Prediction Model for the number of deaths due to COVID-19 in Africa

SANDILE MDUDUZI MCHUNU

MAY20/MIT/037U

Import libaries

In [3]:
```
import pandas as pd
import numpy as np
```

In [4]:
```
path='WHO COVID-19 global table data June 27th 2021 at 4.32.12 PM.csv'
global_df = pd.read_csv(path)
```

In [5]:
```
global_df.dtypes
```

Out[5]:
```
Name                                                  object
WHO Region                                            object
Cases - cumulative total                               int64
Cases - cumulative total per 100000 population       float64
Cases - newly reported in last 7 days                  int64
Cases - newly reported in last 7 days per 100000 population    float64
Cases - newly reported in last 24 hours                int64
Deaths - cumulative total                              int64
Deaths - cumulative total per 100000 population      float64
Deaths - newly reported in last 7 days                 int64
Deaths - newly reported in last 7 days per 100000 population    float64
Deaths - newly reported in last 24 hours               int64
Transmission Classification                           object
dtype: object
```

In [6]:
```
global_df['WHO Region'].value_counts().to_frame()
```

Out[6]:

	WHO Region
Europe	62
Americas	56
Africa	50
Western Pacific	35
Eastern Mediterranean	22
South-East Asia	11
Other	1

In [7]:
```
africa_df = global_df.loc[(global_df['WHO Region'] == 'Africa')]
```

In [8]:
```
africa_df['WHO Region'].value_counts().to_frame()
```

Out[8]:

	WHO Region

	WHO Region
Africa	50

In [16]:
```
africa_df.rename(columns = {'Name':'Country','Cases - cumulative total':'Total_Cases
                            'Cases - cumulative total per 100000 population':'Total_Cas
                            'Cases - newly reported in last 7 days':'Weekly_Cases',
                            'Cases - newly reported in last 7 days per 100000 populatio
                            'Cases - newly reported in last 24 hours':'Daily_Cases',
                            'Deaths - cumulative total':'Total_Deaths',
                            'Deaths - cumulative total per 100000 population':'Total_De
                            'Deaths - newly reported in last 7 days':'Weekly_Deaths',
                            'Deaths - newly reported in last 7 days per 100000 populati
                            'Deaths - newly reported in last 24 hours':'Daily_Deaths',
                             'Transmission Classification' : 'Transmission_Classificati
                              }, inplace = True)
```

```
/srv/conda/envs/notebook/lib/python3.6/site-packages/pandas/core/frame.py:4308: Sett
ingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  errors=errors,
```

Check the table headers

In [22]:
```
africa_df.head()
```

Out[22]:

	Country	WHO Region	Total_Cases	Total_Cases_per_100000_population	Weekly_Cases	Weekly_Cases_p
19	South Africa	Africa	1877143	3165.04	91064	
67	Ethiopia	Africa	275601	239.73	826	
81	Kenya	Africa	181239	337.06	3957	
83	Nigeria	Africa	167401	81.21	259	
87	Zambia	Africa	140620	764.91	18376	

In [23]:
```
africa_df.tail()
```

Out[23]:

	Country	WHO Region	Total_Cases	Total_Cases_per_100000_population	Weekly_Cases	Weekly_Cases
181	Liberia	Africa	3265	64.56	536	
189	Sao Tome and Principe	Africa	2364	1078.67	5	
194	Mauritius	Africa	1852	145.62	79	

	Country	WHO Region	Total_Cases	Total_Cases_per_100000_population	Weekly_Cases	Weekly_Cases
203	United Republic of Tanzania	Africa	509	0.85	0	
233	Saint Helena	Africa	0	0.00	0	

In [24]:
```
africa_df.shape
```

Out[24]: (50, 13)

Data Analysis

In [38]:
```
africa_df.describe()
```

Out[38]:

	Total_Cases	Total_Cases_per_100000_population	Weekly_Cases	Weekly_Cases_per_100000_po
count	5.000000e+01	50.000000	50.000000	5(
mean	7.822606e+04	1060.529800	3215.760000	4!
std	2.653622e+05	2523.380276	13082.564441	15!
min	0.000000e+00	0.000000	0.000000	(
25%	6.333500e+03	101.417500	38.750000	(
50%	1.995850e+04	230.955000	243.000000	2
75%	4.712000e+04	505.582500	947.500000	1(
max	1.877143e+06	15364.980000	91064.000000	100

