Machine Learning 2020-01 Assignment #2

Predicting the Ecstasy Consumption Level through Feature Engineering

Chaeyeon Han 2015147508

1. Introduction

The aim of this project is to predict one's consumption level of ecstasy through machine learning and improve its accuracy through feature engineering. Each row of the original dataset[1] contains 30 attributes including:

- five demographic information (age, gender, education, country in residence, ethnicity),
- seven personality related measurements (neuroticism, extraversion, openness to experience, agreeableness, conscientiousness, impulsiveness, sensation seeking),
- consumption level of 18 types of legal or illegal drugs (alcohol, amphetamines, amyl nitrite, benzodiazepine, cannabis, chocolate, cocaine, caffeine, crack, ecstasy, heroin, ketamine, legal highs, LSD, methadone, mushrooms, nicotine and semeron).

Suggesting the consumption level of 'ecstasy' as a dependent variable, all other attributes are considered independent variables.

2. Dataset Analysis

2.1 Meaning and Distribution of Attributes

The original dataset was given as regularized values as seen in the tables below. The

[1] Dataset Link, https://archive.ics.uci.edu/ml/datasets/Drug+consumption+%28quantified%29

visualization of the correlation table is made through Python3 and Jupyter Notebook. Among the 1885 respondents total:

① Demographic Distribution (Age, Gender, Education, Country, Ethnicity)

Value	Age	Cases	Fraction
−0.95197	18–24	643	34.11%
−0.07854	25–34	481	25.52%
0.49788	35–44	356	18.89%
1.09449	45–54	294	15.60%
1.82213	55–64	93	4.93%
2.59171	65+	18	0.95%

Value	Gender	Cases	Fraction
0.48246	Female	942	49.97%
−0.48246	Male	943	50.03%

Value	Education	Cases	Fraction
−243591	Left school before 16yrs old	28	1.49%
−1.7290	Left school at 16	99	5.25%
−1.43719	Left school at 17	30	1.59%
−1.22751	Left school at 18	100	5.31%
−0.61113	Some college, uni – no degree	506	26.84%
−0.05921	Professional degree	270	14.32%
0.45468	University degree	480	25.46%
1.16365	Master's degree	283	15.01%
1.98437	Doctorate degree	89	4.72%

Value	Country	Cases	Fraction
−0.09765	Australia	54	2.86%
0.24923	Canada	87	4.62%
−0.46841	New Zealand	5	0.27%

