

Drug Addiction Causes in US



BDAS - Iteration 4

Shimali Fernando -1805476

INFOSYS 722 - Data Mining and Big Data

Table of Contents

| | |
|---|----|
| 1 Business and/ or Situation Understanding | 4 |
| 1.1 Identifying the objectives of the business and or/ situation | 4 |
| 1.2 Assess the situation | 4 |
| 1.3 Determine data mining objectives..... | 4 |
| 1.4 Produce a Project Plan | 5 |
| 2 Data Understanding..... | 5 |
| 2.1 Collect initial data | 5 |
| 2.2 Describe the data | 6 |
| 2.3 Explore the data..... | 8 |
| 2.4 Verify the data Quality..... | 12 |
| 3 Data Preparation..... | 14 |
| 3.1 Select the data | 14 |
| 3.2 Clean the data | 15 |
| 3.3 Construct the data | 16 |
| 3.4 Integrate the data | 17 |
| 3.5 Format the data | 17 |
| 4 Data Transformation..... | 17 |
| 4.1 Reduce the data | 17 |
| 4.2 Project the data..... | 19 |
| 5 Data-mining Method Selection | 19 |
| 5.1 Match and discuss the objective of data mining (1.1) to data mining methods | 19 |
| 5.2 Select the appropriate data-mining method(s) based on discussion | 20 |
| 6 Data-mining Algorithm Selection..... | 20 |
| 6.1 Conduct exploratory analysis and discuss | 20 |
| 6.2 Select data-mining algorithm based on discussion..... | 21 |
| 6.3 Build/Select appropriate model(s) and choose relevant parameters | 21 |
| 7 Data Mining..... | 24 |
| 7.1 Create and justify test designs | 24 |
| 7.2 Conduct data mining..... | 25 |
| 7.3 Search for Patterns | 25 |
| 8 Interpretation..... | 27 |
| 8.1 Study and discuss the mined patterns..... | 27 |

8.2 Visualize the data, results, models and patterns..... 28

8.3 Interpret the results, models and patterns 29

8.4 Asses and evaluate results 30

8.5 Iterate step 1-7..... 30

1 Business and/ or Situation Understanding

1.1 Identifying the objectives of the business and or/ situation

Drug addiction has become a serious problem across nations. The recent stats indicate that 63,632 Americans have died due to accidental drug overdose and there is 21.5% increase in the total numbers in contrast to year 2016 (Ducharme,2018)¹. However, the real statistical observations behind these deaths are yet to be discovered. It is assumed by many publics that demographic variables like age, gender, and education level are some of the key factors that contribute to this cause. However, during this project the main objective is to support the US government to find out if Heroin is the primary drug that is causing these majority of the deaths. By doing so, this project also aims to address the Sustainable Development Goal like Good Health and Well-being

1.2 Assess the situation

Existing Research

In this context previous research has been carried out to find the cause of accidental drug related deaths. But it is not focusing on the primary drug that is causing these deaths. The immediate cause in these projects has been focused on the reason for the death (i.e. Multiple Drug Toxicity) and predicting future possibilities. Hence, these researches are not looking at variables that are prone for drug related deaths.

Proposed Solution

The proposed solution is looking at a dataset to predict if the victims are prone for accidental drug related deaths if they are using heroin as their primary drug.

Data Sets

Accidental drug related deaths within 2012-2017. (data.gov)

Tools

The current tools that is available to assess the situation can be listed as follows.

1. AWS Educate student account
2. Jupyter Notebook, pyspark
3. Microsoft Excel 2016

1.3 Determine data mining objectives

The data set involved in this project is looking at data, that is causing for accidental drug related deaths. Thereby the datamining objective of this project is to;

- predict if Heroin is the primary drug that is causing the accidental drug related deaths in US

¹ Retrieved from <https://www.cdc.gov/media/releases/2018/p0329-drug-overdose-deaths.html>

1.4 Produce a Project Plan

| Scope | Time | Risk |
|-------------------------------------|--------|---|
| 1. Business Situation Understanding | 1 day | Incorrect assumptions, poor situation understanding, unrealistic project plan |
| 2. Data Understanding | 2 days | Insufficient data, incorrect data, poor quality data |
| 3. Data Preparation | 3 days | Poor quality data, data integration concerns |
| 4. Data Transformation | 4 days | Losing important data and variables |
| 5. Data-mining method selection | 3 days | Wrong method selection |
| 6. Data-mining algorithm selection | 3 days | Lack of knowledge in data mining algorithms, wrong algorithm selection |
| 7. Data Mining | 4 days | Not finding related patterns |
| 8. Interpretation | 2 days | Unable to give the correct interpretation of findings |

2 Data Understanding

2.1 Collect initial data

Data Sets

Accidental drug related deaths (data. gov)

The initial data was collected via the United States data.gov website. However, the process of finding the appropriate data set to suite my project was a key challenge during the first stage as I was not familiar with data repositories. First, I found a data set with accidental drug related deaths in different County but, the next data set I found was also similar in attributes as the data set was looking at admissions to hospitals in each county due to drug related incidents. Since, both data sets were looking at similar attributes, I only considered the first data set accidental drug related deaths. The data is limited to the Connecticut State in US. This state is in southern US which has 9 different counties. As I'm only having records of one single state, I am assuming that this state will represent the US as a whole. At a glance attributes like Age, Gender, Race, Type of drug caused the death looks interesting for further assessments. The data set I chose have 3227 records and I'm assuming this data set is fair enough to draw generalizable conclusions and at the same time make accurate predictions. There are 33 columns and most of these columns consist of noise, which needs further cleaning before the data mining process.

2.2 Describe the data

Accidental drug related deaths

Description

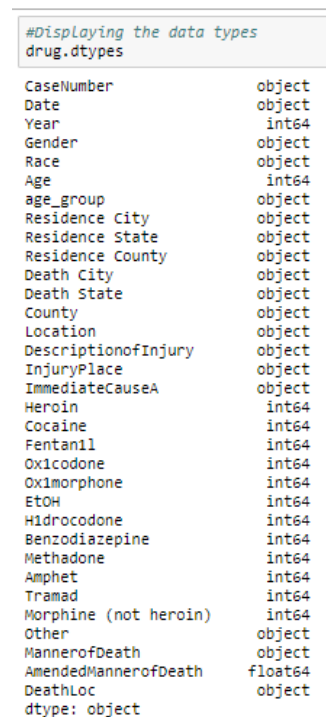
The original data set was downloaded in a “.csv” file, via the US Data.gov. website. The dataset is looking at the gender, ethnicity, age, type of drug used and county within the year 2012 to 2017. The observations in the data set showed that year 2016 had approximately 920 deaths reported whereas 2017 only showed 186 records. Hence it was clear, that the data set was not representing the full records of year 2017. As a result, the 2017 records were deleted via filter option in excel. The quantity of the new dataset is as follows.

Quantity

- Number of records - 3041
- Attributes- 33

Data type

The data set is looking at numerical (Integer) and categorical (object) data types.



```
#Displaying the data types
drug.dtypes
```

| | |
|-----------------------|---------|
| CaseNumber | object |
| Date | object |
| Year | int64 |
| Gender | object |
| Race | object |
| Age | int64 |
| age_group | object |
| Residence City | object |
| Residence State | object |
| Residence County | object |
| Death City | object |
| Death State | object |
| County | object |
| Location | object |
| DescriptionofInjury | object |
| InjuryPlace | object |
| ImmediateCauseA | object |
| Heroin | int64 |
| Cocaine | int64 |
| Fentanil | int64 |
| Oxycodone | int64 |
| Oxymorphone | int64 |
| EtOH | int64 |
| Hydrocodone | int64 |
| Benzodiazepine | int64 |
| Methadone | int64 |
| Amphet | int64 |
| Tramad | int64 |
| Morphine (not heroin) | int64 |
| Other | object |
| MannerofDeath | object |
| AmendedMannerofDeath | float64 |
| DeathLoc | object |
| dtype: | object |

Figure 1: Data Types

The key attributes can be described as follows.

attribute name: Year

The column Year is looking at records from year 2012 to 2016. At a glance it is evident that over the years, number of records increase from 355 to 917.

```
In [81]: #Checking each Year value counts  
drug['Year'].value_counts()
```

```
Out[81]: 2016    917  
        2015    723  
        2014    556  
        2013    490  
        2012    355  
        Name: Year, dtype: int64
```

attribute name: Gender

The column Gender is looking at the death rates between males and females. In general, the male death count is comparatively higher than female death count.

```
In [83]: #Checking each Gender value counts  
drug['Gender'].value_counts()
```

```
Out[83]: Male      2207  
        Female     823  
        Name: Gender, dtype: int64
```

attribute name: Race

The column Race is looking at different types of ethnicities involved in the accidental drug related deaths. At glance 80 % of the recorded deaths are representing the ethnicity White.

```
In [84]: #Checking each Race value counts  
drug['Race'].value_counts()
```

```
Out[84]: White      2448  
        Hispanic    327  
        Black      250  
        Asian Other    11  
        Asian Indian    3  
        Chinese        2  
        Name: Race, dtype: int64
```

attribute name: age_group

The age group is an additional column that was added for better representation of the Age distribution. Hence, it is clear that MiddleAged category has the highest number of recorded deaths due to drug addiction.

```
In [82]: #Checking each age group value counts
drug['age_group'].value_counts()

Out[82]: MiddleAged    1472
         Seniors      731
         Youth       597
         Elderly      201
         Teenage      40
         Name: age_group, dtype: int64
```

attribute name: County

The column County is looking at 9 different counties in Connecticut State. The lowest representation is from Haddam by number deaths at 41 and the highest representation is from Hartford by number of deaths at 900.

```
In [80]: #Checking each county value counts
drug['County'].value_counts()

Out[80]: Hartford      900
         New Haven     849
         Fairfield     491
         New London    311
         Litchfield    175
         Windham       96
         Middletown    91
         Tolland       87
         Haddam        41
         Name: County, dtype: int64
```

attribute name: Heroin

The column Heroin denotes the number deaths that caused due to heroin. This is represented in Boolean notation. It is also evident that the distribution of 0's and 1's is balanced.

```
In [108]: #Checking each Heroin value counts
drug['Heroin'].value_counts()

Out[108]: 1    1669
         0    1361
         Name: Heroin, dtype: int64
```

Out of the given list, attributes like year, gender, race, age_group, county and heroin have a better weight in looking at the prediction to see if heroin is the primary drug that us causing the accidental drug related deaths and what other demographic variable does support this cause.

2.3 Explore the data

In this iteration data exploration is carried out using “pandas” *matplotlib.pyplot* library and by using data structures. As a first step I have looked at different attributes that would bring value in the process of understanding different demographic variables that would cause the accidental drug related deaths in US.

Relationship between ‘Gender’ and ‘age_group’

In order to find the relationship between the two groups, first the two attributes are grouped and the index label is pivoted for better visualization. During the data describing process it was

identified that male representation of accidental drug related deaths were higher than female representation. But however, adding the age group category, it further emphasizes that among the males, middle aged, age group category is more prone for these types of deaths.

```
# Pivoting index labels - Gender and age_group
drug.groupby('Gender').age_group.value_counts(normalize=True).unstack()
```

| age_group | Elderly | MiddleAged | Seniors | Teenage | Youth |
|-----------|----------|------------|----------|----------|----------|
| Gender | | | | | |
| Female | 0.053463 | 0.467801 | 0.279465 | 0.018226 | 0.181045 |
| Male | 0.071137 | 0.491618 | 0.225646 | 0.010874 | 0.200725 |

Relationship between 'Heroin' and 'County'

Similarly, out of the primary drugs that caused for accidental drug related deaths, heroin had a higher representation compared to the other drugs. But looking at this table it is also clear that highest heroin related deaths are been reported from 'Hartford' and the lowest heroin related deaths are reported in 'Haddam'

```
# Pivoting a level of index labels - Heroin, County
drug.groupby('Heroin').County.value_counts(normalize=True).unstack()
```

| County | Fairfield | Haddam | Hartford | Litchfield | Middletown | New Haven | New London | Tolland | Windham |
|--------|-----------|----------|----------|------------|------------|-----------|------------|----------|----------|
| Heroin | | | | | | | | | |
| 0 | 0.174137 | 0.013226 | 0.267450 | 0.049963 | 0.029390 | 0.310066 | 0.093314 | 0.033799 | 0.028655 |
| 1 | 0.151588 | 0.013182 | 0.319952 | 0.063511 | 0.029359 | 0.254044 | 0.109646 | 0.024566 | 0.034152 |

Relationship between 'County' and 'Year'

Looking at the relationship between County and Year it is evident that 2016 had the highest reported deaths due to accidental drug usage and Haddam, Fairfield, Hartford are the top three counties in drug related deaths. Even though Fairfield was in the top three counties in year 2016, in the year 2012, it had the lowest reported deaths in contrast to all other counties.

```
# Pivoting a level of index labels - County, Year
drug.groupby('County').Year.value_counts(normalize=True).unstack()
```

| Year | 2012 | 2013 | 2014 | 2015 | 2016 |
|------------|----------|----------|----------|----------|----------|
| County | | | | | |
| Fairfield | 0.100000 | 0.185714 | 0.183673 | 0.204082 | 0.326531 |
| Haddam | 0.150000 | 0.075000 | 0.125000 | 0.225000 | 0.425000 |
| Hartford | 0.114699 | 0.149220 | 0.181514 | 0.237194 | 0.317372 |
| Litchfield | 0.109195 | 0.218391 | 0.218391 | 0.258621 | 0.195402 |
| Middletown | 0.134831 | 0.157303 | 0.202247 | 0.314607 | 0.191011 |
| New Haven | 0.114657 | 0.160757 | 0.166667 | 0.248227 | 0.309693 |
| New London | 0.129032 | 0.183871 | 0.183871 | 0.200000 | 0.303226 |
| Tolland | 0.126437 | 0.114943 | 0.264368 | 0.218391 | 0.275862 |
| Windham | 0.145833 | 0.072917 | 0.197917 | 0.343750 | 0.239583 |

Relationship between 'Age', 'age_group' and 'Heroin'

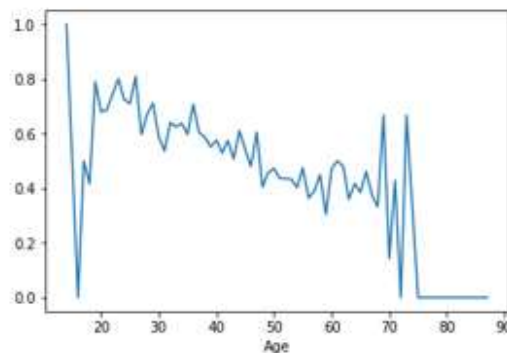
The age column is further categorized into age group column for better representation. However, the of Heroin mean and the age representation can also be looked at as follows.

```
#grouping by age group and Heroin mean
drug.groupby('age_group').Heroin.mean()
```

```
age_group
Elderly      0.412935
MiddleAged   0.564626
Seniors      0.423077
Teenage      0.615385
Youth        0.716216
Name: Heroin, dtype: float64
```

```
#Plotting relationship between age and heroin mean
drug.groupby('Age').Heroin.mean().plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3c409abef0>
```



