**Readme**

**Prerequisites:**

Diversity Browser is distributed as a iPython Jupyter notebook, to run it locally two software need to be installed.

**Install Python Jupyter Notebook**

Anaconda is the recommended software, please follow your platform documentation: <https://www.anaconda.com/distribution/>. There is alternative software available, more details could be found at <https://jupyter.org/install.html>.

**Install PLINK**

PLINK is a free, open-source whole genome association analysis toolset which is picked to do all analyses for the Diversity Browser. Please follow the documentation: <https://www.cog-genomics.org/plink/> to install the toolset

**Get Diversity Browser**

The most convenient way to get the tool is to go to the repository <https://github.com/cnr-ibba/IMAGE-DiversityBrowser>, click the green button, and choose Download Zip option. Unzipped the downloaded file, the tool is there for use.

Along with the tool, there are four data files:

1. IMAGE001\_23\_01\_BPW\_PIG.vcf: the pig reference dataset which contains 149 pigs that were genotyped with the IMAGE ‘*multispecies SNP-chip*’. The ‘*multispecies SNP-chip*’ was specifically designed to handle multiple species, in other words the chip contains around 10,000 SNPs per species, including pigs, cattle, chicken, goat, and sheep.
2. SoI.vcf: the dataset which will be used as the example in the following demonstration
3. legend\_metadata.txt: the dataset which contains the phenotypic (origin and breed) data for the demonstration
4. pigsDataref.vcf: the middle file generated by PLINK from the pig reference dataset

**How to run Diversity Browser**

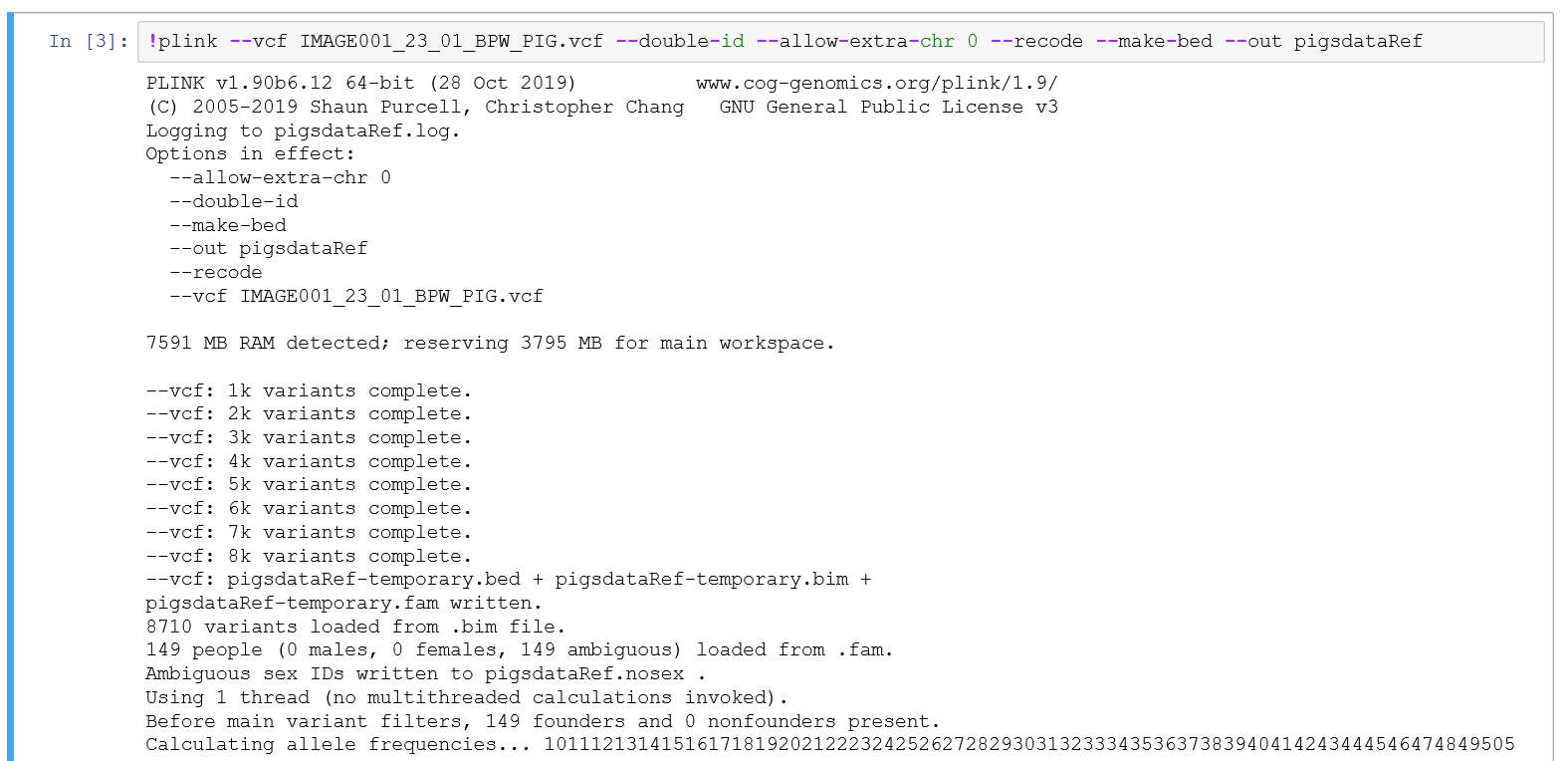
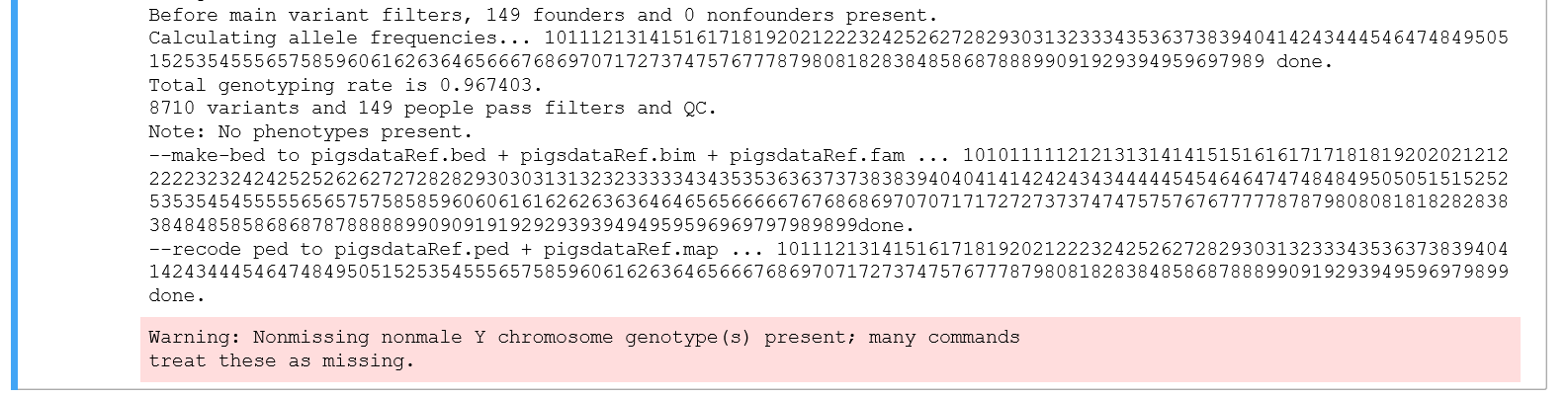
1. Start the notebook: open a terminal, enter the folder where the Diversity Browser is and type the command ‘jupyter notebook’. The notebook page will be opened in your web browser and there is no notebook running. Click the New button to start a new notebook or double click one existing notebook in the file tab.
2. The working folder should be IMAGE-DiversityBrowser. To check, please type the following statement  
   import os  
   os.getcwd()  
   then click the run button, which will provide the output like this:  
   C:/Users/schokker/IMAGE-DiversityBrowser/  
   If not, please use command  
   os.chdir(“C:/Users/schokker/IMAGE-DiversityBrowser/”)  
   to the repository folder.
3. Save your samples data typed by the ‘*multispecies SNP-chip*’ in the same folder. Below we will use the example dataset soI.vcf to do the demonstration.
4. Create the reference dataset (after this step pigsDataref.vcf will be generated in the current folder)

*!plink --vcf IMAGE001\_23\_01\_BPW\_PIG.vcf --double-id --allow-extra-chr 0 --recode --make-bed --out pigsdataRef  
The used options are:*

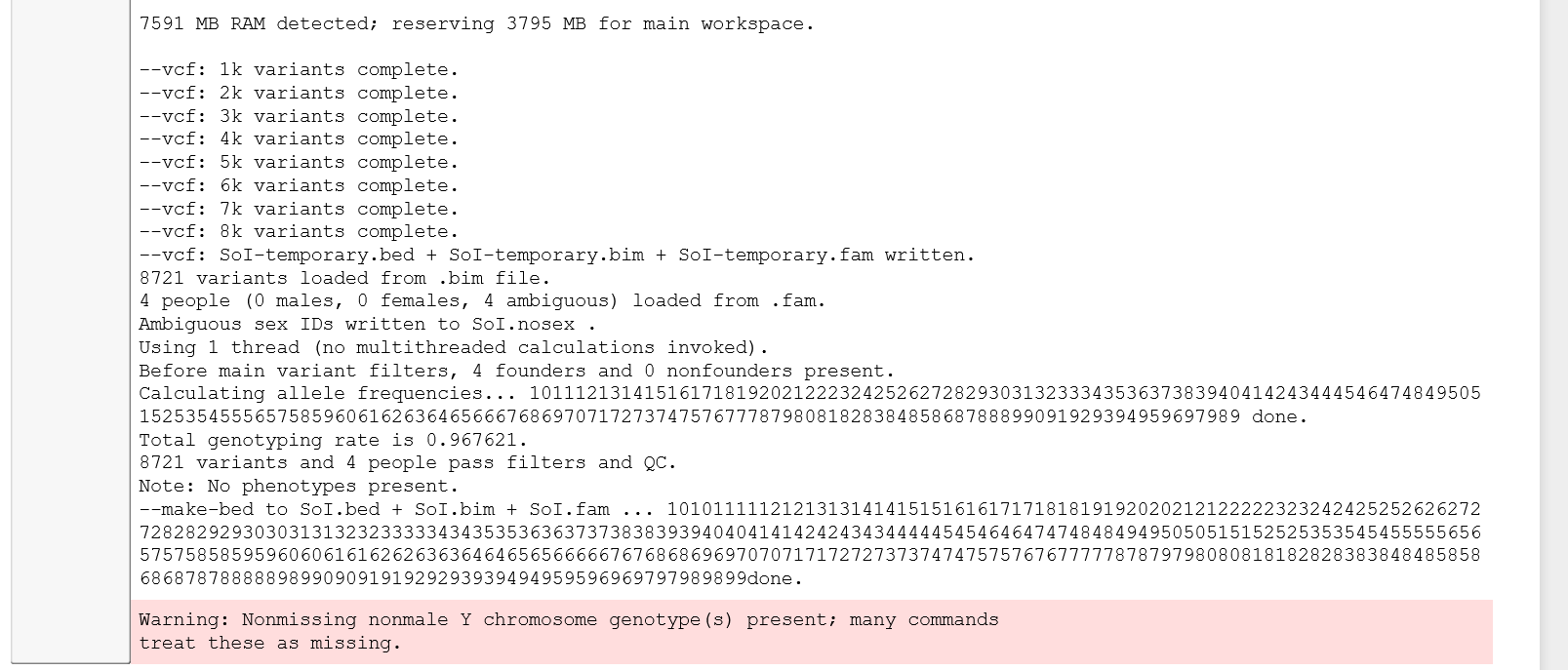
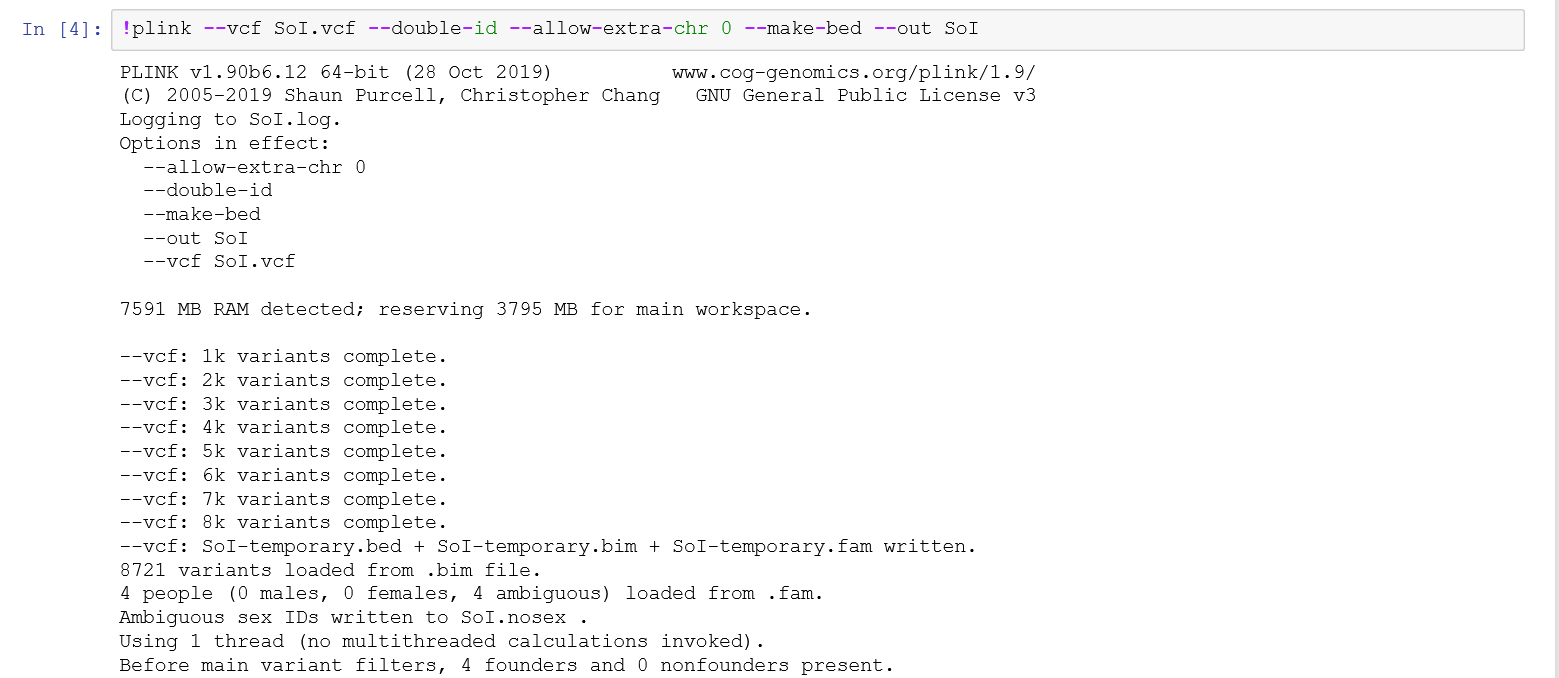
1. *--double-id*
2. *--allow-extra-chr*
3. *--recode*
4. *--make-bed*
5. *--out*

*More details for the option usages please check the PLINK documentation.*

Below is the messages generated by the PLINK.



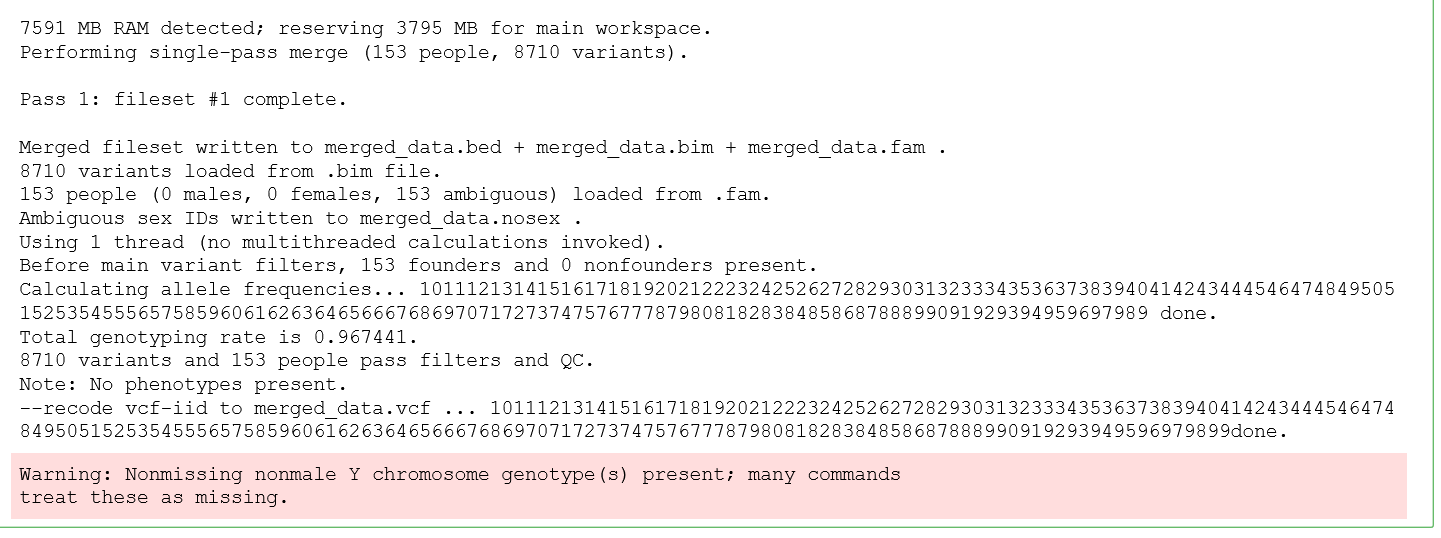
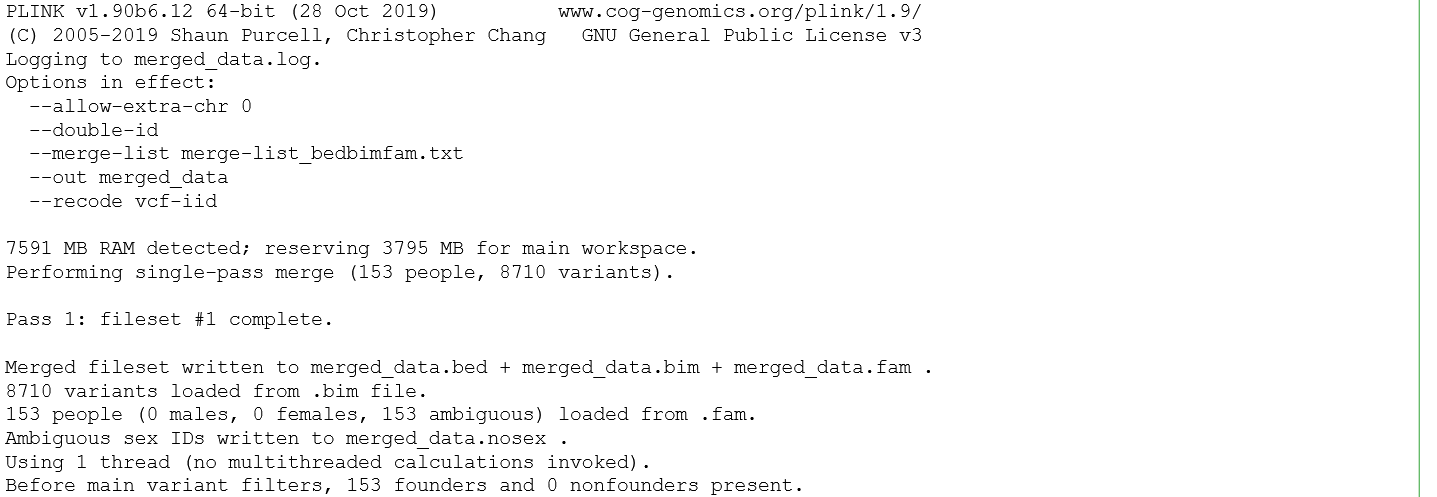
1. Similar to step 4, it is time to generate a file for the samples/dataset of interest. This is done by the next line of code, see below the code and output generated by the Notebook.



1. When both reference and sample of interest files are generated, the merging needs to happen. This is done by the following code within PLINK:

*!plink --merge-list merge-list\_bedbimfam.txt --double-id --allow-extra-chr 0 --recode vcf-iid --out merged\_data*

This will generate the following screen output as well as the file *merged\_data.vcf*. What does the merge-list option do? The latter file will serve as input for the Principal Component Analysis (PCA).



1. Within PLINK it is possible to employ a PCA (code below), here we have set the number of principal components to 25, and generate a new output file named *PCA\_pigsdataRef*.

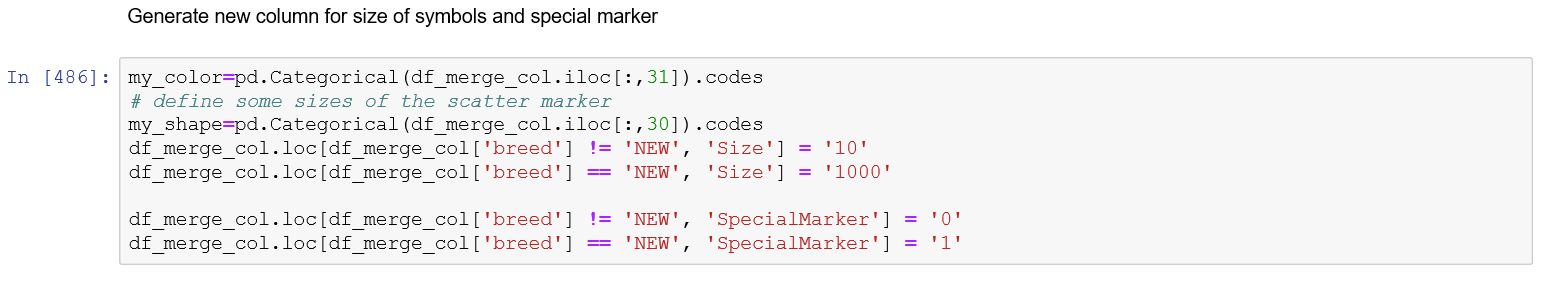
*!plink --vcf merged\_data.vcf --double-id --allow-extra-chr 0 --pca 25 --out PCA\_pigsdataRef*

The output file [*PCA\_pigsdataRef*] contains the eigenvectors needed to create the ‘easy-to-interpret’ scatterplot figure.

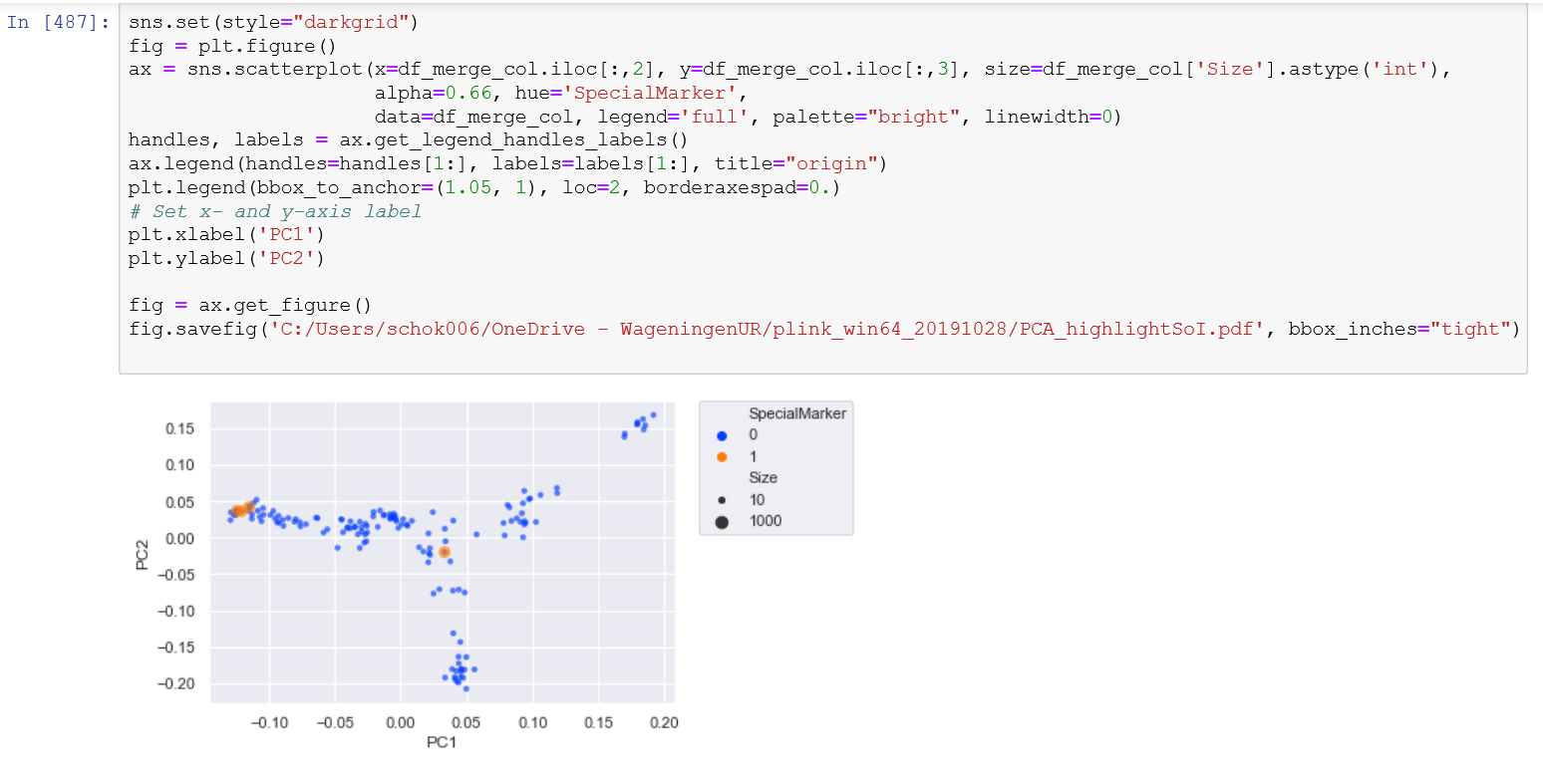
We read the comma-separated file by using pandas - Python Data Analysis Library (for convenience pandas is abbreviated to ‘pd’). From this new data frame called [data], we transform the identifier [IID] column, by removing the “.CEL” text, to generate an identical format to the PCA-output file, as the phenotype-file [legend\_metadata.txt]. The file [legend\_metadata.txt] contains the metadata of each sample, here the samples are individual pigs and the metadata contain origin and breed name. This transformation of the identifiers, i.e. column names, is necessary, because an exact match is needed for the merge of the PCA-output file and the phenotype-file to rule out loss of data. Below the screenshot of the Notebook with the actual code.

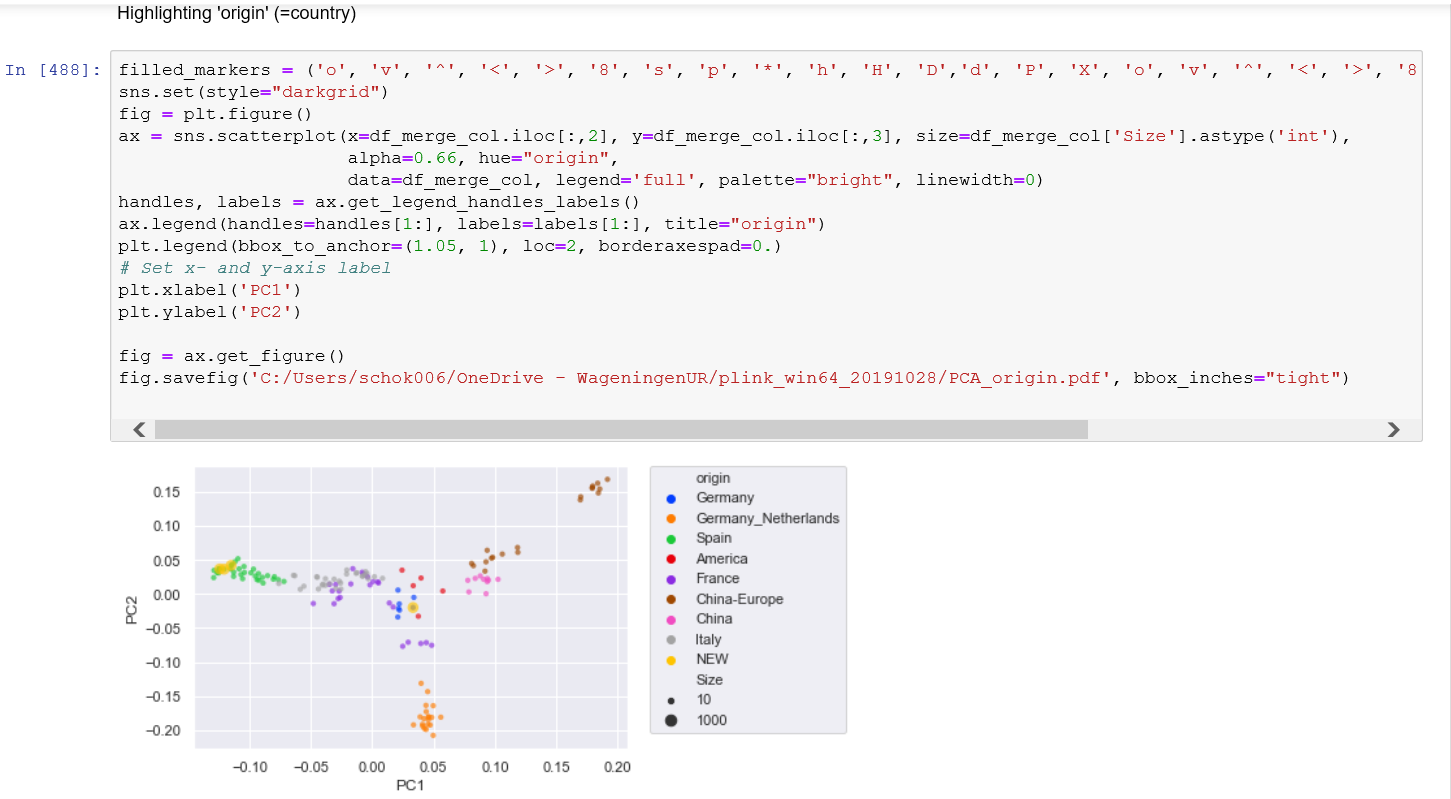


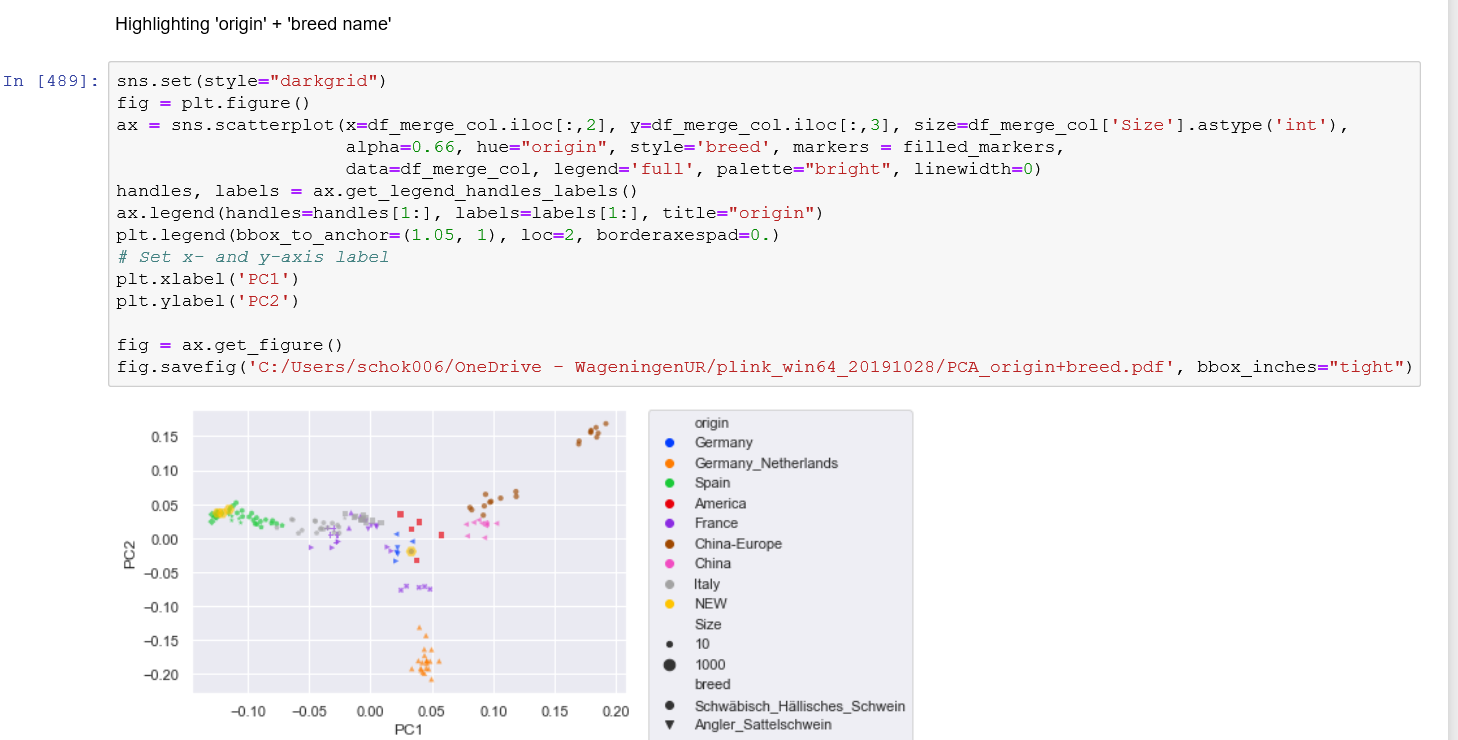
Next step is to generate code that will more easily be translated to different features in the scatterplots which will be generate later on in the Notebook. We generate a variable [my\_color] different colors based upon column 31 of the **‘merged’** dataset, which corresponds to the **origin**. We generate a variable [my\_shape] with different shapes based upon column 30 of the ‘merged’ dataset, which corresponds to the **breed**. And lastly, we generate two new columns *Size* and *SpecialMarker* based upon the column with the **breed** name, namely *Size* and *SpecialMarker*, both to highlight our sample(s) of interest.



The last three blocks of code are generating different flavors of highlighting the sample(s) of interest with the corresponding metadata (origin and/or breed). Below all three block of code with their corresponding output.







The last piece of code generates a three-dimensional (3D) representation of the scatterplot. This may be used when the first two axes do not give a clear separation of the different origins and/or breeds.

