mining*. McGraw-Hill, Ottobre 2003

[15] Scarpa Bruno, Azzalini Adelchi. *Analisi dei dati e data mining*. Springer Verlag, 2004

[16] Weka, Data Mining with Open Source Machine Learning Software in Java http://www.cs.waikato.ac.nz/ml/weka/

[17] WekaDoc - The documentation project for WEKA. http://weka.sourceforge.net/wekadoc/index.php/Main_Page

[18] Weka Frequently Asked Questions http://weka.sourceforge.net/wiki/index.php/Frequently_Asked_Questions