Report on learning practice # 3

Analysis of multivariate random variables

Performed by:

Alexander Yamoldin

J4134c

Saint-Petersburg

2021

Step 1. Choose variables for sampling from your dataset.

We choose weather dataset with the features below:

latitude - degrees

longitude - degrees

rain - bool (1 - is rain, 0 - is not rain)

temp - Air Temperature , °C

wetb - Wet Bulb Air Temperature, °C

dewpt - Dew Point Air Temperature, °C

vappr - Vapour Pressure, hpa

rhum - Relative Humidity, %

mls - Mean Sea Level Pressure, hPa

wdsp - Mean Hourly Wind Speed, kt

First 5 rows of dataset looks like:

	latitude	longitude	rain	temp	wetb	dewpt	vappr	rhum	msl	wdsp
0	51.476	-9.428	0.0	11.5	10.6	9.7	12.0	88.0	1031.1	6.0
1	54.228	-10.007	0.8	13.0	12.9	12.8	14.8	99.0	1013.0	4.0
2	55.372	-7.339	0.0	6.3	3.5	-1.2	5.6	58.0	1025.2	19.0
3	52.690	-8.918	0.0	8.5	6.2	3.1	7.6	69.0	1025.3	10.0
4	53.516	-6.660	0.0	13.6	11.3	9.1	11.6	74.0	1013.2	13.0

5 rows × 10 columns Open in new tab

We choose temp, vappr, dewpt, msl as target variables. The rest are predictors.

Step 2. Using univariate parametric distributions make sampling of chosen target variables.

We use two methods of sampling. There are Inverse transform sampling and Accept-Reject sampling.

The main idea of Inverse Transform sampling is generating random numbers from any probability distribution by using its inverse cumulative distribution $F^{-1}(x)$.

In our case we evaluate the inverse cumulative density function by the given data.

On the first step we find a Probability Density Function(PDF) of the given data. In the second step we find a Cumulative Density Function (CDF) by the definition of CDF : $CDF(x) = P(X \leq x)$. In the second step we just create inverse CDF samples using CDF parameters and universe distribution. Because the values of uniform distribution [0,1) is exactly like probability in CDF. So by this trick we have an Inverse CDF. The examples of this in the pictures below (pictures in the better size you can see in github: https://github.com/AAYamoldin/Methods-and-models-for-multivariate-data-analysis-2021-2022-ITMO_labs/blob/main/Lab3/Pictures/INTF_dewpt_1.png

https://github.com/AAYamoldin/Methods-and-models-for-multivariate-data-analysis-2021-2022-ITMO_labs/blob/main/Lab3/Pictures/INTF_msl_1.png

https://github.com/AAYamoldin/Methods-and-models-for-multivariate-data-analysis-2021-2022-ITMO_labs/blob/main/Lab3/Pictures/INTF_temp_1.png

https://github.com/AAYamoldin/Methods-and-models-for-multivariate-data-analysis-2021-2022-ITMO_labs/blob/main/Lab3/Pictures/INTF_vappr_1.png):

Theoretical part:

According the first lab we expected that we work with normal distribution function:

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma}}$$

CDF of this function (with PDF $f(x, \mu, \sigma)$) is:

$$F(x) = \Phi\left(\frac{x-\mu}{\sigma}\right) = \frac{1}{2}\left[1 + erf\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right], \text{ where}$$

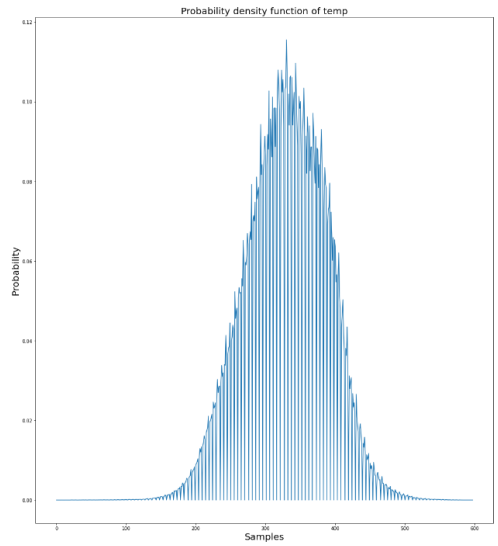
$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ - this error function (also called the Gauss error function), is a complex function of a complex variable.

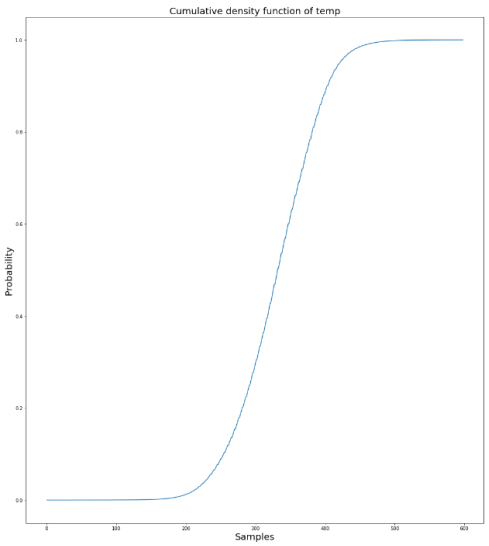
The inverse CDF is

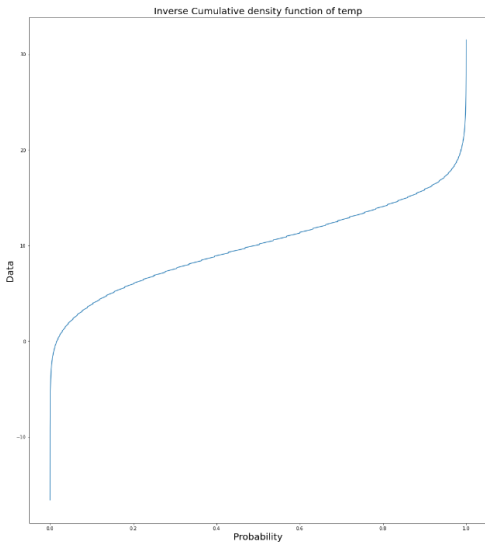
$F^{-1}(p),\ p \in [0.1], is\ the\ unique\ real\ number\ x\ such\ that\ F(x) = p.$

The problem is that F^{-1} **does not exist** in a simple closed formula. So we compute it numerically.

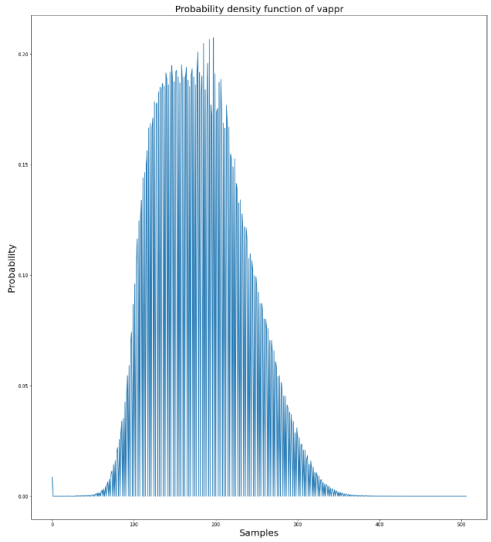
For temperature target (temp):

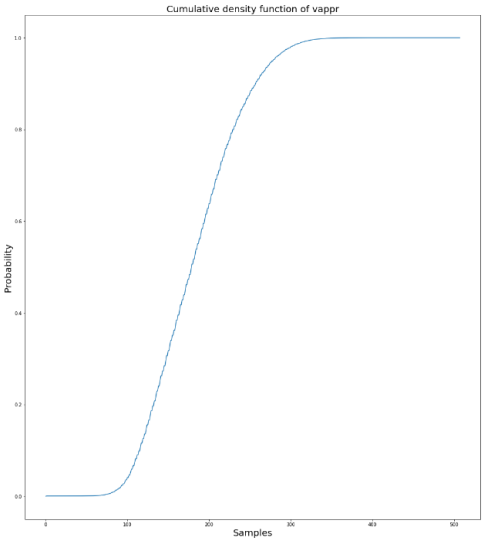


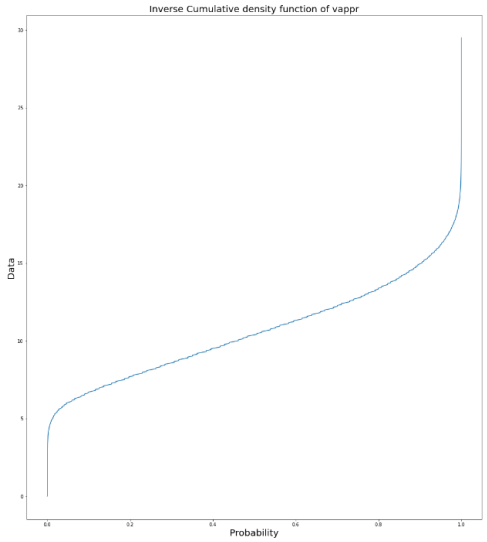




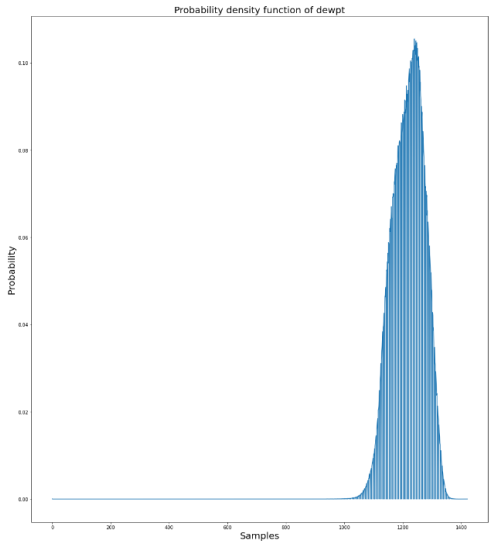
For Vapour Pressure target (vappr):

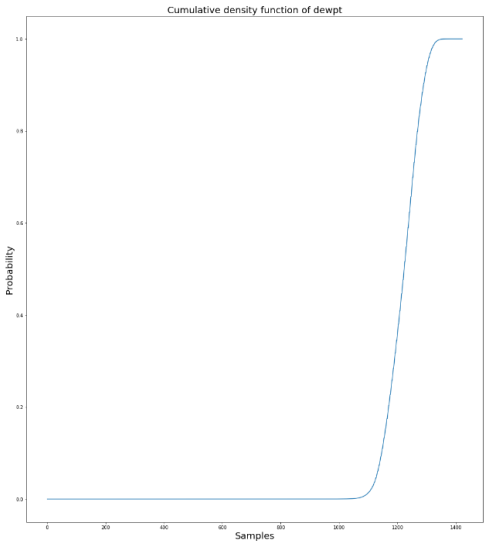


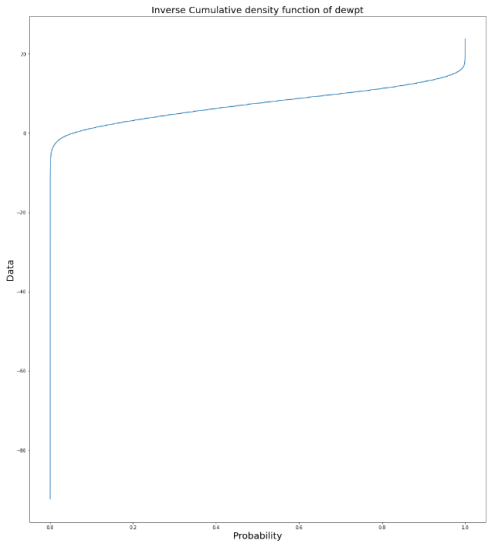




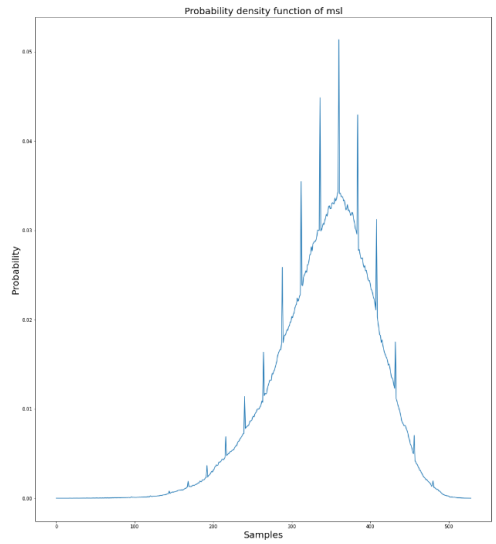
For Dew Point Air Temperature target (dewpt):

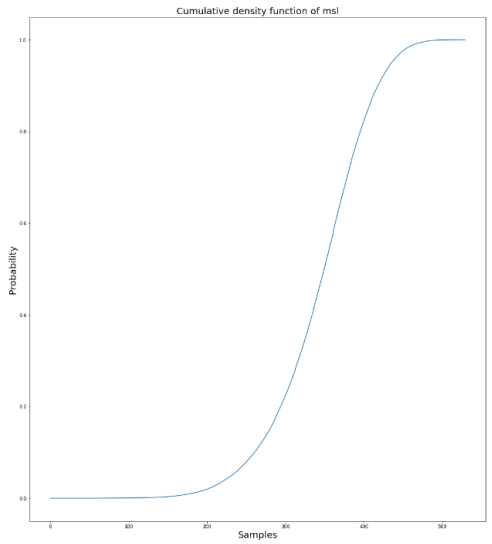


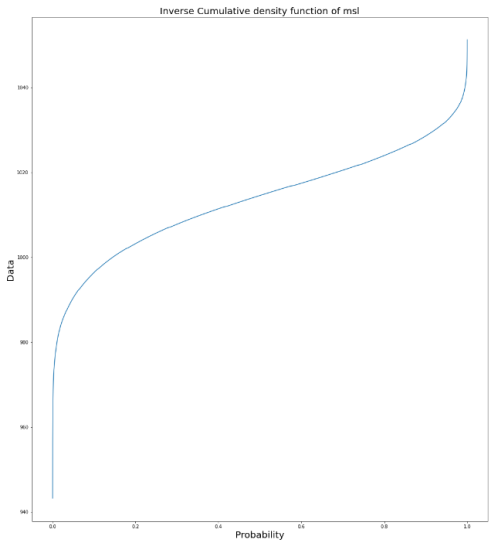




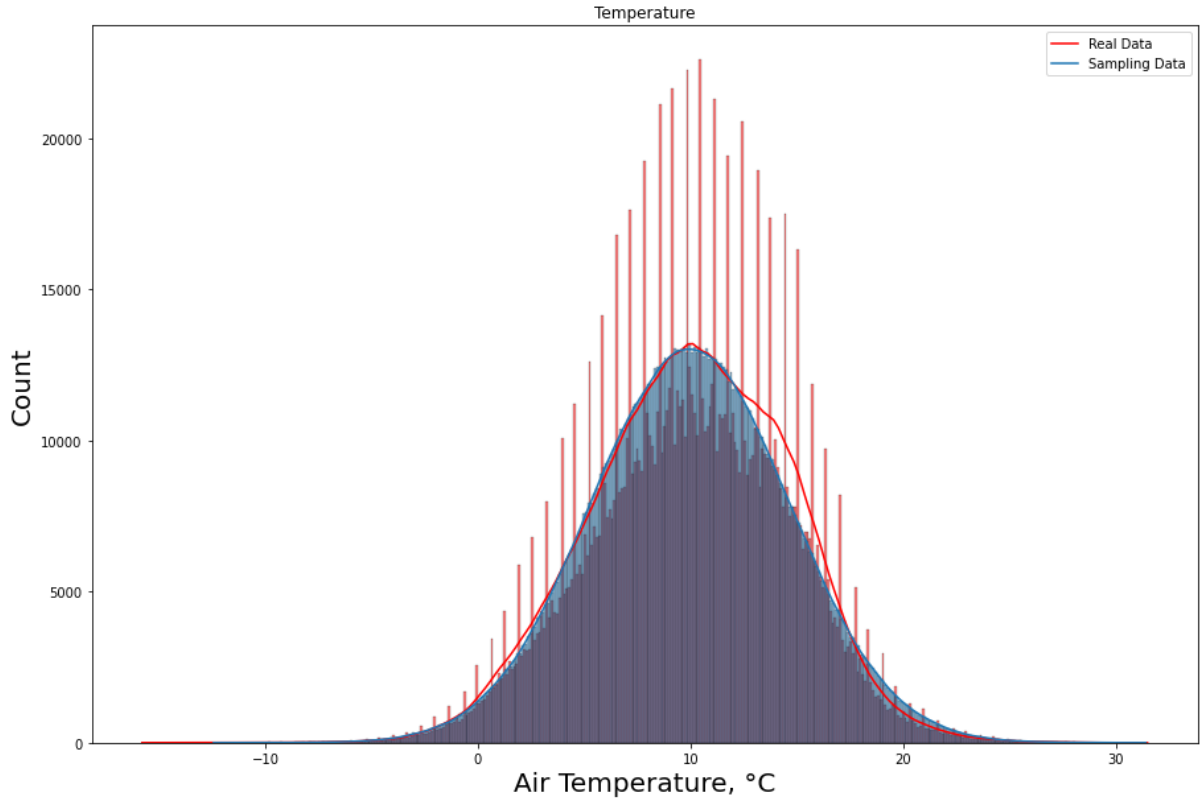
For Mean Sea Level Pressure target (msl):

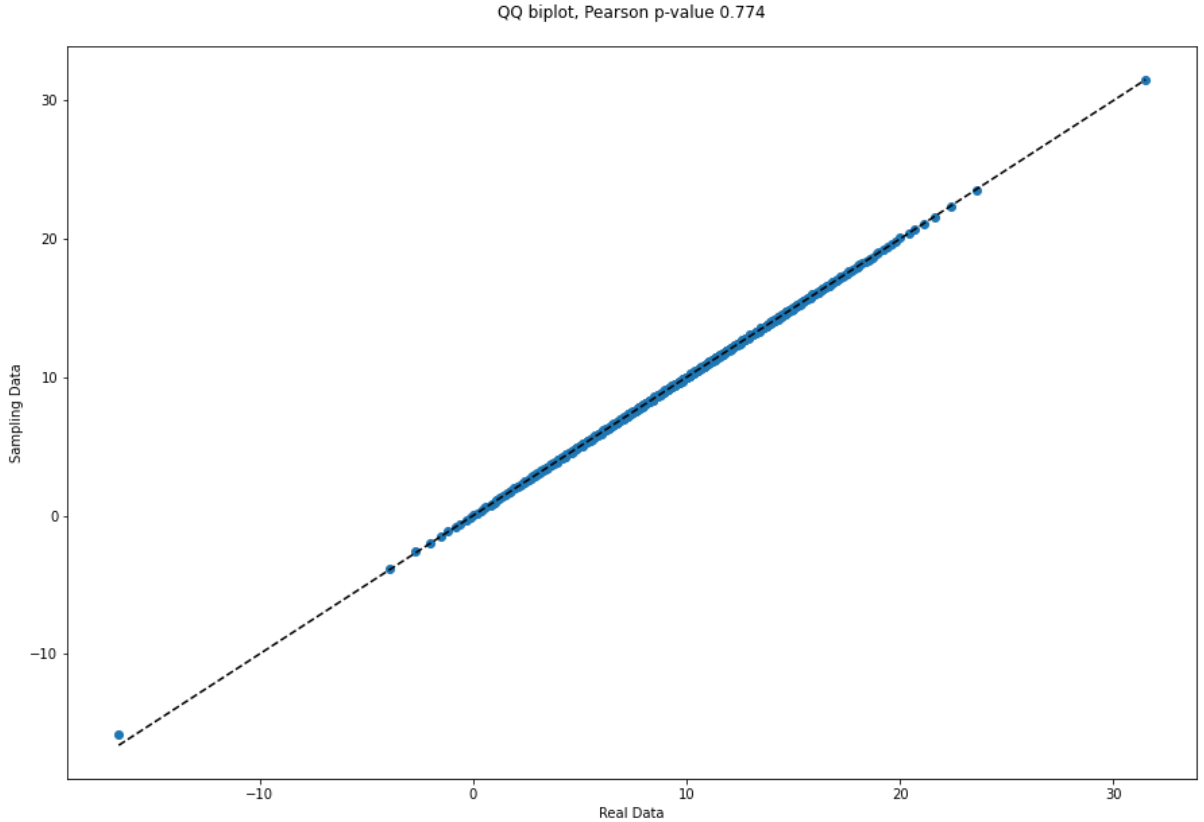




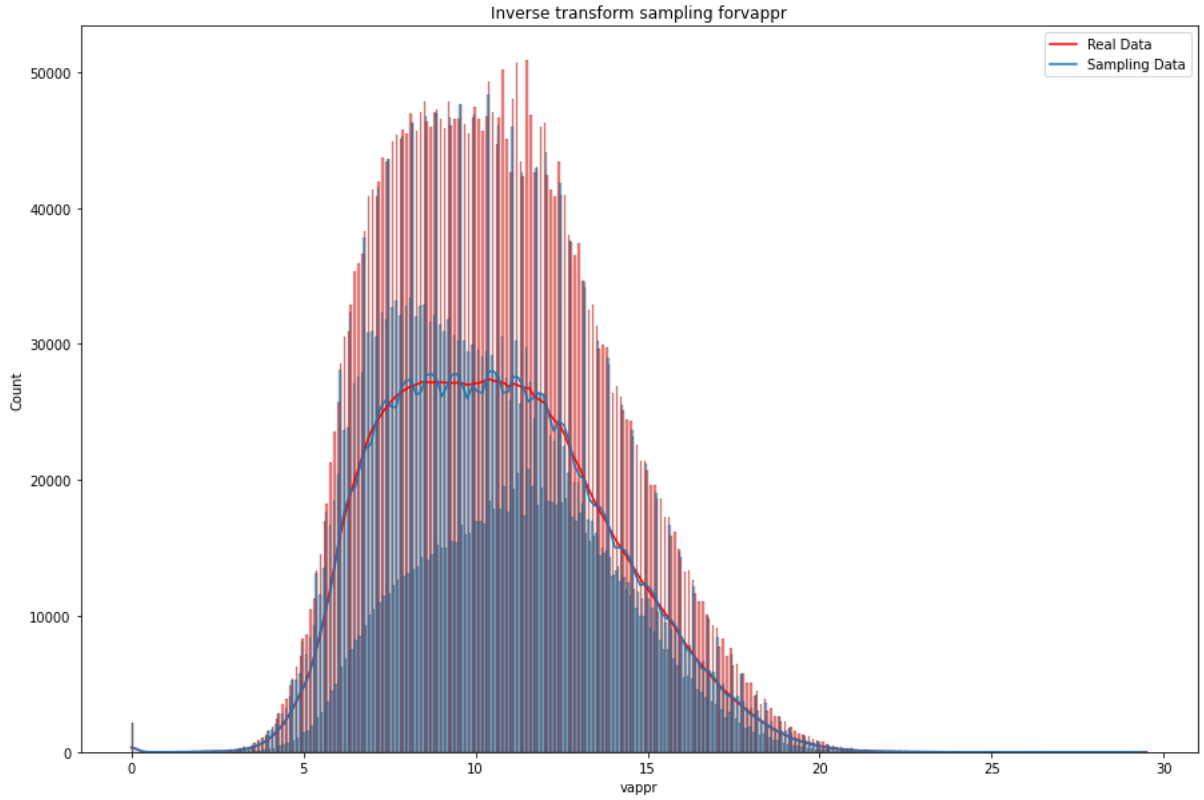


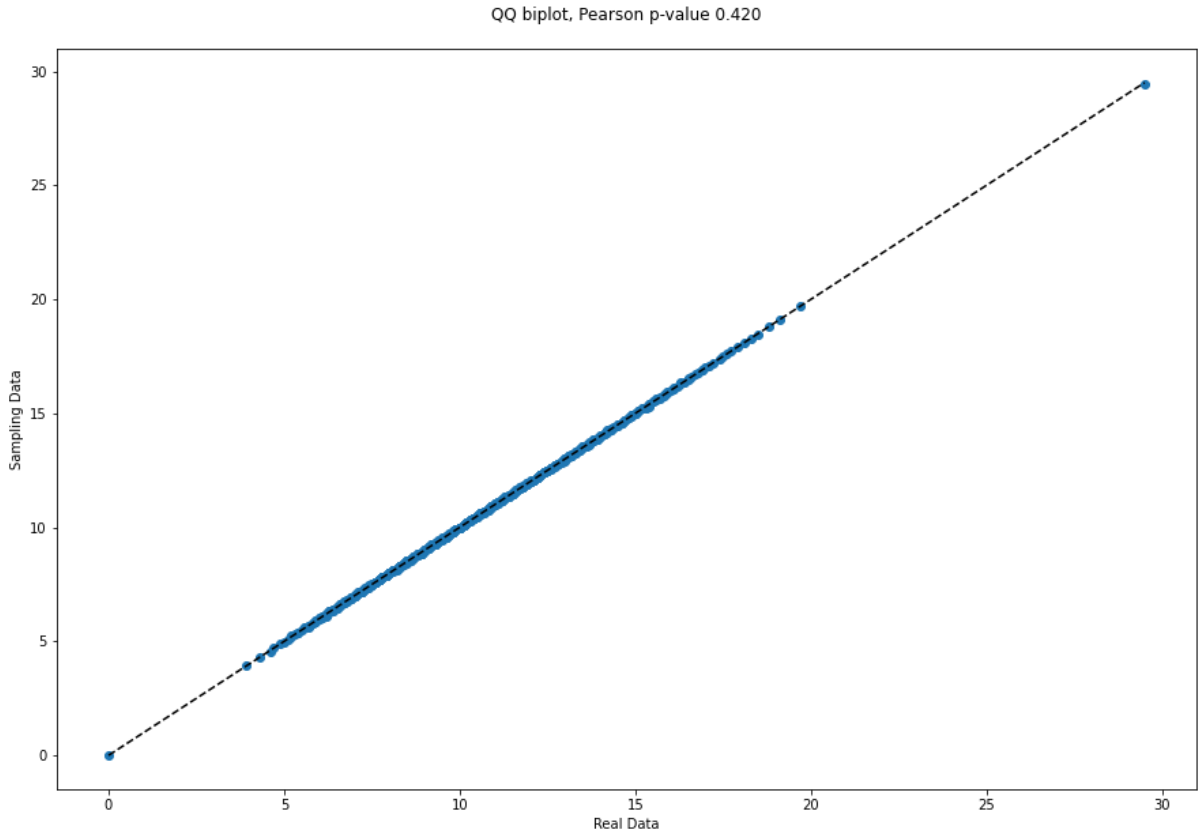
For temperature target (temp):



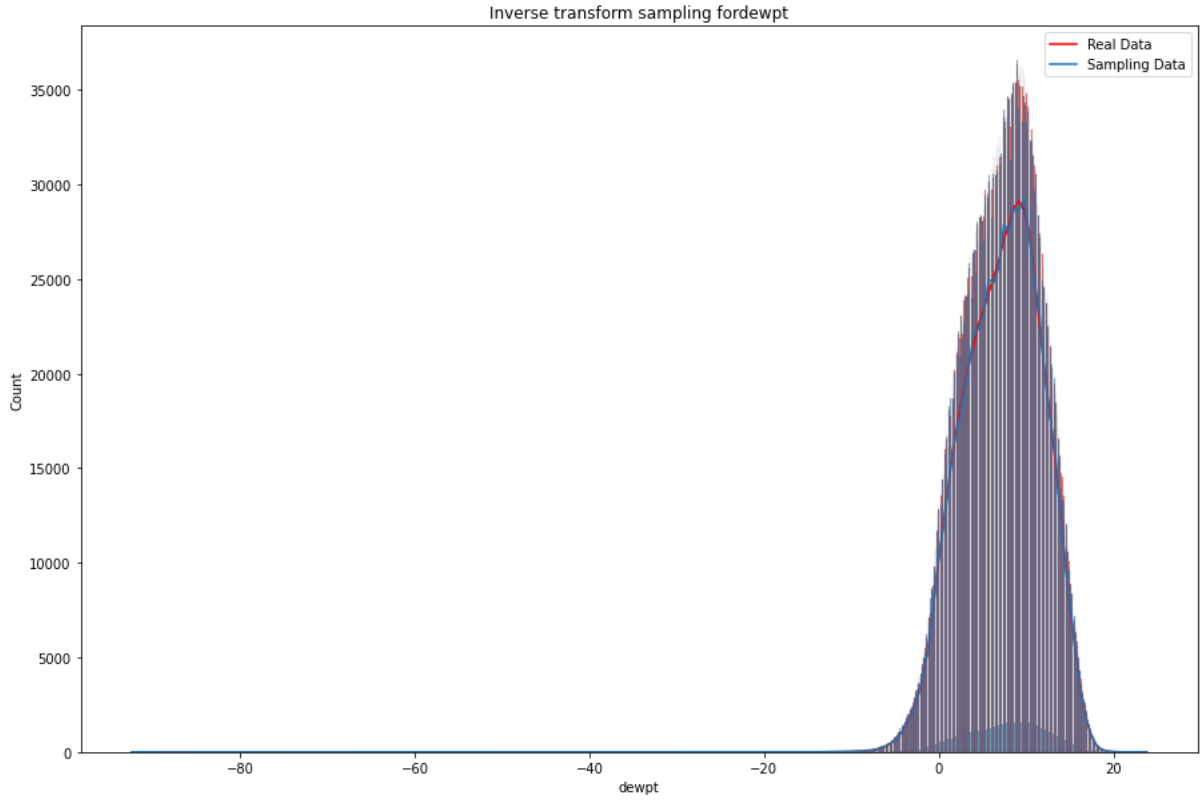


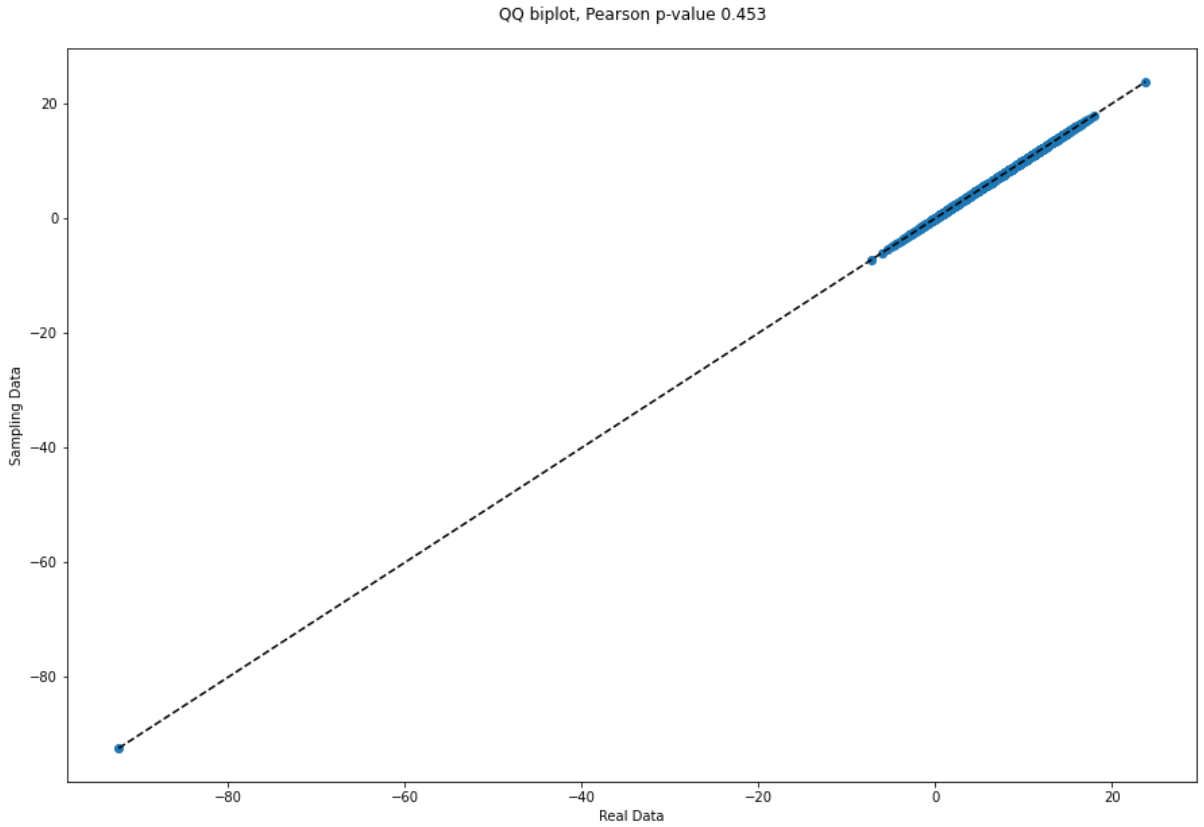
For Vapour Pressure target (vappr):



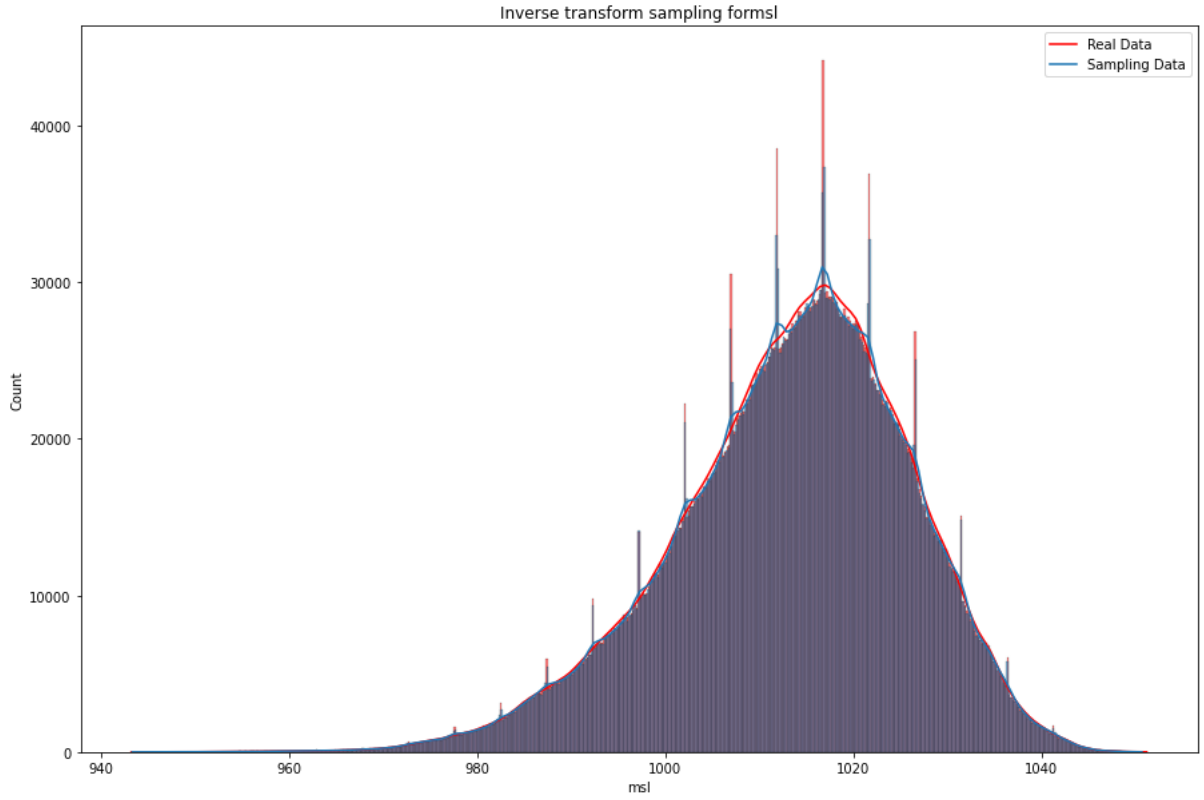


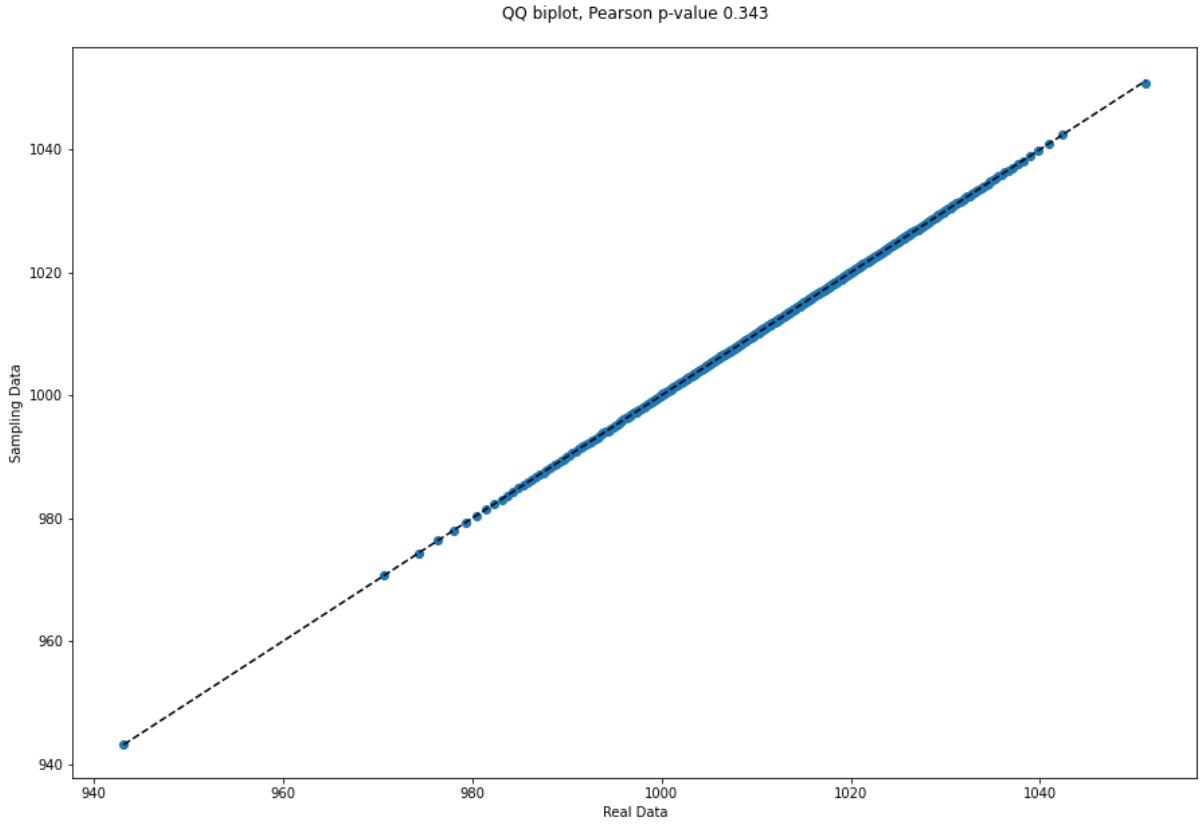
For Dew Point Air Temperature target (dewpt):





For Mean Sea Level Pressure target (msl):





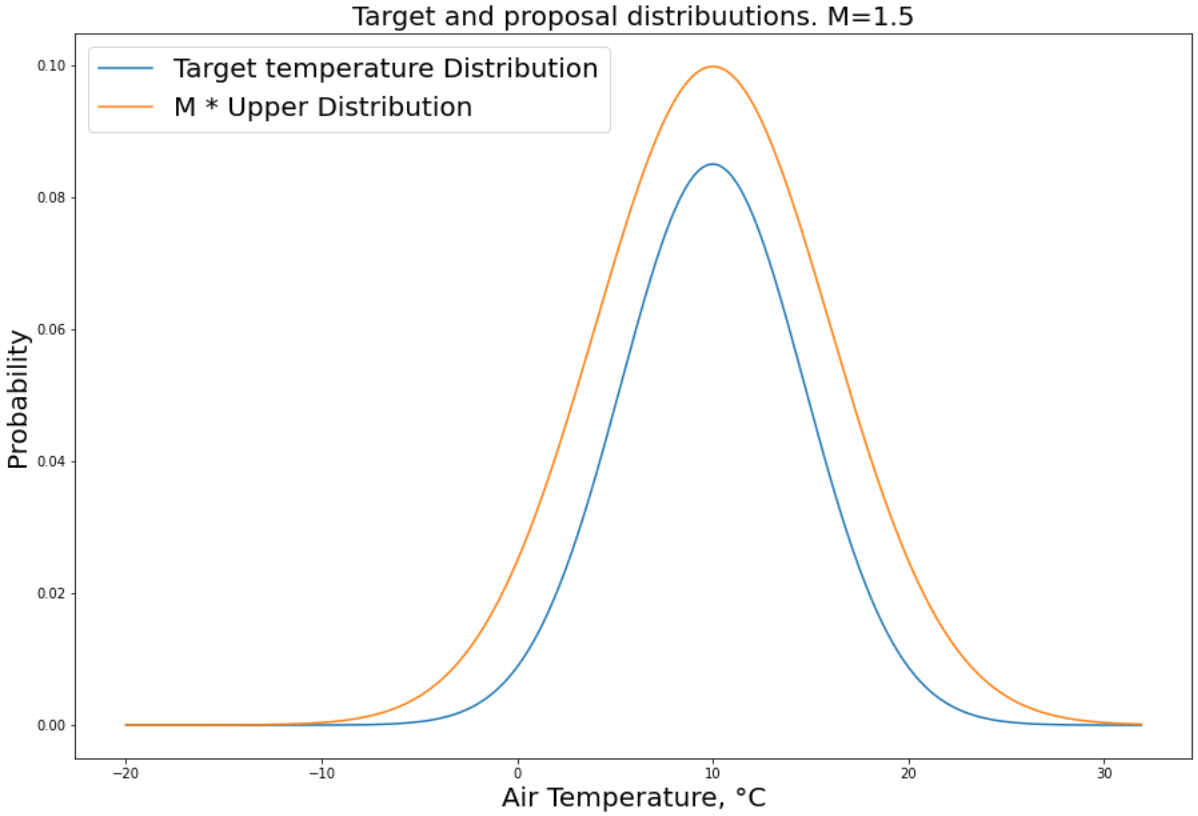
Our next sampling algorithm is Accept-Reject. The main idea of the algorithm is that we can generate a sample value from a target distribution X with PDF f(x) by using a proposal distribution Y with probability function g(x) where

