Part 1 – SQL Syntax

1)

```
SELECT cities.city_name,PERCENTILE_CONT(0.90) WITHIN GROUP (ORDER BY trips.predicted_eta-
trips.actual_eta)

FROM trips INNER JOIN cities ON trips.city_id = cities.city_id

WHERE (trips.status='completed') AND cities.city_name in ('Qarth','Meereen')

AND  trips.request_at >= DATEADD(day, -30, GETDATE())

GROUP BY cities.city_name;
```

2)

```
SELECT cities.city_name,trips.request_at,percentage

FROM  trips INNER JOIN events ON trips.client_id = events.rider_id INNER JOIN cities ON

trips.city_id = cities.city_id

WHERE ((DATEPART(DAY,trips.request_at) BETWEEN (01 AND 07)) AND
((DATEPART(MONTH,trips.request_at) IS 01)) AS first_week

AND (count(events.event_name='sign_up_success')/count(events.event_name))*100 AS percentage

AND DATEDIFF(HOUR,events._ts,trips.request_at)<=168

AND cities.city_name IN ('Qarth','Meereen')

ORDER BY first_week

GROUP BY cities.city_name;
```

Part 2 – Experiment and metrics design

Since it's a new feature to the UBER partner app and also a major change, we can effectively make use of A/B testing to perform some experimentation to know if the new improvements to the already existing app is working or not.

Since we use A/B testing, we can take 2 sets of users and show an old version of the app to one set (control set) and then the newer version to a different set (experiment set). And we evaluate by seeing how these 2 sets of users respond differently to determine which version is better.

User flow (Funnel analysis)

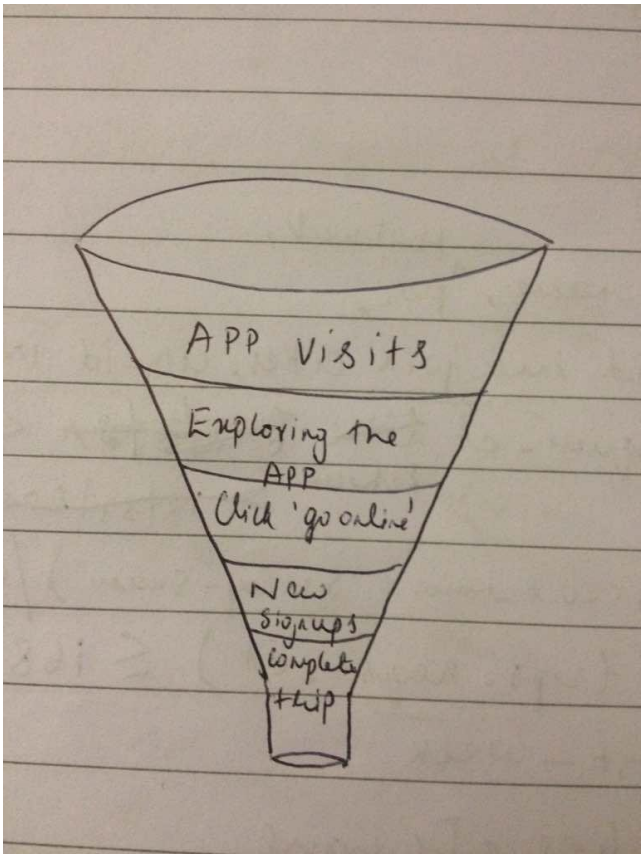
1) Launch the UBER partner app (App visits)

2) Explore the app by navigating to different pages like earnings, rating etc.

3) Signed-up partners click on 'go online' and start making trips

4) Set of new users create account and signup through the app

5) Partners increase their trip count



Experimental change: Redesign of UBER partner app (both UI and new features)

Hypothesis: Redesigning UBER partner app will increase the click through probability of the 'go online' and 'signup' buttons which will ultimately increase the total number of rides and hence the overall business for UBER.

For the above hypothesis, we need a metric to measure this change

Possible success metrics are:

1) Increase in the Click through probability (unique visitors who click 'go online' / unique visitors who load the app) for ('go online' button) - eventually when drivers explore the app and go

online, they take trips and eventually total trip count increases. Hence one level of the funnel will have a positive impact on the end of the funnel as well. Additionally, we can define below metrics which helps in monitoring this main metric

   a. Latency – app load time
   b. Sensitivity – What results are we happy with

2) Increase in the click through probability for 'sign up' button. Total number of new signups

Since we target to measure the total impact of the newer version of the app and not just the UI, click through probability should be a good choice.

Below are few of the possible testing plans:

Since we have agreed on click through probability, a plan might have to be set up to adopt that on every page view of the app, we capture the event and whenever driver clicks, we also capture click events. Once the data is captured, we just sum the page views, we sum the clicks and divide. For the probability, we must match each page view with all the child clicks, so that we count at most one child click per page view.

We can use hypothesis testing or inference, a quantitative way to establish how likely is it that our results occurred by chance. So first we need a null hypothesis or a baseline. In our case that there is no difference in click through probability between our control group (p(cont) and experiment p(exp)). Second is alternative hypothesis. In our case, if click through probability is different or lower or higher.

Both control and experiment groups would follow a binomial distribution (clicked or not clicked), but the probability might be different for each group. We can formally define our plan with respect to hypothesis as:

Null hypothesis H0: p(cont) - p(exp) = 0

Alternative hypothesis H1: p(cont) - p(cont) !=0

W–e can estimate p(cont) and p(exp) from the data we collect. Then we calculate the difference between these and compute the probability that this difference would have come by chance if the null hypothesis were true. Then we want to reject the null hypothesis and conclude that our experiment has an effect if this probability is small enough. By the reference of statistics confidence interval, we can choose the value of 0.05. Hence if the probability is less than alpha value 0.05, we will reject null hypothesis.

For the confidence intervals, we need to compare the proportion of clicks estimated on the control side with the proportion estimated on the experiment side. Then the quantitative task tells us whether it's likely that the results we got, the difference we observed, could have occurred by chance, or if it be extremely unlikely to have occurred if the two sides were the same.

Since we have 2 groups, a control and experiment, we need pooled standard error to calculate the p-value.

pooled standard error:

X(cont) -> users who click in control group

X(exp) -> users who click in the experiment group

N(cont) -> Total number of users in control group

N(exp) -> Total number of users in the experiment group

pooled probability of the click (p-bar) = X(xont) + X(exp) / N(cont) + N(exp)

pooled standard error (se-pool) = sqrt(p-bar*(1-p-bar)*(1/N(cont) + 1/N(exp)))

difference (d) = p(exp) - p(cont)

under null H0: d=0

let's say we find 95% confidence interval, then d>1.96*se-pool or d<-1.96*se-pool, then we can reject null and say the difference we got is statistically significant difference.

Next plan is from a business perspective, we want to know what change in the click through probability is practically significant (what size change matter to us – 1%,2% etc). Hence, we might decide > 2% of the change in click through probability is good enough for our decision making. We need to make sure that practically significant figure also satisfies statistical significance.

We need to decide on statistical power which is, how many page views we need to collect to get a statistically significant result.

So, for example, let's choose N=1000 view the app, X=100 who click, practical significance level = 2%, alpha=0.05, sensitivity=0.2, by calculations we need at least 3600-page view per each control group.

Translating results which we have from testing plan to decision making can be explained as below:

Let's assume few numbers here to explain whether UBER must launch their new version of the app or not assume

N(cont)=10072 page views

N(exp)=9886 page views (numbers not same because people were randomly assigned to groups)

X(cont)=974 users who clicked 'go online' button,

X(exp)=1242 (more people clicked in the experimental group)

Let's calculate p-bar = (974+1242)/(10072+9886) = 0.111

se-pool = sqrt(0.111*(1-0.111)*(1/10072)+(1/9886)) = 0.00445

Let's consider 95% confidence interval and practical significance level of 2%(0.02).

d = 0.0289

margin of error m = se-pool * 1096 = 0.0087

lower bound of confidence interval = d-m = 0.0202

upper bound of confidence interval = d+m = 0.0376

We can conclude from the experiment that it is highly probable that click through probability changed at least by 2%. So, we have both results for statistical significance and practical significance. Based on these results, UBER would want to definitely launch the new version of the app. If the statistical results are good enough and it does not match our practical significance level, we might have to do some additional tests to come to a decision.

Part 3 – Data analysis

(complete python program is part of the answers folder with the name 'problem_3.ipynb' and also a pdf version of iPython notebook file is part of the answers folder)

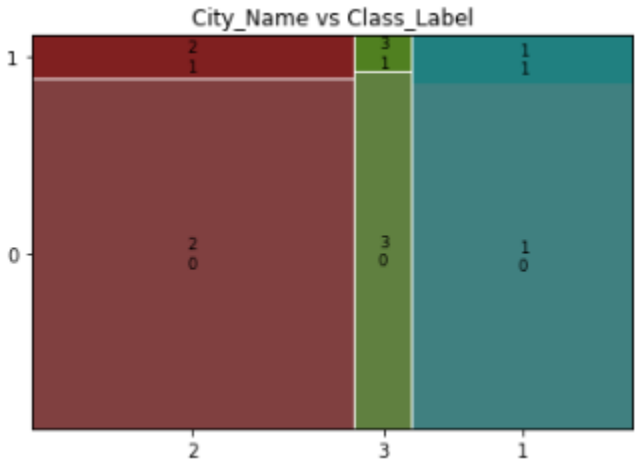
1) It was an interesting dataset to explore. There were many fields which had an obvious information and many others where the information was hidden. Analyzing which signed up driver actually starts driving is definitely a very useful method for UBER to improve their business. With this data, we can build a very good model to predict the future signup-drive conversion rate. In the dataset provided, we had 54681 rows of data and 11 predictors.

To perform Data Analysis, Data Visualization and Machine Learning, I am using various different libraries such as NumPy, Pandas, Matplotlib, Scikit-Learn etc. Numpy helps us to perform computations on data easily and Pandas helps us to do thorough Data Analysis. Matplotlib provides intense libraries which helps us in Data Visualization. We can build a predictive model using sklearn.

First step is to store our data which is present in "uber_data.csv" to Pandas Dataframe named rides. Next is to explore the data to see if there are any missing values and other un- recognized patterns. As part of **Data Cleaning**, we need to replace/exchange/manipulate the missing values in our dataset. The class label (1 – driver starts driving and 0- he doesn't start) is not provided in the dataset and hence we need to generate them. However, we have 'first_completed_day' column which could be used to create our class labels. 'first_completed_day' had a lot of missing values which had to be filled up with zero before converting it to class labels. A function which creates a label of 1 or 0 for each row along with ".apply" method gives us a new column 'class_label'. There were a lot of missing values in 'vehicle_added_date', 'bgc_date' columns. A good idea would be to replace them with its 'signup_date' indicating there was no difference at all after the signup, background check and vehicle was added. The difference between these dates might be one of the potential features for us during model training

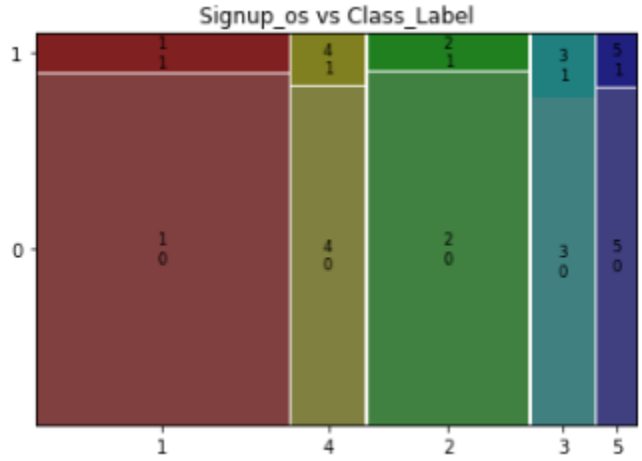
phase. Next there were the date columns with object data types. This had to be converted to pandas datetime type in order to get extract useful time series insights from the data. Next converting all non-numeric columns like 'city_name', 'signup_os' and 'signup_channel' to numeric ones so that it helps in building predictive model. By performing all these changes, our data is now clean and we can now proceed to visualize the data to find useful patterns. For Visualization, I am making use of Mosaic plots which gives an effective comparison between multiple variables. Let's try to visualize our 'class_label' against various parameters like 'city_name', 'signup-os', 'signup_channel', 'signup_week' (extracted from 'signup_date') in order to see if we can get any useful information or trends out of it.

