Method Description

General Information

Type of Entry (*Academic, Practitioner, Researcher, Student*)	**Academic**
First Name	**Team name:**
Last Name	**Tartu M4 seminar**
Country	**Estonia**
Type of Affiliation (*University, Company-Organization, Individual*)	**University**
Affiliation	**University of Tartu, Estonia**

Team Members (*if applicable*):

1st Member	
First Name	Meelis
Last Name	Kull
Country	Estonia
Affiliation	University of Tartu
2nd Member	
First Name	Gunay
Last Name	Abdullayeva
Country	Estonia
Affiliation	University of Tartu
3rd Member	
First Name	Aytaj
Last Name	Aghabayli
Country	Estonia
Affiliation	University of Tartu
4th Member	
First Name	Saumitra
Last Name	Bagchi
Country	Estonia
Affiliation	University of Tartu
5th Member	
First Name	Diana
Last Name	Grygorian
Country	Estonia
Affiliation	University of Tartu
6th Member	
First Name	Anti
Last Name	Ingel
Country	Estonia
Affiliation	University of Tartu
7th Member	
First Name	Sriyal
Last Name	Jayasinghe
Country	Estonia
Affiliation	University of Tartu

8th Member	
First Name	Olha
Last Name	Kaminska
Country	Estonia
Affiliation	University of Tartu
9th Member	
First Name	Kaur
Last Name	Karus
Country	Estonia
Affiliation	University of Tartu
10th Member	
First Name	Viacheslav
Last Name	Komisarenko
Country	Estonia
Affiliation	University of Tartu
11th Member	
First Name	Yevheniia
Last Name	Kryvenko
Country	Estonia
Affiliation	University of Tartu
12th Member	
First Name	Markus
Last Name	Kängsepp
Country	Estonia
Affiliation	University of Tartu
13th Member	
First Name	Martin
Last Name	Liivak
Country	Estonia
Affiliation	University of Tartu
14th Member	
First Name	Markus
Last Name	Loide
Country	Estonia
Affiliation	University of Tartu
15th Member	
First Name	Mikhail
Last Name	Papkov
Country	Estonia
Affiliation	University of Tartu
16th Member	
First Name	Anton
Last Name	Potapchuk
Country	Estonia
Affiliation	University of Tartu
17th Member	
First Name	Joonas
Last Name	Puura
Country	Estonia
Affiliation	University of Tartu

18th Member	
First Name	Maksym
Last Name	Semikin
Country	Estonia
Affiliation	University of Tartu
19th Member	
First Name	Novin
Last Name	Shahroudi
Country	Estonia
Affiliation	University of Tartu
20th Member	
First Name	Prabhant
Last Name	Singh
Country	Estonia
Affiliation	University of Tartu
21st Member	
First Name	Basar
Last Name	Turgut
Country	Estonia
Affiliation	University of Tartu
22nd Member	
First Name	Andre
Last Name	Tättar
Country	Estonia
Affiliation	University of Tartu
23rd Member	
First Name	Martin
Last Name	Valgur
Country	Estonia
Affiliation	University of Tartu
24th Member	
First Name	Märten
Last Name	Veskimäe
Country	Estonia
Affiliation	University of Tartu
25th Member	
First Name	Alina
Last Name	Vorontseva
Country	Estonia
Affiliation	University of Tartu

Information about the method utilized

Name of Method	**Tartu M4 Combination**
Type of Method (*Statistical, Machine Learning, Combination, Other*)	**Combination**
Short Description (up to 200 words)	We decided to apply different prediction methods depending on the prediction interval length (hourly, daily, weekly, monthly, quarterly, yearly). We used 15 different methods (including the 2 given

	naïve methods) to obtain 15 sets of predictions. Each of the final predictions is an ensemble of 2-12 sets of predictions, where the ensemble is different for each prediction interval length. The standard methods that we used were: ARIMA, ETS, STL, random forest, xgboost, LightGBM and variants of naïve. These were complemented with simple tricks, such as ensuring that predictions are non-negative.

Extended Description:

1 Introduction

Our submission is the final result of the M4 seminar at the University of Tartu. This was a course involving 24 students and 1 instructor. Very few of the participants had any experience in time-series forecasting before the course, so we had to learn all the techniques during the course. The publication about the previous M3 competition turned out to be very useful for us, highlighting the promising directions. At the same time there were so many of us that we could afford spending time on exploring less promising directions as well.

As the first step we separated a holdout validation dataset from the training data. From each training time-series we removed the last time points and these removed values formed the validation data. The number of time points was selected such that the validation data would have exactly the same shape as the test data. After this we were able to evaluate any approach by calculating its OWA on the validation set. After having tested many approaches we picked a final set of 15 methods and entered the final phase of building ensembles of these methods. In the following we describe first the individual 15 methods, followed by the description of the ensembles.

2 Descriptions of individual methods used for ensembling

Here is the list of 15 methods (these have our internal code names):

1. Naive1_full_test.csv
2. Naive2_full_test.csv
3. Anti_arima_test.csv
4. Fearless5_all_test.csv
5. XGBpower_stl_submit.csv
6. RF_arima_test.csv
7. RF_ets_test.csv.

8. RF_damped_test.csv
9. RandomF_arima_preprocessed_test.csv
10. RandomF_RandomF1_test.csv
11. RandomF_XGBoost1_test.csv
12. RandomF_XGBoost2_test.csv
13. S_predictions_corr5__full_final.csv
14. stupidity_teamKaur_test.csv
15. TeamPotap

The descriptions are as follows:

2.1 Naive 1 (Naive1_full_test.csv)

Method “Naive 1” predicts always the last value of a timeseries.

2.2 Naive 2 (Naive2_full_test.csv)

Method “Naive 2” predicts the last value of a timeseries, taking account seasonality. For example, if we want to predict a value for January for yearly data, we return the value of last January in dataset.

2.3 Anti_arima_test.csv

The function auto.arima from R forecast package (run through python using rpy2) is run on each row separately using the parameters: max_order = 8, start_p = 0, start_q = 0, stepwise=False, approximation=False.

2.4 Fearless5_all_test.csv

The method is fairly simple and tries to fit a combination of linear, exponential, and sinusoidal function on the trend and residuals of a time series. Prediction is made rowwise (a separate model for each row). Using the seasonal decomposition of a time series one can obtain trend, seasonal and residual signals. The last part of each signal (after decompositions) - with the size of the forecast horizon - is used for the fitting process. Then the obtained coefficients of the fitting process is used to extrapolate *fh* steps. The seasonal signal obtained from the decomposition is periodic and can be used directly for the final prediction. To this end, the extrapolated values of the trend and residuals are added up together with the seasonal values to obtain the final prediction for each time series.

Hacks

- The seasonal decomposition is done with freq=2 for time series of frequency 1 to obtain a non-constant / non-zero seasonal signal
- Different variations of fitting used to obtain final prediction but in general the idea was to fit linear/exponential curve to the trend and sinusoidal to the residuals - the idea came from experiments and observations on the dataset (simple but effective in many cases)

- Performance of each variation on the hold-out dataset used as a reference to perform the best variation for the final submission. So that we know which variation worked best based on the hold-out, hopefully it performs well for the final submission as well.

2.5 ARIMA/ETS/STL (XGBpower_stl_submit.csv)

For each time series 5 models are built on training data and evaluated on validation data. Next, the best model according to sMAPE is selected, and trained again with the whole data, but the parameters do not change. 5 models are (using R forecast): ARIMA (auto.arima), ETS (ets), decomposing time series into trend, seasonality and random and predicting trend with ARIMA (stlm(method = "arima")), decomposing time series and predicting trend with ETS (stlm(method = "ets")), decomposing time series and predicting trend with Theta (stlm(modelfunction = thetaf)).

2.6 Arima (RF_arima_test.csv), initially called (RF_arima_full_test.csv)

Auto arima (R forecast) is trained on all rows separately. Parameters that were changed from default: max_order=20, start_d = 0, start_p = 0.

2.7 ETS (RF_ets_test.csv)

ETS (R forecast) is trained on all rows separately. Default parameters.

2.8 Damped (ETS) (RF_damped_test.csv)

ETS (R forecast) is trained on all rows separately, where damped parameter is set to True.

2.9 RandomF_arima_preprocessed_test.csv

On validation data:
First, a set of transformations is applied to each row, after which auto-ARIMA (R forecast) model is trained and evaluated (6 models in total per row).

For each row, the transformation with the lowest MASE is selected.

The following transformations were applied:

- No transformation
- 1st difference
- Natural logarithm
- 1st difference + natural logarithm
- Box-Cox with automatically selected lambda value
- 1st difference + Box-Cox with automatically selected lambda value

Auto-ARIMA parameters that were changed from default: max_order=20, start_d = 0, start_p = 0.

On training data
For each row, the transformation with the lowest MASE score on validation data is applied. Then, auto-ARIMA (R forecast) model is trained and used for final forecasting.

Auto-ARIMA parameters that were changed from default: max_order=20, start_d = 0, start_p = 0 (same as before).

This method was used only for monthly data.

2.10 Random Forest (RandomF_RandomF1_test.csv)

The random forest model is trained on whole dataset for each frequency (hourly, daily, weekly etc). The models are trained using all the data together, the last column (last timestamp) is used as class “y” for training and previous columns as features “X”.

For implementation sklearn’s method "RandomForestRegressor" was used (http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html)

2.11 XGBoost1 (RandomF_XGBoost1_test.csv)

The XGBoost1 model is trained row by row (timeseries by timeseries). Each row is split using sliding window based on the size of horizon or seasonality. The size of the training features “X” is equal to the size of sliding window as following:

- Hourly: 48 (*2*seasonality*)

This method was used only for hourly data.

For implementation method “XGBoost” was used (http://xgboost.readthedocs.io/en/latest/python/python_intro.html).

