Predicting inflation rates in Poland

1. Introduction

Consumer Price Index (CPI) is a popular way to measure rate of inflation which is an important indicator of economical situation on a given territory. It shows how much prices change in given time period (usually a year).

It is crucial to keep inflation rates on a low level so that the economy can work smoothly. Also the government and businesses rely on CPI to make strategic decisions. Because of that, many financial institutions try to predict CPI for the following months.

The purpose of this study is to create a ML model that would be able to predict CPI for Poland based on data from previous months. The Focus will be to predict the index only for the next month.

2. Literature overview

In order to better understand the problem and get to know how similar problems have been solved, an extensive literature overview has been conducted. It turned out that forecasting CPI is a very popular problem with a lot of papers surrounding the topic. Many suggest to forecast CPI using historical values and other economical factors (e. g. unemployment, GDP). Based on the scores achieved by others[1] [2] [3] [4] [5] it was decided to try 6 different model architectures:

- LSTM network
- Classic neural network
- Random Forest
- KNN regressor
- CatBoost
- XGBoost

Detailed explanation of these methods is present below.

- KNN Regressor:
    - It derives from popular classification algorithm KNN (K-nearest neighbours). First a number K needs to be chosen, and it has to be a positive integer. Training is very simple as the model only needs to see the training data, it doesn't do anything with it. After being given an example to predict, algorithm finds k closest observations from the training dataset, then it aggregates target variables from those k observations. The aggregated result is the predicted value.
- Random forest:
    - It is based on decision tree algorithm. The idea behind decision tree is to split data based on some criterion into two groups, each group can be then split into another two groups with different criterion and so on. We can build a sequence of such splitting nodes and receive a tree-like structure, which receives data in the root and then based on different criteria the data is grouped into leaves. The idea is to have similar observations in each leaf, so that when the new data comes in, it falls into a

[1] https://www.sciencedirect.com/science/article/pii/S2666143820300120#sec0001
[2] https://ijics.com/gallery/18-may-1183.pdf
[3] https://www.sciencedirect.com/science/article/abs/pii/S0957417422012106
[4] https://www.sciencedirect.com/science/article/pii/S2666143823000042
[5] https://arxiv.org/pdf/2104.03757.pdf

leaf with most similar features, then target variable from observations in the leaf are aggregated and returned as a prediction for the sample. Random Forest creates a lot of those trees and aggregates results from them to give its final prediction.

- CatBoost:
    - This algorithm is also based on decision trees. However in this case the trees predict residuals, which is the difference between actual value and current prediction. The residuals are constantly updated during the learning process, while building more and more trees. CatBoost also uses a method called Ordered Target Encoding to convert categorical variables to numerical.
- XGBoost:
    - XGBoost also uses decision trees to predict residuals however, it has a more advanced algorithm to decide how data should be split and which branches should be kept. Contrary to CatBoost, XGBoost also allows non-symmetrical trees.
- Neural Network (NN):
    - Neural network are composed of layers of neurons. Each neuron is like a simple linear regression model – it takes n-dimensional input, multiplies value in each dimension by a corresponding trainable coefficient and adds bias value which is also trainable. Finally activation function is applied to the sum. Neurons are composed into layers, so that input for the next layer is output from the previous layer. Since each neuron produces one value, the input data for each layer has as many dimensions as many neurons were in previous layer. The first layer accepts the data, and last one consists of one neuron with predicted value.
- LSTM Network
    - LSTM is concept introduced to Recurrent NN. The difference between RNN and NN is that RNN neurons contain a feedback loop that passes the data to itself. This allows to work with more complicated data. LSTM extends this concept by creating an advanced feedback loop with long-term and short-term memory. The latter is subject to more changes and manipulations while data flows through the model, and the other is used to preserve data from earlier observations.

3. Preparing dataset

Data has been collected from various sources and includes following features:

- CPI – target variable (https://bdm.stat.gov.pl/)
- Economic growth (https://pl.wikipedia.org/wiki/Gospodarka_Polski)
- Stock indices (https://www.biznesradar.pl/notowania-historyczne/WIG20,152, https://www.biznesradar.pl/notowania-historyczne/WIG,152)
- PLN – USD conversion rate (https://www.money.pl/pieniadze/nbparch/srednie/?symbol=USD)
- Income, expenses of national budget (https://bdm.stat.gov.pl/)
- Unemployment (https://stat.gov.pl/obszary-tematyczne/rynek-pracy/bezrobocie-rejestrowane/stopa-bezrobocia-rejestrowanego-w-latach-1990-2023,4,1.html, https://stat.gov.pl/obszary-tematyczne/rynek-pracy/bezrobocie-rejestrowane/liczba-bezrobotnych-zarejestrowanych-w-latach-1990-2023,6,1.html)
- Detailed CPI indices (https://bdm.stat.gov.pl/)

Variables were selected based on literature overview and personal intuition.

Another variable named „score” has been created by subtracting expenses for income of national budget.

Where it was applicable new features which describe monthly and yearly change of given variable have also been created by dividing new value by older value and multiplying by 100.

Below you can find a list of columns:

- growth – Economic growth in Poland
- wig – Stock exchange index which lists hundreds of companies from Warsaw Stock Exchange
- wig20 – Stock exchange index which lists 20 largest companies of Warsaw Stock Exchange
- USD – price of one US dollar in PLN
- income – income of national budget in Poland
- expenses – expenses of national budget in Poland
- score – calculated by subtracting expenses from income
- unemployed – number of unemployed people in Poland
- unemployment – share of unemployed people
- inflation – consumer price index (target variable)

Columns with _1m or _1y ending represent change of given variable in relations to last month or last year respectively.

