# Statistical Machine Learning – Week 5

Simon Walser

HS 2022/23

## 1 Task

In this lab, we will train and evaluate a classifier based on a real dataset of pedestrian frequency on Vadianstrasse in St. Gallen which is published by Open Data St. Gallen. This dataset and many more can be found at this link. The municipality of St.Gallen has a sensor on Vadianstrasse (near St.Galler Stadtwerke) that measures the number of pedestrians passing by. This dataset shows the number of people coming from the right (Neumarkt) and left (Multergasse) every ten minutes.

The aim of this lab session is to train an logistic regression model which assigns each sample of the dataset to one of the two classes, `day` or `night`, based on the number of pedestrians passing. In general, it can be assumed that the number of pedestrians is lower at night than during the day. Such a classifier might be useful for two purposes:

**Inference** Understanding the relationship between the predictors (passing pedestrians from two directions) and the response (day or night)

**Prediction** Putting counting unit or parts of it into energy saving mode at night (i.e. when almost no pedestrians are passing).

In a first step, we will prepare the data by utilizing pandas, a powerful data analysis and manipulation tool. Next, we will create a training and testing split and fit a sklearn LogisticRegression object using the training dataset. The resulting logistic regression model will then be evaluated on the testing dataset. For this purpose the script `week5_exercise.py` is provided. Try to understand the skeleton code and complete the following TODO's.

- The function `read_data` loads the dataset from `fussganger_vadianstrasse.csv` into a pandas dataframe. The function `preprocess_data` prepares the dataframe by deleting irrelevant columns, sorting entries and determining the target class. Try to understand the purpose of the pandas functions.
- The function `plot_data` visualizes the dataset as time series and as histogram plot, differentiated by classes. Try to get an understanding of the data properties with the help of these plots.
- In the `main` function create a training (90%) and testing (10%) split from the preprocessed dataset using sklearn's train_test_split function. Make sure that each split contains a balanced number of samples from each class (stratified split).
- Instantiate a sklearn LogisticRegression object in the function `fit_logistic_regression`, fit it to the training dataset and return the resulting model. The model should have two predictors (number of pedestrians in two directions) and a categorical response (day / night). Try to understand the parameters of the logistic regression model and set them to the best of your knowledge. Invoke the `fit_logistic_regression` function from the `main` function.

- Evaluate the logistic regression model in the `evaluate_logistic_regression` function based on the testing dataset and print the resulting mean accuracy. Invoke the `evaluate_logistic_regression` function from the `main` function. Hint: sklearn's LogisticRegression objects have a member method which calculates the mean accuracy.
- If you call the function `plot_logistic_regression` with the trained model and the testing dataset as parameters, you can see an illustration of the logistic regression model. Try to understand the meaning of the scatter plot and the surface plot.
- Try to improve the model performance by modifying the hyperparameters of the model. Test the following modifications to the model / data:
  - Add a penalty to the logistic regression loss (e.g. `l1` or `l2`)
  - Fit the model with only one predictor.
  - Create a sklearn pipeline by first applying a StandardScaler and then classifying the standardized features with a logistic regression model. See here for an explanation of pipelines.
  - Take a look at the attributes `coef_` and `intercept_` of the `LogisticRegression` instance. What is their meaning and what can you deduce from them?

## Comments

- If you want to learn more about how to train a logistic regression model or how the `l1` / `l2` regularization works, read this post.