

Временные ряды

Юрченков Иван Александрович

МИРЭА - Российский технологический университет

Кафедра Прикладной математики

План доклада

1. Временные ряды. Задачи анализа временных рядов
2. Стационарность и нестационарность
3. Методы долгосрочного и краткосрочного прогнозирования.
Динамические модели и механизм авторегрессии
4. Методы разложения временного ряда на трендовую и сезонную
составляющие
5. Архитектуры нейронных сетей для построения краткосрочного
прогноза временного ряда
6. Ансамблирование моделей прогнозирования временных рядов

Временные ряды. Задачи анализа временных рядов

Что такое временной ряд

Последовательность вещественных значений

$y \in R^{N+1}$ измеренных через одинаковые промежутки времени $\Delta t \in R$:

$$y_i = y(t_i),$$

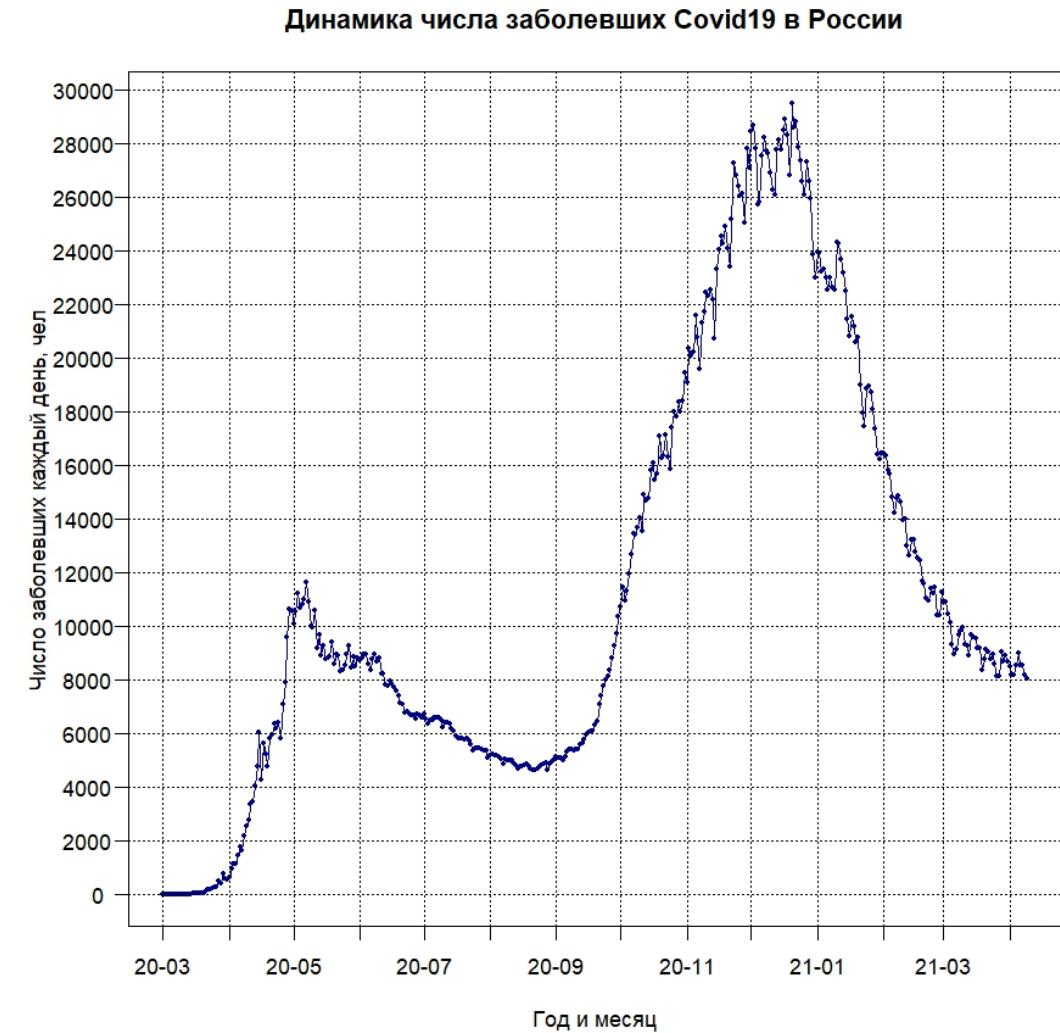
$$t_i = t_0 + i * \Delta t,$$

$$i = 0, 1, \dots, N;$$

где t_i - время замеров,

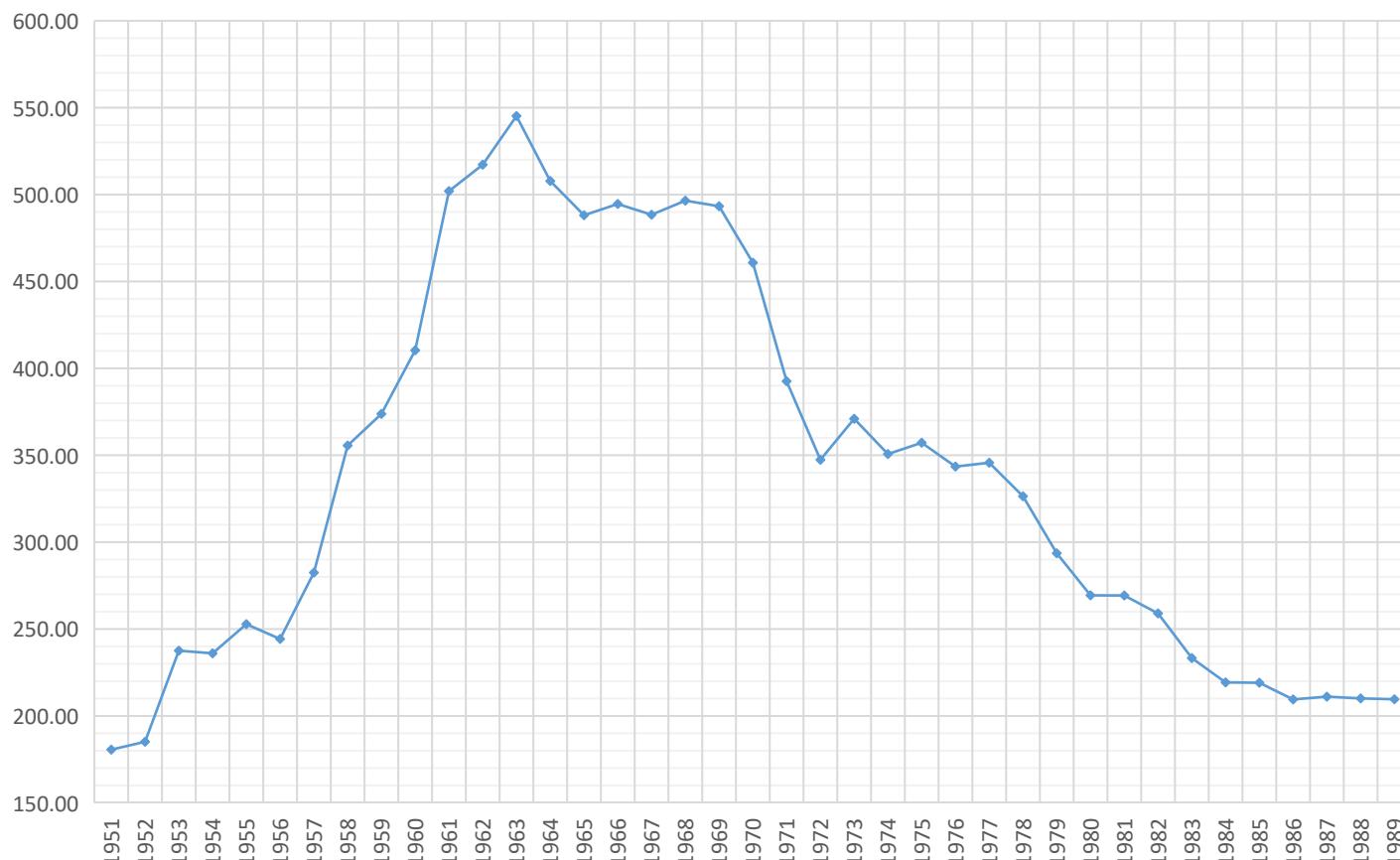
i - отсчёты по времени,

t_0 - характерное время начала процесса



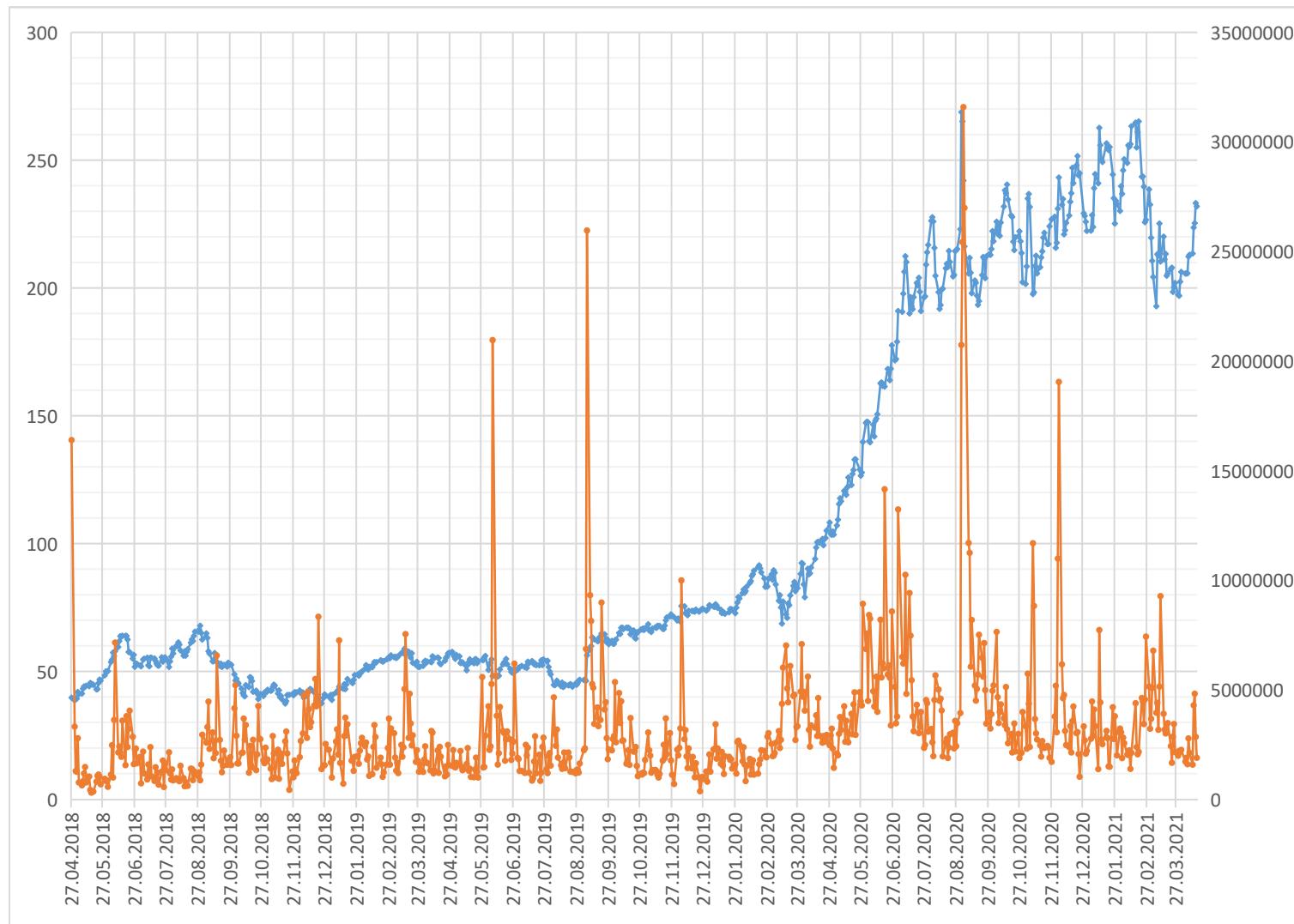
Одномерный временной ряд

Запасы нефти на месторождении Северного Кавказа,
тыс. тонн



Годы	Запасы нефти, тыс. тонн
1951	180,61
1952	185,14
1953	237,57
1954	236,04
1955	252,83
1956	244,30
1957	282,48
1958	355,60
1959	373,80
1960	410,40
1961	502,00
1962	517,20
1963	545,20
1964	507,80
...	...
1989	209,60

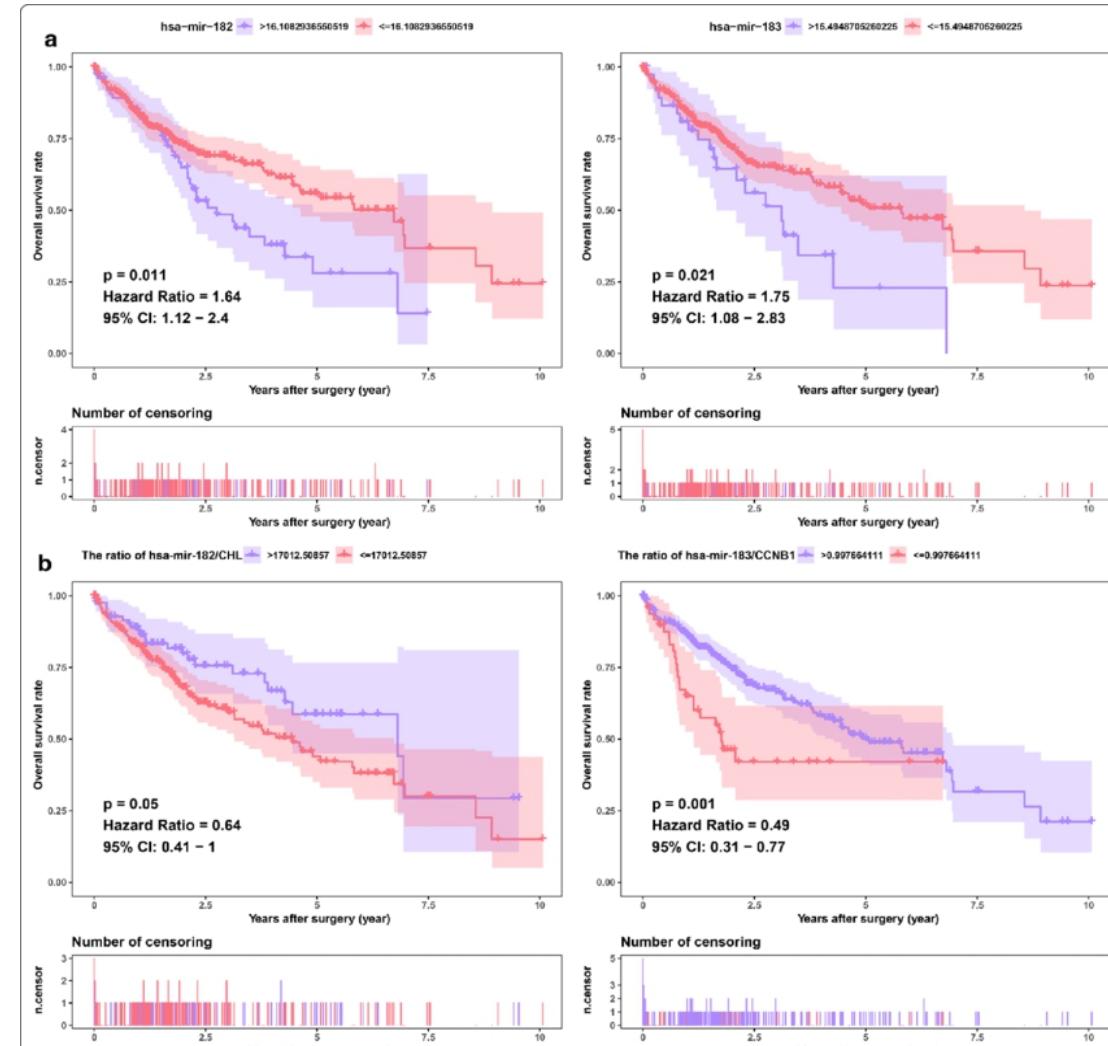
Многомерный временной ряд (неравномерный)



Даты	Цена закрытия	Объём торгов
27.04.2018	39,73	16392400
30.04.2018	38,630001	3312100
01.05.2018	39,450001	1285600
02.05.2018	39,32	1251300
03.05.2018	41,959999	2783600
04.05.2018	41,09	763500
07.05.2018	41,310001	630800
08.05.2018	43,220001	821200
09.05.2018	44,099998	1178100
10.05.2018	44,009998	1467900
11.05.2018	44,369999	761900
14.05.2018	44,189999	1031900
15.05.2018	45,459999	423000
16.05.2018	45,23	299000
17.05.2018	44,900002	339400
...
16.04.2021	231,929993	1888400

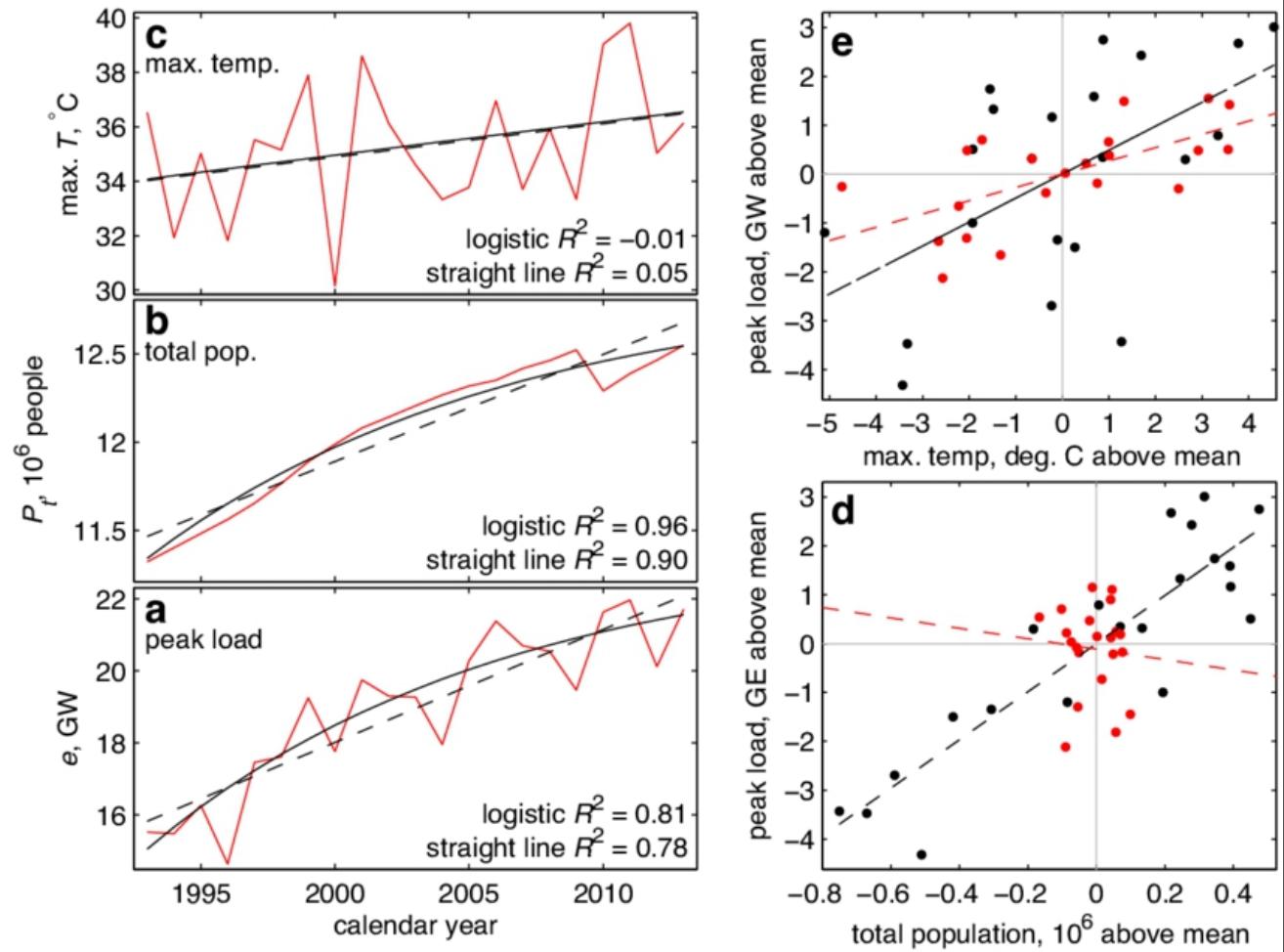
Задачи анализа временных рядов

- Идентификация (установление свойств)
- Прогнозирование (экстраполяция)
- Поиск аномалий
- Классификация
- Кластеризация

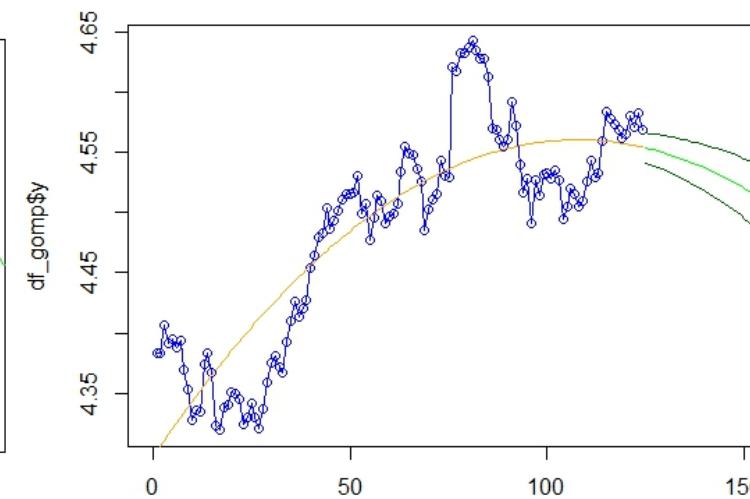
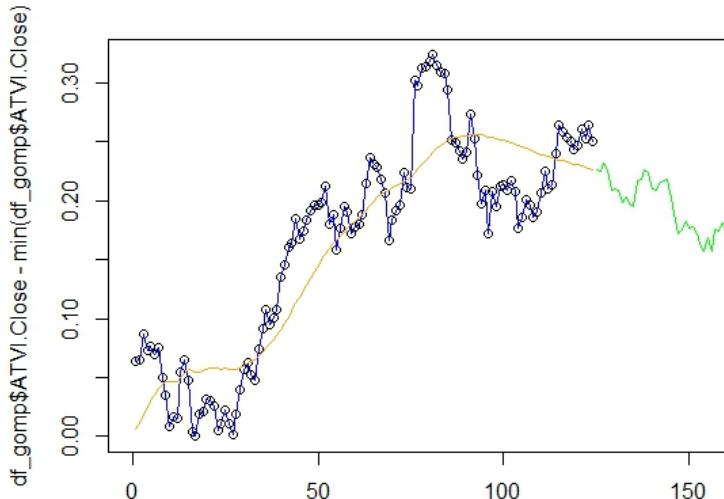
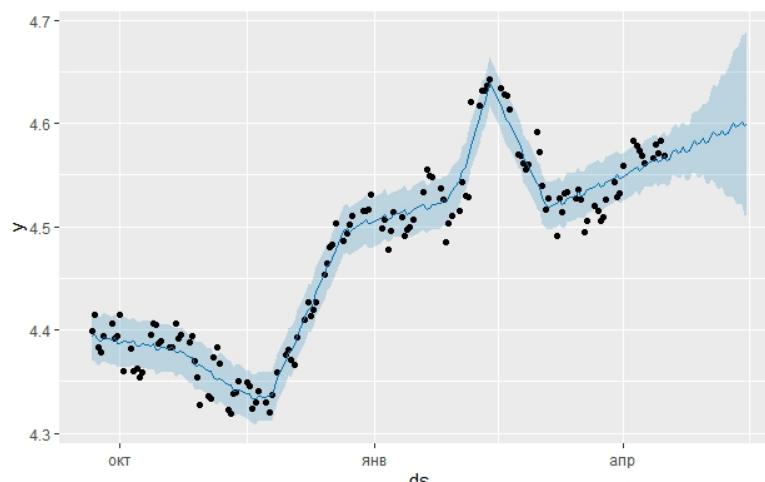
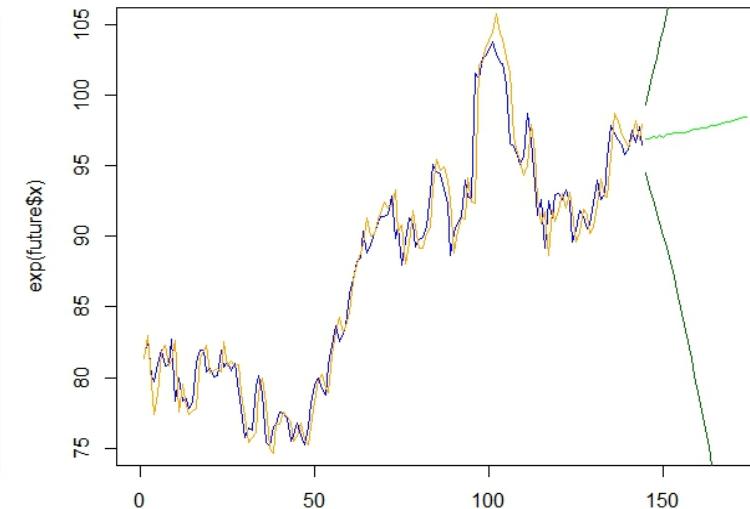
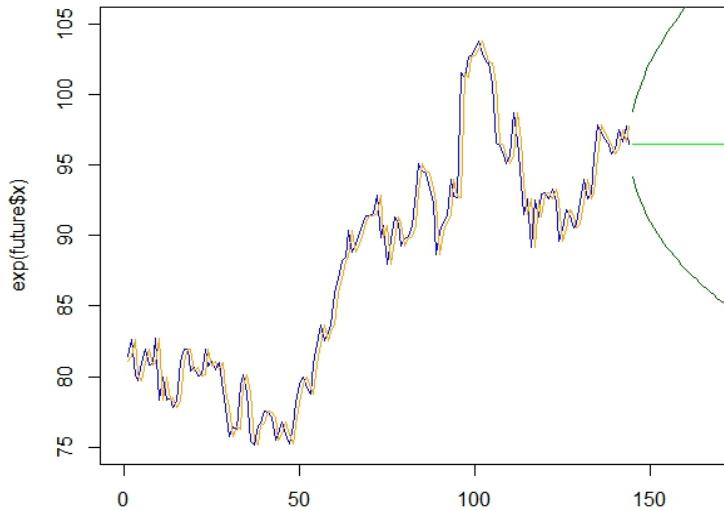
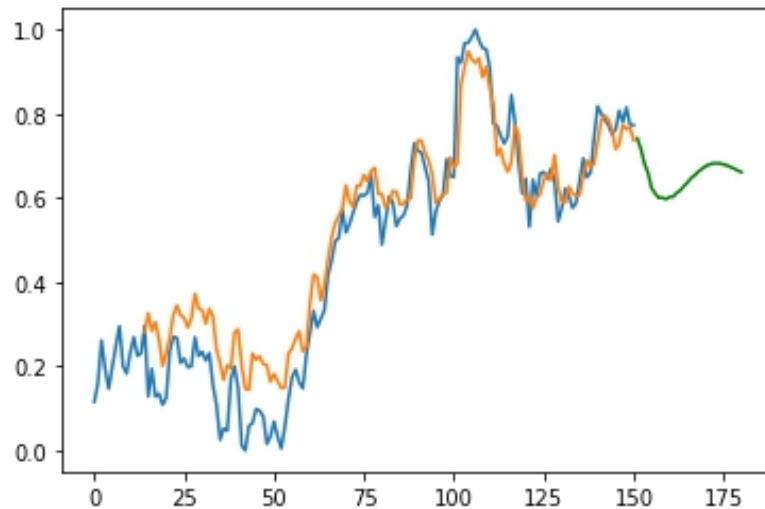


Идентификация

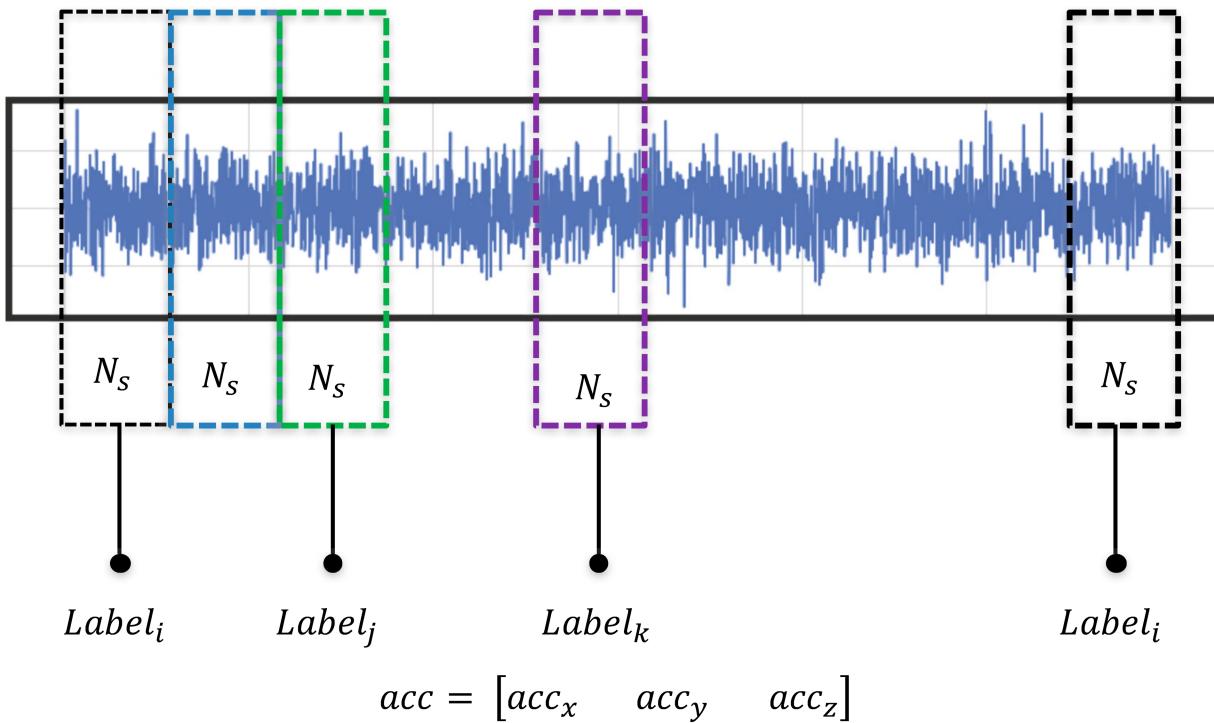
- Установление статистических свойств ряда
- Определение сезонности
- Установление математической модели процесса



Прогнозирование



Классификация



<https://blog.floydhub.com/reading-minds-with-deep-learning/>

Deep Learning for Time Series Classification

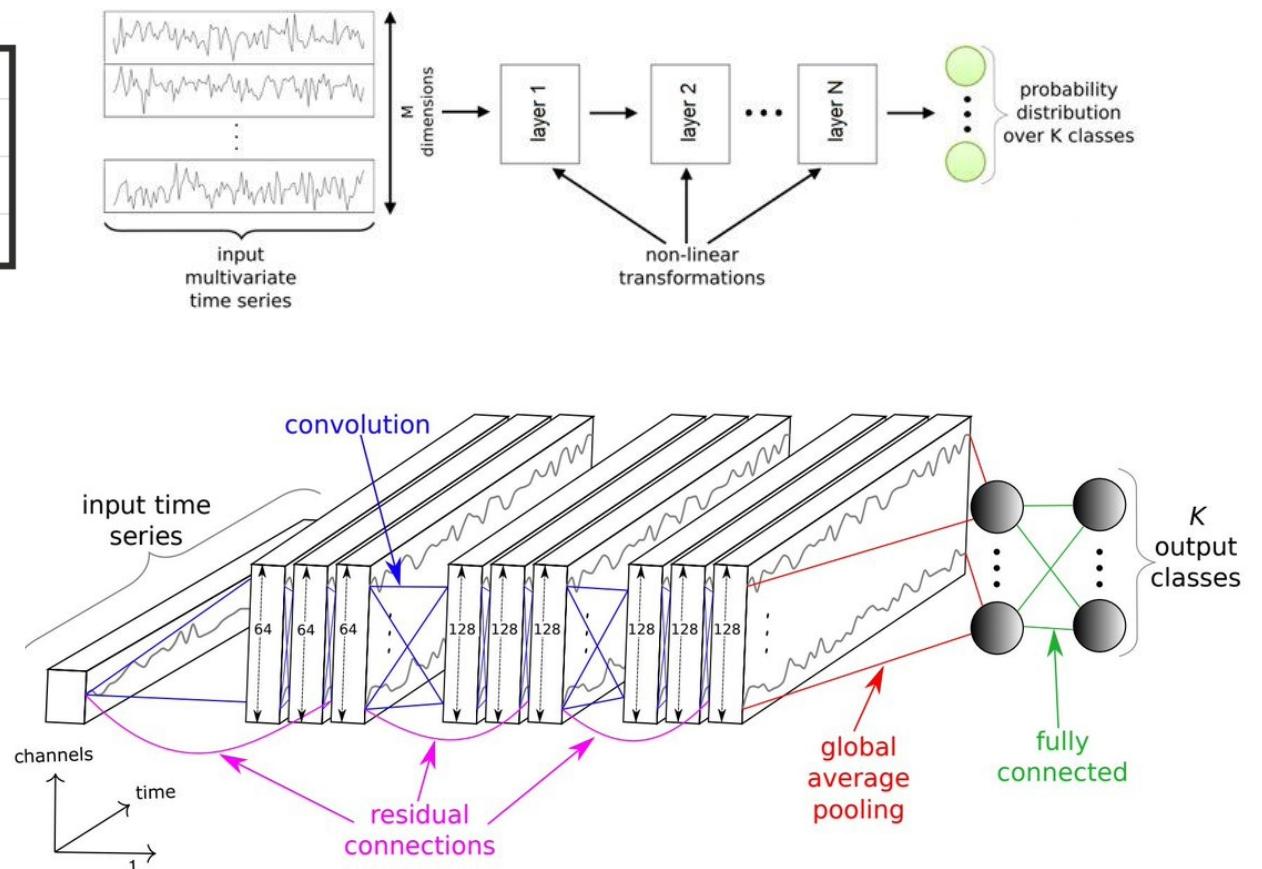
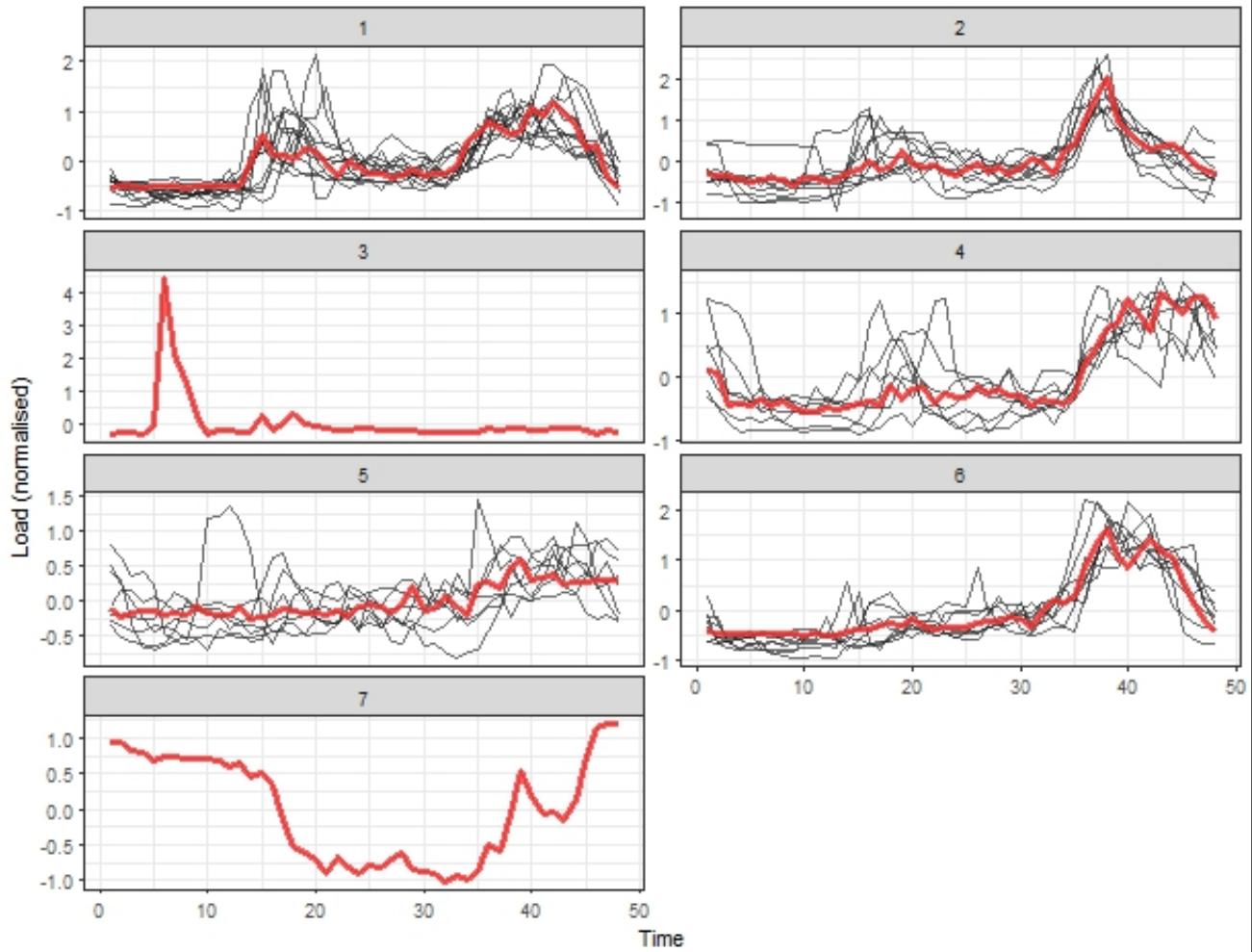
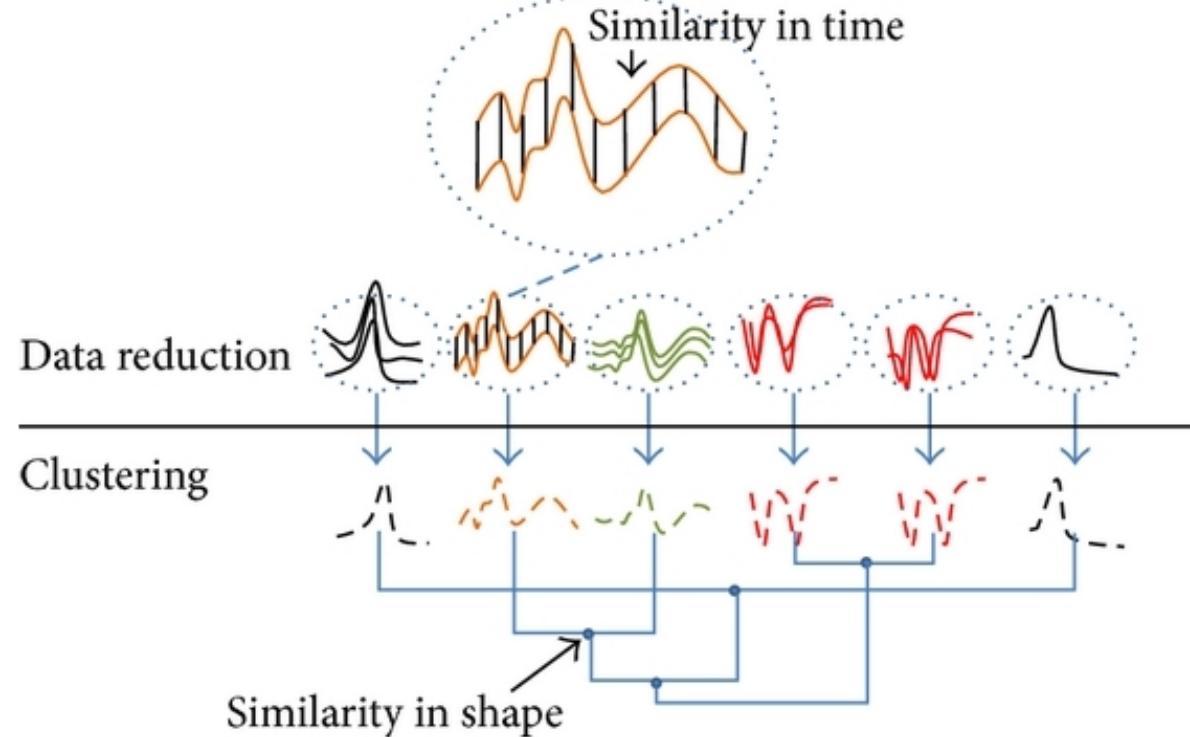


Fig. 3: The deep Residual Network (ResNet) architecture for Time Series Classification (TSC).

Кластеризация



Статистические свойства временных рядов

Главная особенность анализа временных рядов

- В классических задачах анализа данных и машинного обучения предполагается независимость наблюдений подвыборки
- При прогнозировании временных рядов, как и в задачах моделирования языка, мы надеемся на то, что значения ряда в прошлом содержат содержательную информацию о поведении ряда в будущем с сохранением тенденции на определённом интервале

Данная особенность при обработке и анализе временных рядов позволяет нам не только строить прогнозы для рядов значений во времени, но и решать другие задачи обработки временных рядов

Из чего состоят временные ряды

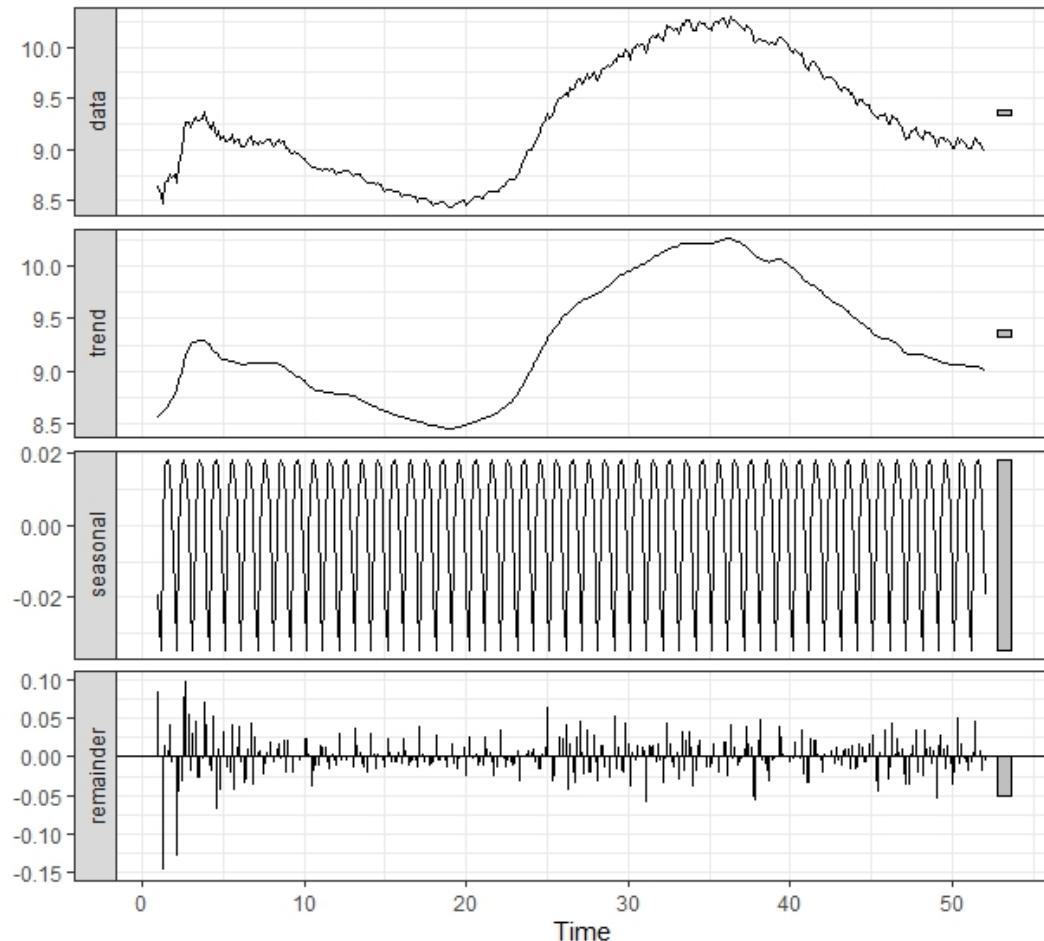
$$y(t_i) = f(T, S, C, E, t_i)$$

Тренд $T(t)$ - плавное изменение уровня ряда

Сезонность $S(t)$ - периодические изменения уровня ряда

Цикличность $C(t)$ - изменения уровня ряда с переменным периодом [часто встречается в экономических, экологических, физических процессах]

Ошибка $E(t)$ - непрогнозируемая случайная компонента ряда



Стационарные временные ряды

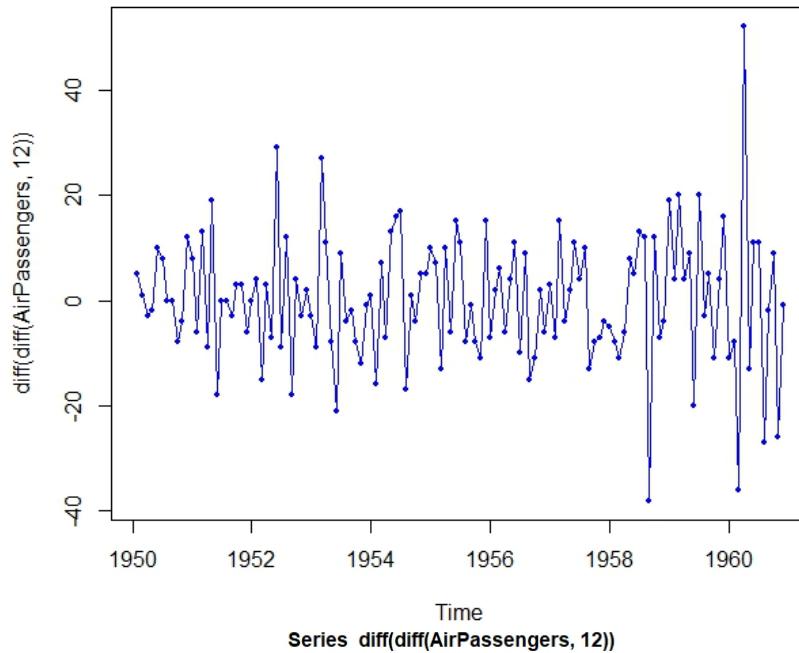
Если рассматривать временные ряды как случайные процессы, тогда **стационарностью** будем определять следующую совокупность его свойств:

- Математическое ожидание ряда не меняется во времени: $M(t_1) = M(t_2) = \dots = M(t_N)$
- Ковариация смещённого на k отсчётов назад ряда не зависит от времени: $Cov(y_t, y_{t-k}) = \tau_k$

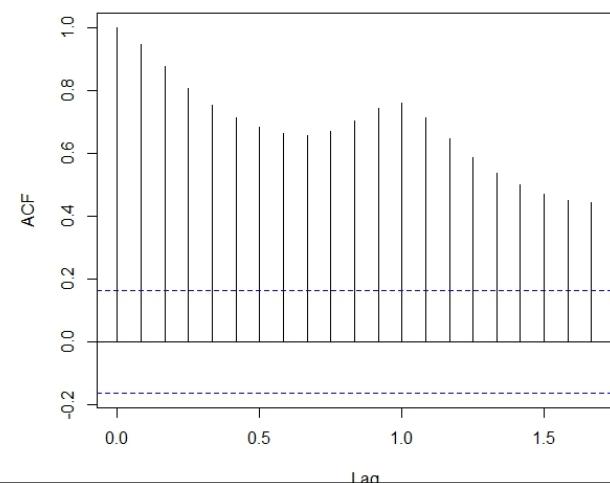
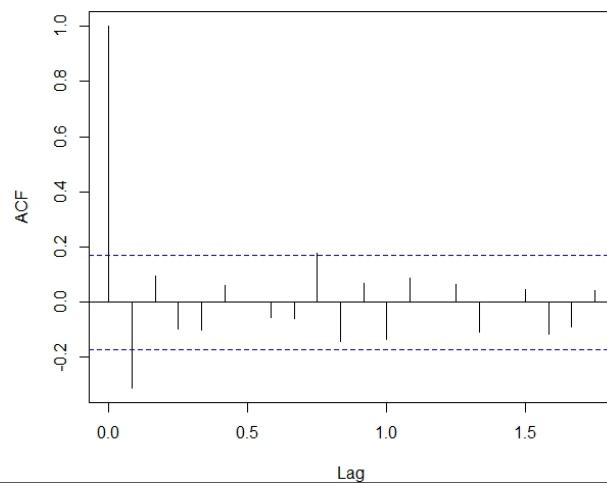
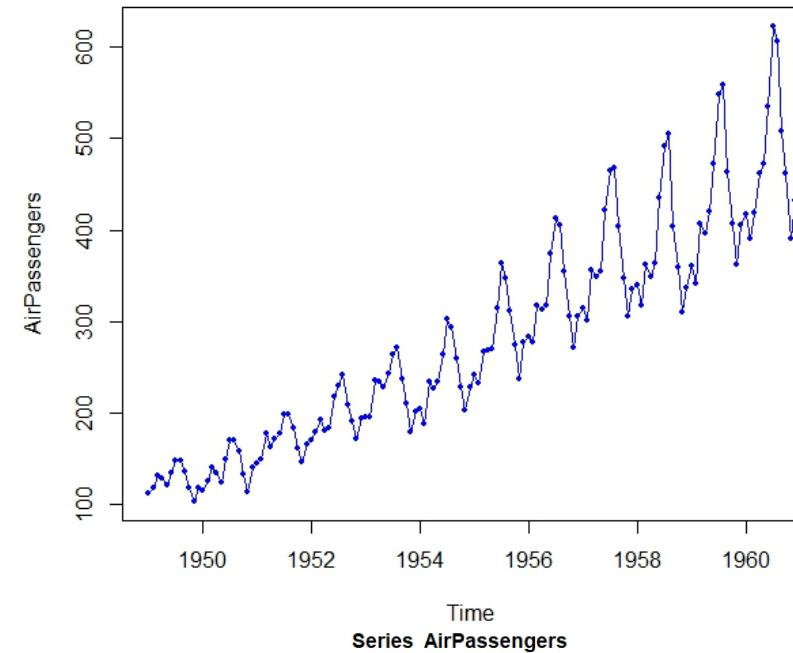
Временной ряд, обладающий данными свойствами будем называть **стационарным в широком смысле**.

Примеры стационарных и нестационарных рядов

Стационарный ряд

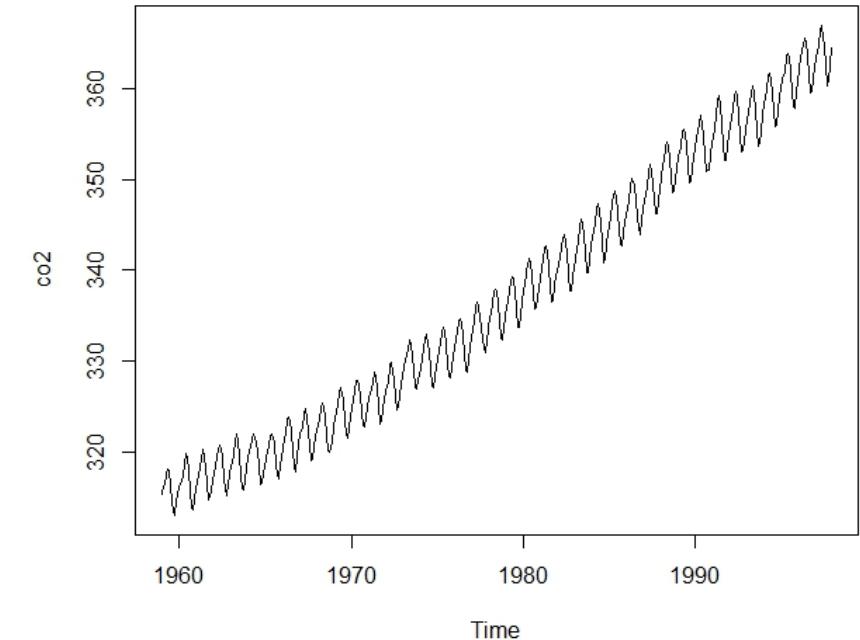
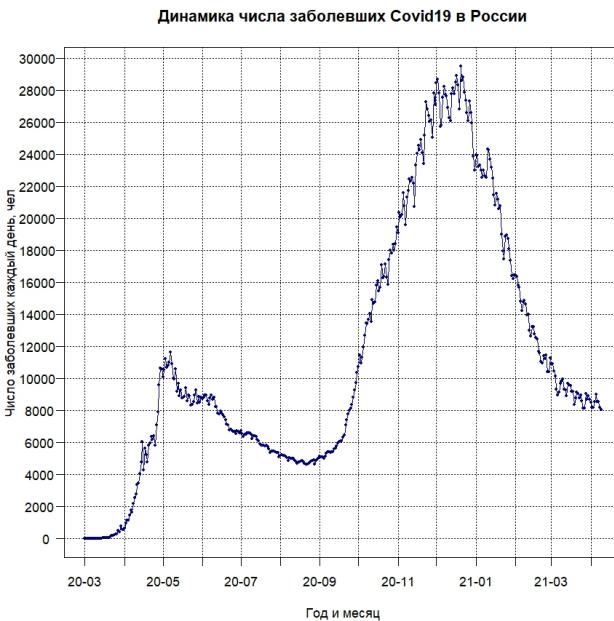


Нестационарный ряд

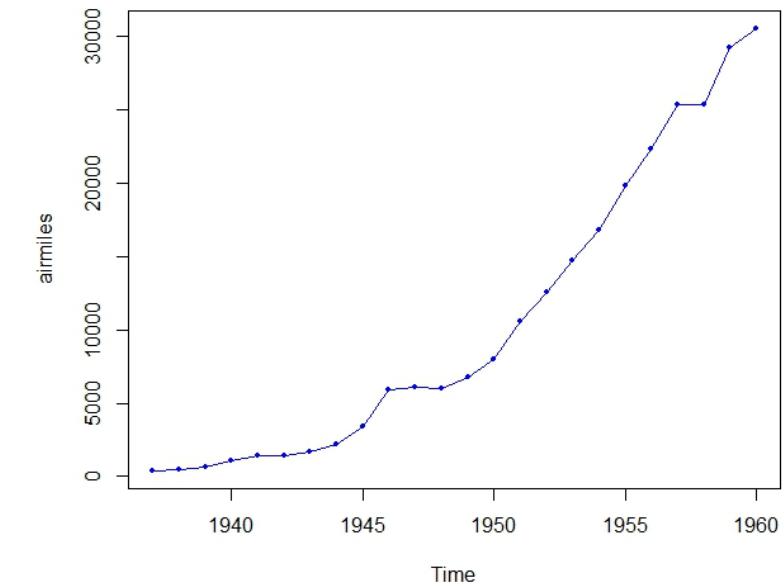
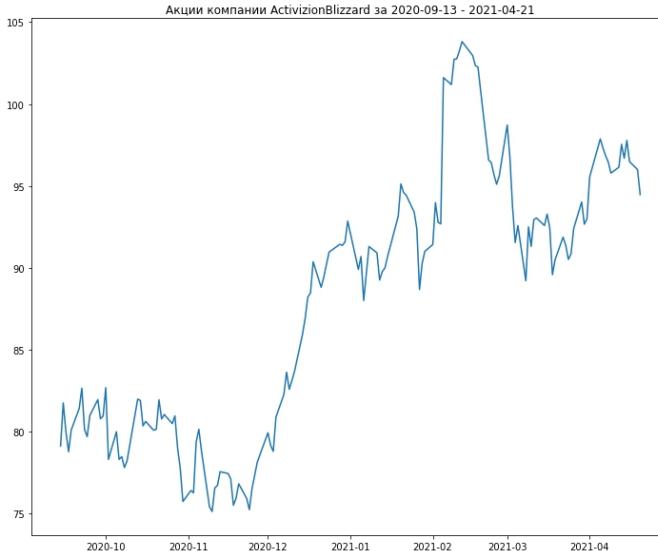


Нестационарные ряды

1. Ряды с трендом



2. Ряды с сезонностью



3. Комбинированные ряды

Тест Дикки-Фуллера

При помощи этого теста проверяют значение коэффициента a в авторегрессионном уравнении первого порядка AR(1):

$$y_t = a * y_{t-1} + \varepsilon_t$$

где y_t - временной ряд, ε_t - ошибка

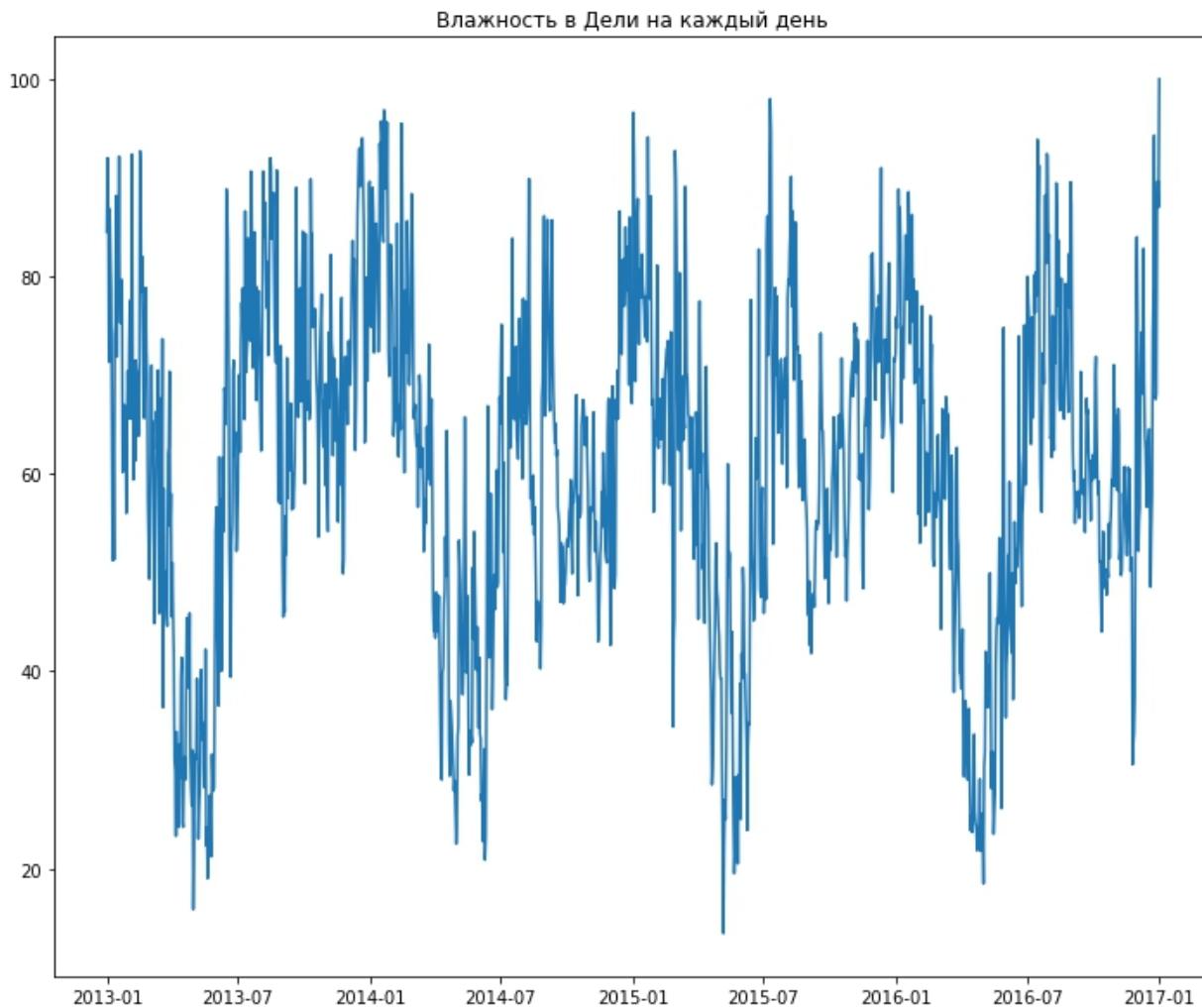
Если $a = 1$, то процесс имеет единичный корень, в этом случае ряд y_t не стационарен. Если $|a| < 1$, то ряд стационарный.

Тест Дикки-Фуллера рассчитывает р-статистику, в случае $p < 0.05$ гипотеза о стационарности ряда не отвергается.

Другие тесты:

- 1) Расширенный тест Дикки-Фуллера
- 2) Kwiatkowski–Phillips–Schmidt–Shin (KPSS)
- 3) Тест Филипса — Перрона

...



```
# ADF Test
result = adfuller(x=df1["humidity"], autolag='AIC')
print(f'ADF Statistic: {result[0]}')
print(f'p-value: {result[1]}')
for key, value in result[4].items():
    print('Critical Values:')
    print(f'    {key}, {value}')


# KPSS Test
result = kpss(x=df1["humidity"], regression='c')
print('\nKPSS Statistic: %f' % result[0])
print('p-value: %f' % result[1])
for key, value in result[3].items():
    print('Critical Values:')
    print(f'    {key}, {value}'')
```

```
# ADF Test
result = adfuller(x=df1["humidity"], autolag='AIC')
print(f'ADF Statistic: {result[0]}')
print(f'p-value: {result[1]}')
for key, value in result[4].items():
    print('Critical Values:')
    print(f'    {key}, {value}'')


# KPSS Test
result = kpss(x=df1["humidity"], regression='c')
print('\nKPSS Statistic: %f' % result[0])
print('p-value: %f' % result[1])
for key, value in result[3].items():
    print('Critical Values:')
    print(f'    {key}, {value}'')
```

Гипотеза о стационарности не отвергается

ADF Statistic: -3.675576919163339

p-value: 0.004470100478130758

Гипотеза о стационарности на константу не отвергается

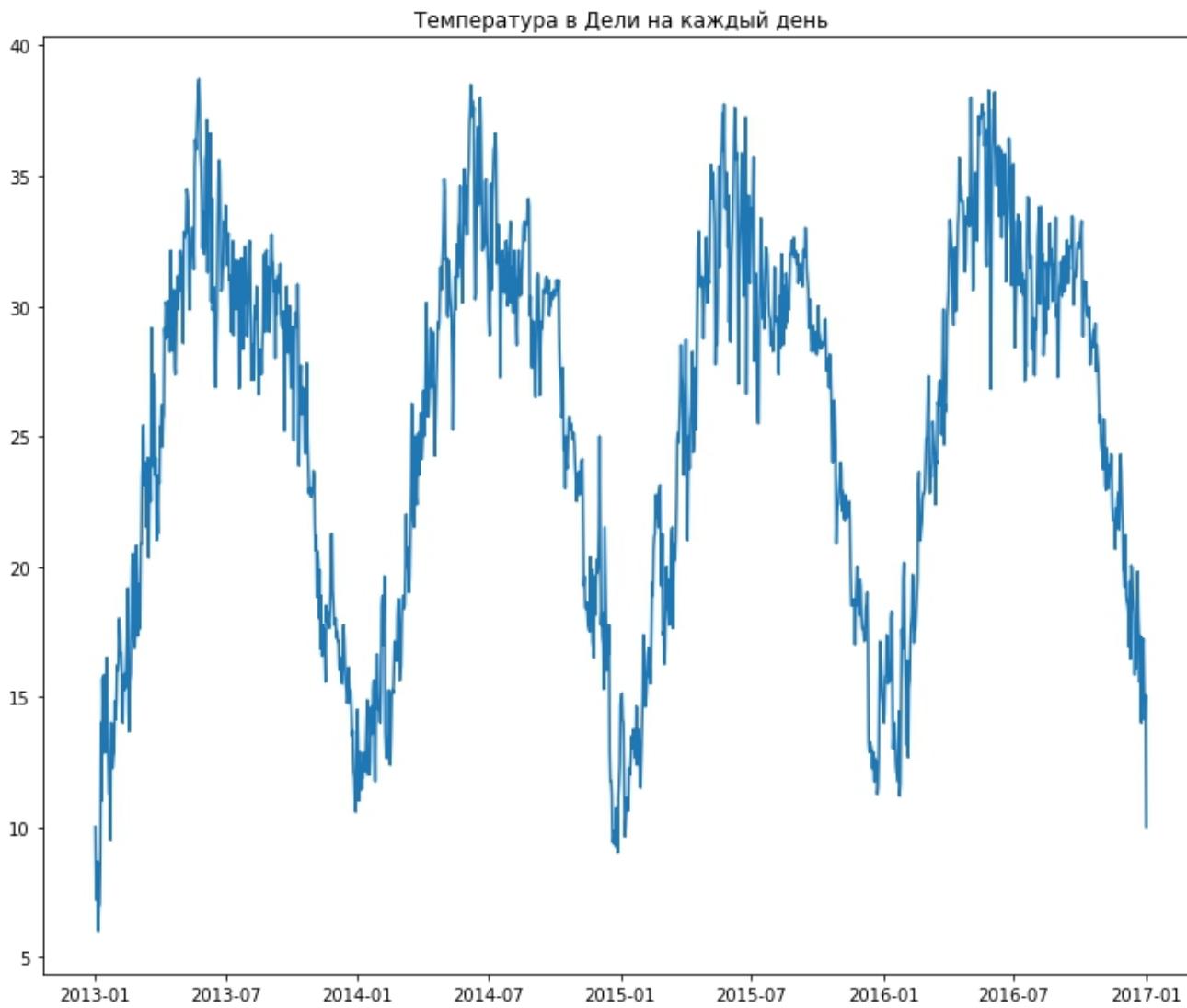
KPSS Statistic: 0.091737

p-value: 0.100000

Гипотеза о тренд-стационарности не отвергается

KPSS Statistic: 0.062512

p-value: 0.100000



Гипотеза о стационарности отвергается

ADF Statistic: -2.021069055920671

p-value: 0.2774121372301611

**Гипотеза о стационарности на константу не
отвергается**

KPSS Statistic: 0.194827

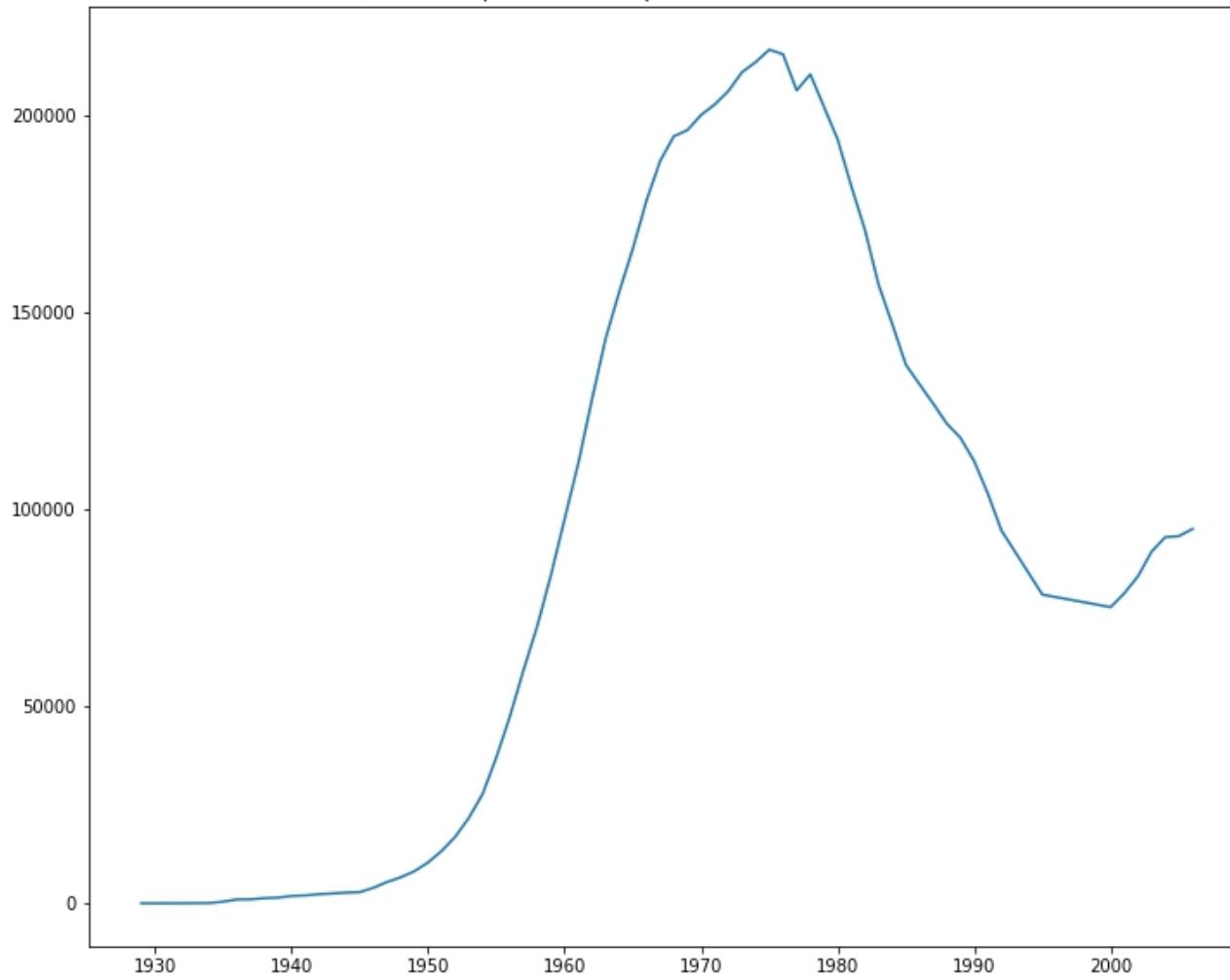
p-value: 0.100000

**Гипотеза о тренд- стационарности не
отвергается**

KPSS Statistic: 0.092940

p-value: 0.100000

Добыча нефти на Волго-Уральском бассейне, тыс. тонн



Гипотеза о стационарности отвергается

ADF Statistic: -2.2071277817347843

p-value: 0.20366866202477607

**Гипотеза о стационарности на константу
не отвергается**

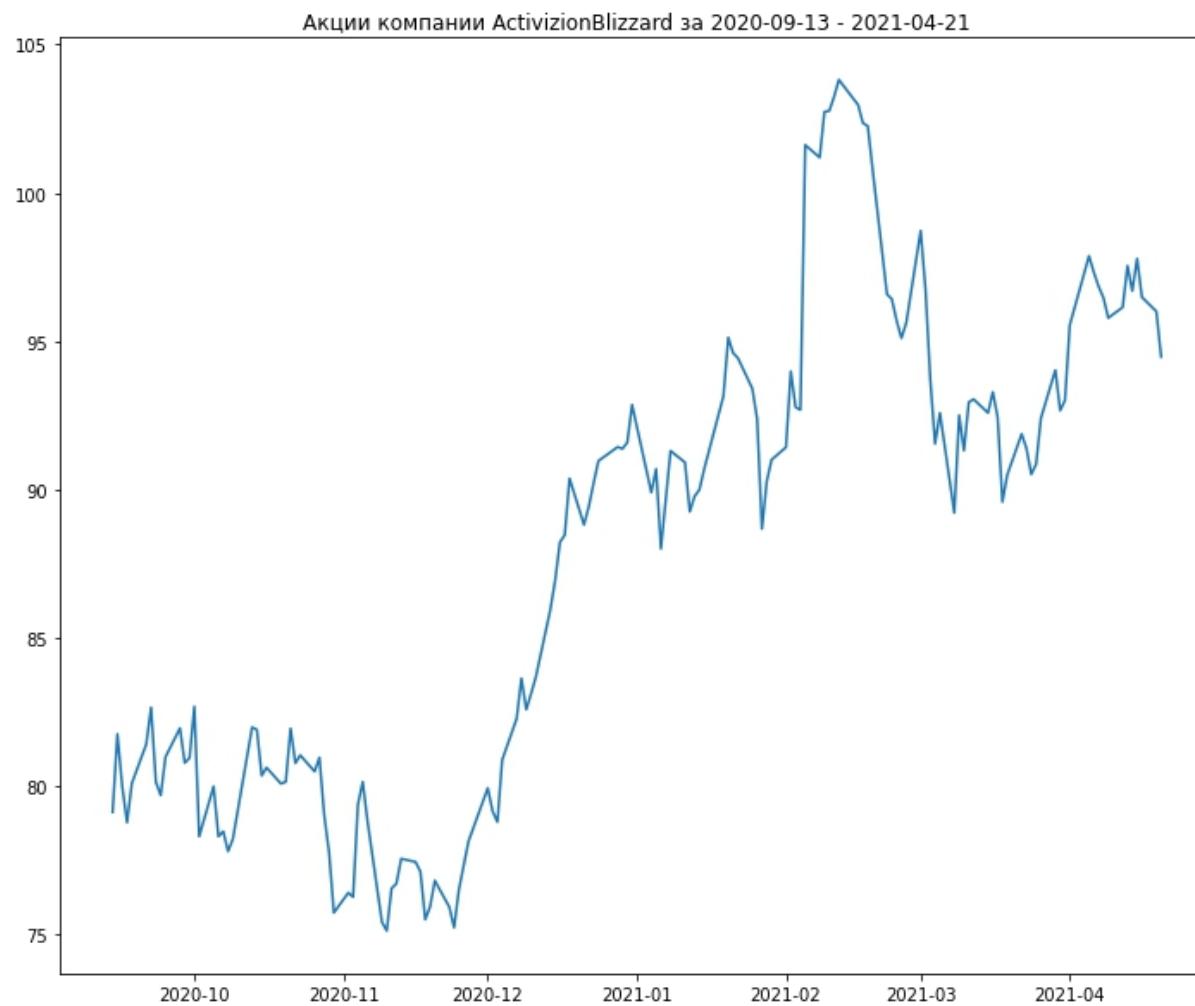
KPSS Statistic: 0.389589

p-value: 0.081642

**Гипотеза о тренд- стационарности не
отвергается**

KPSS Statistic: 0.129709

p-value: 0.080169



Гипотеза о стационарности отвергается

ADF Statistic: -1.3648482020671466

p-value: 0.5990075595044024

**Гипотеза о стационарности на константу
отвергается**

KPSS Statistic: 0.892035

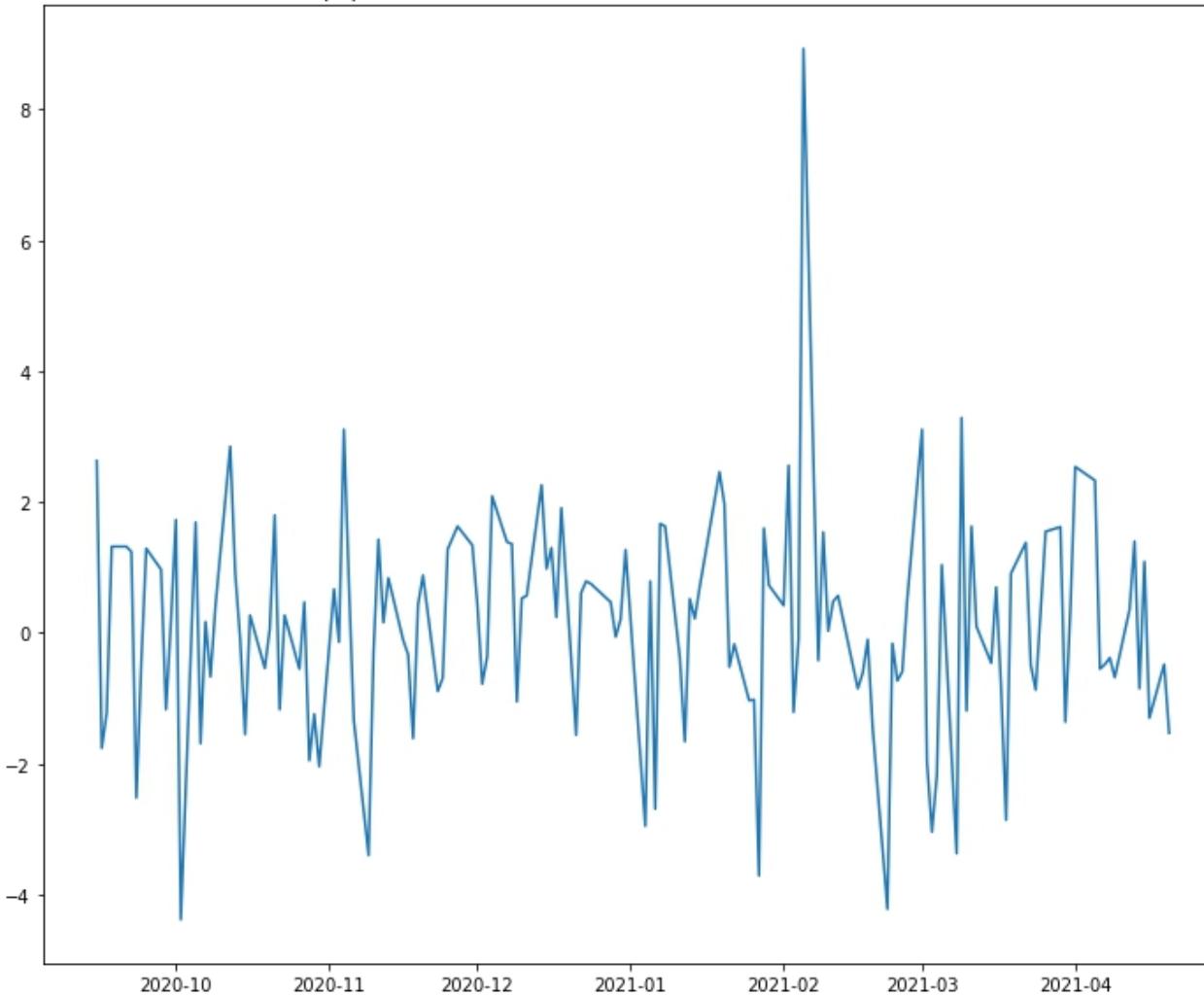
p-value: 0.010000

**Гипотеза о тренд-стационарности не
отвергается**

KPSS Statistic: 0.113863

p-value: 0.100000

Ежедневные приrostы акций компании ActivisionBlizzard за 2020-09-13 - 2021-04-21



Гипотеза о стационарности не отвергается

ADF Statistic: -12.734553769455175

p-value: 9.220656312688467e-24

**Гипотеза о стационарности на константу не
отвергается**

KPSS Statistic: 0.081748

p-value: 0.100000

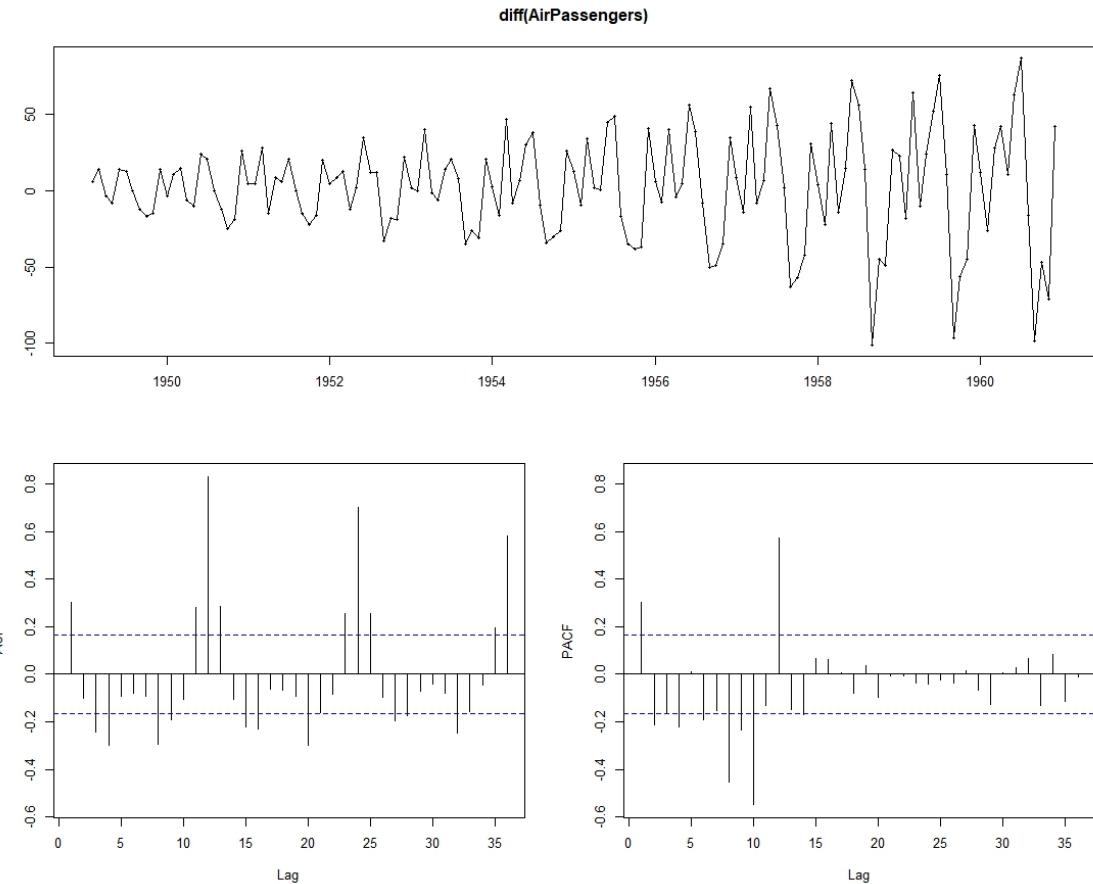
Проверка на сезонность

Автокорреляционная функция:

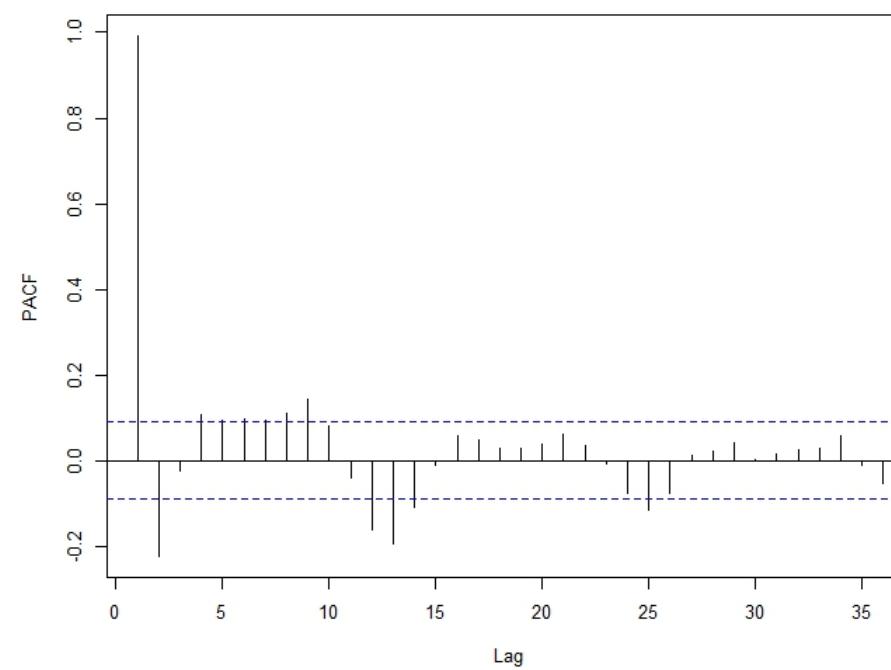
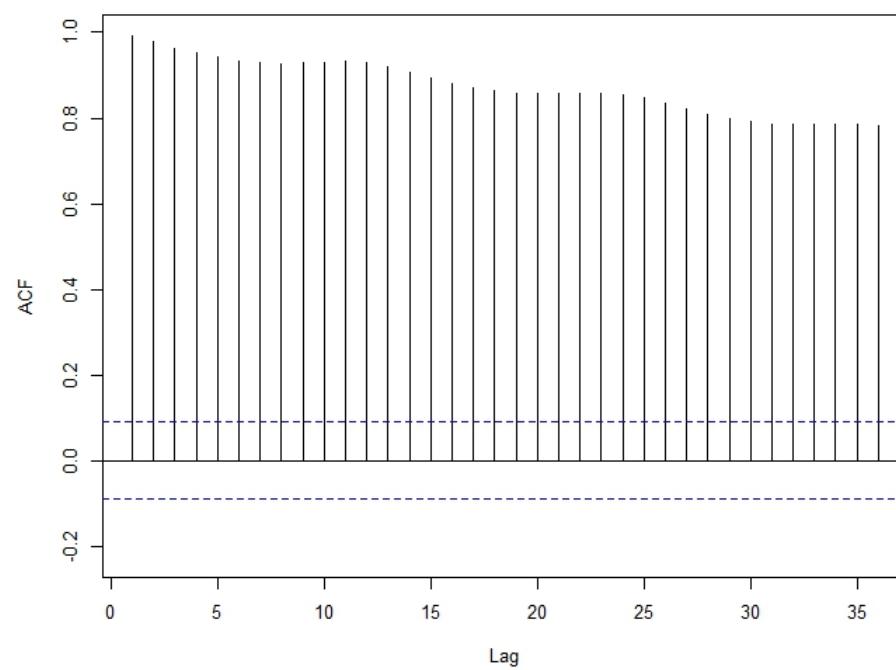
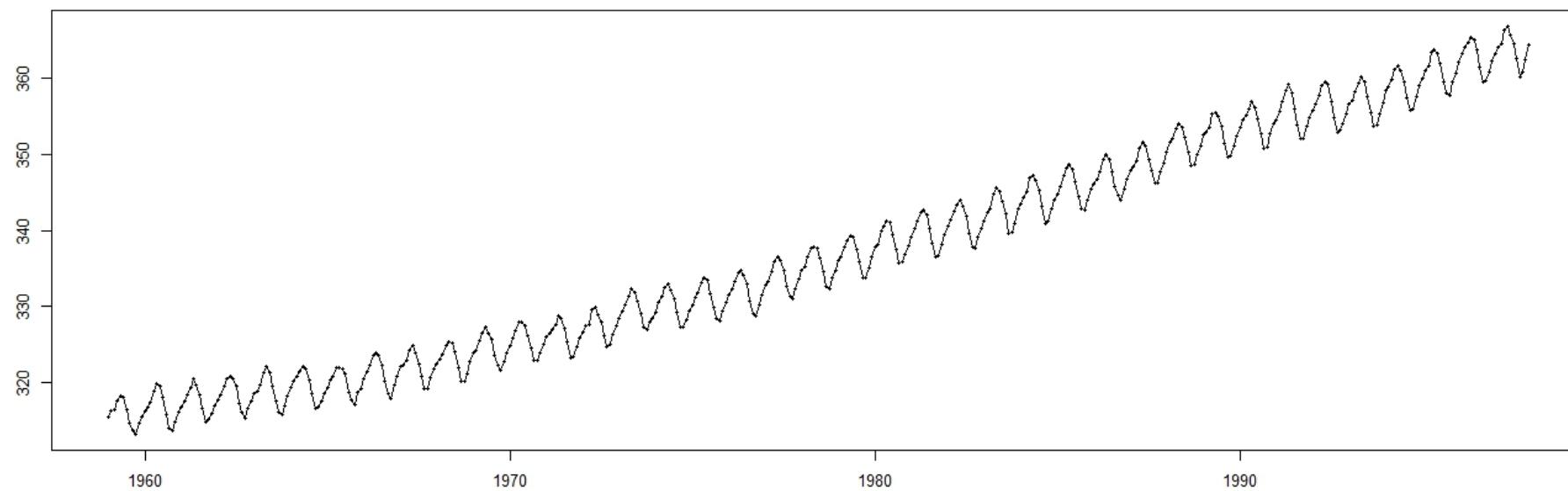
$$acf(y, lag) = \text{Corr}(y_t, y_{t-lag})$$

Доверительный интервал корреляции:

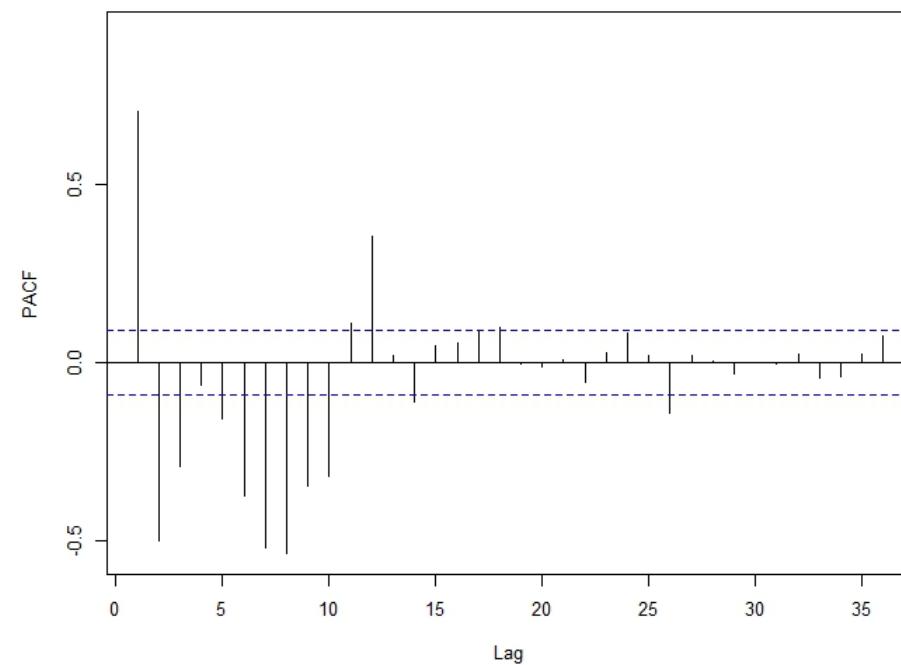
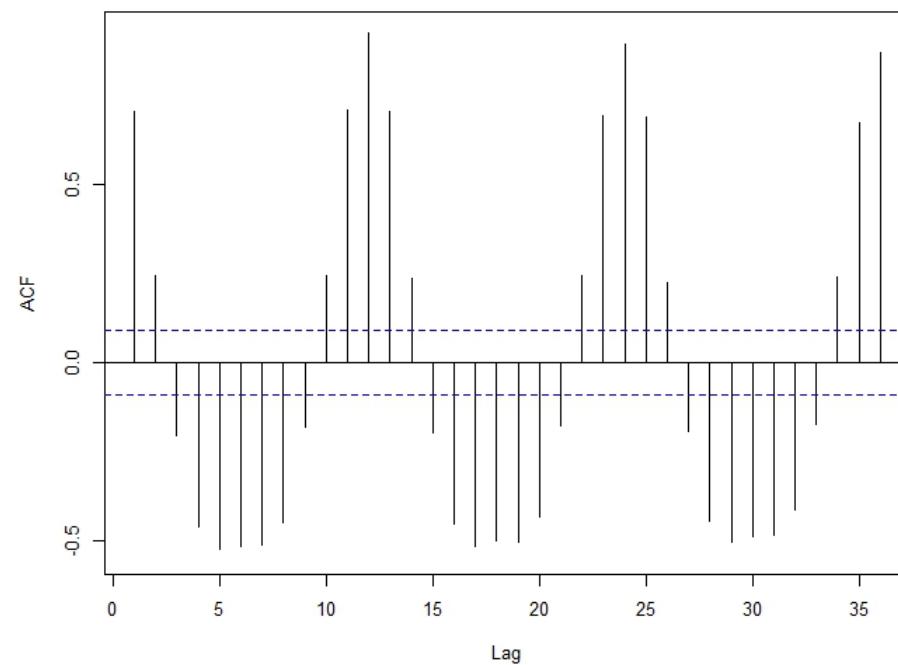
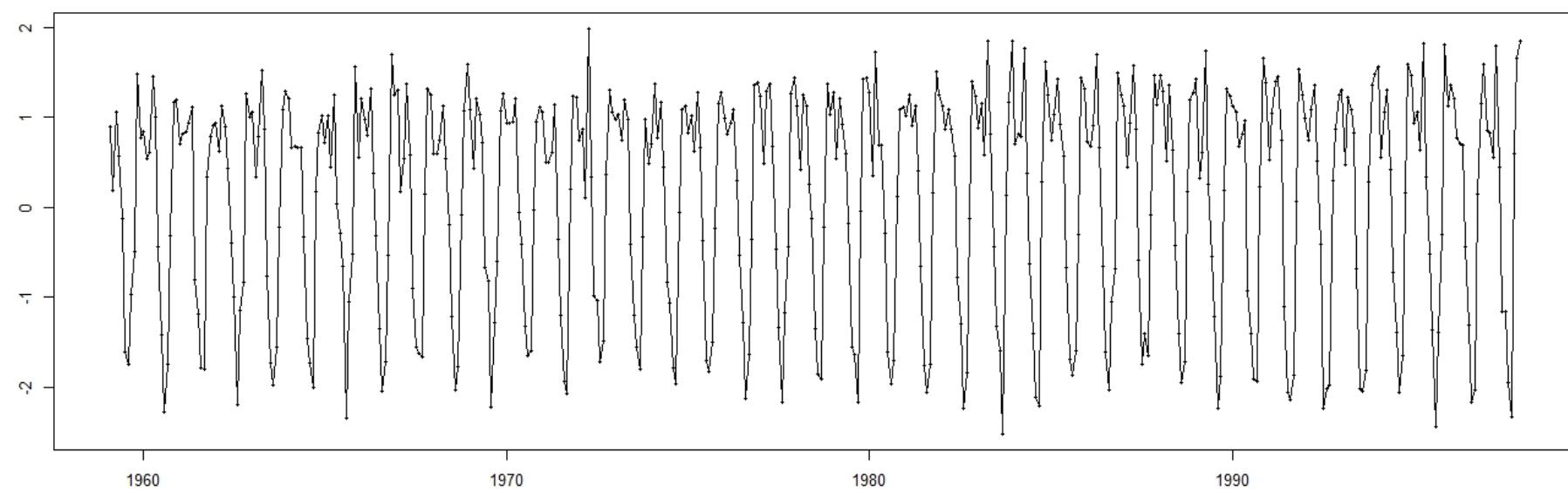
$$-1,96 * \frac{1}{\sqrt{n}} \leq acf(y, lag) \leq 1,96 * \frac{1}{\sqrt{n}}$$



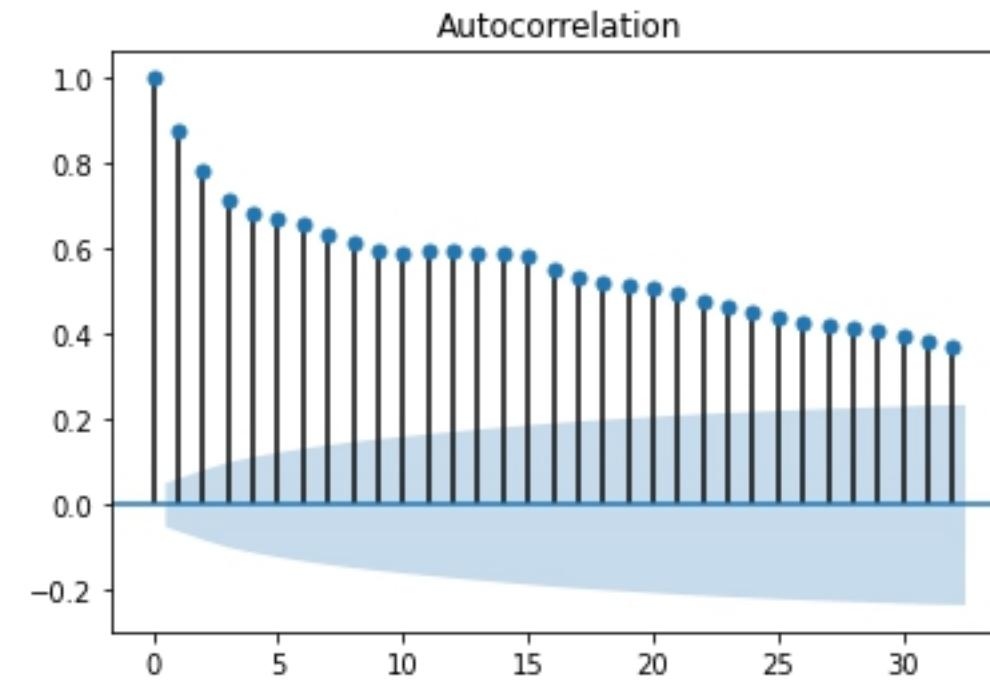
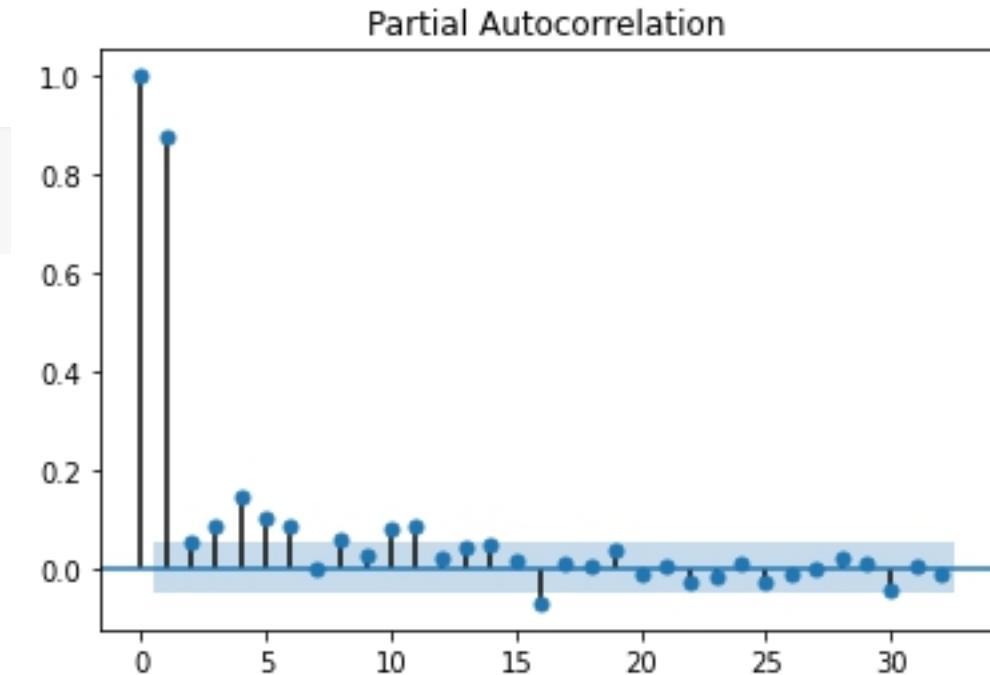
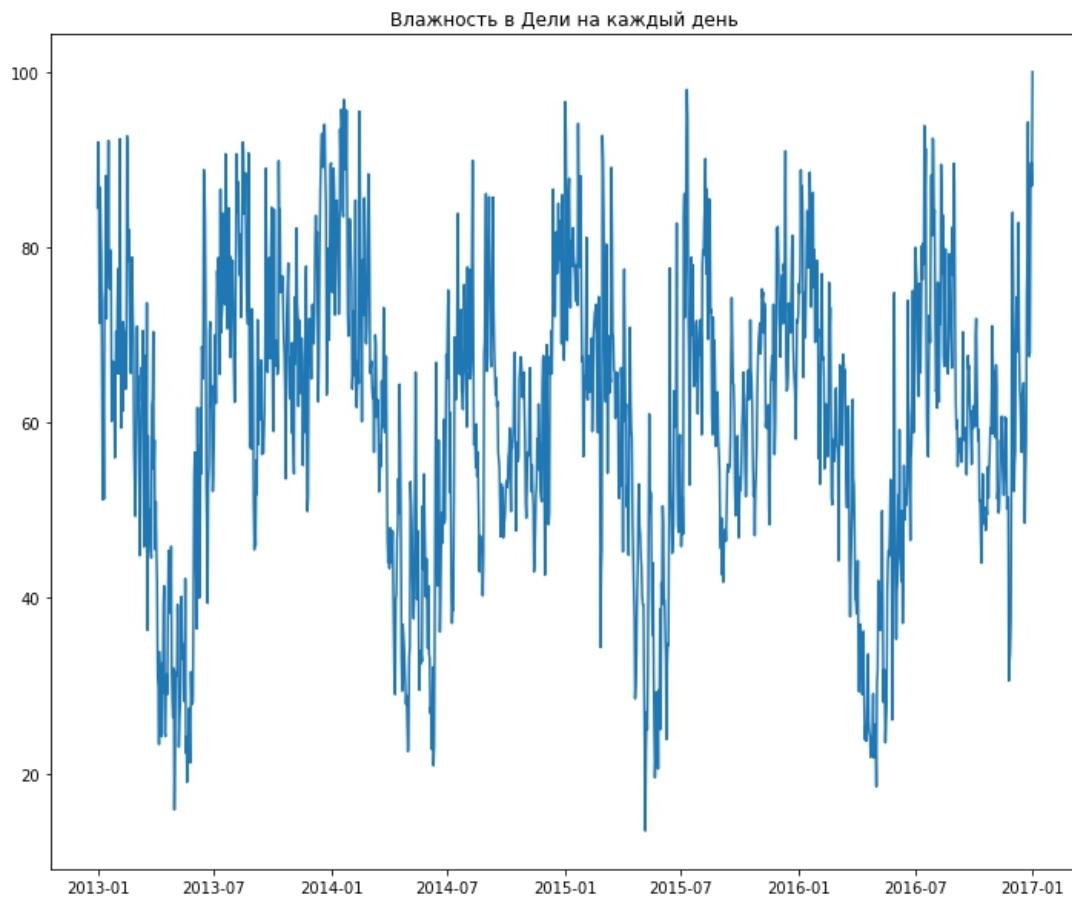
co2

