**PHASE 1 : PROBLEM DEFINITION AND DESIGN THINKING**

**INTRODUCTION:**

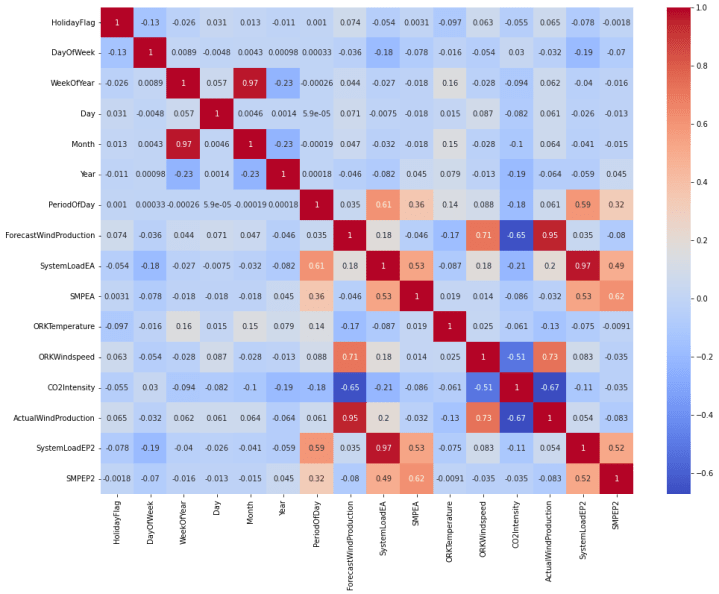
Suppose that your business relies on computing services where the power consumed by your machines varies throughout the day. You do not know the actual cost of the electricity consumed by the machines throughout the day, but the organization has provided you with historical data of the price of the electricity consumed by the machines. Below is the information of the [data](https://raw.githubusercontent.com/amankharwal/Website-data/master/electricity.csv) we have for the task of forecasting electricity prices:

**PROBLEM DEFINITION:**

* DateTime: Date and time of the record
* Holiday: contains the name of the holiday if the day is a national holiday
* HolidayFlag: contains 1 if it’s a bank holiday otherwise 0
* DayOfWeek: contains values between 0-6 where 0 is Monday
* WeekOfYear: week of the year
* Day: Day of the date
* Month: Month of the date
* Year: Year of the date
* PeriodOfDay: half-hour period of the day
* ForcastWindProduction: forecasted wind production
* SystemLoadEA forecasted national load
* SMPEA: forecasted price
* ORKTemperature: actual temperature measured
* ORKWindspeed: actual windspeed measured
* CO2Intensity: actual C02 intensity for the electricity produced
* ActualWindProduction: actual wind energy production
* SystemLoadEP2: actual national system load
* SMPEP2: the actual price of the electricity consumed (labels or values to be predicted)
* So your task here is to use this data to train a machine learning model to predict the price of electricity consumed by the machines. In the section below, I will take you through the task of electricity price prediction with machine learning using Python.

**Electricity Price Prediction Model:**

Now let’s move to the task of training an electricity price prediction model. Here I will first add all the important features to x and the target column to y, and then I will split the data into training and test sets:



As this is the problem of regression, so here I will choose the Random Forest regression algorithm to train the electricity price prediction model:

* from sklearn.ensemble import RandomForestRegressor
* model = RandomForestRegressor()
* model.fit(xtrain, ytrain)

**Discussion:**

**Predictive performance and optimal hedging:**

Overall the TCN model is found to give the lowest predicting errors for both SE1 and SE3 as seen in Figure 6 and 9. For SE1 the forecasting errors are consistently lower for the TCN model during all time periods, although the same clear trend is not observed for SE3 forecasting. For SE3 pricing area forecasting, the forecasting results do not favor the TCN model over the LSTM model. As shown with the correlation analysis SE3 prices are generally more dependent on outside factors such as commodity prices and hydro reservoirs. Long-term future contracts are also much stronger correlated with short-term future contracts in SE3 compared to SE1.

**Data selection:**

The input data used in this project is very complex and consists of many different parameters. The complex input data is used to represent the complex nature of electricity prices. However, the length of the time series is also short as the model performance is evaluated already after two years. Longer time series would have been preferable when training the models, especially given the very complex nature of the input data.

**Ethics and sustainability:**

Understanding electricity markets and forecasting markets can improve resource allocation and planning, thus improving the financial sustainability of the system. Furthermore, forecasting models based on fundamental data can be used to model scenarios to evaluate how renewable energy best can be incorporated into the electricity grid.

**Conclusion:**

This project presents two multivariate machine learning models for predicting the electricity price via future contracts for two Swedish bidding areas. The models use fundamental data such as temperature, wind, hydro reservoirs, and commodity prices to predict the price of future contracts. The best performing model, the TCN model achived WAPE scores of 0.1 or below during normal market conditions with the best-recorded accuracy of 0.05 for quarterly contracts during 2021. The same model scored worse for stressed market conditions however with volatile results for different contracts. The TCN and LSTM model outperforms the ARIMA benchmark model for the same forecasting period within the same forecasting horizon. During more normal market conditions the accuracy for predicting future contracts is much higher compared to forecasts of the daily spot price, with the highest accuracy observed for forecasting long-term future contracts. TCN models are computationally efficient to use for long input sequences when fully trained and more efficient than LSTM models.