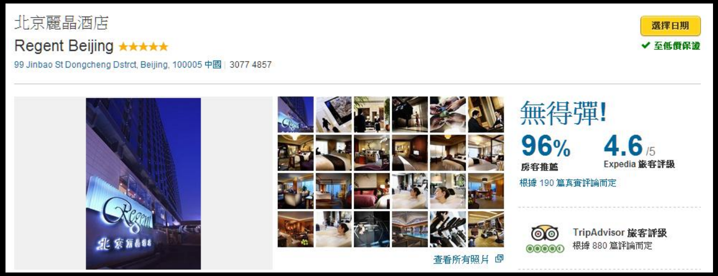
Combination of Diverse Ranking Models for Personalized Expedia Hotel Searches

binghsu & MLRush & BrickMover Team

TEAM MEMBERS:
XUDONG LIU, BING XU, YUYU ZHANG, QIANG YAN,
LIANG PANG, QIANG LI, HANXIAO SUN, BIN WANG

BACKGROUND

- Learning to rank hotels to maximize purchases



BASIC ANALYSIS

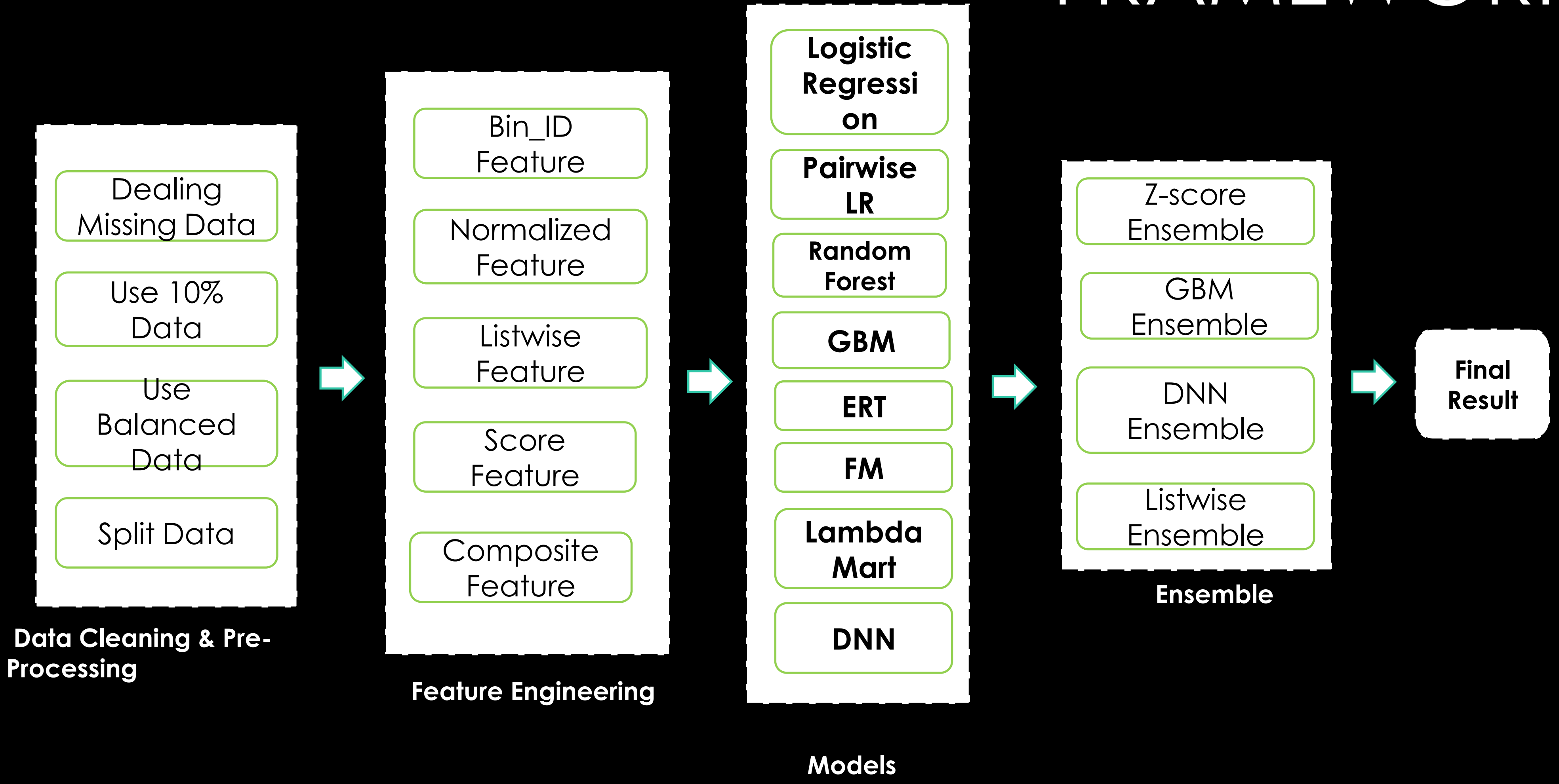
- What we have?
  - Hotel features
  - Query features
  - User features
  - Compare features

Cat	Num
Train	9917530
Test	6622629
Query in Train	336334
Query in Test	266230
Hotel Count in Train	136886
Hotel Count in Test	132888
…	…

TO SOLVE

- How to deal with the complex data.
- How to generate good features from these data.
- How to make these features work in a model.
- How to make models train fast in such a big data.
- How to ensemble models.

FRAMEWORKS


Dealing Missing Data
Use 10% Data
Use Balanced Data
Split Data
Data Cleaning & Pre-Processing
Bin_ID Feature
Normalized Feature
Listwise Feature
Score Feature
Composite Feature
Feature Engineering
Logistic Regression
Pairwise LR
Random Forest
GBM
ERT
FM
Lambda Mart
DNN
Models
Z-score Ensemble
GBM Ensemble
DNN Ensemble
Listwise Ensemble
Ensemble
Final Result

- Preprocess

PREPROCESS

The Big, Complex and Dirty Data

- Not all data is complete. Some is missing
- Data is too big for some models(e,g, GBM)
- Data is not balanced, negative data is much more
- Data appears different by count_id
- Feature is not equally important

PREPROCESS(CONT.)

- Dealing with Missing Data
  - *Use the first quartile to represent the missing data*
- Use 10% Data
  - *Randomly sample 10% by srch_id*
- Use Balanced Data
  - *Choose balanced positive and negative data*
- Split Data
  - *Split data by prop_country_id*

- Features

FEATURES

Keypoints

- Features are built in different methods
- Some special features including listwise features and score features
- Different features fit different models

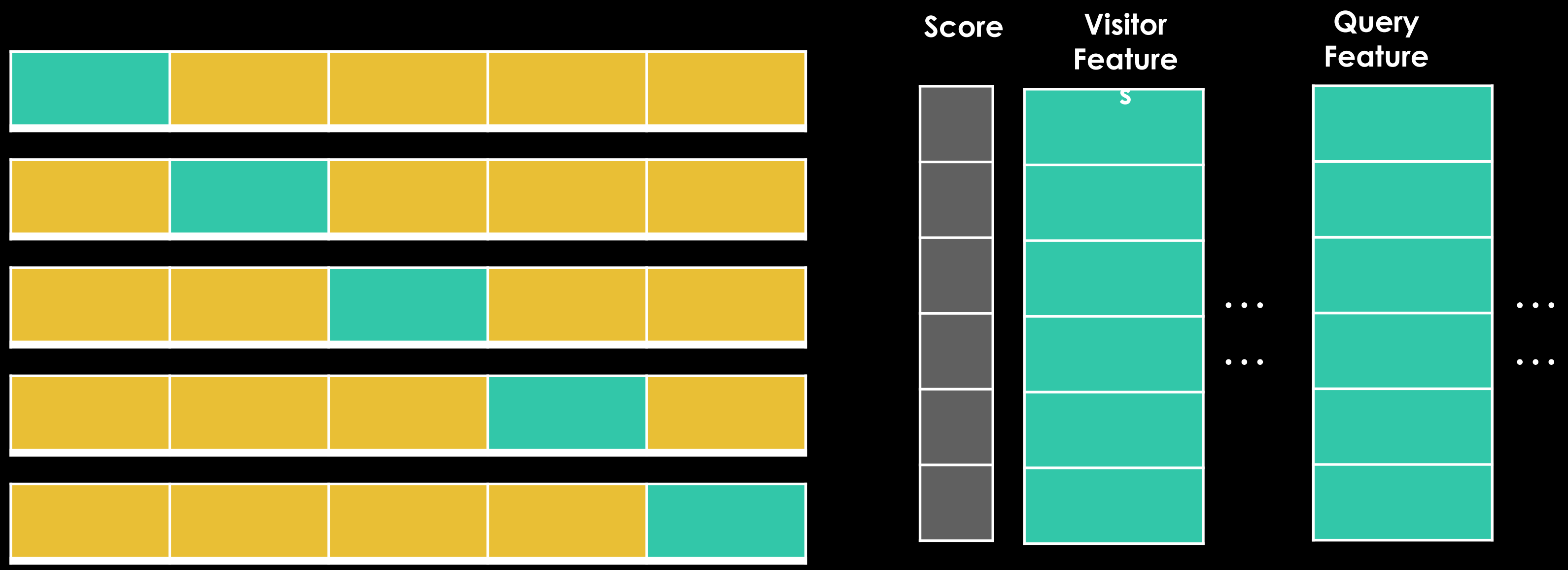
Cat.	Key Features
Bin_ID Feature	Prop_id Search_destination_id Srch_room_count Srch_booking_window
Normalized Feature	Price_usd Prop_location_score1 Prop_location_score2
Listwise Feature	Price_rank Price_diff_rank Star_rank
Score Feature	Fm_score Lr_score
Composite Feature	Roomcount_Bookwindow Adultcount_Childrencount

FEATURES (CONT.)

- Listwise feature calculate the rank value in a query.
- Composite Feature

$$F1\_F2 = F1 * max(Max(F2)) + F2$$

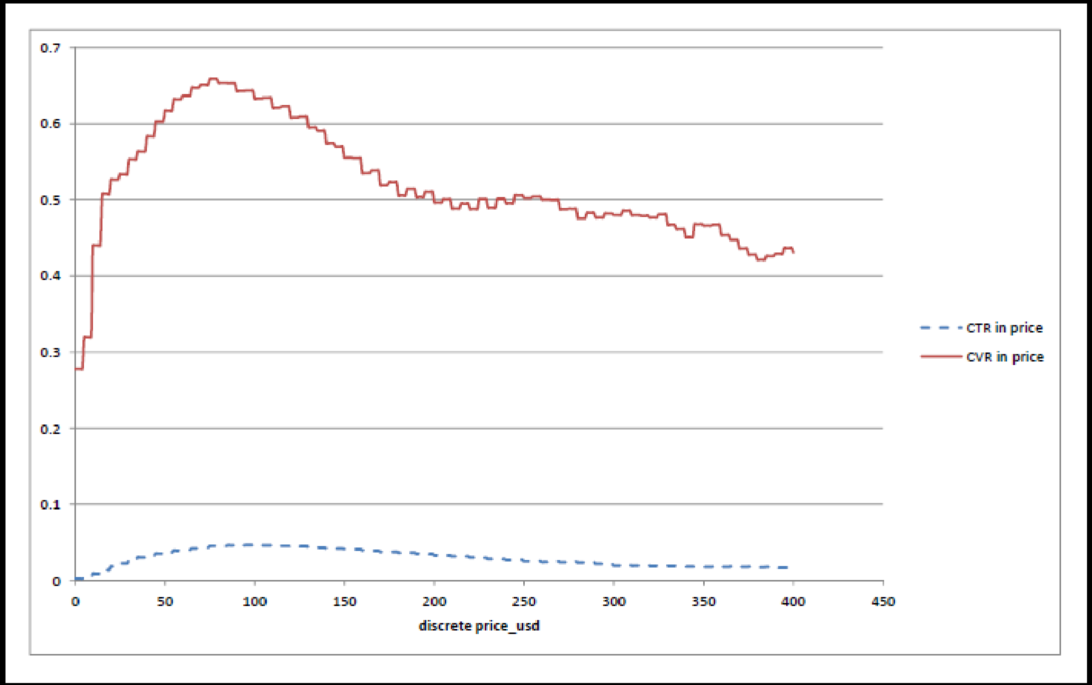
- Score Feature



FEATURES(CONT.)

- How to define good features?
- How to find good features?
- Good features in our models.

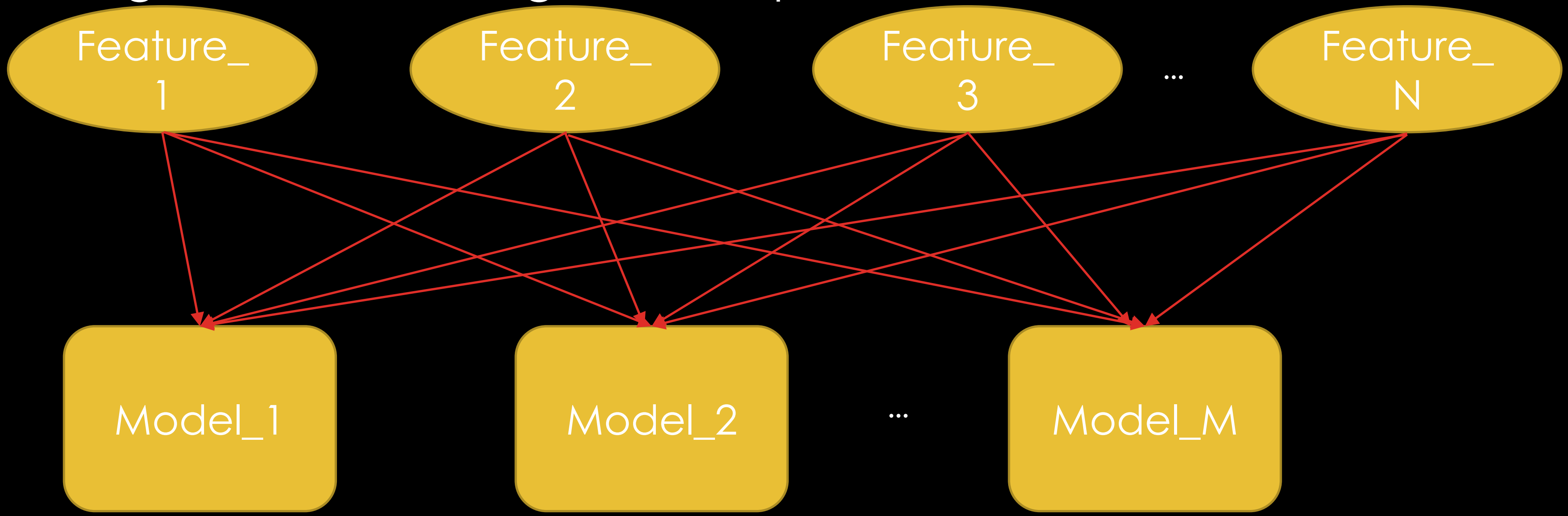
Feature_Name
Price_usd
Prop_location_score1
Prop_location_score2



- Models

MODELS

- Models should be used in the right way.
- Different features fit different models.
- Background knowledge and experiments are needed.



MODELS(CONT.)

Pairwise & Listwise Models	NDCG@38 on Validation Set
Pairwise Logistic Regression	0.51
Lambda Mart	0.5243

Pointwise Models	NDCG@38 on Validation Set
Logistic Regression	0.52
Random Forest	0.52
Gradient Boosting Machine	0.52477
Extremely Randomized Trees	0.51699
Fatorization Machine	0.5171

MODELS (CONT.)

Pointwise Models	Features Fitted
Logistic Regression	Bin_ID Feature, Normalized Feature, Listwise Feature
Random Forest and other tree based	All Feature with Listwise Feature and Composite Feature
Fatorization Machine	Bin_ID Feature, Location Feature, Listwise Feature
Pairwise Logistic Regression	Bin_ID Feature, Listwise Feture, Normalized Location Feature and Composite Feature
Lambda Mart	Score Feature, Normalized Feature, Listwise Feature

MODELS (CONT.)

- Factorization Machine
  - A long time to train.
  - A lot of feature engineering work.
  - Listwise features bring good result.
  - Bin or Normalized.

Bin_ID_Feature	Normalized_Feature	Rank_Feature
Prob_id	Price_usd	Price_rank
Srch_destination_id	Prop_location_score1	Price_diff_rank
Srch_room_count	Prop_location_score2	Star_rank

MODELS (CONT.)

