



ELABORATO DI MACHINE LEARNING & DATA MINING

CLASSIFICAZIONE DEI TIPI DI CRIMINE DI SAN FRANCISCO



BETTINELLI LUCA 85947

MARIO MATTEO 86015

MASCIALE MICHELE 85982

Università degli Studi di Brescia

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Magistrale in Ingegneria Informatica

INDICE DEI CONTENUTI

INDICE DEI CONTENUTI.....	II
CAPITOLO 1 – DESCRIZIONE DEL PROBLEMA.....	1
1.1 - Introduzione.....	1
1.2 – Struttura del dataset	1
1.3 – Strumenti utilizzati.....	3
CAPITOLO 2 – PREPROCESSING DEI DATI	4
2.1 – Fase preliminare	4
2.2 – Attributo <i>Dates</i>	4
2.2.1 – Scomposizione in <i>Season</i> e <i>DailyRange</i>	5
2.2.2 – Scomposizione in <i>Year</i> , <i>Month</i> , <i>DayOfMonth</i> , <i>Hour</i> e <i>Minute</i>	6
2.2.3 – Creazione dell’attributo <i>DateDays</i>	6
2.2.4 – Creazione dell’attributo <i>TimeMinutes</i>	7
2.3 – Attributo <i>DayOfWeek</i>	7
2.4 – Attributo <i>Address</i>	8
2.4.1 – Creazione dell’attributo <i>isCross</i>	9
2.4.2 – Creazione dell’attributo <i>streetType</i>	10
2.4.3 – Riduzione dell’attributo <i>Address</i>	11
2.5 – Attributi <i>X</i> e <i>Y</i>	12
2.5.1 – Creazione di una griglia.....	14
CAPITOLO 3 – ALGORITMI E PRESTAZIONI.....	15

3.1 – Metrica di valutazione dei risultati	15
3.2 – <i>Gaussian Naive Bayes</i>	16
3.3 – <i>Neural Networks</i>	17
3.4 – <i>XGBoost</i>	17
3.5 – <i>Voting Classifier</i>	18
3.5.1 – <i>Voting Classifier</i> applicato a <i>XGBoost</i>	20

CAPITOLO 1 – DESCRIZIONE DEL PROBLEMA

1.1 - Introduzione

Dal 1934 al 1963, San Francisco era nota come patria di alcuni dei criminali più famigerati al mondo. Oggi la città è famosa più per la sua tecnologia che non per il suo passato criminale. Tuttavia, con la crescente disuguaglianza sociale, carenza di alloggi e proliferazione di mezzi tecnologici, tra cui BART (Bay Area Rapid Transit District), non vi è certamente scarsità di crimine. Dai distretti di Sunset a South of Market, da quelli di Marina a Excelsior, il dataset a nostra disposizione offre quasi 13 anni di notizie di reati da tutti i quartieri di San Francisco. Data l'ora e la posizione in cui si sono verificati i reati, è necessario prevederne la categoria. Si tratta di una delle competizioni accessibili a chiunque sul sito di Kaggle.

1.2 – Struttura del dataset

Il dataset a nostra disposizione è suddiviso in due file in formato CSV: il training set (circa 125 MB) e il test set (circa 89 MB). Questo dataset contiene le informazioni su reati e crimini registrati dal sistema di report dei dipartimenti di polizia di San Francisco. In particolare sono stati raccolti i dati del periodo che va dal 01/01/2003 al 13/05/2015 e sono stati successivamente distribuiti nei due file a settimane alterne: le settimane dispari appartengono al test set, quelle pari al training set.

Di seguito l'elenco degli attributi che descrivono un crimine nel dataset:

- **Id:** un numero identificativo auto-incrementale presente solo nel test set;
- **Dates:** una stringa che rappresenta il timestamp, ovvero la data in cui è avvenuto il crimine;
- **DayOfWeek:** una stringa che rappresenta il giorno della settimana in cui è avvenuto il crimine;

- **PdDistrict:** una stringa che rappresenta il nome del distretto nel cui dipartimento di polizia si è registrato il crimine;
- **Address:** una stringa che rappresenta l'indirizzo della via in cui è avvenuto il crimine;
- **X:** un numero che rappresenta la longitudine;
- **Y:** un numero che rappresenta la latitudine;
- **Category:** una stringa che rappresenta la categoria del crimine, presente solo nel training set. Questa è proprio la variabile target da predire;
- **Descript:** una stringa che rappresenta una descrizione dettagliata del crimine, presente solo nel training set;
- **Resolution:** una stringa che rappresenta la modalità con cui il crimine è stato condannato, presente solo nel training set.

In *Figura 1.1* e *Figura 1.2* è riportato un esempio di dataset, estratto da entrambi i file.

Dates	Category	Descript	DayOfWeek	PdDistrict	Resolution	Address	X	Y
2015-05-13 23:53:00	WARRANTS	WARRANT ARREST	Wednesday	NORTHERN	ARREST, BOOKED	OAK ST / LAGUNA ST	-122.425891675136	37.7745985956747
2015-05-13 23:53:00	OTHER OFFENSES	TRAFFIC VIOLATION ARREST	Wednesday	NORTHERN	ARREST, BOOKED	OAK ST / LAGUNA ST	-122.425891675136	37.7745985956747
2015-05-13 23:33:00	OTHER OFFENSES	TRAFFIC VIOLATION ARREST	Wednesday	NORTHERN	ARREST, BOOKED	VANNESS AV / GREENWICH ST	-122.42436302145	37.8004143219856
2015-05-13 23:30:00	LARCENY/THEFT	GRAND THEFT FROM LOCKED AUTO	Wednesday	NORTHERN	NONE	1500 Block of LOMBARD ST	-122.42699532676599	37.80087263276921
2015-05-13 23:30:00	LARCENY/THEFT	GRAND THEFT FROM LOCKED AUTO	Wednesday	PARK	NONE	100 Block of BRODERICK ST	-122.438737622757	37.771541172057795
2015-05-13 23:30:00	LARCENY/THEFT	GRAND THEFT FROM UNLOCKED AUTO	Wednesday	INGLESIDE	NONE	0 Block of TEDDY AV	-122.40325236121201	37.713430704116
2015-05-13 23:30:00	VEHICLE THEFT	STOLEN AUTOMOBILE	Wednesday	INGLESIDE	NONE	AVALON AV / PERU AV	-122.423326976668	37.7251380403778
2015-05-13 23:30:00	VEHICLE THEFT	STOLEN AUTOMOBILE	Wednesday	BAYVIEW	NONE	KIRKWOOD AV / DONAHUE ST	-122.371274317441	37.7275640719518
2015-05-13 23:00:00	LARCENY/THEFT	GRAND THEFT FROM LOCKED AUTO	Wednesday	RICHMOND	NONE	600 Block of 47TH AV	-122.508194031117	37.776601260681204

Figura 1.1: Un estratto del training set.

Id	Dates	DayOfWeek	PdDistrict	Address	X	Y
0	2015-05-10 23:59:00	Sunday	BAYVIEW	2000 Block of THOMAS AV	-122.39958770418998	37.7350510103906
1	2015-05-10 23:51:00	Sunday	BAYVIEW	3RD ST / REVERE AV	-122.391522893042	37.7324323864471
2	2015-05-10 23:50:00	Sunday	NORTHERN	2000 Block of GOUGH ST	-122.426001954961	37.7922124386284
3	2015-05-10 23:45:00	Sunday	INGLESIDE	4700 Block of MISSION ST	-122.437393972517	37.7214120621391
4	2015-05-10 23:45:00	Sunday	INGLESIDE	4700 Block of MISSION ST	-122.437393972517	37.7214120621391
5	2015-05-10 23:40:00	Sunday	TARAVAL	BROAD ST / CAPITOL AV	-122.45902362242902	37.7131719025215
6	2015-05-10 23:30:00	Sunday	INGLESIDE	100 Block of CHENERY ST	-122.42561645123001	37.73935051446279
7	2015-05-10 23:30:00	Sunday	INGLESIDE	200 Block of BANKS ST	-122.41265203979201	37.739750156312105
8	2015-05-10 23:10:00	Sunday	MISSION	2900 Block of 16TH ST	-122.418700097043	37.7651649409646

Figura 1.2: Un estratto del test set.

1.3 – Strumenti utilizzati

Il linguaggio di programmazione utilizzato per questo progetto è *Python* nella sua versione a 64 bit per un uso meno limitato della memoria. Tale linguaggio vanta, tra le altre qualità, di offrire svariate librerie appositamente dedicate al Machine Learning e al preprocessing dei dati, come pure quelle rivolte al tracciamento di grafici. In particolare sono state utilizzate le seguenti librerie:

- **pandas**: dotata di funzioni per la lettura e scrittura di dataset;
- **sklearn**: dotata di funzioni che implementano numerosi algoritmi di Machine Learning, addestrando i relativi classificatori ed estraendo le predizioni su nuovi dati;
- **matplotlib**: dotata di funzionalità per la generazione e personalizzazione di grafici.

CAPITOLO 2 – PREPROCESSING DEI DATI

2.1 – Fase preliminare

La prima fase in assoluto è stata rivolta alla riorganizzazione degli attributi di training set e test set in modo che questi fossero comparabili, ovvero avessero gli stessi attributi su cui i classificatori potessero lavorare. In particolare sono stati rimossi dal training set gli attributi *Descript* e *Resolution*, decisamente ridondanti data la categoria del crimine; è stato inoltre rimosso l'attributo *Id* dal test set in quanto inutile ai fini della predizione e non presente in fase di addestramento.

Al termine di questa fase training set e test set risultavano avere gli stessi attributi ed è stato perciò possibile iniziare il vero e proprio processing dei dati.

2.2 – Attributo *Dates*

Il primo attributo che ha meritato la nostra attenzione è proprio quello del timestamp, ovvero la data in cui il crimine è avvenuto. Riportiamo per comodità un esempio estratto dal dataset in *Figura 2.1*.

Dates
2015-05-13 23:53:00
2014-09-20 12:15:00
2014-02-03 00:37:00
2013-11-26 06:45:00

Figura 2.1: Esempi di valori per l'attributo Dates

Come si può notare dalla *Figura 2.1* i valori assunti da questo attributo sono molto specifici per il crimine in quanto la precisione del timestamp raggiunge il campo dei minuti. Inoltre grazie a Weka è stato possibile osservare come il numero di valori distinti nel training set per questo attributo è 389.257, numero che sfiora la metà del totale delle istanze. Abbiamo dunque pensato di elaborare l'attributo *Dates*, sostituendolo con attributi meno specifici che portassero maggiori informazioni in fase di addestramento. I seguenti paragrafi mostrano le soluzioni da noi adoperate.

2.2.1 – Scomposizione in *Season* e *DailyRange*

La prima soluzione è stata quella di ridurre drasticamente il numero di possibili valori per la data, sostituendo l'attributo *Dates* con i nuovi attributi *Season* e *DailyRange*, che rappresentano rispettivamente la stagione e la fascia giornaliera in cui è avvenuto il crimine. In particolare per quanto riguarda la stagione, abbiamo elaborato il campo del mese del timestamp secondo questo schema approssimativo:

- mesi 1-3: corrispondono alla stagione invernale (*Winter*);
- mesi 4-6: corrispondono alla stagione primaverile (*Spring*);
- mesi 7-9: corrispondono alla stagione estiva (*Summer*);
- mesi 10-12: corrispondono alla stagione autunnale (*Autumn*).

Per quanto riguarda invece la fascia giornaliera, abbiamo preso in considerazione il campo dell'ora del timestamp, secondo le seguenti regole:

- ore 6-12: corrispondono alla fascia mattutina (*Morning*);
- ore 12-18: corrispondono alla fascia pomeridiana (*Afternoon*);
- ore 18-24: corrispondono alla fascia serale (*Evening*);
- ore 0-6: corrispondono alla fascia notturna (*Night*).

Come si può intuire il numero di possibili valori per la data è stato ridotto a 16, ovvero il numero di combinazioni possibili tra i 4 valori della stagione e i 4 valori della fascia giornaliera.

In *Figura 2.2* riportiamo come esempio la suddetta scomposizione di alcune date.


Dates		Season	DailyRange
2015-05-13 23:53:00		Spring	Evening
2014-09-20 12:15:00		Summer	Afternoon
2014-02-03 00:37:00		Winter	Night
2013-11-26 06:45:00		Autumn	Morning

Figura 2.2: Esempio di scomposizione in Season e DailyRange

2.2.2 – Scomposizione in Year, Month, DayOfMonth, Hour e Minute

La seconda soluzione è invece stata quella di sostituire la data con un attributo per ogni campo del timestamp a nostra disposizione (ad eccezione dei secondi in quanto sempre nulli). In questo caso il numero di possibili combinazioni dei valori dei nuovi attributi nel caso peggiore è 6.963.840, in quanto si hanno 13 valori diversi per l'anno, 12 per il mese, 31 per il giorno, 24 per l'ora e 60 per il minuto. Nonostante ciò abbiamo pensato che per la fase di classificazione sarebbe stato più opportuno avere più attributi con meno valori disponibili per ognuno piuttosto che un solo attributo con moltissimi valori.

In Figura 2.3 riportiamo come esempio la suddetta scomposizione di alcune date.


Dates		Year	Month	DayOfMonth	Hour	Minute
2015-05-13 23:53:00		2015	5	13	23	53
2014-09-20 12:15:00		2014	9	20	12	15
2014-02-03 00:37:00		2014	2	3	0	37
2013-11-26 06:45:00		2013	11	26	6	45

Figura 2.3: Esempio di scomposizione in Year, Month, DayOfMonth, Hour e Minute

2.2.3 – Creazione dell'attributo DateDays

Dall'attributo *Dates* è stato possibile ricavare l'attributo *DateDays*, che esprime in quale giorno dall'inizio dell'anno un crimine è stato commesso.

Il nuovo attributo può assumere 366 differenti valori, poiché il dataset fornisce informazioni circa i crimini commessi anche in anni bisestili.

In Figura 2.4 riportiamo esempi di valori per il nuovo attributo elaborando alcune date.


Dates		DateDays
2015-05-13 23:53:00		133
2014-09-20 12:15:00		263
2014-02-03 00:37:00		34
2013-11-26 06:45:00		330

Figura 2.4: Esempio di creazione dell'attributo *DateDays*

2.2.4 – Creazione dell'attributo *TimeMinutes*

L'attributo *TimeMinutes* rappresenta l'aggregazione degli attributi *Hour* e *Minute*, ricavati a loro volta dall'attributo *Dates*. Esso contiene l'orario in cui il crimine relativo è stato commesso, espresso come numero di minuti dall'inizio del giorno. Tale attributo può assumere 1440 differenti valori, uno per ogni minuto contenuto in una giornata.

In *Figura 2.5* riportiamo esempi di valori per il nuovo attributo elaborando alcune date.


Dates		TimeMinutes
2015-05-13 23:53:00		1433
2014-09-20 12:15:00		735
2014-02-03 00:37:00		37
2013-11-26 06:45:00		405

Figura 2.5: Esempio di creazione dell'attributo *TimeMinutes*

2.3 – Attributo *DayOfWeek*

Per quanto riguarda il giorno della settimana, l'unico tipo di elaborazione a cui abbiamo pensato è stato quello di adottare una distinzione binaria tra i giorni del weekend e gli altri. In particolare abbiamo aggiunto un nuovo attributo, *Weekend*, che assumesse il valore *Yes* per i giorni *Saturday* e *Sunday*, e il valore *No* per tutti gli altri.

In *Figura 2.6* riportiamo un esempio di trasformazione dell'attributo in questione.

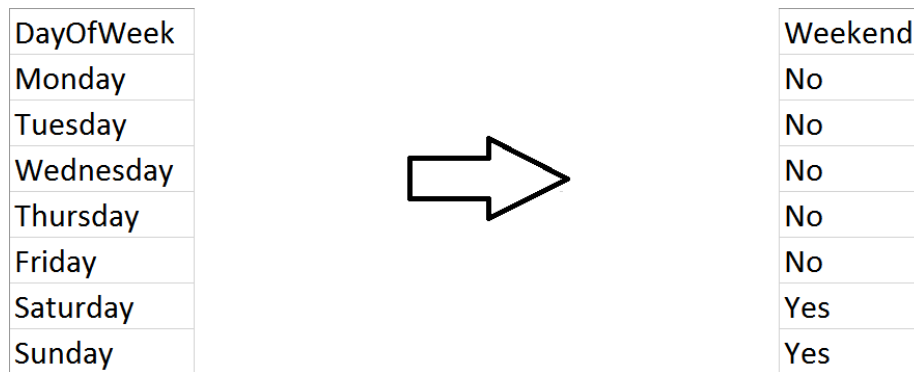


Figura 2.6: Esempio di elaborazione dell'attributo DayOfWeek

2.4 – Attributo *Address*

Un altro attributo che secondo noi andava rielaborato è quello che descrive testualmente il luogo di avvenimento del crimine, ovvero l'indirizzo. Riportiamo per comodità un esempio estratto dal dataset in *Figura 2.7*.

Address
OAK ST / LAGUNA ST
OAK ST / LAGUNA ST
VANNESS AV / GREENWICH ST
1500 Block of LOMBARD ST
100 Block of BRODERICK ST
0 Block of TEDDY AV
AVALON AV / PERU AV
KIRKWOOD AV / DONAHUE ST
600 Block of 47TH AV
JEFFERSON ST / LEAVENWORTH ST
JEFFERSON ST / LEAVENWORTH ST
0 Block of ESCOLTA WY
TURK ST / JONES ST
FILLMORE ST / GEARY BL
200 Block of WILLIAMS AV
0 Block of MENDELL ST
EDDY ST / JONES ST
GODEUS ST / MISSION ST
MENDELL ST / HUDSON AV
100 Block of JONES ST
200 Block of EVELYN WY

Figura 2.7: Esempi di valori per l'attributo Address

Come si può notare dalla *Figura 2.7* i valori assunti da questo attributo sono molto specifici per il crimine in quanto rappresentano il nome di una via con il relativo numero civico; nel caso di un incrocio tra due strade, il valore assunto comprende i nomi, separati da "/", delle vie con il relativo numero civico. Inoltre grazie a Weka è stato possibile osservare come il numero di valori distinti nel training set per questo attributo è 23.228, decisamente troppo elevato per applicare un criterio di *splitting* delle istanze. Abbiamo dunque pensato di elaborare l'attributo Address, creando nuovi attributi più generici che portassero maggiori informazioni in fase di addestramento.

I seguenti paragrafi mostrano le soluzioni da noi adoperate.

2.4.1 – Creazione dell'attributo *isCross*

La prima soluzione è stata quella di ridurre drasticamente il numero di possibili valori per l'indirizzo, creando il nuovo attributo *isCross*, che rappresenta un valore booleano secondo la seguente semplice regola:

- *Yes*: corrisponde ad un incrocio tra due strade;
- *No*: corrisponde ad una sola strada.

Come si può intuire il numero di possibili valori per l'indirizzo è stato ridotto a 2, numero di gran lunga inferiore rispetto a quello iniziale.

In *Figura 2.8* riportiamo come esempio la trasformazione di alcuni indirizzi.

Address	isCross
OAK ST / LAGUNA ST	Yes
OAK ST / LAGUNA ST	Yes
VANNESS AV / GREENWICH ST	Yes
1500 Block of LOMBARD ST	No
100 Block of BRODERICK ST	No
0 Block of TEDDY AV	No
AVALON AV / PERU AV	Yes
KIRKWOOD AV / DONAHUE ST	Yes
600 Block of 47TH AV	No
JEFFERSON ST / LEAVENWORTH ST	Yes
JEFFERSON ST / LEAVENWORTH ST	Yes
0 Block of ESCOLTA WY	No
TURK ST / JONES ST	Yes
FILLMORE ST / GEARY BL	Yes
200 Block of WILLIAMS AV	No
0 Block of MENDELL ST	No
EDDY ST / JONES ST	Yes
GODEUS ST / MISSION ST	Yes
MENDELL ST / HUDSON AV	Yes
100 Block of JONES ST	No
200 Block of EVELYN WY	No

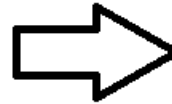


Figura 2.8: Esempio di creazione dell'attributo *isCross*

2.4.2 – Creazione dell'attributo *streetType*

La seconda soluzione è invece stata quella di creare l'attributo *streetType* che rappresentasse il tipo di strada in cui è avvenuto il crimine.

I valori assunti da questo nuovo attributo seguono il seguente schema:

- se l'indirizzo non è un incrocio: il valore assunto è il tipo di strada descritto nell'indirizzo;
- se l'indirizzo è un incrocio tra due strade dello stesso tipo: il valore assunto è il tipo delle strade dell'incrocio;
- se l'indirizzo è un incrocio tra due strade di diverso tipo: il valore assunto è rappresentato dai tipi delle due strade separati da "/".

Adoperando questa convenzione il numero di possibili valori per l'indirizzo è stato ridotto a 98, numero di gran lunga inferiore rispetto a quello iniziale.

In *Figura 2.9* riportiamo come esempio la trasformazione di alcuni indirizzi.

Address		streetType
OAK ST / LAGUNA ST		ST
OAK ST / LAGUNA ST		ST
VANNESS AV / GREENWICH ST		AV / ST
1500 Block of LOMBARD ST		ST
100 Block of BRODERICK ST		ST
0 Block of TEDDY AV		AV
AVALON AV / PERU AV		AV
KIRKWOOD AV / DONAHUE ST		AV / ST
600 Block of 47TH AV		AV
JEFFERSON ST / LEAVENWORTH ST		ST
JEFFERSON ST / LEAVENWORTH ST		ST
0 Block of ESCOLTA WY		WY
TURK ST / JONES ST		ST
FILLMORE ST / GEARY BL		ST / BL
200 Block of WILLIAMS AV		AV
0 Block of MENDELL ST		ST
EDDY ST / JONES ST		ST
GODEUS ST / MISSION ST		ST
MENDELL ST / HUDSON AV		AV / ST
100 Block of JONES ST		ST
200 Block of EVELYN WY		WY

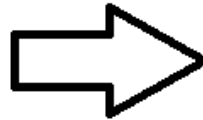


Figura 2.9: Esempio di creazione dell'attributo streetType

2.4.3 – Riduzione dell'attributo *Address*

Un ulteriore tentativo di ridurre il numero di valori possibili per l'attributo *Address* è stato quello di mantenere solo il nome della via, rimuovendo l'eventuale numero civico. L'idea era quella di raggruppare i crimini in base alla via in cui sono stati commessi, a prescindere da informazioni aggiuntive quali appunto il numero civico.

In *Figura 2.10* riportiamo come esempio la riduzione di alcuni indirizzi.

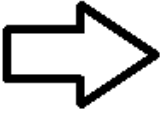
Address		AddressReduced
OAK ST / LAGUNA ST		OAK ST / LAGUNA ST
OAK ST / LAGUNA ST		OAK ST / LAGUNA ST
VANNESS AV / GREENWICH ST		VANNESS AV / GREENWICH ST
1500 Block of LOMBARD ST		LOMBARD ST
100 Block of BRODERICK ST		BRODERICK ST
0 Block of TEDDY AV		TEDDY AV
AVALON AV / PERU AV		AVALON AV / PERU AV
KIRKWOOD AV / DONAHUE ST		KIRKWOOD AV / DONAHUE ST
600 Block of 47TH AV		47TH AV
JEFFERSON ST / LEAVENWORTH ST		JEFFERSON ST / LEAVENWORTH ST
JEFFERSON ST / LEAVENWORTH ST		JEFFERSON ST / LEAVENWORTH ST
0 Block of ESCOLTA WY		ESCOLTA WY
TURK ST / JONES ST		TURK ST / JONES ST
FILLMORE ST / GEARY BL		FILLMORE ST / GEARY BL
200 Block of WILLIAMS AV		WILLIAMS AV
0 Block of MENDELL ST		MENDELL AV
EDDY ST / JONES ST		EDDY ST / JONES ST
GODEUS ST / MISSION ST		GODEUS ST / MISSION ST
MENDELL ST / HUDSON AV		MENDELL ST / HUDSON AV
100 Block of JONES ST		JONES ST
200 Block of EVELYN WY		EVELYN WY

Figura 2.10: Esempio di riduzione dell'attributo Address

2.5 – Attributi X e Y

Infine gli ultimi attributi che sono stati profondamente processati sono X e Y, che rappresentano rispettivamente la longitudine e la latitudine del luogo di avvenimento del crimine.

Riportiamo per comodità un esempio estratto dal dataset in Figura 2.11.

X	Y
-122.425891675136	37.7745985956747
-122.425891675136	37.7745985956747
-122.42436302145	37.8004143219856
-122.42699532676599	37.80087263276921
-122.438737622757	37.771541172057795
-122.40325236121201	37.713430704116
-122.423326976668	37.7251380403778
-122.371274317441	37.7275640719518
-122.508194031117	37.776601260681204
-122.419087676747	37.8078015516515

Figura 2.11: Esempi di valori per gli attributi X e Y

Innanzitutto grazie a Weka è stato possibile notare la presenza di *outliers*, ovvero istanze caratterizzate da valori errati di X e Y . In particolare, riportiamo di seguito i valori delle coordinate degli outliers, confrontandoli con le coordinate limite della città di San Francisco:

- X : -120.5 (*min*: -122.519703, *max*: -122.268906)
- Y : 90.0 (*min*: 37.684092, *max*: 37.871601)

Abbiamo supposto che il motivo di tali anomalie fossero dovute alla mancanza di disponibilità delle informazioni dei satelliti al momento della registrazione del crimine. Poiché questi outliers erano presenti sia nel training set che nel test set, non è stato possibile escluderli in fase di addestramento, né tanto meno in fase di validazione. Abbiamo perciò sfruttato le informazioni dell'attributo Address per recuperare, tramite una API di Google Maps, la latitudine e la longitudine dato l'indirizzo. Nel tentativo di risolvere il suddetto problema, ci siamo imbattuti in un altro tipo di anomalia che riguardava due indirizzi logicamente errati in quanto rappresentavano incroci tra strade fisicamente distanti, ma di cui era necessario ricavare le corrette coordinate X e Y . In particolare gli indirizzi:

- FLORIDA ST / ALAMEDA ST: due strade parallele che non avevano un punto di incrocio. Abbiamo prelevato le coordinate dell'unica strada perpendicolare a entrambe (TREAT ST);
- ARGUELLO BL / NORTHRIDGE DR: due strade diametralmente opposte (la prima a Nord, la seconda nella periferia meridionale di San Francisco). In questo caso abbiamo considerato l'attributo PdDistrict (il distretto di polizia che si è occupato del crimine in questione), notando come il più vicino distretto fosse proprio vicino ad ARGUELLO BL. Abbiamo perciò prelevato le coordinate di tale strada.


Come si può notare dalla *Figura 2.9* le coordinate sono espresse come numeri reali con molte cifre decimali e sono perciò molto sensibili a piccole variazioni. È stato necessario pertanto cercare una soluzione per ridurre drasticamente tale sensibilità al fine di facilitare di molto la fase di addestramento dei modelli. Nel seguente paragrafo viene illustrata la soluzione da noi adoperata.

2.5.1 – Creazione di una griglia

Data la ridotta varianza degli attributi X e Y, abbiamo pensato di creare una griglia quadrata virtuale (con un numero regolabile di celle per lato), definendo dapprima le coordinate limite della città di San Francisco (le stesse riportate nel precedente paragrafo). Detto n il numero di celle per lato, è stata effettuata una normalizzazione di entrambe le coordinate, riportandole nell'intervallo $[0, n]$ (0 coincideva con il minimo, n con il massimo); successivamente le coordinate ottenute sono state quantizzate in valori interi. Chiaramente in seguito a questa operazione istanze caratterizzate da un valore molto simile di X e Y, sono ricadute nella stessa cella, pertanto le nuove coordinate quantizzate erano uguali.

Per una maggiore chiarezza, in *Figura 2.12* riportiamo come esempio la trasformazione di alcune coordinate, usando una griglia di 100 celle per lato.

X	Y
-122.425891675136	37.7745985956747
-122.425891675136	37.7745985956747
-122.42436302145	37.8004143219856
-122.42699532676599	37.80087263276921
-122.438737622757	37.771541172057795
-122.40325236121201	37.713430704116
-122.423326976668	37.7251380403778
-122.371274317441	37.7275640719518
-122.508194031117	37.776601260681204
-122.419087676747	37.8078015516515



X	Y
58	59
58	59
59	82
58	82
50	56
74	4
60	15
95	17
3	60
63	88

Figura 2.12: Esempio di trasformazione degli attributi X e Y (griglia 100x100)

CAPITOLO 3 – ALGORITMI E PRESTAZIONI

3.1 – Metrica di valutazione dei risultati

Poiché la maggior parte degli algoritmi di *Machine Learning* messi a disposizione dalle librerie di *Python* funziona solo su attributi numerici, è stato necessario convertire i valori degli attributi categorici in valori numerici.

Secondo il regolamento previsto dalla competizione di Kaggle, la valutazione viene effettuata su un file csv contenente:

- un'intestazione, contenente la parola "Id" e i nomi di tutte le categorie di crimini in ordine alfabetico;
- per ogni istanza del test set, l'id relativo seguito dalle probabilità assegnate dal modello a ciascuna categoria.

Riportiamo in *Figura 3.1* un esempio di file *csv* da valutare.

[illegible]

Figura 3.1: Esempio di file di file csv da valutare

Ogni partecipante può inviare al massimo 5 file al giorno, che vengono valutati usando la metrica cosiddetta *multi-class logarithmic loss*.

Premesso che ogni crimine appartiene ad una sola categoria, la metrica di valutazione considera le probabilità fornite nel file csv, secondo la seguente formula:

$$\text{logloss} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{ij} \log(p_{ij})$$

dove N è il numero di istanze del test set, M è il numero di classi (categorie di crimine), y_{ij} vale 1 se l'istanza i appartiene alla categoria j , 0 altrimenti, e p_{ij} è la probabilità che all'istanza i sia assegnata la categoria j .

Nei seguenti paragrafi verranno illustrati gli algoritmi utilizzati per addestrare i modelli e predire le probabilità richieste. Verranno riportati anche i valori di *log-loss* (*Score*) e la relativa posizione in classifica; a causa del costante aumento di partecipanti alla competizione, il valore di posizione in classifica è stato riportato all'intervallo [1,100].

3.2 – Gaussian Naive Bayes

Il primo algoritmo che abbiamo utilizzato è stato il *Gaussian Naive Bayes*, una particolare implementazione del *Naive Bayes* che assume una distribuzione di probabilità gaussiana dei valori degli attributi rispetto alla classe. La scelta è stata effettuata sulla base di vari fattori, quali la semplicità, l'efficienza computazionale e la dimensione del dataset.

Sfruttando i vari tipi di preprocessing illustrati nel capitolo precedente, sono state fatte diverse prove. La migliore è stata effettuata rimuovendo gli attributi *Address* e *PdDistrict*, scomponendo l'attributo *Dates in Season* e *DailyRange* ed elaborando le coordinate X e Y in una griglia 10x10.

Riportiamo di seguito i risultati ottenuti:

- *Score*: 2.64736;
- Classifica: 63.

3.3 – Neural Networks

Il secondo modello che abbiamo costruito è stato il *Multi-Layer Perceptron*. È noto che l'efficienza computazionale di una rete neurale decresce all'aumentare della sua complessità, perciò per motivi di tempo e di spazio è stato effettuato un numero limitato di prove. La migliore è stata effettuata rimuovendo gli attributi *Address* e *PdDistrict*, scomponendo l'attributo *Dates* in *Season* e *DailyRange* ed elaborando le coordinate *X* e *Y* in una griglia 10x10. La rete neurale è stata costruita con i seguenti parametri:

- 100 unità di input di tipo tangentoide;
- 2 strati nascosti contenenti rispettivamente 100 unità di tipo tangentoide e 100 di tipo sigmoide;
- 39 unità di output di tipo *softmax*;
- *learning rate*: 0.1;
- numero di iterazioni di training aggiuntive in seguito alla stabilizzazione dell'errore di validazione: 2;
- epoche: 21.

Riportiamo di seguito i risultati ottenuti:

- *Score*: 2.56269;
- Classifica: 37.

3.4 – XGBoost

XGBoost è un algoritmo di *boosting* basato su alberi di decisione. Il classificatore è costituito da una funzione ottenuta addestrando e aggiungendo iterativamente un nuovo albero alla funzione risultato dell'iterazione precedente.

Ad ogni iterazione un nuovo albero viene addestrato a minimizzare una funzione obiettivo calcolata partendo dagli errori di classificazione sul *training set* della funzione misurati all'iterazione precedente.

Per evitare il fenomeno dell'*overfitting*, prima di essere sommato alla funzione complessiva, l'albero viene riscalato per un fattore di riduzione.

La migliore prova è stata effettuata scomponendo l'attributo *Dates* in *Hour* e *Minute*, aggiungendo gli attributi *DateDays*, *TimeMinutes*, *isCross* e *Weekend*. Il modello è stato costruito con i seguenti parametri:

- *max_depth*: 6;
- *learning_rate*: 0.7;
- *n_estimators*: 30.

dove *max_depth* è la profondità massima di ogni albero e *n_estimators* è il numero massimo di alberi creati.

Riportiamo di seguito i risultati ottenuti:

- *Score*: 2.27490;
- *Classifica*: 5.

3.5 – *Voting Classifier*

Dai paragrafi 3.2 e 3.3 si può notare come una maggiore complessità del modello porti ad un notevole miglioramento dello *score*. Abbiamo quindi considerato l'idea di creare un modello che fosse allo stesso tempo complesso e non eccessivamente costoso dal punto di vista computazionale. Il modello migliore che riassumeva tutte queste qualità è stato il *Voting Classifier*, un meta-classificatore basato su un sistema di votazione tra più classificatori creati in modo indipendente. Al contrario delle reti neurali, abbiamo dedicato la maggior parte del nostro tempo ad effettuare prove modificando vari parametri e tipi di preprocessing.

Gli algoritmi scelti per il sistema di votazione sono il *Gaussian Naive Bayes* e gli alberi di decisione (*Decision Tree*): abbiamo notato come l'assenza del primo portasse ad un calo delle prestazioni ed abbiamo perciò deciso di mantenerlo in ogni prova.

La prima prova è stata effettuata rimuovendo gli attributi *Address* e *PdDistrict*, scomponendo l'attributo *Dates* in *Season* e *DailyRange* ed elaborando le coordinate *X* e *Y* in una griglia 100x100.

Oltre al *Gaussian Naive Bayes*, il modello includeva due alberi di decisione aventi rispettivamente i seguenti parametri:

- *criterion: Gini, min_samples_split: 2500;*
- *criterion: Gini, max_leaf_nodes: 39.*

dove *criterion* è il criterio con cui viene valutata l'impurità di un nodo, *min_samples_split* è il numero minimo di esempi richiesto per espandere un nodo e *max_leaf_nodes* è il numero massimo di foglie. I risultati ottenuti sono i seguenti:

- *Score: 2.52774;*
- *Classifica: 29.*

La seconda prova è stata effettuata scomponendo l'attributo *Dates* in *Year*, *Month*, *DayOfMonth* e *Hour* ed elaborando le coordinate X e Y in una griglia 100x100.

A differenza della prima prova inoltre è stato assegnato un peso ad ogni classificatore. Oltre al *Gaussian Naive Bayes* avente un peso pari a 2, il modello includeva due alberi di decisione aventi rispettivamente i seguenti parametri:

- *criterion: Gini, min_samples_split: 2500 e peso: 5;*
- *criterion: Gini, min_sampes_leaf: 39 e peso: 4.*

dove *min_samples_leaf* è il numero minimo di esempi affinché un nodo possa essere considerato foglia.

I risultati ottenuti sono i seguenti:

- *Score: 2.47104;*
- *Classifica: 22.*

Nelle successive due prove è stato inserito nel sistema di votazione un ulteriore albero di decisione e sono stati eseguiti diversi tipi di preprocessing.

In particolare nella terza prova l'attributo *Dates* è stato scomposto sia in *Season* e *DailyRange* che in *Year*, *Month* e *Hour*. Inoltre le coordinate *X* e *Y* sono state elaborate in una griglia 100x100 e l'attributo *DayOfWeek* è stato sostituito dall'attributo *Weekend*.

Oltre al *Gaussian Naive Bayes* avente un peso pari a 2, il modello includeva dunque tre alberi di decisione aventi rispettivamente i seguenti parametri:

- *criterion: Gini, min_samples_split: 2500* e peso: 5;
- *criterion: Entropy, min_samples_leaf: 39* e peso: 4;
- *criterion: Gini, max_depth: 4* e peso: 1.

I risultati ottenuti sono i seguenti:

- *Score: 2.43672*;
- *Classifica: 20*.

Infine l'ultima prova è stata effettuata scomponendo l'attributo *Dates* sia in *Season* e *DailyRange* che in *Year*, *Month*, *Hour* e *Minute*, elaborando le coordinate *X* e *Y* in una griglia 100x100 e aggiungendo gli attributi *Weekend* e *isCross*, senza rimuovere i relativi attributi *DayOfWeek* e *Address*. I classificatori sono gli stessi della prova precedente, ad eccezione del primo albero di decisione, costruito con il parametro *min_samples_split* pari a 250.

I risultati ottenuti da questa ultima prova sono i seguenti:

- *Score: 2.35771*;
- *Classifica: 14*.

3.5.1 – *Voting Classifier* applicato a *XGBoost*

Visti gli ottimi risultati ottenuti con l'algoritmo *XGBoost*, abbiamo deciso di provare ad utilizzarlo in combinazione con l'algoritmo *Voting Classifier*. Il miglior risultato è stato ottenuto utilizzando

come classificatori in input all'algoritmo di *Voting* tre differenti istanze di *XGBoost* costruite rispettivamente con i seguenti parametri:

- *max_depth*: 6, *learning_rate*: 0.7, *n_estimators*: 30 e peso 6;
- *max_depth*: 7, *learning_rate*: 0.75, *n_estimators*: 35 e peso 5;
- *max_depth*: 6, *learning_rate*: 0.65, *n_estimators*: 33 e peso 4.

Tali pesi sono giustificati dal risultato ottenuto in fase di cross-validazione da ognuna delle tre istanze.

Riportiamo di seguito i risultati ottenuti:

- *Score*: 2.25923;
- *Classifica*: 4.