UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO | DIPARTIMENTO DI INFORMATICA

USED CAR PRICE PREDICTION

Documentazione Caso di Studio
AA 2022/2023

STUDENTI

Gadaleta Tiziano, mat. 741214, t.gadaleta2@studenti.uniba.it

Guzzo Marco, mat 739276, m.guzzo2@studenti.uniba.it

DATASET

https://www.kaggle.com/datasets/rakkesharv/used-cars-detailed-dataset

REPOSITORY GITHUB

https://github.com/marcoguzzo/progettoicon22-23.git

Sommario

Introduzione

Sapere il prezzo idoneo delle macchine riveste un ruolo molto importante nel mercato dell'usato, dato che l'utente riesce a determinare in modo preciso se l'automobile che sta osservando possa essere un buon affare o meno.

Inoltre l'utente potrebbe risultare spaesato tra la vastità degli annunci online per le macchine usate, per cui potrebbe trovare comodo un software che li consigli le macchine che possiedono delle caratteristiche da lui cercate.

Lo scopo principale di questo progetto è quello di utilizzare degli algoritmi di apprendimento supervisionato per la predizione del prezzo di una macchina usata, e la realizzazione di un CSP che ,date delle caratteristiche ricercate dall'utente, restituisce l'insieme delle macchine che potrebbe essere per lui di interesse.

Per facilitare l'utente nella scelta delle caratteristiche si è deciso di creare una knowledgebase per estrarre nuove informazioni.

Elenco argomenti di interesse

- Rappresentazione e ragionamento relazionale: attraverso il linguaggio prolog sono state create delle query che restituissero nuove informazioni, partendo dai dati contenuti nella knowledgebase;

- Apprendimento supervisionato: sono stati utilizzati diversi algoritmi di regressione per la predizione del prezzo di una macchina (KNN, AdaBoost, RandomForest, DecisionTree e GradientBoosting);

- CSP: sono stati creati vincoli e variabili che rappresentano gli interessi dell'utente, e attraverso l'algoritmo del backtracking, sono state restituite le macchine di interesse.

Preprocessing

Il dataset utilizzato contiene varie informazioni sulle macchine, utili al fine di rappresentarle in tutti i suoi aspetti.

Le caratteristiche presenti nel dataset sono: nome della macchina, anno di uscita, kilometri percorsi, il consumo di carburante, la coppia, la potenza, la cilindrata, il numero di posti a sedere, la tipologia di carrozzeria (berlina, suv, ecc), la capienza del serbatoio, il tipo di cambio (manuale o automatico), il numero di rapporti, la classe di emissioni, il tipo di carburante e infine il prezzo di vendita.

Su tale dataset sono state eseguite diverse operazioni.

Innanzitutto è stata rimossa la colonna CarName, poichè contenente informazioni già presenti nei campi Make ,Model e MakeYear.

La caratteristica “Mileage_Run” è stata trasformata in una colonna di numeri decimali, dato che inizialmente era una colonna di stringhe, per poi essere rinominata in “kilometers_run”.

Le stesse operazioni sono state eseguite per la caratteristica “Mileage(kmpl)”, rinominata in “kilometer_at_liter”, per comodità e chiarezza.

E' stata aggiunta un ulteriore caratteristica “Id”, per essere in grado di identificare univocamente le macchine, poichè il nome della macchina non è sufficiente, dato che esistono diversi annunci di vendita per la stessa macchina.

Come ultima operazione sono stati rimossi tutti gli spazi presenti nel dataset, dato che possono creare problemi durante la fase di regressione.
E’ stata inoltre eliminata la caratteristica “emission” poiché non facilmente esplicativa e difficile da comprendere per alcuni utenti.

E’ stata rimossa la caratteristica Engine_Type, poichè contenente informazioni poco rilevanti.

Il dataset modificato è stato salvato in “datasetfiltrato.csv”.

Nome dataset	Features categoriche	Features numeriche
Datasetfiltrato.csv	Make, Color, Model, Body_Type, Fuel_Type, Trasmission, Trasmission_Type.	ID, Make_Year, kilometers_run, No_of_Owners, Seating_Capacity, Fuel_Tank_Capacity(L), CC_Displacement, Power(BHP), Torque(Nm), kilometer_at_liter, Price.

Rappresentazione e ragionamento relazionale

E' stata creata una base di conoscenza, attraverso la libreria pyswip di Python, che permette di interfacciarsi direttamente con l'interprete di Prolog(Swi-Prolog).

E' stato definito un unico individuo car.

In ogni proprietà di car è presente un identificativo, che permette di ottenere, attraverso l'apposita query, ogni proprietà della macchina avente quell'identificativo.

```
prolog.assertz('make(ID, M) :- car(ID, M, _, _, _, _, _,_,_,_,_,_,_,_,_,_,_,_)')
prolog.assertz('model(ID, M) :- car(ID, _, M, _, _, _, _,_,_,_,_,_,_,_,_,_,_,_)')
prolog.assertz('make_year(ID, Y) :- car(ID, _, _, Y, _, _, _,_,_,_,_,_,_,_,_,_,_,_)')
prolog.assertz('color(ID, C) :- car(ID, _, _, _, C, _, _,_,_,_,_,_,_,_,_,_,_,_)')
prolog.assertz('body_type(ID, B) :- car(ID, _, _, _, _, B, _,_,_,_,_,_,_,_,_,_,_,_)')
prolog.assertz('kilometers_run(ID, K) :- car(ID, _, _, _, _, _, K, _,_,_,_,_,_,_,_,_,_,_)')
prolog.assertz('no_of_owners(ID, N) :- car(ID, _, _, _, _, _, _, N,_,_,_,_,_,_,_,_,_,_)')
prolog.assertz('seating_capacity(ID, S) :- car(ID, _, _, _, _, _, _, _,S,_,_,_,_,_,_,_,_,_)')
prolog.assertz('fuel_type(ID, T) :- car(ID, _, _, _, _, _, _, _,_,T,_,_,_,_,_,_,_,_)')
prolog.assertz('fuel_tank_capacity(ID, C) :- car(ID, _, _, _, _, _, _, _,_,_,C,_,_,_,_,_,_,_)')
prolog.assertz('cc_displacement(ID, C) :- car(ID, _, _, _, _, _, _, _,_,_,_,C,_,_,_,_,_,_)')
prolog.assertz('transmission(ID, T) :- car(ID, _, _, _, _, _, _, _,_,_,_,_,T,_,_,_,_,_)')
prolog.assertz('transmission_type(ID, T) :- car(ID, _, _, _, _, _, _, _,_,_,_,_,_,T,_,_,_,_)')
prolog.assertz('power(ID, P) :- car(ID, _, _, _, _, _, _, _,_,_,_,_,_,_,P,_,_,_)')
prolog.assertz('torque(ID, T) :- car(ID, _, _, _, _, _, _, _,_,_,_,_,_,_,_,T,_,_)')
prolog.assertz('kilometer_at_liter(ID, K) :- car(ID, _, _, _, _, _, _, _,_,_,_,_,_,_,_,_,K,_)')
prolog.assertz('price(ID, P) :- car(ID, M, _, _, _, _, _, _,_,_,_,_,_,_,_,_,_,P)')
```

Proprietà:

Le proprietà associate agli individui sono le feature presenti nel dataset.

