Problem Set - 3

Please read all of the guidelines carefully before submitting the problem set. (Unless specified) each question is ***10 points*** and there are ***100 points*** in total. You can complete this assignment in a team of max. 2 people.

Due date: Friday, March 7, 11:59 PM. Late submissions will be accepted with a penalty! (5% reduction per day – no submissions will be accepted two days after the deadline.)

Guidelines – Before You Start

1) **You should complete the problem set in a team of max. 2 people.**
2) Talking about class content with other teams is fine. Sharing answers and sharing code will be considered as plagiarism.
3) If you decide to work in a team of 2 people (including yourself), please make sure you go through all of the questions in the assignment and all of the answers provided by your teammate. Make sure that you understand the material thoroughly.
4) You will be using the **Python** programming language. You need to write your codes in an empty **.ipynb** file.
5) Make sure that you provide many comments to describe your code and the variables that you created.
6) Please send written answers in **.pdf** format (hand-written or processed using MS Word or LaTex)
7) For some of the coding exercises, you may need to do a little bit of "**Googling**" or review the documentation.
8) If the question is asking you to code 'from scratch', it means that you cannot use any pre-packaged tools (unless otherwise specified).

Deliverables:

Important Note: Please number all of your answers. Please enter your name and (if you are working with a partner) your partner's name, as well.
There are two deliverables (two files) for this homework assignment:

1) The **code** of the problem set in **.ipynb** format (one file)
2) Your homework submission in **.pdf** format (one file). Your homework submission must contain answers to all of the questions in the homework. This means:
    a. Hand-written and scanned short answers
    b. The codes you have written for code-based questions (you can use screenshots from the .ipynb file) should also be attached to your homework submission
3) You are welcome to use MS Word or LaTex if you would like.

Reminder: If you need to include any code in your answer, please insert the code in your short answer file (.pdf file), as well.
Important note about the deliverables: If you need to include any code in your answer, please insert the code in your short answer file (.pdf file), as well.

Questions

For Questions 1-4, please use the fake news dataset uploaded on *BlackBoard* (called 'corona_fake.csv'). You can find the file under *'Data'* tab. **Please include your code also in your .pdf file (in code blocks).**

Data Pre-Processing (20 points)

1) **[10 points]** Using the pandas package for Python, import the **corona_fake.csv** dataset (found under Data section on BlackBoard), and do the following:
   Import the nltk package. Check the documentation: https://www.nltk.org/. And, do the following:
   - **i. [2 points]** Using nltk.word_tokenize(), tokenize the text**.**
   - ii. **[2 points]** Using the POS-tagging feature (nltk.pos_tag), POS-tag the tokenized words.
   - **iii. [2 points]** Using WordNetLemmatizer (from nltk.stem import WordNetLemmatizer) lemmatize the pos-tagged words you obtained above. (*Hint*: If there is no available tag, append the token as is; else, use the tag to lemmatize the token)
   - iv. **[2 points]** Using the list of stop words that can be imported (nltk.corpus import stopwords), remove the stopwords in lemmatized text [Note: the language needs to be set as 'english'.].
   - v. **[2 points]** Finally, also **remove numbers, words that are shorter than 2 characters, punctuation, links and emojis**. Finally, convert the obtained list of tokenized+tagged+lemmatized+cleaned list of words back into a joined string (joined by space ' ') and add the result as ***text_clean*** column to your dataset.

2) **[10 points]** Let's vectorize the data we produced above by using two approaches: Bag of Words (BOW) and TF-IDF; and, at the end, we will make a prediction:
   a. **[2.5 points]** Read the following page: https://en.wikipedia.org/wiki/N-gram. Explain what an 'n-gram' is and why it is helpful in max. 200 words.
   b. **[2.5 points]** Import CountVectorizer and TfidfVectorizer:
   from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
   c. **[2.5 points]** Using CountVectorizer, create three vectorized representations of ***text_clean*** [**set lowercase=True**]:
      i. One vectorized representation where ngram_range = (1,1)
      ii. One vectorized representation where ngram_range = (1,2)
      iii. One vectorized representation where ngram_range = (1,3)
   d. **[2.5 points]** Using TfidfVectorizer, create three vectorized representations of ***text_clean*** [**set lowercase=True**]:
      i. One vectorized representation where ngram_range = (1,1)
      ii. One vectorized representation where ngram_range = (1,2)
      iii. One vectorized representation where ngram_range = (1,3)

Prediction (10 points)

3) **[10 points]** Now, let's use sklearn.linear_model.LogisticRegressionCV to do some predictions. Set cv = 5, random_state = 265, and max_iter = 1000, and n_jobs = -1 (other parameters should be left as default) [Note: training size is 70%, test size is 30%, split by random_state = 265].
    a. **[5 points]** By using the **three (3)** different versions of the CountVectorizer dataset you created above, run logistic regression to predict class labels (***fake***, ***true***). Report **three (3)** accuracy values associated with each of the regressions.
    b. **[5 points]** By using the **three (3)** different versions of the TfidfVectorizer dataset you created above, run logistic regression to predict class labels (***fake***, ***true***). Report **three (3)** accuracy values associated with each of the regressions. **Combine and report all accuracy values in a table (6 values in total).**

Theoretical question (20 points)

4) **[20 points]** Check the optimizer (solver) functions used by sklearn.linear_model.LogisticRegressionCV. For each function, explain in around 100 words what they mean; specifically:
    a. **[4 points]** What does ***newton-cg*** mean?
    b. **[4 points]** What does ***lbfgs*** mean?
    c. **[4 points]** What does ***liblinear*** mean?
    d. **[4 points]** What does ***sag*** mean?
    e. **[4 points]** What does ***saga*** mean?

Note: For this question you might need to do some online research. It is your task to find out how they work. You are also welcome to use formulas / matrices in your description.

For Questions 5-9, please use the 'country_information.xlsx' dataset uploaded on *BlackBoard*. You can find the file under *'Data'* tab.

5) **[10 points]** Download the dataset called *'country_information.xlsx'* that can be found under the 'Data' tab on *BlackBoard*. Do the following:
    a. **[5 points]** Provide a summary of what the dataset is about (around 100 words) by checking the variable names (you may need to do some Googling here).
    b. **[5 points]** Excluding the '**country**' column, apply 0-1 normalization on the numeric columns. Save the resulting dataset as: *'country_information_normalized.xlsx'* [Note: Do not forget to add the '***country***' column to the normalized dataset. For normalization, you can use a package.]

6) **[10 points]** Code the ***kmeans++*** algorithm from scratch (you can use packages such as numpy and pandas to store the data). For more information about the individual steps of the algorithm, please check here:

https://en.wikipedia.org/wiki/K-means%2B%2B.

As input, your algorithm should take a numpy matrix or a pandas dataframe and a ***k*** value that denotes the expected number of clusters. The output needs to be the labels associated with feature vectors coming from your dataset.

Note: You are welcome to use pre-packaged algorithms to calculate distances and means. If you need to pick a point randomly, please do the following:

i. Import the random package of Python.
ii. Set seed to 265 by running the following line: random.seed(265) **[This should be done at the very beginning of your code file, after importing the packages.]**
iii. Run the following line: randrange(0,len(name_of_your_dataset),1). Use the resulting the number as the index number for the data point that should be randomly picked in different stages of the kmeans++ algorithm.

For the remainder of the analysis, use the *'country_information_normalized.xlsx'* dataset you created in **Q5**.

7) **[10 points]** Now, we will test the code we have written in **Q6** and apply dimension reduction: Specifically, do the following:

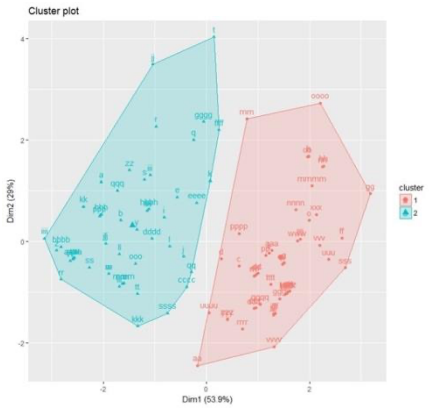
   a. [**5 points**]. Set the random seed to *265* again (to (re-)guarantee the same initialization). Set ***k = 6***. Run your ***kmeans++*** code on the *'country_information_normalized.xlsx'* dataset by excluding the '***country***' column.
   Record the labels. Attach the labels as a new column to your dataset by naming your new variable as ***kmeans_label***.

   b. [**5 points**] Excluding the '***country***' and '***kmeans_label***' columns, run dimension reduction (specifically PCA) on your dataset by using sklearn's PCA function: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html.
   [Note: set n_components = 2 and random_state = 265. Other parameters should be left as 'default'.]. Add the new variables in your dataset as pca_dim_1 and pca_dim_2.

For the next question, use the attached *'visualization_code.py'* file.

8) [**10 points**] Now, let's visualize the results, use the clustering labels to color our data points, and present them in convex hulls. Run the code provided to you in the '***visualization_code.py'*** file. Change the name of the dataset where it says […]. Add the visual to your **.pdf** submission.

   Note: For this exercise, you will need to find and explore the required packages that will need to be imported. The resulting plot should look (somewhat) similar to what is below (but, you will have ***k = 6***).



9) [**10 points**] Interpret the results (in around 300 words) by answering the following:
   a. [**2.5 points**] Which countries seem to be similar? Why do you think these countries are clustered together?
   b. [**2.5 points**] If you run the ***kmeans++*** algorithm more than once, do you think the results will change?
   c. [**2.5 points**] (Subjectively speaking) Do you think this is an accurate clustering of the countries? Would the results change greatly if we had different social/economic variables?
   d. [**2.5 points**] Do you think PCA may have affected the results at all? In other words, if we had a different number of principle components, would our visual interpretation be different?

Grading

Your answers will be evaluated based on the following criteria:

- **Completeness:** Your answers will be checked for completeness. Specifically, for a question that requires several steps of thinking / writing / coding, we expect you to complete the full range of steps to answer the question. The range of steps that needs to be completed will be determined by the course material and the specific nature of the question.
- **Correctness:** Your answers will be checked for correctness. For your answer to be correct, you need to have the correct answer, correct implementation, and the correct result. 'Correct' means that you follow the steps suggested by the assignment and your instructor and obtain the expected result without making any theoretical / mathematical / coding mistakes. 'Correctness' is not a binary term, there may be varying degrees of correctness; and, your grade will be evaluated based on how different your answer is from the expected result.
- **Format:** An indispensable part of every assignment is the format. To make sure that your assignment can be read and processed easily, we expect you to follow the guidelines set by the instructor. These guidelines may include specific requirements about text-based answers, code files, and datasets.

- **Academic Honesty:** We assignment that your submission fulfills the academic honesty expectations set by the instructor and put forward in the syllabus. Specifically, when expected, you need to produce work within the limits defined in the syllabus – some of the assignments may require you to work individually, and some others in a team. For more information, please read the syllabus.