# 04_fce_clean

March 21, 2017

2/23/17 - smiel

## 1 Cleaning the FCE essay data.

In [1]:
```
%matplotlib inline
# run me when first starting this notebook
import os

import numpy as np
import pandas as pd

path = '/research/ella/rivendell/fce'
```

In [15]:
```
# create the essays data frame
from bs4 import BeautifulSoup
import re


def parse_student(candidate):
    ret = {}
    ret['score'] = candidate.find('score').text
    ret['score_type'] = 'FCE'
    ret['language'] = candidate.find('personnel').find('language').text

    age = candidate.find('personnel').find('age')
    ret['age'] = np.nan if age is None else age.text

    return ret


def parse_answer(answer, essay_id):
    prompt_id = answer.find('question_number').text
    ret = {}
    ret['essay_id'] = essay_id
    ret['prompt_id'] = prompt_id
    ret['text'] = parse_text(answer.find('coded_answer'))
```

```
    return ret


def parse_text(ca):
    chunks = []
    for child in ca.recursiveChildGenerator():
        name = getattr(child, 'name', None)
        if name is None and child.parent.name != 'c':
            chunks.append(child)

    ret = ''.join(chunks)
    return re.sub('[ ]+', ' ', ret).strip()


def parse_answers(exam_id, student_id):
    xml_path = os.path.join(path, 'fce-released-dataset', 'dataset', exam_id, 'doc{}.xm
    with open(xml_path, 'r') as fin:
        soup = BeautifulSoup(fin, 'lxml')

    doc_id = '{}_{}'.format(exam_id, student_id)

    try:

        student = parse_student(soup.find('candidate'))
        student['exam_id'] = exam_id
        student['student_id'] = student_id

        recs = []
        i = 1
        while True:
            answer = soup.find('answer{}'.format(i))
            if answer is None:
                break

            essay_id = '{}_{}'.format(doc_id, i)
            rec = parse_answer(answer, essay_id)
            rec.update(student)
            recs.append(rec)
            i += 1
        return recs
    except:
        print(xml_path)
        raise

recs = []

data_path = os.path.join(path, 'fce-released-dataset', 'dataset')
for exam_id in os.listdir(data_path):
```

```
            exam_path = os.path.join(data_path, exam_id)
            for filename in os.listdir(exam_path):
                student_id = filename[3:-4]
                recs.extend(parse_answers(exam_id, student_id))

        df = pd.DataFrame.from_records(recs)
        print('{} essays'.format(len(df)))
        df.to_csv(os.path.join(path, 'all_essays.csv'), encoding='utf8', index=False)
        df.head()
```

```
2481 essays
```

Out[15]:

| | age | essay_id | exam_id | language | prompt_id | score | score_type | \ |
|---|---|---|---|---|---|---|---|---|
| 0 | 16-20 | 0101_2000_6_837_1 | 0101_2000_6 | Greek | 1 | 22.0 | FCE | |
| 1 | 16-20 | 0101_2000_6_837_2 | 0101_2000_6 | Greek | 3 | 22.0 | FCE | |
| 2 | <16 | 0101_2000_6_1156_1 | 0101_2000_6 | Greek | 1 | 27.0 | FCE | |
| 3 | <16 | 0101_2000_6_1156_2 | 0101_2000_6 | Greek | 3 | 27.0 | FCE | |
| 4 | 16-20 | 0101_2000_6_751_1 | 0101_2000_6 | Greek | 1 | 25.0 | FCE | |

| | student_id | text |
|---|---|---|
| 0 | 837 | Dear Mrs Brown,\nI am writing to give you info... |
| 1 | 837 | It was Friday morning when I saw John and said... |
| 2 | 1156 | Dear Mrs Brown,\nI am one of your husband's st... |
| 3 | 1156 | John said he had some good news to tell me. I ... |
| 4 | 751 | Dear Mrs Brown,\nIt would be a pleasure to us ... |

In [16]:
```
df.groupby('language').size()
```

Out[16]:
```
language
Catalan       128
Chinese       132
Dutch           4
French        291
German        138
Greek         148
Italian       152
Japanese      162
Korean        170
Polish        152
Portuguese    136
Russian       166
Spanish       398
Swedish        30
Thai          126
Turkish       148
dtype: int64
```

In [17]:
```
language_map = {
    'Catalan': 'CAT',
```

```
            'Chinese': 'CHN',
            'Dutch': 'NL',
            'French': 'FRA',
            'German': 'GER',
            'Greek': 'GRC',
            'Italian': 'ITA',
            'Japanese': 'JPN',
            'Korean': 'KOR',
            'Polish': 'POL',
            'Portuguese': 'PRT',
            'Russian': 'RUS',
            'Spanish': 'SPA',
            'Swedish': 'SWE',
            'Thai': 'THA',
            'Turkish': 'TUR'
        }
        df['L1'] = df.language.apply(lambda lang: language_map[lang])
        df.to_csv(os.path.join(path, 'all_essays.csv'), encoding='utf8', index=False)
```

Now let's start building the sentences data frame. For unidecode to work properly, the following should print "True":

```
In [2]: import sys
        print(sys.maxunicode > 0xffff)
```

```
True
```

The essays don't have IDs or ages, but we might know grade level. Let's add an ID and put a placeholder for grade. In addition, we do have two kinds of language scores which we'll compute here. They are also all to the same prompt.

```
In [36]: from utilitybelt.text import get_sentences
         import copy
         from unidecode import unidecode
         import numpy as np

         # load data
         df_in = pd.read_csv(os.path.join(path, 'all_essays.csv'), encoding='utf8')

         # convert text to ascii
         print('Converting to ASCII')
         df_in['ascii_text'] = df_in.text.apply(lambda t: unidecode(t))

         # normalize line endings
         df_in.ascii_text = df_in.ascii_text.str.replace('\r\n', '\n')
         df_in.ascii_text = df_in.ascii_text.str.replace('\r', '\n')

         # use space instead of tab
```

```
df_in.ascii_text = df_in.ascii_text.str.replace('\t', ' ')

# now remove any non-printable ascii char
df_in.ascii_text = df_in.ascii_text.str.replace(r'[^ -~\n]', '')

# # make sure all is printable
# for i, t in enumerate(df_in.ascii_text.values):
#     for ci, c in enumerate(t):
#         if (32 <= ord(c) <= 126) or c in '\n\t':
#             continue
#         else:
#             print u"Unprintable character {} in {} at char {}:\n\n{}\n==============
#                 ord(c), i, ci, t, df_in.iloc[i].clean_text
#             )
#             raise ValueError

# shush the utilitybelt sentence splitter logging
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)

print('Splitting sentences')
# create records for every sentence
records = []
for i, row in df_in.iterrows():
    rec = {
        'dataset': 'FCE', 'prompt_id': row.prompt_id, 'essay_id': row.essay_id, 'L1': r
        'score': row.score, 'score_type': 'FCE',
        'age': np.nan,  # we only have broad ranges and would have to guess at exact ag
    }
    prev_end = 0
    text = row.ascii_text
    si = 0
    for start, end, sentence in zip(*get_sentences(text)):
        srec = {}
        srec.update(rec)
        srec['text'] = sentence
        srec['sentence_id'] = si
        srec['trailing_whitespace'] = text[prev_end:start]
        si += 1
        prev_end = end
        records.append(srec)

    if i % 1000 == 0:
        print('{} of {}'.format(i, len(df_in)))

print('Creating data frame')
df_out = pd.DataFrame.from_records(records)
```

```
        df_out['uid'] = df_out[['dataset', 'essay_id', 'sentence_id']].astype(unicode).apply(la

        print('{} sentences'.format(len(df_out)))
        print('Saving data frame')
        df_out.to_csv(os.path.join(path, 'FCE_sentences.csv'), encoding='utf8', index=False)
```

```
Converting to ASCII
Splitting sentences
0
0
85
Dear Mrs Brown,
I am writing to give you information about Mr Brown's surprise party.
```

```
        ---------------------------------------------------------------------------

        ValueError                                Traceback (most recent call last)

        <ipython-input-36-75b5edab6c8b> in <module>()
         55         print(sentence)
         56         print(text[prev_end:start])
    ---> 57         raise ValueError()
         58         srec = {}
         59         srec.update(rec)


        ValueError:
```

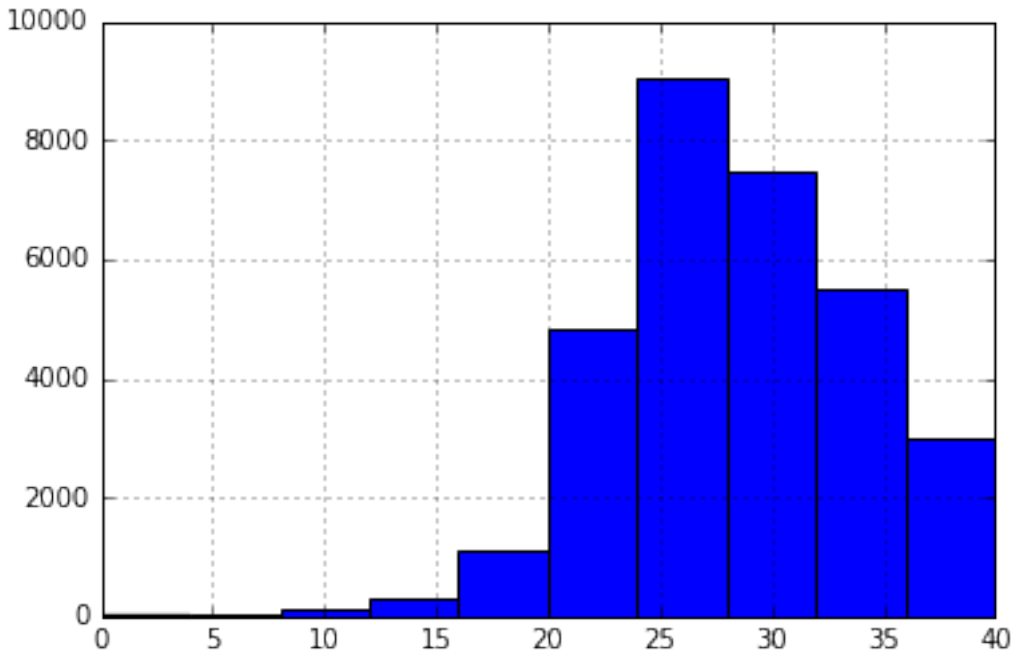
Let's do a little descriptive analysis to make sure we got what we want.

```
In [45]: df = pd.read_csv(os.path.join(path, 'FCE_sentences.csv'), encoding='utf8')
```

```
In [20]: age = df.groupby('age').size()
         print(age)
         print('{} sentences with age data'.format(pd.notnull(df.age).sum()))
```

```
Series([], dtype: int64)
0 sentences with age data
```

```
In [21]: score = df.score.hist()
```




```
In [22]: df.text.apply(len).describe()
```

```
Out[22]: count    31400.000000
         mean        79.075924
         std         49.332025
         min          1.000000
         25%         44.000000
         50%         70.000000
         75%        104.000000
         max        813.000000
         Name: text, dtype: float64
```

```
In [49]: df.trailing_whitespace = df.trailing_whitespace.fillna('')
         essay1_id = df.essay_id.values[0]
         essay1 = df[df.essay_id == essay1_id]
         essay1['text_plus'] = essay1.trailing_whitespace + essay1.text
         text = ''.join(essay1.text_plus.values)
         print(text)
         print(essay1_id)
```

```
Dear Mrs Brown,
I am writing to give you information about Mr Brown's surprise party.
First of all, the reason that we decided to do this party was because Mr Brown helped to the org
Secondly, the party it is on Tuesday 16th pm at the College Canteen. As concerned the food, our
Moreover, at the party they will be come of course all the class, our teachers and the Principal
I hope that to have been of some help you about the party. I look forward to seeing you there.
```

Yours sincerely
0101_2000_6_837_1

```
/home/smiel/.venvs/rivendell/lib/python2.7/site-packages/ipykernel/__main__.py:4: SettingWithCop
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
```