IMS – Inventory / Booking / Performance Forecasting

@ Ads

Agenda

1. Problem statement
2. Concepts
3. Architecture
4. Data structures
5. Algorithm
6. Workflow
7. DL-based pCTR Prediction

Problem statement

- In order to calculate all the inventory for all the combinations for next n days, we must store each combination for next n days in one document. The number of documents is huge (1000 billions level).
- In order to reduce the size of documents, we use TBR concept to store all multiple value attributes' combinations. We assume that all combinations with the same TBR attributes will have the same TBR ratio. This will reduce the number of documents to 10 millions level.
- Even for 10 million documents, to update the booking for million documents in real time is a big challenge.

Agenda

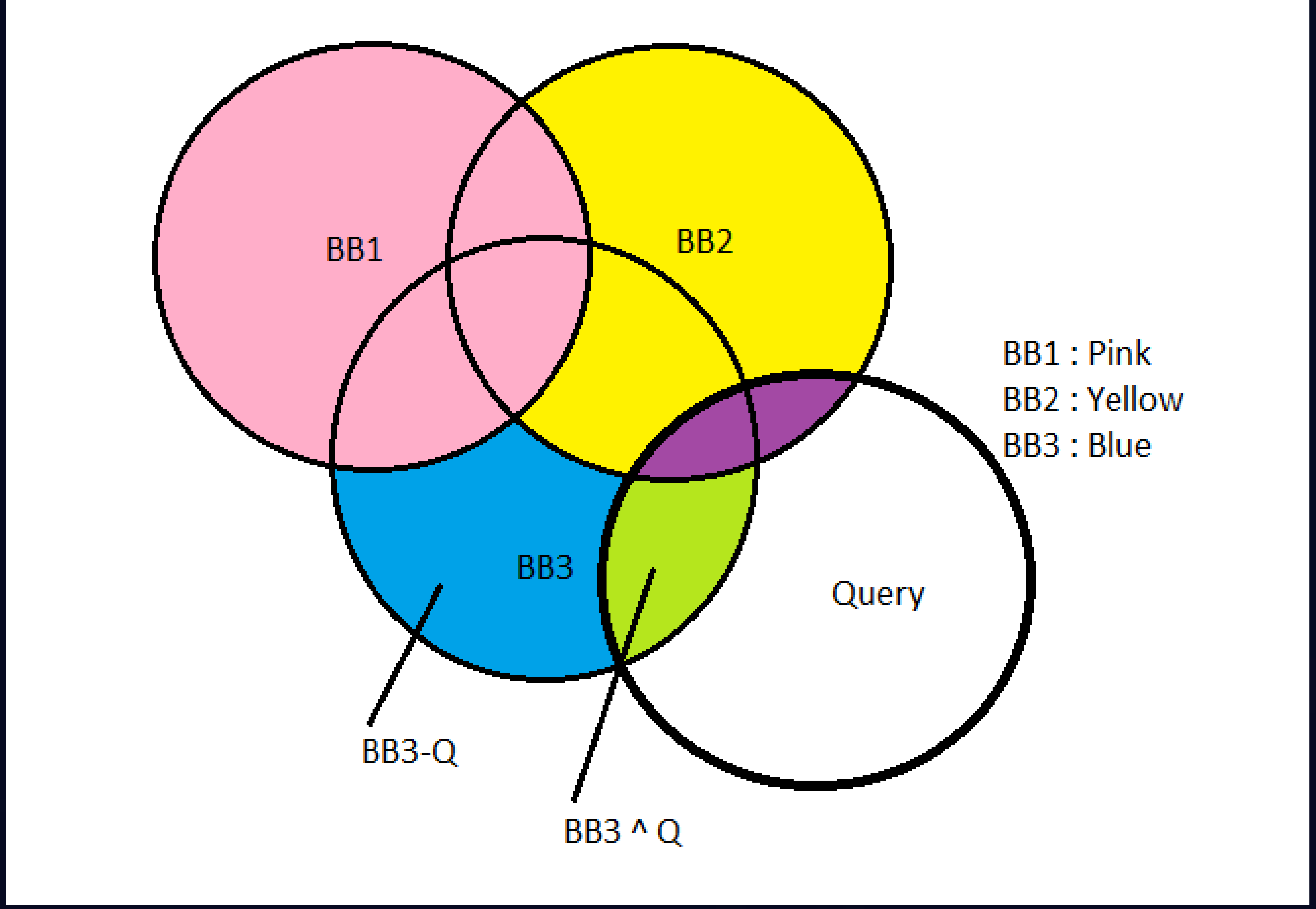
1. Problem statement
2. Concepts
3. Architecture
4. Data structures
5. Algorithm
6. Workflow
7. DL-based pCTR Prediction

Concepts

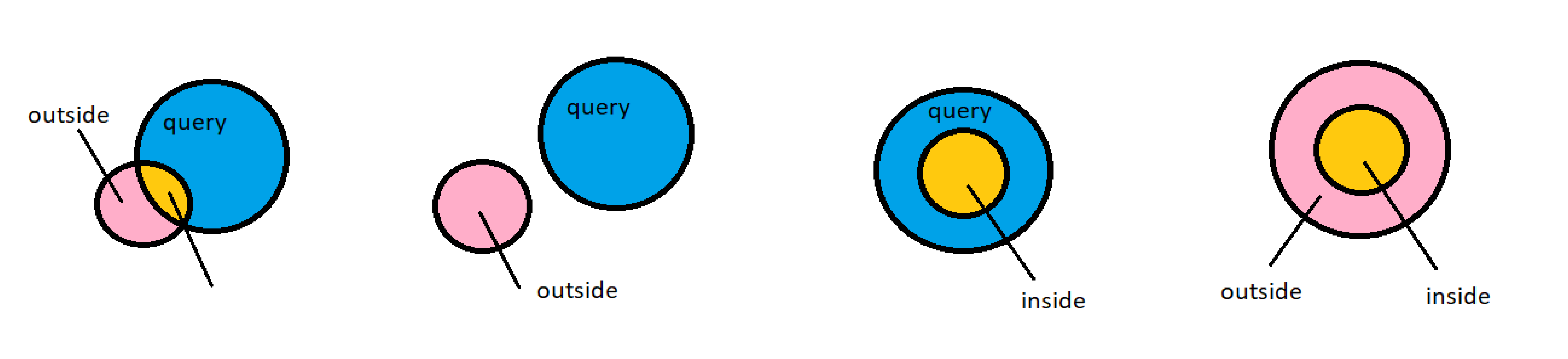
- TBR
  - TBR doc Index represents the total count for each combination of all multi-value attributes like AUS/AIS/PDA/DMS etc. We assume that the ratio of each combination vs. total keeps the same for AGE/GENDER.
  - The maximum number of rows in TBR doc Index in theory is the number of total users.
  - The purpose of TBR is to reduce the number of rows in Predictions.
- Booking Bucket
  - Booking bucket is defined as the current Booking excludes all Bookings with high priority. High priority Booking will take the overlap from all connected low priority Booking’s inventory.
  - Booking bucket has its priority which will be determined by Optimizer. Query and Booking will always loop from the top priority BB to lowest priority BB

Concepts

- Booking Bucket (based on priority)
  - All booking buckets are exclusive each other.
  - Can have multiple bookings inside, but they all share the same BB query.
  - Booking query will split the existing BB if the existing BB-Q has extra inventory than BB booked. (BB3 -> [BB3-Q and BB3^Q])
  - If the existing BB-Q doesn't have extra inventory than BB booked, Booking query will still split existing BB and put current BB's booking into BB-Q and BB^Q. The new Booking will also take BB^Q



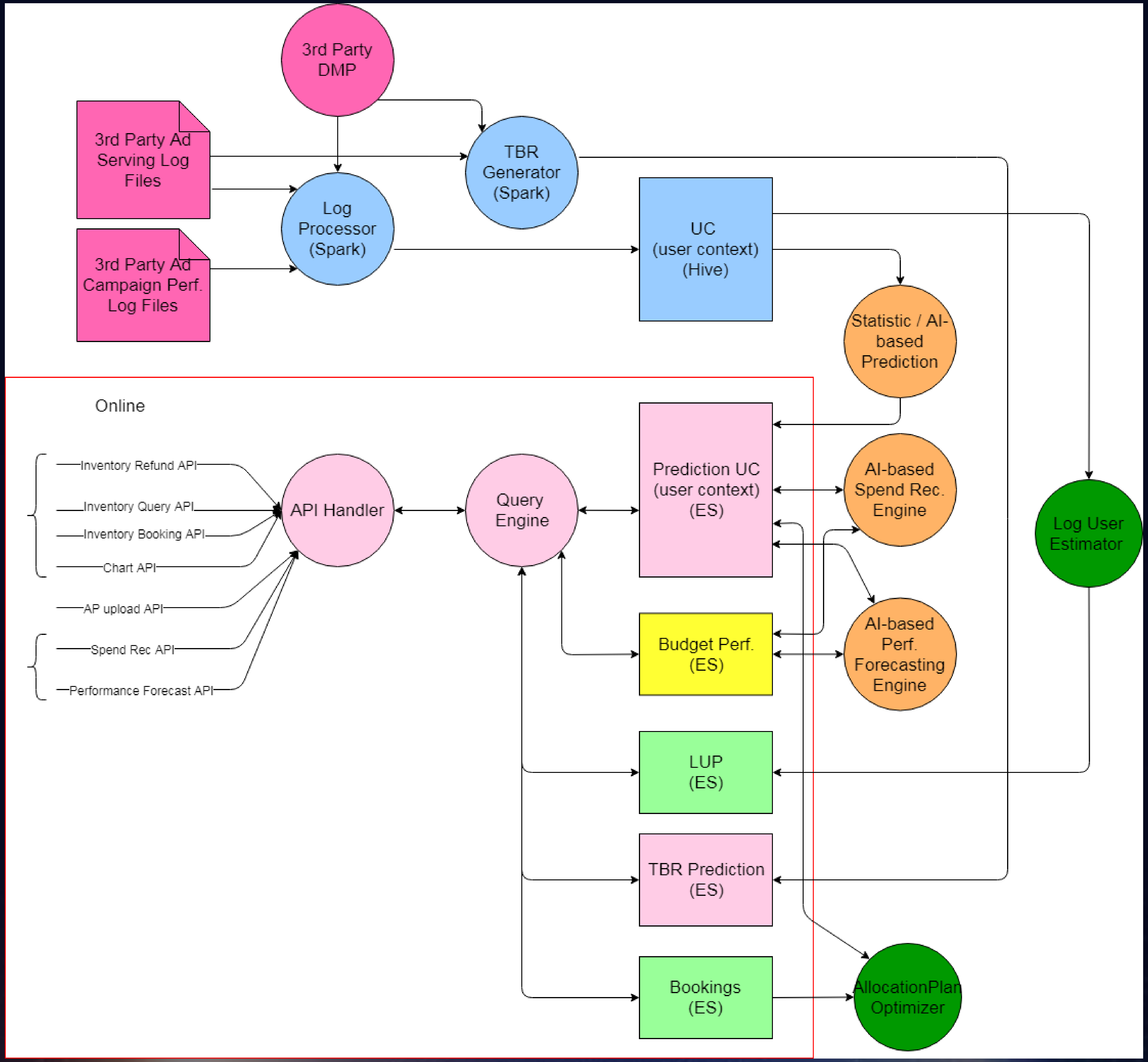
- Overlap between BB and Query



Agenda

1. Problem statement
2. Concepts
3. Architecture
4. Data structures
5. Algorithm
6. Workflow
7. DL-based pCTR Prediction

Architecture



Agenda

1. Problem statement
2. Concepts
3. Architecture
4. Data structures
5. Algorithm
6. Workflow
7. DL-based pCTR Prediction

Data Structures

- Predictions
- TBR
- Bookings
- Booking Buckets

ES-Schema-Data.json

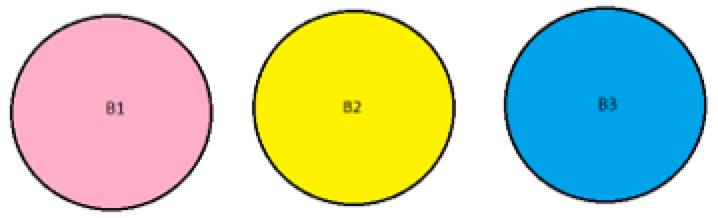
Agenda

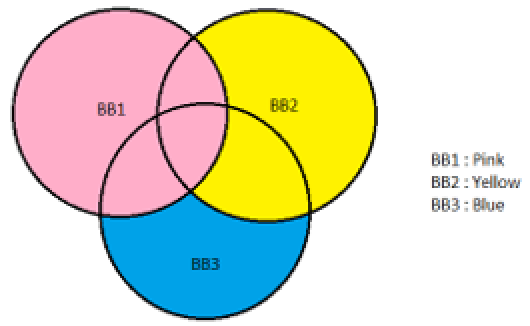
1. Problem statement
2. Concepts
3. Architecture
4. Data structures
5. Algorithm
6. Workflow
7. DL-based pCTR Prediction

Algorithm

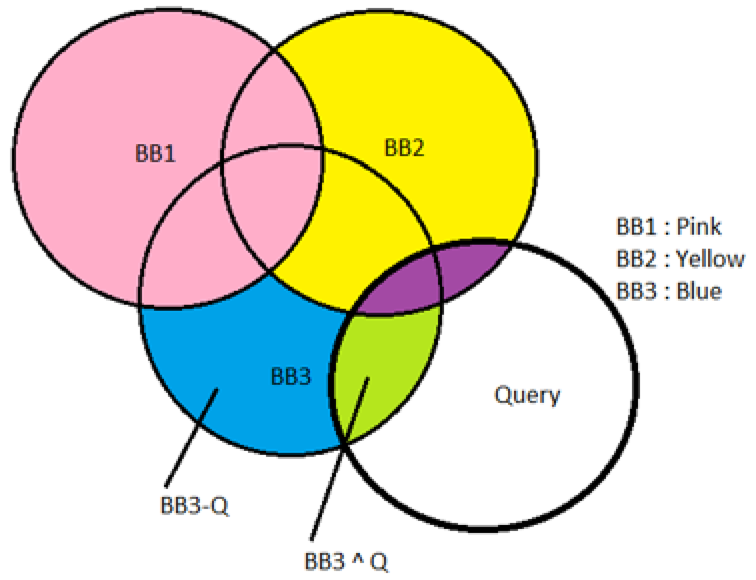
- Query

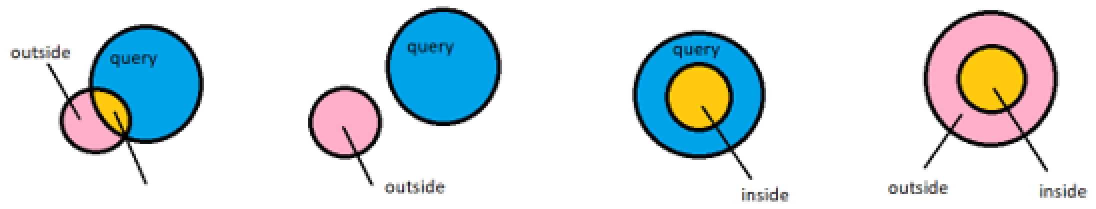
- The background is that we keep all bookings in Booking Index in ES, each booking has a booking query associated with it. This query might have two parts: single value part and multi-value part. We store single value part in Predictions Index and use all single values as the key. And we store all multi-value part in TBR index (adding gender, age to increase the accuracy). TBR is used to calculate system level possibility for each multi-value, for detail, please refer TBR section.
- For each day, we will have a list of Bookings which cover the day. In the mean time, we will have a list of Booking Bucket for the day as well. The relationship between Booking and Booking Bucket is as following:





- One constrains for BB is that each BB might have multiple bookings associated with it, all bookings within this BB share the same BB quey. If Booking Quey (Q) has an intersection with BB, and (BB-Q) has more impressions than BB's booked, this BB will split into two BBs: (BB-Q) and (BB^Q) – for Booking only.



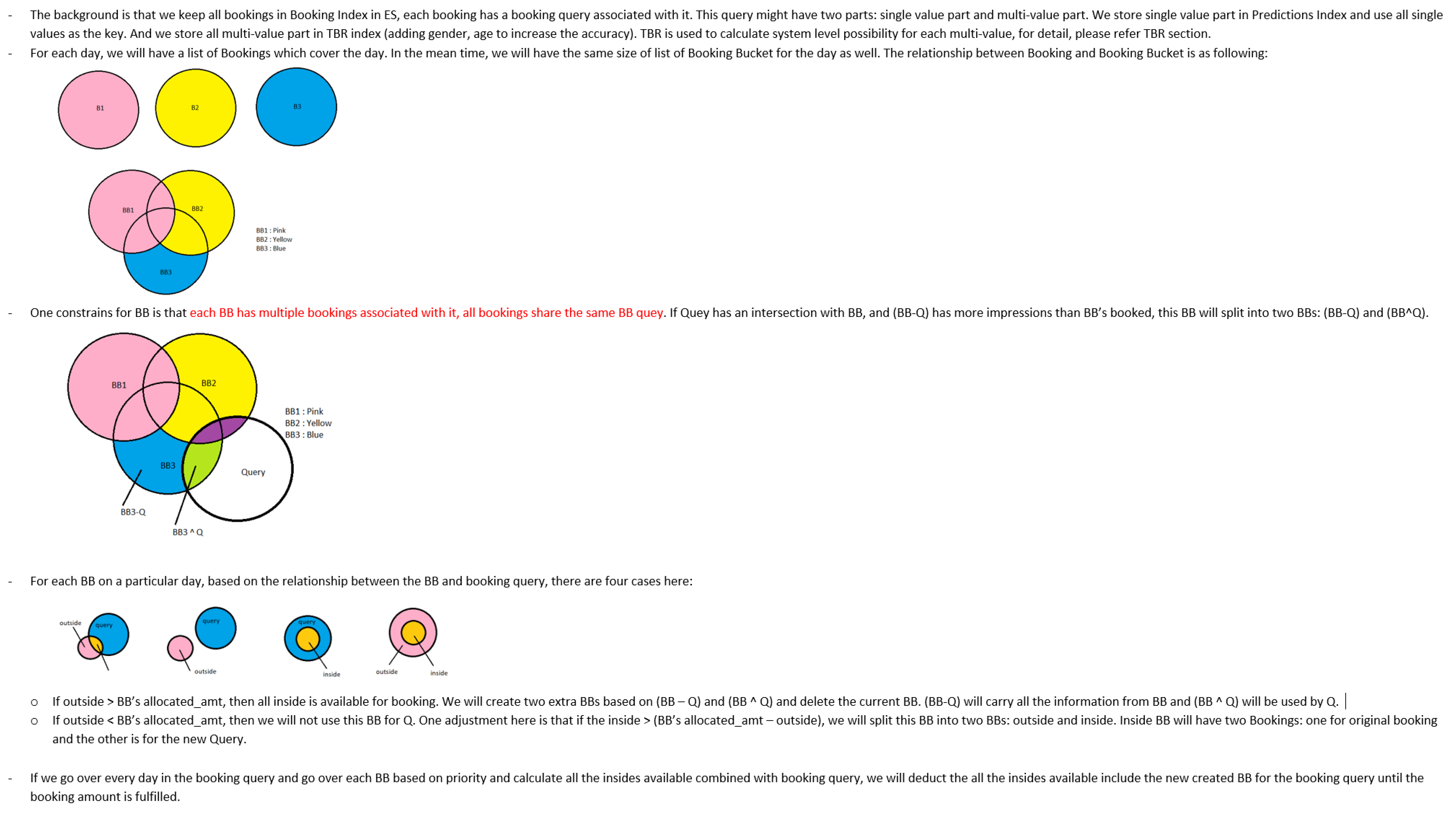


- For each BB on a particular day, based on the relationship between the BB and query, there are four cases here:
  - If outside > BB's allocated_amt, then all inside is available for query
  - If outside < BB's allocated_amt, then we will not use this BB for Q. One adjustment here is that if the inside > (BB's allocated_amt – outside), we will split this BB into two BBs: outside and inside. Inside BB will have two Bookings: one for original booking and the other is for the new Query.

- If we go over every day in the query and go over each BB based on priority and calculate all the insides available combined with query, this will become query result.

Algorithm

- Booking



- The background is that we keep all bookings in Booking Index in ES, each booking has a booking query associated with it. This query might have two parts: single value part and multi-value part. We store single value part in Predictions Index and use all single values as the key. And we store all multi-value part in TBR index (adding gender, age to increase the accuracy). TBR is used to calculate system level possibility for each multi-value, for detail, please refer TBR section.
- For each day, we will have a list of Bookings which cover the day. In the mean time, we will have the same size of list of Booking Bucket for the day as well. The relationship between Booking and Booking Bucket is as following:

- One constrains for BB is that each BB has multiple bookings associated with it, all bookings share the same BB quey. If Quey has an intersection with BB, and (BB-Q) has more impressions than BB's booked, this BB will split into two BBs: (BB-Q) and (BB^Q).

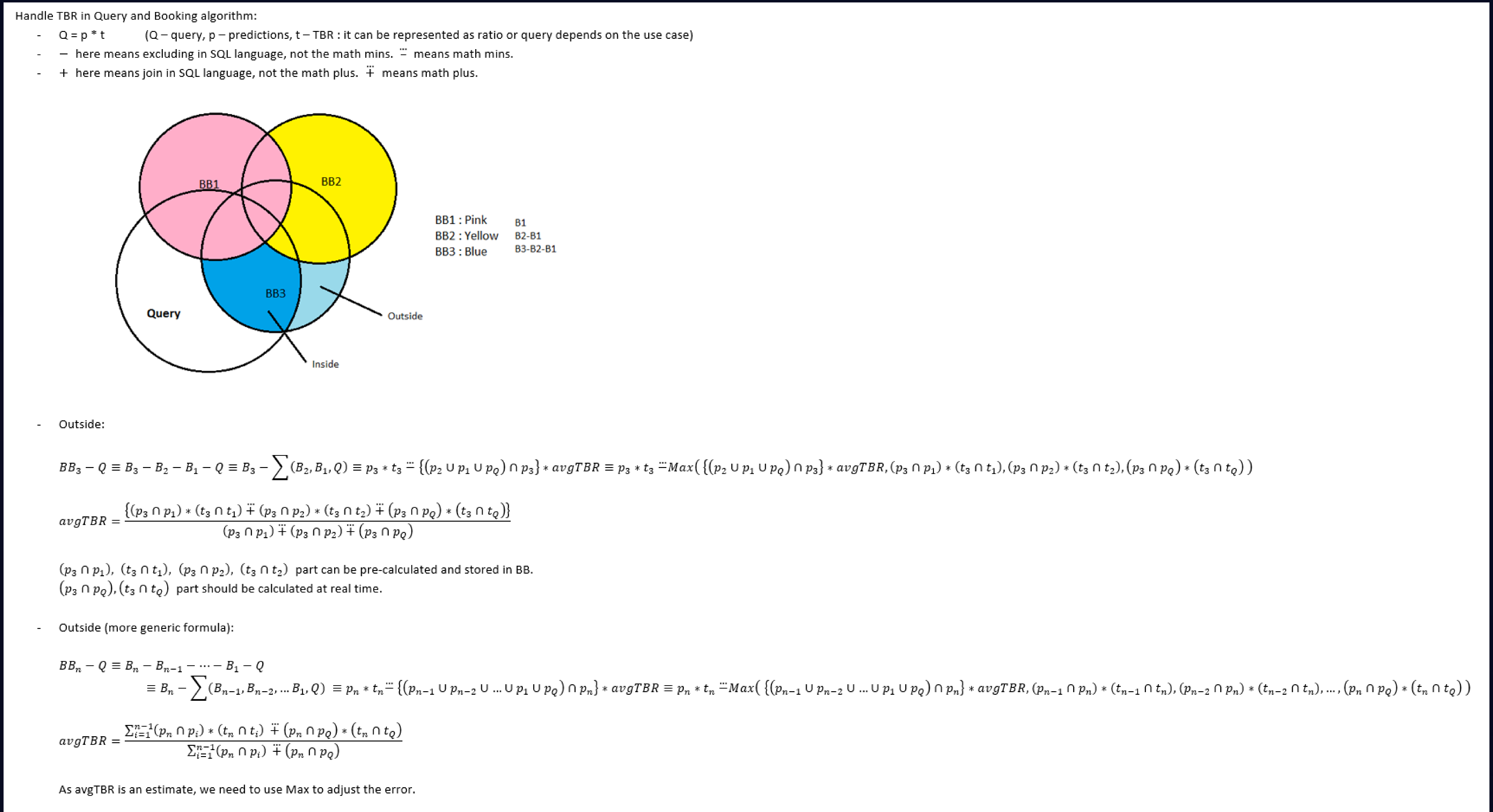
- For each BB on a particular day, based on the relationship between the BB and booking query, there are four cases here:

  - If outside > BB's allocated_amt, then all inside is available for booking. We will create two extra BBs based on (BB – Q) and (BB ^ Q) and delete the current BB. (BB-Q) will carry all the information from BB and (BB ^ Q) will be used by Q.
  - If outside < BB's allocated_amt, then we will not use this BB for Q. One adjustment here is that if the inside > (BB's allocated_amt – outside), we will split this BB into two BBs: outside and inside. Inside BB will have two Bookings: one for original booking and the other is for the new Query.

- If we go over every day in the booking query and go over each BB based on priority and calculate all the insides available combined with booking query, we will deduct the all the insides available include the new created BB for the booking query until the booking amount is fulfilled.

Algorithm

- TBR Handling in Booking



Handle TBR in Query and Booking algorithm:

- Q = p * t (Q – query, p – predictions, t – TBR : it can be represented as ratio or query depends on the use case)
- $-$ here means excluding in SQL language, not the math mins. $\overset{\cdots}{-}$ means math mins.
- $+$ here means join in SQL language, not the math plus. $\overset{\cdots}{+}$ means math plus.

BB1
BB2
BB3
Query
Outside
Inside

BB1 : Pink	B1
BB2 : Yellow	B2-B1
BB3 : Blue	B3-B2-B1

- Outside:

$$BB_3 - Q \equiv B_3 - B_2 - B_1 - Q \equiv B_3 - \sum(B_2, B_1, Q) \equiv p_3 * t_3 \overset{\cdots}{-} \{(p_2 \cup p_1 \cup p_Q) \cap p_3\} * avgTBR \equiv p_3 * t_3 \overset{\cdots}{-} Max(\{(p_2 \cup p_1 \cup p_Q) \cap p_3\} * avgTBR, (p_3 \cap p_1) * (t_3 \cap t_1), (p_3 \cap p_2) * (t_3 \cap t_2), (p_Q \cap p_Q) * (t_3 \cap t_Q))$$

$$avgTBR = \frac{\{(p_3 \cap p_1) * (t_3 \cap t_1) \overset{\cdots}{+} (p_3 \cap p_2) * (t_3 \cap t_2) \overset{\cdots}{+} (p_3 \cap p_Q) * (t_3 \cap t_Q)\}}{(p_3 \cap p_1) \overset{\cdots}{+} (p_3 \cap p_2) \overset{\cdots}{+} (p_3 \cap p_Q)}$$

$(p_3 \cap p_1)$, $(t_3 \cap t_1)$, $(p_3 \cap p_2)$, $(t_3 \cap t_2)$ part can be pre-calculated and stored in BB.
$(p_3 \cap p_Q)$, $(t_3 \cap t_Q)$ part should be calculated at real time.

- Outside (more generic formula):

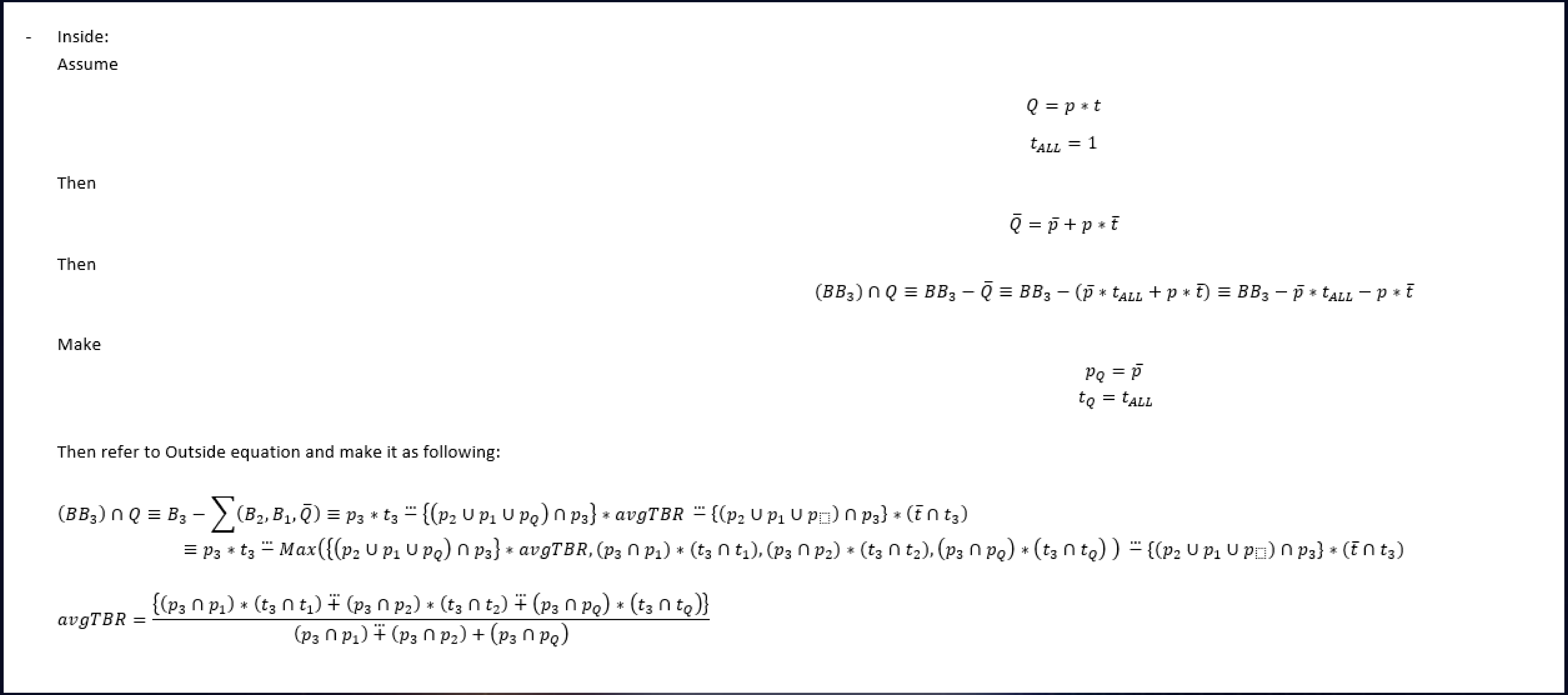
$$\begin{aligned} BB_n - Q &\equiv B_n - B_{n-1} - \cdots - B_1 - Q \\ &\equiv B_n - \sum(B_{n-1}, B_{n-2}, \ldots B_1, Q) \equiv p_n * t_n \overset{\cdots}{-} \{(p_{n-1} \cup p_{n-2} \cup \ldots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR \equiv p_n * t_n \overset{\cdots}{-} Max(\{(p_{n-1} \cup p_{n-2} \cup \ldots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR, (p_{n-1} \cap p_n) * (t_{n-1} \cap t_n), (p_{n-2} \cap p_n) * (t_{n-2} \cap t_n), \ldots, (p_n \cap p_Q) * (t_n \cap t_Q)) \end{aligned}$$

$$avgTBR = \frac{\sum_{i=1}^{n-1}(p_n \cap p_i) * (t_n \cap t_i) \overset{\cdots}{+} (p_n \cap p_Q) * (t_n \cap t_Q)}{\sum_{i=1}^{n-1}(p_n \cap p_i) \overset{\cdots}{+} (p_n \cap p_Q)}$$

As avgTBR is an estimate, we need to use Max to adjust the error.

Algorithm

- TBR Handling in Booking



- Inside:

Assume

$$Q = p * t$$

$$t_{ALL} = 1$$

Then

$$\bar{Q} = \bar{p} + p * \bar{t}$$

Then

$$(BB_3) \cap Q \equiv BB_3 - \bar{Q} \equiv BB_3 - (\bar{p} * t_{ALL} + p * \bar{t}) \equiv BB_3 - \bar{p} * t_{ALL} - p * \bar{t}$$

Make

$$p_Q = \bar{p}$$
$$t_Q = t_{ALL}$$

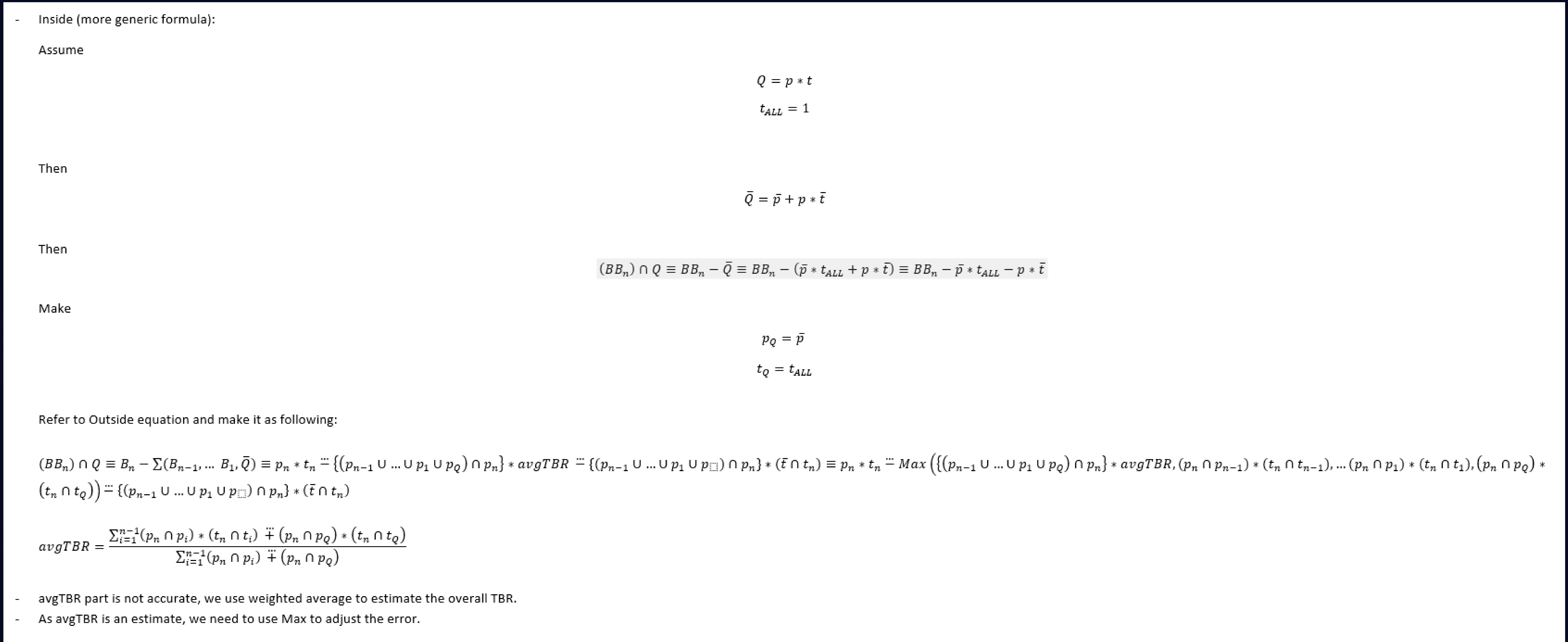
Then refer to Outside equation and make it as following:

$$\begin{aligned}(BB_3) \cap Q &\equiv B_3 - \sum(B_2, B_1, \bar{Q}) \equiv p_3 * t_3 \overset{\cdots}{-} \{(p_2 \cup p_1 \cup p_Q) \cap p_3\} * avgTBR \overset{\cdots}{-} \{(p_2 \cup p_1 \cup p_{\square}) \cap p_3\} * (\bar{t} \cap t_3) \\ &\equiv p_3 * t_3 \overset{\cdots}{-} Max(\{(p_2 \cup p_1 \cup p_Q) \cap p_3\} * avgTBR, (p_3 \cap p_1) * (t_3 \cap t_1), (p_3 \cap p_2) * (t_3 \cap t_2), (p_3 \cap p_Q) * (t_3 \cap t_Q)\ ) \overset{\cdots}{-} \{(p_2 \cup p_1 \cup p_{\square}) \cap p_3\} * (\bar{t} \cap t_3)\end{aligned}$$

$$avgTBR = \frac{\{(p_3 \cap p_1) * (t_3 \cap t_1) \overset{\cdots}{+} (p_3 \cap p_2) * (t_3 \cap t_2) \overset{\cdots}{+} (p_3 \cap p_Q) * (t_3 \cap t_Q)\}}{(p_3 \cap p_1) \overset{\cdots}{+} (p_3 \cap p_2) + (p_3 \cap p_Q)}$$

Algorithm

- TBR Handling in Booking



- Inside (more generic formula):

Assume

$$Q = p * t$$

$$t_{ALL} = 1$$

Then

$$\bar{Q} = \bar{p} + p * \bar{t}$$

Then

$$(BB_n) \cap Q \equiv BB_n - \bar{Q} \equiv BB_n - (\bar{p} * t_{ALL} + p * \bar{t}) \equiv BB_n - \bar{p} * t_{ALL} - p * \bar{t}$$

Make

$$p_Q = \bar{p}$$

$$t_Q = t_{ALL}$$

Refer to Outside equation and make it as following:

$$(BB_n) \cap Q \equiv B_n - \sum(B_{n-1}, \dots\ B_1, \bar{Q}) \equiv p_n * t_n \overset{\cdots}{-} \{(p_{n-1} \cup \dots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR \overset{\cdots}{-} \{(p_{n-1} \cup \dots \cup p_1 \cup p_{\square}) \cap p_n\} * (\bar{t} \cap t_n) \equiv p_n * t_n \overset{\cdots}{-} Max\left(\{(p_{n-1} \cup \dots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR, (p_n \cap p_{n-1}) * (t_n \cap t_{n-1}), \dots (p_n \cap p_1) * (t_n \cap t_1), (p_n \cap p_Q) * (t_n \cap t_Q)\right) \overset{\cdots}{-} \{(p_{n-1} \cup \dots \cup p_1 \cup p_{\square}) \cap p_n\} * (\bar{t} \cap t_n)$$

$$avgTBR = \frac{\sum_{i=1}^{n-1}(p_n \cap p_i) * (t_n \cap t_i) \overset{\cdots}{+} (p_n \cap p_Q) * (t_n \cap t_Q)}{\sum_{i=1}^{n-1}(p_n \cap p_i) \overset{\cdots}{+} (p_n \cap p_Q)}$$

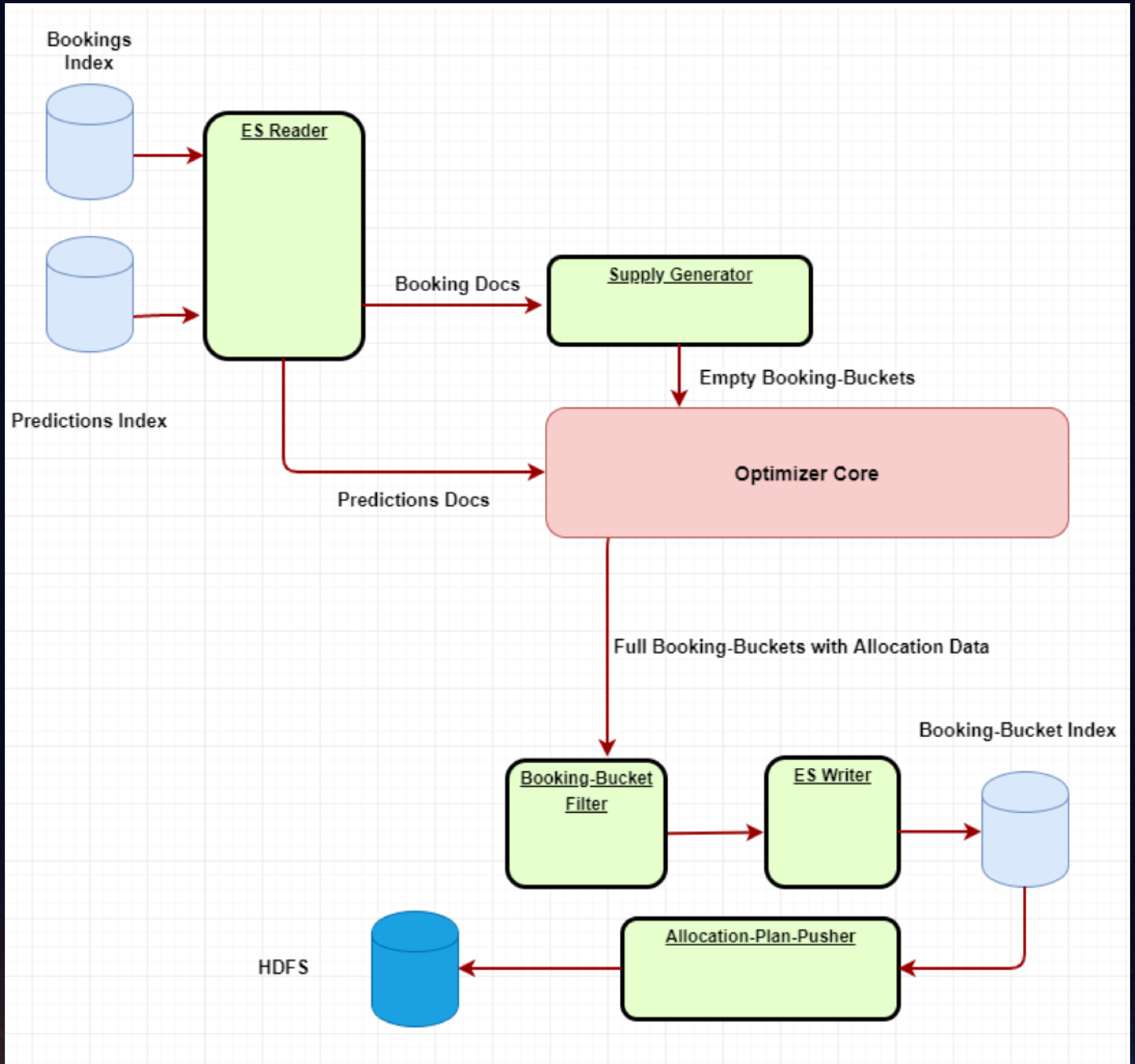
- avgTBR part is not accurate, we use weighted average to estimate the overall TBR.
- As avgTBR is an estimate, we need to use Max to adjust the error.

Agenda

1. Problem statement
2. Concepts
3. Architecture
4. Data structures
5. Algorithm
6. Workflow
7. DL-based pCTR Prediction

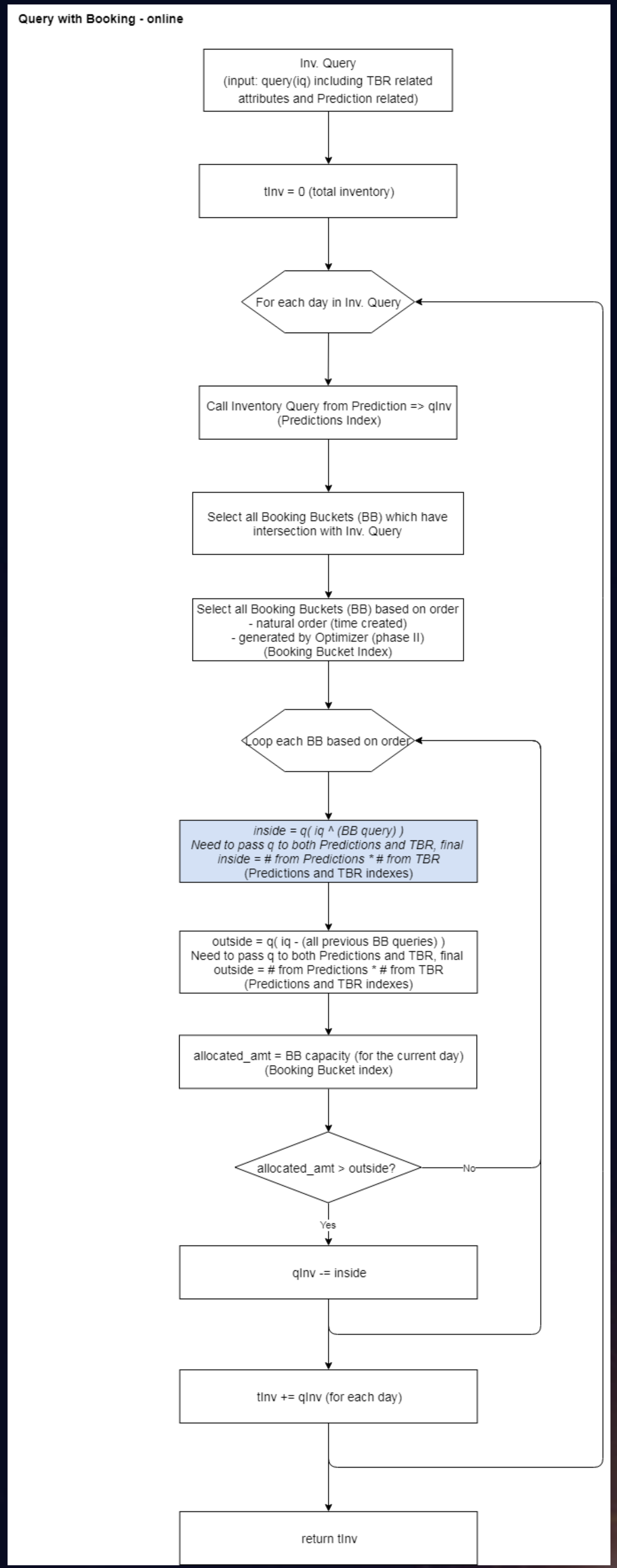
Workflow

- Offline
  - Optimizer



Workflow

- Online - Inventory



Query with Booking - online

Inv. Query
(input: query(iq) including TBR related attributes and Prediction related)

tInv = 0 (total inventory)

For each day in Inv. Query

Call Inventory Query from Prediction => qInv
(Predictions Index)

Select all Booking Buckets (BB) which have intersection with Inv. Query

Select all Booking Buckets (BB) based on order
- natural order (time created)
- generated by Optimizer (phase II)
(Booking Bucket Index)

Loop each BB based on order

inside = q(iq ^ (BB query))
*Need to pass q to both Predictions and TBR, final inside = # from Predictions * # from TBR*
(Predictions and TBR indexes)

outside = q(iq - (all previous BB queries))
Need to pass q to both Predictions and TBR, final outside = # from Predictions * # from TBR
(Predictions and TBR indexes)

allocated_amt = BB capacity (for the current day)
(Booking Bucket index)

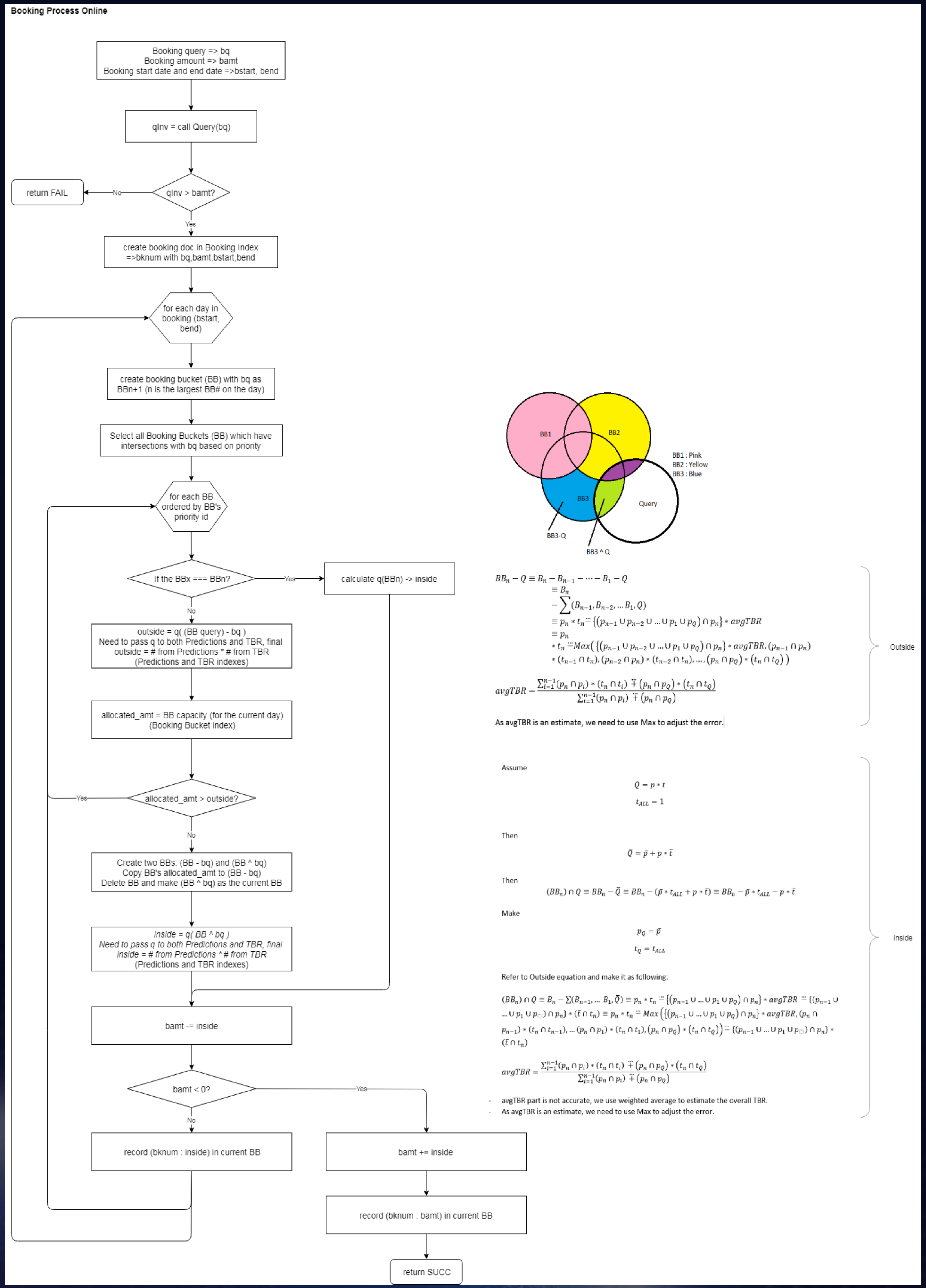
allocated_amt > outside?

qInv -= inside

tInv += qInv (for each day)

return tInv

- Online - Booking



Booking Process Online

Booking query => bq
Booking amount => bamt
Booking start date and end date =>bstart, bend

qInv = call Query(bq)

qInv > bamt?

return FAIL

create booking doc in Booking Index
=>bknum with bq,bamt,bstart,bend

for each day in booking (bstart, bend)

create booking bucket (BB) with bq as BBn+1 (n is the largest BB# on the day)

Select all Booking Buckets (BB) which have intersections with bq based on priority

for each BB ordered by BB's priority id

If the BBx === BBn?

calculate q(BBn) -> inside

outside = q((BB query) - bq)
Need to pass q to both Predictions and TBR, final outside = # from Predictions * # from TBR
(Predictions and TBR indexes)

allocated_amt = BB capacity (for the current day)
(Booking Bucket index)

allocated_amt > outside?

Create two BBs: (BB - bq) and (BB ^ bq)
Copy BB's allocated_amt to (BB - bq)
Delete BB and make (BB ^ bq) as the current BB

inside = q(BB ^ bq)
*Need to pass q to both Predictions and TBR, final inside = # from Predictions * # from TBR*
(Predictions and TBR indexes)

bamt -= inside

bamt < 0?

record (bknum : inside) in current BB

bamt += inside

record (bknum : bamt) in current BB

return SUCC

BB1 : Pink
BB2 : Yellow
BB3 : Blue

$$BB_n - Q \equiv B_n - B_{n-1} - \cdots - B_1 - Q$$
$$\equiv B_n - \sum (B_{n-1}, B_{n-2}, \dots B_1, Q)$$
$$\equiv p_n * t_n \overset{\cdots}{-} \{(p_{n-1} \cup p_{n-2} \cup \dots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR$$
$$\equiv p_n * t_n \overset{\cdots}{-} Max(\ \{(p_{n-1} \cup p_{n-2} \cup \dots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR, (p_{n-1} \cap p_n) * (t_{n-1} \cap t_n), (p_{n-2} \cap p_n) * (t_{n-2} \cap t_n), \dots, (p_n \cap p_Q) * (t_n \cap t_Q)\ )$$

$$avgTBR = \frac{\sum_{i=1}^{n-1}(p_n \cap p_i) * (t_n \cap t_i)\ \ddot{\mp}\ (p_n \cap p_Q) * (t_n \cap t_Q)}{\sum_{i=1}^{n-1}(p_n \cap p_i)\ \ddot{\mp}\ (p_n \cap p_Q)}$$

As avgTBR is an estimate, we need to use Max to adjust the error.

Outside

Assume

$$Q = p * t$$
$$t_{ALL} = 1$$

Then

$$\bar{Q} = \bar{p} + p * \bar{t}$$

Then

$$(BB_n) \cap Q \equiv BB_n - \bar{Q} \equiv BB_n - (\bar{p} * t_{ALL} + p * \bar{t}) \equiv BB_n - \bar{p} * t_{ALL} - p * \bar{t}$$

Make

$$p_Q = \bar{p}$$
$$t_Q = t_{ALL}$$

Refer to Outside equation and make it as following:

$$(BB_n) \cap Q \equiv B_n - \sum(B_{n-1}, \dots B_1, \bar{Q}) \equiv p_n * t_n \overset{\cdots}{-} \{(p_{n-1} \cup \dots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR \overset{\cdots}{-} \{(p_{n-1} \cup \dots \cup p_1 \cup p_{\square}) \cap p_n\} * (\bar{t} \cap t_n) \equiv p_n * t_n \overset{\cdots}{-} Max\left(\{(p_{n-1} \cup \dots \cup p_1 \cup p_Q) \cap p_n\} * avgTBR, (p_n \cap p_{n-1}) * (t_n \cap t_{n-1}), \dots (p_n \cap p_1) * (t_n \cap t_1), (p_n \cap p_Q) * (t_n \cap t_Q)\right) \overset{\cdots}{-} \{(p_{n-1} \cup \dots \cup p_1 \cup p_{\square}) \cap p_n\} * (\bar{t} \cap t_n)$$

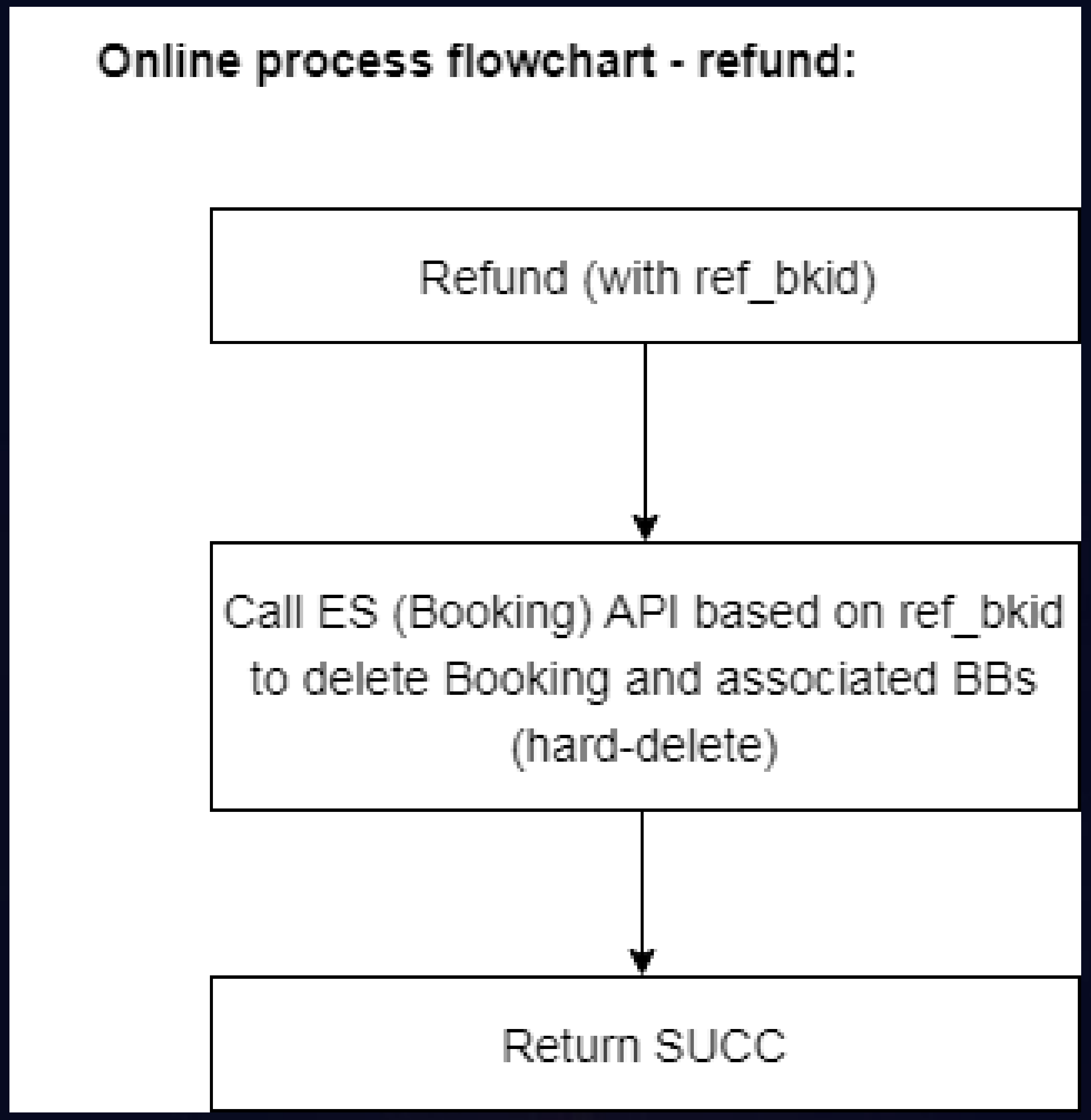
$$avgTBR = \frac{\sum_{i=1}^{n-1}(p_n \cap p_i) * (t_n \cap t_i)\ \ddot{\mp}\ (p_n \cap p_Q) * (t_n \cap t_Q)}{\sum_{i=1}^{n-1}(p_n \cap p_i)\ \ddot{\mp}\ (p_n \cap p_Q)}$$

- avgTBR part is not accurate, we use weighted average to estimate the overall TBR.
- As avgTBR is an estimate, we need to use Max to adjust the error.

Inside

Workflow

- Online - Refund



Online process flowchart - refund:

Refund (with ref_bkid)
Call ES (Booking) API based on ref_bkid to delete Booking and associated BBs (hard-delete)
Return SUCC

Agenda

1. Problem statement
2. Concepts
3. Architecture
4. Data structures
5. Algorithm
6. Workflow
7. DL-based pCTR Prediction

Concepts

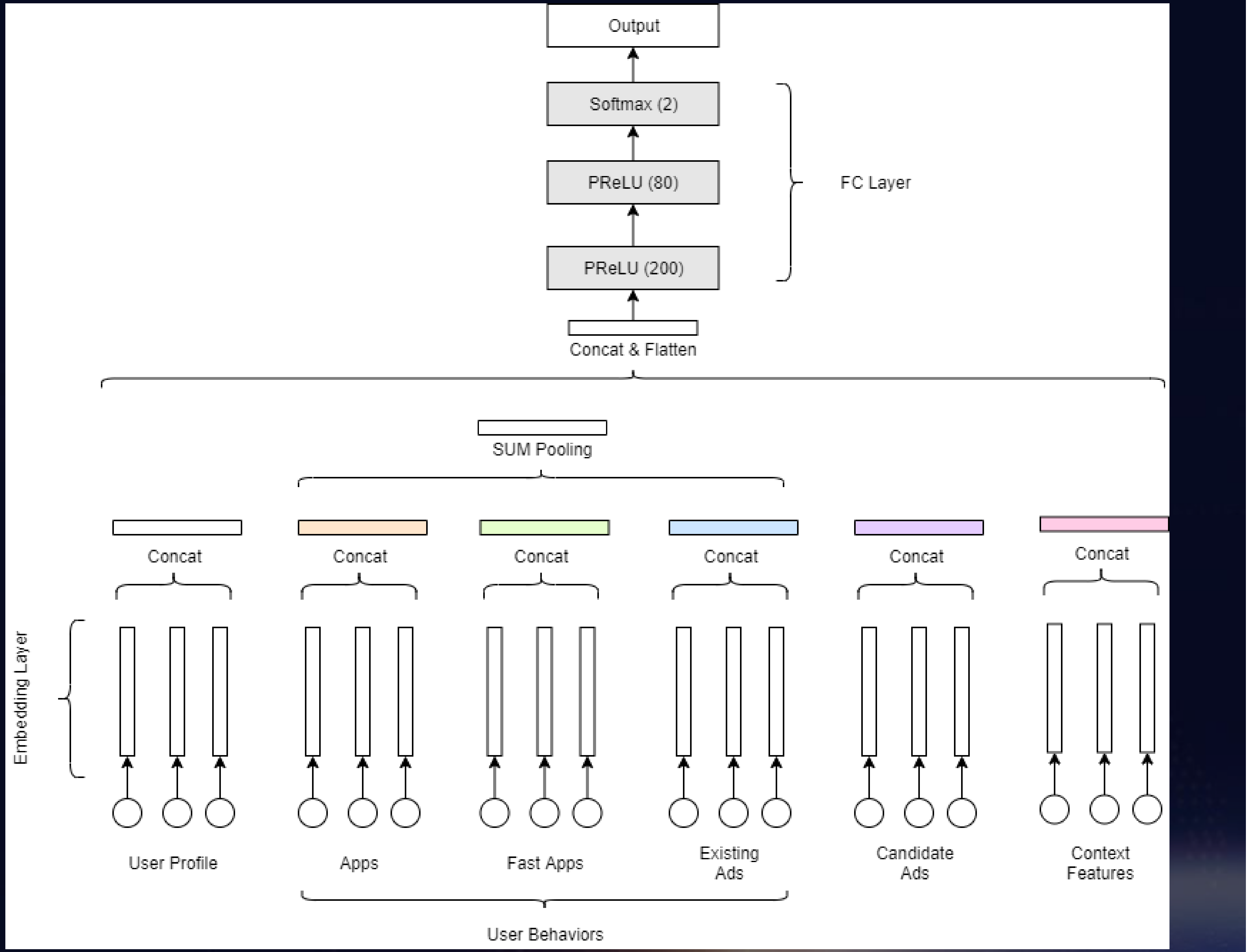
- DIN
  - DIN introduces ad’s oriented user attention in user interests’ representation vector to overcome the limitation. It adaptively calculates the representation vector of user interests by taking into consideration the relevance of historical behaviors given a candidate ad.
  - Limitation of traditional Embedding & MLP model.

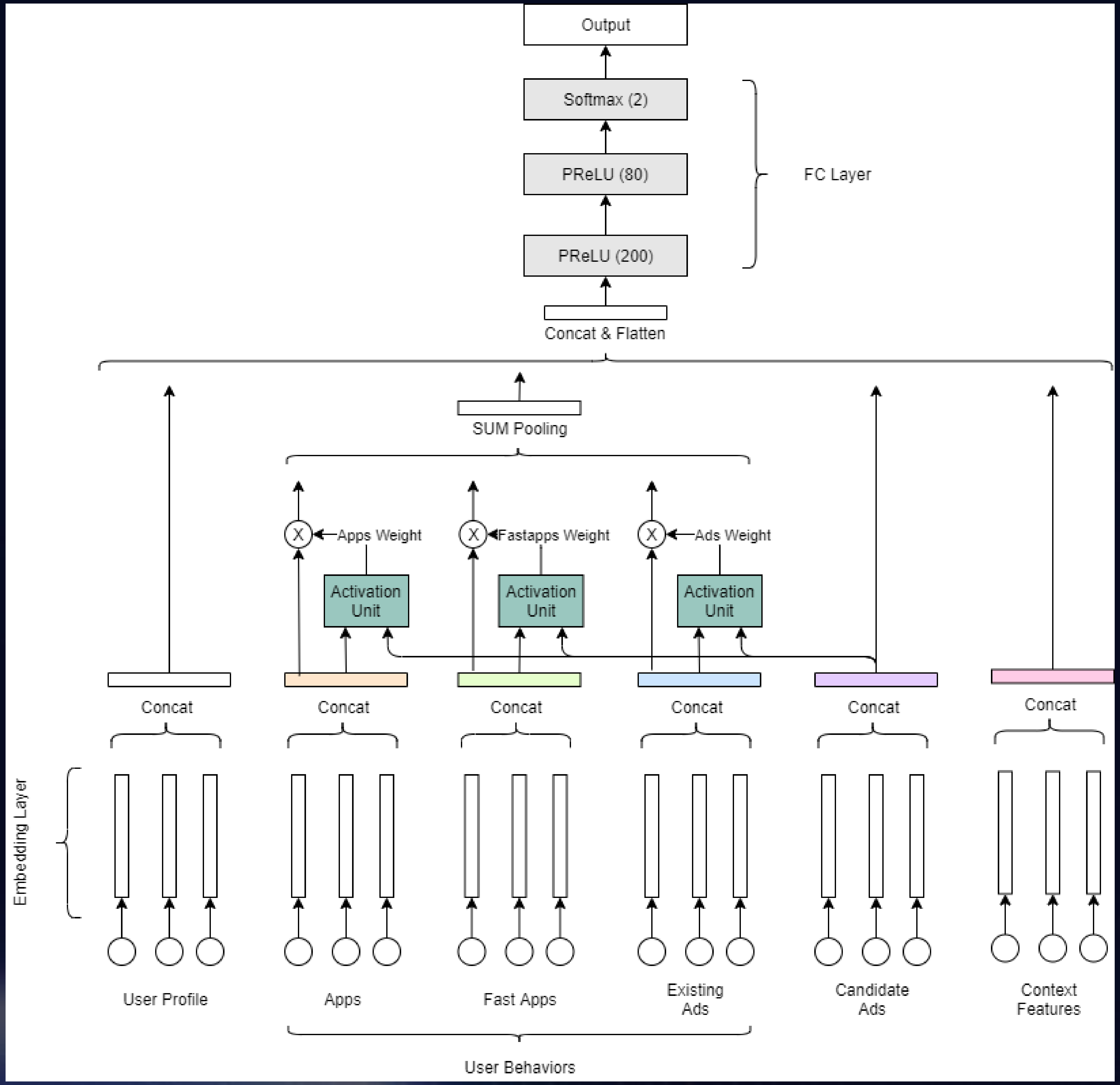
  - Inspired by Alibaba’s previous work https://arxiv.org/pdf/1706.06978.pdf

- Performance Forecasting
  - Giving fixed $$$ and creative, find the inventory to have the best performance

Models

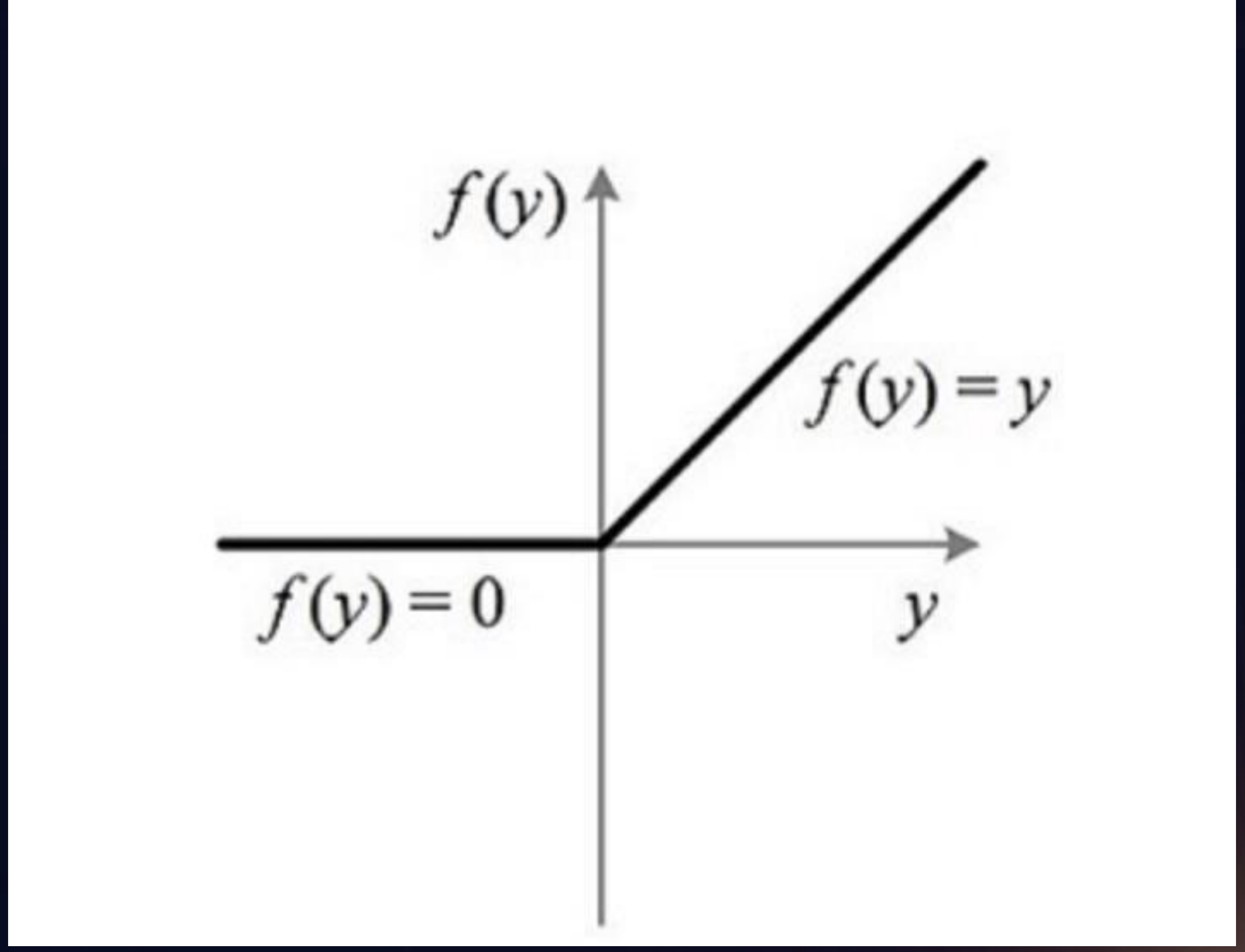
$$v_U(A) = f(v_A, \quad e_1, \quad e_2, \quad \cdots, \quad e_H) = \sum_{j=1}^{H} \alpha(e_j, \quad v_A)e_j = \sum_{j=1}^{H} w_j e_j$$



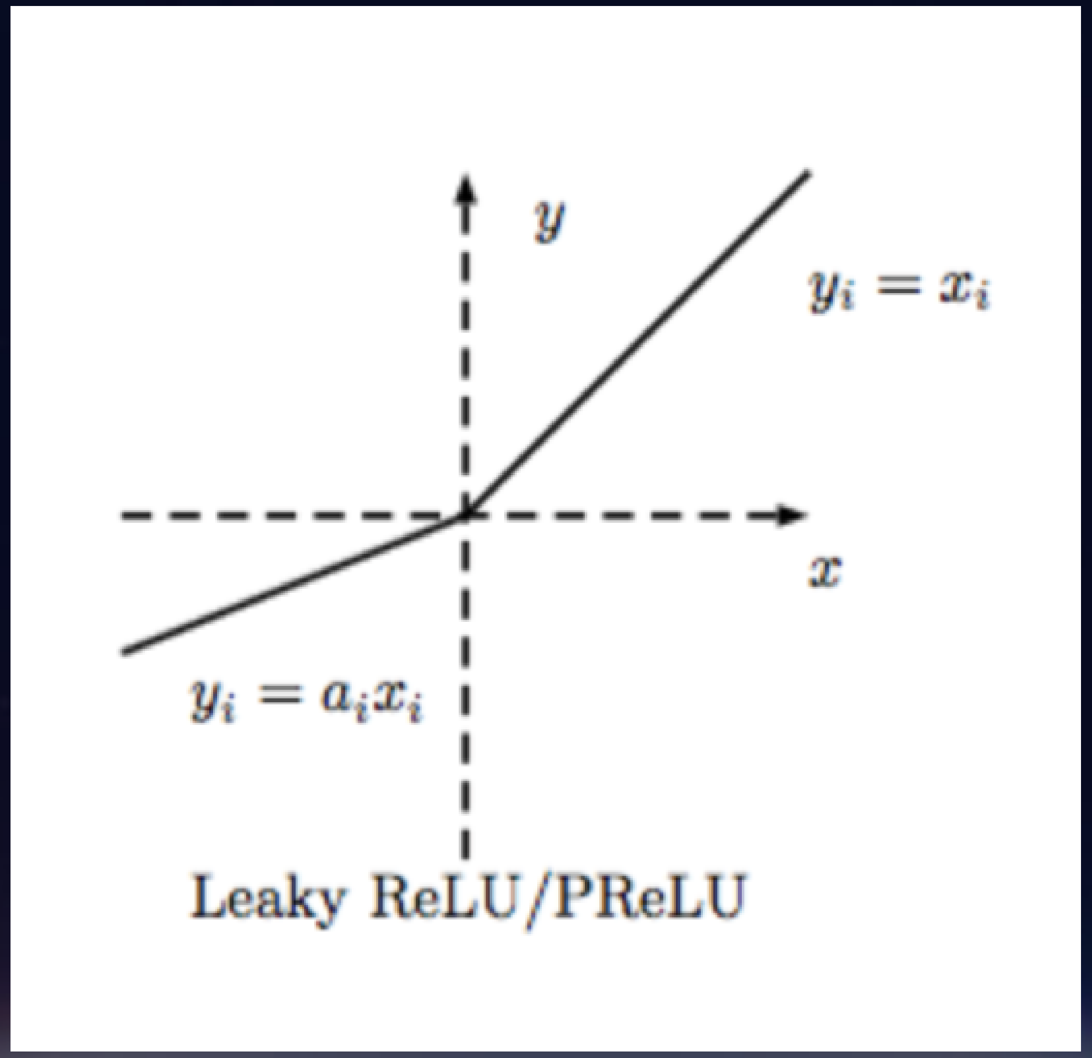


Activation Function

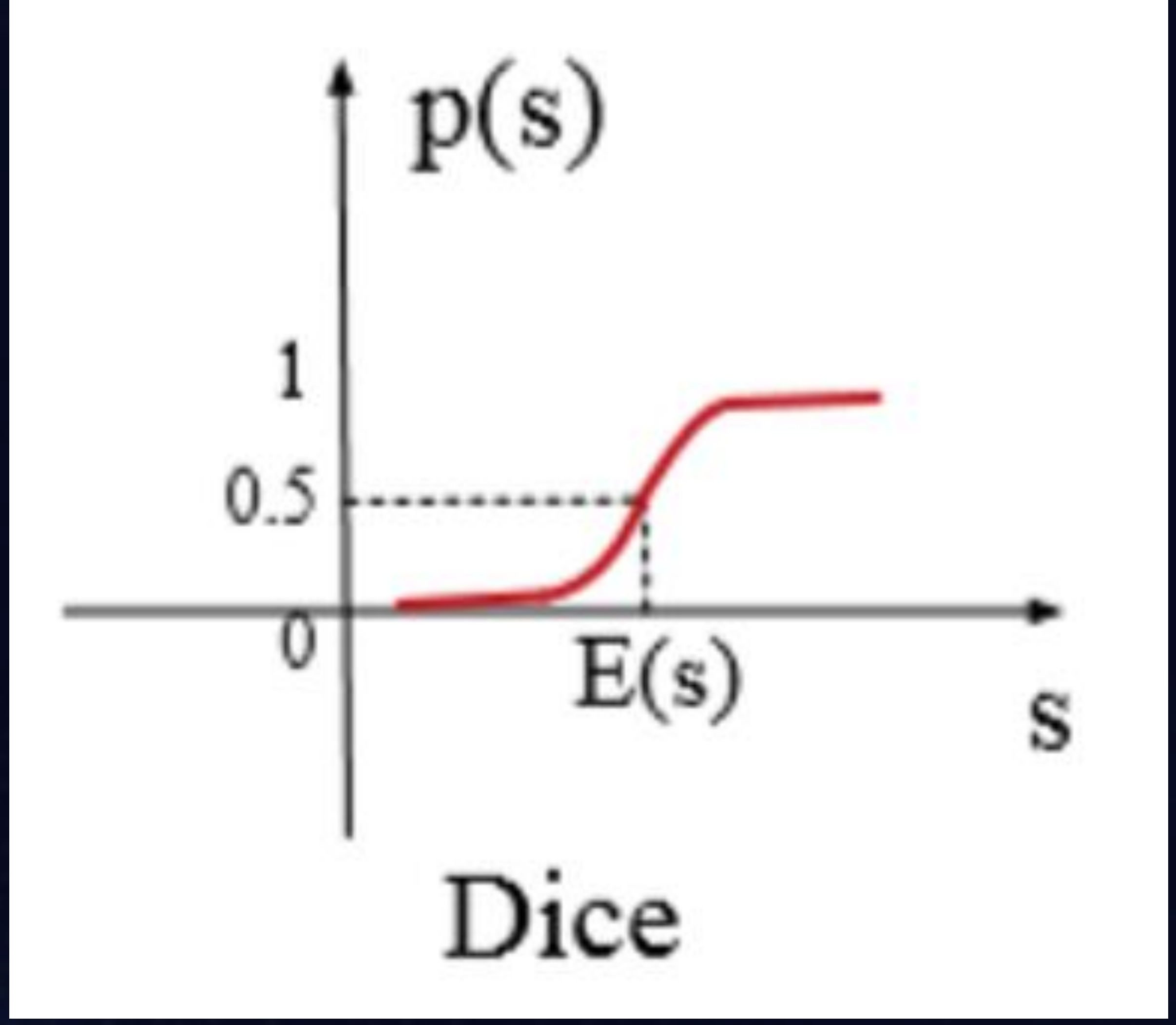
Traditional PReLU (Parametric Rectified Linear Unit) activation function takes a hard-rectified point with value 0, which may not be suitable when the inputs of each layer follow different distributions. DICE activation function is designed to adaptively adjust the rectified point according to distribution of input data, as shown in the figure below.



$$f(s) = \begin{cases} s & \text{if } s > 0 \\ \alpha s & \text{if } s \leq 0. \end{cases} = p(s) \cdot s + (1 - p(s)) \cdot \alpha s,$$



$$f(s) = p(s) \cdot s + (1 - p(s)) \cdot \alpha s, \ \ p(s) = \frac{1}{1 + e^{-\frac{s - E[s]}{\sqrt{Var[s] + \epsilon}}}}$$



Limitation

- Lack of modelling the change in user behavior
- Does not consider the changing trend of interest.

- Can only support limited BBs, we will have a new algo to resolve it soon.

Thank You