گزارش تمرین شماره ۲ درس داده کاوی

اساتید گرامی :

جناب آقای دکتر فراهانی و جناب آقای دکتر خردپیشه

دستیار آموزشی: جناب آقای شریفی

گردآورنده: هدیه آشوری ۹۹۴۲۲۰۲۲

۱۴۰۰/۰۱/۲۰

در ابتدا پکیج های panadas,numpy, matplotlib, seaborn را import می کنم و سپس داده ها را فراخوانی می کنم

بعد به بررسی داده ها می پردازم بدین صورت که با دستور df.shape میزان سطرها و ستون جدول اطلاعات را به دست آورده و با دستور df.types نوع داده ها را مشاهده کردم و سپس با دستور df.describe محاسبات مختلفی از داده را رویت کردم.

داده های جدول به شرح ذیل است

- age - age in years
- sex - (1 = male; 0 = female)
- cp - chest pain type
- trestbps - resting blood pressure (in mm Hg on admission to the hospital)
- chol - serum cholestoral in mg/dl
- fbs - (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- restecg - resting electrocardiographic results
- thalach - maximum heart rate achieved
- exang - exercise induced angina (1 = yes; 0 = no)
- oldpeak - ST depression induced by exercise relative to rest
- slope - the slope of the peak exercise ST segment
- ca - number of major vessels (0-3) colored by flourosopy
- thal - 3 = normal; 6 = fixed defect; 7 = reversable defect
- target - have disease or not (1=yes, 0=no)

۱.الف) آیا داده پرت در دیتاست وجود دارد ؟در صورت وجود آنها را حذف کنید.

وجود داده پرت در دیتاست می تواند باعث ایجاد مشکل و دریافت نتیجه ناصحیح گردد لذا پیدا کرده داده های outlier از اهمیت ویژه ای برخوردار است

شناسایی ناهنجاری و نقاط پرت (Outlier) را می توان در دو بخش انجام دهیم .

بخش اول: در مجموعه داده با مقادیری به صورت تک متغیری یا یک بعدی به دنبال مشاهدات ناهنجار بگردیم.

بخش دوم: این کار را به کمک داده‌هایی چند بعدی از همان مجموعه داده، انجام دهیم .

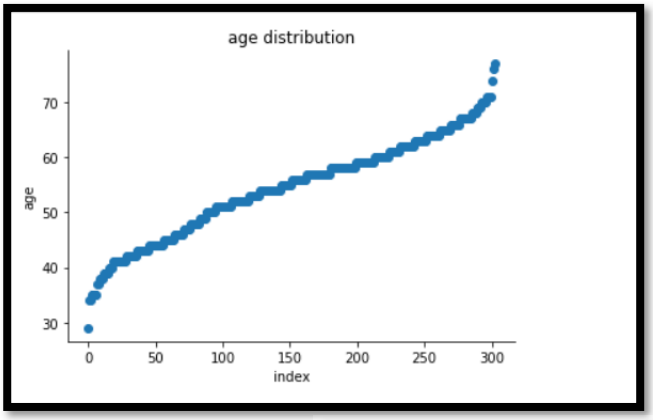
همچنین محاسبه **ضریب چولگی (Skewness)** و **کشیدگی (Kurtosis)** نیز برای نمایش رفتار داده‌ها مناسب است. این شاخص‌ها میزان انحراف از تقارن نسبت به توزیع نرمال را اندازه‌گیری می‌کنند .

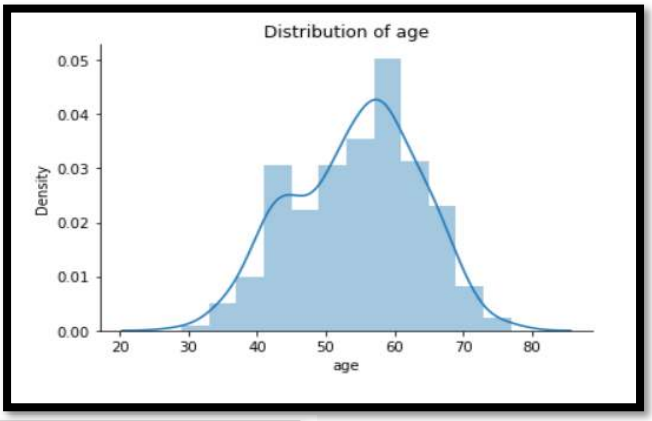
بخش سوم: یک الگوریتم برای شناسایی ناهنجاری به نام الگوریتم جنگل ایزوله (Isolation Forest) نیز وجود دارد

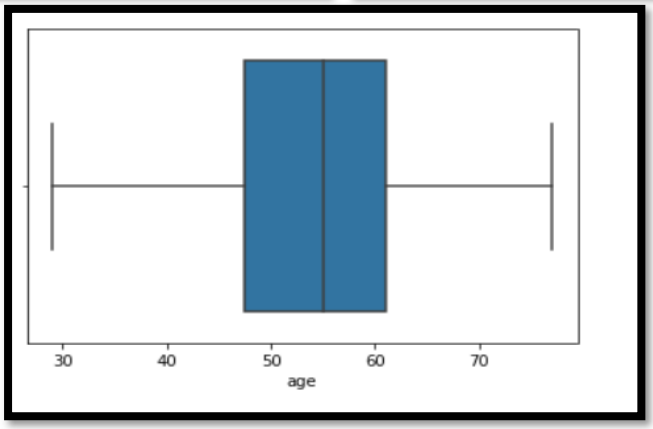
الگوریتم جنگل ایزوله، یکی از مدل‌های مبتنی بر درخت (Tree Models) است که براساس تقسیم و تفکیک مشاهدات به نقاط با فراوانی کم و متفاوت از بقیه عمل می‌کند. به این ترتیب نقطه‌ای تصادفی در بین کوچکترین و بزرگترین مقدار انتخاب شده و همسایگی حول آن سنجیده می‌شود. اگر تعداد همسایه یک نقطه نسبت به بقیه نقاط، کم باشد، آن نقطه به عنوان مقدار مشکوک و ناهنجار شناسایی می‌شود.

ما در این جا به بررسی از طریق شکل توزیع یا پراکندگی داده‌ها به وسیله محاسبه شاخص‌های مرکزی و پراکندگی برای داده‌ها می پردازیم و از روش تک متغیره استفاده می نماییم و بر اساس نمودار جعبه ای داده های پرت را حذف می کنیم

نمودارهای مربوط به ستون Age :

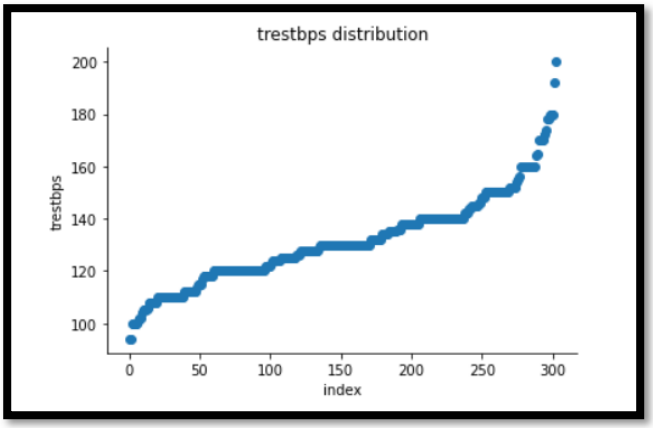


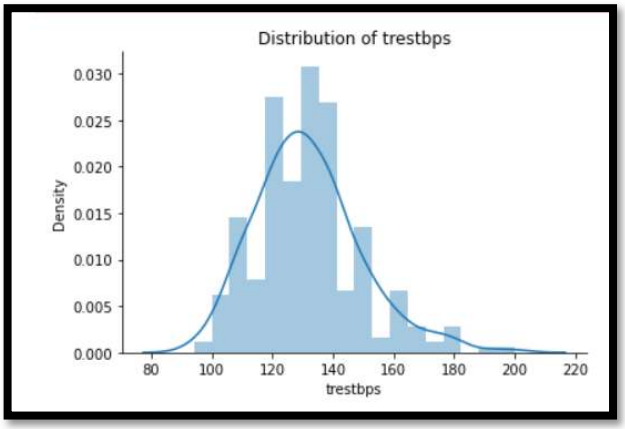


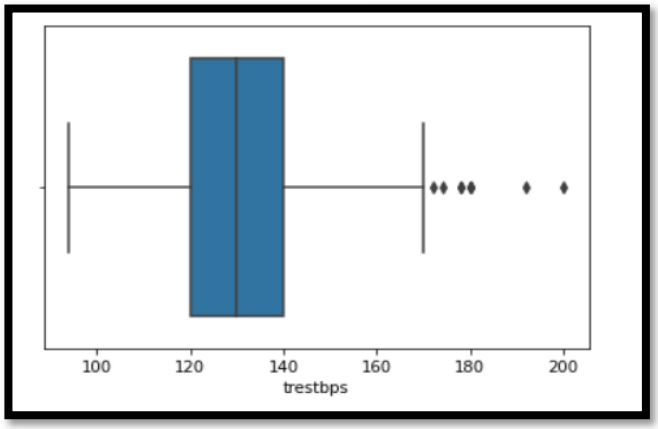


همانطور که در سه نمودار فوق مشاهده می کنیم در این قسمت ویژگی سن داده پرتی وجود ندارد

