Forecasting number of dwelling units approved.

Problem specification

Obtain publicly available data from Australia Bureau of Statistics Beta API and predict number of dwelling units approved in the New South Wales region.

Understanding JSON structure

The URL to the data source contains the following information.

1. Dataset Identifier: ABS_BA_SA2_ASGS2016
2. Measure: Total number of dwelling units(ID:1)
3. Sector of work: Total Sectors(id:9)
4. Type of work: New(id:1)
5. Type of building: all
6. Region: [Gosford, Central Coast, New South Wales, Australia]
7. Frequency: Monthly
8. Start time: 2011-07
9. End time: 2017-07

Environment setup

Docker container and Pycharm are used to achieve the given task.

1. Install Docker

    - Docker installation guide for Windows
    - Docker installtion guide for Mac
    - Docker installation guide for ubuntu

2. Install Pycharm

3. After Docker is successfully installed create a docker image using the Dockerfile provided in this project.

4. Please follow the steps to create the Docker image and run the Docker container.

```
$cd heidelbergcement
$docker build -t python/statsmodels:1.0 .
```

Use the resulting image as an interpreter. Which contains Python 3.7 and all the required packages.

Note: This project can also be run without Docker by installing the necessary packages.

Following software and libraries are required:

1. Python 3.6 or later
2. numpy
3. scipy
4. scikit-learn
5. pandas
6. statsmodels
7. requests
8. matplotlib

Data transformation and time series extraction

Data downloaded from the given URL contains headers, data and meta data or structure of the data. We have to extract only the observations and transform the observations into the data-frame. Later, required time series needs to be extracted.

Below are the steps followed to extract the time series for Total number of new houses in New South Wales.

1. Extract only the observations from downloaded json.

2. Extract time series from the observations using regular expression `'0:0:0:0:2:2:0:*`.

   In the regular expression, sixth field is the region and last field is the duration. Region value `2` is for New south wale region based on the structure. Particular time series is extracted using above pattern.

3. Pandas data-frame is created using the above resulting time series.

4. Data-frame is saved to a file.

Time series summary

Final time series contains 73 instances starting from 2011-07 to 2017-07.

1. Summary

	Dwelling units
count	73.000000
mean	2009.178082
std	488.921783
min	982.000000
25%	1594.000000
50%	2007.000000
75%	2461.000000
max	3006.000000

2. Original time series

Plot of the original time series

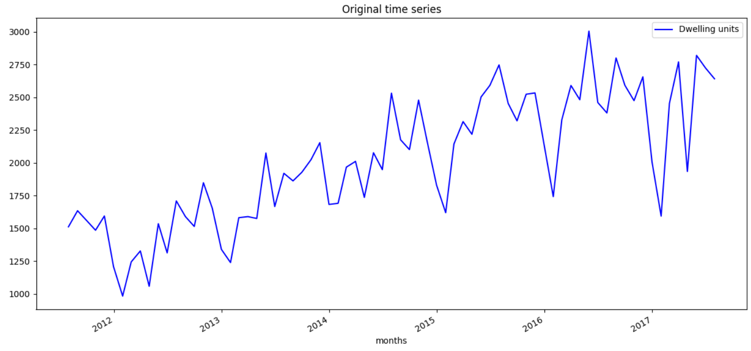


Figure 1: image

3. Components of time series

Important interferences 1. Year on year number of dwelling units approved is increasing. It clearly shws that there is a trend in the time series. 2. We can also infer from the plot that there is a seasonality in the time series.

Stationarity test

Once we know the patterns, trends, seasonality and cycles in the time series, we can check if the time series is stationary or not. In this task, I have used rolling

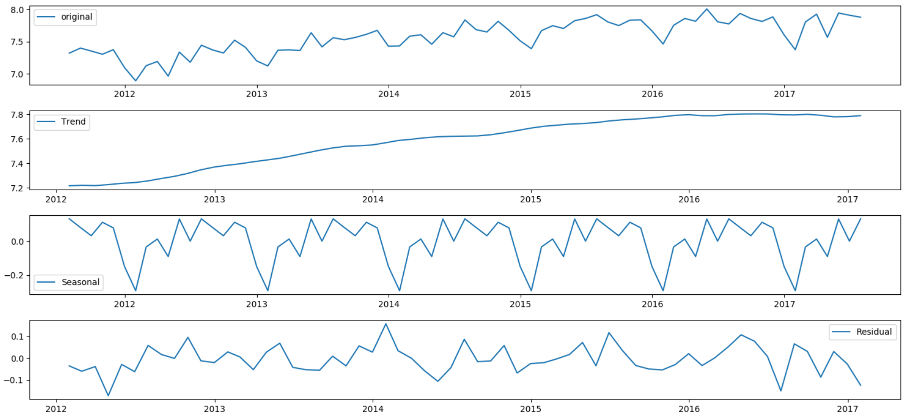


Figure 2: image

statistics and Dickey-Fuller tests.

Rolling stastics

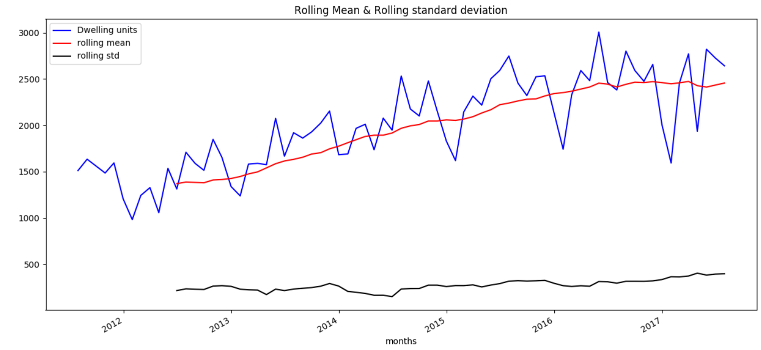


Figure 3: image

Moving average and moving standard deviation are not constants. Hence, the time series is not stationary.

Dickey-Fuller test

`Dickey-Fuller test results with original time series`

```
Test statistic              -1.639759
p-value                      0.462387
Lags used                   12.000000
Number of observations      60.000000
critical values (1%)        -3.544369
critical values (5%)        -2.911073
critical values (10%)       -2.593190
```

Dickey-Fuller test with autolag `ÀIC` is used to test the stationarity of the time series. In the test results, p-value > 0.05, it means data has unit roots. Also, test statistic value is greater than the critical values. Hence, the time series is non-stationary.

Making time series stationary

Applying log scale

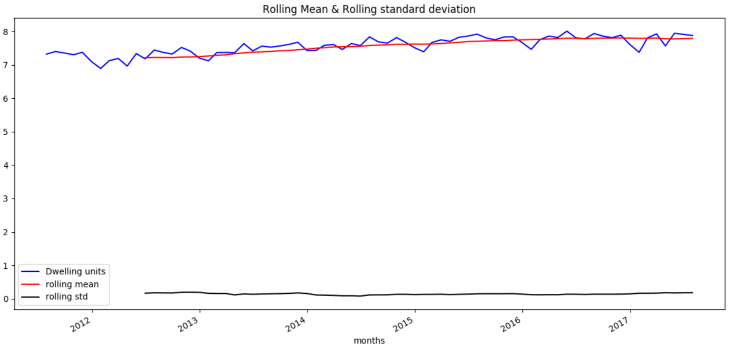


Figure 4: image

Log scale is applied to the original time series and stationary tests results show that the time series is non-stationary.

```
Dickey-Fuller test results with Log scale time series
Test statistic              -2.297622
p-value                      0.172736
Lags used                   12.000000
Number of observations      60.000000
critical values (1%)        -3.544369
critical values (5%)        -2.911073
critical values (10%)       -2.593190
```

Log scale - moving average

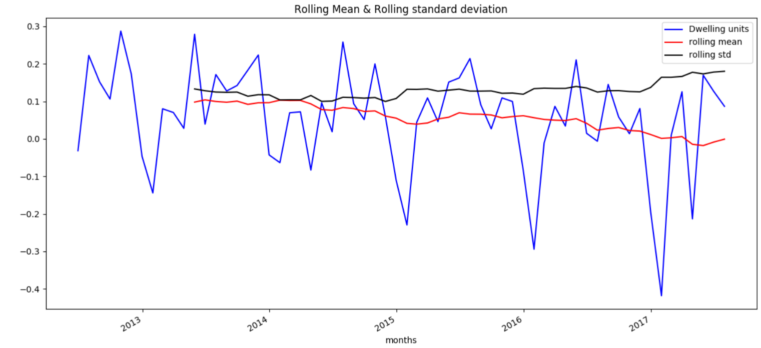


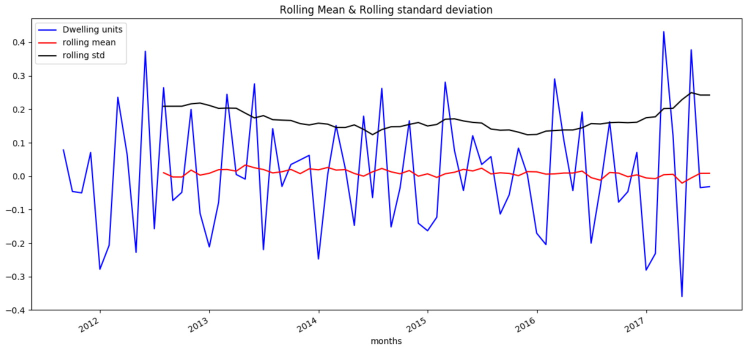
Figure 5: image

Time series obtained by differencing the log scale data with moving average is tested for stationarity and stationary tests. The results show that the time series is non-stationary.

```
Dickey-Fuller test results with Log scale - moving average time series
Test statistic             -0.328929
p-value                     0.921306
Lags used                  11.000000
Number of observations     50.000000
critical values (1%)       -3.568486
critical values (5%)       -2.921360
critical values (10%)      -2.598662
```

Differencing and Shifting

Time series obtained by differencing the log scale data and shifted log scale data is tested for stationarity and the results that time series is stationary.



Dickey fuller test results with shifting time series Test statistic -3.478557 p-value 0.008555 Lags used 11.000000 Number of observations 60.000000 critical values (1%) -3.544369 critical values (5%) -2.911073 critical values (10%) -2.593190

ACF and PACF to find P and Q values

Auto correlation and partial auto correlation are used to find the optimal parameters `p` and `q`.

We have used lag difference 1 while making the time series stationary. Hence, d=1.

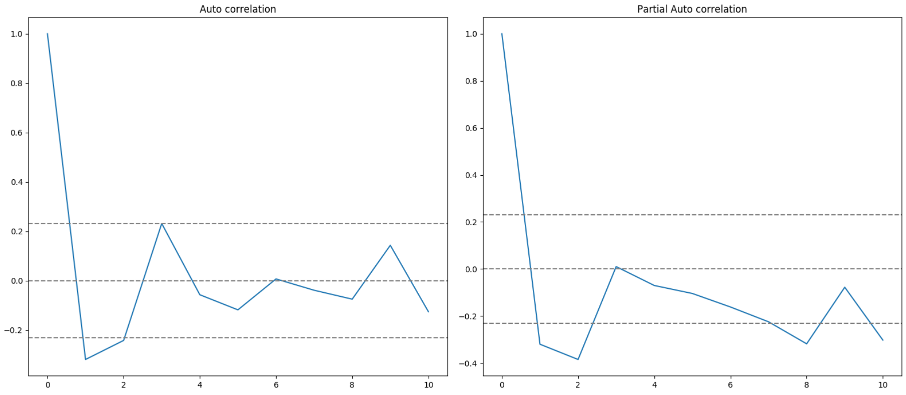


Figure 6: image

p and q values are inferred from the acf and pacf plots. They are as follows.

```
p=1
q=1
```

Build ARIMA model

```
ARIMA(p,d,q) - Auto-regressive integrated moving average.
ARIMA models are the most commonly used models for forecasting non stationary time series
which can be made stationary by means of differencing, deflatting, or logging.
```

With the parameters p,d and q in hand we are now building ARIMA model. The values found in the previous section are the approximate estimations. ARIMA model is trained using different values of p and q. Better model is obtained using p=1 and q=0.

Below are the different combinations of the parameters ans their corresponding AIC, BIC and RSS.

p	d	q	RSS	AIC	BIC
1	1	2	0.1592	-62.64	-51.26
1	1	0	0.0005	-50.29	-43.46
1	1	1	0.0012	-64.54	-55.43
2	1	1	0.0009	-57.50	-46.12
0	1	1	0.0205	-59.17	-52.34
2	1	0	0.0008	-59.49	-50.39

Predictions

Created three year forecast of the number of dwelling units approved. Below is the forecasting graph.

Below are the actual and predicted values.

```
[2765.5910849  2779.93436085 2764.81472373 2807.07099437 2834.06638185
 2846.1474206  2871.62033432 2896.91775641 2917.38889503 2940.55142131
 2964.66624025 2987.61680375 3011.08924997 3035.11616773 3059.02468854
 3083.12355632 3107.5327331  3132.07730253 3156.79597876 3181.74174841
 3206.87721586 3232.20223753 3257.73443242 3283.46844238 3309.40286297
 3335.54343632 3361.89099452 3388.44592132 3415.2107592  3442.18722763
 3469.376615   3496.78076067 3524.4014364  3552.24025513 3580.29895657
 3608.5793078 ]
```

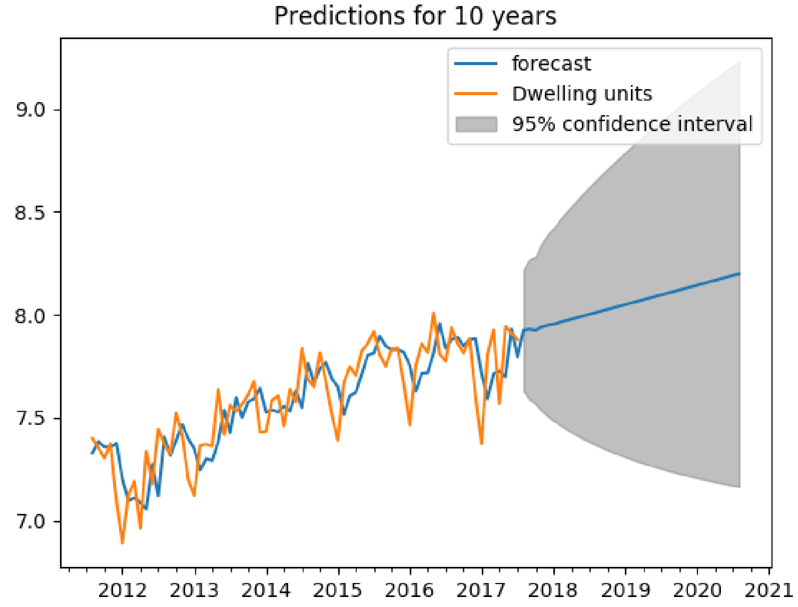


Figure 7: image

How to run the project

Using docker

1. Extract the tar file `heidelbergcement.tar`

2. After successfully installing Docker and building image using Environment setup step run the project using below command.

   ```
   $cd heidelbergcement
   $docker run -v ‘pwd‘:/heidelbergcement python/statsmodels:1.0
   ```

   After running the project, some of the results will be printed in the console and plots can be found in plots directory.

Manual method

Project can also be run manually using below steps.

1. Extract tar file `heidelbergcement.tar`.
2. Install Python 3.6 or later and other packages mentioned in Environment setup step.
3. `cd heidelbergcement`
4. set working directory to `heidelbergcement/lib`
5. execute `python3 run.py`