ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΤΜΗΜΑ ΙΑΤΡΙΚΗΣ

Πρόγραμμα Μεταπτυχιακών Σπουδών
"Μεθοδολογία Βιοϊατρικής Έρευνας, Βιοστατιστική και Κλινική Βιοπληροφορική"

Μεταπτυχιακή Διατριβή / Master Thesis

Τίτλος

Ανάπτυξη λογισμικού στη γλώσσα προγραμματισμού Python για την υλοποίηση διαδικασιών τυχαιοποίησης (απλή, στρωματοποιημένη και τετραγωνισμένη τυχαιοποίηση)

Title

Develop a software in Python for performing different types of randomization (simple, stratified and block randomization)

Βασίλειος Τζιμούρτος
A.M.: M060616008

Τριμελής Συμβουλευτική Επιτροπή

Δρ.	Δρ.	Δρ.
Ιωάννης Στεφανίδης	Χρυσούλα Δοξάνη	Ηλίας Ζιντζαράς
Καθηγητής (επιβλέπων)		Καθηγητής

Λάρισα 2017

Περιεχόμενα

Περίληψη

Ο όρος “Τυχαιοποίηση” αφορά τη διαδικασία της τυχαίας ανάθεσης με στόχο την εξάλειψη της μεροληψίας επιλογής. Με τον τρόπο αυτό διασφαλίζεται πως οι συγκρίσεις που γίνονται είναι δίκαιες και η στατιστική ανάλυση μαθηματικά έγκυρη. Η διατριβή έχει σαν στόχο την ανάπτυξη ενός λογισμικού που υλοποιεί κάποιες από τις γνωστές μεθόδους της διαδικασίας τυχαιοποίησης. Πιο συγκεκριμένα θα παρέχεται η δυνατότητα εκτέλεσης της διαδικασίας για την απλή, τη στρωματοποιημένη και τετραγωνισμένη τυχαιοποίηση.

Στην υλοποίηση του λογισμικού χρησιμοποιήθηκε η γλώσσα προγραμματισμού Python και το περιβάλλον ανάπτυξης Eclipse ενώ αξιοποιήθηκαν κατάλληλα πακέτα εργαλείων λογισμικού (SciKits, Opal). Τα SciKits αποτελούν επιπλέον λειτουργίες του SciPy και είναι διαθέσιμα δωρεάν με τη μορφή λογισμικού ανοιχτού κώδικα παρέχοντας τις απαραίτητες μαθηματικές λειτουργίες στο λογισμικό.

Το λογισμικό παρέχει τη δυνατότητα επιλογής και εκτέλεσης των τριών διαθέσιμων μεθόδων τυχαιοποίησης με τις κατάλληλες παραμέτρους εισαγωγής ανά περίπτωση. Η εκτέλεσή του παράγει μια τυχαία λίστα ή ομάδα λιστών που μπορεί να χρησιμοποιηθεί σε διαδικασίες που απαιτούν τυχαιότητα στις αναθέσεις.

Η υλοποίηση μεθόδων τυχαιοποίησης μέσω λογισμικού δίνει τη δυνατότητα εύκολης εξισορρόπησης των πιθανοτήτων μεταξύ ομάδων ελέγχου. Παράλληλα η αξιοποίηση ανοιχτών και δωρεάν τεχνολογιών δίνει τη δυνατότητα ελέγχου και εξέλιξης του πηγαίου κώδικα συνεισφέροντας στην διατήρηση της ποιότητας λογισμικού.

Abstract

The term “Randomization” refers to the procedure of allocating experimental units via random assignment in order eliminates the selection bias. This way it is possible to balance the groups with respect to fairness and form a sound basis for the use of statistical tests.

This thesis aims to develop a software for performing different types of randomization. More to the point, the software will implement the following types: simple, stratification and block randomization.

In developing the software Python programming language and the Eclipse I.D.E. was used along with Scientific Toolkits (SciKits) and Opal software packages. SciKits are add-on packages for SciPy distributed under Open Source Initiative approved licenses and offer the mathematical functions required for the software.

The software offers the option to choose any of the three randomization methods and give the required input data. The outcome is a list or group of lists that we can used for random allocations.

Implementing randomization procedures through software give us an easy and fast way to eliminate selection bias and balance the control groups. At the same time, the use of open source and free technologies enables quality control and evolution of the source code components through public contribution.

1. Εισαγωγή

Η διαδικασία της τυχαιοποίησης στοχεύει στο να έχει κάθε πειραματική μονάδα την ίδια πιθανότητα να δεχθεί κάθε επέμβαση. Με τον τρόπο αυτό εξασφαλίζεται:

- η αμερόληπτη εκτίμηση του πειραματικού σφάλματος, των μέσων όρων των επεμβάσεων και των διαφορών τους
- η ανεξαρτησία των παρατηρήσεων, απαραίτητη προϋπόθεση για τις δοκιμασίες σημαντικότητας

Η εφαρμογή των διαδικασιών αυτών σε μια κλινική μελέτη αποσκοπεί στην κατανομή των ατόμων της μελέτης στα διάφορα σκέλη κατά τρόπο που οι διαφορές μεταξύ των σκελών να οφείλονται στην τύχη και μόνο.

Στη συνέχεια δίνεται μια σύντομη περιγραφή των μεθόδων τυχαιοποίησης με τις οποίες θα ασχοληθούμε στο πλαίσιο της συγκεκριμένης μεταπτυχιακής διατριβής.

1.1 Απλή Τυχαιοποίηση

Η τυχαιοποίηση η οποία βασίζεται σε μια ακολουθία τυχαίων αριθμών είναι γνωστή ως απλή τυχαιοποίηση (simple randomization). Η μέθοδος αυτή διατηρεί την τυχαιότητα με την οποία γίνεται η ανάθεση σε κάποια από τις διαθέσιμες ομάδες. Οι τυχαίοι αριθμοί που χρησιμοποιούνται μπορούν να προέρχονται από ένα πίνακα τυχαίων αριθμών ενός βιβλίου μαθηματικών ή να δημιουργηθεί με χρήση λογισμικού ηλεκτρονικού υπολογιστή. Αυτή η προσέγγιση στην τυχαιοποίηση είναι απλή και εύκολη να υλοποιηθεί ενώ σε μεγάλα πλήθη που απαιτούν τυχαιοποίηση μπορεί να εγγυηθεί την ομοιόμορφη κατανομή ανάμεσα στις διαθέσιμες ομάδες. Παρόλα αυτά προβλήματα με τη μορφή ανομοιόμορφης κατανομής μπορούν να εμφανιστούν στην περίπτωση που η τυχαιοποίηση αφορά μικρό σε μέγεθος πλήθος.

1.2 Τετραγωνισμένη Τυχαιοποίηση (Block)

Η μέθοδος της τετραγωνισμένης τυχαιοποίησης περιορίζει την τυχαιοποίηση ώστε να διασφαλίσει τη δημιουργία ισοδύναμων σε πλήθος ομάδων. Στη μέθοδο αυτή θεωρούμε πως τα υπό τυχαιοποίηση αντικείμενα ανήκουν σε τετράγωνα / μπλοκ ενός συγκεκριμένου μεγέθους και η ανάθεση των διαθέσιμων ομάδων γίνεται τυχαία στα μπλοκ που υπάρχουν.

Η μέθοδος της τετραγωνισμένης τυχαιοποίησης είναι με τέτοιο τρόπο σχεδιασμένη ώστε να τυχαιοποιεί τους στόχους ώστε να δημιουργήσει ισοπληθείς ομάδες. Στόχος της μεθόδου είναι να διασφαλίσει την ισορροπία στα μεγέθη των ομάδων που δημιουργούνται. Τα χρησιμοποιούμενα μπλοκ είναι μικρά και ίδια σε μέγεθος μεταξύ τους διατηρώντας με αυτόν τον τρόπο την ισορροπία μεταξύ των ομάδων. Το μέγεθος των μπλοκ καθορίζεται από τον εκάστοτε χρήστη και πρέπει να είναι αριθμός πολλαπλάσιος του πλήθους των διαθέσιμων ομάδων. Για παράδειγμα, για δύο ομάδες το μέγεθος των μπλοκ θα μπορούσε να είναι 4 ή 6 ή 8. Μετά τον καθορισμό του μεγέθους των μπλοκ και την τυχαία ανάθεση αριθμών σε αυτά, μπορούν να χρησιμοποιηθούν για τον καθορισμό των αναθέσεων των ομάδων.

1.3 Στρωματοποιημένη Τυχαιοποίηση

Η μέθοδος της στρωματοποιημένης τυχαιοποίησης έρχεται να καλύψει την ανάγκη ύπαρξης ελέγχου και εξισορρόπησης του βαθμού στον οποίο επηρεάζουν οι συμμεταβλητές (covariates). Η μέθοδος αυτή μπορεί να χρησιμοποιηθεί ώστε να επιτευχθεί ισορροπία μεταξύ ομάδων στο πλαίσιο των ειδικών χαρακτηριστικών (covariates) κάθε πειραματικής μονάδας. Οι συμμεταβλητές θα πρέπει να διευκρινιστούν από τον χρήστη της μεθόδου και να κατανοηθεί το πως μπορεί να επηρεάζει κάθε μία από αυτές την εξαρτώμενη μεταβλητή. Η στρωματοποιημένη τυχαιοποίηση επιτυγχάνεται με την δημιουργία ενός μπλοκ για κάθε συνδυασμό συμμεταβλητών και οι πειραματικές μονάδες αναθέτονται στο κατάλληλα μπλοκ συμμεταβλητών. Από τη στιγμή που όλες οι πειραματικές μονάδες έχουν αντιστοιχηθεί σε κάποιο μπλοκ, η διαδικασία της απλής τυχαιοποίησης εφαρμόζεται στα περιεχόμενα κάθε μπλοκ ώστε να γίνει ανάθεση των μονάδων στις διαθέσιμες ομάδες.

2. Μέθοδοι και Εργαλεία

Η ανάπτυξη του λογισμικού ζητείται να γίνει στη γλώσσα προγραμματισμού Python και να υλοποιεί τις μεθόδους της απλής, τετραγωνισμένης και στρωματοποιημένης τυχαιοποίησης.

Η Python είναι μια υψηλού επιπέδου γλώσσα προγραμματισμού η οποία δημιουργήθηκε το 1990. Ο κύριος στόχος της είναι η αναγνωσιμότητα του κώδικά της και η ευκολία χρήσης της ενώ το συντακτικό της επιτρέπει στους προγραμματιστές να εκφράσουν έννοιες σε λιγότερες γραμμές κώδικα. Διακρίνεται λόγω του ότι έχει πολλές βιβλιοθήκες που διευκολύνουν ιδιαίτερα αρκετές συνηθισμένες εργασίες και για την ταχύτητα εκμάθησης της.

Η Python μπορεί να επεκταθεί προσθέτοντάς της καινούργια modules που να είναι γραμμένα σε μια γλώσσα που μεταγλωττίζεται, όπως είναι η C ή η C++. Αυτά τα modules επέκτασης μπορούν να ορίσουν νέες συναρτήσεις (functions) και μεταβλητές (variables) καθώς επίσης και καινούργιους τύπους δεδομένων (object types).

Οι διερμηνευτές της Python είναι διαθέσιμοι για εγκατάσταση σε πολλά λειτουργικά συστήματα, επιτρέποντας στην Python την εκτέλεση κώδικα σε ευρεία γκάμα συστημάτων. Η Python αναπτύσσεται ως ανοιχτό λογισμικό (open source) και ο κώδικας διανέμεται με την άδεια Python Software Foundation License η οποία είναι συμβατή με την GPL.

Για τη συγγραφή προγραμμάτων είναι απαραίτητος ένας κειμενογράφος ή ακόμα καλύτερα ένα ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment - IDE), το οποίο είναι ένα ειδικό λογισμικό για την ανάπτυξη εφαρμογών. Το IDE μας δίνει τη δυνατότητα να χρησιμοποιήσουμε διαδραστικά τον διερμηνευτή της γλώσσας, να γράψουμε και να επεξεργαστούμε προγράμματα, να τα αποθηκεύσουμε σε αρχεία, να τα εκτελέσουμε, να κάνουμε αποσφαλμάτωση καθώς και άλλες χρήσιμες λειτουργίες.

Το Eclipse είναι μία πλατφόρμα ανάπτυξης λογισμικού η οποία διατίθεται δωρεάν, είναι ανοιχτού κώδικα και αποτελείται από ένα IDE και ένα plug-in σύστημα ώστε να είναι εύκολη η επέκτασή της. Η ανάπτυξή του έχει γίνει σε γλώσσα Java και χρησιμοποιείται για την ανάπτυξη εφαρμογών σε Java, αλλά με την χρήση διάφορων plug-ins και σε άλλες γλώσσες όπως: C, C++, COBOL, Python, PHP. Το Eclipse θα χρησιμοποιηθεί για την ανάπτυξη του λογισμικού σε συνεργασία με το plug-in PyDev το οποίο δίνει τη δυνατότητα συγγραφής κώδικα Python στο περιβάλλον αυτό.

Για την διερεύνηση και εγκατάσταση των βιβλιοθηκών που υπάρχουν διαθέσιμες για την Python υπάρχει το pip. Το pip είναι ένα εργαλείο που βοηθά στην διαχείριση και εγκατάσταση πακέτων της και παρέχεται μαζί με την εγκατάστασή της Python >= 2.7.9 ή Python >= 3.4.

Το SciPy, είναι ένα οικοσύστημα λογισμικών ανοιχτού κώδικα βασισμένων στη γλώσσα Python που αφορούν τα μαθηματικά, τη μηχανική και άλλες θετικές επιστήμες. Επιπλέον του λογισμικού που αποτελεί τον πυρήνα του SciPy προσφέρονται επίσης και κάποια επιπλέον πακέτα που υλοποιούν πιο συγκεκριμένες λειτουργίες. Τα επιπλέον αυτά πακέτα ονομάζονται SciKits.

Τα SciKits (από το SciPy Toolkits) είναι βιβλιοθήκες λογισμικού που συντηρούνται σε ξεχωριστό αποθετήριο κώδικα από αυτό του SciPy. Όλες οι βιβλιοθήκες έχουν όνομα της μορφής SciKits.* και διανέμονται με βάση άδειες ανοιχτού λογισμικού. Τα SciKits καλύπτουν ένα μεγάλος εύρος τομέων εφαρμογής όπως υπολογιστική χρηματοοικονομική, επεξεργασία ήχου, γεωεπιστήμες, μηχανική όραση, μηχανική μάθηση, ιατρική υπολογιστική και βιοπληροφορική.

Για την υλοποίηση του λογισμικού τυχαιοποίησης μπορεί να χρησιμοποιηθεί το πακέτο randomization του αποθετηρίου pip, όπως φαίνεται στο παρακάτω αποτέλεσμα αναζήτησης του εργαλείου:

```
user@hostname:~$ pip show randomization
Name: randomization
Version: 1.0.0
Summary: Functions to randomize subjects in clinical trials.
Home-page: http://github.com/louden/randomization
Author: Christopher Louden
Author-email: chris@loudenanalytics.com
License: MIT
Location: /usr/local/lib/python2.7/dist-packages
Requires:
```

Ωστόσο όπως αναφέρει και στην ιστοσελίδα του λογισμικού στο github, το συγκεκριμένο πακέτο έχει πλέον ενσωματωθεί και συνεχίζει να υποστηρίζεται μέσω του πακέτου scikit-Design. Το πακέτο αυτό μπορεί να χρησιμοποιηθεί στο σχεδιασμό στατιστικών μελετών όπως π.χ. Κλινικές Μέθοδοι και Έλεγχοι Ποιότητας. Το scikit-Design δεν περιλαμβάνεται ακόμα στο αποθετήριο του pip αλλά θα πρέπει να το κατεβάσουμε από το github και να το εγκαταστήσουμε με το pip. Για παράδειγμα σε ένα σύστημα Linux θα εκτελούσαμε τις παρακάτω δυο εντολές:

```
> git clone https://github.com/louden/scikit-design.git
PATH_TO_FOLDER
> sudo -H pip install -e PATH_TO_FOLDER
```

Μετά τα παραπάνω είμαστε πλέον σε θέση να χρησιμοποιήσουμε την βιβλιοθήκη στον κώδικά μας με κατάλληλες κλήσεις προς αυτή. Για τις συναρτήσεις που μας ενδιαφέρουν μπορούμε να συμβουλευτούμε τη βοήθεια της βιβλιοθήκης randomization. Για την ανάπτυξη του λογισμικού της διατριβής θα χρησιμοποιήσουμε τις παρακάτω συναρτήσεις που είναι πλέον διαθέσιμες:

- **randomization.simple**(n_subjects, n_groups, p=None, seed=None)
- **randomization.block**(n_subjects, n_groups, block_length, seed=None)
- **randomization.stratification**(n_subjects_per_strata, n_groups, block_length=4, seed=None)

Για την οργάνωση των ερευνητικών μονάδων/ ασθενών θα χρησιμοποιηθεί το λογισμικό Opal. Το Opal αποτελεί μια full stack βιβλιοθήκη ανάπτυξης εργαλείων λογισμικού για υπηρεσίες υγείας και βασίζεται πάνω στο λογισμικό Django, Angular

και Bootstrap δίνοντας τη δυνατότητα γρήγορης ανάπτυξης και συντήρησης κλινικών εφαρμογών. Στο πλαίσιο της συγκεκριμένης διατριβής θα χρησιμοποιήσουμε το Opal για τη λειτουργία καταχώρησης ασθενών (List Views) που παρέχει και σε συνδυασμό με τη δημιουργία ενός μικρού module θα δώσουμε την επιπλέον δυνατότητα στο λογισμικό τυχαιοποίησης να διαβάζει το πλήθος ασθενών κάθε μεθόδου τυχαιοποίησης από τη βάση του Opal.

Για την εγκατάσταση του Opal χρειάζεται να έχουμε εγκατεστημένη την python 2.7 και να εκτελέσουμε την εντολή `pip install opal`. Στη συνέχεια δημιουργούμε μια εφαρμογή βασισμένη στο Opal. Μεταβαίνουμε στο φάκελο της επιλογής μας όπου θέλουμε να αποθηκευτεί η εφαρμογή και τη δημιουργούμε με την εντολή:

```
> opal startproject ΟΝΟΜΑ_ΕΦΑΡΜΟΓΗΣ
```

Η εφαρμογή θα δημιουργηθεί στο φάκελο που βρισκόμαστε και σε φάκελο με όνομα ΟΝΟΜΑ_ΕΦΑΡΜΟΓΗΣ. Με την ολοκλήρωση της εντολής θα έχει δημιουργηθεί όλη η δομή της εφαρμογής. Μπορούμε επίσης να εισάγουμε πληροφορίες από λίστες αναφοράς όπως η elCID εκτελώντας τις παρακάτω δύο εντολές:

```
>wget
https://raw.githubusercontent.com/openhealthcare/elcid/master/data
/lookuplists/lookuplists.json -P ΟΝΟΜΑ_ΕΦΑΡΜΟΓΗΣ/data/lookuplists

>python manage.py load_lookup_lists
```

Για να εκτελέσουμε την εφαρμογή πηγαίνουμε στο φάκελό της και εκτελούμε το server συστατικό της όπως φαίνεται στις παρακάτω εντολές:

```
cd ΟΝΟΜΑ_ΕΦΑΡΜΟΓΗΣ
python manage.py runserver
```

Η εφαρμογή είναι προσβάσιμη μέσω ενός φυλλομετρητή στη διεύθυνση localhost:8000 με αρχικά στοιχεία σύνδεσης το username: super και password: super1.

Για τη δημιουργία του module που αναφέρθηκε προηγουμένως εκτελούμε τις παρακάτω ενέργειες σύμφωνα με το documentation του Opal. Προσθέτουμε τις παρακάτω γραμμές στο τέλος του αρχείου models.py της Opal εφαρμογής μας:

```
class RandomizePatient(models.EpisodeSubrecord):
        simple_randomization = fields.BooleanField(default=False)
        block_randomization  = fields.BooleanField(default=False)
        strata_randomization = fields.BooleanField(default=False)
        strata_name     = fields.TextField(blank=True, null=True)
        _icon = 'fa fa-th-list'
```

Στη συνέχεια εκτελούμε την παρακάτω εντολή ώστε να ενημερωθεί η δομή της εφαρμογής μας με τη νέα κλάση που προσθέσαμε:

```
> opal scaffold mynewapp
```

Μετά την επιτυχή εκτέλεση της εντολής θα έχουν δημιουργηθεί τα κατάλληλα αρχεία. Για την εμφάνιση του module στη λίστα ασθενών θα πρέπει να προσθέσουμε το όνομα του module στο αρχείο patient_lists.py και στο κομμάτι του schema, όπως φαίνεται στην παρακάτω έντονη γραμμή:

```
schema = [
    models.Demographics,
    models.Diagnosis,
    models.Treatment,
    models.RandomizePatient
]
```

Για την εμφάνιση του module ίσως χρειαστεί να σταματήσουμε το server συστατικό με Control+C και να το επανεκκινήσουμε με την εντολή που έχουμε αναφέρει.

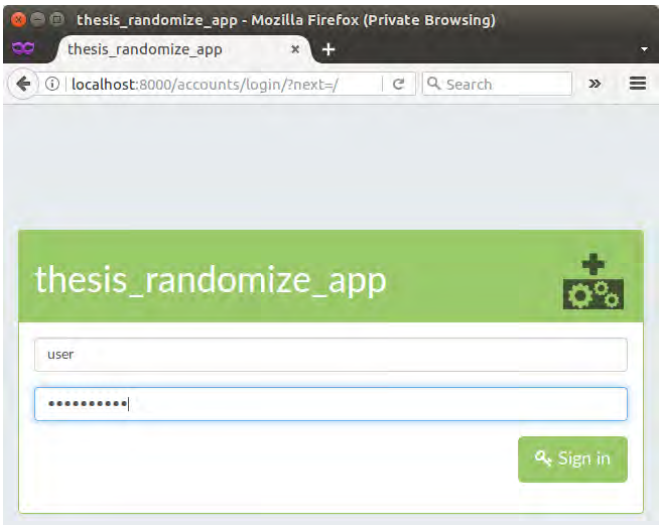
3. Αποτελέσματα

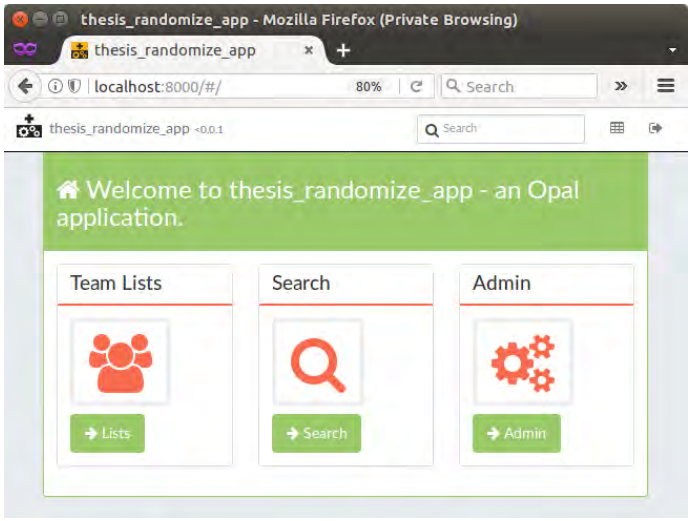
Αξιοποιώντας τις αναφερόμενες στο προηγούμενο κεφάλαιο μεθόδους, εργαλεία και τεχνολογίες αναπτύχθηκε το ζητούμενο λογισμικό του οποίου ο κώδικας υπάρχει παρακάτω:

```
import sqlite3
import os
from skdesign.randomization import randomization

def opal_db_connect(randomization_method):
    try:
        conn =
sqlite3.connect('/home/billtzim/Documents/PMS_BIOSTATS/Thesis/software/opal_sw/thesis_randomize_app/thesis_randomize_app/opal.sqlite')

        if randomization_method.lower() == "simple":
            sqlquery = "select count(simple_randomization) from thesis_randomize_app_randomizepatient where simple_randomization=1;"
        elif randomization_method.lower() == "block":
            sqlquery = "select count(block_randomization) from thesis_randomize_app_randomizepatient where block_randomization=1;"
        elif randomization_method.lower() == "strata":
            sqlquery = "select count(strata_randomization),strata_name from thesis_randomize_app_randomizepatient where strata_randomization=1 and strata_name<>'' group by strata_name;"
        else:
            print("Query type parameter error, valid are simple / block / strata")
            return

        cursor = conn.execute(sqlquery)

        if randomization_method != "strata":
            return cursor.fetchone()
        else:
            return cursor.fetchall()
    finally:
            conn.close()

while True:
    print("Randomization methods")
    print("Choose a method by typing the corresponding number")
    print("\n\tMenu")
    print("1. Simple randomization")
    print("2. Block randomization")
    print("3. Stratified randomization")

    menu_choice = input("Please make a choice (1-->3): ")

    if menu_choice>=1 and menu_choice<=3:
        break

print("\nYou choose %d" %menu_choice) #,"\n"

if menu_choice == 1:
    source_choice = raw_input("Get subjects from Opal database? (y/N): ") or "n"
    if str(source_choice).lower() == 'y':
        subj_number = opal_db_connect("simple")[0]
        print("\nSubjects for simple randomization in Opal db: %d\n" %subj_number)
```

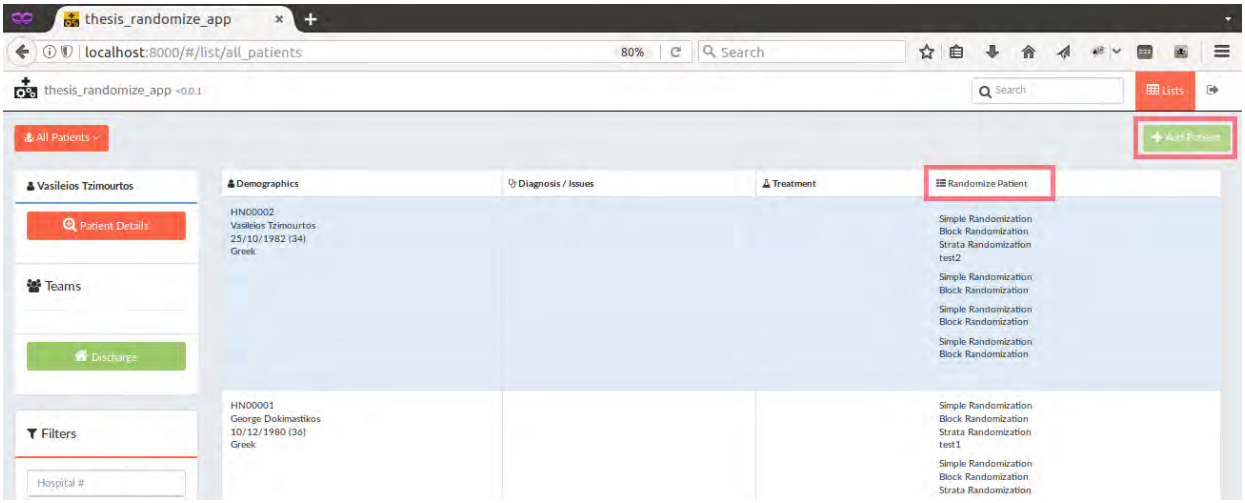
```
    else:
        subj_number = input("Give number of subjects to randomize:")
    group_number = input("Give number of groups to randomize subjects to:")
    simple = randomization.simple(subj_number,group_number)
    print("\nSimple randomization results for %d \n %s \n" %(subj_number,simple)) #,subj_number,"\n",simple
    f = open('simple_randomization_output', 'w')
    for itm in range(len(simple)):
        msg = "Patient "+str(itm+1)+" goes in group "+str(simple[itm])
        print(msg)
        f.write(msg+'\n')
    f.close()
    print("\nResults saved to file %s/simple_randomization_output" %os.getcwd())
elif menu_choice == 2:
    source_choice = raw_input("Get subjects from Opal database? (y/N): ") or "n"
    if str(source_choice).lower() == 'y':
        subj_number = opal_db_connect("block")[0]
        print("\nSubjects for block randomization in Opal db: %d\n" %subj_number)
    else:
        subj_number = input("Give number of subjects to randomize:")

    group_number = input("Give number of groups to randomize subjects to:")
    block_length = input("Give block length (value of `block_length` should be a multiple of `groups` to ensure proper balance):")
    block = randomization.block(subj_number,group_number,block_length)
    print("block randomization results for %d \n %s \n" %(subj_number,block))

    f = open('block_randomization_output', 'w')
    for itm in range(len(block)):
        msg = "Patient "+str(itm+1)+" goes in group "+str(block[itm])
        print(msg)
        f.write(msg+'\n')
    f.close()
    print("\nResults saved to file %s/block_randomization_output" %os.getcwd())
else:
    subjects_per_strata = []
    source_choice = raw_input("Get strata and strata subjects from Opal database? (y/N): ") or "n"
    if str(source_choice).lower() == 'y':
        for itm in opal_db_connect("strata"):
            subjects_per_strata.append(itm[0])
        print("\nSubjects for strata randomization in Opal db: %s\n" %subjects_per_strata)
    else:
        subjects_per_strata = input("Give comma separated list of the number of subjects for each strata:")

    #print(subjects_per_strata)
    group_number = input("Give number of groups to randomize subjects to:")
    block_length = input("Give length of the blocks (value of `block_length` should be a multiple of `groups` to ensure proper balance):")

    strat = randomization.stratification(subjects_per_strata,group_number,block_length)
    print("stratified block randomization results for %s \n %s\n" %(subjects_per_strata,strat))

    f = open('stratified_randomization_output', 'w')
    for strat_list in range(len(subjects_per_strata)):
        msg = "\nStrata "+str(strat_list+1)+" list"
        print(msg)
        f.write(msg+"\n")
        for itm in range(len(strat[strat_list])):
            msg = "\tPatient "+str(itm+1)+" goes in group "+str(strat[strat_list][itm])
            print(msg)
            f.write(msg+"\n")
    f.close()
    print("\nResults saved to file %s/stratified_randomization_output" %os.getcwd())
```

Για το λογισμικό που αναπτύχθηκε με το framework Opal βλέπουμε παρακάτω κάποιες εικόνες από τη χρήση του:

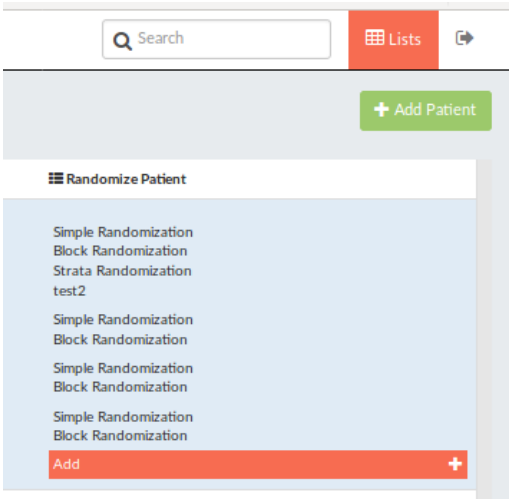


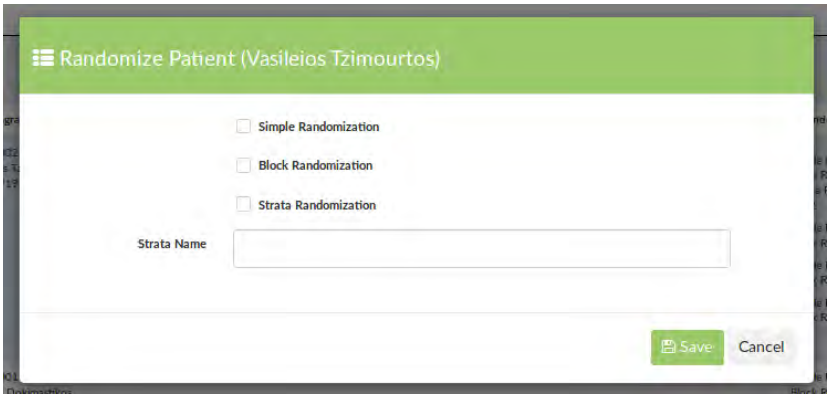


Στην αριστερή εικόνα βλέπουμε τη σελίδα σύνδεσης στην εφαρμογή ενώ στη δεξιά τις επιλογές που έχουμε μόλις συνδεθούμε. Στην Team Lists μπορούμε να προσθέσουμε ασθενείς από το κουμπί Add Patients πάνω δεξιά και φαίνεται στην παρακάτω εικόνα:



Το module που αναπτύχθηκε είναι μέρος του Team Lists και είναι η στήλη Randomize Patient στα στοιχεία κάθε ασθενή που έχει δημιουργηθεί.



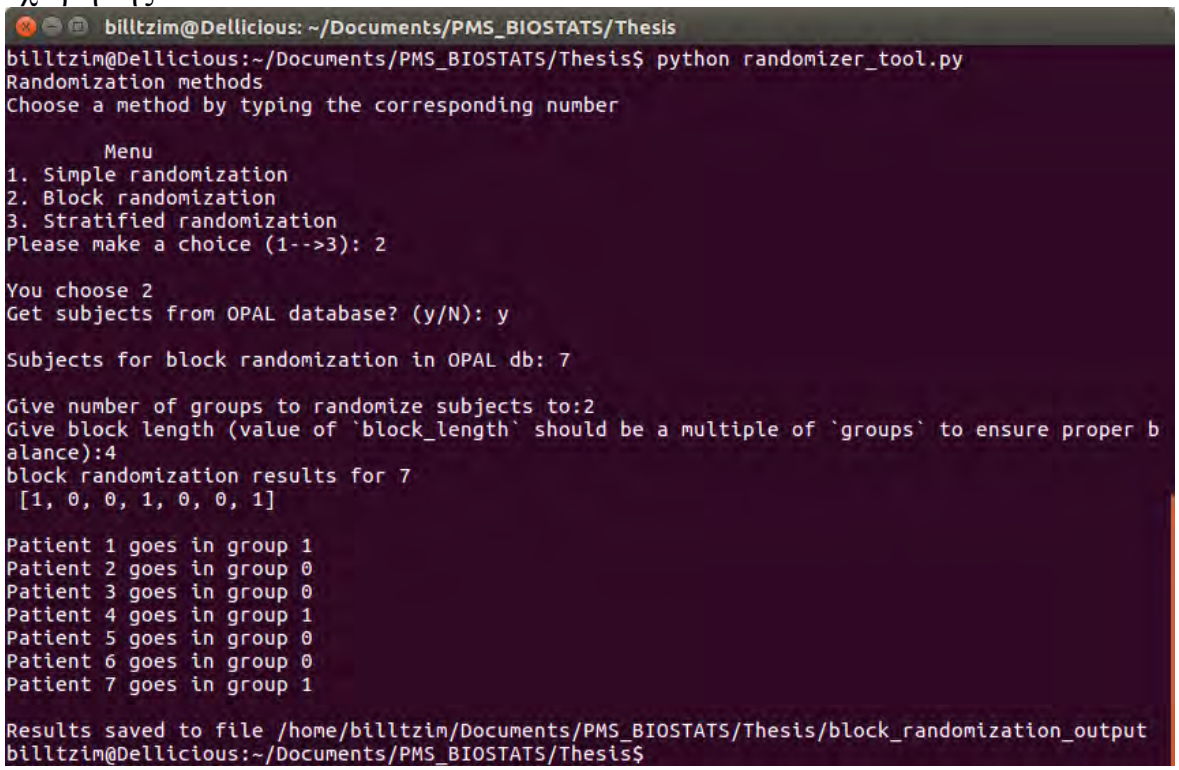


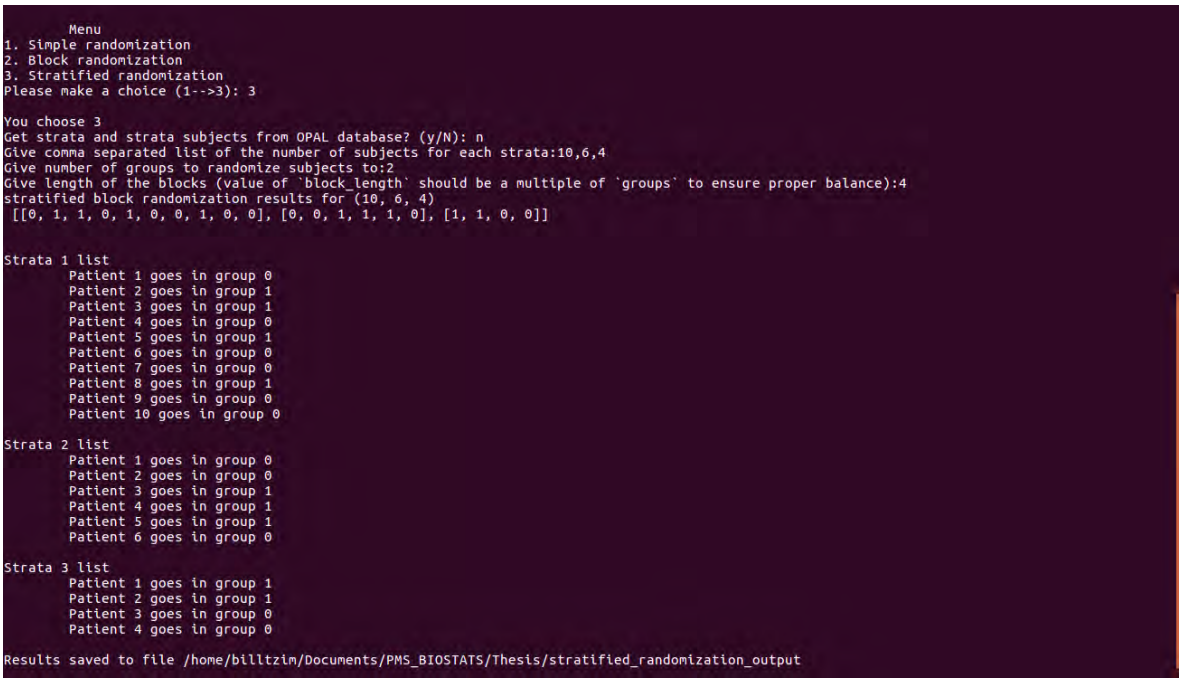
Κάνοντας κλικ στο Add μπορούμε να προσθέσουμε τη μέθοδο τυχαιοποίησης στην οποία θέλουμε να λάβει μέρος ο συγκεκριμένος ασθενής. Για τη μέθοδο της στρωματοποιημένης τυχαιοποίησης πρέπει επιπλέον να γράψουμε και το όνομα του strata στο οποίο θα ανήκει ο ασθενής. Με βάση αυτά τα στοιχεία το λογισμικό που αναπτύχθηκε έχει τη δυνατότητα να διαβάσει το πλήθος ασθενών για κάθε μέθοδο τυχαιοποίησης.
Εκτελώντας το λογισμικό που αναπτύχθηκε καλούμαστε να επιλέξουμε τη μέθοδο που θέλουμε να ακολουθήσουμε, όπως φαίνεται και στην παρακάτω εικόνα:

```
billtzim@Dellicious: ~/Documents/PMS_BIOSTATS/Thesis
billtzim@Dellicious:~/Documents/PMS_BIOSTATS/Thesis$ python randomizer_tool.py
Randomization methods
Choose a method by typing the corresponding number

        Menu
1. Simple randomization
2. Block randomization
3. Stratified randomization
Please make a choice (1-->3): 1

You choose 1
Get subjects from OPAL database? (y/N):
Give number of subjects to randomize:10
Give number of groups to randomize subjects to:2

Simple randomization results for 10
 [1, 2, 1, 2, 2, 2, 1, 1, 2, 1]

Patient 1 goes in group 1
Patient 2 goes in group 2
Patient 3 goes in group 1
Patient 4 goes in group 2
Patient 5 goes in group 2
Patient 6 goes in group 2
Patient 7 goes in group 1
Patient 8 goes in group 1
Patient 9 goes in group 2
Patient 10 goes in group 1

Results saved to file /home/billtzim/Documents/PMS_BIOSTATS/Thesis/simple_randomization_output
billtzim@Dellicious:~/Documents/PMS_BIOSTATS/Thesis$
```

Ανάλογα την επιλογή μεθόδου θα μας ζητηθεί αν θέλουμε να λάβει η εφαρμογή το πλήθος των ασθενών από τη βάση δεδομένων του Opal ή να τους δώσουμε χειροκίνητα καθώς επίσης και το πλήθος των ομάδων. Για τις περιπτώσεις των block και stratified

randomization θα μας ζητηθεί επιπλέον το μέγεθος του block που θέλουμε. Με την ολοκλήρωση της εκτέλεσης θα τυπωθούν στην οθόνη οι τοποθετήσεις των ασθενών στις ομάδες και τα αποτελέσματα θα αποθηκευτούν σε ένα αρχείο κειμένου με κατάλληλο όνομα. Το μονοπάτι του αρχείου τυπώνεται ενημερωτικά και στην οθόνη. Στην παρακάτω εικόνα βλέπουμε μια εκτέλεση για block randomization με λήψη στοιχείων από τη βάση δεδομένων και για stratification randomization με χειροκίνητη καταχώρησης:





4. Συμπέρασμα

Για τις ανάγκες αυτής της μεταπτυχιακής διατριβής δημιουργήθηκε ένα λογισμικό το οποίο μπορεί να χρησιμοποιηθεί στις περιπτώσεις που συνίσταται εφαρμογή μιας μεθόδου τυχαιοποίησης. Η ανάπτυξή του βασίστηκε σε λογισμικό, εργαλεία και τεχνολογίες που είναι ελεύθερα διαθέσιμα για χρήση. Η εφαρμογή γράφτηκε με τέτοιον τρόπο ώστε να μπορεί να εκτελεστεί κάνοντας χρήση του διερμηνευτή οποιασδήποτε έκδοσης της γλώσσας Python (2.7, 3.*<=3.6) με αποτέλεσμα να καθιστά το λογισμικό εύκολα μεταφέρσιμο και χρηστικό. Η επιπλέον δυνατότητα διασύνδεσης μέρους των παραμέτρων εισαγωγής μέσω μιας web γραφικής διεπαφής μπορούν να αναδείξουν καλύτερα τη χρήση του και να του προσδώσουν δυνατότητες επέκτασης καθιστώντας το ένα εργαλείο ικανό να καλύψει ανάγκες τόσο επαγγελματικές όσο και εκπαιδευτικές.

5. Αναφορές

1. Suresh K., An overview of randomization techniques: An unbiased assessment of outcome in clinical research, J Hum Reprod Sci. 2011 Jan;4(1):8-11. doi: 10.4103/0974-1208.82352.
2. Ντζούφρας Ι., Περπέρογλου Α., Εισαγωγή στην Βιοστατιστική και την Επιδημιολογία, Αθήνα, Σεπτέμβριος 2009, 4η Έκδοση, URL: http://www.actuar.aegean.gr/notes/biostatistics-v4-0.pdf
3. Python, URL: https://www.python.org/
4. Eclipse Integrated Development Environment, URL: https://eclipse.org/
5. PyDev, URL: http://www.pydev.org/
6. Pip, URL: https://packaging.python.org/key_projects/#pip
7. SciPy, URL: https://www.scipy.org/
8. SciKits, URL: http://scikits.appspot.com/
9. Randomization Python Package, URL: https://pypi.python.org/pypi/randomization/1.0.0
10. SciKit-Design Python Package, URL: https://github.com/louden/scikit-design
11. Opal, URL: http://opal.openhealthcare.org.uk/