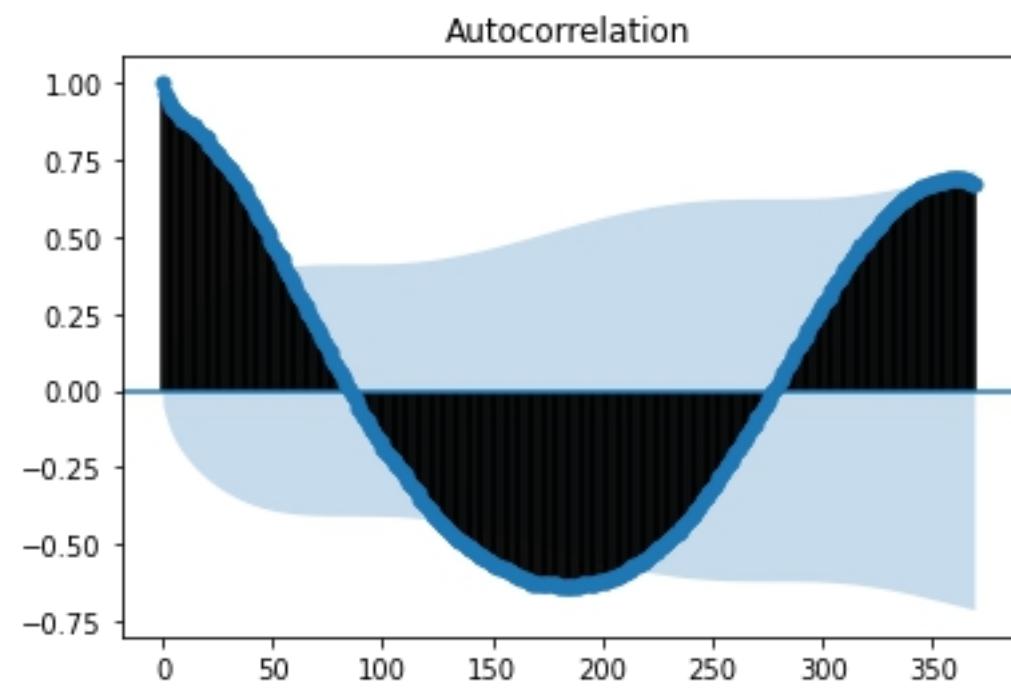
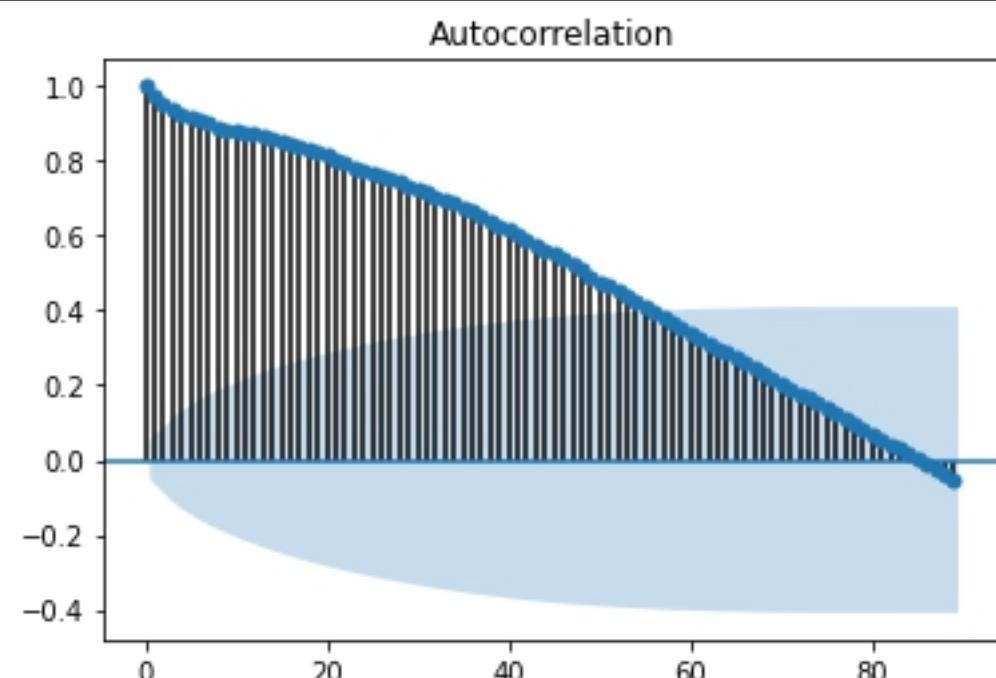
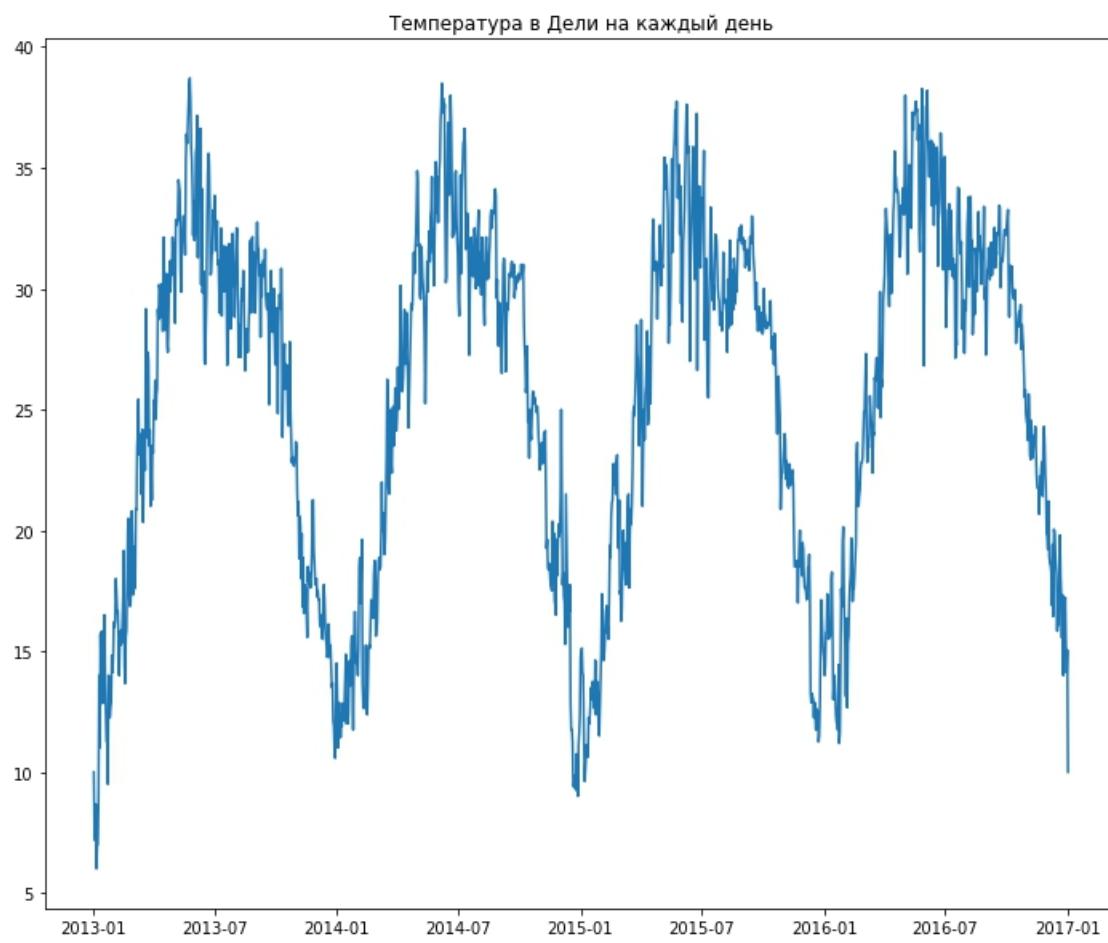


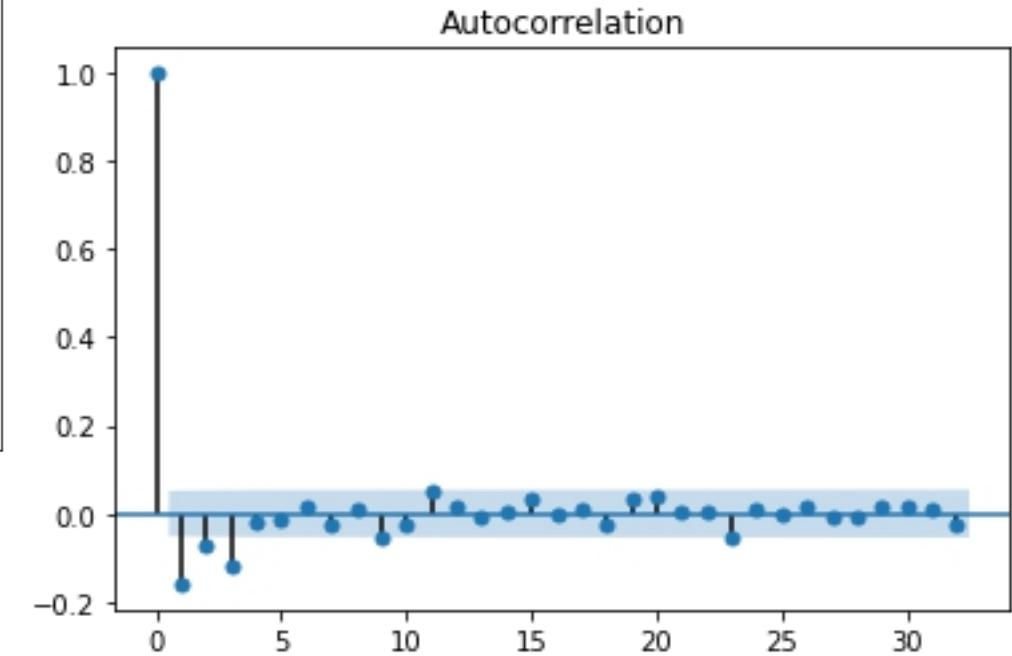
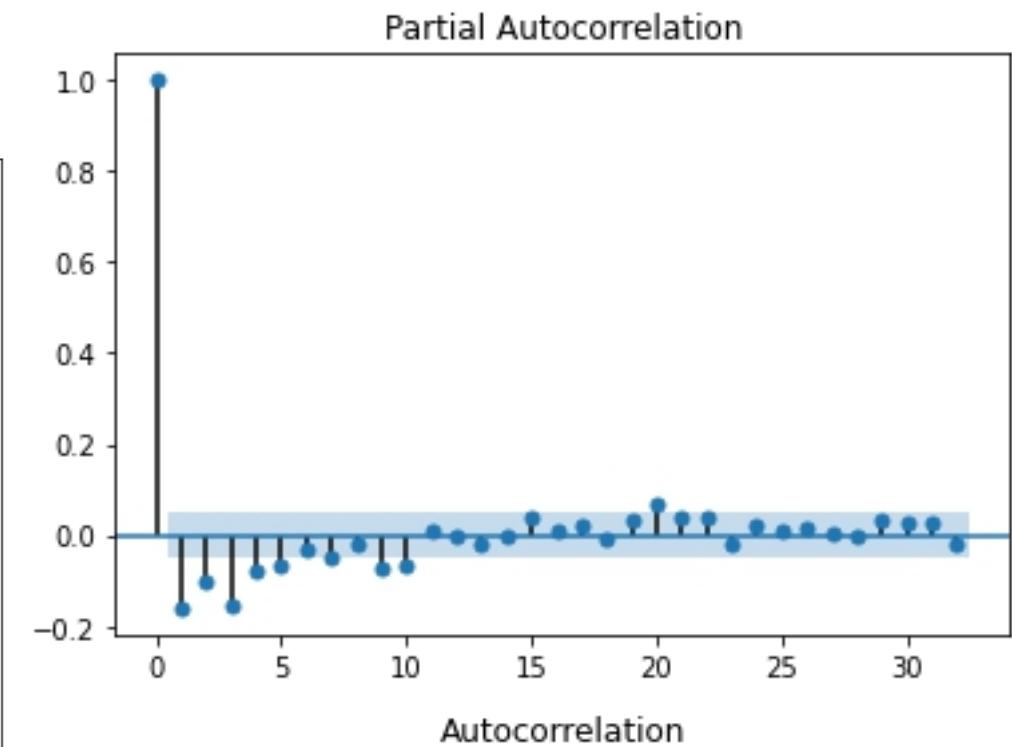
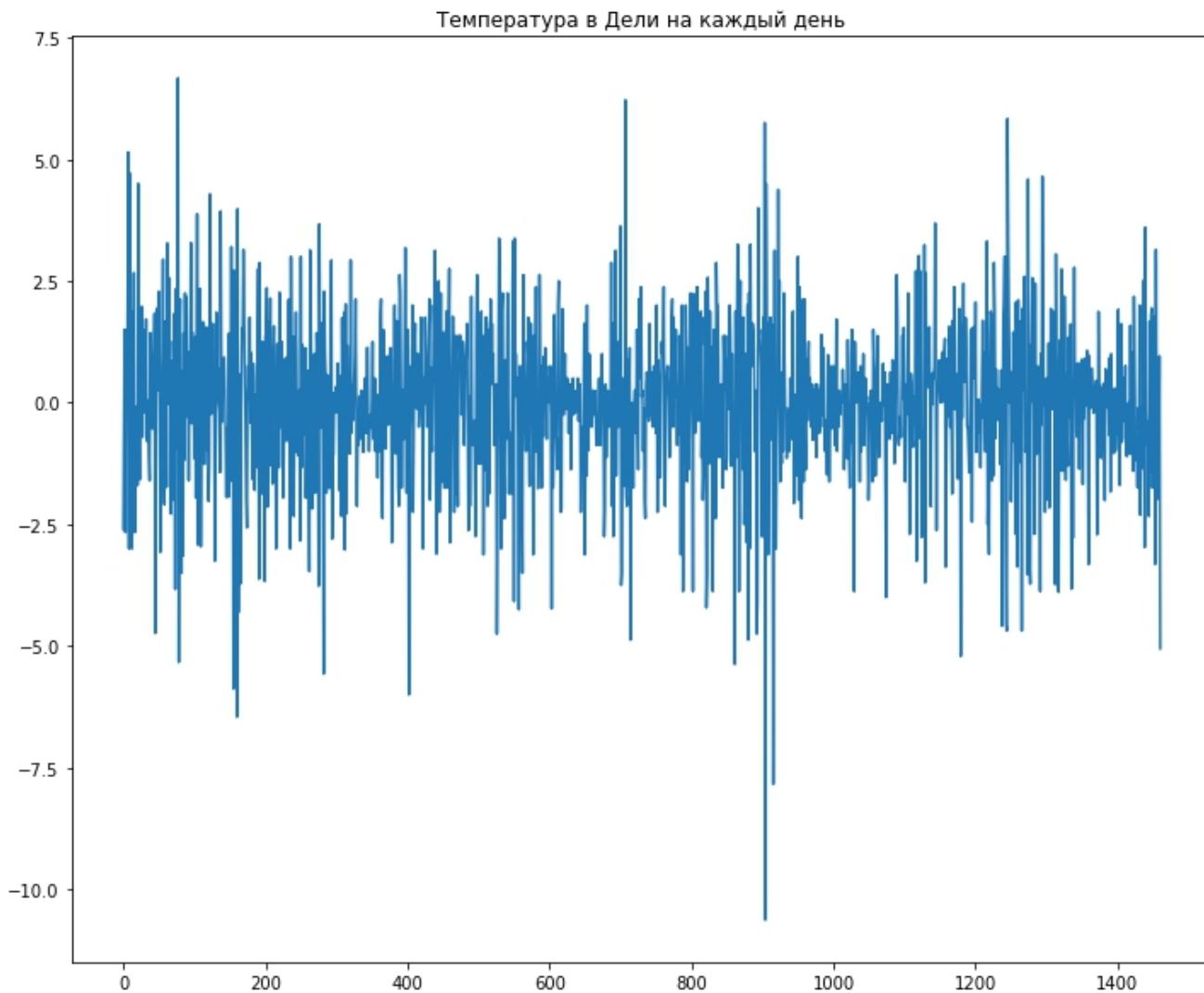
diff(co2)



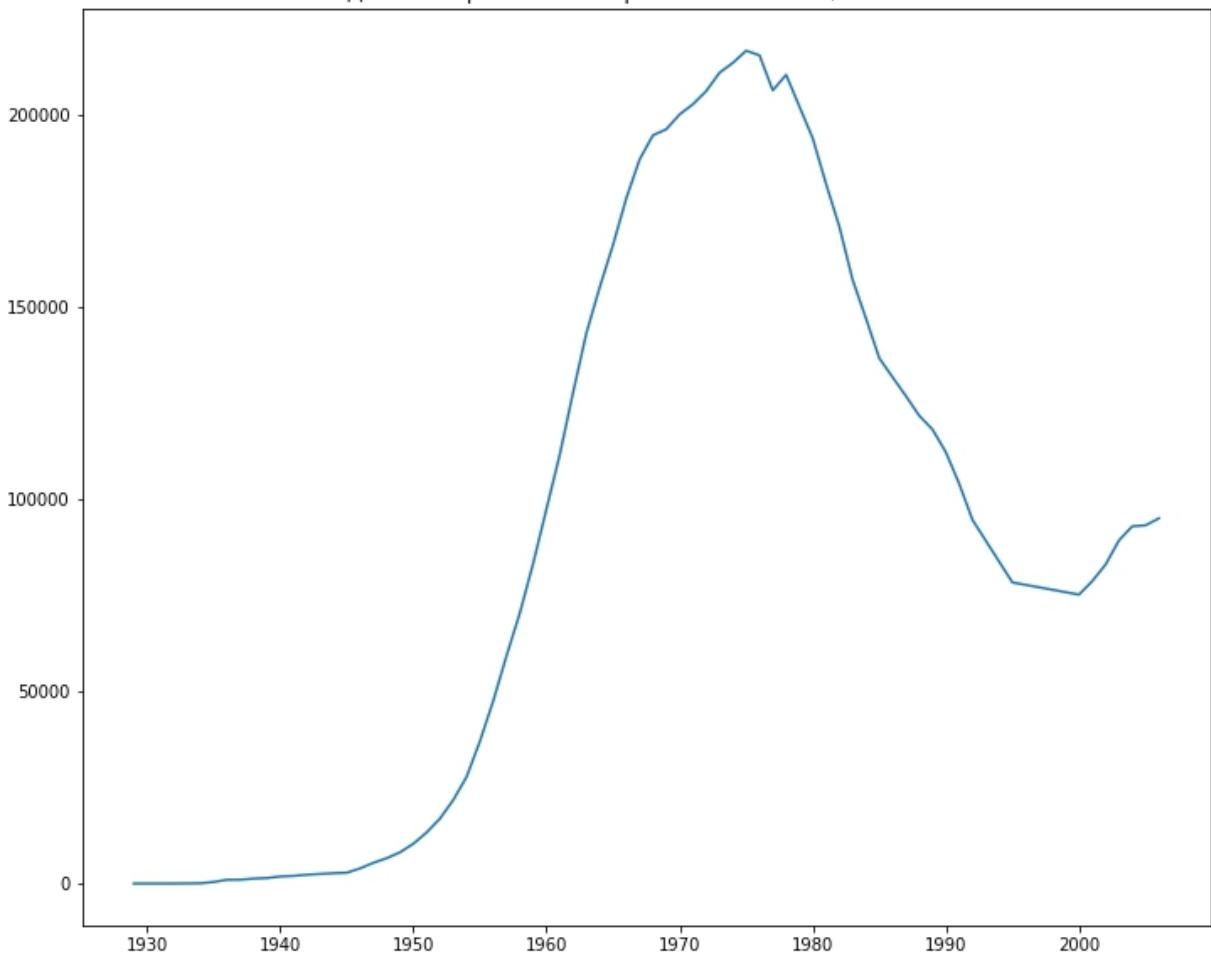
```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf  
plot_acf(df1["humidity"])  
plot_pacf(df1['humidity'])
```



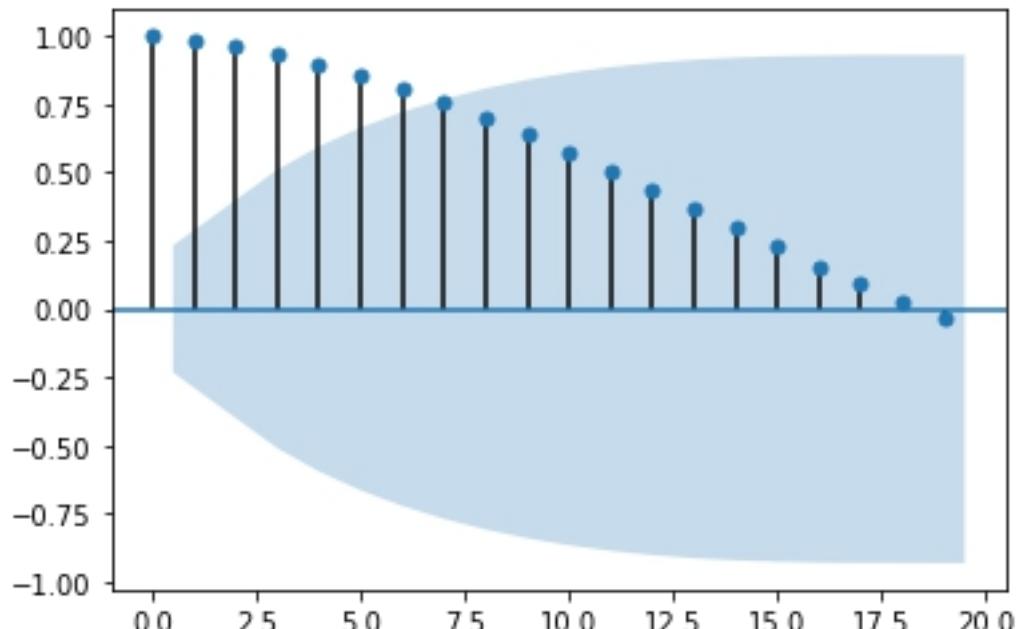




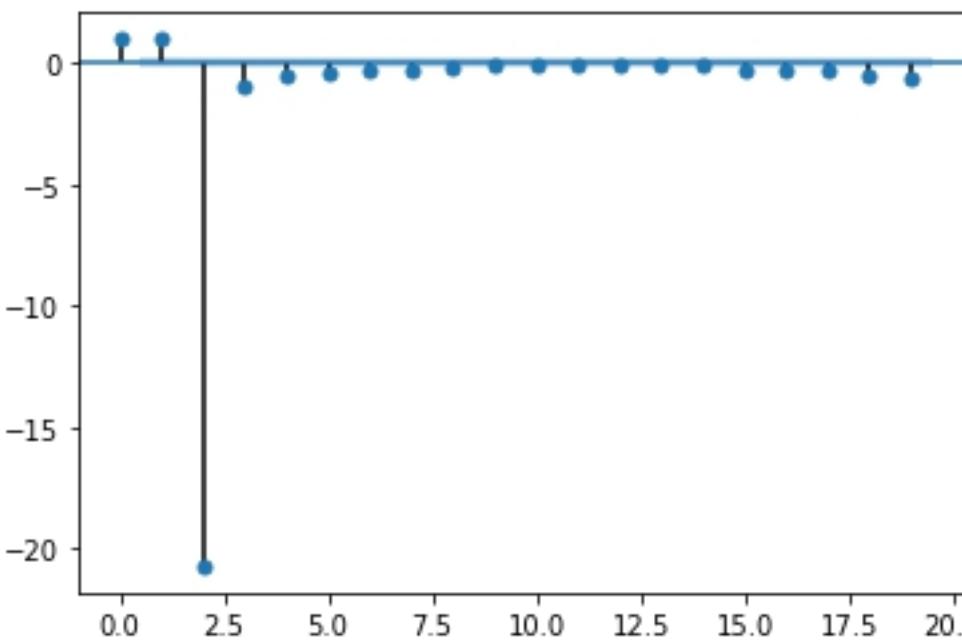
Добыча нефти на Волго-Уральском бассейне, тыс. тонн

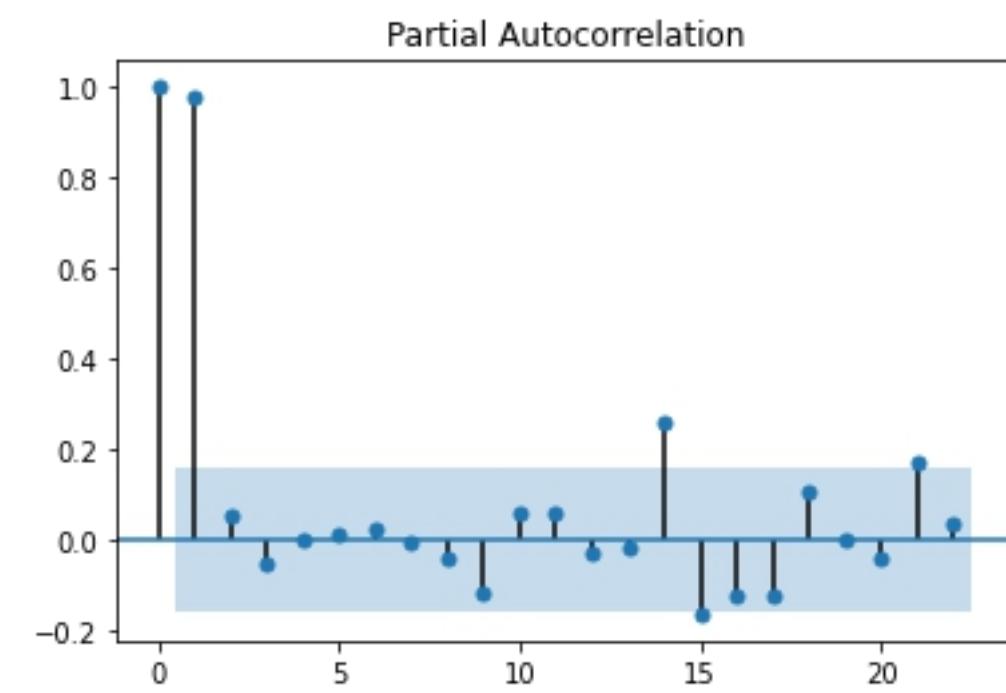
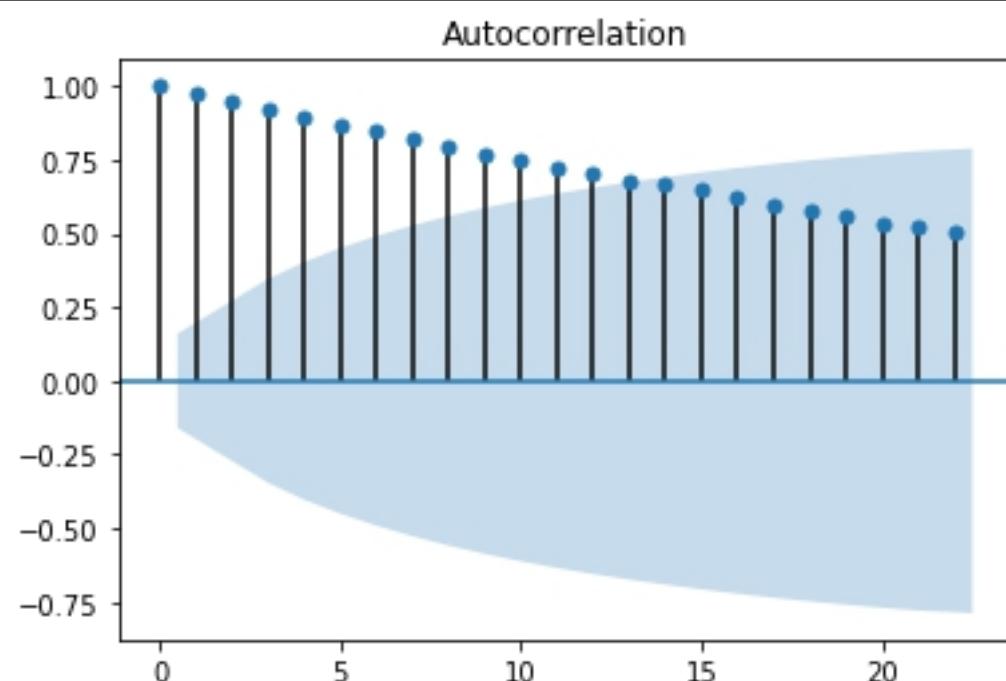
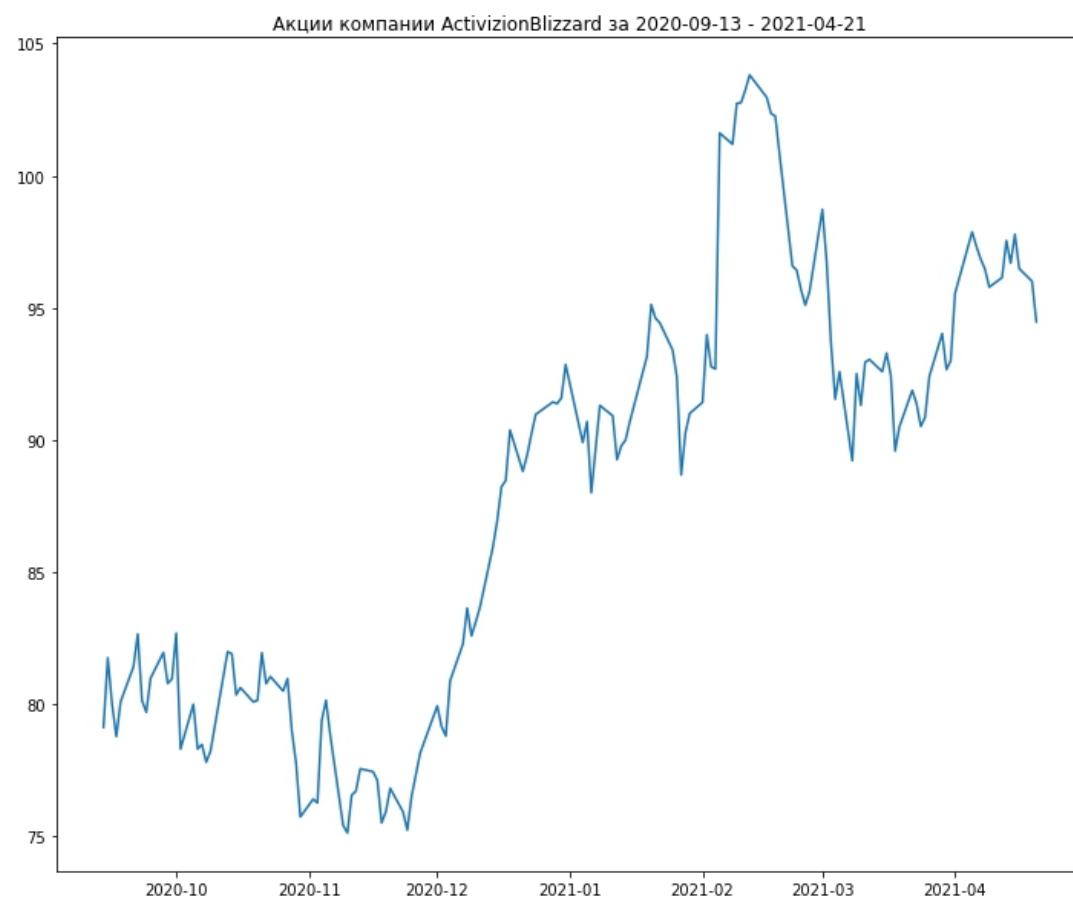


Autocorrelation

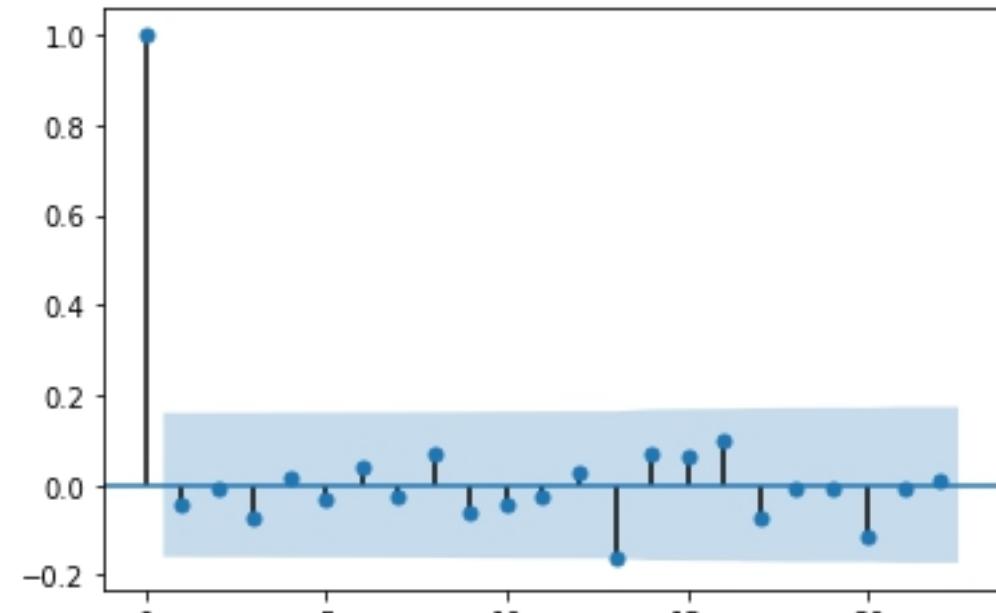


Partial Autocorrelation

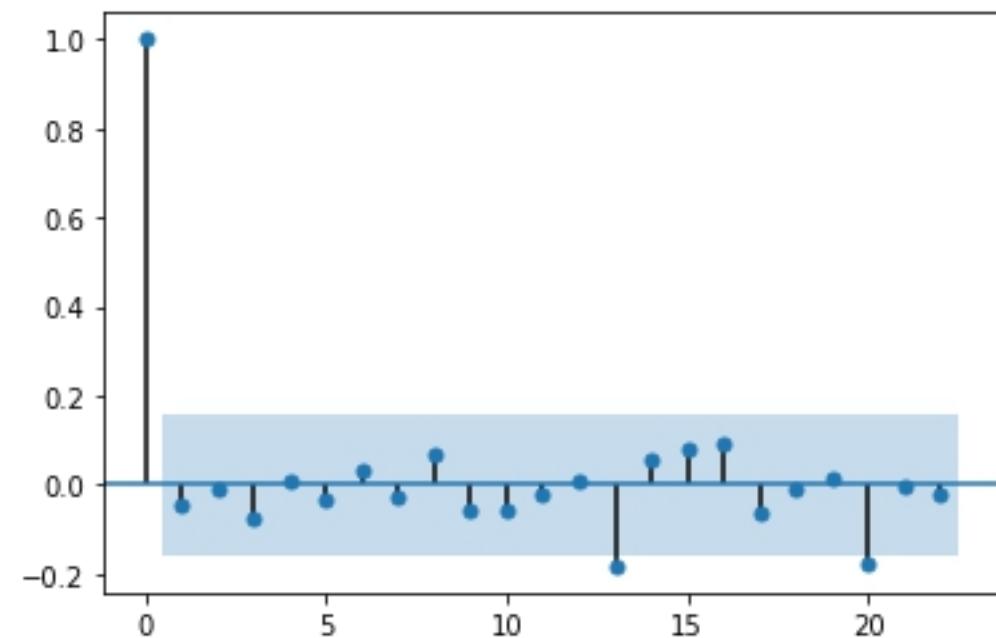




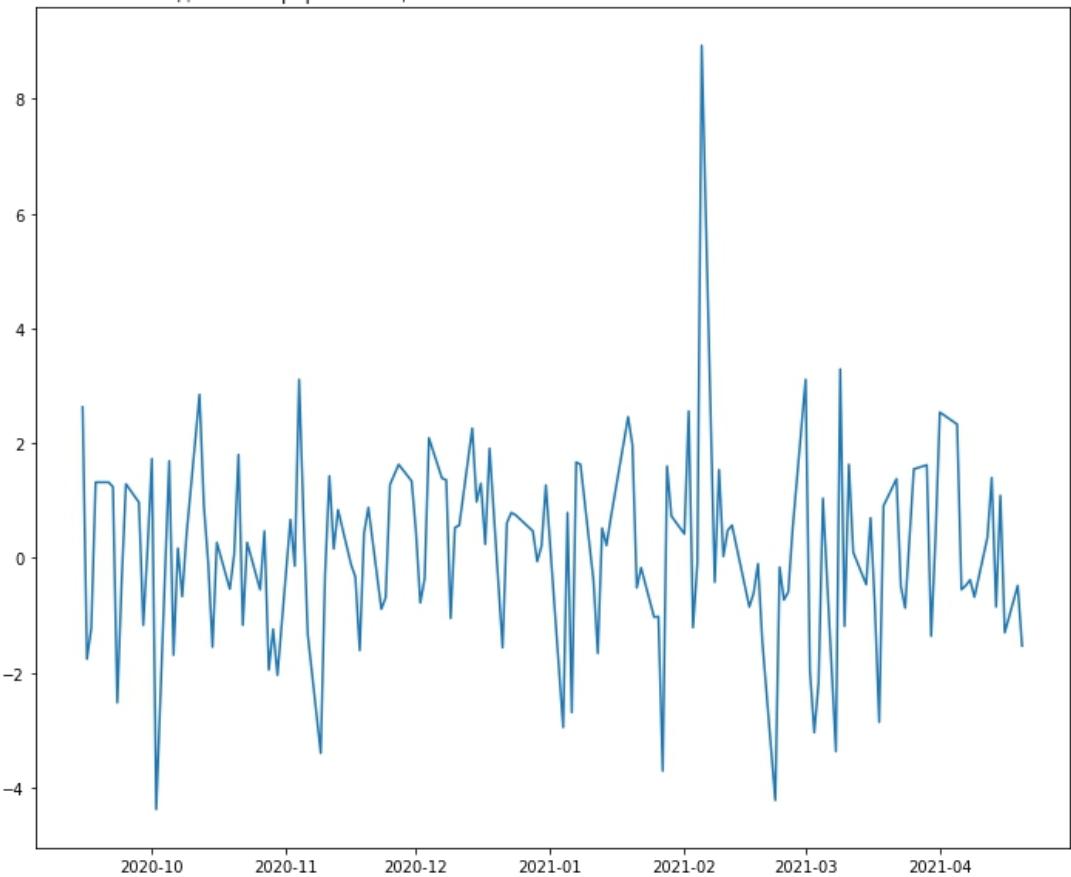
Autocorrelation



Partial Autocorrelation



Ежедневные приросты акций компании ActivisionBlizzard за 2020-09-13 - 2021-04-21



Подходы к моделированию стационарных рядов

Если ряд стационарен - **ARMA(p, q)**

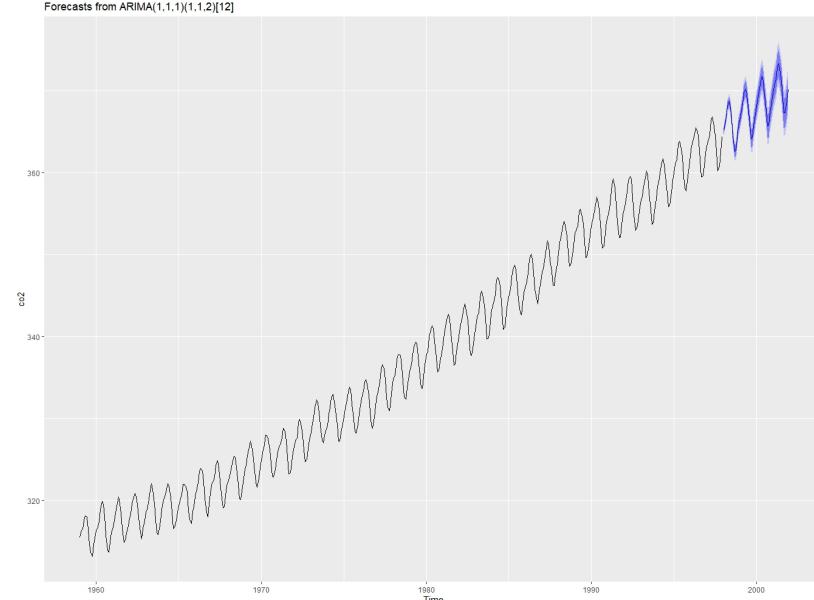
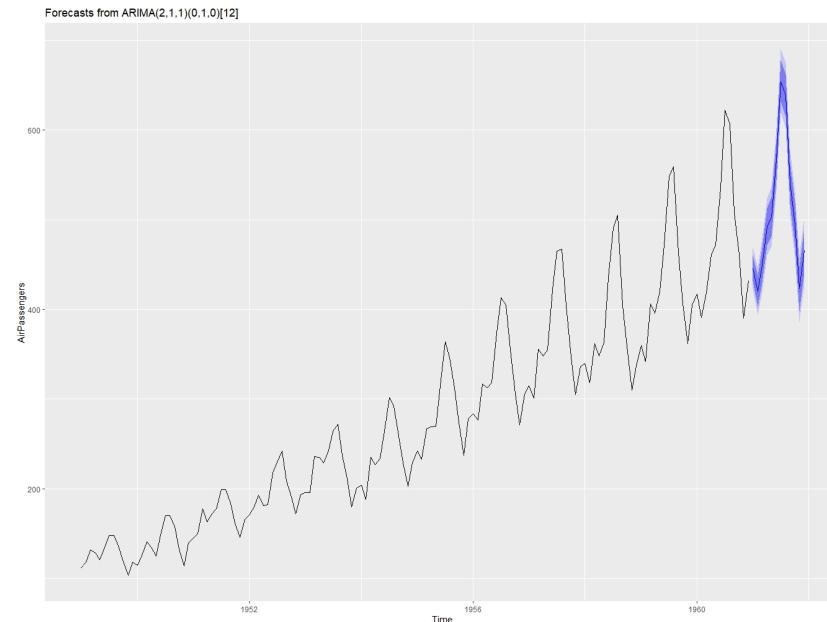
Если ряд имеет тренд - **ARIMA(p, d, q)**

Если ряд имеет сезонность -

SARMA (p, q) x (P, Q)

Если ряд имеет сезонность и тренд -

SARIMA (p, d, q)x(P, D, Q)



Методы разложения временного ряда на
трендовую и сезонную составляющую

Методы разложения временных рядов

<https://otexts.com/fpp2/decomposition.html>

1. Moving Average
2. Classical decomposition
3. X11 decomposition
4. Seasonal Extraction in ARIMA Time Series “SEATS”
5. STL

STL разложение

STL - процедура декомпозиции временного ряда на сезонную, трендовую составляющую и остатки на основе построения наилучшей локальной аппроксимации (Loess)

Для наилучшей работы данного метода необходимо знать сезонный лаг ряда исходных данных

STL - применим только для аддитивных моделей. На самом деле работает со всеми моделями, однако это не предпочтительно.

Аддитивная модель:

$$y_t = T_t + S_t + \varepsilon_t$$

Циклическая компонента для малых лагов сезонности будет включена в тренд.

Является лучшим методом разложения временного ряда.

Внутренний цикл STL

1. Вычесть тренд

$$T_0^v = 0, p = 0, Y_* = Y_v - T_k^v$$

2. Сортируем каждый сезонный компонент по периодам, сглаживаем по Loess

$$C_{k+1}^v = \text{Loess}(Y_*)$$

3. Глубокое сглаживание сезонных компонент методами MA или LOESS

$$L_{k+1}^v$$

4. Детрендирование сглаженных сезонных компонент

$$S_{k+1}^v = C_{k+1}^v - L_{k+1}^v$$

5. Десезонализация исходного ряда

$$Y_{**} = Y_v - S_{k+1}^v$$

Y_v - исходный ряд

p - веса для Loess

T - тренд

C - ряд со сглаженными компонентами сезонности

L - ряд, полученный после MA или Loess

S - сезонный компонент без тренда

6. Сглаживание тренда

$$T_{k+1}^v = \text{Loess}(Y_{**})$$

Внешний цикл

1. Считаем остатки

$$R_v = Y_v - T_v - S_v$$

R - остатки

Y - исходный временной ряд

T - сглаженный тренд без сезонности

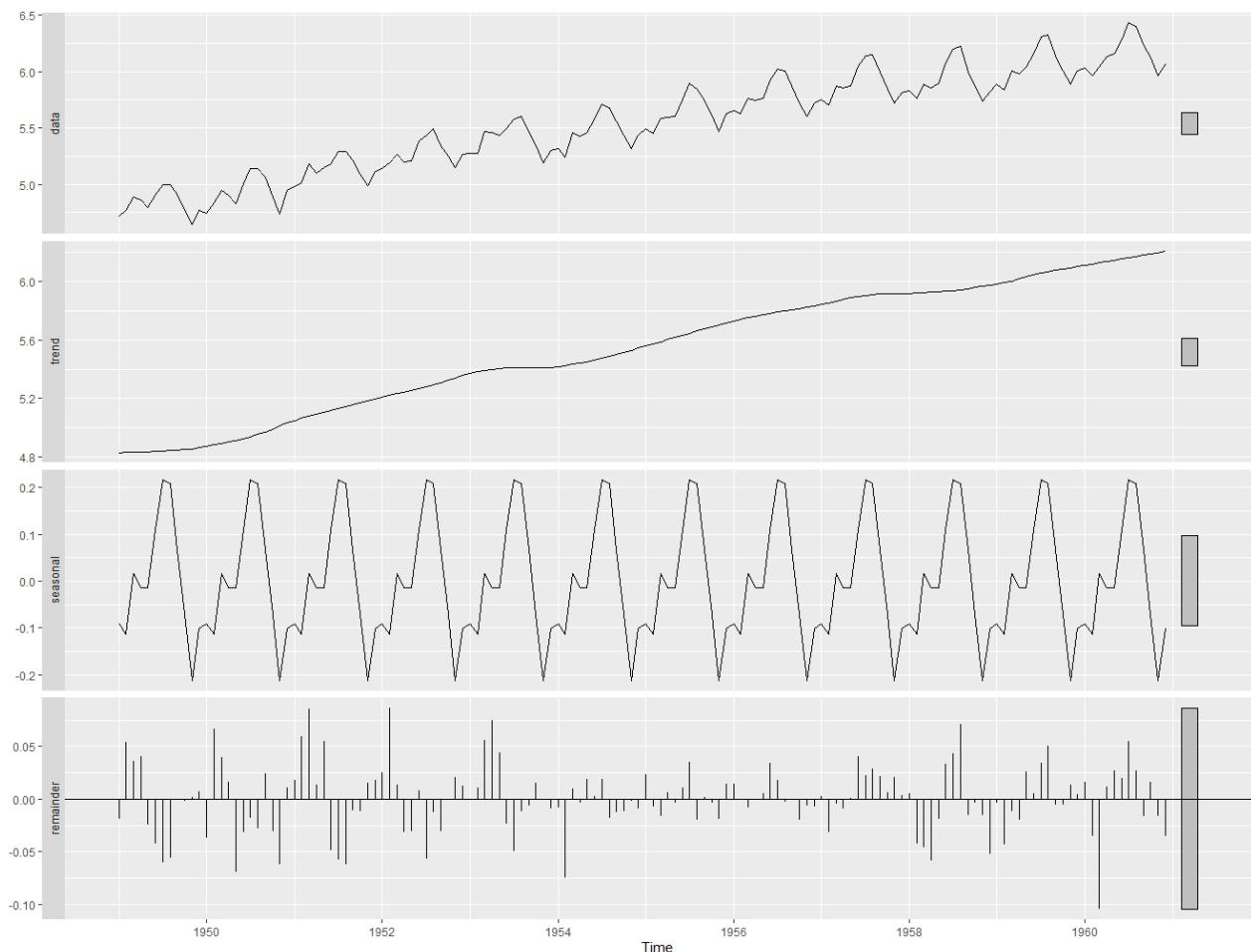
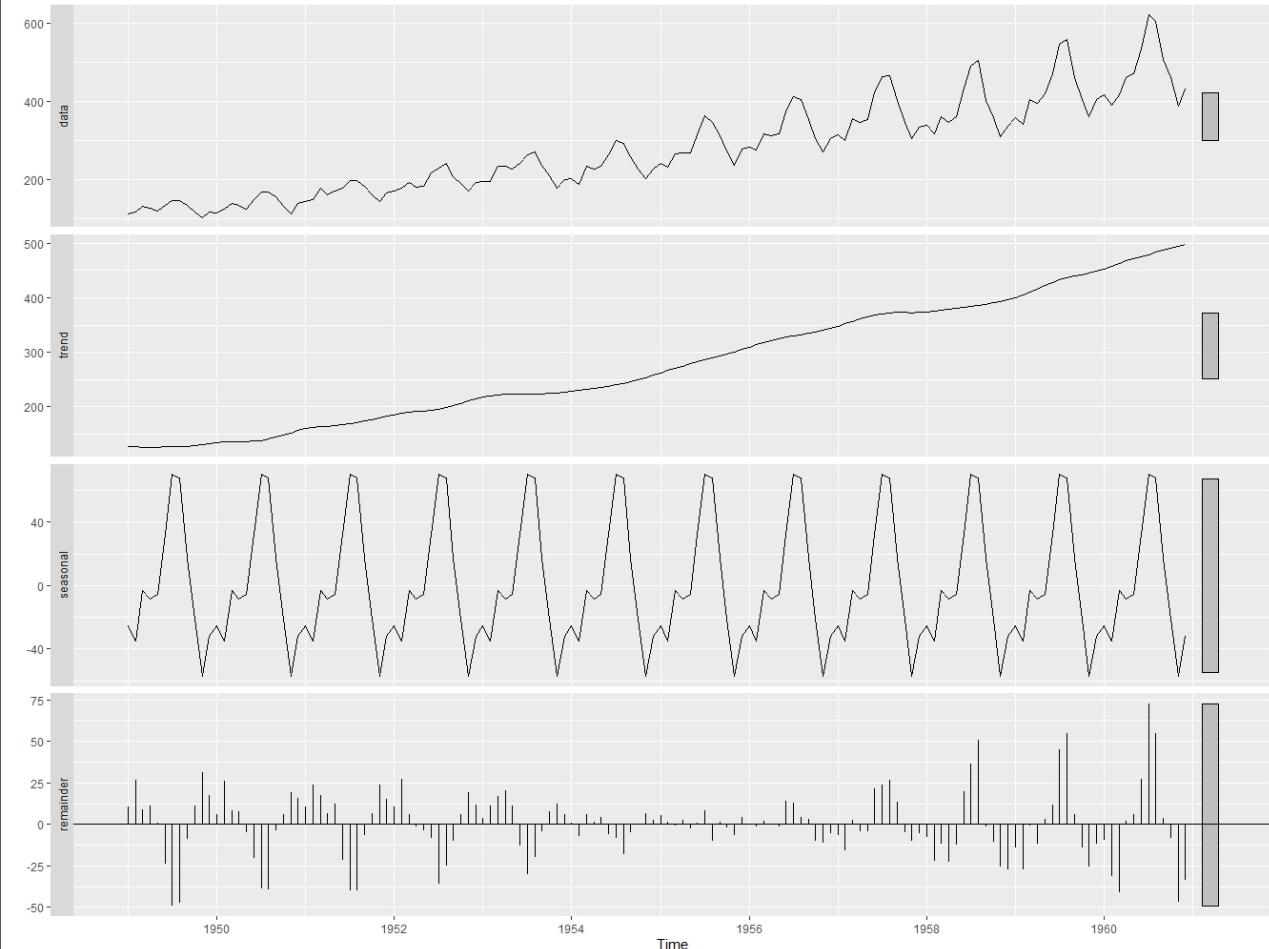
S- сезонная компонента без тренда

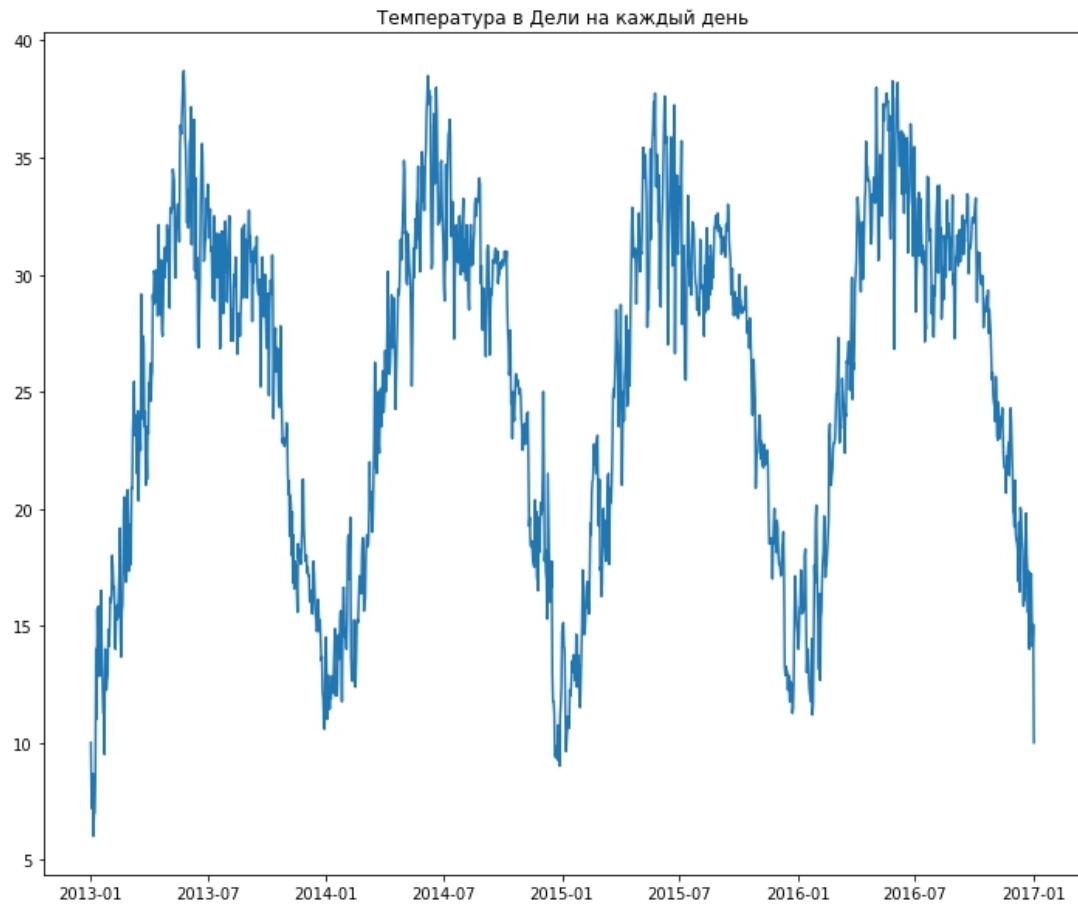
p - веса для Loess

2. Пересчитываем веса для Loess

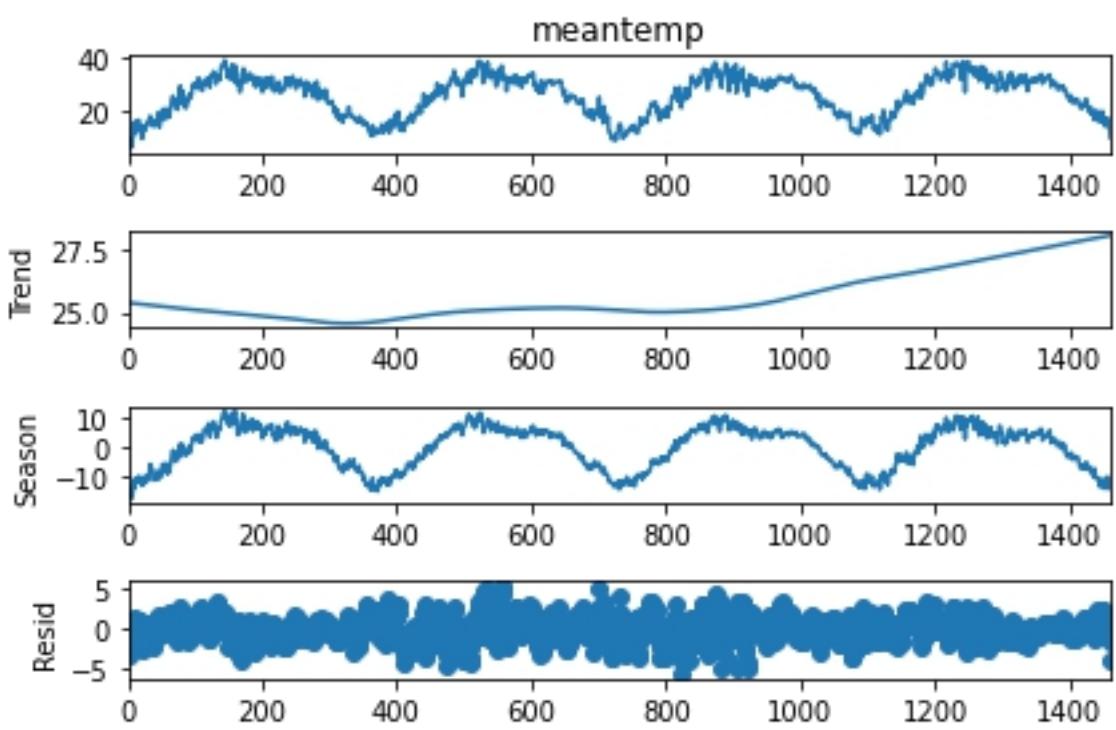
$$p = B \left(\frac{|R_v|}{6 * median(|R_v|)} \right)$$

$$B(x) = \begin{cases} (1 - x)^2, & |x| < 1 \\ 0, & |x| \geq 1 \end{cases}$$





```
from statsmodels.tsa.seasonal import STL  
stl_model = STL(df1['meantemp'], period = 365).fit()  
stl_model.plot()
```



Методы долгосрочного и краткосрочного
прогнозирования. Динамические модели и
механизм авторегрессии

Методы прогнозирования временных рядов

Методы прогнозирования рядов подразделяются на несколько видов:

- 1) **Модели от времени** (экстраполяция на время)
- 2) **Модели авторегрессии**
- 3) **Гибридные модели**

Наибольшее распространение в сфере автоматического прогнозирования получили методы построения авторегрессионных моделей ввиду простоты построения обучающего множества и отсутствия привязки к отдельным предикторам

Модели от времени

Модели прогнозирования по времени имеют некоторые общие подходы к построению:

Дискретный ряд данных y_i рассматривается как некоторый непрерывный во времени процесс, зависящий от времени, а также нескольких других составляющих, таких как тренд, сезонность, случайные ошибки и возможно экзогенные переменные.

За счёт исследования временного ряда на сезонность имеется возможность за счёт техник разделения ряда на тренд и сезонность оценить будущую тенденцию развития процесса.

Аддитивные и мультипликативные модели

Аддитивные модели:

$$y(t) = T(t) + S(t) + \varepsilon(t),$$

Мультипликативные модели:

$$y(t) = T(t) * S(t) * \varepsilon(t),$$

Переход к аддитивной модели:

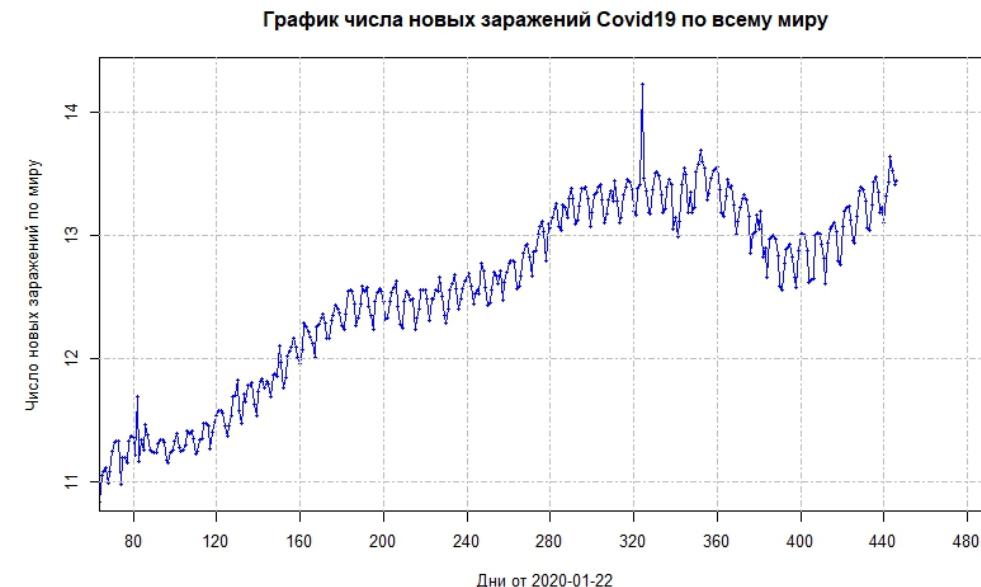
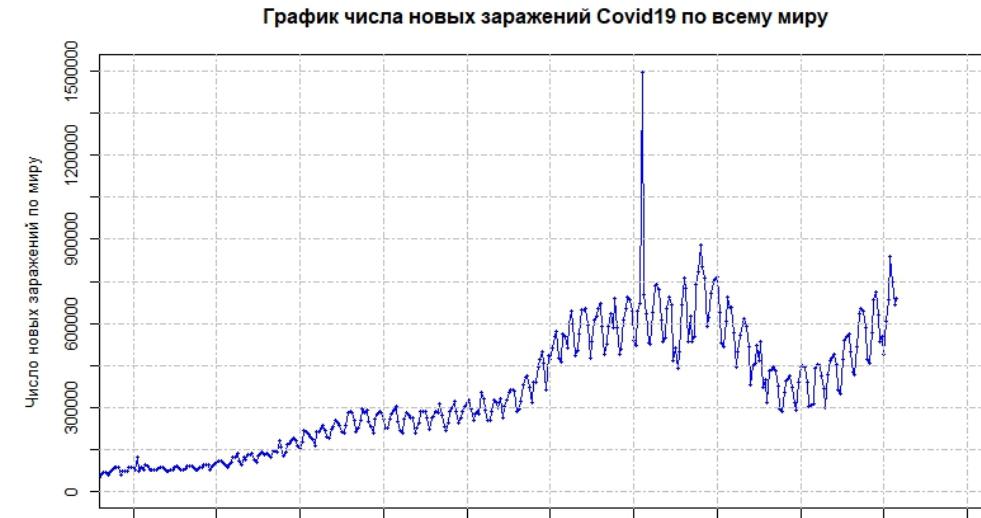
$$\log(y(t)) = \log(T(t) * S(t) * \varepsilon(t))$$

$$\log(y(t)) = \log(T(t)) + \log(S(t)) + \log(\varepsilon(t))$$

Переход к мультипликативной модели:

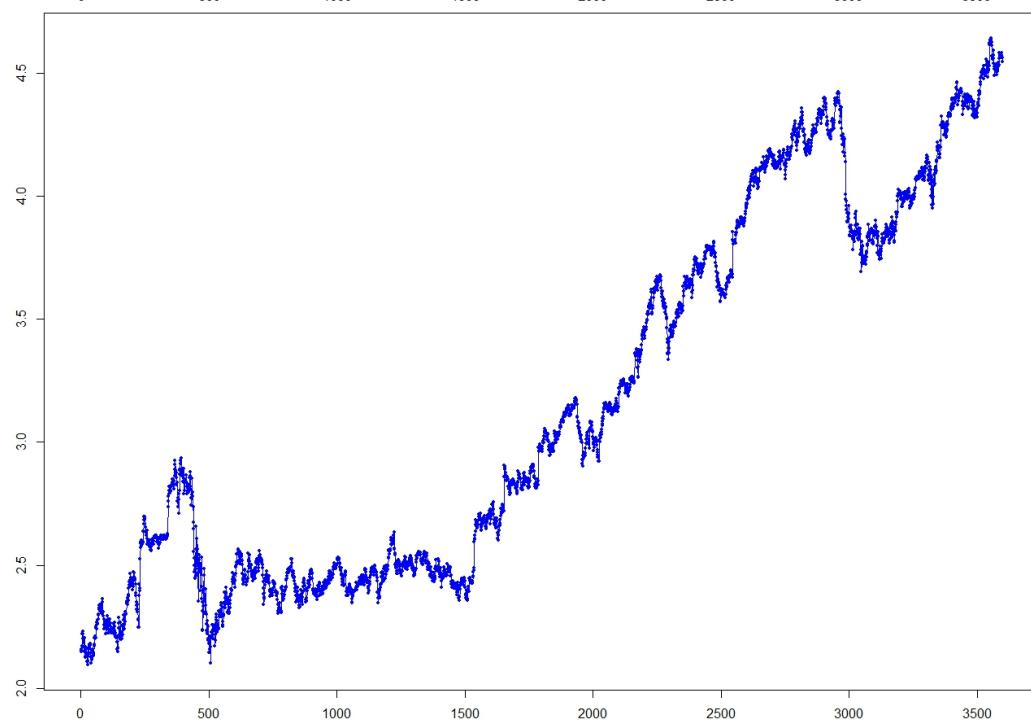
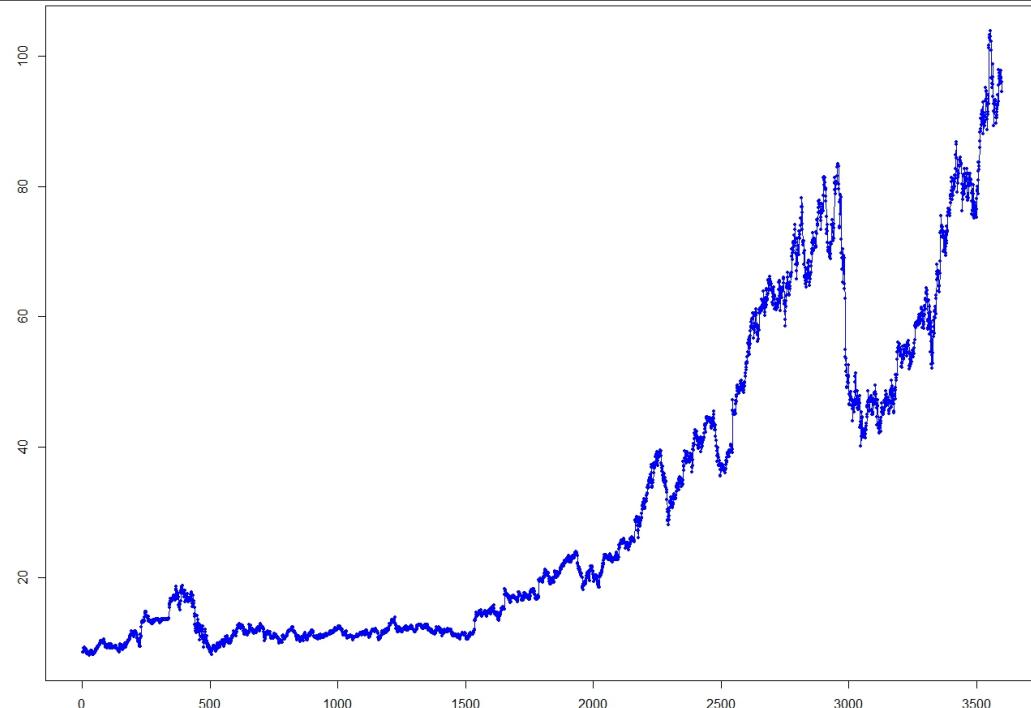
$$\exp(\log(y(t)))$$

$$= \exp(\log(T(t)) + \log(S(t)) + \log(\varepsilon(t)))$$

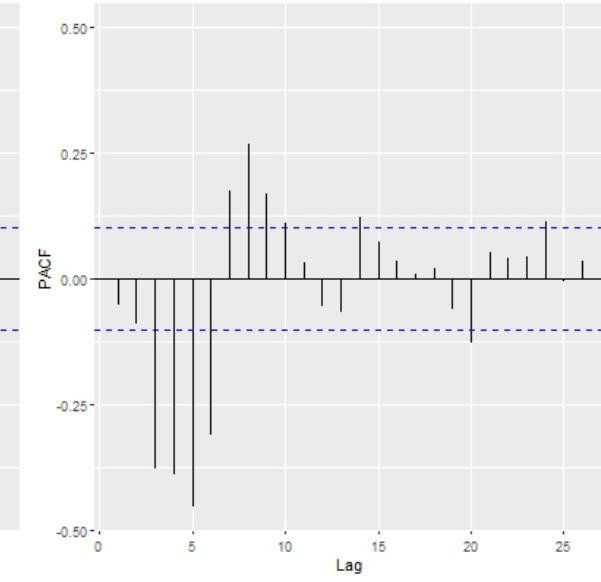
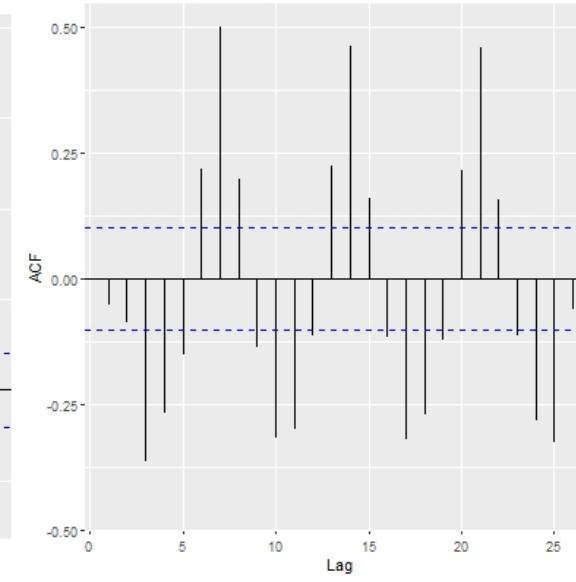
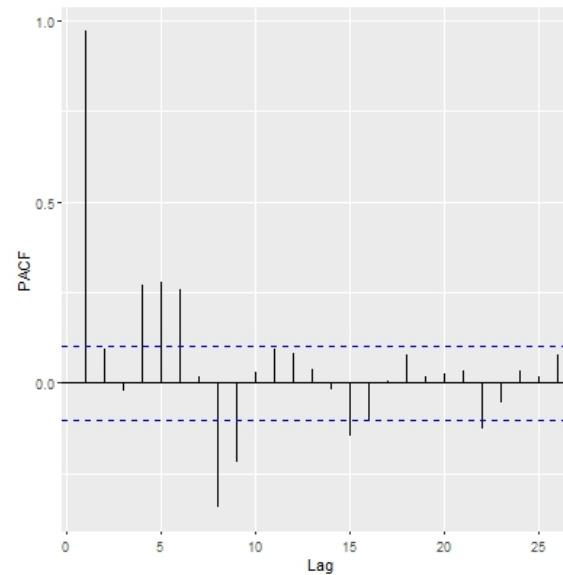
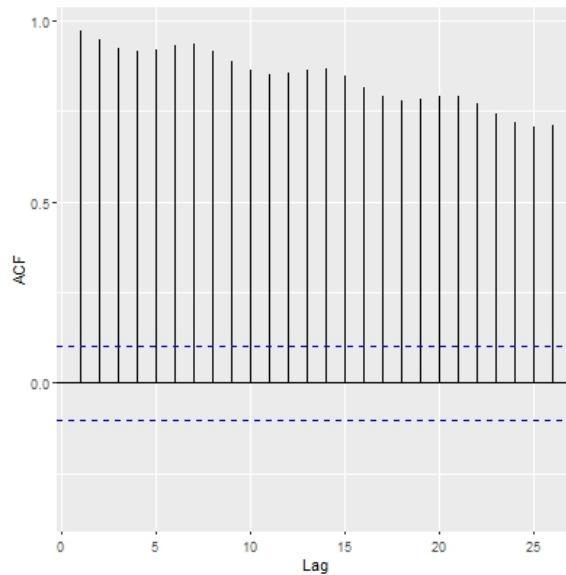
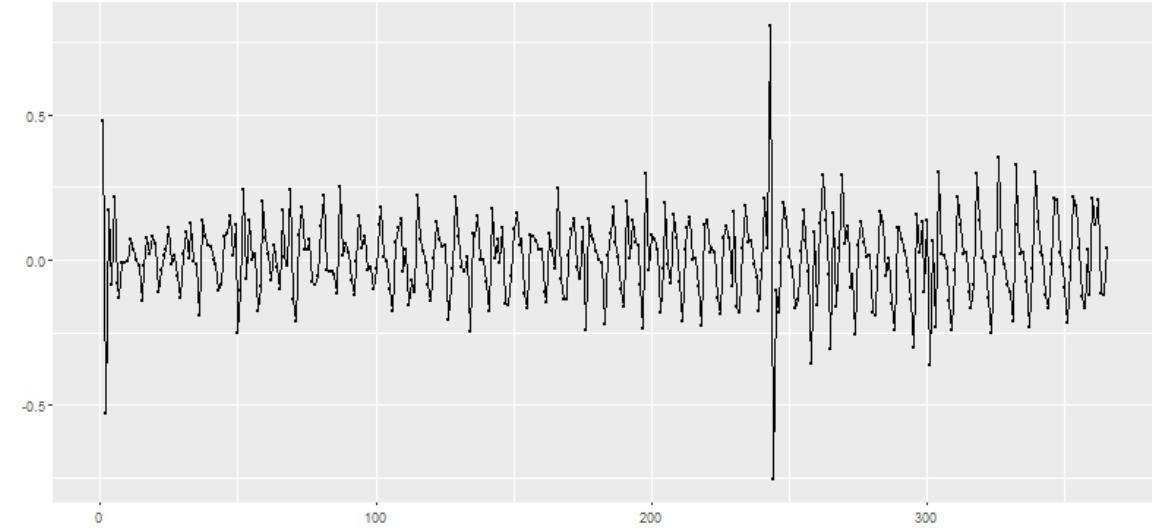
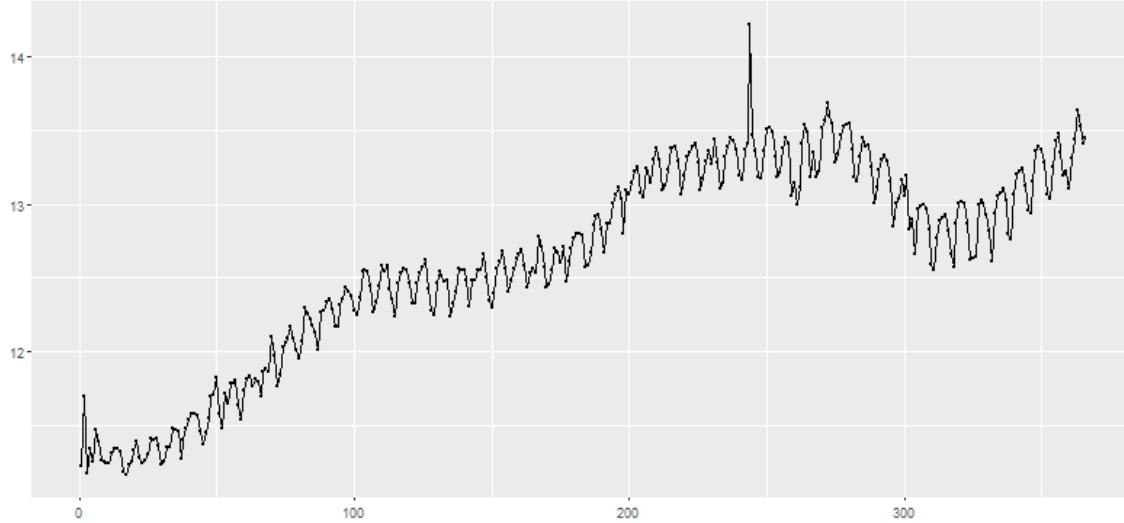


Преобразование Бокса-Кокса

$$Y^\gamma = \begin{cases} \frac{(Y^\gamma - 1)}{\gamma}, & \gamma \neq 0 \\ \log(Y), & \gamma = 0 \end{cases}$$



Выделение сезонной компоненты



STL разложение ряда по известной сезонности

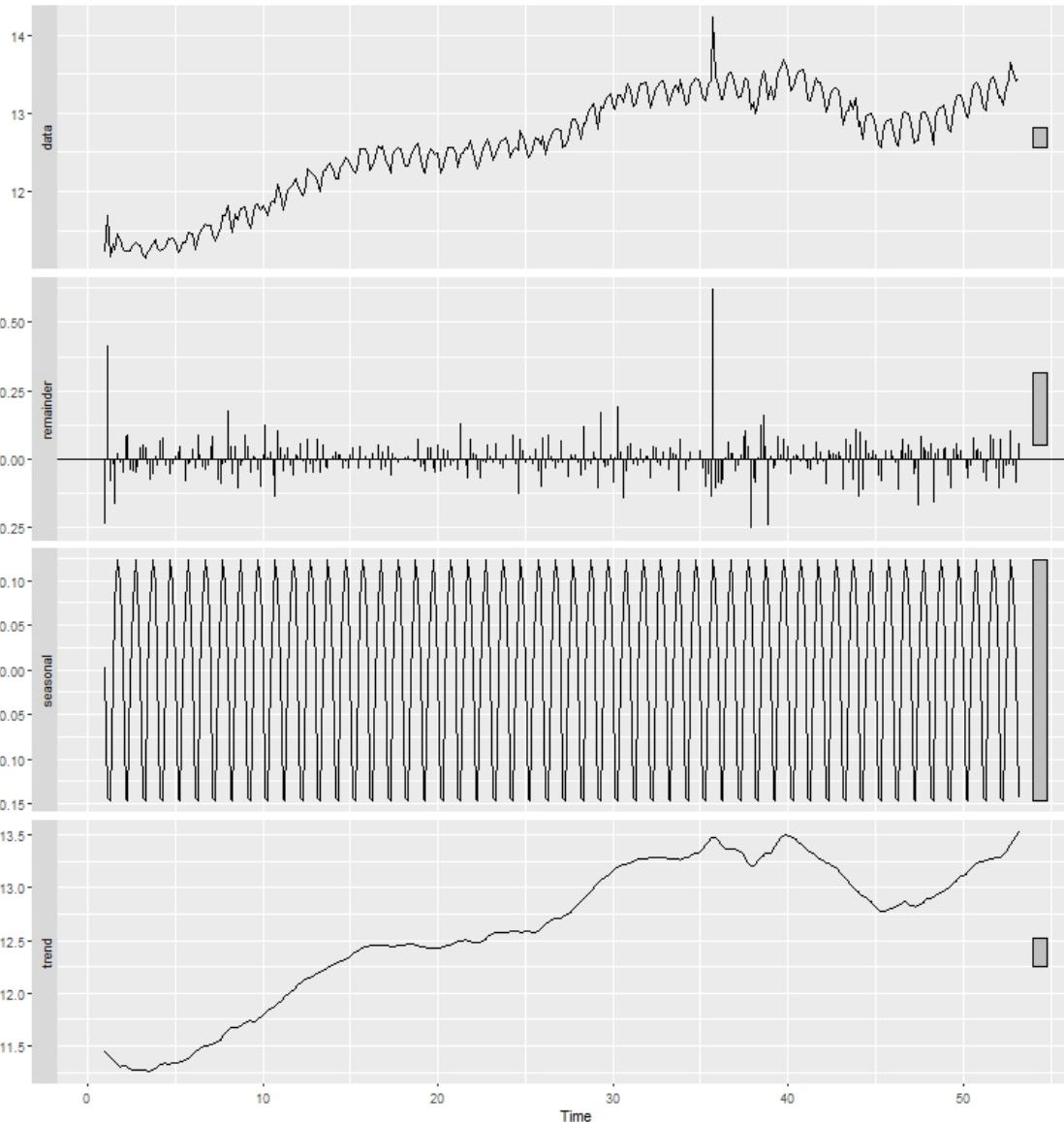
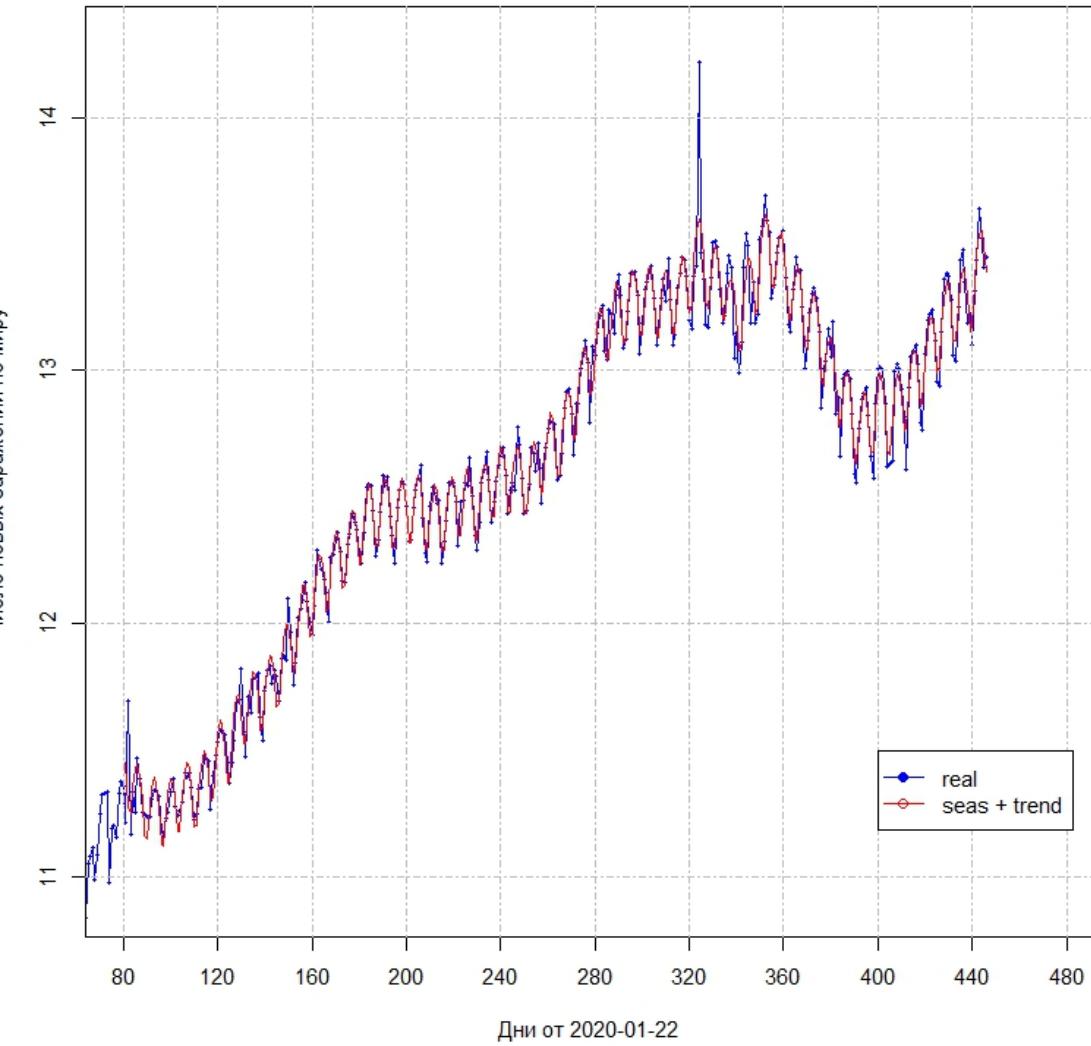


График числа новых заражений Covid19 по всему миру



Прогнозирование на основе тренда и сезонности

График числа новых заражений Covid19 по всему миру

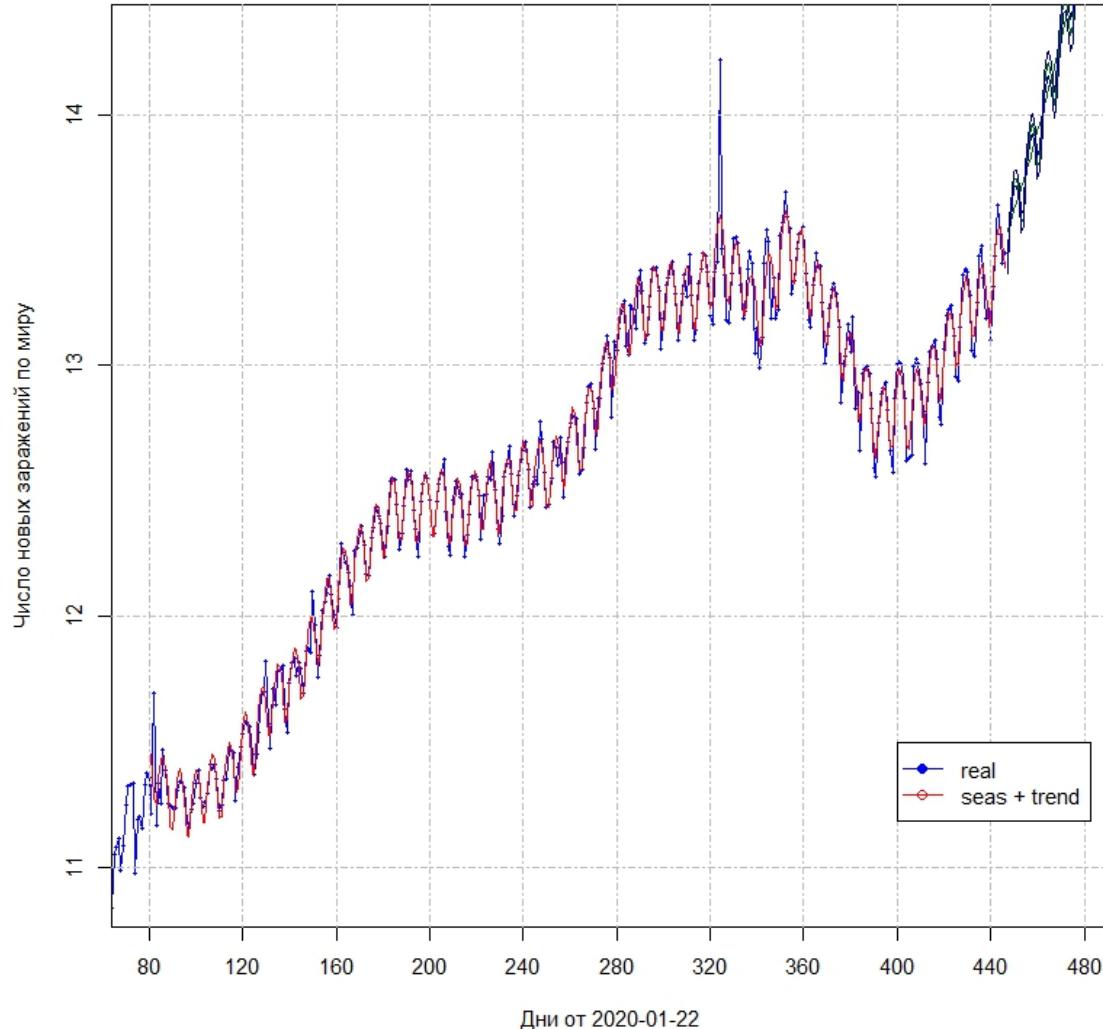
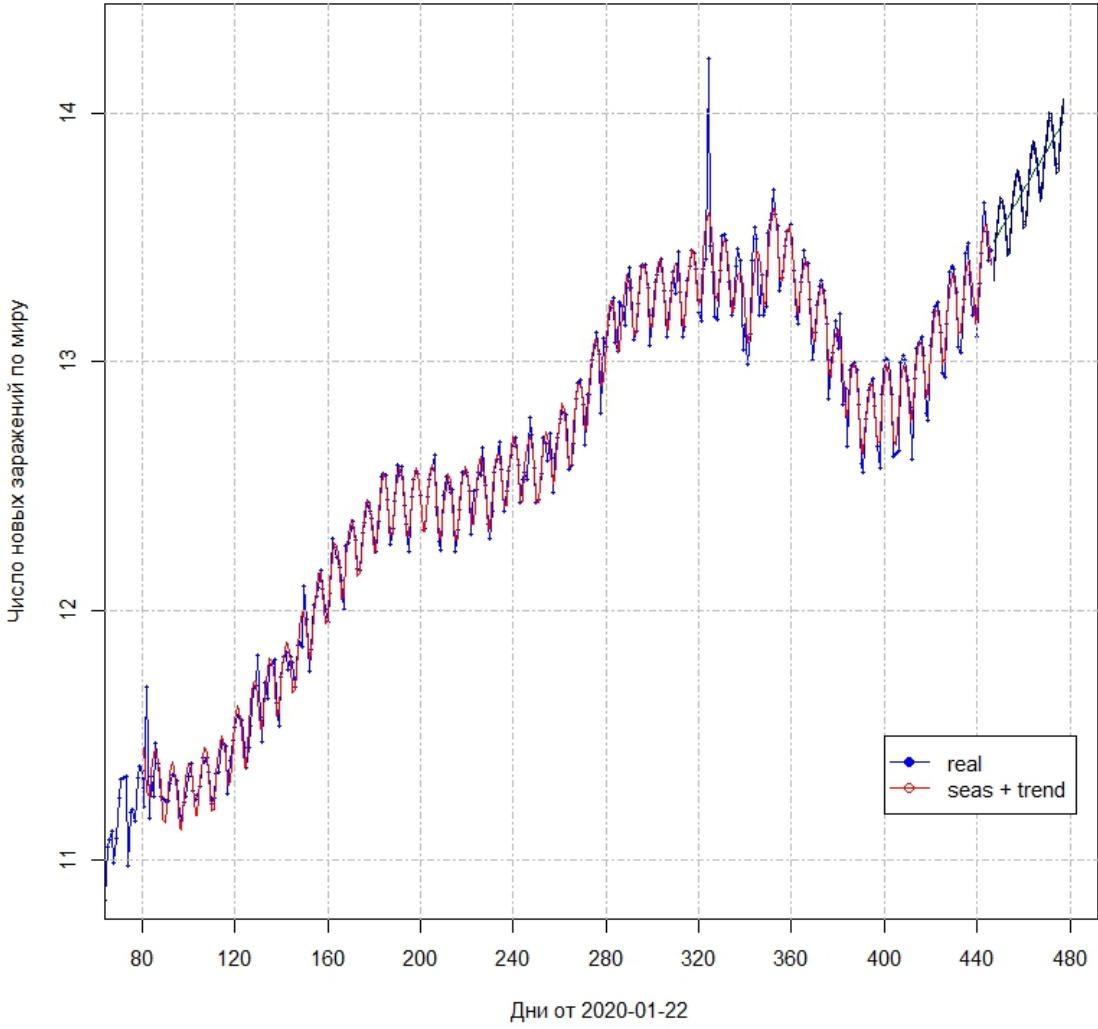


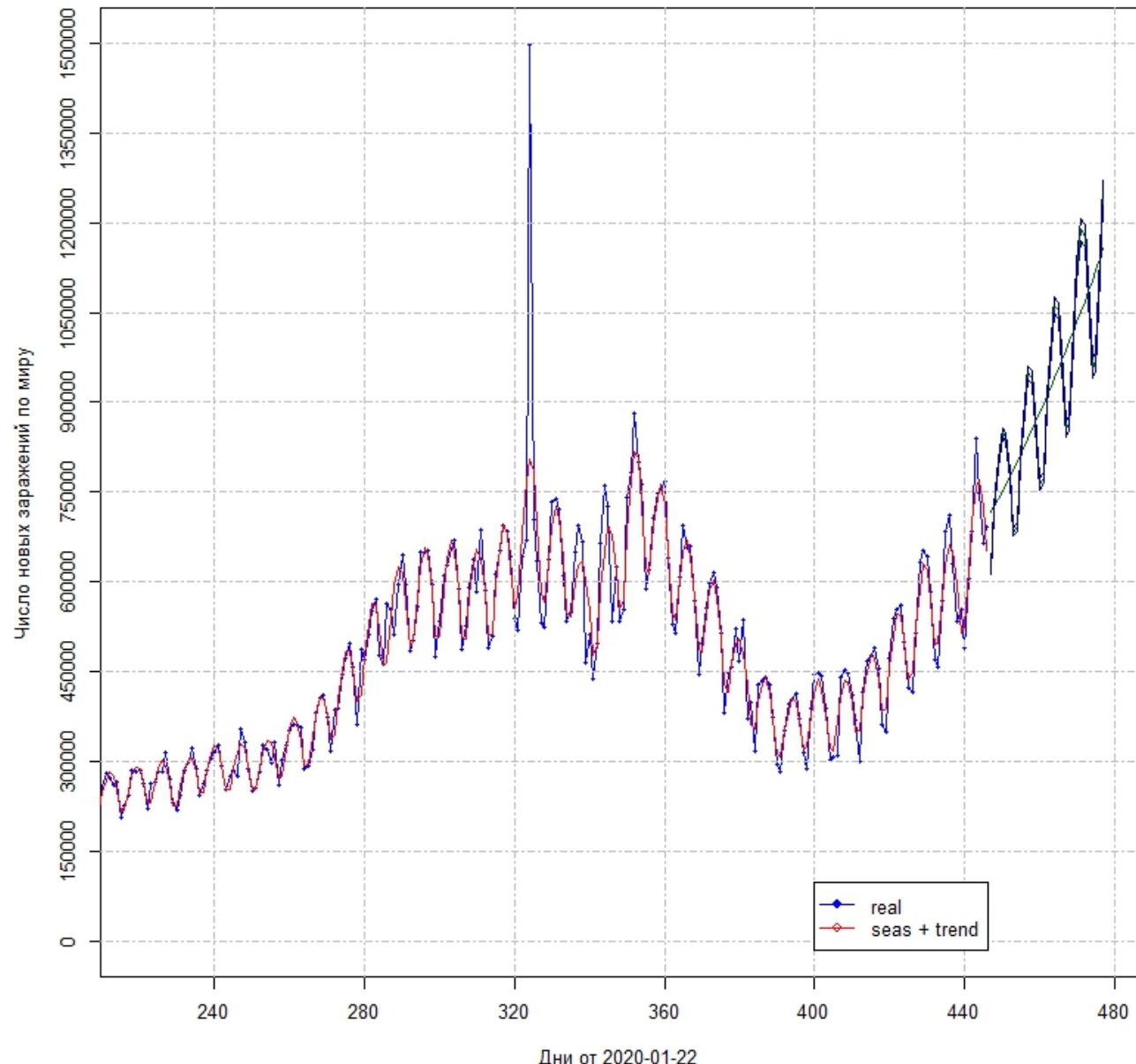
График числа новых заражений Covid19 по всему миру



