AI Environment Setup

Group 3 — Cohort 2 (3C2)

January 2021

GROUP MEMBERS

1. David Norman Amatey Masoperh (Documentation)
2. Mahalinoro Razafimanjato(Tester)
3. Wendy Essuman(Installation Video Presentation)
4. Goodluck Caiser Malata(Tester)

1 Introduction

The world is moving at an unprecedented level of speed. Day in and day out we are producing enormous amounts of data . Unfortunately, given the large amounts of data being produced, more often than not we are unable to analyze a lot of it and hence lose the opportunity to mine valuable insights from this data . In the document below the basic tools to begin some analysis of this data are brought to light and experimented with . The aim of this document is to be a guide to putting together a compelling arsenal of tools with which to kickstart a journey in Artificial Intelligence.

Figure 1: The Universe

Within the area of Artificial Intelligence and data science there are a myriad of tools employed by Developers and other practitioners of the sciences . These tools are essential in easing and facilitating the derivation of insight and making predictions based on data.

In our quest to familiarise ourselves with the latest and greatest tools the team had to install the following on a linux system :

1. Anaconda
2. Python
3. Jupyter Notebook
4. Sklearn
   (a) Numpy
   (b) Scipy
   (c) Scikit-learn
   (d)
5. TensorFlow
6. Keras

7. Matplotlib
8. Pandas

2 Installation

Below lies a detailed description of the tool and the installation process via the Linux (Ubuntu) terminal:

- **Anaconda:** Anaconda can be thought of as an open source bundle of popular python data science and machine learning packages . Using the platform ensures that the user is able to manage different packages as well as deploy them.

  To install Anaconda follow the following steps in your terminal:

  1. Enter "bash /Downloads/Anaconda3-filename"[1]
  2. Review the terms and conditions and answer "Yes"
  3. Answer "Yes" to the prompt initialize Anaconda3 by running conda init ?
  4. Close your terminal and open a new window

- **Python:** Python is a multipurpose object oriented programming language and is one the most preferred programming languages for implementing machine learning algorithms.

  To install Python follow the following steps in your terminal:

  1. Enter "Sudo apt install python3-pip"
  2. Enter your system password
  3. Answer "Yes" to the allocation of disk space to the python package
  4. Type "python 3" in your terminal to ensure it has been installed.

- **Jupyter Notebook:** Jupyter notebook is an open source web application that allows a user to share and preview live code , mathematical equations and perform numerous other functions . It can be thought of as a multi purpose source code editor.

  To install Jupyter Notebook follow the following steps in your linux terminal:

  1. Enter "Sudo apt install jupyter-notebook[2]"
  2. Answer "Yes" to the to the allocation of disk space to the jupyter-notebook package

- **Sklearn/Scikit-learn:** This is a machine learning python library built on Matplotlib, a python library for representing data by plotting , Numpy, a library for scientific computing mainly using arrays and Scipy, a library for scientific computing with different modules for various mathematical computations.

  To install Sklearn/Scikit-learn follow the following steps in your terminal:

  1. pip3 install -U scikit-learn [7].

- **Tensorflow:** TensorFlow is an open source machine learning python library that empowers users to make fast numerical computations.

  To install TensorFlow follow the following steps in your terminal:

  1. pip3 install –upgrade tensorflow[8]

- **Keras:** Keras is a deep learning API written in python , it plays the role of an interface for the TensorFlow library.

  To install keras follow the following steps in your terminal:

  1. pip3 install keras[3]

- **Pandas:** Pandas is an open-source Python library for manipulating data and performing analysis it is popularly used for its tabulating feature.

  To install Pandas follow the following steps in your terminal:

  1. pip3 install Pandas[6]

3 Testing

Upon completion of the installation of the packages, testing the packages was necessary to ensure that they were working as intended. To do this we initialized Anaconda with the "anaconda-navigator" command in the terminal. Anaconda opens to reveal a GUI with different python packages and apps . For the purpose of testing we opened " Jupyter Notebook" and used the data set Crime Statistics for South Africa[5] . This was done with the aim of extracting valuable insight about reported crime in South Africa , insights with which to aim law enforcement with.

Below are some annotated screenshots detailing the testing process and various insights accumulated.

- **Jupyter Notebook:**

```
# importing the libraries
import tensorflow as tf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats

%matplotlib inline
```

The first order of business was the importation of the libraries to be used within the notebook these libraries were loaded in with conventional reference names eg. tensorflow as tf.

- **Pandas:**

```
# Loading the data

# Using pandas and load the dataset as a pandas Dataframe
data = pd.read_csv("SouthAfricaCrimeStats_v2.csv")
# head() method will display the first 5 elements by default if no parameter has been set
data.head()
```

	Province	Station	Category	2005-2006	2006-2007	2007-2008	2008-2009	2009-2010	2010-2011	2011-2012	2012-2013	2013-2014	2014-2015	2015-2016
0	Western Cape	Cape Town Central	All theft not mentioned elsewhere	6692	6341	5966	5187	4985	5127	5285	5937	5600	5335	5176
1	Gauteng	Jhb Central	All theft not mentioned elsewhere	6093	4602	3761	3610	3267	3037	2886	2638	2809	3050	2434
2	Western Cape	Mitchells Plain	All theft not mentioned elsewhere	5341	6093	6316	6803	6035	5761	6108	5514	4975	4043	3635
3	Free State	Park Road	All theft not mentioned elsewhere	5108	4282	3834	3316	3101	3013	2679	3116	2927	2297	2103
4	Gauteng	Pretoria Central	All theft not mentioned elsewhere	5099	4536	3309	2694	2616	2606	2635	3226	3246	2892	3030

```
# Number of rows and columns
data.shape
```

```
(30861, 14)
```

Pandas was used to view the contents of the csv file in tabular form in the image above only the first 5 records are displayed.

```
# Describing data to view basic statistical details
data.describe()
```

	2005-2006	2006-2007	2007-2008	2008-2009	2009-2010	2010-2011	2011-2012	2012-2013	2013-2014	2014-2015	20
count	30861.000000	30861.000000	30861.000000	30861.000000	30861.000000	30861.000000	30861.000000	30861.000000	30861.000000	30861.000000	30
mean	70.527753	69.301610	67.154305	68.756165	69.517773	67.766696	68.259616	69.700658	71.416999	71.498202	70
std	205.491698	198.037635	186.760510	187.173860	185.514638	181.865878	183.334468	184.812420	187.635207	185.019046	17
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	1.000000	1.000000	1.000000	1.000000	1.
50%	8.000000	8.000000	8.000000	9.000000	9.000000	9.000000	10.000000	10.000000	11.000000	11.000000	11
75%	49.000000	50.000000	50.000000	52.000000	54.000000	52.000000	53.000000	55.000000	56.000000	57.000000	58
max	6692.000000	6341.000000	6316.000000	6803.000000	6571.000000	6260.000000	6108.000000	6310.000000	6044.000000	5335.000000	51

Using the ".describe" function we were able to further acquire some basic descriptive data on our data set such as the mean of number of crimes , total number of crimes in a given year etc.

- **Numpy:** To further gain more meaningful insight on the data, we asked the question , " What is the average number of reported crimes for each category in descending order"

```
# Problem 2 => The average number of reported crime over the years for each crime category in descending order

# 1. Create a copy of the original data
# 2. Group it by category using the groubpy() and sum method
# 3. Using lambda and numpy mean() method to calculate the average for each crime category
# 4. Using sort_values() method to sort the values in descending order
df_avg = data.loc[:, ['Category', '2005-2006', '2006-2007',
       '2007-2008', '2008-2009', '2009-2010', '2010-2011', '2011-2012',
       '2012-2013', '2013-2014', '2014-2015', '2015-2016']]
df_avg = df_avg.groupby(["Category"]).sum().reset_index()
rows = ['2005-2006', '2006-2007',
       '2007-2008', '2008-2009', '2009-2010', '2010-2011', '2011-2012',
       '2012-2013', '2013-2014', '2014-2015', '2015-2016']
df_avg['avg'] = df_avg.apply(lambda x: np.mean(x[rows]), axis=1)
df_avg = df_avg.sort_values(by='avg', ascending=False)
df_avg.loc[:, ['Category', 'avg']]
```

	Category	avg
0	All theft not mentioned elsewhere	374577.363636
6	Burglary at residential premises	251268.181818
2	Assault with the intent to inflict grievous bo...	198109.727273
9	Common assault	185751.545455
12	Drug-related crime	170897.363636
25	Theft out of or from motor vehicle	129162.636364
14	Malicious damage to property	127076.818182
19	Robbery with aggravating circumstances	116817.363636
8	Commercial crime	73382.363636
22	Shoplifting	72552.636364
5	Burglary at non-residential premises	68337.909091
24	Theft of motor vehicle and motorcycle	67748.363636
20	Sexual Offences	61668.000000
11	Driving under the influence of alcohol or drugs	60015.818182
10	Common robbery	58885.363636

The numpy mean() method was used and this revealed that apart from General theft the highest crimes were home invasions (Burglary at residential premises).

- **Matplotlib:** To further get a clearer understanding of the data in a descriptive nature, we employed the use of matplotlib features of the graph such as the colors, linewidth and .

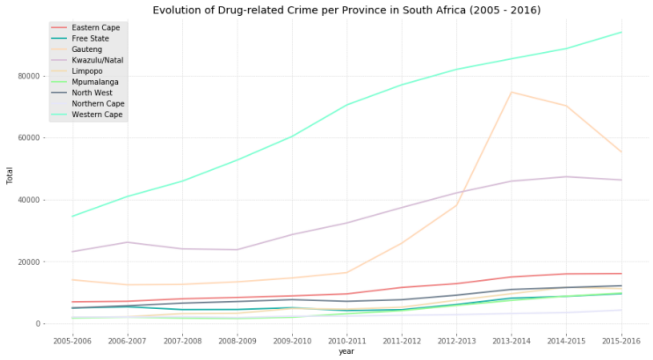
```
# Plotting the evolution of total numbers of "Drug-related crime" reported over the years per province
plt.style.use("ggplot")
col = ['Eastern Cape', 'Free State', 'Gauteng', 'Kwazulu/Natal',
       'Limpopo', 'Mpumalanga', 'North West', 'Northern Cape',
       'Western Cape']

fig, ax = plt.subplots(figsize=(15,8))
ax.plot(df_drug['index'], df_drug['Eastern Cape'], color='lightcoral', linewidth=2)
ax.plot(df_drug['index'], df_drug['Free State'], color='lightseagreen', linewidth=2)
ax.plot(df_drug['index'], df_drug['Gauteng'], color='peachpuff', linewidth=2)
ax.plot(df_drug['index'], df_drug['Kwazulu/Natal'], color='thistle', linewidth=2)
ax.plot(df_drug['index'], df_drug['Limpopo'], color='wheat', linewidth=2)
ax.plot(df_drug['index'], df_drug['Mpumalanga'], color='palegreen', linewidth=2)
ax.plot(df_drug['index'], df_drug['North West'], color='slategray', linewidth=2)
ax.plot(df_drug['index'], df_drug['Northern Cape'], color='lavender', linewidth=2)
ax.plot(df_drug['index'], df_drug['Western Cape'], color='aquamarine', linewidth=2)

ax.set_xlabel('year', fontsize=10, color='black')  # x label
ax.set_ylabel('Total', fontsize=10, color='black') # y label
ax.set_title("Evolution of Drug-related Crime per Province in South Africa (2005 - 2016)") # graph title
ax.legend(col, loc ="best")  # graph legend

ax.set_facecolor('white')
ax.grid(color='lightgrey', linestyle='--', linewidth=0.5)
```

The question above sought to determine the evolution of drug related crimes reported over the years within the provinces.



The plot above revealed some interesting insights, it shows "Free State" has seen a continuous upsurge in drug related crime whereas Gauteng has seen a sharp decline in drug related crime .

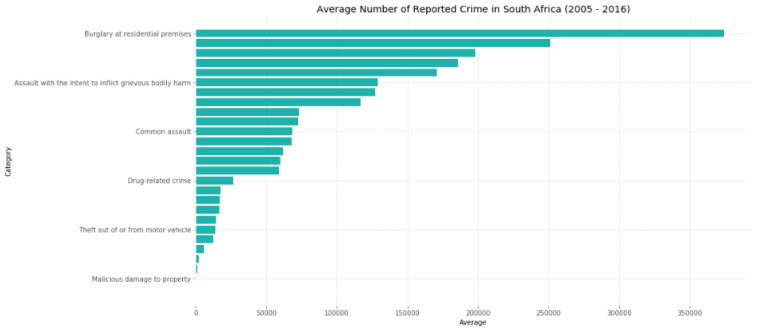
Furthermore, using a barchart we were able to determine the average number of reported crimes in South Africa across all crime categories using the various matplotlib methods we were able to assign axises to metrics such as "Category" and "Average number of reported crimes in South Africa between 2005 and 2016" .

```
# Plotting the average number of reported crime in South Africa
fig2, ax2 = plt.subplots(figsize=(15,8))

ax2.barh(np.arange(len(df_avg['Category'])), df_avg['avg'],align='center', color='lightseagreen')
ax2.set_yticklabels(df_avg['Category'])
ax2.invert_yaxis()# labels read top-to-bottom

ax2.set_xlabel('Average', fontsize=10, color='black')  # x label
ax2.set_ylabel('Category', fontsize=10, color='black') # y label
ax2.set_title("Average Number of Reported Crime in South Africa (2005 - 2016)") # graph title
ax.legend(col, loc ="best") # graph legend

ax2.set_facecolor('white')
ax2.grid(color='lightgrey', linestyle='--', linewidth=0.5)
```



- **Scipy:** Scipy as described above allows the user to perform scientific manipulations to data in the image below we used this library to give a descriptive comparison between the years of 2005/6 and 2009/10 . Using the "describe()" function .

```
# importing the libraries
from scipy.stats import ttest_ind
from scipy.stats import describe

# Data from the dataset column '2005-2006' and '2009-2010'
v1 = df_avg['2005-2006']
v2 = df_avg['2009-2010']

# Using describe function to find the number of observations, the minimum and maximum number,
# the mean, the variance, the skewness and the kurtosis of both variables
rv1 = describe(v1)
rv2 = describe(v2)

# T-test on both variables to check if there is a significant difference between the means of two groups
rt = ttest_ind(v1, v2)

# Results
rv2, rv1, rt
```

```
(DescribeResult(nobs=27, minmax=(0, 360120), mean=79458.81481481482, variance=7885203217.772079, skewness=1.531107577794771
8, kurtosis=2.0658769553510297),
 DescribeResult(nobs=27, minmax=(0, 424690), mean=80613.22222222222, variance=10295237946.79487, skewness=1.822272697825096,
kurtosis=3.1546482408741),
 Ttest_indResult(statistic=0.04448757960247004, pvalue=0.9646861534848246))
```

- **Scikit-learn:** Being a machine learning library we were able to use the data in a K Nearest neighbour algorithm . This algorithm is used to classify a given data based on its nearest classified neighbors.

```
from sklearn.model_selection import train_test_split # For train and test splitting
from sklearn.neighbors import KNeighborsClassifier # Knn classifier model

df_learn =  data.loc[:, ['Province', '2005-2006', '2006-2007',
       '2007-2008', '2008-2009', '2009-2010', '2010-2011', '2011-2012',
       '2012-2013', '2013-2014', '2014-2015', '2015-2016']]

X = df_learn.loc[:, ['2005-2006', '2006-2007',
       '2007-2008', '2008-2009', '2009-2010', '2010-2011', '2011-2012',
       '2012-2013', '2013-2014', '2014-2015', '2015-2016']] # Features
y = df_learn['Province'] # Target value

# Splitting the dataset into training and testing sets where the test size is 25%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
knn = KNeighborsClassifier(n_neighbors = 5) # Using 5 n neighbors as parameter
knn.fit(X_train, y_train) # Fitting the training sets into the knn classifier model
prediction = knn.predict(X_test) # Predicting the value using the test set

prediction # Result of the prediction
```

```
array(['Western Cape', 'Limpopo', 'Free State', ..., 'North West',
       'Western Cape', 'Kwazulu/Natal'], dtype=object)
```

- **Tensorflow and Keras:** Tensorflow and keras are typically used for the creation of neural networks in the example below however it was used to transform ("crime data from 2005- 2006") the data in a 2d array and a tensor for easier manipulation.

```
# Basic calculation with Tensors
a = df_learn['2005-2006'].iloc[3:7].values
a = a.reshape(2, 2) # Reshaping the array into 2d matrix
a_tensor = tf.convert_to_tensor(a) # Converting the array into a tensor

b = df_learn['2005-2006'].iloc[8:12].values
b = b.reshape(2, 2)
b_tensor = tf.convert_to_tensor(b)

print(tf.add(a_tensor, b_tensor), "\n") # Adding two tensors
print(tf.multiply(a_tensor, b_tensor), "\n") # Multiplying two tensors
print(a_tensor @ b_tensor, "\n")  # Matrix multiplication with two tensors
```

```
strg = data['Category'].unique()
strg_tensor = tf.convert_to_tensor(strg) # Converting the array of strings into a tensor of strings
tf.strings.split(strg_tensor) # Splitting the tensor of strings into individual strings
```

Upon transforming the data simple operations such as multiplication and addition were performed using the tensors flow methods tf.add() and tf.multiply().[4]

Keras was used to create a simple sequential model that had 3 layers .

```
# Creating a simple sequential model with Keras
# A sequential model with 3 layers

model = keras.Sequential(
    [
        layers.Dense(2, activation="relu", name="layer1"),
        layers.Dense(3, activation="relu", name="layer2"),
        layers.Dense(4, name="layer3"),
    ]
)
# Call model on a test input
x = df_learn['2005-2006'].iloc[1:26].values
x = x.reshape(5, 5)
x_tensor = tf.convert_to_tensor(x)
y = model(x)

model.summary()
```

4 Conclusion

The tools and techniques used above are mainly used in the descriptive and exploratory manner with the aim of just providing some basic insight into their potential use cases . Moving forward however, with a grounded footing in more advanced concepts they will prove to be useful in the area of predictive and prescriptive analysis.

References

[1] Anaconda. *Install anaconda on linux: Install anaconda*. URL: `https://docs.anaconda.com/anaconda/install/linux/`. (accessed: 01.23.2021).

[2] Jupyter.org. *anaconda: Install jupyter on linux*. URL: `https://jupyter.org/install`. (accessed: 01.23.2021).

[3] keras. *Keras: Install keras*. URL: `https://phoenixnap.com/kb/how-to-install-keras-on-linux`. (accessed: 01.23.2021).

[4] mahalinoro. *environment creation: install environment*. URL: `https://github.com/Mahalinoro/ai-environment-creation-and-testing`. (accessed: 01.23.2021).

[5] South Africa police. *crime-statistics-for-south-africa*. URL: `https://www.kaggle.com/slwessels/crime-statistics-for-south-africa?select=SouthAfricaCrimeStats_v2.csv`. (accessed: 01.23.2021).

[6] pydata.org. *pandas: install pandas*. URL: `https://pandas.pydata.org/`. (accessed: 01.23.2021).

[7] scikit.org. *scikit: Install scikit on linux*. URL: `https://scikit-learn.org/stable/`. (accessed: 01.23.2021).

[8] Tensorflow. *Tensorflow: Install Tensorflow*. URL: `https://www.tensorflow.org/install/pip`. (accessed: 01.23.2021).