Analiza danych dotyczących transportu kojewego we Francji

Autor: Mateusz Śliwka

Wstęp

Zbiór danych pochodzi z serwisu Kaggle.com (https://www.kaggle.com/aslanduru/sncftrain-traffic-data-in-france). Liczba kolumn: 32 Liczba wierszy: 7806

In [1]:
```
# użycie bibliotek:

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
data_set = pd.read_csv('Regularities_by_liaisons_Trains_France.csv')
data_set.head()
```

Out[2]:

	Year	Month	Departure station	Arrival station	Average travel time (min)	Number of expected circulations	Number of cancelled trains	Number of trains at departure	Average delay of trains at departure (min)	Average delay of trains at departing trains (min)	Average number of delays > 15min	Number of delays > 30min	Number of delays > 60min	Pe
0	2018	7.0	ANGOULEME	PARIS MONTPARNASSE	131.914980	247.0	0.0	191.0	3.574353	2.678273	32.980873	7.0	2.0	20
1	2018	7.0	PARIS MONTPARNASSE	LA ROCHELLE VILLE	175.012570	242.0	0.0	178.0	9.760905	7.033609	32.057143	14.0	2.0	20
2	2018	7.0	LE MANS	PARIS MONTPARNASSE	62.385348	430.0	5.0	391.0	3.999974	3.529341	42.967241	13.0	4.0	20
3	2018	7.0	ST MALO	PARIS MONTPARNASSE	157.411053	114.0	0.0	101.0	1.950990	1.695673	27.020833	2.0	0.0	20
4	2018	7.0	PARIS MONTPARNASSE	ST PIERRE DES CORPS	67.310000	404.0	4.0	294.0	8.379106	5.903225	37.656333	12.0	3.0	20

5 rows × 32 columns

Do ostatecznej analizy zostały użyte kolumny dotyczą roku oraz miesiąca, w którym wystąpiły opóźnienia oraz liczba opóźnień w podziale na rodzaj przyczyny, która spowodowała opóźnienie.

- Delay due to external causes' - opóźnienia z przyczyn zewnętrznych (pogoda, przeszkody, podejrzane pakunki, zbolitwe zamiany, ruchy społeczne itp.)
- Delay due to railway infrastructure' - opóźnienia z powodu infrastruktury kolejowej (prace konserwacyjne)
- Delay due to traffic management' - opóźnienia pociągów spowodowanych zarządzaniem ruchem (ruch na linii kolejowej, interakcje w sieci)
- Delay due to rolling stock' - opóźnienia ze względu na tabor kolejowy
- Delay due to station management and reuse of material' - opóźnienia pociągów spowodowanych zarządzaniem stacjami i ponownym wykorzystaniem materiałów
- Delay due to travellers taken into account' - opóźnienia spowodowane ruchem pasażerskim

In [4]:
```
df = data_set[['Year', 'Month','Delay due to external causes',
               'Delay due to railway infrastructure',
               'Delay due to traffic management', 'Delay due to rolling stock',
               'Delay due to station management and reuse of material',
               'Delay due to travellers taken into account']].copy()
df.head()
```

Out[4]:

	Year	Month	Delay due to external causes	Delay due to railway infrastructure	Delay due to traffic management	Delay due to rolling stock	Delay due to station management and reuse of material	Delay due to travellers taken into account
0	2018	7.0	25.000000	15.000000	27.000000	12.000000	2.000000	17.000000
1	2018	7.0	23.000000	19.444444	35.666667	19.444444	0.000000	4.444444
2	2018	7.0	16.178471	32.352941	26.470588	14.705882	2.941176	7.352941
3	2018	7.0	31.394615	15.384615	23.076923	26.461538	0.000000	7.692308
4	2018	7.0	18.463108	12.307692	40.000000	16.923077	7.692308	4.615385

In [5]:
```
# wizualizacja występowania wartości pustych
sns.heatmap(df.isnull(), yticklabels = False, cbar = False, cmap = 'viridis')
```

Out[5]:

In [6]:
```
df.dropna(inplace=True)
sns.heatmap(df.isnull(), yticklabels = False, cbar = False, cmap = 'viridis')
```

Out[6]:

In [7]:
```
# stworzenie kolumny 'Date', zawierającej daty
def month_number_to_text(number):
    if number==1.00:
        return 'January'
    elif number==2.00:
        return 'February'
    elif number==3.00:
        return 'March'
    elif number==4.00:
        return 'April'
    elif number==5.00:
        return 'May'
    elif number==6.00:
        return 'June'
    elif number==7.00:
        return 'July'
    elif number==8.00:
        return 'August'
    elif number==9.00:
        return 'September'
    elif number==10.00:
        return 'October'
    elif number==11.00:
        return 'November'
    elif number==12.00:
        return 'December'
    else:
        return 'Error'

df['Month'] = df.Month.apply(month_number_to_text)
df['Date'] = pd.to_datetime(df['Year'].astype(str) + df['Month'], format='%Y%B')
```

In [8]:
```
df['Delay due to external causes'] = df['Delay due to external causes'].round(decimals=0)
df['Delay due to railway infrastructure'] = df['Delay due to railway infrastructure'].round(decimals=0)
df['Delay due to traffic management'] = df['Delay due to traffic management'].round(decimals=0)
df['Delay due to rolling stock'] = df['Delay due to rolling stock'].round(decimals=0)
df['Delay due to station management and reuse of material'] = df['Delay due to station management and reuse of material'].round(decimals=0)
df['Delay due to travellers taken into account'] = df['Delay due to travellers taken into account'].round(decimals=0)
```

In [9]:
```
df.groupby('Year').sum()
```

Out[9]:

Year	Delay due to external causes	Delay due to railway infrastructure	Delay due to traffic management	Delay due to rolling stock	Delay due to station management and reuse of material	Delay due to travellers taken into account
2015	39211.0	32853.0	25303.0	19326.0	10672.0	5655.0
2016	38944.0	34318.0	23457.0	17935.0	11143.0	5260.0
2017	34547.0	33104.0	27521.0	26272.0	6913.0	4349.0
2018	36330.0	34173.0	30946.0	32179.0	9749.0	4820.0
2019	37146.0	33747.0	33864.0	27985.0	9351.0	4779.0
2020	13537.0	22966.0	10635.0	11036.0	3930.0	2570.0

In [10]:
```
# Zdefiniowanie kolejności występowania miesięcy
months = ['January', 'February', 'March', 'April', 'May', 'June',
          'July', 'August', 'September', 'October', 'November', 'December']
df['Month'] = pd.Categorical(df['Month'], categories=months, ordered=True)
```

In [11]:
```
df.Year.value_counts().sort_index().plot.bar()
```

Out[11]:

In [26]:
```
df.Month.value_counts().sort_index().plot()
plt.show()
```


In [28]:
```
sns.heatmap(df.drop('Year',axis=1).corr(), annot=True, cmap = 'Reds')
plt.show()
```


Time series analysis

In [30]:
```
df['Date'].min()
```

Out[30]:
```
Timestamp('2015-01-01 00:00:00')
```

In [31]:
```
df['Date'].max()
```

Out[31]:
```
Timestamp('2020-06-01 00:00:00')
```

In [32]:
```
# stworzenie dataframe'ów potrzeby stworzenia szeregu czasowych
ts_external = df[['Delay due to external causes','Date']].set_index('Date').groupby('Date').mean().copy()
ts_railway = df[['Delay due to railway infrastructure','Date']].set_index('Date').groupby('Date').mean().copy()
ts_management = df['Delay due to traffic management','Date']].set_index('Date').groupby('Date').mean().copy()
ts_stock = df[['Delay due to rolling stock','Date']].set_index('Date').groupby('Date').mean().copy()
ts_material = df[['Delay due to station management and reuse of material','Date']].set_index('Date').groupby('Date').mean().copy()
ts_account = df[['Delay due to travellers taken into account','Date']].set_index('Date').groupby('Date').mean().copy()
```

In [33]:
```
# narzędzie użyte do analizy szeregów czasowych
from statsmodels.tsa.seasonal import seasonal_decompose
```

In [34]:
```
import matplotlib.pyplot as plt
from matplotlib import rcParams
from cycler import cycler

rcParams['figure.figsize'] = 10, 5
rcParams['axes.spines.top'] = False
rcParams['axes.spines.right'] = False
rcParams['axes.prop_cycle'] = cycler(color=['#3695F7'])
rcParams['lines.linewidth'] = 2.5
```

In [34]:
```
# dekompozycja szeregów
decomposed_external = seasonal_decompose(ts_external, model='additive')
decomposed_railway = seasonal_decompose(ts_railway, model='additive')
decomposed_management = seasonal_decompose(ts_management, model='additive')
decomposed_stock = seasonal_decompose(ts_stock, model='additive')
decomposed_material = seasonal_decompose(ts_material, model='additive')
decomposed_account = seasonal_decompose(ts_account, model='additive')
```

In [35]:
```
# wyświetlenie wyników dekompozycji obok siebie
import matplotlib.pyplot as plt
import statsmodels.api as sm

def plotseasonal(res, axes, title ):
    res.observed.plot(ax=axes[0], legend=False)
    axes[0].set_title(title, fontweight="bold")
    axes[0].set_ylabel('Observed')
    res.trend.plot(ax=axes[1], legend=False)
    axes[1].set_ylabel('Trend')
    res.seasonal.plot(ax=axes[2], legend=False)
    axes[2].set_ylabel('Seasonal')
    res.resid.plot(ax=axes[3], legend=False)
    axes[3].set_ylabel('Residual')

fig, axes = plt.subplots(ncols=2, nrows=4, sharex=True, figsize=(12,5))

plotseasonal(decomposed_external, axes[:,0], 'Delay due to external causes')
plotseasonal(decomposed_railway, axes[:,1], 'Delay due to railway infrastructure')


plt.tight_layout()
plt.show()
```


In [36]:
```
fig1, axes1 = plt.subplots(ncols=2, nrows=4, sharex=True, figsize=(12,5))
plotseasonal(decomposed_management, axes1[:,0], 'Delay due to traffic management')
plotseasonal(decomposed_stock, axes1[:,1], 'Delay due to rolling stock')
plt.tight_layout()
plt.show()
```


In [37]:
```
fig2, axes2 = plt.subplots(ncols=2, nrows=4, sharex=True, figsize=(12,5))
plotseasonal(decomposed_material, axes2[:,0], 'Delay due to station management and reuse of material')
plotseasonal(decomposed_account, axes2[:,1], 'Delay due to travellers taken into account')
plt.tight_layout()
plt.show()
```


Wnioski

- W 2020 nastąpił spadek zgłaszanych opóźnień, ale należy mieć na względzie fakt, iż dane dla roku 2020 zawierają dane tylko do połowy roku. Pozostały spadek zgłaszanych opóźnień w tym roku mógł zostać spowodowany spadkiem liczby kursów pociągów z powodu pandemii COVID-19.
- Liczba zgłaszanych opóźnień w pierwszej połowie roku jest większa niż w drugiej połowie roku.
- Istnieje stała korelacja pomiędzy spadkiem liczby opóźnień spowodowanych pracami konserwacyjnymi a wzrostem liczby opóźnień pociągów spowodowanymi warunkami atmosferycznymi, w okresach zimowych wykonuje się mniej prac remontowych, a śnieg często powoduje opóźnienia na kolej.
- Wzrost opóźnień spowodowanych warunkami atmosferycznymi wykazuje sezonowość, szczytowek liczba takich opóźnień mamy rok do roku.
- Liczba opóźnień spowodowanych zarządzaniem ruchem wykazuje wzrostowy trend w ciągu ostatnich lat.
- Liczba opóźnień spowodowanych zarządzaniem ruchem wystąpiła sezonowo i rośnie rok do roku, co może być spowodowane wzrostem liczby kursów.
- Spółczynnika wynikający z powodu uszkodzenia taboru wykazują charakterystyczną się stałością występowania.
- Opóźnienia pociągów spowodowanych zarządzaniem stacjami i ponownym wykorzystaniem materiałów występują sezonowo.
- Opóźnienia spowodowane ruchem pasażerskim występują sezonowo, głównie w okresie wakacyjnym i świątecznym.

In [ ]:
```

```