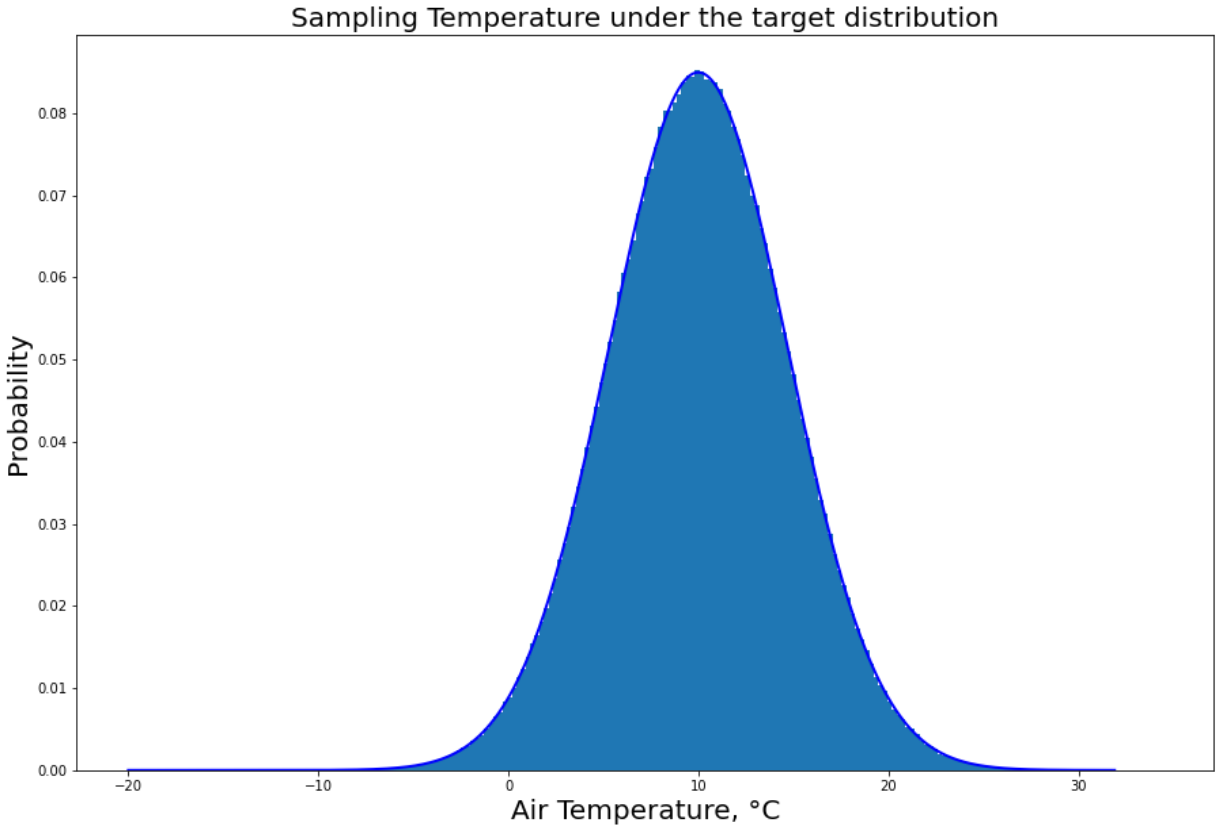
M * g(x) > f(x) in all values of x and accepting the sample from Y with probability f(x) / (M * g(x)) where 1< M < inf .

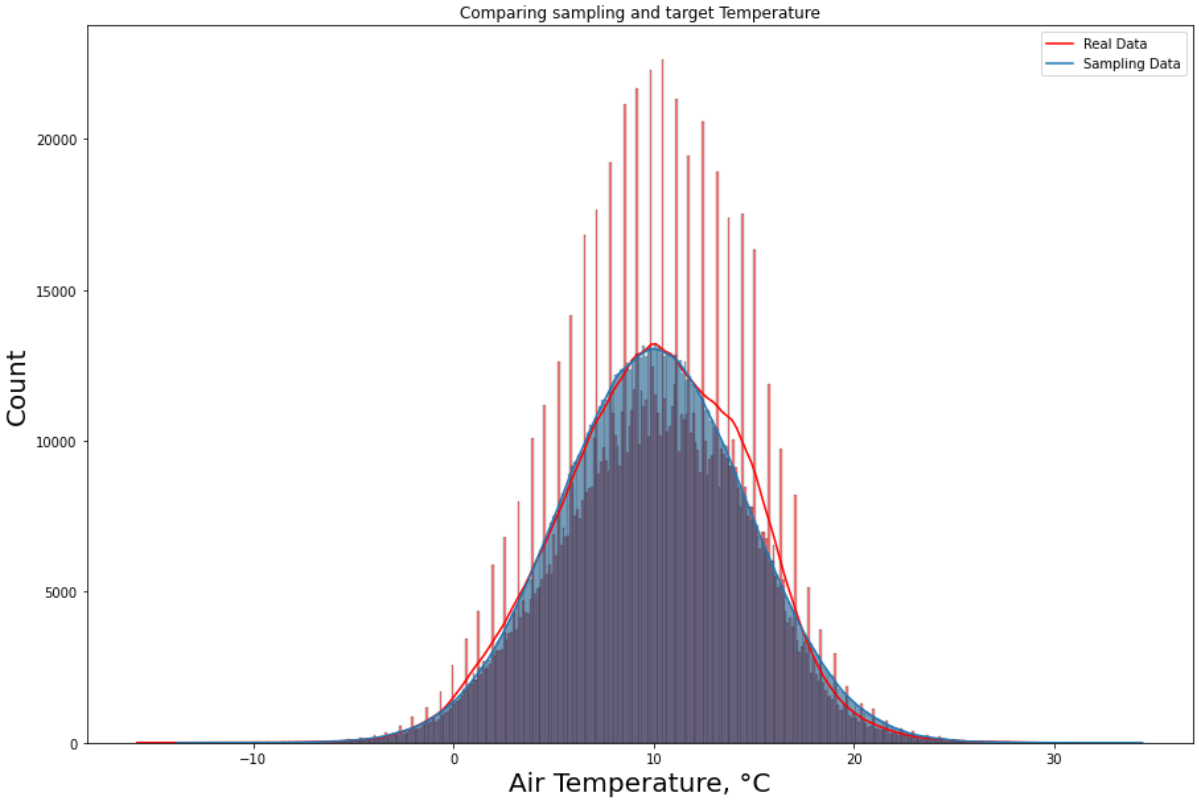
In the our case we use g(x) from normal distribution:

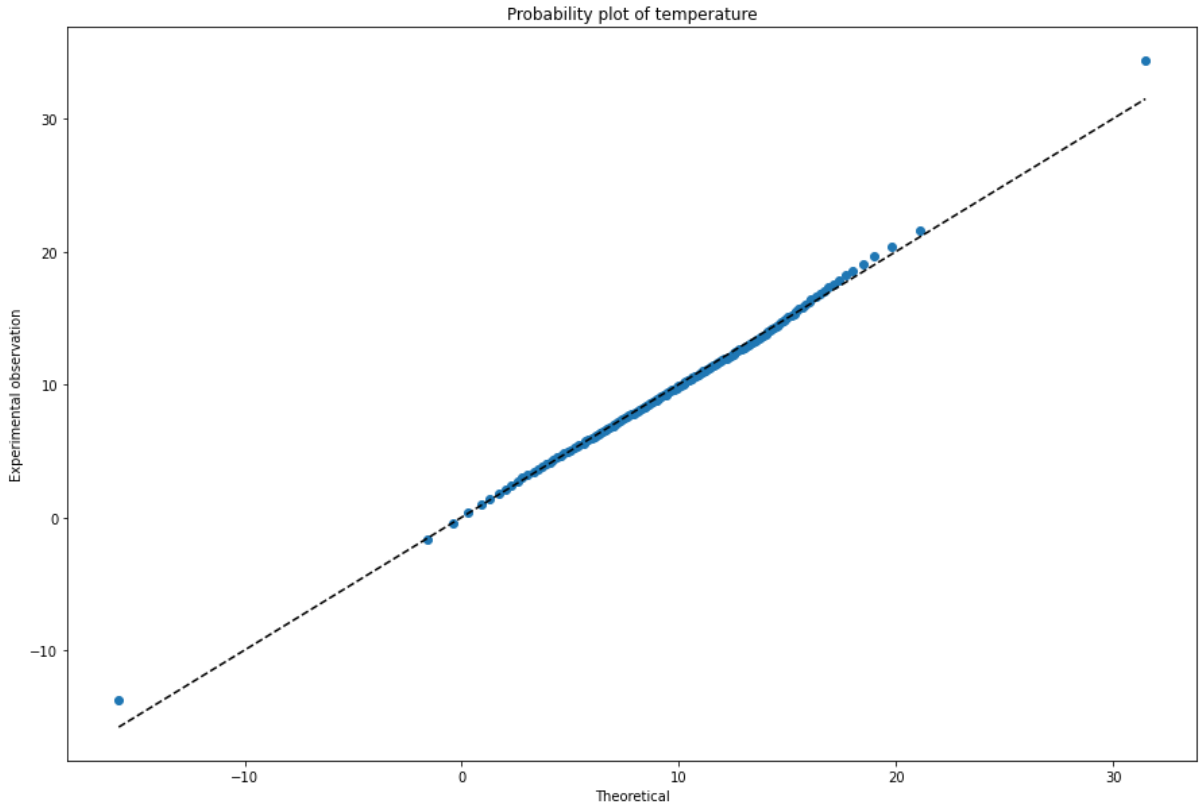
$$g(x,\ \mu,\ \sigma,\ M)\ =\ \frac{M}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma}}$$

For temperature target (temp): $\mu \approx 9.992,\ \sigma \approx 4.696,\ M = 1.5$

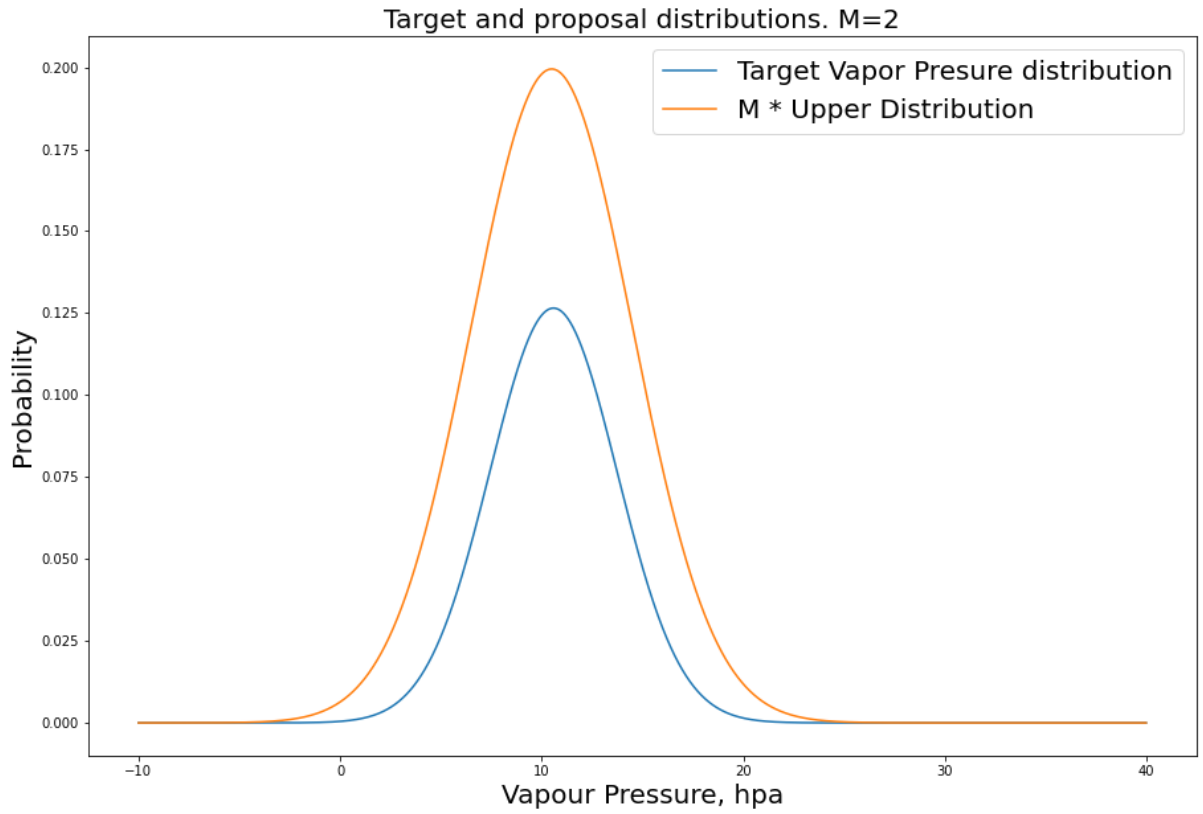


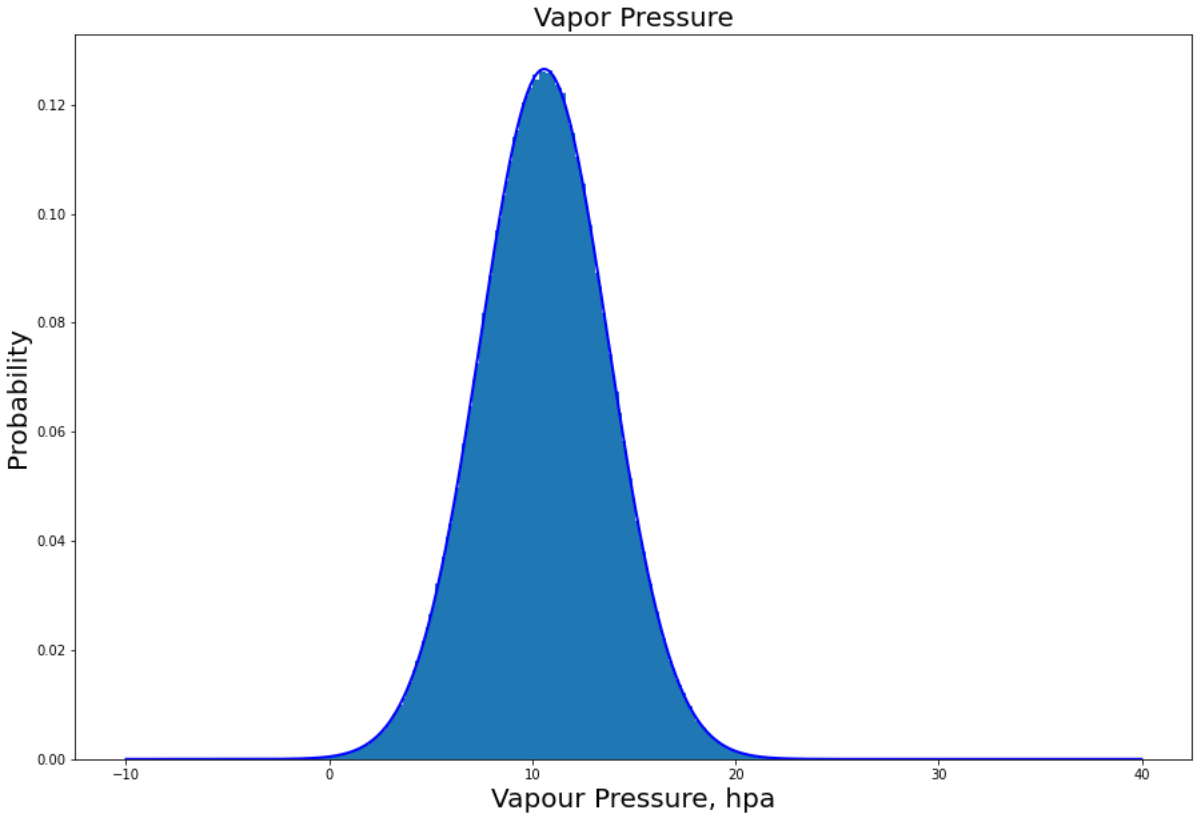

Sampling Temperature under the target distribution
Probability
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
−20
−10
0
10
20
30
Air Temperature, °C

