row 1: question row 2: column name row 3-96: 94 responses

In [80]:
```
# import necessary libraries

import pandas as pd
import numpy as np
from sklearn import preprocessing
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

In [81]:
```
# import survey dataset as dataframe with encoding set to cp1252 (Windows-1252/Western European Encoding)
# and header set to 1 because we do not need to the question row (all question rows already converted as column name)

survey_df = pd.read_csv("SMT202SurveyDataset.csv",encoding="cp1252",header=1)
```

In [82]:
```
survey_df
```

Out[82]:

	Gender	University	Major	Group	VapingLocation	ReasonVapingLocation	VapeSpendingAmount	PreferenceToBeAlone	ReasonPre
0	Female	NUS	Arts/Design	Vaper	University,Dorm,Home,Friends' house,Public spaces	Dorm, to have privacy	10.0	No	wi
1	Male	NUS	STEM	Vaper	University,Dorm	Dorm, more privacy	8.0	Yes	don't ne
2	Female	NUS	Computing	Non-vaper	NaN	NaN	NaN	NaN	
3	Female	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
4	Male	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	
89	Female	NUS	Medicine	Non-vaper	NaN	NaN	NaN	NaN	
90	Male	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
91	Female	NUS	Arts/Design	Non-vaper	NaN	NaN	NaN	NaN	
92	Female	NUS	Law/Social Science	Non-vaper	NaN	NaN	NaN	NaN	
93	Male	NTU	Business	Non-vaper	NaN	NaN	NaN	NaN	

94 rows × 34 columns

In [83]:
```
# see how many non-null values and what data types in this survey dataset

survey_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 94 entries, 0 to 93
Data columns (total 34 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   Gender                          94 non-null     object 
 1   University                      94 non-null     object 
 2   Major                           94 non-null     object 
 3   Group                           94 non-null     object 
 4   VapingLocation                  36 non-null     object 
 5   ReasonVapingLocation            5 non-null      object 
 6   VapeSpendingAmount              36 non-null     float64
 7   PreferenceToBeAlone             13 non-null     object 
 8   ReasonPreferenceToBeAlone       13 non-null     object 
 9   PreferredVapingPeriod           8 non-null      object 
 10  Motivation                      36 non-null     object 
 11  ReasonStress                    17 non-null     object 
 12  EncourageVaping                 36 non-null     object 
 13  ReasonEncourageVaping           4 non-null      object 
 14  Exposure                        94 non-null     object 
 15  PerceptionVapingBeforeU         94 non-null     int64  
 16  PerceptionVapingAfterU          94 non-null     int64  
 17  Dorm                            94 non-null     object 
 18  HallType_NUS                    9 non-null      object 
 19  HallType_NTU                    21 non-null     object 
 20  RoomType                        30 non-null     object 
 21  DaysInDorm                      30 non-null     float64
 22  HoursWithHallmates              30 non-null     float64
 23  DormEncouragesVaping            30 non-null     float64
 24  ReasonDormEncouragesVaping      30 non-null     object 
 25  SeeVapersAtUni                  94 non-null     object 
 26  VapingLocationAtUni             29 non-null     object 
 27  HoursSpentStudying              94 non-null     int64  
 28  UniversityDiscouragingVaping    94 non-null     int64  
 29  UniversityRaiseAwarenessVaping  94 non-null     int64  
 30  VaperFriends                    94 non-null     int64  
 31  VaperFriendsDormStayer          94 non-null     int64  
 32  EncouragedByFriendsToVape       31 non-null     object 
 33  CompelledToVape                 24 non-null     object 
dtypes: float64(4), int64(7), object(23)
memory usage: 25.1+ KB
```

In [84]:
```
survey_df.shape # 94 rows, 34 columns
```

Out[84]:
```
(94, 34)
```

In [85]:
```
survey_df.columns # find out what columns we have in the dataset
```

Out[85]:
```
Index(['Gender', 'University', 'Major', 'Group', 'VapingLocation',
       'ReasonVapingLocation', 'VapeSpendingAmount', 'PreferenceToBeAlone',
       'ReasonPreferenceToBeAlone', 'PreferredVapingPeriod', 'Motivation',
       'ReasonStress', 'EncourageVaping', 'ReasonEncourageVaping', 'Exposure',
       'PerceptionVapingBeforeU', 'PerceptionVapingAfterU', 'Dorm',
       'HallType_NUS', 'HallType_NTU', 'RoomType', 'DaysInDorm',
       'HoursWithHallmates', 'DormEncouragesVaping',
       'ReasonDormEncouragesVaping', 'SeeVapersAtUni', 'VapingLocationAtUni',
       'HoursSpentStudying', 'UniversityDiscouragingVaping',
       'UniversityRaiseAwarenessVaping', 'VaperFriends',
       'VaperFriendsDormStayer', 'EncouragedByFriendsToVape',
       'CompelledToVape'],
      dtype='object')
```

In [86]:
```
survey_df['DaysInDorm'].value_counts() # return counts of unique rows for Days in Dorm
```

Out[86]:
```
5.0    16
6.0     9
7.0     4
4.0     1
Name: DaysInDorm, dtype: int64
```

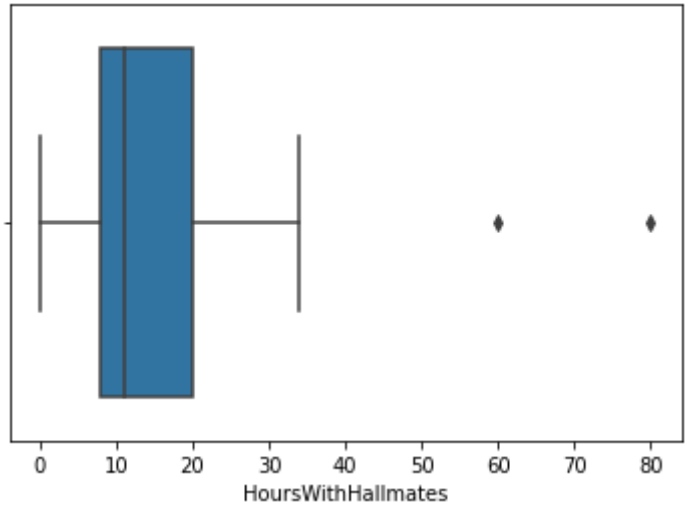
In [87]:
```
survey_df['HoursWithHallmates'].value_counts() # return counts of unique rows for Hours with Hallmates
```

Out[87]:
```
10.0    6
0.0     5
8.0     3
24.0    3
12.0    3
15.0    2
20.0    2
30.0    1
34.0    1
80.0    1
60.0    1
5.0     1
14.0    1
Name: HoursWithHallmates, dtype: int64
```

In [88]:
```
# Draw a box plot to show distributions with respect to Hours with Hallmates
# We use this as reference to create a group range

sns.boxplot(survey_df['HoursWithHallmates'])
```

```
c:\Users\Crissie\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[88]: <AxesSubplot:xlabel='HoursWithHallmates'>

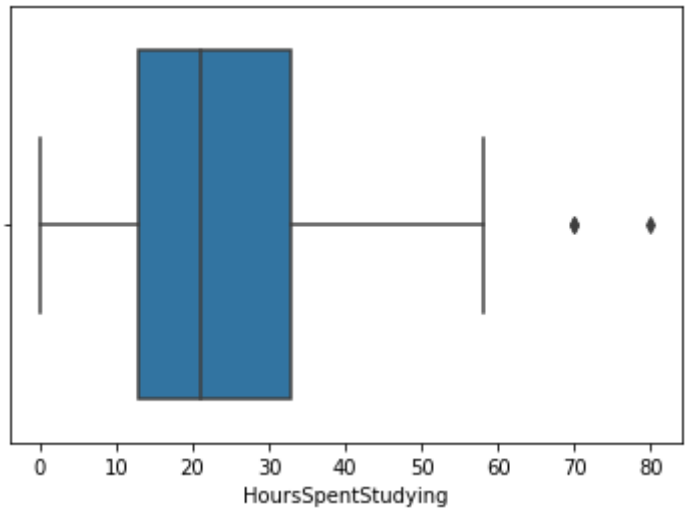


In [89]:
```
# Fill NA/NaN values for Hours with Hallmates column with 0
survey_df.HoursWithHallmates = survey_df.HoursWithHallmates.fillna(0)

# Create grouping range for Hours with Hallmates
survey_df.loc[survey_df['HoursWithHallmates'] >= 60, 'HoursWithHallmatesRangeGrp'] = 3 # people who spent more than or equal to 6
survey_df.loc[(survey_df['HoursWithHallmates'] >= 40) & (survey_df['HoursWithHallmates'] < 60) , 'HoursWithHallmatesRangeGrp'] =
survey_df.loc[(survey_df['HoursWithHallmates'] >= 20) & (survey_df['HoursWithHallmates'] < 40) , 'HoursWithHallmatesRangeGrp'] =
survey_df.loc[survey_df['HoursWithHallmates'] < 20, 'HoursWithHallmatesRangeGrp'] = 0 # people who spent less than 20 hours with
survey_df['HoursWithHallmatesRangeGrp'] = survey_df['HoursWithHallmatesRangeGrp'].astype(int) # convert this new created range gr
```

In [90]:
```
# Draw a box plot to show distributions with respect to Hours Spent Studying
# We use this as reference to create a group range

sns.boxplot(survey_df['HoursSpentStudying'])
```

```
c:\Users\Crissie\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[90]: <AxesSubplot:xlabel='HoursSpentStudying'>



In [91]:
```
# Fill NA/NaN values for Hours Spent Studying column with 0
survey_df.HoursSpentStudying = survey_df.HoursSpentStudying.fillna(0)

# Create grouping range for Hours Spent Studying
```

```
survey_df.loc[survey_df['HoursSpentStudying'] >= 60, 'HoursStudyRangeGrp'] = 3 # people who spent more than or equal to 60 hours
survey_df.loc[(survey_df['HoursSpentStudying'] >= 40) & (survey_df['HoursSpentStudying'] < 60) , 'HoursStudyRangeGrp'] = 2 # peop
survey_df.loc[(survey_df['HoursSpentStudying'] >= 20) & (survey_df['HoursSpentStudying'] < 40) , 'HoursStudyRangeGrp'] = 1 # peop
survey_df.loc[survey_df['HoursSpentStudying'] < 20, 'HoursStudyRangeGrp'] = 0 # people who spent less than 20 hours studying
survey_df['HoursStudyRangeGrp'] = survey_df['HoursStudyRangeGrp'].astype(int) # convert this new created range group for hours sp
```

In [92]:
```
# Drop columns for HoursSpentStudying and HoursWithHallmates because we have already categorised the values into range group
survey_df = survey_df.drop(['HoursSpentStudying', 'HoursWithHallmates'], axis=1)
```

In [93]:
```
survey_df['HallType_NUS'].value_counts() # return counts of unique rows for Hall Type (NUS)
```

Out[93]:
```
Halls of residence      5
Residential colleges    4
Name: HallType_NUS, dtype: int64
```

In [94]:
```
survey_df['HallType_NTU'].value_counts() # return counts of unique rows for Hall Type (NTU)
```

Out[94]:
```
Halls 1-16     17
Halls 17-22     4
Name: HallType_NTU, dtype: int64
```

In [95]:
```
# Recode Hall Type for NUS and NTU to columns to check if people living in halls or not
survey_df['isLivingNUSRC'] = np.where(survey_df["HallType_NUS"].str.contains("Residential colleges"), 1, 0)
survey_df['isLivingNTUHalls1to16'] = np.where(survey_df["HallType_NTU"].str.contains("Halls 1-16"), 1, 0)
```

In [96]:
```
# Drop columns for HallType_NUS and HallType_NTU because we have already recoded the values into binary values
survey_df = survey_df.drop(['HallType_NTU','HallType_NUS'],axis=1)
```

In [97]:
```
survey_df['RoomType'].value_counts() # return counts of unique rows for RoomType
```

Out[97]:
```
Shared    16
Single    14
Name: RoomType, dtype: int64
```

In [98]:
```
# Recode Room Type to column to check if people living in dorm is either shared or single
survey_df['isSharedRoomType'] = np.where(survey_df["RoomType"].str.contains("Shared"), 1, 0)
```

In [99]:
```
survey_df['VapingLocationAtUni'].value_counts() # return counts of unique rows for vaping locations in university
```

Out[99]:
```
In room, gatherings, CCA                                                    2
Staircase                                                                   2
Room                                                                        2
Hall room                                                                   2
Rooms, smoking area                                                         2
Dorm, smoking areas                                                         2
In their own room, hall mates room, toilet, some brave soldiers even do it in class    1
Smoking point, rooms                                                        1
Rooms, staircase, smoking point                                             1
People vape anywhere as long as no prof/security guard                      1
Rooms, toilets, smoking area                                                1
In their rooms or staircase                                                 1
toilet or near dorm areas                                                   1
bathroom, staircase, dorm room                                              1
Dorm                                                                        1
Outside campus                                                              1
Anywhere that doesnâ€™t have a prof                                          1
Smoking point                                                               1
In their dorms                                                              1
bathroom                                                                    1
dorm                                                                        1
Corridor, seminar rooms                                                     1
Hall, smoking area                                                          1
Name: VapingLocationAtUni, dtype: int64
```

In [100…
```
# Fill NA/NaN values for VapingLocationAtUni with empty string
survey_df.VapingLocationAtUni = survey_df.VapingLocationAtUni.fillna('')
```

In [101…
```
# function to encode value where user prefers vaping at indoor location
def encode_VapingIndoorLocationsAtUni(data):
    data_Arr=data.split(",")
    keyWords = ['room', 'hall', 'dorm', 'class', 'staircase', 'toilet']

    flag = 0
    for i in data_Arr:
        for j in keyWords:
            if (i.find(j) != -1):
                flag = 1
                break
    if flag == 1:
        return 1
```

```
    else:
        return 0

# function to encode value where user prefers vaping at outdoor location
def encode_VapingOutdoorLocationsAtUni(data):
    data_Arr=data.split(",")
    keyWords = ['room', 'hall', 'dorm', 'class', 'staircase', 'toilet']

    flag = 0
    for i in data_Arr:
        for j in keyWords:
            if (i.find(j) == -1):
                flag = 1
                break
    if flag == 1:
        return 1
    else:
        return 0
```

In [102…

```
# Applying the function for where user prefers vaping at indoor location
survey_df['isVapingPreferIndoor'] = survey_df['VapingLocationAtUni'].apply(encode_VapingIndoorLocationsAtUni)

# Applying the function for where user prefers vaping at outdoor location
survey_df['isVapingPreferOutdoor'] = survey_df['VapingLocationAtUni'].apply(encode_VapingOutdoorLocationsAtUni)
```

In [103…

```
# View survey dataframe with current columns
survey_df
```

Out[103]:

	Gender	University	Major	Group	VapingLocation	ReasonVapingLocation	VapeSpendingAmount	PreferenceToBeAlone	ReasonPre
0	Female	NUS	Arts/Design	Vaper	University,Dorm,Home,Friends' house,Public spaces	Dorm, to have privacy	10.0	No	wi
1	Male	NUS	STEM	Vaper	University,Dorm	Dorm, more privacy	8.0	Yes	don't ne
2	Female	NUS	Computing	Non-vaper	NaN	NaN	NaN	NaN	
3	Female	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
4	Male	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	
89	Female	NUS	Medicine	Non-vaper	NaN	NaN	NaN	NaN	
90	Male	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
91	Female	NUS	Arts/Design	Non-vaper	NaN	NaN	NaN	NaN	
92	Female	NUS	Law/Social Science	Non-vaper	NaN	NaN	NaN	NaN	
93	Male	NTU	Business	Non-vaper	NaN	NaN	NaN	NaN	

94 rows × 37 columns

In [104…
```
# Fill NA/NaN values for VapingLocation and DormEncouragesVaping with empty string and 0 respectively
survey_df.VapingLocation = survey_df.VapingLocation.fillna('')
survey_df.DormEncouragesVaping = survey_df.DormEncouragesVaping.fillna(0)
```

In [105…
```
survey_df['VapingLocation'].unique() # Looking at unique values of vaping location
```

Out[105]:
```
array(["University,Dorm,Home,Friends' house,Public spaces",
       'University,Dorm', '', "Home,Friends' house",
       "University,Dorm,Friends' house",
       "Dorm,Home,Friends' house,Public spaces",
       "University,Home,Friends' house,Public spaces", 'Home',
       "Home,Friends' house,Public spaces",
       'Public spaces,Others, please specify: drinking places',
       "University,Home,Friends' house", "University,Friends' house",
       'Home,Public spaces', "Friends' house", 'Public spaces',
       'University,Home'], dtype=object)
```

In [106…
```
# Define functions to find out whether user vape where
# There are 6 functions catered to find out if user vapes at uni, home, dorm, etc.

def isVapeUni(data):
    universities = []

    for d in data:
        locations = d.split(",")
        if 'University' in locations:
            universities.append(1)
        else:
            universities.append(0)

    return universities

def isVapeHome(data):
    home = []

    for d in data:
        locations = d.split(",")
        if 'Home' in locations:
            home.append(1)
        else:
            home.append(0)

    return home

def isVapeDorm(data):
    dorm = []
```

```
    for d in data:
        locations = d.split(",")
        if 'Dorm' in locations:
            dorm.append(1)
        else:
            dorm.append(0)

    return dorm

def isVapeFriends(data):
    friends_homes = []

    for d in data:
        locations = d.split(",")
        if "Friends' house" in locations:
            friends_homes.append(1)
        else:
            friends_homes.append(0)

    return friends_homes

def isVapePublic(data):
    public_spaces = []

    for d in data:
        locations = d.split(",")
        if "Public spaces" in locations:
            public_spaces.append(1)
        else:
            public_spaces.append(0)

    return public_spaces

def isVapeOthers(data):
    others = []

    for d in data:
        locations = d.split(",")
        if "Others" in locations:
            others.append(1)
        else:
            others.append(0)
```

```
    return others
```

In [107…
```
# Retrieving list of each different place to vape for later mapping to new columns (see below)
universities = isVapeUni(survey_df['VapingLocation'])
homes = isVapeHome(survey_df['VapingLocation'])
friends_homes = isVapeFriends(survey_df['VapingLocation'])
public_spaces = isVapePublic(survey_df['VapingLocation'])
others = isVapeOthers(survey_df['VapingLocation'])
dorms = isVapeDorm(survey_df['VapingLocation'])
```

In [108…
```
# Map list of each different place to vape to new columns created
# Columns for each different place to vape is either 0 or 1
survey_df['VapeAtHome'] = homes
survey_df['VapeAtUni'] = universities
survey_df['VapeAtDorm'] = dorms
survey_df['VapeAtFriendsHouse'] = friends_homes
survey_df['VapeAtPublicSpaces'] = public_spaces
survey_df['VapeAtOthers'] = others
```

In [109…
```
survey_df['PreferredVapingPeriod'].unique() # Looking at unique values of preferred vaping period
```

Out[109]:
```
array(['Study week,During exam season,During project submission week,Social events e.g., dinners with friends',
       'Study week,During exam season', nan, 'During exam season',
       'Social events e.g., dinners with friends',
       'Study week,During exam season,During project submission week',
       'Study week,During exam season,During project submission week,In between classes,During CCAs,Social events e.g., dinners
with friends'],
      dtype=object)
```

In [110…
```
# Fill NA/NaN values for Preferred Vaping Period column with empty string
survey_df.PreferredVapingPeriod = survey_df.PreferredVapingPeriod.fillna('')
```

In [111…
```
# Create new columns for each preferred vaping period is either 0 or 1
survey_df['isPreferredVapeStudy'] = np.where(survey_df["PreferredVapingPeriod"].str.contains("Study week"), 1, 0)
survey_df['isPreferredVapeProject'] = np.where(survey_df["PreferredVapingPeriod"].str.contains("During project submission week"),
survey_df['isPreferredVapeClass'] = np.where(survey_df["PreferredVapingPeriod"].str.contains("In between classes"), 1, 0)
survey_df['isPreferredVapeExam'] = np.where(survey_df["PreferredVapingPeriod"].str.contains("During exam season"), 1, 0)
survey_df['isPreferredVapeCCA'] = np.where(survey_df["PreferredVapingPeriod"].str.contains("During CCAs"), 1, 0)
survey_df['isPreferredVapeSocial'] = np.where(survey_df["PreferredVapingPeriod"].str.contains("Social events e.g., dinners with f
```

In [112…
```
# View survey dataframe with current columns
survey_df
```

Out[112]:

	Gender	University	Major	Group	VapingLocation	ReasonVapingLocation	VapeSpendingAmount	PreferenceToBeAlone	ReasonPre
0	Female	NUS	Arts/Design	Vaper	University,Dorm,Home,Friends' house,Public spaces	Dorm, to have privacy	10.0	No	wi
1	Male	NUS	STEM	Vaper	University,Dorm	Dorm, more privacy	8.0	Yes	don't ne
2	Female	NUS	Computing	Non-vaper	NaN	NaN	NaN	NaN	
3	Female	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
4	Male	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	
89	Female	NUS	Medicine	Non-vaper	NaN	NaN	NaN	NaN	
90	Male	NTU	STEM	Non-vaper	NaN	NaN	NaN	NaN	
91	Female	NUS	Arts/Design	Non-vaper	NaN	NaN	NaN	NaN	
92	Female	NUS	Law/Social Science	Non-vaper	NaN	NaN	NaN	NaN	
93	Male	NTU	Business	Non-vaper	NaN	NaN	NaN	NaN	

94 rows × 49 columns

In [113…
```
# Fill NA/NaN values for Motivation column with empty string
survey_df.Motivation = survey_df.Motivation.fillna('')
```

In [114…
```
survey_df['Motivation'].unique() # Find unique values for motivation
```

Out[114]:
```
array(['Stress', '', 'No reason (I regularly vape)', 'Social bonding'],
      dtype=object)
```

In [115…
```
# Define function to find out if motivation to vape is stress or not
def encode_motivation(data):
    if data == "Stress":
        return 1
    else:
        return 0
```

In [116…
```
# Apply function to find out if motivation to vape is stress or not
# Column for motivation to vape is either 0 or 1
survey_df['MotivatedbyStress'] = survey_df['Motivation'].apply(encode_motivation)
```

In [117…
```
# View survey dataframe with current columns
survey_df
```

Out[117]:

	Gender	University	Major	Group	VapingLocation	ReasonVapingLocation	VapeSpendingAmount	PreferenceToBeAlone	ReasonPre
0	Female	NUS	Arts/Design	Vaper	University,Dorm,Home,Friends' house,Public spaces	Dorm, to have privacy	10.0	No	wi
1	Male	NUS	STEM	Vaper	University,Dorm	Dorm, more privacy	8.0	Yes	don't ne
2	Female	NUS	Computing	Non-vaper		NaN	NaN	NaN	
3	Female	NTU	STEM	Non-vaper		NaN	NaN	NaN	
4	Male	NTU	STEM	Non-vaper		NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	
89	Female	NUS	Medicine	Non-vaper		NaN	NaN	NaN	
90	Male	NTU	STEM	Non-vaper		NaN	NaN	NaN	
91	Female	NUS	Arts/Design	Non-vaper		NaN	NaN	NaN	
92	Female	NUS	Law/Social Science	Non-vaper		NaN	NaN	NaN	
93	Male	NTU	Business	Non-vaper		NaN	NaN	NaN	

94 rows × 50 columns

In [118…

```
survey_df['Exposure'].unique() # Looking at unique values of how people are exposed to concept of vaping
```

Out[118]:
```
array(['Friends who vape', 'Social media,Family members who vape',
       'Social media,Friends who vape,Campaigns against vaping',
       'Social media,Friends who vape', 'Family members who vape',
       'Social media', 'Promotional ads',
       'Social media,Family members who vape,Friends who vape,Campaigns against vaping',
       'Social media,Friends who vape,Promotional ads',
       'Social media,Promotional ads,Campaigns against vaping',
       'Family members who vape,Friends who vape',
       'Social media,Family members who vape,Friends who vape',
       'Friends who vape,Campaigns against vaping',
       'Social media,Campaigns against vaping',
       'Social media,Family members who vape,Campaigns against vaping',
       'Social media,Friends who vape,Promotional ads,Campaigns against vaping'],
      dtype=object)
```

In [119…
```
# Define functions to find out whether how people are exposed to vaping
# There are 5 functions catered to find out if how user is exposed to concept of vaping

def encode_exposure_friends(data):
    arr = data.split(",")
    if "Friends who vape" in arr:
        return 1
    else:
        return 0

def encode_exposure_family(data):
    arr = data.split(",")
    if "Family members who vape" in arr:
        return 1
    else:
        return 0

def encode_exposure_campaigns(data):
    arr = data.split(",")
    if "Campaigns against vaping" in arr:
        return 1
    else:
        return 0

def encode_exposure_promo(data):
    arr = data.split(",")
```

```
    if "Promotional ads" in arr:
        return 1
    else:
        return 0

def encode_exposure_socialmedia(data):
    arr = data.split(",")
    if "Social media" in arr:
        return 1
    else:
        return 0
```

In [120…

```
# Apply functions (see above) to know how user is exposed to concept of vaping
survey_df['ExposureFromFriends'] = survey_df['Exposure'].apply(encode_exposure_friends)
survey_df['ExposureFromFamily'] = survey_df['Exposure'].apply(encode_exposure_family)
survey_df['ExposureFromCampaigns'] = survey_df['Exposure'].apply(encode_exposure_campaigns)
survey_df['ExposureFromPromo'] = survey_df['Exposure'].apply(encode_exposure_promo)
survey_df['ExposureFromSocialMedia'] = survey_df['Exposure'].apply(encode_exposure_socialmedia)
```

In [121…

```
# View current columns of survey dataframe
survey_df
```

Out[121]:

	Gender	University	Major	Group	VapingLocation	ReasonVapingLocation	VapeSpendingAmount	PreferenceToBeAlone	ReasonPre
0	Female	NUS	Arts/Design	Vaper	University,Dorm,Home,Friends' house,Public spaces	Dorm, to have privacy	10.0	No	wi
1	Male	NUS	STEM	Vaper	University,Dorm	Dorm, more privacy	8.0	Yes	don't ne
2	Female	NUS	Computing	Non-vaper		NaN	NaN	NaN	
3	Female	NTU	STEM	Non-vaper		NaN	NaN	NaN	
4	Male	NTU	STEM	Non-vaper		NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	
89	Female	NUS	Medicine	Non-vaper		NaN	NaN	NaN	
90	Male	NTU	STEM	Non-vaper		NaN	NaN	NaN	
91	Female	NUS	Arts/Design	Non-vaper		NaN	NaN	NaN	
92	Female	NUS	Law/Social Science	Non-vaper		NaN	NaN	NaN	
93	Male	NTU	Business	Non-vaper		NaN	NaN	NaN	

94 rows × 55 columns

In [122…
```
# Create new column for whether user saw vapers in uni is either 0 or 1
survey_df['SawVapersAtUni'] = np.where(survey_df["SeeVapersAtUni"].str.contains("Yes"), 1, 0)
```

In [123…
```
# Check unique values for people are encouraged by friends to vape
survey_df['EncouragedByFriendsToVape'].unique()
```

Out[123]:
```
array(['Yes', 'No', nan], dtype=object)
```

In [124…
```
# Fill NA/NaN values for EncouragedByFriendsToVape with "No"
survey_df.EncouragedByFriendsToVape = survey_df.EncouragedByFriendsToVape.fillna('No')
```

In [125…
```
# Define function to encode whether users are encouraged by friends to vape

def encode_CompelledtoVape(txt):
    sentence = txt.lower()
    txt_arr = sentence.split(" ")
    no_arr = ["no", "nope", "-"]
    yes_arr = ["yes", "follow"]

    for word in txt_arr:
        if word in no_arr:
            return 0
        elif word in yes_arr:
            return 1
```

In [126…
```
# Apply function (see above) to encode wheher users are encouraged by friends to vape
survey_df['isCompelledtoVape'] = survey_df['EncouragedByFriendsToVape'].apply(encode_CompelledtoVape)
```

In [127…
```
# Check current columns in survey dataframe
survey_df
```

Out[127]:

	Gender	University	Major	Group	VapingLocation	ReasonVapingLocation	VapeSpendingAmount	PreferenceToBeAlone	ReasonPre
0	Female	NUS	Arts/Design	Vaper	University,Dorm,Home,Friends' house,Public spaces	Dorm, to have privacy	10.0	No	wi
1	Male	NUS	STEM	Vaper	University,Dorm	Dorm, more privacy	8.0	Yes	don't ne
2	Female	NUS	Computing	Non-vaper		NaN	NaN	NaN	
3	Female	NTU	STEM	Non-vaper		NaN	NaN	NaN	
4	Male	NTU	STEM	Non-vaper		NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	
89	Female	NUS	Medicine	Non-vaper		NaN	NaN	NaN	
90	Male	NTU	STEM	Non-vaper		NaN	NaN	NaN	
91	Female	NUS	Arts/Design	Non-vaper		NaN	NaN	NaN	
92	Female	NUS	Law/Social Science	Non-vaper		NaN	NaN	NaN	
93	Male	NTU	Business	Non-vaper		NaN	NaN	NaN	

94 rows × 57 columns

In [128…

```
survey_df.info() # see which columns are still having null values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 94 entries, 0 to 93
Data columns (total 57 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   Gender                          94 non-null     object 
 1   University                      94 non-null     object 
 2   Major                           94 non-null     object 
 3   Group                           94 non-null     object 
 4   VapingLocation                  94 non-null     object 
 5   ReasonVapingLocation            5 non-null      object 
 6   VapeSpendingAmount              36 non-null     float64
 7   PreferenceToBeAlone             13 non-null     object 
 8   ReasonPreferenceToBeAlone       13 non-null     object 
 9   PreferredVapingPeriod           94 non-null     object 
 10  Motivation                      94 non-null     object 
 11  ReasonStress                    17 non-null     object 
 12  EncourageVaping                 36 non-null     object 
 13  ReasonEncourageVaping           4 non-null      object 
 14  Exposure                        94 non-null     object 
 15  PerceptionVapingBeforeU         94 non-null     int64  
 16  PerceptionVapingAfterU          94 non-null     int64  
 17  Dorm                            94 non-null     object 
 18  RoomType                        30 non-null     object 
 19  DaysInDorm                      30 non-null     float64
 20  DormEncouragesVaping            94 non-null     float64
 21  ReasonDormEncouragesVaping      30 non-null     object 
 22  SeeVapersAtUni                  94 non-null     object 
 23  VapingLocationAtUni             94 non-null     object 
 24  UniversityDiscouragingVaping    94 non-null     int64  
 25  UniversityRaiseAwarenessVaping  94 non-null     int64  
 26  VaperFriends                    94 non-null     int64  
 27  VaperFriendsDormStayer          94 non-null     int64  
 28  EncouragedByFriendsToVape       94 non-null     object 
 29  CompelledToVape                 24 non-null     object 
 30  HoursWithHallmatesRangeGrp      94 non-null     int32  
 31  HoursStudyRangeGrp              94 non-null     int32  
 32  isLivingNUSRC                   94 non-null     int32  
 33  isLivingNTUHalls1to16           94 non-null     int32  
 34  isSharedRoomType                94 non-null     int32  
 35  isVapingPreferIndoor            94 non-null     int64  
 36  isVapingPreferOutdoor           94 non-null     int64  
```

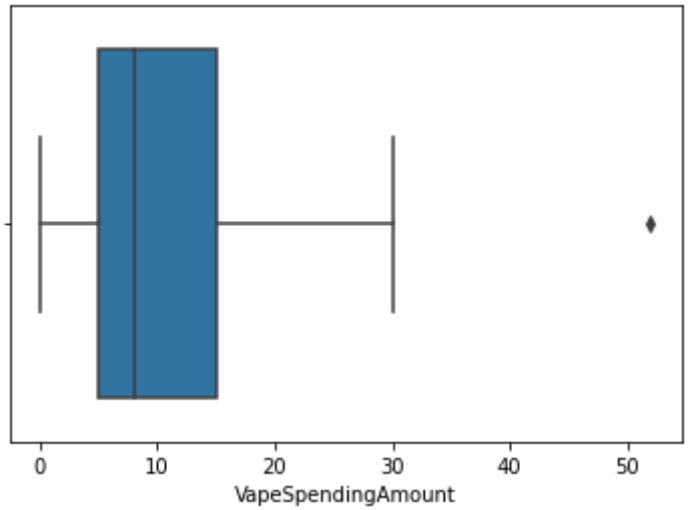
```
 37  VapeAtHome                    94 non-null     int64
 38  VapeAtUni                     94 non-null     int64
 39  VapeAtDorm                    94 non-null     int64
 40  VapeAtFriendsHouse            94 non-null     int64
 41  VapeAtPublicSpaces            94 non-null     int64
 42  VapeAtOthers                  94 non-null     int64
 43  isPreferredVapeStudy          94 non-null     int32
 44  isPreferredVapeProject        94 non-null     int32
 45  isPreferredVapeClass          94 non-null     int32
 46  isPreferredVapeExam           94 non-null     int32
 47  isPreferredVapeCCA            94 non-null     int32
 48  isPreferredVapeSocial         94 non-null     int32
 49  MotivatedbyStress             94 non-null     int64
 50  ExposureFromFriends           94 non-null     int64
 51  ExposureFromFamily            94 non-null     int64
 52  ExposureFromCampaigns         94 non-null     int64
 53  ExposureFromPromo             94 non-null     int64
 54  ExposureFromSocialMedia       94 non-null     int64
 55  SawVapersAtUni                94 non-null     int32
 56  isCompelledtoVape             94 non-null     int64
dtypes: float64(3), int32(12), int64(21), object(21)
memory usage: 37.6+ KB
```

In [129…

```
# Draw a box plot to show distributions with respect to Vape Spending Amount
# We use this as reference to create a group range

sns.boxplot(survey_df['VapeSpendingAmount'])
```

```
c:\Users\Crissie\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword
arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[129]: <AxesSubplot:xlabel='VapeSpendingAmount'>



In [130…
```
# Fill NA/NaN values for Vape Spending Amount column with 0
survey_df.VapeSpendingAmount = survey_df.VapeSpendingAmount.fillna(0)

# Create grouping range for Vape Spending Amount
# Similar methodology to Hours Spent Studying
survey_df.loc[survey_df['VapeSpendingAmount'] >= 50, 'VapeAmtRangeGrp'] = 5
survey_df.loc[(survey_df['VapeSpendingAmount'] >= 40) & (survey_df['VapeSpendingAmount'] < 50) , 'VapeAmtRangeGrp'] = 4
survey_df.loc[(survey_df['VapeSpendingAmount'] >= 30) & (survey_df['VapeSpendingAmount'] < 40) , 'VapeAmtRangeGrp'] = 3
survey_df.loc[(survey_df['VapeSpendingAmount'] >= 20) & (survey_df['VapeSpendingAmount'] < 30) , 'VapeAmtRangeGrp'] = 2
survey_df.loc[(survey_df['VapeSpendingAmount'] >= 10) & (survey_df['VapeSpendingAmount'] < 20) , 'VapeAmtRangeGrp'] = 1
survey_df.loc[survey_df['VapeSpendingAmount'] < 10, 'VapeAmtRangeGrp'] = 0
survey_df['VapeAmtRangeGrp'] = survey_df['VapeAmtRangeGrp'].astype(int) # convert this new created range group for vape amount sp
```

In [131…
```
# Create grouping range for UniversityDiscouragingVaping
# Similar methodology to Vape Spending Amount
survey_df.loc[(survey_df['UniversityDiscouragingVaping'] >= 4) & (survey_df['UniversityDiscouragingVaping'] <= 5) , 'UniSatisfyDi
survey_df.loc[(survey_df['UniversityDiscouragingVaping'] >= 1) & (survey_df['UniversityDiscouragingVaping'] <= 3) , 'UniSatisfyDi
survey_df['UniSatisfyDiscourageVaping'] = survey_df['UniSatisfyDiscourageVaping'].astype(int)

# Drop UniversityDiscouragingVaping column since range grouping is defined
survey_df = survey_df.drop('UniversityDiscouragingVaping',axis=1)
```

In [132…
```
# Create grouping range for UniversityDiscouragingVaping
# Similar methodology to UniSatisfyDiscourageVaping
```

```
survey_df.loc[(survey_df['UniversityRaiseAwarenessVaping'] >= 4) & (survey_df['UniversityRaiseAwarenessVaping'] <= 5) , 'UniSatis
survey_df.loc[(survey_df['UniversityRaiseAwarenessVaping'] >= 1) & (survey_df['UniversityRaiseAwarenessVaping'] <= 3) , 'UniSatis
survey_df['UniSatisfyDiscourageVaping'] = survey_df['UniSatisfyDiscourageVaping'].astype(int)

# Drop UniversityRaiseAwarenessVaping column since range grouping is defined
survey_df = survey_df.drop('UniversityRaiseAwarenessVaping',axis=1)
```

In [133…

```
# Recode Dorm to binary values to know if user stay in dorm or not
survey_df['IsStayDorm'] = np.where(survey_df["Dorm"].str.contains("Yes"), 1, 0)
```

In [134…

```
# Recode Dorm to binary values to know if user prefers vape alone or not
survey_df['isPreferVapeAlone'] = np.where(survey_df["PreferenceToBeAlone"].str.contains("Yes"), 1, 0)
```

In [135…

```
# Creating a new survey dataframe from existing one to avoid confusion
# This will be our current dataset
survey_df1 = survey_df.drop(['VapeSpendingAmount', 'Motivation', 'ReasonVapingLocation', 'CompelledToVape', 'PreferredVapingPerio
survey_df1 = survey_df1.drop(['Dorm','Exposure','PerceptionVapingBeforeU','PerceptionVapingAfterU'], axis=1)
survey_df1
```

Out[135]:

	Gender	University	Major	Group	RoomType	DaysInDorm	DormEncouragesVaping	ReasonDormEncouragesVaping	SeeVapersAtUni	VapingLo
0	Female	NUS	Arts/Design	Vaper	Single	6.0	4.0	no one cares as long as u don't get caught	Yes	In room
1	Male	NUS	STEM	Vaper	Single	5.0	3.0	your room and some places in the hall is very ...	Yes	Dorm, sr
2	Female	NUS	Computing	Non-vaper	NaN	NaN	0.0	NaN	Yes	Corr
3	Female	NTU	STEM	Non-vaper	Single	7.0	3.0	No enforcement and no checking	Yes	
4	Male	NTU	STEM	Non-vaper	Shared	5.0	3.0	no one cares even if you do openly	No	
...	...	...	...	...	...	...	...	...	...	
89	Female	NUS	Medicine	Non-vaper	NaN	NaN	0.0	NaN	No	
90	Male	NTU	STEM	Non-vaper	NaN	NaN	0.0	NaN	Yes	
91	Female	NUS	Arts/Design	Non-vaper	NaN	NaN	0.0	NaN	Yes	
92	Female	NUS	Law/Social Science	Non-vaper	NaN	NaN	0.0	NaN	No	
93	Male	NTU	Business	Non-vaper	NaN	NaN	0.0	NaN	Yes	

94 rows × 45 columns

In [136…

```
# Fill NA/NaN values for EncouragedByFriendsToVape with 0
survey_df1.EncouragedByFriendsToVape = survey_df1.EncouragedByFriendsToVape.fillna(0)

# Create new column for whether user is encouraged by friends to vape is either 0 or 1
survey_df1['IsEncouragedByFriendsToVape'] = np.where(survey_df1["EncouragedByFriendsToVape"].str.contains("Yes"), 1, 0)
```

```
# Drop EncouragedByFriendsToVape column since new column is created with binary values
survey_df1 = survey_df1.drop('EncouragedByFriendsToVape', axis=1)
```

In [137…
```
# Drop the following columns since we have also created the new columns with encoded values
survey_df1 = survey_df1.drop(['RoomType','SeeVapersAtUni','VapingLocationAtUni', 'ReasonDormEncouragesVaping'], axis=1)
```

In [138…
```
survey_df1['DormEncouragesVaping'].value_counts # know what is the value available in DormEncouragesVaping column
```

Out[138]:
```
<bound method IndexOpsMixin.value_counts of 0     4.0
1     3.0
2     0.0
3     3.0
4     3.0
     ...
89    0.0
90    0.0
91    0.0
92    0.0
93    0.0
Name: DormEncouragesVaping, Length: 94, dtype: float64>
```

In [139…
```
# Create grouping range for DormEncouragesVaping
# Similar methodology to Hours Spent Studying
survey_df1.loc[survey_df1['DormEncouragesVaping'] >= 5, 'isDormEncourageVaping'] = 1
survey_df1.loc[(survey_df1['DormEncouragesVaping'] >= 4) & (survey_df1['DormEncouragesVaping'] < 5) , 'isDormEncourageVaping'] =
survey_df1.loc[(survey_df1['DormEncouragesVaping'] >= 3) & (survey_df1['DormEncouragesVaping'] < 4) , 'isDormEncourageVaping'] =
survey_df1.loc[(survey_df1['DormEncouragesVaping'] >= 2) & (survey_df1['DormEncouragesVaping'] < 3) , 'isDormEncourageVaping'] =
survey_df1.loc[(survey_df1['DormEncouragesVaping'] >= 1) & (survey_df1['DormEncouragesVaping'] < 2) , 'isDormEncourageVaping'] =
survey_df1.loc[(survey_df1['DormEncouragesVaping'] == None) , 'isDormEncourageVaping'] = 0
```

In [140…
```
# Fill NA/NaN values for isDormEncourageVaping with 0
survey_df1.isDormEncourageVaping = survey_df1.isDormEncourageVaping.fillna(0)

# Drop DormEncouragesVaping column since new column is created with binary values
survey_df1 = survey_df1.drop('DormEncouragesVaping',axis=1)
```

In [141…
```
# View current columns in current survey dataframe
survey_df1
```

Out[141]:

	Gender	University	Major	Group	DaysInDorm	VaperFriends	VaperFriendsDormStayer	HoursWithHallmatesRangeGrp	HoursStudyRangeGrp	i
0	Female	NUS	Arts/Design	Vaper	6.0	7	7	1	1	
1	Male	NUS	STEM	Vaper	5.0	8	8	1	1	
2	Female	NUS	Computing	Non-vaper	NaN	3	0	0	1	
3	Female	NTU	STEM	Non-vaper	7.0	2	2	0	2	
4	Male	NTU	STEM	Non-vaper	5.0	0	0	3	1	
...	...	...	...	...	...	...	...	...	...	...
89	Female	NUS	Medicine	Non-vaper	NaN	0	0	0	0	
90	Male	NTU	STEM	Non-vaper	NaN	5	4	0	1	
91	Female	NUS	Arts/Design	Non-vaper	NaN	5	3	0	0	
92	Female	NUS	Law/Social Science	Non-vaper	NaN	3	3	0	0	
93	Male	NTU	Business	Non-vaper	NaN	13	7	0	0	

94 rows × 41 columns

In [142…

```
# Checking again which columns still have missing values
survey_df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 94 entries, 0 to 93
Data columns (total 41 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Gender                        94 non-null     object 
 1   University                    94 non-null     object 
 2   Major                         94 non-null     object 
 3   Group                         94 non-null     object 
 4   DaysInDorm                    30 non-null     float64
 5   VaperFriends                  94 non-null     int64  
 6   VaperFriendsDormStayer        94 non-null     int64  
 7   HoursWithHallmatesRangeGrp    94 non-null     int32  
 8   HoursStudyRangeGrp            94 non-null     int32  
 9   isLivingNUSRC                 94 non-null     int32  
 10  isLivingNTUHalls1to16         94 non-null     int32  
 11  isSharedRoomType              94 non-null     int32  
 12  isVapingPreferIndoor          94 non-null     int64  
 13  isVapingPreferOutdoor         94 non-null     int64  
 14  VapeAtHome                    94 non-null     int64  
 15  VapeAtUni                     94 non-null     int64  
 16  VapeAtDorm                    94 non-null     int64  
 17  VapeAtFriendsHouse            94 non-null     int64  
 18  VapeAtPublicSpaces            94 non-null     int64  
 19  VapeAtOthers                  94 non-null     int64  
 20  isPreferredVapeStudy          94 non-null     int32  
 21  isPreferredVapeProject        94 non-null     int32  
 22  isPreferredVapeClass          94 non-null     int32  
 23  isPreferredVapeExam           94 non-null     int32  
 24  isPreferredVapeCCA            94 non-null     int32  
 25  isPreferredVapeSocial         94 non-null     int32  
 26  MotivatedbyStress             94 non-null     int64  
 27  ExposureFromFriends           94 non-null     int64  
 28  ExposureFromFamily            94 non-null     int64  
 29  ExposureFromCampaigns         94 non-null     int64  
 30  ExposureFromPromo             94 non-null     int64  
 31  ExposureFromSocialMedia       94 non-null     int64  
 32  SawVapersAtUni                94 non-null     int32  
 33  isCompelledtoVape             94 non-null     int64  
 34  VapeAmtRangeGrp               94 non-null     int32  
 35  UniSatisfyDiscourageVaping    94 non-null     int32  
 36  UniSatisfyRaiseAwareVaping    94 non-null     float64
```

```
 37  IsStayDorm                  94 non-null     int32
 38  isPreferVapeAlone           94 non-null     int32
 39  IsEncouragedByFriendsToVape  94 non-null     int32
 40  isDormEncourageVaping       94 non-null     float64
dtypes: float64(3), int32(17), int64(17), object(4)
memory usage: 24.0+ KB
```

In [143…

```
# Create new column for Gender to find out if gender is male or not
# Drop Gender column since new column is created
survey_df1['isMale'] = np.where(survey_df1["Gender"] == "Male", 1, 0)
survey_df1 = survey_df1.drop('Gender', axis=1)

# Create new column for University to find out if university is NUS or NTU
# Drop University column since new column is created
survey_df1['isNUS'] = np.where(survey_df1["University"] == "NUS", 1, 0)
survey_df1 = survey_df1.drop('University', axis=1)

# Create new column for Group to find out if user is a vaper user or not
# Drop Group column since new column is created
survey_df1['isVaper'] = np.where(survey_df1["Group"] == "Vaper", 1, 0)
survey_df1 = survey_df1.drop('Group', axis=1)

# Fill NA/NaN values for Days in Dorm with 0
survey_df1.DaysInDorm = survey_df1.DaysInDorm.fillna(0)

# View current columns in survey dataframe
survey_df1
```

Out[143]:

	Major	DaysInDorm	VaperFriends	VaperFriendsDormStayer	HoursWithHallmatesRangeGrp	HoursStudyRangeGrp	isLivingNUSRC	isLivingNTUH
0	Arts/Design	6.0	7	7	1	1	0	
1	STEM	5.0	8	8	1	1	0	
2	Computing	0.0	3	0	0	1	1	
3	STEM	7.0	2	2	0	2	1	
4	STEM	5.0	0	0	3	1	1	
...	...	...	...	...	...	...	...	...
89	Medicine	0.0	0	0	0	0	1	
90	STEM	0.0	5	4	0	1	1	
91	Arts/Design	0.0	5	3	0	0	1	
92	Law/Social Science	0.0	3	3	0	0	1	
93	Business	0.0	13	7	0	0	1	

94 rows × 41 columns

In [144…
```
survey_df1['Major'].value_counts() # Return list of unique values' counts for Major
```

Out[144]:
```
STEM                30
Computing           22
Business            18
Law/Social Science  13
Arts/Design          8
Medicine             3
Name: Major, dtype: int64
```

In [145…
```
# Create grouping range for Major
# Similar methodology to Hours Spent Studying
survey_df1.loc[survey_df1['Major'] == 'STEM', 'Major'] = 0
survey_df1.loc[survey_df1['Major'] == 'Computing', 'Major'] = 1
survey_df1.loc[survey_df1['Major'] == 'Business', 'Major'] = 2
survey_df1.loc[survey_df1['Major'] == 'Law/Social Science', 'Major'] = 3
```

```
survey_df1.loc[survey_df1['Major'] == 'Arts/Design', 'Major'] = 4
survey_df1.loc[survey_df1['Major'] == 'Medicine', 'Major'] = 5
survey_df1['Major'] = survey_df1['Major'].astype(int) # convert column to integer instead of letting pandas decide

# View current columns in survey dataframe
survey_df1
```

Out[145]:

	Major	DaysInDorm	VaperFriends	VaperFriendsDormStayer	HoursWithHallmatesRangeGrp	HoursStudyRangeGrp	isLivingNUSRC	isLivingNTUHalls1t
0	4	6.0	7	7	1	1	0	
1	0	5.0	8	8	1	1	0	
2	1	0.0	3	0	0	1	1	
3	0	7.0	2	2	0	2	1	
4	0	5.0	0	0	3	1	1	
...	...	...	...	...	...	...	...	
89	5	0.0	0	0	0	0	1	
90	0	0.0	5	4	0	1	1	
91	4	0.0	5	3	0	0	1	
92	3	0.0	3	3	0	0	1	
93	2	0.0	13	7	0	0	1	

94 rows × 41 columns

In [146...

```
# Check columns for any null values
# All the columns have been cleaned and converted to numeric/binary values for logistic regression model setup
survey_df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 94 entries, 0 to 93
Data columns (total 41 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Major                         94 non-null     int32
 1   DaysInDorm                    94 non-null     float64
 2   VaperFriends                  94 non-null     int64
 3   VaperFriendsDormStayer        94 non-null     int64
 4   HoursWithHallmatesRangeGrp    94 non-null     int32
 5   HoursStudyRangeGrp            94 non-null     int32
 6   isLivingNUSRC                 94 non-null     int32
 7   isLivingNTUHalls1to16         94 non-null     int32
 8   isSharedRoomType              94 non-null     int32
 9   isVapingPreferIndoor          94 non-null     int64
 10  isVapingPreferOutdoor         94 non-null     int64
 11  VapeAtHome                    94 non-null     int64
 12  VapeAtUni                     94 non-null     int64
 13  VapeAtDorm                    94 non-null     int64
 14  VapeAtFriendsHouse            94 non-null     int64
 15  VapeAtPublicSpaces            94 non-null     int64
 16  VapeAtOthers                  94 non-null     int64
 17  isPreferredVapeStudy          94 non-null     int32
 18  isPreferredVapeProject        94 non-null     int32
 19  isPreferredVapeClass          94 non-null     int32
 20  isPreferredVapeExam           94 non-null     int32
 21  isPreferredVapeCCA            94 non-null     int32
 22  isPreferredVapeSocial         94 non-null     int32
 23  MotivatedbyStress             94 non-null     int64
 24  ExposureFromFriends           94 non-null     int64
 25  ExposureFromFamily            94 non-null     int64
 26  ExposureFromCampaigns         94 non-null     int64
 27  ExposureFromPromo             94 non-null     int64
 28  ExposureFromSocialMedia       94 non-null     int64
 29  SawVapersAtUni                94 non-null     int32
 30  isCompelledtoVape             94 non-null     int64
 31  VapeAmtRangeGrp               94 non-null     int32
 32  UniSatisfyDiscourageVaping    94 non-null     int32
 33  UniSatisfyRaiseAwareVaping    94 non-null     float64
 34  IsStayDorm                    94 non-null     int32
 35  isPreferVapeAlone             94 non-null     int32
 36  IsEncouragedByFriendsToVape   94 non-null     int32
```

```
 37  isDormEncourageVaping        94 non-null     float64
 38  isMale                       94 non-null     int32
 39  isNUS                        94 non-null     int32
 40  isVaper                      94 non-null     int32
dtypes: float64(3), int32(21), int64(17)
memory usage: 22.5 KB
```

In [147…
```
# View available columns for the survey dataframe to create and train logistic regression model
survey_df1.columns
```

Out[147]:
```
Index(['Major', 'DaysInDorm', 'VaperFriends', 'VaperFriendsDormStayer',
       'HoursWithHallmatesRangeGrp', 'HoursStudyRangeGrp', 'isLivingNUSRC',
       'isLivingNTUHalls1to16', 'isSharedRoomType', 'isVapingPreferIndoor',
       'isVapingPreferOutdoor', 'VapeAtHome', 'VapeAtUni', 'VapeAtDorm',
       'VapeAtFriendsHouse', 'VapeAtPublicSpaces', 'VapeAtOthers',
       'isPreferredVapeStudy', 'isPreferredVapeProject',
       'isPreferredVapeClass', 'isPreferredVapeExam', 'isPreferredVapeCCA',
       'isPreferredVapeSocial', 'MotivatedbyStress', 'ExposureFromFriends',
       'ExposureFromFamily', 'ExposureFromCampaigns', 'ExposureFromPromo',
       'ExposureFromSocialMedia', 'SawVapersAtUni', 'isCompelledtoVape',
       'VapeAmtRangeGrp', 'UniSatisfyDiscourageVaping',
       'UniSatisfyRaiseAwareVaping', 'IsStayDorm', 'isPreferVapeAlone',
       'IsEncouragedByFriendsToVape', 'isDormEncourageVaping', 'isMale',
       'isNUS', 'isVaper'],
      dtype='object')
```

In [148…
```
# Get the predictors and target; the x and y variables from the dataset

columns = ['VaperFriends', 'VaperFriendsDormStayer',
       'HoursStudyRangeGrp',
       'isPreferredVapeStudy', 'isPreferredVapeProject',
       'isPreferredVapeClass', 'isPreferredVapeExam', 'isPreferredVapeCCA',
       'isPreferredVapeSocial', 'MotivatedbyStress', 'ExposureFromFriends',
       'ExposureFromFamily', 'ExposureFromCampaigns', 'ExposureFromPromo',
       'ExposureFromSocialMedia', 'SawVapersAtUni',
       'UniSatisfyDiscourageVaping',
       'UniSatisfyRaiseAwareVaping', 'isPreferVapeAlone',
       'IsEncouragedByFriendsToVape']
x = survey_df1[columns] #indep variables
y = survey_df1['isVaper'] #dependent variable (target)
```

In [149…
```
# Split arrays into random train and test subsets.
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.3,
                                                    stratify = y,
                                                    random_state = 1)
```

In [150…
```
# Create a Logistic Regression model
lmodel = LogisticRegression(solver = 'liblinear', random_state = 1)

# Train Logistic Regression model
lmodel.fit(X_train,y_train)
```

Out[150]:
LogisticRegression

LogisticRegression(random_state=1, solver='liblinear')

In [151…
```
# View the Logistic Regression model's coefficients
lmodel.coef_
```

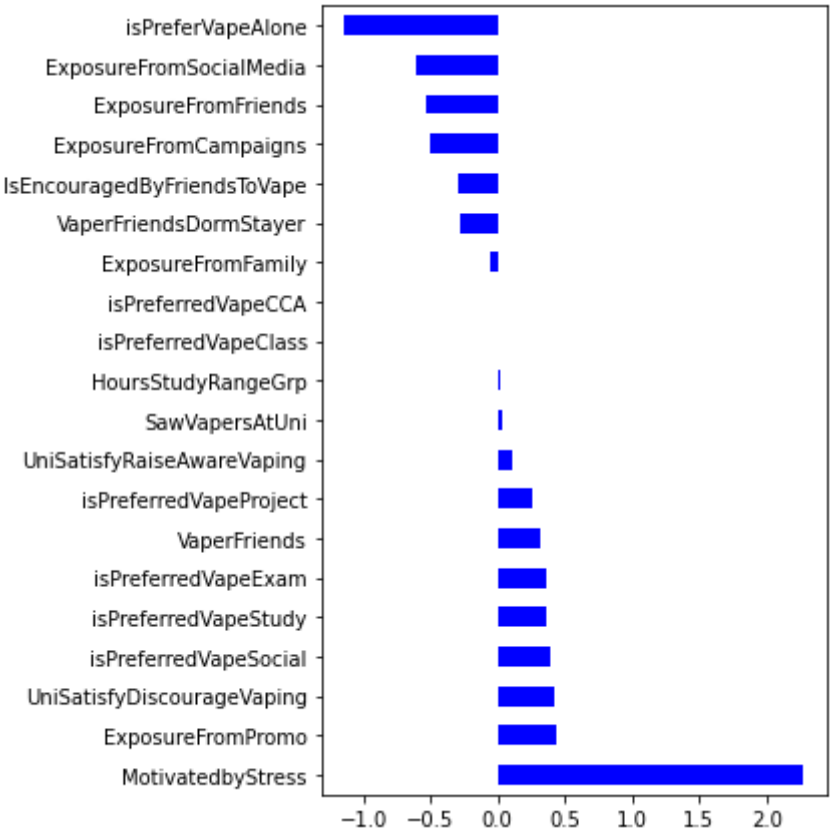
Out[151]:
```
array([[ 0.31790524, -0.28325745,  0.01722461,  0.35521013,  0.25103493,
         0.        ,  0.35521013,  0.        ,  0.38566733,  2.27144897,
        -0.53219853, -0.05596528, -0.50397246,  0.44371597, -0.60519661,
         0.03877237,  0.42171654,  0.10417519, -1.14036924, -0.29431967]])
```

In [152…
```
# Store the coefficients in a Series along with the column names
lm_coef = pd.Series(lmodel.coef_[0], index = x.columns)

# Sort the absolute values of the coefficients
sorted_coef = lm_coef.sort_values(key=pd.Series,ascending=False)
```

In [153…
```
sorted_coef
```

Out[153]:
```
MotivatedbyStress              2.271449
ExposureFromPromo              0.443716
UniSatisfyDiscourageVaping     0.421717
isPreferredVapeSocial          0.385667
isPreferredVapeStudy           0.355210
isPreferredVapeExam            0.355210
VaperFriends                   0.317905
isPreferredVapeProject         0.251035
UniSatisfyRaiseAwareVaping     0.104175
SawVapersAtUni                 0.038772
HoursStudyRangeGrp             0.017225
isPreferredVapeClass           0.000000
isPreferredVapeCCA             0.000000
ExposureFromFamily            -0.055965
VaperFriendsDormStayer        -0.283257
IsEncouragedByFriendsToVape   -0.294320
ExposureFromCampaigns         -0.503972
ExposureFromFriends           -0.532199
ExposureFromSocialMedia       -0.605197
isPreferVapeAlone             -1.140369
dtype: float64
```

In [154…
```
# Make a horizontal bar plot
sorted_coef.plot(kind='barh', color='blue', figsize = (6,6))
plt.tight_layout()
plt.rcParams["figure.facecolor"] = "w"
plt.savefig('Importance features for logistic regression vapers')
plt.show()
```



In [155…
```
# Makr predictions on Logistic Regression model
y_pred1 = lmodel.predict(X_test)

print("Accuracy for LogisticRegression :")
print(round(accuracy_score(y_test, y_pred1),2))
```

```
Accuracy for LogisticRegression :
0.83
```

In [156…
```
# Print the confusion matrix as a DataFrame with columns
cnf_matrix1 = confusion_matrix(y_test, y_pred1, labels = [1,0])
cf1 = pd.DataFrame(cnf_matrix1, columns = ['Predicted pos', 'Predicted neg'], index = ['Actual pos', 'Actual neg'])
cf1
```

Out[156]:

	Predicted pos	Predicted neg
Actual pos	9	2
Actual neg	3	15

In [157…

```
tn, fp, fn, tp = confusion_matrix(y_test, y_pred1).ravel()
specificity = tn / (tn+fp)
precision = tp / (tp + fp)
recall_or_sensitivity = tp / (tp + fn)

print("Specificity :", round(specificity,2))
print("Precision :", round(precision, 2))
print("Recall or Sensitivity :", round(recall_or_sensitivity, 2))
```

```
Specificity : 0.83
Precision : 0.75
Recall or Sensitivity : 0.82
```

In [158…

```
print(classification_report(y_test, y_pred1))
```

```
              precision    recall  f1-score   support

           0       0.88      0.83      0.86        18
           1       0.75      0.82      0.78        11

    accuracy                           0.83        29
   macro avg       0.82      0.83      0.82        29
weighted avg       0.83      0.83      0.83        29
```

In [ ]: