Séries temporelles
Méthodes ML et DL
DL / Probabilistic Forecasting

Intervenant
Guillaume Hochard

6.1. Introduction

De la prévision ponctuelle à la prévision probabiliste

La prévision ponctuelle est la prévision de la valeur d'une variable aléatoire y_t à l'horizon h et conditionnelle à l'information fixée sur une période de référence T

On cherche donc à prédire le comportement futur d'une série temporelle y_i basé sur son passé :

$$y_{i,0}, y_{i,1}, y_{i,2}, \cdots, y_{i,T-1} \Rightarrow y_{i,T}, y_{i,T+1}, \cdots, y_{i,T+h}$$

6.1. Introduction

De la prévision ponctuelle à la prévision probabiliste

La prévision probabiliste est la prévision d'un intervalle de confiance sur la valeur d'une variable aléatoire y_t à l'horizon h et conditionnelle à l'information fixée sur une période de référence T

On cherche donc à prédire la probabilité que le comportement futur d'une série temporelle y_i se trouve dans un certain intervalle :

$$y_{i,0}, y_{i,1}, y_{i,2}, \dots, y_{i,T-1} \Rightarrow P(y_{i,T}, y_{i,T+1}, \dots, y_{i,T+h})$$

6.1. Introduction

Motivations

- Palier aux problèmes d'asymétries : dans la maintenance aéronautique par exemple, avoir une visse supplémentaire ou une visse manquante n'a pas le même impact financier.

- Les décisions sont fonction des extrêmes : il faut avoir assez de produits afin de satisfaire ses clients; il ne faut pas avoir trop de produits périssables au risque de les jeter.

Cas de figure

- Comportements erratiques (ventes de produits de mode, e-commerce, etc.)
- Cold-start : absence d'historique de données (lancement nouveau produit)
- Évènements rares (maintenance, etc.)

6.1. Introduction

Modélisation

Objectif : modéliser la distribution du futur de chaque série temporelle conditionnée par son passé et par les variables exogènes pertinentes et connues à tout instant.

$$P(y_{i,t_c:T} \mid y_{i,1:t_{c-1}}, x_{1:T})$$

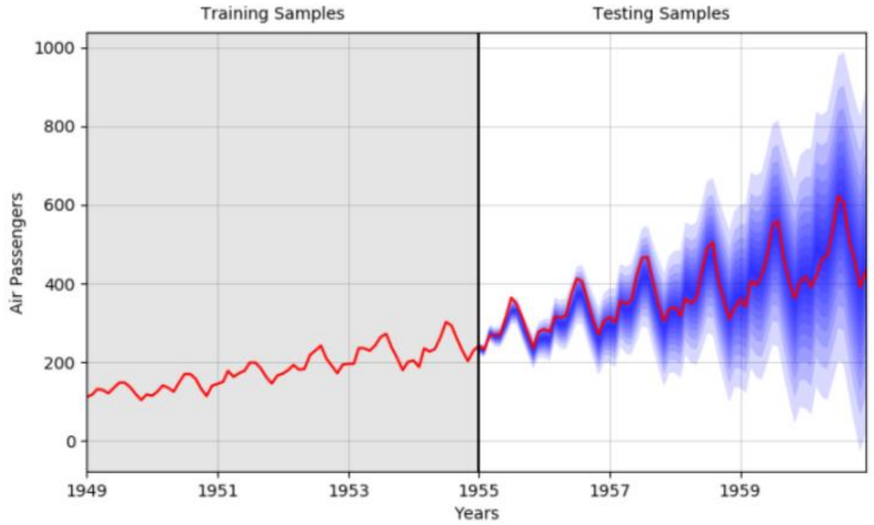
$y_{i,t}$: série temporelle
$x_{i,t}$: variables connues à chaque instant
t_c : temps à partir duquel la série temporelle $y_{i,t}$ est inconnue
T : horizon de prédiction

6.1. Introduction

Intervalle de prediction

Un interval de prediction I_{t+k}^{β} produit à l'instant t pour un futur t+k est définit par ses bornes inférieure et supérieure, qui sont les forecasts quantiles :

$$I_{t+k}^{\beta} = \left[\hat{q}_{t+\mathrm{k}}^{\alpha\_lower} ; \hat{q}_{t+\mathrm{k}}^{\alpha\_upper} \right]$$



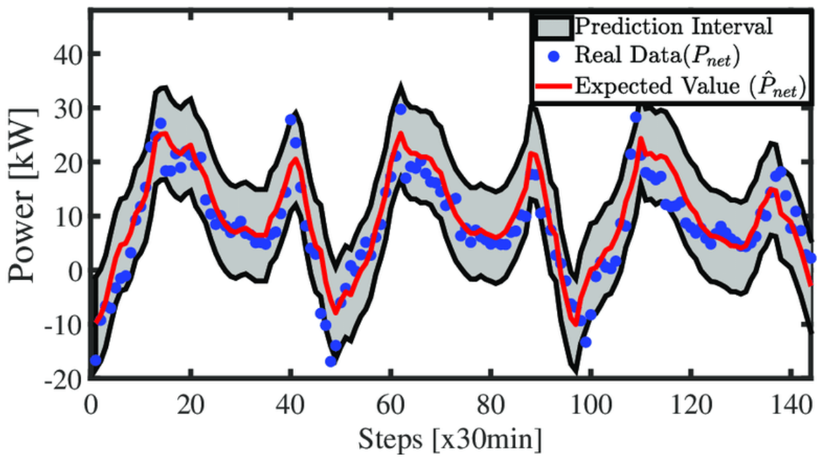
6.1. Introduction

Métriques de performance

- **Prediction Interval Coverage Probability (PICP)**
- Indique le pourcentage de temps où un interval au quantile contient la valeur réelle associée à la prédicition

Performance Indices	Prediction Horizon		
	One Hour Ahead	Six Hours Ahead	One Day Ahead
RMSE (kW)	4.5136	5.0471	5.1974
MAE (kW)	3.2995	3.7316	3.7530
PINAW (%)	22.73	27.62	28.02
PICP (%)	88.22	89.79	89.83

$$PICP = \frac{1}{N_{test}} \sum_{i}^{N_{test}} c_i \quad \text{où} \quad c_i = \begin{cases} 1, t_i \in I^{\alpha}(x_i) \\ 0, t_i \notin I^{\alpha}(x_i) \end{cases}$$



6.1. Introduction

Métriques de performance

Reliability Score : mesure si les intervals de prédiction (PI) couvrent les observations

- **Prediction Interval Coverage Probability (PICP)**
- Indique le pourcentage de temps où un interval au quantile contient la valeur réelle associée à la prédicition

$$PICP = \frac{1}{N_{test}} \sum_{i}^{N_{test}} c_i \text{ où } c_i = \begin{cases} 1, t_i \in I^{\alpha}(x_i) \\ 0, t_i \notin I^{\alpha}(x_i) \end{cases}$$

- **Average coverage Error (ACE)**

$$ACE = |PICP - 100 * (1 - \alpha)|$$

6.1. Introduction

Métriques de performance

Sharpness Score : mesure la largeur des intervals de prédiction

- **Interval Score (IS)**

$$\text{IS} = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} (\hat{q}_t^{1-\alpha} - \hat{q}_t^{\alpha})$$

- Mean Interval Score (MIS),
- Mean Scaled Interval Score (MSIS)

- Autres métriques ponctuelles pouvant être utilisées :
- sMAPE, MASE (ex: métriques de la compétion M4)

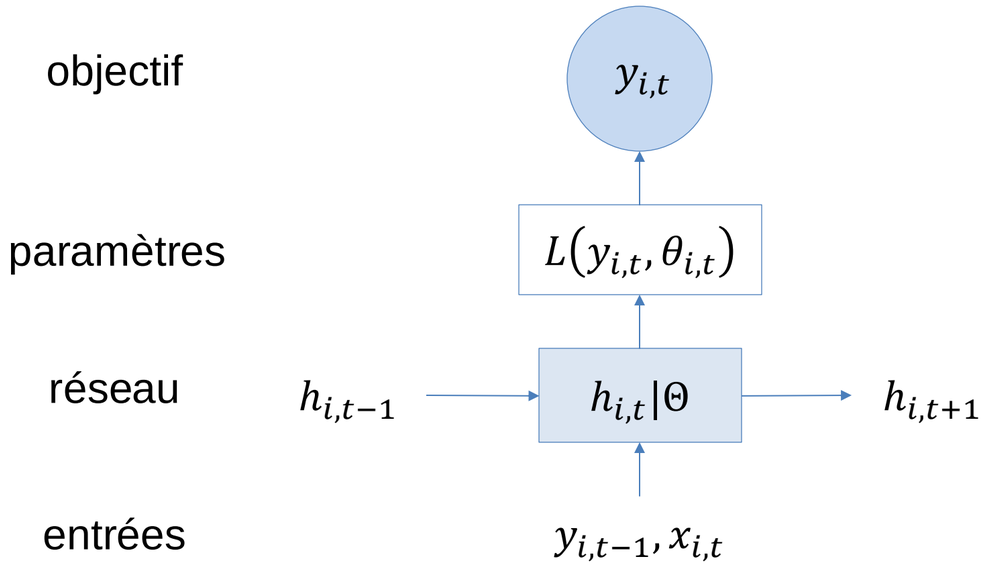
6.2 DeepAR

[1] Salinas, D., Flunkert, V., & Gasthaus, J. (2017). *DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks*. Retrieved from http://arxiv.org/abs/1704.04110

DeepAR [1] est un algorithme qui permet d'obtenir un modèle paramétrique global (1 modèle pour toutes les séries temporelles) de prévision basé sur un réseau de neurones autorégressif et récurrent, incorpore une fonction de vraisemblance afin de produire une prévision probabiliste.

Architecture

$$P(y_{i,t_c:T} \mid y_{i,1:t_{c-1}}, x_{1:T})$$



Objectif : entraîner les paramètres du modèle avec la fonction de vraisemblance associée à la loi choisie ($y_{i,t}$ est la valeur cible)

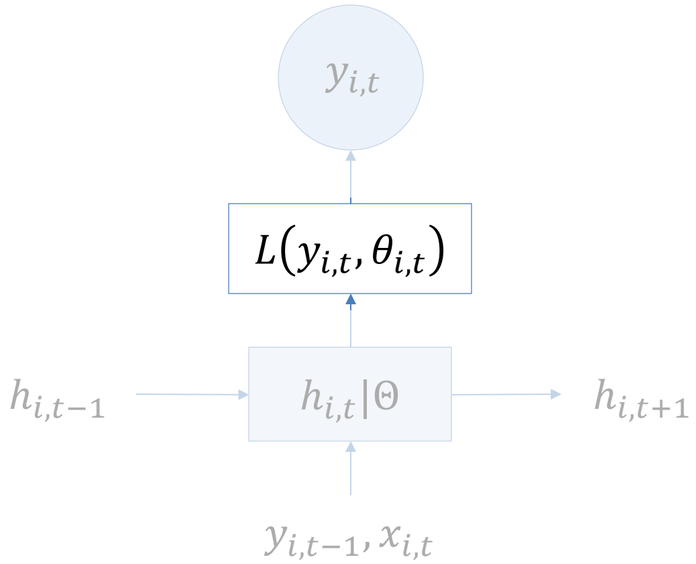
Paramètres : les paramètres $\theta_{i,t} = \theta(h_{i,t}, \Theta)$ qui correspondent sont estimé par transformation de la sortie du RNN $h_{i,t}$

Réseau : $h_{i,t-1}$ sortie du RNN au pas de temps précédent, $h_{i,t}$ sortie du RNN au pas de temps en cours, Θ ensemble des paramètres du RNN partagé $\forall \boldsymbol{i}, \boldsymbol{t}$

Entrées : y série temporelle au pas de temps précédent et $x_{i,t}$ features

6.2 DeepAR

Fonction de vraisemblance



- Fonction de vraisemblance gaussienne (pour les fonctions à valeurs réelles):

$$l_G(\mathrm{y}|\mu,\sigma) = (2\pi\sigma^2)^{-1/2} \cdot \exp\left(-\frac{(\mathrm{y}-\mu)^2}{2\sigma^2}\right)$$

$$\mu(h_{i,t}) = w_\mu^T \cdot h_{i,t} + b_\mu$$

$$\sigma(h_{i,t}) = \log(1 + \exp(w_\sigma^T \cdot h_{i,t} + b_\sigma))$$

- Fonction de vraisemblance binomiale négative (pour les variables de comptage positif):

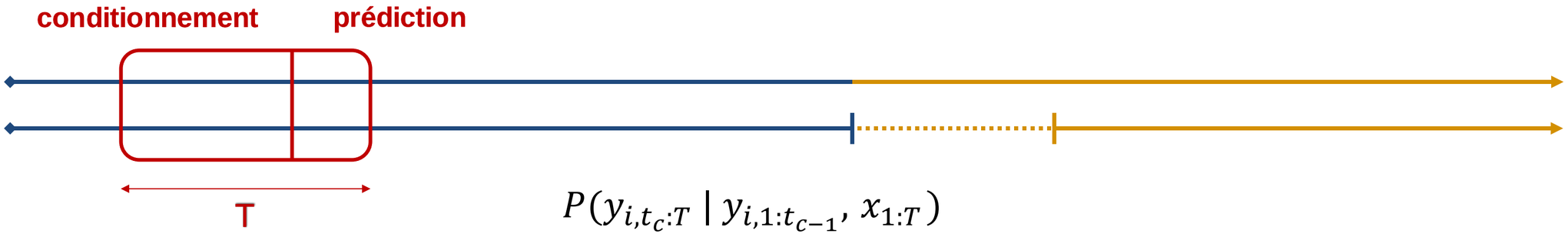
$$l_{BN}(\mathrm{y}|\mu,\alpha) = \frac{\Gamma(\mathrm{y}+1/\alpha)}{\Gamma(\mathrm{y}+1)\Gamma(1/\alpha)}\left(\frac{1}{1+\alpha\mu}\right)^{1/\alpha}\left(\frac{\alpha\mu}{1+\alpha\mu}\right)^{\mathrm{y}}$$

$$\mu(h_{i,t}) = \log(1 + \exp(w_\mu^T \cdot h_{i,t} + b_\mu))$$

$$\alpha(h_{i,t}) = \log(1 + \exp(w_\alpha^T \cdot h_{i,t} + b_\alpha))$$

6.2 DeepAR

Entrainement



- Fixer une fenêtre de longueur T
- Fixer le ratio conditionnement/prédiction
- S'assurer que la partie « prédiction » se trouve dans la zone où la série temporelle est connue
- On peut placer le point de départ hors de la zone dans laquelle la série temporelle est connue afin d'apprendre les comportements de cold-start

NB: Le temps absolu est seulement disponible pour le modèle grâce au features et non grâce aux positions relatives de $y_{i,t}$.

6.2 DeepAR

Prérequis

- Au moins 200 séries temporelles
- Séries temporelles générées par un même processus ou des processus similaires

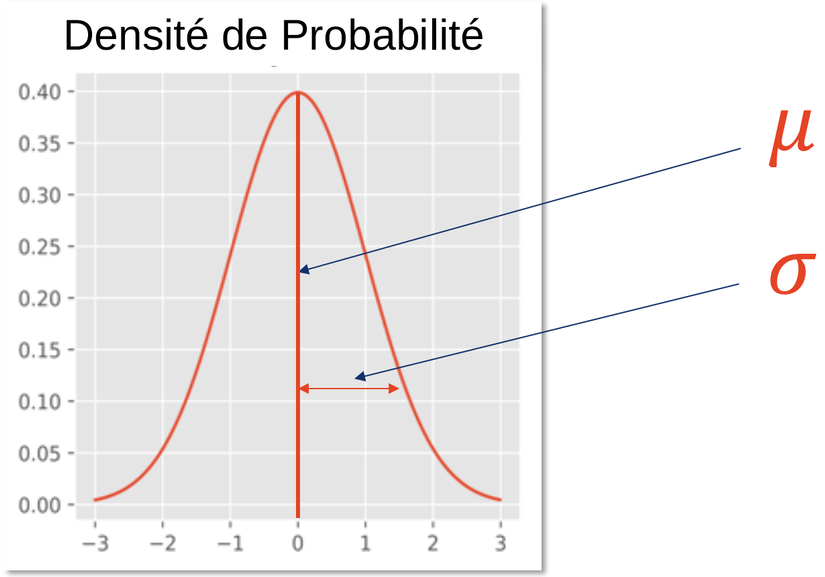
Modèle paramétrique

Comment prédire la densité de probabilité du futur d'une série temporelle ?

Postuler sur une loi de distribution

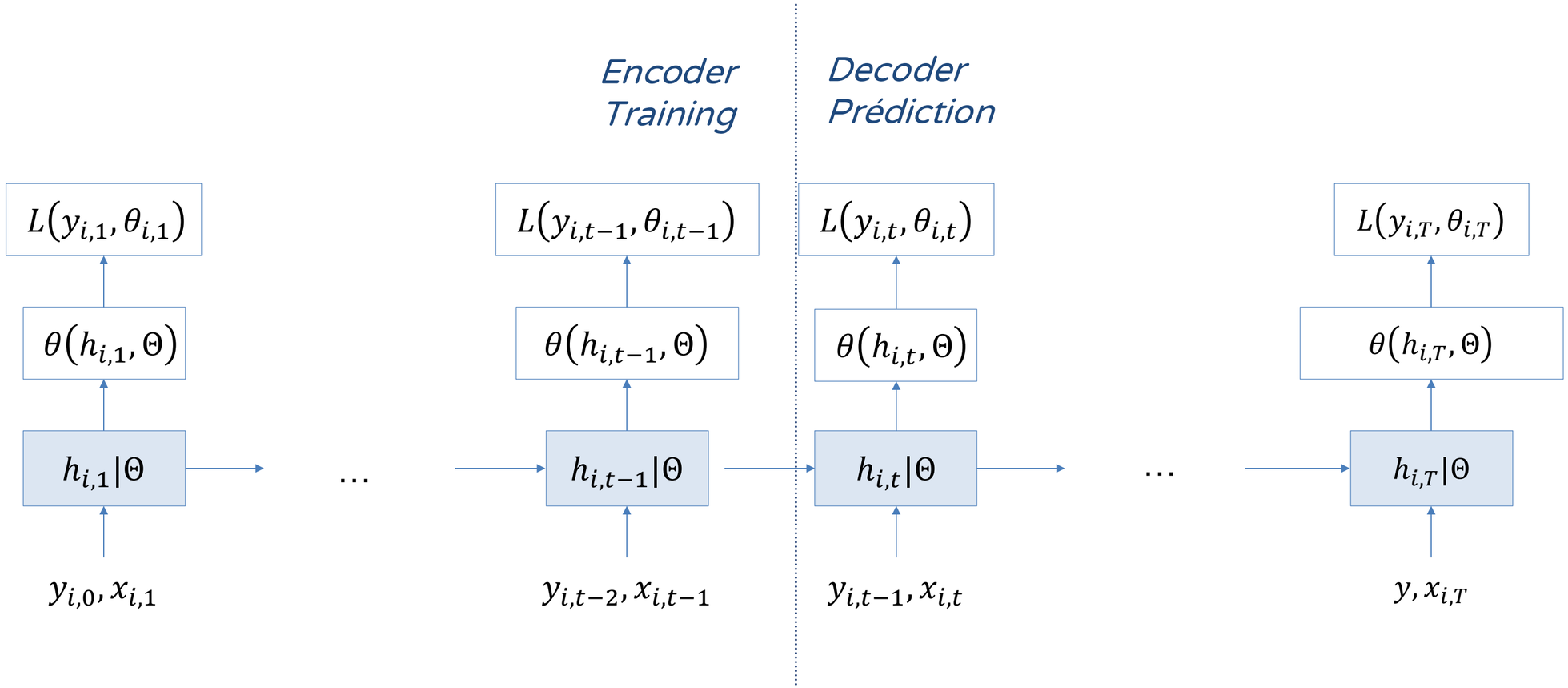
Loi gaussienne, loi binomiale négative, loi de t-student, etc.

Estimer les paramètres de cette loi



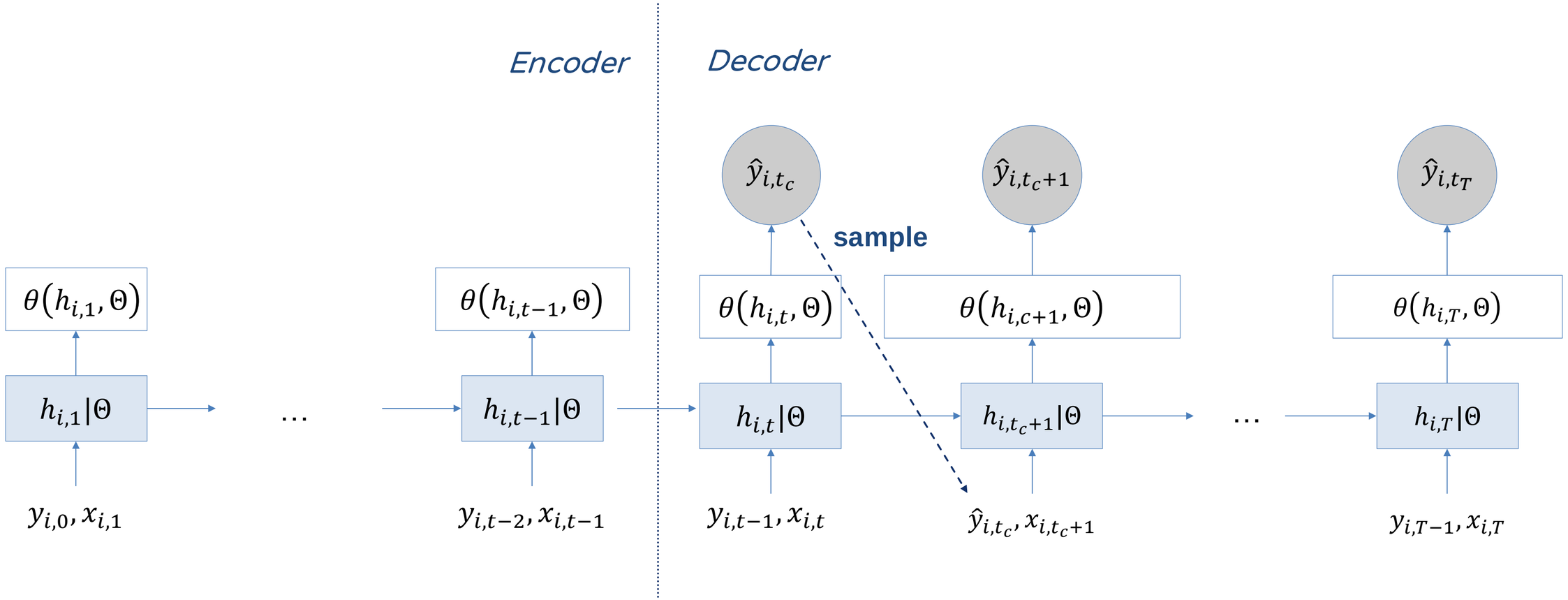
6.2 DeepAR

Entrainement



6.2 DeepAR

Prédiction



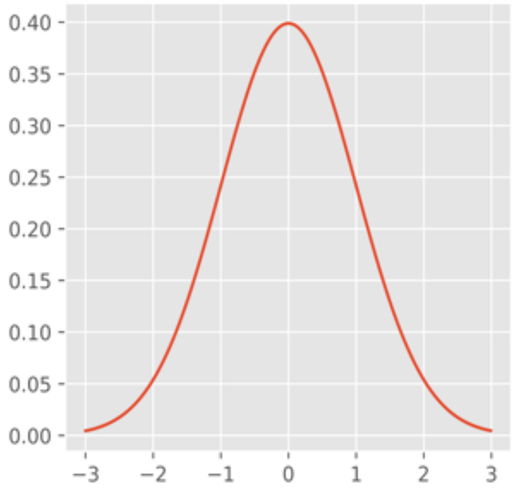
6.3 SQF-RNN : approche non paramétrique

Gasthaus, J., Benidis, K., Wang, Y., Rangapuram, S. S., Salinas, D., Flunkert, V., & Januschowski, T. (2019). Probabilistic Forecasting with Spline Quantile Function RNNs *Proceedings of Machine Learning Research*, *89*, 1901–1910.

Approche non paramétrique

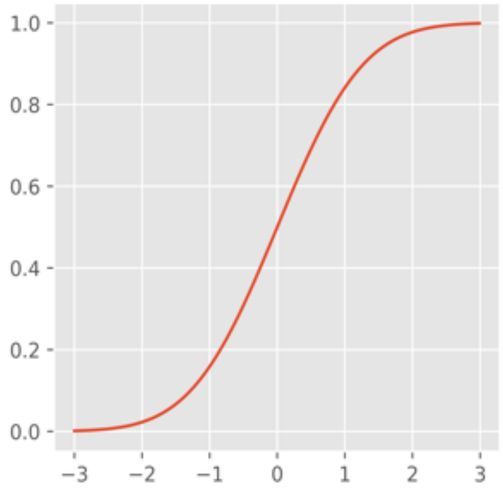
Comment représenter une distribution de probabilité ?

Densité de Probabilité



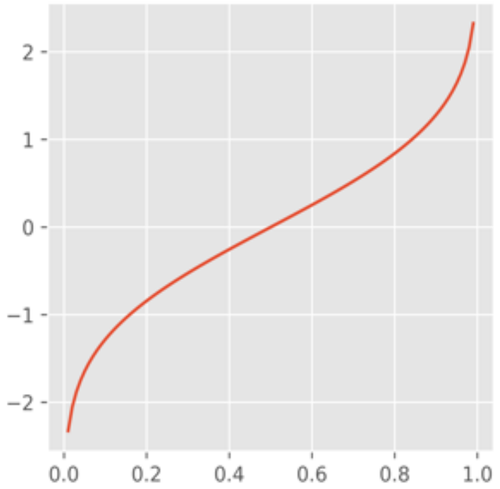
Intégrale égale à 1

Fonction de Répartition



Croissante et bornée dans [0, 1]

Fonction Quantile



Croissante

Modélisation de la fonction quantile par une régression par spline linéaire isotone

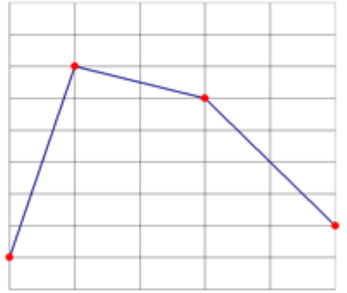
6.3 SQF-RNN : approche non paramétrique

Régression isotone

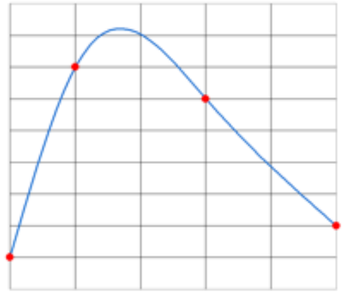
La régression isotone est une technique de modélisation d'un nuage de points par une forme libre qui doit être monotone et « aussi proche que possible des données » (souvent d'un point de vue des moindres carrés)

Spline

- Toute fonction définie par morceaux par des polynômes
- Le degré de la spline est définie par le polynôme de plus haut degré



Spline linéaire (ordre 1) pour 4 nœuds



Spline d'ordre 2 pour 4 nœuds

6.3 SQF-RNN : approche non paramétrique

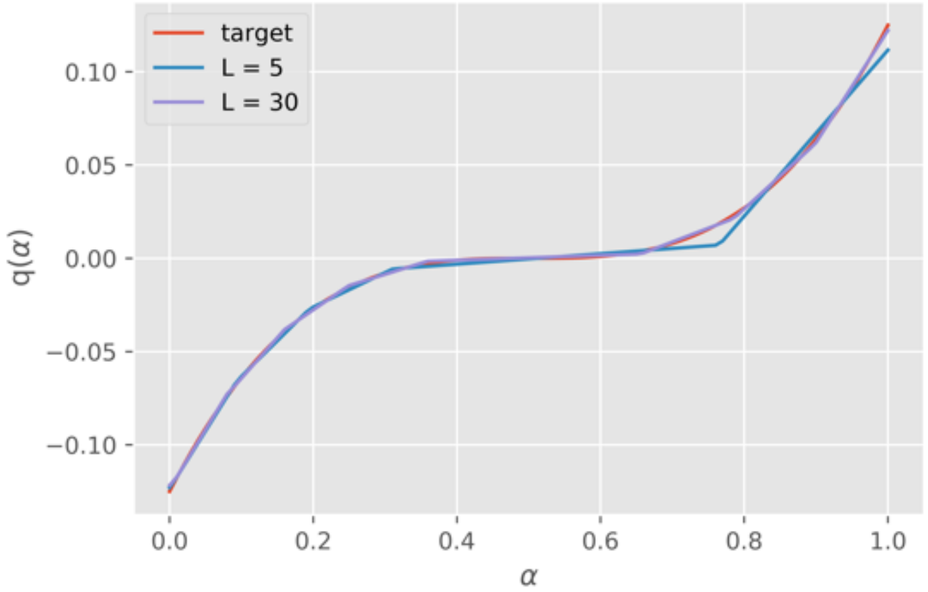
La régression par spline linéaire isotone

- L'expression de la régression par spline linéaire isotone est donnée par :

$$q(\alpha; \gamma, b, d) = \gamma + \sum_{l=0}^{L} b_l \, (\alpha - d_l)_+$$

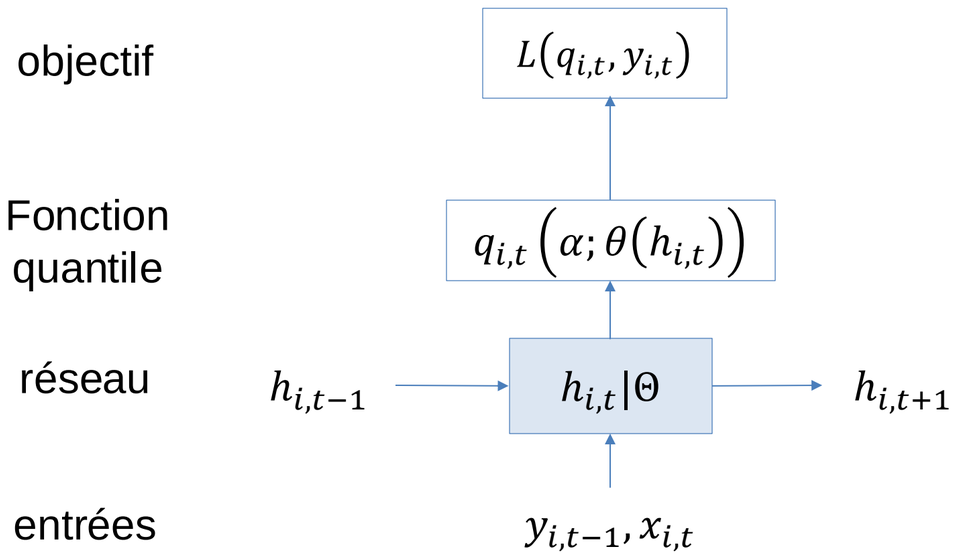
Où

- q(α) est la valeur de la fonction quantile pour le quantile α
- γ représente l'ordonnée à l'origine
- L représente le nombre de nœuds
- $b \in R^{L+1}$ représente les pentes des fonctions par morceaux
- $d \in R^{L+1}$ représente les positions des nœuds de la spline
- $(x)_+ = \max(0, x)$ représente la fonction ReLU



6.3 SQF-RNN : approche non paramétrique

Architecture



Objectif : $L(q_{i,t}, y_{i,t}) = CRPS(q_{i,t}, y_{i,t})$, avec $y_{i,t}$ la valeur cible

Fonction quantile : $q_{i,t}$ spline, $\alpha \in [0,1]$ quantile, θ paramètres de la spline comme fonction de l'état $h_{i,t}$ du RNN série temporelle au pas de temps précédent et $x_{i,t}$ features

Réseau : $h_{i,t-1}$ sortie du RNN au pas de temps précédent, $h_{i,t}$ sortie du RNN au pas de temps en cours, Θ ensemble des paramètres du RNN partagé $\forall \boldsymbol{i}, \boldsymbol{t}$

Entrées : y série temporelle au pas de temps précédent et $x_{i,t}$ features

6.3 SQF-RNN / DeepAR

DeepAR

Objectif: Estimer les paramètres d'une distribution fixée

- (-) Hypothèse forte sur les données
- (-) Difficultés pour capturer des distributions complexes
- (+) Faible nombre de paramètres à estimer

SQF-RNN

Objectif: Estimer une forme libre

- (+) Pas d'hypothèse sur les données
- (+) Capacité à capturer des distributions complexes
- (-) Nombre de paramètres à estimer important

6.4 Pinball Loss 1/3

Fonction objectif pour un seul quantile

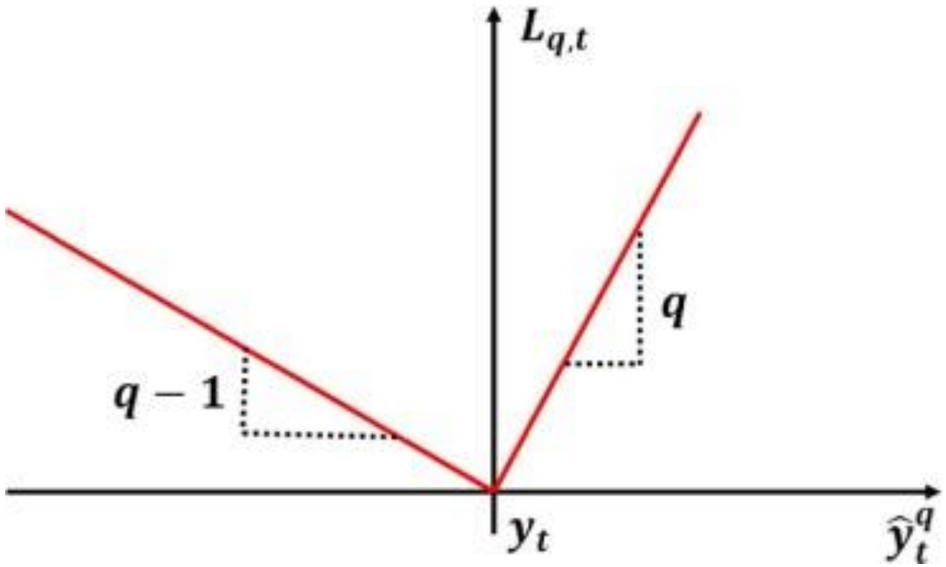
Pour estimer une fonction quantile donnée (α fixé), on utilise la pinball loss :

$$q^* = F^{-1}(\alpha) = \underset{q}{\operatorname{argmin}}\{E_{F(\mathrm{y})}[\Delta_\alpha(q, \mathrm{y})]\} \approx \underset{q}{\operatorname{argmin}}\left\{\frac{1}{N}\sum_i \Delta_\alpha(q, \mathrm{y}_i)\right\}$$

Où

$$\Delta_\alpha(\mathrm{q}, \mathrm{y}) = \begin{cases}(\alpha - 1)(\mathrm{y} - q), & \mathrm{y} < q \\ \alpha(\mathrm{y} - q), & \mathrm{y} \geq q\end{cases}$$

avec α le quantile cible, y la valeur réelle, et q la prévision quantile.



6.4 Pinball Loss 2/3

Fonction objectif pour un seul quantile

- La pinball loss est toujours positive
- Plus basse est la loss, meilleur sera le forecast au quantile considérée

- Dans le cas particulier de la médiane, cette loi correspond au calcul de la MAE :

$$m^* = F^{-1}(0.5) = \underset{m}{\operatorname{argmin}}\{E_{F(\mathrm{y})}[|\mathrm{y} - m|]\} \approx \underset{m}{\operatorname{argmin}}\left\{\frac{1}{N}\sum_i |\mathrm{y}_i - m|\right\}$$

6.4 Pinball Loss 3/3

Fonction objectif pour plusieurs quantiles

- Choisir un set de quantiles fini : $\alpha = \{0.1, 0.5, 0.9\}$
- Minimiser une combinaison des pinball loss associées

$$q_\alpha = \underset{q_\alpha}{\operatorname{argmin}} \left\{ \frac{1}{N} \sum_i \Delta_\alpha(q_\alpha, z_i) \right\}$$

Limitations

- Restriction du set de quantiles

6.6 Méthodes ML probabilistes

Régression quantile

L'utilisation de la quantile loss comme fonction de loss permet d'adapter les algorithmes de machine learning pour la prévision probabiliste, comme par exemple :

- Régression linéaire (OLS)
- Boosting (XGBoost, LightGBM, …) intègrent un hyperparamètre permettant d'estimer les prévisions pour le quantile considéré. Pour chaque quantile, un entrainement du modèle est nécessaire
- Pour les forêts aléatoires, on va plutôt utiliser la variance de l'estimateur global (rappel : la prévision ponctuelle est la moyenne des prévisions pour chaque arbre de la forêt) pour obtenir les intervals de prédiction

6.7 Time to practice

Rendez-vous sans plus attendre sur Python pour la mise en pratique !

6.7 Time to practice

Qu’est-ce que GluonTS ?

GluonTS est une boîte à outils construites sur Apache MXNet qui permet de créer des modèles probabilistes pour les séries temporelles. GluonTS permet en particulier de créer des modèles globaux qui apprennent d’une multitude de séries temporelles tels que DeepAR.

La construction des modèles se fait en suivant la structure suivante :

- Mise en forme des données
- Caractérisation du modèle deep learning
- Définition de la fonction d’approximation de la fonction quantile
- Entraînement et prédictions
- Evaluation des performances

6.7 Time to practice

Préparation des données

```
# train dataset: cut the last window of length "prediction_length", add "target" and "start" fields
custom_ds_train = [custom_ds[i][:-prediction_length] for i in range(len(custom_ds))]
train_ds = ListDataset([{'target': x,
                         'start': start,
                         # optional
                         'feat_static_cat': feat_static_cat,
                         'feat_static_real': feat_static_real,
                         'feat_dynamic_cat': feat_dynamic_cat,
                        }
                        for (x, start,
                             feat_static_cat, feat_static_real, feat_dynamic_cat)
                        in zip(custom_ds_train, start_dates,
                               feat_static_cat, feat_static_real, feat_dynamic_cat)],
                       freq=freq
                      )

# test dataset: use the whole dataset, add "target" and "start" fields
test_ds_2 = ListDataset([{'target': x,
                          'start': start,
                          # optional
                          'feat_static_cat': feat_static_cat,
                          'feat_static_real': feat_static_real,
                          'feat_dynamic_cat': feat_dynamic_cat,
                         }
                         for (x, start,
                              feat_static_cat, feat_static_real, feat_dynamic_cat)
                         in zip(custom_ds_train, start_dates,
                                feat_static_cat, feat_static_real, feat_dynamic_cat)],
                        freq=freq
                       )
```

6.7 Time to practice

Entrainement : Estimator / Predictor

```
estimator = deepar.DeepAREstimator(
    # required
    freq=freq,
    prediction_length=prediction_length,
    # optional
    context_length=84,
    distr_output=GaussianOutput(), # PiecewiseLinearOutput(num_pieces=10)
    num_cells=80,
    num_layers=3,
    trainer=Trainer(batch_size=32,
                    epochs=50,
                    learning_rate=1e-3,
                    num_batches_per_epoch=50
                   ))
```

```
predictor = estimator.train(train_ds)
```

6.7 Time to practice

Optimisation du modèle : Distribution / Sortie

```
forecast_it, ts_it = make_evaluation_predictions(
    dataset=test_ds,  # test dataset
    predictor=predictor,  # predictor
    num_eval_samples=100,  # number of sample paths we want for evaluation
)
```

6.7 Time to practice

Prévision et évaluation

```
evaluator = Evaluator(quantiles=[0.1, 0.5, 0.9])
agg_metrics, item_metrics = evaluator(iter(tss), iter(forecasts), num_series=len(test_ds))
```


```
"MSE": 3536106.055594704,
"abs_error": 64371461.11035156,
"abs_target_sum": 332752469.0,
"abs_target_mean": 7105.540657698058,
"seasonal_error": 1379.319254023263,
"MASE": 1.0161705604453757,
"sMAPE": 0.1979206739536688,
"MSIS": 9.649582605632155,
"QuantileLoss[0.1]": 25851500.576928712,
"Coverage[0.1]": 0.08370702541106127,
"QuantileLoss[0.5]": 64371461.107299805,
"Coverage[0.5]": 0.5480461242793081,
"QuantileLoss[0.9]": 36850704.2774414,
"Coverage[0.9]": 0.8814221652786675,
"RMSE": 1880.453683448413,
"NRMSE": 0.2646461084437188,
"ND": 0.1934514905443168,
"wQuantileLoss[0.1]": 0.07768988357808011,
"wQuantileLoss[0.5]": 0.19345149053514551,
"wQuantileLoss[0.9]": 0.11074509646220355,
"mean_wQuantileLoss": 0.12729549019180972,
"MAE_Coverage": 0.027638977863193134
```

6.7 Time to practice

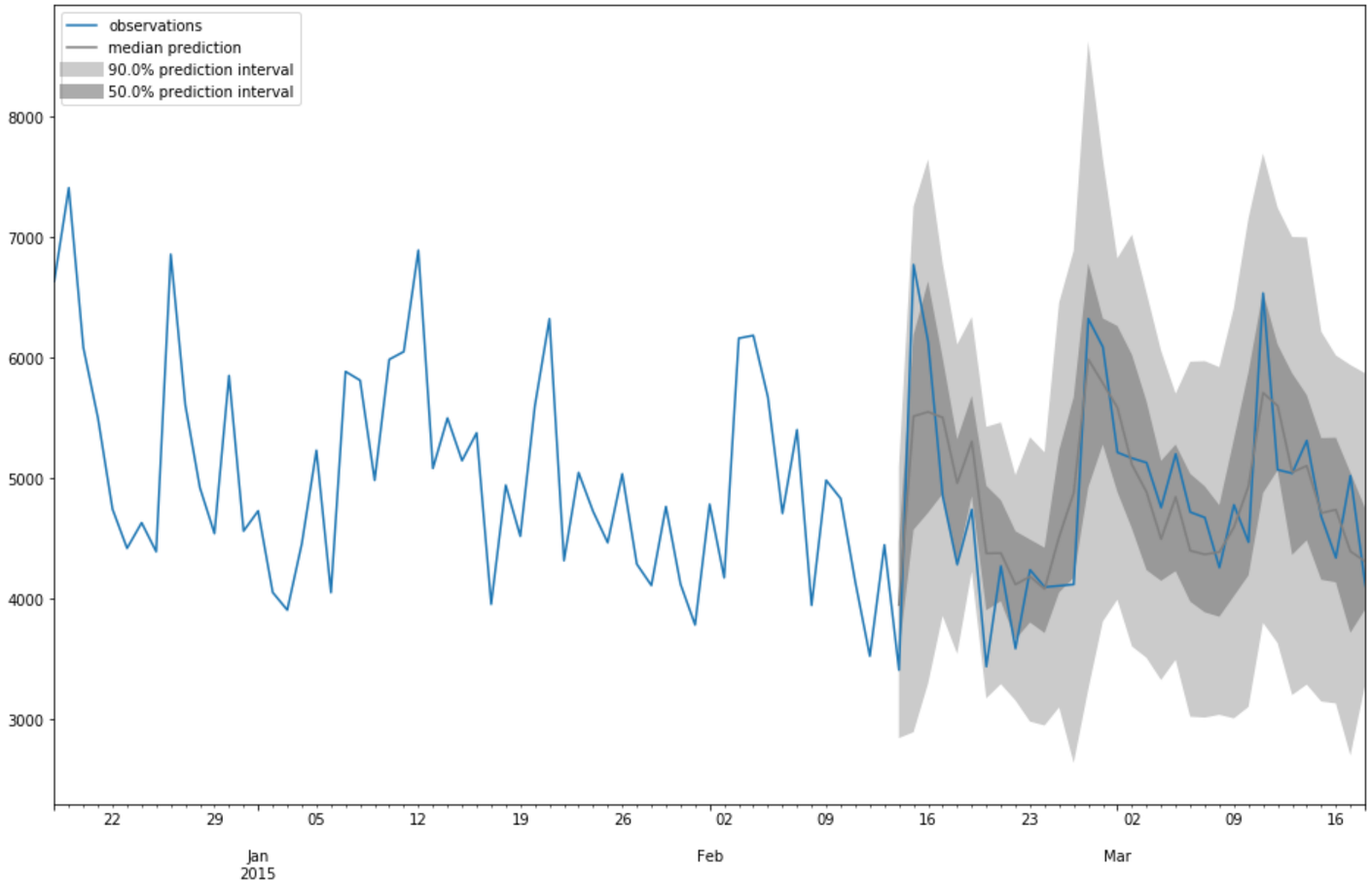
Prévision et évaluation

```
evaluator = Evaluator(quantiles=[0.1, 0.5, 0.9])
agg_metrics, item_metrics = evaluator(iter(tss), iter(forecasts), num_series=len(test_ds))
```


```
"MSE": 3536106.055594704,
"abs_error": 64371461.11035156,
"abs_target_sum": 332752469.0,
"abs_target_mean": 7105.540657698058,
"seasonal_error": 1379.319254023263,
"MASE": 1.0161705604453757,
"sMAPE": 0.1979206739536688,
"MSIS": 9.649582605632155,
"QuantileLoss[0.1]": 25851500.576928712,
"Coverage[0.1]": 0.08370702541106127,
"QuantileLoss[0.5]": 64371461.107299805,
"Coverage[0.5]": 0.5480461242793081,
"QuantileLoss[0.9]": 36850704.2774414,
"Coverage[0.9]": 0.8814221652786675,
"RMSE": 1880.453683448413,
"NRMSE": 0.2646461084437188,
"ND": 0.1934514905443168,
"wQuantileLoss[0.1]": 0.07768988357808011,
"wQuantileLoss[0.5]": 0.19345149053514551,
"wQuantileLoss[0.9]": 0.11074509646220355,
"mean_wQuantileLoss": 0.12729549019180972,
"MAE_Coverage": 0.027638977863193134
```

6.7 Time to practice

Visualisation des forecasts et des intervalles de confiance



Séries temporelles
Méthodes ML et DL
Conclusion / Wrap-up

Intervenant
Guillaume Hochard

7.1 Conclusion

Vous êtes maintenant familier des différentes étapes d'un projet de prévision, connaissez les pièges à éviter et avez acquis des compétences dans les domaines suivants :

- **Data viz**
- **Analyse exploratoire de données**
- **Data cleaning et feature engineering pour les modèles ML**
- **Méthodes et modèles Machine Learning**
- **Modèles Deep Learning**
- **Prévision probabiliste (DL et ML)**

Bonne continuation, ce n'est que le début !

7.1 Conclusion

Méthode	Avantages	Inconvénients	Points d'attention
Statistique/ Econométrique	• Spécifiques aux séries temporelles • Saisonnalité prise en compte • Intervalle de confiance • Prédiction pour toutes les séries temporelles du modèle	• Inefficacité long-terme • Grand nombre de paramètres	• Input signal stationnaire • Résidus ne doivent pas présenter de motif temporel récurrent
Machine Learning	• Des modèles très performants • Interprétabilité (LIME/SHAP)	• Dépendant du choix des variables • Modélisation très longue • Feature engineering complexe • Choix d'un lag difficile	• Choix du modèle employé (XGBoost, LightGBM, CatBoost?) • Normalisation et choix des paramètres
Deep Learning	• Pas/peu de feature engineering • Modélisation de relations non linéaires • RNN / LSTM : mémorisation • Performances excellentes selon historique de données	• Peu d'interprétabilité • Paramétrage complexe (nb de couches, régularisation)	• Ajout de certaines features pour enrichissement • CV pour profondeur historique

7.2 Pour aller plus loin

Voici quelques ouvrages de référence qui vous permettront d'approfondir vos connaissances sur le sujet.

- Introduction to Time Series and Forecasting, Peter J. Brockwell, Richard A. Davis
- Forecasting: Principles and Practice (3rd ed), Rob J Hyndman and George Athanasopoulos

- Practical Time Series Analysis : Prediction with Statistics & Machine Learning, Aileen Nielsen
- Machine Learning for Time Series Forecasting with Python, Francesca Lazzeri

- Data Science for Supply Chain Forecasting, Nicolas Vandeput