

Business Intelligence Case Challenge

Franciszek Kornobis, Jędrzej Słupski

Maj 2024

Uniwersytet im. Adama Mickiewicza w Poznaniu

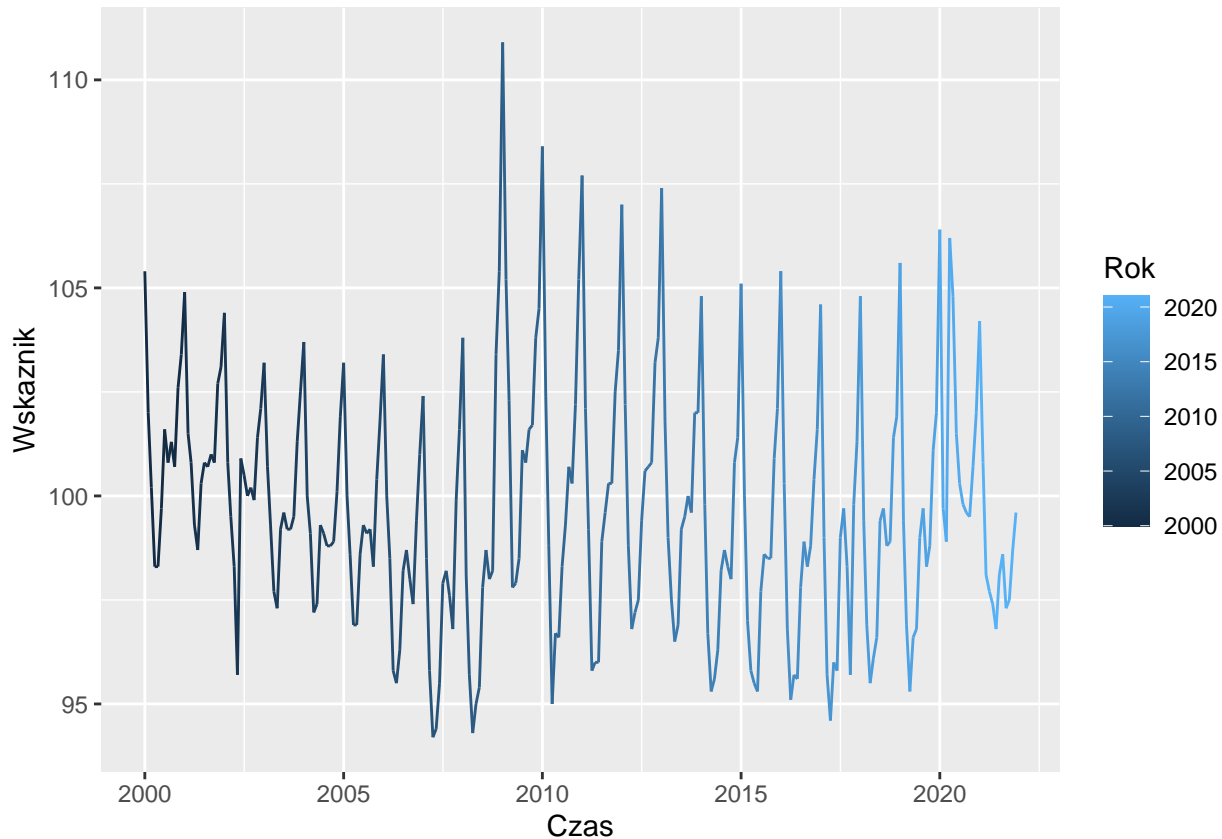
1. Wstęp
2. Opis metod prognozowania
 1. Modele regresyjne
 2. Prophet
 3. TBATS
 4. Ostateczna predykcja
3. Przewidywanie
4. Podsumowanie

1. Wstęp

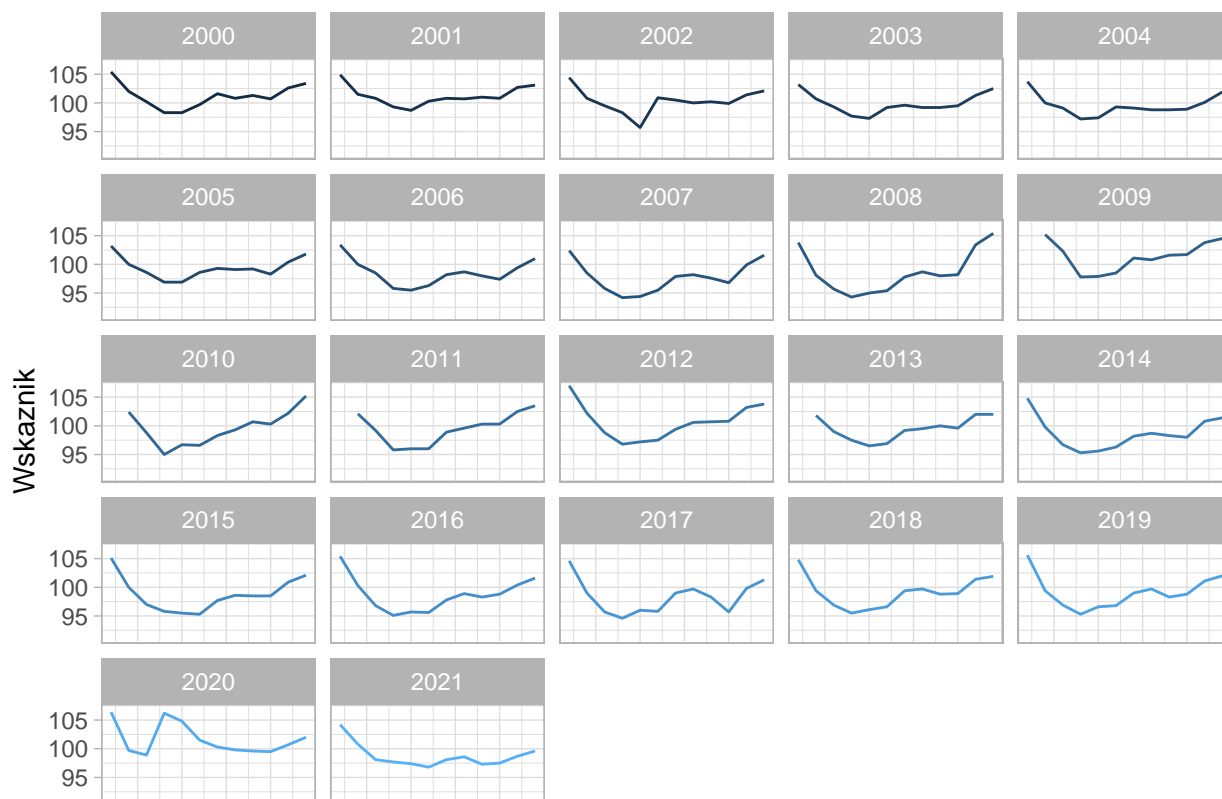
Wyzwanie konkursowe polega na predykcji wskaźnika bezrobocia na lata 2022-2023, na podstawie danych z 2000-2021. Analizę będziemy przeprowadzać przy użyciu języka R w programie RStudio, do obliczeń i przewidywań użyjemy m.in. z bibliotek takich jak *stats*, *tseries*, *forecast*, *prophet*, do wizualizacji - *ggplot2*, a do utworzenia ostatecznego raportu posłużymy się oprogramowaniem *RMarkdown*.

Wejściowe dane przyjmują formę szeregu czasowego - realizacji *procesu stochastycznego*, którego dziedziną jest czas - w tym przypadku po 12 miesięcy z 22 lat. Procesem stochastycznym $(X_t)_{t \in T}$ nazywamy rodzinę zmiennych losowych z pewnej przestrzeni probabilistycznej, przyjmującą wartości z przestrzeni mierzalnej.

Dane wskaźnika bezrobocia w latach 2000-2021 przedstawiają się w następujący sposób:



Z podziałem na lata:

