

تاریخ:۱۸/۱/۱۸

استاد درس: دكتر فراهاني

درس: داده کاوی

مهرانه مقتدائی فر - ۹۷۲۲۲۰۸۶

گزارش تمرین سری ۱ - بخش دوم

## 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 کر دیم. با اینکار چند سطر از دیتاست حذف شد.

سپس 7 ستون دیگر را به علت آنکه تاثیری برروی 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 موچود 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 میگریند. این داده ها باعث میشوند تا مدل ما برای فیت کردن آنها به خطا رود و برروی 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.

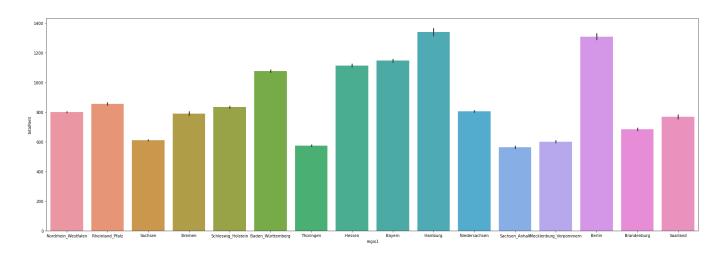
مصور سازی داده ها:

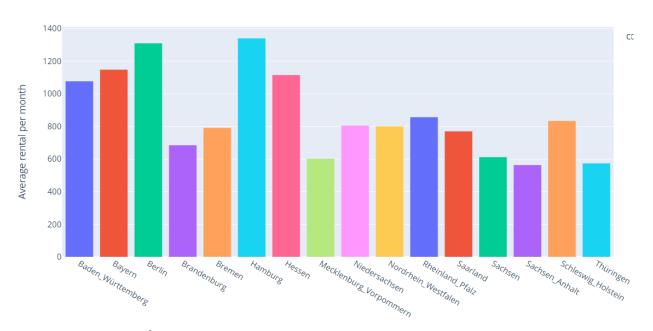
- 1.0

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

totalRent -	0.59	0.23	0.22	0.24	0.37	0.01	1.00	0.18	0.21	0.02	0.28	0.82	0.62	0.27	0.74	0.24	0.41	-0.11	0.01	0.01	0.40	0.02	0.59	
baseRent -	0.62	0.26	0.24	0.27	0.42	0.01	0.82	0.20	0.26	0.02	0.32	1.00	0.69	0.30	0.84	0.28	0.44	-0.13	0.01	0.01	0.42	0.04	0.65	
baseRentRange -	0.60	0.32	0.30	0.30	0.51	0.02	0.74	0.26	0.30	0.02	0.39	0.84	0.69	0.32	1.00	0.36	0.47	-0.15	-0.03	-0.04	0.46	0.04	0.71	
livingSpace -	0.68	0.15	0.25	0.26	0.16	-0.01		0.09	0.07	0.05	0.18	0.69	1.00	0.12	0.69	0.15	0.77	-0.08	-0.02	-0.08	0.76	0.08	0.95	
livingSpaceRange	0.66	0.16	0.25	0.27	0.17	-0.01		0.10	0.07	0.05	0.20		0.95	0.12	0.71	0.16	0.77	-0.08	-0.03	-0.09	0.76	0.08	1.00	
serviceCharge	1.00	0.17	0.23	0.21	0.22	0.01		0.14	0.14	0.07	0.23		0.68	0.26		0.15	0.49	-0.08	0.00	0.00	0.48	0.05	0.66	
noRooms -	0.49	0.07	0.20	0.18	0.04	-0.00	0.41	0.04	-0.05	0.09	0.06		0.77	-0.01		0.07	1.00	-0.03	0.00	-0.07	0.98	0.07	0.77	
noRoomsRange	0.48	0.08	0.20	0.18	0.03	-0.01	0.40	0.04	-0.06	0.09	0.06		0.76	-0.01		0.05	0.98	-0.04	0.01	-0.06	1.00	0.07	0.76	
pricetrend -	0.22	0.15	0.11		1.00		0.37	0.17		-0.04	0.20		0.16	0.19		0.32	0.04	-0.00	-0.00	0.02	0.03	-0.04	0.17	
yearConstructedRange	0.23	0.64	0.24	0.06		-0.02					1.00			0.42		0.18	0.06	-0.40	-0.02	-0.01	0.06	-0.01	0.20	
lift ·		0.32									0.42							-0.24			-0.01			
geo_plz ·		0.09	0.09			-0.02					0.18							0.03				-0.03		
picturecount -		0.02	0.12	1.00		0.01			0.15		0.06		0.26					-0.04		-0.00		0.11	0.27	
newlyConst -		1.00		0.02							0.64									-0.00			0.16	
balcony -			1.00			0.01		0.27	0.07		0.24							-0.16		0.05		0.01	0.25	
hasKitchen		0.04									0.16												0.07	
yearConstructed -		0.44	0.27			-0.03						0.20		0.32				-0.32		0.02	0.04	-0.06		
cellar - garden -		-0.01	0.17		-0.04			0.01		1.00	-0.01 -0.01							-0.01			0.09	0.11	0.05	
garden - floor -		-0.04	0.01		-0.04						-0.02							-0.06		0.44			-0.03	
telekomUploadSpeed		0.02	0.02		0.03	1.00					-0.02					-0.02		0.00				-0.01		
numberOfFloors		-0.00		-0.00							-0.01									1.00		-0.12		
thermalChar											-0.40										-0.04		-0.08	
are marenur	-	-	-	,	,	-	_	_			-	-	,	±	-,-		-		-,-					
	serviceCharge	newlyConst	balcony	picturecount	pricetrend	relekomUploadSpeed	totalRent	yearConstructed	hasKitchen	cellar	arConstructedRange	baseRent	livingSpace	=	baseRentRange	zld_oag	noRooms	thermalChar	floor	numberOfFloors	noRoomsRange	garden	livingSpaceRange	

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

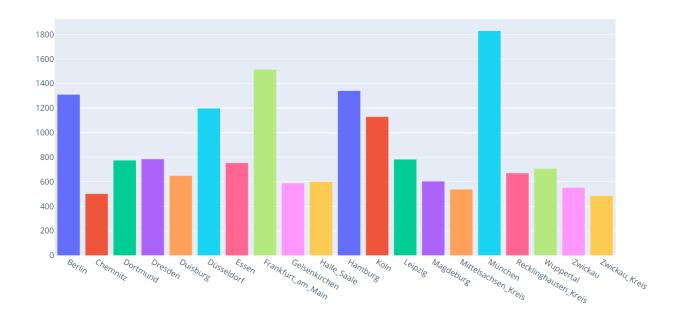




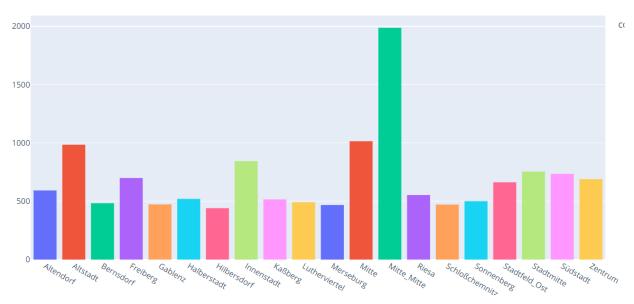
همانطور که در هر دو نمودار مشخص است قیمت خانه در این ناحیه در شهر Hamburg بیشتر از مابقی مناطق است.

\*\* 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

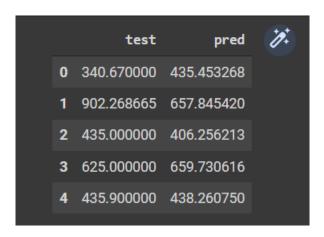
برای مدل سازی نیاز است که داده را طوری آماده کنیم تا بتوانیم به مدل مورد نظر بدهیم. داده های categorical زیاد هستند و لازم است که تمامی آنها را طوری encode کنیم که برای مدل قابل تفسیر باشد. برای encode کردن میخواهیم از one hot encoding استفاده کنیم و برای آنکه بعد داده خیلی زیاد نشود نیاز است تا فیچر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

در ابتدا داده ها را به دو قسمت train و train تقسیم کردیم به طوریکه 80٪ را به عنوان train و بقیه را Loss در ابتدا داده ها را به دو قسمت MSE و بقیه را Loss در نظر گرفتیم. حال برای ساختن مدل رگرسیون باید به روند کلیه کار توجه داشته باشیم . در اینجا مقدار MSE 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 برروی داده های train آموزش میدهیم و سپس پیشبینی آن را در مقایسه با داده های test میبینیم.

	test	pred	<b>7</b> .
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:



## استفاده از بحث multiprocessing در پاکسازی داده ها

بحث موازی سازی را در مواردی که فرایند های درحال 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 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 my\_func و استفاده از cpu 2 اجرا کردیم و زمان اجرا حدود 5 ثانیه شد. بار دیگر با استفاده از ایجاد Multiprocessing و استفاده از این بحث میتوانیم فرایند ها را سرعت برای انجام اینکار میبینیم که زمان اجرا 0.9 ثانیه میشود و این یعنی با استفاده از این بحث میتوانیم فرایند ها را سرعت دهیم و سریع تر به نتیجه برسیم. به طور خاص در دیتاست هایی که حجیم هستند نیز بسیار پرکاربرد است.