Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

README

Анализ и прогнозирование временного ряда

Составили: студенты 312-й группы

Альбатыров Алмаз,

Иванцов Глеб,

Сатвальдина Дана

Проверил: преподаватель Поспелова И.И

Москва 2020

Содержание

1 Постановка задачи

Задание: Провести анализ некоторого временного ряда и попробовать предсказать значения для последующих месяцев. В рамках данного задания необходимо:

- научиться правильно считывать данные и их визуализировать;
- понять как определять свойства временных рядов и познакомиться с различными моделями для предсказания значений;
- оценить качество полученных моделей.

Цель: Продолжить изучение языка **Python**, попутно познакомиться с такими библиотеками, как **Pandas**(работа с данными), **statsmodels**(работа с различными статистическими моделями, в том числе и временными рядами).

2 Теоретические сведения

Временной ряд (или ряд динамики) — собранный в разные моменты времени статистический материал о значении каких-либо параметров (в простейшем случае одного) исследуемого процесса Временные ряды бывают **одномерные** и **многомерные**. Первые содержат наблюдения за изменением только одного параметра исследуемого процесса или объекта, а вторые – за двумя или более параметрами.

Анализ временных рядов — совокупность математико-статистических методов анализа, предназначенных для выявления структуры временных рядов и для их прогнозирования.

Систематическая составляющая временного ряда является результатом воздействия постоянно действующих факторов.

Выделяют три основных систематических компоненты временного ряда:

- **Тренд** $T(t)$ - систематическая линейная или нелинейная компонента, изменяющаяся во времени;
- **Сезонность** $S(t)$ - периодические колебания уровней временного ряда внутри года;
- **Цикличность** $C(t)$ - периодические колебания, выходящие за рамки одного года. Промежуток времени между двумя соседними вершинами или впадинами в масштабах года определяют как длину цикла.

Систематические составляющие характеризуются тем, что они могут одновременно присутствовать во временном ряду.

Случайной составляющей $E(t)$ называется случайный шум или ошибка, которая воздействует на временной ряд нерегулярно.

Стационарный временной ряд (TS-ряд) - это такой случайный процесс, для которого математическое ожиданиие, дисперсия и ковариации между отдельными членами ряда случайно варьируют вокруг постоянного, не зависящего от времени уровня (так называемая "стационарность"в широком смысле, которая только и рассматривается для временных рядов): $m_x(t) = const$; $D_x(t) = const$, где $m_x(t)$ -

неслучайная функция, являющаяся математическим ожиданием процесса $X(t)$, значение которой в момент времени t равно математическому ожиданию множества реализаций в соответствующем сечении t; $D_x(t)$ - неслучайная функция, являющаяся дисперсией случайного процесса, значение которой также равно дисперсии реализаций сечения в каждый момент времени t. Другими словами, временной ряд **стационарен**, если порождающий его механизм не меняется при сдвиге во времени, а соответствующий случайный процесс достиг статистического равновесия.

В теории временных рядов чаще всего применяются два способа записи моделей временных рядов. Первый способ основывается на предположении, что влияние всех его компонент на значения элементов временного ряда носит аддитивный характер. В этом случае модель временного ряда называется **аддитивной** и имеет вид $x(t) = T(t) + S(t) + C(t) + E(t)$.

Второй способ записи модели основан на предположении о мультипликативном характере воздействия компонент временного ряда на x(t). В этом случае модель временного ряда называется **мультипликативной** и записывается в виде произведения: $x(t) = T(t) * S(t) * C(t) * E(t)$.

Временной X_t ряд называется **интегрированным порядка** k (обычно пишут $X_t \sim I(k)$), если разности ряда k-го порядка $\Delta^k\ x_t$ — являются стационарными, в то время как разности меньшего порядка (включая нулевого порядка, то есть сам временной ряд) не являются TS-рядами. В частности $I(0)$ - это стационарный процесс.

ARIMA (англ. autoregressive integrated moving average, иногда модель Бокса — Дженкинса, методология Бокса — Дженкинса) — интегрированная модель авторегрессии — скользящего среднего — модель и методология анализа временных рядов. Является расширением моделей **ARMA** для нестационарных временных рядов, которые можно сделать стационарными взятием разностей некоторого порядка от исходного временного ряда (так называемые интегрированные или разностно-стационарные временные ряды). Модель **ARIMA(p,d,q)** означает, что разности временного ряда порядка d подчиняются модели **ARMA(p,q)**.

Модель **ARIMA(p,d,q)** для нестационарного временного ряда X_t имеет вид:

$$\Delta^d X_t = c + \sum_{i=1}^{p} a_i \Delta^d X_{t-i} + \sum_{j=1}^{q} b_j \epsilon_{t-j} + \epsilon_t, \tag{1}$$

где ϵ_t — стационарный временной ряд; c, a_i, b_j — параметры модели; Δ^d — оператор разности временного ряда порядка d (последовательное взятие d раз разностей первого порядка — сначала от временного ряда, затем от полученных разностей первого порядка, затем от второго порядка и т. д.)

Также данная модель интерпретируется как **ARMA(p+d,q)**- модель с d единичными корнями. При $d = 0$ имеем обычные **ARMA**-модели.

Для того чтобы выбрать параметры p, q, можно воспользоваться автокорреляционной функцией и частной автокорреляционной функцией. **Автокорреляционная функция (АКФ)** - это зависимость коэффициентов автокорреляции от лага. **Частная автокорреляционная функция (ЧАКФ)** – это зависимость частных коэффициентов автокорреляции от лага. ЧАКФ отличается от АКФ тем, что не учитывает влияние промежуточных лагов при расчете частных коэффициентов корреляций. Поэтому ЧАКФ дает более «чистую картину» зависимости ряда от лага.

Информационный критерий Акаике (AIC) — критерий, применяющийся исключительно для выбора из нескольких статистических моделей. Разработан в 1971 как «an information criterion» («(некий) информационный критерий») Хироцугу Акаике и предложен им в статье 1974 года[1].

В общем случае **AIC**:

$$AIC = 2k - 2\ln(L), \tag{2}$$

где k — число параметров в статистической модели, L — максимизированное значение функции правдоподобия модели.

Далее будем полагать, что ошибки модели нормально и независимо распределены. Пусть n — число наблюдений, а остаточная сумма квадратов

$$RSS = \sum_{i=1}^{n} \hat{\epsilon}_i^2. \tag{3}$$

Далее предполагаем, что дисперсия ошибок модели неизвестна, но одинакова для всех них. Следовательно:

$$AIC = 2k + n[\ln(2\pi RSS/n) + 1]. \tag{4}$$

В случае сравнения моделей на выборках одинаковой длины, выражение можно упростить, выкидывая члены зависящие только от n:

$$AIC = 2k + n[\ln(RSS)]. \tag{5}$$

Таким образом, критерий не только вознаграждает за качество приближения, но и штрафует за использование излишнего количества параметров модели. Считается, что наилучшей будет модель с наименьшим значением критерия **AIC**. Стоит отметить, что абсолютное значение AIC не имеет смысла — он указывает только на относительный порядок сравниваемых моделей.

Тест Дики — Фуллера (DF-тест)) — это методика, которая используется в прикладной статистике и эконометрике для анализа временных рядов для проверки на стационарность. Является одним из тестов на единичные корни.

Временной ряд имеет **единичный корень, или порядок интеграции один**, если его первые разности образуют стационарный ряд. Это условие записывается как $y_t \sim I(1)$ если ряд первых разностей $\Delta y_t = y_t - y_{t-1}$ является стационарным $\Delta y_t \sim I(0)$.

$$\Delta y_k \tag{6}$$

При помощи этого теста проверяют значение коэффициента a в авторегрессионном уравнении первого порядка $AR(1)$: $y_t = a \cdot y_{t-1} + \varepsilon_t$, где y_t — временной ряд, а ε — ошибка.

Если $a = 1$, то процесс имеет единичный корень, в этом случае ряд y_t не стационарен, является интегрированным временным рядом первого порядка — $I(1)$. Если $|a| < 1$, то ряд стационарный — $I(0)$. Значение $|a| > 1$ не свойственно для временных рядов, которые встречаются в реальной жизни - требуется более сложный анализ.

Преобразуем уравнение:

$$y_t = a \cdot y_{t-1} + \varepsilon_t$$
$$y_t - y_{t-1} = a \cdot y_{t-1} - y_{t-1} + \varepsilon_t$$
$$\Delta y_t = (a-1) \cdot y_{t-1} + \varepsilon_t$$
$$\Delta y_t = b \cdot y_{t-1} + \varepsilon_t, b = a - 1.$$

Сформулируем основную и альтернативную гипотезы: •

- Основная гипотеза: H_0: $b = 0$ - существует единичный корень, ряд нестационарный;
- Альтернативаная гипотеза: H_1: $b < 0$ - единичного корня нет, ряд стационарный.