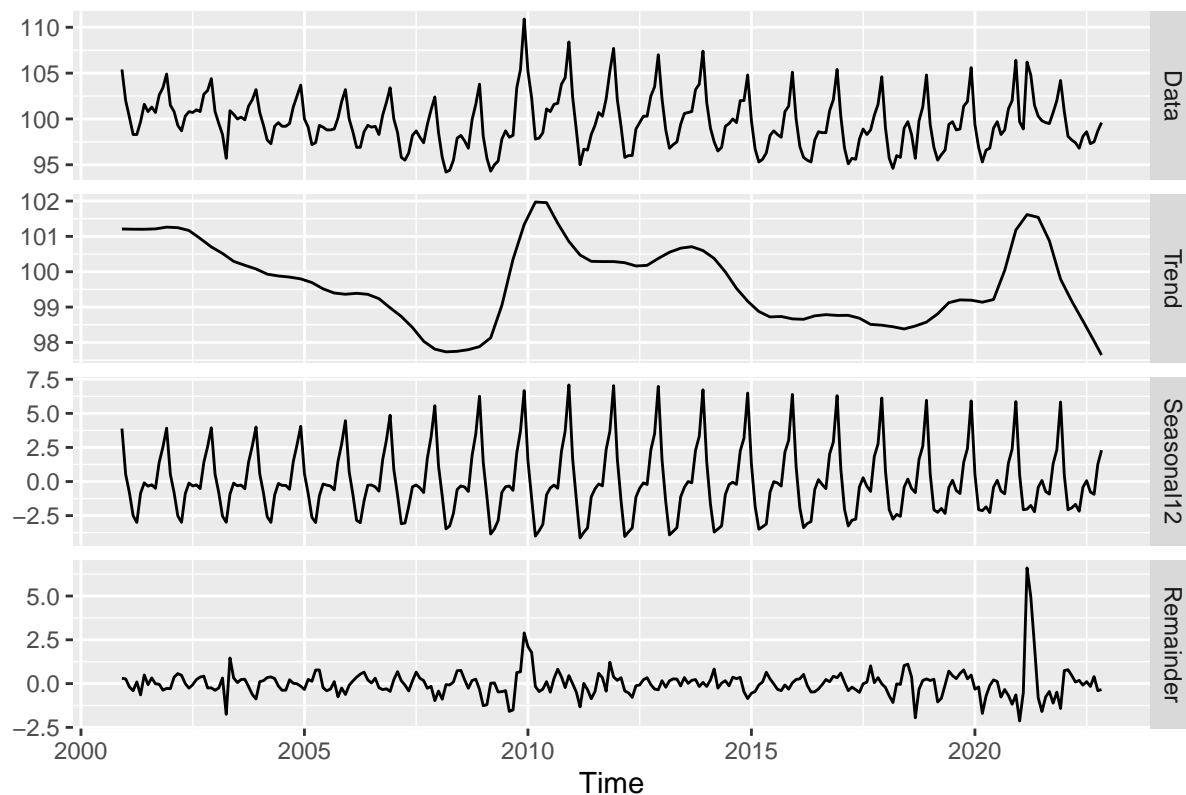


Od razu zauważamy, że dane podlegają dostrzegalnej okresowości - wartość wskaźnika spada na początku roku, w okolicy połowy roku wzrasta, by potem delikatnie spadać, lub utrzymywać się aż do października, a na koniec roku znowu wzrasta. Sezonowość występuje oczywiście w okresie 12 miesięcy.

By sprawdzić podejrzenia wynikające z wizualnej obserwacji powyższych wykresów, przeprowadzamy dekompozycję STL (Seasonal Decomposition of Time Series by Loess). Ma ona wskazać składniki szeregu czasowego - jego trend, sezonowość oraz reszty.

```
library(forecast)
```

```
ts_stl <- ts(df$Wskaźnik, frequency = 12, start = c(2000,12))
autoplot(mstl(ts_stl))
```



Powyższa metoda opiera się na przedstawieniu punktów szeregu czasowego $(y_i, i \in T)$ jako suma komponentów sezonowości s_i , trendu t_i i reszty r_i :

$$y_i = s_i + t_i + r_i$$

oraz estymacja owych komponentów. [1]

Obserwując surowe dane, widzimy pewną anomalię w roku 2020 - wyraźny skok wartości wskaźnika bezrobocia w kwietniu w wyniku wybuchu pandemii COVID-19. To zaburzenie w danych w większości przypadków obniży jakość prognozy, ponieważ trend w tym okresie zostaje naruszony. Z tym problemem możemy poradzić sobie na kilka sposobów. Istnieje opcja podstawienia średniej wartości wskaźnika z każdego miesiąca do odpowiednich miesięcy z 2020. Można wziąć średnie globalne, lub jedynie z kilku ostatnich lat. Nie ma również większych przeszkód, by zupełnie pominąć ten rok w obliczeniach. Zbadamy także ideę, by użyć danych z lat 2000-2019, by “przewidzieć” wartości z 2020, a następnie dokonywać obliczeń przy użyciu nowych danych z 2020 do predykcji 2022-2023. Dokonamy analizy tych metod w rozdziale 2.

2. Opis metod prognozowania

2.1 Modele regresyjne Jako pierwszą metodę predykcji wybraliśmy regresję liniową ze względu na miesiące. Wartości wskaźnika z n-tego miesiąca z lat 2000-2021 wyznaczają przewidywaną wartość wskaźnika z n-tego miesiąca na lata 2022 i 2023. Ze względu na podatność regresji liniowej na obserwacje odstające zdecydowaliśmy się usunąć rok 2020, ponieważ metoda ta nie wymaga ciągłości danych. [2]

```
library(stats)
library(tseries)
library(forecast)

predict_monthly <- function(data, month) {
  monthly_data <- data %>% filter(M.c == month)
  model <- lm(Wskaznik ~ Rok, data = monthly_data)
  future_years <- data.frame(Rok = c(2022, 2023), M.c = month)
```

```

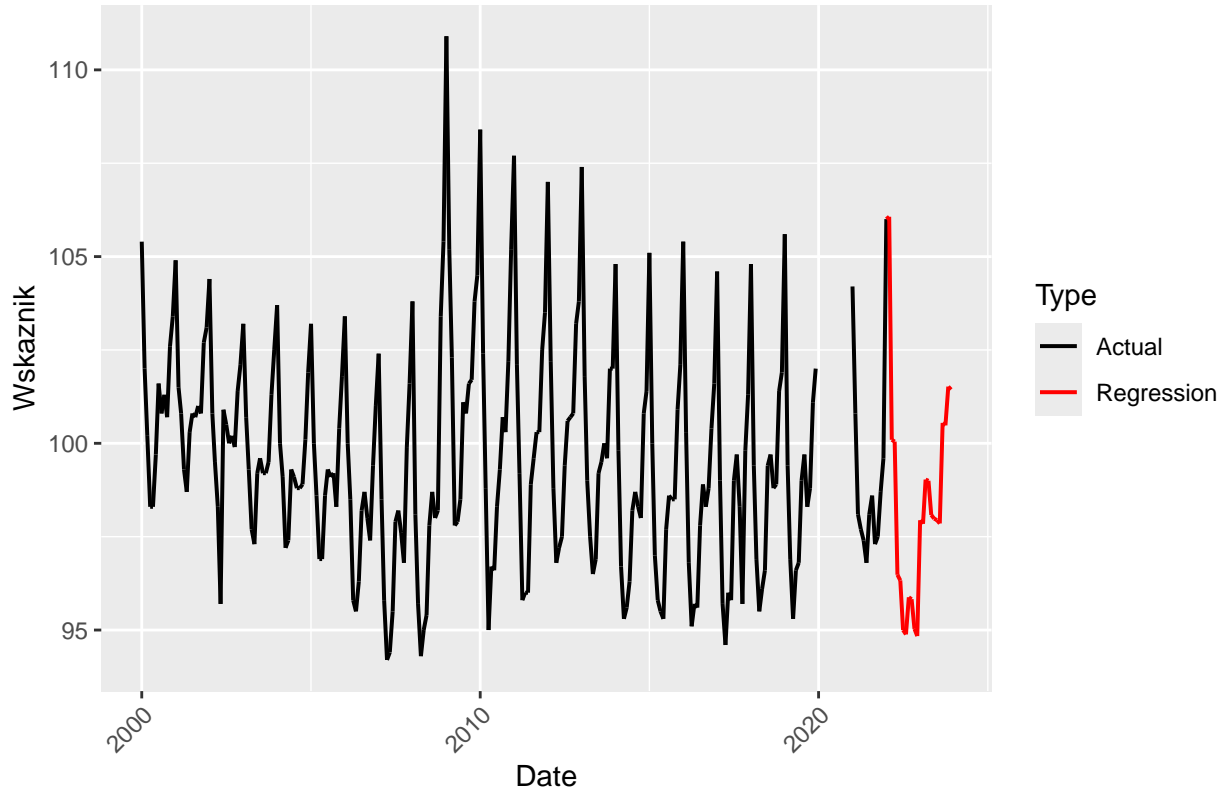
predictionsWith2020 <- predict(model, newdata = future_years)
return(data.frame(Rok = future_years$Rok, M.c = future_years$M.c, Wskaznik = predictionsWith2020))
}

predictionsWithout2020 <- lapply(1:12, function(m) predict_monthly(data, m))

predictionsWithout2020_df <- do.call(rbind, predictionsWithout2020)
combined_data <- bind_rows(data, predictionsWithout2020_df)

```

Wskaznik bezrobocia w czasie wraz z predykcja



Ponieważ regresja liniowa jest ogólnym modelem, który nie jest dedykowany dla szeregów czasowych, zdecydowaliśmy się rozszerzyć tę metodę o model ARIMA (Auto Regressive Integrated Moving Average). [3]

W modelu tym, część autoregresyjna (AR) jest zasadniczo formą regresji liniowej, gdzie bieżąca wartość szeregu czasowego jest modelowana jako liniowa kombinacja jego przeszłych wartości. Drugą częścią modelu ARIMA jest I(integrated), odnosi się do różnicowania celem uczynienia szeregu czasowego stacjonarnym. MA (Moving Average) odnosi się do modelu, który wykorzystuje zależność między bieżącą wartością a wcześniejszymi błędami losowymi. Model ARIMA(p,d,q) można opisać za pomocą 3 liczb, p oznacza parametr autoregresyjny, d - rząd różnicowania oraz q - parametr średniej ruchomej.

Ponieważ ARIMA również jest podatna na obserwacje odstające, ale wymaga ciągłości danych, zdecydowaliśmy się zastąpić rok 2020 predykcją modelu ARIMA na podstawie lat 2000-2019 i wykorzystać te dane aby przewidzieć wartości wskaźnika na podstawie lat 2000-2021 ze sztucznymi danymi z roku 2020. Mając świadomość możliwości wystąpienia efektu kaskadowania błędów, porównamy to rozwiązanie z klasycznym zastąpieniem roku 2020 średnimi z pozostałych lat.

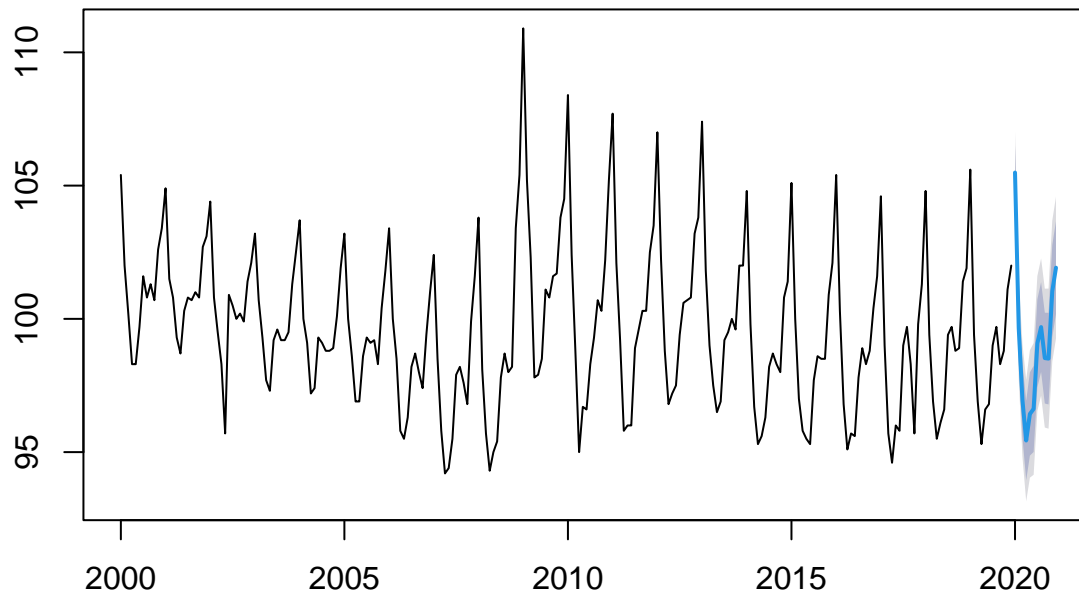
Tak wyglądają wartości z 2020, których użyjemy w predykcji lat 2022-2023:

```

fit_to_2019 <- auto.arima(ts_data)
forecast_2020 <- forecast(fit_to_2019, h=12)

```

Prognoza na 2020 w modelu ARIMA



Korzystając z funkcji `adf.test` sprawdzamy, czy spełniona jest stacjonarność modelowanych danych.

```
adf.test(ts_data_doubleARIMA)
```

```
##
## Augmented Dickey-Fuller Test
##
## data: ts_data_doubleARIMA
## Dickey-Fuller = -9.5335, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(ts_data_mean)
```

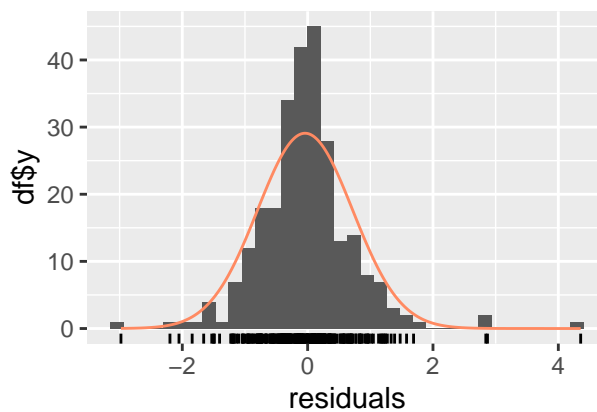
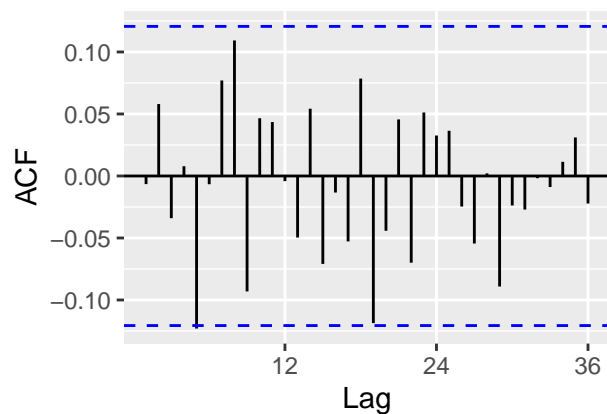
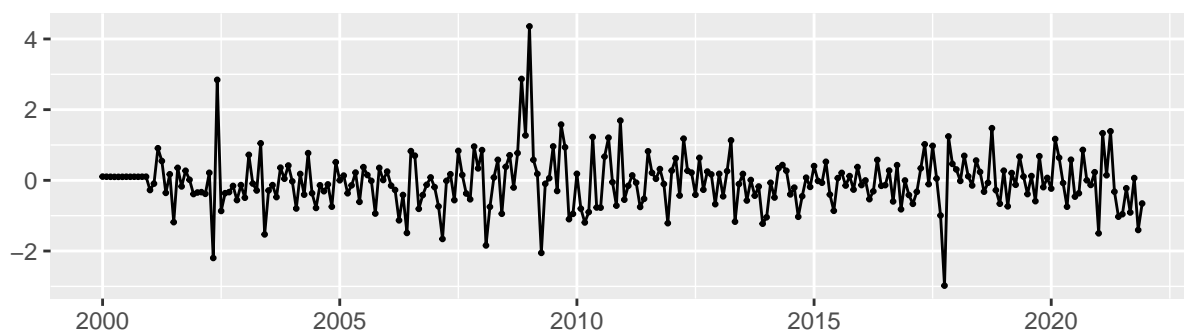
```
##
## Augmented Dickey-Fuller Test
##
## data: ts_data_mean
## Dickey-Fuller = -9.4633, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

Odrzucamy hipotezę zerową ($pvalue = 0.01$) i zakładamy stacjonarność danych. Parametr d będzie więc równy zero, ponieważ nie będzie konieczne różnicowanie.

```
fit_doubleARIMA <- auto.arima(ts_data_doubleARIMA, d=0)
forecast_values_doubleARIMA <- forecast(fit_doubleARIMA, h=24)
fit_mean <- auto.arima(ts_data_mean, d=0)
forecast_values_mean <- forecast(fit_mean, h=24)

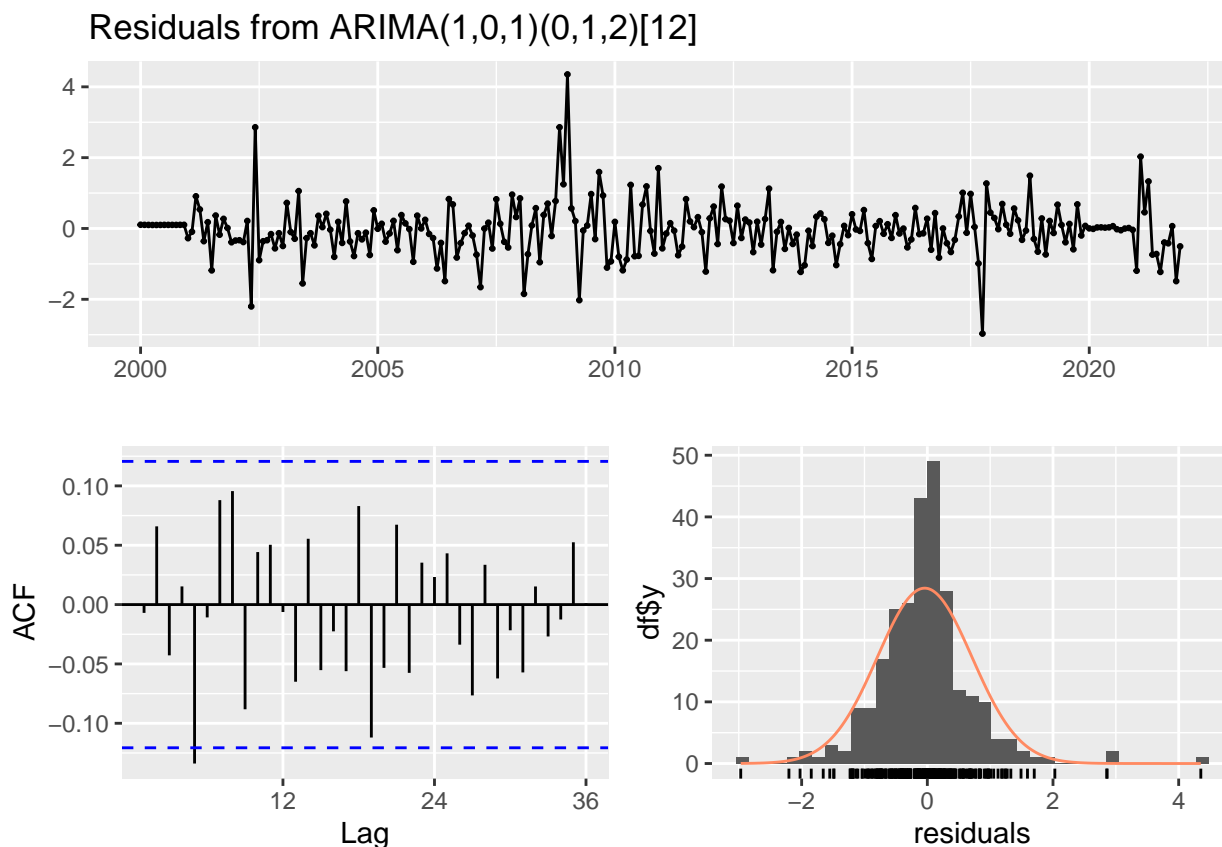
checkresiduals(fit_mean)
```

Residuals from ARIMA(1,0,1)(0,1,2)[12]



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1)(0,1,2)[12]
## Q* = 27.002, df = 20, p-value = 0.1352
##
## Model df: 4.    Total lags used: 24
```

```
checkresiduals(fit_doubleARIMA)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1)(0,1,2)[12]
## Q* = 27.762, df = 20, p-value = 0.1152
##
## Model df: 4.   Total lags used: 24
```

Pierwszą składową wykresu jest rozkład reszt w czasie, drugą jest ACF, pokazuje funkcję autokorelacji reszt. ostatnim wykresem jest histogram reszt, zbliżony do normalnego rozkładu. Funkcja `auto.arima` w obu przypadkach dopasowała parametry $p = 1$ oraz $q = 1$. Na powyższych wykresach błędów modelu widać, że obydwa przedstawione rozwiązania są podobnie dopasowane. Jednakże, błędy w 2020 są praktycznie zerowe - wynika to z faktu, że rok ten jest dopasowany przez ten sam model. Mimo to zdecydowaliśmy się przyjąć wyniki z modelu korzystającego z podwójnej predykcji ze względu na jego odmienny charakter. Testem Ljung-Boxa upewniliśmy się, że nie ma istotnych autokorelacji w resztach. ($pvalue > 0.05$).

```
forecast_values_doubleARIMA <- data.frame(
  Date = seq(as.Date("2022-01-01"), by = "month", length.out = 24),
  Point_Forecast = forecast_values_doubleARIMA$mean[1:24]
)

forecast_values_mean <- data.frame(
  Date = seq(as.Date("2022-01-01"), by = "month", length.out = 24),
  Point_Forecast = forecast_values_mean$mean[1:24]
)

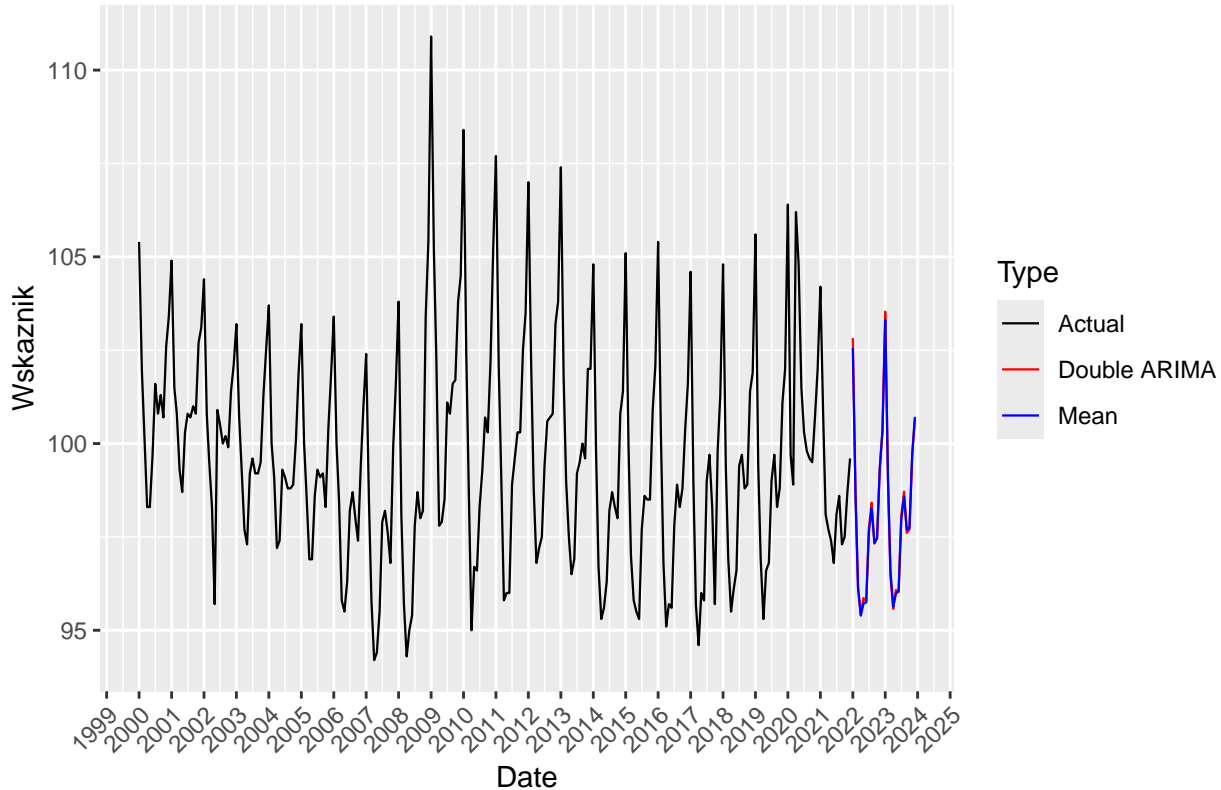
combined_df <- rbind(df_plot[, c("Date", "Wskaźnik", "Type")],
  forecast_values_doubleARIMA,
```



```
forecast_values_mean)
```

Ostatecznie, dane przewidziane przy użyciu omówionych metod modeli regresyjnych przedstawiają się następująco:

Wskaźnik bezrobocia w czasie z predykcją



```
ARIMA <- combined_df[combined_df$Type == 'Double ARIMA',]
```

2.2 Prophet Kolejną metodą przewidywania szeregu czasowego jest Prophet, przedstawiony przez Facebooka.[4] Na wyjściowym szeregu dokonujemy dekompozycji w następujący sposób:

$$y(t) = g(t) + s(t) + h(t) + e_t$$

W tym przypadku $g(t)$ jest funkcją trendu reprezentującą nieokresowe zmiany wartości szeregu czasowego, $s(t)$ jest funkcją zmian okresowych (np. miesięcznych), a $h(t)$ to efekt świąt, występujących w nieregularnych odstępach czasu. Zakładamy, że błąd e_t ma rozkład normalny. Funkcje $g(t)$, $s(t)$, $h(t)$ są estymowane przy użyciu m. in. logistycznego modelu wzrostu oraz szeregów Fouriera, ściślej opisane w [4].

Podstawiając nasze dane uzyskujemy następujące wyniki predykcji:

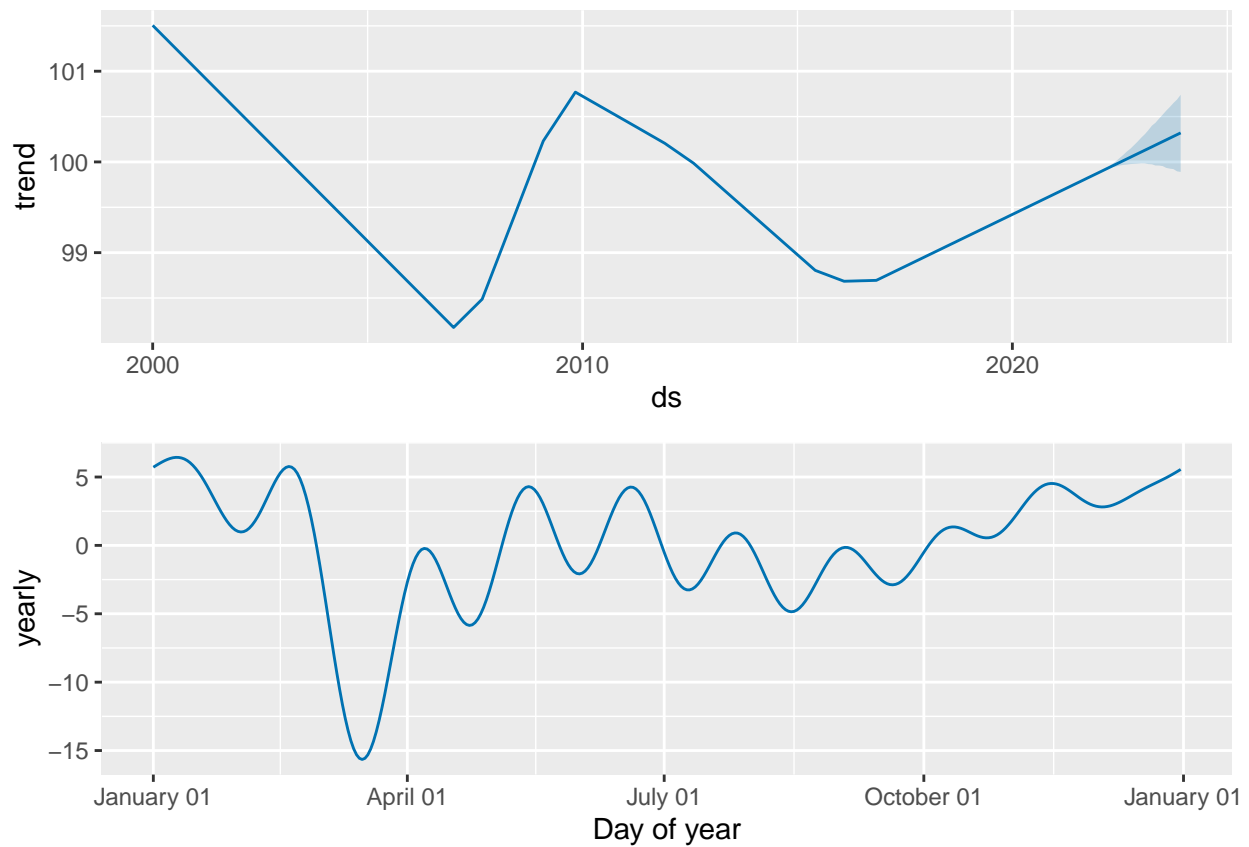
```
library(prophet)

df_prophet <- df
df_prophet$date <- as.Date(paste(df$Rok, df$M.c, "01", sep = "-"))
df_prophet <- subset(df_prophet, select = c(date, Wskaźnik))
colnames(df_prophet) <- c('ds', 'y')
model_prophet <- prophet(df_prophet)
future <- make_future_dataframe(model_prophet,
                                periods = 24,
                                freq = 'month')
```

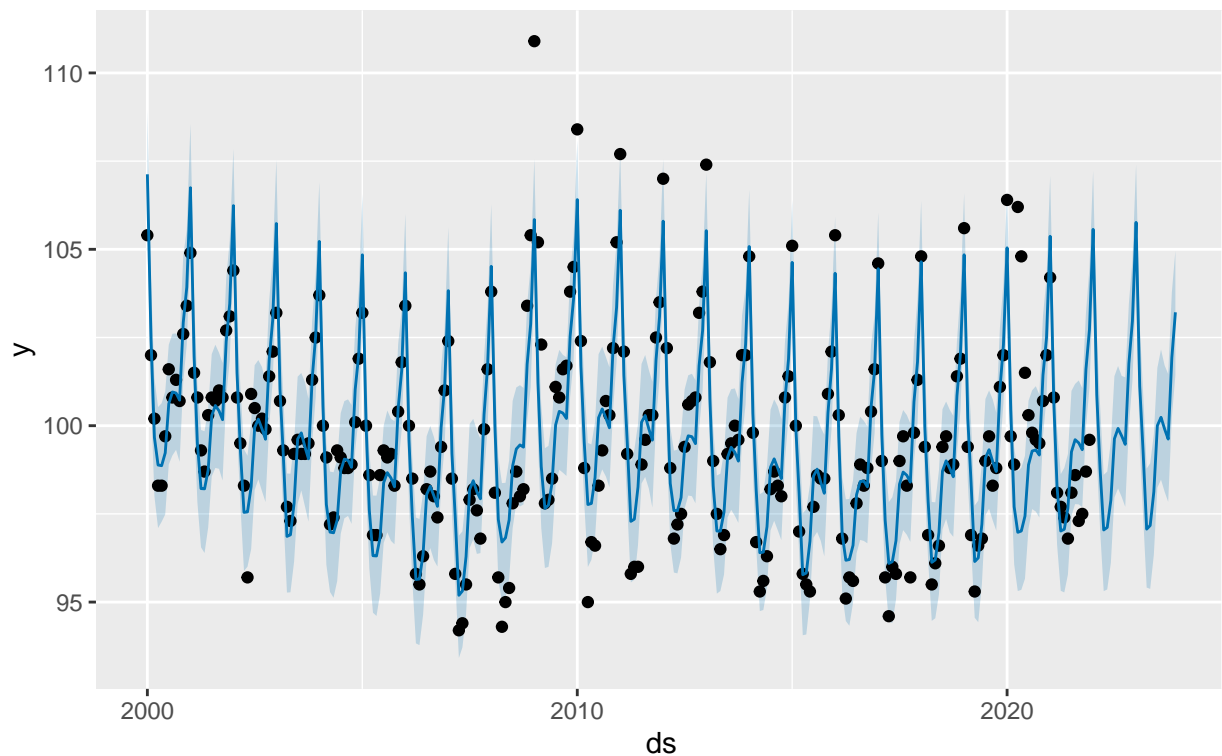
```
forecast <- predict(model_prophet, future)
forecast[c('ds', 'yhat')] %>%
  tail(n = 24)
```

```
##           ds      yhat
## 265 2022-01-01 105.56624
## 266 2022-02-01 100.88611
## 267 2022-03-01  98.78122
## 268 2022-04-01  97.04477
## 269 2022-05-01  97.11621
## 270 2022-06-01  97.89194
## 271 2022-07-01  99.63072
## 272 2022-08-01  99.92296
## 273 2022-09-01  99.70809
## 274 2022-10-01  99.47046
## 275 2022-11-01 101.76467
## 276 2022-12-01 102.97092
## 277 2023-01-01 105.76355
## 278 2023-02-01 101.12426
## 279 2023-03-01  99.31498
## 280 2023-04-01  97.06856
## 281 2023-05-01  97.17096
## 282 2023-06-01  98.12748
## 283 2023-07-01 100.00319
## 284 2023-08-01 100.23865
## 285 2023-09-01  99.90016
## 286 2023-10-01  99.62113
## 287 2023-11-01 101.93168
## 288 2023-12-01 103.21475
```

```
prophet_plot_components(model_prophet, forecast)
```



```
plot(model_prophet, forecast)
```



Na pierwszym wykresie widzimy krzywą trendu wyznaczoną przez model, łącznie z przewidzianymi ostatnimi

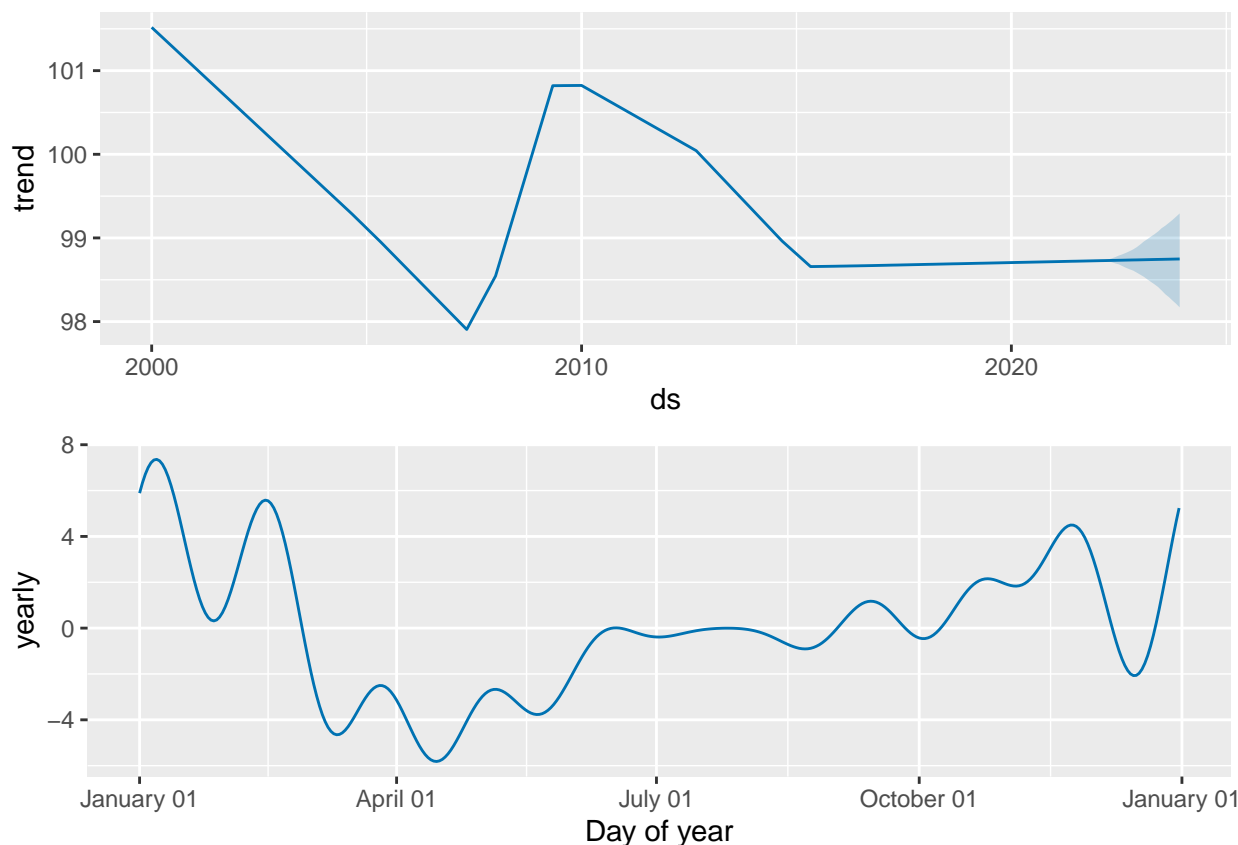
latami wraz z przedziałem ufności (niebieskie pole na końcu krzywej), na drugim rysunku - uśrednione, ogólne roczne zmiany wskaźnika. Ostatni wykres przedstawia dodatkowo porównanie rzeczywistych danych (czarne punkty) z tymi estymowanymi przez Prophet (niebieska linia, wraz z przedziałem ufności).

Zbadajmy teraz jakość predykcji, kiedy pominiemy rok 2020 w rozważaniach:

```
forecast2[c('ds', 'yhat')] %>%  
  tail(n = 24)
```

```
##           ds           yhat  
## 253 2022-01-01 104.49232  
## 254 2022-02-01  99.89575  
## 255 2022-03-01  97.55975  
## 256 2022-04-01  95.63991  
## 257 2022-05-01  95.71130  
## 258 2022-06-01  96.54372  
## 259 2022-07-01  98.34756  
## 260 2022-08-01  98.69376  
## 261 2022-09-01  98.51870  
## 262 2022-10-01  98.31487  
## 263 2022-11-01 100.64201  
## 264 2022-12-01 101.82848  
## 265 2023-01-01 104.37688  
## 266 2023-02-01  99.82513  
## 267 2023-03-01  97.72609  
## 268 2023-04-01  95.70154  
## 269 2023-05-01  95.68539  
## 270 2023-06-01  96.49767  
## 271 2023-07-01  98.36052  
## 272 2023-08-01  98.70824  
## 273 2023-09-01  98.49656  
## 274 2023-10-01  98.33756  
## 275 2023-11-01 100.66390  
## 276 2023-12-01 101.93179
```

```
prophet_plot_components(model_prophet2, forecast2)
```



Obserwujemy, że po wyrzuceniu 2020, linia trendu zdecydowanie się wypłaszcza na koniec badanego okresu. Przez to, że model bierze pod uwagę współczynnik świąt, czyli nieregularnych skoków badanej zmiennej, w tym przypadku warto zostawić dane z 2020, zatem ostatecznie bierzemy pierwotną wersję prognozy.

```
PROPHET <- forecast['yhat'] %>%
  tail(n = 24)
```

2.3 TBATS Metoda TBATS należy do popularnej grupy modeli statystycznych - modeli wygładzania wykładniczego. Do tej samej rodziny należy także STL, którą używaliście do dekompozycji naszych danych. Najczęściej używany model sezonowy przedstawia się w następujący sposób: [5]

$$\begin{aligned}
 y_t &= l_{t-1} + b_{t-1} + s_t + d_t \\
 l_t &= l_{t-1} + b_{t-1} + \alpha d_t \\
 b_t &= b_{t-1} + \beta d_t \\
 s_t &= s_{t-m} + \gamma d_t
 \end{aligned}$$

gdzie m to okres cykli sezonowych, d_t reprezentuje losowy szum w danych, l_t , b_t , s_t to komponenty odpowiednio: poziomu, trendu i sezonowości szeregu czasowego. Wartości α , β i γ są tak zwanymi parametrami wygładzającymi, a l_0 , b_0 , $\{s_{1-m}, \dots, s_0\}$ to zmienne wyjściowe. W tym modelu estymuje się właśnie zmienne wyjściowe oraz parametry. Warto dodać, że można estymować 2 składniki sezonowości $s_t^{(1)}$, $s_t^{(2)}$ wraz z parametrami γ_1 i γ_2 , jednak w naszym przypadku bierzemy tylko jeden składnik - miesięczny.

Następnie dokonujemy modyfikacji tego modelu, przy uwzględnieniu transformacji Box-Cox, błędów modelu ARMA oraz T wzorców sezonowości. Na koniec, składniki sezonowości przedstawiamy jako ich trygonometryczną reprezentację opartą na szeregach Fouriera. Ze wszystkich składowych modelu powstała jego nazwa: T - trigonometrical, B - Box-Cox transformation, A - ARMA errors, TS - T seasonal patterns.

W celu ewaluacji jakości prognozy, będziemy porównywać ostatnie dane 2 lata z przewidzianymi wartościami na następne 2 lata, korzystając z dwóch metryk:

1. średni bezwzględny błąd procentowy - MAPE (Mean Absolute Percentage Error):

$$\frac{1}{n} \sum_{t=1}^n \left| \frac{Y_t - P_t}{Y_t} \right| * 100\%$$

2. pierwiastek błędu średniokwadratowego - RMSE (Root Mean Squared Error):

$$\sqrt{\frac{\sum_{t=1}^n (Y_t - P_t)^2}{n}}$$

gdzie Y_t - rzeczywista wartość, P_t - prognozowana wartość.

Po podstawieniu naszych danych otrzymujemy następujące wyniki:

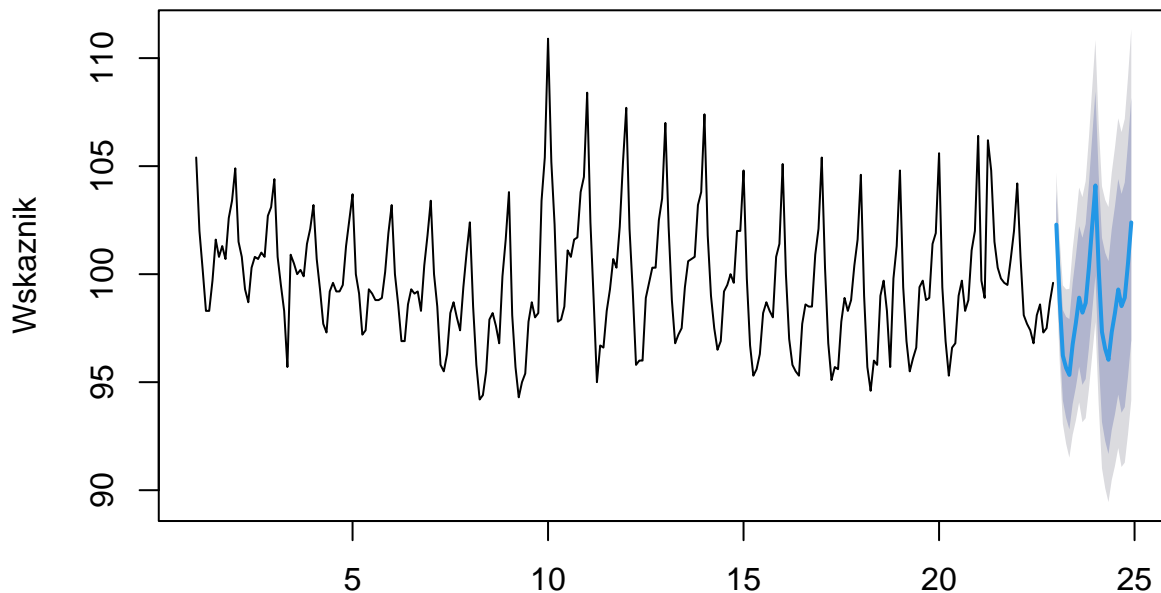
```
ts_tbats <- msts(df$Wskaznik, seasonal.periods = 12)

model.ts_tbats <- tbats(ts_tbats)

model.ts_tbats.forecast <- forecast(model.ts_tbats, h = 24)

plot(model.ts_tbats.forecast, main = 'Prognoza TBATS',
      ylab = 'Wskaznik')
```

Prognoza TBATS



```
model.ts_tbats.forecast$mean
```

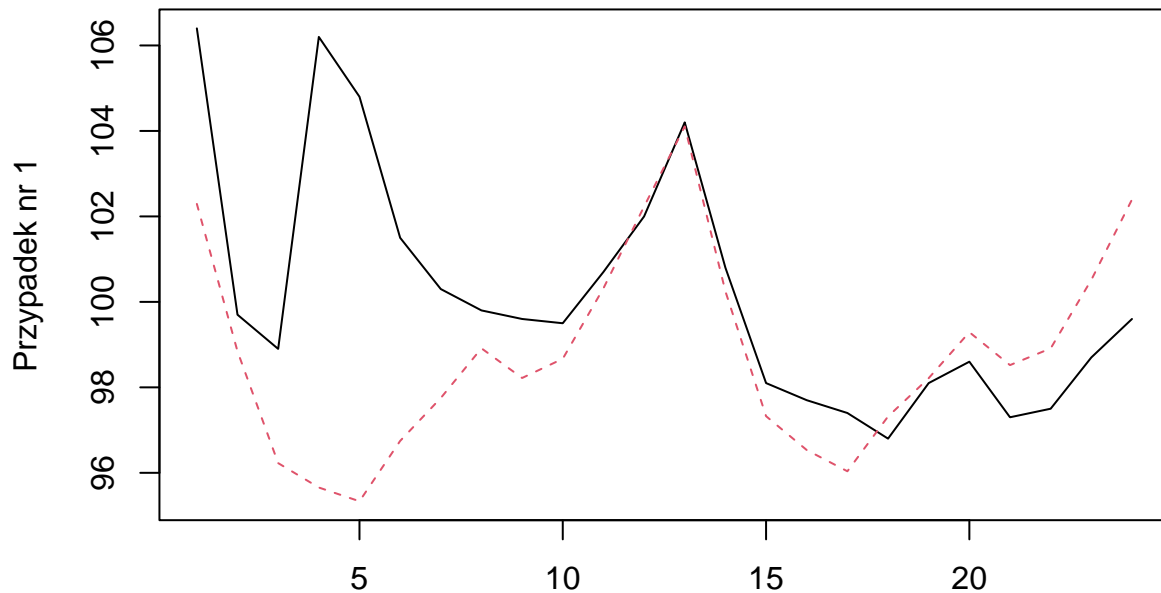
	Jan	Feb	Mar	Apr	May	Jun	Jul
## 23	102.29247	98.84214	96.22711	95.65658	95.33452	96.74659	97.75088
## 24	104.10502	100.24764	97.32554	96.53332	96.03627	97.31860	98.21517
	Aug	Sep	Oct	Nov	Dec		
## 23	98.91372	98.21906	98.66880	100.31739	102.23146		
## 24	99.29118	98.52024	98.91193	100.51604	102.39415		

W wykresie prognozy, niebieska linia oznacza przewidziane dane, wraz z przedziałami ufności na poziomie istotności 0.2 oraz 0.05. Przewidziane wartości zostały wypisane w tabeli pod nim.

```
df.test <- tail(df$Wskaznik, n = 12 * 2)

ts_tsbats.predict <- predict(model.ts_tsbats.forecast, df.test)

ts_tsbats.test_vs_predicted <- data.frame(df.test, model.ts_tsbats.forecast$mean)
matplot(ts_tsbats.test_vs_predicted, type = 'l', lty = 1:2, col = 1:2, ylab = 'Przypadek nr 1')
```



Na tym rysunku widzimy porównanie wartości z ostatnich dwóch lat (czarna linia) z przewidzianymi wartościami na następne 2 lata (czerwona przerywana linia). Obserwujemy tutaj zauważalne niedopasowanie krzywych na początku okresu - jest to następstwo wzięcia pod uwagę odstających wartości z roku 2020.

```
MAPE.error(ts_tsbats.test_vs_predicted$df.test,
            ts_tsbats.test_vs_predicted$model.ts_tsbats.forecast.mean)
```

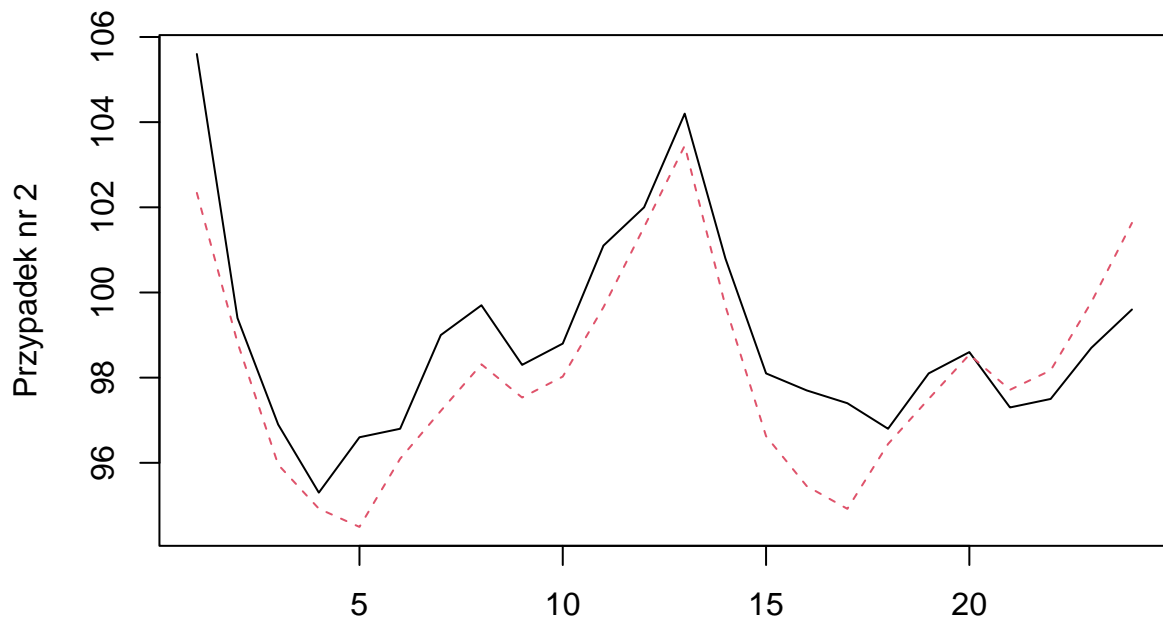
```
## [1] 2.085391
```

```
RMSE.error(ts_tsbats.test_vs_predicted$df.test,
            ts_tsbats.test_vs_predicted$model.ts_tsbats.forecast.mean)
```

```
## [1] 6.855342
```

Ostatecznie, liczymy wartości błędów MAPE i RMSE - są one zawyżone z wyżej wymienionego powodu.

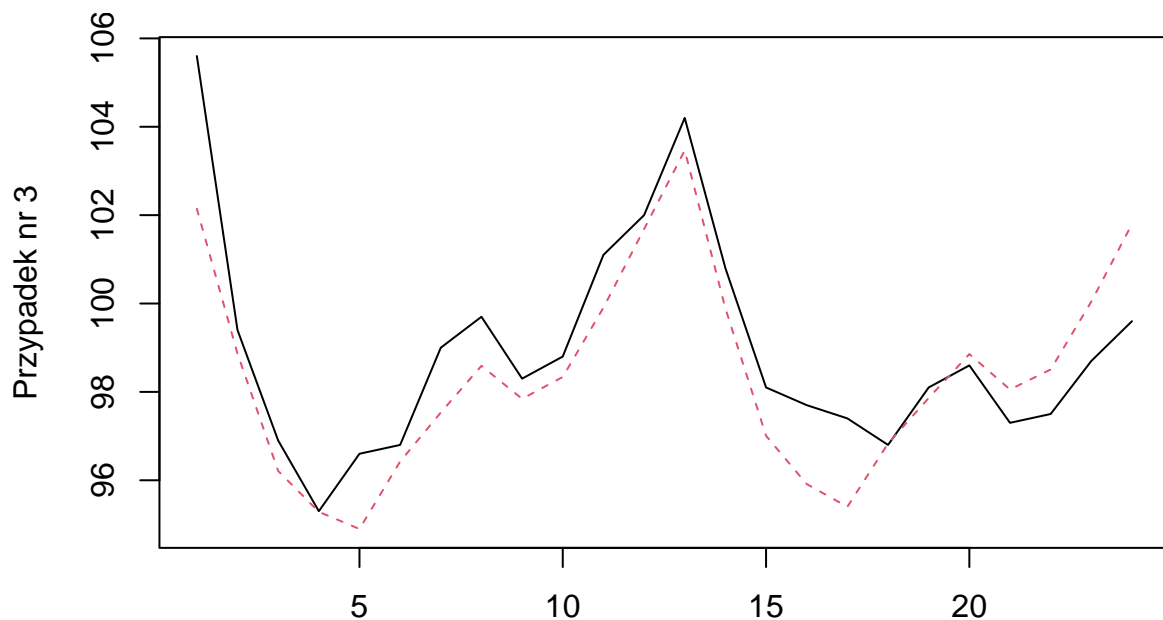
Aby zmniejszyć wartości błędów, dokonujemy analizy bez 2020:



MAPE: 1.168697

RMSE: 3.977294

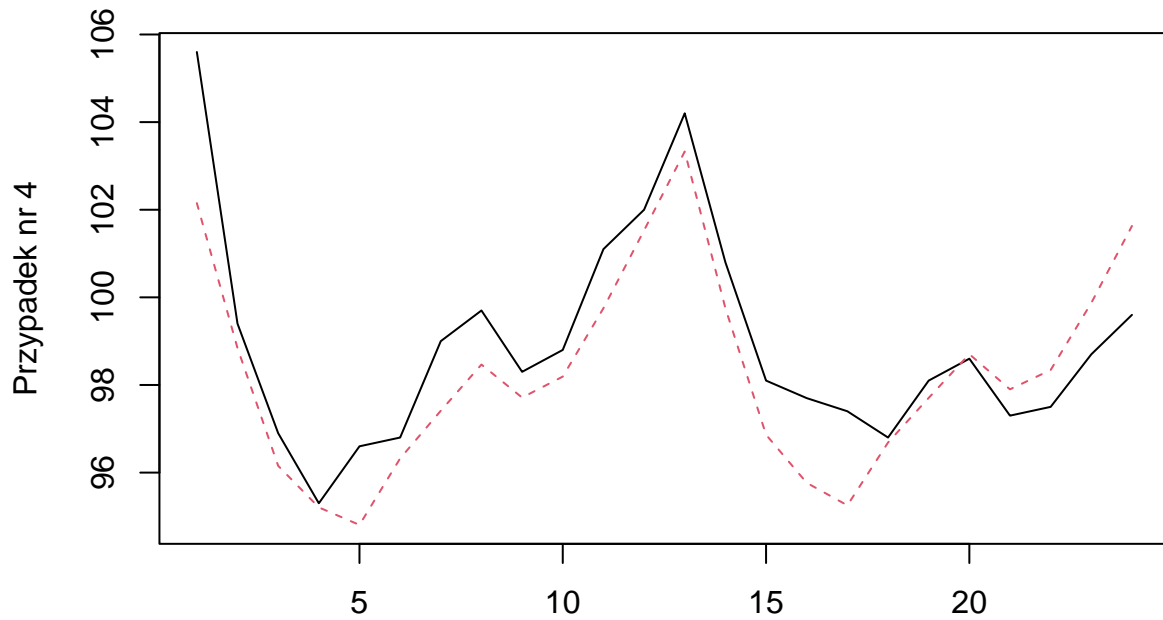
W przypadku wyrzucenia danych z 2020, zgodnie z oczekiwaniami, widzimy znaczny spadek błędów MAPE i RMSE. Zwróćmy uwagę, że do ewaluacji prognozy wzięliśmy lata 2019 i 2021 zamiast 2020-2021. Sprawdzimy jeszcze predykcję dla przypadków, gdy zamiast wartości z tego roku podstawimy średnią globalną wartość wskaźnika:



MAPE: 1.009071

RMSE: 2.643147

Po raz kolejny, wartości błędów zmniejszyły się. Na koniec zbadajmy zachowanie błędów, gdy zamiast 2020 weźmiemy średni wskaźnik z ostatnich 5 lat:



MAPE: 1.06453

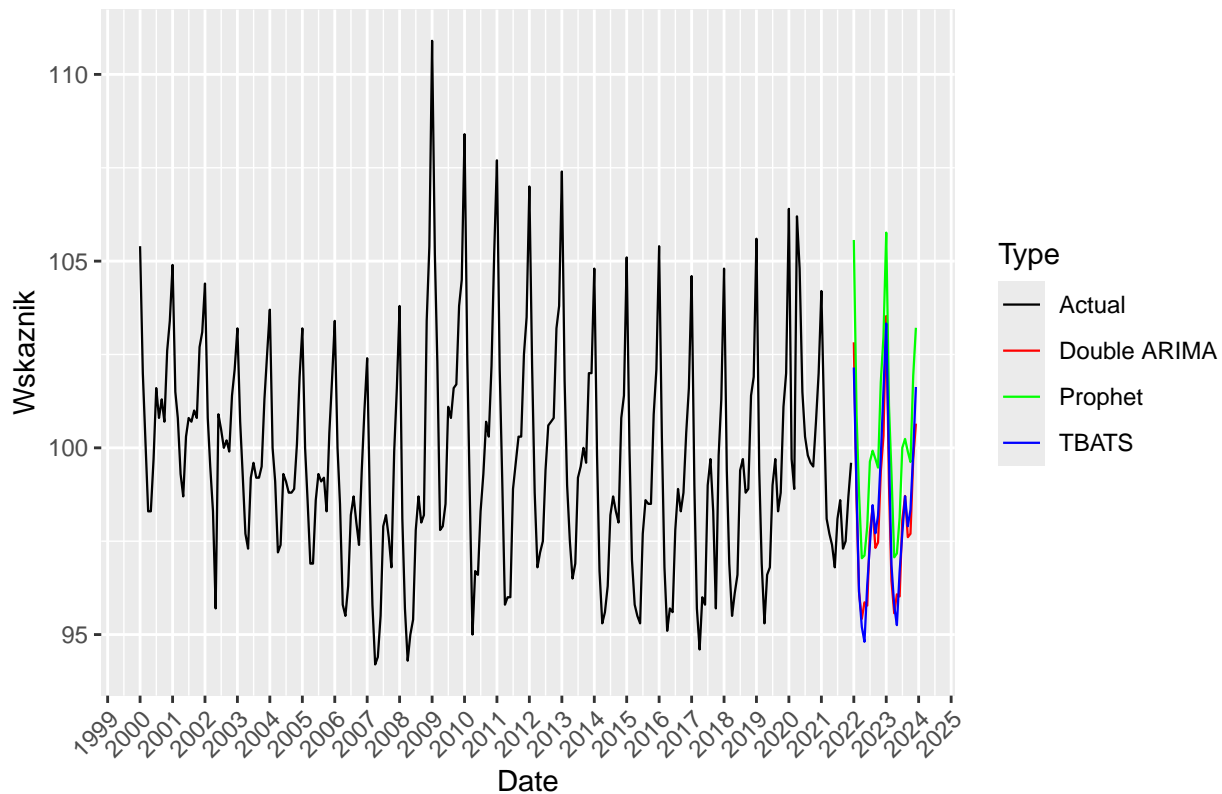
RMSE: 3.255959

Najmniejsze wartości błędów otrzymujemy w przypadku podstawienia globalnej średniej za wartości z odstającego roku. Decydujemy jednak uwzględnić ostatni przypadek w ostatecznej predykcji, gdyż nie chcemy brać takich wyników, które będą zbyt zbliżone do wartości z ostatnich dwóch lat, ponieważ po raz kolejny, trend byłby zbyt spłaszczony na końcu okresu. Ostatecznie:

```
TBATS <- model.ts_tbats4.forecast$mean %>% as.data.frame()
```

2.4 Ostateczna predykcja Wszystkie wyniki wygenerowane przez omówione modele predykcji przedstawiają się w następujący sposób:

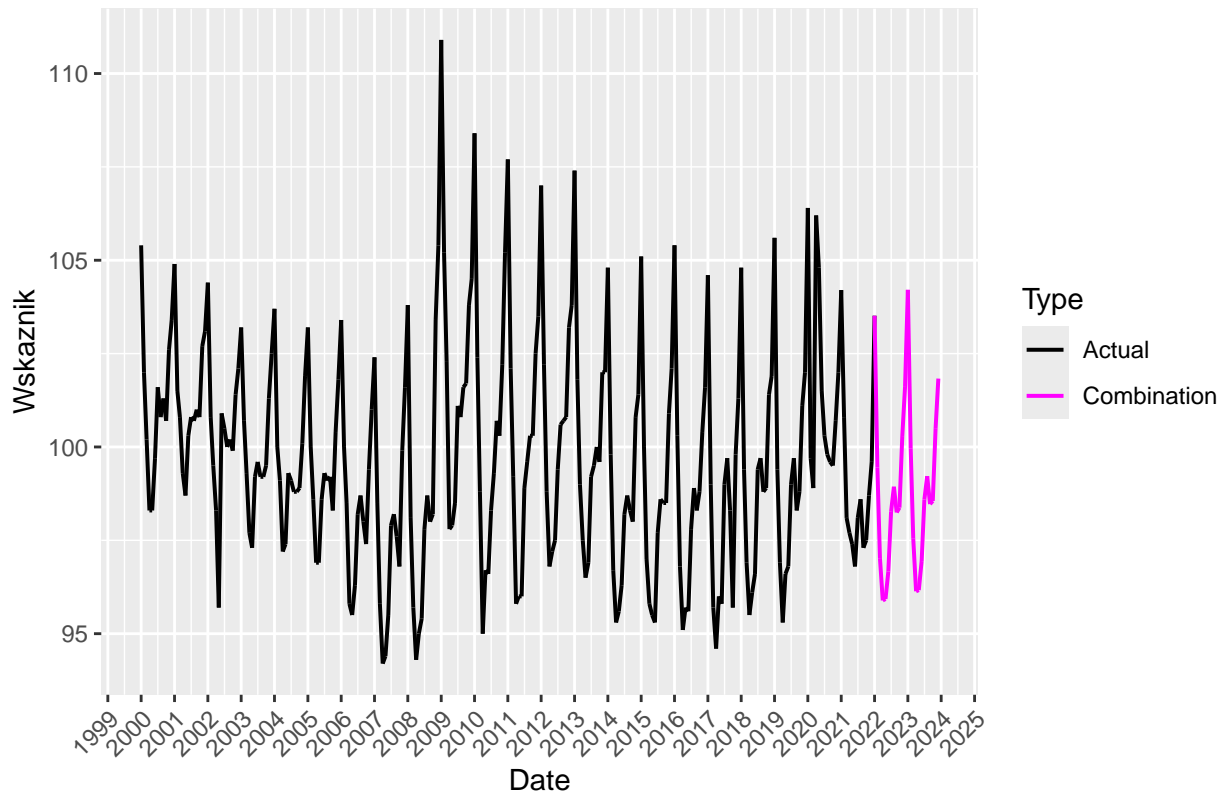
Wskaźnik bezrobocia w czasie z predykcja



##	Date	ARIMA	Prophet	TBATS
## 1	2022-01-01	102.82608	105.56624	102.15098
## 2	2022-02-01	98.62021	100.88611	98.84227
## 3	2022-03-01	96.13074	98.78122	96.15523
## 4	2022-04-01	95.42150	97.04477	95.20574
## 5	2022-05-01	95.86357	97.11621	94.80567
## 6	2022-06-01	95.77652	97.89194	96.32381
## 7	2022-07-01	97.74766	99.63072	97.41151
## 8	2022-08-01	98.41741	99.92296	98.46648
## 9	2022-09-01	97.32115	99.70809	97.71175
## 10	2022-10-01	97.46430	99.47046	98.19079
## 11	2022-11-01	99.36107	101.76467	99.75118
## 12	2022-12-01	100.32231	102.97092	101.52969
## 13	2023-01-01	103.52965	105.76355	103.32886
## 14	2023-02-01	99.01594	101.12426	99.75306
## 15	2023-03-01	96.44852	99.31498	96.86344
## 16	2023-04-01	95.57216	97.06856	95.76631
## 17	2023-05-01	96.07756	97.17096	95.25199
## 18	2023-06-01	96.01987	98.12748	96.68639
## 19	2023-07-01	98.06441	100.00319	97.70474
## 20	2023-08-01	98.71224	100.23865	98.70353
## 21	2023-09-01	97.60887	99.90016	97.89990
## 22	2023-10-01	97.69950	99.62113	98.34201
## 23	2023-11-01	99.70443	101.93168	99.87406
## 24	2023-12-01	100.64744	103.21475	101.62973

Ostateczne wyniki otrzymujemy biorąc średnią dla każdego miesiąca ze wszystkich metod:

Wskaźnik bezrobocia w czasie z predykcją



3. Przewidywanie

Wyznaczona przez nas predykcja, łącząca w sobie modele ARIMA, Prophet oraz TBATS jako średnia arytmetyczna wszystkich prognoz prezentuje się następująco:

##	Date	Wskaźnik
## 1	2022-01	103.51443
## 2	2022-02	99.44953
## 3	2022-03	97.02240
## 4	2022-04	95.89067
## 5	2022-05	95.92849
## 6	2022-06	96.66409
## 7	2022-07	98.26330
## 8	2022-08	98.93562
## 9	2022-09	98.24700
## 10	2022-10	98.37518
## 11	2022-11	100.29231
## 12	2022-12	101.60764
## 13	2023-01	104.20735
## 14	2023-02	99.96442
## 15	2023-03	97.54232
## 16	2023-04	96.13568
## 17	2023-05	96.16684
## 18	2023-06	96.94458
## 19	2023-07	98.59078
## 20	2023-08	99.21814

```
## 21 2023-09 98.46964
## 22 2023-10 98.55421
## 23 2023-11 100.50339
## 24 2023-12 101.83064
```

4. Podsumowanie

Zaproponowaliśmy w naszym badaniu metodę prognozowania opartą na uśrednieniu rezultatów kilku metod statystycznych w celu zwiększenia ostatecznej efektywności. Wykorzystane modele są dedykowane do szeregów czasowych z widoczną sezonowością, aby sprawdzały się w celach analizy i predykcji wartości wskaźnika bezrobocia. Ostateczna prognoza wydaje się dokładnie odzwierciedlać trend i sezonowość danych z przeszłości.

Literatura

1. Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. J. (1990). STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1), 3–33.
2. Hersh, A., and T. B. Newman. “Linear Regression.” *AAP Grand Rounds* 25, no. 6 (June 1, 2011): 68.
3. Box G.E.P., Jenkins G.M., Reinsel G.C., Ljung G.M., (2015), *Time Series Analysis: Forecasting and Control*. Fifth Edition, John Wiley & Sons
4. Taylor SJ, Letham B. *Forecasting at Scale*. The American Statistician. 2018
5. De Livera A.M., Hyndman R.J., Snyder R.D., Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association* 2011, 106, 1513–1527.
6. <https://github.com/cure-lab/LTSF-Linear>