Project Progress Report
David Orona, Nikil Nair

1. Which tasks have been completed?
    a. When we first started work on this proposal our idea was to begin by ensuring that the metapy package was up to date in terms of python versioning and OS cross-functionality. When we embarked on the latter we noticed a number of inherent issues with metapy such as but not limited to the installer not properly recognizing python versions before 2.7 and after 3.6. Much of this had to do with prior configurations and through working on the latter we recognized that completing this initial goal is riddled with some pitfalls that we hadn't necessarily accounted for. The biggest issue we faced in this regard would be summed up as a result of a lack of proper documentation and syntactical formatting across functionality. After some tedious re-installation and bash scripting we were able to get a purview of certain python versioning up and running - outside of the latter specified - however, this was still incompatible and is something that has to do with inherent project misconfigurations that weren't properly accounted for in the respective filing. We also conducted extensive meetings with TA's to describe the issues we were encountering and they recommended that we shift our focus from updating metapy versioning to implementing and integrating functionality via existing libraries such as Gensim and NLTK.
2. Which tasks are pending?
    a. As specified in the latter, and due to the numerous metapy configuration issues that we encountered we were recommended to shift our focus towards implementing Gensim and NLTK functionality within metapy. Over the coming weeks, our goal in this regard is to prioritize adding functions to metapy - from the latter - that are able to improve the package while also - if possible - improving the experience that students have while doing the MPs. Since metapy is a fairly old package, and much of its functionality is outdated or has been improved in more recent packages, we want to ensure that this package gets an update that will allow it to utilize pieces of core functionality whilst still remaining a core part of the class structure. In terms of specific classifications we aim to add anywhere from 5 - 10 new functions within metapy that have supported integration and cross-functional capabilities that will be enabled by more recent packages - NLTK/Gensim/etc. - whilst not holistically altering the inner-workings of the package.

3. Are you facing any challenges?

   One of the main hindrances faced within our project has been upgrading metapy to function on a semi-recent Python platform. Up until this point, most of the assignments and exercises within this class have only operated on Python 2.7, with rare exceptions allowing for functioning on 3.6. While the end result is still the same, the performance of the code will suffer in the long run as Python versions continue to increase. To cope for this, we:

   a. Attempted to rectify all errors in a sequential manner on associated failures when running the latest Python (3.11). The most persistent error occuring was "*ERROR: Could not build wheels for metapy, which is required to install pyproject.toml-based projects' '*. Although we rectified a few of those errors via investigations into both the "wheel" and "pyproject.toml", the message still persisted.
   b. To combat functionality from an embedded perspective, we downloaded the metapy library and attempted to align modules from within the package rather than dealing with errors on the outside. This too proved troublesome, as half of the packages within were either outdated or altogether missing. Though this is expected due to the legacy nature of the code, finding and amending the missing material would prove to create extraneous concerns, well beyond the limits of each individual's 20 hour designation.