irr_measures

July 18, 2022

1 COVID-19 NLP Annotation

1.1 Inter-rater Reliability (IRR) Measures

1.2 Author: Will Bowers & David Carrell

This notebook contains the computation of inter-rater reliability measures. These measures are computed based off of manual review of text snippets from the NLP COVID-19 study with David Carrell. Manual review was performed by Ann Kelley and Will Bowers. There are 50 snippets in total, ten snippets each for five different features: *cough, diarrhea, fever, nausea, and vomiting.*

Reviewers were asked two questions for each snippet:
- Is this term referring to the feature?
- Responses: yes, no, uncertain.

- Does the patient have this feature?
  - Responses: yes, no, uncertain.

1.3 Key Findings

- For question 1, "Is this term referring to the feature?", there is 100% agreement between raters.
  - Cohen's Kappa may not be computed if there is 100% agreement due to divide by zero error.
- For question 2, "Does the patient have this feature?", kappa ~ 0.899.
  - This indicates near perfect agreement.

1.4 Imports

[62]:
```
import os

import numpy as np
import pandas as pd
```

1.5 Load data

1.5.1 Ann's Review

[73]:
```
ann_df = pd.concat(pd.read_excel('../../PheNorm/ManualReview/
 ↪symptoms_20220628_irr/features.review.sample10_Ann.xlsx',
                                  sheet_name=None),
                   ignore_index=True)
```

[74]:
```
ann_df.rename(columns={'Is this term referring to the feature?': 'is_feature?',
                       'Does the patient have this feature?': 'patient_has?'},
              inplace=True)
```

Verify length of Ann's DataFrame.

[11]:
```
len(ann_df)
```

[11]: 50

1.5.2 Will's Review

[29]:
```
will_df = pd.concat(pd.read_excel('../../PheNorm/ManualReview/
 ↪symptoms_20220628_irr/features.review.sample10_Will.xlsx',
                                  sheet_name=None),
                    ignore_index=True)
```

[31]:
```
will_df.rename(columns={'Is this term referring to the feature?': 'is_feature?',
                        'Does the patient have this feature?': 'patient_has?'},
               inplace=True)
```

Verify length of Will's DataFrame

[18]:
```
len(will_df)
```

[18]: 50

1.6 Cohen's Kappa

$\mathrm{k} = \frac{P_o - P_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e}$

1.6.1 Question 1: Is this term referring to the feature?

Calculate P_o - the observed proportional agreement $P_o = n_{agree}$ / N = ($n_{yes} + n_{no} + n_{uncertain}$) / N

[102]:
```
counts = (ann_df['is_feature?'] == will_df['is_feature?']).value_counts()
total_agree = counts[True]
total = len(ann_df)
```

```
p_o = total_agree / total
p_o
```

[102]: 1.0

Calculate P_{yes} - the probability both raters would randomly say 'yes'.

[43]:
```
ann_prob_yes = len(ann_df[ann_df['is_feature?'].str.lower() == 'yes']) / total
will_prob_yes = len(will_df[will_df['is_feature?'].str.lower() == 'yes']) /␣
↪total

prob_yes = ann_prob_yes * will_prob_yes
prob_yes
```

[43]: 1.0

Calculate P_{no} - the probability both raters would randomly say 'no'.

[44]:
```
ann_prob_no = len(ann_df[ann_df['is_feature?'].str.lower() == 'no']) / total
will_prob_no = len(will_df[will_df['is_feature?'].str.lower() == 'no']) / total

prob_no = ann_prob_no * will_prob_no
prob_no
```

[44]: 0.0

Calculate $P_{uncertain}$ - the probability both raters would randomly say 'uncertain'.

[54]:
```
ann_prob_unk = len(ann_df[ann_df['is_feature?'].str.lower() == 'uncertain']) /␣
↪total
will_prob_unk = len(will_df[will_df['is_feature?'].str.lower() == 'uncertain'])␣
↪/ total

prob_unk = ann_prob_unk * will_prob_unk
prob_unk
```

[54]: 0.0

Calculate P_e - the hypothetical probability of chance agreement $P_e = P_{yes} + P_{no} + P_{uncertain}$

[55]:
```
p_e = prob_yes + prob_no + prob_unk
p_e
```

[55]: 1.0

Calculate k

[56]:
```
num = p_o - p_e
denom = 1 - p_e
k = num / denom
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Input In [56], in <cell line: 3>()
      1 num = p_o - p_e
      2 denom = 1 - p_e
----> 3 k = num / denom

ZeroDivisionError: float division by zero
```

In cases of total agreement, Cohen's Kappa can not be computed as $P_e = 1$, 1 - 1 = 0, and we cannot divide by 0.

1.6.2 Question 2: Does the patient have this feature?

Calculate P_o - the observed proportional agreement $P_o = n_{agree}$ / N = ($n_{yes} + n_{no} + n_{uncertain}$) / N

[103]:
```
counts = (ann_df['patient_has?'] == will_df['patient_has?']).value_counts()
total_agree = counts[True]
total = len(ann_df)

p_o = total_agree / total
p_o
```

[103]: 0.96

Calculate P_{yes} - the probability both raters would randomly say 'yes'.

[104]:
```
ann_prob_yes = len(ann_df[ann_df['patient_has?'].str.lower() == 'yes']) / total
will_prob_yes = len(will_df[will_df['patient_has?'].str.lower() == 'yes']) /␣
 ↪total

prob_yes = ann_prob_yes * will_prob_yes
prob_yes
```

[104]: 0.04000000000000001

Calculate P_{no} - the probability both raters would randomly say 'no'.

[105]:
```
ann_prob_no = len(ann_df[ann_df['patient_has?'].str.lower() == 'no']) / total
will_prob_no = len(will_df[will_df['patient_has?'].str.lower() == 'no']) / total

prob_no = ann_prob_no * will_prob_no
```

```
prob_no
```

[105]: 0.5624

Calculate $P_{uncertain}$ - the probability both raters would randomly say 'uncertain'.

[106]:
```
ann_prob_unk = len(ann_df[ann_df['patient_has?'].str.lower() == 'uncertain']) /␣
 ↪total
will_prob_unk = len(will_df[will_df['patient_has?'].str.lower() ==␣
 ↪'uncertain']) / total

prob_unk = ann_prob_unk * will_prob_unk
prob_unk
```

[106]: 0.0024

Calculate P_e - the hypothetical probability of chance agreement $P_e = P_{yes} + P_{no} + P_{uncertain}$

[107]:
```
p_e = prob_yes + prob_no + prob_unk
p_e
```

[107]: 0.6048

Calculate k

[109]:
```
num = p_o - p_e
denom = 1 - p_e
k = num / denom
k
```

[109]: 0.8987854251012145

k = 0.8987854251012145, this indicates near perfect agreement

[ ]: