CSC 5741: Lecture #07—Linear Regression, Classification and Clustering

Lighton Phiri
<lighton.phiri@unza.zm>

May 14 2019

Contents

Introduction

During these "hands-on" activities, we look at a popular clustering algorithm: K Mean Clustering.

In all instances, you are encouraged to make reference to online Python documentation and documentation for specific libraries. You are also encouraged to look up and explore other libraries, especially as you work towards the Mini Projects.

[1]:
```
# Import all libraries and modules for use during lecture session code walkthrough
import pandas as pd
import re
import string

from collections import Counter
from IPython.core.interactiveshell import InteractiveShell
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt

plt.style.use("ggplot")

InteractiveShell.ast_node_interactivity = "all"
pd.set_option('display.latex.repr', True)
pd.set_option('display.latex.longtable', True)
```

Datasets

- We shall work with the Jesuit Centre for Theological Reflection (JCTR) Basic Needs Basket (BnB) historical amounts to explore patterns from various regions in the Republic of Zambia.
- You are also encouraged to make use of "Toy Datasets" available in the Scikit Learn library.

[2]:
```
# Zambian towns BnB amounts
!head -n 5 db-jctr-bnb-zambia.csv | cat -n
```

```
1  Town|Date|Amount
2  Chinsali|Nov 16|2837.4
3  Chinsali|Dec 16|2788.35
4  Chinsali|Jan 17|2728
5  Chinsali|Feb 17|2740.65
```

Import Dataset Using Pandas

[3]:
```
# Convert CSV dataset into pandas DataFrame
var_jctr_bnb_zambia = pd.read_csv("db-jctr-bnb-zambian_towns.csv", sep="|")
```

[4]:
```
var_jctr_bnb_zambia.head(2)
```

[4]:

	Town	Date	Amount
0	Chinsali	Nov 16	2837.40
1	Chinsali	Dec 16	2788.35

Data Cleaning

[5]:
```
# 1. Checking for NULL values
var_jctr_bnb_zambia.isnull().values.any()
var_jctr_bnb_zambia[var_jctr_bnb_zambia.isna().any(axis=1)]

# 2. Get aggregate means per town
var_jctr_bnb_zambia.groupby("Town").mean()

# 3. Replace NaN entries using group aggregates
var_jctr_bnb_zambia["Amount"] = var_jctr_bnb_zambia.groupby("Town").transform(lambda␣
 ↪var_x: var_x.fillna(var_x.mean()))
```

[5]:
```
True
```

[5]:

	Town	Date	Amount
9	Chinsali	Aug 17	NaN
10	Chinsali	Sep 17	NaN
18	Chipata	Nov 16	NaN
53	Choma	Apr 18	NaN
71	Kabwe	Apr 18	NaN
77	Kasama	Apr 17	NaN

Continued on next page

	Town	Date	Amount
90	Kitwe	Nov 16	NaN
125	Livingstone	Apr 18	NaN
126	Luanshya	Nov 16	NaN
130	Luanshya	Mar 17	NaN
137	Luanshya	Oct 17	NaN
170	Mansa	Jul 17	NaN
171	Mansa	Aug 17	NaN
179	Mansa	Apr 18	NaN
185	Mongu	Apr 17	NaN
188	Mongu	Jul 17	NaN
209	Monze	Oct 17	NaN
224	Mpika	Jul 17	NaN
225	Mpika	Aug 17	NaN
257	Solwezi	Apr 17	NaN
263	Solwezi	Oct 17	NaN

[5]:

Town	Amount
Chinsali	2922.953750
Chipata	2563.340000
Choma	3720.444706
Kabwe	3501.190588
Kasama	3174.467647
Kitwe	3925.548235
Livingstone	3904.187647
Luanshya	3755.454000
Lusaka	5043.474444
Mansa	2971.382000
Mongu	3042.466875
Monze	3702.336471
Mpika	2939.607500
Ndola	4805.334444
Solwezi	4106.669375

[6]:
```
# 4. Verify that NaN entries have been replaced
var_jctr_bnb_zambia[var_jctr_bnb_zambia.isna().any(axis=1)]
var_jctr_bnb_zambia[(var_jctr_bnb_zambia["Town"]=="Chinsali") &␣
 ↪(var_jctr_bnb_zambia["Date"]=="Aug 17")]
```

[6]:
```
Empty DataFrame Columns: Index(['Town', 'Date', 'Amount'], dtype='object') Index: Int64Index([], dtype='int64')
```

[6]:

	Town	Date	Amount
9	Chinsali	Aug 17	2922.95375

[7]:
```
# 5. Convert date strings to date objects
var_jctr_bnb_zambia["Date"][0]
type(var_jctr_bnb_zambia["Date"][0])

# df['col'] = pd.to_datetime(df['col'])
# Please see documentation [1, 2] for details on working with date strings
# [1] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.to_datetime.html
# [1] https://docs.python.org/3/library/datetime.html
###pd.to_datetime(var_jctr_bnb_zambia["Date"], format="%b %y").apply(lambda var_x:␣
 ↪var_x.toordinal())

# Replace date string with equivalent date objects
var_jctr_bnb_zambia["DateX"] = pd.to_datetime(var_jctr_bnb_zambia["Date"], format="%b␣
 ↪%y")
```

[7]:
```
'Nov 16'
```

[7]:
```
str
```

[8]:
```
# 6. Convert amount strings to numeric values
var_jctr_bnb_zambia["Amount"][0]
type(var_jctr_bnb_zambia["Amount"][0])
```

[8]:
```
2837.4
```

[8]:
```
numpy.float64
```

[9]:
```
var_jctr_bnb_zambia.head(2)
```

[9]:

	Town	Date	Amount	DateX
0	Chinsali	Nov 16	2837.40	2016-11-01
1	Chinsali	Dec 16	2788.35	2016-12-01

K Means Clustering

Example #1: 2017 Town BnBs Clustering Using K Means

[ ]:
```

```

[10]:
```
# 1. Exploratory Data Analysis of data
var_jctr_bnb_zambia_2017 = var_jctr_bnb_zambia[var_jctr_bnb_zambia["Date"].str.
 ↪contains("17") & var_jctr_bnb_zambia["Date"].str.contains("Dec")]
plt.scatter(var_jctr_bnb_zambia_2017["Town"], var_jctr_bnb_zambia_2017["Amount"])
plt.xticks(rotation=90)
plt.title("2017 JCTR BnB by Town Scatter Plot")
plt.xlabel("Town")
plt.ylabel("Amount (ZMW)")
```

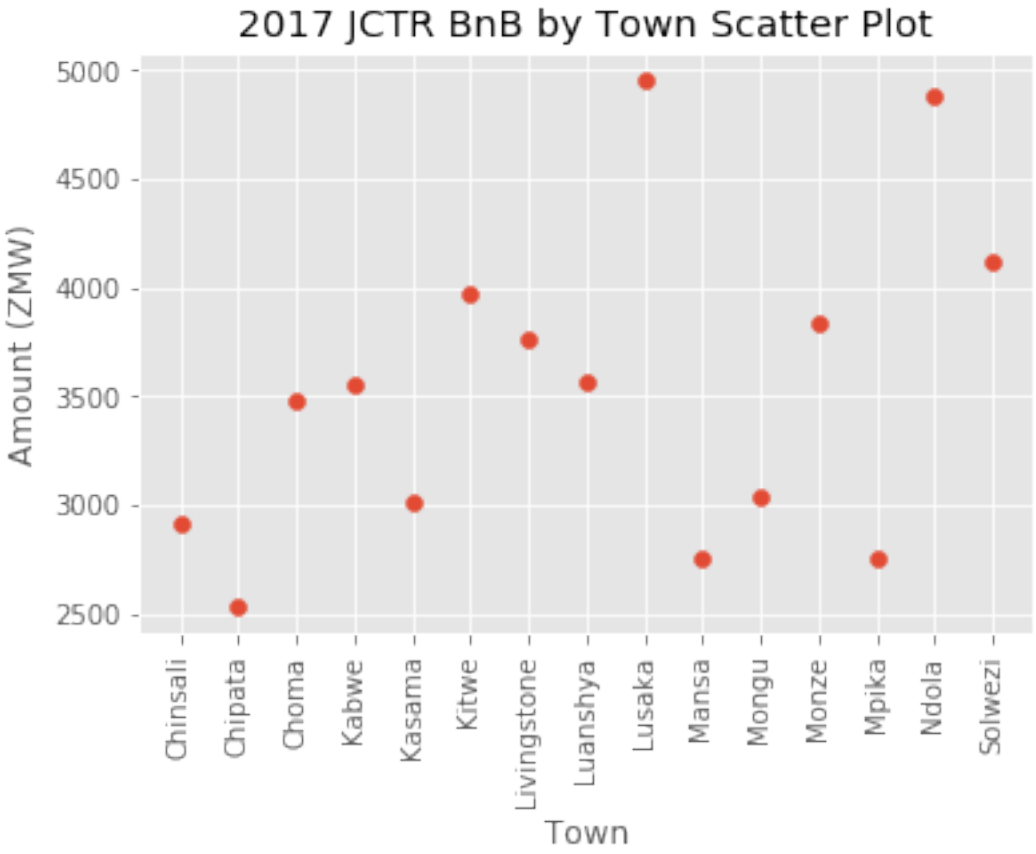
[10]:
```
<matplotlib.collections.PathCollection at 0x7f7ae072ea90>
```

[10]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],
 <a list of 15 Text xticklabel objects>)
```

[10]:
```
Text(0.5, 1.0, '2017 JCTR BnB by Town Scatter Plot')
```

[10]:
```
Text(0.5, 0, 'Town')
```

[10]:
```
Text(0, 0.5, 'Amount (ZMW)')
```



[11]:
```
# Notice how the scatter plot suggests that there are roughly three clusters
```

[12]:
```
var_jctr_bnb_zambia[var_jctr_bnb_zambia["Date"].str.contains("17") &␣
 ↪var_jctr_bnb_zambia["Date"].str.contains("Dec")]
```

[12]:

	Town	Date	Amount	DateX
13	Chinsali	Dec 17	2911.25	2017-12-01
31	Chipata	Dec 17	2526.25	2017-12-01
49	Choma	Dec 17	3476.39	2017-12-01
67	Kabwe	Dec 17	3547.39	2017-12-01
85	Kasama	Dec 17	3006.61	2017-12-01
103	Kitwe	Dec 17	3973.76	2017-12-01
121	Livingstone	Dec 17	3756.51	2017-12-01

Continued on next page

	Town	Date	Amount	DateX
139	Luanshya	Dec 17	3568.94	2017-12-01
157	Lusaka	Dec 17	4957.47	2017-12-01
175	Mansa	Dec 17	2751.78	2017-12-01
193	Mongu	Dec 17	3033.33	2017-12-01
211	Monze	Dec 17	3836.30	2017-12-01
229	Mpika	Dec 17	2751.78	2017-12-01
247	Ndola	Dec 17	4883.65	2017-12-01
265	Solwezi	Dec 17	4115.17	2017-12-01

[13]:
```
var_jctr_bnb_zambia["Town"].unique()
```

[13]:
```
array(['Chinsali', 'Chipata', 'Choma', 'Kabwe', 'Kasama', 'Kitwe',
       'Livingstone', 'Luanshya', 'Lusaka', 'Mansa', 'Mongu', 'Monze',
       'Mpika', 'Ndola', 'Solwezi'], dtype=object)
```

Fit the Data and Predict Clusters

[14]:
```
# 1. Create KMeans object
var_jctr_bnb_zambia_2017_kmeans = KMeans(n_clusters=3)
var_jctr_bnb_zambia_2017_kmeans
```

[14]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

[15]:
```
# 2. Fit the input data on the KMeans object
var_jctr_bnb_zambia_2017_kmeans.fit(var_jctr_bnb_zambia_2017["Amount"].values.
 ↪reshape(-1, 1))
```

[15]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

[16]:
```
# 3. Predict the clusters
var_jctr_bnb_zambia_2017_clusters = var_jctr_bnb_zambia_2017_kmeans.
 ↪predict(var_jctr_bnb_zambia_2017["Amount"].values.reshape(-1, 1))
```

[17]:
```
# 4. Inspect clusters
var_jctr_bnb_zambia_2017_clusters
```

[17]:
```
array([1, 1, 0, 0, 1, 0, 0, 0, 2, 1, 1, 0, 1, 2, 0], dtype=int32)
```

[18]:
```
# 5. Merge predictions with original dataset
var_jctr_bnb_zambia_2017["clusters"] = var_jctr_bnb_zambia_2017_clusters
```

```
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
```

[19]:
```
# 6. Inspect merged DataFrame
var_jctr_bnb_zambia_2017
```

[19]:

	Town	Date	Amount	DateX	clusters
13	Chinsali	Dec 17	2911.25	2017-12-01	1
31	Chipata	Dec 17	2526.25	2017-12-01	1
49	Choma	Dec 17	3476.39	2017-12-01	0
67	Kabwe	Dec 17	3547.39	2017-12-01	0
85	Kasama	Dec 17	3006.61	2017-12-01	1
103	Kitwe	Dec 17	3973.76	2017-12-01	0
121	Livingstone	Dec 17	3756.51	2017-12-01	0
139	Luanshya	Dec 17	3568.94	2017-12-01	0
157	Lusaka	Dec 17	4957.47	2017-12-01	2
175	Mansa	Dec 17	2751.78	2017-12-01	1
193	Mongu	Dec 17	3033.33	2017-12-01	1
211	Monze	Dec 17	3836.30	2017-12-01	0
229	Mpika	Dec 17	2751.78	2017-12-01	1
247	Ndola	Dec 17	4883.65	2017-12-01	2
265	Solwezi	Dec 17	4115.17	2017-12-01	0

Plot Results

[20]:
```
plt.scatter(var_jctr_bnb_zambia_2017["Town"], var_jctr_bnb_zambia_2017["Amount"],␣
 ↪c=var_jctr_bnb_zambia_2017["clusters"], cmap="rainbow")
plt.xticks(rotation=90)
plt.title("2017 JCTR BnB by Town Cluster Plot")
plt.xlabel("Town")
plt.ylabel("Amount (ZMW)")
```

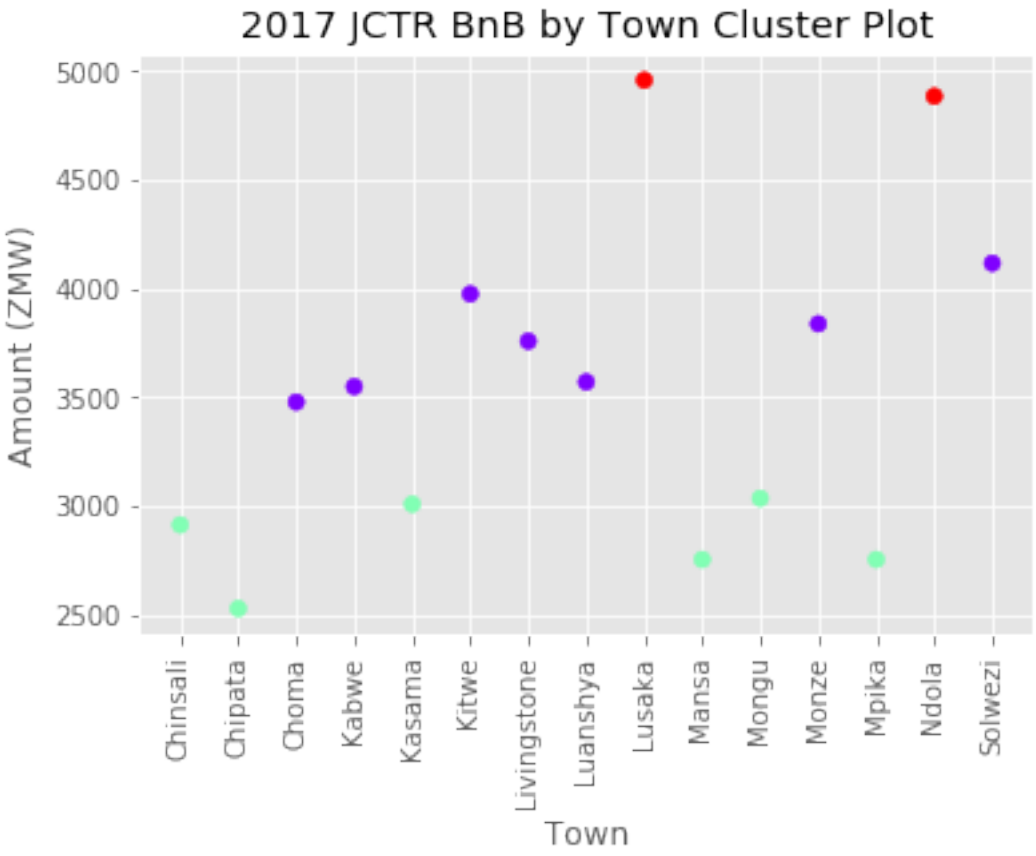
[20]:
```
<matplotlib.collections.PathCollection at 0x7f7adce71780>
```

[20]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],
 <a list of 15 Text xticklabel objects>)
```

[20]:
```
Text(0.5, 1.0, '2017 JCTR BnB by Town Cluster Plot')
```

[20]:
```
Text(0.5, 0, 'Town')
```

[20]:
```
Text(0, 0.5, 'Amount (ZMW)')
```



Clustering with Scaled Values

[21]:
```
# 1. Create instance of MinMaxScaler
var_jctr_bnb_zambia_2017_scaler = MinMaxScaler()
```

[22]:
```
# 2. Scale Amounts to be between 0 and 1
var_jctr_bnb_zambia_2017_scaler.fit(var_jctr_bnb_zambia_2017["Amount"].values.
 ↪reshape(-1, 1))
```

[22]:
```
MinMaxScaler(copy=True, feature_range=(0, 1))
```

[23]:
```
# 3. Transform the BnB amounts
var_jctr_bnb_zambia_2017["Amount_Scaled"] = var_jctr_bnb_zambia_2017_scaler.
 ↪transform(var_jctr_bnb_zambia_2017["Amount"].values.reshape(-1, 1))
```

```
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
```

[24]:
```
# 4. Inspect modified DataFrame to ensure amounts are within 0--1 range
var_jctr_bnb_zambia_2017
```

[24]:

	Town	Date	Amount	DateX	clusters	Amount_Scaled
13	Chinsali	Dec 17	2911.25	2017-12-01	1	0.158357
31	Chipata	Dec 17	2526.25	2017-12-01	1	0.000000
49	Choma	Dec 17	3476.39	2017-12-01	0	0.390808
67	Kabwe	Dec 17	3547.39	2017-12-01	0	0.420011
85	Kasama	Dec 17	3006.61	2017-12-01	1	0.197580
103	Kitwe	Dec 17	3973.76	2017-12-01	0	0.595384
121	Livingstone	Dec 17	3756.51	2017-12-01	0	0.506026
139	Luanshya	Dec 17	3568.94	2017-12-01	0	0.428875
157	Lusaka	Dec 17	4957.47	2017-12-01	2	1.000000
175	Mansa	Dec 17	2751.78	2017-12-01	1	0.092764
193	Mongu	Dec 17	3033.33	2017-12-01	1	0.208570
211	Monze	Dec 17	3836.30	2017-12-01	0	0.538845
229	Mpika	Dec 17	2751.78	2017-12-01	1	0.092764
247	Ndola	Dec 17	4883.65	2017-12-01	2	0.969637
265	Solwezi	Dec 17	4115.17	2017-12-01	0	0.653548

[25]:
```
# 5. Create KMeans object
var_jctr_bnb_zambia_2017_scaled_kmeans = KMeans(n_clusters=3)
var_jctr_bnb_zambia_2017_scaled_kmeans
```

[25]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

[26]:
```
# 6. Fit the input data on the KMeans object
var_jctr_bnb_zambia_2017_scaled_kmeans.fit(var_jctr_bnb_zambia_2017["Amount_Scaled"].
 ↪values.reshape(-1, 1))
```

[26]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

[27]:
```
# 7. Predict the clusters
var_jctr_bnb_zambia_2017_scaled_clusters = var_jctr_bnb_zambia_2017_scaled_kmeans.
 ↪predict(var_jctr_bnb_zambia_2017["Amount_Scaled"].values.reshape(-1, 1))
```

[28]:
```
# 8. Inspect clusters
var_jctr_bnb_zambia_2017_scaled_clusters
```

[28]:
```
array([2, 2, 0, 0, 2, 0, 0, 0, 1, 2, 2, 0, 2, 1, 0], dtype=int32)
```

[29]:
```
# 9. Merge predictions with original dataset
var_jctr_bnb_zambia_2017["scaled_clusters"] = var_jctr_bnb_zambia_2017_scaled_clusters
```

```
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:2:
SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
```

[30]:
```
# 10. Inspect merged DataFrame
var_jctr_bnb_zambia_2017
```

[30]:

	Town	Date	Amount	DateX	clusters	Amount_Scaled	scaled_clusters
13	Chinsali	Dec 17	2911.25	2017-12-01	1	0.158357	2
31	Chipata	Dec 17	2526.25	2017-12-01	1	0.000000	2
49	Choma	Dec 17	3476.39	2017-12-01	0	0.390808	0
67	Kabwe	Dec 17	3547.39	2017-12-01	0	0.420011	0
85	Kasama	Dec 17	3006.61	2017-12-01	1	0.197580	2
103	Kitwe	Dec 17	3973.76	2017-12-01	0	0.595384	0
121	Livingstone	Dec 17	3756.51	2017-12-01	0	0.506026	0
139	Luanshya	Dec 17	3568.94	2017-12-01	0	0.428875	0
157	Lusaka	Dec 17	4957.47	2017-12-01	2	1.000000	1
175	Mansa	Dec 17	2751.78	2017-12-01	1	0.092764	2
193	Mongu	Dec 17	3033.33	2017-12-01	1	0.208570	2
211	Monze	Dec 17	3836.30	2017-12-01	0	0.538845	0
229	Mpika	Dec 17	2751.78	2017-12-01	1	0.092764	2
247	Ndola	Dec 17	4883.65	2017-12-01	2	0.969637	1
265	Solwezi	Dec 17	4115.17	2017-12-01	0	0.653548	0

[31]:
```
# 11. Plot clusters
plt.scatter(var_jctr_bnb_zambia_2017["Town"],␣
 ↪var_jctr_bnb_zambia_2017["Amount_Scaled"],␣
 ↪c=var_jctr_bnb_zambia_2017["scaled_clusters"], cmap="rainbow")
plt.xticks(rotation=90)
plt.title("2017 JCTR BnB by Town Cluster Plot")
plt.xlabel("Town")
plt.ylabel("Scaled Amount")
```

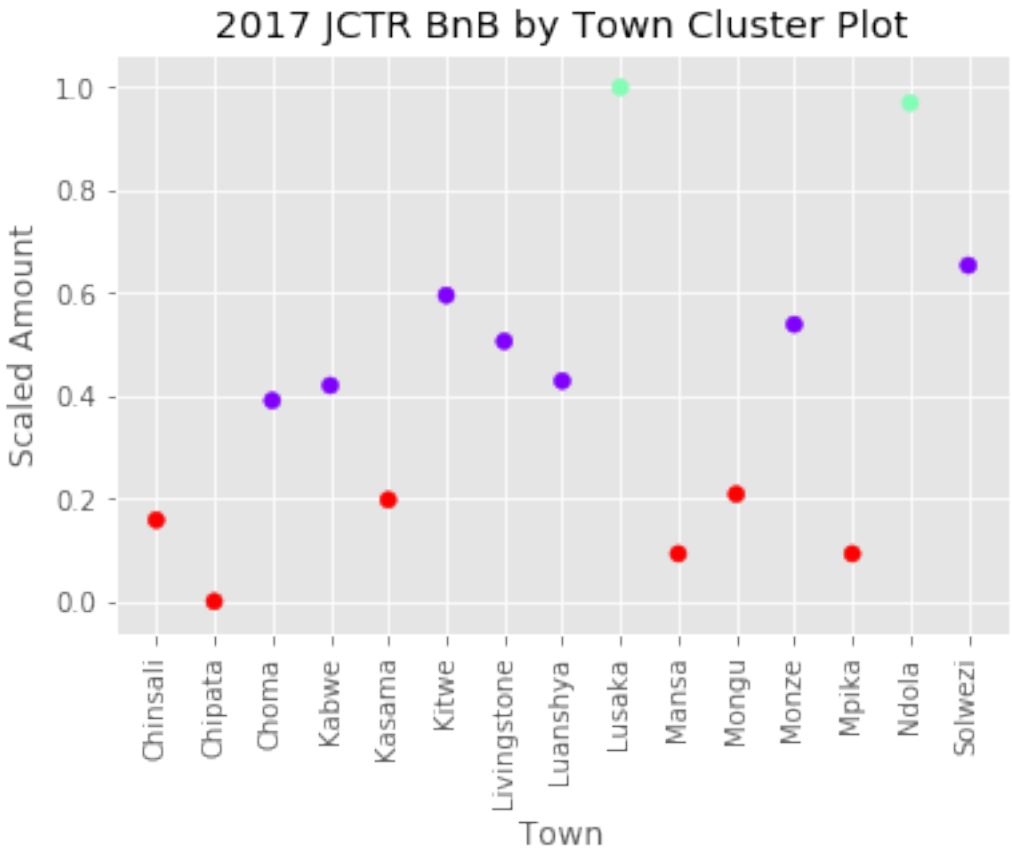
[31]:
```
<matplotlib.collections.PathCollection at 0x7f7adce60d30>
```

[31]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],
 <a list of 15 Text xticklabel objects>)
```

[31]:
```
Text(0.5, 1.0, '2017 JCTR BnB by Town Cluster Plot')
```

[31]:
```
Text(0.5, 0, 'Town')
```

[31]:
```
Text(0, 0.5, 'Scaled Amount')
```



[32]:
```
# KMeans aspects
var_jctr_bnb_zambia_2017_scaled_kmeans.inertia_
var_jctr_bnb_zambia_2017_scaled_kmeans.labels_
```

[32]:
```
0.08896876500570382
```

[32]:
```
array([2, 2, 0, 0, 2, 0, 0, 0, 1, 2, 2, 0, 2, 1, 0], dtype=int32)
```

Example #2: 2017 Town BnBs K Means Optimal Number of Clusters

[33]:
```
# Inspect DataFrame
var_jctr_bnb_zambia_2017.head(2)
```

[33]:

	Town	Date	Amount	DateX	clusters	Amount_Scaled	scaled_clusters
13	Chinsali	Dec 17	2911.25	2017-12-01	1	0.158357	2
31	Chipata	Dec 17	2526.25	2017-12-01	1	0.000000	2

[34]:
```
# We will check against all possible K values
# We will use the scaled amounts
var_data_input = var_jctr_bnb_zambia_2017["Amount_Scaled"].values.reshape(-1, 1)
var_sse = {}
```

```python
for var_k in range(1, 15):
    var_kmeans = KMeans(n_clusters=var_k).fit(var_data_input)
    var_jctr_bnb_zambia_2017["k_clusters"] = var_kmeans.labels_
    #####print(var_jctr_bnb_zambia_2017["k_clusters"])
    var_sse[var_k] = var_kmeans.inertia_ # Inertia: see help
```

```
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
```

```
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  import sys
```

[35]:
```
# Visualise the elbow plot
plt.figure()
plt.plot(list(var_sse.keys()), list(var_sse.values()))
plt.title("2017 JCTR BnB by Town Elbow Plot")
plt.xlabel("Number of Clusters")
plt.ylabel("Sum of Squares Error")
```

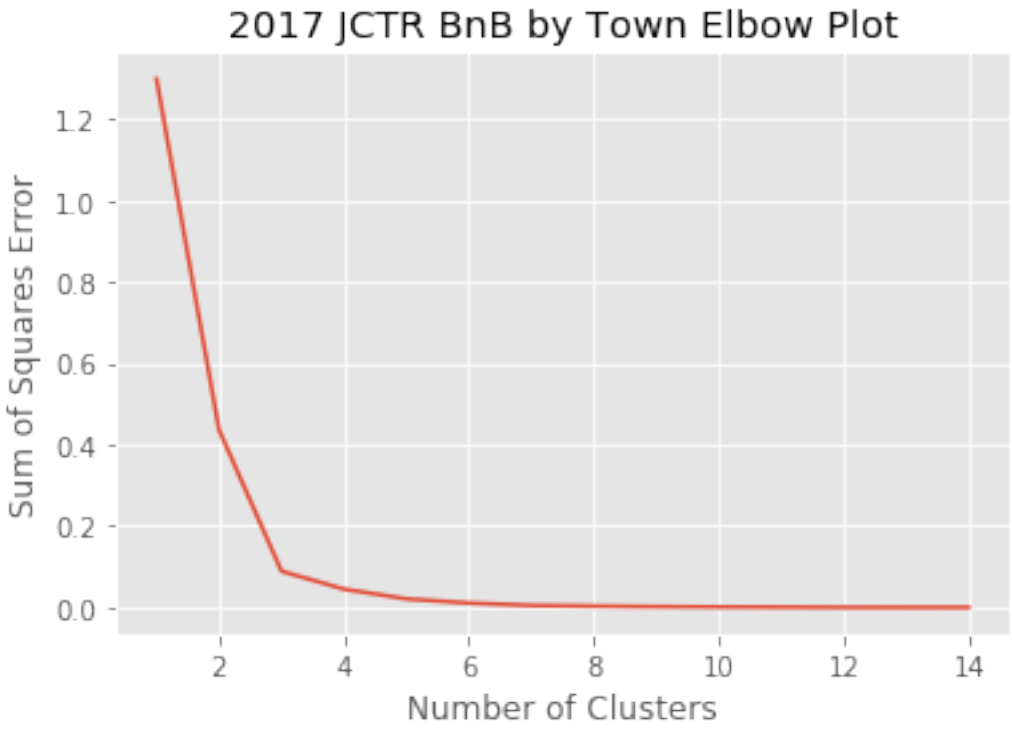
[35]:
```
<Figure size 432x288 with 0 Axes>
```

[35]:
```
[<matplotlib.lines.Line2D at 0x7f7adcdd4e80>]
```

[35]:
```
Text(0.5, 1.0, '2017 JCTR BnB by Town Elbow Plot')
```

[35]:
```
Text(0.5, 0, 'Number of Clusters')
```

[35]:
```
Text(0, 0.5, 'Sum of Squares Error')
```


2017 JCTR BnB by Town Elbow Plot
Sum of Squares Error
1.2
1.0
0.8
0.6
0.4
0.2
0.0
2
4
6
8
10
12
14
Number of Clusters