6. Analýza a zpracování dat

B0B37NSI – Návrh systémů IoT

Stanislav Vítek

Katedra radioelektroniky
Fakulta elektrotechnická
České vysoké učení v Praze

Časové řady

- Časovou řadu lze považovat za uspořádanou posloupnost hodnot proměnné ve (ideálně) stejně dlouhých časových intervalech.
- Experimentální data v podobě časových řad získaných pozorováním reálných dynamických systémů mohou být využita pro konstrukci matematických modelů těchto systémů.
- K modelování takových dat lze použít analýzu časových řad (TSA).
- TSA zohledňuje skutečnost, že datové body pořízené v průběhu času mohou mít vnitřní strukturu (např. autokorelaci, trend nebo sezónní výkyvy),
- Co lze provedením TSA získat?
  - **Popis** Identifikujte vzorce v korelovaných datech, jako jsou trendy a sezónní výkyvy.
  - **Porozumění** Tyto vzorce/modely umožňují rozumět operacím a vazbám, které ovlivňují vznik hodnot časových řad.
  - **Predikce** Při modelování dat lze získat přesné předpovědi budoucích (krátkodobých) trendů.
  - **Intervenční analýza** Lze zkoumat, jak (jednotlivé) události ovlivnily časové řady.
  - **Kontrola kvality** Odchylky v časové řadě mohou naznačovat problémy v procesu, který se v datech odráží.

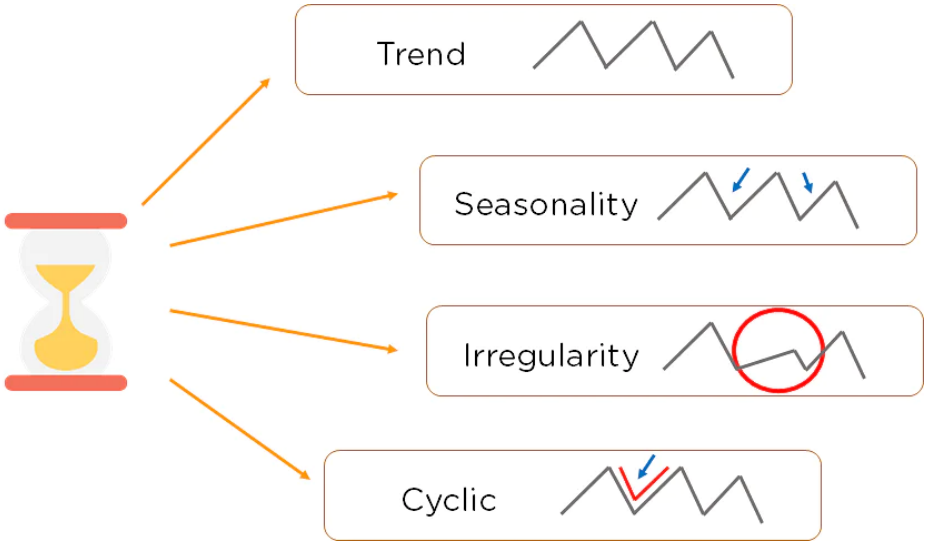
Oblasti uplatnění analýzy časových řad

- Ekonomické prognózy
- Prognózování prodeje
- Rozpočtová analýza
- Analýza akciového trhu
- Odhady výnosů
- Řízení procesů a kvality
- Inventarizační studie
- Odhady pracovního zatížení
- Studie užitečnosti
- Analýza sčítání lidu
- Strategické plánování pracovních sil

Komponenty časových řad 1/2

- Časovou řadu lze dekomponovat na několik samostatných složek. Každá složka popisuje specifické vlastnosti časové řady. Mezi čtyři základní složky patří:
  - **Trend složka** T_t – Vzor, který zachycuje dlouhodobé chování časové řady v čase. Trendová složka se nezabývá krátkodobým růstem či poklesem, a proto se dá vyjádřit matematickou funkcí v celé délce časové řady.
  - **Sezónní složka** S_t – Krátkodobý vzor, který se opakuje v pravidelných intervalech (frekvence výskytu je však rok a méně). Jako hypotetický příklad by se dala použít každodenní návštěvnost nákupních center, kdy se kolem Vánoc rapidně zvýší počet návštěvníků.
  - **Náhodná složka** ϵ_t – Kolísání v datech časové řady neodpovídá trendu nebo sezónnosti. Tyto výkyvy jsou čistě náhodné a obvykle jsou způsobeny nepředvídatelnými okolnostmi, např. náhlým poklesem počtu obyvatel v důsledku přírodní katastrofy.
  - **Cyklická složka** C_t Dlouhodobý vzor, u něhož je frekvence větší jak rok. Oproti sezónní složce je nepravidelná. Většinou vyjadřuje fluktuace (kolísání) kolem trend komponenty. Využití je hlavně v ekonomice, a proto se tato složka někdy ani nezmiňuje. Cyklickou komponentu můžeme využít například při charakteristice deflace a inflace.

Komponenty časových řad 2/2



Modely časových řad

- Abychom mohli dekomponovat časovou řadu na specifické komponenty, je nutné zvolit vhodný model.

Aditivní model

Model použitelný na případy, u kterých víme, že fluktuace sezónní složky kolem trendu jsou v čase relativně konstantní (tzn. se nezvětšují ani nezmenšují, zůstávají pořád stejné).

$$y_t = T_t + S_t + C_t + \epsilon_t$$

Multiplikativní model

Vhodný pro případy, kdy se výkyvy sezónní složky vůči trendu mění v čase.

$$y_t = T_t \cdot S_t \cdot C_t \cdot \epsilon_t$$

$$logy_t = logT_t + logS_t + logC_t + log\epsilon_t.$$

Dekompozice časových řad

- Když známe základní modely časových řad, se můžeme věnovat samotnému procesu dekompozice.
- Dekompozice je způsob, jak lze již z existující časové řady vyextrahovat samotné základní komponenty, případně jejich složené varianty.
- U některých dekompozic se například trendová složka vyskytuje spolu s cyklickou. Z toho důvodu můžeme tuto složku v modelu časové řady úplně vynechat.
- Na dekompozici lze využít několik různých druhů metod, a to od nej primitivnějšího klouzavého průměru až po STL dekompozici.

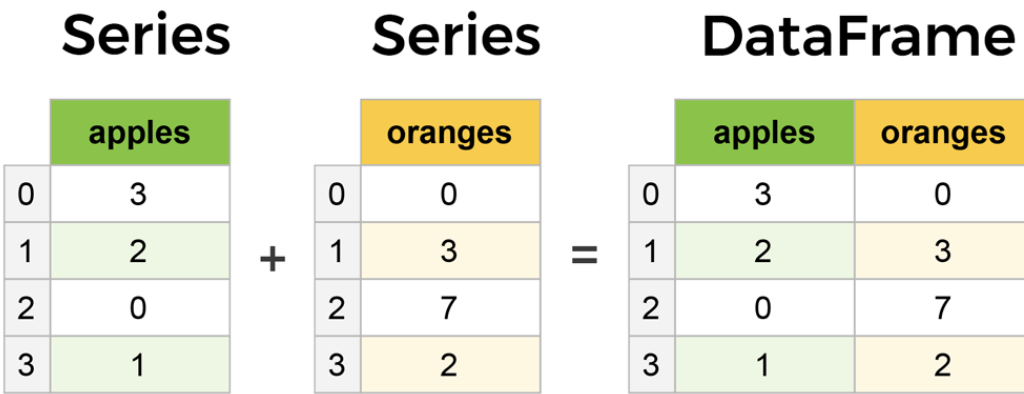
Máme ale k dispozici vhodný nástoj?

Nástroje pro analýzu

- Řekněme, že máme časovou řadu uloženou v CSV / SQL / NoSQL. Chceme:
  - Umět data načíst
  - Vypočítat statistické údaje a odpovědět na otázky týkající se dat, např.
    - Jaký je průměr, medián, maximum nebo minimum každého sloupce?
    - Souvisí sloupec A se sloupcem B?
    - Jak vypadá rozložení dat ve sloupci C?
  - Vyčistit data – odstranění chybějících hodnot a filtrování řádků nebo sloupců podle určitých kritérií.
  - Vizualizovat data – vykreslení sloupců, čár, histogramů, bublin, ...
  - Uložit vyčištěná a transformovaná data zpět do CSV, jiného souboru nebo databáze.
- Co tak třeba Matlab?
- Nebyl by lepší třeba Python?
  - Pandas
    - Postaveno na základech knihovny **NumPy**
    - Vizualizace v **Matplotlib**
    - Algoritmy strojového učení ze **Scikit-learn**

Pandas – klíčové komponenty

- Klíčové kompomenty Pandas jsou `Series` a `DataFrame`
- `Series` je v zásadě sloupe v tabulce popsané `DataFrame`



Vytvoření DataFrame

- Jednoduše pomocí datového typu `dict` (slovník, dictionary)

```
import pandas as pd

data = {
  'apples': [3, 2, 0, 1],
  'oranges': [0, 3, 7, 2]
}

purchases = pd.DataFrame(data)
```

- Index v tabulce je automaticky `0-3`, lze ho ale pojmenovat

```
purchases = pd.DataFrame(
  data,
  index=['June', 'Robert', 'Lily', 'David']
)
```

Čtení zdrojových dat

- Načtení dat z CSV

```
df = pd.read_csv('purchases.csv')
```

- Z CSV se nepřenáší indexy, lze určit, který sloupec bude indexem

```
df = pd.read_csv('purchases.csv', index_col=0)
```

- Z SQL databáze, ke které je třeba se nejprve připojit

```
import sqlite3

con = sqlite3.connect("database.db")
df = pd.read_sql_query("SELECT * FROM purchases", con)
# index lze nastavit samostatnym prikazem
df = df.set_index('index')
```

- Export pomocí metod `to_csv()`, `to_json()`, `to_sql()`

Prohlížení dat a informace o datech

- Prvních 5 (nebo N) řádek tabulky

```
movies_df = pd.read_csv("IMDB-Movie-Data.csv", index_col="Title")
movies.head()
movies.head(10)
```

- Posledních N rádek tabulky

```
movies.tail(10)
```

- Informace o datech

```
movies.info()
```

- Velikost tabulky

```
movies.shape()
```

Hledání duplikátů, práce s daty

- V tabulce žádné duplikáty nejsou, tak nějaké vyrobíme

```
temp_df = movies_df.append(movies_df)
```

- Vyčištění duplikátů

```
temp_df = temp_df.drop_duplicates()
```

- Sloupce tabulky a jejich přejmenování

```
movies_df.columns
movies_df.rename(columns={
  'Runtime (Minutes)': 'Runtime',
  'Revenue (Millions)': 'Revenue_millions'
}, inplace=True)
# nebo treba pro prejmenovani vsech na lower case
movies_df.columns = [col.lower() for col in movies_df]
```

Chybějící data

- Zajímá nás, které řádky mají chybějící údaje
- Takových řádků se buď zbavíme, nebo tam nějaká data vložíme

```
# ktere prvky v DF jsou prazne - vraci True/False
movies_df.isnull()
# agregovana funkce
movies_df.isnull().sum()
```

- Zahození řádků s prázdnými položkami

```
temp_df = movies_df.dropna()
```

- Zahození sloupců s práznými položkami

```
temp_df = movies_df.dropna(axis=1)
```

Doplnění chybějících dat

- Na místa ve sloupci `Revenue (Millions)`, kde chybí data, doplníme střední hodnotu

```
revenue = movies_df['Revenue (Millions)']
revenue_mean = revenue.mean()
revenue.fillna(revenue_mean, inplace=True)
movies_df.isnull().sum()
```

Statistiky

```
# statistika pres cely DF
movies_df.describe()

# statistika vybraneho sloupce
movies_df['Genre'].describe()

# vime, kolik je tam unikatnich zanru, co histogram?
movies_df['Genre'].value_counts().head(10)

# korelace mezi sloupci
movies_df.corr()
```

Hledání v datech

```
# vypis sloupcu
subset = movies_df[['genre', 'rating']]
subset.head()

# hledani hodnot v radcich - hodnota
prom = movies_df.loc["Prometheus"]

# hledani pomoci ciselneho indexu
prom = movies_df.iloc[1]

# slicing
movie_subset = movies_df.loc['Prometheus':'Sing']
movie_subset = movies_df.iloc[1:4]

# podmineny vyber
condition = (movies_df['director'] == "Ridley Scott")
condition.head()

movies_df[movies_df['rating'] >= 8.6].head(3)
```

Volání funkcí

```
# definice hodnotici funkce
def rating_function(x):
  if x >= 8.0:
    return "good"
  else:
    return "bad"

# aplikace funkce
movies_df["rating_category"] = movies_df["rating"].apply(
    rating_function)
movies_df.head(2)

# pomoci lambda funkce
movies_df["rating_category"] = movies_df["rating"].apply(lambda x: '
    good' if x >= 8.0 else 'bad')
movies_df.head(2)

```

Kreslení

```
import matplotlib.pyplot as plt

# nastaveni fontu
plt.rcParams.update({'font.size': 20, 'figure.figsize': (10, 8)})

# vztah mezi obratem a hodnocenim filmu
movies_df.plot(kind='scatter', x='rating', y='revenue_millions',
    title='Revenue (millions) vs Rating');

# histogram
movies_df['rating'].plot(kind='hist', title='Rating');

# vlastnosti sloupce rating
movies_df['rating'].describe()

# vykresleni techto vlastnosti pomoci boxplot
movies_df['rating'].plot(kind="box");

# podle kategorii, co jsme si zavedli
movies_df.boxplot(column='revenue_millions', by='rating_category');
```

Metoda klouzavého průměru

- Pomocí klouzavého průměru se dá jednoduše zjistit trend-cyklická složka
- Klouzavý průměr řádu m je vyjádřen jako

$$\hat{T}_t = \frac{1}{m} \cdot \sum_{j=-k}^{k} y_{t+j}$$

- Míru vyhlazení ovlivňuje parametr $m = 2k + 1$
- Pro $m = 1$ získáme identickou řadu $T_t = \hat{T}_t$
- U klouzavého průměru je důležité, abychom si dali pozor na časové řady s velkou sezónní fluktuací, protože metoda špatně reaguje na okamžité vý kyvy a pro velké hodnoty m můžeme ztratit podstatné informace (sezónnost s vysokou amplitudou, anomálie apod.).