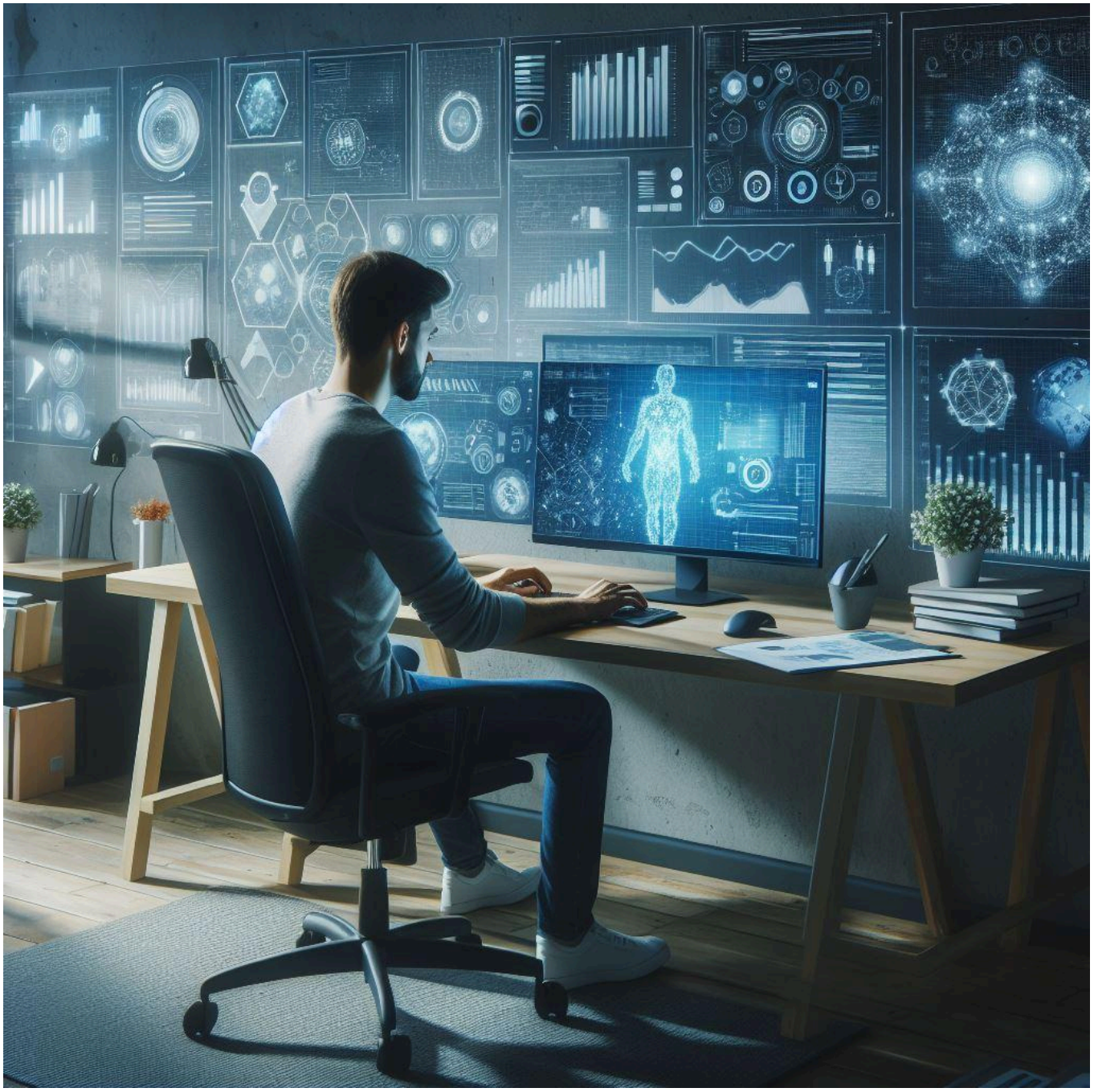
Personal Development Report



Create date: 4 March 2024
Last update: 15 March 2024
Author: **A**nne **E**rik **J**ozef **K**oppers

About me and this semester

This semester I am going to work on and learn what AI means. This wil incleud the internals of finfing out how to read data and what to expect from outcoms if i were to read or write a ai program using the information given in this semester.

I chose AI because I want to learn more about it, I had I small workshop on my last school. This is also why I didn’t choose game disign.

My name is Anne Erik Jozef Koppers, I live in the netherlands and have been born her. Have a mom and dad. More should not be needed and shal most likely be forgotton. If you ask more I would point out that I am not forced to but that down.

Index:

LO1: Data Preparation & Analysis

Intro:

You are able to **aggregate** and **prepare** given datasets as well as other (open) datasets and use them in **data analysis** and identify **opportunities for predictive analytics.**

Aggregate means acquiring data from a variety of different sources and in different formats and putting it together into a meaningful larger total dataset.

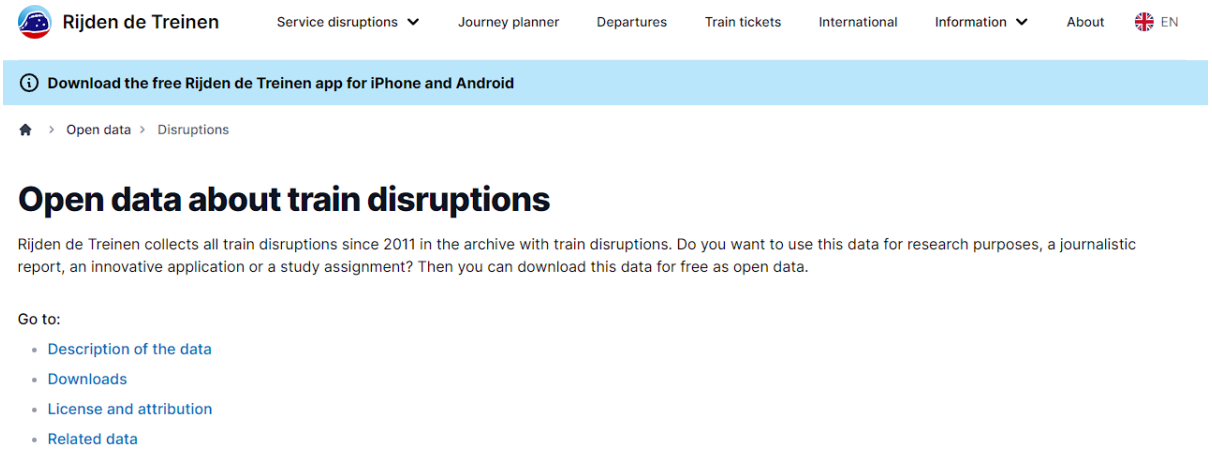
Prepare consists of cleaning the data according to theories of data quality, in such a way that the process of cleaning and preparing those data is repeatable, transparent to others, and the results are suitable for data analysis.

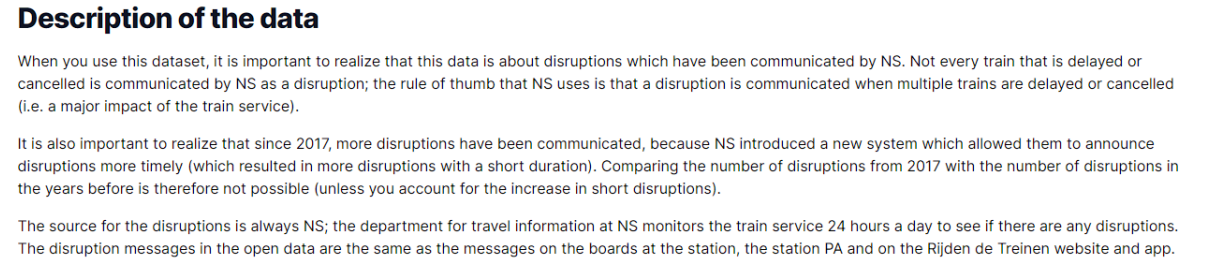
Data analysis implies amongst others: descriptive analytics, statistical overviews, derived columns, trend analysis, etc.

Opportunities for predictive analytics can be identified by finding correlations between features, principle component analysis, summarization, anomaly detection, etc. and include an impact forecast.

What I did for this LO:

For my data preparation, I started off by looking for working and reliable data. This is when for the disruptions I found a site that has all the disruptions in the Netherlands. This site is also used by news outlets in the Netherlands so I hope for something as the news that their source is correct. The website's name is rijdendetrijnen.nl





I also use the dataset retrieved from KNMI.nl, seeing that this is a government-founded operation It can be stated as true and trustworthy in the Netherlands.

Another set is the locations of most train stations. This I got from kaggle.com and got a good score for reliability. This is to see what weather station is closest.

Train Stations in Europe

Names, Coordinates, and Properties of European Railway Stations

Data Card | Code (2) | Discussion (0) | Suggestions (0)

About Dataset

Context

Many European countries possess an extensive net of public transport railroads which connect large and small cities. This dataset contains the names, coordinates, and basic properties of more than 36000 train stations in (and adjacent to) Europe. It was derived from data provided by the Trainline EU ticketing website that has been published on github. I will update this dataset regularly.

Note, that the data contains a few train stations in the European parts of Russia and Turkey, as well as a small number of stations in the African country of Morocco.

Content

The dataset `train_stations_europe.csv` is based on the Trainline EU github repo. It contains 36k+ stations at the time of creation of this Kaggle Dataset. The github dataset contains many more columns, most of which are covering operator-specific properties (e.g. Renfe or Trenitalia) or translations into different languages (most of which are missing, though). I decided to extract this subset to provide a more focussed and complete data source.

Usability ⓘ
10.00

License
Database: Open Database, Cont...

Expected update frequency
Quarterly

Tags
Transportation | Europe | Geospatial Analysis

In programming, I load the data and do some setup:

1. Disruption

```
disruptions_2023 = pd.read_csv(datafolder + 'disruptions-2023.csv', sep=',', low_memory=False, encoding='utf-8')

# Convert the 'start_time' and 'end_time' column to datetime
disruptions_2023['start_time'] = pd.to_datetime(disruptions_2023['start_time'], format="%Y-%m-%d %H:%M:%S")
disruptions_2023['end_time'] = pd.to_datetime(disruptions_2023['end_time'], format="%Y-%m-%d %H:%M:%S")
# Filter the DataFrame to end of march
disruptions_2023 = disruptions_2023[disruptions_2023['start_time'] < datetime(2023, 3, 31, 23, 59, 59)]

dutch_text = False
if dutch_text  :
    #dutch
    disruptions_2023 = disruptions_2023.iloc[:,[2,4,6,8,10,11,12,13]]
else:
    # english
    disruptions_2023 = disruptions_2023.iloc[:,[2,4,7,9,10,11,12,13]]

display(disruptions_2023)
```

✓ 0.0s

	rdt_lines	rdt_station_names	cause_en	statistical_cause_en	cause_group	start_time	end_time	duration_minutes
0	Amsterdam Centraal - Schiphol Airport, Rotterd...	Amsterdam Centraal,Amsterdam Lelylaan,Amsterda...	points failure	points failure	infrastructure	2023-01-01 08:19:26	2023-01-01 22:43:08	864.0
1	Leeuwarden - Zwolle	Heerenveen,Wolvega,Heerenveen IJsstadion	an animal on the railway track	an animal on the railway track	external	2023-01-01 10:31:49	2023-01-01 10:56:17	24.0
2	Aachen Hbf - Heerlen	Aachen Hbf,Eygelshoven Markt,Heerlen,Heerlen D...	problems with the rolling stock	problems with the rolling stock	rolling stock	2023-01-01 13:19:24	2023-01-02 00:02:39	643.0
3	Winterswijk - Zutphen	Vorden,Zutphen	collision	collision	accidents	2023-01-01 17:15:22	2023-01-01 20:14:23	179.0
4	Aachen Hbf - Heerlen	Aachen Hbf,Eygelshoven Markt,Heerlen,Heerlen D...	problems with the rolling stock	problems with the rolling stock	rolling stock	2023-01-02 05:57:27	2023-01-03 02:07:13	1210.0
...	...	...	...	...	...	...	...	...
1218	Breda - Roosendaal	Breda,Etten-Leur,Roosendaal	broken down train	broken down train	rolling stock	2023-03-31 10:44:58	2023-03-31 10:55:53	11.0
1219	Leiden Centraal - Schiphol Airport	Hoofddorp,Leiden Centraal,Nieuw Vennep,Schipho...	broken down train	broken down train	rolling stock	2023-03-31 12:21:49	2023-03-31 12:27:27	6.0
1220	Arnhem Centraal - Basel SBB, Arnhem Centraal -...	Arnhem Centraal,Arnhem Velperpoort,Duiven,Emme...	broken down train	broken down train	rolling stock	2023-03-31 17:48:22	2023-03-31 18:17:16	29.0
1221	Leiden Centraal - Schiphol Airport	Hoofddorp,Schiphol Airport	broken down train	broken down train	rolling stock	2023-03-31 19:17:20	2023-03-31 19:27:58	11.0
1222	Arnhem Centraal - Zutphen	Brummen,Dieren,Zutphen	broken down train	broken down train	rolling stock	2023-03-31 19:21:04	2023-03-31 19:48:51	28.0

Disruption holds a good amount of information, but I removed the other language because that is redundant data. I will still need to see what I keep and what to remove.

2. weather

```
weather_2023 = pd.read_csv(datafolder + '2023-jan-mar.csv', sep=',', low_memory=False, encoding='utf-8', skiprows=80)
display(weather_2023)

weather_2023['start_time'] = weather_2023['start_time'].astype(str) + ' ' + (weather_2023['HH'] - 1).astype(str) + ':00:00'

weather_2023['start_time'] = pd.to_datetime(weather_2023['start_time'], format="%Y%m%d %H:%M:%S")
#  drop columns that are not needed
# weather_2023 = weather_2023.drop(columns=['HH', 'WW', 'IX', 'FH', 'FX', 'TD', 'SQ', 'RH'])
weather_2023 = weather_2023.drop(columns=['HH'])

display(weather_2023.head(12))
```

✓ 0.3s

	STN	start_time	HH	DD	FH	FF	FX	temp	T10N	TD	...	VV	N	U	WW	IX	M	R	S	O	Y
0	209	20230101	1	220.0	140.0	150.0	170.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
1	209	20230101	2	220.0	130.0	130.0	170.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
2	209	20230101	3	240.0	140.0	130.0	200.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
3	209	20230101	4	220.0	120.0	130.0	170.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
4	209	20230101	5	230.0	130.0	140.0	170.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
99355	391	20230331	20	210.0	50.0	40.0	120.0	99.0	NaN	88.0	...	NaN	NaN	93.0	NaN	6	NaN	NaN	NaN	NaN	NaN
99356	391	20230331	21	220.0	40.0	50.0	100.0	99.0	NaN	86.0	...	NaN	NaN	91.0	NaN	6	NaN	NaN	NaN	NaN	NaN
99357	391	20230331	22	210.0	40.0	40.0	110.0	97.0	NaN	87.0	...	NaN	NaN	93.0	NaN	6	NaN	NaN	NaN	NaN	NaN
99358	391	20230331	23	220.0	40.0	40.0	90.0	95.0	NaN	88.0	...	NaN	NaN	95.0	NaN	6	NaN	NaN	NaN	NaN	NaN
99359	391	20230331	24	220.0	50.0	50.0	110.0	94.0	96.0	88.0	...	NaN	NaN	95.0	NaN	6	NaN	NaN	NaN	NaN	NaN

99360 rows × 25 columns

	STN	start_time	DD	FH	FF	FX	temp	T10N	TD	SQ	...	VV	N	U	WW	IX	M	R	S	O	Y
0	209	2023-01-01 00:00:00	220.0	140.0	150.0	170.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
1	209	2023-01-01 01:00:00	220.0	130.0	130.0	170.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
2	209	2023-01-01 02:00:00	240.0	140.0	130.0	200.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
3	209	2023-01-01 03:00:00	220.0	120.0	130.0	170.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
4	209	2023-01-01 04:00:00	230.0	130.0	140.0	170.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
5	209	2023-01-01 05:00:00	220.0	130.0	130.0	170.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
6	209	2023-01-01 06:00:00	220.0	130.0	120.0	160.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
7	209	2023-01-01 07:00:00	230.0	120.0	110.0	160.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
8	209	2023-01-01 08:00:00	220.0	110.0	110.0	130.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
9	209	2023-01-01 09:00:00	220.0	110.0	110.0	140.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
10	209	2023-01-01 10:00:00	220.0	120.0	120.0	150.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN
11	209	2023-01-01 11:00:00	220.0	120.0	130.0	150.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN

12 rows × 24 columns

With the weather, I lood it in and set the time merge the hour with the day, and remove the hour table. I might need to change the names of the tables some more

here.

```
weather_station_location = pd.read_csv(datafolder + 'Weather-stations-loc.csv', sep=',', low_memory=False, encoding='utf-8')
display(weather_station_location)
```

[5] ✓ 0.0s

	STN	LON(east)	LAT(north)	ALT(m)	NAME
0	209	4.518	52.465	0.0	IJmond
1	210	4.430	52.171	-0.2	ValkenburgZh
2	215	4.437	52.141	-1.1	Voorschoten
3	225	4.555	52.463	4.4	IJmuiden
4	235	4.781	52.928	1.2	DeKooy
5	240	4.790	52.318	-3.3	Schiphol
6	242	4.921	53.241	10.8	Vlieland
7	248	5.174	52.634	0.8	Wijdenes
8	249	4.979	52.644	-2.4	Berkhout
9	251	5.346	53.392	0.7	HoornTerschelling
10	257	4.603	52.506	8.5	WijkaanZee
11	258	5.401	52.649	7.3	Houtribdijk
12	260	5.180	52.100	1.9	DeBilt
13	265	5.274	52.130	13.9	Soesterberg
14	267	5.384	52.898	-1.3	Stavoren
15	269	5.520	52.458	-3.7	Lelystad
16	270	5.752	53.224	1.2	Leeuwarden
17	273	5.888	52.703	-3.3	Marknesse
18	275	5.873	52.056	48.2	Deelen
19	277	6.200	53.413	2.9	Lauwersoog
20	278	6.259	52.435	3.6	Heino
21	279	6.574	52.750	15.8	Hoogeveen
22	280	6.585	53.125	5.2	Eelde
23	283	6.657	52.069	29.1	Hupsel
24	285	6.399	53.575	0.0	Huibertgat
25	286	7.150	53.196	-0.2	NieuwBeerta
26	290	6.891	52.274	34.8	Twenthe
27	308	3.379	51.381	0.0	Cadzand
28	310	3.596	51.442	8.0	Vlissingen

The weather station dataset also held a set of coordinates of the weather stations. And i am thinking of dropping the altitude because they seem in my opinion nothing to do with me trying to find out if the weather has a connection to train disturbances.

```python
europa_train_station_locations_original = pd.read_csv(datafolder + 'train_stations_europe.csv', low_memory=False, encoding='utf-8')

europa_train_station_locations_original = europa_train_station_locations_original.dropna(subset=['latitude'])

display(europa_train_station_locations_original)

europa_train_station_locations = europa_train_station_locations_original.copy()

europa_train_station_locations['country'] = pd.Categorical(europa_train_station_locations['country'], categories=['NL', 'DE', 'BE'], ordered=True)

europa_train_station_locations.sort_values('country', inplace=True)

# Convert 'country' back to a regular column (optional)
europa_train_station_locations['country'] = europa_train_station_locations['country'].astype(str)

europa_train_station_locations = europa_train_station_locations.drop_duplicates(subset='name').drop_duplicates(subset='name_norm')

local_train_station_locations = pd.DataFrame(columns=['NAME', 'latitude', 'longitude'])
local_train_station_locations['NAME'] = lose_disruptions_with_station_names['rdt_station_names'].unique()


europa_train_station_locations['name'] = europa_train_station_locations['name'].str.replace('[^a-zA-Z0-9 ]', '')
europa_train_station_locations['name_norm'] = europa_train_station_locations['name_norm'].str.replace('[^a-zA-Z0-9 ]', '')

local_train_station_locations['NAME'] = local_train_station_locations['NAME'].str.replace('[^a-zA-Z0-9 ]', '')


display(local_train_station_locations)



# First, set 'location' as the index in both dataframes for easier merging
europa_train_station_locations.set_index('name', inplace=True)
local_train_station_locations.set_index('NAME', inplace=True)

# Now, fill in the missing coordinates in df2 from df1
local_train_station_locations.update(europa_train_station_locations)

europa_train_station_locations.set_index('name_norm', inplace=True)

local_train_station_locations.update(europa_train_station_locations)

# Reset the index to revert 'location' back to a column
local_train_station_locations.reset_index(inplace=True)
europa_train_station_locations.reset_index(inplace=True)

display(local_train_station_locations)
```

✓ 0.2s

	id	name	name_norm	uic	latitude	longitude	parent_station_id	country	time_zone	is_city	is_main_station	is_airport	entur_id	entur_is_enabled
0	1	Château-Arnoux—St-Auban	Chateau-Arnoux-St-Auban	NaN	44.081790	6.001625	NaN	FR	Europe/Paris	True	False	False	NaN	False
1	2	Château-Arnoux—St-Auban	Chateau-Arnoux-St-Auban	8775123.0	44.061565	5.997373	1.0	FR	Europe/Paris	False	True	False	NaN	False
2	3	Château-Arnoux Mairie	Chateau-Arnoux Mairie	8775122.0	44.063863	6.011248	1.0	FR	Europe/Paris	False	False	False	NaN	False
3	4	Digne-les-Bains	Digne-les-Bains	NaN	44.350000	6.350000	NaN	FR	Europe/Paris	True	False	False	NaN	False
4	6	Digne-les-Bains	Digne-les-Bains	8775149.0	44.088710	6.222982	4.0	FR	Europe/Paris	False	True	False	NaN	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Here I load in all the coordinates of the train stations in Germany, Belgium, and the Netherlands. I also link them to the train stations inside my dataset.

LO2: Model Engineering

Intro:

You are able to use **findings from data analysis** to **preprocess** data, **apply** machine learning algorithms and **evaluate** the quality and usefulness of produced models, for a **defined domain.**

Findings from data analysis implies that your choice of data sources and feature selection is based on opportunities for predictive analytics that you previously identified.

Preprocess refers to applying systematic ways like feature selection, encoding, scaling, etc. of turning raw datasets into formats that are more suitable for model training.

Apply consists of training of different types of models like classification, regression, etc., as well as tuning hyper-parameters.

Evaluate means judging the results of machine learning with respect to recall, precision, accuracy, cross-validation, over/underfitted etc.

A **defined domain** refers to the fact that your evaluation must address the problem and impact definition as given by the domain stakeholders, and evaluation metrics must be translated to be meaningful to them.

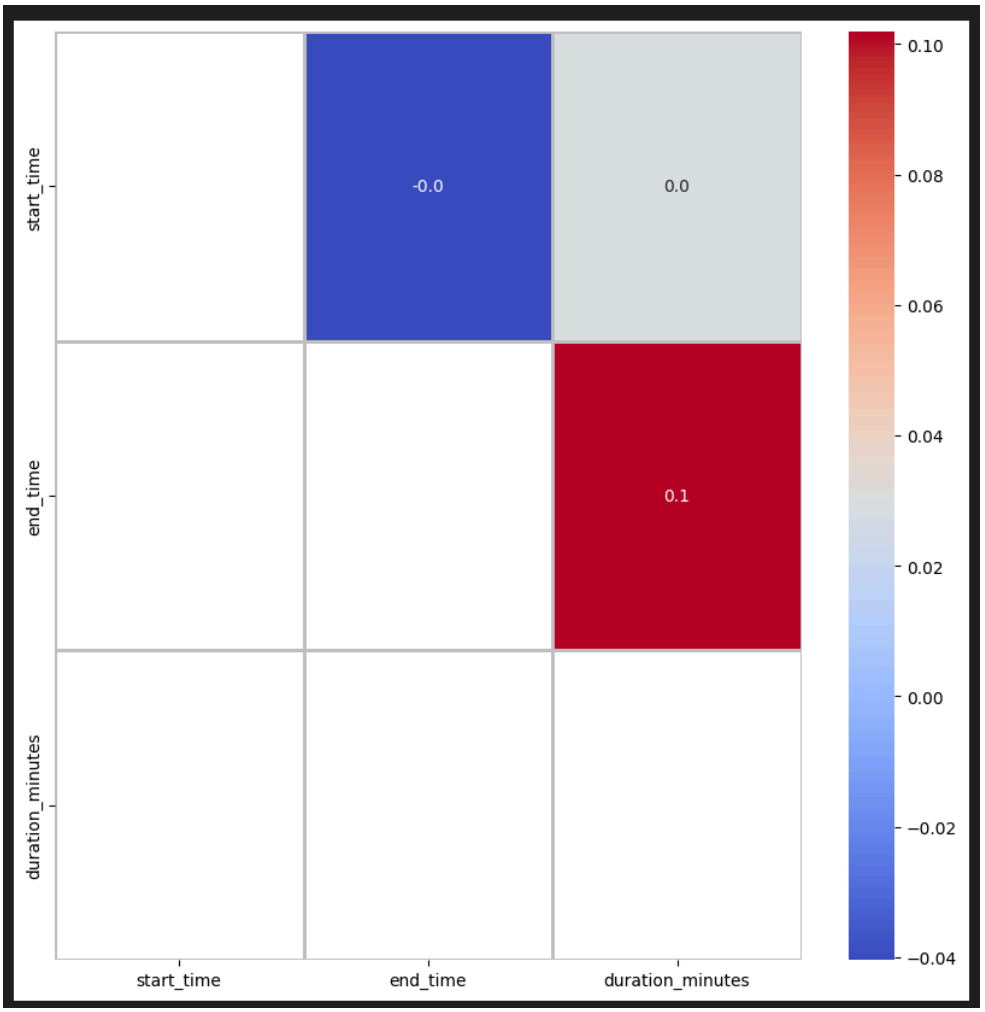
What I did for this LO:

```
Corrolation_disruptions_2023 = disruptions_2023.copy()


# drop all that isn't a number

Corrolation_disruptions_2023 = Corrolation_disruptions_2023.drop(columns=['rdt_station_names', 'statistical_cause_en', 'cause_group', 'rdt_lines'])


display(Corrolation_disruptions_2023)
# some basic visualization for numbered_lose_disruptions_2023
plt.figure(figsize=(10,10))
mask = np.tril(Corrolation_disruptions_2023.corr())
sns.heatmap(Corrolation_disruptions_2023.corr(), cmap="coolwarm", annot=True, fmt=".1f", mask=mask, linewidths=.75, linecolor='silver')
```

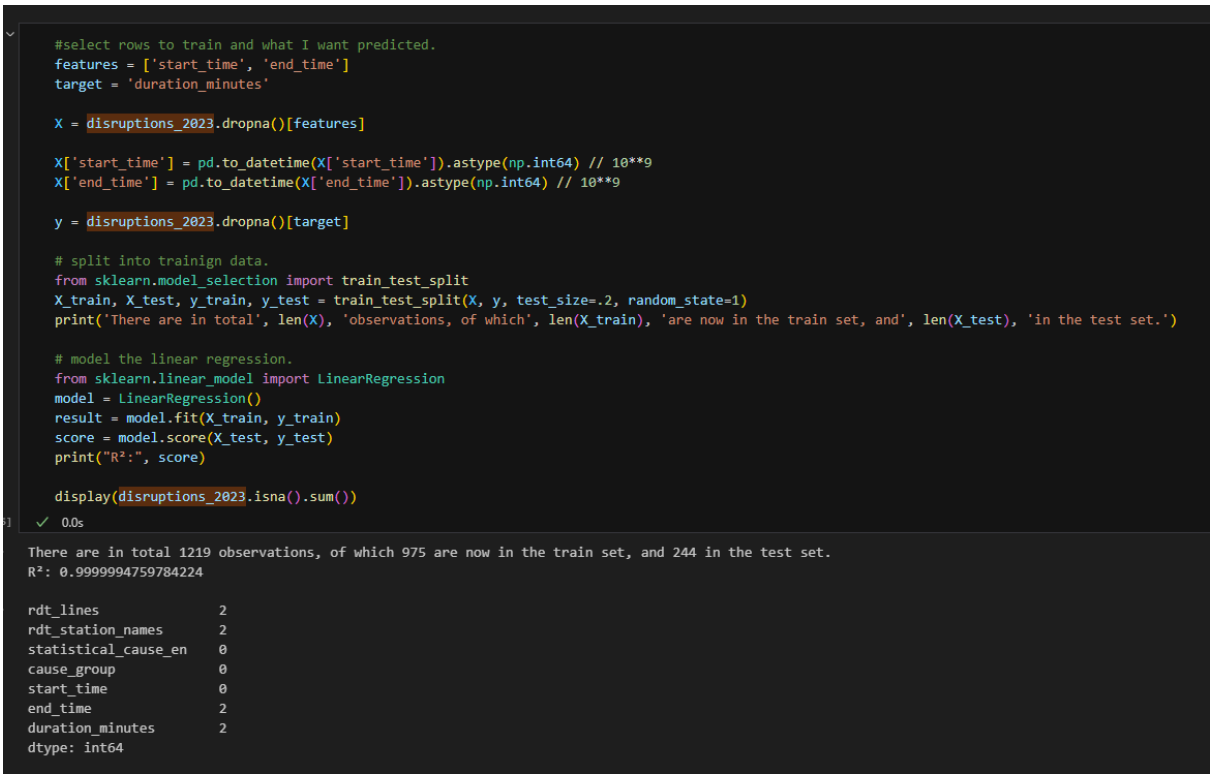
✓ 0.1s

	start_time	end_time	duration_minutes
0	2023-01-01 08:19:26	2023-01-01 22:43:08	864.0
1	2023-01-01 10:31:49	2023-01-01 10:56:17	24.0
2	2023-01-01 13:19:24	2023-01-02 00:02:39	643.0
3	2023-01-01 17:15:22	2023-01-01 20:14:23	179.0
4	2023-01-02 05:57:27	2023-01-03 02:07:13	1210.0
...	...	...	...
1218	2023-03-31 10:44:58	2023-03-31 10:55:53	11.0
1219	2023-03-31 12:21:49	2023-03-31 12:27:27	6.0
1220	2023-03-31 17:48:22	2023-03-31 18:17:16	29.0
1221	2023-03-31 19:17:20	2023-03-31 19:27:58	11.0
1222	2023-03-31 19:21:04	2023-03-31 19:48:51	28.0

1223 rows × 3 columns



Here I look at the numeric data to see if the start and end times are connected to the time duration. Luckily it does. It would be weird if it wasn't though.

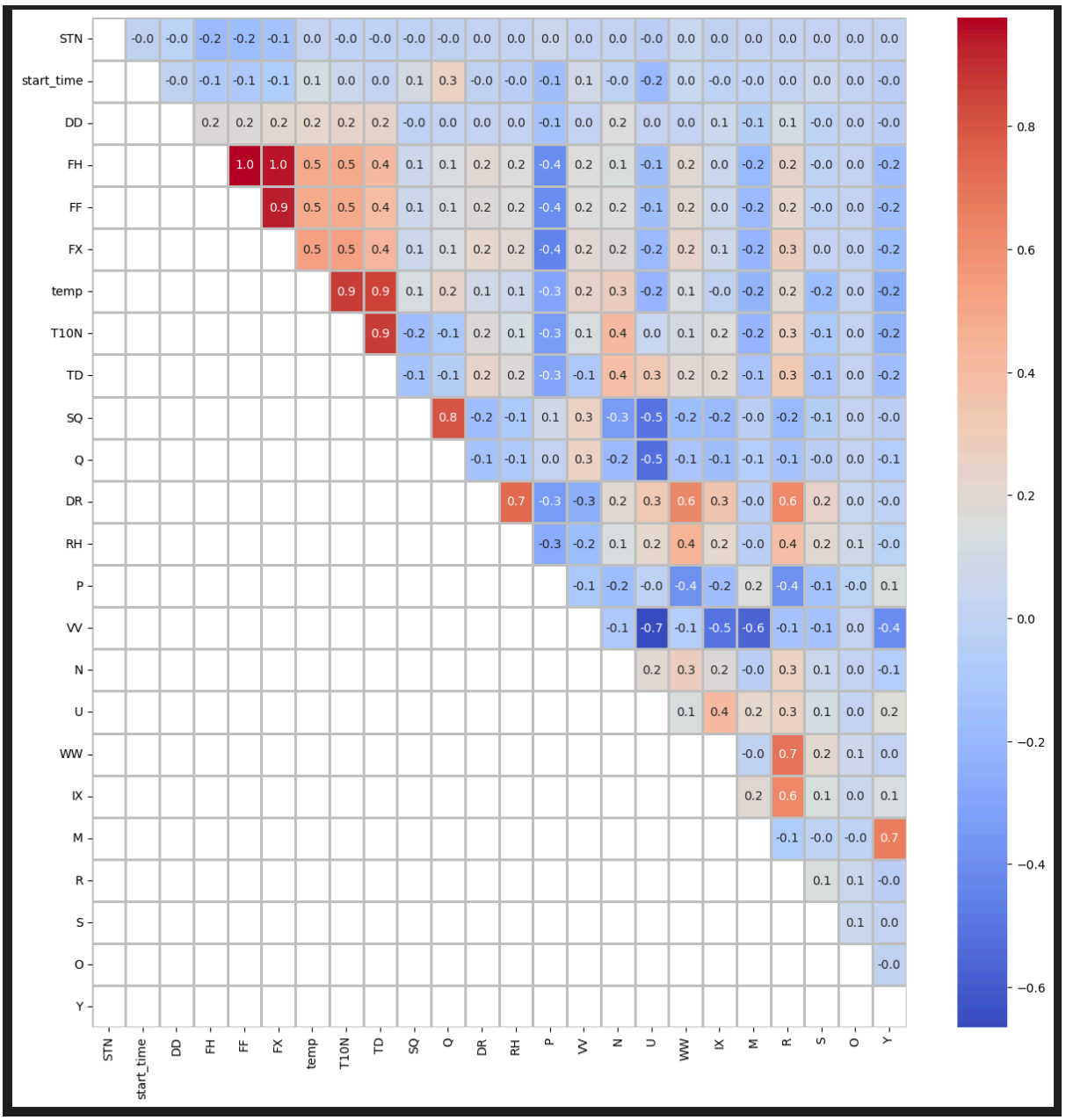


```
#select rows to train and what I want predicted.
features = ['start_time', 'end_time']
target = 'duration_minutes'

X = disruptions_2023.dropna()[features]

X['start_time'] = pd.to_datetime(X['start_time']).astype(np.int64) // 10**9
X['end_time'] = pd.to_datetime(X['end_time']).astype(np.int64) // 10**9

y = disruptions_2023.dropna()[target]

# split into trainign data.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2, random_state=1)
print('There are in total', len(X), 'observations, of which', len(X_train), 'are now in the train set, and', len(X_test), 'in the test set.')

# model the linear regression.
from sklearn.linear_model import LinearRegression
model = LinearRegression()
result = model.fit(X_train, y_train)
score = model.score(X_test, y_test)
print("R²:", score)

display(disruptions_2023.isna().sum())
```

✓ 0.0s

```
There are in total 1219 observations, of which 975 are now in the train set, and 244 in the test set.
R²: 0.9999994759784224

rdt_lines               2
rdt_station_names       2
statistical_cause_en    0
cause_group             0
start_time              0
end_time                2
duration_minutes        2
dtype: int64
```

But for the fun of it, I tried to see if I could predict the duration time with the given info. And it kind of works.



And so that i know i am doing something right i also looked at the weather data i received.

```python
#select rows to train and what I want predicted.
features = ['STN','start_time','FH','FF','FX', 'T10N', 'TD']
target = 'temp'

X = weather_2023.dropna()[features]

X['start_time'] = pd.to_datetime(X['start_time']).astype(np.int64) // 10**9

y = weather_2023.dropna()[target]

# split into trainign data.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2, random_state=1)
print('There are in total', len(X), 'observations, of which', len(X_train), 'are now in the train set, and', len(X_test), 'in the test set.')

# model the linear regression.
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
score = model.score(X_test, y_test)
print("Accuracy:", score)

display(weather_2023.isna().sum())
```

✓ 0.0s

```
There are in total 3619 observations, of which 2895 are now in the train set, and 724 in the test set.
Accuracy: 0.052486187845303865
```

But that requires some more work and finding the best way to figure out what regression works best for this.

LO3: Explainable AI

Intro:

You deliver AI projects that follow the three 'Explainable AI' principles of **transparency**, **interpretability**, and **explainability**.

Transparency means that the process by which the used input data results in prediction models is reproducible, reliably described and its decisions are motivated.

Interpretability addresses the possibility for humans to comprehend the project cohesion and results by making them comparable to the domain knowledge and baselines.

Explainability refers to the application of tools and methods that turn black-box models into grey/white-box models by having the model draw out its decision making process and/or describe its feature importance.

What I did for this LO:

LO4: Professional Standard

Intro:

You show that you conduct work in accordance with an industry supported methodological approach (AI Project Methodology) in terms of your project's **goals**, **stakeholder involvement**, **applied research**, **decision making** and **reporting**.

Goals refer to identified authentic immediate and long term issues that you work towards finding appropriate solutions for. Whilst defining your goals you explore the context and environment of your project and you make the necessary business, sustainable and ethical considerations.

Stakeholder involvement implies that you involve relevant and competent partners in your project from beginning to the end. During your project you communicate constructively with all your stakeholders.

Applied research implies that you effectively use research strategies and methods, like those in the DOT-framework, for your domain understanding and other research activities.

Decision making means that you correctly identify the need for further iterations, using evaluation models like the TIC-tool and your stakeholder feedback.

Reporting refers to well structured and well motivated, correct and relevant documents, using APA style referencing for used external sources, as well as using visualizations, concluding your project proportionally covering all four phases of the methodology.

What I did for this LO:

LO5: Personal Leadership

Intro:

You **are aware of your strengths and pitfalls** in ICT as well as your personal development. To nurture personal growth you **are able to engage in actions** that align with your core values, **in a way that suits you**.

Being aware of your strengths and pitfalls means that you are able to recognize (among other things through self-reflection and asking for feedback) what you are already good at and where growth is still possible.

Being able to engage in actions means that you take responsibility, growing towards a professional ICT practitioner, seeing and seizing opportunities in a structured, planned and efficient way.

In a way that suits you means that in the activities you undertake you apply an approach that fits your style of acquiring knowledge and skills.

What I did for this LO: