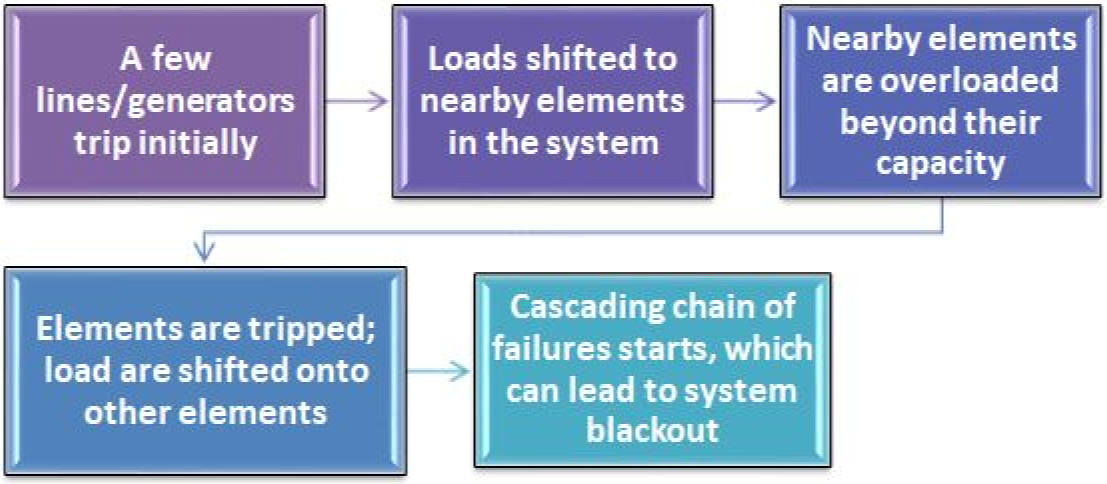
Title: Predicting Cascading Failures in Smart Grids

Abstract:

Although there has been significant progress in modeling cascading failures in smart grids, few works involved using machine learning algorithms. In this project, we classify cascading failures in the smart grid that lead to large blackouts in smart grids using machine learning algorithms. Since real-world cascading failure data is not available, we create a synthetic cascading failure simulator framework to generate cascading-failure data for various smart grid operating parameters. The features include smart grid variables such as the number of failed lines, maximum/minimum capacity of the failed lines, loading level, capability of load shedding, uncertainty over communication (capacity estimation error). We include the topological parameters such as the average degree, the average distance for various combinations of transmission line failures in our data set. We also include the probability of human error, which worsens the risk of a cascade during cascade propagation. Then we apply various machine learning algorithms to classify the cascading effect, which is formed as a function of the number of failed transmission lines and the amount of load shedding, and compare the accuracy of the models. Further, we use regression models to predict the cascading effect. This data-driven technique is useful to quickly classify cascading failures based on the input smart grid conditions, and hence smart grid design engineers can use this to increase the robustness of the grid.

Introduction:

Smart grids are complex systems where the demand-supply chain has to be maintained always between the generation of power, and the demand for power from the users to operate smart grids without interruptions. Although protective measures are in place, smart grids are prone to cascading failures. Cascading failures in the smart grids are scenarios where a small disturbance in the smart grid can create a domino effect due to an imbalance sudden between demand-supply. The steps leading to a cascading failure in smart grids is shown in Figure 1. The number of people getting affected by these events and the economic loss is astronomical. For example, the cascading failures in 2003 in the North-East affected more than 55 million people and the economic impact was in the billions. From 1965 to 2008, there were nine massive blackout events affecting more than 20 million people whereas in the last decade there were seven massive blackouts including the largest one in India [1]. Several modeling efforts are ongoing to model and mitigate cascading failures. I, myself have been using the Markov chain Monte Carlo approach to model cascading failures analytically, which is a different scope of work. During the literature review for my research, I observed that although the prediction of the cascading failure, identification of critical system behaviors that leads to cascading failures can be invaluable, very few machine learning efforts have been available. Digging deep, I found that the rationale for this is the unavailability of enough real-world cascading failures data to make an inference model using machine learning. As a part of my machine learning class during the spring of 2019, we developed a cascading failure simulator framework to mimic real-world cascading failure events based on simulations over a portion of the US smart grid. Using the simulator, we generated a labeled cascading failures dataset and did a project which also resulted in a conference paper [2]. However, the main motivation of that project was to develop a cascading failure simulator to generate a synthetic cascading failure dataset so that machine learning can be applied for cascading failure. The work in [2] misses critical components of a machine learning project such an exploratory data analysis, data manipulation, feature engineering, and pruning. Also, the work did not use key machine learning concepts like hyperparameter tuning, regularizations, and Neural networks. On top of that, we overhauled MATLAB codes to incorporate the effect of human operators on cascading failure. These factors


A few lines/generators trip initially
Loads shifted to nearby elements in the system
Nearby elements are overloaded beyond their capacity
Elements are tripped; load are shifted onto other elements
Cascading chain of failures starts, which can lead to system blackout

,

Fig.1: Flow-chart of cascading failures

motivated me to work on predicting cascading failures in smart grids using Machine learning algorithms a capstone project.

Summary of the project objective:

This project is a domain-specific research project and I will answer the following problems:

- First, perform exploratory data analysis to find the patterns in data, for example observing the role of various features like load-shedding, human error, etc. on cascading failure.
- Using regression techniques to find the effect of cascading failures due to various initiating disturbance conditions when cascading ends.
- Use classification techniques to identify the critical (leads to cascading failure) and non-critical (does not lead to cascading failure) initiating feature values.
- Do a comparison between various machine learning algorithms, do hyperparameter tuning where necessary.

Dataset:

We have developed the cascading failures simulated dataset using my Cascading failure simulation (CFS) framework developed in Matlab. It has more than 70000 simulations of cascading failures. The dataset contains 17 features and 2 target variables (total failed lines, Loadshed).

MATPOWER [3], a package of MATLAB m-files for solving the steady-state DC or AC power flow optimization problem is used in this project. It uses the power-flow distribution framework under the given set of constraints. The standard power flow or load flow problem involves solving for the set of voltages and flows in a power grid network corresponding to a specified pattern of load and generation [4]. MATPOWER includes solvers for both AC and DC power flow problems, both of which involve solving a set of linear equations. The AC power flow captures detailed dynamics of cascading failures in the power grid, including the transient effects. In this project, we generate our data set using the DC power flow because of its simplicity yet effectiveness.

The flowchart in Figure 2 illustrates the CFS framework used in this work. We start with a few (between [2,10] in this work) initial numbers of transmission-line failures in the power grid initiated from an arbitrary initial event. It is important to note that the failure of at least two transmission lines is necessary to start a cascading failure event because the power grid is robust against one transmission line failure due to N-1 security considerations. Our objective is to

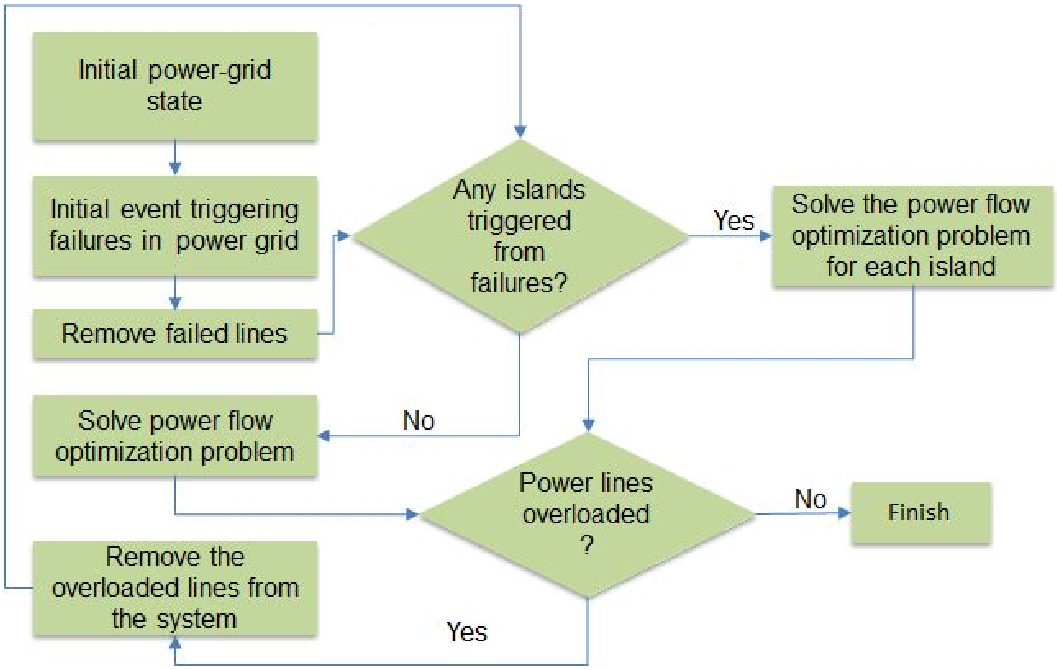

Initial power-grid state
Initial event triggering failures in power grid
Remove failed lines
Any islands triggered from failures?
Yes
Solve the power flow optimization problem for each island
No
Solve power flow optimization problem
Power lines overloaded ?
No
Finish
Remove the overloaded lines from the system
Yes

Fig.2: Flow-chart of the cascading failures simulation framework

classify the cascading failures in power grids into no, small, and large as well as to identify the initial conditions that trigger large transmission line failures. We assume that we have sufficient knowledge regarding the power grid topology and operating parameters before the initial failure event. We then remove the failed transmission lines from the grid and check whether any islands are formed in the power grid. Note that an island is a self-sufficient local network that operates independently when disconnected from the base network having a set of generators and loads [5]. Depending on whether any islands are formed or not, we then solve the DC power flow using MATPOWER on each island. If we have any overloaded lines in the grid, we either fail these lines or probabilistically fail a set of lines among them (e.g., failing the line with the highest overload). In this project, we use a similar approach used in [4] for islanding and overloading calculations and propose two algorithms (discussed later) for calculating islanding and overloading in power grids. We repeat the same process until we end up with a power grid with no overloaded lines, which indicates the end of a cascade.

Model features

Based on smart simulations and prior works we identify the following power-grid operating parameters and features that govern the cascading failure dynamics. In our simulation, we use the IEEE 118-bus system (which is a simple approximation of the American Electric Power system in the U.S. Midwest, as the test case which contains 186 transmission lines, 118 buses (nodes) and 54 generators.

Power grid loading level, *r*: The power grid loading level, $r \epsilon [0,1]$, is the ratio of the total load demand and the generation capacity of the power grid. In the IEEE 118-bus system, the maximum generation is 9966 MW. Here r=1 indicates the demand is also 9966 MW and r scales the power demand with respect to maximum possible generation.

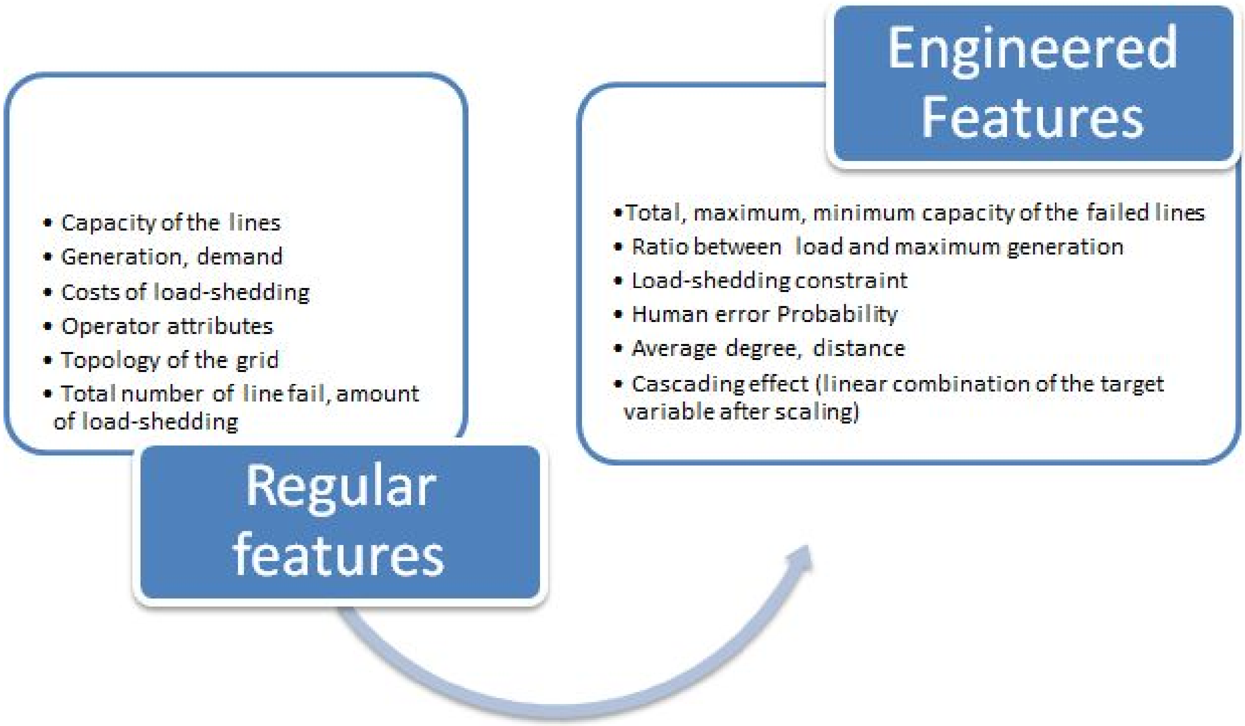
Load shedding constraint, *θ*: The load shedding constraint is the ratio of uncontrollable loads (the loads that do not participate in load shedding) and the total load in the power grid, and it is denoted by $\theta \epsilon [0,1]$. The parameter θ ensures the capability of implementing the control actions by the power grid operator. Namely, θ =1 indicates that all the loads are uncontrollable, and the operators can perform no load shedding. Again, θ =0 indicates that the operators can shed any load on the grid. In this project, we consider equal load shedding constraints over all the loads in the grid for simplicity.

Capacity estimation error, *e* ϵ [0,0.5], is the error by the control center in its estimation of the actual capacity of the lines. In our CFS framework, this parameter is used to calculate overloaded lines. We used the same approach used to calculate overloaded lines. When power flow in a transmission line exceeds (1-*e*)*capacity, we consider that line as an overloaded line. We estimate the capacity of a transmission line using power flow simulation with maximum loads, i.e. when generation equals demand (r=1). Since we use DC power flow simulation, there are no transient effects, and we can use maximum generation without any issues. We quantize the flow capacity of a transmission line into a set of five capacities {20, 80, 200, 500, 800} MW [5] and assign this capacity of the transmission line as a constraint of the MATPOWER power flow optimization problem (discussed later).

Failed lines, Cmax, Cmin, Installed capacity: We keep track of the initially failed lines, the maximum, minimum, and cumulative capacity of the initially failed lines as features of the model

Average degree, average distance: We track the average degree and distance of the network after removing the initially failed lines as topological features of the grid.

Here it is worth mentioning that most of the features described above are engineered features calculated from the regular features of the smart grid.


Regular features
• Capacity of the lines
• Generation, demand
• Costs of load-shedding
• Operator attributes
• Topology of the grid
• Total number of line fail, amount of load-shedding
Engineered Features
•Total, maximum, minimum capacity of the failed lines
• Ratio between load and maximum generation
• Load-shedding constraint
• Human error Probability
• Average degree, distance
• Cascading effect (linear combination of the target variable after scaling)

Human operator error probability: We use our work in [6] to calculate the human operator error probability, randomly drawn from the distribution of the operator attributes as a feature for the model.

Number of failed lines due to cascade: We track the number of failed transmission lines due to cascade after the cascade ends as an output label.

Amount of load shed: We use the optimal power flow algorithm from MATPOWER, which includes the capability of implementing load shedding depending on the cost. Here, we set the cost of load shedding ten times higher than the cost of generation to ensure maximum generation before any load shedding. We track the cumulative amount of load shedding as a critical grid parameter.

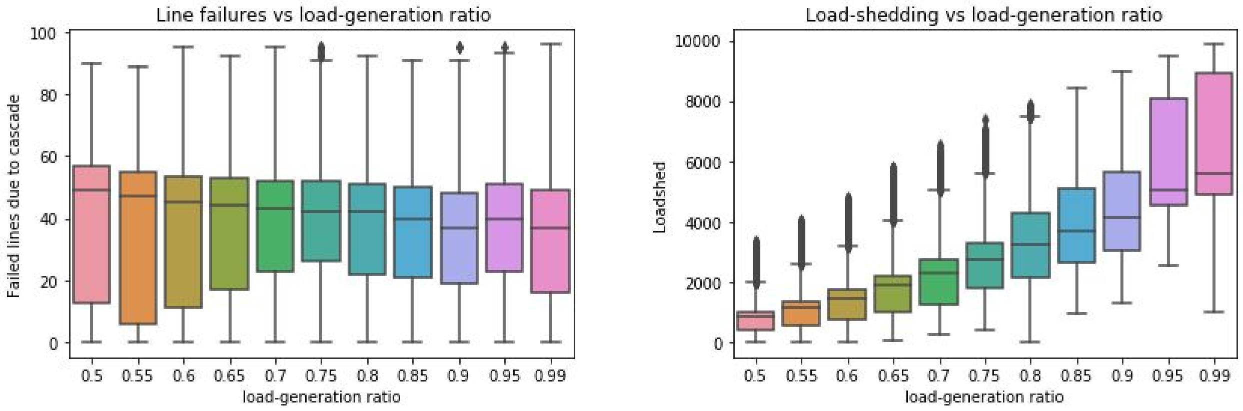


Fig. 3: box plot of the output variables with different *r*

From Figure 3 we can see that the number of failed lines almost remains the same with load increase however the amount of loadshed increases significantly, which indicates that the additional load demand is mostly shaded and fewer line failures are triggered due to this additional load demand. To capture the effect of cascading failures (triggered by file failures and loadshed) we use the following target variable.

Cascading effect: We take the linear combination of the number of failed lines and the amount of load-shed as the output variable we want to predict. The variable is scaled between [0,1]. We use the variable values directly for the regression task. For the classification task, we first calculate the median of the cascading effect and sue the left half of the median as no cascade (class zero) and the right half as cascade (class one) for classification

Data cleaning:

The dataset was mostly clean but we checked the following steps to ensure the cleanliness of the data.

- there are no missing values, null values, outliers in the dataset.
- We removed one duplicate column
- We renamed the columns for better understanding

We further checked the dataset with pandas info and describe the method and found everything consistent.

Statistical Analysis:

We did statistical analysis such as colinearity analysis from pruning features, correlation analysis, histogram analysis for understanding the behavior of the dataset.

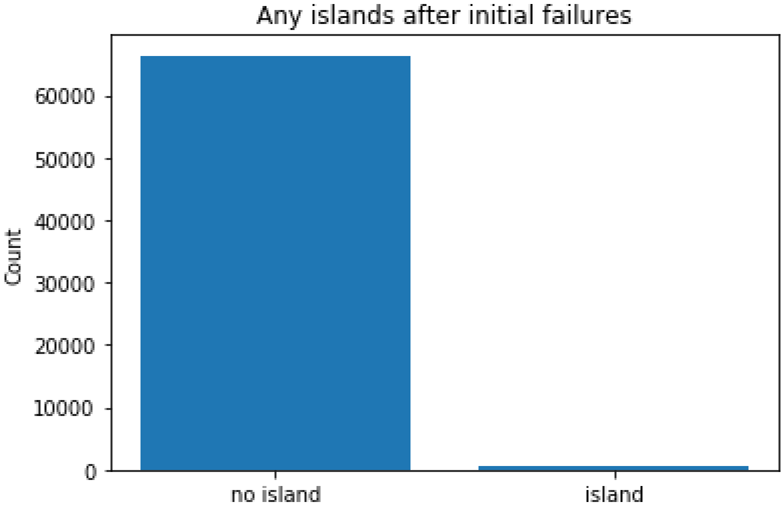


Fig. 4: Island

In figure 4 we show the number of islands triggered after the initial failed lines of the grid. Since the number of islands triggered is very few, we did not consider it as a feature.

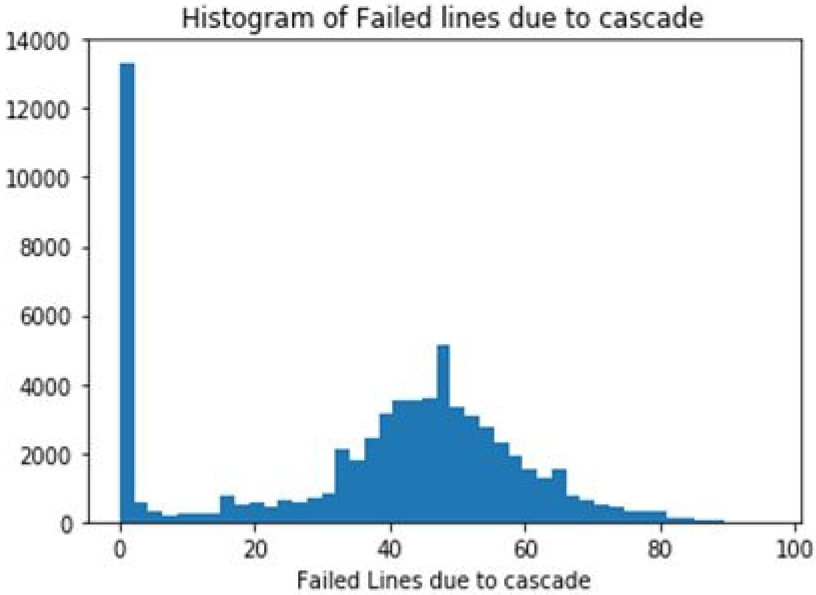


Fig. 5: Histogram of the failed lines due to cascade

From the histogram of the cascading effect in Figure 5, It can be observed that the histogram is bimodal. The first pick indicates a zone where no additional transmission liens were failed due to cascade, and no loads were shed, i.e., for the set of feature values, no cascading occurred. Similarly, the second pick represents that the average cascading effect occurred at 0.35.

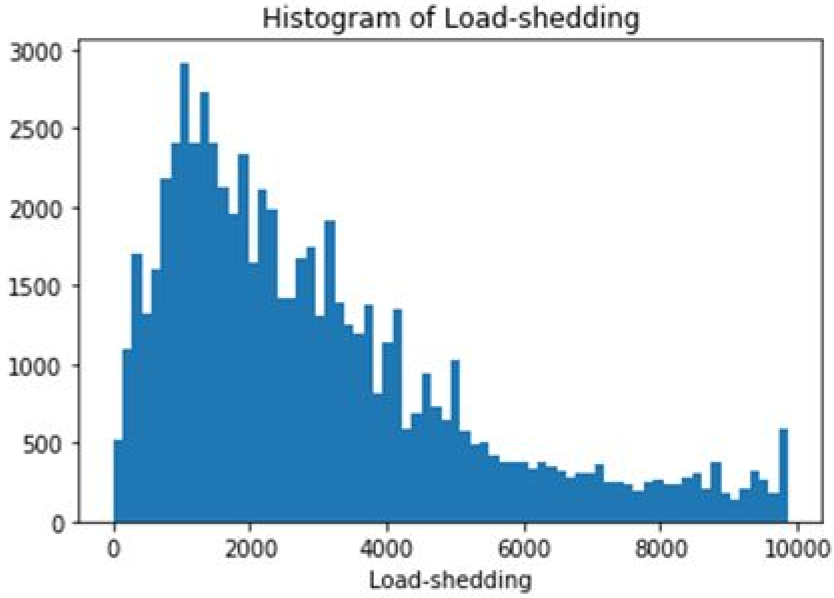


Fig. 6: Histogram of the load-shedding

The histogram of load-shedding in Figure 6 is slightly skewed to the left, which is intuitive. This indicates the probability of a large cascading failure occurring is less. This also indicates that matpower optimal power flow (OPF) is curtailing the loads efficiently to minimize the greater risk of a cascading failure.

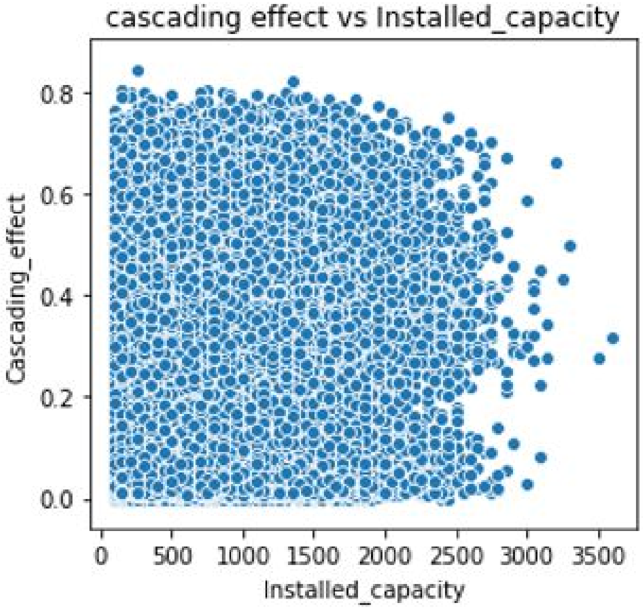


Fig. 7: Cascading effect vs installed capacity

From figure 7, we can visualize that there is no pattern of correlation between the cascading effect and installed (cumulative) capacity. Also, the installed capacity is not a categorical value. Considering this we remove this feature from the dataset.

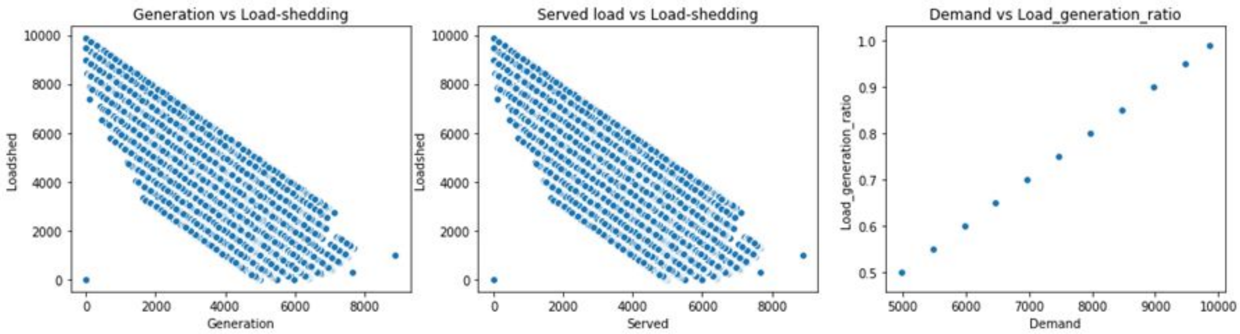


Fig. 8: generation and served load vs load shedding, demand vs *r*

We did not consider generation, served load and load demand as features of the dataset to avoid colinearity which can be visualized in figure 8.

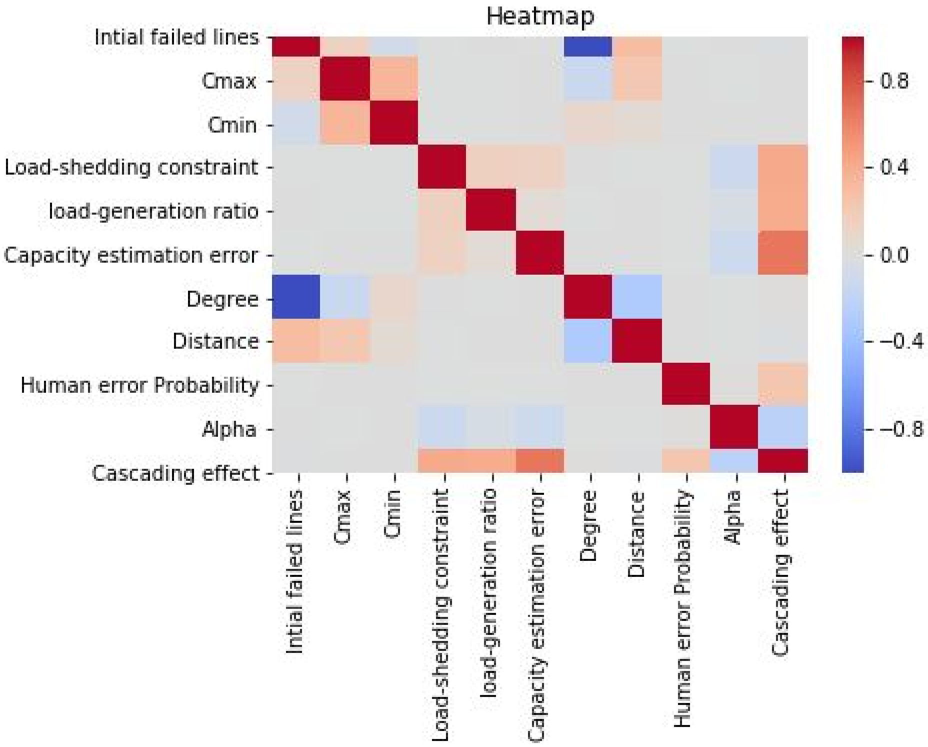


Fig. 9: Correlation between features

We plot the correlation among features in Figure 9. We can observe a strong correlation between the cascading effect and Capacity estimation error. A moderate correlation between cascading effect and load-generation ratio, load-shedding constraint, Human error probability, alpha(negative correlation). Low/minimal Correlation between cascading effect and Cmax, Cmin, Degree, distance, and initially failed lines/

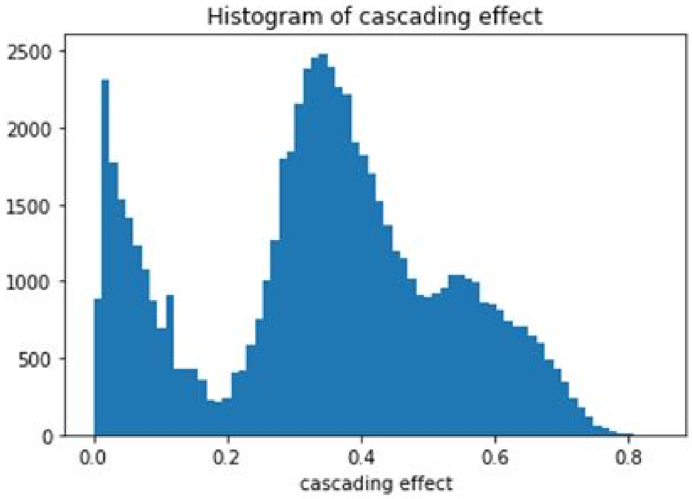


Fig. 10: Histogram of the cascading effect

From Figure 10, the histogram of the cascading effect shows bimodal nature. The first peak is due to no line failure scenarios and the second peak captures the average effect of line failures and load shedding.

Results and In-depth analysis using machine learning:

We have used the following algorithms for regression:

- Linear regression/ Ridge/Lasso regression
- Random Forest regression
- Support vector regressor

We have used the following algorithms for classification:

- Logistic regression
- KNN (k nearest neighbor)
- Random forest
- Decision tree
- Support vector machine

- Adaboost

The following metrics are used for evaluating the regression performance:

- r-squared score.
- mean absolute error
- mean square error

The following metrics are used for evaluating the classification performance:

- Accuracy
- Precision
- Recall
- F1-score

We split the data set for training (80%) and testing (20%) purposes. The precision, recall, and f1-scores are calculated using scikit-learn for all the algorithms and shown in Fig. 11. It can be observed that all the classification algorithms have a relatively higher accuracy of classification with random forest having the best accuracy. Next, we show the individual cascade class classification accuracies in Fig. 12. The purpose of the classification task here is to find the boundary between the cascade and no cascade zones for the given input space. Here also we can see that random forest works best.

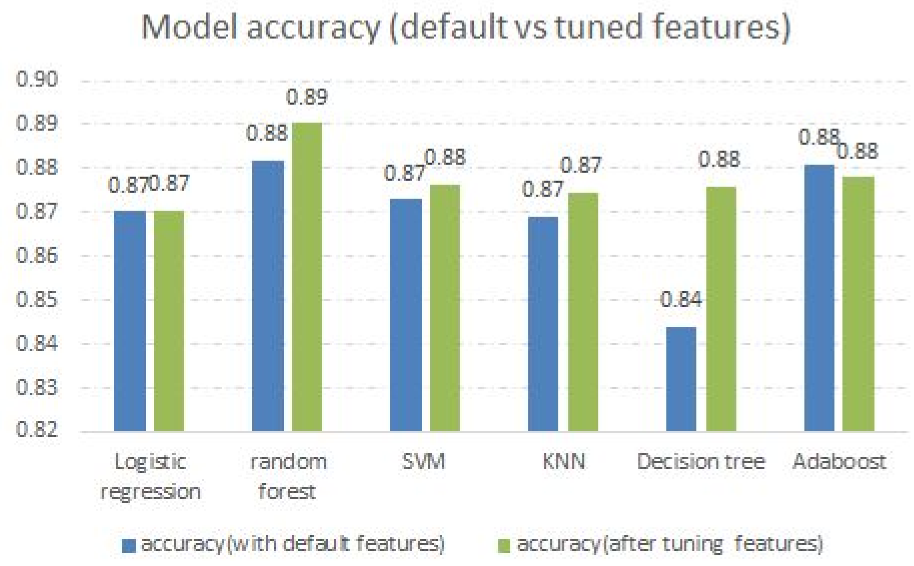


Fig. 11: Classification accuracy

The hyperparameters used to get the best accuracy, precision and recall are given in Table 1.

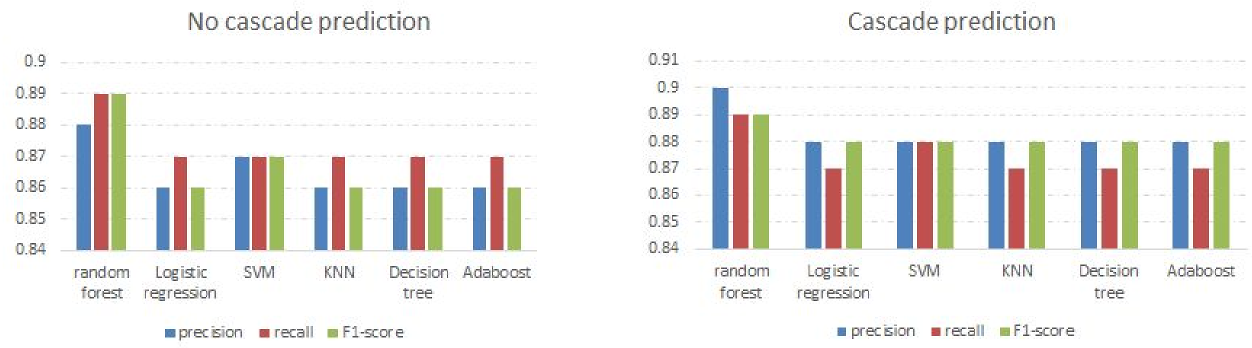


Fig. 12: Classification accuracy for Nocascade/cascade prediction

Table 1	
Model	Hyperparameter
Decision tree	'criterion': 'entropy’, 'min_samples_leaf': 10, 'min_samples_split': 5
KNN	'algorithm': 'auto’,'leaf_size': 1, 'n_neighbors': 10, 'weights': 'distance'
Adaboost	'algorithm': 'SAMME.R', 'learning_rate': 0.5, 'n_estimators': 200
Random forest	'criterion': 'gini', 'min_samples_leaf': 5, 'min_samples_split': 5, 'n_estimators': 50
Logistic regression	C= 10 (penalty =‘l2’)
SVM	'C’: 5, 'kernel': ['rbf']

For the regression task, we also obtained the best r-squared error using random forest regression which is shown in Figure 13. The mean squared error is also reported

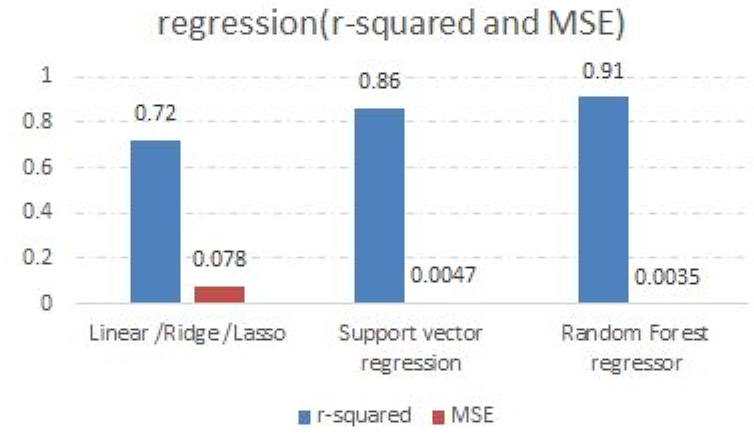


Fig. 13: cascading effect regression

Figure 14 represents the trend of the linear trend between predicted vs test values.

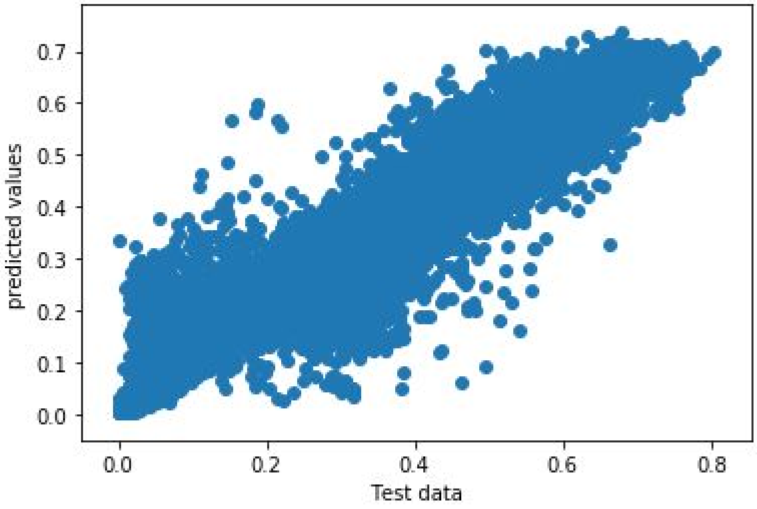


Fig. 14: prediction and test data

Thus we select a random forest model for regression and classification tasks. Finally, we have generated a new 10000 data for testing the model and we have achieved an 89% accuracy for classification and an r-squared score of 0.90.

Discussion:

In this work, we classify and predict cascading failure based on critical power grid attributes like power-flow capacity, average degree, distance, power grid loading, estimation errors, constraints on load-shedding and so forth. The contribution of this work is three-fold. First, we develop a cascading failure framework (CFS) using MATPOWER, a widely used power-flow simulator and generate synthetic cascading failure data using the IEEE 118-bus topology. Using an earlier developed CFS framework, we have effectively generated a labeled cascading failure data set, which is used as an input to the machine learning models. Second, a comparison using different classifiers is shown to evaluate the classification performances. The objective is to do exploratory data analysis on labeled data using various supervised machine learning algorithms and identify the best algorithm based on accuracy. Third, we use a linear regression technique to calculate the level of cascading effect for any given initial condition. The results suggest that cascading failure prediction can be made using machine learning with high accuracy. This data-driven technique can be used to generate cascading failure data set and the power-grid engineers can use this approach for cascade data generation and hence predicting vulnerabilities and enhancing the reliability and robustness of the grid.

References:

1. https://en.wikipedia.org/wiki/List_of_major_power_outages
2. Rezoan A. Shuvro, Mitun Talukder, Pankaz Das, Majeed M. Hayat, “Predicting Cascading Failures in smart grids using Machine Learning Algorithms”, North American Power Symposium (NAPS’2019)
3. Zimmerman, R. D., Murillo-Sánchez, C. E., & Thomas, R. J. (2010). MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on power systems*, *26*(1), 12-19.
4. Rahnamay-Naeini, M., Wang, Z., Ghani, N., Mammoli, A., & Hayat, M. M. (2014). Stochastic analysis of cascading-failure dynamics in power grids. *IEEE Transactions on Power Systems*, *29*(4), 1767-1779.
5. Wood, A. J., Wollenberg, B. F., & Sheblé, G. B. (2013). *Power generation, operation, and control*. John Wiley & Sons.

6. Shuvro, R. A., Das, P., Abreu, J. M., & Hayat, M. M. Correlating Grid-operators' Performance with Cascading Failures in Smart-Grids. In *2019 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)* (pp. 1-5). IEEE.