**Personal Project**

**DS**

**DOCUMENT INFORMATION**

**VERSIEBEHEER:**

|  |  |  |
| --- | --- | --- |
| **Version** | **Description** | **Date** |
| 1.0 | First version written | 29/05/2023 |

**Author:**

|  |  |
| --- | --- |
| **Name** | **Email** |
| Pagter, Robin de | r.depagter@zevenaar.nl |

**TABLE OF CONTENT**

[1. CRISP-DM 4](#_Toc136254309)

[1.1 Structured take on Data Mining 4](#_Toc136254310)

[1.2 Iterational Data Mining 4](#_Toc136254311)

[1.3 Exploration of data 4](#_Toc136254312)

[2. Business understanding 5](#_Toc136254313)

[2.1 Problem context 5](#_Toc136254314)

[2.2 Stakeholders 5](#_Toc136254315)

[3. Data understanding 6](#_Toc136254316)

**INTRODUCTION**

The following report has been written as a personal project for the minor: “Data Driven Decision Making in Business” at the HAN university of applied sciences. The task at hand is to choose a known or even unknown problem, the solution to this problem should then benefit a certain (group of) stakeholder(s). Inside the minor, various techniques and best practices have been taught by the teachers to handle data related problems. A few examples are: writing python code for machine learning, researching data science related subjects and visualizing data inside a program such as Excel or Power BI.

To correctly investigate the solution to the problem, the CRISP-DM model for data science will be used. CRISP-DM stands for ‘Cross-industry standard process for data mining’, it allows researchers to follow a set of steps to help them achieve a solution to the problem they are analysing. In this report, the problem is to predict heart disease in both males and females using certain pieces of data.

# CRISP-DM

As said in the introduction, this report is based around the CRISP-DM model. Each phase of the CRISP-DM model will be followed to ensure the best possible outcome of the project. In this chapter, CRISP-DM will be further explained:

## Structured take on Data Mining

Data mining is, according to the book Data Science for Business, a complicated craft. It takes both scientific and technological knowledge to be executed. It’s an advanced process that requires some form of structure. One of those structures is called the Cross Industry Standard Process for Data Mining or CRISP-DM. It takes Data Mining and turns it into a consistent and structured process to be executed.

(Provost & Fawcett, 2013)

## Iterational Data Mining

CRISP-DM is made to be an iterational process. One that can be executed any number of times before finalizing the data mining process itself. It allows users to redo certain steps to really get an understanding of the data and the factors around it. So the first iteration can be exploration, in the second iteration the scientists know much more and in the third it can be fine tuning the data and the scientist will be well-informed of the data and the context in which it is used.

(Provost & Fawcett, 2013)

## Exploration of data

CRISP-DM allows the scientists to explore their data and, in each iteration, allow them to better understand it. This will help the data mining process to become better with each iteration as it is not necessary to solve the problem right away. It can take many iterations of the CRISP-DM to actually find the solution.

(Provost & Fawcett, 2013)

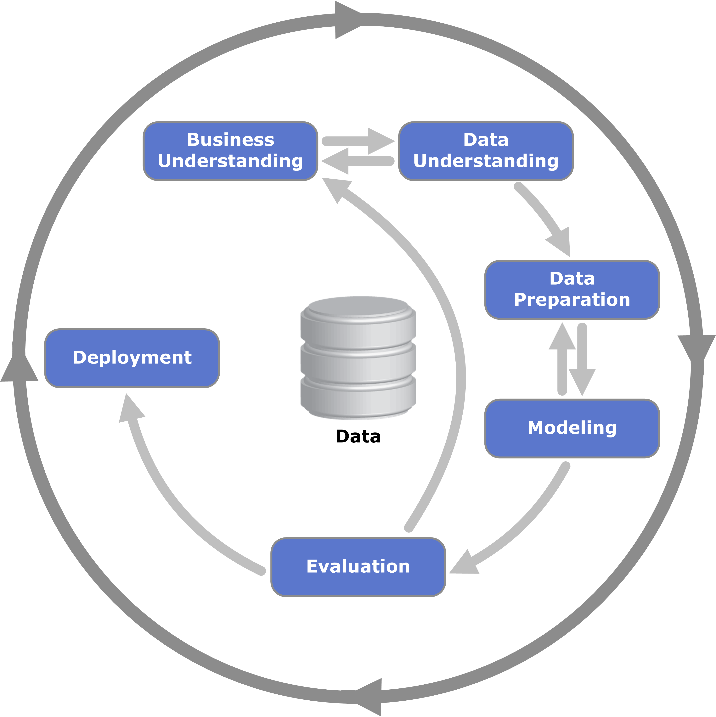


Figure 1 - CRISP-DM model

# Business understanding

In the phase of Business Understanding, one has to deeply understand the business problem. This allows the data scientist to truly understand what the solution has to provide for the problem that needs to be solved. This will make sure that no unnecessary effort will be spent in solving the wrong problem or one that does not exist at all.

## Problem context

According to the context of the dataset, cardiovascular diseases (CVD’s) (Heart and blood vessel disease) is the number one cause of death on a global scale. Around 17.9 million people lose their lives to it every year, which is around 31% of all deaths worldwide. Four out of five of these cases are due to heart attacks or strokes, one third of those cases happen before the age of 70.

These numbers make it vital to prevent CVD’s from appearing and thus predicting them in an early stage so proper medical care can be provided.

## Stakeholders

For this problem, various stakeholders can be defined using a persona. In short, both patients and doctors will benefit from the solution to the described problem. By quickly predicting CVD’s, one can provide and receive the right care.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Age | Description | Occupation |
| Frank | 67 | Frank enjoys good food which isn’t always healthy. He like to take long walks and wants to enjoy his free time with his family. | Retired |
| Sevilla | 49 | Sevilla is a working mom of 2 kids and lives a stressful life. She works while also caring for 2 young children that go to high school. | Office worker |
| Fred | 32 | Fred just graduated from medical school and now works as a doctor in a public hospital in the cardiology section. He doesn’t have much experience but can’t wait to help his patients. | Heart doctor |

# Data understanding

The dataset was found on Kaggle, a website where large data sets can be found for applying data science or business intelligence (*Kaggle: Your Machine Learning and Data Science Community*, n.d.). The dataset page contains information about not only the different columns found but also where the data comes from.

## Data information

The dataset contains various datapoints that, as a whole, allow doctors to determine of their patients have CVD. The goal is, of course, to automatically load this data inside a model that will determine if the patient has CVD. The dataset contains (*Heart Failure Prediction Dataset*, 2021):

1. Age: age of the patient [years]
2. Sex: sex of the patient [M: Male, F: Female]
3. ChestPainType: chest pain type [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]
4. RestingBP: resting blood pressure [mm Hg]
5. Cholesterol: serum cholesterol [mm/dl]
6. FastingBS: fasting blood sugar [1: if FastingBS > 120 mg/dl, 0: otherwise]
7. RestingECG: resting electrocardiogram results [Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria]
8. MaxHR: maximum heart rate achieved [Numeric value between 60 and 202]
9. ExerciseAngina: exercise-induced angina [Y: Yes, N: No]
10. Oldpeak: oldpeak = ST [Numeric value measured in depression]
11. ST\_Slope: the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping]
12. HeartDisease: output class [1: heart disease, 0: Normal]

## Data source

The source of the data is from different parts of the world. In total, there are 918 observations of both male and female patients. The data is a combination of different existing datasets which makes this set, according to Kaggle, the largest dataset for research purposes. The sources are: Hungary, Cleveland, Switzerland, Long Beach VA and Stalog (Heart) Data Set.

# Data preparation

To prepare the data for the model, which will be chosen in the next chapter, one will need to run various analysis to make sure it fits the model perfectly. To start, the data will be loaded into Python using the Pandas package. Then the data will be manipulated to make sure it is ready for the model. All the steps will be described within this chapter.

## Required packages

To load the data and change it, python will need the pandas package. This package will allow python to load excel and csv files and manipulate the data to make sure it fits the situation:

import pandas as pd

## Load the data

The data from Kaggle is supplied as a CSV file, however it uses an older format which is difficult to load into Python. The first step will be to convert it to XLSX in a program such as Microsoft Excel.

The XLSX file can then be loaded into python using the following code:

dataLink = "Heart data.xlsx"

rawData = pd.read\_excel(dataLink)

rawData.info()

Using the info() function, we get to take a quick look at the data:

Afbeelding met tekst, schermopname, Lettertype, menu

Automatisch gegenereerde beschrijving

This overview gives enough information about the data for now. For example, there are no null values in all the columns. Also, it quickly shows what datatype each individual column holds.

## Transform the data

The next step will be to transform the data. To be able to use the data inside of a model, all the datapoints should be numeric. This will make sure that it can be used in most models. Then, it is important to check if there are any outliers in the data and how to deal with these. Finally, one would need to check if the data contains strong correlations and, again, deal with this.

### Turn into numeric

According to the earlier generated info table, there are 5 columns with the data type ‘Object’. Let’s turn these into numeric datatypes like ‘int’ or ‘float’, using the following code:

# Sex

rawData['Sex']               = np.where((rawData['Sex']=='M'), 1, 0)

#1 is male, 0 is female

# Chest pain

rawData['ChestPainType\_ATA'] = np.where((rawData['ChestPainType']=='ATA'), 1, 0)

rawData['ChestPainType\_NAP'] = np.where((rawData['ChestPainType']=='NAP'), 1, 0)

# RestingECG

rawData['RestingECG\_Normal'] = np.where((rawData['RestingECG']=='Normal'), 1, 0)

rawData['RestingECG\_ST']     = np.where((rawData['RestingECG']=='ST'), 1, 0)

rawData['RestingECG\_LVH']    = np.where((rawData['RestingECG']=='LVH'), 1, 0)

# ExerciseAngina

rawData['ExerciseAngina\_N']  = np.where((rawData['ExerciseAngina']=='N'), 1, 0)

rawData['ExerciseAngina\_Y']  = np.where((rawData['ExerciseAngina']=='Y'), 1, 0)

# ST\_Slope

rawData['ST\_Slope\_Down']     = np.where((rawData['ST\_Slope']=='Down'), 1, 0)

rawData['ST\_Slope\_Flat']     = np.where((rawData['ST\_Slope']=='Flat'), 1, 0)

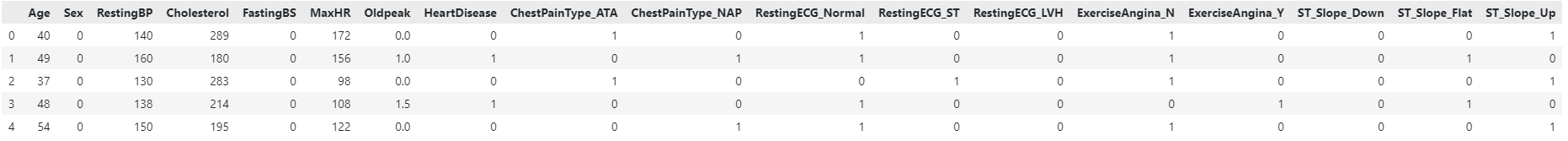
rawData['ST\_Slope\_Up']       = np.where((rawData['ST\_Slope']=='Up'), 1, 0)

# Dropping the old columns

rawData = rawData.drop(['ChestPainType', 'RestingECG', 'ExerciseAngina', 'ST\_Slope'], axis=1)

rawData.head()

This will generate the following result:

…

### Check for outliers

To make sure that the results will not be influenced by any outliers, it is important to look for them using python functions. Once the outliers are found, it can be analyzed whether or not to remove them completely from the data.

