# Introduction

MIDCON Data Services, is an independent records and information management company headquartered in Oklahoma City. They have hired me to find a creative solution on how to manage and govern electronic records. Companies today are creating petabytes of new data, mostly electronic, each year. This has become a cost and logistics problem for many corporations because most do not have an enforceable document management policy for electronic data. This is not because of negligence, but rather ineffective tools on how to govern and regulate the knowledge that is shared electronically today.  
  
One key problem with electronic data is that it can be stored anywhere (personal drive, public drive, USB, email server, etc.) and “titled” in any random way. One solution is to use a common repository like SharePoint, and then enforce a common naming procedure. This solution will work fine for company procedures and official policies, but it is unrealistic to think that solution is scalable.

Approach

A new way to solve this problem is to create a document classification engine that will assign “categories” or “tags” to a document regardless of document location or filename. A tool like this would give flexibility to the user, but also the governing power to apply best practices from physical records management (picture library systems). If this tool is efficient enough, then it could also be used in Discovery work for the professional services industry.

To start, MIDCON will provide physical documents that belong to one of three categories: ‘Operations’, ‘Accounting’, or ‘Legal’. I will use in-house software and equipment to digitize and OCR each record, and then save the output text file to classified folders. Starting from physical documents will enable better quality control, and also there is more data freely available to the client.

In my code you will see these steps: load data, preprocessing, train/test split, algorithm modeling, and finally result analysis. The preprocessing removes stop words, whitespace, and reduced inflected words to their stem. The corpus is then sent to a pandas data frame and split into equal halves - training and test sets. To concisely test different combinations of Transformers and Classifiers I used Pipeline and GridSearchCV.

# Analysis of Results

After analysis it was found that both the TfidfVectorizer and HashVectorizer performed equally, while the SGDClassifier, which contains SVM, was superior to the simple LinearSVM. All three combinations provided results over 99% recall and accuracy, which is remarkably successful. To figure out why this is I looked at the top features, frequency of the words, and the test/train split.

The list of top features is particularly interesting because it gives insight to which words the model gives higher weight to. In the top 20 list I see several unique operations tokens (pull, scan, item), legal tokens (vault, storage, iiii), and an accounting token (2017). The accounting token was an oversight since many of the financials will be forward looking and the 2017 term will be often repeated in accounting documents. The ‘iiii’ token is meaningful because legal documents are typically heavily outlined by numbering and multilevel lists. In contrast there are not many accounting or operations documents that have this degree of outline. If we were to add a new class for a technical field such as engineering then this token would probably have less weight. The same statement could be said for of the unique operations language. This insight should not be overlooked. I believe the main reason why the model is so successful is because I have chosen 3 distinct classes.

Finally, to demonstrate that the splitting of the data into test and training sets was not a fluke I varied the split of training data by 30%, 40%, 50%, and 60%. I then applied these to our best Model parameters and compared the results. There was not a significant change in the classification report or score.

# Future Work and Recommendations

The supervised approach has proven to be successful in identifying the three trained classes. It is now time to add more classes such as engineering, finance, human relations, and others to the model in order for the tool to be more commercially operational. I also expect the accuracy of the classifier to decrease as complexity is increased. For example, it is more difficult to differentiate accounting to finance than accounting to operations.

On a lesser note, it would also be interesting to try a new method, an unsupervised approach, and let the model extract the topics for itself. The supervised approach better addresses the proposed problem, which is to give a records manager a tool to identify and classify documents into subgroups. However, an unsupervised approach could be used to prime the model, and give the records manager new perspective on the shape of the data.

Finally, after more classes have been added to the model I would also recommend that we start adding electronic sourced data. The current dataset were all physical records, where were digitized and forced to a text file. In addressing the original problem, this model needs to be applied freely to an electronic record repository. Anne Regal has given me permission to use her code which will convert a PDF to text without the need for OCR software. It is on her Github account (mtchem).

# References

* The SciKit-Learn documentation was invaluable in putting this capstone project together. You can read more about the different model features such as TfidfVectorizer. HashVectorizer, SGDClassifier, Pipeline, and GridSearchCV at < <http://scikit-learn.org/stable/documentation.html>>
* I also referenced class materials from the Springboard lectures. Particularly these:
  + Patrick Harrison gave ideas for text analysis and pre-processing < <https://www.youtube.com/watch?v=6zm9NC9uRkk>>
  + Module 8.3 Exploratory Data Analysis
  + Module 11.3 Bayesian Methods and Text Data
* Finally here is a book that was my mentor, Bill Skuba had me read and provided clear concise documentation of different machine learning practices:

Brownlee, J. (2016). *Machine learning mastery with Python: understand your data, create accurate models and work projects end-to-end*. Melbourne, Australia: Jason Brownlee.