نمودارهای مربوط به ستون trestbps







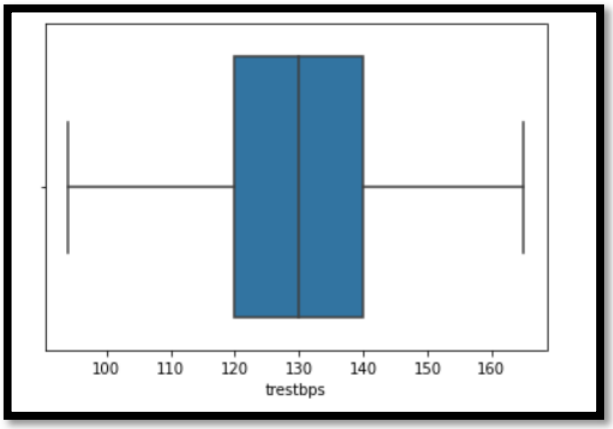
در این سه نمودار داده های پرت نمایش داده شده است و در نمودار boxplot به وضوح می بینیم که از ۱۷۰ به بعد داده پرت هست و باید این داده ها حذف گردد

همین طور که ضریب چولگی و کشیدگی را می بینیم دلیل بر وجود چولگی و کشیدگی هست

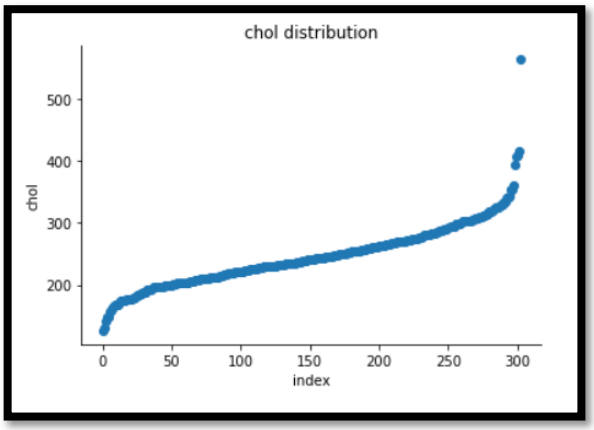
```
print("Skewness: %f" % df['trestbps'].skew())
print("Kurtosis: %f" % df['trestbps'].kurt())
```

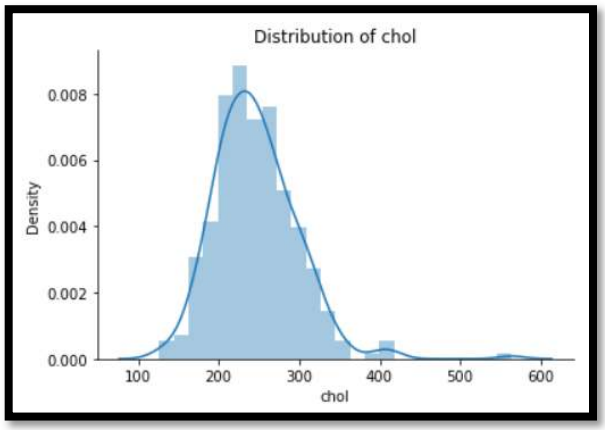
```
Skewness: 0.713768
Kurtosis: 0.929054
```

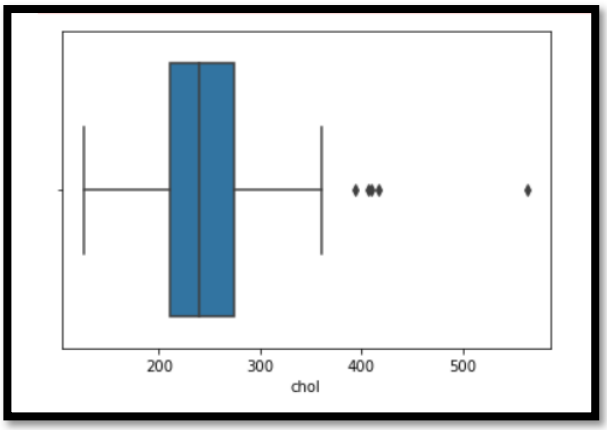
در نمودار boxplot روبرو مشاهده می شود که داده های پرت حذف شده اند



نمودارهای مربوط به ستون Chol







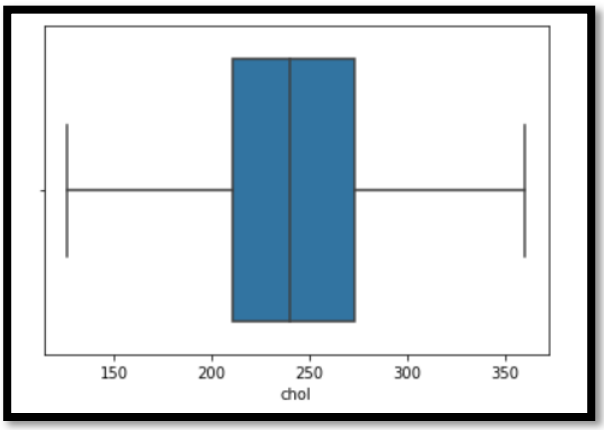
در این قسمت هم نمودارها نشان از داده پرت دارد و داده های بالای ۳۹۰ باید حذف گردد.

همانطور که در ذیل مشاهده می شود ضریب چولگی و کشیدگی نیز نشان از این امر دارد

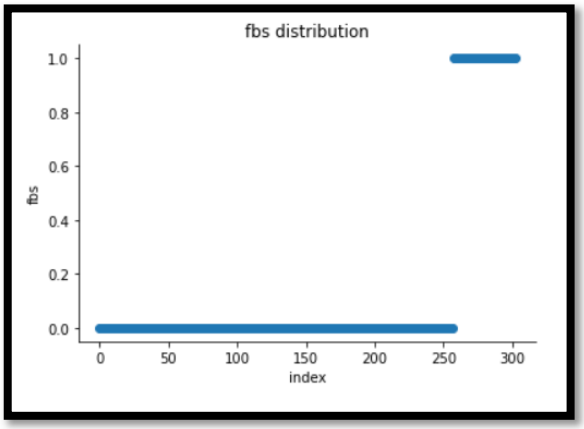
```
print("Skewness: %f" % df['chol'].skew())
print("Kurtosis: %f" % df['chol'].kurt())
```

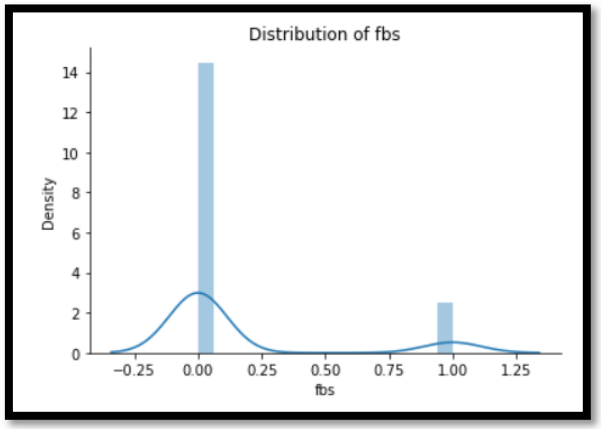
```
Skewness: 1.143401
Kurtosis: 4.505423
```

در ذیل مشاهده می کنید که داده های پرت حذف شده است

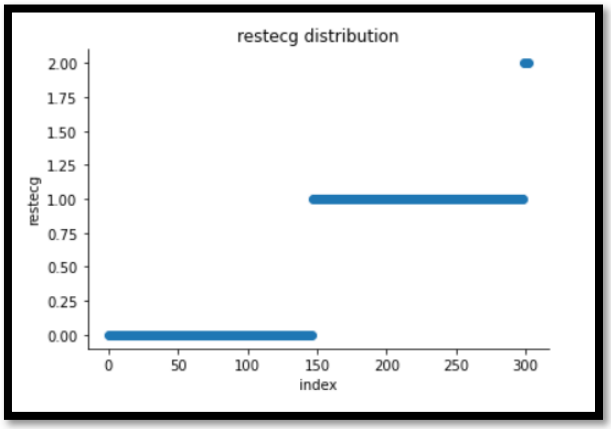


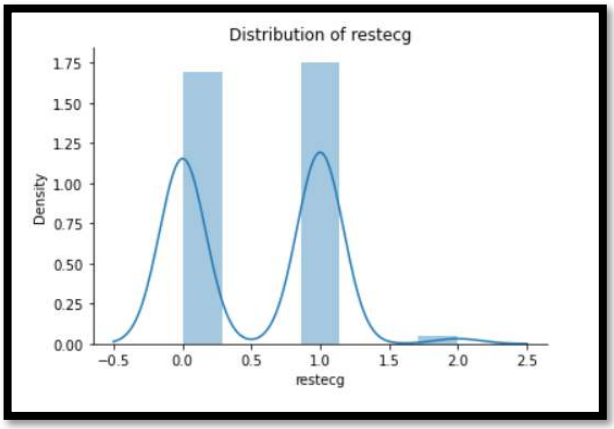
نمودارهای مربوط به ستون Fbs

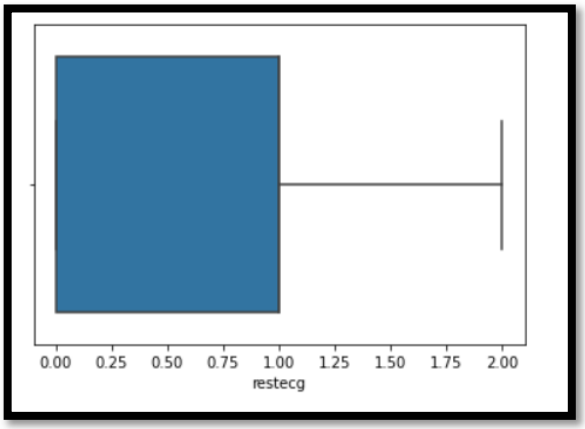




نمودارهای مربوط به ستون restecg

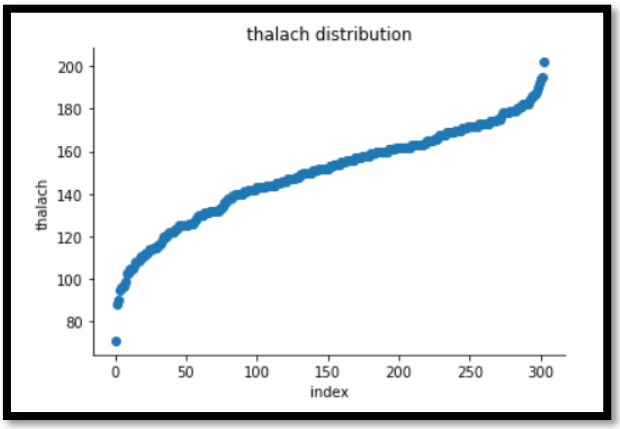


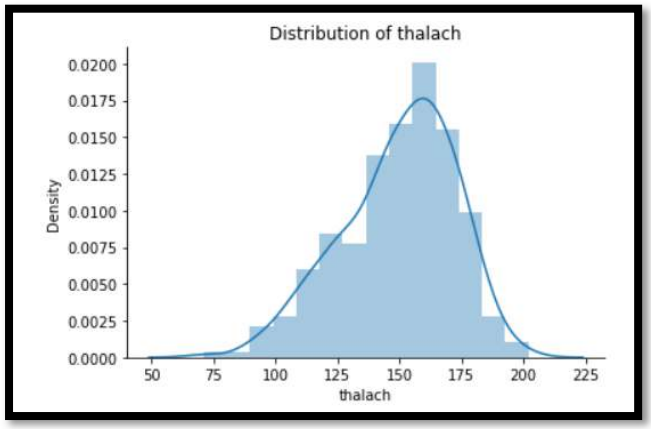


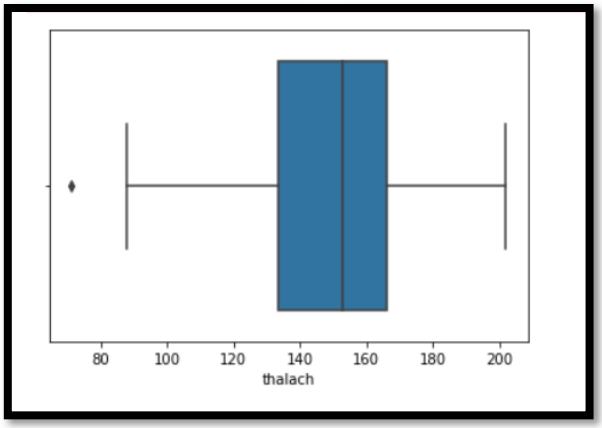


از آنجایی که ویژگی های مذکور دارای دو بازه و یا دسته بزرگ و کوچک هستند و برطبق نمودارها داده های کوچک ، داده پرت به حساب نمی آیند

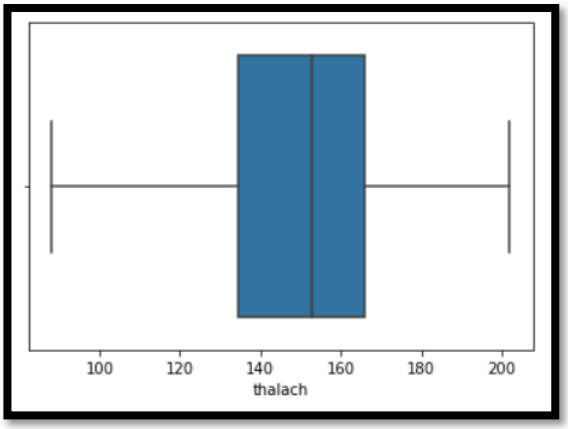
نمودارهای مربوط به ستون thalach



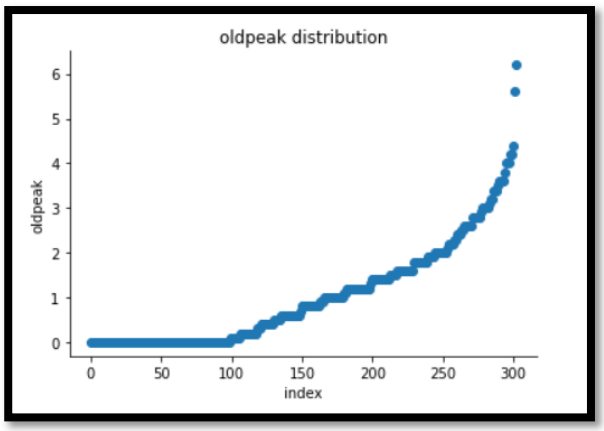


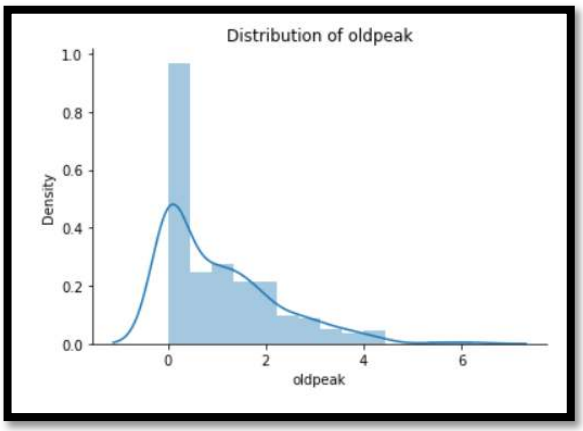


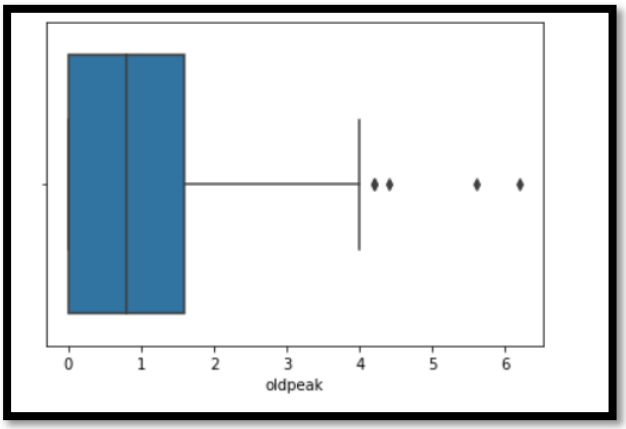
در نمودارهای فوق نیز قابل مشاهده است که داده پرت وجود دارد و حذف می کنیم



