assignment3

May 19, 2024

1 Author

- Author: Nicolas Huber
- Mail: nicolassebastian.huber@student.uts.edu.au
- GitHub: HuberNicolas
- Repository: https://github.com/HuberNicolas/python-data-processing-uts

2 Source

https://www.kaggle.com/datasets/mexwell/drug-consumption-classification

[1]:
```
# Imports

import random

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.graph_objects as go
import seaborn as sns
from scipy.stats import pearsonr
```

[2]:
```
# Constants
PATH = './data/drug_consumption.csv'
```

[3]:
```
# Dev variables
DEV = False
DEV_FRACTION = 0.05
RANDOM_STATE = 31011997
np.random.seed(RANDOM_STATE)
```

[4]:
```
# Explicitly define the order of the target column values
CATEGORIES_ORDER = ['cl0', 'cl1', 'cl2', 'cl3', 'cl4', 'cl5', 'cl6']
```

[5]:
```
# Mapping categories based on source
CATEGORIES = {
    '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'
}
```

[6]:
```
COLUMNS_TYPE = {
    'ID': int,
    'Age': float,
    'Gender': float,
    'Education': float,
    'Country': float,
    'Ethnicity': float,
    'Nscore': float,
    'Escore': float,
    'Oscore': float,
    'Ascore': float,
    'Cscore': float,
    'Impulsive': float,
    'SS': float,
    'Alcohol': str,
    'Amphet': str,
    'Amyl': str,
    'Benzos': str,
    'Caff': str,
    'Cannabis': str,
    'Choc': str,
    'Coke': str,
    'Crack': str,
    'Ecstasy': str,
    'Heroin': str,
    'Ketamine': str,
    'Legalh': str,
    'LSD': str,
    'Meth': str,
    'Mushrooms': str,
    'Nicotine': str,
    'Semer': str,
    'VSA': str
}
```

[7]:
```
# Define columns for better clarity
RENAME_COLUMNS = {
    # Big Five
    # https://en.wikipedia.org/wiki/Big_Five_personality_traits
    'Nscore': 'Neuroticism_score',
```

```
    'Escore': 'Extraversion_score',
    'Oscore': 'Openness_score',
    'Ascore': 'Agreeableness_score',
    'Cscore': 'Conscientiousness_score',

    'Impulsive': 'Impulsive_score',
    'SS': 'Sensation_seeing_score'
}
```

3 Preparation: Load data

[8]:
```
# Load data
try:
    df = pd.read_csv(filepath_or_buffer=PATH, header=0)
except FileNotFoundError:
    print('The specified file path does not exist.')
except Exception as e:
    print(f'An unexpected error occurred: {e}')
```

[9]:
```
# For efficiency, using during development only
if DEV:
    # in dev mode, we reduce the data set quantity by sampling 10% to ease␣
 ↪development speed
    df = df.sample(frac=DEV_FRACTION, random_state=RANDOM_STATE)
else:
    pass
```

[10]:
```
# Display top rows
df.head(n=5)
```

[10]:

	ID	Age	Gender	Education	Country	Ethnicity	Nscore	Escore	\
0	1	0.49788	0.48246	-0.05921	0.96082	0.12600	0.31287	-0.57545	
1	2	-0.07854	-0.48246	1.98437	0.96082	-0.31685	-0.67825	1.93886	
2	3	0.49788	-0.48246	-0.05921	0.96082	-0.31685	-0.46725	0.80523	
3	4	-0.95197	0.48246	1.16365	0.96082	-0.31685	-0.14882	-0.80615	
4	5	0.49788	0.48246	1.98437	0.96082	-0.31685	0.73545	-1.63340	

	Oscore	Ascore	…	Ecstasy	Heroin	Ketamine	Legalh	LSD	Meth	\
0	-0.58331	-0.91699	…	CL0	CL0	CL0	CL0	CL0	CL0	
1	1.43533	0.76096	…	CL4	CL0	CL2	CL0	CL2	CL3	
2	-0.84732	-1.62090	…	CL0	CL0	CL0	CL0	CL0	CL0	
3	-0.01928	0.59042	…	CL0	CL0	CL2	CL0	CL0	CL0	
4	-0.45174	-0.30172	…	CL1	CL0	CL0	CL1	CL0	CL0	

	Mushrooms	Nicotine	Semer	VSA
0	CL0	CL2	CL0	CL0

```
1       CL0      CL4   CL0  CL0
2       CL1      CL0   CL0  CL0
3       CL0      CL2   CL0  CL0
4       CL2      CL2   CL0  CL0

[5 rows x 32 columns]
```

[10]:

[11]:
```python
# Define the mapping for age values
age_mapping = {
    -0.95197: '18 - 24',
    -0.07854: '25 - 34',
    0.49788: '35 - 44',
    1.09449: '45 - 54',
    1.82213: '55 - 64',
    2.59171: '65+'
}

# Create the 'age_values' column using the mapping
df['age_values'] = df['Age'].map(age_mapping)
```

[12]:
```python
# Define the mapping for gender values
gender_mapping = {
    0.48246: 'Female',
    -0.48246: 'Male'
}

# Create the 'gender_values' column using the mapping
df['gender_values'] = df['Gender'].map(gender_mapping)
```

[13]:
```python
# Define the mapping for education values
education_mapping = {
    -2.43591: 'Left School Before 16 years',
    -1.73790: 'Left School at 16 years',
    -1.43719: 'Left School at 17 years',
    -1.22751: 'Left School at 18 years',
    -0.61113: 'Some College, No Certificate Or Degree',
    -0.05921: 'Professional Certificate/Diploma',
    0.45468: 'University Degree',
    1.16365: 'Masters Degree',
    1.98437: 'Doctorate Degree'
}

# Create the 'education_values' column using the mapping
df['education_values'] = df['Education'].map(education_mapping)
```

[14]:
```
# Define the mapping for country values
country_mapping = {
    -0.09765: 'Australia',
    0.24923: 'Canada',
    -0.46841: 'New Zealand',
    -0.28519: 'Other',
    0.21128: 'Republic of Ireland',
    0.96082: 'UK',
    -0.57009: 'USA'
}

# Create the 'country_values' column using the mapping
df['country_values'] = df['Country'].map(country_mapping)
```

[15]:
```
# Define the mapping for ethnicity values
ethnicity_mapping = {
    -0.50212: 'Asian',
    -1.10702: 'Black',
    1.90725: 'Mixed-Black/Asian',
    0.12600: 'Mixed-White/Asian',
    -0.22166: 'Mixed-White/Black',
    0.11440: 'Other',
    -0.31685: 'White'
}

# Create the 'ethnicity_values' column using the mapping
df['ethnicity_values'] = df['Ethnicity'].map(ethnicity_mapping)
```

[16]:
```
# Define the mapping for Nscore values
nscore_mapping = {
    -3.46436: 12,
    -3.15735: 13,
    -2.75696: 14,
    -2.52197: 15,
    -2.42317: 16,
    -2.34360: 17,
    -2.21844: 18,
    -2.05048: 19,
    -1.86962: 20,
    -1.69163: 21,
    -1.55078: 22,
    -1.43907: 23,
    -1.32828: 24,
    -1.19430: 25,
    -1.05308: 26,
    -0.92104: 27,
    -0.79151: 28,
```

```
    -0.67825: 29,
    -0.58016: 30,
    -0.46725: 31,
    -0.34799: 32,
    -0.24649: 33,
    -0.14882: 34,
    -0.05188: 35,
    0.04257: 36,
    0.13606: 37,
    0.22393: 38,
    0.31287: 39,
    0.41667: 40,
    0.52135: 41,
    0.62967: 42,
    0.73545: 43,
    0.82562: 44,
    0.91093: 45,
    1.02119: 46,
    1.13281: 47,
    1.23461: 48,
    1.37297: 49,
    1.49158: 50,
    1.60383: 51,
    1.72012: 52,
    1.83990: 53,
    1.98437: 54,
    2.12700: 55,
    2.28554: 56,
    2.46262: 57,
    2.61139: 58,
    2.82196: 59,
    3.27393: 60
}

# Create the 'nscore_values' column using the mapping
df['nscore_values'] = df['Nscore'].map(nscore_mapping)
```

[17]:
```
# Define the mapping for Escore values
escore_mapping = {
    -3.27393: 16,
    -3.00537: 17,
    -2.72827: 19,
    -2.53830: 20,
    -2.44904: 21,
    -2.32338: 22,
    -2.21069: 23,
    -2.11437: 24,
```

```
    -2.03972: 25,
    -1.92173: 26,
    -1.76250: 27,
    -1.63340: 28,
    -1.50796: 29,
    -1.37639: 30,
    -1.23177: 31,
    -1.09207: 32,
    -0.94779: 33,
    -0.80615: 34,
    -0.69509: 35,
    -0.57545: 36,
    -0.43999: 37,
    -0.30033: 38,
    -0.15487: 39,
    0.00332: 40,
    0.16767: 41,
    0.32197: 42,
    0.47617: 43,
    0.63779: 44,
    0.80523: 45,
    0.96248: 46,
    1.11406: 47,
    1.28610: 48,
    1.45421: 49,
    1.58487: 50,
    1.74091: 51,
    1.93886: 52,
    2.12700: 53,
    2.32338: 54,
    2.57309: 55,
    2.85950: 56,
    3.00537: 58,
    3.27393: 59
}

# Create the 'escore_values' column using the mapping
df['escore_values'] = df['Escore'].map(escore_mapping)
```

[18]:
```
# Define the mapping for Oscore values
oscore_mapping = {
    -3.27393: 24,
    -2.85950: 26,
    -2.63199: 28,
    -2.39883: 29,
    -2.21069: 30,
    -2.09015: 31,
```

```
    -1.97495: 32,
    -1.82919: 33,
    -1.68062: 34,
    -1.55521: 35,
    -1.42424: 36,
    -1.27553: 37,
    -1.11902: 38,
    -0.97631: 39,
    -0.84732: 40,
    -0.71727: 41,
    -0.58331: 42,
    -0.45174: 43,
    -0.31776: 44,
    -0.17779: 45,
    -0.01928: 46,
    0.14143: 47,
    0.29338: 48,
    0.44585: 49,
    0.58331: 50,
    0.72330: 51,
    0.88309: 52,
    1.06238: 53,
    1.24033: 54,
    1.43533: 55,
    1.65653: 56,
    1.88511: 57,
    1.15324: 58,
    2.44904: 59,
    2.90161: 60
}

# Create the 'oscore_values' column using the mapping
df['oscore_values'] = df['Oscore'].map(oscore_mapping)
```

[19]:
```
# Define the mapping for Ascore values
ascore_mapping = {
    -3.46436: 12,
    -3.15735: 16,
    -3.00537: 18,
    -2.90161: 23,
    -2.78793: 24,
    -2.70172: 25,
    -2.53830: 26,
    -2.35413: 27,
    -2.21844: 28,
    -2.07848: 29,
    -1.92595: 30,
```

```
    -1.77200: 31,
    -1.62090: 32,
    -1.47955: 33,
    -1.34289: 34,
    -1.21213: 35,
    -1.07533: 36,
    -0.91699: 37,
    -0.76096: 38,
    -0.60633: 39,
    -0.45321: 40,
    -0.30172: 41,
    -0.15487: 42,
    -0.01729: 43,
    0.13136: 44,
    0.28783: 45,
    0.43852: 46,
    0.59042: 47,
    0.76096: 48,
    0.94156: 49,
    1.11406: 50,
    1.28610: 51,
    1.45039: 52,
    1.61108: 53,
    1.81866: 54,
    2.03972: 55,
    2.23427: 56,
    2.46262: 57,
    2.75696: 58,
    3.15735: 59,
    3.46436: 60
}

# Create the 'ascore_values' column using the mapping
df['ascore_values'] = df['Ascore'].map(ascore_mapping)
```

[20]:
```
# Define the mapping for Cscore values
cscore_mapping = {
    -3.46436: 17,
    -3.15735: 19,
    -2.90161: 20,
    -2.72827: 21,
    -2.57309: 22,
    -2.42317: 23,
    -2.30408: 24,
    -2.18109: 25,
    -2.04506: 26,
    -1.92173: 27,
```

```
    -1.78169: 28,
    -1.64101: 29,
    -1.51840: 30,
    -1.38502: 31,
    -1.25773: 32,
    -1.13788: 33,
    -1.01450: 34,
    -0.89891: 35,
    -0.78155: 36,
    -0.65253: 37,
    -0.52745: 38,
    -0.40581: 39,
    -0.27607: 40,
    -0.14277: 41,
    -0.00665: 42,
    0.12331: 43,
    0.25953: 44,
    0.41594: 45,
    0.58489: 46,
    0.75830: 47,
    0.93949: 48,
    1.13407: 49,
    1.30612: 50,
    1.46191: 51,
    1.63088: 52,
    1.81175: 53,
    2.04506: 54,
    2.33337: 55,
    2.63199: 56,
    3.00537: 57,
    3.46436: 59
}

# Create the 'cscore_values' column using the mapping
df['cscore_values'] = df['Cscore'].map(cscore_mapping)
```

[21]:
```
# Define the mapping for impulsiveness values
impulsive_mapping = {
    -2.55524: 20,
    -1.37983: 276,
    -0.71126: 307,
    -0.21712: 355,
    0.19268: 257,
    0.52975: 216,
    0.88113: 195,
    1.29221: 148,
    1.86203: 104,
```

```
    2.90161: 7
}

# Create the 'impulsive_values' column using the mapping
df['impulsive_values'] = df['Impulsive'].map(impulsive_mapping)
```

[22]:
```
# Define the mapping for sensation values
sensation_mapping = {
    -2.07848: 71,
    -1.54858: 87,
    -1.18084: 132,
    -0.84637: 169,
    -0.52593: 211,
    -0.21575: 223,
    0.07987: 219,
    0.40148: 249,
    0.76540: 211,
    1.22470: 210,
    1.92173: 103
}

# Create the 'sensation_values' column using the mapping
df['sensation_values'] = df['SS'].map(sensation_mapping)
```

[23]:
```
# Define the mapping for drug use values
drug_use_mapping = {
    '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'
}

# List of columns that need this mapping
drug_columns = [
    'Alcohol', 'Amphet', 'Amyl', 'Benzos', 'Cannabis', 'Choc', 'Coke', 'Caff',
    'Crack', 'Ecstasy', 'Heroin', 'Ketamine', 'Legalh', 'LSD', 'Meth',
    'Mushrooms', 'Nicotine', 'Semer', 'VSA'
]

# Apply the mapping to each of the drug columns
for col in drug_columns:
    df[f'{col}_values'] = df[col].map(drug_use_mapping)
```

[24]:
```
mappings = {
    'age_values': age_mapping,
    'gender_values': gender_mapping,
    'education_values': education_mapping,
    'country_values': country_mapping,
    'ethnicity_values': ethnicity_mapping
}
```

[25]:
```
df.head()
```

[25]:

	ID	Age	Gender	Education	Country	Ethnicity	Nscore	Escore	\
0	1	0.49788	0.48246	-0.05921	0.96082	0.12600	0.31287	-0.57545	
1	2	-0.07854	-0.48246	1.98437	0.96082	-0.31685	-0.67825	1.93886	
2	3	0.49788	-0.48246	-0.05921	0.96082	-0.31685	-0.46725	0.80523	
3	4	-0.95197	0.48246	1.16365	0.96082	-0.31685	-0.14882	-0.80615	
4	5	0.49788	0.48246	1.98437	0.96082	-0.31685	0.73545	-1.63340	

	Oscore	Ascore	…	Ecstasy_values	Heroin_values	\
0	-0.58331	-0.91699	…	Never Used	Never Used	
1	1.43533	0.76096	…	Used in Last Month	Never Used	
2	-0.84732	-1.62090	…	Never Used	Never Used	
3	-0.01928	0.59042	…	Never Used	Never Used	
4	-0.45174	-0.30172	…	Used over a Decade Ago	Never Used	

	Ketamine_values	Legalh_values	LSD_values	\
0	Never Used	Never Used	Never Used	
1	Used in Last Decade	Never Used	Used in Last Decade	
2	Never Used	Never Used	Never Used	
3	Used in Last Decade	Never Used	Never Used	
4	Never Used	Used over a Decade Ago	Never Used	

	Meth_values	Mushrooms_values	Nicotine_values	\
0	Never Used	Never Used	Used in Last Decade	
1	Used in Last Year	Never Used	Used in Last Month	
2	Never Used	Used over a Decade Ago	Never Used	
3	Never Used	Never Used	Used in Last Decade	
4	Never Used	Used in Last Decade	Used in Last Decade	

	Semer_values	VSA_values
0	Never Used	Never Used
1	Never Used	Never Used
2	Never Used	Never Used
3	Never Used	Never Used
4	Never Used	Never Used

[5 rows x 63 columns]

[26]:
```
df_orig = df.copy(deep=True)
```

[27]:
```
# Creating a list of columns to include
columns_to_include = ['ID'] + list(df.columns[df.columns.get_loc('VSA') + 1:])

# Creating a new DataFrame with the selected columns
df = df[columns_to_include]
print(df)
```

```
        ID age_values gender_values                      education_values  \
0        1    35 - 44        Female       Professional Certificate/Diploma
1        2    25 - 34          Male                      Doctorate Degree
2        3    35 - 44          Male       Professional Certificate/Diploma
3        4    18 - 24        Female                        Masters Degree
4        5    35 - 44        Female                      Doctorate Degree
…      …          …             …                                     …
1880  1884    18 - 24        Female  Some College, No Certificate Or Degree
1881  1885    18 - 24          Male  Some College, No Certificate Or Degree
1882  1886    25 - 34        Female                      University Degree
1883  1887    18 - 24        Female  Some College, No Certificate Or Degree
1884  1888    18 - 24          Male  Some College, No Certificate Or Degree

           country_values   ethnicity_values  nscore_values  escore_values  \
0                      UK  Mixed-White/Asian             39             36
1                      UK              White             29             52
2                      UK              White             31             45
3                      UK              White             34             34
4                      UK              White             43             28
…                     …                …                 …              …
1880                  USA              White             25             51
1881                  USA              White             33             51
1882                  USA              White             47             30
1883                  USA              White             45             26
1884  Republic of Ireland              White             31             53

      oscore_values  ascore_values  …          Ecstasy_values  \
0              42.0             37  …              Never Used
1              55.0             48  …      Used in Last Month
2              40.0             32  …              Never Used
3              46.0             47  …              Never Used
4              43.0             41  …  Used over a Decade Ago
…               …              …  …                       …
1880           57.0             48  …              Never Used
1881           50.0             48  …     Used in Last Decade
1882           37.0             31  …      Used in Last Month
1883           48.0             32  …       Used in Last Year
1884           56.0             50  …       Used in Last Year
```

```
      Heroin_values      Ketamine_values           Legalh_values  \
0        Never Used           Never Used              Never Used
1        Never Used  Used in Last Decade              Never Used
2        Never Used           Never Used              Never Used
3        Never Used  Used in Last Decade              Never Used
4        Never Used           Never Used  Used over a Decade Ago
…             …                   …                       …
1880     Never Used           Never Used       Used in Last Year
1881     Never Used           Never Used       Used in Last Year
1882     Never Used  Used in Last Decade              Never Used
1883     Never Used           Never Used       Used in Last Year
1884     Never Used           Never Used       Used in Last Year

               LSD_values         Meth_values        Mushrooms_values  \
0              Never Used          Never Used              Never Used
1     Used in Last Decade   Used in Last Year              Never Used
2              Never Used          Never Used  Used over a Decade Ago
3              Never Used          Never Used              Never Used
4              Never Used          Never Used     Used in Last Decade
…                     …                   …                       …
1880    Used in Last Year          Never Used              Never Used
1881    Used in Last Week  Used in Last Month      Used in Last Month
1882  Used in Last Decade          Never Used     Used in Last Decade
1883    Used in Last Year          Never Used       Used in Last Year
1884    Used in Last Year          Never Used       Used in Last Year

          Nicotine_values Semer_values           VSA_values
0     Used in Last Decade   Never Used           Never Used
1      Used in Last Month   Never Used           Never Used
2              Never Used   Never Used           Never Used
3     Used in Last Decade   Never Used           Never Used
4     Used in Last Decade   Never Used           Never Used
…                     …         …                     …
1880           Never Used   Never Used    Used in Last Week
1881    Used in Last Week   Never Used           Never Used
1882     Used in Last Day   Never Used           Never Used
1883   Used in Last Month   Never Used           Never Used
1884     Used in Last Day   Never Used  Used in Last Decade

[1885 rows x 32 columns]
```

[28]:
```
NUM_MISSING_COLUMNS = 4
NUM_OUTLIER_COLUMNS = 2

NUM_MISSING_VALUES_PER_ROW = 10
NUM_DUPLICATES_PER_ROW = 10
NUM_OUTLIERS_PER_ROW = 10
```

```
OUTLIERS_VARIATION = 3
```

[29]:
```
# Randomly select columns
missing_columns = random.choices(df.columns, k=NUM_MISSING_COLUMNS)

# Hardcode candidates for missing values to make it more consistent
missing_columns = ['nscore_values', 'escore_values', 'oscore_values',␣
 ↪'ascore_values', 'cscore_values',
                   'impulsive_values', 'sensation_values']
missing_columns
```

[29]:
```
['nscore_values',
 'escore_values',
 'oscore_values',
 'ascore_values',
 'cscore_values',
 'impulsive_values',
 'sensation_values']
```

[30]:
```
outlier_columns = random.choices(
    ['nscore_values', 'escore_values', 'oscore_values', 'ascore_values',␣
 ↪'cscore_values', 'impulsive_values',
     'sensation_values'], k=NUM_OUTLIER_COLUMNS)
outlier_columns
```

[30]:
```
['impulsive_values', 'sensation_values']
```

[31]:
```
def introduce_missing_values(df, columns, num_missing, random_state=None):
    np.random.seed(random_state)
    df_copy = df.copy()

    for col in columns:
        missing_indices = np.random.choice(df_copy.index, num_missing,␣
 ↪replace=False)
        if df_copy[col].dtype == 'object':
            df_copy.loc[missing_indices, col] = ''
        else:
            df_copy.loc[missing_indices, col] = np.nan

    return df_copy
```

[32]:
```
df = introduce_missing_values(df, columns=missing_columns,␣
 ↪num_missing=NUM_MISSING_VALUES_PER_ROW,
                              random_state=RANDOM_STATE)
```

[33]:
```
def introduce_outliers(df, columns, num_outliers, variation=0,␣
 ↪random_state=None):
```

```
    np.random.seed(random_state)
    df_copy = df.copy()

    for col in columns:
        outlier_count = np.random.randint(max(0, num_outliers - variation),␣
↪num_outliers + variation + 1)
        outlier_indices = np.random.choice(df_copy.index, outlier_count,␣
↪replace=False)

        mean = df_copy[col].mean()
        std_dev = df_copy[col].std()

        # Introduce outliers as values far from the mean
        outliers = np.random.choice([mean + 3 * std_dev, mean - 3 * std_dev],␣
↪outlier_count)

        # Cast outliers to the same dtype as the column
        outliers = outliers.astype(df_copy[col].dtype)

        df_copy.loc[outlier_indices, col] = outliers

    return df_copy
```

[34]:
```
df = introduce_outliers(df, columns=outlier_columns,␣
↪num_outliers=NUM_OUTLIERS_PER_ROW, variation=OUTLIERS_VARIATION,
                        random_state=RANDOM_STATE)
```

[35]:
```
df.shape
```

[35]:
```
(1885, 32)
```

[36]:
```
def introduce_duplicate_rows(df, num_duplicates, random_state=None):
    np.random.seed(random_state)
    df_copy = df.copy()

    duplicate_indices = np.random.choice(df_copy.index, num_duplicates)
    duplicate_rows = df_copy.loc[duplicate_indices]
    df_copy = pd.concat([df_copy, duplicate_rows], ignore_index=True)

    return df_copy
```

[37]:
```
df = introduce_duplicate_rows(df, num_duplicates=NUM_DUPLICATES_PER_ROW,␣
↪random_state=RANDOM_STATE)
```

[38]:
```
df.shape
```

[38]:
```
(1895, 32)
```

[39]:
```
df
```

[39]:

```
        ID age_values gender_values                education_values  \
0        1    35 - 44        Female  Professional Certificate/Diploma
1        2    25 - 34          Male                  Doctorate Degree
2        3    35 - 44          Male  Professional Certificate/Diploma
3        4    18 - 24        Female                    Masters Degree
4        5    35 - 44        Female                  Doctorate Degree
…      …         …            …                              …
1890   848    18 - 24        Female  Professional Certificate/Diploma
1891  1407    25 - 34        Female                 University Degree
1892   143    35 - 44          Male           Left School at 17 years
1893    37    35 - 44          Male                 University Degree
1894  1746    25 - 34        Female           Left School at 16 years

     country_values   ethnicity_values  nscore_values  escore_values  \
0                UK  Mixed-White/Asian           39.0           36.0
1                UK              White           29.0           52.0
2                UK              White           31.0           45.0
3                UK              White           34.0           34.0
4                UK              White           43.0           28.0
…              …                 …             …              …
1890         Canada              White           41.0           37.0
1891             UK              White           38.0           36.0
1892             UK              White           31.0           38.0
1893             UK              White           33.0           36.0
1894             UK              White           20.0           46.0

      oscore_values  ascore_values  …          Ecstasy_values  \
0              42.0           37.0  …              Never Used
1              55.0           48.0  …      Used in Last Month
2              40.0           32.0  …              Never Used
3              46.0           47.0  …              Never Used
4              43.0           41.0  …  Used over a Decade Ago
…              …            …  …                       …
1890           35.0           50.0  …       Used in Last Year
1891           35.0           46.0  …              Never Used
1892           37.0           45.0  …              Never Used
1893           45.0           43.0  …              Never Used
1894           39.0           56.0  …              Never Used

      Heroin_values      Ketamine_values           Legalh_values  \
0        Never Used           Never Used              Never Used
1        Never Used  Used in Last Decade              Never Used
2        Never Used           Never Used              Never Used
3        Never Used  Used in Last Decade              Never Used
4        Never Used           Never Used  Used over a Decade Ago
```

```
…              …                    …                        …
1890     Never Used           Never Used               Never Used
1891     Never Used           Never Used               Never Used
1892     Never Used           Never Used               Never Used
1893     Never Used           Never Used               Never Used
1894     Never Used           Never Used               Never Used

                LSD_values        Meth_values        Mushrooms_values  \
0                Never Used         Never Used                Never Used
1     Used in Last Decade  Used in Last Year                Never Used
2                Never Used         Never Used  Used over a Decade Ago
3                Never Used         Never Used                Never Used
4                Never Used         Never Used     Used in Last Decade
…                     …                  …                         …
1890             Never Used         Never Used     Used in Last Decade
1891             Never Used         Never Used                Never Used
1892             Never Used         Never Used                Never Used
1893             Never Used         Never Used                Never Used
1894             Never Used         Never Used                Never Used

          Nicotine_values Semer_values  VSA_values
0     Used in Last Decade   Never Used  Never Used
1      Used in Last Month   Never Used  Never Used
2                Never Used   Never Used  Never Used
3     Used in Last Decade   Never Used  Never Used
4     Used in Last Decade   Never Used  Never Used
…                     …          …          …
1890     Used in Last Day   Never Used  Never Used
1891    Used in Last Year   Never Used  Never Used
1892    Used in Last Year   Never Used  Never Used
1893             Never Used   Never Used  Never Used
1894     Used in Last Day   Never Used  Never Used

[1895 rows x 32 columns]
```

4 TASK 1: Resolve data quality issues using Python and relevant Python packages

4.1 TASK 1.1: Missing values

[40]:
```python
# Function to count NaN and empty string values per column
def count_nulls(column):
    try:
        nan_count = column.isna().sum()
        empty_string_count = (column == '').sum()
        total_nulls = nan_count + empty_string_count
```

```
        return nan_count, empty_string_count, total_nulls
    except Exception as e:
        print(f"Error processing column: {column.name} - {e}")
        return None, None, None
```

[41]:
```
columns_with_missing_values = []

# Apply the function to each column and display results
for column_name in df.columns:
    nan_count, empty_string_count, total_nulls = count_nulls(df[column_name])
    if nan_count is not None and empty_string_count is not None:
        # print(f"{column_name}: NaN: {nan_count}, '': {empty_string_count},␣
 ↪total null values: {total_nulls}") # DEBUG
        if total_nulls > 0:
            columns_with_missing_values.append(column_name)

columns_with_missing_values
```

[41]:
```
['nscore_values',
 'escore_values',
 'oscore_values',
 'ascore_values',
 'cscore_values',
 'impulsive_values',
 'sensation_values']
```

[42]:
```
# Convert empty strings to NaN
df.replace('', np.nan)


# Function to impute missing values with the median
def impute_median(df, columns_with_missing_values):
    for col in columns_with_missing_values:
        # Convert to numeric type if necessary
        df[col] = pd.to_numeric(df[col], errors='coerce')
        median = df[col].median()
        df[col] = df[col].fillna(median)
    return df
```

[43]:
```
# Apply the function to the DataFrame
df = impute_median(df, columns_with_missing_values)
df
```

[43]:
```
        ID age_values gender_values                  education_values  \
0        1    35 - 44        Female  Professional Certificate/Diploma
1        2    25 - 34          Male                  Doctorate Degree
2        3    35 - 44          Male  Professional Certificate/Diploma
```

```
3        4    18 - 24        Female                    Masters Degree
4        5    35 - 44        Female                  Doctorate Degree
…    …        …           …                                    …
1890   848    18 - 24        Female  Professional Certificate/Diploma
1891  1407    25 - 34        Female                 University Degree
1892   143    35 - 44          Male           Left School at 17 years
1893    37    35 - 44          Male                 University Degree
1894  1746    25 - 34        Female           Left School at 16 years

     country_values   ethnicity_values  nscore_values  escore_values  \
0                UK  Mixed-White/Asian           39.0           36.0
1                UK              White           29.0           52.0
2                UK              White           31.0           45.0
3                UK              White           34.0           34.0
4                UK              White           43.0           28.0
…              …                 …             …            …
1890         Canada              White           41.0           37.0
1891             UK              White           38.0           36.0
1892             UK              White           31.0           38.0
1893             UK              White           33.0           36.0
1894             UK              White           20.0           46.0

      oscore_values  ascore_values  …          Ecstasy_values  \
0              42.0           37.0  …              Never Used
1              55.0           48.0  …      Used in Last Month
2              40.0           32.0  …              Never Used
3              46.0           47.0  …              Never Used
4              43.0           41.0  …  Used over a Decade Ago
…              …              …     …                    …
1890           35.0           50.0  …       Used in Last Year
1891           35.0           46.0  …              Never Used
1892           37.0           45.0  …              Never Used
1893           45.0           43.0  …              Never Used
1894           39.0           56.0  …              Never Used

      Heroin_values      Ketamine_values           Legalh_values  \
0        Never Used           Never Used              Never Used
1        Never Used  Used in Last Decade              Never Used
2        Never Used           Never Used              Never Used
3        Never Used  Used in Last Decade              Never Used
4        Never Used           Never Used  Used over a Decade Ago
…              …                 …                       …
1890     Never Used           Never Used              Never Used
1891     Never Used           Never Used              Never Used
1892     Never Used           Never Used              Never Used
1893     Never Used           Never Used              Never Used
1894     Never Used           Never Used              Never Used
```

```
                LSD_values        Meth_values        Mushrooms_values  \
0                Never Used         Never Used              Never Used
1     Used in Last Decade  Used in Last Year              Never Used
2                Never Used         Never Used  Used over a Decade Ago
3                Never Used         Never Used              Never Used
4                Never Used         Never Used     Used in Last Decade
…                     …                  …                       …
1890             Never Used         Never Used     Used in Last Decade
1891             Never Used         Never Used              Never Used
1892             Never Used         Never Used              Never Used
1893             Never Used         Never Used              Never Used
1894             Never Used         Never Used              Never Used

          Nicotine_values Semer_values  VSA_values
0     Used in Last Decade   Never Used  Never Used
1      Used in Last Month   Never Used  Never Used
2                Never Used   Never Used  Never Used
3     Used in Last Decade   Never Used  Never Used
4     Used in Last Decade   Never Used  Never Used
…                     …          …          …
1890     Used in Last Day   Never Used  Never Used
1891    Used in Last Year   Never Used  Never Used
1892    Used in Last Year   Never Used  Never Used
1893             Never Used   Never Used  Never Used
1894     Used in Last Day   Never Used  Never Used

[1895 rows x 32 columns]
```

4.2 TASK 1.2: Duplicate values

[44]:
```
# Find duplicate rows
duplicate_rows = df.duplicated()
```

[45]:
```
# Displaying duplicate
if duplicate_rows.any():
    # Print all duplicate rows if there are any
    print('Duplicate Rows Found:')
    print(df[duplicate_rows])

    # Remove duplicate rows
    df = df.drop_duplicates()
    print('Duplicates removed.')
    print(df.shape)
else:
    # Inform the user that no duplicate rows were found
```

```
    print('No duplicate rows were found in the data.')  # Detailed and␣
↪informative display of key dataset information.
```

```
Duplicate Rows Found:
        ID age_values gender_values                      education_values  \
1885   528    55 - 64          Male                 Left School at 18 years
1886   230    45 - 54        Female                       University Degree
1887   613    25 - 34          Male                       University Degree
1888   831    18 - 24          Male  Some College, No Certificate Or Degree
1889   566    45 - 54        Female  Some College, No Certificate Or Degree
1890   848    18 - 24        Female       Professional Certificate/Diploma
1891  1407    25 - 34        Female                       University Degree
1892   143    35 - 44          Male                 Left School at 17 years
1893    37    35 - 44          Male                       University Degree
1894  1746    25 - 34        Female                 Left School at 16 years

     country_values ethnicity_values  nscore_values  escore_values  \
1885             UK             White           20.0           43.0
1886             UK             White           26.0           49.0
1887             UK             White           29.0           42.0
1888            USA             White           37.0           44.0
1889             UK             White           31.0           37.0
1890         Canada             White           41.0           37.0
1891             UK             White           38.0           36.0
1892             UK             White           31.0           38.0
1893             UK             White           33.0           36.0
1894             UK             White           20.0           46.0

      oscore_values  ascore_values  …     Ecstasy_values  Heroin_values  \
1885           38.0           43.0  …         Never Used     Never Used
1886           47.0           49.0  …         Never Used     Never Used
1887           42.0           51.0  …         Never Used     Never Used
1888           57.0           51.0  …  Used in Last Year     Never Used
1889           46.0           44.0  …         Never Used     Never Used
1890           35.0           50.0  …  Used in Last Year     Never Used
1891           35.0           46.0  …         Never Used     Never Used
1892           37.0           45.0  …         Never Used     Never Used
1893           45.0           43.0  …         Never Used     Never Used
1894           39.0           56.0  …         Never Used     Never Used

      Ketamine_values      Legalh_values         LSD_values Meth_values  \
1885       Never Used         Never Used         Never Used  Never Used
1886       Never Used         Never Used         Never Used  Never Used
1887       Never Used  Used in Last Year         Never Used  Never Used
1888       Never Used         Never Used  Used in Last Week  Never Used
1889       Never Used         Never Used         Never Used  Never Used
1890       Never Used         Never Used         Never Used  Never Used
1891       Never Used         Never Used         Never Used  Never Used
```

```
1892       Never Used         Never Used         Never Used  Never Used
1893       Never Used         Never Used         Never Used  Never Used
1894       Never Used         Never Used         Never Used  Never Used

            Mushrooms_values         Nicotine_values Semer_values  \
1885              Never Used  Used over a Decade Ago   Never Used
1886              Never Used              Never Used   Never Used
1887     Used in Last Decade        Used in Last Day   Never Used
1888              Never Used      Used in Last Month   Never Used
1889  Used over a Decade Ago        Used in Last Day   Never Used
1890     Used in Last Decade        Used in Last Day   Never Used
1891              Never Used       Used in Last Year   Never Used
1892              Never Used       Used in Last Year   Never Used
1893              Never Used              Never Used   Never Used
1894              Never Used        Used in Last Day   Never Used

                  VSA_values
1885              Never Used
1886              Never Used
1887     Used in Last Decade
1888              Never Used
1889  Used over a Decade Ago
1890              Never Used
1891              Never Used
1892              Never Used
1893              Never Used
1894              Never Used

[10 rows x 32 columns]
Duplicates removed.
(1885, 32)
```

4.3 TASK 1.3: Outliers

[46]:
```python
# Function to detect outliers using the IQR method
def detect_outliers(column):
    if pd.api.types.is_numeric_dtype(column):
        Q1 = column.quantile(0.25)
        Q3 = column.quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        return (column < lower_bound) | (column > upper_bound)
    return pd.Series([False] * len(column))
```

[47]:
```python
# Function to impute outliers with the median
def impute_outliers_with_median(df, columns_with_outliers):
```

```
    for col in columns_with_outliers:
        median = df[col].median()
        outliers = detect_outliers(df[col])
        df.loc[outliers, col] = median
        print(f"Imputed outliers in column '{col}' with the median value:␣
 ↪{median}")
    return df
```

[48]:
```
# Main function to iterate over columns and process outliers
def process_outliers(df):
    columns_with_outliers = []

    for col in df.columns:
        if detect_outliers(df[col]).any():
            columns_with_outliers.append(col)
            print(f"Outliers detected in column '{col}'")
        else:
            print(f"No outliers found in column '{col}'")

    if columns_with_outliers:
        pass
        # df = impute_outliers_with_median(df, columns_with_outliers)
    else:
        print("No outliers to impute.")

    return df
```

[49]:
```
df = process_outliers(df)
```

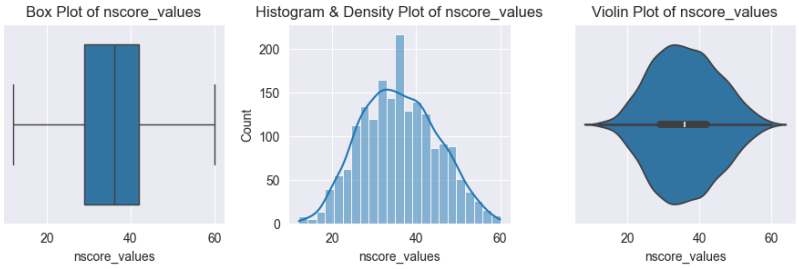
```
No outliers found in column 'ID'
No outliers found in column 'age_values'
No outliers found in column 'gender_values'
No outliers found in column 'education_values'
No outliers found in column 'country_values'
No outliers found in column 'ethnicity_values'
No outliers found in column 'nscore_values'
Outliers detected in column 'escore_values'
Outliers detected in column 'oscore_values'
Outliers detected in column 'ascore_values'
Outliers detected in column 'cscore_values'
Outliers detected in column 'impulsive_values'
Outliers detected in column 'sensation_values'
No outliers found in column 'Alcohol_values'
No outliers found in column 'Amphet_values'
No outliers found in column 'Amyl_values'
No outliers found in column 'Benzos_values'
No outliers found in column 'Cannabis_values'
No outliers found in column 'Choc_values'
```