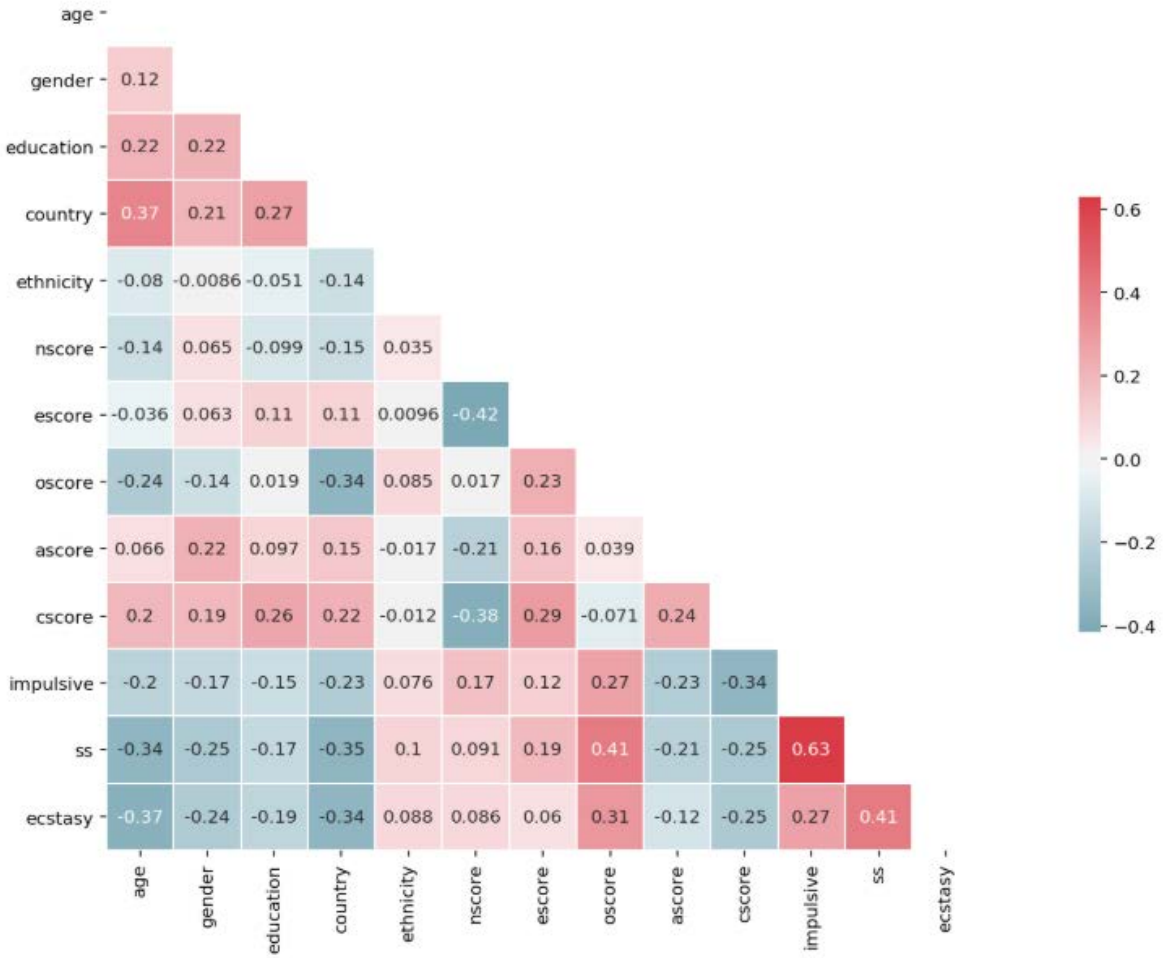
−0.28519	Other	118	6.26%
0.21128	Republic of Ireland	20	1.06%
0.96082	UK	1044	55.38%
−0.57009	USA	557	29.55%

Value	Ethnicity	Cases	Fraction
−0.50212	Asian	26	1.38%
−1.10702	Black	33	1.75%
1.90725	Mixed−Black/Asian	3	0.15%
0.12600	Mixed−White/Asian	20	1.06%
−0.22166	Mixed−White/Black	20	1.06%
0.11440	Other	63	3.34%
−0.31685	White	1720	91.25%

② Personality Measurements and Correlation

The personality measurements provided are from NEO−FFI−R, BIS−11, and ImpSS. Five attributes from NEO−FFI−R are neuroticism, extraversion, openness to experience, agreeableness, and conscientiousness. Each are labeled as ‘nscore’, ‘escore’, ‘oscore’, ‘ascore’, and ‘cscore’. Neuroticism refers to being moody, and vulnerable to feeling anxiety, worry, frustration, guilt and other non−positive moods. Extraversion refers to being social and outgoing. Openness to experience indicates an open−mindedness of a person. Conscientiousness refers to being careful and diligent. Impulsivity from BIS−11 indicates acting without forethought or consideration. SS from ImpSS refers to actively searching new experiences and feelings. The table below shows the correlation between ecstasy consumption and personality traits.

