Academia de Studii Economice
Facultatea de Cibernetică, Statistică și Informatică Economică

Proiect SDA

Predicția prețului unei case

Student:
Trâmbițaș Ioana,
Grupa 1118, Anul I

Cuprins

1. Prezentarea problemei propuse 2
2. Importanța problemei pentru firmă și la nivel personal (pentru membrii echipei) 2
3. Soluțiile încercate în alte cercetări 2
4. Soluția propusă (formalizarea problemei, colectarea datelor, modelare) 4
5. Rezultatele obținute (interpretare) 7
6. Concluzii, limitări și posibile îmbunătățiri 8
Bibliografie 10

1. Prezentarea problemei propuse

În general, atunci când oamenii doresc să cumpere o casă, ei caută ca această să fie accesibilă și să aibă toate caracteristicile dorite. Previziunile privind prețul casei îi vor ajuta să decidă dacă locuința pe care doresc să o cumpere merită sau nu acel preț.

Același lucru se întâmplă și în cazul persoanelor care doresc să vândă o casă. Prin utilizarea unui sistem de predicție a prețului casei, vânzătorul va putea decide dacă toate caracteristicile inerente proprietății și mediului înconjurător pot adăuga valoare vânzării, astfel încât casa să poată fi vândută la cel mai bun preț

În cele ce urmează ne vom ocupa de predicția prețului unei case în funcție de venitul mediu a persoanei care dorește să o achiziționeze, poziția casei (definite prin latitudine și longitudine), vechimea imobilului, numărul de camere și numărul de băi.

2. Importanța problemei pentru firmă și la nivel personal (pentru membrii echipei)

Pentru o firmă de imobiliare, ca pentru oricare alt tip, este important profitul. Având în vedere că de multe ori câștigul de pe urma vânzării unei proprietăți este un procent, cu cât imobilul se vinde la o sumă mai mare, cu atât câștigul este mai mare. Rezolvarea problemei de predicție a prețului unei case ajută angajații acestei tipuri de firme să negocieze pornind de la un preț mai mare decât cel prezis pentru a ajunge să vândă la prețul corect.

Pe plan personal, primul pas pentru cei interesați sa vândă o proprietate, este să pornească de la calcule și evaluarea zonei. Vânzătorul trebuie să înțeleagă că fiecare proprietate are o valoare de piață, mai mult decât o valoare emoțională.

Astfel, comparând prețurile de vânzare de pe piață își pot da seama dacă valoarea respectivei proprietăți este una realistă. Acest tip de evaluare se poate face printr-o cercetare detaliată cu ajutorul unor algoritmi de predicție. De asemenea, aflarea prețului corect a unei proprietăți este foarte importantă și dn poziția de cumpărători pentru a nu risca să plătim mai mult decât ceea ce valorează.

3. Soluțiile încercate în alte cercetări

Din cercetările făcute pentru a putea allege o metodă de predicție a prețului unui imobil, am observat că această problemă a fost tratată și folosind *regresia liniară multiplă,* adică o combinație între *regresia liniară simplă* și *regresia liniară multivariabilă*.

Regresia liniară este un model de învățare automată supervizată care încearcă să modeleze o relație liniară între variabilele dependente (Y) și variabilele independente (X). La fiecare observație evaluată cu un model, valoarea reală a obiectivului (Y) este comparată cu valoarea prezisă a obiectivului (Y), iar diferențele majore dintre aceste

valori se numesc reziduuri. Modelul de regresie liniară urmărește să minimizeze suma tuturor reziduurilor la pătrat. Iată reprezentarea matematică a regresiei liniare:

$$Y= a_0+a_1X+ \varepsilon$$

În ecuația de mai sus:
Y = Variabila dependentă
X = variabila independentă
a0 = Interceptarea dreptei care oferă DOF sau grad de libertate suplimentar.
a1 = Coeficientul de regresie liniară, care reprezintă un factor de scară pentru fiecare valoare de intrare.
ε = Eroare aleatorie

În problema studiată, ca prim pas a fost luată în considerare o companie de imobiliare cu seturi de date conținând prețurile proprietăților dintr-o anumită regiune.

Prețul unei proprietăți se bazează pe factori esențiali precum dormitoarele, baile, poziția.
În principal, o companie imobiliară necesită:

- Găsirea variabilei care afectează prețul unei case.
- Crearea unui model liniar legat cantitativ de prețul casei cu variabile precum suprafețele, numărul de camere și baie etc.
- Găsirea acurateței modelului, adică cât de bine pot prezice variabilele prețurile unei case.

A fost utilizată regresia simplă pentru a prezice cu ușurință prețul casei și regresia multivariabilă pentru a prezice rezultatele cu mai multă acuratețe folosind diferite variabile. A fost folosit un set de date complet care conținea informații exacte cu privire la case.

Ca prim pas au fost împărțite datele în date de antrenare și testare. Aceste grupuri reprezintă două subansamble create aleatoriu din datele noastre. Aceste date de testare/antrenare sunt utilizate pentru a adapta algoritmul de învățare, astfel încât să poată învăța cum să facă predicții. Setul de testare a fost folosit pentru a obține o idee despre funcționarea modelului cu date noi.

Procesul a continuat cu calcularea coeficienților și calcularea unui RMSE sau Root Mean Squared Error (Eroare pătratică medie pătratică), care este cel mai frecvent utilizat parametru pentru evaluarea modelului de regresie pe un set de testare.

Ca și concluzie, am observat că din pricina faptului că regresia liniară presupune o relație liniară între variabilele de intrare și cele de ieșire, nu reușește să se potrivească în mod corespunzător seturilor de date complexe. În cele mai multe scenarii din viața reală, relația dintre variabilele setului de date nu este liniară și, prin urmare, o linie dreaptă nu se potrivește corespunzător datelor. În astfel de situații, o funcție mai complexă poate surprinde datele mai eficient.

De asemenea, într-un set de date există și valori aberante, care reprezintă anomalii sau valori extreme care se abat de la celelalte puncte de date ale distribuției.Valorile aberante ale datelor pot afecta performanța drastică a unui model de învățare automată și pot duce adesea la modele cu performanțe scăzute de precizie, ceea ce reprezintă întocmai dezavantajul algoritmului de regresie liniară.

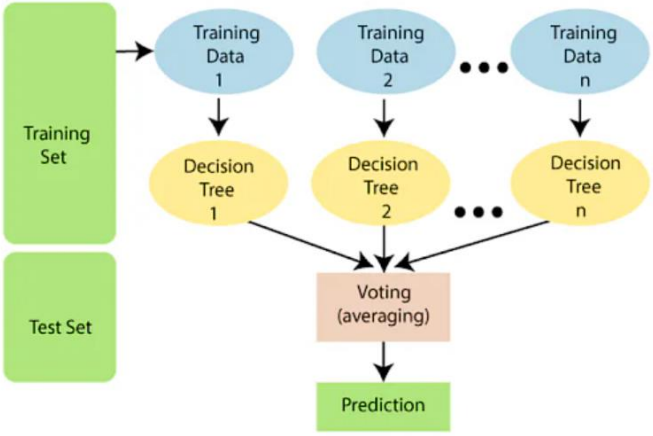
4. Soluția propusă (formalizarea problemei, colectarea datelor, modelare)

Pentru rezolvarea acestei probleme de predicție am decis să utilizez algoritmul *Random Forest,* iar în cele ce urmează voi explica ce anume este și motivul pentru care l-am ales. De asemenea, după multiple încercari de configurare și rulare a aplicației *R Studio,* am ales să folosesc limbajul de programare *Python*.

Algoritmul *Random Forest* este un algoritm de învățare automată supravegheată, extrem de popular și utilizat pentru probleme de clasificare și regresie în învățarea automată. Știm că o pădure cuprinde numeroși arbori, iar cu cât sunt mai mulți arbori, cu atât va fi mai robustă. În mod similar, cu cât numărul de arbori dintr-un algoritm Random Forest este mai mare, cu atât mai mare este acuratețea și capacitatea sa de rezolvare a problemelor. *Random forest* este un clasificator care conține mai mulți arbori de decizie pe diferite subansamble ale setului de date și calculează media pentru a îmbunătăți acuratețea predictivă a acelui set de date. Se bazează pe conceptul de învățare în ansamblu, care este un proces de combinare a mai multor clasificatori pentru a rezolva o problemă complexă și a îmbunătăți performanța modelului.

Am ales acest algoritm deoarece poate fi utilizat pentru a rezolva atât probleme de clasificare, cât și probleme de regresie. *Random forest* tinde să combine sute de *copaci de decizie* și apoi antrenează fiecare arbore de decizie pe un eșantion diferit de observații.

Funcționarea algoritmului este ilustrată mai jos:



Setul de date folosit în acest proiect este *California House* disponibil în librăria *sklearn,* motiv pentru care o vom importa. Caracteristicile acestui set sunt:

Numărul de atribute: 8
Numărul de instanțe: 20640
Informații despre atribute:
- *MedInc* venitul median în grupul de blocuri
- *HouseAge* vârsta mediană a locuinței în grupul de blocuri
- *AveRooms* numărul mediu de camere pe gospodărie
- *AveBedrms* numărul mediu de dormitoare pe gospodărie
- *Population* populația grupului de blocuri
- *AveOccup* numărul mediu de membri ai gospodăriei
- *Latitude* latitudinea grupului de blocuri latitudine
- *Longitudine* grup de blocuri Longitudine

Valori lipsă ale atributului: Niciuna
Acest set de date a fost obținut din depozitul StatLib.
https://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html

Variabila țintă este valoarea mediană a locuinței pentru districtele din California, exprimată în sute de mii de dolari (100.000 de dolari).

Acest set de date a fost derivat din recensământul american din 1990, utilizând un rând pentru fiecare grup de blocuri ale recensământ. Un grup de blocuri este cea mai mică unitate geografică și un grup de blocuri are, de obicei, o populație de 600 până la 3.000 de persoane.

O gospodărie este un grup de persoane care locuiesc într-o locuință. Deoarece media numărul mediu de camere și dormitoare din acest set de date este furnizat pentru fiecare gospodărie, aceste coloane pot lua valori exagerat de mari pentru grupurile de blocuri cu puține gospodării și multe case goale, cum ar fi stațiunile de vacanță.

După importarea librariilor necesare, am citit datele si am obținut caracteristicile din *.data* și etichetele din *.target*, urmând să împart datele în cele de testare și de instruire utilizând *train_test_split* cu o dimensiune de testare de 33%.

Următorul pas este cel de analiză a datelor prin creare unui *endpoint* GET pentru a obține statisticile de bază.

Mai întâi concatenăm caracteristicile și etichetele și apoi le combinăm sub formă de coloane cu nume de coloane specifice.

```
[7]
setdate = pd.concat([pd.DataFrame(daterecensamant.data, columns = daterecensamant.feature_names),
                     pd.DataFrame(daterecensamant.target*100000, columns = ['Price'])], axis = 1)
```

Datele au fost analizate prin intermediul funcției *info().*

```
[23]
setdate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   MedInc      20640 non-null  float64
 1   HouseAge    20640 non-null  float64
 2   AveRooms    20640 non-null  float64
 3   AveBedrms   20640 non-null  float64
 4   Population  20640 non-null  float64
 5   AveOccup    20640 non-null  float64
 6   Latitude    20640 non-null  float64
 7   Longitude   20640 non-null  float64
 8   Price       20640 non-null  float64
dtypes: float64(9)
memory usage: 1.4 MB
```

Se poate observa că în setul de date există un total de 20640 de case, 8 caracteristici și o coloană de etichete. Nu există valori nule.

Procesul a continuat prin calcularea corelației.

```
[25]  0.2s
corelatie = dataset.corr()
corelatie.style.background_gradient(cmap='viridis', low=.5, high=0).highlight_null('red')
```

Pasul următor a consistat în punerea în practică a algoritmului. Am importat *RandomForestRegressor* din *sklearn.ensemble* și *mean_squared_error* din *sklearn.metrics.* Am continuat prin selectarea unor eșantioane aleatorii din setul de date, construirea unui arbore de decizie pentru fiecare dată de instruire și calcularea mediei arborelui de decizie. Variabila *n_estimators* reprezintă numărul de arbori de decizie.

```
[19]
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error

clf = RandomForestRegressor(n_estimators = 100, max_depth = 50)
clf.fit(X_train, y_train)
print("Eroarea Medie Absoluta: {}".format(mean_squared_error(y_test, clf.predict(X_test))))
```

```
Eroarea Medie Absoluta: 0.26296539075811187
```

```
[12]  ▷ 13.0s
endpoint_classifier = RandomForestRegressor(n_estimators = 100, max_depth = 50)
endpoint_classifier.fit(daterecensamant.data, daterecensamant.target)
```

Pe baza individuării arborelui de decizie mediu, am atribuit valorile acestuia la atributele care îl constituie și am individuat prețul mediu a imobilului, pe baza caracteristicilor enumerate. De asemenea, am creat un obiect *REQUEST* aleatoriu pentru *endpointul POST* cu valorile medii din setul nostru de date.

```
[13]
features = pd.DataFrame(daterecensamant.data)
mean_values = features.describe().iloc[1, :]

REQUEST = json.dumps({
    'body': {
        'MedInc': mean_values[0],
        'HouseAge': mean_values[1],
        'AveRooms': mean_values[2],
        'AveBedrms': mean_values[3],
        'Population': mean_values[4],
        'AveOccup': mean_values[5],
        'Latitude': mean_values[6],
        'Longitude': mean_values[7]
    }
})
```

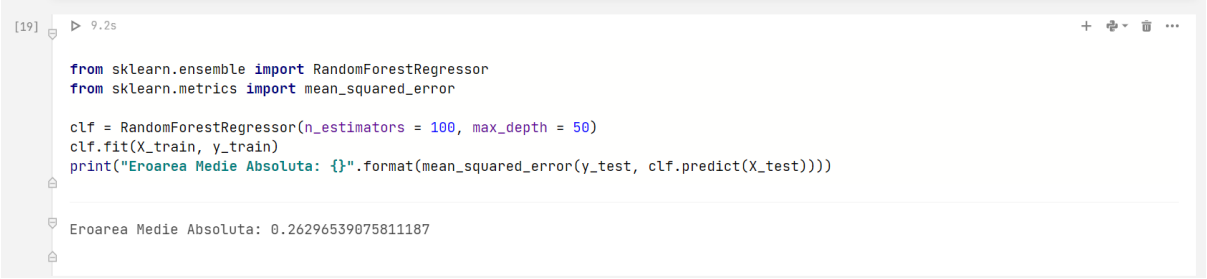
Prin intermediul obiectului *json* importat, am trimis cererea de *POST,* care va accepta toate valorile de la utilizator și va returna prețul estimat. Datele primite se află în partea de *body* a cererii. Acest rezultat a fost apoi afișat.

```
[20]
# POST /get_price
req = json.loads(REQUEST)
req = np.array(list(req['body'].values()))
predicted_price = endpoint_classifier.predict(req.reshape(1, -1))[0]
predicted_price = "{0:.2f}".format(predicted_price*100000)
print(json.dumps({
    'Rezultat': 'Pretul casei care are specificatiile alese este de: $' + predicted_price
}))
```

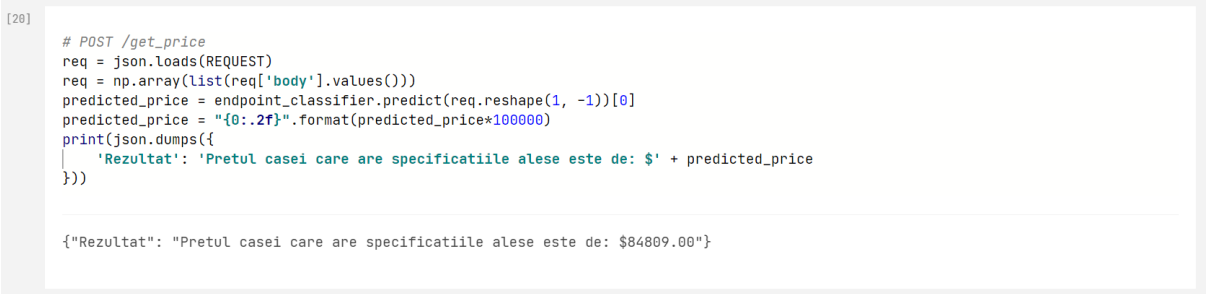
5. Rezultatele obținute (interpretare)

În ceea ce privește rezultatele obinute, se poate observa că în calculul corelației prețul depinde în principal de venitul mediu, cu o corelație de aproximativ ~0,7.

Eroarea medie obținută a fost de ~0,263, după cum se poate observa în imaginea de mai jos.



Se poate oberva că prețul mediu estimat pe baza datelor de intrare este de $84.809.00.



6. Concluzii, limitări și posibile îmbunătățiri

După punerea în aplicare și studierea acestui algoritm am individuat câteva avantaje, dar și dezavantaje a acestuia.

Principalul lucru de care trebuie să ținem cont este că performanța oricărui model este direct proporțională cu cantitatea de date valide din care poate învăța, iar în acest proiect a fost folosită o cantitate foarte limitată de informații pentru instruire. Pentru un

studiu de piață propriu-zis este necesară utilizarea unui set de date de dimensiuni mai mari.

Printre avantajele acestui algoritm se numără faptul că efectuează implicit selecția caracteristicilor și generează arbori de decizie necorelați. Acest lucru se realizează prin alegerea unui set aleatoriu de caracteristici pentru a construi fiecare arbore de decizie. De asemenea, un alt avantaj este faptul că produce predicții bune.

O limitare majoră este faptul că algoritmul nu a fost testat în cazul unui set de date cu multiple valori nule.

Ca și dezavantaje putem individua faptul că folosește multe resurse ale calculatorului și timpul de rulare a codului este destul de mare. De asemenea, devine mai greu de înțeles pe măsură ce creștem numărul de arbori în care împărțim setul de date. De asemenea, ca și minus am individuat faptul ca in Python algoritmul e un fel de *black box* deoarece după importarea librăriilor, utilizatorul are foarte puțin control asupra a ceea ce face modelul.

Ca și îmbunătățire aș propune aplicarea acestui algoritm pe un set mult mai mare de date pentru a îi observa comportamentul.

Bibliografie

[1] https://linuxhint.com/house-price-prediction-linear-regression/
[2] https://www.simplilearn.com/tutorials/machine-learning-tutorial/random-forest-algorithm
[3] https://medium.datadriveninvestor.com/random-forest-pros-and-cons-c1c42fb64f04
[4] https://medium.com/analytics-vidhya/solving-your-first-regression-program-house-price-prediction-bc31a0fe9e7b
[5] https://datalore.jetbrains.com/notebook/