Возврат к экспонентам

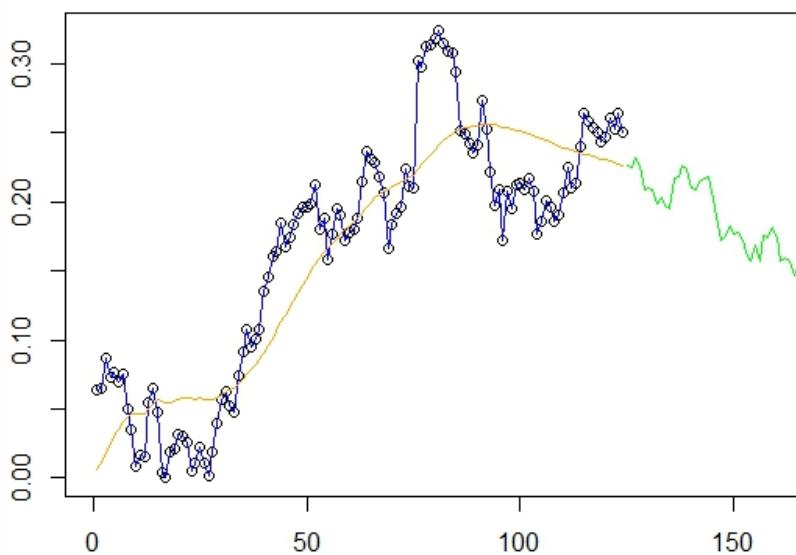
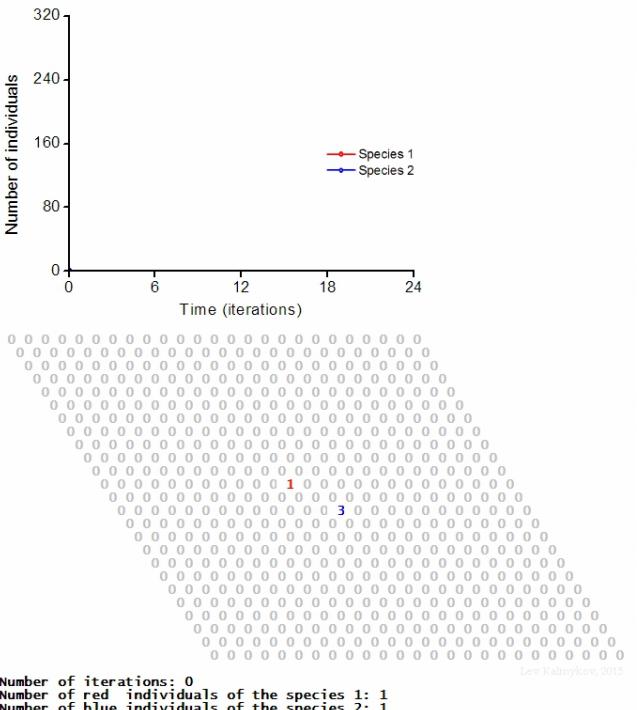
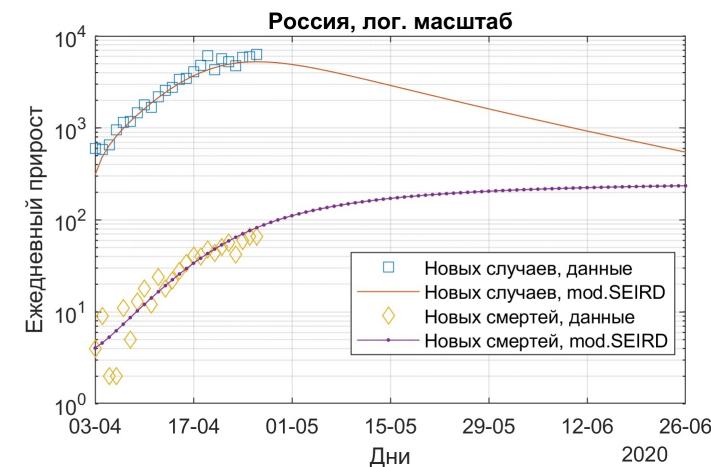
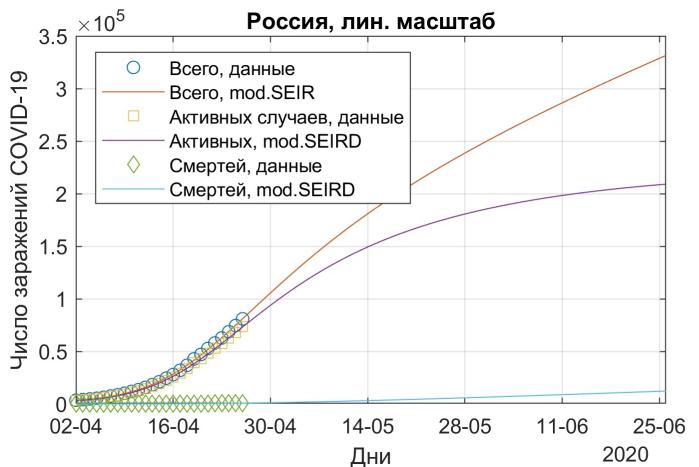
При экспоненцировании
полученной аддитивной модели
получим оценки прогноза
развития будущей тенденции в
динамике процесса

График числа новых заражений Covid19 по всему миру

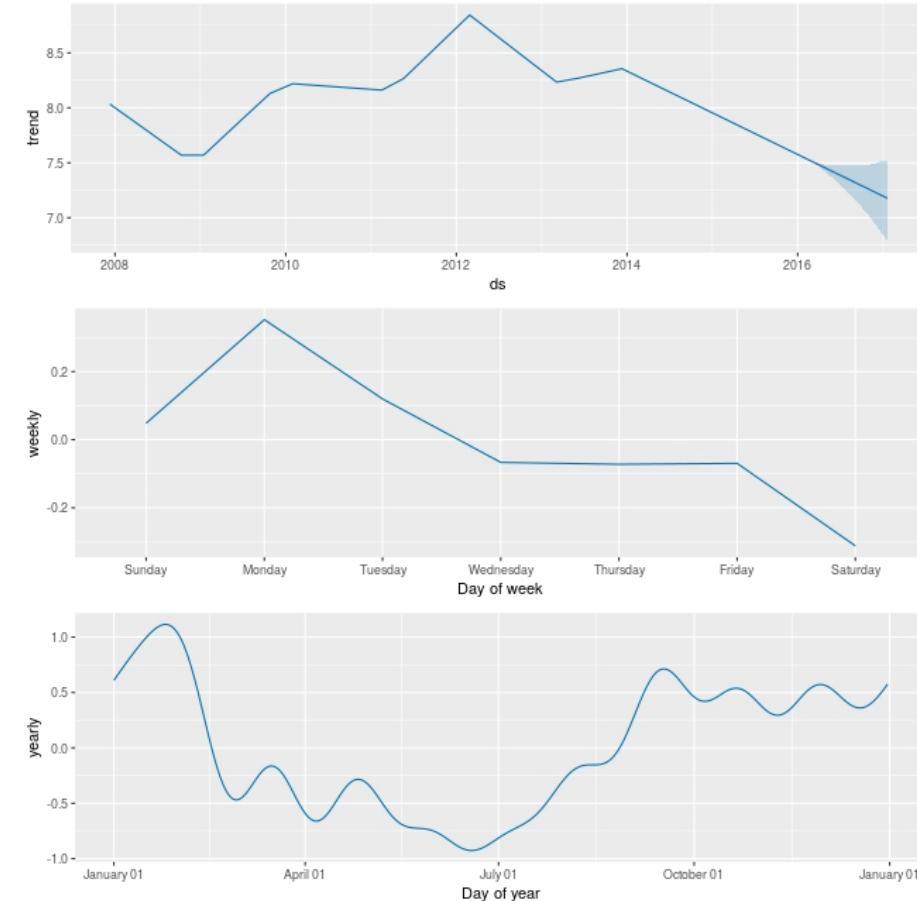
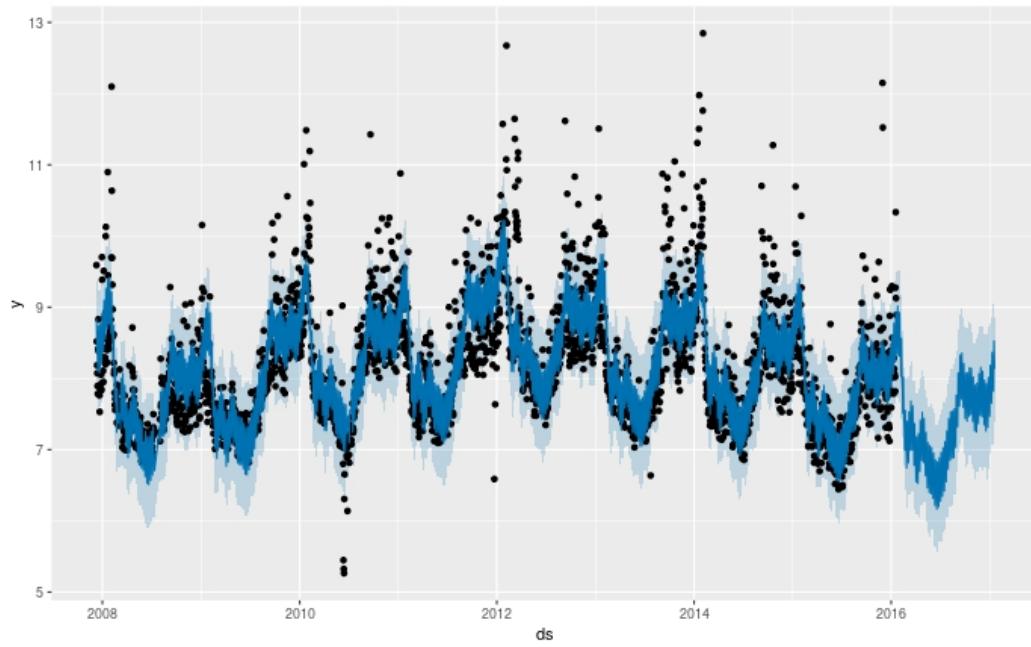


Примеры моделей тренда

1. Модель популяции (модель Ферхюста)
2. Модель Гомперца
3. Экспоненциальная модель
4. Линейная модель
5. Дифференциальные модели природы процесса



FacebookProphet



Модель авторегрессии

Пусть имеется исходный временной ряд $y(t_i)$ с моментами отсчётов $i = 0, 1, \dots, N$; тогда моделью авторегрессии для прогнозирования данного временного ряда является соотношение:

$$\hat{y}(t_{i+d}) = f(y(t_{i-1}), y(t_{i-2}), y(t_{i-3}), \dots, y(t_{i-tw}), \theta),$$

где d - количество отсчётов вперёд, которое мы хотим пропустить;
 tw - окно авторегрессионной зависимости,
 θ - множество параметров модели f .

Сэмплирование для авторегрессии

Сэмплирование производится скользящим окном с шириной авторегрессионной зависимости.
В данном случае пространство признаков для предсказываемой величины будет иметь размерность равную ширине окна t_w .

t	y
t_1	y_1
t_2	y_2
t_3	y_3
t_N	y_N



y_{-tw}	y_{-tw+1}	...	y_{-1}	y
y_1	y_2	...	y_{tw}	y_{tw+1}
y_2	y_3	...	y_{tw+1}	y_{tw+2}
y_3	y_4	...	y_{tw+2}	y_{tw+3}
...
y_{N-tw}	y_{N-tw+1}	...	y_{N-1}	y_N

Рассчёт обновления весов для модели авторегрессии

$X = (\vec{y}_{-tw}, \vec{y}_{-tw+1}, \dots, \vec{y}_{-1})$ - матрица

предикторов модели \hat{y}

$\vec{y}_{-tw}, \vec{y}_{-tw+1}, \dots, \vec{y}_{-1}$ - вектор-столбцы

матрицы, образованные по схеме

скользящего окна

X_i - вектор-строка матрицы предикторов

b - размер подвыборки на котором

считаем один градиент Loss-функции

y_i - истинная зависимость

\hat{y}_i - аппроксимация обучающего набора моделью
авторегрессии

$$\hat{y}_i = \hat{y}(X_i, W),$$

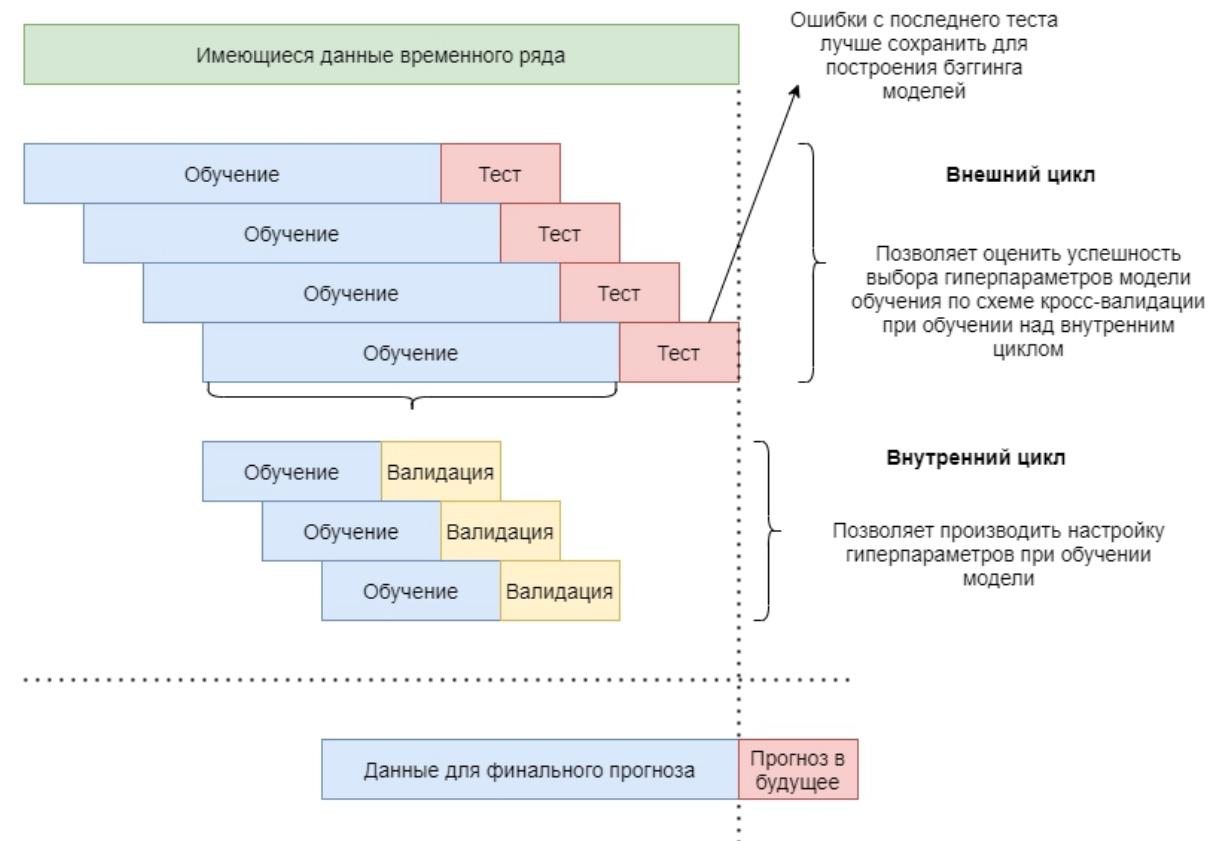
$$MSE_Loss(y, \hat{y}) = \frac{1}{b} * \sum_{i=1}^b (y_i - \hat{y}_i)^2,$$

$$\frac{\partial}{\partial W} MSE_Loss(y, \hat{y}) =$$

$$= -\frac{2}{b} * \sum_{i=1}^b \left[(y_i - \hat{y}_i) * \frac{\partial \hat{y}(X_i, W)}{\partial W} \right]$$

Обучение и кросс-валидация

Из-за имеющейся зависимости от порядка следования данных для всех моделей временного ряда вне зависимости от класса применяется следующая схема обучения и валидации.



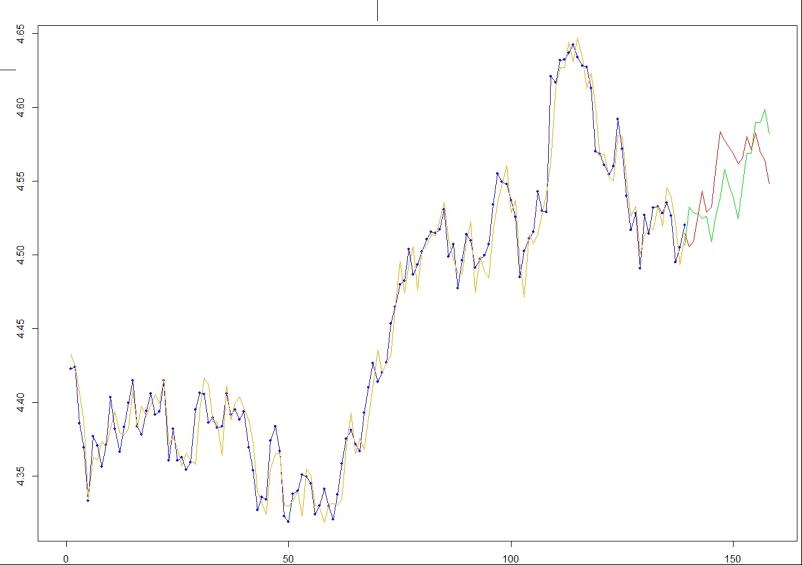
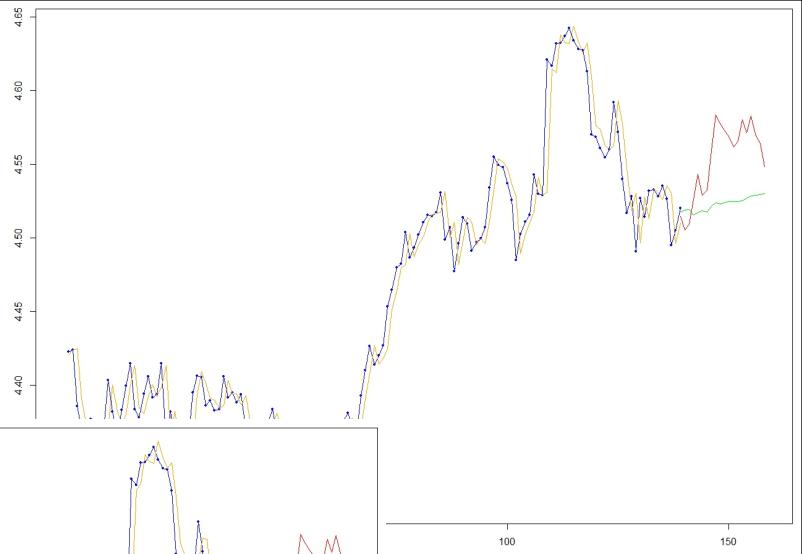
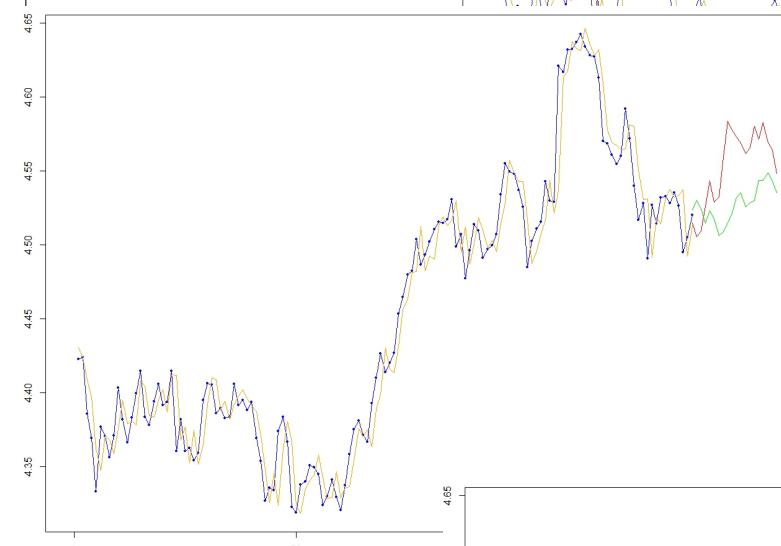
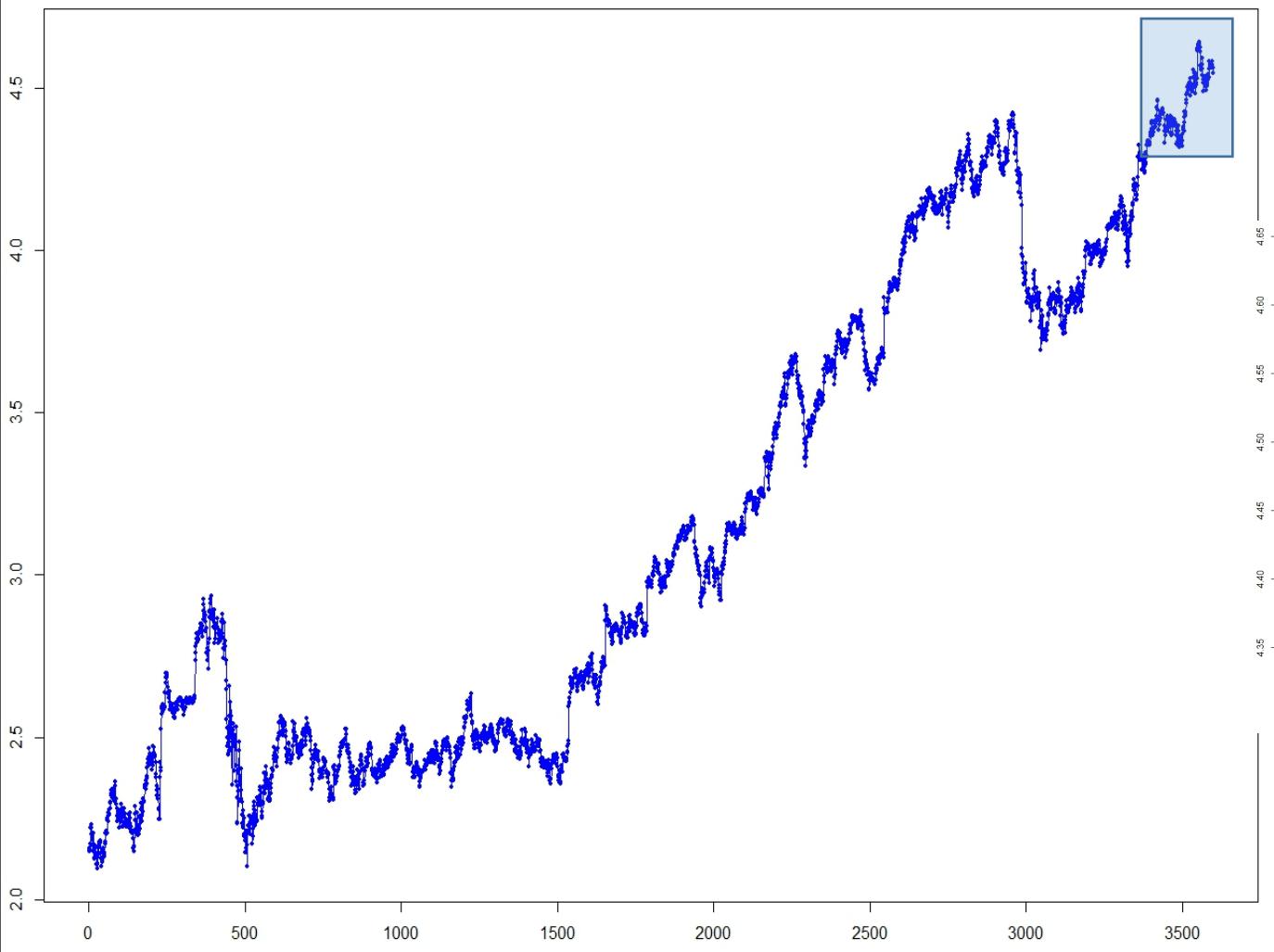
Линейная модель авторегрессии

Пусть стоит задача спрогнозировать величину y_{t+d} на d интервалов вперёд, тогда из предположения о сохранении зависимостей между предыдущими и следующими членами ряда определим следующую модель

$$\hat{y}_{t+d} = \alpha_1 y_t + \alpha_2 y_{t-1} + \dots + \alpha_{1+\tau} y_{t-\tau} + \varepsilon_t$$

$\alpha_1, \alpha_2, \dots, \alpha_{1+\tau}$ — коэффициенты линейной модели авторегрессии.

При $d = 1$, $\varepsilon_t \sim N(0, \delta)$ данная модель обозначается как $AR(\tau + 1)$ и является стандартной компоненты модели авторегрессии и скользящего среднего ARMA(p, q) для оценки будущей тенденции стационарных рядов



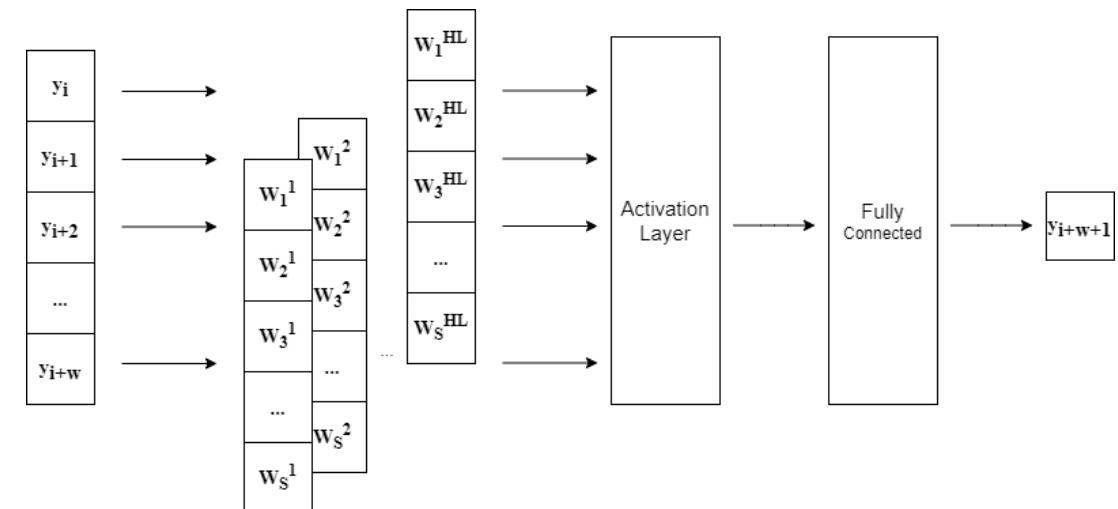
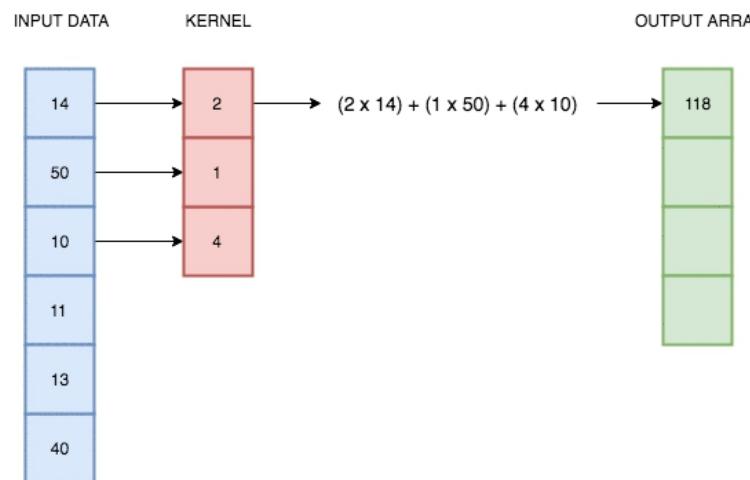
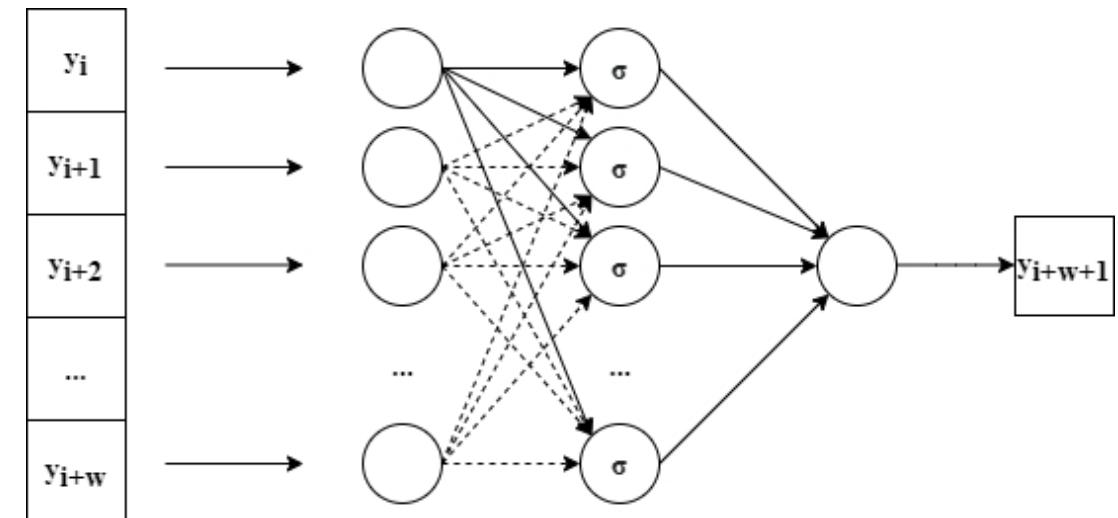
Архитектуры нейронных сетей для построения краткосрочного прогноза временного ряда

Полносвязная сеть и Conv1D

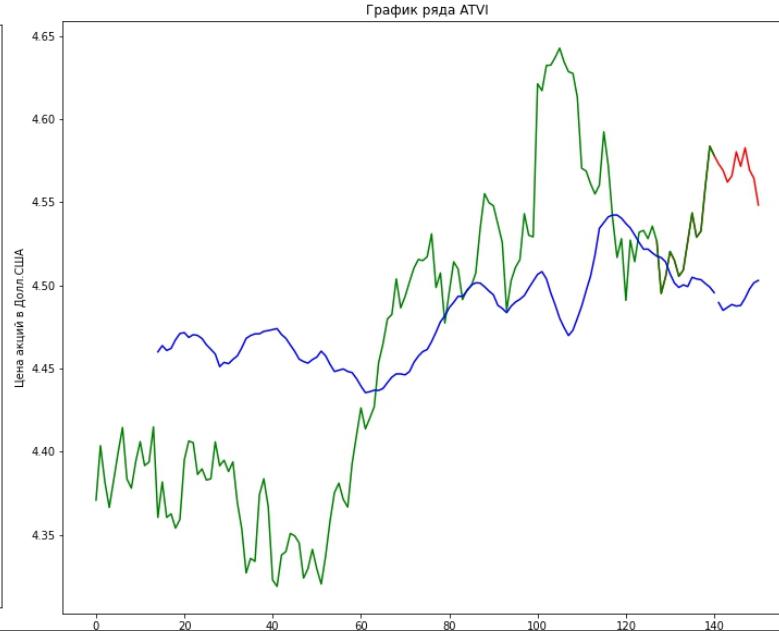
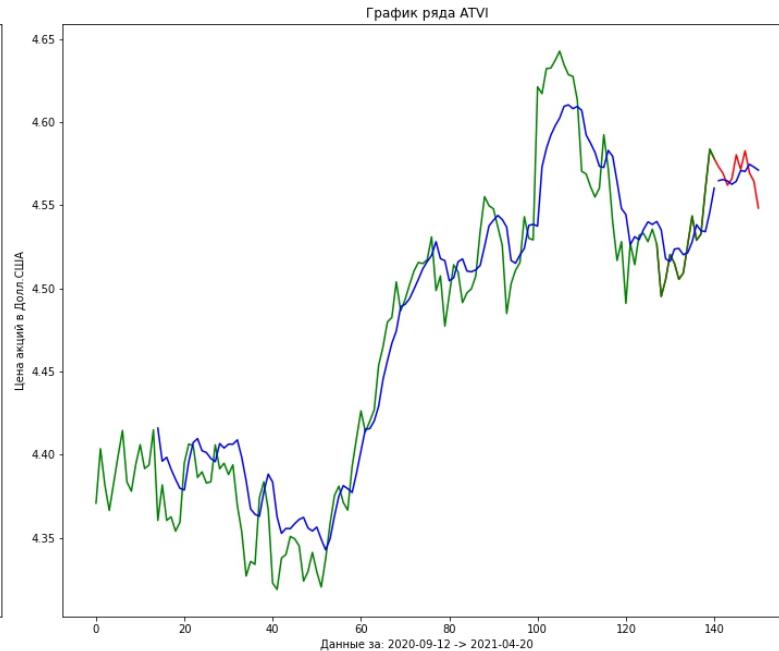
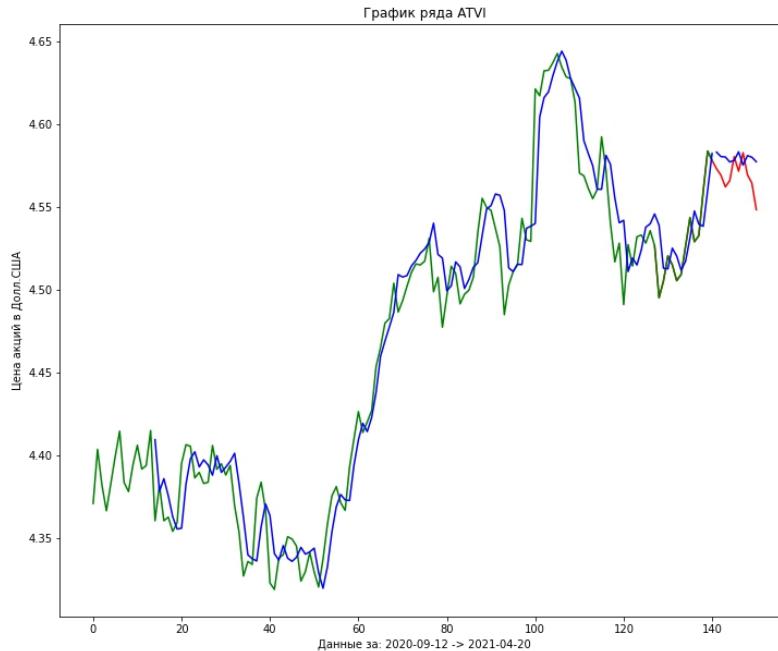
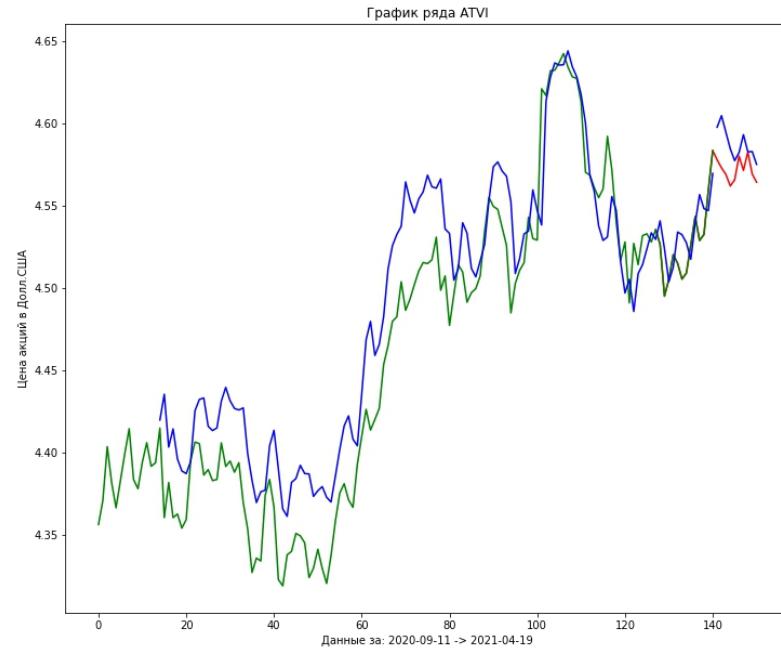
$$\hat{y}_{t+d} = \sum_{i=1}^{HL} \alpha_i * Activation(WX) + bias$$

$$\hat{y}_{t+d} = \sum_{i=1}^{HL} \alpha_i * Activation \left(\sum_{j=0}^w W_j^i * y_j + b_i \right) + bias$$

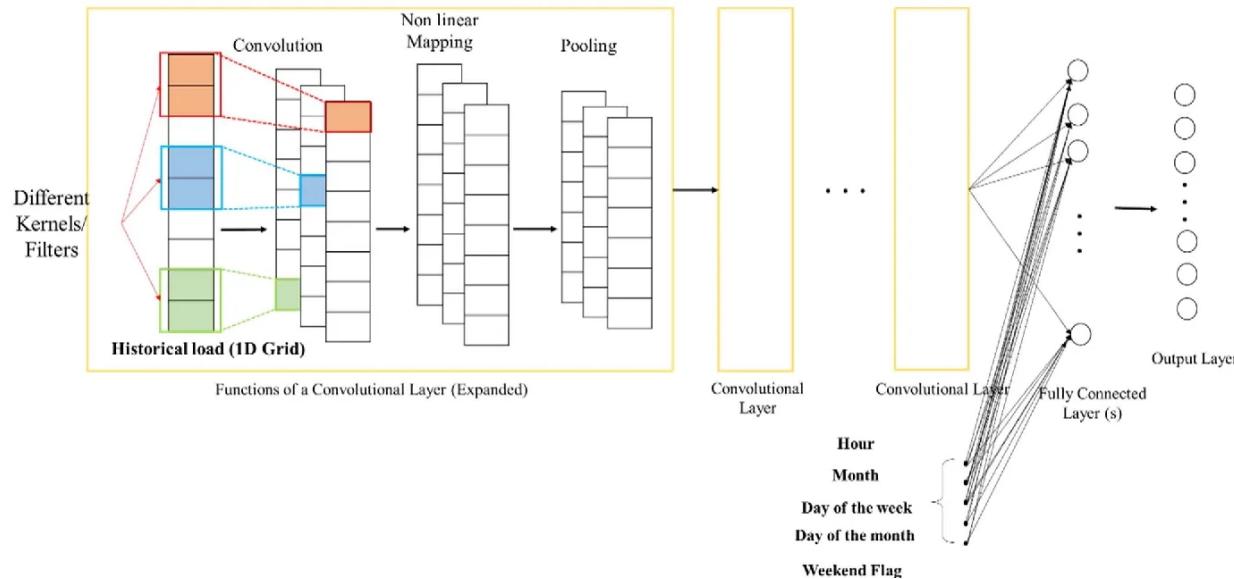
$$Conv1D_i = \sum_{j=0}^w W_j^i * y_j + b_i$$



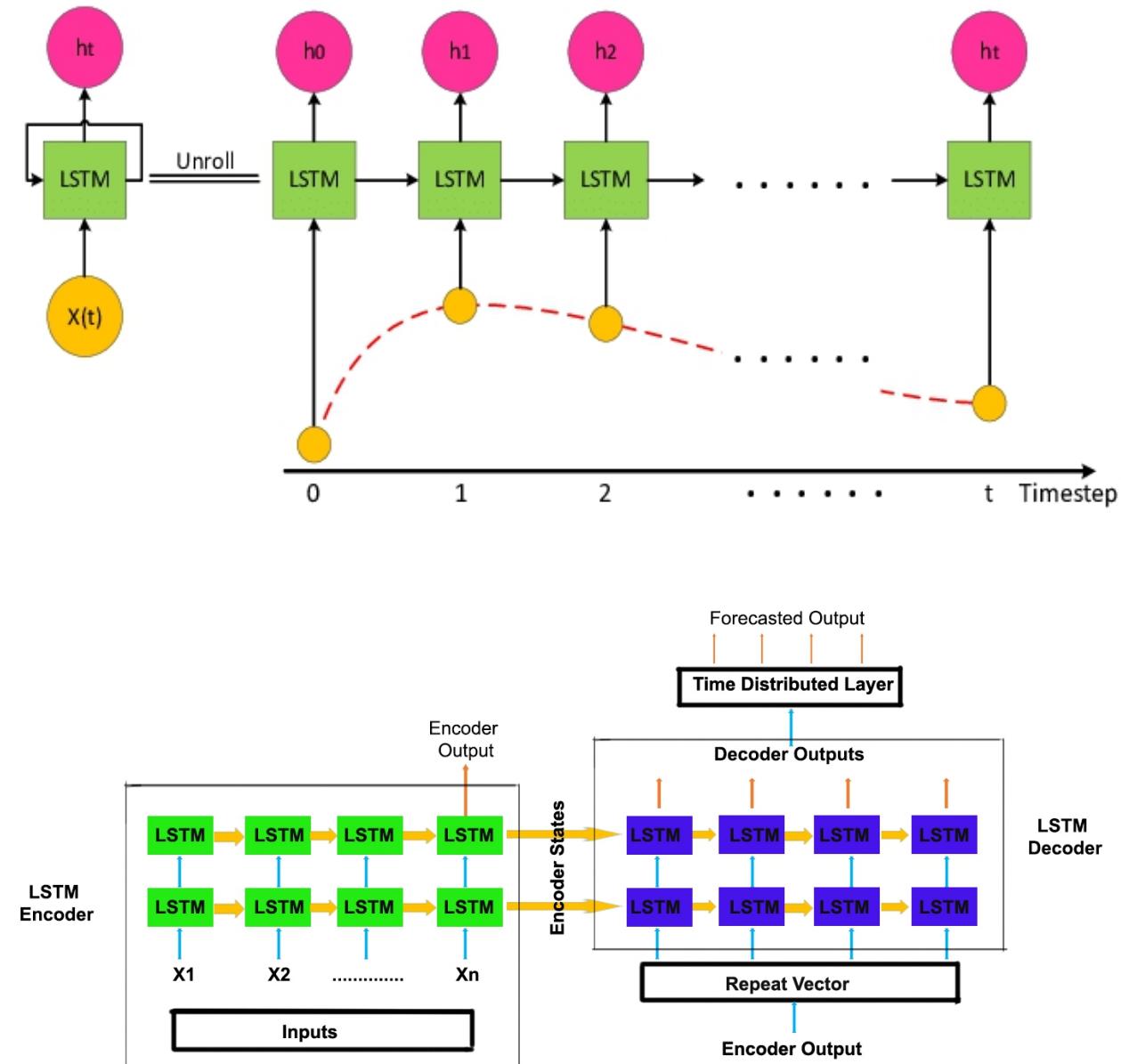
Аккуратнее с размером батча



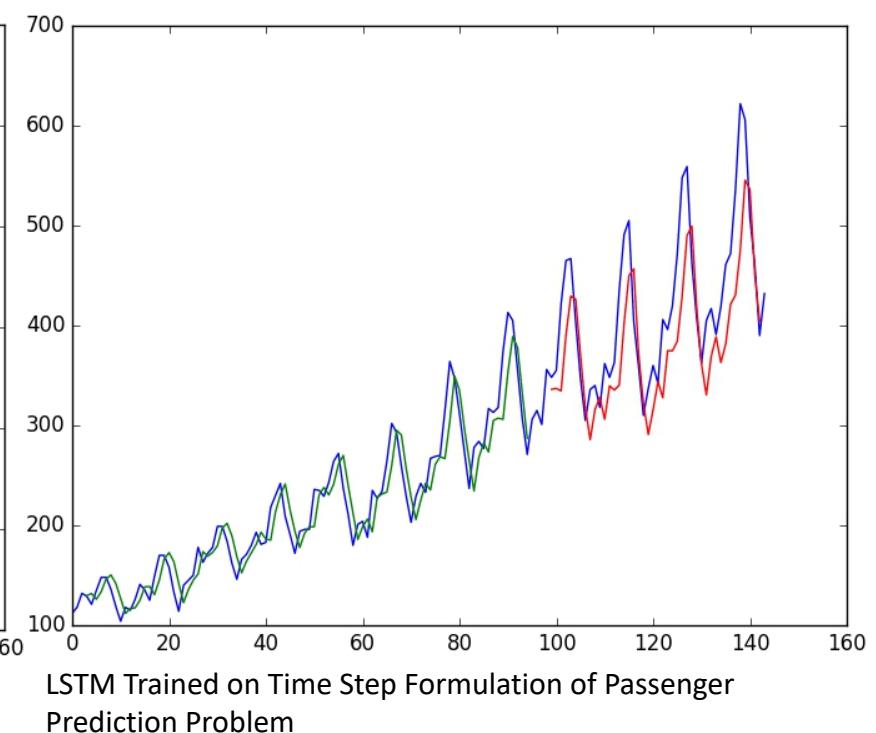
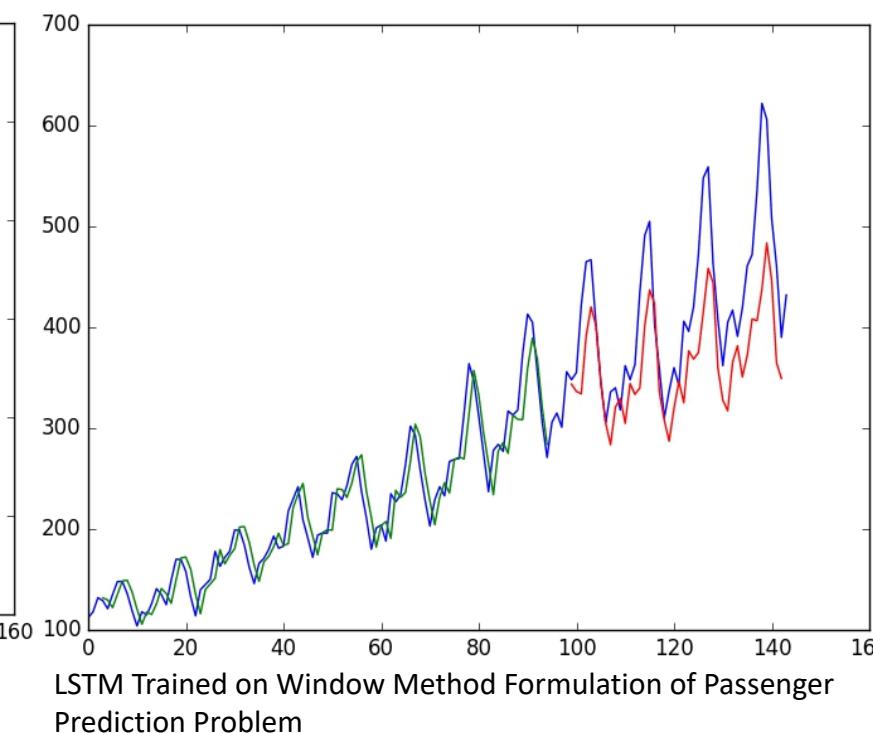
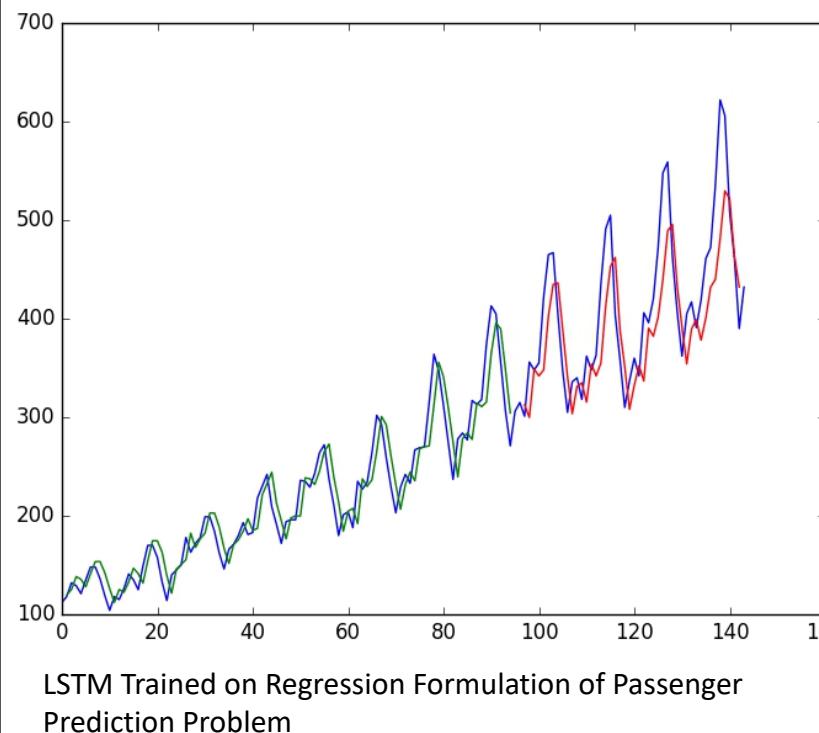
LSTM, Convlutions

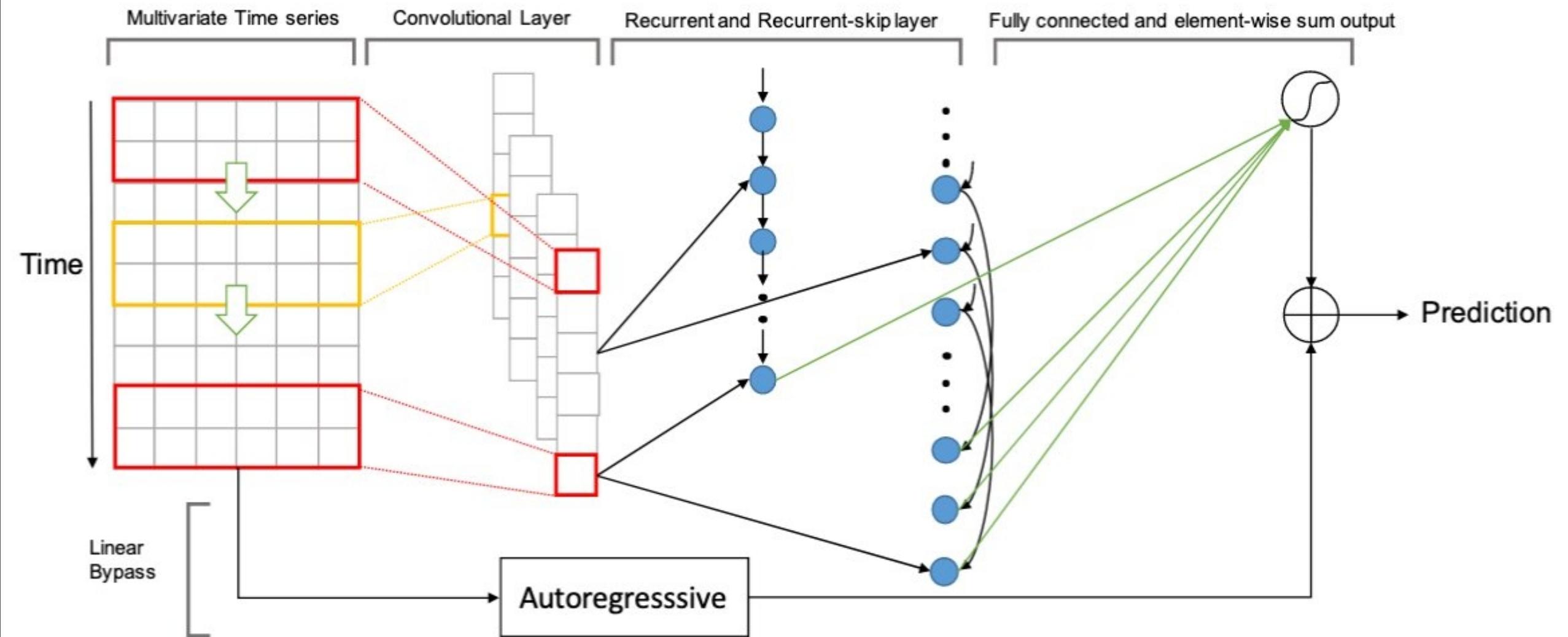


Multivariate Multi-step Time Series Forecasting
using Stacked LSTM sequence to sequence
Autoencoder



<https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/>





Ансамблирование нейронных сетей для прогнозирования временных рядов

Композиция без обучения

Сохраняя ошибки прогноза моделей на последних тестовых данных можно построить лёгкую модель прогнозирования на основе нескольких моделей

ε_{test_i} - ошибка i -ой модели на последних тестовых данных

Бэггинг по валидации

Оценка весов голосования на основе оценок ошибок на тестовой выборке последних известных данных



Композиция моделей

$$\varepsilon_{coeff_i} = \frac{1}{\sum_{i=1}^N \left(\frac{1}{Test_error_i} \right)},$$

$$\hat{y}_{future} = \sum_{i=1}^N (\varepsilon_{coeff_i} * \hat{y}_{model_i})$$

Бэггинг по валидации

Оценка весов голосования на основе оценок ошибок на тестовой выборке последних известных данных



Композиция моделей

$$\varepsilon_{coeff_i} = \frac{e^{\frac{1}{Test_error_i}}}{\sum_{i=1}^N \left(e^{\frac{1}{Test_error_i}} \right)},$$

$$\hat{y}_{future} = \sum_{i=1}^N (\varepsilon_{coeff_i} * \hat{y}_{model_i})$$

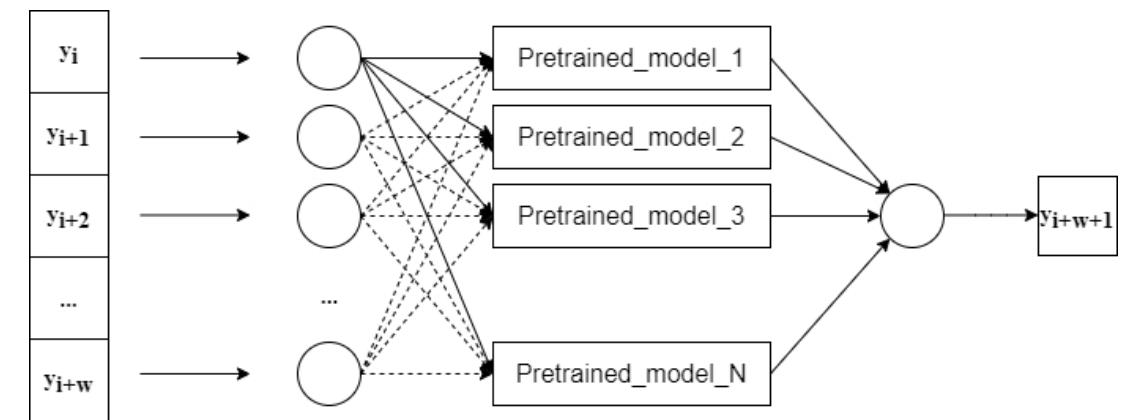
Бэггинг по валидации

Оценка весов голосования на основе оценок ошибок на тестовой выборке последних известных данных

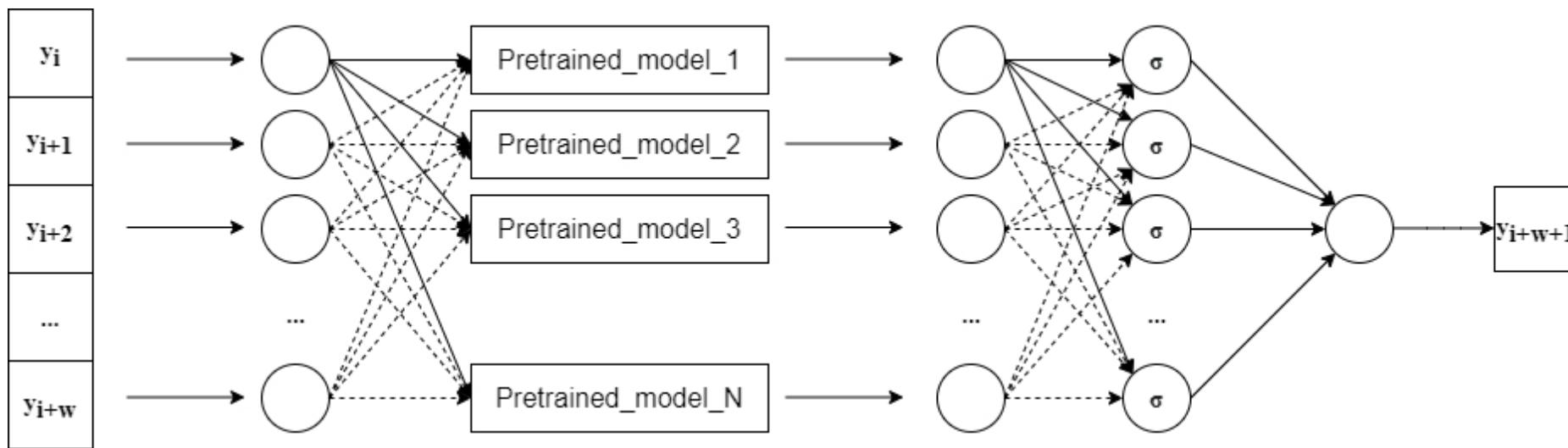


Бэггинг

Соединяя предобученные модели прогнозирования временных рядов линейным слоем с обучаемыми параметрами, при этом можно использовать как модели одинакового класса, так и разноклассовые модели



Стэкинг



Две простые модели

```
class reluLayerNet(nn.Module):
    def __init__(self, input_size, hidden_layer_size, output_size=1, p = 0.15):
        super().__init__()
        self.fc1 = nn.Linear(input_size, hidden_layer_size)
        self.activ1 = nn.ReLU()
        self.out = nn.Linear(hidden_layer_size, output_size)

        self.dropout = nn.Dropout(p = p)

    def forward(self, data):
        x = self.fc1(data)
        x = self.dropout(x)
        x = self.activ1(x)
        x = self.out(x)
        return x

    def predict(self, data):
        x = self.fc1(data)
        x = self.activ1(x)
        x = self.out(x)
        return x

class sigmaLayerNet(nn.Module):
    def __init__(self, input_size, hidden_layer_size, output_size=1, p = 0.15):
        super().__init__()
        self.fc1 = nn.Linear(input_size, hidden_layer_size)
        self.activ1 = nn.Sigmoid()
        self.out = nn.Linear(hidden_layer_size, output_size)

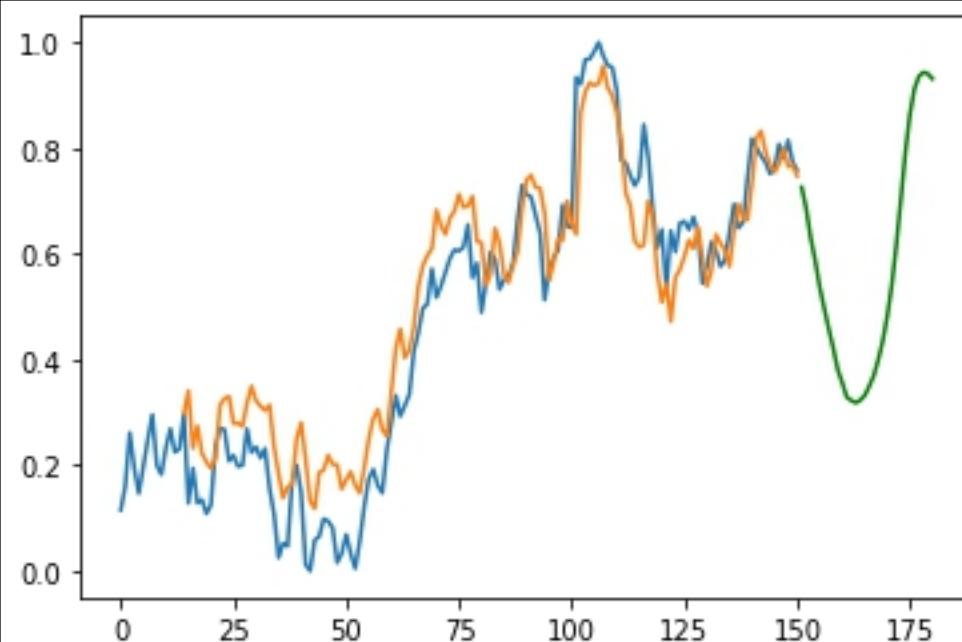
        self.dropout = nn.Dropout(p = p)

    def forward(self, data):
        x = self.fc1(data)
        x = self.dropout(x)
        x = self.activ1(x)
        x = self.out(x)
        return x

    def predict(self, data):
        x = self.fc1(data)
        x = self.activ1(x)
        x = self.out(x)
        return x
```

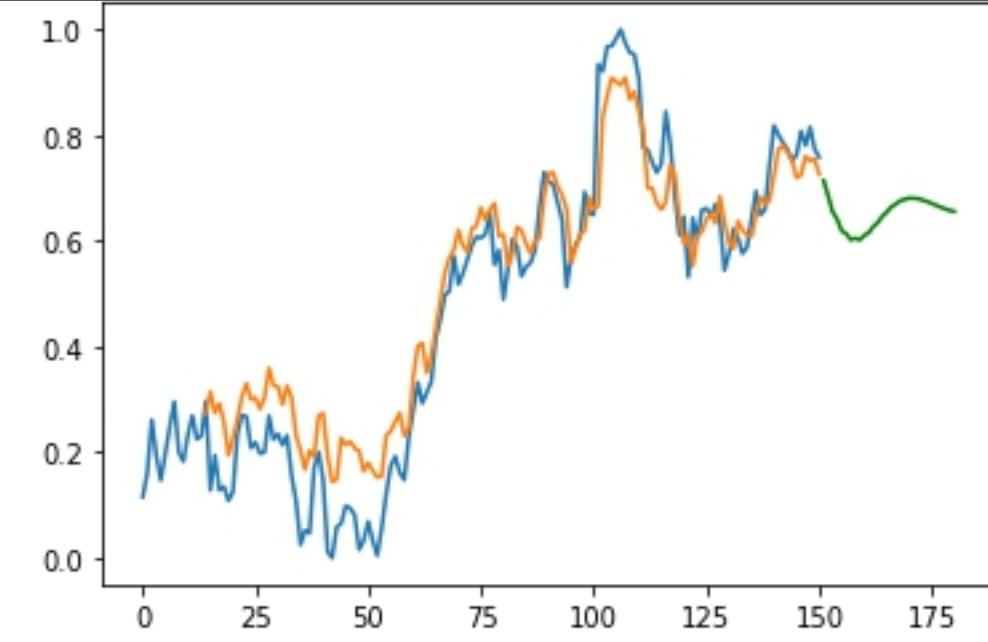
```
class StackModel(nn.Module):
    def __init__(self, input_size, model1, model2, output_size=1):
        super().__init__()
        self.MyList = nn.ModuleList([model1, model2])
        self.fc = nn.Linear(2, out_features=output_size)
        self.trace = []

    def forward(self, data):
        for i, l in enumerate(self.MyList):
            x = l(data)
            self.trace.append(list(x.detach()))
        x = torch.FloatTensor(np.array(self.trace).T)
        x = self.fc(x)
        self.trace = []
        return x
```

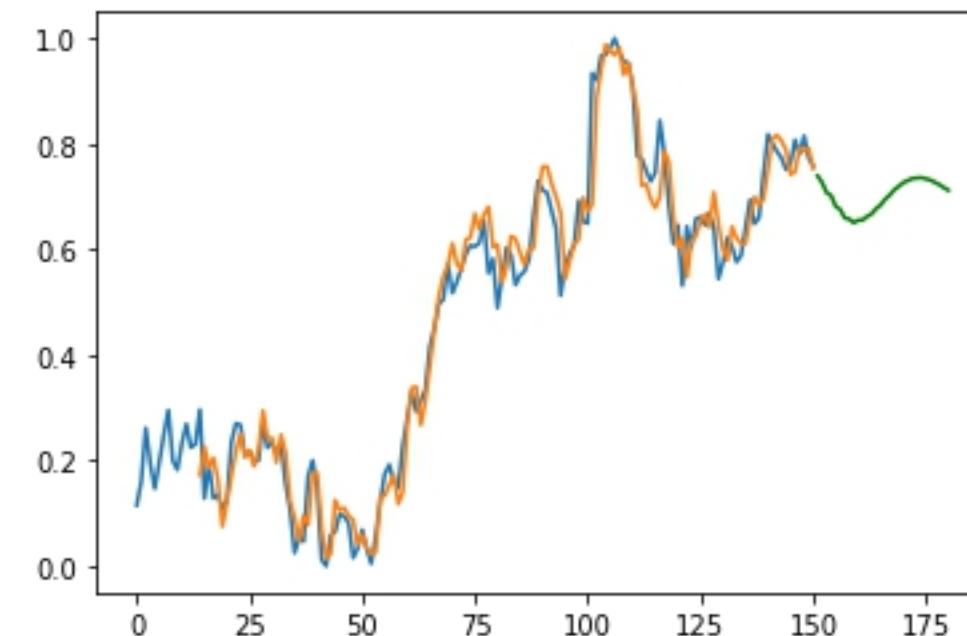


ActivisionBlizzard -
single Sigmoid(tw)

ActivisionBlizzard -
stacked Linear -
Sigmoid(tw) + ReLU(tw)



ActivisionBlizzard -
single ReLU(tw)



Ссылки

- Hyndman, R.J., & Athanasopoulos, G. (2018) Forecasting: principles and practice, 2nd edition, OTexts: Melbourne, Australia. OTexts.com/fpp2. Accessed on <21.04.2021>
- Рудаков К. В., Воронцов К. В., «О методах оптимизации и монотонной коррекции в алгебраическом подходе к проблеме распознавания», Докл. РАН, 367:3 (1999), 314—317 mathscinet zmath
- Воронцов К. В., «Комбинаторный подход к оценке качества обучаемых алгоритмов», Математические вопросы кибернетики, 13, ред. О. Б. Лупанов, Физматлит, М., 2004, 5-36 mathscinet;
- Воронцов К. В., Каневский Д. Ю., «Коэволюционный метод обучения алгоритмических композиций», Таврический вестник информатики и математики, 2005, № 2, 51-66 zmath;

Ссылки

Recurrent Neural Networks for Time Series [LSTM, GRU]

Forecasting <https://arxiv.org/pdf/1901.00069.pdf>

SigmoidLayer -

https://colab.research.google.com/drive/1BVJX8UkgkWtJgf0TgW2AHA-9_ZE2vo3k?usp=sharing

ReLUlayer -

<https://colab.research.google.com/drive/1tSgSQUg4IoZXW70Z4dHB00rUIGpdmx3S?usp=sharing>

StackModel - <https://colab.research.google.com/drive/1BInlhNUJ41hXM-Euza5C6M6fLVJHkLuJ?usp=sharing>