Assignment 1: Dataset and Model Analysis

Name: Vincent Opitz ID: 791170

I have a git repository https://github.com/Quacck/natural-language-processing, where the code, data, and model is located in.

1 Data Acquisition

a) https://eightportions.com/datasets/Recipes/

b) Overall, this dataset contains recipes in the form of

```
title: string
ingredients: string[],
picture_link?: some_url,
instructions: string
```

All of these are in english language, using imperial metrics. The Ingredients are in natural language, sometimes they even give suggestions where to buy certain foods from or advertise special equipdment.

The author of that dataset scraped 3 different websites: https://github.com/rtlee9/recipe-box, so theres also some variance between those sets.

I made a small script with data_metrics.py that gave me the following metrics:

```
Number of documents =  125164
Number of tokens =  39230
Number of Sentences =  373941
Number of Sentences =  27412152
```

2 Simple Linguistic Model

I trained a simple word embedding model using the gensim library. I preprocessed my JSON formatted data into a csv file, which I could then easily use. I only trained it on the recipes' instructions column with 100000 samples.

I could then save and load the model, which was nice. Gensim allows similarity queries which I based my answers upon.

a) In general, the similarities were surprisingly good for common words:

```
>>> print(model.most_similar('cook', topn=3))
[('saut', 0.7895097136497498), ('saute', 0.7230783700942993), ('fry',
0.6336513161659241)]
```

```
>>> print(model.most_similar('pudding', topn=3))
[('custard', 0.7119596600532532), ('eggnog', 0.6233899593353271),
('puddings', 0.6173929572105408)]
```

Since I formatted my data into a CSV, I had to transform \n into a custom token "LINEBREAK", which also embedded itself into similar words / meanings:

```
>>> print(model.most_similar('linebreak', topn=3))
[('meanwhile', 0.5346844792366028), ('and', 0.4158819019794464),
('assembly', 0.37333229184150696)]
```

b) While the strings may be similar to the word embedding, I believe there may be a problem, since different ingredients can yield very different foods. For example exchanging rice with quinoa results in a completly different dish. While this is probably fine for detecting food related sentences, this may be a problem in other tasks:

```
>>> print(model.most_similar('rice,, topn=3))
[('quinoa', 0.744295060634613), ('barley', 0.7103518843650818), ('bulgur',
0.7011288404464722)]
```

Also, sometimes there are spelling mistakes or just unfitting words (convection for preheat?, deg?, preaheat?):

```
>>> print(model.most_similar('preheat', topn=3))
[('convection', 0.6067588329315186), ('preaheat', 0.5712954998016357),
('deg', 0.5201553702354431)]
```

Kitchen appliances also seem to have a bad similarities, perhaps those are just too unique to have similarities. Instead the model thinks they are similar to the kind of food you cook in them:

```
>>> print(model.most_similar('oven', topn=3))
[('broiler', 0.4504087269306183), ('muffins', 0.4163304567337036),
('pastries', 0.4012565016746521)]
```

Adjectives also seem to have be similar to the wrong words. Instead of being similar to similar adjectives, they are similar to the foods they usually describe. For example, adjectives similar to roasted could be "scorched", "burned", "charred". For sweet perhaps "sugary" seems similar.

```
>>> print(model.most_similar('sweet', topn=3))
[('mashed', 0.775657594203949), ('russet', 0.6871610283851624),
('fingerling', 0.681159496307373)]
```

```
>>> print(model.most_similar('roasted', topn=3))
[('piquillo', 0.703610360622406), ('roma', 0.6789405941963196),
('malagueta', 0.6563881039619446)]
```

3 Describe Next Steps for Project

The end goal of my project should be to declutter websites-with-recipes into just recipes. I believe for that I need training data that contains normal language thats not describing a recipe (to train some kind of classifier and to make my model more robust against non-cooking words / sentences). Another way would be to try it via document / sentence similarity.

To get the website's text itself, I would a write script that would scrape inputted websites' data, perhaps format it, and then feed it into a model. Im not yet sure how I would input the website into any model, perhaps I would just cut most of the structural elements (or replace them with something known) in the HTML and hope for the best. I could imagine needing more classifiers, perhaps something like "is comment (section)", "is advertisement", and alike could be useful. Then I'd need to create useful output from that model.

Im not really sure how feasable this project is. Looking at the data I already have, I also had the idea to create a model that would create instructions from a given list of ingredients and maybe even give it a nice title. Or maybe a user would input a title and the model would generate the rest. Would that also be possible and how flexible would I be in rescoping the project?