PYTHON PROJECT: DRUG CONSUMPTION

Emine BOUCHIBA

Hasna MANNAI

Yassine CHENIK

Problems :

- What characterizes a drug consumer ?
- What drugs and substances are the most correlated ?
- Can we predict a future drug consumer using ML ?
- How to prevent the rise of new consumers ?

Summary :

- Data Preprocessing
- Data Visualization
- Machine Learning Model
- API
- Conclusion

Data Preprocessing : Webscraping

- **Dataset before preprocessing :**

	1	0.49788	0.48246	-0.05921	0.96082	0.12600	0.31287	-0.57545	-0.58331	-0.91699	...	CL0.4	CL0.5	CL0.6	CL0.7	CL0.8	CL0.9	CL0.10	CL2.2
0	2	-0.07854	-0.48246	1.98437	0.96082	-0.31685	-0.67825	1.93886	1.43533	0.76096	...	CL4	CL0	CL2	CL0	CL2	CL3	CL0	CL4
1	3	0.49788	-0.48246	-0.05921	0.96082	-0.31685	-0.46725	0.80523	-0.84732	-1.62090	...	CL0	CL0	CL0	CL0	CL0	CL0	CL1	CL0
2	4	-0.95197	0.48246	1.16365	0.96082	-0.31685	-0.14882	-0.80615	-0.01928	0.59042	...	CL0	CL0	CL2	CL0	CL0	CL0	CL0	CL2
3	5	0.49788	0.48246	1.98437	0.96082	-0.31685	0.73545	-1.63340	-0.45174	-0.30172	...	CL1	CL0	CL0	CL1	CL0	CL0	CL2	CL2
4	6	2.59171	0.48246	-1.22751	0.24923	-0.31685	-0.67825	-0.30033	-1.55521	2.03972	...	CL0	CL0	CL0	CL0	CL0	CL0	CL0	CL6
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1879	1884	-0.95197	0.48246	-0.61113	-0.57009	-0.31685	-1.19430	1.74091	1.88511	0.76096	...	CL0	CL0	CL0	CL3	CL3	CL0	CL0	CL0
1880	1885	-0.95197	-0.48246	-0.61113	-0.57009	-0.31685	-0.24649	1.74091	0.58331	0.76096	...	CL2	CL0	CL0	CL3	CL5	CL4	CL4	CL5
1881	1886	-0.07854	0.48246	0.45468	-0.57009	-0.31685	1.13281	-1.37639	-1.27553	-1.77200	...	CL4	CL0	CL2	CL0	CL2	CL0	CL2	CL6
1882	1887	-0.95197	0.48246	-0.61113	-0.57009	-0.31685	0.91093	-1.92173	0.29338	-1.62090	...	CL3	CL0	CL0	CL3	CL3	CL0	CL3	CL4
1883	1888	-0.95197	-0.48246	-0.61113	0.21128	-0.31685	-0.46725	2.12700	1.65653	1.11406	...	CL3	CL0	CL0	CL3	CL3	CL0	CL3	CL6

1884 rows × 32 columns

Data Preprocessing : Webscraping

- **Scraping of "values" and "meanings" on the website of the dataset**

2. Age (Real) is age of participant and has one of the values:

Value	Meaning	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%

- **Creating "value to meaning" dataframes using BeautifulSoup and RegExp**

	age	Meanings
0	-0.95197	18-24
1	-0.07854	25-34
2	0.49788	35-44
3	1.09449	45-54
4	1.82213	55-64
5	2.59171	65+

	vsa	Meanings
0	CL0	Never Used
1	CL1	Used over a Decade Ago
2	CL2	Used in Last Decade
3	CL3	Used in Last Year
4	CL4	Used in Last Month
5	CL5	Used in Last Week
6	CL6	Used in Last Day

Data Preprocessing : Cleaning

- **Renaming columns :**

```
df.rename(columns = {'1':'ID',
                     '0.49788':'age' ,
                     '0.48246':'gender',
                     '-0.05921':'education',
                     '0.96082':'country',
                     '0.12600':'ethnicity',
                     '0.31287':'nscore',
                     '-0.57545':'escore',
                     '-0.58331':'oscore',
                     '-0.91699':'ascore',
                     '-0.00665':'cscore',
                     '-0.21712':'impulsive',
                     '-1.18084':'ss',
                     'CL5':'alcohol',
                     'CL2':'amphet',
                     'CL0':'amyl',
                     'CL2.1':'benzos',
                     'CL6':'caff',
                     'CL0.1':'cannabis',
                     'CL5.1':'choc',
                     'CL0.2':'coke',
                     'CL0.3':'crack',
                     'CL0.4':'ecstasy',
                     'CL0.5':'heroin',
                     'CL0.6':'ketamine',
                     'CL0.7':'legalh',
                     'CL0.8':'lsd',
                     'CL0.9':'meth',
```

- **Using ID as the index :**

```
first_line = list(df.columns)
dic = {}
for k in range(len(first_line)):
    dic[first_line[k]] = first_line[k]
new_row = pd.DataFrame(dic, index=[0])
df = pd.concat([new_row,df.loc[:]]).reset_index(drop=True)
```

Data Preprocessing : Cleaning

- **Delete all NaN variables :**

```
df=df.dropna(axis=0)
```

- **Modifying drug use columns to integers :**

```
temp = df.columns
temp = temp[13:]
for i in temp:
    df[i] = df[i].map({'CL0':0,'CL1':1,'CL2':2,'CL3':3,'CL4':4,'CL5':5,'CL6':6})
```

Data Preprocessing : Cleaning

- Creating " values " and a " meanings " dataframes :

```
df_Meanings_Bis=df.copy(deep=True)
df_Values=df_Bis.copy(deep=True)
```

- Deleting NaN values for each :

```
df_Values=df_Values.dropna(axis=0)
df_Meanings_Bis=df_Meanings.dropna(axis=0)
```

Data Preprocessing : Cleaning

- **Converting the values to its corresponding "meanings" :**

```
df["age"].iloc[0]=AGE["Meanings"].loc[2]
Convert_To_Meanings(df,AGE,"age","Meanings")
df["gender"].iloc[0]=GENDER["Meanings"].loc[0]
Convert_To_Meanings(df,GENDER,"gender","Meanings")
df["education"].iloc[0]=EDUCATION["Meanings"].loc[5]
Convert_To_Meanings(df,EDUCATION,"education","Meanings")
df["country"].iloc[0]=COUNTRY["Meanings"].loc[5]
Convert_To_Meanings(df,COUNTRY,"country","Meanings")
df["ethnicity"].iloc[0]=ETHNICITY["Meanings"].loc[3]
Convert_To_Meanings(df,ETHNICITY,"ethnicity","Meanings")
df_Bis["age"].iloc[0]=float(df_Bis["age"].iloc[0])
df_Bis["gender"].iloc[0]=float(df_Bis["gender"].iloc[0])
df_Bis["education"].iloc[0]=float(df_Bis["education"].iloc[0])
df_Bis["country"].iloc[0]=float(df_Bis["country"].iloc[0])
df_Bis["ethnicity"].iloc[0]=float(df_Bis["ethnicity"].iloc[0])
```

Data Preprocessing : Cleaning

- **Delete from the dataframe rows corresponding to Semer use :**

```
for i in df.index:
    if df['semer'].loc[i] > 0 :
        df = df.drop(labels=i)
```

Data Preprocessing : Cleaning

- **Conversion to float for each dataframe :**

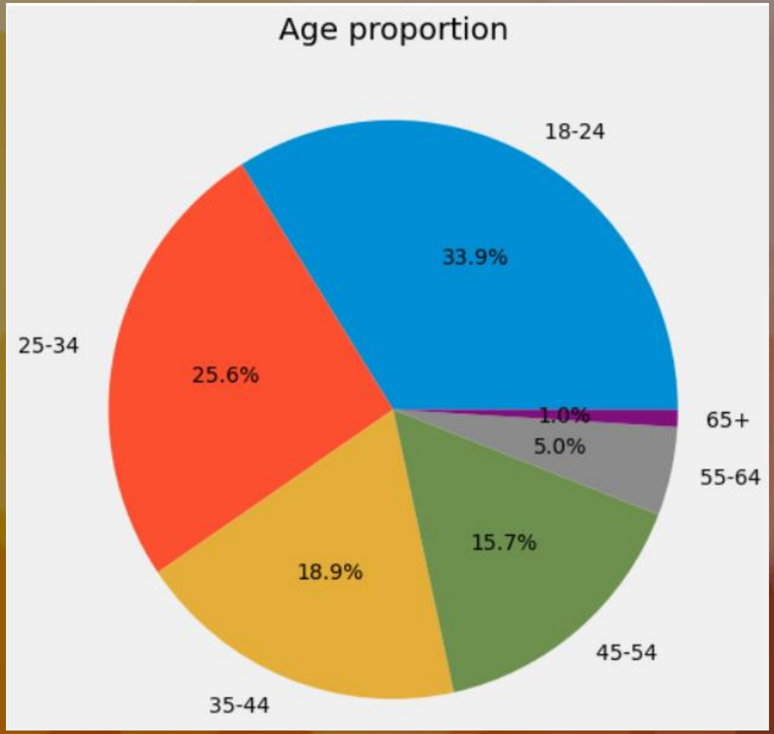
```
df = df.astype({'nscore':'float',
                'oscore':'float',
                'escore':'float',
                'ascore':'float',
                'cscore':'float',
                'ss':'float',
                'impulsive':'float'})
```

```
df_Bis = df_Bis.astype({'nscore':'float',
                        'oscore':'float',
                        'escore':'float',
                        'ascore':'float',
                        'cscore':'float',
                        'ss':'float',
                        'impulsive':'float'})
```

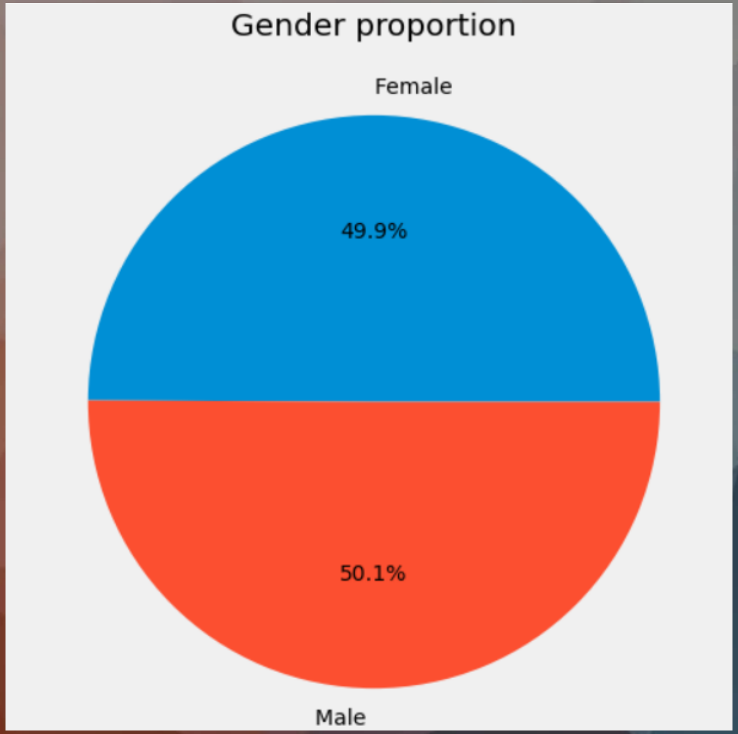
nscore	escore	oscore	ascore	cscore	impulsive	ss
0.31287	-0.57545	-0.58331	-0.91699	-0.00665	-0.21712	-1.18084

Data visualization : General

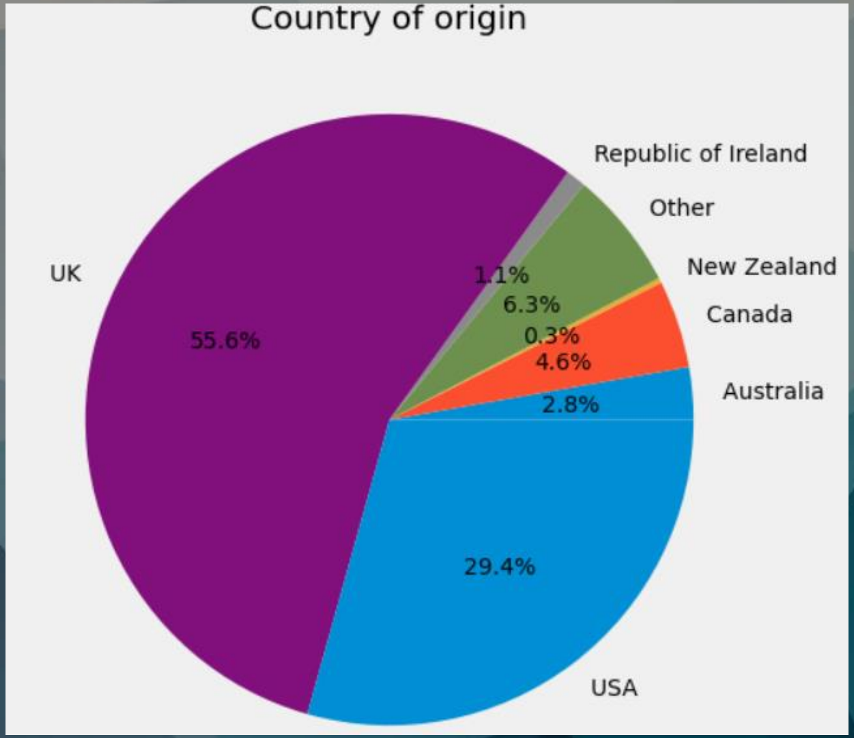
- **Age**



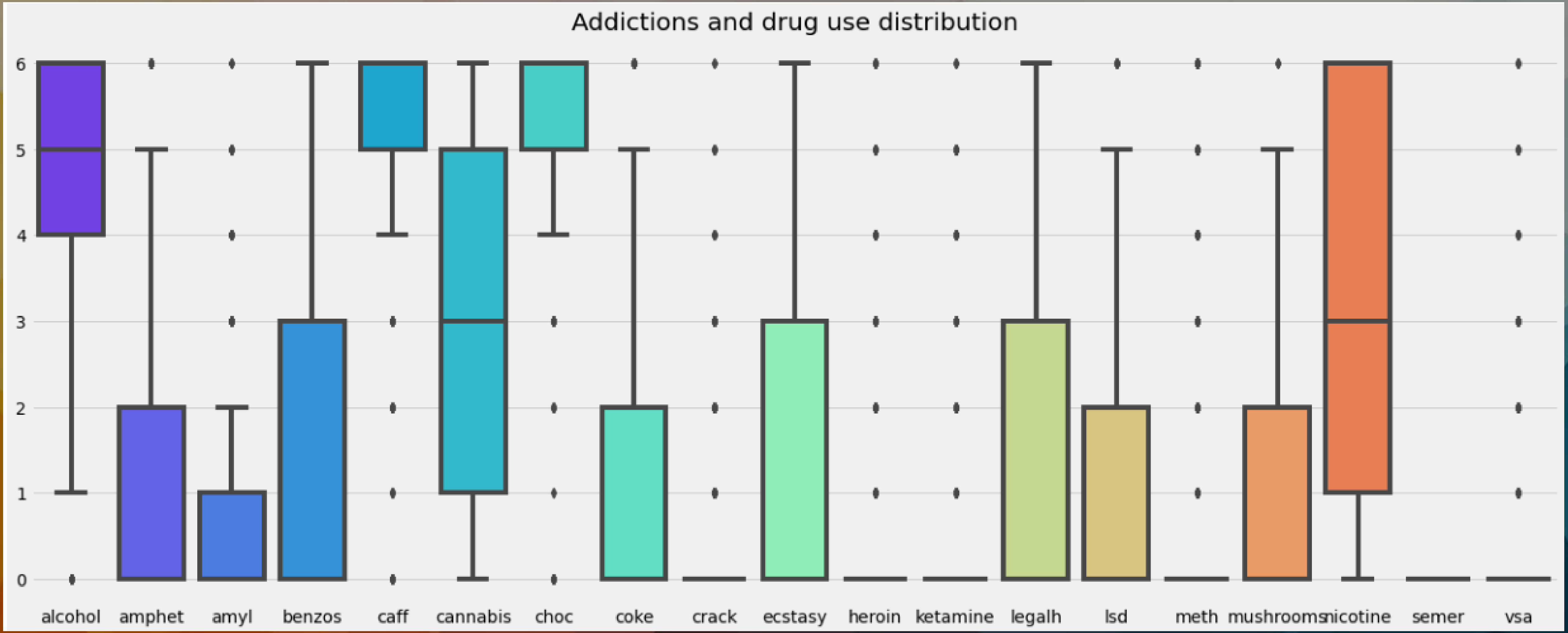
- **Gender**

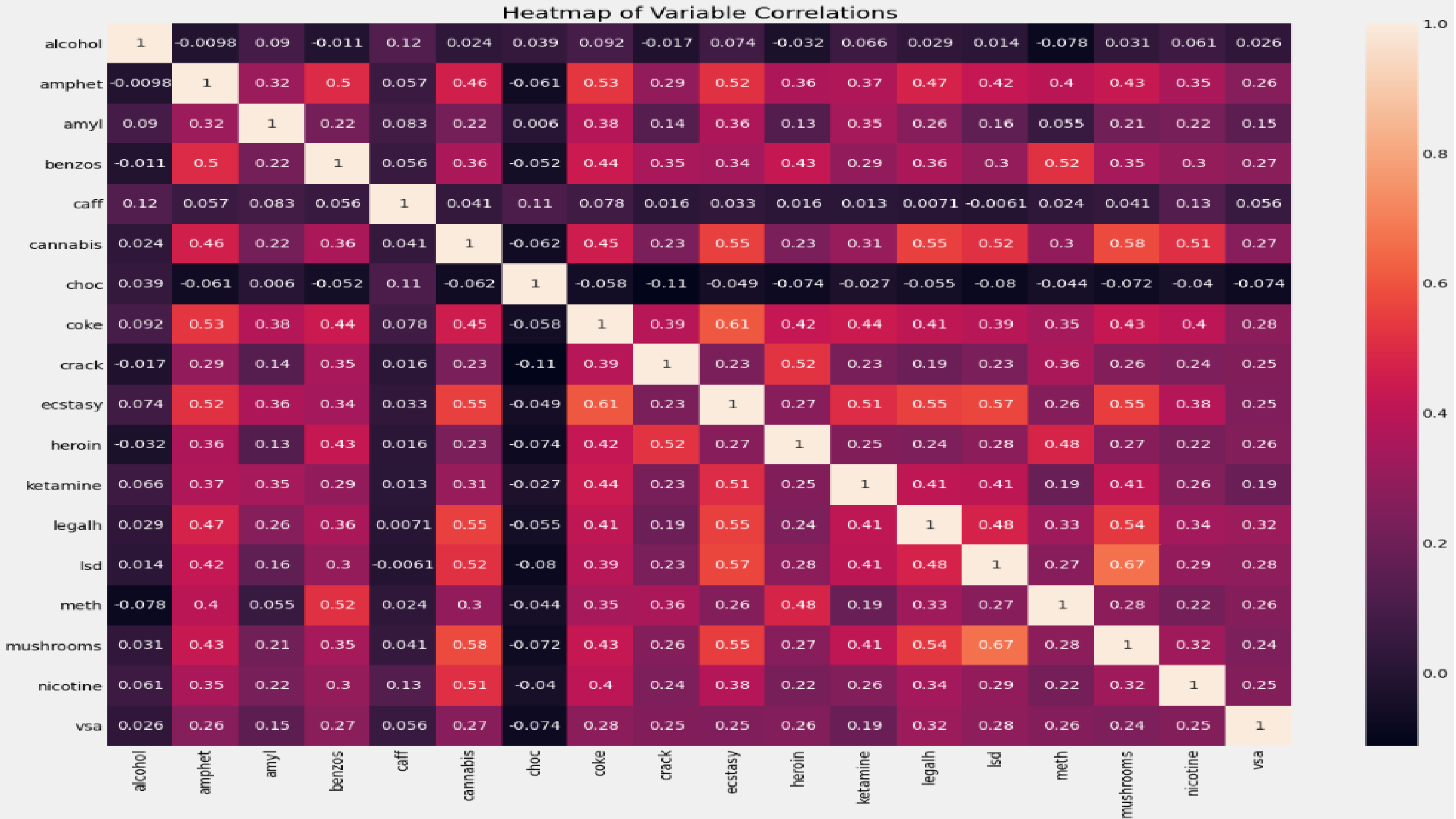


- **Country**

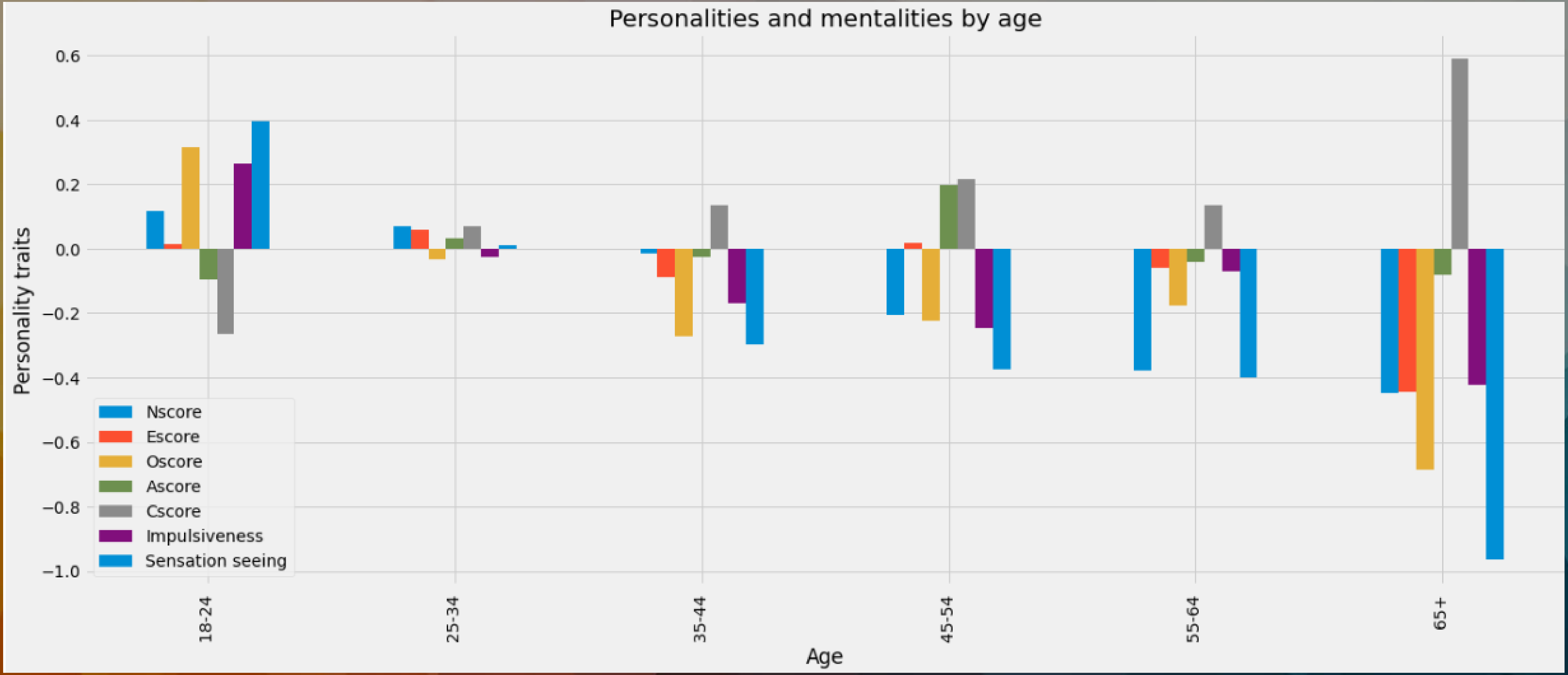


Data visualization : General

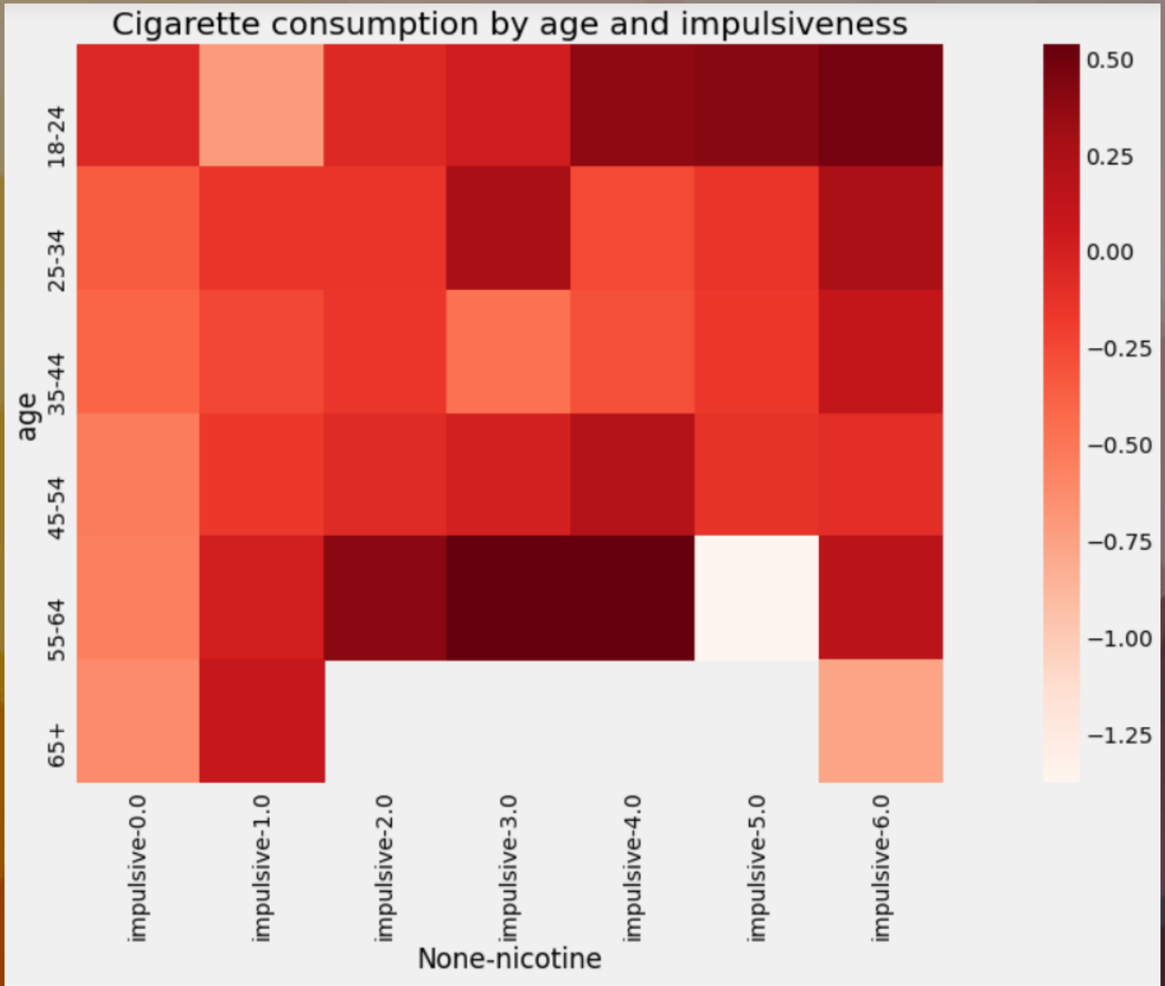


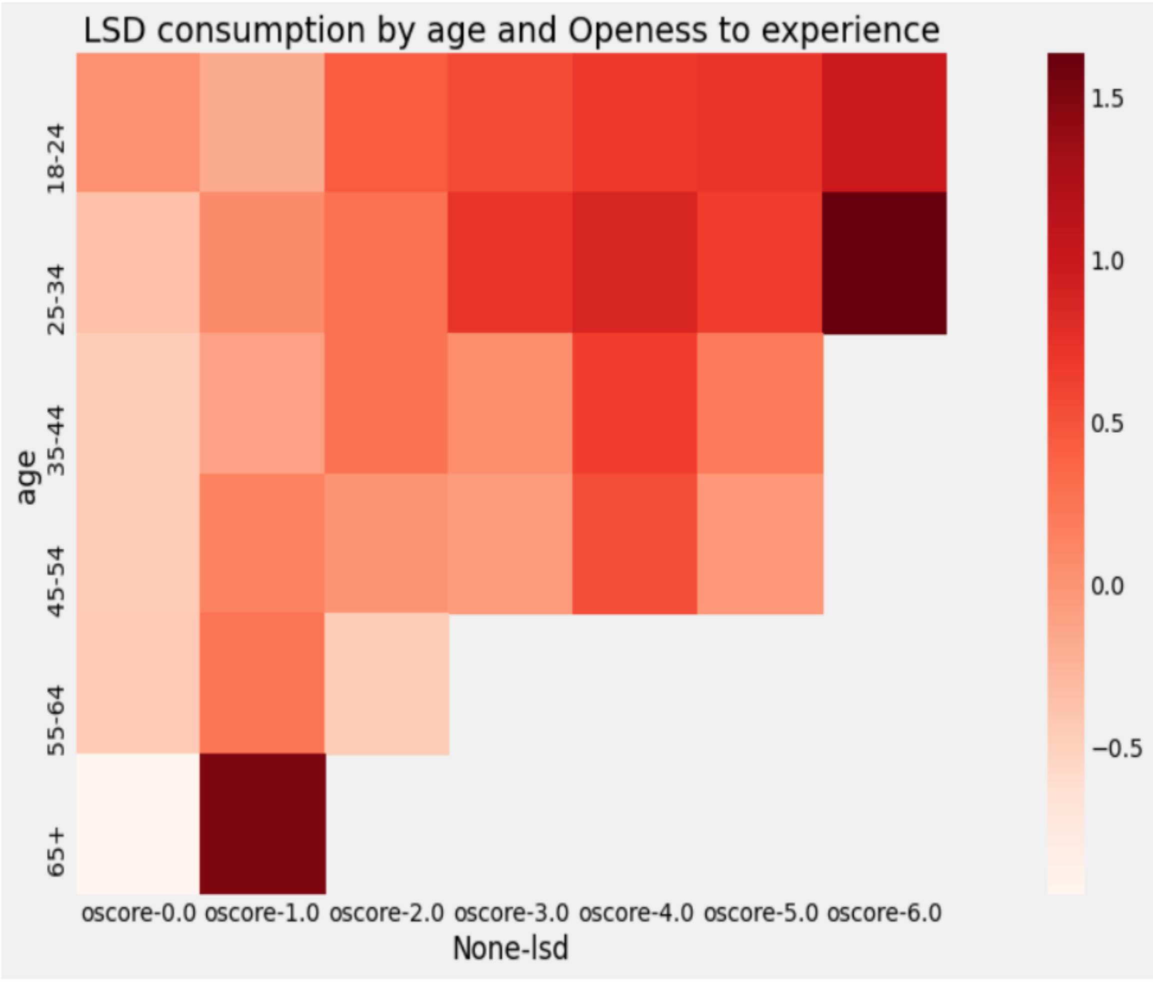

Heatmap of Variable Correlations
alcohol 1 -0.0098 0.09 -0.011 0.12 0.024 0.039 0.092 -0.017 0.074 -0.032 0.066 0.029 0.014 -0.078 0.031 0.061 0.026
amphet -0.0098 1 0.32 0.5 0.057 0.46 -0.061 0.53 0.29 0.52 0.36 0.37 0.47 0.42 0.4 0.43 0.35 0.26
amyl 0.09 0.32 1 0.22 0.083 0.22 0.006 0.38 0.14 0.36 0.13 0.35 0.26 0.16 0.055 0.21 0.22 0.15
benzos -0.011 0.5 0.22 1 0.056 0.36 -0.052 0.44 0.35 0.34 0.43 0.29 0.36 0.3 0.52 0.35 0.3 0.27
caff 0.12 0.057 0.083 0.056 1 0.041 0.11 0.078 0.016 0.033 0.016 0.013 0.0071 -0.0061 0.024 0.041 0.13 0.056
cannabis 0.024 0.46 0.22 0.36 0.041 1 -0.062 0.45 0.23 0.55 0.23 0.31 0.55 0.52 0.3 0.58 0.51 0.27
choc 0.039 -0.061 0.006 -0.052 0.11 -0.062 1 -0.058 -0.11 -0.049 -0.074 -0.027 -0.055 -0.08 -0.044 -0.072 -0.04 -0.074
coke 0.092 0.53 0.38 0.44 0.078 0.45 -0.058 1 0.39 0.61 0.42 0.44 0.41 0.39 0.35 0.43 0.4 0.28
crack -0.017 0.29 0.14 0.35 0.016 0.23 -0.11 0.39 1 0.23 0.52 0.23 0.19 0.23 0.36 0.26 0.24 0.25
ecstasy 0.074 0.52 0.36 0.34 0.033 0.55 -0.049 0.61 0.23 1 0.27 0.51 0.55 0.57 0.26 0.55 0.38 0.25
heroin -0.032 0.36 0.13 0.43 0.016 0.23 -0.074 0.42 0.52 0.27 1 0.25 0.24 0.28 0.48 0.27 0.22 0.26
ketamine 0.066 0.37 0.35 0.29 0.013 0.31 -0.027 0.44 0.23 0.51 0.25 1 0.41 0.41 0.19 0.41 0.26 0.19
legalh 0.029 0.47 0.26 0.36 0.0071 0.55 -0.055 0.41 0.19 0.55 0.24 0.41 1 0.48 0.33 0.54 0.34 0.32
lsd 0.014 0.42 0.16 0.3 -0.0061 0.52 -0.08 0.39 0.23 0.57 0.28 0.41 0.48 1 0.27 0.67 0.29 0.28
meth -0.078 0.4 0.055 0.52 0.024 0.3 -0.044 0.35 0.36 0.26 0.48 0.19 0.33 0.27 1 0.28 0.22 0.26
mushrooms 0.031 0.43 0.21 0.35 0.041 0.58 -0.072 0.43 0.26 0.55 0.27 0.41 0.54 0.67 0.28 1 0.32 0.24
nicotine 0.061 0.35 0.22 0.3 0.13 0.51 -0.04 0.4 0.24 0.38 0.22 0.26 0.34 0.29 0.22 0.32 1 0.25
vsa 0.026 0.26 0.15 0.27 0.056 0.27 -0.074 0.28 0.25 0.25 0.26 0.19 0.32 0.28 0.26 0.24 0.25 1
alcohol amphet amyl benzos caff cannabis choc coke crack ecstasy heroin ketamine legalh lsd meth mushrooms nicotine vsa
1.0 0.8 0.6 0.4 0.2 0.0

Data visualization : Age

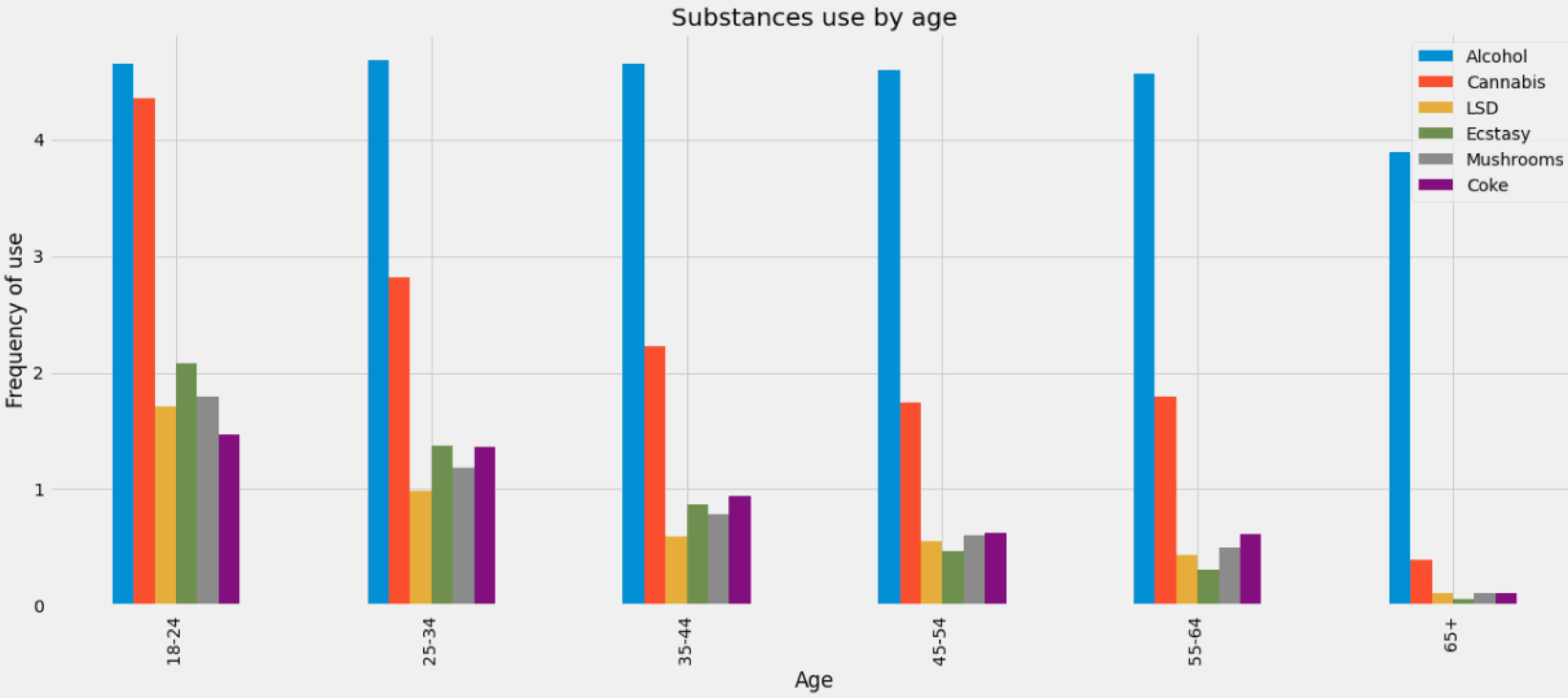


Data visualization : Age

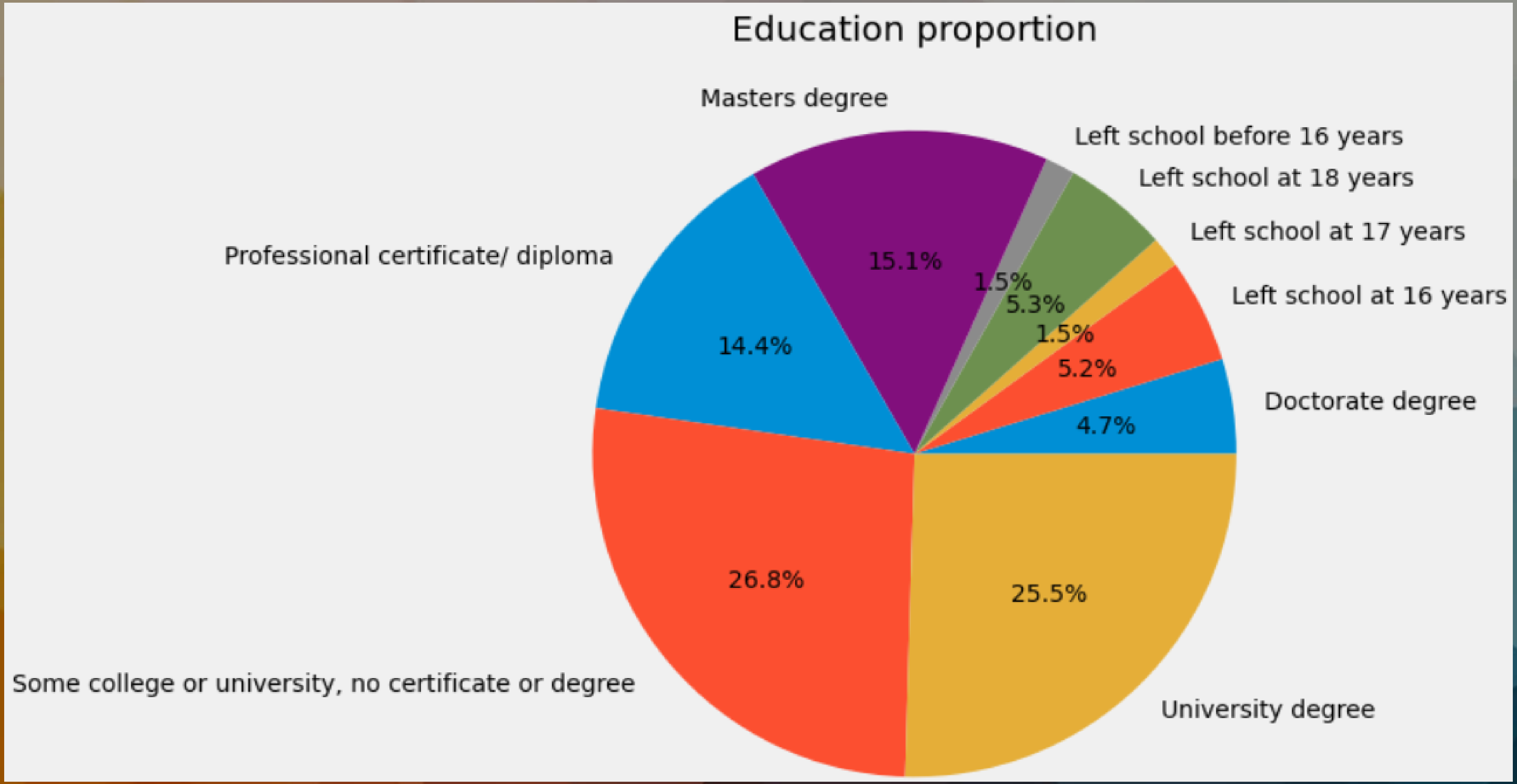




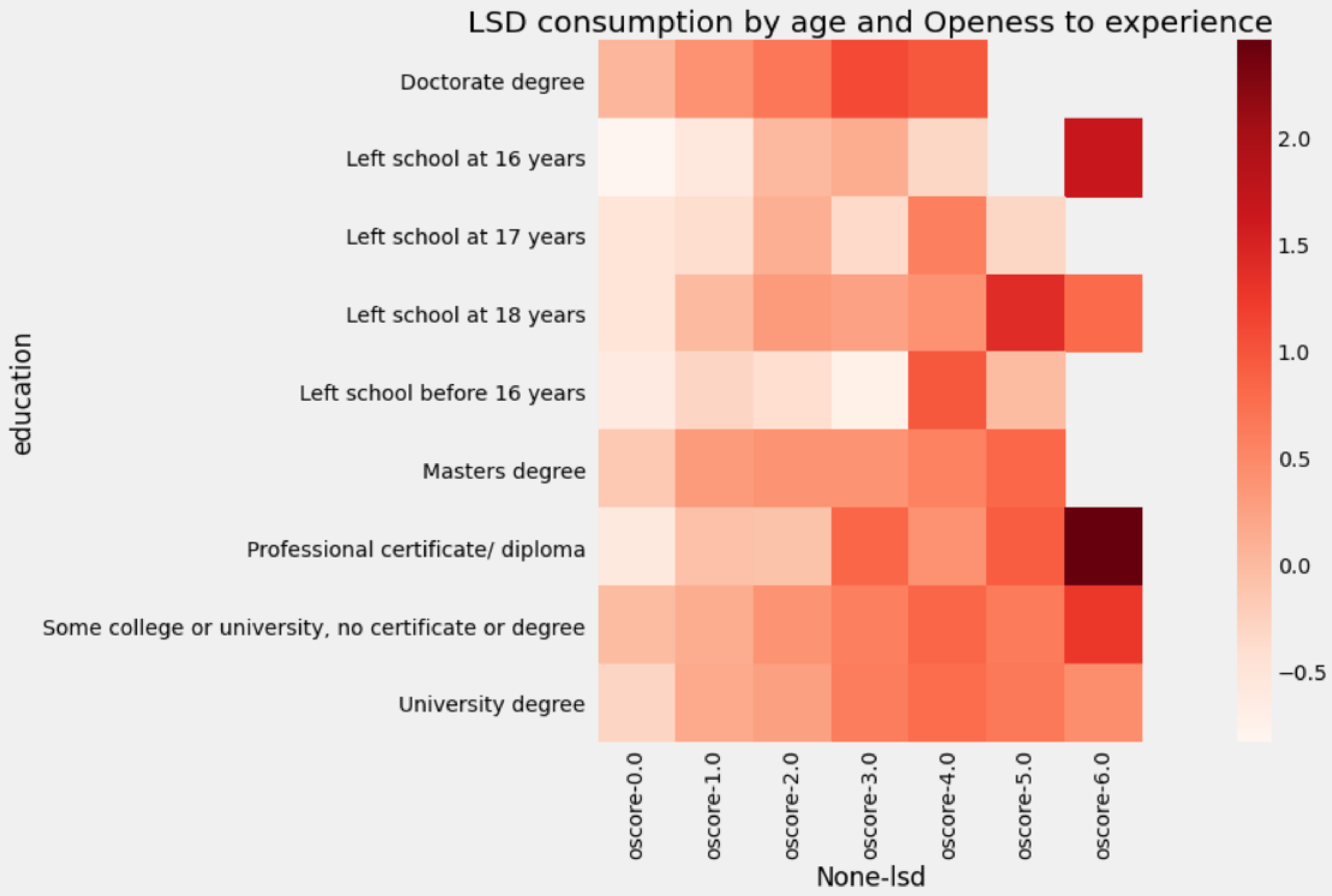
Data visualization : Age



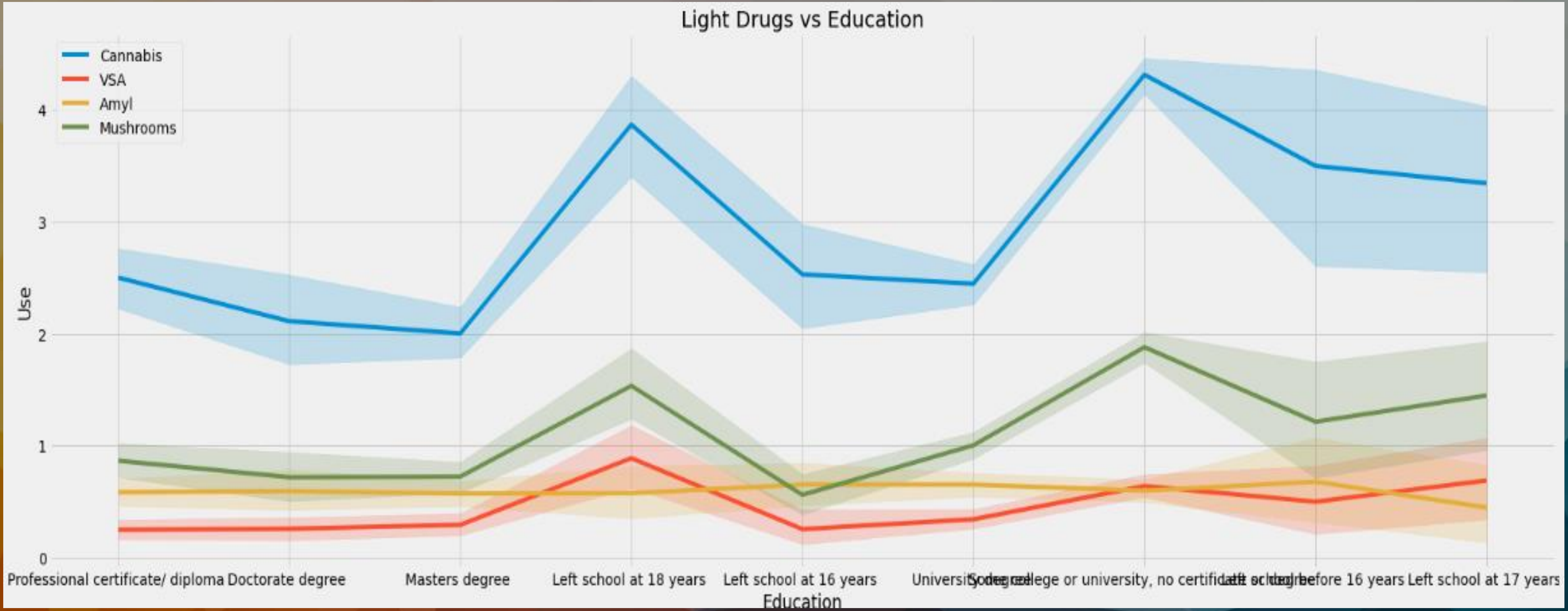
Data visualization : Education



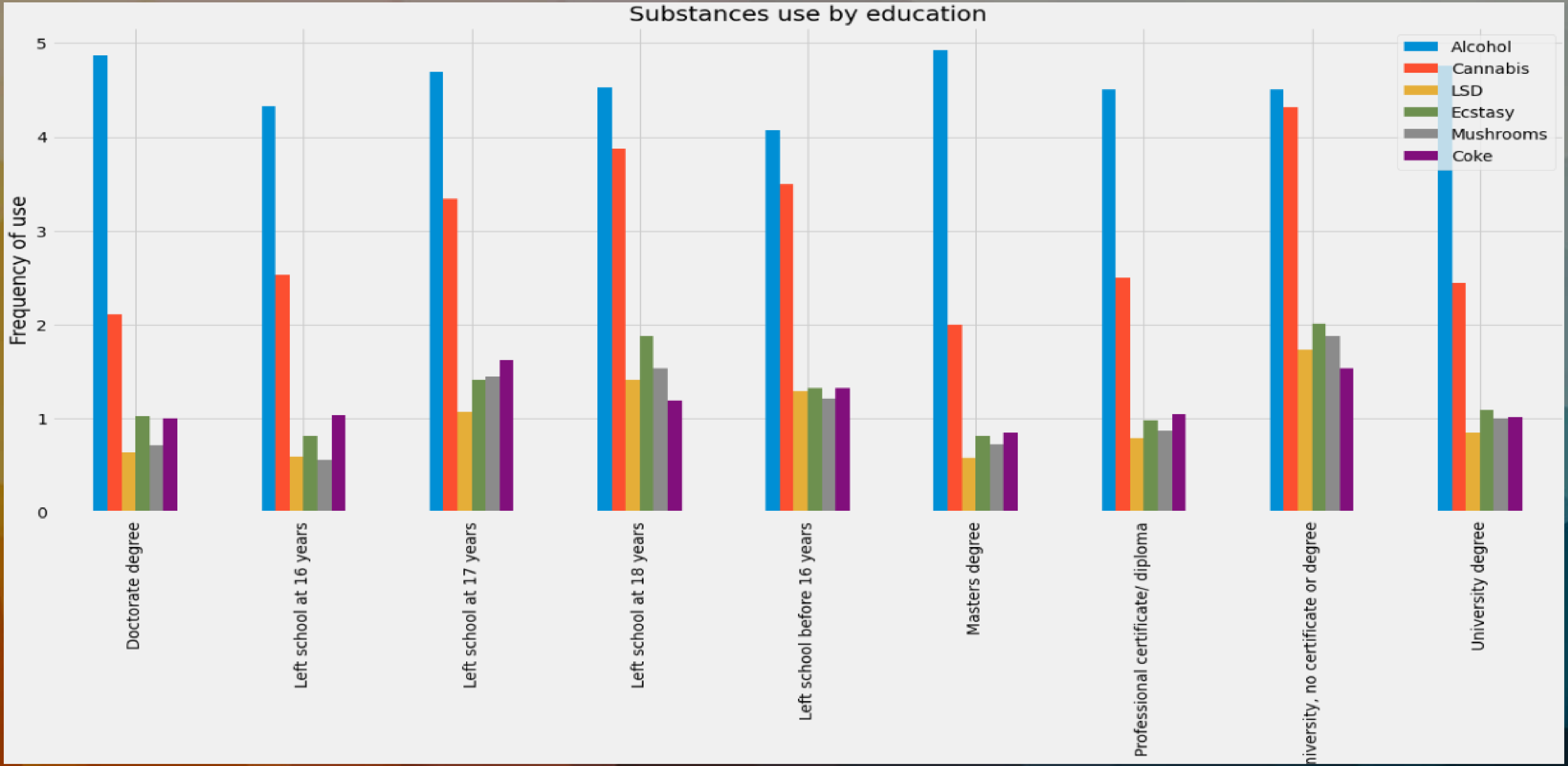
Data visualization : Education



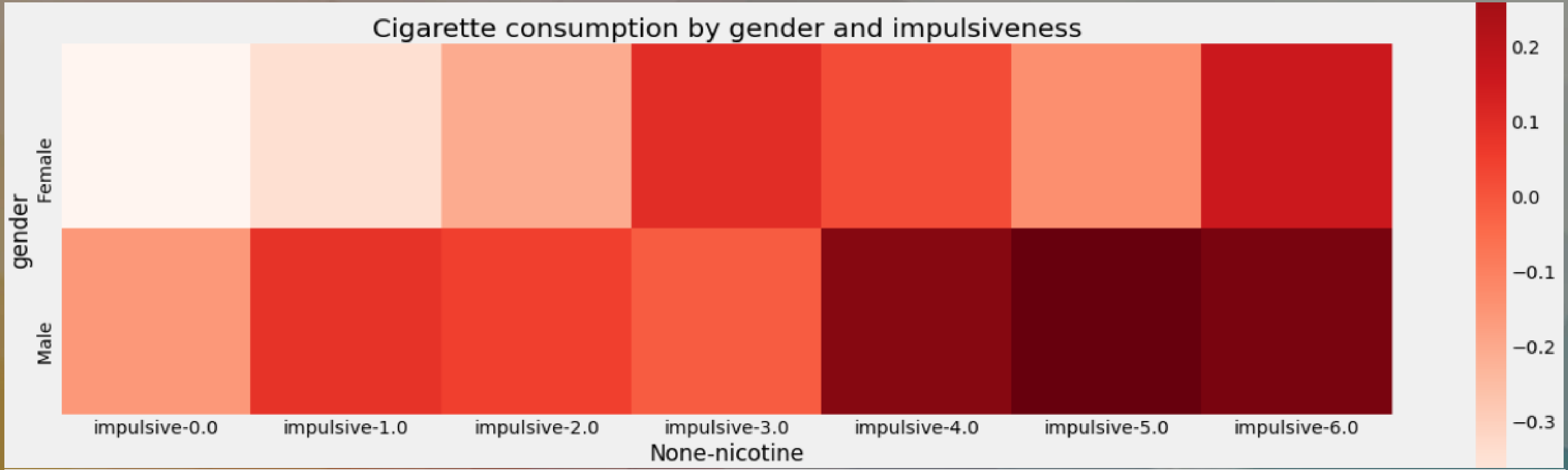
Data visualization : Education

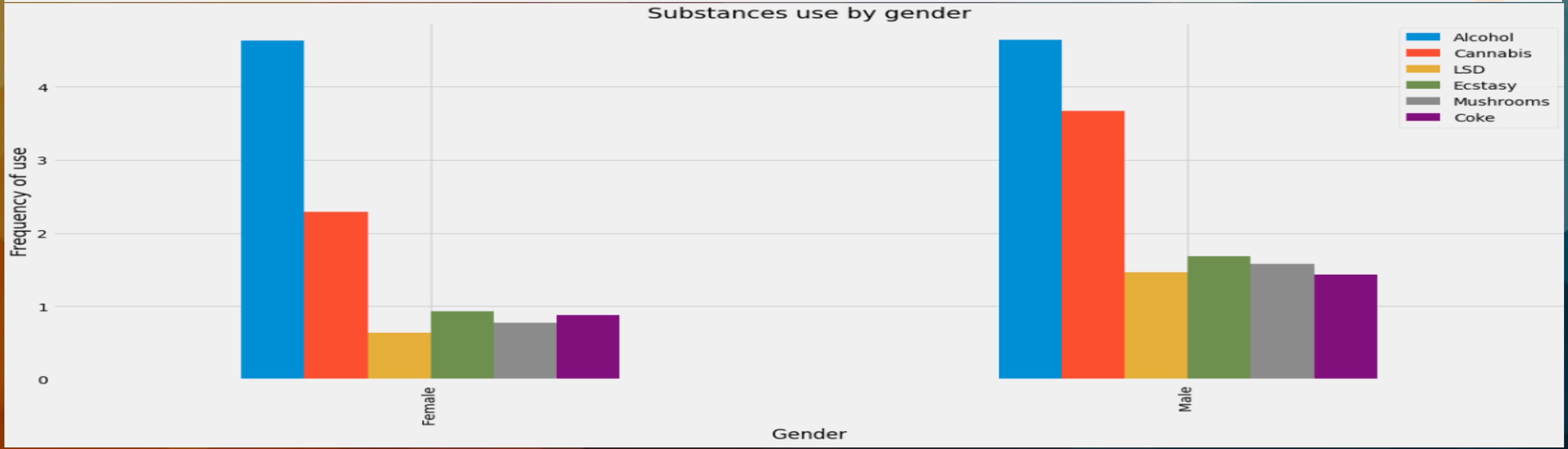


Data visualization : Education

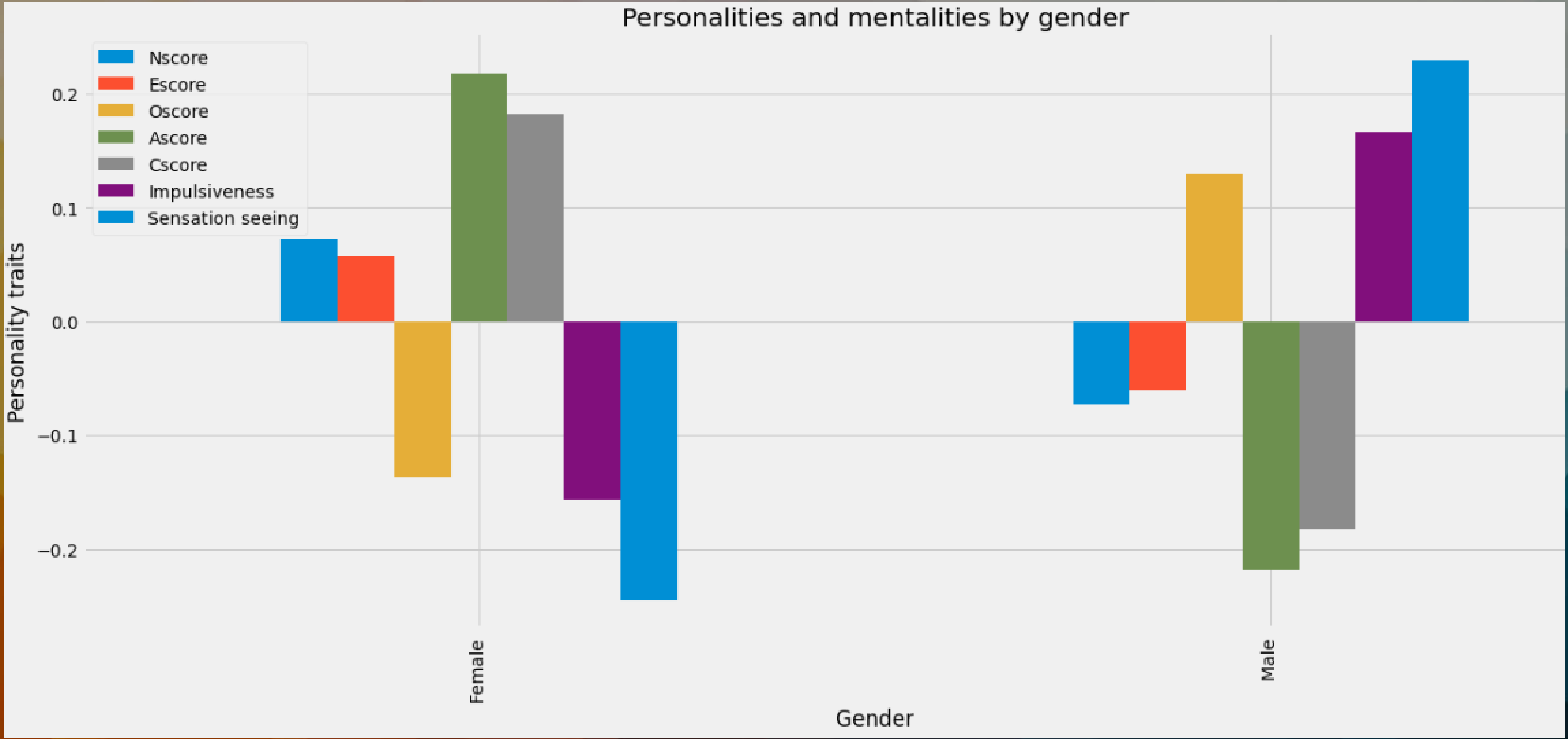


Data visualization : Gender

