46765 Machine Learning for Energy Systems

Assignment 3: Day-ahead generation scheduling

Code: `https://github.com/albertkjoller/ml-energy-systems`

Albert Kjøller Jacobsen (s194253)
Ilias Manolakis (s222834)
Tais Abrahamsson (s193900)
Denisa Enache (s222964)
Andrei Lavric (s222965)

1st of December, 2023

Contents

Contribution table

	Model development	Coding	Results analysis	Report writing
Albert	50%	45%	60%	50%
Ilias	10%	10%	10%	5%
Tais	10%	10%	10%	10%
Denisa	15%	17.5%	10%	17.5%
Andrei	15%	17.5%	10%	17.5%

1 Introduction

In this assignment, the problem of solving the unit commitment problem from the perspective of the system operator is solved from two different approaches; *1)* following an optimization approach and *2)* by exploiting classification methods. Briefly explained, the unit commitment problem (UCP) relates to finding the most cost-efficient schedule of when and which power generation units should produce a to-be-determined amount of power while meeting the demand and satisfying the physical properties of generator units and transmission lines across a 24-hour time span. While the optimization approach is suitable for determining the optimal solution, this approach is computationally demanding as the UCP decision variables and constraints scales with the number of generator units, transmission lines, load profiles (capturing uncertainty in the system) and number of power nodes - as such learning from historical data by training a classification model might be useful for accelerating finding the solution to the problem on the premise of finding the exact optimal solution. Note, that we will exploit the outputs of the optimization model for a variety of demand samples to construct a data set of historical data that can be used for the classification model.

Throughout this assignment, a 118-bus system consisting of 54 generator units, $\mathcal{G}$, 91 loads, $\mathcal{N}$, and 186 transmission lines, $\mathcal{L}$. For notational clarity, the 24-hour time span will be denoted by $\mathcal{T}$ and is defined as $\mathcal{T} = \{1, 2, 3, \cdots 24\}$.

2 Optimization Model

2.1 Formulating the problem

Taking the role of the system operator aiming to solve the unit commitment problem, the goal is to minimize the production cost while satisfying the system requirements. For the 118-bus system, this leads to defining an objective function and a set of constraints that are described in the following sections.

We assume that the day-ahead operation- and start-up costs of all generator units are known in advance and hence should not be optimized for.

2.1.1 Objective function and decision variables

With the goal being minimizing production cost, the objective function can be expressed from the total cost of operating the power generating unit along with the cost associated to turning on a unit. Hence, we define the objective function as:

$$\min_{\boldsymbol{p},\boldsymbol{u},\boldsymbol{b},\boldsymbol{\delta},\boldsymbol{\epsilon}} \sum_{t\in\mathcal{T}} \left(\sum_{g\in\mathcal{G}} \underbrace{C_g^{op} \cdot p_{g,t}}_{\text{Production cost}} + \underbrace{C_g^{s} \cdot u_{g,t}}_{\text{Start-up cost}} \right) + \underbrace{M\left(\epsilon_t + \delta_t\right)}_{\text{Regularization}} \tag{2.1}$$

where C_g^{op} and C_g^s respectively describes the operation cost (i.e. production cost with unit g) and the start up cost for unit g, where the start-up cost term implicitly satisfies non-negativity since $u_{g,t}$ is a binary variable. In addition to the cost-specifying parts of the objective function, an additional term involving the slack variables is included. One can think of this as a regularizing term in the sense that larger values for the slack variable - meaning allowing for larger deviance between the total power generated and the demand - should be punished in the solution, eventually leading to as small a slack as possible when increasing the regularization strength M.

As is seen from the expression, the set of decision variables thus contains production, on/off status and start-up status of all generator units, $g \in \mathcal{G}$ for all hours, $t \in \mathcal{T}$, - i.e. $\boldsymbol{p}$, $\boldsymbol{b}$ and $\boldsymbol{u}$, respectively - along with the slack variables for all hours, $t \in \mathcal{T}$, - i.e. $\boldsymbol{\delta}$ and $\boldsymbol{\epsilon}$.

2.1.2 Power Balance Equation

The power balance equation captures the main constraint, namely, that the total load at each hour, $t \in \mathcal{T}$, can be delivered within the system. For ensuring feasibility of the solution, two non-negative-valued slack variables (ϵ_t and δ_t) are included - these are further remarked upon in Section 2.1.1.

$$\sum_{g\in\mathcal{G}} p_{g,t} = \sum_{n\in\mathcal{N}} D_{n,t}^{sample} + \epsilon_t - \delta_t, \qquad \forall t \in \mathcal{T} \tag{2.2}$$

$$\epsilon_t \geq 0, \qquad \delta_t \geq 0 \qquad \forall t \in \mathcal{T}$$

In addition to the power, $p_{g,t}$, generated at generator unit g at time t, the sample load, $D_{n,t}^{sample}$ at node n and time t is taken as an input. Here, *"sample"* denotes that the daily load profile can be varied for experimental purposes.

2.1.3 Generator Unit Constraints

Each generator unit is physically limited in terms of maximum capacities, minimum on and off times and regarding the variation of production for consecutive hours. These are described in details in the following:

Power generation limit: Each generator unit has a maximum production capacity. The following constraint ensures that the production of generator unit g at time t satisfies its associated maximum capacity.

$$\begin{aligned} b_{g,t} \cdot \underline{P_g} \leq & p_{g,t} \leq \bar{P}_g \cdot b_{g,t} && \forall g \in \mathcal{G}, \forall t \in \mathcal{T} \\ & b_{g,t} \in \{0,1\} && \forall g \in \mathcal{G}, \forall t \in \mathcal{T} \end{aligned} \tag{2.3}$$

As previously mentioned, $b_{g,t}$ denotes the on/off status (thus a binary variable) of the generator and $\underline{P_g}$ and $\bar{P}_g$ should be considered the minimum and maximum generation limits, respectively.

Start-up of generator unit: As should be obvious, the start-up status of a generator can be considered from whether the on/off status changed from off to on between consecutive time steps, thus:

$$\begin{aligned} u_{g,t} \geq & b_{g,t} - b_{g,t-1} && \forall g \in \mathcal{G}, \forall t \in \mathcal{T}_{[1::]} \\ u_{g,t} \in & \{0,1\} && \forall g \in \mathcal{G}, \forall t \in \mathcal{T}_{[1::]} \end{aligned} \tag{2.4}$$

where, $u_{g,t}$ denotes the start-up status of the generator g at time t. It should be noted that Equation 2.4 in itself does not capture the start-up property since e.g. $b_{g,t} = 0$ and $b_{g,t-1} = 1$ implies that $u_{g,t} \geq -1$, however, the binary constraint thus enforces $u_{g,t} = 0$ while still satisfying the inequality for this example. Note, that the inequality is defined for all timesteps except the first one, indicated by the [1 ::] subscript.

Minimum on- and off-times: The turning on of generators is one of the most strenuous operations, that the generator goes through and one that leads to the most wear and tear of the generator. Thus it is an operation that will need to be seriously limited. On the other hand a generator needs a certain period of time off allowing both for it to cool down and to carry out intermittent maintenance to assure reliable and robust operation. Because of these constraints each generator requires both a minimum off- and on-time. These minimum on and off times are modelled as follows:

$$-b_{g,t-1} + b_{g,t} - b_{g,\tau} \leq 0 \qquad \forall g \in \mathcal{G}, \forall \tau \in \{t, .., \bar{v}_g + t - 1\} \tag{2.5}$$

$$b_{g,t-1} - b_{g,t} + b_{g,\tau} \leq 1 \qquad \forall g \in \mathcal{G}, \forall \tau \in \{t, .., \underline{v}_g + t - 1\} \tag{2.6}$$

Here, Equation 2.5 constraints the minimum on-time specified by the input $\bar{v}_g$ while Equation 2.6 models the minimum off-time specified by the input $\underline{v_g}$.

> **Test ex. of eq. 2.5**: Assume the simplest case, where $b_0 = 0$, $b_1 = 1$ and $\bar{v}_g = 2$ for time $t = 1$. Let us now test, whether it is possible for $b_2 = 1$:
>
> $$\overbrace{\tau = t}^{1} \rightarrow \overbrace{-b_0}^{-0} \overbrace{+b_1}^{+1} \overbrace{-b_1}^{-1} = 0 \tag{2.7}$$
>
> $$\overbrace{\tau = \overbrace{\bar{v}_g}^{+2} \overbrace{+t}^{+1} \overbrace{-1}^{-1}}^{2} \rightarrow \overbrace{-b_0}^{-0} \overbrace{+b_1}^{+1} \overbrace{-b_2}^{-0} = \underline{\underline{1}} \tag{2.8}$$
>
> If this was the case, and $b_2 = 0$, then constraint (2.5) would **not** be upheld.
> Thus $b_2 = 1$ must be the case upholding the minimum on-time constraint.
>
> *A similar example can be shown for eq. 2.6 but has been omitted here for brevity.*

Ramping conditions: Lastly, the generator units are restricted to have a limited variation in production between consecutive hours. These constraints are specified as the *ramping constraints* and are presented in Equation 2.9.

$$\begin{aligned} p_{g,t} - p_{g,t-1} \leq & \bar{R}_g && \forall g \in \mathcal{G}, \forall t \in \mathcal{T} \\ p_{g,t-1} - p_{g,t} \leq & \bar{R}_g && \forall g \in \mathcal{G}, \forall t \in \mathcal{T} \end{aligned} \tag{2.9}$$

As is seen, the difference in production between consecutive time steps must satisfy the ramping inequality defined through the ramping input variable, $\bar{R}_g$, for generator g. Other implementations consider an upper and a lower ramping value, however, the data associated with this assignment only contains one ramping value per generator unit.

2.1.4 Line Technical Limits

For modeling the line technical limits, the first thing to realize is that the flow within a transmission line is bounded by its maximum capacity, $\bar{F}_l$ - i.e. the net flow, that can be either positive or negative depending on the main direction of the flow, should be upper and lower bounded by $\bar{F}_l$.

Additionally, the transmission line equation should enforce power to flow across the grid in order to meet the load at the bus nodes where it is needed at time t. For doing so, two matrices, H^{gen} and H^{load}, are constructed by taking the inner product of the bus-generator connection matrix and the bus-load connection matric, respectively, with the Power Transfer Distribution Factors (PTDF) that in essence describes incremental changes in transmission lines with respect to incremental changes at different nodes. As such, H^{gen} and H^{load} models the incremental changes in transmission lines as functions of the generator units and load units.

$$-\bar{F}_l \leq \underbrace{\sum_{g \in \mathcal{G}} H^{gen}_{l,g} \cdot p_{g,t}}_{\text{Generator term}} - \underbrace{\sum_{n \in \mathcal{N}} H^{load}_{l,n} \cdot D^{sample}_{n,t}}_{\text{Load term}} - \epsilon_t + \delta_t \leq \bar{F}_l \qquad \forall l \in \mathcal{L}, \forall t \in \mathcal{T} \tag{2.10}$$

As seen in Equation 2.10, the total flow through line l is thus restricted by ensuring that the generator flow term - i.e. how the generated power flows through a specific line l - balances the load flow term - i.e. how the load demand is requested through a specific line l - while satisfying the bounding capacity of the transmission line.

2.2 Analyzing the Optimization Model

As the UCP described in the previous section considers five time-dependent decision variables - three of which are also generator-dependent - a total of $24 \cdot (54 \cdot 3 + 2) = 3936$ variables to be solved for. In addition, the problem contains 23,006 constraints.

For validating whether the linear program found works as expected, the results obtained by inputting a single sample of a daily load profile are analyzed. Presented in Figure 2.1 is the summarized production and demand profile along with the separate production and price curves for the generators found as active - i.e. where the binary on/off variable is on for at least one of the 24 hours in the period.

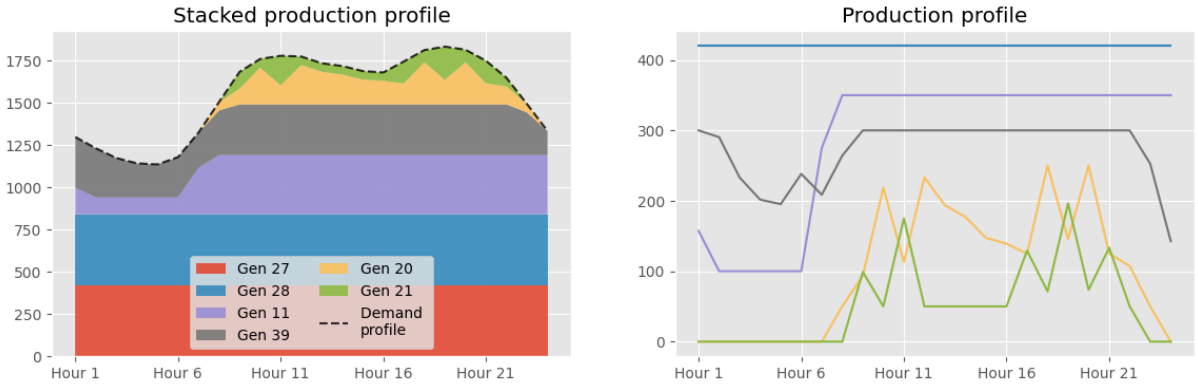


Figure 2.1: *Left:* Disaggregated daily production across active generators along with a sample demand profile. The power balance equation is validly satisfied. *Right:* The individual production profiles for the active generators. While some generators are constantly producing, others react on variations in the demand profile at night or to variations for peak load hours.

As seen in Figure 2.1 (*left*) the total demand profile is definitely met by the sum of power generated across units at all hours. Additionally, it is remarkable how the production is distributed across generators - as can be seen, generator units 27 and 28 are turned on across the entire time period and generating while generating at their maximum capacity, thereby providing power for the base load of the system. Generator units 11 and 39 are also turned on for the full period but both have to down-scale the production during

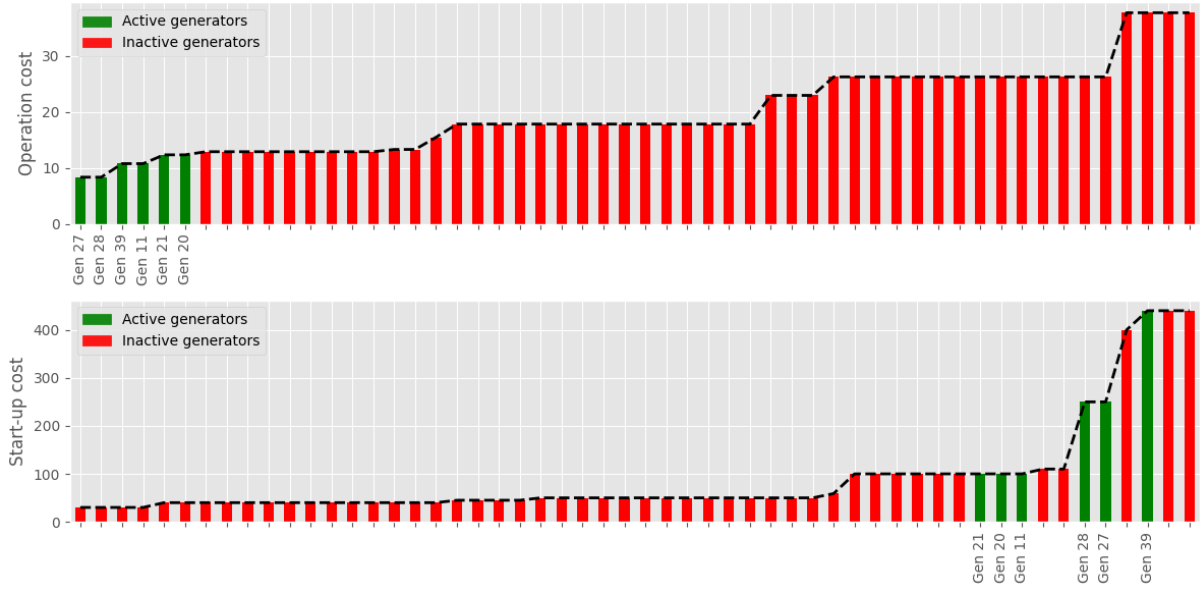


Figure 2.2: *Top:* Cost of operating each generator, sorted from lowest to highest price. Active generators relate to Figure 2.1 and are visualizes with green bars. *Bottom:* Cost of starting up each generated, sorted from lowest to highest price.

nightly hours where the demand is lower. Though the base load is covered, the demand during day time hours - and especially during peak load hours -, is not met by the production when only considering these four generator units. Instead, it becomes necessary to turn on generator unit 20 and 21 which are then flexibly able to capture the demand gap at peak hours. What is remarkable from the production curves of generator 20 and 21 is that the generators seem to compete wrt. being able to deliver the power - a fact that can be further backed up by verifying that their start-up and operation costs are identical, as seen in Figure 2.2.

When taking a closer look at Figure 2.2, it becomes quite evident what type of solution the linear program favors for this sample - namely, one minimizing cost by making the cheapest operating generators produce the power, while relying on the fact that start-up heavy generators can implicitly be on from the beginning (i.e. already running from previous day). When additional power is required, e.g. at peak load hours, other generators that are cheap to operate are turned on.

An interesting analysis regarding emission factors, would be possible if labels regarding the unit type - e.g. whether the plants are wind, solar, coal, etc. - were accessible. In the case at hand the best guess could be that the base load is governed by non-renewable production plants as these are steady and active across the full day and rather expensive to start up, whereas the flexible production coming from unit 20 and 21 could be from renewables.

Active Transmission Lines

Next, the attention is turned towards determining active constraints for transmission lines, i.e. whether the power sent through a transmission line reaches the maximum capacity of the cable. For brevity, the center part of the line technical limits (i.e. Equation 2.10) is denoted as x while only regarding the positive bound in the following. It should be noted that similar arguments can be made for the negative flow constraint.

Using Gurobi as the solver for the optimization part, inequalities are implicitly represented as equalities by associating a slack value to each constraint, thus:

$$x \leq \bar{F}_t \qquad \Rightarrow \qquad x + \epsilon = \bar{F}_t$$

As such, the value of the slack-variable is given by $\epsilon = \bar{F}_t - x$ and is thus only zero when the inequality is exactly an equality. This is exactly the definition of an *active* constraint for which reason we can rely on Gurobi for determining active constraints.

Using the Gurobi model to assess the set of active line constraints yields that no lines are reaching their maximum capacity (i.e. the constraints are inactive) for the specific load sample used for the solution in Figure 2.1. This is however load-profile-dependent.

Extracting Historical Data

What should be evident by now, that the input variable governing the uncertainty of the optimization solution is the load profile. For extracting historical data about how the optimization model behaves under uncertainty, the first 1386 sample load profiles from the provided examples were inputted to the model after which the binary variables regarding on/off status, start-up status as well as the activeness on all lines were stored, resulting in a total of 88 active line constraints.

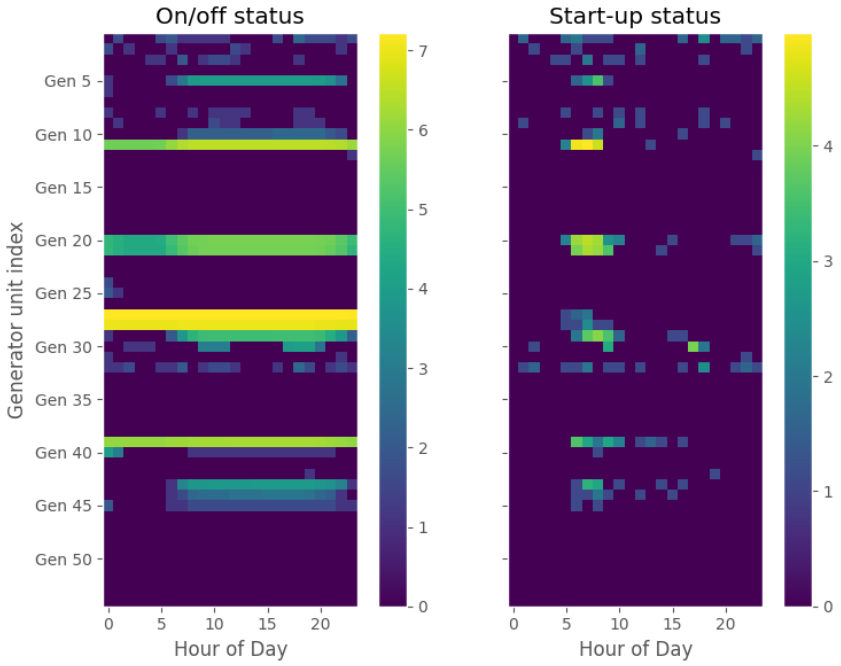


Figure 2.3: *Left:* Total number of times each generator is turned on grouped by all hours of a day. *Right:* Total number of times each generator is started grouped by all hours of a day. Remark that the colorbar is given in log-scale for better being able to compare high-valued generators to low-valued generators.

3 Machine Learning Model

3.1 Why use Machine Learning for solving the Unit Commitment Problem?

As the 118-bus system with 54 generator units, 91 loads and 186 transmission lines is a rather complex system, the associated linear program used for solving the scheduling problem becomes rather large as the number of parameters and constraints scales with the system's configuration. In practice, this means that the optimization model can be rather slow when doing inference, especially for large-scale systems. Considering the problem from a Machine Learning point-of-view instead, the inference time might significantly speed up - from seconds or minutes to milliseconds. Additionally, the machine learning solution might be able to capture the underlying model without exploiting all inputs required for the optimization model, as information about e.g. minimum on/off times would implicitly be captured by the binary data series describing the on/off status of generator units - in other words, the machine learning model could implicitly learn hidden structures due to correlations in the data set.

3.2 Linear and non-linear approaches to classification

Our approach to predict the optimal value of the binary variables and active constraints is a SVM (Support vector machine) classifier that has the advantage of easily being able to go from a linear to a non-linear classification model. For the linear case, the main idea is to create a decision boundary plane that from which the two binary classes of the problem can be distinguished. However, there are multiple solution (i.e. non-uniqueness) in terms of what plane to choose, for which reason the SVM algorithm considers finding the optimal one.

For a given set of point of the form $(x_1, y_1), (x_2, y_2), ..(x_n, y_n)$, where y_i is either 1 or -1, each indicating the class to which the point x_i belongs, the goal is to find the maximum-margin hyperplane that divides

the group of points depending on the y_i values. A hyperplane in a D-dimensional Euclidean space is a flat, *D-1* dimensional subset of that space that divides the space into two disconnected parts. The equation of the searched hyperplane has the following form:

$$\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{b} = 0 \tag{3.1}$$

where $\boldsymbol{w}$ is the normal vector (i.e. perpendicular) to the hyperplane.

For cases where the classes are linearly separable, the data can be separated by two parallel hyperplanes. The region between these is denoted as the margin while the hyperplane that lies halfway between them is known as the maximum-margin hyperplane. Having the hyperplane equation and the maximum-margin defined, these hyperplanes can be described by the following two equations:

$$\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{b} = 1, \qquad \text{(points above this line are part of the class 1)} \tag{3.2}$$

$$\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{b} = -1, \qquad \text{(points below this line are part of the class -1).} \tag{3.3}$$

Mathematically, the distance between them is $\frac{2}{\|\boldsymbol{w}\|}$. Maximizing the distance means that we need to minimize $\|\boldsymbol{w}\|$. The constraints tell us that the points need to lie on the correct side and can be rewritten as

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i - b) \geq 1, \quad \text{for all } 1 \leq i \leq n.$$

By combining the above formulas, the problem can be written as a convex optimisation problem where we aim at minimizing the squared ℓ_2 norm (regularization):

$$\begin{aligned} &\underset{\boldsymbol{w},\, b}{\text{minimize}} && \|\boldsymbol{w}\|_2^2 \\ &\text{subject to} && y_i(\boldsymbol{w}^\top\boldsymbol{x}_i - b) \geq 1 \quad \forall i \in \{1, \ldots, n\} \end{aligned} \tag{3.4}$$

One of the most powerful tools that can be used in conjunction with SVM classifiers is the kernel trick. This procedure can drastically increase the performance of the model by easily allowing for non-linear decision boundaries. In the general form, the classification vector $\boldsymbol{w}$ can be seen as a linear combination between input and output:

$$\boldsymbol{w} = \sum_i^m \alpha_i y_i \boldsymbol{x}_i \tag{3.5}$$

By exploiting this rewriting of $\boldsymbol{w}$ for the minimization task, the optimization equations presented in 3.4 take the following form:

$$\underset{\boldsymbol{w},\, b}{\text{minimize}} = \sum_i^m \sum_j^m \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \tag{3.6}$$

For modeling non-linear data, the inner product between input data, i.e. $\boldsymbol{x}_i^T\boldsymbol{x}_j$, can be mapped to a higher-dimensional space through a design function, $\phi(\cdot)$, thus yielding:

$$\underset{\boldsymbol{w},\, b}{\text{minimize}} = \sum_i^m \sum_j^m \alpha_i \alpha_j y_i y_j \phi\left(\boldsymbol{x}_i\right)^T \phi\left(\boldsymbol{x}_j\right) \tag{3.7}$$

In practice we exploit the kernel trick, stating that inner products can be "kernelized", i.e. that instead of computing an inner product in a high-dimensional space, the inner product can be computed by a kernel function in the original low-dimensional data space. This is computationally much more efficient and allows for solving rather complex non-linear tasks in low dimensions while exploiting high-dimensional processing. This is formalized by:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi\left(\boldsymbol{x}_i\right)^T \phi\left(\boldsymbol{x}_j\right) \tag{3.8}$$

Building kernels is a large research topic, however, a large variety of kernels functions already exists and are anlyzed. Two such examples are:

- the **Polynomial** kernel of order d: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^T \cdot \boldsymbol{x}_j)^d$
- the **Gaussian / RBF** kernel: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma}\right)$

What is well-known throughout literature is that the high-dimensional processing space used for these kernels is d-dependent for the Polynomial kernel while the Gaussain kernel has infinite support, thus implicitly mapping data through $\phi(\cdot)$ to inifinite dimensions.

3.3 Creating the data set

Using predictions of the on/off status of the generators, one could easily obtain predictions for the start-up status, as these statuses are deterministically dependent on each other. Additionally, we consider the task of predicting the activeness (or overload) of the transmission lines.

For both tasks, an input data set and a target vector needs to be defined. We argue that the sample load profiles used as input to the optimization model along with the extracted on/off and activeness data should implicitly contain both temporal information as well as the minimum on/off time constraints, etc. - thereby the task of the machine learning model will be to learn how the optimization model reacts solely when the load profile changes. Note, that this is a modeling choice and that other data could additionally have been included. As such, the data set considered consists of 1,200 load samples with $\mathcal{N} \cdot \mathcal{T} = 91 \cdot 24 = 2184$ features.

On/off status targets

As seen in Figure 2.3, most of the generators are never turned on for which reason training classification models for these generator units does not make much sense. For restricting the classification task we consider all generators that have generated power for all hours on day across all samples - i.e. not necessarily in a consecutive period of 24 hours, resulting in modeling the exact same generators as described in Figure 2.1. This choice is made since the unit commitment problem considers a day-ahead scheduling task where predictions need to be carried out for all hours of a day at once - in this setting, a support vector machine would only know about a single class label (i.e. *turned off*) and would thus fail in constructing a decision boundary between binary labels.

For each of the 6 generators considered, the target attribute associated to a load sample is 24-valued binary vector, indicating whether that generator was on at the associated hours.

Activeness/Overload of Transmission Lines

Regarding the binary classifier for the active constraints concerning the transmission lines, we initially considered following the same strategy as for the on/off status. However, as was also previously mentioned, the number of active lines found with the chosen set of load samples was very low (88 active/overloaded lines) for which solving the problem in a data-driven manner is infeasible. To fix the issue of few active lines, a new data extraction was run where load samples were multiplies by a scalar of 2, thus pushing the overall load of the system. This validly resulted in finding a total of 4312 active line constraints, however, the machine learning analysis was not run due to the limited time for the project. We emphasize that the proposed modeling and analysis procedure would have followed that of the on/off predictions, thus running the active line predictor would not have added anything new to the assignment in terms of applying relevant theory.

3.4 Modeling and Evaluation Approach

For predicting all the on/off status for all 24 hours at once, we fit a multi-label support vector classification (SVC) model to each generator. In practice this is done using the `MultiOutputClassifier` and `SVC` objects from `scikit-learn` - parallelized by specifying the `n_jobs`-parameter to -1. Then, each multi-label classifier is trained on a fixed training set, defined as 80% of the extracted data, thus, evaluating model performances on the remaining 20% of the data. For comparing model performances, the balanced accuracy - that accounts for imbalance in the target distribution at each hour, thereby having random performance equal 0.5 - is computed along with the Area-Under-the-Curve (AUC), F1, recall and precision scores. We are aware that properly comparing model performances requires using cross-validation for estimating the generalization error, however, due to the limited time given for the project (and since it was done in previous projects), the main focus was put elsewhere.

Using this modeling and evaluation approach, three major experiments were run:

1. a Principle Component Analysis.
2. an investigation of the importance of the kernel type (i.e. linear vs. non-linear classification) when training on 2 Principle Components.
3. investigating change in model performance of non-linear model when using 10 Principle Components.

While, the initial thought was to compare training using the full data set vs. the 10 most dominant Principle Components (PC) for both the linear and non-linear classification model, this turned out to be computationally infeasible for the linear model given the size of the data set. This should come as no major surprise since the linear SVC model evaluates the kernel wrt. all training points and since the linear kernel is non-sparse compared to e.g. the Gaussian kernel where the kernel value evaluates to 0 when points are more distant to each other than approx. 3 standard deviations. For this reason, experiment 2) was run as a "proof-of-concept" for comparing kernels solely using 2 PCs.

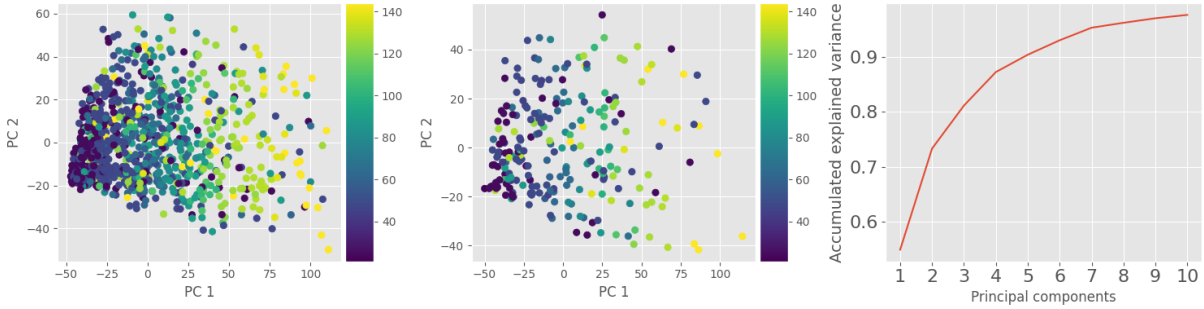


Figure 3.1: *Left:* Training data projected onto the two largest PCs, computed from the training data. *Mid:* Test data projected onto the same two PCs. *Right:* Accumulated explained variance ratio, revealing that 75% variance is captured by the first 2 PCs. For *left* and *mid*, the color of a sample indicates the total number of generator-hour pairs turned out for that sample, i.e. $\sum_{t\in\mathcal{T}}\sum_{g\in\mathcal{G}} y_{g,t}$.

3.5 Results & Discussions

PCA for Dimensionality Reduction

Figure 3.1 visualizes all training (*left*) and test data (*mid*) projected onto the two largest PCs, as computed from the training data. The distribution of colors appear to have an increasing structure (from left to right), suggesting that the load profiles that yield a lot of turned on generators resemble each other with the opposite also holding. In addition, the accumulated explained variance ratio is included (*right*), revealing that 90% of the variance is explained from approx. 5 PCs while approx. 75% is explained by the first 2 PCs. Thus, the take away is that the dimensionality of the data (2184) can be significantly reduced if instead using a handful of Principal Components as features without loosing much information.

Experimenting with Kernel Type

In Table 1, the performance results associated with running the SVC model using a linear kernel, polynomial kernel and a Gaussian kernel - on a data set with features defined as the largest two principle components - are presented. Though the missing uncertainty of the performance metrics makes it statistically impossible to compare the models, the results suggest that the Gaussian kernel is the better modeling approach for all generator units, if basing the choice on balanced accuracy. However, if looking at the AUC, the linear classification model outperforms the Gaussian-based model for some generators - most notably for Generator 27 that based on the balanced accuracy performs equal to random guessing. What is furthermore noticeable is that Generator 27 and 28 obtain very high recall scores - meaning that they capture all positive labels - while also being quite precise. Though this seems counter-intuitive after stating that these models are barely better than random when looking at balanced accuracy, there is an underlying reason governing the high precision/recall scores: the recall score of 1.0 for Generator 27 indicates that all of the active hours (when generator is on) were captured correctly by the model while the precision scores of 0.984 indicate that most of the predicted active hours are actually active. Combining these observations, it should be rather obvious that this could occur from a setting where the model always predicts the generator to be on while there in fact are very limited hours where the generator is off. Considering the outline of Figure 2.3, this seems rather likely as it appears that Generator 27 and Generator 28 are almost always turned on, suggesting that the learning problem suffers from the large class-imbalances.

Table 1: Average performances (across all hours) for each generator using a linear, polynomial and Gaussian (RBF) kernel within the SVC models. Notice, that the models were only evaluated on a single test set, thus no uncertainty range of the generalization errors are included.

		Bal. Acc.	AUC	F1 score	Recall	Precision
Gen 11	Linear	0.702	0.765	0.623	0.579	0.711
	Polynomial	0.603	**0.769**	0.403	0.286	0.712
	Gaussian	**0.729**	0.765	**0.672**	**0.639**	**0.727**
Gen 20	Linear	0.564	**0.750**	0.217	0.150	0.678
	Polynomial	0.544	0.738	0.172	0.107	0.592
	Gaussian	**0.641**	0.723	**0.414**	**0.311**	**0.789**
Gen 21	Linear	0.554	**0.793**	0.182	0.126	0.653
	Polynomial	0.554	0.781	0.199	0.127	0.612
	Gaussian	**0.653**	0.725	**0.440**	**0.336**	**0.659**
Gen 27	Linear	0.500	**0.914**	0.992	1.000	0.984
	Polynomial	0.500	0.743	0.992	1.000	0.984
	Gaussian	0.500	0.761	0.992	1.000	0.984
Gen 28	Linear	0.500	0.744	0.888	**1.000**	0.799
	Polynomial	0.507	0.678	0.889	0.998	0.802
	Gaussian	**0.633**	**0.772**	**0.892**	0.939	**0.849**
Gen 39	Linear	0.726	0.787	0.634	0.583	0.703
	Polynomial	0.626	0.791	0.437	0.311	**0.736**
	Gaussian	**0.746**	**0.801**	**0.666**	**0.647**	0.689

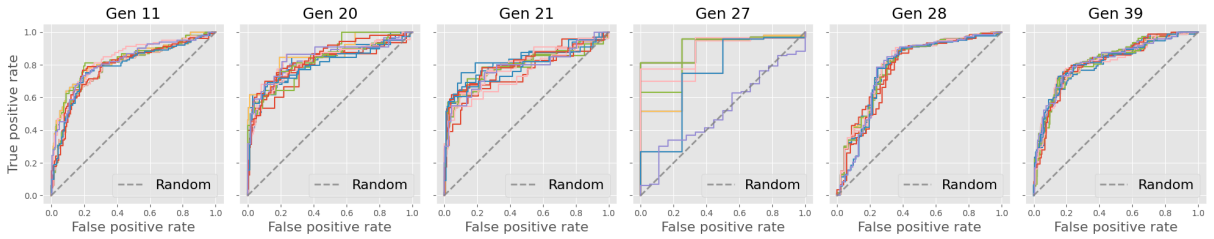
To combat such class-imbalance, one could exploit under- and oversampling techniques for equalizing class count (i.e. number of on- and off hours) per generator per hour. However, we argue that the data imbalance issues originate from the underlying data generation approach since the optimization model is deterministic in terms of start-up and operation costs, thereby restricting the order that generators will be turned on at varying the demand load profiles to a limited set of possible outcomes. Thus, having enough data such that undersampling yields appropriately sized data sets requires extracting data from a very large number of demand profile samples.

Comparing PCA-based Models with Full-Dimensionality Models

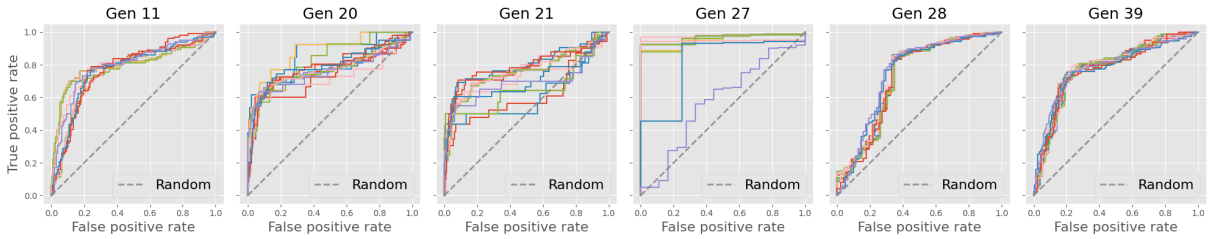
Figure 3.2 visualizes the ROC curves for all hours of all considered generators when using either the training set with all 2184 features or a training set with features defined as the 10 dominating Principal Components. For both cases, the Gaussian kernel was used as it revealed better results (in terms of balanced accuracy) for the previous experiment. Additional performance results are reported in Table 2.

Table 2: Average performances (across all hours) for each generator using a SVC model with a Gaussian kernel. *Left:* full data set.*Right:* 10 Principal Components.

	Bal. Acc.	AUC	F1 score	Recall	Precision	Bal. Acc.	AUC	F1 score	Recall	Precision
Gen 11	0.725	0.798	0.669	0.629	0.733	0.716	0.770	0.653	0.618	0.716
Gen 20	0.655	0.825	0.447	0.331	0.855	0.618	0.784	0.359	0.263	0.759
Gen 21	0.679	0.791	0.504	0.381	0.847	0.631	0.723	0.387	0.288	0.818
Gen 27	0.500	0.864	0.992	1.000	0.984	0.500	0.930	0.992	1.000	0.984
Gen 28	0.601	0.784	0.892	0.955	0.836	0.638	0.727	0.895	0.945	0.850
Gen 39	0.757	0.819	0.681	0.665	0.670	0.709	0.778	0.613	0.572	0.664



(a) Full data set



(b) 10 Principal Components

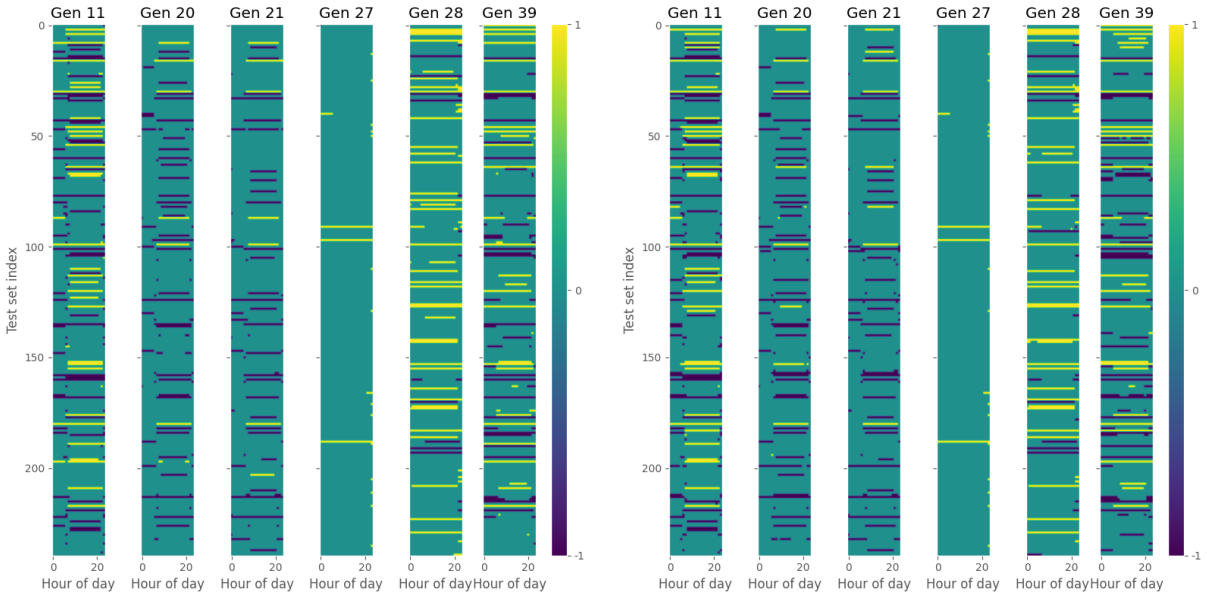
Figure 3.2: ROC curves for all hours (denoted by colors within each sub plot) for each generator using a SVC model with a Gaussian kernel. *Top:* full data set.*Bottom:* 10 Principal Components.

As seen from Figure 3.2 (when disregarding Generator 27), the model performance on all hours for each generator are somewhat consistent as the ROC curves match when using the full dimensionality of the data set. In terminology of fairness in machine learning, this suggests that the models for each generator are fair across hours as the error rates are equalized (also known as the *separation criteria*) - see Fairness in Machine Learning book for further information[1]. Notice, that all ROC curves lies above the diagonal, indicating that the AUC score are higher than 0.5 and that the performances are better than random, as is also seen from the balanced accuracy and AUC scores in Table 2.

What is furthermore seen from Figure 3.2 and Table 2 is that the models clearly react when reducing the dimensionality. As is seen, the ROC curves adjust resulting in generally lower AUC scores and balanced accuracies (except for Generator 27), however, to a minor extent within 4 percent points. In contrast to the globally trained model, Generator 20 and 21 now vary more in terms of error rates across hours, thus not satisfying the definition of being fair ito. the separation criteria. In other words, using the PCA-based model implies a bias in terms of performance on specific hours for these two generators. This could be a problem as the cost of making prediction errors is indirectly hour-dependent in terms of matching the total demand.

Taking a more qualitative evaluation approach, Figure 3.3 presents prediction errors for the two modeling approaches. As was also suggested by the performances in 2 the models behave similarly depdening on whether or not the reduced dimensionality feature set is used. Most noticeable, is the change of predictions for the first handful of samples of Generator 39 where the model has learned to correct for wrongly predicting the generator on for all hours of the day when using the full dimensionality. For Generator 20 and 21, the model fails to detect active hours more frequently when using the PCA-reduced training set (i.e. there are more dark areas) - this can additionally be verified by considering performance drop in terms of recall as stated in Table 2.

[1]Fairness in Machine Learning: `https://fairmlbook.org/classification.html`



(a) Full data set

(b) 10 Principal Components

Figure 3.3: Prediction errors of each hour for each generator considered. The dark blue areas reveal cases where the model fails to predict the generator being on, the yellow areas reveal cases where the model fails to predict the generator being on, while the green areas show the cases where the model is correct. Notice how the errors of Generator 27 are mostly zero while only being wrong for cases where the generator was off - this suggests that the model *only* predicts the positive label across all samples.

For the problem at hand - namely, day-ahead scheduling - one could argue that both a high recall and precision is required. First of all, a high recall would capture that the model should capture as many of the hours where power is actually generated as possible. While a high recall could be obtained by deciding to have all generators on at all hours, this would in practice overload the system as the power-demand relation would be far from balanced. Thus the need for also taking precision into account, in order of being more restrictive in terms of *what* generators are turned on *when*.

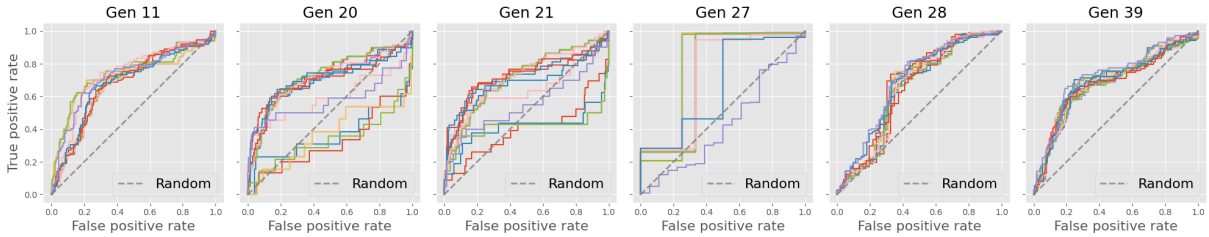
4 Conclusion

Though the idea of approaching the day-ahead scheduling problem through machine learning rather than solving a linear program is good since the inference time can be significantly speeded up while learning general trends of the system, the machine learning solution suffers from not necessarily finding the optimal solution. As such, one would in practice have to determine whether the optimal solution is key for the practical decision-making problem or if the performance can be traded-off for favoring a faster running time. In addition, we argued that naively exploiting data extracted from running the linear program for various load profiles induces a class-imbalance that negatively affects model performances, which was especially seen for one Generator unit, that only learned to predict the positive label. If time had allowed, one would thus have to extract a larger data set and adjust for class imbalance by utilizing under- or oversampling techniques.

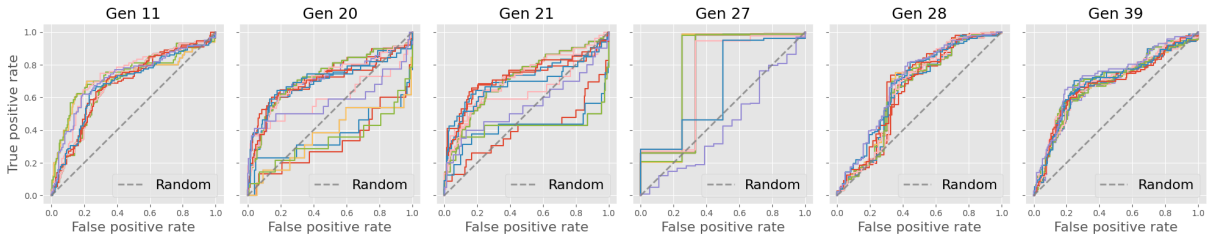
Overall, a PCA-based model gives similar resuts as when exploiting the full dimensionality of the training data, yet, speeds up the training time. If using a support vector classification model that relies on evaluating kernel functions - for which computation time increases with dimensionality -, using PCA for reducing the complexity of the task might be key for being able to train a model at all, when considering larger data sets.

Appendix

A Additional results



(a) Full data set



(b) 10 Principal Components

Figure A.1: ROC curves for all hours (denoted by colors within each sub plot) for each generator using a SVC model with a Polynomial kernel. *Top:* full data set.*Bottom:* 10 Principal Components.

Table 4: Average performances (across all hours) for each generator using a SVC model with a Polynomial kernel. *Left:* full data set.*Right:* 10 Principal Components.

	Bal. Acc.	AUC	F1 score	Recall	Precision	Bal. Acc.	AUC	F1 score	Recall	Precision
Gen 11	0.663	0.771	0.548	0.451	0.717	0.595	0.696	0.431	0.330	0.640
Gen 20	0.637	0.755	0.423	0.299	0.738	0.570	0.614	0.246	0.167	0.630
Gen 21	0.634	0.738	0.411	0.293	0.719	0.581	0.629	0.276	0.184	0.714
Gen 27	0.500	0.835	0.992	1.000	0.984	0.500	0.736	0.992	1.000	0.984
Gen 28	0.525	0.719	0.886	0.981	0.809	0.535	0.650	0.892	0.989	0.812
Gen 39	0.679	0.805	0.550	0.440	0.741	0.597	0.686	0.402	0.305	0.592