math_hw_6

January 12, 2025

[93]:
```
import pandas as pd
import numpy as np
from scipy import stats
import statsmodels.api as sm
import itertools
import pandas as pd
import numpy as np
n_rows = 50000
np.random.seed(42)
ages = np.random.choice(
    ['Under 18', '18-25', '26-35', '36-50', '50+'],
    size=n_rows,
    p=[0.15, 0.35, 0.25, 0.15, 0.1]
)
#Длительность сессии
duration_means = {'Under 18': 20, '18-25': 20, '26-35': 25, '36-50':
  ↪22, '50+': 18}
duration_stds = {'Under 18': 7, '18-25': 7, '26-35': 8, '36-50': 6,
  ↪'50+': 5}
session_durations = [
    max(0, np.random.normal(duration_means[age], duration_stds[age]))
    for age in ages
]
#Количество прослушанных треков за сессию
tracks_listened = [
    max(1, int(duration / np.random.uniform(3, 5)))
    for duration in session_durations
]
#Количество уникальных исполнителей
unique_artists = [
    max(1, int(tracks / np.random.uniform(1.5, 3)))
    for tracks in tracks_listened
]

dataset = pd.DataFrame({
    'age_category': ages,
    'session_duration': session_durations,
```

```
    'tracks_listened': tracks_listened,
    'unique_artists': unique_artists
})
file_path = 'synthetic_music_sessions.csv'
dataset.to_csv(file_path, index=False)
dataset.info
```

[93]:
```
<bound method DataFrame.info of       age_category  session_duration
tracks_listened  unique_artists
0              18-25         19.866556                 4                 1
1                50+         24.036442                 6                 3
2              26-35         30.002340                 7                 2
3              26-35         18.562488                 5                 2
4              18-25         32.509794                 6                 2
…                …               …                   …                 …
49995          26-35         17.733346                 5                 1
49996          18-25         30.494072                 8                 3
49997          26-35         34.111409                 9                 3
49998          26-35         23.129974                 6                 2
49999          36-50         24.741292                 6                 2

[50000 rows x 4 columns]>
```

сделала для разнообразия в ответах, для сегмента младше 18 и от 18 до 25 - одинаковые значения.

[94]:
```
grouped = dataset.groupby('age_category').agg({
    'session_duration': 'mean',
    'tracks_listened': 'mean',
    'unique_artists': 'mean'
}).reset_index()
```

[95]:
```
grouped
```

[95]:

	age_category	session_duration	tracks_listened	unique_artists
0	18-25	20.076631	4.648459	1.761358
1	26-35	24.969298	5.865290	2.272452
2	36-50	21.995551	5.104548	1.910657
3	50+	18.068767	4.102873	1.506731
4	Under 18	20.083561	4.639586	1.748175

[96]:
```
age_categories = dataset['age_category'].unique()
def perform_tests(metric):
    results = []
    for cat1, cat2 in itertools.combinations(age_categories, 2):
        data1 = dataset[dataset['age_category'] == cat1][metric]
        data2 = dataset[dataset['age_category'] == cat2][metric]
```

```
        t_stat, t_pval = stats.ttest_ind(data1, data2,
    ↪equal_var=False)
        mw_stat, mw_pval = stats.mannwhitneyu(data1, data2)
        f_stat, p_value_f = stats.f_oneway(data1, data2)
        results.append({
            'Category 1': cat1,
            'Category 2': cat2,
            'T-test p-value': t_pval,
            'Mann-Whitney p-value': mw_pval,
            'Fishera-test': p_value_f
        })
    return pd.DataFrame(results)
```

[97]:
```
test_results_duration = perform_tests('session_duration')
groups = [dataset[dataset['age_category'] ==
  ↪category]['session_duration'] for category in age_categories]
f_statistic, p_value_duration = stats.f_oneway(*groups)
```

[98]:
```
test_results_duration
```

[98]:
```
   Category 1 Category 2  T-test p-value  Mann-Whitney p-value
  ↪Fishera-test
0      18-25        50+   2.038849e-111          1.502118e-89   4.
  ↪036598e-79
1      18-25      26-35    0.000000e+00          0.000000e+00   0.
  ↪000000e+00
2      18-25   Under 18    9.422067e-01          8.802920e-01   9.
  ↪424994e-01
3      18-25      36-50   2.106150e-105          7.087494e-94   7.
  ↪487996e-94
4        50+      26-35    0.000000e+00          0.000000e+00   0.
  ↪000000e+00
5        50+   Under 18    1.749236e-78          6.050843e-70   1.
  ↪988027e-69
6        50+      36-50    0.000000e+00         1.918205e-291  2.
  ↪731807e-301
7      26-35   Under 18    0.000000e+00          0.000000e+00   0.
  ↪000000e+00
8      26-35      36-50   2.738365e-189         7.135505e-163  4.
  ↪026967e-164
9    Under 18     36-50    2.485399e-72          3.138115e-70   3.
  ↪280330e-72
```

[99]:
```
p_value_duration
```

[99]:
```
0.0
```

[100]:
```
test_results_tracks_listened = perform_tests('tracks_listened')
groups = [dataset[dataset['age_category'] ==
  ↪category]['tracks_listened'] for category in age_categories]
f_statistic, p_value_tracks_listened = stats.f_oneway(*groups)
test_results_tracks_listened
```

[100]:
```
  Category 1 Category 2  T-test p-value  Mann-Whitney p-value
  ↪Fishera-test
0      18-25        50+    8.897915e-99          7.025519e-72   1.
  ↪998355e-74
1      18-25      26-35    0.000000e+00          0.000000e+00   0.
  ↪000000e+00
2      18-25   Under 18    7.395225e-01          5.715216e-01   7.
  ↪403371e-01
3      18-25      36-50    1.734903e-71          2.266231e-70   5.
  ↪860888e-67
4        50+      26-35    0.000000e+00          0.000000e+00   0.
  ↪000000e+00
5        50+   Under 18    3.128023e-68          5.592078e-53   1.
  ↪472841e-61
6        50+      36-50   1.566752e-242         6.940556e-216  1.
  ↪306042e-226
7      26-35   Under 18    0.000000e+00          0.000000e+00  1.
  ↪017775e-321
8      26-35      36-50   4.124285e-149         1.541054e-125  1.
  ↪421829e-132
9    Under 18     36-50    1.776915e-52          4.249919e-55   2.
  ↪007249e-52
```

[101]:
```
p_value_tracks_listened
```

[101]:
```
0.0
```

[102]:
```
test_results_unique_artists = perform_tests('unique_artists')
groups = [dataset[dataset['age_category'] ==
  ↪category]['unique_artists'] for category in age_categories]
f_statistic, p_value_unique_artists = stats.f_oneway(*groups)
test_results_unique_artists
```

[102]:
```
  Category 1 Category 2  T-test p-value  Mann-Whitney p-value
  ↪Fishera-test
0      18-25        50+    1.909086e-93          5.633800e-65   1.
  ↪802860e-72
1      18-25      26-35    0.000000e+00          0.000000e+00   0.
  ↪000000e+00
```

```
2      18-25   Under 18    2.915077e-01          4.344325e-01   2.
↪948116e-01
3      18-25      36-50    1.021367e-30          7.204620e-40   2.
↪382523e-31
4        50+      26-35    0.000000e+00          0.000000e+00   0.
↪000000e+00
5        50+   Under 18    2.296746e-61          3.996138e-48   3.
↪865026e-56
6        50+      36-50   2.609988e-157         1.912131e-140  3.
↪255920e-142
7      26-35   Under 18   2.793660e-268         1.517297e-237  7.
↪123556e-238
8      26-35      36-50   7.134213e-125          6.024025e-98  2.
↪248060e-112
9    Under 18      36-50    5.232480e-27          1.899402e-32   5.
↪062539e-27
```

[103]:
```
p_value_unique_artists
```

[103]:
```
0.0
```

После разбиения на 5 сегментов (по возрасту) были посчитаны тесты Стьюдента, Манна-Уитни и Фишера для величин длительность сессии, кол-во прослушанных треков и кол-во уникальных исполнителей.

Были построенны следующие гипотезы: 1) Тест Стьюдента H0 гипотеза - равенство средних категории_1 и категории_2 H1 гипотеза - неравенство средних категории_1 и категории_2

ИТОГ H0 гипотеза не принимается, так как P_value < 0.05 для всех категорий кроме случая рассмотрения категории младше 18 и от 18 до 25.

2) Тест Манна-Уитни H0 гипотеза - равенство распределений категории_1 и категории_2 H1 гипотеза - неравенство распределений категории_1 и категории_2

ИТОГ H0 гипотеза не принимается, так как P_value < 0.05 для всех категорий кроме случая рассмотрения категории младше 18 и от 18 до 25.

3) Тест Фишера H0 гипотеза - нет статистически значимых различий между возрастными группами по категории_i (длительность сессии/кол-во прослушанных треков/ кол-во уникальных исполнителей) H1 гипотеза - есть статистически значимых различий между возрастными группами по категории_i

ИТОГ H0 гипотеза не принимается, так как P_value < 0.05 для всех категорий кроме случая рассмотрения категории младше 18 и от 18 до 25.

также сразу в таблицах был выполнен пункт 3 задания, проведенны попарные сравнения через тест Стьюдента и тест Фишера. Сложно сделать вывод, так как большинство значений лежат около нуля, так как данные искусственные. Однако,

для категории младше 18 и от 18 до 25 значения теста Стьюдента и теста Фишера очень близки. Если результаты тестов совпадают или близки, это обычно означает, что гипотеза, проверяемая в обоих тестах (например, о равенстве средних или независимости), не отвергается, и данные не показывают значительных различий или зависимостей.

[115]:
```
from sklearn.utils import resample
def diff_mean_conf_interval(data1, data2, confidence=0.95):
    mean1 = np.mean(data1)
    mean2 = np.mean(data2)
    diff = mean1 - mean2
    std_err1 = np.std(data1, ddof=1) / np.sqrt(len(data1))
    std_err2 = np.std(data2, ddof=1) / np.sqrt(len(data2))
    std_err_diff = np.sqrt(std_err1**2 + std_err2**2)
    ci_lower, ci_upper = stats.t.interval(confidence, len(data1) +
  ↪len(data2) - 2, loc=diff, scale=std_err_diff)
    return ci_lower, ci_upper

def efron_diff_mean_conf_interval(data1, data2, n_iterations=1000,
  ↪confidence=0.95):
    diffs = []
    for _ in range(n_iterations):
        sample1 = resample(data1, n_samples=len(data1))
        sample2 = resample(data2, n_samples=len(data2))
        mean1 = np.mean(sample1)
        mean2 = np.mean(sample2)
        diffs.append(mean1 - mean2)

    lower = np.percentile(diffs, (1-confidence)/2*100)
    upper = np.percentile(diffs, (1+confidence)/2*100)
    return lower, upper
metrics = ['session_duration', 'tracks_listened', 'unique_artists']
```

[126]:
```
results = []
for metric in metrics:
    for i, category1 in enumerate(age_categories):
        for category2 in age_categories[i+1:]:
            data1 = dataset[dataset['age_category'] ==
  ↪category1][metric]
            data2 = dataset[dataset['age_category'] ==
  ↪category2][metric]

            t_stat, p_value = stats.ttest_ind(data1, data2)
            ci_lower_exact, ci_upper_exact =
  ↪diff_mean_conf_interval(data1, data2)
            ci_lower_efron, ci_upper_efron =
  ↪efron_diff_mean_conf_interval(data1, data2)
```

```
            ttest_significant = p_value < 0.05
            ci_exact_significant = ci_lower_exact > 0 or
 ↪ci_upper_exact < 0
            ci_efron_significant = ci_lower_efron > 0 or
 ↪ci_upper_efron < 0

            results.append({
                'metric': metric,
                'category1': category1,
                'category2': category2,
                'p_value_ttest': round(p_value,3),
                'ci_lower_exact': ci_lower_exact,
                'ci_upper_exact': ci_upper_exact,
                'ci_lower_efron': ci_lower_efron,
                'ci_upper_efron': ci_upper_efron,
                'ttest_significant': ttest_significant,
                'ci_exact_significant': ci_exact_significant,
                'ci_efron_significant': ci_efron_significant
            })

results_df = pd.DataFrame(results)
```

[127]:
```
results_df.to_csv('confidence_intervals_and_tests.csv', index=False)

results_df
```

[127]:
```
              metric category1 category2  p_value_ttest
 ↪ci_lower_exact  \
0   session_duration     18-25       50+          0.000        1.
 ↪834393
1   session_duration     18-25     26-35          0.000       -5.
 ↪067979
2   session_duration     18-25  Under 18          0.942       -0.
 ↪194299
3   session_duration     18-25     36-50          0.000       -2.
 ↪090152
4   session_duration       50+     26-35          0.000       -7.
 ↪098866
5   session_duration       50+  Under 18          0.000       -2.
 ↪223868
6   session_duration       50+     36-50          0.000       -4.
 ↪121526
7   session_duration     26-35  Under 18          0.000        4.
 ↪675140
8   session_duration     26-35     36-50          0.000        2.
 ↪777370
```

```
9   session_duration  Under 18     36-50          0.000       -2.
  ↪119207
10   tracks_listened     18-25       50+          0.000        0.
  ↪495434
11   tracks_listened     18-25     26-35          0.000       -1.
  ↪266000
12   tracks_listened     18-25  Under 18          0.740       -0.
  ↪043434
13   tracks_listened     18-25     36-50          0.000       -0.
  ↪505828
14   tracks_listened       50+     26-35          0.000       -1.
  ↪819589
15   tracks_listened       50+  Under 18          0.000       -0.
  ↪596607
16   tracks_listened       50+     36-50          0.000       -1.
  ↪059339
17   tracks_listened     26-35  Under 18          0.000        1.
  ↪166631
18   tracks_listened     26-35     36-50          0.000        0.
  ↪703931
19   tracks_listened  Under 18     36-50          0.000       -0.
  ↪524510
20    unique_artists     18-25       50+          0.000        0.
  ↪230542
21    unique_artists     18-25     26-35          0.000       -0.
  ↪535673
22    unique_artists     18-25  Under 18          0.295       -0.
  ↪011313
23    unique_artists     18-25     36-50          0.000       -0.
  ↪174634
24    unique_artists       50+     26-35          0.000       -0.
  ↪794263
25    unique_artists       50+  Under 18          0.000       -0.
  ↪269915
26    unique_artists       50+     36-50          0.000       -0.
  ↪433122
27    unique_artists     26-35  Under 18          0.000        0.
  ↪495387
28    unique_artists     26-35     36-50          0.000        0.
  ↪332190
29    unique_artists  Under 18     36-50          0.000       -0.
  ↪192019

    ci_upper_exact  ci_lower_efron  ci_upper_efron  ttest_significant
  ↪ \
```

0	2.181335	1.835921	2.181930	True
1	-4.717357	-5.069568	-4.717398	True
2	0.180438	-0.198750	0.181646	False
3	-1.747689	-2.093590	-1.761496	True
4	-6.702197	-7.092068	-6.692454	True
5	-1.805720	-2.219697	-1.808204	True
6	-3.732042	-4.107231	-3.741771	True
7	5.096335	4.674594	5.097266	True
8	3.170125	2.787894	3.160366	True
9	-1.704774	-2.130752	-1.710571	True
10	0.595737	0.497791	0.594789	True
11	-1.167661	-1.267065	-1.170255	True
12	0.061180	-0.041056	0.063822	False
13	-0.406350	-0.504457	-0.403418	True
14	-1.705243	-1.822716	-1.704279	True
15	-0.476818	-0.598926	-0.479251	True
16	-0.944010	-1.054768	-0.939550	True
17	1.284776	1.168447	1.283854	True
18	0.817553	0.703050	0.812090	True
19	-0.405414	-0.524447	-0.403784	True
20	0.278713	0.229697	0.280748	True
21	-0.486514	-0.534873	-0.486032	True
22	0.037681	-0.011202	0.037719	False
23	-0.123962	-0.174972	-0.124484	True
24	-0.737179	-0.793613	-0.737859	True
25	-0.212972	-0.270691	-0.211653	True
26	-0.374729	-0.430998	-0.372642	True
27	0.553167	0.495967	0.554725	True
28	0.391400	0.332478	0.390359	True
29	-0.132945	-0.190201	-0.132708	True

	ci_exact_significant	ci_efron_significant
0	True	True
1	True	True
2	False	False
3	True	True
4	True	True
5	True	True
6	True	True
7	True	True
8	True	True
9	True	True
10	True	True
11	True	True
12	False	False
13	True	True
14	True	True

15	True	True
16	True	True
17	True	True
18	True	True
19	True	True
20	True	True
21	True	True
22	False	False
23	True	True
24	True	True
25	True	True
26	True	True
27	True	True
28	True	True
29	True	True

Были построенны точные доверительные интервалы и интервалы по методу Эфрона для каждой пары. Также была построена и проверена гипотеза H0 - соответствующие интервалы перекрывают друг други и указывает на отсутсвие значимых различий, H1 - соотвествующие интервалы не перекрывают друг друга и указывает на статистически значимую разницу между сегментами.

Анализ показал (тест Стьюдента), что только группы младше 18 и от 18 до 25 H0 принимается гипотеза (т.к. p_value >= 0.05), во всех случает отвергается (данная логика верна для всех метрик)

Отрицательные значения в доверительных интервалах говорят о том, что в одной из категорий среднее имеет меньшее значение.

5. Что касается использования других стат. тестов. Думаю, что можно было использовать корреляционный анализ (Пирсона) для изучения линейной зависимости между двумя переменными или Спирмена для проверки монотонной связи между двумя количественными переменными. Так же, если данные имеют распределение не нормальное, то тест Уилкоксона для парных выборок для сравнения двух зависимых выборок. Аналогично тест Колмогороа-Смирнова для проверки гипотезы о том, что выборка принадлежит какому-то теоретическому распределению и другие стат. тесты.