Szegedi Tudományegyetem

Informatikai Intézet

SZAKDOLGOZAT

Bobák Vivien

2022

Szegedi Tudományegyetem
Informatikai Intézet

Adatbányászati és gépi tanulási módszerek alkalmazása ingatlanpiaci adatokra

Szakdolgozat

Készítette:
Bobák Vivien
gazdaságinformatika szakos hallgató

Témavezető:
Dr. London András István
egyetemi adjunktus

Szeged
2022

Feladatkiírás

A feladat célja az OtthonCentrumtól kapott 2 adatbázis feldolgozása, statisztikai elemzés, a kapott adatsorok korrelációjának a vizsgálata. Illetve a gépi tanulás, azon belül a lineáris regresszióval való kiadási ár és az eladási ár előrejelzés megvalósítása különböző ingatlan típusokra, minél pontosabban.

Tartalmi összefoglaló

- **A téma megnevezése:**

  Statisztikai és korrelációs elemzések elvégzése az ingatlanpiaci adatokról, előrejelzés a kiadási és az eladási ár értékéről

- **A megadott feladat megfogalmazása:**

  A feladat a rendelkezésre álló ingatlanpiaci adatsorok összefüggéseinek és korrelációjának elemzése, illetve egy modell elkészítése, amellyel olyan előrejelzés készíthető, ami segít megállapítani az adott ingatlan kiadási vagy eladási árát.

- **A megoldási mód:**

  Először az adatokat kellett megvizsgálni és utána a szükséges adattípusokat encodolni. Encodolás után a korrelációs vizsgálat, illetve gépi tanulási módszerek alkalmazása volt a következő feladat. Ezeket Python nyelven valósítottam meg, a függvénykönytárait felhasználva a Google Colab fejlesztő környezetben.

- **Alkalmazott eszközök, módszerek:**

  Google Colab fejlesztőkörnyzet, Python nyelven, Pandas könyvtár, Sklearn könyvtár, Pydot könyvtár, Graphviz könyvtár, Matplotlib könyvtár, Seaborn könyvtár, 2db adatbázis az OtthonCentrumtól

- **Elért eredmények:**

  Az adatok elemzése különböző módszerekkel, korrelációs vizsgálat különböző esetekben, ezek összevetése, minél pontosabb előrejelzés különböző encodolások esetén, lineáris regresszió és döntési fa összehasonlítása. A vizsgált teszthalmazokon az elért legmagasabb előrejelzési érték 94%-os pontosság volt.

- **Kulcsszavak:**

  Python, Adatbányászat, Korreláció, Ingatlanpiac, Gépi tanulás, Label encoding, One hot encoding

Tartalomjegyzék

Bevezetés

A szakdolgozatom téma ötletének alapját az egyetemi tanulmányaim során megismert mesterséges intelligencia adta, nagyon tetszett ez a tárgy és egy ezzel kapcsolatos témát szerettem volna választani. Így választottam ki, hogy gépi tanulásos feladatot szeretnék csinálni, ugyanakkor érdekelt az adatbányászat is, emiatt esett választásom a korreláció elemzésre, a lineáris regressziós és a regressziós döntési fa modellekre is. Gépi tanulás tárgyat ugyan én nem vettem fel a tanulmányaim során, de az egyetem oldalán fent volt a tananyag és onnan megtanultam és elsajátítottam.

Miután megvolt az irány, hogy mit szeretnék csinálni kérdéses volt még, hogy melyik nyelven valósítsam ezeket meg, illetve, hogy milyen környezetben. Ezekhez én a Python programozási nyelvet választottam, mert tökéletesen megfelelt az általam választott témák megvalósításához és mert egyetemi tanulmányaim során többször is találkoztam vele (Szkriptnyelvek, Python a gyakorlatban) és már akkor is nagyon tetszett. A szakdolgozatom során elég sok nagy méretű külső könyvtárat kellett felhasználom és ezek miatt döntöttem a Google Colab futtató környezet mellett, mert itt mindent nagyon egyszerűen tudtam importálni anélkül, hogy bármit le kellett volna töltenem és ezzel rengeteg tárhelyet spóroltam meg.

A dolgozatomhoz szükséges adatbázisok témáját az alapján választottam, hogy az életben milyen területek érdekelnek, több lehetőség közül végül az ingatlanpiac mellett döntöttem. Szerintem rengeteg kapcsolat van az adatok között, illetve a bérbeadók/eladók döntéshozatala is szerintem felettébb érdekes, hogy mi alapján hozzák meg a döntéseket, hogy mennyire értékelik az ingatlanukat. Érdekelt továbbá, hogy ezek között lehet e logikai kapcsolatokat bizonyítani, illetve, hogy gépi tanulással előre lehet-e jelezni egy ingatlan bérleti díját, vagy eladási értékét.

A két adatbázis, amivel végül dolgoztam az OtthonCentrum jóvoltából a szakdolgozatomhoz való felhasználásra kaptam. Az oldaluk alapján határoztam meg, hogy mely tulajdonságok lehetnek fontosabbak egy-egy bérlőnek, vásárlónak és ezeket felhasználva vontam le következtetéseket. Egy ingatlan értékét mindegy, hogy bérbe adják, vagy eladják, egyaránt fontos, hogy jól tudja a tulajdonos belőni az ingatlanpiaci értékét. Hiszen, ha túl magasra becsüli az értékét akkor lehet nem lesz bérlője, vevője, ha túl alacsonyra értékeli, akkor meg veszteséges üzletet fog kötni. A gépi előrejelzés éppen ezek miatt hasznos lehet itt, hiszen az ingatlan valós értékét fogja tudni megmondani minimális pontatlansággal.

1. Python külső könyvtárak

1.1 Pandas

A neve a Panel Data-ból származik. 2008-ban kezdte el a fejlesztését Wes McKinney nevű fejlesztő. A Pandas egy nyílt forráskódú Python könyvtár, amely lehetővé teszi, hogy a Python programozási nyelvet ne csak adatelőkészítésre tudjuk használni, hanem a teljes adatelemzési folyamatokat egy nyelven tudjuk megírni. Így nem szükséges más nyelvekre áttérni (pl R), elegendő importálni és megfelelően használni. A Pandas könyvtár segítségével 5 főbb dolgot tudunk megvalósítani: betölteni, előkészíteni, módosítani, modellezni és elemezni. A Pandas számomra leghasznosabb része a gyors és hatékony DataFramek létrehozása és használata volt. A Pandas segítségével lehetőségünk nyílik a külső adatbázisainkat (pl csv, excel, sql, json kiterjesztésű) DataFrame-é alakítani, így könnyedén tudjuk őket elemezni, dolgozni velük, módosítani őket. A munka befejeztével természetesen ki is tudjuk őket íratni a fentebb példaként hozott formátumokba, de azokon kívül is még sok lehetőségünk van. Ezeken kívül megkönnyíti még a hiányzó adatok kezelését is, ha például egy nagyobb adatbázisból csak bizonyos oszlopok vagy sorok kellenek, mivel lehetőség van vele címke alapú szeletelésre, indexelésre, ez is könnyen kivitelezhető. Lehetőség van még oszlopok beszúrására vagy törlésére, vagy esetleg két adatbázis megfelelő összefésülésére. Ezek alapján számomra tökéletesen megfelelt a szakdolgozatomhoz, ezért esett erre a választásom. Kapcsolódó irodalomjegyzék [1]

1.2 Sklearn

Teljes neve Scikit-Learn. Ennek a könyvtárnak az alapja a NumPy, a SciPy és a Matplotlib. Eredetileg a Google egy nyári projektjének indult 2007-ben, amit David Cournapeau fejlesztett ki. Később 4 fejlesztő, név szerint Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort és Vincent Michel, a FIRCA-tól, fejlesztették tovább és 2010 elején lett elérhető az első béta verziója. A fejlesztésében bárki segíthet, mivel ez egy nyílt forráskódú könyvtár. A szakdolgozatom során elengedhetetlen volt ennek a könyvtárnak a használata, hiszen ennek segítségével tudtam megfelelően modellezni és jól működő tanuló, illetve kiértékelő adatbázisokat létrehozni és felhasználni. Én felügyelt tanulási algoritmusok miatt használtam, mert regressziós feladatot és döntési fákat valósítottam meg, de ezen kívül lehet még felügyelet nélküli tanuló algoritmusokhoz, fürtszolgáltatáshoz, dimenziócsökkentéshez és még rengeteg más feladathoz is alkalmazni. Kapcsolódó irodalomjegyzék [2]

1.3 Graphviz

A Graphviz egy nyílt forráskódú Python modul, amely gráfobjektumok létrehozására szolgál. A Graphviz szoftver DOT nyelvén alapul. Ennek segítségével különböző csomópontokat és éleket tudunk vizuálisan megjeleníteni a képernyőn. Én a döntési fák megjelenítéséhez használtam, hogy átláthatóbb, könnyebben értelmezhetőbb legyen. Kapcsolódó irodalomjegyzék [3]

1.4 Matplotlib és Pyplot

A Matplotlib a Python programozási nyelv és a NumPy ábrázoló könyvtára. A Pyplot egy Matplotlib modul, amely MATLAB szerű interfészt biztosít. A Matplotlib-et úgy tervezték, hogy ugyanolyan használható legyen, mint a MATLAB, csak Python nyelven. Ez is nyílt forráskódú. Erre a könyvtárra és a Pyplot modulra a szakdolgozatom során szükségem volt, hogy a később beszúrt diagrammokat meg tudjam valósítani, illetve, hogy a korrelációkat is szebben, átláthatóbban tudjam megtekinteni heatmapek segítségével. Kapcsolódó irodalomjegyzék [4] [5]

1.5 Seaborn

A Seaborn Python egy adatvizualizációs könyvtára, amely matplotlib alapú. Magas szintű felületet biztosít hatékony és informatív statisztikai grafikák rajzolásához. A Seaborn szorosan kapcsolódik a Pandas könyvtárhoz, így célravezető volt ezt a könyvtárat is használnom, amikor a gépi tanuláshoz akartam megvizsgálni az egyik sima label encodingos adatbázisomat. Ekkor az adatbázis minden int vagy float típusú adatcsoportot összevetette az összestöbbivel. Itt nem kellett kiválasztani, hogy milyen diagramm típust szeretnék alkalmazni, mert a rendszer az értékeknek megfelelően automatikusan legenerálja az arra leginkább illőt. Ehhez a pairplot-ot használtam. A Seaborn könytárból használtam még a realplot-ot is, amely képes több adatcsoport egyidejű összevetésére egymással. Kapcsolódó irodalomjegyzék [6]

2. Adatbázisok kialakítása

2.1 Beolvasás, adatok formálása

A szakdolgozatomnál használt két darab adatbázist az OtthonCentrum biztosította. Mivel én a Google Colab fejleztői környezetben írtam meg minden kódot, ezért első lépésben a Google Drive-omhoz kellett csatlakoznom. Ezután a Pandas könyvtár read függvényét használva és a megfelelő abszolút elérési útvonalat és a szeparáló jelet megadva beolvastam mindkét csv adatbázisomat. Ezek után DataFrame-ekké kellett alakítanom, hogy használni tudjam őket. Ehhez a Pandas könyvtár DataFrame függvényét használva, meg kellett adnom a változóba beolvasott adatbázis nevét, majd fel kellett sorolnom az összes oszlop nevét, ahogy a csv-ben megtalálhatók voltak. Majd a fillna függvény segítségével az összes olyan mezőt, ami üresen volt hagyva feltöltöttem 0-val, hogy a későbbiekben ne legyen ebből probléma. Így kaptam két darab adatbázist, egyet a bérelt ingatlanok adataiból, ez lett a df_bereltossz, ez egy 200soros, 25 oszlopos adatbázis, és egyet az eladott ingatlanok adataiból, ez lett a df_eladottossz, ez egy 500 soros, 25 oszlopos adatbázis.

