



بررسی داده های مربوط به آگهی های استخراج شده از املاک کشور آلمان

پیش پردازش داده ها :

ابتدا نیاز است تا داده ها را به طور مناسبی **clean** کنیم. با توجه به دیتاست و اطلاعاتی که داریم، ستون های زیادی دارای اطلاعات ناقص و پوچ هستند. برای آنکه این نواقص را برطرف کنیم، در چند مرحله برای **type** ها متفاوت به صورت مختلف عمل کردیم.

در مرحله اول داده هایی را که بیش از 50% آنها **null** هست را از کل دیتاست حذف کردیم. 7 ستون بدین ترتیب حذف شده‌اند.

در مرحله بعد، برخی از داده هایی که مقدار متراژ خانه و همچنین **totalRent** آنها برابر با 0 بود را نیز حذف کردیم. با اینکار چند سطر از دیتاست حذف شد.

سپس 7 ستون دیگر را به علت آنکه تاثیری بر روی **totalRent** نداشتند نیز حذف کردیم.

حال لازم است مقادیر **Null** را پر کنیم. ابتدا داده های عددی را در نظر گرفته و تمامی مقادیر خالی آنها را با میانگین بقیه داده های موجود پر کردیم.

سپس برخی داده های پرت (**outlier**) را از دیتاست خارج میکنیم. منظور از داده های پرت در اینجا داده هایی است که در نمودار توزیع نرمال، در خارج از آن 90٪ اصلی قرار میگیرند. این داده ها باعث میشوند تا مدل ما برای فیت کردن آنها به خطا رود و بر روی بقیه داده ها نیز نتیجه خوبی نخواهد داد پس بهتر است که آنها را حذف کنیم .

حال نوبت آن است که داده های **categorical** را بررسی کنیم. این داده ها نیز همانند داده های عددی دارای دیتا های پوچ هستند. برای پر کردن این داده ها از **mode** یعنی دادهای که بیشترین تکرار را دارد

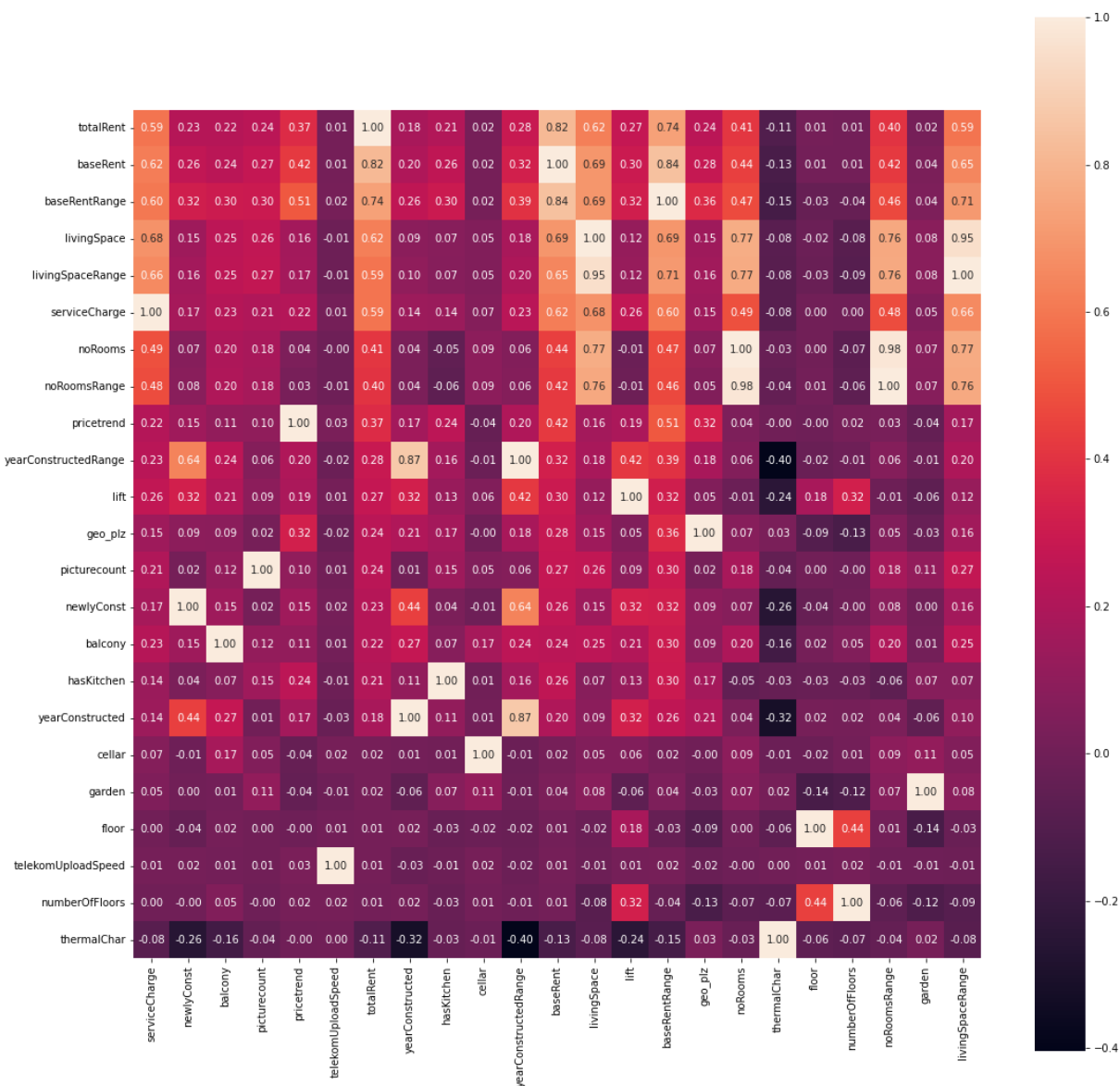
استفاده میکنیم . تعداد داده های موجود در هر ستون را در نظر گرفتیم و بیشترین آن را به جای داده های پوچ قرار دادیم .

پس از این کار موارد مشابهت را نیز بررسی کردیم. 1597 مورد مشابهت وجود داشت که آنها را از جدول حذف میکنیم.

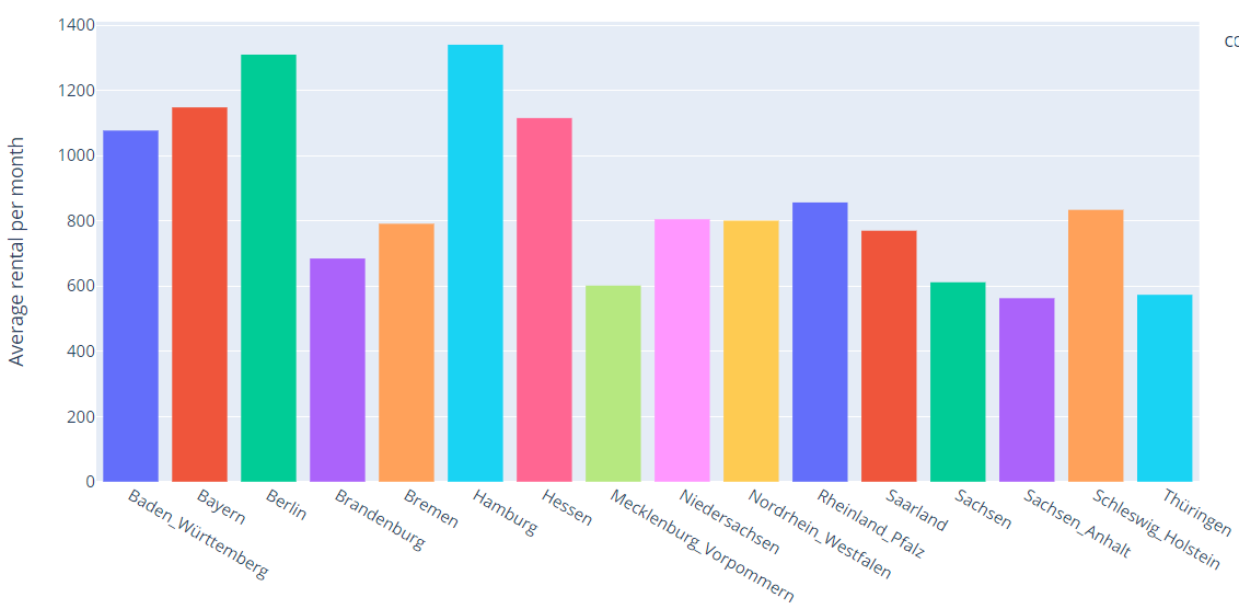
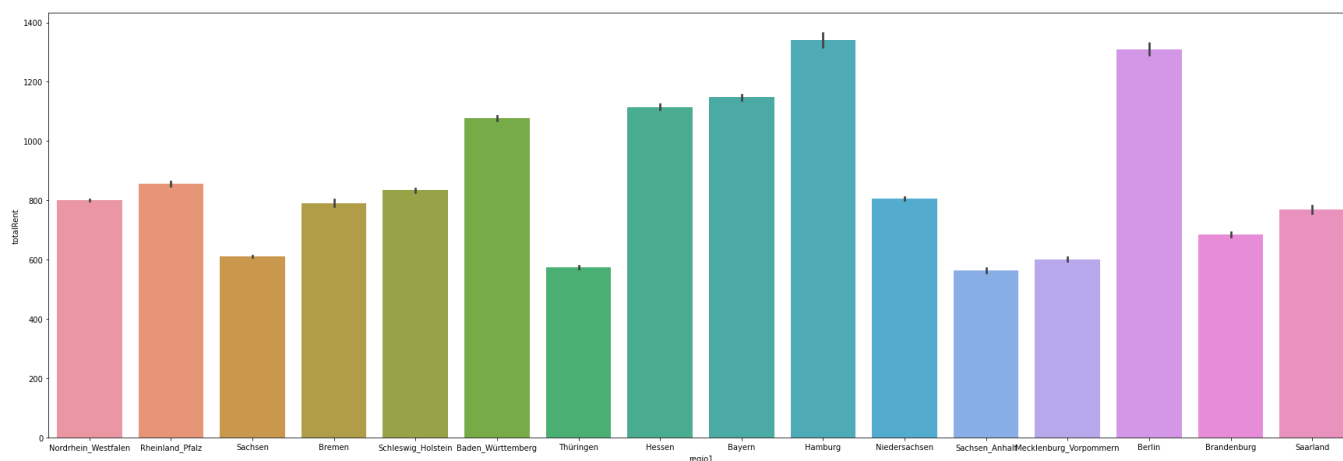
پس از این کار دیگر داده‌ی پوچی در جدول ما وجود نخواهد داشت.

