



Lakásárak becslése, befolyásoló változók elemzése

Haladó adatelemzési módszerek labor (BMEVITMMB10)

Név	Neptun	E-mail
Lovácsi Kristóf	N3EEWB	lovacsi.kristof@gmail.com
Szladek Máté Nándor	TGPZTT	szladekmate@gmail.com
Tóth Ádám László	TK6NT3	adamlaszlototh@edu.bme.hu



Tartalom

Tartalom	2
1. Bevezetés	4
1.1. Feladatkiírás	4
1.2. A projekt célja	4
2. Adatkészlet ismertetése	5
2.1. Adatforrás bemutatása	5
2.2. A felhasznált változók leírása	5
3. Adatelőkészítés és feltáró elemzés	7
3.1 Bevezetés az adatelőkészítéshez	7
3.2 Adattisztítási lépések	7
3.3 Feltáró adatelemzés	19
4. Modellezési stratégia és megvalósítás	24
4.1 Modellválasztás és értékelési módszertan	24
4.2 Alkalmazott modellek bemutatása	24
4.3 Feature importance vizsgálat	26
4.4 Modellek összehasonlítása	28
5. Felhasználói felület, alkalmazás	30
5.1 Alkalmazás architektúra	30
5.2 Frontend működése	30
5.3 Backend működése	31
5.4 Fejlesztési eszközök és környezet	31
6. Komondor szuperszámítógép használata	31
6.1 Környezet kialakítása	32
6.2 Tapasztalatok a használat során	33
6.3 Technikai kihívások és workaroudok	33
6.4 Komondor - Összegzés	33
7. Összefoglalás	34

1. Bevezetés

1.1. Feladatkíírás

A feladat egy olyan megoldás kidolgozása, amely lakáshirdetések adatai alapján becslést ad az ingatlanok irányárára, valamint feltárja az árazást befolyásoló tényezőket. A munka során a rendelkezésre álló adatok feldolgozása, az információk tisztítása és előkészítése, a változók közötti kapcsolatok feltárása, valamint különböző előrejelzési módszerek alkalmazása és összehasonlítása szükséges.

A feladat része egy olyan alkalmazás kialakítása is, amely lehetővé teszi az elkészült megoldások felhasználóbarát elérését és az árbecslés eredményeinek bemutatását.

1.2. A projekt célja

A projekt célja a feladatkíírásban megfogalmazott feladat lépéseinek megvalósítása, amely során az alábbi konkrét részfeladatok elvégzése szükséges:

- Az adatkészlet szerkezetének és a rendelkezésre álló jellemzőknek a feltérképezése.
- Az adatok előkészítése, beleértve az adattisztítást, a hiányzó értékek kezelését, az outlierok vizsgálatát és a változók átalakítását.
- Adatfeltáró elemzések és adatvizualizációk készítése a befolyásoló tényezők vizsgálatához.
- Többféle gépi tanulási modell (például döntési fa, random forest, gradient boosting) alkalmazása az irányárak becslésére.
- A modellek teljesítményének értékelése különböző metrikák alapján, valamint a modellek összehasonlítása.
- A változók fontosságának elemzése a predikció szempontjából.
- Egy alkalmazás kialakítása, amely lehetővé teszi a modell használatát, az árbecslések elkészítését, valamint az eredmények vizuális megjelenítését.

A fenti célok megvalósítása a projekt során meghatározott mérföldkövek (adat-előkészítés, adatvizualizáció, modellezés, értékelés, alkalmazás) mentén történik.

2. Adatkészlet ismertetése

2.1. Adatforrás bemutatása

A projekt során felhasznált adatkészlet a **ME ingatlan.com adatbányászati verseny** (Kaggle, 2018) versenyanyagának melléklete, amely Budapesten található lakáshirdetések adatait tartalmazza. A dataset célváltozója az ingatlanhirdetésekből megadott **irányár**, amely regressziós modellezési feladat alapjául szolgál. Az adatkészlet összesen **262 235 sorból és 23 oszlopból** áll, ahol minden sor egy-egy hirdetés adatait rögzíti. Az oszlopok között numerikus, kategóriális és szöveges változók is szerepelnek.

Az adatbázis nem tartalmaz pontos földrajzi koordinátákat vagy címadatokat, a lokáció kerület és városrész szintjén jelenik meg. A modellezés szempontjából több változó további előfeldolgozást, például kategóriák kódolását és szöveges mezők átalakítását igényli.

Az adatok a **Kaggle felhasználói feltételei** szerint oktatási és kutatási célra szabadon felhasználhatók a verseny adatainak letöltésekor kötelezően elfogadott licencfeltételek mellett. Az adatkészlet a verseny hivatalos oldalán érhető el:

<https://www.kaggle.com/competitions/me-ingatlan-com/data>

Ez az adatkészlet képezi a projekt alapját, és szolgál a lakásárak becslésére, valamint a befolyásoló tényezők vizsgálatára irányuló elemzések kiindulópontjául.

2.2. A felhasznált változók leírása

Az adatkészlet különböző típusú ingatlanjellemzőket tartalmaz, amelyek az ingatlan fizikai adottságait, elhelyezkedését, felszereltségét és a hirdetések metaadatait rögzítik. Az alábbiakban csoportosítva bemutatjuk a főbb változókat és azok tartalmát.

Alapvető ingatlanadatok

- **county**: A vármegye, amely minden esetben „Budapest”.
- **city**: A város (Budapest) neve mellett kerület számozások is szerepelnek, főként ott, ahol az irányítószám hiányzik.
- **postcode**: Az ingatlan irányítószáma. Nem minden rekordnál érhető el, de a kerület minden esetben azonosítható.
- **property_type**: Minden esetben „flat” (lakás), ezért nem hordoz megkülönböztető információt.
- **property_subtype**: Az ingatlan altípusa, például téglalakás, panellakás, sorház, illetve hiányzó értékek.

Ingatlan állapota és elhelyezkedése

- **property_condition_type**: Az ingatlan állapota kilenc kategóriában van megadva (pl. felújítandó, új építésű).

- **property_floor**: Az adott lakás emeletének megnevezése; vegyesen tartalmaz számozott és szöveges értékeket, valamint hiányzó adatokat.
- **building_floor_count**: Az épület emeleteinek száma, 10-ig számozva, valamint „több mint 10” kategóriával és hiányzó értékekkel.

Kilátás és tájolás

- **view_type**: A lakás kilátása: belső udvari, kerti, panorámás vagy utcai. Bizonyos esetekben hiányzó értékeket tartalmaz.
- **orientation**: Az ingatlan égtájak szerinti tájolása, többféle szöveges értékkel és hiányzó adatokkal.

Felszereltség és komfort

- **garden_access**: Igen/nem változó arra vonatkozóan, hogy elérhető-e kertkapcsolat. Sok helyen hiányzik az érték.
- **heating_type**: Az ingatlan fűtésének típusa, többféle elnevezéssel, nem minden esetben kitöltve.
- **elevator_type**: Igen/nem típusú mező, amely a lift meglétét jelzi; szintén számos hiányzó értékkel.

Helyiségek száma és mérete

- **room_cnt**: A szobák száma, de az értelmezés problémás (előfordulnak például feltehetően tévesen méretként megadott értékek is).
- **small_room_cnt**: A kisebb helyiségek száma, hasonló értelmezési problémákkal.
- **property_area**: Az ingatlan hasznos alapterülete négyzetméterben, reális tartományban (5–70 m²).
- **balcony_area**: Az erkély területe, de tartalmaz extrém értékeket is (több száz m²).

Időbeli jellemzők

- **created_at**: A hirdetés létrehozásának dátuma (2015-től kezdődően).
- **active_days**: A hirdetés időtartama napokban kifejezve.

Interakciós jellemzők

- **ad_view_cnt**: Az adott hirdetés megtekintéseinek száma. Nem minden esetben releváns, a kapcsolat az irányárral kérdéses.

Célváltozó

- **price_created_at**: A hirdetéshez tartozó irányár, millió forintban megadva. A feladat célja ennek az értéknek a megbecsülése a többi rendelkezésre álló változó alapján.

3. Adatelőkészítés és feltáró elemzés

3.1 Bevezetés az adatelőkészítéshez

A projekt során a kiindulási adatkészlet tartalmazott néhány hibás, hiányos, illetve nem informatív adatot, amelyek közvetlenül befolyásolhatták volna a modell pontosságát és érvényességét. Ebben a fejezetben bemutatjuk az egyes változókra vonatkozó adatfeldolgozási lépéseket, a tisztítás során alkalmazott megoldásokat, valamint a változók alapvető eloszlását és jellemzőit, megfigyeléseinket vizualizációkkal egészítjük ki.

3.2 Adattisztítási lépések

3.2.1 Nem informatív vagy redundáns oszlopok eltávolítása

A `county` és `property_type` mezők nem hordoznak információt, mivel minden esetben kizárólag Budapest, illetve „flat” értéket tartalmaznak, így ezek törlésre kerültek. Törlésre került a `label` és az `ad_view_cnt` oszlop is, mivel ezek várhatóan nem relevánsak az árazási predikció szempontjából.

3.2.2 Lokációs változók feldolgozása

Az adatkészlet két oszlopban tartalmaz lokációs információt:

- `city`: Budapest és római számmal jelölt kerület (például: Budapest XII.)
- `postcode`: irányítószám, amely bizonyos esetekben hiányzik.

Budapest kerületei egyértelműen meghatározhatók az irányítószámok alapján, mivel Budapesten minden irányítószám 1-essel kezdődik, és a következő két számjegy a kerület sorszámát jelöli. Például a 1126 irányítószám a XII. kerülethez tartozik.

A `city` oszlopból kinyerhető kerületinformáció és a `postcode` közötti kapcsolat vizsgálatakor az alábbi megállapításokra jutottunk:

- **Hiányzó `postcode`:**
Előfordul, hogy az irányítószám nem ismert, de a kerület (`city` oszlopból) meg van adva. Ilyenkor a kerület sorszáma rendelkezésre áll, így lehetőség nyílik a hiányzó irányítószámok mesterséges előállítására.
- **Hiányzó `city`:**
Nem fordult elő olyan eset, amikor az irányítószám meg lenne adva, de a kerület ne lenne ismert (`city` üres lenne).