2.2 Adatok vizsgálata

Először meg kellett vizsgálnom, hogy a csvből milyen típusként olvasta be őket. Az alap adatbázisok, amiket kaptam nem voltak így megfelelők, sok float típusú adatot dátumként kezelt, illetve volt, ahol az int vagy float típusúakat meg object-ként, mintha szövegek lettek volna, ezért a csv-ken módosítanom kellett. Miután mindent kijavítottam, már megfelelően olvasta be. Az adatbázisban szereplő adatok típusait a dtypes függvény segítségével könnyedén le tudtam kérni és ellenőrizni. Ekkor már a tényleges típusaikat láttam és kiválasztottam hét object típust, amiket fel szerettem volna használni a gépi tanulásom során. Ezeket a típusokat az alapján választottam ki, hogy mik lehetnek az elsődleges szempontok, a lefontosabb tulajdonságok, amik alapján az ingatlan közvetítő oldalakon rá lehet szűrni a keresésekre. Viszont object formátumban még így nem tudtam se a lineáris regresszióhoz, se a döntési fákhoz felhasználni őket, ezért encodingra volt szükségem, hogy szám típusúvá tudjam őket váltani, és tudjam őket használni. Kapcsolódó irodalomjegyzék [15]

2.3 Másolatok

Adatbázisok kezelésnél fontos és elhanyagolhatatlan lépés volt nálam, hogy másolatokat hozzak létre belőlük, különben az eredeti adatbázison változtatott volna. Mivel én több mindent is szerettem volna csinálni velük, emiatt szükséges volt annyi másolatot is létrehozni, amit egyszerűen megtehettem a copy függvény segítségével. Így lett hat darab adatbázisom, három a bérbeadott ingatlanokról, 3 az eladott ingatlanokról. Erre azért volt szükség, mert csinálni akartam egy label encodingot, aminél nincs szűrés, egy szintén label encodingot, de ott már csak minden kategóriából, legalábbis ahol volt annyi, ott az első öt legjelentősebbet kilistázva és egy onehot encodingot, ahol szintén minden kategóriából az első ötre szűrök rá. Ezeket lineáris regressziónál és a korrelációnál is külön-külön vizsgáltam meg.

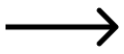
3. Encoding

A legtöbb adatbázis tartalmaz(hat) szöveges adattípusokat, viszont, ha ezek közreműködésével szeretnénk egy előrejelzést megalkotni, ilyen formában nem tudjuk megtenni, mert a legtöbb gépi tanulásos algoritmus numerikus adatokkal működik csak. Ehhez szükséges az encoding, ami lényegében a szöveges, kategorikus változókból numerikus típusút csinál. Ennek több módja is van, de én a szakdolgozatom során a két legjelentősebbel foglalkoztam, a label és a onehot encodinggal. Ezek mindketten a SciKit Python könyvtárba tartoznak, és más-más módon, de numerikus adatokká alakítják át a szöveges, kategorikus adatokat, ezzel segítve, hogy ezeket az adatokat is fel lehessen használni a gépi tanulásos előrejelzés során.

3.1 Label encoding

A label encoding-gal úgy valósítja meg az encodingot az algoritmus, hogy egy adott oszlopot tekintve, minden különböző elemhez más szám értéket rendel hozzá. Így átlátható lesz később, hogy az azonos típusoknál ugyanaz a szám található meg, mégis numerikus értékkel tudunk dolgozni. Ezeket a számokat nem véletlenszerűen osztja ki, 0tól indul és ahhoz ad hozzá mindig egyet, ha talál egy új címkét.

	name	gender	age	city
a	Abby	F	33	Berlin
b	Ben	M	16	Tokyo
c	Charlie	M	22	Sydney
d	Dave	M	65	York
e	Ella	F	18	Sydney



	name	gender	age	city
a	Abby	0	33	0
b	Ben	1	16	2
c	Charlie	1	22	1
d	Dave	1	65	3
e	Ella	0	18	1

1. ábra, 3.1-es fejezet: label encoding bemutatása, Kapcsolódó irodalomjegyzék [8]

Ahhoz, hogy ezt el tudjuk végezni, minden ilyen oszlopnak kategorikus típusúnak kell lennie. Viszont a Python nyelvben a DataFrameknél, ha valami nem numerikus típusú, rendszerint object típusra állítja be automatikusan, de ezt könnyen módosítani lehet az astype beépített függvénnyel. Ehhez ki kell jelölnünk egy oszlopot, amire vonatkozni fog, majd

megadni az astype értékéül, hogy milyen típusra szeretnénk módosítani. Nekem a category típusra volt szükségem.

```
df_masolat_eladott[„Település”] = df_masolat_eladott[„Település”].astype(’category’)
```

Ezek után létrehozunk egy új oszlopot, ehhez meg kell adni, hogy milyen nevet szeretnénk neki, majd a korábban kategorikus típusúvá változtatott oszlop alapján feltölti az új oszlopot a numerikus értékekkel és letárolja őket.

```
df_masolat_eladott[„Település_kategória”] = df_masolat_eladott[„Település”].cat.codes
```

Ennél az egyszerű label encodingnál az ehhez létrehozott két darab másolat adatbázisaimat használtam fel, a df_masolat_berelt, illetve ami a példában is szerepel, a df_masolat_eladott. Az encoding miatt a 25 oszlopomból 32 darab lett, ami azért van, mert a korábban az Adatok vizsgálata fejezetben említett 7 oszlop, amit szintén szeretnék felhasználni a vizsgálatokhoz, mindegyikhez le lett képezve egy új oszlop. Kapcsolódó irodalomjegyzék [7]

3.1.1 Top5-ös label encoding

Itt a hat darab másolat adatbázisból másik kettőt fogok felhasználni, ezek pedig a df_masolat_labeltop5_berelt és a df_masolat_labeltop5_eladott. Ezeknél a hét változónál, amit encodolni kellet, nem minden típusát vizsgálom, hanem mindenhol kategóriánként az első ötöt, ami a legtöbbször szerepel. Ezt úgy kiviteleztem, hogy először mindent ugyanúgy tettem, mint az egyszerű, sima label encodingos verzióknál, kategórikus típussá alakítottam majd beszúrtam az új oszlopokat. Utána az eredeti 7 oszlopot, egyesével lekértem, hogy melyik változójuk hányszor található meg. Ezt a value_counts() beépített függvény segítségével tudtam megtenni, mert ez az adott oszlop változóit összegezve adja vissza. Ahol ötnél kevesebb változó volt, például a Jogi_státusz oszlopban csak használt és új volt, ott azzal csináltam tovább, ahol meg jóval több volt, mint öt, ott pedig csak az első öttel foglalkoztam.

```
df_masolat_labeltop5_berelt[„Település”].value_counts()
```

Miután ezekkel végeztem létrehoztam két új DataFramet, név szerint df_masolat_labeltop5_KESZ_berelt és df_masolat_labeltop5_KESZ_eladott adatbázisokat. Ezekbe mentettem ki az encodolt adatbázisokat arra 7 bizonyos oszlopra vonatkozó szűrési feltételekkel. Így a bérelt ingatlanokhoz tartozó adatbázis ennél 30 sorból és 32 oszlopból állt, míg az eladott ingatlanoké 100sorból és 32 oszlopból.

```
df_masolat_labeltop5_KESZ_eladott = df_masolat_labeltop5_eladott[((df_masolat_labeltop
5_eladott.Település == „Debrecen”) | (df_masolat_labeltop5_eladott.Település == „Miskolc
”) | (df_masolat_labeltop5_eladott.Település == „Pécs”) | (df_masolat_labeltop5_eladott.Tel
```

epülés == „Nyíregyháza”) | (df_masolat_labeltop5_eladott.Település == „Tatabánya”)) & ((df_masolat_labeltop5_eladott.Emelet == „0”) | (df_masolat_labeltop5_eladott.Emelet == „3”) | (df_masolat_labeltop5_eladott.Emelet == „2”) | (df_masolat_labeltop5_eladott.Emelet == „1”) | (df_masolat_labeltop5_eladott.Emelet == „4”)) & ((df_masolat_labeltop5_eladott.Fűtés == „gaz_cirko”) | (df_masolat_labeltop5_eladott.Fűtés == „tavegyedi”) | (df_masolat_labeltop5_eladott.Fűtés == „gaz_konvektor”) | (df_masolat_labeltop5_eladott.Fűtés == „tav”) | (df_masolat_labeltop5_eladott.Fűtés == „hazkozponti”)) & ((df_masolat_labeltop5_eladott.Szobaszám == „2 + 0”) | (df_masolat_labeltop5_eladott.Szobaszám == „3 + 0”) | (df_masolat_labeltop5_eladott.Szobaszám == „1 + 0”) | (df_masolat_labeltop5_eladott.Szobaszám == „2 + 1”) | (df_masolat_labeltop5_eladott.Szobaszám == „1 + 1”))]

3.2 One hot encoding

Ez a másik közkedvelt encodoló módszer. Míg a label encoding-nál fennáll a veszélye, hogy az algoritmus félreértelmezi a numerikus értékeket és azt hiheti, hogy azok között valami féle hierarchia fedezhető fel, ezzel ellentétben a one hot encodingnál nincs ilyen probléma. Ez a módszer úgy valósítja meg az encodolást, hogy az adott oszlophoz, amit encodolni szeretnénk, minden címke típusához létrehoz egy-egy új oszlopot, és aszerint tölti fel elemekkel, hogy az adott sor hozzátartozik-e vagy sem. Ha hozzátartozik akkor abban az egy oszlopban és sorban egy 1-es fog szerepelni az igaz értéket jelölve, míg a többi címke ezen sorában egy 0 a hamis értéket jelölve.

	name	gender	age	city	city_Berlin	city_Sydney	city_Tokyo	city_York
a	Abby	0	33	Berlin	1	0	0	0
b	Ben	1	16	Tokyo	0	0	1	0
c	Charlie	1	22	Sydney	0	1	0	0
d	Dave	1	65	York	0	0	0	1
e	Ella	0	18	Sydney	0	1	0	0

2. ábra, 3.2-es fejezet: one hot encoding bemutatása, Kapcsolódó irodalomjegyzék [8]

Ennek a módszernek a hátránya sajnos viszont az, hogy csak kevés egyedi értékkel rendelkező kategória oszlopokra érdemes alkalmazni, különben ahány egyedi értékünk van, annyival több oszlopunk is lesz. Ez néhány algoritmusnál gondot okozhat, nehezen kezelhetővé és átláthatatlanná változik tőle az adatbázis. Kapcsolódó irodalomjegyzék [9]

3.2.1 Top5-ös one hot encoding

Eleinte úgy terveztem, hogy mint ahogy a label encodingnál is van, csinálok egy egyszerű one hot encodolt adatbázist, illetve egy olyat, ahol a bizonyos 7 oszlopból csak az első ötöt veszem figyelembe. De a föntebb leírt problémák miatt nem szerettem volna egy közel 120 oszlopos adatbázissal dolgozni, emiatt csak a top5-ös változatát valósítottam meg. Itt a másolatoknál létrehozott maradék két adatbázissal dolgoztam, név szerint df_masolat_onehottop5_berelt és df_masolat_onehottop5_eladott. Először egy változóban megadtam a 7 darab oszlopnevemet, majd egy for ciklussal végig mentem rajtuk. A for ciklusban a get_dummies() függvény segítségével létrehozom és beszúrom az adatbázisba az új oszlopokat, majd ha minden érték megvolt, kilép a ciklusból és eldobja a drop() függvény segítségével az adott eredeti oszlopot, ezzel csökkentve a redundanciát. Miután ezzel megvoltam, mind a 7 oszlopomra elkészült a one hot encoding, de ugyanakkor rengeteg felesleges oszlopot tartalmaz még, így két új DataFrambe kimentettem azokat az oszlopokat amikkel dolgozni szerettem volna, ezeket egyesével felsoroltam. Ennek a két változónak a neve df_masolat_onehottop5_berelt_KESZ és df_masolat_onehottop5_eladott_KESZ. Majd a WHERE segítségével kiszűrtem azokat a sorokat, amik megfelelnek annak, hogy mind a 7 encodolt tulajdonság változó értéke a top5ben szerepeljen. Ezután kimentettem őket 1-1 új változóba a loc() beépített függvény segítségével, a változók nevei: df_masolat_onehottop5_berelt_VEGLEGES (64 sor, 49 oszlop) és df_masolat_onehottop5_eladott_VEGLEGES (130 sor, 50 oszlop). Itt több úton módon is megoldhattam volna, például nem azokkal az oszlopokat sorolom fel, amiket használni szeretnék, hanem amiket el akarok dobni, de számomra ez tűnt az egyszerűbb, logikusabb, átláthatóbbnak.

4. Korrelációs elemzés

A korreláció a statisztikában azt jelenti, hogy az általunk megadott két érték között van-e lineáris kapcsolat, ha van akkor ennek a nagyságát és irányát lehet megadni vele, azaz, hogy milyen az egymáshoz való viszonyuk. Ha a két érték közötti korrelációs érték nem nulla, akkor az egyben azt is jelenti, hogy a két érték között kapcsolat lelhető fel, és nem függetlenek egymástól. A korreláció csak a lineáris kapcsolatot tudja kimutatni, minden egyéb esetben más módszert kell alkalmazni a változók közti kapcsolat kimutatására. Normális eloszlású valószínűségi változók esetén kimondható viszont, hogy ha a két érték közötti korrelációs érték nulla, azaz korrelálatlanok, akkor egymástól függetlenek is, legalábbis nem lineáris típusú összefüggés van közöttük. A korábban említett korrelációs irányt a korrelációs együttható (r-rel jelölik) előjele mutatja meg, ha negatív akkor fordított arányosság van köztük, ha pozitív akkor egyenes arányosság, míg nagyságát a 0-1 tartományba eső érték. Minél nagyobb annál szorosabb kapcsolat lelhető fel a két változó között. A korrelációs érték meghatározásában nem játszik szerepet a változók értékének nagysága, viszont a jó mintavételezés elengedhetetlen. Éppen emiatt lesz majd érdekes a szakdolgozatom során elemzett korrelációs értékek, mert ugyanazt a két darab adatbázist három-három különböző verziójában néztem meg a korrelációjukat. Általában a bővebb minták nagyobb korrelációt tudnak kimutatni, mint egy szűkebb mintából merítettek. A mintavételezésen kívül érzékeny még a kiugró adatokra is, hiszen egy kimagasló adat ugyanarra az értékre javíthatja fel a korrelációs értéket, mintha egy nagyon alacsony érték lehúzná a többi mellette lévő adat korrelációját, és ezen két adat nélkül teljesen más eredményeket kellett volna kapni, de ezek miatt mégis ugyanazt kapjuk. Kapcsolódó irodalomjegyzék [10]

4.1 Megvalósítása

A korábban létrehozott hat darab encodolt adatbázisra alkalmaztam a corr() beépített függvényt. Ezután automatikusan kezelte, hogy csak a nummerikus típusú adatokat vizsgálja meg egymással szemben és egy táblázatban kiadta az értékeket. Viszont, hogy kicsit szemléletesebben legyenek az adatok megmutatva, a seaborn könytár segítségével egy heatmapon is ábrázoltam ezeket az értékeket, így szerintem sokkal jobban átláthatóak azonnal. Viszont az alapértelmezett méretek a táblázatnál túl kicsinek bizonyultak (szélesség 6, magasság 4), ezért a figsize változóval újraméreteztem őket. A négy darab label encodingos adatbázishoz elég volt a szélességet 9-re, a magasságot 8-ra állítni. Míg a két darab one hot

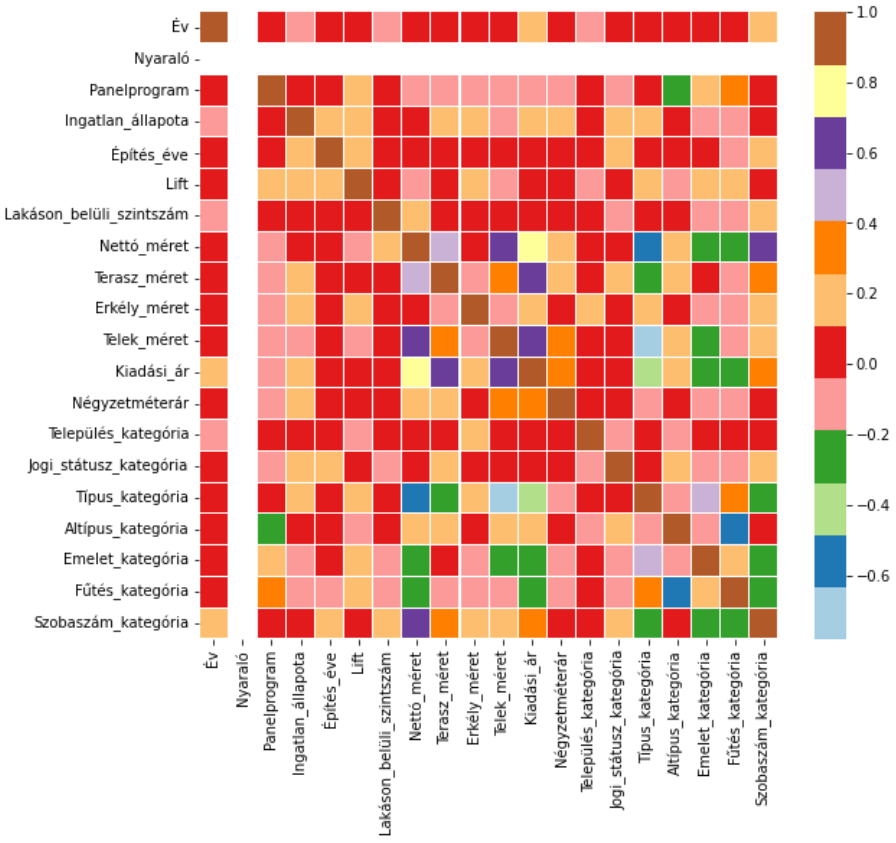
encodingos adatbázisnál a 44-45 darab oszlop miatt a szélességet 12-re, a magasságot 11-re kellett beállítanom, hogy az ábrán minden változó neve megjelenjen és elférjen olvashatóan. Az heatmap színeihez a Paired előre definiált színek kombinációját választottam a Qualitative színtérkébből. A 6 korrelációs vizsgálat során csak azokat az értékeket emelem ki, amik abszolút értéke 5-re kerekítendő felfelé, vagy annál nagyobb érték. Az ennél kisebb értékeket nem láttam értelmét vizsgálni, mivel nem elég erős kapcsolatot mutat a két változó között. A gépi tanulás során felhasznált változókat a korrelációs vizsgálat alapján választottam ki, így ez a része elengedhetetlen volt a szakdolgozatomnak.

4.2 Sima label encodinggal

4.2.1 Bérbeadott adatbázisra

A korrelációt itt egy 200 soros, 32 oszlopos adatbázison néztem meg, de mivel nem minden oszlopát encodoltam az object típusúak közül így 20 oszlop korrelációs értékét tudom megvizsgálni. A legtöbb magas korrelációs érték a nettó mérethez kapcsolódott: Nettó_méret – Telek_méret közti korrelációs érték 0.608971, Nettó_méret – Kiadási_ár közti korrelációs érték 0.713483, Nettó_méret – Típus_kategória közti korrelációs érték -0.549984, Nettó_méret – Szobaszám_Kategória közti korrelációs érték 0.646729. Ezek alapján a nettó méret és a telek mérete, kiadási ár, szobaszám kategória szorosan összefügg és egyenesen arányos kapcsolat van köztük, vagyis, ha a nettó méret nő akkor a többi értéke is, ha csökken akkor a másik 3 változó értéke is csökken. Ellenben a típus kategóriával fordított arányosság figyelhető meg, ami szerintem az encodolás hibájából történhet, hiszen a típusok és a hozzájuk rendelt értékek között nincs kapcsolat, így valószínű a családiház típus és a nagyobb területű ingatlan típusokhoz lettek hozzárendelve a kisebb számok, míg a kisebb területű ingatlanokhoz, mint például egy társasház, azokhoz pedig a nagyobbak. A második helyen a Telek_méret és a Kiadási_ár változók szerepelnek. A telek mérete változónak magas korrelációs értéke volt a korábban említett Nettó_mérettel, a Kiadási_árral, és a Típus_kategóriával. A Kiadási_árral 0.713483, a Típus_kategóriával -0.781419 volt a korrelációs érték. Így a kiadási ár és a telek mérete között egyenes arányosság áll fenn, ami logikus, hiszen minél nagyobb területű épületet bérlünk ki, az általában annál többe is kerül. A típus kategóriával úgy hiszem ugyanaz a helyzet áll fönn, mint ami a nettó méretek esetében is történt. A Kiadási_árnak a fentebb említett két változóval és a Terasz_méret változóval volt magas korrelációja. A Terasz_mérettel 0.559732-es értéket kaptunk, ami azt jelenti, hogy

általában a terasszal rendelkező ingatlanokat magasabb áron lehet kiadni. Ezeken kívül még egy magas korreláció volt megfigyelhető, az Altípus – Fűtés közti korrelációs érték -0.509962 volt, ami megint csak egy fordított arányosságot jelez, ami valószínű szintén az encodolás hibájából ered. Illetve a korreláció során kaptam még egy NaN sort is, a Nyaraló változóval, mert egy darab nyaraló sem volt köztük.

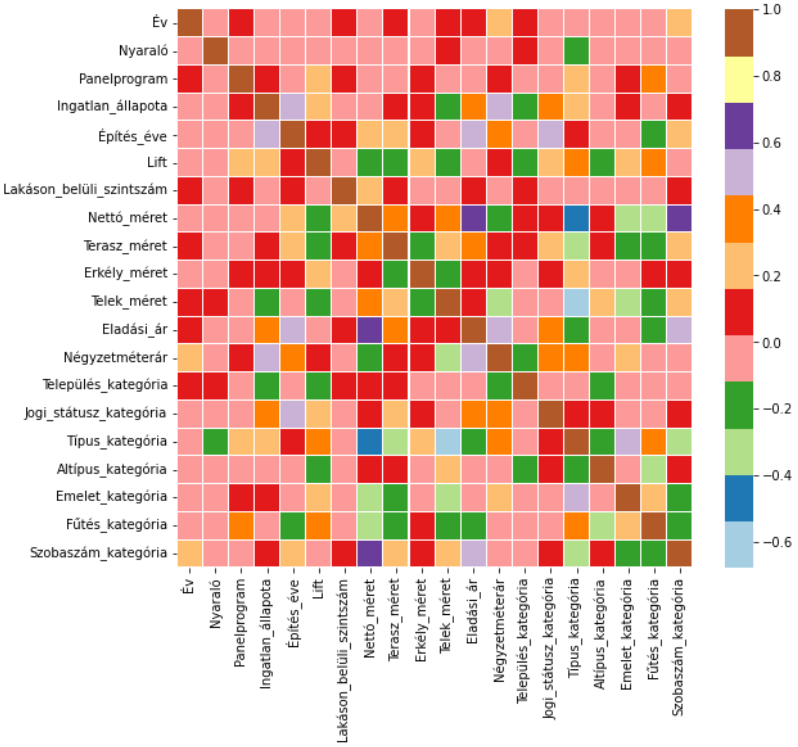


3. ábra, 4.2.1-es fejezet: korrelációs heatmap sima label encodinggal a bérbeadott ingatlanokra

4.2.2 Eladott adatbázisra

A korrelációt itt egy 500 soros, 32 oszlopos adatbázison néztem meg, de mivel nem minden oszlopát encodoltam az object típusúak közül így 20 oszlop korrelációs értékét tudom megvizsgálni. A legtöbb magas korrelációs érték 3 változóhoz kapcsolódott: Nettó_méret, Típus_kategória, Eladási_ár. A Nettó_méretet fogom először megvizsgálni. Itt megfigyelhető, hogy szintén fordított arányosságban áll a Típus_kategóriával, korrelációs értéke -0.490993, és egyenes arányosságban az Eladási_árral, korrelációs értéke 0.631802. Ezen kívül magas korrelációja volt még a Szobaszám_kategóriával, melynek értéke 0.670906, azaz, ha nő a nettó méret, több szoba figyelhető meg az ingatlan része ként. A Típus_kategória változónak a nettó

méreten kívül, magas korrelációja volt még a Telek_méret és az Emelet_kategóriával. A Telek_méret változóval fordított arányosság látható, melynek értéke -0.677007, pontosan úgy, mint a sima label encodingos bérbe adott adatbázisnál is. Az oka itt is ugyanúgy az encoding lehetett. Az Emelet_kategóriával 0.549204 korrelációs érték mutatható ki. Az Eladási_ár a Nettó_méreten kívül, magas korrelációs értéke van a Négyzetméterárral, melynek nagysága 0.547224, és hasonlóan magas étéke van a Szobaszám_kategóriával, 0.532802. Az eladási ár és a négyzetméter ár kapcsolata igencsak logikus, hiszen a négyzetméter árat az eladási árból számolják és az eladási árat is vissza lehet fejteni a négyzetméter árból. Az eladási ár és a szobaszám kategóriával való kapcsolata is egy egyenes arányosság, ami annyit jelent minél több szoba van az ingatlanban annál magasabb áron lehet eladni. Ezeken kívül még két viszonylag magas korrelációt találtam, az Ingatlan_állapota – Építés_éve, melynek korrelációs értéke 0.476443, és az Ingatlan_állapota – Négyzetméterár, melynek értéke 0.493190. Ezek ugyan nem olyan magas értékek, de mégis logikusak ezért nem szerettem volna őket kihagyni, hiszen minél jobb állapotban van egy ingatlan annál nagyobb négyzetméter áron tudják kínálni az ingatlanpiacon, illetve az újabb építésű házak is nagyobb valószínűséggel vannak jobb állapotban, mint egy nála 20-30 évvel korábban épített épület. Itt Nem volt olyan sorom, ahol csak 0 lett volna, mert eladott ingatlanok között volt olyan, amelyik nyaraló volt.

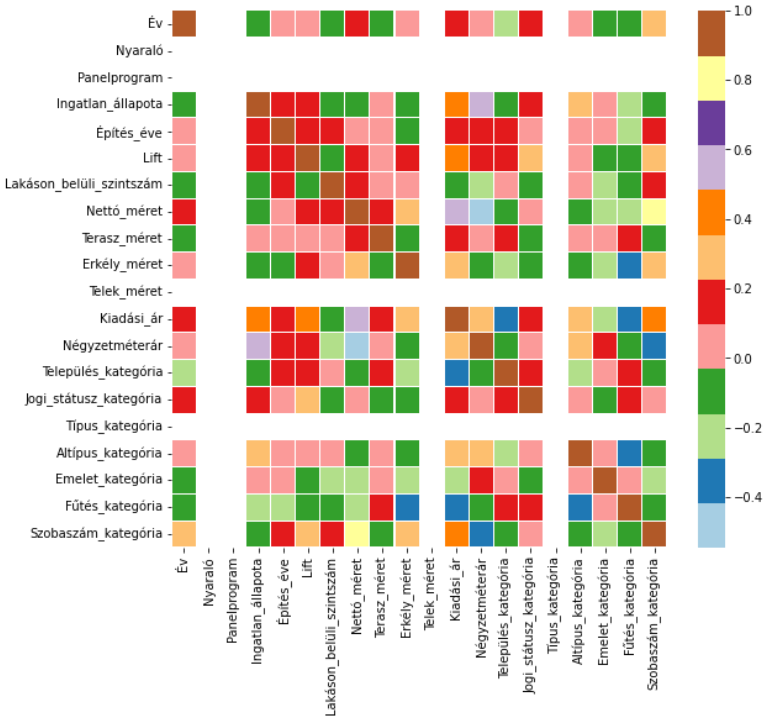


4. ábra 4.2.2-es fejezet: korrelációs heatmap sima label encodinggal az eladott ingatlanokra

4.3 Top5-ös label encodinggal

4.3.1 Bérbeadott adatbázisra

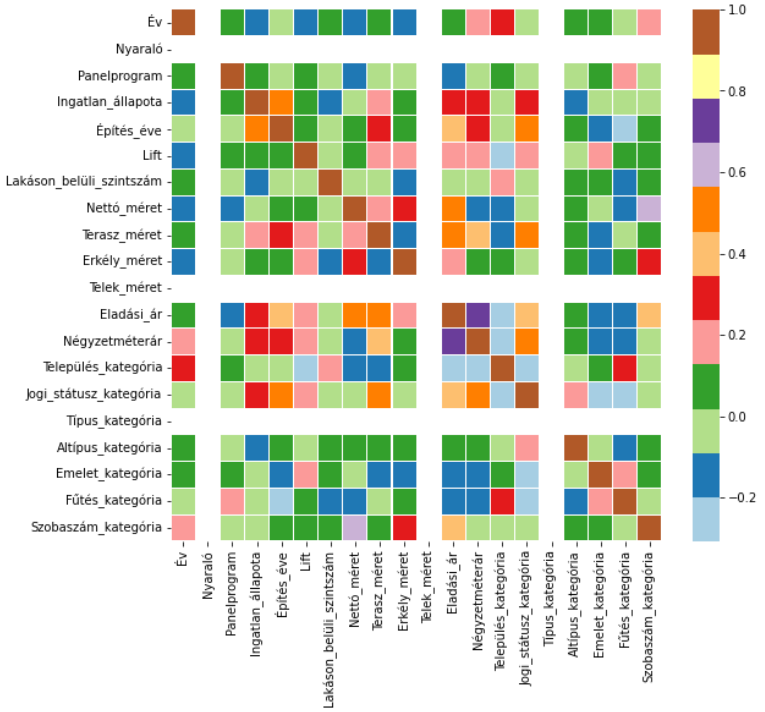
A korrelációt itt egy 59 soros, 32 oszlopos adatbázison néztem meg, de mivel nem minden oszlopát encodoltam az object típusúak közül így 20 oszlop korrelációs értékét tudom megvizsgálni. A legtöbb magas korrelációs érték a Nettó_méret változóhoz kapcsolódott: Nettó_méret – Kiadási_ár 0.575359, Nettó _méret – Négyzetméterár -0.545571, Nettó_méret – Szobaszám_kategória 0.750414. Ezek alapján megfigyelhető, hogy minél nagyobb méretű ingatlant vásárolunk, annál kisebb a kisebb a négyzetméterára, míg a kiadási ára ezzel arányosan növekszik, és érthető módon a szobák száma is vele együtt növekszik. A négyzetméterárnak volt még ezen kívül egy másik erős korrelációs kapcsolata, az Ingatlan_állapota változóval, korrelációs értékük 0.509483. Ezt már korábban más korrelációs vizsgálatnál is kifejtettem, hogy az ingatlanok állapota szoros kapcsolatban áll a négyzetméter árával is, hiszen egy jobb állapotú épületet sokkal többért lehet eladni. Ezen kívül látható még, hogy a Kiadási_ár és a Szobaszám_kategória között is felelhető egy egyenes arányosság, korrelációs értéke 0.468682. Ez annyit jelent, hogy minél több szoba van egy ingatlanban, annál magasabb áron lehet kiadni. Itt 3 darab NaN eredményű korrelációm lett, a Nyaraló, a Panelprogram és a Telek_méret változókkal, ugyancsak amiatt mert minden értékük 0 volt.



5. ábra 4.3.1-es fejezet: korrelációs heatmap label top5 encodinggal a bérelt ingatlanokra

4.3.2 Eladott adatbázisra

A korrelációt itt egy 100 soros, 32 oszlopos adatbázison néztem meg, de mivel nem minden oszlopát encodoltam az object típusúak közül így 20 oszlop korrelációs értékét tudom megvizsgálni. A legtöbb magas korrelációs érték az Eladási_ár változóhoz kapcsolódott: Nettó_méret – Eladási_ár 0.534730, Terasz_méret – Eladási_ár 0.467057, Eladási_ár – Négyzetméterár 0.732574. Ezek mind egyenes arányosságot követnek így, ha az eladási ár értéke növekszik vele arányosan nő a nettó méret is, a terasz mérete is, és a négyzetméter ás is. Ezen kívül még az Épitési_év változónak volt két darab magasabb korrelációja az Ingatlan_állapota változóval, melynek korrelációs értéke 0.465028, és a Jogi_státusz_kategória változóval, melynek korrelációs értéke 0.551819. Az építési év és az ingatlan állapota úgy kapcsolódik össze, hogy minél újabb egy ház, annál jobb állapotban van. A jogi státusznál tudni kell, hogy két féle változó értéket vehetett fel, a használtat és az újat, az encoding során a használt érték kapta a 0, az új érték kapta az 1es jelölő értéket, így ez azt állítja, hogy minél napjainkhoz korábbi évet nézünk, annál nagyobb eséllyel új jogi státuszú lesz az az épület. Ennél a korrelációnál is lett 3 darab NaN értékű korreláció, a Nyaraló és a Telek_méret azért, mert minden értéke 0 volt, míg a Típus_kategória azért, mert csak 5-ös értékelés volt. Mivel ez egy szűrt adatbázis volt, ezért várható volt, hogy lesznek ilyen helyzetek is.



6. ábra 4.3.2-es fejezet: korrelációs heatmap label top5 encodinggal az eladott ingatlanokra

4.4 Top5-ös one hot encodinggal

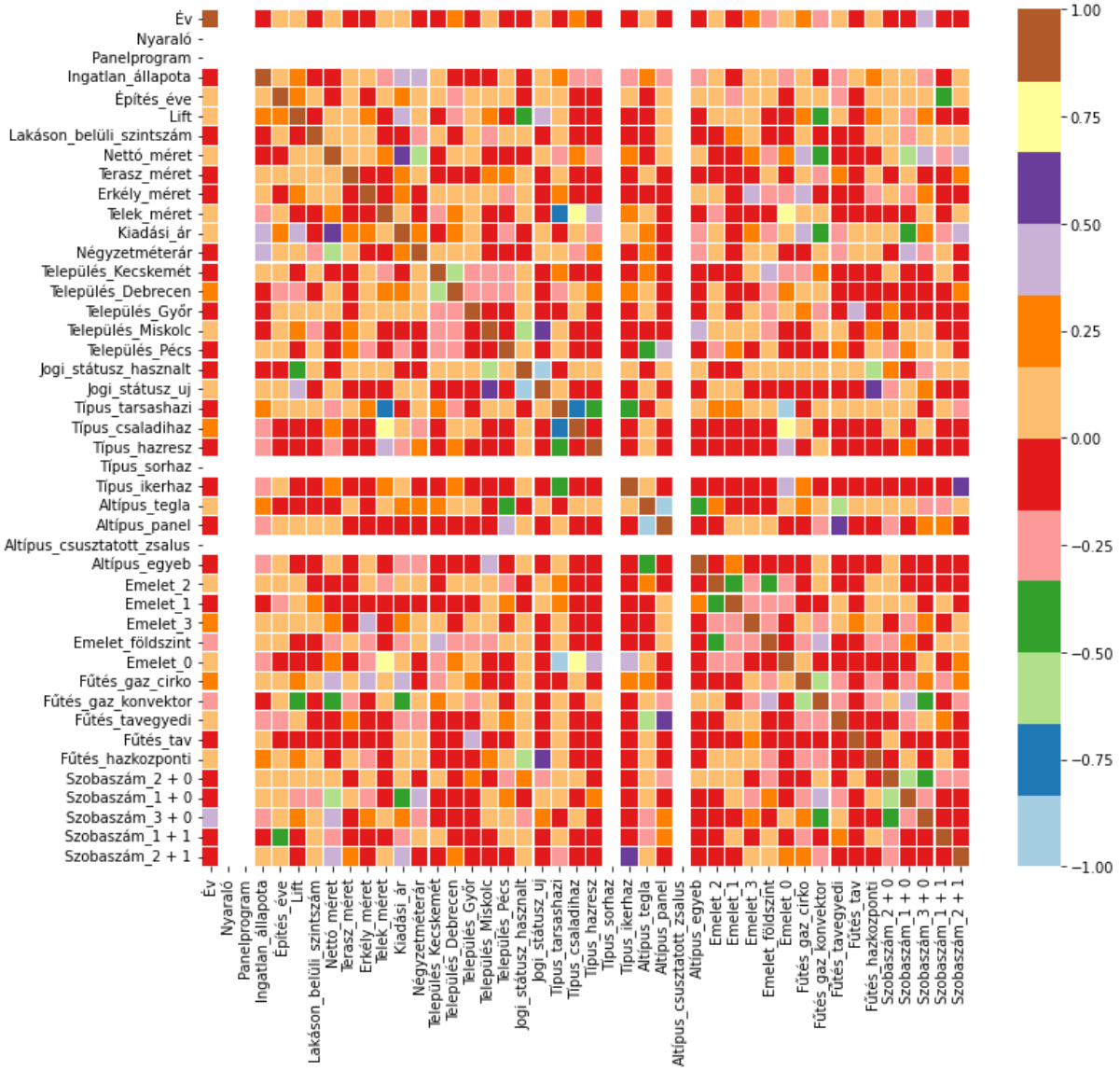
4.4.1 Bérbeadott adatbázisra

A korrelációt itt egy 64 soros, 49 oszlopos adatbázison néztem meg, de mivel nem minden oszlopát encodoltam az object típusúak közül így 44 oszlop korrelációs értékét tudom megvizsgálni. Mivel itt nem az eddigieknél látott encodinggal csináltam meg az alap adatbázist, amin megnéztem a korrelációt lesznek majd olyan változók is, amik a másik 4 adatbázisban nem megtalálhatók. A legtöbb magas korrelációs érték a Nettó_méret változóhoz kapcsolódott: Nettó_méret – Kiadási_ár 0.582635, Nettó_méret – Négyzetméterár -0.588616, Nettó_méret – Fűtés_gáz_cirkó 0.484256, Nettó_méret – Szobaszám1+0 -0.610512. Ezek közül kettő fordítottan arányos, kettő egyenesen. Fordítottan arányos a nettó méret és a négyzetméterár, ezt korábban már többször kifejtettem, ez azért van mert város és ingatlan típus függő, hogy mennyi a négyzetméter ár, illetve minél nagyobb alapterületű épületeket nézünk, a négyzetméter ár valószínűleg arányosan kevesebb lesz. Ezen kívül fordítottan arányos még a szobaszámok közül az 1 + 0 típussal, ami annyit jelent, hogy ez a típus inkább a kisebb méretű ingatlanoknál jelentősebb, a nagyobbakat már nem ez az elrendezés jellemzi. Egyenesen arányos a kiadási árakkal, ez érthető, hiszen minél nagyobb az ingatlan annál többe kerül kibérelni és egyenesen arányos még a gázcirkó típusú fűtéssel rendelkező ingatlanokkal, ez annyit jelent, hogy a gázcirkó fűtés technológiát inkább nagyobb méretű ingatlanoknál szokás használni, kisebbeknél például egy panelnél rendszerint táv-egyedi fűtés van. Ehhez kapcsolódik ez a korrelációs eredmény is, Kiadási_ár – Fűtés_gázcirkó 0.474999, ami azt mutatja, hogy a gázcirkó tényleg a magasabb kiadási költséggel rendelkező ingatlanokra jellemző inkább. A kisebb területű ingatlanokra, amit korábban említettem például panelekre, ez a korreláció megerősíti az elméletemet, Altípus_panel – Fűtés_táv_egyedi 0.632826. A táv-egyedi fűtésnek nem csak a panel altípussal volt magas korrelációs értéke, hanem a tégla építésű ingatlanokkal is, csak ezzel fordítottan arányos, Altípus_tégla – Fűtés_táv_egyedi -0.529971. Ez annyit jelent, hogy nem jellmező a tégla építésű házakra ez a fajta fűtés technológia. Érdekes módon a panel és a tégla között is kimutatható volt a fordított arányosság, értéke -0.858898, illetve a tégla és az Altípus_egyéb között is, értéke -0.487950. A gázcirkó fűtéssel fordítottan arányos volt egy másik fűtés típus, a Fűtés_gázkonvektor, értéke -0.619536, ez amiatt lehet, mert míg a gázcirkót nagyobb alapterületű ingatlanoknál használják, addig a gázkonvektort éppen ellenkezőleg, a kisebbeknél gyakoribb. A Telek _mérete

kapcsolatban áll a típus meghatározásával, Telek_méret – Típus_társas -0.829242, Telek_méret – Típus-családi 0.673199, és akár a szinttel is, Telek_méret – Emelet0 0.829242. Ennek az a magyarázata, hogy egy családiház nagyobb terület mérettel rendelkezik, mint egy másik típusú, jelen esetben egy társasház, emiatt van, hogy a társasház fordítottan arányos, a családiház pedig egyenesen arányos. Az emelt száma is egyenesen arányos a telek méretével, mert ebben az adatbázisban az Emelet0 kategória rendszerint a családiházaknál volt jelölve. Ezt alátámasztja a következő korrelációs érték is, Típus_családi – Emelet0 0.761793. Továbbá fordítottan arányosak egymással, Típus_családi – Típus_társas -0.761793.

Érdekes volt látni, hogy volt olyan változó páros, ahol kereken -1.00 lett a korrelációs érték, ezek voltak a Jogi_státusz_használt és Jogi_státusz_új. Nem véletlen, hogy pont köztük lett ez az érték, hiszen ezek egymás ellentétei, így bármilyen esetben, ha az egyik fentáll, akkor a másik biztosan nem. A fűtéssel kapcsolatban is lett erős korreláció a jogi státuszok között, Jogi_státusz_használt – Fűtés_házközponti -0.512516, Jogi_státusz_új – Fűtés_házközponti 0.512516. Ezek alapján kimodható, hogy az új házakra jellemzőbb a központi fűtés használata, mint a régebbi építésűekre. A jogi státusz alapján meg tudtam állapítani azt is, hogy Miskolcon egyre több új házat építenek, ebben segítségemre volt ezen korrelációs értékek, Település_Miskolc – Jogi_státusz_használt -0.512516, Település_Miskolc – Jogi_státusz_új 0.512516. Megfigyelhető volt még egy fordított arányú korreláció is két város között, Telelpülés_Kecskemét – Település_Debrecen -0.505781.

Magas korrelációs értéket kaptam még a Típus_ikerház – Szobaszám 2+1 0.568112, amiből levonható következtetés, hogy az ikerházakra ez a belső szoba szám eloszlása a jellemző.Szobaszámokkla kapcsolatban magas fordított arányosság volt látható, Szobaszám2+0 – Szobaszám1+0 -0.553283 között, illetve Építés_éve – Szobaszám1+1 -0.468663 között is. Ez utóbbi arra enged következtetni, hogy régebben sokkal több 1+1 szobás ingatlan épült, é s manapság ez annyira már nem jellemző. Ahogy korábban is, ennél a korrelációs vizsgálatnál is előjött, hogy Ingatlan_állapota – Négyzetméterár 0.486487 egyenesen arányosak egymással, azaz egyik nő, akkor másik is. Sajnos itt is kaptam NaN értékeket korrelációra a Nyaraló, a Panelprogram, a Típus_sorház, és az Altípus_csúsztatott_zsalus változóknál.



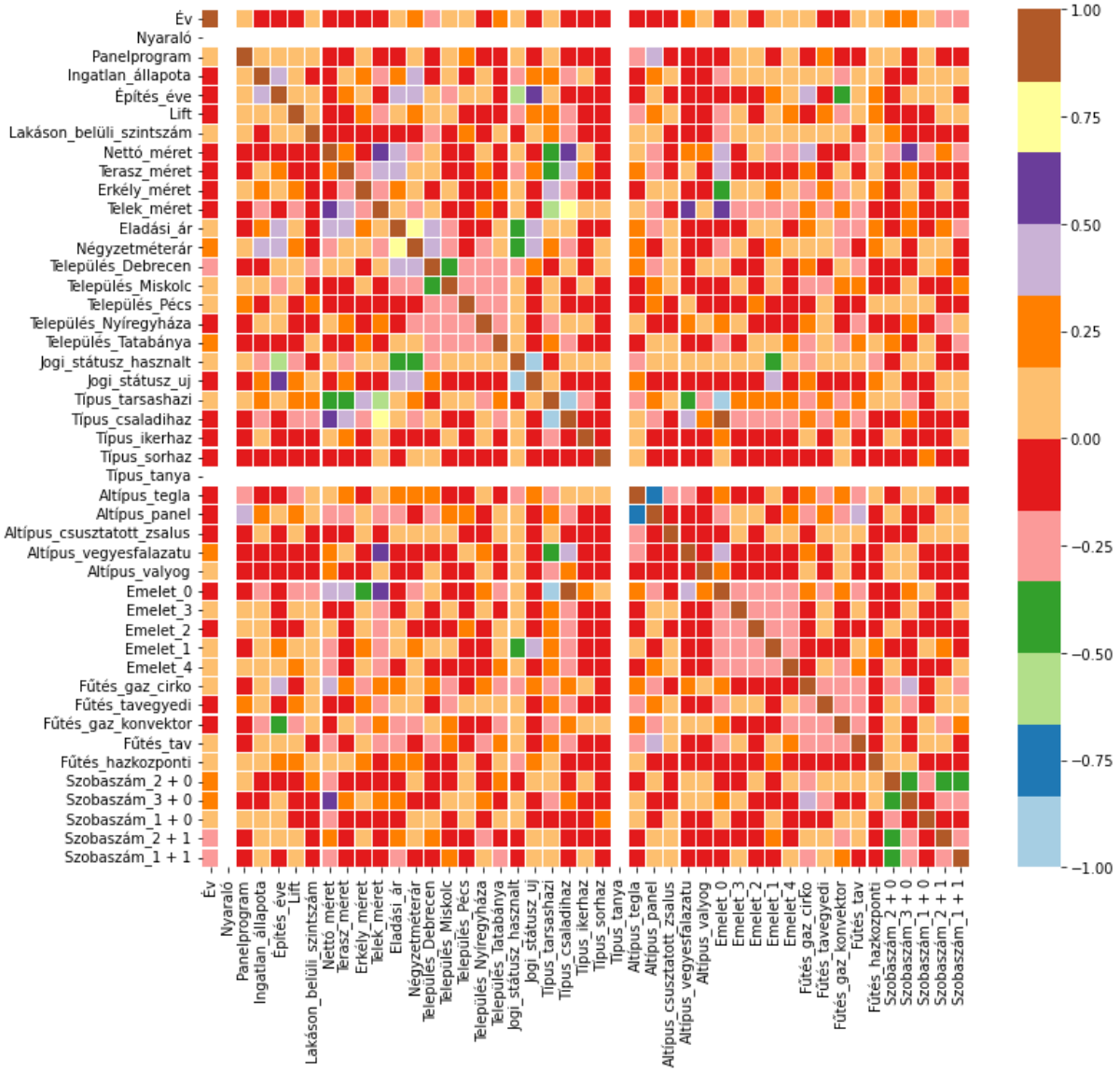
7. ábra 4.4.1-es fejezet: korrelációs heatmap onehot top5 encodinggal a bérelt ingatlanokra

4.4.2 Eladott adatbázisra

A korrelációt itt egy 130 soros, 50 oszlopos adatbázison néztem meg, de mivel nem minden oszlopát encodoltam az object típusúak közül így 45 oszlop korrelációs értékét tudom megvizsgálni. Mivel itt nem az eddigieknél látott encodinggal csináltam meg az alap adatbázist, amin megnéztem a korrelációt lesznek majd olyan változók is, amik a másik 4 adatbázisban nem megtalálhatók. A legtöbb magas korrelációs érték a Nettó_méret és a Telek_méret változókhoz kapcsolódott: Nettó_méret – Telek_méret 0.520766, Nettó_méret – Eladási ár 0.467468, Nettó_méret – Típus_társas -0.485104, Nettó_méret – Típus_családi

0.510284, Nettó_méret – Emelet0 0.485104, Telek_méret – Típus_társas -0.665053, Telek_méret – Típus_családi 0.681609, Telek_méret – Altípus_vegyes 0.511132, Telek_méret – Emelet0 0.665053. Ahogy már többi vizsgálatnál is kimutatta, a nettó méret és az eladási ár között itt is egyenes arányosság van, természetesen a telek méret és a nettó méret között is, viszont a nettó méret és a társas ház típus között fordított arányosság, valószínű, mert a társasházak területe nem szokott olyan nagy lenni, mint egy családi háznál, ami azért is fontos, mert ebben a szűrt adatbázisban társasházból ás családi házból volt a legtöbb, és a nettó méret és a családi házak között egyenes arányosságot mutatott ki, illetve a nettó méret és az emelet0 között is, hiszen a az emelet0 a családi házaknál volt a legtöbb helyen bejelölve. A nettó méretek és a típusok közötti összefüggés igaz a telkek mérete és típusokra is ugyanazon okból. A telek mérettel kimutatható szinte ugyanaz a korreláció, mint a nettó mérettel, mégpedig, hogy a telek mérete egyenesen arányos a családi ház típussal, a 0. emelettel és fordítottan arányos a társasházakkal. Egy dologban különbözik, hogy kimutatott egy új kapcsolatot, az altípusok közül a vegyessel mutat egyenes arányosságot. Ugyanakkor ezek között is sok függőség lelhető fel, Típus_társasház – Típus_családi -0.936301, Típus_társasház – Emelet0 -1.000000, Típus_családi – Emelet0 0.936301. Ezek azt mutatják, hogy ami társasház az kizárt, hogy a 0. emeleten legyen, illetve amit már említettem, hogy a 0. emelet javarészt a családi házaknál lett bejelölve. Más magas korrelációk voltak megfigyelhetők a jogi státusz változóinál is, Négyzetméterár – Jogi_státusz_használt -0.460022, Négyzetméterár – Jogi_státusz_új 0.460022, Jogi_státusz_használt – Jogi_státusz_új -1.000000, Építés_éve – Jogi_státusz_használt -0.507337, Építés_éve – Jogi_státusz_új 0.507337. Többnyire ugyanezek voltak megfigyelhetők a onehot top5 bérelt adatbázis esetén is, hogy a négyzetméterár az újabb ingatlanoknál magasabb, míg a használtnál épp fordítva, hogy a használt és az új változók közötti korrelációs érték -1.00, mert vagy az egyik vagy a másikat lehet megjelölni, így egyszerre sosem lehetnek jelen, illetve hogy minél közelebbi évet nézünk napjainkhoz építési évként természetesen egyre több az új építésű ház, amit eladásra kínálnak. Itt is megfigyelhető az építési év és az Ingatlan_állapota közti egyenes arányosság, melynek értéke 0.464051. Ami még fontos korrelációs kapcsolat, az az Eladási_ár – Négyzetméterár 0.670141. Ezek a változók szorosan összekapcsolódnak, de míg eddig az volt megfigyelhető, hogy ezek kapcsolata fordítottan arányos, itt egyenes arányosságot mutatott ki, ami jelentős változás, hiszen eddig az volt, hogy a kisebb ingatlannak nagyobba négyzetméterára, itt viszont pont fordítva, minél nagyobb az eladási ára, annál nagyobb a négyzetméter ára is. Ezeken kívül már csak egy kapcsolat maradt, az Altípus_tégla –

Altípus_panel -0.765153 fordított arányosságúak. Itt is volt két NaN értékű korrelációs sorom, a Nyaraló és a Típus_tanya.



8. ábra 4.4.2-es fejezet: korrelációs heatmap onehot top5 encodinggal az eladott ingatlanokra

5. Gépi tanulás

A mesterséges intelligencia egyik részét képezi a gépi tanulás. Ez a terület olyan rendszerekkel foglalkozik, amelyek tanulni képesek, azaz a megadott adathalmazokból tudást tudnak generálni. Sok féle algoritmus áll rendelkezésre, ezek közül nekünk kell kiválasztani, a feladatunknak megfelelőt, de utána majd az algoritmus a megfelelő betanítás után képes lesz szabályokat felismerni, akár egyedül, akár emberi segítség által, és ezen szabályok alapján általa még ismeretlen adatokra helyes előrejelzést fog tudni adni.

Szakdolgozatomban én 2 darab regressziós algoritmust alkalmaztam, de ezen kívül még rengeteg sok létezik. A regressziós feladatok felügyelt gépi tanulás részei. Az adatainkat két felé bontjuk, így kapunk egy tanító adatbázist, ami alapján az algoritmus egy modellt tud megtanulni és annak szabályait, illetve kapunk még egy kiértékelő adatbázist is, amin mérni tudjuk a modellünk pontosságát. Ezek segítségével képes lesz egy folytonos érték előrejelzésére a rendszerünk. Kapcsolódó irodalomjegyzék [11]

5.1 Lineáris regresszió

A lineáris regresszió egy olyan paraméteres regressziós modell, amely feltételezi a lineáris kapcsolatot a magyarázó- (X) és a magyarázott (y) változó között. Ha szeretnénk vizuálisan is elképzelni, ez azt jelenti, hogy lineáris regresszió becslése során a mintavételi adatok pontfelhőjére igyekszünk egy egyenest illeszteni a koordináta rendszerben. A célváltozónk értéke az egyenes által felvett értékek lehetnek. Kapcsolódó irodalomjegyzék [12]

5.1.1 Megvalósítás

A már korábban létrehozott hat darab adatbázissal dolgoztam itt is. Más módosítást már nem igényeltek, az encodingon és ahol kellett a kategóriákból az első 5 leggyakoribb típus kimentésén kívül. Pairplot-tal és relplot-tal próbáltam megnézni az adatok egymáshoz való viszonyát. Ezek alapján és a korrelációs adatok alapján döntöttem el, hogy mely változók alapján szeretném az előrejelzést megvalósítani. Rengeteg féle kombinációval próbálkoztam, de végül a legeredményesebbet tartottam csak meg. Fontos, hogy miután megvalósítottam mind a hat adatbázisra, össze akartam vetni egymással, a három különböző bérbeadott és a három különböző eladott ingatlanos adatbázisokat, és úgy gondoltam, hogy ehhez

elengedhetetlen, hogy ugyanazokkal a változókkal valósítsam meg mindnél a lineáris regressziót.

Először az importálásokat kellett végrehajtanom, fontos volt, hogy minden meglegyen, különben nem működött volna az algoritmus. Szükséges volt a LinearRegression, mert ennek segítségével tudtam a modellt rá illeszteni a tanuló adatbázisomra. Hasonlóan fontos volt még a train_test_split függvény, aminek a segítségével fel tudtam darabolni az adataimat egy tanuló és egy kiértékelő adatbázisra. Végül importálnom kellett még a mean_squared_error, r2_score-t, mert ha elkészült a modell, az illesztés, akkor ezek segítségével tudom megállapítani mennyire pontos a programom.

Magában a kódban először két változóba, jelen esetben X és y változókba (mert lineáris regressziónál így szokás jelölni, X változóba teszem, azokat a változókat, amik alapján szeretném előre jelezni az y változóban tárolt változót, ami nálam a Kiadási_ár volt) kellett megadnom azokat az elemeket, amikkel dolgozni szeretnék. X elemeiként több kombinációban próbáltam, de végül csak a legeredményesebbekkel folytattam a munkát. (Ezeket válogattam bele végül X-be: Év, Ingatlan_állapota, Nettó_méret, Négyzetméterár, Település_kategória, Típus_kategória, Fűtés_kategória) Ezután a train_test_split függvény segítségével négy változóba mentettem ki a függvény segítségével feldarabolt részeket. Kellett külön X_train, y_train, és X_test, y_test. Azt, hogy milyen arányban szeretném felosztani az adatokat a tanuló és kiértékelő adatbázisok között a train_size változó értékadásával tettem meg, de megtehettem volna a test_size változóval is. Itt is több verzióban tekintettem meg az eredményeket, és végül a 90% tanuló és 10% kiértékelő adtabázis arányokkal értem el a legmagasabb eredményeket. Majd ezután új modellt hoztam létre a tanuló adatbázisokból (X_train, y_train) a fit() függvény segítségével. Majd a predict() függvény segítségével előre jelezhetünk a kiértékelő adatbázisunk elemeire. Ez alapján tudjuk meghatározni az r^2 és az rmse értékét.

Mindkét változó értéke mutatja, hogy mennyire pontosan jelez előre a rendszerünk. Az r^2 a meghatározási együttható, ez teszi lehetővé, hogy megmérjük a modell válasz- és prediktor változói közötti kapcsolat erősségét. Ez az R korrelációs együttható négyzete, tehát értékei a 0,0–1,0 tartományban vannak. Minél nagyobb az értéke annál nagyobb százalékban magyarázzák a válaszváltozó varianciáját az előrejelző változók. Míg az rmse (root mean squared error) a gyökérátlag négyzet alakú hibát jelöli. Ez megmutatja, hogy mennyire hibás / mennyire rossz a modellünknek az előrejelzései a tényleges megfigyelt értékekhez képest. Emiatt minél alacsonyabb az értéke annál jobb. Sajnos nekem a kevés adatom és / vagy a

modell hibájából fakadva ez az érték elég magas lett mindenhol, de ennek ellenére is jelezhet előre egészen jól a rendszerem, mert az r^2 értékeim viszont kimondottan magasak voltak.

A modellnek két fő attribútuma van, a coef_, amely a modellünk együtthatóinak tömbjét tárolja és az intercept_, amely a lineáris modellünk y-metszetét tárolja. Ezek azért voltak fontosak, hogy tudjam más, a modell által még nem látott adatokra is tesztelni a rendszert. A coef tömbjében letárolt értékeket sorra összeszoroztam a hozzájuk tartozó változóval, majd az interecept értékét hozzáadtam. Így el is készült a saját függvényem, amivel tudok előrejelzéseket kérni különböző adatokra.

5.1.2 Eredmények

Mivel tudtam előre, hogy a végén mind a bérbeadott három, mind az eladott három adatbázis alapján készült modellek előrejelzéseit össze szeretném majd hasonlítani, így ugyanazokkal a változókkal írtam meg őket, ugyanazzal a felosztással a tanuló és kiértékelő adatbázisokra. Ugyan az r^2 értéke nem pontosan ugyanaz lett, mégis eléggé hasonló értékek jöttek ki. A bérbeadott ingatlanoknál az egyszerű label encodingos adatbázissal 81%, míg a label encoding top5ös adatbázissal 94% és a one hot encoding top5ös adatbázissal 87%. A százalékok alapján aza következtetés vonható le, hogy legpontosabban a label encoding top5ös adatbázissal jelez előre, míg második helyen a one hot encodingos adatbázissal és a harmadik helyen a sima label encodingos adatbázis áll. Érdemesnek gondoltam ezt az állítást az rmse értékekkel is alátámasztani, sima label 267941399.83, label top5 41486184.11, one hot top5 68754363.07. Ezek alapján ugyanaz a sorrend állapítható meg, mint az r^2 értékekkel. Mindhárom rendszert ezek után ugyanazokkal az adatokkal akartam tesztelni, ezért olyan adatokat adtam meg, ami mindhárom adatbázissal tesztelhető. Eladás évének 2020at, Ingatlan állapotának 5ös értéket, Nettó méretnek 150m^2-ert, Négyzetméter árnak 2000 Ft-ot, Településnek Debrecent, az ingatlan Típusának társasházat, Fűtésnek pedig gáz cirkót. Ezekre az adatokra a következő előrejelzések születtek a Kiadási ár értékére: sima label 297188.26, label top5ös 289497.71, one hot top5ös 275946.04. A fenti adatokból levont következtetésem szerint a label top5-nek kellett a legjobban megközelítenie a pontos értéket, és a one hot top5 az ingatlan kiadási ára alá, míg a sima label fölé lőtte az értéket. Ugyanezeket az adatokat vizsgáltam az eladott ingatlanok során. Eladott ingatlanoknál is először az r^2 értékét vizsgálnám, a sima label encodingos adatbázissal 91%, label top5ös adatbázissal 94%, one hot encodingos adatbázissal pedig 90%, bár, ha kerekítenénk ez is elérné a 91%-ot, mert 90.7-et 91-re kéne felkerekíteni. Itt megfigyelhető, hogy sokkal közelebbi értékek jöttek ki, de még

itt is a label top5-ös adatbázissal éri el a legjobb eredményt. Következő lépésként megvizsgálom az rmse-k értékét is, egyszerű label adatbázissal 10719713136251.73, label top5-ös adatbázissal 1994285179717.90, és végül one hot top5-ös adatbázissal 5336027942344.90. Ezen adatok alapján viszont ugyanaz a minta figyelhető meg, mint a bérelt ingatlanoknál, label top5ös adatbázissal éri el a legjobb eredményt, majd a onehot top5ös, és végezetül a sima label. Az eladott ingatlanok adatbázisaival betanított három előrejelző rendszert is ugyanazon adatokkal teszteltem le, mint a bérbeadottak esetében. Itt ezek az értékek születtek sima labelnél 18961158.76, label top5nél 19724195.19, onehot top5nél 21514022.06. A korábbi következtetések alapján valószínűleg a label top5os adatbázissal betanított rendszer jelzett előre legpontosabban, és a bérbeadott ingatlanoknál megfigyeltel ellentétben itt a onehot lőtte fölé, és a sima label pedig alá az előrejelzett értéket, ha a label top5ös-höz viszonyítjuk. Összességében vizsgálva, a két darab label top5ös adatbázisnak volt a legkevesebb sora a másik négy adatbázishoz képest, mégis ezek érték el a legpontosabb előrejelzést a legkisebb hibával. Ezek alapján egy jól megszűrt adatbázis többet érhet, egy olyannal szemben, ami ugyan több adatot tartalmaz, de sokkal többféle típusú információ megtalálható benne kisebb mennyiségben, mert ez csak megnehezíti a modell dolgát a minél pontosabb előrejelzésben. Kapcsolódó irodalomjegyzék [13]

5.2 Döntési fa

Maga a döntési fa egy tanuló algoritmus, ami megtanul egy modellt, és ez alapján meghatározza a jellemzők közötti kapcsolatokat. A döntési fák úgy állnak össze, hogy van egy gyökér csúcs, amiből kiindulnak belső csúcsok, ezek a csúcsok egy-egy jellemzőt képviselnek, ezekből a csúcsokból vagy újabb belső csúcsok mennek ki, vagy a csúcs gyerekei, lehet mindkettő is. Egy csúcs gyerekéből már nem jön ki több csúcs, aza vége annak az ágnak. Ezek a csúcsok az adott jellemző egy-egy lehetséges értékét jelentik, ezek alap esetben diszkrét változók, de én regressziós döntési fával dolgoztam, így nálam ezek értéke vagy egy konstans érték vagy egy lineáris regressziós modellnek kellett lennie. A működési elven nem sokban változtat, ugyanúgy, ha egy egyedre döntést akarunk hozni, a fa gyökeréből kell kiindulnunk és tovább haladni a belső csúcsokra, egészen addig amíg egy gyerek csúcsba, levélbe nem érkezünk. Fontos, hogy nem szükséges, hogy minden levél ugyanazon a mélységen legyen, és hogy nem minden jellemző fog szerepelni, de lehet, hogy lesz, ami többször is meg fog jelenni. Ezt a módszert sokkal könnyebb értelmezni, mint a lineáris regressziót, hiszen vizuálisan is látható lesz minden lépés, ha kirajzoltatjuk, viszont csak nagyon nagy tanuló adatbázissal

működik igazán precízen pontosan, illetve itt nem lehet meghatározni, hogy mely információkat használja fel mindenképpen, önállóan dönti el. Kapcsolódó irodalomjegyzék [14]

5.2.1 Megvalósítás

A megvalósításához szükség volt különböző könyvtárak, algoritmusok importálására, először a döntési fa osztályozó algoritmust importáltam a sklearn könyvtárból és ezt egy dt nevű változóba mentettem le, ezután encodolni kellett volna az adatokat, de nekem ez már korábbról készen volt, így egy új DataFrame-et hoztam létre df_dontesi_fa_proba néven és abba belemásoltam a df_masolat_berelt változóm tartalmát a copy() függvény segítségével, hogy ne az eredetin módosítsak. Ezután a classlabel_proba változóba megadtam, hogy a Kiadási_ár változómra szeretném az előrejelzést megvalósítani. Majd minden olyan változót eldobtam a df_dontesi_fa_proba változómból, ami nem numerikus típusú, a pop függvény segítségével. Ezek után az adatbázisom készen áll, hogy összekapcsoljam a classlabel_proba változómmal, így a fit() függvény segítségével alkalmaztam a dt változóban lévő döntési fa osztályozó algoritmust a df_dontesi_fa_proba és classlabel_proba változóimra. Ezek után a pydot és a graphviz könyvtárakat votl szükséges importálni, hogy a döntési fát vizuálisan is meg tudja jeleníteni a program. Majd a dt változóban megadtam a megfelelő módon a minimum levelek számát, illetve a maximum mélységet, ameddig lemehet a fa. Fontos, hogy itt több verziót kipróbáltam, de a legjobb eredményt ezekkel az arányokkal értem el, min_samples_leaf=20, max_depth=6. Majd az importált graphviz-zel a megfelelő imputokat megadva lefutott a döntési fa osztályozó algoritmus és kirajzolódott a döntési fa is. Ezután a tanulási folyamat következett, az adatbázis, amin véghez vittem az eljárást eredetileg 200 soros volt, így az első 150 adatot adtam meg belőle tanulási céllal az az algoritmusnak, a maradék 50 sort pedig a prédikáció tesztelésére, hogy a korábban megtanult modellt mennyire jól tudja alkalmazni, mennyire pontos. Ahhoz, hogy meg tudjam tekinteni ezt az értéket importálni volt szükséges a sklearn.metrics könyvtárból az accuracy_score metódust. Ezután megadtam paraméternek a prediction változót és a classlabel_proba utolsó 50 sorát.

5.2.2 Eredmények

Sajnálatos módon ez a módszer nem volt ideális az én adatbázisaimhoz, mivel a döntési fákat akkor érdemes alkalmazni, ha elsősorban diszkrét jellemzőink van és ezekből nincsen

sok, rengeteg tanuló adat áll rendelkezésünkre és ha feltételezni tudjuk, hogy bonyolultabb kapcsolat lelhető fel közöttük. Az én adatbázisaimra ugyan igaz, hogy van sok komolyabb összefüggés a változók között, ahogy azt a korreláció elemzésnél láthattuk, de ez a modell nem megfelelő rájuk, hiszen a változóim értéke javarészt nem diszkrét volt, illetve a változó, amire előre jelezni szerettem volna sem diszkrét, hanem folytonos és a legtöbb adattal rendelkező adatbázisom is 500 soros volt csupán. Ezek miatt már az első próbálkozás után látszott, hogy nem lesz olyan eredményes, mint a lineáris regressziós modell, így végül csak két adatbázisra valósítottam meg. Ezeknél akárhogy módosítottam a paramétereket, a df_dontesi_fa_proba adatbázisommal a legmagasabb előrejelzési pontosság 60% volt, míg a másik adatbázisommal, a df_masolat_labeltop5_KESZ_berelt-tel 50%. Ezek miatt a többi adatbázisra nem láttam értelmét megvalósítani.

5.3 Eredmények összegezve

		Pontosság	
Módszer	Encoding	Bérbeadott ingatlan	Eladott ingatlan
Lineáris regresszió	label	81%	91%
	label top5	94%	94%
	one hot top5	87%	91%
Döntési fa	label	60%	-
	label top5	50%	-
	one hot top5	-	-

6. Összefoglaló

A szakdolgozatom során sikerült bizonyítanom a kollerációs elemzéssel, hogy az adatok között rengeteg egyenesen arányos és fordított arányosság lelhető fel, ezzel kimutatható, hogy komoly logikai kapcsolatok vannak egy ingatlan különböző tulajdonságai és annak bérbeadási, illetve eladási árának meghatározás között. Fontosnak tartom kiemelni, hogy a korrelációs vizsgálatoknál általában a nagyobb adathalmazok magasabb korrelációt tudnak kimutatni, ha nincsenek benne kiugró értékek. Az én vizsgálataim során éppen ellenkezőleg a kisebb, szűrtebb adatbázisokra kaptam több és erősebb korrelációs értékeket, azért, mert az eredeti két adatbázisban ömlesztve voltak az adatok és emiatt rengeteg kiugró érték volt, ami negatívan befolyásolta a korrelációs eredményeket, de miután megszűrve, csak a leggyakoribb változókkal vizsgáltam sokkal magasabb és erősebb kapcsolatokat tudott kimutatni. Az ott szerzett tudásból fel tudtam építeni a lineáris regressziós modellemet, a megfelelő változókat kiválasztva viszonylag elég pontos előrejelzéseket tudtam velük elérni. Több módszerrel is meg akartam próbálkozni, így nem csak a lineáris regressziós modellt alkalmaztam, hanem a regressziós döntési fákkal is megpróbáltam egy hasonlóan pontos előrejelzést megvalósítani. A modell leképezése sikeres volt, de sajnos az adataim ehhez a modellhez nem voltak megfelelőek, mert nem állt rendelkezésemre elég nagy adathalmaz a modell tanításához, illetve az adatok többsége folytonos volt, nem diszkrét. Emiatt csak a lineáris regresszióval tudtam kellően pontos előrejelzéseket adni. Természetesen egy még nagyobb adatbázissal még az általam elért eredményeknél is sokkal pontosabb előrejelzéseket lehetne elérni. Ezzel akár az egész ingatlanpiacot modernesíteni lehetne és gépi tanulás által meghatározni a különböző ingatlanok bérbeadási és eladási értékét, hogy senki se értékelje túl vagy éppen alul a saját ingatlanát. Nyilván ez a rendszer sem lenne teljesen objektív, mivel néhány változó értékének a meghatározása eléggé szubjektív, ilyen például az adott ingatlan állapota, vagy hogy éppen az adott kilátás kinek mennyire fontos egyáltalán és hogy ki milyen kilátást részesít előnyben. Ennek ellenére én nagy potenciált látok ebben az automatizációban a jövőre tekintve és ennek elengedhetetlen része a megfelelő modell kialakítása és a jó módszer kiválasztása.

Irodalomjegyzék

Az összes weboldalra mutató link utolsó megtekintési dátuma: 2022.05.05-e.

[1] Pandas könyvtár - Python Pandas - Introduction (tutorialspoint.com)

[2] Sklearn könyvtár – Scikit Learn - Introduction (tutorialspoint.com)

[3] Graphviz könyvtár – Graphviz — graphviz 0.19.2 documentation

[4] Matplotlib könyvtár – Matplotlib - Wikipedia

[5] Pyplot könyvtár – Pyplot tutorial — Matplotlib 3.5.0 documentation

[6] Seaborn könyvtár - An introduction to seaborn — seaborn 0.11.2 documentation (pydata.org)

[7] Label encoding – az oldal szerzője: Chris Moffitt - Guide to Encoding Categorical Values in Python - Practical Business Python (pbpython.com)

[8] Encoding ábrák - Feature Engineering: Label Encoding & One-Hot Encoding - Fizzy

[9] Onehot encoding - Pandas get dummies (One-Hot Encoding) Explained • datagy

[10] Korreláció - Korreláció – Wikipédia (wikipedia.org)

[11] Gépi tanulás – az oldal szerzője: Farkas Richárd - Gépi tanulás a gyakorlatban (u-szeged.hu)

[12] Lineáris regresszió - Scikit Learn - Linear Regression (tutorialspoint.com)

[13] R^2 és RMSE értelmezése – az oldal szerzője: Willie Wheeler - Evaluating linear regression models using RMSE and R^2 | by Willie Wheeler | wwblog | Medium

[14] Döntési fa - Scikit Learn - Decision Trees (tutorialspoint.com)

[15] OtthonCentrum ingatlanokra szűrési beállításai - Részletes ingatlankereső (oc.hu)

Nyilatkozat

Alulírott Bobák Vivien gazdaságinformatika szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Intézet Számítógépes Optimalizálás Tanszékén készítettem, gazdaság informatikus BSc diploma megszerzése érdekében.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat használtam fel.

Tudomásul veszem, hogy szakdolgozatomat a Szegedi Tudományegyetem Informatikai Intézet könyvtárában, a helyben olvasható könyvek között helyezik el.

2022.05.12. aláírás

Köszönetnyilvánítás

Szeretnék ezúton köszönetet nyilvánítani a témavezetőmnek Dr. London András István egyetemi adjunktusnak, akinek köszönhetően ez a szakdolgozat elkészült. Köszönöm a türelmét és a segítséget, amelyeket a konzultációk során adott, illetve a segédanyagokat, amiket megosztott velem, ezzel segítve a dolgozatom elkészülését.
Köszönettel tartozom továbbá az OtthonCentrum ingatlanközvetítő cégnek, és cégvezetőjének, aki engedélyezte és kiadta a szakdolgozatom alapjául szolgáló két darab adatbázist.