2.12 XGBoost2 (RandomF_XGBoost2_test.csv)

The XGBoost2 model is trained on whole dataset for each frequency (hourly, daily, weekly etc). The models are trained using all the data together, the last column (last timestamp) is used as class “y” for training and previous columns as features “X”.

For implementation method “XGBoost” was used (http://xgboost.readthedocs.io/en/latest/python/python_intro.html).

2.13 Correlator (s_predictions_corr5__full_final.csv)

For each row, looks at a window of time (W1) at the end of the row and finds the most closely correlating window of the same length (W2) from all time series within a type (Hourly, Monthly etc.) and uses the time points (H2) immediately following that window to make a prediction for the row.
Prediction H1 = (H2 - mean(W2) / std(W2) * std(W1) + mean(W1)

2.14 Stupidity (stupidity_teamKaur_test.csv)

For each predicted time point T(i) and predefined season length S, the prediction is
T(i) = T(i-S) + T(i-1-S) - T(i-1)
In other words, it uses adds the difference between time points T(i-1-S) and T(i-1), which are exactly one season apart, and adds it to the time point exactly one season prior to T(i)

2.15 TeamPotap('yearly_holdout.csv')

Model is LightGBM. One model per row for a yearly dataset. Considered different sliding window sizes and hyperparameters (grid search with cross-validation). Prediction based on a normalized difference between time steps. Used min-max normalization.

3 Ensembling method descriptions

3.1 Hourly

The ensemble included the predictions from the following methods:
1. Anti_arima_test.csv
2. Fearless5_all_test.csv
3. Hourly_XGBpower_stl_submit.csv
4. Naive1_full_test.csv
5. Naive2_full_test.csv
6. stupidity_teamKaur_test.csv
7. RandomF_XGBoost2_test.csv
8. RandomF_RandomF1_test.csv
9. RF_ets_test.csv
10. RF_damped_test.csv
11. RF_arima_test.csv
12. s_predictions_corr5__full_final.csv

These methods were ensembled using the following method:
If for selected time serie the correlation from correlations_test.csv > 0.995, then use prediction from s_predictions_corr5__full_final.csv, else take the median of other methods.

3.2 Daily

The ensemble included the predictions from the following methods:
1. Anti_arima_test.csv
2. Fearless5_all_test.csv
3. Naive2_full_test.csv.
4. RF_arima_test.csv
5. RF_ets_test.csv.

If for selected time series the correlation from correlations_holdout.csv > 0.9999, then use prediction from s_predictions_corr5__full_final.csv, else take the **median** of these methods.

3.3 Weekly

The ensemble included the predictions from the following methods:
1. Anti_arima_test.csv
2. Fearless5_all_test.csv
3. Naive2_full_test.csv
4. stupidity_teamKaur_test.csv

#Current final_weekly_test.csv contains ensembling results of the 4 models on test data.

These methods were ensembled using the following method:
It is used **median** for ensembling. The OWA of ensembling the results of 4 best methods on holdout data is 0.779.

3.4 Monthly

The ensemble included the predictions from the following methods:
1. RF_ets_test.csv
2. RF_damped_test.csv
3. RF_arima_test.csv
4. RandomF_arima_preprocessed_test.csv
5. RandomF_RandomF1_test.csv
6. RandomF_XGBoost2_test.csv
7. Naive1_full_test.csv
8. Naive2_full_test.csv
9. Fearless5_all_test.csv
10. Anti_arima_test.csv
11. stupidity_teamKaur_test.csv

These methods were ensembled using the following method:
The outliers were removed and then 3 or 4 values in the middle were averaged. So for one timestamp with values like [1,1,2,3,5,6,7,10,11,12,13] -> crop the outliers (we called it choose k-medians) [5,6,7] -> find the mean of these -> 6. On holdout we got OWA: 0.835.

3.5 Quarterly

The ensemble included the predictions from the following methods:
1. anti_arima
2. fearless5

These methods were ensembled using the following method:
mean

3.6 Yearly

Combination of Median, Mean ensembles based on performance on holdout.

- Kaur correlations: only picked those of higher correlation (corr > 0.999) which is about 14K samples for yearly
    - It is not used directly but the performance checked on the holdout and used on the testset (at this stage it may not make much of difference because apparently for this threshold the same rows are selected from both holdout and test)
- Predictions worse than naive and 2*naive are eliminated right away
- Also predictions containing at least a negative prediction is dropped for that row
- Combination of predictions (using median/mean) used for ensembling
- Error of each ensembling calculated on the fly on the hold out and Naive1 and naive2 are used when ensemble is not better
- Processes is applied for the test prediction using the performance monitored during the holdout
- Excluding predictions with at least one time step negative value for a row (backup: use naive1)

Full OWA on holdout: 0.565

- RandomF_RandomF1_
- Fearless5_all_
- stupidity_teamKaur_
- Anti_arima_
- RF_damped_full_
- Svmensemble_
- RF_arima_full_
- TeamPotap_Yearly_
- RandomF_XGBoost2_
- RF_ets_full_
- Kaur_correlation (thresh > 0.999 => ~ 14K samples)