پروژه درس یادگیری ماشین

گزارش کامل :

مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

عنوان مقاله:

COVID-19 Patient Health Prediction Using Boosted Random Forest Algorithm (Adaboost)

(Received: 14 may 2020- published: 03 June 2020-Frontiers)

استاد محترم: خانم دکتر ساجدی

دانشجو: سارا جنیدی

زمستان ۹۹

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

فهرست

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

- Recall
- Precision
- F1-Score

مروری بر مقاله و متد مورد استفاده

دیتاست

- **توضیح Features**
- **Pre Processing**

بهبود مقاله (Classifier پیشنهادی)

- **مقایسه نتایج**

کارهای آتی

دیگر منابع مورد مطالعه

پیوست

- **توضیح کدهای پیاده سازی**
- **جدول مقایسه نتایج**

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

دیباچه

امروزه استفاده از داده های حاصل از بررسی مدل های رفتاری افراد و سازمان ها، جهت تصمیم گیری در حوزه های مختلف و ارائه راهکار برای حل معضلات محیط زیست و تغییرات اقلیمی، اقتصاد، تجارت و کسب و کار، بهداشت و بیماری (از جمله پیشگیری، تشخیص و درمانی) و ... بیشتر از هر زمان دیگری مورد توجه قرار گرفته است.

از این رو، توجه به دو مورد ضروری بنظر می رسد:

یکم– استفاده از ابزار ها و روش های هوشمند

دوم– استفاده از داده های واقعی و ترکیبی بعنوان منبع قابل تامل جهت تصمیم گیری

بدین رو، ادغام تکنیک های هوش مصنوعی (AI) در زیرساخت های بی سیم ، جمع آوری real time داده ها و پردازش آنها با هدف دستیابی به نتایج دقیق تر و ارائه راه حلی بهینه، اجتناب ناپذیر می نماید.

این امر بویژه زمانی نمود بیشتر می یابد که معضلی در سطح محلی تبدیل به یک مساله عمومی در سطحی وسیع و گسترده می شود، بطوریکه افراد بیشتری از جوامع بشری را از جهات گوناگون تحت تاثیر قرار می دهد، در این خصوص می توان به بیماری همه گیر کووید۱۹ اشاره کرد که با توجه به آمار منابع رسمی از ووهان چین نشات گرفته ولی امروزه بر اساس آمار بهداشت جهانی و مرکز پیشگیری و کنترل بیماری اروپا تا این لحظه (۲۰ فوریه ۲۰۲۱ ساعت ۱:۵۴ دقیقه بعدازظهر به وقت تهران)حدود 111,315,106 مبتلا و 2,465,353 مرگ و 86,196,300 بهبود یافته در دنیا گزارش شده است.

در این پژوهش، کوتاه به بررسی مقاله ذکر شده پرداخته، در ادامه شرحی از راهکار پیشنهادی جهت بهبود نتایج و مقایسه نتایج بدست آمده با مقاله را مد نظر خواهم داشت. نویسندگان مقاله پس از استفاده از classifierهای مختلف بر روی داده های حاصل از موقعیت جغرافیایی، مسافرت ها، وضعیت سلامت و شدت بیماری و زمان مراجعه به بیمارستان، به این نتیجه می رسد که در پیش بینی سلامت فرد (نسبت به بیمار Covid19)، استفاده از مدلRandom Forest که با Adaboost classifier تقویت شده

نتایج به مراتب بهتری (Accuracy حدود 94% و F1 score حدود 0.86) نسبت به سایر Classifier ها از خود نشان می دهد.

این درحالیست که، نتایج حاصل از پژوهش اینجانب، با استفاده از مدل Random Forest که با Gradient Boosting تقویت شده نه تنها Accuracy را به 95% ارتقا می دهد بلکه F1 score را به میزان 0.90 افزایش داده که این بهبود حاصل از افزایش مقدار Recall از 0.75 (در مقاله) به 0.83 بوده است.

تعاریف و مفاهیم- اندمی، اپیدمی ، پاندمی و شیوع

از واژه های فوق در حوزه های مختلف استفاده می شود، اما تعریف دقیق و علمی در حوزه بهداشت و بیماری به شرح زیر است:

:Epidemic

وقتی رخ می دهد که بیماری تعداد زیادی از افراد را در یک جامعه ، جمعیت یا منطقه تحت تأثیر قرار می دهد.

:PANDEMIC

همه گیری، اپیدمی است که در چندین کشور یا قاره گسترش یافته است، در واقع اپیدمی است که بعلت سفر و سایر ارتباطات جغرافیایی گسترش یافته است.

ENDEMIC:

وقتی بیماری متعلق به یک قوم یا کشور خاص است، در واقع در آن محدوده جغرافیایی بومی است. بیماری های بومی حضور مداوم در یک مکان خاص دارند. بعنوان مثال مالاریا در مناطقی از آفریقا بومی است. اندمی در مرحله ای فراتر می تواند به شیوع منجر شود. در این مرحله دیگر متعلق به تنها یک مکان خاص نیست.

OUTBREAK:

افزایش بیش از حد پیش بینی شده در تعداد موارد بومی، شیوع بیماری نامیده می شود. همچنین می تواند شامل گزارش یک مورد واحد در یک منطقه جدید باشد. اگر به سرعت کنترل نشود ، شیوع آن می تواند به یک اپیدمی تبدیل شود.

اپیدمی در برابر اندمیک

یک اپیدمی به طور فعال در حال گسترش است و موارد جدید بیماری به طور قابل توجهی بیش از حد انتظار است. یک اپیدمی اغلب در یک منطقه قرار دارد ، اما تعداد مبتلایان در آن منطقه به طور قابل توجهی بیشتر از حد طبیعی است. به عنوان مثال ، وقتی COVID-19 محدود به ووهان چین بود، در دسته بیماریهای بومی قرار می گرفت در صورتیکه گسترش جغرافیایی آن را به یک بیماری همه گیر تبدیل کرد. [1]

مروری بر مدل های کلاسیک اپیدمی

دو مدل استاندارد گسترش بیماری در شبکه افراد جامعه، مدل اپیدمی SIR و SIS است. دیگر مشتقات مدل های اپیدمی (SIRS, SIRF, SIRSI و ...) از این دو مدل کلاسیک الگو گرفته اند.

مدل SIR (Susceptible-Infected-Removed) :

در این مدل که برای اولین بار در دهه ۱۹۲۰ توسط Lowell Reed و Wade Hampton Frost ارائه شد، افراد شبکه به سه کلاس تقسیم می شوند :

Susceptible (S) :افرادیکه مستعد ابتلا به بیماری هستند و اگر در معرض افراد آلوده یا مبتلا به بیماری قرار گیرند، مبتلا می شوند.

Infected(I): افرادی که به این بیماری مبتلا هستند و می توانند آن را به دیگران منتقل کنید

Removed(R) : افراد بهبود یافته که معمولا مصونیت دائمی می یابند، بطوریکه فرد در روند اپیدمی شرکت نخواهد کرد .

با توجه به تعریف فوق ، روابط بین S ، Iو R به صورت زیر تعریف می شود :

β:احتمال آلوده شدن فرد مستعد توسط فرد آلوده در یک دوره زمانی

γ:میزان افراد آلوده ، بنابراین ، در مدل اپیدمی SIR داریم:

$$\frac{ds}{dt} = -\beta is, \qquad \frac{di}{dt} = \beta is - \gamma i, \qquad \frac{dr}{dt} = \gamma i$$

در نظر داشته باشیم که S, I و R نسبت به کل جمعیت جامعه محاسبه می شوند.

در روند واقعی شیوع اپیدمی ، اگر فرد با برخی دیگر از افراد آلوده تماس گرفته باشد ، آنگاه مستعد ابتلا خواهد شد.

به این ترتیب ، فرد مستعد می تواند آلوده شود اگر و فقط اگر باشد حداقل یک فرد آلوده را ملاقات کرده باشد.

مدل SIS (Susceptible-Infected-Susceptible):

اگر افراد پس از بهبودی از بیماری مصونیت دریافت نکنند ، ما نمی توانیم از مدل SIR استفاده کنیم، بنابراین برای این نوع موارد ما اغلب از مدل SIS استفاده می کنیم که به این معنی است که پس از بهبودی ، افراد آلوده به مرحله مستعد برمی گردند ، بنابراین ، در این مدل داریم: [3][2]

$$\frac{ds}{dt} = -\beta is + \gamma i, \qquad \frac{di}{dt} = \beta is - \gamma i$$

بیماری کووید۱۹

تاریخچه، علائم و نشانه ها:

- SARS-CoV-2 originated in bats
- Special coronaviruses have jumped species and can be transmitted between people
- This is the third coronavirus to have done so since 2002:
  - ***Severe Acute Respiratory Syndrome (SARS)*** CoV emerged in Guangdong, China, in 2002
  - ***Middle Eastern Respiratory Syndrome (MERS)*** CoV emerged in the Middle East in 2012
  - ***SARS-CoV-2*** emerged in Wuhan, China, in 2019

SARS-CoV-2 viruses coming out of cell

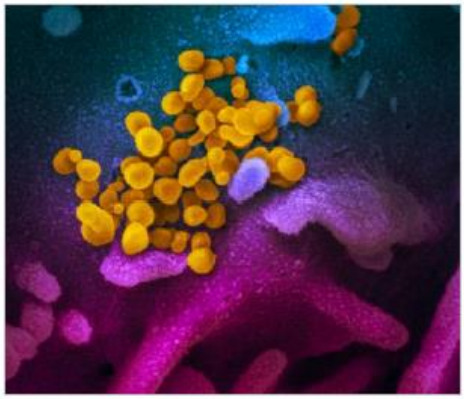


Photo credit: US National Institute of Allergy and Infectious Diseases, Rocky Mountain Laboratories (NIAID-RML).

نوعی خاص از ویروس کرونا است که جهش از گونه جانوری به گونه انسانی داشته است و اکنون از انسان به انسان انتقال می یابد. SARS-CoV2 (COVID19) که در واقع به ویروس کرونایی که سندروم بشدت حاد تنفسی ایجاد می کند گفته می شود. نوع اول این ویروس در سال ۲۰۰۲ در چین دیده شد به اپیدمی رسید، اما نوع دوم این ویروس که در ابتدا در چین دیده شد به پاندمی تبدیل شده است.

علائم(Signs) اندازه گیری عینی برای توصیف بیماری در طول معاینه فیزیکی است:

- دما
- تنفس سریعتر از حد معمول

نشانه ها (Symptoms) همان چیزی است که بیماران در مورد احساس خود می گویند:

- خستگی
- حالت تهوع

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

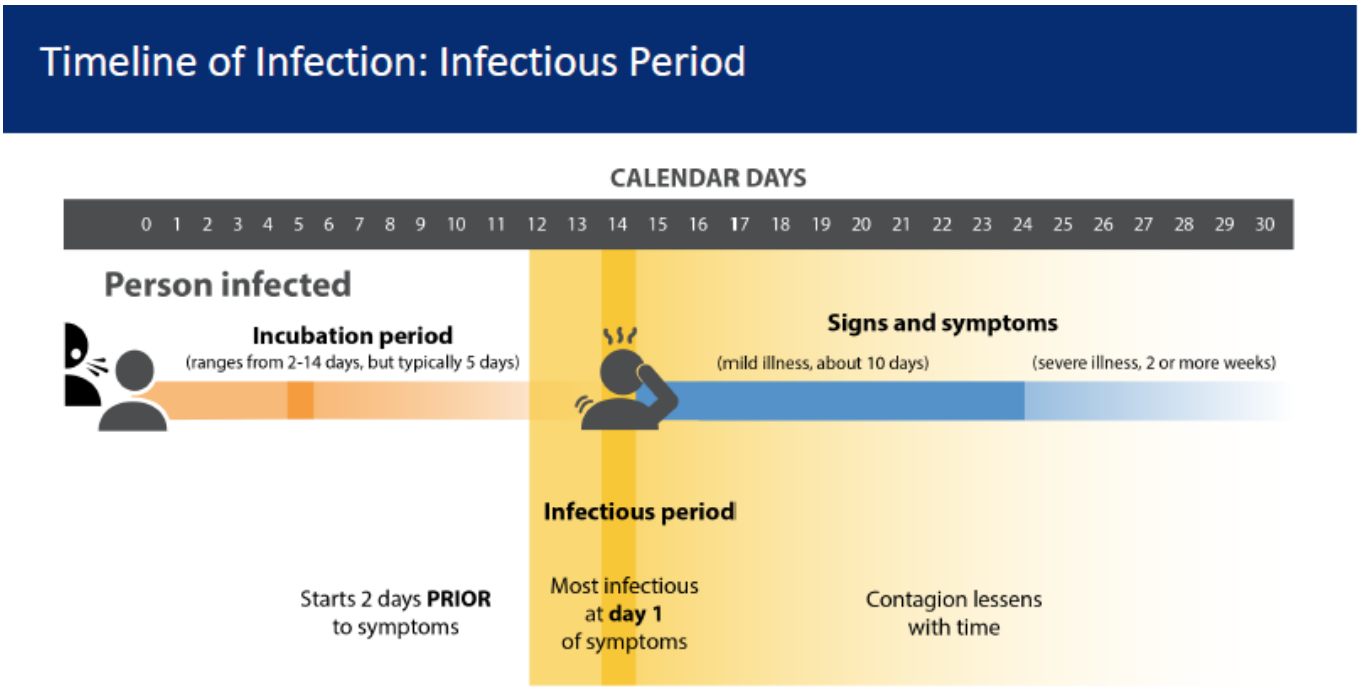
▶ عدم حس طعم یا بو (حس چشایی و بویایی) که از آن بعنوان نشانه خاص بیماری می توان نام برد چرا که یک سوم افراد مبتلا این نشانه را تجربه کرده اند. در صورتیکه در دیگر بیماری های تنفسی این نشانه مورد توجه نبوده است.