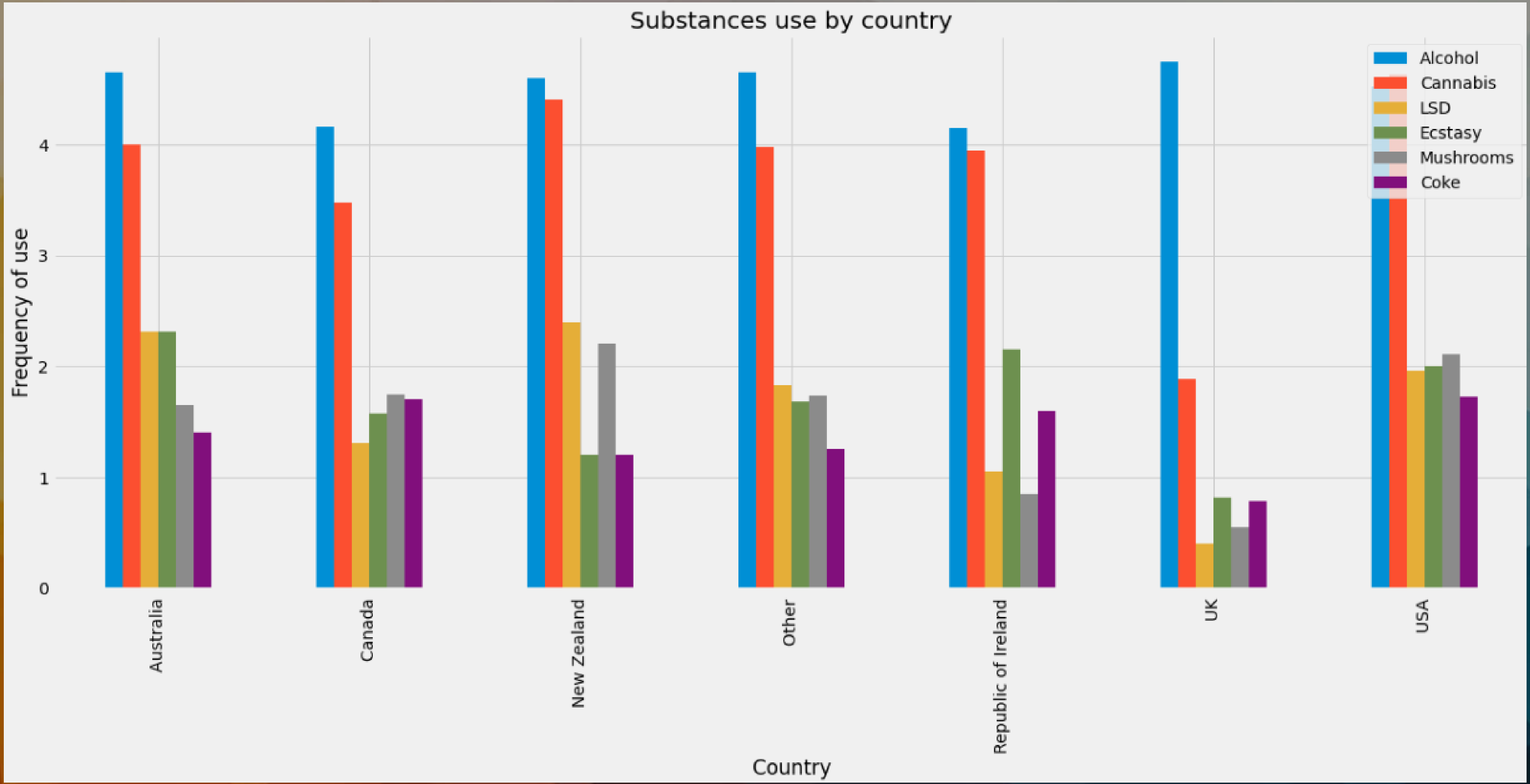




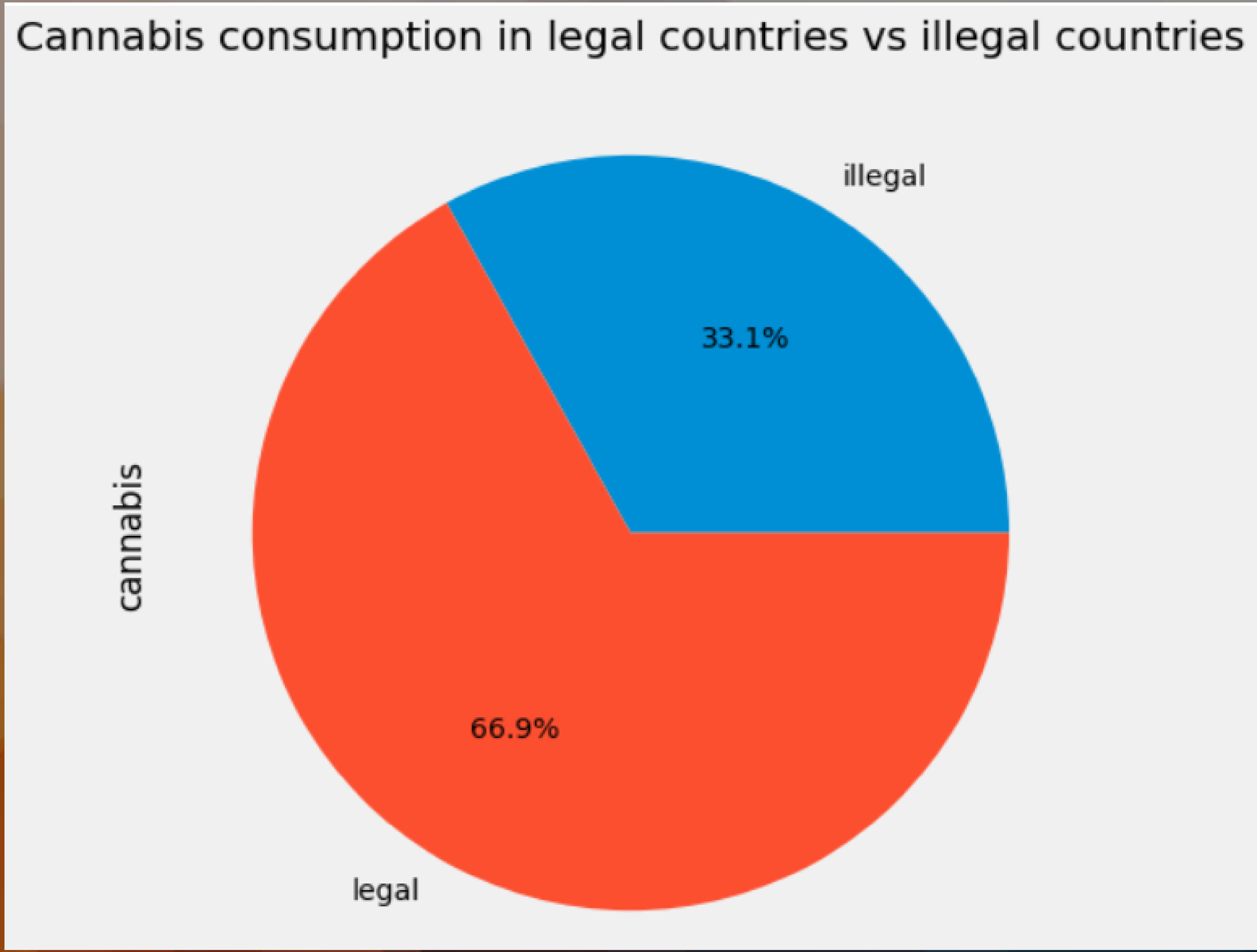
Data visualization : Gender



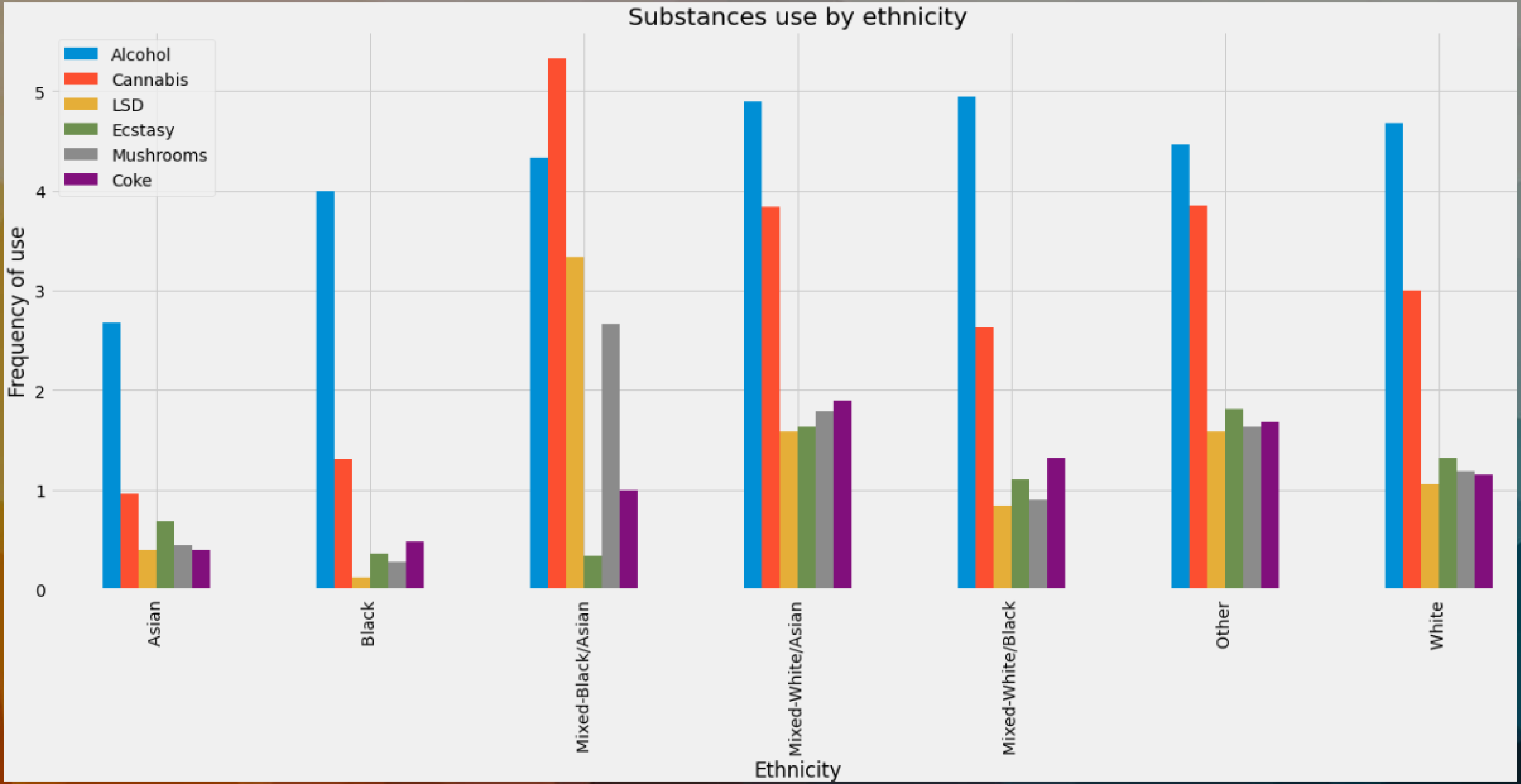
Data visualization : Country



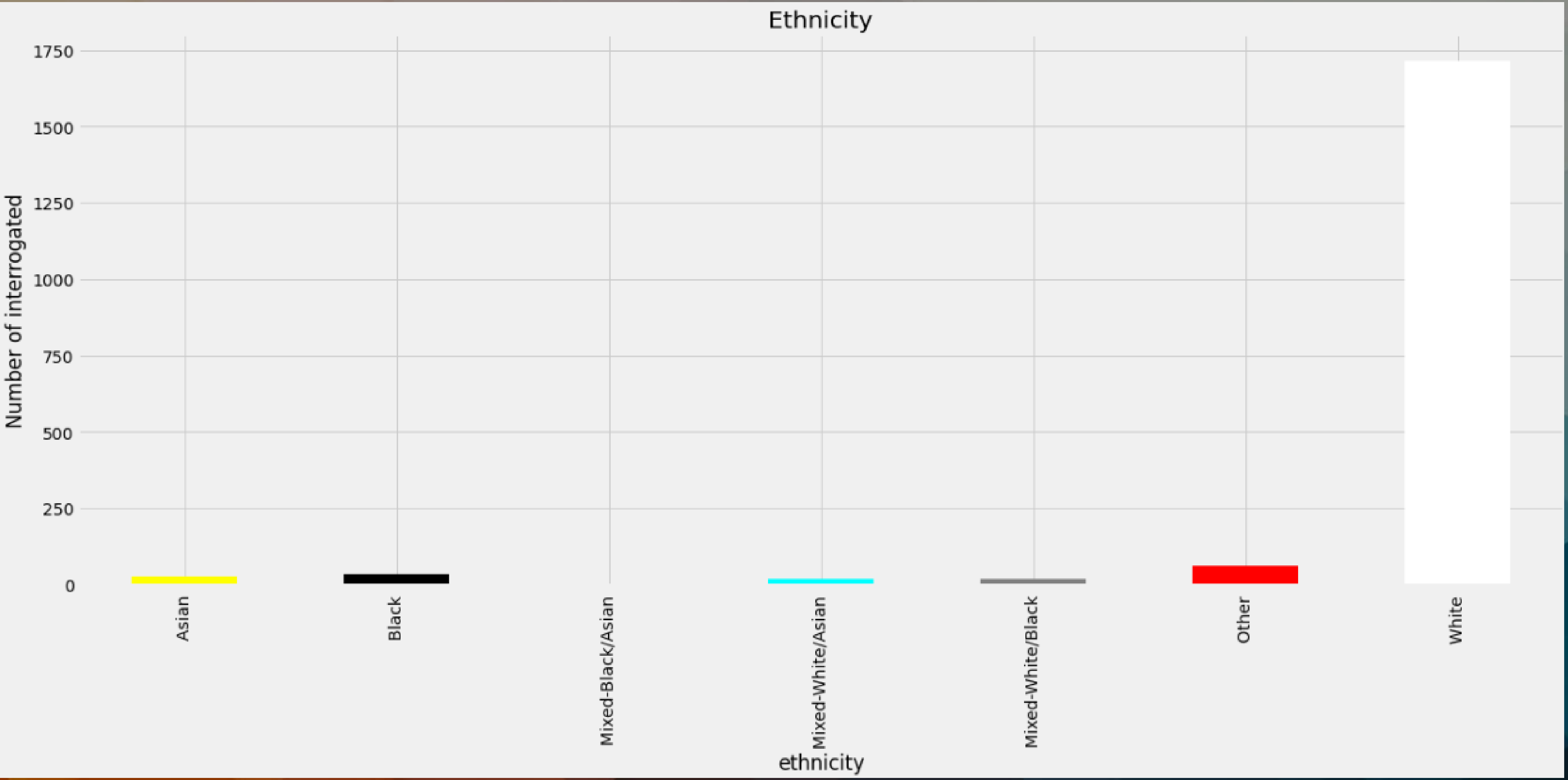
Data visualization : Country



Data visualization : Ethnicity



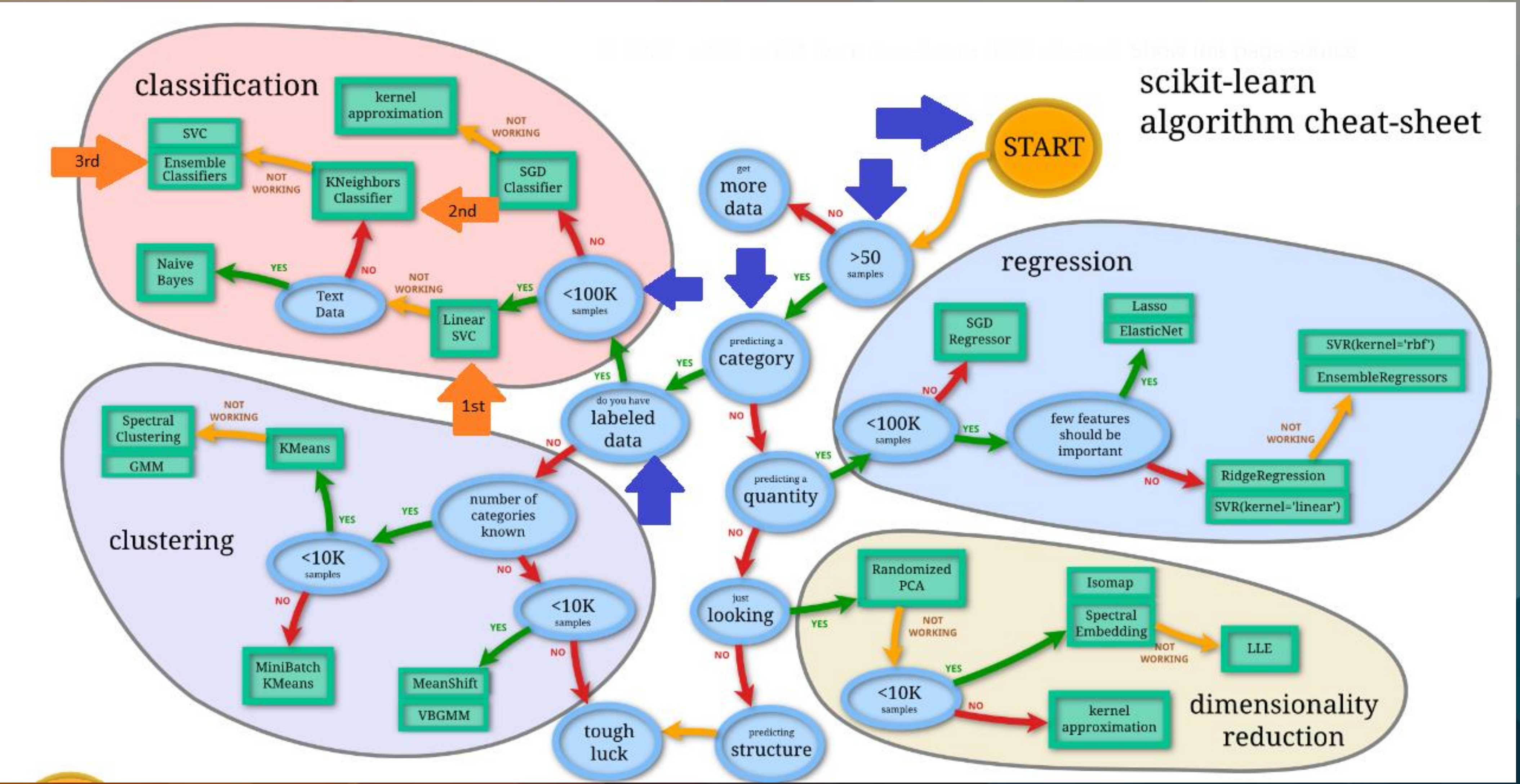
Data visualization : Ethnicity



ML Model : Summary

- How did we choose our models?
- PCA
- 1st Model : Linear SVC
- 2nd Model : kNN Classifier
- 3rd Model : Optimized kNN

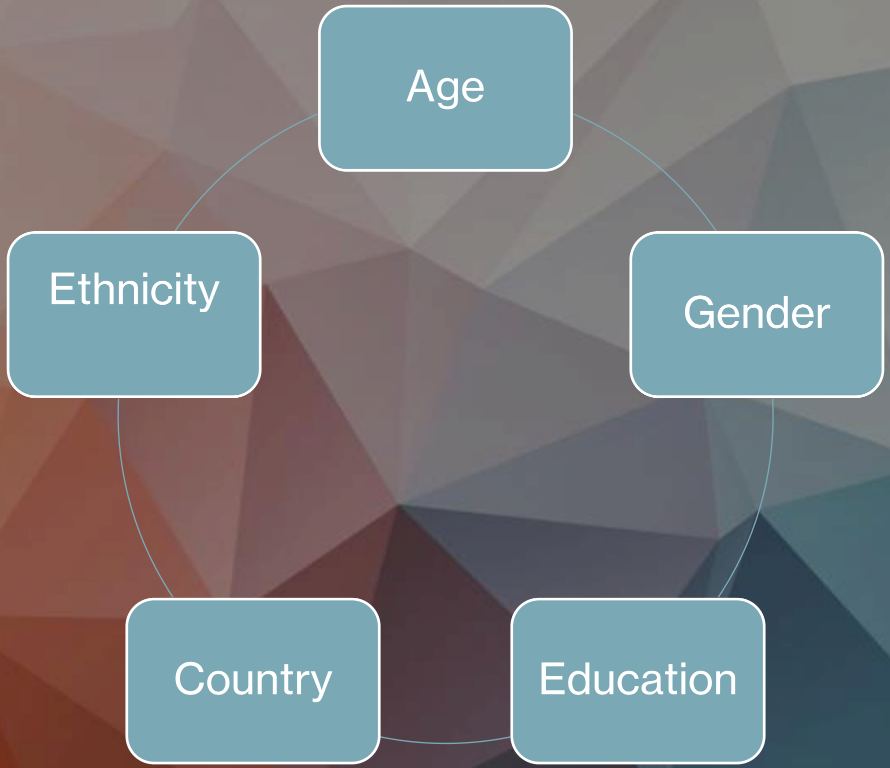
ML Model : How did we choose our models?



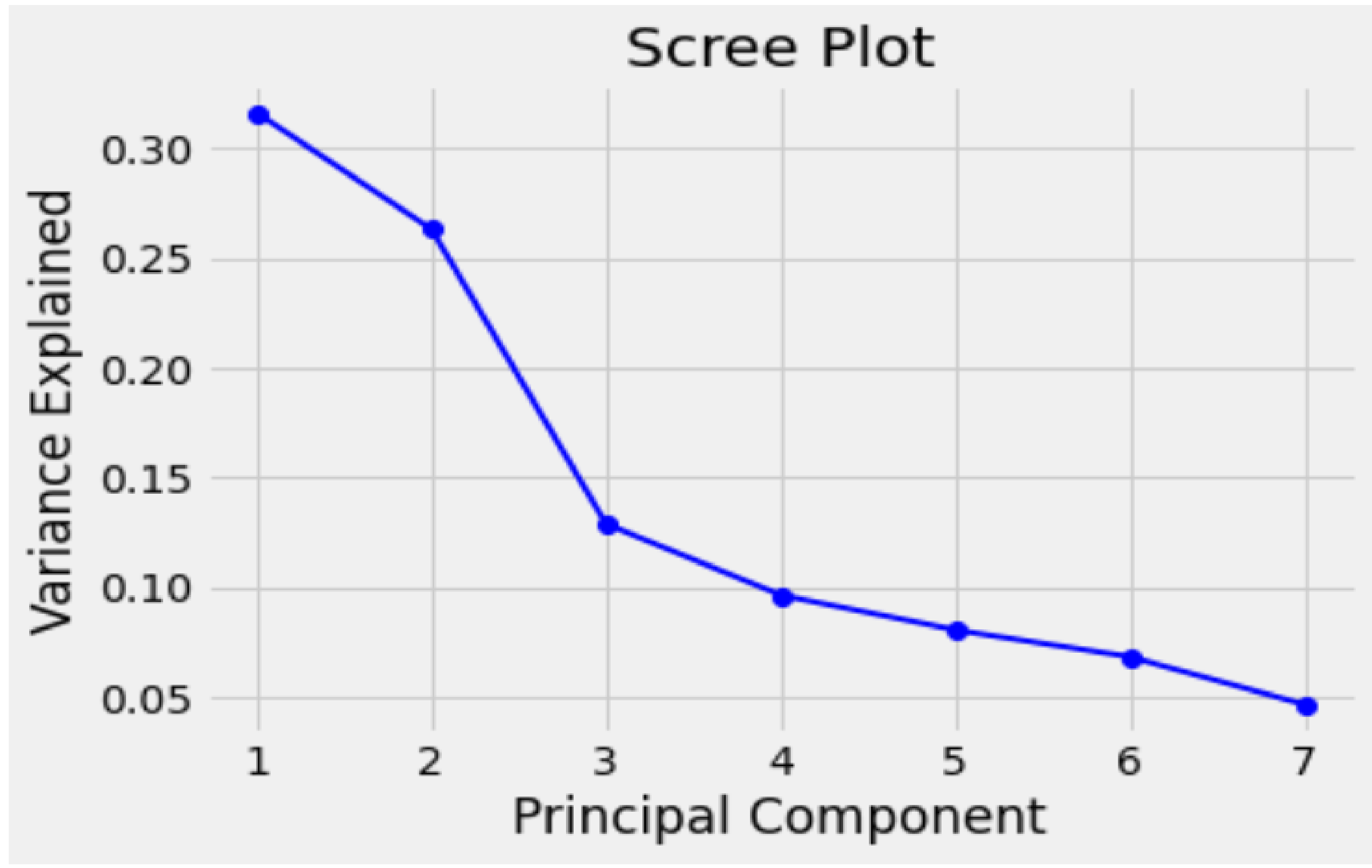
PCA : Defining the features

Features	Description
Nscore	Neuroticism
Escore	Extraversion
Oscore	Openness to experience
Ascore	Agreeableness
Cscore	Conscientiousness
Impulsive	impulsiveness
ss	sensation seeing

PCA : Defining the targets



PCA : Defining the number of features we will use



PCA : total variance explained by the 3 first features

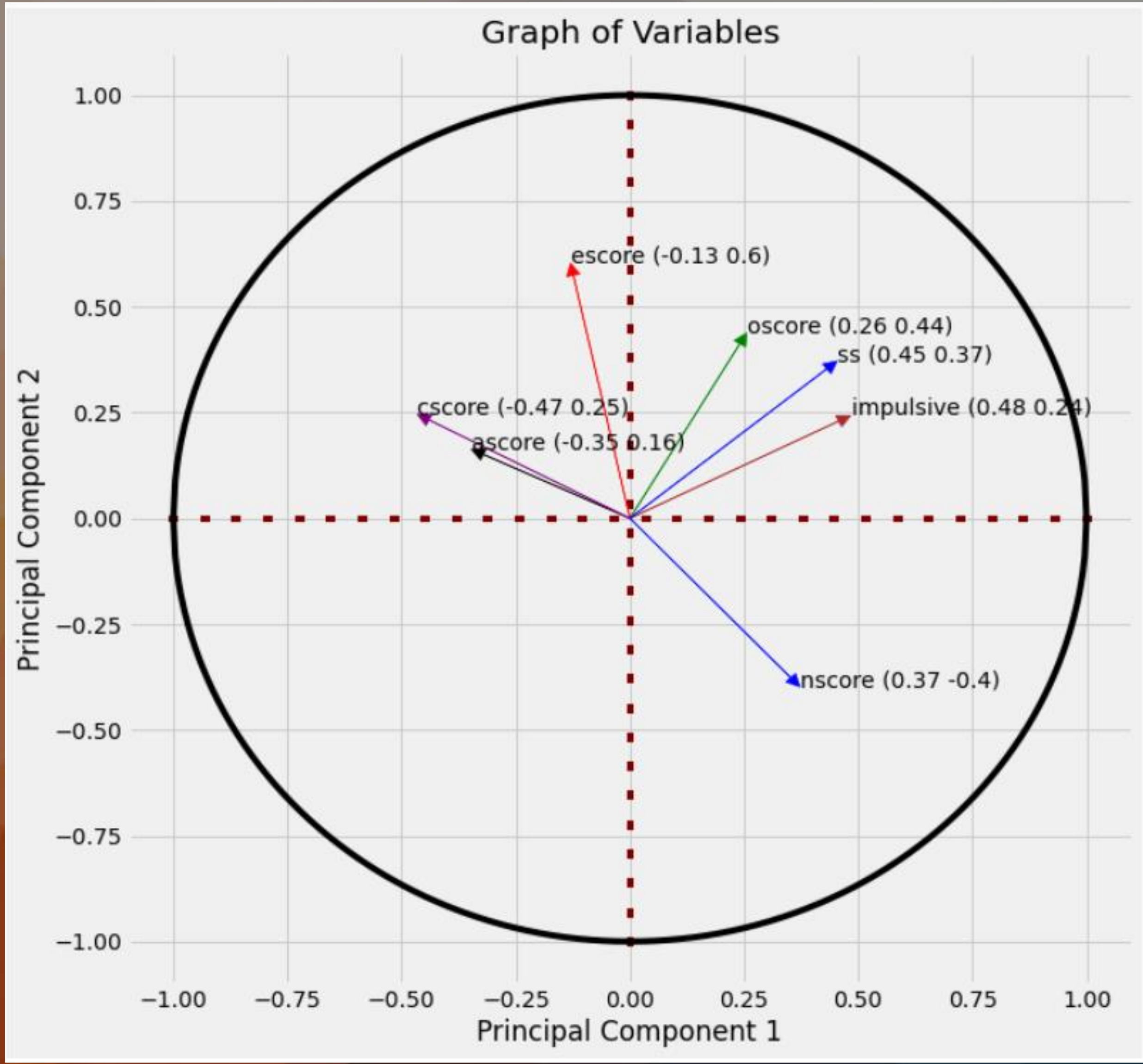
	Explained_Variance_Ratio	Cumulated_Variance_Ratio
0	31.583524	31.583524
1	26.291498	57.875023
2	12.887505	70.762528

PCA : final dataframe

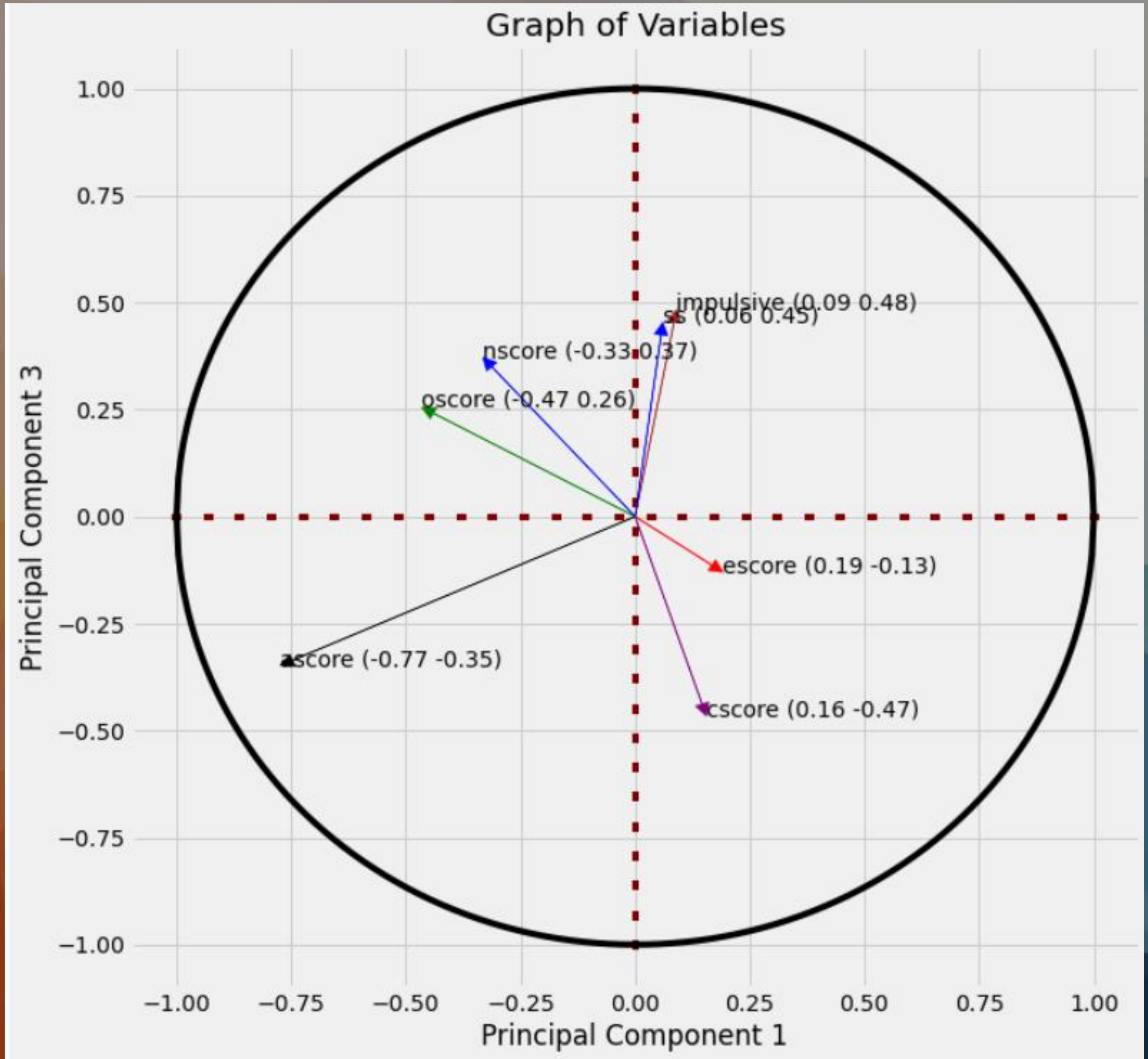
	principal component 1	principal component 2	principal component 3	age	gender	education	country	ethnicity
0	-0.279247	-1.371571	0.674529	35-44	Female	Professional certificate/ diploma	UK	Mixed-White/Asian
1	-0.780152	1.905536	-0.760571	25-34	Male	Doctorate degree	UK	White
2	0.053882	-0.400647	1.700962	35-44	Male	Professional certificate/ diploma	UK	White
3	-1.639724	-0.969600	-0.659309	18-24	Female	Masters degree	UK	White
4	-0.338376	-1.337332	0.055615	35-44	Female	Doctorate degree	UK	White
...	...	...	...	...	...	...	...	...
1880	1.373765	3.126114	-0.718232	18-24	Female	Some college or university, no certificate or ...	USA	White
1881	1.046007	1.653768	-0.555810	18-24	Male	Some college or university, no certificate or ...	USA	White
1882	1.552295	-2.539307	1.125537	25-34	Female	University degree	USA	White
1883	3.607464	-1.523392	0.236595	18-24	Female	Some college or university, no certificate or ...	USA	White
1884	0.371907	3.150401	-0.853361	18-24	Male	Some college or university, no certificate or ...	Republic of Ireland	White

1885 rows × 8 columns

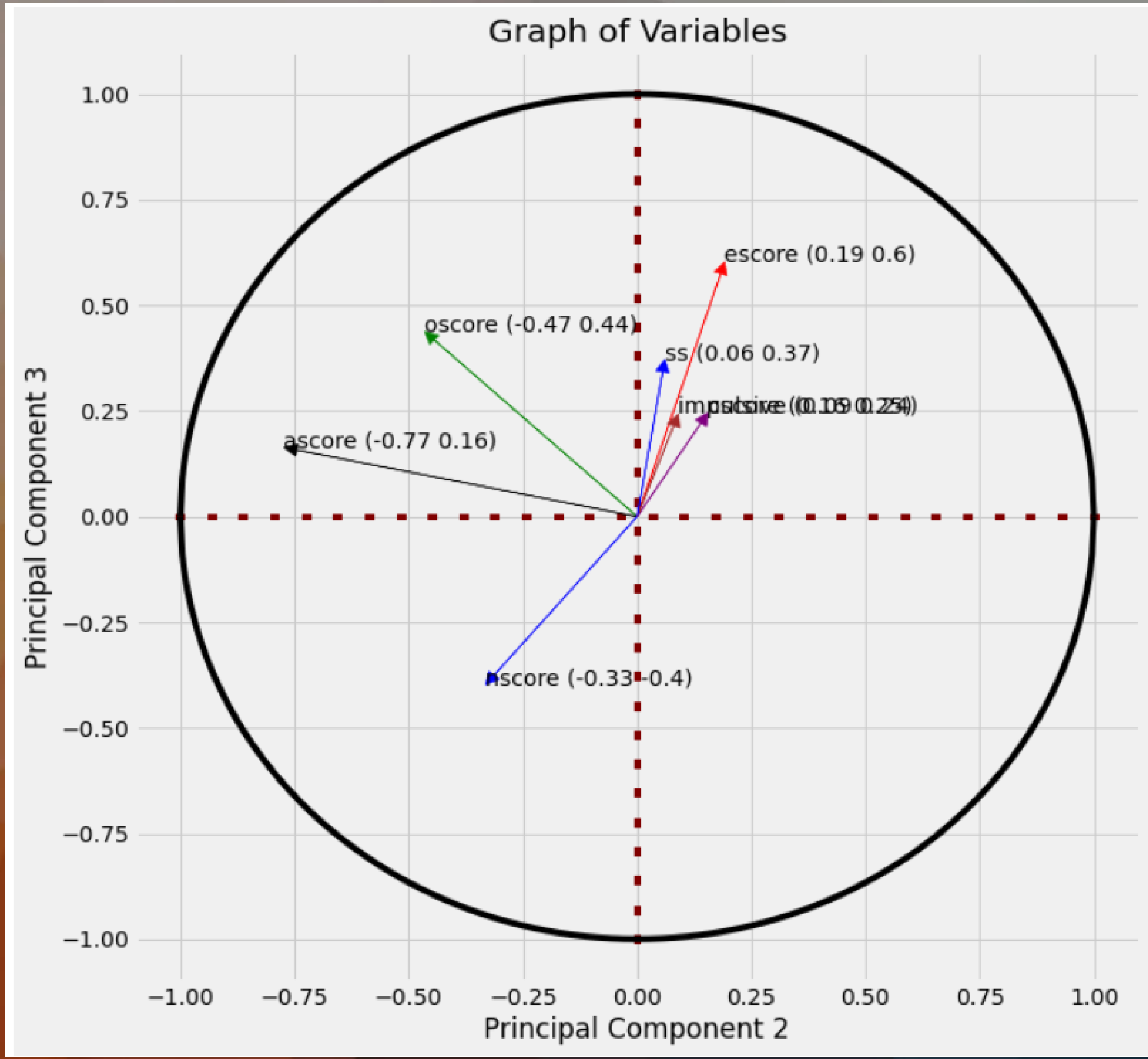
PCA : Graph of Variables (PC1 vs PC2)



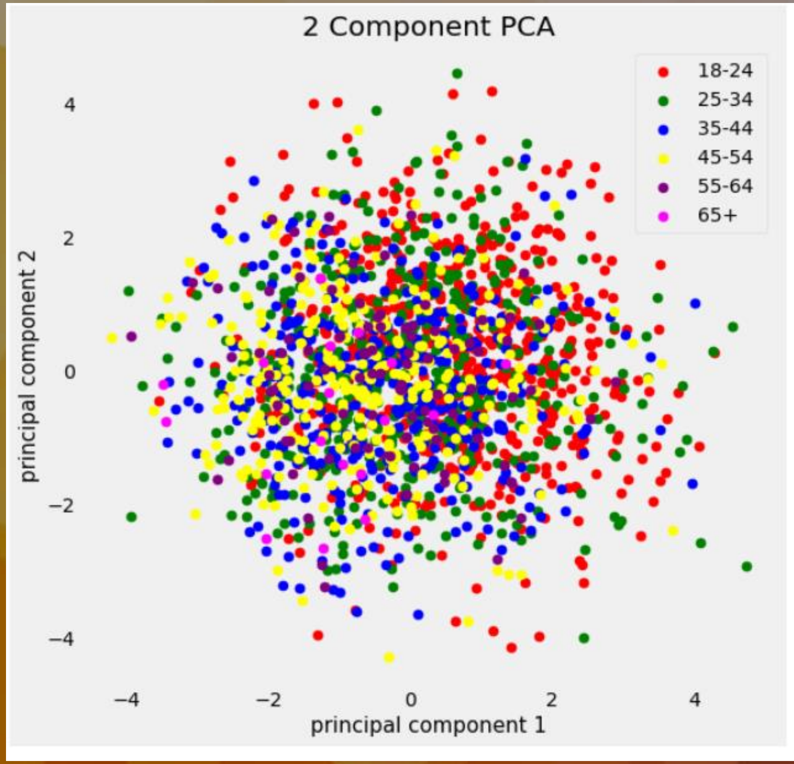
PCA : Graph of Variables (PC3 vs PC1)

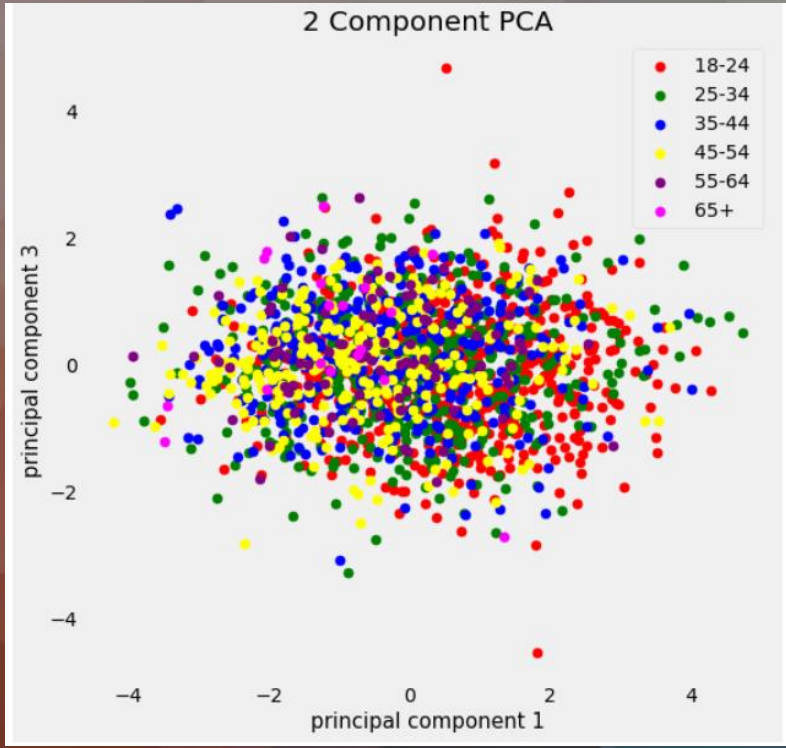


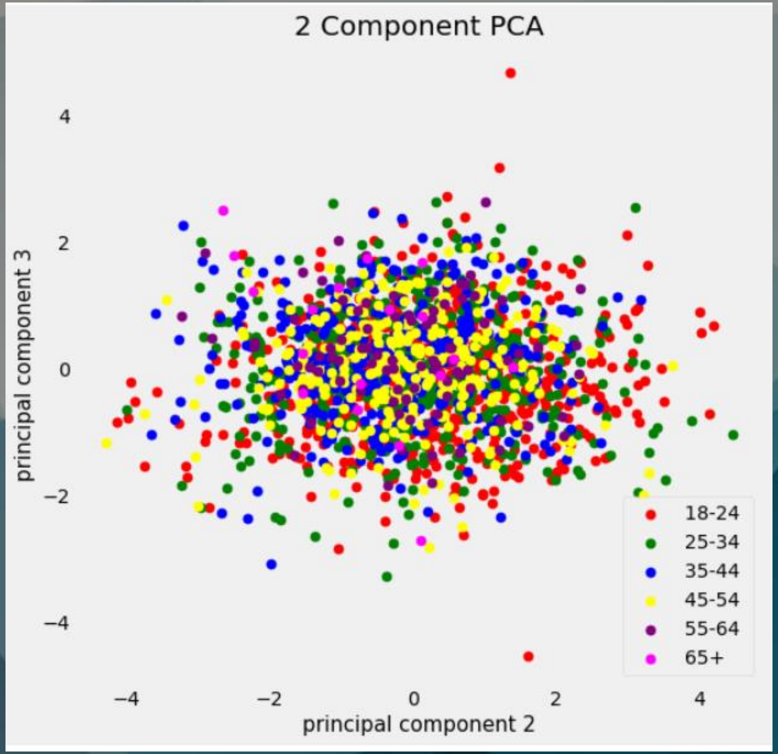
PCA : Graph of Variables (PC3 vs PC2)



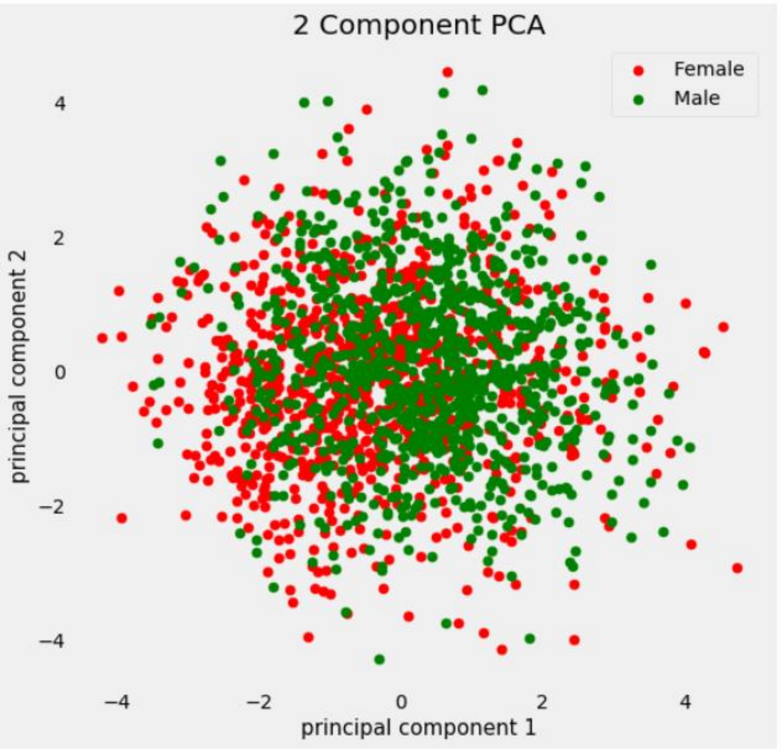
PCA : Graph of Individuals (Age)

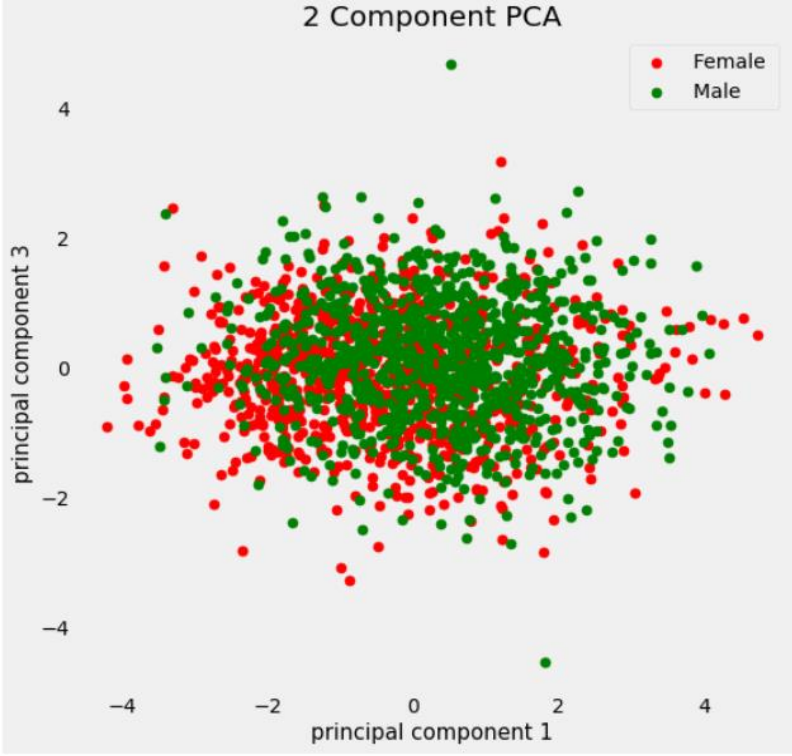


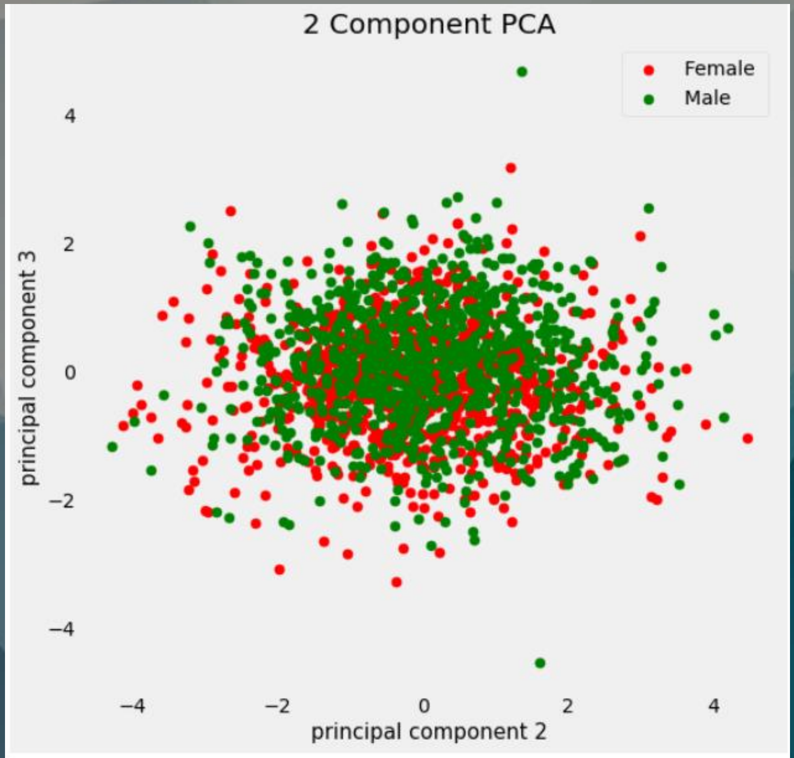




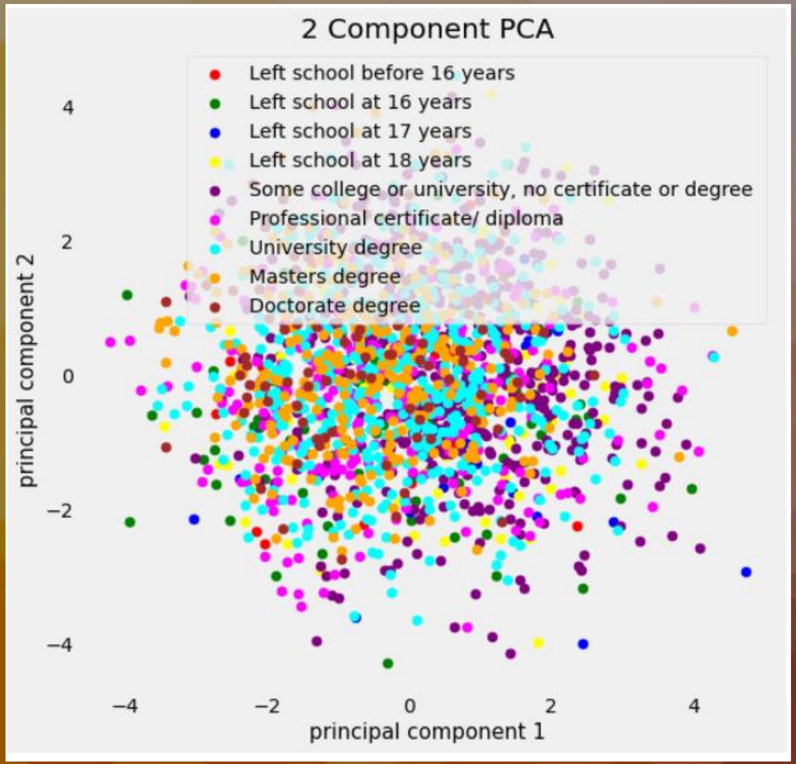
PCA : Graph of Individuals (Gender)

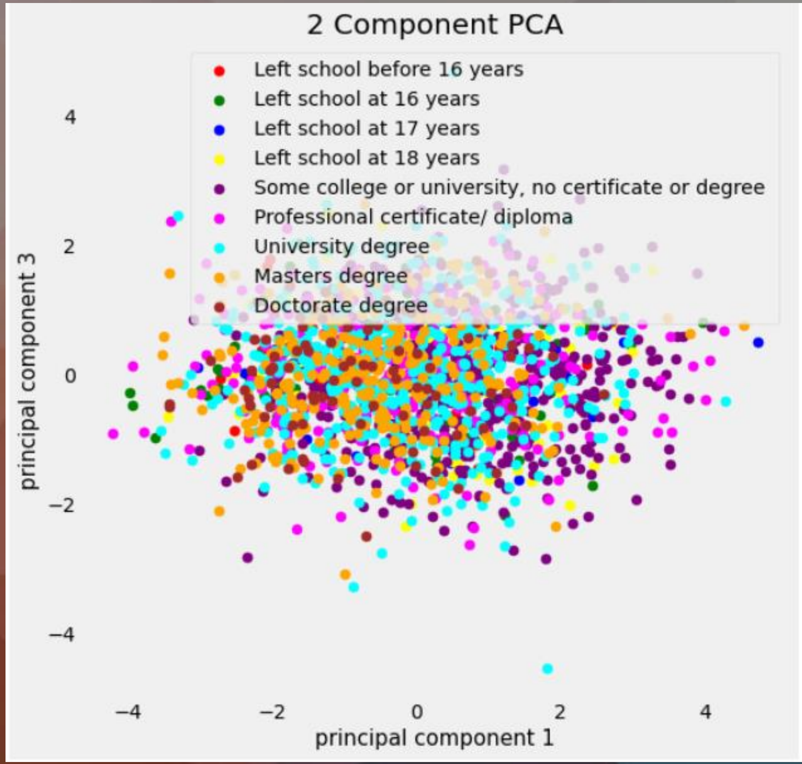


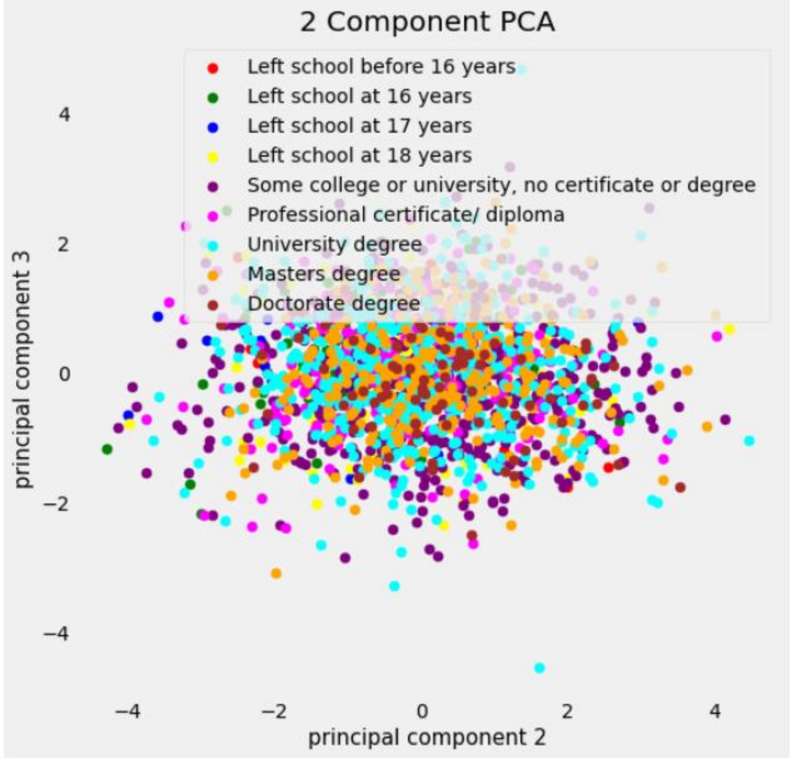




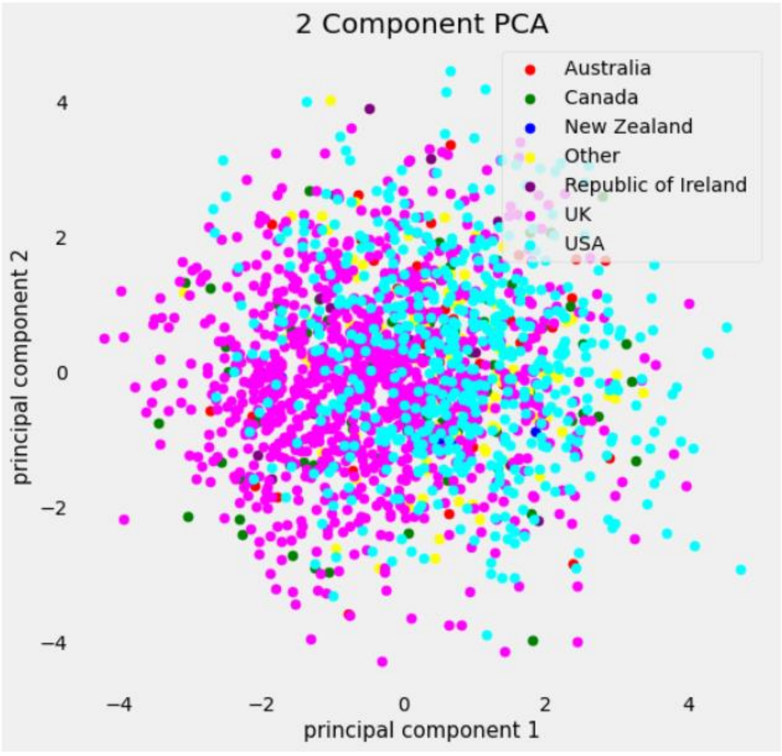
PCA : Graph of Individuals (Education)

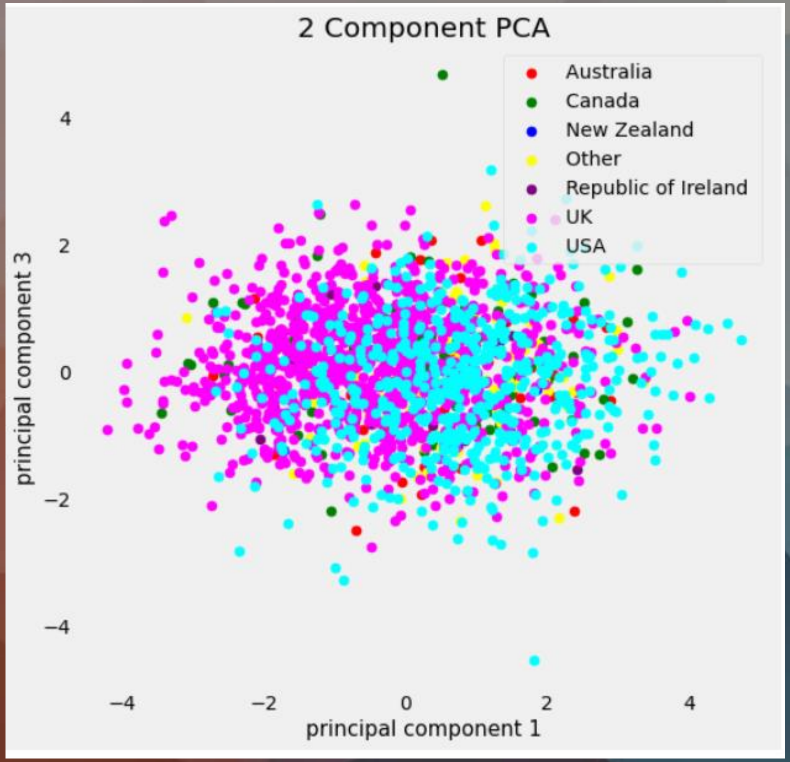


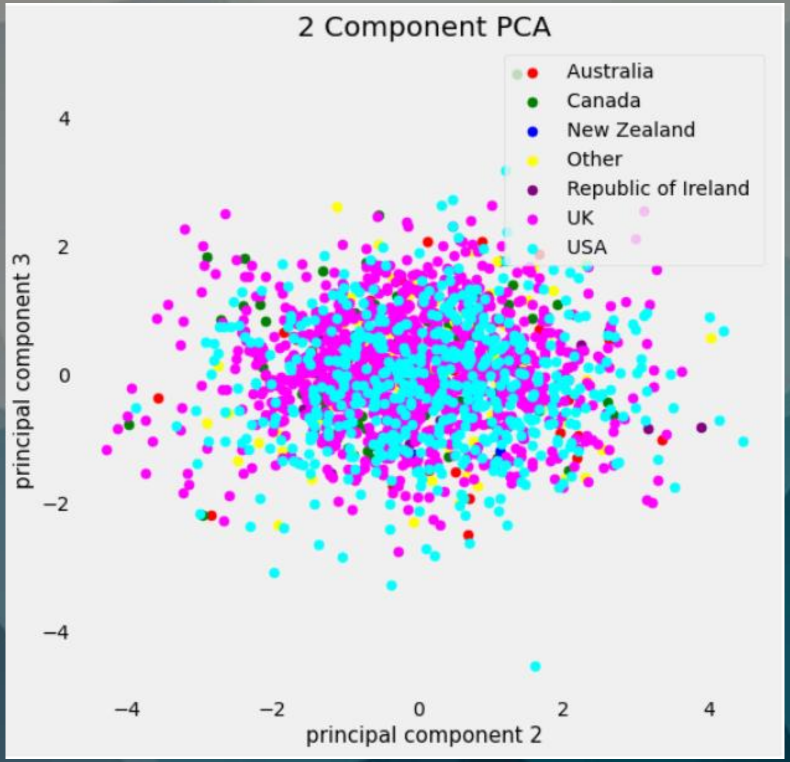




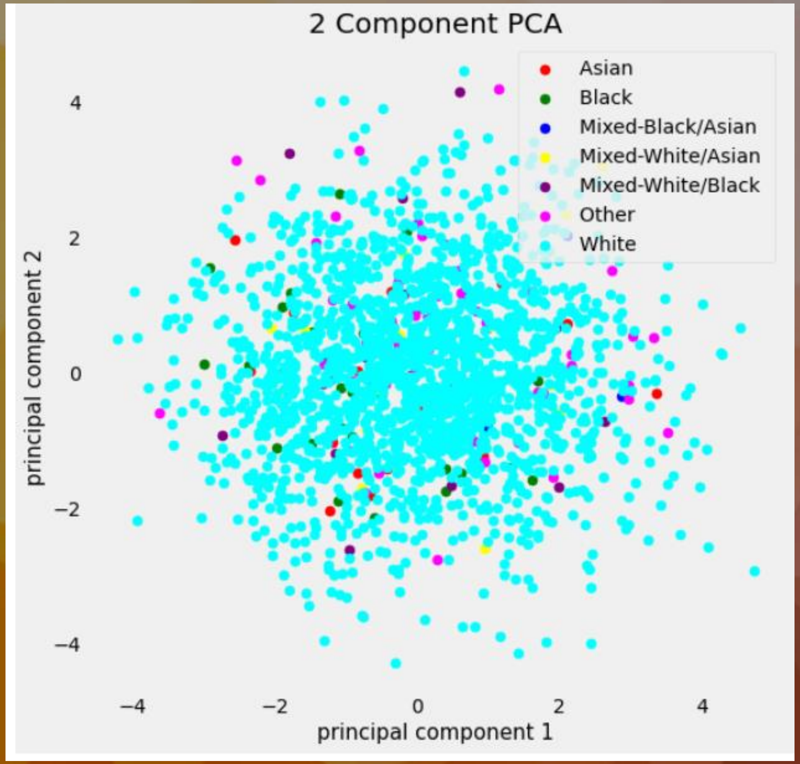
PCA : Graph of Individuals (Country)

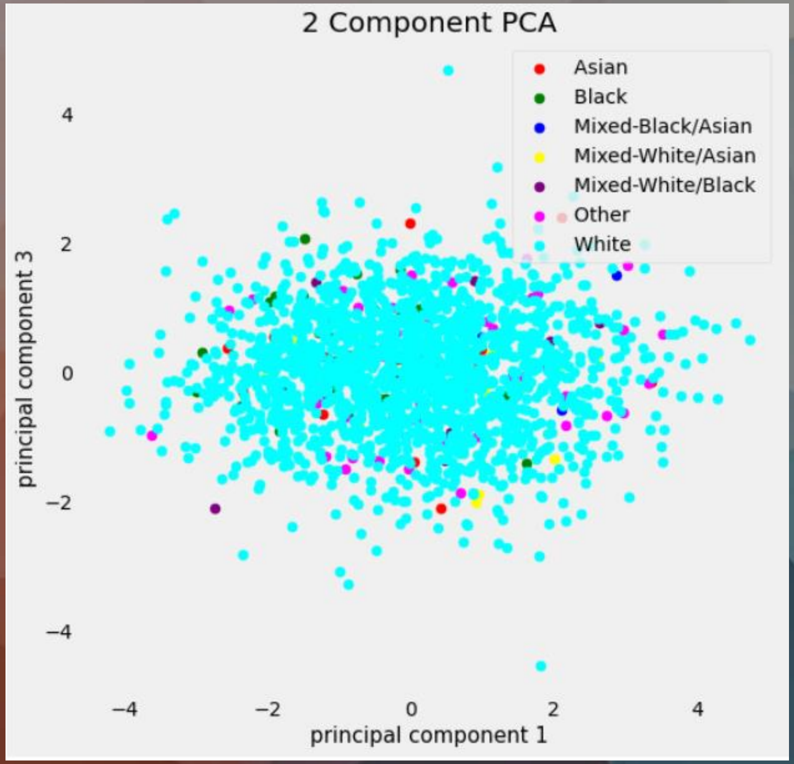


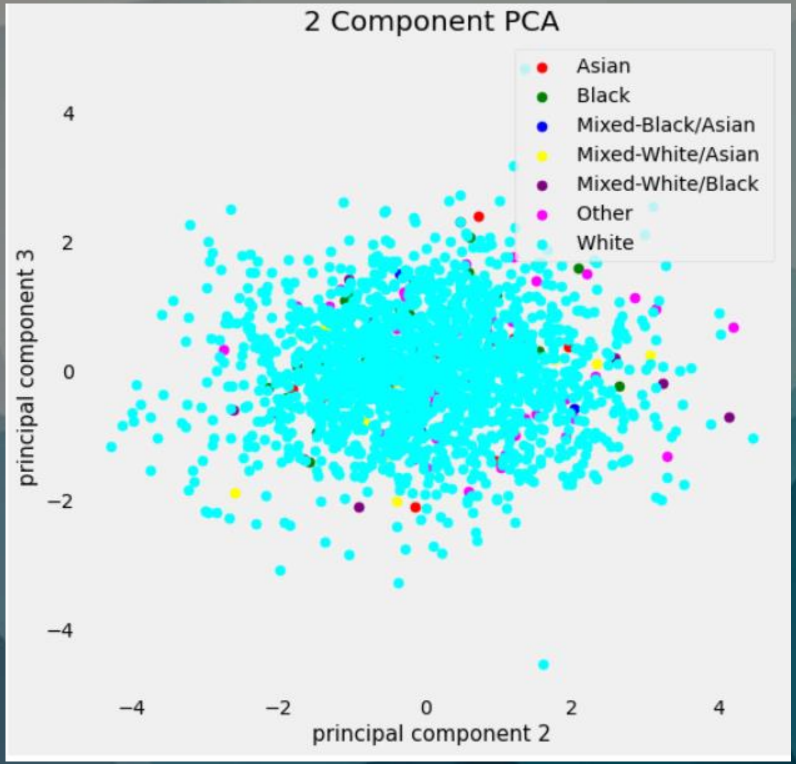




PCA : Graph of Individuals (Ethnicity)







Linear SVC

LSVC : Age

```
Cross-Validation score : 0.347508398656215
```

	precision	recall	f1-score	support
18-24	0.41	0.87	0.56	126
25-34	0.25	0.18	0.21	98
35-44	0.29	0.15	0.20	71
45-54	0.00	0.00	0.00	64
55-64	0.00	0.00	0.00	14
65+	0.00	0.00	0.00	4
accuracy			0.37	377
macro avg	0.16	0.20	0.16	377
weighted avg	0.26	0.37	0.28	377

Confusion Matrix :

```
cm_age = confusion_matrix(ytest_age, yPred_lsvc_age)
print(cm_age)
```

```
[[109  13   4   0   0   0]
 [ 73  18   6   1   0   0]
 [ 37  22  11   1   0   0]
 [ 33  16  15   0   0   0]
 [  9   4   1   0   0   0]
 [  2   0   1   1   0   0]]
```

LSVC : Gender

```
Cross-Validation score : 0.588969764837626
```

	precision	recall	f1-score	support
Female	0.59	0.60	0.60	187
Male	0.60	0.59	0.60	190
accuracy			0.60	377
macro avg	0.60	0.60	0.60	377
weighted avg	0.60	0.60	0.60	377

Confusion Matrix :

```
cm_gender = confusion_matrix(ytest_gender, yPred_lsvc_gender)
print(cm_gender)
```

```
[[113  74]
 [ 77 113]]
```

LSVC : Education

```
Cross-Validation score : 0.31038633818589023
```

	precision	recall	f1-score	support
Doctorate degree	0.00	0.00	0.00	17
Left school at 16 years	0.00	0.00	0.00	20
Left school at 17 years	0.00	0.00	0.00	3
Left school at 18 years	0.00	0.00	0.00	15
Left school before 16 years	0.00	0.00	0.00	4
Masters degree	0.00	0.00	0.00	61
Professional certificate/ diploma	0.00	0.00	0.00	59
Some college or university, no certificate or degree	0.37	0.72	0.49	104
University degree	0.28	0.52	0.37	94
accuracy			0.33	377
macro avg	0.07	0.14	0.09	377
weighted avg	0.17	0.33	0.23	377

Confusion Matrix :

```
cm_education = confusion_matrix(ytest_education, yPred_lsvc_education)
print(cm_education)
```

```
[[ 0  0  0  0  0  0  0  8  9]
 [ 0  0  0  0  0  0  0 11  9]
 [ 0  0  0  0  0  0  0  0  3]
 [ 0  0  0  0  0  0  0  8  7]
 [ 0  0  0  0  0  0  0  0  4]
 [ 0  0  0  0  0  0  0 30 31]
 [ 0  0  0  0  0  0  0 26 33]
 [ 0  0  0  0  0  0  0 75 29]
 [ 0  0  0  0  0  0  0 45 49]]
```

LSVC : Country

```
Cross-Validation score : 0.6101343784994401
```

	precision	recall	f1-score	support
18-24	0.41	0.87	0.56	126
25-34	0.25	0.18	0.21	98
35-44	0.29	0.15	0.20	71
45-54	0.00	0.00	0.00	64
55-64	0.00	0.00	0.00	14
65+	0.00	0.00	0.00	4
accuracy			0.37	377
macro avg	0.16	0.20	0.16	377
weighted avg	0.26	0.37	0.28	377

Confusion Matrix :

```
cm_age = confusion_matrix(ytest_age, yPred_lsvc_age)
print(cm_age)
```

```
[[109  13   4   0   0   0]
 [ 73  18   6   1   0   0]
 [ 37  22  11   1   0   0]
 [ 33  16  15   0   0   0]
 [  9   4   1   0   0   0]
 [  2   0   1   1   0   0]]
```

LSVC : Ethnicity

```
Cross-Validation score : 0.347508398656215
```

	precision	recall	f1-score	support
18-24	0.41	0.87	0.56	126
25-34	0.25	0.18	0.21	98
35-44	0.29	0.15	0.20	71
45-54	0.00	0.00	0.00	64
55-64	0.00	0.00	0.00	14
65+	0.00	0.00	0.00	4
accuracy			0.37	377
macro avg	0.16	0.20	0.16	377
weighted avg	0.26	0.37	0.28	377

Confusion Matrix :

```
cm_age = confusion_matrix(ytest_age, yPred_lsvc_age)
print(cm_age)
```

```
[[109  13   4   0   0   0]
 [ 73  18   6   1   0   0]
 [ 37  22  11   1   0   0]
 [ 33  16  15   0   0   0]
 [  9   4   1   0   0   0]
 [  2   0   1   1   0   0]]
```

kNN Classifier

kNN : Age

```
Cross-Validation score : 0.30509518477043673
```

	precision	recall	f1-score	support
18-24	0.42	0.67	0.52	126
25-34	0.26	0.27	0.26	98
35-44	0.35	0.24	0.28	71
45-54	0.22	0.08	0.11	64
55-64	0.00	0.00	0.00	14
65+	0.00	0.00	0.00	4
accuracy			0.35	377
macro avg	0.21	0.21	0.20	377
weighted avg	0.31	0.35	0.31	377

Confusion Matrix :

```
cm_age = confusion_matrix(ytest_age, yPred_kNN_age)
print(cm_age)
```

```
[[85 30  9  2  0  0]
 [50 26 10 10  1  1]
 [31 20 17  3  0  0]
 [28 20 10  5  1  0]
 [ 8  3  1  2  0  0]
 [ 1  0  2  1  0  0]]
```

kNN : Gender

```
Cross-Validation score : 0.5516517357222844
```

	precision	recall	f1-score	support
Female	0.54	0.49	0.51	187
Male	0.54	0.58	0.56	190
accuracy			0.54	377
macro avg	0.54	0.54	0.53	377
weighted avg	0.54	0.54	0.53	377

Confusion Matrix : ¶

```
cm_gender = confusion_matrix(ytest_gender, yPred_kNN_gender)
print(cm_gender)
```

```
[[ 91  96]
 [ 79 111]]
```

kNN : Education

```
Cross-Validation score : 0.2042273236282194‍6
```

	precision	recall	f1-score	support
18-24	0.42	0.67	0.52	126
25-34	0.26	0.27	0.26	98
35-44	0.35	0.24	0.28	71
45-54	0.22	0.08	0.11	64
55-64	0.00	0.00	0.00	14
65+	0.00	0.00	0.00	4
accuracy			0.35	377
macro avg	0.21	0.21	0.20	377
weighted avg	0.31	0.35	0.31	377

Confusion Matrix :

```
cm_education = confusion_matrix(ytest_education, yPred_kNN_education)
print(cm_education)
```

```
[[ 1  0  0  0  0  2  3  9  2]
 [ 1  1  0  1  0  4  1  7  5]
 [ 0  0  0  0  0  1  1  1  0]
 [ 1  1  0  0  0  3  2  2  6]
 [ 0  0  0  0  0  2  0  1  1]
 [ 6  3  1  1  0 15  4 13 18]
 [ 6  3  2  2  0  7  6 17 16]
 [ 6  2  3  5  0 13 15 43 17]
 [10  9  1  0  0 15  6 35 18]]
```

kNN : Country

```
Cross-Validation score : 0.5227323628219485
```

```
                   precision    recall  f1-score   support

         Australia       0.00      0.00      0.00        14
            Canada       1.00      0.06      0.11        18
             Other       0.00      0.00      0.00        29
Republic of Ireland       0.00      0.00      0.00         3
                UK       0.62      0.80      0.70       211
               USA       0.36      0.34      0.35       102

          accuracy                           0.54       377
         macro avg       0.33      0.20      0.19       377
      weighted avg       0.49      0.54      0.49       377
```

Confusion Matrix :

```
cm_country = confusion_matrix(ytest_country, yPred_kNN_country)
print(cm_country)
```

```
[[  0   0   0   0   7   7]
 [  0   1   0   0  14   3]
 [  0   0   0   0  16  13]
 [  0   0   0   0   1   2]
 [  3   0   4   0 168  36]
 [  0   0   3   0  64  35]]
```

kNN : Ethnicity

```
Cross-Validation score : 0.9071668533034715
```

	precision	recall	f1-score	support
Asian	0.00	0.00	0.00	2
Black	0.00	0.00	0.00	5
Mixed-Black/Asian	0.00	0.00	0.00	1
Mixed-White/Asian	0.00	0.00	0.00	6
Mixed-White/Black	0.00	0.00	0.00	4
Other	0.00	0.00	0.00	17
White	0.91	0.99	0.95	342
accuracy			0.90	377
macro avg	0.13	0.14	0.14	377
weighted avg	0.82	0.90	0.86	377

Confusion Matrix :

```
cm_ethnicity = confusion_matrix(ytest_ethnicity, yPred_kNN_ethnicity)
print(cm_ethnicity)
```

```
[[  0   0   0   0   0   0   2]
 [  0   0   0   0   0   0   5]
 [  0   0   0   0   0   0   1]
 [  0   0   0   0   0   0   6]
 [  0   0   0   0   0   0   4]
 [  0   0   0   0   0   0  17]
 [  1   1   0   0   0   0 340]]
```

Optimized kNN

OPkNN : Age

Getting the best parameters to optimize our kNN model :

```
param_grid={'n_neighbors':np.arange(1,50),'metric':['euclidean','manhattan']}
grid=GridSearchCV(KNeighborsClassifier(),param_grid,cv=5)
model=grid.fit(xtrain_age,np.ravel(ytrain_age))
OPkNN_estimator_age=grid.best_estimator_
OPkNN_estimator_age
```

```
KNeighborsClassifier(metric='euclidean', n_neighbors=24)
```

Getting the best score : The one with (metric='euclidean', n_neighbors=32)

```
CV_OPkNN_age=grid.best_score_
CV_OPkNN_age
```

```
0.37531407449781085
```

OPkNN : Gender

Getting the best parameters to optimize our kNN model :

```
param_grid={'n_neighbors':np.arange(1,100),'metric':['euclidean','manhattan']}
grid=GridSearchCV(KNeighborsClassifier(),param_grid,cv=5)
model=grid.fit(xtrain_gender,np.ravel(ytrain_gender))
OPkNN_estimator_gender=grid.best_estimator_
OPkNN_estimator_gender
```

```
KNeighborsClassifier(metric='euclidean', n_neighbors=89)
```

Getting the best score : The one with (metric='manhattan', n_neighbors=72)

```
CV_OPkNN_gender=grid.best_score_
CV_OPkNN_gender
```

```
0.6525026952102264
```

OPkNN : Education

Getting the best parameters to optimize our kNN model :

```
param_grid={'n_neighbors':np.arange(1,100),'metric':['euclidean','manhattan']}
grid=GridSearchCV(KNeighborsClassifier(),param_grid,cv=5)
model=grid.fit(xtrain_education,np.ravel(ytrain_education))
OPkNN_estimator_education=grid.best_estimator_
OPkNN_estimator_education
```

```
KNeighborsClassifier(metric='manhattan', n_neighbors=73)
```

Getting the best score : The one with (metric='euclidean', n_neighbors=95)

```
CV_OPkNN_education=grid.best_score_
CV_OPkNN_education
```

```
0.334874920243779
```

OPkNN : Country

Getting the best parameters to optimize our kNN model :

```
param_grid={'n_neighbors':np.arange(1,100),'metric':['euclidean','manhattan']}
grid=GridSearchCV(KNeighborsClassifier(),param_grid,cv=5)
model=grid.fit(xtrain_country,np.ravel(ytrain_country))
OPkNN_estimator_country=grid.best_estimator_
OPkNN_estimator_country
```

```
KNeighborsClassifier(metric='euclidean', n_neighbors=90)
```

Getting the best score : The one with (metric='euclidean', n_neighbors=98)

```
CV_OPkNN_country=grid.best_score_
CV_OPkNN_country
```

```
0.6213570658511364
```

OPkNN : Ethnicity

Getting the best parameters to optimize our kNN model :

```
param_grid={'n_neighbors':np.arange(1,100),'metric':['euclidean','manhattan']}
grid=GridSearchCV(KNeighborsClassifier(),param_grid,cv=5)
model=grid.fit(xtrain_ethnicity,np.ravel(ytrain_ethnicity))
OPkNN_estimator_ethnicity=grid.best_estimator_
OPkNN_estimator_ethnicity
```

```
KNeighborsClassifier(metric='euclidean', n_neighbors=10)
```

Getting the best score : The one with (metric='euclidean', n_neighbors=6)

```
CV_OPkNN_ethnicity=grid.best_score_
CV_OPkNN_ethnicity
```

```
0.9137950760159292
```

ML : Comparison Between Models (with 3 PC)

Targets	Linear SVC	kNN	Optimized kNN	Chosen Model
age	0.347508	0.305095	0.375314	OPkNN
gender	0.588970	0.551652	0.652503	OPkNN
education	0.310386	0.204227	0.334875	OPkNN
country	0.610134	0.522732	0.621357	OPkNN
ethnicity	0.907167	0.907167	0.913795	OPkNN

PCA with 4 PCA

	Explained_Variance_Ratio	Cumulated_Variance_Ratio
0	31.583524	31.583524
1	26.291498	57.875023
2	12.887505	70.762528
3	9.650704	80.413231

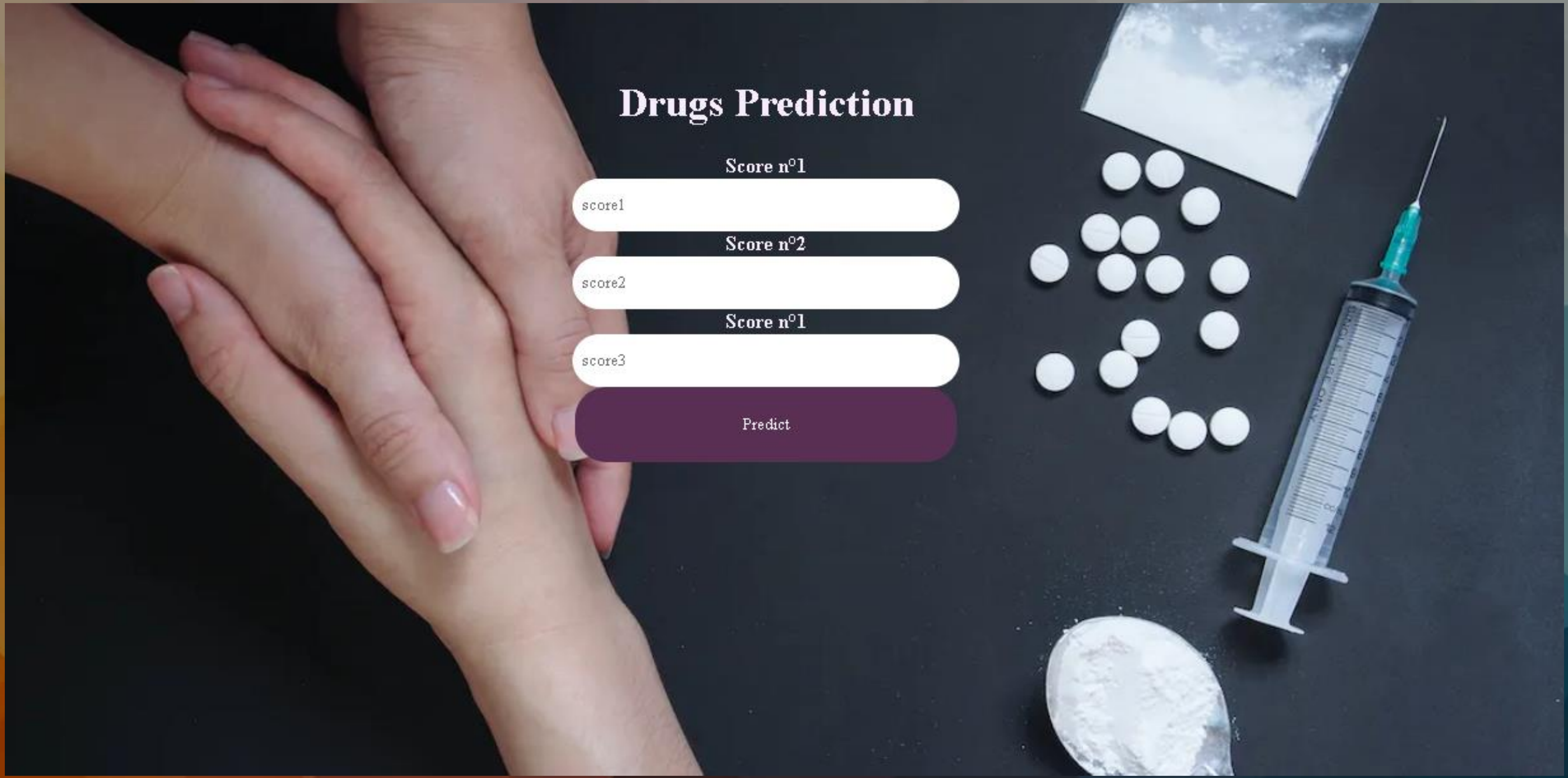
ML : Comparison Between Models (with 4 PC)

Targets	Linear SVC	kNN	Optimized kNN	Chosen Model
age	0.326148	0.339362	0.383281	OPkNN
gender	0.633875	0.541153	0.641231	OPkNN
education	0.297004	0.228135	0.326246	OPkNN
country	0.618085	0.586226	0.612088	LSVC
ethnicity	0.904507	0.904507	0.914457	OPkNN

Algorithm to predict an individual

WEB/API

Frontend of the web page



API code

We save the model using pickle

```
!pip install flask-ngrok
from flask import *
from flask_ngrok import run_with_ngrok
from flask import Flask,request, url_for, redirect, render_template
import pickle
import numpy as np

app = Flask(__name__)
model1=pickle.load(open("/content/finalized_model_age.pkl",'rb'))
model2=pickle.load(open("/content/finalized_model_gender.pkl",'rb'))
model3=pickle.load(open("/content/finalized_model_education.pkl",'rb'))
model4=pickle.load(open("/content/finalized_model_country.pkl",'rb'))
model5=pickle.load(open("/content/finalized_model_ethnicity.pkl",'rb'))
```

API code

```python
@app.route('/')
@app.route('/predict',methods=['POST','GET'])

def predict(score1,score2,score3):
    I=np.array([score1,score2,score3]).reshape(1,3)

    age=model1.predict(I)[0]
    gender=model2.predict(I)[0]
    education=model3.predict(I)[0]
    country=model4.predict(I)[0]
    ethnicity=model5.predict(I)[0]
    s=""
    s+="\n"+"Individual : "+"\n"+"Age : "+age+"\n"+"Gender : "+gender+"\n"+"Education : "+education+"\n"+"Country : "
    return s



if __name__ == '__main__':
    app.run(debug=True)
```

Conclusion :

- Young people tend to be using more drugs.
- Men tend to be using more hard drugs, but it is equal for light drugs, alcohol and nicotine -> explanation : opposite personalities
- Education doesn't seem to make a big difference in drug use but does change for nicotine addiction.
- Cannabis is a gateway into using new drugs.
- Countries where cannabis is legal have more drug consumers.
- Personality does have a big impact on drug consumption; especially openness to experience, impulsiveness and sensation seeing.