ASSIGNMENT TWO – INDEPENDENT PROJECT

PROGRAMMING FOR SOCIAL SCIENTISTS

GEOG5995

University of Leeds

*Word Count – 795*

**DATASET**

In order to write my programme I used data from the NHS Digital (2016) ‘Smoking, Drinking and Drug Use Among Young People in England’ survey, which can be obtained from <https://beta.ukdataservice.ac.uk/datacatalogue/studies/study?id=8320> subject to access permissions. From here the file ‘sdd\_archive.tab’ was downloaded which is required in a desktop folder in order for the code to work.

**INTENTIONS AND PROCESS**

As part of a larger PhD research project I decided to create a piece of software that can be used for statistically analysing the relationship between drug use and wellbeing in school-age adolescents in the UK. My project involves looking at drug use in family law cases and uses forensic samples to attempt to accurately predict whether or not a person has taken a drug. Due to sensitivity issues I was unable to obtain a dataset that examined this relationship, so I decided to choose something similar which was the Smoking, Drinking and Drug Use survey conducted by NHS Digital (2016). This way, I could examine how drug use may affect a person’s life – in this case in terms of their wellbeing.

I decided to use linear regression and logistic regression analysis for this research as they seemed to be the best methods for analysing the data I had. With my limited knowledge of Python and my more in depth knowledge of regression analysis I thought this would be a good start for producing my first real coding assignment, as I could be comfortable with the statistics where I may not be sure on the code.

Firstly, I opened the data from a .tab file using Pandas. I then used Numpy to create functions which could deal with missing values and binary variables in order to clean the dataset in preparation for the analysis. I then used Seaborn and Matplotlib to create some count plots which could provide some insight to the data and used Pandas and Numpy to obtain some descriptive statistics. I then fitted and ran some linear and logistic regression models using Seaborn and Stats Model.

**VARIABLES**

The initial dataset consisted of 718 columns (variables) and 12051 rows (cases). After selecting the required variables needed for the analysis, the dataset used throughout consisted of 13 columns and 12051 rows. Details of the variable names, a short description and their datatypes can be found in Table 1 below.

Table 1 – Description of variables and their datatypes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable Name** | **Description** | **Datatype** | **Converted to datatype** |
| *pupilwt* | Sample weights | float64 | integer |
| *sex* | Male/Female | float64 | Integer |
| *ddwbscore* | Wellbeing Score (0-20) | float64 | Integer |
| *ddwbcat* | Wellbeing Category (Low/Not low wellbeing) | float64 | Integer |
| *dgtdcan* | Ever tried cannabis | float64 | Integer |
| *dgtdamp* | Ever tried amphetamines | float64 | Integer |
| *dgtdlsd* | Ever tried LSD | float64 | Integer |
| *dgtdecs* | Ever tried ecstasy | float64 | integer |
| *dgtdket* | Ever tried ketamine | float64 | Integer |
| *dgtdnox* | Ever tried nitrous oxide | float64 | Integer |
| *dgtdleg* | Ever tried legal highs | float64 | Integer |
| *dgtdany* | Ever tried any drugs | float64 | integer |

**ISSUES**

I was unable to obtain access to the dataset that I originally wanted to use for the analysis, which was the Crime Survey for England and Wales. The reason for this was that the dataset is sensitive and therefore a fairly lengthy approval process is required.

I found some problems when writing the code, first of all when trying to run descriptive statistics on the data. I figured out that this was because my datatypes were all float and that I had changed missing values to NaN using Numpy, which resulted in error messages when trying to run certain code for descriptives. My solution to this was to change the type of all the data to integer and this fixed the problem.

I struggled when running descriptive statistics on all of the variables using a for loop as Pandas was only bringing up some of the results then cutting the rest off. I consulted stack overflow and found that I could change the print options in Pandas in order to show more results.

An issue was also reached when conducting the regression analyses using Stats Model. I noticed that only the X variables were showing up in the summary, and that there was no constant. This meant that the regression was completely incorrect. Again, I looked online at stack overflow and found a way to add in the constant manually also using Stats Models and this fixed the regression problem.

**USAGE NOTES**

Please note the code for this assignment has been written on a Mac, therefore the directory links will differ from those on a Windows. In order to make the code work on a windows computer, please ensure that the file ‘sdd\_archive.tab’ has been downloaded and stored on the Desktop before right clicking then finding ‘Copy as path’. You can then amend the code so that it looks something like below:

﻿df = pd.read\_table(“C:\Users\YourName\Desktop\sdd\_archive.tab', low\_memory=False)

Word Count – 795

**REFERENCES**

NHS Digital (2016). *Smoking, drinking and drug use among young people.* Available at: <http://digital.nhs.uk/catalogue/PUB30132> (Accessed: 17 November 2018).

Stack Overflow (various sources). Available at: <https://stackoverflow.com/> (Last Accessed: 4 January 2019).

Further references are written within code comments, e.g. for the use of Pandas, Numpy, Stat Models, Seaborn, Matplotlib

**APPENDIX 1 - CODE**

All code written is my own, based on documentation as listed within the comments in my code and learning from questions and answers available on stackoverflow.com. My code is open source and can be found at <https://github.com/lkelly36/GEOG5995Assignment2>.

﻿

|  |
| --- |
| "" |
|  | =================================================================================================================================== |
|  |  |
|  | Assignment 2 |
|  | GEOG5995M Programming for Social Scientists |
|  | University of Leeds student ID number: 201282995 |
|  |  |
|  | This assignment looks to build a programme that will clean data, produce |
|  | some descriptive statistics and analyse the data using regression models. |
|  |  |
|  | The data set used for this assignment was the Smoking, Drinking and Drug Use |
|  | Among Young People (2016), which can be obtained from |
|  | https://beta.ukdataservice.ac.uk/datacatalogue/studies/study?id=8320 |
|  |  |
|  | =================================================================================================================================== |
|  |  |
|  | """ |
|  |  |
|  | """ |
|  | Import libraries and data set using Pandas to convert .tab file |
|  | Documentation: https://www.pandas.pydata.org |
|  | """ |
|  | # Import required libraries |
|  |  |
|  | import pandas as pd |
|  | import numpy as np |
|  | import matplotlib.pyplot as plt |
|  | import seaborn as sns |
|  | import statsmodels.api as sm |
|  | from statsmodels.formula.api import ols |
|  |  |
|  | # Read in the data set |
|  |  |
|  | df = pd.read\_table('~/Desktop/Data/sdd\_archive.tab', low\_memory=False) |
|  |  |
|  | """ |
|  | Cleaning the data using Numpy (https://www.numpy.org) and Pandas |
|  | Guides obtained from http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy |
|  | and https://realpython.com/python-data-cleaning-numpy-pandas/ |
|  | and https://machinelearningmastery.com/handle-missing-data-python/ |
|  | """ |
|  |  |
|  | # Select required variables and assign to df1 |
|  | df1 = df.loc[:,('pupilwt', 'sex', 'ddwbscore', 'ddwbcat', 'dgtdcan', 'dgtdamp','dgtdlsd','dgtdecs', 'dgtdcok', |
|  | 'dgtdket', 'dgtdnox', 'dgtdleg', 'ddgany')] |
|  |  |
|  | # Create functions for cleaning missing values |
|  |  |
|  | def CleanData(df1): |
|  | nan\_values = [-1,-8,-9] # These variables have values missing at -1,-8,-9 |
|  | df1.sex.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdcan.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdamp.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdlsd.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdecs.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdcok.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdket.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdnox.replace(nan\_values, np.nan, inplace=True) |
|  | df1.dgtdleg.replace(nan\_values, np.nan, inplace=True) |
|  | df1.ddgany.replace(nan\_values, np.nan, inplace=True) |
|  |  |
|  | def CleanWell(df1): |
|  | nan\_values = [-8,-9,-98] # These variables have values missing at -8,-9,-98 |
|  | df1.ddwbscore.replace(nan\_values, np.nan, inplace=True) |
|  | df1.ddwbcat.replace(nan\_values, np.nan, inplace=True) |
|  |  |
|  | # Run functions |
|  | CleanData(df1) |
|  | CleanWell(df1) |
|  |  |
|  | # Change NaNs to average mean |
|  | df1 = df1.fillna(df1.mean()) |
|  | df1.head() |
|  | # Check datatype |
|  | df1.info() |
|  | # Change floats to int |
|  | df1 = df1.astype(int) |
|  | # Check data |
|  | df1.head() |
|  |  |
|  | # Define binary sex, wellbeing and drug variables as 0 and 1 |
|  |  |
|  | def CleanBin(df1): |
|  | # Replace sex variables 1=male, 0=female |
|  | df1.sex.replace(2.0, 0, inplace=True) |
|  | # Replace ever tried any drug 1=yes, 2=no |
|  | df1.ddgany.replace(2.0, 0, inplace=True) |
|  | # Replace wellbeing category variable |
|  | df1.ddwbcat.replace(2.0, 0, inplace=True) |
|  | # Replace drug variables 1=yes, 2=no |
|  | df1.dgtdcan.replace(2.0, 0, inplace=True) |
|  | df1.dgtdamp.replace(2.0, 0, inplace=True) |
|  | df1.dgtdlsd.replace(2.0, 0, inplace=True) |
|  | df1.dgtdecs.replace(2.0, 0, inplace=True) |
|  | df1.dgtdcok.replace(2.0, 0, inplace=True) |
|  | df1.dgtdket.replace(2.0, 0, inplace=True) |
|  | df1.dgtdnox.replace(2.0, 0, inplace=True) |
|  | df1.dgtdleg.replace(2.0, 0, inplace=True) |
|  |  |
|  | # Run function and check data |
|  | CleanBin(df1) |
|  | df1.head() |
|  |  |
|  | """ |
|  | Descriptive Statistics and Data Visualisation using Seaborn |
|  | Documentation: https://www.seaborn.pydata.org |
|  | """ |
|  |  |
|  | # Calculate some descriptive statistics for outcome variable |
|  | wbmean = np.mean(df1.ddwbscore) # mean wellbeing score |
|  | wbvar = np.var(df1.ddwbscore) # variance |
|  | print(wbmean) |
|  | print(wbvar) |
|  |  |
|  | # Produce descriptives for all drug use data |
|  | pd.set\_option('display.max\_columns', 20) # change pandas print options to show whole output |
|  |  |
|  | desc\_list = [df1.describe()] + [df1.groupby([c])[df1.columns[0]].count() |
|  | for c in df1.columns if df1[c].dtype == 'object'] |
|  | for i in desc\_list: |
|  | print(i) |
|  | print() |
|  |  |
|  | # Produce crosstab for drug use and wellbeing category |
|  | pd.crosstab(df1['ddgany'], df1['ddwbcat']) |
|  | # Produce crosstab for use of each individual drug and wellbeing category |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdcan']) |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdamp']) |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdlsd']) |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdecs']) |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdcok']) |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdket']) |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdnox']) |
|  | pd.crosstab(df1['ddwbcat'], df1['dgtdleg']) |
|  |  |
|  | # Count plot of drug use split by gender |
|  | ax = sns.countplot(x='ddgany', hue='sex', data=df1) |
|  | # Change handle labels |
|  | handles = ax.get\_legend\_handles\_labels()[0] |
|  | ax.legend(handles, ['Female', 'Male'], title='Gender') |
|  | #Set labels, save and show plot |
|  | plt.title('Number of pupils with reported drug use where 0=no and 1=yes') |
|  | plt.xlabel('Ever used any drugs') |
|  | plt.ylabel('Count') |
|  | plt.savefig('../count\_drug.jpg',format='jpg') |
|  | plt.figure() |
|  |  |
|  | # Count plot of wellbeing scores split by drug use |
|  | ax1 = sns.countplot(x='ddwbscore', hue='ddgany', data=df1) |
|  | # Change handle labels |
|  | handles = ax.get\_legend\_handles\_labels()[0] |
|  | ax1.legend(handles, ['No', 'Yes'], title='Ever Used Drugs?') |
|  | # Add labels and save |
|  | plt.title('Relationship between drug use and wellbeing') |
|  | plt.xlabel('Wellbeing Scores') |
|  | plt.ylabel('Count') |
|  | plt.savefig('../wb\_drug.jpg',format='jpg') |
|  | plt.figure() |
|  |  |
|  | """ |
|  | Linear regression showing drug use and gender as predictors of wellbeing using seaborn. |
|  | """ |
|  |  |
|  | # Print regression model |
|  | model = ols("ddwbscore ~ ddgany + sex", df1, weight=df1.pupilwt).fit() |
|  | print(model.summary()) |
|  |  |
|  | """ |
|  | Linear regression showing use of different drugs as predictors of wellbeing scores using stats models. |
|  | Documentation: https://www.statsmodels.org/dev/generated/statsmodels.regression.linear\_model.OLS.html |
|  |  |
|  | """ |
|  |  |
|  | # Create X variable and control for sex |
|  | X = [df1.sex, df1.dgtdcan, df1.dgtdamp, df1.dgtdlsd, df1.dgtdecs, df1.dgtdcok, df1.dgtdket, df1.dgtdnox, df1.dgtdleg] |
|  | X = np.array(X) |
|  | X = X.T |
|  | X = sm.add\_constant(X) # Include constant in regression |
|  | # Create response variable - wellbeing scores |
|  | y = df1.ddwbscore |
|  |  |
|  | # Run linear regression model and print summary |
|  | linear\_model=sm.OLS(y,X, weight=df1.pupilwt) |
|  | result\_lin=linear\_model.fit() |
|  | print(result\_lin.summary2()) |
|  |  |
|  | # Remove insignificant variables from model |
|  | X = [df1.sex, df1.dgtdcan, df1.dgtdamp, df1.dgtdlsd, df1.dgtdecs, df1.dgtdcok, df1.dgtdket, df1.dgtdleg] |
|  | X = np.array(X) |
|  | X = X.T |
|  | X = sm.add\_constant(X) # Include constant |
|  |  |
|  | # Run second linear regression model without insignificant variables |
|  | linear\_model2=sm.OLS(y,X, weight=df1.pupilwt) |
|  | result\_lin2=linear\_model2.fit() |
|  | print(result\_lin2.summary2()) # Print summary |
|  |  |
|  | """ |
|  | Logistic regression model showing use of different drugs as predictors of wellbeing category using stats models. |
|  | Documentation: https://www.statsmodels.org/dev/generated/statsmodels.discrete.discrete\_model.Logit.html |
|  | """ |
|  |  |
|  | # Define X variable |
|  | X = [df1.sex, df1.dgtdcan, df1.dgtdamp, df1.dgtdlsd, df1.dgtdecs, df1.dgtdcok, df1.dgtdket, df1.dgtdnox, df1.dgtdleg] |
|  | X = np.array(X) |
|  | X = X.T |
|  | X = sm.add\_constant(X) # Include constant in regression |
|  | # y variable |
|  | y = df1.ddwbcat |
|  |  |
|  | # Run logistic regression model |
|  | logit\_model=sm.Logit(y,X, weight=df1.pupilwt) |
|  | result=logit\_model.fit() |
|  | print(result.summary2()) |
|  |  |
|  | # Redefine X variable, removing insignificant variables from previous model and controlling for sex |
|  | X = [df1.sex, df1.dgtdlsd, df1.dgtdecs, df1.dgtdcok, df1.dgtdket, df1.dgtdnox, df1.dgtdleg] |
|  | X = np.array(X) |
|  | X = X.T |
|  | X = sm.add\_constant(X) # Include constant in regression |
|  |  |
|  | # Run logistic regression model again |
|  | logit\_model2=sm.Logit(y,X, weight=df1.pupilwt) |
|  | result=logit\_model2.fit() |
|  | print(result.summary2()) |
|  |  |
|  | # Define X without insignificant variables for final time |
|  | X = [df1.sex, df1.dgtdlsd, df1.dgtdcok, df1.dgtdket, df1.dgtdnox] |
|  | X = np.array(X) |
|  | X = X.T |
|  | X = sm.add\_constant(X) # Include constant in regression |
|  |  |
|  | # Final fit and run logit model |
|  | logit\_model2=sm.Logit(y,X, weight=df1.pupilwt) |
|  | result=logit\_model2.fit() |
|  | print(result.summary2()) |