آتوسا مالمیر چگینی

۹۷۱۰۶۲۵۱

گزارش فاز اول پروژه‌ی درس یادگیری ماشین

* در ابتدا باید این نکته ذکر شود که برای تمامی قسمت‌های کد، کامنت‌هایی قرار داده شده اند تا به این ترتیب مراحل انجام کار به راحتی قابل تشخیص باشند.

توضیحات دیتاست

پس از لود کردن دیتاست می‌بینیم که دیتاست ما دارای ۲۲۷ ستون و ۱۹۲۵ نمونه مختلف است. ۳۸۵ بیمار مختلف هستند که موارد مختلف برای آن‌ها ارزیابی شده است. هر بیمار دارای ۵ سطر می‌باشد که هر سطر موارد مختلف را در یک بازه‌ی زمانی خاص که در ستونی با نام "WINDOW" مشخص شده است، نمایش می‌دهد. این ستون دارای ۵ مقدار مختلف است که به صورت ۲-۰و ۴-۲و ۶-۴و ۱۲-۶ و Above-12 هستند. برای مثال ۲-۰ نشان دهنده‌ی این است که آزمایشات مختلف در بازه‌ی زمانی ۰ تا ۲ ساعت بعد از بستری شدن بیمار انجام گرفته اند.

تحلیل اکتشافی داده‌ها و مهندسی ویژگی‌ها

* این دو بخش اول را به صورت ترکیب‌شده با هم انجام می‌دهیم. زیرا در بعضی از مواقع باید یک سری اطلاعات به دست آورده و سپس تغییراتی در دیتاست بدهیم و این کار چند بار انجام می‌شود تا بتوانیم دیتاست را برای مدل‌های مختلف آماده کنیم.

در این قسمت باید ویژگی‌های مختلف دیتاست را بررسی کنیم و با استفاده از آن‌ها دیتاست را مرتب کنیم. در ابتدای نوت بوک در cell اول ابتدا تعدادی کتاب‌خانه را اضافه می‌کنیم که در پروژه‌های یادگیری ماشین به صورت گسترده مورد استفاده قرار می‌گیرند. سپس در cell دوم دیتاست را لود می‌کنیم و در cell سوم می‌بینیم که دیتاست دارای ۱۹۲۵ سطر و ۲۲۷ ستون میباشد که آزمایشات مختلف انجام شده روی بیماران را نشان می‌دهند. ستون PATIENT_VISIT_IDENTIFIER هم شماره‌ی بیماران را نشان

می‌دهد که ۳۸۵ مقدار متفاوت دارد و به این ترتیب می‌توان متوجه شد که ۳۸۵ بیمار مختلف داریم. در cell چهارم می‌بینیم که تقریبا ۱/۴ کل بیماران به ICU نیاز دارند. بنابراین دیتاست چندان بالانس نیست. سپس می‌بینیم که دیتاست ۲۱۹ ستون از نوع اعشاری و ۵ ستون از نوع integer و ۳ ستون از نوع object دارد. این اطلاعات را از طریق info() در cell پنجم به دست میاوریم.

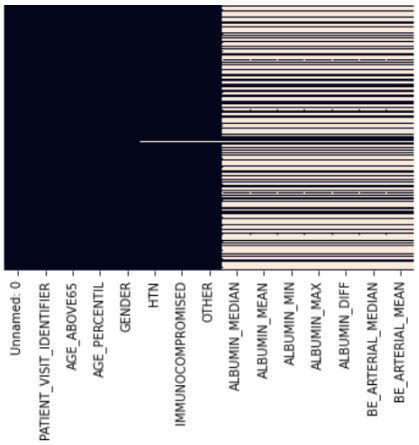
بررسی duplicate ها: اگر نمونه‌های تکراری در دیتاست داشته باشیم، وزن اهمیتی که مدل‌ها به آنها می‌دهند بیشتر خواهد شد که مطابق با چیزی که با می‌خواهیم نیست. بنابراین در cell ششم سطرهای مشابه را پیدا می‌کنیم. میبینیم که دیتاست ما هیچ دو سطر مشابهی ندارد.

بررسی ستون‌های object: چون که می‌خواهیم تمامی ستون‌ها مقادیر عددی داشته باشند در cell هفتم مقدارهای یکتای سه ستون AGE_PERCENTIL و WINDOW و tags را پیدا می‌کنیم تا با داشتن این اطلاعات بتوانیم این ستون‌ها را عددی کنیم. یعنی به هر یک از این مقادیر یکتا یک عدد نسبت دهیم.

- نکته‌ی مهم در رابطه با ستون tags این است که در این ستون تعدادی بیماری داریم که برای افراد مختلف تکرار شده اند. یعنی برای هر فرد مبتلا به ویروس کرونا، یک مجموعه‌ای از بیماری‌های زمینه‌ای برای او در این ستون نوشته شده است. بنابراین استفاده کردن از اطلاعات این ستون به همین شکلی که است فایده‌ای ندارد و باید بیماری‌های یکتا را که در این ستون آمده اند پیدا کنیم و برای آنها ستون جدا درست کنیم و برای هر فرد ببینیم که این بیماری در مجموعه‌ی بیماری‌های ستون tags برای او آمده است یا خیر؟! در ادامه‌ی notebook این کار را انجام داده‌یم.

در cell هشتم با استفاده از describe() تعدادی از ویژگی‌های ستون‌های عددی را بررسی می‌کنیم. این دستور جدولی را خروجی می‌دهد که میانه، میانگین، تعداد و چارک‌های مختلف را به ازای ستون‌های عددی نشان می‌دهد. در این جدول می‌بینیم که میانه بعضی از ستون‌ها بیشتر از میانگین آنها است (یعنی اکثر داده‌ها از عدد میانه کوچک‌تر هستند.) و در بعضی دیگر از ستون‌ها میانه کم‌تر از میانگین می‌باشد.

بررسی هیچ مقدار ها: در cell نهم با استفاده از heatmap بررسی می‌کنیم که دیتاست ما چه تعداد خانه‌ی خالی دارد. البته از آنجایی که تعداد ستون‌ها خیلی زیاد است این نمودار را تنها برای ۱۵ ستون اول کشیدم. در heatmap می‌بینیم که بخش‌های زیادی سفید هستند و این بخش‌ها نشان‌دهنده‌ی خانه‌های خالی دیتاست است. بنابراین به دلیل وجود تعداد زیادی خانه‌ی خالی باید فکری برای پر کردن این‌ها با استفاده از مقادیر مناسب بکنیم.



عددی کردن ستون‌های object:

- AGE_PERCENTIL: (cell دهم) برای عددی کردن این ستون به 10^{th} مقدار ۱، به 20^{th} مقدار 2 و ... می‌دهیم. بنابراین اسن سطر با استفاده از ۱۰ رقم، عددی می‌شود.
- WINDOW: (cell یازدهم) این ستون ۵ حالت مختلف صورت ۲-۰و ۴-۲و ۶-۴و ۱۲-۶ و Above-12 دارد که با اختصاص دادن عدد ۰ تا ۴ به هر یک از آنها میتوان این ستون را عددی کرد.
- tags: در ابتدا در cell دوازدهم خانه‌های خالی ستون tags را با استفاده از رشته‌ی None پر می‌کنیم. (زیرا تعداد زیادی از مبتلایان به کرونا هیچ بیماری زمینه‌ای نداشته‌اند و خانه‌ی مربوط به آنها خالی است.) سپس در cell سیزدهم بیماری‌های یکتا که در لیست بیماری‌های زمینه‌ای مبتلایان بود را پیدا می‌کنیم و در آرایه‌ای به نام uniqueSicknesses ذخیره کردیم. می‌بینیم که این آرایه ۶ بیماری زمینه‌ای را شامل می‌شود که عبارت اند از:

```
['Motor Neurone Disease', 'Smoker', 'Lung cancer', 'asthma',
'Kidney disease', 'heart disease']
```

سپس برای هر بیماری یک ستون ایجاد می‌کنیم و به ازای هر مبتلا مشخص می‌کنیم که او این بیماری را دارد یا نه. در آخر هم ستون tags را از دیتاست پاک می‌کنیم. در cell چهاردهم می‌توان تغییرات ایجاد شده در دیتاست را مشاهده کرد. می‌بینیم که ۶ ستون بیماری‌ها به آخر دیتاست اضافه شده‌اند.

در cell پانزدهم ستون ICU را به ستون آخر منتقل می‌کنیم.

حالا باز هم با استفاده از دستور info() در cell شانزدهم تغییرات ایجاد شده را بررسی می‌کنیم.

پر کردن خانه‌های خالی:

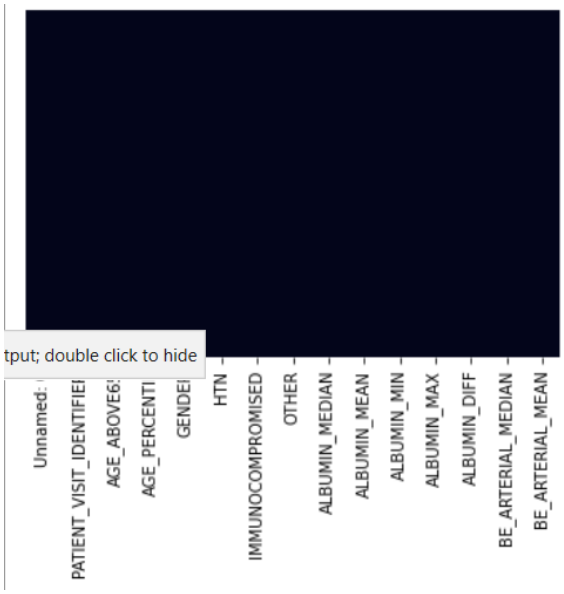
با توجه به heatmap ای که در قسمت‌های قبل notebook دیدیم باید فکری برای خانه‌های خالی جدول بکنیم. دو راه به ذهن من رسید:

۱) پر کردن خانه‌های خالی با توجه به مقادیر خانه‌های همسایه: با توجه به توضیحات دیتاست که در kaggle موجود است، جمع‌آورنده‌ی دیتاست پیشنهاد کرده است که از همین کار برای پر کردن خانه‌های خالی ستون‌ها استفاده کنیم.

۲) پر کردن خانه‌ها با استفاده از میانگین هر ستون: (cell های هجدهم و نوزدهم) برای ستون‌هایی که مقادیر اعشاری دارند می‌توان از میانگین هر ستون برای پر کردن خانه‌های خالی آن ستون استفاده کرد. برای ستون های integer از آنجایی که تمامی این ستون های مقدار 1- دارند بقیه ستون‌های خالی را می‌توان با 0 پر کرد.

با توجه به این که در description خود دیتاست روش اول پیشنهاد شده است ما هم از همین روش استفاده می‌کنیم.

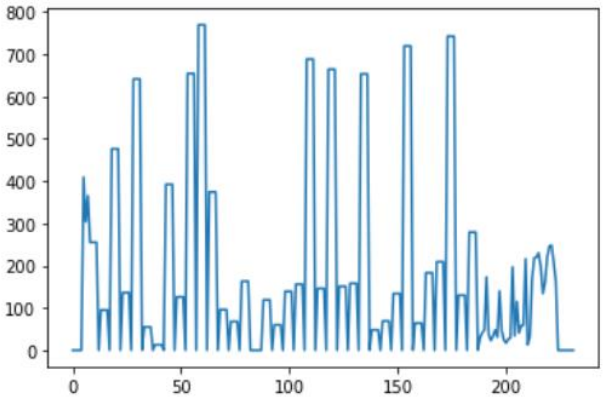
در cell بیستم می‌بینیم که خانه‌های خالی دیتاست پر شده‌اند و دیگر مقدار NaN نداریم. سپس در cell بیست و یکم دوباره heatmap را رسم میکنیم و می‌بینیم که دیگر هیچ بخشی از heatmap سفید نیست و این یعنی دیگر NaN در دیتاست نداریم.



پیدا کردن outlier ها:

معمولا در دیتاست‌ها تعدادی داده‌ی پرت وجود دارد که ممکن است باعث بد شدن نتیجه‌ی مدل‌های ما شوند. به همین دلیل لازم است که این‌ها پیدا شوند و در صورتی که لازم بود از دیتاست حذف شوند. در cell بیست و دوم، داده‌های بین چارک اول و سوم را به عنوان داده‌های معمولی و درست می‌گیریم ولی داده‌های خارج از این محدوده را به عنوان outlier در نظر می‌گیریم. خروجی این function تعداد outlier های ستون ورودی است. در cell بیست و سوم میانگین، میانه، کمترین و بیشترین تعداد outlier های ستون‌های مختلف را مشاهده می‌کنیم و در cell بیست و چهارم، با استفاده از نمودار متوجه می‌شویم که در یک سری از feature ها تعداد outlier ها خیلی زیاد است و در بقیه تقریبا مشابه است پس میتوان با قرار دادن یک threshold ستون‌هایی که outlier بالا دارند را از dataset حذف کرد. ترشلد را برابر با ۵۰۰ قرار می‌دهیم.

نمودار زیر تعداد outlier ها را به ازای feature ها نشان می‌دهد:

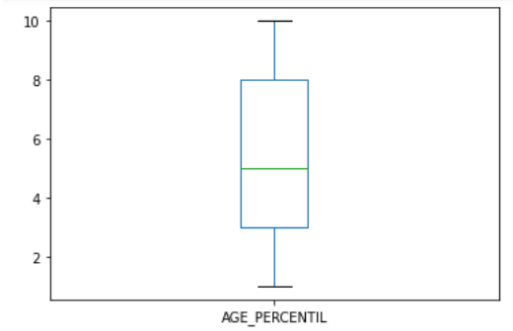


در cell بیست و پنجم ستون‌هایی با تعداد outlier بیشتر از ۵۰۰ عدد را از dataset حذف می‌کنیم و می‌بینیم که ۲۰۰ ستون در دیتاست باقی می‌ماند.

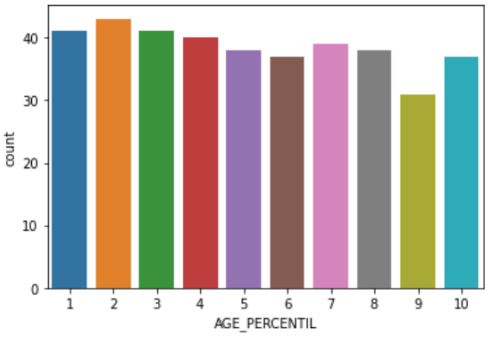
Visualization با نمودارها:

cell های بیست و ششم

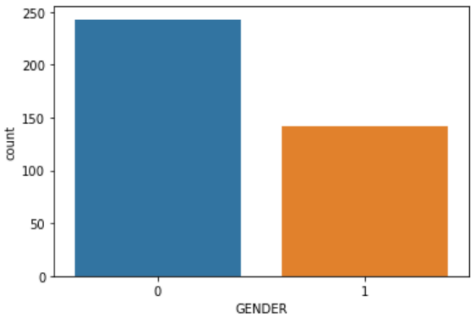
۱) نمودار اول نشان می‌دهد که نیمی از بیماران زیر ۵۰ سال و نیم دیگر بالای ۵۰ سال دارند.



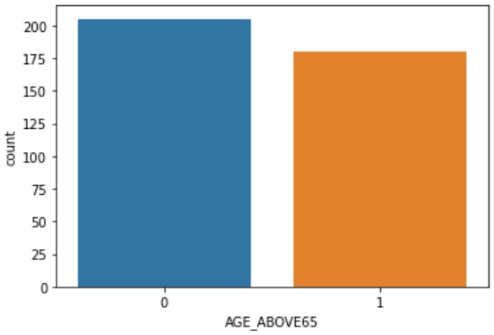
۲) نمودار دوم نشان دهنده‌ی این است که تعداد افراد سنین مختلف که به کرونا مبتلا شده‌اند یکسان است. البته باید این موضوع را در نظر گرفت که این به این معنی نیست که سن در ابتلا به کرونا تاثیر ندارد. زیرا تعداد افراد سنین مختلف در کشور نیاز است تا معلوم شود کدام رده‌ی سنی بیشتر مبتلا می‌شوند.



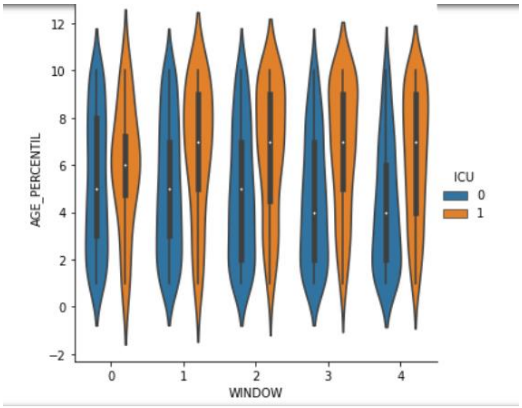
۳) نمودار سوم نشان دهنده‌ی این است که مردان بیشتر از زنان به کرونا مبتلا شده‌اند.



۴) نمودار چهارم در واقع کاملا مورد انتظار است زیرا از نمودار دوم متوجه شدیم که احتمال مبتلا شدن افراد سنین مختلف تقریبا یکسان است. پس تعداد افراد بالای ۶۵ باید کمی کمتر از افراد زیر ۶۵ سال باشند.



۵) از این نمودار می‌توان متوجه شد که به طور کلی افراد سن بالا بیشتر در icu بستری میشوند و این که در چک شدن‌های اولیه بیمار (در window size های کوچکتر) سن افرادی که به ICU نیاز دارند کمتر است. اما هر چه window size بیشتر می‌شود افراد با سن بالا بیشتر به ICU نیاز دارند و افراد کم‌سن تر معمولا به ICU نیاز ندارند.

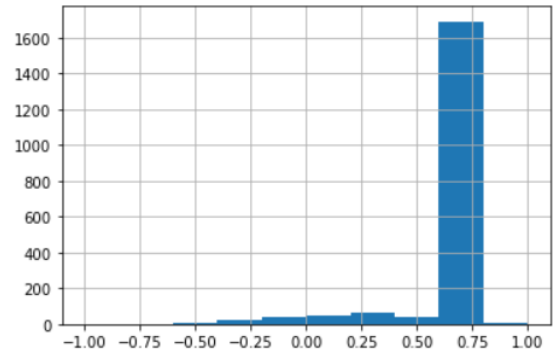


۶) در ۶ نمودار بعدی می‌بینیم که تمامی بیماری ها سبب شده‌اند که افراد بیشتر به ICU نیاز داشته باشند اما احتمال بستری شدن افراد چندان به سیگاری بودن یا نبودن آنها ربطی ندارد. پس از میان این بیماری‌ها فقط سیگاری بودن تاثیری روی احتمال بستری شدن ندارد.

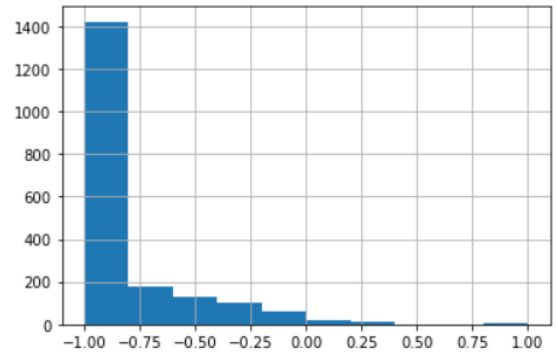
cell بیست و هفتم

در این بخش تعدادی هیستوگرام برای تعدادی از ویژگی‌های دیتاست رسم می‌کنیم. با رسم این نمودارها می‌بینیم که حتی ستون‌هایی با مقادیر float باز هم در یک محدوده‌ی خاصی متمرکز هستند و بنابراین نیاز به نرمال‌سازی دیتاست داریم.

ALBUMIN_MEDIAN

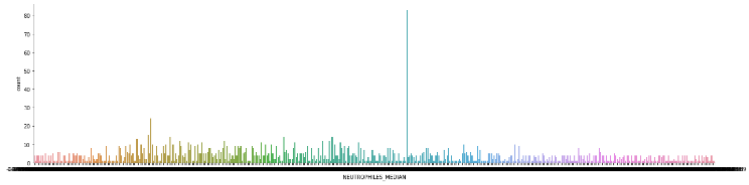


TEMPERATURE_DIFF_REL

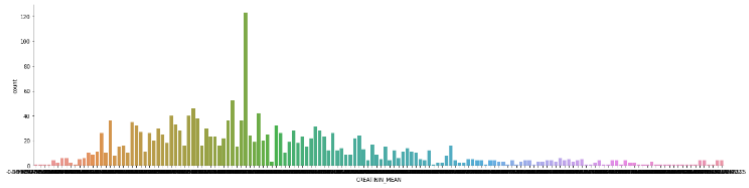


cell بیست و هشتم

این دو نمودار هم مانند ۵ نمودار قبلی نشان‌دهنده‌ی این هستند که دیتاست نیزا به نرمال شدن دارد.



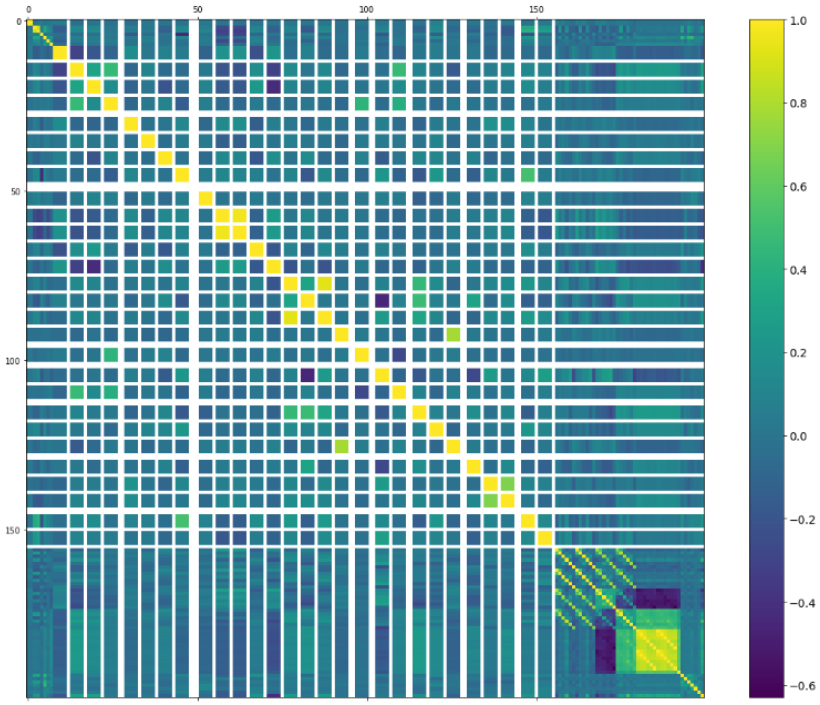
CREATININ_MEAN



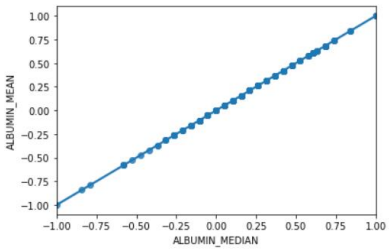
کریلیشن بین ویژگی‌ها:

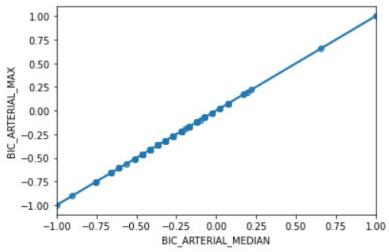
در cell بیست و نهم کریلیشن بین ویژگی‌های موجود را محاسبه می‌کنیم و در نمودار نشان داده شده قسمت هایی که زرد تر هستند نشان دهنده‌ی کریلیشن مثبت و آنهایی که آبی تیره تر هستند نشان

دهنده‌ی کریلیشن منفی هستند. برای مثال می‌بینیم که هر ویژگی با خودش کریلیشن مثبت دارد (مقدار ۱). به طور کلی برای ما ویژگیهایی که با هم کریلیشن مثبت یا منفی زیادی دارند نشان دهنده‌ی این است که اینها به هم وابسته هستند و اگر مثلا یکی را داشته باشیم دیگری تا حد خوبی مشخص است.



به همین دلیل در همین cell آنهایی که کریلیشنشان بیشتر از 0.9 یا کمتر از 0.75- است را در لیستی به نام pos_corrs نگه می‌داریم. در cell سی ام دو مورد از این دوتایی‌ها را به عنوان مثال رسم می‌کنیم که نشان دهنده‌ی این است که این دو ویژگی به هم وابسته هستند و وجود یکی دیگری را معلوم میکند.





تست مدل‌های مختلف و نتایج مدل نهایی:

* دو قسمت ۳ و ۴ پروژه را هم با هم توضیح می‌دهم زیرا به هم مرتبط هستند.

در cell سی و یکم ابتدا کتاب‌خانه‌های لازم را import می‌کنیم. مدل‌های استفاده شده تماما از کتاب‌خانه sklearn هستند.

در cell سی و دوم هفت مدلی که قرار است عملکرد آنها را روی دیتاست بررسی کنیم، نوشته‌ام که عبارت اند از:

۱) Logistic Regression
۲) Gaussian Naive Bayes
۳) Decision Tree Classifier
۴) K-Nearest Neighbors
۵) Support Vector Machine
۶) Neural Network
۷) Random Forest Classifier

در بخش‌های بعدی مدل‌ها را در دو حالت بررسی می‌کنیم.

۱) اول اینکه به ازای ۵ حالت مختلف window مدل‌ها را به گونه‌ای آموزش می‌دهیم که ویژگی‌ها را بگیریم و ICU همان ردیف را پیشبینی کنیم. یعنی مثلا به ازای window = 0 ردیف‌های آن را می‌گیریم و با استفاده از اعداد موجود در cell ها ICU آن ردیف را پیشبینی می‌کنیم.
۲) دوم این که early prediction انجام دهیم و به ازای window size های 0 تا 4 ویژگی‌ها را گرفته و ICU ردیف 5 را پیشبینی کنیم. به این ترتیب میتوانیم از علائم چند ساعت قبلتر بیمار متوجه شویم که به ICU نیاز خواهد داشت یا نه؟!

ابتدا حالت اول را بررسی میکنیم: در cell سی و سوم تابعی به نام trainTest0 نوشتیم که با گرفتن window ردیف‌هایی با آن مقدار window را فیلتر میکند تا مدل مناسبی را برای آن‌ها پیدا کند. سپس با استفاده از SelectKBest سعی می‌کنیم feature های مهم تر را پیدا کنیم ولی با warning این که feature ها ثابت هستند روبرو می‌شویم. بنابراین باید این feature های ثابت را پیدا کنیم و آنها را از دیتاست حذف کنیم. سپس باید preprocess روی دیتاست انجام دهیم در یکی از قسمت‌های قبل

با استفاده از نمودارها متوجه شدیم که دیتاست ما نیاز به نرمال شدن دارد. بنابراین ابتدا مقادیر ستون‌ها را نرمال می‌کنیم و سپس در مرحله‌ی بعد باید کاهش ابعاد انجام دهیم. از آنجا که تعداد feature های ما خیلی زیاد است ممکن است باعث شود زمان اجرای مدل‌های پیچیده تر خیلی زیاد شود و به علاوه به دلیل ریز شدن در جزئیات دقت مدل هم کاهش یابد. من با استفاده از Recursive Feature Elimination و Logistic regression تعداد ۵۵ عدد از میان حدود ۱۷۰ feature باقی مانده انتخاب کردم که بیشترین اطلاعات را دارند. Index این ویژگی‌ها در آرایه‌ای با نام selected_features قرار می‌گیرد و سپس با استفاده از iloc دیتاست را کاهش ابعاد می‌دهیم.

حال که کاملا دیتاست را آماده کردیم باید مدل‌ها را امتحان کنیم. برای این کار ابتدا باید دیتاست را به دو بخش داده‌های آموزشی و آزمایشی افراز کنیم. سپس مدل‌های مختلف را به آرایه‌ی مدل اضافه می‌کنیم و همانطور که مشخص است برای هر نوع (مثلا برای SVM) چند مدل داریم که پارامترهای آنها مختلف است. حالا مدل‌ها را آموزش می‌دهیم. با استفاده از StratifiedKFold و cross_val_score مدل‌ها را آموزش می‌دهیم و برای هر مدل میانگین fscore را برای fold های مختلف به عنوان نتیجه‌ی کلی مدل در نظر می‌گیریم. سپس مدل‌ها را تست می‌کنیم. به ازای هر مدل با fit کردن داده‌های آموزش مدل را روی داده‌های تست امتحان می‌کنیم و مقادیر پیشبینی شده توسط مدل را با لیبل‌های اصلی داده‌های تست مقایسه کرده و مقدار fscore را ذخیره می‌کنیم. خروجی این function، نام مدل‌ها، نتایج آموزش مدل‌ها و مقادیر پیشبینی شده برای داده‌های تست و مدل‌ها و feature های انتخاب شده و تعدادی مورد دیگر است.

در cell سی و چهارم تنها کاری که می‌کنیم این است که warning ها را ignore می‌کنیم تا در خروجی نشان داده نشوند.

در cell سی و پنجم تابع trainTest0 را به ازای window size های مختلف صدا می‌زنیم و نتایج را ذخیره می‌کنیم. نتایج به ازای حالت‌های مختلف در خروجی این قسمت دیده میشوند. این نتایج در بخش‌های بعدی با استفاده از نمودارهایی بهتر تحلیل می‌شوند و روش بهینه از میان این روش‌ها انتخاب می‌شود. البته با یک نگاه کلی متوجه می‌شوبم که نتایج مدل‌ها به ازای window size های بیشتر، بهتر است.

در cell سی و ششم برای window size های مختلف boxplot نتایج مدل‌هایمان را روی داده‌های آموزش رسم می‌کنیم.

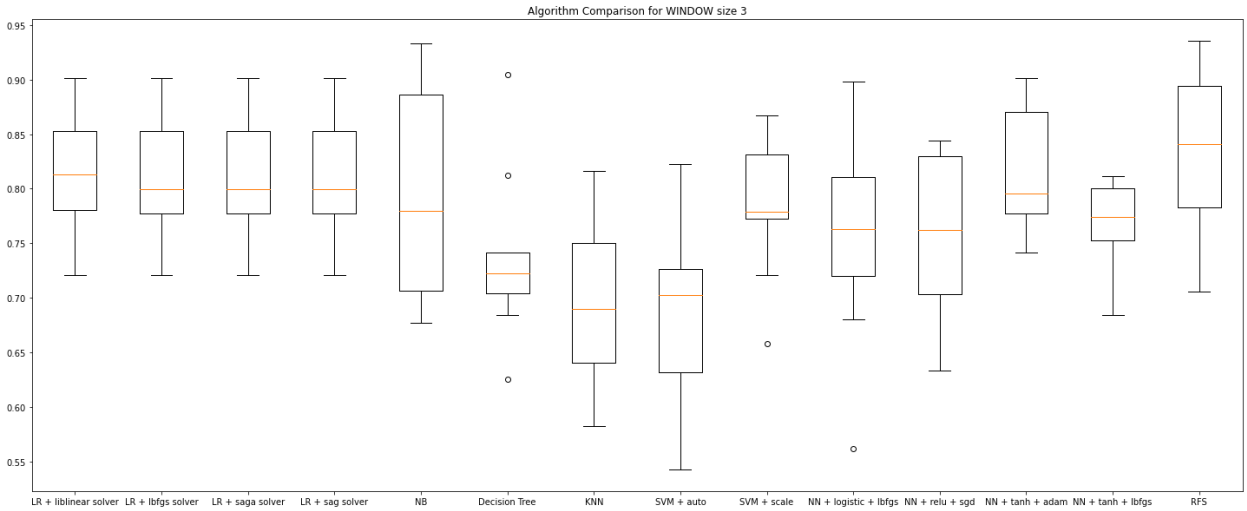
به ازای window = 0 می‌بینیم که شبکه‌های عصبی به طور کلی بهتر عمل کرده‌اند.

به ازای window = 1 و window = 2می‌بینیم که logistic regression ها بهتر از بقیه مدل‌ها بوده‌اند.

به ازای window = 3 می‌بینیم که Random forest بهتر بوده است.

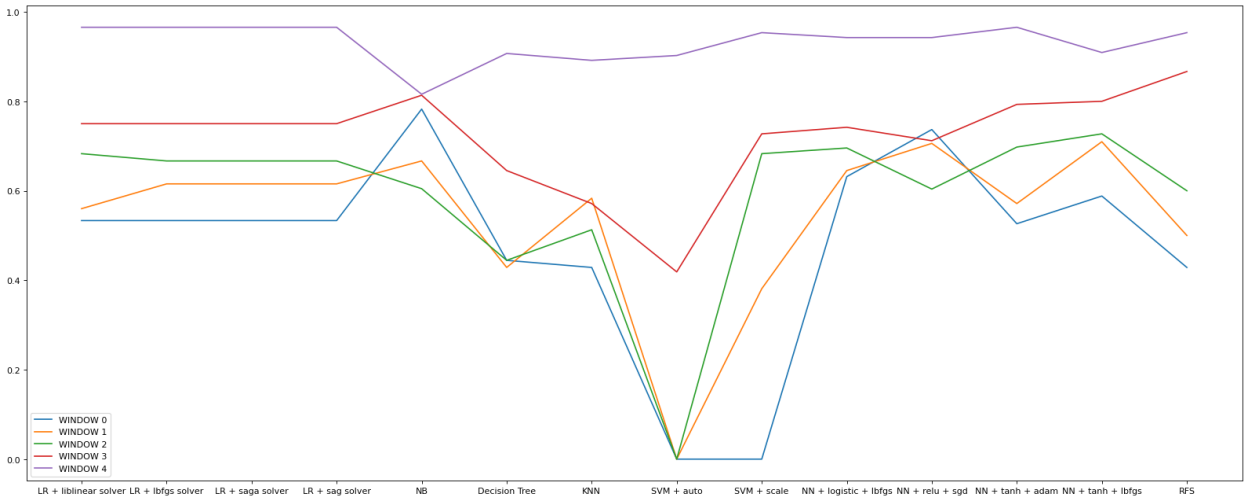
به ازای window = 4 می‌بینیم که Random forest و logistic regression بهتر از بقیه بوده‌اند.

نمودار زیر نتایج مدل‌ها را برای window = 3 نشان می‌دهد.



در cell سی و هفتم نموداری دیگر برای بررسی عملکرد مدل‌ها به ازای window size های مختلف رسم کرده‌ایم. که نشان می‌دهد window = 0 و window = 4 بهتر از بقیه بوده اند. که البته خیلی منطقی است چون در دیتاست این دو ردیف به ازای بیماران مختلف تعداد کمتری خانه‌ی خالی داشته اند.

در cell سی و هشتم همان نمودار قبلی را برای داده‌های تست رسم کرده‌ایم. که می‌بینیم نتایج مدل‌ها برای window size های بزرگتر بهتر هستند. با استفاده از این نمودار متوجه می‌شویم مدل‌های logistic regression و RFS بهتر توانسته اند داده‌های ستون ICU را پیشبینی کنند. بین پارامترهای مختلف logistic regression هم تفاوتی وجود ندارد.



در cell سی و نهم نتایج را برای دو مدل انتخاب شده روی بهترین بازه‌ی آزمایش یعنی بالای ۱۲ ساعت، نشان داده‌ایم.

```
Scores on WINDOW size: 4
model LR + liblinear solver:
precision: [0.91428571 1. ]
recall: [1. 0.93333333]
```

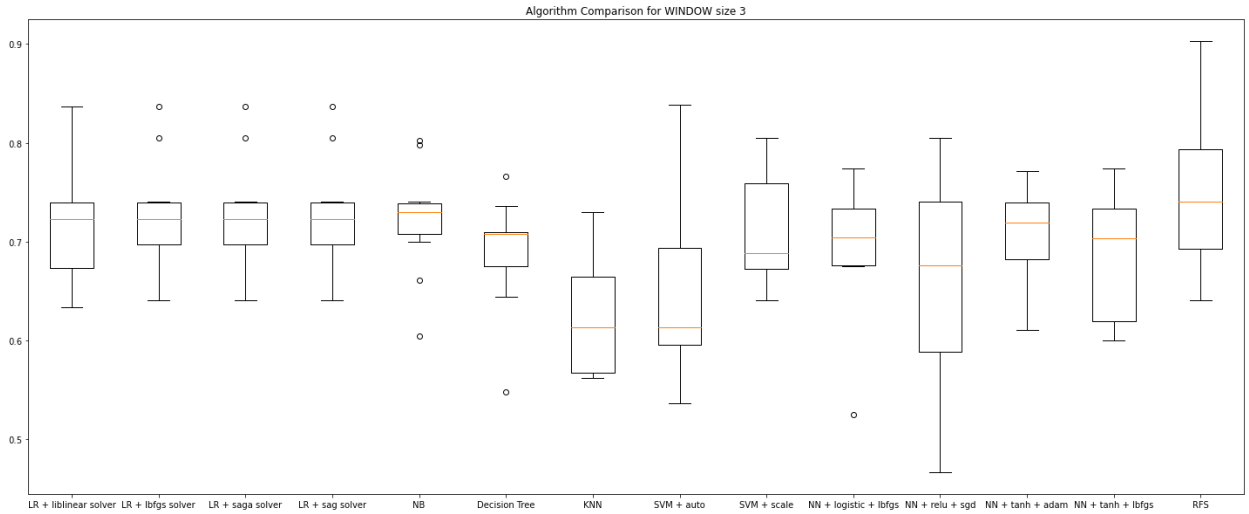
```
fscore: [0.95522388 0.96551724]
model RFS:
precision: [0.88888889 1. ]
recall: [1. 0.91111111]
fscore: [0.94117647 0.95348837]
```

خب از این به بعد نتیج early prediction را بررسی می‌کنیم و مدل بهینه را برای این حالت پیدا می‌کنیم.

در cell چهلم دقیقا کاری مانند تابع trainTest0 انجام می‌دهیم. تنها چیزی که متفاوت است این است که ICU همواره ICU برای window = 4 است.

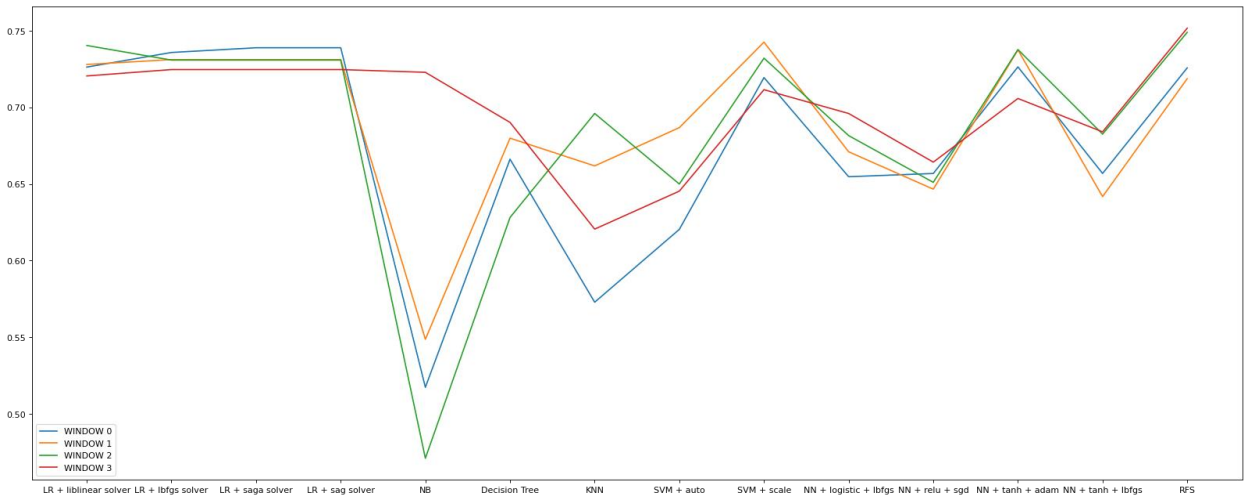
در cell چهل و یکم هم تابع earlyPrediction را به ازای window size های بین ۰ تا ۳ بررسی صدا می‌کنیم. (چون قرار است زودتر متوجه شویم که آیا بیمار به ICU نیاز دارد یا نه پس باید window آخر را استفاده نکنیم.) در این جا می‌بینیم که نتایج برای window های بزرگتر دقیق‌تر است. دلیل آن هم واضح است. چون علائم بیمار هر چه می‌گذرد بیشتر نشان می‌دهند که آیا او بعد از ۱۲ ساعت به ICU نیاز خواهد داشت یا نه؟!

در cell چهل و دوم نمودارهایی رسم کردیم که مشخص می‌کنند به ازای window size های مختلف معمولا RFS و logistic regression و Neural Network با adam بهتر از بقیه هستند.

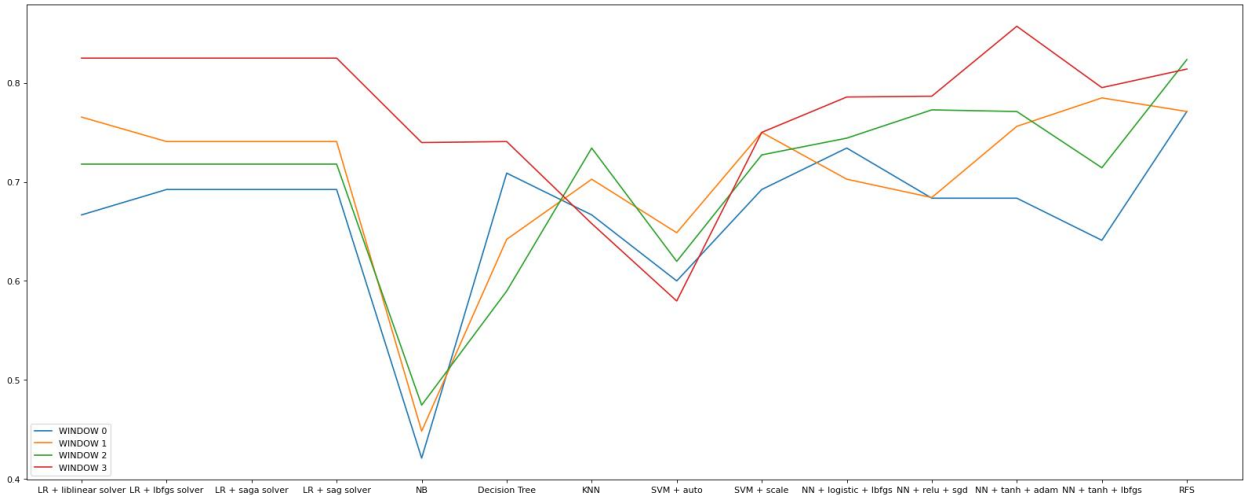


در cell چهل و سوم و چهل و چهارم هم عملکرد مدل‌ها را به ازای window size های مختلف مقایسه کرده‌ایم. که در نمودار تست می‌بینیم که window = 3 خیلی بهتر از بقیه می‌تواند ICU را پیشبینی کند.

نتایج روی داده‌های آموزشی:



نتایج روی داده‌های آزمایشی:



در cell چهل و پنجم هم metric های دیگر را برای این سه مدل انتخاب شده اندازه گرفته‌ایم.

```
Scores on WINDOW size: 3
model LR + liblinear solver:
precision: [0.71428571 0.94285714]
recall: [0.9375 0.73333333]
```

```
fscore: [0.81081081 0.825 ]
model NN + tanh + adam:
precision: [0.76315789 0.92307692]
recall: [0.90625 0.8 ]
fscore: [0.82857143 0.85714286]
model RFS:
precision: [0.72222222 0.85365854]
recall: [0.8125 0.77777778]
fscore: [0.76470588 0.81395349]
```

انتخاب بهترین مدل:

خب دیدیم در حالتی که icu هر سطر را با استفاده از اعداد همان سطر پیشبینی کنیم مدل‌های logistic regression و RFS بهتر از بقیه هستند.

در حالت early prediction هم مدل‌های RFS و logistic regression و Neural Network با adam بهتر بودند. پس باید بین این مدل‌ها یکی را انتخاب کنیم.

در ابتدا بگویم که neural network ها را به دلیل سخت تر بودن و البته زمانبر بودن آموزش آنها از لیست کاندیدهای مدل برتر حذف کردم.

برای حالت اول اگر به نتایج cell سی و نهم مراجعه کنیم می‌بینیم که تمامی metric ها برای logistic regression بالاتر شده‌اند. پس در حالت اول logistic regression بهتر از RSF است. در حالت دوم هم logistic regression به جز precision برای ICU = 0 در تمامی metric های دیگر از RSF بهتر است. با توجه به سادگی این مدل که به همراه libnear solver داریم، آن را به عنوان بهترین مدل روی این دیتاست گزارش می‌کنم. (cell چهل و ششم)

دلایل انتخاب مدل logistic regression:

۱) سادگی
۲) سرعت بالا
۳) Precision و Recall و fscore بالا

نحوه‌ی کارکردن مدل logistic regression:

Logistic regression در مسائل classification به صورت گسترده مورد استفاده قرار می‌گیرد. این مدل نوعی مدل خطی است.

داریم: $f(\mathbf{x}) = b_0 + b_1 x_1 + \cdots + b_r x_r$

$b_0, b_1, \ldots, b_r$ ها در واقع همان وزن‌هایی هستند که مدل به هر یک از feature های x_1 و ... می‌دهد.

تابع logistic regression $p(\mathbf{x}) = 1 / (1 + \exp(-f(\mathbf{x}))$: $f(\mathbf{x})$ است. $p(\mathbf{x})$ در واقع احتمال این که خروجی ۱ باشد را نشان می‌دهد. پس $1- p(\mathbf{x})$ احتمال صفر شدن خروجی را نشان می‌دهد. مدل وزن‌ها را به گونه‌ای انتخاب می‌کند که p(x) تا حد ممکن نزدیک به مقدار واقعی label باشد. برای این کار باید log-likelihood function را بیشینه کنیم.

$$\text{LLF} = \Sigma_i(y_i \log(p(\mathbf{x}_i)) + (1 - y_i) \log(1 - p(\mathbf{x}_i))).$$

برای این که این تابع را بیشینه کنیم میتوان از gradient descent استفاده کرد. تنها کافی است در ابتدا وزن‌ها را یک مقدار اتفاقی بدهیم و سپس الگوریتم را اجرا کنیم.

* نکته‌ی مهم این است که اگر feature های وابسته را حذف نکنیم اصولا مدل logistic regression خوب عمل نمی‌کند.

خب پس به طور خلاصه تا اینجا داریم:

۱) در حالت اول با استفاده از اعداد ردیف پنجم هر بیمار نیاز به ICU او را در همان ردیف پیشبینی می‌کنیم

پس cell چهل و هفتم وزن‌های داده شده به feature های مختلف (که ۵۵ تای مهم آن را برای آموزش مدار انتخاب کرده بودیم.) را نشان میدهد.

۲) برای early prediction از داده‌های ردیف چهارم استفاده کرده و ICU ردیف پنجم را پیشبینی می‌کنیم.

پس cell چهل و هشتم وزن‌های داده شده به feature های مختلف (که ۵۵ تای مهم آن را برای آموزش مدار انتخاب کرده بودیم.) را نشان میدهد.

لینک‌های استفاده‌شده:

۱) https://machinelearningmastery.com/machine-learning-in-python-step-by-step/
۲) https://medium.com/@atanudan/exploratory-data-analysis-eda-in-python-893f963cc0c0
۳) https://towardsdatascience.com/feature-engineering-in-python-part-i-the-most-powerful-way-of-dealing-with-data-8e2447e7c69e
۴) https://machinelearningmastery.com/feature-selection-machine-learning-python/