1) **Mosaic plot of 'city_name' versus 'class_label'**



(1 – Berton, 2 – Strark, 3 – Wrouver)
We can see that Strark had a lot of sign-ups but Berton had more signup-ride conversion rate compared to other cities. This means that UBER wants to concentrate on other 2 cities to increase this conversion.

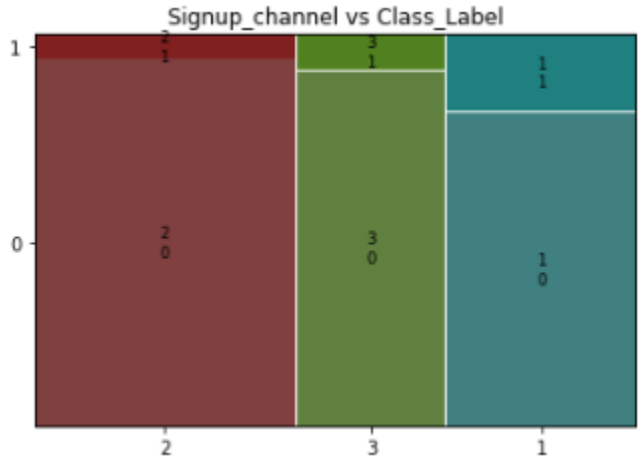
2) **Mosaic plot of 'signup_os' versus 'class_label'**



(1 - ios web, 2 - android web, 3 – mac, 4 – windows, 5 – other)

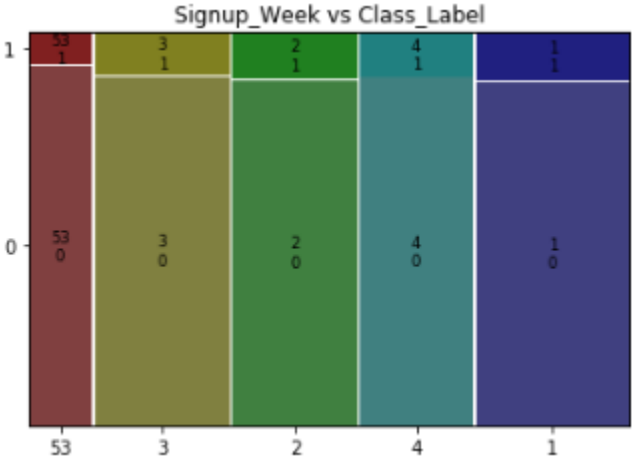
We can clearly see people signed up from ios and android the most. And conversion rate is more or likely not so different. This might give us some useful insights while training our machine learning model.

3) **Mosaic plot of 'signup_channel' versus 'class_label'**



(1 – Referral, 2 – Paid, 3 – Organic)
The above figures help us in a greater way. We can see a lot of signup-ride conversion when signup channel is via referral. Hence UBER might want to concentrate on improving and encouraging more and more people to refer their friends and relatives via some referral programs/rewards and benefits etc.

4) **Mosaic plot of 'signup_week' versus 'class_label'**



The signup week also doesn't give us any critical information.

The fraction of drivers who took first ride = 6137 / 54681 => 0.112 => 11.2%.
only 11.22% of the signed-up drivers start driving!!!