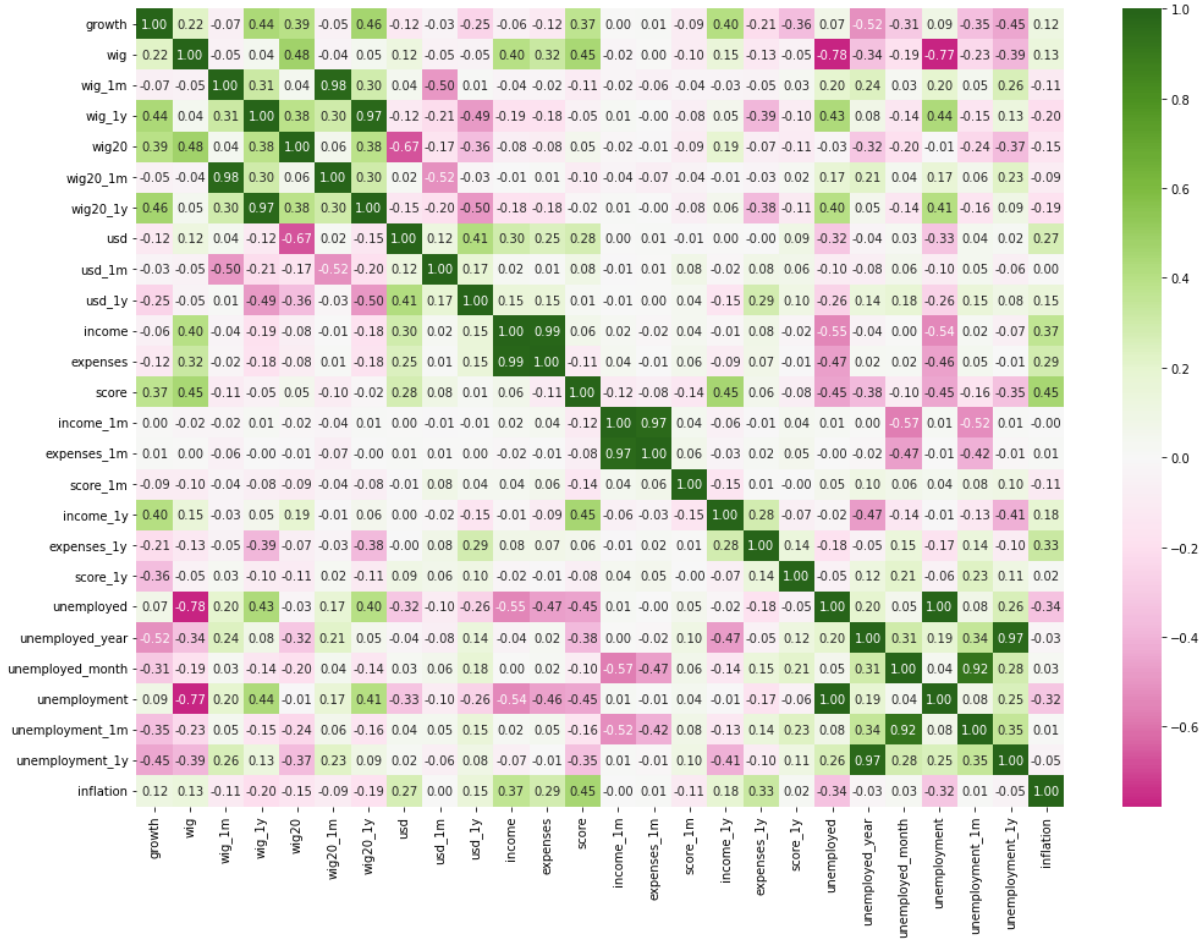
One row consists of data for one month between 2003 and 2022.

4. Data analysis

Exploratory analysis of collected data has been performed to better understand the dataset and influence on CPI by other variables.

Data selected for analysis covers time period from 2003 to 2022 inclusive (240 months in total).

To begin with, a correlation matrix with Pearson coefficients has been plotted with features other than detailed CPI values.



We can see a few interesting things, some of which were less surprising than others:

- Number of unemployed people is extremely correlated with unemployment. This is because there were little changes in population during the period which the dataset covers
- income and expenses in national budget are strongly correlated
- The scores of national budget are high when the income for the national budget is higher than in respective month in previous year.
- The wig index is strongly negatively correlated with unemployment
- Larger national budget also seems to reduce unemployment
- The wig20 index is negatively correlated with the prices of USD
- variables concerning national budget, unemployment, and price of USD seem to affect inflation the most

Then, variables have been plotted to search for seasonality in data. It was noted that income and expenses of national budget are highly seasonal: the government seems to gather and spend the smallest amount of money at the beginning of the year, then budget increases in the linear fashion throughout the year until December, only to fall again at the beginning of the year.

Unemployment also seems to show some seasonality, as it usually peaks around months of February and January.

