Cambridge Data Science Hack

You do not have to work in this Notebook, but a read-through may provide some helpful hints.

Introduction

This challenge focuses on extracting meaning from text. Use cases in a finance company could include:

- Automatic chatbots
- Classifying customer complaints and communications
- Automated underwriting from medical records
- Automated claims handling from accident reports and call transcripts

Here we will use data from government e-petitions to explore two common uses of text in Data Science:

- Predictive Modelling
- Topic Classification

You may wish to work in this Notebook for your solution. But feel free to use any coding language, any approach, any GUI.

A look at the data

We'll start by importing all the packages we might need.

In [19]:

```
import random
import json
import numpy as np
import pandas as pd
from sklearn.cross_validation import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.metrics import confusion_matrix
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.utils import resample
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from collections import Counter
import pyLDAvis
import plotly.express as px
from sklearn.utils import resample
from operator import itemgetter
import seaborn as sns
import gensim
from nltk.stem import PorterStemmer,WordNetLemmatizer
import matplotlib.pyplot as plt
```

Preparing the Data

Loading Data

We have training and holdout data. You'll need to use the training data to create a model and then use your model to label the holdout data.

In [2]:

```
with open('training_data.json', 'rb') as f:
    training_data = json.load(f)

with open('holdout_data.json', 'rb') as f:
    holdout_data = json.load(f)
```

Let's look at the first petition in our training set, and have a look at how many petitions and signatures we're dealing with.

In [3]:

```
print(training_data[0])

print("\nNumber of petitions in training data: {}".format(len(training_data)))
print("\nMean number of signatures: {}".format(int(np.mean([p['numberOfSignatures'] for
p in training_data]))))
print("\nMedian number of signatures: {}".format(int(np.median([p['numberOfSignatures']
for p in training_data]))))
```

```
{'abstract': {'_value': 'MPs should attend all debates, not merely turn up
and vote or strike pairing deals. With other commitments, a five day Commo
ns is not workable for MPs: I suggest three full days (9am to 6pm minimu
m), with one or two days for Committees, leaving at least one day for cons
tituency work.'}, 'created': {'_value': '2016-03-15T15:56:53.752Z', '_data
type': 'dateTime'}, 'label': {'_value': 'Reform the Commons: Three days fu
ll time with compulsory attendance for all MPs.'}, 'numberOfSignatures': 2
7, 'status': 'closed'}

Number of petitions in training data: 12387

Mean number of signatures: 3777

Median number of signatures: 58
```

We can see that the petition text is stored under two keys - the *value* of *label* gives the petition title, and the *value* of *abstract* provides a longer description.

Cleaning the Text

Usually in a Data Science problem we would start with Exploratory Data Analysis (EDA). But this is unstructured text - we can't calculate simple statistics or plot interesting histograms until we have turned this text into numbers. Before thinking about making a model, **structure and clean the text**.

You could consider:

- Tokenizing
- Lemmatizing or Stemming
- Filtering
- Calculating TF-IDF

Packages of use may include:

- NLTK
- Gensim
- Sklearn

Any blog post about any form of text modelling will begin with a section on text preprocessing. Some examples are here:

- https://medium.com/@annabiancajones/sentiment-analysis-of-reviews-text-pre-processing-6359343784fb
- https://www.machinelearningplus.com/nlp/gensim-tutorial/#8howtocreatethetfidfmatrixcorpusingensim
- https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089
- https://towardsdatascience.com/nlp-extracting-the-main-topics-from-your-dataset-using-lda-in-minutes-21486f5aa925

Challenge Part I: Predict Whether a Petition Surpasses 50 Signatures

You need to create a model to **predict whether a petition has surpassed 50 signatures, using only the petition text as input**.

Your first task will probably be feature generation. You may wish to consider:

- Word counts
- Word frequencies (and TF-IDF)
- Word embedding
- Custom rules

This is a supervised learning task using text - multiple blog posts cover walkthroughs for various purposes (e.g. sentiment analysis). Some potential resources are here:

- https://medium.com/@annabiancajones/sentiment-analysis-on-reviews-feature-extraction-and-logistic-regression-43a29635cc81
- https://www.kaggle.com/arunava21/word2vec-and-random-forest-classification

Once you have a model, **predict whether each petition in the holdout set will surpass 50 signatures**. Your predictions will be assessed against the truth using the F1 score. F1 accounts for both Precision (of all the petitions you predict to surpass 50 signatures, how often were you correct) and Recall (of all the petitions which surpass 50 signatures, how many do you correctly identify).

Your submission for Part I should be a CSV list of 3,000 Booleans to represent your predictions for the holdout set - True if you think a petition will surpass 50 signatures, False otherwise.

Challenge Part II: Topic Modelling

In many problems it is useful to cluster texts together which seem to talk about the same topic. You may wish to understand what customers tend to complain about, what news articles tend to be about or what reviews tend to talk about. For this task, **you need to automatically group the petitions into topics.**.

On the hack day you will show us your topic classification and explain how you have decided - quantiatively or qualitatively - on your end result.

You may wish to consider:

- LDA
- Clustering