APP MTH 3001 Applied Probability
Group project: I forced a bot to read the lyrics to all of modern popular music, and this is what it came up with
Due: 3pm Friday 21 May

Preamble

This project is about algorithmic text generation, specifically, about using mathematical models to generate new music lyrics in the "voice" of an artist from existing corpora of lyrics. Text generation is an important field in natural language processing (NLP), being fundamental to fields like topic modelling, text summarisation, or image captioning. It has filtered all the way through into popular culture, recently through the "I forced a bot" meme, and as Dr Lewis Mitchell was preparing the original version of this project in February 2019, just days before it was to be released, a new "fake text generator" developed by the non-profit research company OpenAI was deemed too dangerous to release by its own creators!

NLP is an active research field within our School.

Lewis took no such precautions in his project; and, it appears, neither did other academics, leading OpenAI to reassess their position.

Markov chains are the foundation of most modern text generation techniques, and have been used in fields like authorship attribution. The simplest models consider "bigrams", where the probability of generating the next word is conditionally dependent on the word immediately before it. More sophisticated models use "trigrams", where the next word depends on the previous *two* words, or higher. There is an excellent MATLAB introduction using Shakespeare, which we will take as the starting point for this project. You should work through this tutorial before commencing the project, and you can build on the functions from it.

The project

The aim of the project is to generate new song lyrics in the style of an artist (or artists) of your choosing, progressively incorporating

1. language structure,
2. rhyme, and
3. meter (number of syllables).

You're welcome to find any text corpus of song lyrics that you like for the project, however I strongly recommend starting with the Open Lyrics Database for its size and relative "cleanliness". Choose at least one artist with a large collection of lyrics. You'll need to experiment, but I expect you'll need at least 1000 lines of lyrics for this project.

You should answer the following questions in the form of a project report that is to be well-written, word processed and spell checked with any supporting material being appropriately presented. Treat this project as if I am your employer and I have asked your group to answer these questions and write me an appropriate report that I can use in dealing with a musician client. My meeting with the client has been scheduled for 9am on Monday 24 May. Hence I need the report by 3pm on Friday 21 May. As I will have very little time to check your results, I need your report to be written in such a way as to demonstrate why your results are correct.

If you want to use LaTeX, I'll make a report template available, and can recommend Overleaf for collaboration.

Part 1

First, we'll generate new lyrics, with no constraints around rhyme or meter. Assume first that each new word is chosen randomly, depending only on the current word, and independently of all previous words.

a) Define your own state space $\mathcal{S}$.

b) Explain why this stochastic process is Markovian.

c) Define clearly how your process will generate a line of lyrics and then stop. This may be related to the state space, or to some summary statistic of your corpus of lyrics, or something else again.

d) Construct a $\mathbb{P}$-matrix representation for the evolution of your lyrics corpus, including all states $i \in \mathcal{S}$.

e) Write down the equations that determine the expected length of a line of lyrics, given that you start a line with word $i \in \mathcal{S}$. Solve these equations for all $i \in \mathcal{S}$. How do you know whether your values are correct?

f) Create a new song lyric for your artist, by simulating a sequence(s) from your stochastic process. Comment on this new composition!

g) Repeat the above (a) – f)) using a "trigram" approach, where each new word is chosen randomly, but depending on the pair of previous *two* words. Generate a new song lyric and compare with the "bigram" approach.

Part 2

Next we'll consider the rhyming structure inherent in modern instrumental pop music, namely, that the last syllable of a line is typically manipulated in order to rhyme with the last syllable of the preceding line. To do this, we'll use a dictionary of rhyming words accessible online through the Datamuse Application Programming Interface (API). An API is a way to programmatically send queries to an online database and get responses in a structured format. The Datamuse API is relatively friendly in that you can make up to 100,000 queries

API's are essential to platforms like Twitter and Spotify, and ubiquitous in modern data science.
Keep track of how many queries/day you make during your project!

per day without needing to apply for an *API key*, queries are simple to form, and the documentation is fairly clear.

a) Write a function to take a word and return a list of rhyming words from the Datamuse API. In MATLAB this is best done using the `webread` function.

The API returns results in JSON format, which MATLAB interprets as a structure.

b) Modify your best text generation code to take the last word from a line and produce a new line whose last word rhymes with that of the previous line. There are many ways to do this, here are two possibilities:

- By modifying $\mathbb{P}$ using the relevant result from Assignments to produce an endpoint-conditioned Markov chain for your desired word;
- By considering the Markov chain built by reading your lyrics in reverse, such that you can begin a new line with a given word.

For those who can't wait, we'll prove a result from Waugh's paper on conditioned Markov processes.

You might need to experiment with these and other methods a bit to find the most effective approach. Make sure you make it clear in your report which approach you've taken for your model, and why.

c) Comment on the notion of *time-reversibility* in Markov chains with respect to your approach to this part in your report.

d) Let's consider the case where a song lyric consists of pairs of rhyming lines. Simulate another new song lyric for your artist, and comment on this new composition.

Part 3

The final extension is to add in the constraint that song lyrics typically comprise lines containing roughly the same number of syllables.

a) Write a function which takes a word and returns an estimate of the number of syllables in that word. You might again find the Datamuse API useful here, or a quick Google will show there are many algorithms for approximating this. You don't need to be perfectly accurate here, but make sure you describe your approach clearly.

b) Decide on a way to choose the number of syllables for a line of song lyric for your artist. This could be as simple as choosing a "sensible" value, or by using your syllables function to analyse statistics for your artist's lyrics corpus, or through a scan of pop music theory, or just for fun. Make sure to justify your choice in the report.

e.g., *iambic pentameter* seems appropriate.

c) Use your syllable-counting function to simulate a new song lyric which respects both rhyme and syllable count, and comment on the result.

You should anticipate needing to use something like *rejection sampling* at this point.

Final notes/thoughts

- While this document is set out like a standard (long) assignment, you'll find as you get into it that the instructions are not totally prescriptive about everything you need to do, and that you'll have to make various *choices* and assumptions along the way. You should make sure that your final report clearly records each of these choices (along with some explanation for why you made the choices you did). Think broadly about *reproducible research*: I should be able to reproduce essentially what you did, by reading your report.

- Along these lines, github might be a good service to look into for collaboration amongst your group, and for collating/referencing your final code for the report.

- Keep in mind also that I'm going to be pitching your work to a client, so you should make sure your final "product" is as good as you can make it! I'll leave the interpretation of this up to you.

- Another point about the report: keep in mind that I expect you to write an "Executive Summary", which is different to an "Abstract". Your summary should give the reader sufficient detail such that they can 'fake' their way through the meeting with the client, so keep this in mind as you write it. Things like dot points, bold type, etc., which help make the summary easily digestible, are very much allowed.

- Week 10 will come around sooner than you think, so get started early. I will prompt you throughout the semester by asking you questions related to this project on your assignments, which will give you an idea of where you should be up to.

- It's entirely possible that things will not go according to plan or end up working very well, particularly as I've given you the freedom to choose your own "training" corpora for your Markov chains. If it doesn't work out, that's ok! So long as you do sufficient exploration, and write up your results clearly, your marks won't be impacted if you don't end up with beautiful poetry.

- And finally, if your project goes well, I would genuinely recommend you to enter into something like Poetix Competition for computer-generated poetry writing! Have lots of fun ☺