In [17]:
```python
import matplotlib as plt
import seaborn as sns
import pandas as pd
import numpy as np
import json as js
```

In [18]:
```python
pwd
```

Out[18]: '/Users/mingyuanma/Desktop/HAI/analysis'

In [19]:
```python
before = pd.read_csv("../../data/phase1/combined1.csv")
before = before[before["round"] >= 3]
```

In [20]:
```python
after = pd.read_csv("../../data/phase2/combined2.csv")
after = after[after["round"] >= 3]
```

In [ ]:

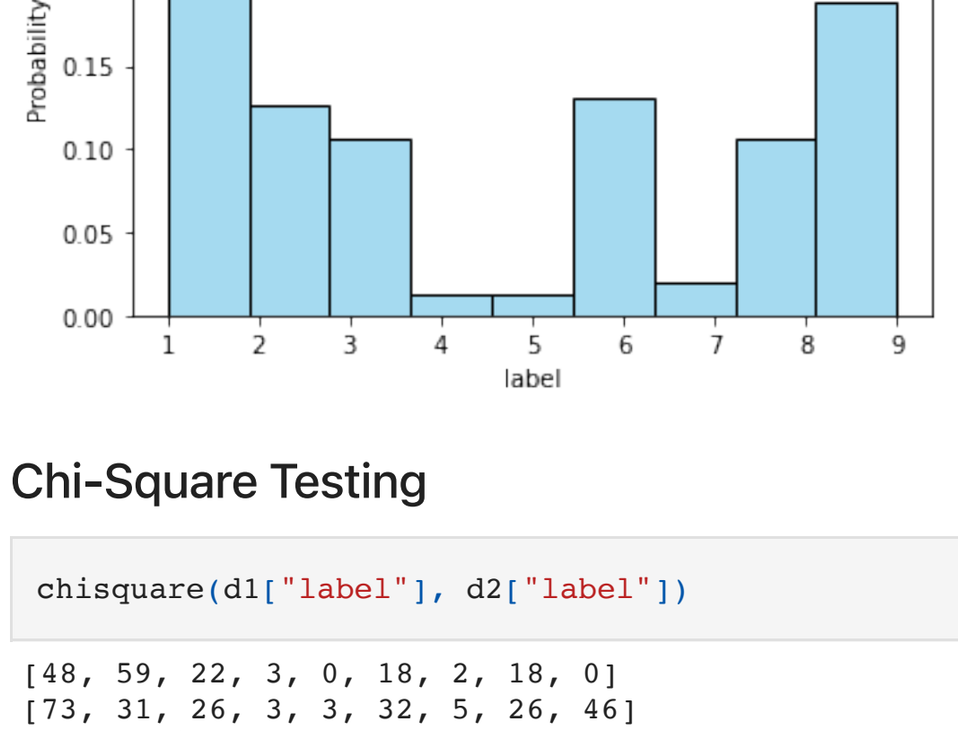
chi-square testing

In [21]:
```python
from scipy.stats import chi2_contingency
from collections import Counter

def chisquare(array1, array2, count=True):
    if not count:
        data = [array1, array2]
        stat, p, dof, expected = chi2_contingency(data)

    else:
        c1, c2 = Counter(array1), Counter(array2)
        before_dis, after_dis = [], []
        for i in set(c1).union(set(c2)):
            before_dis.append(c1[i])
            after_dis.append(c2[i])
        print(before_dis)
        print(after_dis)
        data = [before_dis, after_dis]
        stat, p, dof, expected = chi2_contingency(data)

    alpha = 0.05
    print("p value is " + str(p))
    if p <= alpha:
        print('difference between the two distributions (reject H0)')
    else:
        print('no difference between the two distributions (H0 holds true)')
```

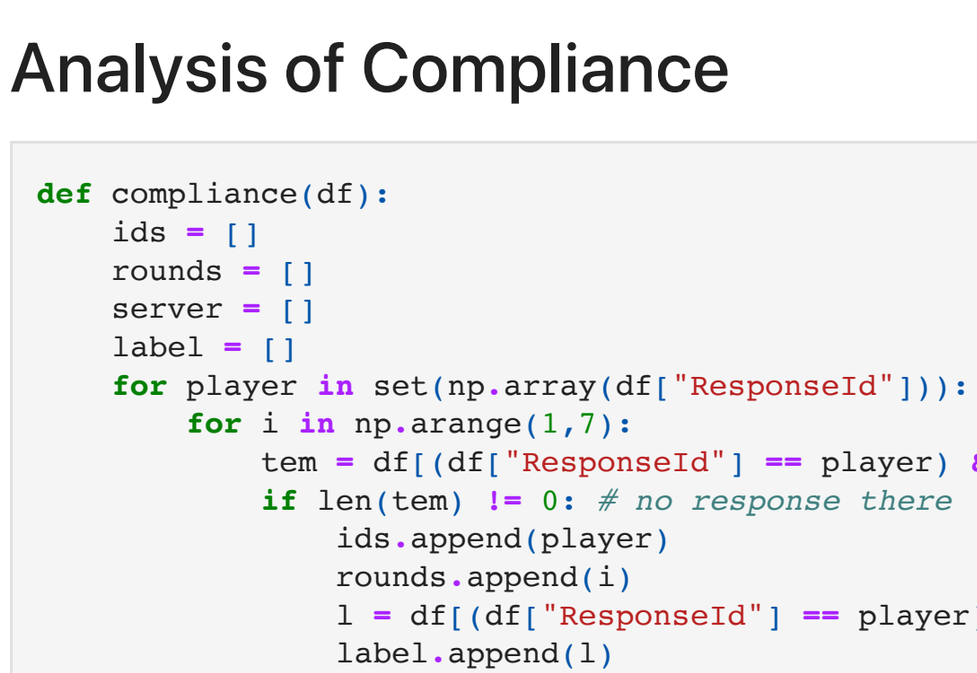
Distribution of label before and after

In [22]:
```python
d1 = before.groupby("ResponseId", as_index=False).agg(lambda x:x.iloc[0])
d2 = after.groupby("ResponseId", as_index=False).agg(lambda x:x.iloc[0])
```

In [23]:
```python
sns.histplot(data=d1, x="label", stat="probability", color="skyblue");
```



In [24]:
```python
sns.histplot(data=d2, x="label", stat="probability", color="skyblue");
```



Chi-Square Testing

In [25]:
```python
chisquare(d1["label"], d2["label"])
```

```
[48, 59, 22, 3, 0, 18, 2, 18, 0]
[73, 31, 26, 3, 3, 32, 5, 26, 46]
p value is 1.0415585870358675e-09
difference between the two distributions (reject H0)
```

Analysis of Compliance

In [52]:
```python
def compliance(df):
    ids = []
    rounds = []
    server = []
    label = []
    for player in set(np.array(df["ResponseId"])):
        for i in np.arange(1,7):
            tem = df[(df["ResponseId"] == player) & (df["round"] == i)]
            if len(tem) != 0: # no response there
                ids.append(player)
                rounds.append(i)
                l = df[(df["ResponseId"] == player)]["label"].iloc[0]
                label.append(l)
                if i <= 2:
                    num = sum((tem["workerId"] == 2) & (tem["taskId"] == 1))
                else:
                    num = sum((tem["workerId"] == 1) & (tem["taskId"] == 1))
                server.append(num)
#            else:
#                print(player, i)
    d = {
        "ResponseId": ids,
        "round": rounds,
        "numServerCook": server,
        "label": label
    }
    return pd.DataFrame(d)
```

In [53]:
```python
before_compliance = compliance(before)
```

In [54]:
```python
after_compliance = compliance(after)
```

In [29]:
```python
before_compliance
```

Out[29]:

	ResponseId	round	numServerCook	label
0	R_2OZq8YOITqiMVDf	3	2	2
1	R_2OZq8YOITqiMVDf	4	3	2
2	R_2OZq8YOITqiMVDf	5	2	2
3	R_2OZq8YOITqiMVDf	6	3	2
4	R_3kyKwP8ts3QFYh5	3	3	1
...	...	...	...	...
675	R_vTSzwpAtFN8uqZj	6	4	6
676	R_1gCM2pcwrB60Lti	3	2	2
677	R_1gCM2pcwrB60Lti	4	2	2
678	R_1gCM2pcwrB60Lti	5	2	2
679	R_1gCM2pcwrB60Lti	6	2	2

680 rows × 4 columns

In [30]:
```python
after_compliance
```

Out[30]:

	ResponseId	round	numServerCook	label
0	R_3oSM6w545XM6p6T	3	1	1
1	R_3oSM6w545XM6p6T	4	3	1
2	R_3oSM6w545XM6p6T	5	2	1
3	R_3oSM6w545XM6p6T	6	1	1
4	R_1mf7bqHoccP7ipz	3	3	2
...	...	...	...	...
975	R_3KMC9Sle6fCZt4P	6	1	9
976	R_BWbwwiykDgkKmat	3	2	6
977	R_BWbwwiykDgkKmat	4	2	6
978	R_BWbwwiykDgkKmat	5	2	6
979	R_BWbwwiykDgkKmat	6	2	6

980 rows × 4 columns

In [ ]:

group by rounds

In [31]:
```python
round_before = before_compliance.groupby("round").agg(np.average).loc[:,["numServerCook"]]
round_before
```

Out[31]:

round	numServerCook
3	2.552941
4	2.670588
5	2.617647
6	2.500000

In [32]:
```python
round_after = after_compliance.groupby("round").agg(np.average).loc[:,["numServerCook"]]
round_after
```

Out[32]:

round	numServerCook
3	2.408163
4	2.232653
5	2.163265
6	2.044898

In [33]:
```python
a = np.array(round_before["numServerCook"])
```

In [34]:
```python
b = np.array(round_after["numServerCook"])
```

chi-square testing

In [35]:
```python
chisquare(a, b, count=False)
```

```
p value is 0.9994561534639331
no difference between the two distributions (H0 holds true)
```

In [ ]:

group by tip

In [36]:
```python
tip_before = before_compliance.groupby("label", as_index=False).agg(np.average).loc[:,["label","numServerCook"]]
tip_before
```

Out[36]:

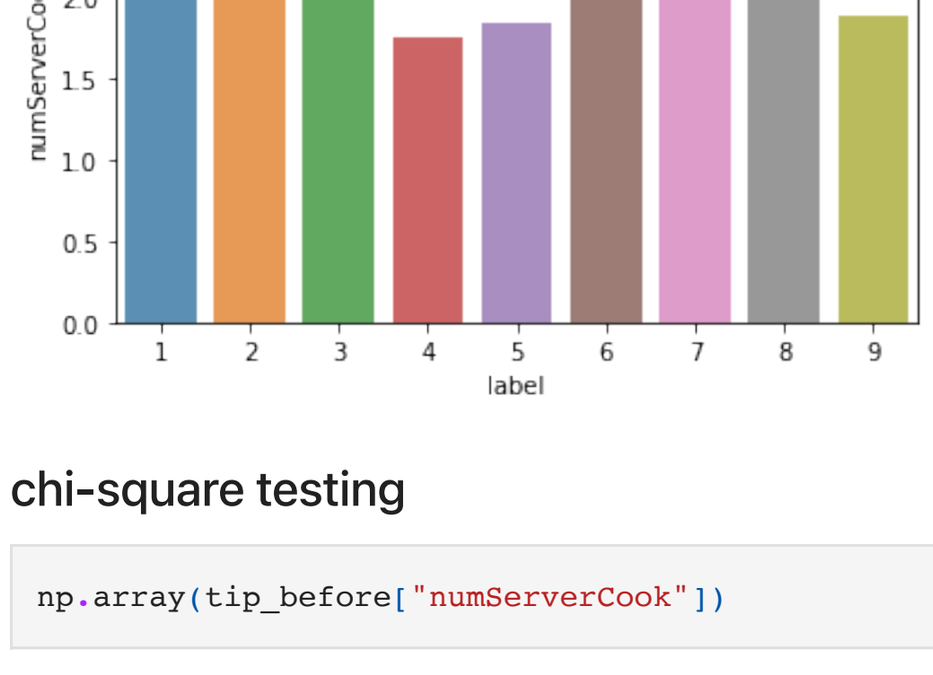
	label	numServerCook
0	1	2.609375
1	2	2.516949
2	3	2.477273
3	4	2.500000
4	6	2.736111
5	7	3.375000
6	8	2.652778

