**Readme file for Telomere Length Estimator Project**

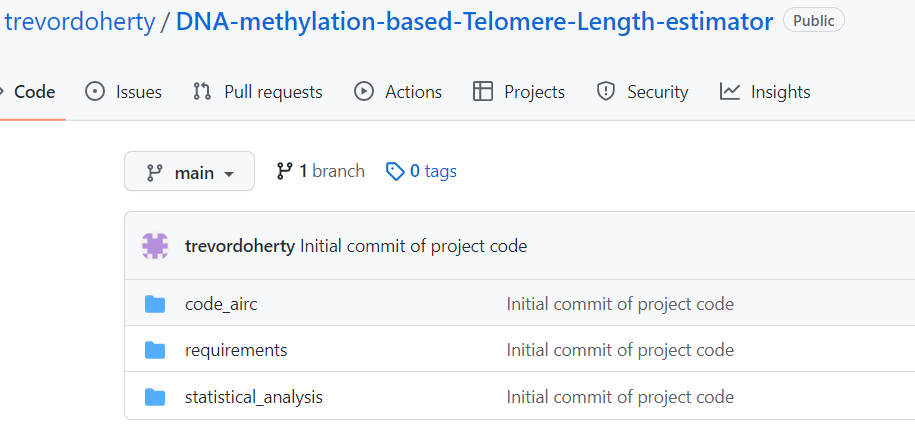
**Section 1: Set up Models**

First, create a virtual environment using the supplied requirements.txt file. This file contains all the Python packages necessary to implement the models.

To run the models on Windows, you can:

1. Set up a virtual environment using python.
2. Go to command prompt.
3. Type python -m venv C:/Users/User/venvs/name (where “C:/Users/User/venvs/name ” is the path you specify for the location of your virtual environment and “name” is the name you want to give the virtual env).
4. Activate the virtual environment with the command: C:/Users/User/venvs/name/Scripts/activate (“name” is the name you gave the virtual environment).
5. The codebase can be retrieved from the GitLab repository: <https://github.com/trevordoherty/DNA-methylation-based-Telomere-Length-estimator>

The GitHub repository looks like this - with subfolders *code\_airc*, *requirements and statistical analysis*:



Within the *code\_airc* folder, there are 2 subfolders:

Graphical user interface, text, application, email

Description automatically generated

1. The folder *nested\_cv\_analysis* contains the scripts for running the nested cross-validation models which utilise the Dunedin data set. *main4.py* is the executable script which calls the other supporting scripts. *get\_data4.py* contains the functionality to read in the data files, applying preprocessing, cleaning and structuring steps. *models4\_gridsearch.py* contains the code for applying the nested CV analysis with hyperparameter tuning and model selection. *feature\_selection4.py* contains the functions for applying the feature selection and transformation methods. Note that *launch\_en\_timings.sh* is the shell script that was used for running models on the local cluster.

Graphical user interface, text, application, email

Description automatically generated

1. The folder *final\_estimators* contains scripts for running the models which were developed using the discovery data set (Dunedin) and evaluated on the 2 independent data sets (EXTEND and TWIN). *get\_data.py* contains the functionality to read in the data files, applying preprocessing, cleaning and structuring steps. *feature\_selection.py* contains the functions for applying the feature selection and transformation methods. *test\_sets\_evaluation.py* contains the main executable code with functions and code for developing models on the discovery data and evaluating these models on the 2 independent test data sets. The *utils.py* script contains supporting utility functions.

Graphical user interface, text, application, email

Description automatically generated

1. The *statistical\_analysis* folder contains the following files:

Graphical user interface, text, application, email

Description automatically generated

The Jupyter notebook *tuning\_and\_feat\_sel.ipynb* contains functions for the creation of graphs for the paper and visualisation of results. The plots showing the performance of feature selection approaches in the nested CV analysis and the number of features remaining in models at each stage of feature reduction are shown in this file.

The notebook *final\_statistical\_analysis\_fs\_paper.ipynb* contains the various statistical analysis including correlation analysis between actual/estimated TL and a range of subject clinical and phenotypic data such as blood cell concentrations etc. It also contains the association analyses between actual/estimated TL and age controlling for a range of confounding variables. Additionally, the correlation plots between age and TL, actual/estimated TL and monocyte counts etc. are generated in this file.

1. The *requirements* folder contains the requirements.txt file for loading the necessary python libraries to run models.
2. Navigate to the *requirements* folder within the terminal and enter the command: pip install -r requirements.txt

(This will load all necessary Python packages within the activated virtual environment)

1. It should now be possible to run Jupyter notebooks and python scripts within this virtual environment.
2. For any Jupyter notebooks, navigate to the folder containing the relevant notebook and type: jupyter notebook - this should activate Jupyter notebook in a browser window.
3. Once the code is cloned to your local computer, there should be a folder structure with files as below (this mirrors the GitHub repository folder structure):

Graphical user interface, application

Description automatically generated

1. It will be necessary to acquire the data files in order to run the models. The data is not publicly available but can be requested.

Assuming that the data has been obtained, it is necessary to create a folder in which to place it. Therefore, create a static folder within the top-level file structure. Additionally, create a results folder for storing results files.

Table

Description automatically generated with medium confidence

1. The file paths in the code have been adjusted to be relative i.e., the code should pick up the data and output results in the appropriate folder if the file structure is adhered to.
2. To run the nested cross-validation analysis, navigate to the folder *nested\_cv\_analysis* and run the script main4.py. This is the main executable file. This script calls the functions that read in and preprocess the data, construct models using each of the feature selection methods and saves the results of the nested CV analysis.
3. Each of the scripts are described in more detail in Section 2 of this document “File Descriptions”.

**Section 2: File Descriptions**

The models need to run with the entire data sets i.e., up to approximately 1600 samples with ~420k features. Given the memory and computational requirements, the full models cannot be run on a standard laptop and require cluster computing resources. The models reported in our paper were run on either the TUDublin AIRC computing cluster or the Irish Centre for High End Computing (ICHEC) cluster.

**The following files relate to the nested cross-validation analysis – contained in the nested\_cv\_analysis folder.**

**main4.py**

This file is the main executable used in the nested CV analysis. It calls the functions that read in and preprocess the data, apply each of the feature selection techniques and run the nested CV analysis.

The models include: pearson, mutual info, linear SVR coeff. rankings, random forest importances, F-test with FDR (0.01), F-test with FDR (0.05), BoostaRoota and PCA. All feature pre-selection methods were run in advance of applying elastic net for model training and testing. Elastic net regression itself acts as a second stage of feature selection. A baseline model utilising elastic net with no prior feature selection was also developed.

Location: “../code\_airc/nested\_cv\_analysis/”

Input data location: “../static/”

Results location: “../results/…?...”, “../rankings/…?...”

**models4\_gridsearch.py**

This script contains the functionality for running the nested CV analysis. The structure of the 5 x 3 nested CV is coded here with the hyperparameter tuning conducted using grid search over the elastic net parameter ranges. The performance metrics are generated with results saved for each model.

Location: “../code\_airc/nested\_cv\_analysis/”

**feature\_selection4.py**

This script contains the code for running each of the feature selection/reduction methods. These include those approaches utilising Pearson correlation rankings between each input feature and the response variable, mutual information, linear SVR rankings, random forest feature importances, F-tests with FDR (0.01, 0.05), BoostaRoota, PCA and the baseline model.

Location: “../code\_airc/nested\_cv\_analysis/”

**get\_data4.py**

This script contains code for importing the various data sets and applying cleaning and preprocessing steps. This includes structuring and merging data sets, filtering on specific/common columns and dropping columns with missing values.

Location: “../code\_airc/nested\_cv\_analysis/”

**utils4.py**

This script specifies the parameter ranges for model selection for each learning algorithm. It also contains functions relating to performance metrics and storage of results.

Location: “../code\_airc/nested\_cv\_analysis/”

**The following files relate to the models trained on the full Dunedin data set and applied to the 2 independent test sets (EXTEND and TWIN) – contained in the final\_estimators folder.**

**test\_sets\_evaluation.py**

This file contains the code for running each model that is trained on the full Dunedin data set and tested on both independent test sets i.e., EXTEND and TWIN. This file utilises the same modules used in the nested cross-validation analysis above i.e., feature\_selection4.py, get\_data4.py and utils4.py.