Weather Effects on Domestic Trips in Finland

Group 3

Muhammad Ali
Elnur Garibov
Ekaterina Ustiukhina
Mariia Vlasova
Rafaela von Frenckell

The project work was divided into two parts: 1) writing the code and 2) preparing the report and the presentation
In group 1 was Ekaterina, Elnur and Muhammad. Ekaterina wrote the code while the rest looked it over once the code was finished.
Mariia and Rafaela shared the work in group 2.

Introduction

This project aims to find if the weather has any affect on domestic trips to several destinations in Finland: Uusimaa, Southwest Finland, North Ostrobothnia and Lapland. The destinations were chosen due to data availablity and relatively high number of domestic trips compared to other destinations. The project assesses 2012-2024 period of time. The median yearly temperature is used for final analysis. Results showcase a direct correlation between increase in temperature and increase of trips. The project also looks at most poular destinations by season and trends in domestic travels over the years 2012-2024.

In [63]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
```

Load the data

Dataset: Average Temperature Data-Finnish Meteorological Institute

Processing and merging monthly average temperature data from five Finnish cities Helsinki, Oulu, Rovaniemi, Tampere, and Turku, obtained from the **Finnish Meteorological Institute (FMI)**. It loads individual CSV files, removes unnecessary columns (day and time), renames the temperature column for clarity, and merges all datasets based on year and month using an outer join. The final DataFrame (*df_weather*) provides a unified view of temperature trends across key cities.

Finnish Meteorological Institute (no date). Temperature and precipitation statistics from 1961 onwards. Available at https://en.ilmatieteenlaitos.fi/statistics-from-1961-onwards (Assessed 30 March 2025).

In [64]:
```
helsinki_temp = pd.read_csv("Helsinki_temp.csv")
oulu_temp = pd.read_csv("Oulu_temp.csv")
rovaniemi_temp = pd.read_csv("Rovaniemi_temp.csv")
tampere_temp = pd.read_csv("Tampere_temp.csv")
turku_temp = pd.read_csv("Turku_temp.csv")

columns_to_keep = ["Year", "Month", "Monthly mean temperature [°C]"]
helsinki_temp = helsinki_temp[columns_to_keep].rename(columns={"Monthly mean tem
oulu_temp = oulu_temp[columns_to_keep].rename(columns={"Monthly mean temperature
rovaniemi_temp = rovaniemi_temp[columns_to_keep].rename(columns={"Monthly mean t
tampere_temp = tampere_temp[columns_to_keep].rename(columns={"Monthly mean tempe
turku_temp = turku_temp[columns_to_keep].rename(columns={"Monthly mean temperatu

df_weather = helsinki_temp.merge(oulu_temp, on=["Year", "Month"], how="outer") \
                          .merge(rovaniemi_temp, on=["Year", "Month"], how="oute
                          .merge(tampere_temp, on=["Year", "Month"], how="outer"
                          .merge(turku_temp, on=["Year", "Month"], how="outer")

print(df_weather.head())
```

```
   Year  Month  Helsinki   Oulu  Rovaniemi  Tampere  Turku
0  2012      1      -3.4   -9.1      -10.7     -6.7   -4.3
1  2012      2      -6.8  -11.7      -13.3     -8.7   -6.0
2  2012      3       0.8   -2.7       -3.6     -0.4    1.1
3  2012      4       4.1    0.4       -0.4      2.9    4.5
4  2012      5      10.9    7.8        7.3     10.5   11.0
```

Checking the merged dataset

In [65]:
```
df_weather.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Year       156 non-null    int64  
 1   Month      156 non-null    int64  
 2   Helsinki   156 non-null    float64
 3   Oulu       156 non-null    object 
 4   Rovaniemi  156 non-null    float64
 5   Tampere    156 non-null    float64
 6   Turku      156 non-null    float64
dtypes: float64(4), int64(2), object(1)
memory usage: 8.7+ KB
```

The "Oulu, [°C]" column is of type object, change the dtype to float.

In [66]:
```
df_weather["Oulu"] = pd.to_numeric(df_weather["Oulu"], errors="coerce")
print(df_weather.dtypes)
```

```
Year           int64
Month          int64
Helsinki     float64
Oulu         float64
Rovaniemi    float64
Tampere      float64
Turku        float64
dtype: object
```

Change the dataset to have columns Year, Month, City and Temperature (move cities from columns to rows)

In [67]:
```
df_weather = df_weather.melt(id_vars=["Year", "Month"],
                             var_name="City",
                             value_name="Temperature")

print(df_weather.head())
```

```
   Year  Month      City  Temperature
0  2012      1  Helsinki         -3.4
1  2012      2  Helsinki         -6.8
2  2012      3  Helsinki          0.8
3  2012      4  Helsinki          4.1
4  2012      5  Helsinki         10.9
```

In [68]:
```
df_weather.tail(5)
```

Out[68]:

	Year	Month	City	Temperature
775	2024	8	Turku	18.0
776	2024	9	Turku	14.2
777	2024	10	Turku	8.6
778	2024	11	Turku	3.9
779	2024	12	Turku	0.9

Final weather dataset contains mean temperatures of each city, for each month between the years 2012-2024. The columns are Year, Month, City and Temperature.

Dataset: Domestic Leisure Trips Statistics-Statistics Finland

Loading and pre-processing domestic travel data from **Statistics Finland**. This part of code reads an Excel file containing statistics on domestic overnight leisure trips in Finland. It loads the specified sheet while skipping metadata rows, assigns proper column names, removes redundant header rows, and renames the first column to "Region" for clarity.

Statistics Finland (no date). Finnish travel. Available at https://stat.fi/en/statistics/smat#cubes (Assessed: 30 March 2025)

In [69]:
```
xls = pd.ExcelFile("domestic_trips.xlsx")

df_trips = pd.read_excel(xls, sheet_name="001_13fi_2024", skiprows=2)
```

```
df_trips.columns = df_trips.iloc[0]
df_trips_fixed = df_trips[2:].reset_index(drop=True)
df_trips_fixed.rename(columns={df_trips_fixed.columns[0]: "Region"}, inplace=Tru

df_trips_fixed.head()
```

Out[69]:

	Region	Spring (January-April)	Summer (May-August)	Autumn (September-December)	Spring (January-April)	Summer (May-August)	Autumn (September-December)	Sp (Jan A
0	WHOLE COUNTRY	7190	11750	7830	7370	11500	7660	
1	MK01 Uusimaa	1310	1960	1320	1230	1700	1230	
2	MK02 Southwest Finland	500	1120	660	530	980	700	
3	MK04 Satakunta	190	440	280	240	380	230	
4	MK05 Kanta-Häme	170	390	260	160	300	240	

5 rows × 40 columns

Assigning correct column names to the dataset. This code manually constructs column names by pairing each year (2012–2024) with its corresponding season (Spring, Summer, Autumn).

In [70]:

```
years = list(range(2012, 2025))
seasons = ["Spring (January-April)", "Summer (May-August)", "Autumn (September-D
new_column_names = ["Region"] + [f"{year} {season}" for year in years for season
df_trips_fixed.columns = new_column_names
print(df_trips_fixed.head())
```

	Region	2012 Spring (January-April)	2012 Summer (May-August)	2012 Autumn (September-December)	2013 Spring (January-April)	2013 Summer (May-August)	2013 Autumn (September-December)	2014 Spring (January-April)	2014 Summer (May-August)	2014 Autumn (September-December)	...	2021 Autumn (September-December)	2022 Spring (January-April)	2022 Summer (May-August)	2022 Autumn (September-December)	2023 Spring (January-April)	2023 Summer (May-August)	2023 Autumn (September-December)	2024 Spring (January-April)
0	WHOLE COUNTRY	7190	11750	7830	7370	11500	7660	6410	11180	7620	...	8410	6030	11240	7290	5750	9660	5920	6250
1	MK01 Uusimaa	1310	1960	1320	1230	1700	1230	1050	1760	1340	...	1510	930	1750	1350	1010	1440	880	1080
2	MK02 Southwest Finland	500	1120	660	530	980	700	430	1040	630	...	840	490	1180	700	340	1060	650	520
3	MK04 Satakunta	190	440	280	240	380	230	210	430	300	...	220	180	390	200	120	380		
4	MK05 Kanta-Häme	170	390	260	160	300	240	180	400	190	...	290	240	430	230	230	390		

```
3                                160                                250
4                                150                                190

  2024 Summer (May-August) 2024 Autumn (September-December)
0                    10780                               6970
1                     1630                               1220
2                     1270                                590
3                      420                                210
4                      390                                290

[5 rows x 40 columns]
```

Filtering the dataset to include only selected regions.

In [71]:
```
regions_to_keep = ["MK01 Uusimaa", "MK02 Southwest Finland", "MK13 Central Finla
df_trips_filtered = df_trips_fixed[df_trips_fixed["Region"].isin(regions_to_keep
df_trips_filtered.head()
```

Out[71]:

	Region	2012 Spring (January-April)	2012 Summer (May-August)	2012 Autumn (September-December)	2013 Spring (January-April)	2013 Summer (May-August)	2013 Autumn (September-December)	(
1	MK01 Uusimaa	1310	1960	1320	1230	1700	1230	
2	MK02 Southwest Finland	500	1120	660	530	980	700	
12	MK13 Central Finland	510	750	510	540	790	500	
16	MK17 North Ostrobothnia	750	780	680	690	830	650	
18	MK19 Lapland	590	670	540	690	700	530	

5 rows × 40 columns

Reshaping the dataset into long format for analysis. This code converts the wide-format DataFrame *(df_trips_filtered)*, where each year-season combination is a separate column, into a long format using pd.melt().

In [ ]:
```
df_domestic_trips = df_trips_filtered.melt(id_vars=["Region"], var_name="Year_Se
df_domestic_trips[['Year', 'Season']] = df_domestic_trips['Year_Season'].str.ext
df_domestic_trips.drop(columns=["Year_Season"], inplace=True)
df_domestic_trips["Year"] = df_domestic_trips["Year"].astype(int)

print(df_domestic_trips.head())
```

```
                    Region Trips (thousand)  Year                 Season
0             MK01 Uusimaa             1310  2012  Spring (January-April)
1   MK02 Southwest Finland              500  2012  Spring (January-April)
2     MK13 Central Finland              510  2012  Spring (January-April)
3  MK17 North Ostrobothnia              750  2012  Spring (January-April)
4             MK19 Lapland              590  2012  Spring (January-April)
```

In [73]:
```
df_domestic_trips.tail()
```

Out[73]:

	Region	Trips (thousand)	Year	Season
190	MK01 Uusimaa	1220	2024	Autumn (September-December)
191	MK02 Southwest Finland	590	2024	Autumn (September-December)
192	MK13 Central Finland	370	2024	Autumn (September-December)
193	MK17 North Ostrobothnia	630	2024	Autumn (September-December)
194	MK19 Lapland	460	2024	Autumn (September-December)

The final domestic trips-dataset contains the number of trips (in thousands) to each area of interest, for each year between 2012-2024 for each season. The columns are Region, Trips, Year and Season.

Merging Datasets

Now that we have the domestic trips dataset **(df_domestic_trips)** in long format, we need to merge it with the weather dataset **(df_weather)**, which contains monthly mean temperatures for key cities.

Since **df_domestic_trips** is grouped by Region, Year, and Season, and **df_weather** contains Year, Month, and Temperature for each city. To get same format we need to map months to seasons (Spring, Summer, Autumn) in **df_weather** and calculate seasonal average temperatures for each region.

In [ ]:
```
region_mapping = {
    "Helsinki": "MK01 Uusimaa",
    "Turku": "MK02 Southwest Finland",
    "Tampere": "MK13 Central Finland",
    "Oulu": "MK17 North Ostrobothnia",
    "Rovaniemi": "MK19 Lapland"
}

df_weather["Region"] = df_weather["City"].map(region_mapping)

season_mapping = {
    1: "Spring (January-April)", 2: "Spring (January-April)", 3: "Spring (Januar
    5: "Summer (May-August)", 6: "Summer (May-August)", 7: "Summer (May-August)"
    9: "Autumn (September-December)", 10: "Autumn (September-December)", 11: "Au
}

df_weather["Season"] = df_weather["Month"].map(season_mapping)
df_weather_avg = df_weather.groupby(["Year", "Season", "Region"], as_index=False
df_weather = df_weather.drop(columns=["Month"])
```

```
# Merge the two dataframes on Year, Season, and Region
df_merged = pd.merge(df_domestic_trips, df_weather_avg, on=["Year", "Season", "R

# Rename regions
df_merged["Region"] = df_merged["Region"].str.replace(r"MK\d+\s", "", regex=True

df_merged.head()
```

Out[ ]:

	Region	Trips (thousand)	Year	Season	Temperature
0	Uusimaa	1310	2012	Spring (January-April)	-1.325
1	Southwest Finland	500	2012	Spring (January-April)	-1.175
2	Central Finland	510	2012	Spring (January-April)	-3.225
3	North Ostrobothnia	750	2012	Spring (January-April)	-5.775
4	Lapland	590	2012	Spring (January-April)	-7.000

Final merged dataset *df_merged* contains columns Region, Trips (thousand), Year, Season and Temperature (average). It tells the average temperature for and total trips to each region for each season within the set timeframe.

Analysis

In the following part the dataset will be analysed and we will investigare if there is any correlation between weather and trip frequency (i.e. does warmer weather encourage more people to travel to the region?).

Correlation Between Temperature and Trip Frequency

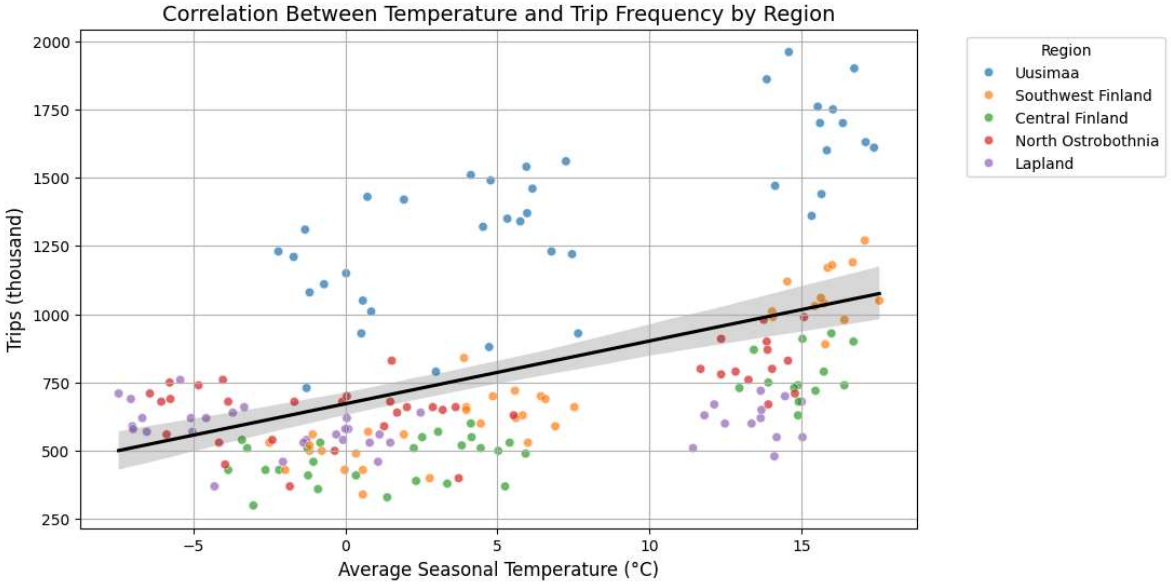
We display the correlation between temperature and trip frequency by a line & scatter plot. To do this we utilised matplot as it's a very simple yet effectful tool in Python. We chose a line plot as it's very simple and can be read without needing too much extra information. The scatter in the backgroup help give extra background information, and to help the viewer think critically about the plot. The scatter is colourcoded by destination.

In [ ]:

```
df_cleaned = df_merged.dropna(subset=["Temperature", "Trips (thousand)", "Region

df_cleaned["Trips (thousand)"] = pd.to_numeric(df_cleaned["Trips (thousand)"], e
df_cleaned["Temperature"] = pd.to_numeric(df_cleaned["Temperature"], errors='coe

correlation = df_cleaned["Temperature"].corr(df_cleaned["Trips (thousand)"])
print(f"Correlation between Temperature and Trip Frequency: {correlation:.3f}")

plt.figure(figsize=(10, 6))
scatter = sns.scatterplot(
    x="Temperature",
    y="Trips (thousand)",
    data=df_cleaned,
```

```
    hue="Region",
    palette="tab10",
    alpha=0.7
)

sns.regplot(
    x="Temperature",
    y="Trips (thousand)",
    data=df_cleaned,
    scatter=False,
    line_kws={"color": "black"}
)

plt.title("Correlation Between Temperature and Trip Frequency by Region", fontsi
plt.xlabel("Average Seasonal Temperature (°C)", fontsize=12)
plt.ylabel("Trips (thousand)", fontsize=12)
plt.legend(title="Region", bbox_to_anchor=(1.05, 1), loc="upper left")  # Adjust
plt.grid(True)
plt.show()
```

```
Correlation between Temperature and Trip Frequency: 0.474
```



The resulting graph showcases a direct correlation between these two variables. Each datapoint is related to the domestic trip destination and the corresponding temperature at the time. There is a notable variation in correlation data points. However, the overall trend is positive with an increase in temperature directly impacting the number of trips. With other kinds of data it would be a more complex task to assign the most impactful factor, but with temperature being an independent actor, we can conclude that it is the temperature that designates the increase in trips.

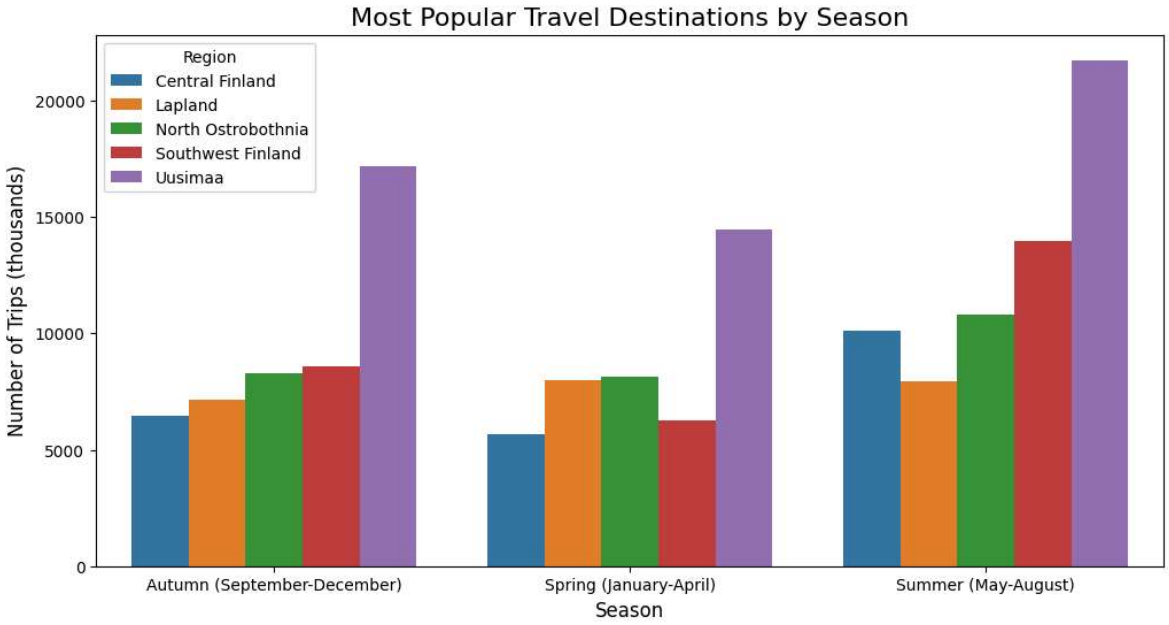
This chart also shows the scatter of each destination individually. From this is can be seen that temperature does not affect the travel to all region equally. For Lapland there's no clear change in trips taken when the temperature rises. For North Ostrobothnia the change is significantly smaller than for example Southwest Finland.

However, it is very important to realise here that the data has not been normalised or checked for any other influensing factors (such as summer holidays occuring during warm weather and influencing people taking trips).

Most Preferred Vacation Destinations Per Season

The most preffered destinations per seasons was also displayed with help of the useful matplot-tool. It is displayed as a bar graph, as it's easy to cathegorise and compare the destinations and seasons.

In [ ]:
```
df_seasonal_trips = df_merged.groupby(["Region", "Season"])["Trips (thousand)"].

plt.figure(figsize=(12, 6))
sns.barplot(data=df_seasonal_trips, x="Season", y="Trips (thousand)", hue="Regic
plt.xlabel("Season", fontsize=12)
plt.ylabel("Number of Trips (thousands)", fontsize=12)
plt.title("Most Popular Travel Destinations by Season", fontsize=16)
plt.legend(title="Region")
plt.show()
```



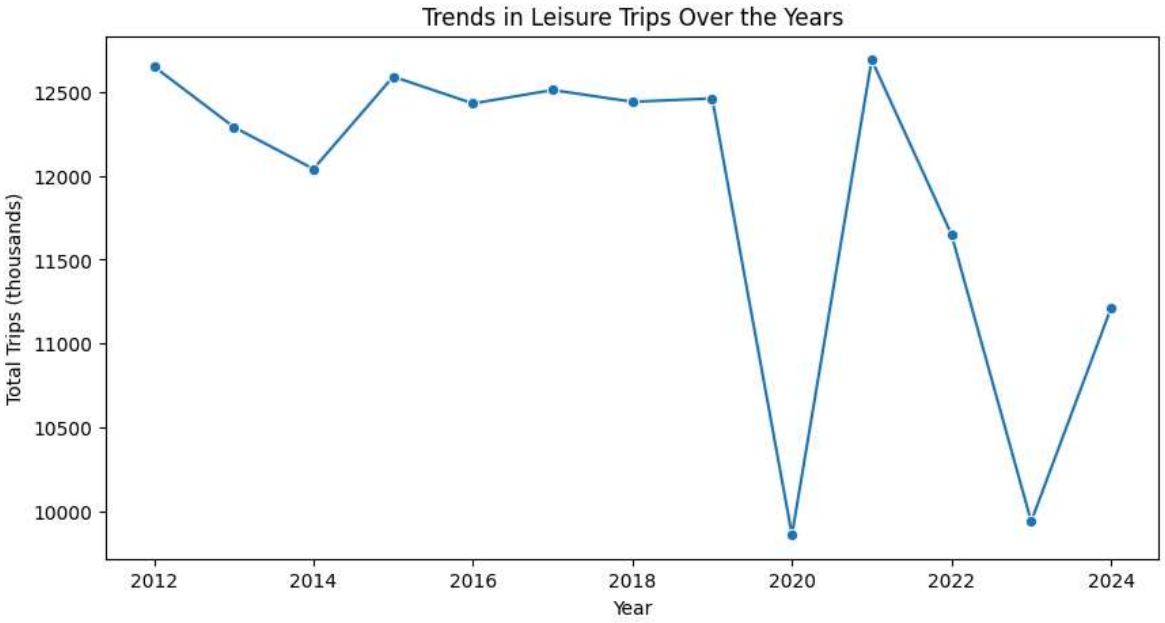
This barchart shows that Uusimaa is the most popular destination out of the four investigated. Something to keep in mind is that the area, population, and popular sights in the regions have not been taken into account in this project. It can also be seen that summer is the most popular season for travelling, and spring is the least popular. Some reasons for this could be summer and Christmas holidays. Southwest Finland is the second most popular destination in summer and autumn, but least popular in the spring. Lapland, on the other hand, is the least popular destination during the summer out of the investigated regions. North Ostrobothnia and Lapland have similar tourism-numbers in autumn and spring, while Central Finland is the least popular destination in all seasons expect summer, where it's second to last, just after North Ostrobothnia.

Even with no temperature data, is possible to imply that the number of trips is directly impacted by the temperature, with the summer season having the highest number of trips.

Yearly Trends in Leisure Trips

Yearly trips over the timeframe was displayed as a line graph, again as it's easy to read and effectively shows the trends. It was also plotted with Matplot.

In [77]:
```
df_yearly_trips = df_merged.groupby("Year")["Trips (thousand)"].sum().reset_inde

plt.figure(figsize=(10, 5))
sns.lineplot(data=df_yearly_trips, x="Year", y="Trips (thousand)", marker="o")
plt.xlabel("Year")
plt.ylabel("Total Trips (thousands)")
plt.title("Trends in Leisure Trips Over the Years")
plt.show()
```



From this plot, three dips can be seen in domestic trips 2012-2024. The biggest one, in 2020, is easily explainable by Covid-19. There was also a dip in domestic trips in 2023, and a smaller dip in 2014. Other than that, the trips have been at a consistant level until 2021. One possible reason for the decrease in domestic trips since 2021 could be the current economic situation.

Conclusion

Findings

The report has found that weather impacts the amount of domestic trips taken in Finland. It was also discovered that it affects different regions differently, and on a varying scale. Uusimaa was the most popular destination regardless of season or weather, while Central Finland was the least popular during the colder months and Lapland was the least popular during the summer, out of the included regions. Finally, it was concluded that there are many other factors affecting travel destinations other than weather.

As stated earlier in the report, travel is influenced by many things, but warm weather seems to encourage people to travel more around the country. This could be due to

holidays or people wanting to get out more during the warmer and lighter summer months. It's also good to note here that the data was generalised to season averages, and thus it's hard to make conclusions stating that people travel more on warm, sunny days than on rainy days, if all travel happened in the same season.

What we learned

During this project, we learned how to work and code together on the same project. We also learned how to make graphs with layers of information. During the projects, some mistakes were made (especially in the coding part), but we learned from those and ended up with better code. We also learned that there are big differences in how popular different regions are as travel destinations, and how the domestic travel changes throughout the seasons.

Sources

Statistics Finland (no date). Finnish travel. Available at https://stat.fi/en/statistics/smat#cubes (Assessed: 30 March 2025)

Finnish Meteorological Institute (no date). Temperature and precipitation statistics from 1961 onwards. Available at https://en.ilmatieteenlaitos.fi/statistics-from-1961-onwards (Assessed 30 March 2025).