1. Pearson Correlations

Check Correlation of the entire data

In [40]:
```
africa_df.corr()
```

Out[40]:

	Total_Cases	Total_Cases_per_100000_population	Weekly_Cases
Total_Cases	1.000000	0.105260	0.969138
Total_Cases_per_100000_population	0.105260	1.000000	0.129520
Weekly_Cases	0.969138	0.129520	1.000000
Weekly_Cases_per_100000_population	0.099370	0.858236	0.157503
Daily_Cases	0.947494	0.146628	0.990264
Total_Deaths	0.993836	0.106699	0.972590
Total_Deaths_per_100000_population	0.575956	0.688792	0.606945

	Total_Cases	Total_Cases_per_100000_population	Weekly_Cases
Weekly_Deaths	0.933022	0.120342	0.981895
Weekly_Deaths_per_100000_population	0.148798	0.503043	0.246266
Daily_Deaths	0.830222	0.140741	0.909088

Check Correlation of the entire two variables

In [12]:
```
africa_df[['Daily_Cases','Weekly_Cases']].corr()
```

Out[12]:

	Daily_Cases	Weekly_Cases
Daily_Cases	1.000000	0.990264
Weekly_Cases	0.990264	1.000000

Looking at the data i found that there is a strong relationship between the Daily_Cases and Weekly_Cases

Hypothesis

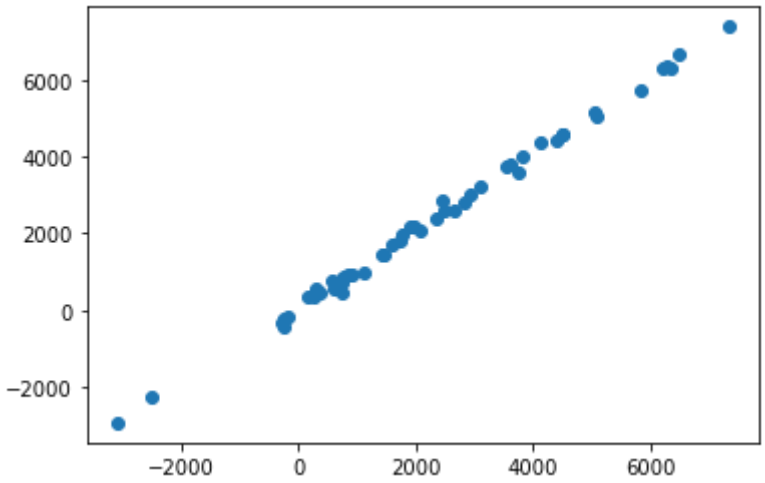
1. Null Hypthesis, **Ho**: Weekly_Deaths are not affected by Daily_Cases
2. Alternative Hypthosesis, **H1**: Weekly_Deaths are affected by Daily_Cases

Visualising related variables

In [9]:
```
#df1=Daily_Cases,std=2367.162193,mean=610.260000
#df2=Weekly_Death, std=163.081701,mean =49.460000
from numpy import mean
from numpy import std
from numpy.random import randn
from numpy.random import seed
from matplotlib import pyplot
# seed random number generator
seed(1)
# prepare data
df1 = 2367.162193 * randn(50) + 2367.162193
df2 = df1 + (163.081701 * randn(50) + 49.460000)
# summarize
print('df1: mean=%.3f stdv=%.3f' % (mean(df1), std(df1)))
print('df2: mean=%.3f stdv=%.3f' % (mean(df2), std(df2)))
# plot
pyplot.scatter(df1, df2)

pyplot.show()
```

```
df1: mean=2306.764 stdv=2295.175
df2: mean=2380.145 stdv=2295.844
```



covariance of the variables

In [10]:
```
np. cov(df1, df2)
```

Out[10]:
```
array([[5375335.4175142 , 5368596.66360723],
       [5368596.66360723, 5378468.6255801 ]])
```

The covariance between the two variables is positive, **5368596.66360723**. suggesting the variables change in the same direction as we expect.

calculate the Pearson's correlation between two variables

In [11]:
```
from numpy.random import randn
from numpy.random import seed
from scipy.stats import pearsonr
# seed random number generator
seed(1)
# prepare data
df1 = 2367.162193 * randn(50) + 2367.162193
df2 = df1 + (163.081701 * randn(50) + 49.460000)
# calculate Pearson's correlation
corr, _ = pearsonr(df1, df2)
print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: 0.998
```

The two variables are positively correlated and that the correlation is **0.998**. This suggests a high level of correlation between cases reported daily and deaths reported weekly, given that the value is above 0.5 and close to 1.0.

Testing Hypothesis

In [12]:
```
#import libraries
import scipy.stats
```

In [13]:
```
r,p =scipy.stats.pearsonr(df1, df2)
r
```

Out[13]:
```
0.9984554060985779
```

In [14]:
```
p
```

Out[14]:
```
6.425586190780168e-62
```

Since the p-value, **0.6426** is 0.5 then I reject Ho and accept the H1 that Deaths reported weekly are arising from cases reported daily

Decision tree

In [37]:
```
%conda install seaborn
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler,PolynomialFeatures
%matplotlib inline
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.9.2
  latest version: 4.10.3

Please update conda by running

    $ conda update -n base conda



# All requested packages already installed.


Note: you may need to restart the kernel to use updated packages.
```

In [38]:
```
africa_df['Transmission_Classification'].value_counts().to_frame()
```

Out[38]:

	Transmission_Classification
Community transmission	47
Pending	1
Clusters of cases	1
No cases	1

Feature before One Hot Encoding

In [39]:
```
import itertools
from matplotlib.ticker import NullFormatter
import matplotlib.ticker as ticker
from sklearn import preprocessing
%matplotlib inline
```

In [40]:
```
africa_df[['Weekly_Deaths','Total_Cases','Weekly_Cases','Daily_Cases', 'Transmission
```

Out[40]:

	Weekly_Deaths	Total_Cases	Weekly_Cases	Daily_Cases	Transmission_Classification
19	1083	1877143	91064	16078	Community transmission
67	34	275601	826	99	Community transmission
81	104	181239	3957	741	Community transmission
83	1	167401	259	70	Community transmission
87	330	140620	18376	3594	Community transmission

Use one hot encoding technique to convert categorical varables to binary variables and append them to the training Data

In [41]:
```
train_africa = africa_df[['Total_Cases','Weekly_Deaths']]
train_africa = pd.concat([train_africa,pd.get_dummies(africa_df['Transmission_Classi
train_africa.drop(['No cases'], axis = 1,inplace=True)
train_africa.drop(['Pending'], axis = 1,inplace=True)
#Feature.drop(['Not applicable'], axis = 1,inplace=True)
train_africa.head()
```

Out[41]:

	Total_Cases	Weekly_Deaths	Clusters of cases	Community transmission
19	1877143	1083	0	1
67	275601	34	0	1
81	181239	104	0	1
83	167401	1	0	1
87	140620	330	0	1

Feature selection

Lets defind feature sets, X:

In [42]:
```
X = train_africa
X[0:5]
```

Out[42]:

	Total_Cases	Weekly_Deaths	Clusters of cases	Community transmission
19	1877143	1083	0	1
67	275601	34	0	1
81	181239	104	0	1
83	167401	1	0	1
87	140620	330	0	1

What are our lables?

In [43]:
```
y = africa_df['Transmission_Classification'].values
y[0:5]
```

Out[43]:
```
array(['Community transmission', 'Community transmission',
       'Community transmission', 'Community transmission',
       'Community transmission'], dtype=object)
```

In [44]:
```
y_collection = africa_df['Transmission_Classification'].replace(to_replace=['Communi
y_collection[0:5]
```

Out[44]:
```
array([2, 2, 2, 2, 2], dtype=object)
```

Normalize Data

Data Standardization give data zero mean and unit variance (technically should be done after train test split

In [45]:
```
from sklearn.preprocessing import StandardScaler
X_initial = X
scaler = preprocessing.StandardScaler().fit(X_initial)
X= scaler.transform(X)
X[0:10]
```

Out[45]:
```
array([[ 6.8479251 ,  6.40190195, -0.14285714,  0.25264558],
       [ 0.75134587, -0.09576156, -0.14285714,  0.25264558],
       [ 0.39213867,  0.33782895, -0.14285714,  0.25264558],
       [ 0.33946165, -0.30016852, -0.14285714,  0.25264558],
       [ 0.23751459,  1.73770689, -0.14285714,  0.25264558],
       [ 0.22526846,  0.09006294, -0.14285714,  0.25264558],
       [ 0.06483918, -0.28158607, -0.14285714,  0.25264558],
       [ 0.01158735,  0.92007906, -0.14285714,  0.25264558],
       [ 0.00860671, -0.24442117, -0.14285714,  0.25264558],
       [-0.01023643,  1.17403922, -0.14285714,  0.25264558]])
```

Import libraries

In [46]:
```
from sklearn import metrics
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import jaccard_score
from sklearn.metrics import f1_score
from sklearn.metrics import log_loss
%conda install six
%conda install pydotplus
import six
import sys
sys.modules['sklearn.externals.six'] = six
from six import StringIO
%conda install graphviz
from sklearn.externals.six import StringIO
import pydotplus
import matplotlib.image as mpimg
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.9.2
  latest version: 4.10.3

Please update conda by running

    $ conda update -n base conda
```

```
# All requested packages already installed.


Note: you may need to restart the kernel to use updated packages.
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.9.2
  latest version: 4.10.3

Please update conda by running

    $ conda update -n base conda



# All requested packages already installed.


Note: you may need to restart the kernel to use updated packages.
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.9.2
  latest version: 4.10.3

Please update conda by running

    $ conda update -n base conda



# All requested packages already installed.


Note: you may need to restart the kernel to use updated packages.
```

Build an empty DecisionTree object with depth 5

In [50]:
```
CovidTree = DecisionTreeClassifier(criterion="entropy", max_depth = 5)
CovidTree # it shows the default parameters
```

Out[50]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=5)
```

In [51]:
```
from sklearn import preprocessing
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import Binarizer
from sklearn.preprocessing import StandardScaler
X= preprocessing.StandardScaler().fit(X).transform(X)
X[0:5]
```

Out[51]:
```
array([[ 6.8479251 ,  6.40190195, -0.14285714,  0.25264558],
       [ 0.75134587, -0.09576156, -0.14285714,  0.25264558],
       [ 0.39213867,  0.33782895, -0.14285714,  0.25264558],
       [ 0.33946165, -0.30016852, -0.14285714,  0.25264558],
       [ 0.23751459,  1.73770689, -0.14285714,  0.25264558]])
```

Train the decision tree using the global X data set.

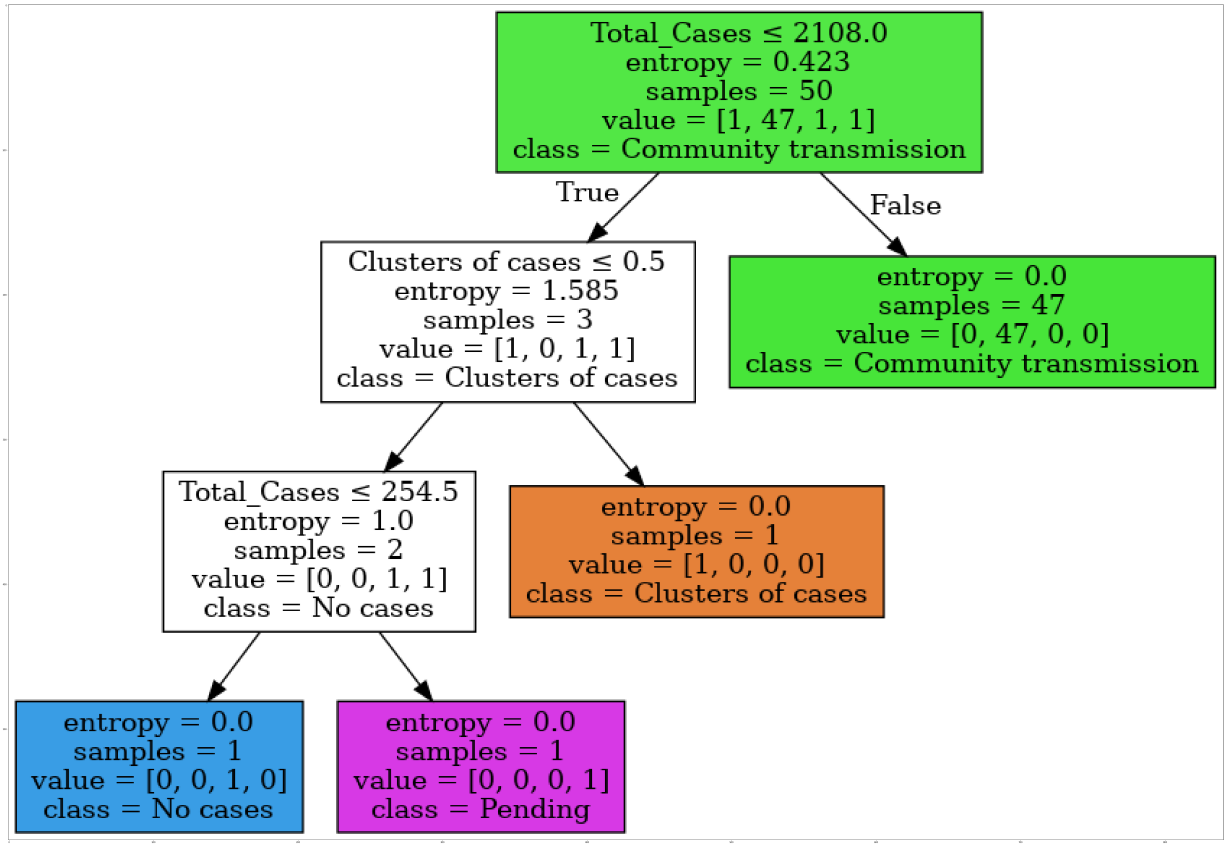
In [52]:
```
CovidTree.fit(X_initial, y)
```

Out[52]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=5)
```

Print the Decision Tree

In [53]:
```
dot_data = StringIO()
filename = "TCtree.png"
featureNames = train_africa.columns
targetNames = africa_df['Transmission_Classification'].tolist()
out=tree.export_graphviz(CovidTree,feature_names=featureNames, out_file=dot_data, cl
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png(filename)
img = mpimg.imread(filename)
plt.figure(figsize=(100, 200))
plt.imshow(img,interpolation='nearest')
```

Out[53]:
```
<matplotlib.image.AxesImage at 0x7fc744fa9080>
```



In [56]:
```
from sklearn.datasets import load_iris
from sklearn import tree
clf = tree.DecisionTreeClassifier()
iris = load_iris()
clf = clf.fit(iris.data, iris.target)
tree.export_graphviz(clf, out_file='tree.dot')
```

Evaluating the Accuracy of Decison Tree Algorithm

Use Decision Tree object previously trained.

In [57]:
```
yhat_tree=CovidTree.predict(X_initial)
tree_jacc_test=metrics.jaccard_score(y, yhat_tree, average='micro')
tree_f1_test=metrics.f1_score(y, yhat_tree, average ='macro')
```

```
tree_cnf_matrix = confusion_matrix(y, yhat_tree)
print("Accuracy is ", tree_jacc_test, " F1 is" , tree_f1_test)
print(classification_report(y, yhat_tree))
tree_cnf_matrix
```

```
Accuracy is  1.0  F1 is 1.0
                        precision    recall  f1-score   support

     Clusters of cases       1.00      1.00      1.00         1
Community transmission       1.00      1.00      1.00        47
             No cases        1.00      1.00      1.00         1
               Pending       1.00      1.00      1.00         1

              accuracy                           1.00        50
             macro avg       1.00      1.00      1.00        50
          weighted avg       1.00      1.00      1.00        50
```

Out[57]:
```
array([[ 1,  0,  0,  0],
       [ 0, 47,  0,  0],
       [ 0,  0,  1,  0],
       [ 0,  0,  0,  1]])
```

REMARKS

The Accuracy is 1.0 and F1 is 1.0 ::indicating that the Community Transmission mode is the most significance and dangerous modes of transmission of Covid-19 which reqiuire interventions from government.

End

By: Sandile Mduduzi Mchunu