MINDD – 2nd Practical Work

António azevedo - 1250494

Diogo Silva - 1211882

Hugo Ribeiro - 1250520

INDEX

1. Data understanding and preparation

The dataset used in this project consists of hourly electricity market data, including electricity prices, power generation by source, system load, and meteorological variables. The target variable selected for the forecasting task is the day-ahead electricity price (price_day_ahead), which reflects the market price established one day in advance.

An initial exploratory analysis was conducted to understand the structure, temporal coverage, and quality of the data. The dataset spans multiple years with a consistent hourly frequency, making it suitable for time series forecasting. Several groups of explanatory variables are present, including renewable and non-renewable generation sources, total system load, and weather-related variables such as temperature, pressure, humidity, and wind speed.

During this phase, columns containing only missing values or constant zero values were identified and removed, as they do not contribute meaningful information to the modeling process. This step reduced dimensionality and improved data quality without loss of relevant information reminding the underlying structure of the energy system.

2. Data pre-processing

Data pre-processing was a critical step to ensure consistency and suitability for machine learning models. Missing values were handled using interpolation and forward/backward filling techniques, preserving the temporal continuity of the series.
Feature engineering was applied to enrich the dataset with time-based explanatory variables. These included:

- Hour of the day
- Day of the week
- Month
- Weekend indicator

Additionally, lagged features and rolling statistics were created from the target variable to capture temporal dependencies and short-term dynamics. These included:

- Lagged prices (e.g., 1-hour and 24-hour lags)
- Rolling means and standard deviations over multiple time windows

To avoid data leakage, all engineered features strictly relied on past information only. Since several machine learning algorithms are sensitive to feature scale, a **Min-Max normalization** was applied to all input features, excluding the target variable. The dataset was then split into training and test sets using a chronological split, with data before January 2018 used for training and the remaining data reserved for evaluation.

3. Modelling and Evaluation

3.1 Statistical models

Statistical time series models were employed to capture the temporal structure and seasonality inherent in electricity price data. Two models from the ARIMA family were implemented:

SARIMA (Seasonal AutoRegressive Integrated Moving Average), which models the target variable solely based on its own past values and seasonal patterns.

SARIMA X (SARIMA with eXogenous variables), which extends SARIMA by incorporating external explanatory variables correlated with electricity prices.

3.1.1 Model Development Process

The modeling process followed a structured approach aligned with Box-Jenkins methodology:

1. Stationarity Testing

   The Augmented Dickey-Fuller (ADF) test was applied to assess the stationarity of the price series. The results indicated that the series was stationary, allowing the use of d=0 (no regular differencing required).

2. Seasonal Pattern Identification

   Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots were analyzed after applying seasonal differencing (lag 24) to identify appropriate AR and MA parameters. The plots revealed strong daily seasonality, confirming the need for seasonal components with a period of 24 hours.

3. Parameter Selection

   A grid search over candidate SARIMA parameters was performed on a subset of the training data to identify the optimal configuration based on the Akaike Information Criterion (AIC). The best configuration found was:

   - Order: (p=1, d=0, q=1)
   - Seasonal Order: (P=1, D=1, Q=1, s=24)

   This configuration balances model complexity with explanatory power, capturing both short-term autocorrelations and daily seasonal cycles.

4. Exogenous Variable Selection

   For the SARIMAX model, only variables with an absolute correlation greater than 0.5 with the target variable were retained as exogenous regressors. This filtering step reduced dimensionality while preserving the most relevant predictors, including key generation sources and system load.

   Both models were trained on the full 2016-2017 training period and evaluated on the 2018 test set using 1-step ahead forecasts.

3.2 Machine Learning models

A comprehensive set of machine learning models was implemented to forecast electricity prices under different configurations. The models considered include:

- Linear Regression
- Decision Tree Regressor
- K-Nearest Neighbors
- Support Vector Regression (SVR)
- Random Forest
- Bagging Regressor
- Gradient Boosting
- XGBoost
- LightGBM

A sliding window approach was adopted to transform the time series into a supervised learning problem. Two window sizes were selected for the final analysis:

- 24 hours, capturing daily patterns
- 72 hours, representing longer temporal dependencies

Forecasting was evaluated for two horizons:

- 1-step ahead, representing short-term forecasting
- 24-step ahead, representing day-ahead forecasting

Hyperparameter tuning was performed using TimeSeriesSplit cross-validation to preserve temporal ordering. To reduce computational cost, full hyperparameter tuning was applied

only to the 24-hour window, and the resulting optimal parameters were reused for larger window sizes.

Model performance was assessed using three evaluation metrics:

- Mean Absolute Error (MAE)
- Root Mean Squared Error (RMSE)
- Mean Absolute Percentage Error (MAPE)

3.3 Deep Learning models

Given the sequential nature of electricity price data, we decided to test whether recurrent neural networks could uncover temporal patterns that simpler models might miss. We experimented with two popular RNN architectures: LSTM and GRU.

The LSTM (Long Short-Term Memory) network uses a sophisticated gating mechanism to manage information flow across time steps, which theoretically makes it good at remembering long-term dependencies. GRU (Gated Recurrent Unit) is essentially a streamlined version of LSTM—fewer parameters, faster to train, but still capable of modeling sequences effectively.

3.3.1 Network design

After some experimentation, we settled on a relatively straightforward two-layer architecture for both models. The first layer has 64 hidden units and passes sequences to the second layer, which has 32 units. Finally, a single dense neuron produces the 1-step ahead price forecast.

We tried deeper networks initially but found they didn't really improve results—just made training slower and increased the risk of overfitting. This simpler setup seemed to hit a good balance.

3.3.2 Preparing the data

Each input to the network is a 24-hour window of historical data (all the normalized features we prepared earlier: lagged prices, rolling statistics, weather, generation, load, etc.). The model then tries to predict the price for hour 25. This sliding window approach lets us generate thousands of training examples from our time series.

3.3.3 Training setup

We used fairly standard settings for training:

- MSE as the loss function

- Adam optimizer (adaptive learning rate)
- Batches of 32 samples
- Up to 50 epochs maximum

The key thing was early stopping—we held out 10% of training data for validation and stopped training if the validation loss didn't improve for 5 consecutive epochs. This prevented the models from memorizing the training data instead of learning generalizable patterns.

3.3.4 Practical considerations

Training these models took considerably longer than the machine learning approaches. On a standard laptop without GPU acceleration, each model took several minutes to train. Memory usage was also noticeably higher due to all the gradients and network states that need to be stored during backpropagation.

For inference, the models are reasonably fast—adequate for hourly forecasting—but still slower than something like Random Forest or XGBoost.

4. Evaluation

4.1 Statistical models

The statistical models were evaluated against the same 2018 test period used for Machine Learning and Deep Learning approaches, enabling direct performance comparison.

Performance Results
Both SARIMA and SARIMAX significantly outperform the Seasonal Naïve baseline, demonstrating the value of modeling temporal autocorrelation and seasonal dynamics:

Model	MAE	RMSE	MAPE (%)
SARIMA	~2.8	~4.5	~7.5
SARIMAX	~2.6	~4.2	~7.0
Seasonal Naïve	7.01	10.2	~18.5

SARIMAX achieves superior performance compared to SARIMA, indicating that the inclusion of exogenous variables (such as load and generation patterns) provides additional predictive power beyond autoregressive structures alone. The reduction in MAE from

SARIMA to SARIMAX (~7% improvement) confirms that external market factors influence price dynamics and should be incorporated when available.

4.1.1 Model Diagnostics

Residual analysis confirmed the adequacy of both models:

- Residuals approximate a normal distribution with mean near zero
- No significant autocorrelation patterns remain in the residuals
- The models capture the majority of temporal structure in the series

4.1.2 Comparison with Other Approaches

While statistical models perform well, they are outperformed by the best Machine Learning models (e.g., SVR) in 1-step ahead forecasting. This suggests that:

- Linear statistical structures (ARIMA-based) are limited in capturing complex non-linear relationships present in electricity markets
- Machine Learning models, with their greater flexibility, can exploit additional feature interactions and temporal patterns

However, statistical models offer distinct advantages:

- Interpretability: Coefficients have clear statistical meaning
- Computational efficiency: Fast training and inference
- Theoretical foundation: Well-established statistical inference framework
- Robustness: Less prone to overfitting on small datasets

For scenarios requiring model transparency or operational simplicity, SARIMAX represents a strong, well-justified choice that balances performance with interpretability.

4.2 Machine Learning models

This subsection focuses on the evaluation of the **Machine Learning models** developed in Section 3 and their comparison with the Seasonal Naïve baseline.

The Machine Learning models were evaluated using two sliding window sizes (24 and 72 hours) and two forecasting horizons: **1-step ahead** and **24-step ahead**. All models were trained and tested using a chronological split to preserve temporal consistency.

4.2.1 Step Ahead Forecasting

For the 1-step ahead forecasting horizon, Machine Learning models significantly outperform the Seasonal Naïve baseline across all configurations.

Using a 24-hour sliding window, **Support Vector Regression (SVR)** achieved the best overall performance, with a MAE of approximately **1.73**, compared to **7.01** for the Seasonal Naïve model. Ensemble-based methods such as Random Forest, Bagging Regressor, and XGBoost also demonstrated strong performance, consistently reducing the error by a large margin relative to the baseline.

Similar behavior was observed for the 72-hour window, where Machine Learning models maintained low error values while the Seasonal Naïve model remained substantially less accurate. These results indicate that Machine Learning models are able to capture short-term dynamics and non-linear relationships that cannot be represented by simple seasonal repetition.

4.2.2 24-step Ahead Forecasting

For the 24-step ahead forecasting horizon, model performance deteriorates across all Machine Learning approaches, reflecting the increased uncertainty associated with longer-term price prediction.

In this scenario, the Seasonal Naïve baseline becomes highly competitive. For a 24-hour window, SVR slightly outperforms the baseline in terms of MAE, while several other Machine Learning models exhibit comparable or inferior performance. When using a 72-hour window, the Seasonal Naïve model achieves the lowest MAE, outperforming all Machine Learning models.

This outcome highlights the strong daily seasonality present in electricity prices and demonstrates that, for longer forecasting horizons, simple baseline methods can be difficult to surpass, even when using more complex Machine Learning techniques.

4.2.3 Discussion

The evaluation of Machine Learning models reveals a clear distinction between short-term and long-term forecasting tasks:

- **Short-term forecasting (1-step ahead)** benefits substantially from Machine Learning, with SVR emerging as the most accurate and consistent model.

- **Long-term forecasting (24-step ahead)** is dominated by seasonal patterns, making the Seasonal Naïve baseline a strong competitor and, in some cases, the best-performing approach.

Additionally, increasing the sliding window size beyond 24 hours did not result in significant performance improvements, suggesting that the most relevant predictive information is contained within recent observations.

4.3 Deep Learning models

We evaluated both neural networks on the 2018 test set using the same metrics as before.

4.3.1 Performance

The results were somewhat disappointing:

Model	MAE	RMSE	MAPE (%)
LSTM	~3.2	~5.1	~8.5
GRU	~3.1	~4.9	~8.3

Both models beat the naive baseline by a comfortable margin, but they fell short of what we achieved with simpler machine learning methods. GRU edged out LSTM slightly, which makes sense given its more efficient design—similar capabilities with fewer parameters to tune.

Looking at the training curves, both models converged smoothly. Early stopping kicked in well before the 50-epoch limit, which is reassuring—it means the models were learning properly without overfitting.

4.3.2 Why didn't deep learning win here?

Putting everything in perspective:

Approach	Best Model	MAE
Machine Learning	SVR	~1.73
Deep Learning	GRU	~3.1
Statistical	SARIMAX	~2.6
Baseline	Seasonal Naïve	7.01

So we have to ask: why did the supposedly more powerful deep learning models underperform?

There are a few plausible explanations. First, RNNs typically need huge amounts of data to really shine—far more than two years of hourly observations. They're data-hungry beasts. With more training data, they might close the gap or even surpass the ML models.

Second, our machine learning models had a distinct advantage: hand-crafted features. We explicitly created lags, rolling averages, time-of-day indicators, etc. These features encode domain knowledge that the RNNs have to figure out from scratch by looking at raw sequences. That's a harder learning problem.

Third, and perhaps most importantly, electricity price forecasting is fundamentally about understanding how different variables interact—generation mix, weather, demand, time of day, etc. Tree-based models are really good at capturing these feature interactions in tabular data. RNNs, on the other hand, are optimized for pure sequential patterns. There's probably just not enough "sequence magic" here to justify the added complexity.

Finally, there's the cost-benefit angle. Even if RNNs matched the performance of SVR or Random Forest, would the extra computational expense be worth it? Probably not for an operational system.

4.3.3 What the predictions look like

When we plot the predictions, both LSTM and GRU track the general trends reasonably well. However, they tend to smooth things out—predictions are less volatile than actual prices, and extreme spikes often get underestimated. This is pretty typical neural network behavior.

4.3.4 Final thoughts

For this particular forecasting task, traditional machine learning simply works better. It's more accurate, much faster to train, and easier to deploy.

That said, deep learning might be worth revisiting if circumstances change:

- Much larger datasets (say, 10+ years of data across multiple markets)
- Evidence of complex long-term dependencies that simpler models can't capture
- Access to GPUs for efficient training

Future experiments could look at attention-based models (which have been successful in other time series domains), hybrid architectures that combine different network types, or even transfer learning from models pre-trained on similar forecasting problems.

5. Model Selection

After testing all these different approaches, we needed to actually pick one for deployment. This wasn't simply a matter of choosing whichever had the lowest MAE—real-world constraints matter too.

5.1 What we evaluated

We looked at five things when comparing models:

1. The numbers — MAE, RMSE, MAPE on the test set
2. Consistency — Does it perform similarly across different validation periods, or is it unstable?
3. Practical constraints — Training time, computational requirements, deployment complexity
4. Explainability — Can we justify predictions to stakeholders when they ask questions?
5. Robustness — Does it work equally well in different seasons and market conditions?

All of these mattered. A model that's marginally better but takes forever to train isn't necessarily the right choice. Similarly, a model nobody trusts won't actually get used, regardless of its accuracy.

5.2 The results

Here's how the best model from each category performed:

Category	Model	MAE	RMSE	MAPE (%)
Baseline	Seasonal Naïve	7.01	10.2	18.5
Statistical	SARIMAX	2.6	4.2	7.0
Machine Learning	SVR	1.73	3.1	4.8
Deep Learning	GRU	3.1	4.9	8.3

SVR came out ahead, and not by a small margin. We're looking at a 75% reduction in error compared to the baseline. Even against SARIMAX, which was pretty solid, SVR reduced MAE by about 35%.

Random Forest was actually very close—just slightly behind SVR. XGBoost and LightGBM were also competitive. But SVR had a small edge and, importantly, showed the most consistent behavior during validation.

5.3 Why SVR?

Let's go through each evaluation criterion:

5.3.1 Performance

SVR achieved the best scores: MAE of 1.73 and RMSE of 3.1. In practical terms, predictions are now over 5 euros closer to actual prices on average. For electricity trading, that matters.

The 4.8% MAPE is also reasonable—it means our percentage errors are fairly low across the full price range, not just for typical values.

5.3.2 Stability

During cross-validation, SVR performed consistently across all three time folds. We didn't see wild swings or signs that it was memorizing specific patterns that didn't generalize.

The hyperparameter search found good values (C, epsilon, kernel settings) that worked well on held-out data. This suggests the model learned actual relationships rather than overfitting to noise.

5.3.3 Computational cost

SVR takes a few minutes to train on our dataset. It's slower than Linear Regression but much faster than training neural networks.

For our use case—updating forecasts daily—this is fine. We're not trying to retrain every minute or handle massive datasets. Inference is quick enough for real-time forecasting.

If speed became critical, we'd probably look at ensemble methods instead. But for now, SVR fits our needs.

5.3.4 Interpretability

This is SVR's weakness. It uses kernel transformations that make it essentially a black box. We can't easily point to features and explain their contribution to specific predictions.

Tree-based models have feature importance. SARIMAX has interpretable coefficients. Linear Regression is completely transparent. SVR? Not really.

That said, we can verify it behaves reasonably by examining prediction patterns and residuals. For an operational system, stakeholders mostly care whether forecasts are accurate. If full explainability becomes a hard requirement (regulatory compliance, for example), we could switch to SARIMAX or Random Forest.

5.3.5 Prediction quality over time

We checked how SVR performed across the entire 2018 test year, looking at different seasons and market conditions.

Performance was fairly consistent seasonally. Winter, summer, spring, autumn—MAE varied a bit but nothing concerning. The model wasn't systematically worse during any particular period.

Residual analysis looked good. They were roughly normally distributed, centered near zero, with no obvious patterns left over. The Q-Q plot showed residuals tracking the theoretical normal distribution reasonably well.

Extreme price spikes weren't perfectly captured (no model nailed them), but SVR handled them better than most alternatives. Some underestimation during peaks is acceptable given the overall accuracy.

5.4 What we're not choosing

Worth discussing the alternatives:

SARIMAX would be the fallback if we needed full interpretability. It has a solid theoretical foundation, interpretable parameters, and still performs well (MAE of 2.6). If stakeholders demanded complete transparency, we'd go with this.

Random Forest was extremely close in accuracy and provides feature importance scores. If understanding which variables matter most becomes important, this would be a strong choice. It's also a bit faster to train.

XGBoost and LightGBM offer similar accuracy with better computational efficiency. If training time became a bottleneck, these would be worth considering.

Deep Learning didn't work well here. Two years of hourly data just isn't enough for LSTM/GRU to really shine. With much more data (say, a decade), they might become competitive. But for now, they're more trouble than they're worth.

5.5 Deployment considerations

If we actually deploy this, here's roughly what it would look like:

Set up an automated pipeline that:

- Fetches new data each day (prices, load, generation, weather)
- Computes all the engineered features (lags, rolling stats, time indicators)
- Retrains the SVR model on updated historical data
- Generates the next 24 hours of price forecasts
- Monitors performance and alerts if accuracy degrades

The feature engineering is probably riskier than the model itself. Getting lags aligned correctly, handling occasional missing data, making sure features match what the model expects—that's where bugs tend to appear. We'd need good validation checks.

Performance monitoring would track MAE over rolling 7-day and 30-day windows, comparing against the baseline and against training validation metrics. If error starts creeping up, that signals something might be changing in the market.

5.6 Potential problems

Being realistic about what could go wrong:

Market changes: Training used 2016-2017 data. If the electricity market fundamentally shifts—major regulatory changes, huge renewable buildout, market restructuring—performance could suffer. All models face this risk, not just SVR. Regular monitoring and periodic retraining helps.

Rare extremes: Very unusual events (like simultaneous plant failures during peak demand) will probably be underestimated. The model hasn't seen many of these during training. Human oversight might be needed for such cases.

Data quality: SVR relies on having good data for all input features. If weather forecasts are off, or if generation data gets delayed or revised, forecast quality drops. This is more of an infrastructure issue than a model issue.

Overfitting to recent patterns: Daily retraining could lead to overweighting recent short-term patterns that don't generalize. We'd need to think about how much emphasis to place on recent versus historical observations.

5.7 Final call

We're going with **SVR** for operational deployment.

It's the most accurate model we tested—75% better than baseline, 35% better than the best statistical alternative. Validation shows good stability and generalization. Computational requirements are manageable. And while it's not perfectly interpretable, we can verify reasonable behavior through standard diagnostic checks.

For a production forecasting system with our current data and constraints, SVR is the best choice. If things change—if explainability becomes non-negotiable, or if we get access to much larger datasets—we can reassess. But right now, based on what we know, SVR wins.