مصور سازی داده ها:

ابتدا ماتریس کورولیشین و ارتباط فیچر ها به یکدیگر را در زیر میبینیم:



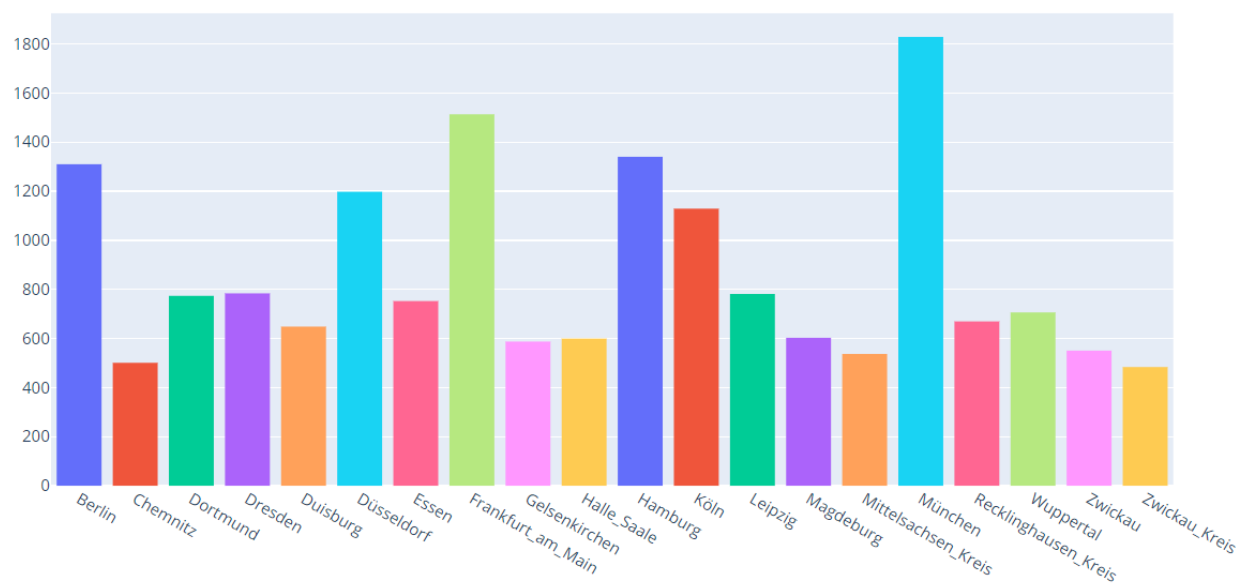
در نمودار های زیر قیمت totalRent خانه ها در regio1,2,3 میبینیم:
برای منطقه ۱:



همانطور که در هر دو نمودار مشخص است قیمت خانه در این ناحیه در شهر Hamburg بیشتر از مابقی مناطق است.

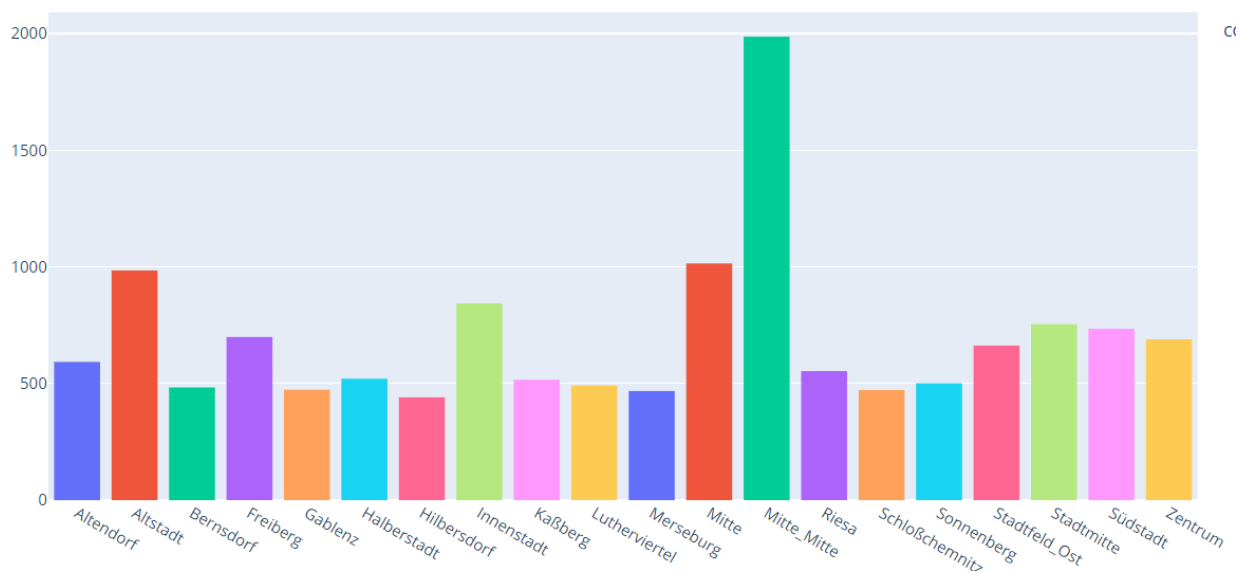
*** در مورد مناطق ۲ و ۳ چون تعداد شهر های یونیک آنها بسیار زیاد بود، ۲۰ شهر اولی که بیشترین تکرار را داشتند در نظر گرفتیم

برای منطقه ۲:



در این منطقه شهر munchen بیشترین totalRent را دارد.

برای منطقه ۳:



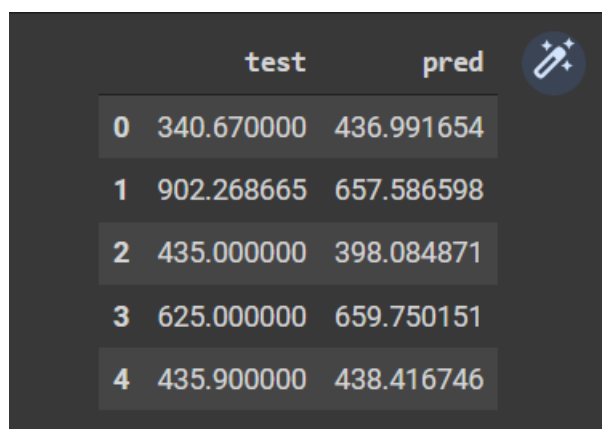
در این منطقه شهر Mitte_Mitte بیشترین قیمت را دارد.