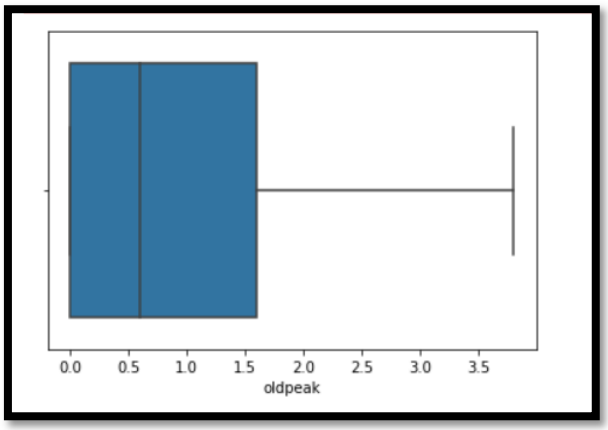
نمودارهای مربوط به ستون oldpeak





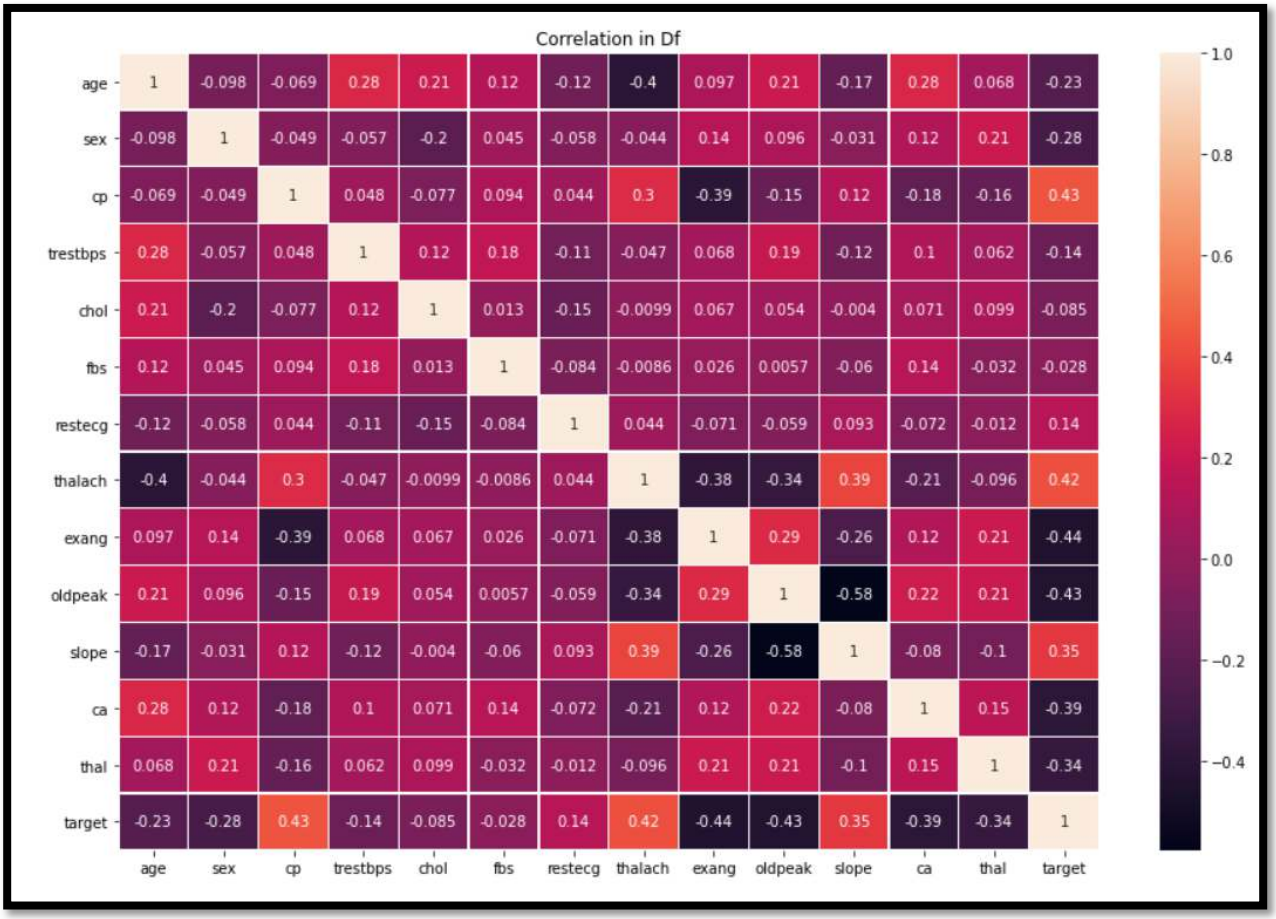


با مشاهده نمودارها، این ویژگی هم دارای داده پرت هست که حذف خواهد شد



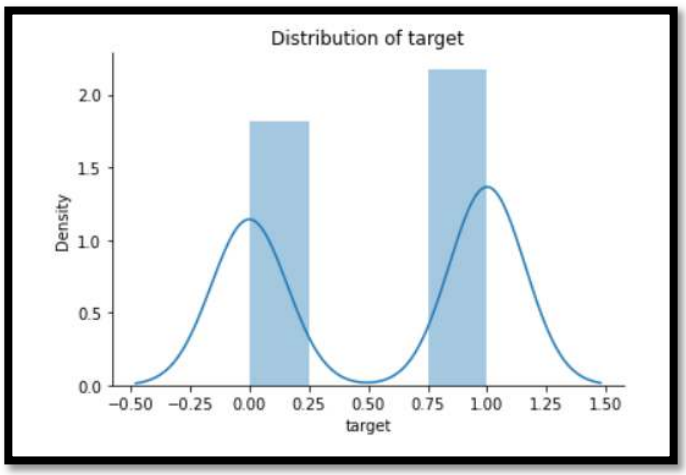
برای سایر داده ها هم بدین صورت عمل شده است که در فایل کدها قابل رویت است و به جهت عدم طولانی بودن گزارش در این جا قید نمی گردد

برای این که بتوانیم داده های پرت را بهتر به دست آوریم می توانیم از مقایسه چند متغیر که با هم همبستگی دارند درنظر بگیریم .سپس داده هایی که بر اساس همبستگی این چند متغیر پرت هستند را به عنوان داده پرت در نظر بگیریم .پس ابتدا لازم است که ضرایب همبستگی متغیر ها را بدست آوریم



۱-ب) بررسی کنید آیا تعداد نمونه ها در هر کلاس متوازن است ؟(به صورت مختصر توضیح دهید اگر داده ها متوازن نباشد چه مشکلاتی ممکن است پیش بیاید وچه راه حل هایی برای آن وجود دارد

داده های دو کلاس متوازن هستند .اگر داده های یک دسته کمتر از۵ درصد باشد یعنی داده های کلاس ها متوازن نیستند و این مسئله باعث می شود که الگوریتم های یادگیری جهتدار شوند.



در چنین وضعیتی، مدل پیشگویانه ای که با به کارگیری الگوریتم های یادگیری ماشین ایجاد شده است، جهت دار و یکطرفه شده و دقت آن بسیار پایین خواهد بود.
این اتفاق بدین خاطر میافتد که الگوریتم های یادگیری ماشین معمولا طوری طراحی شده اند که با کاهش خطا، دقت مدل را افزایش دهند .بنابراین، این الگوریتم ها توزیع/نسبت یک کلاس نسبت به کل کلاس ها، یا توازن کلاس ها را در محاسبات خود به حساب نمی آورند.
رویکردهای متنوعی برای حل مشکل داده های نامتوازن وجود دارد که تکنیکهای نمونه برداری مختلفی را به کار میگیرند.
حل مشکل کلاس های نامتوازن در در الگوریتم های پیش بینی
رویکردهای مختلفی برای مواجهه با داده های نامتوازن وجود دارند که در زیر فهرستی از آنها آورده شده است:
الف)رویکرد در سطح داده :تکنیکهایResampling

Random Under Sampling
Random Over Sampling
Cluster-Based Over Sampling
Informed Over Sampling: Synthetic Minority Over Sampling Technique
Modified synthetic minority oversampling technique (MSMOTE)

ب)تکنیک های الگوریتمی تجمعی(Algorithmic Ensemble Techniques)

Bagging Based
Boosting-Based
Adaptive Boosting- Ada Boost
Gradient Tree Boosting
XG Boost

۲. نمونه های موجود در دیتاست را با نسبت ۸۰ به ۲۰ به دو بخش داده های آموزشی و داده های تست تقسیم بندی کنید . برای این کار میتوانید از پکیج sklearn ۱ بهره ببرید.

```
from sklearn.model_selection import train_test_split

train, test = train_test_split(df1, test_size=0.20, train_size=0.80)
```

ونتیجه دستور فوق به صورت ذیل قابل مشاهده می باشد

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 222 entries, 86 to 243
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       222 non-null    int64
 1   sex       222 non-null    int64
 2   cp        222 non-null    int64
 3   trestbps  222 non-null    int64
 4   chol      222 non-null    int64
 5   fbs       222 non-null    int64
 6   restecg   222 non-null    int64
 7   thalach   222 non-null    int64
 8   exang     222 non-null    int64
 9   oldpeak   222 non-null    float64
 10  slope     222 non-null    int64
 11  ca        222 non-null    int64
 12  thal      222 non-null    int64
 13  target    222 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 26.0 KB
```

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 56 entries, 297 to 290
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       56 non-null     int64
 1   sex       56 non-null     int64
 2   cp        56 non-null     int64
 3   trestbps  56 non-null     int64
 4   chol      56 non-null     int64
 5   fbs       56 non-null     int64
 6   restecg   56 non-null     int64
 7   thalach   56 non-null     int64
 8   exang     56 non-null     int64
 9   oldpeak   56 non-null     float64
 10  slope     56 non-null     int64
 11  ca        56 non-null     int64
 12  thal      56 non-null     int64
 13  target    56 non-null     int64
dtypes: float64(1), int64(13)
memory usage: 6.6 KB
```

۳. قضیه بیز را در حداقل یک پاراگراف بیان کنید . سپس دسته بند های Gaussian Naive Bayes، Multinomial Naive Bayes ،Bernoulli Naive Bayes را با یکدیگر مقایسه کنید و بیان کنید هر کدام از این دسته بندها بیشتر در کجا کاربرد دارند.

قضیه بیز از روشی برای دسته بندی پدیده ها بر پایه احتمال استفاده میکند. و احتمال رخداد پیشامد A به شرط B برابر است با احتمال رخداد پیشامد B به شرط A ضرب در احتمال رخداد پیشامد A تقسیم بر احتمال رخداد پیشامدB

$$P(A|B) = \frac{P(A|B)P(A)}{P(B)}$$

دسته بندی به این صورت انجام می شود که : احتمال این که یک نمونه در هر کدام از دسته ها باشد را بدست ی آوریم و نمونه را در دسته ای قرار میدهیم که مقدار احتمال آن بیشتر است .درقضیه بیز فرمول بالا به صورت زیر تعریف میشود:

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}$$

که در آن c دسته مورد نظر و x مقادیر نمونه است.

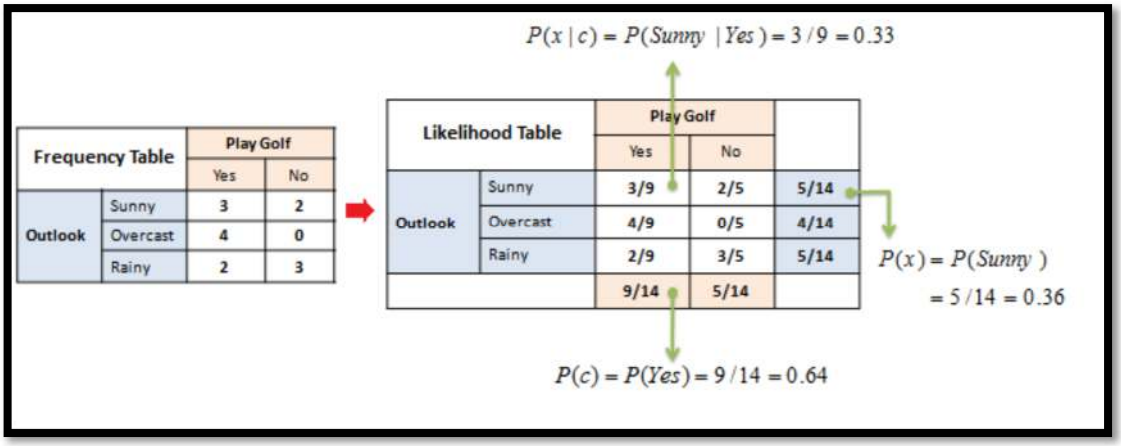
- $P(c \mid x)$احتمال پیش بینی کننده(ویژگی)است.
- $P(c)$احتمال قبلی کلاس است.
- $P(x \mid c)$این احتمال است که احتمال کلاس پیش بینی کننده داده شده است.
- $P(x)$ احتمال قبلی پیش بینی کننده است.

در مثال زیر بهتر میتوانیم هرکدام از این احتمالات را با مثال بدست آوریم:

$p(x|c)$احتمال این که نمونه

$p(c)$از تقسیم تعداد برچسب های آن دسته به نسبت کل داده ها بدست می آید

$p(x)$احتمال قبلی که احتمال وقوع مقدار مورد نظر برای نمونه را نشان می دهد



قضیه بیز بسته به داده هایی که داریم میتواند به صورت های مختلفی به کار برده شود .مثلا برای داده های پیوسته و دارای توزیع نرمال از الگوریتم دسته بند بیز گاوسی استفاده می شود .در ادامه سه دسته بند را معرفی و با هم مقایسه می نماییم.

- gaussian naive bayse

برای داده های پیوسته و با توزیع نرمال مناسب است و احتمال آن با فرمول زیر محاسبه می شود

در این حالت هر دسته یا گروه دارای توزیع گاوسی است. به این ترتیب اگر kدسته یا کلاس داشته باشیم می‌توانیم برای هر دسته میانگین و واریانس را محاسبه کرده و پارامترهای توزیع نرمال را برای آن‌ها برآورد کنیم. فرض کنید که μkمیانگین و $\sigma 2k$واریانس دسته kام یعنی Ckباشد. همچنین vرا مشاهدات حاصل از متغیرهای تصادفی Xدر نظر بگیرید. از آنجایی که توزیع Xدر هر دسته گاوسی (نرمال) فرض شده است، خواهیم داشت:

$$p(x = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}}$$

- **Multinomial naive Bayes**

برای زمانی که feature vectors ها دارای خاصیت احتمال چندجمله ای هستند مناسب است.

بیز ساده چندجمله‌ای، به عنوان یک دسته‌بند متنی بسیار به کار می‌آید. در این حالت برحسب مدل احتمالی یا توزیع **چند جمله‌ای**، برداری از nویژگی برای یک مشاهده به صورت $X=(x1,\ldots,xn)$ با احتمالات $(p1,\ldots,pn)$ در نظر گرفته می‌شود. مشخص است که در این حالت بردار Xبیانگر تعداد مشاهداتی است که ویژگی خاصی را دارا هستند.
به این ترتیب تابع درستنمایی در چنین مدلی به شکل زیر نوشته می‌شود.

$$p(\mathsf{x} \mid C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i {p_{ki}}^{x_i}$$

اگر مدل بیز ساده را براساس لگاریتم تابع درستنمایی بنویسیم، به صورت یک دسته‌بند خطی درخواهد آمد.

$$\begin{aligned}
\log p(C_k \mid \mathsf{x}) &\propto \log\left(p(C_k) \prod_{i=1}^{n} {p_{ki}}^{x_i}\right) \\
&= \log p(C_k) + \sum_{i=1}^{n} x_i \cdot \log p_{ki} \\
&= b + \mathsf{w}_k^\top \mathsf{x}
\end{aligned}$$

- **Bernoulli naive Bayes**

در این دسته بند ویژگی ها مستقل هستند.

این نوع از دسته‌بند بیز بیشترین کاربرد را در دسته‌بندی متن‌های کوتاه داشته، به همین دلیل محبوبیت بیشتری نیز دارد. در این مدل در حالت چند متغیره، فرض بر این است که وجود یا ناموجود بودن یک ویژگی در نظر گرفته شود. برای مثال با توجه به یک لغتنامه مربوط به اصطلاحات ورزشی، متن دلخواهی مورد تجزیه و تحلیل قرار می‌گیرد و بررسی می‌شود که آیا کلمات مربوط به لغتنامه ورزشی در متن وجود دارند یا خیر. به این ترتیب مدل تابع درستنمایی متن براساس کلاس های مختلف Ckبه شکل زیر نوشته می‌شود.

$$p(\mathrm{x} \mid C_k) = \prod_{i=1}^{n} p_{ki}^{x_i}(1 - p_{ki})^{(1-x_i)}$$

مشخص است که منظور از pkiاحتمال تولید مشاهده xiاز کلاس Ckاست.

نکته :با توجه به استقلال مشاهدات، تابع درستنمایی به صورت حاصلضرب نوشته شده است.

۴. با در نظر گرفتن فیچر ها chol ، trestbps ، thalach و لیبل target یک دسته بند Bayes Naive Gaussian ۲را از پایه پیاده سازی کنید برای این کار شما نیاز است که در دیتاست آموزشی خود اعضای مختلف قاعده بیز را محاسبه کنید.

4

```
from sklearn.metrics import classification_report

import numpy as np
import math

df2 = df[['chol','trestbps','thalach']]
X_train, X_test, y_train, y_test = train_test_split(df2,
                                                    df['target'], test_size=0.2, shuffle=True)

class gaussClf:
    def separate_by_classes(self, X, y):
        self.classes = np.unique(y)
        classes_index = {}
        subdatasets = {}
        cls, counts = np.unique(y, return_counts=True)
        self.class_freq = dict(zip(cls, counts))
        print(self.class_freq)
        for class_type in self.classes:
            classes_index[class_type] = np.argwhere(y==class_type)
            subdatasets[class_type] = X[classes_index[class_type], :]
            self.class_freq[class_type] = self.class_freq[class_type]/sum(list(self.class_freq.values()))
        return subdatasets

    def fit(self, X, y):
        separated_X = self.separate_by_classes(X, y)
        self.means = {}
        self.std = {}
        for class_type in self.classes:
            self.means[class_type] = np.mean(separated_X[class_type], axis=0)[0]
            self.std[class_type] = np.std(separated_X[class_type], axis=0)[0]

    def calculate_probability(self, x, mean, stdev):
        exponent = math.exp(-((x - mean) ** 2 / (2 * stdev ** 2)))
        return (1 / (math.sqrt(2 * math.pi) * stdev)) * exponent

    def predict_proba(self, X):
        self.class_prob = {cls:math.log(self.class_freq[cls], math.e) for cls in self.classes}
        for cls in self.classes:
            for i in range(len(self.means)):
                self.class_prob[cls]+=math.log(self.calculate_probability(X[i], self.means[cls][i], self.std[cls][i]), math.e)
        self.class_prob = {cls: math.e**self.class_prob[cls] for cls in self.class_prob}
        return self.class_prob
```

```
    def predict(self, X):
        pred = []
        for x in X:
            pred_class = None
            max_prob = 0
            for cls, prob in self.predict_proba(x).items():
                if prob>max_prob:
                    max_prob = prob
                    pred_class = cls
            pred.append(pred_class)
        return pred
```

۵.سپس پیاده سازی Gaussian Naive Bayes و آموزش آن بر روی داده های آموزشی (۸۰ درصد دیتاست). نتایج را برای داده های تست (۲۰ درصد باقی دیتاست) بررسی کنید به عبارت دیگر برای داده ورودی بررسی کنید در بخش تست لیبل را پیش بینی کنید
.با توجه به این لیبل های واقعی را نیز دارید معیار های زیر گزارش دهید.

- F1 score
- Recall
- Precision

5

```
clf = gaussClf()
clf.fit(X_train.values,y_train.values)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
{0: 106, 1: 136}
              precision    recall  f1-score   support

           0       1.00      0.06      0.12        32
           1       0.49      1.00      0.66        29

    accuracy                           0.51        61
   macro avg       0.75      0.53      0.39        61
weighted avg       0.76      0.51      0.38        61
```

۶. با استفاده از پکیج *sklearn* و *GaussianNB* یک مدل بسازید و بر روی داده های آموزشی ، ترین کنید سپس بر روی داده های تست همانند سوال قبل سه معیار را گزارش دهید

مانند سوال قبلی از sklearn استفاده کرده و به این شکل که یک Object از کلاس GaussianNB از sklearn.naive_bayesاستفاده کردم و روی x_train و y_train فیت شدند

6

```
from sklearn.naive_bayes import GaussianNB

clf = GaussianNB()
clf.fit(X_train.values,y_train.values)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.79      0.47      0.59        32
           1       0.60      0.86      0.70        29

    accuracy                           0.66        61
   macro avg       0.69      0.67      0.65        61
weighted avg       0.70      0.66      0.64        61
```

۷ . بررسی کنید که در سه معیار مطرح شده مدلی که با استفاده از پکیج ساخته اید و مدلی که خود پیاده سازی کرده اید به چه صورتی عمل کرده اند.

پکیج sklearn یک پکیج بر ای استفاده عموم می سازد و بنابراین سعی می کند که به بهترین حالت ممکن جوابدهی داشته باشد و دارای بهترین پیش فرض های ممکن است .

وقتی ما برنامه نویسی میکنیم صرفا هدف ما خروجی گرفتن است، هر کدام از پکیج هایsklearn نتیجه مد تها کار تحقیقاتی است و نتیجه خروجی آن بهتر است

۸ . کلاسیفایر SVM را با استفاده از پکیج sklearn بر سه فیچر مطرح شده در سوال شماره ۴ با استفاده از داده های آموزشی ترین کنید. سپس روی داده های تست سه معیار Recall و F1 score و Precision را گزارش کنید

8

```
from sklearn.svm import SVC

clf = SVC()
clf.fit(X_train.values,y_train.values)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.73      0.34      0.47        32
           1       0.54      0.86      0.67        29

    accuracy                           0.59        61
   macro avg       0.64      0.60      0.57        61
weighted avg       0.64      0.59      0.56        61
```

۹. حداقل دو حالت مختلف را برای کرنل در SVM ساخته شده با پکیج در نظر بگیرید و نتایج آن را گزارش دهید آیا کرنل های مختلف نتایج مختلفی ارایه دادند ؟ به صورت کلی علت استفاده از کرنل ها در SVM چیست ؟ توضیح دهید

انواع کرنل در SVM:

۱. Radial basis function
۲. Polynomial
۳. Sigmoied
۴. linear

در مورد روش شماره ۱، پیشنهاد شده است در صورت زیاد بودن تعداد ویژگی‌ها (۱۰۰۰<) استفاده گردد

در این جا من از روش شماره ۲ و ۴ استفاده کردم.

