Πανεπιστήμιο Πειραιώς

Τμήμα Στατιστικής & Ασφαλιστικής Επιστήμης

Ακαδημαϊκό έτος 2020-21 (χειμερινό εξάμηνο)

Υποχρεωτική ατομική εργασία εξαμήνου

Μάθημα: «Θέματα Επιστήμης Δεδομένων»

Ονοματεπώνυμο : Τσιαμασιώτη Μυρτώ-Αγλαΐα (Σ17198)

Contents

ΣΥΝΟΨΗ

Η πρόβλεψη ενδεχομένων καρδιοπαθειών με τη χρήση δεδομένων, δίνει ένα πολύ σημαντικό εργαλείο στην ιατρική. Συγκεκριμένα, με το να μπορεί ο γιατρός να ταξινομήσει τους ασθενείς σε αυτούς με χαμηλή ή υψηλή επικινδυνότητα, του επιτρέπει να περιορίσει ένα πληθυσμό σε ανθρώπους που πραγματικά έχουν ανάγκη να εξεταστούν περαιτέρω.

Η παρούσα εργασία αποτελεί μια πρώτη προσπάθεια στο παραπάνω πρόβλημα. Διαθέτουμε ένα σημαντικό αριθμό δεδομένων ασθενών καθώς και τα αποτελέσματα τους (δηλαδή αν έχει καρδιοπάθεια ή όχι).

Πάνω στα δεδομένα αυτά φτιάχνουμε αρχικά ένα αριθμό μοντέλων κατηγοριοποίησης και επιλέγουμε σαν μετρική το recall. Από αυτά ξεχωρίζει το Logistic Regression δίνοντας ένα recall της τάξης του 0.82.

Στη συνέχεια, επιχειρήθηκε μια συσταδοποίηση προκειμένου να καλυφθούν και περιπτώσεις στις οποίες το τελικό αποτέλεσμα (labels) δεν είναι διαθέσιμο. Από τη συσταδοποίηση προκύπτει ότι μπορεί με κάποια σφάλματα να προβλεφθούν οι ετικέτες, μέσω του αλγορίθμου kmeans.

Τέλος εφαρμόσαμε κανόνες συσχέτισης προκειμένου να εντοπίσουμε ποια υποσύνολα δεδομένων εμφανίζονται συχνότερα στο dataset, και να δούμε κατά πόσο διαφοροποιούνται αυτά μεταξύ ασθενών και υγειών ατόμων.

ΕΙΣΑΓΩΓΗ

Μας δίνεται το dataset “Heart Disease”, και πρέπει να διεξάγουμε από αυτό συμπεράσματα. Σκοπός μας είναι να προβλέψουμε με βάση τα δοθέντα δεδομένα παλιών ασθενών, εάν ένα καινούριο άτομο είναι ασθενής ή όχι. Τα δεδομένα του καινούριου ατόμου θα είναι καινούρια για το μοντέλο, και με βάση τα παλιότερα θα καταλήξουμε στο εάν νοσεί. Για να το καταφέρουμε αυτό, θα πρέπει να φτιάξουμε ένα μοντέλο, το οποίο θα δίνει τα καλύτερα πιθανά αποτελέσματα για τα δεδομένα μας.

Ας ξεκινήσουμε ρίχνοντας μια πρώτη ματιά στα δεδομένα μας. Από την ιστοσελίδα του University of California Irvine, συγκεντρώνω τις παρακάτω πληροφορίες για το dataset:

Name	Description	Type
age	Age in years	continuous
sex	sex (1 = male; 0 = female)	Binary categorical
cp	chest pain type -- Value 1: typical angina -- Value 2: atypical angina -- Value 3: non-anginal pain -- Value 4: asymptomatic	categorical
trestbps	resting blood pressure (in mm Hg on admission to the hospital)	continuous
chol	serum cholestoral in mg/dl	continuous
fbs	(fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)	Binary categorical
restecg	resting electrocardiographic results -- Value 0: normal -- Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV) -- Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria	categorical
thalach	maximum heart rate achieved	continuous
exang	exercise induced angina (1 = yes; 0 = no)	Binary categorical
oldpeak	ST depression induced by exercise relative to rest	continuous
slope	the slope of the peak exercise ST segment -- Value 1: upsloping -- Value 2: flat -- Value 3: downsloping	Categorical
ca	number of major vessels (0-3) colored by flourosopy	Categorical (ordinal)
thal	3 = normal; 6 = fixed defect; 7 = reversable defect	categorical
target	refers to the presence of heart disease in the patient, it is integer valued from 0 (no presence) to 4.	Categorical → binary

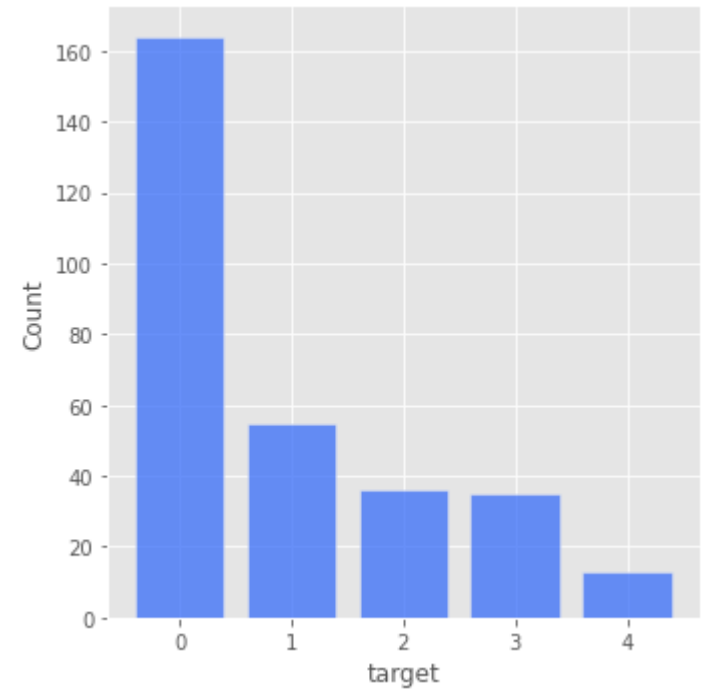
Part 1: Data Preprocessing

Το πρώτο βήμα αφού έχουμε δει κάποιες αρχικές πληροφορίες για τις μεταβλητές μας, είναι να διερευνήσουμε περεταίρω πώς λειτουργεί η κάθε μεταβλητή στο dataset, εξετάζοντας δηλαδή πώς κατανέμεται και εάν χρειάζονται κάποια μετατροπή οι τιμές της. Έπειτα, θα δούμε και πώς συσχετίζεται η καθεμία από αυτές με την μεταβλητή target, την οποία θέλουμε να προβλέψουμε.

Target Variable (Y)

Η μεταβλητή target, ή αλλιώς η μεταβλητή που θέλουμε να προβλέψουμε, μας δίνει σε μία κλίμακα από το 0 έως το 4 την πιθανότητα ο ασθενής να έχει καρδιακή πάθηση, με το 0 να σημαίνει ότι το άτομο είναι υγιές, και το 1 έως το 4 να σημαίνουν ότι το άτομο νοσεί. Είναι δηλαδή, μια κατηγορική μεταβλητή που παίρνει 5 τιμές.

Αρχικά, ας εξετάσουμε εάν η στήλη αυτή έχει missing values. Με την εντολή count βλέπουμε πως έχουμε 303 στοιχεία(όσες και οι γραμμές μας), και με την εντολή unique βλέπουμε πως αυτές θα είναι 0, 1, 2, 3, ή 4. Ας εξετάσουμε τώρα πως κατανέμονται οι τιμές αυτές στα δεδομένα μας :



Όπως διαπιστώσαμε νωρίτερα, βλέπουμε και εδώ ότι η στήλη target, ή αλλιώς η στήλη που μας λέει εάν το άτομο πάσχει ή όχι από καρδιασθένεια, έχει 5 διαφορετικές τιμές, τις 0, 1, 2, 3 και 4.

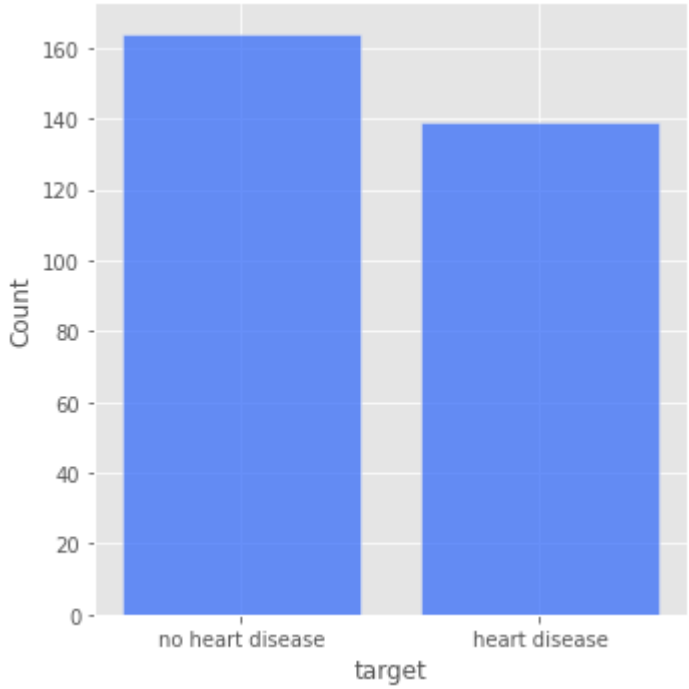
Το γεγονός ότι έχουμε 4 διαφορετικές τιμές στην target αποτελεί πρόβλημα για το μοντέλο που θέλουμε να φτιάξουμε. Αρχικά, τα πλήθος των δεδομένων που έχουμε δεν επαρκεί για να

προβλέψουμε 5 διαφορετικές τιμές. Επιπλέον, για να λειτουργήσει σωστά το μοντέλο μας, θα πρέπει κάθε τιμή του target να έχει ίσες παρατηρήσεις, δηλαδή η μεταβλητή μας να είναι ισορροπημένη. Εάν η μεταβλητή μας δεν είναι ισορροπημένη, όπως εδώ, τότε το μοντέλο θα έχει πολύ περισσότερα δεδομένα για να προβλέψει την 0, και ακόμα λιγότερα για να προβλέψει τις 1, 2, 3 και 4. Ένα τέτοιο μοντέλο θα έχει την τάση να μας δίνει συχνά λάθος πρόβλεψη 0, και πολύ σπανίως κάποια άλλη πρόβλεψη, καθώς για την 0 έχει διπλάσια δεδομένα απ' ότι για όλες τις υπόλοιπες τιμές μαζί. Τέλος, το δικό μας ερώτημα έχει μόνο δύο, και όχι 5, πιθανές απαντήσεις : Πάσχει το άτομο από την ασθένεια ή όχι;

Λόγω των παραπάνω, θα μετατρέψουμε την μεταβλητή target από κατηγορική με 5 κατηγορίες σε binary μεταβλητή με δύο. Θα παίρνει δηλαδή μόνο τις τιμές

0 = δεν πάσχει και 1 = πάσχει από την ασθένεια.

Αντιστοιχίζουμε το 0 με την τιμή 0, και το 1 με τις τιμές 1, 2, 3 και 4, μέσω του map. Μετά από αυτό, η μεταβλητή target έχει ως εξής :



Η target έχει πλέον μόνο δύο τιμές :

0 : no heart disease
1 : heart disease,

Παρατηρούμε πως οι τιμές 1, 2, 3, και 4 που έχουν πλέον συγχωνευτεί στην 1, είναι πολύ κοντά σε πλήθος με την τιμή 0. Λόγω του ότι τα 0,1 είναι ισομοιρασμένα, θα έχουμε πολύ πιο έγκυρες προβλέψεις. Εν κατακλείδι, το μοντέλο μας τώρα θα δίνει την πρόβλεψη 0, εάν το άτομο είναι υγιές, και 1, εάν δεν είναι.

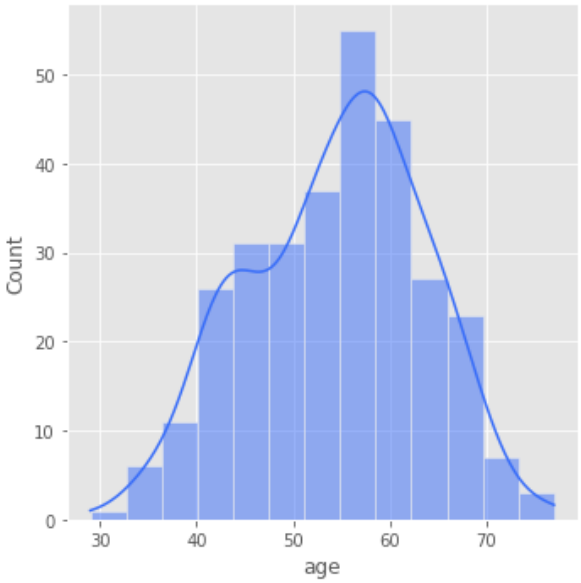
Age Variable

Η μεταβλητή age μας δίνει την ηλικία του ατόμου σε χρόνια. Είναι συνεχής μεταβλητή.

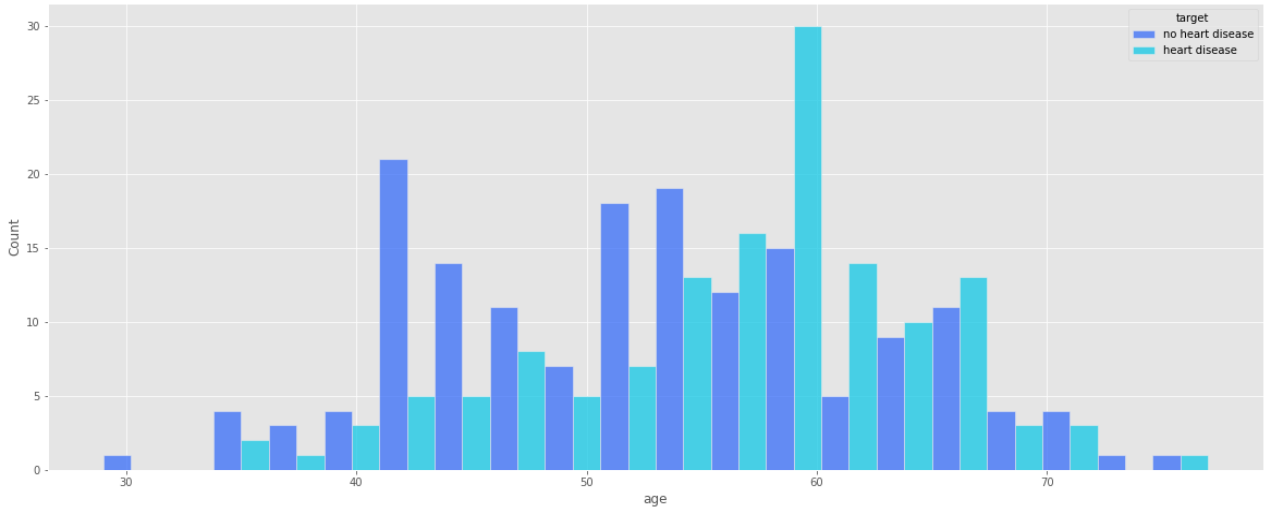
Την εξετάζουμε αρχικά για missing values. Όπως και πριν, βλέπουμε πως έχει 303 γραμμές(count), και όλα της τα μοναδικά στοιχεία είναι numeric(unique). Επομένως, δεν έχουμε missing values.

Με την εντολή describe, και φτιάχνοντας το ιστόγραμμά της, παίρνουμε τα εξής στοιχεία για την κατανομή της:

```
count    303.000000
mean      54.438944
std        9.038662
min       29.000000
25%       48.000000
50%       56.000000
75%       61.000000
max       77.000000
Name: age, dtype: float64
```



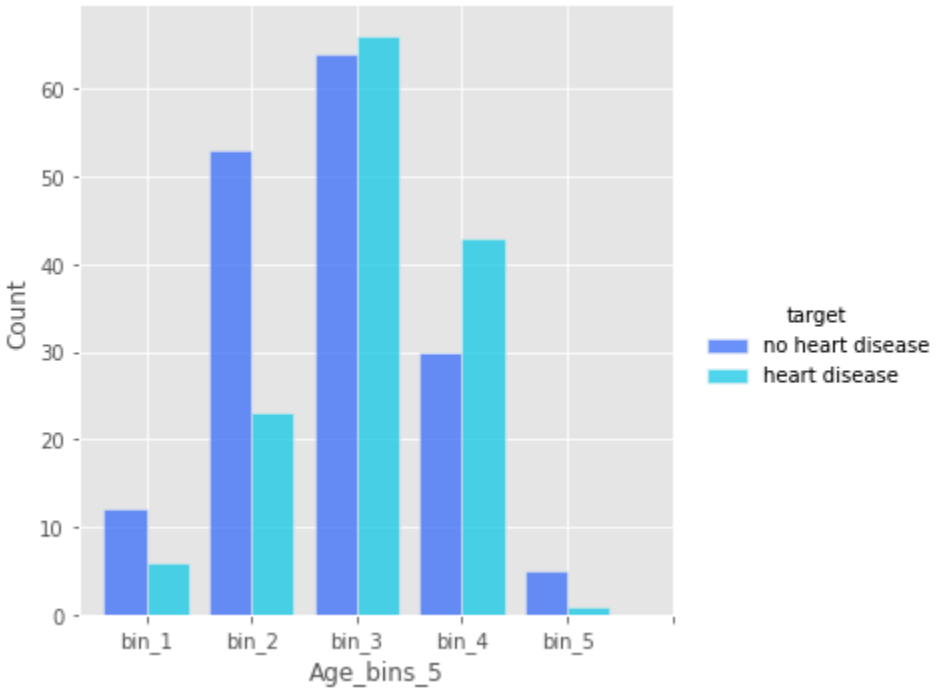
Μπορούμε να δούμε πως η ηλικία ακολουθεί μία κατανομή αρκετά κοντά στην κανονική. Ας εξετάσουμε τώρα πως κατανέμεται η ηλικία σε σχέση με το εάν το άτομο πάσχει από την ασθένεια ή όχι.



Από το γράφημα αυτό μπορούμε να δούμε πως η ηλικία των ατόμων που έχουν την ασθένεια τείνει να είναι μεγαλύτερη, ενώ τα άτομα μικρότερης ηλικίας έχουν λιγότερες πιθανότητες να είναι ασθενείς. Όπως περιμέναμε λοιπόν, η μεταβλητή age είναι παράγοντας που επηρεάζει την μεταβλητή target, καθώς αυτή παίρνει διαφορετικές τιμές ανάλογα την ηλικία.

Η age είναι μία μεταβλητή που θα μπορούσαμε να χωρίσουμε σε bins, καθώς όπως είδαμε παραπάνω, οι καρδιοπαθείς τείνουν να είναι μεγαλύτερης ηλικίας. Συνεπώς, αν κάναμε διακριτοποίηση στις μικρότερες και μεγαλύτερες ηλικίες, θα φαινόταν πιο καθαρά τα το εάν το άτομο νοσεί ή όχι στις μεγαλύτερες ηλικίες. Διαλέγω το q με βάση το διάγραμμα της κατανομής του age σε σχέση με το target.

Για q = [0, 40, 50, 60, 70, 100], τα 5 bins διαμορφώνονται ως εξής :

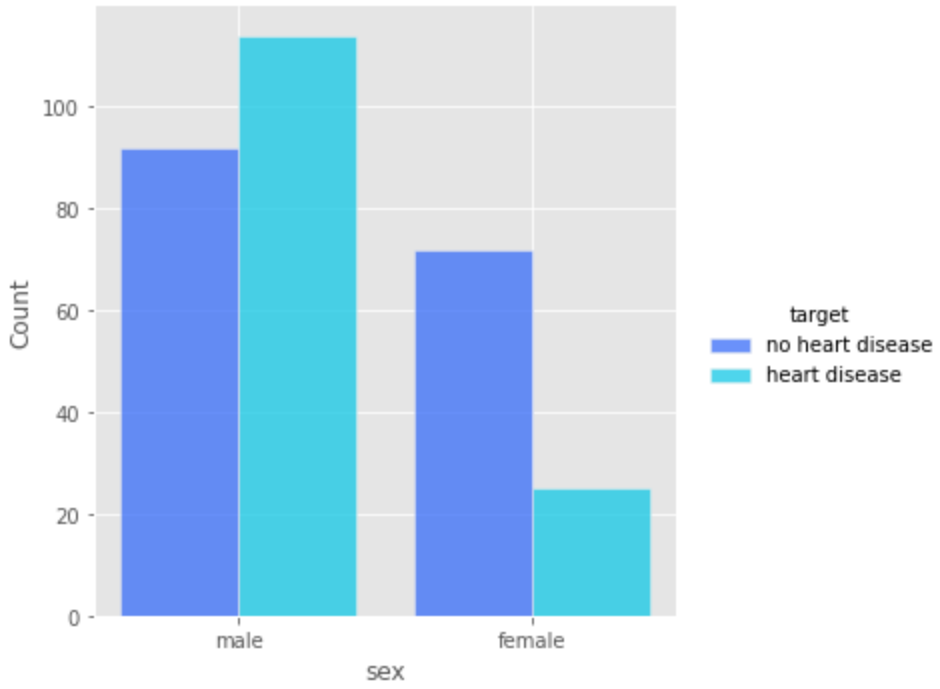


Sex Variable

Η μεταβλητή sex μας δίνει το φύλο του ατόμου, και παίρνει τις τιμές 1 → άντρας, και 0 → γυναίκα. Είναι κατηγορική binary μεταβλητή. (παίρνει δηλαδή μόνο δύο τιμές).

Ελέγχουμε μήπως υπάρχουν missing values. Η στήλη sex έχει 303 γραμμές, με μοναδικές τιμές τις 1, 0. Άρα, δεν έχουμε missing values.

Ας εξετάσουμε τώρα πως κατανέμεται το φύλο σε σχέση με την target, για να δούμε κατά πόσο συμβάλλει στην πρόβλεψη της.



Αρχικά, διαπιστώνουμε ότι οι άντρες είναι περισσότεροι από τις γυναίκες. Πράγματι, οι άντρες αποτελούν το 68% των ατόμων στο dataset, ενώ οι γυναίκες το 32% (value_counts). Μπορούμε να δούμε πως οι άντρες που πάσχουν από καρδιοπάθεια είναι περισσότεροι από αυτούς που δεν πάσχουν. Αντιθέτως, οι γυναίκες που πάσχουν από την ασθένεια είναι πολύ λιγότερες σε σχέση με αυτές που δεν πάσχουν. Οι άντρες ασθενείς φαίνεται να καταβάλλουν το μεγαλύτερο ποσοστό του dataset, ενώ οι γυναίκες ασθενείς το μικρότερο.

CP Variable

Η μεταβλητή cp αναφέρεται στον πόνο που έχει ο ασθενής στο στήθος. Αυτός ο πόνος, ανάλογα του είδους του, χωρίζεται σε 4 κατηγορίες :

1 : τυπική στηθάγχη (πόνος που οφείλεται στην μη επαρκή τροφοδότηση της καρδιάς με αίμα)

2 : μη τυπική στηθάγχη

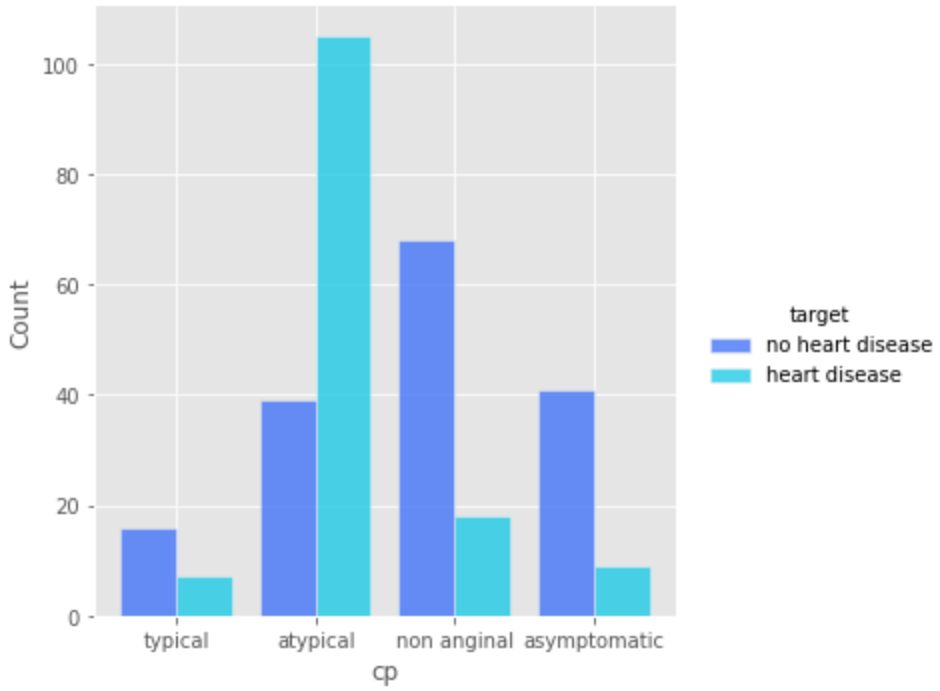
3 : πόνος που δεν οφείλεται σε στηθάγχη

4 : ασυμπτωματικός (χωρίς πόνο)

Είναι κατηγορική μεταβλητή.

Όπως πάντα, ελέγχουμε αν υπάρχουν ελλιπείς τιμές. Η στήλη cp έχει 303 γραμμές, και τα μοναδικά της στοιχεία είναι 1, 2, 3 και 4. Συνεπώς, δεν έχουμε ελλιπείς τιμές.

Εξετάζουμε την σχέση που έχει η cp με την target :



Από το παραπάνω διάγραμμα, παρατηρούμε ότι οι ασθενείς που έχουν μη τυπική στηθάγχη(atypical), έχουν περισσότερες πιθανότητες να έχουν καρδιασθένεια. Επιπλέον, οι ασθενείς που έχουν πόνο που δεν οφείλεται σε στηθάγχη(non anginal), όπως και ασυμπτωματικό(asymptomatic) και τυπικό(typical) πόνο, έχουν λιγότερες πιθανότητες να νοσούν.

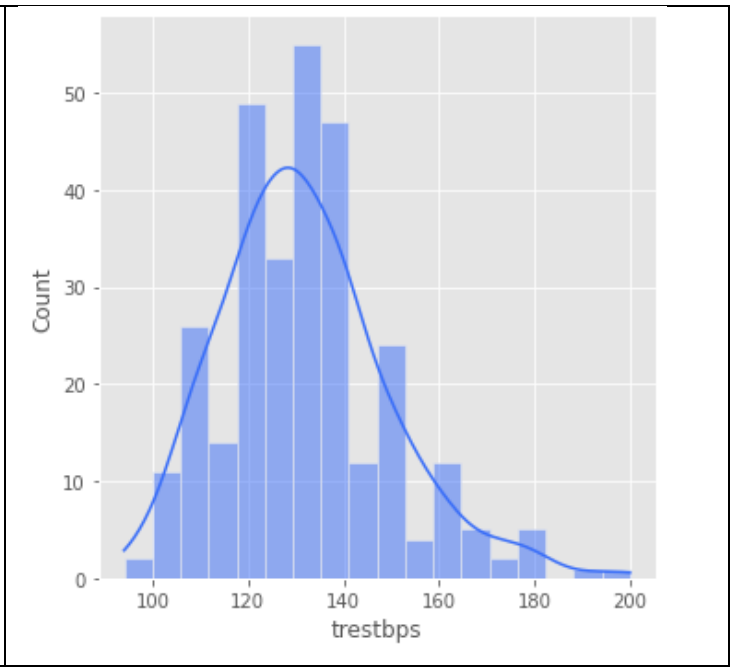
Trestbps Variable

Η μεταβλητή trestbps αναφέρεται στην πίεση του αίματος σε mm Hg, όταν το άτομο καταφτάνει στο νοσοκομείο. Είναι συνεχής μεταβλητή.

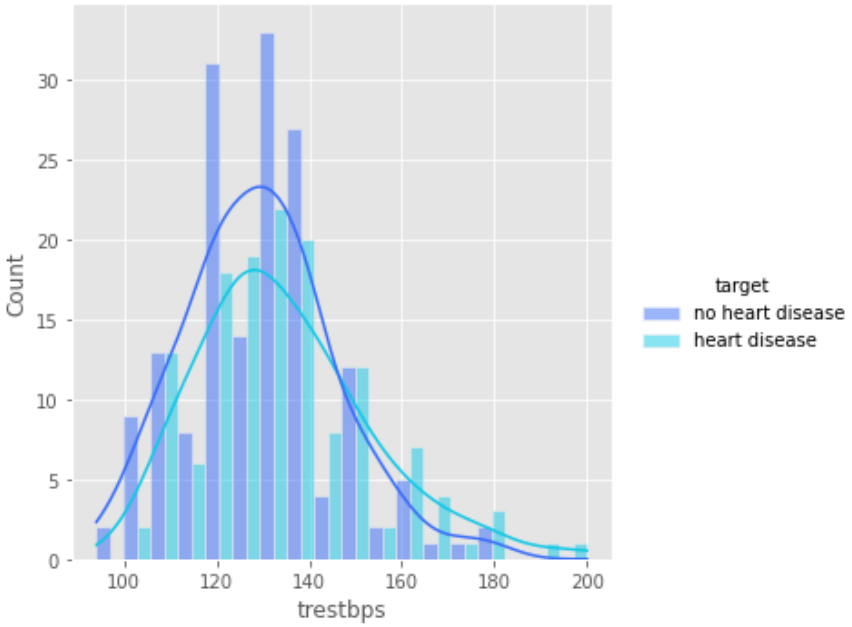
Ελέγχουμε εάν υπάρχουν ελλιπείς τιμές. Η στήλη trestbps έχει 303 γραμμές, και όλες οι μοναδικές τιμές της είναι numerical. Συνεπώς, δεν έχουμε ελλιπείς τιμές.

Με την εντολή describe, και φτιάχνοντας το ιστόγραμμά της, παίρνουμε τα εξής στοιχεία για την κατανομή της:

```
count    303.000000
mean     131.689769
std       17.599748
min       94.000000
25%      120.000000
50%      130.000000
75%      140.000000
max      200.000000
Name: trestbps, dtype: float64
```



Ας εξετάσουμε τώρα και τη σχέση της με την target :



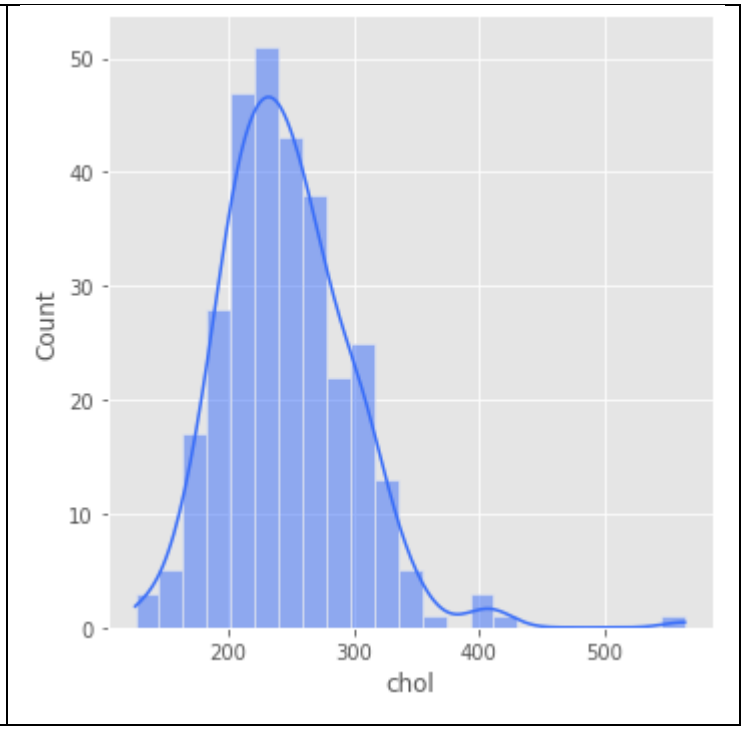
Chol Variable

Η μεταβλητή chol αναφέρεται στην χοληστερίνη στο αίμα, σε mg/dl. Είναι συνεχής μεταβλητή.

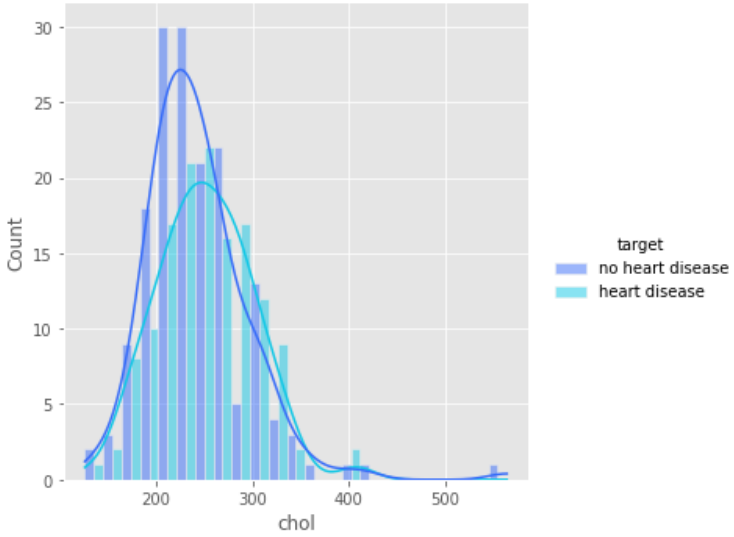
Ελέγχουμε εάν υπάρχουν ελλιπείς τιμές. Η στήλη trestbps έχει 303 γραμμές, και όλες οι μοναδικές τιμές της είναι numerical. Συνεπώς, δεν έχουμε ελλιπείς τιμές.

Με την εντολή describe, και φτιάχνοντας το ιστόγραμμά της, παίρνουμε τα εξής στοιχεία για την κατανομή της:

```
count    303.000000
mean     246.693069
std       51.776918
min      126.000000
25%      211.000000
50%      241.000000
75%      275.000000
max      564.000000
Name: chol, dtype: float64
```



Ας εξετάσουμε τώρα και τη σχέση της με την target :



Όπως είναι λογικό, παρατηρούμε ότι υψηλά επίπεδα χοληστερίνης σχετίζονται με περισσότερους ασθενείς, και χαμηλότερα επίπεδα με υγιείς. Η τιμή που είναι μεγαλύτερη από

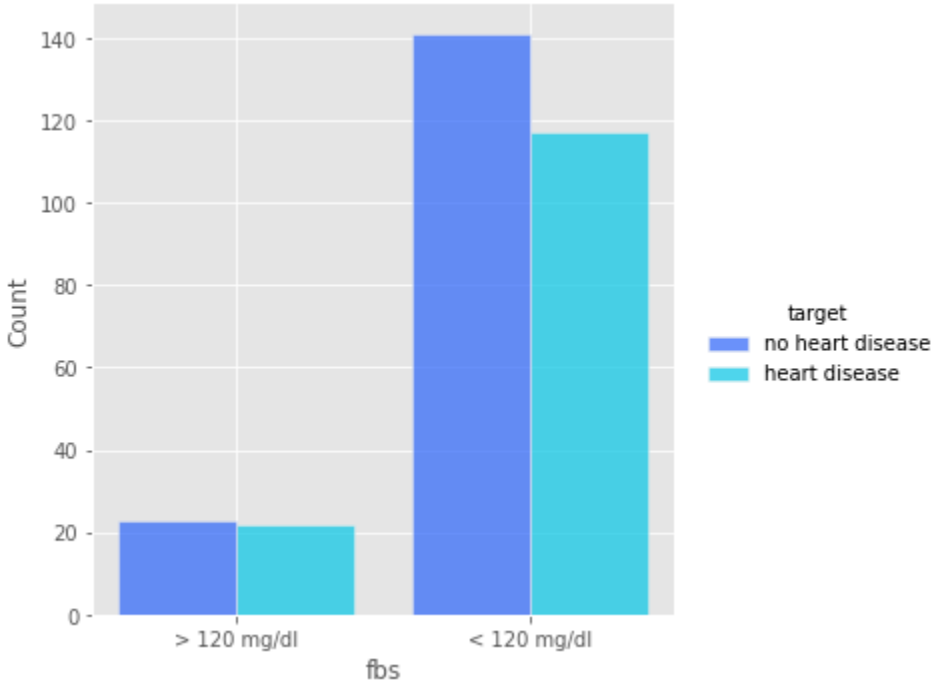
το 500 θα μπορούσε να θεωρηθεί outlier, αλλά λόγω του ότι είναι μία ρεαλιστική τιμή της χοληστερίνης, δεν θα την αφαιρέσω.

Fbs Variable

Η μεταβλητή fbs αναφέρεται στα επίπεδα ζάχαρης στο αίμα, όταν το άτομο ακόμα δεν έχει τραφεί. Παίρνει την τιμή 1 εάν η ζάχαρη στο αίμα είναι μεγαλύτερη από 120 mg/dl, και 0 εάν είναι μικρότερη. Είναι binary κατηγορική μεταβλητή.

Ελέγχουμε μήπως υπάρχουν missing values. Η στήλη fbs έχει 303 γραμμές, με μοναδικές τιμές τις 1, 0. Άρα, δεν έχουμε missing values.

Ας εξετάσουμε τώρα πως κατανέμεται σε σχέση με την target, για να δούμε κατά πόσο συμβάλλει στην πρόβλεψη της.



Εδώ βλέπουμε ότι τα περισσότερα άτομα έχουν fbs μικρότερη από 120 mg/dl. Εάν η fbs είναι μικρότερη από 120, υπάρχουν παραπάνω πιθανότητες το άτομο να μη νοσεί. Εάν η fbs είναι μεγαλύτερη από 120, δεν έχουμε σοβαρές ενδείξεις για την τιμή που θα πάρει η target.

Restecg Variable

Η μεταβλητή αυτή αναφέρεται στα αποτελέσματα ηλεκτροκαρδιογραφικής εξέτασης. Οι τιμές που παίρνει είναι :

0 : κανονικές

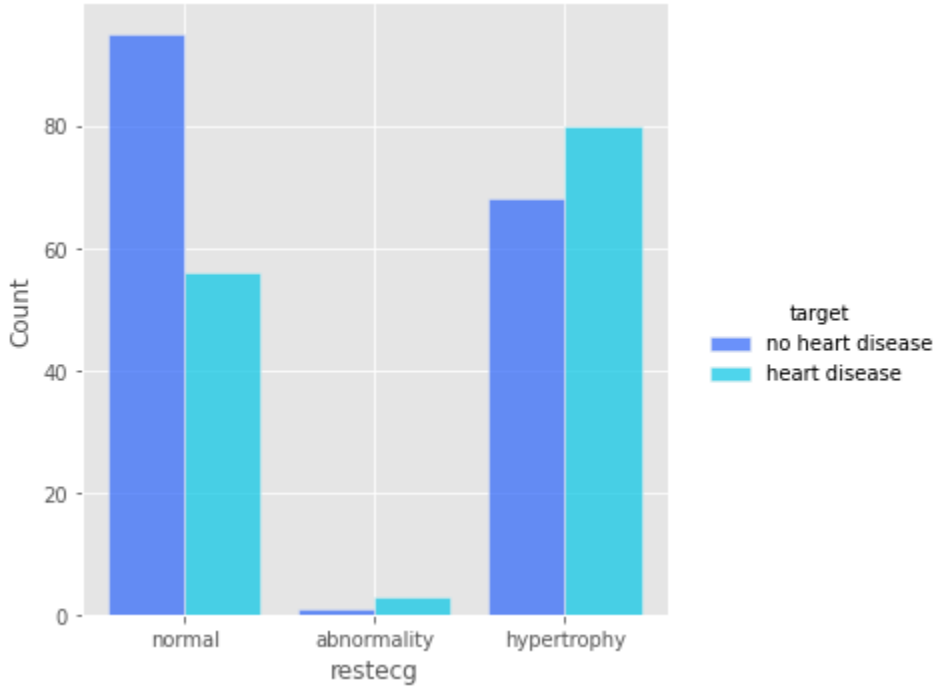
1 : έχουν κάποια ανωμαλία στο κύμα ST-T

2 : δείχνουν πιθανή ή σίγουρη υπερτροφία της αριστερής κοιλίας της καρδιάς, με βάση το κριτήριο του Estes

Είναι κατηγορική μεταβλητή.

Ελέγχουμε μήπως υπάρχουν missing values. Η στήλη restecg έχει 303 γραμμές, με μοναδικές τιμές τις 2, 1, 0. Άρα, δεν έχουμε missing values.

Ας εξετάσουμε τώρα πως κατανέμεται σε σχέση με την target, για να δούμε κατά πόσο συμβάλλει στην πρόβλεψη της.



Εδώ βλέπουμε ότι, εάν οι εξετάσεις είναι κανονικές(normal), τα περισσότερα άτομα δεν νοσούν, ενώ εάν υπάρχει ένδειξη υπερτροφίας της αριστερής κοιλίας(hypertrophy), τα περισσότερα άτομα νοσούν. Οι παρατηρήσεις με κάποια ανωμαλία(abnormality) στο κύμα ST-T φαίνεται να είναι σπάνιες, και να μην επηρεάζουν ιδιαίτερα την target.

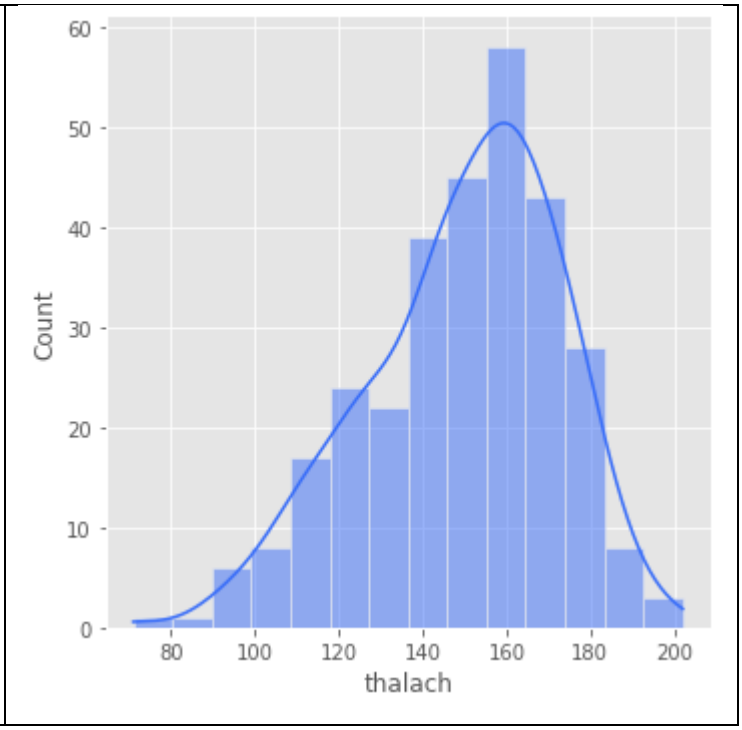
Thalach Variable

Η μεταβλητή thalach αναφέρεται στην υψηλότερο αριθμό ρυθμού καρδιακών παλμών που φτάνει ο ασθενής. Είναι συνεχής μεταβλητή.

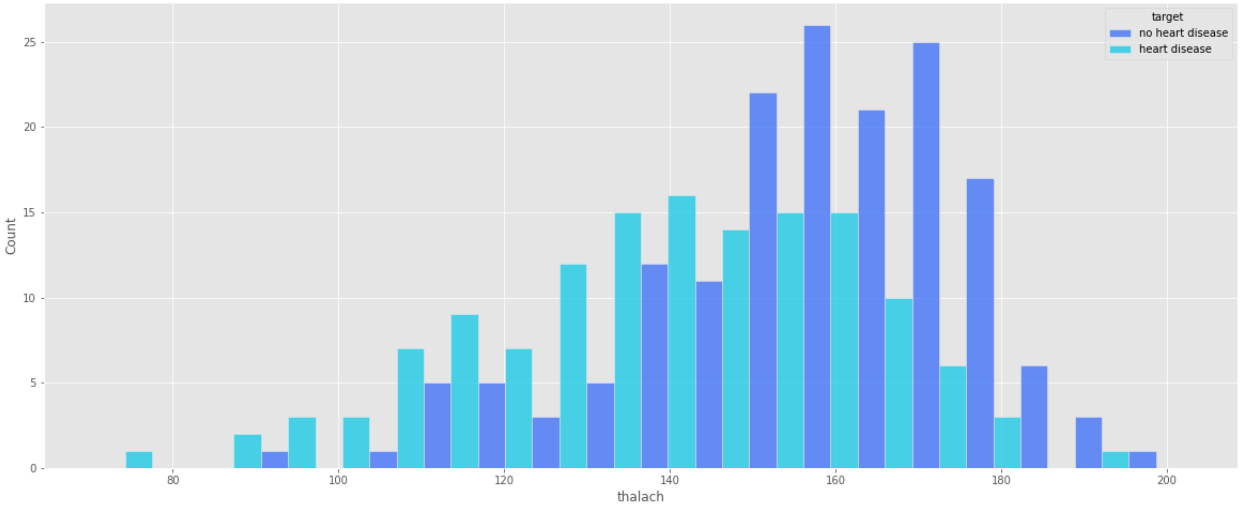
Ελέγχουμε εάν υπάρχουν ελλιπείς τιμές. Η στήλη thalach έχει 303 γραμμές, και όλες οι μοναδικές τιμές της είναι numerical. Συνεπώς, δεν έχουμε ελλιπείς τιμές.

Με την εντολή describe, και φτιάχνοντας το ιστόγραμμά της, παίρνουμε τα εξής στοιχεία για την κατανομή της:

```
count    303.000000
mean     149.607261
std       22.875003
min       71.000000
25%      133.500000
50%      153.000000
75%      166.000000
max      202.000000
Name: thalach, dtype: float64
```



Ας εξετάσουμε τώρα και την σχέση που έχουν οι τιμές της thalach με αυτές που παίρνει η target:



Απ'ότι φαίνεται, οι μεγαλύτερες αυτές τιμές της thalach είναι καλό σημάδι : τα περισσότερα άτομα στο δείγμα με υψηλή thalach, δηλαδή άτομα που μπορούν να επιτύχουν υψηλό ρυθμό καρδιακών παλμών, είναι υγιή. Αντιθέτως, φαίνεται πως η χαμηλότερη thalach είναι μία ένδειξη

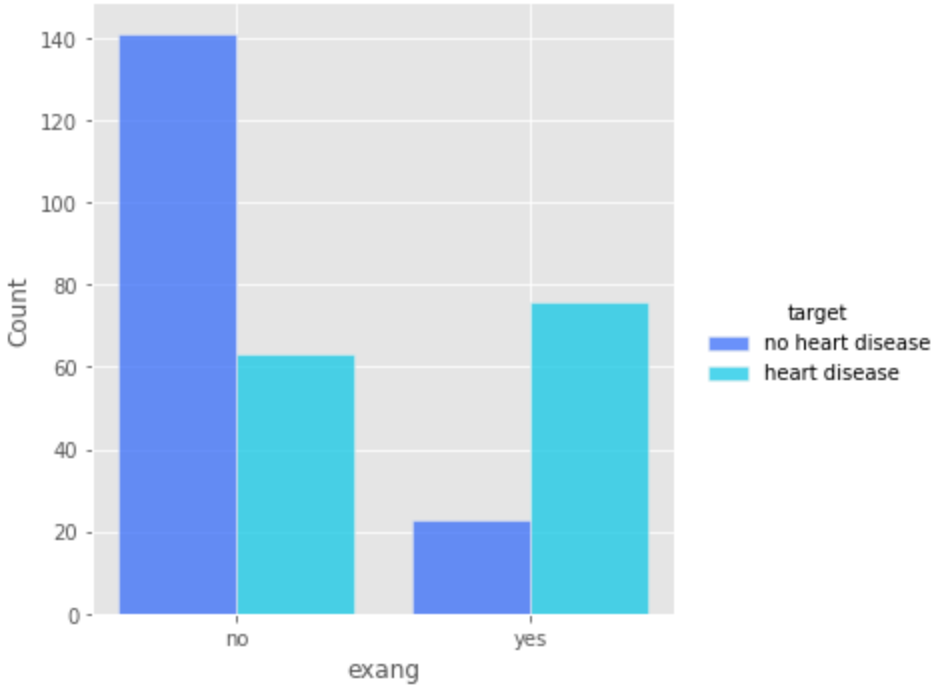
ασθένειας. Πράγματι, η αδυναμία του ατόμου να αυξήσει τους καρδιακούς παλμούς του συνήθως σημαίνει πως δεν είναι υγιές.

Exang Variable

Η μεταβλητή exang αναφέρεται στην ύπαρξη πόνου στηθάγχης λόγω της άσκησης. Παίρνει την τιμή 1 εάν ο πόνος είναι υπαρκτός, και 0 εάν το άτομο δεν πονάει. Είναι binary κατηγορική μεταβλητή.

Ελέγχουμε μήπως υπάρχουν missing values. Η στήλη fbs έχει 303 γραμμές, με μοναδικές τιμές τις 1, 0. Άρα, δεν έχουμε missing values.

Ας εξετάσουμε τώρα πως κατανέμεται σε σχέση με την target, για να δούμε κατά πόσο συμβάλλει στην πρόβλεψη της.



Όπως είναι λογικό, βλέπουμε ότι οι περισσότεροι ασθενείς που δε νοσούν, δεν έχουν στηθάγχη. Αντιθέτως, τα περισσότερα άτομα με στηθάγχη πάσχουν από την ασθένεια.

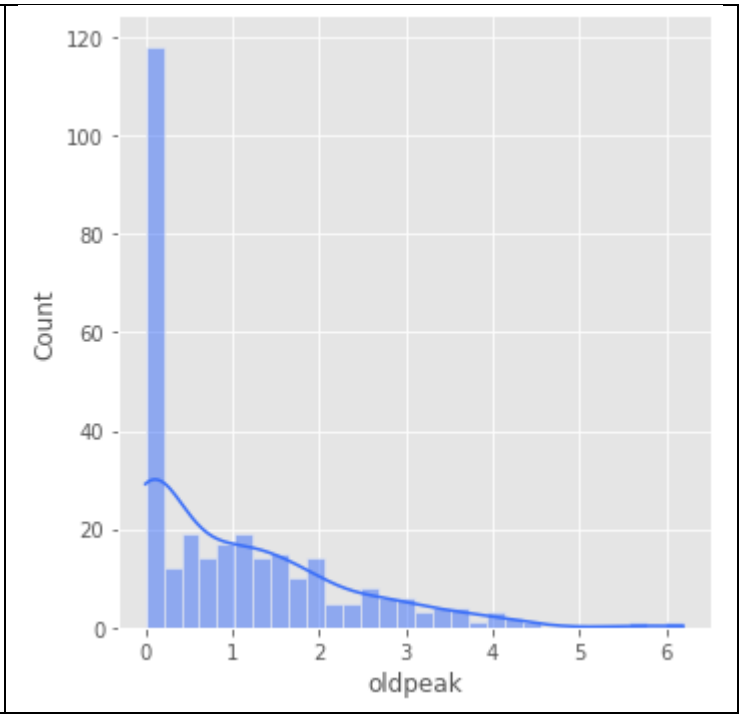
OldPeak Variable

Η μεταβλητή oldpeak αναφέρεται στην συμπίεση του ST που οφείλεται στην άσκηση, σε σχέση με το ST σε κατάσταση απάθειας. Είναι συνεχής μεταβλητή.

Ελέγχουμε εάν υπάρχουν ελλιπείς τιμές. Η στήλη oldpeak έχει 303 γραμμές, και όλες οι μοναδικές τιμές της είναι numerical. Συνεπώς, δεν έχουμε ελλιπείς τιμές.

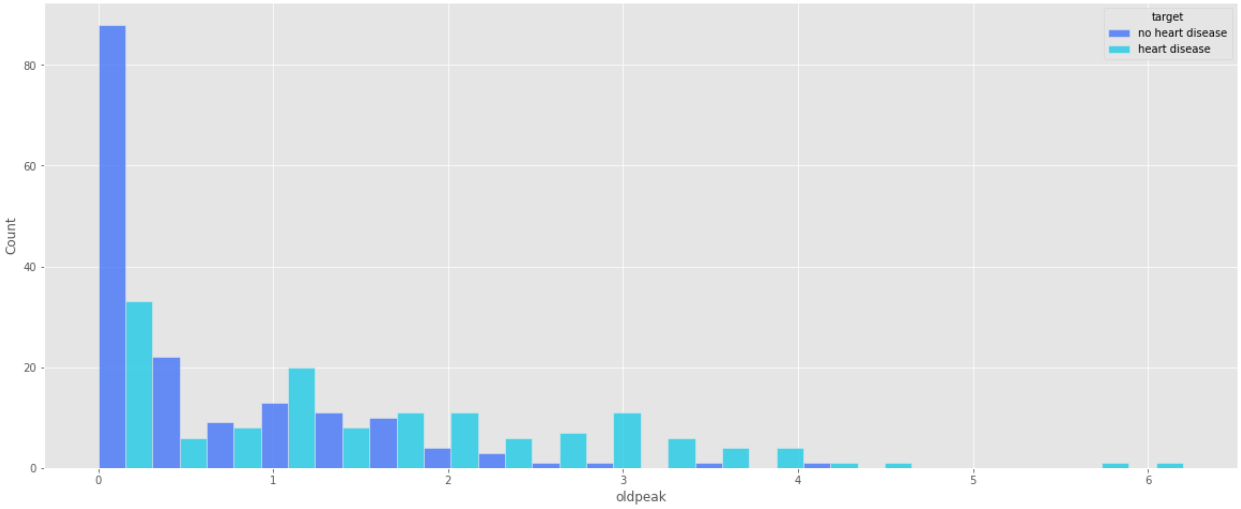
Με την εντολή describe, και φτιάχνοντας το ιστόγραμμά της, παίρνουμε τα εξής στοιχεία για την κατανομή της:

```
count    303.000000
mean       1.039604
std        1.161075
min        0.000000
25%        0.000000
50%        0.800000
75%        1.600000
max        6.200000
Name: oldpeak, dtype: float64
```



Παρατηρούμε ότι η κατανομή έχει βαριά δεξιά ουρά, έχουμε δηλαδή ορισμένες αρκετά μεγαλύτερες τιμές της oldpeak που δημιουργούν μία δεξιά ουρά στην κατανομή της. Επίσης, φαίνεται πως τα περισσότερα άτομα έχουν oldpeak 0.

Ας εξετάσουμε τώρα και την σχέση που έχουν οι τιμές της oldpeak με αυτές που παίρνει η target:



Όπως φαίνεται στο γράφημα αυτό, τα περισσότερα άτομα που έχουν oldpeak κοντά στο 0, δηλαδή τα άτομα για τα οποία το ST δεν συμπιέστηκε κατά τη διάρκεια της άσκησης, είναι υγιή.

Επιπλέον, φαίνεται πως όσο η τιμή του oldpeak αυξάνεται, δηλαδή όσο η συμπίεση μεγαλώνει, τόσο πιο πιθανό είναι το άτομο να νοσεί.

Slope Variable

Η μεταβλητή αυτή αναφέρεται στην κλίση που έχει το ST όταν το άτομο βρίσκεται στην αιχμή της άσκησης, δηλαδή την ώρα που καρδιακοί του παλμοί είναι ψηλότεροι. Οι τιμές που παίρνει είναι :

1 : ανηφορική

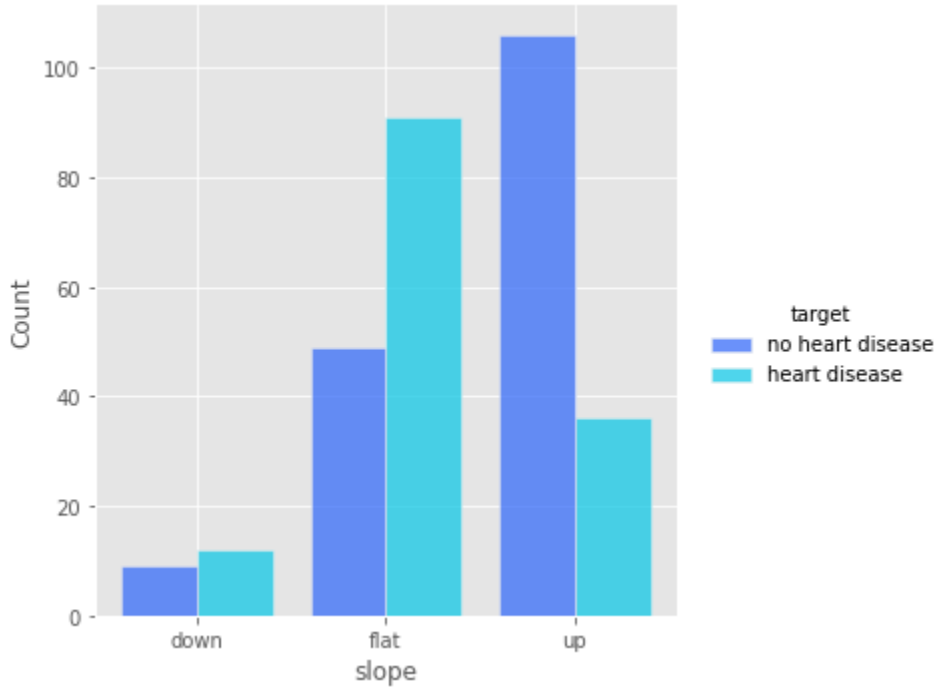
2 : ίσια

3 : κατηφορική

Είναι κατηγορική μεταβλητή.

Ελέγχουμε μήπως υπάρχουν missing values. Η στήλη slope έχει 303 γραμμές, με μοναδικές τιμές τις 3, 2, 1. Άρα, δεν έχουμε missing values.

Ας εξετάσουμε τώρα πως κατανέμεται σε σχέση με την target, για να δούμε κατά πόσο συμβάλλει στην πρόβλεψη της.



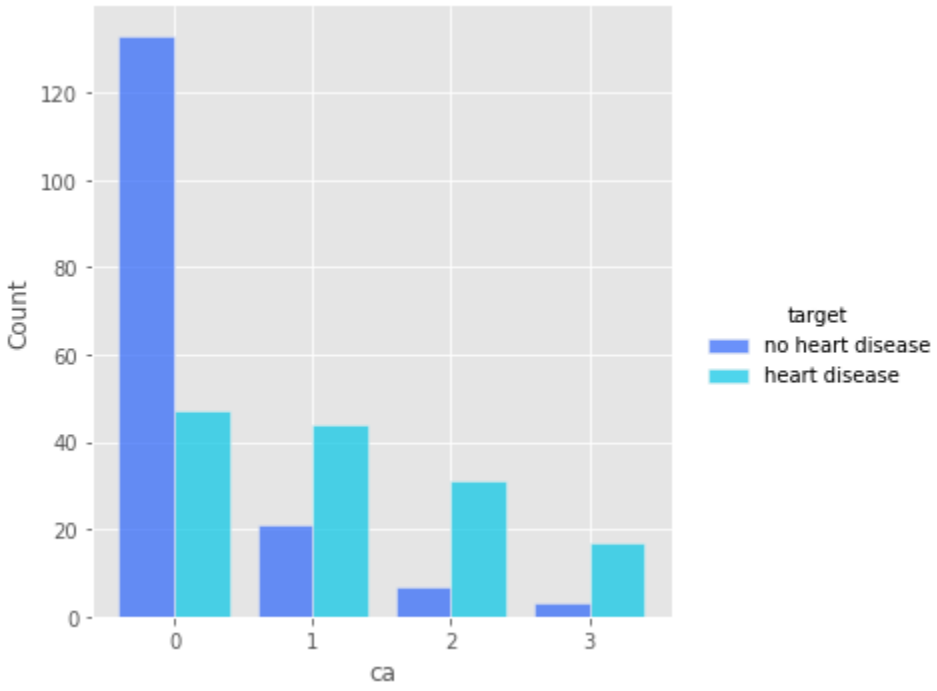
Φαίνεται πως τα περισσότερα άτομα με ανηφορική(up) κλίση του ST είναι υγιή, ενώ τα περισσότερα άτομα με ίσια(flat) κλίση είναι καρδιοπαθείς. Η κατηφορική(down) κλίση δε μας δίνει κάποια ιδιαίτερη πληροφορία για την target.

Ca Variable

Η μεταβλητή ca αναφέρεται στον αριθμό των σημαντικών αγγείων (από 0 έως 3) που η ακτινογράφηση έδειξε χρωματισμένα. Οι τιμές που παίρνει είναι από το 0 έως το 3, ανάλογα το πλήθος των αγγείων. Η μεταβλητή αυτή θεωρείται ποιοτική/κατηγορική, αφού παίρνει μόνο 4 τιμές. Επειδή όμως οι τιμές αυτές μπορούν να ιεραρχηθούν, με το 0 να είναι η μικρότερη και το 3 η μεγαλύτερη, θα θεωρήσουμε την μεταβλητή αυτή διατάξιμη, και δε θα εφαρμόσουμε σε αυτή τις τακτικές που θα εφαρμόσουμε στη συνέχεια στις υπόλοιπες κατηγορικές.

Ελέγχουμε εάν υπάρχουν missing values. Η στήλη ca έχει 303 γραμμές, όμως βλέπουμε πως πέρα από τις τιμές 0, 1, 2 και 3, υπάρχει και μία τιμή '?'. Αυτή η τιμή θεωρείται missing value, και συνεπώς πρέπει να την αντικαταστήσουμε. Για να μπορέσουμε να το κάνουμε αυτό, αρχικά πρέπει να την μετατρέψουμε σε τιμή τύπου **None**, για να μπορέσουμε να εφαρμόσουμε στις τιμές αυτές τον **SimpleImputer**. Επιλέγοντας την μέθοδο most_frequent, οι τιμές **None** αντικαθιστούνται με την επικρατούσα τιμή της στήλης ca. Στη συνέχεια, μετατρέπω όλη τη στήλη από τύπου object σε τύπου float, για να την συμπεριλάβει το μοντέλο μου.

Αφού πλέον δεν έχουμε ελλιπείς τιμές, μπορούμε να εξετάσουμε πως κατανέμεται η ca σε σχέση με την target.



Βλέπουμε ότι τα περισσότερα άτομα που δεν έχουν κανένα χρωματισμένο αγγείο είναι υγιή. Αντιθέτως, τα άτομα που έχουν έστω 1 χρωματισμένο αγγείο είναι κατά κύριο λόγο καρδιοπαθείς.

Thal Variable

Η μεταβλητή thal αναφέρεται στη ροή του αίματος του ασθενή, όταν του γίνει ένεση thallium. Το thallium είναι ένα ραδιενεργό στοιχείο, και λειτουργεί ως ιχνηλάτης. Γίνονται μετρήσεις όταν το άτομο ασκείται, και όταν είναι απαθές. Οι τιμές που παίρνει είναι :

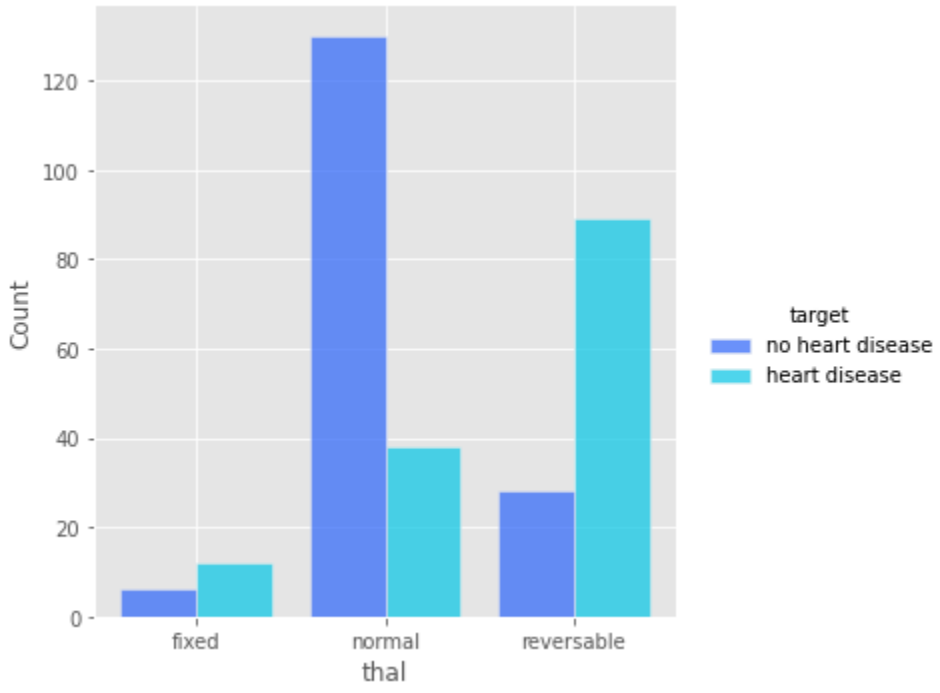
3 : κανονική ροή αίματος

6 : εντοπίζεται καθορισμένο ελάττωμα (δεν ενεργοποιεί το thallium)

7 : εντοπίζεται αναστρέψιμο ελάττωμα (ενεργοποιεί το thallium μόνο σε κατάσταση απάθειας)

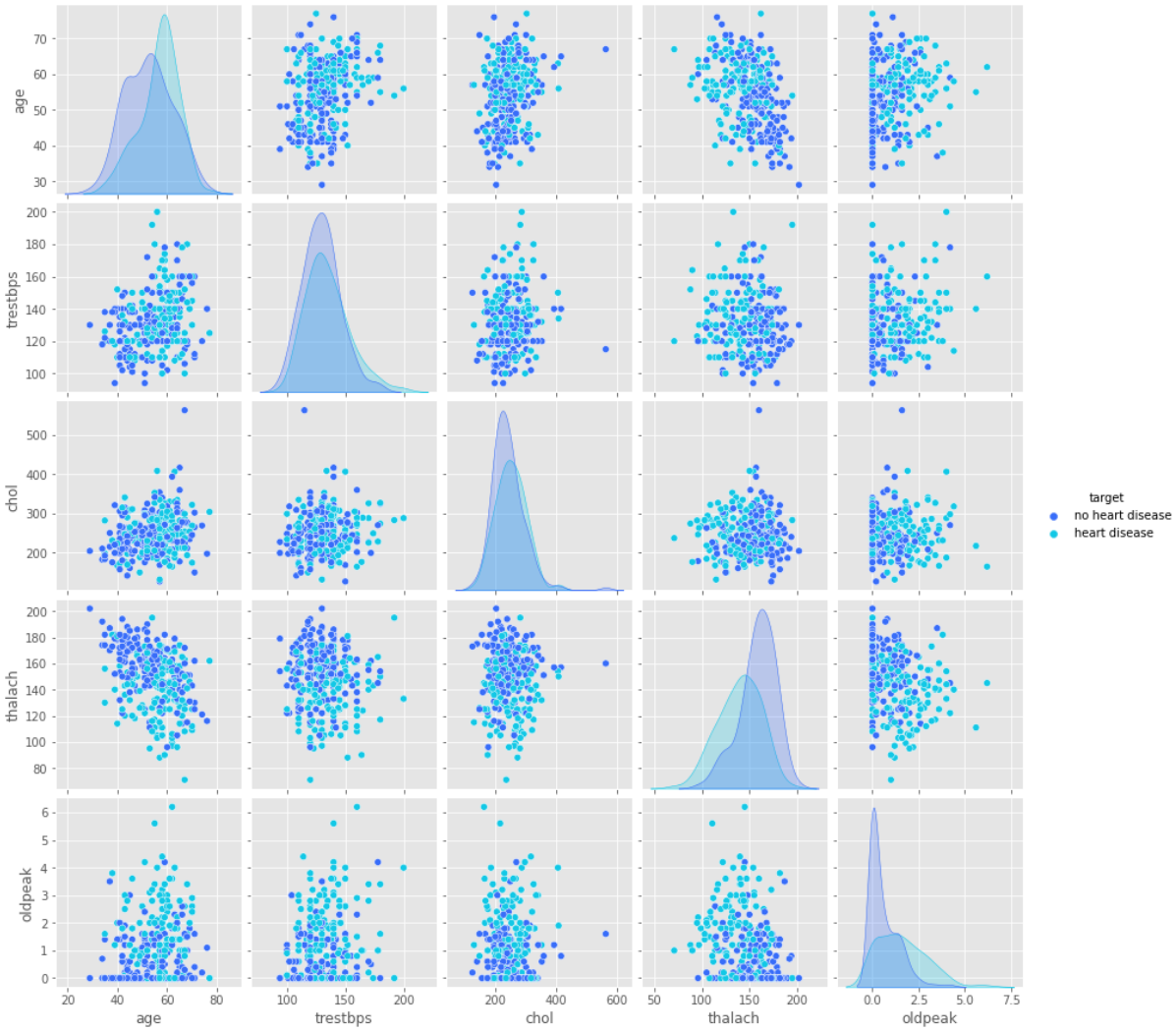
Ελέγχουμε εάν υπάρχουν missing values. Η στήλη thal έχει 303 γραμμές, όμως βλέπουμε πως πέρα από τις τιμές 3, 6 και 7, υπάρχει και μία τιμή '?'. Αυτή η τιμή θεωρείται missing value, οπότε θα την αντικαταστήσουμε. Αρχικά πρέπει να την μετατρέψουμε σε τιμή τύπου **None**, για να μπορέσουμε να εφαρμόσουμε στις τιμές αυτές τον **SimpleImputer**. Επιλέγοντας την μέθοδο most_frequent, οι τιμές **None** αντικαθιστώνται με την επικρατούσα τιμή της στήλης thal. Στη συνέχεια, μετατρέπω όλη τη στήλη σε τύπου float, για να την συμπεριλάβει το μοντέλο μου.

Αφού πλέον δεν έχουμε ελλιπείς τιμές, μπορούμε να εξετάσουμε πως κατανέμεται η ca σε σχέση με την target.

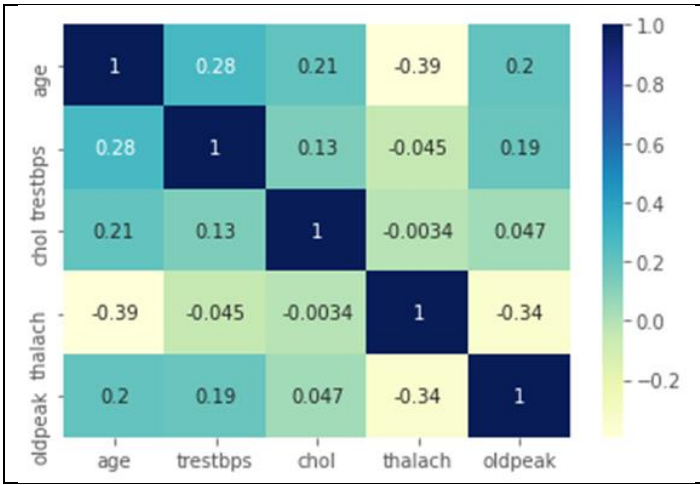


Βλέπουμε ότι τα περισσότερα άτομα που έχουν κανονική(normal) ροή αίματος είναι υγιή, όπως είναι λογικό. Αντιθέτως, τα άτομα που έχουν αναστρέψιμο(reversable) ελάττωμα είναι κατά κύριο λόγο καρδιοπαθείς. Το καθορισμένο(fixed) ελάττωμα δε φαίνεται να επηρεάζει ιδιαίτερα την target.

Σχέση μεταξύ όλων των συνεχών μεταβλητών με scatterplots:



Μπορούμε να διαπιστώσουμε καλύτερα την σχέση μεταξύ τους αναπαριστώντας τα correlations που έχουν, μέσω ενός διαγράμματος heatmap:



Βλέπουμε πως οι μεταβλητές που έχουν την μεγαλύτερη συσχέτιση μεταξύ τους είναι τα ζευγάρια thalach-age και thalach-oldpeak.

Part 2: Feature Extraction / Selection

Πραγματοποιώντας τις διαδικασίες του Πρώτου Μέρους (εξάλειψη ελλιπών τιμών, επαναπροσδιορισμού της target), και παίρνοντας μια αρχική εικόνα των δεδομένων μας, μπορούμε να προχωρήσουμε στο δεύτερο κομμάτι της επιλογής των χαρακτηριστικών που θα χρησιμοποιήσουμε στο μοντέλο μας. Τα χαρακτηριστικά αυτά θα είναι αυτά που θα έχουν την βέλτιστη προβλεπτική ικανότητα της μεταβλητής target. Για να μπορέσουμε όμως να χρησιμοποιήσουμε τις τεχνικές επιλογές χαρακτηριστικών αποτελεσματικά, θα πρέπει να φέρουμε το μοντέλο μας σε μια μορφή στην οποία μπορούν να αξιολογηθούν όλα τα χαρακτηριστικά ισάξια.

Get Dummies

Αρχικά, πρέπει να βρούμε έναν τρόπο να συμπεριλάβουμε τις κατηγορικές μεταβλητές στο μοντέλο μας. Οι κατηγορικές μεταβλητές Για αυτόν τον λόγο, θα χρησιμοποιήσουμε την τεχνική **get_dummies**. Το **get_dummies** παίρνει σαν είσοδο μία στήλη που περιέχει κατηγορική μεταβλητή και επιστρέφει τόσες στήλες όσα και τα επίπεδα αυτής. Η κάθε καινούρια στήλη θα παίρνει 1, εάν το επίπεδο της γραμμής αντιστοιχίζεται με το επίπεδο της στήλης, και 0 εάν είναι διαφορετικό.

Οι binary κατηγορικές μεταβλητές είναι ήδη σε αυτή τη μορφή. Οι υπόλοιπες κατηγορικές μεταβλητές είναι οι cp, restecg, slope, ca, thal. Για να προκύψουν και οι καινούριες ονομασίες στις κολώνες του dataset, θα αντικαταστήσω τα επίπεδα τους με τις ονομασίες τους. Ως συνέπεια, οι κατηγορικές μεταβλητές είναι πλέον τύπου object. Αυτό βέβαια δεν θα επηρεάσει την **get_dummies**, καθώς επικεντρώνεται στην διαφοροποίηση ξεχωριστών στοιχείων, και όχι στην αριθμητική τιμή που έχουν. Επειδή οι τιμές της μεταβλητής ca έχουν μια σειρά, θα την θεωρήσουμε ως μεταβλητή ordinal, και συνεπώς δεν θα χρειαστεί να την μετατρέψω. Αφού δημιουργήσουμε τις στήλες dummies για κάθε τιμή των κατηγορικών μεταβλητών (για την ακρίβεια, για κάθε τιμή -1, λόγω της εντολής drop_first=True, την οποία χρησιμοποιούμε για να αποφύγουμε τη γραμμική εξάρτηση μεταξύ των στηλών), τις προσθέτουμε στο αρχικό dataset μας. Το τελικό μας dataset μας θα έχει όλες τις αρχικές στήλες, πλην των αρχικών κατηγορικών, τις οποίες αφαιρέσαμε και τις αντικαταστήσαμε με αυτές των dummies.

Train–Test–Split

Αμέσως επόμενο βήμα πριν προχωρήσω σε feature extraction/selection, είναι να διαχωρίσω τις ανεξάρτητες X από την εξαρτημένη Y μεταβλητή (target). Ο λόγος που κάνω αυτό το βήμα τώρα,

είναι επειδή για να κάνω σωστή επιλογή των χαρακτηριστικών της X, που προβλέπουν καλύτερα την Y, θα πρέπει να διαχειριστώ την X και την Y σαν διαφορετικές οντότητες. Θα χρησιμοποιήσω τη μέθοδο **train_test_split**. Με τη μέθοδο αυτή, χωρίζω επίσης τα δεδομένα σε δεδομένα train, που θα τα χρησιμοποιήσω για να κατασκευάσω το μοντέλο μου, και δεδομένα test, που θα τα χρησιμοποιήσω για να κάνω evaluate στο μοντέλο που προέκυψε.

Standardization

Τώρα που χώρισα τα δεδομένα μου, χρειάζεται να αντιμετωπίσω την διαφορά που έχουν το range των τιμών των στηλών πέρα από τις κατηγορικές, καθώς αυτή η διαφορά θα δώσει στο μοντέλο μας ανακριβή αποτελέσματα. Για να το κάνουμε αυτό, θα εφαρμόσουμε τον **StandardScaler**. Ο λόγος που κάνουμε αυτή την διαδικασία αφού έχουμε χωρίσει τα δεδομένα μας σε train και test, είναι επειδή θέλουμε να αποφύγουμε την διέρρευση πληροφορίας από το train set, που χρησιμοποιείται για την εκπαίδευση του μοντέλου, στο test set, που χρησιμοποιείται για την αξιολόγησή του. Έτσι, θα κάνουμε fit μόνο στα train data, και θα εφαρμόσουμε τον imputer που προέκυψε σε όλο το X. Με αυτόν τον τρόπο, το train set δεν θα πάρει καμία πληροφορία για την κατανομή του test set, και συνεπώς τα τεελταία θα θεωρούνται δεδομένα που το μοντέλο μας δεν έχει ξαναδεί. Οι κολώνες που χρειάζονται κανονικοποίηση είναι όλες οι συνεχείς και η ordinal, προκειμένου να έρθουν στην ίδια κλίμακα. Αυτές είναι οι age, trestbps, chol, thalach, oldpeak και ca.

Feature Selection

Τέλος, προχωρώ στη διαδικασία επιλογής γνωρισμάτων που είναι πιο σημαντικά για την πρόβλεψη της y. Θα κάνω την επιλογή αυτή χρησιμοποιώντας τα **SelectFromModel**, **SelectKBest**. Στο επόμενο κομμάτι (Part 3), θα υλοποιήσω τους αλγορίθμους Logistic Regression, Random Forest και Naïve Bayes, μέσω της βιβλιοθήκης sklearn. Συνεπώς, θα πρέπει να επιλέξω τα σωστά γνωρίσματα για κάθε αλγόριθμο ξεχωριστά, όπου αυτό είναι δυνατόν. Επίσης, ζητείται να δω την βελτίωση που γίνεται πριν και μετά την επιλογή των γνωρισμάτων. Θα υπολογίσω λοιπόν το accuracy στο αρχικό dataset, και έπειτα σε αυτό που έχει τις στήλες που επέλεξε το **SelectFromModel** και **SelectKBest**. Το accuracy ορίζεται ως το ποσοστό των τιμών της y που το μοντέλο μας πρόβλεψε σωστά (= σωστές προβλέψεις / όλες οι προβλέψεις).

Για κάθε αλγόριθμο, θα ακολουθήσουμε τα εξής βήματα :

1. Υπολογισμός του αρχικού accuracy, όταν το μοντέλο μας εφαρμόζεται σε όλα μας τα δεδομένα
2. Επιλογή των χρήσιμων γνωρισμάτων μέσω των αλγορίθμων

3. Υπολογισμός του καινούριου accuracy, όταν το μοντέλο μας εφαρμόζεται στις επιλεγμένες στήλες

Logistic Regression

Αρχικά, φτιάχνουμε ένα απλό μοντέλο LogisticRegression σε όλο μας το dataset. Το accuracy αυτού του μοντέλου είναι 0.85. Εκτελούμε το **SelectFromModel**, χρησιμοποιώντας το παραπάνω μοντέλο ως estimator, και ως αποτέλεσμα μας δίνονται οι εννιά πιο σημαντικές στήλες για την πρόβλεψη του y, για το μοντέλο LogisticRegression (sex, exang, ca, cp_atypical, cp_non anginal, cp_typical, slope_flat, thal_normal, thal_reversable). Όταν το εφαρμόσουμε, βλέπουμε πως το accuracy έχει μειωθεί : είναι 0.83.

Random Forest

Όπως και πριν, φτιάχνουμε ένα απλό μοντέλο Random Forest σε όλο μας το dataset. Το accuracy αυτού του μοντέλου είναι 0.79. Εκτελούμε το **SelectFromModel**, χρησιμοποιώντας το παραπάνω μοντέλο ως estimator, και ως αποτέλεσμα μας δίνονται οι εννιά πιο σημαντικές στήλες για την πρόβλεψη του y, για το μοντέλο Random Forest (age, trestbps, chol, thalach, exang, oldpeak, ca, thal_normal, thal_reversable). Όταν το εφαρμόσουμε, βλέπουμε πως το accuracy έχει πάλι μειωθεί : είναι 0.75.

Πρόβλημα : Στα παραπάνω μοντέλα, μπορέσαμε να χρησιμοποιήσουμε την **SelectFromModel** διότι διαθέτουν τα attributes coef_ ή feature_importances_. Με βάση αυτά είναι που ορίζει τα πιο σημαντικά γνωρίσματα για το κάθε μοντέλο. Το επόμενο μοντέλο όμως που θέλουμε να υλοποιήσουμε (Naïve Bayes), δεν διαθέτει αυτά τα χαρακτηριστικά. Επομένως, θα χρησιμοποιήσουμε το **SelectKBest.** Το **SelectKBest** δεν λαμβάνει υπ' οψιν το μοντέλο που θα εφαρμόσουμε στα δεδομένα μας. Επομένως, θα χρησιμοποιήσουμε τις στήλες που θα επιλέξει αυτό στο μοντέλο Naïve Bayes. Για k=9 (επιλέγουμε το 9, μιας και έχουμε 9 μεταβλητές και στους παραπάνω αλγορίθμους), προκύπτει πως οι πιο σημαντικές στήλες είναι οι thalach, exang, oldpeak, ca, cp_non anginal, slope_flat, thal_normal και thal_reversable.

Naïve Bayes

Ξανά, φτιάχνουμε ένα απλό μοντέλο Naïve Bayes σε όλο μας το dataset. Το accuracy αυτού του μοντέλου είναι 0.84. Χρησιμοποιώντας μόνο τις στήλες που προέκυψαν από το **SelectKBest** σε ένα νέο μοντέλο, το καινούριο accuracy είναι 0.8.

Παρατηρούμε πως σε κάθε περίπτωση, το accuracy έχει μειωθεί μετά την εφαρμογή του μοντέλου μόνο στις επιλεγμένες στήλες. Αυτό είναι λογικό, καθώς στο καινούριο μοντέλο χρησιμοποιούμε λιγότερες στήλες, δηλαδή λιγότερα δεδομένα εκπαίδευσης, οπότε και η

απόδοση του είναι μικρότερη. Η μείωση όμως αυτή της απόδοσης, συνεπάγεται την μείωση του χρόνου εκπαίδευσης του αλγορίθμου. Πράγματι, ο αλγόριθμος μας θα χρειαστεί λιγότερο χρόνο για να εκπαιδεύσει λιγότερα δεδομένα. Συνεπώς, μια μικρή μείωση της απόδοσης δεν είναι απαραίτητα κάτι αρνητικό για το μοντέλο μας, καθώς μπορεί να σημαίνει εξοικονόμηση του χρόνου που χρειάζεται για να τον τρέξουμε.

Part 3: Classification / Prediction

Αφού επιλέξαμε τις στήλες που θα χρησιμοποιήσουμε σε κάθε μοντέλο, το επόμενο βήμα είναι να προσαρμόσουμε τις παραμέτρους του κάθε μοντέλου, προκειμένου να μεγιστοποιήσουμε την ικανότητα του να προβλέψει την target. Για να αξιολογήσουμε την αποτελεσματικότητα των παραμέτρων αυτών στο μοντέλο μας, θα χρησιμοποιήσουμε τα κριτήρια Precision, Recall, F-score και Accuracy.

- Το Precision αφορά την ακρίβεια των θετικών προβλέψεων του μοντέλου, μας δίνει δηλαδή το ποσοστό των σωστών θετικών προβλέψεων από όλες τις προβλέψεις που βγήκαν θετικές.
- Το Recall αφορά την ικανότητα του μοντέλου μας να εντοπίζει τις θετικές τιμές. Υπολογίζει το ποσοστό των σωστών θετικών προβλέψεων που βρίσκει το μοντέλο από όλες τις προβλέψεις που είναι στην πραγματικότητα θετικές.
- Το F1 Score είναι ο αρμονικός μέσος των Precision και Recall.
- Το Accuracy είναι το ποσοστό όλων των σωστών προβλέψεων από όλες τις προβλέψεις.

Λόγω του ότι τα δεδομένα μας αφορούν ιατρική διάγνωση, το κριτήριο που θα λάβουμε κυρίως υπ' όψιν μας θα είναι το Recall. Αυτό είναι επειδή μας ενδιαφέρει περισσότερο το πόσες σωστές θετικές προβλέψεις θα εντοπίσει το μοντέλο μας, καθώς θέλουμε να μειώσουμε όσο είναι δυνατόν αυτές που δεν θα εντοπίσει(False Negative). Πράγματι, σε μία ιατρική εξέταση, προτιμούμε να διαγνώσουμε λάθος ένα υγιές άτομο ως ασθενή (False Positive), παρά να το διαγνώσουμε λάθος ως υγιές ενώ νοσεί (False Negative). Ακριβώς αυτό μας δίνει σε ποσοστό το Recall.

Τα True Negative και True Positive, σημαίνουν πως κάναμε σωστή αρνητική διάγνωση (προβλέψαμε σωστά πως το άτομο είναι υγιές) και σωστή θετική διάγνωση (προβλέψαμε σωστά πως το άτομο είναι ασθενής) αντίστοιχα.

Επίσης, θα χρησιμοποιήσουμε και το confusion matrix. Με βάση των παρακάτω πίνακα, το confusion matrix μας δείχνει όλες τις σωστές αλλά και λάθος προβλέψεις που έκανε το μοντέλο μας.

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

Τέλος, θα χρησιμοποιήσουμε Cross Validation για να εξετάσουμε τη σταθερότητα του μοντέλου μας. Το Cross Validation χωρίζει τα train data μας σε καινούρια σετ X,y, και σε κάθε σετ X αξιολογεί την ικανότητα που έχει να προβλέψει το y. Η διαδικασία αυτή επαναλαμβάνεται όσες φορές θέσω τον αλγόριθμο (μέσω του cv=....), για διαφορετικό και τυχαίο set X,y κάθε φορά.

Παίρνουμε τα αποτελέσματα του Cross Validation υπολογίζοντας το cross_val_score του κάθε αλγορίθμου. Το cross_score_val, μας δείχνει πως διακυμαίνεται η μεταβλητότητα της απόδοσης του κάθε μοντέλου, όταν αυτό εκτελείται σε διαφορετικές διαμοιράσεις του dataset. Δηλαδή, εάν παρατηρήσουμε μεγάλη διαφορά της απόδοσης στις διαφορετικές διαμερίσεις του dataset, αυτό σημαίνει πως για κάθε X υποσύνολο, το y προβλέπεται με διαφορετική σιγουριά, το οποίο συνεπάγεται στο ότι το μοντέλο μας δεν κάνει σωστή γενίκευση όταν βλέπει καινούρια δεδομένα.

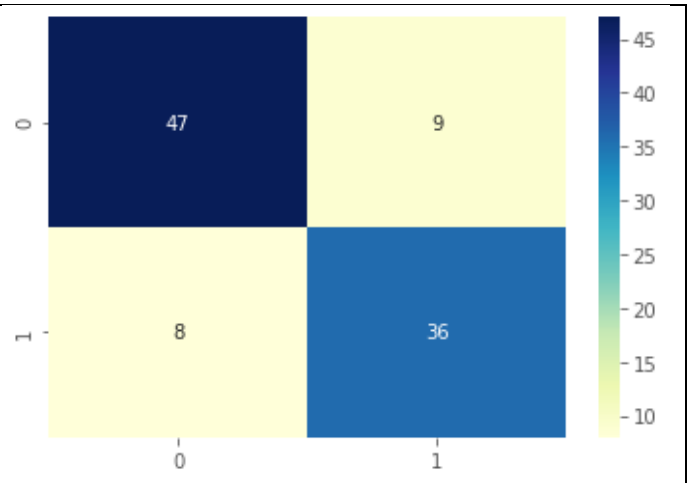
Logistic Regression

Ξεκινάμε με το μοντέλο Logistic Regression. Εφαρμόζοντας το στις στήλες που επιλέξαμε στο δεύτερο μέρος, χωρίς να αλλάξουμε τις default παραμέτρους του, τα μέτρα αξιολόγησης του έχουν ως εξής :

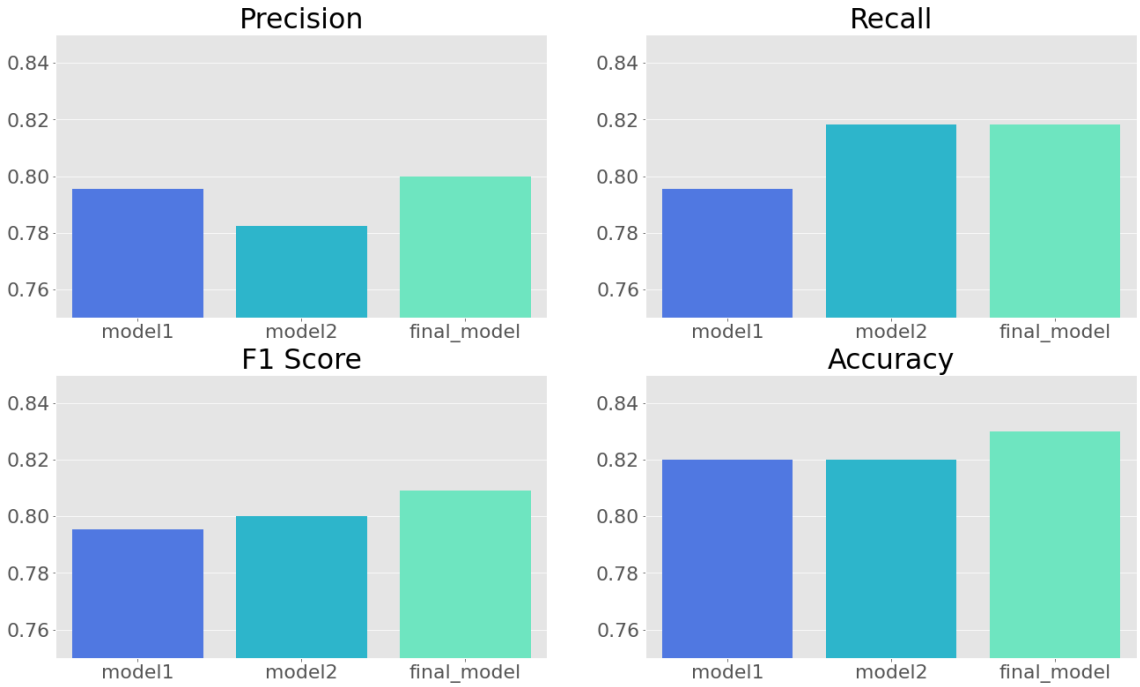
```
Precision : 0.8
Recall : 0.8181818181818182
F1 Score : 0.8089887640449439
Accuracy : 0.83
```

Confusion Matrix →

Έχουμε 47 true negatives, 36 true positives, 9 false positives και 8 false negatives.



Θα δοκιμάσουμε τώρα να κάνουμε αλλαγές στις παραμέτρους penalty και solver, για να δούμε μήπως καταφέρνουμε να καταλήξουμε σε κάποιο καλύτερο μοντέλο. Παρ' όλα αυτά, διαπιστώνουμε πως κανένας συνδυασμός δεν μπορεί να βγάλει καλύτερα αποτελέσματα από τον πρώτο αλγόριθμο με τις default παραμέτρους. Πράγματι, αν συγκρίνουμε την αποτελεσματικότητα του πρώτου μοντέλου, με ένα model1 (παράμετροι : penalty='none', solver='lbfgs') και ένα model2 (παράμετροι : penalty='l1', solver='liblinear'), βλέπουμε πως το final_model (παράμετροι : default) υπερτερεί.



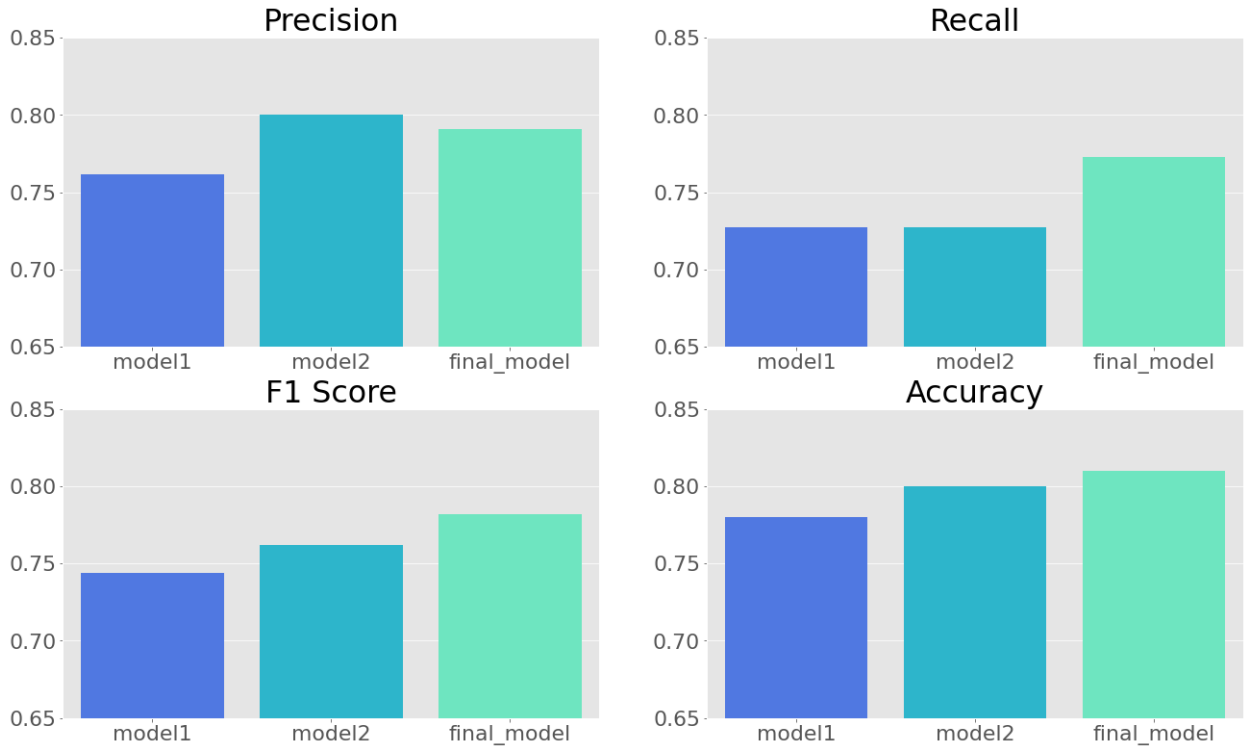
Συνεπώς, επιλέγουμε το μοντέλο με τις αρχικές παραμέτρους.

Ας ελέγξουμε τώρα και την σταθερότητα του μοντέλου μας. Υπολογίζοντας το cross_val_score, βλέπουμε πως το μέσο accuracy σε 5 folds, δηλαδή σε 5 διαφορετικές διαμοιράσεις, είναι 0.8229268292682926, και η διακύμανση αυτού 0.000976591314693634. Με μία τόσο μικρή διακύμανση, συμπεραίνουμε ότι το μοντέλο μας είναι αρκετά σταθερό, και έχει πολύ ικανοποιητική απόδοση.

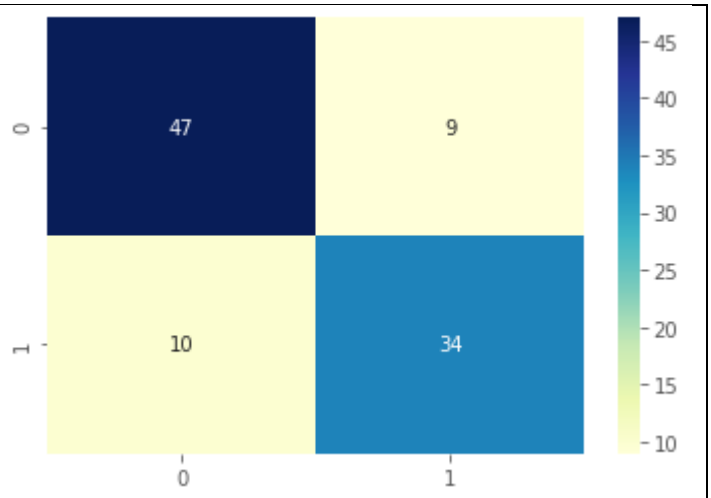
Random Forest

Το μοντέλο Random Forest παίρνει αρκετές διαφορετικές παραμέτρους. Αυτές που θα δοκιμάσουμε να ρυθμίσουμε θα είναι οι n_estimators (πόσα decision trees θα χρησιμοποιήσει

ο αλγόριθμος), max_depth (το μέγιστο βάθος, δηλαδή πόσες διακλαδώσεις θα έχει κάθε δέντρο) και min_samples_leaf (οι ελάχιστες παρατηρήσεις που ανήκουν σε κάθε φύλλο). Δοκιμάζουμε αρχικά διάφορες τιμές για αυτές τις παραμέτρους, και συνδυασμούς μεταξύ τους, έτσι ώστε να πάρουμε μια ιδέα για το ποιες τιμές τους δίνουν καλύτερα αποτελέσματα. Πειραματιζόμαστε δοκιμάζοντας τις τιμές 5, 10, 15, 20, 25, 30 για το n_estimators, τις τιμές 2, 5, 8, 10 για το max_depth και τις τιμές 1, 2, 3, 4, 5 για το min_samples_leaf. Κάποια μοντέλα που δοκιμάσαμε έχουν ως εξής:



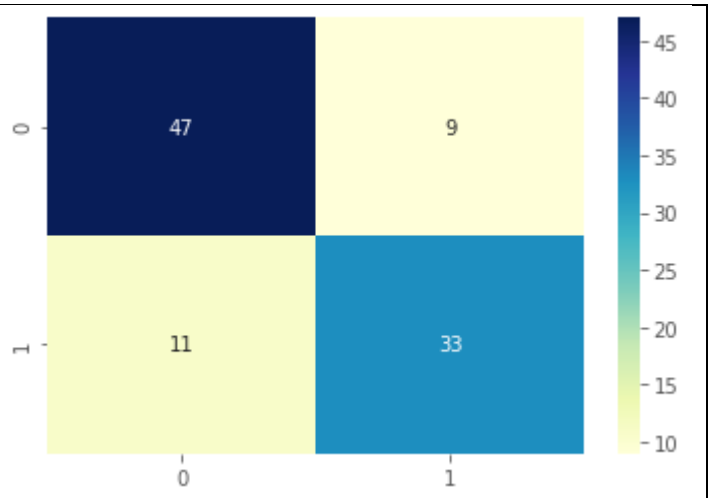
Εν τέλει, ο συνδυασμός n_estimators= 20, max_depth = 5, min_samples_leaf=2 (δηλαδή το final_model στο παραπάνω γράφημα) μας δίνει το μοντέλο με τα καλύτερα αποτελέσματα. Παρ' όλο που το model2 έχει το υψηλότερο precision, δεν μπορούμε να το επιλέξουμε καθώς δεν έχει και το υψηλότερο recall, που είναι το πιο σημαντικό σε αυτή την περίπτωση . Έτσι, το τελικό μοντέλο έχει ως εξής:

Precision : 0.7906976744186046 Recall : 0.7727272727272727 F1 Score : 0.7816091954022988 Accuracy : 0.81 Confusion Matrix → Έχουμε 47 true negatives, 34 true positives, 9 false positives και 10 false negatives.	

Τέλος, ελέγχουμε τη σταθερότητα του μοντέλου που επιλέξαμε. Υπολογίζοντας το cross_val_score, βλέπουμε πως το μέσο accuracy σε 5 folds είναι 0.7880487804878048, και η διακύμανση αυτού 0.0023825996430696006. Εφόσον έχουμε αρκετά μικρή διακύμανση, μπορούμε να πούμε ότι το μοντέλο μας είναι αρκετά σταθερό, με ικανοποιητική απόδοση.

Naïve Bayes

Ο αλγόριθμος δεν έχει πολλές διαφορετικές παραμέτρους που μπορούμε να επεξεργαστούμε. Συνεπώς, χρησιμοποιώντας τις default παραμέτρους, το evaluation του αλγορίθμου έχει ως εξής:

Precision : 0.7857142857142857 Recall : 0.75 F1 Score : 0.7674418604651163 Accuracy : 0.8 Confusion Matrix → Έχουμε 47 true negatives, 33 true positives, 9 false positives και 11 false negatives.	

Υπολογίζοντας το cross_val_score του αλγορίθμου, βλέπουμε πως το μέσο accuracy σε 5 folds είναι 0.8525609756097561, και η διακύμανση αυτού 0.0016142772159428917. Συμπεραίνουμε

δηλαδή ότι το μοντέλο μας είναι σχετικά σταθερό, αφού έχει μικρή διακύμανση, και δίνει ένα πολύ καλό accuracy.

Επιλογή τελικού μοντέλου

Όπως αναφέραμε και στην αρχή, το κριτήριο με βάση το οποίο θα επιλέξουμε το τελικό μοντέλο θα είναι το recall. Με recall = 0.8181818181818182, είναι προφανές ότι το μοντέλο που δίνει τα καλύτερα αποτελέσματα θα είναι το Logistic Regression.

Part 4: Clustering

Το clustering θεωρείται κομμάτι του unsupervised learning, δηλαδή έχουμε τα δεδομένα X για τα οποία δεν μας δίνεται καμία πληροφορία για τις ετικέτες τους(target), και καλούμαστε να τα χωρίσουμε σε clusters τα οποία περιγράφουν τη δομή τους προκειμένου να εξάγουμε κάποιο συμπέρασμα για το πώς διαφοροποιούνται (ιδανικά, τα clusters που θα σχηματίζονται θα αντιστοιχίζονται στην μεταβλητή y). Θα χρησιμοποιήσω την μεταβλητή y(target) μόνο για να αξιολογήσω τα μοντέλα μου στο τέλος, και όχι για να τα εκπαιδεύσω.

Συνεπώς, για να προχωρήσουμε σε συσταδοποίηση, θα πρέπει να αφαιρέσουμε από το dataset την μεταβλητή target, καθώς όπως είπαμε σε ένα clustering project δεν γνωρίζουμε ποτέ τα labels των δεδομένων μας.

Για να φέρουμε το dataset στην ιδανική μορφή για να εφαρμόσουμε αλγορίθμους συσταδοποίησης, θα επαναλάβουμε τα βήματα της διαδικασίας 2, μόνο που τώρα δε θα χωρίσουμε τα δεδομένα μας σε train και test. Αφού μετατρέψω και τις ελλιπείς τιμές, κανονικοποιώ τις συνεχείς μεταβλητές. Τώρα τα δεδομένα μας είναι στην σωστή μορφή για να συσταδοποιηθούν.

KMeans

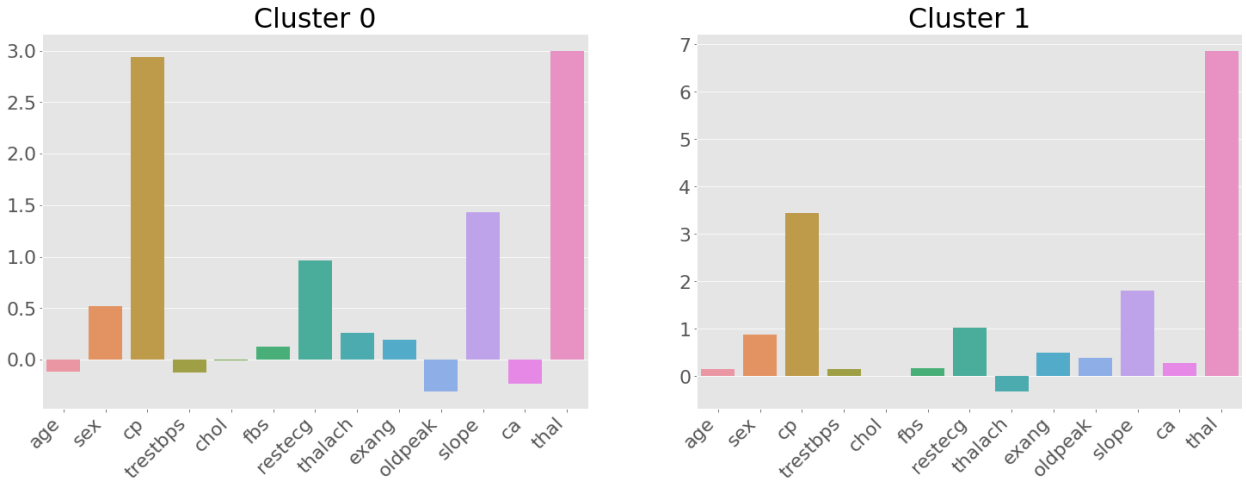
Ο αλγόριθμος KMeans επιχειρεί να χωρίσει τα δεδομένα μας σε k clusters, όπου το k είναι προκαθορισμένο από εμάς. Εκτιμούμε πως ο βέλτιστος αριθμός των συστάδων θα είναι ανάμεσα στο 2 και στο 5, όπως δηλαδή κατανέμονται και οι τιμές της target. Για να αποφασίσουμε ποιο k είναι το ιδανικό, θα πραγματοποιήσουμε ένα for loop για το k, δοκιμάζοντας τις τιμές από k=2 μέχρι k=10. Για να αξιολογήσουμε την αποτελεσματικότητα του αλγορίθμου για τα διαφορετικά k, θα υπολογίσουμε για το καθένα το silhouette score του. Το silhouette score παίρνει τιμές από το 1 έως το -1, και όσο πιο κοντά βρίσκεται στο 1, τόσο πιο καλά έχουν κατανεμηθεί οι συστάδες μας.

Το silhouette_score για κάθε k έχει ως εξής :

Βλέπουμε πως για k=2, ο αλγόριθμος KMeans μας δίνει το υψηλότερο silhouette_score. Επομένως, θα χρησιμοποιήσουμε αυτόν για να χωρίσουμε τα δεδομένα μας σε clusters.

Φτιάχνουμε ένα μοντέλο KMeans με k=2, και το κάνουμε fit στα δεδομένα μας, από τα οποία έχουμε αφαιρέσει την y. Για k=2, το silhouette_score είναι 0.3103211089162096, το οποίο βρίσκεται αρκετά κοντά στο 1. Οι ετικέτες του μοντέλου μας δείχνουν ότι τα δεδομένα μας έχουν χωριστεί σε δύο συστάδες, με ετικέτες 0 και 1.

Θα ήταν χρήσιμο να εξετάσουμε πώς διαφοροποιούνται τα γνωρίσματα του dataset, ανάλογα το cluster στο οποίο έχουν τοποθετηθεί. Για να το κάνουμε αυτό, θα δημιουργήσουμε 2 διαφορετικά dataframes, ένα που περιέχει τα στοιχεία του cluster με ετικέτα 0, και ένα που περιέχει τα στοιχεία του cluster με ετικέτα 1. Αν πάρουμε τη μέση τιμή κάθε γνωρίσματος για το κάθε dataframe, μπορούμε να δούμε πώς διαφοροποιούνται οι μεταβλητές μας, ανάλογα το cluster στο οποίο έχουν τοποθετηθεί.

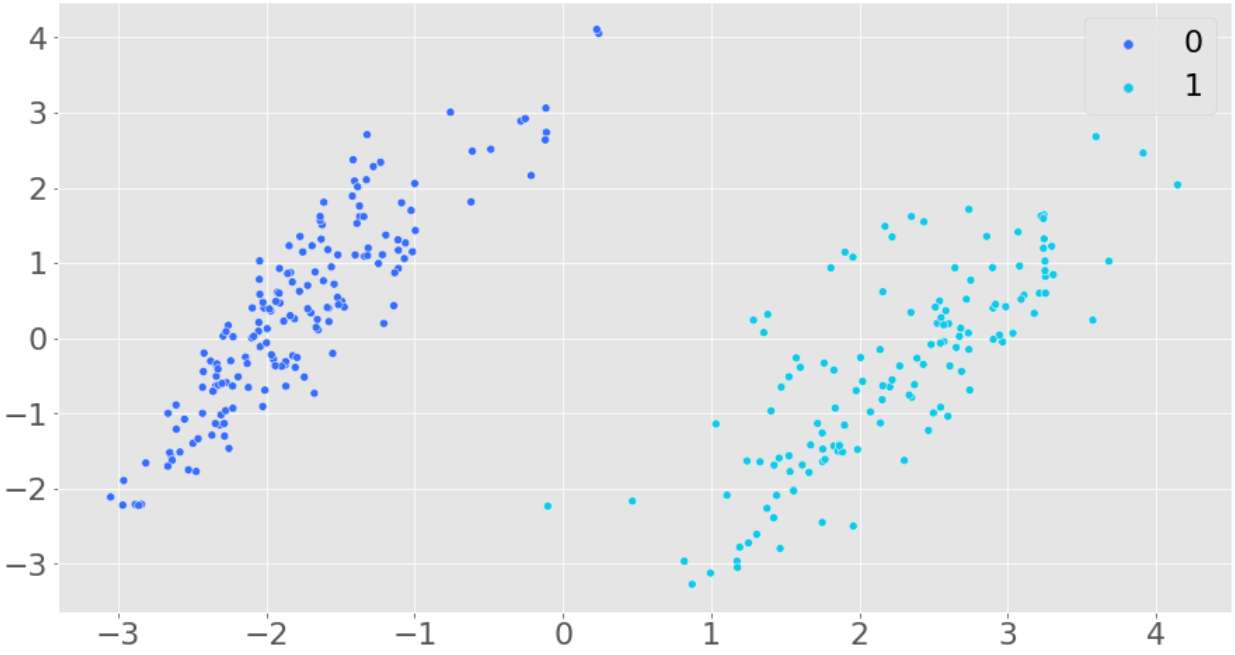


Μπορούμε να δούμε πως, οι μεταβλητές που παίζουν τον κυριότερο ρόλο στην διαμόρφωση των συστάδων είναι η age, trestbps, thalach, oldpeak και ca, επειδή αλλάζει η διεύθυνσή τους.

Ας προσπαθήσουμε τώρα να κάνουμε visualize τα clusters που έφτιαξε ο αλγόριθμός μας, για να δούμε εάν τα όρια του διαχωρισμού των cluster είναι ξεκάθαρα. Το αρχικό σχήμα του dataset μας είναι 303x13, δηλαδή έχει 13 διαστάσεις. Όπως καταλαβαίνουμε, οι 13 διαστάσεις είναι αδύνατο να οπτικοποιηθούν. Έτσι, για να μπορέσουμε να κατανοήσουμε τον τρόπο που δημιουργούνται οι συστάδες, θα πρέπει να μειώσουμε τις διαστάσεις του dataset από 18 σε 2, χρησιμοποιώντας PCA.

Δημιουργούμε ένα καινούριο dataframe, το οποίο περιέχει τα δεδομένα του heart_disease ανελυμμένες σε δύο principal components, ή αλλιώς σε δύο διαστάσεις. Οι 2 components εξηγούν τα 36% και 14% της διακύμανσης του αρχικού dataset αντίστοιχα, οπότε μπορούμε να θεωρήσουμε πως το καινούριο dataframe είναι μια σχετικά ακριβής αναπαράσταση του αρχικού. Παράλληλα, εκτελούμε την ίδια διαδικασία και για τα cluster_centers_, δηλαδή τα κέντρα των clusters που εντόπισε ο kmeans. Καθένα από αυτά, αποτελούσε ένα σημείο με 13 διαστάσεις. Με την PCA, μετατρέπουμε τα δύο κέντρα των συστάδων σε ένα σημείο που ορίζεται από δύο συντεταγμένες, την x και την y, και επομένως μπορεί να αναπαρασταθεί σε γράφημα προκειμένου να ξεχωρίσουμε τα clusters μεταξύ τους.

Οι συστάδες διαμορφώνονται ως εξής :



Βλέπουμε πως με το decomposition των δεδομένων μας, είναι πολύ πιο εύκολο να διακρίνουμε πώς χωρίστηκαν σε συστάδες. Επίσης, φαίνεται πως ο διαχωρισμός των δεδομένων είναι εμφανής, κάθε σημείο έχει ανατεθεί σε μία συστάδα, χωρίς θόρυβο.

DBSCAN

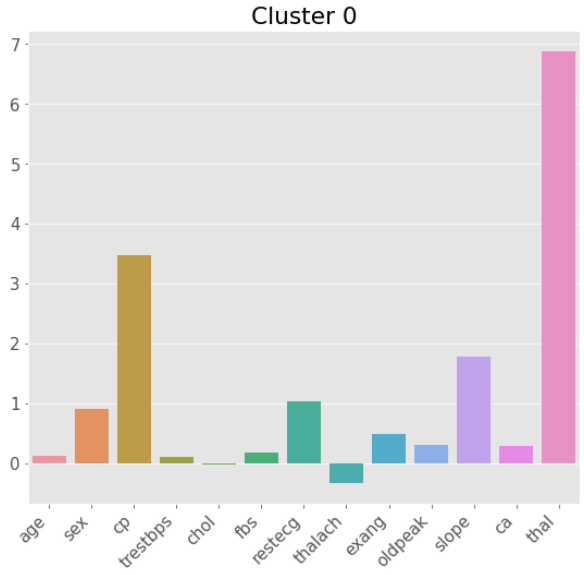
Ο αλγόριθμος DBSCAN εντοπίζει τις πυκνότητες μεταξύ των δεδομένων, δηλαδή τις ομαδοποιήσεις που προκύπτουν από τα δεδομένα που εμφανίζουν τις περισσότερες ομοιότητες, και με βάση αυτά δημιουργεί συστάδες. Θα τον υλοποιήσουμε ρυθμίζοντας τις δύο κυριότερες παραμέτρους του :

- MinPts (min-samples): Ο ελάχιστος αριθμός σημείων που απαιτούνται για να δημιουργηθεί ένα cluster.
- Epsilon (eps): Η μέγιστη απόσταση που μπορεί να έχουν δύο σημεία δεδομένου ότι ανήκουν στο ίδιο cluster.

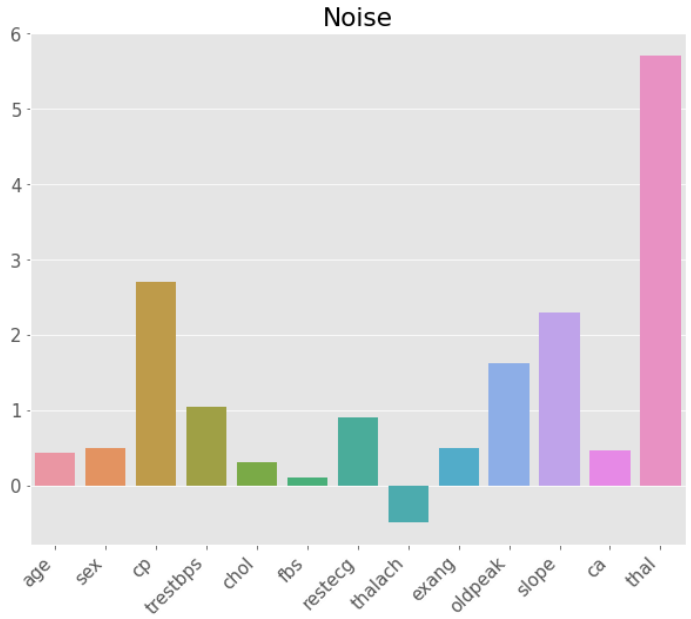
Σκοπός μας είναι να βρούμε τον συνδυασμό αυτών των παραμέτρων που θα μας δίνει το μεγαλύτερο silhouette_score. Υπολογίζουμε τις αποστάσεις μεταξύ των σημείων με τον αλγόριθμο kd-tree.

Χρησιμοποιώντας τις παραμέτρους eps=3 και min_samples=5, παίρνουμε silhouette_score = 0.2896035847127383, το οποίο είναι το καλύτερο δυνατό που μπορεί να προκύψει από τον αλγόριθμο. Οι συστάδες που προκύπτουν είναι 2, και έχουν ετικέτες 0,1. Δέκα παρατηρήσεις δεν ανατέθηκαν σε καμία συστάδα. Αυτά τα σημεία-παρατηρήσεις θεωρούνται θόρυβος, και έχουν ετικέτα -1.

Όπως και νωρίτερα, θα εξετάσουμε πώς διαφοροποιούνται τα γνωρίσματα του dataset, ανάλογα το cluster στο οποίο έχουν τοποθετηθεί. Παίρνοντας τη μέση τιμή κάθε γνωρίσματος στη κάθε συστάδα, μπορούμε να δούμε πώς διαφοροποιούνται οι μεταβλητές μας, ανάλογα το cluster στο οποίο έχουν τοποθετηθεί.

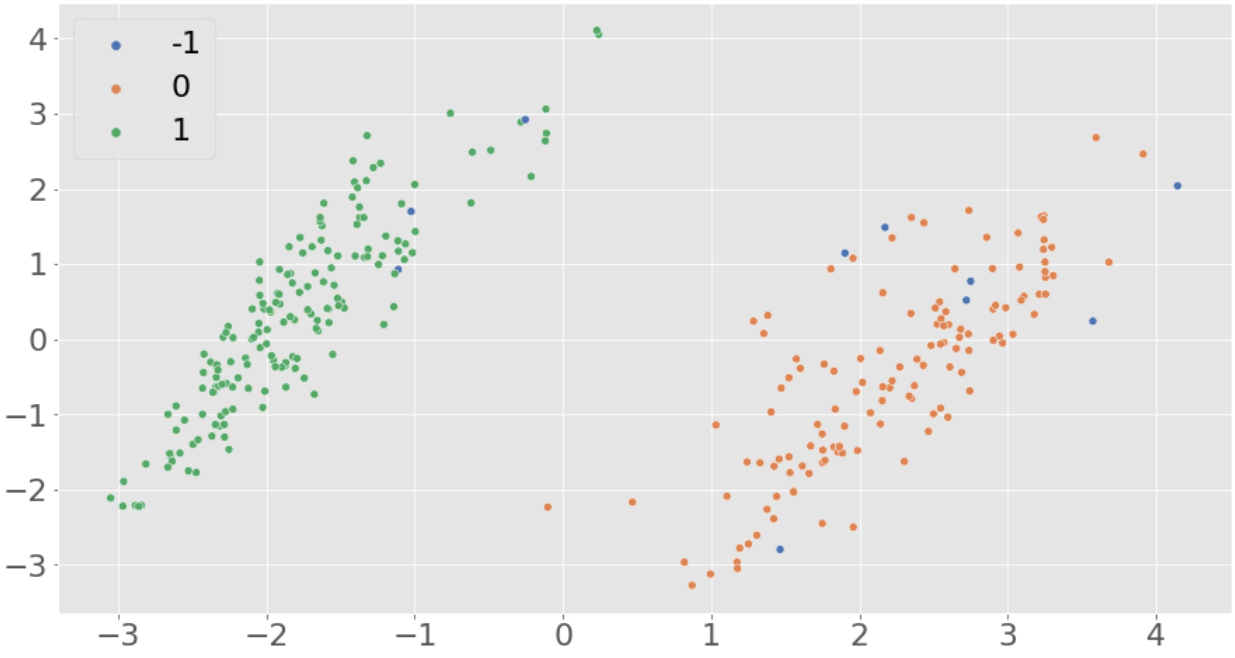


Οι παρατηρήσεις που θεωρούνται θόρυβος έχουν ως εξής :



Φαίνεται δηλαδή πως τα γνωρίσματα που καθόρισαν κατά κύριο λόγο την διαχώριση των δεδομένων ήταν τα age, trestbps, thalach, oldpeak και ca.

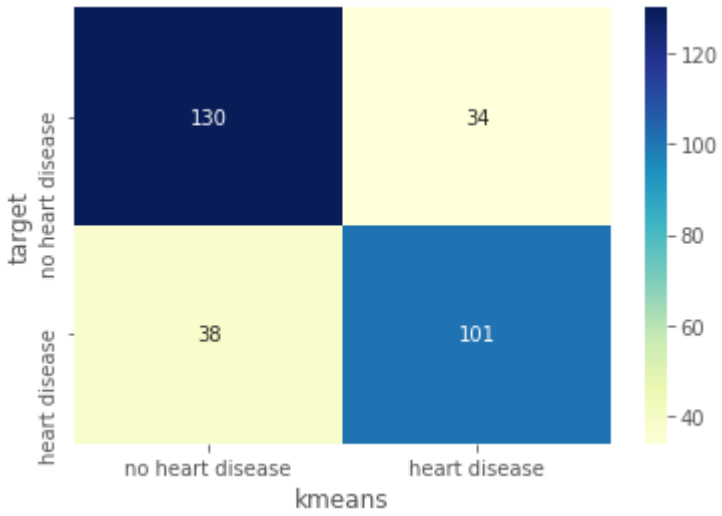
Θα χρησιμοποιήσουμε πάλι την PCA για να πάρουμε μία εικόνα για το πώς συσταδοποιήθηκαν τα δεδομένα μας. Οι συστάδες που διαμορφώνονται στο dataset με δύο διαστάσεις είναι :



Παρατηρούμε πως τα δεδομένα έχουν συσταδοποιηθεί με τρόπο παρόμοιο του kmeans. Τα σημεία που έχουν μπλε χρώμα(έχουν ετικέτα -1), δεν ανήκουν σε κάποια συστάδα.

Επιλογή Τελικού Μοντέλου

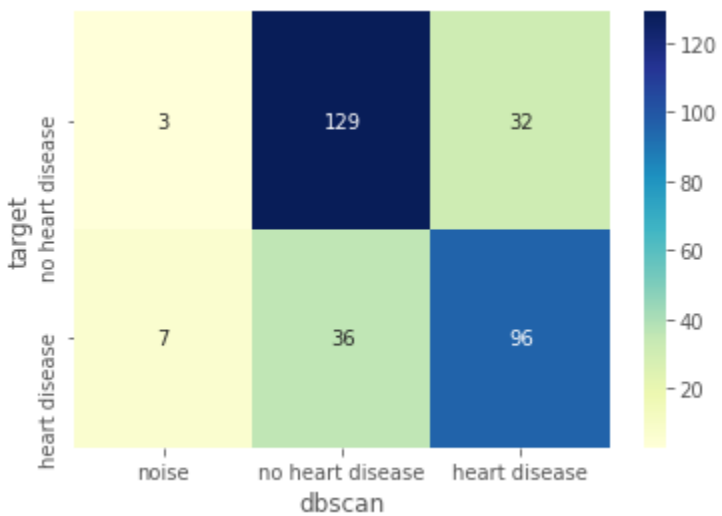
Για να καταλήξουμε στο καλύτερο μοντέλο, θα συγκρίνουμε τις τιμές της target με αυτές που ανατέθηκαν σε κάθε συστάδα, για κάθε μοντέλο. Με άλλα λόγια, θα συγκρίνω τις τιμές της target με τα cluster.labels_ (ετικέτες που προέκυψαν από τον kmeans) και την db.labels_ (ετικέτες που προέκυψαν από τον DBSCAN). Φτιάχνω πίνακες συνάφειας(pd.crosstab) για το κάθε ζευγάρι μοντέλων που θέλω να συγκρίνω, και παίρνω τα αντίστοιχα heatmap. Έχουμε ως εξής:



KMeans:

Βλέπω πως ο kmeans διακρίνει σωστά την απουσία ασθένειας 130 φορές, και την παρουσία της 101 φορές.

Συνολικά, έχουμε 130+101 = 231 σωστές αναθέσεις και 34+38 = 72 λάθος αναθέσεις.



DBSCAN:

Βλέπω πως ο DBSCAN διακρίνει σωστά την απουσία ασθένειας 129 φορές, και την παρουσία της 96 φορές.

Συνολικά, έχουμε 129+96 = 225 σωστές αναθέσεις και 10+34+38 = 78 λάθος αναθέσεις.

‘Ετσι, συμπεραίνουμε πως ο αλγόριθμος kmeans έχει καταφέρει να εντοπίσει τα επίπεδα της target πιο αποτελεσματικά από τον DBSCAN. Επιπλέον, έχει μεγαλύτερο silhouette_score. Συνεπώς, θα επιλέξουμε τον KMeans ως το καλύτερο μοντέλο.

Part 5: Association Rules

Στο τελευταίο αυτό κομμάτι, ζητείται να εξάγουμε κανόνες συσχέτισης από τα δεδομένα μας, δηλαδή να εξετάσουμε εάν υπάρχουν σχέσεις μεταξύ των υποσυνόλων τους, οι οποίες δεν ήταν εμφανείς στα πρώτα στάδια επεξεργασίας του project. Θα δούμε δηλαδή, ποια υποσύνολα των στηλών μας έχουν την τάση να εμφανίζονται συχνότερα, με τι συνδυασμούς, και εν τέλει θα καταλήξουμε στο κατά πόσο οι συνδυασμοί αυτοί επηρεάζουν την target. Για να εξάγουμε τα συμπεράσματα αυτά, θα χρησιμοποιήσουμε τον αλγόριθμο Apriori, με κριτήριο το confidence. Το confidence εκφράζεται ως η πιθανότητα να αγοραστεί ένα αντικείμενο Y, δεδομένου ότι αγοράστηκε μαζί με το X (X → Y). Σκοπός μας είναι να προσδιορίσουμε όλους τους κανόνες συσχέτισης X → Y μέσω του αλγόριθμου Apriori, οι οποίοι δεν ξεπερνούν το κατώφλι min_confidence (ελάχιστη εμπιστοσύνη).

Αφού πραγματοποιήσουμε τις αρχικές διαδικασίες (μετατροπή της target σε binary, αντικατάσταση ελλιπών τιμών, κανονικοποίηση), θα αλλάξουμε τα ονόματα των στηλών μας προκειμένου να είναι εμφανή στη συνέχεια τα itemsets που προκύπτουν. Στην περίπτωση των συνεχών μεταβλητών, θα πραγματοποιήσουμε διακριτοποίηση – τις χωρίζουμε σε υποκατηγορίες για να είναι φανερά τα υποκομμάτια τους που εμφανίζονται συχνότερα στο dataset. Έπειτα από τις μετατροπές μας, τα δεδομένα μας θα έχουν την εξής μορφή :

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak
0	age=(0.505, 2.5]	sex=male	cp=typical	trestbps=(-0.0962, 3.888]	chol=(-0.478, 0.322]	fbs=>120 mg/dl	restecg=hypertrophy	thalach=(-0.289, 0.543]	exang=no	oldpeak=(-0.207, 4.452]
1	age=(0.505, 2.5]	sex=male	cp=asymptomatic	trestbps=(-0.0962, 3.888]	chol=(0.322, 6.138]	fbs=<=120 mg/dl	restecg=hypertrophy	thalach=(-3.443, -0.289]	exang=yes	oldpeak=(-0.207, 4.452]

(δεν απεικονίζονται όλες οι στήλες)

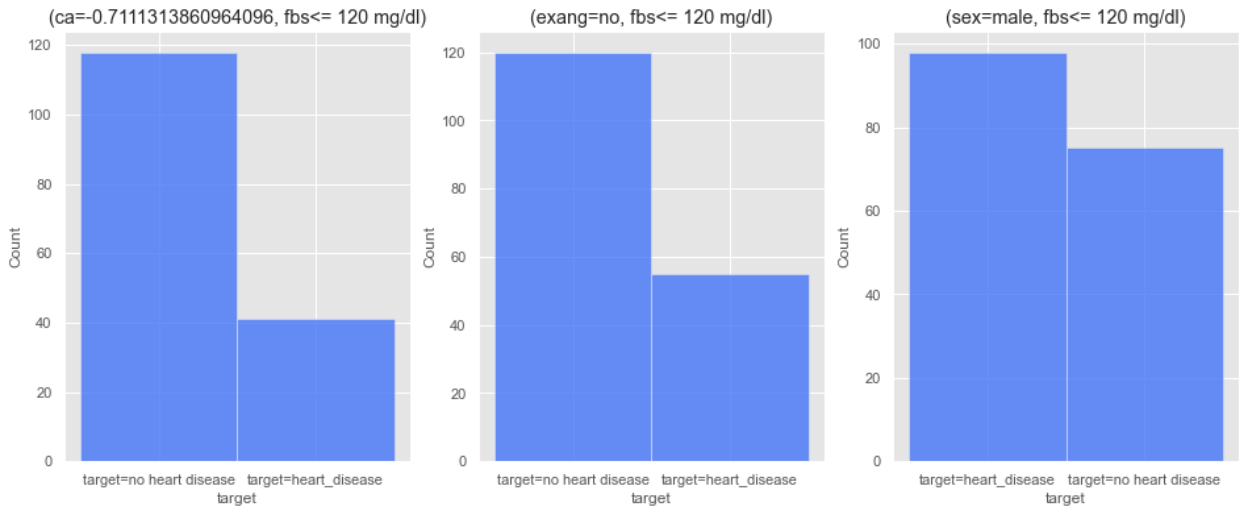
Στη συνέχεια, πρέπει να φέρουμε το dataset μας σε μορφή την οποία μπορεί να επεξεργαστεί ο αλγόριθμος Apriori. Το πρώτο βήμα είναι να το μετατρέψουμε σε μορφή list of lists (κάθε στήλη είναι μία λίστα, που κάθε εσωτερικό της στοιχείο είναι μία λίστα για κάθε γραμμή της).

Το δεύτερο βήμα είναι να μετατρέψουμε το list of lists dataset που έχουμε σε ένα array. Αυτό γίνεται μέσω του **TransactionEncoder.** Ο **TransactionEncoder** μαθαίνει όλες τις unique τιμές όλων των δεδομένων για κάθε μεταβλητή, και μετατρέπει την κάθε γραμμή σε μία λίστα που περιέχει Boolean τιμές, ανάλογες του εάν η κάθε τιμή του dataset εμπεριέχεται σε αυτή την γραμμή ή όχι. Τέλος, μετατρέπουμε το array σε dataframe, κρατώντας τα αρχικά ονόματα των στηλών μας. Στο συγκεκριμένο dataframe, μπορούμε να εφαρμόσουμε τον Apriori.

Ο αλγόριθμος Apriori μας δίνει ποια είναι τα πιο συχνά υποσύνολα(κάθε μεγέθους) σε όλο το dataset. Εμάς μας ενδιαφέρει ποιος συνδυασμός δύο ή παραπάνω υποσυνόλων είναι ο πιο συχνός. Εξάγοντας τους κανόνες συσχέτισης χρησιμοποιώντας το association rules, με βάση το confidence και κατώφλι το min-confidence = 0.7 , βλέπουμε πως αυτοί είναι εξής :

Itemset 1	Itemset 2	Confidence
ca=-0.7111313860964096	fbs=<= 120 mg/dl	0.883333
exang=no	fbs=<= 120 mg/dl	0.857843
sex=male	fbs=<= 120 mg/dl	0.839806

Ας εξετάσουμε τώρα το πώς καθένας από αυτούς του κανόνες επηρεάζει την τιμή που θα πάρει η target.



Μπορούμε να δούμε πως όταν εμφανίζεται ο συνδυασμός των τιμών ca=-0.7111313860964096 και fbs=<= 120 mg/dl, όπως και των τιμών exang=no και fbs=<= 120 mg/dl, τα περισσότερα άτομα είναι υγιή, επομένως έχουμε μία ένδειξη πως οι συνδυασμοί αυτοί θα προβλέψουν υγιή ασθενή. Ο συνδυασμός sex=male και fbs=<= 120 mg/dl έχει επίσης περισσότερα υγιή άτομα,

όμως δεν μπορούμε να πούμε πως η διαφορά ανάμεσα σε ασθενείς και υγιείς είναι καθοριστική, ειδικά σε ένα μικρό dataset όπως αυτό.