Yellow taxi

DeepAR

Dataset Description

- Source data: daily ride of yellow taxi in New York (source);
- Data used: before July-2016;
- The source contains data on rides with Lan/Lat markers;
- New York is divided into square regions: link
  - Only squares with 5+ rides in May-2016 are selected for prediction;
- **Goal is to predict number of rides from each square per hour**
  - **Validation period: June 2016**
  - **Train/test period: April 2016/May 2016**
    (short training range is taken for sake of speed training)
- Nature of data: timeseries -> DeepAR is selected as main model.

Raw Data Example

```
VendorID,tpep_pickup_datetime,tpep_dropoff_datetime,passenger_count,trip_distance
,pickup_longitude,pickup_latitude,RatecodeID,store_and_fwd_flag,dropoff_longitude
,dropoff_latitude,payment_type,fare_amount,extra,mta_tax,tip_amount,tolls_amount,
improvement_surcharge,total_amount
1,2016-04-01 00:00:00,2016-04-01 00:01:59,1,.50,-73.976882934570313,40.7584953308
10547,1,N,-73.977668762207031,40.753902435302734,2,3.5,0.5,0.5,0,0,0.3,4.8
1,2016-04-01 00:00:00,2016-04-01 00:12:07,2,2.20,-73.985206604003906,40.757293701
171875,1,N,-73.989288330078125,40.732658386230469,1,10,0.5,0.5,2.25,0,0.3,13.55
2,2016-04-01 00:00:00,2016-04-01 00:10:41,2,.96,-73.979202270507812,40.7588691711
42578,1,N,-73.990676879882813,40.751319885253906,2,8.5,0.5,0.5,0,0,0.3,9.8
2,2016-04-01 00:00:00,2016-04-01 00:10:30,5,1.54,-73.984855651855469,40.767723083
496094,1,N,-73.990829467773437,40.751186370849609,1,8.5,0.5,0.5,1.96,0,0.3,11.76
2,2016-04-01 00:00:00,2016-04-01 00:00:00,2,10.45,-73.863739013671875,40.76947021
484375,1,N,-73.976814270019531,40.775283813476563,1,34,0,0.5,8.07,5.54,0.3,48.41
1,2016-04-01 00:00:01,2016-04-01 00:15:04,1,3.50,-73.973734130859375,40.757076263
427734,1,N,-73.9334716796875,40.766304016113281,1,14,0.5,0.5,3,0,0.3,18.3
```

Data Transformation

Monthly data is manually downloaded

DataProcessing_preps.ipynb:
All files are checked for same columns naming - differences are corrected as stored as dictionary

DataProcessing_preps.ipynb:
Based on May data squares with 5+ rides per hour in average are defined: in total 102 squares

data_processing.py:
Each year is processed separately:
- columns re-named uniformly
- data converted to hour rides
- locations are converted to square_id

TimeSeriesContcat.ipynb:
Year data is concatenated and pivoted.
In columns squares of interest.
For 3 hour in 2014,2013,2012, 2011 no data is found: filled with average

FE_clustering.ipynb:
Timeseries are standardized and clustered with kmeans.
based on 10 simulations optimal number of clusters was define: 7

utils.py:
Supporting module with functions on data transformations.
Used in:
data_processing.py
DataProcessing_preps.ipynb

```
pivoted_dt.head()
```

	Time	1075	1076	1077	1125	1126	1127	1128	1129	1130	...	1630	1684	1733	1734	1783	2068	2069	2118	2119	2168
0	2011-01-01 00:00:00	33.0	68.0	23.0	39.0	156.0	261.0	287.0	354.0	371.0	...	12.0	0.0	4.0	20.0	20.0	11.0	1.0	47.0	1.0	19.0
1	2011-01-01 01:00:00	42.0	68.0	31.0	59.0	182.0	256.0	245.0	264.0	252.0	...	10.0	0.0	4.0	22.0	13.0	10.0	5.0	34.0	4.0	18.0
2	2011-01-01 02:00:00	40.0	59.0	18.0	62.0	170.0	225.0	228.0	255.0	235.0	...	14.0	0.0	4.0	1.0	1.0	0.0	2.0	11.0	2.0	0.0
3	2011-01-01 03:00:00	35.0	52.0	18.0	47.0	129.0	216.0	208.0	213.0	183.0	...	7.0	0.0	4.0	2.0	1.0	0.0	0.0	12.0	0.0	0.0
4	2011-01-01 04:00:00	17.0	29.0	9.0	31.0	83.0	149.0	185.0	173.0	142.0	...	13.0	0.0	3.0	1.0	2.0	0.0	0.0	4.0	1.0	0.0