Для того чтобы проверить временной ряд y на порядок интегрируемости, следует рассчитать значение t-статистики Стьюдента для параметра b и сравнить его с верхним и нижним критическими значениями DF-статистики из таблицы теста Дики-Фуллера. Если для n наблюдений значение расчетной t-статистики меньше, чем нижнее критическое значение, нулевую гипотезу $b = 0$ отклоняют и принимают альтернативную о стационарности процесса y_t. В противном случае, нулевая гипотеза не может быть отклонена. Если нулевая гипотеза не отклоняется, можно лишь утверждать, что процесс y_t нестационарен, т.е. либо он интегрируем более высокого порядка, либо неинтегрируем вообще.

R2-score — коэффициент детерминации (R^2) - доля дисперсии зависимой переменной, объясняемая рассматриваемой моделью зависимости. Лучший результат, который может дать $R2 - score$ - это 1.0. Если $R2$ принимает отрицательные значения или значения, превышающие 1.0, то это говорит об очень большой неточности предсказания модели.

3 Этапы решения

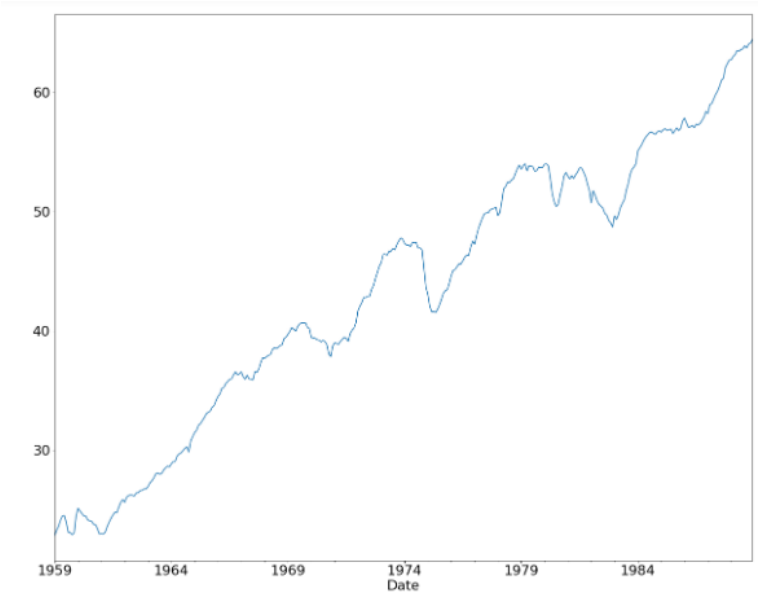
Используемые библиотеки и модули:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels as sm
import seaborn as sns
from statsmodels.iolib.table import SimpleTable
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.seasonal import seasonal_decompose
from sklearn.metrics import r2_score
from pylab import rcParams
```

Этапы

1. Считывание данных из файла *training.xlsx*, их сортировка и преобразовывание к более удобному виду, где в качестве индексов выступают даты. А также визуальное представление данных.

   ```
   training_data = pd.read_excel('training.xlsx')
   #print(training_data)
   training_data.index  = training_data.Date
   training_data = training_data.drop('Date', axis=1)
   training_data.sort_values('Date', ascending = True)
   training_data.Value.plot()
   ```



2. Проверка ряда на стационарность.

   Для этого был проведен тест Дики-Фулера с помощью фукнции *adfuller* из библиотеки *statsmodels*. Также, выводится на каком процентном уровне значимости (1%, 5%, 10%) отвергается нулевая гипотеза.

```
def ADF(data):
dftest = adfuller(series.dropna())
    print(" > Is the data stationary ?")
    dftest = adfuller(data)
    print("Test statistic = {:.3f}".format(dftest[0]))
    print("P-value = {:.3f}".format(dftest[1]))
    print("Critical values :")
    for k, v in dftest[4].items():
        print("\t{}: {} - The data is {} stationary with {}\% confidence".format(
```

   Результат вызова функции **ADF(training_data)**:

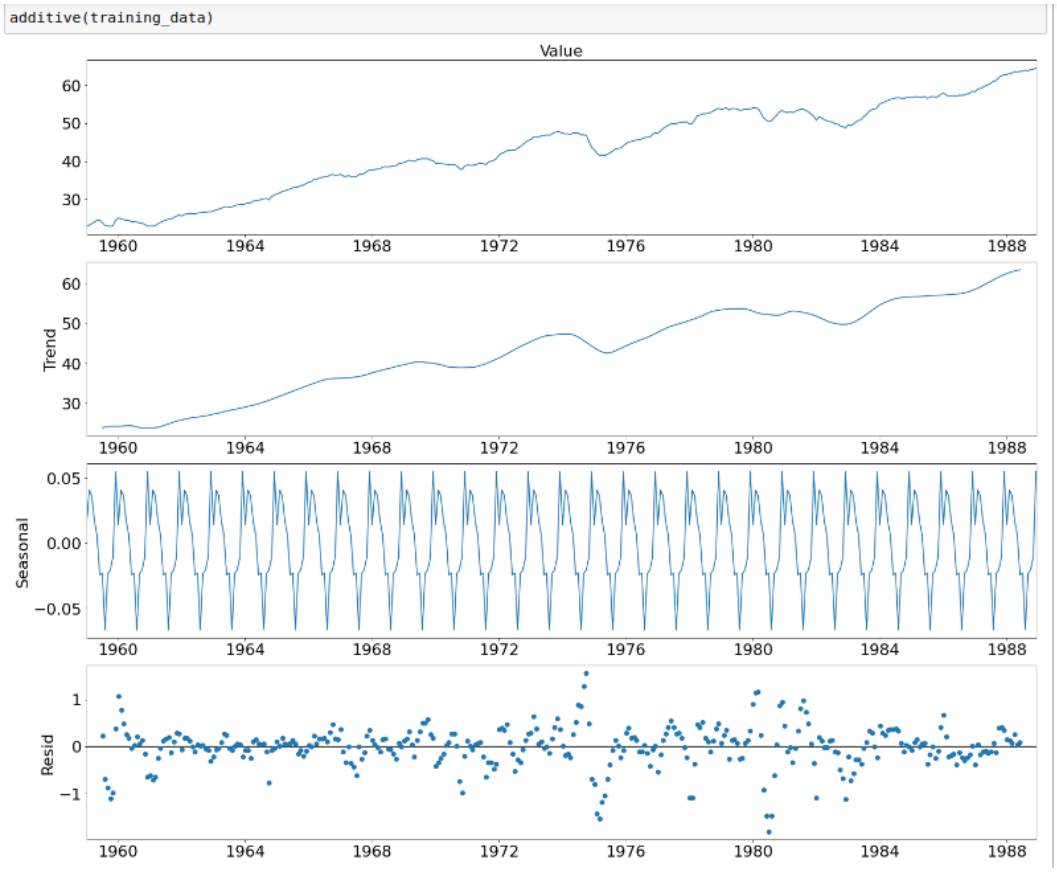
```
adfuller2(training_data)
```

```
 > Is the data stationary ?
Test statistic = -0.253
P-value = 0.932
Critical values :
        1%: -3.4489583388155194 - The data is not stationary with 99% confidence
        5%: -2.869739378430086 - The data is not stationary with 95% confidence
        10%: -2.5711381780459 - The data is not stationary with 90% confidence
```

3. Разложение временного ряда на тренд, сезональность, остаток в соответствии с аддитивной моделью с помощью функции **seasonal_decompose** из библиотеки **statsmodels** с указанием параметра **model='additive'**. Визуализирование вычисленных компонентов ряда, оценка стационарности получившихся рядов.

```
def additive(data):
    result = seasonal_decompose(data.Value, model='additive')
    result.plot()
    plt.show()

    from random import randrange
    print("\nCheck the components:")
    print('Trend:')
    ADF(result.trend)

    print('')
    print('Seasonal:')
    ADF(result.seasonal)

    print('')
    print('Resid:')
    ADF(result.resid)
```

Результат вызова функции **additive(training_data)**:




```
Check the components:
Trend:
 > Is the data stationary ?
Test statistic = -0.862
P-value = 0.800
Critical values :
        1%: -3.4503224123605194 - The data is not stationary with 99% confidence
        5%: -2.870338478726661 - The data is not stationary with 95% confidence
        10%: -2.571457612488522 - The data is not stationary with 90% confidence

Seasonal:
 > Is the data stationary ?
Test statistic = -495653880224065.938
P-value = 0.000
Critical values :
        1%: -3.4492815848836296 - The data is  stationary with 99% confidence
        5%: -2.8698813715275406 - The data is  stationary with 95% confidence
        10%: -2.5712138845950587 - The data is  stationary with 90% confidence

Resid:
 > Is the data stationary ?
Test statistic = -7.486
P-value = 0.000
Critical values :
        1%: -3.4496162602188187 - The data is  stationary with 99% confidence
        5%: -2.870028369720798 - The data is  stationary with 95% confidence
        10%: -2.5712922615505627 - The data is  stationary with 90% confidence
```

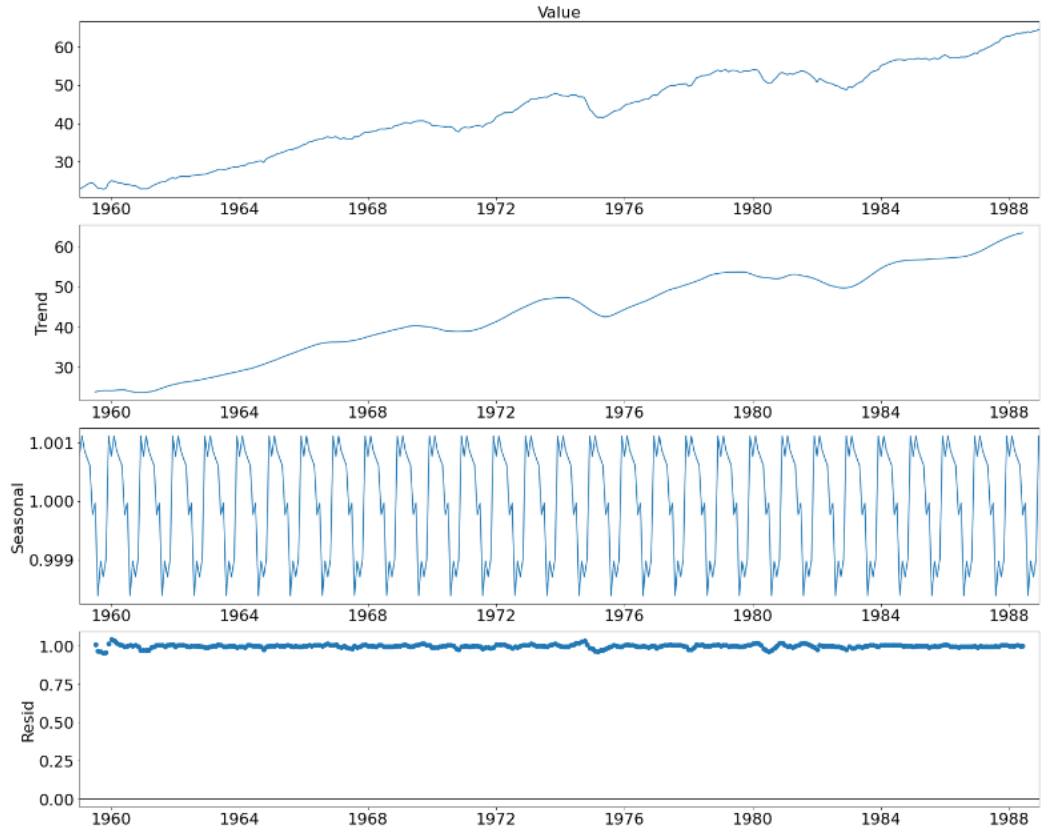
4. Разложение временного ряда на тренд, сезональность, остаток в соответствии

с мультипликативной моделью с помощью функции **seasonal_decompose** из библиотеки **statsmodels** с указанием параметра **model='multiplicative'**. Визуализирование вычисленных компонентов ряда, оценка стационарности получившихся рядов.

```
def multiplicative(data):
    result = seasonal_decompose(data.Value, model='multiplicative')
    result.plot()
    plt.show()

    from random import randrange
    print("\nCheck the components:")
    print('Trend:')
    ADF(result.trend)

    print('')
    print('Seasonal:')
    ADF(result.seasonal)

    print('')
    print('Resid:')
    ADF(result.resid)
```

Результат вызова функции **additive(training_data)**:

```
multiplicative(training_data)
```



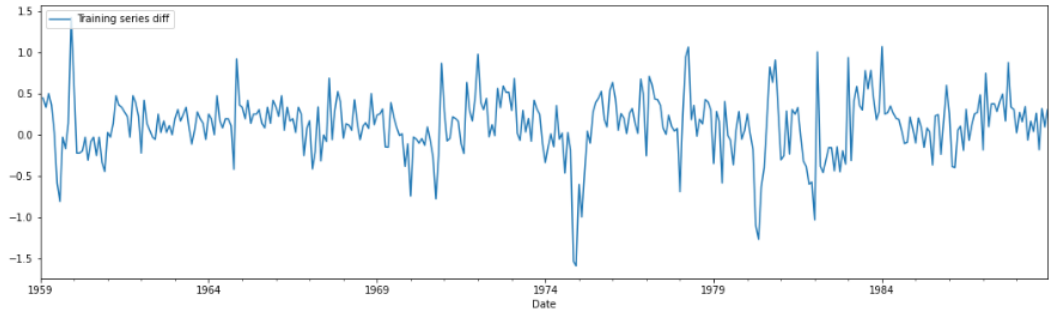

```
Check the components:
Trend:
 > Is the data stationary ?
Test statistic = -0.862
P-value = 0.800
Critical values :
        1%: -3.4503224123605194 - The data is not stationary with 99% confidence
        5%: -2.870338478726661 - The data is not stationary with 95% confidence
        10%: -2.571457612488522 - The data is not stationary with 90% confidence

Seasonal:
 > Is the data stationary ?
Test statistic = -39820684499748.508
P-value = 0.000
Critical values :
        1%: -3.4496162602188187 - The data is  stationary with 99% confidence
        5%: -2.870028369720798 - The data is  stationary with 95% confidence
        10%: -2.5712922615505627 - The data is  stationary with 90% confidence

Resid:
 > Is the data stationary ?
Test statistic = -7.466
P-value = 0.000
Critical values :
        1%: -3.4496162602188187 - The data is  stationary with 99% confidence
        5%: -2.870028369720798 - The data is  stationary with 95% confidence
        10%: -2.5712922615505627 - The data is  stationary with 90% confidence
```

5. Нахождение порядка интегрируемости ряда и проверка полученного ряда на

стационарность.

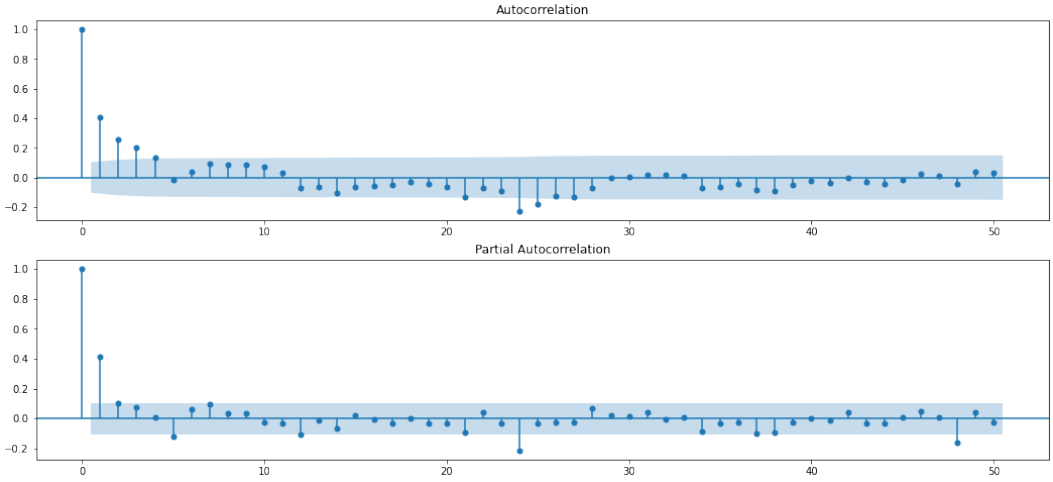
```
data_training_diff=data_training.Value.diff(periods=1).dropna()
ADF(data_training_diff)
data_training_diff.plot(label='Training series diff', figsize=(18,5))
plt.legend(loc='upper left')
plt.show()
```

```
adf: -7.367
p-value: 0.000
critical values:  {'1%': -3.4489583388155194, '5%': -2.869739378430086, '10%': -2.5711381780459}
единичных корней нет, ряд стационарен
```



6. Построение автокорреляции и частичной автокорреляции ряда. После изучения коррелограммы PACF можно сделать вывод, что p = 1, т.к. на ней только 1 лаг сильно отличнен от нуля. По коррелограмме ACF можно увидеть, что q = 1, т.к. после лага 1 значении функций резко падают.

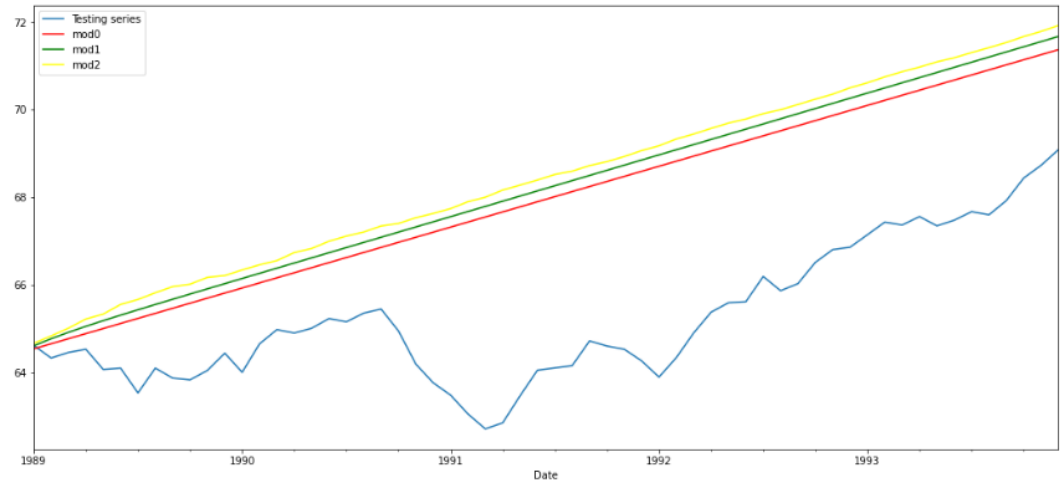
```
fig = plt.figure()
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(data_training_diff.squeeze(), lags=50, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(data_training_diff.squeeze(), lags=50, ax=ax2)
```



7. Итак, построим несколько моделей ARIMA. Для построения возмем данные из файла *training.xlsx*. Данные из *testing.xlsx* будем использовать для проверки точности прогноза нашей модели.

```
%%time

model_0 = sm.tsa.ARIMA(data_training, order=(0,1,0), freq='MS').fit()
model_1 = sm.tsa.ARIMA(data_training, order=(1,1,1), freq='MS').fit()
model_2 = sm.tsa.ARIMA(data_training, order=(12,1,4), freq='MS').fit()

pred_0 = model_0.predict('1989-01-01','1993-12-01', typ='levels')
pred_1 = model_1.predict('1989-01-01','1993-12-01', typ='levels')
pred_2 = model_2.predict('1989-01-01','1993-12-01', typ='levels')

data_testing.Value.plot(label='Testing series')
pred_0.plot(color='red', label='mod0')
pred_1.plot(color='green', label='mod1')
pred_2.plot(color='yellow', label='mod2')
plt.legend(loc='upper left')
plt.show()
```

```
print('AIC model0: %1.3f' % model_0.aic)
print('AIC model1: %1.3f' % model_1.aic)
print('AIC model2: %1.3f' % model_2.aic)

r2_0 = r2_score(data_testing, pred_0)
r2_1 = r2_score(data_testing, pred_1)
r2_2 = r2_score(data_testing, pred_2)
print('r2_0 R^2: %1.3f' % r2_0)
print('r2_1 R^2: %1.3f' % r2_1)
print('r2_2 R^2: %1.3f' % r2_2)
```




```
AIC model0: 315.986
AIC model1: 248.734
AIC model2: 250.599
r2_0 R^2: -2.683
r2_1 R^2: -3.281
r2_2 R^2: -3.851
```

8. Произведем отбор наилучшей модели с помощью информационного критерия Акаике.

```
%%time
ps = range(0, 2)
d=1
qs = range(0, 4)
```

```
parameters = product(ps, qs)
parameters_list = list(parameters)

results = []
best_aic = float("inf")

for param in parameters_list:
    #try except нужен, потому что на некоторых наборах параметров модель не обуча
    try:
        model=sm.tsa.ARIMA(data_training.Value, order=(param[0], d, param[1])).fi
    #выводим параметры, на которых модель не обучается и переходим к следующему н
    except ValueError:
        print('wrong parameters:', param)
        continue
    aic = model.aic
    #сохраняем лучшую модель, aic, параметры
    if aic < best_aic:
        best_model = model
        best_aic = aic
        best_param = param
    results.append([param, model.aic])
```

9. Результат вычислений из предыдущего пункта - таблица значений AIC для разных наборов параметров.

```
result_table = pd.DataFrame(results)
result_table.columns = ['parameters', 'aic']
print(result_table.sort_values(by = 'aic', ascending=True))
```

```
  parameters         aic
5     (1, 1)  248.733657
6     (1, 2)  250.623522
7     (1, 3)  251.083211
4     (1, 0)  251.982662
3     (0, 3)  254.652487
2     (0, 2)  259.218223
1     (0, 1)  267.450283
0     (0, 0)  315.985631
```

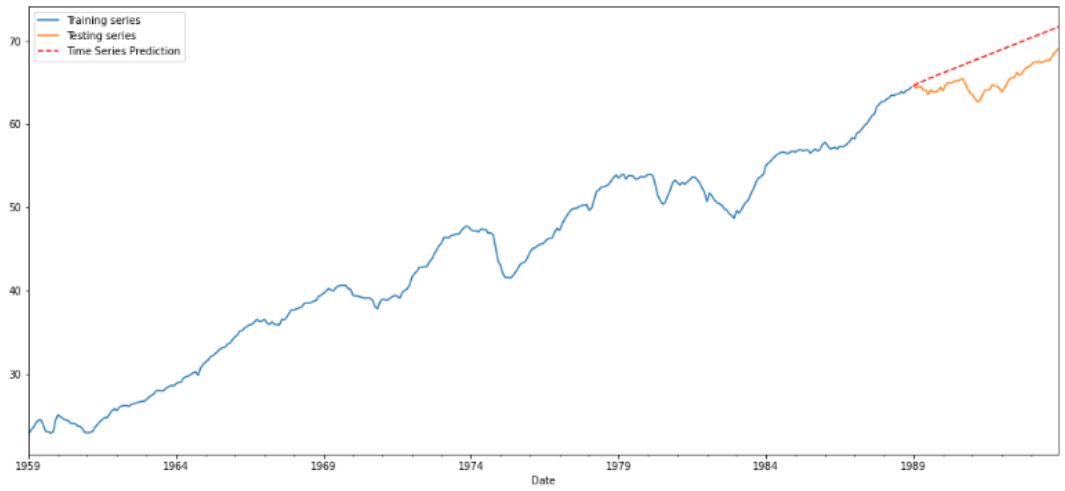
10. Информация о наилучшей модели ARIMA.

```
print(best_model.summary())
```

```
                             ARIMA Model Results
==============================================================================
Dep. Variable:                 D.Value   No. Observations:                  359
Model:                 ARIMA(1, 1, 1)   Log Likelihood                -120.367
Method:                       css-mle   S.D. of innovations              0.338
Date:                Sat, 19 Dec 2020   AIC                            248.734
Time:                        05:26:14   BIC                            264.267
Sample:                    02-01-1959   HQIC                           254.911
                         - 12-01-1988
==============================================================================
                    coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const             0.1176      0.036      3.271      0.001       0.047       0.188
ar.L1.D.Value     0.6531      0.091      7.185      0.000       0.475       0.831
ma.L1.D.Value    -0.2984      0.115     -2.601      0.009      -0.523      -0.074
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1            1.5313           +0.0000j            1.5313            0.0000
MA.1            3.3516           +0.0000j            3.3516            0.0000
-----------------------------------------------------------------------------
```

11. Прогнозирование и визуализация полученного прогноза.

```
data_training.Value.plot(label='Training series')
data_testing.Value.plot(label='Testing series')

pred = best_model.predict('1989-01-01','1993-12-01', typ='levels')
pred.plot(label='Time Series Prediction', style='r--')

plt.legend(loc='upper left')
plt.show()
```


Training series
Testing series
Time Series Prediction
70
60
50
40
30
1959
1964
1969
1974
1979
1984
1989
Date

4 Инструкция по запуску

В командной строке ввести:

1. Jupyter notebook Task2.ipynb;

2. Cells -> Run all.

5 Необходимое ПО

- Диструбутив **Anaconda**;

- Графическая веб-оболочка для IPython **Jupyter notebook**.

6 Вклад участников

Студент	**Вклад**
Альбатыров Алмаз	Пункты 5-11. Написание readme.pdf
Иванцов Глеб	Пункты 5-11. Написание readme.pdf
Сатвальдина Дана	Пункты 1-4. Написание readme.pdf

Список литературы

[1] Кантарович Г.Г. Анализ временных рядов: Лекции / Экономический журнал ВШЭ; — Москва, 2003. — 129 с.

[2] Магнус Я.Р., Катышев П.К., Пересецкий А.А. Эконометрика. Начальный курс / Издательство "Дело"; — Москва, 2004. — 576 с.

[3] Смирнова И.В., Клянина Л.Н. Статья: Определение порядка интегрируемости экономических временных рядов / Электронный научный журнал «Инженерный вестник Дона»; — Ростов-на-Дону, 2016. — 10 с.