Serie 11

Aufgabe 11.1

Das Bundesamt für Statistik stellt eine sehr grosse Anzahl von Datensätzen auf ihrer Webseite zur Verfügung. Mit Hilfe des Tools `STAT-TAB` ist es relativ einfach, nach bestimmten Datensätze Ausschau zu halten and spezifische Tabellen herunterzuladen und individuell zusammenzustellen.

a) Besuchen Sie die Webseite

`https://www.bfs.admin.ch/bfs/en/home/services/recherche/stat-tab-online-data-search.html`

und versuchen Sie, mit dem Datenbrowser zurechtzukommen.

Hinweis: Konsultieren Sie folgendes Hilfsmenu: *Guideline for online data search*, welches Sie am Ende der Seite herunterladen können.

b) Die Mehrzahl der Datensätze auf dieser Webseite beinhalten tatsächlich Zeitreihen. Versuchen Sie die Zeitreihe zu finden, welche die Anzahl Elektro-Personenwagen (PW) von 1990 - 2017 enthält (separat für jeden Kanton). Laden Sie die entsprechende Tabelle als `.csv` Datei herunter.

c) Lesen Sie die Datei in **`Python`** ein, und definieren Sie eine Zeitreihe für die Anzahl Elektro-Autos in Luzern. Erstellen Sie eine Graphik der Zeitreihe.

Hinweis: Die Dateien, welche mit Hilfe von `STAT-TAB` generiert werden, enthalten manchmal zusätzliche Informationen neben den regulären Headern mit den Variablennamen. Schauen Sie sich die `.csv` Datei in einem Texteditor an. Finden Sie heraus, wieviele Zeilen dem tatsächlichen Datensatz vorangehen (ausgenommen vom Header). Dann können Sie mit der **`Python`**-Funktion **`pd.read_csv()`** die Daten einlesen, wobei Sie mit Hilfe des Parameters **`skiprows=k`** die ersten k Zeilen überspringen können:

```
import matplotlib.pyplot as plt
import pandas as pd
from pandas import DataFrame

pw_electric = pd.read_csv('.../PW_electric.csv', sep=',',
                          skiprows=2, header=0,
                          encoding = "utf-8",
                          index_col=0)
pw_electric.head()
pw_electric_luzern = DataFrame(pw_electric.ix["Luzern",1:])
pw_electric_luzern
pw_electric_luzern["Year"] = pd.DatetimeIndex(pw_electric_luzern.index)
```

```
pw_electric_luzern.set_index("Year", inplace=True)
pw_electric_luzern.plot()
plt.xlabel("Jahr")
plt.ylabel("Anzahl Elektro-Autos Luzern")
plt.show()
```

d) Wiederholen Sie die Vorgehensweise aus Teilaufgabe (c) für den Kanton Zürich.

e) Wie können Sie die Daten zwischen den Kantonen Luzern und Zürich korrekt miteinander vergleichen?

Aufgabe 11.2

In dieser Aufgabe behandeln wir die vierteljährliche Bierproduktion in Australien.

a) Lesen Sie die Datei `AustralianBeer.csv` in **Python** ein, und konvertieren Sie die Daten in das Zeitreihenformat. Zeichnen Sie die Daten auf.

Hinweis: Um die Daten in **Python** in eine Zeitreihe zu konvertieren, benützen Sie

```
AusBeer = pd.read_csv(".../AustralianBeer.csv",sep=";",header=0)
AusBeer.head()
AusBeer["Quarter"] = pd.DatetimeIndex(AusBeer["Quarter"])
AusBeer.set_index("Quarter", inplace=True)
AusBeer.columns=["Megalitres"]
AusBeer.head()
AusBeer.describe()
AusBeer.plot()
plt.ylabel("Megalitres Beer")
```

b) Zeichnen Sie die aggregierten jährlichen Reihen auf und die Boxplots, welche die beobachteten Daten im Vierteljahr-Zyklus zusammenfassen. Kommentieren Sie Ihre Beobachtungen der Graphiken.

Hint:

```
AusBeer.resample("A").mean().plot()
AusBeer['quarter'] = AusBeer.index.quarter
AusBeer.boxplot(by="quarter")
```

c) Zerlegen Sie die Zeitreihe in die Komponenten *trend*, *saisonalen Effekt* und *Residuen* mit Hilfe der Funktion **`seasonal_decompose()`**. Kommentieren Sie die Resultate. Denken Sie, eine Datentransformation vor der Zerlegung wäre angebracht?

Hinweis:

```
from statsmodels.tsa.seasonal import seasonal_decompose
seasonal_decompose(AusBeer1, model="additive", freq=4).plot()
```

d) Zerlegen Sie die Zeitreihe nun mit Hilfe des STL-Verfahrens. Wählen Sie eine passende Fensterbreite, indem Sie für den Parameter **`period=...`** in **`decompose()`** den Wert variieren. Vergleichen Sie das Resultat mit dem Resultat in Teilaufgabe c).

```
from stldecompose import decompose
AusBeer_stl = decompose(AusBeer["Megalitres"], period=...)
AusBeer_stl.plot();
```

Aufgabe 11.3

In dieser Aufgabe behandeln wir die vierteljährliche Produktion von Elektrizität.

a) Lesen Sie die Datei `AustralianElectricity.csv` in **`Python`** ein, und konvertieren Sie die Daten in das Zeitreihenformat. Zeichnen Sie die Daten auf.

Hinweis: Um die Daten in **`Python`** in eine Zeitreihe zu konvertieren, benützen Sie

```
Electricity = pd.read_csv(".../AustralianElectricity.csv", sep=";", header=
Electricity.head()
Electricity["Quarter"] = pd.DatetimeIndex(Electricity["Quarter"])
Electricity.set_index("Quarter", inplace=True)
Electricity.columns=["Electricity production Australia"]
Electricity.head()
Electricity.plot()
plt.ylabel("Million Kilowatthours")
```

b) Wenden Sie eine passende Daten-Transformation an, so dass sich die Varianz der Zeitreihe stabilisiert wird. **Hinweis:** Wenden Sie die Funktion **`box.cox`** an, und bestimmen Sie einen optimalen Wert für **`lambda`**, indem Sie die Graphik der transformierten Zeitreihe betrachten. Hier ist der Code mit **`lambda = 1.3`**:

```
def boxcox(x,lambd):
  return np.log(x) if (lambd==0) else (x**lambd-1)/lambd

# replace "yourSeries" by the name of your series
yourSeries_tr = boxcox(yourSeries, 1.3)
yourSeries_tr.plot()
plt.show()
```

c) Zerlegen Sie die Zeitreihe in die Komponenten *trend*, *saisonale Effekte* und *Residuen* mit Hilfe der Funktion **`seasonal_decompose()`**. Kommentieren Sie die Resultate. Denken Sie, eine Datentransformation vor der Zerlegung wäre angebracht?

Hinweis:

```
from statsmodels.tsa.seasonal import seasonal_decompose
seasonal_decompose(Electricity_tr, model="additive", freq=4).plot()
plt.show()
```

d) Zerlegen Sie die Zeitreihe erneut, indem Sie die Funktion **`decompose()`** aus dem Paket **`stldecompose`** benützen. Wählen Sie einen geeigneten Wert für den Parameter **`period`**. Vergleichen Sie die Resultate mit denjenigen der Teilaufgabe c).

Kurzlösungen vereinzelter Aufgaben