Student Name: Chandra Likhitha Chopparapu

Student Email:Chandra.Likhitha.Chopparapu-1@ou.edu (mailto:Chandra.Likhitha.Chopparapu-1@ou.edu)

Project 3: The Smart City Slicker

Imagine you are a stakeholder in a rising Smart City and want to know more about themes and concepts about existing smart cities. You also want to know where does your smart city place among others. In this project, you will perform exploratory data analysis, often shortened to EDA, to examine a data from the 2015 Smart City Challenge (https://www.transportation.gov/smartcity) to find facts about the data and communicating those facts through text analysis and visualizations.

In order to explore the data and visualize it, some modifications might need to be made to the data along the way. This is often referred to as data preprocessing or cleaning. Though data preprocessing is technically different from EDA, EDA often exposes problems with the data that need to be fixed in order to continue exploring. Because of this tight coupling, you have to clean the data as necessary to help understand the data.

In this project, you will apply your knowledge about data cleaning, machine learning, visualizations, and databases to explore smart city applications.

Part 1 of the notebook will explore and clean the data.
Part 2 will take the results of the preprocessed data to create models and visualizations.

Empty cells are code cells. Cells denoted with [Your Answer Here] are markdown cells. Edit and add as many cells as needed.

Output file for this notebook is shown as a table for display purposes. Note: The city name can be Norman, OK or OK Norman.

city	raw text	clean text	clusterid	topicids	summary	keywords
Norman, OK	Test, test , and testing.	test test test	0	T1, T2	test	test

Introduction

```
The Dataset: 2015 Smart City Challenge Applicants (non-finalist).
In this project you will use the applicant's PDFs as a dataset.
The dataset is from the U.S Department of Transportation Smart City Challenge.

On the website page for the data, you can find some basic information about the
challenge. This is an interesting dataset. Think of the questions that you might be
able to answer! A few could be:

1. Can I identify frequently occurring words that could be removed during data
preprocessing?
2. Where are the applicants from?
3. Are there multiple entries for the same city in different applicantions?
4. What are the major themes and concepts from the smart city applicants?
```

```
Let's load the data!
```

Loading and Handling files

Load data from `smartcity/` .

To extract the data from the pdf files, use the pypdf.pdf.PdfFileReader (https://pypdf.readthedocs.io/en/stable/index.html) class. It will allow you to extract pages and pdf files and add them to a data structure (dataframe, list, dictionary, etc). To install the module, use the command `pipenv install pypdf` . You only need to handle PDF files, handling docx is not necessary.

In [88]:

```
pip install PyPDF2
```

```
Defaulting to user installation because normal site-packages is not write
able
Requirement already satisfied: PyPDF2 in c:\users\likitha\appdata\roaming
\python\python310\site-packages (3.0.1)
Note: you may need to restart the kernel to use updated packages.
```

In [89]:

```
import os
from PyPDF2 import PdfReader
folder_path = "C:\\Users\\Likitha\\Project3-TextAnalytics\\SmartCity"
files = os.listdir(folder_path)
all_text = []
for file_name in files:
    if file_name.endswith(".pdf"):
        file_path = os.path.join(folder_path, file_name)
        with open(file_path, 'rb') as pdf_file:
            pdf_reader = PdfReader(pdf_file)
            for page in range(len(pdf_reader.pages)):
                pdf_page = pdf_reader.pages[page]
                text = pdf_page.extract_text()
                all_text.append(text)
                print(all_text)
```

```
.....................................................................
.. 11 \n5.11  Vision Element #11: Low-Cost, Efficient, Secure, and Res
ilient Information and Communications \nTechnology
.....................................................................
.....................................................................
.. 11 \n5.12  Vision Element #12: Smart Land Use
.....................................................................
.....................  12 \n6 RISKS
.....................................................................
.....................................................................
........ 12 \n7 TEAM
.....................................................................
.....................................................................
....... 12 \n8 EXISTING INFRASTRUCTURE
.....................................................................
..................................... 13 \n9 DATA COLLECTION
.....................................................................
.....................................................  15 \n10 EXI
STING POLICIES
.....................................................................
```

Create a data structure to add the city name and raw text. You can choose to split the city name from the file.

In [90]:

```
import pandas as pd
data = []
pdf_files = [f for f in os.listdir(folder_path) if f.endswith('.pdf')]
for pdf_file in pdf_files:
    file_path = os.path.join(folder_path, pdf_file)
# Extract the city name from the file name
    city_name = pdf_file[:-4]  # Remove '.pdf' from the file name
     # Extract the raw text from the PDF file
    with open(file_path, 'rb') as f:
        pdf_reader = PdfReader(f)
        raw_text = ''
        for page in pdf_reader.pages:
            raw_text += page.extract_text()
    data.append([city_name, raw_text])
df = pd.DataFrame(data, columns=['city', 'raw_text'])
print(df)
```

```
unknown widths :
[0, IndirectObject(532, 0, 2062972931712)]
unknown widths :
[0, IndirectObject(535, 0, 2062972931712)]
unknown widths :
[0, IndirectObject(538, 0, 2062972931712)]
unknown widths :
[0, IndirectObject(541, 0, 2062972931712)]
unknown widths :
[0, IndirectObject(544, 0, 2062972931712)]
unknown widths :
[0, IndirectObject(547, 0, 2062972931712)]
unknown widths :
[0, IndirectObject(550, 0, 2062972931712)]
unknown widths :
[0, IndirectObject(553, 0, 2062972931712)]

                city                                           raw_text
0        AK Anchorage  CONTENTS \n1 VISION ..........................
1       AL Birmingham  aBirmingham\nRising\nBirmingham Rising! Meetin...
2       AL Montgomery   \n \n U.S. Department of Transportation - “BE...
3    AZ Scottsdale AZ    \n  \n \n \n \nFederal Agency Name:   U.S. D...
4           AZ Tucson  Tucson Smart City Demonstration Proposal\nPart...
..                ...                                                ...
64        VA Richmond    \n \n \n   \n \n \n  \n      Contact Informa...
65  VA Virginia Beach    \n1.  Project Vision  ........................
66         WA Seattle  Beyond Traffic: USDOT Smart City Challenge\nAp...
67         WA Spokane  USDOT Smart City Challenge -  Spokane  \nPage ...
68         WI Madison  Building a Smart Madison  \nfor Shared Prosper...

[69 rows x 2 columns]
```

In [ ]:

```
```

Cleaning Up PDFs

One of the more frustrating aspects of PDF is loading the data into a readable format. The first order of business will be to preprocess the data. To start, you can use code provided by Text Analytics with Python, Chapter 3 (https://github.com/dipanjanS/text-analytics-with-python/blob/master/New-Second-Edition/Ch03%20-%20Processing%20and%20Understanding%20Text/Ch03a%20-%20Text%20Wrangling.ipynb): contractions.py (https://github.com/dipanjanS/text-analytics-with-python/blob/master/New-Second-Edition/Ch05%20-%20Text%20Classification/contractions.py) (Pages 136-137), and text_normalizer.py (https://github.com/dipanjanS/text-analytics-with-python/blob/master/New-Second-Edition/Ch05%20-%20Text%20Classification/text_normalizer.py) (Pages 155-156). Feel free to download the scripts or add the code directly to the notebook (please note this code is performed on dataframes).

In addition to the data cleaning provided by the textbook, you will need to:

1. Consider removing terms that may effect clustering and topic modeling. Words to consider are cities, states, common words (smart, city, page, etc.). Keep in mind n-gram combinations are important; this can also be revisited later depending on your model's performance.
2. Check the data to remove applicants that text was not processed correctly. Do not remove more than 15 cities from the data.

In [91]:

```
```

In [92]:

```
```

```
  Cell In[92], line 1
    pip install spacy
        ^
SyntaxError: invalid syntax
```

In [ ]:

```
```

In [ ]:

```
```

In [3]:

```
```

In [ ]:

```
```

Clean Up: Discussion

Answer the questions below.

In [ ]:

```
```

Which Smart City applicants did you remove? What issues did you see with the documents?

[Your Answer Here]

Add the cleaned text to the structure you created.

Explain what additional text processing methods you used and why.

[Your Answer Here]

Did you identify any potientally problematic words?

[Your Answer Here]

Experimenting with Clustering Models

Now, you'll start to explore models to find the optimal clustering model. In this section, you'll explore K-means (https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html), Hierarchical (https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html), and DBSCAN (https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN) clustering algorithms. Create these algorithms with k_clusters for K-means and Hierarchical. For each cell in the table provide the Silhouette score (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score), Calinski and Harabasz score (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html#sklearn.metrics.calinski_ha and Davies-Bouldin score (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.davies_bouldin_score.html#sklearn.metrics.davies_bould

In each cell, create an array to store the values. For example,

Algorithm	k = 9	k = 18	k = 36	Optimal k
K-means	[S,CH,DB]	[S,CH,DB]	[S,CH,DB]	[S,CH,DB]
Hierarchical	[S,CH,DB]	[S,CH,DB]	[S,CH,DB]	[S,CH,DB]
DBSCAN	X	X	X	[S,CH,DB]

Optimality

You will need to find the optimal k for K-means and Hierarchical algorithms. Find the optimality for k in the range 2 to 50. Provide the code used to generate the optimal k and provide justification for your approach.

Algorithm	k = 9	k = 18	k = 36	Optimal k
K-means	--	--	--	--
Hierarchical	--	--	--	--
DBSCAN	X	X	X	--

In [ ]:

In [ ]:

How did you approach finding the optimal k?

[Your answer here]

What algorithm do you believe is the best? Why?

[Your Answer]

Add Cluster ID to output file

In your data structure, add the cluster id for each smart city respectively. Show the to append the clusterid code below.

In [ ]:

Save Model

After finding the best model, it is desirable to have a way to persist the model for future use without having to retrain. Save the model using model persistance (https://scikit-learn.org/stable/model_persistence.html). This model should be saved in the same directory as this notebook and should be loaded as the model for

your `project3.py` .

Save the model as `model.pkl` . You do not have to use pickle, but be sure to save the persistance using one of the methods listed in the link.

In [ ]:

Derving Themes and Concepts

Perform Topic Modeling on the cleaned data. Provide the top five words for `TOPIC_NUM = Best_k` as defined in the section above. Feel free to reference Chapter 6 (https://github.com/dipanjanS/text-analytics-with-python/tree/master/New-Second-Edition/Ch06%20-%20Text%20Summarization%20and%20Topic%20Models) for more information on Topic Modeling and Summarization.

In [ ]:

In [ ]:

In [ ]:

Extract themes

Write a theme for each topic (atleast a sentence each).

[Your Answer]

[Your Answer]

[Your Answer]

Add Topid ID to output file

Add the top two topics for each smart city to the data structure.

In [ ]:

In [ ]:

```
```

Gathering Applicant Summaries and Keywords

For each smart city applicant, gather a summary and keywords that are important to that document. You can use gensim to do this. Here are examples of functions that you could use.

```
from gensim.summarization import summarize

def summary(text, ratio=0.2, word_count=250, split=False):
    return summarize(text, ratio= ratio, word_count=word_count, split=split)

from gensim.summarization import keywords

def keys(text, ratio=0.01):
    return keywords(text, ratio=ratio)
```

In [ ]:

```
```

In [ ]:

```
```

Add Summaries and Keywords

Add summary and keywords to output file.

In [ ]:

```
```

Write output data

The output data should be written as a TSV file. You can use `to_csv` method from Pandas for this if you are using a DataFrame.

```
Syntax: df.to_csv('file.tsv', sep = '')
df.to_csv('smartcity_eda.tsv', sep='\t')
```

In [ ]:

```
```

Moving Forward

Now that you have explored the dataset, take the important features and functions to create your `project3.py` . Please refer to the project spec for more guidance.

Type *Markdown* and LaTeX: α^2