2021

# AI in Enterprise Systems – Final Project Report

TITLE: HEART DISEASE PREDICTOR

AUTHORS: RAMON VILARINS

UMUTCAN ASUTLU

## 1. Introduction

Heart disease refers to several types of heart conditions, varying from coronary artery disease to a heart that does not pump well, for instance. Nowadays, it is common sense that smoking, unhealthy diets and lack of physical exercises increase the risk for having cardiac diseases. With regard to its signs and symptoms, most of the time, it can be suspected through conditions such as: chest pain, shortness of breath and pain in the arms.

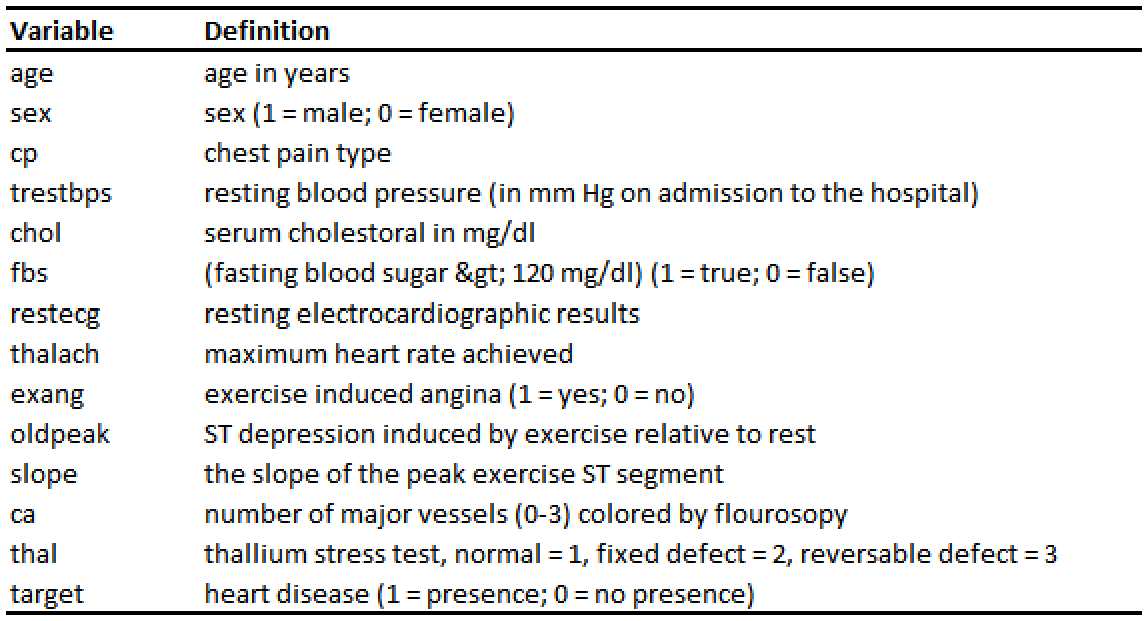
According to the Centers for Disease Control and Prevention, about 610,000 people die of heart disease in the United States every year. Moreover, it estimates that heart diseases cost $219 billion to the country annually. This includes spending with health care services, medicines and lost productivity due to deaths. In Canada, the situation is not different, and every hour 12 Canadian adults age 20+ with diagnosed heart disease die.

Using a dataset provided by the Cleveland Heart Disease, this work aims to apply machine learning techniques to detect hidden patterns and to predict the presence of heart disease. By early detecting this abnormality, it will be possible to act preventively, reducing the risk of heart attacks and heart failures in the future.

## 2. Dataset

Our dataset is composed of 303 entries and includes a total of 14 attributes. In Table 1 we present the definition of all 13 features and the target variable. Except from the feature *oldpeak,* which is a floating-point type, all the remaining ones are integers values. It is worth noting that there are no missing values in the dataset.

**Table 1 - Variables Definition**



| Variable | Definition |
|---|---|
| age | age in years |
| sex | sex (1 = male; 0 = female) |
| cp | chest pain type |
| trestbps | resting blood pressure (in mm Hg on admission to the hospital) |
| chol | serum cholestoral in mg/dl |
| fbs | (fasting blood sugar &gt; 120 mg/dl) (1 = true; 0 = false) |
| restecg | resting electrocardiographic results |
| thalach | maximum heart rate achieved |
| exang | exercise induced angina (1 = yes; 0 = no) |
| oldpeak | ST depression induced by exercise relative to rest |
| slope | the slope of the peak exercise ST segment |
| ca | number of major vessels (0-3) colored by flourosopy |
| thal | thallium stress test, normal = 1, fixed defect = 2, reversable defect = 3 |
| target | heart disease (1 = presence; 0 = no presence) |

In Table 2 we have a statistical description of the dataset. It can be seen that the sample is aged between 29 and 77. Moreover, we observe that, on average, the resting blood pressure is 131.623 and chest pain is 0.966.

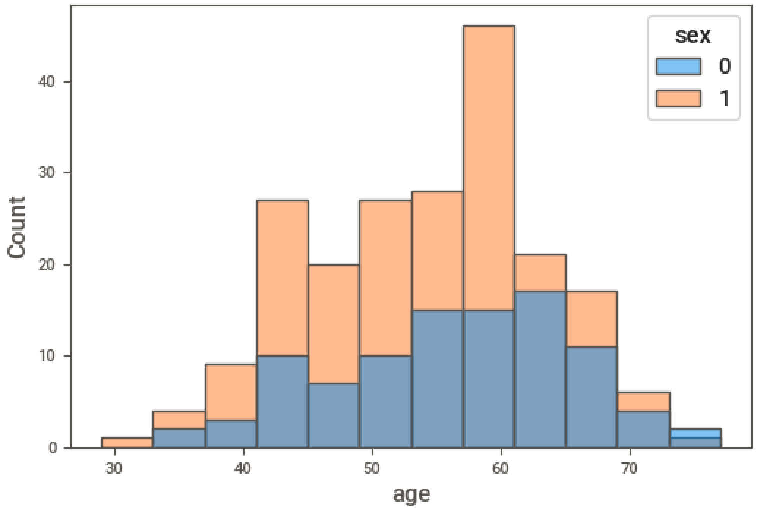
**Table 2 - Dataset Description**

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 303 | 303 | 303 | 303 | 303 | 303 | 303 | 303 | 303 | 303 | 303 | 303 | 303 | 303 |
| **mean** | 54.366 | 0.683 | 0.966 | 131.623 | 246.264 | 0.148 | 0.528 | 149.646 | 0.326 | 1.039 | 1.399 | 0.729 | 2.313 | 0.544 |
| **std** | 9.082 | 0.466 | 1.032 | 17.538 | 51.830 | 0.356 | 0.525 | 22.905 | 0.469 | 1.161 | 0.616 | 1.022 | 0.612 | 0.498 |
| **min** | 29.000 | 0.000 | 0.000 | 94.000 | 126.000 | 0.000 | 0.000 | 71.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| **25%** | 47.500 | 0.000 | 0.000 | 120.000 | 211.000 | 0.000 | 0.000 | 133.500 | 0.000 | 0.000 | 1.000 | 0.000 | 2.000 | 0.000 |
| **50%** | 55.000 | 1.000 | 1.000 | 130.000 | 240.000 | 0.000 | 1.000 | 153.000 | 0.000 | 0.800 | 1.000 | 0.000 | 2.000 | 1.000 |
| **75%** | 61.000 | 1.000 | 2.000 | 140.000 | 274.500 | 0.000 | 1.000 | 166.000 | 1.000 | 1.600 | 2.000 | 1.000 | 3.000 | 1.000 |
| **max** | 77.000 | 1.000 | 3.000 | 200.000 | 564.000 | 1.000 | 2.000 | 202.000 | 1.000 | 6.200 | 2.000 | 4.000 | 3.000 | 1.000 |

## 3. Exploratory Data Analysis

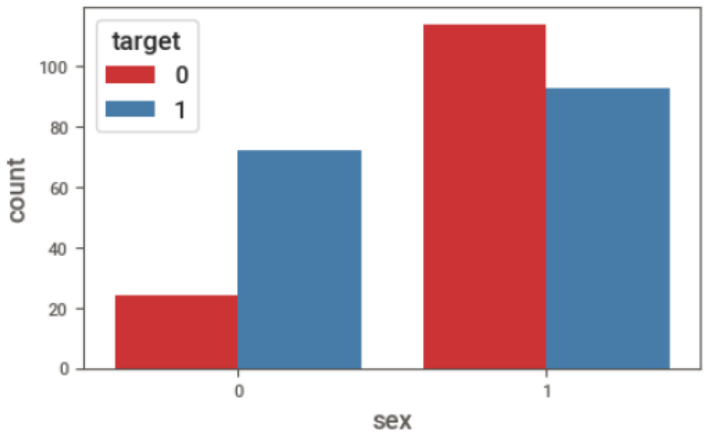
Observing Figure – 1, we can verify that the data contains mores male than females. That is, from a total of 303 entries, 96 are females and 207 males.

**Figure 1 - Age Distribution by Sex**



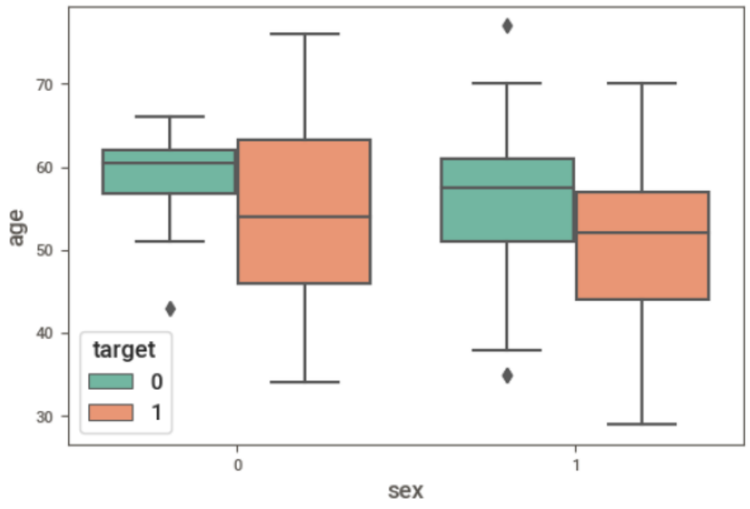
From Figure 2, we note that the number of patients who has heart disease (target = 1) is 165, being 72 females and 93 males. Therefore, the percentage of females with cardiac condition is much higher than the percentage of males.

**Figure 2 – Heart Disease by Sex**



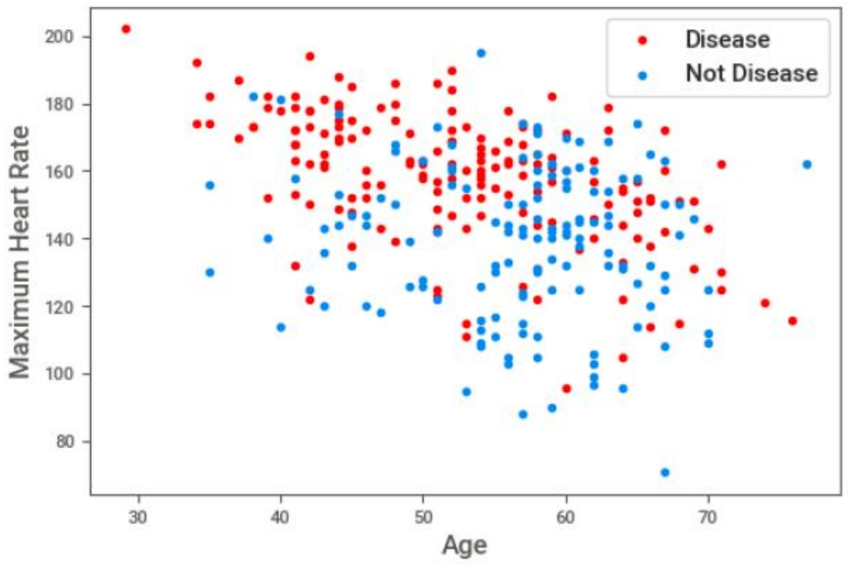
Checking the relationship between age, sex and heart disease in Figure 3, we found that females with positive cardiac condition are on average older than males. That is, according to our data, while most part of the men with heart disease are around between 40 and 60 years, women are between 40 and 65 years old.

**Figure 3 - Heart Disease by sex and age**



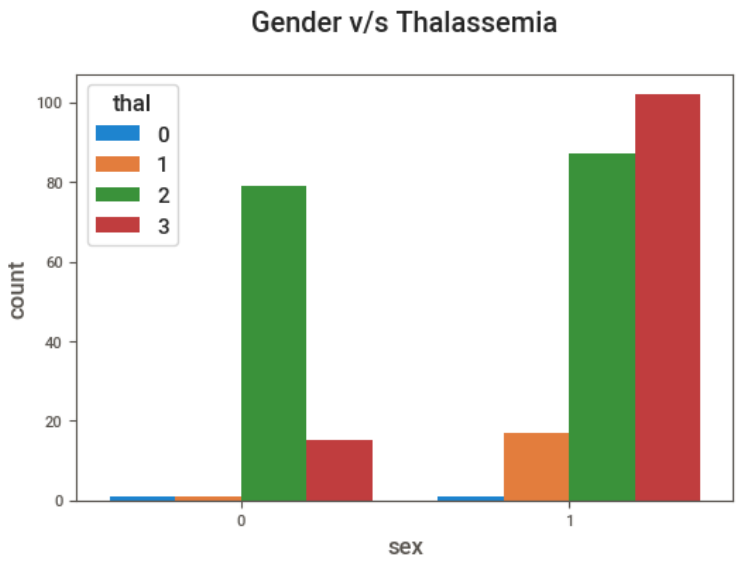
From Figure - 4, one can see that there seems to be a negative correlation between maximum heart rate and age. In addition, there are indications that heart condition is associated with higher heart rates.

**Figure 4 - Maximum Heart Rate by Age**



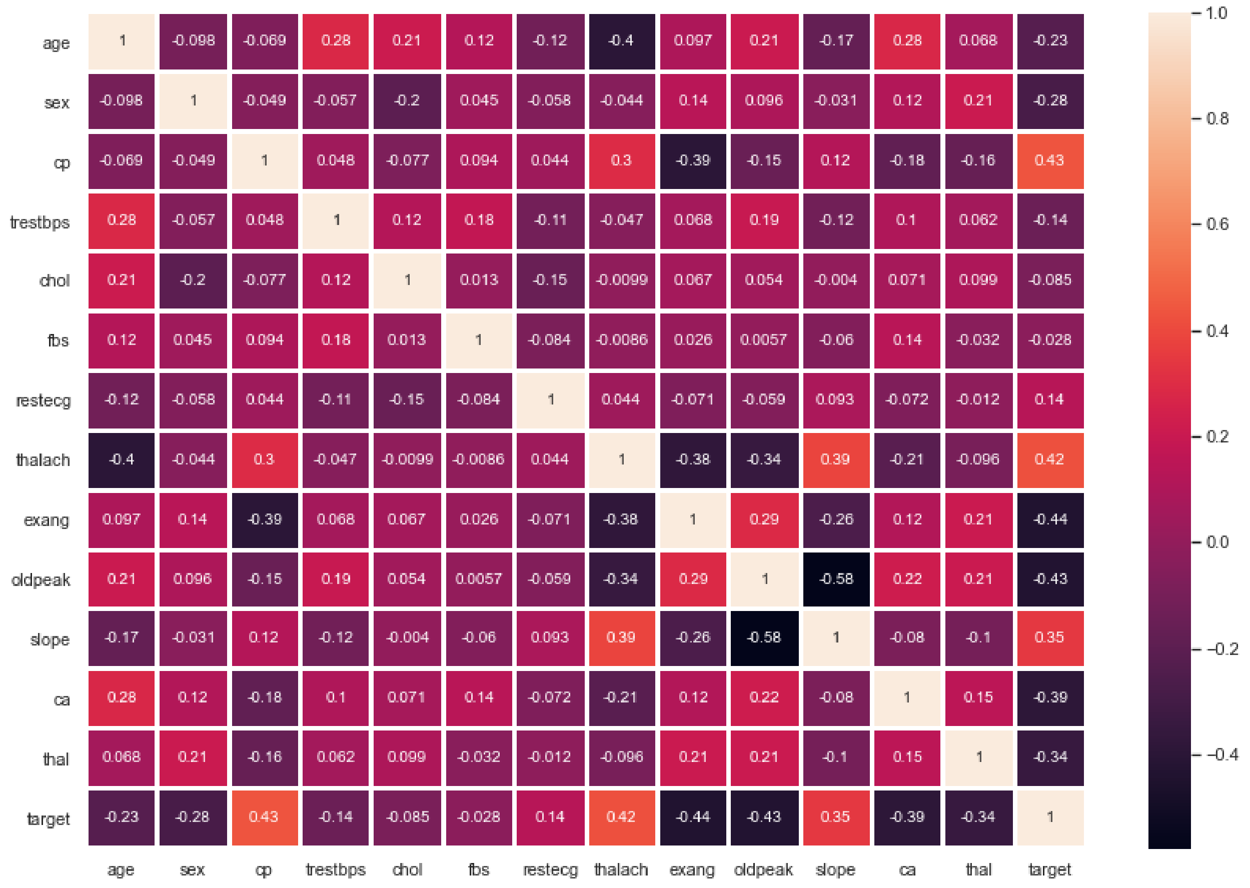
People with thalassemia can get too much iron in their bodies, either from the disease or from frequent blood transfusions. By its turn, too much iron can result in damage to the heart. So, from Figure – 5, we observe that the number of males with Thalassemia (reversable defect, thal = 3) is greater than the number of females.

**Figure 5 - Gender Versus Thalassemia**



A typical symptom of heart attack is chest pain. While examining Figure – 6, we have clear indications that confirm this association. That is, correlation coefficient between chest pain and the target variable is 0.43. Additional examining indicates that maximum heart rate has also a very strong positive correlation with heart condition, reaching 0.42.

**Figure 6 - Correlation Coefficients**



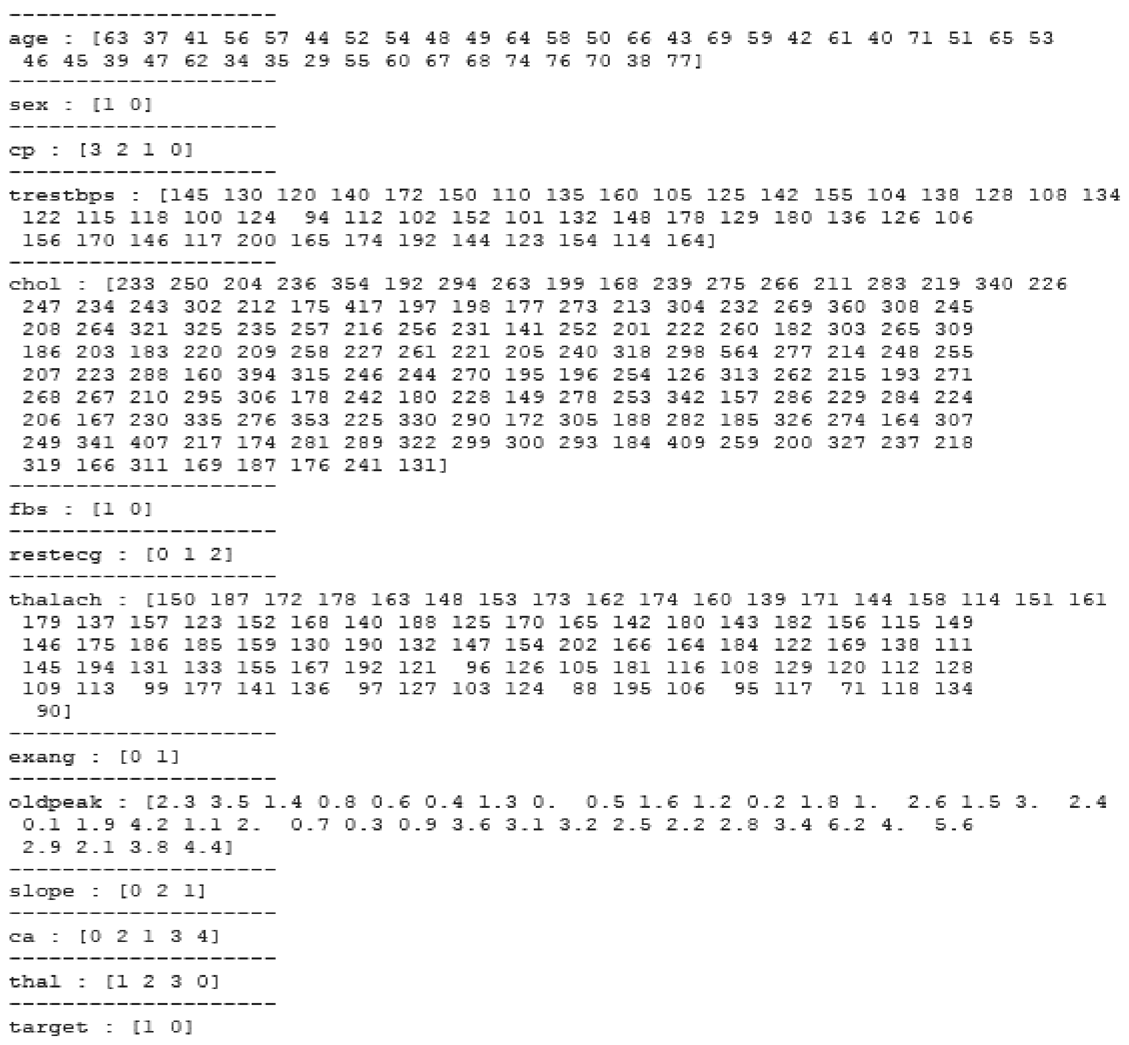
## 4. Modelling

As there is a process of training our machine learning model on a labelled dataset, that is, a dataset in which the target variable is known, it is a supervised model. Moreover, considering that the goal of the model is to predict discrete values, we have a classification problem. So, logistic regression, decision tree and KNN are initially good options of algorithms to train the model.

The features in our data set are at varying scales. For example, Table 3 shows that variables *chol* and *oldpeak* are on completely different scales. Thus, depending on the machine learning model being used, this situation requires some form of data transformation, such as normalization or standardization. Normally, this is necessary for algorithms that calculate the distance between data, like KNN and PCA.

Nonetheless, standardization is not required for logistic regression nor for decision tree or random forests. As these algorithms depend on rules and not on distance, they are not affected by any monotonic transformations of the variables.  As we are using KNN, Logistic Regression and Decision Tree, we opted for standardizing our variables.

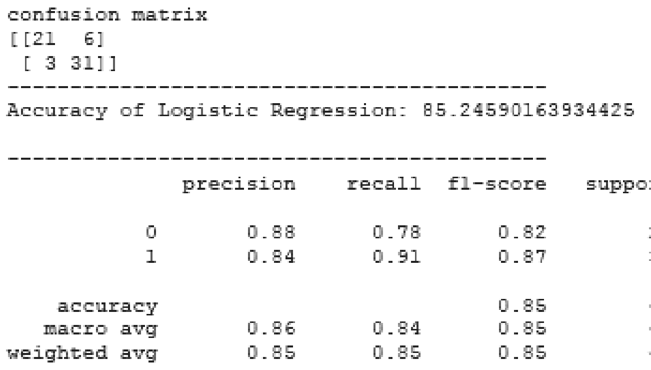
**Table 3 - Feature Engineering**



```
--------------------
age : [63 37 41 56 57 44 52 54 48 49 64 58 50 66 43 69 59 42 61 40 71 51 65 53
 46 45 39 47 62 34 35 29 55 60 67 68 74 76 70 38 77]
--------------------
sex : [1 0]
--------------------
cp : [3 2 1 0]
--------------------
trestbps : [145 130 120 140 172 150 110 135 160 105 125 142 155 104 138 128 108 134
 122 115 118 100 124  94 112 102 152 101 132 148 178 129 180 136 126 106
 156 170 146 117 200 165 174 192 144 123 154 114 164]
--------------------
chol : [233 250 204 236 354 192 294 263 199 168 239 275 266 211 283 219 340 226
 247 234 243 302 212 175 417 197 198 177 273 213 304 232 269 360 308 245
 208 264 321 325 235 257 216 256 231 141 252 201 222 260 182 303 265 309
 186 203 183 220 209 258 227 261 221 205 240 318 298 564 277 214 248 255
 207 223 288 160 394 315 246 244 270 195 196 254 126 313 262 215 193 271
 268 267 210 295 306 178 242 180 228 149 278 253 342 157 286 229 284 224
 206 167 230 335 276 353 225 330 290 172 305 188 282 185 326 274 164 307
 249 341 407 217 174 281 289 322 299 300 293 184 409 259 200 327 237 218
 319 166 311 169 187 176 241 131]
--------------------
fbs : [1 0]
--------------------
restecg : [0 1 2]
--------------------
thalach : [150 187 172 178 163 148 153 173 162 174 160 139 171 144 158 114 151 161
 179 137 157 123 152 168 140 188 125 170 165 142 180 143 182 156 115 149
 146 175 186 185 159 130 190 132 147 154 202 166 164 184 122 169 138 111
 145 194 131 133 155 167 192 121  96 126 105 181 116 108 129 120 112 128
 109 113  99 177 141 136  97 127 103 124  88 195 106  95 117  71 118 134
  90]
--------------------
exang : [0 1]
--------------------
oldpeak : [2.3 3.5 1.4 0.8 0.6 0.4 1.3 0.  0.5 1.6 1.2 0.2 1.8 1.  2.6 1.5 3.  2.4
 0.1 1.9 4.2 1.1 2.  0.7 0.3 0.9 3.6 3.1 3.2 2.5 2.2 2.8 3.4 6.2 4.  5.6
 2.9 2.1 3.8 4.4]
--------------------
slope : [0 2 1]
--------------------
ca : [0 2 1 3 4]
--------------------
thal : [1 2 3 0]
--------------------
target : [1 0]
```

Logistic regression is a linear model and is considered one of the simplest classification models. Thus, it will be the starting point and will set a low bar for future models to surpass. To put it simply, it computes the probability of an event occurrence utilizing logit function where logit = log (p/(1 − p)) = log (probability of occurrence/probability of event not occurrence).  In Table 4 we have the classification report of the logistic regression model. It can be seen that the accuracy level of the model is 85%.

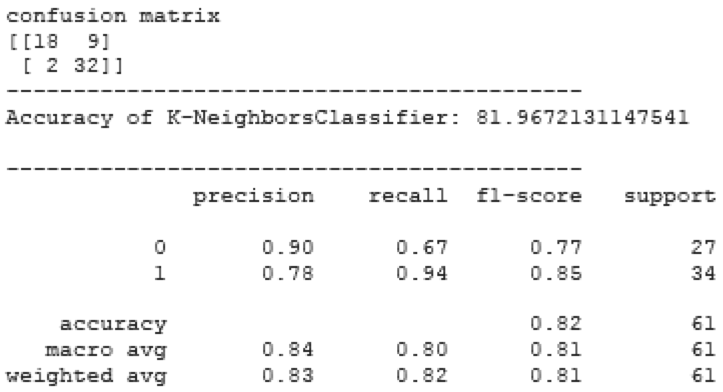
**Table 4 - Logistic Regression Model Classification Report**



```
confusion matrix
[[21  6]
 [ 3 31]]
-------------------------------------------
Accuracy of Logistic Regression: 85.24590163934425

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

           0       0.88      0.78      0.82        27
           1       0.84      0.91      0.87        34

    accuracy                           0.85        61
   macro avg       0.86      0.84      0.85        61
weighted avg       0.85      0.85      0.85        61
```

In Table 5 we have the classification report of the KNN model. It can be seen that the accuracy level of the model is 82%.

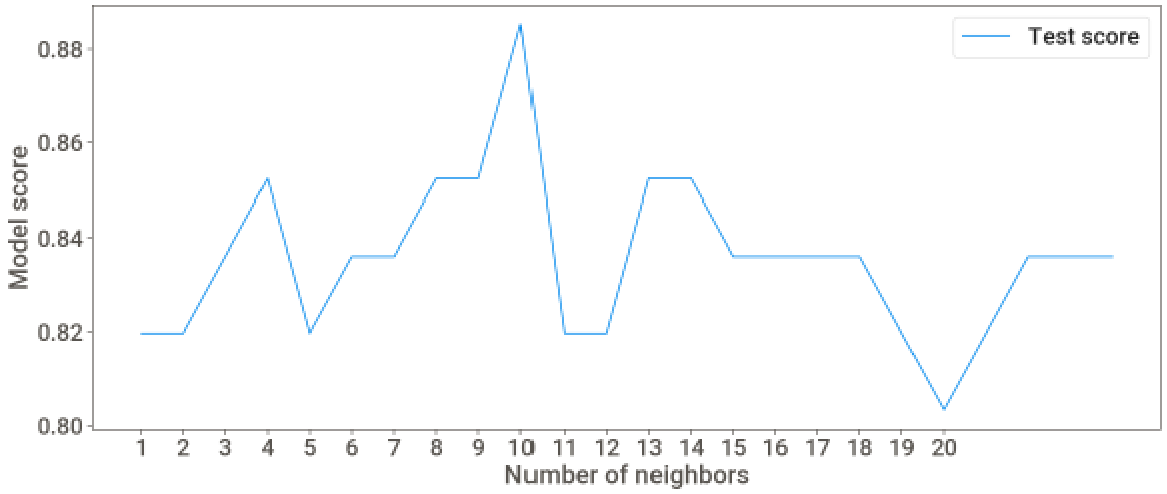
**Table 5 - KNN Model Classification Report**



```
confusion matrix
[[18  9]
 [ 2 32]]
-------------------------------------------
Accuracy of K-NeighborsClassifier: 81.9672131147541

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

           0       0.90      0.67      0.77        27
           1       0.78      0.94      0.85        34

    accuracy                           0.82        61
   macro avg       0.84      0.80      0.81        61
weighted avg       0.83      0.82      0.81        61
```

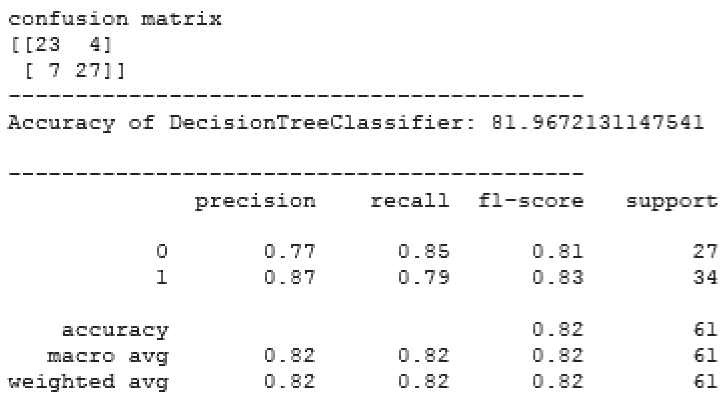
In Figure 7, we show the level of accuracy of the KNN model for different numbers of neighbors. The highest accuracy is reached when the number of neighbors is equal to 10.

**Figure 7 - KNN Model Score**



In Table 6 we have the classification report of the Decision Tree model. It can be seen that the accuracy level of the model is 82%.

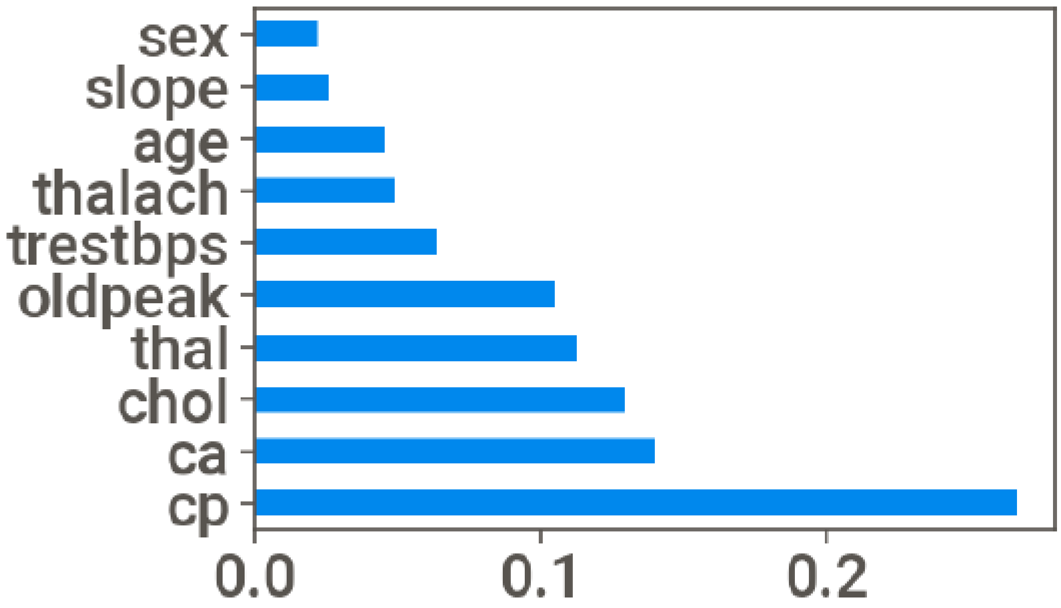
**Table 6 – Decision Tree Model Classification Report**



```
confusion matrix
[[23  4]
 [ 7 27]]
-------------------------------------------
Accuracy of DecisionTreeClassifier: 81.9672131147541

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

           0       0.77      0.85      0.81        27
           1       0.87      0.79      0.83        34

    accuracy                           0.82        61
   macro avg       0.82      0.82      0.82        61
weighted avg       0.82      0.82      0.82        61
```

In Figure 8, we can visualize the feature importances of the decision tree model. It can be seen that *cp*, *ca* and *chol* are the three most important features.

**Figure 8 - Features Importance**



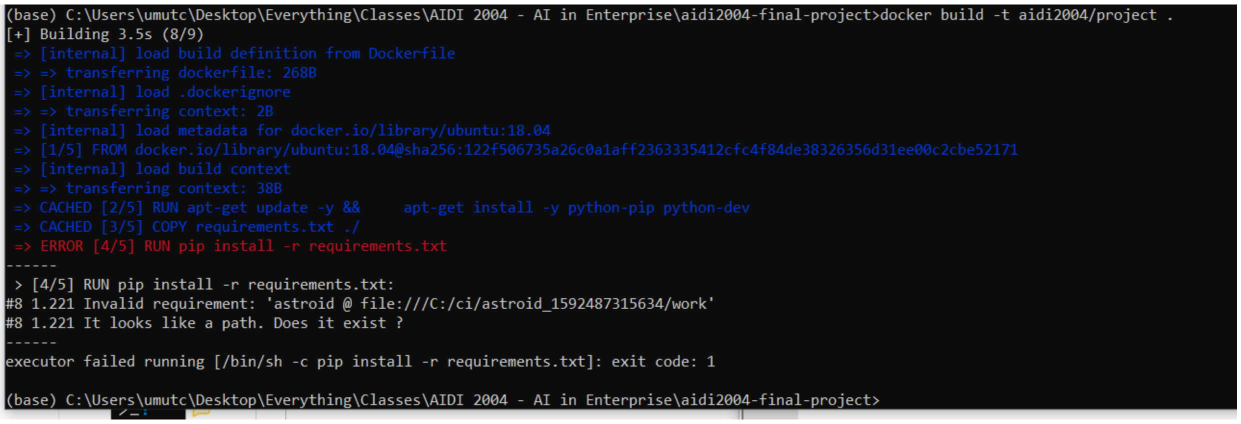
To conclude, we observe that the Logistic Regression model has the best level of accuracy, totaling 85.25%. So, it should be our first choice. However, it is also worth saying that the recall of the Logistic Regression model is 91% while the recall of the KNN model is 94%. Therefore, as in this case it is particularly important to minimize the number of patients with cardiac condition who are classified as a healthy person, we should consider doing further analysis with the KNN model as well.

## 5. Deployment

Deployment is done in two phases. The first one is GitHub deployment. Throughout the project GitHub is used effective to maintain communication and development process between team members. The GitHub link for the project:

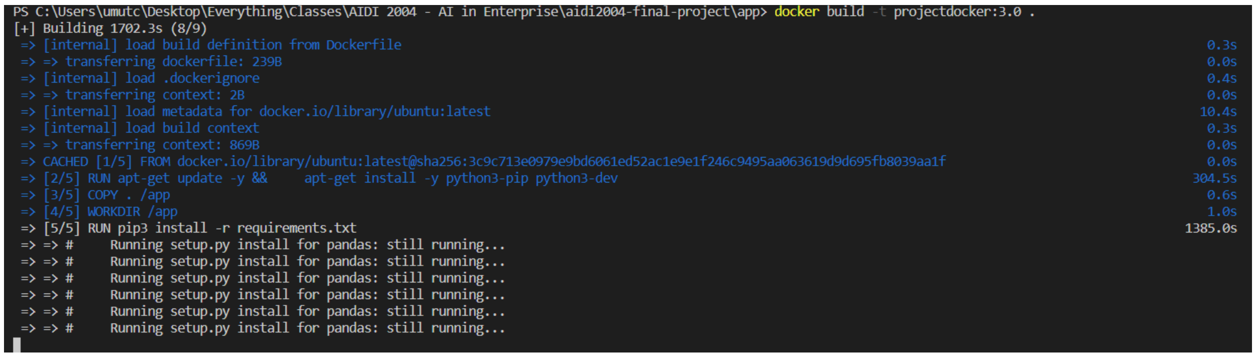
https://github.com/umutcanasutlu/aidi2004-final-project

For the docker deployment, it was not as smooth as expected. We created the requirements file using “pip freeze > requirement.txt” but it did not work.



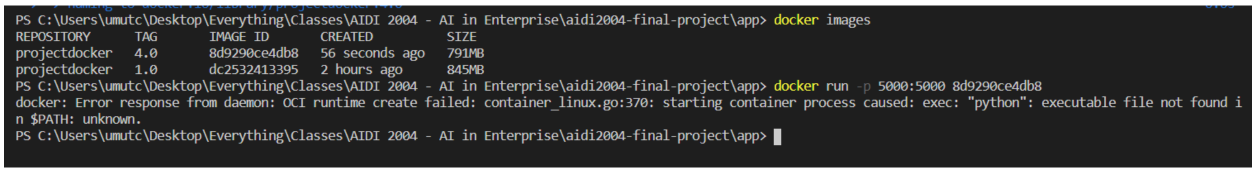
```
(base) C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project>docker build -t aidi2004/project .
[+] Building 3.5s (8/9)
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 268B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/ubuntu:18.04
 => [1/5] FROM docker.io/library/ubuntu:18.04@sha256:122f506735a26c0a1aff2363335412cfc4f84de38326356d31ee00c2cbe52171
 => [internal] load build context
 => => transferring context: 38B
 => CACHED [2/5] RUN apt-get update -y &&     apt-get install -y python-pip python-dev
 => CACHED [3/5] COPY requirements.txt ./
 => ERROR [4/5] RUN pip install -r requirements.txt
------
 > [4/5] RUN pip install -r requirements.txt:
#8 1.221 Invalid requirement: 'astroid @ file:///C:/ci/astroid_1592487315634/work'
#8 1.221 It looks like a path. Does it exist ?
------
executor failed running [/bin/sh -c pip install -r requirements.txt]: exit code: 1

(base) C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project>
```

Afterwards we had to prepare al the requirements manually, it was an easy task but as we learned from our research as well, Pandas is not very integrable with docker to be used. Due to its inner files, it takes around 20 minutes for docker to be deployed. As in the figure below you can see it was still running for pandas after 1385 seonds.



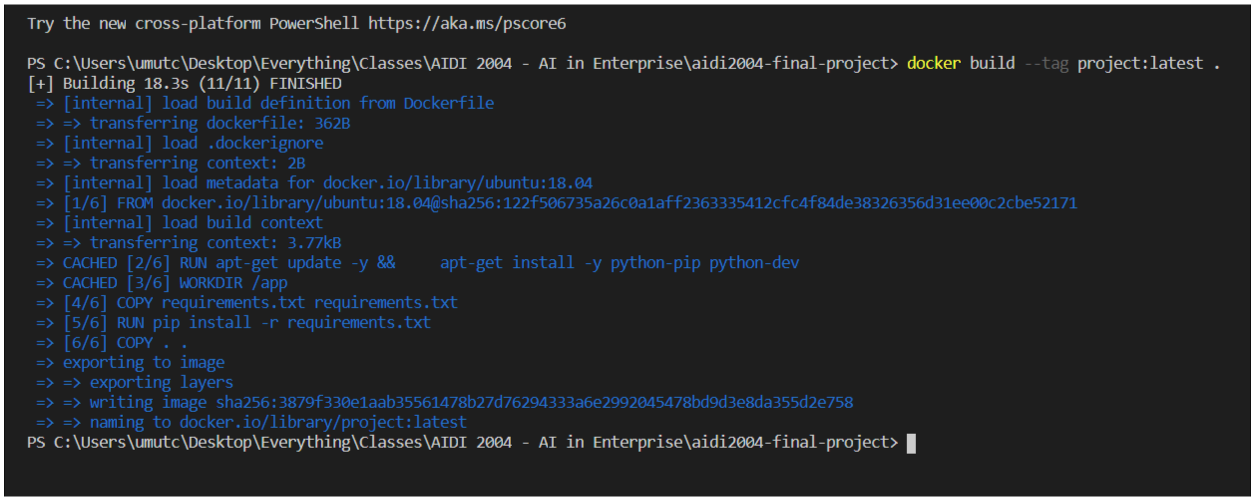
```
PS C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project\app> docker build -t projectdocker:3.0 .
[+] Building 1702.3s (8/9)
 => [internal] load build definition from Dockerfile                                                                          0.3s
 => => transferring dockerfile: 239B                                                                                          0.0s
 => [internal] load .dockerignore                                                                                             0.4s
 => => transferring context: 2B                                                                                               0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest                                                             10.4s
 => [internal] load build context                                                                                             0.3s
 => => transferring context: 869B                                                                                             0.0s
 => CACHED [1/5] FROM docker.io/library/ubuntu:latest@sha256:3c9c713e0979e9bd6061ed52ac1e9e1f246c9495aa063619d9d695fb8039aa1f  0.0s
 => [2/5] RUN apt-get update -y &&     apt-get install -y python3-pip python3-dev                                           304.5s
 => [3/5] COPY . /app                                                                                                         0.6s
 => [4/5] WORKDIR /app                                                                                                        1.0s
 => [5/5] RUN pip3 install -r requirements.txt                                                                             1385.0s
 => => #     Running setup.py install for pandas: still running...
 => => #     Running setup.py install for pandas: still running...
 => => #     Running setup.py install for pandas: still running...
 => => #     Running setup.py install for pandas: still running...
 => => #     Running setup.py install for pandas: still running...
 => => #     Running setup.py install for pandas: still running...
```

After many attempts with docker with 20 minutes build time it was obvious that we had not enough time to solve it with pandas. Then we had to eliminate the ‘pandas’ library. To do that we had to use ‘joblib’ library from Sklearn. After that we implemented a Numpy solution which required minimum amount of external libraries, however; it was giving this error after building the image.

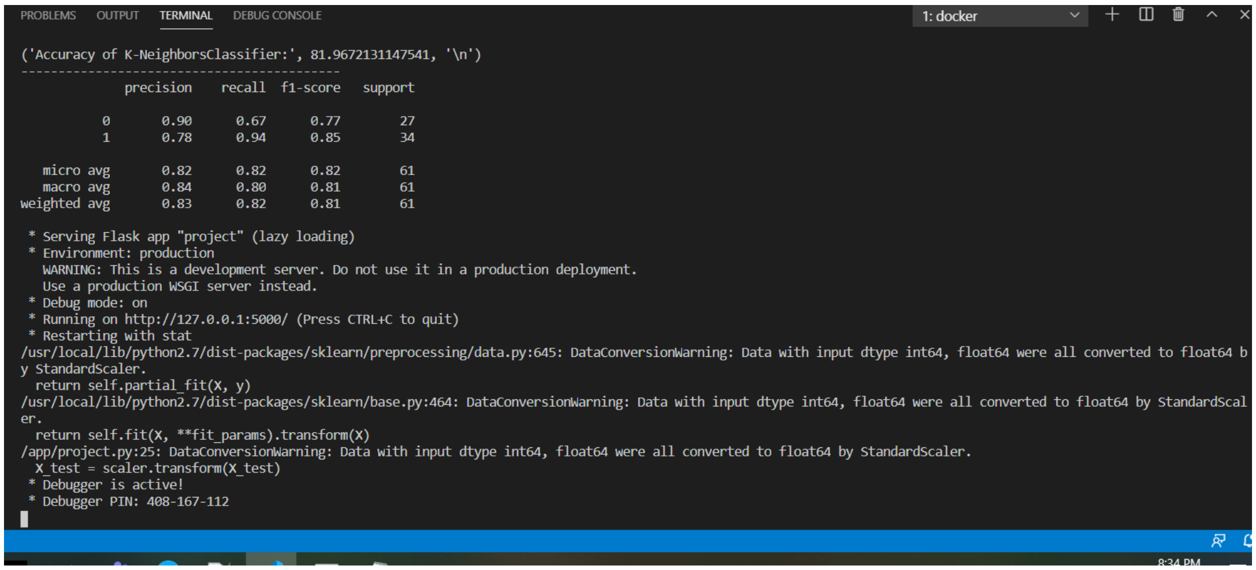


```
PS C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project\app> docker images
REPOSITORY      TAG       IMAGE ID       CREATED          SIZE
projectdocker   4.0       8d9290ce4db8   56 seconds ago   791MB
projectdocker   1.0       dc2532413395   2 hours ago      845MB
PS C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project\app> docker run -p 5000:5000 8d9290ce4db8
docker: Error response from daemon: OCI runtime create failed: container_linux.go:370: starting container process caused: exec: "python": executable file not found i
n $PATH: unknown.
PS C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project\app>
```

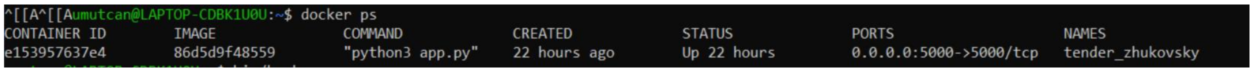
Finally we managed to build proper image that can be run without any further problems.



```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project> docker build --tag project:latest .
[+] Building 18.3s (11/11) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 362B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/ubuntu:18.04
 => [1/6] FROM docker.io/library/ubuntu:18.04@sha256:122f506735a26c0a1aff2363335412cfc4f84de38326356d31ee00c2cbe52171
 => [internal] load build context
 => => transferring context: 3.77kB
 => CACHED [2/6] RUN apt-get update -y &&     apt-get install -y python-pip python-dev
 => CACHED [3/6] WORKDIR /app
 => [4/6] COPY requirements.txt requirements.txt
 => [5/6] RUN pip install -r requirements.txt
 => [6/6] COPY . .
 => exporting to image
 => => exporting layers
 => => writing image sha256:3879f330e1aab35561478b27d76294333a6e2992045478bd9d3e8da355d2e758
 => => naming to docker.io/library/project:latest
PS C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\aidi2004-final-project>
```



```
('Accuracy of K-NeighborsClassifier:', 81.9672131147541, '\n')
------------------------------------------------
              precision    recall  f1-score   support

           0       0.90      0.67      0.77        27
           1       0.78      0.94      0.85        34

   micro avg       0.82      0.82      0.82        61
   macro avg       0.84      0.80      0.81        61
weighted avg       0.83      0.82      0.81        61

 * Serving Flask app "project" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
/usr/local/lib/python2.7/dist-packages/sklearn/preprocessing/data.py:645: DataConversionWarning: Data with input dtype int64, float64 were all converted to float64 b
y StandardScaler.
  return self.partial_fit(X, y)
/usr/local/lib/python2.7/dist-packages/sklearn/base.py:464: DataConversionWarning: Data with input dtype int64, float64 were all converted to float64 by StandardScal
er.
  return self.fit(X, **fit_params).transform(X)
/app/project.py:25: DataConversionWarning: Data with input dtype int64, float64 were all converted to float64 by StandardScaler.
  X_test = scaler.transform(X_test)
 * Debugger is active!
 * Debugger PIN: 408-167-112
```



```
^[[A^[[Aumutcan@LAPTOP-CDBK1U0U:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                    NAMES
e153957637e4        86d5d9f48559        "python3 app.py"    22 hours ago        Up 22 hours         0.0.0.0:5000->5000/tcp   tender_zhukovsky
```

However; after this step, we got stuck with web app view.