# DATA CLEANING, EXPLORATION AND MACHINE LEARNING OF INPC DATASET

In [3]:

```
# Importing the libraries required
from pyspark.sql import Row
from pyspark.sql.types import *
from pyspark.sql.functions import sum
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from pyspark.sql.functions import rank, col, unix_timestamp, from_unixtime, t
from pyspark.sql import functions as F
import seaborn as sns
timeFmt = "yyyy-MM-dd"
from pyspark.sql.functions import *
```

In [4]:

```
#Reading INPC data
df_inpc_raw = spark.read.csv("inpc_master.csv", header='true', inferSchema='t
```

## DATA EXPLORATION AND CALCULATION

In [7]:

```
df_inpc_raw.show(2)
```

```
+---------+------+---------------+----------+----------+----------+-------
---+---------+---------+----------+-----------+-------------------+
|person_id|gender|           race|F_T2D_Diag|F_T1D_Diag|F_DKA_Diag|F_CVD_D
iag|F_LD_Diag|F_KD_Diag|F_ALZ_Diag|F_ALZD_Diag|          Birthdate|
+---------+------+---------------+----------+----------+----------+-------
---+---------+---------+----------+-----------+-------------------+
|        1|     M|african_american|      null|      null|      null|      n
ull|     null|     null|      null|       null|1919-08-02 00:00:00|
|       10|     F|          white|      null|      null|      null|      n
ull|     null|     null|      null|       null|1943-02-19 00:00:00|
+---------+------+---------------+----------+----------+----------+-------
---+---------+---------+----------+-----------+-------------------+
only showing top 2 rows
```

In [5]:
```
df_inpc_raw.cache()
df_inpc_raw.printSchema()
```

```
root
 |-- person_id: integer (nullable = true)
 |-- gender: string (nullable = true)
 |-- race: string (nullable = true)
 |-- F_T2D_Diag: timestamp (nullable = true)
 |-- F_T1D_Diag: timestamp (nullable = true)
 |-- F_DKA_Diag: timestamp (nullable = true)
 |-- F_CVD_Diag: timestamp (nullable = true)
 |-- F_LD_Diag: timestamp (nullable = true)
 |-- F_KD_Diag: timestamp (nullable = true)
 |-- F_ALZ_Diag: timestamp (nullable = true)
 |-- F_ALZD_Diag: timestamp (nullable = true)
 |-- Birthdate: timestamp (nullable = true)
```

In [6]:
```
print('Total population: ', df_inpc_raw.count())
print('---------------------------------------------')
print('Population count by gender')
df_inpc_raw.groupBy('gender').count().show()
```

```
Total population:  1060976
---------------------------------------------
Population count by gender
+------+------+
|gender| count|
+------+------+
|     F|557014|
|     M|503739|
|     U|   223|
+------+------+
```

In [8]:
```
#Certain selected columns as required
df_inpc=df_inpc_raw.select('person_id','gender','Birthdate','F_T2D_Diag','F_A
df_inpc.show(5)
```

```
+---------+------+-------------------+-------------------+----------+
|person_id|gender|          Birthdate|         F_T2D_Diag|F_ALZ_Diag|
+---------+------+-------------------+-------------------+----------+
|        1|     M|1919-08-02 00:00:00|               null|      null|
|       10|     F|1943-02-19 00:00:00|               null|      null|
|      100|     F|1935-07-31 00:00:00|               null|      null|
|     1000|     F|1955-02-20 00:00:00|2016-02-22 00:00:00|      null|
|    10000|     M|1969-09-10 00:00:00|2016-06-05 00:01:00|      null|
+---------+------+-------------------+-------------------+----------+
only showing top 5 rows
```

In [9]:
```
print('Total population: ', df_inpc.count())
print('---------------------------------------------')
print('Population count by gender')
df_inpc.groupBy('gender').count().show()
```

```
Total population:  1060976
---------------------------------------------
Population count by gender
+------+------+
|gender| count|
+------+------+
|     F|557014|
|     M|503739|
|     U|   223|
+------+------+
```

In [11]:
```
#print('Average age of total population')
#df_inpc.select(mean("age")).show()
```

## Which Disease is diagnosed first?

In [12]:
```
# Order of Disease Diagnosis
T2D_OR_AD_FIRST =F.round((F.col("F_T2D_Diag").cast("long") - F.col("F_ALZ_Dia
df_inpc=df_inpc.withColumn("T2D_OR_AD_FIRST",T2D_OR_AD_FIRST)
```

## People who have only T2D meaning AD not at all(control data)

In [13]:

```
#people who have only T2D but not AD------------------control data
df_inpc_control=df_inpc.filter(df_inpc.F_T2D_Diag.isNotNull() & df_inpc.F_ALZ
df_inpc_control.show(5)
```

```
+---------+------+-------------------+-------------------+----------+------
---------+
|person_id|gender|          Birthdate|         F_T2D_Diag|F_ALZ_Diag|T2D_OR
_AD_FIRST|
+---------+------+-------------------+-------------------+----------+------
---------+
|     1000|     F|1955-02-20 00:00:00|2016-02-22 00:00:00|      null|
null|
|    10000|     M|1969-09-10 00:00:00|2016-06-05 00:01:00|      null|
null|
|   100000|     F|1955-07-20 00:00:00|2016-05-21 00:00:00|      null|
null|
|   100002|     F|1956-03-27 00:00:00|2018-06-25 00:00:00|      null|
null|
|  1000036|     M|1974-04-02 00:00:00|2016-04-02 22:09:00|      null|
null|
+---------+------+-------------------+-------------------+----------+------
---------+
only showing top 5 rows
```

In [14]:

```
print('Total number of people with only T2D diagnosed: ')
df_inpc_control.count()
```

```
Total number of people with only T2D diagnosed:
```

Out[14]:

```
293354
```

In [15]:

```
print('Total number of people with only T2D diagnosed by gender')
df_inpc_control.groupby(["gender"]).count().show()
```

```
Total number of people with only T2D diagnosed by gender
+------+------+
|gender| count|
+------+------+
|     F|150492|
|     M|142846|
|     U|    16|
+------+------+
```

## People diagonosed with both or either one disease

In [16]:
```
# Disease not null(people diagnosed either one or both diseases)
df_inpc_disease=df_inpc.filter(df_inpc.F_T2D_Diag.isNotNull() & df_inpc.F_ALZ
df_inpc_disease.show(5)
```

```
+---------+------+-------------------+-------------------+-----------------
--+----------------+
|person_id|gender|          Birthdate|         F_T2D_Diag|        F_ALZ_Di
ag|T2D_OR_AD_FIRST|
+---------+------+-------------------+-------------------+-----------------
--+----------------+
|    10009|     F|1965-06-16 00:00:00|2016-01-20 00:00:00|2016-11-08 15:29:
00|         -0.805|
|   100095|     M|1936-04-16 00:00:00|2016-04-27 00:00:00|2018-05-29 00:00:
00|         -2.088|
|  1001407|     M|1961-03-19 00:00:00|2016-01-31 00:01:00|2017-04-24 13:59:
04|         -1.232|
|  1001652|     M|1963-05-16 00:00:00|2017-04-16 00:01:00|2016-05-09 00:01:
00|          0.937|
|   100195|     M|1986-08-26 00:00:00|2016-03-31 00:00:00|2016-02-03 00:00:
00|          0.156|
+---------+------+-------------------+-------------------+-----------------
--+----------------+
only showing top 5 rows
```

In [17]:
```
print('Total number of people with both diseases: ')
df_inpc_disease.count()
```

```
Total number of people with both diseases:
```

Out[17]:
```
8044
```

In [18]:
```
print('Total number of people with both diseases by gender')
df_inpc_disease.groupby(["gender"]).count().show()
```

```
Total number of people with both diseases by gender
+------+-----+
|gender|count|
+------+-----+
|     F| 4211|
|     M| 3833|
+------+-----+
```

## People diagonosed with AD only meaning T2D not at all

In [19]:
```
#people who have only AD but not T2D
df_inpc_AD_only=df_inpc.filter(df_inpc.F_T2D_Diag.isNull() & df_inpc.F_ALZ_Di
df_inpc_AD_only.show(5)
```

```
+---------+------+-------------------+----------+-------------------+------
---------+
|person_id|gender|          Birthdate|F_T2D_Diag|         F_ALZ_Diag|T2D_OR
_AD_FIRST|
+---------+------+-------------------+----------+-------------------+------
---------+
|  1000104|     F|1951-07-21 00:00:00|      null|2017-05-20 00:01:00|
null|
|  1000122|     M|1998-07-30 00:00:00|      null|2016-09-19 21:58:00|
null|
|  1000299|     M|1995-10-23 00:00:00|      null|2016-01-26 08:49:57|
null|
|  1000390|     M|2007-12-08 00:00:00|      null|2017-04-01 14:05:00|
null|
|  1000851|     F|2010-07-14 00:00:00|      null|2018-02-10 10:12:00|
null|
+---------+------+-------------------+----------+-------------------+------
---------+
only showing top 5 rows
```

In [20]:
```
print('Total number of people with only AD diagnosed: ')
df_inpc_AD_only.count()
```

```
Total number of people with only AD diagnosed:
```

Out[20]:
```
2539
```

In [21]:
```
print('Total number of people with only AD diagnosed by gender')
df_inpc_AD_only.groupby(["gender"]).count().show()
```

```
Total number of people with only AD diagnosed by gender
+------+-----+
|gender|count|
+------+-----+
|     F| 1296|
|     M| 1243|
+------+-----+
```

## Population calculation based on order of diagnosis

In [22]:
```
df_inpc_disease.show(5)
```

```
+---------+------+-------------------+-------------------+-----------------
--+---------------+
|person_id|gender|          Birthdate|         F_T2D_Diag|         F_ALZ_Di
ag|T2D_OR_AD_FIRST|
+---------+------+-------------------+-------------------+-----------------
--+---------------+
|    10009|     F|1965-06-16 00:00:00|2016-01-20 00:00:00|2016-11-08 15:29:
00|         -0.805|
|   100095|     M|1936-04-16 00:00:00|2016-04-27 00:00:00|2018-05-29 00:00:
00|         -2.088|
|  1001407|     M|1961-03-19 00:00:00|2016-01-31 00:01:00|2017-04-24 13:59:
04|         -1.232|
|  1001652|     M|1963-05-16 00:00:00|2017-04-16 00:01:00|2016-05-09 00:01:
00|          0.937|
|   100195|     M|1986-08-26 00:00:00|2016-03-31 00:00:00|2016-02-03 00:00:
00|          0.156|
+---------+------+-------------------+-------------------+-----------------
--+---------------+
only showing top 5 rows
```

## People with T2D diagnosed first

In [23]:
```
print('People with T2D diagnosed first')
T2D_First=df_inpc_disease.filter(df_inpc_disease.T2D_OR_AD_FIRST<=0)
T2D_First.count()
```

```
People with T2D diagnosed first
```

Out[23]:
```
5311
```

In [24]:
```
print('People with T2D diagnosed first by gender')
T2D_First.groupby('gender').count().show()
```

```
People with T2D diagnosed first by gender
+------+-----+
|gender|count|
+------+-----+
|     F| 2799|
|     M| 2512|
+------+-----+
```

## People with both T2D and AD diagnosed at the same time

In [25]:
```
print('People with both T2D and AD diagnosed at the same time')
T2D_AD=df_inpc_disease.filter(df_inpc_disease.T2D_OR_AD_FIRST==0)
T2D_AD.count()
```

```
People with both T2D and AD diagnosed at the same time
```

Out[25]:
```
715
```

In [26]:
```
print('People with both T2D and AD diagnosed at the same time by gender')
T2D_AD.groupby('gender').count().show()
```

```
People with both T2D and AD diagnosed at the same time by gender
+------+-----+
|gender|count|
+------+-----+
|     F|  337|
|     M|  378|
+------+-----+
```

## People with AD diagnosed first

In [27]:
```
print('People with AD diagnosed first')
AD_First=df_inpc_disease.filter(df_inpc_disease.T2D_OR_AD_FIRST>0)
AD_First.count()
```

```
People with AD diagnosed first
```

Out[27]:
```
2733
```

In [28]:
```
print('People with AD diagnosed first by gender')
AD_First.groupby('gender').count().show()
```

```
People with AD diagnosed first by gender
+------+-----+
|gender|count|
+------+-----+
|     F| 1412|
|     M| 1321|
+------+-----+
```

## Calculation of age of diagnosis

In [29]:
```
#Age at which diseases diagnosed
Age_T2D_First =F.round((F.col("F_T2D_Diag").cast("long") - F.col("Birthdate")
Age_AD_First =F.round((F.col("F_ALZ_Diag").cast("long") - F.col("Birthdate").
```

In [30]:
```
df_inpc=df_inpc.withColumn("Age_T2D_First",Age_T2D_First).withColumn("Age_AD_
```

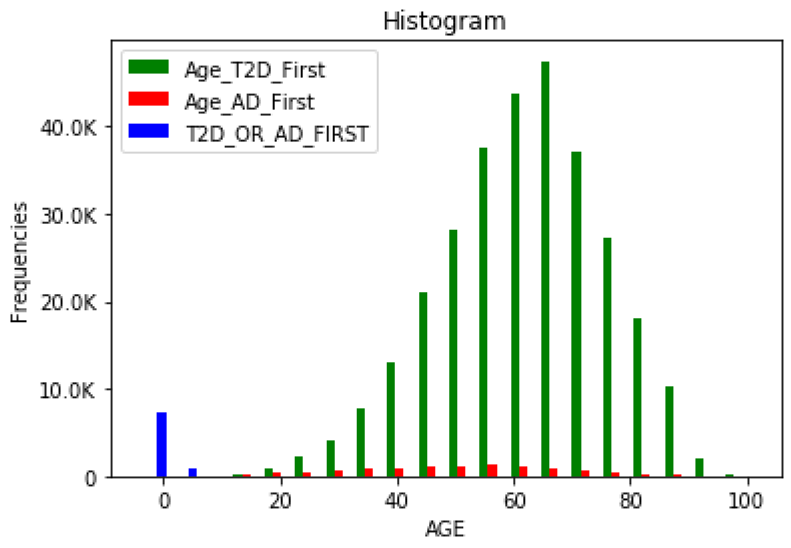
In [31]:
```
df_inpc.show(5)
```

```
+---------+------+-------------------+-------------------+---------+------
---------+-------------+------------+
|person_id|gender|          Birthdate|         F_T2D_Diag|F_ALZ_Diag|T2D_OR
_AD_FIRST|Age_T2D_First|Age_AD_First|
+---------+------+-------------------+-------------------+---------+------
---------+-------------+------------+
|        1|     M|1919-08-02 00:00:00|               null|      null|
null|         null|        null|
|       10|     F|1943-02-19 00:00:00|               null|      null|
null|         null|        null|
|      100|     F|1935-07-31 00:00:00|               null|      null|
null|         null|        null|
|     1000|     F|1955-02-20 00:00:00|2016-02-22 00:00:00|      null|
null|       61.046|        null|
|    10000|     M|1969-09-10 00:00:00|2016-06-05 00:01:00|      null|
null|       46.767|        null|
+---------+------+-------------------+-------------------+---------+------
---------+-------------+------------+
only showing top 5 rows
```

In [32]:
```
#histogram  disease data frame plot age T2D first and gender
df_inpc_hist=df_inpc.select('Age_T2D_First','Age_AD_First','T2D_OR_AD_FIRST')
```

```
In [33]:  #histogram of age of T2D and age of AD in one graph
          from pyspark_dist_explore import hist
          fig, ax = plt.subplots()
          hist(ax, df_inpc_hist, bins = 20, color=['green','red','blue'])
          ax.set_ylabel('Frequencies')
          ax.set_xlabel('AGE')
          ax.set_title('Histogram')
          ax.legend(prop={'size': 10})
```

```
Out[33]: <matplotlib.legend.Legend at 0x1489725a348>
```



```
In [35]:  df_inpc=df_inpc.select('person_id','gender','Age_T2D_First','Age_AD_First','T
```

```
In [36]:  df_inpc.columns
```

```
Out[36]: ['person_id', 'gender', 'Age_T2D_First', 'Age_AD_First', 'T2D_OR_AD_FIRST']
```

```
In [37]:  pd_inpc=df_inpc.toPandas()
```

```
In [38]:  pd_inpc.head(5)
```

Out[38]:

| | person_id | gender | Age_T2D_First | Age_AD_First | T2D_OR_AD_FIRST |
|---|---|---|---|---|---|
| **0** | 1 | M | NaN | NaN | NaN |
| **1** | 10 | F | NaN | NaN | NaN |
| **2** | 100 | F | NaN | NaN | NaN |
| **3** | 1000 | F | 61.046 | NaN | NaN |
| **4** | 10000 | M | 46.767 | NaN | NaN |

## Saving INPC as inpc_final CSV file

```
In [40]: pd_inpc.to_csv('final_inpc.csv')
```

```
In [ ]:
```