Some Stats

July 2, 2019

[1]:
```
import pandas as pd
import numpy as np
from scipy import stats

data_by_question = pd.read_csv("surveys.csv", header=1)
data_by_code = pd.read_csv("surveys.csv", header=0, skiprows=[2])

def score(questions, answers, reverse_questions, subscales=[],␣
 ↪data=data_by_code, missing='fill'):
    scores_list = []
    if subscales:
        subscores_lists = [[] for i in range(len(subscales))]
    for index, participant in data_by_code.iterrows():

        # reset score counters
        score_total = 0
        if subscales:
            subscores_total = [0 for i in range(len(subscales))]
        # Convert written answers to cumulative scores
        for question in questions:
            answer = participant[question]
            if answer not in answers:
                if missing == 'fill':
                    answer = answers[int((len(answers) - 1) / 2)]  # Will not␣
 ↪work if len of answers not odd

            q_score = (answers.index(answer)) - ((len(answers) - 1) / 2)  # e.g.␣
 ↪'strongly agree' = (4+1 - (5-1)/2) = 3
            # print(q_score)
            # Add to total and normalize
            question_index = questions.index(question) + 1
            if question_index in reverse_questions:
                q_score = q_score * -1

            score_total += (float(q_score) / len(questions))

            if subscales:
```

```
                for i in range(len(subscales)):
                    if question_index in subscales[i]:
                        subscores_total[i] += float(q_score)

        scores_list.append(score_total)
        if subscales:
            for i in range(len(subscales)):
                subscores_lists[i].append(subscores_total[i] / len(subscales[i]))

    if subscales:
        return (scores_list, subscores_lists)
    return (scores_list)

def scoring_trust(section):
    answers = ['Strongly Disagree', 'Disagree', 'Somewhat Disagree', 'Neither␣
↪Agree nor Disagree', 'Somewhat Agree', 'Agree', 'Strongly Agree']  # Higher␣
↪scores indicate a more negative attitude
    robot_trust_questions = [section + str(i) for i in range(1, 33 + 1)]     #␣
↪'Q18_1' ->'Q18_33'
    subscale_1 = [i for i in range(1, 6 + 1)]
    subscale_2 = [i for i in range(6 + 1, 13 + 1)]
    subscale_3 = [i for i in range(13 + 1, 33 + 1)]
    r = [2, 4, 6, 7, 8, 12, 14, 18]
    subscales = [subscale_1, subscale_2, subscale_3]
    scores, subscores = score(robot_trust_questions, answers, r,␣
↪subscales=subscales)
    [sub_1, sub_2, sub_3] = subscores
    return(scores)
```

[10]:
```
"""
This cell looks at the difference in trust values between the
participants before and after the session
"""

a = scoring_trust("Q33_")
b = scoring_trust("Q34_")
c = scoring_trust("Q35_")

a = np.array(a)
b = np.array(b)
c = np.array(c)

x = scoring_trust("Q57_")
y = scoring_trust("Q58_")
z = scoring_trust("Q59_")

x = np.array(x)
```

```
y = np.array(y)
z = np.array(z)

"""
The scores are only divided by two here because the participants only
fill out two out of the three evaluation forms (excluding their own
position)
"""

before = (a+b+c)/2
after = (x+y+z)/2

print("Mean of trust before: ", np.mean(before))
print("Standard Deviation of trust before: ", np.std(before))
print("Mean of trust after: ", np.mean(after))
print("Standard Deviation of trust after: ", np.std(after))

"""
A T-test is then performed to determine statistical significance. The
p-value is divided by two for a one-tail test
"""

t, p = stats.ttest_ind(after, before)
print("\n")
print("t-value = " + str(t))
print("p-value = " + str(p/2))
```

```
Mean of trust before:  0.3200295639320028
Standard Deviation of trust before:  0.6454338566334004
Mean of trust after:  0.8344419807834438
Standard Deviation of trust after:  0.6466402664292145


t-value = 5.067355021059815
p-value = 5.448384284789485e-07
```

We can conclude with almost 100% confidence that there was an increase of trust between the participants after the sessions.

[15]:
```
"""
This cell looks at the difference the trust values given to the robot
by the participants before and after the sessions.
"""

a = scoring_trust("Q18_")
b = scoring_trust("Q56_")

print("Mean of trust before: ", np.mean(a))
```

```python
print("Standard Deviation of trust before: ", np.std(a))
print("Mean of trust after: ", np.mean(b))
print("Standard Deviation of trust after: ", np.std(b))

t, p = stats.ttest_ind(b, a)
print("\n")
print("t-value = " + str(t))
print("p-value = " + str(p/2))
```

```
Mean of trust before:  0.237250554323725
Standard Deviation of trust before:  0.6279100075502677
Mean of trust after:  0.6592756836659273
Standard Deviation of trust after:  0.7284045297683023


t-value = 3.9495402793244265
p-value = 5.828742204363072e-05
```

We can conclude with almost 100% confidence that there was an increase of trust given to the robot by the participants after the going through the session.

[28]:
```python
"""
This cell compares the trust values given to a stranger and the trust
values given to the robot by the participants.
"""
data_by_code = data_by_code[2:]

a = data_by_code['Q17_3'].astype('float64').dropna()
b = data_by_code['Q17_4'].astype('float64').dropna()

print("Before the Session:")

print("Mean of trust in strangers before: ", np.mean(a))
print("Standard Deviation of trust in strangers before: ", np.std(a))
print("Mean of trust in robot before: ", np.mean(b))
print("Standard Deviation of trust in robot before: ", np.std(b))

t, p = stats.ttest_ind(b, a)
print("\n")
print("t-value = " + str(t))
print("p-value = " + str(p/2))
print("\n")

print("After the Session:")

a = data_by_code['Q60_3'].astype('float64').dropna()
b = data_by_code['Q60_4'].astype('float64').dropna()
```

```python
print("Mean of trust in strangers after: ", np.mean(a))
print("Standard Deviation of trust in strangers after: ", np.std(a))
print("Mean of trust in robot after: ", np.mean(b))
print("Standard Deviation of trust in robot after: ", np.std(b))

t, p = stats.ttest_ind(b, a)
print("\n")
print("t-value = " + str(t))
print("p-value = " + str(p/2))
```

```
Before the Session:
Mean of trust in strangers before:  42.58225806451613
Standard Deviation of trust in strangers before:  24.748225900531242
Mean of trust in robot before:  44.975
Standard Deviation of trust in robot before:  24.627342378889743


t-value = 0.5477387426831144
p-value = 0.29241259115999574


After the Session:
Mean of trust in strangers after:  44.42388059701493
Standard Deviation of trust in strangers after:  24.571492337406685
Mean of trust in robot after:  52.38235294117647
Standard Deviation of trust in robot after:  25.71839546029074


t-value = 1.8242179468683541
p-value = 0.03518189793073869
```

I just thought that this relationship was interesting as, before the session, we cannot conclude that the participants give more trust to the robot than strangers. After the session, there is a significant difference showing that we can conclude with about 96% confidence that the participants give more trust to the robot than a stranger.