FACULTY OF COMPUTING ENGINEERING AND THE BUILT ENVIRONMENT

MSc. Advanced Computer Science Research

An experiment on the effects of the anti-anxiety drug on memory recovery

Module Leader: Yevgeniya Kovalchuk

Module: Advanced Data Science
Code: CMP7161
Student: Filip Paskalev
№ S19155623

Academic year 2019/2020

Contents

List of Figures

List of Tables

Abstract

The report is over an experiment on the effects of anti-anxiety medicament on memory recall when being primed with happy or sad memories. The focus of the analysis is on the placebo effect and the effect of anti-anxiety medicine on memory recall. Data set is created from the data of participants in the experiment, done on novel Islanders who mimic real-life humans in response to external factors. The experiment was executed under the supervision of Mr. Almohalwas at UCLA. All aspects of the experiment such as experimental design, data collection, and preprocessing were done from Steve Ahn.

1 Introduction

1.1 Background

1.1.1 Building the Case

Anti-Anxiety Medicine Obstructive effects of Benzodiazepines:
Long term adverse effects on Long Term Potentiation of synapses, metacognition and memory recall ability http://www.jstor.org/stable/43854146

Happy Memories Research showed positive memories to have a deeper and greater volume of striatum representation under an
fMRI https://www.sciencedirect.com/science/article/pii/S0896627314008484

Sad Memories Research shown sad memories invoke better memory recall for evolutionary
purpose whereas, happy memories are more susceptible to false memories
http://www.jstor.org/stable/40064315

1.1.2 Participants

All genders above 25+ years old to ensure a fully developed pre-frontal cortex, a region responsible for higher-level cognition and memory recall.

1.2 Aim and objectives

1. General observations and analysis of effective Benzodiazepines?
2. Does the anti-anxiety drug work differently at different ages?
3. To establish the effect of placebo and its correlation with the subject.
    - If there is a placebo effect to analyze at what point it occurs.
4. Does anxiety medicine drugs have an effect on memory?
5. To analyze whether there are an influence and connection between the application of positive or negative emotions before taking the drug.

2 Data set

2.1 Description

Data containing Islander general, drug, and test information.

Columns:

first$_{n}ame$:	First name of Islander.
last$_{n}ame$:	Last name of Islander.
age:	Age of Islander.
Happy$_{S}ad_{g}roup$:	Happy or Sad Memory priming block.
Dosage:	1-3 to indicate the level of dosage (low - medium - over recommended daily intake).
Drug:	Type of Drug administered to Islander.
Mem$_{S}core_{B}efore$:	Seconds - how long it took to finish a memory test before drug exposure.
Mem$_{S}core_{A}fter$:	Seconds - how long it took to finish a memory test after addiction achieved.
Diff:	Seconds - difference between memory score before and after.

2.2 Preparing data

Loading libraries Importing all the libraries that will be needed for the project.

```
# math tools
import numpy as np
# import and manage data sets
import pandas as pd
# visualization & sub libarary with tool for visualization
import matplotlib.pyplot as plt

import sklearn
# for splitting the data set
from sklearn.model_selection import train_test_split
# label encoding the data
from sklearn.preprocessing import LabelEncoder
# importing one hot encoder from sklearn
from sklearn.preprocessing import OneHotEncoder

from sklearn.utils import shuffle
```

2.3 Sample

Index	first_nan	last_name	age	Happy_Sad_group	Dosage	Drug	Mem_Score_Before	Mem_Score_After	Diff
0	Bastian	Carrasco	25	H	1	A	63.5	61.2	-2.3
1	Evan	Carrasco	52	S	1	A	41.6	40.7	-0.9
2	Florenc…	Carrasco	29	H	1	A	59.7	55.1	-4.6
3	Holly	Carrasco	50	S	1	A	51.7	51.2	-0.5
4	Justin	Carrasco	52	H	1	A	47	47.1	0.1
5	Liam	Carrasco	37	S	1	A	66.4	58.1	-8.3
6	Ava	Durand	35	S	1	A	44.1	56	11.9
7	Jamie	Durand	38	H	1	A	76.3	74.8	-1.5
8	Josefa	Durand	29	H	1	A	56.2	45	-11…
9	Mark	Durand	36	S	1	A	54.8	75.9	21.1
10	Maximil…	Durand	63	S	1	A	90	102	12
11	Bastian	Gonzalez	27	S	1	A	52.3	63.7	11.4
12	Ayano	Kennedy	39	H	1	A	35.5	40.7	5.2
13	Grace	Kennedy	26	H	1	A	85.6	84.3	-1.3
14	Ai	Lopez	26	S	1	A	42.3	32.8	-9.5
15	Kaito	McCarthy	48	H	1	A	53.5	56.3	2.8

Figure 1: Raw data sample

2.4 Information

2.4.1 Field types

Check what type are the fields in the data.

```
source.info()
```

Result Visualization of the result.

```
RangeIndex: 198 entries, 0 to 197
Data columns (total 9 columns):
# Column Non-Null Count Dtype
--- ------ -------------- -----
0 first_name 198 non-null object
1 last_name 198 non-null object
2 age 198 non-null int64
3 Happy_Sad_group 198 non-null object
4 Dosage 198 non-null int64
5 Drug 198 non-null object
6 Mem_Score_Before 198 non-null float64
7 Mem_Score_After 198 non-null float64
8 Diff 198 non-null float64
dtypes: float64(3), int64(2), object(4)
memory usage: 14.0+ KB
```

2.4.2 NULL's

Check if data have any NULL values or missing values.

```
print(source.isnull().sum())
```

Result Visualization of the result.

```
first_name 0
last_name 0
age 0
Happy_Sad_group 0
Dosage 0
Drug 0
Mem_Score_Before 0
Mem_Score_After 0
Diff 0
dtype: int64
```

2.4.3 Describe

Brief summary data description.

```
print(source.describe())
```

Result Visualization of the result.

	age	Dosage	MSB	MSA	Diff
count	198.000000	198.000000	198.000000	198.000000	198.000000
mean	39.530303	1.989899	57.967677	60.922222	2.954545
std	12.023099	0.818504	15.766007	18.133851	10.754603
min	24.000000	1.000000	27.200000	27.100000	−40.400000
25%	30.000000	1.000000	46.525000	47.175000	−3.175000
50%	37.000000	2.000000	54.800000	56.750000	1.700000
75%	48.000000	3.000000	68.400000	73.250000	5.925000
max	83.000000	3.000000	110.000000	120.000000	49.000000

2.5 Analysis

Data don't contain any NULL values or any missing values. All the attributes will be usable. Field[1] "first_name" and "last_name" can be concatenate in one field - "name". Types of the field "age", "Dosage", "second_name", should be optimized in a smaller unit. "Happy_Sad_group" and "Drug" need to be grouped by type.

3 Pre-processing

3.1 Trim

For the purpose of the study, we do not need the fields - first name and second name. They will be combined into one, as shown in the following lines.

```
# Combine "first_name" and "last_name" field
in one field with label name
data["name"] = data["first_name"] + " " + data["last_name"]
# Remove column first_name
del data['first_name']
# Remove column last_name
del data['last_name']
```

3.2 Fiels drug

Label field "Drug" by labels:

- A (Alprazolam) = 0
- T (Triazolam) = 1
- S (Sugar Tablet - Placebo) = 2

The function takes and add a string for parameters. Later they will be converted in integers.

```
# Iterate over data and label all data with correct labels
data['Drug'] = data['Drug'].str.replace('A', '0', case = False)
data['Drug'] = data['Drug'].str.replace('T', '1', case = False)
data['Drug'] = data['Drug'].str.replace('S', '2', case = False)
```

3.3 Field Happy_Sad_group

In the same methodological field "Happy_Sad_group" will be labeled -

- H (Happy) = 0
- S (Sad) = 1

The field is in a string, later it will be converted in integers.

```
data['Happy_Sad_group'] =
    data['Happy_Sad_group'].str.replace('H', '0', case = False)
data['Happy_Sad_group'] =
    data['Happy_Sad_group'].str.replace('S', '1', case = False)
```

3.4 String fields

Localization fields Localization which fields need to be handled by casting to integers.

```
data.info()
```

Result Visualization of the result from the check.

```
RangeIndex: 198 entries, 0 to 197
Data columns (total 8 columns):
# Column Non-Null Count Dtype
--- ------ -------------- -----
0 age 198 non-null int64
1 Happy_Sad_group 198 non-null object
2 Dosage 198 non-null int64
3 Drug 198 non-null object
4 Mem_Score_Before 198 non-null float64
5 Mem_Score_After 198 non-null float64
6 Diff 198 non-null float64
7 name 198 non-null object
dtypes: float64(3), int64(2), object(3)
```

3.4.1 Casting

Converting the fields.

```
data['Drug'] = data['Drug'].astype(int)
data['Happy_Sad_group'] = data['Happy_Sad_group'].astype(int)
```

Result Visualization of the result from casting.

```
RangeIndex: 198 entries, 0 to 197
Data columns (total 8 columns):
# Column Non-Null Count Dtype
--- ------ -------------- -----
0 age 198 non-null int64
1 Happy_Sad_group 198 non-null int32
2 Dosage 198 non-null int64
3 Drug 198 non-null int32
4 Mem_Score_Before 198 non-null float64
5 Mem_Score_After 198 non-null float64
6 Diff 198 non-null float64
7 name 198 non-null object
dtypes: float64(3), int32(2), int64(2), object(1)
memory usage: 11.0+ KB
```

3.5 Field age_group

Purpose Create an additional field that will be used for the purpose of analysis in the study. The group will be separated on four[1] -

- child (under 15) = 0
- young adult (16 - 30) = 1
- middle-aged adult (31 - 50) = 2
- senior adult (over 50) = 3

Code Preview of the steps for creating the new field.

```
# Create obj age_group and fill it with
categorizated data from column age
age_group = []
# Grouping the ages 0-15, 16-30, 30-50 , 51-max
for i in data.itertuples():
index = 0
if i[1] <= 15:
age_group.append(0)
elif i[1] <= 30:
age_group.append(1)
elif i[1] <= 50:
age_group.append(2)
else:
age_group.append(3)
index = index + 1
# Add columnto the data
data['age_group'] = age_group
```

Result Visualization of the result.

```
angeIndex: 198 entries, 0 to 197
Data columns (total 9 columns):
# Column Non-Null Count Dtype
--- ------ -------------- -----
0 age 198 non-null int64
1 Happy_Sad_group 198 non-null int32
2 Dosage 198 non-null int64
3 Drug 198 non-null int32
4 Mem_Score_Before 198 non-null float64
5 Mem_Score_After 198 non-null float64
6 Diff 198 non-null float64
7 name 198 non-null object
8 age_group 198 non-null int64
dtypes: float64(3), int32(2), int64(3), object(1)
memory usage: 12.5+ KB
```

3.6 Encoder

3.6.1 Field name

Creating an object of type LabelEncoder() which will encode all names in the field "name".

```
label_encoder = LabelEncoder()
data['name']= label_encoder.fit_transform(data['name'])
```

3.6.2 Rest of the fields

Using objects from the previous paragraph, and use the same methodology to encode the rest of the fields. The fields are already in a numerical category, but if some of the categories missing (Example "ages_group" - the data don't contain group 0 - children) recategorization is possible for optimization. This will help for better automatization of the code structure in the future.

```
data['age_group']= label_encoder.fit_transform(data['age_group'])
data['Drug']= label_encoder.fit_transform(data['Drug'])
data['Happy_Sad_group']= label_encoder.fit_transform(data['Happy_Sad_group'
```

3.7 Optimization

3.7.1 int64 to int32

Optimize the size of the stored types in the field. This step is not necessary, but it will be a helpful feature if the training model data grow.

```
data['Dosage'] = data['Dosage'].astype(int)
data['age'] = data['age'].astype(int)
data['age_group'] = data['age_group'].astype(int)
```

3.7.2 Cleaning unused objects

Clean all the objects that have been used to now for better optimization and cleaner code. Variable "i" is used temporarily, and it doesn't need to be store. Object "age_group is stored in object "data", and it doesn't need to have a copy of it. Object "label_encoder", has finished its purpose, it doesn't need to be saved.

```
del age_group
del i
del label_encoder
```

4 Simple Linear Regression

Building the Simple linear regression model, based on data set.

4.1 Import

Importing needable libraries

```
import numpy as np
import pandas as pd
import sklearn
from sklearn import linear_model
import pickle
import matplotlib.pyplot as plt
```

4.2 Load

Loading the data. Shuffling the data is optional. Not every fields are inserted in to the data, fields are inserted base on what will be investigated and the relation between them and target.

```
data = pd.read_csv('data.csv')
from sklearn.utils import shuffle
data = shuffle(data)
data = data[[
"Happy_Sad_group",
"Dosage",
"Drug",
"Mem_Score_Before",
"Mem_Score_After",
"age_group"]]
```

4.3 Prediction

Targeting the prediction field.

```
predict = "Mem_Score_After"
x = np.array(data.drop([predict], 1))
y = np.array(data[predict])
```

4.4 Split

Preraring data to train and split data lfor tests.

```
x_train, x_test, y_train, y_test =
    sklearn.model_selection.train_test_split(
        x, y, test_size = 0.1, random_state = 0)
```

4.5 Train

Train the model. Pickle module is ised to save the model.

```
best = 0
for _ in range(10000):
x_train, x_test, y_train, y_test =
    sklearn.model_selection.train_test_split(x, y, test_size=0.1)
linear = linear_model.LinearRegression()
linear.fit(x_train, y_train)
acc = linear.score(x_test, y_test)
print(acc)
if acc > best:
best = acc
with open("simple_linear_regression.pickle", "wb") as f:
pickle.dump(linear, f)
pickle_in = open("simple_linear_regression.pickle", "rb")
linear = pickle.load(pickle_in)
predicted = linear.predict(x_test)
for x in range(len(predicted)):
print(predicted[x], x_test[x], y_test[x])
```

5 Random Forest Regression

Building the Random Forest Regression model, based on data set.

5.1 Build

Importing needable libraries

```
import numpy as np
import pandas as pd
import sklearn as sklearn
from sklearn import linear_model
from sklearn.ensemble import RandomForestRegressor
from sklearn.utils import shuffle
import pickle
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import r2_score
from sklearn.ensemble import RandomForestRegressor
```

5.2 Load

Loading the data. On the same methodological like on linear regression data is load.

```
data = pd.read_csv('data.csv')
data = shuffle(data)
data = data[[
"age",
"Happy_Sad_group",
"Dosage",
"Drug",
"Mem_Score_Before",
"Mem_Score_After",
"Diff",
"name",
"age_group"
]]
predict = "Mem_Score_After"
x = np.array(data.drop([predict], 1))
y = np.array(data[predict])
```

5.3 Split

Split the data on train sets and save 10% from it for tests.

```
x_train, x_test, y_train, y_test =
    sklearn.model_selection.train_test_split(
        x, y, test_size = 0.1, random_state = 0)
```

5.4 Train

Train the model. Pickle module is used to save the model.

```
best = 0
for _ in range(10):
x_train, x_test, y_train, y_test =
    sklearn.model_selection.train_test_split(
        x, y, test_size=0.1)

forest = RandomForestRegressor()

forest.fit(x_train, y_train)
acc = forest.score(x_test, y_test)
print(acc)

if acc > best:
best = acc
with open("random_forest_regression.pickle", "wb") as f:
pickle.dump(forest, f)
```

6 Explore

6.1 Correlation

Correlation between features and target.

```
correlation = data.corr()
plt.figure(figsize=(10,8))
heatmap = sns.heatmap(correlation ,
                       annot=True ,
                       linewidths=1,
                       vmin=1)
```

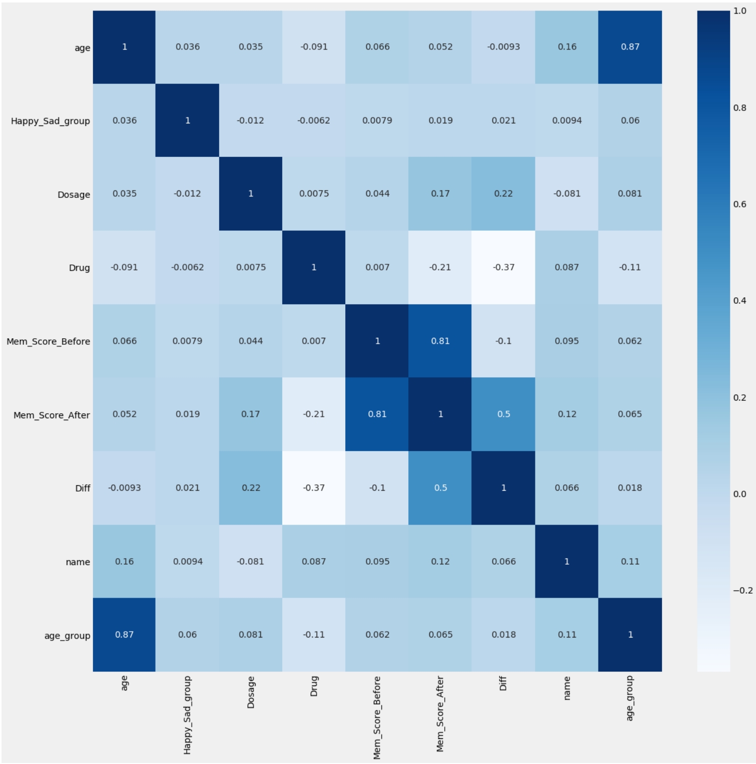

Figure 2: Finding correlation visualization

6.2 Memory score after medicament

General view of the memory score of the subjects after effect of drugs.

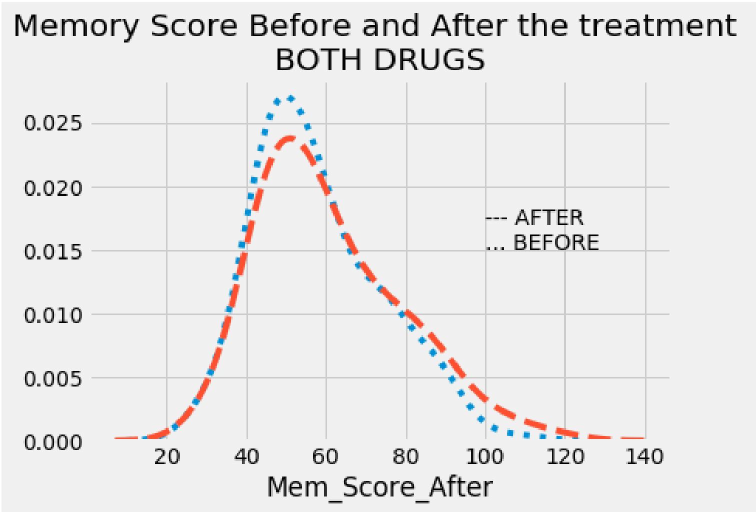

Figure 3: Memory score after drug

6.3 Dosage effect

Visualization memory score based on the effect of the dosage. Drugs are represented in 3 groups.

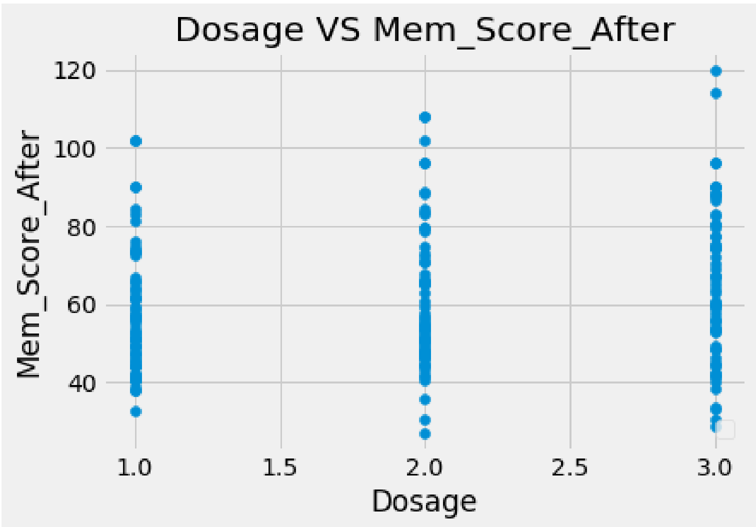

Figure 4: Effect of dosage

6.4 Age analysis

Visual representing of age groups of tested subjects.

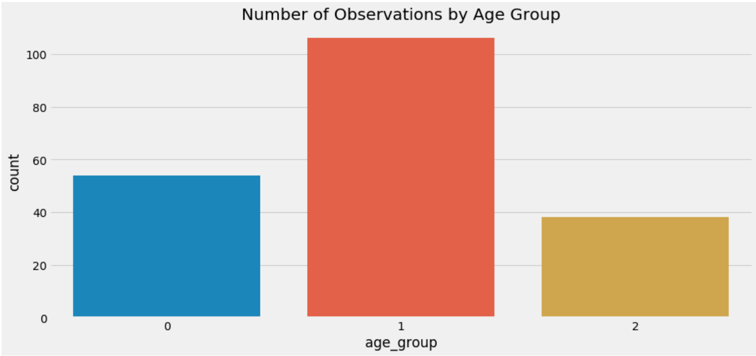

Figure 5: Age groups

Visual of age groups of tested subjects.

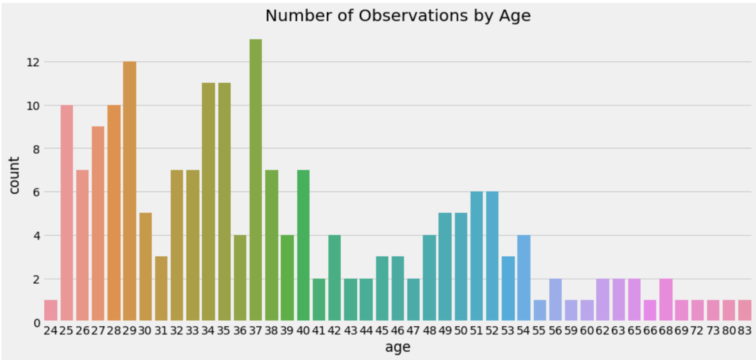

Figure 6: Age stats

7 References

7.1 Glossary

1 https://www.facebook.com/WebMD (2007). Benzodiazepine Abuse. [online] WebMD. Available at: https://www.webmd.com/mental-health/addiction/benzodiazepine-abuse#1.

2 Wikipedia Contributors (2019). Benzodiazepine. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Benzodiazepine.

7.2 Data set

1 Wikipedia. (2020). Column (database).
online
Available at: https://en.wikipedia.org/wiki/Column_(database) [Accessed 11 May 2020].

7.3 Pre-processing

1 Yarlagadda, A., Murthy, J.V.R. and Krishna Prasad, M.H.M. (2015). A novel method for human age group classification based on Correlation Fractal Dimension of facial edges. Journal of King Saud University - Computer and Information Sciences, [online] 27(4), pp.468–476. Available at: https://www.sciencedirect.com/science/article/pii/S1319157815000671 [Accessed 7 Jul. 2019].