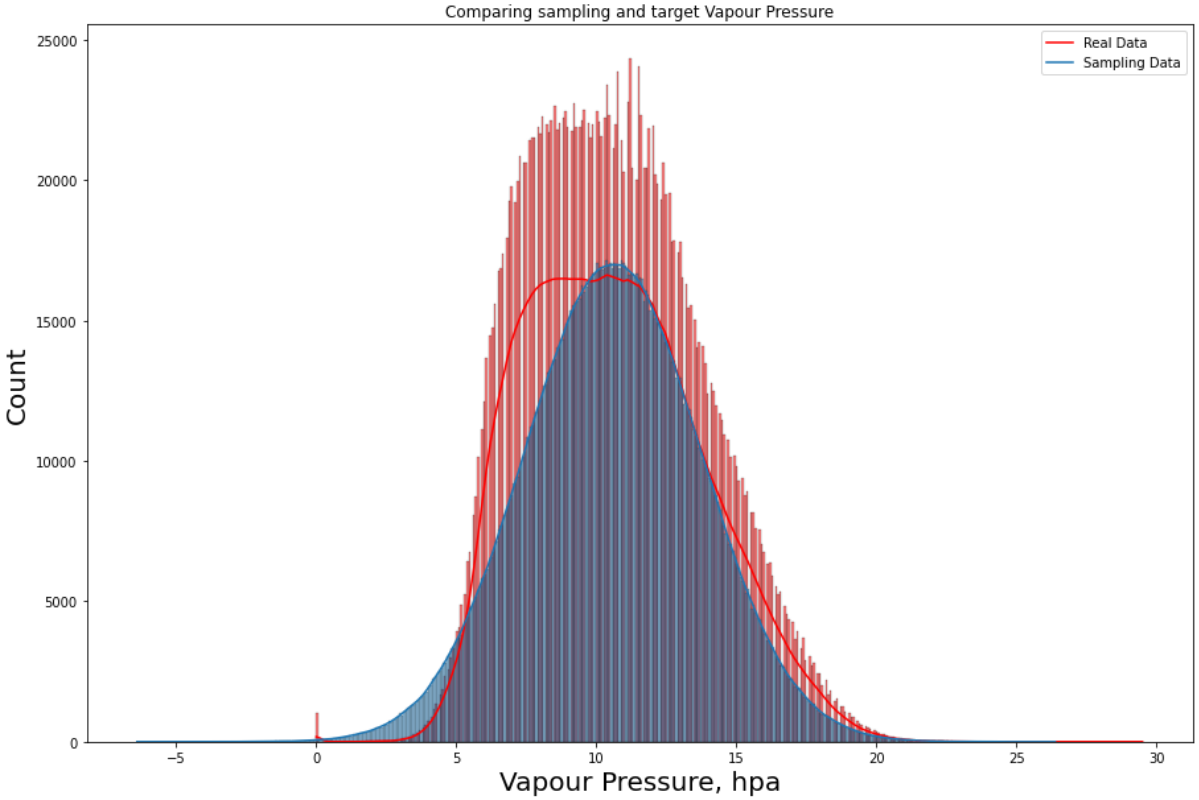

Comparing sampling and target Temperature
Real Data
Sampling Data
Count
0
5000
10000
15000
20000
−10
0
10
20
30
Air Temperature, °C

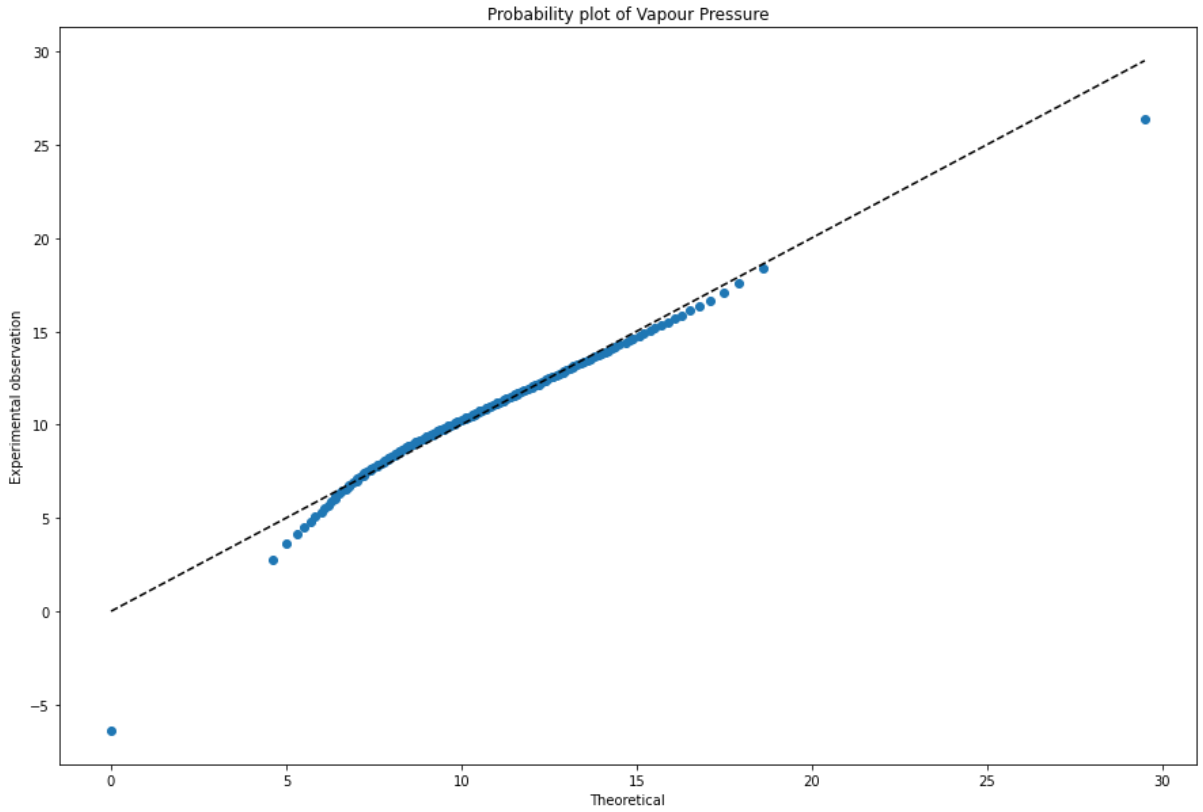


For Vapour Pressure target (vappr): $\mu \approx 10.587,\ \sigma \approx 3.153,\ M = 2$

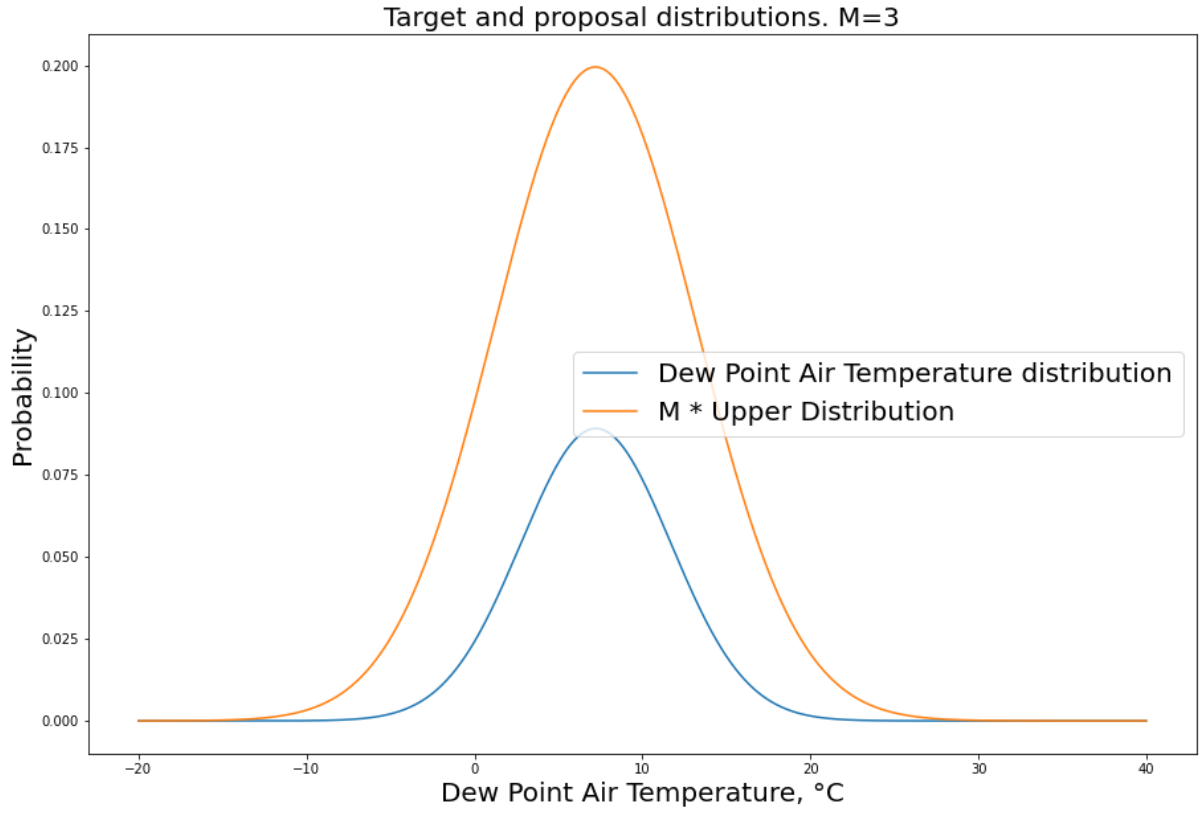


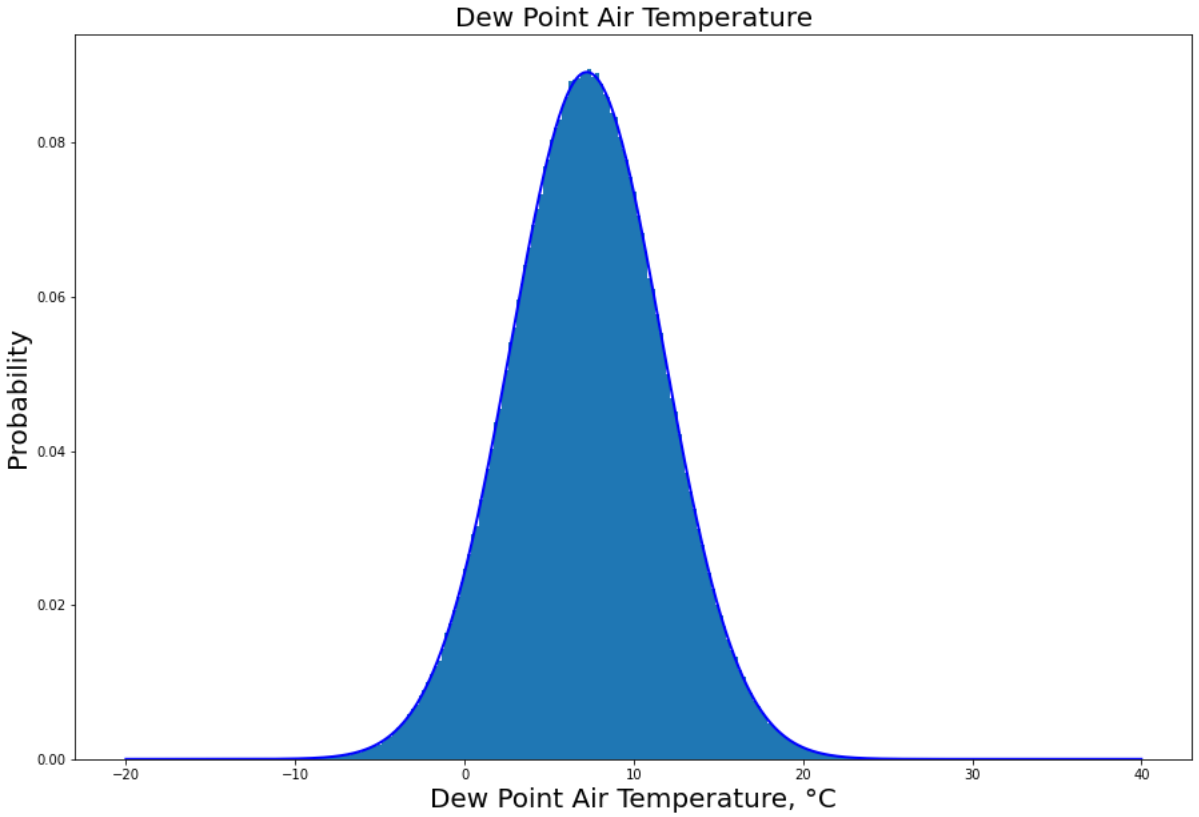

Vapor Pressure
Probability
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−10
0
10
20
30
40
Vapour Pressure, hpa

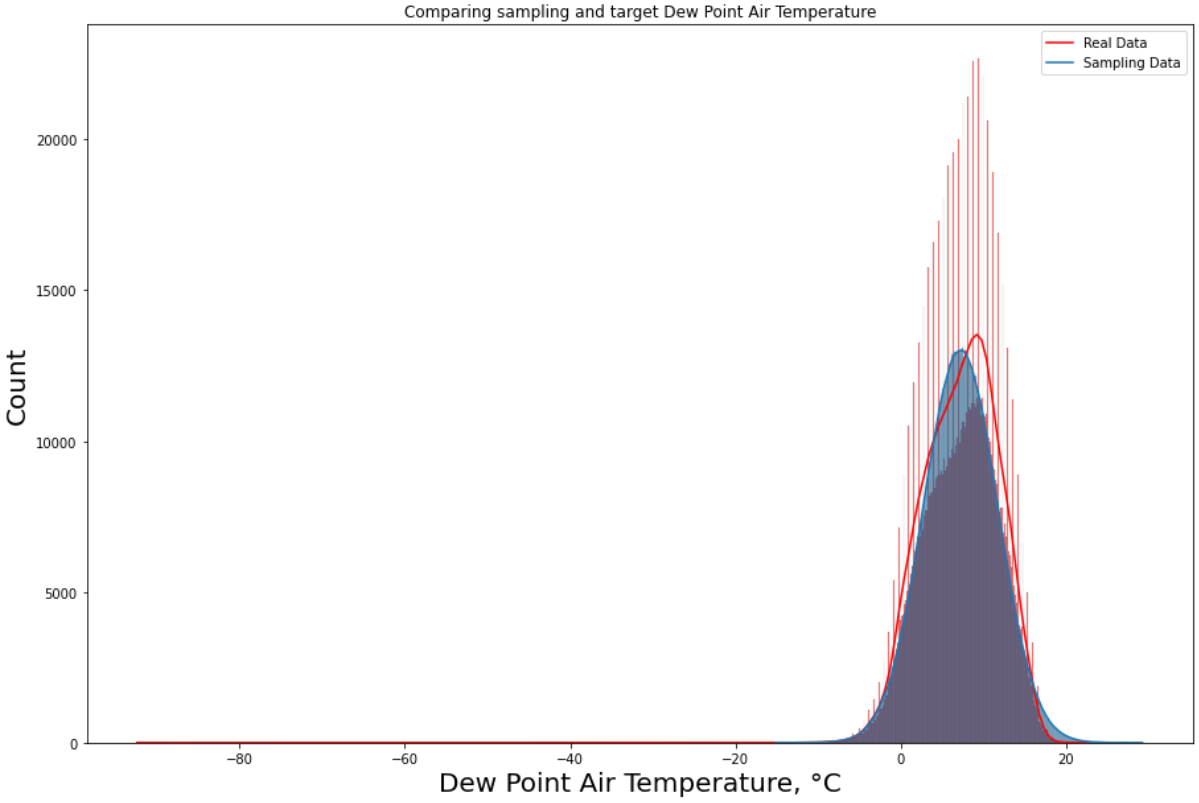

Comparing sampling and target Vapour Pressure
Real Data
Sampling Data
Count
25000
20000
15000
10000
5000
0
−5
0
5
10
15
20
25
30
Vapour Pressure, hpa

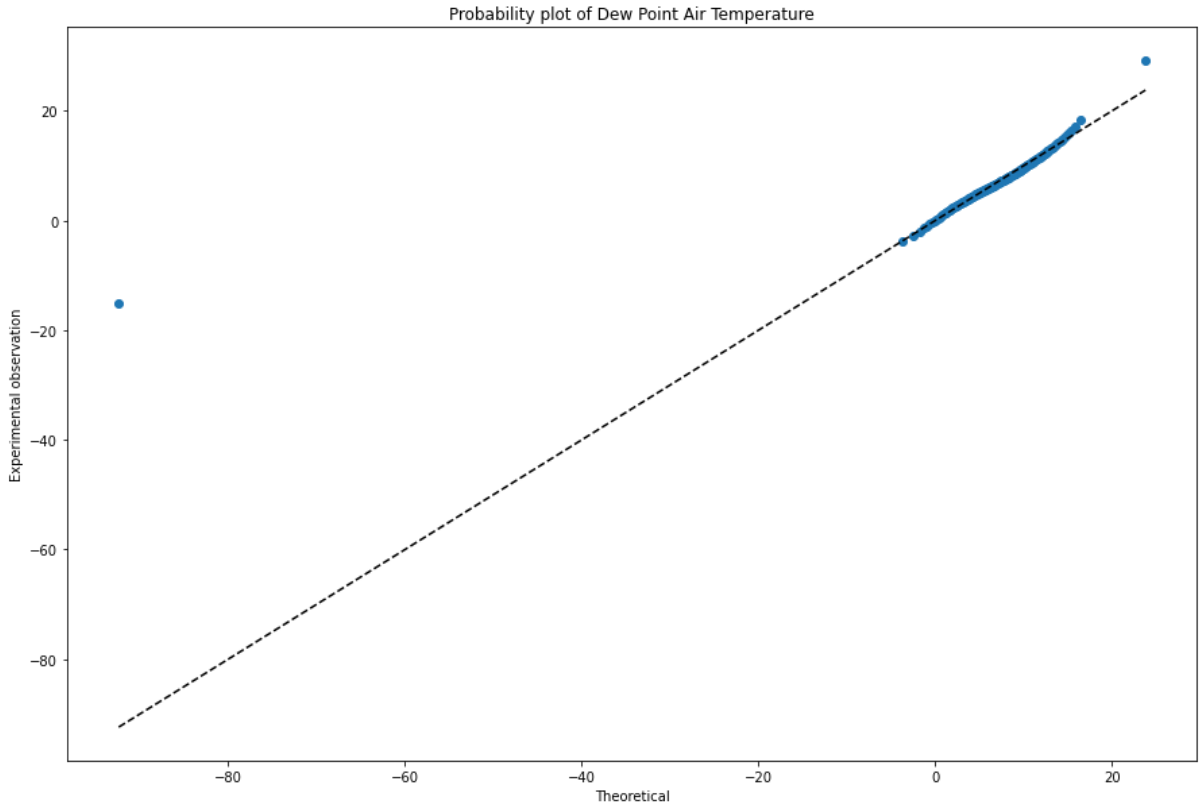


For Dew Point Air Temperature target (dewpt): $\mu \approx 7.230,\ \sigma \approx 4.473,\ M = 3$

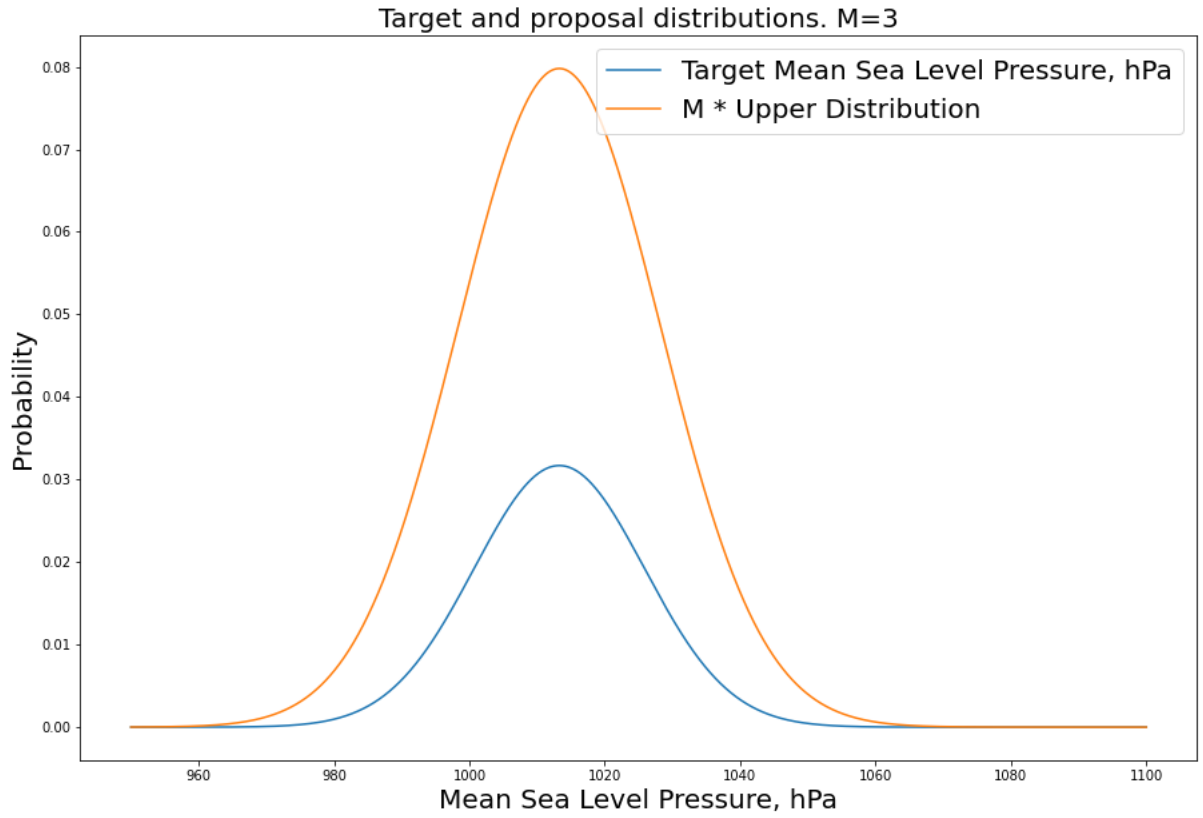


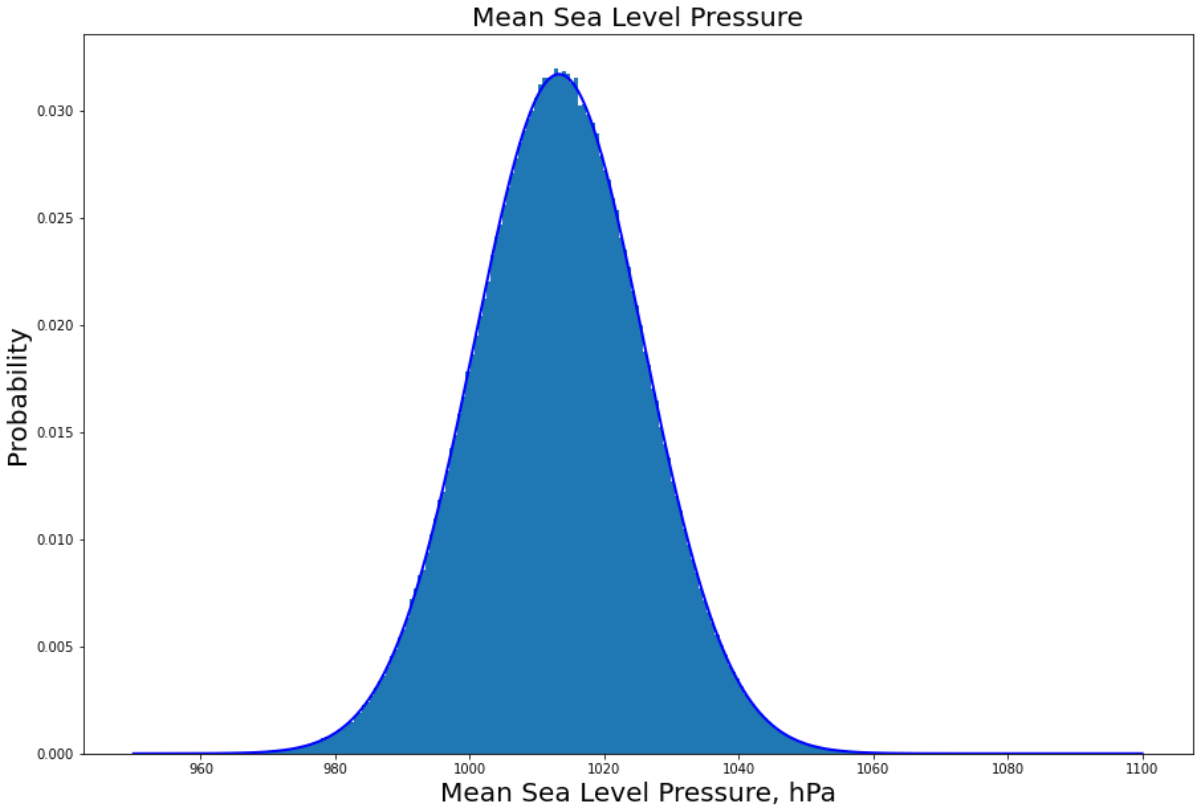

Dew Point Air Temperature
Probability
0.00
0.02
0.04
0.06
0.08
−20
−10
0
10
20
30
40
Dew Point Air Temperature, °C

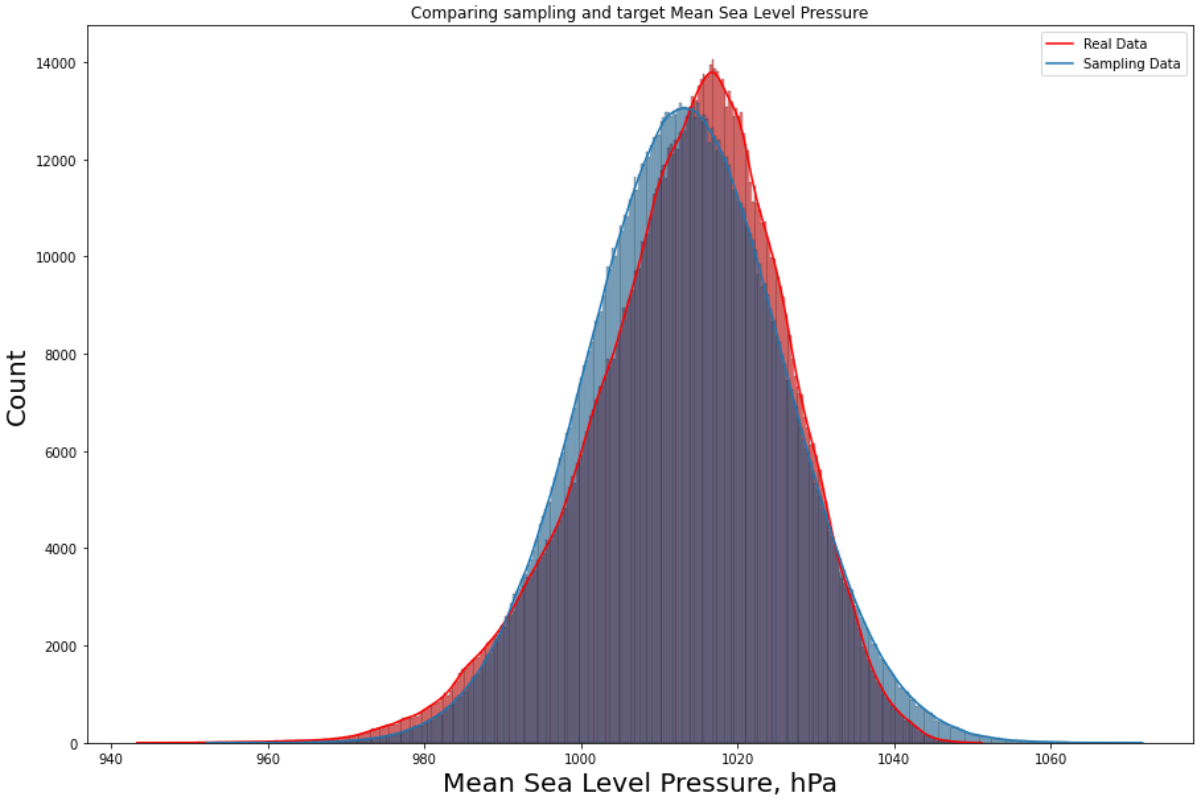

Comparing sampling and target Dew Point Air Temperature
Real Data
Sampling Data
Count
0
5000
10000
15000
20000
−80
−60
−40
−20
0
20
Dew Point Air Temperature, °C

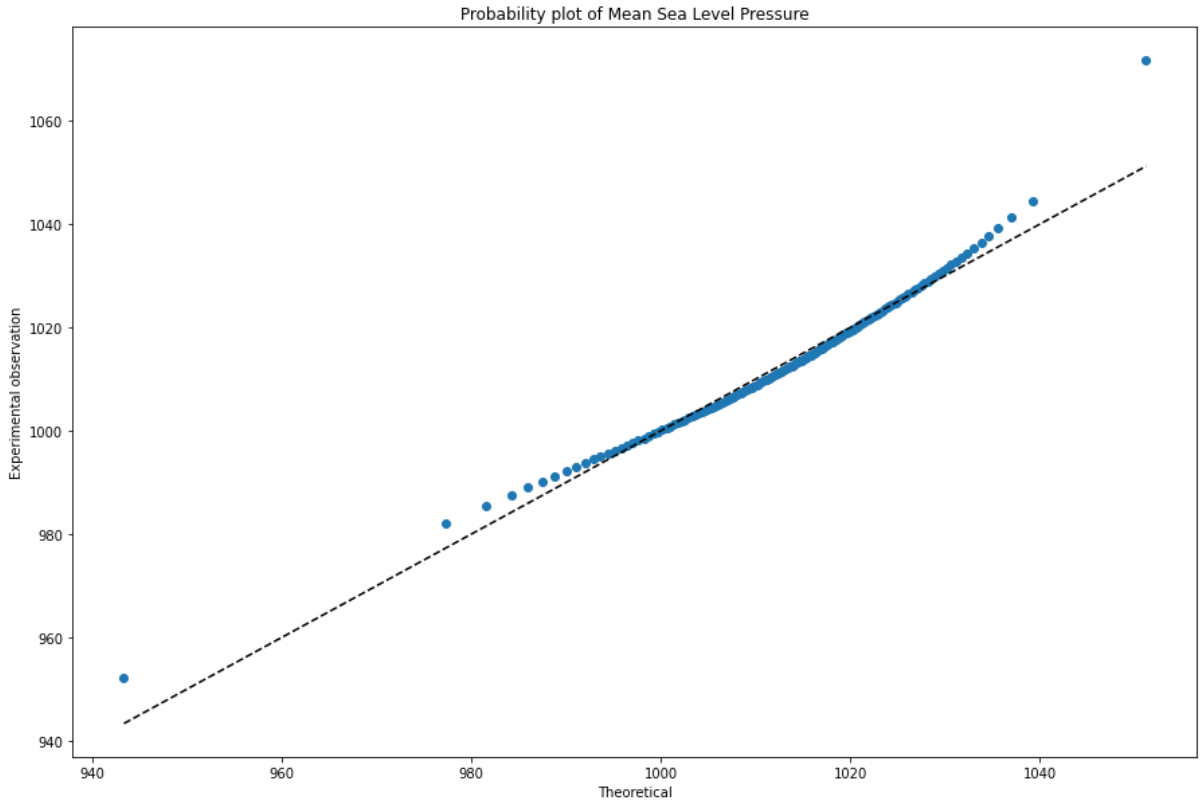


For Mean Sea Level Pressure target (msl): $\mu \approx 1013.281,\ \sigma \approx 12.588,\ M = 3$




Mean Sea Level Pressure
Probability
0.030
0.025
0.020
0.015
0.010
0.005
0.000
960
980
1000
1020
1040
1060
1080
1100
Mean Sea Level Pressure, hPa


Comparing sampling and target Mean Sea Level Pressure
Real Data
Sampling Data
Count
14000
12000
10000
8000
6000
4000
2000
0
940
960
980
1000
1020
1040
1060
Mean Sea Level Pressure, hPa



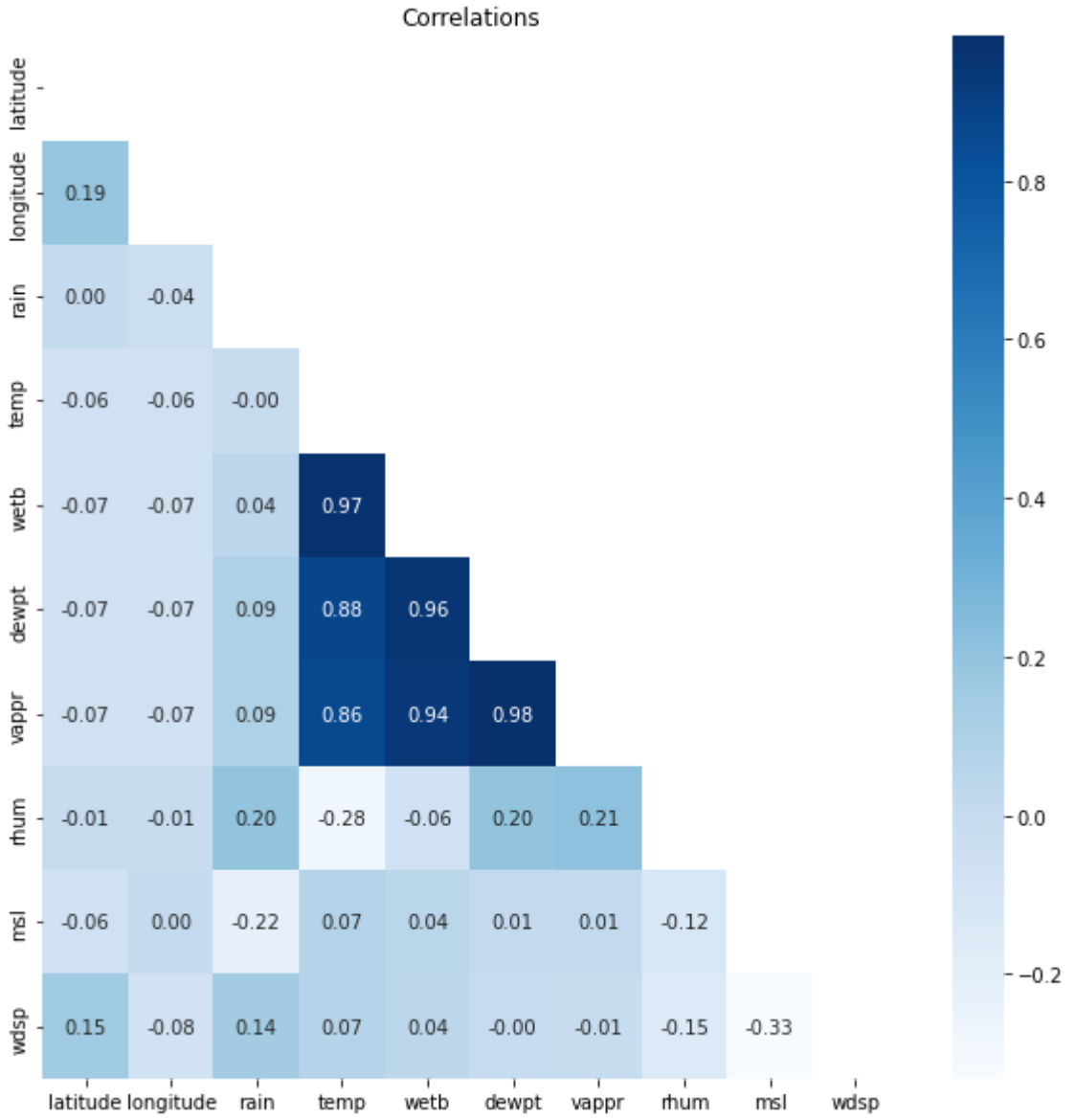
Results of the methods:

Comparing p-value of the different methods:

Target variable	Inverse transform sampling	Accept-Reject sampling
Air Temperature	0.77	0.62
Vapour Pressure	0.42	0.45
Dew Point Air Temperature	0.45	0.83
Mean Sea Level Pressure	0.34	0.57

We see that Inverse transform sampling works better for our task. It may be explain that our data distribute not ideal by the normal distribution, so Inverse transform sampling which create the own inverse CDF function of giving data.

Step 3. Estimate correlation coefficients between predictors and chosen target variables.



Step 4. Build a Bayesian network for a chosen set of variables. Choose its structure on the basis of multivariate analysis and train distributions in nodes using chosen algorithm

Define a Bayesian network

```
model = BayesianModel([
    ('msl', 'wdsp'),
    ('vappr', 'wetb'),
    ('dewpt', 'wetb'),
    ('temp', 'rhum'),
    ('temp', 'wetb'),
    ('msl', 'rain')
])
```

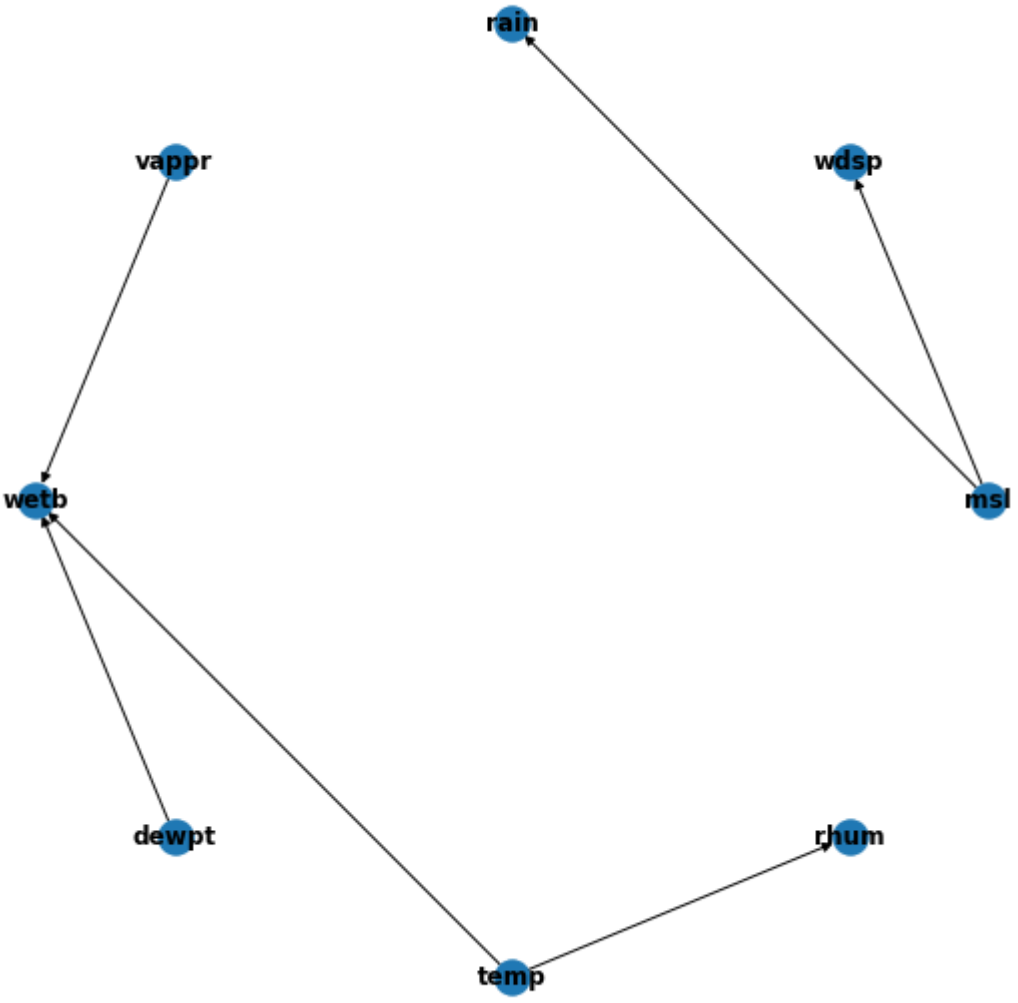
We must prepare data before starting working with BN. In this laboratory work we use `KBinsDiscretizer` from `sklearn.preprocessing`. We defined continuous variables and convert it to discrette:

```
#data preparation
df_trans = df.iloc[:, 2:].copy()

data_keys = ['rain', 'wetb', 'rhum', 'wdsp', 'temp', 'vappr', 'dewpt', 'msl']
data = df_trans[data_keys]
data = data.dropna()

discretizer = KBinsDiscretizer(n_bins=5, encode="ordinal", strategy="kmeans")

data_descrete = discretizer.fit_transform(data.values)
data_descrete = pd.DataFrame(data_descrete, columns=data_keys)

data_descrete.head()
```

	rain	wetb	rhum	wdsp	temp	vappr	dewpt	msl
0	0.0	3.0	3.0	1.0	3.0	2.0	3.0	2.0
1	0.0	1.0	2.0	2.0	1.0	0.0	1.0	0.0
2	0.0	3.0	4.0	1.0	3.0	3.0	4.0	2.0
3	0.0	3.0	3.0	1.0	2.0	2.0	3.0	1.0
4	0.0	3.0	4.0	1.0	2.0	2.0	3.0	2.0

5 rows × 8 columns Open in new tab

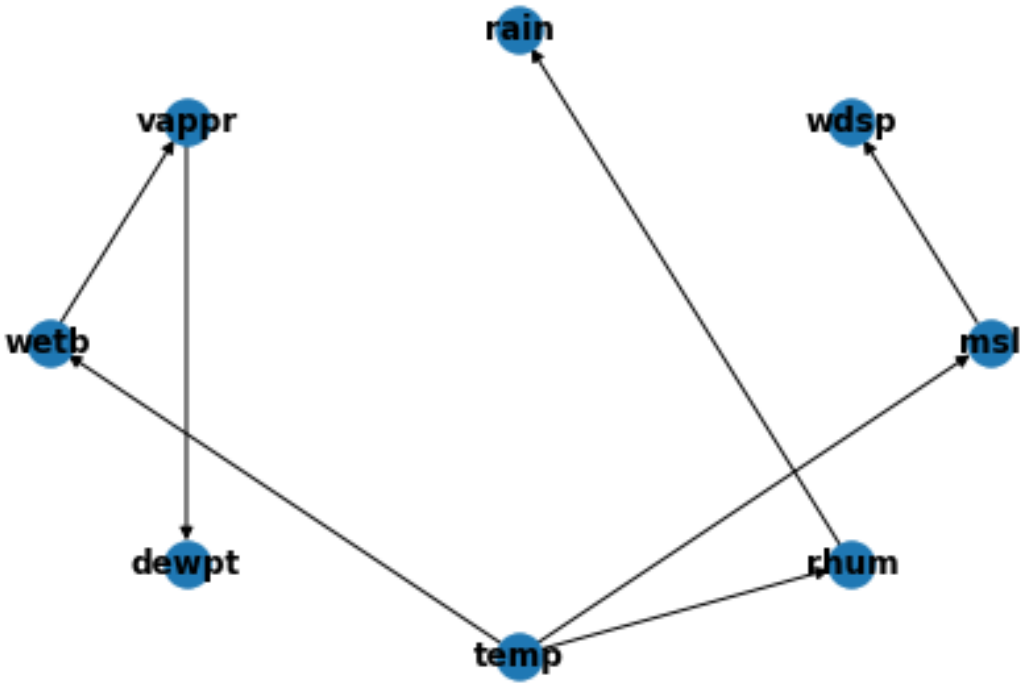
The handle choosing structure of a nodes in the picture below



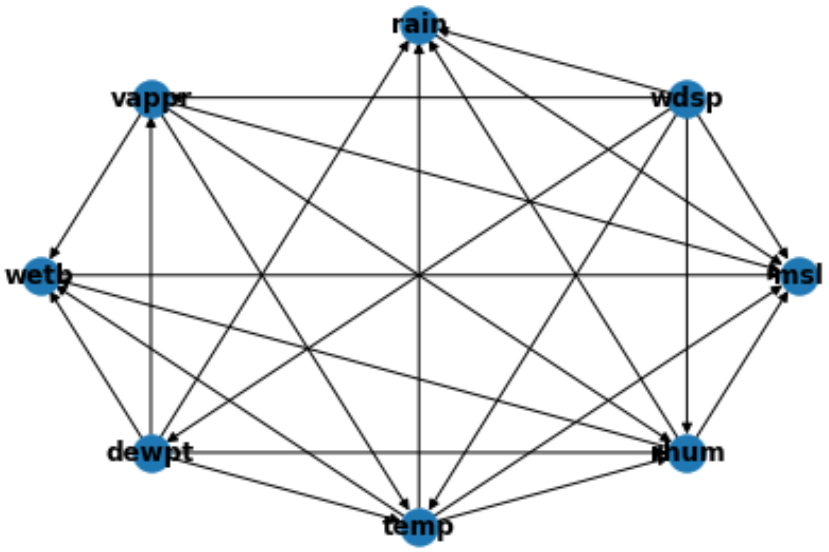
Step 5. Build a Bayesian network for the same set of variables but using 2 chosen algorithms for structural learning.

The algorithms were Tree Search and Hill Climb Search.

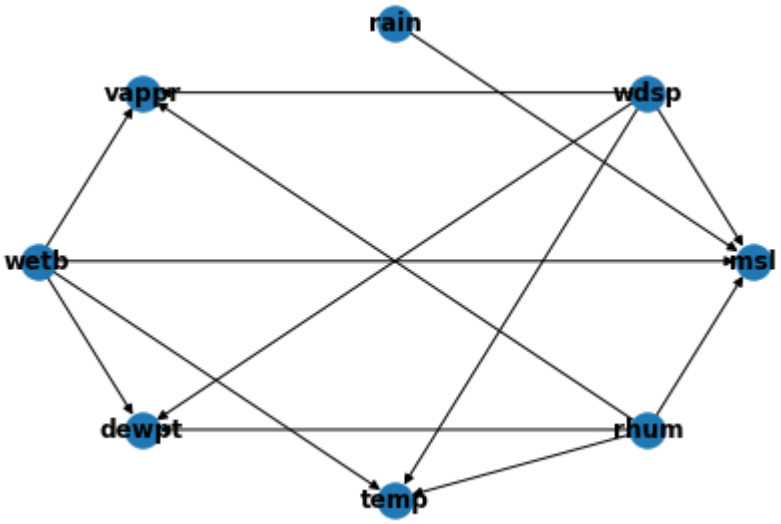
Tree Search gets the result



The Hill Climb Search gives the result below.



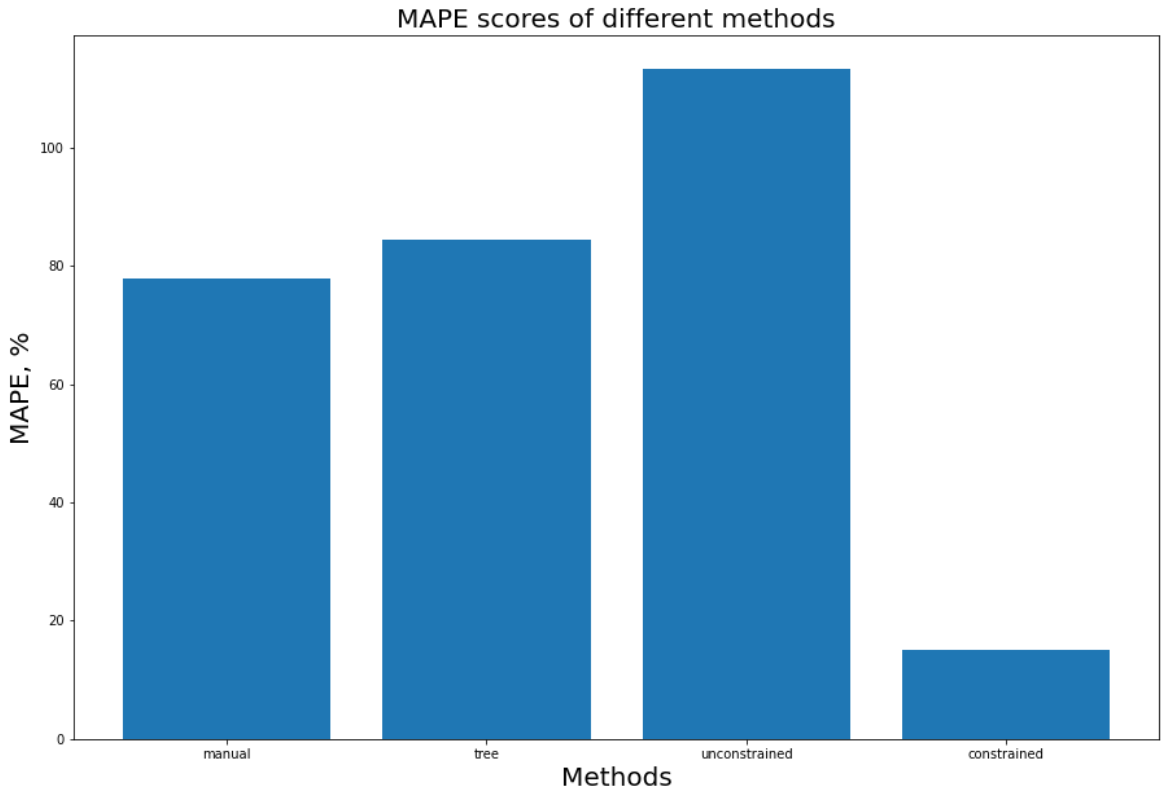
We see that it targets variables Temperature, Vapour Pressure and Dew Point Air Temperature. So add constraints to our model.



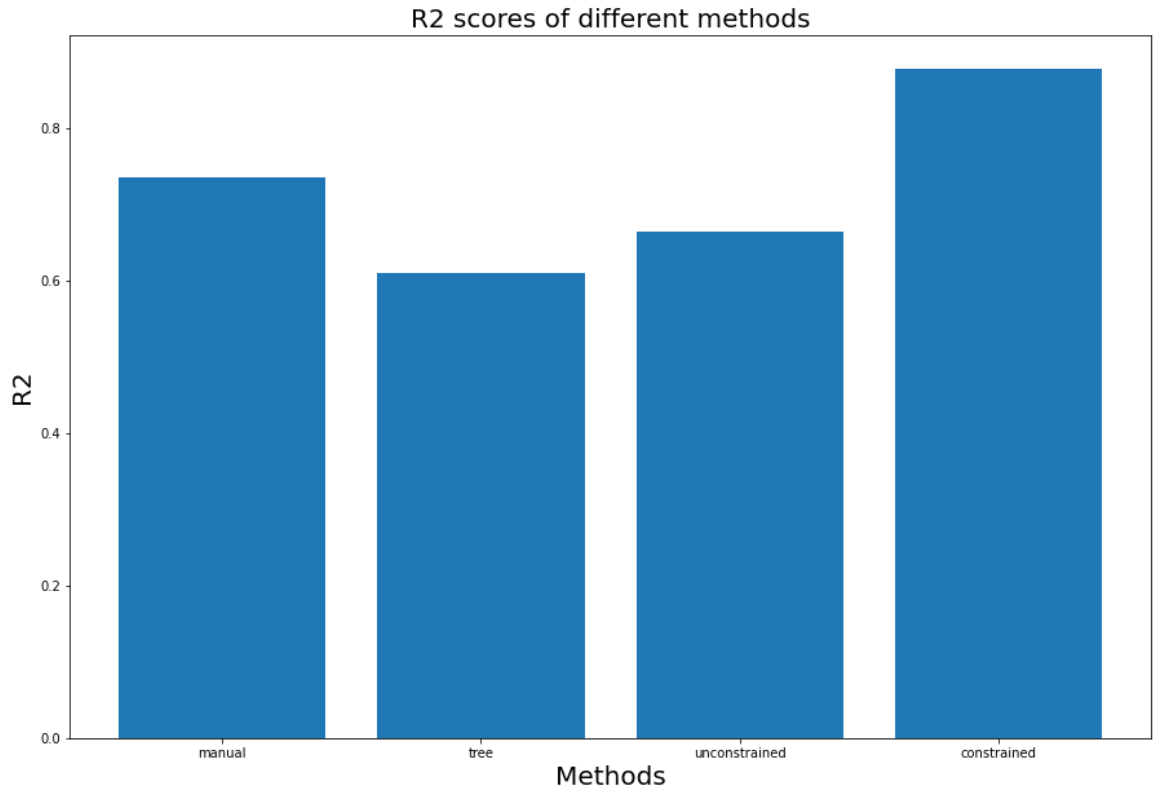
Step 6. Analyze a quality of sampled target variables.

For making quality analysis we need to do an inverse transformation of discretizing, because the initial parameters were continuous. So we can evaluate metrics for the continuous parameters. We chose R2, RMSE, MAPE.

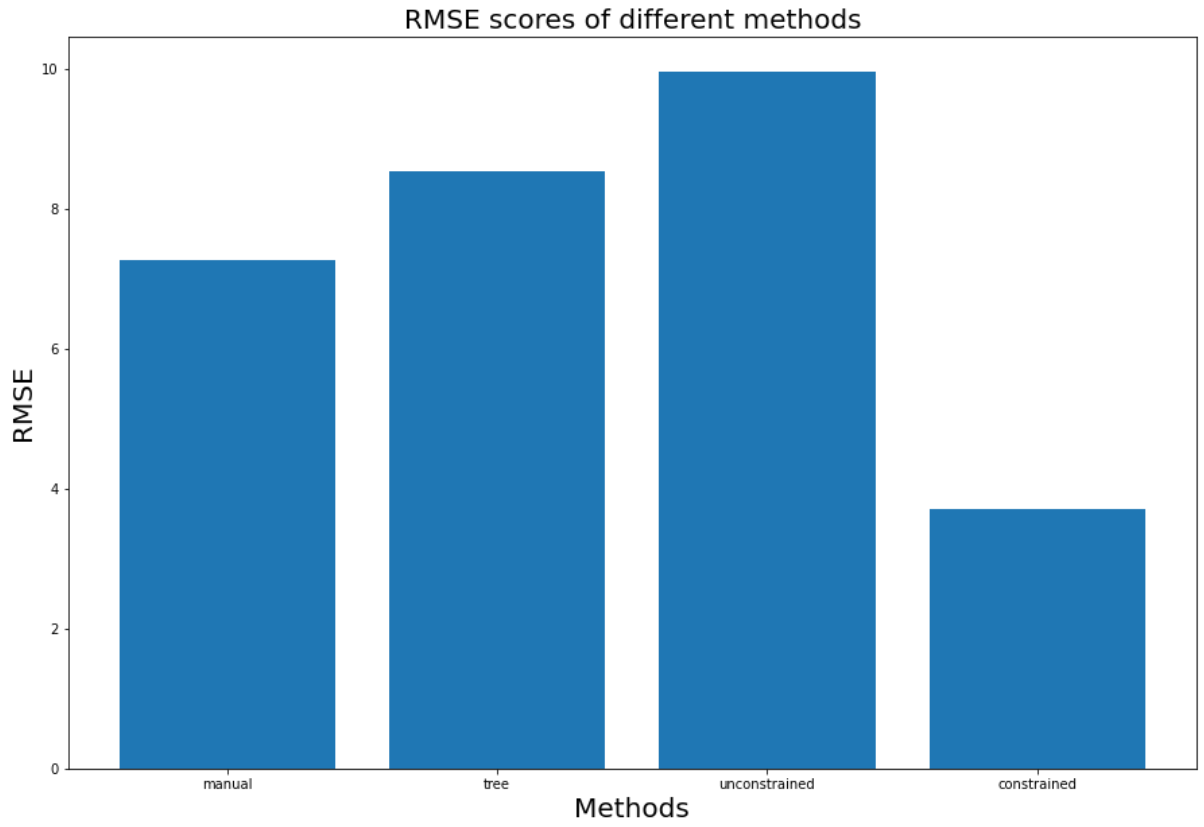
We can see that the best MAPE has constrained BN. The worst is the unconstrained.



The best R^2 score has a constrained model. The worst is the tree search model.



The best RMSE score has has constrained BN. The worst is the unconstrained.



	Manual model	Tree search model	Unconstrained model	Constrained model
MAPE	77.8%	84.3%	113.2%	15.0%
R^2	0.74	0.61	0.66	0.88
RMSE	7.27	8.54	9.95	3.7

Conclusion:

The best model over the all test is the Constrained model. The worst model is the Unconstrained model. All other model have the same not so well result.

Sourcecode

https://github.com/AAYamoldin/Methods-and-models-for-multivariate-data-analysis-2021-2022-ITMO_labs/blob/main/Lab3/main_lab3.ipynb