Matthew Afsahi

In this project, I will be using Appache Spark engine and SQL to queries the data sets.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.r
```

```
# Installing  java, appache spark and related libraries

!apt-get install openjdk-8-jdk-headless -qq > /dev/null

# install spark (change the version number if needed)
!wget -q https://archive.apache.org/dist/spark/spark-3.0.0/spark-3.0.0

# unzip the spark file to the current folder
!tar xf spark-3.0.0-bin-hadoop3.2.tgz

# set your spark folder to your system path environment.
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.0.0-bin-hadoop3.2"


# install findspark using pip
!pip install -q findspark

!pip install pyspark


```

```
Requirement already satisfied: pyspark in /usr/local/lib/python3.6/dist-packages (3.0
Requirement already satisfied: py4j==0.10.9 in /usr/local/lib/python3.6/dist-packages
```

```
# Iniating the spark on this notebook
import findspark
```

```
findspark.init('/content/spark-3.0.0-bin-hadoop3.2')
findspark.init('/content/spark-3.0.0-bin-hadoop3.2')

```

```
from pyspark.sql import SparkSession
```

```
spark=SparkSession.builder.appName('MyAssignment_3').getOrCreate()

```

```
#Reading the census data set from the could
census=spark.read.csv('/content/drive/My Drive/Colab Notebooks/data/Ce
```

```
census.printSchema()
```

```
root
 |-- COMMUNITY_AREA_NUMBER: integer (nullable = true)
 |-- COMMUNITY_AREA_NAME: string (nullable = true)
 |-- PERCENT OF HOUSING CROWDED: double (nullable = true)
 |-- PERCENT HOUSEHOLDS BELOW POVERTY: double (nullable = true)
 |-- PERCENT AGED 16+ UNEMPLOYED: double (nullable = true)
 |-- PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA: double (nullable = true)
 |-- PERCENT AGED UNDER 18 OR OVER 64: double (nullable = true)
 |-- PER_CAPITA_INCOME : integer (nullable = true)
 |-- HARDSHIP_INDEX: integer (nullable = true)
```

```
#Reading the school data set from the cloud
school=spark.read.csv('/content/drive/My Drive/Colab Notebooks/data/Ch
```

```
school.printSchema()
```

```
 |-- Environment Icon : string (nullable = true)
 |-- Environment Score: integer (nullable = true)

 |-- Instruction Icon : string (nullable = true)
 |-- Instruction Score: integer (nullable = true)
 |-- Leaders Icon : string (nullable = true)
 |-- Leaders Score : string (nullable = true)
 |-- Teachers Icon : string (nullable = true)
 |-- Teachers Score: string (nullable = true)
 |-- Parent Engagement Icon : string (nullable = true)
 |-- Parent Engagement Score: string (nullable = true)
 |-- Parent Environment Icon: string (nullable = true)
 |-- Parent Environment Score: string (nullable = true)
 |-- AVERAGE_STUDENT_ATTENDANCE: string (nullable = true)
 |-- Rate of Misconducts (per 100 students) : double (nullable = true)
 |-- Average Teacher Attendance: string (nullable = true)
 |-- Individualized Education Program Compliance Rate : string (nullable = true)
 |-- Pk-2 Literacy %: string (nullable = true)
 |-- Pk-2 Math %: string (nullable = true)
 |-- Gr3-5 Grade Level Math %: string (nullable = true)
 |-- Gr3-5 Grade Level Read % : string (nullable = true)
 |-- Gr3-5 Keep Pace Read %: string (nullable = true)
 |-- Gr3-5 Keep Pace Math %: string (nullable = true)
 |-- Gr6-8 Grade Level Math %: string (nullable = true)
 |-- Gr6-8 Grade Level Read %: string (nullable = true)
 |-- Gr6-8 Keep Pace Math%: string (nullable = true)
 |-- Gr6-8 Keep Pace Read %: string (nullable = true)
 |-- Gr-8 Explore Math %: string (nullable = true)
 |-- Gr-8 Explore Read %: string (nullable = true)
 |-- ISAT Exceeding Math %: double (nullable = true)
 |-- ISAT Exceeding Reading % : double (nullable = true)
 |-- ISAT Value Add Math: double (nullable = true)
 |-- ISAT Value Add Read: double (nullable = true)
 |-- ISAT Value Add Color Math: string (nullable = true)
 |-- ISAT Value Add Color Read: string (nullable = true)
 |-- Students Taking  Algebra %: string (nullable = true)
 |-- Students Passing  Algebra %: string (nullable = true)
 |-- 9th Grade EXPLORE (2009) : string (nullable = true)
 |-- 9th Grade EXPLORE (2010) : string (nullable = true)
 |-- 10th Grade PLAN (2009) : string (nullable = true)
 |-- 10th Grade PLAN (2010) : string (nullable = true)
 |-- Net Change EXPLORE and PLAN: string (nullable = true)
 |-- 11th Grade Average ACT (2011) : string (nullable = true)
 |-- Net Change PLAN and ACT: string (nullable = true)
 |-- College Eligibility %: string (nullable = true)
 |-- Graduation Rate %: string (nullable = true)
 |-- College Enrollment Rate %: string (nullable = true)
 |-- COLLEGE_ENROLLMENT: integer (nullable = true)
 |-- General Services Route : integer (nullable = true)
 |-- Freshman on Track Rate %: string (nullable = true)
 |-- X_COORDINATE: double (nullable = true)
 |-- Y_COORDINATE: double (nullable = true)
 |-- Latitude: double (nullable = true)
 |-- Longitude: double (nullable = true)
 |-- COMMUNITY_AREA_NUMBER: integer (nullable = true)
 |-- COMMUNITY_AREA_NAME: string (nullable = true)
 |-- Ward: integer (nullable = true)
 |-- Police District: integer (nullable = true)
 |-- Location: string (nullable = true)
```

```
census.show(10)
```

```
+---------------------+-------------------+--------------------------+-------------
|COMMUNITY_AREA_NUMBER|COMMUNITY_AREA_NAME|PERCENT OF HOUSING CROWDED|PERCENT HOUSEH
+---------------------+-------------------+--------------------------+-------------
|                    1|        Rogers Park|                       7.7|
|                    2|         West Ridge|                       7.8|
|                    3|             Uptown|                       3.8|
|                    4|     Lincoln Square|                       3.4|
|                    5|       North Center|                       0.3|
|                    6|          Lake View|                       1.1|
|                    7|       Lincoln Park|                       0.8|
|                    8|    Near North Side|                       1.9|
|                    9|        Edison Park|                       1.1|
|                   10|       Norwood Park|                       2.0|
+---------------------+-------------------+--------------------------+-------------
only showing top 10 rows
```

```
# Creating a SQL temp for more queries, Note that SQL is working on to
census.createOrReplaceTempView('Census')
```

Q1. What was the per capita income in North Park comunity? 26576

```
spark.sql("select * \
            from Census\
            where Census.COMMUNITY_AREA_NAME== 'North Park'").show()
```

```
+--------------------+-------------------+-------------------------+--------------
|COMMUNITY_AREA_NUMBER|COMMUNITY_AREA_NAME|PERCENT OF HOUSING CROWDED|PERCENT HOUSEHO
+--------------------+-------------------+-------------------------+--------------
|                  13|         North Park|                      3.9|
+--------------------+-------------------+-------------------------+--------------
```

Q2. How many comunity area were in the census data base? 77

```
spark.sql("select count(COMMUNITY_AREA_NUMBER) \
            from Census ").show()

```

```
+---------------------------+
|count(COMMUNITY_AREA_NUMBER)|
+---------------------------+
|                         77|
+---------------------------+
```

```
school.show(10)
```

```
+---------+--------------------+--------------------------------------+------------------
|School ID|      NAME_OF_SCHOOL|Elementary, Middle, or High School|      Street Addre
+---------+--------------------+--------------------------------------+------------------
|   610038|Abraham Lincoln E...|                                    ES|    615 W Kemper P
|   610281|Adam Clayton Powe...|                                    ES|7511 S South Shor
|   610185|Adlai E Stevenson...|                                    ES| 8010 S Kostner Av
|   609993|Agustin Lara Elem...|                                    ES| 4619 S Wolcott Av
|   610513|Air Force Academy...|                                    HS|    3630 S Wells S
|   610212|Albany Park Multi...|                                    MS|  4929 N Sawyer Av
|   609720|Albert G Lane Tec...|                                    HS|  2501 W Addison S
|   610342|Albert R Sabin El...|                                    ES|   2216 W Hirsch S
|   610524|Alcott High Schoo...|                                    HS|   2957 N Hoyne Av
|   610209|Alessandro Volta ...|                                    ES|   4950 N Avers Av
+---------+--------------------+--------------------------------------+------------------
only showing top 10 rows
```

```
# Creating the School temp data base on the cloud
# creating the School temp data base on the cloud
school.createOrReplaceTempView('School')
```

Q3. How many unique School are in Chicago data base system? 566

```
spark.sql("select  distinct count(NAME_OF_SCHOOL) \
            from School").show()
```

```
+---------------------+
|count(NAME_OF_SCHOOL)|
+---------------------+
|                  566|
+---------------------+
```

```
census.describe( ).show()
```

```
+-------+--------------------+-------------------+-------------------------+-------
|summary|COMMUNITY_AREA_NUMBER|COMMUNITY_AREA_NAME|PERCENT OF HOUSING CROWDED|PERCENT
+-------+--------------------+-------------------+-------------------------+-------
|  count|                  77|                 78|                        78|
|   mean|                39.0|               null|         4.920512820512823|
| stddev|  22.371857321197094|               null|           3.6589814413502|
|    min|                   1|       Albany Park|                       0.3|
|    max|                  77|          Woodlawn|                      15.8|
+-------+--------------------+-------------------+-------------------------+-------
```

```
census.summary().show()
```

```
+-------+--------------------+-------------------+-------------------------+-------
|summary|COMMUNITY_AREA_NUMBER|COMMUNITY_AREA_NAME|PERCENT OF HOUSING CROWDED|PERCENT
+-------+--------------------+-------------------+-------------------------+-------
|  count|                  77|                 78|                        78|
|   mean|                39.0|               null|         4.920512820512823|
| stddev|  22.371857321197094|               null|           3.6589814413502|
|    min|                   1|       Albany Park|                       0.3|
|    25%|                  20|               null|                       2.3|
|    50%|                  39|               null|                       3.8|
|    75%|                  58|               null|                       6.8|
|    max|                  77|          Woodlawn|                      15.8|
+-------+--------------------+-------------------+-------------------------+-------
```

```
census.toPandas().shape
```

```
(78, 9)
```

```
census.toPandas().isnull().sum()
```

```
COMMUNITY_AREA_NUMBER                           1
COMMUNITY_AREA_NAME                             0
PERCENT OF HOUSING CROWDED                      0
PERCENT HOUSEHOLDS BELOW POVERTY                0
PERCENT AGED 16+ UNEMPLOYED                     0
PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA    0
PERCENT AGED UNDER 18 OR OVER 64                0
PER_CAPITA_INCOME                               0
HARDSHIP_INDEX                                  1
dtype: int64
```

```
census.createOrReplaceTempView('censusDataSQL')
```

Q4. What recoreds or record has fount with no community area number in census data base? No name and No community area name has found.

