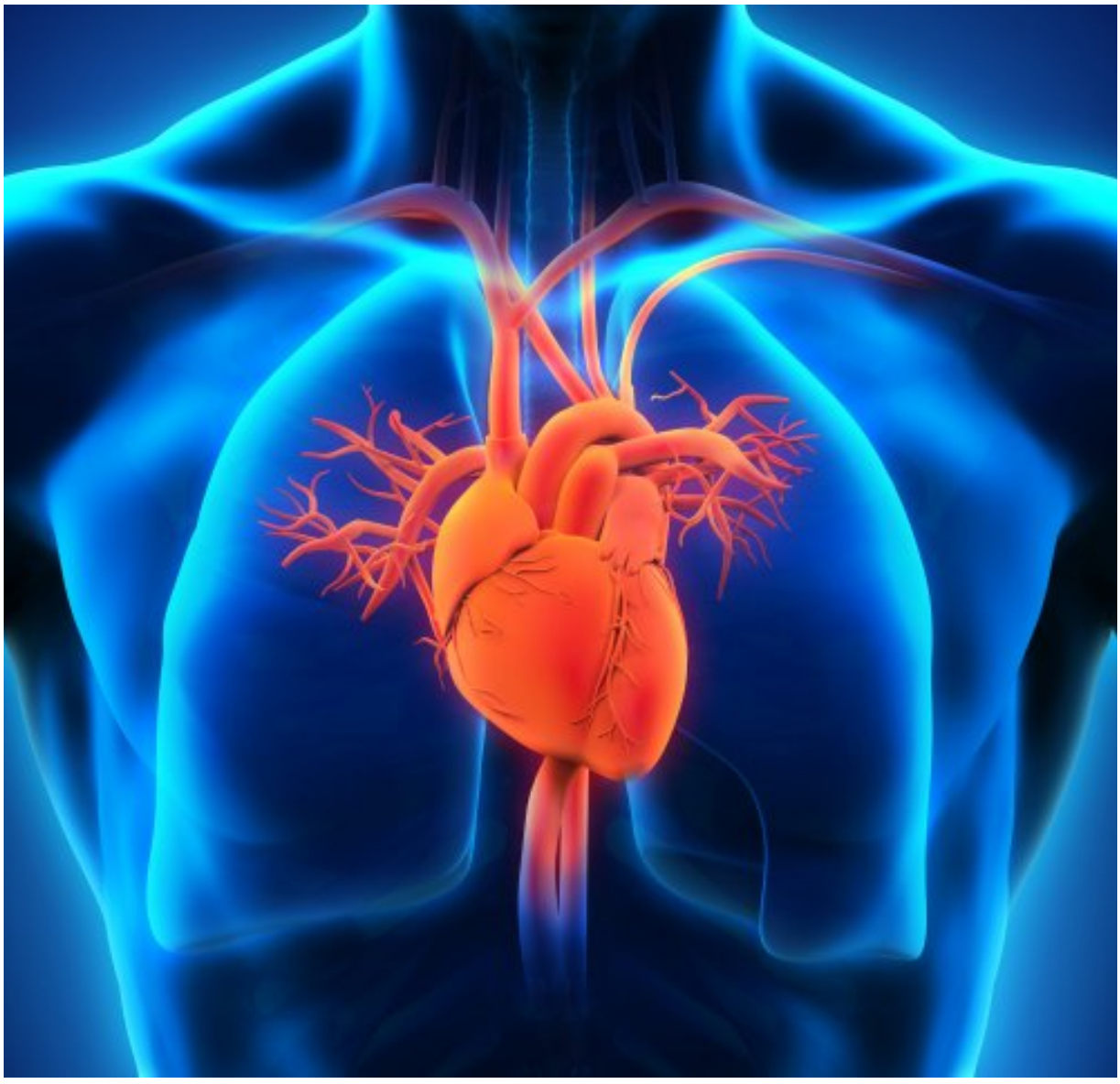
جامعة جدة
University of Jeddah

Machine Learning on Cardiovascular Disease Dataset



Submitted by:

Ruba Alsulami 2110618
Teif Alghamdi 2111370
Arjwan Alharbi 2110826

Submitted to:

Dr: Safaa Alsafari

Introduction

Cardiovascular disease (CVD) is a general term for conditions affecting the heart or blood vessels.It's usually associated with a build-up of fatty deposits inside the arteries (atherosclerosis) and an increased risk of blood clots. It can also be associated with damage to arteries in organs such as the brain, heart, kidneys and eyes. CVD is one of the main causes of death, but it can often largely be prevented by leading a healthy lifestyle.
The benefits of using this dataset and the results of the KNN and NB algorithm can have many real-world applications. For example, healthcare professionals can use this information to identify individuals who are at a higher risk of developing cardiovascular disease and develop appropriate interventions.

Problem Description

Because of the seriousness of cardiovascular diseases, we need to develop a predictive model to help identifying individuals at higher risk of developing disease. Thus, lower the prevalence of cardiovascular disease and enhance overall health and wellbeing.

Data Description

The cardiovascular disease dataset available on Kaggle is a comprehensive dataset that contains various features related to the health of individuals and their likelihood of developing cardiovascular disease. The dataset includes objective features such as age, height, weight, gender, and examination features such as systolic and diastolic blood pressure, cholesterol, and glucose levels. Additionally, the dataset includes subjective features such as smoking, alcohol intake, and physical activity. The target variable in this dataset is the presence or absence of cardiovascular disease.
In order to predict the presence or absence of cardiovascular disease, various machine learning algorithms can be used. One such algorithm is logistic regression. This algorithm is commonly used in medical research and can be used to predict the probability of a binary outcome.

Method

In our project we used K-Nearest Neighbors (KNN) and Naive Bayes (NB)

K-Nearest Neighbors (KNN)

K - Nearest Neighbors (KNN) is a simple and popular classification algorithm, KNN tries to predict the correct class for the test data by calculating the distance between the test data and all the training points. Then select the K number of points which is closet to the test data.In the code below, we call KNeighborsClassifier from sklearn library and train our data on a labeled dataset using the fit method and ask it to predict the label for an unlabeled point using the predict method.By using grid search cross-validation, we can systematically search through different combinations of hyperparameters and identify the best configuration that yields the highest accuracy for the KNeighborsClassifier model. This helps in finding the optimal hyperparameters and improves the performance of the model on unseen data.

```
scaler = MinMaxScaler()
# fit the scaler to the training data and transform it
X_train_scaled = scaler.fit_transform(X_train)
# transform the test data using the fitted scaler
X_test_scaled = scaler.transform(X_test)
```

```
# KNeighborsClassifier
kn = KNeighborsClassifier()
# fit the model
kn.fit(X_train_scaled, y_train)
# test the model on the test data
y_pred = kn.predict(X_test_scaled)
```

```
# Define the hyperparameters to tune
param_grid = {'n_neighbors': [3, 5, 7],
              'weights': ['uniform', 'distance']}

# Create the KNeighborsClassifier model
kn = KNeighborsClassifier()

# Perform grid search cross-validation
grid_kn = GridSearchCV(kn, param_grid, cv=5, scoring='accuracy')
grid_kn.fit(X_train_scaled, y_train)

# Get the best hyperparameters and model
best_kn = grid_kn.best_estimator_
best_params_kn = grid_kn.best_params_

# Test the model on the test data
y_pred = best_kn.predict(X_test_scaled)
```

Naive Bayes (NB)

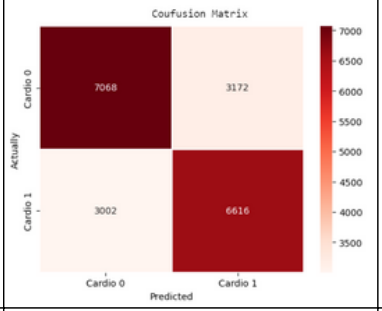
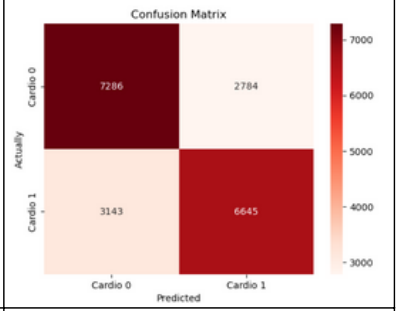
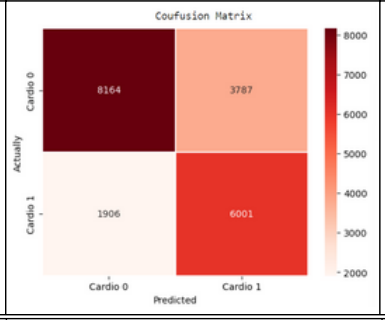
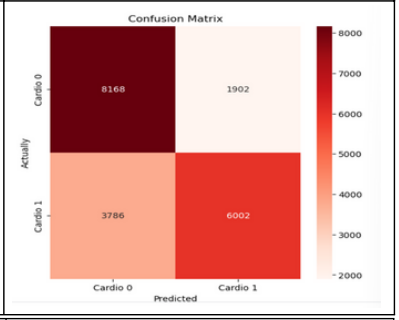
The Naive Bayes Algorithm is one of the crucial algorithms in machine learning that helps with classification problems Naive Bayes is a probabilistic algorithm that uses Bayes' theorem to make predictions. It is known for its simplicity, fast training and prediction times..In the code below, we call NaiveBayesClassifie from sklearn library and train our data on a labeled dataset using the fit method and ask it to predict the label for an unlabeled point using the predict method.By using grid search cross-validation, we can systematically search through different combinations of hyperparameters and identify the best configuration that yields the highest accuracy for the NB classifier model. This helps in finding the optimal hyperparameters and improves the performance of the model on unseen data.

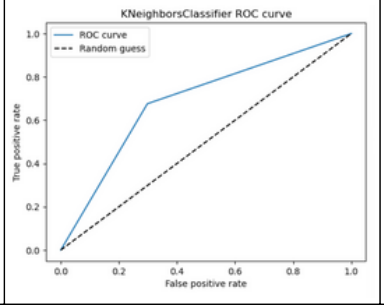
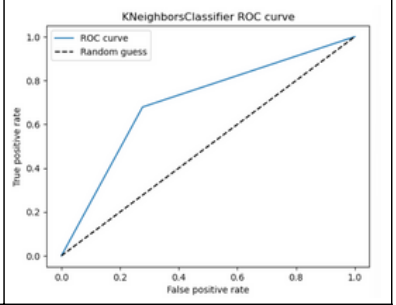
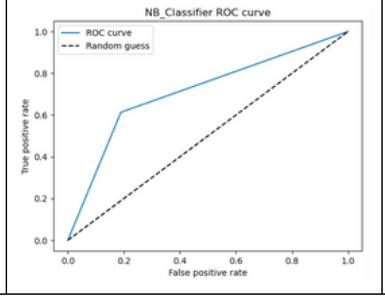
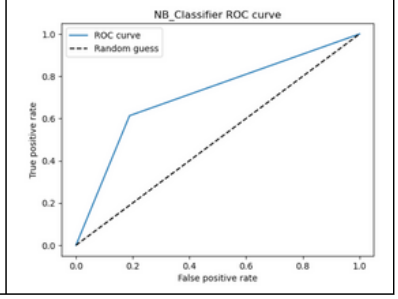
```
# NBclassifier
nb = GaussianNB()

# fit the model
nb.fit(X_train_scaled, y_train)

# test the model on the test data
y_pred = nb.predict(X_test_scaled)
```

```
from sklearn.model_selection import RepeatedStratifiedKFold
# Cross-Validation method
cross_validation = RepeatedStratifiedKFold(n_splits=3, n_repeats=6, random_state=999)
# Define the hyperparameters to tune
param_nb = {'var_smoothing': np.logspace(0,-15, num=120)}

# Create the NBClassifier model
nb = GaussianNB()
# Perform grid search cross-validation
grid_nb = GridSearchCV(estimator=nb, param_grid=param_nb,cv=cross_validation, verbose=1,scoring='accuracy')
grid_nb.fit(X_train_scaled, y_train)
# Get the best hyperparameters and model
best_nb = grid_nb.best_estimator_
best_params_nb = grid_kn.best_params_
# Test the model on the test data
y_pred = best_nb.predict(X_test_scaled)
```

Experiment and results

Measurements	K-NN Before using hyperparameter	K-NN After using hyperparameter	NB Before using hyperparameter	NB After using hyperparameter
Accuracy Score	0.689	0.702	0.7133	0.7136
Precision	0.688	0.705	0.7589	0.7594
Recall	0,676	0.679	0.6131	0.6132
F1 - Score	0.682	0.692	0.6783	0.6785

Confusion Matrix	Before using hyperparameter	After using hyperparameter
K-NN		
NB		

ROC Curve	Before using hyperparameter	After using hyperparameter
K-NN		
NB		

Discussion

The reasons for the difference in outcomes between NB and KNN classifier, because Naive Bayes It tends to be faster when applied to big data.KNN is usually slower for large amounts of data, because of the calculations required for each new step in the process. If speed is important, choose Naive Bayes over K-NN.

The classification report for GaussianNB shows that it achieved an accuracy of 0.7136. In terms of precision, it scored 0.7594, which indicates the proportion of correctly predicted positive instances out of all instances predicted as positive. The recall value of 0.6132 indicates the proportion of correctly predicted positive instances out of all actual positive instances. The F1-score of 0.6785 represents the harmonic mean of precision and recall, which provides a balance between the two metrics. Overall, GaussianNB achieved reasonably good performance on the given dataset.

The classification report for KNeighborsClassifier shows that it achieved an accuracy of 0.702, which is slightly lower than the accuracy of GaussianNB. The precision value of 0.705 indicates that it correctly predicted 70.5% of positive instances out of all instances predicted as positive. The recall value of 0.679 shows that it identified 67.9% of actual positive instances correctly. The F1-score of 0.692 indicates a balance between precision and recall for this model.

Comparing the two models, GaussianNB outperformed.KNeighborsClassifier in terms of accuracy, precision, recall, and F1-score, because the Datasetis Balance. However, it's important to note that the choice of the best model depends on the specific requirements of the problem at hand and the trade-offs between different evaluation metrics.

Conclusion

Both KNN and Naive Bayes are popular and effective classification algorithms that can be used in various applications. k-NN is a non-parametric algorithm that makes predictions based on similarity between a new instance and its k-nearest neighbors. It is simple and flexible in handling different data types, but can be computationally intensive and sensitive to the choice of k.The choice between KNN and Naive Bayes depends on the specific problem and the trade-offs between computational complexity and solution optimality.

References

- SVETLANA ULIANOVA. (2019). Cardiovascular Disease dataset. Canada. Retrieved from https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset
- Sadier, J. (2020). K-Nearest Neighbors Algorithm.
Retrieved from What is knn. Ibm. https://www.ibm.com/topics/knn
- Nearest Neighbors. Scikit-Learn.
Retrieved from https://scikit-learn.org/stable/modules/neighbors.html
-Andrea D'Agostino . Exploratory Data Analysis. Towardsdatascience. Retrieved from https://towardsdatascience.com/exploratory-data-analysis-in-python-a-step-by-step-process-d0dfa6bf94ee-
-EDA - Exploratory Data Analysis: Using Python Functions. Digitalocean. Retrieved from https://www.digitalocean.com/community/tutorials/exploratory-data-analysis-python
-Naive Bayes. Scikit-Learn. Retrieved from https://scikit-learn.org/stable/modules/naive_bayes.html
-NaiveBayes.MachineLearningMastery.Retrievedfrom https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/