Analysis of cardiovascular disease risk factors and prediction of its presence in patients

Maryam Lotfigolian

Oslo Metropolitian University
ACIT 4510– Statistical Learning

Table of Contents

Table of Figures

Table of Tables

1. Background

According to NHS [1], Cardiovascular disease (CVD) refers to conditions affecting the heart or blood vessels. Blood clots and fat deposits may form inside the inner walls of the blood vessels in this situation. Consequently, CVD may damage arteries in organs such as the brain, heart, kidneys, and eyes. There are four main types of CVD [1]:

1. **Coronary heart disease:** In this disease, the blood vessels supplying the heart muscle are affected, resulting in:
   - Angina: An insufficient supply of blood to the heart muscle that causes pain.
   - Heart attacks: Sudden interruption of heart muscle blood flow.
   - Heart failure: An inability to pump blood throughout the body.

2. **Strokes and TIAs:** A stroke occurs when blood flow is cut off to a part of the brain, potentially leading to death or serious injury. The transient ischaemic attack (TIA) disrupts brain blood flow for a short period.

3. **Peripheral arterial disease:** In peripheral arterial disease, the arteries to the limbs, most commonly the legs, are blocked.

4. **Aortic disease:** Aortic diseases refer to conditions that affect the aorta. As the largest vessel in the body, the aortic carries blood from the heart to other parts of the body.

Heart attacks and strokes account for more than four out of five CVD deaths, and one-third of these deaths **occur before age 70.** In 2019, CVD contributed to 17.9 million deaths worldwide, representing 32% of all deaths worldwide. Heart attacks and strokes accounted for 85% of this statistic. In the UK, CVD is one of the leading causes of death and disability. In 2016, 840,768 people died from CVD in the United States. While simple lifestyle changes and screenings each year could prevent nearly 200,000 deaths [2].

The exact cause of CVD is unknown and several studies have been conducted to identify important risk factors. However, hypertension, smoking, abdominal obesity, abnormal lipids, diabetes mellitus, as well as stress, low consumption of fruits and vegetables, and inactivity are among the most common risk factors associated with CVD and are responsible for more than 90% of myocardial infarctions [3, 4]. Increasingly, it has been identified that too little (6 hours) or too much (>9 hours) sleep can also contribute to hypertension and metabolic syndrome [5].

Often unnoticed **increases in multiple risk factors rather than a significant increase in a single factor, lead to CVD**, and there is an alarming increase in the number of people with multiple cardiovascular risk factors [3]. The presence of some risk factors such as **obesity** can increases CVD risk since obesity is strongly associated with other cardiovascular risk factors, such as hypertension, diabetes, and high cholesterol levels. So, a person with obesity is more likely to have other risk factors. According to the World Health Organization, **hypertension** is one of the leading causes of premature death in the world [2]. Hypertension can be determined by two numbers, namely systolic and diastolic blood pressure as shown in figure 1[6].

Systolic blood pressure: As a heartbeat occurs, the heart pushes blood into the arteries. This force is measured by systolic blood pressure. During this phase, known as systole, blood

pressure is at its highest. The systolic blood pressure of a person is considered normal if it is less than 120 mmHg (millimeters of mercury). Obtaining multiple readings of your systolic blood pressure above 180 mmHg indicates that you should seek medical care. However, overall systolic blood pressure greater than 130 is considered to be high blood pressure at different levels. A second variable called diastolic pressure is necessary in order to determine the blood pressure [6].

Diastolic blood pressure: Between beats, the heart rests to refill with blood. Diastole refers to the pause between beats. During this pause between heartbeats, your diastolic blood pressure is measured. In healthy individuals, normal diastolic blood pressure during quiet rest is under 80 mmHg.1 If you suffer from high blood pressure, your diastolic blood pressure is typically higher even when you are at rest. A dangerously low diastolic blood pressure is defined as less than 60 mmHg, and a dangerously high diastolic blood pressure is defined as greater than 110 mmHg [6].

BLOOD PRESSURE CATEGORY	SYSTOLIC mm Hg (upper number)		DIASTOLIC mm Hg (lower number)
NORMAL	LESS THAN 120	and	LESS THAN 80
ELEVATED	120 – 129	and	LESS THAN 80
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 1	130 – 139	or	80 – 89
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 2	140 OR HIGHER	or	90 OR HIGHER
HYPERTENSIVE CRISIS (consult your doctor immediately)	HIGHER THAN 180	and/or	HIGHER THAN 120

Figure 1: An identification system for understanding blood pressure readings developed by the American Heart Association

Various and more effective treatments for blocked arteries are now available, such as clot-buster drug therapy. A number of heart attacks, sudden deaths, and strokes can often be prevented by the cessation of tobacco use, the reduction of salt in the diet, the consumption of fruit and vegetables, regular physical activity, and the avoidance of harmful alcohol use, controlling high blood pressure, and taking medications such as statins, aspirin, and beta-blockers. Health policies that create conducive environments are necessary to motivate people to adopt and maintain healthy behaviors [1, 2].

2. Objective

It is essential to determine which features and types of data are most helpful in predicting CVD. As a result of the information gained from this kind of dataset, physicians may be able to modify their current case history methods to collect more useful information from their patients. And they contribute to the **enhancement of diagnostic accuracy** and the **streamlining of the diagnostic process**.

In this study, we analyzed some risk factors (not all of them as mentioned in the background) to understand their importance and used classification methods to detect

the presence of CVD in the patient. Additionally, we compared classification methods to determine which method is more likely to accurately predict CVD presence in the patient. it is vital to detect CVD at an early stage so that lifestyle changes can be made that will minimize the complications in patients at high risk. And it is possible to prevent premature death by identifying those at the highest risk of CVD and ensuring they receive appropriate treatment.

3. Data Dictionary

This dataset was collected during a medical examination in a CVD study on residents of the town of Framingham, Massachusetts, and was provided by https://www.kaggle.com. There are 7000 entries in this dataset, as well as 15 variables (features) that provide information from patients. Dataset features are divided into three types.

1. Demographic (Objective): Data based on facts.
2. Behavioral (Subjective): Data provided by the participant.
3. Medical (examination): Medical examination results.

Features		Based on the dataset	**Numerical (unit)/Categorical**
Demographic	Age	age	Numerical (day)
	Gender	gender	Categorical (0: Female, 1: Male)
	Height	height	Numerical (cm)
	Weight	weight	Numerical (kg)
Medical (current)	Systolic blood pressure	ap-hi	Numerical (mm Hg)
	Diastolic blood pressure	ap-lo	Numerical (mm Hg)
	Cholesterol	cholesterol	Categorical (1: normal, 2: above normal, 3: well above normal)
	Glucose	gluc	Categorical (1: normal, 2: above normal, 3: well above normal)
Behavioral	Smoking	smoke	Categorical (0: Non-Smoker, 1: Smoker)
	Alcohol intake	alco	Categorical (0: Non-Alcoholic, 1: Alcoholic)
	Physical activity	active	Categorical (0: No Exercise, 1: Exercise)
Target	Health status in relation to cardiovascular disease	cardio	Categorical (0: No Disease 1: Disease)

Table 1: Variables and their types.

The **major CVD risk factors can be altered**: being overweight, systolic and diastolic blood pressure (both indicate high blood pressure), high cholesterol, excessive glucose, alcohol consumption, physical activity, and smoking [6]. The **most critical factors that cannot be changed** are increasing age, male gender, and height.

4. Data visualization

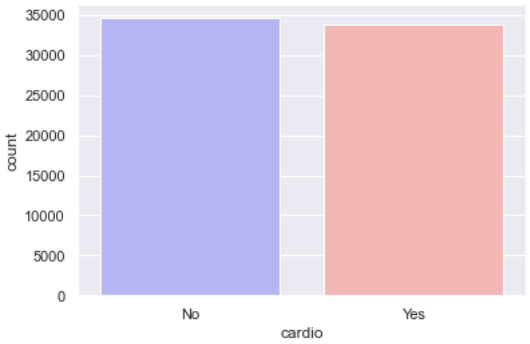


Figure 2: The number of people in the sample with or without CVD.

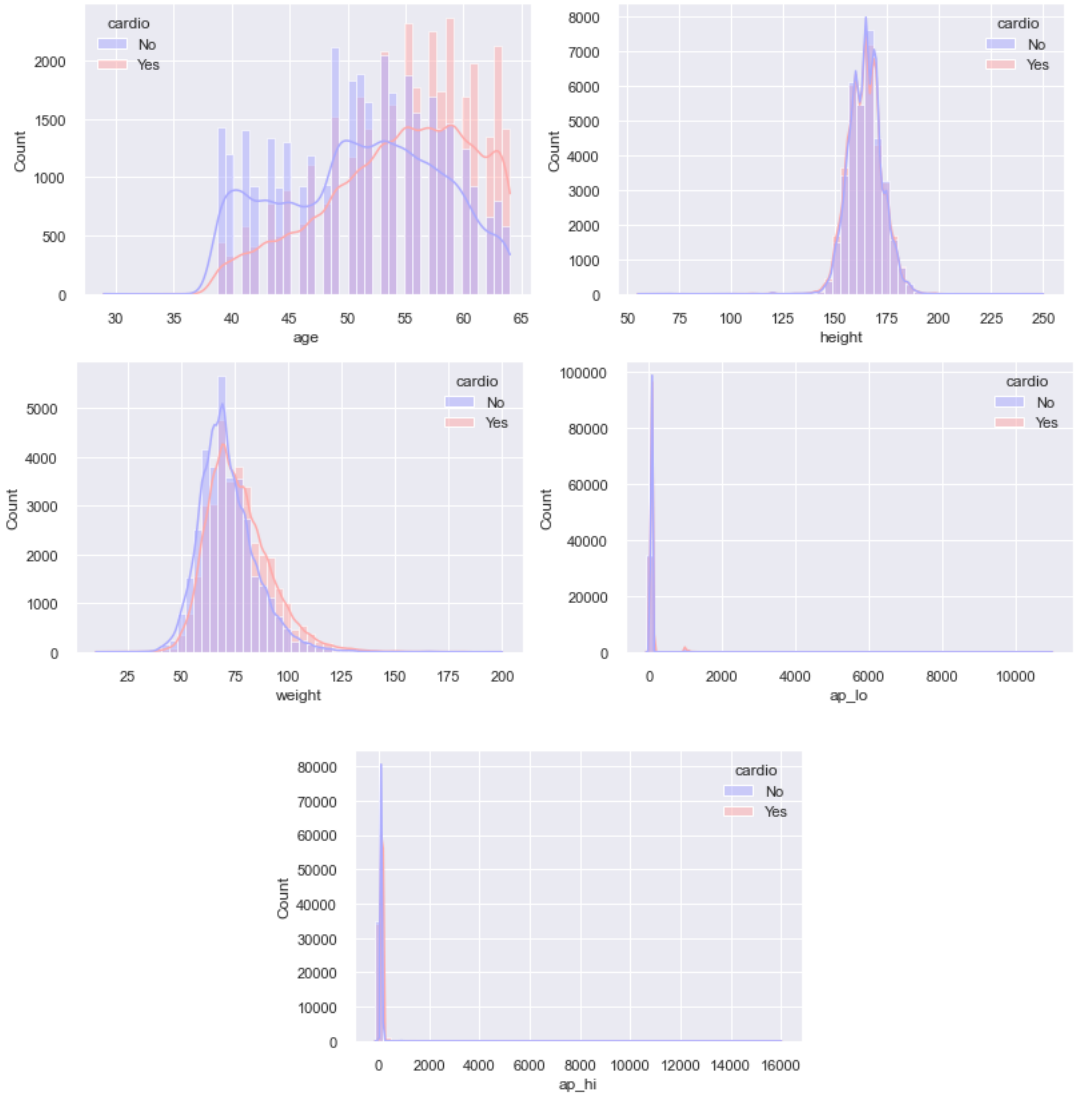


Figure 3: Histograms depict the distribution of numerical variables such as age, height, weight, and systolic and diastolic blood pressure. ap_hi and ap_lo stand for systolic and diastolic blood pressure respectively.

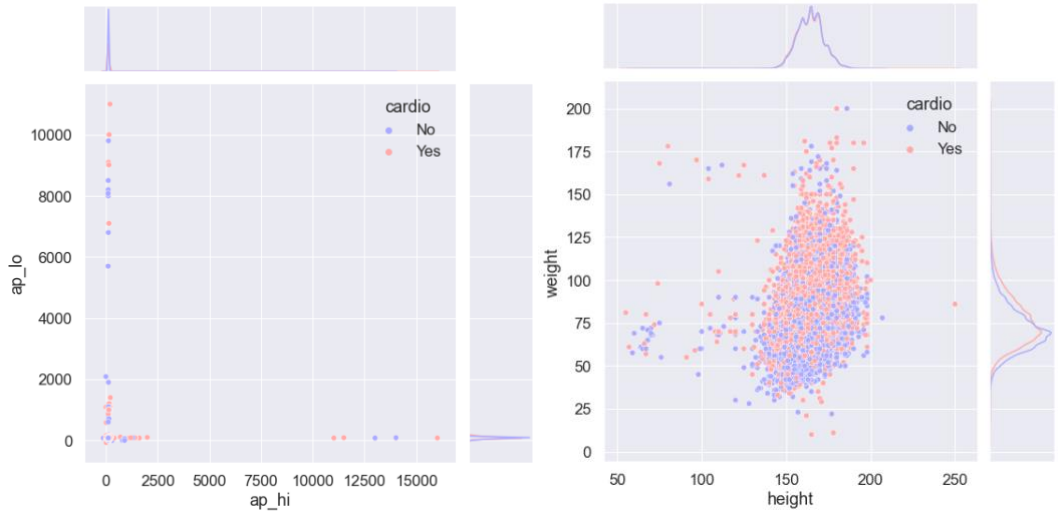


Figure 4: These scatter plots illustrate the relationship between weight and height (mass body identifier) on the right, as well as systolic and diastolic blood pressure (hypertension identifier) on the left.

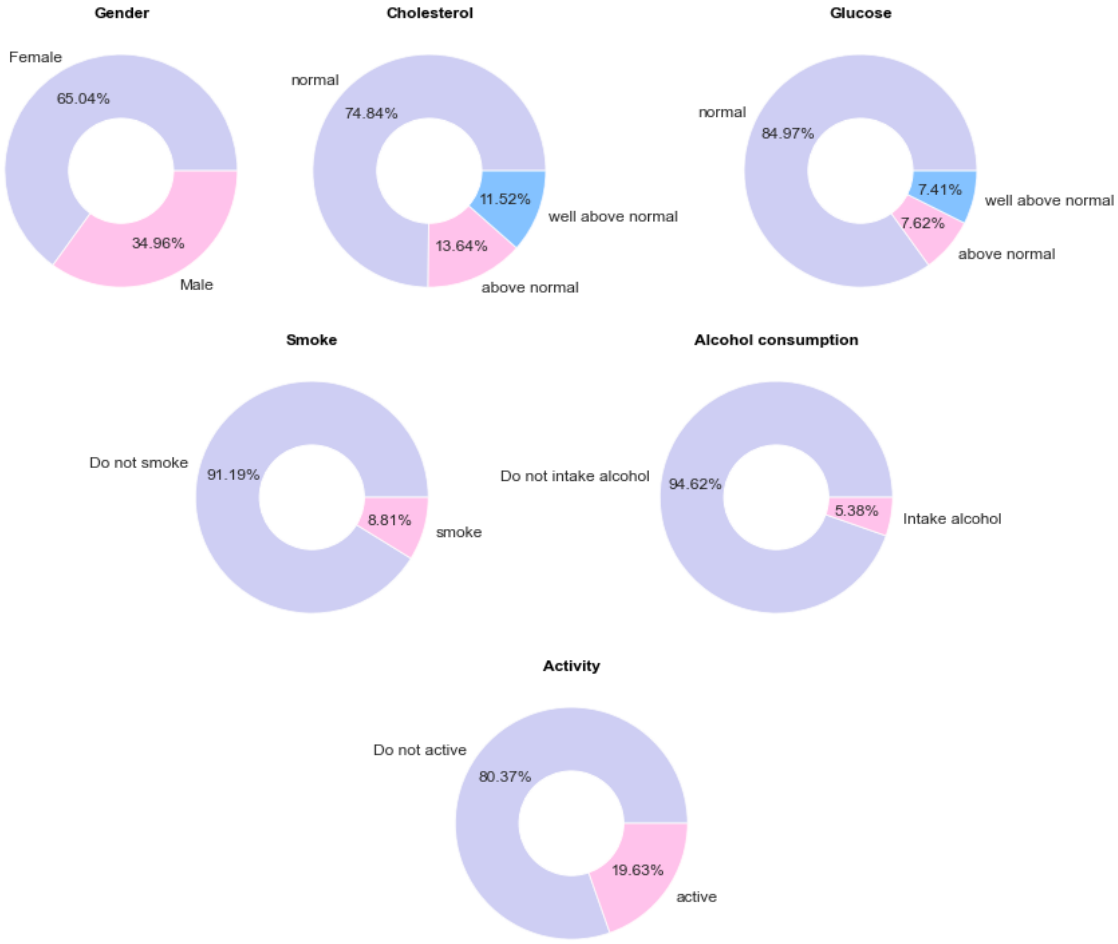


Figure 5: Population percentages for categorical variables.

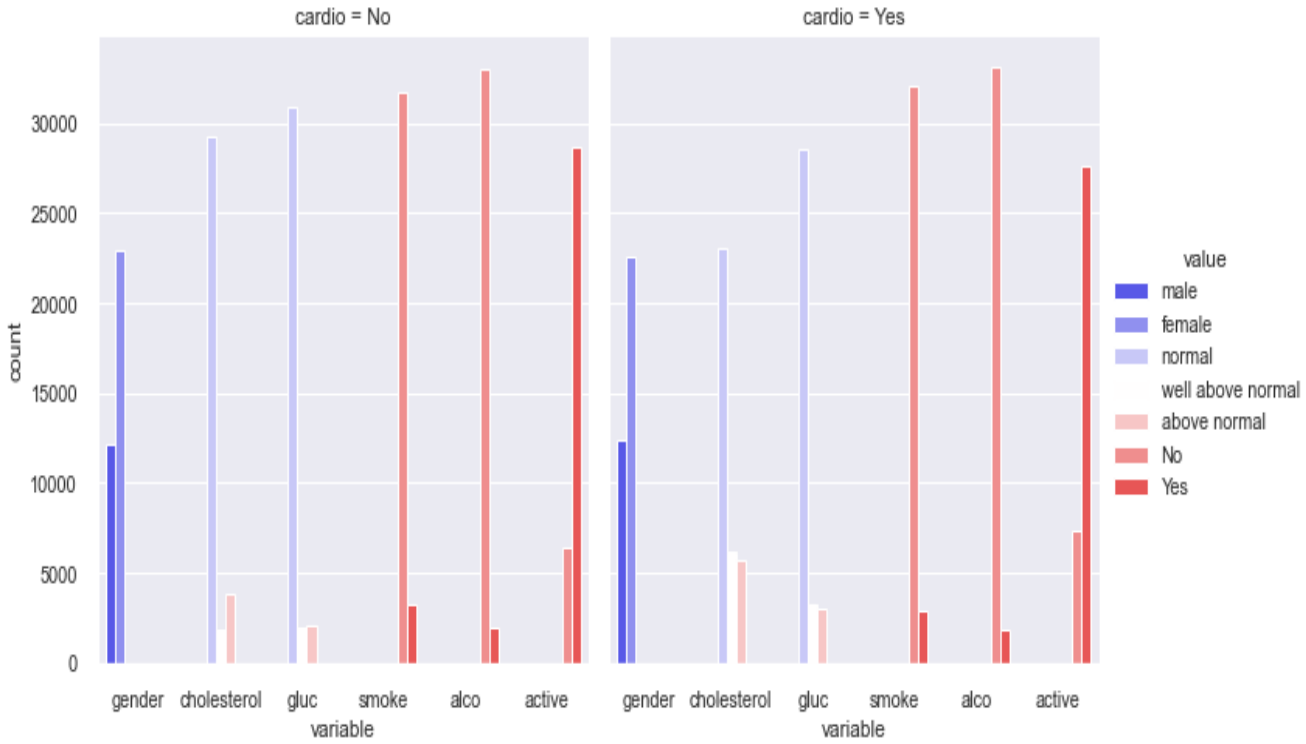


Figure 6: Count plots of categorical variables, such as gender, cholesterol, glucose, smoking status, alcohol consumption, and physical activity, based on whether a person has cardiovascular disease.

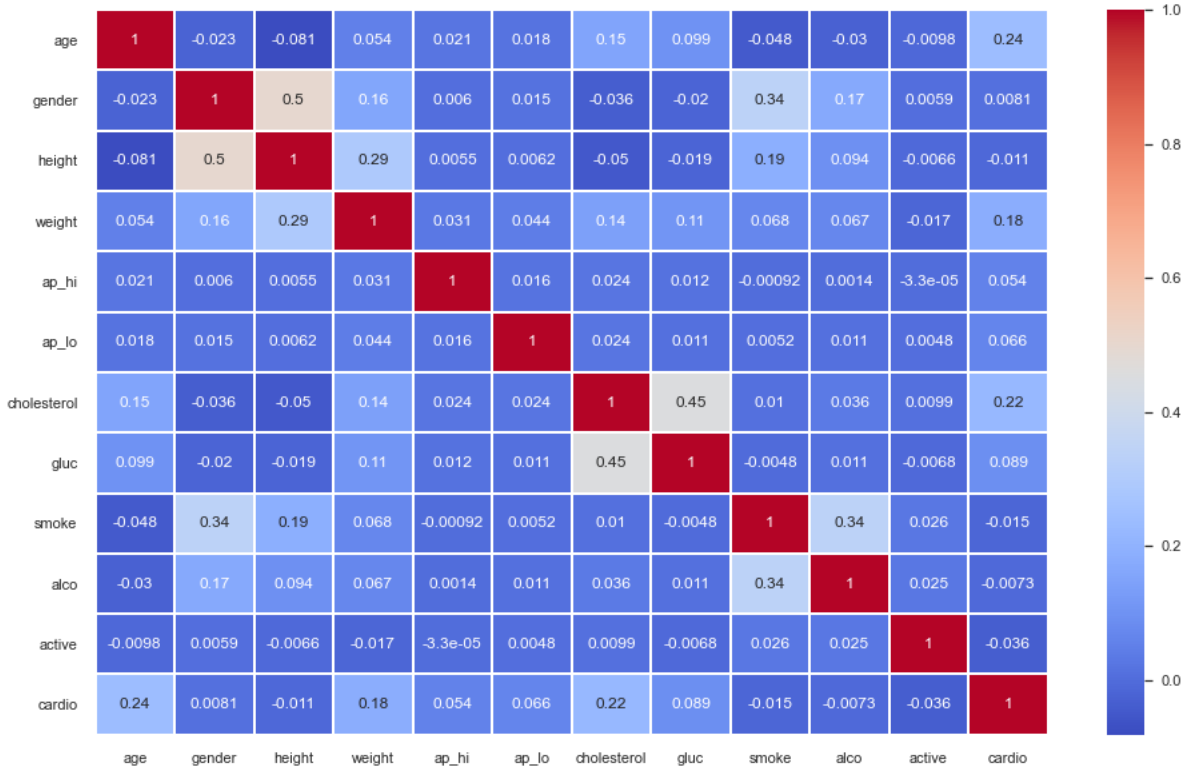


Figure 7: Correlation matrix for analyzing linear relationships between variables.

4.1. Data visualization analysis

Figure 2 shows that, of the 70000 cases in this dataset, 35021 are categorized as having no CVD, whereas 34979 are categorized as having CVD. As a result of this analysis, it appears that the dataset is almost balanced.

Figure 3 illustrates that the age distribution is not normal and exhibits fluctuations. Therefore, CVD is unlikely to be predicted solely by age. However, in general, the risk of CVD as a function of age increases with age increasing. The weight and height distributions are roughly normal. Systolic and diastolic distributions are skewed with long tails at the right end. We can better understand these variables by viewing the scatter plot (Figure 4). The plots indicate that people who weigh more are more likely to develop CVD, regardless of their height. Based on the scatter plot, weight appears to be more influential in detecting the presence of disease than height. The accumulation of points seems much higher in areas where people are shorter and heavier. There appear to be some outliers for weights and heights that are very low or extremely high in both groups. Referring again to the distributions, we can say that the presence of these outliers has an impact on them. It is more obvious in the case of systolic and diastolic blood pressure. Both scatterplots and distributions are affected by outliers. We have values around zero, which means the heart doesn't pump blood to the arteries. Some values are too high and aren't right (see figure 1). Weight and height are considered together due to the mass of the body. Our background suggests that obesity is a risk factor for CVD in patients. This can be determined based on body mass. It is the same for both systolic and diastolic blood pressure. Remember in CVD patients, hypertension is the main cause of premature death.

Categorical variables are depicted in figures 5 and 6. Only 34.96% of the dataset contains males, while 65.04% contains females. The percentage of women in the sample is significantly higher than that of men. The prevalence of CVD among the two groups is, however, the same. Both groups have a nearly equal percentage of CVD, and almost half of each group has CVD. **Consequently, our model might suggest that gender plays a smaller role in diagnosing CVD.**

According to the count plots, cholesterol levels are significantly related to CVD. Therefore, higher cholesterol levels are associated with a greater risk of CVD. Using Python, we find that 76.54% of individuals with cholesterol levels well above normal have CVD. It should be noted, **however, that a normal cholesterol level does not guarantee the absence of CVD**. Cardiovascular disease affects 44.01 percent of people with normal cholesterol levels.

The same applies to glucose. There 84.97% of people have normal glosses. Glucose levels are positively related to CVD, and 62.20% of people with blood glucose levels well above normal are at increased risk for developing the disease. However, based on a count plot, this relationship does not appear to be as strong as that between cholesterol levels and CVD. **As a result, we anticipate that our model will demonstrate that cholesterol is a more relevant characteristic than glucose.**

In our study, smoking and alcohol do not appear strongly correlated with CVD. There is a minor relationship between being active and developing CVD; inactive people are more likely to suffer from CVD. **We find this part of the dataset to be strange.** According to the WHO, the National Health Service, the American Heart Association, and many other publications, these three risk factors are considered significant. After investigating the source, I couldn't find any clear reason why the dataset is in this form in Kaggle. These factors may be

intentionally ignored to focus more on obesity, which can result in other factors, such as hypertension. They might want American society has become more aware of this problem.

Therefore, based on the data presented in the Figures, **cholesterol plays a significant role in identifying CVD. In the next level is glucose. There is also a minor relationship between activity and health. Gender, smoking, and alcohol consumption are not significant factors in this study.**

Based on the correlation matrix (figure 7), we observed that systolic and diastolic blood pressure are weakly correlated with our target variable, cardio. Additionally, cholesterol, weight, and age may affect the target variable more. And glucose levels have a moderate correlation with CVD. But we will see the form of correlation will change after we clean our dataset.

5. Data Description

	count	mean	std	min	25%	50%	75%	max
age	70000.0	52.807329	6.762506	29.0	48.0	53.0	58.0	64.0
gender	70000.0	1.349571	0.476838	1.0	1.0	1.0	2.0	2.0
height	70000.0	164.359229	8.210126	55.0	159.0	165.0	170.0	250.0
weight	70000.0	74.205690	14.395757	10.0	65.0	72.0	82.0	200.0
ap_hi	70000.0	128.817286	154.011419	-150.0	120.0	120.0	140.0	16020.0
ap_lo	70000.0	96.630414	188.472530	-70.0	80.0	80.0	90.0	11000.0
cholesterol	70000.0	1.366871	0.680250	1.0	1.0	1.0	2.0	3.0
gluc	70000.0	1.226457	0.572270	1.0	1.0	1.0	1.0	3.0
smoke	70000.0	0.088129	0.283484	0.0	0.0	0.0	0.0	1.0
alco	70000.0	0.053771	0.225568	0.0	0.0	0.0	0.0	1.0
active	70000.0	0.803729	0.397179	0.0	1.0	1.0	1.0	1.0
cardio	70000.0	0.499700	0.500003	0.0	0.0	0.0	1.0	1.0

Table 2: Statistical description of data.

It is recorded that the minimum age is 30 years old and the maximum age is 65 years old. A minimum height of 55 cm and a maximum height of 250 cm indicate that there are some outliers in the dataset. Also, it's about weight and systolic and diastolic blood pressure.

The minimum weight is 10 kilograms and the maximum weight is 200 kilograms. Both ap_hi and ap_lo recorded extreme values: ap_hi, min:-150, max:16020, ap_low, min:-70, max:11000. According to this statistical description (table 2), the data visualization matches exactly what we have seen.

There is a possibility that these outliers are the result of human error when entering data into .csv format. By removing these outliers from the data, we might be able to improve our prediction model.

6. Data cleaning

There are 70000 entries in this dataset and no missing values, and all data values are either ints or floats. The data frame does not contain any duplicate rows.

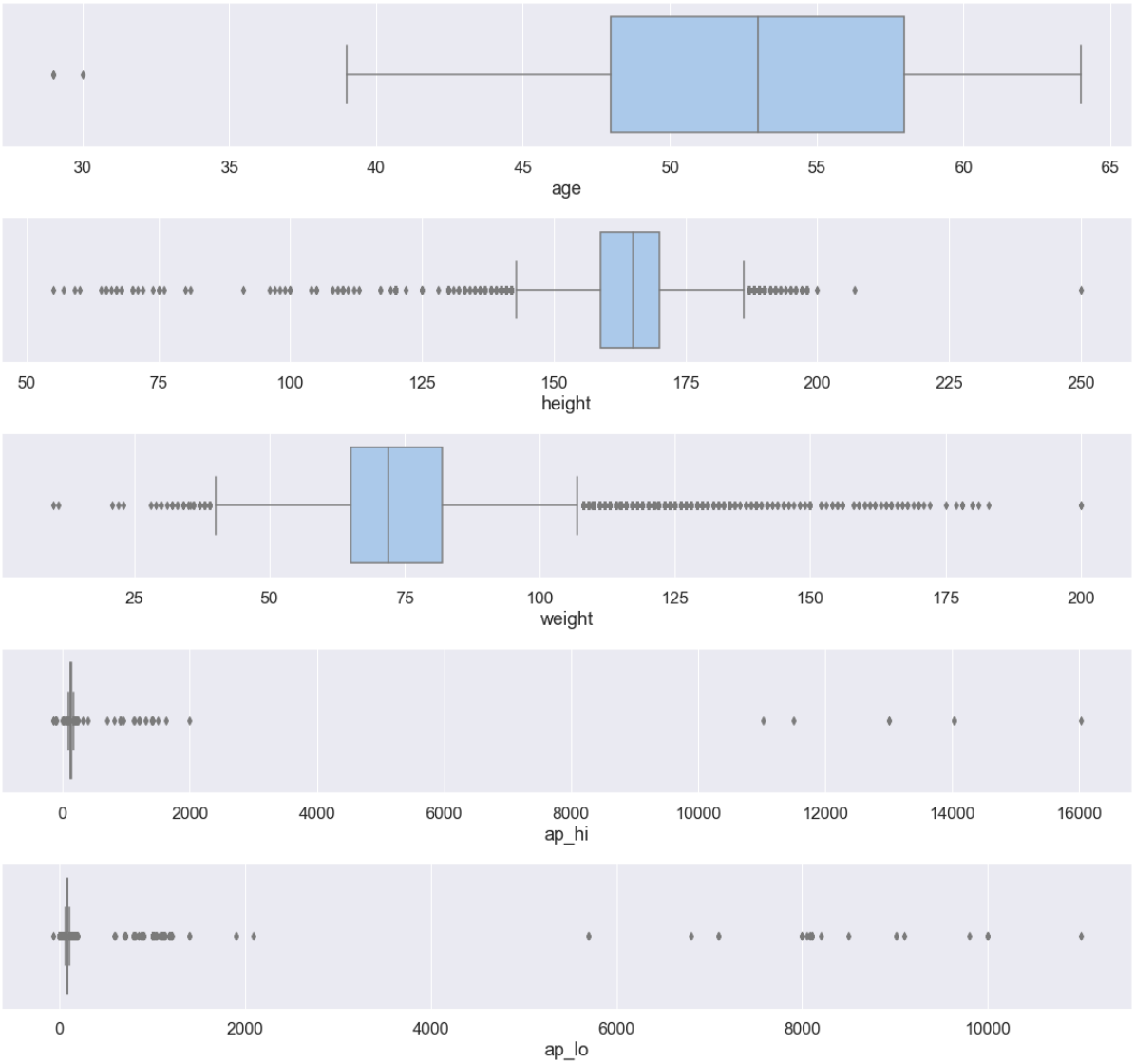


Figure 8: Boxplots of numerical features.

Box plots are used to visualize outliers within a dataset. We can conclude from the tails in the boxplots and the form of the boxes that height and weight are roughly normally distributed. There seems to be some variation in age. As we move on, we'll see how outliers affect the systolic and diastolic distributions by the second visualization of data.

6.1. Outlier removal technique: Interquartile Range method (IQR)

In order to detect and remove outliers, the Inter Quartile Range (IQR) is one of the most widely used methods. This procedure requires the following steps to be followed:

- Calculate the first quartile, Q1.
- Calculate the third quartile, Q3.
- Determine the IQR. Q3-Q1 is the IQR.
- Set the lower and upper limits of the normal data range as Q1 - 1.5 QIR and Q3 + 1.5 QIR, respectively.
- In the case of data points that fall outside of this range, they should be considered outliers and removed from any further analysis.

As part of boxplots, this IQR method is implemented to detect any extreme data points where the maximum point (the end of the high whisker) is Q3+1.5*IQR and the minimum point (the start of the low whisker) is Q1–1.5*IQR.

On the other hand, we need to consider data in which diastolic blood pressure is greater than systolic blood pressure. Again, it may be a result of human error during data entry. In the data set, there are 46 of them. When compared to the data set, the number of outliers and incorrect value entries is very small, just about 2 percent. Therefore, we decided to remove them. After removing outliers from the data and verifying that there are no missing data, 34619 patients have been labeled as being free from CVD, while 33757 individuals have been labeled as having CVD. Taking this analysis into account, it appears that the dataset is relatively balanced.

Here are the revised histograms, scatter plots and correlation matrix which are clearly different from the previous ones. We can now determine the extent to which the outliers affected the ap_hi and ap_lo distributions.

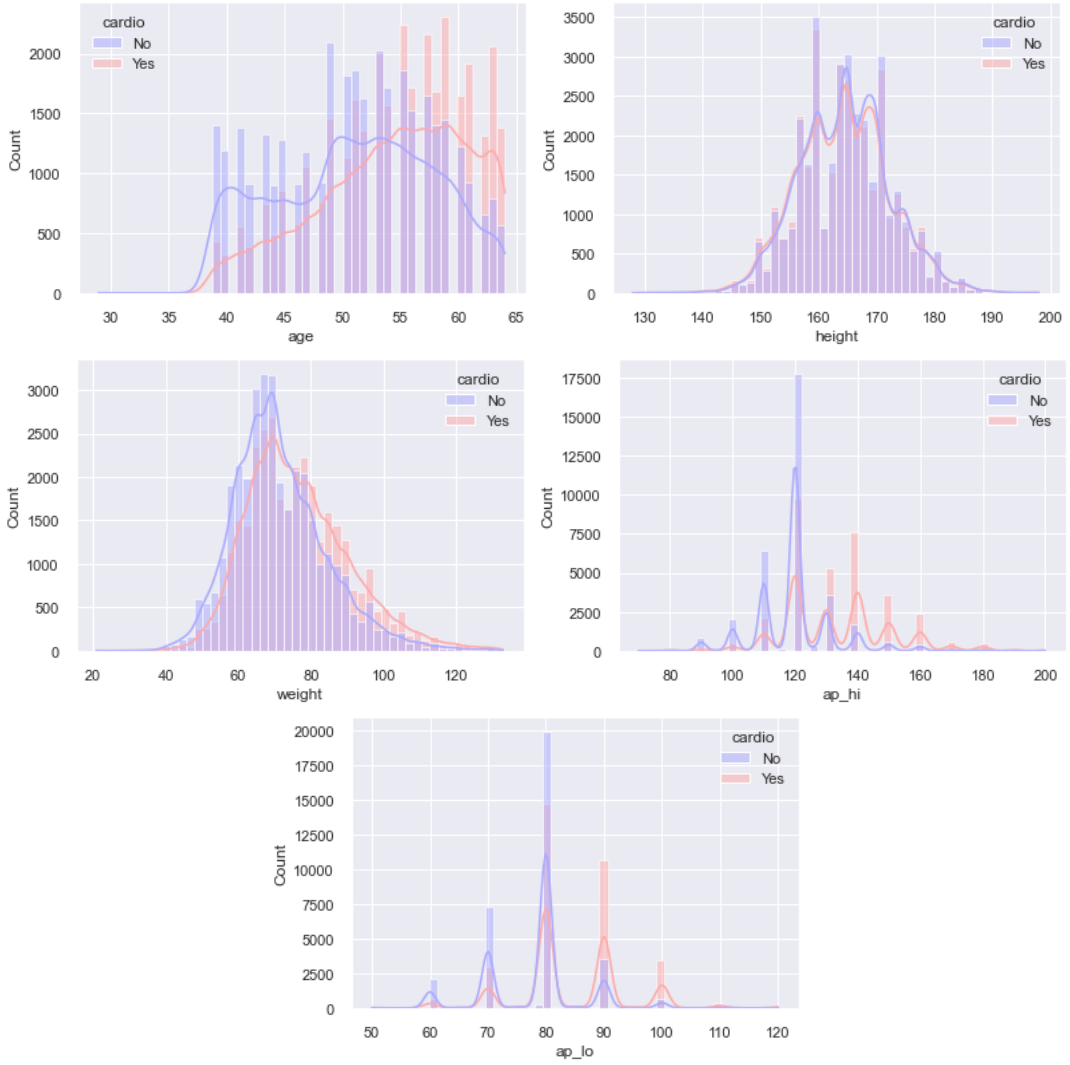


Figure 9: Histogram of numerical variables after removing outliers.

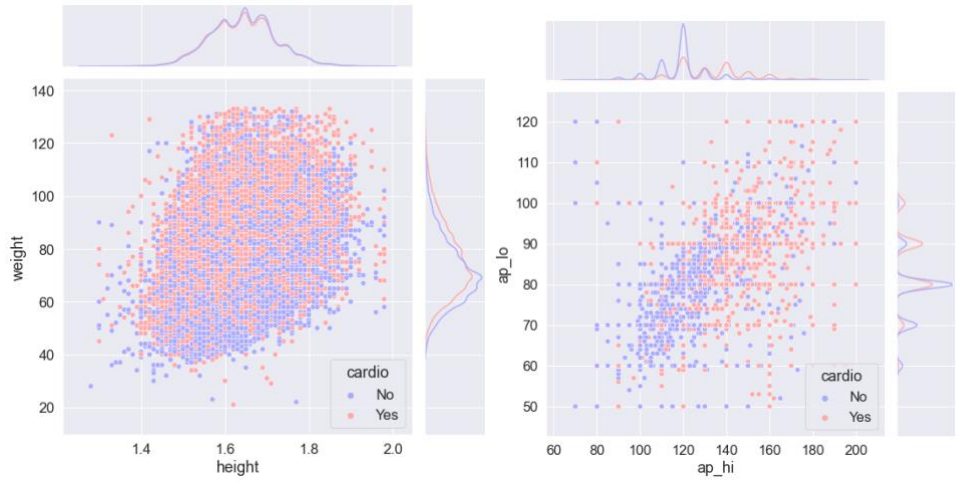


Figure 10: Examining the linear relationship between numerical variables after removing the outliers.

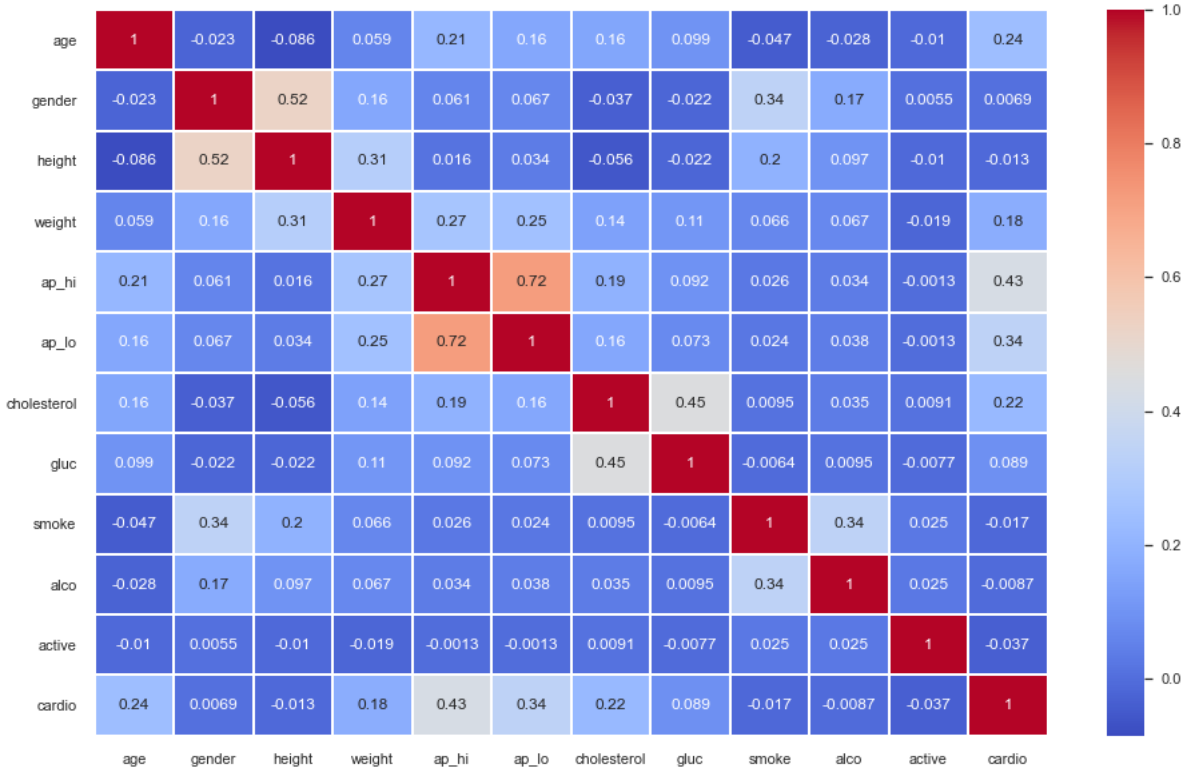


Figure 11: Examining the correlation between numerical variables after removing the outliers.

Based on the scatterplot and kernel densities (figure 10), patients may experience higher ap_hi and ap_lo if they have CVD. In addition, the plot indicates almost a linear relationship between ap_hi and ap_lo, as well as between weight and height. The correlation matrix shows us systolic, and diastolic blood pressure, age, cholesterol, and weight have the highest correlation with our target variable, CVD. And the highest linear relation is between systolic and diastolic blood pressure.

7. Evaluating the importance of each predictor

Feature selection

The selection of features and reduction of dimensionality on sample sets are often used to improve the accuracy scores of estimators or to enhance their performance on datasets with very high dimensions. It can be accomplished in a variety of ways. The following are some of the methods that can be used [7, 8]:

- **Removing features with low variance:** To select features, we use the variance threshold approach as a simple baseline. All features with variances below some thresholds are removed. Default settings remove all zero-variance features, i.e., features with the same value across all samples.
- **Univariate feature selection:** When features are selected using a univariate approach, they are examined individually to determine their relationship with the response variable. Methods such as these are easy to run and understand and are generally effective for gaining a deeper understanding of data. The selection of univariate variables can be accomplished in several ways. SelectKBest, for instance, removes all features except the k highest scoring ones, while SelectPercentile, removes all but the highest scoring percentage of features.
- **Recursive feature elimination:** With recursive feature elimination (RFE), features are systematically selected by considering smaller and smaller sets of features, given a weighting estimator. In the estimation process, an initial set of features is used to train the estimator, and each feature's importance is determined either by using a specific attribute (such as coef_, feature_importances_) or by calling a function. From the current set of features, the least important features are pruned. Until the desired number of features is selected, the process is repeated recursively on the pruned set.
- **Sequential Feature Selection (SFS):** When a proper algorithm is used, multiple features are selected from the set of features. They are evaluated for model iteration with the number of features being reduced and improved because the model meets optimal performance.

Using the univariate selection and applying SelectKBest, we were able to extract the top 13 best feature scores (Table 3). As a result, the following scores have been obtained:

	Feature	Score
4	ap_hi	27106.059803
5	ap_lo	8461.363037
3	weight	5752.759421
0	age	3414.668152
6	cholesterol	1128.981176
7	gluc	144.203457
10	active	18.745411
8	smoke	17.510894
9	alco	4.924292
1	gender	0.542439
2	height	0.043474

Table 3: Univariate Selection scores.

As expected, the presence of CVD is significantly correlated with variables such as systolic and diastolic blood pressure, weight, and age. The relationship between cholesterol and CVD

is stronger than that between glucose and CVD. In addition, we use the inbuilt class features_importances of tree-based classifiers to determine the importance of features.

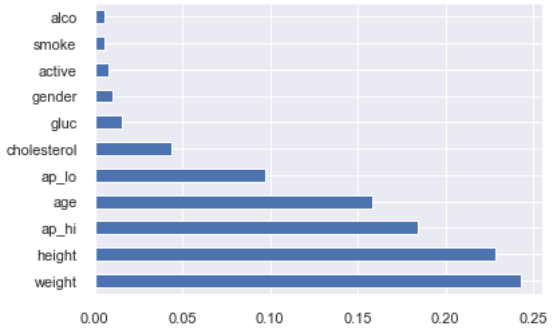


Figure 12: Feature importance visualization.

Based on feature importance, we have eliminated variables such as glucose, gender, activity level, smoking, and alcohol consumption.

8. Machine learning algorithms for prediction

At this stage, we are using Machine Learning (ML) to predict CVD among patients based on our dataset. Since there are many Machine Learning (ML) algorithms that are widely available, we will use multiple algorithms and compare their results.

Our primary algorithms are as follows:

- Logistic Regression
- Decision Tree
- Random Forest
- K Nearest Neighbors
- SVM

8.1. Feature Scaling

Standardization

When comparing measurements with different units, it is helpful to standardize the features around the center 0 and with a standard deviation of 1. **Different measurement scales do not contribute equally to the analysis and may result in bias.** During standardization, your data is assumed to have a Gaussian distribution (bell curve). It is not strictly necessary for the attribute distribution to be Gaussian, but the technique will be more effective if it is. Standardizing data when the data you are analyzing has varying scales is beneficial. For standardization, we divide each feature's value (mean has already been subtracted) by its standard deviation [9].

$$x' = \frac{x - \bar{x}}{\sigma} \qquad (1)$$

Normalization

Similarly, normalization is intended to change the numeric values within a dataset to a common scale without distorting differences between the ranges of values. It is not necessary to normalize every dataset for machine learning. Only **when features have different ranges is it essential to use this method.** It is a wise idea to use normalization when you are unfamiliar with the distribution of your data or when you know the distribution is not Gaussian (a bell curve). In general, the min-max normalization formula for [0, 1] is as follows [9]:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad (2)$$

To understand why scaling is necessary, consider the KNN classifier as an example. KNN classifier predicts the class of a given test observation by identifying the nearest observations to it, the scale of the variables is critical. Variables that are large in scale will have a much greater impact on the distance between the observations and, thus, on the KNN classifier. By contrast, variables are small in scale.

There is a wide range of dimensions and scales of features in the columns of the data. Data features of different scales have adverse effects on the modeling of a dataset. We have scaled our selected features to improve performance in some classifiers since they have different scales.

8.2. Definitions

Logistic Regression

In statistics, logistic regression is a statistical model for estimating the probability of an event occurring by combining one or more independent variables. There are two values labeled "0" and "1" in binary logistic regression. The independent variables are either binary variables (two classes) or continuous variables (any real value). A linear regression cannot be used to estimate the probability of a dependent variable since some of the estimated probabilities become negative! To avoid this problem, we must use a function that produces outputs between 0 and 1 for all values of X. Here, the logistic function is referred to as the sigmoid function, and its value lies between zero and one. There will always be an S-shaped (Figure 12 [10]) curve resulting from the logistic function, so regardless of the value of X, we will obtain a sensible prediction that is not negative [11]. The function is as follows (function 2 is for multiple logistic regression):

Log_odds or logit

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X \qquad (3)$$

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_1 X_p \qquad (4)$$

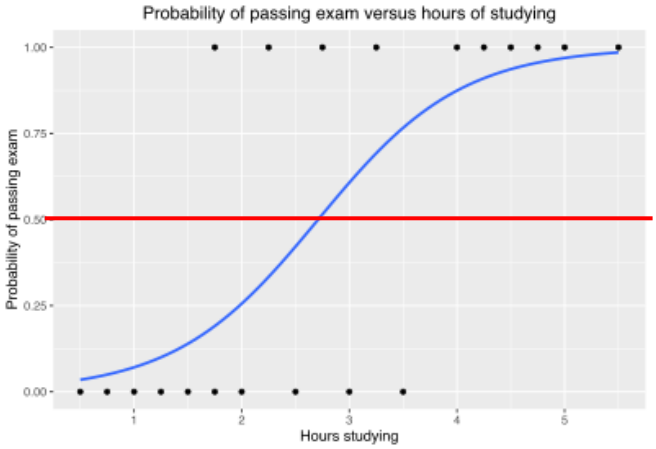


Figure 13: This graph illustrates an example of a logistic regression curve fitted to data. This graph illustrates the probability of passing an exam (binary dependent variable) versus the number of hours spent studying (scalar independent variable).

Considering threshold=0.5 in Figure 13, those who study for more than three hours are more likely to pass the test. These groups may be in category 1.

A ROC curve can be used to visualize the trade-offs of different thresholds. This plot shows the true positive rate versus the false positive rate. An accurate classification model should have a greater number of true positives than false positives at all thresholds. For the ROC curve, the optimal position is towards the top left corner, where specificity and sensitivity are at their highest levels (Figure 14 [12]).

The area under the ROC curve represents the model's classification accuracy; the higher the area, the greater the disparity between true and false positives, and the better the model's ability to predict members of the training dataset. A model with an area of 0.5 is no better than random classification, and a good classifier keeps as far away from that as possible. It is ideal to have an area of 1. It is best if the AUC is close to 1.

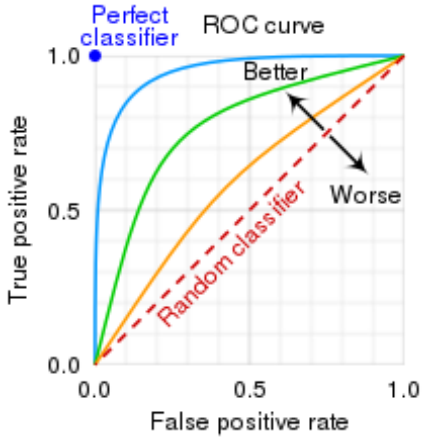


Figure 14: An illustration of the ROC space for a "better" and a "worse" classifier.

Decision Tree

Supervised learning techniques, such as Decision Trees, can be applied to both classification and regression problems, but are most commonly used to solve classification problems. Essentially, it is a tree-structured classifier with internal nodes representing features, branches representing decision rules, and leaf nodes representing outcomes as we can see in figure 15. We can use classification error rate, Gini index, or cross-entropy as a criterion for doing a binary split.

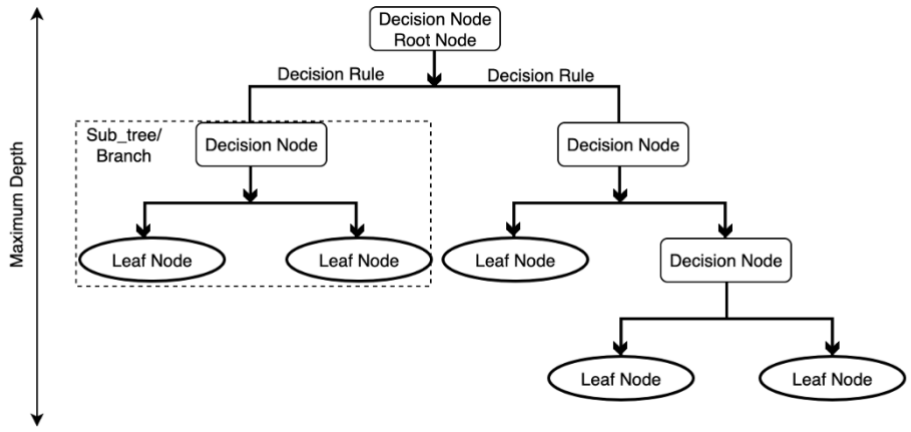


Figure 15: Decision tree structure.

Random Forest

For making decisions, Random Forest utilizes the power of multiple decision trees. Every tree in the random forest predicts a class and the class with the most votes becomes our model's prediction. Several relatively uncorrelated models working together as a committee are likely to outperform any model operating separately.

We force each split in Random Forest to take into account only a subset of the predictors. Therefore, (p - m)/p of all splits won't take the strong predictor into account, but other predictors will have a higher chance of being selected. This process is referred to as decorrelating the trees, which makes the average of the resulting trees less variable and therefore more reliable [11]. This is a simplified picture of a random forest with four classes.

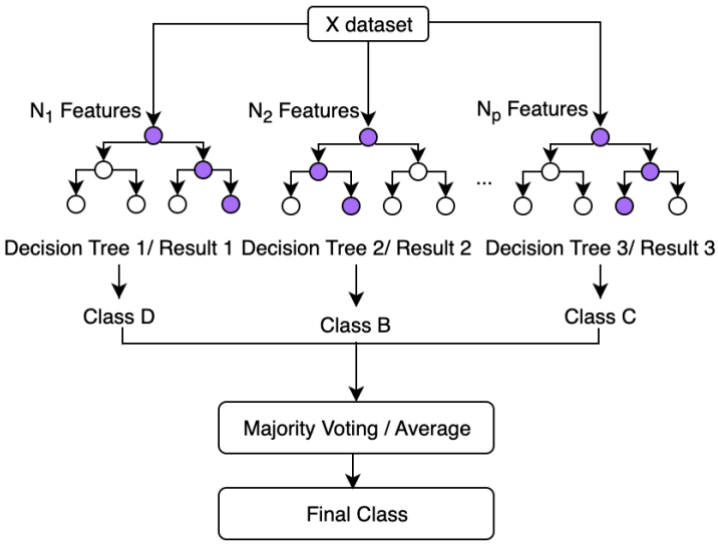


Figure 16: Random Forest classifier.

K Nearest Neighbors

The K-nearest neighbor algorithm is used in machine learning to solve classification and regression problems. Based on a positive integer K and a test observation x_0, the KNN classifier

first identifies the K points in the training data closest to x_0, represented by N_0. In this case, the conditional probability for class j is estimated as the fraction of points in N_0 whose response values equal j [11]:

$$\Pr(Y = j \mid X = x_0) = \frac{1}{K} \sum_{i \in N_0} I\,(y_i = j) \qquad (5)$$

Finally, KNN applies the Bayes rule and classifies the test observation x_0 into the class with the greatest probability.

Support Vector Machine

An SVM algorithm is designed to find a hyperplane in P-dimensional space (P corresponds to the number of features) that distinguishes data points from one another. Several possible hyperplanes may be selected to separate the two classes of data points. Hyperplanes are decision boundaries that categorize data points. There are different classes of data points that fall on either side of the hyperplane. Additionally, the hyperplane's dimension depends on how many features there are.

Data points near the hyperplane are called support vectors, and they influence the hyperplane's position and orientation. The margin of the classifier is maximized by using these support vectors (Figure 17).

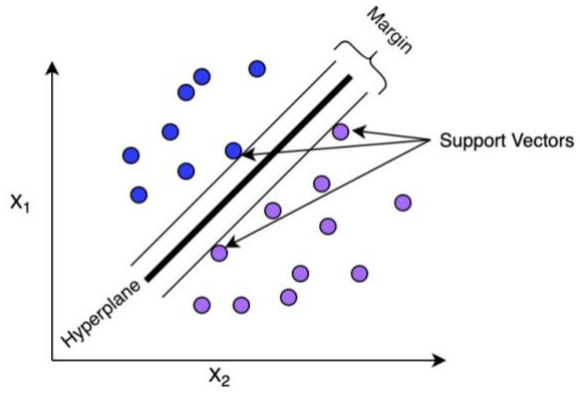


Figure 17: Support Vector Machine visualization.

A support vector machine (SVM) is an extension of the support vector classifier created by enlarging the feature space in a specific manner using kernels.
Kernels are functions that quantify the degree of similarity between two observations. As an example, we could take:

$$K(x_i x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j} \qquad (6)$$

It is known as a linear kernel. The linear kernel measures the similarity between two observations based on Pearson's (standard) correlation. Another example is a polynomial kernel as follows:

$$K(x_i x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d \qquad (7)$$

In the support vector classifier algorithm, a kernel with d > 1 is used instead of the standard linear kernel, which results in a much more flexible decision boundary [11].

GridSearch

Almost all Machine Learning projects involve training different models on a dataset and selecting the most effective model. The model could be improved, however, as we cannot be certain that it is the most appropriate one for the situation at hand. It is therefore our objective to improve the model as much as possible. In order to improve the performance of these models, it is important to set appropriate values for their hyperparameters. Once these parameters are appropriately set, the performance of the model can be greatly improved [13].

For example, some of the Hyperparameters for logistic regression are penalty and C. Overfitting is disincentivized and regulated through penalty (or regularization). The C (or regularization strength) works in conjunction with the penalty to regulate overfitting. Lower values indicate stronger regularization, whereas higher values indicate that the training data should be given a higher weight [14]. The hyperparameters for a decision tree are the maximum depth (figure 15), the minimum sample leaf, and the minimum split. This is the same for the random forest model. We can also set our criterion based on the Gini index or Entropy. For the KNN model identifying the best k can be considered as our hyperparameter. For the SVM model also we can have C and kernels such as polynomial, and linear as our hyperparameters.

Choosing the right parameters can be challenging! We can, however, be a bit lazy and simply try a number of combinations and see what works best. GridSearch refers to this idea of creating a grid of parameters and exploring all of their possible combinations. This method is so common that Scikit-Learn has implemented the functionality in the form of GridSearchCV. We can set our GridSearchCV to give us the best accuracy or recall. **CV stands for cross-validation**, which is the process of testing. This method of resampling is known as cross-validation, which involves collecting different samples from the dataset to perform a test and training on a model at different iterations. Machine learning models can be evaluated using cross-validation, a statistical method. In predictive models, it is used to avoid overfitting, particularly when there is a limited amount of data. Cross-validation involves making a fixed number of folds (or partitions) of the data, analyzing each fold, and averaging the error estimates [13]. [15]

The **hold-out method** is one of the simplest and most basic approaches to Cross Validation, in which the entire dataset is split into training and testing segments. Using training data, we train the model and evaluate it against testing data. Before splitting, the data is first shuffled randomly. Since the model is trained on a different set of data points each time, it may produce different results each time. We cannot also guarantee that the training set we selected is representative of the entire dataset. Additionally, if our dataset is not too large, the testing data may contain some important information that we lose because the model is not trained on the testing data.

The holdout method can be improved by performing **K-fold cross-validation**. By using this method, we ensure that the performance of our model is independent of how we select the train and test sets. The Holdout method is repeated k times on the data set divided into k subsets.

There is a chance that every observation in the original dataset will appear in the training and test sets, so the results are less biased. In the case of limited input data, this approach is one of the most effective.

The training algorithm must be rerun k times, which requires k times the amount of computation needed for an evaluation to be performed.

GridSearchCV requires a dictionary describing the parameters that need to be tested and a model that needs to be trained. As a dictionary, the grid of parameters consists of keys representing parameters and values representing settings to be tested.

Our goal is to improve the models as we go through them with the Gridsearch method with 5-fold cross-validation. For instance, for the KNN model, K is a hyperparameter. We want to maintain a balance between bias and variance by tuning K. A small value of K will result in high variance and low bias. Increasing K values results in a decrease in variance and an increase in bias. After searching for the best k, which is a parameter in our dictionary, with the GridSearch approach we reached the best K, which was 98.

Evaluation Metrics

Various metrics can be used to measure a classifier's or predictor's performance; different fields have different preferences due to varying goals. It is common in medicine to use the terms specificity and sensitivity, while precision and recall are preferred in the computer science [16].

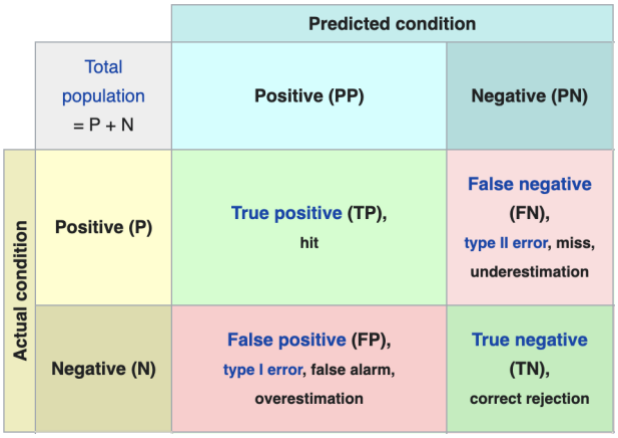


Figure 18: Comparison between actual and predicted conditions.

Accuracy is one of the most commonly used metrics. in classification accuracy can be determined by assessing how often the algorithm correctly classifies a data point. Based on all the data points, accuracy is the percentage of correctly predicted data points. Base don Figure 18 [16] accuracy can be calculated as follow:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (8)$$

Recall or Sensitivity is another evaluation metric. It is one of the important metrics in medicine. A test's sensitivity refers to its ability to identify an individual with a disease as positive. When a test is highly sensitive, there are few false negative results, resulting in fewer cases of the disease being missed [17]. It can be calculated as follow:

$$recall = \frac{TP}{TP + FN} \qquad (9)$$

Precision wants to answer this question: How many positive identifications were correct out of all the positive identifications? This can be calculated as follow:

$$precision = \frac{TP}{TP + FP} \tag{10}$$

It is possible, under certain circumstances, to maximize either recall or precision at the expense of the other metric. We prefer a recall near 1.0 for preliminary disease screening and follow-up examinations, because we want to find all patients with the disease, and we are willing to accept a low precision. We may see some patients with the disease who do not have it - if the follow-up examination does not incur a high cost. In cases where we seek an optimal balance between precision and recall, we can calculate F1. The F1 score represents the harmonic mean of those two metrics. Both measures are equally weighted in the F1 score and calculated as follows:

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{11}$$

Specificity is another common metric in medicine. It refers to the ability of a test or instrument to obtain negative results for a person without a disease. The high specificity of a test indicates that there are few false positives. In screening, a test with low specificity may not be appropriate, as many individuals without the disease will test positive, resulting in unnecessary diagnostic tests.

$$Specificity = \frac{TN}{TN + FP} \tag{12}$$

9. Implementation results and Evaluation Metrics comparison

This dataset has been divided into three parts after **shuffling**: training, testing, and validation. A training part is used to train the model. Then, examine the model's performance in the validation part. After that, we tune the model to see if we can improve its performance. Finally, check the model's performance on the unseen dataset i.e., the test part. The training dataset is 80 percent of the data, validation is 20 percent of the training set, and the test is 20 percent of the dataset.
After we implement the model, before tunning models, we reached the following results:

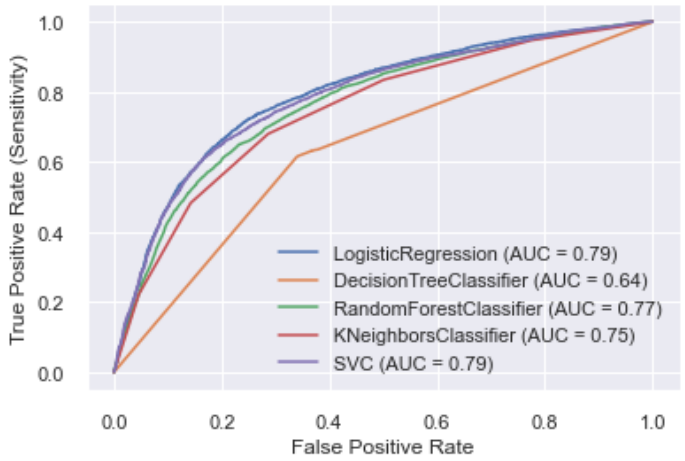


Figure 19: AUC comparison between different models.

Model	Accuracy	Sensitivity	Specificity	F_1	AUC
Logistic regression	**0.73**	**0.67**	**0.79**	**0.71**	**79.15%**
Decision Tree	**0.64**	**0.60**	**0.68**	**0.62**	**64.8%**
Random Forest	**0.70**	**0.68**	**0.71**	**0.69**	**75.59%**
K Nearest Neighbor	**0.70**	**0.68**	**0.71**	**0.69**	**74.85%**
Support Vector Machine	**0.73**	**0.67**	**0.79**	**0.71**	**79%**

Table 4: Evaluation metric results before tuning the models.

After tunning models with the GridSearch method and 5-fold cross-validation:

Figure 20: AUC comparison between different models after tunning.

Model	Accuracy	Sensitivity	Specificity	F_1	AUC
Logistic regression	**0.73**	**0.67**	**0.79**	**0.71**	**79.11%**
Decision Tree	**0.73**	**0.72**	**0.73**	**0.70**	**78.78%**
Random Forest	**0.70**	**0.68**	**0.71**	**0.69**	**75.59%**
K Nearest Neighbor	**0.73**	**0.67**	**0.79**	**0.71**	**79.54%**
Support Vector Machine	**0.73**	**0.67**	**0.79**	**0.71**	**79%**

Table 5: Evaluation metric results after tuning the models.

Model	Test Accuracy with tunned models
Logistic regression	**0.72**
Decision Tree	**0.72**
Random Forest	**0.70**
K Nearest Neighbor	**0.72**
Support Vector Machine	**0.72**

Table 6: Test dataset accuracy based on tunned models.

It has been pointed out that the area under the ROC curve represents the model's classification accuracy; the higher its area, the higher its disparity between true and false positives, and the better the model's ability to predict members of the training set. Figure 19 shows that Logistic Regression and SVM exhibit the best performance. In Table 4, four different metrics are compared, namely Accuracy, Sensitivity, Specificity, and F1. As we mentioned, sensitivity and specificity are the most commonly used metrics when comparing models in medicine. The dataset we are using here is balanced. It means that the number of people with CVD and those without is almost equal. As a result, Accuracy can be considered as a measurement of how often the algorithm can correctly classify items. F1 is used as a balance between precision and recall (see evaluation metrics in the definition part). Upon comparison, Logistic Regression and SVM are almost identical before further tuning.

Next, we tuned our models using GridSearchCV and found the best hyperparameters (see GridSearchCV definition). At this point, the models perform pretty much the same (Figure 20 and table 5). Our tuned models were applied to our test dataset in order to see how well they perform on unseen datasets. In almost 72 percent of cases (table 6), the classifiers correctly predicted whether a person had CVD or not. Since Logistic Regression has the same performance as other classifiers and is a simple model, it was chosen as our classifier!! And another thing we should mention is that tuning for Logistic regression and SVM did not change anything. Before training, we scaled our models, and scaling improve their performance to this level. There might be better hyperparameters for tuning that we could not put inside in Gridsearch dictionory.

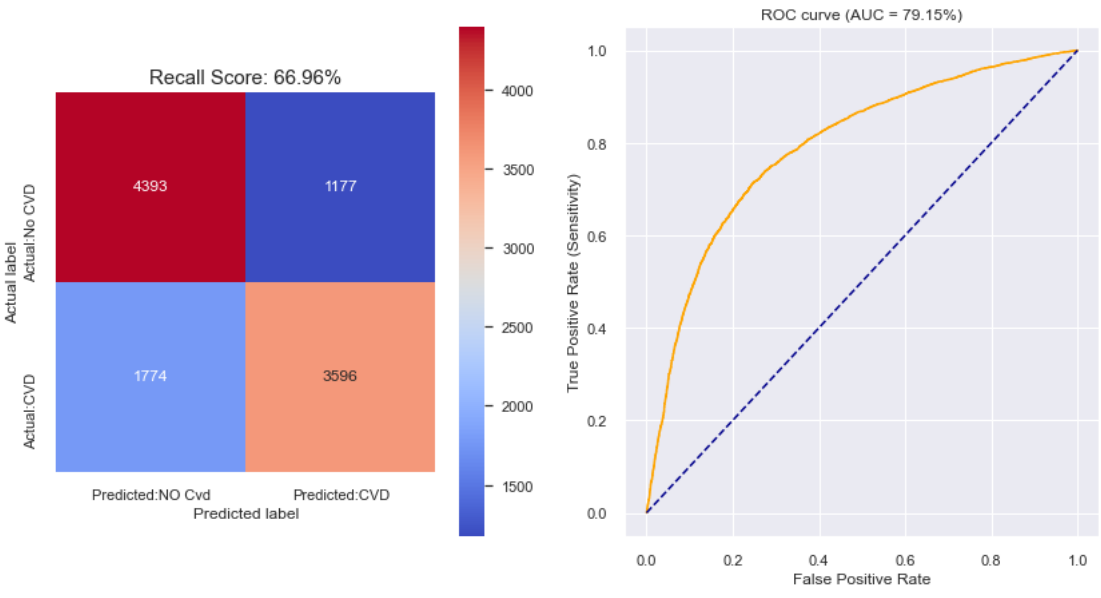


Figure 21: What the confusion matrix tells us?

Based on the confusion matrix above, evaluation metrics are calculated. The actual label is what exists in the real world and the predicted label is what we predict based on our classifier. According to the confusion matrix, 3596 individuals are true positives. The classifier classified them correctly as having CVD. 1774 individuals are classified as FN. It means they have CVD, but our classification system categorizes them incorrectly as they don't have CVD. This is a type II error and is considered to be a misclassification. In total, 1177 individuals are classified as FPs. Essentially, they don't have CVD, but the classifier raised a false alarm about them and placed them in the wrong category, which is error type one. In 4393 cases, the classifier correctly predicted that the individual did not have CVD.

Model	Accuracy	Sensitivity	Specificity	F_1	AUC
Logistic regression	**0.73**	**0.67**	**0.79**	**0.71**	**79.11%**

Table 7: Logistic Regression model performance.

As can be seen from the above statistics, the model is more specific than sensitive. Predictions of negative values are more accurate than those of positive values. Most of the models after tuning get better accuracy, in the meanwhile, they become more specific than sensitive. A low level of specificity may not be appropriate, since many individuals without the disease will test positive, resulting in unnecessary and costly diagnostic tests being performed. The classes in this dataset are balanced for comparison accuracy, and the F1s are acceptable. I would have chosen a Decision Tree classifier if we did not have such a dataset since it is a serious disease and I wanted two metrics of sensitivity and specificity in balance and even sensitivity over specificity.

9.1. Logistic regression

Logit Regression Results

Dep. Variable:	cardio	No. Observations:	68375
Model:	Logit	Df Residuals:	68368
Method:	MLE	Df Model:	6
Date:	Sat, 12 Nov 2022	Pseudo R-squ.:	0.1893
Time:	20:02:04	Log-Likelihood:	-38419.
converged:	True	LL-Null:	-47388.
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-11.1747	0.226	-49.425	0.000	-11.618	-10.732
age	0.0515	0.001	38.083	0.000	0.049	0.054
height	-0.6032	0.120	-5.031	0.000	-0.838	-0.368
weight	0.0110	0.001	15.425	0.000	0.010	0.012
ap_hi	0.0554	0.001	60.178	0.000	0.054	0.057
ap_lo	0.0128	0.001	8.791	0.000	0.010	0.016
cholesterol	0.4448	0.014	31.992	0.000	0.418	0.472

Table 8: Logistic regression results.

Based on the above results, all attributes have P values less than the preferred alpha (5%), indicating a statistically significant relationship between CVD and these attributes. We can use backward elimination to eliminate features with a less significant relationship to CVD. But here all the features have a significant relationship with CVD.

Based on table 4, exp(0.0515) = 1.052856, we will see a 5% increase in the odds of getting diagnosed with CVD for every year increase in age, holding all other features

constant. Similarly, there are 1.1%, 5.6%, and 1.2%, increases in odds of getting diagnosed with CVD for every unit increase in weight, systolic, and diastolic blood pressure.

After plugging the features, we have:

$$\log\left(\frac{p(X)}{1-p(X)}\right) = -11.1747 + 0.0515 * age - 0.6032 * height + 0.0110 * weight + 0.0552 * ap\_hi + 0.0128 * ap\_lo + 0.4448 * cholesterol$$

	Prob of no heart disease (0)	Prob of Heart Disease (1)
0	0.081625	0.918375
1	0.662142	0.337858
2	0.143782	0.856218
3	0.306195	0.693805
4	0.646319	0.353681

Table 9: Individuals' chances of developing cardiovascular disease.

With the knowledge of age, height, weight, systolic and diastolic blood pressures, and cholesterol levels of the individual, we can calculate the probability of having cardiovascular disease and not having cardiovascular disease using the default threshold of 0.5 (see definition of ROC). Table 9 just shows 4 samples. In the first case, the patient is a 63-year-old male, with a height of 1.59 m and a weight of 56 kg. His blood pressure is 130 mm Hg systolic and 90 mm Hg diastolic, his cholesterol and glucose levels are well above normal, and he does not smoke or drink alcohol. There is a 0.92 probability that this person will develop CVD and a 0.081 chance that he will not develop CVD. This person is considered to have CVD by our classifier.

10. Discussion

Based on the information provided by the WHO, the National Health Service, the American Heart Association, and other research, many risk factors may contribute to cardiovascular disease (CVD). Smoking, drinking alcohol, inactivity, eating a poor diet, obesity, high cholesterol, high blood sugar, hypertension, stress, and not getting enough sleep are all examples of risk factors that lead to CVD. Our data only contain some risk factors, not all. Surprisingly, based on this dataset, some of the risk factors, which undoubtedly play an essential role in the development of CVD, do not appear to have any significant relationship with CVD. For example, based on our data, alcohol consumption, smoking, and inactivity have no significant relationship with CVD! There was also a minor relationship between glucose levels and cardiovascular disease! We do not consider this correct in light of different sources such as WHO and NHS. And there was no specific reason or reference in the Kaggle for that.

Regarding this problem with the data set, this report presents what the data tell us. The data indicate that six features significantly correlate with the target variables, including systolic and diastolic blood pressure, weight, height, age, and cholesterol. A person's age or height cannot be changed or controlled. The rest we can control.

We classified individuals as having CVD or not having CVD based on these six features. Logistic regression was the best classifier in terms of accuracy and simplicity.

Cholesterol was one of the features we used for our classification. Blood tests can measure cholesterol levels, which require a hospital examination. Now Let us consider the case in which we do not consider cholesterol in the training phase and eliminate it from the model. We will undoubtedly have less accuracy (we tested our model once without cholesterol, and the accuracy was almost 70%). Putting aside cholesterol, we can set up a simple examination in a simple application for individuals based on other features that we have. People can easily measure their weight, height, and systolic and diastolic blood pressure at home. They can determine whether they are at risk of CVD or not. From these four features, two famous risk factors appear, namely BMI and Hypertension, which I believe is the primary objective of this dataset to raise public awareness.

The body mass index (BMI) is calculated by taking a person's weight in kilograms and dividing it by their height in meters squared. Underweight, healthy weight, overweight, and obesity can be determined using the BMI. Although BMI does not measure body fat directly, it appears to be moderately correlated with more direct measures of body fatness. Moreover, BMI seems to be as strongly associated with metabolic and disease outcomes (CVD) as other more direct measures of body fatness [18]. In the case of Hypertension, systolic and diastolic blood pressure are two characteristics that indicate it in an individual. It is also possible to easily obtain both systolic and diastolic blood pressure at home without needing to undergo costly examinations. A suitable screening test should be available. Suitability criteria include adequate sensitivity and specificity, low cost, ease of administration, safe, imposes minimal discomfort upon administration, and acceptable to both patients and practitioners!

In summary, the data indicates that people can determine whether they are at risk for cardiovascular disease by calculating their body mass index and blood pressure. As a result, we can encourage individuals to take some of the most basic precautions regarding their health to prevent CVD, namely maintaining a healthy weight, without requiring them to measure quantities at home that are more difficult. Furthermore, it also assists in quickly and easily identifying those at a high risk of cardiovascular disease.

11. References

The first picture is available from: https://www.news-medical.net/news/20220511/Association-between-physical-activity-paradox-and-risk-variables-for-cardiovascular-disease.aspx

1. *NHS*. Available from: https://www.nhs.uk/conditions/cardiovascular-disease/.
2. *World Health Organization*. Available from: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds).
3. Dahlöf, B., *Cardiovascular disease risk factors: epidemiology and risk assessment.* The American journal of cardiology, 2010. **105**(1): p. 3A-9A.
4. Anderson, K.M., et al., *Cardiovascular disease risk profiles.* American heart journal, 1991. **121**(1): p. 293-298.
5. Dzau, V.J., et al., *The cardiovascular disease continuum validated: clinical evidence of improved patient outcomes: part I: Pathophysiology and clinical trial evidence (risk factors through stable coronary artery disease).* Circulation, 2006. **114**(25): p. 2850-2870.
6. *American heart association* Available from: https://www.heart.org/en/health-topics/heart-attack/understand-your-risks-to-prevent-a-heart-attack.
7. *Feature selection*. Available from: https://scikit-learn.org/stable/modules/feature_selection.html.
8. *Feature Selection Methods* Available from: https://www.analyticsvidhya.com/blog/2020/10/feature-selection-techniques-in-machine-learning/.
9. *Scaling*. Available from: https://en.wikipedia.org/wiki/Feature_scaling.
10. *logistic regression*
11. James, G., et al., *An introduction to statistical learning*. Vol. 112. 2013: Springer.
12. *ROC curve*
13. *GridSearchCV*. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.
14. *Hyperparameters*. Available from: https://medium.com/codex/do-i-need-to-tune-logistic-regression-hyperparameters-1cb2b81fca69.
15. *Cross validation*. Available from: https://en.wikipedia.org/wiki/Cross-validation_(statistics).
16. *Evaluation Metrics*. Available from: https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers.
17. Bolin, E. and W. Lam, *A review of sensitivity, specificity, and likelihood ratios: evaluating the utility of the electrocardiogram as a screening tool in hypertrophic cardiomyopathy.* Congenital Heart Disease, 2013. **8**(5): p. 406-410.
18. *Centers for Disease Control and Prevention*. Available from: https://www.cdc.gov/healthyweight/assessing/bmi/adult_bmi/index.html.

Code references

The first reference is from a 25-hour udemy course that I have passed: https://www.udemy.com/course/python-for-data-science-and-machine-learning-bootcamp/learn/lecture/5733454?start=15#overview

The rest of the references are from Kaggle and GitHub and search other people's work
https://www.kaggle.com/code/neisha/heart-disease-prediction-using-logistic-regression
https://www.kaggle.com/code/marcosvafg/iesb-cia035-aula-03-cross-validation-e-oob
https://www.kaggle.com/code/raihanrizkidwiputra/heart-disease-classification
https://www.kaggle.com/code/raihanrizkidwiputra/heart-disease-classification
https://www.kaggle.com/code/neisha/heart-disease-prediction-using-logistic-regression
https://www.kaggle.com/code/ekramasif/cardiovasculardiseasepredictionusingml
https://www.kaggle.com/code/mertoezcan/comparison-of-different-machine-learning-models
https://www.kaggle.com/code/kazimshaikh/cardiovascular-classification
https://www.kaggle.com/code/zawlinmaung/eda-decisiontree-randomforest-and-adaboost
https://github.com/LeadingIndiaAI/Cardiovascular-Disease-Prediction/blob/master/prediction.ipynb
https://github.com/abhijeetgawas19/CardioVascular-Disease-Prediction-Machine-Learning/blob/main/CardioVascular_Disease_Prediction_Completed.ipynb
https://github.com/shreeya07/Predicting-Cardiovascular-Disease/blob/master/Predicting%20Possibility%20of%20Cardiovascular%20disease.ipynb
https://github.com/nahacka/Cardiovascular_Disease_Prediction/blob/main/Cardio_VN.ipynb

I do believe that my course was the most useful one since people had a different ideas about the dataset and ways of analyzing that. I also preferred to search inside https://scikit-learn.org/stable/ which was very good.

Final Assignment

November 16, 2022

Code references

The first reference is from a 25-hour udemy course that I have passed: https://www.udemy.com/course/python-for-data-science-and-machine-learning-bootcamp/learn/lecture/5733454?start=15#overview

The rest of the references are from Kaggle and GitHub and search other people's work https://www.kaggle.com/code/neisha/heart-disease-prediction-using-logistic-regression https://www.kaggle.com/code/marcosvafg/iesb-cia035-aula-03-cross-validation-e-oob https://www.kaggle.com/code/raihanrizkidwiputra/heart-disease-classification https://www.kaggle.com/code/raihanrizkidwiputra/heart-disease-classification https://www.kaggle.com/code/neisha/heart-disease-prediction-using-logistic-regression https://www.kaggle.com/code/ekramasif/cardiovasculardiseasepredictionusingml https://www.kaggle.com/code/mertoezcan/comparison-of-different-machine-learning-models https://www.kaggle.com/code/kazimshaikh/cardiovascular-classification https://www.kaggle.com/code/zawlinmaung/eda-decisiontree-randomforest-and-adaboost https://github.com/LeadingIndiaAI/Cardiovascular-Disease-Prediction/blob/master/prediction.ipynb https://github.com/abhijeetgawas19/CardioVascular-Disease-Prediction-Machine-Learning/blob/main/CardioVascular_Disease_Prediction_Completed.ipynb https://github.com/shreeya07/Predicting-Cardiovascular-Disease/blob/master/Predicting%20Possibility%20of%2 https://github.com/nahacka/Cardiovascular_Disease_Prediction/blob/main/Cardio_VN.ipynb

I do believe that my course was the most useful one since people had a different ideas about the dataset and ways of analyzing that. I also preferred to search inside https://scikit-learn.org/stable/ which was very good.

[68]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import scipy.stats as st
import statsmodels.api as sm
import statsmodels.formula.api as smf
from sklearn import svm, metrics
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import roc_curve
```

```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import accuracy_score, r2_score, confusion_matrix,␣
 ↪plot_confusion_matrix, plot_roc_curve, classification_report, recall_score,␣
 ↪precision_score
from xgboost import XGBClassifier
from xgboost import plot_importance
from sklearn import metrics
from sklearn.feature_selection import SelectKBest
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.feature_selection import chi2
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split, KFold, cross_val_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import make_pipeline
from sklearn.metrics import (
    zero_one_loss,
    accuracy_score,
    f1_score,
    confusion_matrix,
    ConfusionMatrixDisplay,
    classification_report
)
from sklearn.metrics import accuracy_score as acc
from mlxtend.feature_selection import SequentialFeatureSelector as sfs
from sklearn.linear_model import RidgeCV
from time import time
from sklearn.svm import SVC
import warnings
from yellowbrick.style import set_palette
from yellowbrick.model_selection import LearningCurve, FeatureImportances
from yellowbrick.contrib.wrapper import wrap
from yellowbrick.classifier import PrecisionRecallCurve, ROCAUC, ConfusionMatrix
import yellowbrick
import pickle

from matplotlib.collections import PathCollection
from statsmodels.graphics.gofplots import qqplot
warnings.filterwarnings('ignore')
%matplotlib inline
```

[3]:
```
df = pd.read_csv('cardio_train.csv', sep = ';')
```

[4]:
```
df.head()
```

[4]:

	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
0	0	18393	2	168	62.0	110	80	1	1	0	0	1	0
1	1	20228	1	156	85.0	140	90	3	1	0	0	1	1
2	2	18857	1	165	64.0	130	70	3	1	0	0	0	1
3	3	17623	2	169	82.0	150	100	1	1	0	0	1	1
4	4	17474	1	156	56.0	100	60	1	1	0	0	0	0

[5]:
```
df.shape
```

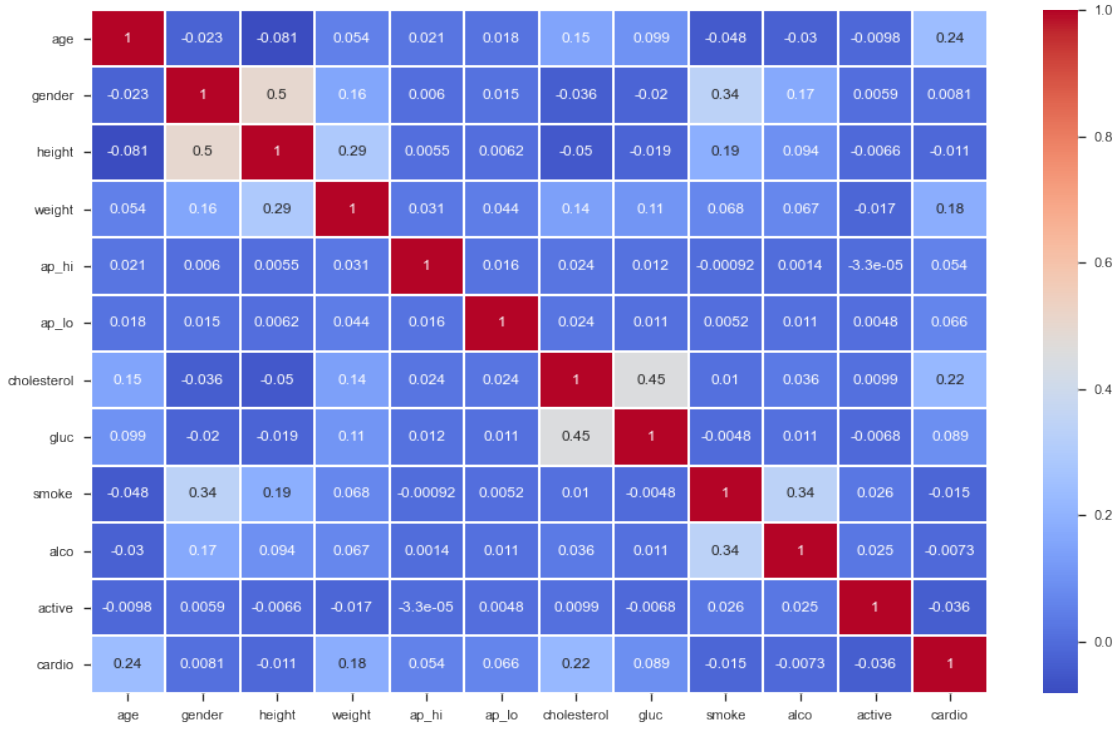
[5]:
```
(70000, 13)
```

[6]:
```
df.drop(['id'],axis=1,inplace=True)
```

[7]:
```
df['age'] = df['age'].map(lambda x: round(x/365.25,2)).astype(int)
```

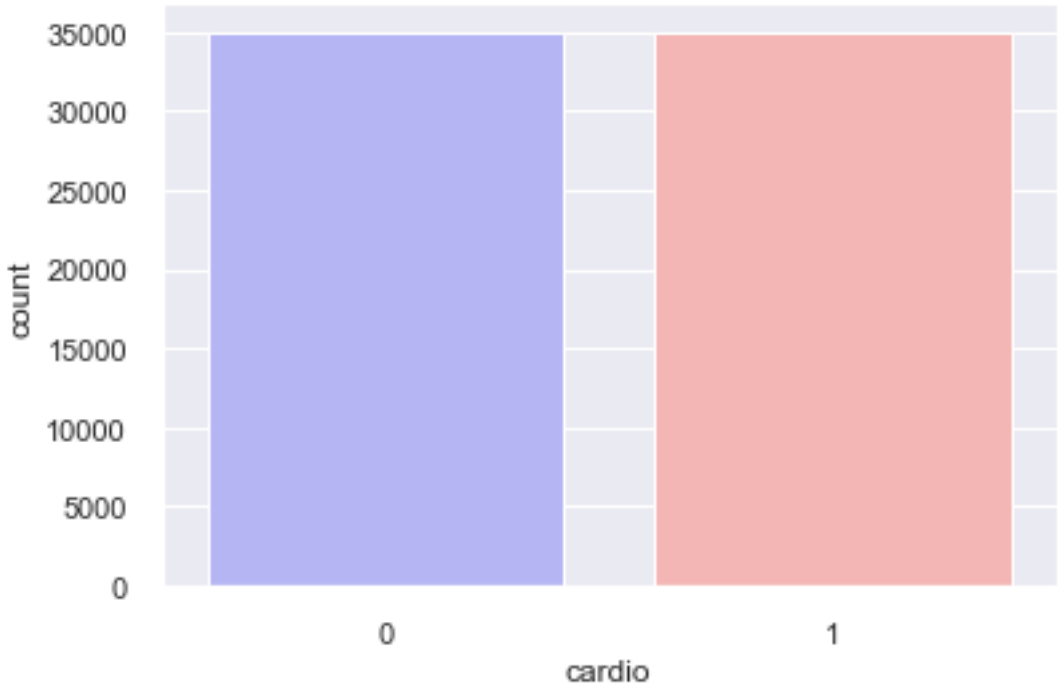
[8]:
```
sns.set_context("notebook", font_scale=1)
plt.subplots(figsize=(16,10))
sns.heatmap(df.corr(),annot=True, linewidth=2,cmap='coolwarm')
```

[8]:
```
<AxesSubplot: >
```



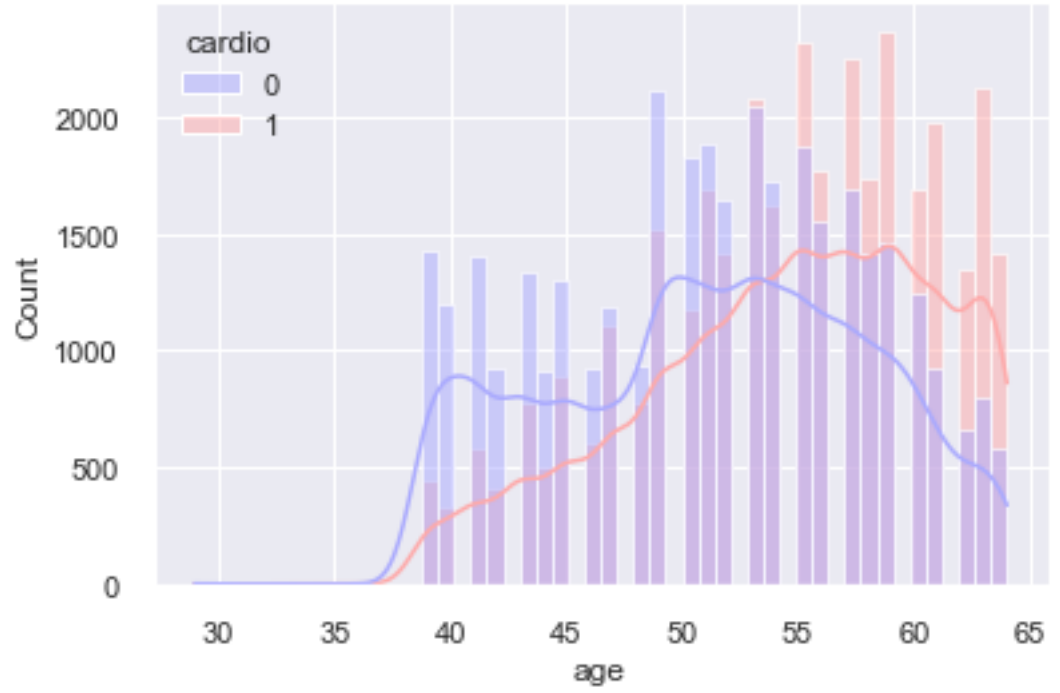
[9]:
```
sns.set(font_scale=1)
sns.countplot(x='cardio',data=df,palette='bwr')
```

[9]: <AxesSubplot: xlabel='cardio', ylabel='count'>



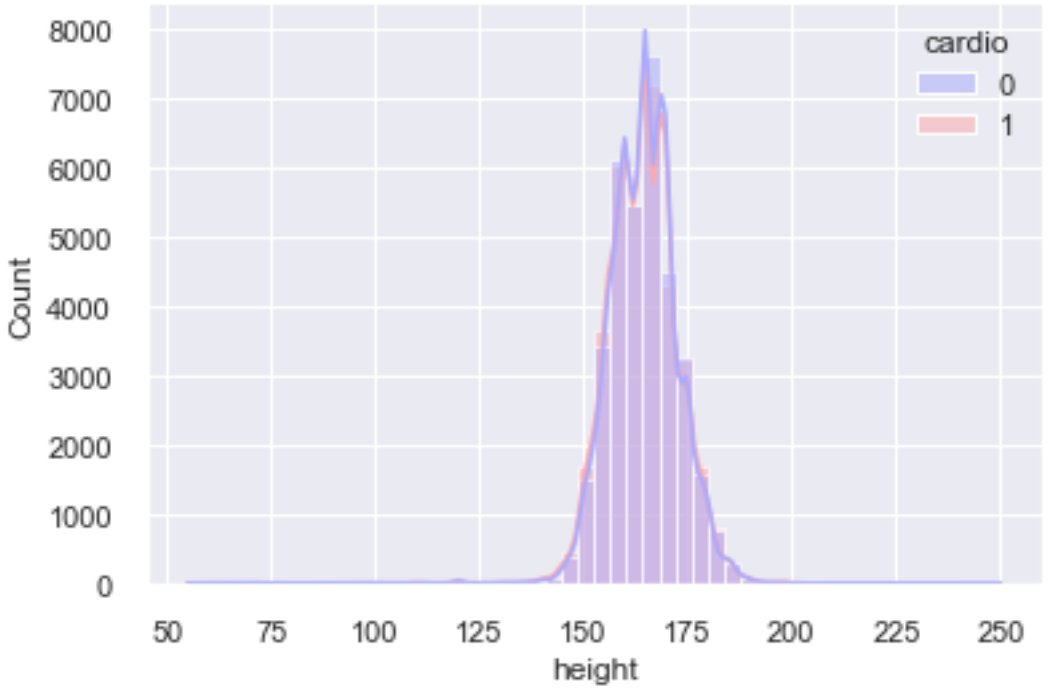
[10]:
```
sns.set()
sns.histplot(data=df,x='age',kde=True,bins=50,hue='cardio',palette='bwr')
```

[10]: <AxesSubplot: xlabel='age', ylabel='Count'>



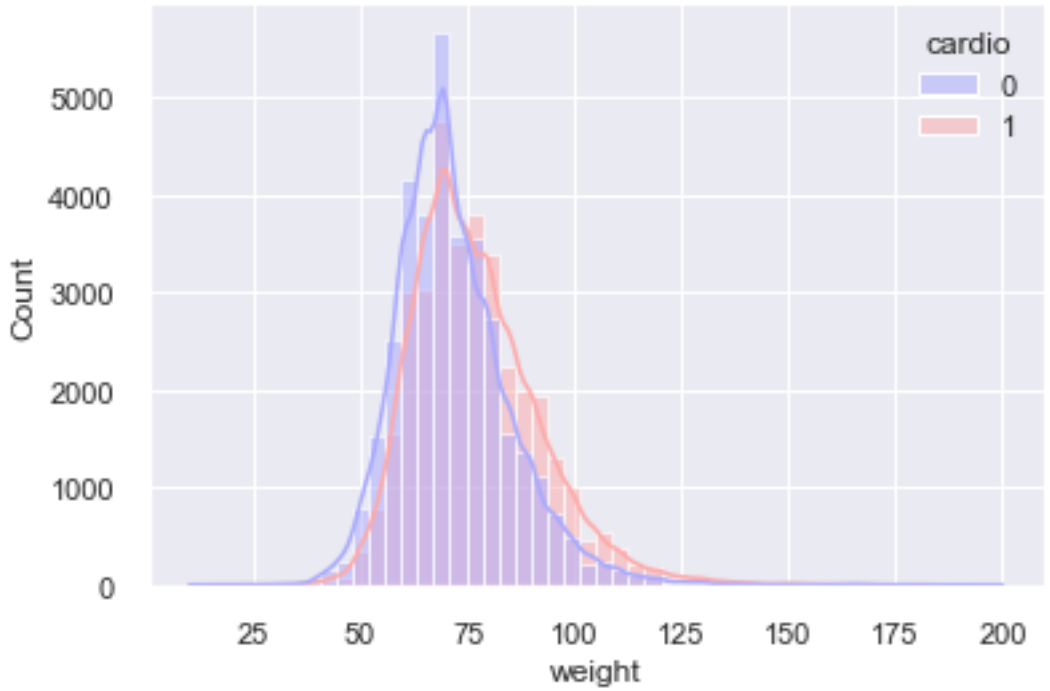
[11]:
```
sns.set()
sns.histplot(data=df,x='height',kde=True,bins=50,hue='cardio',palette='bwr')
```

[11]: <AxesSubplot: xlabel='height', ylabel='Count'>



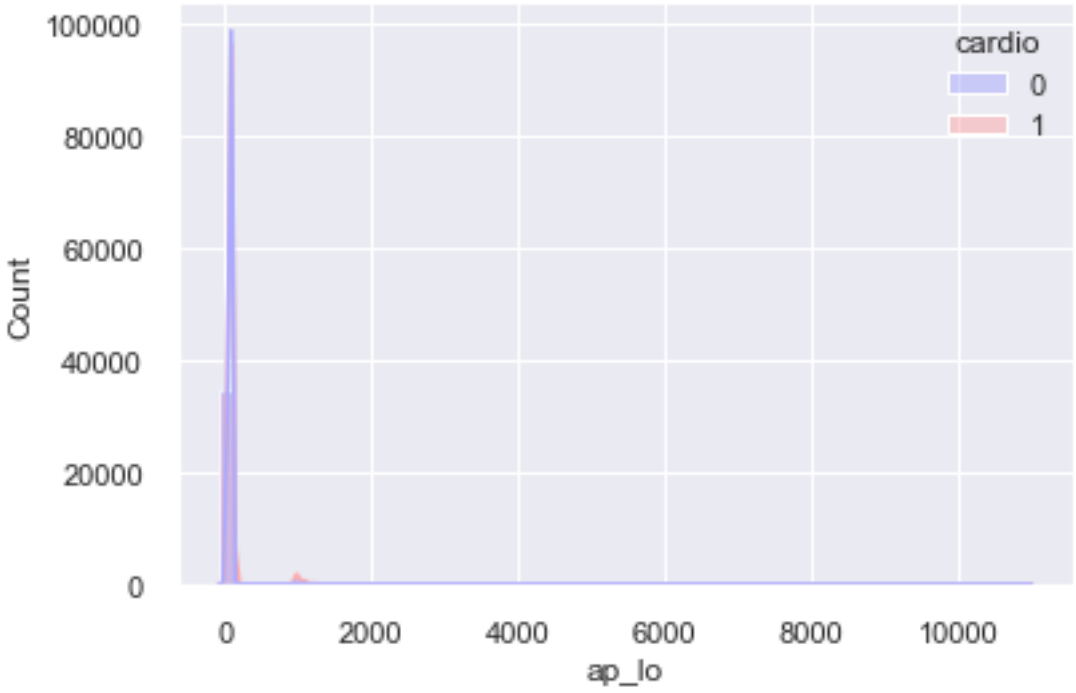
[12]:
```
sns.set()
sns.histplot(data=df,x='weight',kde=True,bins=50,hue='cardio',palette='bwr')
```

[12]: <AxesSubplot: xlabel='weight', ylabel='Count'>



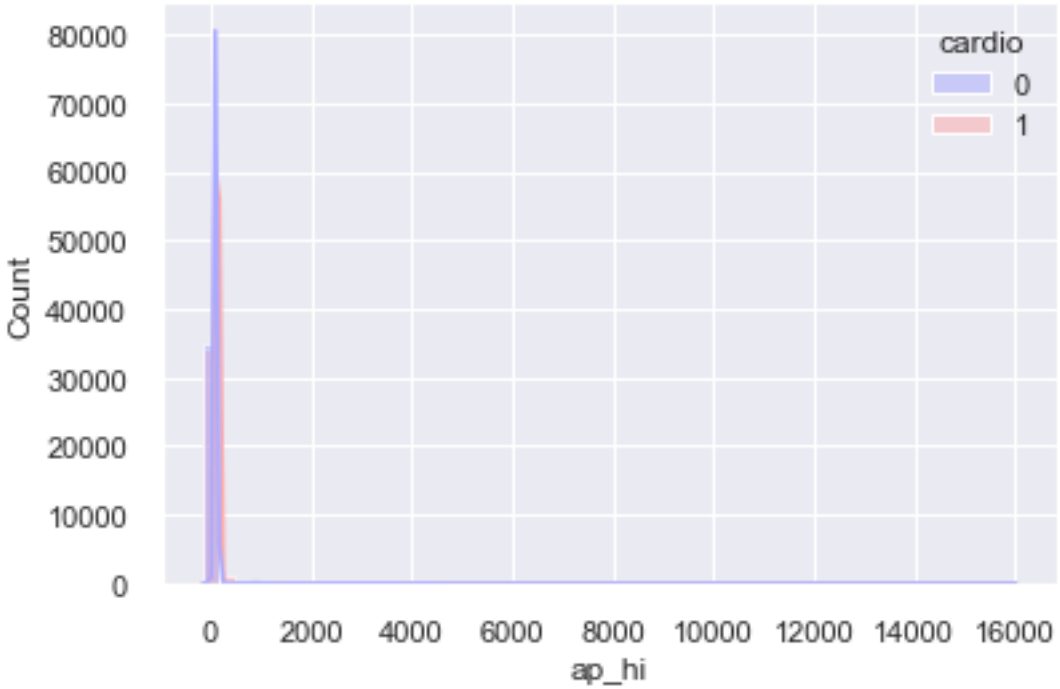
[13]:
```
sns.set()
sns.histplot(data=df,x='ap_lo',kde=True,bins=50,hue='cardio',palette='bwr')
```

[13]: <AxesSubplot: xlabel='ap_lo', ylabel='Count'>



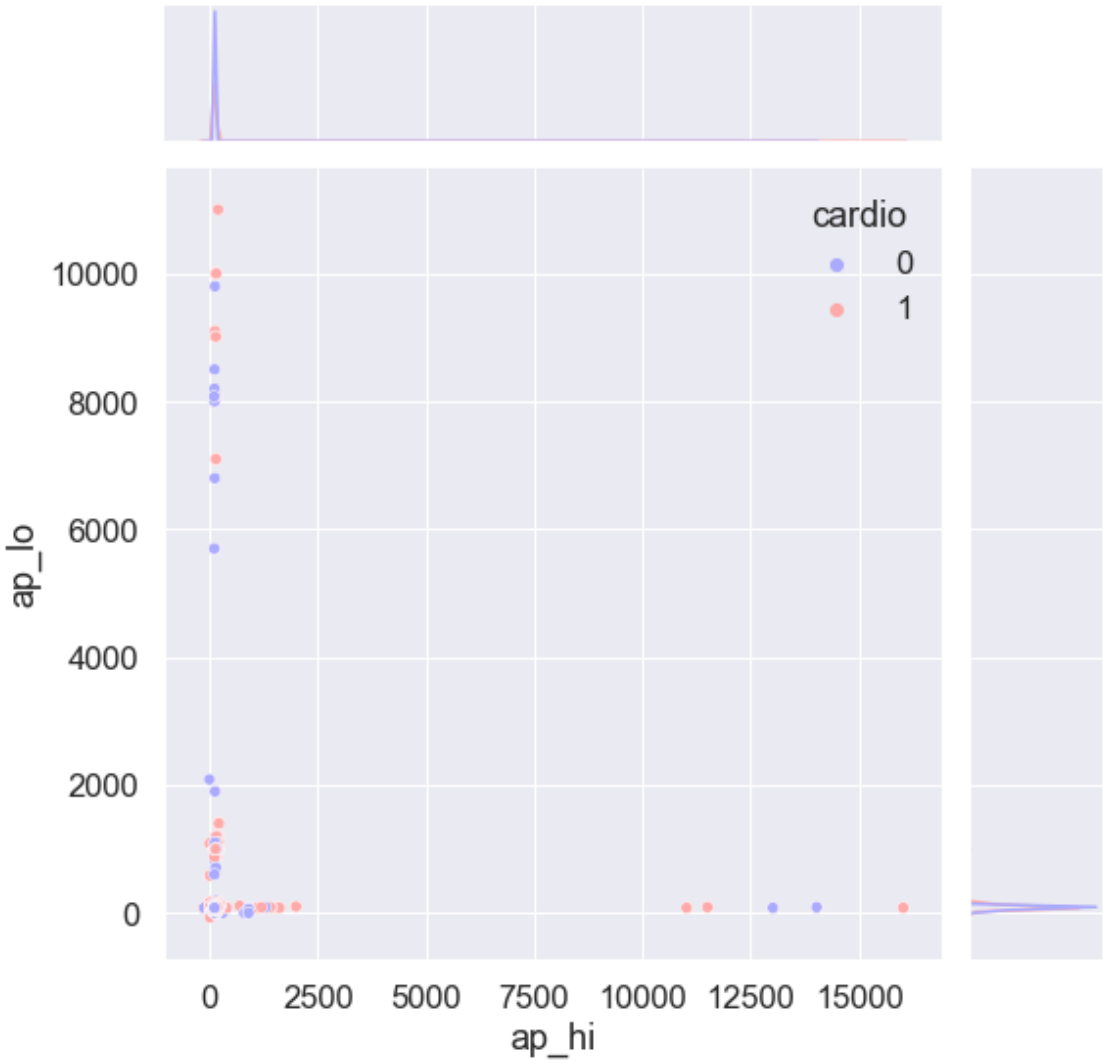
[14]:
```
sns.set()
sns.histplot(data=df,x='ap_hi',kde=True,bins=50,hue='cardio',palette='bwr')
```

[14]: <AxesSubplot: xlabel='ap_hi', ylabel='Count'>



[15]:
```
sns.set()
sns.set_context("notebook", font_scale=1.5)
g = sns.JointGrid(data = df, x = 'ap_hi', y = 'ap_lo', hue = 'cardio', palette␣
↪= 'bwr', height = 8)
g.plot_joint(sns.scatterplot)
g.plot_marginals(sns.kdeplot)
```

[15]: <seaborn.axisgrid.JointGrid at 0x7fb4e02beca0>

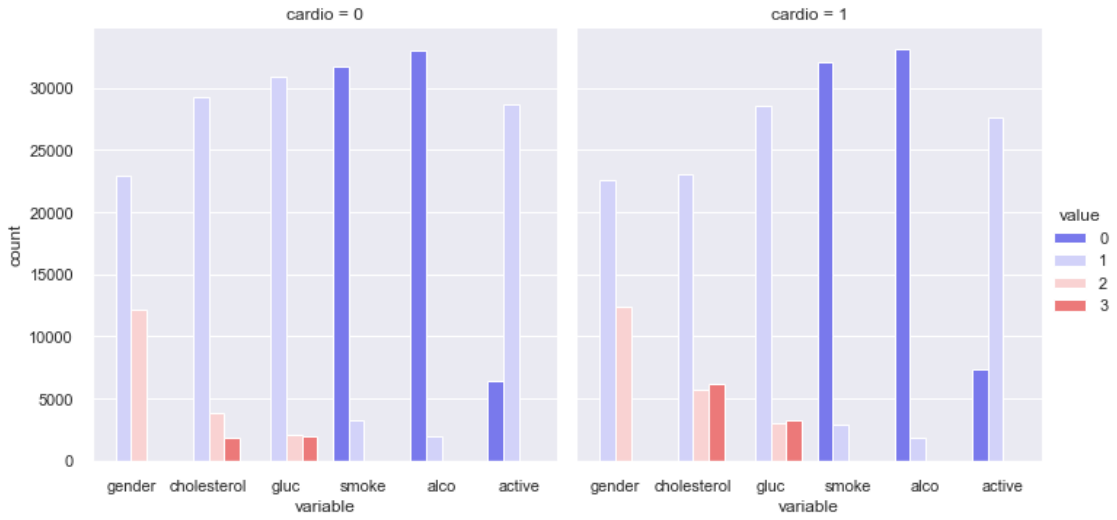


[16]:
```
sns.set()
sns.set_context("notebook", font_scale=1.5)
g = sns.JointGrid(data = df, x = 'height', y = 'weight', hue = 'cardio',␣
 ↪palette = 'bwr', height = 8)
g.plot_joint(sns.scatterplot)
g.plot_marginals(sns.kdeplot)
```

[16]: <seaborn.axisgrid.JointGrid at 0x7fb4df760730>

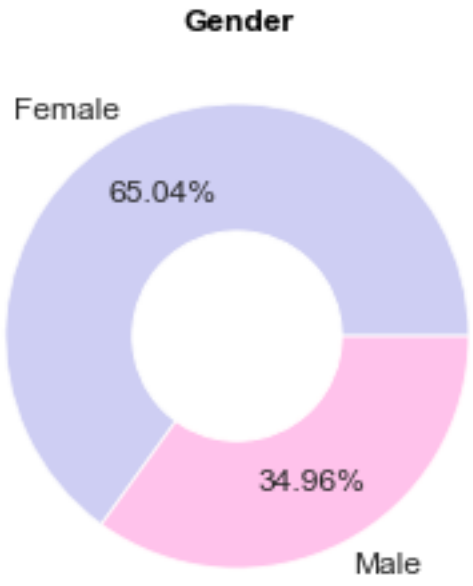
[17]:
```
sns.set()
df_categorical = pd.melt(df, id_vars=['cardio'],␣
↪value_vars=['gender','cholesterol','gluc', 'smoke', 'alco', 'active'])
sns.catplot(x="variable", hue="value", col="cardio",data=df_categorical,␣
↪kind="count",palette='bwr')
```

[17]: <seaborn.axisgrid.FacetGrid at 0x7fb4deb96160>



[18]:
```
labels=['Female', 'Male']
order=df['gender'].value_counts().index

plt.title('Gender', fontweight='heavy',
             fontsize='12', fontfamily='sans-serif', color="black")
colors =['#c2c2f0','#ffb3e6']
plt.pie(df['gender'].value_counts(), labels=labels, pctdistance=0.7,
        autopct='%.2f%%', wedgeprops=dict(alpha=0.8),
        textprops={'fontsize':12},colors = colors)
centre=plt.Circle((0, 0), 0.45, fc='white')
plt.gcf().gca().add_artist(centre)
```

[18]:
```
<matplotlib.patches.Circle at 0x7fb4e07f9b80>
```

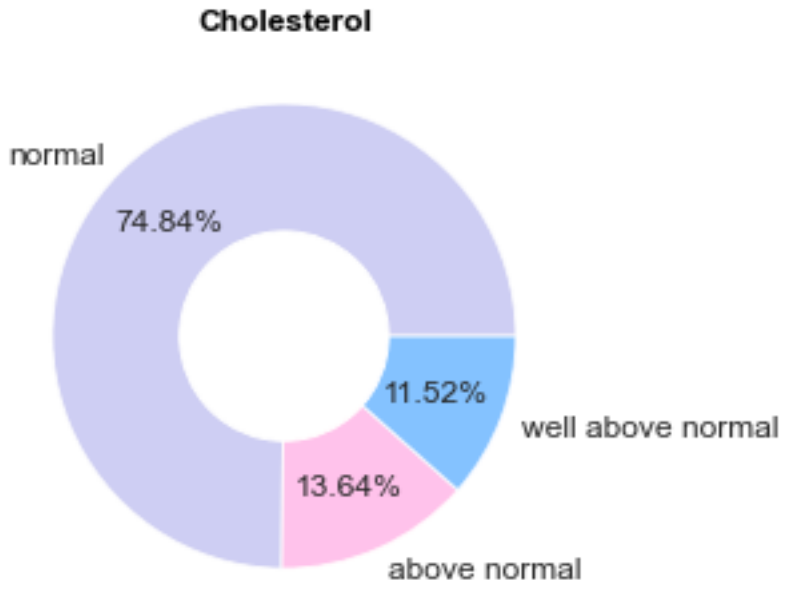


[19]:
```
labels=['normal', 'above normal', 'well above normal']
order=df['cholesterol'].value_counts().index

plt.title('Cholesterol', fontweight='heavy',
             fontsize='12', fontfamily='sans-serif', color="black")
colors =['#c2c2f0','#ffb3e6', '#66b3ff','#c2c2f0']
plt.pie(df['cholesterol'].value_counts(), labels=labels, pctdistance=0.7,
        autopct='%.2f%%', wedgeprops=dict(alpha=0.8),
        textprops={'fontsize':12},colors = colors)
centre=plt.Circle((0, 0), 0.45, fc='white')
plt.gcf().gca().add_artist(centre)
print('*' * 25)
print("percentage of CVD Presence based on Cholesterol:")
print('*' * 25)
print(round(df.groupby('cholesterol')['cardio'].value_counts() / df.
 ↪groupby('cholesterol')['cardio'].count() * 100,2))
print('*' * 25)
print("The total number of each group:")
print('*' * 25)
df.gender.value_counts(dropna=False)
```

```
*************************
percentage of CVD Presence based on Cholesterol:
*************************
cholesterol  cardio
1            0         55.99
```

```
                1          44.01
2               1          60.22
                0          39.78
3               1          76.54
                0          23.46
Name: cardio, dtype: float64
************************
The total number of each group:
************************
```

[19]:
```
1    45530
2    24470
Name: gender, dtype: int64
```

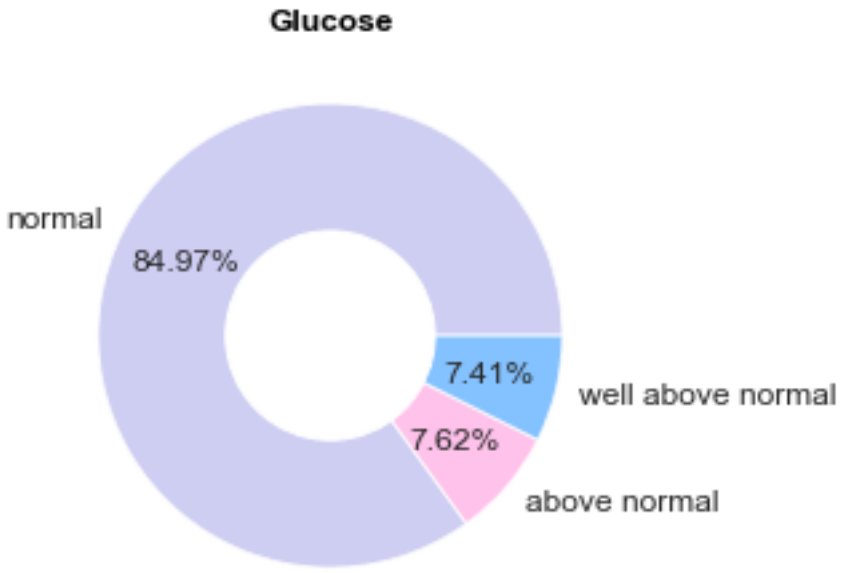


[20]:
```
labels=['normal', 'above normal', 'well above normal']
order=df['gluc'].value_counts().index
plt.title('Glucose', fontweight='heavy',
             fontsize='12', fontfamily='sans-serif', color="black")
colors =['#c2c2f0','#ffb3e6', '#66b3ff','#c2c2f0']
plt.pie(df['gluc'].value_counts(), labels=labels, pctdistance=0.7,
        autopct='%.2f%%', wedgeprops=dict(alpha=0.8),
        textprops={'fontsize':12},colors = colors)
centre=plt.Circle((0, 0), 0.45, fc='white')
plt.gcf().gca().add_artist(centre)
print('*' * 25)
print("percentage of CVD Presence based on glucose:")
```

