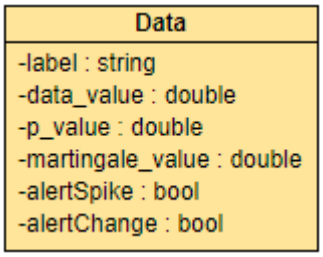
Relazione progetto per l'individuazione dei cambiamenti in una serie di dati.

Il progetto si propone di individuare i punti di cambiamento temporanei e permanenti in una serie di dati indipendenti sfruttando le funzionalità offerte dalla libreria *Microsft.ML*.

Dati in input

La serie di dati in input viene prelevata da un file .csv (con la virgola come carattere divisore) organizzato in due colonne in cui ogni riga è una tupla *<etichetta;valore>*.

Lato software le varie tuple vengono mappate in oggetti della classe custom *Data* mostrato graficamente qui di seguito:



Questo mapping avviene automaticamente invocando il metodo *LoadFromTextFile* della classe *DataOperationsCatalog* che restituisce un oggetto *IDataView* (essenzialmente un Enumerable della libreria Microsoft.ML).

```
IDataView dataView = mlContext.Data.LoadFromTextFile<Data>(path: DatasetPath, hasHeader:
true, separatorChar: ',');
```

Gli attributi *p_value*, *martingale_value*, *alertSpike* e *alertChange*, invece, verranno valorizzati nelle fasi successive (vedi paragrafo *Aggregazione dei dati*).

N.B. Nel progetto attuale è disponibile un file .csv già organizzato con dati trovati in rete riguardanti la produttività annuale in un particolare settore industriale/agricolo dal 1978 al 2018.

Elaborazione dei dati

L'elaborazione dei dati è delegata alle funzioni *DetectSpike* e *DetectChangepoint* in cui vengono sfruttati due *Estimator* forniti dalla libreria Microsof.ML: *DetectIidSpike* e *DetectIidChangePoint*.

```
var estimator = mlContext.Transforms.DetectIidSpike(outputColumnName:
nameof(Prediction.Values), inputColumnName: nameof(Data.data_value), confidence: 95,
pvalueHistoryLength: size / 4);

…

var estimator = mlContext.Transforms.DetectIidChangePoint(outputColumnName:
nameof(Prediction.Values), inputColumnName: nameof(Data.data_value), confidence: 95,
changeHistoryLength: size / 4);
```

Ai due *Estimator* vengono passati come parametri il *livello di confidenza* secondo il quale l'Estimator deve discriminare uno spike o un changepoint e la *dimensione della sliding windows* su cui vengono fatti i calcoli.

I dati restituiti dai suddetti *Estimator* vengono mappati tramite l'oggetto custom *Prediction* nel quale la memorizzazione è affidata ad un vettore di double:

- Il primo double può essere 0 o 1 e definisce un alert per informare se il valore a cui fa riferimento (uno dei valori della serie in input) è o meno uno spike/changepoint.
- Il secondo double è esattamente il valore di input (uno dei valori della serie passata in ingresso) a cui si riferisce questa *Prediction*.
- Il terzo double è il p-value calcolato per questo particolare valore di input.
- Il quarto double è il martingale-value calcolato per questo particolare valore di input (restituito solo nel caso *DetectIidChangePoint*).

Aggregazione dei dati

L'aggregazione dei dati è delegata alle funzioni *SaveSpikePredictions* e *SaveChangePredictions*. Queste due funzioni integreranno le informazioni ottenuto durante l'elaborazione con i dati caricati inizialmente memorizzandoli negli appositi attributi degli oggetti *Data* inizialmente tralasciati.

Creazione del report

La creazione del report invece viene implementata nella funzione *printData*.

Quest'ultima ha il compito di creare un file di testo in cui vengono visualizzati, per ogni riga, i vari oggetti *Data* ottenuti dall'esecuzione del programma, evidenziando quindi i punti di *spike* e di *changepoint* in formato testuale con relative informazioni su p-value e martingale-value per ogni singolo valore della serie in ingresso.

Massimo Smiraglio