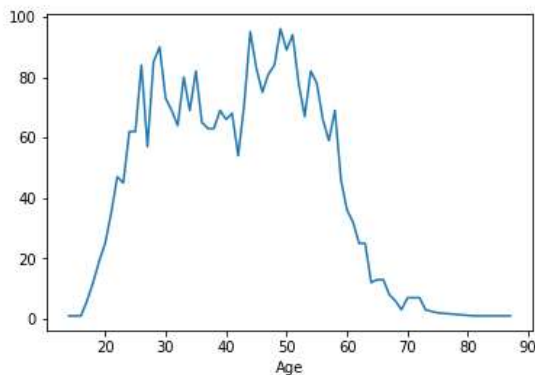
The plotted graph in 'heroin mean' shows that there is spike among the youth for heroin usage, but however as the years pass by the interest drops and again the interest spikes in late 60's. Thereafter, either the interest of drug addiction completely drops or there are no recorded incidents of accidental deaths due to drugs.

The above graph can be further elaborated looking at the relationship between Age and Heroin Count. Once looking at the heroin count it is visible that as teenagers many starts to get addicted for drugs. After the age of 40 some drop and moves on, but immediately after that some get extremely addicted to drugs and hence the number of death count has gone

comparatively high. Then gradually the interest has gone down as the number of reported deaths has dropped significantly.

```
# Plotting year and heroin mean
drug.groupby('Age').Heroin.count().plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3c4084d048>
```



Relationship between 'Year' and 'Heroin'

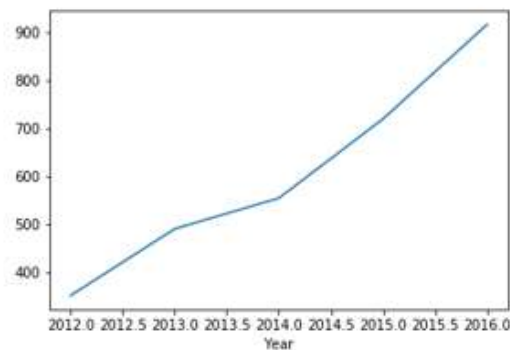
Similarly, the continuous growth in heroin usage over the years can be visualized as follows.

```
#grouping year and heroin count
drug.groupby('Year').Heroin.count()
```

```
Year
2012    351
2013    490
2014    554
2015    719
2016    916
Name: Heroin, dtype: int64
```

```
#Plotting year and heroin count
drug.groupby('Year').Heroin.count().plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3c408c86d8>
```



Looking at the above data exploration it is clearly evident that over the years the Heroin usage has increased significantly. The main goal of this data mining project is to find out if Heroin is the primary drug that is causing the death. But however, the data exploration also generalizes that other demographic variables like age-group, gender, race and county can also be reasons for these deaths.

As I'm only considering the attributes that is relevant to my data mining goal I'm dropping certain attributes. But some of these attributes like 'InjuryPlace' and 'Date' can be key attributes for future study areas, as it can give insights for further evaluations. E.g. What time of the month does number of accidental drug related deaths increase, how long does it take to

know an accidental drug related death? Does it have a relationship to the place of Injury? The data exploration also brought my attention in general hypotheses like “white men are more prone for accidental drug related deaths”. All of these data exploration insights can be used in future studies.

2.4 Verify the data Quality

Generally, when we collect data there can be many missing values, typing errors and inconsistencies. The data set I downloaded also had similar data quality concerns. The accidental drug related deaths data set initially had 33 columns. Out of which 13 columns did not add value to my data mining objective. For example, attributes like ‘InjuryPlace’, ‘DeathLoc’, ‘ImmediateCauseA’. These attributes needed to be cleaned before data mining process. Also, these 33 columns had lot of missing values, that needed careful attention. In this iteration, the missing values are handled through pandas. A general report of missing values can be viewed as follows.

| #identifying missing values drug.isnull().sum() | |
|--|------|
| CaseNumber | 0 |
| Date | 0 |
| Year | 0 |
| Gender | 11 |
| Race | 0 |
| Age | 0 |
| age_group | 0 |
| Residence City | 90 |
| Residence State | 1901 |
| Residence County | 0 |
| Death City | 0 |
| Death State | 1863 |
| County | 0 |
| Location | 18 |
| DescriptionofInjury | 2579 |
| InjuryPlace | 72 |
| ImmediateCauseA | 0 |
| Heroin | 0 |
| Cocaine | 0 |
| Other | 0 |

The table above shows that attributes like Gender, Residence City, Residence State, Death State, Description of Injury and Injury Place are some of the missing values that needs to be looked at.

The next data quality concern was the primary drug that caused the death. Initially the data set was only showing the value “Y” in certain fields and the remaining cells were empty fields. In this situation, I assumed, value “Y” was the primary drug that is causing the death, and empty fields is when it is not the primary drug for the cause of the death. The below image shows how preview of the initial dataset.

| Heroin | Cocaine | Fentanyl | Oxycodon | Oxymorpl |
|--------|---------|----------|----------|----------|
| | | Y | | |
| | | Y | | |
| | Y | | | |
| | | | Y | Y |
| Y | | | | |
| Y | | | | |
| Y | | | | |
| Y | Y | | | |
| | | | | |
| | Y | | | |
| Y | | | | |

Figure: Image of the initial Excel file

The number of records the data set was holding was 3227. But, out of which 186 records were deaths in year 2017. But however, these records clearly did not represent the entire year as it was evident from the date column, the records were only up till two quarters of the year. So as result, I had to only consider the records from year 2012-2016. Then the number of records dropped down to 3041. However, I also assumed that my dataset is fair enough to draw generalizable conclusions and at the same time make accurate predictions.

Also, the dataset was only representing one single State in southern US, named Connecticut. An overview of the Connecticut State is as follows.



Even though the dataset is limited to a single State, I am assuming that this state will represent the United State as whole.

3 Data Preparation

3.1 Select the data

In the data selection process, it is import to differentiate the attributes and records that is relevant to the datamining goal. In this project, I'm looking if Heroin is the main cause for the death and if any other demographic variables are supporting directly on the accidental drug related deaths. As explained in 'verifying data quality', the records of 2017 need to be discarded. Hence as a first step, the 2017 records are filtered and deleted using the Excel filter option. This function was not carried out in the python environment as it was technically difficult for me to filter all 186 records from the dataset and to delete them separately. After the filter the dataset was having the following records and attributes.

```
#Identifying records,attributes
drug.shape
```

```
(3041, 33)
```

The next step was to identify and select most appropriate columns that would help in the data mining objective. At a glance from the 'data exploration' step it was evident that certain columns can be discarded as it made no value. Specially the columns that had comparatively high missing values. The following image shows the data type of all 33 attributes and the number of missing values in each attribute.

```
#Displaying the data types
drug.dtypes
```

| | |
|-----------------------|---------|
| CaseNumber | object |
| Date | object |
| Year | int64 |
| Gender | object |
| Race | object |
| Age | int64 |
| age_group | object |
| Residence City | object |
| Residence State | object |
| Residence County | object |
| Death City | object |
| Death State | object |
| County | object |
| Location | object |
| DescriptionofInjury | object |
| InjuryPlace | object |
| ImmediateCauseA | object |
| Heroin | int64 |
| Cocaine | int64 |
| Fentanil | int64 |
| Oxycodone | int64 |
| Oxymorphone | int64 |
| EtOH | int64 |
| Hydrocodone | int64 |
| Benzodiazepine | int64 |
| Methadone | int64 |
| Amphet | int64 |
| Tramad | int64 |
| Morphine (not heroin) | int64 |
| Other | object |
| MannerofDeath | object |
| AmendedMannerofDeath | float64 |
| DeathLoc | object |
| dtype: | object |

```
#identifying missing values
drug.isnull().sum()
```

| | |
|-----------------------|-------|
| CaseNumber | 0 |
| Date | 0 |
| Year | 0 |
| Gender | 11 |
| Race | 0 |
| Age | 0 |
| age_group | 0 |
| Residence City | 90 |
| Residence State | 1901 |
| Residence County | 0 |
| Death City | 0 |
| Death State | 1863 |
| County | 0 |
| Location | 18 |
| DescriptionofInjury | 2579 |
| InjuryPlace | 72 |
| ImmediateCauseA | 0 |
| Heroin | 0 |
| Cocaine | 0 |
| Fentanil | 0 |
| Oxycodone | 0 |
| Oxymorphone | 0 |
| EtOH | 0 |
| Hydrocodone | 0 |
| Benzodiazepine | 0 |
| Methadone | 0 |
| Amphet | 0 |
| Tramad | 0 |
| Morphine (not heroin) | 0 |
| Other | 2742 |
| MannerofDeath | 6 |
| AmendedMannerofDeath | 3041 |
| DeathLoc | 0 |
| dtype: | int64 |

After clear observation, 13 columns were removed. The selected columns can be listed as follows.

| | |
|-----------------------|----|
| CaseNumber | 0 |
| Date | 0 |
| Year | 0 |
| Gender | 11 |
| Race | 0 |
| Age | 0 |
| age_group | 0 |
| County | 0 |
| Heroin | 0 |
| Cocaine | 0 |
| Fentanyl | 0 |
| Oxycodone | 0 |
| Oxymorphone | 0 |
| EtOH | 0 |
| Hydrocodone | 0 |
| Benzodiazepine | 0 |
| Methadone | 0 |
| Amphet | 0 |
| Tramad | 0 |
| Morphine (not heroin) | 0 |

dtype: int64

3.2 Clean the data

Out of the listed dataset attributes, certain attributes like residence state, Death state, Description of injury, Other and Amended Death of Manner had 1000 + number of missing records and, the entire data set was only looking at 3000 + records. Hence the missing 1000+ records in each column was unrealistic and was best option in cleaning the data was to drop the entire column. Similarly, the columns like Residence City, Residence County, Death City, Death State was looking at data that was 80% identical to the general County column. Hence, as one Column is suitable to represent the County of the reported death, the remaining identical columns can also be dropped for better representation of the dataset. Next it is required to see what columns would add value to find if Heroin is the primary drug that is causing the death. At the same time, it appropriate to look into other demographic variables like age, year, County, gender and ethnicity. However, looking at the above discussed points the cleaning process can be started as follows.

Identifying initial records and attributes

```
#Identifying records, attributes
drug.shape
```

```
(3041, 33)
```

Dropping columns discussed above

```
#dropping Columns that is not relevant for data mining objective
drug.drop(['Residence City', 'Residence State', 'Residence County', 'Death City', 'Death State', 'Location', 'DescriptionofInjury',
```

The attributes after dropping

```
# Checking records, attributes after dropping
drug.shape
```

```
(3041, 20)
```

Checking column names

```
drug.columns
```

```
Index(['CaseNumber', 'Date', 'Year', 'Gender', 'Race', 'Age', 'age_group',
      'County', 'Heroin', 'Cocaine', 'Fentan1l', 'Ox1codone', 'Ox1morphone',
      'EtOH', 'Hidrocodone', 'Benzodiazepine', 'Methadone', 'Amphet',
      'Tramad', 'Morphine (not heroin)'],
      dtype='object')
```

Renaming appropriate columns, as certain fields have long or incorrect names.

```
drug.rename(columns={'Morphine (not heroin)': 'Morphine', 'Fentan1l': 'Fentanyl'}, inplace=True)
```

```
drug.columns
```

```
Index(['CaseNumber', 'Date', 'Year', 'Gender', 'Race', 'Age', 'age_group',
      'County', 'Heroin', 'Cocaine', 'Fentanyl', 'Ox1codone', 'Ox1morphone',
      'EtOH', 'Hidrocodone', 'Benzodiazepine', 'Methadone', 'Amphet',
      'Tramad', 'Morphine'],
      dtype='object')
```

3.3 Construct the data

As discussed in point 2.4 ‘verify the quality of data’ the primary drug needed to be constructed as per the assumption - value “Y” was the primary drug that is causing the death, and empty fields are not the primary drug for the cause of the death. Hence this assumption was derived to a Boolean notation of 1 if it’s the primary and else 0 if its not the primary drug. This construction was done using advance find and replace option. The constructed primary drug columns were as follows.

| Heroin | Cocain | Fentar |
|--------|--------|--------|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

The next construct was done for better understanding of the age column. In order to do so, a simple table was created with different age ranges and age categories. The table is as follows.

| AgeRange | AgeCategory |
|----------|-------------|
| 13-19 | Teenage |
| 20-29 | Youth |
| 30-49 | Middle-aged |
| 50-59 | Seniors |
| 60+ | Elderly |

For all the age records in the dataset, the above table age category and range was used to construct the new column “age_group”. The construction was done using Excel conditional formatting and the following formula was used in creating each record.

```
=IF(F2<=19,"Teenage",IF(F2<=29,"Youth",IF(F2<=49,"MiddleAged",IF(F2<=59,"Seniors","Elderly"))))
```

3.4 Integrate the data

Hence, this data set is looking at only one single dataset, in this iteration, there won't be any additional data sources, that will merge with the original dataset.

But however, the dataset needs to integrate with ‘jupyter notebook’ to perform the data representation and visualization. Hence, I have created a folder inside notebook called “Datasets” to store the data. Then I have called the dataset separately for ‘dataframes’ and ‘pandas’.

3.5 Format the data

Even though, this dataset is not integrating with another data source, the construction of primary dug and age_group needs to format for all 3041 records.

4.1 Reduce the data

4 Data Transformation

The data reduction can be done in two different forms namely horizontal reduction and vertical reduction. As discussed in point 2.4 ‘verify the quality of data’ a horizontal reduction was carried out to ensure all 2017 records were discarded. This was done through Excel filter function. The reason for this horizontal data reduction was due to the inconsistency in representing the 2017 as a whole. This was clearly evident from the Data column, where the records only had two quarters of the year.

The next horizontal data reduction was done to perform the missing data in Gender column. This was handled in pandas and in order to do so, the following steps were followed.

```
#Identifying records,attributes  
drug.shape
```

```
(3041, 20)
```

```
#Dropping any value that has a null value... But this is temporary  
drug.dropna(how='any').shape
```

```
(3030, 20)
```

```
# Hence its temporary, the original value records and attributes displays  
drug.shape
```

```
(3041, 20)
```

```
# Removing all missing records of Gender column  
drug.dropna(subset=['Gender'],how='all').shape  
# Applying changes to the dataset  
drug.dropna(inplace=True)
```

```
#New data set records and attributes after the change  
drug.shape
```

```
(3030, 20)
```

Similarly, before starting the machine learning process, in the dataframes the same step was repeated in 'pyspark' to remove the missing values in Gender column.

```
# dropping missing values and assigning a new dataframe  
print("Total data points:", dfN.count())  
df=dfN.dropna()  
print("Total data points:", df.count())
```

```
Total data points: 3041
```

```
Total data points: 3030
```

The main reason for me to carry out this horizontal data reduction was, the original data set had 3041 records, and out of which it had only 11 missing records after selecting the most appropriate columns. If I dropped these 11 records, it would only bring a 0.36% lost for entire dataset. Hence, this was much easier way of handling the missing records as it did not seriously affect my data set. At the same time, no further horizontal reductions were carried out as the original data set was only holding 3030 records after cleaning the missing values.

Again, as discussed in point 3.1 and 3.2, the vertical reduction was carried out in reducing the number of columns in the dataset. This was done through pandas. The reason to drop these columns was discussed in point 3.2 and the main causes were record duplication, 1000+ missing records and not adding value to the data mining objective, "if heroin is the main cause for the accidental drug related deaths" The vertical data reduction through pandas is as follows.

```
#dropping Columns that is not relevant for data mining objective
drug.drop(['Residence City', 'Residence State', 'Residence County', 'Death City', 'Death State', 'Location', 'DescriptionofInjury'],
```

4.2 Project the data

During the data projection, the reduced data set in point 4.1 was taken into consideration. Out of the reduced dataset, further projection was carried out by using a select statement to ensure the most relevant demographic variables are selected. Hence, the projection of attributes came down from 33 to 7, during the process 26 attributes were discarded. This was done to ensure that data mining objective; if heroin is the primary drug that is the cause for the accidental drug related deaths and to see if any other key demographic variables like age, gender, race, etc. has a relationship to these deaths.

```
#selecting the relevant features.
```

```
df = df.select('Year', 'Gender', 'Race', 'Age', 'age_group', 'County', 'Heroin')
df.printSchema()
```

```
root
```

```
|-- Year: integer (nullable = true)
|-- Gender: string (nullable = true)
|-- Race: string (nullable = true)
|-- Age: integer (nullable = true)
|-- age_group: string (nullable = true)
|-- County: string (nullable = true)
|-- Heroin: integer (nullable = true)
```

5 Data-mining Method Selection

5.1 Match and discuss the objective of data mining (1.1) to data mining methods

The main objective of this project is to predict if the victims are prone to accidental drug related deaths if they are using Heroin as the primary drug. The prediction is looking at a specific class called “Heroin”. The prediction value would be “1” if the primary drug is Heroin and value “0” for Heroin not being the primary drug. Hence, I’m trying to classify the predicted class as 0 or 1, the data mining method that is supporting the goal is classification. As a result, in this iteration, I would be looking at two methods namely ‘Logistic Regression’ and ‘Naïve Bayes’ to fulfill the data mining objective.

Logistic Regression

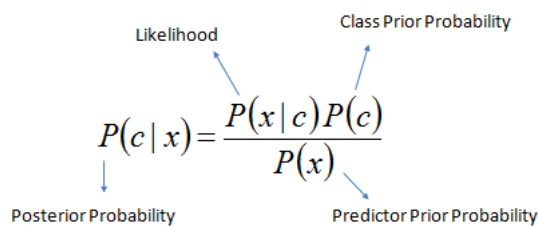
The logistic regression is an algorithm that is most commonly used for classification tasks. It can predict the probability of an outcome that has two values and it supports both numerical and categorical predicted class. In this project the predicted class “Heroin” is representing a binary value and hence its numerical. The other key reason to select logistic regression is that it produces a logistic curve, which is limited to values between 0 and 1. Even though logistic regression is similar to a linear regression, the target variable is constructed using the natural

algorithm of the odds rather than the probability. In this scenario, I would be using logistic regression over linear regression as it suits the data mining objective

Naïve Bayes

The Naïve Bayes is a classification algorithm that is based on the Bayes' theorem. This incorporates strong independent assumptions that often do not have an impact on reality. Hence, it is considered as naïve. The algorithm assumes that the effect of the value of a predictor(x) on a given class (c), which is independent of the other predictors. In this project the predictor is the cause of the death and the predicting class is Heroin. The Naïve Bayes algorithm is also easy to construct because it has no complicated iterative parameter estimations, which is also helpful in large datasets.

The calculation is as follows.



The diagram shows the formula for the posterior probability in Naïve Bayes classification:
$$P(c | x) = \frac{P(x | c)P(c)}{P(x)}$$
 Blue arrows point from labels to parts of the formula: 'Likelihood' points to $P(x | c)$, 'Class Prior Probability' points to $P(c)$, 'Posterior Probability' points to $P(c | x)$, and 'Predictor Prior Probability' points to $P(x)$.

$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

Why Classification?

It is also important to understand why I'm only looking at classification instead of using any other algorithms like regression or clustering. Regression is an algorithm that predict an output value using the trained data. Instead, I'm trying to group the output into a single class, hence in this case, regression won't help. Similarly, clustering is unsupervised learning and its used, when the predicted output is unknown. However, in my scenario, I'm trying to predict if the primary drug is a cause for the accidental drug related death. In this situation, there is a key predicted output and therefore clustering is not appropriate.

5.2 Select the appropriate data-mining method(s) based on discussion

The appropriate data mining method(s) would be selected based on performing a trial and error method. In order to do so, the above two classification algorithms would be used. The purpose of this trial and error method is to select an informal selection method, that would best suit the dataset. Based on the output results and accuracy levels, the most appropriate data mining algorithm and the best model could be determined for accidental drug related deaths.

6 Data-mining Algorithm Selection

6.1 Conduct exploratory analysis and discuss

As discussed in point 5.1 the algorithms can be classified based on the input variables provided. In general, there can be two different learning techniques in the data mining process, namely supervised and unsupervised learning.

Supervised Learning - is a data mining technique that is appropriate when we know the target value that is required to predict the data. If the input and the output variables are known, then it is best to use supervised learning. The target value can be two or more outcomes and the data type could be either numerical or categorical. Classification and regression are some of the supervised learning algorithms that is commonly used.

Unsupervised Learning - is a data mining technique that is used when the input value is known and the output variable is unknown. In this situation, there can be one to many outcome predictions and there is no correct answer. The purpose of unsupervised learning is to understand the underlying structure of the model and its relation among data. A commonly used, unsupervised learning algorithm is clustering and association.

However, in this project I'm trying to predict if a person is prone for accidental drug related deaths if they are using heroin as their primary drug. Hence, during this project, I would be using the supervised learning technique as I have a clear target value that can predict the data.

6.2 Select data-mining algorithm based on discussion

The aim of the business objective is to predict if a person is prone for death (output variable) if they are using Heroin (input variable) as the primary drug. If Heroin is the primary drug that is causing the death, then value is 1 and else the value would be 0. Since the prediction is based on two values and it is clearly having an input variable and a predicted output, the data mining algorithm selection would be classification. As discussed in point 5.1, the two classification algorithms that is used in this project is Logistic Regression and Naïve Bayes classifiers.

Since I'm performing a trial and error method to find out the most suitable algorithm, in this step, I would be selecting both the algorithms.

6.3 Build/Select appropriate model(s) and choose relevant parameters

As per the discussion point 6.2, both the algorithms will be used in building and selecting the appropriate model and parameters.

Building and Selecting Relevant Parameters

As a first step, in choosing the relevant parameters, a selection statement was written to the dataframe.

```
#selecting the relevant features.
```

```
df = df.select('Year', 'Gender', 'Race', 'Age', 'age_group', 'County', 'Heroin')  
df.printSchema()
```

```
root  
|-- Year: integer (nullable = true)  
|-- Gender: string (nullable = true)  
|-- Race: string (nullable = true)  
|-- Age: integer (nullable = true)  
|-- age_group: string (nullable = true)  
|-- County: string (nullable = true)  
|-- Heroin: integer (nullable = true)
```

Then using a for loop, further evaluating the integer columns

```
In [54]: # Using a for loop to find all columns that belong to the integer data type.
numeric_features = [t[0] for t in df.dtypes if t[1] == 'int']

# Selecting the numeric features, generating summary statistics, and converting to a Pandas DataFrame.
df.select(numeric_features).describe().toPandas().transpose()
```

```
Out[54]:
```

| | 0 | 1 | 2 | 3 | 4 |
|---------|-------|--------------------|--------------------|------|------|
| summary | count | mean | stddev | min | max |
| Year | 3030 | 2014.4485148514852 | 1.3678702804975316 | 2012 | 2016 |
| Age | 3030 | 41.795709570957094 | 12.328031925398939 | 14 | 87 |
| Heroin | 3030 | 0.5508250825082508 | 0.4974922046215441 | 0 | 1 |

The next step is to create a string indexer which converts all the string in to a number

```
In [55]: # First creating a string indexer which converts every string into a number, such as male = 0 and female = 1.
# A number will be assigned to every category in the column.

gender_indexer = StringIndexer(inputCol='Gender',outputCol='genderIndex')
race_indexer = StringIndexer(inputCol='Race',outputCol='raceIndex')
agegroup_indexer = StringIndexer(inputCol='age_group',outputCol='agegroupIndex')
county_indexer = StringIndexer(inputCol='County',outputCol='countyIndex')
year_indexer = StringIndexer(inputCol='Year',outputCol='yearIndex')
age_indexer = StringIndexer(inputCol='Age',outputCol='ageIndex')
heroin_indexer = StringIndexer(inputCol='Heroin',outputCol='label')
```

Then using the OneHotEncoder, encode all the number, and convert all outputs in to single vector

```
# Then cone hot encode these numbers. This converts the various outputs into a single vector.
# Multiple columns are collapsed into one.
# This makes it easier to process when having multiple classes.

gender_encoder = OneHotEncoder(inputCol='genderIndex',outputCol='genderVec')
race_encoder = OneHotEncoder(inputCol='raceIndex',outputCol='raceVec')
agegroup_encoder = OneHotEncoder(inputCol='agegroupIndex',outputCol='agegroupVec')
county_encoder = OneHotEncoder(inputCol='countyIndex',outputCol='countyVec')
year_encoder = StringIndexer(inputCol='yearIndex',outputCol='yearVec')
age_encoder = StringIndexer(inputCol='ageIndex',outputCol='ageVec')
label_encoder = StringIndexer(inputCol='label',outputCol='label')
```

Then by the use of the vector assembler, turning all columns in to a single column

```
# And finally, using vector assembler to turn all of the columns into one column (named features).
assembler = VectorAssembler(inputCols=['genderVec','raceVec','agegroupVec','countyVec','yearVec','ageVec'], outputCol="features")
```

The next step is to create the pipeline, assigning it to a dataframe, transforming the results and removing all variables other than 'label' and 'features'

```

from pyspark.ml import Pipeline

# Then creating a pipeline
pipeline = Pipeline(stages=[gender_indexer, race_indexer, agegroup_indexer, county_indexer, year_indexer, age_indexer,
                             heroin_indexer, gender_encoder, race_encoder, agegroup_encoder, county_encoder, year_encoder,
                             age_encoder, assembler])

# Applying pipeline into a DataFrame.
pipeline_model = pipeline.fit(df)

# Incorporate results into a new DataFrame.
pipe_df = pipeline_model.transform(df)

# Remove all variables other than features and label.
pipe_df = pipe_df.select('label', 'features')

```

Selecting Appropriate Model

In order to select the appropriate model, the following steps are followed.

Logistic Regression

Importing the relevant modules from pyspark

```

In [62]: # Importing Logistic Regression model in pyspark.ml.classification
         from pyspark.ml.classification import LogisticRegression

```

Selecting the split for test and train data

```

#Selecting the split for train data and test data
train_data, test_data = pipe_df.randomSplit([0.7,0.3])
print("Training Dataset Count: " + str(train_data.count()))
print("Test Dataset Count: " + str(test_data.count()))

```

```

Training Dataset Count: 2113
Test Dataset Count: 917

```

The next step is to call the logistic regression module and assign train data and test data separately

```

#Assigning a variable and calling the Logistic regression
lr = LogisticRegression(labelCol = 'label', featuresCol = 'features', maxIter=10)

#fitting and training the lr model
lrModel = lr.fit(train_data)

```

```

#Creating a prediction and assigning the lr model test data
predictions = lrModel.transform(test_data)

```

The predicted first row can be displayed as follows

```
#Displaying the 1 row of the predicted values
predictions.take(1)

Out[64]: [Row(label=0.0, features=SparseVector(20, {0: 1.0, 1: 1.0, 6: 1.0, 10: 1.0}), rawPrediction=DenseVector([0.7432, -0.7432]), probability=DenseVector([0.6777, 0.3223]), prediction=0.0)]
```

Naïve Bayes

Importing the relevant modules from pyspark

```
#Importing NaiveBayes
from pyspark.ml.classification import NaiveBayes

#Importing MulticlassClassificationEvaluator
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
```

The next step is to call the Naïve Bayes module and assign train data and test data separately

```
# creating the trainer and set its parameters
nb = NaiveBayes(smoothing=1.0, modelType="multinomial")

# training the model
nbmodel = nb.fit(train_data)

# selecting example rows to display.
predictions = nbmodel.transform(test_data)
```

The predicted first row can be displayed as follows

```
predictions.take(1)
#predictions.show()

: [Row(label=0.0, features=SparseVector(20, {0: 1.0, 1: 1.0, 6: 1.0, 10: 1.0}), rawPrediction=DenseVector([-15.6704, -16.282]), probability=DenseVector([0.6483, 0.3517]), prediction=0.0)]
```

As discussed, I have built and selected both the algorithms and calculated metrics such as Test Accuracy and Test Area Under ROC for further evaluations of the models.

7 Data Mining

7.1 Create and justify test designs

The test design was created having the purpose in mind to find the best model that would suit the dataset. The key measures that was tested during the test design was to find the “test set accuracy”, “Coefficients” and the “area under the ROC curve”. As discussed in point 5.2, both Logistic Regression and Naïve Bayes algorithms were used in finding the best model.

The test model was executed using an Ubuntu platform through running a Jupyter Notebook. First the dataset was loaded, then encoded all variables into numeric values, and created an assembler vector with labels and features. The dataset was then split to train data and test data and each algorithm (Logistic Regression, Naïve Bayes) was tested separately to find out the accuracy of each algorithm.

As it was a trial and error method to find the best fit, during the test design, different split values were considered to see if it would make a difference. As a result, the split values that was used during the test design was 0.8/0.2, 0.7/0.3, 0.6/0.4, and 0.5/0.5. The reason to use these particular split values was to identify, which split would fit perfectly without overfitting or underfitting my model.

Using the following code, the split of each train and test sets was changed to find the best fit.

```
#Selecting the split for train data and test data
train_data, test_data = pipe_df.randomSplit([0.7,0.3])
print("Training Dataset Count: " + str(train_data.count()))
print("Test Dataset Count: " + str(test_data.count()))
```

7.2 Conduct data mining

The datamining is carried out using AWS, Pyspark and Jupyter notebook. The file is attached and shared for better understanding.

7.3 Search for Patterns

The following patterns were discovered during the data mining process.

Different Splits Results

| Split (train, test) | Training Dataset Count | Test Dataset Count | Algorithm | Test set Accuracy | Are Under ROC Curve |
|------------------------|---------------------------|-----------------------|---------------------|----------------------|------------------------|
| 0.8, 0.2 | 2425 | 605 | Logistic Regression | 0.618 | 0.7 |
| | | | Naïve Bayes | 0.593 | 0.5 |
| 0.7, 0.3 | 2122 | 908 | Logistic Regression | 0.621 | 0.7 |
| | | | Naïve Bayes | 0.630 | 0.5 |
| 0.6, 0.4 | 1830 | 1200 | Logistic Regression | 0.618 | 0.7 |
| | | | Naïve Bayes | 0.605 | 0.5 |
| 0.5, 0.5 | 1499 | 1531 | Logistic Regression | 0.619 | 0.7 |
| | | | Naïve Bayes | 0.611 | 0.5 |

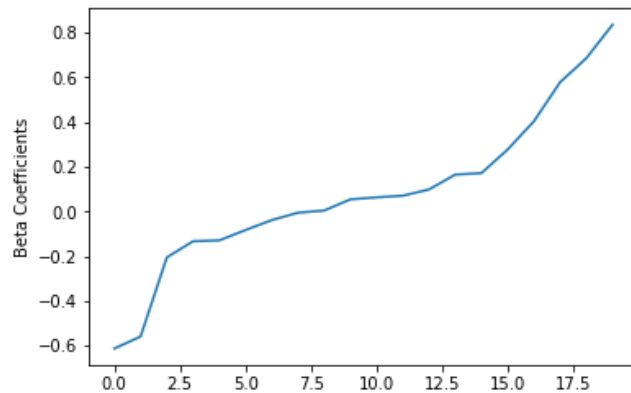
The Core-efficiency

```
# Visualising the coefficients. Sort from lowest to highest.
beta = np.sort(lrModel.coefficients)

# Plot the data.
plt.plot(beta)

# Add a Label to the data.
plt.ylabel('Beta Coefficients')

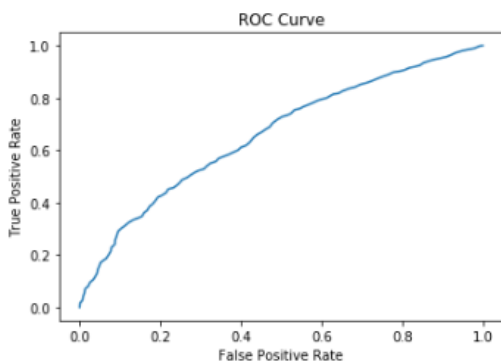
# Show the graph.
plt.show()
```



The ROC Curve

```
# Let's get a summary of the data.
training_summary = lrModel.summary
# Convert the DataFrame to a Pandas DataFrame.
ROC = training_summary.roc.toPandas()
# Plot the true positive and false positive rates.
plt.plot(ROC['FPR'], ROC['TPR'])
# Define the Labels.
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title('ROC Curve')
plt.show()

# Print the AUC statistic.
print('Area Under the Curve: ' + str(training_summary.areaUnderROC))
```



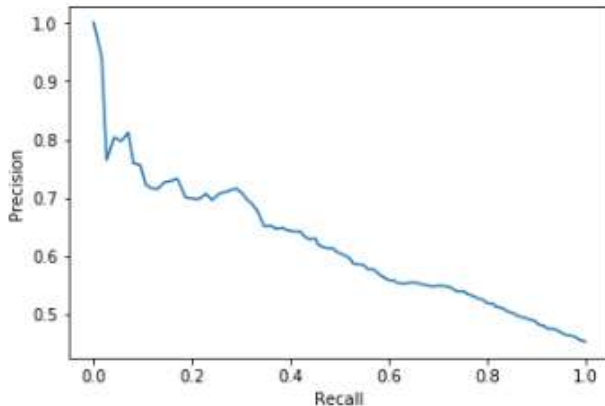
Area Under the Curve: 0.6597260965093538

Precision and Recall

```
# Convert DataFrame to Pandas DataFrame.
pr = training_summary.pr.toPandas()

# Plot model recall and precision.
plt.plot(pr['recall'],pr['precision'])

# Define the labels and show the graph.
plt.ylabel('Precision')
plt.xlabel('Recall')
plt.show()
```



8 Interpretation

8.1 Study and discuss the mined patterns

As discussed and visualized in point 7.1 and 7.3, the following key study areas can be highlighted within the mined patterns.

Test set accuracy: The test set accuracy represents the prediction strength of each algorithm used to evaluate the given dataset. So, if higher the accuracy the better the prediction in results. Which means that if the two models Logistic Regression and Naïve Bayes is giving higher accuracy levels then the prediction power will be high. Similarly, if the models have a lower accuracy, then lower the prediction results.

Area under the ROC curve: ROC stands for Receiver Operating Characteristic and the area under the ROC curve is a measure to evaluate the test. If the area under the curve is high, then the usefulness of the test is high. If the area under the curve is low, then the usefulness is also comparatively low. In an ideal situation, the model should be able to predict if a person is prone for death, if their using Heroin as the primary drug. In this instance, the ROC curve will not have any overlap between sensitivity and specificity.

Coefficients: The transformation of a binary outcome model has a linear relationship to the predicted variable. Thus, it's a monotonic transformation from probability to odds. Which means, as the odds increase the probability transformation will increase accordingly. If y is binary outcome variable (Heroin/Not Heroin) then x is a set of predictor variables (Cause for the death). Then it estimates parameter values β that can find the likelihood of the model. The equation that calculates the coefficients is as follows.

$$\text{logit}(p) = \log(p/(1-p)) = \beta_0 + \beta_1 \cdot x_1 + \dots$$

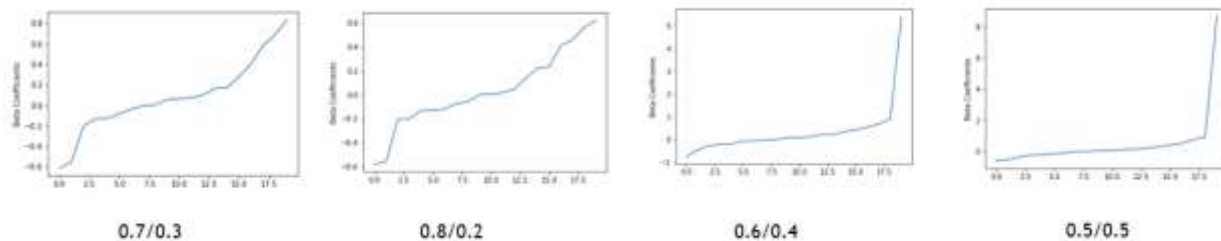
Precision and Recall: It is said² that precision is the measure of result relevancy, while recall is a measure of how many truly relevant results are returned. As a result, a high area under the curve represents high recall and high precision. Similarly, if the model is having high precision rates, then it relates to a low false positive rate; and if it has high recalls, it relates to low false negative rates.

8.2 Visualize the data, results, models and patterns

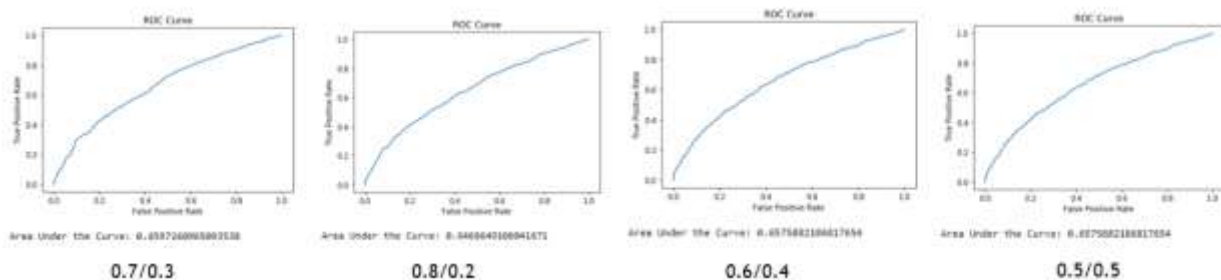
Best split comparison

| Split (train, test) | Training Dataset Count | Test Dataset Count | Algorithm | Test set Accuracy | Are Under ROC Curve |
|------------------------|---------------------------|-----------------------|---------------------|----------------------|------------------------|
| 0.8, 0.2 | 2425 | 605 | Logistic Regression | 0.618 | 0.7 |
| | | | Naïve Bayes | 0.593 | 0.5 |
| 0.7, 0.3 | 2122 | 908 | Logistic Regression | 0.621 | 0.7 |
| | | | Naïve Bayes | 0.630 | 0.5 |
| 0.6, 0.4 | 1830 | 1200 | Logistic Regression | 0.618 | 0.7 |
| | | | Naïve Bayes | 0.605 | 0.5 |
| 0.5, 0.5 | 1499 | 1531 | Logistic Regression | 0.619 | 0.7 |
| | | | Naïve Bayes | 0.611 | 0.5 |

Coefficients Comparison

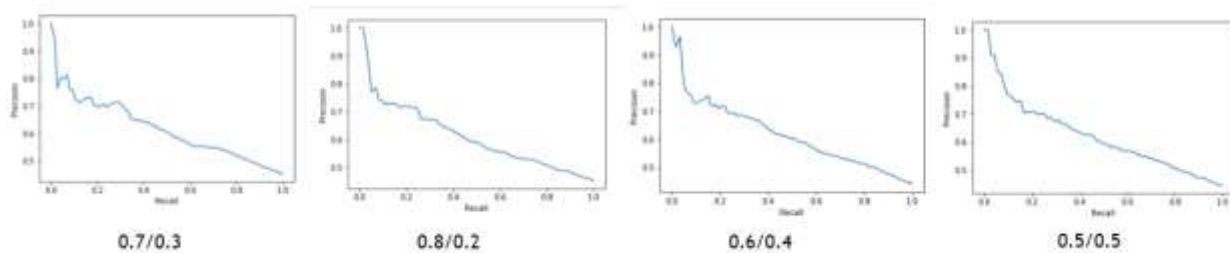


ROC Curve Comparison



² Retrieved from: http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

Precision and Recall Comparison



8.3 Interpret the results, models and patterns

Interpretation in different split values

Comparing the table in point 8.2, the test accuracy gave the highest results for both Logistic Regression (LR) and Naïve Bayes (NV) when the split value was 0.7 and 0.3 for train and test data. But however, during this split, the LR ROC curve value was higher than NV ROC curve value. It is said³ that area under ROC curve of a perfect test is 1 when there is 100% sensitivity and 100% specificity. Also, the ROC curve of the worthless test falls on the diagonal line, when there is 50% sensitivity and 50% specificity giving the result of 0.5. Even though NV gave a slightly higher accuracy in the test set, the area under the curve of NV is remaining at 0.5. Hence it is clear that LR gave an accuracy of 62 % and ROC curve of 0.7 when the split value was 0.7 and 0.3 which is also the highest out of all the carried tests.

During split values 0.8/0.2, 0.6/0.4 and 0.5/0.5; the test accuracy of both the LR and NV dropped slightly in contrast to split value 0.7/0.3. Also, it was clear that ROC curve of both LR and NV remained same for all four splits. So as result the best split value during the design was given in split 0.7/0.3 with **Logistic Regression** model.

Hence, Logistic regression is giving the best fit, further evaluation can be carried out as follows.

Interpretation in Coefficients

The coefficients in logistic regression is meant to give a cubic parabola curve, once it is in its ideal state. During this state the values of the curve will balance between 0 and 1 based on the predicted probability. Looking at the four coefficients, it is observable that the curve is somewhat similar to a cubic parabola curve during 0.6/0.4 and 0.5/0.5 splits compared to 0.7/0.3 and 0.8/0.2. But this could be a reason of underfitting the two splits of 0.5/0.5 and 0.6/0.4.

Interpretation in Precision and Recall

The precision and recall of all the four diagrams give a similar interpretation during all the splits. Hence it is arguable that the dataset used in the data mining process is producing a similar output regardless of the split variation. However, the ideal state is a bend curve that is

³ Retrieved from: <https://acutecaretesting.org/en/articles/roc-curves-what-are-they-and-how-are-they-used>

decreasing between 0 and 1. Even though the four diagrams are not in its ideal state, it is somewhat similar to an average state of the precision and recall. Which means the true positive rate and false negative rate of the model is in its average level in predicting the accuracy for heroin being the primary drug for being the reason for accidental drug related deaths.

8.4 Asses and evaluate results

The data mining process was carried out using two algorithms namely, Logistic Regression and Naïve Bayes. This was a trial and error method that was carried out to find the model that best fit the dataset. The test design was created out using four different splits for training and test data. In each of these splits the output test accuracy and ROC curve was compared between the two algorithms. However, the test result supported the Logistic Regression with the split value 0.7 and 0.3. Similarly, measuring points like coefficients, precision and recall was also considered during the data mining process to support the model that best suit the prediction “if heroin is the primary drug that is causing for accidental drug related deaths”.

The prediction accuracy is 62% and this can also be argued that the model is giving an average result because its not biased with overfitted or underfitted datasets. However, further studies can be carried out in future to improve the accuracy using different algorithms.

8.5 Iterate step 1-7

Step 1-7 was carried out multiple times during previous iterations and the following conclusions are reached.

The CRISP-DM model was used during 2nd and 3rd iteration to better understand the data mining method selection. During 2nd iteration unsupervised learning technique (Clustering Node) was used to understand the relationship between each variable. Later during the project, it was realized that the business objective was to predict if the victims are prone for accidental death if they are using Heroin as their primary drug. Since I’m trying to predict a class, the data mining method selection was classification. Then during the 3rd iteration MS azure machine learning was used to test different classification algorithms for better accuracy and fast training. Also, in previous iterations, the step 1 to 7 was re-iterated multiple times to come in to these conclusions.

Reference

- A. Accidental drug related deaths within 2012-2017. (data. gov)
https://catalog.data.gov/dataset?res_format=CSV&tags=drug
- B. Retrieved from: http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html
- C. Retrieved from: <https://acutecaretesting.org/en/articles/roc-curves-what-are-they-and-how-are-they-used>
- D. Retrieved from <https://www.cdc.gov/media/releases/2018/p0329-drug-overdose-deaths.html>