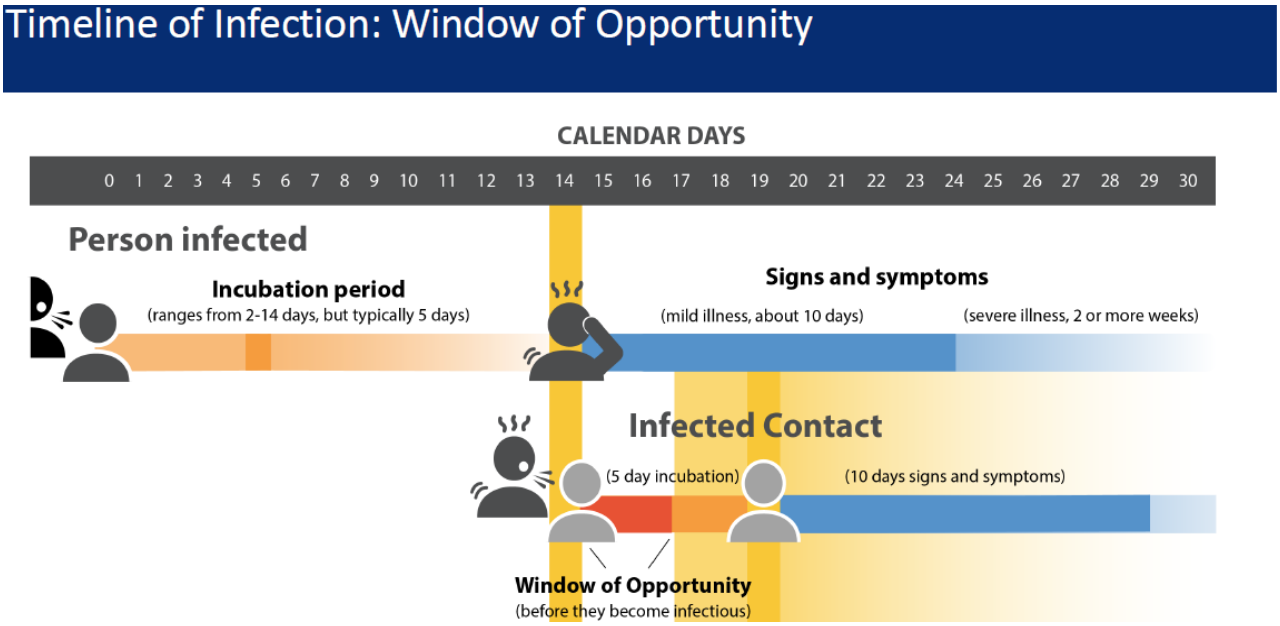
▶ درد عضله

برخی موارد چون تب هم یک نشانه است و هم یک علامت

شناسایی و تمایز میان نشانه و علامت در بحث Feature Extraction می تواند مفید باشد.

تایم لاین ابتلا بیماری کووید۱۹





Source: Eisenberg, J. (2020 March 17). R0: How scientists quantify the intensity of an outbreak like coronavirus and predict the pandemic's spread. The Conversation US. Accessed May 4, 2020.

دو نمودار زمانی فوق بخوبی دوره کمون بیماری، دوره ابتلا به بیماری تا دوره بهبود را نشان می دهد، همچنین مبتلا شدن دیگر افراد که در تماس با شخص بیمار هستند و زمان مناسب قطع زنجیره ابتلا را پیشنهاد می دهد.

در ادامه به توضیح دو نمودار زمانی فوق می پردازم:

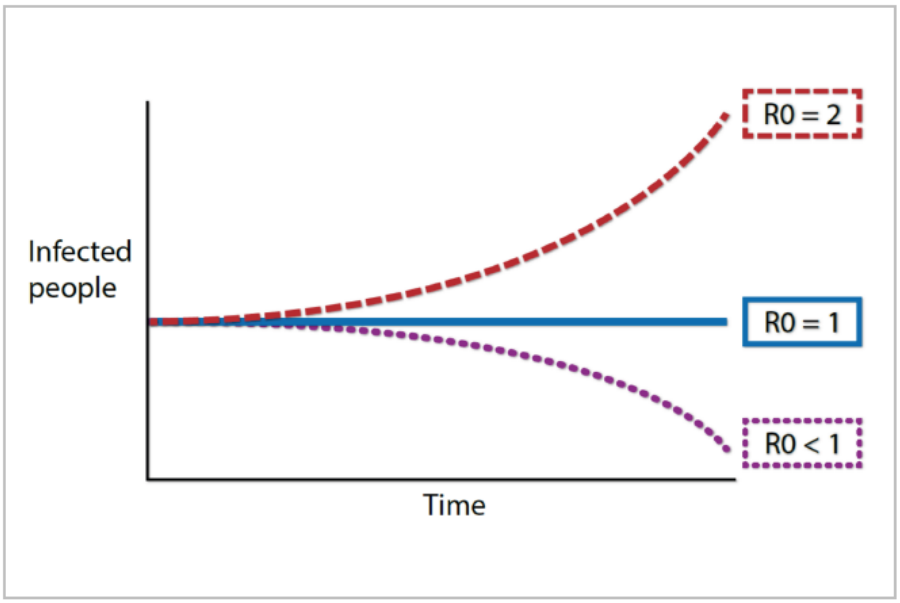
وقتی فرد یا افرادی آلوده شده اند علائم و نشانه های بیماری را معمولاً حدود پنج روز پس از آلوده شدن ، اما حداکثر ۱۴ روز پس از ابتلا بروز می دهند. سپس خود آنها می توانند دیگران را (دو روز قبل از شروع نشانه) آلوده کنند ، به خصوص در روزی که بیمار می شوند. و آنها تا زمانی که علائم و نشانه هایی داشته باشند ، (حداقل ۷ روز ادامه دارد)، می توانند افراد دیگر را آلوده کنند.

تصور کنیم که افراد در روزی که بیمار شده اند ، با شروع احساس بیماری ، با شخص دیگری در تماس باشند. فرض کنیم آن شخص یک دوره نهفتگی پنج روز داشته باشد. بنابراین پس از آلوده شدن ، پنج روز بعد علائم و نشانه هایی را بروز می دهد.

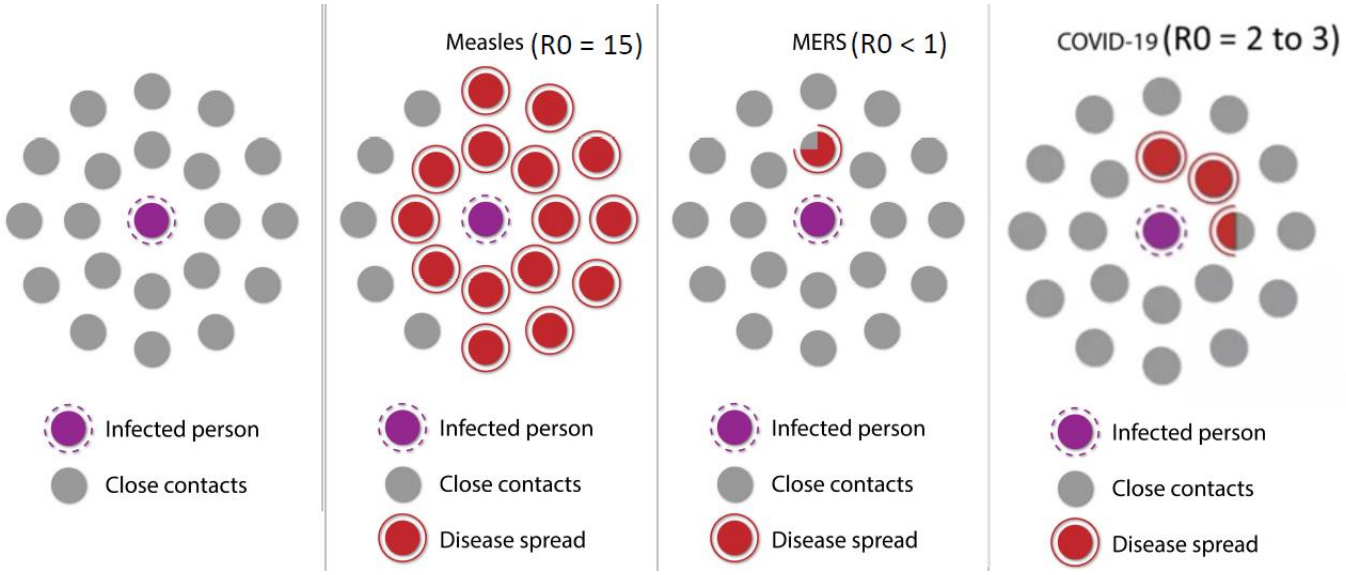
در اینجا می توان دید که پس از ایجاد علائم و نشانه ها ، به مدت هفت روز دچار بیماری می شود. همچنین می توانیم درناحیه ای که زرد رنگ برجسته شده است دوره واگیری را مشاهده کنیم. بنابراین می بینید که آنها دو روز قبل از بروز بیماری مبتلا هستند ، که در واقع تنها سه روز پس از ابتلا به آن آلوده هستند. بنابراین اگر می خواهیم زنجیره انتقال COVID-19 را متوقف کنیم باید افراد آلوده را پیدا کنیم و تعداد افرادی را که با آنها تماس دارند محدود کنیم تا زنجیره انتقال ادامه پیدا نکند. اما می توانید در این شماتیک ببینید که چرا انجام این کار بسیار دشوار است. زیرا خیلی سریع بعد از ابتلا ، خود افراد آلوده شده و می توانند به افراد دیگر بیماری را انتقال دهند.

این بدان معنی است که، دریچه فرصت برای ما جهت یافتن افرادی که در معرض بیماری قرار گرفته اند ، پیدا کردن افرادی که آلوده هستند اما هنوز بیمار نیستند و از آنها بخواهیم رفتار خود را تغییر دهند برای محدود کردن ارتباط آنها با افراد دیگر ، جلوگیری از انتقال ، بسیار کوتاه است . بنابراین سیستم های بهداشت عمومی باید سریعاً به موارد COVID-19 پاسخ دهند و سعی کنند همه کسانی را که در معرض این موارد قرار گرفته اند شناسایی کنند، تا بتوان اقدامی در جهت توقف یا کاهش زنجیره انتقال انجام داد.

Reproductive Number (R0 or R Naught) [6][7]



یک اصطلاح ریاضی است که نشان می دهد یک بیماری عفونی تا چه اندازه مسری است .همچنین از آن به عنوان شماره تولید مثل یاد می شود. این عدد معیاری مناسب برای نمایش سرعت شیوع یک بیماری در جمعیت است. و هرچه R0 بیشتر باشد ، افراد بیشتری در طول هر شیوع آلوده می شوند .در واقع این عدد نشان میدهد یک فرد بیمار (Infected) چه تعداد افرادیکه در ارتباط با او هستند را می تواند بیمار کند در صورتیکه آن افراد مستعد (Susceptible) باشند.

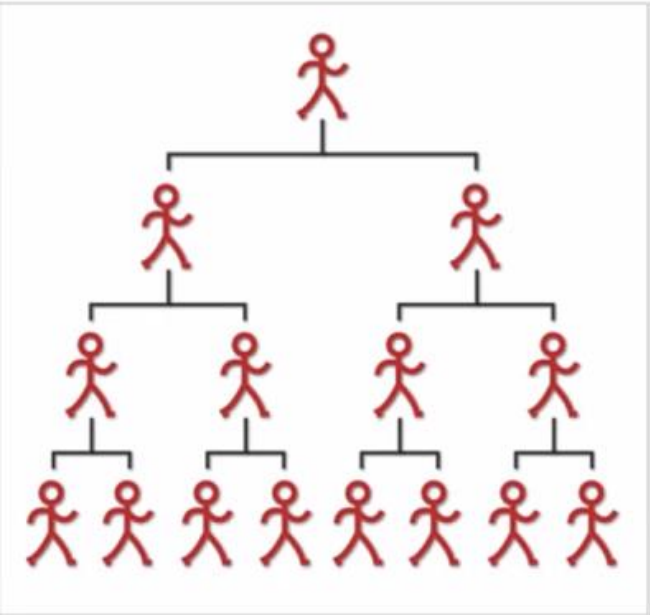
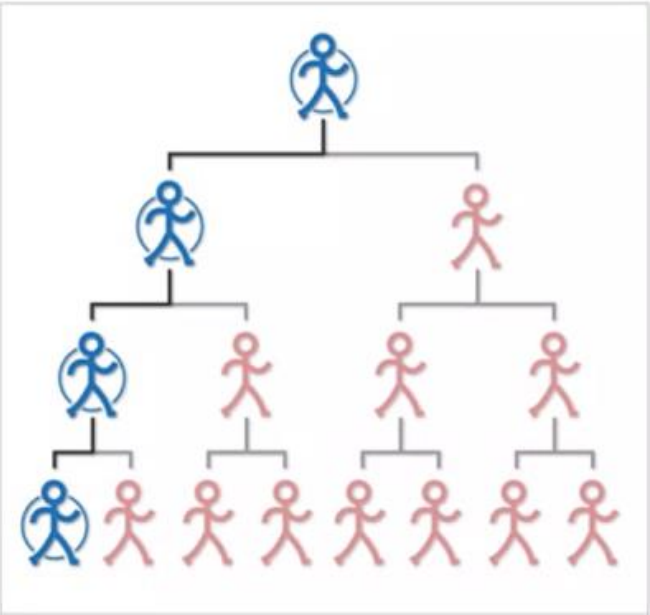


Source: Eisenberg, J. (2020 March 17). R0: How scientists quantify the intensity of an outbreak like coronavirus and predict the pandemic's spread. The Conversation US. Accessed May 4, 2020.

در شکل فوق میزان R0 را در بیماریهای ویروسی مختلف مشاهده می کنیم. همانطور که در تصویر بیان شده در بیماری کووید۱۹، هر فرد مبتلا می تواند بطور متوسط ۲ تا ۳ نفر دیگر را آلوده کند. این در حالیست که در نوع جدید این بیماری مقدار R0 به میزان ۴۰ تا ۸۰ درصد رشد داشته است. اما برای وضوح بیشتر اثر R0 در زنجیره انتقال بیماری شکل زیر، نمایی از زنجیره انتقال با R0=2 را نشان می دهد. **تصویر سمت راست بیان می کند اگر فرد بیمار سریعتر تشخیص داده شود (R0=1) و تنها یک فرد را مبتلا کند، چه اثری در روند کاهشی زنجیره انتقال می گذارد. R0 از طریق زیر محاسبه می شود:[9]**

$$R_0 = \tau\, c\, d$$

τ is the transmission probability per contact
c is the contact rate (number of contacts between individuals per unit time)
d is the length of the infectious period

R0 = 2	اگر هر فرد مبتلا از آلوده کردن نفر دوم بازبماند
	

Source: Eisenberg, J. (2020 March 17). R0: How scientists quantify the intensity of an outbreak like coronavirus and predict the pandemic's spread. The Conversation US. Accessed May 4, 2020.

اکنون پس از بیان مقدمات و شرح بیماری کووید۱۹، درجه ابتلا و نیز میزان اهمیت تشخیص افراد بیمار برای قطع یا محدود کردن زنجیره انتقال وارد مبحث Classifiers یا طبقه بندی کننده ها می شویم.

در اینجا سعی بر آن دارم تا توضیح کوتاه و مختصری بر معرفی Classifier هایی که استفاده شده است بدهم، سپس شرحی مختصر بر معیارهای ارزیابی و درجه اهمیت و اولویت هر یک در حوزه بیماری ها خواهم داشت.

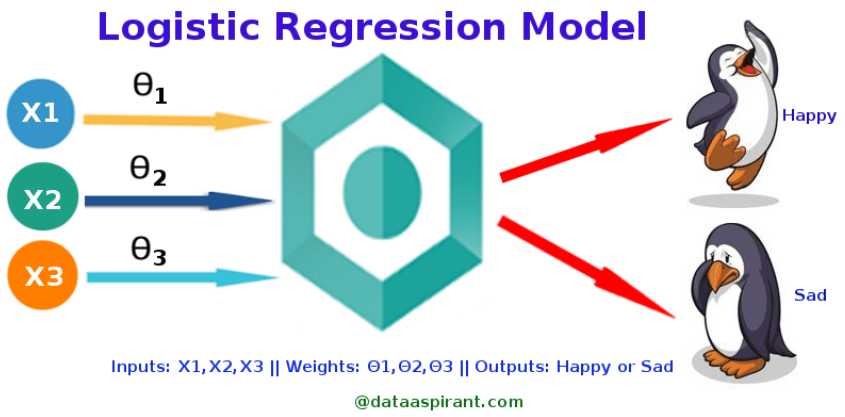
طبقه بندی کننده ها(Classifiers):

SVM (Support Vector Machine)

SVM یا ماشین بردار پشتیبان برای Classification مورد استفاده قرار می گیرد. نسخه ای از این الگوریتم با نام SVR یا Support Vector Regressionبرای Regression استفاده می شود. این الگوریتم قدیمی است و برای برای Binary Classification پیشنهاد شده بود، بطور کلی این الگوریتم یک خط پیدا می کند که دو کلاس را بنواند از هم جدا کند، بنابراین SVM برای دیتاست هایی که Linear Separable هستند مناسب است.

در واقع، ماشین های بردار پشتیبانی (SVM) مجموعه ای از روش های یادگیری نظارت شده است که برای طبقه بندی ، رگرسیون و تشخیص داده ههای پرت استفاده می شود.

LR (Logestic Regression)



رگرسیون لجستیک در اوایل قرن بیستم در علوم زیستی مورد استفاده قرار گرفت. سپس در بسیاری از کاربردهای علوم اجتماعی بکار گرفته شد. رگرسیون لجستیک هنگامی استفاده می شود که متغیر وابسته (هدف) Categorical باشد .مثلا، برای پیش بینی اینکه آیا ایمیلی هرزنامه هست یا خیر (۱) یا (0) . یا اینکه تومور بدخیم باشد (۱) یا نه(0)

انواع رگرسیون لجستیک

رگرسیون لجستیک دودویی

پاسخ Categorical بوده و تنها دو خروجی می تواند داشته باشد بله یا خیر، ۰ یا ۱ مثال: ایمیل، هرزنامه هست یا خیر

رگرسیون لجستیک چند جمله ای

سه دسته یا بیشتر بدون درجه بندی. مثال: پیش بینی اینکه کدام غذا بیشتر ترجیح داده می شود (گیاهی ، غیر گیاهی ، مخلوط)

رگرسیون لجستیک عادی

سه دسته یا بیشتر با در نظر گرفتن درجه اهمیت یا ترتیب. مثال: رتبه بندی فیلم از ۱ تا ۵

مرز تصمیم گیری (Decision Boundary) برای پیش بینی اینکه کدام داده مربوط به کدام کلاس است ، می توان یک آستانه تعیین کرد. بر اساس این آستانه ، احتمال تخمینی بدست آمده به کلاس ها داده می شود. بعنوان مثال اگر Predicted value> 0.5 باشد ایمیل هرزنامه تشخیص داده می شود. این مرز می تواند خطی یا غیرخطی باشد. برای بدست آوردن مرز تصمیم گیری پیچیده می توان درجه چند جمله ای را افزایش داد.

Gaussian Naïve Bayes

Classifier های Naive Bayes براساس قضیه Bayes ساخته شده است. فرض بر این است که استقلال قوی بین ویژگی ها وجود دارد. این Classifier ها فرض می کنند که مقدار یک ویژگی خاص مستقل از ارزش هر ویژگی دیگر است. در یک وضعیت یادگیری تحت نظارت ، طبقه بندی کننده های Naive Bayes بسیار خوب آموزش می بینند و برای برآورد پارامترهای مورد نیاز برای طبقه بندی، به داده های آموزشی کوچکی نیاز دارند.

Decision Tree

درخت تصمیم یک نمایش ساده برای طبقه بندی نمونه هاست . بر پایه یادگیری ماشین نظارت شده است که در آن داده ها به طور مداوم بر اساس یک پارامتر خاص تقسیم می شوند .درخت تصمیم شامل گره ها (ریشه، میانی، برگ) و یالها یا شاخه هاست.

دو نوع اصلی درخت تصمیم وجود دارد:

Classification

درخت طبقه بندی است که در آن نتیجه یک متغیر "مناسب" یا "نامناسب" یا "بله" و "خیر" است. در اینجا متغیر تصمیم گیری گسسته است.

چنین درختی از طریق فرایندی ساخته می شود که به پارتیشن بازگشتی باینری معروف است. این یک فرایند تکراری است که داده ها را به پارتیشن تقسیم می کند ، و سپس آنها را بیشتر بر روی هر یک از شاخه ها تقسیم می کند.

Regression

درختان تصمیمی که متغیر هدف مقادیر پیوسته بگیرد به عنوان مثال قیمت خانه ، یا مدت اقامت بیمار در بیمارستان

قبل از اینکه تصمیم بزرگی بگیریم، از نظرات دیگران استفاده می کنیم تا دچار bias یا variance نشویم.

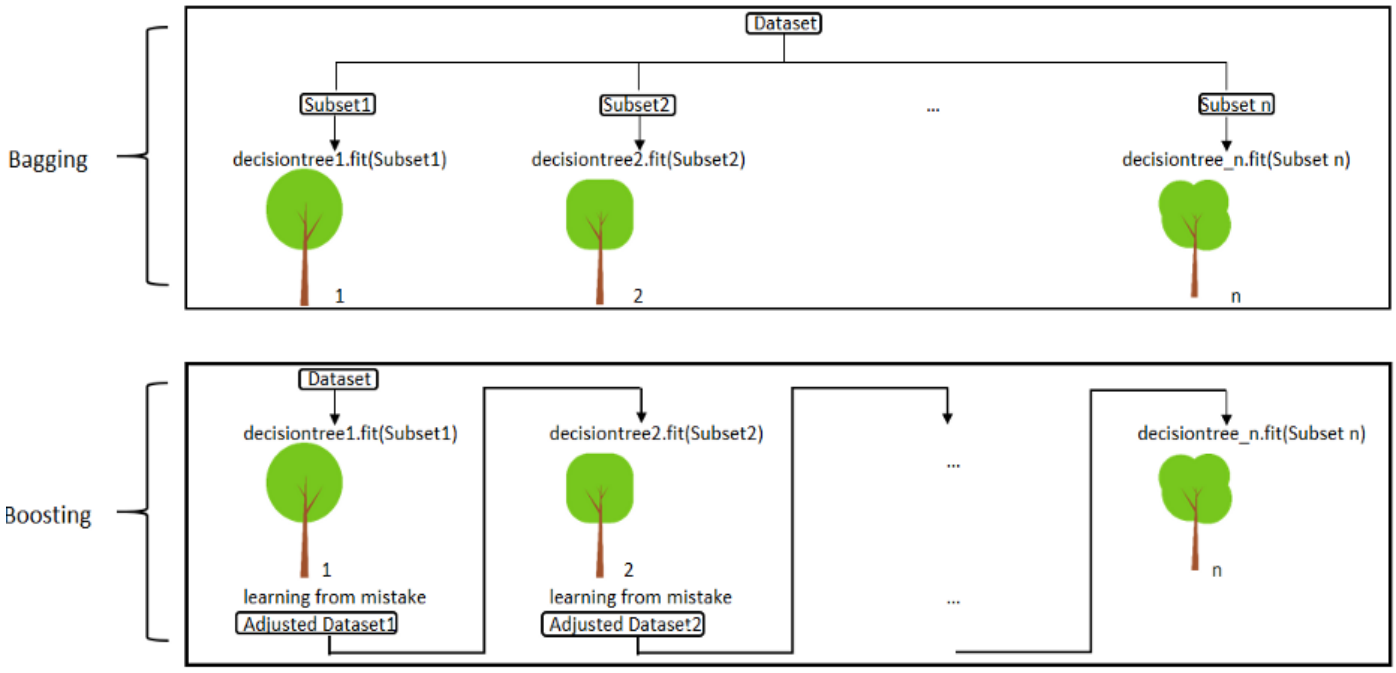
بهمین دلیل از مدل انفرادی به آموزش گروهی رجوع می کنیم.

به طور کلی یادگیری گروهی مدلی است که براساس تعدادی از مدل های مختلف پیش بینی می کند. با ترکیب مدلهای منفرد ، مدل گروه تمایل به انعطاف پذیری بیشتری دارد (بایاس کمتر) و حساسیت کمتری به داده نشان می دهد (واریانس کمتر).

دو متد بسیار مورد استفاده در Ensembling شامل Bagging و Boosting است.

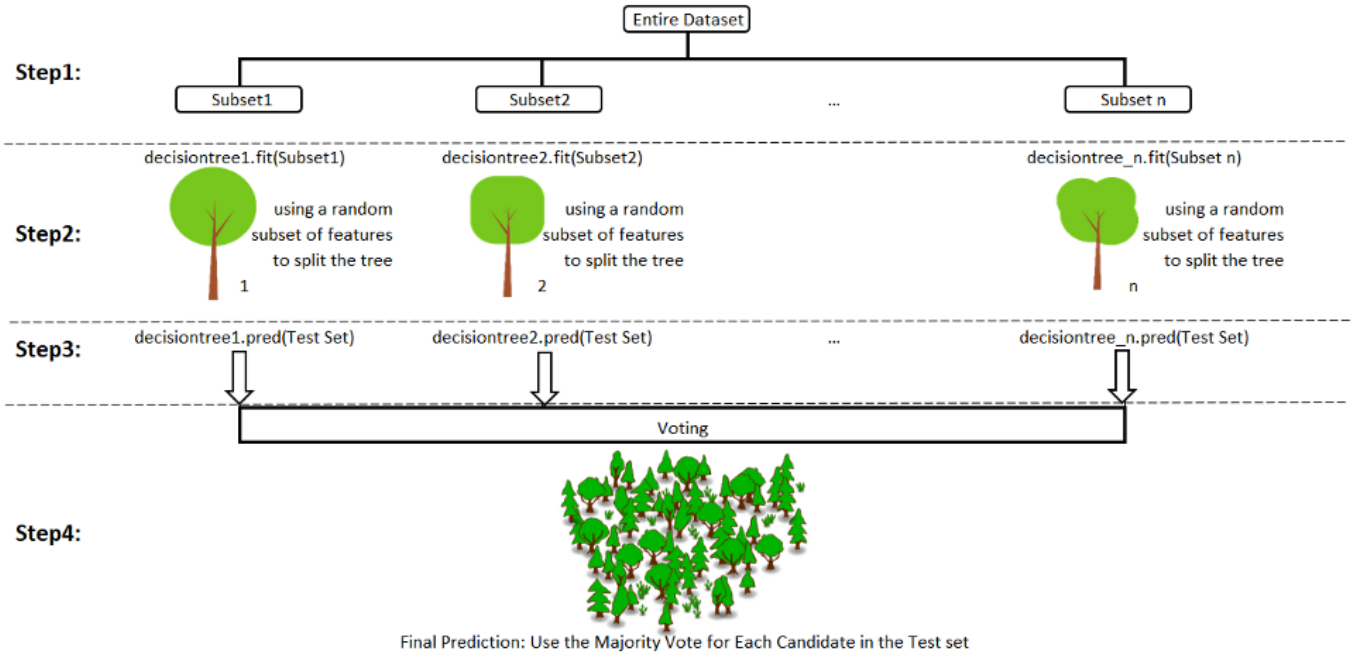
Bagging: دسته ای از مدل های منفرد را به صورت موازی آموزش می دهیم. هر مدل توسط زیر مجموعه تصادفی داده ها آموزش داده می شود

Boosting: آموزش دسته ای از مدلهای منفرد به روشی متوالی. هر مدل منفرد از اشتباهات مدل قبلی درس می گیرد.


Bagging
Dataset
Subset1
Subset2
...
Subset n
decisiontree1.fit(Subset1)
decisiontree2.fit(Subset2)
decisiontree_n.fit(Subset n)
1
2
n
Boosting
Dataset
decisiontree1.fit(Subset1)
decisiontree2.fit(Subset2)
...
decisiontree_n.fit(Subset n)
1
2
n
learning from mistake
Adjusted Dataset1
learning from mistake
Adjusted Dataset2
...

Random Forest

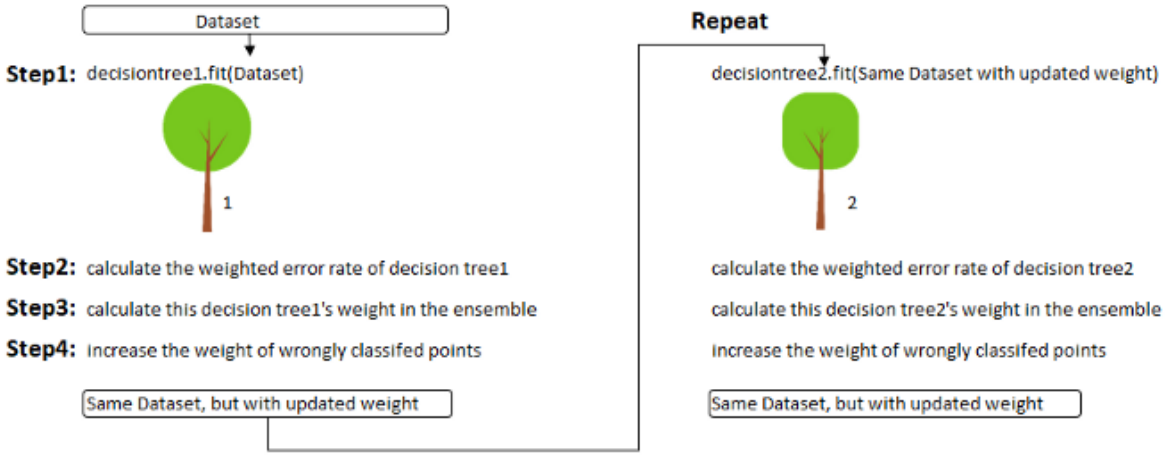
جنگل تصادفی یک Ensemble Model است که از Bagging به عنوان روش Ensemble و درخت تصمیم به عنوان مدل فردی استفاده می کند.



مطابق تصویر فوق در مرحله اول n زیر مجموعه بصورت تصادفی از مجموعه آموزشی انتخاب می شود. سپس n درخت تصمیم آموزش می بیند. از یک زیر مجموعه تصادفی برای آموزش یک درخت تصمیم استفاده می شود تقسیم بهینه برای هر درخت تصمیم بر اساس یک زیر مجموعه تصادفی از ویژگی ها است (به عنوان مثال اگر ۱۰ ویژگی در کل داشته باشیم، به طور تصادفی ۵ از ۱۰ ویژگی را برای تقسیم انتخاب می کنیم) . در مرحله سوم هر درخت جداگانه رکوردها / داوطلبان را در مجموعه تست predict می کند و در مرحله آخر پیش بینی نهایی با استفاده از Majoritz Voting انجام می شود.

Adaboost (Adaptive Boosting)

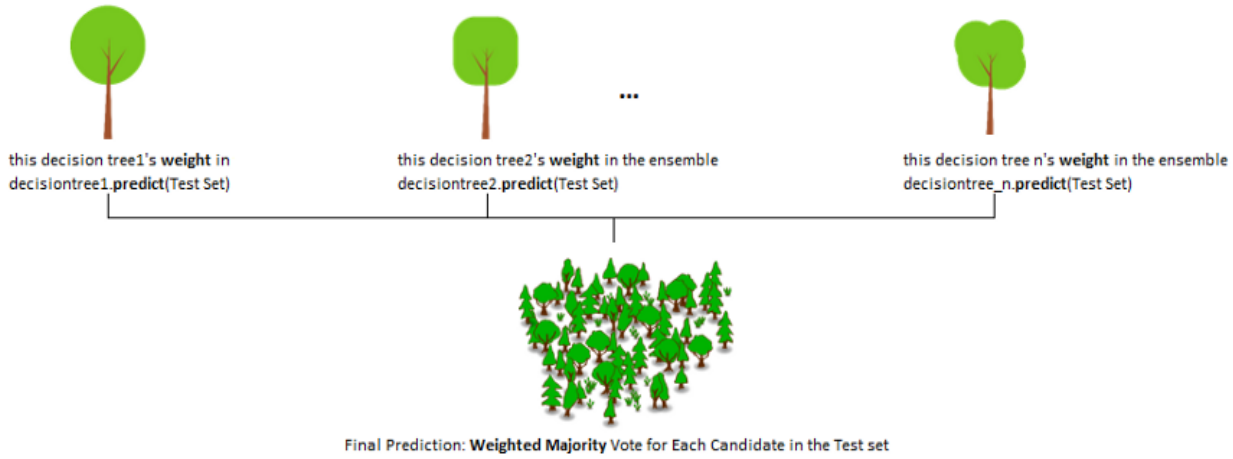
AdaBoost یک مدل ensemble boosting است که با درخت تصمیم بسیار خوب کار می کند . رمز موفقیت مدل یادگیری از اشتباهات قبلی است ، AdaBoost با افزایش وزن داده هایی که نادرست طبقه بندی شده اند از اشتباهات می آموزد.



در ابتدا به داده ها وزن اولیه داده می شود. بعنوان مثال اگر مجموعه آموزش دارای ۱۰۰ داده باشد ، وزن اولیه هر داده ۰/۰۱ است. سپس آموزش درخت تصمیم و در مرحله بعد، محاسبه میزان خطای وزنی (e) که شامل تعداد پیش بینی اشتباه از کل است. در مرحله سوم، محاسبه وزن درخت تصمیم در ensemble :

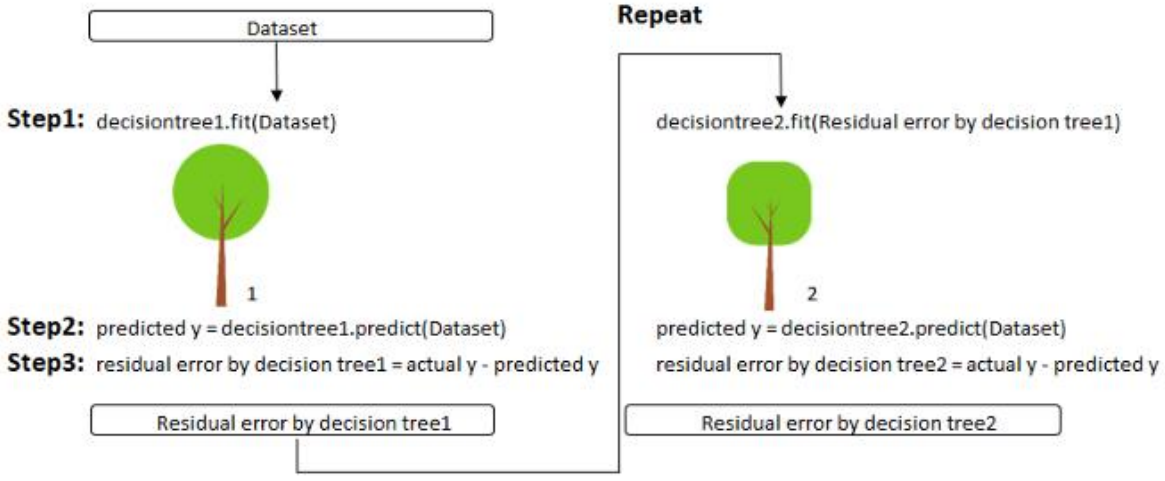
$$\text{وزن این درخت} = \text{learning rate} * \log((1 - e) / e)$$

هرچه میزان خطای وزنی یک درخت بالاتر باشد، قدرت تصمیم گیری کمتری به درخت در رای گیری بعدی داده می شود و برعکس. مرحله چهارم شامل بروز رسانی وزن نقاطی است که به اشتباه طبقه بندی شده اند. مرحله پنجم، تکرار مرحله یکم (تا رسیدن به تعداد درختانی که برای آموزش مد نظر بوده است). مرحله ششم: پیش بینی نهایی ، درخت با وزن بالاتر قدرت تأثیر بیشتری در تصمیم نهایی خواهد داشت.



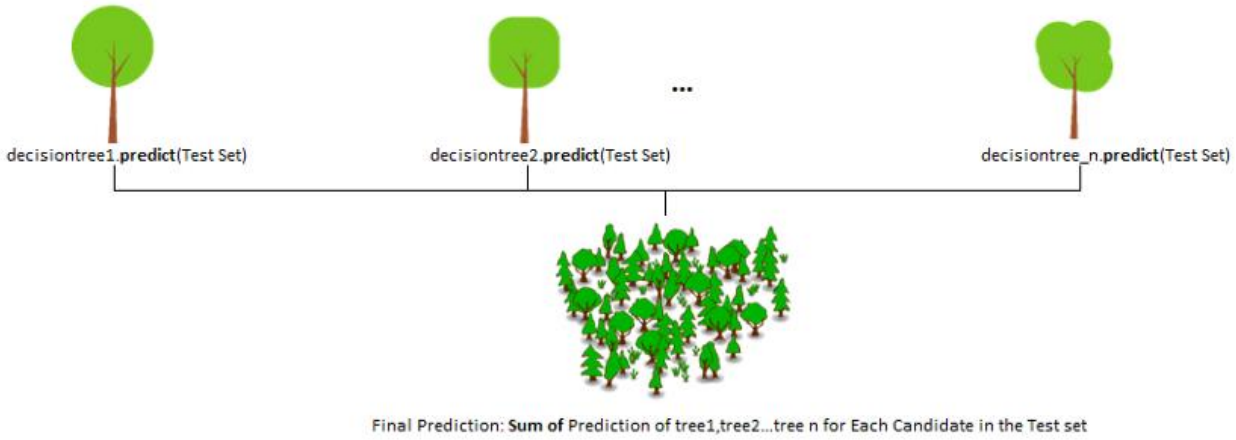
Gradient Boosting

از دیگر مدل های boosting است. به یاد داریم این نوع از مدل بر پایه یادگیری از روی اشتباهات پیشین بنا شده اند. این مدل مستقیما بر پایه خطاهایی که از درخت قبلی است، بر خلاف Adaboost که بر پایه بروز رسانی وزن هاست.



در مرحله یکم آموزش درخت تصمیم را داریم، سپس اعمال درخت تصمیمی که برای پیش بینی آموزش دیده، در مرحله بعد محاسبه residual error بعنوان y جدید. سپس تکرار مرحله یکم تا زمانیکه به تعداد درخت هایی که برای آموزش مد نظر هستند برسیم. در آخر پیش بینی نهایی که از جمع پیش بینی های درخت ها حاصل می شود.

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله



معیارهای ارزیابی مدل[5][4]

Accuracy

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Accuracy = (TP+TN) / (TP+FP+FN+TN)

از معیارهای اصلی ارزیابی مدل است و برای مسائل classification دو یا چند کلاسه مناسب است. Accuracy نسبت نتایج درست به کل نتایج بدست آمده است.

Recall

از دیگر معیارهای مهم در ارزیابی است. Recall به ما یادآوری می کند که چه نسبتی از مثبت واقعی بدرستی Classify شده است. آنچه Recall را در این پژوهش مهم تر می کند ارتباط آن با FN بطوریکه برای داشتن Recall ماکسیمم تا جای ممکن باید FN را کاهش داد.

Recall = (TP) / (TP+FN)

Precision

Precision نسبت مشاهدات مثبت درست پیش بینی شده به کل مشاهدات مثبت پیش بینی شده است . زمانی که بخواهیم از پیش بینی خود بسیار مطمئن باشیم، استفاده از Precision انتخابی مناسب از معیارهای ارزیابی است.

Precision = (TP) / (TP+FP)

F1-Score

F1 score عددی بین ۰ تا ۱ است و میانگین هارمونیک از Recall و Precision به ما می دهد.

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

$$F_1 = \frac{2}{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} = \frac{\mathrm{tp}}{\mathrm{tp} + \frac{1}{2}(\mathrm{fp} + \mathrm{fn})}$$

مروری بر مقاله و متد مورد استفاده

این مقاله با هدف پر کردن جای خالی داده های حاصل از الگوی رفتاری افراد در کنارسیستم مراقبت های بهداشتی سنتی ارائه شده است. از اینرو، استفاده از الگوریتم های یادگیری ماشین برای پردازش همزمان داده های حاصل از وضعیت سلامت افراد، سفرها همراه با سایر پارامترهای بیماران COVID-19 ، در ووهان ، برای پیش بینی نتیجه احتمالی بیمار ، براساس علائم ، سابقه سفر و تأخیر در گزارش موارد، را مد نظر قرار داده است.

• پردازش داده های مراقبت های بهداشتی و سفر با استفاده از الگوریتم های یادگیری ماشین به جای سیستم مراقبت های بهداشتی سنتی برای شناسایی فرد آلوده به COVID.

• با بکارگیری الگوریتم های مختلفی که برای پردازش داده های بیماران در دسترس هستند و مقایسه آنها Random Forest-Adaboost را به عنوان بهترین روش برای پردازش داده ها شناسایی کرده است.

• در عین حال، نویسندگان مقاله بیان می کنند، نتایج بدست آمده نیاز به مقایسه مجدد الگوریتم های موجود برای پردازش داده های بیمار COVID-19 را از بین می برد.

همچنین داده های استفاده شده در این مقاله بجهت تنوع راه را برای پیاده سازی الگوریتم ها با اضافه کردن داده های حاصل از scan بیماران (تصویری) در کنار سایر انواع داده ها باز می کند. که در اینصورت منتظر نتایج بهتر و دقیق تری می توان بود.

Classifier های استفاده شده:

SVM, Gaussian Naive Bayes, LR, Decision Tree, R. F. Adaboost

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

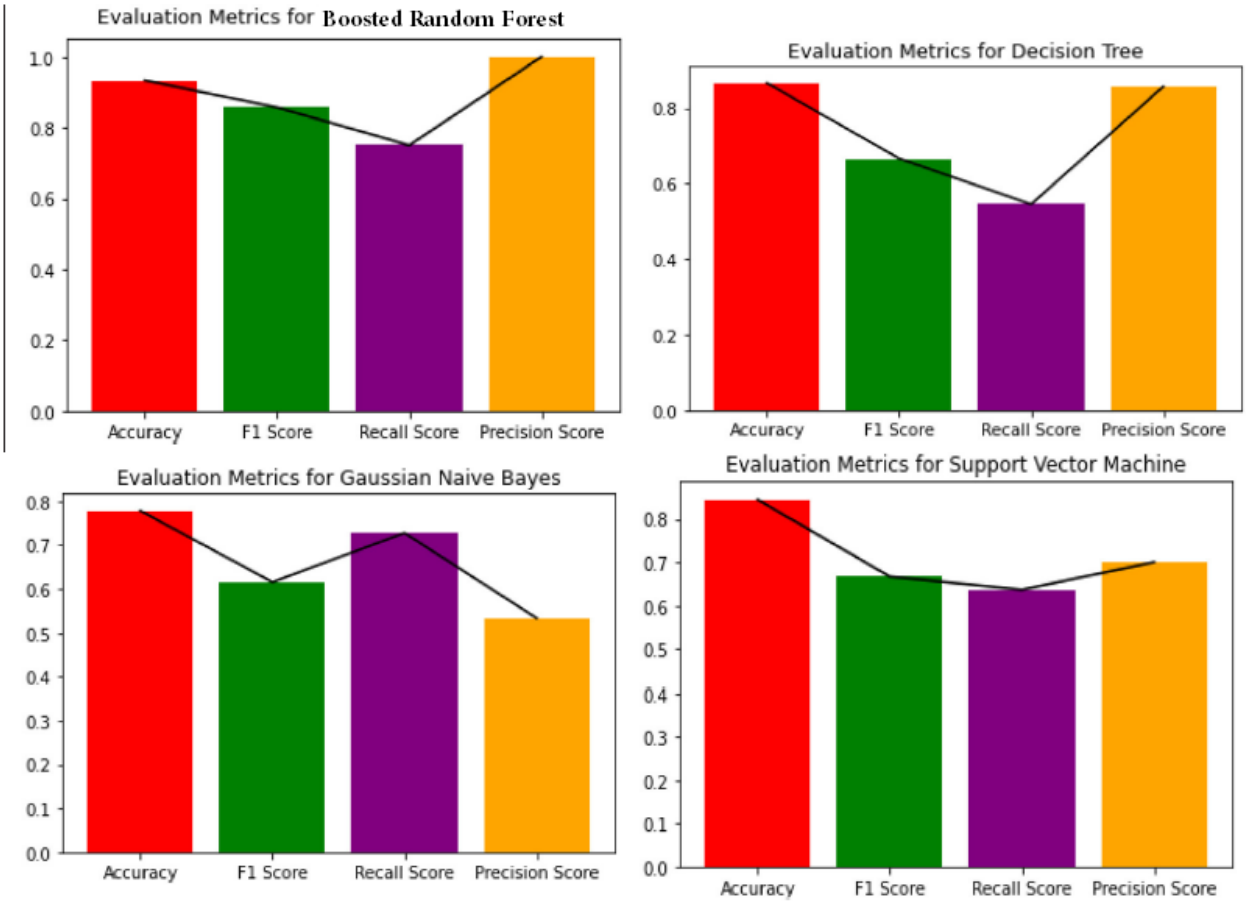
معیارهای ارزیابی مورد استفاده در مقاله شامل:

Accuracy, Recall, Precision و F1-Score است.

نتایج مقاله

نویسندگان مقاله پس از استفاده از classifierهای مختلف بر روی داده های حاصل از موقعیت جغرافیایی، مسافرت ها، وضعیت سلامت و شدت بیماری و زمان مراجعه به بیمارستان، به این نتیجه می رسد که در پیش بینی سلامت فرد (نسبت به بیمار Covid19)، استفاده از مدل Random Forest که با Adaboost classifier تقویت شده نتایج به مراتب بهتری (Accuracy حدود 94% و F1 score حدود 0.86) نسبت به سایر Classifier ها از خود نشان می دهد.

جداول زیر نتایج را نشان می دهند:


Evaluation Metrics for Boosted Random Forest
Accuracy
F1 Score
Recall Score
Precision Score
Evaluation Metrics for Decision Tree
Accuracy
F1 Score
Recall Score
Precision Score
Evaluation Metrics for Gaussian Naive Bayes
Accuracy
F1 Score
Recall Score
Precision Score
Evaluation Metrics for Support Vector Machine
Accuracy
F1 Score
Recall Score
Precision Score

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

دیتاست

دیتاست شامل ویژگی های زیر است:

Column	Description	Values (for categorical variables)	Type
id	Patient Id	NA	Numeric
location	The location where the patient belongs to	Multiple cities located throughout the world	String, Categorical
country	Patient's native country	Multiple countries	String, Categorical
gender	Patient's gender	Male, Female	String, Categorical
age	Patient's age	NA	Numeric
sym_on	The date patient started noticing the symptoms	NA	Date
hosp_vis	Date when the patient visited the hospital	NA	Date
vis_wuhan	Whether the patient visited Wuhan, China	Yes (1), No (0)	Numeric, Categorical
from_wuhan	Whether the patient belonged to Wuhan, China	Yes (1), No (0)	Numeric, Categorical
death	Whether the patient passed away due to COVID-19	Yes (1), No (0)	Numeric, Categorical
Recov	Whether the patient recovered	Yes (1), No (0)	Numeric, Categorical
symptom1. symptom2, symptom3, symptom4, symptom5, symptom6	Symptoms noticed by the patients	Multiple symptoms noticed by the patients	String, Categorical

دیتا ست شامل شش symptoms هست که بیشتر در بیماران دیده شده اند. این علائم شامل موارد زیر است:

Fever, cough, cold, fatigue, body pain, malaise

Pre Processing Data

دیتاست از انواع داده های مختلف چون Date، String، Numeric و Categorical تشکیل شده است. از آنجا که مدل ML نیاز دارد تمام داده هایی که به عنوان ورودی منتقل می شوند به شکل عددی باشند، بر روی داده های Categorical ، label-encoding انجام شده است بطوریکه به هر متغیر یونیک categorical یک عدد اختصاص داده شده است.

برای missed-values در دیتاست که بهنگام ارسال بعنوان ورودی موجب خطا می شوند مقدار "NA" در نظر گرفته شده است. همچنین دیتاست از ستونهایی در قالب "تاریخ" تشکیل شده است، از آنجا که تاریخ مستقیماً مورد استفاده قرار نمی گیرد، یک ستون جدید با نام (hosp_vis—sym_on) و مقدار عددی تشکیل شده است که مقادیر آن شامل تعداد روزهایی است که از مشاهده علائم توسط بیمار تا مراجعه به بیمارستان سپری شده است.

بهبود مقاله (Classifier پیشنهادی)

با توجه به ماهیت بیماری واگیردار و هزینه حاصل از عدم تشخیص فرد بیمار چرا که فرد در هر دو حالت Susceptible and Infected، خود عامل انتقال بیماری به افراد دیگر است و این امر به گسترش زنجیره انتقال بیماری کمک می کند. در اینجا لازم است به Reproductive Number اشاره کنیم که برای بیماری کووید ۱۹ در بهترین حالت R0 بین ۲و ۳ است که البته در فاز جهش یافته این مقدار حداقل بین ۴۰ تا ۸۰ درصد و در مواردی مقدار R0 بین ۴ و ۵ برآورد شده است.[9]

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

از سوی دیگر عدم توقف زنجیره انتقال موجب افزایش هزینه تامین منابع (بهداشتی و درمانی و تهیه واکسن) می شود این درحالیست که نمی توان هزینه حاصل از اعمال Lockdown و Quarantine را نادیده گرفت.

از این رو معیار Recall نسبت به Precision اولویت می یابد زیرا ما باید بدنبال بحداقل رساندن مقدار FN باشیم.

با استفاده از Gradient Boosting بعنوان Classifier این مطلوب رخ می دهد. نتایج بدست آمده نشان می دهد نه تنها Accuracy به ۰,۹۵ افزایش می یابد بلکه مقدار Recall از ۰,۷۵ در مقاله به ۰,۸۳ افزایش می یابد که در مقایسه با نتیجه حاصل از مقاله رشد قابل توجهی داشته است. افزایش اهمیت Recall در مقایسه با Precision ما را به سمت استفاده از Weighted-F1score هدایت می کند بطوریکه:

استفاده از F1score وزن دار، ضریب β اهمیت Recallبر Precision را بیان می کند.[8]

$$F = \frac{(\beta^2 + 1.0) \times P \times R}{\beta^2 \times P + R}$$

بطوریکه اگر:

β =1 باشد دو معیار Precision و Recall دارای وزن و اهمیت برابر هستند.

β =0.5 باشد معیار Recall دارای اهمیت نصف Precision است.

β =2 باشد معیار Recall دارای اهمیت دو برابر Precision است.

در ادامه جدول مقایسه نتایج پس از اعمال Classifier های مختلف را می بینید:

	Accuracy	Recall	Precision	F1score	Comments
LR	0.84	0.72	0.66	0.69	
D. Tree	0.86	0.54	0.85	0.66	با افزایش عمق درخت به ۳معیار Recall به ۰,۷۲ بهبود می یابد
R.F. (Adaboost)	0.93	0.75	1.0	0.85	با افزایش test.size=0.3 حدود ۰,۲ بهبود در نتایج داریم.
SVM	0.84	0.63	0.70	0.66	
Guassian NB	0.77	0.72	0.53	0.61	
R. F. (Grad. Boosting)	0.95	0.83	1.0	0.90	Suggested Classifier

کارهای آتی (Future Works)

بعنوان ادامه پژوهش در این حوزه موارد زیر قابل تامل می نماید که در صورت نظر مثبت استاد محترم بر ادامه این پژوهش ممارست می ورزم:

- **استفاده از داده های تصویر (اسکن ریه و ...) در کنار سایر داده ها در جهت بهبود نتایج بدست آمده**

- **شناسایی Supper spreaderها با استفاده از Centrality Measuresو ایمن سازی جامعه (با توجه به محدودیت منابع)**
- **اعمال مدل MLP در کنار سایر Classifiers در جهت بهبود نتایج**

دیگر منابع مورد استفاده

---

[1] What Is a Pandemic? David M. Morens, Gregory K. Folkers, Anthony S. Fauci The Journal of Infectious Diseases, Volume 200, Issue 7, 1 October 2009, Pages 1018–1021, https://doi.org/10.1086/644537

[2] M. E. J. Newman, The structure and function of complex networks, SIAM Review 45, (2003), pp. 167-256.

[3] T. Zhou, F. Zhongqian, B. Wang, Epidemic dynamics on complex networks, Progress in Natural Science, (2006), pp. 452-457.

[4] Vaid S, Kalantar R, Bhandari M. Deep learning COVID-19 detection bias: accuracy through artificial intelligence. Int Orthop. (2020);44(8):1539-1542. doi: 10.1007/s00264-020-04609-7. Epub 2020 May 27. PMID: 32462314; PMCID: PMC7251557.

[5] Petrova, Tatiana & Soshnikov, Dmitri & Grunin, Andrey. (2020). Estimation of Time-Dependent Reproduction Number for Global COVID-19 Outbreak. 10.20944/preprints 202006.0289. v1.

[6] Alessandro Annunziatio, Tommi Asikainen, Effective Reproduction Number Estimation from Data Series, EUR 30300 EN, Publications Office of the European Union, Luxembourg, 2020, ISBN 978-92-76-20749-8, doi:10.2760/036156, JRC121343

[7] Linka, K., Peirlinck, M. & Kuhl, E. (2020). The reproduction number of COVID-19 and its correlation with public health interventions. Comput Mech 66,1035–1050.springer link: https://doi.org/10.1007/ s00466-020-01880-8

[8] Chinchor, N., (1992). Evaluation metrics. Message Understanding Conference(MUC). PP. 22–29 https://doi.org/10.3115/1072064.1072067

[9] A. Annunziato, T. Asikainen, (2020). Effective Reproduction Number Estimation from Data Series. EU Science Hub, doi:10.2760/036156.

پیوست- توضیح کدهای پیاده سازی

Import کتابخانه ها و ماژول های مورد استفاده

```
import numpy as np
import pandas as pd
import datetime as dt
import sklearn
from scipy import stats
from sklearn import preprocessing
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import recall_score as rs
from sklearn.metrics import precision_score as ps
from sklearn.metrics import f1_score as fs
from sklearn.metrics import log_loss
```

Data Pre Processing

دیتاست از انواع داده های مختلف چون Date، String، Numeric و Categorical تشکیل شده است. از آنجا که مدل ML نیاز دارد تمام داده هایی که به عنوان ورودی منتقل می شوند به شکل عددی باشند، بر روی داده های Categorical ، label-encoding انجام شده است بطوریکه به هر متغیر یونیک categorical یک عدد اختصاص داده شده است.

برای missed-values در دیتاست که بهنگام ارسال بعنوان ورودی موجب خطا می شوند مقدار "NA" در نظر گرفته شده است. همچنین دیتاست از ستونهایی در قالب "تاریخ" تشکیل شده است، از آنجا که تاریخ مستقیماً مورد استفاده قرار نمی گیرد، یک ستون

جدید با نام (hosp_vis—sym_on) و مقدار عددی تشکیل شده است که مقادیر آن شامل تعداد روزهایی است که از مشاهده علائم توسط بیمار تا مراجعه به بیمارستان سپری شده است.

```
## Data Pre processing:
data = pd.read_csv('data.csv')
data = data.drop('id',axis=1)
data = data.fillna(np.nan,axis=0)
data['location'] = encoder.fit_transform(data['location'].astype(str))
data['country'] = encoder.fit_transform(data['country'].astype(str))
data['gender'] = encoder.fit_transform(data['gender'].astype(str))
data[['symptom1']] = encoder.fit_transform(data['symptom1'].astype(str))
data[['symptom2']] = encoder.fit_transform(data['symptom2'].astype(str))
data[['symptom3']] = encoder.fit_transform(data['symptom3'].astype(str))
data[['symptom4']] = encoder.fit_transform(data['symptom4'].astype(str))
data[['symptom5']] = encoder.fit_transform(data['symptom5'].astype(str))
data[['symptom6']] = encoder.fit_transform(data['symptom6'].astype(str))
```

```
[3]  data['sym_on'] = pd.to_datetime(data['sym_on'])
     data['hosp_vis'] = pd.to_datetime(data['hosp_vis'])
     data['sym_on']= data['sym_on'].map(dt.datetime.toordinal)
     data['hosp_vis']= data['hosp_vis'].map(dt.datetime.toordinal)
     data['diff_sym_hos']= data['hosp_vis'] - data['sym_on']
```

```
[4]  data['diff_symp_hos'] = data['hosp_vis']-data['sym_on']
```

```
[5]  data = data.drop(['sym_on','hosp_vis'],axis=1)
```

```
[6]  print(data.dtypes)
```

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

برای دیتاست train هم مراحل Pre Process را انجام می دهیم:

```
## Pre Processing Train Data for Training:
tdata = pd.read_csv('train.csv')
print(tdata.head())
```

```
tdata = pd.read_csv('train.csv')
tdata = tdata.drop('id',axis=1)
tdata = tdata.fillna(np.nan,axis=0)
tdata['age'] = tdata['age'].fillna(value=tdata['age'].mean())
tdata['location'] = encoder.fit_transform(tdata['location'].astype(str))
tdata['country'] = encoder.fit_transform(tdata['country'].astype(str))
tdata['gender'] = encoder.fit_transform(tdata['gender'].astype(str))
tdata[['symptom1']] = encoder.fit_transform(tdata['symptom1'].astype(str))
tdata[['symptom2']] = encoder.fit_transform(tdata['symptom2'].astype(str))
tdata[['symptom3']] = encoder.fit_transform(tdata['symptom3'].astype(str))
tdata[['symptom4']] = encoder.fit_transform(tdata['symptom4'].astype(str))
tdata[['symptom5']] = encoder.fit_transform(tdata['symptom5'].astype(str))
tdata[['symptom6']] = encoder.fit_transform(tdata['symptom6'].astype(str))
```

```
[9] tdata['sym_on'] = pd.to_datetime(tdata['sym_on'])
    tdata['hosp_vis'] = pd.to_datetime(tdata['hosp_vis'])
    tdata['sym_on']= tdata['sym_on'].map(dt.datetime.toordinal)
    tdata['hosp_vis']= tdata['hosp_vis'].map(dt.datetime.toordinal)
    tdata['diff_sym_hos']= tdata['hosp_vis'] - tdata['sym_on']
```

```
[10] tdata = tdata.drop(['sym_on','hosp_vis'],axis=1)
     print(tdata)
```

```
[11] print(tdata.isna().sum())
```

Import معیارهای ارزیابی(Accuracy, Recall, Precision,F1score, Confusion matrics)

```
[12] ## import Evaluation Metrics:

     from sklearn.metrics import recall_score as rs
     from sklearn.metrics import precision_score as ps
     from sklearn.metrics import f1_score as fs
     from sklearn.metrics import balanced_accuracy_score as bas
     from sklearn.metrics import confusion_matrix as cm
```

اعمال Classifierهای مختلف و مشاهده نتایج ارزیابی مدل بر اساس هر Classifier

Logistic Regression

```
## Logistic Regression:
from sklearn.linear_model import LogisticRegression as lr

classifier = lr()
```

```
[14] X = tdata[['location','country','gender','age','vis_wuhan','from_wuhan','symptom1',
                'symptom2','symptom3','symptom4','symptom5','symptom6','diff_sym_hos']]
     Y = tdata['death']
```

```
[15] X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,random_state=10)
     classifier.fit(X_train,np.array(Y_train).reshape(Y_train.shape[0],1))
```

اعمال معیارهای ارزیابی مدل

```
[16] ## Evaluation of the Model:

     pred = np.array(classifier.predict(X_test))
     recall_lr = rs(Y_test,pred)
     precision_lr = ps(Y_test,pred)
     f1_lr = fs(Y_test,pred)
     ma_lr = classifier.score(X_test,Y_test)
```

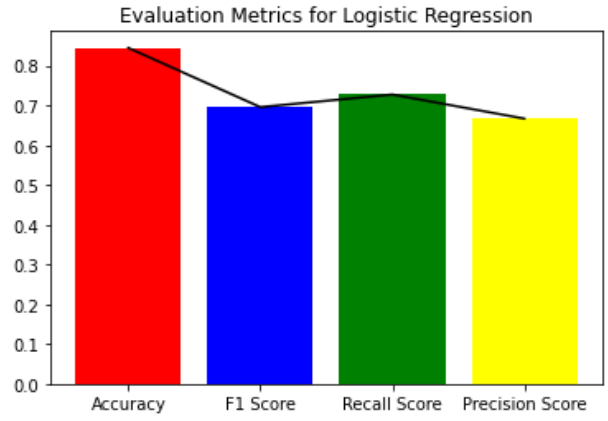
نتایج حاصل برای Logistic Regression

```
print('*** Evaluation metrics for test dataset ***\n')
print('Recall Score: ',recall_lr)
print('Precision Score: ',precision_lr)
print('F1 Score: ',f1_lr)
print('Accuracy: ',ma_lr)
a = pd.DataFrame(Y_test)
a['pred']= classifier.predict(X_test)
print('\n\tTable \n')
print(a.head())
```

```
*** Evaluation metrics for test dataset ***

Recall Score:  0.7272727272727273
Precision Score:  0.6666666666666666
F1 Score:  0.6956521739130435
Accuracy:  0.8444444444444444

        Table

     death  pred
184      0     0
170      1     0
142      0     0
182      0     0
49       1     1
```

```
import matplotlib.pyplot as plt

plt.bar(['Accuracy','F1 Score','Recall Score','Precision Score'],
        [ma_lr,f1_lr,recall_lr,precision_lr],color=['red','Blue','green','Yellow'])
plt.plot([ma_lr,f1_lr,recall_lr,precision_lr],color='black')
plt.title('Evaluation Metrics for Logistic Regression')
```

```
Text(0.5, 1.0, 'Evaluation Metrics for Logistic Regression')
```



مشاهده نتایج ارزیابی مدل در قالب نمودار

ساخت prediction

```
print(pd.DataFrame({'Val':Y_test,'Pred':classifier.predict(X_test)}))
```

```
     Val  Pred
184    0     0
170    1     0
142    0     0
182    0     0
49     1     1
117    0     0
63     1     0
144    0     0
35     1     1
101    0     0
24     1     1
200    0     0
129    0     0
26     1     1
116    0     0
76     0     0
99     0     0
47     1     1
70     0     0
121    0     0
146    0     0
220    1     1
60     0     1
188    0     0
```

Decision Tree

```
[20] ## Decision Tree Classifier:

     from sklearn.tree import DecisionTreeClassifier as dtc
     classifier = dtc(max_depth=2)
```

```
X = tdata[['location','country','gender','age','vis_wuhan','from_wuhan',
           'symptom1','symptom2','symptom3','symptom4','symptom5','symptom6','diff_sym_hos']]
Y = tdata['death']
```

```
[22] X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,random_state=10)
     classifier.fit(X_train,np.array(Y_train).reshape(Y_train.shape[0],1))
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=2, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

ارزیابی مدل و نتایج حاصل از آن

```
[23] ## Evaluation of the Model:

     pred = np.array(classifier.predict(X_test))

     recall_dtc = rs(Y_test,pred)
     precision_dtc = ps(Y_test,pred)
     f1_dtc = fs(Y_test,pred)
     ma_dtc = classifier.score(X_test,Y_test)
```

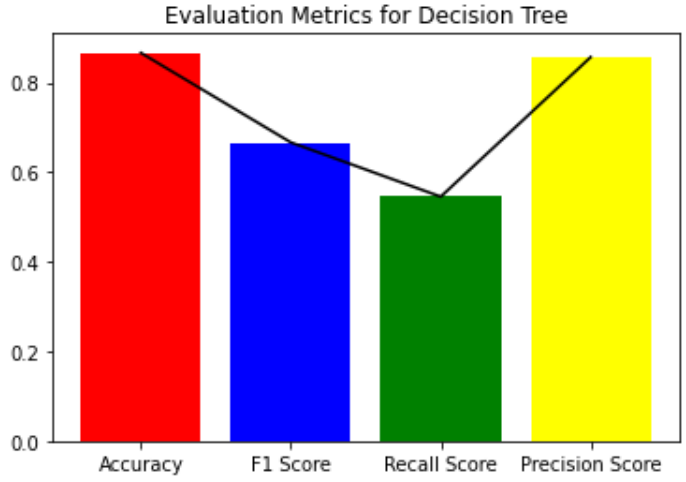
```
print('*** Evaluation metrics for test dataset ***\n')
print('Recall Score: ',recall_dtc)
print('Precision Score: ',precision_dtc)
print('F1 Score: ',f1_dtc)
print('Accuracy: ',ma_dtc)
a = pd.DataFrame(Y_test)
a['pred']= classifier.predict(X_test)
print('\n\tTable \n')
print(a.head())
```

```
*** Evaluation metrics for test dataset ***

Recall Score:  0.5454545454545454
Precision Score:  0.8571428571428571
F1 Score:  0.6666666666666665
Accuracy:  0.8666666666666667

            Table

        death  pred
    184      0     0
    170      1     0
    142      0     0
    182      0     0
    49       1     0
```

مشاهده نتایج حاصل از ارزیابی در قالب نمودار

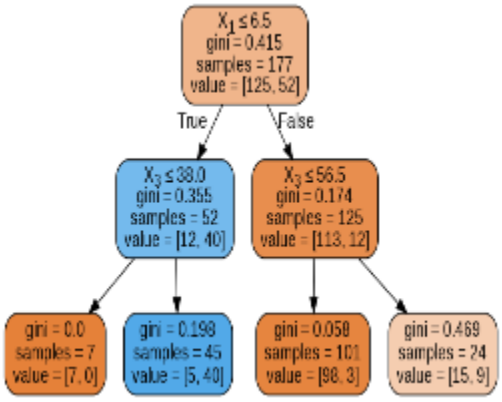
```
import matplotlib.pyplot as plt

plt.bar(['Accuracy','F1 Score','Recall Score','Precision Score'],
        [ma_dtc,f1_dtc,recall_dtc,precision_dtc],color=['red','Blue','green','Yellow'])
plt.plot([ma_dtc,f1_dtc,recall_dtc,precision_dtc],color='black')
plt.title('Evaluation Metrics for Decision Tree')
```

```
Text(0.5, 1.0, 'Evaluation Metrics for Decision Tree')
```



ساخت درخت تصمیم

```
[26] ## Decision Tree Visualization:

     classifier.fit(X_train,np.array(Y_train).reshape(Y_train.shape[0],1))
     from sklearn.externals.six import StringIO
     from IPython.display import Image
     from sklearn.tree import export_graphviz
     import pydotplus

     estimator = classifier
     dot_data = StringIO()
     export_graphviz(estimator, out_file=dot_data,
                     filled=True, rounded=True,
                     special_characters=True)
     graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
     Image(graph.create_png(),width=250,height=200)
```



SVM Classifier

```
[27] ## SVM Classifier:

     from sklearn import svm
     classifier = svm.SVC()
```

```
X = tdata[['location','country','gender','age','vis_wuhan','from_wuhan',
           'symptom1','symptom2','symptom3','symptom4','symptom5','symptom6','diff_sym_hos']]
Y = tdata['death']
```

```
[29] X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,random_state=10)
     classifier.fit(X_train,np.array(Y_train).reshape(Y_train.shape[0],1))
```

اعمال معیارهای ارزیابی و نتایج حاصل از آن بر روی مدل

```
## Evaluation of the Model:

pred = np.array(classifier.predict(X_test))

recall_svm = rs(Y_test,pred)
precision_svm = ps(Y_test,pred)
f1_svm = fs(Y_test,pred)
ma_svm = classifier.score(X_test,Y_test)
```

```
[31] print('*** Evaluation metrics for test dataset ***\n')
     print('Recall Score: ',recall_svm)
     print('Precision Score: ',precision_svm)
     print('F1 Score: ',f1_svm)
     print('Accuracy: ',ma_svm)
     a = pd.DataFrame(Y_test)
     a['pred']= classifier.predict(X_test)
     print('\n\tTable \n')
     print(a.head())
```

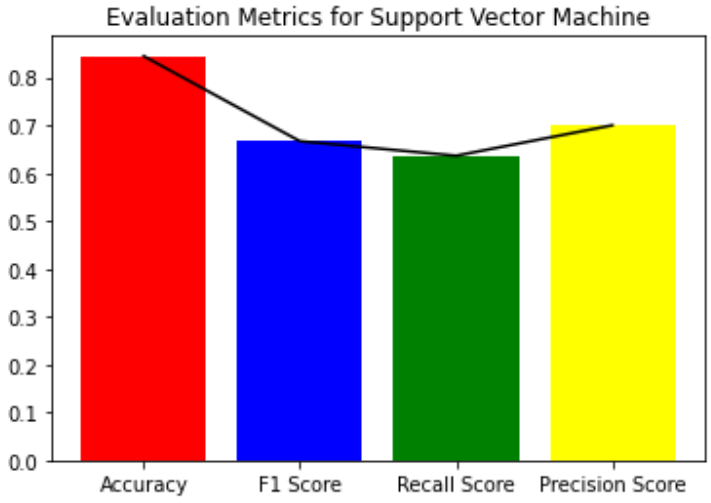
```
*** Evaluation metrics for test dataset ***

Recall Score:  0.6363636363636364
Precision Score:  0.7
F1 Score:  0.6666666666666666
Accuracy:  0.8444444444444444
```

```
Table

      death  pred
184       0     0
170       1     0
142       0     0
182       0     0
49        1     1
```

مشاهده نتایج در قالب نمودار

```
import matplotlib.pyplot as plt

plt.bar(['Accuracy','F1 Score','Recall Score','Precision Score'],
        [ma_svm,f1_svm,recall_svm,precision_svm],color=['red','Blue','green','Yellow'])
plt.plot([ma_svm,f1_svm,recall_svm,precision_svm],color='black')
plt.title('Evaluation Metrics for Support Vector Machine')
```

```
Text(0.5, 1.0, 'Evaluation Metrics for Support Vector Machine')
```



Gaussian Naive Bayes

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

```
## Gaussian Naive Bayes:

from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
```

```
[34] X = tdata[['location','country','gender','age','vis_wuhan',
                'from_wuhan','symptom1','symptom2','symptom3','symptom4','symptom5','symptom6','diff_sym_hos']]
     Y = tdata['death']
```

```
[35] X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,random_state=10)
     classifier.fit(X_train,np.array(Y_train).reshape(Y_train.shape[0],1))
```

ارزیابی مدل و نتایج حاصل از آن

```
## Evaluation of the Model:

pred = np.array(classifier.predict(X_test))

recall_gnb = rs(Y_test,pred)
precision_gnb = ps(Y_test,pred)
f1_gnb = fs(Y_test,pred)
ma_gnb = classifier.score(X_test,Y_test)
```

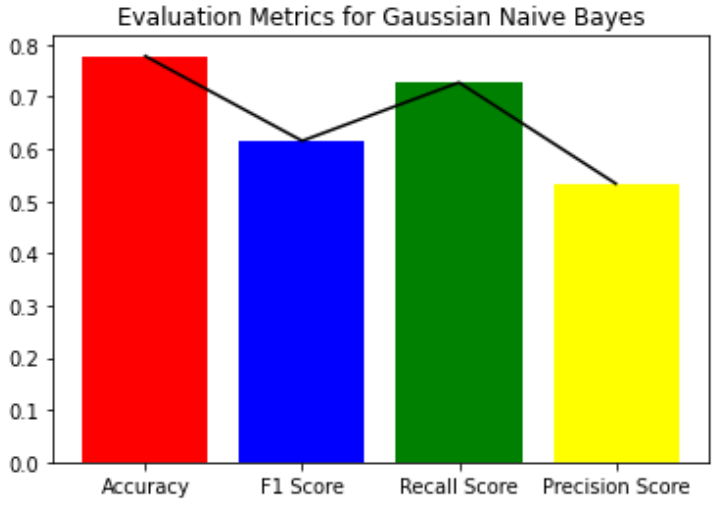
```
print('*** Evaluation metrics for test dataset ***\n')
print('Recall Score: ',recall_gnb)
print('Precision Score: ',precision_gnb)
print('F1 Score: ',f1_gnb)
print('Accuracy: ',ma_gnb)
a = pd.DataFrame(Y_test)
a['pred']= classifier.predict(X_test)
print('\n\tTable \n')
print(a.head())
```

```
*** Evaluation metrics for test dataset ***

Recall Score:  0.7272727272727273
Precision Score:  0.5333333333333333
F1 Score:  0.6153846153846153
Accuracy:  0.7777777777777778


         Table

      death  pred
184       0     0
170       1     0
142       0     0
182       0     0
49        1     1
```

مشاهده نتایج در قالب نمودار

```
import matplotlib.pyplot as plt

plt.bar(['Accuracy','F1 Score','Recall Score','Precision Score'],
        [ma_gnb,f1_gnb,recall_gnb,precision_gnb],color=['red','Blue','green','Yellow'])
plt.plot([ma_gnb,f1_gnb,recall_gnb,precision_gnb],color='black')
plt.title('Evaluation Metrics for Gaussian Naive Bayes')
```

```
Text(0.5, 1.0, 'Evaluation Metrics for Gaussian Naive Bayes')
```



Boosted Random Forest (Adaboost)

```
## Boosted Random Forest(Adaboost):
from sklearn.metrics import recall_score as rs
from sklearn.metrics import precision_score as ps
from sklearn.metrics import f1_score as fs
from sklearn.metrics import balanced_accuracy_score as bas
from sklearn.metrics import confusion_matrix as cm
```

```
[40] rf = RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                                 criterion='gini', max_depth=2, max_features='auto',
                                 max_leaf_nodes=None, max_samples=None,
                                 min_impurity_decrease=0.0, min_impurity_split=None,
                                 min_samples_leaf=2, min_samples_split=2,
                                 min_weight_fraction_leaf=0.0, n_estimators=100,
                                 n_jobs=None, oob_score=False, random_state=None,
                                 verbose=0, warm_start=False)
     classifier = AdaBoostClassifier(rf,50,0.01,'SAMME.R',10)
```

```
[41] X = tdata[['location','country','gender','age','vis_wuhan','from_wuhan','symptom1',
                'symptom2','symptom3','symptom4','symptom5','symptom6','diff_sym_hos']]
     Y = tdata['death']
```

```
[42] X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,random_state=0)
     classifier.fit(X_train,np.array(Y_train).reshape(Y_train.shape[0],1))
```

```
AdaBoostClassifier(algorithm='SAMME.R',
                   base_estimator=RandomForestClassifier(bootstrap=True,
                                                         ccp_alpha=0.0,
                                                         class_weight=None,
                                                         criterion='gini',
                                                         max_depth=2,
                                                         max_features='auto',
                                                         max_leaf_nodes=None,
                                                         max_samples=None,
                                                         min_impurity_decrease=0.0,
                                                         min_impurity_split=None,
                                                         min_samples_leaf=2,
                                                         min_samples_split=2,
                                                         min_weight_fraction_leaf=0.0,
                                                         n_estimators=100,
                                                         n_jobs=None,
                                                         oob_score=False,
                                                         random_state=None,
                                                         verbose=0,
                                                         warm_start=False),
                   learning_rate=0.01, n_estimators=50, random_state=10)
```

اعمال معیارهای ارزیابی مدل و نتایج حاصل از آن

```
## Evaluation of the Model:

pred = np.array(classifier.predict(X_test))

recall = rs(Y_test,pred)
precision = ps(Y_test,pred)
f1 = fs(Y_test,pred)
ma = classifier.score(X_test,Y_test)
```

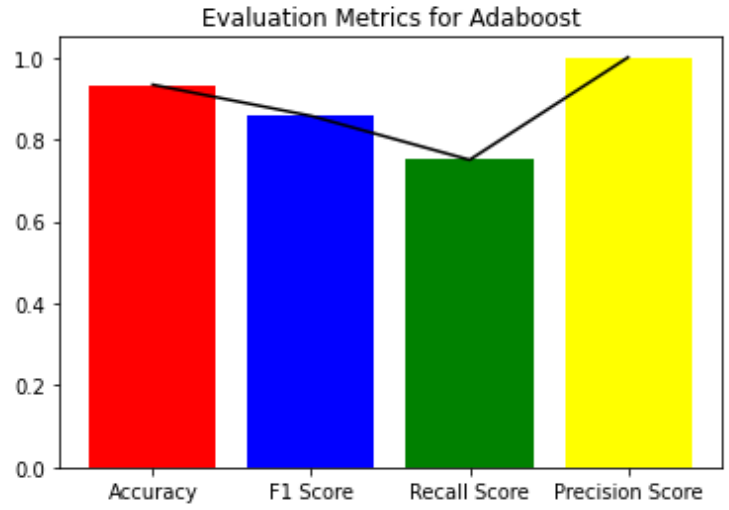
```
print('*** Evaluation metrics for test dataset ***\n')
print('Recall Score: ',recall)
print('Precision Score: ',precision)
print('F1 Score: ',f1)
print('Accuracy: ',ma)
a = pd.DataFrame(Y_test)
a['pred']= classifier.predict(X_test)
print('\n\tTable \n')
print(a.head())
```

```
*** Evaluation metrics for test dataset ***

Recall Score:  0.75
Precision Score:  1.0
F1 Score:  0.8571428571428571
Accuracy:  0.9333333333333333


         Table

      death  pred
130       0     0
203       0     0
170       1     0
66        0     0
181       0     0
```

مشاهده نتایج در قالب نمودار

```
import matplotlib.pyplot as plt

plt.bar(['Accuracy','F1 Score','Recall Score','Precision Score'],
        [ma,f1,recall,precision],color=['red','blue','green','Yellow'])
plt.plot([ma,f1,recall,precision],color='black')
plt.title('Evaluation Metrics for Adaboost')
```

```
Text(0.5, 1.0, 'Evaluation Metrics for Adaboost')
```



Boosting Random Forest (with Gradient Boosting)

```
## Random Forest with Gradient Boosting:
from sklearn.metrics import recall_score as rs
from sklearn.metrics import precision_score as ps
from sklearn.metrics import f1_score as fs
from sklearn.metrics import balanced_accuracy_score as bas
from sklearn.metrics import confusion_matrix as cm
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
```

```
rf = RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=2, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=2, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
classifier = GradientBoostingClassifier(n_estimators=100)
```

```
X = tdata[['location','country','gender','age','vis_wuhan','from_wuhan',
           'symptom1','symptom2','symptom3','symptom4','symptom5','symptom6','diff_sym_hos']]
Y = tdata['death']
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,random_state=0)
classifier.fit(X_train,np.array(Y_train).reshape(Y_train.shape[0],1))
```

```
GradientBoostingClassifier(ccp_alpha=0.0, criterion='friedman_mse', init=None,
                           learning_rate=0.1, loss='deviance', max_depth=3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_impurity_split=None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=100,
                           n_iter_no_change=None, presort='deprecated',
                           random_state=None, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
```

گزارش پروژه درس یادگیری ماشین: مطالعه و بررسی، پیاده سازی و ارائه روشی جهت بهبود نتایج مقاله

اعمال معیارهای ارزیابی و نتایج حاصل از آن

```
## Evaluation of the Model:

pred = np.array(classifier.predict(X_test))

recall_rfgb = rs(Y_test,pred)
precision_rfgb = ps(Y_test,pred)
f1_rfgb = fs(Y_test,pred)
ma_rfgb = classifier.score(X_test,Y_test)
```

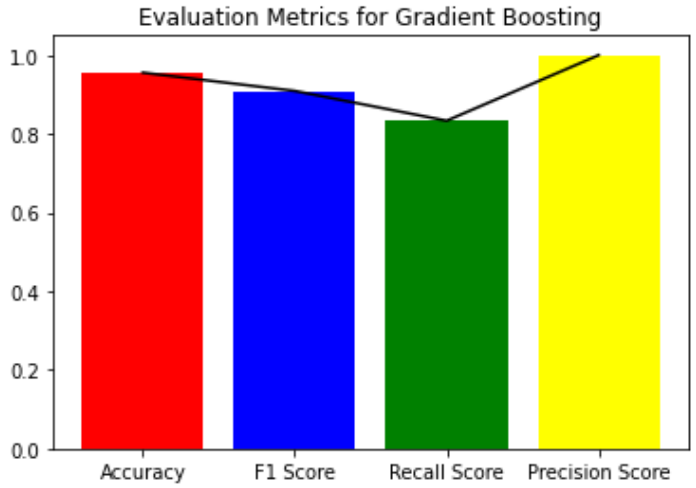
```
print('*** Evaluation metrics for test dataset ***\n')
print('Recall Score: ',recall_rfgb)
print('Precision Score: ',precision_rfgb)
print('F1 Score: ',f1_rfgb)
print('Accuracy: ',ma_rfgb)
a = pd.DataFrame(Y_test)
a['pred']= classifier.predict(X_test)
print('\n\tTable \n')
print(a.head())
```

```
*** Evaluation metrics for test dataset ***

Recall Score:  0.8333333333333334
Precision Score:  1.0
F1 Score:  0.9090909090909091
Accuracy:  0.9555555555555556


        Table

     death  pred
130      0     0
203      0     0
170      1     1
66       0     0
181      0     0
```

مشاهده نتایج ارزیابی برای Gradient Boosting در قالب نمودار

```
import matplotlib.pyplot as plt

plt.bar(['Accuracy','F1 Score','Recall Score','Precision Score'],
        [ma_rfgb,f1_rfgb,recall_rfgb,precision_rfgb],color=['red','blue','green','Yellow'])
plt.plot([ma_rfgb,f1_rfgb,recall_rfgb,precision_rfgb],color='black')
plt.title('Evaluation Metrics for Gradient Boosting')
```

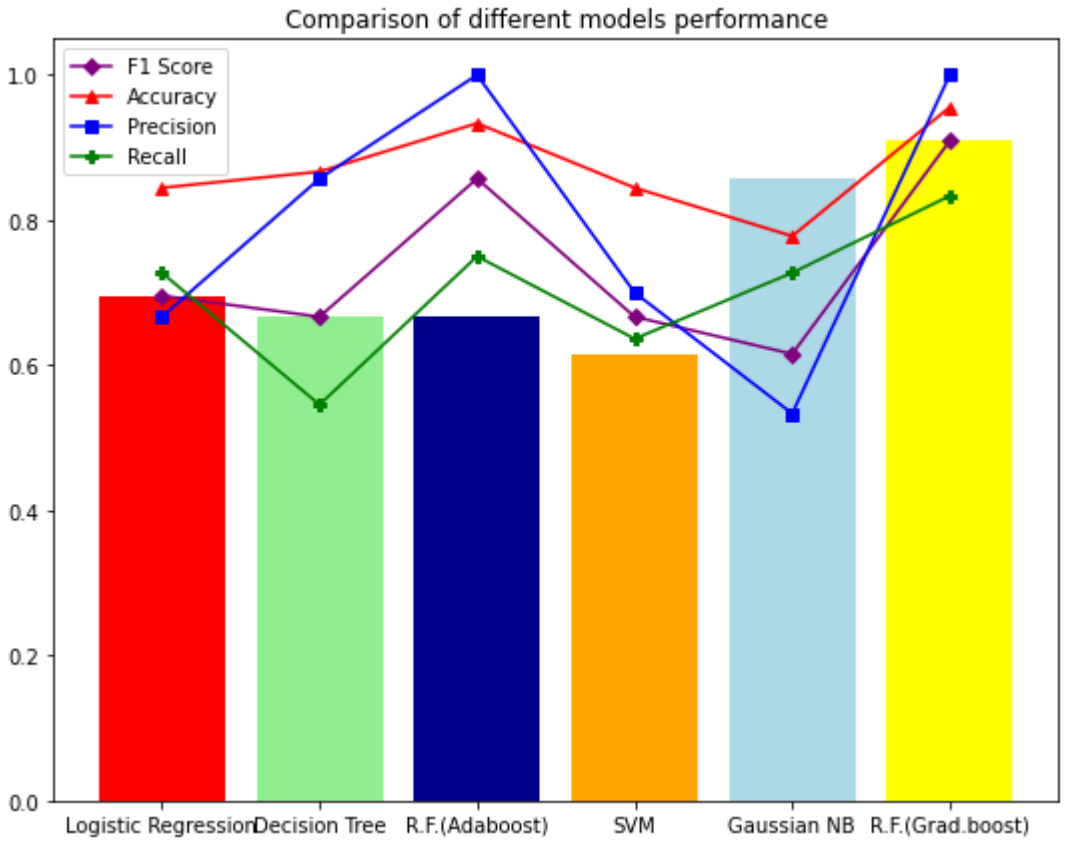
```
Text(0.5, 1.0, 'Evaluation Metrics for Gradient Boosting')
```



مقایسه نتایج ارزیابی بدست آمده برای تمام مدل ها در قالب نمودار

```
## Comparison of Evaluation Metrics for different Models:

import matplotlib.pyplot as plt
fig = plt.figure(figsize=(11,9))
plt.bar(['Logistic Regression','Decision Tree','Random Forest(Adaboost)',
         'SVM','Gaussian NB','Random Forest(Grad.boost)'],
        [f1_lr,f1_dtc,f1_svm,f1_gnb,f1,f1_rfgb],color=['red','green','purple','orange','Blue','Yellow'])
plt.plot(['Logistic Regression','Decision Tree','Random Forest(Adaboost)',
          'SVM','Gaussian NB','Random Forest(Grad.boost)'],
         [f1_lr,f1_dtc,f1,f1_svm,f1_gnb,f1_rfgb],color='purple',marker='D')
plt.plot(['Logistic Regression','Decision Tree','Random Forest(Adaboost)',
          'SVM','Gaussian NB','Random Forest(Grad.boost)'],
         [ma_lr,ma_dtc,ma,ma_svm,ma_gnb,ma_rfgb],color='red',marker='^')
plt.plot(['Logistic Regression','Decision Tree','Random Forest(Adaboost)',
          'SVM','Gaussian NB','Random Forest(Grad.boost)'],
         [precision_lr,precision_dtc,precision,precision_svm,precision_gnb,precision_rfgb],color='blue',marker='s')
plt.plot(['Logistic Regression','Decision Tree','Random Forest(Adaboost)',
          'SVM','Gaussian NB','Random Forest(Grad.boost)'],
         [recall_lr,recall_dtc,recall,recall_svm,recall_gnb,recall_rfgb],color='green',marker='P')
plt.legend(('F1 Score','Accuracy','Precision','Recall'))
plt.title('Comparison of different models performance')

plt.show(fig)
```

مقایسه نتایج حاصل از ارزیابی تمام مدل ها در قالب نمودار



	Accuracy	Recall	Precision	F1score	Comments
LR	0.84	0.72	0.66	0.69	
D. Tree	0.86	0.54	0.85	0.66	با افزایش عمق درخت به ۳معیار Recall به ۰,۷۲ بهبود می یابد
R.F. (Adaboost)	0.93	0.75	1.0	0.85	با افزایش test.size=0.3 حدود ۰,۲ بهبود در نتایج داریم.
SVM	0.84	0.63	0.70	0.66	

Guassian NB	0.77	0.72	0.53	0.61	
R. F. (Grad. Boosting)	0.95	0.83	1.0	0.90	Suggested Classifier