```
print('*' * 25)
print(round(df.groupby('gluc')['cardio'].value_counts() / df.
 ↪groupby('gluc')['cardio'].count() * 100,2))
print('*' * 25)
print("The total number of each group:")
print('*' * 25)
df.gender.value_counts(dropna=False)
```

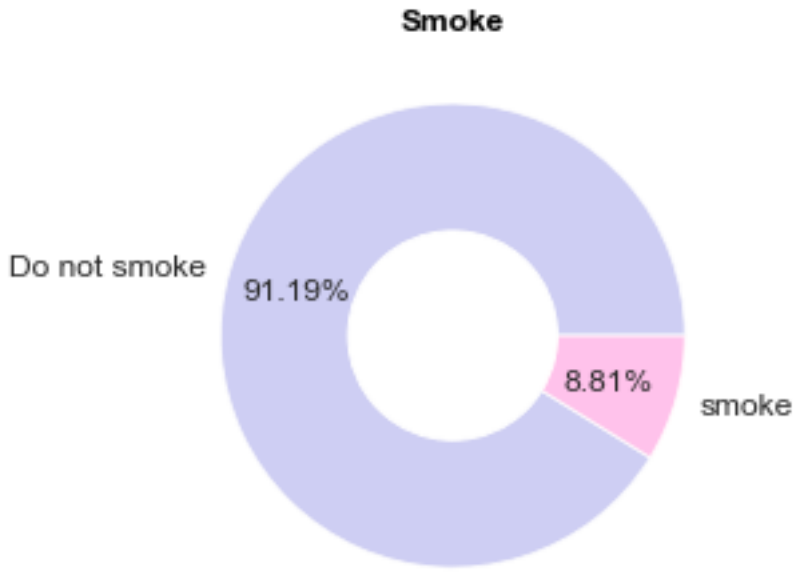
```
*************************
percentage of CVD Presence based on glucose:
*************************
gluc  cardio
1     0         51.94
      1         48.06
2     1         59.31
      0         40.69
3     1         62.20
      0         37.80
Name: cardio, dtype: float64
*************************
The total number of each group:
*************************
```

[20]:
```
1    45530
2    24470
Name: gender, dtype: int64
```



[21]:
```
labels=['Do not smoke', 'smoke']
order=df['smoke'].value_counts().index

plt.title('Smoke', fontweight='heavy',
             fontsize='12', fontfamily='sans-serif', color="black")

colors =['#c2c2f0','#ffb3e6', '#66b3ff','#c2c2f0']
plt.pie(df['smoke'].value_counts(), labels=labels, pctdistance=0.7,
        autopct='%.2f%%', wedgeprops=dict(alpha=0.8),
        textprops={'fontsize':12},colors = colors)
centre=plt.Circle((0, 0), 0.45, fc='white')
plt.gcf().gca().add_artist(centre)
```

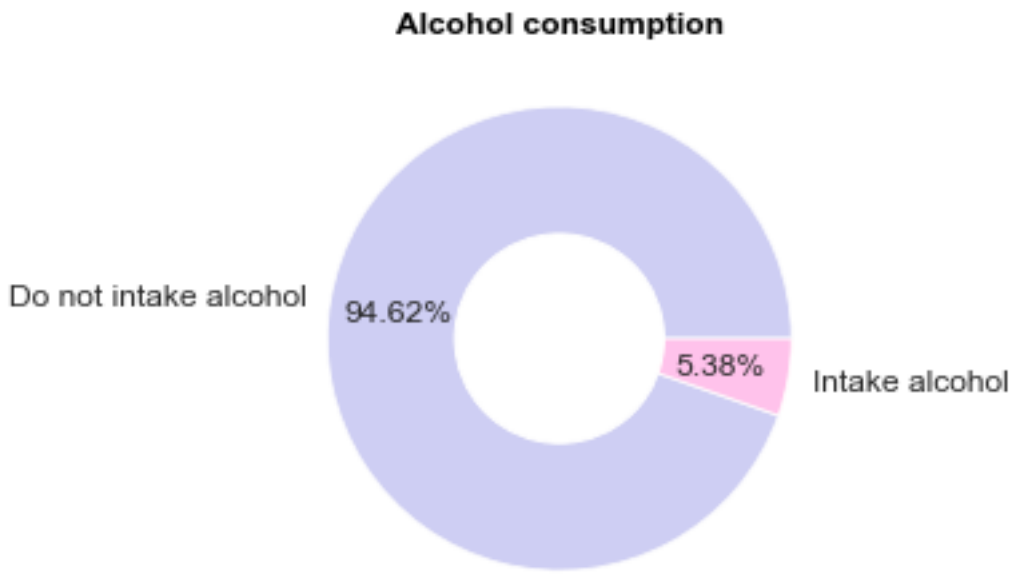
[21]:
```
<matplotlib.patches.Circle at 0x7fb4e005ca30>
```



[22]:
```
labels=['Do not intake alcohol', 'Intake alcohol']
order=df['alco'].value_counts().index

plt.title('Alcohol consumption', fontweight='heavy',
             fontsize='12', fontfamily='sans-serif', color="black")

colors =['#c2c2f0','#ffb3e6', '#66b3ff','#c2c2f0']
plt.pie(df['alco'].value_counts(), labels=labels, pctdistance=0.7,
        autopct='%.2f%%', wedgeprops=dict(alpha=0.8),
        textprops={'fontsize':12},colors = colors)
centre=plt.Circle((0, 0), 0.45, fc='white')
```

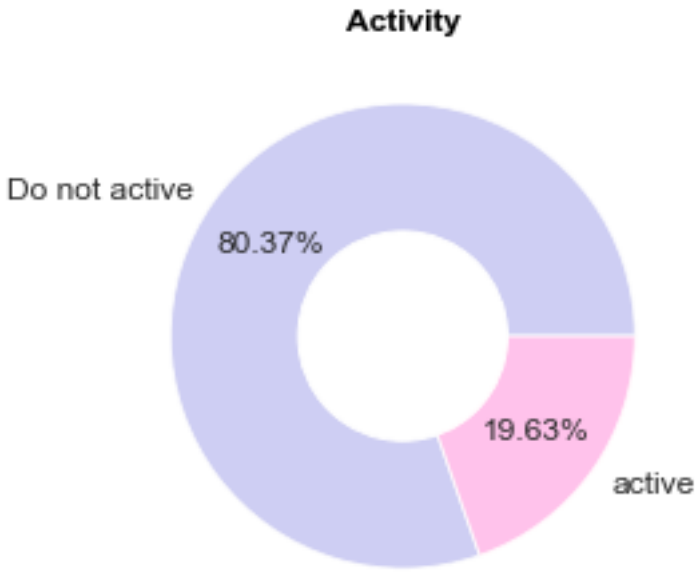
```
plt.gcf().gca().add_artist(centre)
```

[22]: <matplotlib.patches.Circle at 0x7fb4df32ef10>



[23]:
```
labels=['Do not active', 'active']
order=df['active'].value_counts().index

plt.title('Activity', fontweight='heavy',
             fontsize='12', fontfamily='sans-serif', color="black")
colors =['#c2c2f0','#ffb3e6', '#66b3ff','#c2c2f0']
plt.pie(df['active'].value_counts(), labels=labels, pctdistance=0.7,
        autopct='%.2f%%', wedgeprops=dict(alpha=0.8),
        textprops={'fontsize':12},colors = colors)
centre=plt.Circle((0, 0), 0.45, fc='white')
plt.gcf().gca().add_artist(centre)
```

[23]: <matplotlib.patches.Circle at 0x7fb4df314e80>



[24]:
```
df.describe().T
```

[24]:

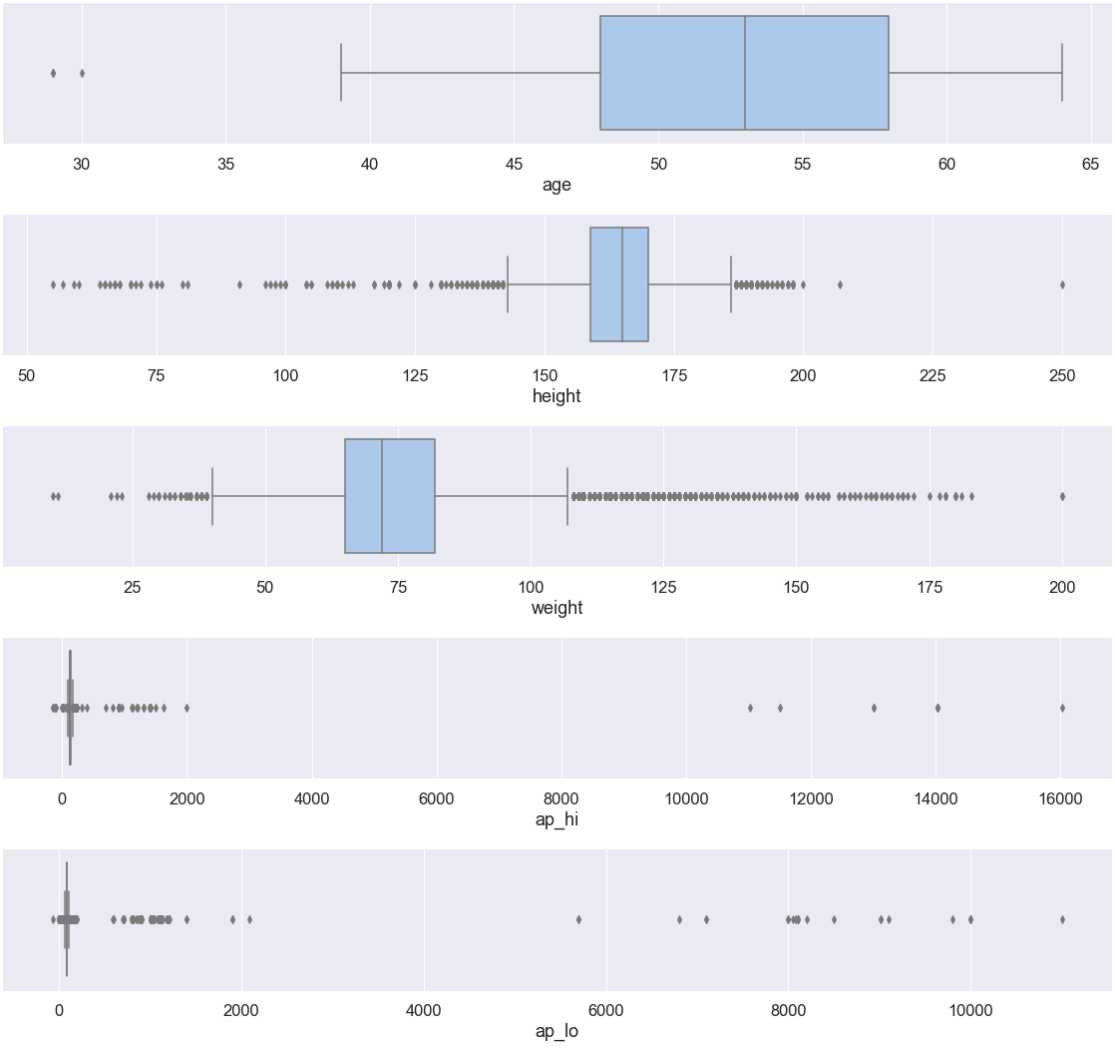
	count	mean	std	min	25%	50%	75%	max
age	70000.0	52.807329	6.762506	29.0	48.0	53.0	58.0	64.0
gender	70000.0	1.349571	0.476838	1.0	1.0	1.0	2.0	2.0
height	70000.0	164.359229	8.210126	55.0	159.0	165.0	170.0	250.0
weight	70000.0	74.205690	14.395757	10.0	65.0	72.0	82.0	200.0
ap_hi	70000.0	128.817286	154.011419	-150.0	120.0	120.0	140.0	16020.0
ap_lo	70000.0	96.630414	188.472530	-70.0	80.0	80.0	90.0	11000.0
cholesterol	70000.0	1.366871	0.680250	1.0	1.0	1.0	2.0	3.0
gluc	70000.0	1.226457	0.572270	1.0	1.0	1.0	1.0	3.0
smoke	70000.0	0.088129	0.283484	0.0	0.0	0.0	0.0	1.0
alco	70000.0	0.053771	0.225568	0.0	0.0	0.0	0.0	
active	70000.0	0.803729	0.397179	0.0	1.0	1.0	1.0	
cardio	70000.0	0.499700	0.500003	0.0	0.0	0.0	1.0	

```
alco            1.0
active          1.0
cardio          1.0
```

[25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70000 entries, 0 to 69999
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   age          70000 non-null  int64
 1   gender       70000 non-null  int64
 2   height       70000 non-null  int64
 3   weight       70000 non-null  float64
 4   ap_hi        70000 non-null  int64
 5   ap_lo        70000 non-null  int64
 6   cholesterol  70000 non-null  int64
 7   gluc         70000 non-null  int64
 8   smoke        70000 non-null  int64
 9   alco         70000 non-null  int64
 10  active       70000 non-null  int64
 11  cardio       70000 non-null  int64
dtypes: float64(1), int64(11)
memory usage: 6.4 MB
```

[26]:
```
sns.set_context("notebook", font_scale=1.5)
f, axs = plt.subplots(5,1,figsize = (16,15))
colors =['#c2c2f0','#ffb3e6']
sns.boxplot(x=df['age'],ax=axs[0],palette='pastel')
sns.boxplot(x=df['height'],ax=axs[1],palette='pastel')
sns.boxplot(x=df['weight'],ax=axs[2],palette='pastel')
sns.boxplot(x=df['ap_hi'],ax=axs[3],palette='pastel')
sns.boxplot(x=df['ap_lo'],ax=axs[4],palette='pastel')

plt.tight_layout()
```



[27]:
```
def outliers(x):
    Q1, Q3 = np.percentile(x, [25,75])
    IQR = Q3 - Q1
    lower_bound = Q1 - (IQR*3)
    upper_bound = Q3 + (IQR*3)

    print(f'Q1:{Q1}, Q3:{Q3}, IQR:{IQR}')
    print(f'Lower Bound:{lower_bound}, Upper Bound:{upper_bound}')

    result = np.where((x> upper_bound) | (x < lower_bound))
    boundary = (lower_bound, upper_bound)

    print(f'Number of outliers: {len(result[0])}')

    return result, boundary
```

[28]:
```
systolic_outlier=list(outliers(df['ap_hi'])[0][0])
df.iloc[systolic_outlier]

diastolic_outlier=list(outliers(df['ap_lo'])[0][0])
df.iloc[diastolic_outlier]

height_outlier=list(outliers(df['height'])[0][0])
df.iloc[height_outlier]

weight_outlier=list(outliers(df['weight'])[0][0])
df.iloc[weight_outlier]
```

```
Q1:120.0, Q3:140.0, IQR:20.0
Lower Bound:60.0, Upper Bound:200.0
Number of outliers: 288
Q1:80.0, Q3:90.0, IQR:10.0
Lower Bound:50.0, Upper Bound:120.0
Number of outliers: 1136
Q1:159.0, Q3:170.0, IQR:11.0
Lower Bound:126.0, Upper Bound:203.0
Number of outliers: 93
Q1:65.0, Q3:82.0, IQR:17.0
Lower Bound:14.0, Upper Bound:133.0
Number of outliers: 171
```

[28]:

	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	\
338	57	1	157	142.0	120	80	1	1	0	
389	45	2	165	169.0	120	80	1	1	0	
435	45	1	186	200.0	130	70	1	1	0	
1197	54	1	171	139.0	140	90	1	1	0	
1270	52	2	175	150.0	180	120	2	1	0	
…	…	…	…	…	…	…	…	…	…	
66832	49	2	180	145.0	130	90	1	1	0	
66928	63	2	170	134.0	140	100	3	3	0	
67121	44	2	178	146.0	140	1100	1	1	1	
67435	50	2	190	147.0	150	90	1	2	0	
69109	52	2	175	155.0	110	100	1	1	0	

	alco	active	cardio
338	0	1	1
389	0	1	0
435	0	0	0
1197	0	1	1
1270	1	1	1
…	…	…	…
66832	0	1	0
66928	0	1	0
67121	0	1	1

```
67435     1       1       1
69109     0       1       1

[171 rows x 12 columns]
```

[29]:
```
outliers= systolic_outlier+ diastolic_outlier+height_outlier+ weight_outlier
outliers= list(set(outliers))
len(outliers)
```

[29]:
```
1624
```

[30]:
```
df=df.drop(index = outliers).reset_index()
df.drop(columns=['index'], inplace=True)
```

[31]:
```
df[df['ap_lo'] > df['ap_hi']].shape[0]
```

[31]:
```
46
```

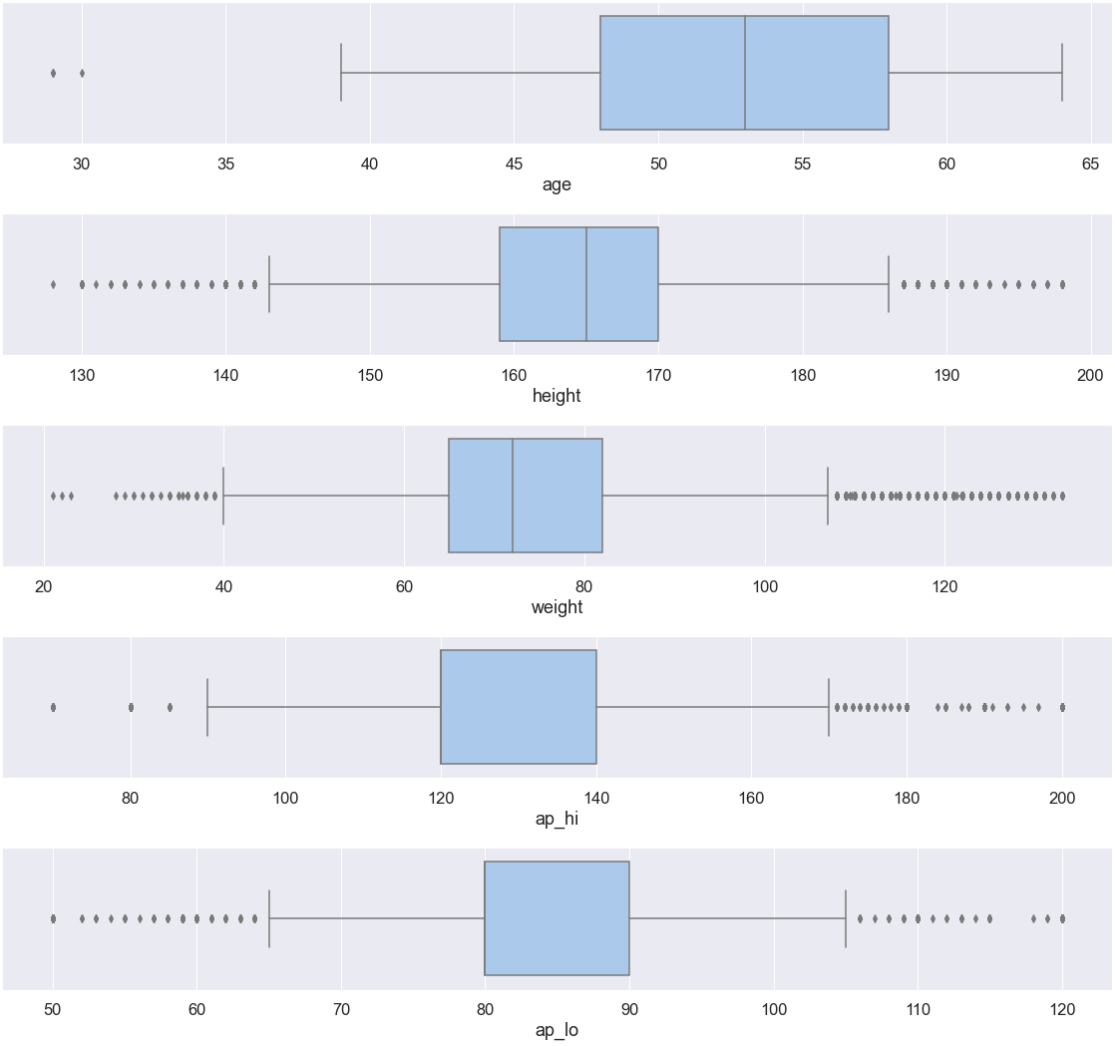
[32]:
```
df.drop(df[df['ap_lo'] > df['ap_hi']].shape[0], inplace=True)
```

[33]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 68375 entries, 0 to 68375
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   age          68375 non-null  int64
 1   gender       68375 non-null  int64
 2   height       68375 non-null  int64
 3   weight       68375 non-null  float64
 4   ap_hi        68375 non-null  int64
 5   ap_lo        68375 non-null  int64
 6   cholesterol  68375 non-null  int64
 7   gluc         68375 non-null  int64
 8   smoke        68375 non-null  int64
 9   alco         68375 non-null  int64
 10  active       68375 non-null  int64
 11  cardio       68375 non-null  int64
dtypes: float64(1), int64(11)
memory usage: 6.8 MB
```

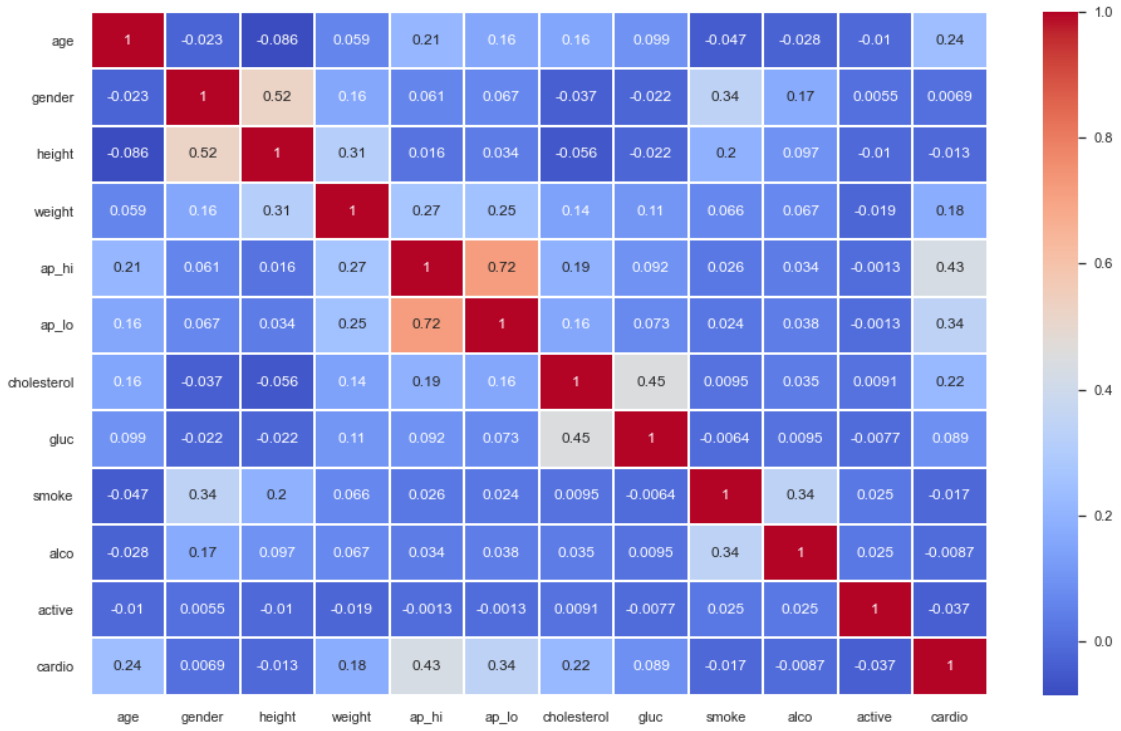
[34]:
```
sns.set_context("notebook", font_scale=1.5)
f, axs = plt.subplots(5,1,figsize = (16,15))
sns.boxplot(x=df['age'],ax=axs[0],palette='pastel')
sns.boxplot(x=df['height'],ax=axs[1],palette='pastel')
sns.boxplot(x=df['weight'],ax=axs[2],palette='pastel')
sns.boxplot(x=df['ap_hi'],ax=axs[3],palette='pastel')
```

```
sns.boxplot(x=df['ap_lo'],ax=axs[4],palette='pastel')

plt.tight_layout()
```



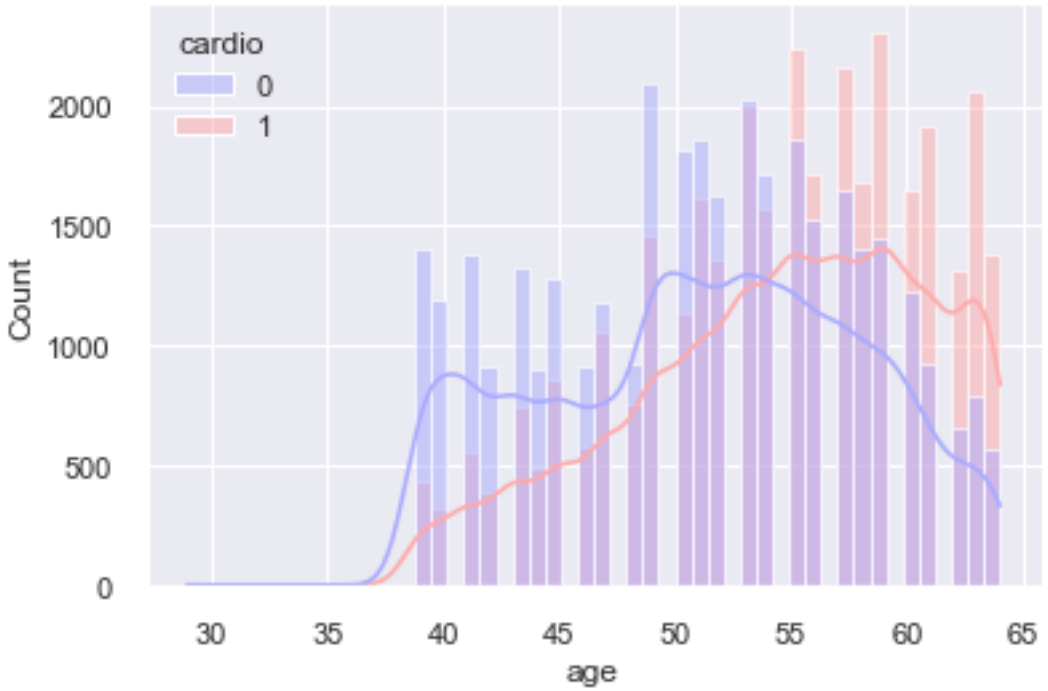
[35]:
```
sns.set_context("notebook", font_scale=1)
plt.subplots(figsize=(16,10))
sns.heatmap(df.corr(),annot=True, linewidth=2,cmap='coolwarm')
```

[35]: <AxesSubplot: >



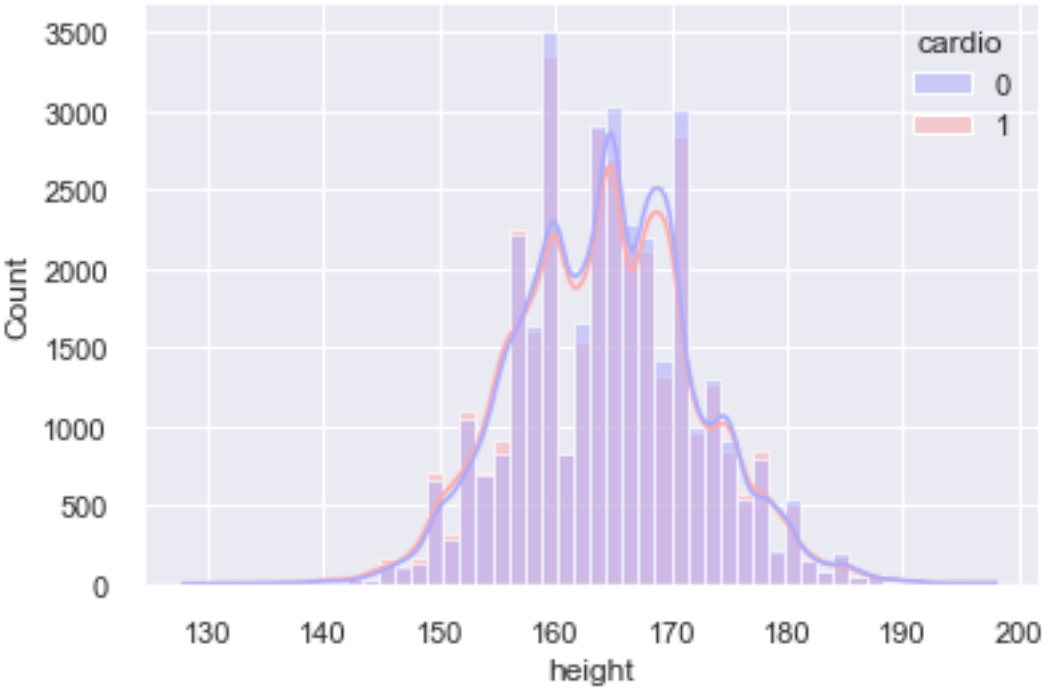
[36]:
```
sns.set()
sns.histplot(data=df,x='age',kde=True,bins=50,hue='cardio',palette='bwr')
```

[36]: <AxesSubplot: xlabel='age', ylabel='Count'>



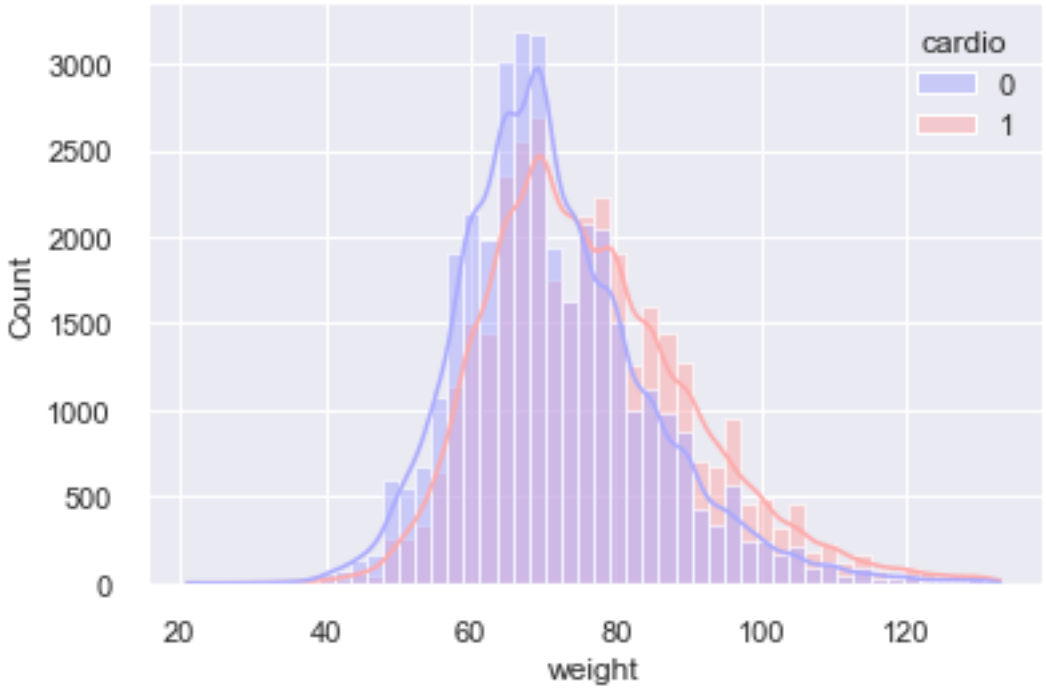
[37]:
```
sns.set()
sns.histplot(data=df,x='height',kde=True,bins=50,hue='cardio',palette='bwr')
```

[37]: <AxesSubplot: xlabel='height', ylabel='Count'>



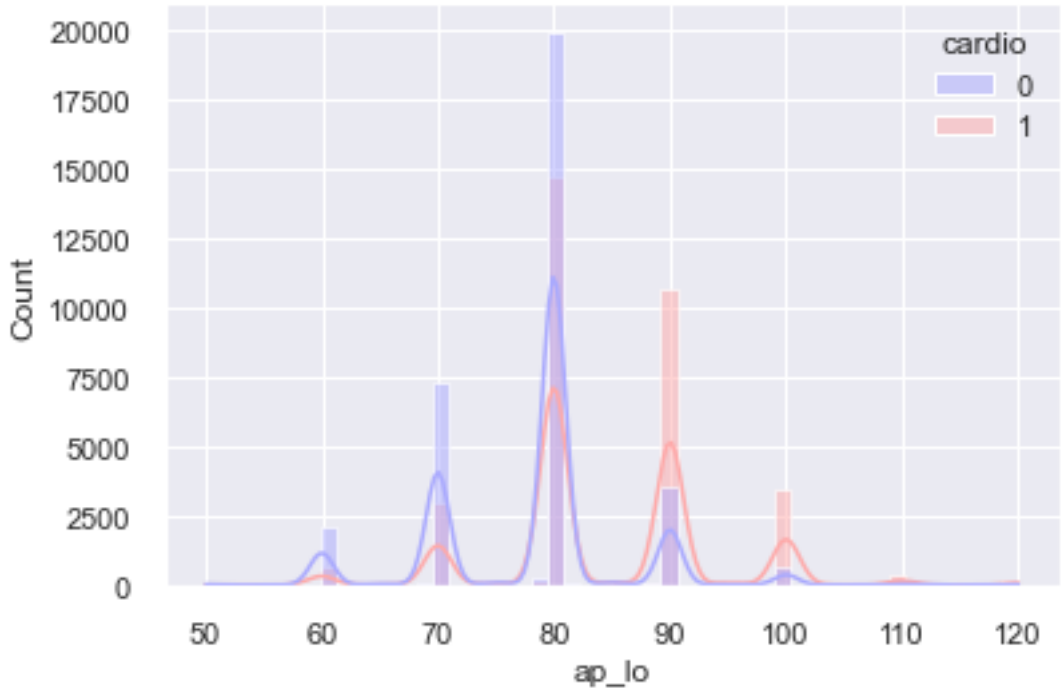
[38]:
```
sns.set()
sns.histplot(data=df,x='weight',kde=True,bins=50,hue='cardio',palette='bwr')
```

[38]: <AxesSubplot: xlabel='weight', ylabel='Count'>



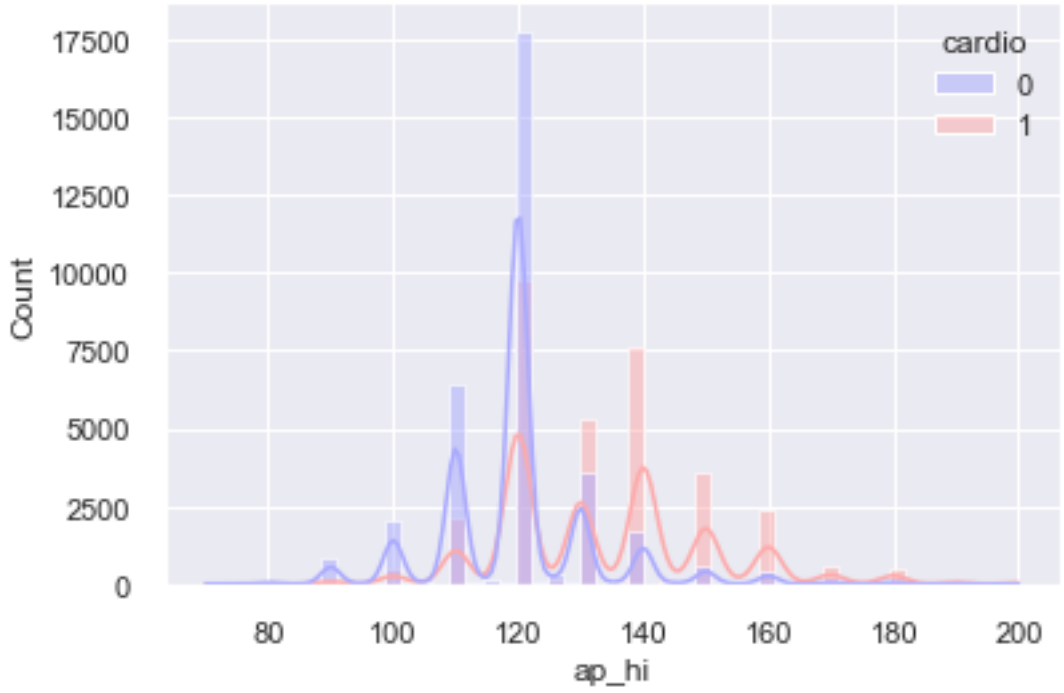
[39]:
```
sns.set()
sns.histplot(data=df,x='ap_lo',kde=True,bins=50,hue='cardio',palette='bwr')
```

[39]: <AxesSubplot: xlabel='ap_lo', ylabel='Count'>



[40]:
```
sns.set()
sns.histplot(data=df,x='ap_hi',kde=True,bins=50,hue='cardio',palette='bwr')
```

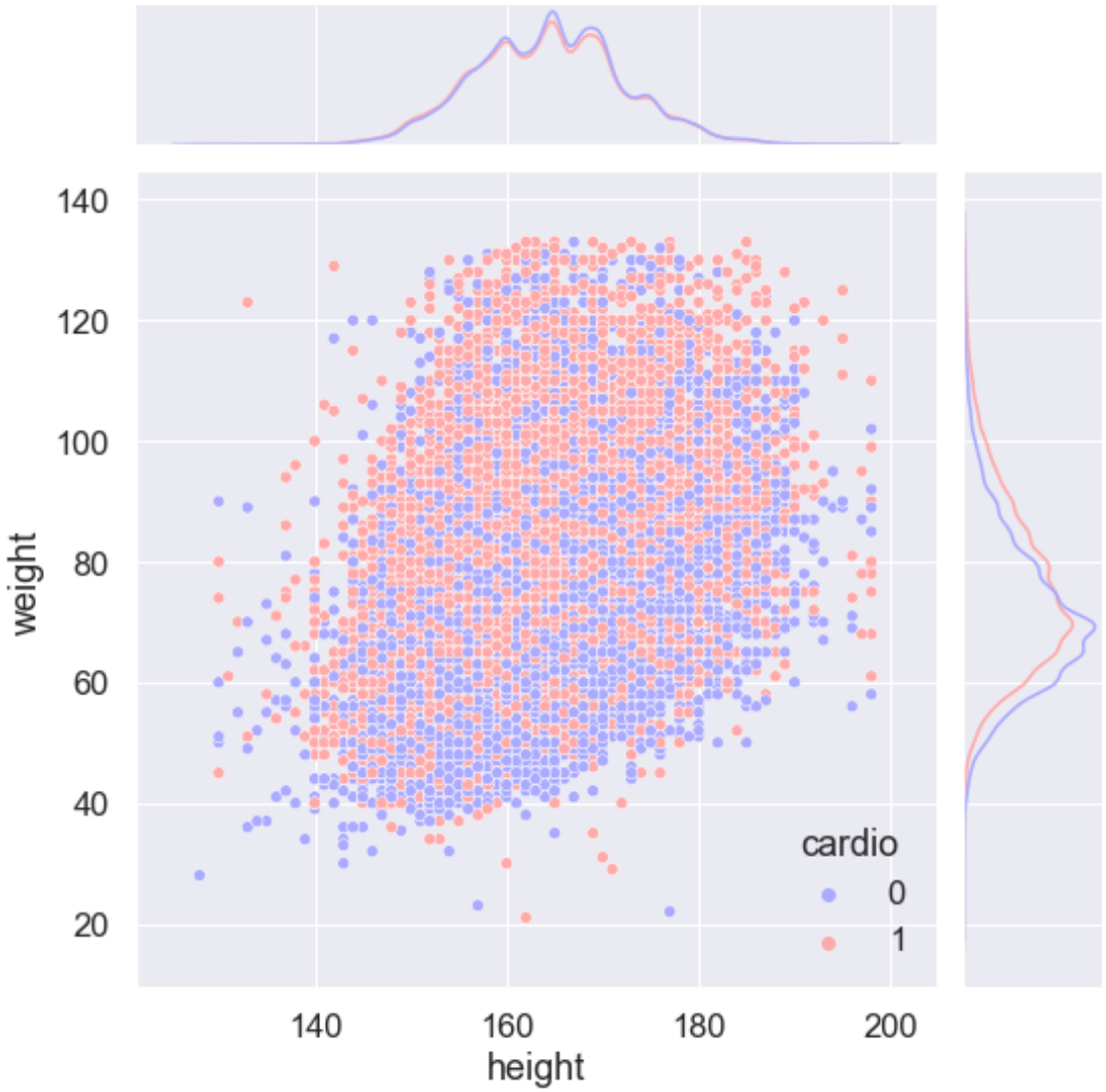
[40]: <AxesSubplot: xlabel='ap_hi', ylabel='Count'>



[41]:
```
#numerical features
numerical = ['age','height' ,'weight', 'ap_hi', 'ap_lo']
#Categorical features
Categorical= ['gender','cholesterol','gluc','smoke','alco','active','cardio']
```

[42]:
```
sns.set()
sns.set_context("notebook", font_scale=1.5)
g = sns.JointGrid(data = df, x = 'height', y = 'weight', hue = 'cardio',␣
 ↪palette = 'bwr', height = 8)
g.plot_joint(sns.scatterplot)
g.plot_marginals(sns.kdeplot)
```

[42]:
```
<seaborn.axisgrid.JointGrid at 0x7fb4e578f040>
```



[43]:
```
sns.set()
sns.set_context("notebook", font_scale=1.5)
g = sns.JointGrid(data = df, x = 'ap_hi', y = 'ap_lo', hue = 'cardio', palette␣
↪= 'bwr', height = 8)
g.plot_joint(sns.scatterplot)
g.plot_marginals(sns.kdeplot)
```

[43]:
```
<seaborn.axisgrid.JointGrid at 0x7fb4e0cec8e0>
```

[44]:
```
df['cardio'].value_counts()
```

[44]:
```
0    34619
1    33756
Name: cardio, dtype: int64
```

[45]:
```
df['height'] = df['height']/100
```

[46]:
```
df
```

[46]:

	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
0	50	2	1.68	62.0	110	80	1	1	0	0	1	0
1	55	1	1.56	85.0	140	90	3	1	0	0	1	1
2	51	1	1.65	64.0	130	70	3	1	0	0	0	1
3	48	2	1.69	82.0	150	100	1	1	0	0	1	1
4	47	1	1.56	56.0	100	60	1	1	0	0	0	0
…	…	…	…	…	…	…	…	…	…	…	…	…
68371	52	2	1.68	76.0	120	80	1	1	1	0	1	0
68372	61	1	1.58	126.0	140	90	2	2	0	0	1	1
68373	52	2	1.83	105.0	180	90	3	1	0	1	0	1
68374	61	1	1.63	72.0	135	80	1	2	0	0	0	1
68375	56	1	1.70	72.0	120	80	2	1	0	0	1	0

[68375 rows x 12 columns]

[47]:
```
X =␣
 ↪df[['age','gender','height','weight','ap_hi','ap_lo','cholesterol','gluc','smoke','alco','a
y= df[['cardio']]
```

[48]:
```
X.shape , y.shape
```

[48]:
```
((68375, 11), (68375, 1))
```

[49]:
```
from sklearn.feature_selection import SelectKBest, f_classif
```

[50]:
```
bestfeatures= SelectKBest(score_func=chi2,k=11)
fit= bestfeatures.fit(X,y)
```

[51]:
```
scores = pd.DataFrame(fit.scores_)
columns = pd.DataFrame(X.columns)
```

[52]:
```
dataScore2 = pd.concat([columns,scores],axis=1)
dataScore2.columns = ['Feature','Score']
dataScore2
```

[52]:

	Feature	Score
0	age	3414.668152
1	gender	0.542439
2	height	0.043474
3	weight	5752.759421
4	ap_hi	27106.059803
5	ap_lo	8461.363037
6	cholesterol	1128.981176
7	gluc	144.203457
8	smoke	17.510894
9	alco	4.924292
10	active	18.745411

[53]:
```
print(dataScore2.nlargest(11,'Score'))
```

```
        Feature         Score
4         ap_hi  27106.059803
5         ap_lo   8461.363037
3        weight   5752.759421
0           age   3414.668152
6   cholesterol   1128.981176
7          gluc    144.203457
10       active     18.745411
8         smoke     17.510894
9          alco      4.924292
1        gender      0.542439
2        height      0.043474
```

[54]:
```
model = ExtraTreesClassifier()
model.fit(X,y)
```

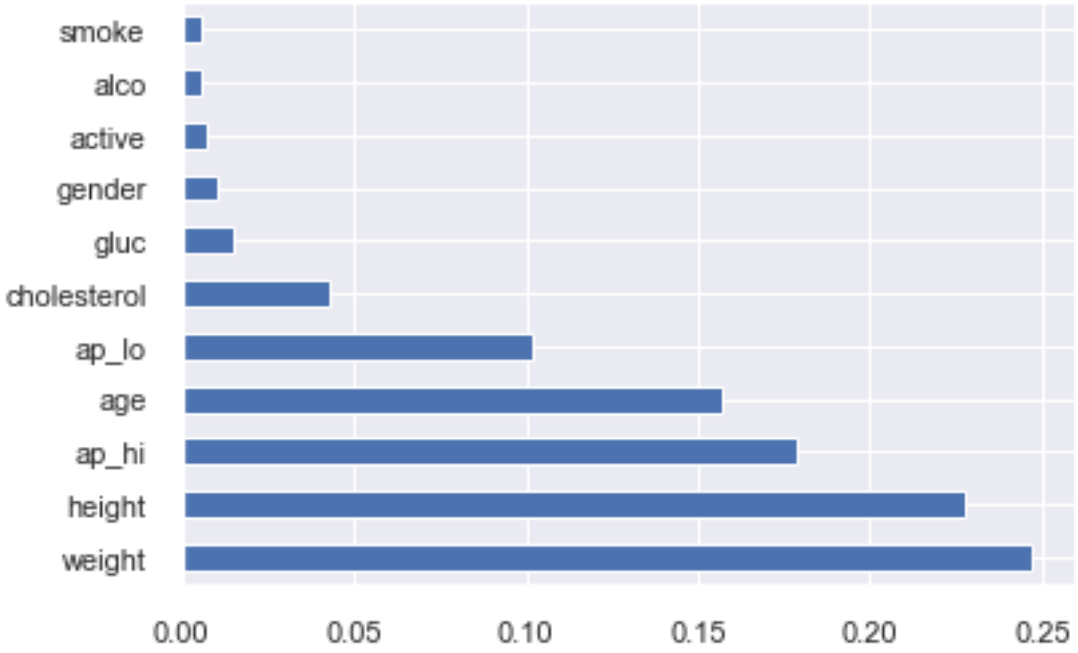
[54]:
```
ExtraTreesClassifier()
```

[55]:
```
print(model.feature_importances_)
```

```
[0.15720876 0.01066167 0.22727691 0.2473066  0.17859524 0.1020839
 0.0428237  0.01481337 0.00591175 0.00600639 0.00731171]
```

[56]:
```
sns.set()
feat_importances_= pd.Series(model.feature_importances_, index=X.columns)
feat_importances_.nlargest(13).plot(kind='barh')
```

[56]: <AxesSubplot: >



[57]:
```
from sklearn.preprocessing import StandardScaler
scale= ['age', 'weight', 'ap_hi',␣
 ↪'ap_lo','cholesterol','height',"gender","gluc","smoke","alco","active"]
scaler = StandardScaler()
X[scale] = scaler.fit_transform(X[scale])
X.head()
```

[57]:

	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active
0	-0.413353	1.367790	0.456100	-0.861402	-1.002701	-0.136733	-0.536439	-0.394167	-0.310262	-0.237333	0.494693
1	0.325673	-0.731107	-1.074701	0.799252	0.818033	0.936324	2.411799	-0.394167	-0.310262	-0.237333	0.494693
2	-0.265548	-0.731107	0.073400	-0.716997	0.211122	-1.209791	2.411799	-0.394167	-0.310262	-0.237333	-2.021457
3	-0.708963	1.367790	0.583667	0.582645	1.424945	2.009382	-0.536439	-0.394167	-0.310262	-0.237333	0.494693
4	-0.856768	-0.731107	-1.074701	-1.294616	-1.609613	-2.282848	-0.536439	-0.394167	-0.310262	-0.237333	-2.021457

[58]:
```
mmscaler = MinMaxScaler()
X[scale] = mmscaler.fit_transform(X[scale])
```

```
X.head()
```

[58]:
```
        age  gender    height    weight     ap_hi     ap_lo  cholesterol  \
0  0.600000     1.0  0.571429  0.366071  0.307692  0.428571          0.0
1  0.742857     0.0  0.400000  0.571429  0.538462  0.571429          1.0
2  0.628571     0.0  0.528571  0.383929  0.461538  0.285714          1.0
3  0.542857     1.0  0.585714  0.544643  0.615385  0.714286          0.0
4  0.514286     0.0  0.400000  0.312500  0.230769  0.142857          0.0

   gluc  smoke  alco  active
0   0.0    0.0   0.0     1.0
1   0.0    0.0   0.0     1.0
2   0.0    0.0   0.0     0.0
3   0.0    0.0   0.0     1.0
4   0.0    0.0   0.0     0.0
```

[59]:
```
X.drop(["gender","gluc","smoke","alco","active"],axis=1,inplace=True)
```

[60]:
```
X
```

[60]:
```
            age    height    weight     ap_hi     ap_lo  cholesterol
0      0.600000  0.571429  0.366071  0.307692  0.428571          0.0
1      0.742857  0.400000  0.571429  0.538462  0.571429          1.0
2      0.628571  0.528571  0.383929  0.461538  0.285714          1.0
3      0.542857  0.585714  0.544643  0.615385  0.714286          0.0
4      0.514286  0.400000  0.312500  0.230769  0.142857          0.0
…           …         …         …         …         …            …
68371  0.657143  0.571429  0.491071  0.384615  0.428571          0.0
68372  0.914286  0.428571  0.937500  0.538462  0.571429          0.5
68373  0.657143  0.785714  0.750000  0.846154  0.571429          1.0
68374  0.914286  0.500000  0.455357  0.500000  0.428571          0.0
68375  0.771429  0.600000  0.455357  0.384615  0.428571          0.5

[68375 rows x 6 columns]
```

[61]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42, shuffle = True)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.
↪2, random_state= 42, shuffle = True)
```

[62]:
```
print("X_train shape: {}".format(X_train.shape))
print("X_test shape: {}".format(X_test.shape))
print("y_train shape: {}".format(y_train.shape))
print("y_test shape: {}".format(y_test.shape))
print("X_val shape: {}".format(X_val.shape))
print("y val shape: {}".format(y_val.shape))
```

```
X_train shape: (43760, 6)
```

```
X_test shape: (13675, 6)
y_train shape: (43760, 1)
y_test shape: (13675, 1)
X_val shape: (10940, 6)
y val shape: (10940, 1)
```

[63]:
```
X = df.drop(['cardio'], axis=1)
y = df['cardio']

def predict_class(model, X_val):
    y_pred = model.predict(X_val)
    y_prob = model.predict_proba(X_val)[:,1]
    return y_pred, y_prob

def show_summary(model, X_val, y_val):
    y_pred, y_prob = predict_class(model, X_val)
    model.report = classification_report(y_val, y_pred,␣
 ↪target_names=["no","yes"])
    print(model.report)


def plot_conf_ROC(model, X_val, y_val):
    y_pred, y_prob = predict_class(model, X_val)
    accuracy = accuracy_score(y_val, y_pred)
    precision= precision_score(y_val, y_pred)
    recall = recall_score(y_val, y_pred)
    cm=confusion_matrix(y_val,y_pred)
    conf=pd.DataFrame(data=cm,columns=['Predicted:NO Cvd','Predicted:
 ↪CVD'],index=['Actual:No CVD','Actual:CVD'])
    specificity= cm[0,0]/(cm[0,0]+cm[0,1])
    print('Specificity : ', specificity)

    sensitivity= cm[1,1]/(cm[1,0]+cm[1,1])
    print('Sensitivity : ', sensitivity)


    plt.figure(figsize=(14,7))
    plt.subplot(121)
    sns.heatmap(conf, annot=True, fmt=".0f", square = True,cmap='coolwarm')
    plt.ylabel('Actual label')
    plt.xlabel('Predicted label')
    all_sample_title = 'Recall Score: {0:.2f}%'.format(100*recall)
    plt.title(all_sample_title, size = 15)

    fpr, tpr, thresholds = roc_curve(y_val, y_prob)
    auc_score = roc_auc_score(y_val, y_prob)
```

```
    plt.subplot(122)
    plt.plot(fpr, tpr, color='orange')
    plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--')
    plt.xlim([-0.05, 1.05])
    plt.ylim([-0.05, 1.05])
    plt.title('ROC curve (AUC = {0:.2f}%)'.format(100*auc_score))
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate (Sensitivity)')
    plt.show()
```

[64]:
```
import statsmodels.formula.api as smf
logit_model=smf.logit("cardio~age+height+weight+ap_hi+ap_lo+cholesterol",␣
 ↪data=df).fit()
```

```
Optimization terminated successfully.
         Current function value: 0.561886
         Iterations 6
```

[65]:
```
print(logit_model.summary2())
```

```
                          Results: Logit
=================================================================
Model:              Logit            Pseudo R-squared: 0.189
Dependent Variable: cardio           AIC:              76851.8851
Date:               2022-11-15 10:20 BIC:              76915.8145
No. Observations:   68375            Log-Likelihood:   -38419.
Df Model:           6                LL-Null:          -47388.
Df Residuals:       68368            LLR p-value:      0.0000
Converged:          1.0000           Scale:            1.0000
No. Iterations:     6.0000
-----------------------------------------------------------------
               Coef.   Std.Err.     z     P>|z|   [0.025   0.975]
-----------------------------------------------------------------
Intercept    -11.1747   0.2261 -49.4252 0.0000 -11.6178 -10.7315
age            0.0515   0.0014  38.0825 0.0000   0.0489   0.0542
height        -0.6032   0.1199  -5.0307 0.0000  -0.8382  -0.3682
weight         0.0110   0.0007  15.4250 0.0000   0.0096   0.0124
ap_hi          0.0554   0.0009  60.1780 0.0000   0.0536   0.0572
ap_lo          0.0128   0.0015   8.7907 0.0000   0.0100   0.0157
cholesterol    0.4448   0.0139  31.9915 0.0000   0.4175   0.4720
=================================================================
```

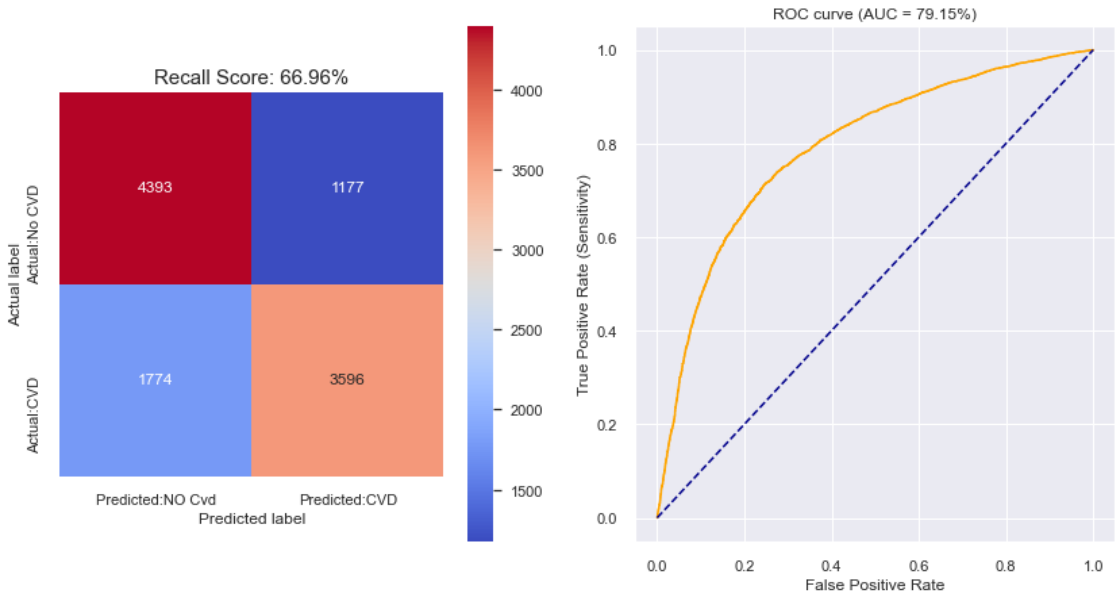
[66]:
```
params = np.exp(logit_model.params)
conf = np.exp(logit_model.conf_int())
conf['OR'] = params
pvalue=round(logit_model.pvalues,3)
```

```
conf['pvalue']=pvalue
conf.columns = ['CI 95%(2.5%)', 'CI 95%(97.5%)', 'Odds Ratio','pvalue']
print ((conf))
```

```
             CI 95%(2.5%)  CI 95%(97.5%)  Odds Ratio  pvalue
Intercept        0.000009       0.000022    0.000014     0.0
age              1.050068       1.055650    1.052856     0.0
height           0.432477       0.691980    0.547051     0.0
weight           1.009635       1.012457    1.011045     0.0
ap_hi            1.055023       1.058834    1.056927     0.0
ap_lo            1.010021       1.015817    1.012915     0.0
cholesterol      1.518185       1.603218    1.560122     0.0
```

[66]:
```
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
show_summary(logreg, X_val, y_val)
plot_conf_ROC(logreg, X_val, y_val)
```

```
              precision    recall  f1-score   support

          no       0.71      0.79      0.75      5570
         yes       0.75      0.67      0.71      5370

    accuracy                           0.73     10940
   macro avg       0.73      0.73      0.73     10940
weighted avg       0.73      0.73      0.73     10940

Specificity :  0.788689407540395
Sensitivity :  0.6696461824953445
```



[67]:
```
y_pred_prob=logreg.predict_proba(X_val)[:,:]
y_pred_prob_df=pd.DataFrame(data=y_pred_prob, columns=['Prob of no heart␣
↪disease (0)','Prob of Heart Disease (1)'])
y_pred_prob_df.head()
```

[67]:

	Prob of no heart disease (0)	Prob of Heart Disease (1)
0	0.081625	0.918375
1	0.662142	0.337858
2	0.143782	0.856218
3	0.306195	0.693805
4	0.646319	0.353681

[67]:
```
X_val
```

[67]:

	age	height	weight	ap_hi	ap_lo	cholesterol
59536	0.971429	0.557143	0.410714	0.692308	0.714286	0.0
13065	0.914286	0.442857	0.544643	0.307692	0.285714	0.0
44927	0.828571	0.328571	0.294643	0.538462	0.571429	1.0
15393	0.714286	0.600000	0.526786	0.538462	0.571429	0.0
22946	0.628571	0.528571	0.482143	0.384615	0.428571	0.0
…	…	…	…	…	…	…
59667	0.628571	0.457143	0.517857	0.384615	0.428571	0.5
52368	0.685714	0.500000	0.383929	0.384615	0.428571	0.0
48256	0.914286	0.385714	0.392857	0.307692	0.285714	0.0
58397	0.657143	0.442857	0.446429	0.307692	0.285714	0.0
30470	0.342857	0.514286	0.401786	0.615385	0.571429	0.0

[10940 rows x 6 columns]

[139]:
```
y_pred = logreg.predict(X_test)
accuracy_score(y_test, y_pred)
```

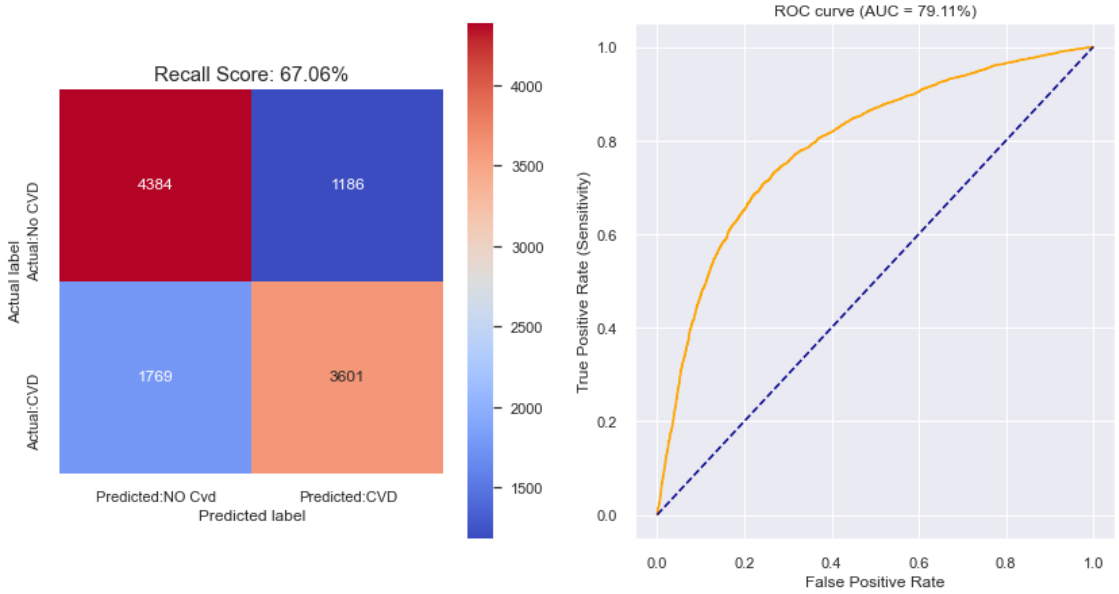
[139]:
```
0.7225594149908592
```

[68]:
```
#penalty - Specify the norm of the penalty.
#C - Inverse of regularization strength; smaller values specify stronger␣
↪regularization.
```

[69]:
```
parameters = {'penalty': ('l1', 'l2','elasticnet', 'none'),
    'C': (1.0, 0.75, 0.5, 0.25, 0.1, 1.25, 1.5)}

logreg= LogisticRegression()
logreg_optimized = GridSearchCV(logreg, param_grid=parameters, n_jobs=-1,␣
↪verbose=True, cv=5, scoring = 'recall')
logreg_optimized.fit(X_train, y_train)
print("Hyperparameters :", logreg_optimized.best_params_)
```

```
show_summary(logreg_optimized, X_val, y_val)
plot_conf_ROC(logreg_optimized, X_val, y_val)
```

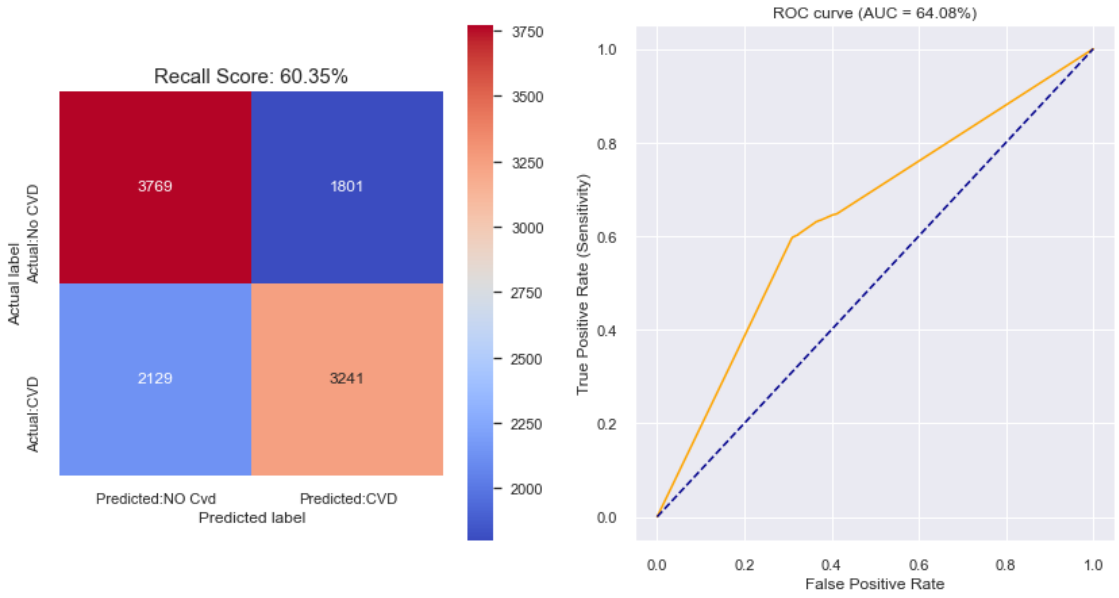
```
Fitting 5 folds for each of 28 candidates, totalling 140 fits
Hyperparameters : {'C': 0.1, 'penalty': 'l2'}
              precision    recall  f1-score   support

          no       0.71      0.79      0.75      5570
         yes       0.75      0.67      0.71      5370

    accuracy                           0.73     10940
   macro avg       0.73      0.73      0.73     10940
weighted avg       0.73      0.73      0.73     10940

Specificity :  0.7870736086175942
Sensitivity :  0.6705772811918064
```



[70]:
```
decision_tree = DecisionTreeClassifier()
decision_tree = decision_tree.fit(X_train, y_train)
show_summary(decision_tree, X_val, y_val)
plot_conf_ROC(decision_tree, X_val, y_val)
```

```
              precision    recall  f1-score   support

          no       0.64      0.68      0.66      5570
         yes       0.64      0.60      0.62      5370

    accuracy                           0.64     10940
```

```
   macro avg       0.64      0.64      0.64     10940
weighted avg       0.64      0.64      0.64     10940

Specificity :  0.6766606822262119
Sensitivity :  0.6035381750465549
```



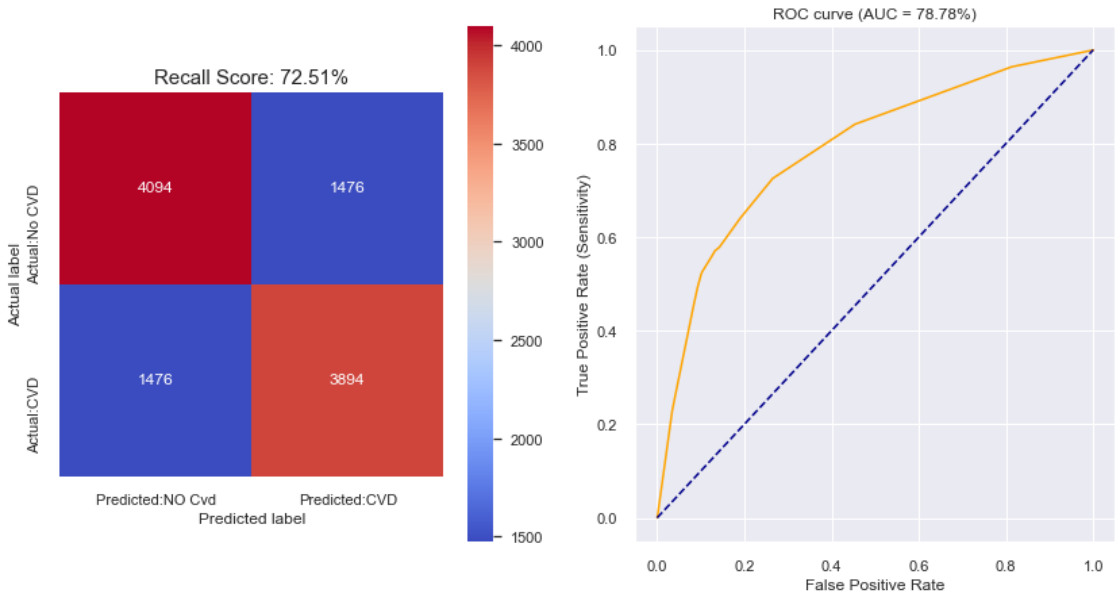
[348]:
```
param_grid = {'min_samples_leaf': range(1,5),
              'min_samples_split': range(1,10),
              'max_depth': range(1,10)}

grid_search = GridSearchCV(DecisionTreeClassifier(criterion␣
 ↪="entropy",random_state = 42),param_grid=param_grid,cv=5,verbose=True,␣
 ↪n_jobs=-1, scoring="accuracy")
grid_search = grid_search.fit(X_train, y_train)
print("Hyperparameters :", grid_search.best_params_)
```

```
Fitting 5 folds for each of 324 candidates, totalling 1620 fits
Hyperparameters : {'max_depth': 4, 'min_samples_leaf': 1, 'min_samples_split':
2}
```

[71]:
```
tree_tuned =DecisionTreeClassifier(max_depth=4,min_samples_leaf=1,␣
 ↪min_samples_split=2,random_state = 42)
tree_tuned = tree_tuned.fit(X_train, y_train)
show_summary(tree_tuned, X_val, y_val)
plot_conf_ROC(tree_tuned, X_val, y_val)
```

```
              precision    recall  f1-score   support
```

```
              precision    recall  f1-score   support

          no       0.74      0.74      0.74      5570
         yes       0.73      0.73      0.73      5370

    accuracy                           0.73     10940
   macro avg       0.73      0.73      0.73     10940
weighted avg       0.73      0.73      0.73     10940

Specificity :  0.7350089766606822
Sensitivity :  0.7251396648044692
```



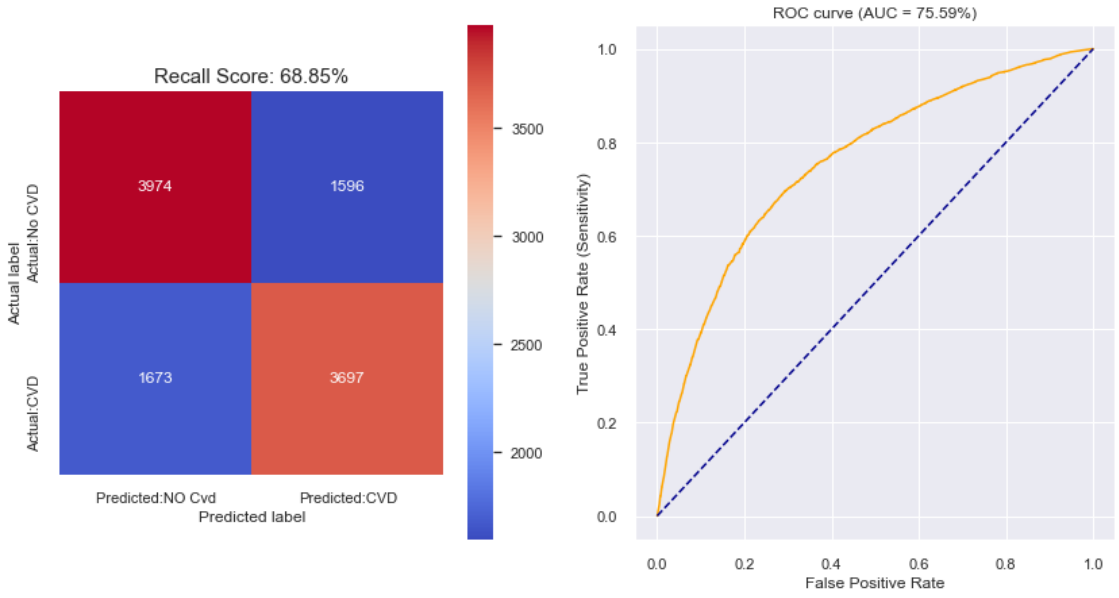
[72]:
```
y_pred = tree_tuned.predict(X_test)
accuracy_score(y_test, y_pred)
```

[72]:
```
0.7206581352833638
```

[86]:
```
forest=RandomForestClassifier()
forest.fit(X_train, y_train)
show_summary(forest, X_val, y_val)
plot_conf_ROC(forest, X_val, y_val)
```

```
              precision    recall  f1-score   support

          no       0.70      0.71      0.71      5570
         yes       0.70      0.69      0.69      5370

    accuracy                           0.70     10940
   macro avg       0.70      0.70      0.70     10940
weighted avg       0.70      0.70      0.70     10940
```

```
Specificity :  0.7134649910233393
Sensitivity :  0.6884543761638734
```



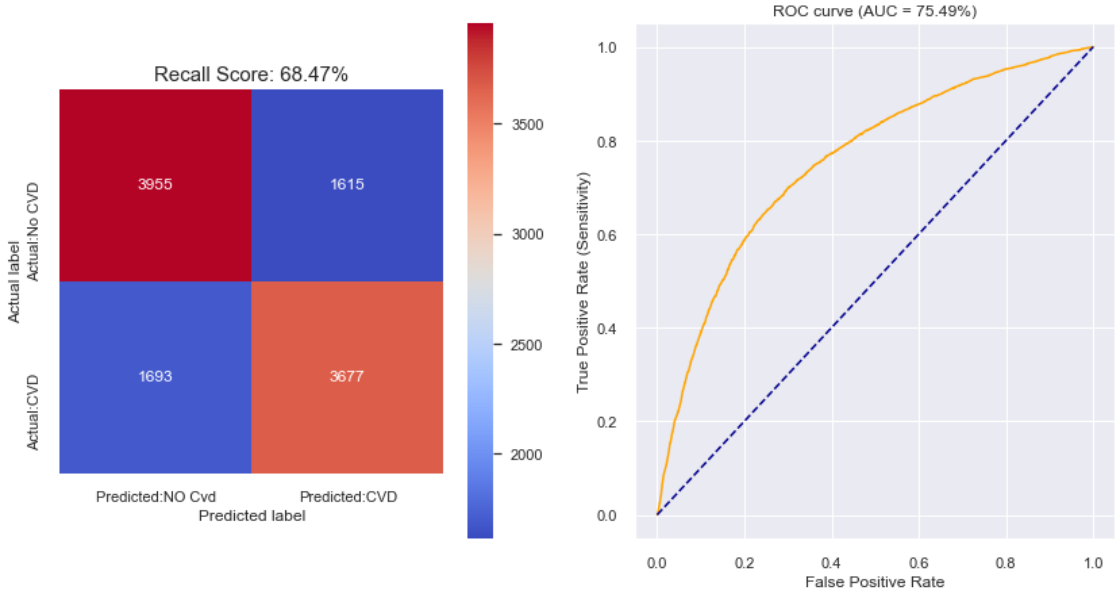
[485]:
```
param_grid = {'min_samples_leaf': range(1,5),
              'min_samples_split': range(1,10),
              'max_depth': range(1,45)}
forest_tuned=RandomForestClassifier(criterion = "entropy")
grid_search =␣
 ↪GridSearchCV(forest_tuned,param_grid=param_grid,scoring="accuracy",n_jobs =␣
 ↪-1,cv=5)
grid_search.fit(X_train, y_train)
print(grid_search.best_params_)
```

```
{'max_depth': 8, 'min_samples_leaf': 2, 'min_samples_split': 2}
```

[75]:
```
forest_tuned =␣
 ↪RandomForestClassifier(max_depth=8,min_samples_leaf=2,min_samples_split=2,criterion␣
 ↪= "entropy",random_state = 42,n_jobs = -1)
forest_tuned= forest.fit(X_train, y_train)
show_summary(forest_tuned, X_val, y_val)
plot_conf_ROC(forest_tuned, X_val, y_val)
```

```
              precision    recall  f1-score   support

          no       0.70      0.71      0.71      5570
         yes       0.69      0.68      0.69      5370
```

```
    accuracy                           0.70     10940
   macro avg       0.70      0.70      0.70     10940
weighted avg       0.70      0.70      0.70     10940

Specificity :  0.7100538599640933
Sensitivity :  0.6847299813780261
```

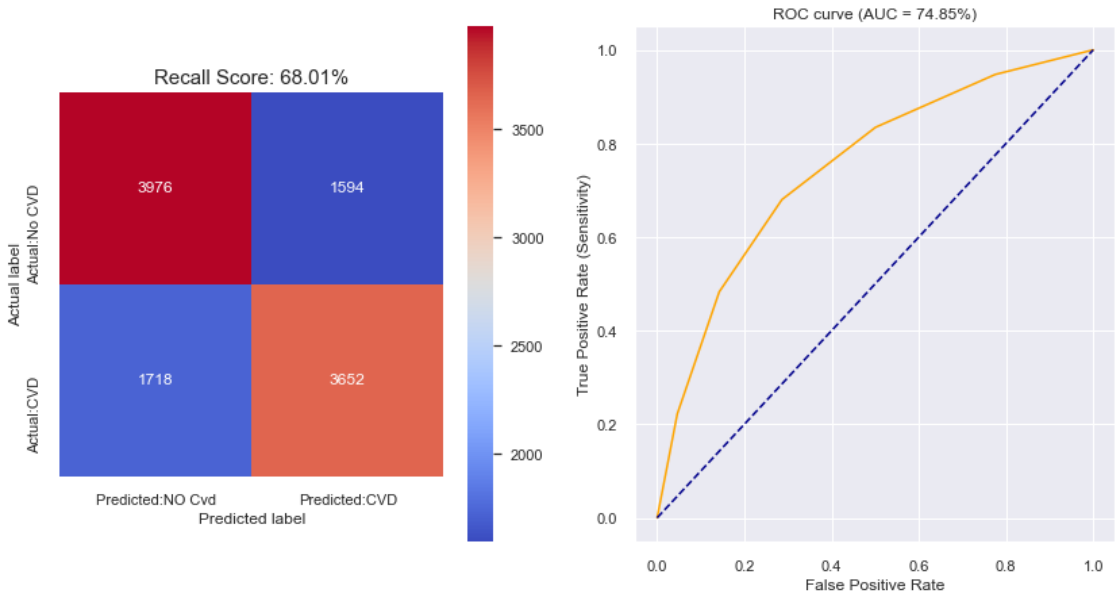


[147]:
```
y_pred = forest_tuned .predict(X_test)
accuracy_score(y_test, y_pred)
```

[147]: 0.7021572212065813

[148]:
```
knn=KNeighborsClassifier()
knn.fit(X_train, y_train)
show_summary(knn, X_val, y_val)
plot_conf_ROC(knn, X_val, y_val)
```

```
              precision    recall  f1-score   support

          no       0.70      0.71      0.71      5570
         yes       0.70      0.68      0.69      5370

    accuracy                           0.70     10940
   macro avg       0.70      0.70      0.70     10940
weighted avg       0.70      0.70      0.70     10940

Specificity :  0.7138240574506284
Sensitivity :  0.6800744878957169
```



[484]:
```
knn = KNeighborsClassifier()
k_range = list(range(1, 300))
param_grid = dict(n_neighbors=k_range)
grid = GridSearchCV(knn, param_grid, cv=10, scoring='accuracy',␣
 ↪return_train_score=False,verbose=3)
grid_search=grid.fit(X_train, y_train)
print(grid_search.best_params_)
```

```
Fitting 10 folds for each of 299 candidates, totalling 2990 fits
[CV 1/10] END …n_neighbors=1;, score=0.644 total time=   0.2s
[CV 2/10] END …n_neighbors=1;, score=0.636 total time=   0.2s
[CV 3/10] END …n_neighbors=1;, score=0.636 total time=   0.2s
[CV 4/10] END …n_neighbors=1;, score=0.640 total time=   0.2s
[CV 5/10] END …n_neighbors=1;, score=0.629 total time=   0.2s
[CV 6/10] END …n_neighbors=1;, score=0.624 total time=   0.2s
[CV 7/10] END …n_neighbors=1;, score=0.647 total time=   0.2s
[CV 8/10] END …n_neighbors=1;, score=0.632 total time=   0.2s
[CV 9/10] END …n_neighbors=1;, score=0.634 total time=   0.2s
[CV 10/10] END …n_neighbors=1;, score=0.639 total time=   0.2s
[CV 1/10] END …n_neighbors=2;, score=0.646 total time=   0.2s
[CV 2/10] END …n_neighbors=2;, score=0.645 total time=   0.2s
[CV 3/10] END …n_neighbors=2;, score=0.630 total time=   0.3s
[CV 4/10] END …n_neighbors=2;, score=0.644 total time=   0.3s
[CV 5/10] END …n_neighbors=2;, score=0.639 total time=   0.2s
[CV 6/10] END …n_neighbors=2;, score=0.634 total time=   0.2s
[CV 7/10] END …n_neighbors=2;, score=0.642 total time=   0.2s
[CV 8/10] END …n_neighbors=2;, score=0.636 total time=   0.2s
[CV 9/10] END …n_neighbors=2;, score=0.642 total time=   0.2s
```

```
[CV 10/10] END …n_neighbors=2;, score=0.640 total time=   0.2s
[CV 1/10] END …n_neighbors=3;, score=0.683 total time=   0.2s
[CV 2/10] END …n_neighbors=3;, score=0.673 total time=   0.3s
[CV 3/10] END …n_neighbors=3;, score=0.672 total time=   0.3s
[CV 4/10] END …n_neighbors=3;, score=0.671 total time=   0.2s
[CV 5/10] END …n_neighbors=3;, score=0.673 total time=   0.3s
[CV 6/10] END …n_neighbors=3;, score=0.669 total time=   0.2s
[CV 7/10] END …n_neighbors=3;, score=0.675 total time=   0.2s
[CV 8/10] END …n_neighbors=3;, score=0.667 total time=   0.2s
[CV 9/10] END …n_neighbors=3;, score=0.684 total time=   0.2s
[CV 10/10] END …n_neighbors=3;, score=0.672 total time=   0.2s
[CV 1/10] END …n_neighbors=4;, score=0.684 total time=   0.3s
[CV 2/10] END …n_neighbors=4;, score=0.677 total time=   0.4s
[CV 3/10] END …n_neighbors=4;, score=0.673 total time=   0.4s
[CV 4/10] END …n_neighbors=4;, score=0.679 total time=   0.3s
[CV 5/10] END …n_neighbors=4;, score=0.681 total time=   0.2s
[CV 6/10] END …n_neighbors=4;, score=0.675 total time=   0.2s
[CV 7/10] END …n_neighbors=4;, score=0.677 total time=   0.2s
[CV 8/10] END …n_neighbors=4;, score=0.668 total time=   0.3s
[CV 9/10] END …n_neighbors=4;, score=0.686 total time=   0.2s
[CV 10/10] END …n_neighbors=4;, score=0.679 total time=   0.3s
[CV 1/10] END …n_neighbors=5;, score=0.698 total time=   0.4s
[CV 2/10] END …n_neighbors=5;, score=0.691 total time=   0.4s
[CV 3/10] END …n_neighbors=5;, score=0.696 total time=   0.2s
[CV 4/10] END …n_neighbors=5;, score=0.686 total time=   0.2s
[CV 5/10] END …n_neighbors=5;, score=0.691 total time=   0.2s
[CV 6/10] END …n_neighbors=5;, score=0.685 total time=   0.2s
[CV 7/10] END …n_neighbors=5;, score=0.694 total time=   0.2s
[CV 8/10] END …n_neighbors=5;, score=0.677 total time=   0.2s
[CV 9/10] END …n_neighbors=5;, score=0.700 total time=   0.2s
[CV 10/10] END …n_neighbors=5;, score=0.689 total time=   0.2s
[CV 1/10] END …n_neighbors=6;, score=0.706 total time=   0.2s
[CV 2/10] END …n_neighbors=6;, score=0.695 total time=   0.2s
[CV 3/10] END …n_neighbors=6;, score=0.696 total time=   0.2s
[CV 4/10] END …n_neighbors=6;, score=0.695 total time=   0.2s
[CV 5/10] END …n_neighbors=6;, score=0.696 total time=   0.2s
[CV 6/10] END …n_neighbors=6;, score=0.693 total time=   0.2s
[CV 7/10] END …n_neighbors=6;, score=0.701 total time=   0.2s
[CV 8/10] END …n_neighbors=6;, score=0.681 total time=   0.2s
[CV 9/10] END …n_neighbors=6;, score=0.704 total time=   0.2s
[CV 10/10] END …n_neighbors=6;, score=0.693 total time=   0.2s
[CV 1/10] END …n_neighbors=7;, score=0.710 total time=   0.2s
[CV 2/10] END …n_neighbors=7;, score=0.705 total time=   0.2s
[CV 3/10] END …n_neighbors=7;, score=0.699 total time=   0.2s
[CV 4/10] END …n_neighbors=7;, score=0.699 total time=   0.2s
[CV 5/10] END …n_neighbors=7;, score=0.709 total time=   0.2s
[CV 6/10] END …n_neighbors=7;, score=0.697 total time=   0.2s
[CV 7/10] END …n_neighbors=7;, score=0.700 total time=   0.2s
```

```
[CV 8/10] END …n_neighbors=7;, score=0.694 total time=   0.2s
[CV 9/10] END …n_neighbors=7;, score=0.711 total time=   0.2s
[CV 10/10] END …n_neighbors=7;, score=0.699 total time=   0.2s
[CV 1/10] END …n_neighbors=8;, score=0.710 total time=   0.2s
[CV 2/10] END …n_neighbors=8;, score=0.702 total time=   0.2s
[CV 3/10] END …n_neighbors=8;, score=0.708 total time=   0.3s
[CV 4/10] END …n_neighbors=8;, score=0.702 total time=   0.2s
[CV 5/10] END …n_neighbors=8;, score=0.709 total time=   0.2s
[CV 6/10] END …n_neighbors=8;, score=0.703 total time=   0.2s
[CV 7/10] END …n_neighbors=8;, score=0.713 total time=   0.3s
[CV 8/10] END …n_neighbors=8;, score=0.702 total time=   0.2s
[CV 9/10] END …n_neighbors=8;, score=0.713 total time=   0.2s
[CV 10/10] END …n_neighbors=8;, score=0.705 total time=   0.2s
[CV 1/10] END …n_neighbors=9;, score=0.715 total time=   0.3s
[CV 2/10] END …n_neighbors=9;, score=0.706 total time=   0.2s
[CV 3/10] END …n_neighbors=9;, score=0.700 total time=   0.3s
[CV 4/10] END …n_neighbors=9;, score=0.705 total time=   0.3s
[CV 5/10] END …n_neighbors=9;, score=0.715 total time=   0.3s
[CV 6/10] END …n_neighbors=9;, score=0.703 total time=   0.2s
[CV 7/10] END …n_neighbors=9;, score=0.713 total time=   0.2s
[CV 8/10] END …n_neighbors=9;, score=0.698 total time=   0.2s
[CV 9/10] END …n_neighbors=9;, score=0.715 total time=   0.2s
[CV 10/10] END …n_neighbors=9;, score=0.709 total time=   0.2s
[CV 1/10] END …n_neighbors=10;, score=0.716 total time=   0.2s
[CV 2/10] END …n_neighbors=10;, score=0.706 total time=   0.2s
[CV 3/10] END …n_neighbors=10;, score=0.708 total time=   0.2s
[CV 4/10] END …n_neighbors=10;, score=0.709 total time=   0.2s
[CV 5/10] END …n_neighbors=10;, score=0.715 total time=   0.3s
[CV 6/10] END …n_neighbors=10;, score=0.710 total time=   0.3s
[CV 7/10] END …n_neighbors=10;, score=0.719 total time=   0.2s
[CV 8/10] END …n_neighbors=10;, score=0.708 total time=   0.2s
[CV 9/10] END …n_neighbors=10;, score=0.719 total time=   0.2s
[CV 10/10] END …n_neighbors=10;, score=0.708 total time=   0.2s
[CV 1/10] END …n_neighbors=11;, score=0.714 total time=   0.3s
[CV 2/10] END …n_neighbors=11;, score=0.708 total time=   0.2s
[CV 3/10] END …n_neighbors=11;, score=0.706 total time=   0.3s
[CV 4/10] END …n_neighbors=11;, score=0.710 total time=   0.2s
[CV 5/10] END …n_neighbors=11;, score=0.718 total time=   0.3s
[CV 6/10] END …n_neighbors=11;, score=0.713 total time=   0.3s
[CV 7/10] END …n_neighbors=11;, score=0.715 total time=   0.2s
[CV 8/10] END …n_neighbors=11;, score=0.703 total time=   0.3s
[CV 9/10] END …n_neighbors=11;, score=0.720 total time=   0.2s
[CV 10/10] END …n_neighbors=11;, score=0.706 total time=   0.3s
[CV 1/10] END …n_neighbors=12;, score=0.720 total time=   0.3s
[CV 2/10] END …n_neighbors=12;, score=0.708 total time=   0.2s
[CV 3/10] END …n_neighbors=12;, score=0.711 total time=   0.3s
[CV 4/10] END …n_neighbors=12;, score=0.713 total time=   0.3s
[CV 5/10] END …n_neighbors=12;, score=0.719 total time=   0.3s
```

```
[CV 6/10] END …n_neighbors=12;, score=0.713 total time=   0.2s
[CV 7/10] END …n_neighbors=12;, score=0.721 total time=   0.3s
[CV 8/10] END …n_neighbors=12;, score=0.708 total time=   0.2s
[CV 9/10] END …n_neighbors=12;, score=0.717 total time=   0.2s
[CV 10/10] END …n_neighbors=12;, score=0.710 total time=   0.3s
[CV 1/10] END …n_neighbors=13;, score=0.721 total time=   0.3s
[CV 2/10] END …n_neighbors=13;, score=0.707 total time=   0.3s
[CV 3/10] END …n_neighbors=13;, score=0.712 total time=   0.3s
[CV 4/10] END …n_neighbors=13;, score=0.715 total time=   0.3s
[CV 5/10] END …n_neighbors=13;, score=0.720 total time=   0.3s
[CV 6/10] END …n_neighbors=13;, score=0.712 total time=   0.2s
[CV 7/10] END …n_neighbors=13;, score=0.718 total time=   0.3s
[CV 8/10] END …n_neighbors=13;, score=0.701 total time=   0.2s
[CV 9/10] END …n_neighbors=13;, score=0.715 total time=   0.3s
[CV 10/10] END …n_neighbors=13;, score=0.712 total time=   0.3s
[CV 1/10] END …n_neighbors=14;, score=0.725 total time=   0.3s
[CV 2/10] END …n_neighbors=14;, score=0.711 total time=   0.3s
[CV 3/10] END …n_neighbors=14;, score=0.715 total time=   0.3s
[CV 4/10] END …n_neighbors=14;, score=0.716 total time=   0.3s
[CV 5/10] END …n_neighbors=14;, score=0.720 total time=   0.3s
[CV 6/10] END …n_neighbors=14;, score=0.716 total time=   0.3s
[CV 7/10] END …n_neighbors=14;, score=0.720 total time=   0.3s
[CV 8/10] END …n_neighbors=14;, score=0.701 total time=   0.3s
[CV 9/10] END …n_neighbors=14;, score=0.721 total time=   0.3s
[CV 10/10] END …n_neighbors=14;, score=0.715 total time=   0.3s
[CV 1/10] END …n_neighbors=15;, score=0.724 total time=   0.3s
[CV 2/10] END …n_neighbors=15;, score=0.708 total time=   0.3s
[CV 3/10] END …n_neighbors=15;, score=0.713 total time=   0.3s
[CV 4/10] END …n_neighbors=15;, score=0.716 total time=   0.3s
[CV 5/10] END …n_neighbors=15;, score=0.719 total time=   0.3s
[CV 6/10] END …n_neighbors=15;, score=0.715 total time=   0.3s
[CV 7/10] END …n_neighbors=15;, score=0.717 total time=   0.3s
[CV 8/10] END …n_neighbors=15;, score=0.701 total time=   0.3s
[CV 9/10] END …n_neighbors=15;, score=0.722 total time=   0.3s
[CV 10/10] END …n_neighbors=15;, score=0.715 total time=   0.3s
[CV 1/10] END …n_neighbors=16;, score=0.729 total time=   0.3s
[CV 2/10] END …n_neighbors=16;, score=0.712 total time=   0.3s
[CV 3/10] END …n_neighbors=16;, score=0.717 total time=   0.3s
[CV 4/10] END …n_neighbors=16;, score=0.715 total time=   0.3s
[CV 5/10] END …n_neighbors=16;, score=0.723 total time=   0.3s
[CV 6/10] END …n_neighbors=16;, score=0.720 total time=   0.3s
[CV 7/10] END …n_neighbors=16;, score=0.720 total time=   0.3s
[CV 8/10] END …n_neighbors=16;, score=0.704 total time=   0.3s
[CV 9/10] END …n_neighbors=16;, score=0.725 total time=   0.3s
[CV 10/10] END …n_neighbors=16;, score=0.720 total time=   0.3s
[CV 1/10] END …n_neighbors=17;, score=0.729 total time=   0.3s
[CV 2/10] END …n_neighbors=17;, score=0.710 total time=   0.3s
[CV 3/10] END …n_neighbors=17;, score=0.716 total time=   0.3s
```

```
[CV 4/10] END …n_neighbors=17;, score=0.713 total time=   0.3s
[CV 5/10] END …n_neighbors=17;, score=0.722 total time=   0.3s
[CV 6/10] END …n_neighbors=17;, score=0.718 total time=   0.3s
[CV 7/10] END …n_neighbors=17;, score=0.722 total time=   0.3s
[CV 8/10] END …n_neighbors=17;, score=0.704 total time=   0.3s
[CV 9/10] END …n_neighbors=17;, score=0.723 total time=   0.3s
[CV 10/10] END …n_neighbors=17;, score=0.719 total time=   0.3s
[CV 1/10] END …n_neighbors=18;, score=0.730 total time=   0.3s
[CV 2/10] END …n_neighbors=18;, score=0.712 total time=   0.3s
[CV 3/10] END …n_neighbors=18;, score=0.719 total time=   0.3s
[CV 4/10] END …n_neighbors=18;, score=0.714 total time=   0.3s
[CV 5/10] END …n_neighbors=18;, score=0.722 total time=   0.3s
[CV 6/10] END …n_neighbors=18;, score=0.720 total time=   0.3s
[CV 7/10] END …n_neighbors=18;, score=0.722 total time=   0.3s
[CV 8/10] END …n_neighbors=18;, score=0.706 total time=   0.3s
[CV 9/10] END …n_neighbors=18;, score=0.728 total time=   0.3s
[CV 10/10] END …n_neighbors=18;, score=0.722 total time=   0.3s
[CV 1/10] END …n_neighbors=19;, score=0.730 total time=   0.3s
[CV 2/10] END …n_neighbors=19;, score=0.710 total time=   0.3s
[CV 3/10] END …n_neighbors=19;, score=0.716 total time=   0.3s
[CV 4/10] END …n_neighbors=19;, score=0.713 total time=   0.3s
[CV 5/10] END …n_neighbors=19;, score=0.724 total time=   0.3s
[CV 6/10] END …n_neighbors=19;, score=0.719 total time=   0.3s
[CV 7/10] END …n_neighbors=19;, score=0.721 total time=   0.3s
[CV 8/10] END …n_neighbors=19;, score=0.709 total time=   0.3s
[CV 9/10] END …n_neighbors=19;, score=0.726 total time=   0.3s
[CV 10/10] END …n_neighbors=19;, score=0.725 total time=   0.3s
[CV 1/10] END …n_neighbors=20;, score=0.731 total time=   0.3s
[CV 2/10] END …n_neighbors=20;, score=0.712 total time=   0.3s
[CV 3/10] END …n_neighbors=20;, score=0.719 total time=   0.3s
[CV 4/10] END …n_neighbors=20;, score=0.715 total time=   0.3s
[CV 5/10] END …n_neighbors=20;, score=0.728 total time=   0.3s
[CV 6/10] END …n_neighbors=20;, score=0.720 total time=   0.3s
[CV 7/10] END …n_neighbors=20;, score=0.723 total time=   0.3s
[CV 8/10] END …n_neighbors=20;, score=0.711 total time=   0.3s
[CV 9/10] END …n_neighbors=20;, score=0.730 total time=   0.3s
[CV 10/10] END …n_neighbors=20;, score=0.727 total time=   0.3s
[CV 1/10] END …n_neighbors=21;, score=0.731 total time=   0.3s
[CV 2/10] END …n_neighbors=21;, score=0.713 total time=   0.3s
[CV 3/10] END …n_neighbors=21;, score=0.715 total time=   0.3s
[CV 4/10] END …n_neighbors=21;, score=0.712 total time=   0.3s
[CV 5/10] END …n_neighbors=21;, score=0.724 total time=   0.3s
[CV 6/10] END …n_neighbors=21;, score=0.723 total time=   0.3s
[CV 7/10] END …n_neighbors=21;, score=0.724 total time=   0.3s
[CV 8/10] END …n_neighbors=21;, score=0.707 total time=   0.3s
[CV 9/10] END …n_neighbors=21;, score=0.727 total time=   0.3s
[CV 10/10] END …n_neighbors=21;, score=0.730 total time=   0.3s
[CV 1/10] END …n_neighbors=22;, score=0.725 total time=   0.3s
```

```
[CV 2/10] END …n_neighbors=22;, score=0.712 total time=   0.3s
[CV 3/10] END …n_neighbors=22;, score=0.718 total time=   0.3s
[CV 4/10] END …n_neighbors=22;, score=0.713 total time=   0.3s
[CV 5/10] END …n_neighbors=22;, score=0.726 total time=   0.3s
[CV 6/10] END …n_neighbors=22;, score=0.723 total time=   0.3s
[CV 7/10] END …n_neighbors=22;, score=0.727 total time=   0.3s
[CV 8/10] END …n_neighbors=22;, score=0.712 total time=   0.3s
[CV 9/10] END …n_neighbors=22;, score=0.730 total time=   0.3s
[CV 10/10] END …n_neighbors=22;, score=0.729 total time=   0.3s
[CV 1/10] END …n_neighbors=23;, score=0.726 total time=   0.3s
[CV 2/10] END …n_neighbors=23;, score=0.713 total time=   0.3s
[CV 3/10] END …n_neighbors=23;, score=0.719 total time=   0.3s
[CV 4/10] END …n_neighbors=23;, score=0.712 total time=   0.3s
[CV 5/10] END …n_neighbors=23;, score=0.727 total time=   0.3s
[CV 6/10] END …n_neighbors=23;, score=0.723 total time=   0.3s
[CV 7/10] END …n_neighbors=23;, score=0.723 total time=   0.3s
[CV 8/10] END …n_neighbors=23;, score=0.709 total time=   0.3s
[CV 9/10] END …n_neighbors=23;, score=0.729 total time=   0.3s
[CV 10/10] END …n_neighbors=23;, score=0.729 total time=   0.3s
[CV 1/10] END …n_neighbors=24;, score=0.726 total time=   0.3s
[CV 2/10] END …n_neighbors=24;, score=0.715 total time=   0.3s
[CV 3/10] END …n_neighbors=24;, score=0.721 total time=   0.3s
[CV 4/10] END …n_neighbors=24;, score=0.716 total time=   0.3s
[CV 5/10] END …n_neighbors=24;, score=0.731 total time=   0.3s
[CV 6/10] END …n_neighbors=24;, score=0.722 total time=   0.3s
[CV 7/10] END …n_neighbors=24;, score=0.726 total time=   0.3s
[CV 8/10] END …n_neighbors=24;, score=0.712 total time=   0.3s
[CV 9/10] END …n_neighbors=24;, score=0.730 total time=   0.3s
[CV 10/10] END …n_neighbors=24;, score=0.728 total time=   0.3s
[CV 1/10] END …n_neighbors=25;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=25;, score=0.714 total time=   0.3s
[CV 3/10] END …n_neighbors=25;, score=0.722 total time=   0.3s
[CV 4/10] END …n_neighbors=25;, score=0.712 total time=   0.3s
[CV 5/10] END …n_neighbors=25;, score=0.728 total time=   0.3s
[CV 6/10] END …n_neighbors=25;, score=0.724 total time=   0.3s
[CV 7/10] END …n_neighbors=25;, score=0.725 total time=   0.3s
[CV 8/10] END …n_neighbors=25;, score=0.713 total time=   0.3s
[CV 9/10] END …n_neighbors=25;, score=0.731 total time=   0.3s
[CV 10/10] END …n_neighbors=25;, score=0.728 total time=   0.3s
[CV 1/10] END …n_neighbors=26;, score=0.726 total time=   0.3s
[CV 2/10] END …n_neighbors=26;, score=0.715 total time=   0.3s
[CV 3/10] END …n_neighbors=26;, score=0.723 total time=   0.3s
[CV 4/10] END …n_neighbors=26;, score=0.715 total time=   0.3s
[CV 5/10] END …n_neighbors=26;, score=0.729 total time=   0.3s
[CV 6/10] END …n_neighbors=26;, score=0.726 total time=   0.3s
[CV 7/10] END …n_neighbors=26;, score=0.730 total time=   0.3s
[CV 8/10] END …n_neighbors=26;, score=0.711 total time=   0.3s
[CV 9/10] END …n_neighbors=26;, score=0.732 total time=   0.3s
```

```
[CV 10/10] END …n_neighbors=26;, score=0.729 total time=   0.3s
[CV 1/10] END …n_neighbors=27;, score=0.724 total time=   0.3s
[CV 2/10] END …n_neighbors=27;, score=0.716 total time=   0.3s
[CV 3/10] END …n_neighbors=27;, score=0.721 total time=   0.3s
[CV 4/10] END …n_neighbors=27;, score=0.717 total time=   0.3s
[CV 5/10] END …n_neighbors=27;, score=0.732 total time=   0.3s
[CV 6/10] END …n_neighbors=27;, score=0.726 total time=   0.3s
[CV 7/10] END …n_neighbors=27;, score=0.728 total time=   0.3s
[CV 8/10] END …n_neighbors=27;, score=0.714 total time=   0.3s
[CV 9/10] END …n_neighbors=27;, score=0.733 total time=   0.3s
[CV 10/10] END …n_neighbors=27;, score=0.730 total time=   0.3s
[CV 1/10] END …n_neighbors=28;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=28;, score=0.714 total time=   0.3s
[CV 3/10] END …n_neighbors=28;, score=0.724 total time=   0.3s
[CV 4/10] END …n_neighbors=28;, score=0.719 total time=   0.3s
[CV 5/10] END …n_neighbors=28;, score=0.731 total time=   0.3s
[CV 6/10] END …n_neighbors=28;, score=0.729 total time=   0.3s
[CV 7/10] END …n_neighbors=28;, score=0.728 total time=   0.3s
[CV 8/10] END …n_neighbors=28;, score=0.714 total time=   0.3s
[CV 9/10] END …n_neighbors=28;, score=0.731 total time=   0.3s
[CV 10/10] END …n_neighbors=28;, score=0.728 total time=   0.3s
[CV 1/10] END …n_neighbors=29;, score=0.726 total time=   0.3s
[CV 2/10] END …n_neighbors=29;, score=0.713 total time=   0.3s
[CV 3/10] END …n_neighbors=29;, score=0.722 total time=   0.3s
[CV 4/10] END …n_neighbors=29;, score=0.720 total time=   0.3s
[CV 5/10] END …n_neighbors=29;, score=0.731 total time=   0.3s
[CV 6/10] END …n_neighbors=29;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=29;, score=0.725 total time=   0.3s
[CV 8/10] END …n_neighbors=29;, score=0.712 total time=   0.3s
[CV 9/10] END …n_neighbors=29;, score=0.732 total time=   0.3s
[CV 10/10] END …n_neighbors=29;, score=0.727 total time=   0.3s
[CV 1/10] END …n_neighbors=30;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=30;, score=0.714 total time=   0.3s
[CV 3/10] END …n_neighbors=30;, score=0.723 total time=   0.3s
[CV 4/10] END …n_neighbors=30;, score=0.720 total time=   0.3s
[CV 5/10] END …n_neighbors=30;, score=0.732 total time=   0.3s
[CV 6/10] END …n_neighbors=30;, score=0.729 total time=   0.3s
[CV 7/10] END …n_neighbors=30;, score=0.730 total time=   0.3s
[CV 8/10] END …n_neighbors=30;, score=0.714 total time=   0.3s
[CV 9/10] END …n_neighbors=30;, score=0.732 total time=   0.3s
[CV 10/10] END …n_neighbors=30;, score=0.728 total time=   0.3s
[CV 1/10] END …n_neighbors=31;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=31;, score=0.714 total time=   0.3s
[CV 3/10] END …n_neighbors=31;, score=0.725 total time=   0.3s
[CV 4/10] END …n_neighbors=31;, score=0.720 total time=   0.3s
[CV 5/10] END …n_neighbors=31;, score=0.732 total time=   0.3s
[CV 6/10] END …n_neighbors=31;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=31;, score=0.730 total time=   0.3s
```

```
[CV 8/10] END …n_neighbors=31;, score=0.714 total time=   0.3s
[CV 9/10] END …n_neighbors=31;, score=0.732 total time=   0.3s
[CV 10/10] END …n_neighbors=31;, score=0.728 total time=   0.3s
[CV 1/10] END …n_neighbors=32;, score=0.726 total time=   0.3s
[CV 2/10] END …n_neighbors=32;, score=0.713 total time=   0.3s
[CV 3/10] END …n_neighbors=32;, score=0.724 total time=   0.3s
[CV 4/10] END …n_neighbors=32;, score=0.719 total time=   0.3s
[CV 5/10] END …n_neighbors=32;, score=0.734 total time=   0.3s
[CV 6/10] END …n_neighbors=32;, score=0.727 total time=   0.4s
[CV 7/10] END …n_neighbors=32;, score=0.731 total time=   0.3s
[CV 8/10] END …n_neighbors=32;, score=0.714 total time=   0.3s
[CV 9/10] END …n_neighbors=32;, score=0.733 total time=   0.3s
[CV 10/10] END …n_neighbors=32;, score=0.727 total time=   0.3s
[CV 1/10] END …n_neighbors=33;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=33;, score=0.713 total time=   0.3s
[CV 3/10] END …n_neighbors=33;, score=0.725 total time=   0.3s
[CV 4/10] END …n_neighbors=33;, score=0.719 total time=   0.3s
[CV 5/10] END …n_neighbors=33;, score=0.734 total time=   0.3s
[CV 6/10] END …n_neighbors=33;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=33;, score=0.730 total time=   0.3s
[CV 8/10] END …n_neighbors=33;, score=0.713 total time=   0.3s
[CV 9/10] END …n_neighbors=33;, score=0.733 total time=   0.3s
[CV 10/10] END …n_neighbors=33;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=34;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=34;, score=0.714 total time=   0.3s
[CV 3/10] END …n_neighbors=34;, score=0.726 total time=   0.3s
[CV 4/10] END …n_neighbors=34;, score=0.722 total time=   0.3s
[CV 5/10] END …n_neighbors=34;, score=0.735 total time=   0.3s
[CV 6/10] END …n_neighbors=34;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=34;, score=0.729 total time=   0.3s
[CV 8/10] END …n_neighbors=34;, score=0.718 total time=   0.3s
[CV 9/10] END …n_neighbors=34;, score=0.732 total time=   0.3s
[CV 10/10] END …n_neighbors=34;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=35;, score=0.725 total time=   0.3s
[CV 2/10] END …n_neighbors=35;, score=0.715 total time=   0.3s
[CV 3/10] END …n_neighbors=35;, score=0.724 total time=   0.3s
[CV 4/10] END …n_neighbors=35;, score=0.722 total time=   0.3s
[CV 5/10] END …n_neighbors=35;, score=0.734 total time=   0.3s
[CV 6/10] END …n_neighbors=35;, score=0.730 total time=   0.3s
[CV 7/10] END …n_neighbors=35;, score=0.731 total time=   0.3s
[CV 8/10] END …n_neighbors=35;, score=0.715 total time=   0.3s
[CV 9/10] END …n_neighbors=35;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=35;, score=0.729 total time=   0.3s
[CV 1/10] END …n_neighbors=36;, score=0.726 total time=   0.3s
[CV 2/10] END …n_neighbors=36;, score=0.716 total time=   0.3s
[CV 3/10] END …n_neighbors=36;, score=0.725 total time=   0.3s
[CV 4/10] END …n_neighbors=36;, score=0.720 total time=   0.3s
[CV 5/10] END …n_neighbors=36;, score=0.735 total time=   0.3s
```

```
[CV 6/10] END …n_neighbors=36;, score=0.729 total time=   0.3s
[CV 7/10] END …n_neighbors=36;, score=0.732 total time=   0.3s
[CV 8/10] END …n_neighbors=36;, score=0.717 total time=   0.3s
[CV 9/10] END …n_neighbors=36;, score=0.735 total time=   0.3s
[CV 10/10] END …n_neighbors=36;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=37;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=37;, score=0.716 total time=   0.3s
[CV 3/10] END …n_neighbors=37;, score=0.723 total time=   0.3s
[CV 4/10] END …n_neighbors=37;, score=0.719 total time=   0.3s
[CV 5/10] END …n_neighbors=37;, score=0.734 total time=   0.3s
[CV 6/10] END …n_neighbors=37;, score=0.729 total time=   0.3s
[CV 7/10] END …n_neighbors=37;, score=0.731 total time=   0.3s
[CV 8/10] END …n_neighbors=37;, score=0.715 total time=   0.3s
[CV 9/10] END …n_neighbors=37;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=37;, score=0.730 total time=   0.3s
[CV 1/10] END …n_neighbors=38;, score=0.729 total time=   0.3s
[CV 2/10] END …n_neighbors=38;, score=0.717 total time=   0.3s
[CV 3/10] END …n_neighbors=38;, score=0.724 total time=   0.3s
[CV 4/10] END …n_neighbors=38;, score=0.721 total time=   0.3s
[CV 5/10] END …n_neighbors=38;, score=0.737 total time=   0.3s
[CV 6/10] END …n_neighbors=38;, score=0.727 total time=   0.3s
[CV 7/10] END …n_neighbors=38;, score=0.732 total time=   0.3s
[CV 8/10] END …n_neighbors=38;, score=0.716 total time=   0.3s
[CV 9/10] END …n_neighbors=38;, score=0.735 total time=   0.3s
[CV 10/10] END …n_neighbors=38;, score=0.733 total time=   0.3s
[CV 1/10] END …n_neighbors=39;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=39;, score=0.716 total time=   0.3s
[CV 3/10] END …n_neighbors=39;, score=0.723 total time=   0.3s
[CV 4/10] END …n_neighbors=39;, score=0.719 total time=   0.3s
[CV 5/10] END …n_neighbors=39;, score=0.735 total time=   0.3s
[CV 6/10] END …n_neighbors=39;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=39;, score=0.730 total time=   0.3s
[CV 8/10] END …n_neighbors=39;, score=0.715 total time=   0.3s
[CV 9/10] END …n_neighbors=39;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=39;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=40;, score=0.729 total time=   0.3s
[CV 2/10] END …n_neighbors=40;, score=0.716 total time=   0.3s
[CV 3/10] END …n_neighbors=40;, score=0.725 total time=   0.3s
[CV 4/10] END …n_neighbors=40;, score=0.721 total time=   0.3s
[CV 5/10] END …n_neighbors=40;, score=0.735 total time=   0.3s
[CV 6/10] END …n_neighbors=40;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=40;, score=0.732 total time=   0.3s
[CV 8/10] END …n_neighbors=40;, score=0.717 total time=   0.3s
[CV 9/10] END …n_neighbors=40;, score=0.735 total time=   0.3s
[CV 10/10] END …n_neighbors=40;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=41;, score=0.729 total time=   0.3s
[CV 2/10] END …n_neighbors=41;, score=0.715 total time=   0.3s
[CV 3/10] END …n_neighbors=41;, score=0.723 total time=   0.3s
```

```
[CV 4/10] END …n_neighbors=41;, score=0.721 total time=   0.3s
[CV 5/10] END …n_neighbors=41;, score=0.732 total time=   0.3s
[CV 6/10] END …n_neighbors=41;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=41;, score=0.731 total time=   0.3s
[CV 8/10] END …n_neighbors=41;, score=0.719 total time=   0.3s
[CV 9/10] END …n_neighbors=41;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=41;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=42;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=42;, score=0.718 total time=   0.3s
[CV 3/10] END …n_neighbors=42;, score=0.727 total time=   0.3s
[CV 4/10] END …n_neighbors=42;, score=0.722 total time=   0.3s
[CV 5/10] END …n_neighbors=42;, score=0.733 total time=   0.3s
[CV 6/10] END …n_neighbors=42;, score=0.730 total time=   0.3s
[CV 7/10] END …n_neighbors=42;, score=0.735 total time=   0.3s
[CV 8/10] END …n_neighbors=42;, score=0.719 total time=   0.3s
[CV 9/10] END …n_neighbors=42;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=42;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=43;, score=0.725 total time=   0.3s
[CV 2/10] END …n_neighbors=43;, score=0.715 total time=   0.3s
[CV 3/10] END …n_neighbors=43;, score=0.724 total time=   0.3s
[CV 4/10] END …n_neighbors=43;, score=0.719 total time=   0.3s
[CV 5/10] END …n_neighbors=43;, score=0.731 total time=   0.3s
[CV 6/10] END …n_neighbors=43;, score=0.730 total time=   0.3s
[CV 7/10] END …n_neighbors=43;, score=0.732 total time=   0.3s
[CV 8/10] END …n_neighbors=43;, score=0.716 total time=   0.3s
[CV 9/10] END …n_neighbors=43;, score=0.736 total time=   0.3s
[CV 10/10] END …n_neighbors=43;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=44;, score=0.728 total time=   0.3s
[CV 2/10] END …n_neighbors=44;, score=0.718 total time=   0.3s
[CV 3/10] END …n_neighbors=44;, score=0.727 total time=   0.3s
[CV 4/10] END …n_neighbors=44;, score=0.727 total time=   0.3s
[CV 5/10] END …n_neighbors=44;, score=0.734 total time=   0.3s
[CV 6/10] END …n_neighbors=44;, score=0.729 total time=   0.3s
[CV 7/10] END …n_neighbors=44;, score=0.735 total time=   0.3s
[CV 8/10] END …n_neighbors=44;, score=0.716 total time=   0.3s
[CV 9/10] END …n_neighbors=44;, score=0.735 total time=   0.3s
[CV 10/10] END …n_neighbors=44;, score=0.732 total time=   0.3s
[CV 1/10] END …n_neighbors=45;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=45;, score=0.717 total time=   0.3s
[CV 3/10] END …n_neighbors=45;, score=0.725 total time=   0.3s
[CV 4/10] END …n_neighbors=45;, score=0.721 total time=   0.3s
[CV 5/10] END …n_neighbors=45;, score=0.733 total time=   0.3s
[CV 6/10] END …n_neighbors=45;, score=0.728 total time=   0.3s
[CV 7/10] END …n_neighbors=45;, score=0.733 total time=   0.3s
[CV 8/10] END …n_neighbors=45;, score=0.714 total time=   0.3s
[CV 9/10] END …n_neighbors=45;, score=0.737 total time=   0.3s
[CV 10/10] END …n_neighbors=45;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=46;, score=0.726 total time=   0.3s
```

```
[CV 2/10] END …n_neighbors=46;, score=0.719 total time=   0.3s
[CV 3/10] END …n_neighbors=46;, score=0.726 total time=   0.3s
[CV 4/10] END …n_neighbors=46;, score=0.726 total time=   0.3s
[CV 5/10] END …n_neighbors=46;, score=0.736 total time=   0.3s
[CV 6/10] END …n_neighbors=46;, score=0.729 total time=   0.3s
[CV 7/10] END …n_neighbors=46;, score=0.735 total time=   0.3s
[CV 8/10] END …n_neighbors=46;, score=0.716 total time=   0.3s
[CV 9/10] END …n_neighbors=46;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=46;, score=0.734 total time=   0.3s
[CV 1/10] END …n_neighbors=47;, score=0.727 total time=   0.3s
[CV 2/10] END …n_neighbors=47;, score=0.717 total time=   0.3s
[CV 3/10] END …n_neighbors=47;, score=0.726 total time=   0.3s
[CV 4/10] END …n_neighbors=47;, score=0.723 total time=   0.3s
[CV 5/10] END …n_neighbors=47;, score=0.734 total time=   0.3s
[CV 6/10] END …n_neighbors=47;, score=0.731 total time=   0.3s
[CV 7/10] END …n_neighbors=47;, score=0.730 total time=   0.3s
[CV 8/10] END …n_neighbors=47;, score=0.715 total time=   0.3s
[CV 9/10] END …n_neighbors=47;, score=0.735 total time=   0.3s
[CV 10/10] END …n_neighbors=47;, score=0.731 total time=   0.3s
[CV 1/10] END …n_neighbors=48;, score=0.729 total time=   0.3s
[CV 2/10] END …n_neighbors=48;, score=0.717 total time=   0.3s
[CV 3/10] END …n_neighbors=48;, score=0.729 total time=   0.3s
[CV 4/10] END …n_neighbors=48;, score=0.724 total time=   0.3s
[CV 5/10] END …n_neighbors=48;, score=0.734 total time=   0.3s
[CV 6/10] END …n_neighbors=48;, score=0.730 total time=   0.3s
[CV 7/10] END …n_neighbors=48;, score=0.735 total time=   0.3s
[CV 8/10] END …n_neighbors=48;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=48;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=48;, score=0.733 total time=   0.3s
[CV 1/10] END …n_neighbors=49;, score=0.728 total time=   0.3s
[CV 2/10] END …n_neighbors=49;, score=0.717 total time=   0.3s
[CV 3/10] END …n_neighbors=49;, score=0.728 total time=   0.4s
[CV 4/10] END …n_neighbors=49;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=49;, score=0.734 total time=   0.5s
[CV 6/10] END …n_neighbors=49;, score=0.729 total time=   0.4s
[CV 7/10] END …n_neighbors=49;, score=0.733 total time=   0.4s
[CV 8/10] END …n_neighbors=49;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=49;, score=0.734 total time=   0.3s
[CV 10/10] END …n_neighbors=49;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=50;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=50;, score=0.717 total time=   0.3s
[CV 3/10] END …n_neighbors=50;, score=0.728 total time=   0.4s
[CV 4/10] END …n_neighbors=50;, score=0.724 total time=   0.3s
[CV 5/10] END …n_neighbors=50;, score=0.732 total time=   0.4s
[CV 6/10] END …n_neighbors=50;, score=0.731 total time=   0.4s
[CV 7/10] END …n_neighbors=50;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=50;, score=0.718 total time=   0.3s
[CV 9/10] END …n_neighbors=50;, score=0.733 total time=   0.4s
```

```
[CV 10/10] END …n_neighbors=50;, score=0.735 total time=   0.4s
[CV 1/10] END …n_neighbors=51;, score=0.728 total time=   0.3s
[CV 2/10] END …n_neighbors=51;, score=0.715 total time=   0.4s
[CV 3/10] END …n_neighbors=51;, score=0.728 total time=   0.4s
[CV 4/10] END …n_neighbors=51;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=51;, score=0.734 total time=   0.4s
[CV 6/10] END …n_neighbors=51;, score=0.730 total time=   0.4s
[CV 7/10] END …n_neighbors=51;, score=0.732 total time=   0.4s
[CV 8/10] END …n_neighbors=51;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=51;, score=0.736 total time=   0.4s
[CV 10/10] END …n_neighbors=51;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=52;, score=0.729 total time=   0.3s
[CV 2/10] END …n_neighbors=52;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=52;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=52;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=52;, score=0.734 total time=   0.4s
[CV 6/10] END …n_neighbors=52;, score=0.730 total time=   1.6s
[CV 7/10] END …n_neighbors=52;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=52;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=52;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=52;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=53;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=53;, score=0.717 total time=   0.4s
[CV 3/10] END …n_neighbors=53;, score=0.726 total time=   0.4s
[CV 4/10] END …n_neighbors=53;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=53;, score=0.734 total time=   0.4s
[CV 6/10] END …n_neighbors=53;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=53;, score=0.731 total time=   0.4s
[CV 8/10] END …n_neighbors=53;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=53;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=53;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=54;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=54;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=54;, score=0.725 total time=   0.4s
[CV 4/10] END …n_neighbors=54;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=54;, score=0.733 total time=   0.4s
[CV 6/10] END …n_neighbors=54;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=54;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=54;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=54;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=54;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=55;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=55;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=55;, score=0.725 total time=   0.4s
[CV 4/10] END …n_neighbors=55;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=55;, score=0.734 total time=   0.4s
[CV 6/10] END …n_neighbors=55;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=55;, score=0.733 total time=   0.5s
```

```
[CV 8/10] END …n_neighbors=55;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=55;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=55;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=56;, score=0.727 total time=   0.4s
[CV 2/10] END …n_neighbors=56;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=56;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=56;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=56;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=56;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=56;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=56;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=56;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=56;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=57;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=57;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=57;, score=0.728 total time=   0.4s
[CV 4/10] END …n_neighbors=57;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=57;, score=0.735 total time=   0.4s
[CV 6/10] END …n_neighbors=57;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=57;, score=0.732 total time=   0.4s
[CV 8/10] END …n_neighbors=57;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=57;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=57;, score=0.730 total time=   0.4s
[CV 1/10] END …n_neighbors=58;, score=0.726 total time=   0.4s
[CV 2/10] END …n_neighbors=58;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=58;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=58;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=58;, score=0.733 total time=   0.4s
[CV 6/10] END …n_neighbors=58;, score=0.731 total time=   0.4s
[CV 7/10] END …n_neighbors=58;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=58;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=58;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=58;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=59;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=59;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=59;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=59;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=59;, score=0.734 total time=   0.4s
[CV 6/10] END …n_neighbors=59;, score=0.731 total time=   0.4s
[CV 7/10] END …n_neighbors=59;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=59;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=59;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=59;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=60;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=60;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=60;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=60;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=60;, score=0.735 total time=   0.4s
```

```
[CV 6/10] END …n_neighbors=60;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=60;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=60;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=60;, score=0.736 total time=   0.4s
[CV 10/10] END …n_neighbors=60;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=61;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=61;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=61;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=61;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=61;, score=0.734 total time=   0.4s
[CV 6/10] END …n_neighbors=61;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=61;, score=0.732 total time=   0.4s
[CV 8/10] END …n_neighbors=61;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=61;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=61;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=62;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=62;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=62;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=62;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=62;, score=0.735 total time=   0.4s
[CV 6/10] END …n_neighbors=62;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=62;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=62;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=62;, score=0.736 total time=   0.4s
[CV 10/10] END …n_neighbors=62;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=63;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=63;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=63;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=63;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=63;, score=0.735 total time=   0.4s
[CV 6/10] END …n_neighbors=63;, score=0.731 total time=   0.4s
[CV 7/10] END …n_neighbors=63;, score=0.732 total time=   0.4s
[CV 8/10] END …n_neighbors=63;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=63;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=63;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=64;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=64;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=64;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=64;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=64;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=64;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=64;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=64;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=64;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=64;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=65;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=65;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=65;, score=0.730 total time=   0.4s
```

```
[CV 4/10] END …n_neighbors=65;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=65;, score=0.735 total time=   0.4s
[CV 6/10] END …n_neighbors=65;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=65;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=65;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=65;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=65;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=66;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=66;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=66;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=66;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=66;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=66;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=66;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=66;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=66;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=66;, score=0.730 total time=   0.4s
[CV 1/10] END …n_neighbors=67;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=67;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=67;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=67;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=67;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=67;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=67;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=67;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=67;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=67;, score=0.730 total time=   0.5s
[CV 1/10] END …n_neighbors=68;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=68;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=68;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=68;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=68;, score=0.737 total time=   0.4s
[CV 6/10] END …n_neighbors=68;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=68;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=68;, score=0.715 total time=   0.4s
[CV 9/10] END …n_neighbors=68;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=68;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=69;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=69;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=69;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=69;, score=0.723 total time=   0.4s
[CV 5/10] END …n_neighbors=69;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=69;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=69;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=69;, score=0.714 total time=   0.4s
[CV 9/10] END …n_neighbors=69;, score=0.732 total time=   0.4s
[CV 10/10] END …n_neighbors=69;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=70;, score=0.730 total time=   0.4s
```

```
[CV 2/10] END …n_neighbors=70;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=70;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=70;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=70;, score=0.737 total time=   0.4s
[CV 6/10] END …n_neighbors=70;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=70;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=70;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=70;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=70;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=71;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=71;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=71;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=71;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=71;, score=0.737 total time=   0.4s
[CV 6/10] END …n_neighbors=71;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=71;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=71;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=71;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=71;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=72;, score=0.727 total time=   0.4s
[CV 2/10] END …n_neighbors=72;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=72;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=72;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=72;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=72;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=72;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=72;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=72;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=72;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=73;, score=0.727 total time=   0.4s
[CV 2/10] END …n_neighbors=73;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=73;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=73;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=73;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=73;, score=0.735 total time=   0.4s
[CV 7/10] END …n_neighbors=73;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=73;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=73;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=73;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=74;, score=0.726 total time=   0.4s
[CV 2/10] END …n_neighbors=74;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=74;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=74;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=74;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=74;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=74;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=74;, score=0.716 total time=   0.4s
[CV 9/10] END …n_neighbors=74;, score=0.734 total time=   0.4s
```

```
[CV 10/10] END …n_neighbors=74;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=75;, score=0.726 total time=   0.4s
[CV 2/10] END …n_neighbors=75;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=75;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=75;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=75;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=75;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=75;, score=0.734 total time=   0.4s
[CV 8/10] END …n_neighbors=75;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=75;, score=0.732 total time=   0.4s
[CV 10/10] END …n_neighbors=75;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=76;, score=0.725 total time=   0.4s
[CV 2/10] END …n_neighbors=76;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=76;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=76;, score=0.728 total time=   0.5s
[CV 5/10] END …n_neighbors=76;, score=0.735 total time=   0.4s
[CV 6/10] END …n_neighbors=76;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=76;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=76;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=76;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=76;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=77;, score=0.725 total time=   0.4s
[CV 2/10] END …n_neighbors=77;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=77;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=77;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=77;, score=0.735 total time=   0.4s
[CV 6/10] END …n_neighbors=77;, score=0.735 total time=   0.4s
[CV 7/10] END …n_neighbors=77;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=77;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=77;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=77;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=78;, score=0.726 total time=   0.4s
[CV 2/10] END …n_neighbors=78;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=78;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=78;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=78;, score=0.737 total time=   0.4s
[CV 6/10] END …n_neighbors=78;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=78;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=78;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=78;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=78;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=79;, score=0.725 total time=   0.4s
[CV 2/10] END …n_neighbors=79;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=79;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=79;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=79;, score=0.737 total time=   0.4s
[CV 6/10] END …n_neighbors=79;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=79;, score=0.737 total time=   0.4s
```

```
[CV 8/10] END …n_neighbors=79;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=79;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=79;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=80;, score=0.726 total time=   0.5s
[CV 2/10] END …n_neighbors=80;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=80;, score=0.732 total time=   0.4s
[CV 4/10] END …n_neighbors=80;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=80;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=80;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=80;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=80;, score=0.718 total time=   0.5s
[CV 9/10] END …n_neighbors=80;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=80;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=81;, score=0.725 total time=   0.4s
[CV 2/10] END …n_neighbors=81;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=81;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=81;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=81;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=81;, score=0.736 total time=   0.4s
[CV 7/10] END …n_neighbors=81;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=81;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=81;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=81;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=82;, score=0.725 total time=   0.4s
[CV 2/10] END …n_neighbors=82;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=82;, score=0.733 total time=   0.4s
[CV 4/10] END …n_neighbors=82;, score=0.728 total time=   0.4s
[CV 5/10] END …n_neighbors=82;, score=0.736 total time=   0.4s
[CV 6/10] END …n_neighbors=82;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=82;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=82;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=82;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=82;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=83;, score=0.725 total time=   0.4s
[CV 2/10] END …n_neighbors=83;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=83;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=83;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=83;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=83;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=83;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=83;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=83;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=83;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=84;, score=0.724 total time=   0.4s
[CV 2/10] END …n_neighbors=84;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=84;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=84;, score=0.729 total time=   0.4s
[CV 5/10] END …n_neighbors=84;, score=0.738 total time=   0.4s
```

```
[CV 6/10] END …n_neighbors=84;, score=0.735 total time=   0.4s
[CV 7/10] END …n_neighbors=84;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=84;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=84;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=84;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=85;, score=0.724 total time=   0.4s
[CV 2/10] END …n_neighbors=85;, score=0.717 total time=   0.4s
[CV 3/10] END …n_neighbors=85;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=85;, score=0.729 total time=   0.4s
[CV 5/10] END …n_neighbors=85;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=85;, score=0.735 total time=   0.4s
[CV 7/10] END …n_neighbors=85;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=85;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=85;, score=0.732 total time=   0.4s
[CV 10/10] END …n_neighbors=85;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=86;, score=0.726 total time=   0.5s
[CV 2/10] END …n_neighbors=86;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=86;, score=0.732 total time=   0.5s
[CV 4/10] END …n_neighbors=86;, score=0.729 total time=   0.5s
[CV 5/10] END …n_neighbors=86;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=86;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=86;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=86;, score=0.720 total time=   0.4s
[CV 9/10] END …n_neighbors=86;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=86;, score=0.732 total time=   0.6s
[CV 1/10] END …n_neighbors=87;, score=0.727 total time=   0.4s
[CV 2/10] END …n_neighbors=87;, score=0.720 total time=   0.5s
[CV 3/10] END …n_neighbors=87;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=87;, score=0.728 total time=   0.4s
[CV 5/10] END …n_neighbors=87;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=87;, score=0.735 total time=   0.4s
[CV 7/10] END …n_neighbors=87;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=87;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=87;, score=0.732 total time=   0.4s
[CV 10/10] END …n_neighbors=87;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=88;, score=0.726 total time=   0.4s
[CV 2/10] END …n_neighbors=88;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=88;, score=0.732 total time=   0.4s
[CV 4/10] END …n_neighbors=88;, score=0.728 total time=   0.4s
[CV 5/10] END …n_neighbors=88;, score=0.738 total time=   0.4s
[CV 6/10] END …n_neighbors=88;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=88;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=88;, score=0.721 total time=   0.4s
[CV 9/10] END …n_neighbors=88;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=88;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=89;, score=0.726 total time=   0.4s
[CV 2/10] END …n_neighbors=89;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=89;, score=0.730 total time=   0.4s
```

```
[CV 4/10] END …n_neighbors=89;, score=0.730 total time=   0.4s
[CV 5/10] END …n_neighbors=89;, score=0.739 total time=   0.4s
[CV 6/10] END …n_neighbors=89;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=89;, score=0.736 total time=   0.5s
[CV 8/10] END …n_neighbors=89;, score=0.721 total time=   0.5s
[CV 9/10] END …n_neighbors=89;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=89;, score=0.731 total time=   0.4s
[CV 1/10] END …n_neighbors=90;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=90;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=90;, score=0.731 total time=   0.4s
[CV 4/10] END …n_neighbors=90;, score=0.728 total time=   0.4s
[CV 5/10] END …n_neighbors=90;, score=0.739 total time=   0.4s
[CV 6/10] END …n_neighbors=90;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=90;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=90;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=90;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=90;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=91;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=91;, score=0.718 total time=   0.4s
[CV 3/10] END …n_neighbors=91;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=91;, score=0.729 total time=   0.4s
[CV 5/10] END …n_neighbors=91;, score=0.740 total time=   0.4s
[CV 6/10] END …n_neighbors=91;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=91;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=91;, score=0.720 total time=   0.4s
[CV 9/10] END …n_neighbors=91;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=91;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=92;, score=0.727 total time=   0.4s
[CV 2/10] END …n_neighbors=92;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=92;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=92;, score=0.729 total time=   0.4s
[CV 5/10] END …n_neighbors=92;, score=0.741 total time=   0.4s
[CV 6/10] END …n_neighbors=92;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=92;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=92;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=92;, score=0.735 total time=   0.4s
[CV 10/10] END …n_neighbors=92;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=93;, score=0.726 total time=   0.4s
[CV 2/10] END …n_neighbors=93;, score=0.719 total time=   0.4s
[CV 3/10] END …n_neighbors=93;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=93;, score=0.729 total time=   0.4s
[CV 5/10] END …n_neighbors=93;, score=0.740 total time=   0.4s
[CV 6/10] END …n_neighbors=93;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=93;, score=0.735 total time=   0.4s
[CV 8/10] END …n_neighbors=93;, score=0.718 total time=   0.5s
[CV 9/10] END …n_neighbors=93;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=93;, score=0.735 total time=   0.4s
[CV 1/10] END …n_neighbors=94;, score=0.727 total time=   0.5s
```

```
[CV 2/10] END …n_neighbors=94;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=94;, score=0.732 total time=   0.4s
[CV 4/10] END …n_neighbors=94;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=94;, score=0.742 total time=   0.4s
[CV 6/10] END …n_neighbors=94;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=94;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=94;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=94;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=94;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=95;, score=0.728 total time=   0.4s
[CV 2/10] END …n_neighbors=95;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=95;, score=0.730 total time=   0.4s
[CV 4/10] END …n_neighbors=95;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=95;, score=0.741 total time=   0.4s
[CV 6/10] END …n_neighbors=95;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=95;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=95;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=95;, score=0.731 total time=   0.4s
[CV 10/10] END …n_neighbors=95;, score=0.732 total time=   0.5s
[CV 1/10] END …n_neighbors=96;, score=0.727 total time=   0.5s
[CV 2/10] END …n_neighbors=96;, score=0.721 total time=   0.5s
[CV 3/10] END …n_neighbors=96;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=96;, score=0.728 total time=   0.5s
[CV 5/10] END …n_neighbors=96;, score=0.742 total time=   0.5s
[CV 6/10] END …n_neighbors=96;, score=0.736 total time=   0.4s
[CV 7/10] END …n_neighbors=96;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=96;, score=0.722 total time=   0.5s
[CV 9/10] END …n_neighbors=96;, score=0.733 total time=   0.5s
[CV 10/10] END …n_neighbors=96;, score=0.733 total time=   0.5s
[CV 1/10] END …n_neighbors=97;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=97;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=97;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=97;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=97;, score=0.741 total time=   0.4s
[CV 6/10] END …n_neighbors=97;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=97;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=97;, score=0.720 total time=   0.4s
[CV 9/10] END …n_neighbors=97;, score=0.731 total time=   0.4s
[CV 10/10] END …n_neighbors=97;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=98;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=98;, score=0.722 total time=   0.4s
[CV 3/10] END …n_neighbors=98;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=98;, score=0.727 total time=   0.4s
[CV 5/10] END …n_neighbors=98;, score=0.742 total time=   0.5s
[CV 6/10] END …n_neighbors=98;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=98;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=98;, score=0.721 total time=   0.5s
[CV 9/10] END …n_neighbors=98;, score=0.733 total time=   0.4s
```

```
[CV 10/10] END …n_neighbors=98;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=99;, score=0.727 total time=   0.4s
[CV 2/10] END …n_neighbors=99;, score=0.722 total time=   0.5s
[CV 3/10] END …n_neighbors=99;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=99;, score=0.727 total time=   0.5s
[CV 5/10] END …n_neighbors=99;, score=0.741 total time=   0.5s
[CV 6/10] END …n_neighbors=99;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=99;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=99;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=99;, score=0.732 total time=   0.5s
[CV 10/10] END …n_neighbors=99;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=100;, score=0.728 total time=   0.6s
[CV 2/10] END …n_neighbors=100;, score=0.722 total time=   0.6s
[CV 3/10] END …n_neighbors=100;, score=0.728 total time=   0.5s
[CV 4/10] END …n_neighbors=100;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=100;, score=0.741 total time=   0.5s
[CV 6/10] END …n_neighbors=100;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=100;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=100;, score=0.722 total time=   0.5s
[CV 9/10] END …n_neighbors=100;, score=0.733 total time=   0.5s
[CV 10/10] END …n_neighbors=100;, score=0.733 total time=   0.5s
[CV 1/10] END …n_neighbors=101;, score=0.727 total time=   0.5s
[CV 2/10] END …n_neighbors=101;, score=0.722 total time=   0.5s
[CV 3/10] END …n_neighbors=101;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=101;, score=0.728 total time=   0.5s
[CV 5/10] END …n_neighbors=101;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=101;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=101;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=101;, score=0.720 total time=   0.5s
[CV 9/10] END …n_neighbors=101;, score=0.731 total time=   0.5s
[CV 10/10] END …n_neighbors=101;, score=0.733 total time=   0.5s
[CV 1/10] END …n_neighbors=102;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=102;, score=0.723 total time=   0.5s
[CV 3/10] END …n_neighbors=102;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=102;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=102;, score=0.741 total time=   0.6s
[CV 6/10] END …n_neighbors=102;, score=0.733 total time=   0.9s
[CV 7/10] END …n_neighbors=102;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=102;, score=0.720 total time=   0.6s
[CV 9/10] END …n_neighbors=102;, score=0.732 total time=   1.0s
[CV 10/10] END …n_neighbors=102;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=103;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=103;, score=0.722 total time=   0.5s
[CV 3/10] END …n_neighbors=103;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=103;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=103;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=103;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=103;, score=0.736 total time=   0.5s
```

```
[CV 8/10] END …n_neighbors=103;, score=0.719 total time=   0.5s
[CV 9/10] END …n_neighbors=103;, score=0.731 total time=   0.5s
[CV 10/10] END …n_neighbors=103;, score=0.732 total time=   0.5s
[CV 1/10] END …n_neighbors=104;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=104;, score=0.721 total time=   1.0s
[CV 3/10] END …n_neighbors=104;, score=0.728 total time=   0.5s
[CV 4/10] END …n_neighbors=104;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=104;, score=0.741 total time=   0.4s
[CV 6/10] END …n_neighbors=104;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=104;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=104;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=104;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=104;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=105;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=105;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=105;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=105;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=105;, score=0.739 total time=   0.4s
[CV 6/10] END …n_neighbors=105;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=105;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=105;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=105;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=105;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=106;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=106;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=106;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=106;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=106;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=106;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=106;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=106;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=106;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=106;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=107;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=107;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=107;, score=0.728 total time=   0.4s
[CV 4/10] END …n_neighbors=107;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=107;, score=0.740 total time=   0.4s
[CV 6/10] END …n_neighbors=107;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=107;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=107;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=107;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=107;, score=0.732 total time=   0.5s
[CV 1/10] END …n_neighbors=108;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=108;, score=0.722 total time=   0.4s
[CV 3/10] END …n_neighbors=108;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=108;, score=0.726 total time=   0.4s
[CV 5/10] END …n_neighbors=108;, score=0.740 total time=   0.4s
```

```
[CV 6/10] END …n_neighbors=108;, score=0.732 total time=   0.4s
[CV 7/10] END …n_neighbors=108;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=108;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=108;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=108;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=109;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=109;, score=0.722 total time=   0.4s
[CV 3/10] END …n_neighbors=109;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=109;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=109;, score=0.740 total time=   0.4s
[CV 6/10] END …n_neighbors=109;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=109;, score=0.736 total time=   0.4s
[CV 8/10] END …n_neighbors=109;, score=0.719 total time=   0.4s
[CV 9/10] END …n_neighbors=109;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=109;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=110;, score=0.729 total time=   0.4s
[CV 2/10] END …n_neighbors=110;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=110;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=110;, score=0.724 total time=   0.4s
[CV 5/10] END …n_neighbors=110;, score=0.740 total time=   0.4s
[CV 6/10] END …n_neighbors=110;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=110;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=110;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=110;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=110;, score=0.733 total time=   0.4s
[CV 1/10] END …n_neighbors=111;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=111;, score=0.720 total time=   0.4s
[CV 3/10] END …n_neighbors=111;, score=0.729 total time=   0.4s
[CV 4/10] END …n_neighbors=111;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=111;, score=0.739 total time=   0.4s
[CV 6/10] END …n_neighbors=111;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=111;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=111;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=111;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=111;, score=0.734 total time=   0.4s
[CV 1/10] END …n_neighbors=112;, score=0.730 total time=   0.4s
[CV 2/10] END …n_neighbors=112;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=112;, score=0.728 total time=   0.4s
[CV 4/10] END …n_neighbors=112;, score=0.725 total time=   0.4s
[CV 5/10] END …n_neighbors=112;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=112;, score=0.733 total time=   0.4s
[CV 7/10] END …n_neighbors=112;, score=0.737 total time=   0.4s
[CV 8/10] END …n_neighbors=112;, score=0.718 total time=   0.4s
[CV 9/10] END …n_neighbors=112;, score=0.733 total time=   0.4s
[CV 10/10] END …n_neighbors=112;, score=0.732 total time=   0.4s
[CV 1/10] END …n_neighbors=113;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=113;, score=0.721 total time=   0.4s
[CV 3/10] END …n_neighbors=113;, score=0.729 total time=   0.4s
```

```
[CV 4/10] END …n_neighbors=113;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=113;, score=0.741 total time=   0.4s
[CV 6/10] END …n_neighbors=113;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=113;, score=0.738 total time=   0.4s
[CV 8/10] END …n_neighbors=113;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=113;, score=0.734 total time=   0.4s
[CV 10/10] END …n_neighbors=113;, score=0.733 total time=   0.5s
[CV 1/10] END …n_neighbors=114;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=114;, score=0.721 total time=   0.5s
[CV 3/10] END …n_neighbors=114;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=114;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=114;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=114;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=114;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=114;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=114;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=114;, score=0.731 total time=   0.5s
[CV 1/10] END …n_neighbors=115;, score=0.729 total time=   0.6s
[CV 2/10] END …n_neighbors=115;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=115;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=115;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=115;, score=0.740 total time=   0.4s
[CV 6/10] END …n_neighbors=115;, score=0.734 total time=   0.4s
[CV 7/10] END …n_neighbors=115;, score=0.739 total time=   0.4s
[CV 8/10] END …n_neighbors=115;, score=0.717 total time=   0.4s
[CV 9/10] END …n_neighbors=115;, score=0.736 total time=   0.4s
[CV 10/10] END …n_neighbors=115;, score=0.732 total time=   0.5s
[CV 1/10] END …n_neighbors=116;, score=0.728 total time=   0.6s
[CV 2/10] END …n_neighbors=116;, score=0.721 total time=   0.6s
[CV 3/10] END …n_neighbors=116;, score=0.728 total time=   0.6s
[CV 4/10] END …n_neighbors=116;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=116;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=116;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=116;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=116;, score=0.718 total time=   0.5s
[CV 9/10] END …n_neighbors=116;, score=0.733 total time=   0.5s
[CV 10/10] END …n_neighbors=116;, score=0.731 total time=   0.5s
[CV 1/10] END …n_neighbors=117;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=117;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=117;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=117;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=117;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=117;, score=0.733 total time=   0.6s
[CV 7/10] END …n_neighbors=117;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=117;, score=0.718 total time=   0.5s
[CV 9/10] END …n_neighbors=117;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=117;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=118;, score=0.727 total time=   0.5s
```

```
[CV 2/10] END …n_neighbors=118;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=118;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=118;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=118;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=118;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=118;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=118;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=118;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=118;, score=0.733 total time=   0.5s
[CV 1/10] END …n_neighbors=119;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=119;, score=0.720 total time=   0.5s
[CV 3/10] END …n_neighbors=119;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=119;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=119;, score=0.740 total time=   1.1s
[CV 6/10] END …n_neighbors=119;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=119;, score=0.739 total time=   0.6s
[CV 8/10] END …n_neighbors=119;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=119;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=119;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=120;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=120;, score=0.720 total time=   0.5s
[CV 3/10] END …n_neighbors=120;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=120;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=120;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=120;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=120;, score=0.739 total time=   0.5s
[CV 8/10] END …n_neighbors=120;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=120;, score=0.733 total time=   0.6s
[CV 10/10] END …n_neighbors=120;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=121;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=121;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=121;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=121;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=121;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=121;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=121;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=121;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=121;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=121;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=122;, score=0.728 total time=   0.5s
[CV 2/10] END …n_neighbors=122;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=122;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=122;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=122;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=122;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=122;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=122;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=122;, score=0.734 total time=   0.5s
```

```
[CV 10/10] END …n_neighbors=122;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=123;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=123;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=123;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=123;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=123;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=123;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=123;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=123;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=123;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=123;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=124;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=124;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=124;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=124;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=124;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=124;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=124;, score=0.739 total time=   0.5s
[CV 8/10] END …n_neighbors=124;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=124;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=124;, score=0.736 total time=   0.7s
[CV 1/10] END …n_neighbors=125;, score=0.729 total time=   0.7s
[CV 2/10] END …n_neighbors=125;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=125;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=125;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=125;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=125;, score=0.732 total time=   0.6s
[CV 7/10] END …n_neighbors=125;, score=0.739 total time=   0.6s
[CV 8/10] END …n_neighbors=125;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=125;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=125;, score=0.737 total time=   0.6s
[CV 1/10] END …n_neighbors=126;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=126;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=126;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=126;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=126;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=126;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=126;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=126;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=126;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=126;, score=0.737 total time=   0.5s
[CV 1/10] END …n_neighbors=127;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=127;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=127;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=127;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=127;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=127;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=127;, score=0.739 total time=   0.5s
```

```
[CV 8/10] END …n_neighbors=127;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=127;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=127;, score=0.737 total time=   0.5s
[CV 1/10] END …n_neighbors=128;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=128;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=128;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=128;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=128;, score=0.737 total time=   0.5s
[CV 6/10] END …n_neighbors=128;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=128;, score=0.739 total time=   0.5s
[CV 8/10] END …n_neighbors=128;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=128;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=128;, score=0.737 total time=   0.5s
[CV 1/10] END …n_neighbors=129;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=129;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=129;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=129;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=129;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=129;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=129;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=129;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=129;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=129;, score=0.737 total time=   0.5s
[CV 1/10] END …n_neighbors=130;, score=0.728 total time=   0.6s
[CV 2/10] END …n_neighbors=130;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=130;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=130;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=130;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=130;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=130;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=130;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=130;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=130;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=131;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=131;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=131;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=131;, score=0.727 total time=   0.5s
[CV 5/10] END …n_neighbors=131;, score=0.737 total time=   0.5s
[CV 6/10] END …n_neighbors=131;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=131;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=131;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=131;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=131;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=132;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=132;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=132;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=132;, score=0.724 total time=   0.5s
[CV 5/10] END …n_neighbors=132;, score=0.737 total time=   0.5s
```

```
[CV 6/10] END …n_neighbors=132;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=132;, score=0.739 total time=   0.5s
[CV 8/10] END …n_neighbors=132;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=132;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=132;, score=0.737 total time=   0.5s
[CV 1/10] END …n_neighbors=133;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=133;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=133;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=133;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=133;, score=0.737 total time=   0.5s
[CV 6/10] END …n_neighbors=133;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=133;, score=0.739 total time=   0.5s
[CV 8/10] END …n_neighbors=133;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=133;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=133;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=134;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=134;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=134;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=134;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=134;, score=0.737 total time=   0.5s
[CV 6/10] END …n_neighbors=134;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=134;, score=0.739 total time=   0.5s
[CV 8/10] END …n_neighbors=134;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=134;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=134;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=135;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=135;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=135;, score=0.729 total time=   0.5s
[CV 4/10] END …n_neighbors=135;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=135;, score=0.737 total time=   0.5s
[CV 6/10] END …n_neighbors=135;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=135;, score=0.739 total time=   0.5s
[CV 8/10] END …n_neighbors=135;, score=0.715 total time=   0.5s
[CV 9/10] END …n_neighbors=135;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=135;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=136;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=136;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=136;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=136;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=136;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=136;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=136;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=136;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=136;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=136;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=137;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=137;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=137;, score=0.730 total time=   0.5s
```

```
[CV 4/10] END …n_neighbors=137;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=137;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=137;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=137;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=137;, score=0.718 total time=   0.5s
[CV 9/10] END …n_neighbors=137;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=137;, score=0.737 total time=   0.5s
[CV 1/10] END …n_neighbors=138;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=138;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=138;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=138;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=138;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=138;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=138;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=138;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=138;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=138;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=139;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=139;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=139;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=139;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=139;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=139;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=139;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=139;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=139;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=139;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=140;, score=0.730 total time=   0.6s
[CV 2/10] END …n_neighbors=140;, score=0.717 total time=   0.5s
[CV 3/10] END …n_neighbors=140;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=140;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=140;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=140;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=140;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=140;, score=0.718 total time=   0.5s
[CV 9/10] END …n_neighbors=140;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=140;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=141;, score=0.729 total time=   0.5s
[CV 2/10] END …n_neighbors=141;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=141;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=141;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=141;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=141;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=141;, score=0.738 total time=   0.5s
[CV 8/10] END …n_neighbors=141;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=141;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=141;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=142;, score=0.730 total time=   0.5s
```

```
[CV 2/10] END …n_neighbors=142;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=142;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=142;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=142;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=142;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=142;, score=0.736 total time=   0.5s
[CV 8/10] END …n_neighbors=142;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=142;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=142;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=143;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=143;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=143;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=143;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=143;, score=0.738 total time=   0.5s
[CV 6/10] END …n_neighbors=143;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=143;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=143;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=143;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=143;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=144;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=144;, score=0.716 total time=   0.5s
[CV 3/10] END …n_neighbors=144;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=144;, score=0.724 total time=   0.5s
[CV 5/10] END …n_neighbors=144;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=144;, score=0.730 total time=   0.5s
[CV 7/10] END …n_neighbors=144;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=144;, score=0.715 total time=   0.5s
[CV 9/10] END …n_neighbors=144;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=144;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=145;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=145;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=145;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=145;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=145;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=145;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=145;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=145;, score=0.715 total time=   0.5s
[CV 9/10] END …n_neighbors=145;, score=0.736 total time=   0.5s
[CV 10/10] END …n_neighbors=145;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=146;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=146;, score=0.717 total time=   0.5s
[CV 3/10] END …n_neighbors=146;, score=0.732 total time=   0.5s
[CV 4/10] END …n_neighbors=146;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=146;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=146;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=146;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=146;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=146;, score=0.735 total time=   0.6s
```

```
[CV 10/10] END …n_neighbors=146;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=147;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=147;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=147;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=147;, score=0.725 total time=   1.0s
[CV 5/10] END …n_neighbors=147;, score=0.741 total time=   0.6s
[CV 6/10] END …n_neighbors=147;, score=0.732 total time=   0.6s
[CV 7/10] END …n_neighbors=147;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=147;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=147;, score=0.737 total time=   0.6s
[CV 10/10] END …n_neighbors=147;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=148;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=148;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=148;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=148;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=148;, score=0.741 total time=   0.5s
[CV 6/10] END …n_neighbors=148;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=148;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=148;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=148;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=148;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=149;, score=0.733 total time=   0.5s
[CV 2/10] END …n_neighbors=149;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=149;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=149;, score=0.724 total time=   0.5s
[CV 5/10] END …n_neighbors=149;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=149;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=149;, score=0.736 total time=   0.5s
[CV 8/10] END …n_neighbors=149;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=149;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=149;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=150;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=150;, score=0.720 total time=   0.5s
[CV 3/10] END …n_neighbors=150;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=150;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=150;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=150;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=150;, score=0.735 total time=   0.5s
[CV 8/10] END …n_neighbors=150;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=150;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=150;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=151;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=151;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=151;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=151;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=151;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=151;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=151;, score=0.736 total time=   0.5s
```

```
[CV 8/10] END …n_neighbors=151;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=151;, score=0.736 total time=   0.5s
[CV 10/10] END …n_neighbors=151;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=152;, score=0.732 total time=   0.5s
[CV 2/10] END …n_neighbors=152;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=152;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=152;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=152;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=152;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=152;, score=0.735 total time=   0.5s
[CV 8/10] END …n_neighbors=152;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=152;, score=0.736 total time=   0.5s
[CV 10/10] END …n_neighbors=152;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=153;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=153;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=153;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=153;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=153;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=153;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=153;, score=0.735 total time=   0.5s
[CV 8/10] END …n_neighbors=153;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=153;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=153;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=154;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=154;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=154;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=154;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=154;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=154;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=154;, score=0.734 total time=   0.5s
[CV 8/10] END …n_neighbors=154;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=154;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=154;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=155;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=155;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=155;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=155;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=155;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=155;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=155;, score=0.735 total time=   0.5s
[CV 8/10] END …n_neighbors=155;, score=0.718 total time=   0.5s
[CV 9/10] END …n_neighbors=155;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=155;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=156;, score=0.730 total time=   0.5s
[CV 2/10] END …n_neighbors=156;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=156;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=156;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=156;, score=0.740 total time=   0.5s
```

```
[CV 6/10] END …n_neighbors=156;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=156;, score=0.734 total time=   0.5s
[CV 8/10] END …n_neighbors=156;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=156;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=156;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=157;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=157;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=157;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=157;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=157;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=157;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=157;, score=0.735 total time=   0.5s
[CV 8/10] END …n_neighbors=157;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=157;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=157;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=158;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=158;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=158;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=158;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=158;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=158;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=158;, score=0.735 total time=   0.5s
[CV 8/10] END …n_neighbors=158;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=158;, score=0.737 total time=   0.5s
[CV 10/10] END …n_neighbors=158;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=159;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=159;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=159;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=159;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=159;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=159;, score=0.732 total time=   0.6s
[CV 7/10] END …n_neighbors=159;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=159;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=159;, score=0.736 total time=   0.6s
[CV 10/10] END …n_neighbors=159;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=160;, score=0.730 total time=   0.6s
[CV 2/10] END …n_neighbors=160;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=160;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=160;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=160;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=160;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=160;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=160;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=160;, score=0.736 total time=   0.6s
[CV 10/10] END …n_neighbors=160;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=161;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=161;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=161;, score=0.729 total time=   0.5s
```

```
[CV 4/10] END …n_neighbors=161;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=161;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=161;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=161;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=161;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=161;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=161;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=162;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=162;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=162;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=162;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=162;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=162;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=162;, score=0.736 total time=   0.5s
[CV 8/10] END …n_neighbors=162;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=162;, score=0.735 total time=   0.5s
[CV 10/10] END …n_neighbors=162;, score=0.734 total time=   0.5s
[CV 1/10] END …n_neighbors=163;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=163;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=163;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=163;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=163;, score=0.739 total time=   0.5s
[CV 6/10] END …n_neighbors=163;, score=0.731 total time=   0.5s
[CV 7/10] END …n_neighbors=163;, score=0.735 total time=   0.5s
[CV 8/10] END …n_neighbors=163;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=163;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=163;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=164;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=164;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=164;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=164;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=164;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=164;, score=0.732 total time=   0.5s
[CV 7/10] END …n_neighbors=164;, score=0.736 total time=   0.5s
[CV 8/10] END …n_neighbors=164;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=164;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=164;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=165;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=165;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=165;, score=0.730 total time=   0.5s
[CV 4/10] END …n_neighbors=165;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=165;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=165;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=165;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=165;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=165;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=165;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=166;, score=0.731 total time=   0.5s
```

```
[CV 2/10] END …n_neighbors=166;, score=0.717 total time=   0.5s
[CV 3/10] END …n_neighbors=166;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=166;, score=0.725 total time=   0.6s
[CV 5/10] END …n_neighbors=166;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=166;, score=0.732 total time=   0.6s
[CV 7/10] END …n_neighbors=166;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=166;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=166;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=166;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=167;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=167;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=167;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=167;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=167;, score=0.741 total time=   0.5s
[CV 6/10] END …n_neighbors=167;, score=0.734 total time=   0.5s
[CV 7/10] END …n_neighbors=167;, score=0.736 total time=   0.5s
[CV 8/10] END …n_neighbors=167;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=167;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=167;, score=0.736 total time=   0.5s
[CV 1/10] END …n_neighbors=168;, score=0.732 total time=   0.5s
[CV 2/10] END …n_neighbors=168;, score=0.719 total time=   0.5s
[CV 3/10] END …n_neighbors=168;, score=0.731 total time=   0.5s
[CV 4/10] END …n_neighbors=168;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=168;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=168;, score=0.733 total time=   0.5s
[CV 7/10] END …n_neighbors=168;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=168;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=168;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=168;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=169;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=169;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=169;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=169;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=169;, score=0.741 total time=   0.5s
[CV 6/10] END …n_neighbors=169;, score=0.734 total time=   0.6s
[CV 7/10] END …n_neighbors=169;, score=0.737 total time=   0.5s
[CV 8/10] END …n_neighbors=169;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=169;, score=0.734 total time=   0.5s
[CV 10/10] END …n_neighbors=169;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=170;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=170;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=170;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=170;, score=0.726 total time=   0.5s
[CV 5/10] END …n_neighbors=170;, score=0.740 total time=   0.5s
[CV 6/10] END …n_neighbors=170;, score=0.733 total time=   0.6s
[CV 7/10] END …n_neighbors=170;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=170;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=170;, score=0.734 total time=   0.6s
```

```
[CV 10/10] END …n_neighbors=170;, score=0.735 total time=   0.5s
[CV 1/10] END …n_neighbors=171;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=171;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=171;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=171;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=171;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=171;, score=0.733 total time=   0.6s
[CV 7/10] END …n_neighbors=171;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=171;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=171;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=171;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=172;, score=0.731 total time=   0.5s
[CV 2/10] END …n_neighbors=172;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=172;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=172;, score=0.725 total time=   0.5s
[CV 5/10] END …n_neighbors=172;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=172;, score=0.732 total time=   0.6s
[CV 7/10] END …n_neighbors=172;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=172;, score=0.717 total time=   0.5s
[CV 9/10] END …n_neighbors=172;, score=0.733 total time=   0.6s
[CV 10/10] END …n_neighbors=172;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=173;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=173;, score=0.717 total time=   0.5s
[CV 3/10] END …n_neighbors=173;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=173;, score=0.725 total time=   0.6s
[CV 5/10] END …n_neighbors=173;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=173;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=173;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=173;, score=0.716 total time=   0.5s
[CV 9/10] END …n_neighbors=173;, score=0.733 total time=   0.5s
[CV 10/10] END …n_neighbors=173;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=174;, score=0.730 total time=   0.6s
[CV 2/10] END …n_neighbors=174;, score=0.718 total time=   0.5s
[CV 3/10] END …n_neighbors=174;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=174;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=174;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=174;, score=0.732 total time=   0.6s
[CV 7/10] END …n_neighbors=174;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=174;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=174;, score=0.732 total time=   0.6s
[CV 10/10] END …n_neighbors=174;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=175;, score=0.730 total time=   0.6s
[CV 2/10] END …n_neighbors=175;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=175;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=175;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=175;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=175;, score=0.732 total time=   0.6s
[CV 7/10] END …n_neighbors=175;, score=0.736 total time=   0.6s
```

```
[CV 8/10] END …n_neighbors=175;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=175;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=175;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=176;, score=0.730 total time=   0.6s
[CV 2/10] END …n_neighbors=176;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=176;, score=0.730 total time=   0.6s
[CV 4/10] END …n_neighbors=176;, score=0.727 total time=   0.5s
[CV 5/10] END …n_neighbors=176;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=176;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=176;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=176;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=176;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=176;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=177;, score=0.730 total time=   0.6s
[CV 2/10] END …n_neighbors=177;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=177;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=177;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=177;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=177;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=177;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=177;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=177;, score=0.733 total time=   0.6s
[CV 10/10] END …n_neighbors=177;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=178;, score=0.729 total time=   0.6s
[CV 2/10] END …n_neighbors=178;, score=0.717 total time=   0.6s
[CV 3/10] END …n_neighbors=178;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=178;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=178;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=178;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=178;, score=0.734 total time=   0.6s
[CV 8/10] END …n_neighbors=178;, score=0.714 total time=   0.6s
[CV 9/10] END …n_neighbors=178;, score=0.732 total time=   0.6s
[CV 10/10] END …n_neighbors=178;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=179;, score=0.729 total time=   0.6s
[CV 2/10] END …n_neighbors=179;, score=0.716 total time=   0.6s
[CV 3/10] END …n_neighbors=179;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=179;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=179;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=179;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=179;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=179;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=179;, score=0.733 total time=   0.6s
[CV 10/10] END …n_neighbors=179;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=180;, score=0.729 total time=   0.6s
[CV 2/10] END …n_neighbors=180;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=180;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=180;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=180;, score=0.739 total time=   0.6s
```

```
[CV 6/10] END …n_neighbors=180;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=180;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=180;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=180;, score=0.732 total time=   0.6s
[CV 10/10] END …n_neighbors=180;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=181;, score=0.729 total time=   0.6s
[CV 2/10] END …n_neighbors=181;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=181;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=181;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=181;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=181;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=181;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=181;, score=0.713 total time=   0.6s
[CV 9/10] END …n_neighbors=181;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=181;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=182;, score=0.729 total time=   0.6s
[CV 2/10] END …n_neighbors=182;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=182;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=182;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=182;, score=0.741 total time=   0.6s
[CV 6/10] END …n_neighbors=182;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=182;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=182;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=182;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=182;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=183;, score=0.730 total time=   0.6s
[CV 2/10] END …n_neighbors=183;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=183;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=183;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=183;, score=0.741 total time=   0.6s
[CV 6/10] END …n_neighbors=183;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=183;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=183;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=183;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=183;, score=0.736 total time=   0.6s
[CV 1/10] END …n_neighbors=184;, score=0.729 total time=   0.6s
[CV 2/10] END …n_neighbors=184;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=184;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=184;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=184;, score=0.741 total time=   0.6s
[CV 6/10] END …n_neighbors=184;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=184;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=184;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=184;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=184;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=185;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=185;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=185;, score=0.731 total time=   0.6s
```

```
[CV 4/10] END …n_neighbors=185;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=185;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=185;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=185;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=185;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=185;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=185;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=186;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=186;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=186;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=186;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=186;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=186;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=186;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=186;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=186;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=186;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=187;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=187;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=187;, score=0.731 total time=   0.7s
[CV 4/10] END …n_neighbors=187;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=187;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=187;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=187;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=187;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=187;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=187;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=188;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=188;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=188;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=188;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=188;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=188;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=188;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=188;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=188;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=188;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=189;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=189;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=189;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=189;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=189;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=189;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=189;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=189;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=189;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=189;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=190;, score=0.731 total time=   0.6s
```

```
[CV 2/10] END …n_neighbors=190;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=190;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=190;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=190;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=190;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=190;, score=0.734 total time=   0.6s
[CV 8/10] END …n_neighbors=190;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=190;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=190;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=191;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=191;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=191;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=191;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=191;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=191;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=191;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=191;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=191;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=191;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=192;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=192;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=192;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=192;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=192;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=192;, score=0.730 total time=   0.7s
[CV 7/10] END …n_neighbors=192;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=192;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=192;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=192;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=193;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=193;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=193;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=193;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=193;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=193;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=193;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=193;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=193;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=193;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=194;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=194;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=194;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=194;, score=0.729 total time=   0.6s
[CV 5/10] END …n_neighbors=194;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=194;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=194;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=194;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=194;, score=0.736 total time=   0.6s
```

```
[CV 10/10] END …n_neighbors=194;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=195;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=195;, score=0.718 total time=   0.8s
[CV 3/10] END …n_neighbors=195;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=195;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=195;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=195;, score=0.731 total time=   0.7s
[CV 7/10] END …n_neighbors=195;, score=0.736 total time=   1.0s
[CV 8/10] END …n_neighbors=195;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=195;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=195;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=196;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=196;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=196;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=196;, score=0.729 total time=   0.6s
[CV 5/10] END …n_neighbors=196;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=196;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=196;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=196;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=196;, score=0.736 total time=   0.6s
[CV 10/10] END …n_neighbors=196;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=197;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=197;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=197;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=197;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=197;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=197;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=197;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=197;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=197;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=197;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=198;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=198;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=198;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=198;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=198;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=198;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=198;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=198;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=198;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=198;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=199;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=199;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=199;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=199;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=199;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=199;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=199;, score=0.737 total time=   0.6s
```

```
[CV 8/10] END …n_neighbors=199;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=199;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=199;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=200;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=200;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=200;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=200;, score=0.728 total time=   1.0s
[CV 5/10] END …n_neighbors=200;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=200;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=200;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=200;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=200;, score=0.735 total time=   0.7s
[CV 10/10] END …n_neighbors=200;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=201;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=201;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=201;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=201;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=201;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=201;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=201;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=201;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=201;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=201;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=202;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=202;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=202;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=202;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=202;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=202;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=202;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=202;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=202;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=202;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=203;, score=0.733 total time=   0.6s
[CV 2/10] END …n_neighbors=203;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=203;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=203;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=203;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=203;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=203;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=203;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=203;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=203;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=204;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=204;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=204;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=204;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=204;, score=0.739 total time=   0.6s
```

```
[CV 6/10] END …n_neighbors=204;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=204;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=204;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=204;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=204;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=205;, score=0.733 total time=   0.6s
[CV 2/10] END …n_neighbors=205;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=205;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=205;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=205;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=205;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=205;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=205;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=205;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=205;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=206;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=206;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=206;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=206;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=206;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=206;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=206;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=206;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=206;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=206;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=207;, score=0.733 total time=   0.6s
[CV 2/10] END …n_neighbors=207;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=207;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=207;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=207;, score=0.740 total time=   0.6s
[CV 6/10] END …n_neighbors=207;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=207;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=207;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=207;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=207;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=208;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=208;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=208;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=208;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=208;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=208;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=208;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=208;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=208;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=208;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=209;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=209;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=209;, score=0.731 total time=   0.6s
```

```
[CV 4/10] END …n_neighbors=209;, score=0.729 total time=   0.6s
[CV 5/10] END …n_neighbors=209;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=209;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=209;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=209;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=209;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=209;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=210;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=210;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=210;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=210;, score=0.729 total time=   0.6s
[CV 5/10] END …n_neighbors=210;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=210;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=210;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=210;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=210;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=210;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=211;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=211;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=211;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=211;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=211;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=211;, score=0.731 total time=   0.6s
[CV 7/10] END …n_neighbors=211;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=211;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=211;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=211;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=212;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=212;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=212;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=212;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=212;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=212;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=212;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=212;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=212;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=212;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=213;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=213;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=213;, score=0.731 total time=   0.6s
[CV 4/10] END …n_neighbors=213;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=213;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=213;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=213;, score=0.739 total time=   0.6s
[CV 8/10] END …n_neighbors=213;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=213;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=213;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=214;, score=0.731 total time=   0.6s
```

```
[CV 2/10] END …n_neighbors=214;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=214;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=214;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=214;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=214;, score=0.730 total time=   0.6s
[CV 7/10] END …n_neighbors=214;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=214;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=214;, score=0.736 total time=   0.7s
[CV 10/10] END …n_neighbors=214;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=215;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=215;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=215;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=215;, score=0.729 total time=   0.6s
[CV 5/10] END …n_neighbors=215;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=215;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=215;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=215;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=215;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=215;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=216;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=216;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=216;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=216;, score=0.728 total time=   0.6s
[CV 5/10] END …n_neighbors=216;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=216;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=216;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=216;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=216;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=216;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=217;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=217;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=217;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=217;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=217;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=217;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=217;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=217;, score=0.718 total time=   0.6s
[CV 9/10] END …n_neighbors=217;, score=0.735 total time=   0.6s
[CV 10/10] END …n_neighbors=217;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=218;, score=0.731 total time=   0.6s
[CV 2/10] END …n_neighbors=218;, score=0.720 total time=   0.6s
[CV 3/10] END …n_neighbors=218;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=218;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=218;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=218;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=218;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=218;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=218;, score=0.734 total time=   0.7s
```

```
[CV 10/10] END …n_neighbors=218;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=219;, score=0.732 total time=   0.8s
[CV 2/10] END …n_neighbors=219;, score=0.720 total time=   0.6s
[CV 3/10] END …n_neighbors=219;, score=0.732 total time=   0.8s
[CV 4/10] END …n_neighbors=219;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=219;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=219;, score=0.727 total time=   0.7s
[CV 7/10] END …n_neighbors=219;, score=0.737 total time=   0.7s
[CV 8/10] END …n_neighbors=219;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=219;, score=0.735 total time=   0.7s
[CV 10/10] END …n_neighbors=219;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=220;, score=0.730 total time=   0.7s
[CV 2/10] END …n_neighbors=220;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=220;, score=0.732 total time=   0.7s
[CV 4/10] END …n_neighbors=220;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=220;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=220;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=220;, score=0.738 total time=   0.7s
[CV 8/10] END …n_neighbors=220;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=220;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=220;, score=0.735 total time=   0.8s
[CV 1/10] END …n_neighbors=221;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=221;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=221;, score=0.732 total time=   0.7s
[CV 4/10] END …n_neighbors=221;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=221;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=221;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=221;, score=0.737 total time=   0.7s
[CV 8/10] END …n_neighbors=221;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=221;, score=0.735 total time=   0.7s
[CV 10/10] END …n_neighbors=221;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=222;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=222;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=222;, score=0.733 total time=   0.8s
[CV 4/10] END …n_neighbors=222;, score=0.726 total time=   0.8s
[CV 5/10] END …n_neighbors=222;, score=0.739 total time=   0.8s
[CV 6/10] END …n_neighbors=222;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=222;, score=0.737 total time=   0.7s
[CV 8/10] END …n_neighbors=222;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=222;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=222;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=223;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=223;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=223;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=223;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=223;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=223;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=223;, score=0.737 total time=   0.7s
```

```
[CV 8/10] END …n_neighbors=223;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=223;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=223;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=224;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=224;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=224;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=224;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=224;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=224;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=224;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=224;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=224;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=224;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=225;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=225;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=225;, score=0.732 total time=   0.7s
[CV 4/10] END …n_neighbors=225;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=225;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=225;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=225;, score=0.738 total time=   0.6s
[CV 8/10] END …n_neighbors=225;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=225;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=225;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=226;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=226;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=226;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=226;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=226;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=226;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=226;, score=0.738 total time=   0.7s
[CV 8/10] END …n_neighbors=226;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=226;, score=0.735 total time=   0.7s
[CV 10/10] END …n_neighbors=226;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=227;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=227;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=227;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=227;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=227;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=227;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=227;, score=0.737 total time=   0.7s
[CV 8/10] END …n_neighbors=227;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=227;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=227;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=228;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=228;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=228;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=228;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=228;, score=0.739 total time=   0.7s
```

```
[CV 6/10] END …n_neighbors=228;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=228;, score=0.737 total time=   0.7s
[CV 8/10] END …n_neighbors=228;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=228;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=228;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=229;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=229;, score=0.718 total time=   0.6s
[CV 3/10] END …n_neighbors=229;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=229;, score=0.725 total time=   0.6s
[CV 5/10] END …n_neighbors=229;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=229;, score=0.728 total time=   0.6s
[CV 7/10] END …n_neighbors=229;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=229;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=229;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=229;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=230;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=230;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=230;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=230;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=230;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=230;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=230;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=230;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=230;, score=0.735 total time=   0.7s
[CV 10/10] END …n_neighbors=230;, score=0.735 total time=   0.6s
[CV 1/10] END …n_neighbors=231;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=231;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=231;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=231;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=231;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=231;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=231;, score=0.738 total time=   0.7s
[CV 8/10] END …n_neighbors=231;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=231;, score=0.735 total time=   0.7s
[CV 10/10] END …n_neighbors=231;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=232;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=232;, score=0.718 total time=   0.8s
[CV 3/10] END …n_neighbors=232;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=232;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=232;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=232;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=232;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=232;, score=0.717 total time=   0.6s
[CV 9/10] END …n_neighbors=232;, score=0.735 total time=   0.7s
[CV 10/10] END …n_neighbors=232;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=233;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=233;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=233;, score=0.733 total time=   0.7s
```

```
[CV 4/10] END …n_neighbors=233;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=233;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=233;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=233;, score=0.737 total time=   0.6s
[CV 8/10] END …n_neighbors=233;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=233;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=233;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=234;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=234;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=234;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=234;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=234;, score=0.737 total time=   0.6s
[CV 6/10] END …n_neighbors=234;, score=0.728 total time=   0.6s
[CV 7/10] END …n_neighbors=234;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=234;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=234;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=234;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=235;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=235;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=235;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=235;, score=0.725 total time=   0.6s
[CV 5/10] END …n_neighbors=235;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=235;, score=0.729 total time=   0.6s
[CV 7/10] END …n_neighbors=235;, score=0.737 total time=   0.7s
[CV 8/10] END …n_neighbors=235;, score=0.715 total time=   0.6s
[CV 9/10] END …n_neighbors=235;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=235;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=236;, score=0.732 total time=   0.6s
[CV 2/10] END …n_neighbors=236;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=236;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=236;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=236;, score=0.738 total time=   0.6s
[CV 6/10] END …n_neighbors=236;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=236;, score=0.736 total time=   0.6s
[CV 8/10] END …n_neighbors=236;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=236;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=236;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=237;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=237;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=237;, score=0.732 total time=   0.6s
[CV 4/10] END …n_neighbors=237;, score=0.725 total time=   0.6s
[CV 5/10] END …n_neighbors=237;, score=0.739 total time=   0.6s
[CV 6/10] END …n_neighbors=237;, score=0.728 total time=   0.6s
[CV 7/10] END …n_neighbors=237;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=237;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=237;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=237;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=238;, score=0.731 total time=   0.6s
```

```
[CV 2/10] END …n_neighbors=238;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=238;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=238;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=238;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=238;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=238;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=238;, score=0.716 total time=   0.6s
[CV 9/10] END …n_neighbors=238;, score=0.734 total time=   0.6s
[CV 10/10] END …n_neighbors=238;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=239;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=239;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=239;, score=0.732 total time=   0.7s
[CV 4/10] END …n_neighbors=239;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=239;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=239;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=239;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=239;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=239;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=239;, score=0.733 total time=   0.6s
[CV 1/10] END …n_neighbors=240;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=240;, score=0.720 total time=   0.6s
[CV 3/10] END …n_neighbors=240;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=240;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=240;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=240;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=240;, score=0.735 total time=   0.6s
[CV 8/10] END …n_neighbors=240;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=240;, score=0.733 total time=   0.6s
[CV 10/10] END …n_neighbors=240;, score=0.734 total time=   0.6s
[CV 1/10] END …n_neighbors=241;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=241;, score=0.719 total time=   0.6s
[CV 3/10] END …n_neighbors=241;, score=0.733 total time=   0.6s
[CV 4/10] END …n_neighbors=241;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=241;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=241;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=241;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=241;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=241;, score=0.733 total time=   0.6s
[CV 10/10] END …n_neighbors=241;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=242;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=242;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=242;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=242;, score=0.727 total time=   0.6s
[CV 5/10] END …n_neighbors=242;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=242;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=242;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=242;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=242;, score=0.733 total time=   0.6s
```

```
[CV 10/10] END …n_neighbors=242;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=243;, score=0.730 total time=   0.7s
[CV 2/10] END …n_neighbors=243;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=243;, score=0.732 total time=   0.7s
[CV 4/10] END …n_neighbors=243;, score=0.726 total time=   0.6s
[CV 5/10] END …n_neighbors=243;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=243;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=243;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=243;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=243;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=243;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=244;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=244;, score=0.720 total time=   0.6s
[CV 3/10] END …n_neighbors=244;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=244;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=244;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=244;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=244;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=244;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=244;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=244;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=245;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=245;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=245;, score=0.732 total time=   0.7s
[CV 4/10] END …n_neighbors=245;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=245;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=245;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=245;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=245;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=245;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=245;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=246;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=246;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=246;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=246;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=246;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=246;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=246;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=246;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=246;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=246;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=247;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=247;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=247;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=247;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=247;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=247;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=247;, score=0.735 total time=   0.7s
```

```
[CV 8/10] END …n_neighbors=247;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=247;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=247;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=248;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=248;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=248;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=248;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=248;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=248;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=248;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=248;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=248;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=248;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=249;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=249;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=249;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=249;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=249;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=249;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=249;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=249;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=249;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=249;, score=0.736 total time=   0.7s
[CV 1/10] END …n_neighbors=250;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=250;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=250;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=250;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=250;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=250;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=250;, score=0.735 total time=   0.7s
[CV 8/10] END …n_neighbors=250;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=250;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=250;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=251;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=251;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=251;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=251;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=251;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=251;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=251;, score=0.735 total time=   0.7s
[CV 8/10] END …n_neighbors=251;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=251;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=251;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=252;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=252;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=252;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=252;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=252;, score=0.739 total time=   0.7s
```

```
[CV 6/10] END …n_neighbors=252;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=252;, score=0.735 total time=   0.7s
[CV 8/10] END …n_neighbors=252;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=252;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=252;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=253;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=253;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=253;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=253;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=253;, score=0.738 total time=   0.7s
[CV 6/10] END …n_neighbors=253;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=253;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=253;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=253;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=253;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=254;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=254;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=254;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=254;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=254;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=254;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=254;, score=0.735 total time=   0.7s
[CV 8/10] END …n_neighbors=254;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=254;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=254;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=255;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=255;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=255;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=255;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=255;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=255;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=255;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=255;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=255;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=255;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=256;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=256;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=256;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=256;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=256;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=256;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=256;, score=0.736 total time=   0.7s
[CV 8/10] END …n_neighbors=256;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=256;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=256;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=257;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=257;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=257;, score=0.734 total time=   0.7s
```

```
[CV 4/10] END …n_neighbors=257;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=257;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=257;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=257;, score=0.735 total time=   0.7s
[CV 8/10] END …n_neighbors=257;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=257;, score=0.731 total time=   0.7s
[CV 10/10] END …n_neighbors=257;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=258;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=258;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=258;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=258;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=258;, score=0.739 total time=   0.8s
[CV 6/10] END …n_neighbors=258;, score=0.729 total time=   0.8s
[CV 7/10] END …n_neighbors=258;, score=0.735 total time=   0.7s
[CV 8/10] END …n_neighbors=258;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=258;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=258;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=259;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=259;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=259;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=259;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=259;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=259;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=259;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=259;, score=0.716 total time=   0.8s
[CV 9/10] END …n_neighbors=259;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=259;, score=0.735 total time=   0.7s
[CV 1/10] END …n_neighbors=260;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=260;, score=0.720 total time=   0.8s
[CV 3/10] END …n_neighbors=260;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=260;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=260;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=260;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=260;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=260;, score=0.715 total time=   0.8s
[CV 9/10] END …n_neighbors=260;, score=0.733 total time=   0.8s
[CV 10/10] END …n_neighbors=260;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=261;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=261;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=261;, score=0.735 total time=   0.8s
[CV 4/10] END …n_neighbors=261;, score=0.726 total time=   0.8s
[CV 5/10] END …n_neighbors=261;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=261;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=261;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=261;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=261;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=261;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=262;, score=0.733 total time=   0.7s
```

```
[CV 2/10] END …n_neighbors=262;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=262;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=262;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=262;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=262;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=262;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=262;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=262;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=262;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=263;, score=0.731 total time=   0.7s
[CV 2/10] END …n_neighbors=263;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=263;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=263;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=263;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=263;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=263;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=263;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=263;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=263;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=264;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=264;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=264;, score=0.735 total time=   0.7s
[CV 4/10] END …n_neighbors=264;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=264;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=264;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=264;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=264;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=264;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=264;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=265;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=265;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=265;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=265;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=265;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=265;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=265;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=265;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=265;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=265;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=266;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=266;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=266;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=266;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=266;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=266;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=266;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=266;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=266;, score=0.733 total time=   0.7s
```

```
[CV 10/10] END …n_neighbors=266;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=267;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=267;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=267;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=267;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=267;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=267;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=267;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=267;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=267;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=267;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=268;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=268;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=268;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=268;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=268;, score=0.741 total time=   0.8s
[CV 6/10] END …n_neighbors=268;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=268;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=268;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=268;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=268;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=269;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=269;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=269;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=269;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=269;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=269;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=269;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=269;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=269;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=269;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=270;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=270;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=270;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=270;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=270;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=270;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=270;, score=0.733 total time=   0.7s
[CV 8/10] END …n_neighbors=270;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=270;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=270;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=271;, score=0.734 total time=   0.7s
[CV 2/10] END …n_neighbors=271;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=271;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=271;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=271;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=271;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=271;, score=0.734 total time=   0.7s
```

```
[CV 8/10] END …n_neighbors=271;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=271;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=271;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=272;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=272;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=272;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=272;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=272;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=272;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=272;, score=0.733 total time=   0.7s
[CV 8/10] END …n_neighbors=272;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=272;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=272;, score=0.731 total time=   0.7s
[CV 1/10] END …n_neighbors=273;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=273;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=273;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=273;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=273;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=273;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=273;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=273;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=273;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=273;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=274;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=274;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=274;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=274;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=274;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=274;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=274;, score=0.733 total time=   0.7s
[CV 8/10] END …n_neighbors=274;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=274;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=274;, score=0.731 total time=   0.7s
[CV 1/10] END …n_neighbors=275;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=275;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=275;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=275;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=275;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=275;, score=0.730 total time=   0.7s
[CV 7/10] END …n_neighbors=275;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=275;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=275;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=275;, score=0.731 total time=   0.7s
[CV 1/10] END …n_neighbors=276;, score=0.733 total time=   0.8s
[CV 2/10] END …n_neighbors=276;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=276;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=276;, score=0.725 total time=   0.7s
[CV 5/10] END …n_neighbors=276;, score=0.739 total time=   0.7s
```

```
[CV 6/10] END …n_neighbors=276;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=276;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=276;, score=0.714 total time=   0.7s
[CV 9/10] END …n_neighbors=276;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=276;, score=0.731 total time=   0.7s
[CV 1/10] END …n_neighbors=277;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=277;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=277;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=277;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=277;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=277;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=277;, score=0.733 total time=   0.7s
[CV 8/10] END …n_neighbors=277;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=277;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=277;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=278;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=278;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=278;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=278;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=278;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=278;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=278;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=278;, score=0.714 total time=   0.7s
[CV 9/10] END …n_neighbors=278;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=278;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=279;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=279;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=279;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=279;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=279;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=279;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=279;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=279;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=279;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=279;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=280;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=280;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=280;, score=0.735 total time=   0.7s
[CV 4/10] END …n_neighbors=280;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=280;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=280;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=280;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=280;, score=0.715 total time=   0.7s
[CV 9/10] END …n_neighbors=280;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=280;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=281;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=281;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=281;, score=0.735 total time=   0.7s
```

```
[CV 4/10] END …n_neighbors=281;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=281;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=281;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=281;, score=0.733 total time=   0.7s
[CV 8/10] END …n_neighbors=281;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=281;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=281;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=282;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=282;, score=0.721 total time=   0.7s
[CV 3/10] END …n_neighbors=282;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=282;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=282;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=282;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=282;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=282;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=282;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=282;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=283;, score=0.734 total time=   0.7s
[CV 2/10] END …n_neighbors=283;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=283;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=283;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=283;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=283;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=283;, score=0.733 total time=   0.7s
[CV 8/10] END …n_neighbors=283;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=283;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=283;, score=0.731 total time=   0.7s
[CV 1/10] END …n_neighbors=284;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=284;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=284;, score=0.735 total time=   0.7s
[CV 4/10] END …n_neighbors=284;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=284;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=284;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=284;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=284;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=284;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=284;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=285;, score=0.734 total time=   0.7s
[CV 2/10] END …n_neighbors=285;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=285;, score=0.735 total time=   0.8s
[CV 4/10] END …n_neighbors=285;, score=0.727 total time=   0.8s
[CV 5/10] END …n_neighbors=285;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=285;, score=0.729 total time=   0.8s
[CV 7/10] END …n_neighbors=285;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=285;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=285;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=285;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=286;, score=0.733 total time=   0.7s
```

```
[CV 2/10] END …n_neighbors=286;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=286;, score=0.735 total time=   0.7s
[CV 4/10] END …n_neighbors=286;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=286;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=286;, score=0.729 total time=   0.7s
[CV 7/10] END …n_neighbors=286;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=286;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=286;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=286;, score=0.731 total time=   0.7s
[CV 1/10] END …n_neighbors=287;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=287;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=287;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=287;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=287;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=287;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=287;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=287;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=287;, score=0.731 total time=   0.7s
[CV 10/10] END …n_neighbors=287;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=288;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=288;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=288;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=288;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=288;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=288;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=288;, score=0.735 total time=   0.7s
[CV 8/10] END …n_neighbors=288;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=288;, score=0.731 total time=   0.7s
[CV 10/10] END …n_neighbors=288;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=289;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=289;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=289;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=289;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=289;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=289;, score=0.727 total time=   0.7s
[CV 7/10] END …n_neighbors=289;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=289;, score=0.716 total time=   1.1s
[CV 9/10] END …n_neighbors=289;, score=0.732 total time=   0.8s
[CV 10/10] END …n_neighbors=289;, score=0.733 total time=   0.8s
[CV 1/10] END …n_neighbors=290;, score=0.732 total time=   0.8s
[CV 2/10] END …n_neighbors=290;, score=0.719 total time=   0.8s
[CV 3/10] END …n_neighbors=290;, score=0.734 total time=   0.8s
[CV 4/10] END …n_neighbors=290;, score=0.726 total time=   0.8s
[CV 5/10] END …n_neighbors=290;, score=0.740 total time=   0.8s
[CV 6/10] END …n_neighbors=290;, score=0.728 total time=   0.8s
[CV 7/10] END …n_neighbors=290;, score=0.735 total time=   0.8s
[CV 8/10] END …n_neighbors=290;, score=0.716 total time=   0.8s
[CV 9/10] END …n_neighbors=290;, score=0.732 total time=   0.7s
```

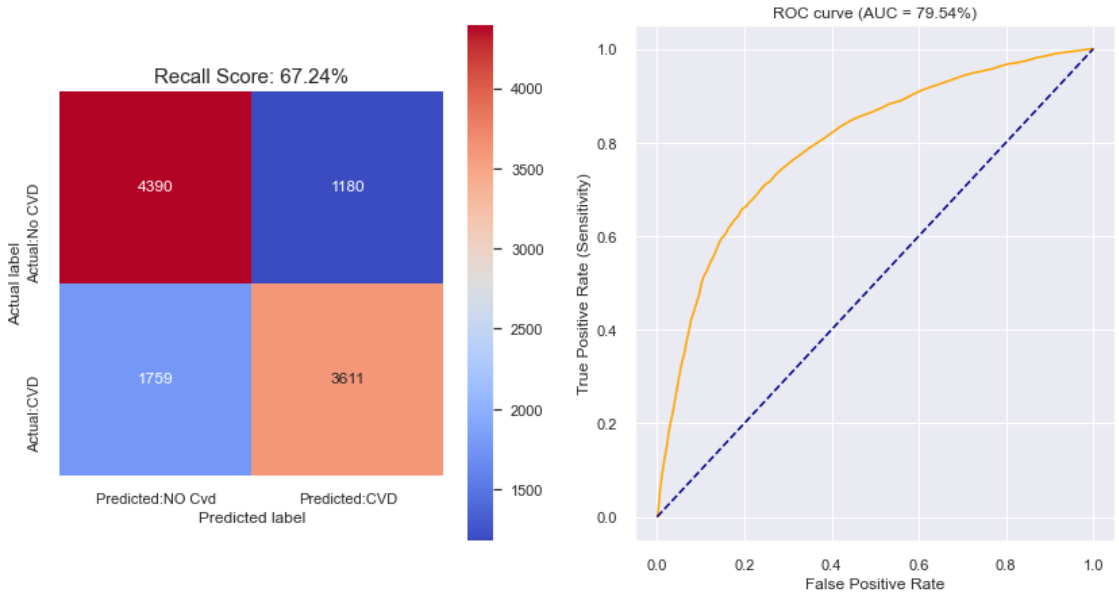
```
[CV 10/10] END …n_neighbors=290;, score=0.732 total time=   0.8s
[CV 1/10] END …n_neighbors=291;, score=0.733 total time=   0.8s
[CV 2/10] END …n_neighbors=291;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=291;, score=0.734 total time=   0.8s
[CV 4/10] END …n_neighbors=291;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=291;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=291;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=291;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=291;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=291;, score=0.732 total time=   0.7s
[CV 10/10] END …n_neighbors=291;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=292;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=292;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=292;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=292;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=292;, score=0.740 total time=   0.7s
[CV 6/10] END …n_neighbors=292;, score=0.727 total time=   0.7s
[CV 7/10] END …n_neighbors=292;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=292;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=292;, score=0.731 total time=   0.7s
[CV 10/10] END …n_neighbors=292;, score=0.732 total time=   0.7s
[CV 1/10] END …n_neighbors=293;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=293;, score=0.718 total time=   0.7s
[CV 3/10] END …n_neighbors=293;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=293;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=293;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=293;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=293;, score=0.735 total time=   0.8s
[CV 8/10] END …n_neighbors=293;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=293;, score=0.731 total time=   0.7s
[CV 10/10] END …n_neighbors=293;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=294;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=294;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=294;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=294;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=294;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=294;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=294;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=294;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=294;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=294;, score=0.733 total time=   0.8s
[CV 1/10] END …n_neighbors=295;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=295;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=295;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=295;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=295;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=295;, score=0.727 total time=   0.7s
[CV 7/10] END …n_neighbors=295;, score=0.734 total time=   0.7s
```

```
[CV 8/10] END …n_neighbors=295;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=295;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=295;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=296;, score=0.733 total time=   0.7s
[CV 2/10] END …n_neighbors=296;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=296;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=296;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=296;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=296;, score=0.727 total time=   0.7s
[CV 7/10] END …n_neighbors=296;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=296;, score=0.716 total time=   0.7s
[CV 9/10] END …n_neighbors=296;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=296;, score=0.734 total time=   0.7s
[CV 1/10] END …n_neighbors=297;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=297;, score=0.719 total time=   0.7s
[CV 3/10] END …n_neighbors=297;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=297;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=297;, score=0.741 total time=   0.7s
[CV 6/10] END …n_neighbors=297;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=297;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=297;, score=0.717 total time=   0.8s
[CV 9/10] END …n_neighbors=297;, score=0.733 total time=   0.7s
[CV 10/10] END …n_neighbors=297;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=298;, score=0.732 total time=   0.7s
[CV 2/10] END …n_neighbors=298;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=298;, score=0.733 total time=   0.7s
[CV 4/10] END …n_neighbors=298;, score=0.727 total time=   0.7s
[CV 5/10] END …n_neighbors=298;, score=0.739 total time=   0.7s
[CV 6/10] END …n_neighbors=298;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=298;, score=0.734 total time=   0.7s
[CV 8/10] END …n_neighbors=298;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=298;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=298;, score=0.733 total time=   0.7s
[CV 1/10] END …n_neighbors=299;, score=0.732 total time=   0.8s
[CV 2/10] END …n_neighbors=299;, score=0.720 total time=   0.7s
[CV 3/10] END …n_neighbors=299;, score=0.734 total time=   0.7s
[CV 4/10] END …n_neighbors=299;, score=0.726 total time=   0.7s
[CV 5/10] END …n_neighbors=299;, score=0.741 total time=   0.8s
[CV 6/10] END …n_neighbors=299;, score=0.728 total time=   0.7s
[CV 7/10] END …n_neighbors=299;, score=0.734 total time=   0.8s
[CV 8/10] END …n_neighbors=299;, score=0.717 total time=   0.7s
[CV 9/10] END …n_neighbors=299;, score=0.734 total time=   0.7s
[CV 10/10] END …n_neighbors=299;, score=0.732 total time=   0.8s
{'n_neighbors': 98}
```

[76]:
```
knn_tuned=KNeighborsClassifier(n_neighbors = 98)
knn_tuned.fit(X_train, y_train)
```

```
show_summary(knn_tuned, X_val, y_val)
plot_conf_ROC(knn_tuned, X_val, y_val)
```

```
              precision    recall  f1-score   support

          no       0.71      0.79      0.75      5570
         yes       0.75      0.67      0.71      5370

    accuracy                           0.73     10940
   macro avg       0.73      0.73      0.73     10940
weighted avg       0.73      0.73      0.73     10940

Specificity :  0.7881508078994613
Sensitivity :  0.67243947858473
```



[77]:
```
y_pred = knn_tuned.predict(X_test)
accuracy_score(y_test, y_pred)
```

[77]: 0.7238756855575869

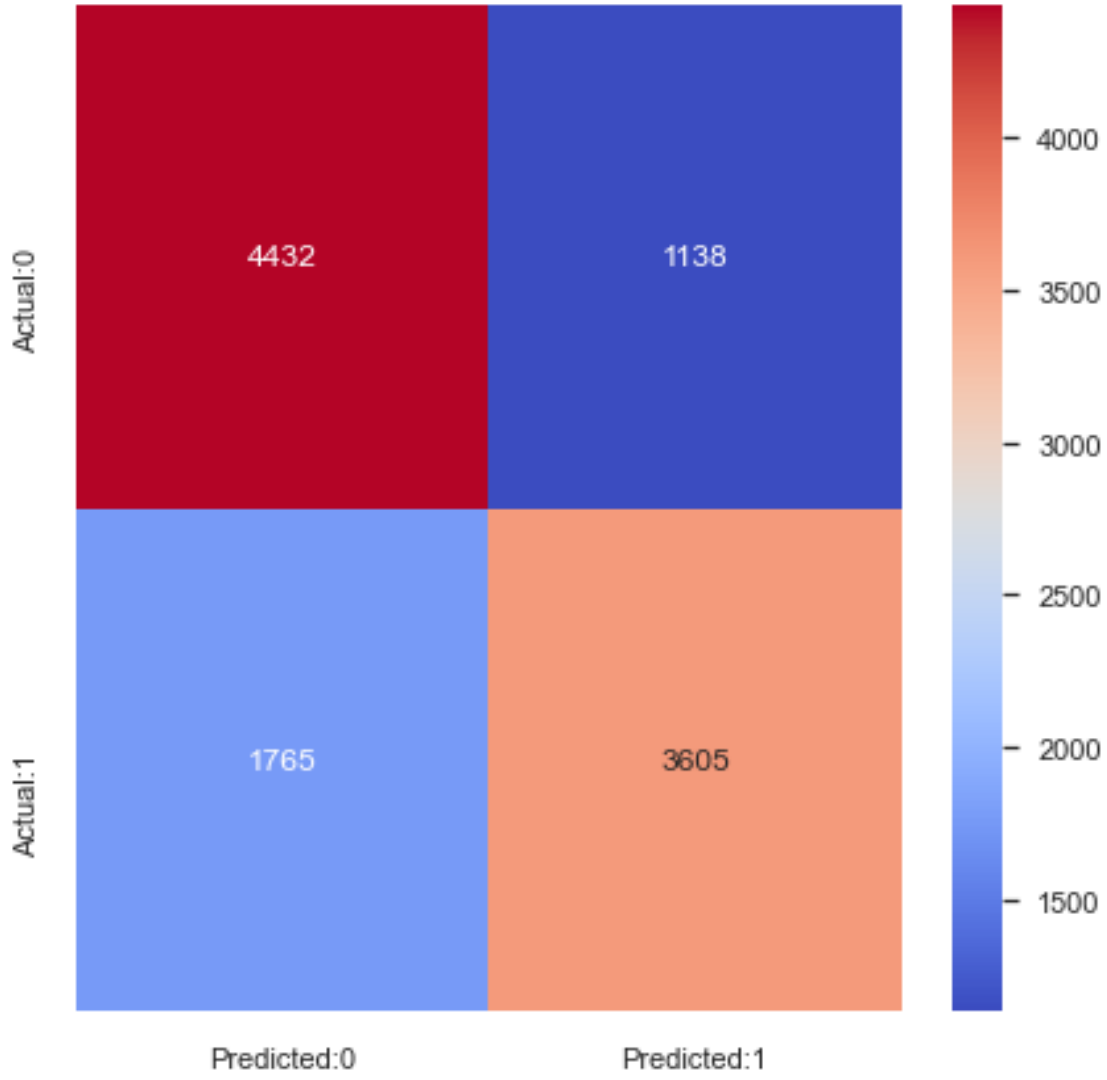
[78]:
```
svm = SVC()
svm.fit(X_train, y_train)
svm_predictions = svm.predict(X_val)
print(classification_report(y_val, svm_predictions))
```

```
              precision    recall  f1-score   support

           0       0.72      0.80      0.75      5570
```

```
           1       0.76      0.67      0.71      5370

    accuracy                           0.73     10940
   macro avg       0.74      0.73      0.73     10940
weighted avg       0.74      0.73      0.73     10940
```

[81]:
```
plt.figure(figsize=(7,7))
cm=confusion_matrix(y_val, svm_predictions)
conf_matrix=pd.DataFrame(data=cm,columns=['Predicted:0','Predicted:
 ↪1'],index=['Actual:0','Actual:1'])
sns.heatmap(conf_matrix, annot=True,fmt='d',cmap="coolwarm")
specificity= cm[0,0]/(cm[0,0]+cm[0,1])
print('Specificity : ', specificity)

sensitivity= cm[1,1]/(cm[1,0]+cm[1,1])
print('Sensitivity : ', sensitivity)
```

```
Specificity :  0.7956912028725314
Sensitivity :  0.6713221601489758
```



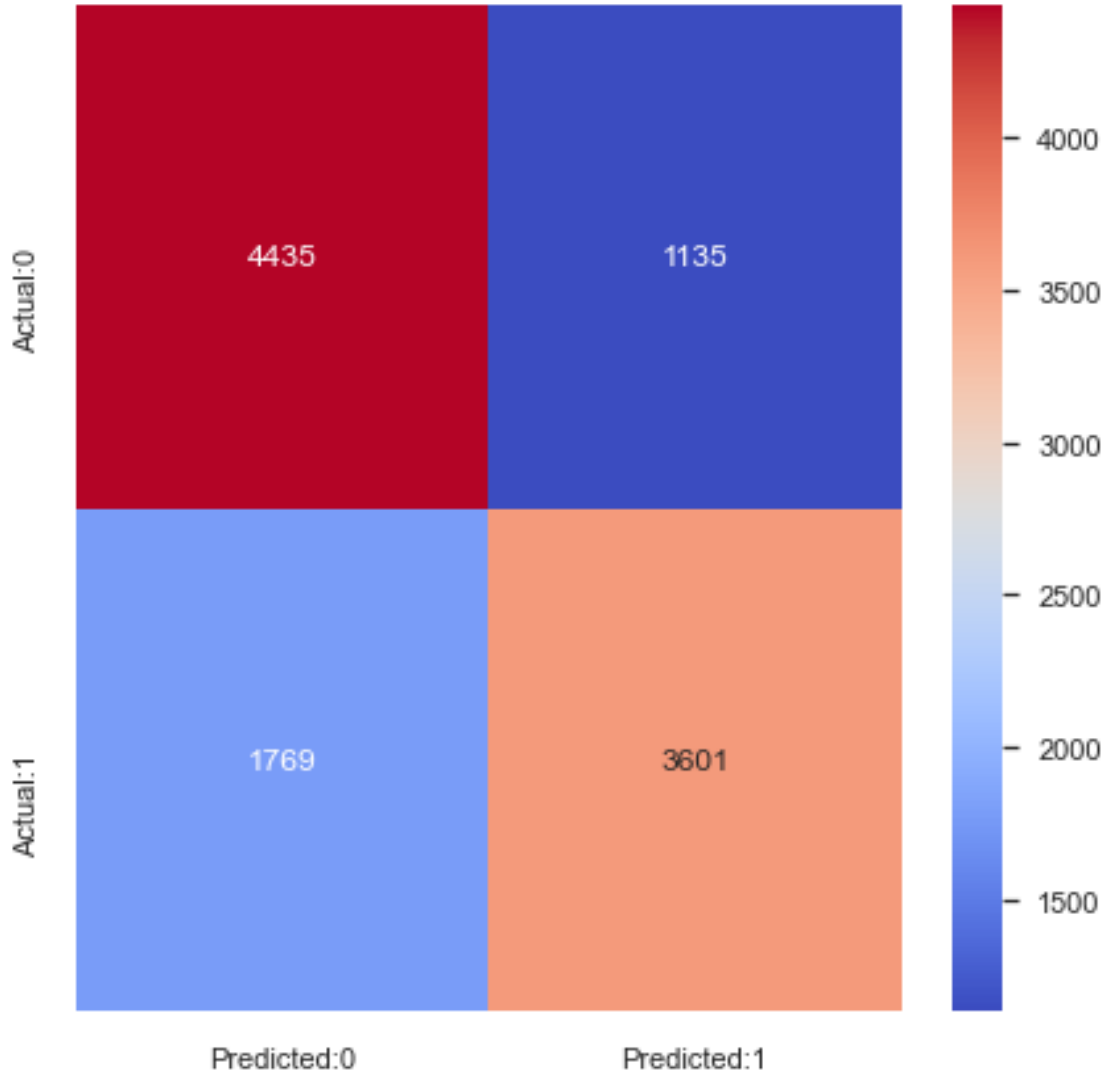
[82]:
```
param_grid = {'C': [0.1, 1, 10, 100],
              'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
              'gamma':['scale', 'auto'],
              'kernel': ['rbf','linear','poly']}

svm_tunned= GridSearchCV(SVC(), param_grid, refit = True, verbose =␣
↪3,n_jobs=-1,cv=5)
svm_tunned.fit(X_train, y_train)
print(svm_tunned.best_params_)
print(svm_tunned.best_estimator_)
svm_tunned_predictions = svm_tunned.predict(X_val)
print(classification_report(y_val, svm_tunned_predictions))
```

```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
```

```
{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}
SVC(C=10)
              precision    recall  f1-score   support

           0       0.71      0.80      0.75      5570
           1       0.76      0.67      0.71      5370

    accuracy                           0.73     10940
   macro avg       0.74      0.73      0.73     10940
weighted avg       0.74      0.73      0.73     10940
```

[84]:
```
plt.figure(figsize=(7,7))
cm=confusion_matrix(y_val, svm_tunned_predictions)
conf_matrix=pd.DataFrame(data=cm,columns=['Predicted:0','Predicted:
 ↪1'],index=['Actual:0','Actual:1'])
sns.heatmap(conf_matrix, annot=True,fmt='d',cmap="coolwarm")
specificity= cm[0,0]/(cm[0,0]+cm[0,1])
print('Specificity : ', specificity)

sensitivity= cm[1,1]/(cm[1,0]+cm[1,1])
print('Sensitivity : ', sensitivity)
```

```
Specificity :  0.796229802513465
Sensitivity :  0.6705772811918064
```



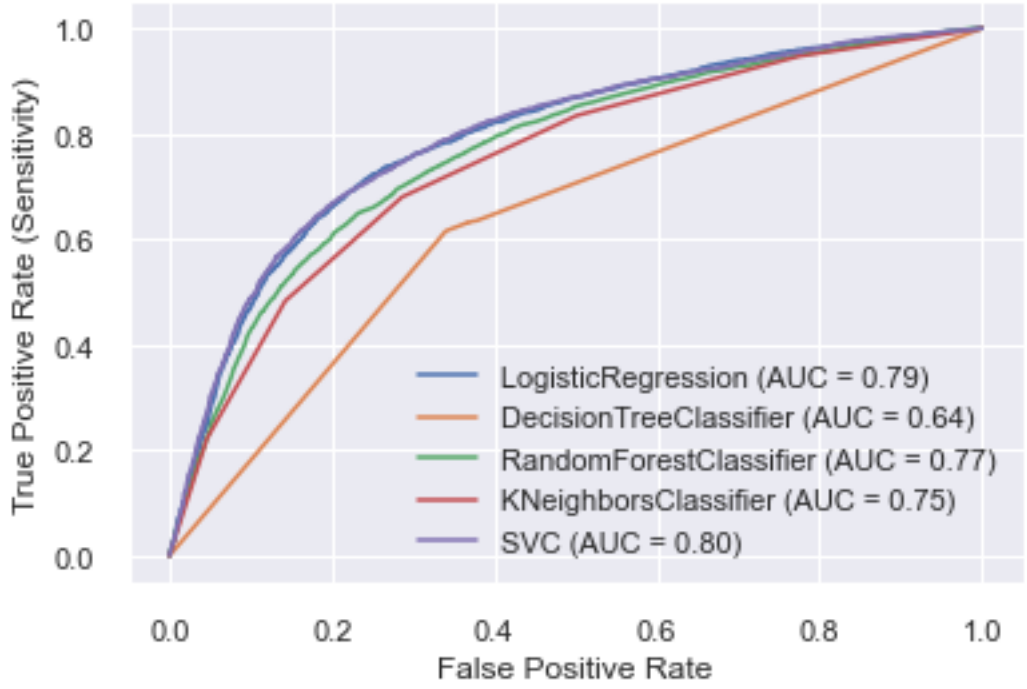
[87]:
```
y_pred = svm_tunned.predict(X_test)
accuracy_score(y_test, y_pred)
```

[87]: 0.7259963436928702

[165]:
```
disp = plot_roc_curve(logreg, X_val, y_val)

plot_roc_curve(decision_tree,X_val, y_val, ax = disp.ax_)

plot_roc_curve(forest,X_val, y_val, ax = disp.ax_)

plot_roc_curve(knn,X_val, y_val, ax = disp.ax_)
```

```
plot_roc_curve(svm,X_val, y_val, ax = disp.ax_)

plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate (Sensitivity)')
```

[165]:
```
Text(0, 0.5, 'True Positive Rate (Sensitivity)')
```



[85]:
```
disp = plot_roc_curve(logreg_optimized, X_val, y_val)

plot_roc_curve(tree_tuned,X_val, y_val, ax = disp.ax_)

plot_roc_curve(forest_tuned,X_val, y_val, ax = disp.ax_)

plot_roc_curve(knn_tuned,X_val, y_val, ax = disp.ax_)

plot_roc_curve(svm_tunned,X_val, y_val, ax = disp.ax_)

plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate (Sensitivity)')
```

[85]:
```
Text(0, 0.5, 'True Positive Rate (Sensitivity)')
```


True Positive Rate (Sensitivity)
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
GridSearchCV (AUC = 0.79)
DecisionTreeClassifier (AUC = 0.79)
RandomForestClassifier (AUC = 0.75)
KNeighborsClassifier (AUC = 0.80)
GridSearchCV (AUC = 0.79)

[ ]: