Data Science and AI : Mini Project

GitHub link : *https://github.com/KentinGuillemot/Data-AI*

Question 2 : Flower

I started to inform myself about the dataframe so I did a **.shape** to have the number of row and columns and I did a **.describe()** to have just the important informations about the different features.

Then I separated the features and the targets in X and Y, for the X I took all the dataframe without the last column and this last column represent the Y (the targets). I needed to separate the X and Y into a training dataframe and a test dataframe, for this I used the function **train_test_split** that I imported from sklearn.model_selection previously in the code. I chose test_size=0.25 and random_state=32 as parameters for the train_test_split.

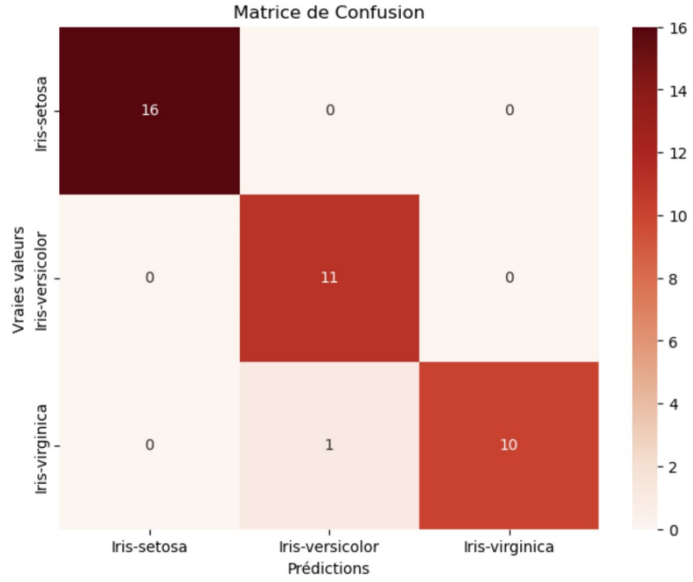
Then I decided to create a function **train_and_evaluate_model** that I will use everytime I need to train a new model et see what are his statistics. In this function I'm training the model with **.fit()** and I'm asking him some predictions with **.predict**, then I print the accuracy of the model with **accuracy_score()**, the classification report with **classification_report()** and the confusion matrix with **confusion_matrix().** All this information will be useful to see the evolution of the model.

Now it's time to create the model and to train him and see what kind of result do we have. I create a random forest model with the following line of code :

"rf_model = RandomForestClassifier(n_estimators=50, random_state=32)"

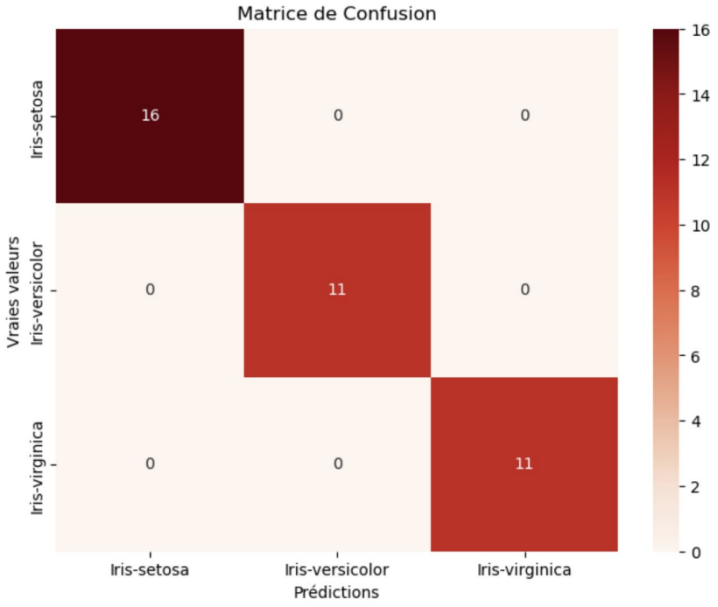
Then I trained him with the **train_and_evaluate_model** function and I had the following results :

```
Accuracy : 97.37%

Classification_report :
                precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        16
Iris-versicolor       0.92      1.00      0.96        11
 Iris-virginica       1.00      0.91      0.95        11

       accuracy                           0.97        38
      macro avg       0.97      0.97      0.97        38
   weighted avg       0.98      0.97      0.97        38
```

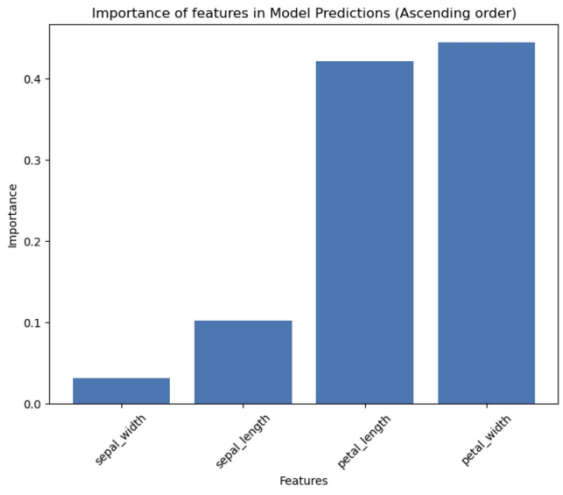


I decided to make a second model and change the parameters, I put random_state=42 instead of random_state=32 and I had the following results :

```
Accuracy : 100.00%

Classification_report :
                precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        16
Iris-versicolor       1.00      1.00      1.00        11
 Iris-virginica       1.00      1.00      1.00        11

       accuracy                           1.00        38
      macro avg       1.00      1.00      1.00        38
   weighted avg       1.00      1.00      1.00        38
```



Now that I have a model with 100% of accuracy I decided to make a graph that shows us the importance of each features into the prediction, what are their influence :



Finally I made a function **flower()** that ask the user the different characteristic of the flower (sepal_length, sepal_width…) and return wich type of iris it should be.

Question 3 : Titanic

At the beginning we have 3 document gender_submission.csv, test.csv, train.csv. After analysing all this documents on excel I identified the utility of each document and creatd 3 dataframes :

y_test = pd.read_csv('gender_submission.csv')

X_test = pd.read_csv('test.csv')

train = pd.read_csv('train.csv')

We can see that we almost have all the train/test dataframe so we will not have to use the train_test_split function.

Then I made some .shape, .head() on each dataframe to have some information on the lines, the columns… After this I decided to drop all the columns that have no influence on the prediction, such as “Cabin”, “Name”, “Ticket”, ”PassengerID” .

Then I identified in “train” the columns that contain NAN values that could possibly return us some errors and I dropped all the lines that have NaN values. For X_test it’s a bit different because all the lines that I drop I need to drop them also in y_test so I made I loop that in a first time drop the lines in y_test with the index of the lines that contains NaN values in the columns of X_test. And then I did it on X_test.

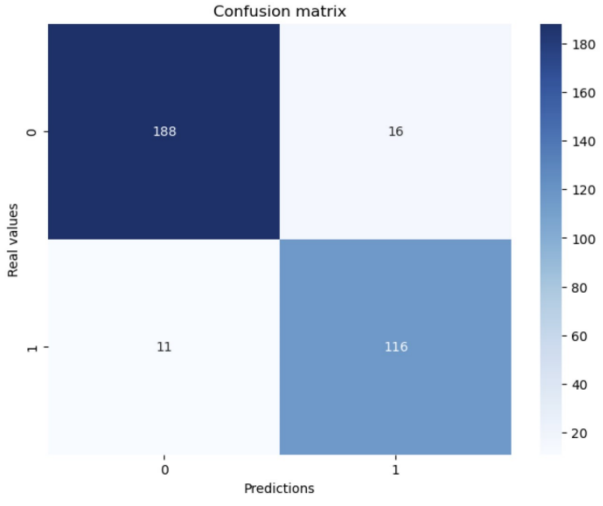
Now I need to separate the dataframe train into 2 dataframe X_train and y_train, X_train will be all train but just without the column “Survived” contrary to y_train that will be only the column “Survived”.

Then I needed to put all the string values remaining in the dataframe into int or float values, so I used the function **.map()**. I did this for the “Sex” column with 'male': 0, 'female': 1 and for the “Embarked” columns with 'S': 0, 'C': 1,'Q' : 2.

I got some problems with NaN values so I decided to make another verification on X_test and x_train if they still have NaN values. This test showed that X_test still have NaN value and I dropped it.

Now that I cleaned all the data it’s time to create the model and evaluate it, like in the flower I made a function train_and_evaluate_model that shows us the same statistics then I created the Logistic Regression model and I put it into the functions. I had the following result :

```
Accuracy : 91.84%

Classification report :
              precision    recall  f1-score   support

           0       0.94      0.92      0.93       204
           1       0.88      0.91      0.90       127

    accuracy                           0.92       331
   macro avg       0.91      0.92      0.91       331
weighted avg       0.92      0.92      0.92       331
```



I made a ROC curve as well as a learning curve to visualize the evolution of the model in more detail.

Finally I made a function **is_alive()** that ask the user the different characteristic of a person and return if he most likely to survive or to die.