```
spark.sql('select * \
          from censusDataSQL \
          where COMMUNITY_AREA_NUMBER IS NULL').show()
```

```
+---------------------+-------------------+--------------------------+--------------
|COMMUNITY_AREA_NUMBER|COMMUNITY_AREA_NAME|PERCENT OF HOUSING CROWDED|PERCENT HOUSEHO
+---------------------+-------------------+--------------------------+--------------
|                 null|            CHICAGO|                       4.7|
+---------------------+-------------------+--------------------------+--------------
```

```
spark.sql('select * \
          from censusDataSQL \
          where HARDSHIP_INDEX  IS NULL').show()
```

```
+---------------------+-------------------+--------------------------+--------------
|COMMUNITY_AREA_NUMBER|COMMUNITY_AREA_NAME|PERCENT OF HOUSING CROWDED|PERCENT HOUSEHO
+---------------------+-------------------+--------------------------+--------------
|                 null|            CHICAGO|                       4.7|
+---------------------+-------------------+--------------------------+--------------
```

```
school.show(5)
```

```
+---------+-------------------+-----------------------------------+-----------------
|School ID|     NAME_OF_SCHOOL|Elementary, Middle, or High School|      Street Addre
+---------+-------------------+-----------------------------------+-----------------
|   610038|Abraham Lincoln E...|                                 ES|    615 W Kemper P
|   610281|Adam Clayton Powe...|                                ES|7511 S South Shor
|   610185|Adlai E Stevenson...|                                ES| 8010 S Kostner Av
|   609993|Agustin Lara Elem...|                                ES| 4619 S Wolcott Av
|   610513|Air Force Academy...|                                HS|    3630 S Wells S
+---------+--------------------+----------------------------------+-----------------
only showing top 5 rows
```

```
# Droping the row wich will not affect the other rows.

census=census.na.drop()

```

```
# checking wich records have the null values?
census.toPandas().isnull().sum()
```

```
COMMUNITY_AREA_NUMBER                          0
COMMUNITY_AREA_NAME                            0
PERCENT OF HOUSING CROWDED                     0
PERCENT HOUSEHOLDS BELOW POVERTY               0
PERCENT AGED 16+ UNEMPLOYED                    0
PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA   0
PERCENT AGED UNDER 18 OR OVER 64               0
PER_CAPITA_INCOME                              0
HARDSHIP_INDEX                                 0
dtype: int64
```

```
census.toPandas().shape
```

```
(77, 9)
```

```
# Checking the value data types in our data. In order to fir the data

census.toPandas().dtypes
```

```
COMMUNITY_AREA_NUMBER                            int32
COMMUNITY_AREA_NAME                             object
PERCENT OF HOUSING CROWDED                     float64
PERCENT HOUSEHOLDS BELOW POVERTY               float64
PERCENT AGED 16+ UNEMPLOYED                    float64
PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA   float64
PERCENT AGED UNDER 18 OR OVER 64               float64
PER_CAPITA_INCOME                                int32
HARDSHIP_INDEX                                   int32
dtype: object
```

```
# Sine we have a unique number for the community area name we do not r

census=census.drop(census.COMMUNITY_AREA_NAME)
```

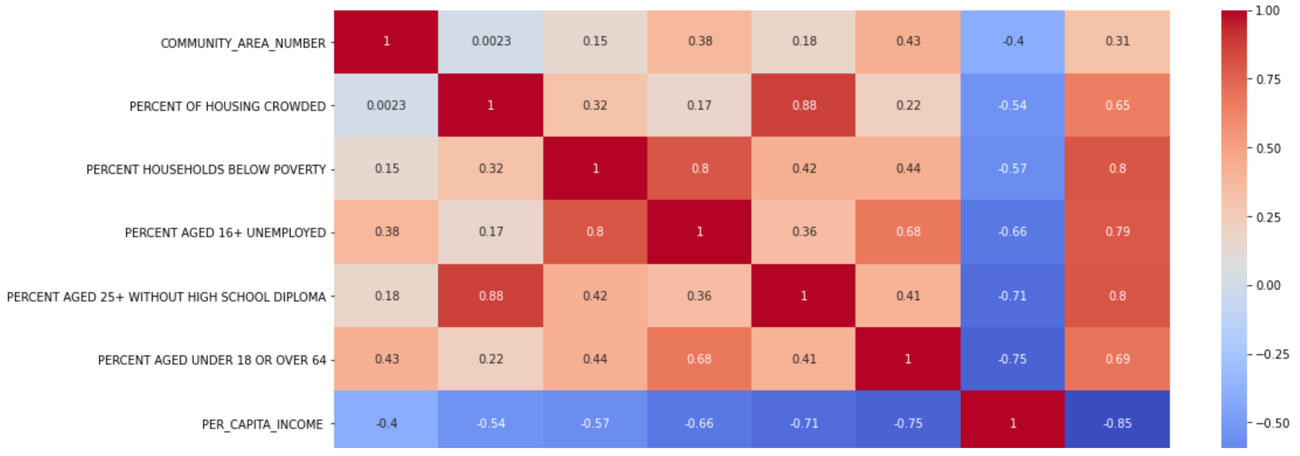
```
census.toPandas().dtypes
census.toPandas().dtypes
```

```
COMMUNITY_AREA_NUMBER                            int32
PERCENT OF HOUSING CROWDED                     float64
PERCENT HOUSEHOLDS BELOW POVERTY               float64
PERCENT AGED 16+ UNEMPLOYED                    float64
PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA   float64
PERCENT AGED UNDER 18 OR OVER 64               float64
PER_CAPITA_INCOME                                int32
HARDSHIP_INDEX                                   int32
dtype: object
```

With a quick glance we can have an idea that which variabls may be more effective for the deep finding which variabls have more stronger coefficient we need to feature extraction process, which I will impliment those methods later on this project.

```
import seaborn as sns
import matplotlib.pyplot as plt
% matplotlib inline
plt.figure(figsize=(16,8))
sns.heatmap(census.toPandas().corr(),annot=True,cmap='coolwarm')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8bfdab0b70>
```



```
census.show(2)
```

```
+---------------------+--------------------------+--------------------------------+--
|COMMUNITY_AREA_NUMBER|PERCENT OF HOUSING CROWDED|PERCENT HOUSEHOLDS BELOW POVERTY|PE
+---------------------+--------------------------+--------------------------------+--
|                    1|                       7.7|                            23.6|
|                    2|                       7.8|                            17.2|
+---------------------+--------------------------+--------------------------------+--
only showing top 2 rows
```

```
y=census.select('PER_CAPITA_INCOME ').collect()
```

```
X=census.drop('PER_CAPITA_INCOME ').collect()
```

```
census.columns
```

```
['COMMUNITY_AREA_NUMBER',
 'PERCENT OF HOUSING CROWDED',
 'PERCENT HOUSEHOLDS BELOW POVERTY',
 'PERCENT AGED 16+ UNEMPLOYED',
 'PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA',
 'PERCENT AGED UNDER 18 OR OVER 64',
 'PER_CAPITA_INCOME ',
 'HARDSHIP_INDEX']
```

Feature Selection and Extraction to find the most effective explantory variable in our data. Here I am using ExtraTreesClassifier to find out and later on I wil implimenting more effective methods to do so.

```
from sklearn.ensemble import ExtraTreesClassifier
```

```
model=ExtraTreesClassifier()
```

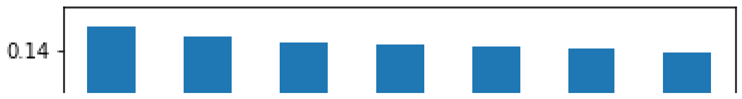
```
model.fit(X,y)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:1: DataConversionWarning
  """Entry point for launching an IPython kernel.
ExtraTreesClassifier(bootstrap=False, ccp_alpha=0.0, class_weight=None,
                     criterion='gini', max_depth=None, max_features='auto',
                     max_leaf_nodes=None, max_samples=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, n_estimators=100,
                     n_jobs=None, oob_score=False, random_state=None, verbose=0,
                     warm_start=False)
```

```
import pandas as pd
feat_importance=pd.Series(model.feature_importances_,index=['COMMUNITY
 'PERCENT OF HOUSING CROWDED',
 'PERCENT HOUSEHOLDS BELOW POVERTY',
 'PERCENT AGED 16+ UNEMPLOYED',
 'PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA',
 'PERCENT AGED UNDER 18 OR OVER 64',
 'HARDSHIP_INDEX'])
```

```
feat_importance.nlargest(7).plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8bf4168d68>
```



in addition to the correlation plot I tested to which variables should

I keep and importance feature testing and it turns out that the most variables

are highly correlated each other.Therfore, I will keep the rest of the

variables in this data.

```
school.show(10)
```

```
+---------+--------------------+----------------------------------+------------------
|School ID|      NAME_OF_SCHOOL|Elementary, Middle, or High School|      Street Addre
+---------+--------------------+----------------------------------+------------------
|   610038|Abraham Lincoln E...|                                ES|    615 W Kemper P
|   610281|Adam Clayton Powe...|                                ES|7511 S South Shor
|   610185|Adlai E Stevenson...|                                ES| 8010 S Kostner Av
|   609993|Agustin Lara Elem...|                                ES| 4619 S Wolcott Av
|   610513|Air Force Academy...|                                HS|    3630 S Wells S
|   610212|Albany Park Multi...|                                MS|  4929 N Sawyer Av
|   609720|Albert G Lane Tec...|                                HS|  2501 W Addison S
|   610342|Albert R Sabin El...|                                ES|   2216 W Hirsch S
|   610524|Alcott High Schoo...|                                HS|   2957 N Hoyne Av
|   610209|Alessandro Volta ...|                                ES|   4950 N Avers Av
+---------+--------------------+----------------------------------+------------------
only showing top 10 rows
```

```
# More SQL
school.createOrReplaceTempView('SchoolDataSQL')
```

```
school.toPandas().shape
```

```
(566, 78)
```

Finding whic variables in our data is object type, as we see there

```
school.dtypes
```

```
('Family Involvement Score', 'string'),
('Environment Icon ', 'string'),
('Environment Score', 'int'),
('Instruction Icon ', 'string'),
('Instruction Score', 'int'),
('Leaders Icon ', 'string'),
('Leaders Score ', 'string'),
('Teachers Icon ', 'string'),
('Teachers Score', 'string'),
('Parent Engagement Icon ', 'string'),
('Parent Engagement Score', 'string'),
('Parent Environment Icon', 'string'),
('Parent Environment Score', 'string'),
('AVERAGE_STUDENT_ATTENDANCE', 'string'),
('Rate of Misconducts (per 100 students) ', 'double'),
('Average Teacher Attendance', 'string'),
('Individualized Education Program Compliance Rate ', 'string'),
('Pk-2 Literacy %', 'string'),
('Pk-2 Math %', 'string'),
('Gr3-5 Grade Level Math %', 'string'),
('Gr3-5 Grade Level Read % ', 'string'),
('Gr3-5 Keep Pace Read %', 'string'),
('Gr3-5 Keep Pace Math %', 'string'),
('Gr6-8 Grade Level Math %', 'string'),
('Gr6-8 Grade Level Read %', 'string'),
('Gr6-8 Keep Pace Math%', 'string'),
('Gr6-8 Keep Pace Read %', 'string'),
('Gr-8 Explore Math %', 'string'),
('Gr-8 Explore Read %', 'string'),
('ISAT Exceeding Math %', 'double'),
('ISAT Exceeding Reading % ', 'double'),
('ISAT Value Add Math', 'double'),
('ISAT Value Add Read', 'double'),
('ISAT Value Add Color Math', 'string'),
('ISAT Value Add Color Read', 'string'),
('Students Taking  Algebra %', 'string'),
('Students Passing  Algebra %', 'string'),

('9th Grade EXPLORE (2009) ', 'string'),
('9th Grade EXPLORE (2010) ', 'string'),
('10th Grade PLAN (2009) ', 'string'),
('10th Grade PLAN (2010) ', 'string'),
('Net Change EXPLORE and PLAN', 'string'),
('11th Grade Average ACT (2011) ', 'string'),
('Net Change PLAN and ACT', 'string'),
('College Eligibility %', 'string'),
('Graduation Rate %', 'string'),
('College Enrollment Rate %', 'string'),
('COLLEGE_ENROLLMENT', 'int'),
('General Services Route ', 'int'),
('Freshman on Track Rate %', 'string'),
('X_COORDINATE', 'double'),
('Y_COORDINATE', 'double'),
('Latitude', 'double'),
('Longitude', 'double'),
('COMMUNITY_AREA_NUMBER', 'int'),
('COMMUNITY_AREA_NAME', 'string'),
('Ward', 'int'),
('Ward', 'int'),
('Police District', 'int'),
('Location', 'string')]
```

Looping through the school data set to find the String Objects.

```
# the columns which are string and not numeric
c=0
for i,j in school.dtypes:
  if j=='string':
    c+=1
    print(i)

print('-----------------------------------------------------------------
print('\n')

print(f'There are {c} categorical variable in school data set')

```

```
State
Phone Number
Link
Network Manager
Collaborative Name
Adequate Yearly Progress Made?
Track Schedule
CPS Performance Policy Status
CPS Performance Policy Level
HEALTHY_SCHOOL_CERTIFIED
Safety Icon
Family Involvement Icon
Family Involvement Score
Environment Icon
Instruction Icon
Leaders Icon
Leaders Score
Teachers Icon
Teachers Score
Parent Engagement Icon
Parent Engagement Score
Parent Environment Icon
Parent Environment Score
AVERAGE_STUDENT_ATTENDANCE
Average Teacher Attendance
Individualized Education Program Compliance Rate
Pk-2 Literacy %
Pk-2 Math %
Gr3-5 Grade Level Math %
Gr3-5 Grade Level Read %
Gr3-5 Keep Pace Read %
Gr3-5 Keep Pace Math %
Gr6-8 Grade Level Math %
Gr6-8 Grade Level Read %
Gr6-8 Keep Pace Math%
Gr6-8 Keep Pace Read %
Gr-8 Explore Math %
```

```
Gr-8 Explore Math %
Gr-8 Explore Read %
ISAT Value Add Color Math
ISAT Value Add Color Read
Students Taking  Algebra %
Students Passing  Algebra %
9th Grade EXPLORE (2009)
9th Grade EXPLORE (2010)
10th Grade PLAN (2009)
10th Grade PLAN (2010)
Net Change EXPLORE and PLAN
11th Grade Average ACT (2011)
Net Change PLAN and ACT
College Eligibility %
Graduation Rate %
College Enrollment Rate %
Freshman on Track Rate %
COMMUNITY_AREA_NAME
Location
---------------------------------------------------------------------------------
```

```
1  school.show(10)
```

```
+---------+--------------------+----------------------------------+-----------------
|School ID|      NAME_OF_SCHOOL|Elementary, Middle, or High School|      Street Addre
+---------+--------------------+----------------------------------+-----------------
|   610038|Abraham Lincoln E...|                                ES|    615 W Kemper P
|   610281|Adam Clayton Powe...|                                ES|7511 S South Shor
|   610185|Adlai E Stevenson...|                                ES| 8010 S Kostner Av
|   609993|Agustin Lara Elem...|                                ES| 4619 S Wolcott Av
|   610513|Air Force Academy...|                                HS|    3630 S Wells S
|   610212|Albany Park Multi...|                                MS|  4929 N Sawyer Av
|   609720|Albert G Lane Tec...|                                HS|  2501 W Addison S
|   610342|Albert R Sabin El...|                                ES|   2216 W Hirsch S
|   610524|Alcott High Schoo...|                                HS|   2957 N Hoyne Av
|   610209|Alessandro Volta ...|                                ES|   4950 N Avers Av
+---------+--------------------+----------------------------------+-----------------
only showing top 10 rows
```

There are too many variable in school data sets that shoud be drop becasue many of them have the same informatio, for example since we know that this data set is for Chicago city and IL state, so we do not need to repeat same information again. For example, City,State are the same thing and also as we have a unique id's for the school, so we do not need the name of the school in our data.

Also, there are many more variables that I think not having usefull information and they are listed below.

```
school=school.drop('NAME_OF_SCHOOL','Street Address','City','State','F
                  'CPS Performance Policy Level','Safety Icon','Family Invol
                  'Instruction Icon','Leaders Icon','Teachers Icon','Parent
```

```
school.show()
```

```
+---------+-----------------------------------+--------+--------------------+--------
|School ID|Elementary, Middle, or High School|ZIP Code|                Link |     Netw
+---------+-----------------------------------+--------+--------------------+--------
|   610038|                                 ES|   60614|http://schoolrepo...|Fullerto
|   610281|                                 ES|   60649|http://schoolrepo...|Skyway E
|   610185|                                 ES|   60652|http://schoolrepo...|Midway E
|   609993|                                 ES|   60609|http://schoolrepo...|Pershing
|   610513|                                 HS|   60609|http://schoolrepo...|Southwest
|   610212|                                 MS|   60625|http://schoolrepo...|O'Hare E
|   609720|                                 HS|   60618|http://schoolrepo...|North-No
|   610342|                                 ES|   60622|http://schoolrepo...|Fulton E
|   610524|                                 HS|   60618|http://schoolrepo...|North-No
|   610209|                                 ES|   60625|http://schoolrepo...|O'Hare E
|   609799|                                 ES|   60618|http://schoolrepo...|Ravenswo
|   609947|                                 ES|   60609|http://schoolrepo...|Pershing
|   609963|                                 ES|   60657|http://schoolrepo...|Ravenswo
|   610210|                                 ES|   60622|http://schoolrepo...|Fulton E
|   609808|                                 ES|   60628|http://schoolrepo...|Rock Isl
|   610028|                                 ES|   60628|http://schoolrepo...|Rock Isl
|   610098|                                 ES|   60651|http://schoolrepo...|Garfield
|   609788|                                 ES|   60643|http://schoolrepo...|Rock Isl
|   610334|                                 HS|   60624|http://schoolrepo...|West Sid
|   610131|                                 ES|   60608|http://schoolrepo...|Austin-N
+---------+-----------------------------------+--------+--------------------+--------
only showing top 20 rows
```

```
school=school.drop('Link ','Network Manager','CPS Performance Policy S
                  'Leaders Icon ','Parent Engagement Icon ')
```

```
school.show()
```

```
+---------+-----------------------------------+--------+-----------------------------
|School ID|Elementary, Middle, or High School|ZIP Code|Adequate Yearly Progress Made
+---------+-----------------------------------+--------+-----------------------------
|   610038|                                 ES|   60614|                            N
|   610281|                                 ES|   60649|                            N
|   610185|                                 ES|   60652|                            N
|   609993|                                 ES|   60609|                            N
|   610513|                                 HS|   60609|                           NI
|   610212|                                 MS|   60625|                           Ye
|   609720|                                HS|   60618|                            Ye
|   610342|                                ES|   60622|                             N
|   610524|                                HS|   60618|                            ND
|   610209|                                ES|   60625|                             N
|   609799|                                ES|   60618|                             N
|   609947|                                ES|   60609|                             N
|   609963|                                ES|   60657|                             N
|   610210|                                ES|   60622|                             N
|   609808|                                ES|   60628|                             N
|   610028|                                ES|   60628|                             N
|   610098|                                ES|   60651|                             N
|   609788|                                ES|   60643|                             N
|   610334|                                HS|   60624|                             N
|   610131|                                ES|   60608|                             N
+---------+----------------------------------+--------+------------------------------
only showing top 20 rows
```

```
c=0
for i,j in school.dtypes:
  if j=='string':
    c+=1
    print(i)

print()
print('------------------------------------------------------->>>')
print('The categorical variables reduced to ------->', c)
```

```
Elementary, Middle, or High School
Adequate Yearly Progress Made?
Track Schedule
HEALTHY_SCHOOL_CERTIFIED
Family Involvement Score
Leaders Score
Teachers Icon
Teachers Score
Parent Engagement Score
Parent Environment Score
AVERAGE_STUDENT_ATTENDANCE
Average Teacher Attendance
Individualized Education Program Compliance Rate
Pk-2 Literacy %
Pk-2 Math %
Gr3-5 Grade Level Math %
Gr3-5 Grade Level Read %
Gr3-5 Keep Pace Read %
Gr3-5 Keep Pace Math %
Gr6-8 Grade Level Math %
Gr6-8 Grade Level Read %
Gr6-8 Keep Pace Math%
Gr6-8 Keep Pace Read %
Gr-8 Explore Math %
Gr-8 Explore Read %
ISAT Value Add Color Math
ISAT Value Add Color Read
Students Taking  Algebra %
Students Passing  Algebra %
9th Grade EXPLORE (2009)
9th Grade EXPLORE (2010)
10th Grade PLAN (2009)
10th Grade PLAN (2010)
Net Change EXPLORE and PLAN
11th Grade Average ACT (2011)
Net Change PLAN and ACT
College Eligibility %
Graduation Rate %
College Enrollment Rate %
Freshman on Track Rate %

-------------------------------------------------->>>
The categorical variables reduced to -------> 40
```

Choosing the "Adequate Yearly Progress Made?" variable as a target variables could lead us to answer this reserch question that *which schoo did have very well progress among the all schools in Chicao?*

```
## chosing ==============> Adequate Yearly Progress Made? yes,no as a
```

```
school.select('Adequate Yearly Progress Made? ').show()
```

```
+-------------------------------+
|Adequate Yearly Progress Made? |
+-------------------------------+
|                             No|
|                             No|
|                             No|
|                             No|
|                            NDA|
|                            Yes|
|                            Yes|
|                             No|
|                            NDA|
|                             No|
|                             No|
|                             No|
|                             No|
|                             No|
|                             No|
|                             No|
|                             No|
|                             No|
|                             No|
|                             No|
+-------------------------------+
only showing top 20 rows
```

```
school=school.withColumnRenamed('Adequate Yearly Progress Made? ','Tar
```

```
# from now the Adequate Yearly Progress Made? variable name has change

school.select('Target').show()
```

```
+-------+
|Target|
+-------+
|    No|
|    No|
|    No|
|    No|
|   NDA|
|   Yes|
|   Yes|
|    No|
|   NDA|
|    No|
|    No|
|    No|
|    No|
|    No|
|    No|
|    No|
|    No|
|    No|
|    No|
|    No|
+-------+
only showing top 20 rows
```

```
school.select('Target').toPandas().value_counts()
```

```
Target
No        476
Yes        72
NDA        18
dtype: int64
```

```
school.select('Target').toPandas().describe()
```

	Target
count	566
unique	3
top	No
freq	476

As we see the most frequent variable is our data has responded

```
school.select('Target').toPandas().value_counts().to_dict()
```

```
{('NDA',): 18, ('No',): 476, ('Yes',): 72}
```

```
school.show(10)

```

```
+---------+--------------------------------+--------+------+--------------+-------
|School ID|Elementary, Middle, or High School|ZIP Code|Target|Track Schedule|HEALTHY_
+---------+--------------------------------+--------+------+--------------+-------
|   610038|                              ES|   60614|    No|      Standard|
|   610281|                              ES|   60649|    No|       Track_E|
|   610185|                              ES|   60652|    No|      Standard|
|   609993|                              ES|   60609|    No|       Track_E|
|   610513|                              HS|   60609|   NDA|      Standard|
|   610212|                              MS|   60625|   Yes|      Standard|
|   609720|                              HS|   60618|   Yes|      Standard|
|   610342|                              ES|   60622|    No|      Standard|
|   610524|                              HS|   60618|   NDA|      Standard|
|   610209|                              ES|   60625|    No|       Track_E|
+---------+--------------------------------+--------+------+--------------+-------
only showing top 10 rows
```

```
pip install pyjanitor
```

```
Requirement already satisfied: more-itertools>=4.0.0 in /usr/local/lib/python3.6/
Requirement already satisfied: atomicwrites>=1.0 in /usr/local/lib/python3.6/dist
Requirement already satisfied: virtualenv>=20.0.8 in /usr/local/lib/python3.6/dis
Requirement already satisfied: importlib-metadata; python_version < "3.8" in /usr
Requirement already satisfied: nodeenv>=0.11.1 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: importlib-resources; python_version < "3.7" in /us
Requirement already satisfied: identify>=1.0.0 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: cfgv>=2.0.0 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: nbconvert!=5.4 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: nbformat in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: docutils in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: jinja2 in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: sphinx>=1.8 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: ipython>=4.0.0 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: coverage>=4.4 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: sortedcontainers<3.0.0,>=2.1.0 in /usr/local/lib/p
Requirement already satisfied: decorator in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: ipython-genutils in /usr/local/lib/python3.6/dist-
Requirement already satisfied: pyflakes<2.3.0,>=2.2.0 in /usr/local/lib/python3.6
Requirement already satisfied: mccabe<0.7.0,>=0.6.0 in /usr/local/lib/python3.6/d
Requirement already satisfied: pycodestyle<2.7.0,>=2.6.0a1 in /usr/local/lib/pyth
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/l
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: filelock<4,>=3.0.0 in /usr/local/lib/python3.6/dis
Requirement already satisfied: distlib<1,>=0.3.1 in /usr/local/lib/python3.6/dist
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: testpath in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: bleach in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.6/d
Requirement already satisfied: defusedxml in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.6/dist
Requirement already satisfied: pygments in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.6/dis
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in /usr/local/lib/python3.

Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: imagesize in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: requests>=2.0.0 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: alabaster<0.8,>=0.7 in /usr/local/lib/python3.6/di
Requirement already satisfied: babel!=2.0,>=1.3 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: sphinxcontrib-websupport in /usr/local/lib/python3
Requirement already satisfied: packaging in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: snowballstemmer>=1.1 in /usr/local/lib/python3.6/d
Requirement already satisfied: pexpect; sys_platform != "win32" in /usr/local/lib
Requirement already satisfied: prompt-toolkit<2.0.0,>=1.0.4 in /usr/local/lib/pyt
Requirement already satisfied: simplegeneric>0.8 in /usr/local/lib/python3.6/dist
Requirement already satisfied: pickleshare in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: webencodings in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dis
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/lo
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: sphinxcontrib-serializinghtml in /usr/local/lib/py
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: wcwidth in /usr/local/lib/python3.6/dist-packages
```

```python
from pyspark.sql import DataFrame
import janitor.spark
```

```
/usr/local/lib/python3.6/dist-packages/distributed/config.py:20: YAMLLoadWarning: ca
  defaults = yaml.load(f)
```

Cleaning the data sets are the most important task. In these data sets all the variables names and contents are not properly provided. There are many issues that can affect the queries, so let do cleaning task on both data sets.

```python
school=school.clean_names()
```

```python
school.show(5)
```

```
+---------+------------------------------+--------+------+--------------+---------
|school_id|elementary_middle_or_high_school|zip_code|target|track_schedule|healthy_sc
+---------+------------------------------+--------+------+--------------+---------
```

```
|   610038|                               ES|   60614|    No|      Standard|
|   610281|                               ES|   60649|    No|       Track_E|
|   610185|                               ES|   60652|    No|      Standard|
|   609993|                               ES|   60609|    No|       Track_E|
|   610513|                               HS|   60609|   NDA|      Standard|
+---------+-------------------------------+--------+------+-------------+---------
only showing top 5 rows
```

```
school.select('elementary_middle_or_high_school').toPandas().value_cou
```

```
elementary_middle_or_high_school
ES                                  462
HS                                   93
MS                                   11
dtype: int64
```

```
from pyspark.ml.feature import StringIndexer,VectorAssembler,OneHotEnc
from pyspark.sql.functions import when,col

```

```
school.toPandas().shape
```

```
(566, 59)
```

```
census.printSchema()
```

```
root
 |-- COMMUNITY_AREA_NUMBER: integer (nullable = true)
 |-- PERCENT OF HOUSING CROWDED: double (nullable = true)
 |-- PERCENT HOUSEHOLDS BELOW POVERTY: double (nullable = true)
 |-- PERCENT AGED 16+ UNEMPLOYED: double (nullable = true)
 |-- PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA: double (nullable = true)
 |-- PERCENT AGED UNDER 18 OR OVER 64: double (nullable = true)
 |-- PER_CAPITA_INCOME : integer (nullable = true)
 |-- HARDSHIP_INDEX: integer (nullable = true)
```

```
census=census.clean_names()
```

```
census.show(10)
```

```
+---------------------+--------------------------+--------------------------------+--
|community_area_number|percent_of_housing_crowded|percent_households_below_poverty|pe
+---------------------+--------------------------+--------------------------------+--
|                    1|                       7.7|                            23.6|
|                    2|                       7.8|                            17.2|
|                    3|                       3.8|                            24.0|
|                    4|                       3.4|                            10.9|
|                    5|                       0.3|                             7.5|
|                    6|                       1.1|                            11.4|
|                    7|                       0.8|                            12.3|
|                   8|                       1.9|                               12.9|
|                   9|                       1.1|                                3.3|
|                  10|                       2.0|                                5.4|
+--------------------+--------------------------+-----------------------------------+-
only showing top 10 rows
```

```
# Saving schools as s temp and census as c temp for more sql queries
census.createOrReplaceTempView('c')
school.createOrReplaceTempView('s')
```

```
school.show(5)
```

```
+---------+------------------------------+-------+------+--------------+---------
|school_id|elementary_middle_or_high_school|zip_code|target|track_schedule|healthy_sc
+---------+------------------------------+-------+------+--------------+---------
|   610038|                             ES|   60614|    No|      Standard|
|   610281|                             ES|   60649|    No|       Track_E|
|   610185|                             ES|   60652|    No|      Standard|
|   609993|                             ES|   60609|    No|       Track_E|
|   610513|                             HS|   60609|   NDA|      Standard|
+---------+------------------------------+-------+------+--------------+---------
only showing top 5 rows
```

```
# Finding which columns are common between two data sets.
set(school.columns).intersection(set(census.columns))
```

```
{'community_area_number'}
```

```
census.toPandas().shape
```

```
(77, 8)
```

```
# Joining two data sets based on common column
data=school.join(census,on='community_area_number',how='inner').distir
```

```
data.printSchema()
```

```
root
 |-- community_area_number: integer (nullable = true)
 |-- school_id: integer (nullable = true)
 |-- elementary_middle_or_high_school: string (nullable = true)
 |-- zip_code: integer (nullable = true)
 |-- target: string (nullable = true)
 |-- track_schedule: string (nullable = true)
 |-- healthy_school_certified: string (nullable = true)
 |-- safety_score: integer (nullable = true)
 |-- family_involvement_score: string (nullable = true)
 |-- environment_score: integer (nullable = true)
 |-- instruction_score: integer (nullable = true)
 |-- leaders_score_: string (nullable = true)
 |-- teachers_icon_: string (nullable = true)
 |-- teachers_score: string (nullable = true)
 |-- parent_engagement_score: string (nullable = true)
 |-- parent_environment_score: string (nullable = true)
 |-- average_student_attendance: string (nullable = true)
 |-- rate_of_misconducts_per_100_students_: double (nullable = true)
 |-- average_teacher_attendance: string (nullable = true)
 |-- individualized_education_program_compliance_rate_: string (nullable = true)
 |-- pk_2_literacy_%: string (nullable = true)
 |-- pk_2_math_%: string (nullable = true)
 |-- gr3_5_grade_level_math_%: string (nullable = true)
 |-- gr3_5_grade_level_read_%_: string (nullable = true)
 |-- gr3_5_keep_pace_read_%: string (nullable = true)
 |-- gr3_5_keep_pace_math_%: string (nullable = true)
 |-- gr6_8_grade_level_math_%: string (nullable = true)
 |-- gr6_8_grade_level_read_%: string (nullable = true)
 |-- gr6_8_keep_pace_math%: string (nullable = true)
 |-- gr6_8_keep_pace_read_%: string (nullable = true)
 |-- gr_8_explore_math_%: string (nullable = true)
 |-- gr_8_explore_read_%: string (nullable = true)
 |-- isat_exceeding_math_%: double (nullable = true)
 |-- isat_exceeding_reading_%_: double (nullable = true)
 |-- isat_value_add_math: double (nullable = true)
 |-- isat_value_add_read: double (nullable = true)
 |-- isat_value_add_color_math: string (nullable = true)
 |-- isat_value_add_color_read: string (nullable = true)
 |-- students_taking_algebra_%: string (nullable = true)
 |-- students_passing_algebra_%: string (nullable = true)
 |-- 9th_grade_explore_2009_: string (nullable = true)
 |-- 9th_grade_explore_2010_: string (nullable = true)
 |-- 10th_grade_plan_2009_: string (nullable = true)
 |-- 10th_grade_plan_2010_: string (nullable = true)
 |-- net_change_explore_and_plan: string (nullable = true)
 |-- 11th_grade_average_act_2011_: string (nullable = true)
 |-- net_change_plan_and_act: string (nullable = true)
 |-- college_eligibility_%: string (nullable = true)
 |-- graduation_rate_%: string (nullable = true)
 |-- college_enrollment_rate_%: string (nullable = true)
 |-- college_enrollment: integer (nullable = true)
 |-- general_services_route_: integer (nullable = true)
 |-- freshman_on_track_rate_%: string (nullable = true)
 |-- x_coordinate: double (nullable = true)
 |-- y_coordinate: double (nullable = true)
 |-- latitude: double (nullable = true)
 |-- longitude: double (nullable = true)
 |-- ward: integer (nullable = true)
```

```
data.limit(5).show()
```

```
+---------------------+---------+--------------------------------+--------+------+---
|community_area_number|school_id|elementary_middle_or_high_school|zip_code|target|tra
+---------------------+---------+--------------------------------+--------+------+---
|                    4|   609852|                              ES|   60625|   Yes|
|                   10|   609937|                              ES|   60656|    No|
|                   24|   609828|                              ES|   60622|   Yes|
|                    6|   610355|                              ES|   60613|   Yes|
|                   61|   609929|                              ES|   60609|    No|
+--------------------+---------+--------------------------------+--------+------+--
```

```
# droping the na values from the data
data=data.na.drop()
```

```
data.toPandas().shape
```

```
(436, 66)
```

```
# Since this col, does not give much information, dropping it.
data=data.drop('teachers_icon_')
```

```
data.show(5)
```

```
+--------------------+---------+--------------------------------+--------+------+--
|community_area_number|school_id|elementary_middle_or_high_school|zip_code|target|tra
+--------------------+---------+--------------------------------+--------+------+--
|                   4|   609852|                              ES|   60625|   Yes|
|                  10|   609937|                              ES|   60656|    No|
|                  24|   609828|                              ES|   60622|   Yes|
|                   6|   610355|                              ES|   60613|   Yes|
|                  61|   609929|                              ES|   60609|    No|
+--------------------+---------+--------------------------------+--------+------+--
only showing top 5 rows
```

```
data.describe().show()
```

```
+-------+---------------------+------------------+--------------------------------+--
|summary|community_area_number|         school_id|elementary_middle_or_high_school|
+-------+---------------------+------------------+--------------------------------+--
|  count|                  436|               436|                             436|
|   mean|    38.41743119266055| 610068.1880733945|                            null|6(
| stddev|    21.84027848770573|180.80035714785598|                            null|2:
|    min|                    1|            609725|                              ES|
|    max|                   77|            610544|                              MS|
+-------+---------------------+------------------+--------------------------------+--
```

Counting the rows with the respect of their columns ti find out how many NDA's are in our table. NOTE: The NDA gives us NO information. Let find out!

```
from pyspark.sql.functions import count
```

```
for i in data.columns:
  print(data.filter(data[i]=='NDA').count(),'========>>   ',data[i],"=
#data.filter(data['students_taking_algebra_%']=='NDA')
```

```
0 ========>>    Column<b'healthy_school_certified'> ==========>  Have NDA's
0 ========>>    Column<b'safety_score'> ==========>  Have NDA's
204 ========>>    Column<b'family_involvement_score'> ==========>  Have NDA's
0 ========>>    Column<b'environment_score'> ==========>  Have NDA's
0 ========>>    Column<b'instruction_score'> ==========>  Have NDA's
206 ========>>    Column<b'leaders_score_'> ==========>  Have NDA's
206 ========>>    Column<b'teachers_score'> ==========>  Have NDA's
78 ========>>    Column<b'parent_engagement_score'> ==========>  Have NDA's
78 ========>>    Column<b'parent_environment_score'> ==========>  Have NDA's
0 ========>>    Column<b'average_student_attendance'> ==========>  Have NDA's
0 ========>>    Column<b'rate_of_misconducts_per_100_students_'> ==========>  Hav
0 ========>>    Column<b'average_teacher_attendance'> ==========>  Have NDA's
0 ========>>    Column<b'individualized_education_program_compliance_rate_'> ====
68 ========>>    Column<b'pk_2_literacy_%'> ==========>  Have NDA's
130 ========>>    Column<b'pk_2_math_%'> ==========>  Have NDA's
22 ========>>    Column<b'gr3_5_grade_level_math_%'> ==========>  Have NDA's
22 ========>>    Column<b'gr3_5_grade_level_read_%_'> ==========>  Have NDA's
22 ========>>    Column<b'gr3_5_keep_pace_read_%'> ==========>  Have NDA's
22 ========>>    Column<b'gr3_5_keep_pace_math_%'> ==========>  Have NDA's
9 ========>>    Column<b'gr6_8_grade_level_math_%'> ==========>  Have NDA's
8 ========>>    Column<b'gr6_8_grade_level_read_%'> ==========>  Have NDA's
9 ========>>    Column<b'gr6_8_keep_pace_math%'> ==========>  Have NDA's
8 ========>>    Column<b'gr6_8_keep_pace_read_%'> ==========>  Have NDA's
27 ========>>    Column<b'gr_8_explore_math_%'> ==========>  Have NDA's
27 ========>>    Column<b'gr_8_explore_read_%'> ==========>  Have NDA's
0 ========>>    Column<b'isat_exceeding_math_%'> ==========>  Have NDA's
0 ========>>    Column<b'isat_exceeding_reading_%_'> ==========>  Have NDA's
0 ========>>    Column<b'isat_value_add_math'> ==========>  Have NDA's
0 ========>>    Column<b'isat_value_add_read'> ==========>  Have NDA's
0 ========>>    Column<b'isat_value_add_color_math'> ==========>  Have NDA's
0 ========>>    Column<b'isat_value_add_color_read'> ==========>  Have NDA's
284 ========>>    Column<b'students_taking_algebra_%'> ==========>  Have NDA's
312 ========>>    Column<b'students_passing_algebra_%'> ==========>  Have NDA's
427 ========>>    Column<b'9th_grade_explore_2009_'> ==========>  Have NDA's
428 ========>>    Column<b'9th_grade_explore_2010_'> ==========>  Have NDA's
429 ========>>    Column<b'10th_grade_plan_2009_'> ==========>  Have NDA's
428 ========>>    Column<b'10th_grade_plan_2010_'> ==========>  Have NDA's
428 ========>>    Column<b'net_change_explore_and_plan'> ==========>  Have NDA's
429 ========>>    Column<b'11th_grade_average_act_2011_'> ==========>  Have NDA's
429 ========>>    Column<b'net_change_plan_and_act'> ==========>  Have NDA's
429 ========>>    Column<b'college_eligibility_%'> ==========>  Have NDA's
430 ========>>    Column<b'graduation_rate_%'> ==========>  Have NDA's
430 ========>>    Column<b'college_enrollment_rate_%'> ==========>  Have NDA's
0 ========>>    Column<b'college_enrollment'> ==========>  Have NDA's
0 ========>>    Column<b'general_services_route_'> ==========>  Have NDA's
428 ========>>    Column<b'freshman_on_track_rate_%'> ==========>  Have NDA's
0 ========>>    Column<b'x_coordinate'> ==========>  Have NDA's
0 ========>>    Column<b'y_coordinate'> ==========>  Have NDA's

0 ========>>    Column<b'latitude'> ==========>  Have NDA's
0 ========>>    Column<b'longitude'> ==========>  Have NDA's
0 ========>>    Column<b'ward'> ==========>  Have NDA's
0 ========>>    Column<b'police_district'> ==========>  Have NDA's
0 ========>>    Column<b'percent_of_housing_crowded'> ==========>  Have NDA's
0 ========>>    Column<b'percent_households_below_poverty'> ==========>  Have NDA
0 ========>>    Column<b'percent_aged_16+_unemployed'> ==========>  Have NDA's
```

```
0 ========>>    Column<b'percent_aged_25+_without_high_school_diploma'> =========
0 ========>>    Column<b'percent_aged_under_18_or_over_64'> ==========>  Have NDA
0 ========>>    Column<b'per_capita_income_'> ==========>  Have NDA's
```

Since some columns have many NDA's it is better todrop them

Since does not give any usefull information

```
# these cols have many NDA as results of the above calculation and sir
to_be_dropped=['family_involvement_score','leaders_score_',
               'teachers_score','parent_engagement_score',
               'parent_environment_score','pk_2_literacy_%',
               'pk_2_math_%','gr3_5_grade_level_math_%',
               'gr3_5_grade_level_read_%_','gr3_5_keep_pace_read_%',
               'gr3_5_keep_pace_math_%','gr6_8_grade_level_math_%',
               'gr6_8_grade_level_read_%','gr6_8_keep_pace_math%',
               'gr6_8_keep_pace_read_%','gr_8_explore_math_%',
               'gr_8_explore_read_%','students_taking_algebra_%',
               'students_passing_algebra_%','9th_grade_explore_2009_',
               '9th_grade_explore_2010_','10th_grade_plan_2009_',
               '10th_grade_plan_2010_','net_change_explore_and_plan',
               '11th_grade_average_act_2011_','net_change_plan_and_act
               'college_eligibility_%','graduation_rate_%',
               'college_enrollment_rate_%','freshman_on_track_rate_%',
               ]
```

```
data_NDA_drop=data.drop(*to_be_dropped)
```

```
# Data reduced to 35 columns with the 436 records
data_NDA_drop.toPandas().shape
```

```
(436, 35)
```

```
len(data_NDA_drop.dtypes)
```

```
35
```

Ckecking out that how many catgorical variables are in our data

```
## These coulumns has the string type and they should be converted to
for i,j in data_NDA_drop.dtypes:
  if j=='string':
    print(i,'============> has a string type of a value should be nume
    print(i, '============> has a string type of a value should be nume
    print()
    print('---'*50)
```

```
elementary_middle_or_high_school ============> has a string type of a value should be

------------------------------------------------------------------------------------
target ============> has a string type of a value should be numeric type

------------------------------------------------------------------------------------
track_schedule ============> has a string type of a value should be numeric type

------------------------------------------------------------------------------------
healthy_school_certified ============> has a string type of a value should be numeric

------------------------------------------------------------------------------------
average_student_attendance ============> has a string type of a value should be numer

------------------------------------------------------------------------------------
average_teacher_attendance ============> has a string type of a value should be numer

------------------------------------------------------------------------------------
individualized_education_program_compliance_rate_ ============> has a string type of

------------------------------------------------------------------------------------
isat_value_add_color_math ============> has a string type of a value should be numeri

------------------------------------------------------------------------------------
isat_value_add_color_read ============> has a string type of a value should be numeri

------------------------------------------------------------------------------------
```

```
data_NDA_drop.select('elementary_middle_or_high_school').show()
```

```
+--------------------------------+
|elementary_middle_or_high_school|
+--------------------------------+
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
|                              HS|
|                              ES|
|                              ES|
|                              ES|
|                              MS|
|                              ES|
|                              ES|
|                              ES|
|                              ES|
+--------------------------------+
only showing top 20 rows
```

```
data_NDA_drop.select('elementary_middle_or_high_school').toPandas().va
```

```
elementary_middle_or_high_school
ES                                  417
MS                                   11
HS                                    8
dtype: int64
```

Encoding the categorical variable to dummies process doing with StringIndexer function from Mlib

```
indexer=StringIndexer(inputCol='elementary_middle_or_high_school',outp
```

```
indexed=indexer.fit(data_NDA_drop).transform(data_NDA_drop)
```

```
indexed.show(10)
```

```
+---------------------+---------+--------------------------------+--------+------+---
|community_area_number|school_id|elementary_middle_or_high_school|zip_code|target|tra
+---------------------+---------+--------------------------------+--------+------+---
|                    4|   609852|                              ES|   60625|   Yes|
|                   10|   609937|                              ES|   60656|    No|
|                   24|   609828|                              ES|   60622|   Yes|
|                    6|   610355|                              ES|   60613|   Yes|
|                   61|   609929|                              ES|   60609|    No|
|                   28|   610180|                              ES|   60608|    No|
|                   61|   610239|                              ES|   60609|    No|
|                   71|   609805|                              ES|   60620|    No|
|                   61|   610167|                              ES|   60609|    No|
|                   69|   609813|                              ES|   60637|    No|
+---------------------+---------+--------------------------------+--------+------+---
only showing top 10 rows
```

```
indexed=indexed.drop('elementary_middle_or_high_school')
```

```
indexed=indexed.withColumnRenamed('elementary_middle_or_high_school_ir
```

```
indexed.show(10)
```

```
+---------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+---------------------+---------+--------+------+--------------+--------------------
|                    4|   609852|   60625|   Yes|      Standard|
|                   10|   609937|   60656|    No|      Standard|
|                  24|   609828|   60622|   Yes|       Track_E|
|                   6|   610355|   60613|   Yes|      Standard|
|                  61|   609929|   60609|    No|       Track_E|
|                  28|   610180|   60608|    No|       Track_E|
|                  61|   610239|   60609|    No|       Track_E|
|                  71|   609805|   60620|    No|       Track_E|
|                  61|   610167|   60609|    No|      Standard|
|                  69|   609813|   60637|    No|       Track_E|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 10 rows
```

```
indexed.select('track_schedule').toPandas().value_counts()
```

```
track_schedule
Standard         237
Track_E          199
dtype: int64
```

```
indexer=StringIndexer(inputCol='track_schedule',outputCol='track_schec
```

```
indexed=indexer.fit(indexed).transform(indexed)
```

```
indexed.show(5)
```

```
+--------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+--------------------+---------+--------+------+--------------+--------------------
|                   4|   609852|   60625|   Yes|      Standard|
|                  10|   609937|   60656|    No|      Standard|
|                  24|   609828|   60622|   Yes|       Track_E|
|                   6|   610355|   60613|   Yes|      Standard|
|                  61|   609929|   60609|    No|       Track_E|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 5 rows
```

```
indexed=indexed.withColumn('track_schedule',indexed['track_schedule_ir
```

```
indexed.show(4)
indexed=indexed.drop('track_schedule_indexer')
indexed.show(4)
```

```
+--------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+--------------------+---------+--------+------+--------------+--------------------
|                   4|   609852|   60625|   Yes|           0.0|
|                  10|   609937|   60656|    No|           0.0|
|                  24|   609828|   60622|   Yes|           1.0|
|                   6|   610355|   60613|   Yes|           0.0|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 4 rows

+--------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+--------------------+---------+--------+------+--------------+--------------------
|                   4|   609852|   60625|   Yes|           0.0|
|                  10|   609937|   60656|    No|           0.0|
|                  24|   609828|   60622|   Yes|           1.0|
|                   6|   610355|   60613|   Yes|           0.0|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 4 rows
```

```
indexed.select('healthy_school_certified').toPandas().value_counts()
```

```
healthy_school_certified
No                         424
Yes                         12
dtype: int64
```

```
indexer=StringIndexer(inputCol='healthy_school_certified',outputCol='h
```

```
indexed=indexer.fit(indexed).transform(indexed)
```

```
indexed.show(50)
```

```
+--------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+--------------------+---------+--------+------+--------------+--------------------
|                   4|   609852|   60625|   Yes|           0.0|
|                  10|   609937|   60656|    No|           0.0|
|                  24|   609828|   60622|   Yes|           1.0|
|                   6|   610355|   60613|   Yes|           0.0|
|                  61|   609929|   60609|    No|           1.0|
|                  28|   610180|   60608|    No|           1.0|
|                  61|   610239|   60609|    No|           1.0|
|                  71|   609805|   60620|    No|           1.0|
|                  61|   610167|   60609|    No|           0.0|
|                  69|   609813|   60637|    No|           1.0|
|                  30|   609973|   60623|    No|           1.0|
|                  39|   609746|   60615|    No|           0.0|
|                  58|   610353|   60632|    No|           1.0|
|                  55|   609856|   60633|    No|           0.0|
|                  24|   610076|   60647|    No|           1.0|
|                  63|   610532|   60632|    No|           1.0|
|                  27|   610251|   60612|    No|           1.0|
|                   5|   610010|   60657|    No|           0.0|
|                  31|   610125|   60608|    No|           0.0|
|                  43|   610103|   60649|    No|           0.0|
|                  25|   610367|   60644|    No|           1.0|
|                  52|   610198|   60617|    No|           0.0|
|                  40|   609819|   60615|    No|           1.0|
|                  28|   609812|   60612|    No|           0.0|
|                  25|   610092|   60644|    No|           1.0|
|                  73|   610362|   60628|    No|           1.0|
|                  70|   609879|   60652|    No|           0.0|
|                  24|   610073|   60612|   Yes|           0.0|
|                  61|   609885|   60609|    No|           0.0|
|                  35|   610110|   60653|    No|           1.0|
|                  22|   610138|   60647|    No|           0.0|
|                  29|   609851|   60608|    No|           1.0|
|                   6|   609850|   60613|    No|           0.0|
|                  53|   610224|   60628|    No|           1.0|
|                  66|   610347|   60636|    No|           1.0|
|                  73|   610027|   60620|    No|           0.0|
|                  43|   609815|   60649|    No|           0.0|
|                   6|   609974|   60657|   Yes|           0.0|
|                  29|   610131|   60608|    No|           1.0|
|                  49|   610188|   60628|    No|           1.0|
|                  10|   609995|   60646|    No|           0.0|
|                  66|   610396|   60629|    No|           0.0|
|                  68|   610339|   60621|    No|           1.0|
|                  17|   609810|   60634|    No|           0.0|
|                  66|   610057|   60629|    No|           1.0|
|                  53|   610160|   60628|    No|           1.0|
|                  38|   609781|   60609|    No|           0.0|
|                  13|   610127|   60625|    No|           0.0|
|                  69|   610233|   60621|    No|           1.0|
|                  44|   610093|   60619|    No|           0.0|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 50 rows
```

```
indexed=indexed.withColumn('healthy_school_certified',indexed['healthy
indexed=indexed.drop('healthy_school_certified_indexer')
```

```
indexed.show(5)
```

```
+--------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+--------------------+---------+--------+------+--------------+--------------------
|                   4|   609852|   60625|   Yes|           0.0|
|                  10|   609937|   60656|    No|           0.0|
|                  24|   609828|   60622|   Yes|           1.0|
|                   6|   610355|   60613|   Yes|           0.0|
|                  61|   609929|   60609|    No|           1.0|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 5 rows
```

There are very inconvinienice number format with the % sign infront of them also they are string type that they should be numeric. Procedure to clean these columns as belowe

```
indexed.select('average_student_attendance').show()
```

```
+--------------------------+
|average_student_attendance|
+--------------------------+
|                    95.10%|
|                    95.30%|
|                    95.90%|
|                    94.90%|
|                    95.10%|
|                    92.40%|
|                    94.90%|
|                    95.00%|
|                    95.70%|
|                    92.80%|
|                    96.60%|
|                    88.40%|
|                    96.50%|
|                    95.20%|
|                    93.00%|
|                    95.50%|
|                    94.90%|
|                    94.70%|
|                    95.50%|
|                    92.10%|
+--------------------------+
only showing top 20 rows
```

```
from pyspark.sql.functions import format_number,format_string,split

```

Getting rid of the % percentage sign stuck to the number

```
indexed=indexed.withColumn('average_student_attendance',split(indexed|
```

```
indexed.select('average_student_attendance').show()
```

```
+--------------------------+
|average_student_attendance|
+--------------------------+
|                     95.10|
|                     95.30|
|                     95.90|
|                     94.90|
|                     95.10|
|                     92.40|
|                     94.90|
|                     95.00|
|                     95.70|
|                     92.80|
|                     96.60|
|                     88.40|
|                     96.50|
|                     95.20|
|                    93.00|
|                    95.50|
|                    94.90|
|                    94.70|
|                    95.50|
|                    92.10|
+-------------------------+
only showing top 20 rows
```

Casting the string format to float format

```
indexed=indexed.withColumn('average_student_attendance',indexed['avera
```

```
indexed.select('average_student_attendance').dtypes
```

```
[('average_student_attendance', 'float')]
```

```
indexed.select('average_teacher_attendance').dtypes
```

```
[('average_teacher_attendance', 'string')]
```

```
indexed.select('average_teacher_attendance').show(4)
```

```
+-------------------------+
|average_teacher_attendance|
+-------------------------+
|                   96.70%|
|                   96.50%|
|                   96.90%|
|                   94.20%|
+-------------------------+
only showing top 4 rows
```

Getting rid of the % sign and transforming that to a numebr

```
indexed=indexed.withColumn('average_teacher_attendance',
                    split(indexed['average_teacher_attendance'],'%').ge
```

```
indexed.select('average_teacher_attendance').show(10)
```

```
+-------------------------+
|average_teacher_attendance|
+-------------------------+
|                    96.70|
|                    96.50|
|                    96.90|
|                    94.20|
|                    96.80|
|                    93.30|
|                    96.00|
|                    94.60|
|                    95.70|
|                    95.50|
+-------------------------+
only showing top 10 rows
```

```
indexed=indexed.withColumn('average_teacher_attendance',indexed['avera
```

```
indexed.select('average_teacher_attendance').dtypes
```

```
[('average_teacher_attendance', 'float')]
```

```
indexed.select('individualized_education_program_compliance_rate_').sh
```

```
+--------------------------------------------------+
|individualized_education_program_compliance_rate_|
+--------------------------------------------------+
|                                            98.90%|
|                                            97.30%|
|                                           100.00%|
|                                           100.00%|
|                                           100.00%|
+--------------------------------------------------+
only showing top 5 rows
```

```
indexed=indexed.withColumn('individualized_education_program_complianc
                      split(indexed['individualized_education_program_com
```

```
indexed.select('individualized_education_program_compliance_rate_').sh
```

```
+--------------------------------------------------+
|individualized_education_program_compliance_rate_|
+--------------------------------------------------+
|                                             98.90|
|                                             97.30|
|                                            100.00|
|                                            100.00|
|                                            100.00|
+--------------------------------------------------+
only showing top 5 rows
```

Categorical variables and useless for ML models, so they should be numeric

```
indexed.select('isat_value_add_color_math').show()
indexed.select('isat_value_add_color_math').show()
```

```
+-------------------------+
|isat_value_add_color_math|
+-------------------------+
|                   Yellow|
|                   Yellow|
|                   Yellow|
|                      Red|
|                    Green|
|                    Green|
|                    Green|
|                   Yellow|
|                      Red|
|                    Green|
|                   Yellow|
|                      Red|
|                      Red|
|                   Yellow|
|                    Green|
|                      Red|
|                    Green|
|                   Yellow|
|                      Red|
|                   Yellow|
+-------------------------+
only showing top 20 rows
```

Transforming the SAT values to numeric encoding

```
indexer=StringIndexer(inputCol='isat_value_add_color_math',outputCol='
indexed=indexer.fit(indexed).transform(indexed)

```

```
indexed.show(5)
```

```
+--------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+--------------------+---------+--------+------+--------------+--------------------
|                   4|   609852|   60625|   Yes|           0.0|
|                  10|   609937|   60656|    No|           0.0|
|                  24|   609828|   60622|   Yes|           1.0|
|                   6|   610355|   60613|   Yes|           0.0|
|                  61|   609929|   60609|    No|           1.0|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 5 rows
```

```
indexed=indexed.drop('isat_value_add_color_math')
```

```
indexed=indexed.withColumnRenamed('isat_value_add_color_math_indexer',
indexed.select('individualized_education_program_compliance_rate_').sh
```

```
+-------------------------------------------------+
|individualized_education_program_compliance_rate_|
+-------------------------------------------------+
|                                            98.90|
|                                            97.30|
|                                           100.00|
|                                           100.00|
|                                           100.00|
+-------------------------------------------------+
only showing top 5 rows
```

This variable is also string and it should be transformd to the float nuumber

```
indexed.select('individualized_education_program_compliance_rate_').dt
```

```
[('individualized_education_program_compliance_rate_', 'string')]
```

```
indexed=indexed.withColumn('individualized_education_program_complianc
                       indexed['individualized_education_program_complianc
```

```
indexed.select('individualized_education_program_compliance_rate_').dt
```

```
[('individualized_education_program_compliance_rate_', 'float')]
```

Finding wich variable is remaining as n string

```
for i,j in indexed.dtypes:
  if j=='string':
    print(i,'=================> string type yet!')
    print()
```

```
target =================> string type yet!

isat_value_add_color_read =================> string type yet!
```

```
indexed.select('isat_value_add_color_read').show(5)
```

```
+-------------------------+
|isat_value_add_color_read|
+-------------------------+
|                    Green|
|                  Yellow|
|                  Yellow|
|                  Yellow|
|                     Red|
+------------------------+
only showing top 5 rows
```

Transfoming, encoding the variable isat_value_add_color_read

```
indexer=StringIndexer(inputCol='isat_value_add_color_read',outputCol='
indexed=indexer.fit(indexed).transform(indexed)
```

```
indexed.show(5)
```

```
+--------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+--------------------+---------+--------+------+--------------+--------------------
|                   4|   609852|   60625|   Yes|           0.0|
|                  10|   609937|   60656|    No|           0.0|
|                  24|   609828|   60622|   Yes|           1.0|
|                   6|   610355|   60613|   Yes|           0.0|
|                  61|   609929|   60609|    No|           1.0|
+--------------------+---------+--------+------+--------------+--------------------
only showing top 5 rows
```

```
indexed=indexed.drop('isat_value_add_color_read')
```

```
indexed=indexed.withColumnRenamed('isat_value_add_color_read_indexer',
```

All the variables tranfformed to numerical with the encoding them

```
indexed.dtypes
```

```
[('community_area_number', 'int'),
 ('school_id', 'int'),
 ('zip_code', 'int'),
 ('target', 'string'),
 ('track_schedule', 'double'),
 ('healthy_school_certified', 'double'),
 ('safety_score', 'int'),
 ('environment_score', 'int'),
 ('instruction_score', 'int'),
 ('average_student_attendance', 'float'),
 ('rate_of_misconducts_per_100_students_', 'double'),
 ('average_teacher_attendance', 'float'),
 ('individualized_education_program_compliance_rate_', 'float'),
 ('isat_exceeding_math_%', 'double'),
 ('isat_exceeding_reading_%_', 'double'),
 ('isat_value_add_math', 'double'),
 ('isat_value_add_read', 'double'),
 ('college_enrollment', 'int'),
 ('general_services_route_', 'int'),
 ('x_coordinate', 'double'),
 ('y_coordinate', 'double'),
 ('latitude', 'double'),
 ('longitude', 'double'),
 ('ward', 'int'),
 ('police_district', 'int'),
 ('percent_of_housing_crowded', 'double'),
 ('percent_households_below_poverty', 'double'),
 ('percent_aged_16+_unemployed', 'double'),
 ('percent_aged_25+_without_high_school_diploma', 'double'),
 ('percent_aged_under_18_or_over_64', 'double'),
 ('per_capita_income_', 'int'),
 ('hardship_index', 'int'),
 ('elementary_middle_or_high_school', 'double'),
 ('isat_value_add_color_math', 'double'),
 ('isat_value_add_color_read', 'double')]
```

```
indexed.select('target').toPandas().value_counts()
```

```
target
No       370
Yes       63
NDA        3
dtype: int64
```

There are three record in our target columns that should be fixed. I will use the frequency method to replace the NDA's with the most frequent recods shown in the data.

```
indexed.filter(indexed['target']=='NDA').show()
```

```
+---------------------+---------+--------+------+--------------+---------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+---------------------+---------+--------+------+--------------+---------------------
|                   28|   610075|   60608|   NDA|           1.0|
|                   24|   610085|   60642|   NDA|           0.0|
|                   48|   610280|   60617|   NDA|           0.0|
+---------------------+---------+--------+------+--------------+---------------------
```

Viewing that how the target variabl's records are showing up in our data set.

```
indexed.filter((indexed['community_area_number']==28) |(indexed['commu
```

```
+---------------------+---------+--------+------+--------------+---------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+---------------------+---------+--------+------+--------------+---------------------
|                   24|   609828|   60622|   Yes|           1.0|
|                   28|   610180|   60608|    No|           1.0|
|                   24|   610076|   60647|    No|           1.0|
|                   28|   609812|   60612|    No|           0.0|
|                   24|   610073|   60612|   Yes|           0.0|
|                   28|   610075|   60608|   NDA|           1.0|
|                   24|   610107|   60622|    No|           0.0|
|                   28|   610023|   60612|    No|           1.0|
|                   28|   610121|   60612|    No|           1.0|
|                   28|   610009|   60607|   Yes|           0.0|
|                   24|   610031|   60622|    No|           1.0|
|                   28|   609989|   60612|    No|           1.0|
|                   48|   610218|   60617|    No|           0.0|
|                   24|   610320|   60622|    No|           0.0|
|                   24|   610085|   60642|   NDA|           0.0|
|                   28|   610177|   60607|   Yes|           0.0|
|                   24|   610342|   60622|    No|           0.0|
|                   24|   610210|   60622|    No|           0.0|
|                   24|   610313|   60622|    No|           0.0|
|                   24|   610529|   60622|   Yes|           0.0|
+---------------------+---------+--------+------+--------------+---------------------
only showing top 20 rows
```

```
# Replacing the NDA's with the None values then changing them. Spark a
# This is an easy trick to doing so.

indexed=indexed.withColumn('target',when(indexed['target']=='NDA',None
```

```
indexed.select('target').toPandas().isnull().sum()
```

```
target    3
dtype: int64
```

Because the top most frequesncy value is No in target variable I will replace the 3 null values with 0

```
indexed=indexed.fillna('No')
```

```
indexed.select('target').toPandas().isnull().sum()
```

```
target    0
dtype: int64
```

Encoding the target variable

```
indexer=StringIndexer(inputCol='target',outputCol='target_indexer',)
indexed=indexer.fit(indexed).transform(indexed)
```

```
indexed.toPandas().isnull().sum()
```

```
community_area_number                                0
school_id                                            0
zip_code                                             0
target                                               0
track_schedule                                       0
healthy_school_certified                             0
safety_score                                         0
environment_score                                    0
instruction_score                                    0
average_student_attendance                           0
rate_of_misconducts_per_100_students_                0
average_teacher_attendance                           0
individualized_education_program_compliance_rate_    0
isat_exceeding_math_%                                0
isat_exceeding_reading_%_                            0
isat_value_add_math                                  0
isat_value_add_read                                  0
college_enrollment                                   0
general_services_route_                              0
x_coordinate                                         0
y_coordinate                                         0
latitude                                             0
longitude                                            0
ward                                                 0
police_district                                      0
percent_of_housing_crowded                           0
percent_households_below_poverty                     0
percent_aged_16+_unemployed                          0
percent_aged_25+_without_high_school_diploma         0
percent_aged_under_18_or_over_64                     0
per_capita_income_                                   0
hardship_index                                       0
elementary_middle_or_high_school                     0
isat_value_add_color_math                            0
isat_value_add_color_read                            0
target_indexer                                       0
dtype: int64
```

```
indexed.show(10)
```

```
+---------------------+---------+--------+------+--------------+--------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+---------------------+---------+--------+------+--------------+--------------------
|                    4|   609852|   60625|   Yes|           0.0|
|                   10|   609937|   60656|    No|           0.0|
|                   24|   609828|   60622|   Yes|           1.0|
|                   6|   610355|   60613|   Yes|           0.0|
|                  61|   609929|   60609|    No|           1.0|
|                  28|   610180|   60608|    No|           1.0|
|                  61|   610239|   60609|    No|           1.0|
|                  71|   609805|   60620|    No|           1.0|
|                  61|   610167|   60609|    No|           0.0|
|                  69|   609813|   60637|    No|           1.0|
+--------------------+---------+--------+------+-------------+--------------------
only showing top 10 rows
```

More Queries abour finding the interesting facts from our data

Q6. What was the maximum amount of the Per Capita Income amoung the all schools in Chicago?

```
indexed.createOrReplaceTempView('inx')
spark.sql("select max(per_capita_income_) from inx").show()

```

```
+-----------------------+
|max(per_capita_income_)|
+-----------------------+
|                  88669|
+-----------------------+
```

Q7. Interesting question

Did students with the highest per capita income rate do well performance in their education ? 75 % did NOT and 25 % did.

```
spark.sql('select target from inx where per_capita_income_==88669').sh
```

```
+------+
|target|
+------+
|    No|
|    No|
|    No|
|   Yes|
+------+
```

Q8. What was the average percetage of housing population? 5 % per house.

```
spark.sql('select avg(percent_of_housing_crowded) from inx').show()
```

```
+-------------------------------+
|avg(percent_of_housing_crowded)|
+-------------------------------+
|              5.415825688073393|
+-------------------------------+
```

Q9. How many students did enroll amoung the shools in the state in total? 250629 students.

```
spark.sql('select sum(college_enrollment) from inx').show()
```

```
+-----------------------+
|sum(college_enrollment)|
+-----------------------+
|                 250629|
+-----------------------+
```

Q10. How many distinct police distriction were found in data? 436

```
spark.sql('select distinct count(police_district)  from inx').show()
```

```
+----------------------+
|count(police_district)|
+----------------------+
|                   436|
+----------------------+
```

```
indexed.show(5)
```

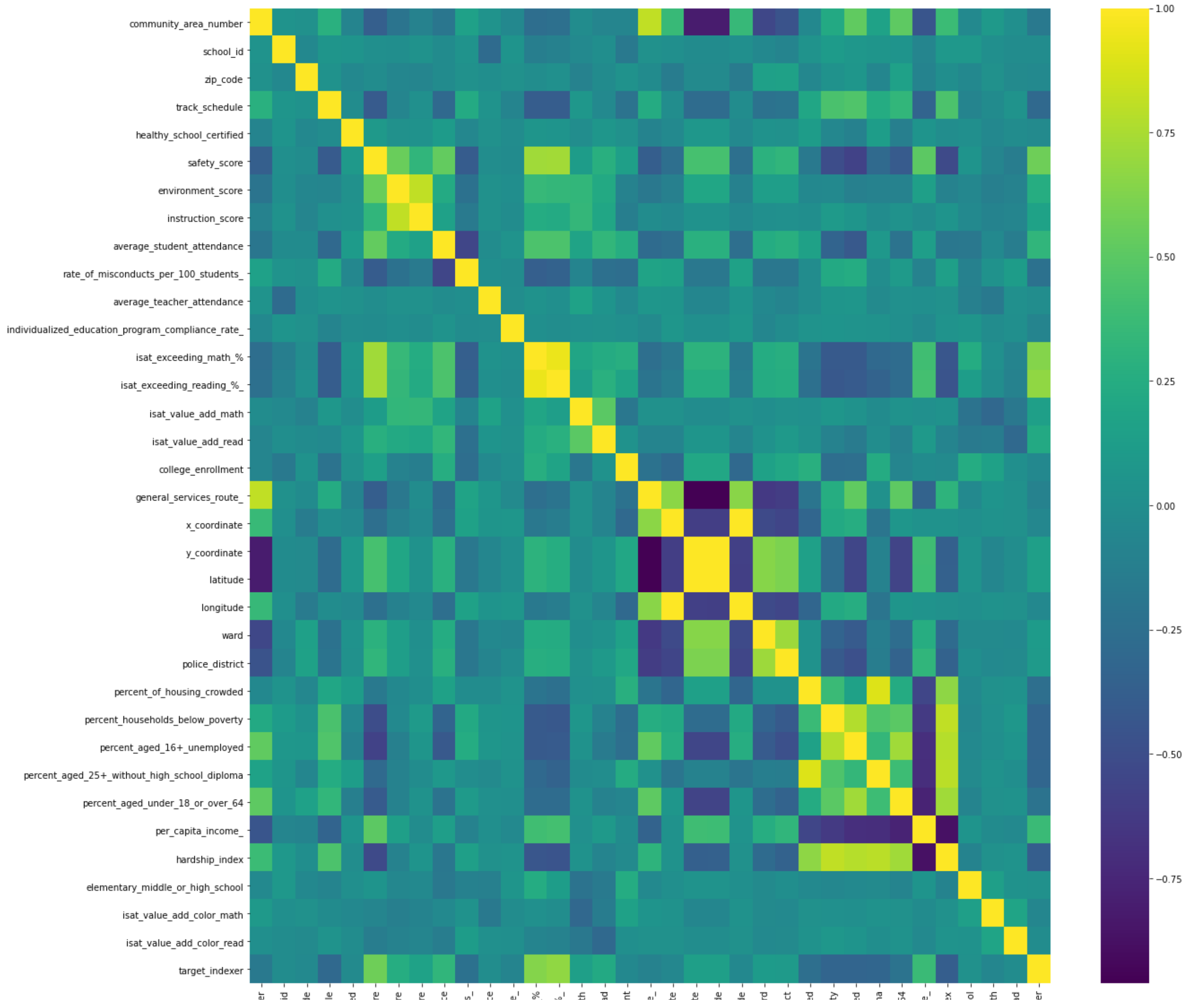
```
+---------------------+---------+--------+------+--------------+---------------------
|community_area_number|school_id|zip_code|target|track_schedule|healthy_school_certi
+---------------------+---------+--------+------+--------------+---------------------
|                    4|   609852|   60625|   Yes|           0.0|
|                   10|   609937|   60656|    No|           0.0|
|                   24|   609828|   60622|   Yes|           1.0|
|                    6|   610355|   60613|   Yes|           0.0|
|                   61|   609929|   60609|    No|           1.0|
+---------------------+---------+--------+------+--------------+---------------------
only showing top 5 rows
```

Findig the coefficient correlations with visualization to Feature Extraction in the data after joing both data sets.

```
plt.figure(figsize=(20,20))
sns.heatmap(indexed.toPandas().corr(),cmap='viridis',)

```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8bf1d297b8>
```



As we see visually finding the most efficient features are hard to determin

We need more suffisticated procedures to pick our feartures.

```
from pyspark.ml.feature import RFormula
```

```
formula = RFormula(
    formula="target ~ .",
    featuresCol="features",
    labelCol="label")
```

```
output = formula.fit(indexed).transform(indexed)
output.select("features", "label").show()
```

```
+--------------------+-----+
|            features|label|
+--------------------+-----+
|[4.0,609852.0,606...|  1.0|
|[10.0,609937.0,60...|  0.0|
|[24.0,609828.0,60...|  1.0|
|[6.0,610355.0,606...|  1.0|
|[61.0,609929.0,60...|  0.0|
|[28.0,610180.0,60...|  0.0|
|[61.0,610239.0,60...|  0.0|
|[71.0,609805.0,60...|  0.0|
|[61.0,610167.0,60...|  0.0|
|[69.0,609813.0,60...|  0.0|
|[30.0,609973.0,60...|  0.0|
|[39.0,609746.0,60...|  0.0|
|[58.0,610353.0,60...|  0.0|
|[55.0,609856.0,60...|  0.0|
|[24.0,610076.0,60...|  0.0|
|[63.0,610532.0,60...|  0.0|
|[27.0,610251.0,60...|  0.0|
|[5.0,610010.0,606...|  0.0|
|[31.0,610125.0,60...|  0.0|
|[43.0,610103.0,60...|  0.0|
+--------------------+-----+
only showing top 20 rows
```

```
output.summary().show()
```

```
+-------+---------------------+------------------+----------------+------+----------
|summary|community_area_number|         school_id|        zip_code|target|     track
+-------+---------------------+------------------+----------------+------+----------
|  count|                  436|               436|             436|   436|
|   mean|    38.41743119266055| 610068.1880733945|60630.05275229358|  null|0.4564220
| stddev|    21.84027848770573|180.80035714785598|21.77258566776887|  null|0.4986695
|    min|                    1|            609725|           60605|    No|
|    25%|                   23|            609918|           60618|  null|
|    50%|                   34|            610067|           60625|  null|
|    75%|                   60|            610200|           60639|  null|
|    max|                   77|            610544|           60827|   Yes|
+-------+---------------------+------------------+----------------+------+----------
```

Feature Extraction and Selection on the best scores for both data sets, procedure.

Setting the explantory and response variables to a panda data fram. Later on I'll use the sklearn library to doing so.

```
y=indexed.select('target').toPandas()

```

```
y
```

	target
0	Yes
1	No
2	Yes
3	Yes
4	No
...	...
431	Yes
432	Yes
433	No
434	Yes
435	No

436 rows × 1 columns

```
X=indexed.select('*').drop('target').toPandas()
X

```

	community_area_number	school_id	zip_code	track_schedule	healthy_school_cert
0	4	609852	60625	0.0	
1	10	609937	60656	0.0	
2	24	609828	60622	1.0	
3	6	610355	60613	0.0	
4	61	609929	60609	1.0	
...	...	...	...	...	
431	48	610316	60617	1.0	
432	2	610191	60659	0.0	
433	68	610173	60621	1.0	
434	24	609863	60622	0.0	
435	15	610179	60634	0.0	

436 rows × 35 columns

```
from sklearn.ensemble import ExtraTreesClassifier
```

```
model=ExtraTreesClassifier()
```

```
model.fit(X,y)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:1: DataConversionWarning
  """Entry point for launching an IPython kernel.
ExtraTreesClassifier(bootstrap=False, ccp_alpha=0.0, class_weight=None,
                     criterion='gini', max_depth=None, max_features='auto',
                     max_leaf_nodes=None, max_samples=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, n_estimators=100,
                     n_jobs=None, oob_score=False, random_state=None, verbose=0,
                     warm_start=False)
```

```
model.feature_importances_
```

```
array([0.00454804, 0.00633722, 0.00607385, 0.01697049, 0.00058912,
       0.04865729, 0.00885919, 0.00691484, 0.01871791, 0.00871088,
       0.00539243, 0.00710986, 0.05613585, 0.07849525, 0.00806118,
       0.00823698, 0.0093317 , 0.00526295, 0.00541539, 0.00764897,
       0.00721356, 0.00576654, 0.00565357, 0.00502916, 0.01118319,
       0.01577276, 0.01416577, 0.01511919, 0.00820842, 0.01409268,
       0.02018348, 0.0026574 , 0.00653247, 0.00499776, 0.54595469])
```

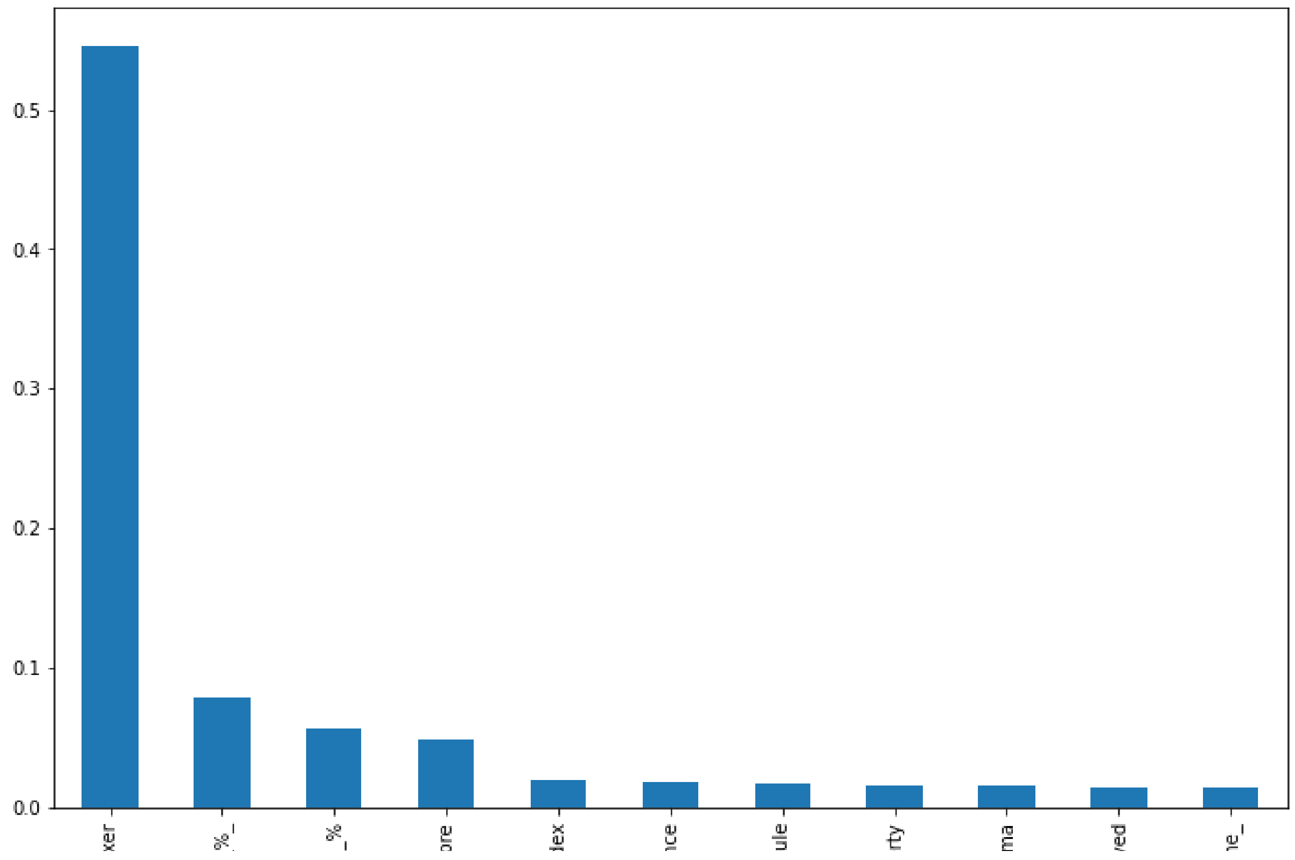
```
feat_importance=pd.Series(model.feature_importances_,index=X.columns)
```

Visualizing the most impotarn features base of the feature importance algorithms with the 10 variables

```
plt.figure(figsize=(12,8))
feat_importance.nlargest(11).plot(kind='bar')

```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8bf1b91b70>
```



Here, found the top 10 high correlated columns. Also, lets filter out the columns to report the choosen colmns

```
indexed=indexed.drop('target')
indexed=indexed.withColumnRenamed('target_indexr','target')
indexed.show(5)
```

```
+---------------------+---------+--------+--------------+------------------------+---
|community_area_number|school_id|zip_code|track_schedule|healthy_school_certified|saf
+---------------------+---------+--------+--------------+------------------------+---
|                    4|   609852|   60625|           0.0|                     0.0|
|                   10|   609937|   60656|           0.0|                     0.0|
|                   24|   609828|   60622|           1.0|                     0.0|
|                    6|   610355|   60613|           0.0|                     0.0|
|                   61|   609929|   60609|           1.0|                     0.0|
+---------------------+---------+--------+--------------+------------------------+---
only showing top 5 rows
```

```
final_data_clean_data=indexed.select(['per_capita_income_',
                    'percent_households_below_poverty',
                    'percent_aged_16+_unemployed',
                    'college_enrollment',
                    'percent_aged_25+_without_high_school_diploma',
                    'hardship_index',
                    'average_student_attendance',
                    'safety_score',
                    'isat_exceeding_math_%',
                    'isat_exceeding_reading_%_',
                    'target_indexer'])
```

Here is my final data, clean data, all colomns with the high corelation and selection feature process have done with many Machine Learning algorithms.

```
final_data_clean_data.show(20)
```

```
+-----------------+--------------------------------+--------------------------+----
|per_capita_income_|percent_households_below_poverty|percent_aged_16+_unemployed|col
+-----------------+--------------------------------+--------------------------+----
|            37524|                            10.9|                       8.2|
|            32875|                             5.4|                       9.0|
|            43198|                            14.7|                       6.6|
|            60058|                            11.4|                       4.7|
|            12765|                            29.0|                      23.0|
|            44689|                            20.6|                      10.7|
|            12765|                            29.0|                      23.0|
|            15528|                            27.6|                      28.3|
|            12765|                            29.0|                      23.0|
|            17285|                            29.6|                      23.0|
|            10402|                            30.7|                      15.8|
|            35911|                            21.7|                      15.7|
|            13089|                            23.6|                      13.9|
|            22677|                            17.1|                       9.6|
|            43198|                            14.7|                       6.6|
|            12171|                            23.4|                      18.2|
|            12961|                            42.4|                      19.6|
|            57123|                             7.5|                       5.2|
|            16444|                            25.8|                      15.8|
|            19398|                            31.1|                      20.0|
+-----------------+--------------------------------+--------------------------+----
only showing top 20 rows
```

Our final clean data a Spark engine on top of the Sqlite3 data base.

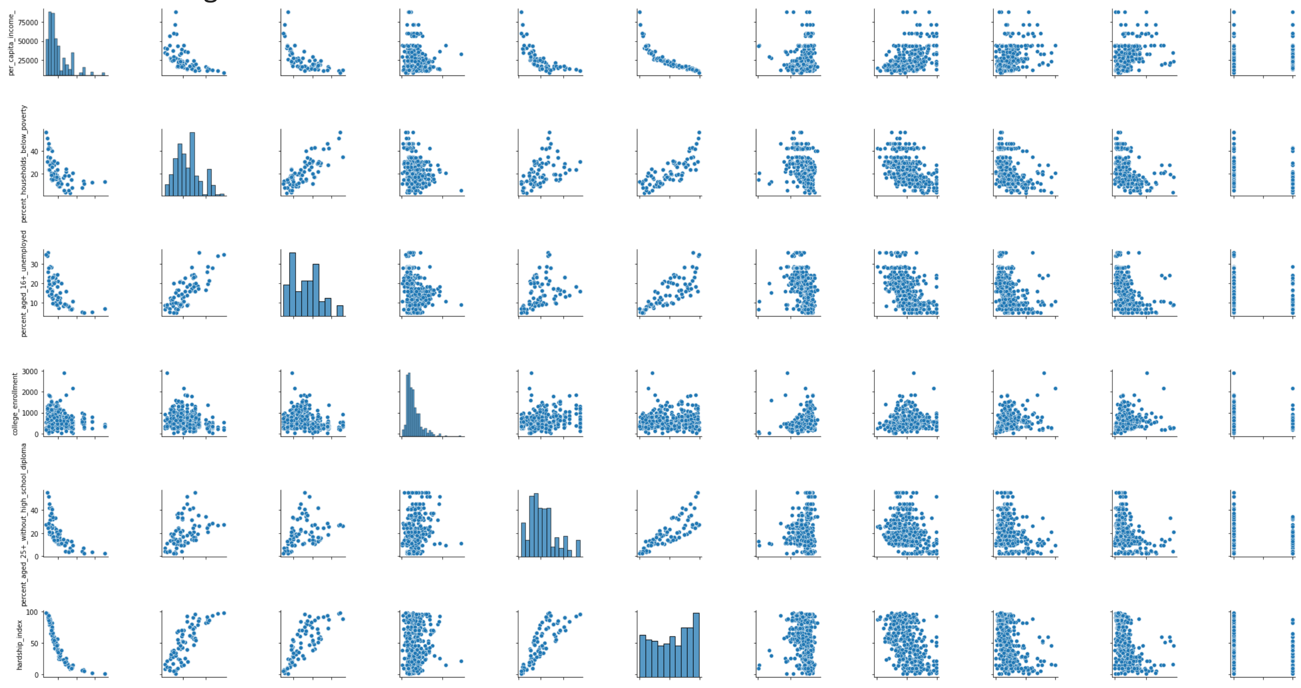
```
type(final_data_clean_data)
```

```
pyspark.sql.dataframe.DataFrame
```

Let's get have an idea what are the relation ships amoung all the

```
sns.pairplot(final_data_clean_data.toPandas())
```

```
<seaborn.axisgrid.PairGrid at 0x7f8bf1d1ec18>
```



It seems that ISAT math and reading have a highly linearly relation, Also, as it seems there are some skewness and unnormality in the data, so it is better to transfom and normalize the data to get the better and precise reasults for our final goal.

Modelig the Data

Research question was, did the student do well according the prepared data information and the responses were YES, or NO. Therefore, classification is the one that can be implemented in our case. Further, I want to find a model that responds to the question and it should be a Yes or No response.

Logestic Regresion

In this project, I am using the Big Data Application Analysis With Spark and SQL, using the Mlib library

The Mlib machine learning libraries just accept two final columns to be processing one is feature and other is a label column, so for that we should transform all the data somehow that the model understands it.Therfore, we should use the Vectorixzation technique to transform our data to two vectors, which one is vector feature and the other is the label as told,so and in my case is target_indexr, also for reminding purpose the trageted variable is target_indexer or as it's original name to "Adequate Yearly Progress Made?"

```
from pyspark.ml.feature import VectorAssembler
```

```
final_data_clean_data.columns
```

```
['per_capita_income_',
 'percent_households_below_poverty',
 'percent_aged_16+_unemployed',
 'college_enrollment',
 'percent_aged_25+_without_high_school_diploma',
 'hardship_index',
 'average_student_attendance',
 'safety_score',
 'isat_exceeding_math_%',
 'isat_exceeding_reading_%_',
 'target_indexer']
```

Vectorizing the variables

```
vec_assembler=VectorAssembler(inputCols=['per_capita_income_',
 'percent_households_below_poverty',
 'percent_aged_16+_unemployed',
 'college_enrollment',
 'percent_aged_25+_without_high_school_diploma',
 'hardship_index',
 'average_student_attendance',
 'safety_score',
 'isat_exceeding_math_%',
 'isat_exceeding_reading_%_'],outputCol='features')
```

```
out_vec=vec_assembler.transform(final_data_clean_data.drop('traget_inc
```

Scaling the Data

It is an important task scalinig and normalizing the data.idea We should scale the data before more progress at this point.

```
from pyspark.ml.feature import StandardScaler
```

```
scaler=StandardScaler(inputCol='features',outputCol='scaledFeatures',w
```

```
scalerModel=scaler.fit(out_vec)
```

```
final_vec=scalerModel.transform(out_vec)
```

```
final_vec.show()
```

```
+-----------------+--------------------------------+--------------------------+----
|per_capita_income_|percent_households_below_poverty|percent_aged_16+_unemployed|col
+-----------------+--------------------------------+--------------------------+----
|            37524|                            10.9|                       8.2|
|            32875|                             5.4|                       9.0|
|            43198|                            14.7|                       6.6|
|            60058|                            11.4|                       4.7|
|            12765|                            29.0|                      23.0|
|            44689|                            20.6|                      10.7|
|            12765|                            29.0|                      23.0|
|            15528|                            27.6|                      28.3|
|            12765|                            29.0|                      23.0|
|            17285|                            29.6|                      23.0|
|            10402|                            30.7|                      15.8|
|            35911|                            21.7|                      15.7|
|            13089|                            23.6|                      13.9|
|            22677|                            17.1|                       9.6|
|            43198|                            14.7|                       6.6|
|            12171|                            23.4|                      18.2|
|            12961|                            42.4|                      19.6|
|            57123|                             7.5|                       5.2|
|            16444|                            25.8|                      15.8|
|            19398|                            31.1|                      20.0|
+-----------------+--------------------------------+--------------------------+----
only showing top 20 rows
```

```
final_vec_selection=final_vec.select('features','target_indexer')
```

```
final_vec_selection.show()
```

```
+--------------------+--------------+
|            features|target_indexer|
+--------------------+--------------+
|[37524.0,10.9,8.2...|           1.0|
|[32875.0,5.4,9.0,...|           0.0|
|[43198.0,14.7,6.6...|           1.0|
|[60058.0,11.4,4.7...|           1.0|
|[12765.0,29.0,23....|           0.0|
|[44689.0,20.6,10....|           0.0|
|[12765.0,29.0,23....|           0.0|
|[15528.0,27.6,28....|           0.0|
|[12765.0,29.0,23....|           0.0|
|[17285.0,29.6,23....|           0.0|
|[10402.0,30.7,15....|           0.0|
|[35911.0,21.7,15....|           0.0|
|[13089.0,23.6,13....|           0.0|
|[22677.0,17.1,9.6...|           0.0|
|[43198.0,14.7,6.6...|           0.0|
|[12171.0,23.4,18....|           0.0|
|[12961.0,42.4,19....|           0.0|
|[57123.0,7.5,5.2,...|           0.0|
|[16444.0,25.8,15....|           0.0|
|[19398.0,31.1,20....|           0.0|
+--------------------+--------------+
only showing top 20 rows
```

Training data and testing data is another important procedures that it should be done if we want to have a precise and accurate results.

Train Test Split

```
train,test=final_vec_selection.randomSplit([0.7,0.3])
```

```
from pyspark.ml.classification import LogisticRegression
```

Fitting the model

```
lR=LogisticRegression(labelCol='target_indexer')
```

```
lRModel=lR.fit(train)
```

Model Evaluation

```
lRModel.summary.accuracy
```

```
0.9405940594059405
```

Got a 95% accuracy which is awesome, the one reason for this is that the cleaning the data has completed with very causion!

```
from pyspark.ml.evaluation import BinaryClassificationEvaluator
```

More evaluation on test data

```
pred_and_labels=lRModel.evaluate(test)
```

```
pred_and_labels.predictions.show()
```

```
+--------------------+-------------+--------------------+--------------------+-----
|            features|target_indexer|       rawPrediction|         probability|predic
+--------------------+-------------+--------------------+--------------------+-----
|[60058.0,11.4,4.7...|          1.0|[-0.6515596351473...|[0.34263816322578...|
|[12765.0,29.0,23....|          0.0|[7.29700090165460...|[0.99932289096374...|
|[12765.0,29.0,23....|          0.0|[6.96074077557897...|[0.99905250483119...|
|[15528.0,27.6,28....|          0.0|[4.33107666639969...|[0.98701738715167...|
|[44689.0,20.6,10....|          0.0|[4.24415312603299...|[0.98585507036340...|
|[10402.0,30.7,15....|          0.0|[6.24319313358492...|[0.99806013110185...|
|[17285.0,29.6,23....|          0.0|[4.15767516075531...|[0.98459707637453...|
|[35911.0,21.7,15....|          0.0|[3.87049129402733...|[0.97957764347850...|
|[13089.0,23.6,13....|          0.0|[5.75033280462228...|[0.99682836951703...|
|[12171.0,23.4,18....|          0.0|[7.57340917750129...|[0.99948632664099...|
|[15957.0,28.6,22....|          0.0|[4.76899566805198...|[0.99158255332108...|
|[57123.0,7.5,5.2,...|          0.0|[2.47310912392501...|[0.92223503662239...|
|[17104.0,19.2,12....|          0.0|[5.03173997964759...|[0.99351488813276...|
|[44689.0,20.6,10....|          0.0|[3.46789338736027...|[0.96976030265187...|
|[15957.0,28.6,22....|          0.0|[4.65861776877529...|[0.99060946184607...|
|[23482.0,10.4,11....|          0.0|[4.33040866835338...|[0.98700882459623...|
|[43198.0,14.7,6.6...|          1.0|[-1.2483477625146...|[0.22298628007815...|
|[12034.0,43.1,21....|          0.0|[4.95375309879908...|[0.99299257648424...|
|[23791.0,29.6,18....|          0.0|[2.29927345140624...|[0.90881684866717...|
|[16563.0,25.9,19....|          0.0|[4.53935387111489...|[0.98943255832379...|
+--------------------+-------------+--------------------+--------------------+-----
only showing top 20 rows
```

Comparing the predictions and the original values shows that how

AUC ------> area under the curve evaluation process

```
AUC_eval=BinaryClassificationEvaluator(rawPredictionCol='prediction',l
```

```
AUC=AUC_eval.evaluate(pred_and_labels.predictions)
```

87% accuracy with the AUC test which is pretty good.

```
AUC
```

```
0.7898073022312374
```

```
final_data_clean_data.show(6)
```

```
+-----------------+-------------------------------+-------------------------+----
|per_capita_income_|percent_households_below_poverty|percent_aged_16+_unemployed|col
+-----------------+-------------------------------+-------------------------+----
|            37524|                           10.9|                      8.2|
|            32875|                            5.4|                      9.0|
|            43198|                           14.7|                      6.6|
|            60058|                           11.4|                      4.7|
|            12765|                           29.0|                     23.0|
|            44689|                           20.6|                     10.7|
+-----------------+-------------------------------+-------------------------+----
only showing top 6 rows
```

Reports to Census

With Quering and analyzing the data based on the machin learning techniques and feature selecting techniques, it has found that there are some columns to be consisder from amoung of the data sets.

1.per_capita_income_,

2.percent_households_below_poverty,

3.percent_aged_16+_unemployed,

4.college_enrollment,

5.percent_aged_25+_without_high_school_diploma,

6.hardship_index,

7.average_student_attendance,

8.safety_score,

9.isat_exceeding_math_%,

10.isat_exceeding_reading_%_

These are the most highly coefficiently linearly related and as a results these coloumns could be considered for further decisions.Also these predictors are trying to respond to the target variable as it is in the original data as "Adequate Yearly Progress Made?"