amadeus

Search Findability
Machine learning on Journey Server for flight search optimization

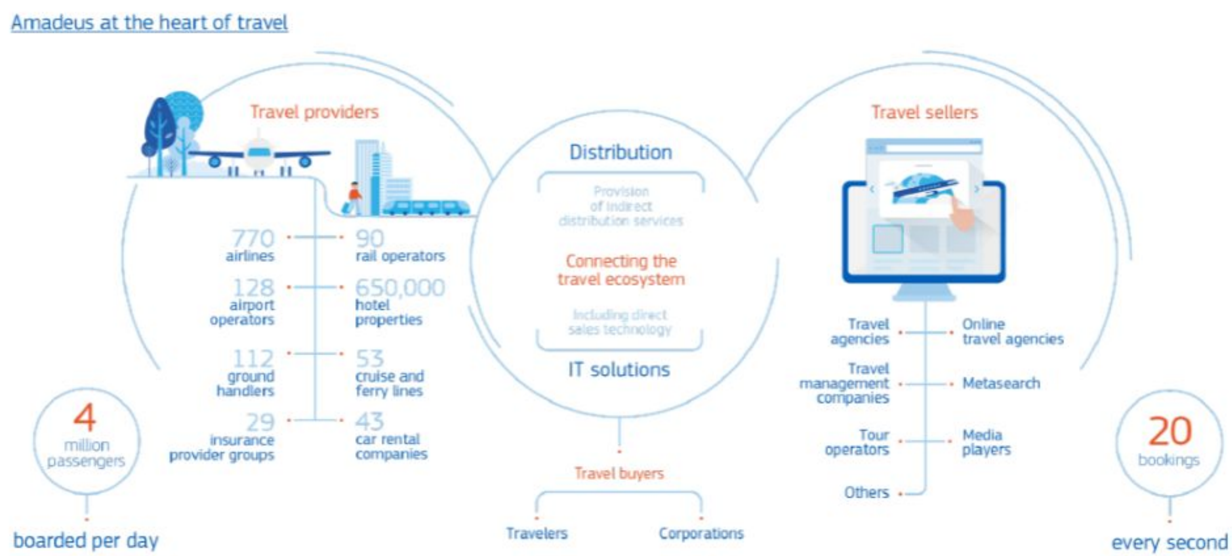
Agenda

1. Introduction & problem description
2. Solution design : ML model
3. Experiments
4. Some techniques & next steps

1. Introduction & Problem description

Introduction
Amadeus IT Group

- Amadeus IT Group is a major IT provider for the global travel and tourism industry
- Two major components :
  - ***Global distribution system*** provides search, pricing, booking, ticketing and other processing services in real-time to travel providers and travel agencies
  - ***Information and technology business*** Offers computer software that automates processes : reservations, departure control, …etc.



Amadeus Business model

- *“**Powering better journeys through travel technology**”*

Introduction

Flight Search Process

Client Request :

NCE □ NYC, 02/09/20

Route exploration

NCE – PAR – NYC
NCE – LON – NYC

Route Selection:

NCE – PAR – NYC
NCE – LON – NYC

Eliminated Routes

Adding Airlines

NCE – PAR – NYC, AF-AF, 20h00
NCE – LON – NYC, AF-BA, 08h00
NCE – LON – NYC, BA-BA, 18h00
…
…
…

Check Availability

NCE – PAR – NYC, AF-AF, 20h00
NCE – LON – NYC, AF-BA, 08h00

Pricing

NCE – PAR – NYC, AF-AF, 20h00, 500$
NCE – LON – NYC, AF-BA, 08h00, 550$

Introduction

Flight Search complexity

Flight search is a complexe task :

Example Boston – San Francisco

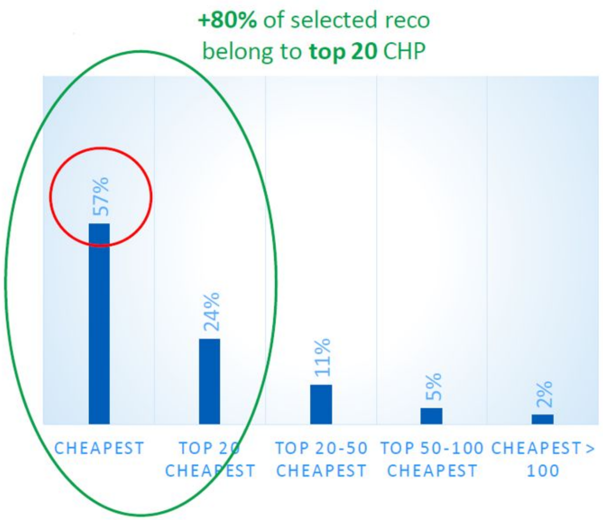
- Number of outbound flights considered (BOS-SFO): **403**
- Number of return flights considered (BOS-SFO): **402**
- Total number of journey itineraries: 403 x 402 = **162 006**
- Average number of ways to overlay fares onto each journey itinerary: **14 749**
- **~10 000 paths** are normalized by removing unworkable itineraries

Introduction
Overview

Focus on Findability !

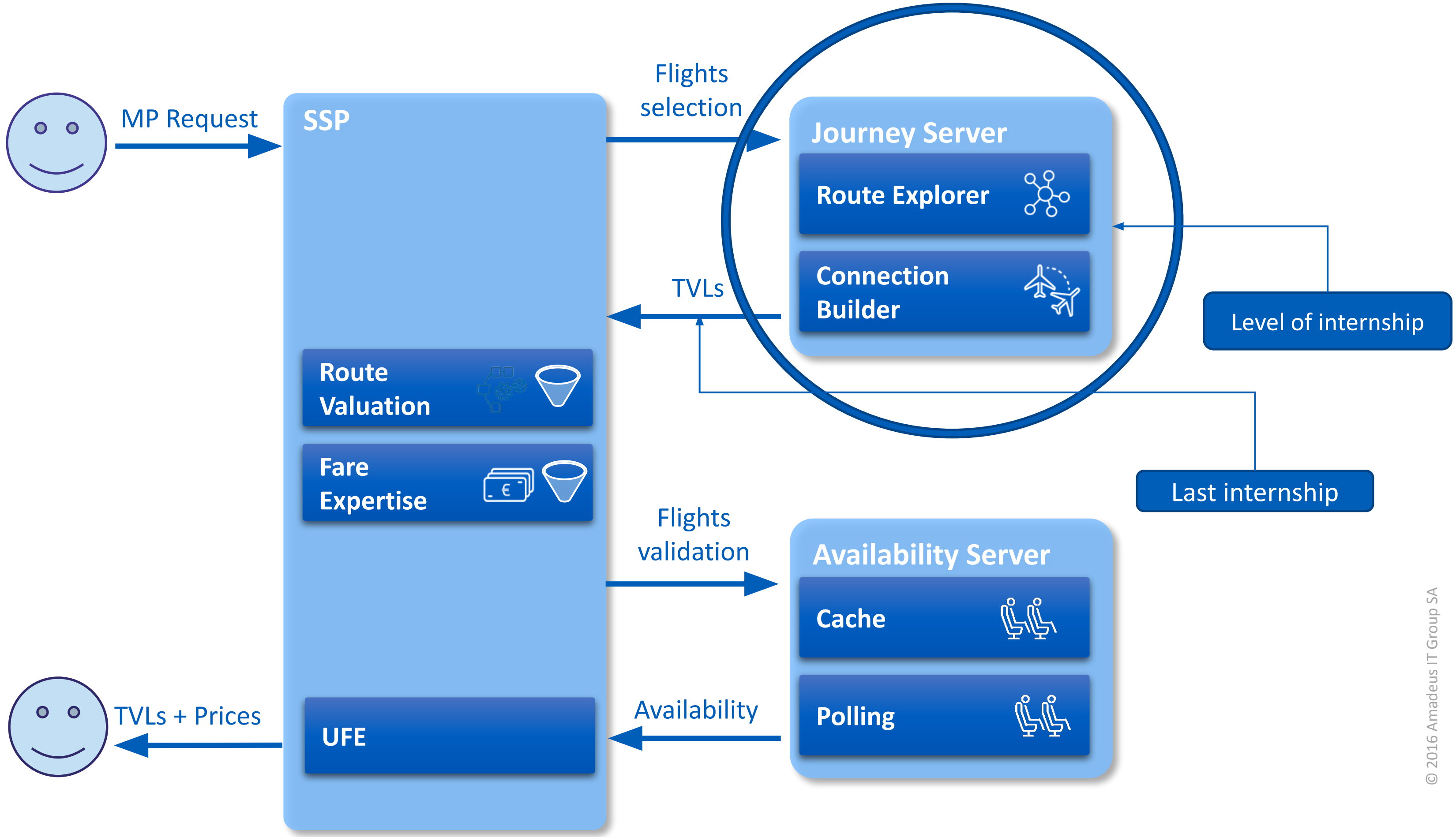
- How many time do people choose the cheapest price?

<u>What is findability ?</u>
The percentage of times when Amadeus returns the cheaper travel solution compared to its competitors



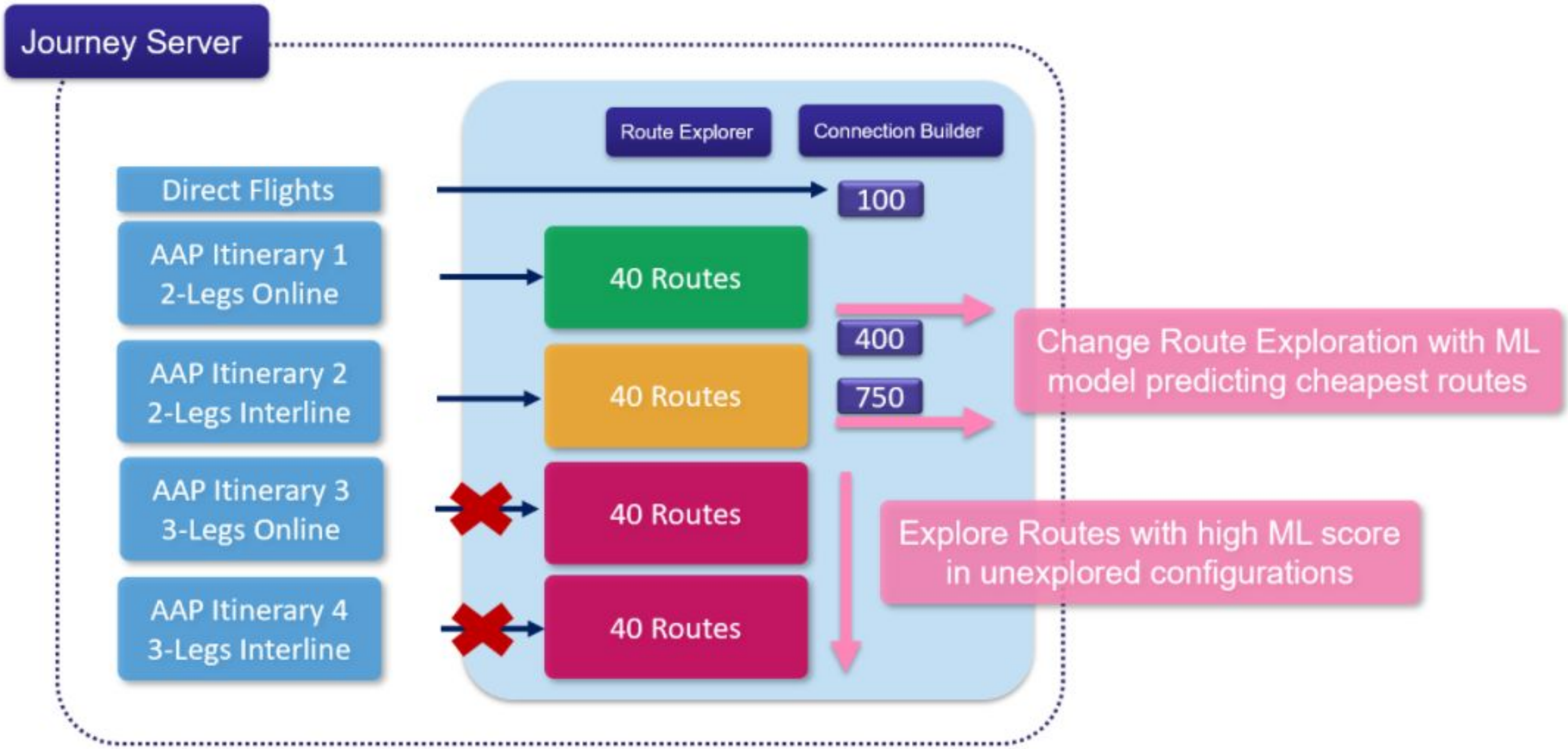
- Critical metric : 80% clients choose top 20 cheapest flights.
- ***<u>Are machine learning tools intelligent enough to help reducing the complexity of flight search w.r.t the findability metric ?</u>*** ===> this internship

Problem description
FSDA architecture



Fare Search Dynamic Architecture

Problem description
Machine Learning on Journey Server



Problem description

Machine Learning scheme

Setup

Route + other information :

origin, connection1, connection2, destination, dates, distance, …

→ **ML model** → **Price/Class**

Problem description
Machine Learning scheme

Setup

<u>Route + other information :</u>

origin, connection1, connection2, destination, dates, distance, …

→ **ML model** → **Price/Class**

Regression vs Classification

- Regression : predict the price
- Classification : predict the class (cheapest / non cheapest)

Problem description
Machine Learning scheme

Setup

Route + other information :

origin, connection1, connection2, destination, dates, distance, …

→ **ML model** → **Price/Class**

Regression vs Classification

_ Regression : predict the price

_ Classification : predict the class (cheapest / non cheapest)

Price is tricky !

_ Example : NICE – LONDON, 20th July 2021 one way direct:

- Possibility 1 : price = 500 $
- Possibility 2 : price = 600 $
- Possibility 3 : price = 900 $
- === > price (NICE – LONDON) = 500

_ Price ratio : imagine we found also for one way with one leg :

- Possibility NICE – PARIS – LONDON : price = 800$
- Possibility NICE – PARIS – LONDON : price = 750$
- === > price ratio (NICE – LONDON) = 500/500
- === > price ratio (NICE – PARIS – LONDON) = 750/500

Problem description
Machine Learning scheme

Setup

Route + other information :

origin, connection1, connection2, destination, dates, distance, …

→ **ML model** → **Price/Class**

Regression vs Classification

_ Regression : predict the price

_ Classification : predict the class (cheapest / non cheapest)

Classification vs Regression

_ Classification :

- Price ratio ≥ 1,05 === > non cheapest
- Price ratio ≤ 1,05 === > cheapest

_ Regression much richer than classification :

- Classification : two values vs Regression : many values
- Predicting perfectly price will lead to extremely accurate ranking of cheapest routes in production
- From last internship : Improving regression === > Improving findability

2. Solution design : ML model

Solution design
Machine Learning

_Challenge :

- A lot of categorical variables with high cardinality
- 72% of routes are new in test set ==> Algorithm with **high generalization capabilities**

Potential algorithms

Support vector machines	Tree algorithms	Deep Learning
compute intense	partition very finely the state of each variable	can handle sentences
cannot handle sentences	cannot handle sentences	can approximate any l^1 function
cannot capture the context where a city appears	cannot capture the context where a city appears	high generalization capabilities
Not considered	In production	Research

Solution design
Embeddings For Generalization?

- Routes $R_i = [origin, connection1, connection2, destination]$.

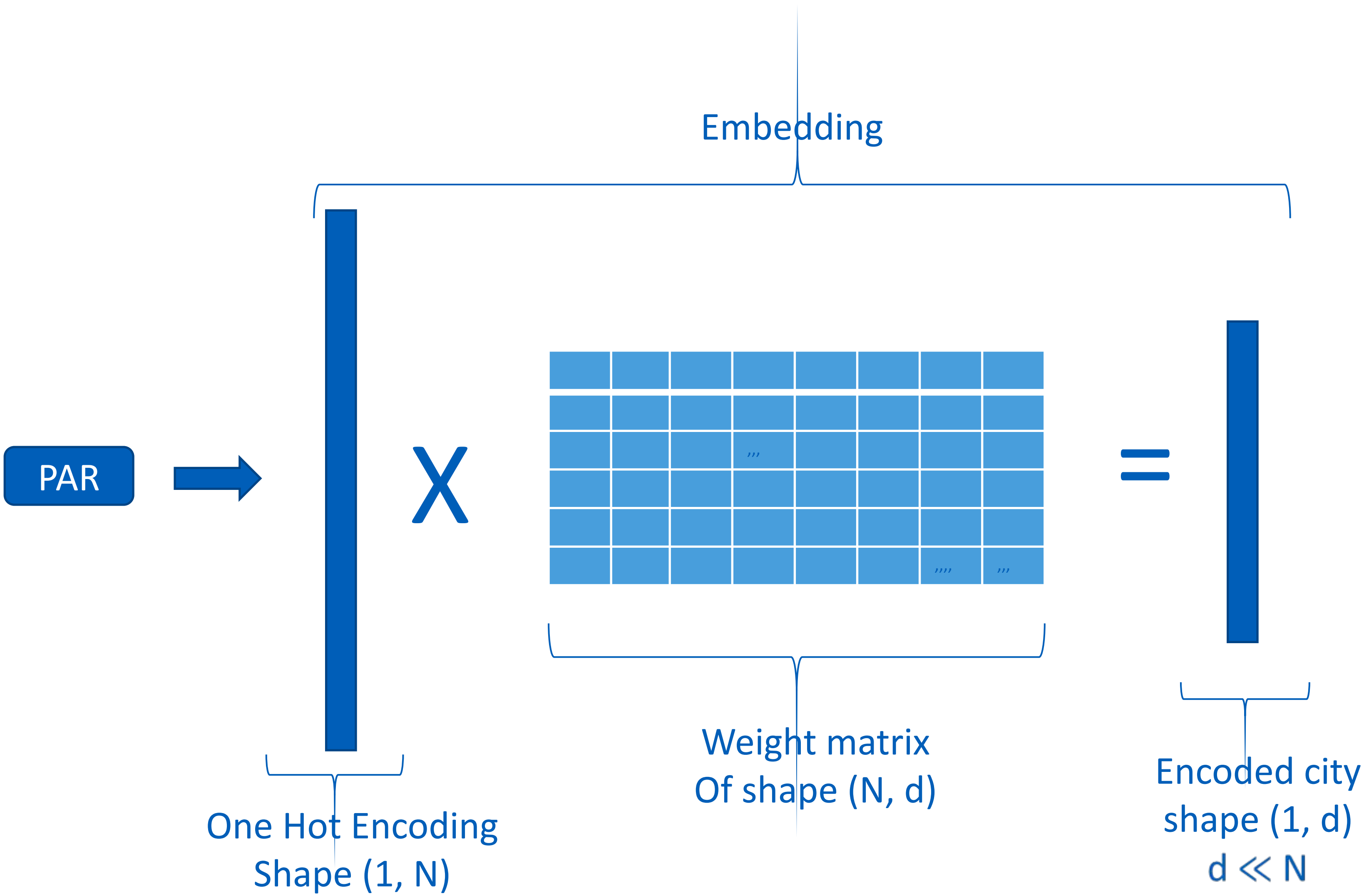
ML Model

- Training : R_1, R_2, R_3, R_4
- Test : R_5

- **Traditional ML models :** The route R_5 is unseen → they can't generalize for it.
- **Embeddings :** The route R_5 is unseen but **very similar** to R_2 → the same behavior that model learned for R_2 will be adopted for R_5.

Similar in terms of their influence on the price

Solution design
What are embeddings



- We have just to learn this weight matrix !

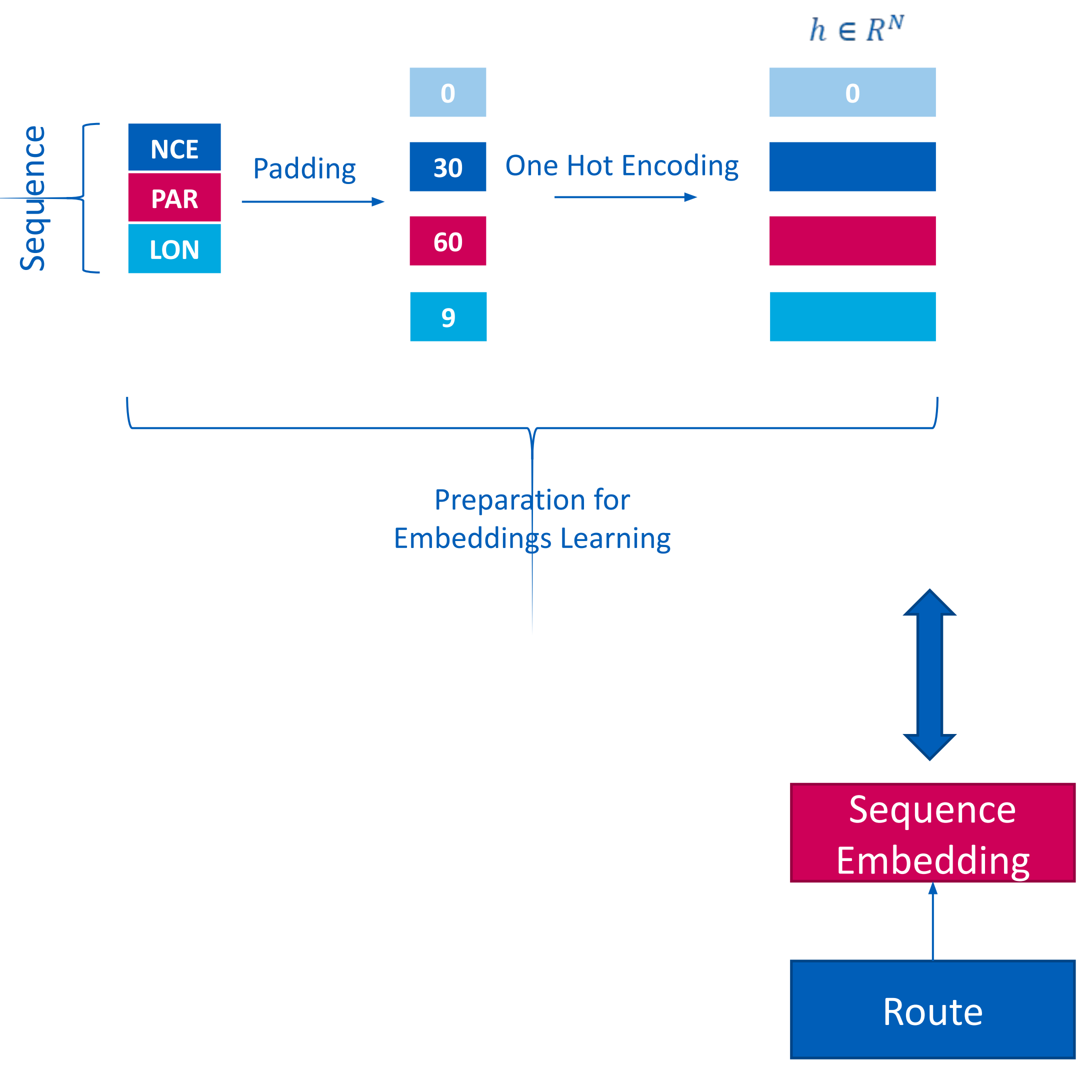
Solution design
Good / Bad points

- Learning the weights might done :
  - Supervised way : low representations are learned with respect to the target (suitable case)
  - Unsupervised way : with deep learning also (auto-encoders)

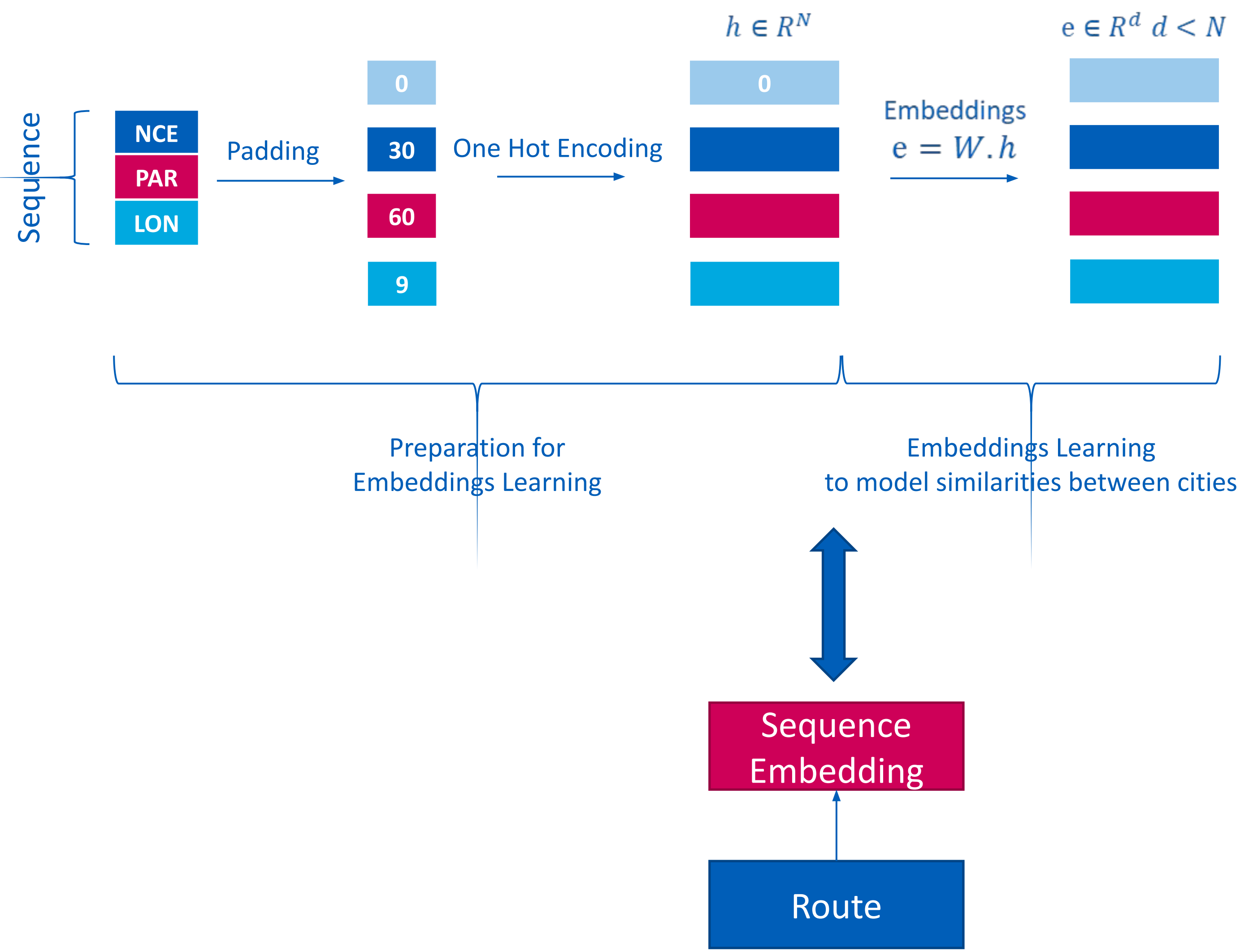
- New representations :
  - Continuous == > Good for deep nets models

- BUT !
  - Cannot directly handle sentences
  - Work with **<u>Route</u>** : PAR – NCE – BOS – NYC instead of separate categorical variables

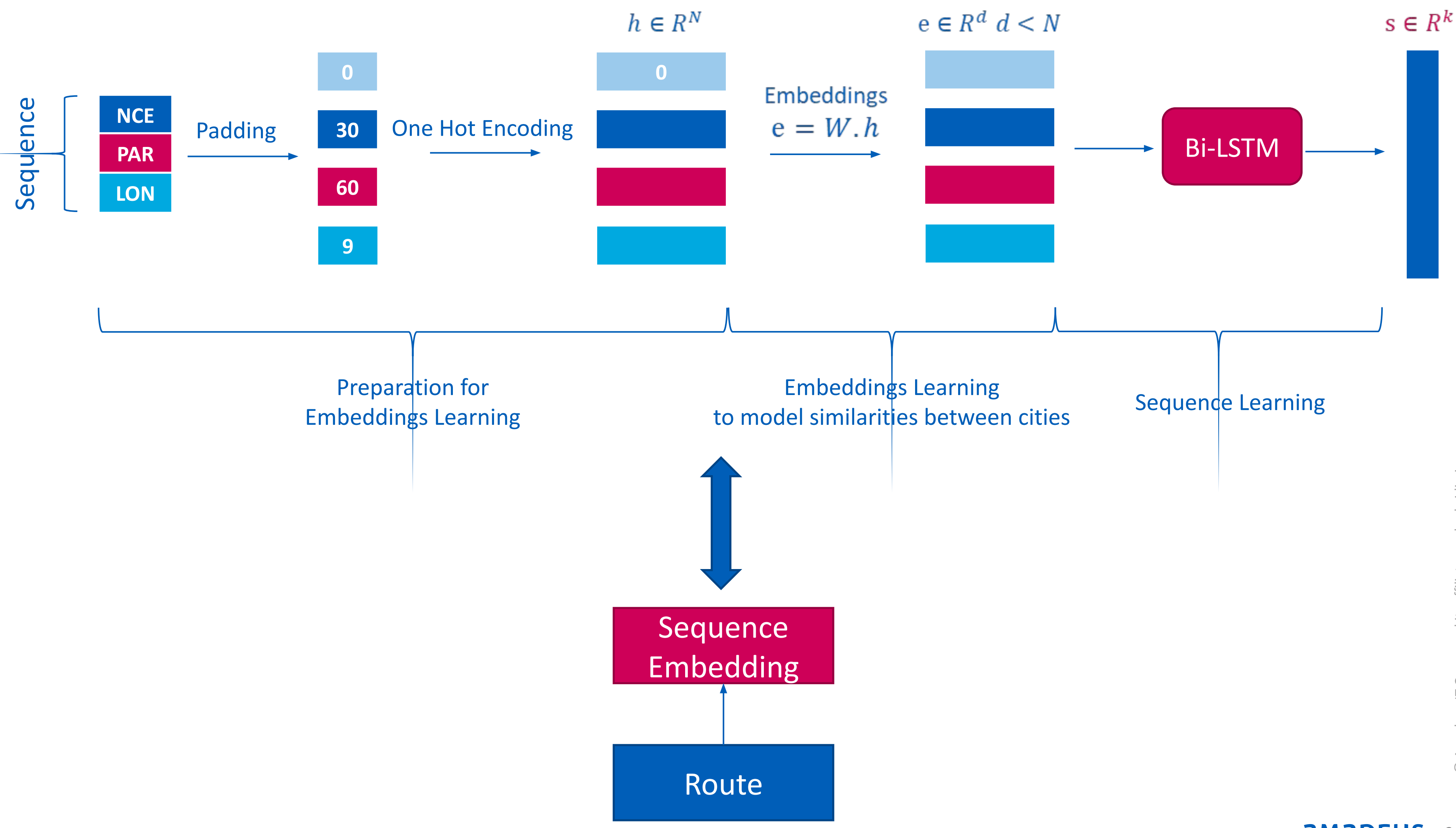
Solution design
Sequence Embeddings



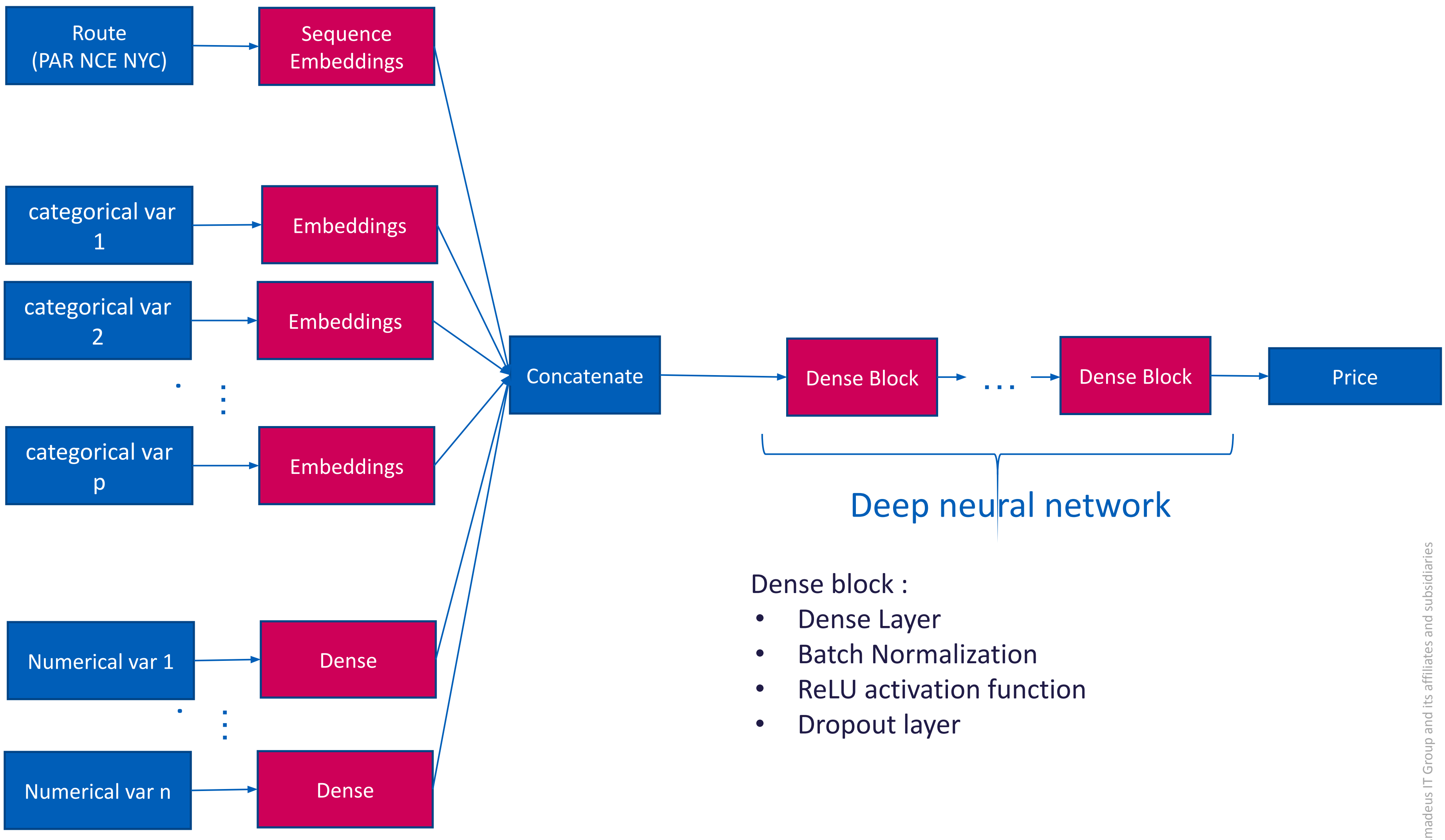
Solution design
Zoom in on Sequence Embeddings



Solution design
Zoom in on Sequence Embeddings



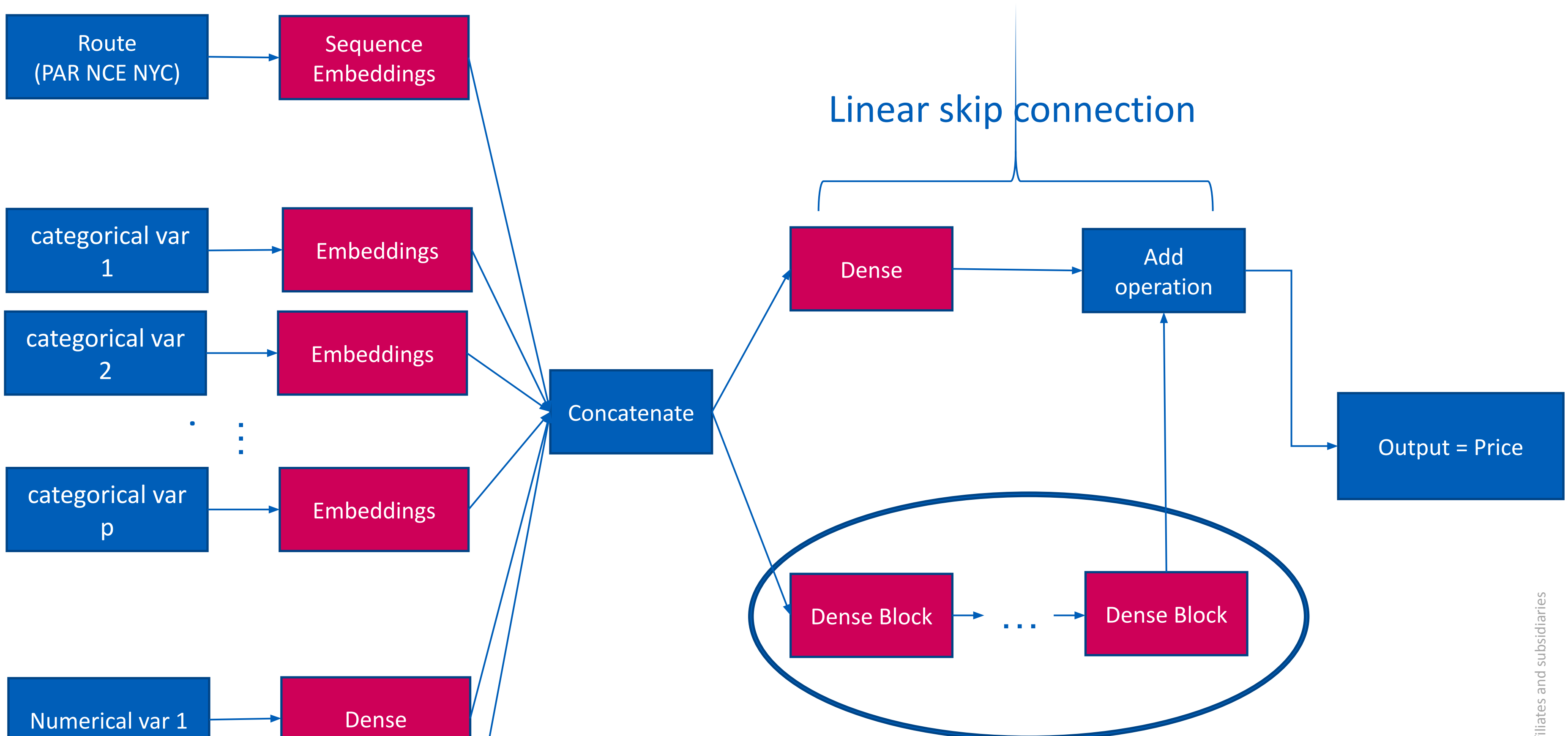
Solution design
Deep learning model



Dense block :

- Dense Layer
- Batch Normalization
- ReLU activation function
- Dropout layer

Solution design
Deep learning model



Dense block :

- Dense Layer
- Batch Normalization
- ReLU activation function
- Dropout layer

Solution design
Deep learning model

_ Remain to do from last year :

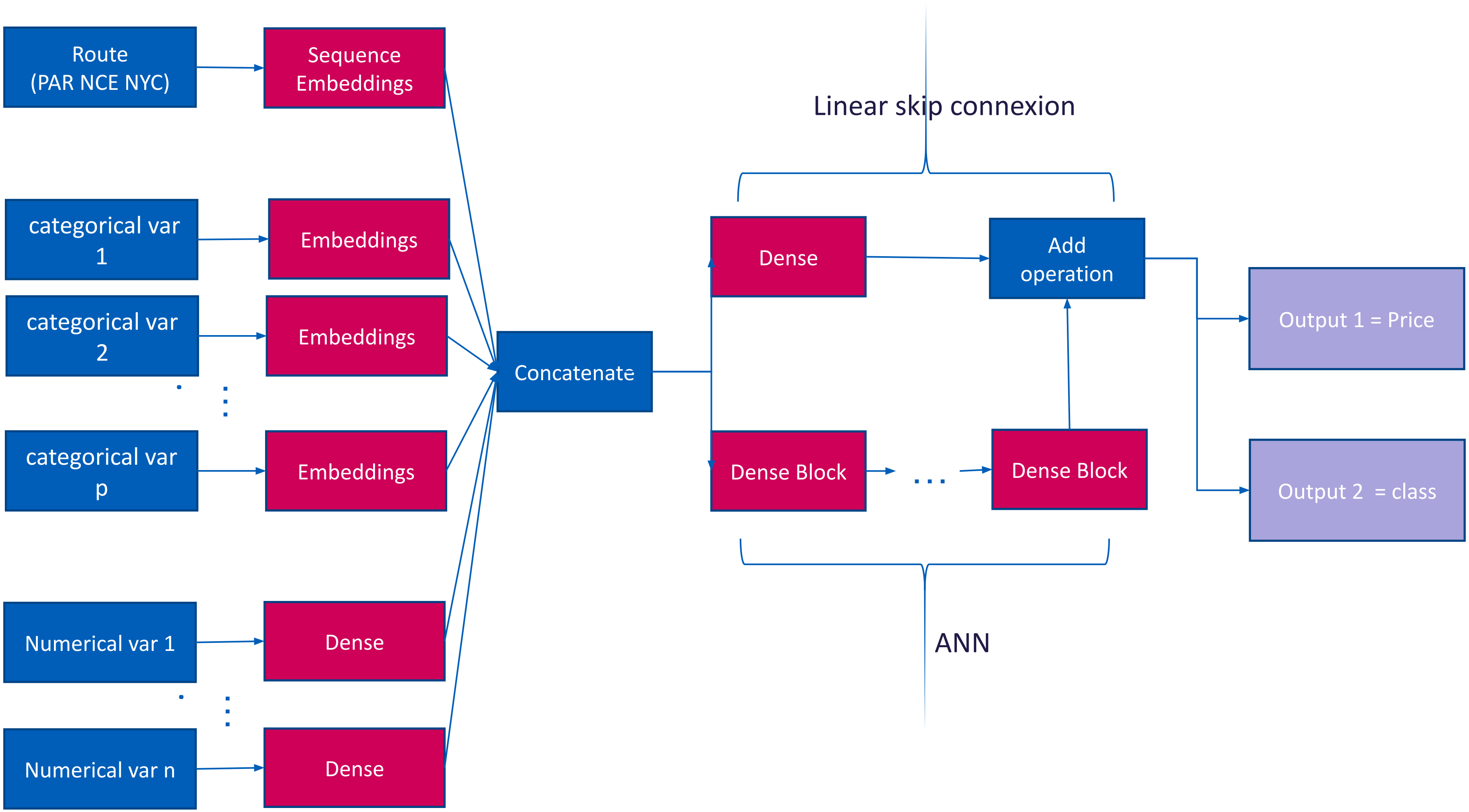
- GBM vs DL ?
- How much DL is robust to data distributions shift ?
- Feature importance ?
- How can the model adapt to a different env with less features ?

_ New developement :

- Answer previous questions
- Understand the role of encoding
- Find a way to improve regression and classification
- Suggest new modifications to the model

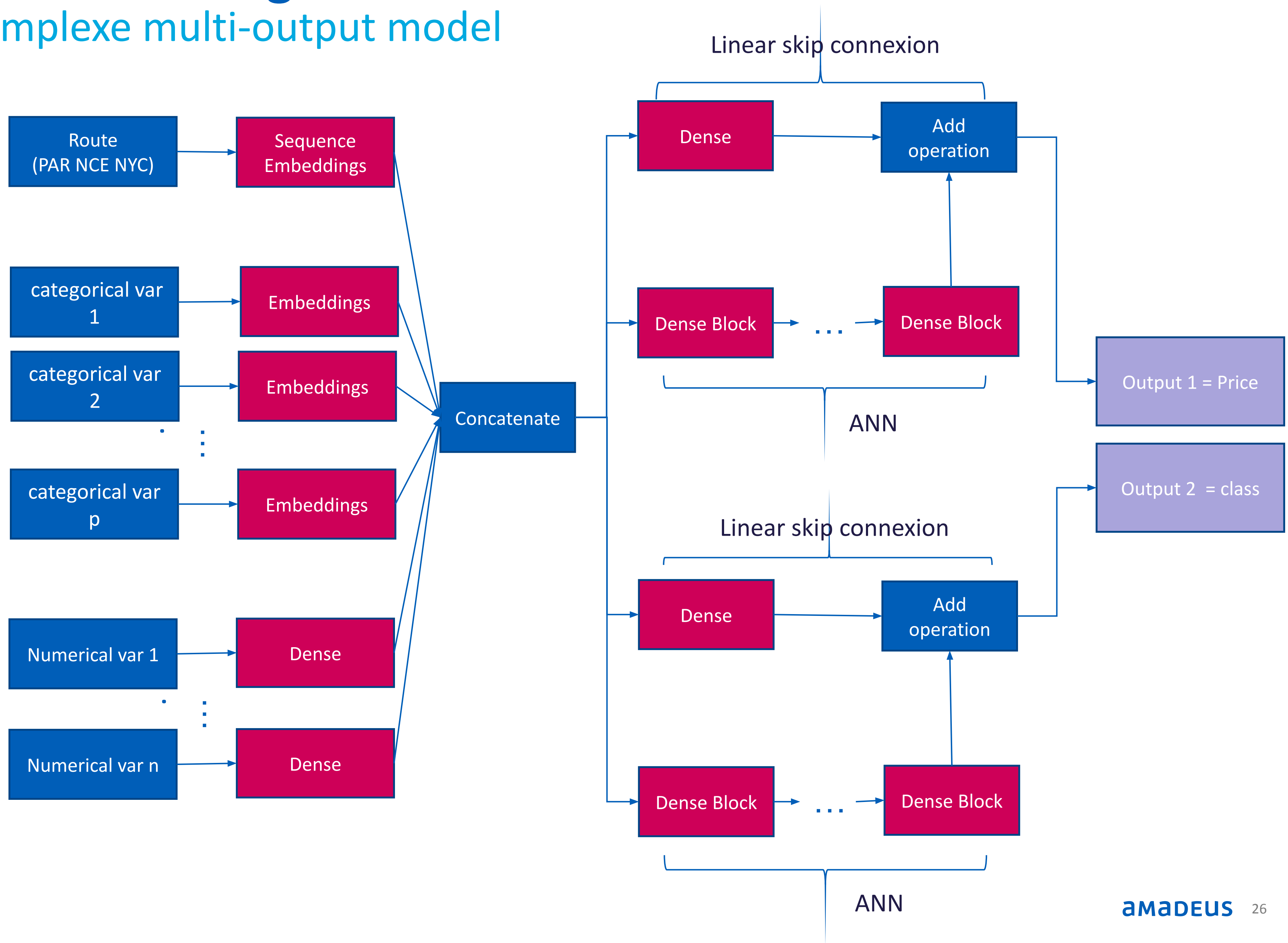
Solution design

Simple multi-output model



Solution design

Complexe multi-output model



3. Experiments

Experiments
Data Collection

_Overview :

- **30 million** travel solutions with more than **4 million *OneWay*** flights.
- A large number of travel solutions from different countries and continents (Europe, America, Asia, …)

_Features :

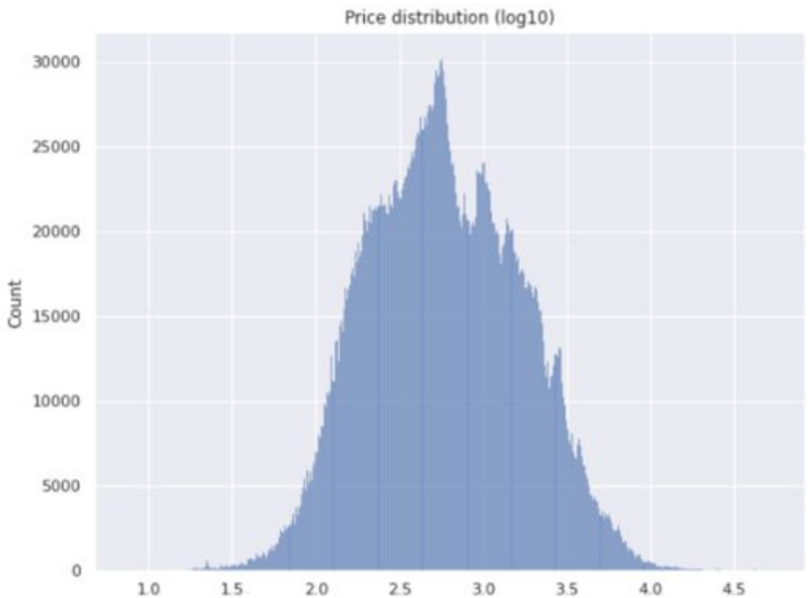
- Many categorical variables with high cardinality (more than cities of origin and destination , and connections, 400 office cities, …).
- Dates variables that we encode using cyclic features encoding techniques.
- Continuous variables like flight time and distance.

_Targets :

- **Price :** The continuous variable target is skewed !
- === > **log price** as target !

Experiments
Target variable for regression

Log



Experiments
Training setup

_Minimization problem :

- Loss function : MAE

$$\mathcal{L}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|$$

_Constraints :

- Focus on ***cheapest class*** of routes

- The trick of weighting the loss :

$$\mathcal{L}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} w_i |y_i - \hat{y}_i| \qquad \omega_i = \frac{1}{\text{price\_ratio}_i}$$

- Possible weighting schemes :

$$\omega_i \propto e^{-(\text{price\_ratio}_i - 1.0)}, \qquad \omega_i \propto \frac{1}{\text{price\_ratio}_i^{\alpha}}$$

Experiments
Evaluation metrics

Evaluation metrics

Internal performance

Online performance

Regression

Classfication

Ranking

- $$\text{R2 score} = 1 - \frac{\sum_{i=1}^{i=n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{i=n} (y_i - \bar{y}_i)^2}$$
- $$\text{MAPE} = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} \frac{|y_i - \hat{y}_i|}{y_i}$$
- $$\text{RMSE} = \sqrt{\frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} (y_i - \hat{y}_i)^2}$$

- $$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{True positive}}{\text{Total actual positive}}$$
- $$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{True positive}}{\text{Total predicted positive}}$$
- Area under the Precision-Recall curve

- findability area under the curve
- $$\text{MAP} = \frac{1}{n} \sum_{k=1}^{n} \text{AP}_k$$
- $$\text{DCG}_n = \sum_{k=1}^{n} \frac{\text{rel}_k}{\log_2(k+1)}$$

AP_k : average precision of class k
n : number of classes
rel_k : relevance of class k

Experiments
Benchmarking results

Model in production

New models

Models	type	Rank cheapest avg	f_auc	nDCG	MAP
GBM_old	Classification	1,96	0,972	0,933	0,760
GBM_new	Classification	1,96	0,972	0,933	0,760
Deep Learning	Regression	2,01	0,969	0,959	0,707
Multi-output model (simple)	Regression	2,25	0,970	0,951	0,703
	Classification	1,93	0,973	0,945	0,734
Multi-output model (complexe)	Regression	2,15	0,971	0,953	0,718
	Classification	2,0	0,972	0,933	0,733
Deep Learning + weighted Loss	Regression	1,85	0,973	0,958	0,737

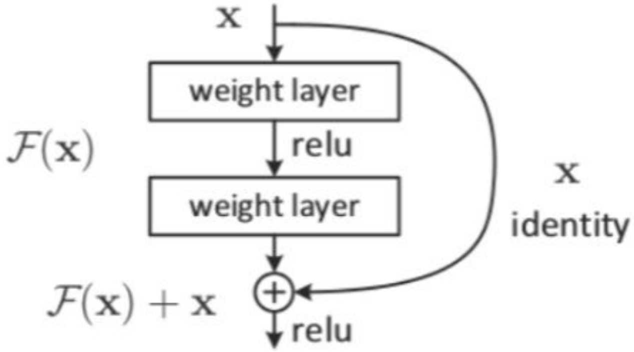
TOP 3 models so far :

- Regression DL model + weighted loss
- Multi-output DL model (classification part)
- GBM classification model

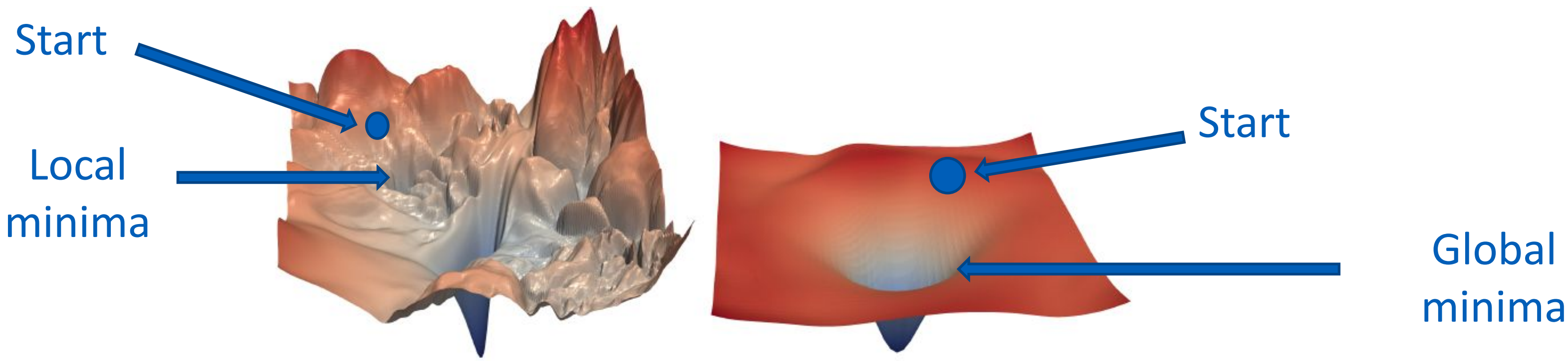
4. Some techniques

Some techniques

Skip connection layer (ref [0, 1])



Linear skip connection



Without Skip-connection

With Skip-connection

ResNet-56 Loss by Microsoft with and without skip-connection

Some techniques
Learning rate dynamics

- We've used TensorFlow gradient-based training to fit the model to the data

- Gradient update equation (gradient step) :

Learning rate

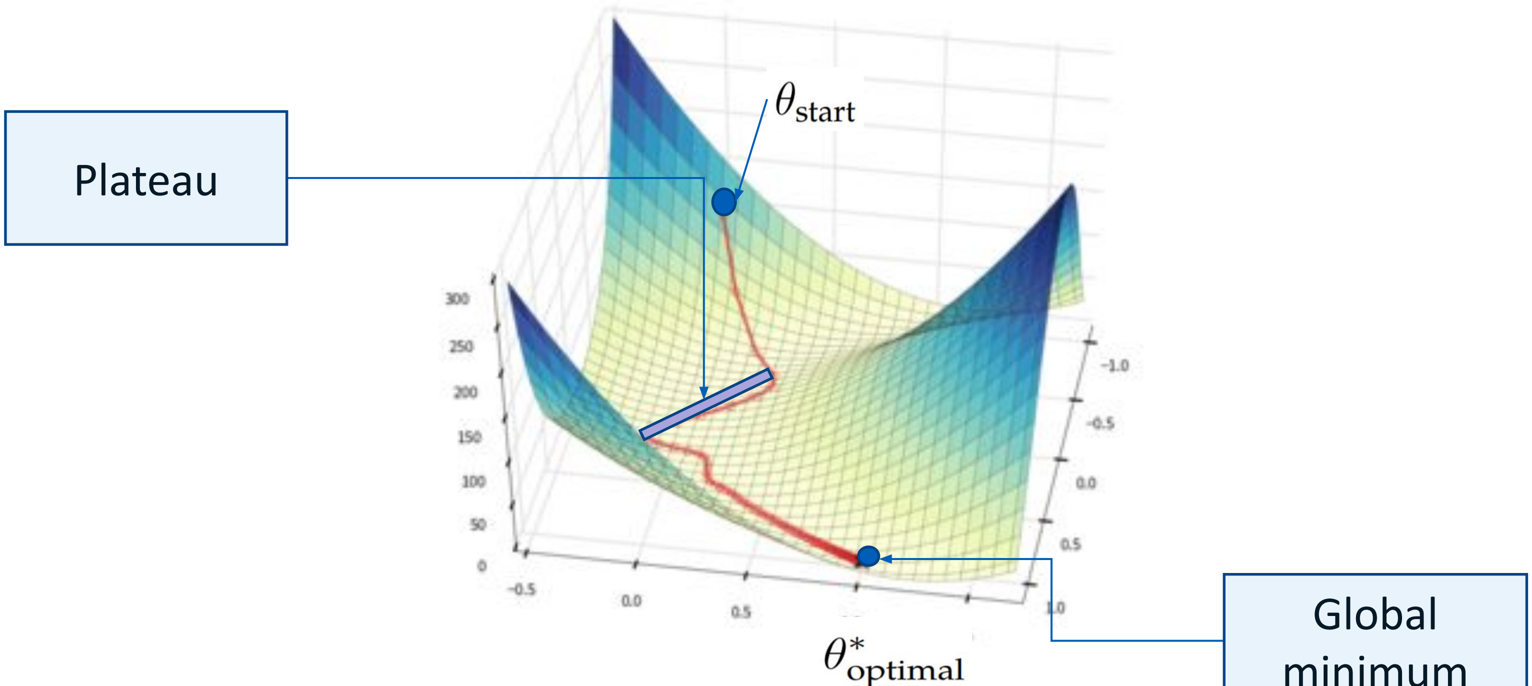
Params of the model

$$\theta_{t+1} = \theta_t - \eta_t \nabla \mathcal{L}(\theta_t)$$

Gradient of the loss

- Usually, the **learning rate is kept constant**
- But ! Loss landscape is not convex, nor concave !

Some techniques
Learning rate dynamics (ref [2])

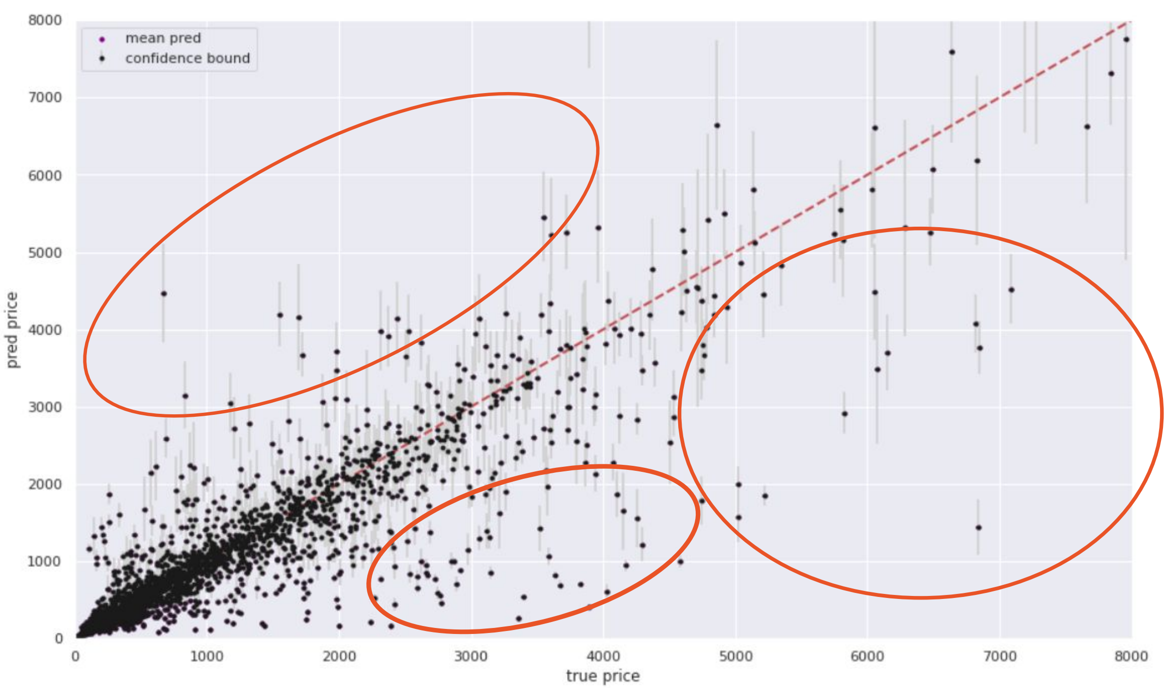


- Theory :
  - $\eta_t = \frac{\tau \eta_0}{\min(\tau, t^{\alpha})}$ (τ=20 for example, Francis Bach, Eric Moulines : choose $\alpha \leq 1$ for convex case)
  - But ! In a plateau need a push to escape !
- Practice :
  - Take the best of theory (previous update)
  - Increase learning rate in each plateau encountred

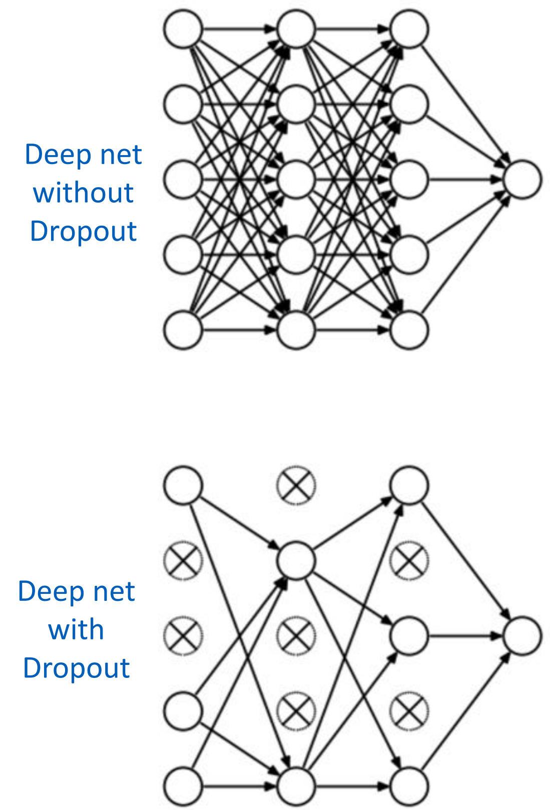
Some techniques

Confidence bounds of the predictions with dropout (ref [3,4])

- Initially dropout was introduced to overcome the problem of overfitting
- Recent research show it can be used to give the model's confidence



Actual price vs **mean predicted** price with bounds$\pm$std



Some techniques

Interpretability of machine learning model (ref [5,6])

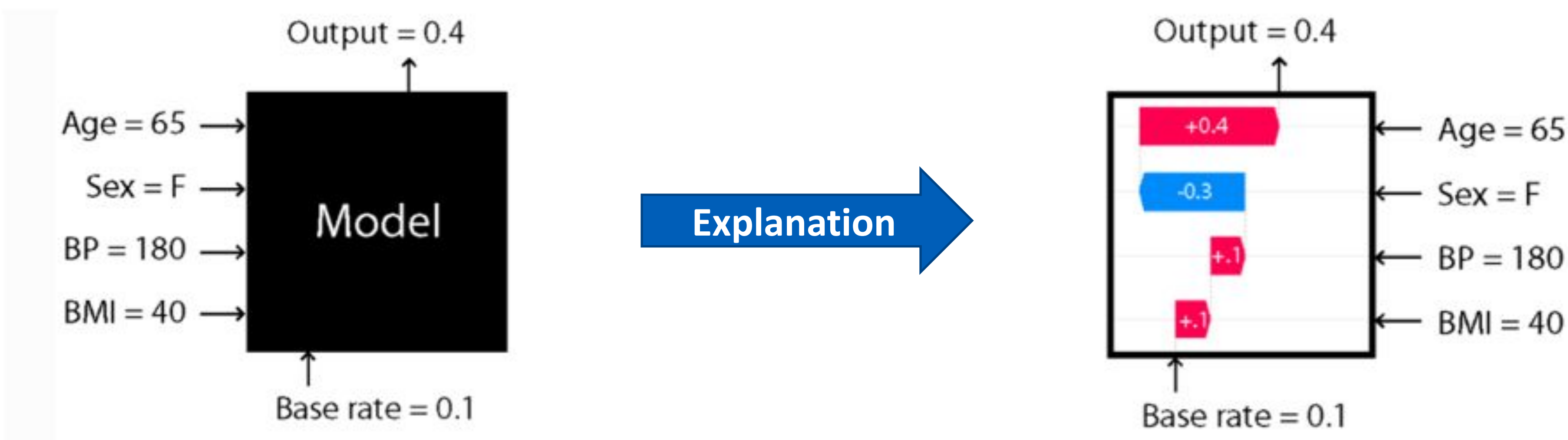


Figure taken from SHAP documentation page

Some techniques

Interpretability (ref [5,6])

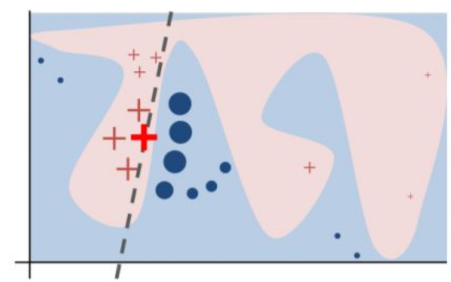
_ PFI

- **P**emutation **F**eature **I**mportance
- Get an idea about how much the model is robust w.r.t to a shift of each feature values.

182	155
175	147
…	…
156	142
153	130

_ LIME

- **L**ocal **I**nterpretable **M**odel-Agnostic **E**xplanations
- Construct linear regression model as a local approximation of the model
- Explain local predictions



$$\xi(x) = \underset{g \in \mathcal{G}}{\arg\min}\, \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

_ SHAP

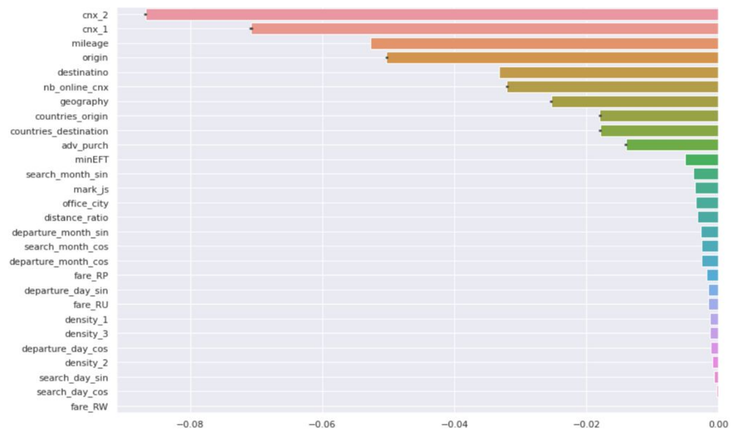
- **SH**apley **A**dditive ex**P**lanations
- Distribute fairly the outcome (prediction) between the players (features) in a coalitional game.
- Explain local and batch of predictions

$$\phi_i := \frac{1}{M} \sum_{S \subseteq F \setminus \{i\}} \frac{1}{\binom{M-1}{|S|}} C(i|S)$$

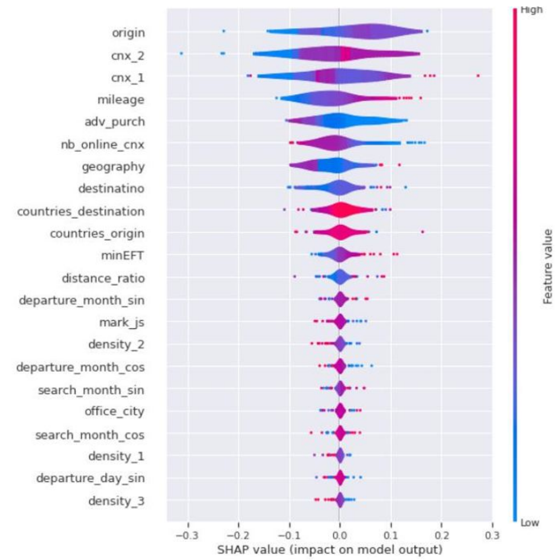
$$f(x) - \mathbb{E}_{y \sim \mathcal{D}}[f(y)] = \sum_{i=1}^{M} \phi_i$$

Some techniques

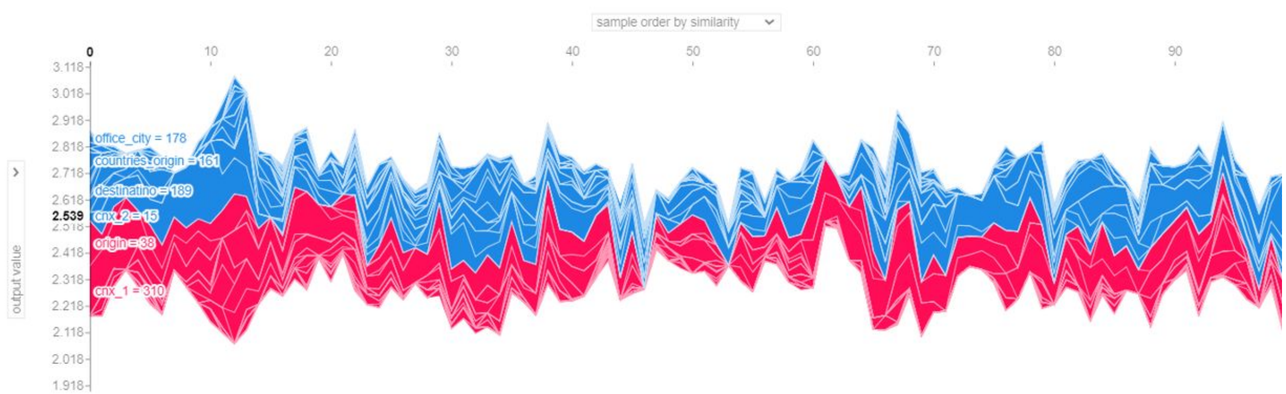
Interpretability



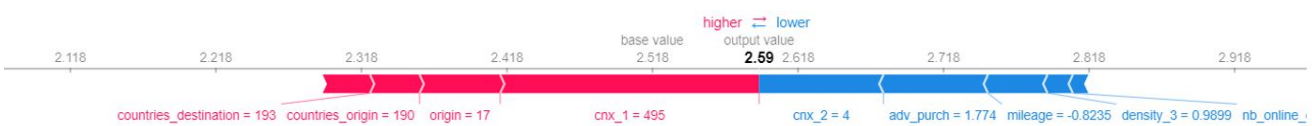
Permutation feature importance results



Shap feature impotance results



Batch of predictions explanation



Single prediction explanation

Next steps
challenges

- Loss function :

$$\mathcal{L}_{\text{total}} = \beta \mathcal{L}_{\text{regression}} + (1 - \beta)\mathcal{L}_{\text{classification}}$$

- Challenges :
  - How to choose β ?
  - Highly imbalanced data set !
  - Unclear encoding : are the low representations learned w.r.t price better/worser than those w.r.t class ?

References

- [0] : Hao Li et al. "***Visualizing the Loss Landscape of Neural Nets***", Arxiv
- [1] : Kaiming He, Xiangyu Zhang, Jian Sun "***Deep Residual Learning for Image Recognition***", Arxiv
- [2] : Yoshua Bengio "***Practical recommendations for gradient-based training of deep architectures***", Arxiv
- [3] : Yarin Gal, Zoubin Ghahramani, "***Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning***", Arxiv
- [4] : Geoffrey Hinton "***Dropout: A Simple Way to Prevent Neural Networks from Over-fitting***", JMLR
- [5] : Christoph Molnar, "***Interpretable Machine Learning***", online Book
- [6] : Marco Tulio Ribeiro "***Why should I trust you ? Explaining the Predictions of Any Classifier*** ", Arxiv

Thank you !