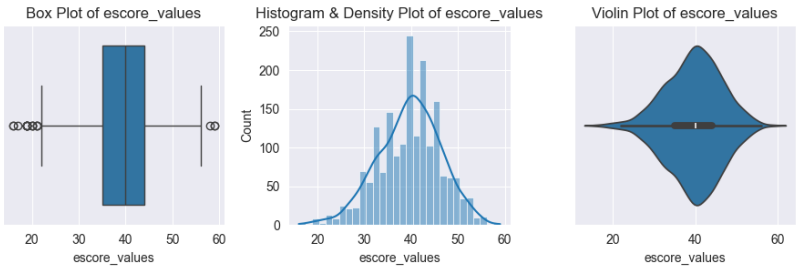
```
No outliers found in column 'Coke_values'
No outliers found in column 'Caff_values'
No outliers found in column 'Crack_values'
No outliers found in column 'Ecstasy_values'
No outliers found in column 'Heroin_values'
No outliers found in column 'Ketamine_values'
No outliers found in column 'Legalh_values'
No outliers found in column 'LSD_values'
No outliers found in column 'Meth_values'
No outliers found in column 'Mushrooms_values'
No outliers found in column 'Nicotine_values'
No outliers found in column 'Semer_values'
No outliers found in column 'VSA_values'
```

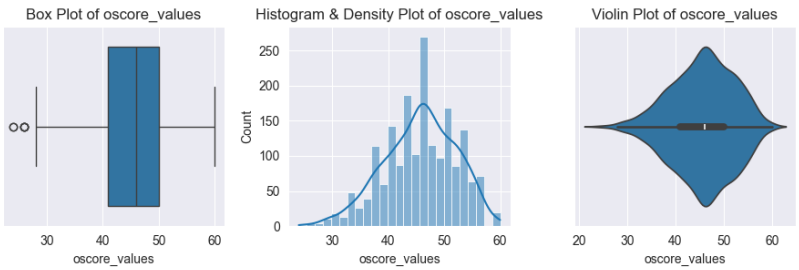
[50]:
```
# Get the list of numeric columns
numeric_df = df.select_dtypes(include=[np.number])
numeric_columns = numeric_df.columns.tolist()

# Get the list of categorical columns
categorical_columns = ['age_values', 'gender_values', 'education_values',␣
 ↪'country_values', 'ethnicity_values']

# Remove ID column since this is not considered to be an attribute
numeric_columns.remove('ID')
```

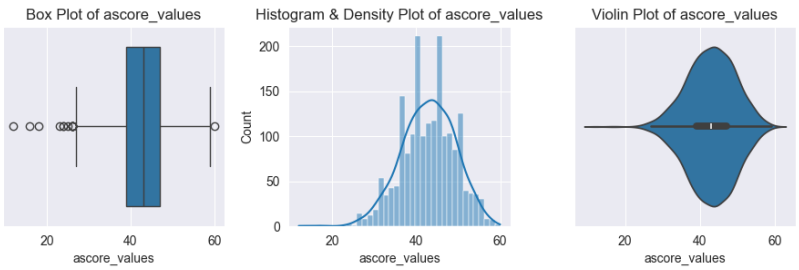
[51]:
```
# Set up the subplots grid
num_plots = len(numeric_columns)
num_cols = 3  # Three types of plots for each column: Box, Histogram & Density,␣
 ↪Violin

# Calculate the number of rows needed for all types of plots for each column
num_rows = len(numeric_columns)  # One row for each column type of plot

fig, axes = plt.subplots(num_rows, num_cols, figsize=(num_cols * 3, num_rows *␣
 ↪3))

# Flatten the axes array to make iteration easier
axes = axes.flatten()

# Iterate over each numeric column and plot each type in a separate cell
for i, col in enumerate(numeric_columns):
    # Box Plot
    sns.boxplot(x=df[col], ax=axes[i * num_cols])
    axes[i * num_cols].set_title(f'Box Plot of {col}')

    # Histogram & Density Plot
    sns.histplot(df[col], kde=True, ax=axes[i * num_cols + 1])
```

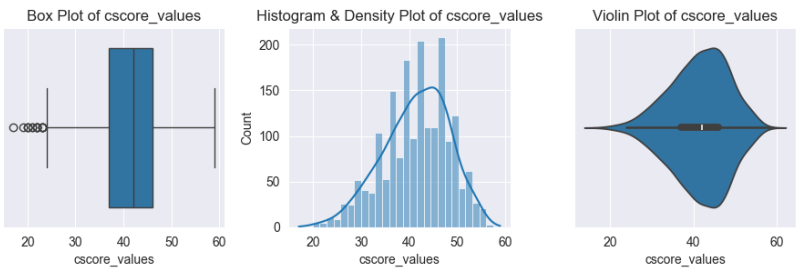
```
    axes[i * num_cols + 1].set_title(f'Histogram & Density Plot of {col}')

    # Violin Plot
    sns.violinplot(x=df[col], ax=axes[i * num_cols + 2])
    axes[i * num_cols + 2].set_title(f'Violin Plot of {col}')

# Adjust layout
plt.tight_layout()

# Show plots
plt.show()  # Detailed and informative display of key dataset information.
```

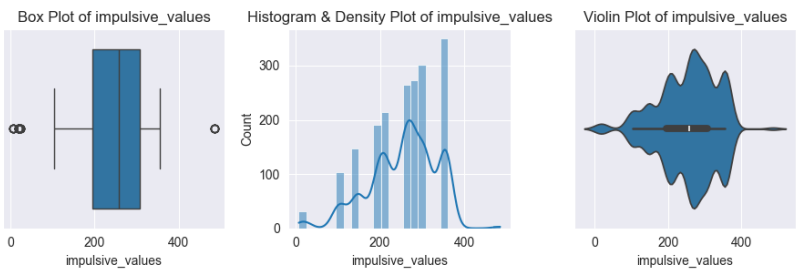

Box Plot of nscore_values
nscore_values
Histogram & Density Plot of nscore_values
Count
nscore_values
Violin Plot of nscore_values
nscore_values

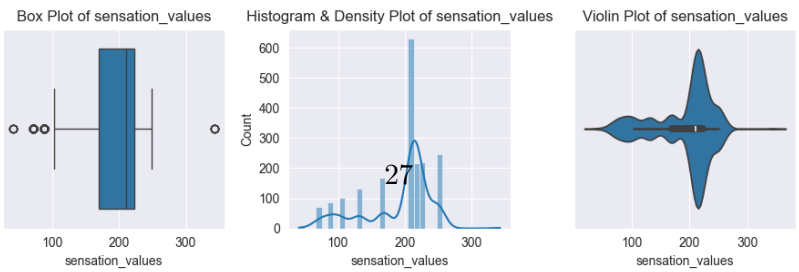

Box Plot of escore_values
escore_values
Histogram & Density Plot of escore_values
Count
escore_values
Violin Plot of escore_values
escore_values


Box Plot of oscore_values
oscore_values
Histogram & Density Plot of oscore_values
Count
oscore_values
Violin Plot of oscore_values
oscore_values


Box Plot of ascore_values
ascore_values
Histogram & Density Plot of ascore_values
Count
ascore_values
Violin Plot of ascore_values
ascore_values


Box Plot of cscore_values
cscore_values
Histogram & Density Plot of cscore_values
Count
cscore_values
Violin Plot of cscore_values
cscore_values


Box Plot of impulsive_values
impulsive_values
Histogram & Density Plot of impulsive_values
Count
impulsive_values
Violin Plot of impulsive_values
impulsive_values


Box Plot of sensation_values
sensation_values
Histogram & Density Plot of sensation_values
Count
sensation_values
Violin Plot of sensation_values
sensation_values

[52]:
```
def analyze_outliers(df):
    outlier_counts = {}

    for col in df.columns:
        if df[col].dtype in ['int64', 'float64']:  # Check if the column is␣
↪numeric
            # Calculate the first and third quartiles
            Q1 = df[col].quantile(0.25)
            Q3 = df[col].quantile(0.75)
            IQR = Q3 - Q1  # Interquartile range

            # Define the bounds for outliers
            lower_bound = Q1 - 1.5 * IQR
            upper_bound = Q3 + 1.5 * IQR

            # Count outliers using the defined bounds
            outliers = df[(df[col] < lower_bound) | (df[col] >␣
↪upper_bound)][col].count()

            # Store the count of outliers in the dictionary
            outlier_counts[col] = outliers

    # Optionally, print the count of outliers for each column
    for col, count in outlier_counts.items():
        print(f"Number of outliers in {col}: {count}")

    return outlier_counts
```

[53]:
```
outlier_counts = analyze_outliers(df)
```

```
Number of outliers in ID: 0
Number of outliers in nscore_values: 0
Number of outliers in escore_values: 18
Number of outliers in oscore_values: 6
Number of outliers in ascore_values: 15
Number of outliers in cscore_values: 17
Number of outliers in impulsive_values: 39
Number of outliers in sensation_values: 168
```

[54]:
```
def remove_outliers(df, column_name, remove='both'):
    # Calculate the first and third quartiles
    Q1 = df[column_name].quantile(0.25)
    Q3 = df[column_name].quantile(0.75)
    IQR = Q3 - Q1  # Interquartile range
```

```
    # Define the bounds for outliers
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    if remove == 'both':
        # Remove both upper and lower outliers
        return df[(df[column_name] >= lower_bound) & (df[column_name] <=␣
↪upper_bound)]
    elif remove == 'upper':
        # Remove only upper outliers
        return df[df[column_name] <= upper_bound]
    elif remove == 'lower':
        # Remove only lower outliers
        return df[df[column_name] >= lower_bound]
    else:
        raise ValueError("Invalid value for remove. Use 'both', 'upper', or␣
↪'lower'.")
```

[55]:
```
df = remove_outliers(df, 'oscore_values', remove='lower')
```

[56]:
```
outlier_counts = analyze_outliers(df)
```

```
Number of outliers in ID: 0
Number of outliers in nscore_values: 0
Number of outliers in escore_values: 18
Number of outliers in oscore_values: 0
Number of outliers in ascore_values: 15
Number of outliers in cscore_values: 17
Number of outliers in impulsive_values: 39
Number of outliers in sensation_values: 164
```

[57]:
```
# Advanced data exploration
non_numeric_stats = {}

for column in df.columns:
    data = df[column]

    print(column, data.dtype, pd.api.types.is_string_dtype(data.dtype))

    # Check if the column is numeric
    if pd.api.types.is_numeric_dtype(data.dtype):
        pass
    else:
        # Calculate value counts for each category in the column
        value_counts = data.value_counts()
        # Store the counts in the dictionary, ensuring all possible categories␣
↪are represented
```

```
        non_numeric_stats[column] = value_counts
```

```
ID int64 False
age_values object True
gender_values object True
education_values object True
country_values object True
ethnicity_values object True
nscore_values float64 False
escore_values float64 False
oscore_values float64 False
ascore_values float64 False
cscore_values float64 False
impulsive_values float64 False
sensation_values float64 False
Alcohol_values object True
Amphet_values object True
Amyl_values object True
Benzos_values object True
Cannabis_values object True
Choc_values object True
Coke_values object True
Caff_values object True
Crack_values object True
Ecstasy_values object True
Heroin_values object True
Ketamine_values object True
Legalh_values object True
LSD_values object True
Meth_values object True
Mushrooms_values object True
Nicotine_values object True
Semer_values object True
VSA_values object True
```

[58]:
```
df.head()
```

[58]:

	ID	age_values	gender_values	education_values	country_values	ethnicity_values	nscore_values	escore_values
0	1	35 - 44	Female	Professional Certificate/Diploma	UK	Mixed-White/Asian	39.0	36.0
1	2	25 - 34	Male	Doctorate Degree	UK	White	29.0	52.0
2	3	35 - 44	Male	Professional Certificate/Diploma	UK	White	31.0	45.0

```
3             UK              White           34.0           34.0
4             UK              White           43.0           28.0

   oscore_values  ascore_values  …          Ecstasy_values  Heroin_values  \
0           42.0           37.0  …              Never Used     Never Used
1           55.0           48.0  …     Used in Last Month     Never Used
2           40.0           32.0  …              Never Used     Never Used
3           46.0           47.0  …              Never Used     Never Used
4           43.0           41.0  …  Used over a Decade Ago     Never Used

       Ketamine_values           Legalh_values           LSD_values  \
0           Never Used              Never Used           Never Used
1  Used in Last Decade              Never Used  Used in Last Decade
2           Never Used              Never Used           Never Used
3  Used in Last Decade              Never Used           Never Used
4           Never Used  Used over a Decade Ago           Never Used

         Meth_values        Mushrooms_values      Nicotine_values  \
0         Never Used              Never Used  Used in Last Decade
1  Used in Last Year              Never Used   Used in Last Month
2         Never Used  Used over a Decade Ago           Never Used
3         Never Used              Never Used  Used in Last Decade
4         Never Used     Used in Last Decade  Used in Last Decade

  Semer_values  VSA_values
0   Never Used  Never Used
1   Never Used  Never Used
2   Never Used  Never Used
3   Never Used  Never Used
4   Never Used  Never Used

[5 rows x 32 columns]
```

[59]:
```python
def plot_categorical_data(df, column, mappings, colors='viridis'):
    # This took me like 2 hours :')
    if column in mappings:
        # Convert column to a categorical type with the specified order
        category_order = mappings[column]
        df[column] = pd.Categorical(df[column], categories=category_order.
 ↪values(), ordered=True)

    counts = df[column].value_counts().sort_index()  # Get counts of each␣
 ↪category in the specified order
    # print(counts)  # Print counts for verification

    if not counts.empty:
        # Create a bar chart using Plotly
```

```
        fig = go.Figure([go.Bar(x=counts.index, y=counts.values, text=counts.
↪values,
                                textposition='auto', marker_color=counts.
↪values, marker=dict(colorscale=colors))])
        fig.update_layout(title=f'Frequency Count for {column}',
                          xaxis_title='Category',
                          yaxis_title='Counts',
                          template='plotly_white')
        fig.show()
    else:
        print(f"No data to plot for {column}")
```

[81]:
```
for column in categorical_columns:
    plot_categorical_data(df, column, mappings)
```

4.4 TASK 1.4: Dataset verification

[61]:
```
def sanity_check(df):
    # Check for null/NaN values
    null_check = df.isnull().sum()
    null_columns = null_check[null_check > 0]

    if null_columns.empty:
        print("No null or NaN values found in the DataFrame.")
    else:
        print("Columns with null or NaN values found:")
        print(null_columns)

    # Check for outliers in each column
    for col in df.columns:
        outliers = detect_outliers(df[col])
        if outliers.any():
            print(f"Outliers found in column '{col}'.")
        else:
            print(f"No outliers found in column '{col}'.")

    # Check for duplicate rows
    if df.duplicated().any():
        print("Duplicate rows found in the DataFrame.")
    else:
        print("No duplicate rows found in the DataFrame.")

    # Check for duplicate columns
    duplicate_columns = df.columns[df.columns.duplicated()]
    if duplicate_columns.any():
        print(f"Duplicate columns found: {list(duplicate_columns)}")
    else:
```

```
        print("No duplicate columns found in the DataFrame.")
```

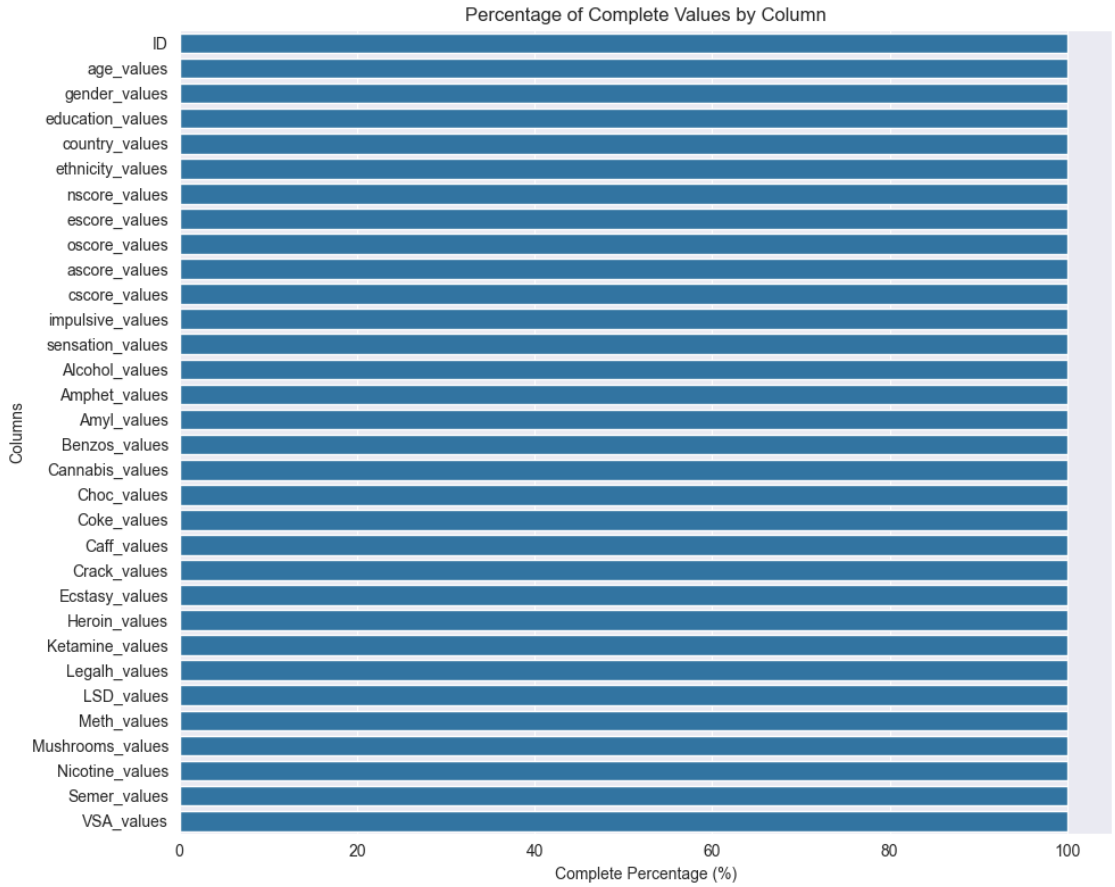
[62]:
```
sanity_check(df)
```

```
No null or NaN values found in the DataFrame.
No outliers found in column 'ID'.
No outliers found in column 'age_values'.
No outliers found in column 'gender_values'.
No outliers found in column 'education_values'.
No outliers found in column 'country_values'.
No outliers found in column 'ethnicity_values'.
No outliers found in column 'nscore_values'.
Outliers found in column 'escore_values'.
No outliers found in column 'oscore_values'.
Outliers found in column 'ascore_values'.
Outliers found in column 'cscore_values'.
Outliers found in column 'impulsive_values'.
Outliers found in column 'sensation_values'.
No outliers found in column 'Alcohol_values'.
No outliers found in column 'Amphet_values'.
No outliers found in column 'Amyl_values'.
No outliers found in column 'Benzos_values'.
No outliers found in column 'Cannabis_values'.
No outliers found in column 'Choc_values'.
No outliers found in column 'Coke_values'.
No outliers found in column 'Caff_values'.
No outliers found in column 'Crack_values'.
No outliers found in column 'Ecstasy_values'.
No outliers found in column 'Heroin_values'.
No outliers found in column 'Ketamine_values'.
No outliers found in column 'Legalh_values'.
No outliers found in column 'LSD_values'.
No outliers found in column 'Meth_values'.
No outliers found in column 'Mushrooms_values'.
No outliers found in column 'Nicotine_values'.
No outliers found in column 'Semer_values'.
No outliers found in column 'VSA_values'.
No duplicate rows found in the DataFrame.
No duplicate columns found in the DataFrame.
```

[63]:
```
# Comprehensive use of techniques in identifying missing values

# Prepare data for the plot
complete_data = []

for column in df.columns:
    try:
        total_count = len(df[column])
```

```
        missing_count = df[column].isna().sum()

        # Check for division by zero
        if total_count == 0:
            raise ValueError(f'No data available in column {column}')

        missing_percentage = (missing_count / total_count) * 100
        complete_percentage = 100 - missing_percentage

        complete_data.append({'Column': column, 'Complete Percentage':␣
↪complete_percentage})

    except ValueError as ve:
        print(f'Error in column {column}: {ve}')
    except Exception as e:
        print(f'An unexpected error occurred for column {column}: {e}')
```

[64]:
```
# Convert the list of dictionaries into a DataFrame for plotting
complete_df = pd.DataFrame(complete_data)

# Plotting
plt.figure(figsize=(10, 8))
sns.barplot(x='Complete Percentage', y='Column',
            data=complete_df)  # https://seaborn.pydata.org/generated/seaborn.
↪barplot.html
plt.title('Percentage of Complete Values by Column')
plt.xlabel('Complete Percentage (%)')
plt.ylabel('Columns')
plt.tight_layout()
plt.show()  # Detailed and informative display of key dataset information.
```



[65]:
```
df.to_csv('drug_consumption_preprocessed.csv')
```

5 TASK 2: You must specify and answer three questions using appropriate data visualisation techniques

Q1: **What is the distribution of drug usage frequency among different age groups?** - Context: Understanding how drug usage varies across different age demographics can help tailor prevention and intervention programs. - Relevance: Age-specific trends can inform policy-making and targeted health campaigns. - Visualization Technique: Bar Chart - This will clearly show the frequency of drug usage across different age groups, making it easy to compare and identify trends.

Q2: **Is there a correlation between personality traits and drug usage frequency?** - Context: Examining if certain personality traits are associated with higher or lower frequencies of drug usage can provide insights for psychological and medical professionals. - Relevance: Helps in identifying at-risk individuals based on their personality profiles. - Visualization Technique: Scatter Plot Matrix (Pairs Plot) - This will allow for the visualization of relationships between multiple personality traits and drug usage frequencies simultaneously.

Q3: **What is the gender distribution in the frequency of drug usage?** - Context: Under-

standing the gender-based differences in drug usage can help in developing gender-specific interventions and support programs. - Relevance: Gender-targeted strategies can be more effective in addressing drug-related issues. - Visualization Technique: Pie Chart or Stacked Bar Chart - A pie chart can show the overall distribution, while a stacked bar chart can compare the frequency of usage across genders for each drug.

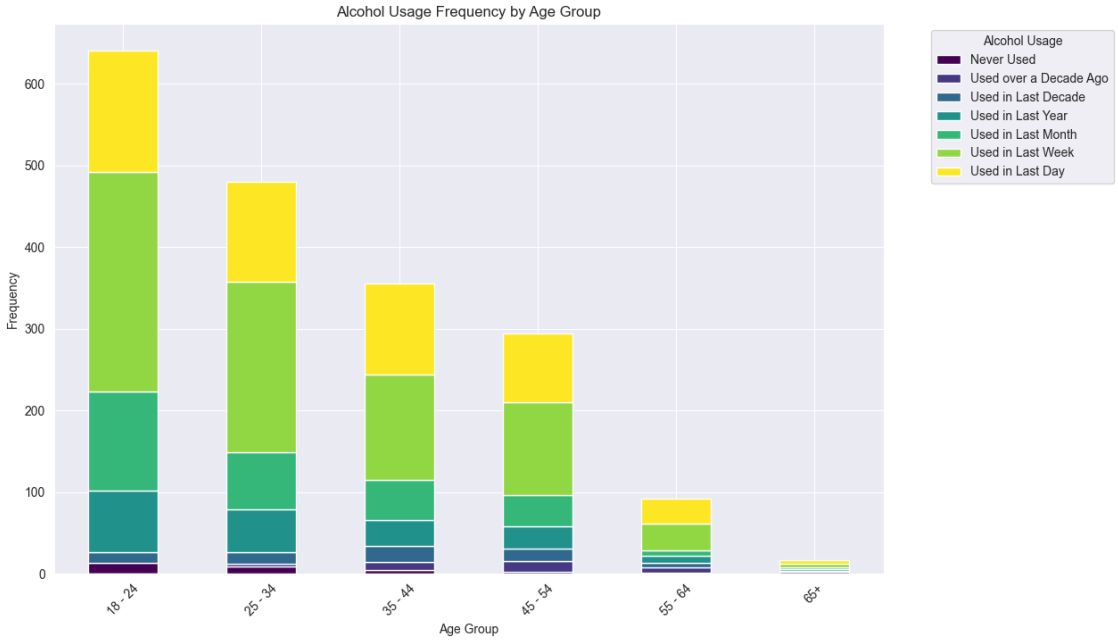
5.1 TASK 2.1: Q1 - What is the distribution of drug usage frequency among different age groups?

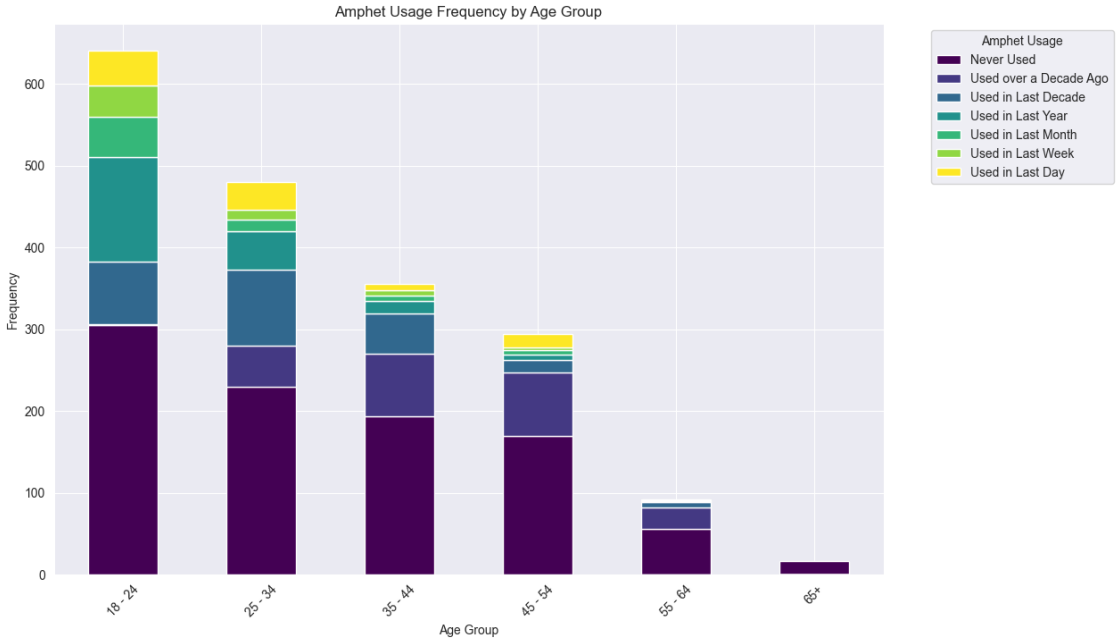
[66]:
```
# Define all names of demographics columns
demographics_columns = ['age_values', 'gender_values', 'education_values',␣
 ↪'country_values', 'ethnicity_values']
# Define all names of personality traits columns
personality_traits_scores_columns = [
    'nscore_values', 'escore_values', 'oscore_values', 'ascore_values',
    'cscore_values', 'impulsive_values', 'sensation_values'
]
# Filter out all names of drug columns
drug_columns = [col for col in df.columns if
                'values' in col and col not in demographics_columns and col not␣
 ↪in personality_traits_scores_columns]
# Sort age groups based on the age_mapping order
age_order = [age_mapping[key] for key in sorted(age_mapping)]
```

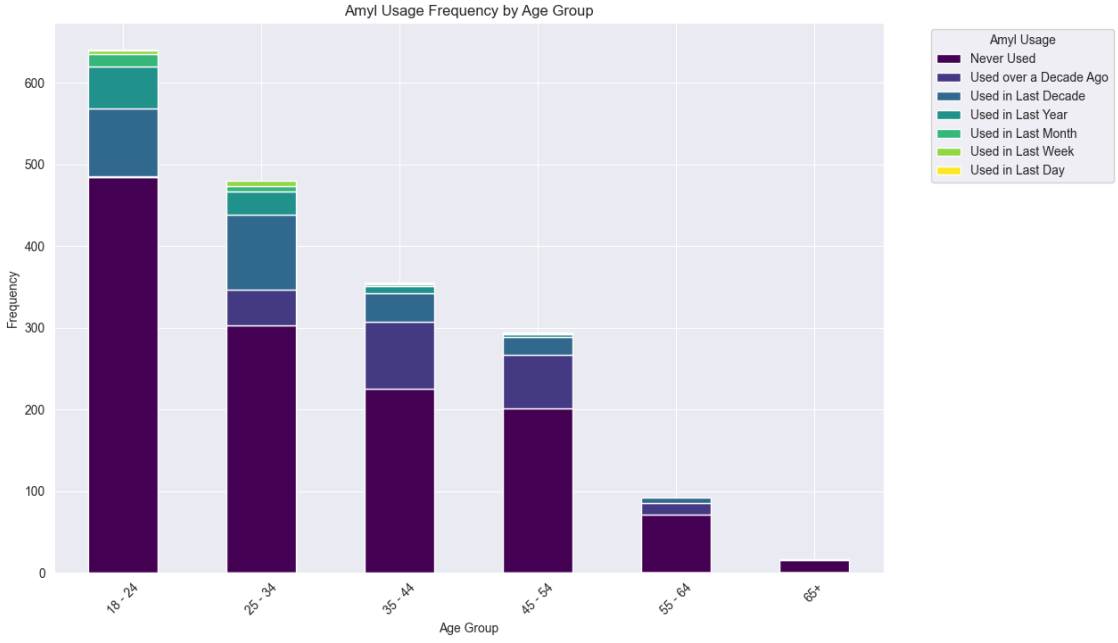
[67]:
```
# Mapping drug usage categories to numerical values
usage_mapping = {
    "Never Used": 0, "Used over a Decade Ago": 1, "Used in Last Decade": 2,
    "Used in Last Year": 3, "Used in Last Month": 4, "Used in Last Week": 5,
    "Used in Last Day": 6
}
```

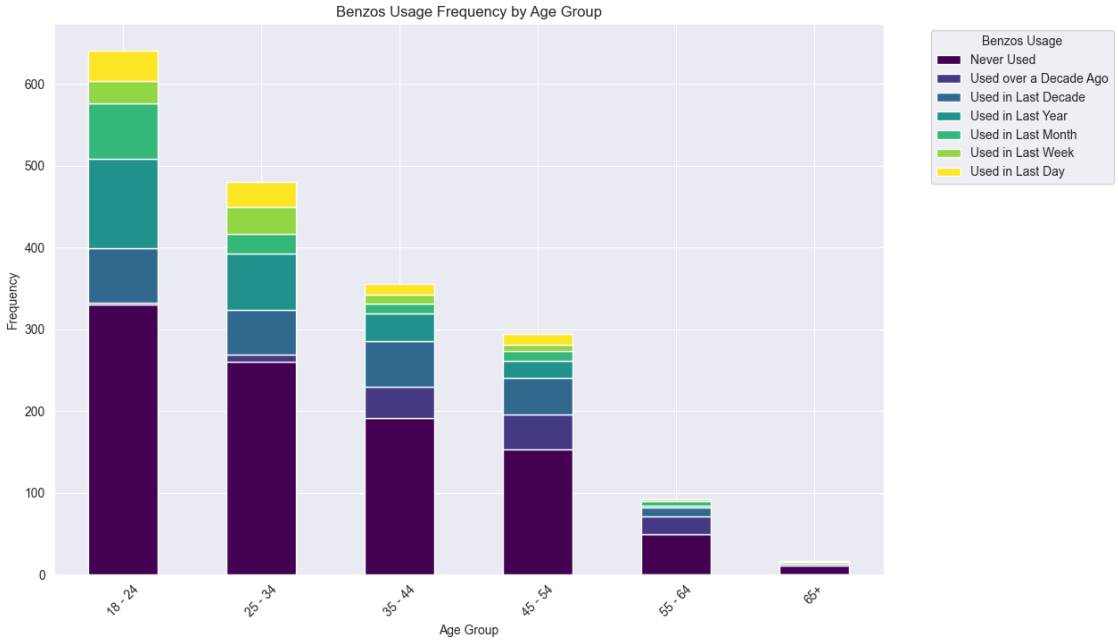
[68]:
```
# Question 1: Distribution of Drug Usage Frequency by Age Group


# Bar Chart per Drug Usage by Age Group
for drug in drug_columns:
    # Group by 'age_values' and drug category, then count occurrences
    drug_usage_age_group = df.groupby(['age_values', drug], observed=False).
 ↪size().unstack(fill_value=0)

    # Reorder DataFrame columns according to the custom category order in␣
 ↪CATEGORIES
    ordered_columns = [CATEGORIES[key] for key in
                       sorted(CATEGORIES, key=lambda x: int(x[2:]))]  # Sort␣
 ↪keys based on numerical part
    drug_usage_age_group = drug_usage_age_group.reindex(columns=ordered_columns)
```

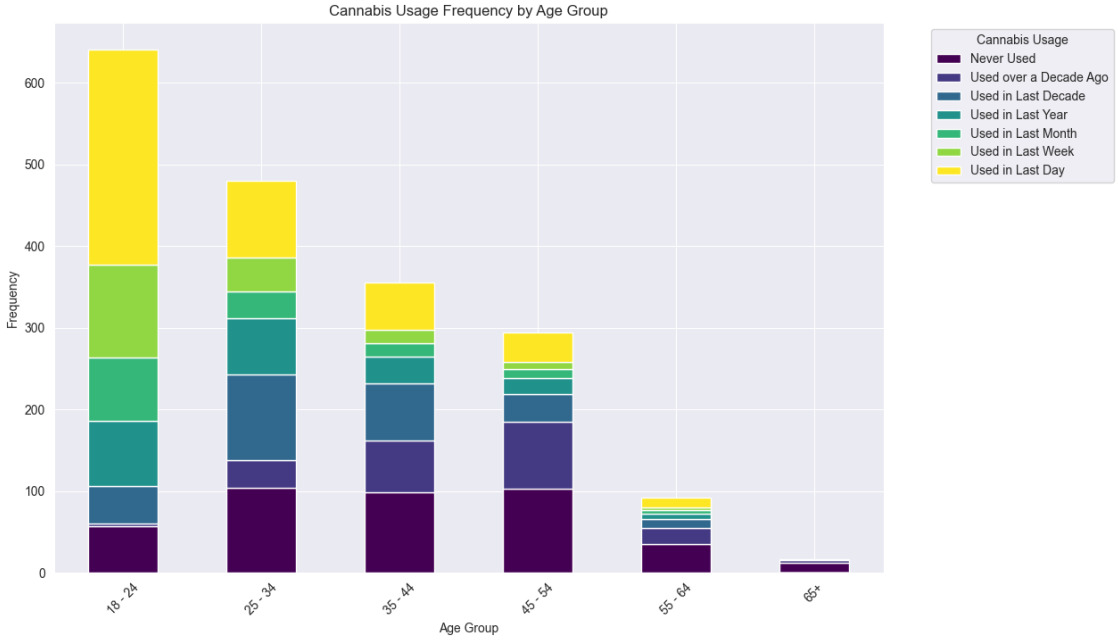
```
    # Convert age_values to more readable age group labels if necessary
    drug_usage_age_group.index = drug_usage_age_group.index.map(lambda x:␣
↪age_mapping.get(x, x))

    # Plot the data with the custom order
    drug_usage_age_group.plot(kind='bar', stacked=True, figsize=(12, 8),␣
↪cmap='viridis')
    plt.title(f'{drug.replace("_values", "")} Usage Frequency by Age Group')
    plt.xlabel('Age Group')
    plt.ylabel('Frequency')
    plt.legend(title=f'{drug.replace("_values", "")} Usage', bbox_to_anchor=(1.
↪05, 1), loc='upper left')
    plt.xticks(rotation=45)
    plt.show()
```

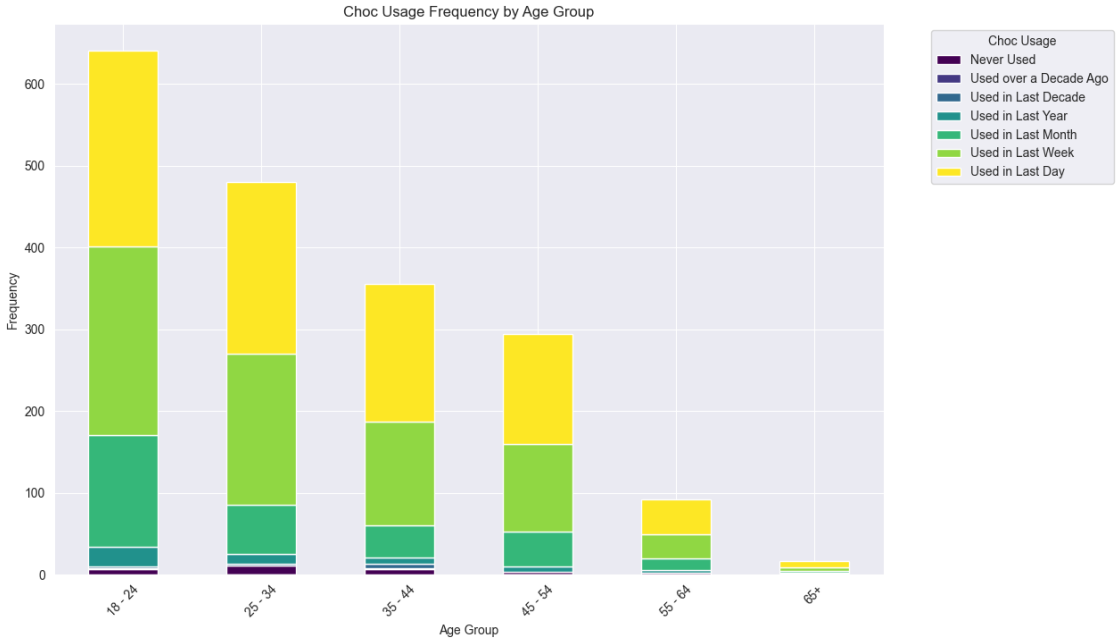


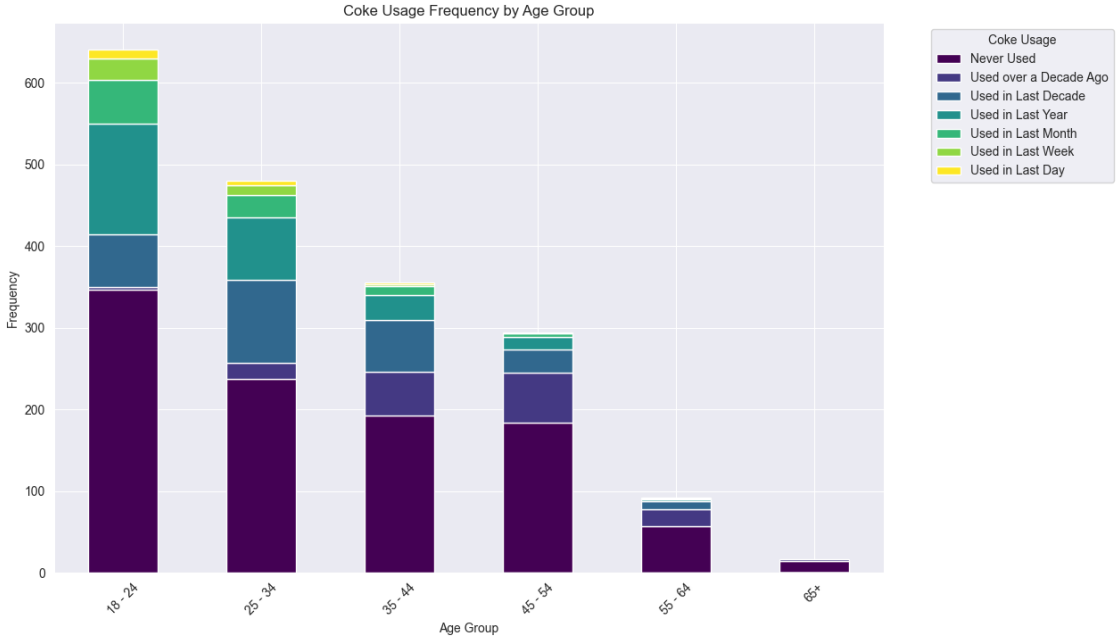

Amphet Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Amphet Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

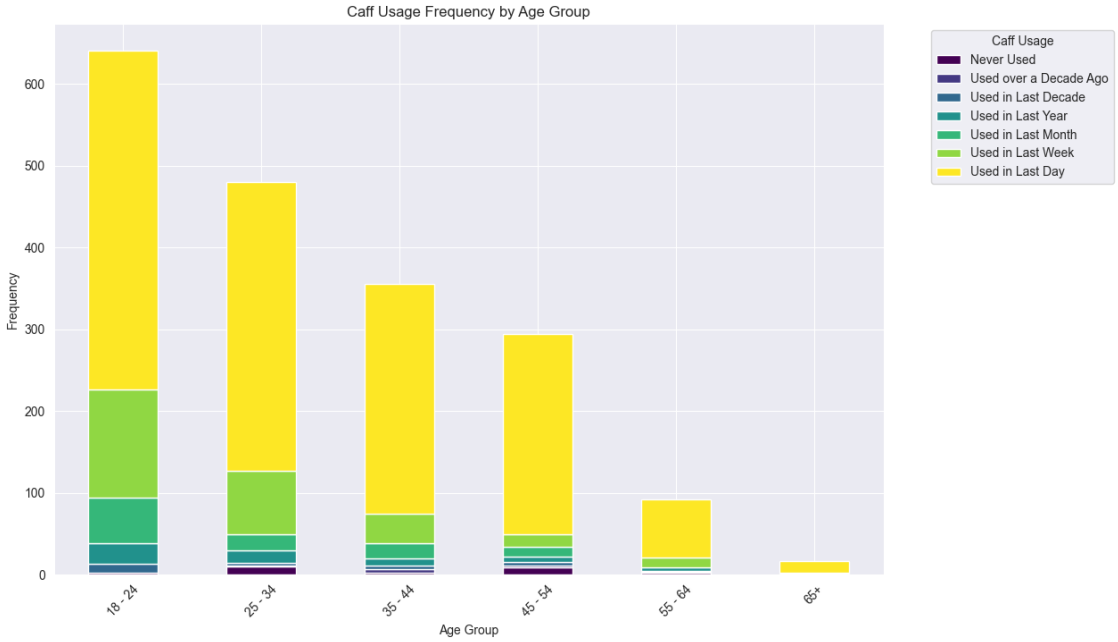

Amyl Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Amyl Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

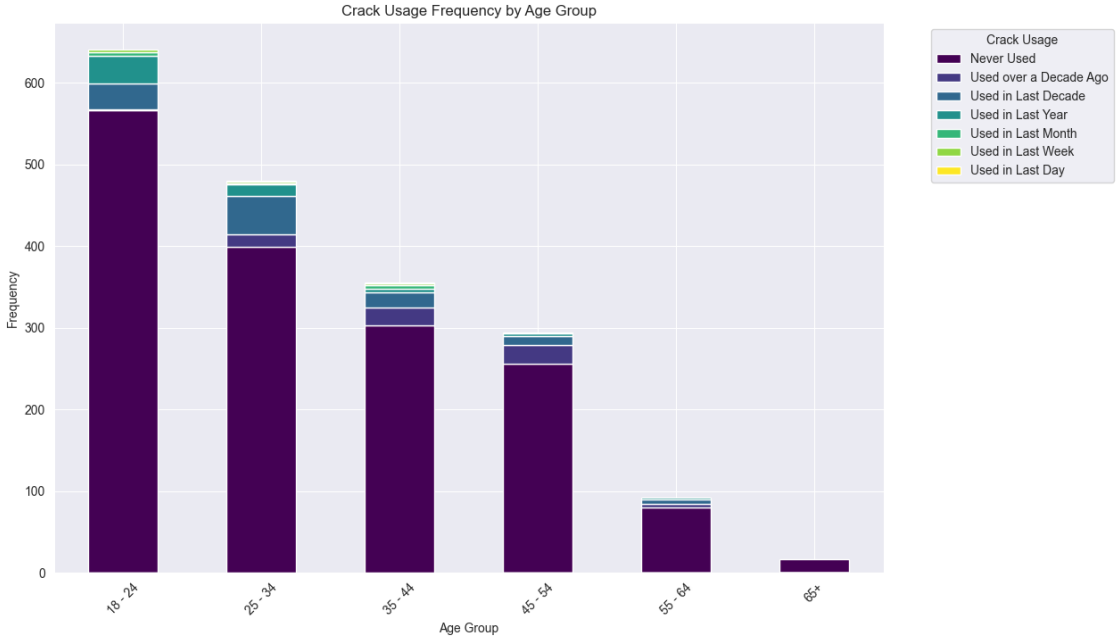

Benzos Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Benzos Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

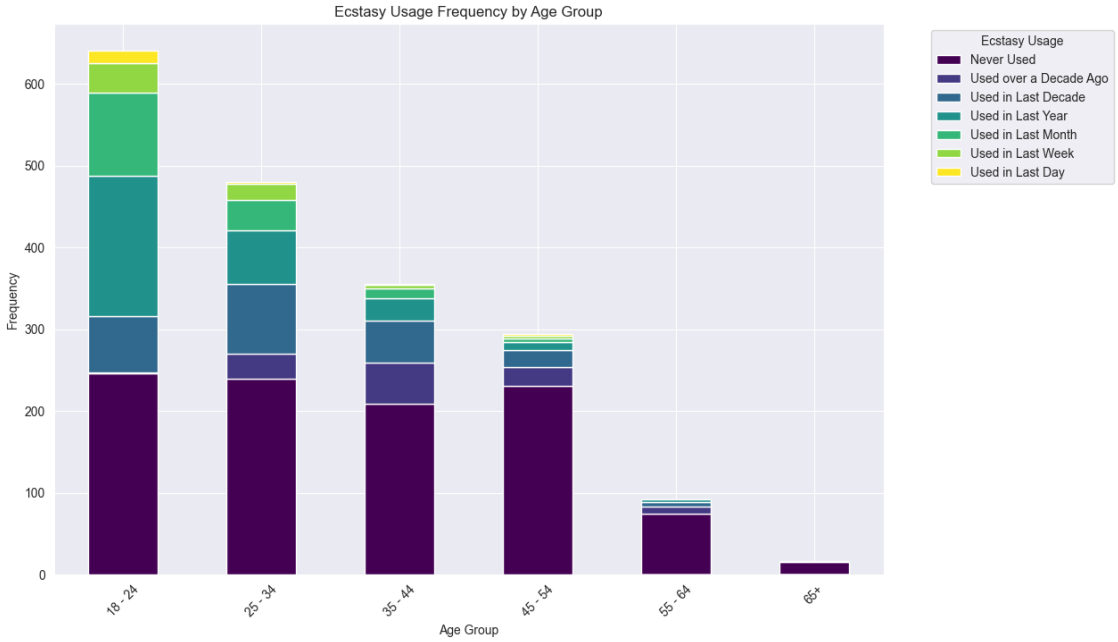

Cannabis Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Cannabis Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

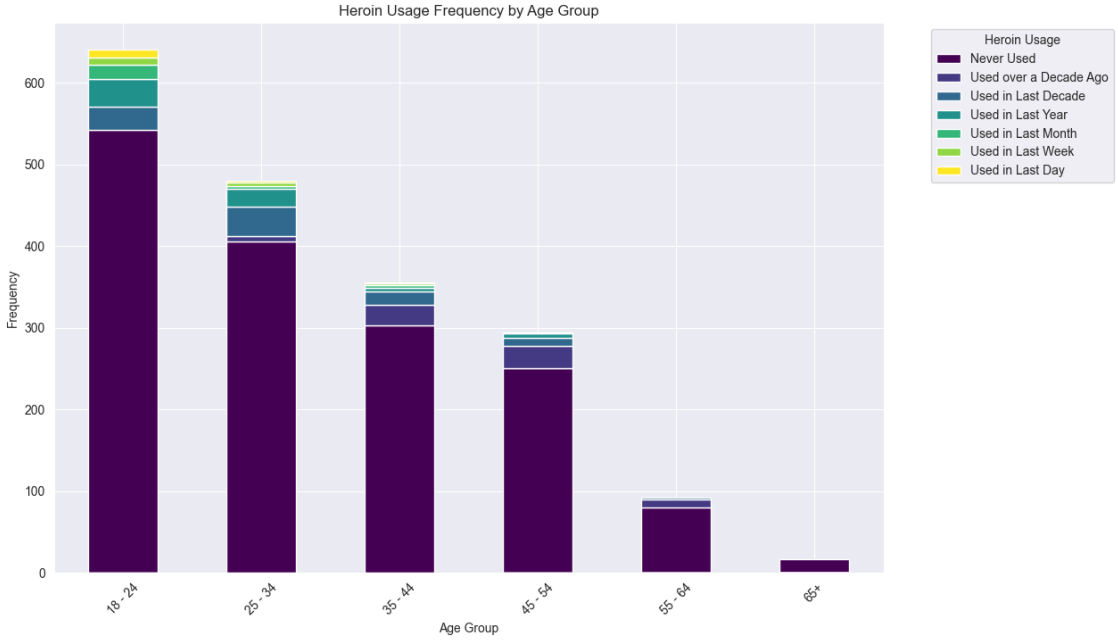

Choc Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Choc Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

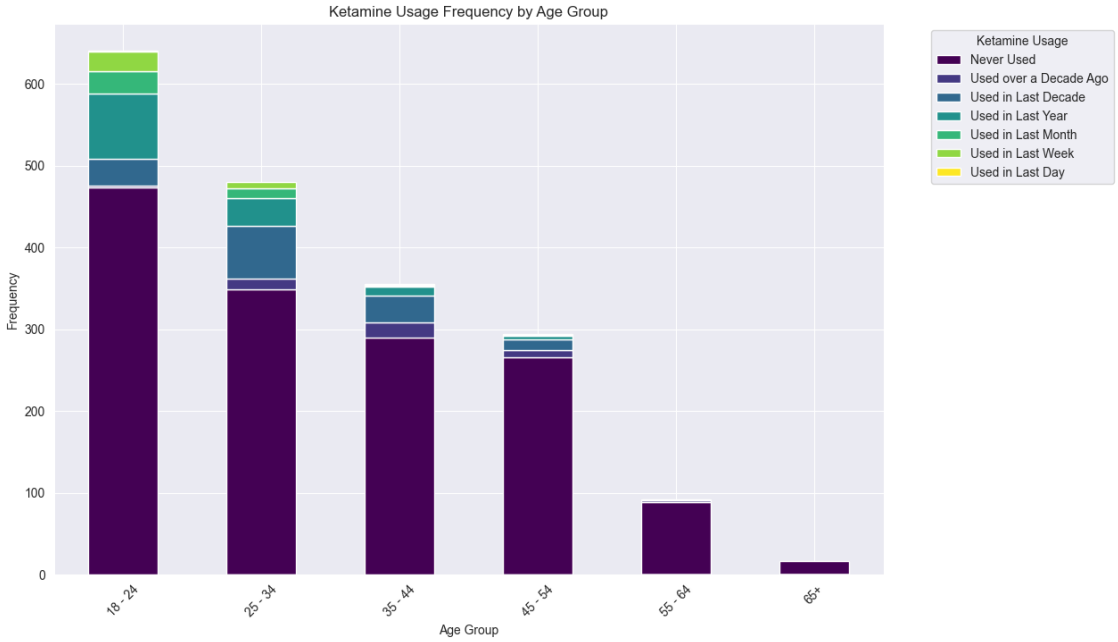

Coke Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Coke Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

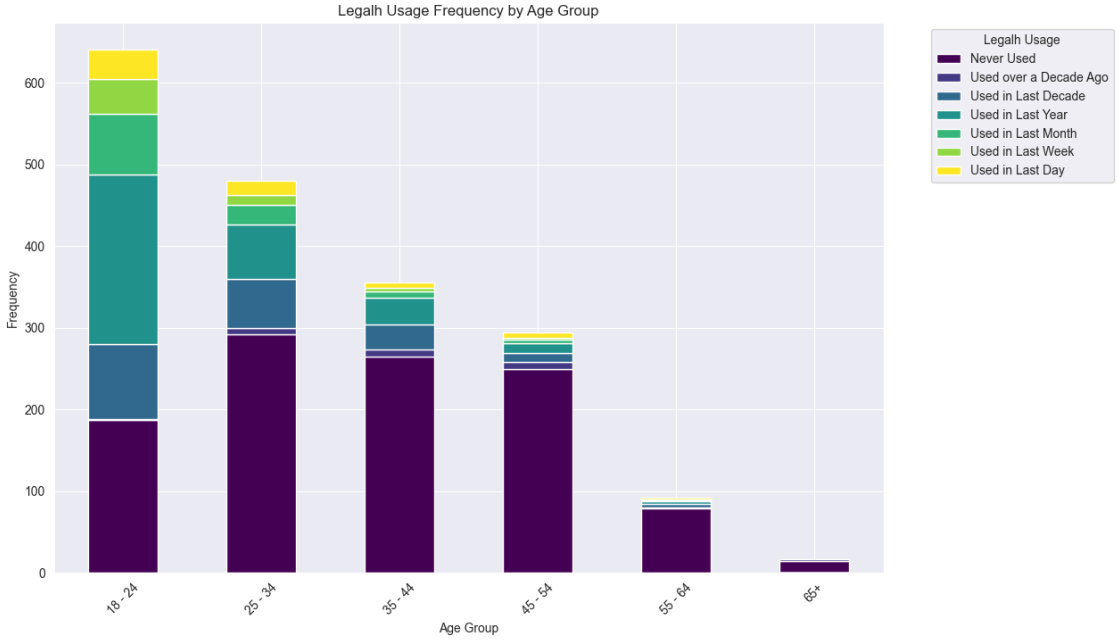

Caff Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Caff Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

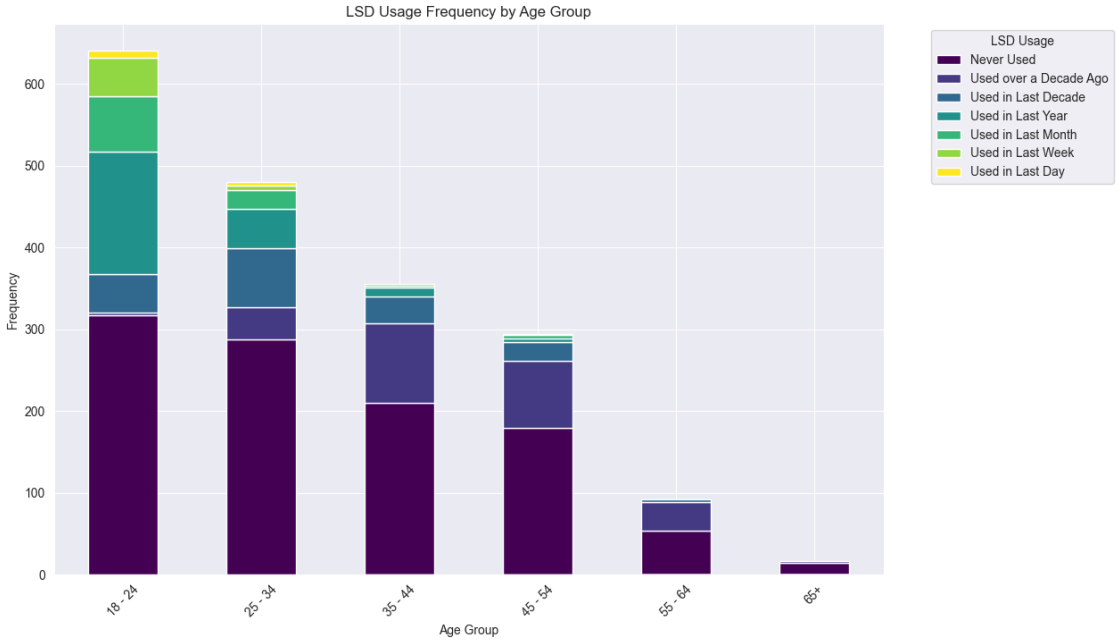

Crack Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Crack Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

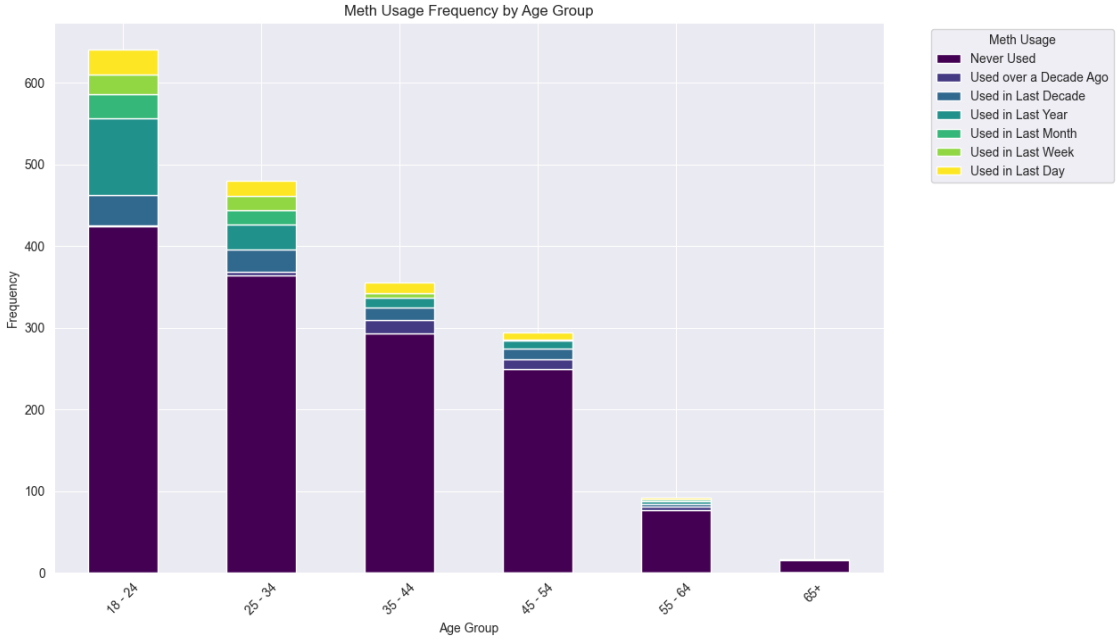

Ecstasy Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Ecstasy Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

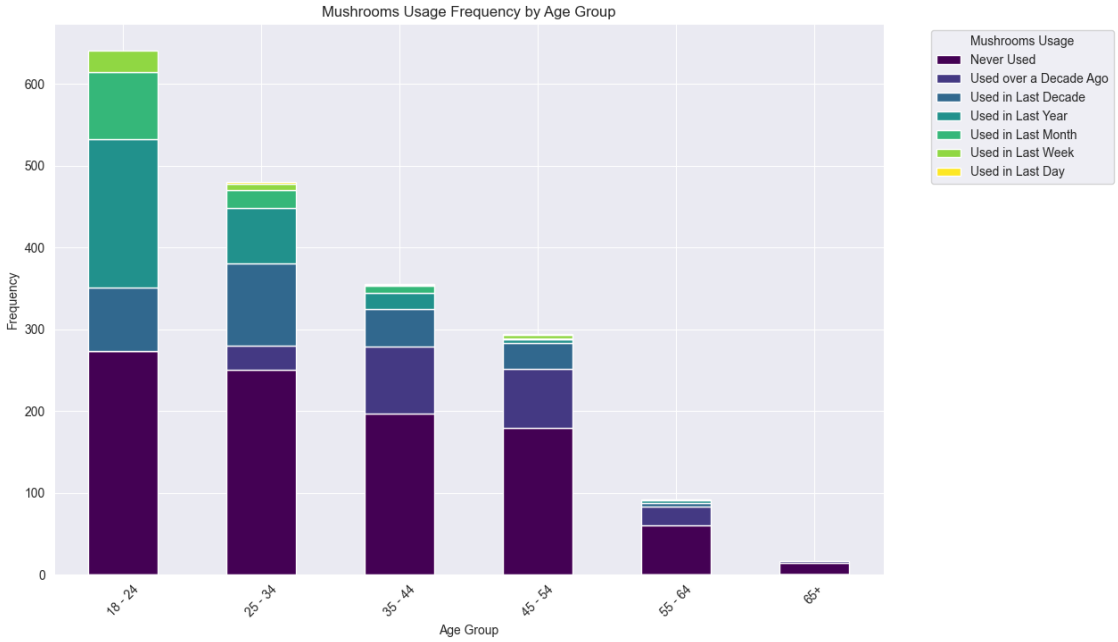

Heroin Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Heroin Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

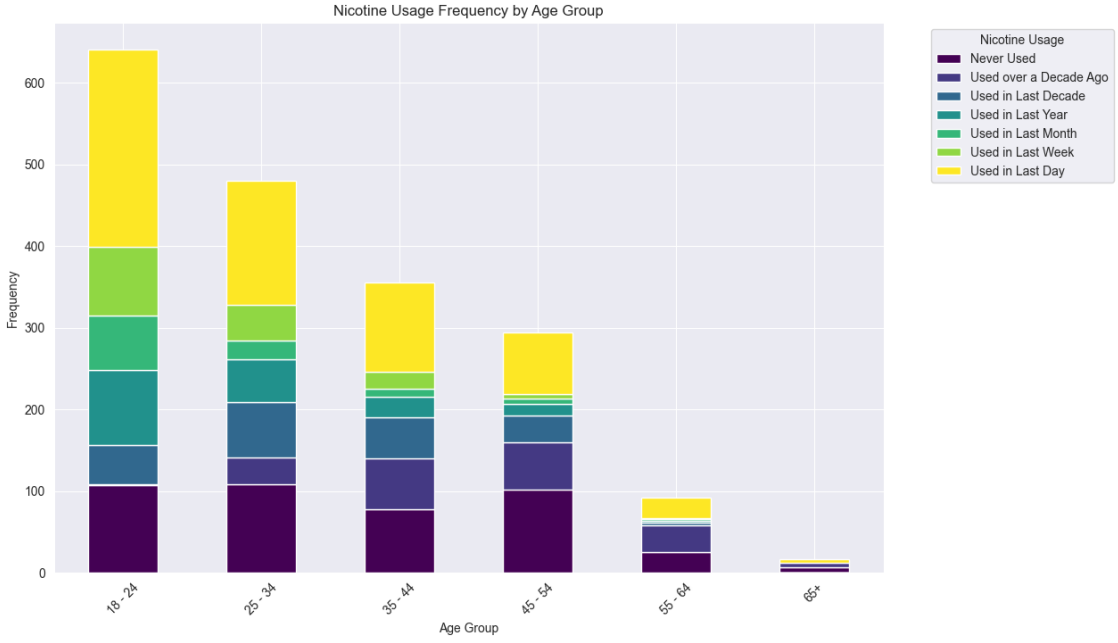

Ketamine Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Ketamine Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

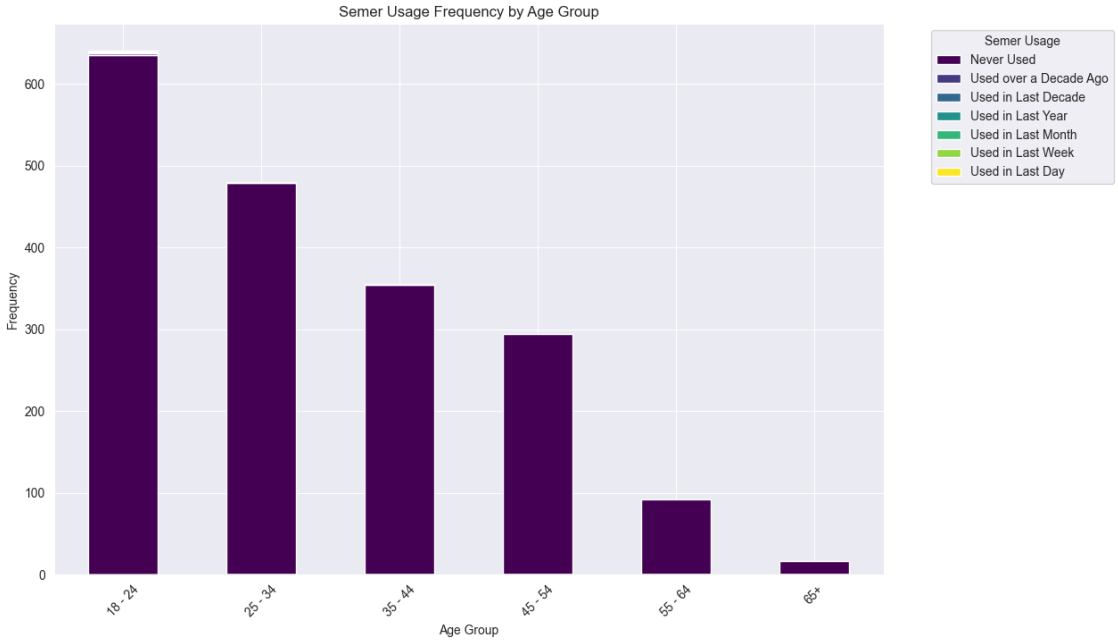

Legalh Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Legalh Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

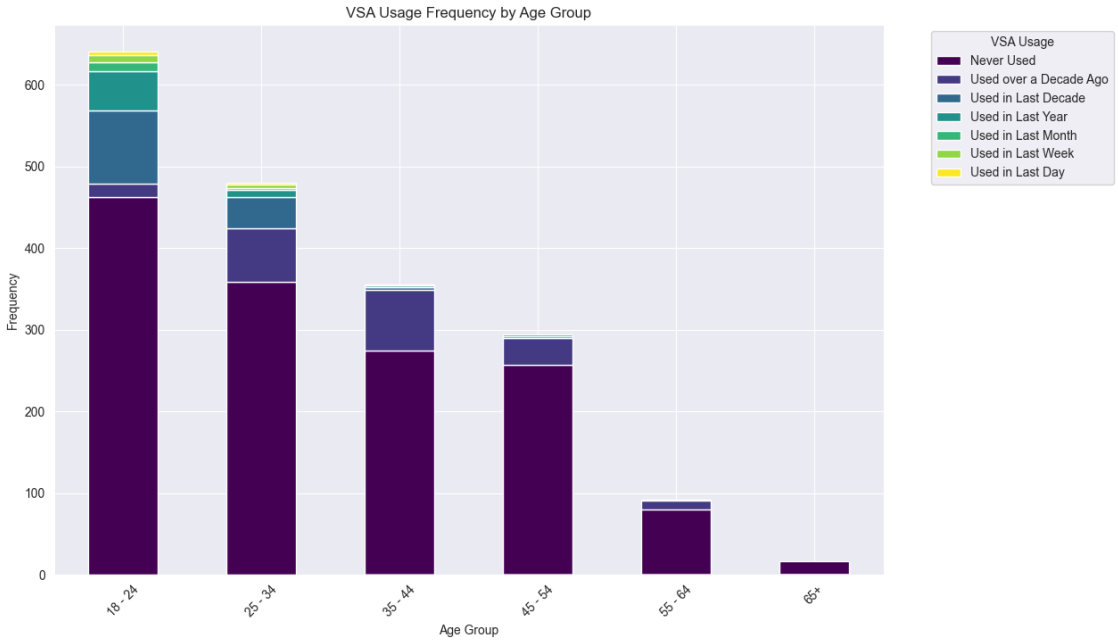

LSD Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
LSD Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day


Meth Usage Frequency by Age Group
Frequency
0
100
200
300
400
500
600
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Age Group
Meth Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day


Mushrooms Usage Frequency by Age Group
Frequency
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Mushrooms Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

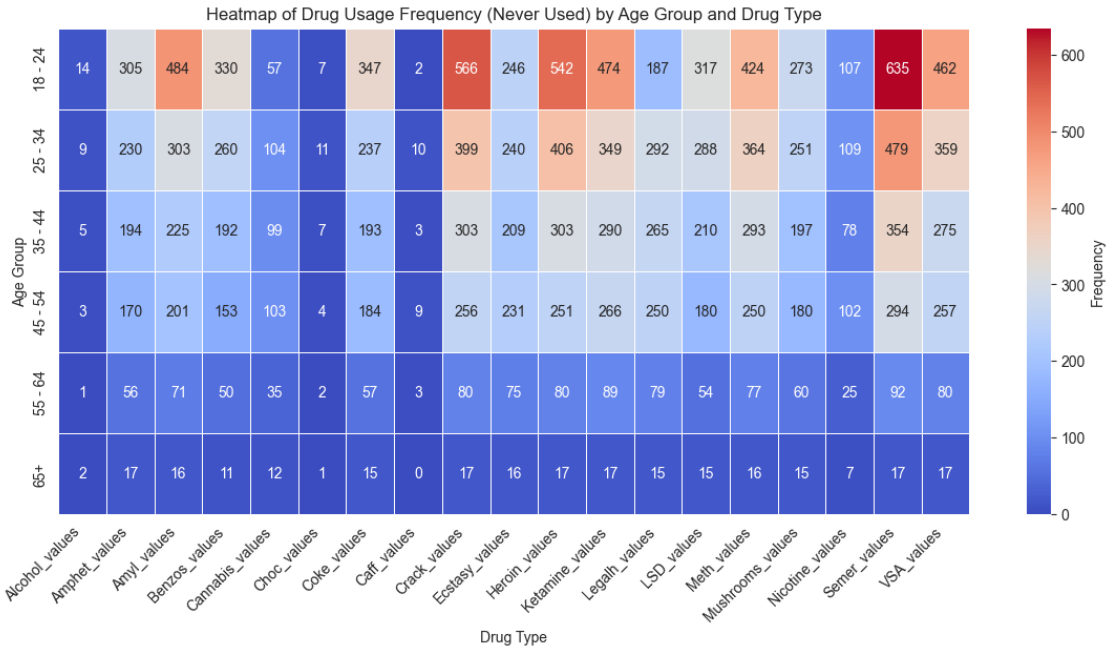

Nicotine Usage Frequency by Age Group
Frequency
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Nicotine Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

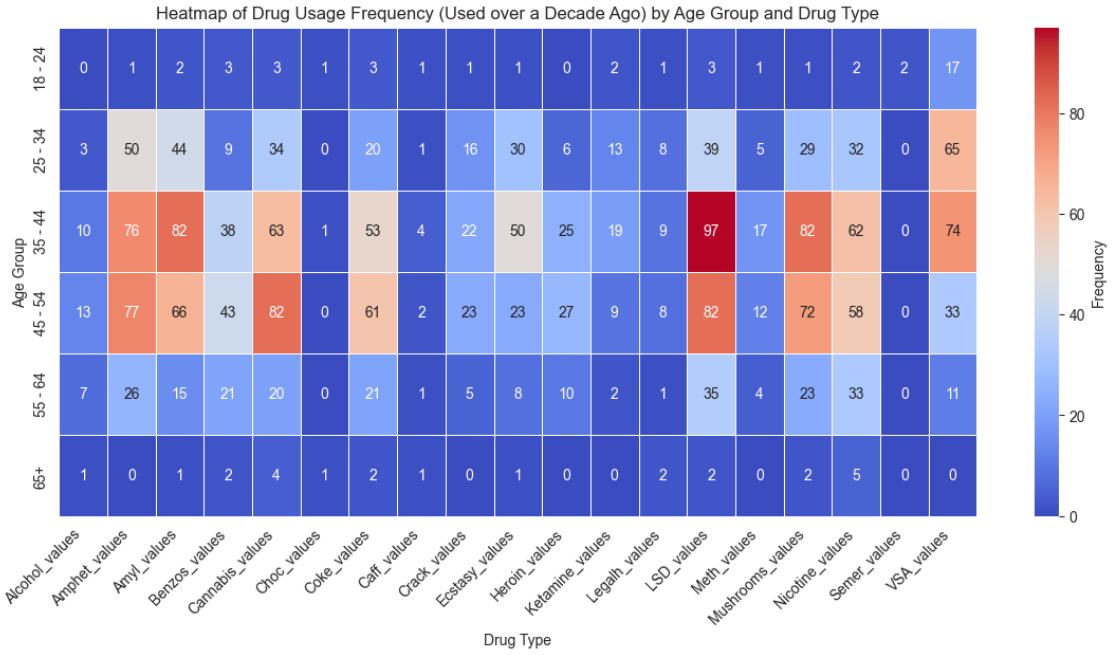


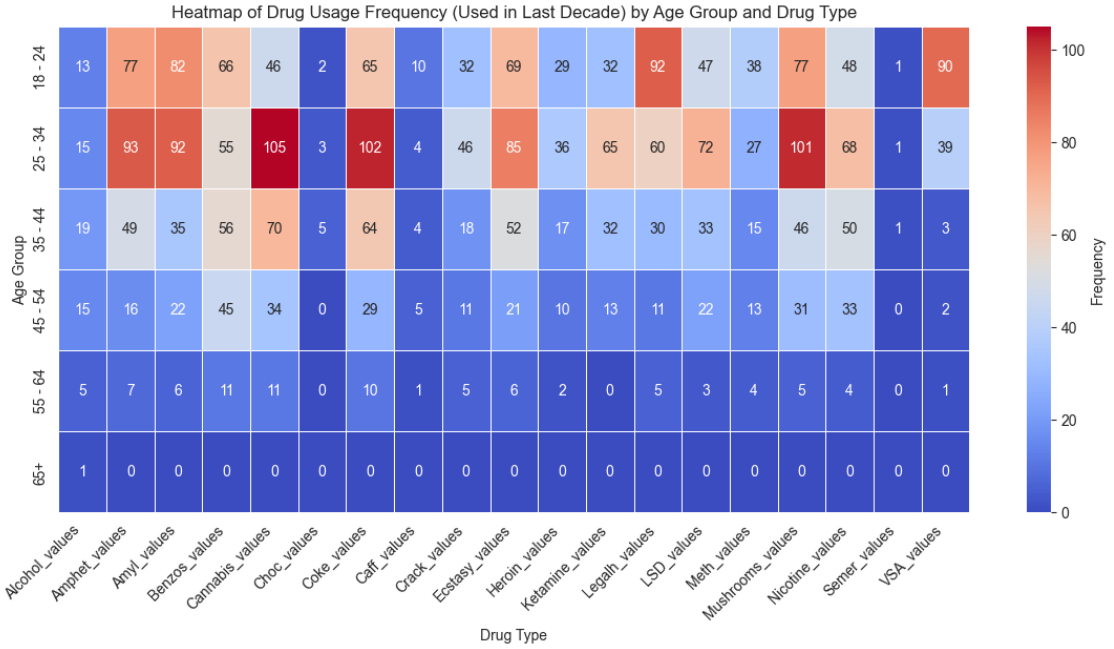


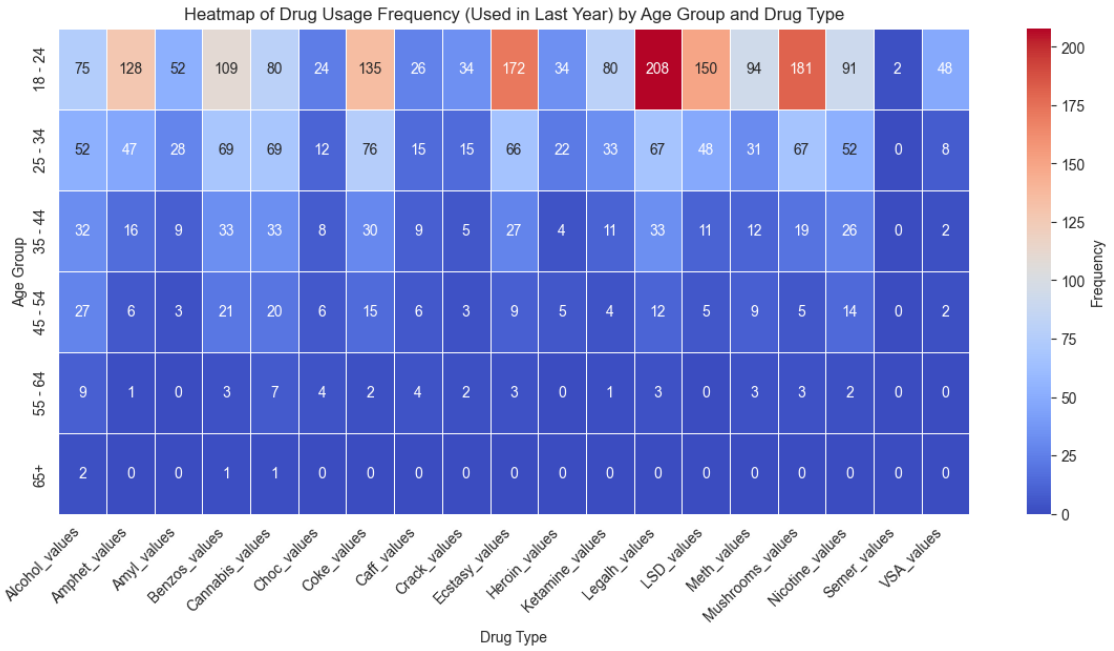
[69]:
```
# Initialize the usage matrix for all age groups
usage_matrix = pd.DataFrame(0, index=age_order, columns=drug_columns)

for category in CATEGORIES.values():
    # Populate the usage matrix
```

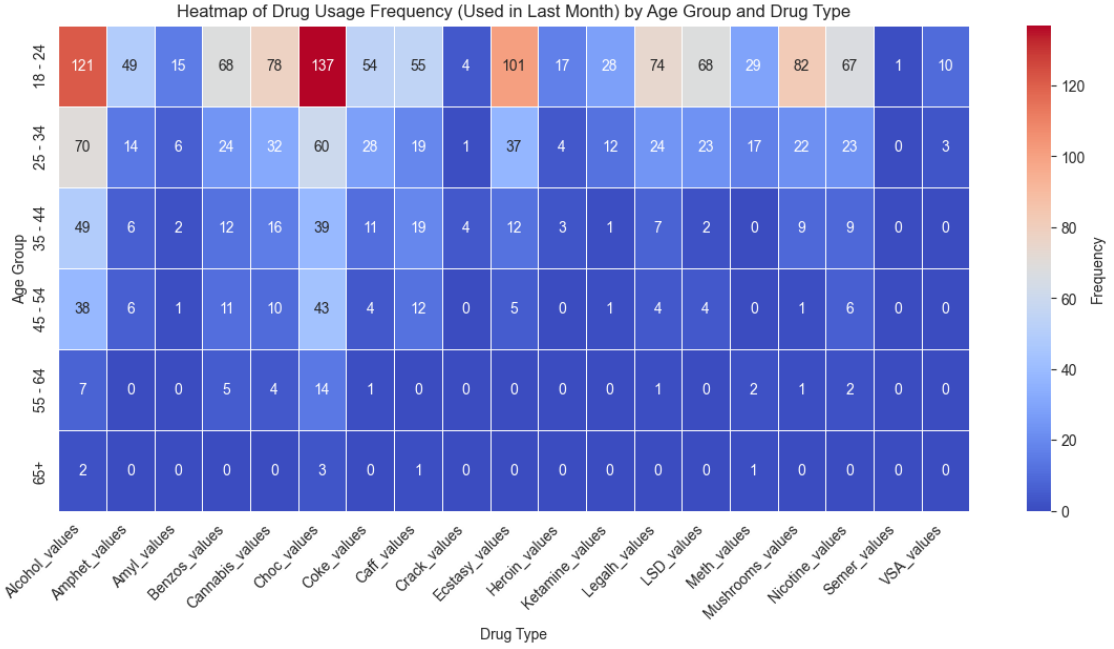
```
    for age_key, age_group in age_mapping.items():
        age_group_data = df[df['age_values'] == age_group]
        for drug in drug_columns:
            usage_counts = age_group_data[drug].value_counts()
            # Aggregate the counts for the category "Used in Last Day"
            usage_matrix.loc[age_group, drug] = usage_counts.get(category, 0)

    # Plotting the combined heatmap for all age groups
    plt.figure(figsize=(14, 6))  # Adjust size as necessary
    # Format: https://stackoverflow.com/a/65020192
    sns.heatmap(usage_matrix, annot=True, fmt="g", cmap="coolwarm",␣
↪cbar_kws={'label': 'Frequency'}, linewidths=0.5)
    plt.title(f'Heatmap of Drug Usage Frequency ({category}) by Age Group and␣
↪Drug Type')
    plt.xlabel('Drug Type')
    plt.ylabel('Age Group')
    plt.xticks(rotation=45, ha='right')
    plt.show()
```

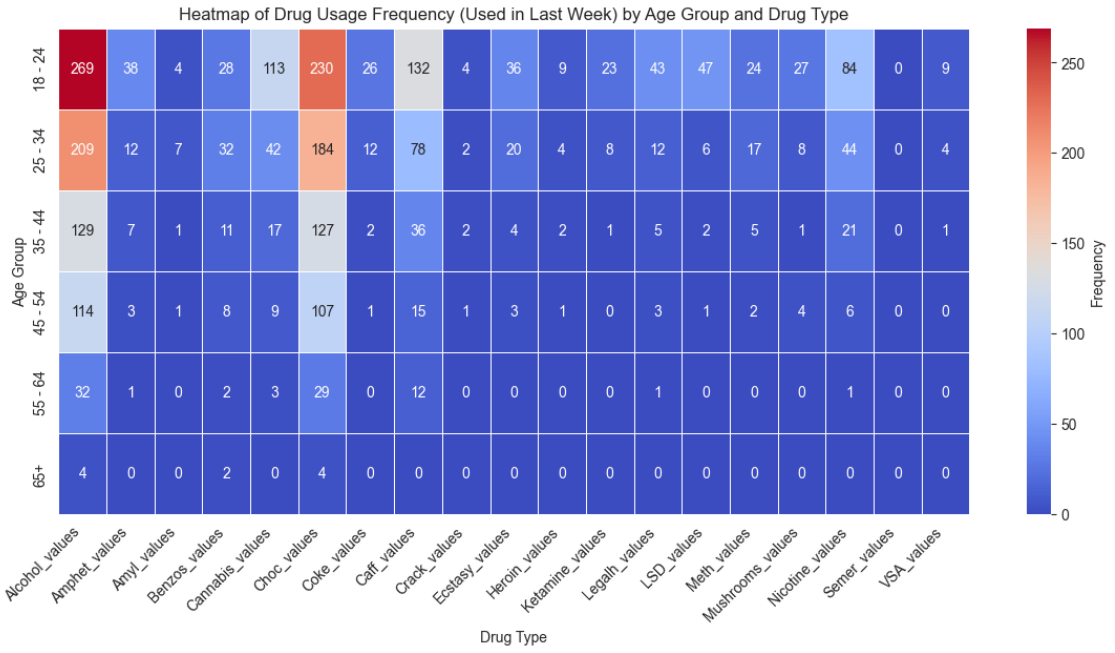


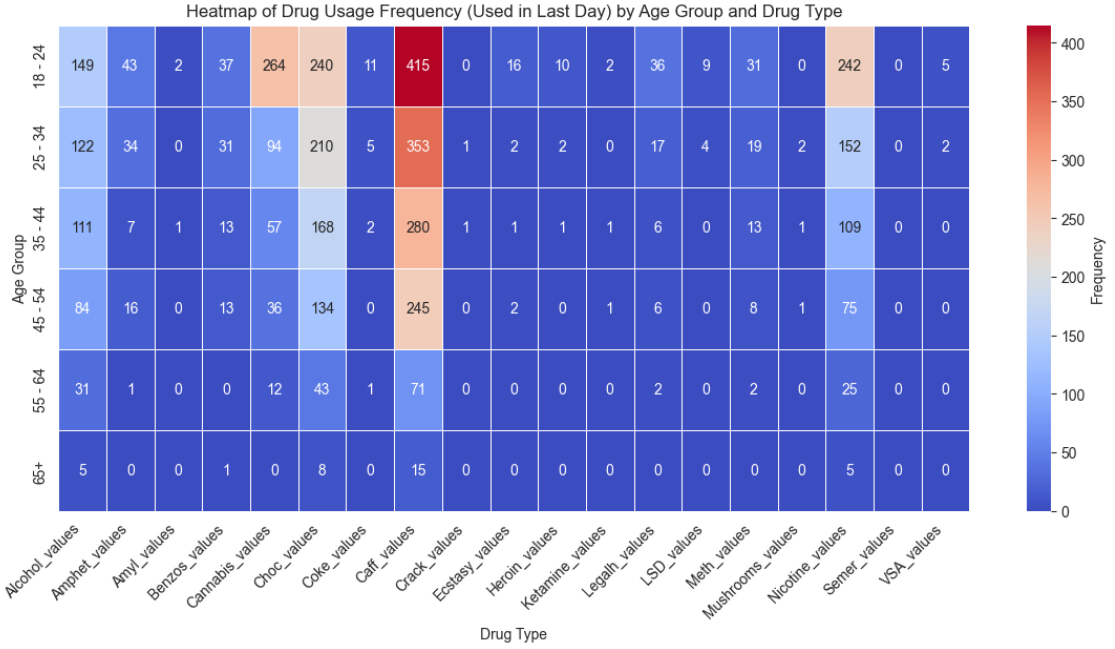

Heatmap of Drug Usage Frequency (Used over a Decade Ago) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Frequency
0
20
40
60
80


Heatmap of Drug Usage Frequency (Used in Last Decade) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Frequency
0
20
40
60
80
100


Heatmap of Drug Usage Frequency (Used in Last Year) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
75 128 52 109 80 24 135 26 34 172 34 80 208 150 94 181 91 2 48
52 47 28 69 69 12 76 15 15 66 22 33 67 48 31 67 52 0 8
32 16 9 33 33 8 30 9 5 27 4 11 33 11 12 19 26 0 2
27 6 3 21 20 6 15 6 3 9 5 4 12 5 9 5 14 0 2
9 1 0 3 7 4 2 4 2 3 0 1 3 0 3 3 2 0 0
2 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Frequency
200
175
150
125
100
75
50
25
0


Heatmap of Drug Usage Frequency (Used in Last Month) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
121 49 15 68 78 137 54 55 4 101 17 28 74 68 29 82 67 1 10
70 14 6 24 32 60 28 19 1 37 4 12 24 23 17 22 23 0 3
49 6 2 12 16 39 11 19 4 12 3 1 7 2 0 9 9 0 0
38 6 1 11 10 43 4 12 0 5 0 1 4 4 0 1 6 0 0
7 0 0 5 4 14 1 0 0 0 0 0 1 0 2 1 2 0 0
2 0 0 0 0 3 0 1 0 0 0 0 0 0 1 0 0 0 0
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Frequency
120
100
80
60
40
20
0





[70]:
```
# Iterate over each category in the dictionary CATEGORIES, whose values may␣
 ↪represent specific groups of interest
for category in CATEGORIES.values():
    # Populate the usage matrix with percentages
```

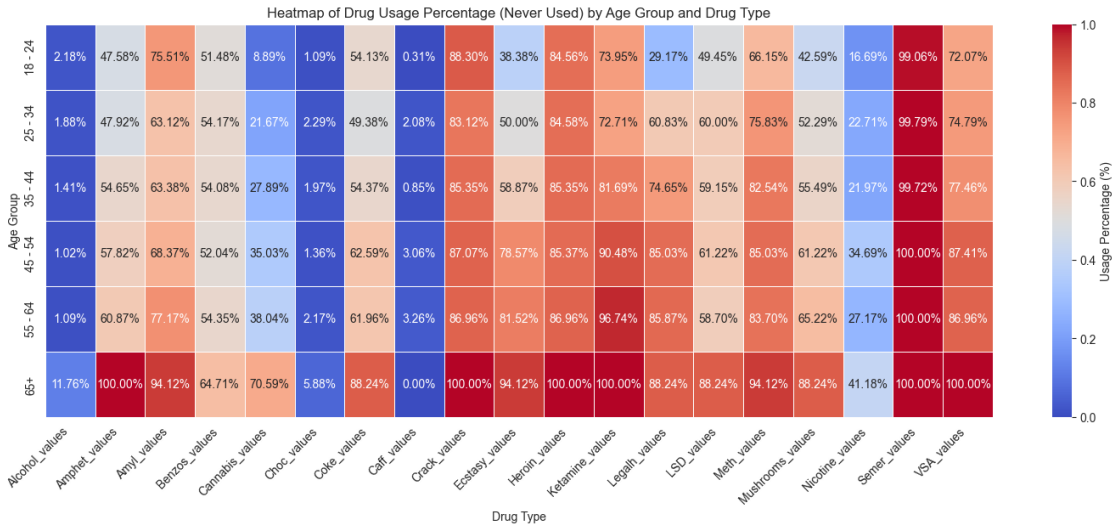
```
    for age_key, age_group in age_mapping.items():
        # Filter the dataframe to include only data for the current age group
        age_group_data = df[df['age_values'] == age_group]
        # Total number of respondents in this age group
        total_respondents = age_group_data.shape[0]

        # Iterate over the columns in the dataframe that correspond to␣
↪different drugs
        for drug in drug_columns:
            # Get the count of each category's usage within the specified drug␣
↪column
            usage_counts = age_group_data[drug].value_counts()
            if total_respondents > 0:
                # Calculate the percentage of respondents using the drug␣
↪categorized under 'category'
                usage_percentage = (usage_counts.get(category, 0) /␣
↪total_respondents)
                # Update the corresponding cell in the usage matrix with the␣
↪calculated percentage
                usage_matrix.loc[age_group, drug] = usage_percentage
            else:
                # If there are no respondents in this age group, set usage␣
↪percentage to 0 to avoid division by zero
                usage_matrix.loc[age_group, drug] = 0

    # Plotting the combined heatmap for each drug usage category
    plt.figure(figsize=(18, 6))  # Set the size of the figure
    sns.heatmap(usage_matrix, annot=True, fmt=".2%", cmap="coolwarm",␣
↪cbar_kws={'label': 'Usage Percentage (%)'},
                linewidths=0.5)

    plt.title(f'Heatmap of Drug Usage Percentage ({category}) by Age Group and␣
↪Drug Type')
    plt.xlabel('Drug Type')
    plt.ylabel('Age Group')
    # Rotate x-axis labels for better readability
    plt.xticks(rotation=45, ha='right')
    plt.show()
```

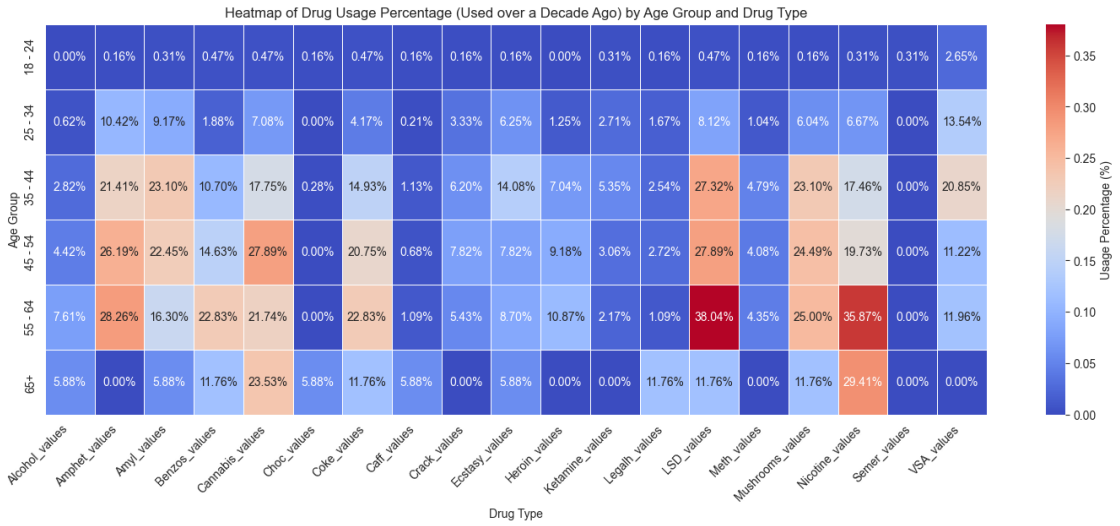
```
/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.0218408736349454' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
```

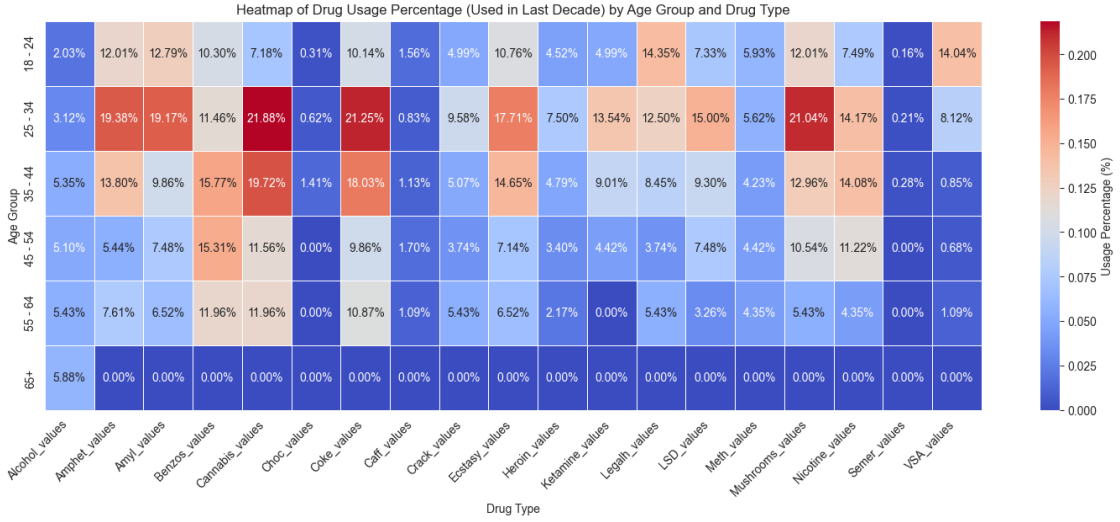
```
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.47581903276131043' has dtype incompatible
with int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.7550702028081123' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.514820592823713' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.08892355694227769' has dtype incompatible
with int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.0109204368174727' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.5413416536661466' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.0031201248049922' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.
```

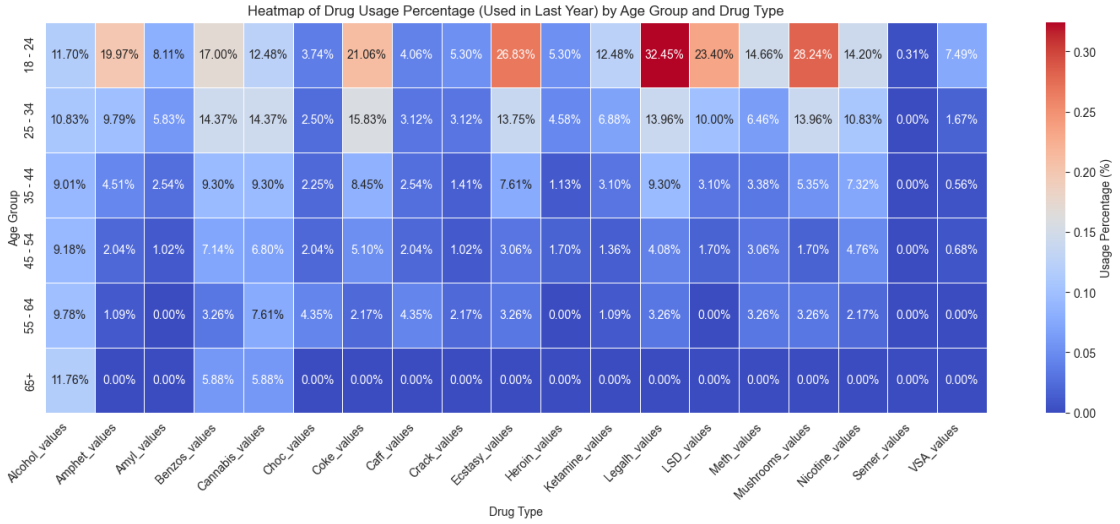
```
/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.8829953198127926' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.3837753510140406' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.8455538221528861' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.7394695787831513' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.2917316692667707' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.49453978159126366' has dtype incompatible
with int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.6614664586583463' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.
```

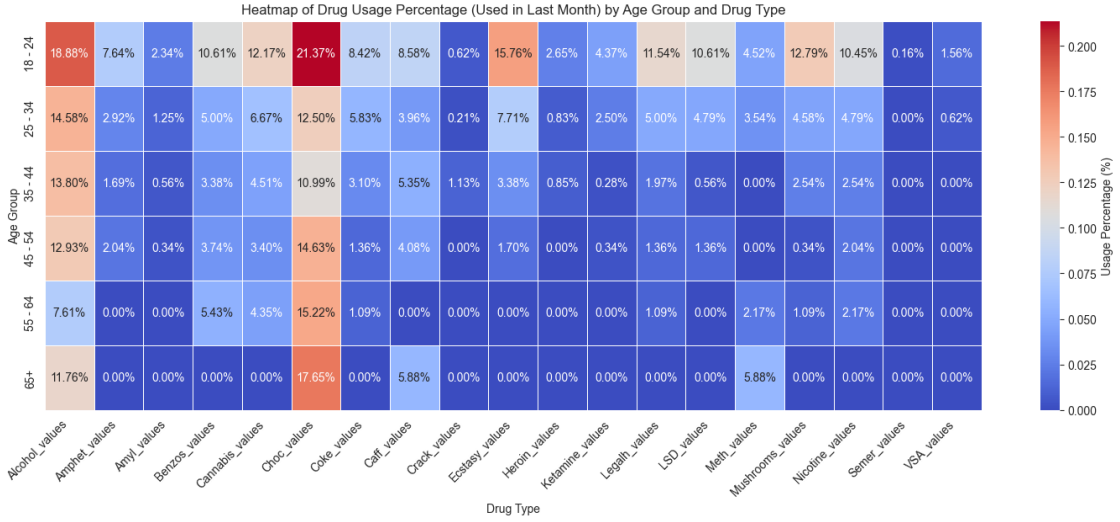
```
/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.42589703588143524' has dtype incompatible
with int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.1669266770670827' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.9906396255850234' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.

/var/folders/ck/vgxn_9f907d122j56f3203gm0000gn/T/ipykernel_35710/922468807.py:18
: FutureWarning:

Setting an item of incompatible dtype is deprecated and will raise an error in a
future version of pandas. Value '0.7207488299531981' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.
```

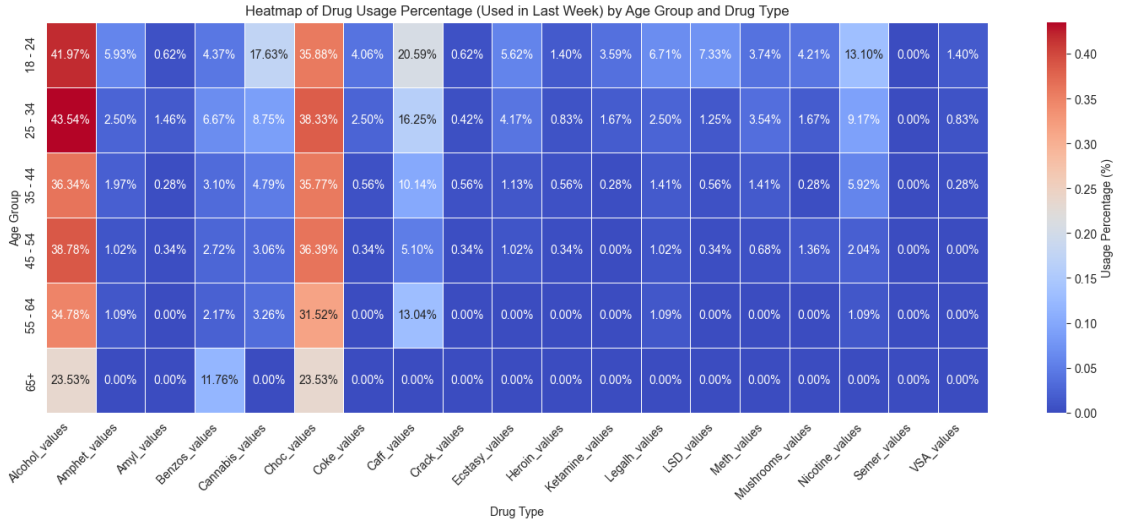


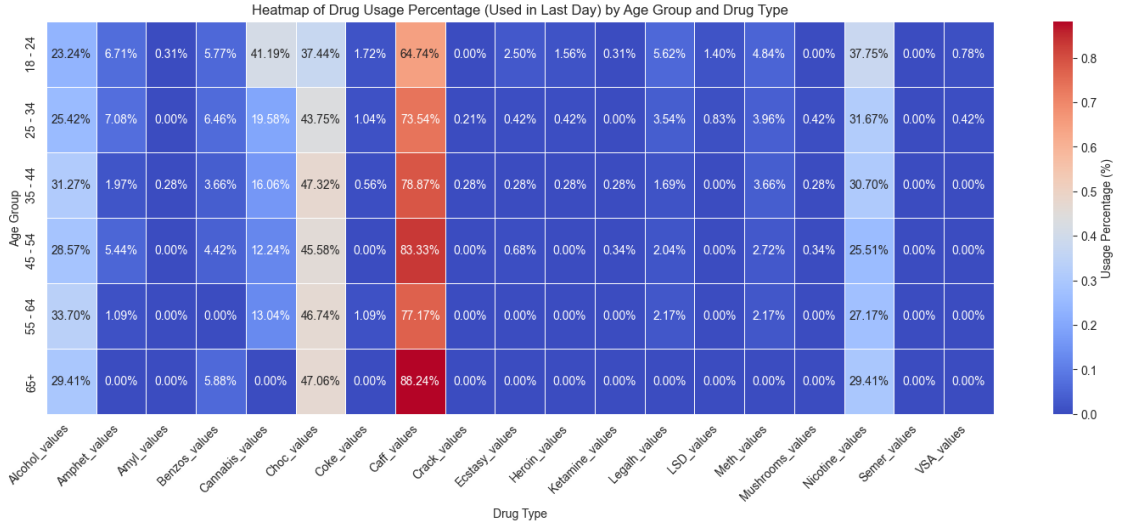

Heatmap of Drug Usage Percentage (Used over a Decade Ago) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
0.00% 0.16% 0.31% 0.47% 0.47% 0.16% 0.47% 0.16% 0.16% 0.16% 0.00% 0.31% 0.16% 0.47% 0.16% 0.16% 0.31% 0.31% 2.65%
0.62% 10.42% 9.17% 1.88% 7.08% 0.00% 4.17% 0.21% 3.33% 6.25% 1.25% 2.71% 1.67% 8.12% 1.04% 6.04% 6.67% 0.00% 13.54%
2.82% 21.41% 23.10% 10.70% 17.75% 0.28% 14.93% 1.13% 6.20% 14.08% 7.04% 5.35% 2.54% 27.32% 4.79% 23.10% 17.46% 0.00% 20.85%
4.42% 26.19% 22.45% 14.63% 27.89% 0.00% 20.75% 0.68% 7.82% 7.82% 9.18% 3.06% 2.72% 27.89% 4.08% 24.49% 19.73% 0.00% 11.22%
7.61% 28.26% 16.30% 22.83% 21.74% 0.00% 22.83% 1.09% 5.43% 8.70% 10.87% 2.17% 1.09% 38.04% 4.35% 25.00% 35.87% 0.00% 11.96%
5.88% 0.00% 5.88% 11.76% 23.53% 5.88% 11.76% 5.88% 0.00% 5.88% 0.00% 0.00% 11.76% 11.76% 0.00% 11.76% 29.41% 0.00% 0.00%
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Usage Percentage (%)
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00


Heatmap of Drug Usage Percentage (Used in Last Decade) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
2.03% 12.01% 12.79% 10.30% 7.18% 0.31% 10.14% 1.56% 4.99% 10.76% 4.52% 4.99% 14.35% 7.33% 5.93% 12.01% 7.49% 0.16% 14.04%
3.12% 19.38% 19.17% 11.46% 21.88% 0.62% 21.25% 0.83% 9.58% 17.71% 7.50% 13.54% 12.50% 15.00% 5.62% 21.04% 14.17% 0.21% 8.12%
5.35% 13.80% 9.86% 15.77% 19.72% 1.41% 18.03% 1.13% 5.07% 14.65% 4.79% 9.01% 8.45% 9.30% 4.23% 12.96% 14.08% 0.28% 0.85%
5.10% 5.44% 7.48% 15.31% 11.56% 0.00% 9.86% 1.70% 3.74% 7.14% 3.40% 4.42% 3.74% 7.48% 4.42% 10.54% 11.22% 0.00% 0.68%
5.43% 7.61% 6.52% 11.96% 11.96% 0.00% 10.87% 1.09% 5.43% 6.52% 2.17% 0.00% 5.43% 3.26% 4.35% 5.43% 4.35% 0.00% 1.09%
5.88% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00%
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Usage Percentage (%)
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000


Heatmap of Drug Usage Percentage (Used in Last Year) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
11.70% 19.97% 8.11% 17.00% 12.48% 3.74% 21.06% 4.06% 5.30% 26.83% 5.30% 12.48% 32.45% 23.40% 14.66% 28.24% 14.20% 0.31% 7.49%
10.83% 9.79% 5.83% 14.37% 14.37% 2.50% 15.83% 3.12% 3.12% 13.75% 4.58% 6.88% 13.96% 10.00% 6.46% 13.96% 10.83% 0.00% 1.67%
9.01% 4.51% 2.54% 9.30% 9.30% 2.25% 8.45% 2.54% 1.41% 7.61% 1.13% 3.10% 9.30% 3.10% 3.38% 5.35% 7.32% 0.00% 0.56%
9.18% 2.04% 1.02% 7.14% 6.80% 2.04% 5.10% 2.04% 1.02% 3.06% 1.70% 1.36% 4.08% 1.70% 3.06% 1.70% 4.76% 0.00% 0.68%
9.78% 1.09% 0.00% 3.26% 7.61% 4.35% 2.17% 4.35% 2.17% 3.26% 0.00% 1.09% 3.26% 0.00% 3.26% 3.26% 2.17% 0.00% 0.00%
11.76% 0.00% 0.00% 5.88% 5.88% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00%
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Usage Percentage (%)
0.30
0.25
0.20
0.15
0.10
0.05
0.00


Heatmap of Drug Usage Percentage (Used in Last Month) by Age Group and Drug Type
Age Group
18 - 24
25 - 34
35 - 44
45 - 54
55 - 64
65+
18.88% 7.64% 2.34% 10.61% 12.17% 21.37% 8.42% 8.58% 0.62% 15.76% 2.65% 4.37% 11.54% 10.61% 4.52% 12.79% 10.45% 0.16% 1.56%
14.58% 2.92% 1.25% 5.00% 6.67% 12.50% 5.83% 3.96% 0.21% 7.71% 0.83% 2.50% 5.00% 4.79% 3.54% 4.58% 4.79% 0.00% 0.62%
13.80% 1.69% 0.56% 3.38% 4.51% 10.99% 3.10% 5.35% 1.13% 3.38% 0.85% 0.28% 1.97% 0.56% 0.00% 2.54% 2.54% 0.00% 0.00%
12.93% 2.04% 0.34% 3.74% 3.40% 14.63% 1.36% 4.08% 0.00% 1.70% 0.00% 0.34% 1.36% 1.36% 0.00% 0.34% 2.04% 0.00% 0.00%
7.61% 0.00% 0.00% 5.43% 4.35% 15.22% 1.09% 0.00% 0.00% 0.00% 0.00% 0.00% 1.09% 0.00% 2.17% 1.09% 2.17% 0.00% 0.00%
11.76% 0.00% 0.00% 0.00% 0.00% 17.65% 0.00% 5.88% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 5.88% 0.00% 0.00% 0.00% 0.00%
Alcohol_values
Amphet_values
Amyl_values
Benzos_values
Cannabis_values
Choc_values
Coke_values
Caff_values
Crack_values
Ecstasy_values
Heroin_values
Ketamine_values
Legalh_values
LSD_values
Meth_values
Mushrooms_values
Nicotine_values
Semer_values
VSA_values
Drug Type
Usage Percentage (%)
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000

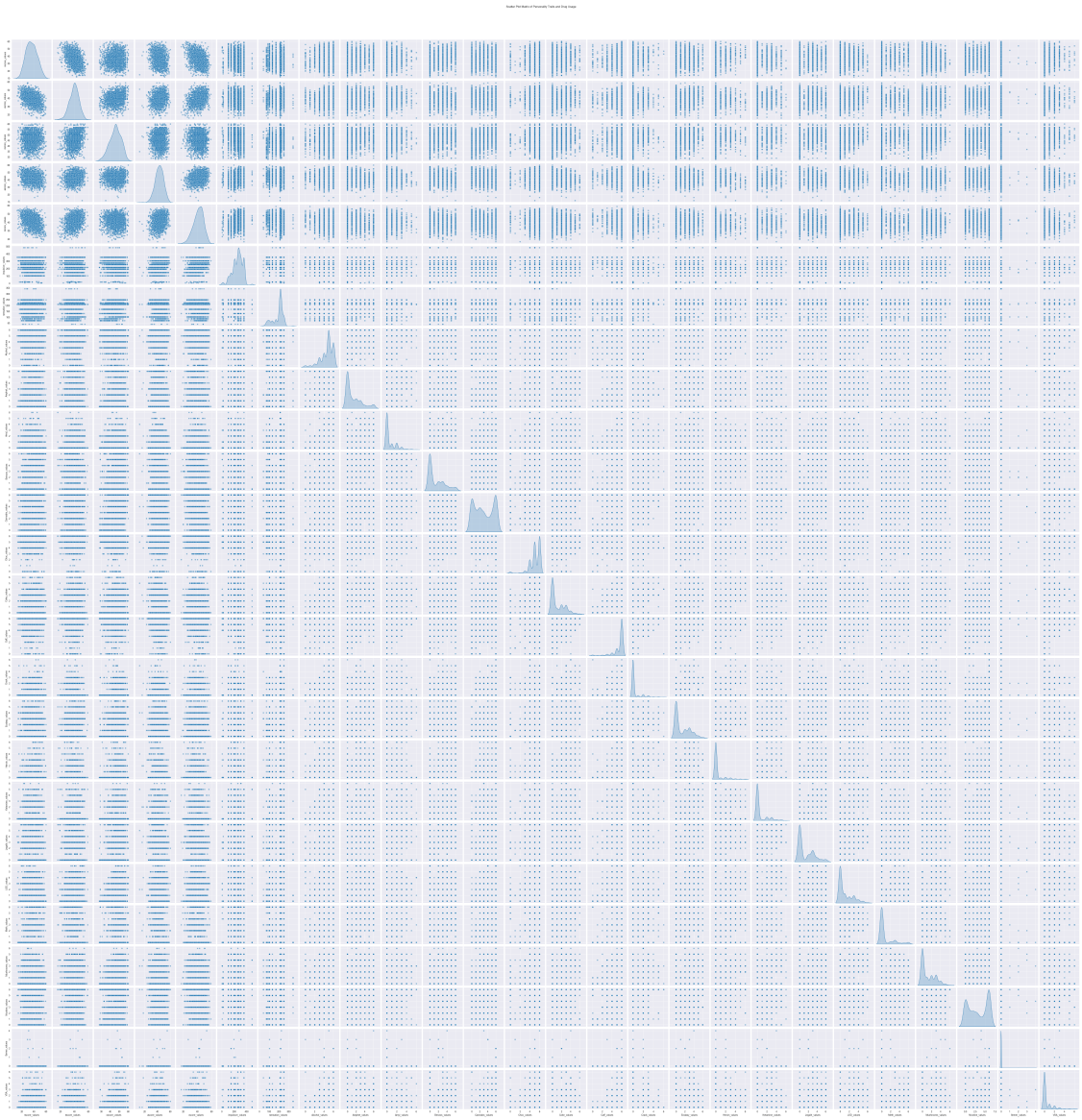


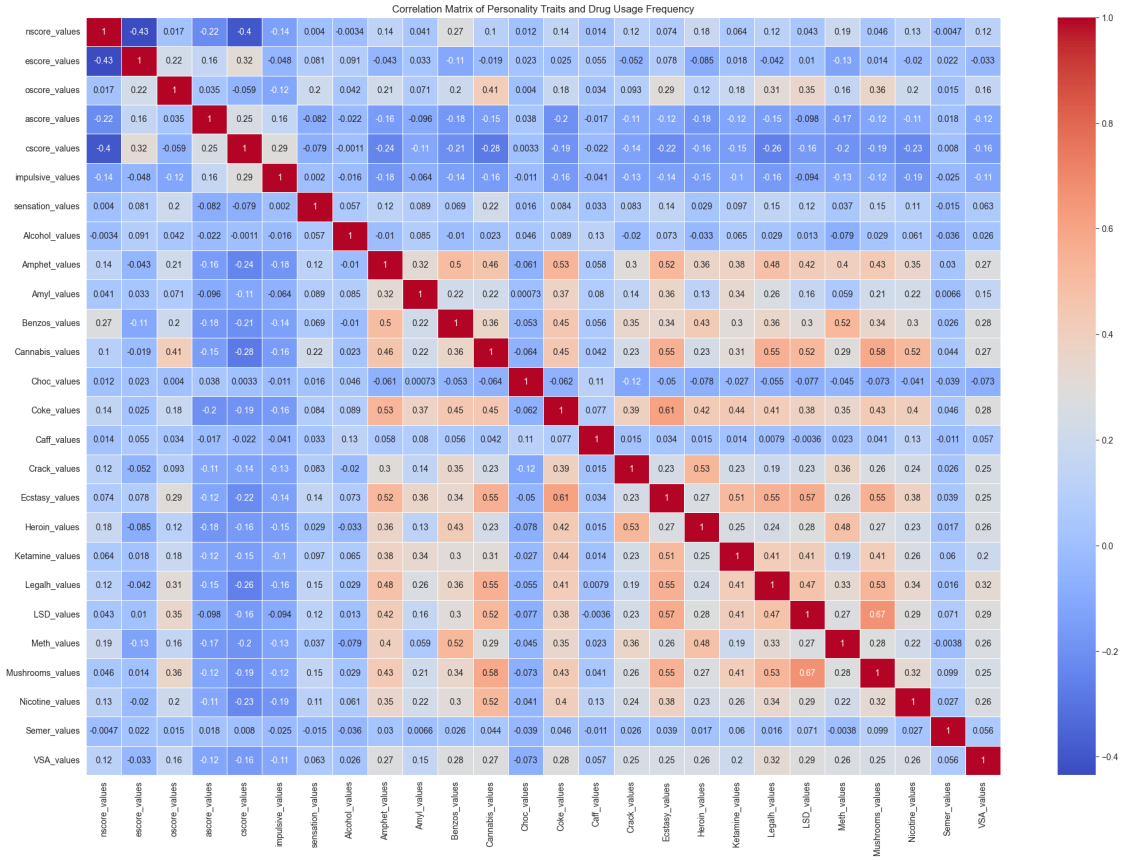


5.2 TASK 2.2: Q2 - Is there a correlation between personality traits and drug usage frequency?

[71]:
```
# Question 2: Correlation Between Personality Traits and Drug Usage Frequency


# Ensuring personality traits and drug usage columns are numeric
# Make a copy of the dataframe for mapping and conversion
data_mapped = df.copy()

# Apply the mapping to the drug usage columns
```

```
for col in drug_columns:
    data_mapped[col] = data_mapped[col].map(usage_mapping)

# Convert all relevant columns to numeric, coercing errors
for col in personality_traits_scores_columns + drug_columns:
    data_mapped[col] = pd.to_numeric(data_mapped[col], errors='coerce')

# Drop rows with missing values in the relevant columns
data_mapped_clean = data_mapped.dropna(subset=personality_traits_scores_columns␣
 ↪+ drug_columns)

# Scatter Plot Matrix (Pairs Plot)
sns.pairplot(data_mapped_clean[personality_traits_scores_columns +␣
 ↪drug_columns], diag_kind='kde', kind='scatter',
             plot_kws={'alpha': 0.5})
plt.suptitle('Scatter Plot Matrix of Personality Traits and Drug Usage', y=1.02)
plt.show()

# Correlation Matrix with Heatmap
correlation_matrix_clean = data_mapped_clean[personality_traits_scores_columns␣
 ↪+ drug_columns].corr()
plt.figure(figsize=(24, 16))
sns.heatmap(correlation_matrix_clean, annot=True, cmap='coolwarm', linewidths=0.
 ↪5)
plt.title('Correlation Matrix of Personality Traits and Drug Usage Frequency')
plt.show()
```


Scatter Plot Matrix of Personality Traits and Drug Usage



[72]:
```
# Apply the mapping to the drug usage columns
for col in drug_columns:
    df.loc[:, col] = df[col].map(usage_mapping)

# Ensure all relevant columns are numeric
for col in personality_traits_scores_columns + drug_columns:
    df.loc[:, col] = pd.to_numeric(df[col], errors='coerce')

# Drop rows with missing values in the selected columns
data_clean = df.dropna(subset=personality_traits_scores_columns + drug_columns)

# Calculate the correlation matrix
correlation_matrix = data_clean[personality_traits_scores_columns +␣
 ↪drug_columns].corr()


# Function to calculate p-values
def calculate_pvalues(df):
    dfcols = pd.DataFrame(columns=df.columns)
    pvalues = dfcols.transpose().join(dfcols, how='outer')
```

```
    for r in df.columns:
        for c in df.columns:
            if df[r].isnull().all() or df[c].isnull().all() or len(df[r].
 ↪unique()) <= 1 or len(df[c].unique()) <= 1:
                pvalues.loc[r, c] = None
            else:
                pvalues.loc[r, c] = round(pearsonr(df[r].dropna(), df[c].
 ↪dropna())[1], 4)
    return pvalues


# Calculate p-values for the correlations
pvalues = calculate_pvalues(data_clean[personality_traits_scores_columns +␣
 ↪drug_columns])

# Display the correlation matrix and p-values
print("Correlation Matrix:\n", correlation_matrix)
print("\nP-Values Matrix:\n", pvalues)

mask = np.zeros_like(correlation_matrix, dtype=bool)
mask[np.triu_indices_from(mask)] = True

# Plot the correlation matrix with heatmap
plt.figure(figsize=(24, 16))
sns.heatmap(correlation_matrix, mask=mask, annot=True, cmap='coolwarm',␣
 ↪linewidths=0.5)
plt.title('Correlation Matrix of Personality Traits and Drug Usage Frequency')
plt.show()
```

```
Correlation Matrix:
                  nscore_values  escore_values  oscore_values  ascore_values  \
nscore_values          1.000000      -0.434433       0.017496      -0.216593
escore_values         -0.434433       1.000000       0.222698       0.159242
oscore_values          0.017496       0.222698       1.000000       0.035290
ascore_values         -0.216593       0.159242       0.035290       1.000000
cscore_values         -0.398768       0.315238      -0.058959       0.250090
impulsive_values      -0.142065      -0.047732      -0.116869       0.163815
sensation_values       0.004036       0.080847       0.203656      -0.081683
Alcohol_values        -0.003362       0.090604       0.042386      -0.021889
Amphet_values          0.135564      -0.043192       0.214022      -0.157710
Amyl_values            0.040909       0.033000       0.070800      -0.095507
Benzos_values          0.274599      -0.108904       0.198489      -0.175409
Cannabis_values        0.102512      -0.019353       0.412245      -0.153998
Choc_values            0.012031       0.023249       0.003961       0.038432
Coke_values            0.144977       0.024839       0.176186      -0.201972
Caff_values            0.014421       0.055298       0.034371      -0.017119
Crack_values           0.116359      -0.052286       0.092511      -0.112849
```

Ecstasy_values	0.074168	0.077681	0.294675	-0.118138
Heroin_values	0.178590	-0.084889	0.117000	-0.179755
Ketamine_values	0.064186	0.017542	0.179477	-0.116749
Legalh_values	0.120252	-0.042151	0.306167	-0.146385
LSD_values	0.042558	0.010447	0.351376	-0.098169
Meth_values	0.188026	-0.127263	0.158483	-0.168429
Mushrooms_values	0.046484	0.013779	0.358467	-0.120387
Nicotine_values	0.130254	-0.020352	0.198731	-0.114659
Semer_values	-0.004684	0.022168	0.015456	0.017825
VSA_values	0.118556	-0.032645	0.157053	-0.119565

	cscore_values	impulsive_values	sensation_values	\
nscore_values	-0.398768	-0.142065	0.004036	
escore_values	0.315238	-0.047732	0.080847	
oscore_values	-0.058959	-0.116869	0.203656	
ascore_values	0.250090	0.163815	-0.081683	
cscore_values	1.000000	0.291051	-0.079489	
impulsive_values	0.291051	1.000000	0.001980	
sensation_values	-0.079489	0.001980	1.000000	
Alcohol_values	-0.001083	-0.015520	0.057263	
Amphet_values	-0.239148	-0.180267	0.124204	
Amyl_values	-0.114146	-0.063902	0.088525	
Benzos_values	-0.207290	-0.139123	0.069360	
Cannabis_values	-0.279231	-0.157647	0.221052	
Choc_values	0.003326	-0.010619	0.015687	
Coke_values	-0.194920	-0.161114	0.084392	
Caff_values	-0.021641	-0.041101	0.032946	
Crack_values	-0.135004	-0.132548	0.082656	
Ecstasy_values	-0.216616	-0.142657	0.143951	
Heroin_values	-0.163454	-0.149049	0.029442	
Ketamine_values	-0.152871	-0.101800	0.097479	
Legalh_values	-0.258477	-0.160646	0.146043	
LSD_values	-0.161002	-0.094178	0.123797	
Meth_values	-0.195394	-0.126107	0.036976	
Mushrooms_values	-0.192862	-0.124842	0.154596	
Nicotine_values	-0.229790	-0.187054	0.109070	
Semer_values	0.008028	-0.024720	-0.014738	
VSA_values	-0.162322	-0.109928	0.063011	

	Alcohol_values	Amphet_values	Amyl_values	…	\
nscore_values	-0.003362	0.135564	0.040909	…	
escore_values	0.090604	-0.043192	0.033000	…	
oscore_values	0.042386	0.214022	0.070800	…	
ascore_values	-0.021889	-0.157710	-0.095507	…	
cscore_values	-0.001083	-0.239148	-0.114146	…	
impulsive_values	-0.015520	-0.180267	-0.063902	…	
sensation_values	0.057263	0.124204	0.088525	…	
Alcohol_values	1.000000	-0.010169	0.085450	…	

```
Amphet_values          -0.010169       1.000000     0.317518  …
Amyl_values             0.085450       0.317518     1.000000  …
Benzos_values          -0.010233       0.499381     0.220477  …
Cannabis_values         0.023422       0.457373     0.220545  …
Choc_values             0.045564      -0.061143     0.000729  …
Coke_values             0.089391       0.532640     0.374506  …
Caff_values             0.126551       0.058208     0.079986  …
Crack_values           -0.019697       0.295130     0.143111  …
Ecstasy_values          0.073301       0.521387     0.356814  …
Heroin_values          -0.033374       0.361854     0.130161  …
Ketamine_values         0.064584       0.375117     0.338023  …
Legalh_values           0.028858       0.476587     0.262292  …
LSD_values              0.012691       0.420789     0.161349  …
Meth_values            -0.078942       0.395788     0.058526  …
Mushrooms_values        0.028519       0.429467     0.214004  …
Nicotine_values         0.060834       0.351802     0.223382  …
Semer_values           -0.036080       0.029713     0.006572  …
VSA_values              0.025559       0.271748     0.154282  …

                  Ecstasy_values  Heroin_values  Ketamine_values  \
nscore_values           0.074168       0.178590         0.064186
escore_values           0.077681      -0.084889         0.017542
oscore_values           0.294675       0.117000         0.179477
ascore_values          -0.118138      -0.179755        -0.116749
cscore_values          -0.216616      -0.163454        -0.152871
impulsive_values       -0.142657      -0.149049        -0.101800
sensation_values        0.143951       0.029442         0.097479
Alcohol_values          0.073301      -0.033374         0.064584
Amphet_values           0.521387       0.361854         0.375117
Amyl_values             0.356814       0.130161         0.338023
Benzos_values           0.342832       0.427763         0.300114
Cannabis_values         0.552516       0.233256         0.309891
Choc_values            -0.049516      -0.077728        -0.027271
Coke_values             0.609946       0.424655         0.439511
Caff_values             0.034326       0.015226         0.013734
Crack_values            0.232557       0.526926         0.230725
Ecstasy_values          1.000000       0.265967         0.507147
Heroin_values           0.265967       1.000000         0.250868
Ketamine_values         0.507147       0.250868         1.000000
Legalh_values           0.554973       0.241602         0.413323
LSD_values              0.571132       0.279097         0.410969
Meth_values             0.259047       0.478885         0.188749
Mushrooms_values        0.548400       0.267700         0.412616
Nicotine_values         0.380524       0.225008         0.256039
Semer_values            0.039139       0.016894         0.059669
VSA_values              0.253497       0.256761         0.198435

                  Legalh_values  LSD_values  Meth_values  Mushrooms_values  \
```

nscore_values	0.120252	0.042558	0.188026	0.046484
escore_values	-0.042151	0.010447	-0.127263	0.013779
oscore_values	0.306167	0.351376	0.158483	0.358467
ascore_values	-0.146385	-0.098169	-0.168429	-0.120387
cscore_values	-0.258477	-0.161002	-0.195394	-0.192862
impulsive_values	-0.160646	-0.094178	-0.126107	-0.124842
sensation_values	0.146043	0.123797	0.036976	0.154596
Alcohol_values	0.028858	0.012691	-0.078942	0.028519
Amphet_values	0.476587	0.420789	0.395788	0.429467
Amyl_values	0.262292	0.161349	0.058526	0.214004
Benzos_values	0.358030	0.303349	0.518122	0.344943
Cannabis_values	0.553959	0.520839	0.293753	0.579882
Choc_values	-0.054537	-0.077453	-0.044753	-0.072782
Coke_values	0.413272	0.383533	0.345316	0.426918
Caff_values	0.007878	-0.003626	0.023378	0.041268
Crack_values	0.193986	0.229790	0.356384	0.261011
Ecstasy_values	0.554973	0.571132	0.259047	0.548400
Heroin_values	0.241602	0.279097	0.478885	0.267700
Ketamine_values	0.413323	0.410969	0.188749	0.412616
Legalh_values	1.000000	0.471239	0.325316	0.530902
LSD_values	0.471239	1.000000	0.267284	0.668226
Meth_values	0.325316	0.267284	1.000000	0.277664
Mushrooms_values	0.530902	0.668226	0.277664	1.000000
Nicotine_values	0.342646	0.293266	0.220128	0.324612
Semer_values	0.015864	0.071185	-0.003822	0.099069
VSA_values	0.320792	0.287321	0.260421	0.246207

	Nicotine_values	Semer_values	VSA_values
nscore_values	0.130254	-0.004684	0.118556
escore_values	-0.020352	0.022168	-0.032645
oscore_values	0.198731	0.015456	0.157053
ascore_values	-0.114659	0.017825	-0.119565
cscore_values	-0.229790	0.008028	-0.162322
impulsive_values	-0.187054	-0.024720	-0.109928
sensation_values	0.109070	-0.014738	0.063011
Alcohol_values	0.060834	-0.036080	0.025559
Amphet_values	0.351802	0.029713	0.271748
Amyl_values	0.223382	0.006572	0.154282
Benzos_values	0.302845	0.026105	0.276185
Cannabis_values	0.515911	0.043922	0.273478
Choc_values	-0.041459	-0.039440	-0.073294
Coke_values	0.402182	0.046468	0.283765
Caff_values	0.126371	-0.010999	0.057300
Crack_values	0.241727	0.026375	0.250956
Ecstasy_values	0.380524	0.039139	0.253497
Heroin_values	0.225008	0.016894	0.256761
Ketamine_values	0.256039	0.059669	0.198435
Legalh_values	0.342646	0.015864	0.320792

```
LSD_values               0.293266      0.071185    0.287321
Meth_values              0.220128     -0.003822    0.260421
Mushrooms_values         0.324612      0.099069    0.246207
Nicotine_values          1.000000      0.026855    0.255802
Semer_values             0.026855      1.000000    0.056207
VSA_values               0.255802      0.056207    1.000000

[26 rows x 26 columns]

P-Values Matrix:
                  nscore_values escore_values oscore_values ascore_values  \
Alcohol_values           0.8842        0.0001        0.0662         0.343
Amphet_values               0.0        0.0612           0.0           0.0
Amyl_values              0.0763        0.1527        0.0021           0.0
Benzos_values               0.0           0.0           0.0           0.0
Caff_values              0.5321        0.0165        0.1364        0.4583
Cannabis_values             0.0        0.4018           0.0           0.0
Choc_values              0.6022        0.3138        0.8638        0.0958
Coke_values                 0.0        0.2818           0.0           0.0
Crack_values                0.0        0.0234        0.0001           0.0
Ecstasy_values           0.0013        0.0008           0.0           0.0
Heroin_values               0.0        0.0002           0.0           0.0
Ketamine_values          0.0054        0.4473           0.0           0.0
LSD_values               0.0651        0.6509           0.0           0.0
Legalh_values               0.0        0.0677           0.0           0.0
Meth_values                 0.0           0.0           0.0           0.0
Mushrooms_values         0.0439        0.5506           0.0           0.0
Nicotine_values             0.0        0.3779           0.0           0.0
Semer_values             0.8392        0.3368        0.5031          0.44
VSA_values                  0.0        0.1572           0.0           0.0
ascore_values               0.0           0.0        0.1262           0.0
cscore_values               0.0           0.0        0.0106           0.0
escore_values               0.0           0.0           0.0           0.0
impulsive_values            0.0        0.0386           0.0           0.0
nscore_values               0.0           0.0        0.4485           0.0
oscore_values            0.4485           0.0           0.0        0.1262
sensation_values         0.8612        0.0005           0.0        0.0004

                  cscore_values impulsive_values sensation_values  \
Alcohol_values           0.9626           0.5014            0.013
Amphet_values               0.0              0.0              0.0
Amyl_values                 0.0           0.0056           0.0001
Benzos_values               0.0              0.0           0.0026
Caff_values              0.3485           0.0749           0.1534
Cannabis_values             0.0              0.0              0.0
Choc_values              0.8854           0.6455           0.4968
Coke_values                 0.0              0.0           0.0002
Crack_values                0.0              0.0           0.0003
```

Ecstasy_values	0.0	0.0	0.0
Heroin_values	0.0	0.0	0.2021
Ketamine_values	0.0	0.0	0.0
LSD_values	0.0	0.0	0.0
Legalh_values	0.0	0.0	0.0
Meth_values	0.0	0.0	0.1091
Mushrooms_values	0.0	0.0	0.0
Nicotine_values	0.0	0.0	0.0
Semer_values	0.728	0.2842	0.5232
VSA_values	0.0	0.0	0.0063
ascore_values	0.0	0.0	0.0004
cscore_values	0.0	0.0	0.0006
escore_values	0.0	0.0386	0.0005
impulsive_values	0.0	0.0	0.9316
nscore_values	0.0	0.0	0.8612
oscore_values	0.0106	0.0	0.0
sensation_values	0.0006	0.9316	0.0

	Alcohol_values	Amphet_values	Amyl_values	…	Ecstasy_values	\
Alcohol_values	0.0	0.6596	0.0002	…	0.0015	
Amphet_values	0.6596	0.0	0.0	…	0.0	
Amyl_values	0.0002	0.0	0.0	…	0.0	
Benzos_values	0.6576	0.0	0.0	…	0.0	
Caff_values	0.0	0.0116	0.0005	…	0.1369	
Cannabis_values	0.3102	0.0	0.0	…	0.0	
Choc_values	0.0483	0.008	0.9748	…	0.0319	
Coke_values	0.0001	0.0	0.0	…	0.0	
Crack_values	0.3935	0.0	0.0	…	0.0	
Ecstasy_values	0.0015	0.0	0.0	…	0.0	
Heroin_values	0.1481	0.0	0.0	…	0.0	
Ketamine_values	0.0051	0.0	0.0	…	0.0	
LSD_values	0.5825	0.0	0.0	…	0.0	
Legalh_values	0.2112	0.0	0.0	…	0.0	
Meth_values	0.0006	0.0	0.0112	…	0.0	
Mushrooms_values	0.2166	0.0	0.0	…	0.0	
Nicotine_values	0.0083	0.0	0.0	…	0.0	
Semer_values	0.1179	0.198	0.7759	…	0.0899	
VSA_values	0.2681	0.0	0.0	…	0.0	
ascore_values	0.343	0.0	0.0	…	0.0	
cscore_values	0.9626	0.0	0.0	…	0.0	
escore_values	0.0001	0.0612	0.1527	…	0.0008	
impulsive_values	0.5014	0.0	0.0056	…	0.0	
nscore_values	0.8842	0.0	0.0763	…	0.0013	
oscore_values	0.0662	0.0	0.0021	…	0.0	
sensation_values	0.013	0.0	0.0001	…	0.0	

	Heroin_values	Ketamine_values	Legalh_values	LSD_values	\
Alcohol_values	0.1481	0.0051	0.2112	0.5825	

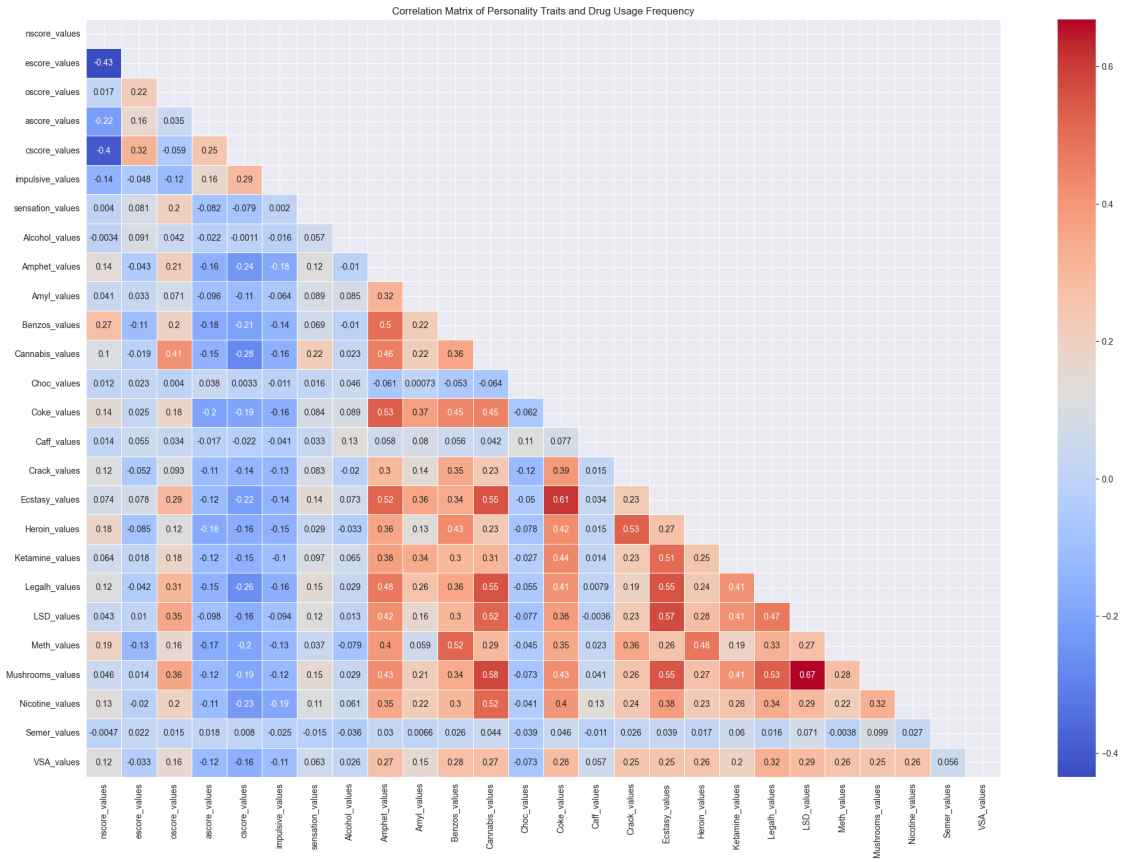
Amphet_values	0.0	0.0	0.0	0.0
Amyl_values	0.0	0.0	0.0	0.0
Benzos_values	0.0	0.0	0.0	0.0
Caff_values	0.5095	0.5519	0.7329	0.8752
Cannabis_values	0.0	0.0	0.0	0.0
Choc_values	0.0007	0.2374	0.0181	0.0008
Coke_values	0.0	0.0	0.0	0.0
Crack_values	0.0	0.0	0.0	0.0
Ecstasy_values	0.0	0.0	0.0	0.0
Heroin_values	0.0	0.0	0.0	0.0
Ketamine_values	0.0	0.0	0.0	0.0
LSD_values	0.0	0.0	0.0	0.0
Legalh_values	0.0	0.0	0.0	0.0
Meth_values	0.0	0.0	0.0	0.0
Mushrooms_values	0.0	0.0	0.0	0.0
Nicotine_values	0.0	0.0	0.0	0.0
Semer_values	0.4643	0.0097	0.4919	0.002
VSA_values	0.0	0.0	0.0	0.0
ascore_values	0.0	0.0	0.0	0.0
cscore_values	0.0	0.0	0.0	0.0
escore_values	0.0002	0.4473	0.0677	0.6509
impulsive_values	0.0	0.0	0.0	0.0
nscore_values	0.0	0.0054	0.0	0.0651
oscore_values	0.0	0.0	0.0	0.0
sensation_values	0.2021	0.0	0.0	0.0

	Meth_values	Mushrooms_values	Nicotine_values	Semer_values	\
Alcohol_values	0.0006	0.2166	0.0083	0.1179	
Amphet_values	0.0	0.0	0.0	0.198	
Amyl_values	0.0112	0.0	0.0	0.7759	
Benzos_values	0.0	0.0	0.0	0.258	
Caff_values	0.3111	0.0737	0.0	0.6337	
Cannabis_values	0.0	0.0	0.0	0.057	
Choc_values	0.0524	0.0016	0.0724	0.0874	
Coke_values	0.0	0.0	0.0	0.044	
Crack_values	0.0	0.0	0.0	0.2531	
Ecstasy_values	0.0	0.0	0.0	0.0899	
Heroin_values	0.0	0.0	0.0	0.4643	
Ketamine_values	0.0	0.0	0.0	0.0097	
LSD_values	0.0	0.0	0.0	0.002	
Legalh_values	0.0	0.0	0.0	0.4919	
Meth_values	0.0	0.0	0.0	0.8685	
Mushrooms_values	0.0	0.0	0.0	0.0	
Nicotine_values	0.0	0.0	0.0	0.2446	
Semer_values	0.8685	0.0	0.2446	0.0	
VSA_values	0.0	0.0	0.0	0.0148	
ascore_values	0.0	0.0	0.0	0.44	
cscore_values	0.0	0.0	0.0	0.728	

escore_values	0.0	0.5506	0.3779	0.3368
impulsive_values	0.0	0.0	0.0	0.2842
nscore_values	0.0	0.0439	0.0	0.8392
oscore_values	0.0	0.0	0.0	0.5031
sensation_values	0.1091	0.0	0.0	0.5232

	VSA_values
Alcohol_values	0.2681
Amphet_values	0.0
Amyl_values	0.0
Benzos_values	0.0
Caff_values	0.013
Cannabis_values	0.0
Choc_values	0.0015
Coke_values	0.0
Crack_values	0.0
Ecstasy_values	0.0
Heroin_values	0.0
Ketamine_values	0.0
LSD_values	0.0
Legalh_values	0.0
Meth_values	0.0
Mushrooms_values	0.0
Nicotine_values	0.0
Semer_values	0.0148
VSA_values	0.0
ascore_values	0.0
cscore_values	0.0
escore_values	0.1572
impulsive_values	0.0
nscore_values	0.0
oscore_values	0.0
sensation_values	0.0063

[26 rows x 26 columns]



[73]:
```
correlation_matrix_drugs = correlation_matrix[drug_columns]
correlation_matrix_drugs = correlation_matrix_drugs.loc[drug_columns]
correlation_matrix_drugs
```

[73]:

	Alcohol_values	Amphet_values	Amyl_values	Benzos_values	\
Alcohol_values	1.000000	-0.010169	0.085450	-0.010233	
Amphet_values	-0.010169	1.000000	0.317518	0.499381	
Amyl_values	0.085450	0.317518	1.000000	0.220477	
Benzos_values	-0.010233	0.499381	0.220477	1.000000	
Cannabis_values	0.023422	0.457373	0.220545	0.357383	
Choc_values	0.045564	-0.061143	0.000729	-0.052533	
Coke_values	0.089391	0.532640	0.374506	0.445307	
Caff_values	0.126551	0.058208	0.079986	0.055862	
Crack_values	-0.019697	0.295130	0.143111	0.348196	
Ecstasy_values	0.073301	0.521387	0.356814	0.342832	
Heroin_values	-0.033374	0.361854	0.130161	0.427763	
Ketamine_values	0.064584	0.375117	0.338023	0.300114	
Legalh_values	0.028858	0.476587	0.262292	0.358030	
LSD_values	0.012691	0.420789	0.161349	0.303349	
Meth_values	-0.078942	0.395788	0.058526	0.518122	

Mushrooms_values	0.028519	0.429467	0.214004	0.344943
Nicotine_values	0.060834	0.351802	0.223382	0.302845
Semer_values	-0.036080	0.029713	0.006572	0.026105
VSA_values	0.025559	0.271748	0.154282	0.276185

	Cannabis_values	Choc_values	Coke_values	Caff_values	\
Alcohol_values	0.023422	0.045564	0.089391	0.126551	
Amphet_values	0.457373	-0.061143	0.532640	0.058208	
Amyl_values	0.220545	0.000729	0.374506	0.079986	
Benzos_values	0.357383	-0.052533	0.445307	0.055862	
Cannabis_values	1.000000	-0.064032	0.448739	0.041578	
Choc_values	-0.064032	1.000000	-0.062333	0.114660	
Coke_values	0.448739	-0.062333	1.000000	0.077411	
Caff_values	0.041578	0.114660	0.077411	1.000000	
Crack_values	0.234233	-0.117590	0.390782	0.015085	
Ecstasy_values	0.552516	-0.049516	0.609946	0.034326	
Heroin_values	0.233256	-0.077728	0.424655	0.015226	
Ketamine_values	0.309891	-0.027271	0.439511	0.013734	
Legalh_values	0.553959	-0.054537	0.413272	0.007878	
LSD_values	0.520839	-0.077453	0.383533	-0.003626	
Meth_values	0.293753	-0.044753	0.345316	0.023378	
Mushrooms_values	0.579882	-0.072782	0.426918	0.041268	
Nicotine_values	0.515911	-0.041459	0.402182	0.126371	
Semer_values	0.043922	-0.039440	0.046468	-0.010999	
VSA_values	0.273478	-0.073294	0.283765	0.057300	

	Crack_values	Ecstasy_values	Heroin_values	\
Alcohol_values	-0.019697	0.073301	-0.033374	
Amphet_values	0.295130	0.521387	0.361854	
Amyl_values	0.143111	0.356814	0.130161	
Benzos_values	0.348196	0.342832	0.427763	
Cannabis_values	0.234233	0.552516	0.233256	
Choc_values	-0.117590	-0.049516	-0.077728	
Coke_values	0.390782	0.609946	0.424655	
Caff_values	0.015085	0.034326	0.015226	
Crack_values	1.000000	0.232557	0.526926	
Ecstasy_values	0.232557	1.000000	0.265967	
Heroin_values	0.526926	0.265967	1.000000	
Ketamine_values	0.230725	0.507147	0.250868	
Legalh_values	0.193986	0.554973	0.241602	
LSD_values	0.229790	0.571132	0.279097	
Meth_values	0.356384	0.259047	0.478885	
Mushrooms_values	0.261011	0.548400	0.267700	
Nicotine_values	0.241727	0.380524	0.225008	
Semer_values	0.026375	0.039139	0.016894	
VSA_values	0.250956	0.253497	0.256761	

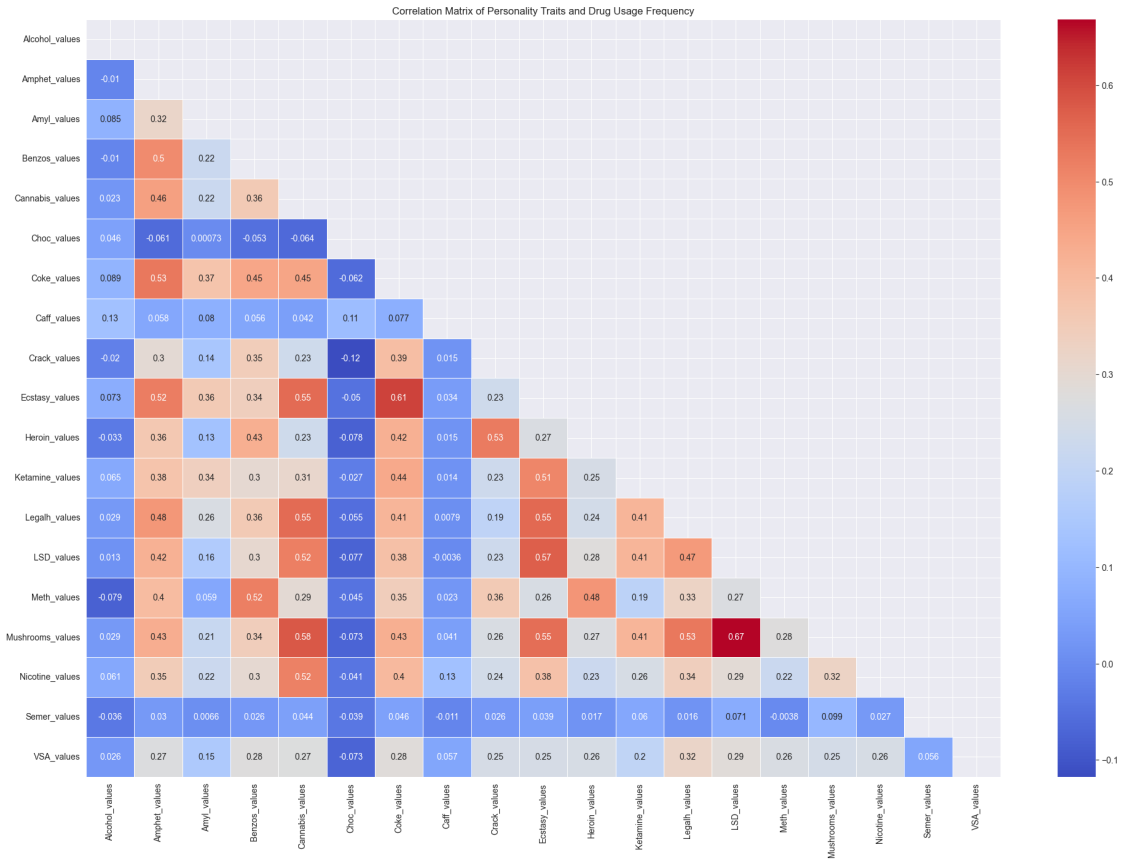
	Ketamine_values	Legalh_values	LSD_values	Meth_values	\
Alcohol_values	0.064584	0.028858	0.012691	-0.078942	
Amphet_values	0.375117	0.476587	0.420789	0.395788	
Amyl_values	0.338023	0.262292	0.161349	0.058526	
Benzos_values	0.300114	0.358030	0.303349	0.518122	
Cannabis_values	0.309891	0.553959	0.520839	0.293753	
Choc_values	-0.027271	-0.054537	-0.077453	-0.044753	
Coke_values	0.439511	0.413272	0.383533	0.345316	
Caff_values	0.013734	0.007878	-0.003626	0.023378	
Crack_values	0.230725	0.193986	0.229790	0.356384	
Ecstasy_values	0.507147	0.554973	0.571132	0.259047	
Heroin_values	0.250868	0.241602	0.279097	0.478885	
Ketamine_values	1.000000	0.413323	0.410969	0.188749	
Legalh_values	0.413323	1.000000	0.471239	0.325316	
LSD_values	0.410969	0.471239	1.000000	0.267284	
Meth_values	0.188749	0.325316	0.267284	1.000000	
Mushrooms_values	0.412616	0.530902	0.668226	0.277664	
Nicotine_values	0.256039	0.342646	0.293266	0.220128	
Semer_values	0.059669	0.015864	0.071185	-0.003822	
VSA_values	0.198435	0.320792	0.287321	0.260421	

	Mushrooms_values	Nicotine_values	Semer_values	VSA_values
Alcohol_values	0.028519	0.060834	-0.036080	0.025559
Amphet_values	0.429467	0.351802	0.029713	0.271748
Amyl_values	0.214004	0.223382	0.006572	0.154282
Benzos_values	0.344943	0.302845	0.026105	0.276185
Cannabis_values	0.579882	0.515911	0.043922	0.273478
Choc_values	-0.072782	-0.041459	-0.039440	-0.073294
Coke_values	0.426918	0.402182	0.046468	0.283765
Caff_values	0.041268	0.126371	-0.010999	0.057300
Crack_values	0.261011	0.241727	0.026375	0.250956
Ecstasy_values	0.548400	0.380524	0.039139	0.253497
Heroin_values	0.267700	0.225008	0.016894	0.256761
Ketamine_values	0.412616	0.256039	0.059669	0.198435
Legalh_values	0.530902	0.342646	0.015864	0.320792
LSD_values	0.668226	0.293266	0.071185	0.287321
Meth_values	0.277664	0.220128	-0.003822	0.260421
Mushrooms_values	1.000000	0.324612	0.099069	0.246207
Nicotine_values	0.324612	1.000000	0.026855	0.255802
Semer_values	0.099069	0.026855	1.000000	0.056207
VSA_values	0.246207	0.255802	0.056207	1.000000

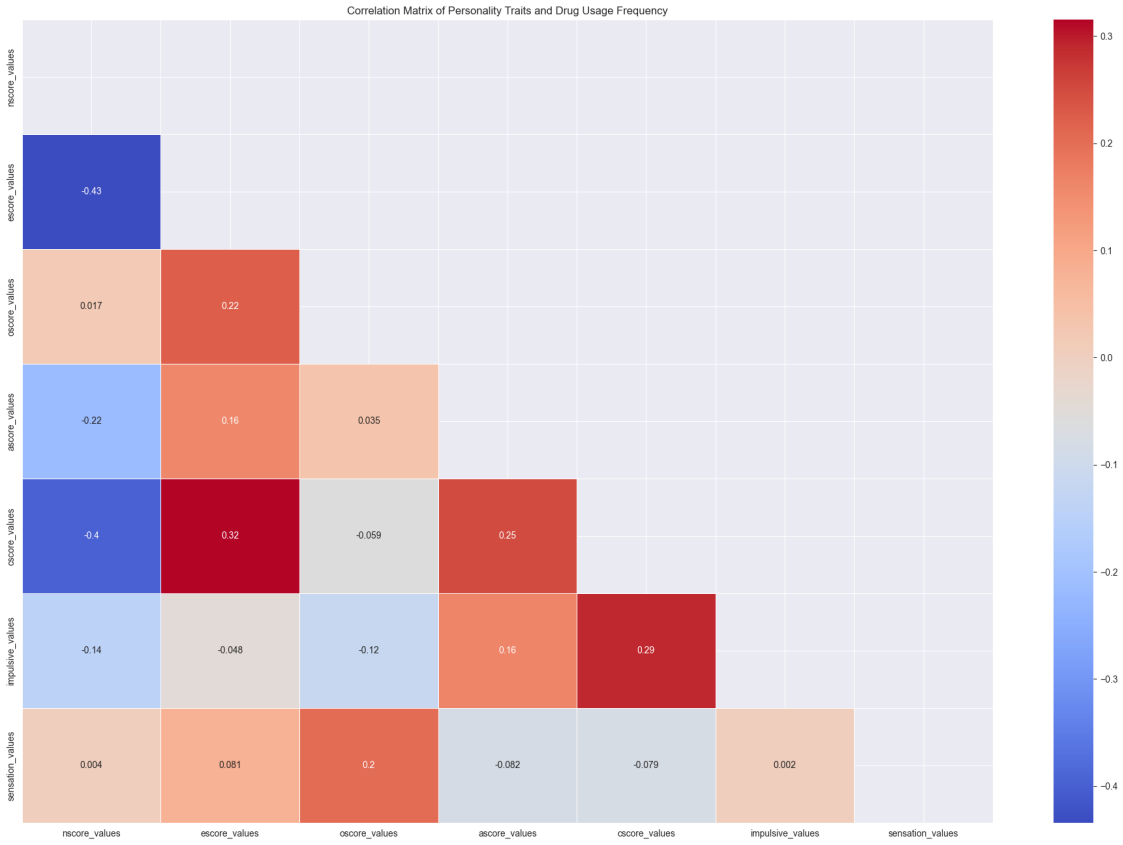
[74]:
```
correlation_matrix_personality_traits =␣
 ↪correlation_matrix[personality_traits_scores_columns]
correlation_matrix_personality_traits = correlation_matrix_personality_traits.
 ↪loc[personality_traits_scores_columns]
correlation_matrix_personality_traits
```

[74]:
```
                   nscore_values  escore_values  oscore_values  ascore_values  \
nscore_values           1.000000      -0.434433       0.017496      -0.216593
escore_values          -0.434433       1.000000       0.222698       0.159242
oscore_values           0.017496       0.222698       1.000000       0.035290
ascore_values          -0.216593       0.159242       0.035290       1.000000
cscore_values          -0.398768       0.315238      -0.058959       0.250090
impulsive_values       -0.142065      -0.047732      -0.116869       0.163815
sensation_values        0.004036       0.080847       0.203656      -0.081683

                   cscore_values  impulsive_values  sensation_values
nscore_values          -0.398768         -0.142065          0.004036
escore_values           0.315238         -0.047732          0.080847
oscore_values          -0.058959         -0.116869          0.203656
ascore_values           0.250090          0.163815         -0.081683
cscore_values           1.000000          0.291051         -0.079489
impulsive_values        0.291051          1.000000          0.001980
sensation_values       -0.079489          0.001980          1.000000
```

[75]:
```
mask = np.zeros_like(correlation_matrix_drugs, dtype=bool)
mask[np.triu_indices_from(mask)] = True

# Plot the correlation matrix with heatmap
plt.figure(figsize=(24, 16))
sns.heatmap(correlation_matrix_drugs, mask=mask, annot=True, cmap='coolwarm',␣
 ↪linewidths=0.5)
plt.title('Correlation Matrix of Personality Traits and Drug Usage Frequency')
plt.show()
```

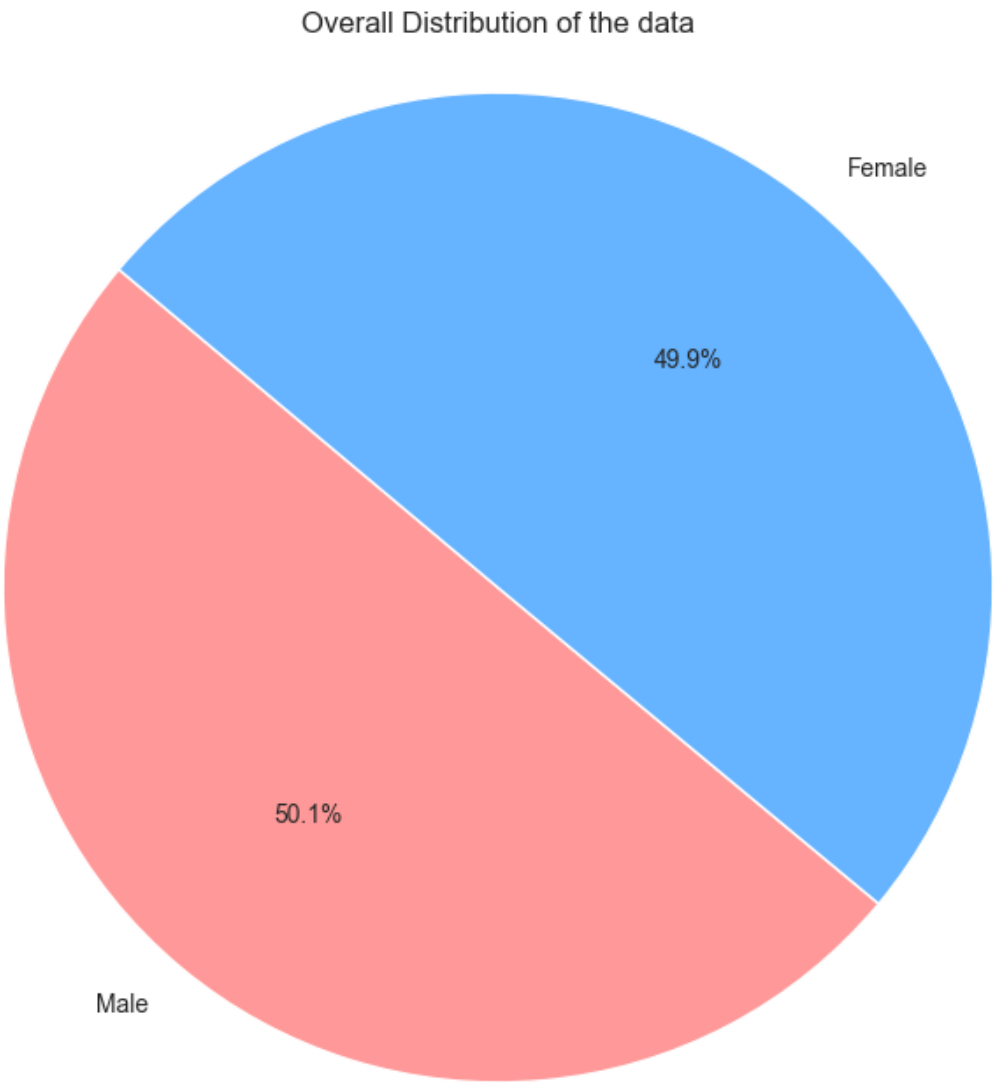


[76]:
```
mask = np.zeros_like(correlation_matrix_personality_traits, dtype=bool)
mask[np.triu_indices_from(mask)] = True

# Plot the correlation matrix with heatmap
plt.figure(figsize=(24, 16))
sns.heatmap(correlation_matrix_personality_traits, mask=mask, annot=True,␣
 ↪cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix of Personality Traits and Drug Usage Frequency')
plt.show()
```



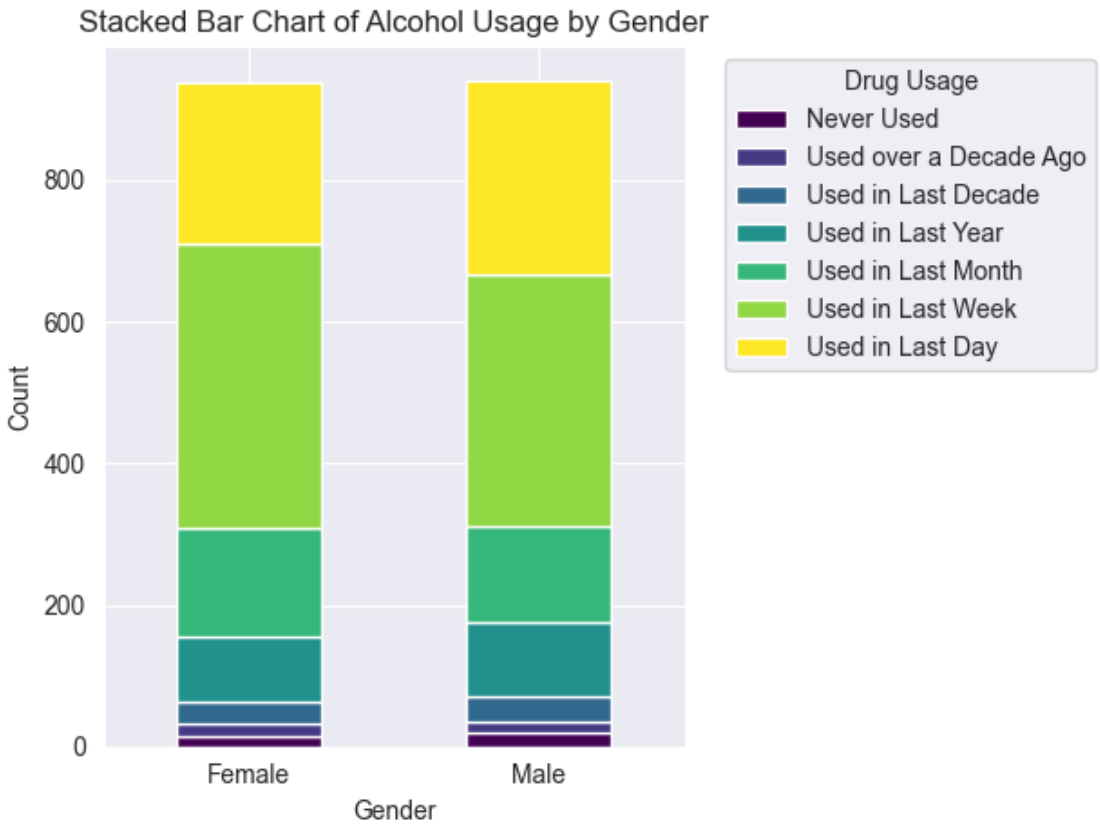
5.3 TASK 2.3: Q3 - What is the gender distribution in the frequency of drug usage?

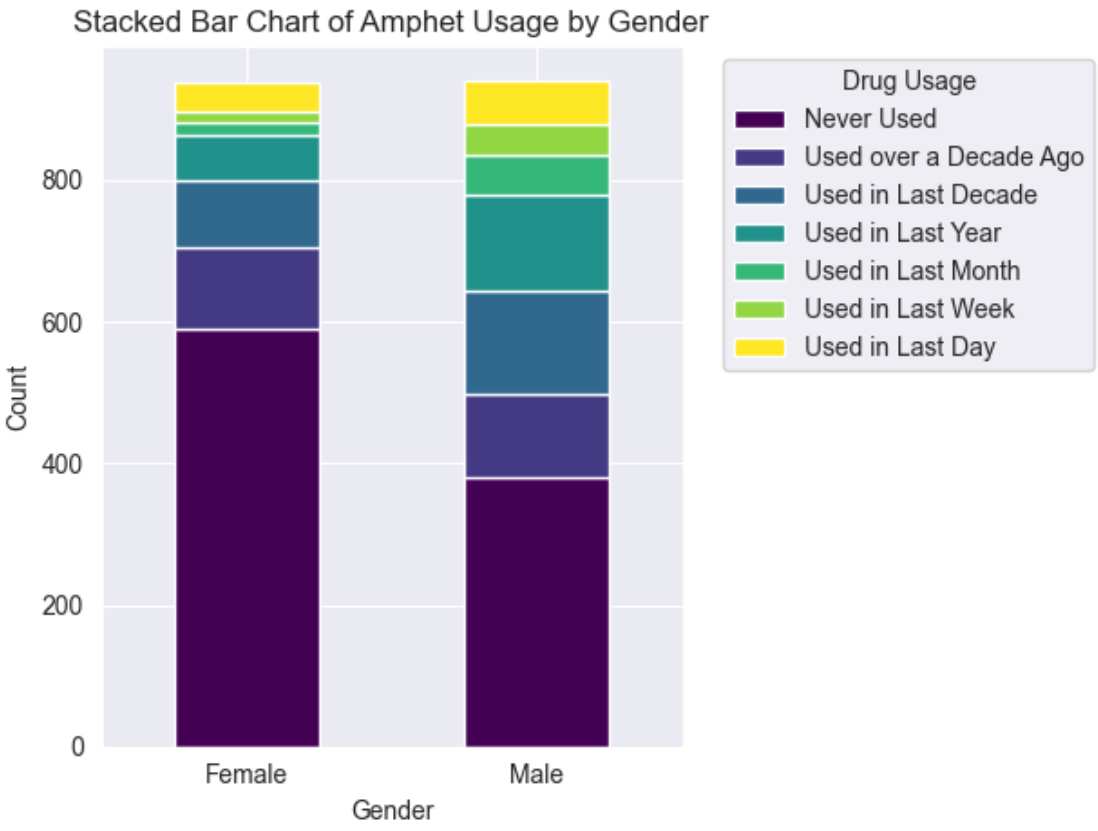
[77]:
```
# Question 3: Gender Distribution in Drug Usage Frequency
# Pie Chart for Nicotine Usage by Gender

gender_distribution = df['gender_values'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(gender_distribution, labels=gender_distribution.index, autopct='%1.
  ↪1f%%', startangle=140,
        colors=['#ff9999', '#66b3ff', '#99ff99', '#ffcc99'])
plt.title('Overall Distribution of the data')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

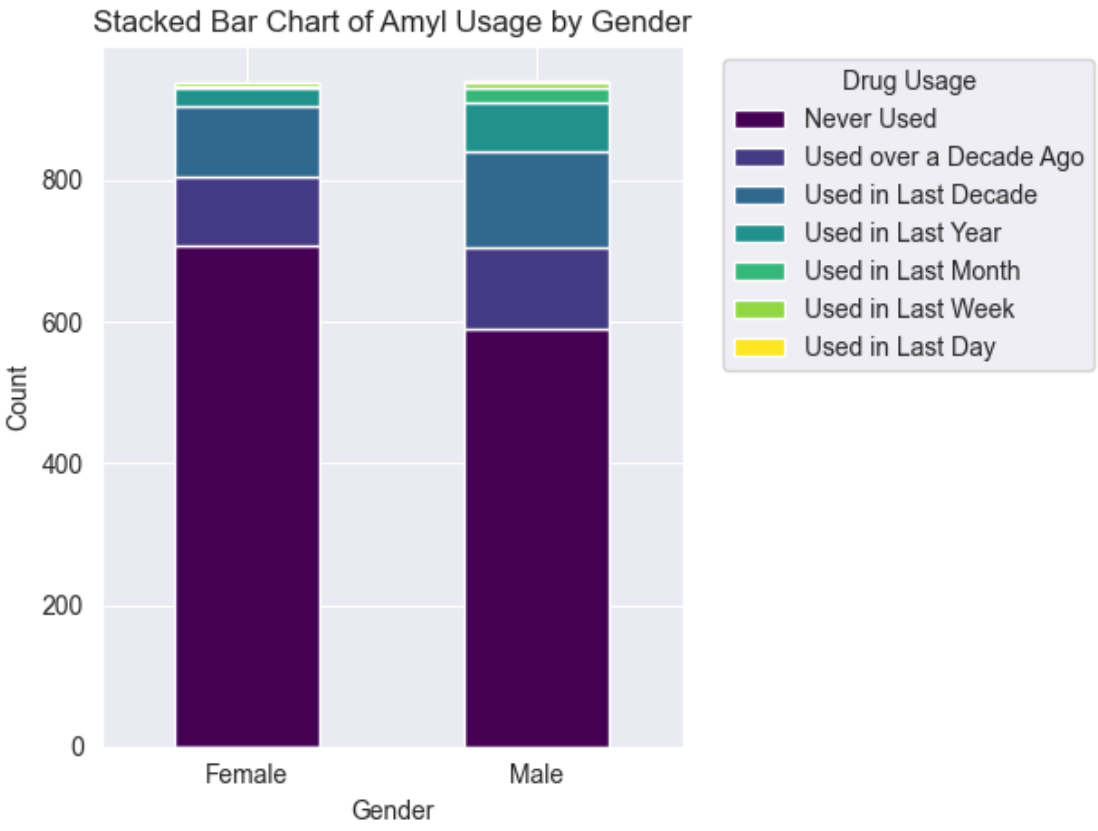


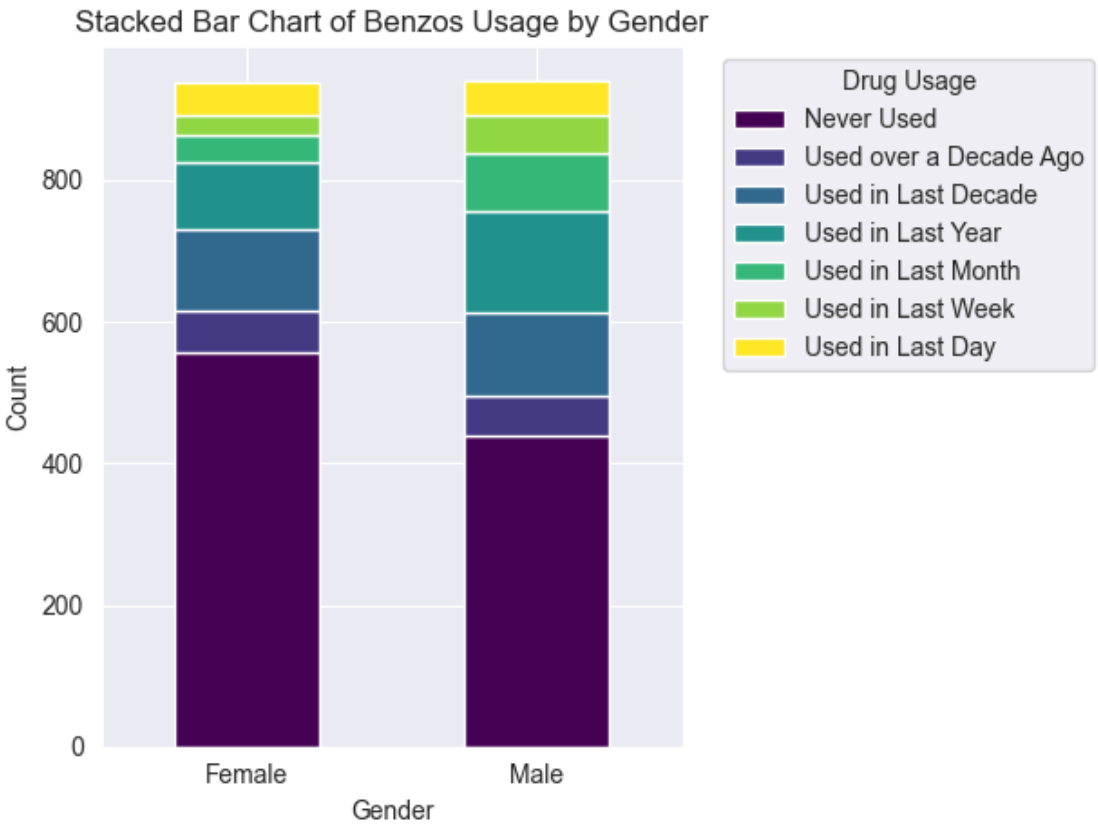
[78]:
```
# Prepare ordered category list for consistent plotting and legend
category_order = [CATEGORIES[key] for key in sorted(CATEGORIES.keys())]
for drug in drug_columns:
    # Prepare data for plotting
    plot_data = df.groupby(['gender_values', drug], observed=True).size().
 ↪unstack(fill_value=0)

    # Convert the columns to descriptive labels for the plot only
    plot_data = plot_data.rename(columns=CATEGORIES)

    # Create a stacked bar chart
    ax = plot_data.plot(kind='bar', stacked=True, colormap='viridis')
```

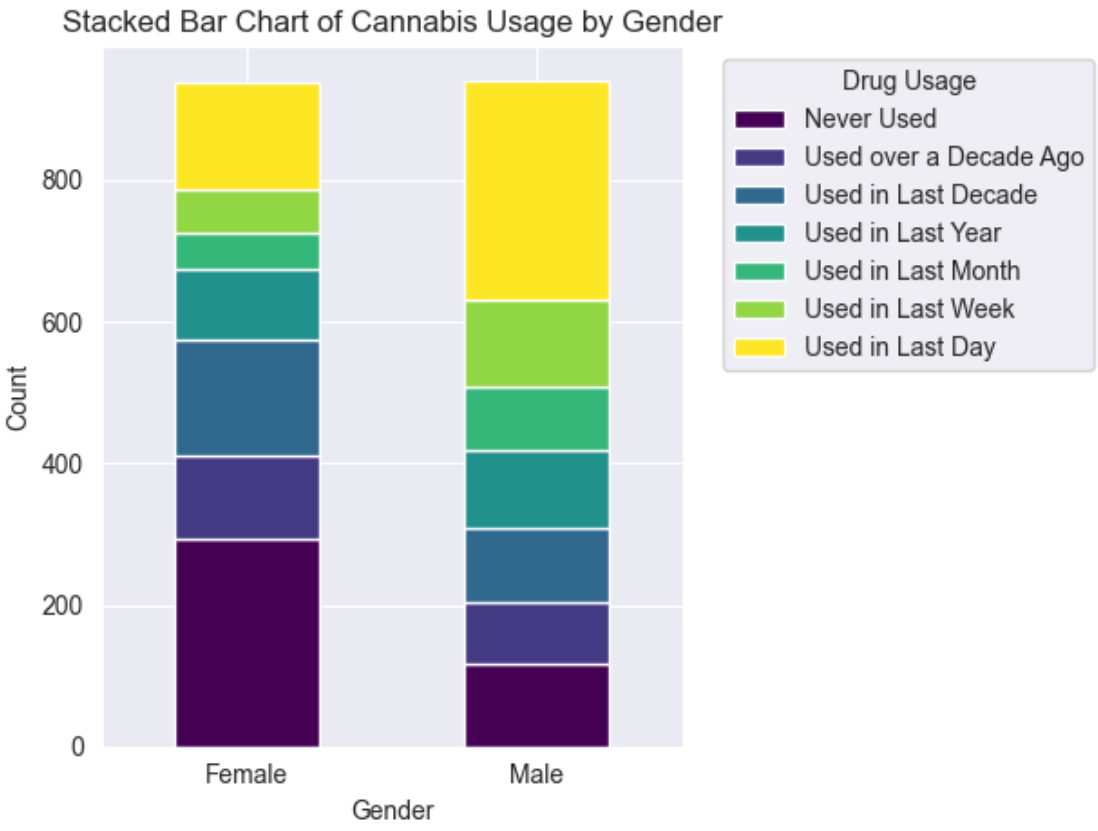
```
    plt.title(f'Stacked Bar Chart of {drug.replace("_values", "")} Usage by␣
↪Gender')
    plt.xlabel('Gender')
    plt.ylabel('Count')
    plt.xticks(rotation=0)  # Keep the gender labels horizontal for readability

    # Set legend directly using the category order
    plt.legend(category_order, title='Drug Usage', bbox_to_anchor=(1.05, 1),␣
↪loc='upper left')

    plt.tight_layout()
    plt.show()
```

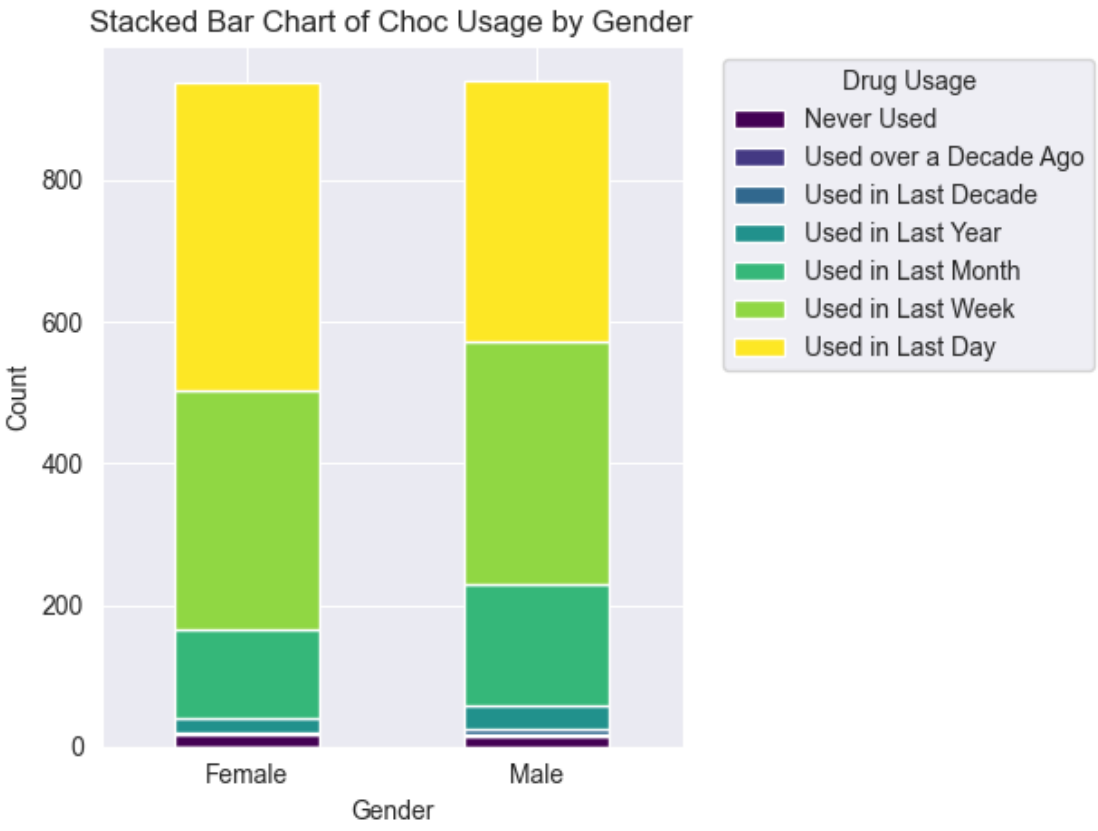


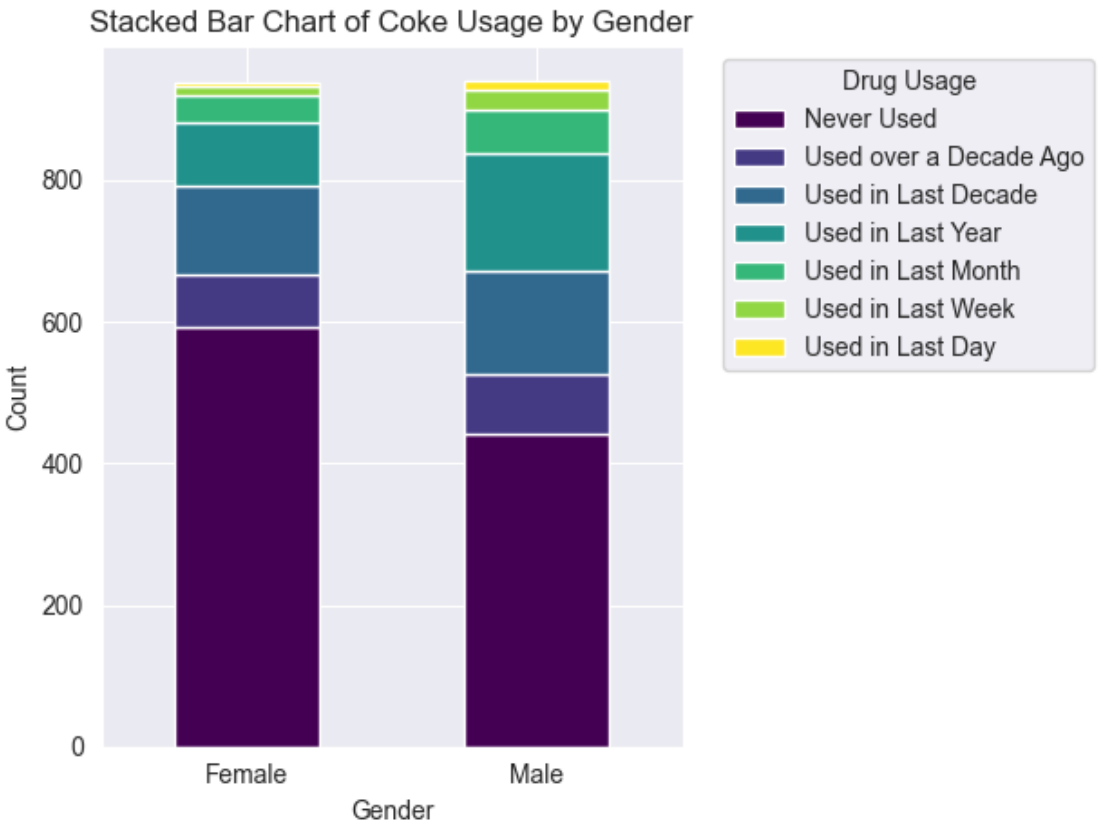

Stacked Bar Chart of Amphet Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

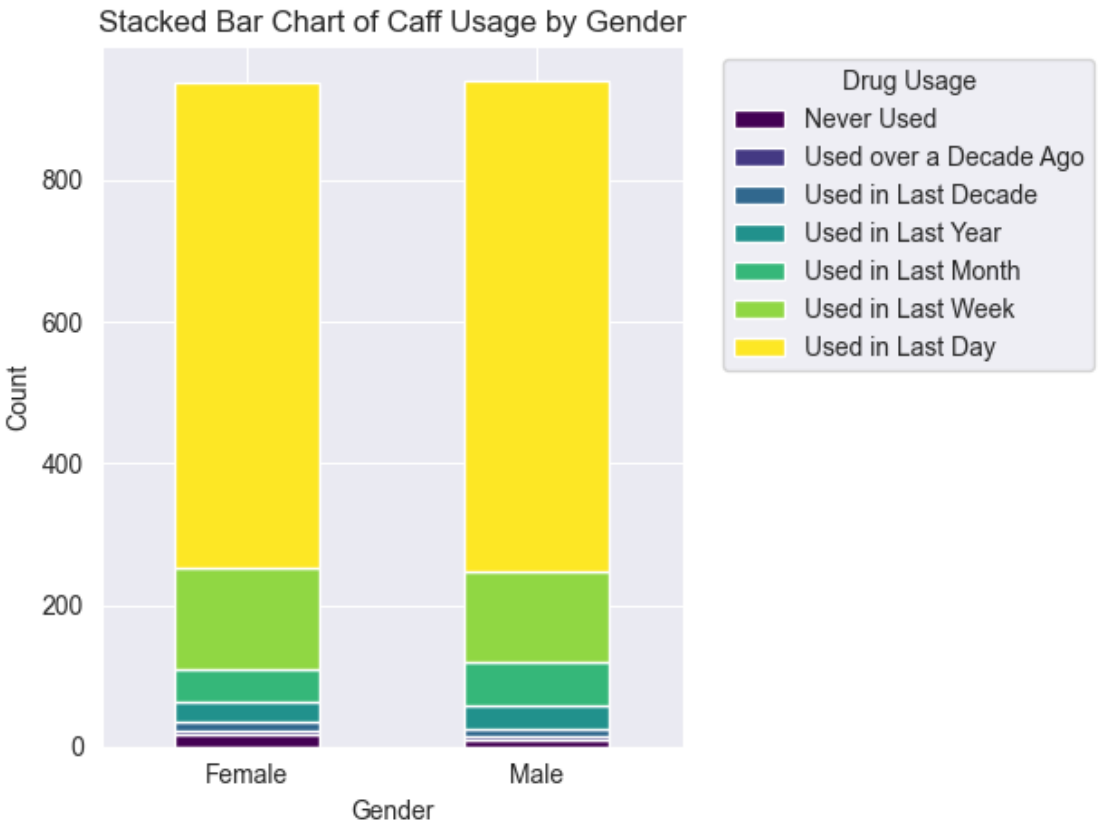

Stacked Bar Chart of Amyl Usage by Gender
Count
800
600
400
200
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

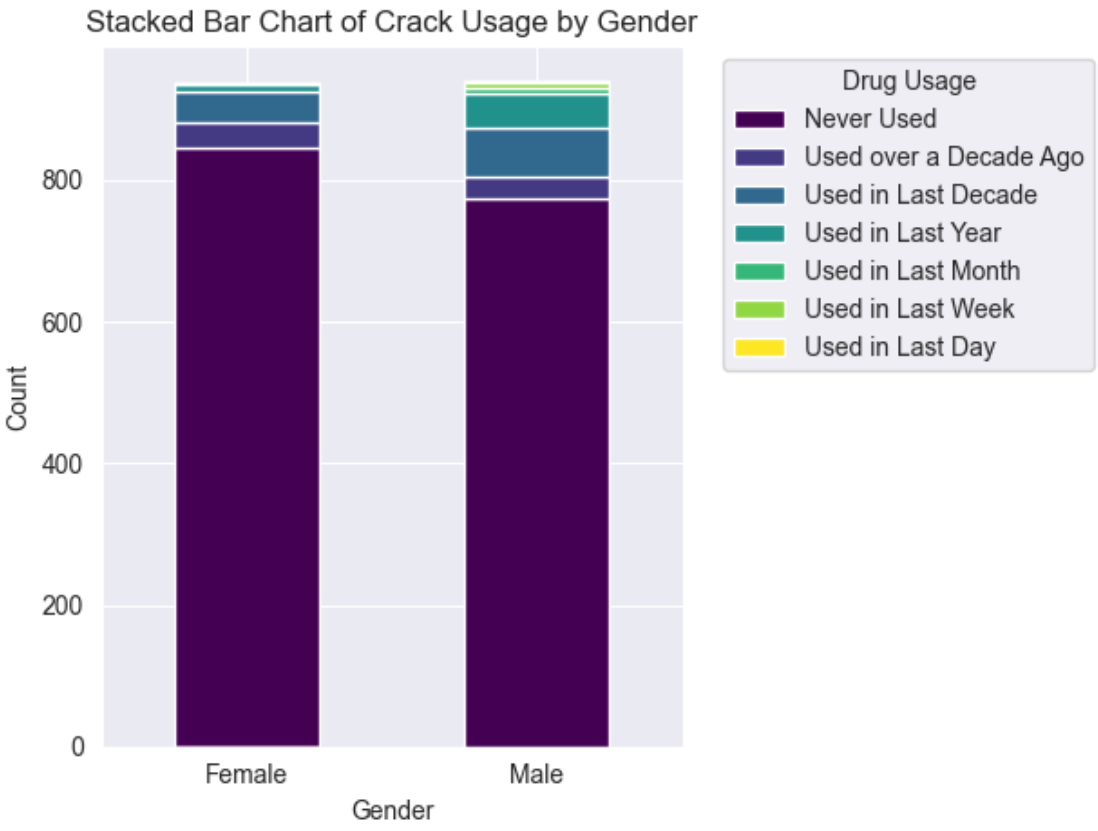

Stacked Bar Chart of Benzos Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

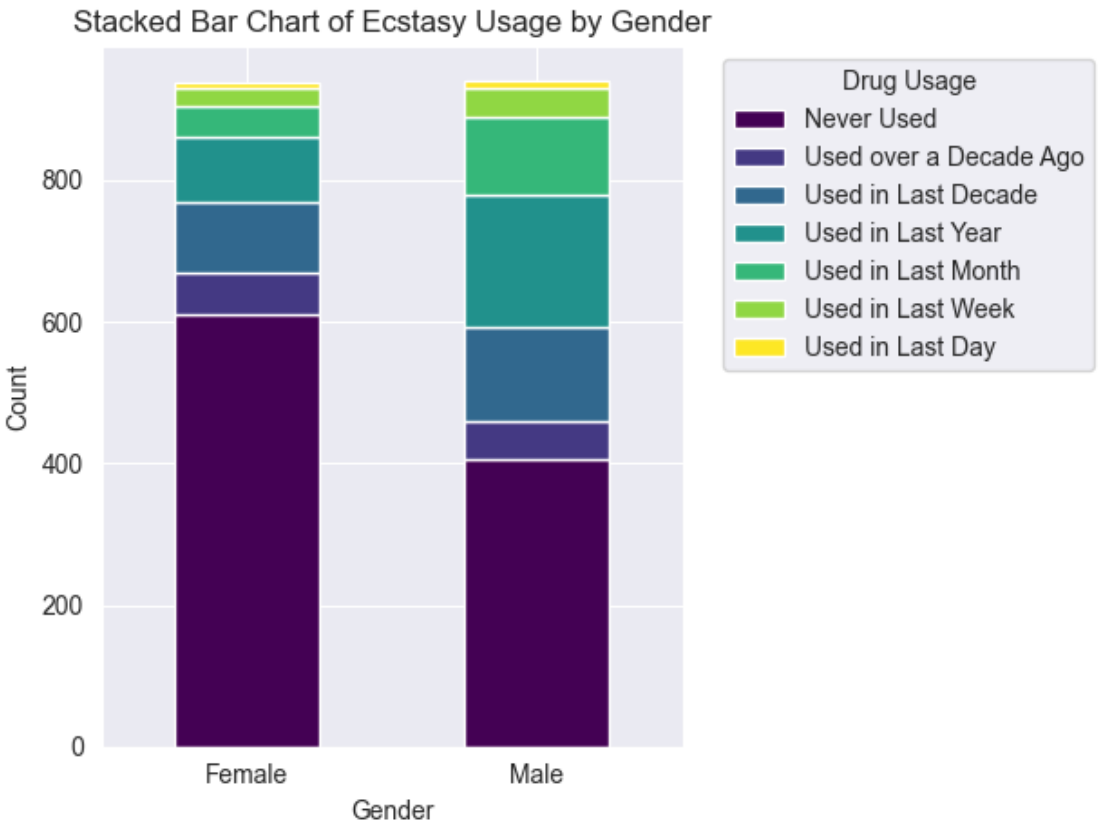

Stacked Bar Chart of Cannabis Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

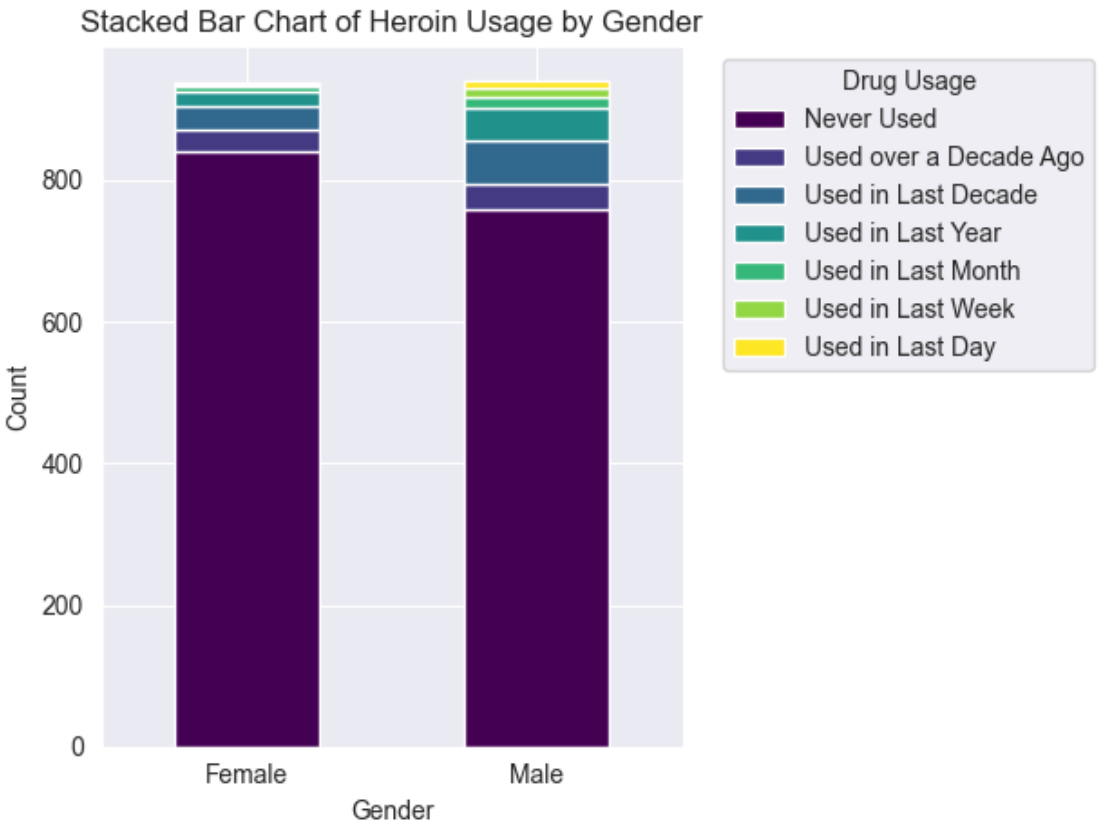

Stacked Bar Chart of Choc Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

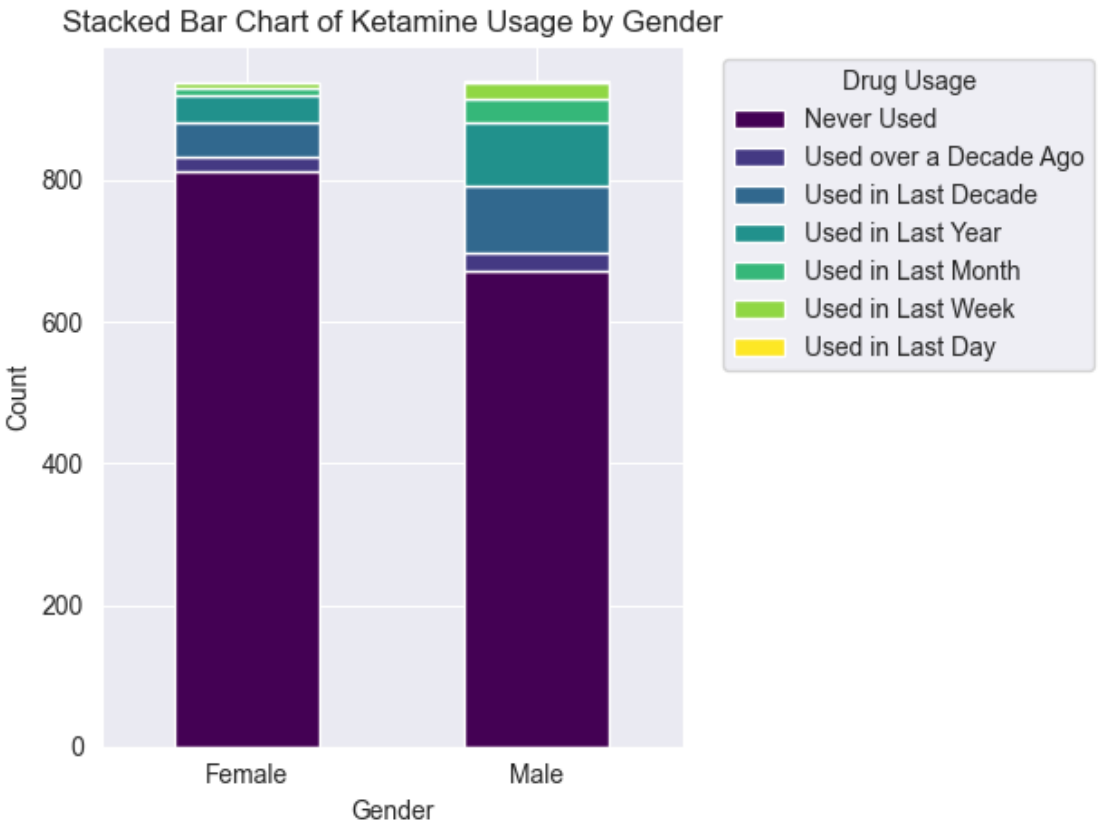

Stacked Bar Chart of Coke Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

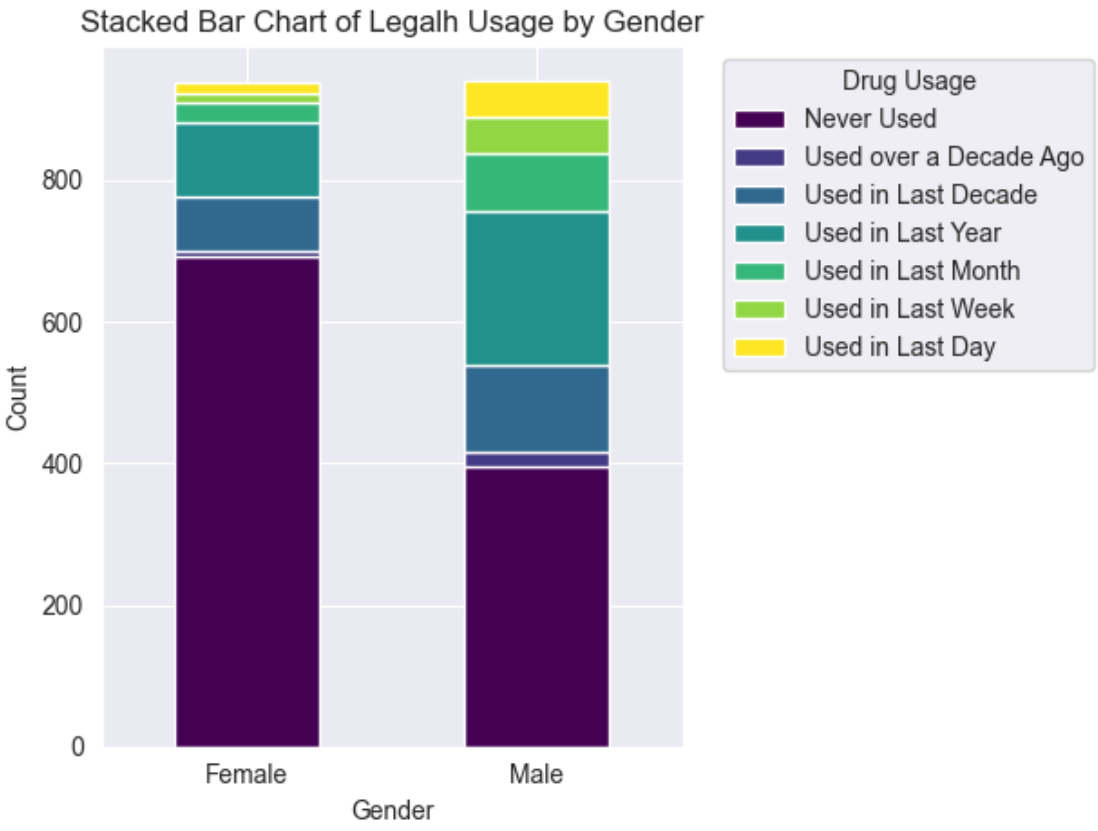

Stacked Bar Chart of Caff Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

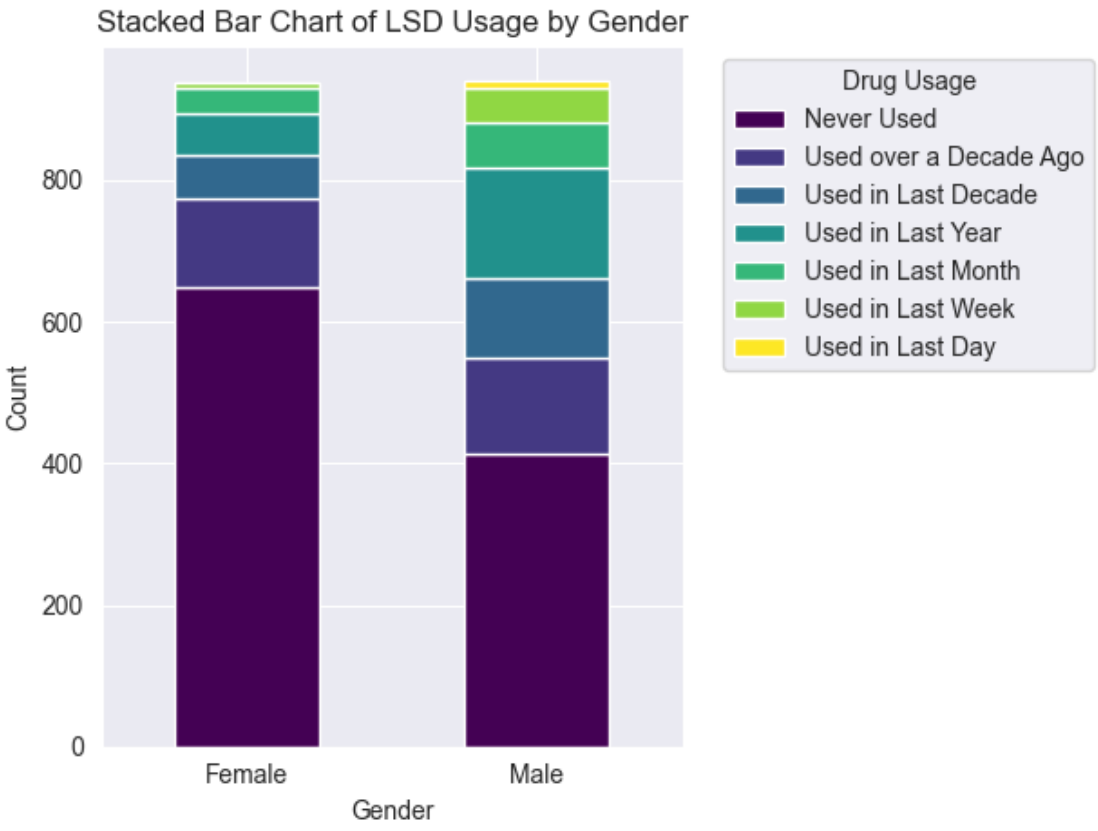

Stacked Bar Chart of Crack Usage by Gender
Count
0
200
400
600
800
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

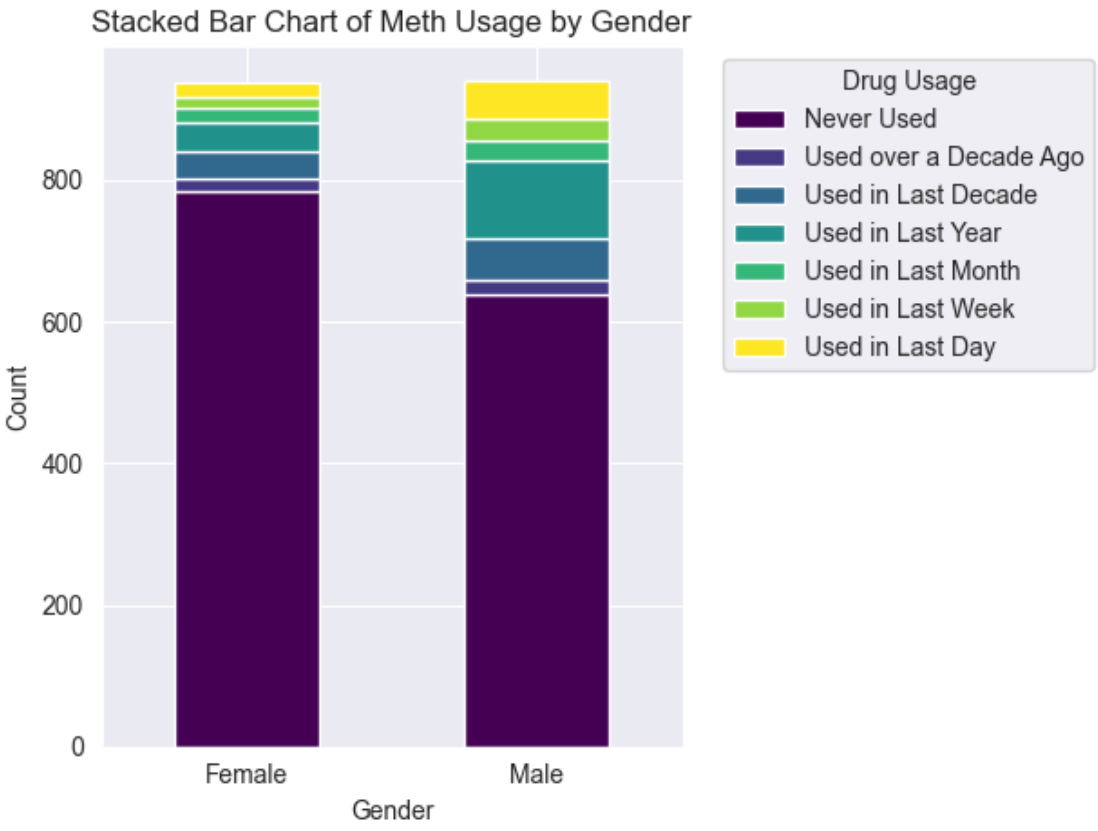

Stacked Bar Chart of Ecstasy Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

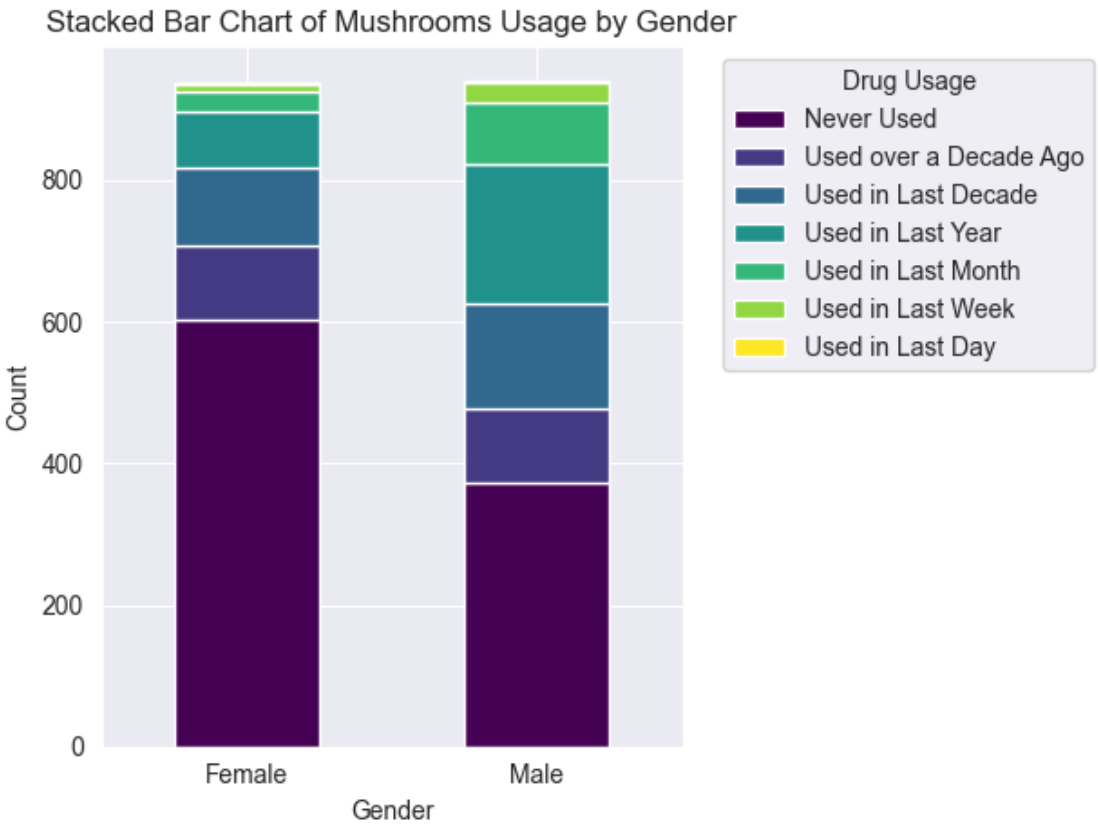

Stacked Bar Chart of Heroin Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

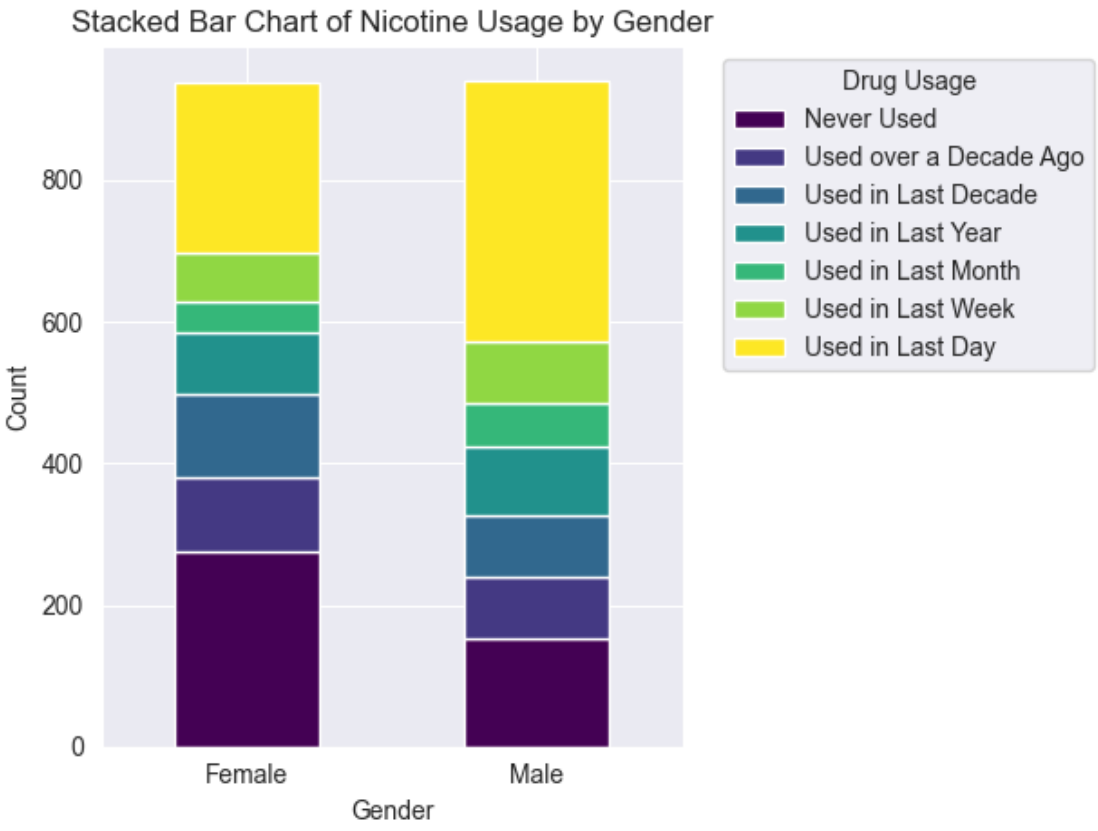

Stacked Bar Chart of Ketamine Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

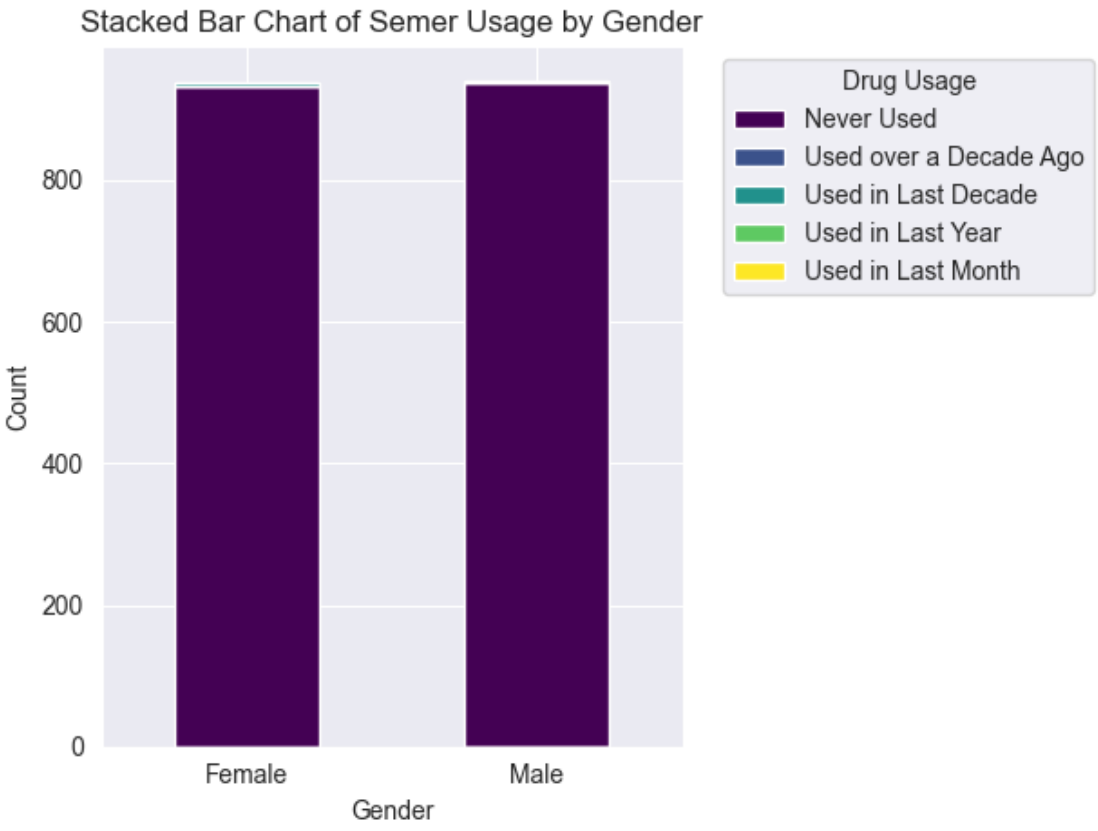

Stacked Bar Chart of Legalh Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

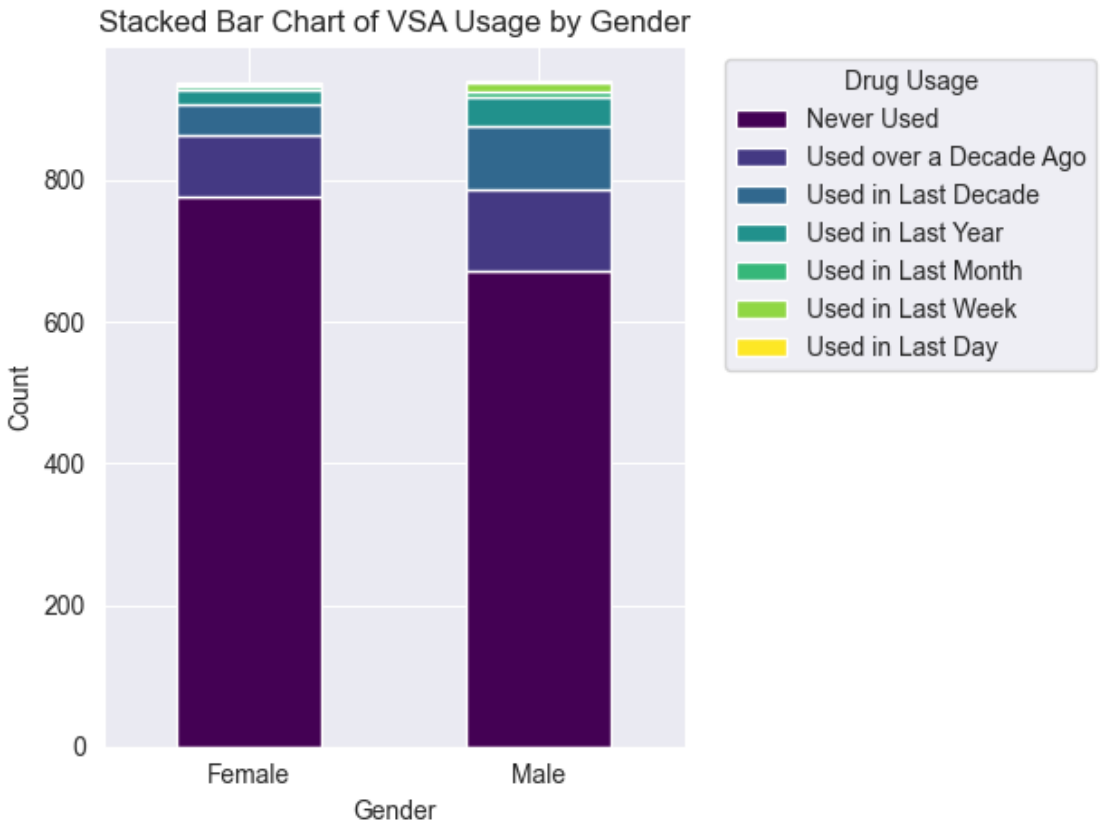

Stacked Bar Chart of LSD Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender


Stacked Bar Chart of Meth Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender


Stacked Bar Chart of Mushrooms Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

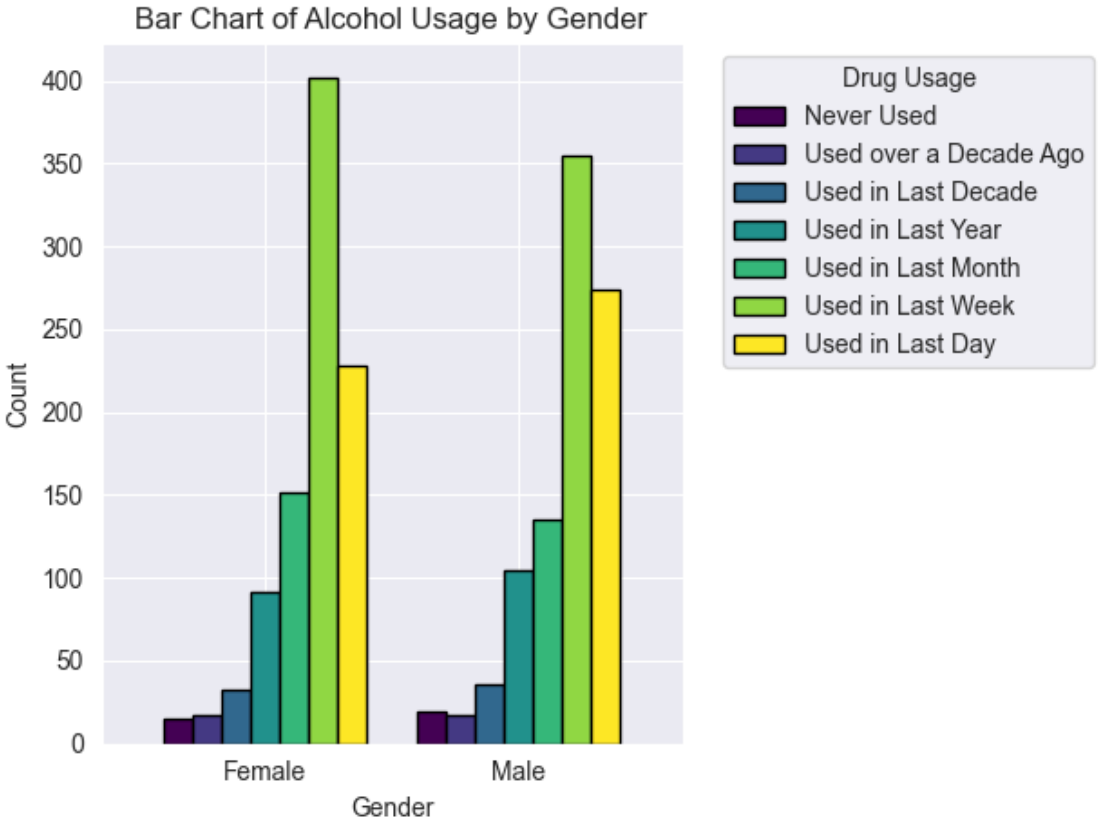

Stacked Bar Chart of Nicotine Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
200
400
600
800
Female
Male
Gender

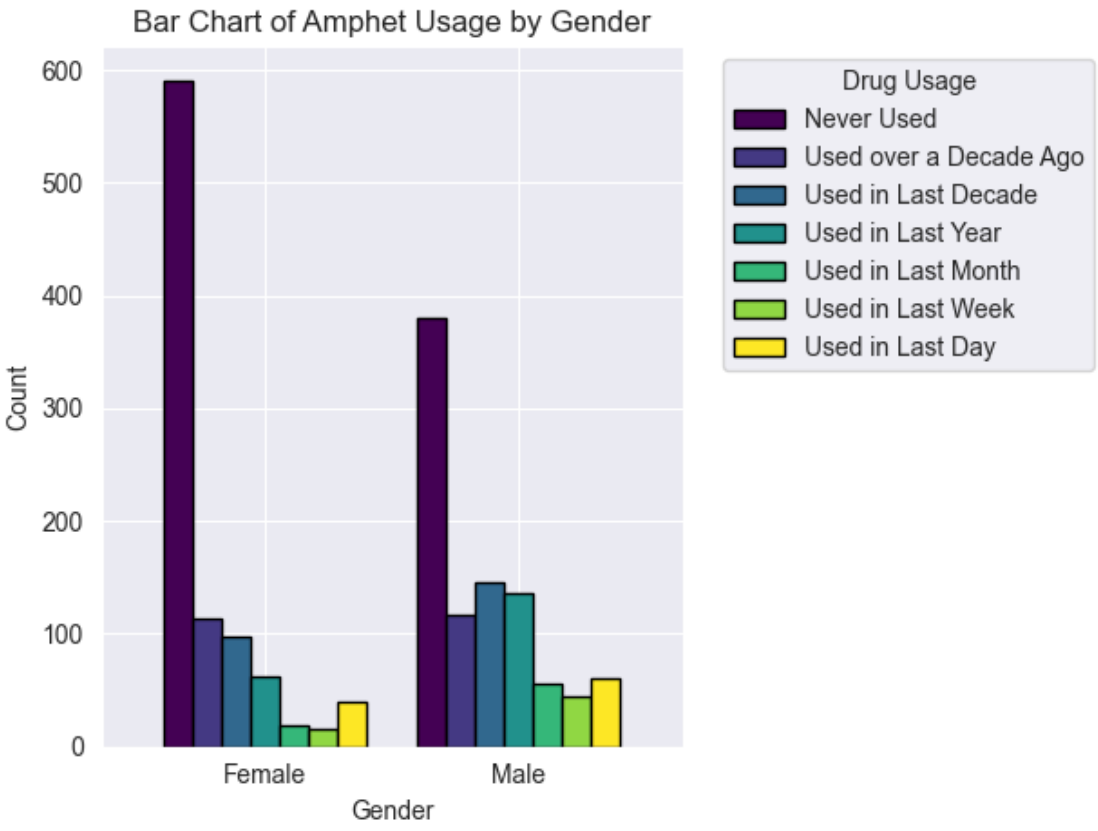

Stacked Bar Chart of Semer Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Count
0
200
400
600
800
Female
Male
Gender

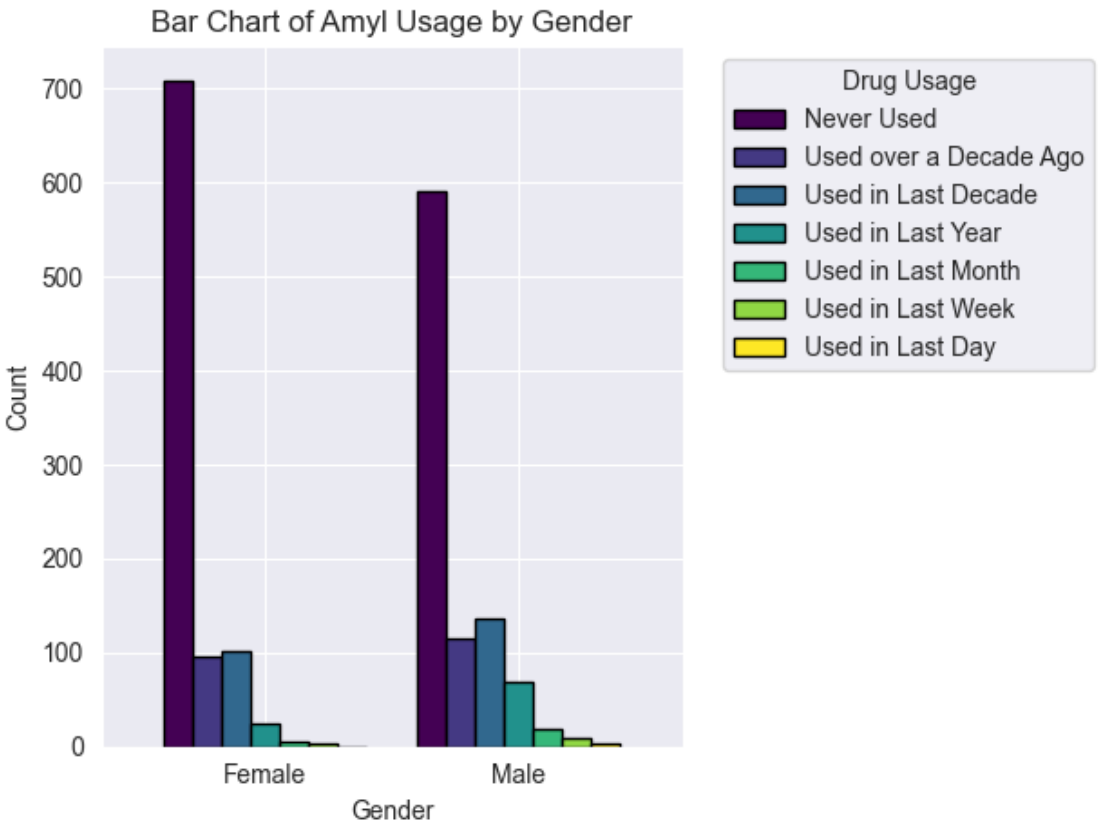


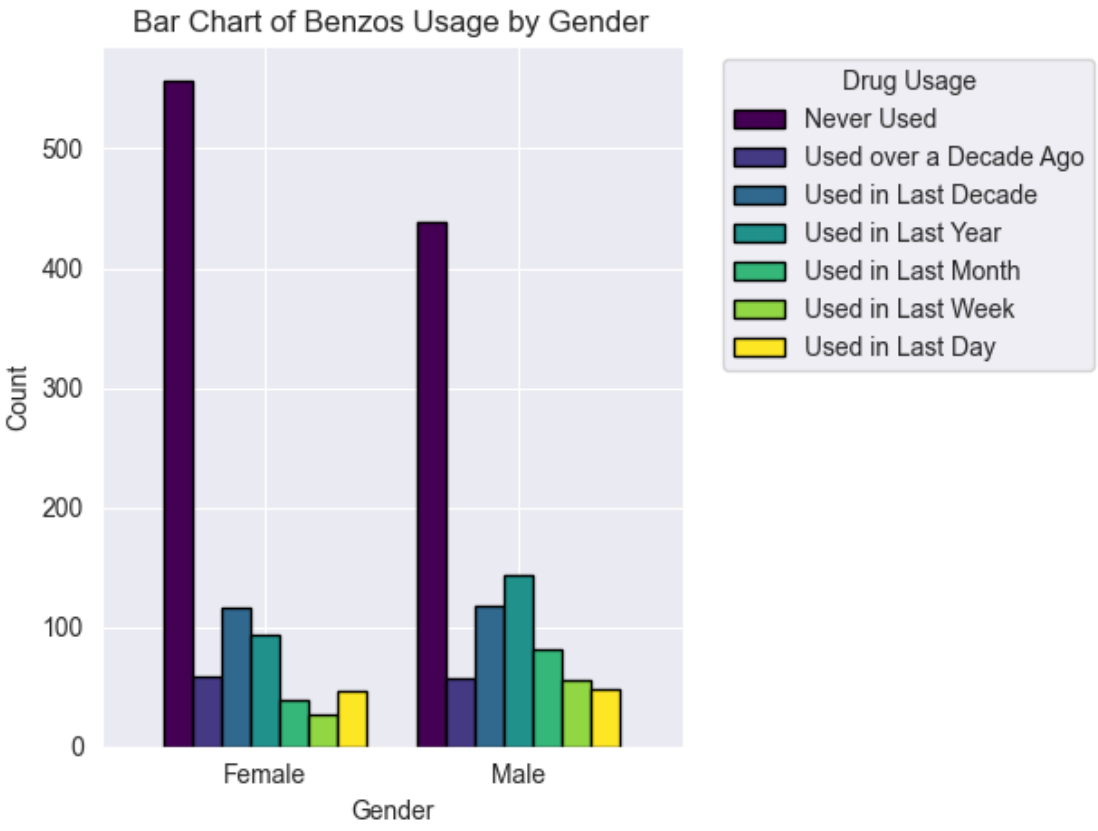
[79]:
```
for drug in drug_columns:
    # Prepare data for plotting
    plot_data = df.groupby(['gender_values', drug], observed=True).size().
 ↪unstack(fill_value=0)

    # Convert the columns to descriptive labels for the plot only
    plot_data = plot_data.rename(columns=CATEGORIES)

    # Plotting as grouped bars instead of stacked
    ax = plot_data.plot(kind='bar', width=0.8, colormap='viridis', position=0.
 ↪5, edgecolor='black')
    plt.title(f'Bar Chart of {drug.replace("_values", "")} Usage by Gender')
    plt.xlabel('Gender')
    plt.ylabel('Count')
    plt.xticks(rotation=0)  # Keep the gender labels horizontal for readability

    # Set legend directly using the category order
    plt.legend(category_order, title='Drug Usage', bbox_to_anchor=(1.05, 1),␣
 ↪loc='upper left')
```

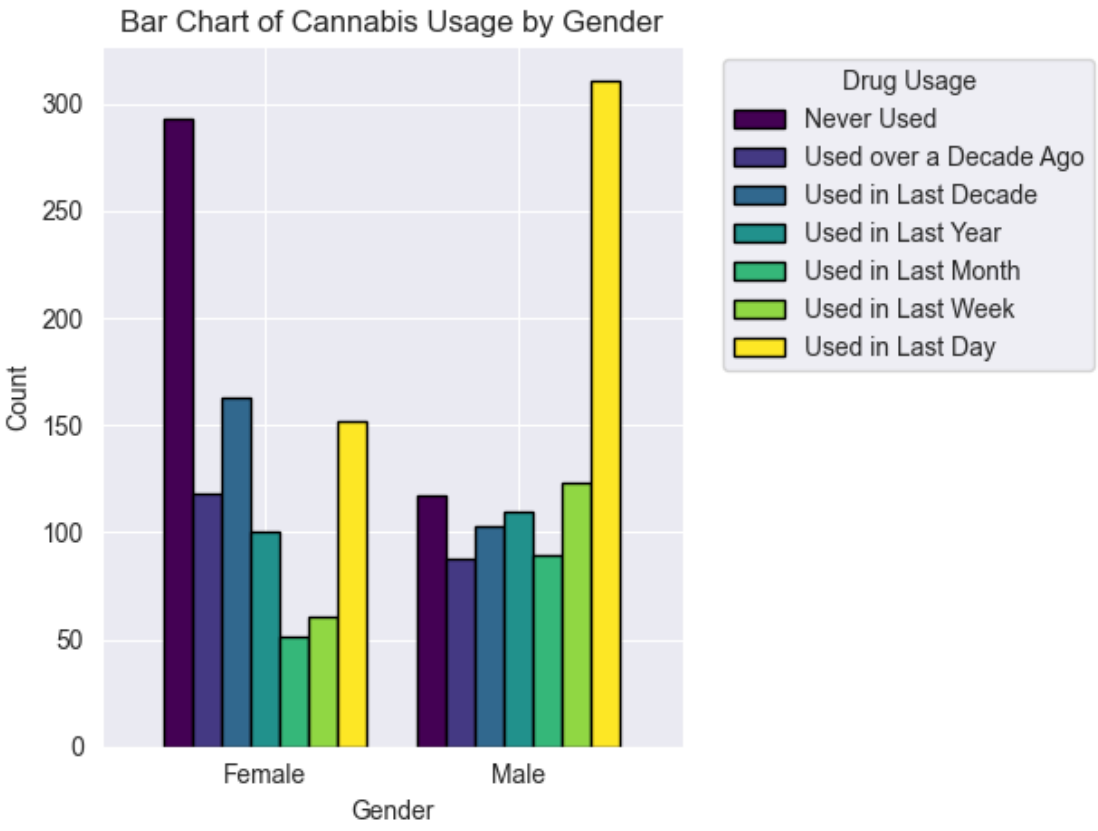
```
    plt.tight_layout()
    plt.show()
```

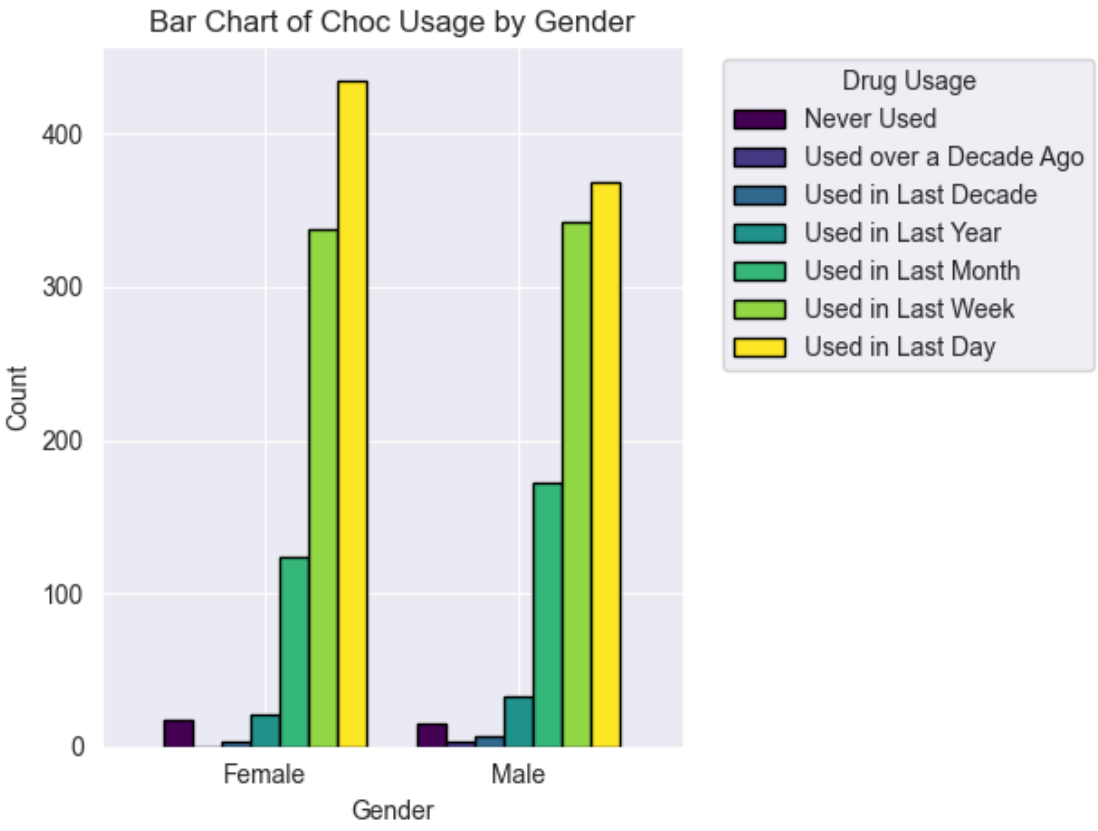


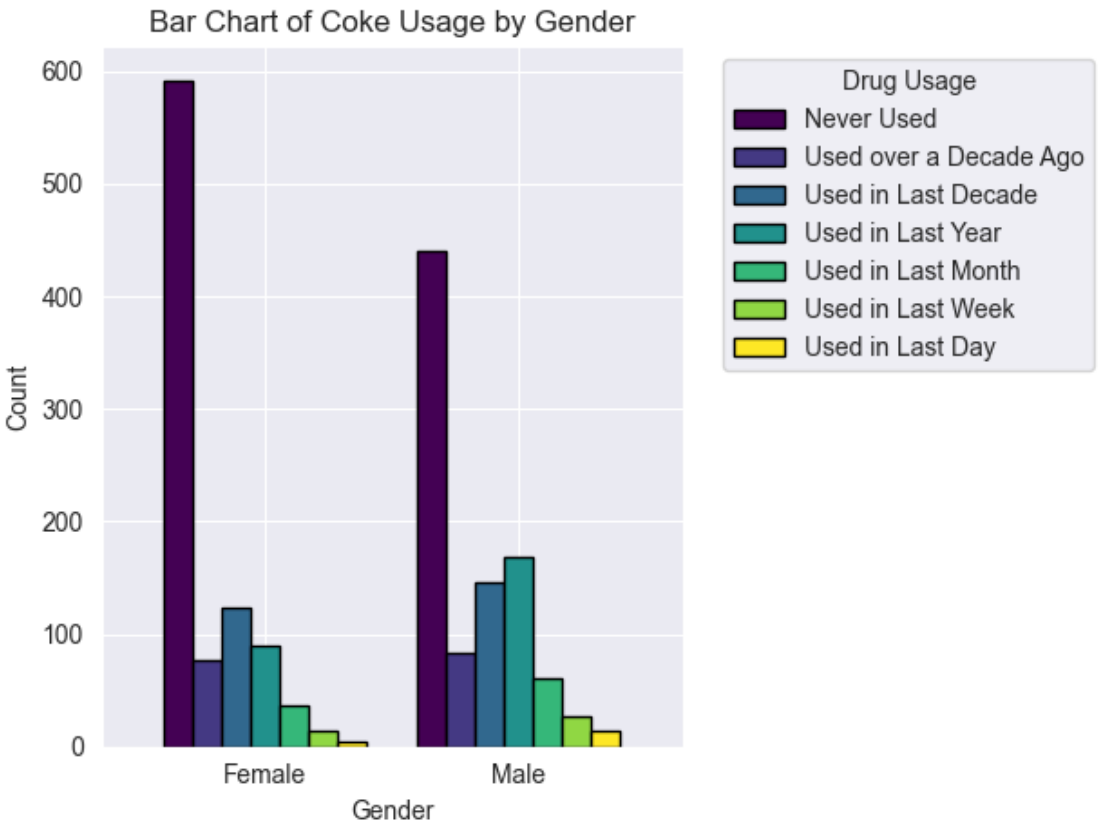

Bar Chart of Amphet Usage by Gender
Count
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

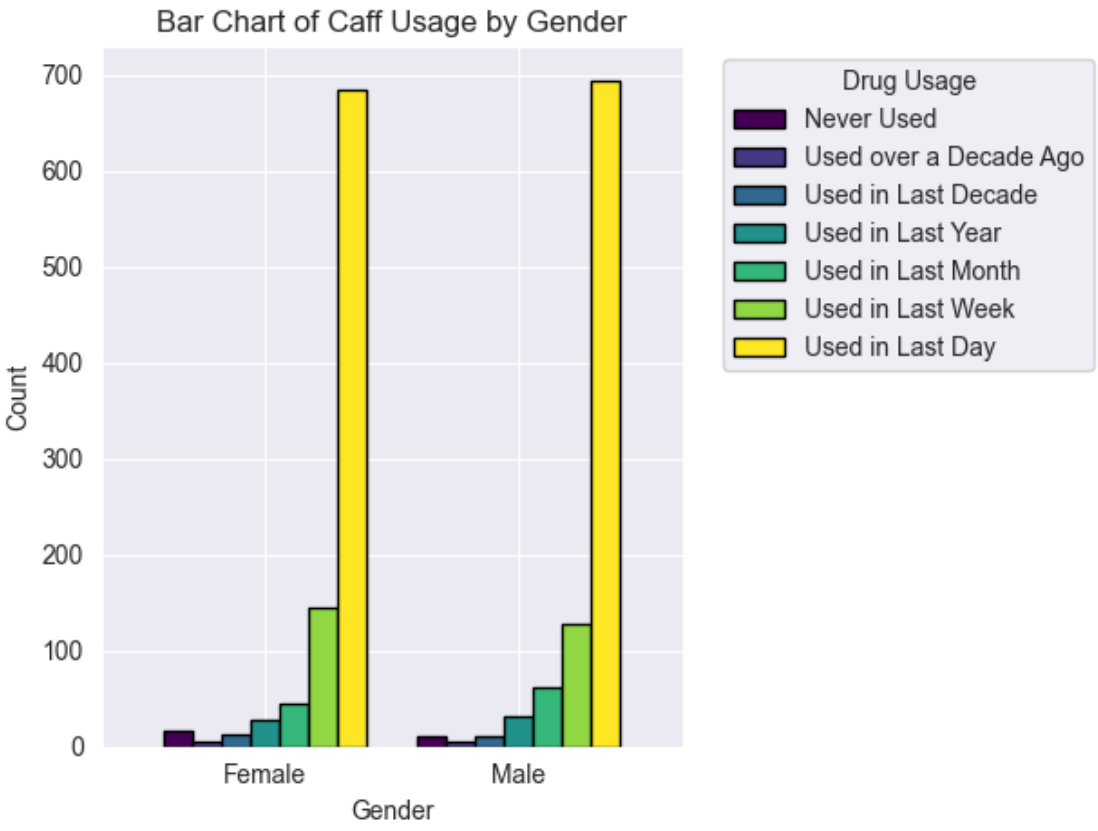

Bar Chart of Amyl Usage by Gender
Count
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

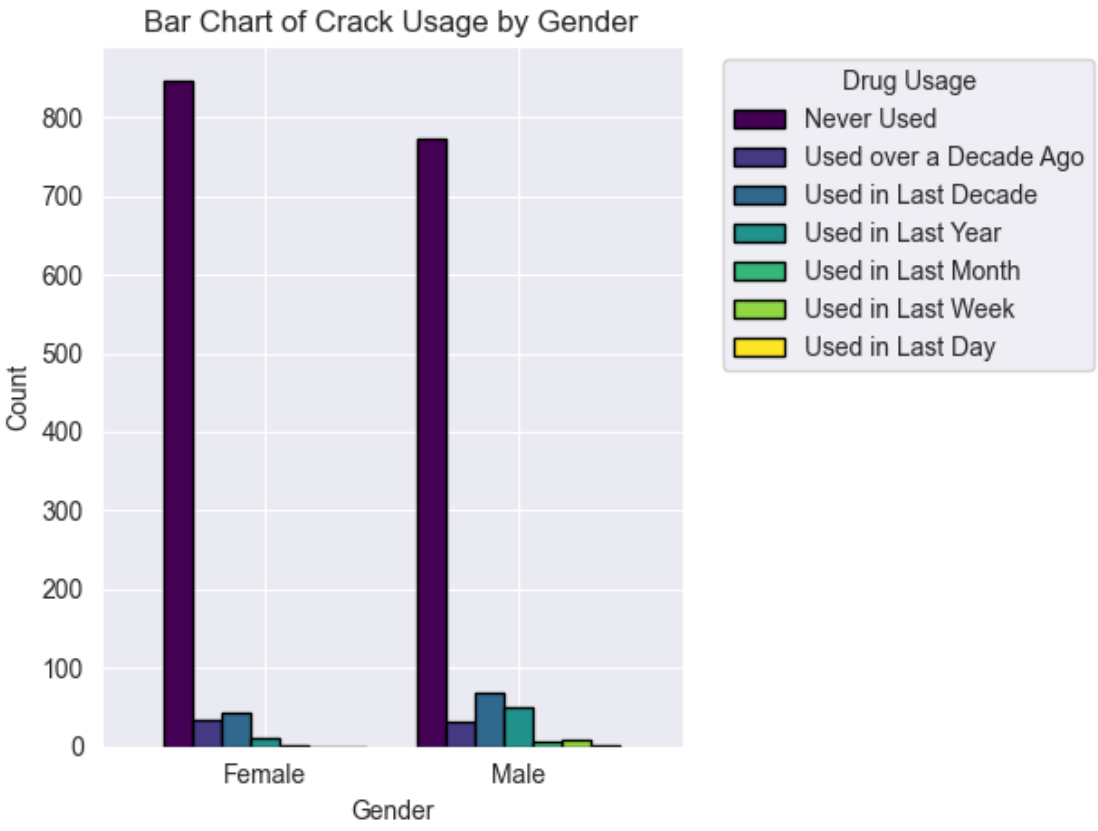

Bar Chart of Benzos Usage by Gender
Count
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

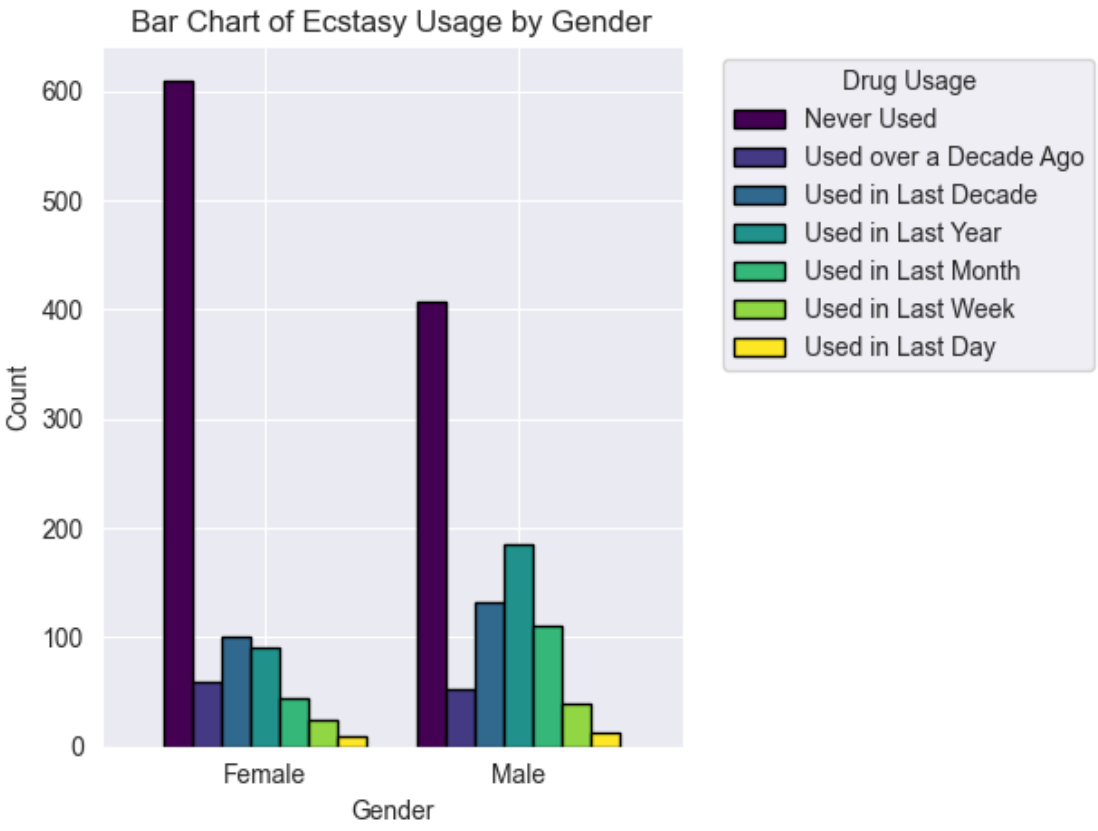

Bar Chart of Cannabis Usage by Gender
Count
300
250
200
150
100
50
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

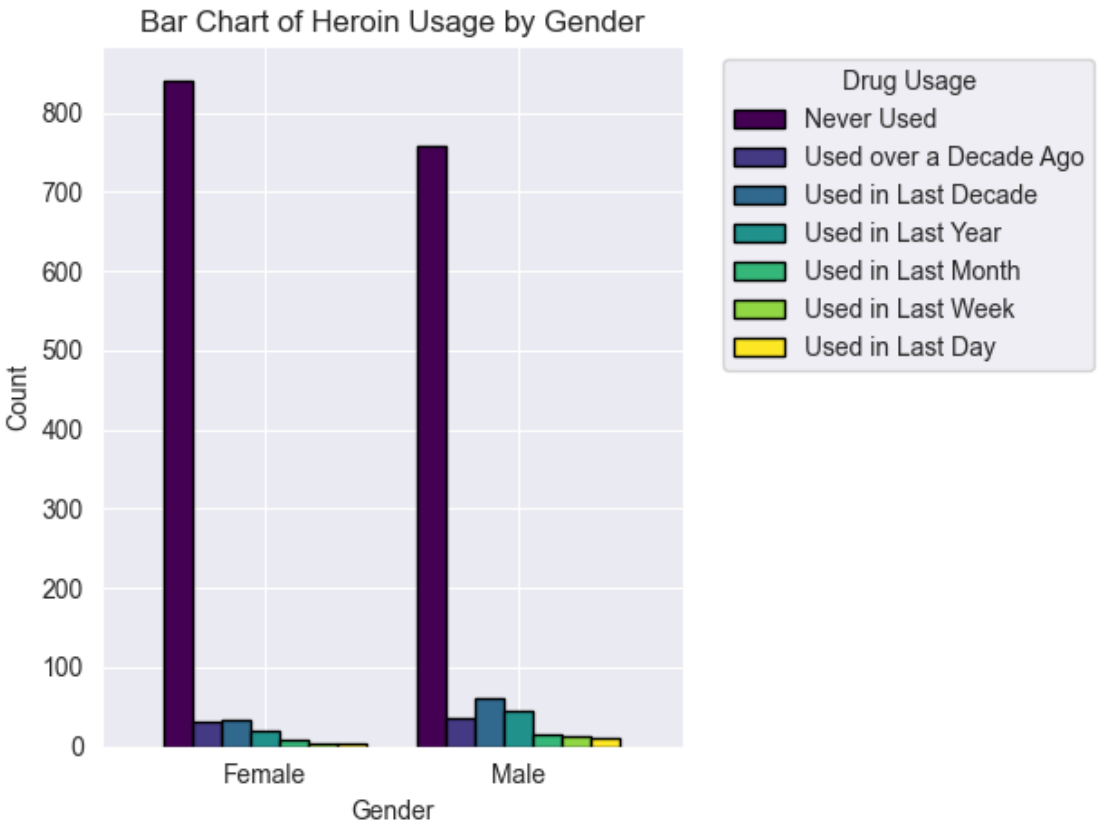

Bar Chart of Choc Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
100
200
300
400
Female
Male
Gender

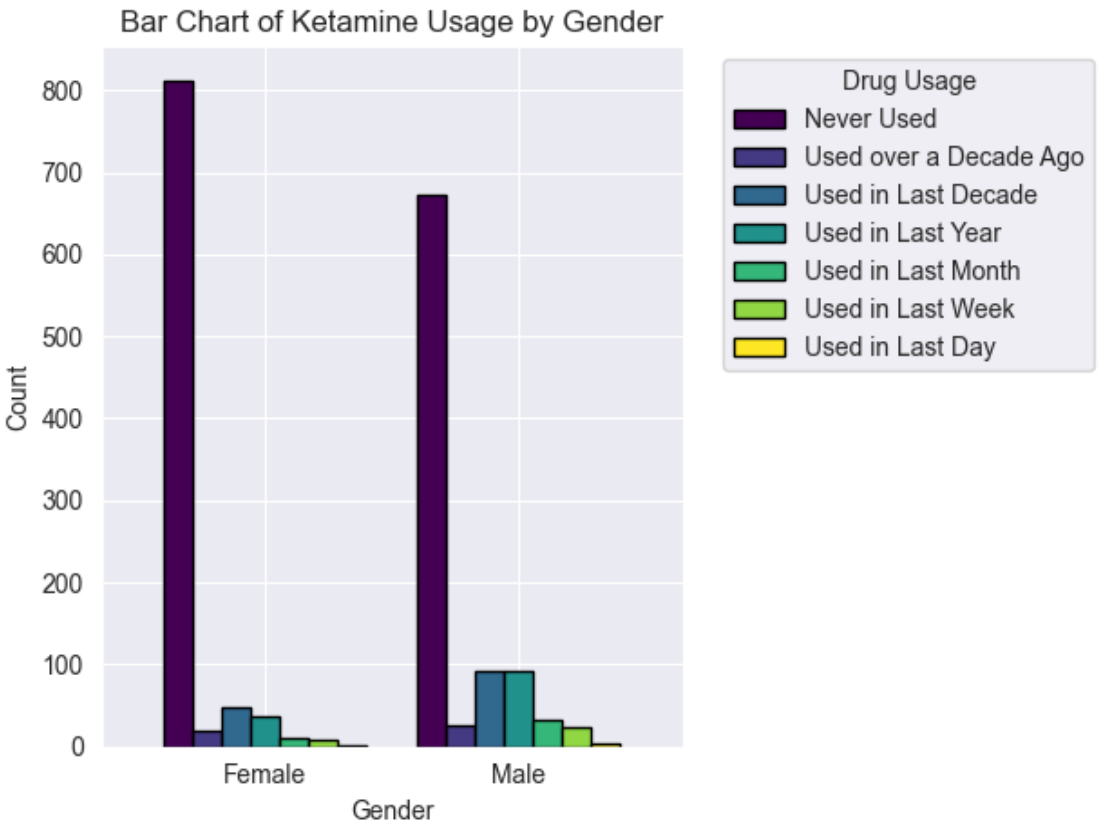

Bar Chart of Coke Usage by Gender
Count
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

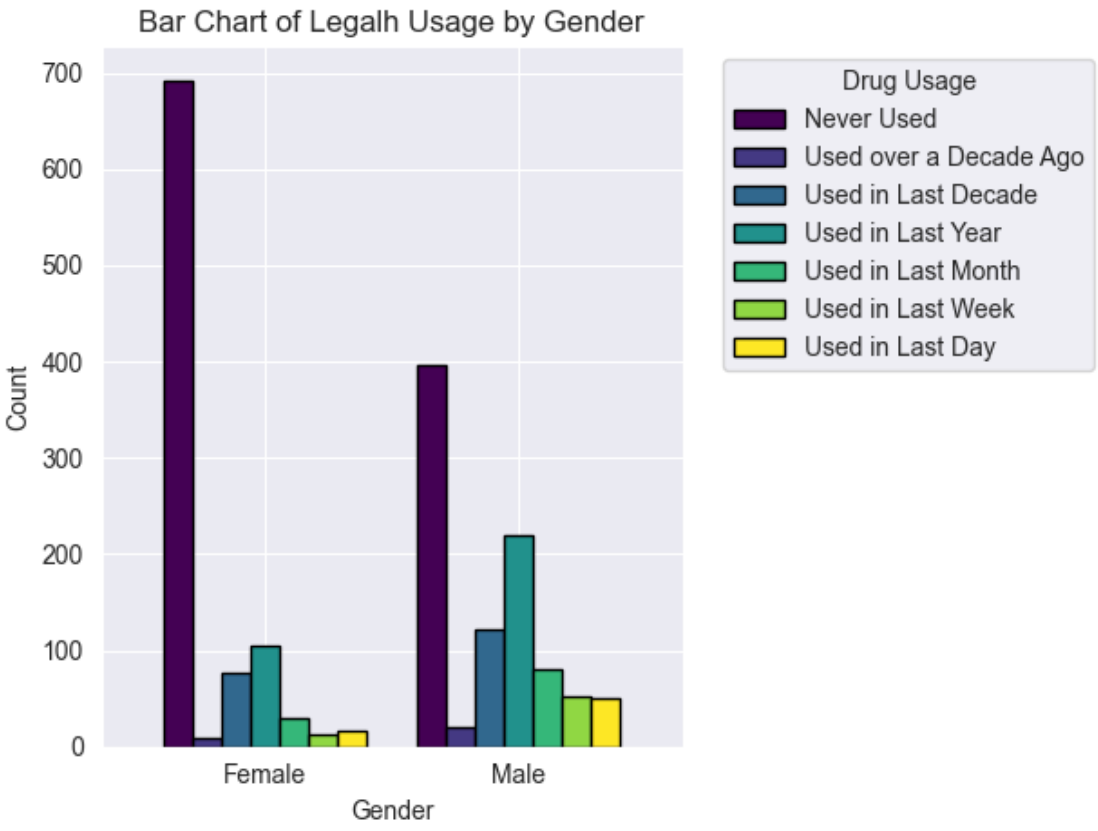

Bar Chart of Caff Usage by Gender
Count
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

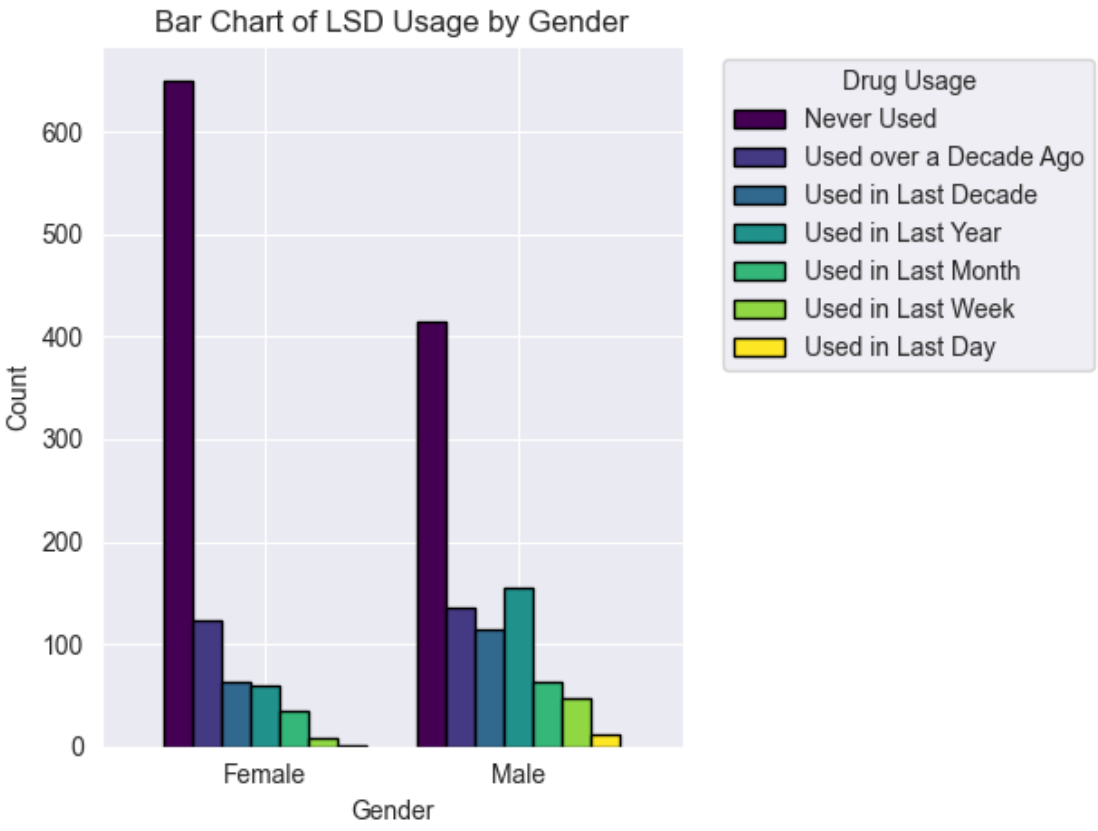

Bar Chart of Crack Usage by Gender
Count
800
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

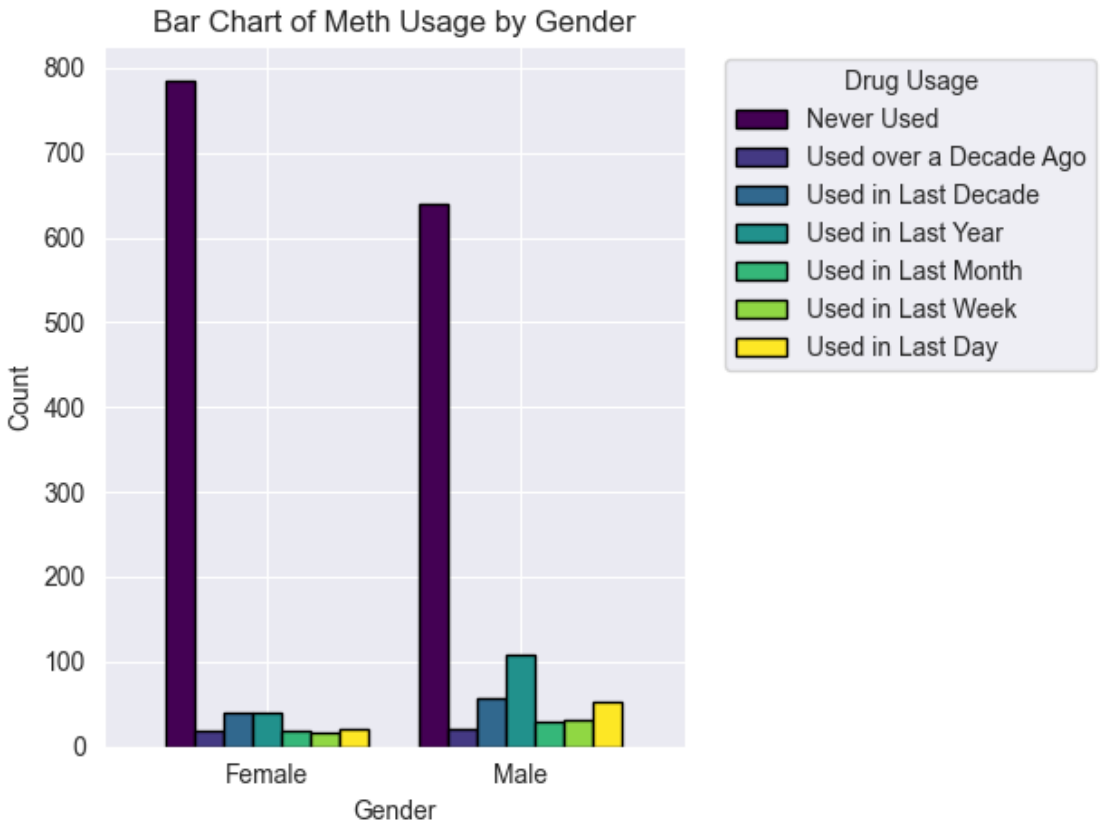

Bar Chart of Ecstasy Usage by Gender
Count
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

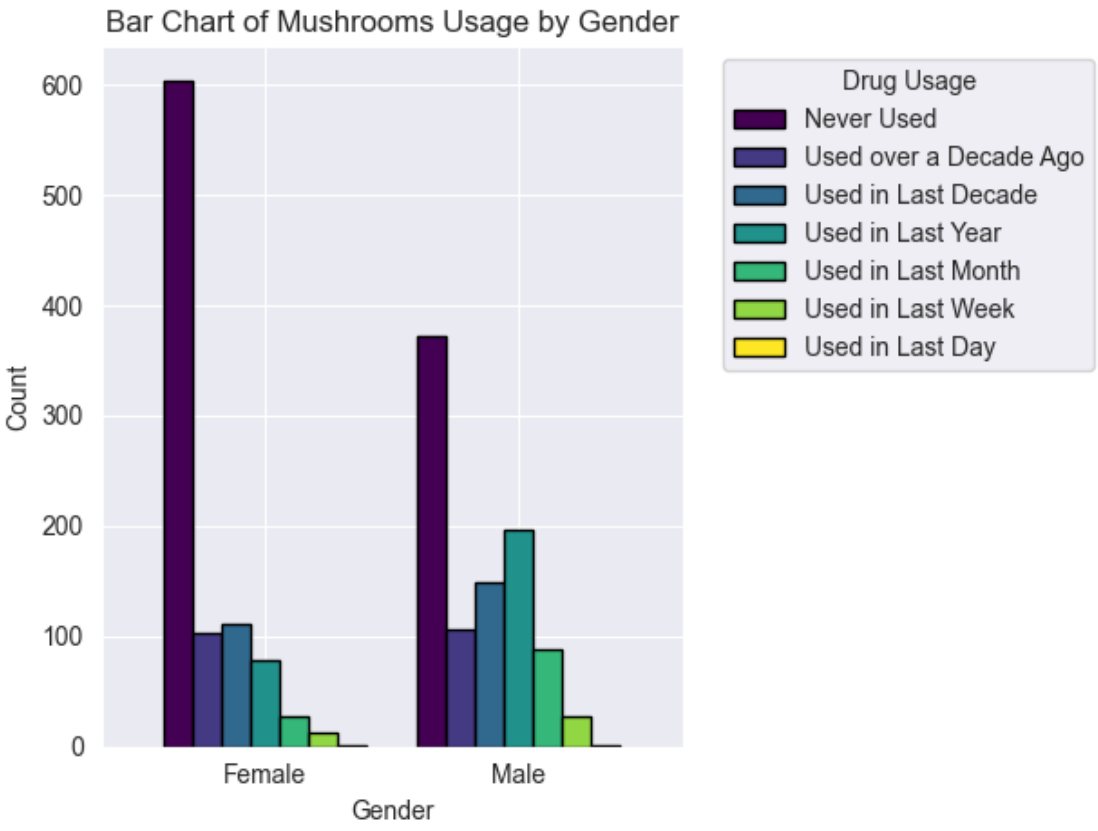

Bar Chart of Heroin Usage by Gender
Count
800
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

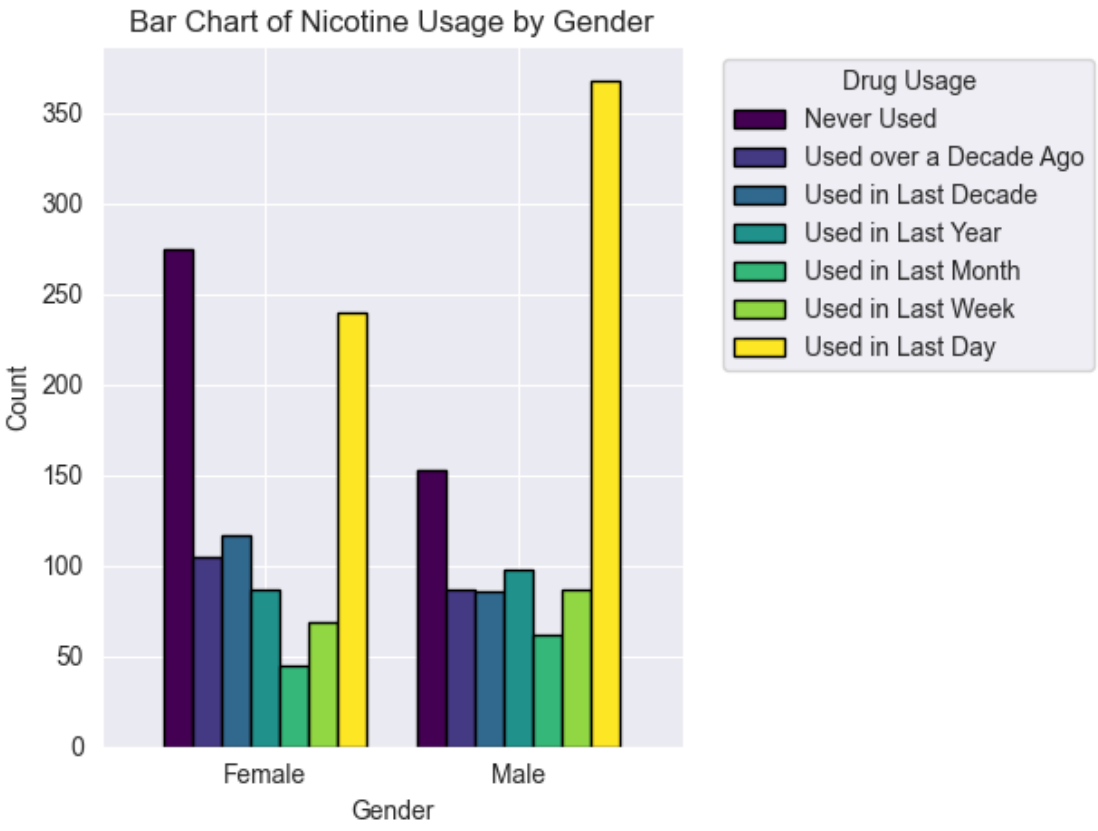

Bar Chart of Ketamine Usage by Gender
Count
800
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

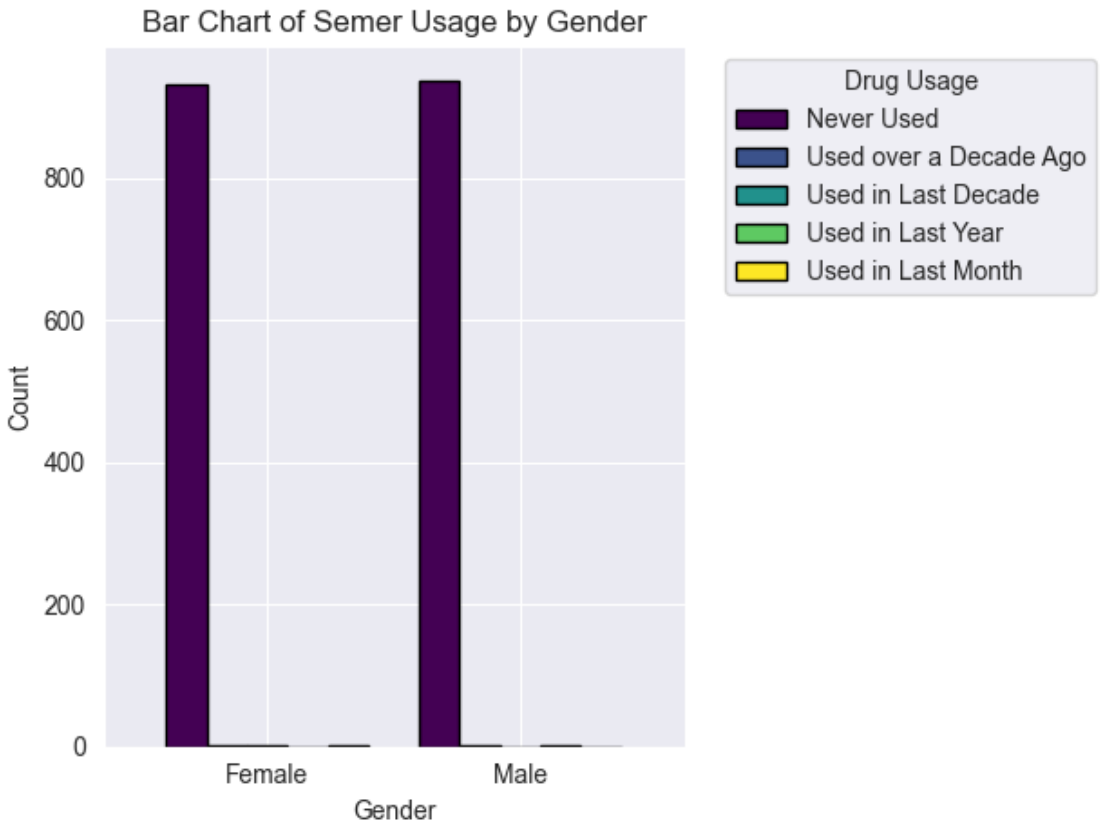

Bar Chart of Legalh Usage by Gender
Count
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day

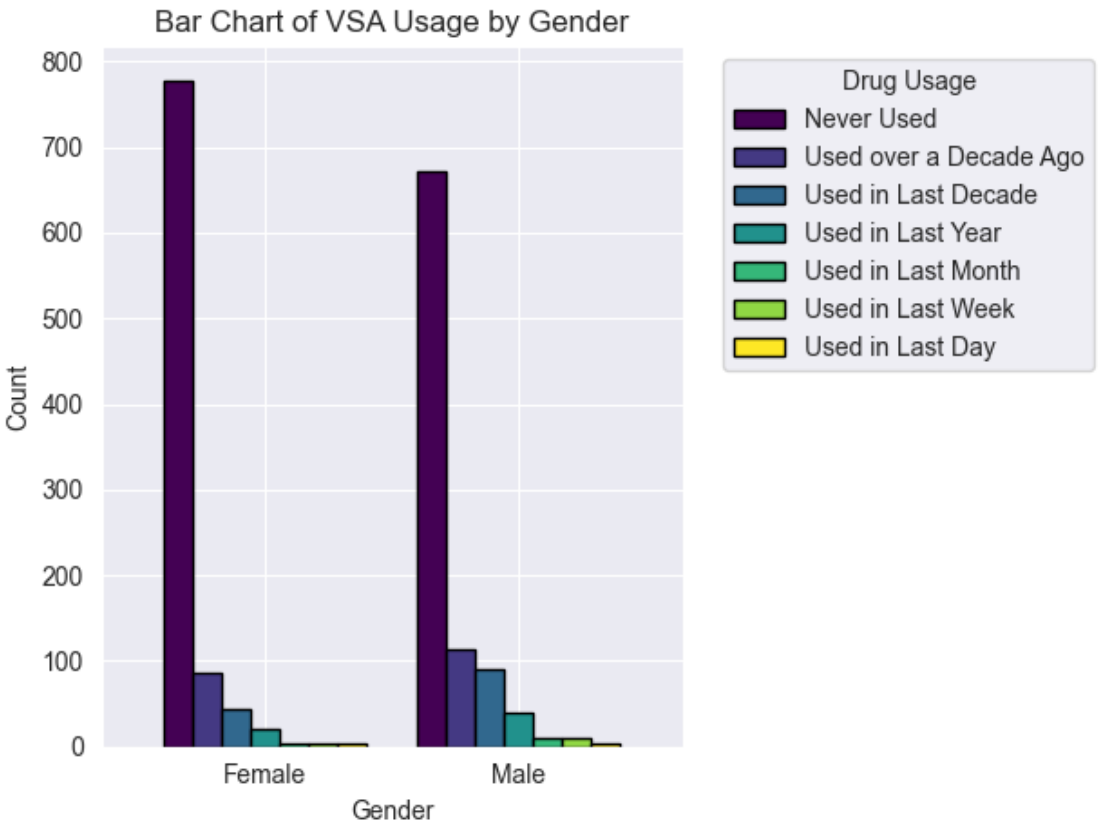

Bar Chart of LSD Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day
Count
0
100
200
300
400
500
600
Female
Male
Gender


Bar Chart of Meth Usage by Gender
Count
800
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day


Bar Chart of Mushrooms Usage by Gender
Count
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day


Bar Chart of Nicotine Usage by Gender
Count
0
50
100
150
200
250
300
350
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day


Bar Chart of Semer Usage by Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Count
800
600
400
200
0
Female
Male
Gender


Bar Chart of VSA Usage by Gender
Count
800
700
600
500
400
300
200
100
0
Female
Male
Gender
Drug Usage
Never Used
Used over a Decade Ago
Used in Last Decade
Used in Last Year
Used in Last Month
Used in Last Week
Used in Last Day