یکی دیگر از راه هایی که برای دسته بندی چنین داده هایی استفاده می شود (kernel function)برای گسترش فضاست.
حالا اینکه کرنل در svm کارش این هست که فضای داده را عوض می کند و می برد در دستگاه دیگه و یا همان فضای دیگه در واقع داده ها را نگاشت می دهد در فضای دیگه و کار طبقه بندی در فضای جدید انجام می دهد .به عنوان مثال یک فضای دو بعدی در نظر بگیریم با فرض اینکه یک دایره داشته باشیم و یک دسته داخل دایره و دسته دیگر روی محیط دایره و بخواهیم طبقه بندی روی این دو تا انجام بدهیم در
واقع اون مرز تصمیم گیری همون دایره است و یک چیز پیچیده است و کرنل بزنیم بریم در فضای سه بعدی و مرز تصمیم گیری تبدیل می شود به یک صفحه و مرز تصمیم گیری ساده تر می شود .و فضای را نگاشت می دهیم و با مرز تصمیم گیری معادله اش را معکوس می دهیم به فضای اولیه یا طبق توضیحات استاد او ن خط تصمیم تبدیل می کنیم به یک رویه و داده ها راحت تر می شود جدا شودو به جای ضرب داخلی x, xi کرنل را قرار می دهیم.

یکی از مزیت های استفاده از تابع های کرنل برای گسترش فضا در مقایسه با استفاده از توان های بالاتر ویژگی ها (features) ،مزیت محاسباتی است که در حالت استفاده از کرنل تنها نیاز به محاسبه$\binom{n}{2}$ تابع کرنل داریم .اما وقتی برای گسترش فضا از توان های چندم ویژگی ها استفاده می کنیم، ممکن است محاسبات بسیار زیاد شود.

مزایا

- حاشیه جداسازی برای دسته‌های مختلف کاملاً واضح است.
- در فضاهای با ابعاد بالاتر کارایی بیشتری دارد.
- در شرایطی که تعداد ابعاد بیش از تعداد نمونه‌ها باشد نیز کار می‌کند.
- یک زیر مجموعه از نقاط تمرینی را در تابع تصمیم‌گیری استفاده می‌کند (که به آن‌ها بردارهای پشتیبان گفته می‌شود)، بنابراین در مصرف حافظه نیز به صورت بهینه عمل می‌کند.

معایب

- هنگامی که مجموعه داده‌ها بسیار بزرگ باشد، عملکرد خوبی ندارد، زیرا نیازمند زمان آموزش بسیار زیاد است.
- هنگامی که مجموعه داده نوفه (نویز) زیادی داشته باشد، عملکرد خوبی ندارد و کلاس‌های هدف دچار همپوشانی می‌شوند.
- ماشین بردار پشتیبان به طور مستقیم تخمین‌های احتمالاتی را فراهم نمی‌کند و این موارد با استفاده از یک اعتبارسنجی متقابل (Cross Validation) پرهزینه پنج‌گانه انجام می‌شوند. این امر با روش SVCموجود در کتابخانه scikit-kearn پایتون، مرتبط است.

```
clf = SVC(kernel = 'rbf')
clf.fit(X_train.values,y_train.values)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.77      0.59      0.67        29
           1       0.69      0.84      0.76        32

    accuracy                           0.72        61
   macro avg       0.73      0.71      0.71        61
weighted avg       0.73      0.72      0.72        61
```

```
from sklearn.svm import SVC

clf = SVC(kernel = 'linear')
clf.fit(X_train.values,y_train.values)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.78      0.44      0.56        32
           1       0.58      0.86      0.69        29

    accuracy                           0.64        61
   macro avg       0.68      0.65      0.63        61
weighted avg       0.68      0.64      0.62        61
```

```
clf = SVC(kernel = 'poly')
clf.fit(X_train.values,y_train.values)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.79      0.47      0.59        32
           1       0.60      0.86      0.70        29

    accuracy                           0.66        61
   macro avg       0.69      0.67      0.65        61
weighted avg       0.70      0.66      0.64        61
```

۱۰ . دسته بند SVM را با استفاده از پکیج Sklearn بسازید و با در نظر گرفتن کلیه فیچرهای دیتاست بر روی داده های آموزشی ترین کنید سپس نتایج را بر روی داده های تست،ارزیابی کنید.

10

```
from sklearn.svm import SVC

X_train, X_test, y_train, y_test = train_test_split(df1.drop(['target'],axis=1),
                                                    df1['target'], test_size=0.2, shuffle=True)

clf = SVC(kernel = 'linear')
clf.fit(X_train.values,y_train.values)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.86      0.78      0.82        23
           1       0.86      0.91      0.88        33

    accuracy                           0.86        56
   macro avg       0.86      0.85      0.85        56
weighted avg       0.86      0.86      0.86        56
```

طبیعتا معیار ها خیلی بهتر است چون روی کل داده اعمال می شود

۱۱. برای سوال شماره ۱۰، یکبار مدل را با 5-fold Cross Validation اجرا کنید و نتایج را گزارش دهید(در این جا برای فولد کردن داده ها از کل دیتاست استفاده میکنیم)

اگر مجموعه داده‌های آموزشی را به طور تصادفی به k زیرنمونه یا لایه (Fold) با حجم یکسان تفکیک کنیم، می‌توان در هر مرحله از فرایندCV ، تعداد k-1 از این لایه‌ها را به عنوان مجموعه داده آموزشی و یکی را به عنوان مجموعه داده اعتبارسنجی در نظر گرفت. با انتخابk=10 ، تعداد تکرارهای فرآیند CV برابر با ۱۰ خواهد بود و دستیابی به مدل مناسب به سرعت امکان‌پذیر می‌شود.

دیتا ست را به ۵ قسمت می کنیم و در ۵ مرحله مختلف آموزش می دهیم و به این شکل که هر بار 1/5 این داده ها را به عنوان ۲۰ درصد داده validation در نظر می گیریم و برای مثال اگر 100 تا داده داشته باشیم بار اول ۲۰ تا اول برای validation استفاده می شود و۸۰ تا بعدی برای ترین و ۲۰ تا بعدی و۶۰ تا و ۲۰ تا بعدی ترین و....

11

```
]: from sklearn.model_selection import cross_val_score

clf = SVC(kernel='linear')
scores = cross_val_score(clf, X_train, y_train, cv=5)

print(f"mean score: {scores.mean()}")
print(f"score std: {scores.std()}")
```

```
mean score: 0.8515151515151516
score std: 0.05788592689591065
```

۱۲ . با استفاده از پکیج **sklearn** دسته بند **K-NN** را بسازید با به کارگیری تمامی فیچرهای موجود در دیتاست آموزشی، مدل را ترین کنید . سپس بر روی دیتاست تست ارزیابی کنید .

12

```
: from sklearn.neighbors import KNeighborsClassifier

clf = KNeighborsClassifier(n_neighbors=3)
clf.fit(X_train, y_train)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.67      0.43      0.53        23
           1       0.68      0.85      0.76        33

    accuracy                           0.68        56
   macro avg       0.67      0.64      0.64        56
weighted avg       0.68      0.68      0.66        56
```

۱۳ . بررسی کنید در سوال شماره ۱۲ تعداد همسایه ها k چه نقشی ایفا می کند ؟ زیاد شدن همسایه ها خوب است ؟ چگونه می توان مشخص کرد چه تعداد همسایه برای مسئله ما مناسب است .

با تعداد همسایه های مختلف بررسی انجام شد
به نظرمیاید که ۳ همسایه مناسب بود
تعداد همسایه بالا برود عضویت در کلاس سخت و برعکس .
الگوریتم supervise و تعداد دسته ها از قبل مشخص است.

۱۴ . در سوال شماره ۱۲ به جای استفاده از تمامی فیچرها از سه فیچر chol, trestbps,thalach استفاده کنید و مدل را بسازید سپس نتایج ارزیابی را گزارش کنید .

14

```
df2 = df[['chol','trestbps','thalach']]
X_train, X_test, y_train, y_test = train_test_split(df2,
                                                    df['target'], test_size

clf = KNeighborsClassifier(n_neighbors=3)
clf.fit(X_train, y_train)
print(classification_report(y_test.values,clf.predict(X_test.values)))
```

```
              precision    recall  f1-score   support

           0       0.63      0.59      0.61        29
           1       0.65      0.69      0.67        32

    accuracy                           0.64        61
   macro avg       0.64      0.64      0.64        61
weighted avg       0.64      0.64      0.64        61
```

۱۵ . تفاوت بین روش های کلاس بندی پارامتری و غیرپارامتری را به صورت خلاصه بیان کنید. هرکدام بهتر است در چه مواقعی استفاده شود ؟

در حوزه «تجزیه و تحلیل آماری داده‌ها(Statistical Data Analysis) «، توزیع جامعه آماری که نمونه از آن گرفته شده، مهم است زیرا هر چه اطلاعات بیشتر در زمینه رفتار داده‌ها و شکل پراکندگی و توزیع آن‌ها وجود داشته باشد، نتایج قابل اعتمادتر و دقیق‌تر خواهند بود. در مقابل، وجود اطلاعات کم از توزیع جامعه آماری مربوط به نمونه، باعث کاهش اعتماد به نتایج حاصل از روش‌های معمول (پارامتری) آماری می‌شود. بنابراین در این حالت

مجبور به استفاده از روش‌های ناپارامتری هستیم که برای اجرای آن‌ها فرضیاتی در مورد توزیع داده‌ها وجود ندارد. به همین علت به روش‌های ناپارامتری گاهی «روش‌های توزیع-آزاد (Distribution-free Methods) «نیز می‌گویند.

آمار پارامتری و روش‌های تجزیه و تحلیل مرتبط

داده‌های پارامتری به نمونه‌ای گفته می‌شود که از توزیع جامعه آماری آن مطلع هستیم. معمولا این توزیع آماری برای داده‌های کمی، نرمال یک یا چند متغیره در نظر گرفته می‌شود. در این حالت از آزمون‌های آماری پارامتری مثل آزمون T، آزمون F و یا آزمون Z استفاده می‌کنیم. همچنین برای اندازه‌گیری میزان همبستگی بین متغیرهای دو یا چند بعدی نیز از ضریب همبستگی پیرسون استفاده خواهیم کرد.

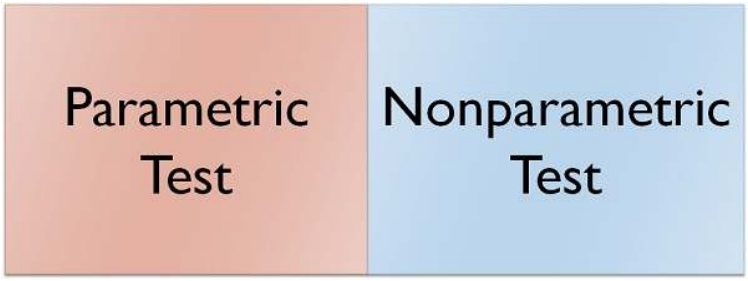
آمار ناپارامتری و روش‌های تجزیه و تحلیل مرتبط

اگر توزیع جامعه آماری نامشخص باشد و از طرفی حجم نمونه نیز کوچک باشد بطوری که نتوان از قضیه حد مرکزی برای تعیین توزیع حدی یا مجانبی جامعه آماری، استفاده کرد، از تحلیل‌های ناپارامتری استفاده می‌شود، زیرا در این حالت کارآمدتر از روش‌های پارامتری هستند. به این ترتیب در زمانی که توزیع جامعه مشخص نباشد و یا حجم نمونه کم باشد، روش‌ها و آزمون‌های ناپارامتری نسبت به روش‌ها و آزمون‌های پارامتری از توان آزمون بیشتری برخوردارند و نسبت به آن‌ها ارجح هستند.

بهتر است شرایط بهره‌گیری از روش‌های ناپارامتری را به صورت زیر لیست کنیم:

- برای داده‌ها، نتوان توزیع آماری مناسبی در نظر گرفت.
- وجود داده‌های پرت(Outlier) ، وجود چند نما و ... امکان انتخاب توزیع نرمال را برایشان میسر نمی‌کند.
- کم بودن حجم نمونه برآورد پارامترهای توزیع نرمال مانند میانگین و بخصوص واریانس را دچار مشکل می‌کند و در عمل امکان بررسی توزیع نرمال به علت حجم کم نمونه برای جامعه وجود ندارد.

روش‌های ناپارامتری در چنین موقعیت‌های می‌تواند راهگشا باشد و به تحلیل‌گر داده (Data Scientist) برای شناخت داده‌ها یاری برساند.



باید توجه داشت که اگر توزیع جامعه آماری قابل تحقیق و تعیین باشد، اجرای روش‌های پارامتری بر روش‌های ناپارامتری ارجح هستند زیرا در این حالت روش‌های پارامتری نسبت به روش‌های ناپارامتری از دقت بیشتری برخوردارند. بنابراین فقط زمانی که از توزیع جامعه آماری مطلع نیستم، به اجبار از روش‌های ناپارامتری استفاده خواهیم کرد. البته اگر حجم نمونه بزرگ باشد، در اکثر موارد، نتایج حاصل از آزمون‌های پارامتری و ناپارامتری با یکدیگر همخوانی دارند.

۱۶ . معیار Matthews Correlation Coefficient(MCC) چیست و در چه جاهایی استفاده می شود .

شاید بتوان تحلیل سیگنال‌های رادار در جنگ جهانی دوم را اولین زمان ظهور منحنی ROC و کاربردهای آن دانست. البته بعدها از چنین منحنی در «نظریه شناسایی سیگنال (Signal Detection Theory) «نیز استفاده شد. پس از جنگ در «پرل هاربر (Pearl Harber) «در سال ۱۹۴۱، که نیروهایی آمریکایی به شدت آسیب دیدند، ارتش آمریکا تصمیم گرفت سیگنال‌های راداری (Radar Signal) به منظور کشف و شناسایی هواپیماهای ژاپنی را بهبود دهد. برای این کار، آن‌ها توانایی یک گیرنده رادار را در تشخیص هواپیما اندازه‌گیری کردند و از آن پس واژه «مشخصه عملکرد گیرنده (Receiver Operating Characteristic) «برای ارزیابی عملکرد دستگاه‌های تشخیص سیگنال، مورد استفاده قرار گرفت.

در دهه ۱۹۵۰، منحنی‌های ROC در روانشناسی نیز به کار گرفته شدند تا ضعف در قوه تشخیص انسان‌ها (و گاهی حیوان) را مورد بررسی و ارزیابی قرار دهند. در پزشکی، تجزیه و تحلیل ROC به طور گسترده‌ای در سنجش صحت آزمایش‌های تشخیص پزشکی و تعیین میزان دقت چنین آزمایشاتی، مورد استفاده قرار گرفته است.

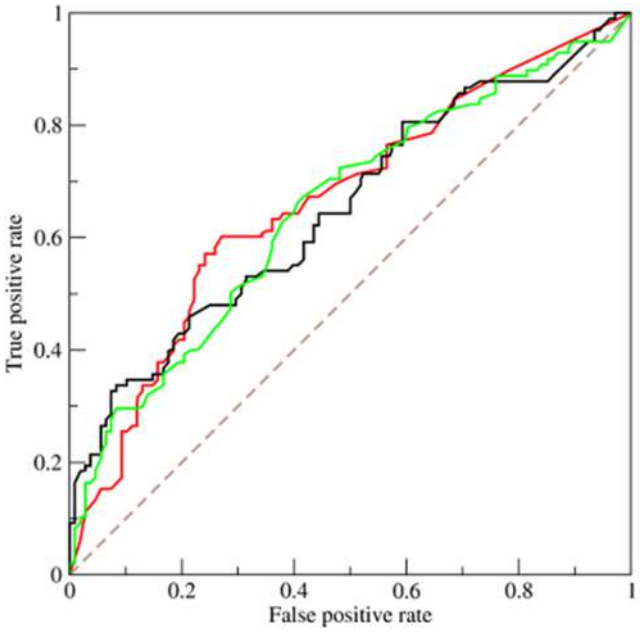
منحنی‌های ROC همچنین در اپیدمیولوژی و تحقیقات پزشکی بسیار مورد استفاده قرار می‌گیرند. در رادیولوژی، تجزیه و تحلیل ROC یک روش معمول برای ارزیابی تکنیک‌های جدید رادیولوژی است. همچنین در علوم اجتماعی، آنالیز منحنی ROC اغلب به عنوان «نسبت دقت مشخصه عملکرد(ROC Accuracy Ratio) «، یاد شده و یک تکنیک معمول برای قضاوت در مورد مدل‌های احتمال پیش‌بین **(Predictive Probability Model)** است.

منحنی ROC و کاربردهای آن در یادگیری ماشین بخصوص در شاخه نظارت شده آن، مفید بوده است. به این ترتیب مقادیر مربوط به منحنی ROC می‌تواند مبنایی برای مقایسه و ارزیابی الگوریتم‌های دسته‌بندی (Classifiers Algorithms)باشد. منحنی‌های ROC همچنین در تأیید پیش بینی‌ها در هواشناسی نیز مورد بهره‌برداری قرار می‌گیرد.

منحنی مشخصه عملکرد چیست؟

یک منحنی مشخصه عملکرد که به اختصار آن را منحنی ROC می‌نامیم، یک نمودار برای نمایش توانایی ارزیابی یک سیستم دسته‌بندی باینری محسوب می‌شود که آستانه تشخیص آن نیز متغیر است.

منحنی ROC توسط ترسیم نسبت یا نرخ مثبت صحیح (True Positive Rate) که به اختصار TPR نامیده می‌شود برحسب نرخ مثبت کاذب (False Positive Rate) با نام اختصاریFPR ، ایجاد می‌شود. البته توجه داشته باشید که آستانه برای این مقادیر، متغیر است. به همین دلیل، یک نمودار پیوسته ایجاد خواهد شد.



نرخ مثبت صحیح را در یادگیری ماشین(Machine Learning) گاهی حساسیت (Sensitivity) یا «بازیابی (Recall) «یا «احتمال شناسایی (Probability Detection) «می‌نامند. همچنین «نرخ مثبت کاذب» هم به صورت «احتمال دریافت اخطار کاذب (Probability False Alarm) «شناخته شده و براساس متمم «ویژگی (Specificity) «سنجیده می‌شود. البته در ادامه هر یک از این نسبت‌ها، در جدول ۱، توصیف خواهند شد.

نکته :با توجه به آزمون فرض آماری و مفاهیم مربوط به آن، منحنی ROC را می‌توان به معنی «توان آزمون « (Test Power)برحسب مقادیر مختلف «خطای نوع اول (Error I Type) «در نظر گرفت که براساس یک نمونه تصادفی تولید شده است.

اگر با زبان شناسایی سیگنال و مخابرات به ROC توجه کنیم، به طور کلی با شرط مشخص بودن توزیع احتمالی برای هر دو بخش (TPR) و (FPR) منحنی ROC در صورتی حاصل خواهد شد، که» تابع توزیع تجمعی (Cumulative Distribution function) «یا سطح زیر منحنی توزیع احتمال تشخیص درست سیگنال را در محور عمودی و تابع توزیع تجمعی تشخیص نادرست سیگنال را در محور افقی در نظر بگیریم.

منحنی ROC را به عنوان «نمودار مشخصه نسبی عملکرد (Relative Operating Characteristic) «نیز می‌شناسند زیرا مقایسه‌ای بین دو نحوه عملکرد (TPR , FPR) ارائه می‌کند. به این ترتیب آنچه امروزه) بدون در نظر گرفتن مفهوم شناسایی سیگنال توسط دستگاه دریافت کننده یا (Receiver به عنوان منحنی ROC می‌شناسیم همان نمودار مشخصه نسبی عملکرد است.

مفاهیم اولیه در منحنیROC

یک مدل دسته‌بندی، یک نگاشت از مشاهدات به دسته یا گروه‌های مشخص است. از آنجایی که دسته‌بندها (Classifiers)مقادیر حقیقی و پیوسته را تولید می‌کنند، لازم است برحسب یک مقدار آستانه، دسته یا گروه‌ها را مشخص و از یکدیگر مجزا کنیم. برای مثال با توجه به میزان فشار خون فرد (که مقداری پیوسته است) باید افراد را در یکی از دسته‌های بیمار فشار خون یا بدون بیماری، قرار دهیم. به این ترتیب نگاشت از **مجموعه اعداد حقیقی** به زیر مجموعه‌ای متناهی از **اعداد طبیعی**، صورت می‌گیرد.

یک مسئله «دسته‌بندی دو دویی (Binary Classification) «را در نظر بگیرید. نتایج این دسته‌بندی با دو برچسب مثبت (P) و منفی (N) مشخص می‌شوند. درست مانند یک آزمایش پزشکی که دارای نتیجه مثبت (بیمار بودن طبق آزمایش) و منفی (نداشتن بیماری طبق آزمایش) است. از این آزمایش پزشکی می‌توان چهار حالت مختلف را در نظر گرفت.

- فرض کنید نتیجه براساس پیش‌بینی (آزمایش پزشکی) مثبت (P) است. به این ترتیب:
  - اگر نتیجه واقعی نیز مثبت (P) است، این نتیجه را «مثبت صحیح (TP) «می‌نامیم.
  - اگر نتیجه واقعی منفی (N) باشد، این نتیجه را به نام «مثبت کاذب (FP) «می‌شناسیم.
- فرض کنید نتیجه براساس پیش‌بینی (آزمایش پزشکی) منفی (N) است. در این حالت:

- اگر نتیجه واقعی نیز منفی بوده، چنین وضعیتی را به نام «منفی صحیح (TN) «به کار خواهیم برد.
- ولی اگر نتیجه واقعی مثبت باشد، چنین حالتی به نام «منفی کاذب (FN) «شناخته می‌شود.

حال P را تعداد افرادی در نظر بگیرید که در واقعیت در گروه مثبت قرار دارند. همچنین N نیز بیانگر تعداد افرادی است که در گروه N عضویت دارند. همچنین F و T را هم به صورت اعداد صحیح در نظر داشته باشید که تعداد مشاهدات در هر گروه براساس پیش‌بینی را مشخص می‌کنند. این چهار وضعیت را مطابق با یک جدول توافقی با دو سطر و دو ستون یا یک ماتریس دو در دو، نمایش می‌دهیم. توجه داشته باشید که در اینجا TP, TN, NP, FNتعداد مشاهداتی هستند که در هر یک از این چهار گروه قرار می‌گیرند.

جدول ۱: جدول توافقی برای مقایسه تعداد اعضای در گروه‌های مختلف براساس دسته‌بندی دو دویی

واقعیت پیش‌بینی	P	N
T	TP	TN
F	FP	FN

در این صورت گزاره‌های زیر را برای این جدول می‌توان به کار برد:

- مثبت صحیح (TP) بیانگر ضربه (hit) یا شناسایی صحیح سیگنال است.
- منفی صحیح (TN) به معنی رد صحیح (Correct Reject) سیگنال رسیده است. یعنی تشخیص صحیح یک نویز که ممکن بود به اشتباه سیگنال تلقی شود.
- مثبت کاذب (FP) همان اخطار کاذب (False Alarm) است، که می‌توان در آزمون فرض آماری آن را معادل خطای نوع اول (Type I Error) در نظر گرفت. در این حالت یک نویز به عنوان سیگنال شناخته شده است.
- منفی کاذب (FN) معادل سیگنال گمشده (miss) یا «خطای نوع دوم (Type II Error) «در آزمون فرض آماری است. به این ترتیب دستگاه دریافت کننده، نتوانسته سیگنال موجود را شناسایی کند.

حال محاسبات مربوط به نسبت یا نرخ‌های مختلف برحسب این مقادیر را مطابق با جدول زیر معرفی می‌کنیم.

جدول ۲: اندیس‌های شناسایی نرخ یا نسبت دسته‌بندی صحیح

نام شاخص	شرح	نحوه محاسبه
Sensitivity, recall, hit rate, or true positive rate (TPR)	حساسیت، بازیابی، نرخ اصابت یا نرخ مثبت صحیح	$\mathrm{TPR} = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR$
Specificity, selectivity or true negative rate (TNR)	ویژگی، گزینشی یا نرخ منفی صحیح	$\mathrm{TNR} = \frac{TN}{N} = \frac{TN}{TN + FP} = 1 - FPR$
Precision or positive predictive value (PPV)	دقت، مقدار پیش‌بینی مثبت	$\mathrm{PPV} = \frac{TP}{TP + FP} = 1 - FDR$
Negative predictive value (NPV)	مقدار پیش بینی منفی	$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}} = 1 - \mathrm{FOR}$
Miss rate or false negative rate (FNR)	نرخ گمشدن سیگنال یا نرخ منفی کاذب	$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} = 1 - \mathrm{TPR}$
Fall-out or false positive rate (FPR)	خطا یا نرخ مثبت کاذب	$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} = 1 - \mathrm{TNR}$
False discovery rate (FDR)	نرخ کشف کاذب	$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TP}} = 1 - \mathrm{PPV}$
False omission rate (FOR)	نرخ حذف خطا	$\mathrm{FOR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TN}} = 1 - \mathrm{NPV}$
Prevalence Threshold (PT)	آستانه شیوع	$\mathrm{PT} = \frac{\sqrt{TPR(-TNR + 1)} + TNR - 1}{(TPR + TNR - 1)}$
Threat score (TS) or critical success index (CSI)	نمره تهدید یا شاخص موفقیت بحرانی	$\mathrm{TS} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}}$

همچنین شاخص‌های ارزیابی دسته‌بندی نیز می‌تواند به یکی از روش‌های معرفی شده در جدول ۳، مورد محاسبه قرار گیرد. واضح است که پارامترهای مورد استفاده، باید از جدول ۲ استخراج شوند.

جدول ۳: شاخص‌های ارزیابی دسته‌بندی

نام شاخص	شرح	نحوه محاسبه
Accuracy (ACC)	صحت یا دقت	$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$
Balanced accuracy (BA)	صحت یا دقت متعادل	$\mathrm{BA} = \frac{TPR + TNR}{2}$
F1 score	امتیاز اف وان- میانگین توافقی دقت و حساسیت	$\mathrm{F_1} = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$
Matthews correlation coefficient (MCC)	ضریب همبستگی ماتیوس	$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}}$
Fowlkes–Mallows index (FM)	شاخص فولکس-مالوز	$\mathrm{FM} = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}} = \sqrt{PPV \cdot TPR}$
Informedness or bookmaker informedness (BM)	آگاهی بخشی یا نشانگر آگاهی بخشی	$\mathrm{BM} = \mathrm{TPR} + \mathrm{TNR} - 1$
Markedness (MK) or deltaP	علامت‌داری یا دلتای پی	$\mathrm{MK} = \mathrm{PPV} + \mathrm{NPV} - 1$

فضای ROC

جدول ۱ که یک جدول توافقی است، می‌تواند شاخص‌های متعددی برای ارزیابی دسته‌بندی ارائه کند که بیشتر آن‌ها را در جدول ۲ و ۳ مشاهده کردید. ولی برای ترسیم نمودار یا منحنی ROC فقط به «نرخ مثبت صحیح (TPR)» و «نرخ مثبت کاذب (FPR)» احتیاج داریم.

به این ترتیب TPR مشخص می‌کند که به چه نسبتی پیش‌بینی صحیح صورت گرفته است. یعنی تعداد پیش‌بینی‌های صحیح بر تعداد نتایج مثبت واقعی تقسیم شده و نرخ پیش‌بینی صحیح مثبت محاسبه می‌شود. از طرف دیگر FPR نشانگر تعداد شناسایی‌های مثبت از میان مشاهدات منفی است. این نسبت نیز به عنوان نرخ مثبت کاذب در نمودار ROC به کار می‌رود.

بنابراین فضای ROC بوسیله این دو شاخص یعنی FPR روی محور افقی و TPR روی محور عمودی شکل داده می‌شود. در نتیجه یک توازن بین سود (TP) و هزینه (FP) روی نمودارROC ، شکل می‌گیرد. توجه داشته باشید که هر عنصر از «ماتریس درهم‌ریختگی (Confusion Matrix)» یک نقطه در منحنی ROC را تشکیل می‌دهد.

منابع مورد استفاده:

https://towardsdatascience.com/how-to-impliment-a-gaussian-naive-bayes-classifier-in-python-from-scratch-11e0b80faf5a

https://www.antoniomallia.it/lets-implement-a-gaussian-naive-bayes-classifier-in-python.html

https://blog.faradars.org/parametric-and-non-parametric-statistics/

https://towardsdatascience.com/the-best-classification-metric-youve-never-heard-of-the-matthews-correlation-coefficient-3bf50a2f3e9a