2)**Machine Learning:**
Since this is a classification problem, let us try to build a model using Logistic Regression and Random Forest Classifier using sklearn library. Logistic Regression helps in pulling out those features which has a good correlation with the target variable and Random Forest classifier can pull out the nonlinear relationships from the features. We can also compare their accuracies in parallel to check which model performs better (meaning which model predicts better). To avoid overfitting, I am using the method of cross validation and training the model with train data and testing it with test data. For Logistic Regression, 5 folds of cross-validation is activated and for Random Forest, 3 folds. Finally, after getting the individual accuracies for the respective models, taking the average gives us the mean accuracy. For prediction purpose, initially I am using three main predictors namely "city_name","signup_os","signup_channel". In the later part, I am doing some feature engineering to extract useful insights from the date columns.

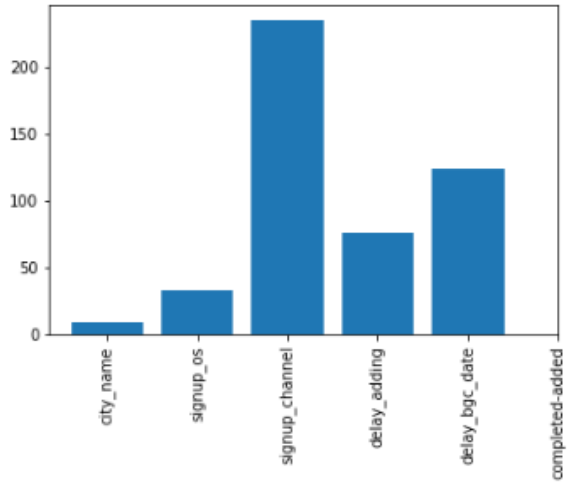
We get a good accuracy score of **88.77%** using the above three predictors which proves that our visualization above gave us the same insights which told us that these three were the attributes which influenced the signup-ride conversion rate more. The Random Forest classifier almost gives us the same accuracy score. Let's try by changing n_estimators and min_sample_split to check if that would increase our prediction rate. The accuracy remains the same of 88.77%.

Feature Engineering:

We have three main date fields namely 'signup_date', 'vehicle_added_date' and 'first_completed_date'. Let's do some feature engineering to get useful insights with this data.

1) **delay_adding** -> After the driver singed up, his/her information has to be approved and added. The difference between these two dates might influence the driver. More the difference, the driver might change his plans and not start driving at all or he might opt for any other UBER competitor like Lyft. Hence let's try to check if this has any effect to our predictions.
2) **delay_bgc_date** -> After the driver singed up, his/her information has to be background checked. Let us check if the difference in these days has any effect on the driver to make his first ride.
3) **completed - added** -> This is the difference in timings between the drivers first ride date and the vehicle added date. This might be one of the factors during our prediction.

Let's not just blindly consider these 3 new features along with the above 3 older ones. Let us try to select the best predictors by feature selection using the p-values. We get the following graph:



Above graph is the proof from our mosaic plot visualization which said ‘signup_channel’ influences our model in a greater way. Next are our new features ‘delay_bgc_date’, ‘delay_adding’ and then signup_os and city_name. ‘compelted-added’ feature has a least influence. Hence let us not consider that for prediction purpose.

Using our new features 'delay_adding','delay_bgc_date' along with old predictors "city_name","signup_os","signup_channel" increased our accuracy to **92.8%**. This shows that our model is performing quite well by predicting almost 93% of the things correctly.

3) Hence this predictive model helps UBER to predict which of their future signed up users would start driving and this model along with above visualizations also give sufficient information like

- Referral scheme – As we saw, referral scheme works a lot here. Hence UBER might enhance the referral model by introducing referral benefits, rewards etc. which will increase more and more drivers to refer their friends, relatives and hence more and more trips

- Which city to target – UBER needs to target Strark and Wrouver more because the conversion rates are less for these cities.

- Not to delay with vehicle added – We saw how delay in vehicle adding effects the conversion rate. During this period, a driver might change his mind and opt out or start with any other UBER competitors.

- Not to delay with background check – This delay might also have a bad effect on the business

These are the suggestions which we extracted from our above model/visualizations to improve UBER’s processes to make more business.