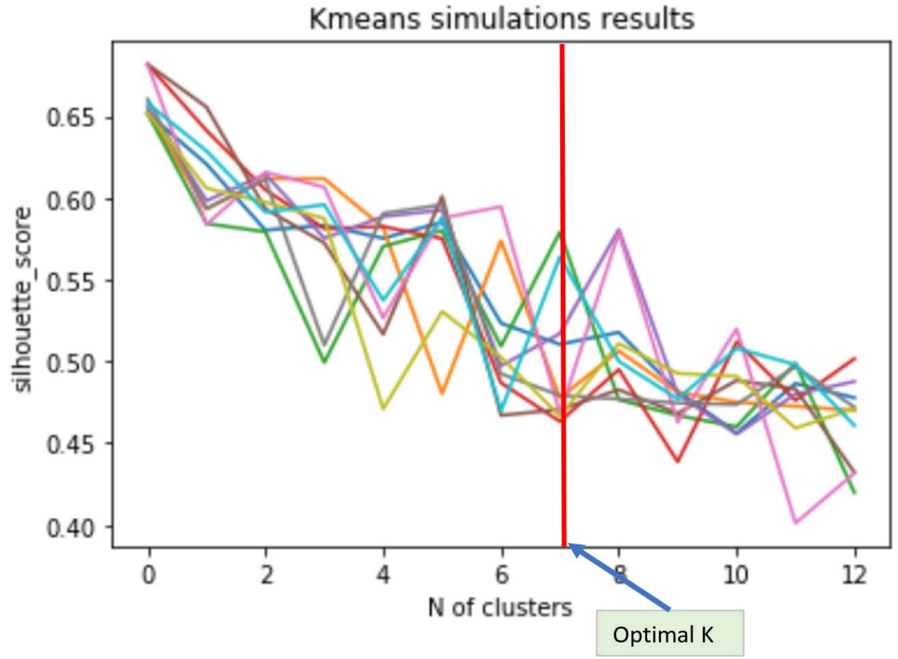
5 rows × 103 columns

Notes on FE

Quite advanced algorithm was selected, it creates multiple useful features by itself, so FE was selected primarily for educational purposes

The DeepAR algorithm automatically generates these feature time series. The following table lists the derived features for the supported basic time frequencies.

Frequency of the Time Series	Derived Features
Minute	minute-of-hour, hour-of-day, day-of-week, day-of-month, day-of-year
Hour	hour-of-day, day-of-week, day-of-month, day-of-year
Day	day-of-week, day-of-month, day-of-year
Week	day-of-month, week-of-year
Month	month-of-year



Training Notes

- All steps are based on github: [SageMaker/DeepAR demo on electricity dataset]
- All executed in local notebooks
- What is different: no lambda function was created in that notebook, endpoint is created from the notebook with

  estimator = sagemaker.estimator.Estimator()
  estimator.fit()
  estimator.deploy()

  in addition class DeepARPredictor(sagemaker.predictor.Predictor) was created with supplemented functions
- 2 metrics are reported: RMSE, MAPE

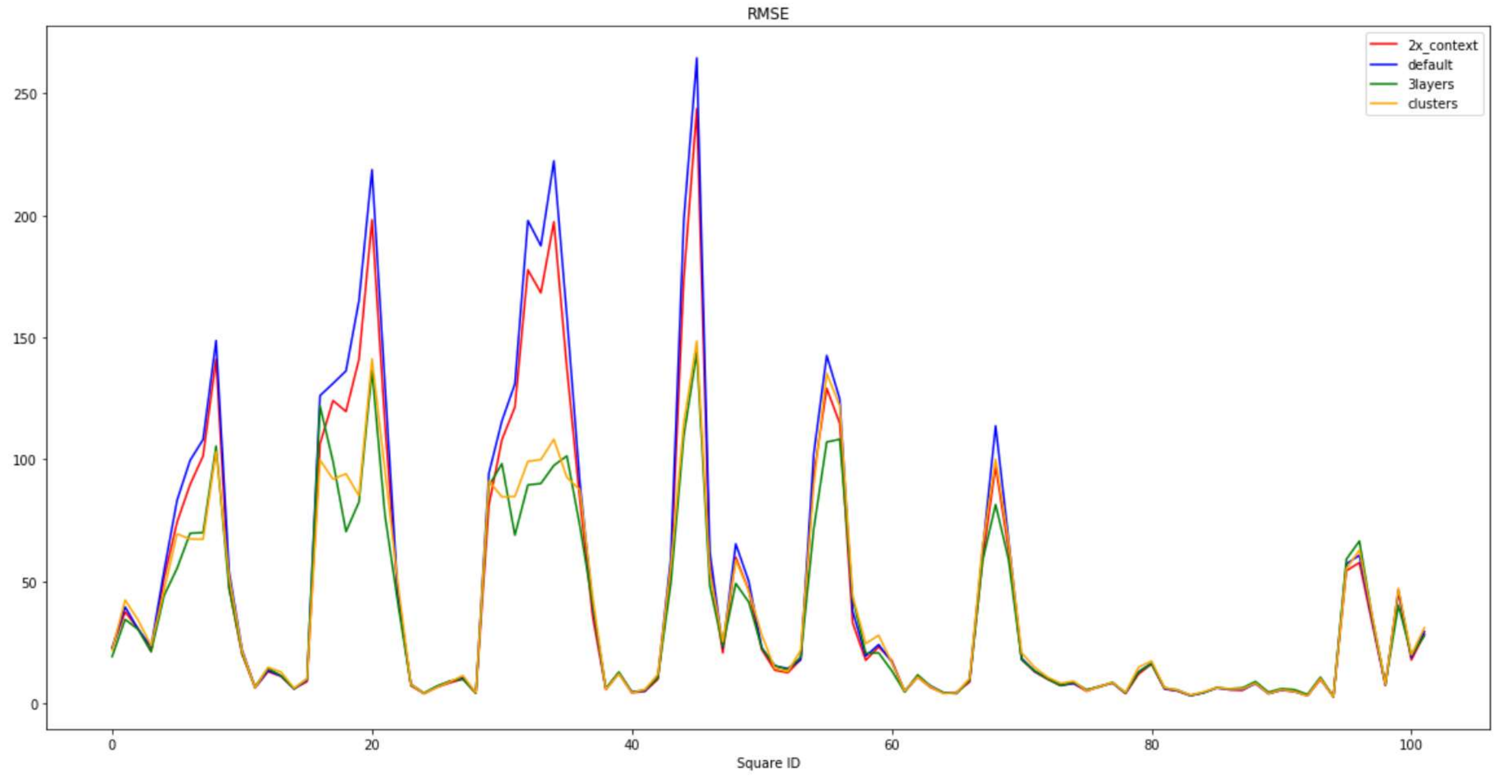
```
Signature:
 sagemaker.estimator.Estimator.deploy(
    self,
    initial_instance_count,
    instance_type,
    serializer=None,
    deserializer=None,
    accelerator_type=None,
    endpoint_name=None,
    use_compiled_model=False,
    wait=True,
    model_name=None,
    kms_key=None,
    data_capture_config=None,
    tags=None,
    **kwargs,
)
Docstring:
Deploy the trained model to an Amazon SageMaker endpoint and return a
``sagemaker.Predictor`` object.
```

Training scenarios

- Default scenario:
  - Frequency: hourly
  - Context length: 24*7 (1 week)
  - Prediction length: 4 weeks forward
- Scenario 1: [default scenario] + context=2*[default scenario context]
- Scenario 2: [default scenario] + num_layers (number of layers in the network) increased from 2(default) to 3
- Scenario 3: [default scenario] + "cat" is defined as result of clustering (FE)

Source notebook: [link]