- LambdaMART
  - Listwise model.
  - Some features(e.g. time and locations) won't work.
  - Score features work well.

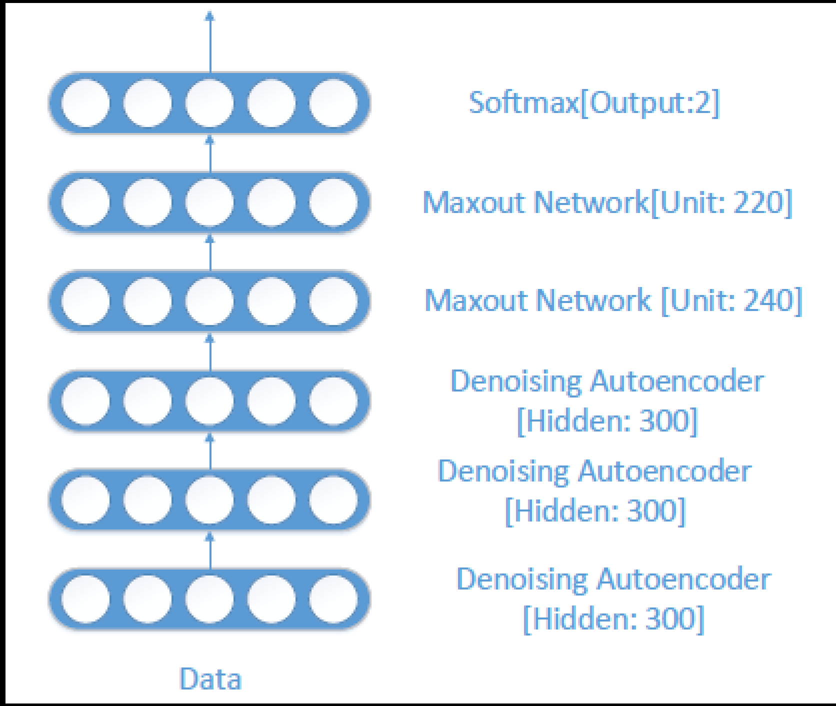
Score_Feature	**Normalized_Feature**	**Rank_Feature**
Fm_score	Price_usd	Price_rank
LR_score	Prop_location_score1	Price_diff_rank
	Prop_location_score2	Star_rank

MODELS (CONT.)

- Deep Learning Approach (Failure Case)

1. Softmax fails in the unblanaced case

2. Highly composite features contribute little (similar results find in combination random forest)



- Ensemble models

ENSEMBLE MODELS

- Why Ensemble?
  - Single model can't get the best result.
- Key points of Ensemble?
  - Diverse
    - Model Diverse
    - Feature Diverse
- How to Ensemble?

ENSEMBLE MODELS(CONT.)

- Z-score: simplest but work best

$$Z(x) = \frac{x - \bar{x}}{\sigma(x)} = \frac{x - \bar{x}}{\sqrt{(x - \bar{x})^2 / n}}$$

- GBM Ensemble

ENSEMBLE MODELS (CONT.)

- Deep Learning Ensemble
  - Dropout Logistic Regression (0.527)

- Listwise Ensemble
  - LambdaMart (Final Model, 0.53102)

TRAINING TIME IN SUMMARY

Single Model	Who	Training Time
Linear Model	Qiang Yan	Minutes to an hour
Random Forest	Bing Xu	4 hours
Factorization Machine	Xudong Liu & Liang Pang	10 hours
GBM/LambdaMART	Yuyu Zhang, Qiang Yan, etc	10 hours

Ensemble Model	Who	Training Time
Dropout Logistic Regression	Bing Xu	minutes
zscore	Yuyu Zhang	minutes
GBM	Liang Pang	An hour
LambdaMART	Qiang Yang	An hour

- Conclusion

CONCLUSION

- Tree based ranker (Random Forest/GBM/LambdaMart is robust in ranking hotel
- Linear Model is efficient but need more time to do feature engineering
- Deep Learning in pointwise may not be efficient
- Representation Learning is hard in this task by using trivial neural network

FURTHER INVESTIGATE

- 1. Using Bayesian Database to deal with the missing data
- 2. Deep Neural Network in pairwise and listwise
- 3. Representation learning to solve cold start problem

Thanks

binghsu & MLRush & BrickMover Team