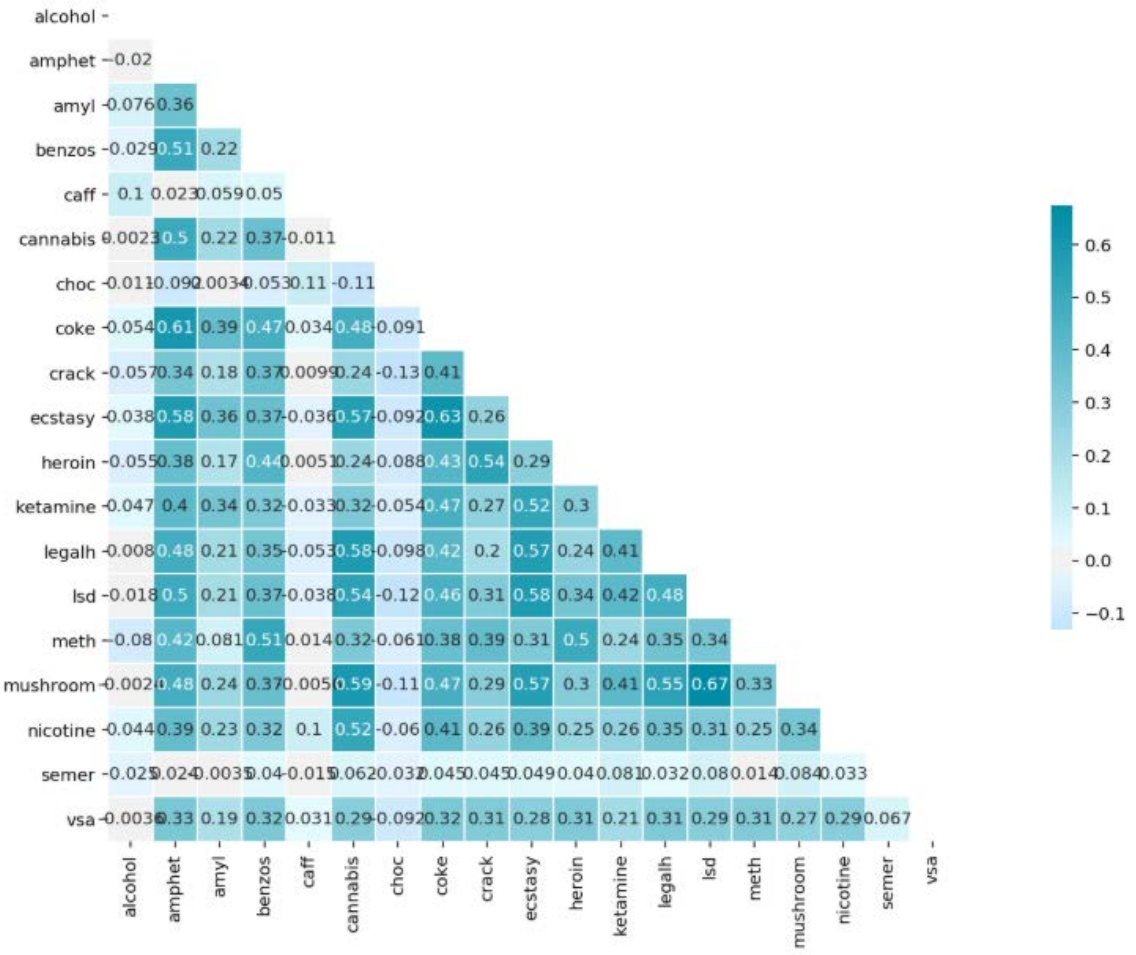
〈Correlation Table 2 – Ecstasy and Personality Trait〉



③ Drug Consumption Levels and Correlation

Drug consumption level is categorized into seven levels – CL0: never used, CL1: Used over a Decade Ago, CL2: Used in Last Decade, CL3: Used in Last Year, CL4: Used in Last Month, CL5: Used in Last Week, CL6: Used in Last Day. Below is a correlation table between each drug consumption level.

〈 Correlation Table 〉



2.2 Why Ecstasy?

Considering the correlation between each drug consumption level, ecstasy has shown most meaningful p-value. Therefore, I assumed that meaningful feature engineering would be possible through combining and elaborating existing features.

3. Overall Approach of Feature Engineering

① Sum of consumption level values of the most related drugs

Secondly, I created a new feature 'drug_sum', which contains a value that is a summation of the consumption levels of each strongly correlated drug.

② Number of consumed drugs among the most related drugs

According to the correlation table, the consumption level of an ecstasy is most related with the consumption level of amphetamine, cannabis, coke, ketamine, legalh, lsd, and mushroom. In order to give some weight to those drugs, I created a new feature named 'drug_count', which contains a value of the number of strongly related drugs that a person has consumed at least once in his life.

③ Correlated Personality traits – Openness to experience, Sensation Seeking

Lastly, I created a new feature named 'demo_count' which contains a value of openness to experience and sensation seeking. The two factors were chosen because they show high correlation with ecstasy consumption as seen in the table above.

4. Source Code (with detailed comments)

I used 'scikit-learn' and Python 3.

〈 Modeling with Original Dataset〉

- 'drug_consumption_numeric.csv' file is a manually modified data. I have transformed the original dataset which was '.data' format into CSV file and changed 'CL0'~'CL6' into numerical values 0~6 in order to use them as proper machine learning features.
- In order to use 'ecstasy' column as a y value column, I changed the order of the columns. Next, I used XGboost algorithm to predict ecstasy. The hyperparameter 'objective' is set to 'multi:softprob' since the prediction value is multiclass. Lastly, I used classification report metrics to print f1 score of the model.

```
import numpy as np
import pandas as pd
import csv
import numpy as np
import random
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
import xgboost as xgb
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns

# saved the original .data file in csv format and imported
input_file = "drug_consumption_numeric.csv"

df = pd.read_csv(input_file, header = 0)
original_headers = list(df.columns.values)

# dragging 'ecstasy' column to the end in order to seperate X and y
col = ['age', 'gender', 'education', 'country', 'ethnicity', 'nscore',
       'escore', 'oscore', 'ascore', 'cscore', 'impulsive', 'ss', 'alcohol',
       'amphet', 'amyl', 'benzos', 'caff', 'cannabis', 'choc', 'coke', 'crack',
       'heroin', 'ketamine', 'legalh', 'lsd', 'meth', 'mushroom',
       'nicotine', 'semer', 'vsa','ecstasy']

df = df[col]

print("< Prediction with Original Dataset >₩n")

# defining X and y by slicing
X = df.iloc[:,:-1]
y = df.iloc[:, -1]

# split train set and test set 7:3
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.3, random_state = 0)

# set xgboost model and run
xgb_model = xgb.XGBClassifier(num_class=7, objective="multi:softprob")

xgb_model.fit(X_train, y_train)

y_pred = xgb_model.predict(X_test)

# evaluation metrics - original (no feature engineering)
print(classification_report(y_test,y_pred))
```

〈 Feature Engineering – 1〉

- As mentioned above in the overall approach, I added every consumption level value of highly correlated drugs.

```
# 1: amphet, cannabis, coke, ketamine, legalh, lsd, mushroom has the most correlation. so I gave weight to them.

print("< Feature Engineering - 1: Using the Sum of Important Drug consumptions >₩n")

col2 = ['age', 'gender', 'education', 'country', 'ethnicity', 'nscore',
       'escore', 'oscore', 'ascore', 'cscore', 'impulsive', 'ss', 'alcohol',
       'amphet', 'amyl', 'benzos', 'caff', 'cannabis', 'choc', 'coke', 'crack',
       'heroin', 'ketamine', 'legalh', 'lsd', 'meth', 'mushroom',
       'nicotine', 'semer', 'vsa','ecstasy']

# first, make a copy of the original dataset
df = df[col2]
df2 = df.copy()

# defined a list to save new column data
drug_sum = []

# for each respondent, if they have consumed most correlated drugs at least once in life (not 0),
# sum up the level of consumption
for i in range(0,len(df2)):
    count = 0
    if df2.loc[[i],['amphet']].values[0] != [0]:
        count += df2.loc[[i],['amphet']].values[0][0]
    if df2.loc[[i],['cannabis']].values[0] != [0]:
        count += df2.loc[[i],['cannabis']].values[0][0]
    if df2.loc[[i],['coke']].values[0] != [0]:
        count += df2.loc[[i],['coke']].values[0][0]
    if df2.loc[[i],['ketamine']].values[0] != [0]:
        count += df2.loc[[i],['ketamine']].values[0][0]
    if df2.loc[[i],['legalh']].values[0] != [0]:
        count += df2.loc[[i],['legalh']].values[0][0]
    if df2.loc[[i],['lsd']].values[0] != [0]:
        count += df2.loc[[i],['lsd']].values[0][0]
    if df2.loc[[i],['mushroom']].values[0] != [0]:
        count += df2.loc[[i],['mushroom']].values[0][0]

    drug_sum.append(count)

df2['drug_sum'] = drug_sum

new_col = ['age', 'gender', 'education', 'country', 'ethnicity', 'nscore',
       'escore', 'oscore', 'ascore', 'cscore', 'impulsive', 'ss', 'alcohol',
       'amphet', 'amyl', 'benzos', 'caff', 'cannabis', 'choc', 'coke', 'crack',
       'heroin', 'ketamine', 'legalh', 'lsd', 'meth', 'mushroom',
       'nicotine', 'semer', 'vsa','drug_sum','ecstasy']

df2 = df2[new_col]

# defining X and y by slicing
X = df2.iloc[:,:-1]
y = df2.iloc[:, -1]

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.3, random_state = 0)

xgb_model = xgb.XGBClassifier()
xgb_model.fit(X_train, y_train)
y_pred = xgb_model.predict(X_test)

# evaluation metrics
print(classification_report(y_test,y_pred))
```

〈 Feature Engineering – 2〉

- I counted the number of highly related drugs that the person has ever consumed.

```
# 2: amphet, cannabis, coke, ketamine, legalh, lsd, mushroom has the most correlation. so I gave weight to them.

print("< Feature Engineering - 2: Counting Important Drugs >₩n")

df3 = df2.copy()

drug_count = []

# for each respondent, if they have consumed most correlated drugs at least once in life (not 0) count as 1
for i in range(0,len(df3)):
    count = 0
    if df2.loc[[i],['amphet']].values[0] != [0]:
        count += 1
    if df2.loc[[i],['cannabis']].values[0] != [0]:
        count += 1
    if df2.loc[[i],['coke']].values[0] != [0]:
        count += 1
    if df2.loc[[i],['ketamine']].values[0] != [0]:
        count += 1
    if df2.loc[[i],['legalh']].values[0] != [0]:
        count += 1
    if df2.loc[[i],['lsd']].values[0] != [0]:
        count += 1
    if df2.loc[[i],['mushroom']].values[0] != [0]:
        count += 1

    drug_count.append(count)

df3['drug_count'] = drug_count

new_col = ['age', 'gender', 'education', 'country', 'ethnicity', 'nscore',
       'escore', 'oscore', 'ascore', 'cscore', 'impulsive', 'ss', 'alcohol',
       'amphet', 'amyl', 'benzos', 'caff', 'cannabis', 'choc', 'coke', 'crack',
       'heroin', 'ketamine', 'legalh', 'lsd', 'meth', 'mushroom',
       'nicotine', 'semer', 'vsa','drug_sum','drug_count','ecstasy']

df3 = df3[new_col]

# defining X and y by slicing
X = df3.iloc[:,:-1]
y = df3.iloc[:, -1]

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.3, random_state = 0)

xgb_model = xgb.XGBClassifier()
xgb_model.fit(X_train, y_train)
y_pred = xgb_model.predict(X_test)

# evaluation metrics
print(classification_report(y_test,y_pred))
```

〈 Feature Engineering – 3〉

– I added highly related personality traits that are under certain limits.

```
# 3: oscore, ss --> personality

print("< Feature Engineering - 3: Using Personality Traits >₩n")

df4 = df3.copy()

demo_count = []

for i in range(0,len(df4)):
    count = 0
    if df4.loc[[i],['oscore']].values[0][0] < 0:
        count += 1
    if df4.loc[[i],['ss']].values[0][0] < 0.7:
        count += 1

    demo_count.append(count)

df4['demo_count'] = demo_count

new_col = ['age', 'gender', 'education', 'country', 'ethnicity', 'nscore',
       'escore', 'oscore', 'ascore', 'cscore', 'impulsive', 'ss', 'alcohol',
       'amphet', 'amyl', 'benzos', 'caff', 'cannabis', 'choc', 'coke', 'crack',
       'heroin', 'ketamine', 'legalh', 'lsd', 'meth', 'mushroom',
       'nicotine', 'semer', 'vsa','drug_sum','drug_count','demo_count', 'ecstasy']

df4 = df4[new_col]

X = df4.iloc[:,:-1]
y = df4.iloc[:, -1]

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.3, random_state = 0)

xgb_model = xgb.XGBClassifier()
xgb_model.fit(X_train, y_train)
y_pred = xgb_model.predict(X_test)

# evaluation metrics
print(classification_report(y_test,y_pred))
```

5. Execution Results

```
< Prediction with Original Dataset >

              precision    recall  f1-score   support

           0       0.80      0.88      0.84       316
           1       0.46      0.37      0.41        30
           2       0.41      0.36      0.38        69
           3       0.41      0.39      0.40        95
           4       0.38      0.27      0.31        45
           5       0.12      0.12      0.12         8
           6       0.00      0.00      0.00         3

    accuracy                           0.64       566
   macro avg       0.37      0.34      0.35       566
weighted avg       0.62      0.64      0.63       566
```

< Feature Engineering - 1: Using the Sum of Important Drug consumptions >

	precision	recall	f1-score	support
0	0.81	0.91	0.85	316
1	0.56	0.50	0.53	30
2	0.38	0.29	0.33	69
3	0.42	0.40	0.41	95
4	0.39	0.27	0.32	45
5	0.12	0.12	0.12	8
6	0.00	0.00	0.00	3
accuracy			0.66	566
macro avg	0.38	0.36	0.37	566
weighted avg	0.63	0.66	0.64	566

< Feature Engineering - 2: Counting Important Drugs >

	precision	recall	f1-score	support
0	0.82	0.90	0.86	316
1	0.50	0.43	0.46	30
2	0.45	0.42	0.43	69
3	0.45	0.42	0.44	95
4	0.32	0.22	0.26	45
5	0.11	0.12	0.12	8
6	0.00	0.00	0.00	3
accuracy			0.67	566
macro avg	0.38	0.36	0.37	566
weighted avg	0.64	0.67	0.65	566

< Feature Engineering - 3: Using Personality Traits >

	precision	recall	f1-score	support
0	0.83	0.90	0.86	316
1	0.54	0.50	0.52	30
2	0.49	0.41	0.44	69
3	0.46	0.47	0.47	95
4	0.43	0.27	0.33	45
5	0.09	0.12	0.11	8
6	0.00	0.00	0.00	3
accuracy			0.68	566
macro avg	0.41	0.38	0.39	566
weighted avg	0.66	0.68	0.67	566

From the classification report metrics, I could see meaningful improvements in f1-score and overall accuracy. Accuracy has been increased from 0.64 to 0.68. f1-score has increased from [0.84, 0.41, 0.38, 0.40, 0.31, 0.12, 0.00] to [0.86, 0.52, 0.44, 0.47, 0.33, 0.11, 0.00].

Reference

1) Xgboost parameter, https://statkclee.github.io/model/model-python-xgboost-hyper.html