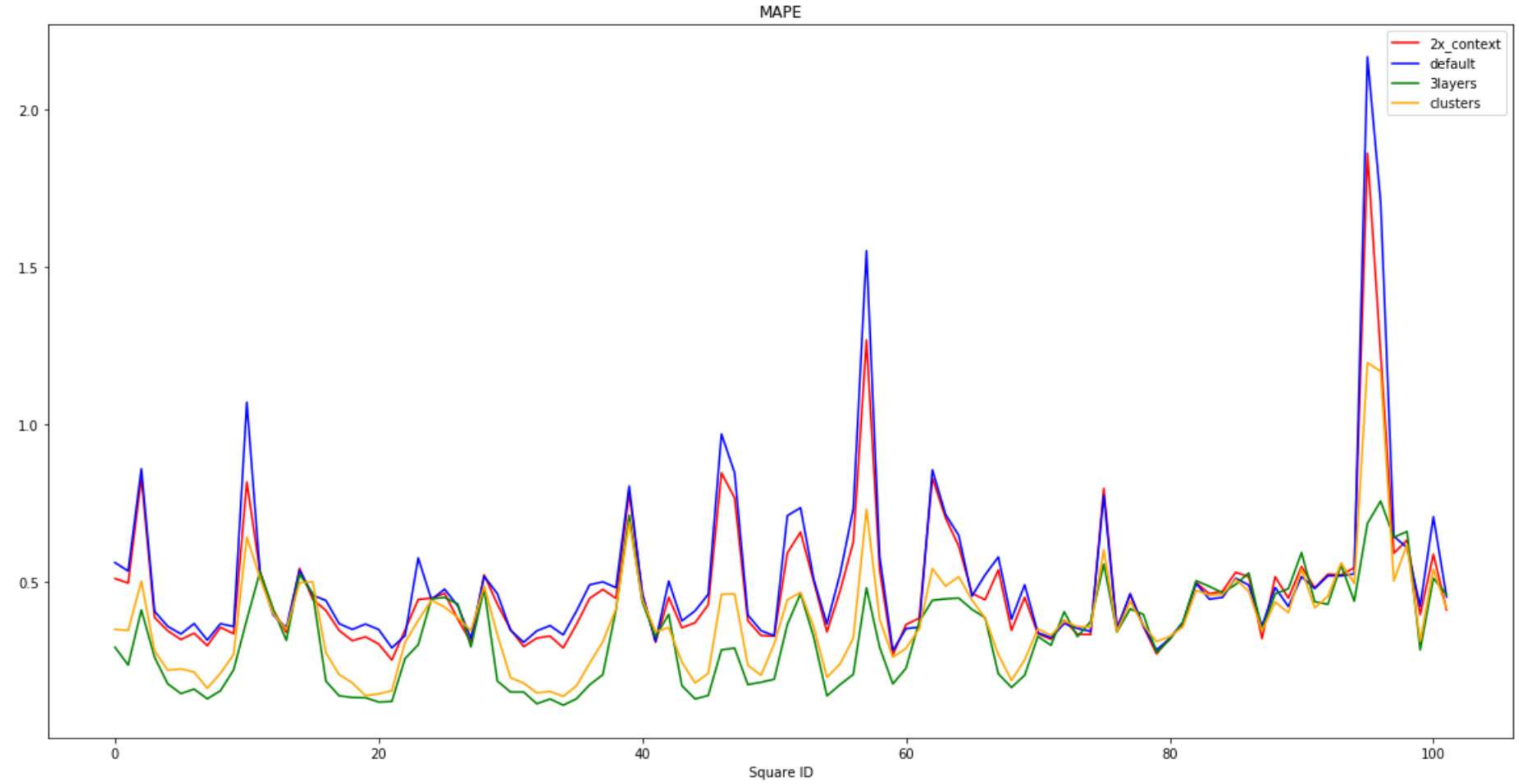
RMSE Charts



Based on simple simulations best models are:

- Model with 3 layers
- Model that used clusters as additional feature

MAPE Charts



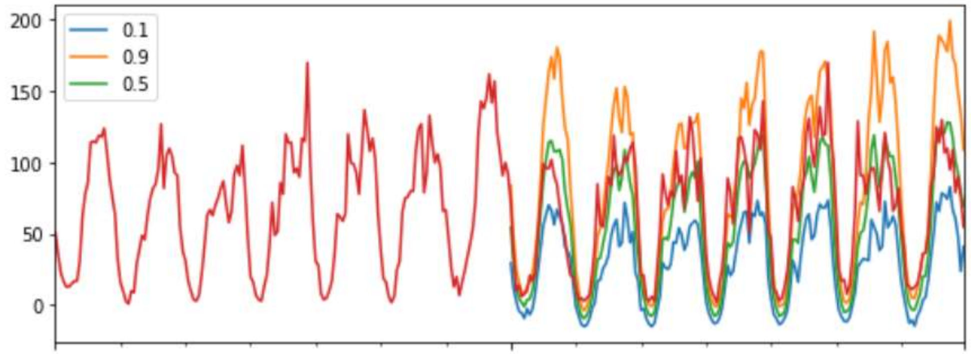
Based on simple simulations best model is:

- Model with 3 layers

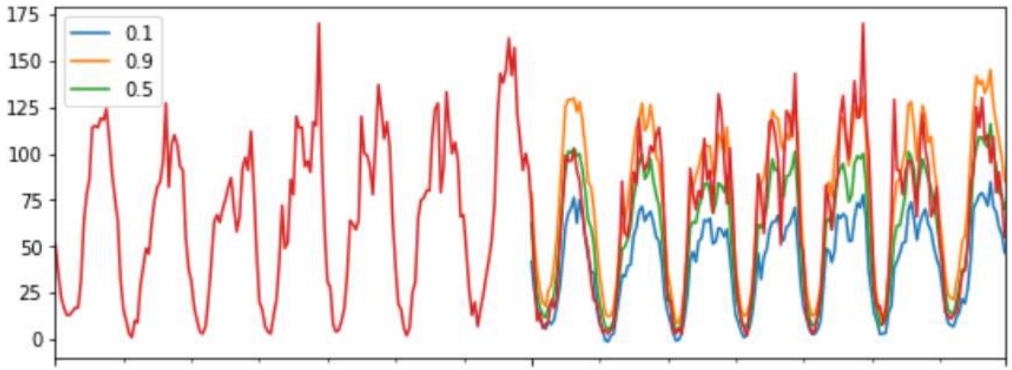
But it was not primary metric model optimized on: thus t is probably even stronger vote for 3 layers model

Predictions per one square:
different scenarios example (square_id=1075)

default



3 layers



With Clusters