Delivery 1

Quora Objective:

Make a basic model to solve the Quora challenge https://www.kaggle.com/c/quora-question-pairs The deliverable should contain a simple solution and a 'improved solution'.

- Try a simple solution
    - What problems/limitations do you think the model has?
    - What type of errors do you get ?
    - What type of features can you build to improve the basic naive solution?
- Improve your simple solution:
    - Construct features for each input and use them to compute distances between the inputs.
    - Investigate and code a feature vector or a distance between two strings. Used you implementation to define a feature used to capture the similarity between two documents.
    - Implement from scratch the feature vector or the distance function for two input documents
    - Split implementations between members in the group (do not code the same thing twice).
    - Explain the implemented code in `main.pdf` .

Format and delivery rules

General "how to deliver" rules

- The project can be done in groups up to 5 people.
- The project has to be uploaded to the virtual campus. If it is too heavy to upload it has to be sent by email by a single member of each group (in case it is too heavy send a link to dropbox or similar) to `davidbuchaca@ub.edu` The Date for the project is April 26 pm at 12 pm (delivering on the 27 of april is already too late).
- The project needs to be in a zip file containing all the code to reproduce the results.

- The zip file has to be self contained and with the following form: `name1_name2_name3.zip` Where `name1` , `name2` ,... are the names of the members of each group. Please write your full name in `CamelCase` form.
    - If "Elisenda Grau" and "John Snow" make a team the zip filename has to be `ElisendaGrau_JohnSnow.zip`

What it needs to be delivered

The zip file should contain inside something analogous to

```
ElisendaGrauJohnSnow.zip
|
|___ main.pdf
|
|___ train_models.ipynb
|
|___ reproduce_results.ipynb
|
|___ utils.py
|
|___ utils_ElisendaGrau.ipynb
|
|___ utils_JohnSnow.ipynb
|
|___ requirements.txt
```

Explanation

- The zip file `name1_name2_name3.zip` **DOES NOT** have to include train or test data.
    - The data has to be read from `$HOME/Datasets/QuoraQuestionPairs` , ensure that the code can be runned in other computers as long as the data is in the same path.
- The `name1_name2_name3.zip` file must contain:
    - **`main.pdf`** : A description of your work
    - **`model_arfifacts`** : A folder containing the trained models. This folder should be created by **`train_models.ipynb`** and models should be stored there after running **`train_models.ipynb`** notebook. The code should check if the folder is there and in such a case do not overwrite/store the models.
    - **`utils.py`** : A python module with the functions used in `train_models.ipynb` and `reproduce_results.ipynb` .

    - **`train_models.ipynb`**
        - Notebook with the code needed to train and store models to disc.
        - This Notebook has to be clean (do not define functions here, do them in an external `utils.py` and import them).
        - This notebook has to be reproducible (if you run it twice, the same output has to be displayed and stored to disk).
    - **`reproduce_results.ipynb`**:
        - This notebook needs to load models from disk, run evaluations and make a dataframe with the evaluations of the results.
    - Several notebooks `utils_namek.ipynb` containing an explanation of the functions from **`utils.py`** that person `namek` created.
        - This can be used to show/explain the usage of functions in `util.py`
        - Only person `namek` can write and is the owner/responsible for `utils_namek.ipynb`
- **`requirements.txt`** file: requirements file to ensure that your code is runable in another machine.
    - Note that to build your reproducible project you can do before you start
        - `conda create --name quora_challange_env python=3.9`
        - `conda activate quora_challange_env`
        - Install all your dependencies ensuring that you are in the environment
        - Make all your project code
        - Do a `conda list -e > requirements.txt`
        - Run `conda deactivate` to go outside the `quora_challange_env` environment
        - Then, to ensure that everything works do `conda create --name quora_test_env --file requirements.txt`
        - Run **`reproduce_results.ipynb`** within your `quora_test_env` environment

Notes on **`requirements.txt`**

What Will the teacher do to test your code?

- Create an environment `conda create --name quora_test_env --file requirements.txt`
- Run **`reproduce_results.ipynb`** within `quora_test_env` environment
- Check that the results match what you report on `main.pdf`

Notes on `train_models.ipynb`

This notebook:

- is responsability for all memebrs of a group. All of you should ejecute this and ensure it works as expected.
- has to use the code done by each member in the group to generate features for the challange.

This is a Kaggle challange: There is no validation/test data with labels.

Therefore **you have to create the following split** in order to share the same train validation and test splits across teams:

```
train_df = pd.read_csv(os.path.join(path_folder_quora, "quora_train_da
ta.csv"))

A_df, te_df = sklearn.model_selection.train_test_split(train_df,
                                                       test_size=0.05,
                                                       random_state=123)

tr_df, va_df = sklearn.model_selection.train_test_split(A_df,
                                                        test_size=0.05,
                                                        random_state=123

print('tr_df.shape=',tr_df.shape)
print('va_df.shape=',va_df.shape)
print('te_df.shape=',te_df.shape)
```

```
tr_df.shape= (291897, 6)
va_df.shape= (15363, 6)
te_df.shape= (16172, 6)
```

Notes on `Reproducible_results.ipynb`

If there are random parts in the code, make sure to have seeds to make your results reproducible.

This notebook:

- is responsability for all memebrs of a group. All of you should ejecute this and ensure it works as expected.
- contain Train/validation/test results of roc auc (sklearn.metrics.roc*auc*score) as well as

precision and recall.

Note that the teacher will only load and run this notebook, unless I see something very suspicious that makes me run `train_models.ipynb` as well.

- This notebook does not have to train anything.
- It should be relatively fast to execute (probably less than 10 minutes since there is no training).
- This notebook should only load from disk trained models, make predictions and compute metrics.