- make(ID,M),
- model(ID,M),
- make_year(ID,Y),
- color(ID,C),
- body_type(ID,B),
- kilometers_run(ID,K),
- no_of_owners(ID,N),
- seating_capacity(ID,S),
- fuel_type(ID,T),

- fuel_tank_capacity(ID,C),
- cc_displacement(ID,C),
- transimmision(ID,T),
- transmission_type(ID,T),
- power(ID,P),
- torque(ID,T),
- kilometer_at_liter(ID,K),
- price(ID,P).

Clausole:

Al fine di facilitare l'inserimento delle caratteristiche da parte dell'utente sono state create delle clausole che permettessero la creazione di classi che possano essere di interesse per l'utente finale.

- Si prendono in analisi le caratteristiche che determinano la struttura del veicolo e il tipo di trasmissione della macchina. Da ciò possiamo rispondere alla domanda: la macchina è semplice da guidare? E' adatta ad una persona che ha scarsa esperienza?

```
% regola per definire se una macchina è semplice da portare
drive_easy(ID):- car(ID,_,_,_,_,Body_Type,_,_,_,_,_,_,_,Transmission_Type,_,_,_,_), Body_Type \== 2, Transmission_Type \== 1.
```

```
% regola per definire se una macchina non è semplice da portare
drive_not_easy(ID):- car(ID,_,_,_,_,Body_Type,_,_,_,_,_,_,_,Transmission_Type,_,_,_,_), Body_Type = 2, Transmission_Type = 1.
```

- Si prende in analisi l'anno di fabbricazione della macchina per determinare in quale classe di inquinamento euro rientra la macchina presa in considerazione. Viene quindi studiato l'anno per determinare la fasce di anni nelle quali sono state rilasciate macchine con una determinata classe di inquinamento euro.

```
% regola per definire la classe euro
euro0(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year<1993.

euro1(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year<1996.

euro2(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year>=1996, Make_Year<2000.

euro3(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year>=2000, Make_Year<2005.

euro4(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year>=2005, Make_Year<2009.

euro5(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year>=2009, Make_Year<2014.

euro6(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year>=2014.
```

- Si prende in analisi la tipologia di carrozzeria (Suv, berlina, ecc) e la capienza del serbatoio per determinare se una macchina è propensa a percorrere molti kilometri in comodità. In base a questo possiamo rispondere alla domanda: è una macchina adatta per viaggiare?

```
% REGOLA PER STABUILIRE SE E UNA MACCHINA ADATTA A FARE VIAGGI
travel(ID):-car(ID,_,_,_,_,Body_Type,_,_,_,_,Fuel_Tank_Capacity,_,_,_,_,_,_,_), Body_Type = 2, Fuel_Tank_Capacity > 50.
```

```
% REGOLA PER STABUILIRE SE NON E UNA MACCHINA ADATTA A FARE VIAGGI
not_travel(ID):-car(ID,_,_,_,_,Body_Type,_,_,_,_,Fuel_Tank_Capacity,_,_,_,_,_,_,_), Body_Type \== 2, Fuel_Tank_Capacity < 51.
```

- Prendiamo in analisi l'anno di fabbricazione della macchina. In base a questo dato possiamo dividere le vetture in auto più o meno recenti. In base a questo possiamo rispondere alle domande: è una macchina uscita negli ultimi anni? E' una macchina recente? E' una macchina "datata"?

```
% REGOLA PER STABILIRE IN QUALE RANGE DI ANNO RIENTRA UNA MACCHINA
old_years_car(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year<2014.

recent_years_car(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year<2020, Make_Year>2013.

new_years_car(ID):-car(ID,_,_,Make_Year,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_), Make_Year>2019.
```

- Analizziamo la potenza espressa in cavalli per poter suddividere le vetture in sportive o utilitarie. Studiamo quindi il numero di cavalli di una macchina per poter rispondere alle domande: E' una macchina sportiva? E 'una macchina utilitaria e meno potente?

```
% REGOLA PER DEFINIRE SE UN AUTO è SPORTIVA O UTILITARIA
sport_cars(ID):-car(ID,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,Power,_,_,_), Power>121.
utilitarian_cars(ID):-car(ID,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,Power,_,_,_), Power<122.
```

- Prendiamo in analisi la potenza espressa in cavalli per poter calcolare l'equivalente potenza della medesima vettura espressa in kilowatt. Una volta fatto ciò sarà possibile definire il costo del bollo analizzando la classe di inquinamento euro e la potenza espressa in kilowatt della macchina.

```
% regola per definire i kilowatt
kilowatt(ID, Kilowatt) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, Power, _, _, _),
    Kilowatt is Power * 0.7355.

% regola per definire il bollo auto per macchine euro0
rate_euro0(ID, Rate) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _),
    euro0(ID),
    kilowatt(ID, Kilowatt),
    (Kilowatt < 101 -> Rate is 3.00 * Kilowatt ; Result is Kilowatt - 100, Rate is 3.00 * 100 + Result * 4.50).


% regola per definire il bollo auto per macchine euro1
rate_euro1(ID, Rate) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _),
    euro1(ID),
    kilowatt(ID, Kilowatt),
    (Kilowatt < 101 -> Rate is 2.90 * Kilowatt ; Result is Kilowatt - 100, Rate is 2.90 * 100 + Result * 4.35).

% regola per definire il bollo auto per macchine euro2
rate_euro2(ID, Rate) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _),
    euro2(ID),
    kilowatt(ID, Kilowatt),
    (Kilowatt < 101 -> Rate is 2.80 * Kilowatt ; Result is Kilowatt - 100, Rate is 2.80 * 100 + Result * 4.35).
```

```
% regola per definire il bollo auto per macchine euro3
rate_euro3(ID, Rate) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _),
    euro3(ID),
    kilowatt(ID, Kilowatt),
    (Kilowatt < 101 -> Rate is 2.70 * Kilowatt ; Result is Kilowatt - 100, Rate is 2.70 * 100 + Result * 4.05).

% regola per definire il bollo auto per macchine euro4
rate_euro4(ID, Rate) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _),
    euro4(ID),
    kilowatt(ID, Kilowatt),
    (Kilowatt < 101 -> Rate is 2.58 * Kilowatt ; Result is Kilowatt - 100, Rate is 2.58 * 100 + Result * 3.87).

% regola per definire il bollo auto per macchine euro5
rate_euro5(ID, Rate) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _),
    euro5(ID),
    kilowatt(ID, Kilowatt),
    (Kilowatt < 101 -> Rate is 2.58 * Kilowatt ; Result is Kilowatt - 100, Rate is 2.58 * 100 + Result * 3.87,write(rate)).

% regola per definire il bollo auto per macchine euro6
rate_euro6(ID, Rate) :-
    car(ID, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _, _),
    euro6(ID),
    kilowatt(ID, Kilowatt),
    (Kilowatt < 101 -> Rate is 2.58 * Kilowatt ; Result is Kilowatt - 100, Rate is 2.58 * 100 + Result * 3.87).
```

- Prendiamo in analisi il consumo espresso in kilometri percorribili per litro per poter suddividere le autovetture in tre sottocategorie di consumi. In base a ciò sarà possibile rispondere alla domanda: è una macchina che consuma poco? E' una macchina che consuma tanto? E' una macchina con consumi medi?

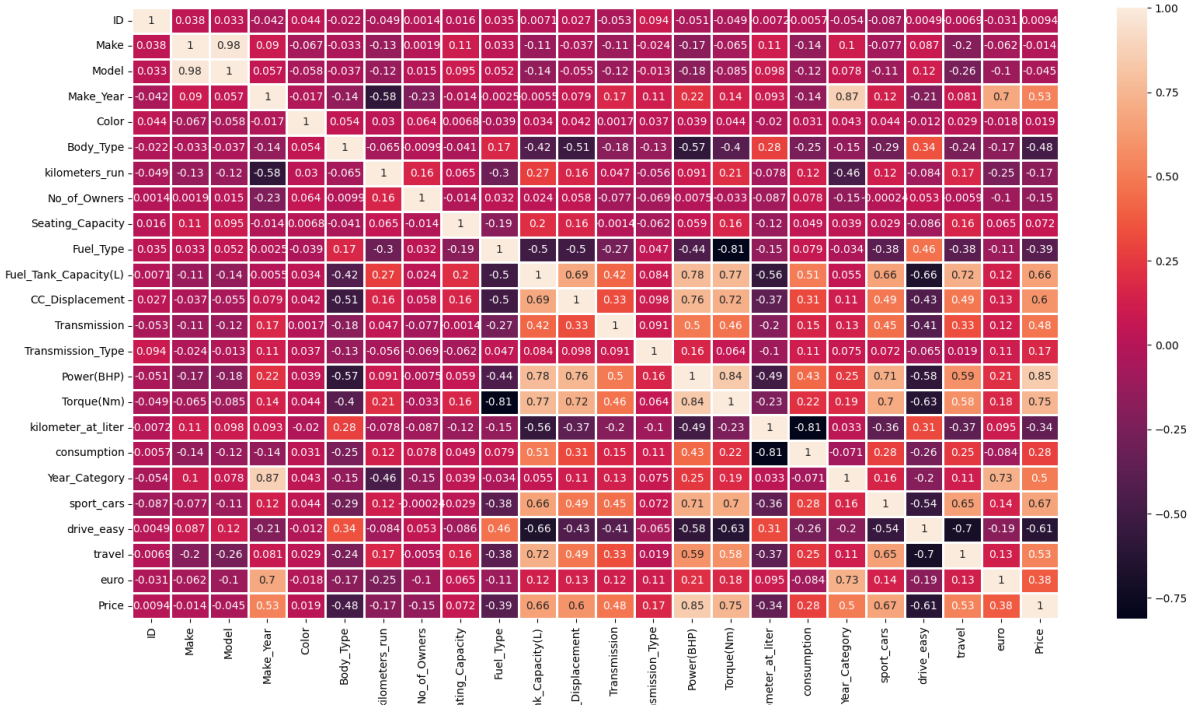
```
% regola per definire se una macchina è a bassi consumi
low_fuel_consumption(ID) :- car(ID,_,_,_,_,_,_,_,_,_,_,_,_,_,_,KilometerAtLiter,_), KilometerAtLiter > 19.9.

% regola per definire se una macchina ha dei consumi giusti
medium_fuel_consumption(ID) :- car(ID,_,_,_,_,_,_,_,_,_,_,_,_,_,_,KilometerAtLiter,_), KilometerAtLiter > 10.0, KilometerAtLiter < 20.0.

% regola per definire se una macchina ha dei consumi alti
high_fuel_consumption(ID) :- car(ID,_,_,_,_,_,_,_,_,_,_,_,_,_,_,KilometerAtLiter,_), KilometerAtLiter < 10.1.
```

Apprendimento supervisionato

Una volta aggiunta la nuova conoscenza all'interno del dataset "datasetoperativo.csv", procediamo con la valutazione di diversi modelli di regressione (KNN, Decision Tree, Ada Boost, Random Tree e Gradient Boosting) al fine di individuare il più adatto ai nostri scopi.

Correlazione tra caratteristiche:

Nell'immagine sottostante sono riportate le correlazioni, espresse in forma numerica, tra ogni coppia di variabili.



In particolare ci soffermiamo sulla correlazione tra la variabile target e le altre caratteristiche

Price	1.000000
Power(BHP)	0.845615
Torque(Nm)	0.750300
sport_cars	0.667676
Fuel_Tank_Capacity(L)	0.660050
drive_easy	0.613977
CC_Displacement	0.603398
travel	0.528788
Make_Year	0.527597
Year_Category	0.500079
Body_Type	0.478838
Transmission	0.477879
Fuel_Type	0.386948
euro	0.377115
kilometer_at_liter	0.344536
consumption	0.277405
Transmission_Type	0.172258
kilometers_run	0.168158
No_of_Owners	0.149180
Seating_Capacity	0.072004
Model	0.025905
Color	0.018730
Make	0.013966
ID	0.009379

Preprocessing

Il file “datasetfiltrato.csv” è stato sottoposto ad una ulteriore fase di preprocessing, dove è stata applicata una fase di codifica per poter trasformare le variabili categoriche in variabili discrete, mappando ogni singolo valore che essa può assumere ad un unico valore numerico.

Si è deciso di optare per questa soluzione, poiché utilizzando la codifica onehot encoding, il numero di variabili presenti nel dataset sarebbe aumentato notevolmente, complicando così la fase di regressione.

Infine è stata rimossa la colonna 'Id' poiché non ha valore informativo per la predizione del prezzo.

Sono state aggiunte tutte le features ritenute più rilevanti per la predizione del prezzo, sostituendole alla corrispettive features originali.

In particolare sono state aggiunte le caratteristiche travel(valore 0 se non è una macchina adatta viaggiare 1 altrimenti),sport_cars(ha valore 1 se la macchina è una sportiva 0 se è una utilitaria),

consumption(ha valore 0 per indicare i bassi consumi, 1 per indicare i medi consumi, e 2 per indicare gli alti consumi) ,Year_Category(ha valore 0 per indicare una macchina uscita prima del 2014, ha valore 1 per indicare una macchina uscita tra il 2014 e il 2019, ha valore 2 per indicare una macchina uscita dopo il 2019), la classe di inquinamento euro indicata con 'euro'(ha come valore il corrispondente numero di classe di inquinamento), Car_Tax (bollo dell'auto) e drive_easy(1 se è una macchina semplice da portare 0 altrimenti).

Le fature power, make_year,kilometer_at_liter sono state rimosse, e sostituite dalle corrispondenti feature sport_cars,Year_Category,consumption.

Ogni variabile del dataset non binaria è stata normalizzata attraverso il MinMaxScaler, così da aumentare le prestazioni dell'algoritmo KNN, poiché tale algoritmo richiede che le variabili presenti all'interno del dataset utilizzato per la predizione abbiano una scala simile di valori.

Valutazione dei modelli

In un primo momento è stata eseguita la valutazione di ogni regressore senza apportare modifiche ai parametri di ognuno, e senza l'utilizzo della K fold cross validation, dunque ogni regressore è stato addestrato e valutato esclusivamente su due set di dati, ovvero training set e test set+.

In seguito sono riportate l'importanza delle feature per ogni regressore.

```
Importanza delle feature per randomforest:
sport_cars : 0.38649419357878984
Torque(Nm) : 0.22366057368490874
Year_Category : 0.09369968996677241
kilometers_run : 0.081883822206942
Fuel_Tank_Capacity(L) : 0.0813217463795434
Model : 0.02276456327415823
Body_Type : 0.022316430498942698
Make : 0.01582322809197866
CC_Displacement : 0.015052707185933585
Transmission : 0.012581877147484622
euro : 0.012407208268412963
Color : 0.009919593162974902
drive_easy : 0.006071281859259967
No_of_Owners : 0.005798254404265468
consumption : 0.003282280245556377
Transmission_Type : 0.002635265447381003
Fuel_Type : 0.0024072604190422163
Seating_Capacity : 0.0014463622516075047
travel : 0.0004336619260454252
```

```
Importanza delle feature per adaboost:
sport_cars : 0.22369212452497178
kilometers_run : 0.20761091823825892
Torque(Nm) : 0.17227989575185101
Fuel_Tank_Capacity(L) : 0.11419804568980628
Year_Category : 0.07598675113166958
Transmission : 0.05715692928658028
Body_Type : 0.05110896792413559
Make : 0.030316277825007173
Model : 0.021903418568836038
drive_easy : 0.010756087518411036
Fuel_Type : 0.008009387763611412
Transmission_Type : 0.007658327101734437
euro : 0.00764855048895813
CC_Displacement : 0.006310547286777924
Color : 0.005361074043181999
No_of_Owners : 2.6968562085317414e-06
consumption : 0.0
Seating_Capacity : 0.0
travel : 0.0
```

```
Importanza delle feature per knn:
Torque(Nm): 0.24451583559911208
sport_cars: 0.152695279583615
Fuel_Tank_Capacity(L): 0.14654928617148616
CC_Displacement: 0.10868719721964357
travel: 0.07368295552157632
Year_Category: 0.06330371968808719
Body_Type: 0.056474346422081864
Transmission: 0.056181324634448235
Fuel_Type: 0.03342838996275251
euro: 0.031472807381464504
consumption: 0.015826088572095366
Transmission_Type: 0.0058050430042008196
kilometers_run: 0.005524060736745781
No_of_Owners: 0.004320768455134162
Seating_Capacity: 0.0009893154058511371
Model: 0.0003848478175452754
Color: 6.661517546139406e-05
drive_easy: 5.508718810632465e-05
Make: 3.703146059241238e-05
```

```
Importanza delle feature per decisiontree:
sport_cars : 0.44579153277027667
Torque(Nm) : 0.16120898571813186
Year_Category : 0.12342794054346266
kilometers_run : 0.088863401220204
Fuel_Tank_Capacity(L) : 0.07404198277607238
Transmission : 0.018629204174453386
Body_Type : 0.017679081373927708
Make : 0.013664464955949848
CC_Displacement : 0.01332730195280603
Color : 0.0102402842322454
euro : 0.006875458828950179
Fuel_Type : 0.006298820748700813
drive_easy : 0.006174224896997728
Model : 0.005541162620190262
consumption : 0.002782942142112524
No_of_Owners : 0.0024036964252515656
Transmission_Type : 0.0017122350626532295
Seating_Capacity : 0.001197206325167095
travel : 0.00014007323244686557
```

```
Importanza delle feature per gradientboosting:
sport_cars : 0.3130775210191326
Torque(Nm) : 0.2580997475316982
Year_Category : 0.10611916487674383
kilometers_run : 0.07856327201468759
Body_Type : 0.07260341513625544
Fuel_Tank_Capacity(L) : 0.0602400081094875
Transmission : 0.03322279741245719
Model : 0.03181154728246404
euro : 0.013247906061910132
CC_Displacement : 0.00836847221517129
No_of_Owners : 0.007719903777088912
Make : 0.0077104617579284
Transmission_Type : 0.0023758108894584434
consumption : 0.001748214750238941
Color : 0.0016972125084873289
Fuel_Type : 0.001458580340247027
Seating_Capacity : 0.001001699573238293
drive_easy : 0.000704247353729797
travel : 0.00023001738957504633
```

I punteggi delle metriche di valutazione ottenuti da ognuno sono i seguenti:

```
ADABOOST:
Average MSE Score: 0.0017535712978927235,
Average MAE Score: 0.025437249340088945,
Average R^2 Score: 0.8712558616416552,
```

```
DECISION TREE:
Average MSE Score: 0.0031958296613599 15,
Average MAE Score: 0.03335767365226361,
Average R^2 Score: 0.8067491596590664,
```

```
GRADIENT BOOSTING:
Average MSE Score: 0.0016904162672069013,
Average MAE Score: 0.027003678613932252,
Average R^2 Score: 0.897781046307484,
```

```
KNN:
Average MSE Score: 0.003813981138122 84,
Average MAE Score: 0.038356513819906,
Average R^2 Score: 0.7693697292761618,
```

```
RANDOM FOREST:
Average MSE Score: 0.0021067062558047 65,
Average MAE Score: 0.028217028129104618,
Average R^2 Score: 0.8726081182585523,
```

N.B

Le metriche di valutazione adottate sono la MSE, MAE, e R^2.
E' stato scelto di utilizzare la MAE e la MSE per avere due visioni diverse sulla predizione del prezzo della macchina, una che penalizza gli errori maggiori (MSE) e l'altra che attribuisce uno stesso grado di importanza agli errori.
La metrica di valutazione R^2, invece, è stata adottata per catturare la variabilità dei dati, poiché essa è molto importante per eseguire delle predizioni accurate.

Inoltre per ogni regressore si è deciso di mostrare l'importanza di ciascuna feature, prima e dopo aver eseguito l'ottimizzazione degli iperparametri .

Per il regressore KNN si è scelto di utilizzare il test anova, mentre per gli altri regressori è stato utilizzato un attributo presente nelle rispettive classi che indicasse direttamente l'importanza di ciascuna feature.

In seguito sono riportati i punteggi di valutazione di un esecuzione.

Analizzando i risultati della regressione per i diversi modelli, possiamo trarre le seguenti conclusioni:

- Ada Boost: ha ottenuto buoni punteggi per le metriche di valutazione MAE (0.025437249340088945), MSE (0.0017535712978927235), il che indica che il modello riesce a predire con una buona accuratezza i valori. Ha un punteggio R^2 pari a 0.8712558616416552, il che sta a significare che riesce a spiegare la variazione dei dati nel set di test.

- Decision Tree: ha ottenuto buoni punteggi per le metriche di valutazione MAE (0.03335767365226361), e MSE (0.003195829661359915) , il che indica che il modello riesce a predire i valori con una accuratezza più bassa rispetto all'Ada Boost. Ha un punteggio R^2 pari a 0.8067491596590664, il che significa che riesce a spiegare la variabilità dei dati in maniera meno efficace rispetto all'Ada Boost.

- Gradient Boosting: ha ottenuto buoni punteggi per le metriche di valutazione MAE (0.027003678613932252), e MSE (0.0016904162672069013) , il che indica che il modello ha un punteggio in termini di errore medio-assoluto leggermente più alto rispetto all'Ada Boost, presenta un errore quadratico leggermente più basso rispetto all'Ada Boost. Ha un punteggio R^2 pari a 0.897781046307484, dunque possiede una capacità di cattura della variabilità dei dati ancora più alta rispetto all'Ada Boost.

- KNN: ha ottenuto buoni punteggi per le metriche di valutazione MAE (0.038356513819906),e MSE (0.00381398113812284), il che indica che il modello ha un errore medio-assoluto molto più grande rispetto all'Ada Boost, l'errore quadratico invece è leggermente più alto rispetto all'Ada Boost. Ha un punteggio R^2 pari a 0.7693697292761618, che sta a significare che ha una minor capacità di cattura della variabilità dei dati rispetto al Gradient Boosting.

- Random Forest: ha ottenuto buoni punteggi per le metriche di valutazione MAE (0.028383357785533192), e MSE (0.0021153437306229085) , quindi presenta un'accuratezza leggermente inferiore rispetto all'Ada Boost. Ha un punteggio R^2 pari a 0.8726081182585523, dunque ha una minor capacità di cattura della variabilità dei dati rispetto al Gradient Boosting.

