Electric Grid Stability Analysis

William G. Stalnaker (wgstalnaker)

June 21, 2020

Introduction

Today's electrical grid, which the majority of businesses and residents connect to, is experiencing a new market disruption. The disruptive technology of distributed energy generation, such as residential solar, has introduced new factors impacting grid stability. The grid stability is not immune to other disruptions, such as aging infrastructure, cyber security, and natural disasters. Each of these disruptions have a direct impact to the supply and demand economics of the electrical grid as a whole.

To reduce the impact, decentralizing the grid has been proposed. Arzamasov, Böhm and Jochem, of the Karlsruhe Institute of Technology in Germany suggest (2018), "Decentralize Smart Grid Control (DSGC) is a new system implementing demand response without significant changes of the infrastructure." Analysis of the factors and impacts associated with implementing a DSGC system are outside of the scope of this paper. This paper will focus on implementation of machine learning models to predict stability of the grid, utilizing the data set Arzamasov created (2018). This data set is a simulation of a four-node star electrical grid with centralized production. The data set includes 14 total attributes - 11 predictive attributes, one non-predictive (p1), and two goal fields, as Arzamasov details below (2018):

- tau[x]: reaction time of participant (real from the range [0.5,10]s). Tau1 - the value for electricity producer.
- p[x]: nominal power consumed(negative)/produced(positive)(real). For consumers from the range [-0.5,-2]sˆ-2; p1 = abs(p2 + p3 + p4)
- g[x]: coefficient (gamma) proportional to price elasticity (real from the range [0.05,1]sˆ-1). g1 - the value for electricity producer.
- stab: the maximal real part of the characteristic equation root (if positive - the system is linearly unstable)(real)
- stabf: the stability label of the system (categorical: stable/unstable)

The Electrical Grid Stability Simulated Data data set was split 80/20. According to the random creation process Vadim Arzamasov described (2018), this allowed for the test set to be large enough to be a satisfactory representation of the data set as a whole and return statistically meaningful results. The caret package was leveraged heavily for modeling. Model algorithms were selected for the tasks of performing classification and regression as well as those referenced in *Towards Concise Models of Grid Stability* (Arzamasov, Böhm, and Jochem, 2018). The selected algorithms were:

- Classification and Regression Trees
- k-Nearest Neighbors
- Naive Bayes
- Random Forest
- Support Vector Machines with Radial Basis Function Kernel

These algorithms were executed using the caret package's train function while applying 10 fold cross validation. The model with the highest accuracy in predicting stabf was selected for making final predictions. Support Vector Machines with Radial Basis Function Kernel, with an accuracy of 95%, will prove to be the best algorithm for predicting grid stability.

Methodology

Data Ingestion

The Electrical Grid Stability Simulated data set used for this project is held in the UCI Machine Learning Repository (2019). This copy is stored in a comma delimited format which requires data types to be mutated to usable formats. Attributes 1-13 are converted to numeric, and the 14th attribute is converted to a factor. During data ingestion, the training and testing data sets are created and represent an 80/20 split of the original data set. As

previously mentioned, this constitutes a test set large enough to be a satisfactory representation of the data set as a whole and sufficient to return statistically meaningful results. To avoid unanticipated interference during training and testing, non-predictor attributes were removed from the data set.

Data Structure

```
## 'data.frame':    10000 obs. of  14 variables:
##  $ tau1 : num  2.959 9.304 8.972 0.716 3.134 ...
##  $ tau2 : num  3.08 4.9 8.85 7.67 7.61 ...
##  $ tau3 : num  8.38 3.05 3.05 4.49 4.94 ...
##  $ tau4 : num  9.78 1.37 1.21 2.34 9.86 ...
##  $ p1   : num  3.76 5.07 3.41 3.96 3.53 ...
##  $ p2   : num  -0.783 -1.94 -1.207 -1.027 -1.126 ...
##  $ p3   : num  -1.26 -1.87 -1.28 -1.94 -1.85 ...
##  $ p4   : num  -1.723 -1.255 -0.92 -0.997 -0.554 ...
##  $ g1   : num  0.65 0.413 0.163 0.446 0.797 ...
##  $ g2   : num  0.86 0.862 0.767 0.977 0.455 ...
##  $ g3   : num  0.887 0.562 0.839 0.929 0.657 ...
##  $ g4   : num  0.958 0.782 0.11 0.363 0.821 ...
##  $ stab : num  0.05535 -0.00596 0.00347 0.02887 0.04986 ...
##  $ stabf: Factor w/ 2 levels "stable","unstable": 2 1 2 2 2 1 2 2 1 2 ...
```

