Probabilistic Forecasting with DeepAR and AWS SageMaker
DeepAR – Yet Another Forecasting Algorithm?
Advantages of DeepAR

Probabilistic Forecasts

Like...
• ARIMA
• Regression Models

But not...
• Plain LSTMs (Neural Network)
Advantages of DeepAR

Automatic Feature Engineering

Like...
- Plain LSTMs

But not...
- ARIMA
- Regression Models
- Etc.
Advantages of DeepAR

One algorithm for multiple timeseries

Like...
- Meta Learning?
- Transfer Learning?
Disadvantages

- Time- and resource intensive to train
- Difficult to set hyperparameters and to tune
DeepAR – How does it work?

\[ \tilde{z} \sim \ell(\cdot | \theta) \]

\[ \ell(z_{i,t-1} | \theta_{i,t-1}) \quad \ell(z_{i,t} | \theta_{i,t}) \quad \ell(z_{i,t+1} | \theta_{i,t+1}) \]

inputs: \[ \tilde{z}_{i,t-2}, x_{i,t-1} \quad \tilde{z}_{i,t-1}, x_{i,t} \quad \tilde{z}_{i,t}, x_{i,t+1} \]

network: \[ h_{i,t-1} \quad h_{i,t} \quad h_{i,t+1} \]

samples: \[ \tilde{z}_{i,t-1} \quad \tilde{z}_{i,t} \quad \tilde{z}_{i,t+1} \]
Example Datasets for DeepAR

• Sales at Amazon
• Sales in Stores
• Forecast Load of Servers in Datacenter
• Car traffic
• Energy Consumption in Households
AWS SageMaker

Label

Amazon SageMaker Ground Truth
Build and manage training data sets

Build

Amazon SageMaker Studio
Integrated development environment (IDE) for machine learning

Amazon SageMaker Autopilot
Automatically build and train models

Amazon SageMaker Notebooks
One-click notebooks with elastic compute

Amazon SageMaker Experiments
Capture, organize, and search every step

AWS Marketplace
Pre-built algorithms and models

Amazon SageMaker Debugger
Debug and profile training runs

Automatic Model Tuning
One-click hyperparameter optimization

Train & Tune

Deploy & Manage

Amazon SageMaker Model Monitor
Automatically detect concept drift

Amazon SageMaker Model Monitor
Train once, deploy anywhere

Amazon Augmented AI
Add human review of model predictions

https://aws.amazon.com/sagemaker/?nc1=h_ls

EuroPython 2020 - Probabilistic Forecasting with DeepAR and AWS SageMaker
Let's code - Imports

```python
import boto3
import s3fs
import sagemaker
from sagemaker import get_execution_role
from sagemaker.amazon.amazon_estimator import get_image_uri
```
Data preparation

{"start": "2009-11-01 00:00:00", "target": [4.3, "NaN", 5.1, ...], "cat": [0, 1], "dynamic_feat": [[1.1, 1.2, 0.5, ...]]}

{"start": "2012-01-30 00:00:00", "target": [1.0, -5.0, ...], "cat": [2, 3], "dynamic_feat": [[1.1, 2.05, ...]]}

{"start": "1999-01-30 00:00:00", "target": [2.0, 1.0], "cat": [1, 4], "dynamic_feat": [[1.3, 0.4]]}
Hyperparameter

hyperparameters = {
    "time_freq": "H",
    "context_length": "72",
    "prediction_length": "24",
    "num_cells": "50",
    "num_layers": "3",
    "likelihood": "gaussian",
    "epochs": "25",
    "mini_batch_size": "64",
    "learning_rate": "0.001",
    "dropout_rate": "0.05",
    "early_stopping_patience": "30"
}
Train Model - I

sagemaker_session = sagemaker.Session()

role = sagemaker.get_execution_role()

image_name = sagemaker.amazon.

    amazon_estimator.get_image_uri(region,
    "forecasting-deepar",
    "latest")
Train Model - II

estimator = sagemaker.estimator.Estimator(
    sagemaker_session=sagemaker_session,
    image_name=image_name,
    role=role,
    train_instance_count=1,
    train_instance_type="ml.c4.xlarge",
    base_job_name="electricity-deepar",
    output_path="s3://" + s3_output_path)

estimator.set_hyperparameters(**hyperparameters)
Fit Model

data_channels = {
    "train": f"s3://{s3_data_path}/train/",
    "test": f"s3://{s3_data_path}/test/"
}

estimator.fit(inputs=data_channels)
Deployment

```
job_name = estimator.latest_training_job.name

endpoint_name = sagemaker_session.endpoint_from_job(
    job_name = job_name,
    initial_instance_count = 1,
    instance_type = "ml.m4.xlarge",
    deployment_image = image_name,
    role = role)
```
Questions?

Feel free to contact me!

NK@data-convolution.de
+49 176 43 69 6691
www.data-convolution.de