- **Inkonzisztencia ellenőrzése:**

Azon rekordokat vizsgáltuk, ahol mindkét mező (city és postcode) kitöltött. Ezek összevetése során **211 olyan esetet találtunk**, ahol az irányítószám nem a megjelölt kerülethez tartozik. (Ez az adatmennyiség a teljes datasethez képest elenyésző, így akár törölni is lehetett volna ezeket a sorokat.) Az irányítószámok információtartalmát a kerület megnevezésénél erősebbnek tekintettük, hiszen a postai címzés során valószínűbb, hogy az irányítószámot adták meg helyesen. Ennek megfelelően az inkonzisztens esetekben a **kerület sorszámát az irányítószám alapján javítottuk**.

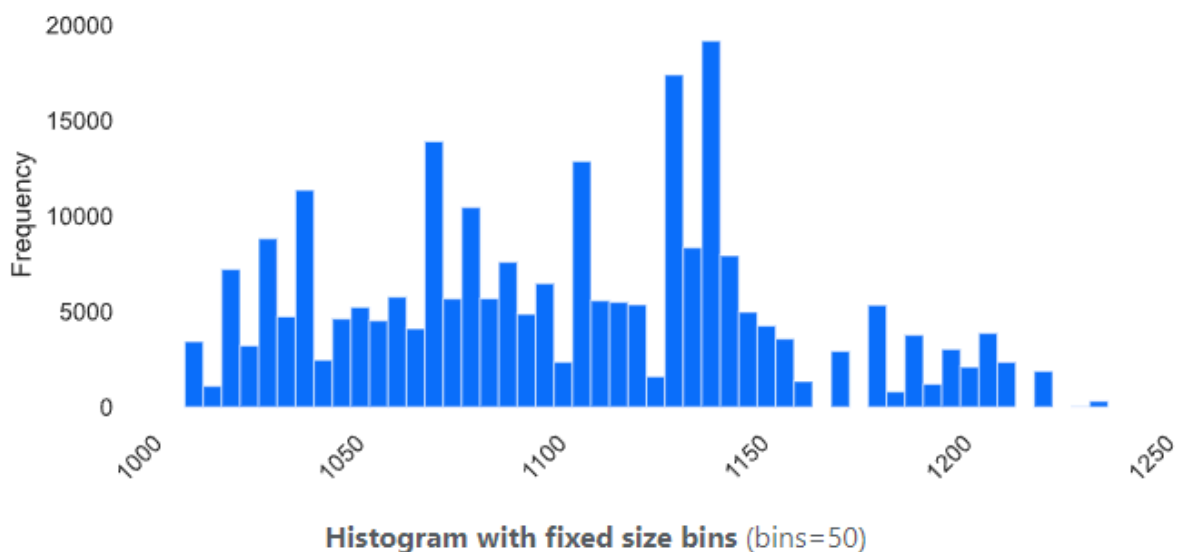
- **Hiányzó postcode pótlása:**

Azokban az esetekben, amikor az irányítószám hiányzott, de a kerület ismert volt, a hiányzó értékeket az alábbi logika szerint generáltuk:

- Az irányítószámok Budapesten **1-essel kezdődnek**.
- A következő két számjegy a kerület sorszáma, amit a **city** oszlop római számainak arab számmá alakításával nyertünk.
- Az utolsó (negyedik) számjegyet mesterségesen konstans **1-re állítottuk**. Ez az egyszerűsítés nem okoz jelentős torzítást, de pontosabb lokációs információt biztosít a modellezéshez, mint a hiányzó érték.

Kerületi eloszlás és reprezentativitás

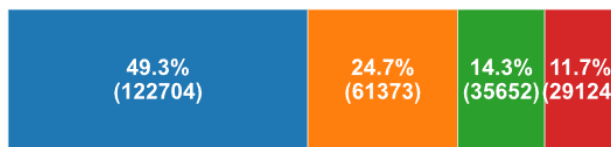
Az irányítószámok értékeinek hisztogramja alapján megállapítható, hogy bizonyos kerületek az adatkészletben **alulreprezentáltak**. Ez az eloszlás az aktuális kínálat függvényében változhat, de mivel a cél nem a teljes piac lefedése, hanem a meglévő mintán belüli predikció, így az eloszlás kiegyensúlyozatlansága a modellezést nem akadályozza. Érdeemes lesz viszont ezt figyelembe venni az értelmezés során.



3.2.3 Kategóriák aggregálása, módosítása

A `property_condition_type` kategóriáit 9-ről 4 csoportra aggregáltuk:

Category	Original Conditions	Group
Poor	`to_be_renovated`, `missing_info`	1
Fair	`medium`, `under_construction`	2
Good	`good`, `renewed`	3
Excellent	`can_move_in`, `new_construction`, `novel`	4



A csoportosítás után az eloszlás az ábrán látott módon alakul. Ez ugyan nem teljesen egyenletes, de elfogadhatónak minősül.



A `building_floor_count` oszlop az épület teljes magasságát, azaz az emeletek számát tartalmazza. Az értékek számozott formában, illetve szöveges kategóriaként szerepelnek az adatkészletben az alábbi módon:

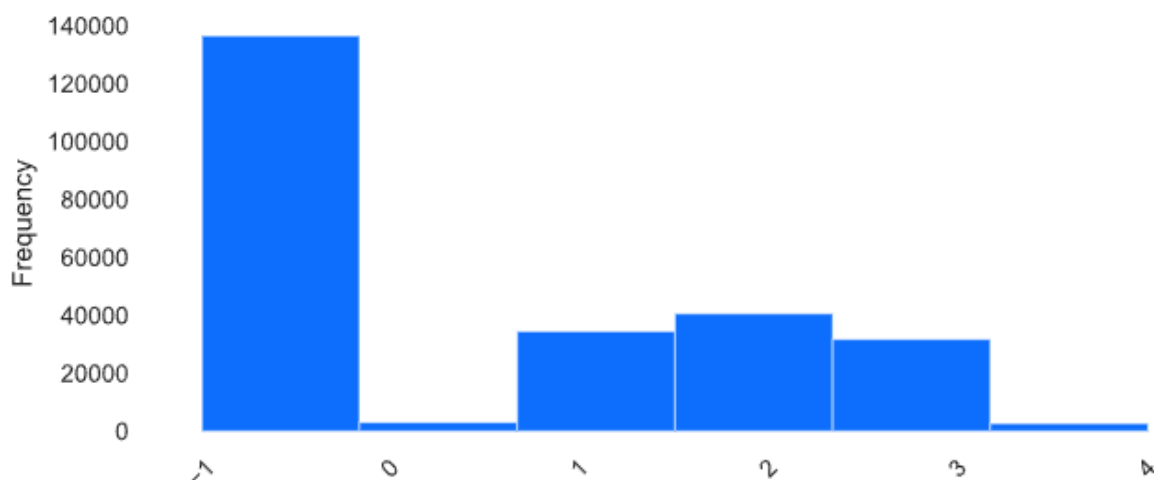
- numerikus értékek: '1', '2', ..., '10'
- szöveges kategória: 'more than 10'
- hiányzó értékek: `nan`

Összesen **11 különböző érték** található a mezőben, ugyanakkor jelentős a hiányzó adat: több mint **136 000 sorban** nem szerepel érték, ami az adatkészlet **55%-át** érinti. Ez az adatminőségi probléma ugyan csökkenti a változó megbízhatóságát, de figyelmen kívül hagyni nem indokolt, mert a rendelkezésre álló információk hasznosak lehetnek.

Az értékeket tovább kategorizáltuk, az alábbi csoportokat hoztuk létre:

Category	Original Values	Group
Single-Floor	1	0
Low-Rise	2, 3	1
Mid-Rise	4, 5, 6	2
High-Rise	7, 8, 9, 10	3
Very High-Rise	more than 10	4
Missing Data	nan	-1

A hiányzó adatokat külön kategóriában kezeltük (-1), így a modellezés során ez az információ is figyelembe vehető marad.



Histogram with fixed size bins (bins=6)

Az eloszlásból látszik továbbá, hogy az egyszintes és a 10+ szintes épületek viszonylag ritkák, míg a 2-10 szint közötti épületek egész közel esnek egymáshoz az új bontást gyakoriság szerint megvizsgálva. A -1 érték dominál, emiatt az adatminőség nem ideális.

A **property_floor** mező az adott ingatlan elhelyezkedését rögzíti az épületen belül. Az adatok között számok és szöveges kategóriák is szerepelnek:

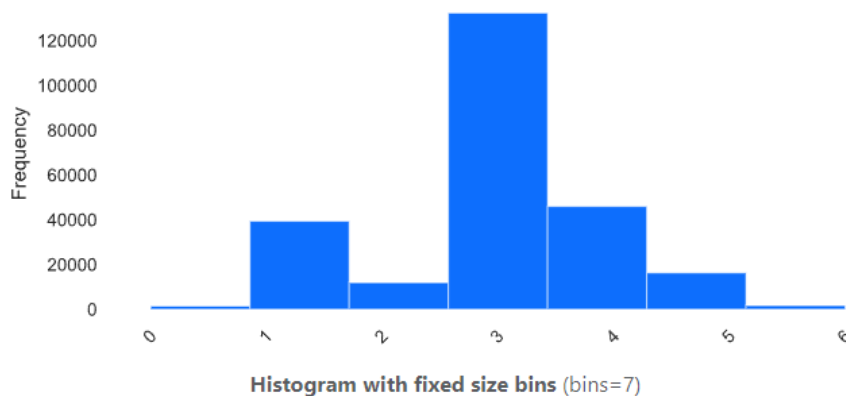
- numerikus értékek: '1'-'10'
- szöveges értékek: 'basement', 'ground floor', '10 plus', 'mezzanine floor'
- hiányzó értékek: **nan**

A hiányzó adatok aránya ebben a mezőben **kb. 5%**, amely lényegesen kisebb, mint a `building_floor_count` esetében. A hiányzó értékek statisztikai alapon kerültek pótlásra a rendelkezésre álló adatok eloszlása alapján, így nem került sor rekordok törlésére.

A különböző szöveges és numerikus értékeket egységes kategóriákba soroltuk az alábbiak szerint:

Category	Original Values	Group
Basement	basement	0
Ground Floor	ground floor	1
Mezzanine	mezzanine floor	2
Low Floor	1, 2, 3	3
Mid Floor	4, 5, 6, 7	4
High Floor	8, 9, 10	5
Very High Floor	10 plus	6

A kategorizáció során figyelembe vettük, hogy a különböző emeletek ingatlanpiaci értéke eltérő lehet (például magasabb emeleteken gyakran drágábbak a lakások panorama vagy zajterhelés szempontjából). Véleményünk szerint a kialakított kategóriákon belül hasonlóak lehetnek a megválasztott lakásárak.

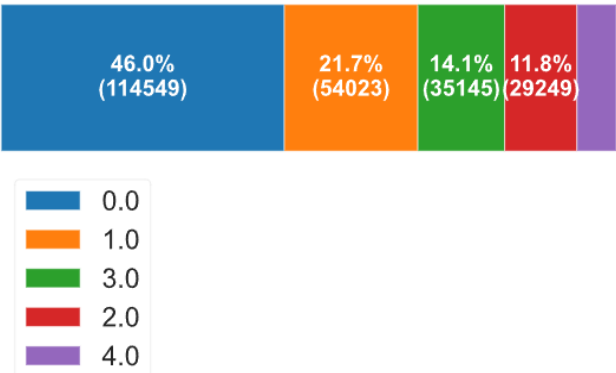


A kapott kategóriák a hisztogramon látható elosztást mutatják, az adatkészletben látszólag dominálnak az alacsonyabb, de emeleti (1/2/3 szinten lévő) lakások.

A **view_type** mező az ingatlan kilátását tartalmazza, amely esztétikai szempontból jelentős mértékben befolyásolhatja az árképzést. A következő kódolást alkalmaztuk:

Category	Original Values	Group
No View (Missing)	nan	0
Street View	street view	1
Courtyard View	courtyard view	2
Garden View	garden view	3
Panoramic View	panoramic	4

Az adathalmazban jelentős számú (**nan**) hiányzó érték szerepelt ebben az oszlopban. Ezek eltávolítása az adatkészlet nagyságának csökkenéséhez vezetett volna, amit nem tartottunk célszerűnek. Feltételeztük, hogy a hiányzó kilátásinformáció is hordozhat implicit információt (pl. átlagos kilátás, amelyet a hirdető nem emelt ki), ezért a hiányzó értékeket **0** értékkel kódoltuk.



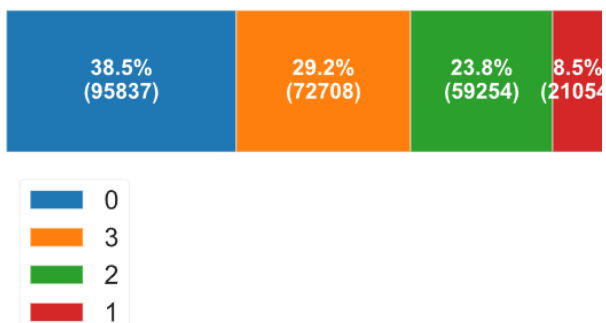
A vizualizáció alapján megfigyelhető, hogy a kitöltött adatok közül az utcára néző és belső udvari kilátás volt a leggyakoribb, míg a panorámás és kerti kilátás jóval ritkábban fordult elő, így várhatóan ezek kiemelten hozzájárulnak a magasabb árhoz a modellezés során.

Az **orientation** mező az ingatlan égtáji tájolását tartalmazza. Az adatokban nyolc különböző irány volt megadva (alapégtájak és köztes égtájak). Az adatok körülbelül 40%-a hiányzott ebből az oszlopból, ezért a hiányzó értékek kezelésére külön stratégiát kellett alkalmaznunk. A tájolás értékének piaci jelentősége Magyarországon eltér a melegebb éghajlatú országokban tapasztaltaktól: a napfényes, déli, dél-keleti fekvésű ingatlanok általában előnyösebbek, míg az északi fekvés kevésbé kívánatos. Ezt figyelembe véve a tájolási irányokat a következő csoportokba soroltuk:

Orientation	Group
North	1
North-East	1
North-West	1

East	2
West	2
South-East	3
South-West	3
South	3
Unknown	0

A **nan** értékeket **0** kódolással jelöltük meg, mivel feltételeztük, hogy az ilyen esetekben nem specifikált tájolás átlagos vagy kevésbé releváns az árazás szempontjából. Az ábra alapján megfigyelhető, hogy a csoportosított adatok között a napos fekvésű (csoport 3) ingatlanok dominálnak, ami várhatóan pozitív hatással van az ingatlanárakra.



A **heating_type** mező az ingatlan fűtési rendszerének típusát rögzíti. Az adatokban 11 különböző szöveges érték fordult elő. Az eredeti értékek eloszlásának főbb megfigyelései:

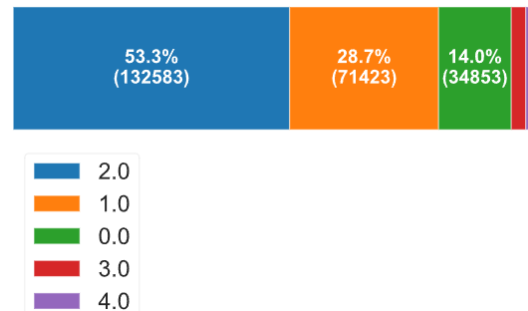
- A **konvection gas burner** típus tette ki az esetek 27%-át.
- A **gas furnace, circulating hot water** 19%-os arányban szerepelt.
- A **district heating** (távfűtés) az esetek 16%-át adta.
- A többi kategória együttesen az esetek 23%-át fedte le.

Az értékkészlet jelentős átfedéseket tartalmazott, például több különböző megnevezés is ugyanahhoz a technológiához tartozhatott. A kategóriák nem egységes definíciója miatt szükség volt az értékek csoportosítására. A fűtéstípusokat így kategorizáltuk:

Heating Type	Group
Unknown	0
Gas	1
Central	2
Electric	3
Other	4

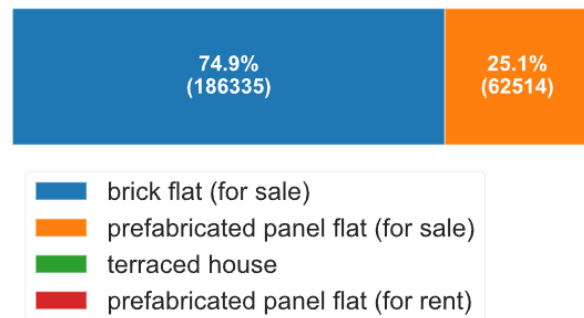
A hiányzó vagy nem értelmezhető fűtéstípusokat az **Unknown** kategóriába soroltuk.

Az ábra a fűtéstípusok gyakorisági eloszlását mutatja a kódolt kategóriák szerint. A központi fűtés alapú rendszerek dominanciája jól megfigyelhető, ugyanakkor a gáz fűtések is jelentős részt képviselnek az ingatlanállományban.



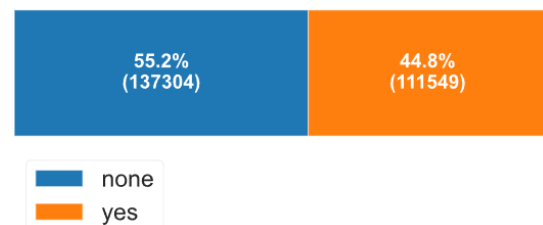
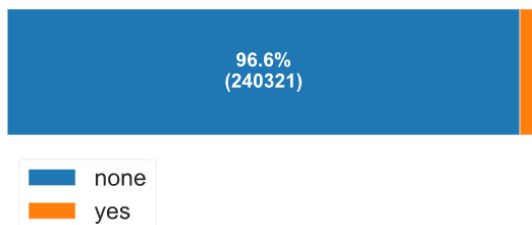
3.2.4 További hiányzó értékek

A `property_subtype` mező több kategóriát tartalmazott, de a hiányzó értékek (~2%) problémát jelentettek. Ezeket a leggyakoribb kategóriával pótoltuk (brick flat), mivel statisztikai alapon ez volt a legvalószínűbb választás.



Az ábrán látható hogy az eloszlás itt sem kedvező, az értékek háromnegyede egy kategóriába esik. A két meg nem jelenített érték előfordulása összesítve is 0,1% alatt van, ezek alapján nem várható, hogy lényegben hozzá fognak járulni a modellek teljesítményéhez.

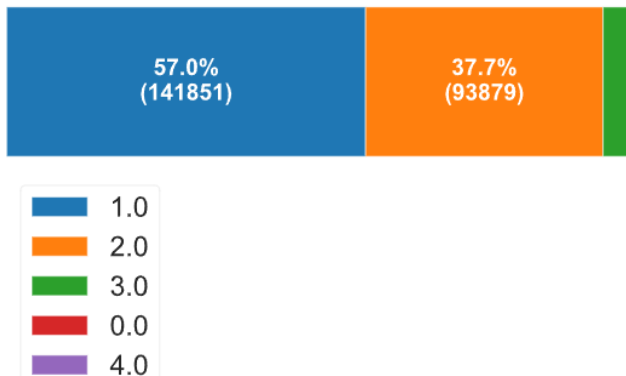
A `garden_access` és `elevator_type` oszlopok bináris információt hordoznak, jellemzően **igen/nem** típusú válaszlehetőségekkel. Az adatok között azonban számos esetben hiányzó (`null`) érték szerepel, ami kérdésessé teszi, hogy a hiány a tényleges állapotot tükrözi-e, vagy adatfelvételi hibából ered. Mindkét esetben feltételeztük, hogy amennyiben egy ingatlan rendelkezik az adott tulajdonsággal, azt a hirdetők jellemzően jelzik is, mivel ez piaci értéket növelő tulajdonság. A hiányzó értékeket ezért a „**none**” kategóriával töltöttük fel, azt feltételezve, hogy a kertkapcsolat vagy a lift hiányzik ezekben az esetekben. A bal oldal mutatja, hogy a városban nagyon kevés a kertkapcsolatos ingatlan, ugyanakkor a jobb oldal szerint felvonó az esetek közel felében rendelkezésre áll.



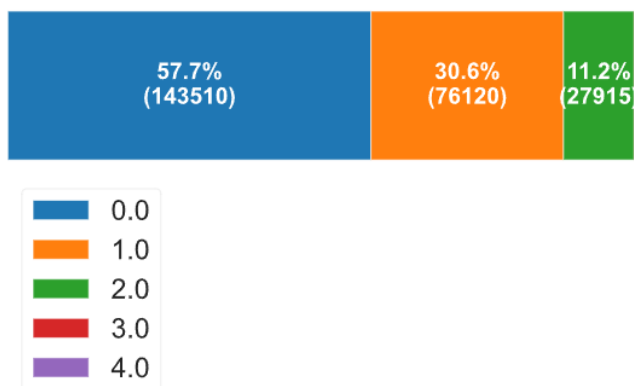
3.2.5 Hibás vagy szélsőséges értékek kezelése

A `room_cnt` és `small_room_cnt` esetében előfordult, hogy a mezőkben nem szobaszám, hanem (feltehetően) négyzetméter értékek szerepeltek. Ezeket az outliereket a következő vágási szabályok alapján szűrtük ki:

- `room_cnt < 5`
- `small_room_cnt < 5` és `small_room_cnt >= 0`



Az ábrán a room count értékei láthatóak. A legtöbb (57%) lakás csupán egyetlen szobával került meghirdetésre, majd a kétszobások (37,7%) dominálnak a második helyen. 3 szobával feladott hirdetések aránya 4,8%. Vannak olyan ingatlanok, melyeket csak félszobákkal hirdettek. Ezek aránya 0,4%. A 4 szobával hirdetettek pedig csupán 0,1%.



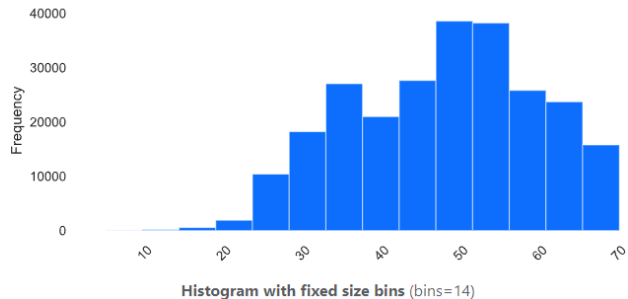
A félszobák tekintetben viszonylag magas azon ingatlanok száma, amelyekben nincs ilyen szoba (57,7%). 30,6 százalékukban egy, 11,2 százalékukban kettő, 0,5 százalékukban három, és kevesebb, mint 0,1 százalékukban található négy félszoba.

A `balcony_area` mező esetében az értékek már négyzetméterben vannak megadva, azonban az adatokban jelentős kilengések figyelhetők meg. A statisztikai elemzés során azt tapasztaltuk, hogy:

- Az ingatlanok **65,6%-ának nincs erkélye vagy terasza** (nulla érték).
- A **75. percentilis értéke mindössze 3 m²**, ami jellemző a budapesti ingatlanpiacon.
- Ennek ellenére előfordulnak irreálisan nagy értékek is, akár **1115 m²** nagyságrendben.

Az elemzés után a maximum értéket **100 m²-ben határoztuk meg**, mivel még nagyobb tetőteraszok vagy télikertek is ritkán haladják meg ezt a méretet. Az ennél nagyobb értékeket **levágtuk (capping)**, és 100 m²-re állítottuk be.

A **property_area** mező az ingatlan teljes hasznos alapterületét rögzíti négyzetméterben. Az adatok vizsgálata során azonban több olyan esetet találtunk, ahol az alapterület értéke **0 m²**, ami a valóságban értelmezhetetlen. A nullás területű rekordokat az adatkészletből **eltávolítottuk**, mivel ezek biztosan hibás adatfelvételre utalnak, és nem pótolhatók megbízhatóan. Az értékek itt már változatosabbak, 25 m² alatt viszont láthatóan kevés ingatlant tartalmaz a dataset.



A **price_created_at** oszlop tartalmazza az ingatlanhirdetésekben szereplő irányarat, millió forintban megadva. Ez a projekt regressziós feladatának célváltozója, amelynek predikciójára a modellezési folyamat irányul. A célváltozó alapstatisztikai jellemzőinek és eloszlásának elemzése után a főbb meglátások a következők:

- A legnagyobb megfigyelt érték közel **42 milliárd forint** (42 000 millió Ft), ami a budapesti lakáspiac realitását tekintve irreális.
- A **75. percentilis értéke 24 millió Ft**, vagyis az ingatlanhirdetések túlnyomó többsége ennél jóval alacsonyabb árkategóriába esik.
- Több rekord esetében **0,0 millió Ft** szerepel irányárként, ami szintén értelmezhetetlen és adatfelvételi hibára utal.

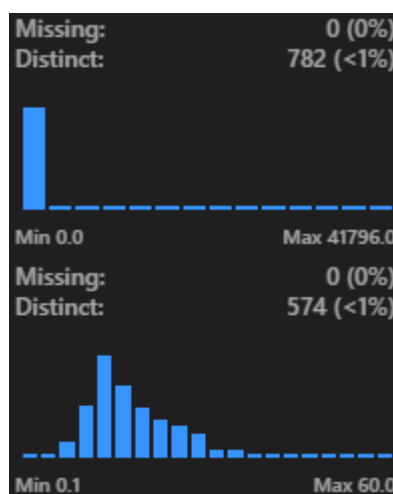
A célváltozó eloszlása így erősen torzult, és a szélsőséges, irreális értékek jelentősen befolyásolhatnák a regressziós modellek tanulását, növelve az outlierekhez való illeszkedési kényszert, rontva ezzel a becslés általánosíthatóságát. Ezen problémák orvoslására az alábbi tisztítási lépéseket alkalmaztuk:

1. **Nulla árú hirdetések törlése:** Azokat a rekordokat, amelyekben az irányár **0,0 millió Ft**, teljes egészében eltávolítottuk az adatkészletből, mivel ezek biztosan hibás vagy hiányos bejegyzések.
2. **Ármaximum korlátozása (capping):** A maximális árértéket **60 millió forintban** határoztuk meg. A 60 millió feletti értékeket 60 millióra vágtuk vissza. Ez a döntés az árak 75. percentilise (24 millió Ft) és az extrém maximum közötti jelentős eltérés, valamint az ingatlanpiac ésszerű ártartománya alapján született.

Ezzel a lépéssel csökkentettük a modellre nehezedő extrém outlierok hatását, így a becslés az árak szélesebb, de még reális tartományában optimalizálhatóvá vált.

Fontos megjegyezni, hogy az adatkészlet az **évekkel előtti hirdetési árakat** tartalmazza, és nem tükrözi a jelenlegi (2025-ös) ingatlanpiaci helyzetet. Az elmúlt években bekövetkezett jelentős infláció és áremelkedés miatt a mostani piaci maximum ennél jóval magasabb is lehetne. A modellezés ugyanakkor az adatok rögzítésének időpontjára vonatkozik, így az inflációval korrigált árbecslés nem célja ennek a projektnek.

- **Tisztítás előtt:** extrém nagy értékek, torz eloszlás.
- **Tisztítás után:** 0 értékek eltávolítva, 60 millió Ft felett capping alkalmazva.



3.2.6 Duplikált sorok vizsgálata

Az adatok között 153 olyan sor található, amelyek paramétereikben teljesen megegyeznek. Ezek ugyanakkor nem feltétlenül duplikált hirdetések, hanem jellemzően új társasházakban azonos típusú lakások (pl. Cordia, BigGeorge projektek).

3.2.7 Adattisztítás összefoglalása

- Eredeti rekordok száma: 262 235
- Tisztítás utáni rekordok száma: 248 245
- Adatvesztés aránya: **5,33%**

Az adattisztítás során a problémás, hiányos és irreális rekordok kiszűrésével biztosítottuk, hogy a modellezés során megbízható, konzisztens adatokat használjunk, ezzel javítva a becslés várható pontosságát.

3.2.8 További adatátalakítások a modellezés támogatásához

Az adattisztítási lépések után néhány további átalakítást végeztünk el annak érdekében, hogy az adatkészlet teljes mértékben alkalmas legyen gépi tanulási modellek betanítására:

- **Kerület (city) átalakítása numerikus értéké:**
A `city` mezőben szereplő római számos kerületjelzést (Budapest XII. stb.) arab számokká alakítottuk át (12). Ez lehetővé tette a numerikus feldolgozást.
- **Property subtype kategorizálása:**
A `property_subtype` szöveges változót szintén numerikus kódokra térképeztük fel az egyedi értékek sorszámozásával.
- **Bináris mezők átalakítása:**
A `garden_access` és az `elevator_type` mezők értékeit ('yes', 'none') 1-es és 0-ás értékekre alakítottuk, hogy numerikus bemenetként kezelhetők legyenek.
- **Dátum normalizálása:**
A `created_at` mező időbélyegét a minimális dátumhoz viszonyított napok számává konvertáltuk, így eltérő idősíki hirdetések összehasonlíthatóvá váltak.
- **Új feature képzése (meroszam):**
Létrehoztunk egy új változót, amely az adott hirdetés megtekintésszámát (`ad_view_cnt`) az aktív napok számával (`active_days`) arányosította, majd szorozta 100-zal. Ez egyfajta normalizált érdeklődési mutatóként funkcionál. A feature importance elemzések alapján később láthatjuk majd, hogy ez is hasznos érték a predikciónál, noha nem az ingatlan, hanem a hirdetés tulajdonsága.
- **Redundáns mezők eltávolítása:**
A `city`, `active_days` és `ad_view_cnt` oszlopokat eltávolítottuk, mivel az információjukat már más változók (pl. kerület szám, `meroszam`) képviselték, vagy közvetlenül nem járultak hozzá a predikcióhoz.
- **Duplikált rekordok kezelése:**
Az adatkészletből eltávolítottuk az esetleges teljesen azonos sorokat végül nem dobtuk el, a fentebb hivatkozott okok miatt.

3.3 Feltáró adatelemzés

Az adattisztítást követően egy részletes feltáró adatelemzést (EDA) végeztünk a `ydata-profiling` csomag segítségével. Ez lehetővé tette az adatminőség, az érték-eloszlások, az esetleges anomáliák és a változók közötti kapcsolatok gyors áttekintését.

3.3.1 Általános adatjellemzők

- **Rekordszám:** 248 245 hirdetés
- **Változók száma:** 19
- **Hiányzó értékek:** nincs (0%)
- **Duplikált sorok:** 153 db (0,1%), ami az új építésű társasházak hasonló lakásainak ismétlődése miatt nem feltétlenül hiba.
- **Változótípusok:** 10 kategóriális, 8 numerikus és 1 dátum típusú változó

A fenti adatok alapján az adatkészlet tiszta, jól strukturált, és alkalmas gépi tanulási feladatok ellátására.

3.3.2 Figyelmeztetések (Alerts)

- **Duplikált adatsorok**
- **Magas korreláció:**
 - `active_days` ↔ `ad_view_cnt`
 - `city` ↔ `postcode`
- **Imbalanced változók:**
 - `property_subtype`: 59,3%-ban „brick flat”
 - `garden_access`: 78,5%-ban nincs kertkapcsolat
- **Sok nullás érték:**
 - `balcony_area` (65,6% 0 érték)
 - `building_floor_count` (1,2% 0 érték)

Ezek az eredmények alátámasztják az adattisztítás során hozott döntéseket, mint például a `city` és `postcode` közötti redundancia kezelése, illetve az egyenlőtlen eloszlások megfelelő kezelése. A tisztítás után fennmaradó alertek elismertek, további beavatkozás nem igényelt.

3.3.3 Változók részletes vizsgálata

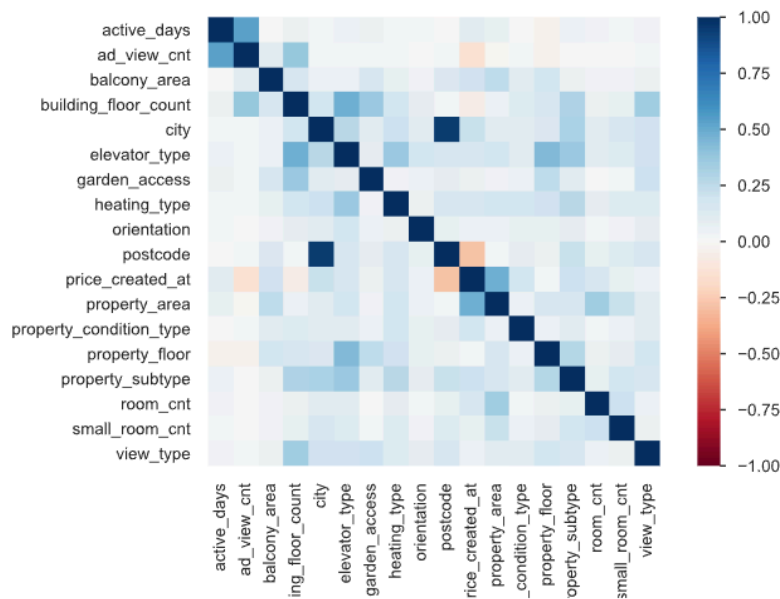
- **Lokációs adatok:**
 - A legtöbb hirdetés a XIV., XIII. és XI. kerületekben található.
 - Az irányítószámok eloszlása szintén ezt tükrözi, ahogy várnánk.
- **Ingtatlan jellemzők:**
 - A `property_subtype` mező erősen torzított: az ingatlanok nagy része téglalakás.
 - A `property_condition_type` mezőben a legtöbb lakás „good” állapotúnak van hirdetve.
- **Épület jellemzők:**
 - A `property_floor` értékek közül a földszinti és 1-3. emeleti ingatlanok dominálnak.
 - A `building_floor_count` esetében sok a hiányzó vagy nulla érték, de a legtöbb épület 2–6 emelet közötti.
- **Kilátás és tájolás:**
 - A `view_type` esetében az adatok kb. fele hiányzott vagy átlagos kilátást jelentett.
 - A `orientation` változóban a napfényes (déli) fekvés a leggyakoribb.
- **Fűtéstípus:** Leginkább központi fűtés és gázfűtés jellemző.
- **Erkély:** A hirdetések 65%-ában nincs terasz vagy erkély.
- **Szobaszám:** A legtöbb lakás egy vagy két szobás.

Ezek nem okoztak nagy meglepetést, a tisztítási folyamat során a legtöbbet már felismertük.

3.3.4 Célváltozó (`price_created_at`) elemzése

- A legtöbb ingatlan ára 10–30 millió Ft között helyezkedik el.
- A tisztítás után a maximális értéket 60 millió Ft-ban limitáltuk.
- Az ár eloszlása jobbra ferde, de a log-transzformáció nem volt szükséges, mivel a Gradient Boosting modellek ezt a torzulást jól tudják kezelni.

3.3.5 Változók közötti korrelációk



Erősebben pozitív korrelációk:

1. **active_days – ad_view_cnt**: Minél tovább aktív egy hirdetés, annál több megtekintést kap. Idő–érdeklődés kapcsolat.
2. **city – postcode**: A kerület és az irányítószám között szoros megfeleltetés.

Mérsékelt pozitív korrelációk:

3. **property_area – price_created_at**: Nagyobb alapterületű lakások általában magasabb árával rendelkeznek.
4. **room_cnt – property_area**: Több szobás lakások jellemzően nagyobb alapterülettel bírnak.

Közepes pozitív korrelációk:

5. **property_area – balcony_area**: A nagyobb lakásokhoz gyakrabban tartozik nagyobb erkély.
6. **elevator_type – building_floor_count**: Többszintes épületek gyakrabban rendelkeznek lifttel.
7. **property_floor – elevator_type**: A magasabb emeleteken található lakások gyakrabban liftes épületben helyezkednek el.

A korrelációs mátrixból jól látható, hogy a legtöbb változó csak gyenge-közepes mértékben függ össze egymással, így a multikollinearitás problémája nem jelentős.

Eredeti adathalmaz profilja:

Overview	Alerts 32	Reproduction
Dataset statistics		Variable types
Number of variables	23	Categorical 12
Number of observations	262,235	Numeric 10
Missing cells	859,483	DateTime 1
Missing cells (%)	14.3%	
Duplicate rows	0	
Duplicate rows (%)	0.0%	
Total size in memory	46.0 MiB	
Average record size in memory	184.0 B	

Tisztított adathalmaz profilja:

Overview	Alerts 9	Reproduction
Dataset statistics		Variable types
Number of variables	19	Categorical 10
Number of observations	248,245	Numeric 8
Missing cells	0	DateTime 1
Missing cells (%)	0.0%	
Duplicate rows	153	
Duplicate rows (%)	0.1%	
Total size in memory	36.0 MiB	
Average record size in memory	152.0 B	

Adattisztítás hatása az adatkészlet minőségére

A kiinduló adatkészlet 262 235 rekordot és 23 változót tartalmazott, 14,3%-os hiányzó érték aránnyal és 32 adatminőségi riasztással. Az adattisztítás során a következő fő lépések történtek:

- Hiányzó értékek kezelése, irreális értékek szűrése, capping alkalmazása.
- Redundáns és erősen korrelált változók eltávolítása.
- Kategóriák ésszerű összevonása, numerikus típusok egységesítése.

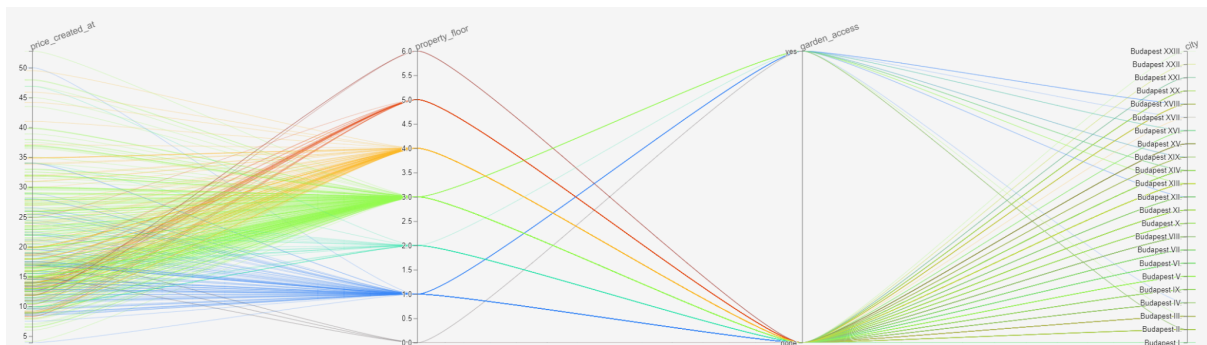
Az adattisztítás eredményeként:

- 248 245 rekord és 19 változó maradt meg,
- a hiányzó értékek teljesen megszűntek (0%),
- a duplikált rekordok száma 0,1%-ra csökkent és kezelve lett,
- a memóriahasználat optimalizálódott.

A profilozás alapján a 32 eredeti adatminőségi riasztásból mindössze 9 maradt fenn, amelyek a projekt szempontjából elfogadhatónak bizonyultak. Összességében az adatkészlet robusztusabbá vált, és megfelelő alapot biztosít a gépi tanulási modellek tanításához.

3.3.2 Változók közötti kapcsolatok vizsgálata – hiplot vizualizáció

A változók közötti összefüggések feltárására HiPlot párhuzamos koordináta-ábrát is alkalmaztunk, amely 500 véletlenszerűen kiválasztott rekordot ábrázolt. A vizualizáció alapján megfigyelhető, hogy a magasabb emeleti elhelyezkedés (`property_floor`) és a kertkapcsolat megléte (`garden_access` = yes) összességében magasabb irányárakkal (`price_created_at`) társul. A városrészek (`city`) szerinti eloszlás változatos képet mutat, nem kizárólag a prémium kerületek dominálnak a drágább szegmensben. Fontos megjegyezni, hogy a HiPlot elsősorban szemléltetési célt szolgál, a vizuálisan észlelt összefüggések statisztikai megerősítése további elemzést igényelne.



3.3.3 EDA - Konklúzió

A feltáró adatelemzés eredményei alapján megerősítést nyert, hogy **a tisztított adatkészlet jól strukturált és predikciós feladatokra alkalmas**. A **változók eloszlása összhangban van a budapesti ingatlanpiacról alkotott várakozásokkal**: a legtöbb hirdetés kis- és közép méretű lakásokról szól, a leggyakoribb fűtéstípus a központi és gázfűtés, a kertkapcsolat és a terasz ritkaságszámba megy.

A célváltozó (irányár) eloszlása jobbra ferde, de a tisztítás utáni formája kezelhető maradt, nem igényelt további transzformációt. A korrelációs vizsgálatok alapján **néhány változó között közepes kapcsolat figyelhető meg** (pl. alapterület és ár között), azonban **erős multikollinearitás nem alakult ki**, így a bemeneti változók használata biztonságosnak tekinthető.

Összességében az EDA megerősítette, hogy **az adatbázis alkalmas regressziós modellek tanítására**, és hogy **a tisztítás során végzett lépések lényegesen javították az adatok minőségét és felhasználhatóságát**.

4. Modellezési stratégia és megvalósítás

4.1 Modellválasztás és értékelési módszertan

Az ingatlanárak előrejelzésére regressziós megközelítést választottunk, mivel a probléma típusa numerikus célváltozóra történő predikció. A célváltozó a hirdetésekben megadott irányár (millió forintban), amelynek előrejelzéséhez olyan modelleket alkalmaztunk, amelyek képesek numerikus és kategóriális prediktorok együttes kezelésére.

Három különböző modell tanítását és összehasonlítását végeztük el:

- **Lineáris regresszió (LR):** egyszerű baseline modell, amely feltételezi, hogy a prediktorok és a célváltozó között lineáris kapcsolat van.
- **Gradient Boosting Regressor (GBM):** összetettebb, nemlineáris összefüggéseket is kezelni képes ensemble modell.
- **XGBoost Regressor (XGBM):** optimalizált gradient boosting implementáció, amely hatékonyan képes nagy adathalmazokon tanulni és regularizálási technikákat is alkalmaz.

Értékelési metrikák

A modellek teljesítményét egységesen az alábbi két regressziós metrikák segítségével értékeltük:

- **Mean Squared Error (MSE):** A négyzetes hibaátlag az előrejelzett és a tényleges értékek különbségének négyzetét átlagolja.
- **Mean Absolute Percentage Error (MAPE):** A százalékos abszolút hibaátlag a predikció hibáját relatív, százalékos formában méri.

A MAPE előnye, hogy könnyen értelmezhető: azt mutatja meg, hogy átlagosan hány százalékkal tér el a becslés a tényleges értéktől. Fontos megjegyezni, hogy a MAPE érzékeny a nagyon kicsi tényleges értékekre, többek között ezért is fontos volt az adattisztítás során a 0 irányárú hirdetések eltávolítása.

4.2 Alkalmazott modellek bemutatása

4.2.1 Lineáris regresszió (LR)

A lineáris regresszió egy alapvető prediktív modell, amely a bemeneti változók és a célváltozó közötti lineáris kapcsolatot feltételezi. A modell betanítása során nem alkalmaztunk hyperparaméter-hangolást, a standard implementációt használtuk.

- **Cél:** baseline eredmény biztosítása, a főbb prediktorok lineáris hatásának feltérképezése.
- **Értékelési eredmények:** MSE: **33,41**, MAPE: **24,24%**

A lineáris modell eredményei viszonylag gyenge illeszkedést mutattak, ami arra utal, hogy az ingatlanárak előrejelzéséhez nem elegendő a lineáris megközelítés, a piaci árképzés komplexebb mintázatokat követ.

4.2.2 Gradient Boosting Regressor (GBM)

A Gradient Boosting Regressor egy iteratív ensemble modell, amely gyenge tanulók (döntési fák) sorozatát építi fel, és minden lépésben az előző modellek hibáját próbálja csökkenteni. Ennek köszönhetően képes nemlineáris összefüggések és összetett mintázatok kezelésére.

- **Hiperparaméterek:**
 - `n_estimators = 300`
 - `learning_rate = 0.05`
 - `max_depth = 5`
 - `random_state = 39612`
- **Hiperparaméter-hangolás:**
A **GridSearchCV** segítségével kerestünk kedvező hiperparamétereket, 3-as keresztvalidációval, a MAPE minimalizálását célozva.
- **Értékelési eredmények:** MSE: **11,35**, MAPE: **11,71%**

A GBM lényegesen jobb predikciós teljesítményt nyújtott, mint a lineáris regresszió, különösen a nemlineáris összefüggések felismerésében és az outlierok kezelésében.

4.2.3 XGBoost Regressor (XGBM)

Az XGBoost egy optimalizált gradient boosting algoritmus, amely a GBM logikáját követi, de hatékonyabb tanulási algoritmussal és regularizációs technikákkal dolgozik. Ezáltal jobban képes elkerülni a túltanulást és gyorsabb is a tanítása.

- **Hiperparaméterek:**
 - `n_estimators = 650`
 - `learning_rate = 0.07`
 - `max_depth = 13`
 - `random_state = 39612`
- **Értékelési eredmények:** MSE: **6,82**, MAPE: **8,11%**

Az XGBoost adta a legjobb eredményeket a vizsgált modellek közül, jelentősen csökkentve az előrejelzési hibát.

4.3 Feature importance vizsgálat

A modellek betanítása után megvizsgáltuk, hogy az egyes bemeneti változók milyen mértékben járultak hozzá az árbecsléshez. A különböző modellek eltérő módon határozzák meg a változók fontosságát:

- A **lineáris regresszió** esetében a regressziós együtthatók abszolút értékeit és azok relatív súlyát vettük figyelembe.
- A **GBM** és az **XGBoost** modellek esetében a döntési fák osztási gyakorisága és a split gain alapján számított importance értékeket vizsgáltuk.

Legfontosabb prediktorok modelljeinkben

Mindhárom modellnél hasonló mintázat rajzolódott ki a változók fontosságát illetően, ugyanakkor kisebb eltérések is megfigyelhetők:

- A **property_subtype** (lakás típusa) minden modellben az egyik legfontosabb prediktor. A lineáris regresszió esetében különösen domináns, itt az összesített fontosság több mint 50%-át képviseli.
- A **property_area** (ingatlan alapterülete) szintén kiemelkedő jelentőségű mind a GBM, mind az XGBoost modellekben, az LR-ben viszont kisebb súllyal szerepel.
- A **postcode** (irányítószám, lokáció) a boosting alapú modellekben (GBM, XGBoost) az egyik legfontosabb prediktor, míg a lineáris regresszióban kevésbé domináns.

Ezek az eredmények alátámasztják, hogy az ingatlan típusának, alapterületének és lokációjának együttes figyelembevétele elengedhetetlen az irányárak pontos becsléséhez.

Eltérések a modellek között

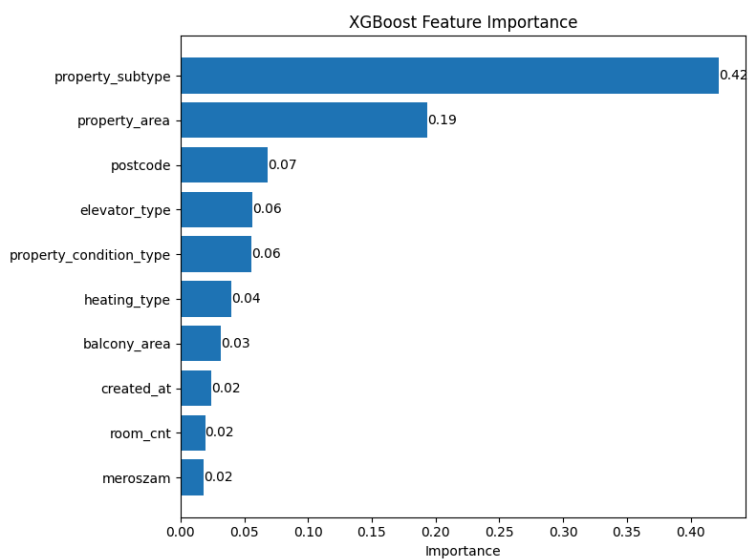
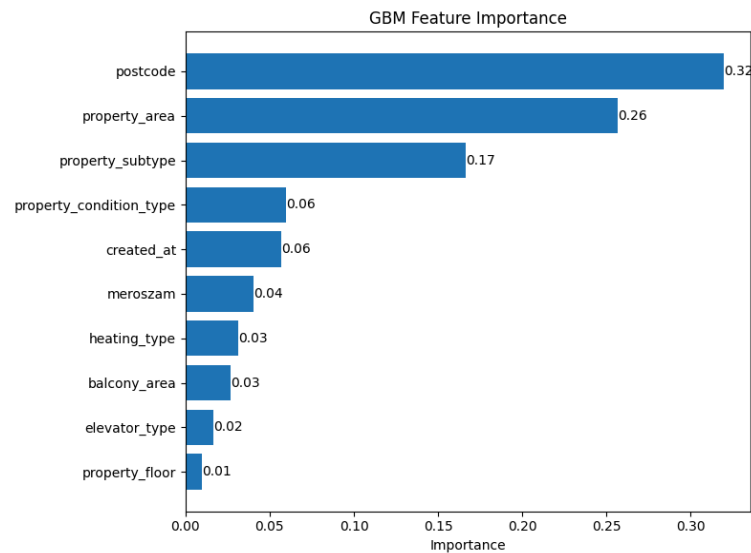
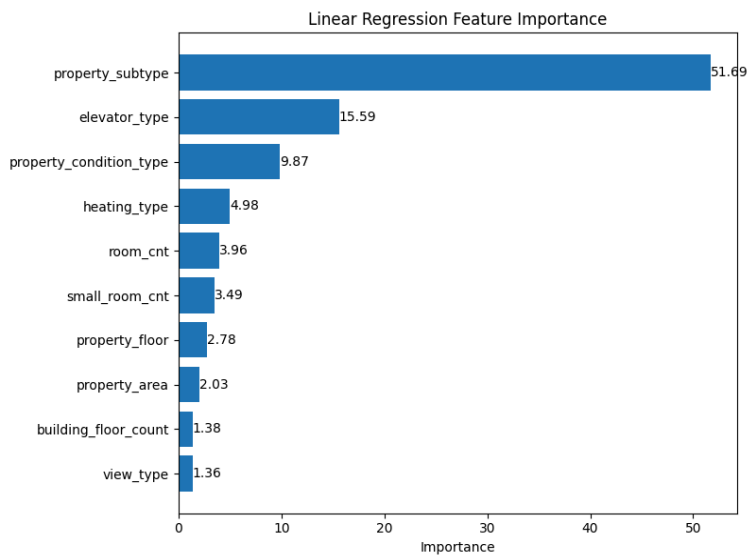
Érdekes megfigyelés, hogy míg a **lineáris regresszió** szinte kizárólag a **property_subtype**-ra támaszkodik, addig a boosting modellek egy kiegyensúlyozottabb fontossági eloszlást mutatnak a főbb prediktorok között. Ez is alátámasztja, hogy a komplexebb modellek jobban képesek kihasználni a több prediktor közötti nemlineáris kapcsolatrendszer.

A boosting modellekben továbbá olyan változók is jelentős szerepet kaptak, amelyek a lineáris modellben háttérbe szorultak, például:

- **created_at** (hirdetés létrehozási dátuma),
- **heating_type** (fűtés típusa),
- **balcony_area** (erkély mérete).

Ez azt sugallja, hogy a nemlineáris modellek érzékenyebbek az összetett mintázatokra, és több információt képesek kinyerni a prediktorok kombinációiból.

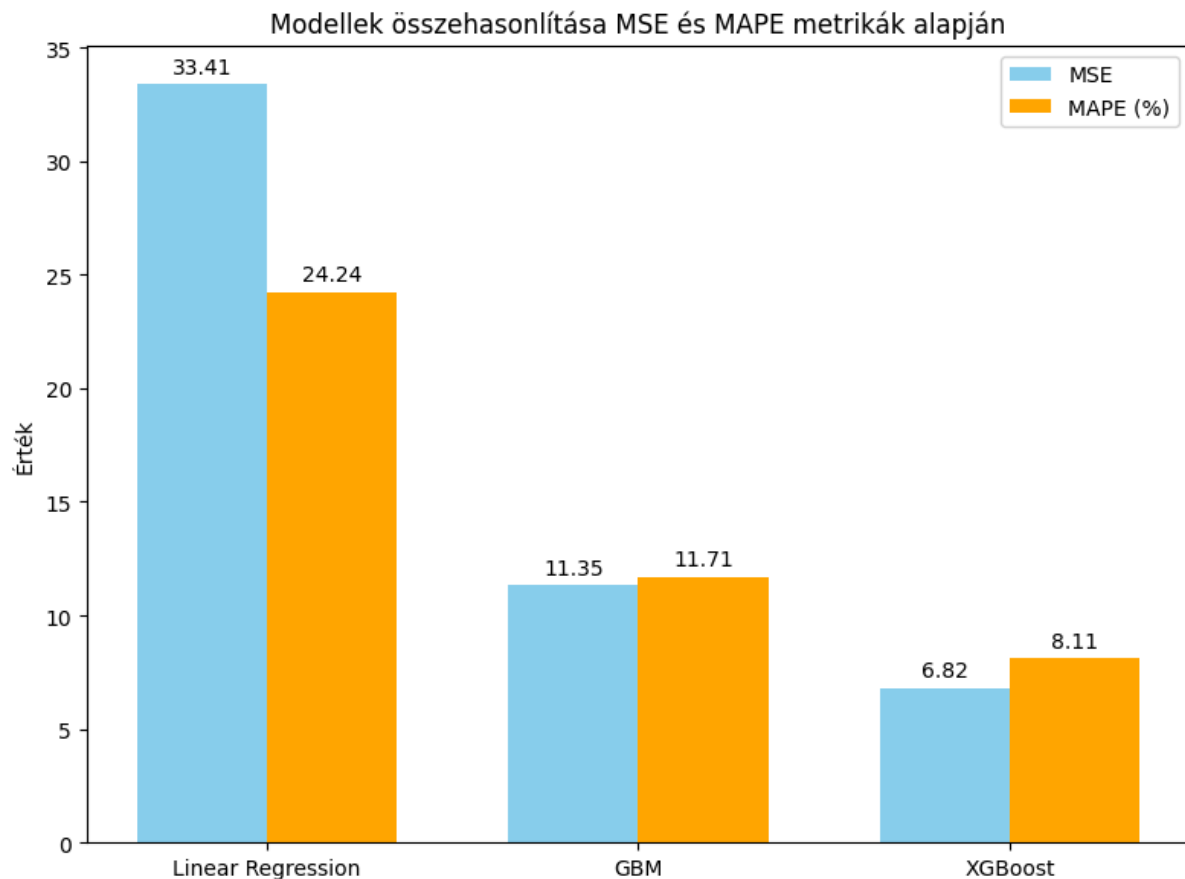
A top 10 legfontosabb prediktor fontossági értékeit a három vizsgált modell esetében:



4.4 Modellek összehasonlítása

A három vizsgált modell teljesítményét a tesztadatokon mért **MSE** és **MAPE** metrikák alapján hasonlítottuk össze. A következő táblázat a modellek eredményeit foglalja össze:

Modell	MSE	MAPE (%)	Megjegyzés
Lineáris regresszió	33,41	24,24	Baseline modell, lineáris kapcsolatot feltételez
GBM	11,35	11,71	Boosting alapú modell, nemlineáris mintázatokat is kezel
XGBoost	6,82	8,11	Optimalizált boosting implementáció, a legjobb eredményekkel



Eredmények értelmezése

A táblázat alapján egyértelműen látható, hogy a boosting alapú modellek lényegesen alacsonyabb hibát eredményeztek, mint a lineáris regresszió:

- A **lineáris regresszió** MAPE értéke több mint kétszerese a GBM-nek és közel háromszorosa az XGBoostnak.
- A **GBM** modell jelentős javulást mutatott a baseline-hoz képest, MAPE értéke 11,71%.
- Az **XGBoost** adta a legjobb eredményeket, a legalacsonyabb MSE (6,82) és MAPE (8,11%) értékekkel.

Ezek az eredmények alátámasztják, hogy a boosting típusú modellek jobban képesek megragadni az ingatlanárak előrejelzéséhez szükséges komplex összefüggéseket, szemben az egyszerű lineáris modellel.

Megfigyelt különbségek a modellek között

A lineáris regresszió esetében a predikciók viszonylag nagy hibával tértek el a tényleges értékektől, ami azt mutatja, hogy a bemeneti változók és az árak közötti kapcsolat nem írható le pusztán lineáris összefüggéssel.

Mind a GBM, mind az XGBoost modell esetében jelentős pontosságjavulást tapasztaltunk. A két boosting modell közül az **XGBoost** bizonyult a legpontosabbnak. A különbség az XGBoost magasabb komplexitású paraméterezéséből ($n_estimators=650$, $max_depth=13$) és optimalizált tanulási algoritmusából eredhet.

Vizsgálati körülmények

A modellek értékelése egységes tanulóteszt felosztással (70% tanító, 30% teszt) és rögzített random state mellett történt. A GBM esetében GridSearchCV-vel történő hyperparaméter-hangolást alkalmaztunk, míg a lineáris regresszió és az XGBoost esetében előre definiált paraméterekkel dolgoztunk. Az XGBoost modell tuningja nem tartalmazott külön keresztvalidációt, a paramétereket kézi beállítással választottuk.

5. Felhasználói felület, alkalmazás

5.1 Alkalmazás architektúra

A megoldás részeként egy egyszerű, böngészőből elérhető felhasználói felületet készítettünk, amely lehetővé teszi a lakásparaméterek megadását és az árbecslési modellek előrejelzéseinek lekérdezését. A rendszer mindhárom modell előrejelzését megjeleníti.

Frontend	Backend	ML Modellek
React (TypeScript)	Python (Flask) REST API	LR, GBM, XGBM pickle modellek
Material UI (MUI) elemek	/api/predict végpont a predikciókhoz	Adattisztítás a predikció előtt
Buildelt React statikus fájlok hosztolva Flask-ból		

5.2 Frontend működése

- A frontend React + TypeScript alapú, az `App.tsx` fájlban található meg.
- A felületet **Material UI** (MUI) elemekkel alakítottuk ki.
- A form kitöltése után a **"SUBMIT"** gombra kattintva indul el a predikció.
- Az eredményt egy modális ablakban (popup) jelenítjük meg.
- Az árak millió forintban (HUF) jelennek meg, két tizedesjegyre kerekítve.

Predict the Price of a Flat

District

Budapest I.

Postcode

Property Subtype

Condition

Floor

Total Floors in Building

View

Orientation

Garden Access

Heating Type

Elevator

Room Count

Small Room Count

Listing Date

dd/mm/yyyy

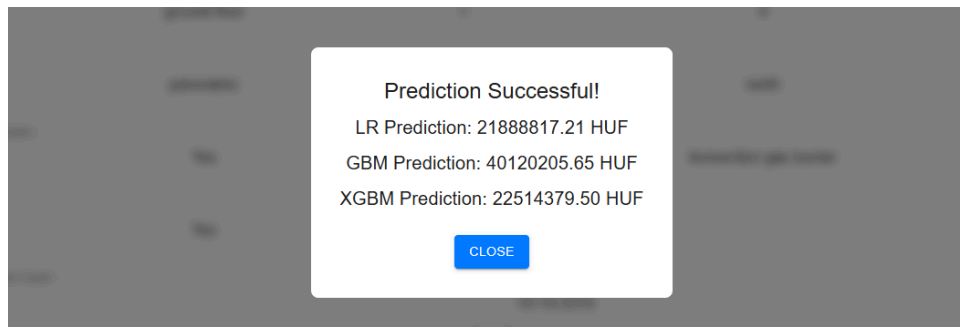
Property Area (m²)

Balcony Area (m²)

Ad View Count

SUBMIT

Példa válasz, a három modell változatos becsléseivel:



Hiba esetén az alábbi figyelmeztető üzenet jelenik meg a felületen:

Lovácsi Kristóf N3EEWB • Szladek Máté Nándor TGPZTT • Tóth Ádám László TK6NT3

❗ Error submitting the form. ✕

5.3 Backend működése

- A Flask backend szolgálja ki a buildelt frontend statikus fájljait, illetve az API-t.
- Az API végpontja: **/api/predict** (HTTP POST)
- A POST kérés body-jában az alábbi mezők megadása szükséges:
`required_fields = ["city", "postcode", "property_subtype", "property_condition_type", "property_floor", "building_floor_count", "view_type", "orientation", "garden_access", "heating_type", "elevator_type", "room_cnt", "small_room_cnt", "created_at", "property_area", "balcony_area", "ad_view_cnt"]`
- A backend a beérkező adatokat átalakítja, hogy azok a modellek által elvárt formátumban legyenek (egyszerűsített „Convert DF to ML data” logika).
- A modellkimenetek JSON válaszban érkeznek vissza:
`{ "predicted_price": 19.35, "predicted_price_gbm": 33.95, "predicted_price_xgbm": 24.42 }`

A visszakapott értékek millió forintban értendők, a frontend ezeket a felhasználó számára emészthető formátumra transzformálja és így jeleníti meg.

5.4 Fejlesztési eszközök és környezet

- **Frontend:** React + TypeScript + Material UI
- **Backend:** Python 3.x, Flask
- **Modellek:** Scikit-learn és XGBoost pickle (.pkl) fájlokban tárolt tanított modellek
- **Kommunikáció:** REST API JSON alapú adatátadással
- **Hosztolás:** Flask app szolgálja ki mind a statikus frontend fájlokat, mind az API-t

6. Komondor szuperszámítógép használata

6.1 Környezet kialakítása

A projekt részeként kipróbáltuk a magyarországi **Komondor** szuperszámítógép által biztosított HPC (High-Performance Computing) környezetet, noha a feladat számítási igénye ezt nem indokolta volna feltétlenül. A Komondor rendszeréhez a HPC@hu JupyterHub platformján keresztül csatlakoztunk.

- A munkakörnyezet konfigurálásához a **Simple** beállítást választottuk.
- A futtatási környezet paraméterei:
 - **Partition:** **cpu**
 - **CPU:** 8 mag (Minimum szintű erőforrás)
 - **Memória:** ~14 GB szabad RAM
 - **Job duration:** 1 óra
 - **Notebook környezet:** **Data Science**

Simple

Advanced

Partition Selection

CPU Partition
(HPE Cray EX425)
cpu

GPU Partition
(HPE Cray EX235n)
gpu

ML Partition
(HPE Apollo 6500)
ai

Big Data Partition
(HPE Superdome)
bigdata

CPU selection

Minimum
8 core

Quarter node
32 core

Half node
64 core

Whole node
128 core

Options

Jupyter notebook:

Data Science

Job duration:

1 hour

Available resources

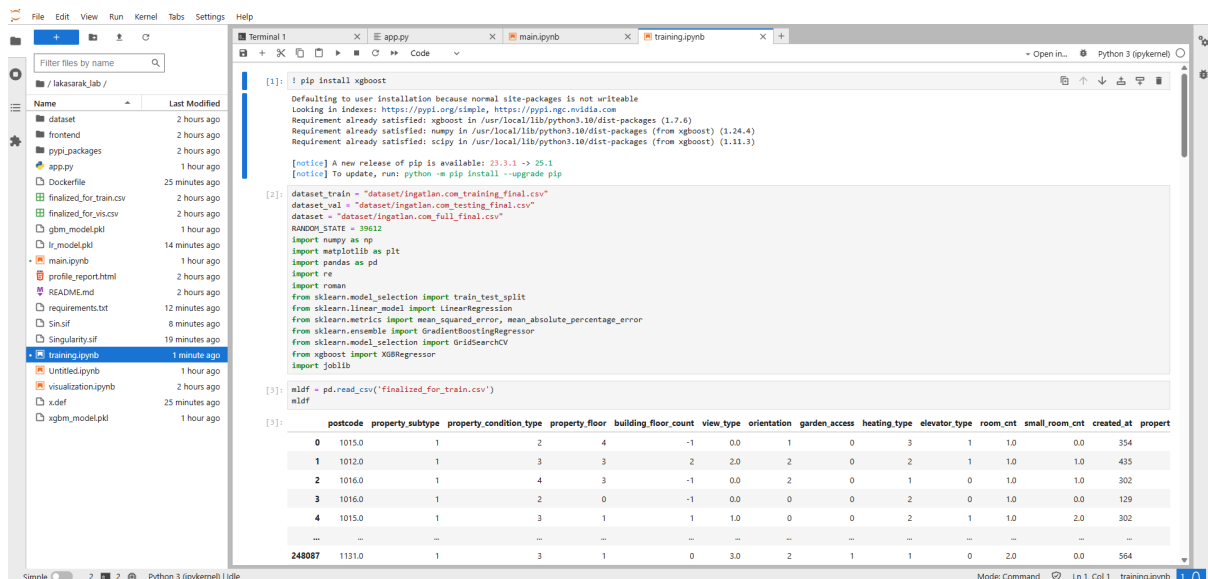
Partition	Total nodes	Free nodes	Total GPUs	Free GPUs	Total CPUs	Free CPUs	Total Mem GB	Free Mem GB
cpu	184	0	0	0	23552	1720	46000	14378
gpu	58	4	232	47	3712	1115	14500	10128
ai	4	0	32	8	512	192	2009	1384
bigdata	1	0	0	0	288	17	11904	804

Start

A beállítások alapján egy kisebb teljesítményű, CPU-alapú környezetet használtunk, amely megfelelő volt a notebookok futtatásához és az alkalmazás teszteléséhez.

6.2 Tapasztalatok a használat során

- A Jupyter notebookokat sikerült elindítani és futtatni a Komondoron (lenti kép).
- A **modellek tanítása** (Linear Regression, Gradient Boosting, XGBoost) problémamentesen végbement a notebookokban.
- A tanítási idő **körülbelül 11 perc** volt, ami a kisebb igényelt erőforrások miatt enyhén hosszabbnak bizonyult, mint az otthoni gépeken tapasztalt idő.
- A **frontendet is sikerült elindítani**, bár a környezet nem kínált közvetlen lehetőséget publikus elérésre.



```
[1]: ! pip install xgboost

Defaulting to user installation because normal site-packages is not writeable
Looking in indexes: https://pypi.org/simple, https://pypi.ngc.nvidia.com
Requirement already satisfied: xgboost in /usr/local/lib/python3.10/dist-packages (1.7.6)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from xgboost) (1.24.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from xgboost) (1.11.3)

[notice] A new release of pip is available: 23.3.1 -> 25.1
[notice] To update, run: python -m pip install --upgrade pip

[2]: dataset_train = "dataset/ingatlan.com_training_final.csv"
dataset_val = "dataset/ingatlan.com_testing_final.csv"
dataset = "dataset/ingatlan.com_full_final.csv"
RANDOM_STATE = 39612
import numpy as np
import matplotlib as plt
import pandas as pd
import re
import roman
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_percentage_error
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import GridSearchCV
from xgboost import XGBRegressor
import joblib

[3]: mldf = pd.read_csv('finalized_for_train.csv')
mldf

[3]:
```

	postcode	property_subtype	property_condition	property_floor	building_floor	count	view_type	orientation	garden_access	heating_type	elevator_type	room_cnt	small_room_cnt	created_at	property
0	1015.0	1	2	4	-1	0.0	1	1.0	0	3	1	1.0	0.0	354	
1	1012.0	1	3	3	2	2.0	2	0	2	1	1.0	1.0	1.0	435	
2	1016.0	1	4	3	-1	0.0	2	0	1	0	1.0	1.0	1.0	302	
3	1016.0	1	2	0	-1	0.0	0	0	2	0	1.0	0.0	0.0	129	
4	1015.0	1	3	1	1	1.0	0	0	2	1	1.0	2.0	2.0	302	
...
248087	1131.0	1	3	1	0	3.0	2	1	1	0	2.0	0.0	0.0	564	

6.3 Technikai kihívások és workaroudok

A rendszer használata során néhány kisebb nehézséggel szembesültünk:

- **Erőforrás allokációs problémák:** Előfordult, hogy nem állt rendelkezésre szabad node, így a futtatás nem mindig sikerült azonnal.
- **Port forwarding hiánya:** A Komondor jelenlegi rendszerében nem volt lehetőség saját szolgáltatás (pl. Flask app) közvetlen publikálására.
- **Workaround:** A frontend alkalmazást egy alternatív módszerrel, **ngrok** segítségével tettük ideiglenesen elérhetővé. Ehhez a Python környezetben lokálisan indítottuk a Flask appot, majd ngrok segítségével külső hozzáférést biztosítottunk egy ideiglenes URL-en keresztül.

6.4 Komondor - Összegzés

Összességében sikeresen kipróbáltuk a Komondor szuperszámítógép által biztosított HPC környezetet a projektünk során. Bár a rendszer nem kínált natív támogatást webalkalmazások hosztolására, alternatív megoldásokkal az alkalmazás funkcionálisan működőképes volt. A választott konfiguráció és a projekt jellege miatt a Komondor használata inkább érdekes próbalehetőség volt, mint praktikum. A tapasztalatok hasznosak voltak a nagy teljesítményű számítási rendszerek gyakorlati korlátainak és lehetőségeinek megismerésében is.

7. Összefoglalás

A projekt során célunk az volt, hogy budapesti **lakóingatlanok hirdetési adatai alapján megbecsüljük az ingatlanok irányárát**, valamint **elemezzük a lakásárakat befolyásoló tényezőket**. Az elvégzett feladatok a **teljes adatfeldolgozási és modellezési pipeline-t** lefedték, az adatelőkészítéstől kezdve az alkalmazásfejlesztésig.

Első lépésben az ingatlan.com adatbázisából származó **262 235 rekordot** tartalmazó adathalmazt dolgoztuk fel. Az adattisztítás során az **inkonzisztens, hibás és hiányzó adatok kezelésével** egy 248 245 rekordot tartalmazó, tisztított adatkészletet hoztunk létre, amely stabil alapot adott a modellezéshez. A változók kontextusának megértésére részletes **feltáró adatelemzést** végeztünk, majd megfelelő feature engineering lépések alkalmazásával javítottuk az adatok predikciós erejét.

A modellezés során három regressziós megközelítést alkalmaztunk: **Lineáris regresszió, Gradient Boosting, XGBoost**. A modellek összehasonlítása egységes értékelési metrikák (**MSE, MAPE**) alapján történt. Az eredmények szerint az **XGBoost** modell teljesített a legjobban (**MSE: 6,82; MAPE: 8,11%**), messze meghaladva a lineáris regressziós alapmodell teljesítményét. A **feature importance elemzések** alapján **az ingatlan típusa (property subtype), az alapterület (property area) és a lokáció (postcode) bizonyultak a legfontosabb ármeghatározó tényezőeknek**.

A projekt kimeneteként egy egyszerű **felhasználói felületet** is készítettünk, amely lehetővé teszi az előrejelzési modellek használatát valós adatok alapján. A **frontend React + TypeScript** alapokon készült, a **backend pedig Flask API** segítségével szolgálta ki a predikciókat.

A feladat részeként a magyarországi **Komondor szuperszámítógép** környezetében is futtattuk az alkalmazás egyes részeit, amely értékes gyakorlati tapasztalatot adott a HPC rendszerek működéséről és korlátairól.

Összességében elmondható, hogy **a kitűzött célokat teljesítettük**: a lakásárak becslésére alkalmas, több modellre épülő rendszert hoztunk létre, amely képes a hirdetési adatokból érdemi előrejelzéseket adni.

További fejlesztési lehetőségek:

- Haladó feature engineering (pl. új kombinált változók képzése, időbeli trendek kezelése)
- Még fejlettebb modellek alkalmazása (pl. LightGBM, CatBoost)
- A felhasználói felület továbbfejlesztése (pl. vizualizációk megjelenítése az eredmények mellett)
- Több adatforrás bevonása (pl. piaci átlagárak, makrogazdasági adatok) a modellek finomhangolásához.