Using Cook’s D, one can find the outliers in the data and put them in a separate variables. One variable will contain all the data including outliers, one will only contain non-outlier data. The variables will then be loaded into a model to check if removing outliers has any effect on the influence of each independent variable on the dependent variable.

# Let's investigate the outliers using Cook's D

#First create the model

model = sm.ols('HeartDisease ~ Age+Sex+RestingBP+Cholesterol+FastingBS+MaxHR+Oldpeak+ChestPainType\_ATA+ChestPainType\_NAP+RestingECG\_Normal+RestingECG\_ST+RestingECG\_LVH+ExerciseAngina\_N+ExerciseAngina\_Y++ST\_Slope\_Down+ST\_Slope\_Flat+ST\_Slope\_Up', data = rawData).fit()

cooksD = model.get\_influence().cooks\_distance #Calculate Cook's D values from model 1

n = len(rawData) # Calculate the size of the sample

rawData['Outlier'] = cooksD[0] > 4/n

DF\_outlier\_Data = rawData[rawData.Outlier == True]

DF\_remove\_Outliers = rawData[rawData.Outlier == False]

DF\_outlier\_Data.count() #38 outliers, according to cook's D in this data set

# Let's create 2 models, (1) one with all the outliers, (2) one with all outliers removed:

model1 = sm.ols('HeartDisease ~ Age+Sex+RestingBP+Cholesterol+FastingBS+MaxHR+Oldpeak+ChestPainType\_ATA+ChestPainType\_NAP+RestingECG\_Normal+RestingECG\_ST+RestingECG\_LVH+ExerciseAngina\_N+ExerciseAngina\_Y++ST\_Slope\_Down+ST\_Slope\_Flat+ST\_Slope\_Up', data = rawData).fit()

model2 = sm.ols('HeartDisease ~ Age+Sex+RestingBP+Cholesterol+FastingBS+MaxHR+Oldpeak+ChestPainType\_ATA+ChestPainType\_NAP+RestingECG\_Normal+RestingECG\_ST+RestingECG\_LVH+ExerciseAngina\_N+ExerciseAngina\_Y++ST\_Slope\_Down+ST\_Slope\_Flat+ST\_Slope\_Up', data = DF\_remove\_Outliers).fit()

print(model1.summary())

print(model2.summary())

#Removing all outliers does changes the outcome of the model,

# for example R-Squared goes from 0.554 in model1 to 0.651 in model2, the P values remain mostly the same so model 2 will be used.

### Check for correlations

To make sure that the data do not influence each other too strongly, it is possible to check for correlations within the data itself. This will allow for insights in how the data effects each other when a certain value goes up or down.

For this method, it is possible to use the .corr() function in python combined with a heatmap to visualize the correlations. The following code applies:

# Let's create a matrix to show the correlations between values

correlationMatrix = DF\_remove\_Outliers[['Age','Sex','RestingBP','Cholesterol','FastingBS','MaxHR','Oldpeak','ChestPainType\_ATA','ChestPainType\_NAP','RestingECG\_Normal','RestingECG\_ST','RestingECG\_LVH','ExerciseAngina\_N','ExerciseAngina\_Y','ST\_Slope\_Down','ST\_Slope\_Flat','ST\_Slope\_Up']].corr()

plt.figure(figsize = (16,5))

sns.heatmap(correlationMatrix, annot=True, linewidths=.6)

A picture containing text, screenshot, pattern, square

Description automatically generated# As seen in the matrix below, there is no strong correlation between any columns. The highest is 0.46

Figure 2 - Heatmap of correlations

## Creating test and training data

To be able to use the data inside a model, it is necessary to split the data into both a training and testing set. This will allow the model to be able to train itself and test it’s capabilities with data it has not seen before inside the training phase.

To make sure that data with larger values do not outshine the data with lower values, it is important to normalize the data to make everything a value between 0 and 1. For this transformation python has the function of the standardscaler(). This will be used to normalize all the training and testing data.

# As far as the dataset goes, there are no columns that cannot be used for predicting heart disease in other words:

# Nothing has to be eliminated from the dataset.

DF\_remove\_Outliers =  DF\_remove\_Outliers.drop(['Outlier'], axis=1)

prepData = DF\_remove\_Outliers

prepData.head()

# The column 'Outcome' is what we want to predict, 1 is tested and diagnosed with heart disease and 0 is tested but not diagnosed with heart disease.

cntOutcome = prepData['HeartDisease'].value\_counts()

propOutcome = prepData['HeartDisease'].value\_counts(normalize=True)

# Transform the column 'Outcome' to the type Category instead of integer.

# This is necessary since most models in general cannot use an Integer as a category, they require a column of the type 'Category'

catType = pd.CategoricalDtype(categories=[0, 1], ordered=False)

prepData['HeartDisease'] = prepData['HeartDisease'].astype(catType)

prepData['HeartDisease']

# To see if it worked, uncomment the follow line of code (22):

# prepData.info()

excluded = ['HeartDisease'] #This columns will be excluded

X = prepData.loc[:, ~prepData.columns.isin(excluded)] #This will fetch the data minus the excluded column and put it into the variable X

scaler = MinMaxScaler()

data = scaler.fit\_transform(X)

X[['Age','Sex','RestingBP','Cholesterol','FastingBS','MaxHR','Oldpeak','ChestPainType\_ATA','ChestPainType\_NAP','RestingECG\_Normal','RestingECG\_ST','RestingECG\_LVH','ExerciseAngina\_N','ExerciseAngina\_Y','ST\_Slope\_Down','ST\_Slope\_Flat','ST\_Slope\_Up']].describe() #This is the same as earlier code but now normalized

labels = prepData['HeartDisease']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(data, labels, test\_size=0.25, random\_state=123)

# Modeling

Finally, the modelling phase. Within this phase, the data scientist decides upon the model to use for the purpose of his/her research. That is exactly what will be done within this chapter. Now that the data is completely ready and thoroughly examined, it is possible to pick a model and try to load the data into it.

For the purpose of predicting heart disease, given the data that I supplied for this particular scenario, the model of kNN nearest neighbor will suit the needs best for the task. The reason for picking kNN is that the dataset contains data which describes people that have heart disease or not, if given new data, the data scientist could hold this new data next to the existing one and find the nearest neighbors to decide if that person has heart disease.

The following code will allow kNN to be used for this scenario, the results will be presented inside a confusion matrix:

knn = KNeighborsClassifier(n\_neighbors=5)

# This means that it will classify the data using it's 5 nearest neighbours.

knn.fit(X\_train, y\_train)

y\_pred = knn.predict(X\_test)

cm = confusion\_matrix(y\_test, y\_pred, labels=knn.classes\_)

cm

disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=knn.classes\_)

disp.plot()

plt.show()

acc = accuracy\_score(y\_test, y\_pred)

print("Accuracy of the model is", acc)

A picture containing screenshot, colorfulness, rectangle, square

Description automatically generatedAccuracy of the model is 0.91743119266055

Figure 3 - Confusion matrix

# Evaluation

Within the chapter, the data scientist will evaluate the results of the model used and try to understand why it performed the way it did.

**The outcome**First of all, the accuracy of the model is 91.7%. Whether this is high or low is up to the company or institute behind the project, the data scientist cannot say much about whether this a good or bad results. Of course, when looking at the confusion table, one can see that the model performed well at classifying heart disease. Only 11 false negatives and 7 false positives out of 218 cases.

**The parameters**The kNN model was given the task to find the 5 nearest neighbors to classify whether or not heart disease was in play. When supplied with 6, the accuracy went down to 88% and when supplied with 4 it went down to 90%. Of course these are very small decreases in performance but nonetheless worth mentioning to prove why 5 neighbors is the sweet spot.

The same goes for the testing and training set which is a 25/75 split. When increasing the amount of testing and decreasing the training set to, let’s say, 30/70 the accuracy turns into 90%. This means that the combination of 25/75 training and 5 neighbors allows the model to be most accurate.

# deployment

The deployment phase is meant to deploy the developed model into a (production) environment. This will sometimes call for a total rewrite of the code used to make it more effective, efficient or compatible with other systems (Provost & Fawcett, 2013).

## Deploying the heart model

To deploy the model described in earlier chapters, one would need to take certain steps first to make the code usable:

1. Rewrite it to allow for predicting new unseen data. Right now, the model can only classify the testing data. To use the model in the future for new patient data, it would need to have code that allows a user to load this data into the model.
2. Make the code efficient for the system it’s being deployed in. The current version of the code is for research purposes. If the code is every used one would need to rewrite it to fit the system it is being run on.

Overall, the deployment of this particular code would require a user interface to allow the stakeholders to have a user friendly experience in classifying new patients.

# reflection

Finally, I’d like to reflect on my own work during this project. First of all I started off trying to find an interesting case study which could benefit a certain group of individuals positively. I believe by executing this project with the results achieved, I have done so. However, the process could have been better.

## process evaluation

Before I even started writing the code, I had my eyes on the kNN model to use for this dataset. This is because I had previously experienced it during the classes of data mining. Ideally, one does not know which model to use beforehand to be able to have no bias when executing a data science project. This caused me to not consider other models which could potentially have a better outcome than the achieved one.

Furthermore, I would have liked to compare different cases like this to pick the most impactful one. Right now, I picked a dataset out of the first list on Kaggle which seemed interesting. Luckily it was extremely interesting and fun to do.

## Results evaluation

The results that were achieved are better than the ones I achieved during the data mining course. 91% compared to 78% which I’m quite happy about. I believe what helped was the outlier removal and the correlation check. I really believe that these data checks are essential to knowing for certain that the data is being loaded in correctly. It gives the data scientist a clear view of how the data effects each other and if all the data is realistic to use.

As I said, I would have liked to compare different models to each other to see which bring the most accurate results. Right now, I noticed that the results, while seeming high, don’t tell me much about how good the model is. Compared to what am I supposed to evaluate the results? This questions would have been answered had I used multiple or even one more model than just the kNN.

# ethics

Now that the model stands and the data is loaded into it. It is time to describe the ethical side of this story. The ethics regarding this data is mostly due to it being medical data from real people across the world. Let’s dive into how this should be managed.

The data used in this project is data from various parts of the world and institutes. Scientists have gathered this data and published it for data scientists to research the possibilities of using machine learning to predict heart diseases. The data is gathered from people who were examined and diagnosed with either heart disease or not. Here are a few reasons why it is ethically responsible to use data such as this:

1. It is impossible to trace the person that supplied the data (the person being examined)
   1. All the data is completely anonymous, there are no names or any other personal information supplied inside the dataset.
2. The data is published by an institute with the sole purpose of research such as this.
3. It can benefit the world by creating models that can predict heart disease by simply supplying the model with data of the patient.

The last reason is of course important if the other 2 are present as well. One cannot use reason 3 as the only means to execute a project such as this. If this were the most important factor in data science, then all project would be ethically sound no matter the privacy invasion.

**BIBLIOGRAPHY**

Provost, F., & Fawcett, T. (2013). *Data Science for Business*. Oreilly & Associates Incorporated. Retrieved May 29, 2023

*Kaggle: Your Machine Learning and Data Science Community*. (n.d.). Retrieved May 29, 2023, from https://www.kaggle.com/

*Heart Failure Prediction Dataset*. (2021, September 10). Kaggle. Retrieved May 29, 2023, from https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction