train_and_evaluate_percieving_train2100eval900_and_judging_train21

March 14, 2018

In [1]:
```
import pandas as pd
import os
import requests
import numpy as np
import operator
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
import seaborn as sns
import numpy as np
```

In [2]:
```
!pwd
```

```
/Users/mos/Dropbox/memeticscience/typealyzer-dataset/notebooks/jungian_classification
```

In [3]:
```
df_pickle_path = "../../pickles/dataframe_survey_2018-01-23_enriched.pickle"
```

In [4]:
```
indata = pd.read_pickle(df_pickle_path)
indata[["actual","actual_temp","is_s"]].head(5)
```

Out[4]:

	actual	actual_temp	is_s
1	INFJ	nf	0
2	INFP	nf	0
3	INTP	nt	0
5	ENFJ	nf	0
10	INFP	nf	0

In [5]:
```
indata.domain.value_counts()
```

Out[5]:
```
tumblr       21938
blogspot       513
wordpress      468
Name: domain, dtype: int64
```

In [6]:
```
indata.lang.value_counts().head()
```

```
Out[6]: en    22588
        fr       50
        da       34
        de       25
        no       23
        Name: lang, dtype: int64
```

1 Filter out English texts only

```
In [7]: len(indata)
```

```
Out[7]: 22919
```

```
In [8]: indata = indata[indata.lang == "en"]
```

```
In [9]: len(indata)
```

```
Out[9]: 22588
```

2 Add columns percieving and judging for evaluation of s/n and t/f classifiers

```
In [10]: indata["perc_func"] = indata.actual_temp.str.extract("(\w)\w", expand=False)
         indata["judg_func"] = indata.actual_temp.str.extract("\w(\w)", expand=False)
```

```
In [11]: len(indata[pd.isnull(indata["tokens"])])
```

```
Out[11]: 0
```

```
In [12]: indata[pd.isnull(indata["tokens"])]
```

```
Out[12]: Empty DataFrame
         Columns: [url, typealyzer, actual, e, s, t, sntf_s, sntf_n, sntf_t, sntf_f, date, tex
         Index: []

         [0 rows x 117 columns]
```

3 Inspect original data function distributions

```
In [13]: s_series = indata[indata.perc_func == "s"]["tokens"]
         n_series = indata[indata.perc_func == "n"]["tokens"]
         t_series = indata[indata.judg_func == "t"]["tokens"]
         f_series = indata[indata.judg_func == "f"]["tokens"]

         avg_tkns = {
             "s":s_series.mean(),
             "n":n_series.mean(),
```

```
            "t":t_series.mean(),
            "f":f_series.mean()
        }
        avg_tkns
```

```
Out[13]: {'f': 488.2908894968084,
          'n': 511.4496560721063,
          's': 457.63189127105665,
          't': 511.27211970074814}
```

```
In [14]: indata.perc_func.value_counts()
```

```
Out[14]: n    16864
         s     5224
         Name: perc_func, dtype: int64
```

```
In [15]: indata.judg_func.value_counts()
```

```
Out[15]: f    12063
         t    10025
         Name: judg_func, dtype: int64
```

4 Sample equal size text chunks for training and evaluation data

See: Pandas sample()

4.1 Percieving function

```
In [25]: # We have 5224 cases in the smallest class s
         perc_samples = pd.concat([
                     indata[indata.perc_func == "s"].sample(3000, random_state=123456)[["text"
                     indata[indata.perc_func == "n"].sample(3000, random_state=123456)[["text"
                     ])
```

```
In [26]: len(perc_samples)
```

```
Out[26]: 6000
```

```
In [27]: perc_samples.perc_func.value_counts()
```

```
Out[27]: s    3000
         n    3000
         Name: perc_func, dtype: int64
```

```
In [28]: perc_samples.judg_func.value_counts()
```

```
Out[28]: f    3363
         t    2637
         Name: judg_func, dtype: int64
```

```
In [29]: perc_samples.to_pickle("jung_percieving_functions_samples_blogs_totn6000.pickle")
```

```
In [37]: perc_samples = pd.read_pickle("jung_percieving_functions_samples_blogs_totn6000.pickle
```

4.2 Judging function

```
In [30]: # We have 10025 cases in the smallest class t
         judg_samples = pd.concat([
                     indata[indata.judg_func == "t"].sample(3000, random_state=123456)[["text"
                     indata[indata.judg_func == "f"].sample(3000, random_state=123456)[["text"
                     ])

In [31]: len(judg_samples)

Out[31]: 6000

In [32]: judg_samples.judg_func.value_counts()

Out[32]: t    3000
         f    3000
         Name: judg_func, dtype: int64

In [33]: judg_samples.perc_func.value_counts()

Out[33]: n    4598
         s    1402
         Name: perc_func, dtype: int64

In [34]: judg_samples.to_pickle("jung_judging_functions_samples_blogs_totn6000.pickle")

In [44]: judg_samples = pd.read_pickle("jung_judging_functions_samples_blogs_totn6000.pickle")
```

5 Setup uClassify classifer and prepare training and evaluation datasets

The variable `os.environ["UCLASSIFY_WRITE"]` is created by adding a line to e.g. `~/.profile`:

```
export UCLASSIFY_WRITE = "<your_uclassify_write_key>"
```

6 Split percieving samples into train and eval subsets.

```
In [35]: len(perc_samples)

Out[35]: 6000

In [36]: # prepare column to keep track of what's been used for training
         zeros = np.zeros(len(perc_samples))
         perc_samples["perc_training_set"] = zeros
         perc_samples["perc_training_set"] = perc_samples.perc_training_set.astype("int")
         perc_samples.head(3)
```

Out[36]:

	text	tokens	perc_func	judg_func	actual_temp	perc_training_set
8623	Sonny Jooooooooon INDEX ASK PAST THEME Sonny J...	386	s	f	sf	0
11987	Log in | Tumblr Sign up Terms Privacy Posted b...	52	s	t	st	0
5340	a thing of blood I' hi im logan and i love the ...	440	s	f	sf	0

In [37]:
```
sn_traing_set_size = 2100 # e.g. 2100 is 70% of 3000 samples
perc_s_train = perc_samples[perc_samples.perc_func == "s"].sample(sn_traing_set_size)
perc_n_train = perc_samples[perc_samples.perc_func == "n"].sample(sn_traing_set_size)

perc_train = perc_s_train.union(perc_n_train)

perc_samples.loc[perc_train, "perc_training_set"] = 1
perc_samples.head(3)
```

Out[37]:

	text	tokens	perc_func	judg_func	actual_temp	perc_training_set
8623	Sonny Jooooooooon INDEX ASK PAST THEME Sonny J...	386	s	f	sf	0
11987	Log in | Tumblr Sign up Terms Privacy Posted b...	52	s	t	st	1
5340	a thing of blood I' hi im logan and i love the ...	440	s	f	sf	1

In [38]:
```
len(perc_samples[perc_samples.perc_training_set == 1])
```

Out[38]: 4200

In [39]:
```
# Separate evaluation DataFrame
perc_eval_set = perc_samples[perc_samples.perc_training_set == 0]
perc_eval_set.head(3)
```

Out[39]:

	text	tokens	perc_func	judg_func	actual_temp	perc_training_set
8623	Sonny Jooooooooon INDEX ASK PAST THEME Sonny J...	386	s	f	sf	0
18909	Wit Beyond Measure Wit Beyond Measure Aug 14, ...	340	s	t	st	0
7557	IT'S ALL COMIN' DOWN ON US, BOYS why is my das...	336	s	t	st	0

In [40]:
```
len(perc_eval_set)
```

Out[40]: 1800

6.1 Train SN classifer

https://uclassify.com/manage/classifiers/jung-percieving-2100

```
In [41]: def train_jung_cognitive_functions_en_classes(func, classifier):
             """Presupposes that classifier is created and that setup_jung_functions_en_classe
             func: expects one of ["s","n","t","f"]
             classifier: expects on of ["sntf", "tf", "sn"]

             """
             trained_ix = []
             text_count = 1
             if classifier == "sn":
                 for ix, row in perc_samples.loc[(perc_samples.perc_func == name) & (perc_sampl
                     trained_ix.append(ix)
                     data = {"texts":[row["text"]]}
                     header = {"Content-Type": "application/json",
                          "Authorization": "Token " + os.environ["UCLASSIFY_WRITE"]}

                     response = requests.post('https://api.uclassify.com/v1/me/jung-percieving-
                             json = data,
                             headers = header)
                     if text_count % 100 == 0:
                         print("{}:{}".format(name, text_count))
                     text_count += 1

             elif classifier == "tf":
                 for ix, row in judg_samples.loc[(judg_samples.judg_func == name) & (judg_sampl
                     trained_ix.append(ix)
                     data = {"texts":[row["text"]]}
                     header = {"Content-Type": "application/json",
                          "Authorization": "Token " + os.environ["UCLASSIFY_WRITE"]}

                     response = requests.post('https://api.uclassify.com/v1/me/jung-judging-210
                             json = data,
                             headers = header)
                     if text_count % 100 == 0:
                         print("{}:{}".format(name, text_count))
                     text_count += 1


             print("Finished training Jung Cognitive Functions: {}".format(name))
             return trained_ix

In [42]: perc_trained_ix = []
         for name in ["s","n"]:
             functions_trained_ix = train_jung_cognitive_functions_en_classes(name, classifier=
             perc_trained_ix.append(perc_trained_ix)
```

```
s:100
s:200
s:300
s:400
s:500
s:600
s:700
s:800
s:900
s:1000
s:1100
s:1200
s:1300
s:1400
s:1500
s:1600
s:1700
s:1800
s:1900
s:2000
s:2100
Finished training Jung Cognitive Functions: s
n:100
n:200
n:300
n:400
n:500
n:600
n:700
n:800
n:900
n:1000
n:1100
n:1200
n:1300
n:1400
n:1500
n:1600
n:1700
n:1800
n:1900
n:2000
n:2100
Finished training Jung Cognitive Functions: n
```

```
In [43]: print("length perc_eval_set: {}".format(len(perc_eval_set)))
         print("length perc_trained_ix: {}".format(len(perc_trained_ix)))
```

```
length perc_eval_set: 1800
length perc_trained_ix: 2
```

7 Split judging samples into train and eval subsets.

In [45]:
```
# prepare column to keep track of what's been used for training
zeros = np.zeros(len(judg_samples))
judg_samples["judg_training_set"] = zeros
judg_samples["judg_training_set"] = judg_samples.judg_training_set.astype("int")
judg_samples.head(3)
```

Out[45]:

	text	tokens	perc_func	judg_func	actual_temp	judg_training_set
22981	it is what it is About Name: Heidi Age:16 Wher...	565	s	t	st	0
24378	https://www.tumblr.com/themes/by/leentheme htt...	582	n	t	nt	0
5187	three things cannot be long hidden I' three thi...	516	n	t	nt	0

In [46]:
```
tf_traing_set_size = 2100 # e.g. 2100 is 70% of 3000 samples
judg_t_train = judg_samples[judg_samples.judg_func == "t"].sample(tf_traing_set_size)
judg_f_train = judg_samples[judg_samples.judg_func == "f"].sample(tf_traing_set_size)

judg_train = judg_t_train.union(judg_f_train)

judg_samples.loc[judg_train, "judg_training_set"] = 1
judg_samples.head(15)
```

Out[46]:

	text	tokens	perc_func	judg_func	actual_temp	judg_training_set
22981	it is what it is About Name: Heidi Age:16 Wher...	565	s	t	st	0
24378	https://www.tumblr.com/themes/by/leentheme htt...	582	n	t	nt	0
5187	three things cannot be long hidden I' three thi...	516	n	t	nt	0
4307	none gf with left feel 12442 August 1st, 201...	499	s	t	nt	1
2606	God, Faith, & Fitness God, Faith, & Fitness Me...	924	n	t	nt	1
18214	big hype, big letdown i'm charlotte and i li...	101	s	t	nt	1
10718	Love the life you live Live the life you love ...	353	n	t	nt	1
20277	- - - - - - - | momo | 14 | | ESTJ | |...	473	s	t	nt	1
846	Cynically Marvelous | It's Axiomatic. Cynicall...	6973	n	t	nt	1
16405	I'll just pretend that youth will never end I'...	693	n	t	nt	1
11753	Something-or-other Something-or-other Let's ju...	483	n	t	nt	1
18004	/ ._> | | | '_>/ ._>| . \/ ._>| || |/ ._> | _/...	180	n	t	nt	1
23577	WINTERFELL About Mariele. 21. Germany. ISTJ. M...	102	s	t	nt	1
4829	The bears are in 684,715 plays we-r-who-...	445	s	t	nt	1
4225	XANADU XANADU	2	n	t	nt	1

```
      judg_func actual_temp  judg_training_set
22981         t          st                  1
24378         t          nt                  1
5187          t          nt                  1
4307          t          st                  1
2606          t          nt                  1
18214         t          st                  1
10718         t          nt                  1
20277         t          st                  1
846           t          nt                  1
16405         t          nt                  0
11753         t          nt                  1
18004         t          nt                  1
23577         t          st                  1
4829          t          st                  1
4225          t          nt                  1
```

In [47]:
```
# Separate evaluation DataFrame
judg_eval_set = judg_samples[judg_samples.judg_training_set == 0]
judg_eval_set.head(3)
```