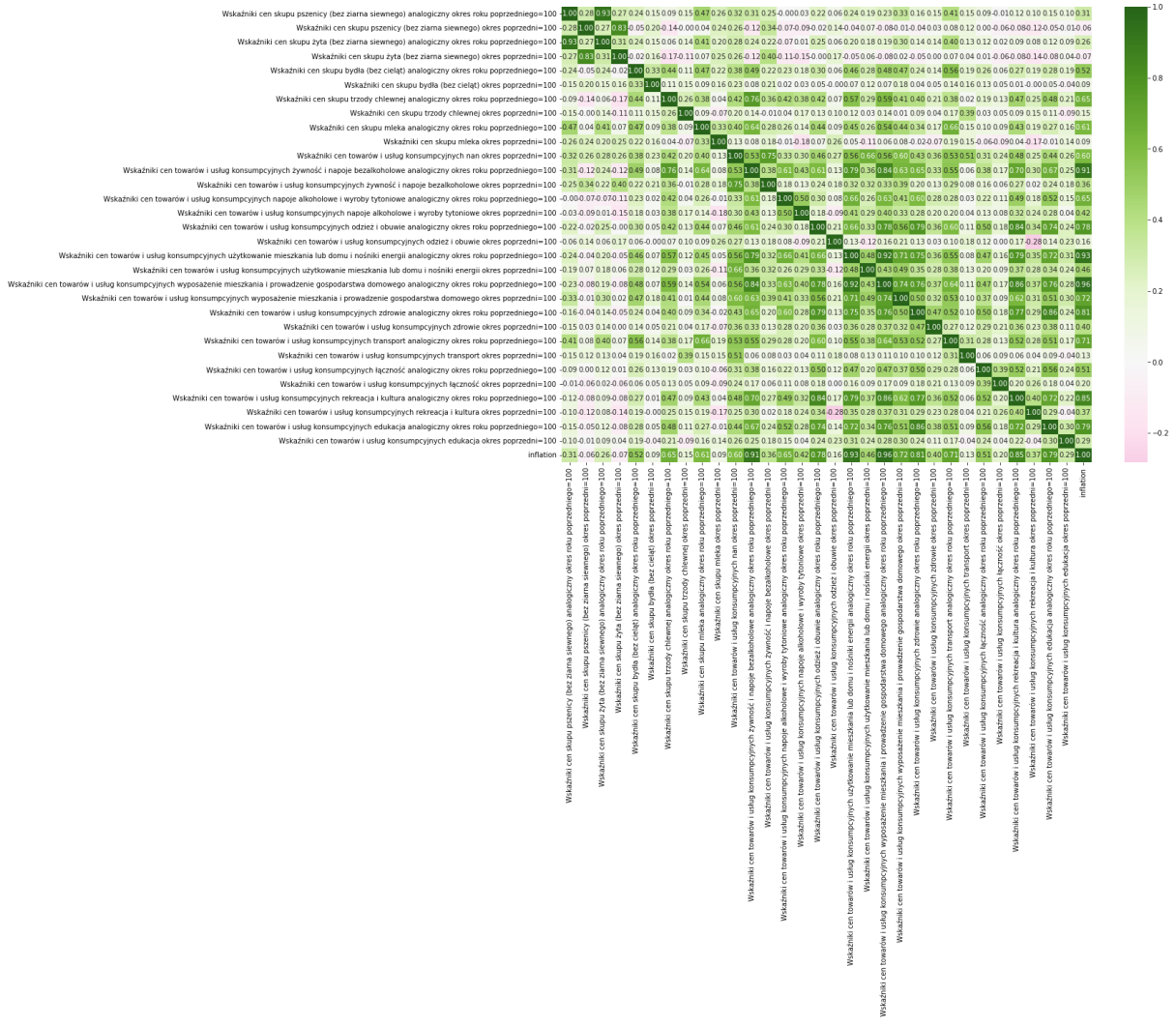
There was also an attempt to cluster data using KMeans algorithm. Based on the elbow method it was decided to cluster data into 7 clusters.

Quite interestingly, one cluster contained all the records from January, it looks like its the only distinguishable month as other clusters contained data from different months. It also looked like each cluster contained data for around 3-4 consecutive years, with small amount of exceptions. This means that the economy tends to be in a similar state for a couple of years, rather than constantly change according to time of year. One of the clusters was also interesting as it contained exclusively data for months in 2022. Thus, it would appear that 2022 was the most unique year for economy out of all taken into consideration.

There has also been an attempt to divide data into 12 clusters however, the conclusions are almost the same as with 7 clusters. However one cluster consisted only of April 2020. This distinction likely comes from COVID-19 lockdowns which were introduced shortly before that month.

When clustering was performed with only 2 clusters, data has been divided into a cluster with months during which economy was growing and performing very well, and other cluster which contained data for more difficult times.

After that, data concerning detailed CPI values was analysed. However it quickly appeared that almost all the features in this part of the dataset were highly correlated with each other. It should not be surprising considering that this data consists only of CPI indices for various goods and services.



Yet, it can be seen that there are few variables that behave slightly different the others. CPI for clothes and shoes seems to be the most outstanding one and is actually slightly negatively correlated with the general CPI. CPI of milk also seems to behave differently than other variables.

There was an attempt to cluster months based solely on CPI indices, but the conclusions from the results were pretty much the same as with clustering based on the rest of the data.

5. Preparing models

The first attempts to create specific ML models have been summarised in document called „first_attempt.pdf".

To summarize this part: various models and datasets have been tested. It was noted that KNN and LSTM models were the most effective ones.

Feature selection task was also performed. Despite the presence of many different features it has been ultimately decided to only use two columns: inflation and USD. Adding more columns did not significantly improve models' performance.

The models struggled to predict values from recent months, that is because the rate of inflation has been so high. In fact all samples from training data have significantly lower CPI values than those in one of the testing datasets. This confused all of the models, as testing dataset required them to predict values much higher than they ever saw.

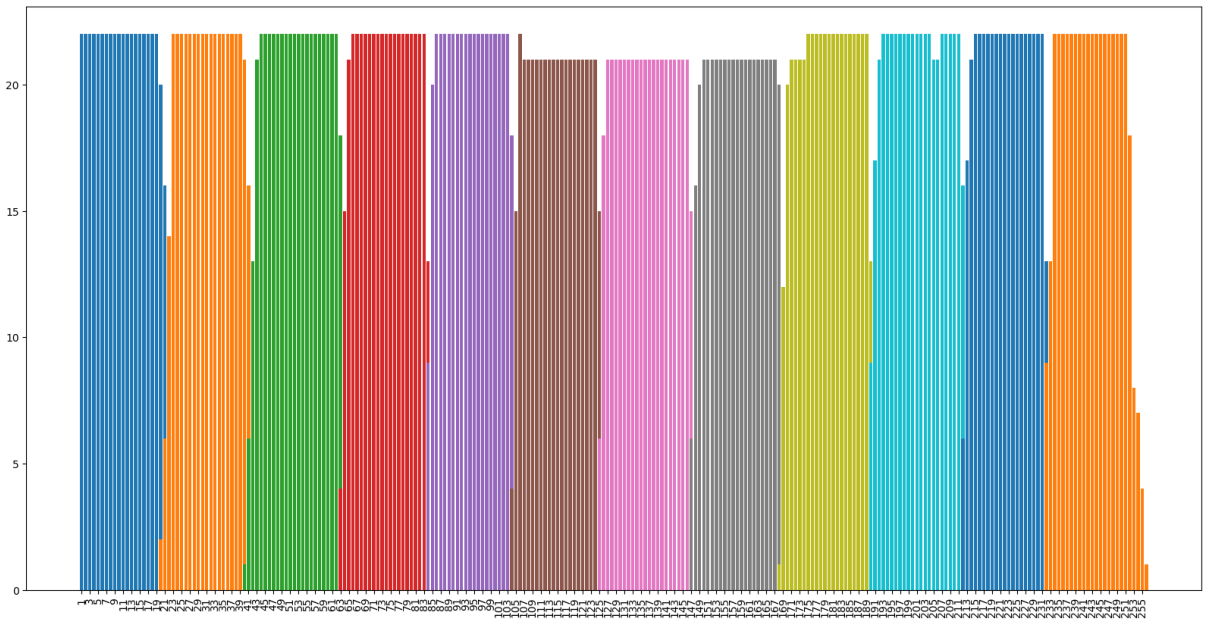
However, now that it has been decided to use only two columns, the dataset can be extended as data for USD and inflation is available for more years than other variables. This would allow to create a training dataset with values that would be comparable with the test dataset.

Hence it has been decided to redo this stage with more data, and more thorough grid search for models' parameters.

6. Creating final models

The dataset for this part has been created using only values for USD conversion rate and cpi. However there was a problem with USD data as some values have mislabelled months. It was possible to get the correct value from record id as it included the correct year and day of the year, however the number of days counted only working days in the polish calendar, so it would be tiresome to map the day number to the correct month.

Because of that a bar plot has been created to show which day number corresponds to which month (this plot is based only on data that had correct labels):



On x-axis one can see number of the day that given bar is concerning, y-axis shows how many rows with this day number have been assigned to a given month. Colours are used to represent different months.

There is a gap for summer months which is due to the fact that correctly labelled dataset had values from mid-September 2001 to early June 2023.

Other smaller gaps represent which day numbers were assigned to more than one month. Based on these gaps, 11 thresholds have been set to divide the rest of the data into correct months.

This estimation had 97% months labelled correctly in this part of the dataset, so to save time it was decided to estimate months for the rest of the data using this method. The gain from exactly mapping days to months was likely not worth it.

This allowed to create a dataset with 352 rows. Each row contained USD and CPI data for each month in a 12 months window and a target variable which would be CPI value for the next month.

Data was split into 3 datasets:

- Train – consisted of months from 1993-12 to 2017-04
- Test 1 – consisted of months from 2017-05 to 2021-01
- Test 2 - consisted of months from 2021-02 to 2023-03

This split is used for a real-world-like simulation for time series, because we have to predict future values based on what we already know from the past. CPI in this problem can be obviously treated as time series.

The decision to use two datasets comes from the fact that most recent data has unusually high CPI values, so the metrics of the models can become really confusing. The split of test dataset will allow to have a better understanding of the model metrics.

Before data was fed into the models it has been standardised by subtracting mean and dividing by standard deviation of values in the training dataset.

Models were evaluated based on mean squared error.

Then grid search took place and the best parameters for each model have been decided based on results achieved on test 1 dataset.

Below you can find 6 models selected for final comparison, each of these was the best model in its own architecture.

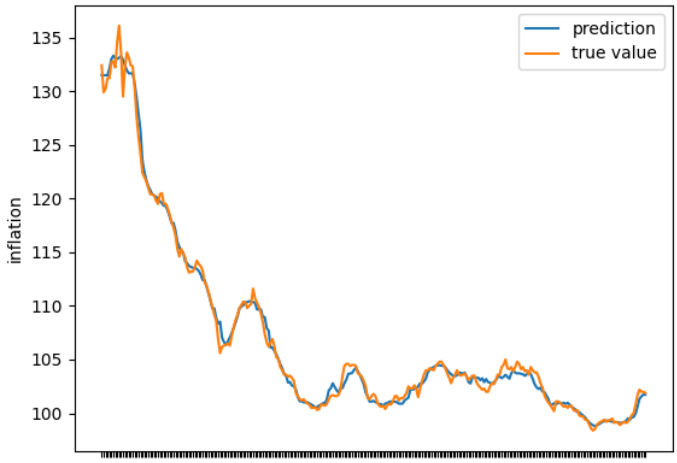
1. KNN
    a. n_neighbors=7
    b. metric="cityblock"
2. CatBoost
    a. depth=4
    b. n_estimators=200
    c. learning_rate=0.03
3. XGB
    a. max_depth=4
    b. n_estimators=200
    c. learning_rate=0.03
4. RF
    a. max_depth=5,
    b. n_estimators=200
    c. criterion=”squared_error”
5. NN
    a. Layers:
        i. Dense(32); activation='relu'
        ii. Dense(32); activation='relu'
        iii. Dense(16); activation='relu'
        iv. Dense(1); activation=’linear’
    b. loss='mean_squared_error'
    c. optimizer='adam'
    d. epochs=40
    e. batch_size=32
6. LSTM
    a. Layers:
        i. LSTM(32); activation='tanh'
        ii. LSTM(64); activation='tanh'
        iii. LSTM(32); activation='tanh'
        iv. Dense(1); activation=’linear’
    b. loss='mean_squared_error'
    c. optimizer='adam'
    d. epochs=40
    e. batch_size=32

Quite interestingly all models based on decision trees achieve best score with similar parameters. It is also interesting that KNN achieved a best score using city block metric, as this metric is rarely useful in such problems.

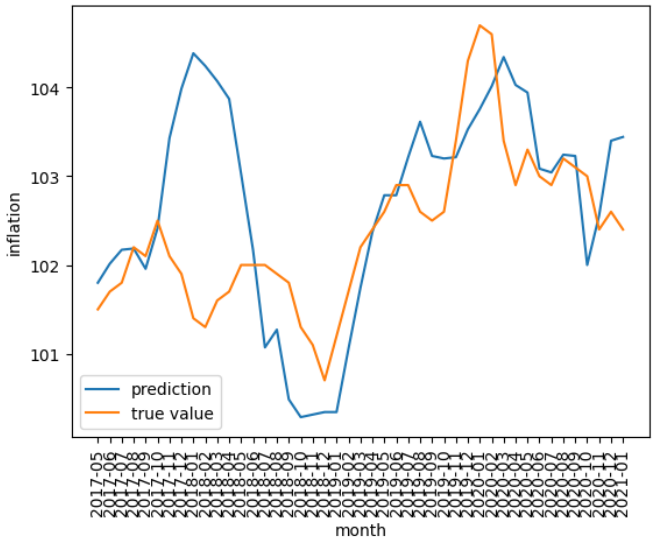
To visualise performance of those models, plots with actual and predicted data have been created:

KNN

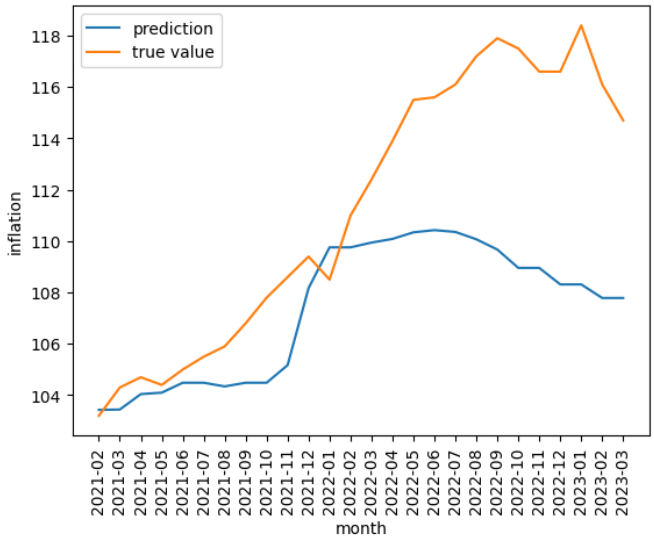
Train:



Test 1:

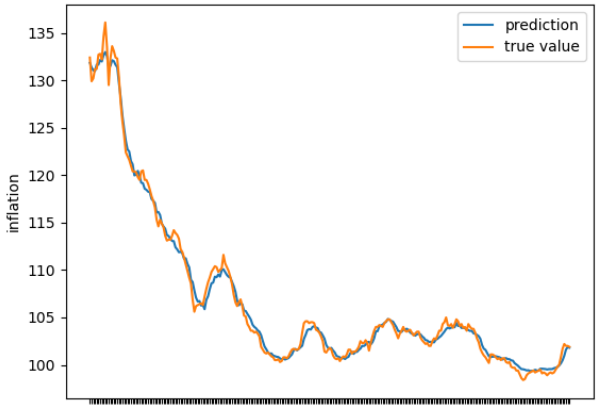


Test 2:



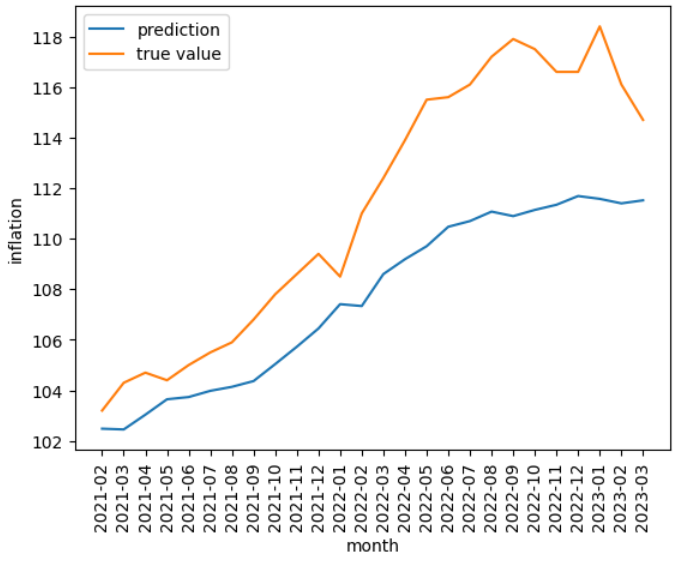
CatBoost

Train:



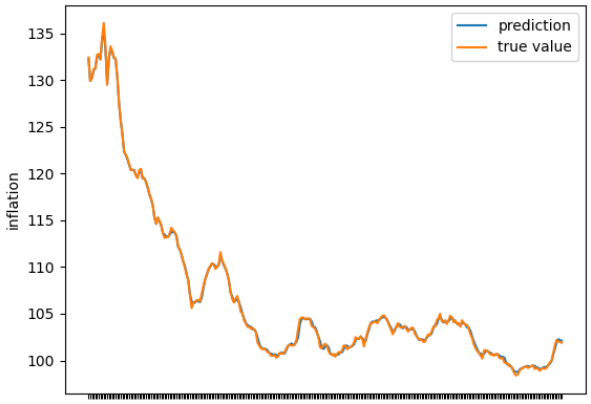
Test 1:

Test 2:

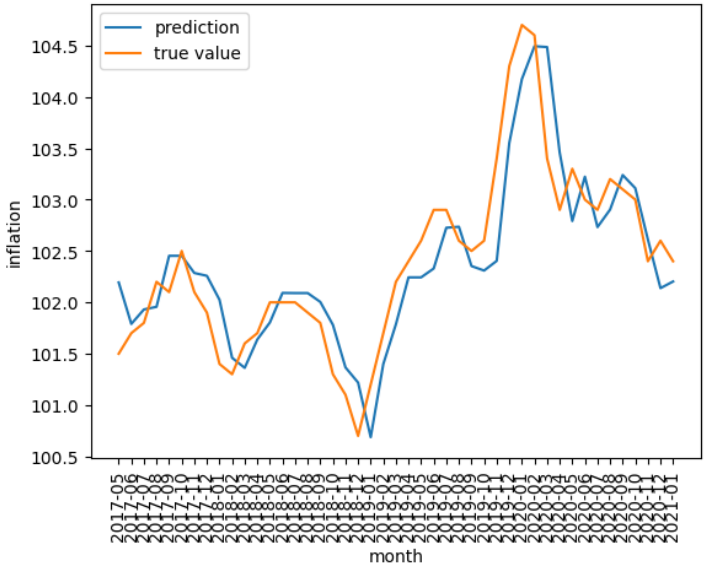


XGB

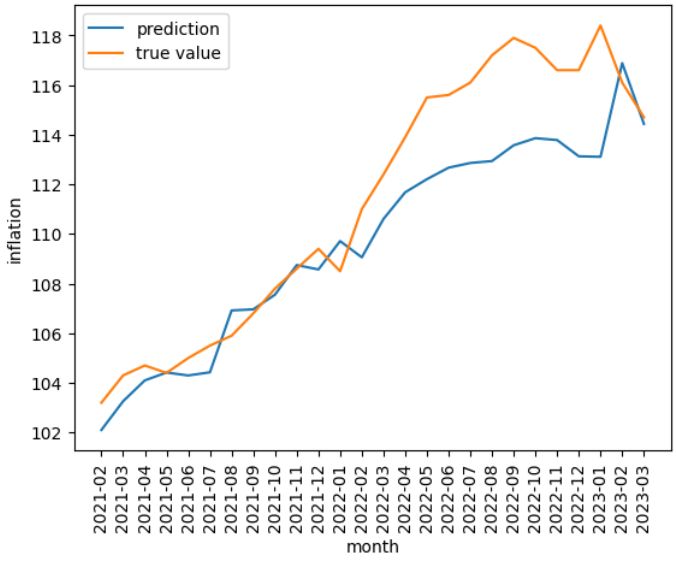
Train:



Test 1:

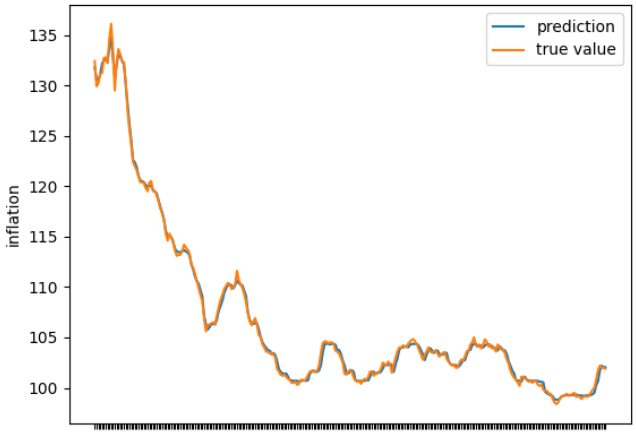


Test 2:

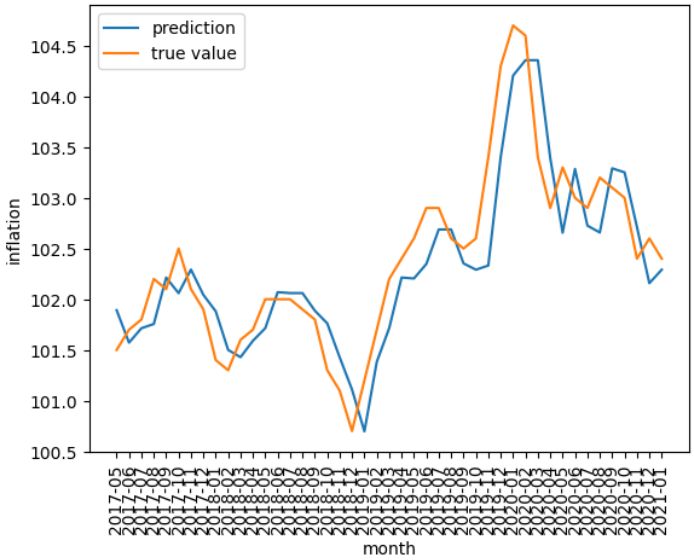


RF

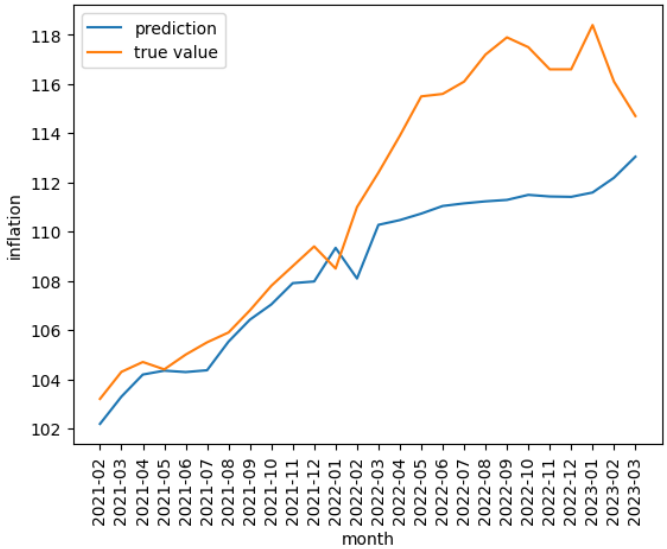
Train:



Test 1:

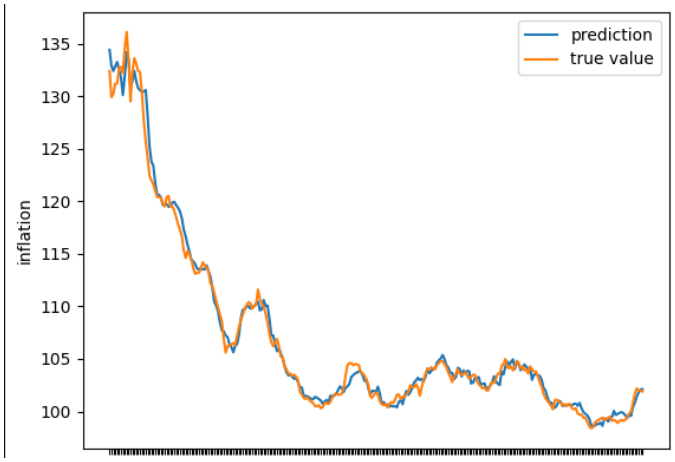


Test 2:

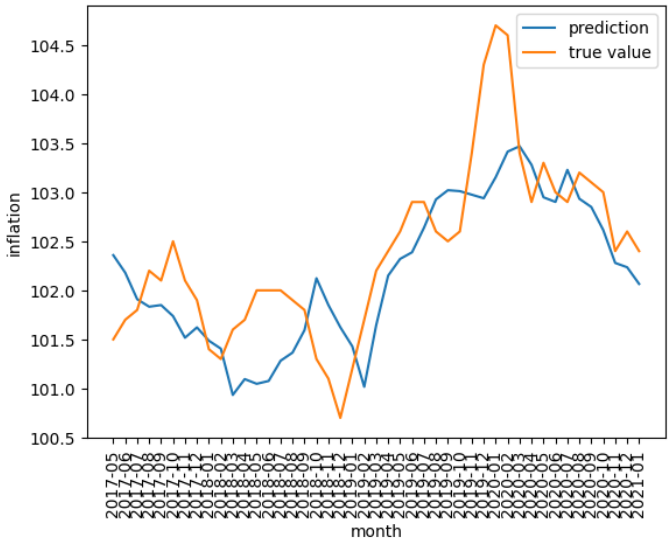


NN

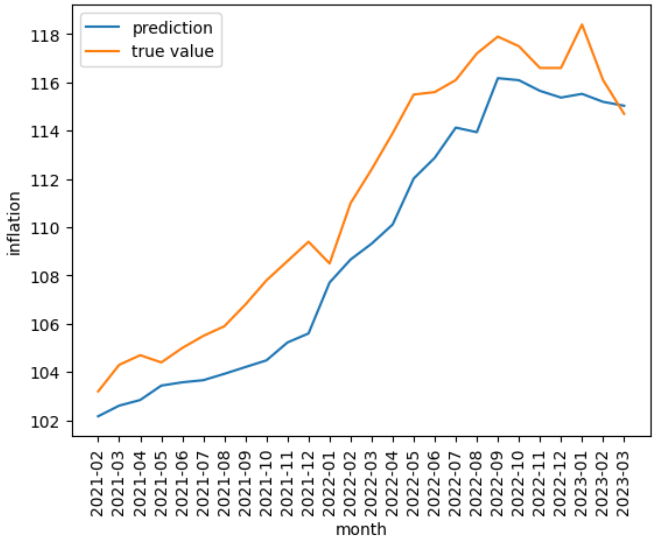
Train:



Test 1:

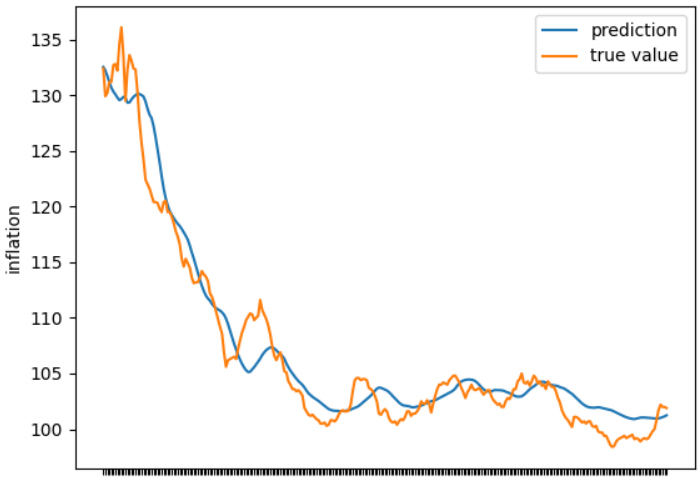


Test 2:

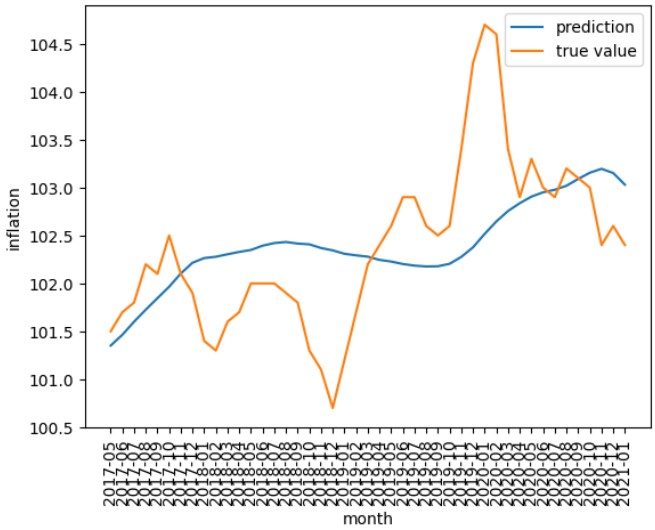


LSTM

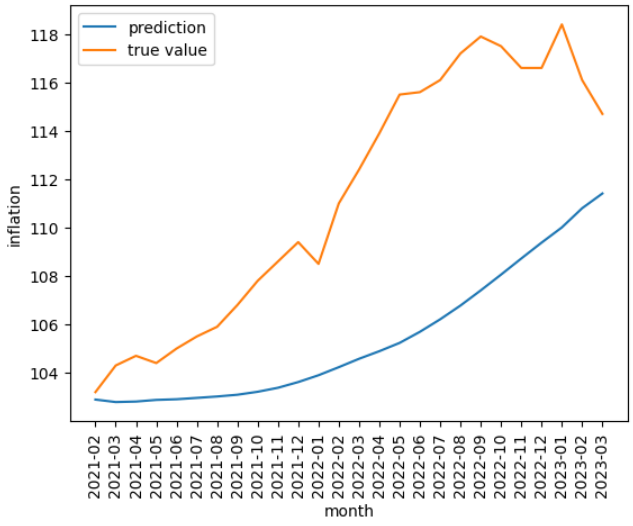
Train:



Test 1:



Test 2:



It can be seen that almost all models are able to predict training data flawlessly. Except LSTM model which doesn't really work. This architecture probably requires more training samples to be able to work properly.

KNN model seem to be somewhat following the proper curve but it seems to have some random predictions that are very far from expected value. It also doesn't cope well with extreme values.

Based on test 1, CatBoost looks like one of the best here. It seems to properly identify trends however the exact values are often slightly off. It also has troubles with extreme values.

XGB and RF models seem to just expect the cpi value to remain stationary. They usually return the most recent CPI value as its prediction for the next month. There was an attempt to combat this behaviour by removing the most recent CPI from the dataset, however those models would just use value from 2 months before which became the most recent value. It is interesting that on test 2 they seem to get more creative and while RF just stops working with values above certain threshold, XGB works really well with extreme values.

NN model is also very interesting. It seems to be identifying trends very well, except some places when its prediction is way off. It usually predicts value which is slightly smaller than actual observed CPI. Apart from that flaw, it deals well with extreme data.

To better understand the results, the MSE metrics have been collected for all of these models:

Model	Train	Test 1	Test 2	Test 1 and Test 2
KNN	0.462	1.142	26.227	10.328
CatBoost	0.496	0.213	17.030	6.371
XGB	0.043	**0.162**	**5.701**	**2.190**
RF	0.125	**0.166**	12.854	4.812
NN	0.770	0.362	**5.444**	**2.224**
LSTM	4.654	0.640	44.815	16.816

It appears that the tactic with copying most recent value is beneficial as it results in best MSE for test 1.

What we can see is that results from test 2 are much more influential on the combination of two test sets. That is because the values are simply larger, so they increase MSE more drastically.

When it comes to extreme values the NN is coping the best, however it doesn't really do so well on test 1. On the other hand CatBoost is doing very well on test 1 and is the best model that doesn't just copy the most recent value. But it doesn't work well with extreme values.

XGB model also seems to be doing very well, and it would be a way to go if one doesn't mind his oversimplified predictions that appear very often. Even if they make the MSE score very low, one can never predict if cpi will increase or not, because the model almost always assumes that it will not change. Also XGB has a very low train score compared to other values, so it might have been overfitted.

The recommendation would be to either use NN model as it is the best for extreme values, and is not that bad with regular CPI, or use CatBoost model. In fact one might switch between these two models based on how stable the economy is, as this appears to be the best solution for this problem.

7. Future work

Some models can definitely still be improved, especially LSTM model. XGB model should likely be examined to verify overfitting and to possibly eliminate this problem.

Best models in this research are very large and complicated, so it might be worth to conduct some analysis on how they behave e. g. SHAP.

It might be worth it to experiment with more solutions based on neural networks.

Predicting CPI for the next month is not actually that useful for business and government, and it would be desirable to predict inflation for a few upcoming months or even longer. Hence it might be worth to evaluate models on longer prediction time or build new models suitable for long time predictions.