Weather forecasting with data science approaches

Andrii Shchur Follow
Oct 7, 2019 · 12 min read ★

In this article, I would like to show how we can use a data science algorithm for weather forecasting and compare some frameworks for classification and regression tasks.

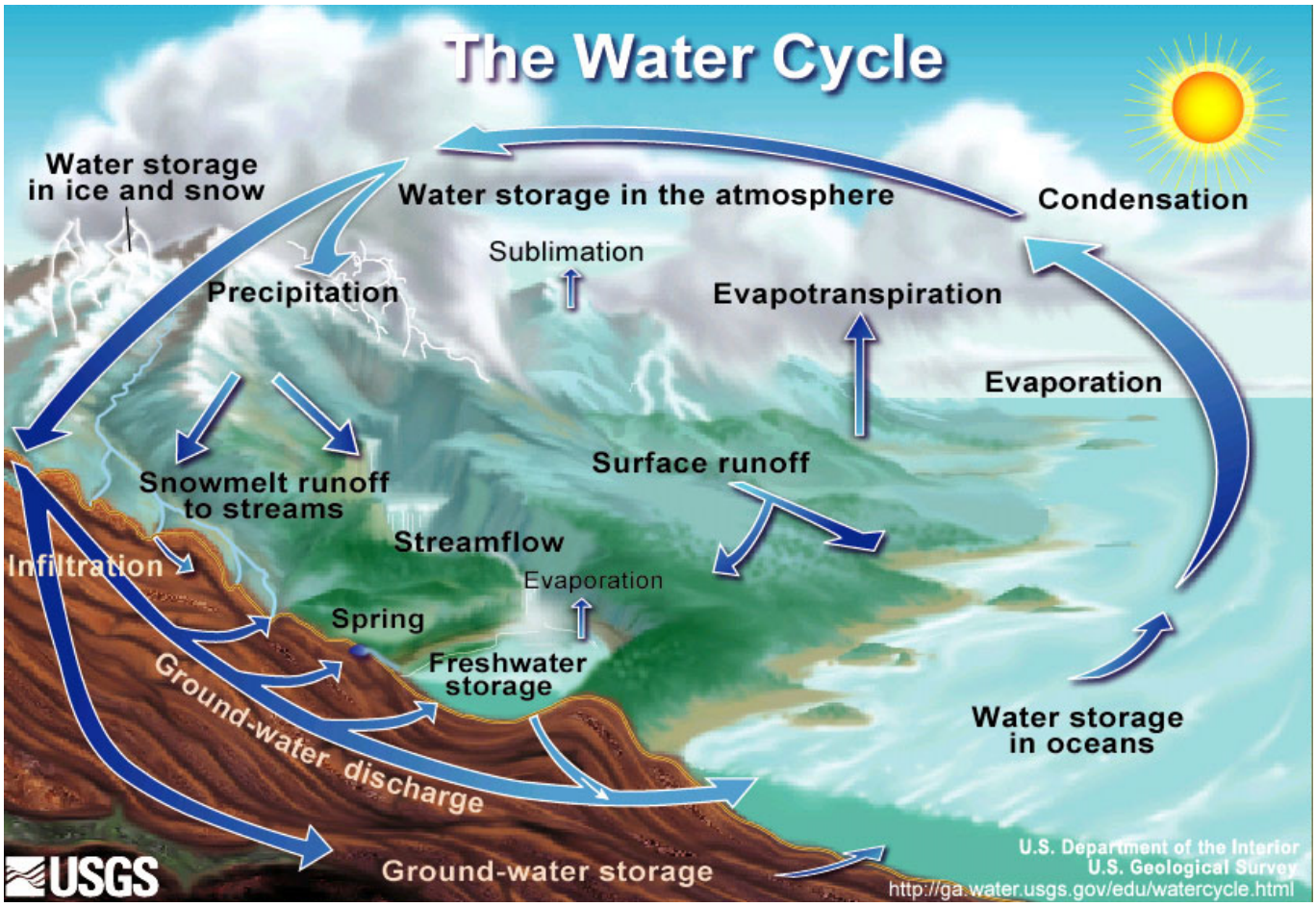
Central Otago District, New Zealand

Weather forecasting *is a quite difficult task. The Wiki said, "* **Weather forecasting** is the application of science and technology to predict the **conditions of the atmosphere** for a given **location** and **time**. **Weather forecasts** are made by collecting **quantitative data**

about the **current state of the atmosphere** at a given place and using meteorology to project how the atmosphere will change."

So in general, weather forecasting is about the data about the atmosphere. Of course, I would not model the atmosphere condition, but I can use a lot of data about it for weather forecasting. In this article, I will try to predict the precipitation for the next 3 hours. I will try to solve this task in two ways — as a binary classification (1- precipitation, 0- no precipitation) and regression, where I will try to predict the amount of precipitation in mm.

So, what is precipitation? Let's ask it in Wiki." In meteorology, **precipitation** is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel, and hail. Precipitation occurs when a portion of the atmosphere becomes saturated with water vapor so that the water condenses and "precipitates". " Let look at the schema, which shows the process of the water cycle and precipitation occurrence.



Condensation and coalescence are important parts of the water cycle.

Of course, the precipitation occurrence is a very complicated process and those schema does not show the whole process. The Wiki said, “Precipitation is a major component of the water cycle and is responsible for depositing most of the freshwater on the planet. Mechanisms of producing precipitation include convective, stratiform, and orographic rainfall. Convective processes involve strong vertical motions that can cause the overturning of the atmosphere in that location within an hour and cause heavy precipitation, while stratiform processes involve weaker upward motions and less intense precipitation. Precipitation can be divided into three categories, based on whether it falls as liquid water, liquid water that freezes on contact with the surface, or ice. Mixtures of different types of precipitation, including types in different categories, can fall simultaneously. Liquid forms of precipitation include rain and drizzle. Rain or drizzle that freezes on contact within a subfreezing air mass is called “freezing rain” or “freezing drizzle”. Frozen forms of precipitation include snow, ice needles, ice pellets, hail, and graupel.”

Exploratory data analysis

So, let’s try to find some data and try to predict this natural phenomenon. In my experiment, I will use data from one meteorology station. Let’s examine the header of data.

So, I have got **29 variables** and **22 064 records**. Let me describe the column names:

- **T** — Air temperature (degrees Celsius) at a 2-meter height above the earth\’s surface;
- **Po** — Atmospheric pressure at weather station level (millimeters of mercury);
- **P** — Atmospheric pressure reduced to mean sea level (millimeters of mercury);
- **Pa** — Pressure tendency: changes in atmospheric pressure over the last three hours (millimeters of mercury);
- **U** — Relative humidity (%) at a height of 2 meters above the earth\’s surface;
- **DD** — Mean wind direction (compass points) at a height of 10–12 meters above the earth’s surface over the 10 minutes immediately preceding the observation;

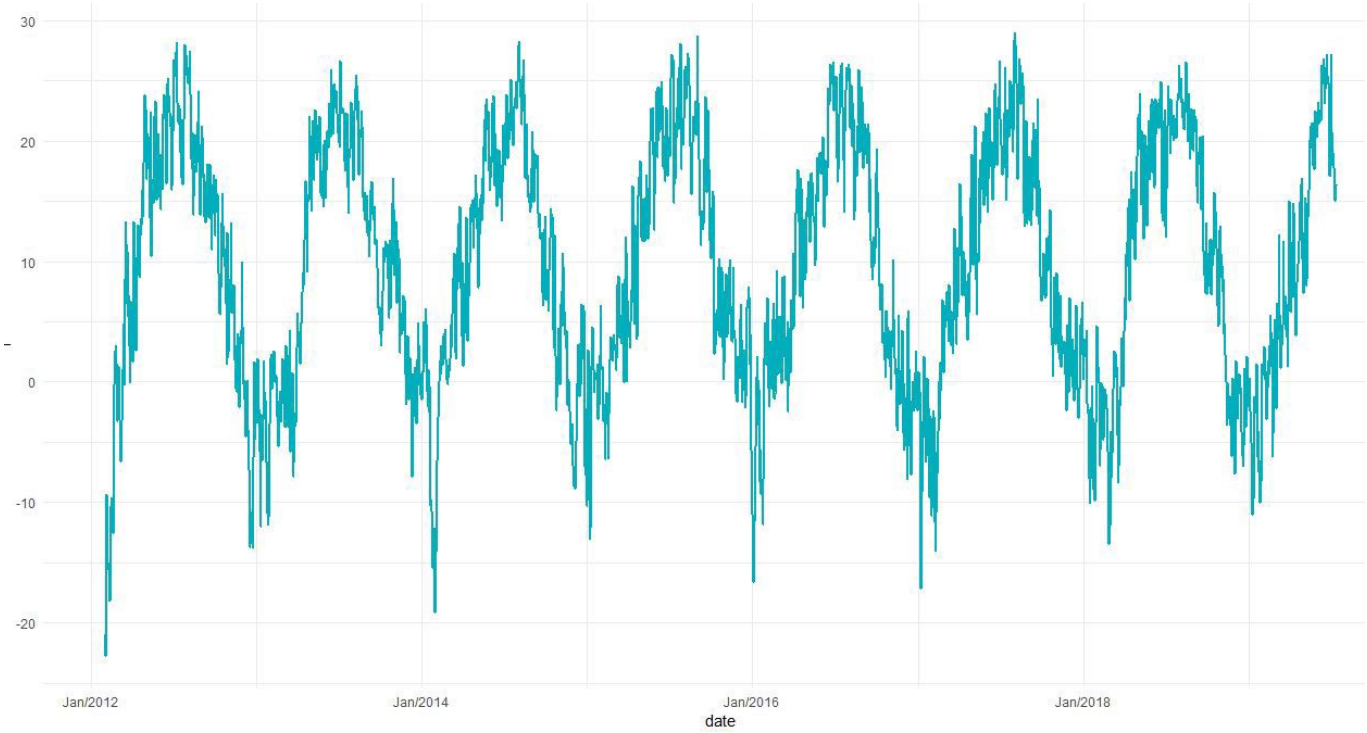
- **Ff** — Mean wind speed at a height of 10–12 meters above the earth's surface over the 10 minutes immediately preceding the observation (meters per second);
- **ff10** — Maximum gust value at a height of 10–12 meters above the earth's surface over the 10 minutes immediately preceding the observation (meters per second);
- **ff3** — Maximum gust value at a height of 10–12 meters above the earth's surface between the periods of observations (meters per second);
- **N** — Total cloud cover;
- **WW** — Present weather reported from a weather station;
- **W1** — Past weather (weather between the periods of observation) 1;
- **W2** — Past weather (weather between the periods of observation) 2;
- **Tn** — Minimum air temperature (degrees Celsius) during the past period (not exceeding 12 hours);
- **Tx** — Maximum air temperature (degrees Celsius) during the past period (not exceeding 12 hours;
- **Cl** — Clouds of the genera Stratocumulus, Stratus, Cumulus, and Cumulonimbus;
- **Nh** — Amount of all the CL cloud present or, if no CL cloud is present, the amount of all the CM cloud present;
- **H** — Height of the base of the lowest clouds (m);
- **Cm** — Clouds of the genera Altocumulus, Altostratus, and Nimbostratus;
- **Ch** — Clouds of the genera Cirrus, Cirrocumulus, and Cirrostratus;
- **VV** — Horizontal visibility (km);
- **Td** — Dewpoint temperature at a height of 2 meters above the earth\'s surface (degrees Celsius);
- **RRR** — Amount of precipitation (millimeters);

- **tR** — The period during which the specified amount of precipitation was accumulated;
- **E** — State of the ground without snow or measurable ice cover;
- **Tg** — The minimum soil surface temperature at night. (degrees Celsius);
- **E** — State of the ground with snow or measurable ice cover;
- **sss** — Snow depth (cm).

