Small Exercise

Pietro della Briotta Parolo

Data Processing

The first thing I did was to convert the csv data into a sqlite database, which is easier to handle. This was done in python simply by reading the csv file and adding each row into the newly created *events* table that I had previously created. Furthermore, I decided to add an index for each *uid* in order to be able to perform user based searches faster.

Question 1

Calculate the average number of song/minigame_plays before a user subscribes.
3.55 or 4.33 according to whether users who create an account without playing songs or minigames are included.

In order to obtain this result I needed a subscription time stamp for each user that subscribed. This was easily done with an SQL query:

Code Example 1: Generate subscription table

```
CREATE TABLE subscribed AS
SELECT uid,date
FROM events
WHERE event = 'subscribed'
```

This allows me to have a time stamp to be used in order to check which uid events took place before the user subscribed. The next step therefore was to return the number of pre-subscription events.

Code Example 2: Return pre-sub events

```
SELECT COUNT(*)
FROM subscribed
LEFT JOIN events
USING (uid)
WHERE events.date < subscribed.date
AND  event in ('minigame_played','song_played')
```

The script returns 2146 events, which divided by the number of subscriptions (605) gives me **3.55** songs or minigames on average before subscription.
A more elegant and direct way to solve the issue is to use SQL directly. This can be achieved by counting the events for each uid and then averaging.

Code Example 3: Return SQL average

```
SELECT AVG(cnt)
FROM (
SELECT count(*) AS cnt
FROM subscribed
LEFT JOIN events USING (uid)
WHERE (events.date < subscribed.date)
AND event in ('minigame_played','song_played')
GROUP BY (uid)
)
```

The result, for this implementation is different: **4.33**. This is because in this way of formulating the query the users who subscribed directly without first playing songs or games are not counted in the statistics. In order to obtain the same result as before however, one can use a simple trick to force the LEFT JOIN to return at least one event: the subscription event, which can later be discarded.

Code Example 4: Return SQL average - Fixed

```
SELECT AVG(cnt)
FROM (
SELECT count(*) -1  AS cnt
FROM subscribed
LEFT JOIN events USING (uid)
WHERE (events.date <= subscribed.date)
AND event in ('minigame_played','song_played','subscribed')
GROUP BY (uid) )
```

This method returns again an average of **3.55**.

Question 2

Does the group B have a better conversion rate between account created and the first song played than the control group?
No. Group B has a higher conversion rate, but it is statistically not significant. However, more data could change the picture.

In order to answer this question I started by doing a similar job as in code example 2, thus creating a table that stores the time stamp for the account creation. This allows me to check for individual uid activity after account creation. Similarly again, the SQL search is very similar to the the previous question, where in this case I filter for *song_played* events after the account creation.

Code Example 5: Return conversion rates

```
SELECT COUNT(DISTINCT(uid))
FROM account
LEFT JOIN events USING (uid)
WHERE events.date > account.date
AND event = 'song_played'
AND groupName = (?)
```

Table 1 shows the data about the two groups. While the numbers are similar, suggesting no significant difference between the two groups, one needs to take into account the sample size and perform further statistical test. Also, I took a look at whether it's likely that the users who have not played a song yet might do it later. I noticed that the maximum number of days between account creation and first song played is ≈ 12, with the mean being of 1. However, the most recent account creation without a song played is 21 days old and the longest any of those users has used the app after account creation is 4 days. We

Table 1: Conversion Rates for Groups A and B

Group	**Accounts Created**	**Songs Played**	**Conversion Rates**
A	332	326	98.193%
B	332	330	99.398%

can therefore safely assume, that the missing users are not likely to play a song anytime soon.
In this case we want to test the null hypothesis $p_a = p_b$ which means that both samples are taken from the same binomial distribution having identical success rate by performing a *z test*

$$z = \frac{p_a - p_b}{\sqrt{p(1-p)(\frac{1}{n_a} + \frac{1}{n_b})}} \tag{1}$$

where $n_a = n_b = 332$ is the sample size for each group and $p = \frac{n_a p_a + n_b p_b}{(n_a + n_b)}$ is the weighed average of the two rates, which in this case is just the arithmetic mean since the two sample groups are identical in size: $p = 0.98795$. The tests returns a z score of 1.42, which gives us a $\approx 8\%$ probability for the results of group B to be generated from a distribution with higher conversion rate. While this value is low, it's usually not considered low enough to reach a definite conclusion. A z score of 1.65 would give a 5% probability, which is often considered a reliable indicator. In this case the problem seems to lie in the high conversion rate of the control group, which requires group B to be pretty much "perfect" in its conversion rate. However, it is important to notice that the previous method is based on the assumption that the binomial distribution can be approximated with a Gaussian one. Such approximation increases in validity the higher the sample size and the closer the probability is to 1/2. In our case we do have a significant sample size, but the high conversion rate limits the validity of the approximation.
Another way to tackle the problem is to take the control group as a reference point to see whether its statistical properties would be able to justify the performance of group B. This can be simply done with a binomial test, as shown in code example 6. In this case, the probability of obtaining 330 or more successes with the conversion rate of the control group is $\approx 8\%$, coherently with what discussed before.

Code Example 6: Binomial Test

```
scipy.stats.binom_test(330,332,326./332,alternative='greater')
0.079568660226078308
```

It's also important to note that had the successes been just one more, this binomial test would have shown statistical significance for the two conversion rates being different. The same would have happened with a sample size of 500 users for group B with the same conversion rate. This means that even though the current data cannot show an out-performance by group B, the result is on the ”edge” of statistical signficance and further data should be collected in order to have stronger statistical results.

NB. I found it unclear from the assignment whether the conversion rate meant the probability of playing a song after creating an account or the opposite, that is the probability of opening an account after playing the first song. I also checked the latter scenario, which however turned out to be statistically irrelevant. No user has opened an account *after* playing a song. There are only 4 users who did play a song and did not create an account. However, in this case, the conversion rate is 0% and the problem cannot be handled statistically.

Question 3

In your opinion, is the experiment of the group C bringing more money to the company? Please explain why.
Yes. More than 60% of the subscribed users in both groups are likely to remain subscribed indefinitely, allowing the higher weekly cost of group C to counterbalance the higher short-term unsubscription costs of group B.

Table 2 shows the numbers of subscribed and unsubscribed users for each group, along with their rates (the unsubscription rates are based on the number of subscribed users, of course).

Table 2: Subscriptions and Unsubscriptions for Groups B and C

Group	**Subscribed**		**Unsubscribed**	
B	199	59.76%	73	36.68%
C	205	61.56%	76	37.07%

With these numbers we can check whether the subscription rates are statistically significantly different, similarly to done in the previous question. The z-score for the hypothesis that the two subscription distributions are different is 0.475, implying that there is no significant statistical evidence for one group being more likely to have subscriptions over the other. The same result applies to the unsubscription rate, which has a z score of 0.145, also revealing no statistical difference [1].
We can therefore conclude, on a first level of analysis, that the two groups do not appear to perform differently in terms of subscription rates and unsubscription rates. If this pattern is confirmed, the difference in revenue for each group ultimately depends on how long the users remain subscribed. In fact, group C

[1] In this scenario the z-test can be safely used, since the probabilities are less extreme

has a higher price on a weekly basis (5,6€ vs 5€), which is however counterbalanced by the fact that an early unsubscription during the week will be less profitable compared to the unsubscriptions for group B. If we assume that there is no preferred day of unsubscription in the 7 days interval, group B will give a $\frac{\sum_{i=1}^{7} 5-0.8i}{7} = 1.8$€ average extra revenue per user due to the asymmetry in the payments. Given also the 0.6€ weekly extra revenue for group C, we can therefore conclude that if the similarity in subscription and unsubscription rates is confirmed, the daily payment type becomes more profitable only starting from the beginning of the fourth week. Is it the case with our data?

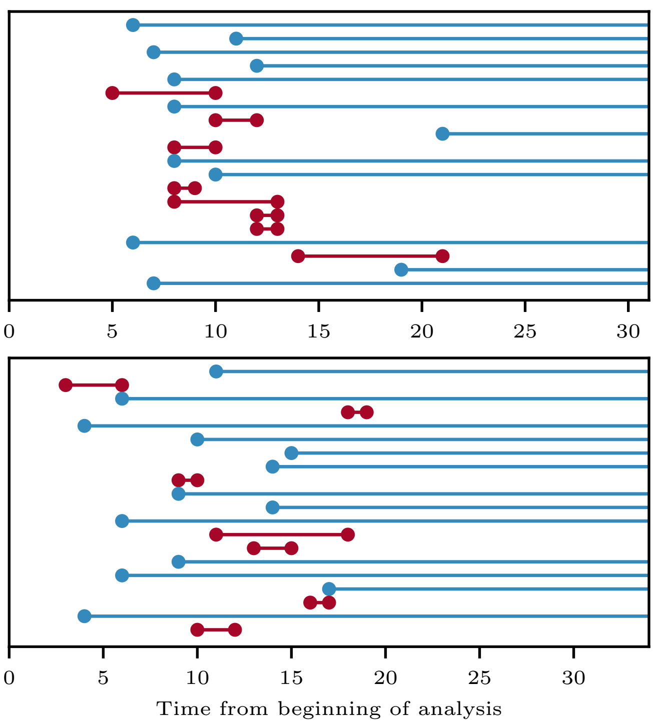

Figure 1: Example of lifetimes for 40 randomly selected users from group B (top) and group C (bottom). Red lines correspond to users who have unsubscribed, while blue lines correspond to users who haven't.

This problem is linked to the known problem of *survival analysis*, which aims at estimating "death times" where deaths represent the withdrawal of members from a certain group, in this case from the subscribed group. In our scenario, we are also facing the issue of censorship, i.e. the presence of users whose death might not have been observed yet. In Fig.1 we can see practically what that

means: while the red lines show a birth (subscription) and a death (unsubscription), the blue lines do have a birth but are still missing a death time.
The function that gives the probability of an user to survive time T is called survival function and it can be estimated with a non-parametric statistic called the Kaplan-Meier estimator, which is defined as:

$$\hat{S}(T) = \prod_{i:t_i<T} \left(1 - \frac{d_i}{n_i}\right) \tag{2}$$

What the estimator does is calculate the probability of survival by multiplying the death probabilities in each previous time interval, which are calculated using the number of deaths n_i over the alive population d_i. The important aspect of this estimator is that at the same time it does not require information on the actual functional form of the survival function and it also allows to approximate its variance. A similarly related concept is the hazard function, which measures the instantaneous failure rate for a single user: $\lambda(t) = \frac{-S'(t)}{S(t)}$. This is in turn allows to calculate the Nelson-Aalen estimator, which estimates the cumulative function of $\lambda(t)$,that is the sum of all the risk of failure that users have experienced until time T:

$$\hat{\Lambda}(T) = \sum_{i:t_i<T} \frac{d_i}{n_i} \tag{3}$$

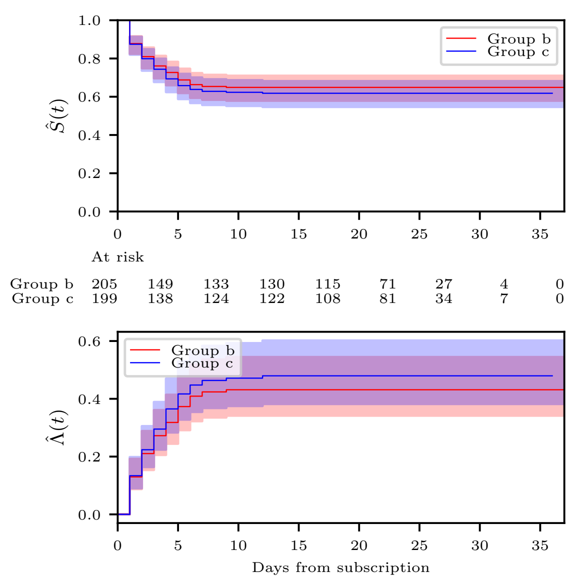

Figure 2: Survival function (top) and cumulative hazard function (bottom). The shaded area corresponds to a 95% confidence interval.

Fig.2 shows the fitting of both estimators, along with the errors. We get two important pieces of information from this plots:

- Each group's curve lies within the confidence interval of the other group's curve, thus indicating that there is no difference in unsubscription dynamics between the two groups. In code example 7 we can also see the result of a statistical test that compares the survival curves of the two groups, showing that there is no significant statistical difference.
- Users either unsubscribe within an initial period of time of roughly 2 weeks or else are likely to keep their subscription indefinitely. Statistically speaking, the absence of deaths in the $t \in [16, 36]$ range is a strong indicator that users of both groups are extremely unlikely to randomly unsubscribe after the first two week period.

Code Example 7: LogRank test

```
results = logrank_test(b_life ,c_life ,b_events ,c_events , alpha=.99)
results.print_summary()
Results
   df: 1
   alpha: 0.99
   t 0: -1s
   test: logrank
   null distribution: chi squared
    p-value | test statistic |  test result        |  is significant
    0.53285 |          0.389  | Cannot Reject Null |        False
```

To summarize we have seen that:

- There is no statistical difference between the subscription and unsubscription rates in the two groups.
- The profitability of one group over the other depends on how long the users remain within the system.
- The critical time after which group C statistically is more profitable than group B is of 3 weeks.
- There is no statistical difference between the unsubscription dynamics in the two groups.
- Users most likely either unsubscribe in the first 14 days or else will remain subscribed indefinitely.

We can thus conclude that, since the payment option of group C does not influence the subscription/unsubscription rates or dynamics, its higher weekly cost will eventually make it more profitable.