Data Mining Project: Covid-19 Analysis

Réalisé par :

EL MAHFOUD RADOUANE

NOUALI TAHA

Encadré par : IKRAM EL ASRI

Filière : Data Engineer

2022-2023

Plan:

- Introduction
- Covid Analysis and prediction
  - Data Presentation
  - Data Exploration
  - Data Preprocessing and Modeling
- Implementation with Django
- App Dockerizing

Introduction :

Data Mining and machine learning have played a significant role in predicting COVID-19 test results. Data mining is the process of discovering patterns and insights from large datasets, while machine learning is a subset of artificial intelligence that allows systems to automatically learn and improve from experience without being explicitly programmed. By using these techniques, researchers and healthcare professionals have been able to build models that can predict COVID-19 test results with high accuracy. These models analyze a range of factors, such as age, symptoms, exposure history, and comorbidities, to predict the likelihood of a positive or negative test result. The ability to accurately predict COVID-19 test results can help healthcare professionals triage patients, allocate resources, and take appropriate measures to prevent the spread of the disease.

In addition to predicting COVID-19 test results, data mining and machine learning can also be used to develop new diagnostic tests for the disease. Machine learning algorithms can analyze vast amounts of data from different sources, such as patient records, genetic information, and epidemiological data, to identify new biomarkers or

patterns that could be used to develop more accurate and efficient diagnostic tests. This could lead to the development of tests that are faster, less invasive, and more reliable, which would be particularly useful in low-resource settings or areas where there are limited testing facilities. Overall, data mining and machine learning have the potential to revolutionize the way we diagnose and manage COVID-19, and could help to mitigate the impact of the pandemic on public health and the economy.

In addition to healthcare, AI and ML have applications in a range of other industries, such as finance, manufacturing, and transportation. For example, in finance, AI and ML can be used to analyze large datasets to identify patterns and make predictions about market trends, which can help investors to make better-informed decisions. In manufacturing, these technologies can be used to optimize production processes, reduce waste, and improve quality control. In transportation, AI and ML can be used to improve the safety and efficiency of autonomous vehicles.

Covid Analysis and prediction :

1. Data presentation

The World Health Organization (WHO) characterized the COVID-19, caused by the SARS-CoV-2, as a pandemic on March 11, while the exponential increase in the number of cases was risking to overwhelm health systems around the world with a demand for ICU beds far above the existing capacity, with regions of Italy being prominent examples.

Brazil recorded the first case of SARS-CoV-2 on February 26, and the virus transmission evolved from imported cases only, to local and finally community transmission very rapidly, with the federal government declaring nationwide community transmission on March 20.

Until March 27, the state of São Paulo had recorded 1,223 confirmed cases of COVID-19, with 68 related deaths, while the county of São Paulo, with a population of approximately 12 million people and where Hospital Israelita Albert Einstein is located, had 477 confirmed cases and 30 associated death, as of March 23. Both the state and the county of São Paulo decided to establish quarantine and social distancing measures, that will be enforced at least until

early April, in an effort to slow the virus spread.

One of the motivations for this project is the fact that in the context of an overwhelmed health system with the possible limitation to perform tests for the detection of SARS-CoV-2, testing every case would be impractical and tests results could be delayed even if only a target subpopulation would be tested.

The dataset was obtained from the Kaggle https://www.kaggle.com/datasets/einsteindata4u/covid19

This dataset contains anonymized data from patients seen at the Hospital Israelita Albert Einstein, at São Paulo, Brazil, and who had samples collected to perform the SARS-CoV-2 RT-PCR and additional laboratory tests during a visit to the hospital.

- `**Patient ID**`: A unique identifier for each patient in the dataset.

- `**Patient age quantile**`: A categorical variable that indicates the age group the patient belongs to, ranging from 0 to 19 years old.

- `**SARS-Cov-2 exam result**`: A binary variable indicating whether the patient tested positive (1) or negative (0) for SARS-Cov-2, the virus responsible for COVID-19.

- `**Patient admitted to regular ward (1=yes, 0=no)**`: A binary variable indicating whether the patient was admitted to a regular hospital ward during their hospital stay.

- `Patient admitted to intensive care unit (1=yes, 0=no)`: A binary variable indicating whether the patient was admitted to an intensive care unit during their hospital stay.

- `Hematocrit`: The proportion of red blood cells in the blood.

- `Patient admitted to semi-intensive unit (1=yes, 0=no)`: A binary variable indicating whether the patient was admitted to a semi-intensive care unit during their hospital stay.

Along with the blood test columns, the analysis encompasses several other columns pertaining to viral testing and other diagnostic procedures, showcasing the diverse approaches utilized to detect and manage infectious diseases.

2. Data Exploration

This step consists of exploring data and getting many insights from histograms,bar plots and other graphs.

Shape Analysis

the target is `SARS-Cov-2 exam result`

Shape:

exploring data

```
df.shape
```

```
(5644, 111)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5644 entries, 0 to 5643
Columns: 111 entries, Patient ID to ctO2 (arterial blood gas analysis)
dtypes: float64(70), int64(4), object(37)
memory usage: 4.8+ MB
```

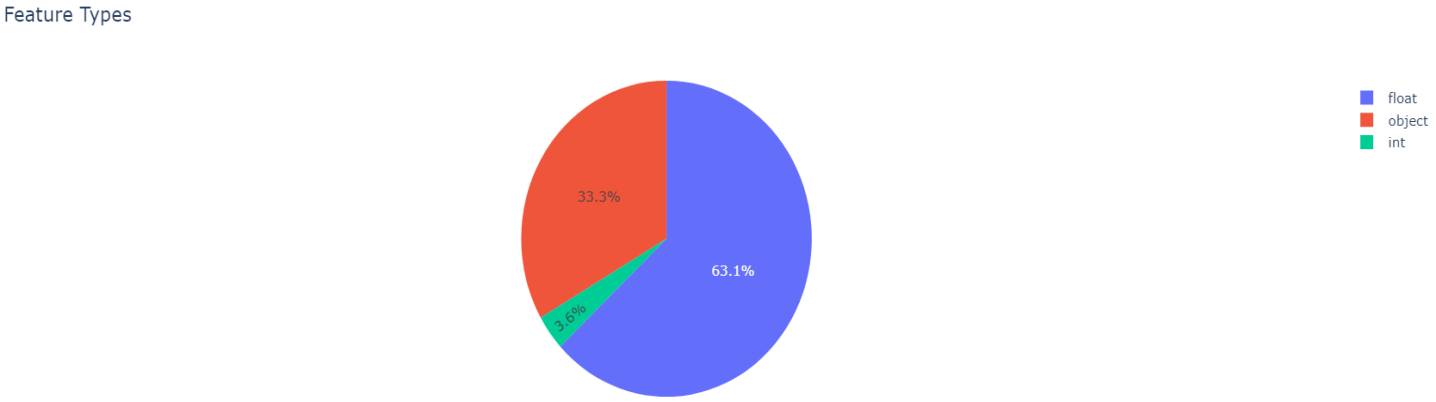
The data contains 111 columns and 5644 rows

Types of variables:

```
import plotly.express as px

dtypes_count=pd.DataFrame(zip(list(df.dtypes.value_counts().index),list(df.dtypes.value_counts())),columns=["types","count"])
# Create a sample DataFrame
data = {'types': ['float', 'int', 'object'],
        'count': [70, 4, 37]}
dtypes_count = pd.DataFrame(data)

# Create the pie chart
fig = px.pie(dtypes_count, names="types", values='count', title='Feature Types')

# Display the chart
fig.show()
```

[12]

`



74 variables of `numeric` type(70 `float`,4 `int`) and 37 of type `object.

Missing Values:

```
(df.isna().sum()/df.shape[0]).sort_values(ascending=False)
```

```
Output exceeds the size limit. Open the full output data in a text editor
Urine - Sugar                                          1.000000
Mycoplasma pneumoniae                                  1.000000
Partial thromboplastin time (PTT)                      1.000000
Prothrombin time (PT), Activity                        1.000000
D-Dimer                                                1.000000
Fio2 (venous blood gas analysis)                       0.999823
Urine - Nitrite                                        0.999823
Vitamin B12                                            0.999468
Lipase dosage                                          0.998583
Albumin                                                0.997697
Phosphor                                               0.996456
Arteiral Fio2                                          0.996456
Ferritin                                               0.995925
Arterial Lactic Acid                                   0.995216
ctO2 (arterial blood gas analysis)                     0.995216
Hb saturation (arterial blood gases)                   0.995216
Total CO2 (arterial blood gas analysis)                0.995216
pCO2 (arterial blood gas analysis)                     0.995216
Base excess (arterial blood gas analysis)              0.995216
pO2 (arterial blood gas analysis)                      0.995216
HCO3 (arterial blood gas analysis)                     0.995216
pH (arterial blood gas analysis)                       0.995216
Magnesium                                              0.992913
Ionized calcium                                        0.991141
```

A lot of Nan (half of columns > 90% Nan)

2 groupes :

- **76%NAN : viral Test**
- **89% NAN: blood test**

Deep Analysis

1. exploring target variable:

```
df["SARS-Cov-2 exam result"].value_counts(normalize=True)
```

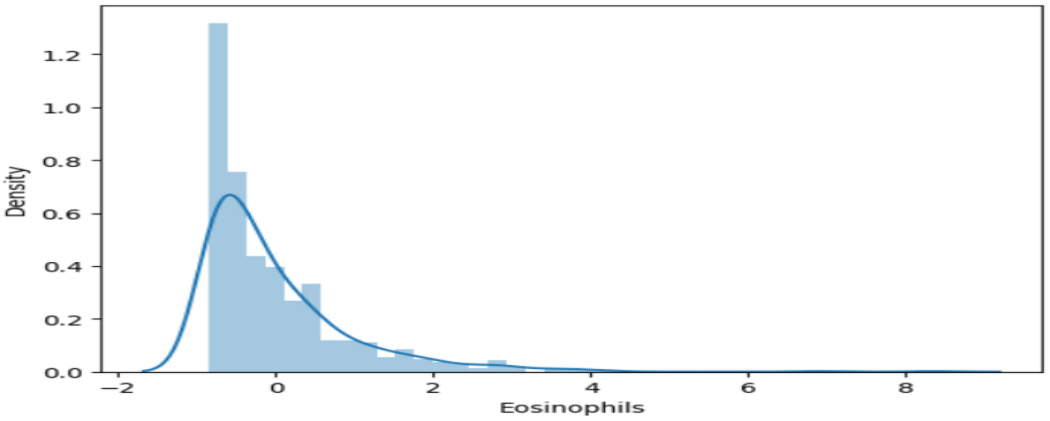
```
negative    0.901134
positive    0.098866
Name: SARS-Cov-2 exam result, dtype: float64
```

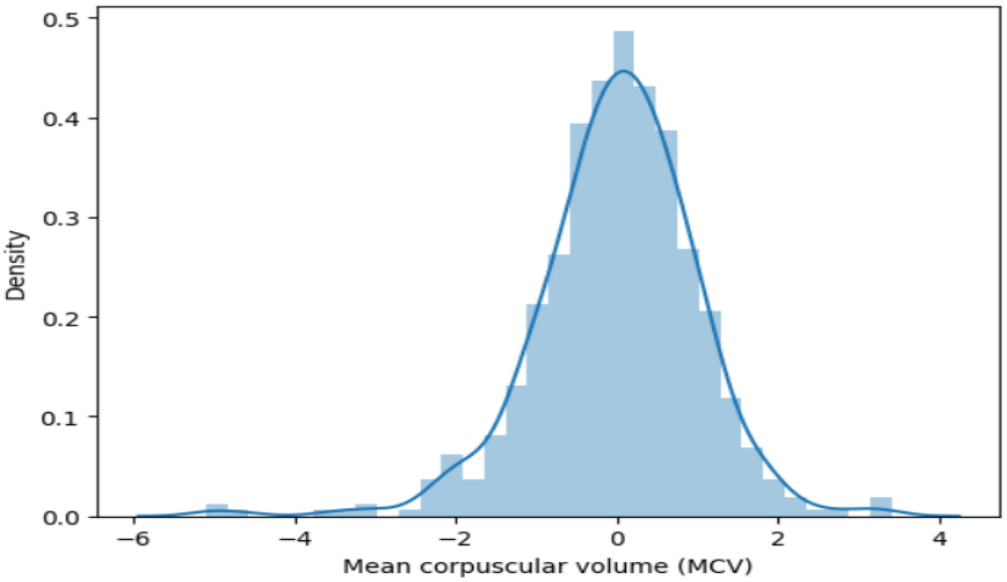
from the snippet of code above we have 90% negative and 10% positive so we a challenge of imbalanced dataset (we must be careful not to use accuracy for evaluation)

2. exploring independent variables:

let's start by visualizing continuous features

```
for col in df.select_dtypes('float'):
    plt.figure()
    sns.distplot(df[col])
```





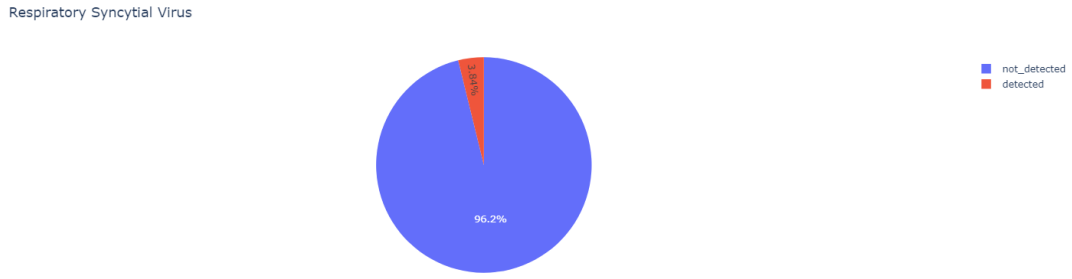
we notice from graphs above that some features follow normal distributions

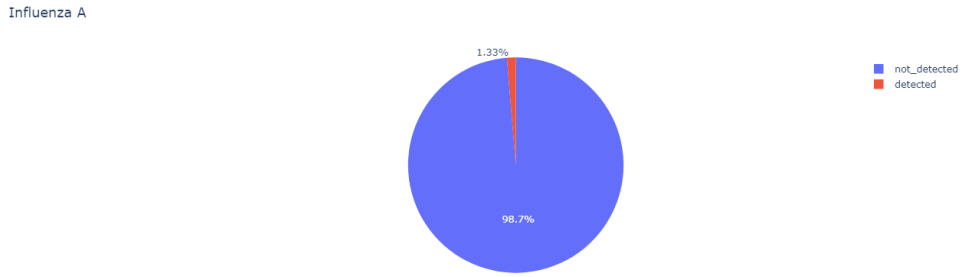
for categorical features we need to use barplots:

```
categorical_values=df.select_dtypes("object").columns
```

```
cat_values=['Respiratory Syncytial Virus', 'Influenza A',
       'Influenza B', 'Parainfluenza 1', 'CoronavirusNL63',
       'Rhinovirus/Enterovirus', 'Coronavirus HKU1', 'Parainfluenza 3',
       'Chlamydophila pneumoniae', 'Adenovirus', 'Parainfluenza 4',
       'Coronavirus229E', 'CoronavirusOC43', 'Inf A H1N1 2009',
       'Bordetella pertussis', 'Metapneumovirus', 'Parainfluenza 2',
       'Influenza B, rapid test', 'Influenza A, rapid test']
```

```
for col in cat_values:

    df_=pd.DataFrame(zip(df[col].value_counts().index,list(df[col].value_counts())),columns=["index","values"])
    fig = px.pie(df_, names="index", values='values', title=f'{col}')
    fig.show()
```





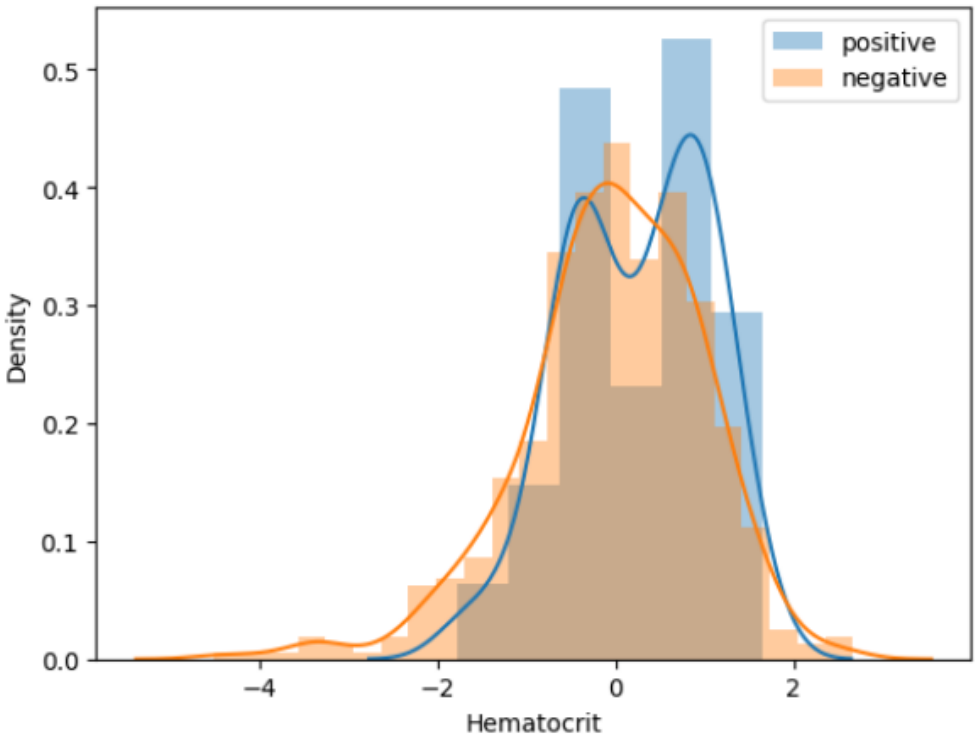
we notice :

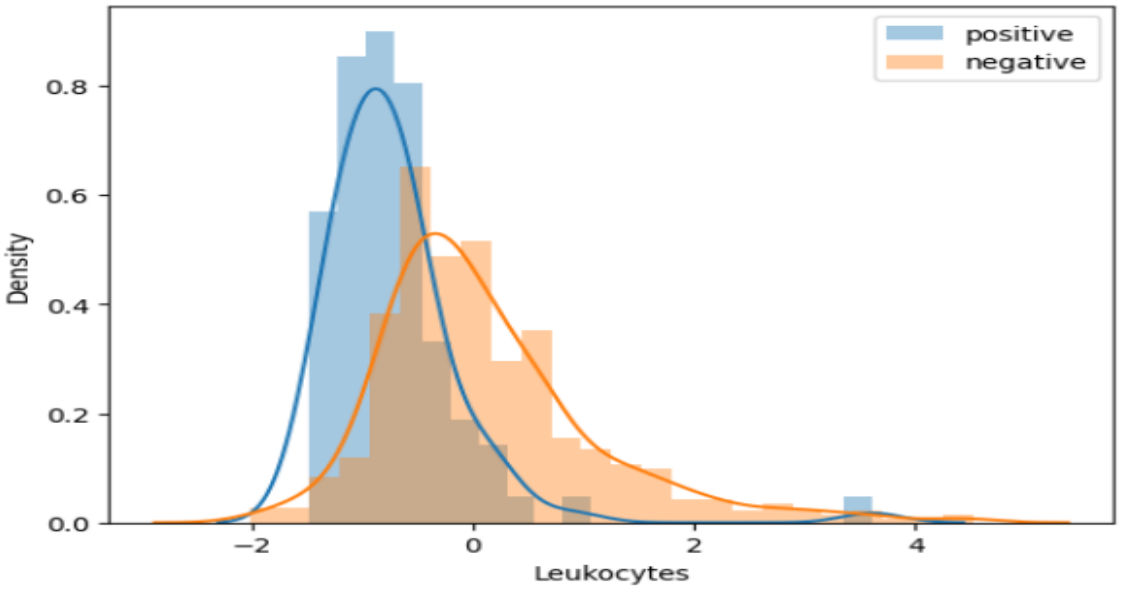
* That `detected` category in each test is a rare minority compared to `non detected` class .
* in the most of features the 'detected' class is less than 10% ,except Rhinovirus (28% 'detected' and 72% 'non detected')

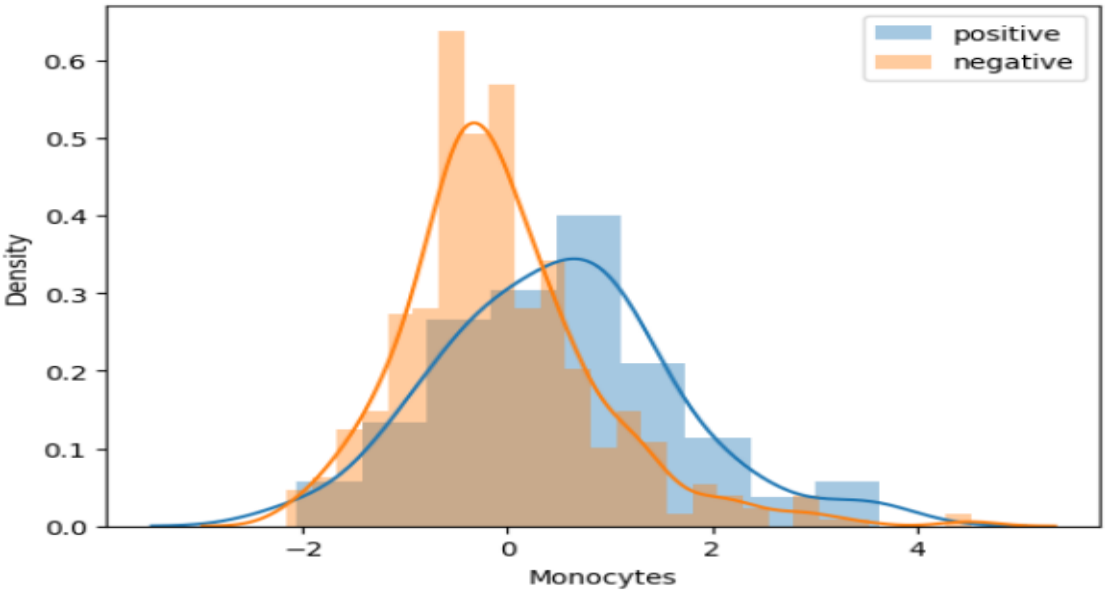
3. exploring relation target-variables:

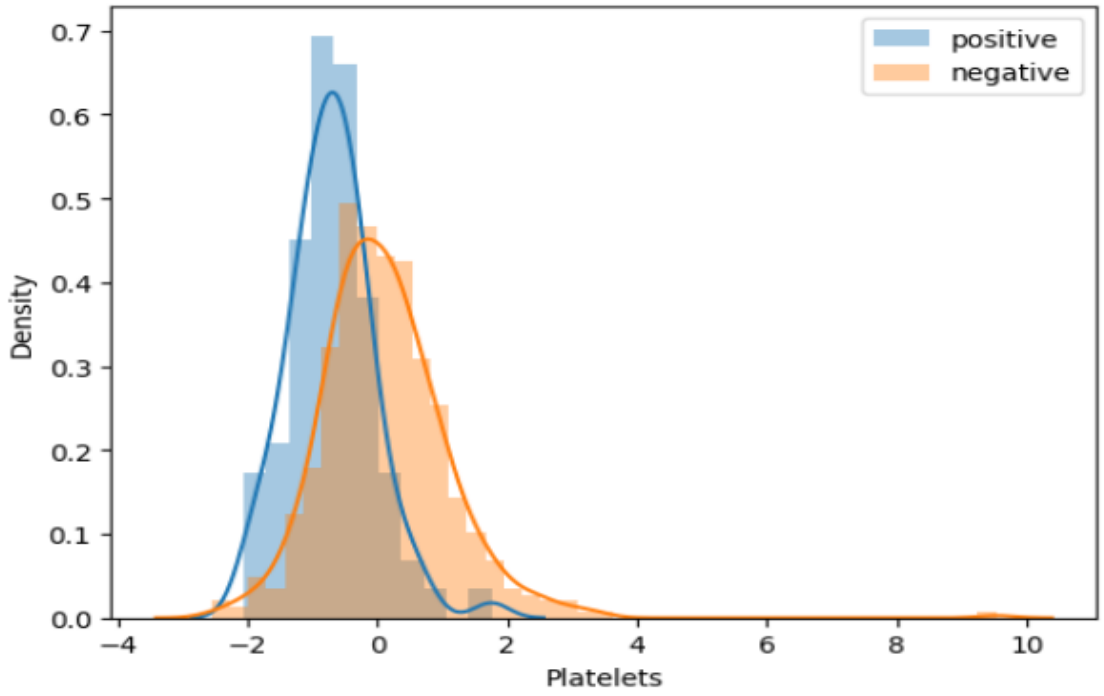
Our initial step should involve partitioning our dataset into positive and negative categories, while also segregating the blood and viral columns.
and then plotting the graphs:

```
for col in blood_columns:
    plt.figure()
    sns.distplot(df_positive[col],label="positive")
    sns.distplot(df_negative[col],label="negative")
    plt.legend()
```




positive
negative
Density
0.0
0.2
0.4
0.6
0.8
−2
0
2
4
Leukocytes


positive
negative
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
−2
0
2
4
Monocytes

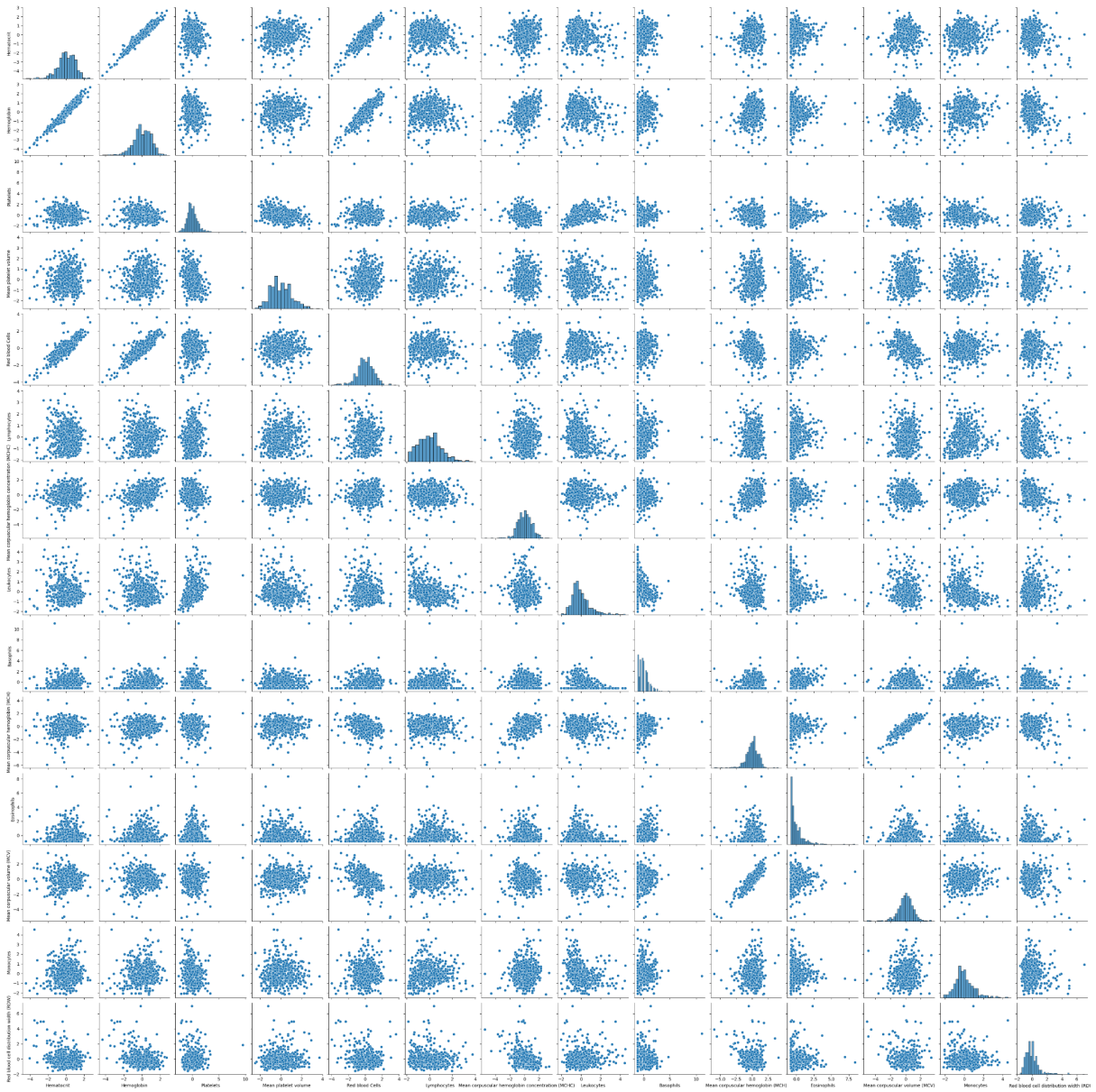


we notice that the distribution of positive class of leuKocytes ,Palettes,and Monocytes features are quite different from the negative one ,so we can suppose that are related to Covid -19:

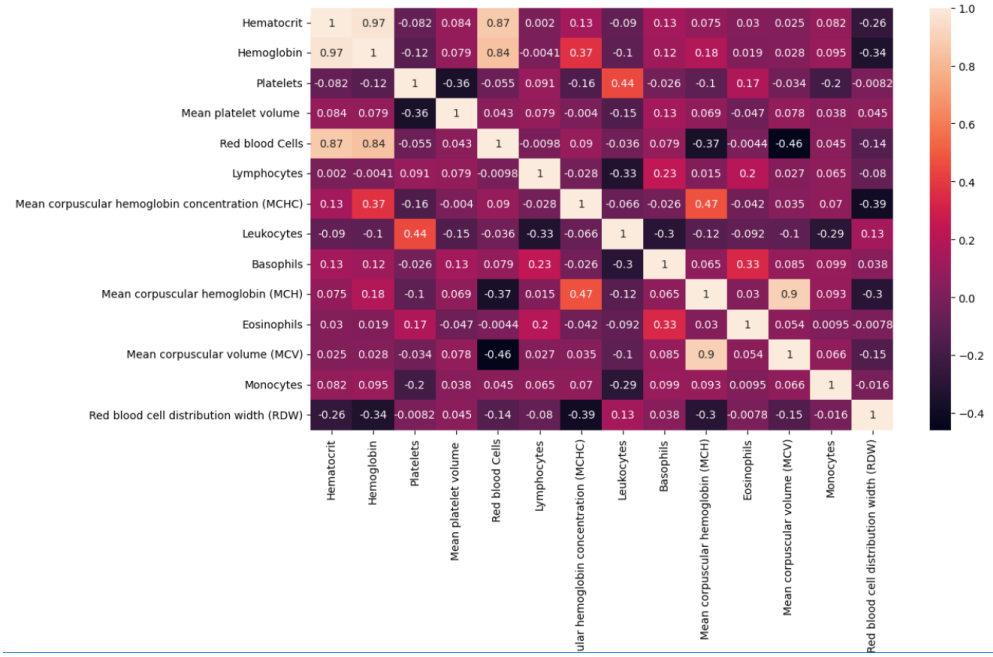
4. exploring relation variable-variable:

analyzing the relation between variables are extremely important

```
### blood/blood
sns.pairplot(df[blood_columns])
```



```
### correlation
plt.figure(figsize=(12,7))
sns.heatmap(df[blood_columns].corr(),annot=True)
```



3. Data preprocessing and modeling:

we have tried a bunch of ideas:

* **doing basic preprocessing** (simple encoding , dropping nan values) and evaluation model:
    * in test data with 3 positive cases our model detect 2 among 3 with a recall of 67%
    * from the learning curve we need more data to make the model more generalized (and avoid overfitting)

* **filling nan values** :fill the null values with a the mean strategy using SimpleImputer :
    * recall 0.05 among 111 positive cases it detects just 6

* **using feature_importance attribute of the model**: viral columns don't have importances
    * feature engineering ==> creating a new column "est malade" that gather all
    * recall of 44% among 16 cases our model detect just 7
* **feature selection**:
    * feature selection using Anova-value
* **adding columns with PolynomialFeatures of sklearn** :it gives an recall with 38%

```
### Encodage
def encodage(df):
    code={
        'positive':1,
        'negative':0,
        'detected':1,
        'not_detected':0
    }
    for col in df.select_dtypes("object").columns:
        df.loc[:,col]=df[col].map(code)

    return df
```

```
def feature_engineering(df):
    df['est malade']=df[viral_columns].sum(axis=1)>=1
    df=df.drop(viral_columns,axis=1)
    return df
```

```
def imputation(df):
    df=df.dropna(axis=0)
    return df
```

```
def preprocessing(df):
    df=encodage(df)
    df=feature_engineering(df)
    df=imputation(df)
    X=df.drop("SARS-Cov-2 exam result",axis=1)
    y=df["SARS-Cov-2 exam result"]
    print(y.value_counts())

    return X,y
```

modeling:

we have tried multiple algorithms like **DecisionTreeClassifier,RandomForestClassifier,AdaBoostClassifier and other algorithms**

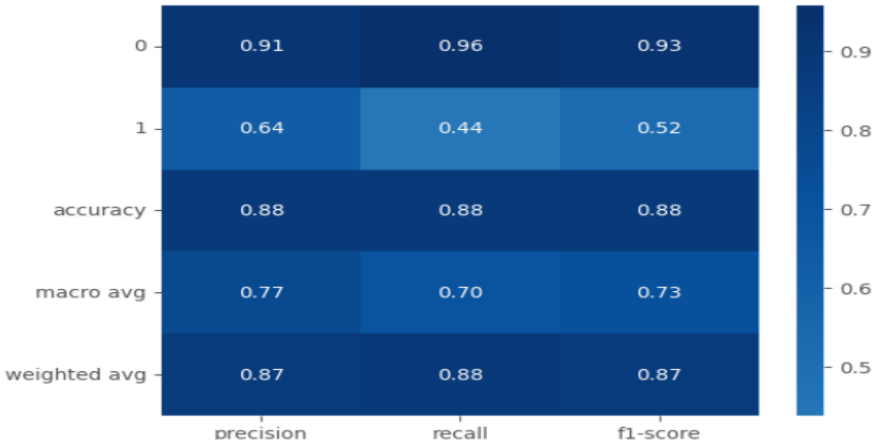
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import SelectKBest,f_classif
from sklearn.preprocessing import PolynomialFeatures
from sklearn.preprocessing import StandardScaler
```

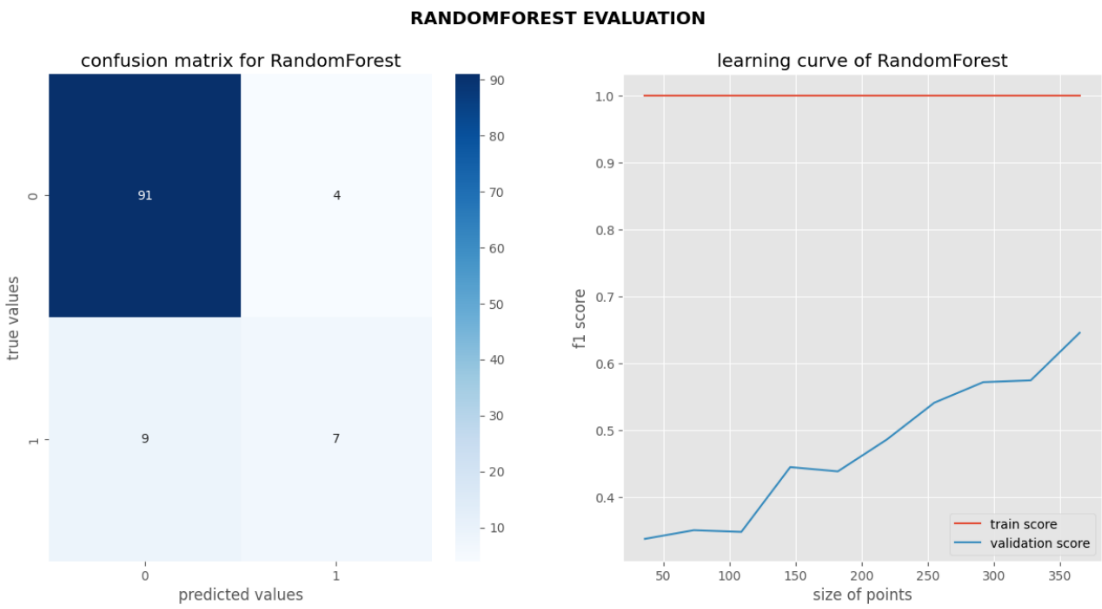
```
RandomForest = Pipeline([
    ('kbest', SelectKBest(k=5)),
    ('rf', RandomForestClassifier())
])
AdaBoost=Pipeline([
    ('kbest', SelectKBest(k=5)),
    ('adaboost', AdaBoostClassifier())
])
SVM = Pipeline([
    ('kbest', SelectKBest(k=5)),
    ('scaler', StandardScaler()),
    ('svm', SVC())
])
KNN = Pipeline([
    ('kbest', SelectKBest(k=5)),
    ('scaler', StandardScaler()),
    ('knn', KNeighborsClassifier())
])
```

Model Evaluation:

```
def evaluation(model,name):
    use("ggplot")
    model.fit(X_train,y_train)
    ypred=model.predict(X_test)
    print(termcolor.colored(pyfiglet.figlet_format(name),color="yellow"))
    report = classification_report(y_test, ypred, output_dict=True)
    sns.heatmap(pd.DataFrame(report).iloc[:-1, :].T, annot=True, cmap="Blues", fmt='.2f', center=0)
    plt.show()
    f, (ax1, ax2) = plt.subplots(1, 2,figsize=(15,7))
    ax = sns.heatmap(confusion_matrix(y_test, ypred) , annot=True , cmap = "Blues" , ax=ax1)
    ax.set_xlabel("predicted values")
    ax.set_ylabel("true values")
    ax.set_title("confusion matrix for " + name)

    N, train_score, val_score= learning_curve(model,X_train,y_train,
                                              cv=4, scoring='f1',
                                              train_sizes=np.linspace(0.1,1,10))

    ax2.plot(N,train_score.mean(axis=1), label="train score")
    ax2.plot(N,val_score.mean(axis=1), label="validation score")
    ax2.set_title(f"learning curve of {name}")
    ax2.set_ylabel(f"f1 score")
    ax2.set_xlabel(f"size of points")
    ax2.legend()
    plt.suptitle(f'{name} Evaluation'.upper(),fontsize=14, fontweight='bold')
    plt.show()
```

```python
for name,model in list_of_models.items():
    # print(name)
    evaluation(model,name)
```





Optimisation SVM with GridSearchCV

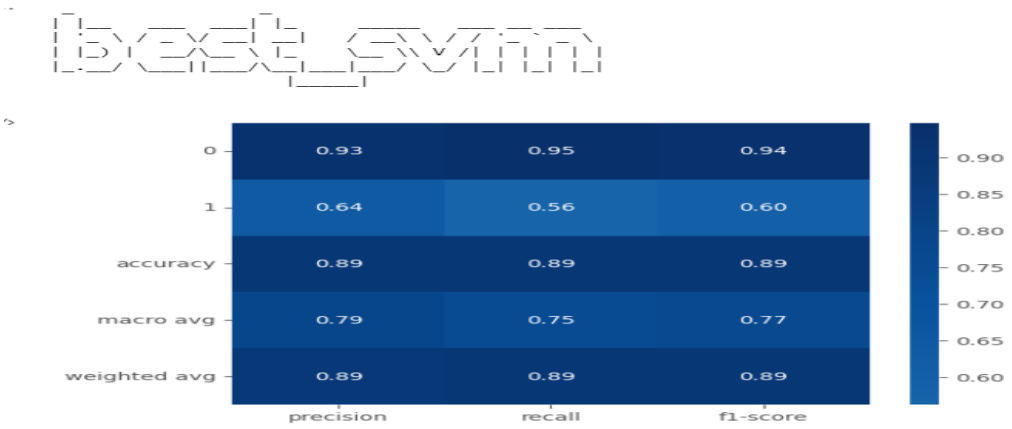
on va optimiser le modèle SVM en utilisant GridSearchCV

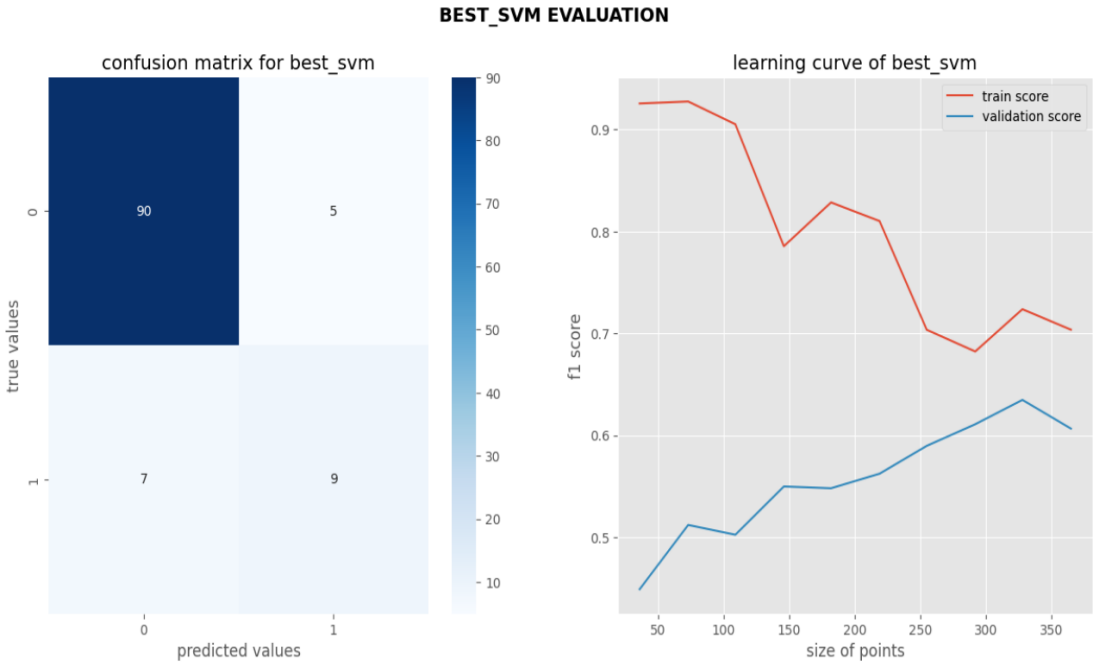
```
from sklearn.model_selection import GridSearchCV
```

```
params = {
    'svm__C': [0.1, 1, 10], # regularization parameter
    'svm__gamma': [0.1, 1, 10] # kernel coefficient
}

# Define GridSearchCV object
grid = GridSearchCV(SVM, param_grid=params, cv=5, scoring='recall')
```

```
grid.fit(X_train,y_train)
print("Best parameters:", grid.best_params_)
print("Best score:", grid.best_score_)

y_pred=grid.predict(X_test)
print(classification_report(y_test,y_pred))
```

```
Best parameters: {'svm__C': 10, 'svm__gamma': 0.1}
Best score: 0.6461538461538462
              precision    recall  f1-score   support

           0       0.93      0.95      0.94        95
           1       0.64      0.56      0.60        16

    accuracy                           0.89       111
   macro avg       0.79      0.75      0.77       111
weighted avg       0.89      0.89      0.89       111
```





finally we saved the model using joblib

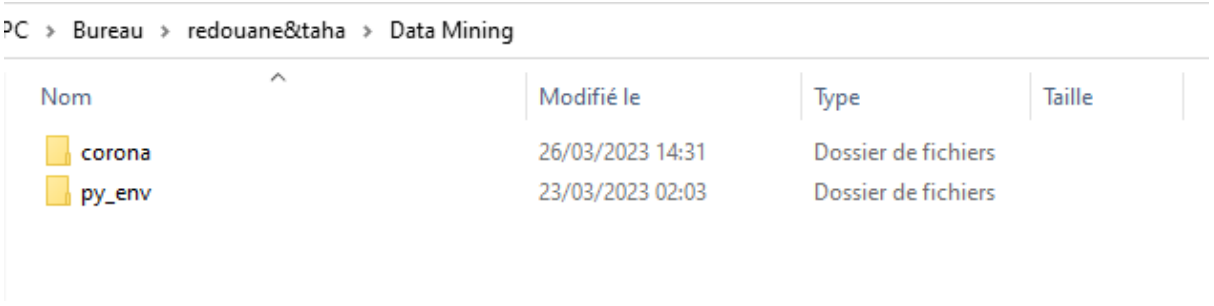
```
from  joblib import dump,load
dump(model,"model")
```

```
model=load("model")
```

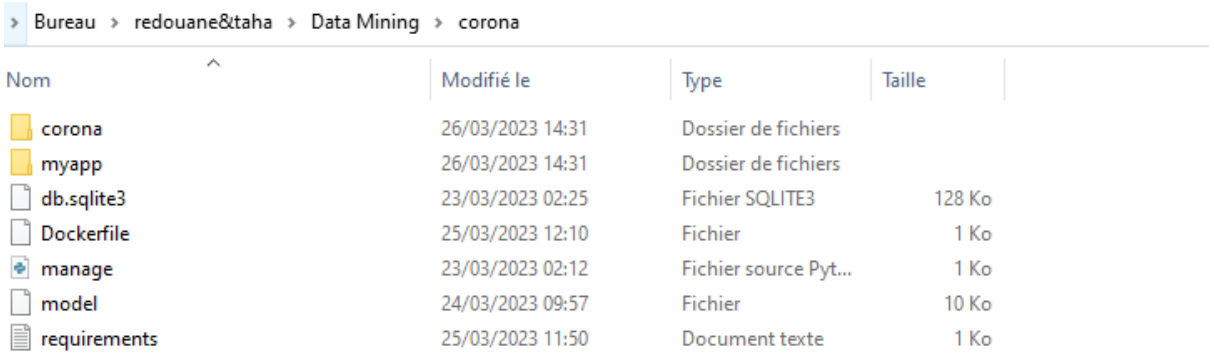
Covid Analysis and prediction

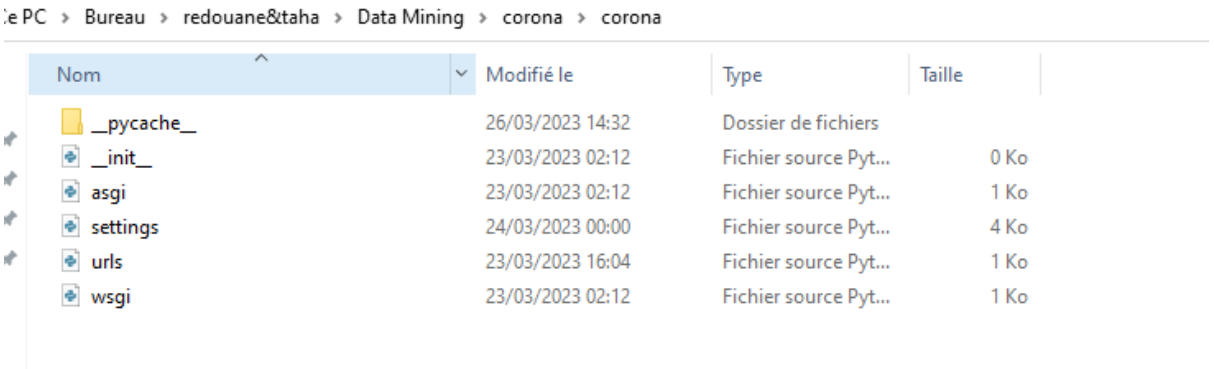
This is our project Django :

i create also a virtual environment called "py_env" :



and then we create our app "my_app":





dict2.html × urls.py ×

sers > user > Desktop > redouane&taha > Data Mining > corona > corona > urls.py > ...

```python
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('',include('myapp.urls'))
]
```

predict2.html × settings.py ×

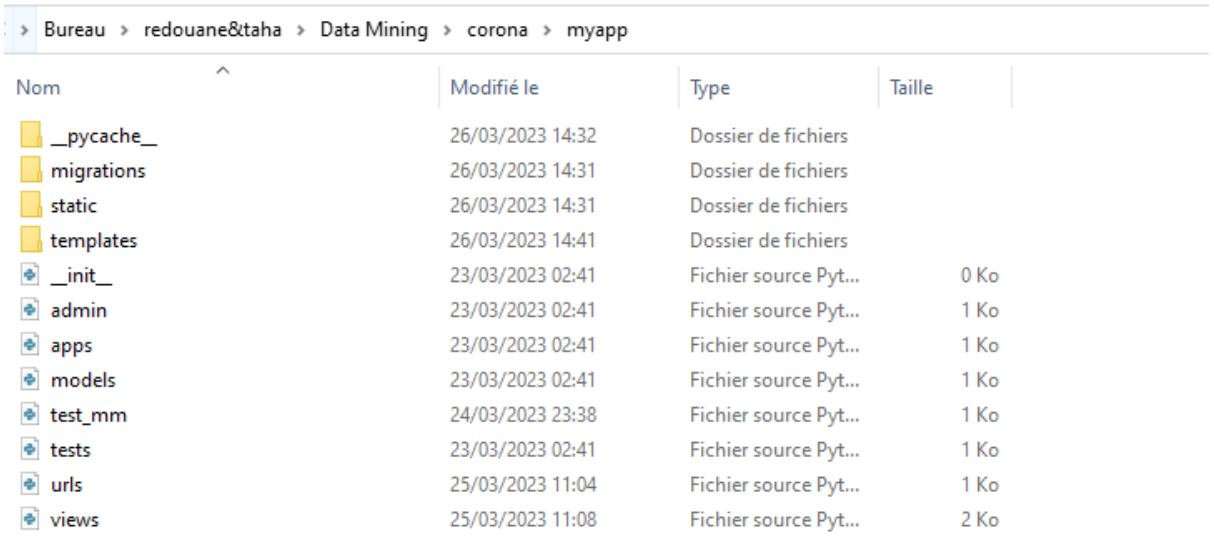
Users > user > Desktop > redouane&taha > Data Mining > corona > corona > settings.py > ...

```python
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,"myapp/templates")],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

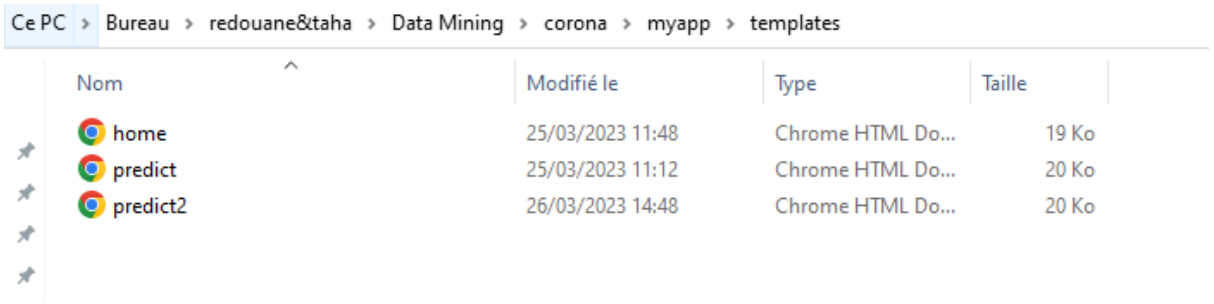
dict2.html settings.py ×

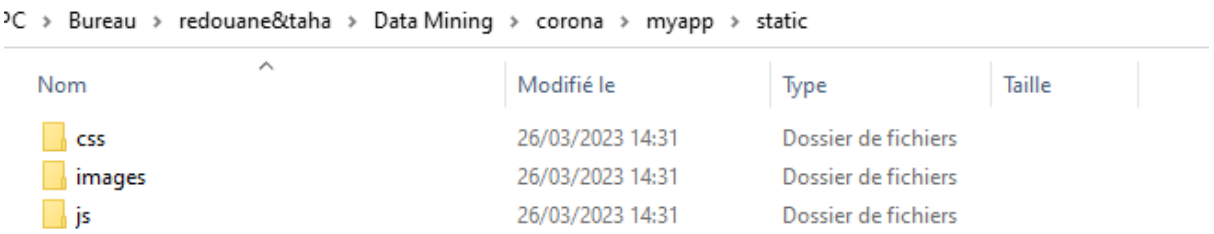
sers > user > Desktop > redouane&taha > Data Mining > corona > corona > settings.py > ...

```python
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/4.1/howto/static-files/

STATIC_URL = 'static/'
STATICFILES_DIRS=[os.path.join(BASE_DIR,"myapp/static")]
STATIC_ROOT=os.path.join(BASE_DIR,"static")

# Default primary key field type
# https://docs.djangoproject.com/en/4.1/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

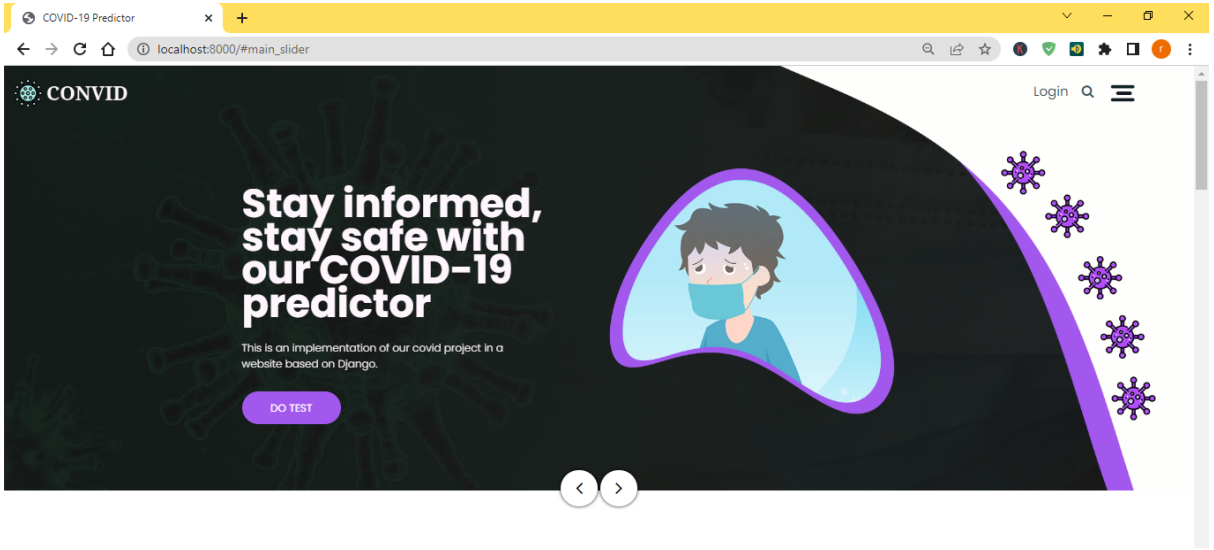


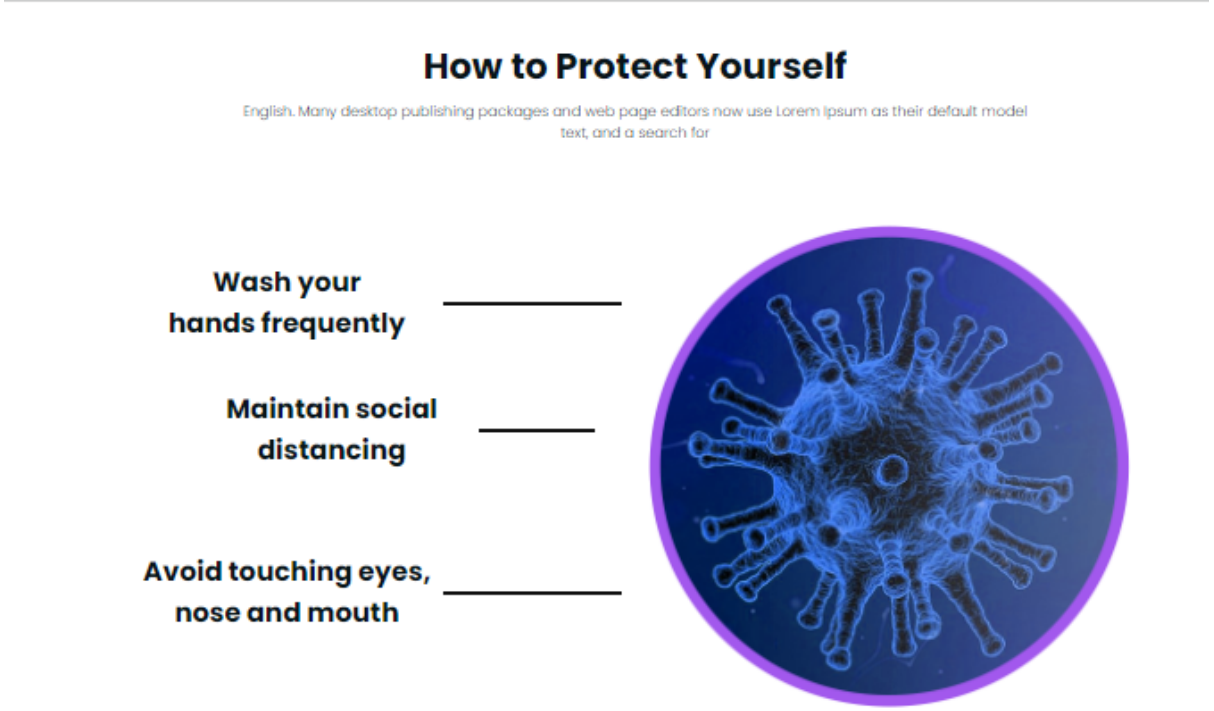
and here we have defined our templates:



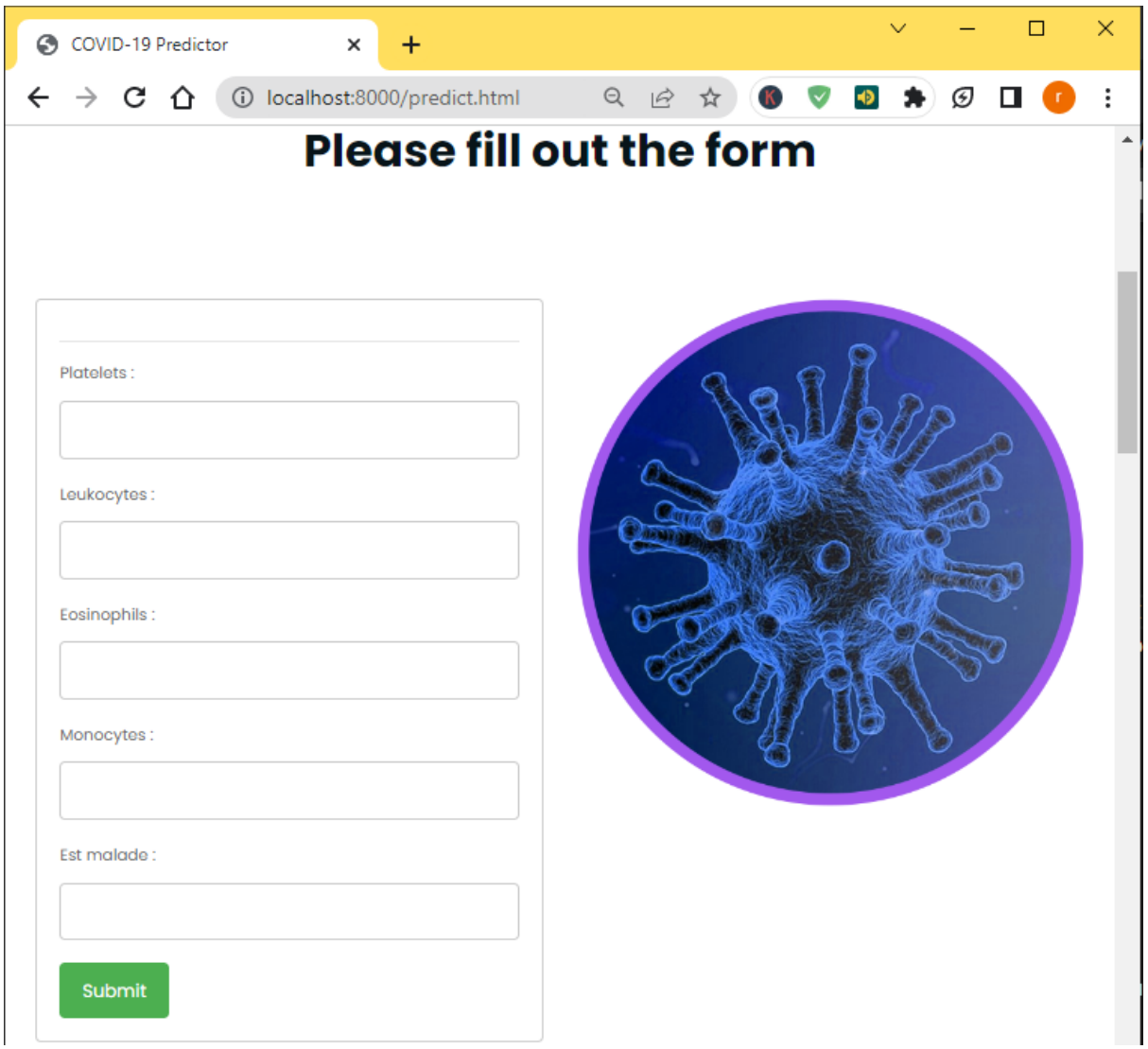



```
C:\Users\user\Desktop\redouane&taha\Data Mining>py_env\Scripts\activate

(py_env) C:\Users\user\Desktop\redouane&taha\Data Mining>cd corona

(py_env) C:\Users\user\Desktop\redouane&taha\Data Mining\corona>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
March 26, 2023 - 14:35:25
Django version 4.1.7, using settings 'corona.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```


COVID-19 Predictor
localhost:8000/#main_slider
CONVID
Login
Stay informed, stay safe with our COVID-19 predictor
This is an implementation of our covid project in a website based on Django.
DO TEST


How to Protect Yourself
English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for
Wash your hands frequently
Maintain social distancing
Avoid touching eyes, nose and mouth


FEVER
HEADACHE
MUSCLE PAIN
COUGH
DYSPNOEA
Coronavirus what it is?
Coronaviruses are a family of viruses that can cause respiratory illness in humans. They are called "corona" because of crown-like spikes on the surface of the virus. Severe acute respiratory syndrome (SARS), Middle East respiratory syndrome (MERS) and the common cold are examples of coronaviruses that cause illness in humans.
READ MORE


COVID-19 Predictor
localhost:8000/predict.html
Please fill out the form
Platelets :
Leukocytes :
Eosinophils :
Monocytes :
Est malade :
Submit

Latest News

Morocco's National Airports Office (ONDA) announced on Tuesday that Moroccan airports welcomed a total of 1,981,294 passengers in January 2023, recording a 6% increase compared to the same period in 2019. The COVID-19 crisis has heavily impacted global air traffic, including that of Morocco, a country whose economy heavily relies on the tourism sector.

Coronavirus is Very dangerous

Although scientists won't know for sure until testing becomes widespread, COVID-19 could be about 10 times more deadly than the seasonal flu, which leads to death in about 0.1% of those it infects, says Donald N. Forthal, MD, professor of medicine and molecular biology and biochemistry, and chief of infectious diseases at UCI School of Medicine.

Read More

DATA MININNG PROJECT

Feedback

SUBSCRIBE NOW

RESOURCES

What we do
Media
Travel Advice
Protection
Care

ABOUT

Our website is dedicated to providing accurate predictions and updates on the COVID-19 pandemic. With the ongoing global health crisis, it is essential to have access to reliable information to make informed decisions. We are committed to promoting public health and safety, and our goal is to contribute to the global efforts to overcome the COVID-19 pandemic.

CONTACT US

RABAT
Call +212 *********
redouanemh@gmail.com
taha@gmail.com

Platelets :

2

Leukocytes :

33

Eosinophils :

12

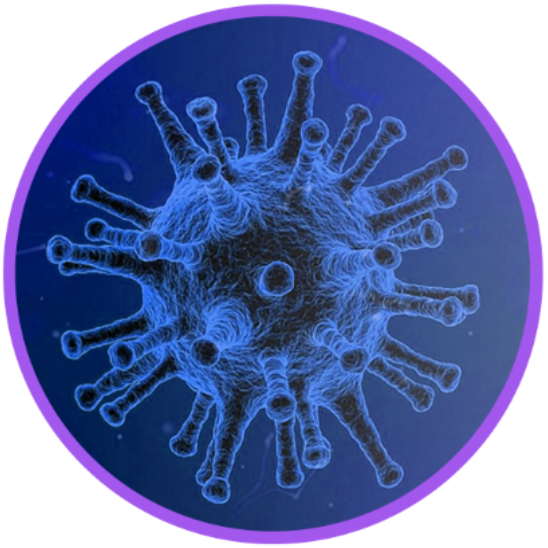
Monocytes :

15

Est malade :

1

Submit



Platelets :

Leukocytes :

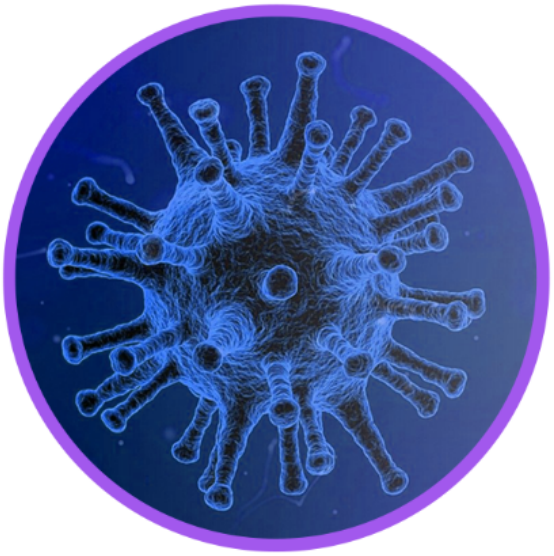
Eosinophils :

Monocytes :

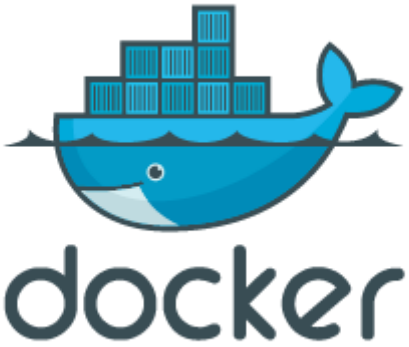
Est malade :

Submit

your test is negative



Dockerizing the app :



first we create a Dockerfile to build the docker image:

Dockerfile > ...

```
1   FROM python:3.9-slim-buster
2
3
4   COPY . /app
5
6   WORKDIR /app
7
8   RUN pip install -r requirements.txt
9
10  EXPOSE 8000
11
12  CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]
```

then we use the command:

```
docker build -t my-django-image .
```

then we run the container:

```
docker run -p 8000:8000 my-django-image
```