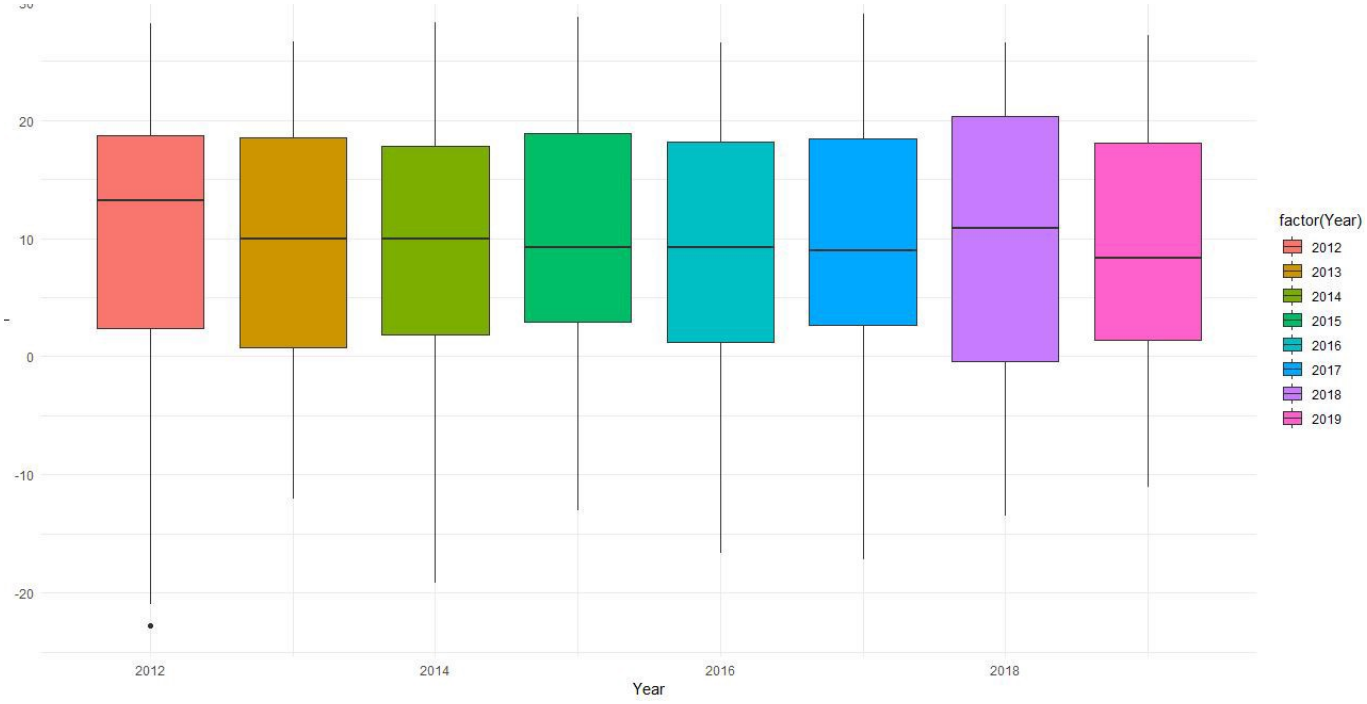
First of all, I would like to know the summary of this dataset and then select a useful column for further analysis.

I replace NULL numeric value with mean and then I can visualize this dataset. Let's start with Temperature(T).

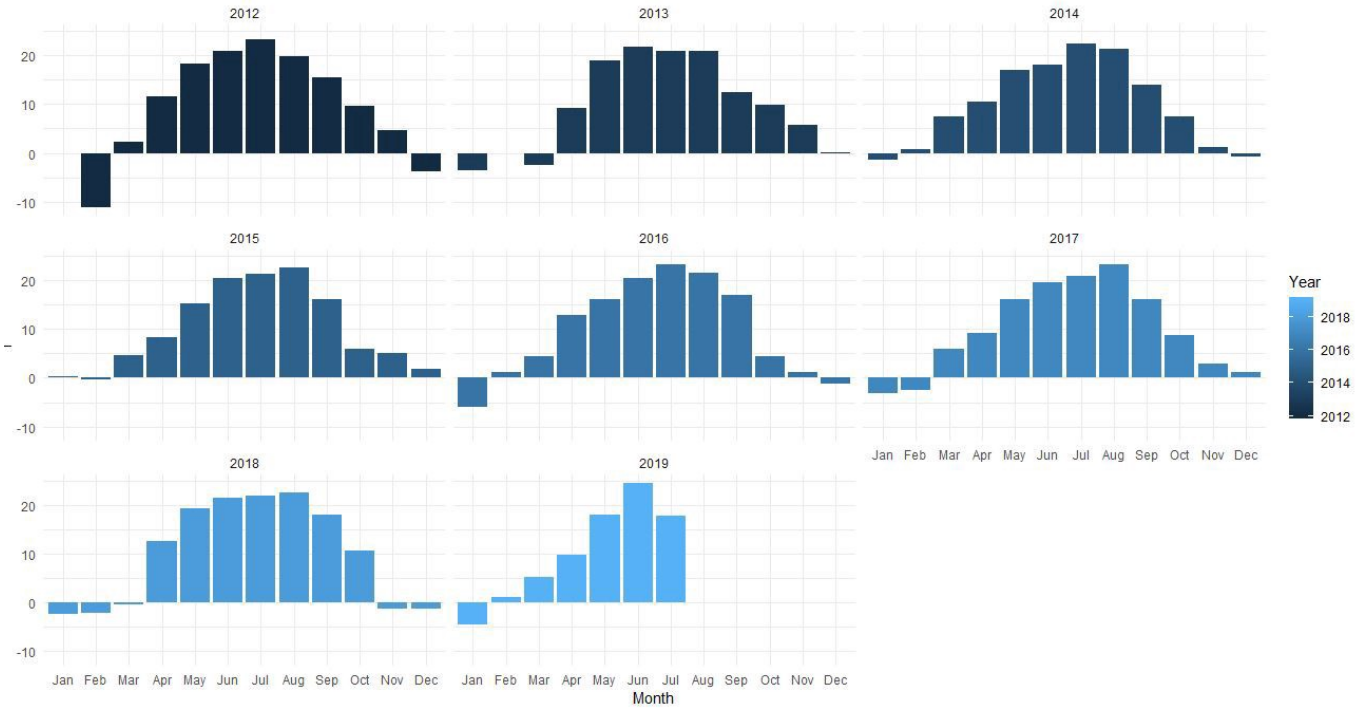
First of all, I would like to plot the time series of Temperature. On the plot, you can see how Temperature changes during the "2012–02–01" — "2019–07–13".



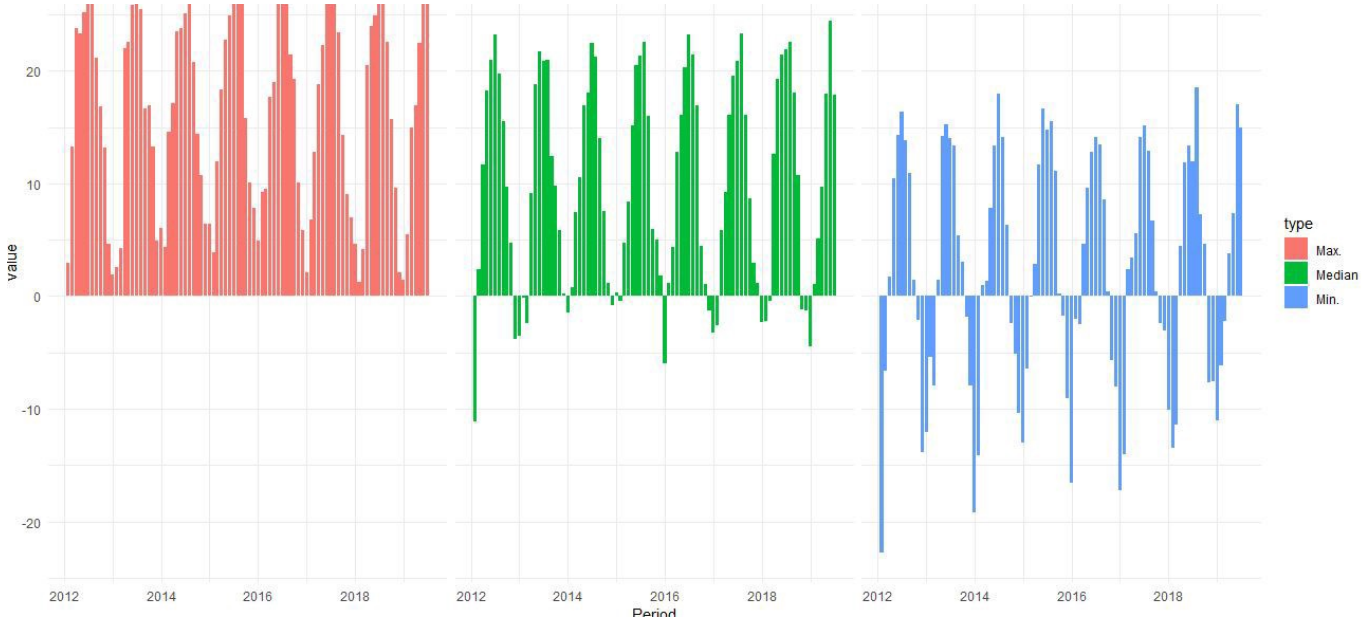
Here we can find seasonal of our data, but I would like to examine it more detail. For examining the changes year by year I create a box plot.



The changes are very little on this chart, so let's examine it by month and year.

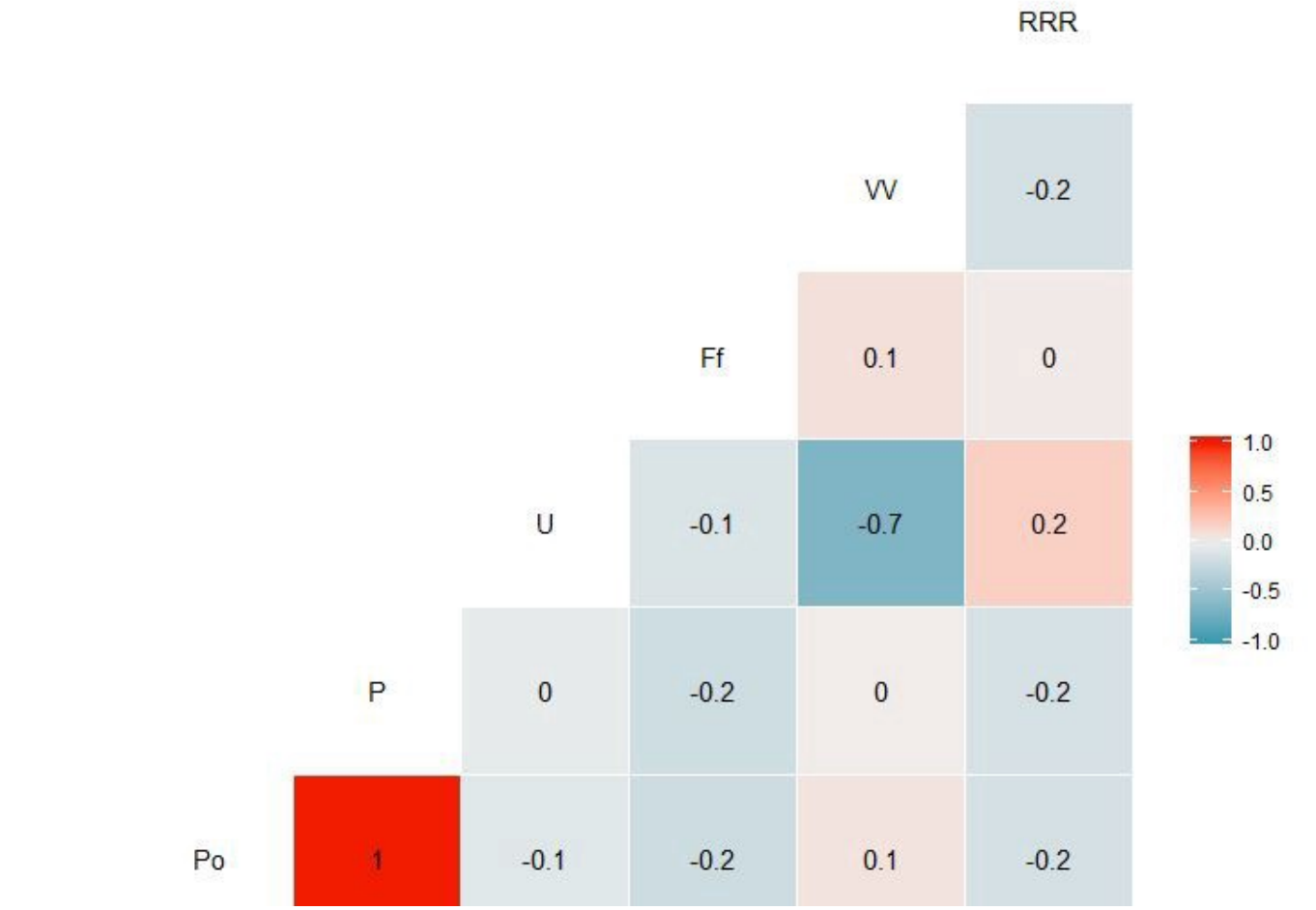


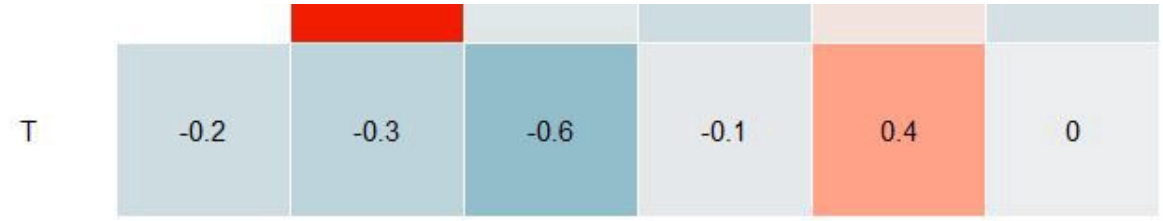
Wow, It was interesting to find that summer in most years is similar, but autumn and spring are different. Also, it would be interesting to plot some trends in maximum, minimum and median temperature overall period.



It is interesting, that the minimum temperature for the last years became higher. For more details, you can examine it in the next table.

The next stage of my data analysis is the calculation of the correlation between all numeric values.



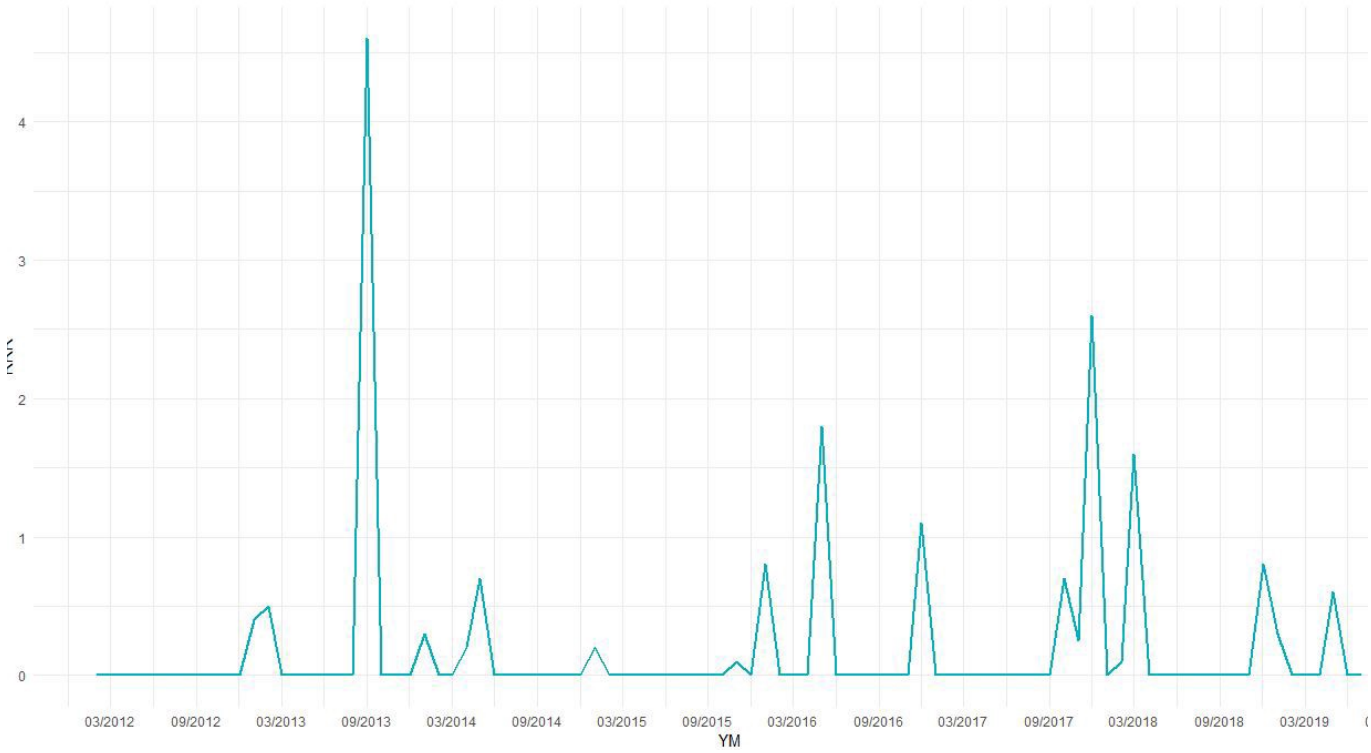


On this plot, we can see three interesting correlation:

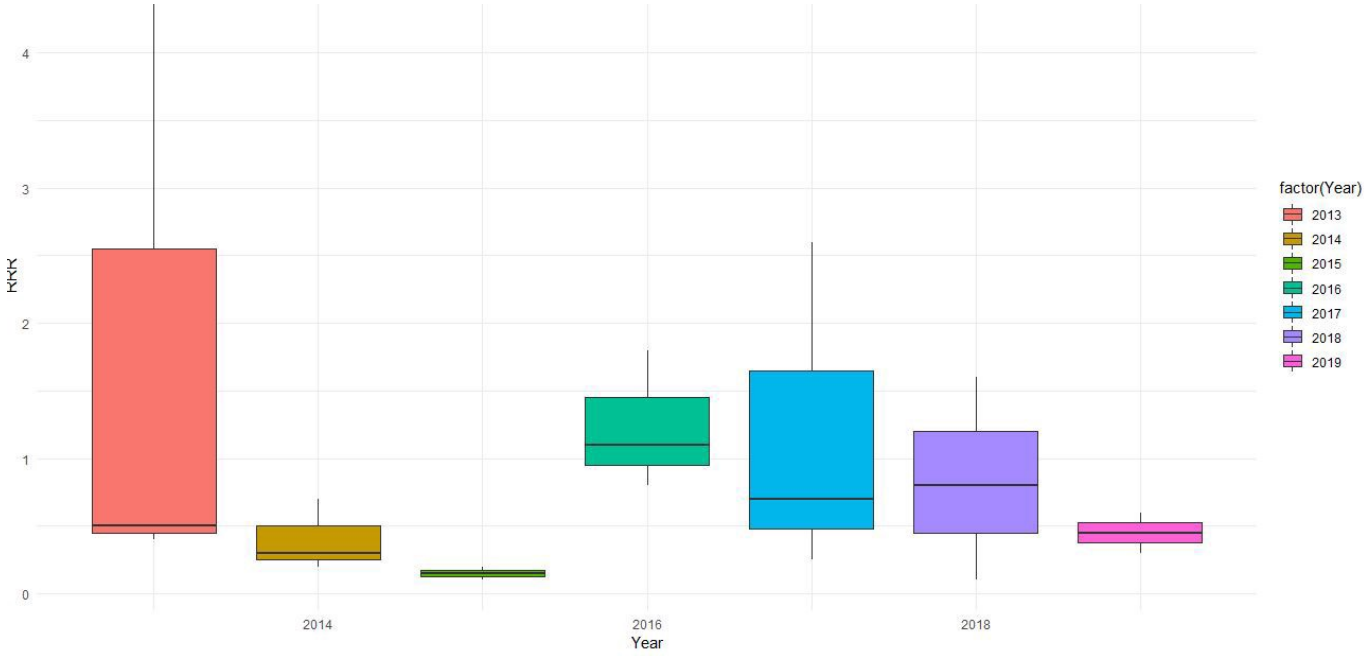
- The positive correlation (0.4) between Temperature and Horizontal visibility;
- The negative correlation (-0.6) between Temperature and Relative humidity;
- The negative correlation (-0.7) between Relative humidity and Horizontal visibility.

Also, I would like to analyze more detail Amounts of precipitation and to plot a similar to Temperature charts.

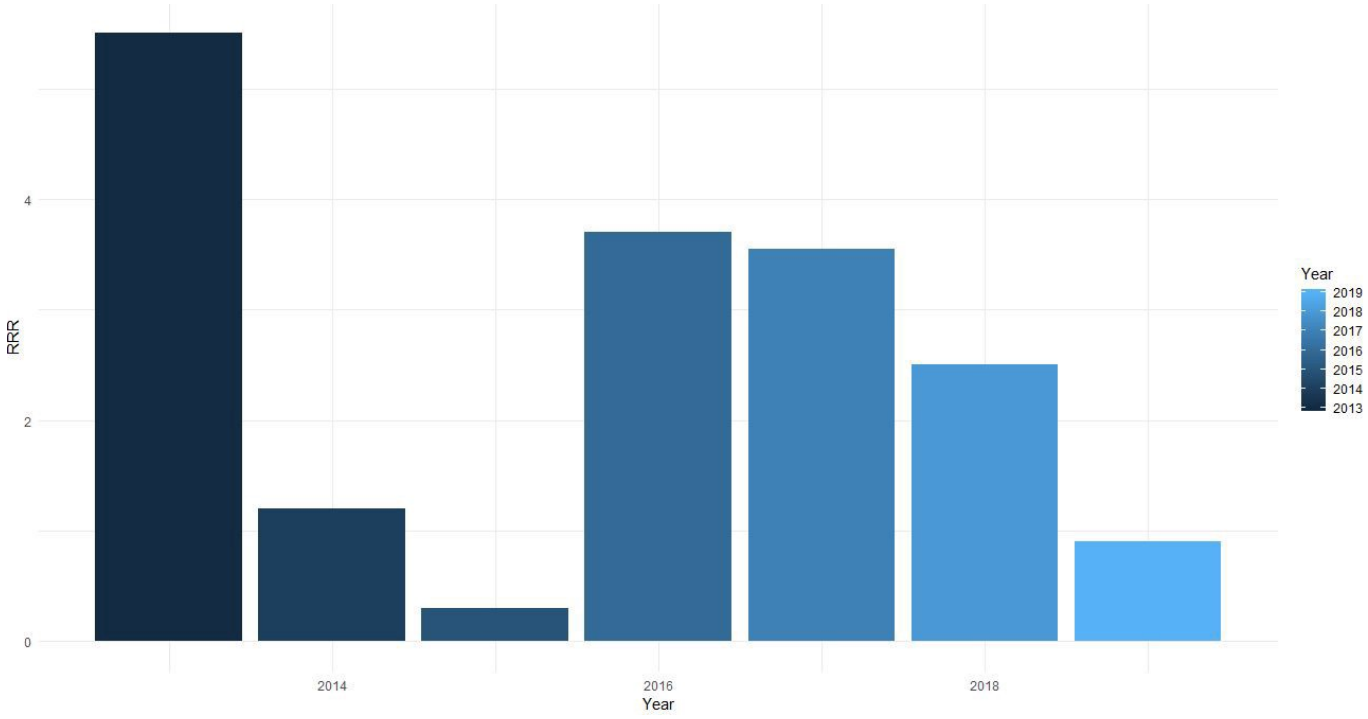
Let's begin with the time series plot.



The summary of precipitation in autumn 2013 was the highest for the last 5 years. Let's calculate summary statistics for non zero precipitation values.

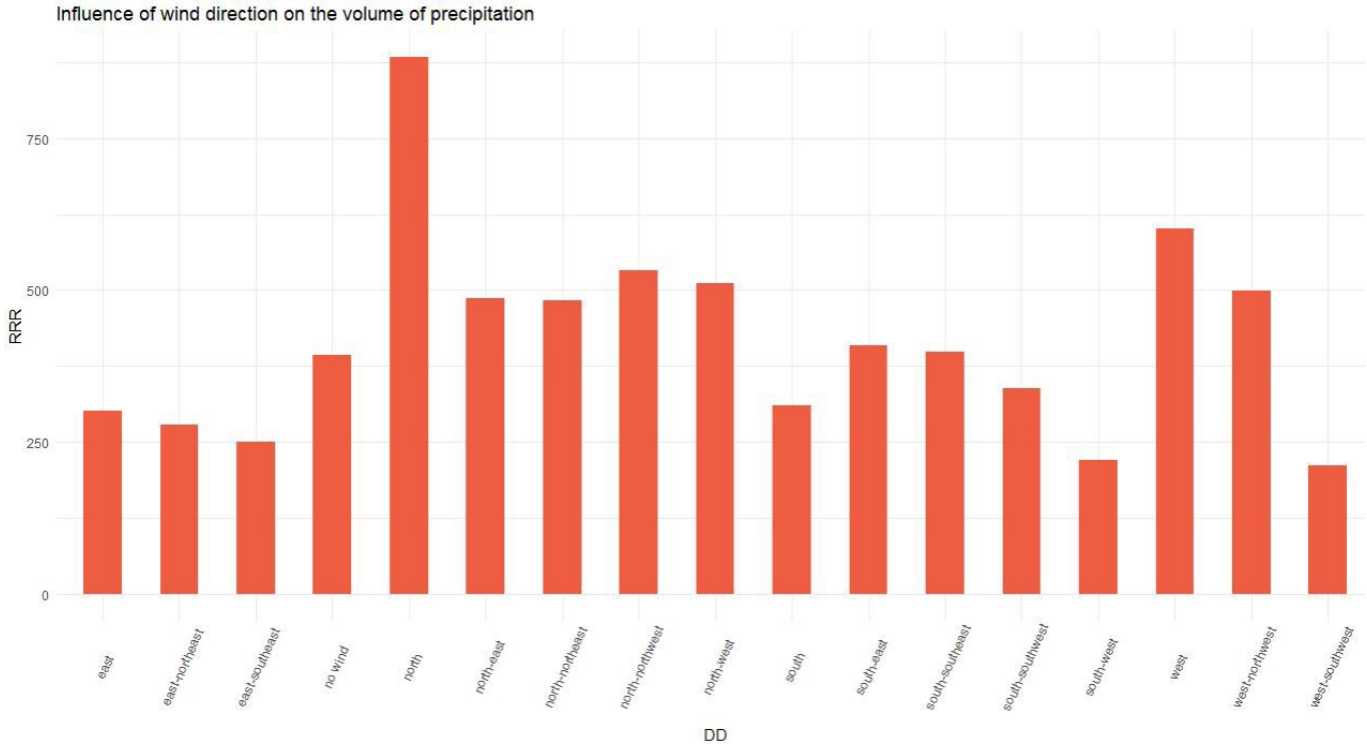


And also I would like to know the summary value of precipitation overall period.



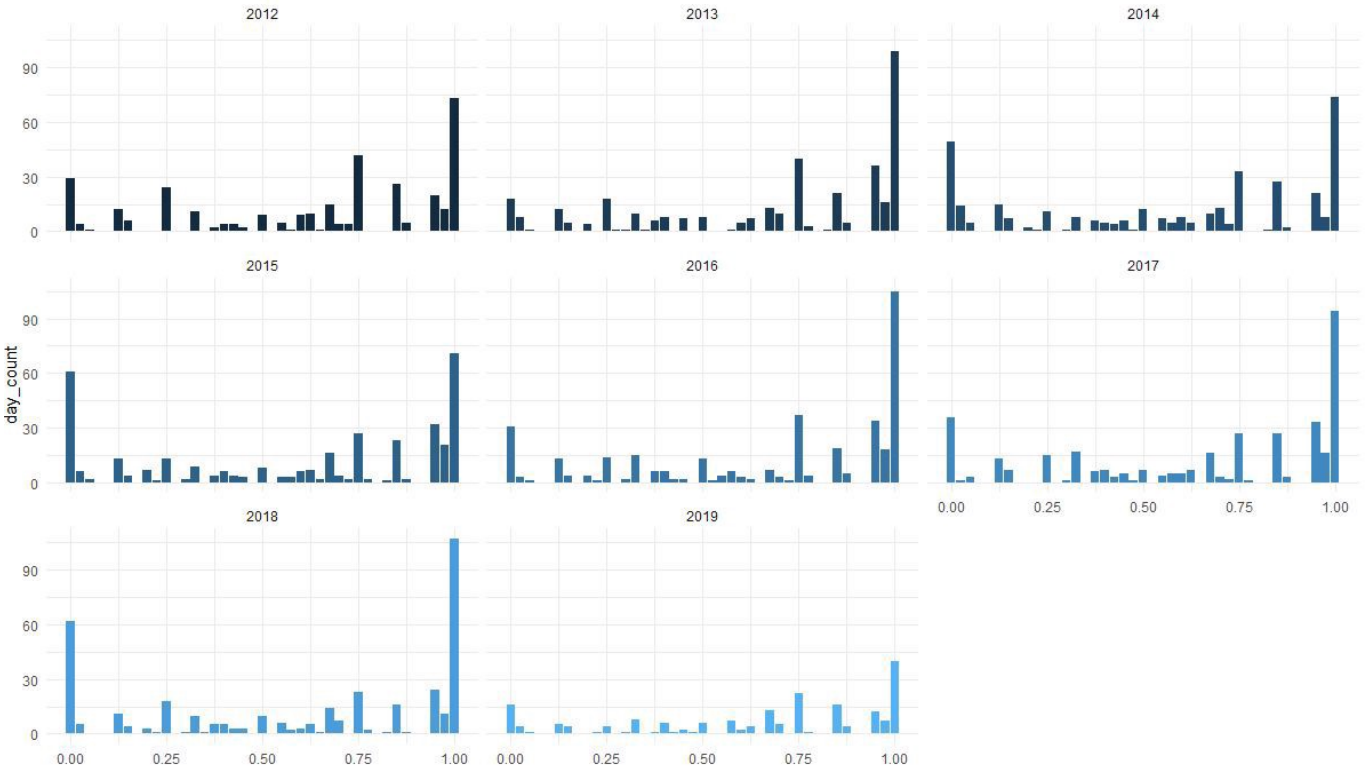
Here we can see that in 2014 and 2015 were the lowest level of precipitation in this period.

What about categorical data and its influence on precipitation? Let's start with wind direction (DD). We have got 16 different directions in the dataset and one event with no wind.

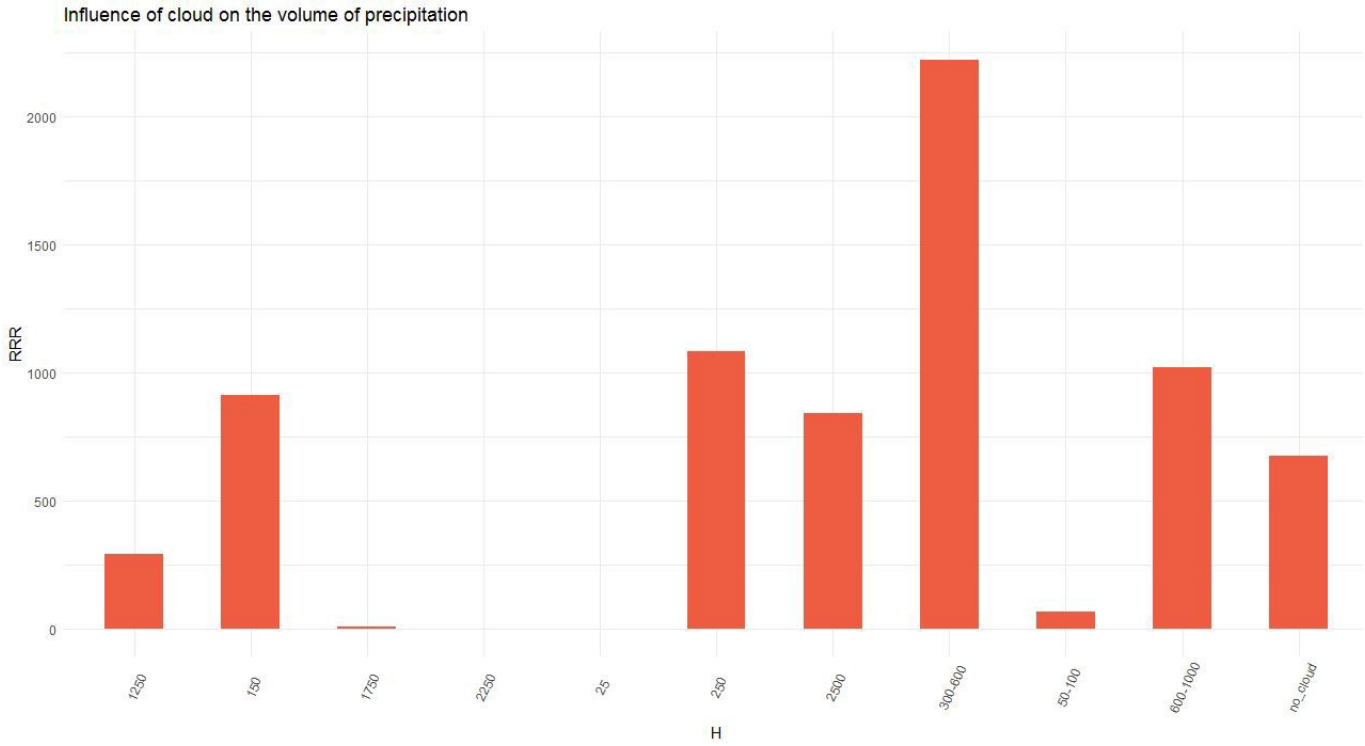


We can see that most precipitation may come from north or north-west.

The next stage is the analysis of the cloud. First of all, I would like to analyze the count of a day with cloud and analysis of the influence of cloud characteristics on the precipitation.



Cloud coverage is a very similar overall period and mostly it is not changed. Let's also analyze cloud characteristics.



This chart shows, that most precipitation brings with cloud on high 300–600m. That's all, that I want in exploratory data analysis.

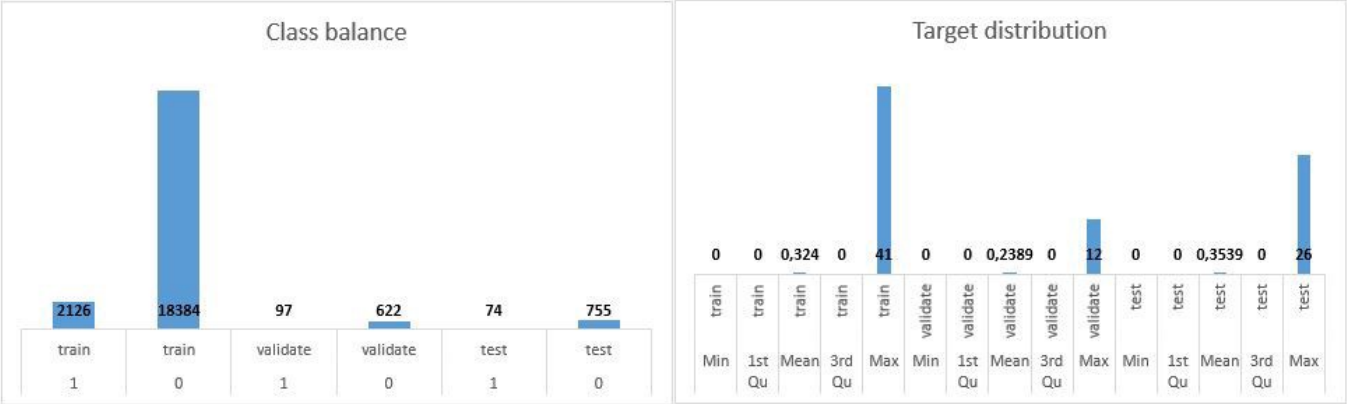
Feature engineering and target preparation

Firstly, I calculate the date feature — quarter, month, week, weekday, month day, year day, the hour. All this feature I can calculate with the lubridate library.

The second stage of feature engineering is the calculation of lag value for some numerical features. For my analysis, I would like to calculate the lag value for T, Po, P, U, Ff, N, Nh, VV and my target RRR. It would be 3 lag on a particular day and 1 lag for the previous day. Final data set:

So, it is time to split the dataset for the train, validation, and test subset. Train data set in the range "2012–02–02 02:00:00"-"2018–12–31 23:00:00", validation data set in the range "2019–01–01 02:00:00"-"2019–03–31 21:00:00", and test data set in the range "2019–04–01 00:00:00"-"2019–07–13 15:00:00".

Also, it is important to check the balance of classes in the subset and distribution of the target value for the regression task.



So, as a result, we have got unbalanced data sets for classification and a little bit different distribution for regression.

Modeling

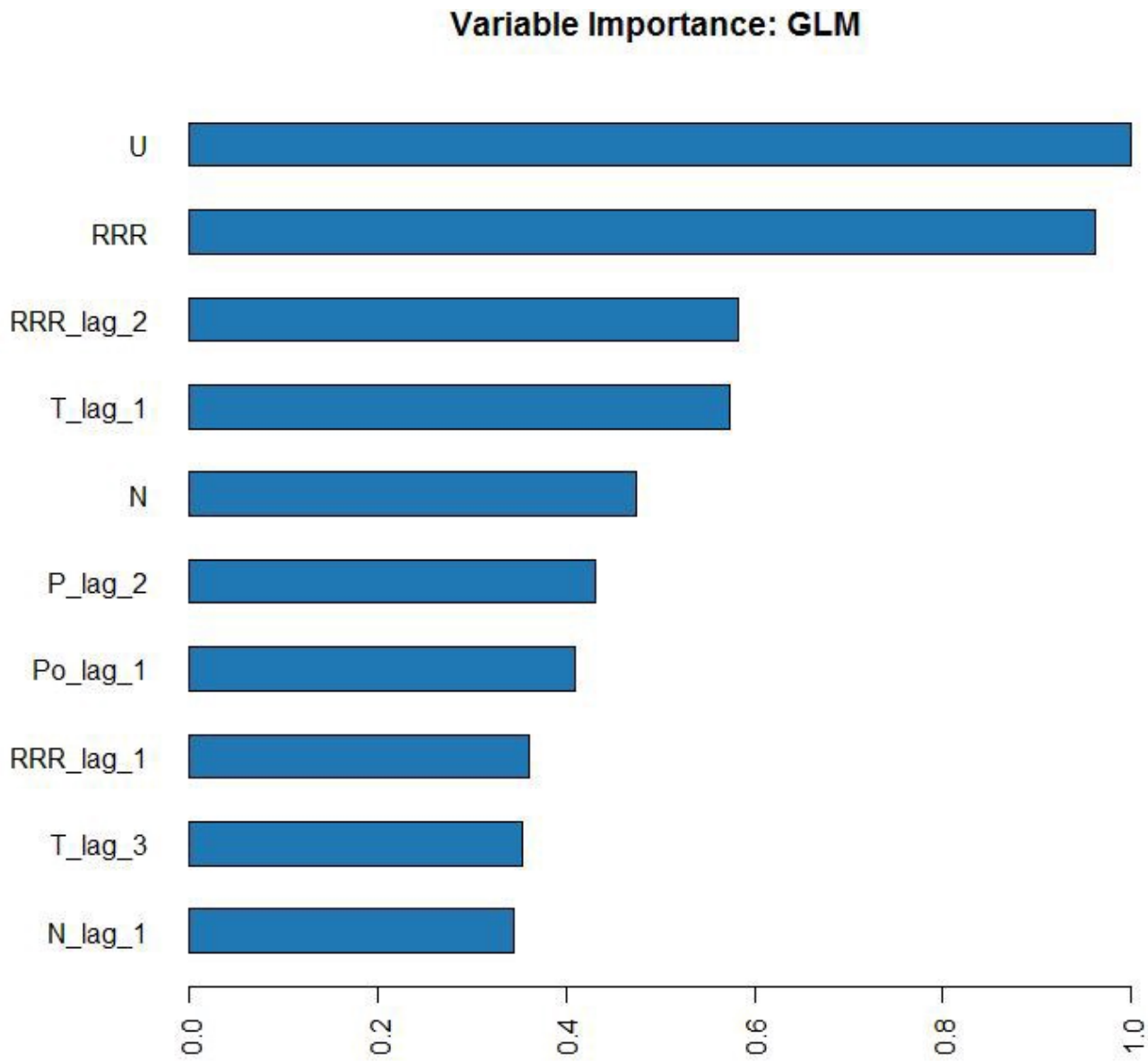
Classification

Ok, let's start modeling. The framework, that I would like to test is H2O.

H2O is a fully open-source, distributed in-memory machine learning platform with linear scalability. H2O supports the most widely used statistical & machine learning algorithms including gradient boosted machines, generalized linear models, deep learning and more. H2O also has an industry-leading AutoML functionality that automatically runs through all the algorithms and their hyperparameters to produce a leaderboard of the best models.

In my experiment, I would like to use gradient boosted machines, generalized linear model, deep learning and AutoML. One of the H2O benefits is an integrated analysis of the importance of factors.

My first model would be — the generalized linear model for the classification task. As we see before our label is unbalanced, so we need to set *balance_classes = TRUE*. Also for classification, we need to set *family = 'binomial'*. So, let's build it and look at the importance of the features.

