**Временные ряды.**

Временной ряд представляет собой последовательность данных, описывающих объект в последовательные моменты времени. В отличие от анализа случайных выборок, анализ временных рядов основывается на предположении, что последовательные данные наблюдаются **через равные промежутки времени.**

Ряды состоят из систематической составляющей и случайного шума. Регулярные составляющие принадлежат к двум классам: трендам или циклическим составляющим.

Временной ряд можно разложить на 4 составляющие: , где -тренд, -цикл, -сезонная компонента, ξ-шум (Аддитивная модель). В мультипликативной модели предполагается, что вместо суммы ряд состоит из произведения тренда, цикла и сезонной компоненты.

Тренд – систематическая компонента, закономерно изменяющаяся во времени, обычно показывает плавное долгосрочное изменение уровня ряда.

Сезонность – циклические изменения уровня ряда с постоянным периодом.

Цикл – изменения уровня ряда с переменным периодом.

Ошибка – непрогнозируемая случайная компонента ряда.

Ряд является стационарным, если с течением времени не меняет некоторые свои характеристики, а именно: матожидание, дисперсия, ковариация.

Наличие тренда у ряда сразу говорит о его нестационарности, поскольку меняются средние значения.

Ряды с сезонностью так же не являются стационарными, поскольку, если мы возьмем окно меньшего размера, чем период сезонности, то мы получим разные распределения в таких окнах.

**Тест Дики-Фуллера.**

Если параметр , то ряд является стационарным. Если же , то процесс имеет единичный корень, тогда ряд является интегрированным рядом первого порядка.

**Интегрированные ряды.**

Интегрированный временной ряд-нестационарный временной ряд, разности некоторого порядка которого являются стационарным временным рядом.

Для начала необходимо определить понятие TS-ряда (ряда, называемого стационарным относительно тренда. Ряд называется TS рядом, если существует некоторая детерминированная функция такая, что разность является стационарной. К TS-рядам относятся все стационарные ряды.

Временной ряд называется интегрированным рядом порядка , если разности -го порядка являются стационарными, а разности меньшего порядка стационарными не являются.

Количество единичных корней временного ряда совпадает с порядком его интегрированности.

**Модель *ARIMA*.**

*ARIMA*-модель, в которой соединены нахождение разностей временного ряда порядка и *ARMA*.

*ARMA*-комбинация авторегрессионного процесса и процесса скользящей средней. Она основана на предположении о том, что текущее значение временного ряда зависит только от линейной комбинации предыдущих значений временного ряда и белого шума. Такая модель выглядит следующим образом:

Где – константа, – белый шум, то есть последовательность независимых и одинаково распределенных случайных величин, с нулевым средним, а и – действительные числа, авторегрессионные коэффициенты и коэффициенты скользящего среднего соответственно.

При помощи модели ARMA мы можем описать любой стационарный ряд с наперед заданной точностью.

*ARIMA*

Где – стационарный ряд, - оператор разности временного ряда порядка (последовательное взятие раз разностей первого порядка – сначала от временного ряда, затем от полученных разностей первого порядка, затем второго и т.д.)

Данная модель интерпретируется как *ARMA*-модель с единичными корнями. При получаем обычную *ARMA* модель.

**Процесс построения *ARIMA* модели.**

1. Необходимо определить, является ли ряд стационарным. Если ряд таковым не является, или не очевидно, что он стационарен, то переходим ко второму шагу.
2. Нужно построить диаграмму выборочных автокорреляционной и частной автокорреляционной функций.

Автокорреляция показывает меру силы линейной связи значений ряда в настоящем и в прошлом.

Функция автокорреляции задается таким образом:

где , а – лаг автокорреляции

Проверка значимости отличия автокорреляции от нуля:

Временной ряд:

Нулевая гипотеза:

Альтернатива:

Статистика:

Для выборочной автокорреляции формула выглядит несколько иначе:

В этой формуле все математические ожидания заменены на выборочные средние, а все дисперсии на выборочные дисперсии.

Частичная автокорреляция – автокорреляция после снятия авторегрессии предыдущего порядка.

Авторегрессия - линейное регрессивное уравнение, в котором в качестве результативного признака выступает значение ряда в момент времени , а в качестве факторных признаков значение ряда в предыдущие моменты времени.

Формула авторегрессии -го порядка:

где – константа (постоянный член авторегрессии), – случайная компонента, – коэффициенты авторегрессии.

1. Анализируем диаграммы на признаки стационарности ряда. Быстрое затухание АКФ (автокорреляционной функции) скажет нам о стационарности ряда. Если требуется, то проводим тест Дики-Фуллера. Если в соответствии с тестами и оценками ряд не является стационарным, то для перехода к стационарному ряду применяем оператор взятия последовательных разностей, тем самым определяем значение параметра .
2. После получения стационарного ряда исследуется характер поведения выборочных АКФ и ЧАКФ, выдвигаются гипотезы о значениях параметров . Формируется базовый набор гипотез.
3. На этом шаге идет оценка параметров методом максимального правдоподобия.

Для этого построим диаграммы АКФ и КФ. На оси Ox будет отмечен номер лага, а на оси Oy значение на этом номере, именно количество лагов определяет количество значимых коэффициентов в функции. Посмотрим на диаграмме АКФ на то, после какого последнего по счету элемента начинается незначительное отклонение значений от нуля, и сколько элементов в начале значительно отклоняются от нуля, это и буду коэффициенты Q и q соответственно. На диаграмме ЧАКФ аналогичным образом определим значения P и p.

1. Для проверки каждой модели анализируем ее ряд остатков. Если модель адекватна, то ряд остатков должен быть похож на белый шум, то есть их выборочные АКФ не должны отличаться от нуля. Для проверки гипотезы о том, что остатки являются белым шумом, используется Q-статистика Льюинга-Бокса. Она определяется, как

Где – объем выборки, – максимальное количество лагов, – коэффициенты автокорреляционной функции. Если после оценки остается более одной модели, то стоит учесть такие параметры, как повышение точности и уменьшение числа параметров модели. Это можно проверить при помощи критерия Акаике, который применяется для выбора одной из нескольких моделей.

В общем случае он задается формулой , где – число параметров в статистической модели (, а – максимизированное значение функции правдоподобности модели.

Так как ошибки у нас являются белым шумом, то можем ввести функцию

Так как мы сравниваем модели на выборке одинаковой длины, то функция Акаике принимает окончательный вид

Таким образом для каждой модели мы сможем посчитать значение критерия Акаике. Наилучшей окажется там модель, у которой значение этого критерия будет наименьшим.

1. С помощью выбранной в конечном итоге модели можно строить точный и интервальный прогнозы на L шагов вперед. Для этого необходимо
2. Получить уравнение нашего ряда вида

где надо заменить , ошибку из будущего мы заменяем на 0, так как считаем ее случайной величиной со средним значением 0, а ошибки из прошлого заменяем на наши остатки в этих же точках.

1. При прогнозе времени больше, чем на 1 в нашем ряду появятся значения рядов из будущего, их необходимо будет заменить на наши прогнозы

**Алгоритм работы программы.**

1. Для написания кода нам нужно подгрузить список библиотек
2. Библиотека pandas – предоставляет нам различные функции для анализа данных
3. Библиотека numpy – предоставляет нам поддержку многомерных массивов и матриц
4. Библиотека statsmodels.api – предоставляет нам функции для работы со статистическими данными и временными рядами.
5. Библиотек matplotlib – предоставляет нам функции для графической работы с данными
6. Первым делом нам требуется считать данные из файла. Это реализуется при помощи функции read\_csv из библиотеки pandas, которая читает файл формата csv. Для этого мы в параметрах указываем имя файла, sep – формат разделителя “;”, parse\_dates – обрабатывает даты в колонке ‘Date’, index\_col – изменяет индексы на значения в колонке ‘Date’.

После чтения мы получаем данные вот в таком формате:

Date

1959-01-01 22.8331

1959-01-02 23.2802

1959-01-03 23.6156

1959-01-04 24.1186

1959-01-05 24.4820

1959-01-06 24.5099

1959-01-07 23.9230

1959-01-08 23.1125

1959-01-09 23.0846

1959-01-10 22.9169

1959-01-11 23.0567

1959-01-12 24.4820

1960-01-01 25.1248

1960-01-02 24.9012

1960-01-03 24.6776

1960-01-04 24.4820

1960-01-05 24.4540

1960-01-06 24.1466

1960-01-07 24.0628

1960-01-08 24.0348

1960-01-09 23.7833

1960-01-10 23.7554

1960-01-11 23.4200

1960-01-12 22.9728

1961-01-01 23.0008

1961-01-02 22.9728

1961-01-03 23.1126

1961-01-04 23.5877

1961-01-05 23.9510

1961-01-06 24.2864

...

Это готовый временной ряд, в котором все замеры произведены 1 числа каждого месяца.

1. Функции rolling(window).mean и rolling(window).std высчитывают скользящее среднее и стандартное отклонение при заданных параметром window размерах окна(промежутка, в котором измеряется среднее), затем при помощи функции plot из библиотеки matplotlib, принимающей в нашем случае параметр цвета, параметр legend, который обозначает, что на графике будет размещаться так называемая легенда(область с местом для лэйблов), и параметр label, который создает лэйбл с описанием, мы строим графики.
2. На следующем этапе мы проверяем ряд на стационарность при помощи функции, реализующей тест Дики-Фуллера(которая не всегда выдает абсолютно верный результат :может быть такое, что ряд все же не стационарный, но значение теста Дики-Фуллера не сможет это распознать, потому что у него есть некоторая погрешность). Для этого мы используем функцию tsa.adfuller из библиотеки statsmodels, которая на вход принимает временной ряд, а на выходе выдает набор параметров, состоящий из:
3. Тестовая статистика
4. Приблизительная ценность
5. Количество использованных лаг
6. Число наблюдений
7. Критические значения для тестовой статистики на уровнях 1%,5%,10%

После обработки функцией нашего временного ряда мы сравниваем полученные значения таким образом: если критические значения для тестовой статистики на уровне 5% больше, чем значение тестовой статистики, то мы считаем, что есть единичные корни и ряд не стационарен, если же значение меньше, то считаем, что ряд стационарен.

1. На этом этапе мы проводим декомпозицию функции – разложение ее на тренд, сезонность и остаток. Мы будем использовать и сравнивать две модели – аддитивную и мультипликативную. Для этого будем использовать функцию seasonal\_decompose из библиотеки statsmodels, которая в нашем случае принимает на вход два параметра – ряд и модель (аддитивная или мультипликативная). Эта функция возвращает объект с атрибутами тренда, сезонности и остаточности. После этого при помощи функции plot, описанной выше, мы строим все три графика отдельно, разделяя их командами plot.show(), которые показывают график, который построен на данном этапе.
2. На этом этапе для каждого из предложенных параметров, которые вернула нам функция seasonal\_decompose, а именно ряда тренда, ряда сезонности и ряда остатков мы проводим тест Дики-Фуллера, описанный выше, и выясняем, являются ли эти ряды стационарными, предварительно заполнив возможные отсутствующие значения функцией dropna, принимающей параметр inplace=True, обозначающий, что отсутствующие значения надо дополнить.
3. После проверки всех составляющих ряда мы переходим к вычислению порядка интегрированности ряда и приведения его к стационарному виду. Это осуществляется при помощи теста Дики-Фуллера и дифференцирования ряда, которое осуществляется функцией diff, принимающей на вход параметр, задающий период деления ряда.
4. После вычисления порядка интегрированности ряда ряда мы переходим к построениям диаграмм автокорреляционной и частичной автокорреляционной функции. Для этого мы используем функции plot\_acf и plot\_pacf из библиотеки statsmodels, которые принимают на вход такой набор параметров: временной ряд, количество лагов, ax-который указывает, что диаграмму надо строить на уже заданной фигуре (указывает на какой).
5. На основе диаграмм мы выбираем приблизительные промежутки параметров, для которых мы будем методом перебора выбирать оптимальную или оптимальные модели ARIMA. После выбора промежутков мы создаем набор всевозможных наборов параметров с помощью функции product из библиотеки itertools, которая реализует вложенный цикл, создающий всевозможные, упорядоченные в порядке вхождения параметров, комбинации (на примере параметров ‘ABCD’, ‘xy’ на выходе мы получим Ax, Ay, Bx, By, Cx, Cy, Dx, Dy). Затем мы преобразуем этот набор в список при помощи функции list, принимающей массив значений, и преобразующей его в список.
6. После таких преобразований мы можем приступить к последовательному оцениванию качества наших моделей. Для начала мы в цикле проверяем все наши модели на возможность их сходимости, если такой критерий не выполняется, то мы отбрасываем их.

Проверку и оценку мы выполняем при помощи функции SARIMAX, которая получает на вход параметры:

1. Временной ряд
2. Набор коэффициентов p,d,q

На выходе мы получаем полноценную ARIMA модель и сравниваем ее значение aic для теста Акаике с лучшим показателем, который у нас уже имеется. Если текущая модель лучше, то мы обновляем все значения лучшей модели и добавляем результаты тестов в массив результатов при помощи команды append, которая принимает на вход массив элементов и добавляет его к концу текущего массива, и переходим к проверке следующей, если она не лучше, то просто переходим к проверке следующей. После окончания всех проверок мы выводим верхушку таблицы результатов (используя метод head, выводящий только голову списка, который может принимать параметром количество элементов, которые требуется вывести), предварительно сортируя ее функцией sort\_values из библиотеки pandas, принимающей первым параметром значение, по которому надо сортировать, вторым параметром ascending, указывающий на то, что сортировать надо по возрастанию.

1. Затем мы выводим сведения о лучшей модели при помощи метода summary модели ARIMA.
2. И теперь, после получения оптимальной модели мы можем перейти к процессу предсказания значений временного ряда. Для этого мы будем использовать метод predict модели ARIMA, который принимает на вход значения даты начала предсказания и даты окончания предсказания. После чего мы будем сравнивать наши предсказания с полученными реальными данными. Для этого мы строим графики описанными выше функциями. После чего мы вычисляем коэффициент r2 при помощи функции r2\_score из библиотеки sklearn, который принимает на вход два временных ряда, первый это истинные значения, а второй это предсказываемые значения, а на выходе выдает значение коэффициента.