In [37]:
```python
tip_after = after_compliance.groupby("label", as_index=False).agg(np.average).loc[:,["label","numServerCook"]]
tip_after
```

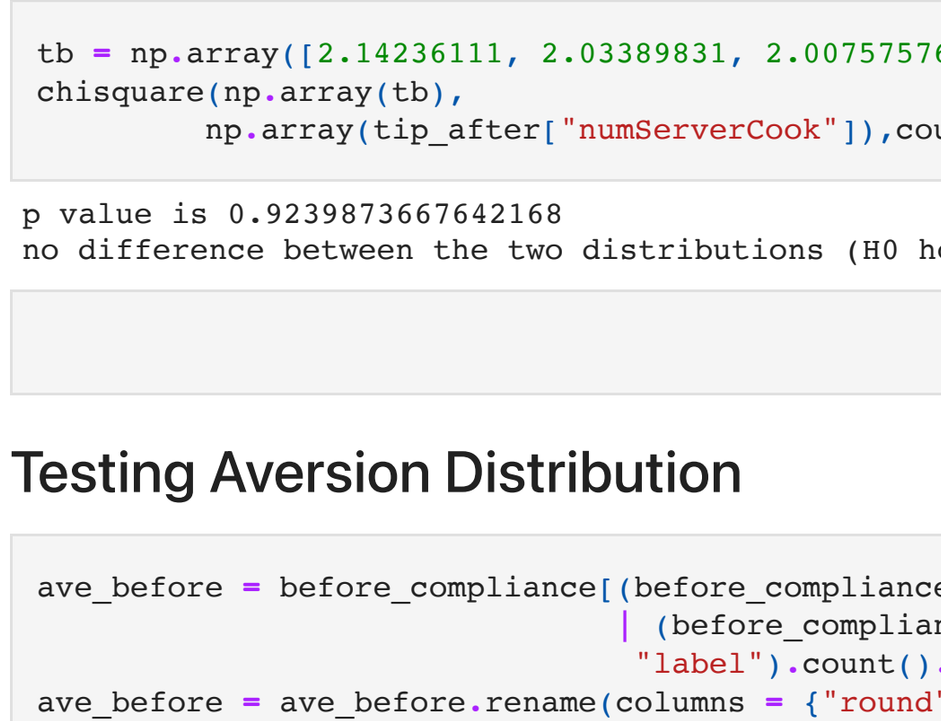
Out[37]:

	label	numServerCook
0	1	2.171233
1	2	2.193548
2	3	2.538462
3	4	1.750000
4	5	1.833333
5	6	2.164062
6	7	3.050000
7	8	2.615385
8	9	1.875000

In [38]:
```python
sns.barplot(x="label", y="numServerCook", data=tip_before, alpha=0.8);
```



In [39]:
```python
sns.barplot(x="label", y="numServerCook", data=tip_after, alpha=0.8);
```



chi-square testing

In [40]:
```python
np.array(tip_before["numServerCook"])
```

Out[40]:
```
array([2.609375  , 2.51694915, 2.47727273, 2.5       , 2.73611111,
       3.375     , 2.65277778])
```

In [41]:
```python
tb = np.array([2.14236111, 2.03389831, 2.00757576, 1.94444444, 0, 2.15740741, 2.41666667, 2.05555556, 0])
chisquare(np.array(tb),
          np.array(tip_after["numServerCook"]),count=False)
```

```
p value is 0.9239873667642168
no difference between the two distributions (H0 holds true)
```

In [ ]:

Testing Aversion Distribution

In [42]:
```python
ave_before = before_compliance[(before_compliance["numServerCook"] >= 3)
                               | (before_compliance["numServerCook"] == 0)].groupby(
                                "label").count().loc[:,["round"]]
ave_before = ave_before.rename(columns = {"round":"numAversion"})
```

In [43]:
```python
ave_after = after_compliance[(after_compliance["numServerCook"] >= 3)
                               | (after_compliance["numServerCook"] == 0)].groupby(
                                "label").count().loc[:,["round"]]
ave_after = ave_after.rename(columns = {"round":"numAversion"})
```

In [44]:
```python
def counting(df):
    count = []
    for i in np.arange(1,10):
        cou = len(df[df["label"] == i])
        count.append(cou)
    return count
```

In [45]:
```python
counting(before_compliance), counting(after_compliance)
[_ for _ in counting(before_compliance) if _]
```

Out[45]: [192, 236, 88, 12, 72, 8, 72]

In [46]:
```python
ave_before['total_num_label'] = [_ for _ in counting(before_compliance) if _]
ave_before["proportion_aversion"] = ave_before["numAversion"] / ave_before['total_num_label']
ave_before["proportion_label"] = ave_before['total_num_label'] / before_compliance.shape[0]
ave_before
```

Out[46]:

label	numAversion	total_num_label	proportion_aversion	proportion_label
1	71	192	0.369792	0.282353
2	99	236	0.419492	0.347059
3	43	88	0.488636	0.129412
4	4	12	0.333333	0.017647
6	27	72	0.375000	0.105882
7	8	8	1.000000	0.011765
8	34	72	0.472222	0.105882

In [47]:
```python
ave_after['total_num_label'] = counting(after_compliance)
ave_after["proportion_aversion"] = ave_after["numAversion"] / ave_after['total_num_label']
ave_after["proportion_label"] = ave_after['total_num_label'] / after_compliance.shape[0]
ave_after
```

Out[47]:

label	numAversion	total_num_label	proportion_aversion	proportion_label
1	69	292	0.236301	0.297959
2	36	124	0.290323	0.126531
3	47	104	0.451923	0.106122
4	1	12	0.083333	0.012245
5	2	12	0.166667	0.012245
6	24	128	0.187500	0.130612
7	14	20	0.700000	0.020408
8	39	104	0.375000	0.106122
9	31	184	0.168478	0.187755

In [ ]:

Testing Compliance Distribution

In [48]:
```python
com_before = before_compliance[before_compliance["numServerCook"] ==2].groupby(
                                "label").count().loc[:,["round"]]
com_before = com_before.rename(columns = {"round":"numCompliance"})
```

In [49]:
```python
com_after = after_compliance[after_compliance["numServerCook"] ==2].groupby(
                                "label").count().loc[:,["round"]]
com_after = com_after.rename(columns = {"round":"numCompliance"})
com_after
```

Out[49]:

label	numCompliance
1	171
2	66
3	48
4	7
5	6
6	89
7	3
8	45
9	77

In [50]:
```python
com_before['total_num_label'] = [288, 354, 132, 18, 108, 108]
com_before["proportion_compliance"] = com_before["numCompliance"] / com_before['total_num_label']
com_before["proportion_label"] = com_before['total_num_label'] / before_compliance.shape[0]
com_before
```

Out[50]:

label	numCompliance	total_num_label	proportion_compliance	proportion_label
1	117	288	0.406250	0.423529
2	132	354	0.372881	0.520588
3	45	132	0.340909	0.194118
4	7	18	0.388889	0.026471
6	42	108	0.388889	0.158824
8	29	108	0.268519	0.158824

In [51]:
```python
com_after['total_num_label'] = counting(after_compliance)
com_after["proportion_compliance"] = com_after["numCompliance"] / com_after['total_num_label']
com_after["proportion_label"] = com_after['total_num_label'] / after_compliance.shape[0]
com_after
```

Out[51]:

label	numCompliance	total_num_label	proportion_compliance	proportion_label
1	171	292	0.585616	0.297959
2	66	124	0.532258	0.126531
3	48	104	0.461538	0.106122
4	7	12	0.583333	0.012245
5	6	12	0.500000	0.012245
6	89	128	0.695312	0.130612
7	3	20	0.150000	0.020408
8	45	104	0.432692	0.106122
9	77	184	0.418478	0.187755

In [ ]:

In [ ]:

In [ ]: