**User Manual for BoPD screening tool**

**Oct 22, 2020**

Contents

[1. Introduction and technical support 2](#_Toc52366991)

[2. Preparing input dataset for the screening tool 3](#_Toc52366992)

[2.1 Overall process for preparing input dataset for the screening tool 3](#_Toc52366993)

[2.2 Step-by-step illustration 4](#_Toc52366994)

[ INITIAL INPUT: datasets from EHR database 4](#_Toc52366995)

[ Step 1: Initial inclusion and exclusion 4](#_Toc52366996)

[ Step 2: Data cleaning 6](#_Toc52366997)

[ Step 3: Filter subjects with sufficient history 6](#_Toc52366998)

[ OUTPUT: input dataset for the screening tool 7](#_Toc52366999)

[3. Execution of the screening tool 7](#_Toc52367000)

[3.1 Screening tool overview 8](#_Toc52367001)

[3.2 Step-by-step implementation guide 8](#_Toc52367002)

[3.3 Debugging 13](#_Toc52367013)

[4. ACRONYMS 14](#_Toc52367014)

[5. Appendices 15](#_Toc52367015)

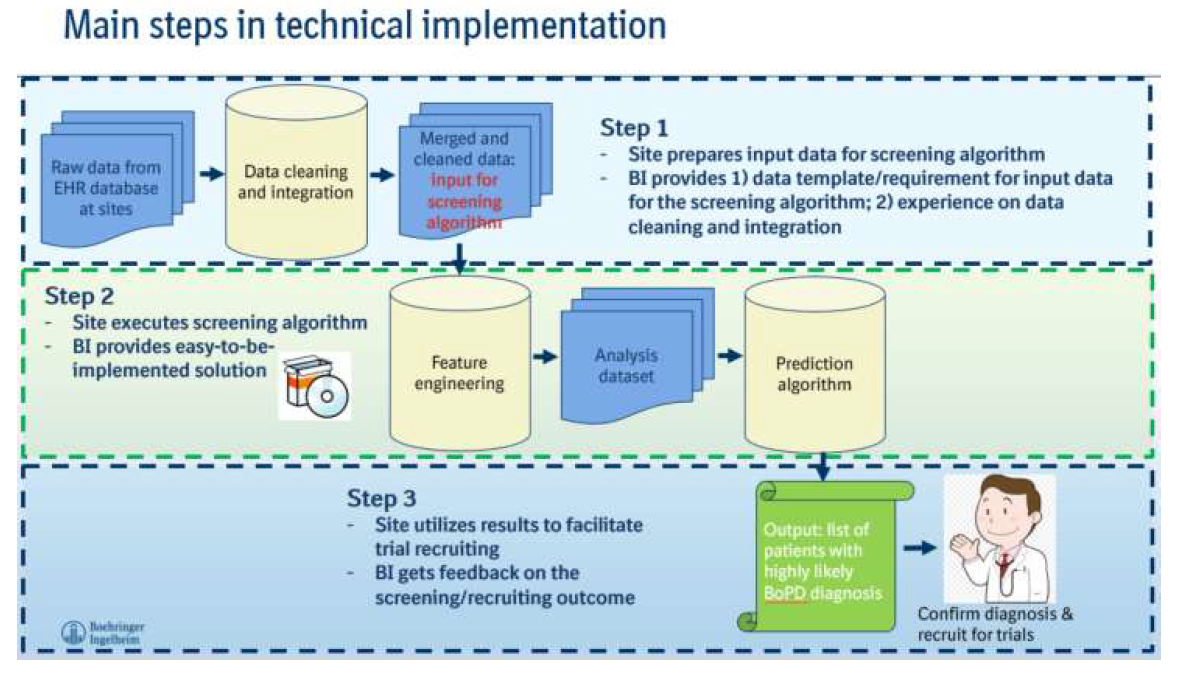
# Introduction and technical support

The objective of the borderline personality disorder (BoPD) Automatic Screening Initiative is to advance timely identification of potential BoPD patients. This was made possible by the development of a machine learning algorithm. The algorithm screens electronic health record (EHR) data and automatically identifies patients with highly likely BoPD diagnosis for confirmation of clinical diagnosis.

In the pilot phase, the algorithm has been incorporated into a portable screening tool based on WinPython, a portable distribution of Python programming language for Windows.

This document includes detailed instruction for the technical implementation of the BoPD screening tool in Step 1 and Step 2 as illustrated below: 1) preparing input dataset for the screening tool, and 2) execution of the screening tool.

Specifically, Step 1 involves EHR database extraction, cleaning, and integration, and is to be executed at site EHR database platform; Step 2 involves execution of the screening tool at site local environment where the dataset prepared from Step 1 is located.



If encountering any difficulty during the implementation of the screening tool, please send email to: [zzMEDBDSBoPDscreeningsupport@boehringer-ingelheim.com](mailto:zzMEDBDSBoPDscreeningsupport@boehringer-ingelheim.com), and your question will be addressed during regular business hour 9am EST to 5pm EST, Monday through Friday excluding holidays, with potential follow up using MS Teams or Skype calls platforms to provide hands-on support.

# Preparing input dataset for the screening tool

This is to be executed at site EHR database platform.

## 2.1 Overall process for preparing input dataset for the screening tool

In this section, the key steps are demonstrated in the flow chart below. Some key information is included in the box on the right side of flow chart.

In next section, more details are provided with sample SQL codes. The SQL codes are for demonstration purpose and please adjust them based on your database. The appendices include additional information needed for preparing the input dataset for the screening tool.

Necessary fields to include: Unique subject identifier, diagnosis codes, visit dates, visit type, demographic information

*Please note that the inclusion and exclusion are for the algorithm, not the inclusion and exclusion criteria for the trial*

***Inclusion****: 1). subjects who ever had an ICD-10 diagnosis code in our inclusion list (provided in* [*appendix 1*](https://boehringer.sharepoint.com/sites/z365bpdexploration/Shared%20Documents/Disease%20screening%20model%20development/Appendix_1.xlsx)*) with the discharged date* ***starting from Jan 01, 2017****; 2). Age 18 - 65*

***Exclusion****: Subjects who ever had an ICD-10 diagnosis code in exclusion list: F60.3, F00-F09, F70-F79*

*Data issue examples (details of cleaning rule in section 2)*

1. *Admitted date>discharged date*
2. *Different subjects corresponding to the same encounter\_id*
3. *Different genders were recorded in different visits*

*>= 5 encounters/visits on different discharged dates*

*or*

*>= 2 emergency visits on different discharged dates*

*Data specification and sample data are provided in Appendix\_2*

## 2.2 Step-by-step illustration

### INITIAL INPUT: datasets from EHR database

This process starts with integrating multiple datasets in EHR database. Usually, EHR data is stored as a relational database. Different information are stored in different datasets. For example, demographic information in one dataset, diagnosis information in another dataset, etc. Hence, those pieces of useful information need to be pulled from different tables.

For this screening tool, the essential information includes demographic (age, gender), visit/encounter date (start date and end date), diagnosis codes in each encounter/visit and encounter type. Therefore, the initial step is to locate the necessary information for later use.

### Step 1: Initial inclusion and exclusion

#### Step 1.1: Get the patient list

Once datasets with all needed information are identified, these fields can be pulled together and the initial inclusion and exclusion criteria can be applied.

Since EHR database has a large population, the first filter of inclusion needs to be applied to select the subjects who ever had the diagnosis codes related to the study since 2017-01-01. The inclusion list is provided in **appendix 1**. Based on our experience, approximately 10% to 15% of the total population may be retained from this step. However, this percentage might differ in different EHR databases.

In addition, all encounters between age 18 to 65 are included in the dataset.

From the subjects who met the inclusion rule, those who ever had an ICD-10-CM diagnosis code of F60.3, F00-F09, F70-F79 need to be excluded. (F60.3: Borderline personality disorder; F00-F09: Mental disorder due to known physiological condition; F70-F79: Intellectual disabilities)

#### Programming Note

1. In appendix\_1, the ICD-10 codes are in a format of combination of letters and numbers without any symbols, i.e. F603. In EHR systems, however, ICD-10 codes usually have ‘.’ between numbers, e.g. ‘F60.3’. So, when using Appendix\_1, please remove any ‘.’ in your dataset to match the diagnosis code in Appendix\_1.
2. Appendix\_1 shall be imported into your database for further data integration and selection. The first column contains all the ICD\_10 codes in the inclusion list. The second column is diagnosis\_description which won’t be used. It is included here for better understanding the meaning of codes.
3. Use SQL queries to merge different tables by INNER JOIN on unique keys to create the tables.
4. Add inclusion conditions in “WHERE” statement of the query. It should contain 18<=age<= 65 and discharged date >= 2017-01-01.
5. Please make sure the discharged\_date is in a date format.

##### Sample code

/\* inclusion step \*/

**create table** inclusion\_patient\_list as

**select distinct** a.patient\_sk

**from** database.table1 as a

**inner join** database.table2 as b

**on a.key1=b.key1** /\* link the tables with unique key \*/

**inner join** database.table3 as c **on b.key2=c.key2**

**where** (translate(a.diagnosis\_code,'.','') in (**inclusion\_list**)

/\* inclusion list is the first column of appendix\_1 \*/

and a.diagnosis\_type = 'ICD10-CM')

and 18<= b.age\_in\_years <=65

and **c.**discharged\_date >=2017-01-01

;

/\* Exclusion step \*/

/\* Create exclusion patient list by putting the exclusion condition in where statement \*/

**create table** exclusion\_patient\_list as

select **distinct** a.patient\_sk

**from** database.table1 as a

**inner join** database.table2 as b

**on a.key1=b.key1** /\* link the tables with unique key \*/

**inner join** database.table3 as c **on b.key2=c.key2**

**where**

(diagnosis\_code = 'F60.3' and diagnosis\_type = 'ICD10-CM')

or

( substr(dd.diagnosis\_code,1,3) in ( 'F00','F01','F02','F03','F04','F05','F06','F07','F08','F09'

,'F70','F71','F72','F73','F74','F75','F76','F77','F78','F79' )

and dd.diagnosis\_type = 'ICD10-CM' );

/\* Use left join and where b.patient\_sk is NULL to exclude the patients in table exclusion\_patient\_list \*/

**create table** patient\_list\_of\_step1 as

select a.patient\_sk

from inclusion\_patient\_list as a

**left join**

exclusion\_patient\_list as b

on a.patient\_sk=b.patient\_sk

**where b.patient\_sk is null**

#### Step 1.2 Gather all needed information

Once the patient list is obtained from step 1.1, the next step is to gather all diagnosis codes **after 2017-01-01**. You can use similar SQL to select all visits/encounters and their non-missing diagnosis codes by using the patient\_sk (unique patient identifier) you get from step 1.1.

**After step 1.2, the output dataset should look exactly like the sample data in Appendix\_2.**

##### Sample code

/\* Gather all diagnosis information \*/

**create table** your\_dataset\_of\_step\_1 as

**select distinct** pl.patient\_sk, a.column1, b.column2

,c.column3, …

from database.patient\_list\_of\_step1 as pl

inner join database.table1 as a on **pl.patient\_sk=a.patient\_sk**

**inner join** database.table2 as b **on a.key1=b.key1**

**inner join** database.table3 as c **on b.key2=c.key2**

**where** **a.diagnosis\_code is NOT NULL**

and a.diagnosis\_type = 'ICD10-CM')

**and**  18<= b.age\_in\_years <=65

**and** discharged\_date >=2017-01-01

### Step 2: Data cleaning

Once all diagnosis codes, visit dates, and demographic information are gathered in one dataset, you can start investigating potential abnormalities. Usually, in the EHR system, the demographic information was entered or updated at each visit. Therefore, inconsistency may exist in gender and year of birth. Another abnormality can happen to the admitted date and discharged date of each visit if wrong dates was entered. And appropriate data cleaning is needed.

Below are possible abnormities to check:

* *Issue 1: Admitted date>discharged date*
* *Issue 2: Different subjects were corresponding to the same encounter\_id*
* *Issue 3: Different birth year was recorded in different visits*
* *Issue 4: Different gender was recorded in different visits*
* *Issue 5: Missing values*

Please follow the data cleaning suggestions as below:

1. For issue 1 and 2, exclude the records by encounter level. It means only remove the specific rows with those issues.
2. For issue 3, calculate the year of birth at each encounter by discharged date and age at each encounter (birth year = year of discharged date – age at that visit), if the max(year of birth)- min(year of birth) >3, then exclude this patient.
3. If there is more than one gender, then use "majority vote" (Normally the gender information is recorded at each visit, and it should be consistent. However, for some instances, different genders are observed over time. Therefore, majority vote should be applied); In addition, if equal vote happens, exclude the patient from the analysis. In addition, the possible values for gender are “Female” and “Male”, other values should be excluded.
4. Records with missing values in any columns should be removed except for “patient\_type\_desc” where the Null/missing is allowed.

### Step 3: Filter subjects with sufficient history

In this step, subjects with sufficient clinical history are retained for the screening tool. These subjects need to have at least five encounters on different discharge dates or at least two emergency visits on different discharge dates.

#### Programming Note

1. For condition of “at least five encounters on different dates”, select the patients from previous steps and use the condition” having count(distinct(discharged\_date))>=5”, create patient-list 1.
2. For condition of “at least two emergency visits”, first identify the visits which are emergency, put them into “emergency group” or set a flag to it. Then select the patients having count(distinct(discharged\_date))>=2 when the visit type is emergency, create patient list 2.
3. There might be some overlap in patient-list 1 and 2. Then get a UNION list of these two.

##### Sample code

/\* create a list of 5+ encounters \*/

/\* patient\_sk is the unique patient identifier \*/

create table pt\_of\_5plus as

select distinct patient\_sk

from your\_dataset\_of\_step2

group by patient\_sk

**having count(distinct(discharged\_date))>=5 ;**

/\* this is the list of 2+ emergency visits \*/

create table pt\_of\_2emer as

select distinct patient\_sk

from your\_dataset\_of\_step2

**where patient\_type\_desc in ('Emergency')**

group by patient\_sk

**having count(distinct(discharged\_date))>=2;**

/\* Union above two lists to get list of patients with sufficient clinical history \*/

create table pt\_list\_of\_step3 as

**select a.\***

from pt\_of\_5plus as a

**union**

**select b.\***

from pt\_of\_2emer as b;

/\* use this patient list to merge back with previous dataset to get other columns \*/

create table your\_dataset\_of\_step3 as

select b.\*

from pt\_list\_of\_step3 as a

inner join your\_dataset\_of\_step2 as b

on a.patient\_sk=b.patient\_sk ;

### OUTPUT: input dataset for the screening tool

Once all the previous steps are completed, you would already have a patient list who meets all those filtering requirements. Please then merge all the required information of these patients into one single dataset and clean up all the abnormities as specified in the previous section.

The data specification and a sample dataset can be found in **Appendix\_2.**

This dataset will be used as the input dataset for the screening tool. Please save it as “.csv” file and place it in the local environment under screening tool folder “BoPDScreeningTool/Application Demo” (refer to step 1 in section 3.2 below).

# Execution of the screening tool

This is to be executed at site local environment where the dataset prepared from the previous section, i.e., input dataset for the screening tool, is located.

## 3.1 Screening tool overview

The screening tool aims at providing a centralized place to process built-in functions and deliver patient screening results. It can be implemented on a regular PC with Windows 8 or higher version, with >=4GB free space on C drive. Additional C drive space is needed for placing the dataset prepared from the previous section and it depends on the size of the dataset. The memory requirement depends on the volume of data and details can be found in the next section.

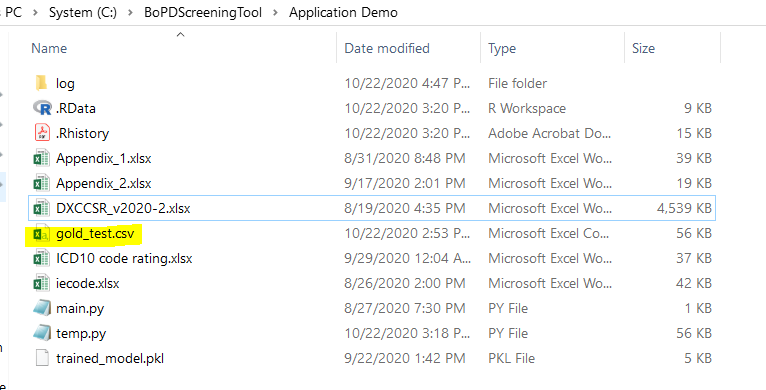
## 3.2 Step-by-step implementation guide

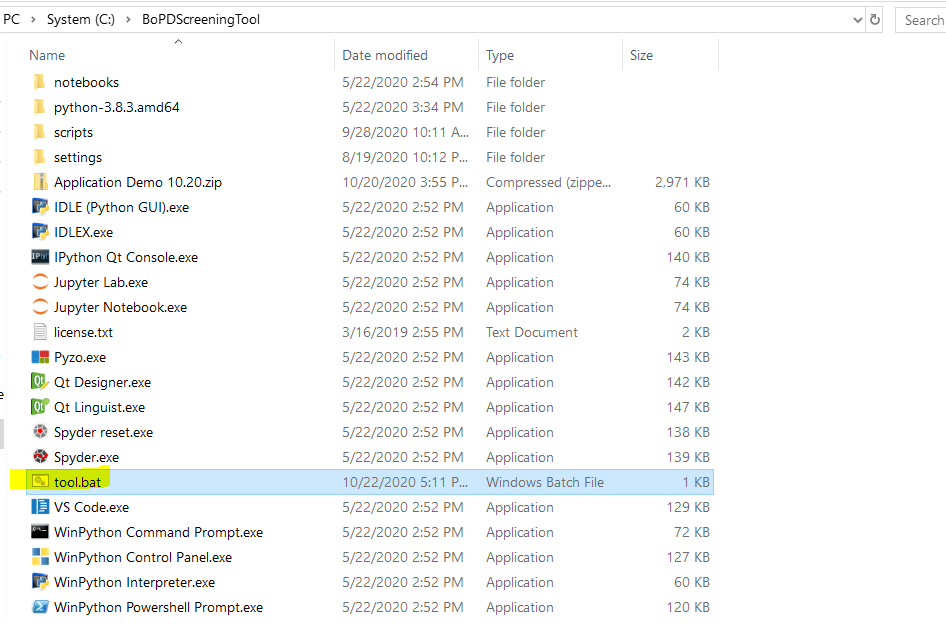
Step 0. Save and unzip BoPDScreeningTool.zip file under **C:/**, otherwise the application can’t be initiated.

*Note: The downloading can take approximate 1 hour to 3 hours depending on the internet speed; to unzip the file, overnight unzipping is recommended.*

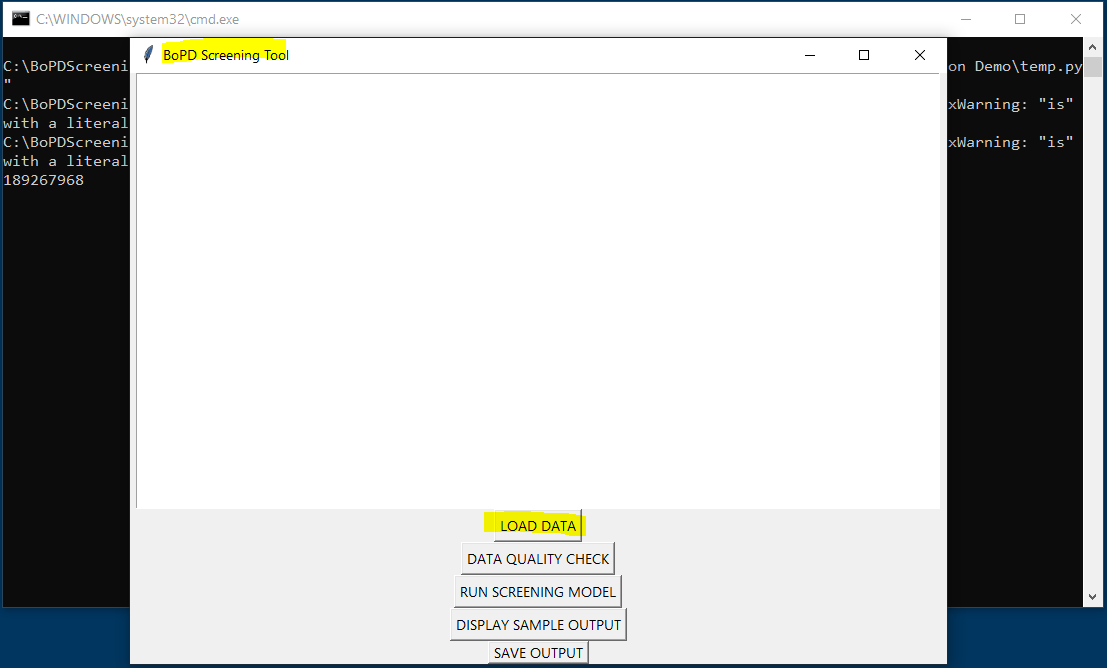
Step 1. Place the input dataset prepared from the previous section under folder “BoPDScreeningTool/Application Demo”, and it should be saved as .csv file.

Here are already preloaded files working as required built-in components for running the screening tool. The input sample dataset, *gold\_test.csv,* is for testing purpose.



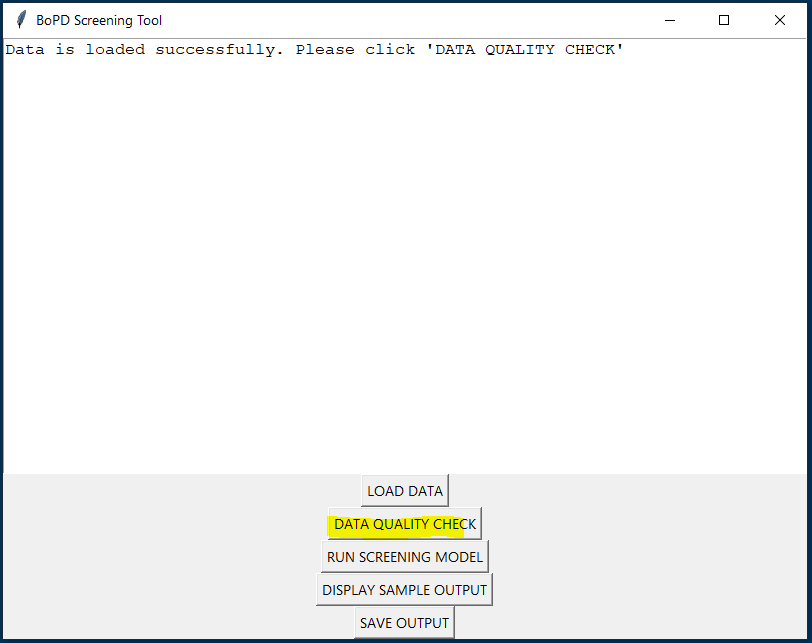
Step 2. Open “tool.bat” (highlighted in yellow below) **.** 

Step 3. A black command prompt window (cmd.exe) will appear first and then a window called ‘tk’ will pop up after a few seconds. Click on ‘LOAD DATA’ button to load the dataset from Step 1. Again, this dataset should be in **.csv** format and it is **placed under folder “BoPDScreeningTool/Application Demo”.**



Step 4. After seeing following notice on the window, click ‘DATA QUALITY CHECK’ button.

Note: If the dataset is not loaded successfully, a reminder will show up on the screen and the program will terminate if ‘DATA QUALITY CHECK’ button is clicked.



Step 5. Notice will appear on the window if data quality check is passed. Then please move on to click “RUN SCREENING MODEL”.

If data quality check is not passed, there will be notice on the window. Please check the log file and make updates on the input dataset. The tool will terminate automatically if clicking on next model running button without passing the data quality check.

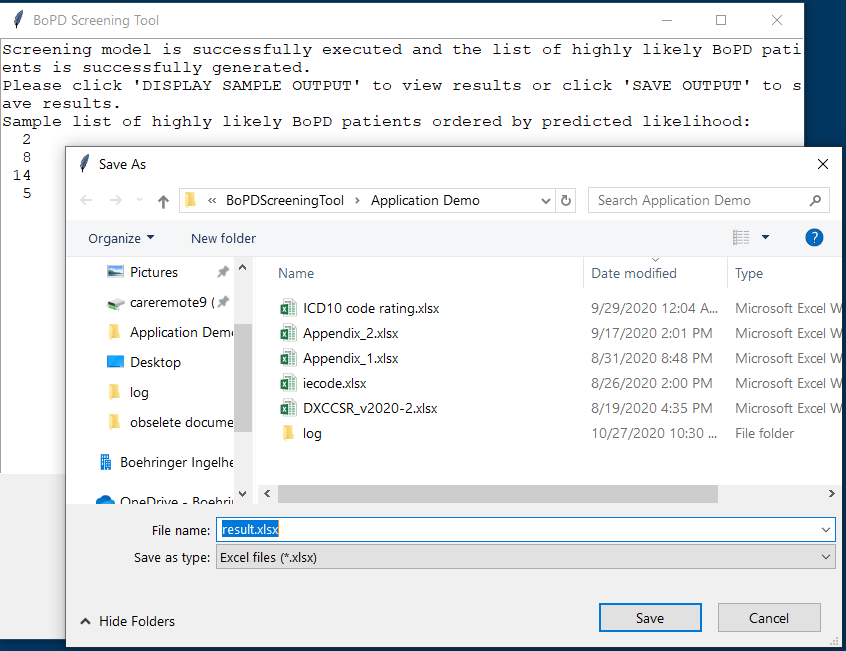
Notice will appear on the window if model runs through successfully. If not, the tool will terminate automatically and please refer to log file (details in separate section) for error shooting.

Run-time estimation (based on computer with Intel Core i5-8350U CPU and 8 GB memory):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # of patients | Total required memory (GB) | Runtime - data loading (minutes) | Runtime - data quality check (minutes) | Runtime - execution of screening model (minute) |
| 200 | <1 | <1 | <1 | <1 |
| 500 | <1 | <1 | <1 | <1 |
| 1000 | <1 | <1 | <1 | <1 |
| 2000 | <1 | <1 | <1 | 2 |
| 5000 | <1 | <1 | <1 | 2 |
| 180,000 | <1 | <1 | 2 | 36 |

Step 6. Click display and save button to review and save the results as **.xlsx** file. The results includes patient\_sk (patient identification number) for those predicted by the algorithm as highly likely BoPD patients.

*Note: please make sure to type “.xlsx” in the File name when saving the results.*

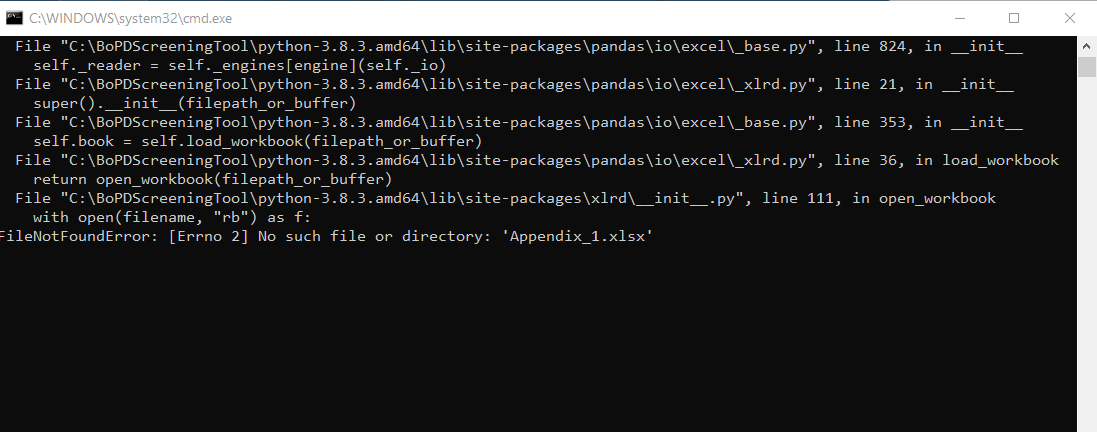


## 3.3 Debugging

If the program stops working, it is highly likely that there is an unmet requirement in the input dataset or missing steps during implementation. Please refer to the self-generated log file under **“log”** folder to check details.

For further help, please send the **log file** together with **messages on the black command prompt window (cmd.exe)** to [zzMEDBDSBoPDscreeningsupport@boehringer-ingelheim.com](mailto:zzMEDBDSBoPDscreeningsupport@boehringer-ingelheim.com)

To save command window interface:



# ACRONYMS

| **Acronym** | **Translation** |
| --- | --- |
| BoPD | Borderline Personality Disorder |

# Appendices

Appendix\_1: Inclusion list



Appendix\_2: Data specification