The data is tidy, and the ingestion script has successfully transformed the data to the correct format for this analysis.

Null Values Check

```
##  tau1  tau2  tau3  tau4    p1    p2    p3    p4    g1    g2    g3    g4  stab
## FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## stabf
## FALSE
```

No null values to be handled.

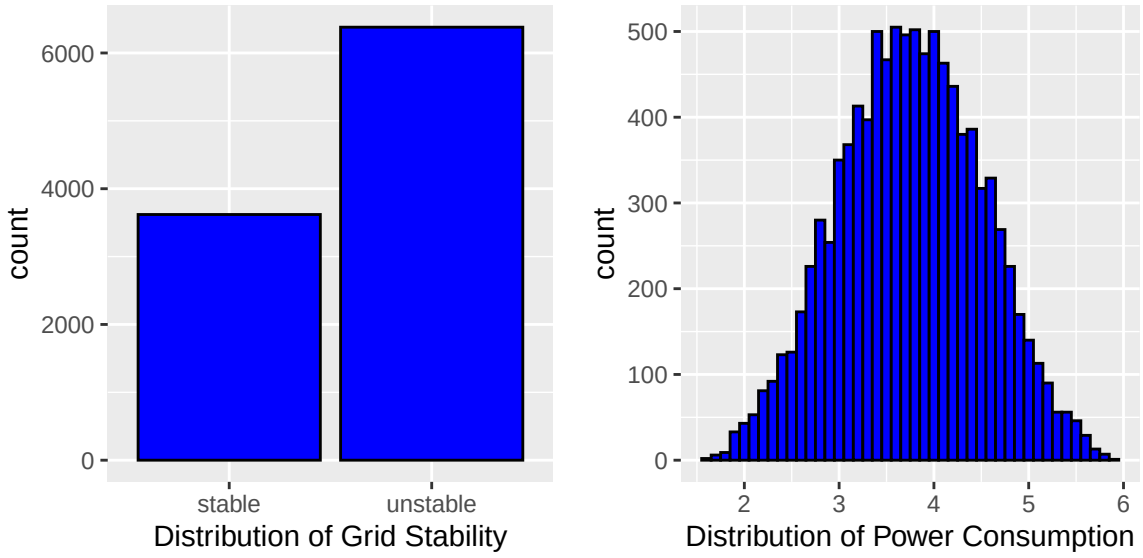
Data Summary

```
##       tau1             tau2              tau3             tau4
##  Min.   :0.5008   Min.   : 0.5001   Min.   :0.5008   Min.   :0.5005
##  1st Qu.:2.8749   1st Qu.: 2.8751   1st Qu.:2.8755   1st Qu.:2.8750
##  Median :5.2500   Median : 5.2500   Median :5.2500   Median :5.2497
##  Mean   :5.2500   Mean   : 5.2500   Mean   :5.2500   Mean   :5.2500
##  3rd Qu.:7.6247   3rd Qu.: 7.6249   3rd Qu.:7.6249   3rd Qu.:7.6248
##  Max.   :9.9995   Max.   : 9.9998   Max.   :9.9994   Max.   :9.9994
##        p1               p2                p3                p4
##  Min.   :1.583   Min.   :-1.9999   Min.   :-1.9999   Min.   :-1.9999
##  1st Qu.:3.218   1st Qu.:-1.6249   1st Qu.:-1.6250   1st Qu.:-1.6250
##  Median :3.751   Median :-1.2500   Median :-1.2500   Median :-1.2500
##  Mean   :3.750   Mean   :-1.2500   Mean   :-1.2500   Mean   :-1.2500
##  3rd Qu.:4.282   3rd Qu.:-0.8750   3rd Qu.:-0.8750   3rd Qu.:-0.8751
##  Max.   :5.864   Max.   :-0.5001   Max.   :-0.5001   Max.   :-0.5000
##        g1                g2                g3                g4
##  Min.   :0.05001   Min.   :0.05005   Min.   :0.05005   Min.   :0.05003
##  1st Qu.:0.28752   1st Qu.:0.28755   1st Qu.:0.28751   1st Qu.:0.28749
##  Median :0.52501   Median :0.52500   Median :0.52501   Median :0.52500
```

```
##  Mean   :0.52500   Mean   :0.52500   Mean   :0.52500   Mean   :0.52500
##  3rd Qu.:0.76243   3rd Qu.:0.76249   3rd Qu.:0.76244   3rd Qu.:0.76243
##  Max.   :0.99994   Max.   :0.99994   Max.   :0.99998   Max.   :0.99993
##       stab               stabf
##  Min.   :-0.08076   stable  :3620
##  1st Qu.:-0.01556   unstable:6380
##  Median : 0.01714
##  Mean   : 0.01573
##  3rd Qu.: 0.04488
##  Max.   : 0.10940
```

Figure 1

Illustration of grid stability and power consumption distributions

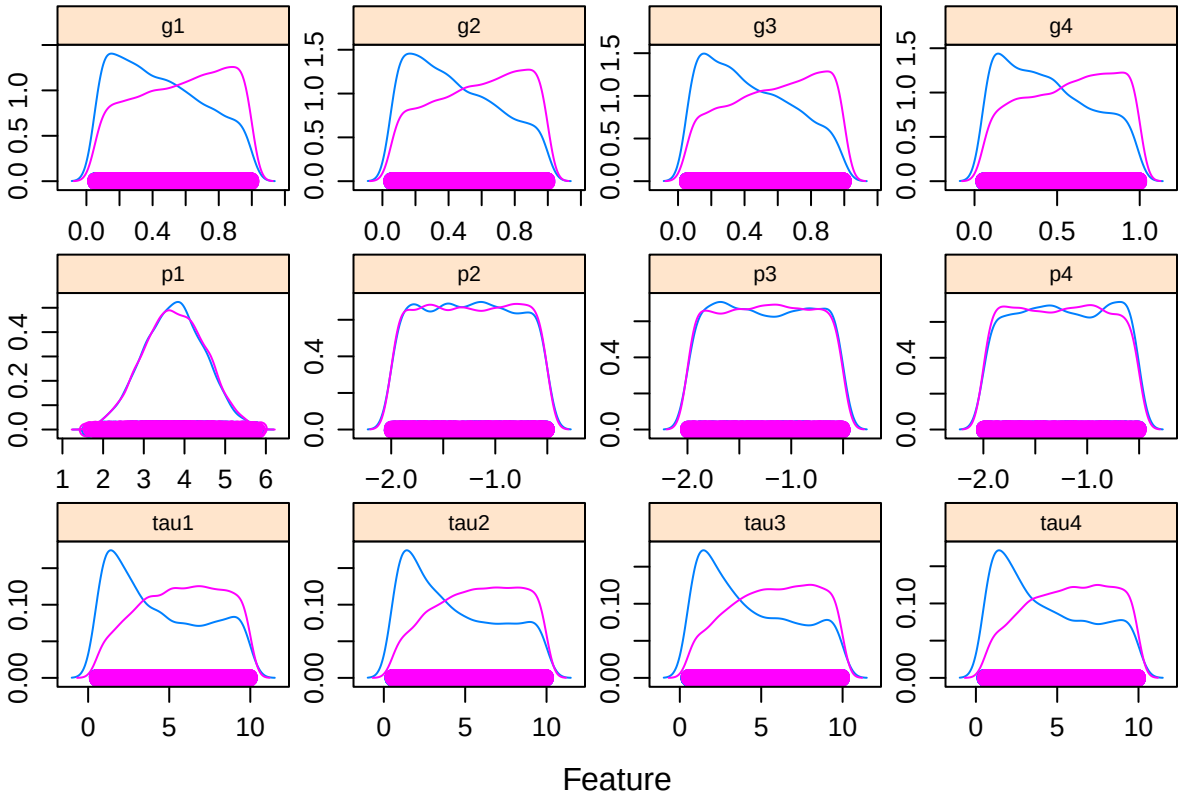


This data exploration shows that the data is tidy. There are no null values to be handled. The proportion of unstable is greater than the stable, representing 64% of the entire data set. Finally, there is a normal distribution of the Distribution of Power Consumption of the p1 attribute. The p1 was defined as representation of the consumers and the actual is calculated as p1 = abs(p2 + p3 + p4).

When visualizing the 11 predictors across the stabf goal attribute there is even distribution with very little variance. Could this be an indication that the data set is overtly normal or simply minimal variances in electrical grid supply and demand disruptions?

Figure 2

Illustration of the density of the stabf attribute within p1 and each of the predictor attributes



Modeling

The modeling approach used for this analysis was to leverage the caret package. The previous data exploration and the data set document on the UCI Repository (2019) show the following:

- Data is clean and normalized
- Data includes 10,000 rows of 14 attributes
- Data includes 11 predictors
- Data includes one non-predictive value (p1)
- Data includes two goal attributes (stab and stabf)
- Well-suited for Classification and Regression

Note: If further exploration is desired, additional information on this package may be found here https://cran.r-project.org/web/packages/caret/vignettes/caret.html

Given the data is well-suited for Classification and Regression, these model types will be utilized by leveraging the train function of the caret package. Kuhn (2019) instructs that the train function can be used to:

- evaluate, using resampling, the effect of model tuning parameters on performance
- choose the "optimal" model across these parameters
- estimate model performance from a training set

Training model evaluation will use 10-fold cross validation. Five different models will be evaluated, and the model that produces the greatest accuracy will be chosen to make the final prediction. Where appropriate, that model will be tuned for optimal accuracy.

```
# Run algorithms using 10-fold cross validation
fitControl <- trainControl(method="cv", number=10)
```

```r
# Goal is to find the most accurate model, the metric parameter
# will be used for this.
metric <- "Accuracy"
# Classification and Regression Trees (CART)
set.seed(825)
fit_cart <- train(stabf~., data=train,
                  method="rpart",
                  metric=metric,
                  trControl=fitControl)
# Classification and Regression Trees (CART) - Results
# fit_cart$bestTune
# k-Nearest Neighbors (kNN).
set.seed(825)
fit_knn <- train(stabf~., data=train,
                 method="knn",
                 metric=metric,
                 tuneGrid = data.frame(k = seq(2 , 50 , 2)),
                 trControl=fitControl)
# k-Nearest Neighbors (kNN) results
#fit_knn$bestTune
#plot(fit_knn)
#fit_knn$finalModel
# Naive Bayes (naive_bayes)
set.seed(825)
fit_nb <- train(stabf~., data=train,
                 method="naive_bayes",
                 metric=metric,
```

```
                 trControl=fitControl)
# Naive Bayes (naive_bayes) - Results
#fit_nb$finalModel
#  Random Forest (RF)
set.seed(825)
fit_rf <- train(stabf~., data=train,
                method="rf",
                metric=metric,
                tuneGrid = expand.grid(.mtry=c(1:5)),
                trControl=fitControl)
# Support Vector Machines (SVM) with a linear kernel
set.seed(825)
fit_svm <- train(stabf~., data=train,
                 method ="svmRadial",
                 metric = metric,
                 tuneGrid = expand.grid(sigma = c(.01, .50, .05),
                                        C = c(0.75, 0.9, 1, 1.1, 1.25)),
                 trControl = fitControl,
                 preProcess = c("center","scale"))
# Support Vector Machines (SVM) with a linear kernel - Results
#fit_svm$bestTune
#plot(fit_svm)
#fit_svm$finalModel
# summarize accuracy of models
results <- resamples(list(cart=fit_cart,
                          nb=fit_nb,
                          knn=fit_knn,
```

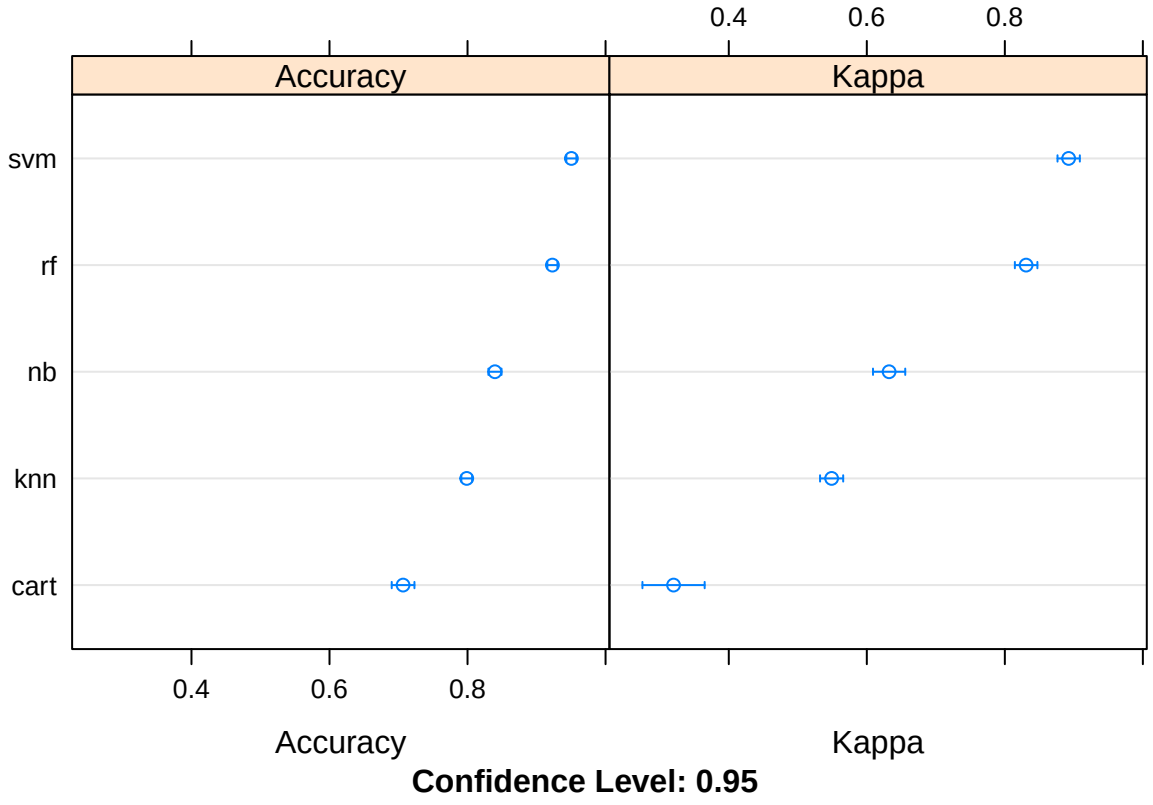
```
                        rf=fit_rf,
                        svm=fit_svm))
```

Results

The accuracy of the five models is shown in the following plot. Support Vector Machines with Radial Basis Function Kernel achieved the highest accuracy on the training data set and will be used for the final prediction using the test data set.

