SETUP

Installing Python using Anaconda

Python is a popular language for scientific computing, and great for general-purpose programming as well. Python lets you work quickly and integrate systems more effectively. Nowadays many Linux and UNIX distributions include a recent Python. Even some Windows computers (notably those from HP) now come with Python already installed. Before you start check that you don't already have Python installed by entering python in a command line window. If you see a response from a Python interpreter, it will include a version number in its initial display.

Step 1

Mac: Open a Web browser and go to https://www.anaconda.com/download/#macos
Widows:Open a Web browser and go to https://www.anaconda.com/download/#windows

Click 'Download' under Python 2.7 version

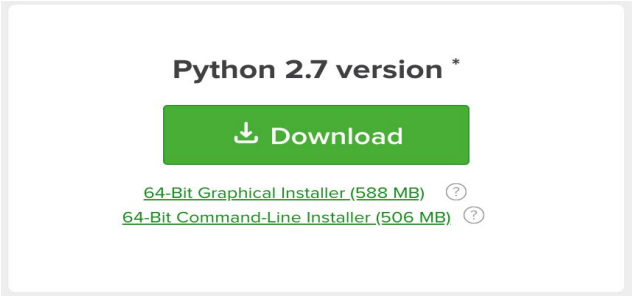

Figure 1: Anaconda website homepage

Step 2
Open Installer

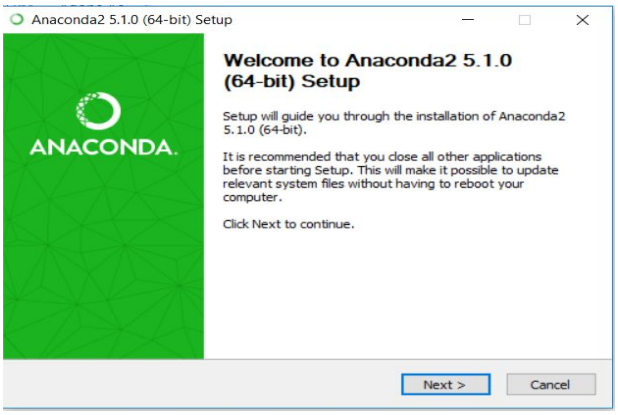

Figure 2: Setup window

Step 3

Read the licensing terms and click “I Agree”

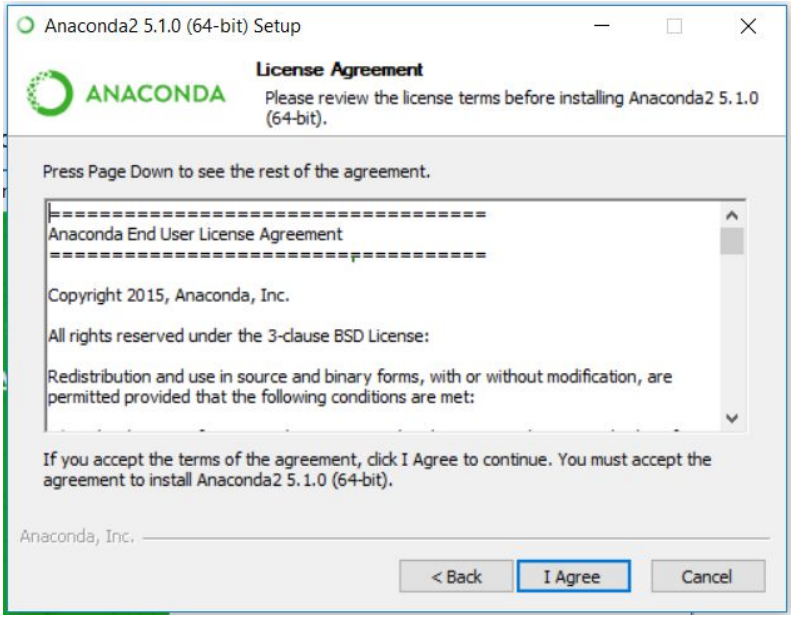

Figure 3: End user License Agreement Window

Step 4

Select “Just Me” unless you’re installing for all users

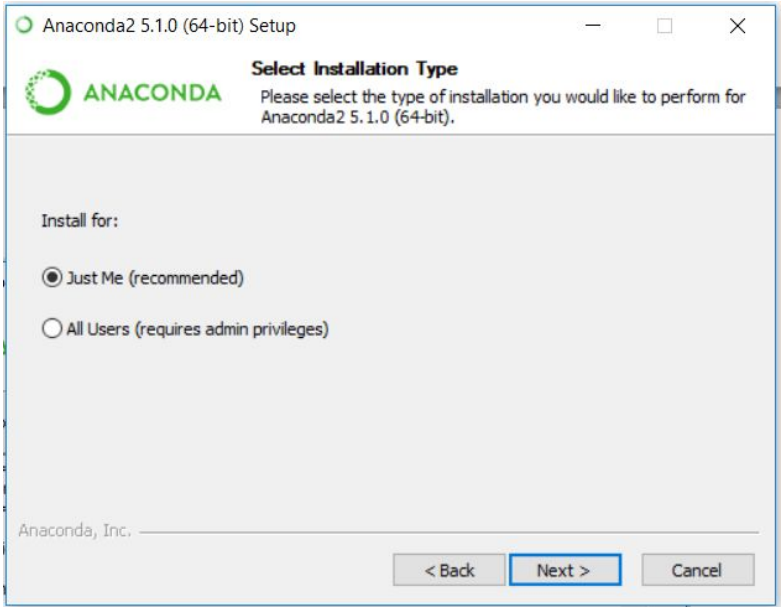

Figure 4: Installation type Window

Step 5

Select a destination folder to install Anaconda and click the Next button

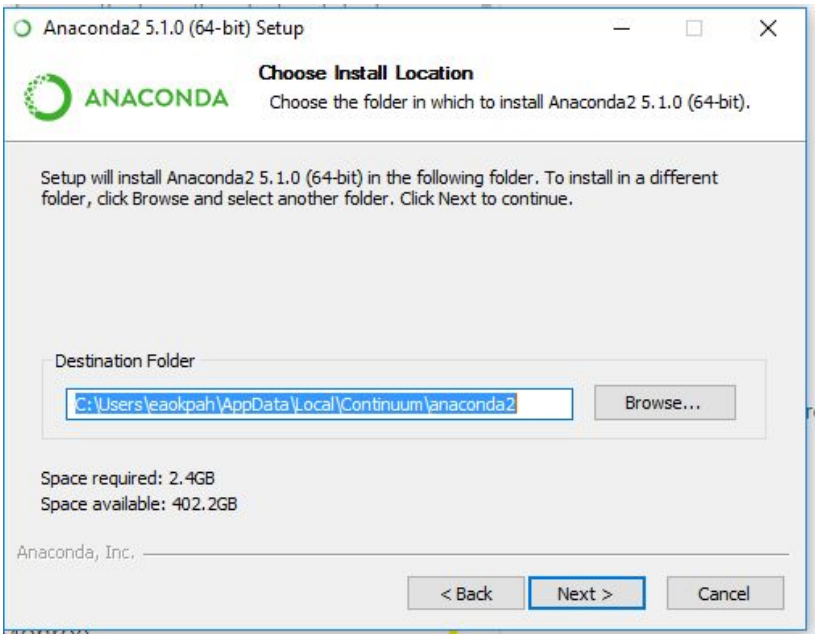

Figure 5: Installation Location

Step 6

Choose whether to add Anaconda to your PATH environment variable

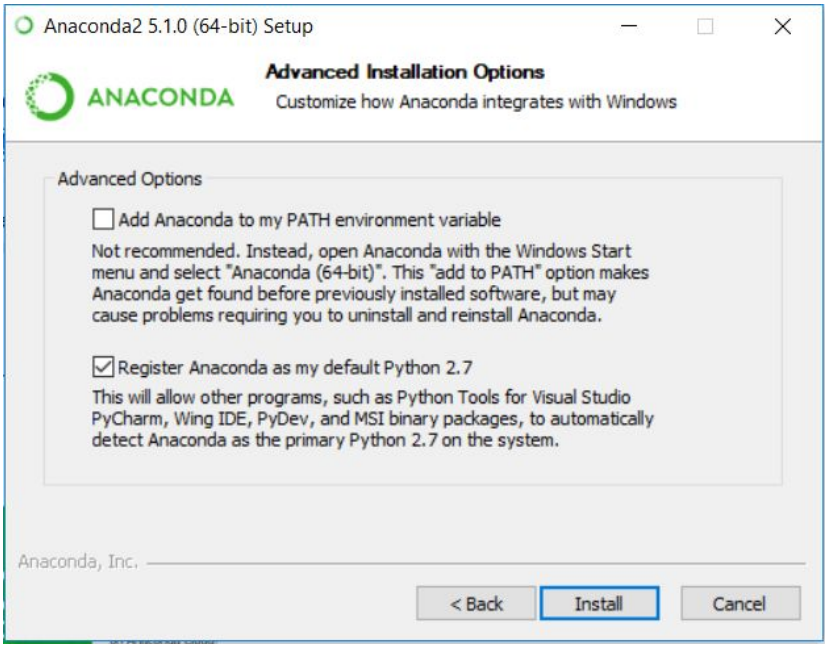

Figure 6: Advance Installation Option

Step 7

Choose whether to register Anaconda as your default then click the Install button and follow the prompt

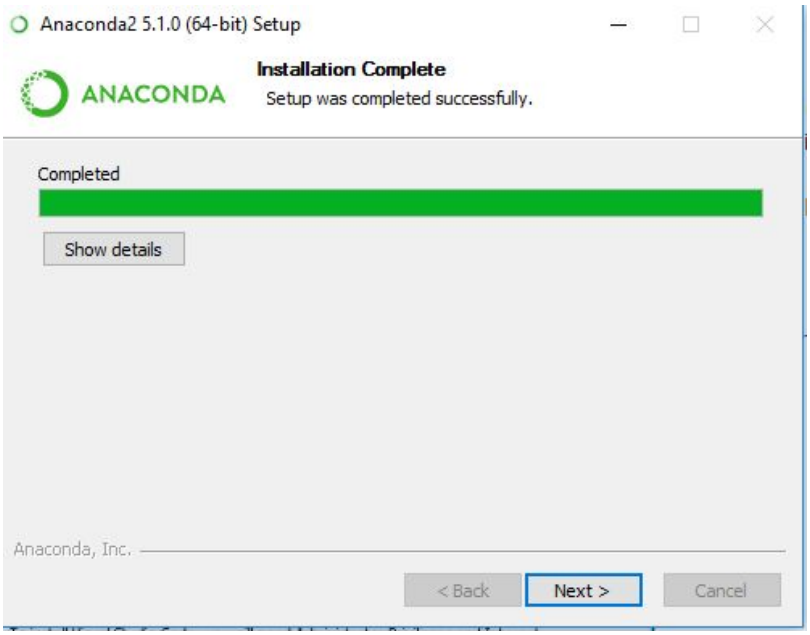

Figure 7a: Setup Complete

Step 8

After a successful installation you will see the "Thanks for installing Anaconda" dialog box:

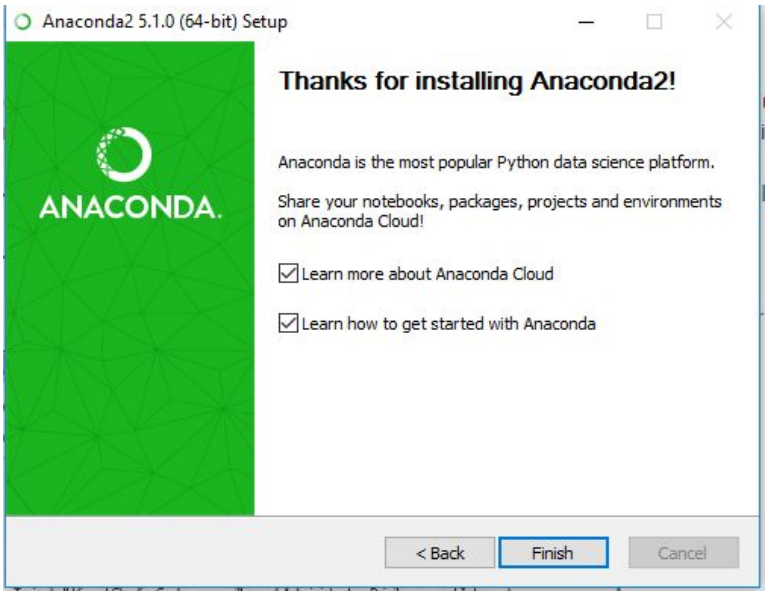

Figure 7b: Setup Complete

Step 9

Verify that Python is properly installed by entering **python** in the Anaconda Prompt.

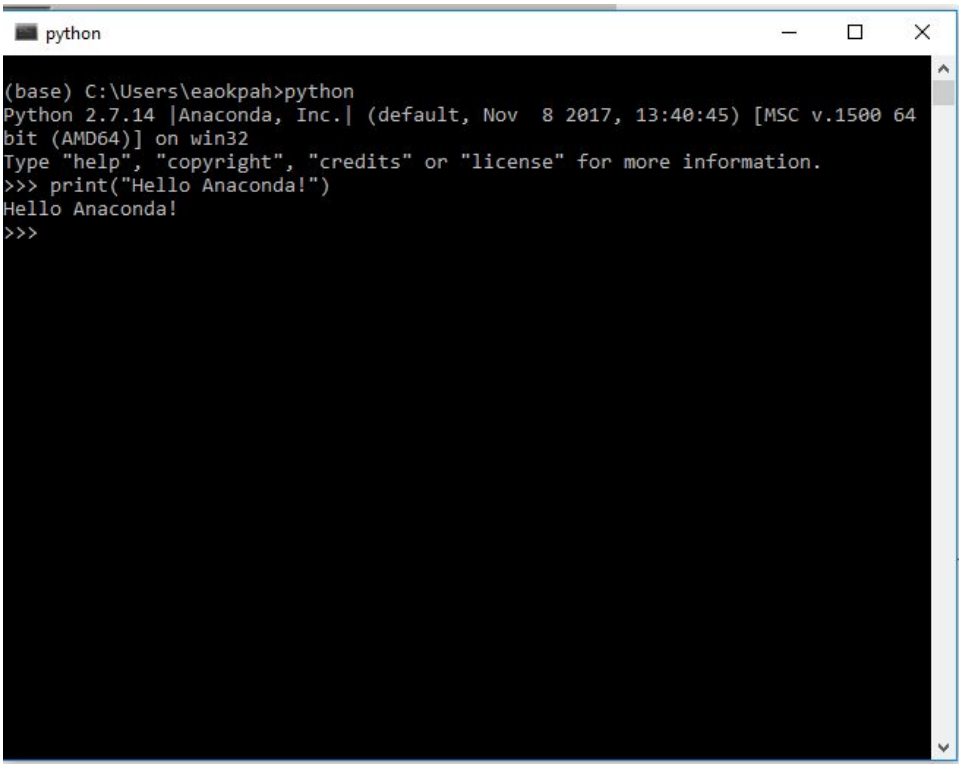

Figure 8 : Anaconda Prompt

Predicting Phenotype from Genotype with Machine Learning

- Download project from github: https://github.com/bioInfoResearch/genopheno
- From the repository, select 'Clone or download'.

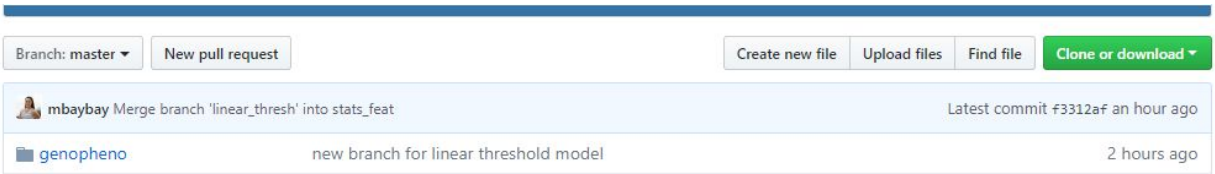

Figure 9a:Screenshot of github Repository

- Copy the clone command (the HTTPS format) We will clone the repository by pasting the HTTP address to use with git clone.

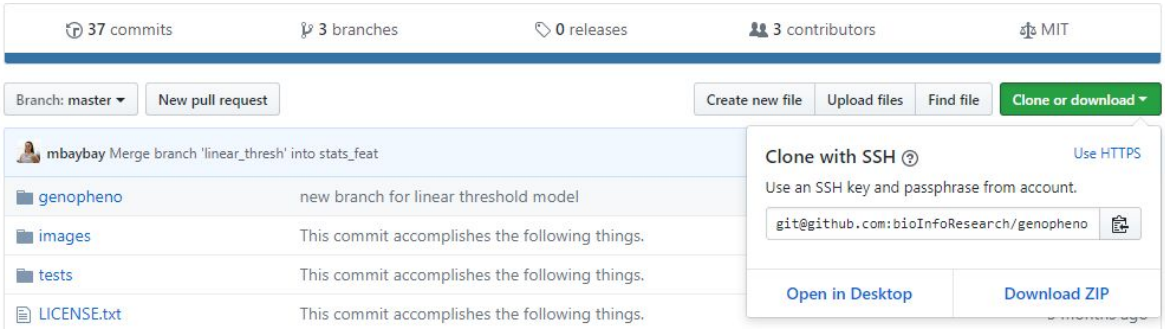

Figure 9b: Screenshot of github repository

- Open your terminal window, change to the local directory where you want to clone the repository. For example Below, we clone the repository into the a folder called pp in our desktop.

```
Amenze@DESKTOP-K4C1R0E MINGW64 ~/desktop
$ git clone https://github.com/bioInfoResearch/genopheno.git
Cloning into 'genopheno'...
remote: Counting objects: 386, done.
remote: Compressing objects: 100% (29/29), done.
Receiving objects:  46% (181/386), 117.93 MiB | 1010.00 KiB/s
```

Figure 9c:Screenshot of command used to Cloning a repository

- A successful cloning will create a new sub-directory that contains the files and metadata that Git requires to maintain the changes you make to the source files.

Running the Application

Each project has its own requirements.txt file, and this install the dependencies for that project into its virtual environment. In the command line type in

pip install -r requirements.txt

```
Anaconda Prompt - pip  install -r requirements.txt
Collecting patsy (from -r requirements.txt (line 8))
  Downloading patsy-0.5.0-py2.py3-none-any.whl (232kB)
     100% |################################| 235kB 2.2MB/s
Collecting graphviz (from -r requirements.txt (line 9))
  Downloading graphviz-0.8.2-py2.py3-none-any.whl
Collecting backports.functools-lru-cache (from matplotlib->-r requirements.txt (line 1))
  Downloading backports.functools_lru_cache-1.5-py2.py3-none-any.whl
Requirement already satisfied: six>=1.10 in c:\users\eaokpah\desktop\genopheno\genopheno_ve\lib\site-packages (from matplotlib->-r re
quirements.txt (line 1))
Requirement already satisfied: pytz in c:\users\eaokpah\desktop\genopheno\genopheno_ve\lib\site-packages (from matplotlib->-r require
ments.txt (line 1))
Collecting cycler>=0.10 (from matplotlib->-r requirements.txt (line 1))
  Downloading cycler-0.10.0-py2.py3-none-any.whl
Collecting kiwisolver>=1.0.1 (from matplotlib->-r requirements.txt (line 1))
  Downloading kiwisolver-1.0.1-cp27-none-win_amd64.whl (64kB)
     100% |################################| 71kB 4.5MB/s
Requirement already satisfied: python-dateutil>=2.1 in c:\users\eaokpah\desktop\genopheno\genopheno_ve\lib\site-packages (from matplo
tlib->-r requirements.txt (line 1))
Collecting pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 (from matplotlib->-r requirements.txt (line 1))
  Downloading pyparsing-2.2.0-py2.py3-none-any.whl (56kB)
     100% |################################| 61kB 3.8MB/s
Collecting py>=1.5.0 (from pytest->-r requirements.txt (line 5))
  Downloading py-1.5.2-py2.py3-none-any.whl (88kB)
     100% |################################| 92kB 3.0MB/s
Collecting attrs>=17.2.0 (from pytest->-r requirements.txt (line 5))
  Downloading attrs-17.4.0-py2.py3-none-any.whl
Collecting pluggy<0.7,>=0.5 (from pytest->-r requirements.txt (line 5))
  Downloading pluggy-0.6.0.tar.gz
Collecting funcsigs; python_version < "3.0" (from pytest->-r requirements.txt (line 5))
  Downloading funcsigs-1.0.2-py2.py3-none-any.whl
Collecting colorama; sys_platform == "win32" (from pytest->-r requirements.txt (line 5))
  Downloading colorama-0.3.9-py2.py3-none-any.whl
Requirement already satisfied: setuptools in c:\users\eaokpah\desktop\genopheno\genopheno_ve\lib\site-packages (from pytest->-r requi
rements.txt (line 5))
Building wheels for collected packages: pydotplus, pluggy
  Running setup.py bdist_wheel for pydotplus ... done
  Stored in directory: C:\Users\eaokpah\AppData\Local\pip\Cache\wheels\43\31\48\e1d60511537b50a8ec28b130566d2fbbe4ac302b0def4baa48
  Running setup.py bdist_wheel for pluggy ... done
  Stored in directory: C:\Users\eaokpah\AppData\Local\pip\Cache\wheels\df\44\8e\e136760ae525eac46b3e3db643ef58ff1753177b5a722b0c96
Successfully built pydotplus pluggy
Installing collected packages: backports.functools-lru-cache, cycler, kiwisolver, pyparsing, matplotlib, pydotplus, py, attrs, pluggy
, funcsigs, colorama, pytest, scikit-learn, scipy, patsy, graphviz
```

Figure 10a: A screenshot of installation of requirements for the application

To understand the arguments for each step , execute **python <script name> --help** . This displays additional flags. see figure below

```
C:\Python27\genopheno\genopheno>python preprocess.py --help
usage: preprocess.py [-h] [--user-geno <directory path>]
                     [--known-phenos <file path>] [--snp <directory path>]
                     [--output <directory path>]

optional arguments:
  -h, --help            show this help message and exit
  --user-geno <directory path>, -u <directory path>
                        The directory containing user genomic data. Each file contains data for one user. 23andMe and Ancestry.com data formats are supported. File names must start with the numeric user ID follo
wed by an underscore and end with either 23andme.txt or ancestry.txt.

                        Examples:
                        user44_file19_yearofbirth_1970_sex_XY.23andme.txt
                        user44_file19_yearofbirth_1970_sex_XY.ancestry.txt

                        Default: resources/data/users
  --known-phenos <file path>, -p <file path>
                        The file path to the file that contains the known phenotypes. This is used to train the model. This must be a CSV file with the following format with columns user_id and phenotype.
                        user_id is the numeric user ID. The user ID should match a user ID in the --user-geno directory.
                        phenotype is the phenotype classification for the user. For example, if using eye color then thephenotype column would contain a value of 'Brown' or 'Blue_Green'.

                        Example:
                        user_id,phenotype
                        44,Brown
                        124,Blue_Green
                        55,Blue_Green

                        Default: resources/data/known_phenotypes.csv
  --snp <directory path>, -s <directory path>
                        The directory containing the SNP data for each genome. The supported file format is VCF. Each filemust end in .vcf. Optionally, the files can be compressed with gzip and must end in .gz.

                        Default: resources/data/snp
  --output <directory path>, -o <directory path>
                        The directory that the out files should be written to. This will include all files required for themachine learning input.

                        Default: resources/data/preprocessed
```

Figure 10b: Screenshot showing the various arguments used in the preprocess script.

The application is broken down into three steps, each with their own CLI.

1. Preprocessing the genomic data. This step converts user genotypes at each SNP to a mutation count.

   Note: We will be skipping this step for this tutorial

2. Building the model. This step uses the preprocessed data to build a model to predict phenotype.To build the model, type the command - python model.py

```
(base) c:\Python27\genopheno\genopheno>python model.py
Started reading the preprocessed files...
412 users and 1304138 SNPs for phenotype 'Blue_Green'
416 users and 1304138 SNPs for phenotype 'Brown'
Finished reading the preprocessed files in 246 seconds
Started creating model data set...
805259 (0.38%) SNPs removed due to too many missing user observations for phenotype 'Brown'
```

```
Started building model...
-- Data Summary --
        Negative (Blue_Green):  403 (50.1%)
        Positive (Brown):       402 (49.9%)
        TOTAL:  805

Training Count: 539
Test Count:     266
Number of SNP Features: 31

Fitting 3 folds for each of 54 candidates, totalling 162 fits
[Parallel(n_jobs=1)]: Done 162 out of 162 | elapsed: 18.7min finished
Best estimator params found during grid search: {'max_features': 'sqrt', '
 'max_depth': 7}
Confusion Matrix Metrics:
    Accuracy:    0.917
    Sensitivity: 0.925
    Specificity: 0.91
    TP: 123
```

```
Confusion Matrix Metrics:
    Accuracy:    0.911
    Sensitivity: 0.952
    Specificity: 0.87
    TP: 256
    TN: 235
    FP: 35
    FN: 13

Finished building model in 1139 seconds
Output written to "resources\out"
```

Figure 10c: Screenshot of the Terminal showing successful execution of the model stage

3. Prediction

To use use the model for prediction. type in the command

- python predict.py

```
(base) c:\Python27\genopheno\genopheno>python predict.py
Started calculating mutations for user 1892 (1/23)...
Finished calculating mutations for user 1892 (1/23) in 2 seconds
Started prediction mutations for user 1892 (1/23)...
Finished prediction mutations for user 1892 (1/23) in 0 seconds
Started calculating mutations for user 189 (2/23)...
Finished calculating mutations for user 189 (2/23) in 2 seconds
Started prediction mutations for user 189 (2/23)...
Finished prediction mutations for user 189 (2/23) in 0 seconds
Started calculating mutations for user 200 (3/23)...
Finished calculating mutations for user 200 (3/23) in 2 seconds
Started prediction mutations for user 200 (3/23)...
Finished prediction mutations for user 200 (3/23) in 0 seconds
```

```
Finished calculating mutations for user 53 (23/23) in 2 seconds
Started prediction mutations for user 53 (23/23)...
Finished prediction mutations for user 53 (23/23) in 0 seconds
Output written to "resources\data\prediction"
```

Figure 10d: Screenshot of the Terminal showing successful execution of the prediction stage.

,