Complessivamente tutti i modelli hanno ottimi punteggi di valutazione, ma quella che risulta essere complessivamente(ovvero quello che ha il numero di punteggi di valutazione più alti) la soluzione migliore è il Gradient Boosting.

K Fold Cross Validation

Per ottenere una valutazione più accurata dei modelli è stato scelto di utilizzare il k fold cross validation.

Ciò poiché la cross validation suddivide il dataset di partenza in diverse fold, una utilizzata per il test e le altre utilizzate per l'addestramento, per poi valutare le prestazioni di ogni modello su ogni fold creato.

Prima della valutazione dei regressori si è ritenuto opportuno stabilire il numero di fold ottimale, attraverso la metrica di valutazione mean squared error.

A seguito di vari test il numero di fold ottimale risulta essere 10.

Di seguito sono riportati i punteggi per ognuna delle metriche adottate per ciascun regressore:

```
ADA:
Average MSE Score: 1.152545780741894e-05,
Average MAE Score: 0.0011687549646140273,
Average R2 Score: 0.9993553529213166,
```

```
KNN:
Average MSE Score: 0.0033549872067836 61,
Average MAE Score: 0.038209026716109894,
Average R2 Score: 0.8138191975862377,
```

```
Gradient Boosting Regressor:
Average MSE Score: 0.0006265652772137399,
Average MAE Score: 0.01894495772754389,
Average R2 Score: 0.9647726244639359,
```

```
RANDOM FOREST:
Average MSE Score: 0.00022546788276483787,
Average MAE Score: 0.008815911718745004,
Average R2 Score: 0.9877383686382137,
```

```
DECISION TREE:
Average MSE Score: 0.0,
Average MAE Score: 0.0,
Average R2 Score: 1.0,
```

Il modello che ha ottenuto i risultati migliori per tutte le metriche è il decision tree, il quale ha ottenuto un punteggio di MAE e MSE pari a 0.0 e un R2 score pari a 1, dunque riesce ad adattarsi bene ai dati e riesce a catturare perfettamente la variabilità dei dati.

Ottimi risultati sono stati raggiunti anche dall'AdaBoostRegressor ha ottenuto un punteggio MSE(1.152545780741894e-05),MAE(0.0011687549646140273), e R2(0.9993553529213166).

Il Gradient Boosting Regressor ha ottenuto anch'esso delle buone performance, con una MAE(0.01894495772754389),MSE(0.0006265652772137399) e R2(0.9647726244639359).

Il randomforest ha ottenuto dei punteggi MSE(0.00022546788276483787), MAE(0.008815911718745004), e R2(0.9877383686382137).

Il regressore che ha ottenuto le performance peggiori è il KNN, che presenta un MSE(0.003354987206783661), MAE(0.038209026716109894) e R2(0.8138191975862377).

Se andiamo a confrontare i risultati ottenuti con un semplice addestramento sul training set, e quelli ottenuti con la metodologia 10-fold cross-validation possiamo notare le seguenti differenze:

- Ada Boost: dopo aver applicato la K fold cross validation ha ottenuto notevoli miglioramenti, ciò significa che il modello Ada Boost non riesce a generalizzare correttamente su dati mai visti (overfitting).
- Decision Tree: anche questo modello ha ottenuto notevoli miglioramenti dopo l'applicazione della k fold, di conseguenza si può dire che anche il Decision Tree non abbia una buona capacità di generalizzazione su nuovi dati.
- Gradient Boosting: in questo sono stati osservati leggeri miglioramenti sulle metriche di valutazione MAE e R^2, mentre per quanto riguarda la metrica MSE sono stati osservati miglioramenti elevati. Dato ciò, si può dire che
- KNN: questo modello, a differenza dei precedenti analizzati, risente meno del problema dell'overfitting, poichè sono stati ottenuti solo leggeri miglioramenti rispetto alla valutazione senza il K fold.
- Random Forest: come nei modelli citati sopra (tolto il modello KNN) ha ottenuto notevoli miglioramenti di prestazioni.

In conclusione il regressore che ha ottenuto il miglioramento più significativo per tutte le metriche di valutazione è il Decision Tree, quindi è il regressore che soffre maggiormente del problema dell'overfitting.

Valutazione e ottimizzazione degli iperparametri

Successivamente si è scelto di stabilire quali fossero i parametri migliori per ogni regressore. Ciò è stato realizzato tramite l'applicazione dell'algoritmo Grid Search, utilizzando come metrica l'MSE e, per ogni combinazione di parametri, è stata applicata la Cross Validation.

Di seguito sono illustrati i range di valori da testare fissati per ogni regressore:

- Ada Boost: per l'Ada Boost Regressor, i parametri selezionati per la Grid Search sono il numero di stimatori (n_estimators) e il learning rate (learning_rate). Sono stati testati valori compresi tra 50 e 200 per il numero di stimatori al fine di valutare come il modello si adatta ai diversi numeri. Servendosi di un range così ampio per testare i valori per gli stimatori permette di mantenere invariata la complessità del regressore e, al tempo stesso, migliorare la capacità di predizione del modello.
  Per quanto riguarda il learning_rate sono stati testati i valori [0.01,0.1,1].

  Per il learning rate i valori più bassi influiranno sul modello, riducendo l'influenza di ogni stimatore e, al tempo stesso, richiedendo un numero maggiore di stimatori per ottenere un'accuratezza buona.

- Decision Tree: per il Decision Tree Regressor, il parametro selezionato per la Grid Search è la max_depth, ovvero la massima profondità dell'albero, mentre il range di valori da testare è stato scelto tra 3 e 10. Questa volta il range di valori si riduce rispetto allo scorso modello, questo viene fatto per trovare il giusto compromesso tra la capacità del modello di adattarsi ai dati e la capacità del modello di generalizzare su dati nuovi. Tramite il parametro legato alla massima profondità si opera per prevenire il problema dell'overfitting, in più si controlla il livello di complessità dell'albero.
- Gradient Boosting: per quanto riguarda il Gradient Boosting Regressor, sono stati selezionati come parametri per la Grid Search il numero di stimatori (n_estimators) e il learning rate (learning_rate). I valori testati per il numero di stimatori sono compresi tra 50 e 200, mentre per learning rate sono stati scelti i valori 0.01, 0.1 e 1.0. Servendosi di un range ampio per testare i valori sul numero di stimatori, ciò permette di mantenere invariata la complessità del regressore e, al tempo stesso, migliorare la capacità di predizione del modello.
- KNN: per K-Nearest Neighbors Regressor, il parametro scelto per la Grid Search è il numero di vicini (n_neighbors) e i valori testati sono compresi tra 3 e 10. Anche in questo caso è fondamentale trovare un equilibrio tra un numero minimo di vicini per ottenere una buona precisione e un numero troppo alto che causerebbe una riduzione della sensibilità del modello in questione. In questo caso i valori relativamente bassi sono stati scelti per ridurre la possibilità di prendere in analisi valori che sono rumore, senza complicare eccessivamente il modello considerando che si sta lavorando su un dataset di grosse dimensioni. Nonostante ciò un numero più alto di vicini può ridurre il rischio di overfitting.
- Random Forest: per il Random Forest Regressor, i parametri selezionati per la Grid Search sono la massima profondità dell'albero (max_depth) e il numero di stimatori (n_estimators). Per quanto riguarda il numero di stimatori, sono stati testati valori compresi tra 50 e 200, mentre per la massima profondità sono stati testati valori compresi tra 5 e 20. L'obiettivo è bilanciare la complessità del modello con la sua capacità di generalizzare su nuovi dati.

Sono riportati i migliori parametri per ogni regressore, riportando i punteggi delle metriche di valutazione:

- Ada Boost Regressor best parameters: {'learning_rate': 1.0, 'n_estimators': 100}

  Average mae Score: 0.0220, Average mse Score: 0.0015, Average R2 Score: 0.9230.

- Decision Tree best parameters: {'max_depth': 7}

  Average mae Score: 0.0313, Average mse Score: 0.0022, Average R2 Score: 0.8760.

- Gradient Boosting Classifier best parameters: {'learning_rate': 0.1, 'n_estimators': 200}

  Average mae Score: 0.0237, Average mse Score: 0.0017, Average R2 Score: 0.9098.

- KNN best parameters: {'n_neighbors': 4}

  Average mae Score: 0.0362, Average mse Score: 0.0030, Average R2 Score: 0.8417.

- Random Forest best parameters: {'max_depth': 20, 'n_estimators': 100}

  Average mae Score: 0.0243, Average mse Score: 0.0017, Average R2 Score: 0.9078.

Notiamo che il miglioramento più significativo è stato ottenuto dal regressore adaboost, ottimi miglioramenti sono stati raggiunti anche dai regressori Random Forest,DecisionTree invece

i regressori GradientBoosting e KNN hanno ottenuto dei miglioramenti molto trascurabili.

Ciascun regressore è stato infine addestrato sul training set con i parametri migliori, e valutato sul testset.

In seguito sono mostrati i punteggi di valutazione

```
Valutazione dei modelli sul test set:

KNN
R2: 0.8597220465303681
mse: 0.0021984931907967358
mae: 0.032187264243203215
Decision Tree
R2: 0.8526810538345089
mse: 0.002308842494557053
mae: 0.028562756154376617
Random Forest
R2: 0.9160724338637933
mse: 0.0013153469815236332
mae: 0.022906955582255268
Ada Boost Classifier
R2: 0.9407138901166079
mse: 0.00092915604814324
mae: 0.01986085109949984
Gradient Boosting Classifier
R2: 0.938068626113546
mse: 0.0009706137024270 99
mae: 0.022232409538152018
```

Tutti i modelli,addestrati con i loro iperparametri migliori, hanno ottenuto dei leggeri miglioramenti rispetto ai modelli addestrati con parametri di default.

In seguito, è stato scelto di effettuare gli stessi test con gli stessi parametri (per la Grid Search) selezionati per i vari modelli ma con range di valori più ampi.

```
regressors = [
    ('KNN', KNeighborsRegressor(), {'n_neighbors': range(2, 11)}),
    ('Decision Tree', DecisionTreeRegressor(), {'max_depth': range(3, 21)}),
    ('Random Forest', RandomForestRegressor(), {'n_estimators': range(50, 301, 50), 'max_depth': range(5, 31, 5)}),
    ('Ada Boost Regressor', AdaBoostRegressor(estimator=DecisionTreeRegressor()), {'n_estimators': range(50, 301, 50), 'learning_rate': [0.01, 0.1, 1.0]}),
    ('Gradient Boosting Regressor', GradientBoostingRegressor(), {'n_estimators': range(50, 201, 50), 'learning_rate': [0.01, 0.1, 1.0]})
]
```

I risultati sono i seguenti:

- KNN best parameters: {'n_neighbors': 2}

Average mae Score: 0.0342, Average mse Score: 0.0030, Average R2 Score: 0.8417.

- Decision Tree best parameters: {'max_depth': 8}

Average mae Score: 0.0293, Average mse Score: 0.0021, Average R2 Score: 0.8834.

- Random Forest best parameters: {'max_depth': 25, 'n_estimators': 200}

Average mae Score: 0.0241, Average mse Score: 0.0016, Average R2 Score: 0.9114.

- Ada Boost Regressor best parameters: {'learning_rate': 1.0, 'n_estimators': 150}

Average mae Score: 0.0218, Average mse Score: 0.0015, Average R2 Score: 0.9230.

- Gradientboosting ha ottenuto gli stessi iperparametri.

I punteggi riportati sul test set dopo l'addestramento sul training set con i migliori parametri per ognuno sono i seguenti:

```
Valutazione dei modelli sul test set:

KNN
R2: 0.8597220465303681
mse: 0.0021984931907967358
mae: 0.032187264243203215
Decision Tree
R2: 0.8808849535385602
mse: 0.0018668194972177431
mae: 0.026843931950976008
Random Forest
R2: 0.9205052440965675
mse: 0.0012458741750575914
mae: 0.022167890377602 48
Ada Boost Classifier
R2: 0.9422333382920719
mse: 0.0009053426378039656
mae: 0.01986457663900453
Gradient Boosting Classifier
R2: 0.9388604595359886
mse: 0.0009582037666928838
mae: 0.02219813152652697
```

Confrontando i punteggi tra i parametri ottenuti ora con quelli ottenuti precedentemente, notiamo che le prestazioni per il knn sono rimaste invariate, il decisiontree,randomforest hanno ottenuto lievi miglioramenti, mentre l'adaboost e il gradientboosting hanno ottenuto dei miglioramenti molto trascurabili.

Infine si è ricalcolata la feature importance per ogni regressore

```
Importanza delle feature per knn:
Torque(Nm): 0.24451583559911208
sport_cars: 0.152695279583615
Fuel_Tank_Capacity(L): 0.14654928617148616
CC_Displacement: 0.10868719721964357
travel: 0.07368295552157632
Year_Category: 0.06330371968808719
Body_Type: 0.056474346422081864
Transmission: 0.056181324634448235
Fuel_Type: 0.03342838996275251
euro: 0.031472807381464504
consumption: 0.015826088572095366
Transmission_Type: 0.0058050430042008196
kilometers_run: 0.005524060736745781
No_of_Owners: 0.004320768455134162
Seating_Capacity: 0.0009893154058511371
Model: 0.0003848478175452754
Color: 6.661517546139406e-05
drive_easy: 5.508718810632465e-05
Make: 3.703146059241238e-05
```

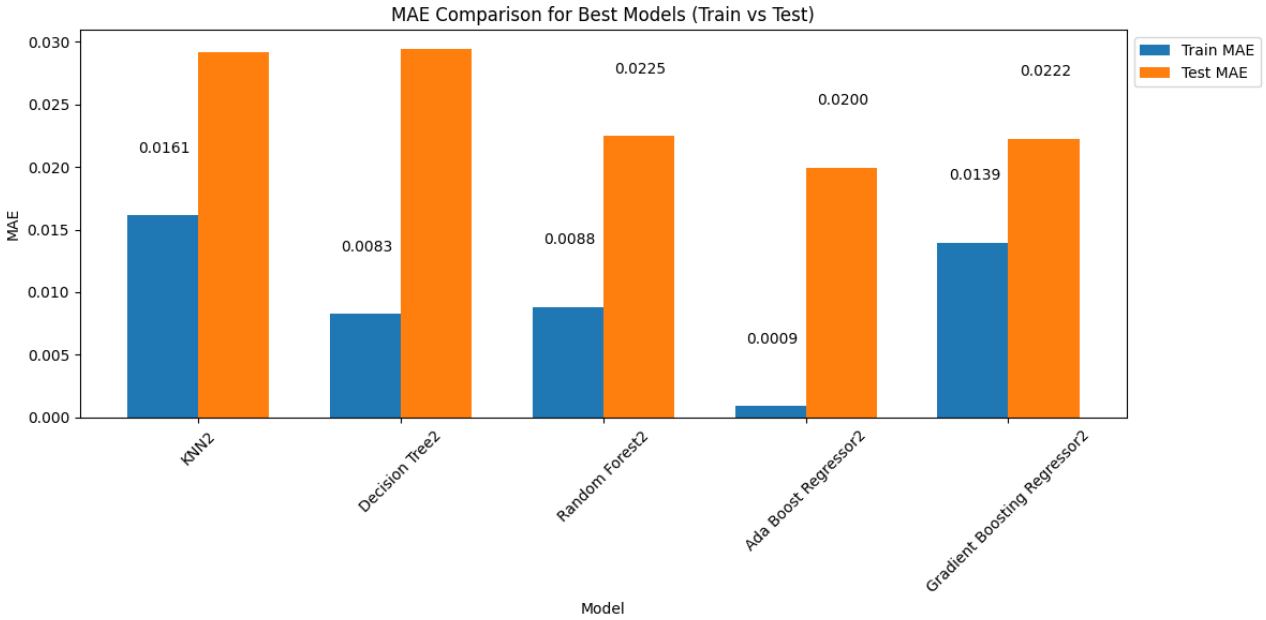
```
Importanza delle feature per DecisionTreeRegressor
sport_cars : 0.4457915327702762
Torque(Nm) : 0.16784498067200787
Year_Category : 0.12257982975583726
kilometers_run : 0.08844352020128812
Fuel_Tank_Capacity(L) : 0.07627916957041161
CC_Displacement : 0.014789149051234601
Body_Type : 0.014735287399775636
Model : 0.013291439127152596
Transmission : 0.013179288587155015
Color : 0.011065314119525799
euro : 0.006875474046307225
drive_easy : 0.005731809126902842
Fuel_Type : 0.004065573966979272
travel : 0.003394095058775965
consumption : 0.0027766474825024765
Make : 0.002715825903285355
Transmission_Type : 0.0024018034052201245
No_of_Owners : 0.0023775485995430772
Seating_Capacity : 0.001661711155818922
```

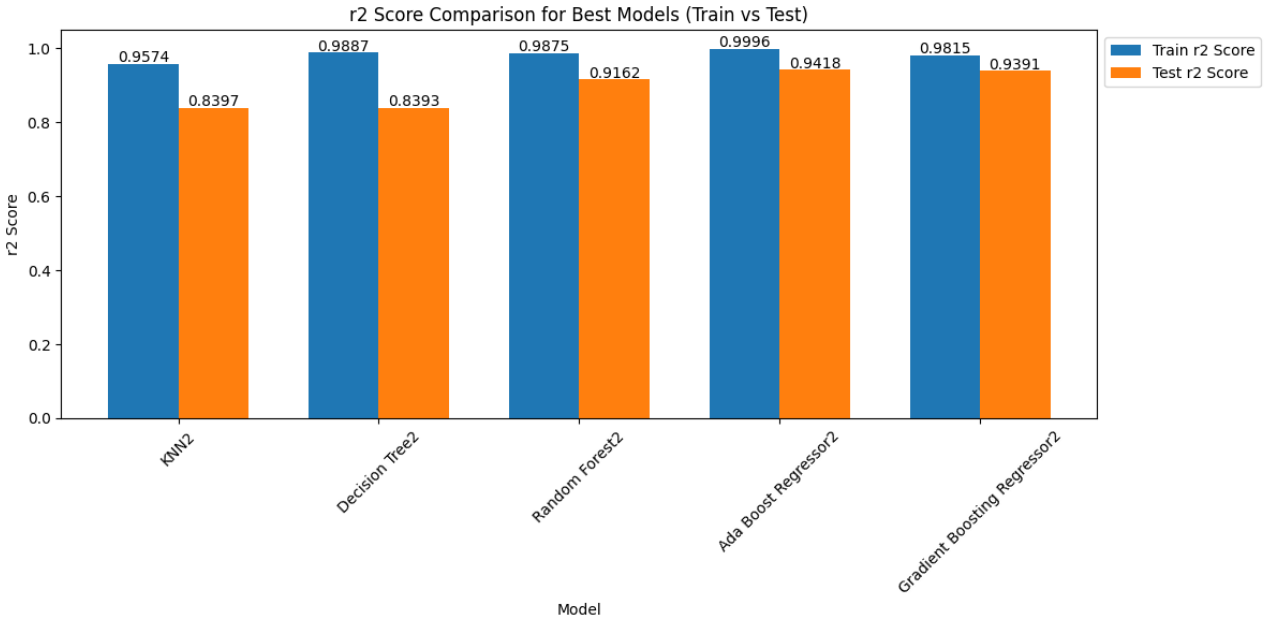
```
Importanza delle feature per RandomForestRegressor
sport_cars : 0.37348058166635195
Torque(Nm) : 0.23933949336565763
Year_Category : 0.08461858205186454
Fuel_Tank_Capacity(L) : 0.08265782535948946
kilometers_run : 0.07378281376257818
Model : 0.028557690571058607
Make : 0.022226852533903076
Body_Type : 0.020886204555483828
CC_Displacement : 0.015766820040947798
euro : 0.01274388697300839
Transmission : 0.01206716259584227
Color : 0.010248727082145428
No_of_Owners : 0.007004845381856509
drive_easy : 0.006280348736530272
consumption : 0.0032335623616574953
Fuel_Type : 0.0029010007009100855
Transmission_Type : 0.0020795642504575 26
Seating_Capacity : 0.0014232331517010065
travel : 0.0007008048585560225
```

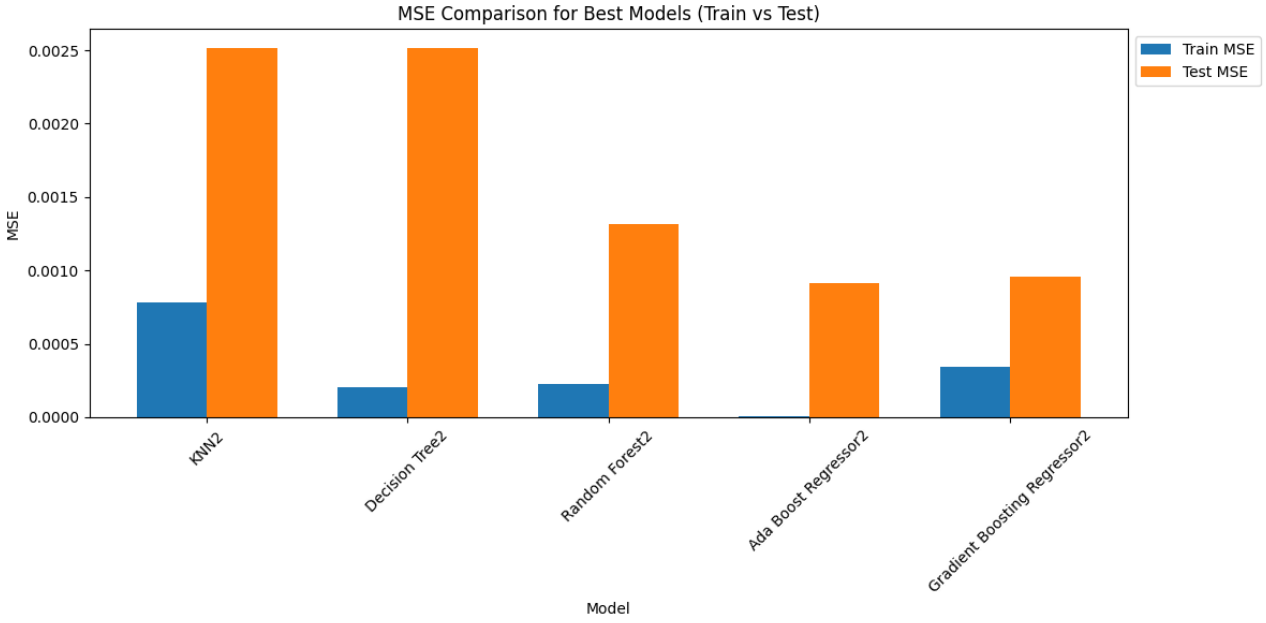
```
Importanza delle feature per AdaBoostRegressor
sport_cars : 0.1985565162348198
Torque(Nm) : 0.1592343211015069
Fuel_Tank_Capacity(L) : 0.15683059286118392
kilometers_run : 0.1319635773503909
Transmission : 0.0969699641468843
Year_Category : 0.06977054496019042
Body_Type : 0.053424785457321655
Model : 0.034644415355614 89
Make : 0.024017671079387 49
CC_Displacement : 0.022093020159437 55
Color : 0.014082440793093 41
drive_easy : 0.009485398941676076
euro : 0.008217915474491636
No_of_Owners : 0.005017610594639051
Transmission_Type : 0.004631272462384381
Fuel_Type : 0.003359589663869334
Seating_Capacity : 0.003076867565218537
travel : 0.002318861425890 8276
consumption : 0.002304634371998 74
```

```
Importanza delle feature per GradientBoostingRegressor(
sport_cars : 0.45041573572705224
Torque(Nm) : 0.16546672910703827
Year_Category : 0.10697993109795673
kilometers_run : 0.08467556883805492
Fuel_Tank_Capacity(L) : 0.07464538725856217
euro : 0.030188874000698175
Model : 0.019853400492592967
Body_Type : 0.014664352683621066
drive_easy : 0.012482785682407058
CC_Displacement : 0.011983107305520855
Fuel_Type : 0.007663203010288286
Color : 0.006511031182016739
No_of_Owners : 0.0035737339484655855
Transmission_Type : 0.002901245082865293
Transmission : 0.0026991527973115833
consumption : 0.0019893700284883994
Seating_Capacity : 0.0016686912538268872
travel : 0.0009825876628925395
Make : 0.0006551128403403582
```

Di seguito sono riportati punteggi di ogni regressore su set di train e set di test







Per il knn è stato registrato un R2 score di 0.95 sul set di training, mentre sul set di test il punteggio è di 0.83.
Per l'mse, invece, è stato registrato un punteggio di 0.0008 sul set di train e 0.0025 sul set di test.
Per il mae è stato registrato un punteggio di 0.0161 sul set di train e sul set di test 0.029.

Per il DecisionTree è stato registrato un punteggio r2 di 0.98 sul set di train e 0.83 sul set di test.
Invece per l'mse è 0.0002 sul set di train, mentre 0.0025 sul set di test.
Per la mae 0.0083 sul set di train e 0.030 sul set di test.

Per il RandomForest il punteggio r2 è di 0.98 sul set di train, mentre sul set di test è 0.91 .
Per l'mse 0.0002 sul set di train, mentre sul set di test è 0.0013.
Per la mae 0.0088 sul set di train, mentre sul set di test 0.022.

Per l'Adaboost è stato registrato un punteggio r2 di 0.99 sul set di train, mentre sul set di test è 0.94.

La mse è 0.0 sul set di train, mentre sul set di test 0.0008.
Per la mae è 0.0009 sul set di train, mentre 0.02 sul set di test.

Per il Gradient è stato raggiunto un punteggio di 0.98 per r2 sul set di train, mentre sul set di test 0.93.
Per la mse 0.0003 sul set di train, mentre sul set di test è stato raggiunto un punteggio 0.0009.
Per la mae 0.0013 sul set di train, mentre sul set di test è 0.022.I regressori DecisionTree e KNN soffrono di un leggero overfitting, mentre i restanti regressori AdaBoost, Gradient Boosting e Random Forest soffrono meno il problema dell'overfitting.

Prendendo in esame i migliore regressori, ovvero il Gradient Boosting e l'Adaboost, possiamo affermare che essi possiedono delle ottime capacità di predizione, senza soffrire particolarmente dell'overfitting come precedentemente indicato, riuscendo anche a catturare la maggior variabilità dei dati, garantendo cosi' delle previsioni ancora più accurate.

CSP (Constraint Satisfaction Problem)

Come già accennato, è stato utilizzato un CSP (Costraint Satisfaction Problem) per mostrare all'utente un insieme di macchine che rispettano le caratteristiche da lui introdotte in input.

All'avvio del programma verrà richiesto all'utente le caratteristiche di interesse quali: tipologia di macchina, macchina adatti per viaggi, tipologia del carburante ecc…

In base alle caratteristiche introdotte dall'utente verranno aggiunti vincoli e variabili.

Alcuni vincoli sono aggiunti a priori, ciò viene fatto per acquisire i dati più importanti, come ad esempio se la macchina di interesse all'utente è una macchina di tipologia sportiva o utilitaria, oppure la tipologia di carburante d'interesse, mentre altri vincoli sono aggiunti solamente se esplicitamente richiesti dall'utente.

Naturalmente il CSP sarà applicato all'intero dataset, per filtrarlo, così da mostrare all'utente solo le macchine che per lui sono rilevanti.

Per la risoluzione del CSP è stata adottata la libreria constraint.

Variabili aggiunte al CSP

Le variabili che possono essere aggiunte al problema CSP sono:

- Year_Category con dominio [0,1,2];
- Sport_cars con dominio [0,1];
- Consumption con dominio [0,1,2];
- Fuel_Type con dominio [1,2,3];
- Travel con dominio [0,1].

Per il significato dei valori salire nella parte relativa al preprocessing in apprendimento supervisionato (pagina 8 della documentazione).

Vincoli variabili

Sono stati definiti diversi vincoli per ogni variabile.

Per la variabile Make_Year sono stati definiti i vincoli oldyear, recentyear e newyear, i quali controllano rispettivamente se si tratta di una macchina rilasciata prima del 2014, uscita tra il 2014 e il 2019, o uscita dal 2019 in poi.

Per la variabile consumption è stato aggiunto l'omonimo vincolo , il quale verifica che si tratti di una macchina dai bassi consumi (ovvero che percorra almeno 20 kilometri al litro).

Per la variabile Sport_cars sono stati aggiunti i vincoli utilitaria e sportiva, per definire la tipologia di macchina.

Per la variabile Fuel_Type sono stati aggiunti i vincoli benzina, diesel e ibrida, i quali verificano appunto quale sia la tipologia di carburante utilizzata dalla macchina in questione.

Infine per la variabile travel è stato aggiunto l'omonimo vincolo, che controlla se è una macchina adatta a viaggiare o meno.

Risoluzione CSP

Per la risoluzione del CSP è stato utilizzato il metodo getsolution della classe Problem, presente all'interno della libreria constraint.

Tale metodo utilizza l'algoritmo del backtracking per la risoluzione.

Una volta risolto il csp viene mostrato il tempo impiegato per la risoluzione, il numero di vincoli e il numero di variabili aggiunte al problema csp.