HW1

February 3, 2022

1 Homework 1

1.1 Haoyu Sheng

In this course we will be using Python for writing code to apply machine learning and text analysis methods to economics topics. Python is free, flexible, offers a variety of predefined packages, and is popular. It can handle everything from the statistical analysis of Stata to the matrix algebra and simulation of Matlab.

This assignment is meant to introduce you to how we will be using Python in this course. For this assignment, you should write/type your answers into this worksheet. You may discuss the problem set with your class mates, but every student must do their own work.

It is always important to cite our references that help us in our work. Please list the students you work with here:

1.

2.

3.

1.1.1 I. PRELIMINARIES (20 points total)

Preliminaries are listed in the HW1 Repository README.md. It includes the following:

- Downloading and installing Python/Anaconda
- Installing necessary pacakges for the homework assignment
- Setting up your GitHub account and connecting it to GitHub Classroom for Econ 1680
- How to sumbit your homework assignment and code (including how to turn your .ipynb file into a .pdf to submit on Canvas)

1.1.2 II. NUMERICAL DATA (40 points total)

1. Access Zillow Real Estate Data using the Nasdaq Data Link API. Nasdaq Data Link is a dataset aggregation website that also has other economics datasets. These types of websites can make it easier to get data and to explore what types of datasets are available.

   a. Set up free account with Nasdaq Data Link (https://data.nasdaq.com/). Find your API Key in your Account Settings. You will need this to download the data.

b. Find the "Zillow Real Estate Data" that is Free (https://data.nasdaq.com/databases/ZILLOW/data) This will be the data you will download.

c. Click on the "Usage" tab, then select the "Python" sub-tab for instructions on using the Nasdaq Data Link API.

d. To decide which variables and regions we want to download data for, we will first download information on the indicators and regions. In a python environment, you will run the code above to import packages, setup your API connection, and download the indicator and region dataframes:

```
[ ]: import nasdaqdatalink
     import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     %matplotlib inline

     # Change the API key to yours
     nasdaqdatalink.ApiConfig.api_key = 'YOUR_API_KEY'
     df_zillow_indicators = nasdaqdatalink.get_table('ZILLOW/INDICATORS',␣
      ↪paginate=True)
     df_zillow_regions = nasdaqdatalink.get_table('ZILLOW/REGIONS', paginate=True)
```

```
[2]: print('Hello World')
```

```
Hello World
```

i. What does ZVHI in the indicator descriptions stand for? (1 point)

ii. What is the indicator, description, and category of row 38 in df_zillow_indicators? Hint: use `.iloc[]` (2 points)

iii. In df_zillow_regions, how many regions are there when you search for "Providence; RI"? What is the region_id number for Providence, RI? Hint: use `.str.contains('Providence; RI')` (4 points)

iv. Download a dataframe the city of Providence, RI on ZHVI Single-Family Home values with the correct indicator and region IDs entered using the following line:

```
df_zillow_sfh = nasdaqdatalink.get_table('ZILLOW/DATA', indicator_id=' ' , region_id=' ',pagin
```

```
[1]: # Write the code you need (if any) to answer the questions above:
```

Answers:

i.
ii.
iii.
iv.

2. Descriptive statistics

a. What is the data frequency in df_zillow_sfh? (1 points)

b. What is the median dollar value of a home in df_zillow_sfh? (4 points)

c. What is the median dollar value of a home in df_zillow_sfh for the year of 2020? Hint: use `[df_zillow_sfh['date'].dt.year==2020]` (4 points)

```
[ ]: # Write the code you need (if any) to answer the questions above:
```

Answer:

a.
b.
c.

3. Visualize the Data

a. Plot a time series graph for values df_zillow_sfh. Be sure to title your graph and label your axes. (7 points)

b. Plot time series graph for yearly median values df_zillow_sfh. Be sure to title your graph and label your axes. Hint: you will can create a new dataframe by creating a 'year' column using .dt.year and then use `.groupby(by=['year']).median()` to make a yearly dataframe. (10 points)

c. What looks different in these graphs? Why? (3 points)

d. Describe the patterns in the graph. What does it say about the housing market in Providence, RI over time? In recent years? What additional data would you need to make claims about what is changing this price? (4 points)

```
[ ]: # Write the code you need (if any) to answer the questions above:
```

Answer:

a.
b.
c.
d.

1.1.3 II. TEXT DATA (40 points total)

4. Download US Economic News Dataset from Kaggle.com: Sign up for a free account with Kaggle.com. This website hosts data science competitions and often has cool datasets available for download. We will be using the US Economic News Dataset at https://www.kaggle.com/heeraldedhia/us-economic-news-articles. Download the CSV file from the website by clicking "Download."

5. Load a subset of the data into Jupyter/Spyder/Python: Sometimes you may be working with a large dataset and it is therefore important to understand how to load a subset of the data at a time. The US Economic News dataset has 8,000 observations.

a. Run the code below and explain in words each of the lines of code with comments (use # to comment): (5 points)

b. What code would you write to keep only the ‘date’, ‘headline’, and ‘text’ columns in the dataframe? Run that code. (2 points)

[ ]:
```
import os
import csv

#
folder_path = 'TYPE FILE PATH TO WHERE YOU HAVE THE DATA'

#
fileReader = open(os.path.join(folder_path, "US-Economic-News.csv"), "r",␣
 ↪encoding="unicode_escape")
csvReader = csv.reader(fileReader)

#
fileWriter = open(os.path.join(folder_path, "Subset_US_Economic_News.csv"),␣
 ↪"w", encoding="unicode_escape", newline='')
csvWriter = csv.writer(fileWriter)

#
acHeader = next(csvReader)
csvWriter.writerow(acHeader)

#
for index, acRow in enumerate(csvReader):
    if index < 800:
        csvWriter.writerow(acRow)

#
fileReader.close()
fileWriter.close()

#
df_news = pd.read_csv(os.path.join(folder_path,"Subset_US_Economic_News.csv"),␣
 ↪encoding='unicode_escape')
df_news['date'] = pd.to_datetime(df_news['date'])
```

[ ]:
```
# Write the code you need (if any) for b.:
```

6. This dataframe is full of text data about US Economic News. When we try to extract information from text, formatting of words and string in code is very important.

   a. Count the number of headlines that have ‘US’ in them. Hint: loop over `df_news[‘headlines’]`. (3 points)

   b. Count the number of headlines that have ‘us’ in them. (3 points)

   c. Why are these counts different? Hint: tell python to check if ‘us’ is in the string ‘trust’. Then tell python to check if ‘ us ‘ is in the string ‘trust’. (3 points)

[ ]:

```
# Write the code you need (if any) to answer the questions above:
```

Answer:

a.
b.
c.

7. In text analysis, we will need to perform a few tasks to clean the data to prepare it for consistent analysis. Run the code and explain what each line does as comments (use # to comment): (5 points)

[ ]:

```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

#
stops = set(stopwords.words('english'))

#
table_punctuation = str.maketrans('', '', '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~')

#
token_list = []
for i, row in enumerate(df_news['text']):
    text = row.translate(table_punctuation)
    tokens = [word.lower() for word in nltk.tokenize.word_tokenize(text) if␣
 ↪word.lower() not in stops]
    token_list.append(tokens)

#
df_news['tokens'] = token_list

#
monetary_policy_wordlist = ['monetary', 'fed ', 'federal reserve', 'Federal␣
 ↪Reserve', 'Monetary']

#
tally = 0
monetary_text = []
for row in df_news['text']:
    mon = 0
    if any(keyword in row for keyword in monetary_policy_wordlist):
        tally += 1
        mon = 1
    monetary_text.append(mon)
print(tally)
df_news['monetary_flag'] = monetary_text
```

```
#
df_monetarynews = df_news[df_news['monetary_flag']==1]
df_nonmonetarynews = df_news[df_news['monetary_flag']!=1]
```

8. Compare and contrast the news articles about monetary policy in the US and those about non-monetary-policy economics in the US.

   a. This code calculates the top 30 most common words in df_news.

```
from collections import Counter
top_N = 30
words = [word for tokenlist in df_news['tokens'].tolist() for word in tokenlist]
topwords = pd.DataFrame(Counter(words).most_common(top_N),
                        columns=['Word', 'Count']).set_index('Word')
print(topwords)
```

Adapt it for the following subquestions:

   i. What are the 15 most common words from df_monetarynews? (3 points)

   ii. What are the 15 most common words from df_nonmonetarynews? (3 points)

   iii. What differences do you notice? (1 points)

```
[ ]: # Write the code you need (if any) to answer the questions above:
```

Answer:

   i.
   ii.
   iii.

   b. Building on the code from part a, you will visualize the word use in the different types of articles using a word cloud. Below is the code for making the word cloud for the df_news dataframe. You must adapt it to the other dataframes:

```
from wordcloud import WordCloud
allwords = ' '.join(words)
word_cloud = WordCloud(collocations=False, background_color='white').generate(allwords)
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.title('Word Cloud for US Economics Articles')
plt.show()
```

   i. What is the word cloud for df_monetarynews? (3 points)

   ii. What is the word cloud df_nonmonetarynews? (3 points)

   iii. What differences do you notice? Do these differences seem consistent with your list of top 15 most common words? (1 points)

```
[ ]: # Write the code you need (if any) to answer the questions above:
```

Answer:

i.
ii.
iii.

9. Monetary Uncertainty in the News: Loughran and McDonald (2011) have created a commonly used bank of word-sentiment lists. One list is a list of “uncertainty words” You can find this dataset in the Github HW1 Repository. The following is code to make a Monetary Uncertain Score from df_monetarynews and to plot the figure over time. However, there are three things wrong with in the code. Identify the typos, run the correct code, and insert the graph below. HINT: Run the code line by line and manually view the objects that were created and/or the error codes that appear. (5 points)

[ ]:
```
folder_path = 'TYPE FILE PATH TO WHERE YOU HAVE THE DATA'

# Word Lists
uncertainty_wordlist_LM = pd.read_csv(os.path.join(folder_path,"LM_Uncertainty.
 ↪csv"), encoding='utf-8')
uncertainty_wordlist_LM = uncertainty_wordlist_LM['uncertain words'].tolist()

# Text Uncertainty Score for Each Article
uncertainty_score = []
for row in df_monetarynews['tokens']:
    u_tally = 0
    for word in uncertainty_wordlist_LM:
        if word in row:
            u_tally += 1
uncertainty_score.append(u_tally)

df_monetarynews['text_uncertainty_score'] = uncertainty_score

# Plot Yearly Mean Monetary Policy Uncertainty Over Time

#Take mean over years
df_monetarynews['year'] = df_monetarynews['date'].dt.year.astype(str)
df_monetarynews_yearly = df_monetarynews.groupby(by=['year']).average()

df_monetarynews_yearly['year'] = pd.to_datetime(df_monetarynews_yearly.index)

#Plot time Series
plt.
 ↪plot(df_monetarynews_yearly['year'],df_monetarynews['text_uncertainty_score'])
plt.xlabel('Years')
plt.ylabel('Mean Uncertainty Score, Yearly')
plt.title('Uncertainty in US Monetary Policy News Articles')
plt.show()
```