

# ML Bonus Project



## PREPARED BY

**Samin Mehdizadeh - 810100526**

**Mohsen Fayyaz - 810100524**

Winter 2022

## Table of Contents

3	پردازش داده‌ها
9	طبقه‌بندی
9	KNN
12	Artificial Neural Network
13	SVM
15	RandomForestClassifier
18	منابع

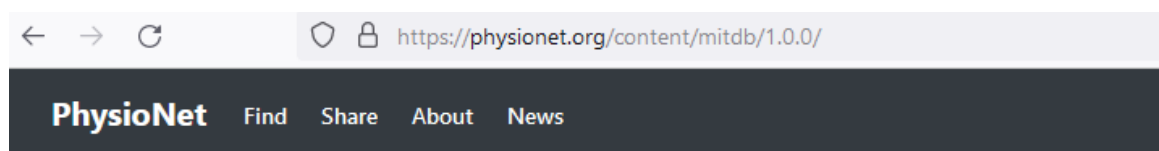
## پردازش داده‌ها

در این پروژه به پیاده‌سازی مقاله زیر می‌پردازیم.

Congestive heart failure detection using random forest classifier

هدف این مقاله طبقه‌بندی نوار قلب ECG به دسته‌های نرمال و نارسایی احتقانی قلب یا congestive heart failure CHF است.

داده‌های این مقاله به صورت عمومی و رایگان در دسترس بود.

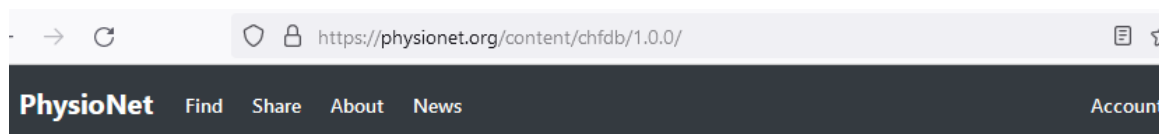


Database Open Access

### MIT-BIH Arrhythmia Database

George Moody  , Roger Mark 

Published: Feb. 24, 2005. Version: 1.0.0



Database Open Access

### BIDMC Congestive Heart Failure Database

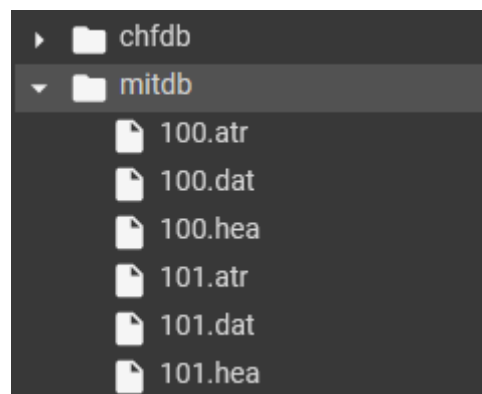
Published: Oct. 14, 2000. Version: 1.0.0

بنابراین ابتدا داده ها را دانلود کردیم.


## Download database & Test for an example

```
[3] 1 import os
     2 wfdb.dl_database('mitdb', os.path.join(os.getcwd(), 'mitdb'))
     3 wfdb.dl_database('chfdb', os.path.join(os.getcwd(), 'chfdb'))
```

پس از دانلود فایل‌ها به شکل زیر بودند.



که فایل‌های با پسوند `dat` فایل اصلی موج ذخیره شده هستند. برای تحلیل این امواج ابتدا لازم بود خوانده شوند. با استفاده از کتابخانه `wfdb` که مخصوص این کار است ابتدا فایلها را می‌خوانیم.

 <https://wfdb.readthedocs.io/en/latest/>

## wfdb

### Introduction

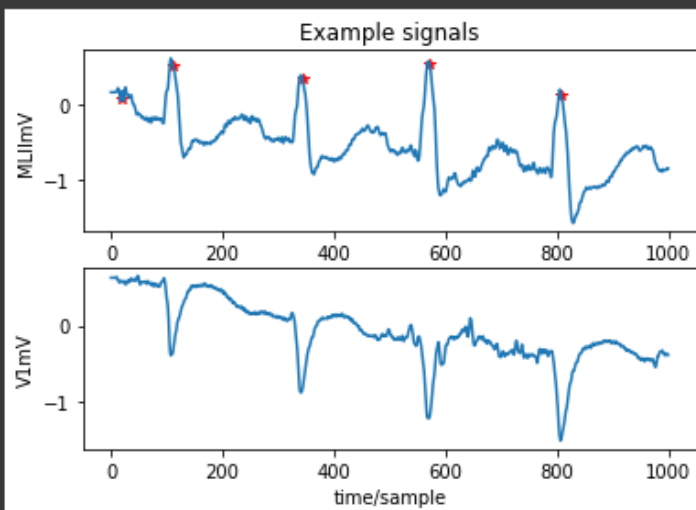
The native Python waveform-database (WFDB) package. A library of tools for reading, writing, and processing WFDB signals and annotations.

Core components of this package are based on the original WFDB specifications. This package does not contain the exact same functionality as the original WFDB package. It aims to implement as many of its core features as possible, with user-friendly APIs. Additional useful physiological signal-processing tools are added over time.

```
record = wfdb.rdrecord(example, sampfrom=0, sampto=1000)
ann = wfdb.rdann(example, ann_extension, sampto=1000)
```

همچنین برای اطمینان با دستور زیر نمودار موج را می کشیم.

```
12 # plot the record to screen
13 wfdb.plot_wfdb(record=record, title='Example signals', annotation=ann)
```



به این صورت مطمئن می شویم که امواج به درستی خوانده شده اند.

همانطور که در مقاله آمده است ابتدا با استفاده از روش autoregressive Berg از این دیتای خام فیچرها را استخراج می کنیم.

برای این کار از کتابخانه [Spectrum](https://pyspectrum.readthedocs.io/en/latest/index.html)<sup>1</sup> استفاده می کنیم.

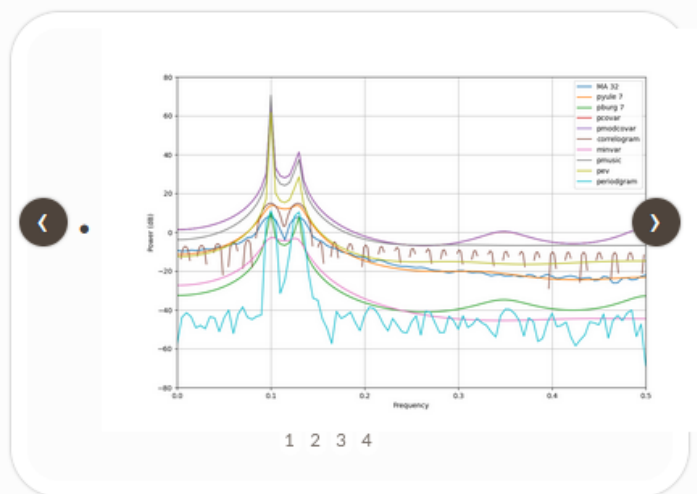
<sup>1</sup> <https://pyspectrum.readthedocs.io/en/latest/index.html>

## Quick installation

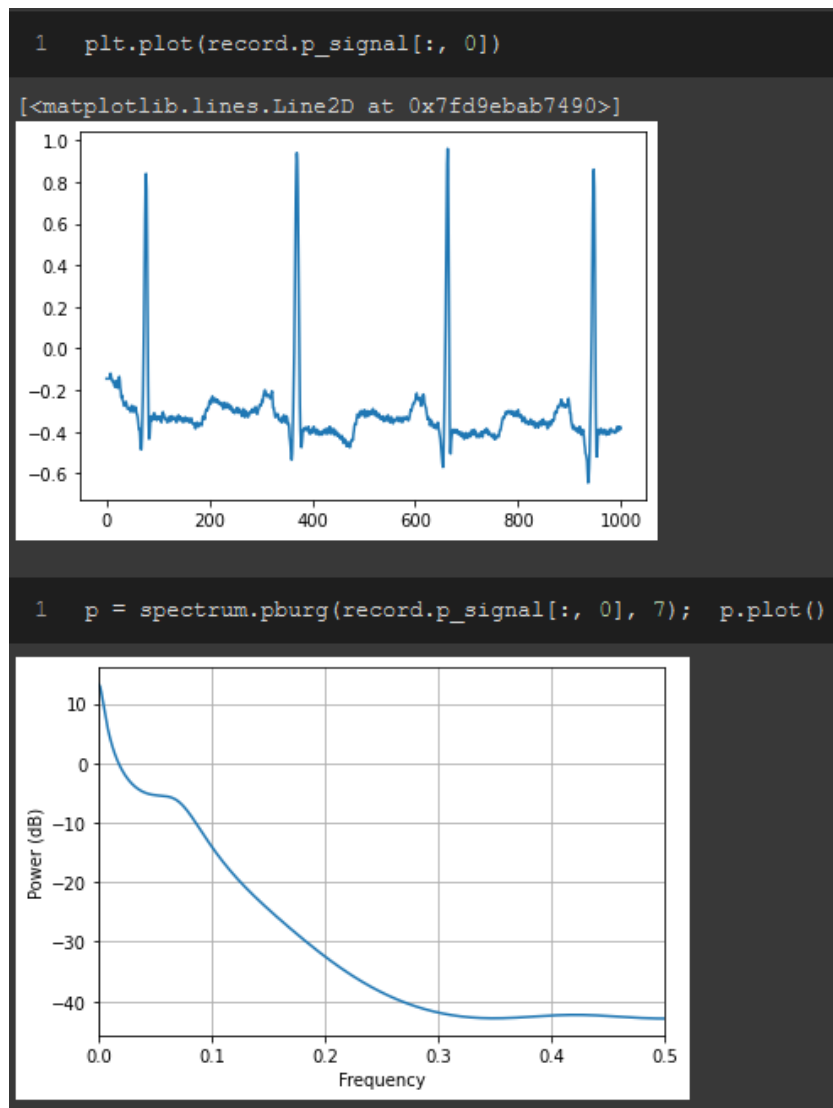
conda install spectrum

## Examples

Visit our example gallery or jump to the main documentation



با استفاده از این کتابخانه روش AR Berg را اعمال می کنیم که به شکل زیر است.



دیده می شود که این کار مشابه مثالی که داخل خود مقاله بود انجام شده است.

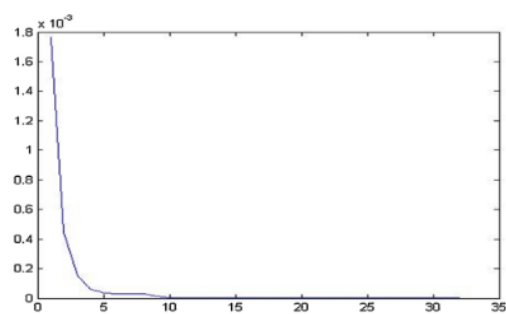


Fig. 3 – ECG data after AR Burg feature extraction is applied.

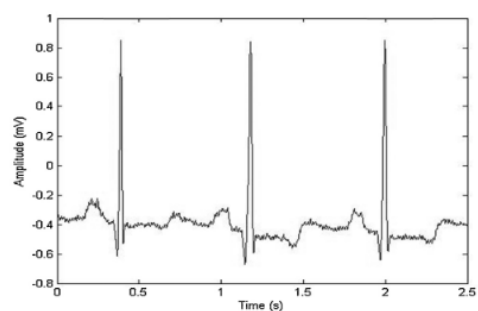


Fig. 2 – Raw ECG signal.

بنابراین به شکل زیر این تحلیل انجام شد و فیچرها به دست آمدند

```
record = wfdb.rdrecord(f'mitdb/{i}', sampfrom=sampfrom, sampto=sampfrom+step)
AR, P, k = spectrum.arburg(record.p_signal[:, 0], 32)
PSD = spectrum.arma2psd(AR)
features.append(10*log10(PSD[:2048]/max(PSD)))
```

برای داشتن طبقه‌بندی مناسب، این دیتاست ساخته شده را ابتدا shuffle می‌کنیم.

```
X, y = shuffle(features, targets)
```

الان دیتاها آماده طبقه‌بندی هستند که نتایج آن در بخش بعدی توضیح داده خواهند شد.



## طبقه‌بندی

پس از پردازش داده‌ها و به دست آوردن ویژگی‌ها چندین طبقه‌بند مختلف آموزش داده شد و به ازای هر یک یک `classification report`, `confusion matrix` و نمودار ROC کشیده شد. در ادامه نتایج حاصل از این طبقه‌بند‌ها و مقایسه آن‌ها میان هم آمده است.

### KNN

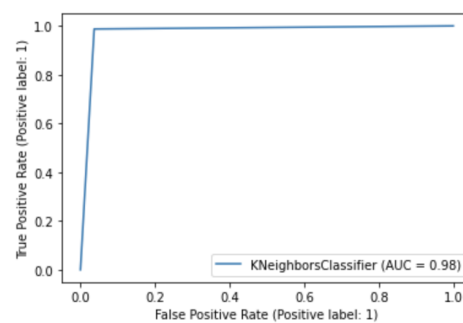
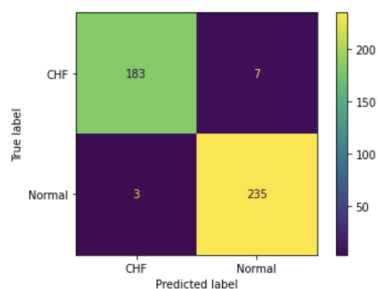
برای این روش تعداد همسایه‌های ۱، ۳، ۵، ۷ و ۱۰ مورد بررسی قرار گرفت نتایج به دست آمده به ازای هر  $k$  به صورت زیر است:

$k=1$

```
KNeighborsClassifier(n_neighbors=1)
precision    recall  f1-score   support

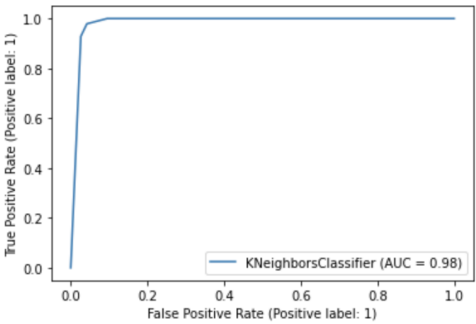
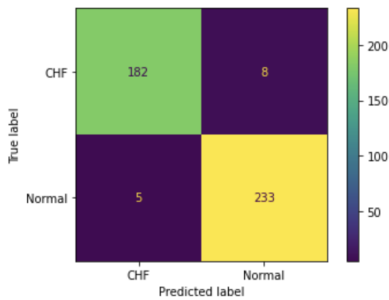
   CHF      0.98     0.96     0.97       190
  Normal    0.97     0.99     0.98       238

 accuracy    0.98       428
 macro avg   0.98     0.98     0.98       428
 weighted avg 0.98     0.98     0.98       428
```



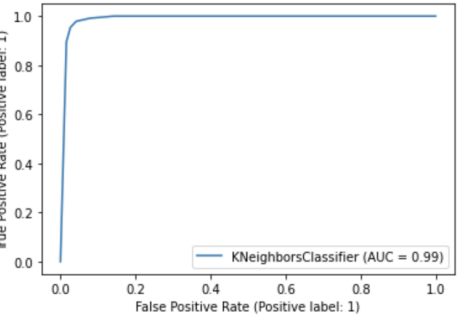
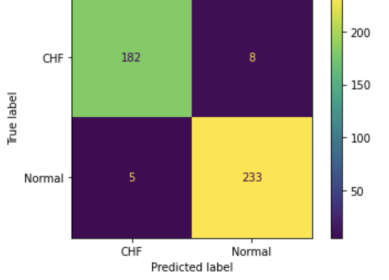
k=3

KNeighborsClassifier(n_neighbors=3)				
	precision	recall	f1-score	support
CHF	0.97	0.96	0.97	190
Normal	0.97	0.98	0.97	238
accuracy			0.97	428
macro avg	0.97	0.97	0.97	428
weighted avg	0.97	0.97	0.97	428



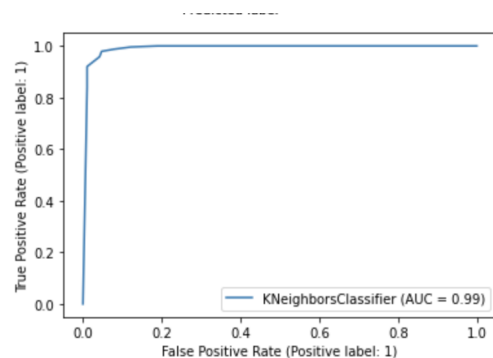
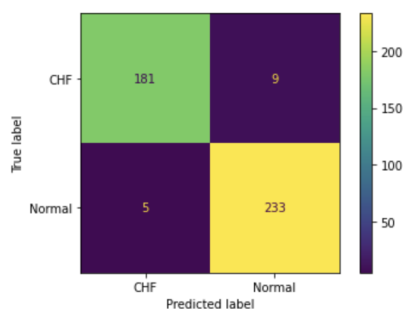
k=5

KNeighborsClassifier()				
	precision	recall	f1-score	support
CHF	0.97	0.96	0.97	190
Normal	0.97	0.98	0.97	238
accuracy			0.97	428
macro avg	0.97	0.97	0.97	428
weighted avg	0.97	0.97	0.97	428



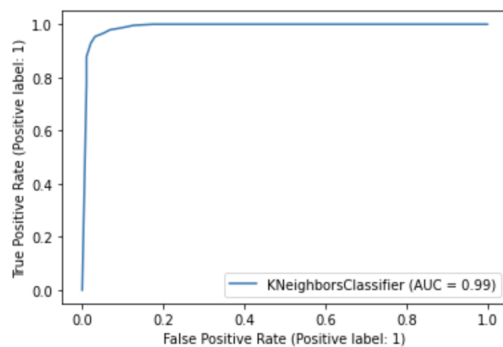
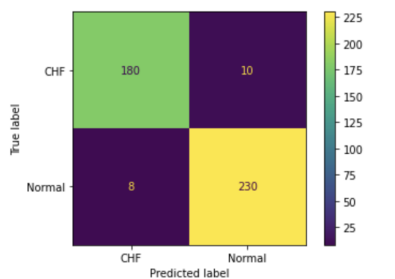
k=7

KNeighborsClassifier(n_neighbors=7)				
	precision	recall	f1-score	support
CHF	0.97	0.95	0.96	190
Normal	0.96	0.98	0.97	238
accuracy			0.97	428
macro avg	0.97	0.97	0.97	428
weighted avg	0.97	0.97	0.97	428



k=10

KNeighborsClassifier(n_neighbors=10)				
	precision	recall	f1-score	support
CHF	0.96	0.95	0.95	190
Normal	0.96	0.97	0.96	238
accuracy			0.96	428
macro avg	0.96	0.96	0.96	428
weighted avg	0.96	0.96	0.96	428



همان طور که مشاهده می شود دقت در KNN با تعداد همسایه های مختلف تقریباً نزدیک به هم هستند و نمودار ROC مربوط به آن ها تا حد خوبی به نمودار ایده آل نزدیک شده است که نشان می دهد مدل به خوبی توانسته است افراد نرمال را با کسانی که بیماری قلبی دارند متمایز کند.

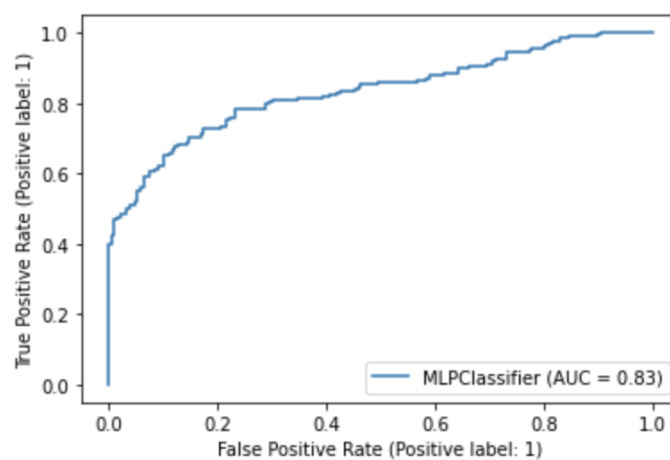
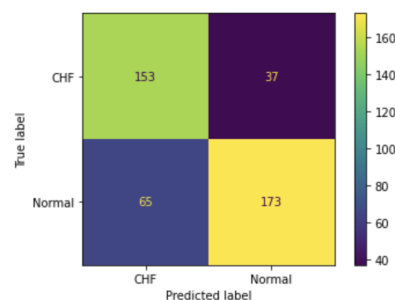
## Artificial Neural Network

در این روش یک شبکه عصبی با دو لایه پنهان آموزش داده شد و نتایج زیر به دست آمد:

```
MLPClassifier(early_stopping=True, hidden_layer_sizes=(100, 100), max_iter=500,
               validation_fraction=0.3, verbose=True)
precision    recall  f1-score   support

   CHF        0.70    0.81    0.75     190
 Normal        0.82    0.73    0.77     238

 accuracy          0.76     428
 macro avg         0.76    0.77    0.76     428
 weighted avg      0.77    0.76    0.76     428
```



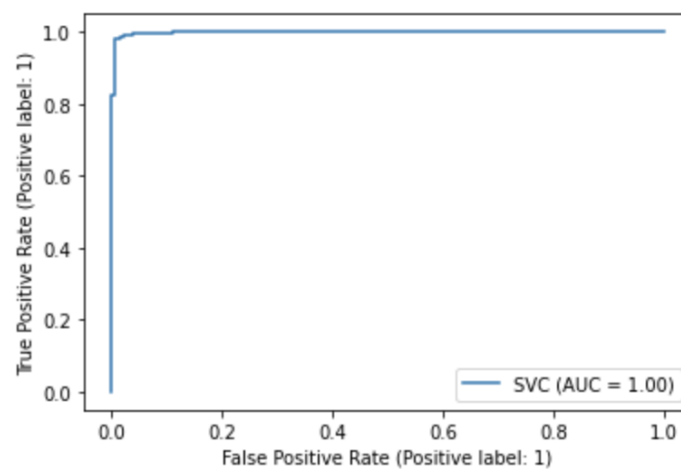
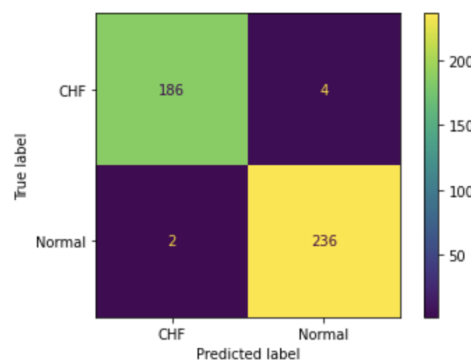
مشاهده می شود که دقت به دست آمده از شبکه عصبی حدود ۷۶ درصد است که به نسبت حالت قبل کمتر است. ممکن است اضافه کردن لایه های پنهان بیشتر و یا استخراج ویژگی های مناسب تر دقت این مدل را افزایش دهد.

## SVM

الگوریتم SVM سعی می کند داده های مربوط به دو کلاس را با حداکثر margin از یکدیگر جدا کند. از آن جایی که معمولا داده ها کاملا به صورت خطی جدا نمی شوند استفاده از کرنل ها می تواند بسیار کمک کننده باشد. به همین منظور و برای پیاده سازی این الگوریتم از دو کرنل rbf و puk استفاده شده است:

### RBFB

SVC()				
	precision	recall	f1-score	support
CHF	0.99	0.98	0.98	190
Normal	0.98	0.99	0.99	238
accuracy			0.99	428
macro avg	0.99	0.99	0.99	428
weighted avg	0.99	0.99	0.99	428



در این جا نیز به دقت خوبی رسیده ایم و نمودار AUC به مقدار بیشینه خود رسیده است و می بینیم که تنها در دو مورد فرد نرمال به بیمار نسبت داده شده است و در ۴ مورد شخص بیمار به اشتباه سالم تشخیص داده شده است.

در صورت استفاده از کرنل puk نیز نتایج زیر حاصل می شود که به نسبت حالت قبلی بدتر است.

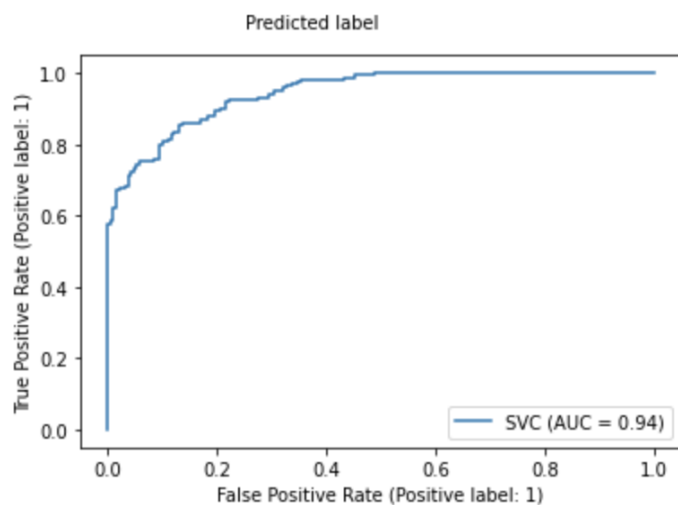
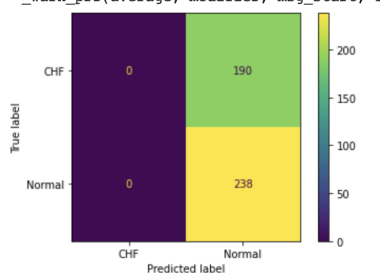
## PUK

```
SVC(kernel=<function PUK_kernel at 0x7fd9f5615dd0>)
precision    recall  f1-score   support

   CHF        0.00    0.00    0.00     190
  Normal    0.56    1.00    0.71     238

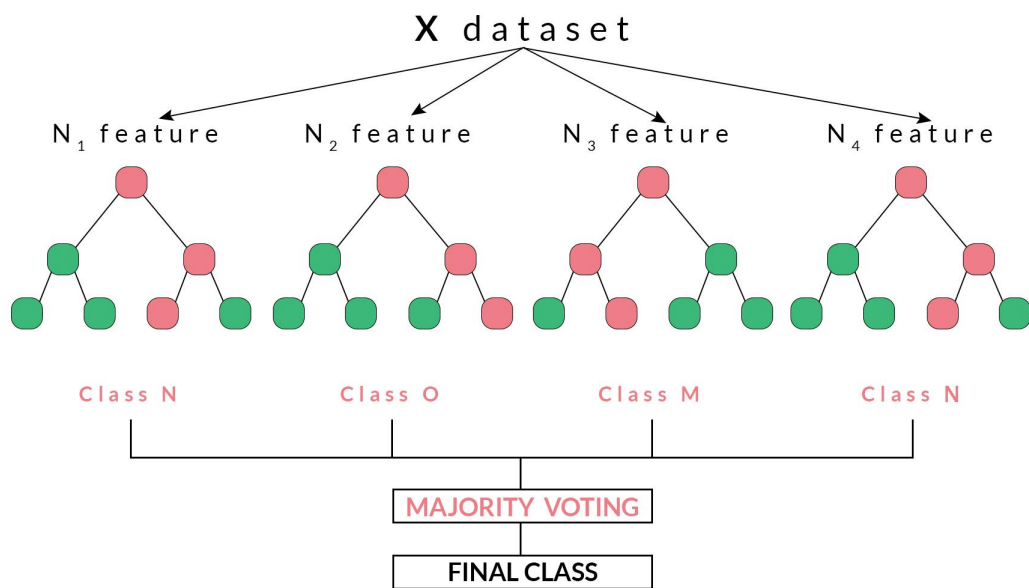
 accuracy          0.56     428
 macro avg         0.28    0.50    0.36     428
weighted avg         0.31    0.56    0.40     428
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:136: UserWarning:
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:136: UserWarning:
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:136: UserWarning:
  _warn_prf(average, modifier, msg_start, len(result))
```



## RandomForestClassifier

در این روش چندین طبقه بند وجود دارند و به هر کدام زیر مجموعه ای از داده ها داده می شود. هر کدام از طبقه بند ها نتایج را برای داده ی تست باز می گردانند و در نهایت کلاسی به داده مورد نظر تعلق می گیرد که بیش ترین رای را آورده باشد.



برای پیاده سازی این قسمت از ۲۰ درخت به عنوان طبقه بند های مدل استفاده شده است و هر کدام از این درخت ها از ۶ ویژگی استفاده می کنند. نتایج به دست آمده از اجرای این طبقه بند بر روی داده ها در صفحه ی بعد آمده است.

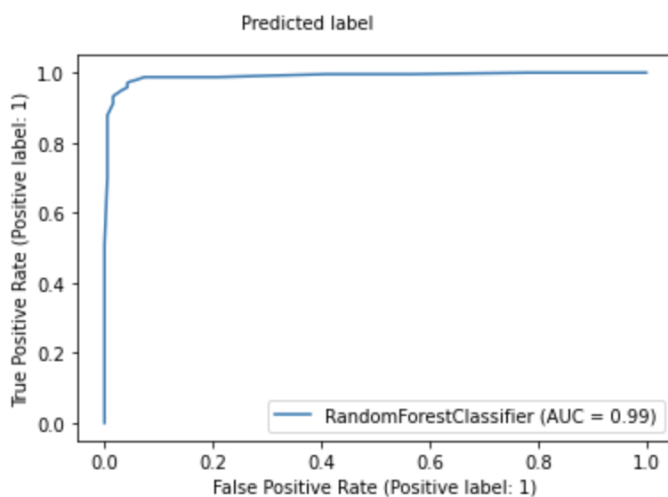
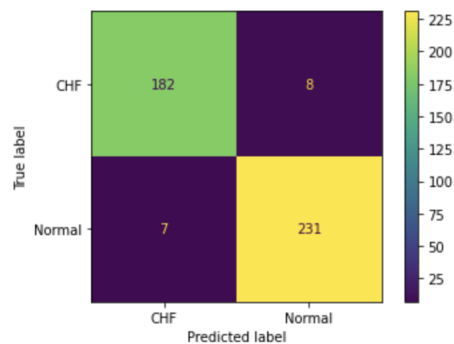
```

RandomForestClassifier(max_features=6, n_estimators=20)
precision    recall  f1-score   support

   CHF       0.96     0.96     0.96     190
  Normal     0.97     0.97     0.97     238

 accuracy          0.96     428
 macro avg       0.96     0.96     0.96     428
weighted avg       0.96     0.96     0.96     428

```



مشاهده می شود که این الگوریتم دقت خوبی بر روی داده ها به دست آورده و تا حد خوبی توانسته است پیش بینی را انجام دهد به گونه ای که مشاهده می شود تنها در ۱۵ مورد دچار اشتباه شده است. علاوه بر این نمودار ROC نیز بسیار به حالت ایده آل خود نزدیک است و مقدار AUC آن تقریباً برابر با یک است.



تمام کدهای نوشته شده در آدرس زیر موجود است و می‌توان با استفاده از کولب به راحتی اجرایشان کرد.

<https://colab.research.google.com/drive/1pLnFvDrrpmDAonBmyVLxkJ28s0TUINDw?usp=sharing>

مراجعی که در این پیاده سازی استفاده شد:

- <https://wfdb.readthedocs.io/en/latest/>
- [https://en.wikipedia.org/wiki/Autoregressive\\_model](https://en.wikipedia.org/wiki/Autoregressive_model)
- <https://pyspectrum.readthedocs.io>
- [https://www.sciencedirect.com/science/article/abs/pii/S0169260715303369?casa\\_token=QJTxB3mH5MAAAAAA:0ODrkB-5BdD\\_dLCeGo7mMX9Eo1ilzbLr4nHzF274hBYG5LV0MoWZBZlZwPkz3Zo4Vl2-F8liGQ#bib0405](https://www.sciencedirect.com/science/article/abs/pii/S0169260715303369?casa_token=QJTxB3mH5MAAAAAA:0ODrkB-5BdD_dLCeGo7mMX9Eo1ilzbLr4nHzF274hBYG5LV0MoWZBZlZwPkz3Zo4Vl2-F8liGQ#bib0405)
- Boser, Bernhard E, Guyon, Isabelle M, and Vapnik, Vladimir N. A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory, pages 144–152, 1992.
- McCulloch, Warren S and Pitts, Walter. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133, 1943.