Oblig 4

Behandling av store datamengder

I denne oppgaven skal vi se på trender i salgsdata fra en britisk nettbutikk. Datasettet er åpent tilgjengelig og publisert uten copyright. Datasettet har over 500 000 kolonner, noe som ville gjort det veldig vanskelig å analysere i regnearkprogram som Microsoft Excel eller Apple Numbers.

Innelveringsmåte

I alle oppgaver som ber om forklarende tekst skal du skrive det som kommentarer i koden din. Lever koden din på Canvas.

1. Preprosessering av data

1 a)

Opprett et Python script som heter `oblig4.py` i samme mappe som `data.csv`. Start programmet med å importere `matplotlib.pyplot` og `pandas`

1 b)

Bruk `read_csv` funksjonen i `pandas` biblioteket til å lese inn `data.csv` og lagre innholdet i en variabel, `rådata`. Første kolonnen i datafilen er *indeksen* så husk å sende inn nøkkelordargumentet `index_col=0` når du kaller på `read_csv`. Dette forteller `pandas` hvilken kolonne som inneholder indeksen.

1 c)

`rådata` er nå et `DataFrame` objekt. Kall på metoden `info()` fra objectet for å få informasjon om hva det inneholder. Skriv ut det du får til terminalvinduet.

1 d)

Under er en kodesnutt. Skriv av eller lim inn koden i programmet ditt. Kjør koden og skriv med egne ord hva koden gjør med `DataFrame` objektet og hva resultatet av å kjøre den blir.

```
print(rådata["InvoiceDate"])
rådata["InvoiceDate"] = pd.to_datetime(
    rådata["InvoiceDate"], format="%m/%d/%Y %H:%M"
)
print(rådata.info())
```

1 e)

Hent ut `"Quantity"` fra rådata og lagre det i en variabel, `antall_produkt`. Hent så ut `"UnitPrice"` og lagre det i en variabel, `pris_per_produkt`. Bruk disse

variablene til å regne ut hva den totale salgsprisen blir og lagre det i en variabel, `salgspris`.

1 f)

Å lage en ny kolonne i en `DataFrame` gjøres på samme måte som å legge til en nøkkel i et oppslagsverk (`dictionary`). Legg til en ny kolonne i `rådata` med navn "Salgspris" som inneholder salgsprisen du regnet it i oppgave **1 e)**

1 g)

For å endre indeksen i en `DataFrame` kan man bruke en metode `set_index` og sende inn en streng med navnet på kolonnen man ønsker å bruke som indeks. Bruk `rådata` sin `set_index` metode til å endre indeksen dens til å være gitt av `"InvoiceDate"`. Kall på `rådata` sin `info()` metode for å få informasjon om `DataFramen` igjen og skriv ut det du får til terminalen. Hva har endret seg etter `set_indeks` kallet?

1 h)

Under er en kodesnutt. Lim inn eller skriv av koden inn i programmet ditt. Kjør koden. Forklar med egne ord hva forskjellen på `månedssalg` og `dagssalg` er. (**Hint:** se på indeksen, **Hint:** `månedssalg.head)()` gir en ny `DataFrame` som inneholder de fem første radene av `månedssalg`)

```
månedsgrupper = rådata["Salgspris"].groupby(pd.Grouper(freq="M"))
månedssalg = månedsgrupper.sum()

dagsgrupper = rådata["Salgspris"].groupby(pd.Grouper(freq="D"))
dagssalg = dagsgrupper.sum()

print(månedssalg.head())
print(dagssalg.head())
```

2. Datavisualisering med Pandas

2 a)

Når vi har tidsserie-data er det ofte mye jobb å plotte det på en fin måte. Vi kan plotte månedsdataen slik:

`DataFrame` objekter fra `Pandas` biblioteket har en egen plottemetode vi kan bruke slik:

```
linjeplot_figur = plt.figure()
månedssalg_akser = linjeplot_figur.add_subplot(2, 1, 1)
# Linja over sier at vi ønsker to rader og en kolonne med akser
# Så ber den om første settet med akser (det øverste)
månedssalg.plot(ax=månedssalg_akser)
```

```
plt.tight_layout()
# Linja over forteller matplotlib at det er trangt
# så vi må gjøre aksene våre mindre enn normalt.
plt.show()
```

Modifiser koden over slik at du plotter både `månedssalg` variabelen og `dagssalg` variabelen i hver sin akse. `månedssalg`-plottet skal tegnes over `dagssalg` plottet.

2b)

Plottet til dags-data variabelen skal se slik ut:

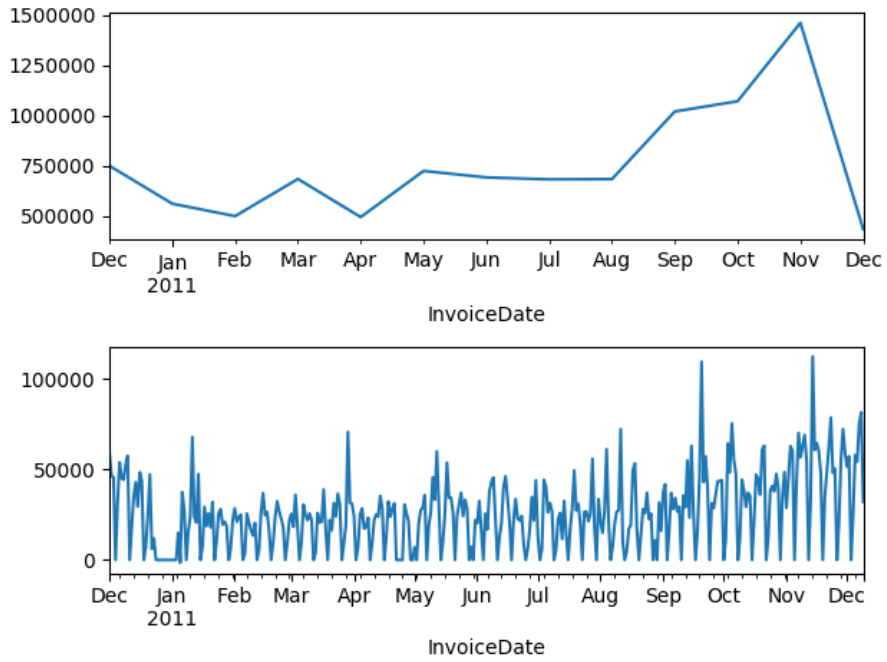

Figure 1: dagsdata plot

Er det noe pussig med noen av plottene?

3. Datavisualisering med Matplotlib

Pandas gjør ofte jobben vår mye lettere når vi skal visualisere data. Men, vi har ikke like mye frihet og kontroll med Pandas som vi har med Matplotlib. Dessuten blir Matplotlib brukt av Pandas hver gang du ber Pandas om å lage en figur! Derfor er det og veldig nyttig å vite hvordan vi kan lage plot med Matplotlib og!

3a)

Bruk denne koden for å regne ut gjennomsnittsprisen hvor hver ukedag.

```
ukedag = rådata.index.weekday
ukedagsgrupper = rådata["Salgspris"].groupby(ukedag)
ukedagssalg = ukedagsgrupper.sum()
print("Salg per ukedag:")
print(ukedagssalg)
# Ukedag 0 er mandag, ukedag 6 er søndag
```

Forklar hva hver linje i koden over gjør.

3b)

Er det noen ukedager vi ikke har data fra? Isåfall, kan du sette det i kontekst av en tidligere oppgave?

3c)

Kopier koden

```
dag_navn = [
    "Mandag", "Tirsdag", "Onsdag", "Torsdag",
    "Fredag", "Lørdag", "Søndag"
]
ukedagsdata_dagnummer = ukedagssalg.indeks
print(ukedager)
```

til bunnen av programmet ditt. Bruk en listebeskrivelse (list comprehension) for å lage en ny variabel `ukedagssalg_dagnavn` med like mange elementer som `ukedagssalg_dagnummer`. Første element i `ukedagssalg_dagnavn` skal være navnet til dagen som representeres av første element i `ukedagssalg_dagnummer` osv.

3d)

Vi kan bruke `plt.bar(kategorinavn, verdier)` for å tegne et **stolpediagram**. Hver kategori får en plassering på førsteaksen (x-aksen). For hver kategori tegnes det en stolpe som når opp til den korresponderende plasseringen langs y-aksen. Her er et eksempel:

```
x = ["A", "B", "C"]
y = [ 3 ,  1 ,  2 ]

figur = plt.figure()
akser = figur.add_subplot()

akser.bar(x, y)
```

```
plt.show()
```

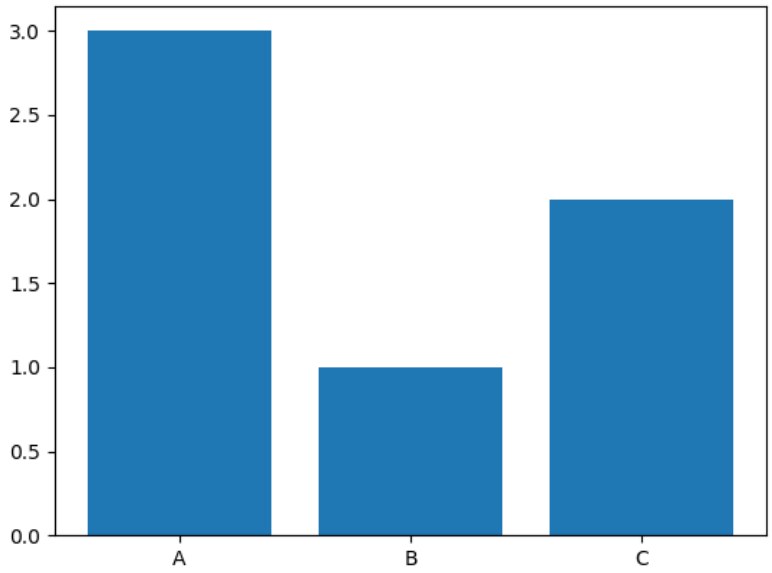

Figure 2: søylediagram

Lag et søylediagram hvor vi har ukedager på førsteaksen (x-aksen) og totalt salg for den dagen på andreaksen (y-aksen).