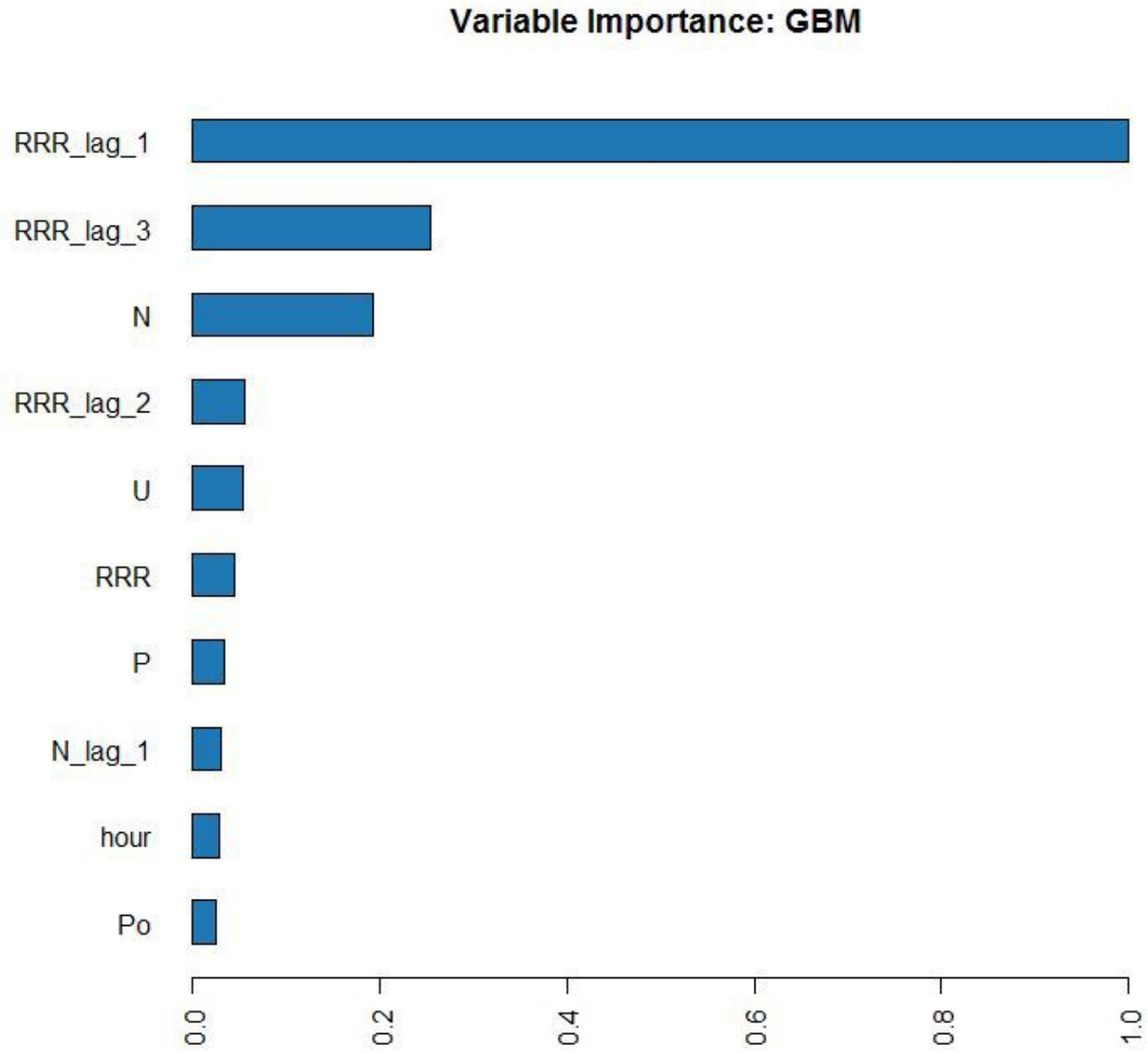


As we can see, the most important factor is **U** humidity one hour before precipitation. And what about accuracy? Let's calculate all main metrics:

```
"Accuracy -  0.9252"
"Precision -  0.9603"
"True negative rate -  0.5676"
"Recall -  0.9577"
"F1_Score -  0.959"
"AUC -  0.7639"
```

So in general, our model predicts good, but the precipitation we can predict only in 56%.

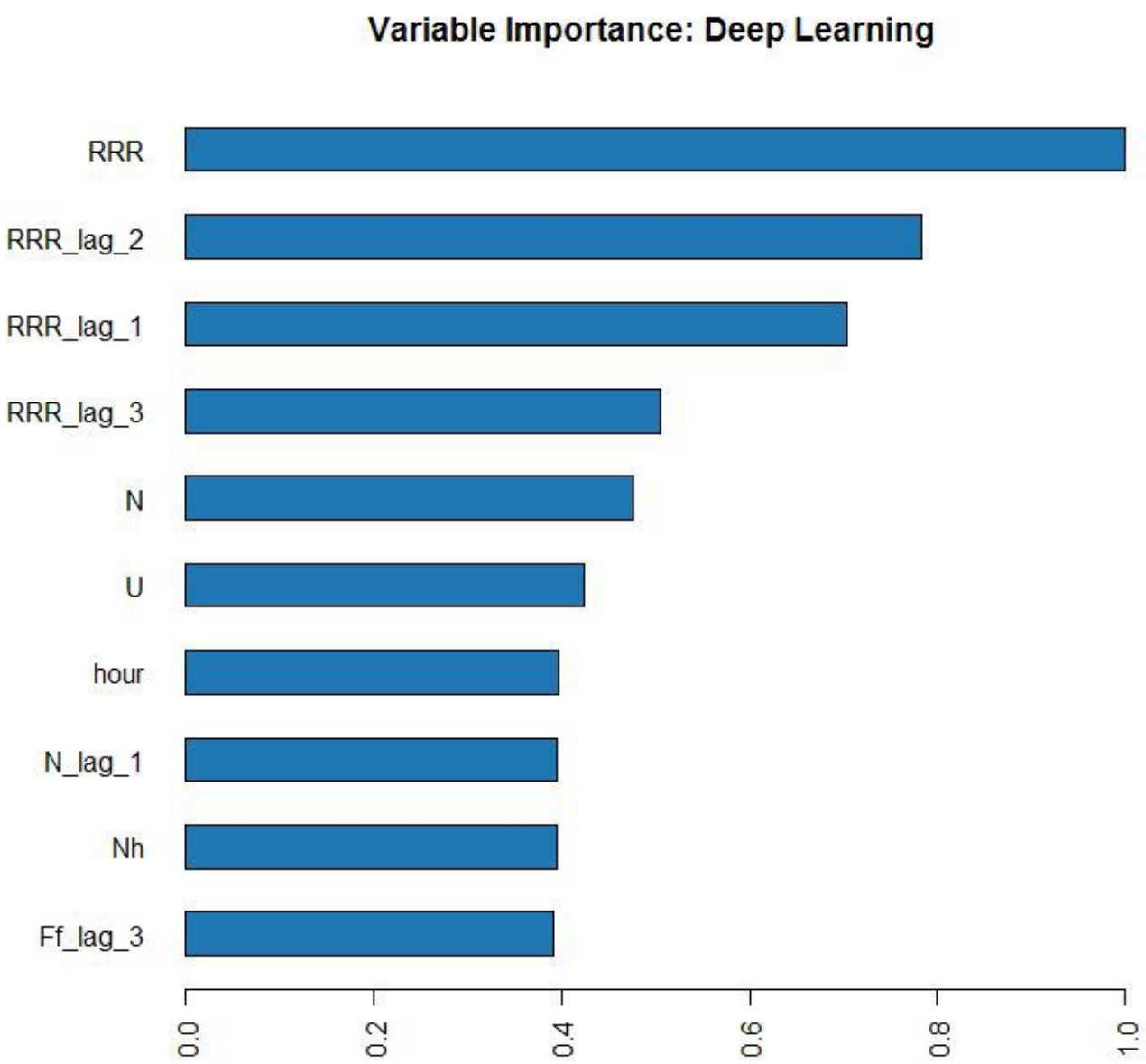
Let's try another model. My second model is gradient boosted machines. I also to set *balance_classes = TRUE*. The importance of the features will look like:



The most important factors are different lags of precipitation and the cloud cover. And what about the accuracy of this model:

```
"Accuracy -  0.9216"
"Precision -  0.9272"
"True negative rate -  0.8649"
"Recall -  0.9859"
"F1_Score -  0.9556"
"AUC -  0.896"
```

The result is better. We can predict precipitation more accurate. I would like to try a more complicated model. It would be a deep learning algorithm, which H2O also provided. I also to set *balance_classes = TRUE*. The importance of the features will look like:



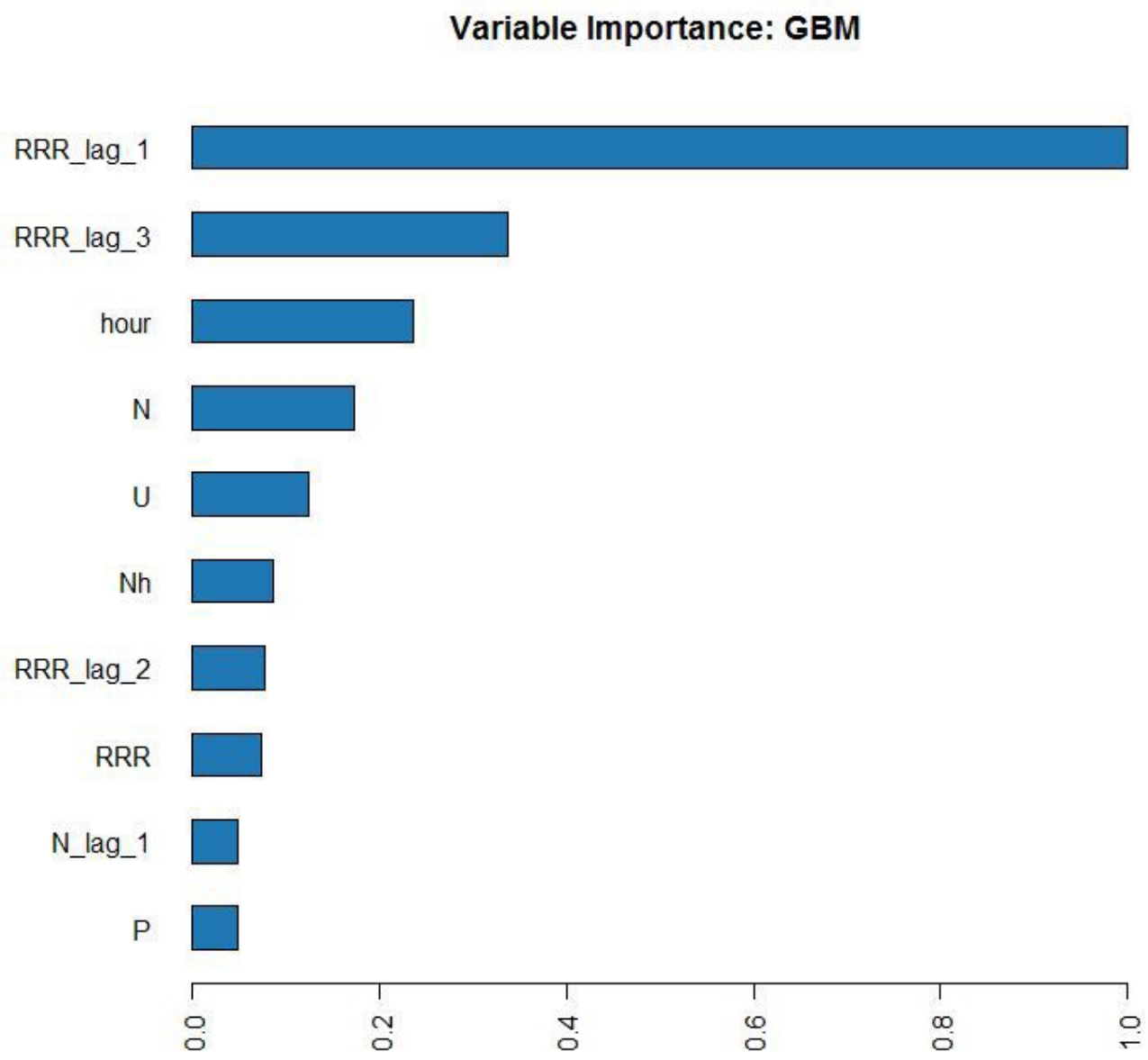
Wow, it is interesting the most important factors if all lag of precipitation and only then are humidity and cloud coverage. And what about the accuracy:

```
"Accuracy -  0.9312"
"Precision -  0.955"
"True negative rate -  0.6892"
"Recall -  0.9691"
"F1_Score -  0.962"
"AUC -  0.8221"
```

The result is worst then in GBM algorithm it can be connected with my deep learning model overfitting.

H2o also has got Automated machine learning. Let's build 50 models and look at the results. TOP 5 models that were created — GBM with different hyperparameters. As

usual. let's look at the most important factors:



You can see factors importance are very similar to my own GBM model, and what about accuracy:

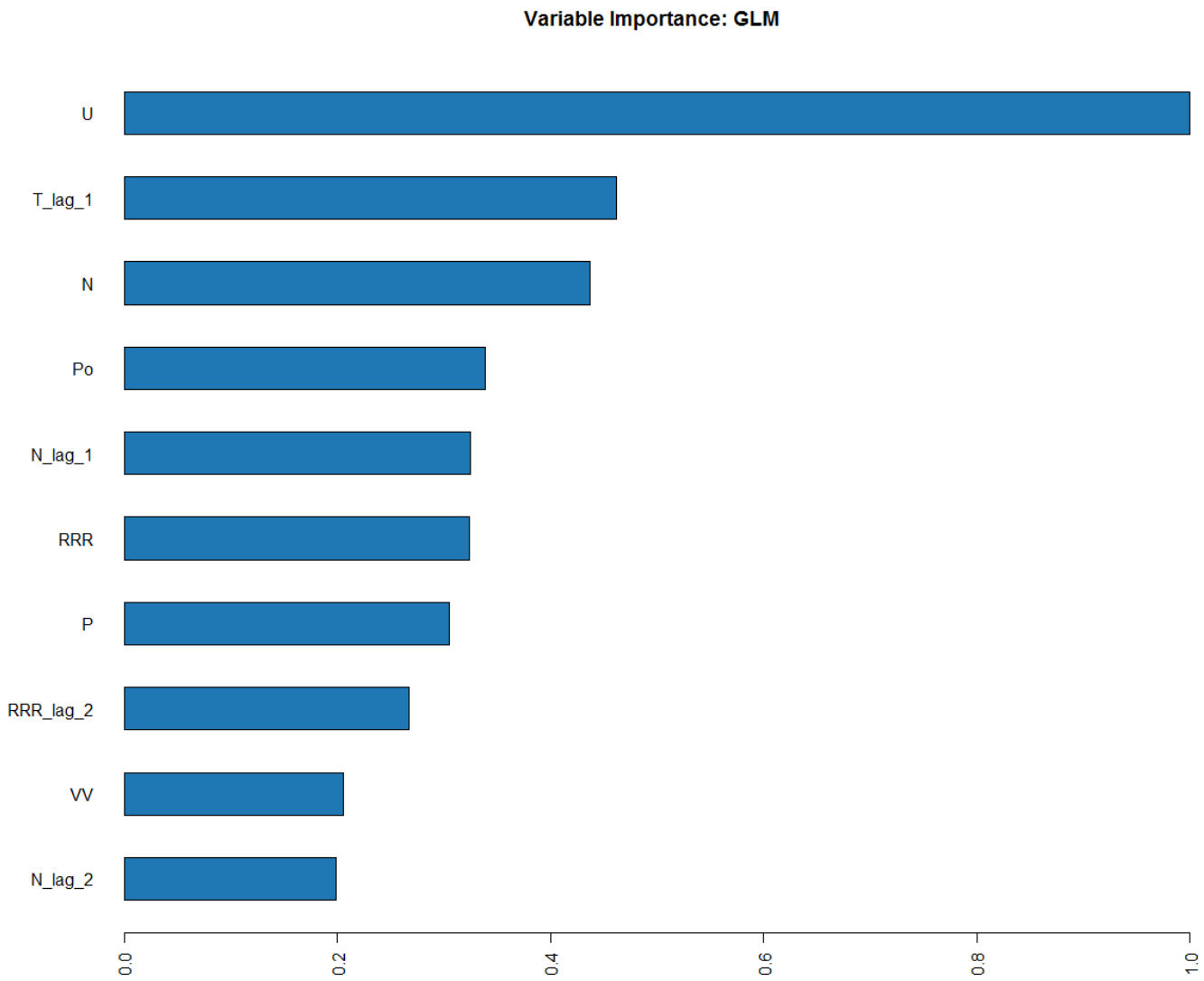
```
"Accuracy -  0.9517"
"Precision -  0.9629"
"True negative rate -  0.8378"
"Recall -  0.9838"
"F1_Score -  0.9732"
"AUC -  0.9004"
```

AUC is better than in my own GBM model, but TNR is worst. Also, it took more time for calculation.

Regression

Let's build the regression model and try to predict the amount of precipitation.

Our first model is linear regression. As in the classification task, we can look at the features importances:

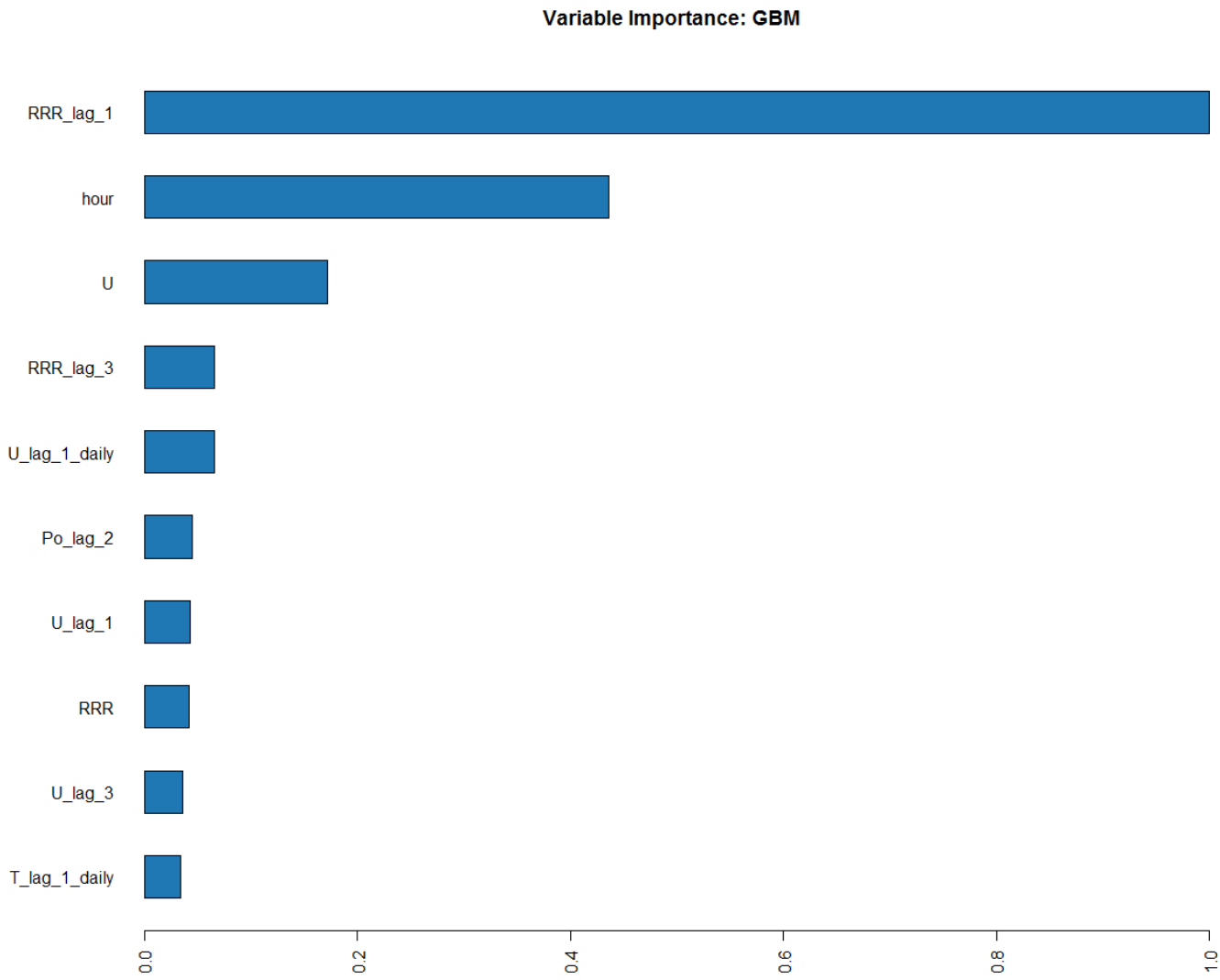


The humidity has the most significant value. And what about accuracy? In this task, I would like to use 3 metrics: MAPE, MAE, MSE.

```
"Accuracy -  0.415"
"MAE -  0.4509"
"MSE -  3.4099"
"R2 -  0.2256"
```

The result of the GLM model is pretty bad.

Let's try another model — gradient boosted machines. Firstly, I would like to look at the features' importance:

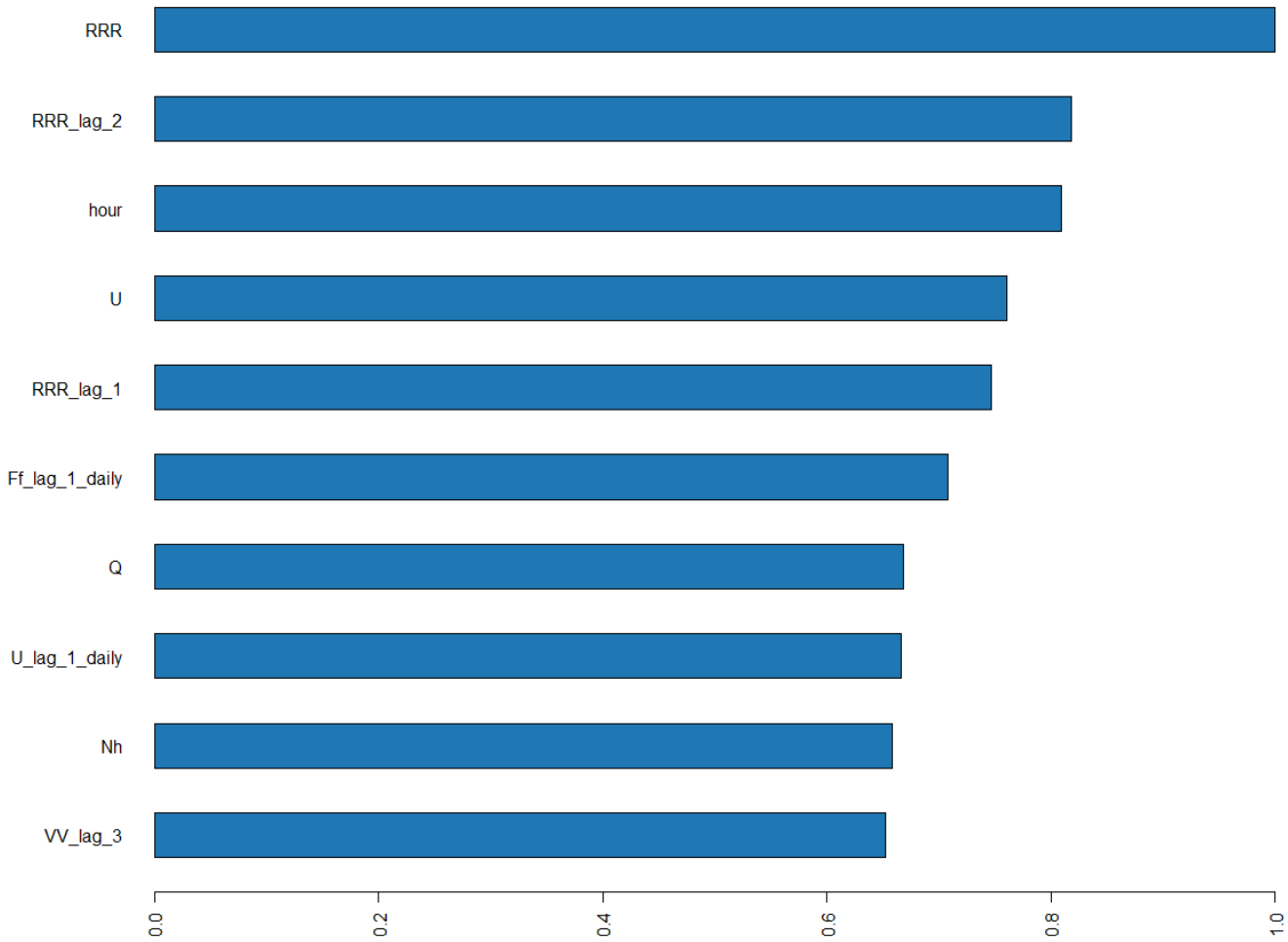


It is different from the GLM model, here we can see that feature **precipitation in the previous hour** has the most value.

The metrics are pretty good:

```
"Accuracy -  0.6634"
"MAE -  0.3019"
"MSE -  2.2242"
"R2 -  0.4949"
```

Our complicated model as in classification task would be deep learning. The features importance looks like this:



The most important features are the amount of **precipitation in the previous and current hours**. What about accuracy:

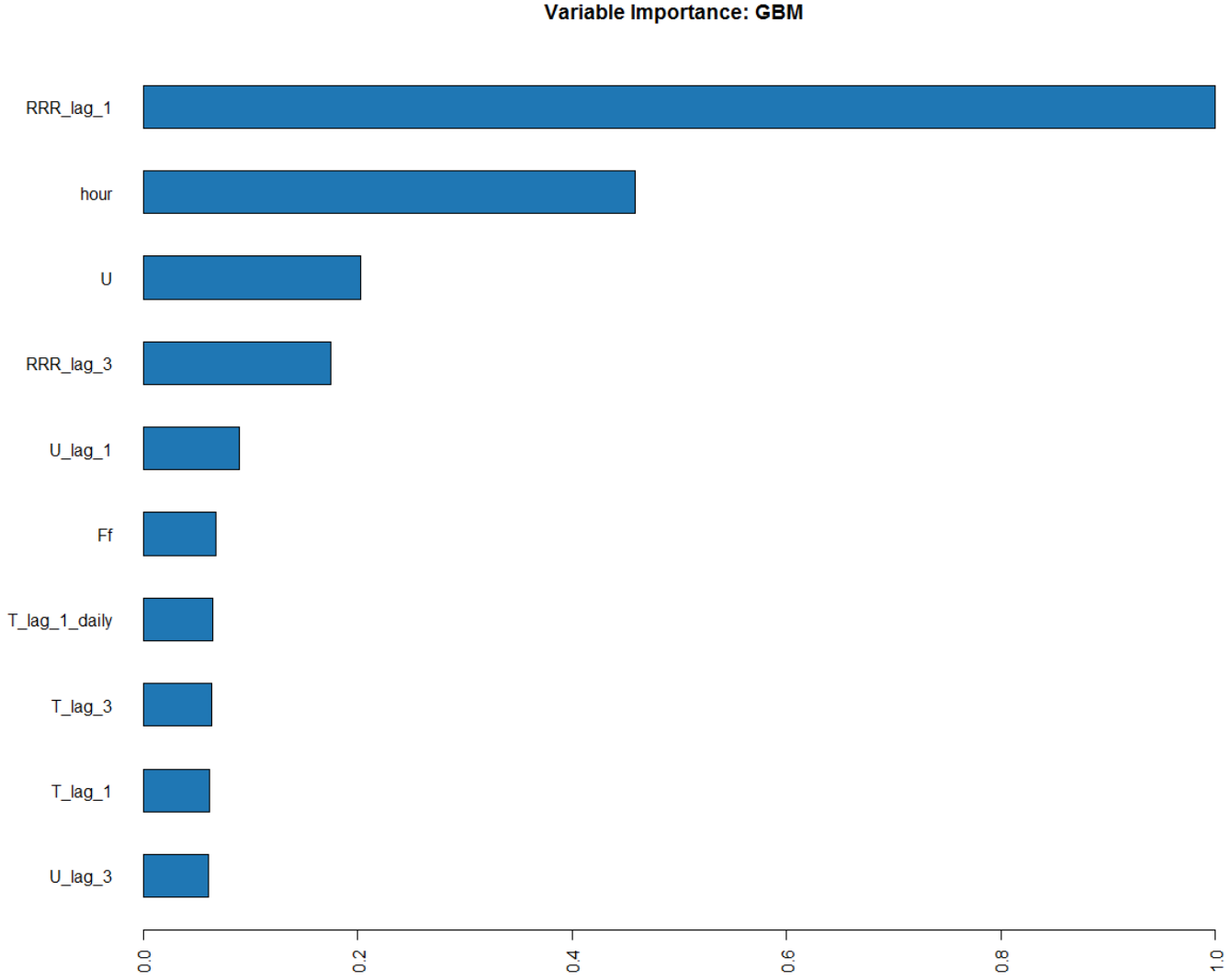
```
"Accuracy -  0.5935"
"MAE -  0.4426"
"MSE -  3.1094"
"R2 -  0.2938"
```

The accuracy is lover then in GBM model, I think it is because of DL is too complicated.

The last one is AML. TOP 5 models are:

model_id	mean_residual_deviance	rmse	mse	mae	rmsle
StackedEnsemble_AllModels_AutoML_20191007_113408	1.329939	1.153230	1.329939	0.3246897	0.2907141
StackedEnsemble_BestOfFamily_AutoML_20191007_113408	1.374510	1.172395	1.374510	0.3281313	NaN
GBM_2_AutoML_20191007_113408	1.406519	1.185967	1.406519	0.3179825	NaN
GBM_grid_1_AutoML_20191007_113408_model_3	1.416175	1.190031	1.416175	0.3312299	NaN
GBM_1_AutoML_20191007_113408	1.425891	1.194107	1.425891	0.3215260	NaN
GBM_4_AutoML_20191007_113408	1.460572	1.208541	1.460572	0.3245931	NaN

As usual. let's look at the most important factors of GBM model from AML:



The result is very similar to my GBM model.

Let's took at the results of the best AML model — StackedEnsemble_AllModels_AutoML:

```
"Accuracy -  0.6779"
"MAE -  0.3604"
"MSE -  2.3642"
"R2 -  0.4631"
```

It is the best result of all the models that I created.

The H2O framework is good and easy to use. It can give a good result with a minimum time to implementation. It also has got a lot of hyperparameters to tune.

Let's try another one — KERAS.

Keras is a high-level neural networks API developed with a focus on enabling fast experimentation. *Being able to go from idea to result with the least possible delay is key to doing good research.* Keras has the following key features:

- Allows the same code to run on CPU or on GPU, seamlessly.
- User-friendly API which makes it easy to quickly prototype deep learning models.
- Built-in support for convolutional networks (for computer vision), recurrent networks (for sequence processing), and any combination of both.
- Supports arbitrary network architectures: multi-input or multi-output models, layer sharing, model sharing, etc. This means that Keras is appropriate for building essentially any deep learning model, from a memory network to a neural Turing machine.
- Is capable of running on top of multiple back-ends including TensorFlow, CNTK, or Theano.

I would like to build my neural network with multi-input and multi-output.

For input, I would like to use the same features that I use for H2O models. For categorical features, I would use the Entity Embeddings approach. More detail about this approach you can read in this paper — link.

Multi-output helps me to solve 2 tasks (classification and regression) at the same time.

The Keras framework has not own features importance module, but you can use Lime.

Let's look at the results:

- Classification:

```
"Accuracy -  0.9252"
"Precision -  0.9616"
"True negative rate -  0.5541"
"Recall -  0.9565"
"F1_Score -  0.959"
"AUC -  0.7578"
```

- Regression:

```
"Accuracy -  0.8794"
"MAE -  0.3303"
"MSE -  2.9412"
"R2 -  0.332"
```

Keras' neural network brings cool results. This model performs mostly better than the previous ones.

Conclusions

Weather forecasting is a really difficult task. The machine learning algorithms can help in prediction for a short term period. To get good results we need to use lag features or use RNN architecture in the neural networks.

All code you can find in the Git repository — link.

Machine Learning AI Data Science Neural Networks

About Help Legal