مدلسازی قیمت

برای مدل سازی نیاز است که داده را طوری آماده کنیم تا بتوانیم به مدل مورد نظر بدهیم. داده های categorical زیاد هستند و لازم است که تمامی آنها را طوری encode کنیم که برای مدل قابل تفسیر باشد. برای encode کردن می‌خواهیم از one hot encoding استفاده کنیم و برای آنکه بعد داده خیلی زیاد نشود نیاز است تا فیچر هایی که تعداد داده های unique آنها زیاد است را حذف کنیم. دو ستون regio2 و regio3 را حذف نمودیم و باقی داده های categorical را one hot میکنیم. حال تمامی داده های ما عددی هستند و مدل آنها را میتواند بررسی کند. مدل رگرسیون را بر روی داده های خود امتحان میکنیم

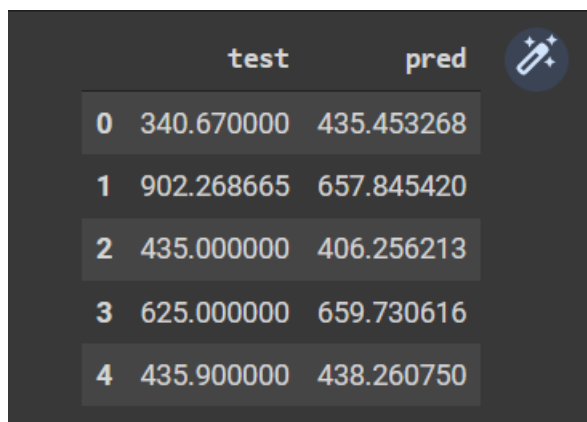
در ابتدا داده ها را به دو قسمت train و test تقسیم کردیم به طوریکه 80٪ را به عنوان train و بقیه را test در نظر گرفتیم. حال برای ساختن مدل رگرسیون باید به روند کلیه کار توجه داشته باشیم. در اینجا مقدار Loss را با استفاده از MSE محاسبه میکنیم. MSE در واقع همان میانگین خطای ما است.

مدل regression را با استفاده از sklearn پیاده سازی کردیم، مدل را بر روی داده های train آموزش میدهم و سپس پیشبینی آن را در مقایسه با داده های test میبینیم.



	test	pred
0	340.670000	436.991654
1	902.268665	657.586598
2	435.000000	398.084871
3	625.000000	659.750151
4	435.900000	438.416746

مدل دیگر ridge regression است که آن را نیز پیاده سازی کردیم، مقدار خطای آن کمتر از regression معمولی شد. نتایج حاصل:



	test	pred
0	340.670000	435.453268
1	902.268665	657.845420
2	435.000000	406.256213
3	625.000000	659.730616
4	435.900000	438.260750

استفاده از بحث multiprocessing در پاکسازی داده ها

بحث موازی سازی را در مواردی که فرایندهای در حال انجام ارتباطی با یکدیگر نداشته باشند، میتوان استفاده کرد. در بخش پاکسازی داده نیز در مرحله حذف outlier ها میتوانیم به جای آنکه کل دیتاست را با یک core بگردیم تا به نتیجه برسیم، میتوانیم از 2 cpu استفاده کنیم تا سریع تر کار انجام شود.

تابع `my_func` همان تابع انجام عملیات حذف داده های پرت است. یک بار این تابع را بدون بحث Multiprocessing اجرا کردیم و زمان اجرا حدود 5 ثانیه شد. بار دیگر با استفاده از ایجاد `pool` و استفاده از 2 cpu برای انجام اینکار میبینیم که زمان اجرا 0.9 ثانیه میشود و این یعنی با استفاده از این بحث میتوانیم فرایندها را سرعت دهیم و سریع تر به نتیجه برسیم. به طور خاص در دیتاست هایی که حجیم هستند نیز بسیار پرکاربرد است.