Out[47]:
```
                                                    text  tokens perc_func  \
16405  I'll just pretend that youth will never end I'...     693         n
24806  The Queen The Queen Raquel Alexis | 17 | FL | ...      77         n
15132  this could have been worse this could have bee...     492         n

      judg_func actual_temp  judg_training_set
16405         t          nt                  0
24806         t          nt                  0
15132         t          nt                  0
```

7.1 Train TF classifier

In [57]:
```
tf_trained_ix = []
for name in ["t","f"]:
    try:
        tf_trained_ix = train_jung_cognitive_functions_en_classes(name, classifier="t
        tf_trained_ix.append(tf_trained_ix)
    except Exception as e:
        print(e)
```

```
t:100
t:200
t:300
t:400
t:500
t:600
t:700
t:800
```

```
t:900
t:1000
t:1100
t:1200
t:1300
t:1400
t:1500
t:1600
t:1700
t:1800
t:1900
t:2000
t:2100
t:2200
t:2300
t:2400
t:2500
t:2600
t:2700
t:2800
t:2900
t:3000
t:3100
t:3200
t:3300
t:3400
t:3500
t:3600
t:3700
t:3800
t:3900
t:4000
t:4100
t:4200
t:4300
t:4400
t:4500
t:4600
t:4700
t:4800
t:4900
t:5000
t:5100
t:5200
t:5300
t:5400
t:5500
t:5600
```

```
t:5700
t:5800
t:5900
t:6000
t:6100
t:6200
t:6300
t:6400
t:6500
t:6600
t:6700
t:6800
t:6900
t:7000
Finished training Jung Cognitive Functions.
f:100
f:200
f:300
f:400
f:500
f:600
f:700
f:800
f:900
f:1000
f:1100
f:1200
f:1300
f:1400
f:1500
f:1600
f:1700
f:1800
f:1900
f:2000
f:2100
f:2200
f:2300
f:2400
f:2500
f:2600
f:2700
f:2800
f:2900
f:3000
f:3100
f:3200
f:3300
```

```
f:3400
f:3500
f:3600
f:3700
f:3800
f:3900
f:4000
f:4100
f:4200
f:4300
f:4400
f:4500
f:4600
f:4700
f:4800
f:4900
f:5000
f:5100
f:5200
f:5300
f:5400
f:5500
f:5600
f:5700
f:5800
f:5900
f:6000
f:6100
f:6200
f:6300
f:6400
f:6500
f:6600
f:6700
f:6800
f:6900
f:7000
Finished training Jung Cognitive Functions.
```

```
In [48]: print("length judg_eval_set: {}".format(len(judg_eval_set)))
         print("length tf_trained_ix: {}".format(len(tf_trained_ix)))
```

```
length judg_eval_set: 1800
```

```
NameError                                 Traceback (most recent call last)

<ipython-input-48-9a63d8973ddd> in <module>()
  1 print("length judg_eval_set: {}".format(len(judg_eval_set)))
----> 2 print("length tf_trained_ix: {}".format(len(tf_trained_ix)))


NameError: name 'tf_trained_ix' is not defined
```

8 Classify percieving function

```
In [50]: def classify_jung_percieving_function_of_text(text):
             """Does what it says, pretty much."""
             header = {"Content-Type": "application/json",
                       "Authorization": "Token " + os.environ["UCLASSIFY_READ"]}
             data = {"texts":[text]} # send a one-item list for now, since we don't have a fee
             result = requests.post("https://api.uclassify.com/v1/prfekt/jungian-cognitive-fun
                                    json = data,
                                    headers = header)
             json_result = result.json()

             res_dict = {"s":0, "n":0}

             for classItem in json_result[0]["classification"]:
                 res_dict[classItem["className"]] = classItem["p"]

             sorted_dict = sorted(res_dict.items(), key=operator.itemgetter(1), reverse=True)
             return sorted_dict

In [57]: zeros = np.zeros(len(perc_eval_set))
         sn_results = []
         row_cnt = 1
         for ix, row in perc_eval_set.iterrows():
             print("row: {} of {}".format(row_cnt, len(perc_eval_set)),end="\r")
             res = classify_jung_percieving_function_of_text(row["text"])
             sn_results.append(res[0][0])
             row_cnt += 1

row: 1800 of 1800

In [58]: len(sn_results)

Out[58]: 1800
```

Add the percieving classification results to the evaulation dataset

```
In [59]: perc_eval_set = pd.concat([perc_eval_set,
                                    pd.DataFrame(sn_results, index=perc_eval_set.index)
                                   ], axis=1, ignore_index=True)
         perc_eval_set.columns = ["text","tokens","perc_func","judg_func","actual_temp","perc_t
         perc_eval_set.to_pickle("classification_results_percieving_function_blogs_n5000_datafi
         print(perc_eval_set.head(3))

                                                      text  tokens perc_func  \
8623   Sonny Jooooooooon INDEX ASK PAST THEME Sonny J...     386         s
18909  Wit Beyond Measure Wit Beyond Measure Aug 14, ...     340         s
7557   IT'S ALL COMIN' DOWN ON US, BOYS why is my das...     336         s

      judg_func actual_temp  perc_training_set sn
8623          f          sf                  0  s
18909         t          st                  0  n
7557          t          st                  0  s
```

9 Classify TF

```
In [60]: def classify_jung_judging_function_of_text(text):
             """Does what it says, pretty much."""
             header = {"Content-Type": "application/json",
                       "Authorization": "Token " + os.environ["UCLASSIFY_READ"]}
             data = {"texts":[text]} # send a one-item list for now, since we don't have a fee
             result = requests.post("https://api.uclassify.com/v1/prfekt/jungian-cognitive-func
                                    json = data,
                                    headers = header)
             json_result = result.json()

             res_dict = {"t":0, "f":0}

             for classItem in json_result[0]["classification"]:
                 res_dict[classItem["className"]] = classItem["p"]

             sorted_dict = sorted(res_dict.items(), key=operator.itemgetter(1), reverse=True)
             return sorted_dict

In [63]: zeros = np.zeros(len(judg_eval_set))
         tf_results = []
         row_cnt = 1
         for ix, row in judg_eval_set.iterrows():
             print("row: {} of {}".format(row_cnt, len(judg_eval_set)),end="\r")
             res = classify_jung_judging_function_of_text(row["text"])
             tf_results.append(res[0][0])
             row_cnt += 1

row: 1800 of 1800
```

Add judging classification results to evaluation set

```
In [64]: judg_eval_set = pd.concat([judg_eval_set,
                                    pd.DataFrame(tf_results, index=judg_eval_set.index)
                                   ], axis=1, ignore_index=True)
         judg_eval_set.columns = ["text","tokens","perc_func","judg_func","actual_temp","judg_t
         judg_eval_set.to_pickle("classification_results_judging_function_blogs_n10000_datafra
         print(judg_eval_set.head(3))
```

```
                                                    text  tokens perc_func  \
16405  I'll just pretend that youth will never end I'...     693         n
24806  The Queen The Queen Raquel Alexis | 17 | FL | ...      77         n
15132  this could have been worse this could have bee...     492         n

      judg_func actual_temp  judg_training_set tf
16405         t          nt                  0  t
24806         t          nt                  0  t
15132         t          nt                  0  t
```

10 Evaluation of percieving classification

10.1 Classification report percieving

```
In [65]: sn_cr = classification_report(perc_eval_set['perc_func'], perc_eval_set['sn'])
         print(sn_cr)
```

```
             precision    recall  f1-score   support

          n       0.87      0.86      0.87       900
          s       0.86      0.88      0.87       900

avg / total       0.87      0.87      0.87      1800
```

10.2 Percieving accuracy

```
In [66]: sn_accuracy = sum(perc_eval_set['perc_func']==perc_eval_set['sn'])/len(perc_eval_set)
         print(sn_accuracy)
```

```
0.8688888888888889
```

10.3 Percieving Kappa

```
In [67]: sn_kappa = sn_accuracy - (1/len(perc_eval_set))/(1 - len(perc_eval_set))
         print(sn_kappa)
```

```
0.8688891977024273
```

11 Evaluation of judging classificiation

11.1 Classification report judging

```
In [68]: tf_cr = classification_report(judg_eval_set['judg_func'], judg_eval_set['tf'])
         print(tf_cr)
```

```
             precision    recall  f1-score   support

          f       0.87      0.90      0.88       900
          t       0.89      0.86      0.88       900

avg / total       0.88      0.88      0.88      1800
```

11.2 Judging accuracy

```
In [69]: tf_accuracy = sum(judg_eval_set['judg_func']==judg_eval_set['tf'])/len(judg_eval_set)
         print(tf_accuracy)
```

```
0.8794444444444445
```

11.3 Judging Kappa

```
In [70]: tf_kappa = tf_accuracy - (1/len(judg_eval_set))/(1 - len(judg_eval_set))
         print(tf_kappa)
```

```
0.8794447532579829
```

12 Conclusion and further research

- Two dichotomic classifiers seems to improve the results tremendously.
- Is the experiment done correctly? Peer-review the code.
- Check what words, and later, phrases are the most influential for each class.