This compulsory assignment consists of 4 exercises. Exercises must be solved in groups (assigned in Canvas) and uploaded to Canvas by the submission deadline. Please provide a single `.html` or `.pdf`-file, entitled `dat320_comp3_groupX.html` (or `.pdf`), where `X` should be replaced by your group number (1-16). The file should be structured into sections (one section per exercise) and subsections (one subsection per task). The usage of R markdown (R package `knitr`) is strongly recommended - you find a sample file in the folder "R markdown example". Further, check out this short introduction video: `https://rmarkdown.rstudio.com/lesson-1.html`.

Exercise 1 (Hidden Markov models)

romanian_electricity.csv is the complete dataset used in Task 2 / Compulsory Assignment 1. It contains information about electricity consumption and production in Romania.

Material provided:

- `romanian_electricity.csv`: dataset

Tasks:

(a) *Load the dataset and perform an exploratory data analysis. In particular, investigate the properties of the time series (stationarity, autocorrelation, partial autocorrelation). Next, add a column "Import_noNC", which is given by Consumption-Production+Nuclear. This is the import of energy of all types, except for nuclear production. Hint: Note that there are duplicated and missing time points due to daylight saving time (changes between summer and winter time). Do not impute or delete these time points!*

(b) *Investigate the columns "Nuclear" (nuclear power production) and "Import_noNC", separately. Assume that you fit a Hidden Markov Model with 2 states to each time series. Answer the following questions:*

- *What would the 2 states represent?*
- *Which types of stochastic processes are used to model the observable and the latent sequence, respectively?*
- *Are the model assumptions fulfilled?*

(c) *Fit one HMM for each of the columns "Nuclear" and "Import_noNC" (two distinct models). Print the model parameters of each model and estimate the latent state sequence using the Viterbi algorithm. Next, plot both time series colored by their predicted latent states. What do you see?*

(d) *The states predicted for "Import_noNC" are probably more noisy. Why is that? Try to smooth the time series and re-fit the same model. Plot the same graph as in (c) and compare.*

Exercise 2 (Hidden Markov models)

The dataset *tipburn.csv* contains data for two types of a plant disease named "tip-burn" ("inner tip-burn" and "outer tip-burn"). Both types of the diseaes are assessed on a weekly basis by human observers, who count the number of affected plants in a greenhouse. The columns in the dataset represent the percentage of assessed plants that are affected by each type of tip-burn in each calendar week of the year.

Material provided:

- `tipburn.csv`: dataset

Tasks:

(a) *Load in the dataset and perform an exploratory data analysis. Convert the calendar_week column to appropriate date object. Next, create two columns named "group_inner" and "group_outer" that assign a degree of tip burn based off the following conditions:*

$$group = \begin{cases} "None" & \text{if tip burn} = 0 \\ "Weak" & \text{if } 0 < \text{tip burn} \leq 0.25 \\ "Strong" & \text{if tip burn} > 0.25 \end{cases}$$

(b) *Experts assume that tip burn is caused by some unobserved climate conditions. Explain why it makes sense to apply a discrete Hidden Markov Model with two latent states, which can be interpreted as "risk" and "non-risk" climate conditions. The three levels of the observable variable are: "none", "weak" and "strong" tip burn. You may treat inner and outer tip-burn as two separate models first. Which parameters would you suggest to use for initialization (π_0, A and B)? Which dimensions do these parameters have for each of the tip-burn types?*

(c) *Fit three distinct HMM models:*

- *a discrete HMM with 2 latent states for group_inner,*
- *a discrete HMM with 2 latent states for group_outer,*
- *a discrete HMM with 2 latent states both, for group_inner and group_outer simultaneously.*

where the latent states represents the "risk" and "non-risk" climate conditions. For each model, apply the Viterbi algorithm to estimate the latent state sequences.
Hint: use the package "depmixS4" package with family = "multinomial".

(d) *Add the predicted state sequences to the original dataframe so you get the three new columns states_inner, states_outer and states_joint. Next, plot the groups along the time axis, colored by the predicted states for each model. Comment on what you see. Compare the estimated state sequences against each other (e.g. by computing a pairwise accuracy between the 3 fitted sequences).*

Exercise 3 (Feature-based classification)

pedestrian.csv contains data for the pedestrian count in Chinatown-Swanston St (North) for 12 months of the year 2017. Classes are based on whether data are recorded on a workday (Mon-Fri) or weekend (Sat-Sun). Class 1 represents weekend and class 2 represents workday.

Material provided:

- `pedestrian.csv`: dataset

Tasks:

(a) *Load the dataset and perform an exploratory data analysis. Plot the averaged pedestrian counts for both groups along the time axis, shaded by ± 1 standard deviation. Which differences can you see between the groups?*

(b) *Split the samples into a 70%-30% train-test split. Use the tsfeatures function from the tsfeatures package to extract features from the time series (check out ?tsfeatures or the vignette to get information about the input format and which features can be extracted). By default, tsfeatures extracts features based on frequency, entropy, or the acf. You can use the default setting but may as well try more feature extractors offered by the package.*

(c) *Train predictive models for binary classification: a) a logistic regression model and b) a random forest. Use the features extracted in (b). Tune the models using machine learning approaches (e.g., data standardization, PCA, cross-validation, grid search, regularization,...). Predict the test data. Compute the accuracy and F1 scores for both classes. Compare and interpret the results.*

(d) *As a baseline for the classification model based on feature selection, extract daily mean, standard deviation etc. manually for each day and use them as features for the models in (c), instead. Compare the performance of the models. Does the model in (c) beat the baseline?*

Exercise 4 (Distance-based clustering)

Food spectrographs are used in chemometrics to classify food types, a task that has obvious applications in food safety and quality assurance. The coffee data set is a two class problem to distinguish between Robusta and Aribica coffee beans. Further information can be found in the original paper Briandet et al. Discrimination of Arabica and Robusta in Instant Coffee by Fourier Transform Infrared Spectroscopy and Chemometrics J. Agricultural and Food Chemistry, 44 (1), 1996. The first column is an index, the second encoded target variable, and the rest are the spectrographs.
Note: for this exercise we refer to the wavelengths as time.

Material provided:

- `coffee_train.csv`: dataset
- `coffee_test.csv`: dataset

Tasks:

(a) *Import the data and perform an exploratory data analysis. Plot the class distributions of the target column. Further, plot the class averages across the time axis, shaded by* ± 1 *standard deviation.*

(b) *Compute pair-wise distances between the given time series and perform hierarchical clustering. Try with the following distance metrics: Euclidean, DTW and ACF. Plot the dendrogram and divide the data into* $k = 3$ *clusters.*

(c) *The training data contains information about both the spectrographs for each sample and the class label. In some cases you would expect that the time series belonging to each group would be clustered together due to similarities in the spectral pattern. Look at the ground truth target distribution in each cluster and select the best clustering method from (b) based on the "purity" of each cluster; i.e., the method that yield clusters containing the least mixed target classes. Which distance metric matches best with the true groups?*

(d) *Use the selected model in (c) to create a prototype model of the training data. By prototype, it means computing the average of the samples at all times points within each cluster to get the centroids. Next, assign the clusters and target labels to the test data as follows:*

i For each sample in the test set $i \in 1, .., n$*, compute the distance between the samples and centroids* $c \in 1, 2, 3$ *at every time point* $t \in 1, .., T$*. Denote the distance measure between each prototype* c *and samples* i *at time* t *as* $d_{i,c} \in \mathbb{R}$ *(use the same distance measure as in (c)).*

ii Assign a new sample i *to cluster* $c^\star$ *given by*

$$c^\star = \arg\min_c \, d_{i,c}.$$

That is, the sample gets assigned to the cluster with the closest centroid.

iii Based on your observations in (c), assign the clusters to the appropriate target label.

Compute the confusion matrix of the predictions.