Documentazione Caso di Studio
Ingegneria della Conoscenza

A.A. 2022/23

NYC Accidents

Gruppo di lavoro

- Serena Di Cuia, 738718, s.dicuia4@studenti.uniba.it
- Samantha Liuzzi, 735327, s.liuzzi17@studenti.uniba.it

Repository GitHub

https://github.com/sam-e-sere/NYC

Sommario

Introduzione

Il caso di studio riguarda gli incidenti stradali avvenuti nella città di New York da settembre 2019 ad agosto 2020, utilizzando dati reali. È stato utilizzato un dataset sugli incidenti, caricato dal Police Department (NYPD) su NYC OpenData, un dataset sulle informazioni del traffico delle strade di NYC, caricato dal Department of Transportation (DOT) su NYC OpenData e un dataset sulle condizioni meteorologiche della città presente su Open-Meteo.
Dopo aver effettuato il pre-processing di questi dati, al fine di prepararli alle fasi successive, sono stati integrati all'interno di una base di conoscenza, arricchita poi, di nuove clausole che permettessero di inferire nuove informazioni attraverso il ragionamento automatico. Tutte queste informazioni sono state utilizzate per determinare le feature rilevanti che influiscono sulla gravità di un incidente, mediante algoritmi di apprendimento supervisionato. Inoltre, è stato utilizzato il clustering per identificare le aree della città in cui si verificano incidenti stradali con caratteristiche simili, come ad esempio un alto numero di persone ferite o uccise. Infine, è stata realizzata una Belief Network per calcolare la probabilità che un incidente sia pericoloso o meno, in base alle condizioni meteorologiche.

Elenco argomenti di interesse

- **Rappresentazione e ragionamento relazionale**: vengono inferite nuove informazioni, partendo dai dati contenuti all'interno dei dataset, mediante l'utilizzo di una base di conoscenza in Prolog.
- **Apprendimento supervisionato**: k-NN, Decision Tree, Random Forest, Ada Boost, Naive Bayes categorico.
- **Apprendimento non supervisionato**: utilizzo del k-means per il clustering.
- **Modelli di conoscenza incerta**: creazione di una Belief Network per il ragionamento probabilistico.

Descrizione dei dataset

Dal sito NYC OpenData sono stati scaricati i dataset già filtrati, perchè quelli originali erano di grandi dimensioni e non sarebbe stato possibile inserirli all'interno del progetto, poiché non supportati. Per comprendere come effettuare il filtraggio, è stato preso in considerazione il dataset di dimensione maggiore, cioè il dataset originale del traffico; dopo un'attenta analisi, è stata individuata la mancanza di dati per alcuni mesi dell'anno in alcuni anni, quindi si è deciso di considerare solamente i dati dal settembre del 2019 all'agosto del 2020, per ottenere le informazioni di un intero anno. Questo stesso filtraggio è stato effettuato per il dataset degli incidenti stradali, in modo da escludere informazioni irrilevanti.

Il dataset **"NYC Accidents"** contiene dati riguardanti:

- Data e ora dell'incidente;
- Luogo dell'incidente (latitudine e longitudine, borgo, strada…);
- Numero di persone coinvolte nell'incidente (numero di persone ferite e uccise);
- Altre informazioni non rilevanti ai fini del caso di studio.

Il dataset **"NYC Traffic"** contiene dati riguardanti:

- Data e ora del traffico;
- Luogo del traffico (borgo e strada);
- Informazioni sul traffico (volume e direzione).

Il dataset contenente i dati sulle condizioni meteorologiche non presentava il problema della dimensione, quindi è stato scaricato senza alcun tipo di filtraggio dal sito Open-Meteo.

Il dataset **"NYC weather"** contiene dati riguardanti:

- Data e ora del meteo;
- Informazioni sulle condizioni meteorologiche (temperatura, precipitazioni, pioggia, copertura nuvolosa, velocità del vento…).

Pre-processing

Estrazione delle informazioni

Il primo passaggio del pre-processing è stata l'estrazione delle informazioni necessarie dai vari dataset.

Per quanto riguarda il dataset **"NYC Accidents"** è stata eliminata la colonna 'CRASH DATE', contenente la data dell'incidente, ed è stata sostituita da tre colonne 'Y', 'M', 'D', rispettivamente anno, mese e giorno. Lo stesso procedimento è stato applicato alla colonna 'CRASH TIME' sostituita con 'HH', 'MM', rispettivamente ora e minuti. Le colonne contenenti le informazioni non rilevanti sono state rimosse e le righe contenenti valori mancanti nelle colonne 'LATITUDE' e 'LONGITUDE' sono state eliminate, poiché ritenute fondamentali in una fase successiva. Inoltre, è stato determinato il giorno della settimana utilizzando la data ed è stato memorizzato in una nuova colonna 'DAY OF WEEK'. Il dataset ottenuto è stato salvato nel file **"New NYC Accidents"**.

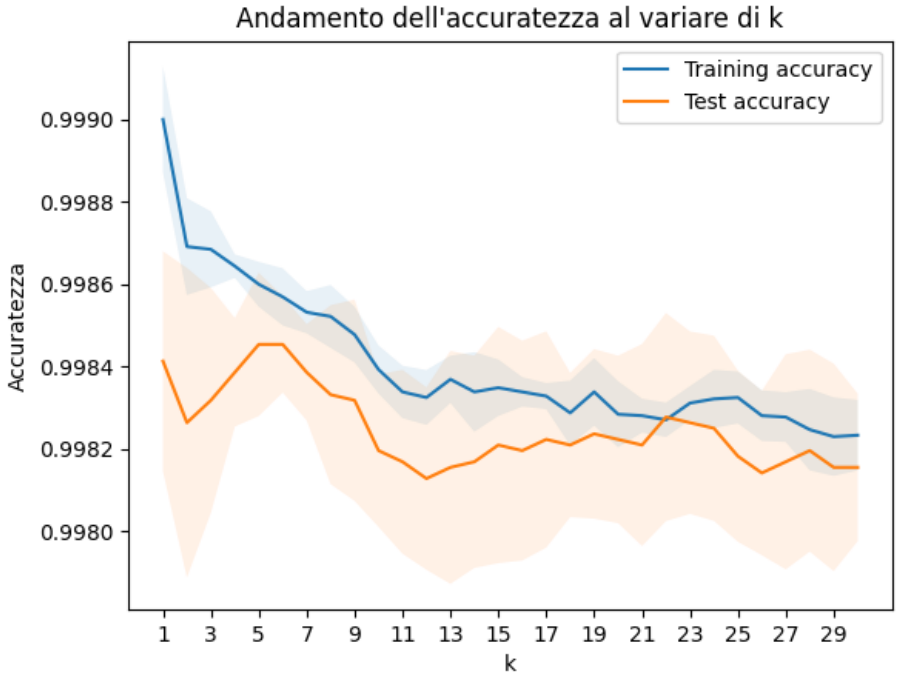
Il dataset **"NYC Traffic"** conteneva la feature superflua 'Direction'; la sua eliminazione ha generato righe identiche che si differenziavano solamente per 'Vol', il volume del traffico. Quindi, i valori della colonna 'Vol' sono stati sostituiti con il valore medio dei volumi, per ogni gruppo di righe duplicate su tutte le colonne ad eccezione del volume e della direzione non più considerata. Il dataset ottenuto è stato salvato nel file **"New NYC Traffic"**.

Per il dataset **"NYC weather"** viene eliminata la colonna 'TIME' contenente la data e l'ora del meteo, ed è stata sostituita da quattro colonne 'Y', 'M', 'D', 'HH', rispettivamente anno, mese, giorno e ora. I minuti non sono stati considerati, in quanto il valore per ogni riga era '00'. In seguito, sono state selezionate solamente le righe il cui anno era compreso tra 2019 e 2020. Abbiamo quindi rimosso le informazioni irrilevanti 'cloudcover (%)', 'cloudcover_mid (%)', 'cloudcover_high (%)', perché 'cloudcover_low (%)' è l'unico dato rilevante sulla copertura nuvolosa per la visibilità stradale. Il dataset ottenuto è stato salvato nel file "**New NYC weather"**.

Predizione dei valori mancanti (k-NN)

Il dataset **"New NYC Accidents"** presenta delle righe in cui il valore della feature 'BOROUGH' non è specificato. Tale caratteristica è rilevante per disambiguare i dati poiché nella città di New York ci sono strade con lo stesso nome, appartenenti a borghi differenti.
Quindi, è stato ritenuto opportuno predire i valori mancanti, utilizzando l'algoritmo di apprendimento supervisionato k-NN per la classificazione; le feature di input sono 'LATITUDE' e 'LONGITUDE', mentre la feature target è 'BOROUGH'.
Per l'apprendimento sono state utilizzate tutte le righe del dataset con campo 'BOROUGH' avvalorato, successivamente suddivise in train e test set. Il modello di classificazione, dato un

nuovo esempio, per predire il valore target, cerca i k esempi più simili memorizzati. Il valore di k può influire sull'accuratezza del modello infatti, se k è troppo piccolo il modello potrebbe essere troppo sensibile al rumore nei dati e potrebbe soffrire di overfitting, ovvero adattarsi troppo bene ai dati di addestramento e generalizzare male su nuovi dati. D'altra parte, se k è troppo grande, il modello potrebbe essere troppo insensibile ai dettagli dei dati e potrebbe soffrire di underfitting, ovvero non adattarsi adeguatamente ai dati di addestramento. Per questo motivo, si utilizza la funzione GridSearchCV della libreria scikit-learn per eseguire la ricerca a griglia, utile a trovare il valore di k che massimizzi l'accuratezza del modello.
Si è sperimentato anche l'andamento dell'accuratezza predittiva al variare di k nel range (1,31), tracciando la curva di validazione per il train set e il test set.



Come si evince dal grafico, il miglior valore di k è 5.

In seguito, il modello è stato addestrato con l'intero set di addestramento, utilizzando tale valore di k, ed è stata calcolata l'accuratezza del modello in base alle predizioni del set di test.

Accuratezza	0.997938479900179

Il modello addestrato viene utilizzato per fare previsioni sui dati con valori di 'BOROUGH' nulli. I valori predetti vengono poi aggiunti al dataset originale, inserito nel file **"Complete Accidents"**.

Corrispondenza dei dati

L'obiettivo di questa fase è quello di rimuovere dai tre dataset le righe non corrispondenti. Per righe non corrispondenti si intendono quelle che non hanno una corrispondenza con gli

altri dataset. In altre parole, vengono rimosse le righe che non hanno valori comuni nei campi che vengono utilizzati per unire i dataset.
Per la corrispondenza tra il dataset degli incidenti e il dataset del traffico, è stata necessaria una trasformazione dei valori del campo dei minuti 'MM' dell'incidente, in quanto i dati riguardanti il traffico sono stati analizzati ogni 15 minuti. In base al valore dei minuti del dataset originale **"Complete Accidents"** abbiamo applicato le seguenti modifiche:

- se il valore è minore di '08' viene approssimato a '00';
- altrimenti se il valore è minore di '24' viene approssimato a '15';
- altrimenti se il valore è minore di '38' viene approssimato a '30';
- altrimenti il valore viene approssimato a '45'.

I primi dataset ad esser messi in corrispondenza sono quelli degli incidenti e del meteo, in base alle feature 'Y', 'M', 'D', 'HH'.
Il dataset ottenuto contiene tre colonne 'ON STREET NAME', 'CROSS STREET NAME', 'OFF STREET NAME', quindi per la corrispondenza con il dataset del traffico (contenente solo la colonna 'STREET NAME'), sono stati considerati tre merge differenti, le cui feature comuni sono 'Y', 'M', 'D', 'HH', 'MM' e 'BOROUGH', mentre la differenza si ha sulla feature riguardante la strada. Da questa operazione sono stati ottenuti tre dataset, poi concatenati in uno unico. In questo dataset potrebbero esserci righe che fanno riferimento allo stesso incidente, dato che per un incidente possono essere registrate una strada principale, una secondaria ed una terziaria, e possono essere registrati volumi di traffico differenti per ognuna. Quindi, vengono eliminati i duplicati in base alle feature 'Y', 'M', 'D', 'HH', 'MM', 'BOROUGH', 'STREET NAME', 'CROSS STREET NAME', 'OFF STREET NAME', mantenendo la prima occorrenza, che sarà quella contenente il volume di traffico della strada principale, se esiste, altrimenti quella della secondaria o della terziaria. Per tenere traccia della strada che ha permesso il collegamento tra incidenti e traffico, viene effettuato un ulteriore merge per mezzo dell'identificativo del dataset del traffico, contenente solo 'TRAFFIC ID' e la strada, rinominata 'TRAFFIC STREET'.
Il dataset risultante contiene solo gli incidenti che hanno una corrispondenza con il traffico e le condizioni meteorologiche. Poiché necessari nelle fasi successive, a partire dal dataset completo sono stati generati i tre dataset **"Selected NYC Accidents"**, **"Selected NYC Traffic"** e **"Selected NYC Weather"**, privi di duplicati. Si noti che durante la suddivisione dei dataset non sono state considerate 'STREET NAME', 'CROSS STREET NAME', 'OFF STREET NAME', ma solo 'TRAFFIC STREET', perché è l'unica rilevante.

Knowledge Base (KB)

Una Knowledge Base (KB) è una raccolta di conoscenze organizzate in modo da consentire la gestione delle informazioni, per rappresentare la conoscenza e utilizzarla per prendere decisioni e risolvere problemi. La Knowledge Base implementata è utile per effettuare ragionamento automatico sfruttando la struttura di individui e relazioni, potendo inferire nuova conoscenza, e per ingegnerizzare nuove feature, sfruttabili per i task successivi. Per questa implementazione è stato utilizzato il linguaggio Prolog, mediante la libreria pyswip.

Per la creazione della KB sono stati definiti i seguenti individui e relazioni.

Individui

Sono state individuate le seguenti classi di individui:

- **accident(Collision_id)**, che rappresenta l'incidente stradale cui identificativo è Collision_id;
- **traffic(Traffic_id)**, che rappresenta il traffico stradale cui identificativo è Traffic_id;
- **date(Y,M,D,HH)**, che rappresenta le previsioni meteorologiche dell'indicazione temporale data.

Proprietà

Le proprietà associate agli individui corrispondono a feature presenti nei rispettivi dataset.

Proprietà per un generico incidente:
year(accident(Collision_id), Y)
month(accident(Collision_id), M)
day(accident(Collision_id), D)
hour(accident(Collision_id), HH)
minutes(accident(Collision_id), MM)
accident_date(accident(Collision_id), date(Y,M,D,HH,MM))
borough(accident(Collision_id), Borough)
location(accident(Collision_id), Latitude, Longitude)
street_name(accident(Collision_id), Traffic_street)
num_injured(accident(Collision_id), Num_Injured)
num_killed(accident(Collision_id), Num_Killed)
day_of_week(accident(Collision_id),Day_of_week)

Proprietà per un generico traffico:
year(traffic(Traffic_id), Y)
month(traffic(Traffic_id), M)

day(traffic(Traffic_id), D)
hour(traffic(Traffic_id), HH)
minutes(traffic(Traffic_id), MM)
traffic_date(traffic(Traffic_id), date(Y, M, D, HH, MM))
borough(traffic(Traffic_id), Borough)
volume(traffic(Traffic_id), Vol)
traffic_street(traffic(Traffic_id), Traffic_street)

Proprietà per una generica previsione meteorologica:
temperature(date(Y,M,D,HH), Temperature)
precipitation(date(Y,M,D,HH), Precipitation)
rain(date(Y,M,D,HH), Rain)
cloudcover(date(Y,M,D,HH), Cloudcover)
windspeed(date(Y,M,D,HH), Windspeed)
winddirection(date(Y,M,D,HH), Wind_direction)

Relazioni

Le relazioni che permettono di collegare gli individui sono:

has_Traffic(accident(Collision_id), traffic(Traffic_id))
has_Weather(accident(Collision_id), date(Y,M,D,HH))

Clausole

Per effettuare ragionamento sono state definite le clausole, che si possono raggruppare in base ai loro argomenti.
Per quanto riguarda le clausole che fanno riferimento al luogo dell'incidente, è stato calcolato il numero di collisioni avvenute nello stesso borgo, utilizzando una clausola booleana che definisce se due incidenti sono avvenuti nello stesso borgo o meno.

- *accidents_same_borough(accident(ID1), accident(ID2)) :- borough(accident(ID1), Borough), borough(accident(ID2), Borough)*
- *num_of_accidents_in_borough(accident(ID),Count) :- findall(ID1, accidents_same_borough (accident(ID), accident(ID1)), L), length(L, Count)*

Le stesse clausole sono state definite, confrontando la strada dell'incidente anziché il borgo.

- *accidents_same_street(accident(ID1), accident(ID2)) :- street_name(accident(ID1), Street), street_name(accident(ID2), Street)*
- *num_of_accidents_on_street(accident(ID), Count) :- street_name(accident(ID), OnStreet), borough(accident(ID), Borough), (OnStreet = 'unknown' -> Count =*

'unknown' ; OnStreet \\= 'unknown', findall(StreetID, (street_name(accident(StreetID), OnStreet), borough(accident(StreetID), Borough)), StreetIDs), sort(StreetIDs, UniqueStreetIDs), length(UniqueStreetIDs, Count))

Per quanto riguarda le clausole che fanno riferimento alle condizioni meteorologiche, è stato deciso, per ogni fattore meteorologico, di suddividere i valori continui in range attribuendo a ciascuno un'etichetta. Vengono considerati tutti i fattori meteorologici, ad eccezione di 'cloudcover', per il quale si definisce una clausola booleana, che suggerisce se la visibilità stradale è compromessa dalla copertura nuvolosa.

- *temperature_classification(accident(ID), Temperature) :- has_Weather(accident(ID), Data), temperature(Data,Temp), (Temp < 10.0, Temperature = 'cold'; Temp > 26.0, Temperature = 'hot'; Temp >= 10.0, Temp =< 26.0, Temperature = 'mild')*
- *rain_intensity(accident(ID), RainIntensity) :- has_Weather(accident(ID), Data), rain(Data,Rain), (Rain >= 0.1, Rain =< 4, RainIntensity = 'weak'; Rain > 4, Rain =< 6, RainIntensity = 'moderate'; Rain > 6, Rain =< 10, RainIntensity = 'heavy'; Rain > 10, RainIntensity = 'shower activity'; Rain = 0.0, RainIntensity = 'unknown')*
- *wind_intensity(accident(ID), WindIntensity) :- has_Weather(accident(ID), Data), windspeed(Data,Wind), (Wind > 0, Wind =< 19, WindIntensity = 'weak'; Wind > 19, Wind =< 39, WindIntensity = 'moderate'; Wind > 39, Wind =< 59, WindIntensity = 'strong'; Wind > 59, Wind =< 74, WindIntensity = 'gale'; Wind > 74, Wind =< 89, WindIntensity = 'strong gale'; Wind >= 90, WindIntensity = 'storm'; Wind = 0.0, WindIntensity = 'unknown')*
- *cloudcover(accident(ID)) :- has_Weather(accident(ID), Data), cloudcover(Data,Cloud), Cloud > 70*

Per quanto riguarda le clausole che fanno riferimento alle condizioni del traffico, è stata definita una clausola che permette di suddividere i valori dei volumi del traffico in range, attribuendo a ciascuno un'etichetta, e una clausola per calcolare il traffico medio di una strada. Quest'ultima clausola, ne comprende un'altra al suo interno, che effettua la somma dei volumi di traffico per ogni strada di ogni borgo, utilizzando le clausole definite precedentemente.

- *traffic_volume(accident(ID), Volume) :- has_Traffic(accident(ID), traffic(TrafficID)), volume(traffic(TrafficID), Vol), (Vol < 100, Volume = 'light'; Vol > 500, Volume = 'heavy'; Vol >= 100, Vol =< 500, Volume = 'medium')*
- *volume_sum_same_location(accident(ID), TotalVolume) :- findall(Vol, (accidents_same_borough (accident(ID), accident(ID1)), accidents_same_street(accident(ID), accident(ID1)), borough(accident(ID1), Borough), street_name(accident(ID1), Street), has_Traffic(accident(ID1),*

traffic(TrafficID)), volume(traffic(TrafficID), Vol)), Volumes), sum_list(Volumes, TotalVolume)

- *average_volume_same_location(accident(ID), AvgVolume) :- num_of_accidents_on_street(accident(ID), NumAccidents), (NumAccidents = 'unknown' -> AvgVolume = 'unknown'; volume_sum_same_location(accident(ID), TotalVolume), AvgVolume is TotalVolume / NumAccidents)*

Le restanti clausole sono:

- *time_of_day(accident(ID), TimeOfDay) :- hour(accident(ID), Hour), (Hour >= 4, Hour < 7, TimeOfDay = 'dawn'; Hour >= 7, Hour < 10, TimeOfDay = 'early morning'; Hour >= 10, Hour < 12, TimeOfDay = 'late morning'; Hour >= 12, Hour < 15, TimeOfDay = 'early afternoon'; Hour >= 15, Hour < 18, TimeOfDay = 'late afternoon'; Hour >= 18, Hour < 20, TimeOfDay = 'early evening'; Hour >= 20, Hour < 22, TimeOfDay = 'late evening'; Hour >= 22, Hour < 24, TimeOfDay = 'early night'; Hour >= 0, Hour < 4, TimeOfDay = 'late night')*
- *severity(accident(ID), Severity) :- num_injured(accident(ID), NumInjured), num_killed(accident(ID), NumKilled), (NumInjured = 0, NumKilled = 0, Severity = 'minor'; NumInjured >= 1, NumKilled = 0, Severity = 'moderate'; NumInjured >= 0, NumKilled > 0, Severity = 'major')*
- *is_not_dangerous(accident(ID)) :- severity(accident(ID), 'minor')*

La prima clausola effettua una suddivisione della giornata in fasce orarie. La seconda determina la gravità dell'incidente in base al numero di persone ferite e il numero di persone uccise. L'ultima clausola booleana definisce se l'incidente è lieve, cioè se non ha causato morti o feriti.
Come si evince dalle clausole precedenti, i valori nulli o, in alcuni casi, i valori pari a 0 sono assegnati all'etichetta 'unknown'.

Query

Le query su una KB in Prolog servono per interrogare la KB e recuperare informazioni specifiche. Per generare il dataset contenente tutte le feature inferite, si effettua la query corrispondente ad ogni nuova feature che si intende calcolare, sostituendo la variabile ID in accident(ID) con Collision_id, identificativo dell'incidente.
Al termine di questa operazione, viene creato il dataset **"generated_dataset"**, caratterizzato dalle seguenti feature:
'COLLISION_ID', 'NUM_ACCIDENTS_BOROUGH', 'NUM_ACCIDENTS_ON_STREET', 'TIME_OF_DAY', 'SEVERITY', 'TEMPERATURE', 'RAIN_INTENSITY', 'CLOUDCOVER', 'WIND_INTENSITY', 'TRAFFIC_VOLUME', 'AVERAGE_VOLUME', 'IS_NOT_DANGEROUS'.

Apprendimento supervisionato

Considerando sia le feature originarie dei dataset che quelle ingegnerizzate, si è individuato un possibile target per un task di classificazione, cioè 'IS_NOT_DANGEROUS' che assume valore 1 se l'incidente è di gravità 'minor', altrimenti 0.

L'obiettivo è individuare le feature che influiscono nel determinare se un incidente ha causato feriti e morti, non considerando le caratteristiche direttamente collegate, ad esempio 'SEVERITY', 'NUMBER OF PERSONS INJURED', 'NUMBER OF PERSONS KILLED', etc.
Per poter raggiungere questo obiettivo, sono state analizzate le prestazioni di vari modelli di classificazione in termini di accuratezza, precisione, recall e F1-score.

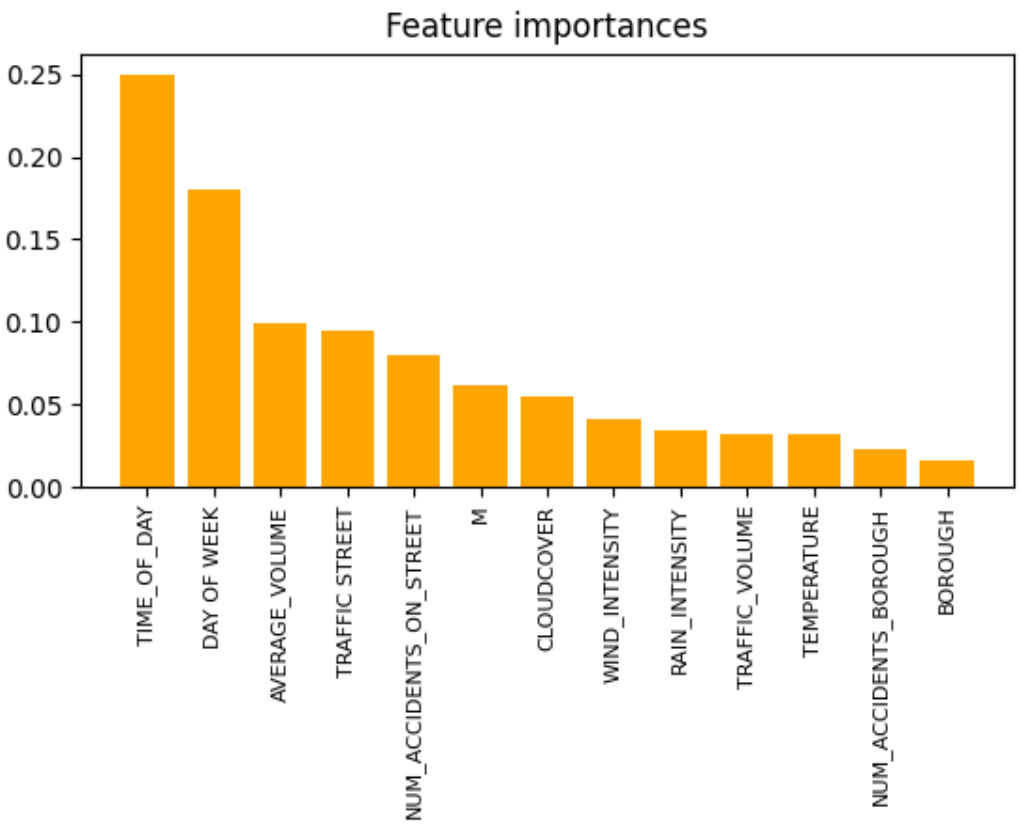
Le feature utilizzate per la classificazione sono state suddivise in variabili categoriche e numeriche, in quanto le variabili categoriche devono essere convertite, per il corretto funzionamento del modello, mediante la funzione OrdinalEncoder.
Le feature categoriche sono: 'BOROUGH', 'TRAFFIC STREET', 'TIME_OF_DAY', 'TEMPERATURE', 'RAIN_INTENSITY', 'WIND_INTENSITY', 'TRAFFIC_VOLUME', 'DAY OF WEEK'.
Le feature numeriche sono: 'M', 'CLOUDCOVER', 'AVERAGE_VOLUME', 'NUM_ACCIDENTS_BOROUGH', 'NUM_ACCIDENTS_ON_STREET'.

Decision Tree

Il Decision Tree o albero di decisione è un algoritmo di machine learning utilizzato per la classificazione e regressione, che si basa sulla creazione di un albero di decisione.
Per il task corrente il dataset considerato si divide in test set e train set, con i quali si sono misurate le performance del modello, al variare della profondità dell'albero. Inoltre, sono state considerate le misure di impurità: log-loss, gini, entropy; poiché non sono emerse differenze significative tra le prestazioni, è stato utilizzato il criterio entropy.

Accuratezza	0.673173073077077
Precisione	0.536534972944906
Recall	0.519074970259999
F1-score	0.503022315944364

Di seguito è riportato il grafico delle feature più importanti derivate dall'addestramento del modello. La somma dell'importanza di tutte le feature è 1.

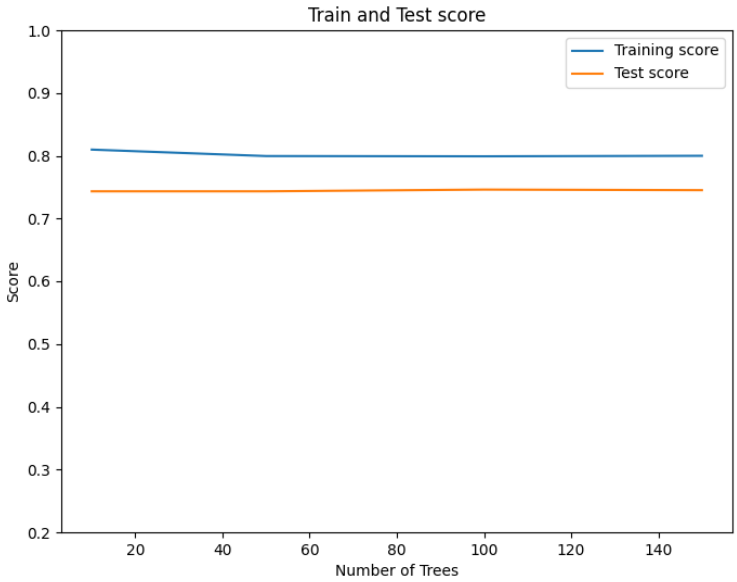


Feature ranking:
1. TIME_OF_DAY (0.250219)
2. DAY OF WEEK (0.180641)
3. AVERAGE_VOLUME (0.099318)
4. TRAFFIC STREET (0.094887)
5. NUM_ACCIDENTS_ON_STREET (0.080299)

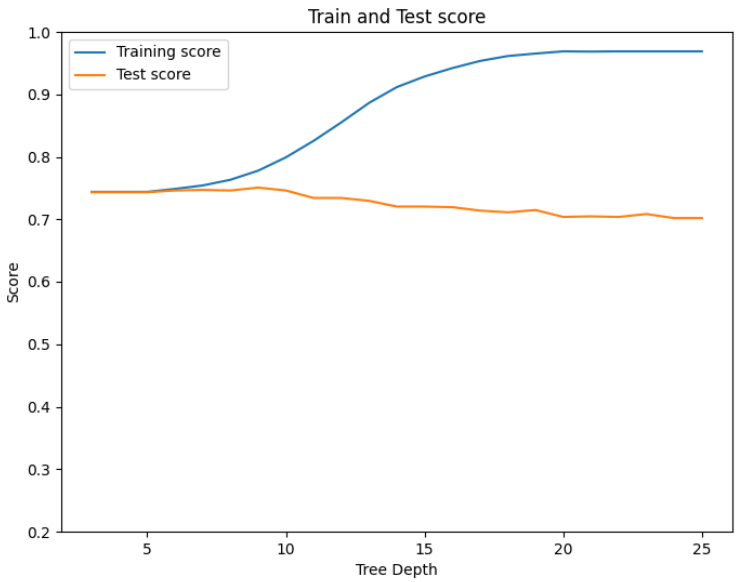
Conclusioni: dal grafico 'Train e Test score' si può ipotizzare che il modello stia diventando troppo complesso e stia iniziando a soffrire di overfitting.

Random Forest

Il Random Forest è un tipo di ensemble learning basato sulla tecnica del bagging, dove ogni modello è un albero di decisione casuale creato su un sottoinsieme casuale delle feature di addestramento. Il dataset considerato è stato suddiviso in test set e train set, con i quali si sono misurate le performance del modello, all'aumentare del numero di alberi.

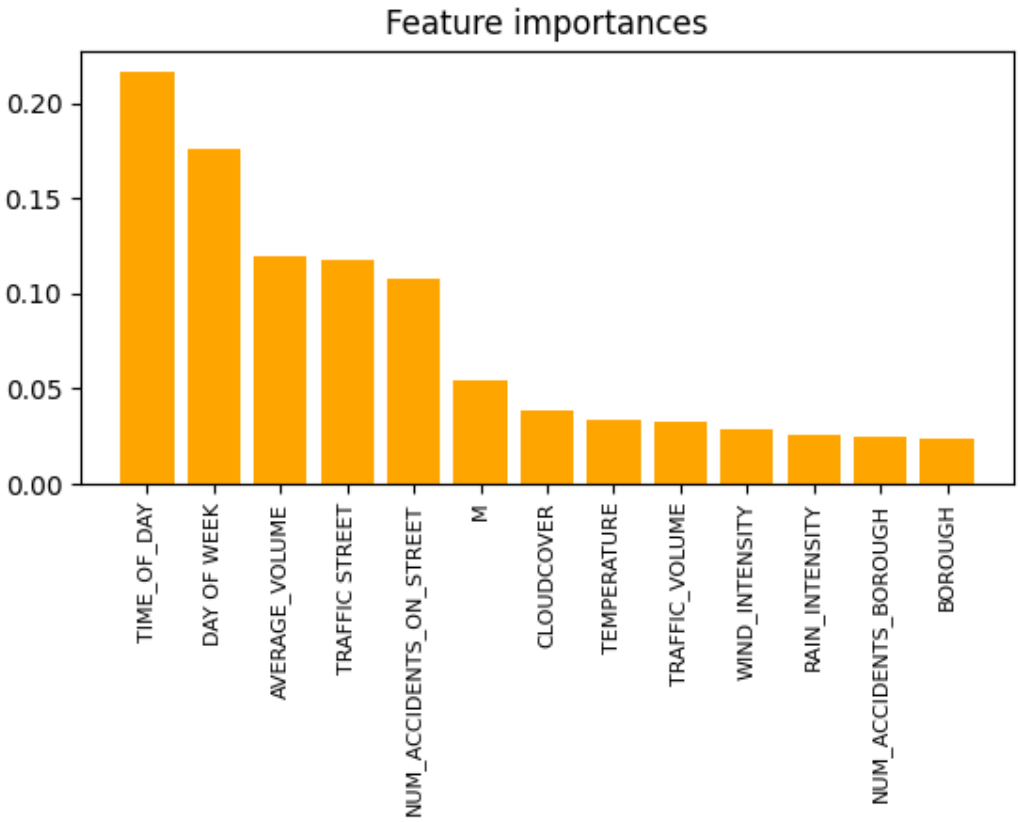


Da questo grafico si evince che l'accuratezza rimane costante all'aumentare del numero di alberi, e per questo motivo si è effettuata la stessa operazione calcolando le metriche di valutazione al variare della profondità.



Accuratezza	0.7255709771609137
Precisione	0.5821355297652572
Recall	0.5182030212137623
F1-score	0.484954386902098

Di seguito è riportato il grafico delle feature più importanti derivate dall'addestramento del modello.



Feature ranking:
1. TIME_OF_DAY (0.216728)
2. DAY OF WEEK (0.176219)
3. AVERAGE_VOLUME (0.119381)
4. TRAFFIC STREET (0.117763)
5. NUM_ACCIDENTS_ON_STREET (0.107641)

Conclusioni: rispetto agli alberi di decisione le metriche di valutazione sono migliori, ma osservando entrambi i grafici di 'Train e Test score' si può ipotizzare che il modello stia iniziando a soffrire di overfitting.

Naive Bayes

Il Naive Bayes categorico è un tipo di algoritmo di classificazione basato sulla teoria delle probabilità di Bayes che viene utilizzato per la classificazione di dati categorici. Quindi, sono state considerate solo le feature categoriche per la suddivisione in train set e test set.

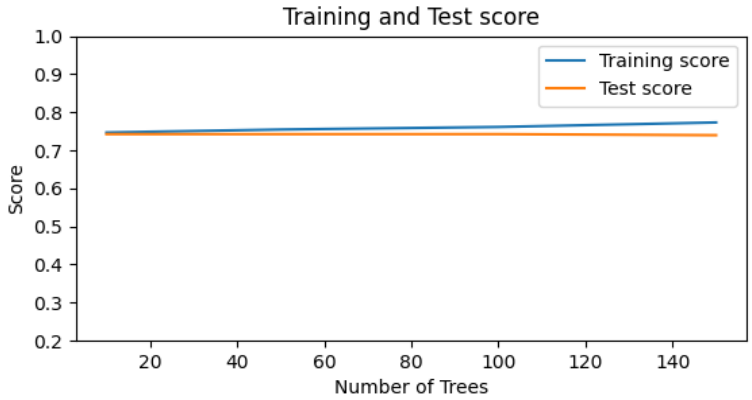
L'accuratezza calcolata sui dati di training è 0.7573007103393844, mentre sui dati di test sono riportate le seguenti metriche:

Accuratezza	0.7148114075436982
Precisione	0.5419181757209927
Recall	0.5171892189218922
F1-score	0.4973177950425453

Conclusioni: il Naive Bayes Categorico non fornisce una feature importance poiché utilizza una semplice regola di classificazione basata su probabilità condizionate e assume che le feature siano indipendenti tra loro. Inoltre le performance sono più basse rispetto al Random Forest.

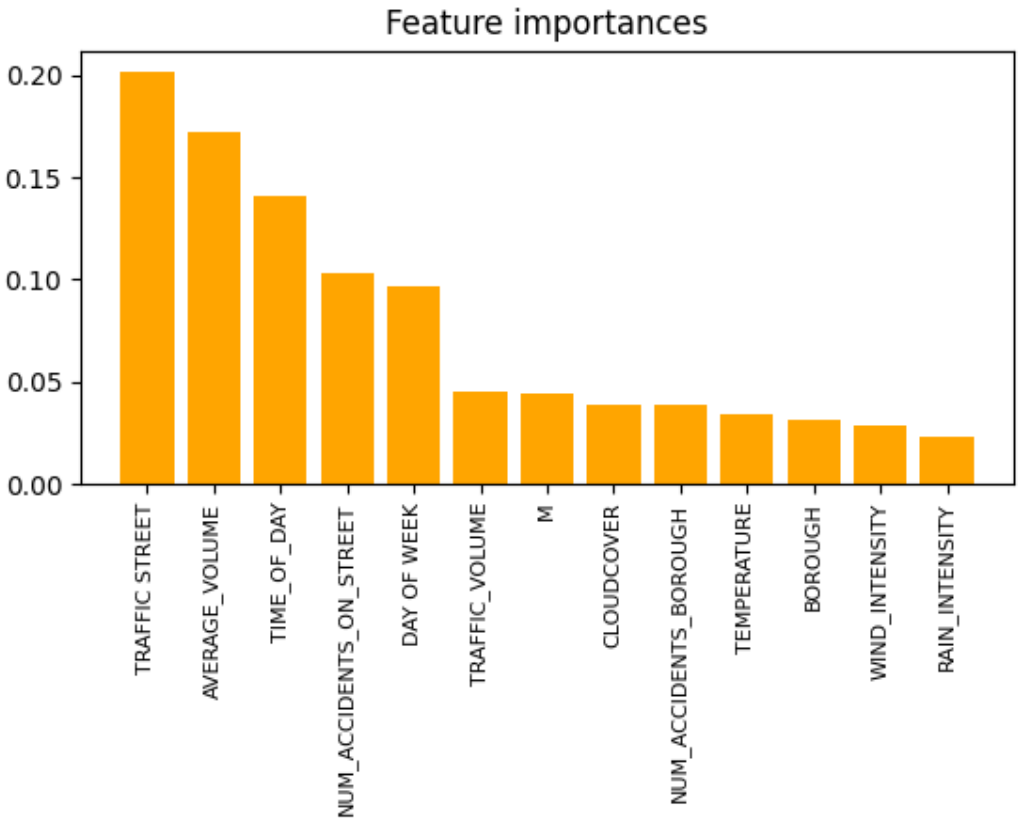
AdaBoost

AdaBoost (Adaptive Boosting) è un algoritmo di apprendimento automatico per la classificazione binaria e la regressione. L'idea alla base di AdaBoost è quella di combinare diversi classificatori deboli in un classificatore forte. Il dataset considerato è stato suddiviso in test set e train set, con i quali si sono misurate le performance del modello, all'aumentare del numero di alberi. La massima profondità è 3, una scelta comune, per evitare un overfitting del modello.



Accuratezza	0.7417203311867525
Precisione	0.5973046815046894
Recall	0.5080101760176018
F1-score	0.4510237688977675

Di seguito è riportato il grafico delle feature più importanti derivate dall'addestramento del modello.



Feature ranking:
1. TRAFFIC STREET (0.202227)
2. AVERAGE_VOLUME (0.172782)
3. TIME_OF_DAY (0.140584)
4. NUM_ACCIDENTS_ON_STREET (0.103468)
5. DAY OF WEEK (0.096934)

Conclusioni: le metriche di valutazione in questo modello sono leggermente migliori rispetto ai modelli precedenti. Inoltre, nel grafico 'Train e Test score' sono molto vicine, e ciò indica che il modello è in grado di adattarsi adeguatamente ai dati di addestramento e al contempo generalizzare bene su nuovi dati. Questo è un segno positivo, indicando che l'AdaBoost sta ottenendo buone prestazioni sui dati di test senza soffrire di overfitting.

Conclusioni generali

Osservando le conclusioni dei modelli si evince che tutti i modelli, tranne l'AdaBoost, soffrono di overfitting. In aggiunta, nel classificatore **AdaBoost** le metriche di valutazione hanno valori più elevati rispetto agli altri e quindi questo risulta il modello migliore.

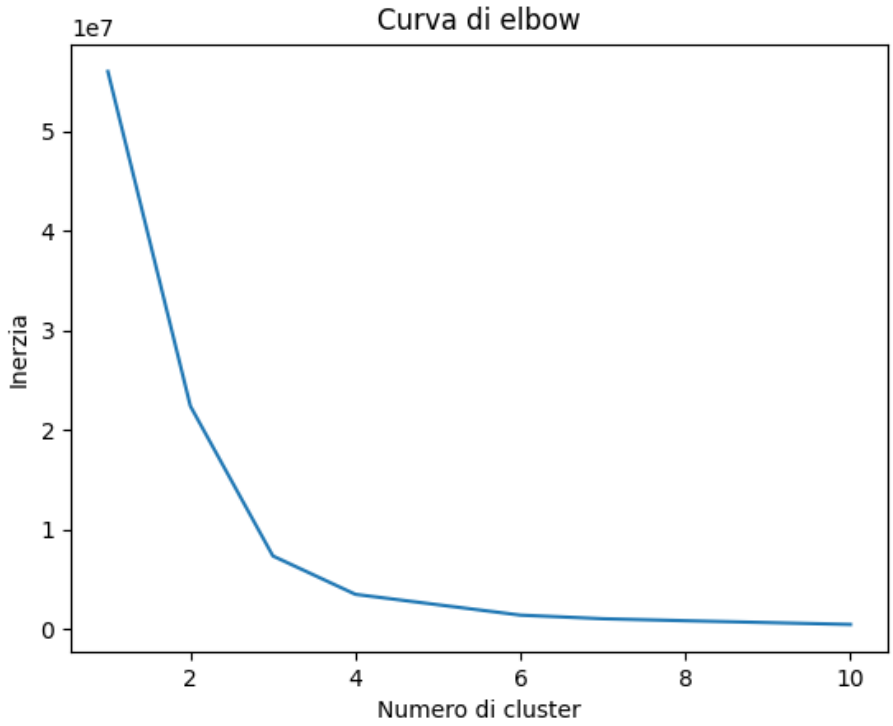
Quindi, si deduce che le feature più significative che influiscono nel determinare se un incidente ha causato feriti e morti, non considerando le caratteristiche direttamente collegate, sono: **TRAFFIC STREET**, **AVERAGE_VOLUME**, **TIME_OF_DAY, NUM_ACCIDENTS_ON_STREET** , **DAY OF WEEK**.

Apprendimento non supervisionato

Un altro task del caso di studio consiste nella creazione di un grafico che evidenzia le aree della città di New York, dove si verificano incidenti stradali con caratteristiche simili.
Per fare ciò è stato applicato il clustering, una tecnica di analisi dei dati utilizzata per raggruppare un insieme di oggetti in sottoinsiemi (cluster) in base alle loro somiglianze. Ci sono diversi algoritmi di clustering, tra questi è stato utilizzato il K-Means. Questo algoritmo cerca di suddividere il dataset in un numero prefissato di cluster (k) in modo che la distanza tra le osservazioni all'interno di ciascun cluster sia minima e la distanza tra i centroidi dei cluster sia massima. L'algoritmo inizia assegnando casualmente i punti ai cluster e poi ripete il processo di assegnazione finché i centroidi dei cluster non si stabilizzano.

Sono stati utilizzati i dataset **“Selected Accidents”** e **“generated_dataset”**, combinati mediante l’identificativo ‘COLLISION_ID’, e le feature considerate sono state suddivise in variabili categoriche e numeriche, in quanto le variabili categoriche devono essere convertite, per il corretto funzionamento del modello, mediante la funzione OrdinalEncoder.
Le feature categoriche sono ‘TEMPERATURE’, ’RAIN_INTENSITY’, ‘WIND_INTENSITY’, mentre le feature numeriche sono ‘NUMBER OF PERSONS INJURED’, ‘NUMBER OF PERSONS KILLED’, ‘CLOUDCOVER’, ‘AVERAGE_VOLUME’.
Quindi le aree sono state evidenziate in base a queste feature.

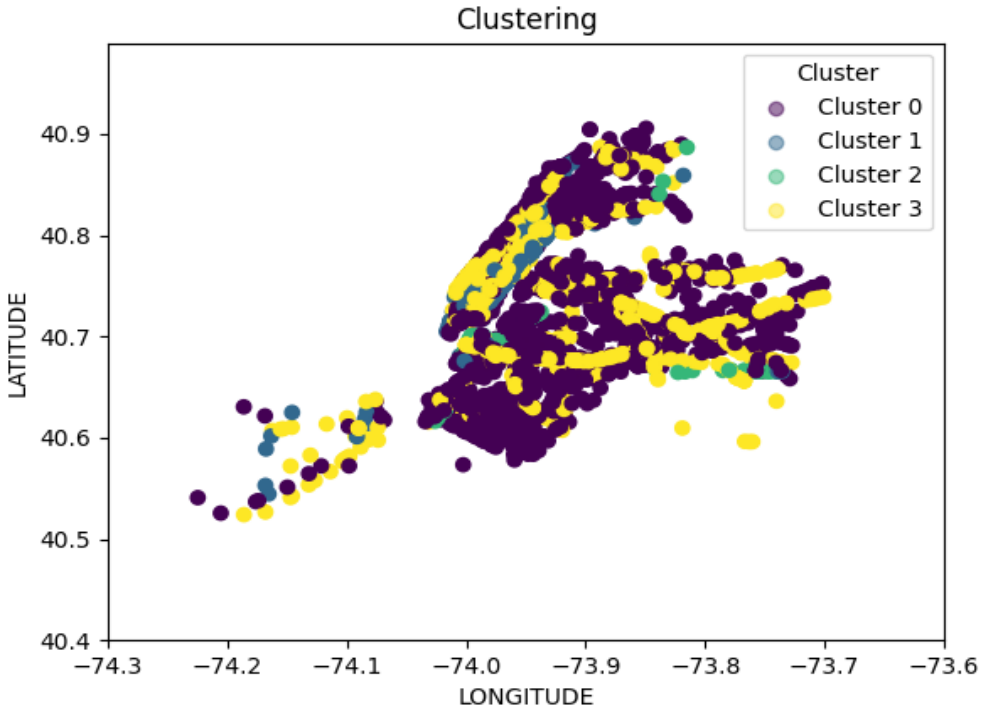
Per determinare il valore più appropriato di k (numero di cluster) si è utilizzata l’elbow rule: all’interno di esso vi è il punto di "gomito" sul grafico indica il numero di cluster in cui l'aggiunta di un ulteriore cluster non produce un miglioramento significativo dell'inerzia (cioè la somma dei quadrati delle distanze tra ogni osservazione e il centroide del suo cluster).



Di seguito il numero di cluster al variare dei due valori possibili di k:

k	0	1	2	3
3	1586	1931	104	-
4	1515	500	104	1502

I valori del numero di cluster non presentano differenze sostanziali al variare di k, ma si è optato per la scelta k = 4, che sembra fornire una migliore suddivisione.
Per visualizzare graficamente le aree in cui sono avvenuti incidenti stradali simili, è stato realizzato il grafico scatter seguente:



Caratteristiche per ogni cluster:

Cluster	TEMPERATURE	RAIN_INTENSITY	WIND_INTENSITY	NUMBER OF PERSONS INJURED	NUMBER OF PERSONS KILLED	CLOUDCOVER	AVERAGE_VOLUME
0	1.545215	3.058746	1.713531	0.368317	0.002640	0.165017	89.316343
1	1.508	3.088	1.7	0.236000	0.000000	0.296000	298.973860
2	1.740385	3.134615	1.5	0.576923	0.000000	0.221154	740.210385
3	1.521971	3.081891	1.711052	0.302264	0.000666	0.236352	195.672503

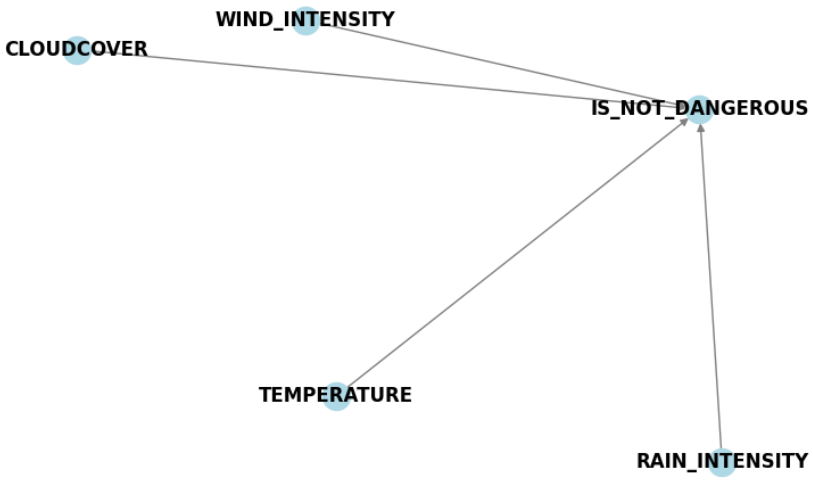
Per valutare la qualità del clustering, è stato calcolato l'indice di silhouette medio, che valuta quanto ogni osservazione all'interno di un cluster sia simile ad altre osservazioni nello stesso cluster rispetto a quelle presenti in cluster diversi. L'indice di Silhouette medio restituisce un valore compreso tra -1 e 1, dove un valore più vicino a 1 indica che i cluster sono ben separati e che le osservazioni all'interno di ogni cluster sono molto simili, mentre un valore più vicino a -1 indica che i cluster sono poco separati e che le osservazioni all'interno di ogni cluster sono poco simili.

Indice di silhouette medio	0.6918512307075179

Belief Network

Infine, l'ultimo task sviluppato è il calcolo della probabilità che l'evento 'IS_NOT_DANGEROUS' si verifichi dati i valori sul meteo; ovvero quello di calcolare la probabilità che un incidente sia pericoloso o meno, in base alle condizioni meteorologiche che si sono verificate.
Per fare ciò è stata realizzata una Belief Network, anche nota come rete bayesiana o rete di probabilità grafica, cioè un tipo di modello grafico probabilistico utilizzato per rappresentare e ragionare su relazioni di dipendenza probabilistica tra diverse variabili. Consiste in un grafo diretto aciclico (DAG) in cui i nodi rappresentano le variabili e gli archi rappresentano le relazioni di dipendenza probabilistica tra di esse. Ogni nodo ha associata una distribuzione di probabilità condizionata, che descrive la probabilità della variabile in funzione dei suoi genitori, ovvero le variabili a cui è direttamente collegata.

Sono stati utilizzati i dataset **"Selected Accidents"** e **"generated_dataset"**, combinati mediante l'identificativo 'COLLISION_ID', e le feature considerate sono state suddivise in variabili categoriche e booleane, in quanto le variabili categoriche devono essere convertite mediante la funzione OrdinalEncoder.
Le feature categoriche sono 'TEMPERATURE', 'RAIN_INTENSITY', 'WIND_INTENSITY', mentre le feature booleane sono 'IS_NOT_DANGEROUS' e 'CLOUDCOVER'.
Queste feature sono state aggiunte come nodi alla rete e sono stati definiti gli archi tra 'IS_NOT_DANGEROUS' e tutte le altre feature.



Dopo aver aggiunto i nodi alla rete, sono state create le tabelle di distribuzione di probabilità condizionata per ciascun nodo (CPT o TabularCPD), che rappresentano le probabilità condizionate delle variabili, cioè la probabilità che una variabile si verifichi dato lo stato delle sue variabili genitore.

Per ogni feature è stato definito il numero di possibili valori per ogni variabile, utilizzato in seguito per la creazione delle CPT. I valori assegnati sono:

'TEMPERATURE': 3, 'RAIN_INTENSITY': 5, 'WIND_INTENSITY': 3, 'CLOUDCOVER': 2, 'IS_NOT_DANGEROUS': 2

Per i nodi senza genitori, come ad esempio il nodo 'TEMPERATURE', sono state semplicemente calcolate le frequenze del nodo corrente basandosi sui valori osservati nel dataset.
Per il nodo con genitori, cioè 'IS_NOT_DANGEROUS', sono state calcolate le frequenze delle diverse combinazioni di valori dei genitori e del nodo corrente nel dataset.
Successivamente, le frequenze sono state normalizzate dividendole per la somma delle frequenze di ogni riga. Questo approccio si basa sul principio di Bayes per stimare le probabilità condizionate, che sono utilizzate per la creazione delle CPD. Ciò significa che la rete bayesiana tiene conto solo delle combinazioni di valori osservate nel dataset durante il ragionamento. Questo procedimento è stato realizzato mediante la classe 'TabularCPD' fornita dal modulo 'pgmpy', che consente di definire le distribuzioni di probabilità condizionata utilizzando tabelle di probabilità.
Le CPT create sono state poi aggiunte alla rete bayesiana.

Prima di procedere con il calcolo della probabilità dell'evento 'IS_NOT_DANGEROUS', dato un insieme di valori di evidenza, si verifica se la struttura del modello della rete bayesiana è valida.
Nel caso in cui sia valida viene creato un oggetto 'infer', utilizzando la tecnica di Variable Elimination, che sfrutta le proprietà delle reti bayesiane per ridurre la complessità computazionale. In seguito, vengono specificate le evidenze come un dizionario di valori, includendo le feature nominate precedentemente. Queste evidenze vengono organizzate in un DataFrame per consentire una manipolazione agevole e le variabili categoriche vengono codificate mediante OrdinalEncoder.
Viene eseguita l'inferenza esatta, utilizzando le evidenze per calcolare la probabilità condizionata dell'evento 'IS_NOT_DANGEROUS'.

Nel caso di studio, l'evidenza considerata è la seguente:
'TEMPERATURE':"mild", 'RAIN_INTENSITY':"weak", 'WIND_INTENSITY':"moderate", 'CLOUDCOVER':1

La probabilità risultante è:

IS_NOT_DANGEROUS	phi(IS_NOT_DANGEROUS)
IS_NOT_DANGEROUS(0)	0.3200
IS_NOT_DANGEROUS(1)	0.6800

Conclusioni

Il caso di studio comprende quattro task principali che spaziano tra diversi argomenti del corso di Ingegneria della Conoscenza. Questi task potrebbero essere utilizzati, per possibili estensioni, considerando i dati di traffico e meteo in tempo reale, con l'obiettivo di calcolare la probabilità e i fattori che influenzano maggiormente la gravità di un incidente stradale.