Figure 3

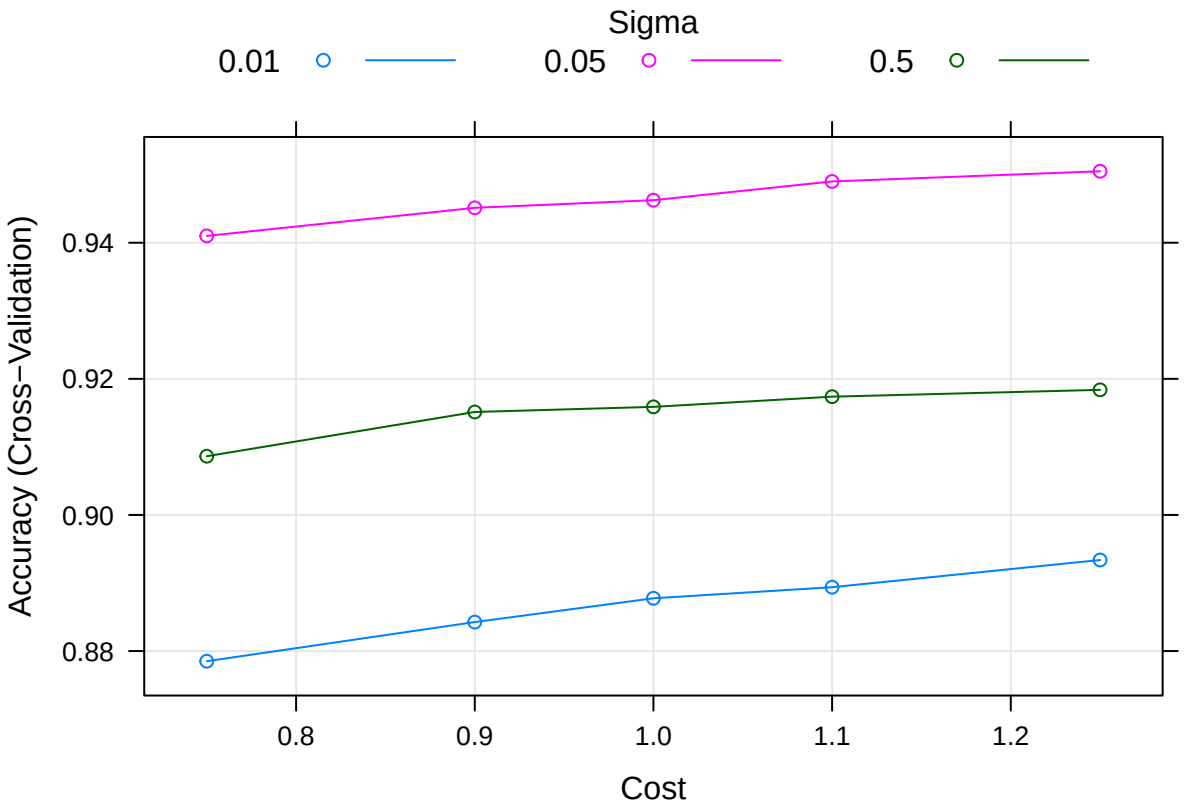
Dot plot illustrating the accuracy of the five different models



Additionally, we can use the following plot to visualize the optimal sigma tuning parameter:

Figure 4

Illustration of the tuning parameters of the Support Vector Machines (SVM)




```
# Support Vector Machines (SVM) with a linear kernel

# Results as shown in the stored finalModel

fit_svm$finalModel
```

```
## Support Vector Machine object of class "ksvm"

##

## SV type: C-svc  (classification)

##  parameter : cost C = 1.25

##

## Gaussian Radial Basis kernel function.

##  Hyperparameter : sigma =  0.05

##

## Number of Support Vectors : 2086
```

```
##
## Objective Function Value : -1904.416
## Training error : 0.035375
```

Final Model

```
# Support Vector Machines (SVM) with a linear kernel on the test dataset
predictions <- predict(fit_svm, test)
cm <- confusionMatrix(predictions, test$stabf)
print(cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction stable unstable
##   stable      670       36
##   unstable     54     1240
##
##                Accuracy : 0.955
##                  95% CI : (0.945, 0.9637)
##     No Information Rate : 0.638
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9021
##
##  Mcnemar's Test P-Value : 0.07314
##
```

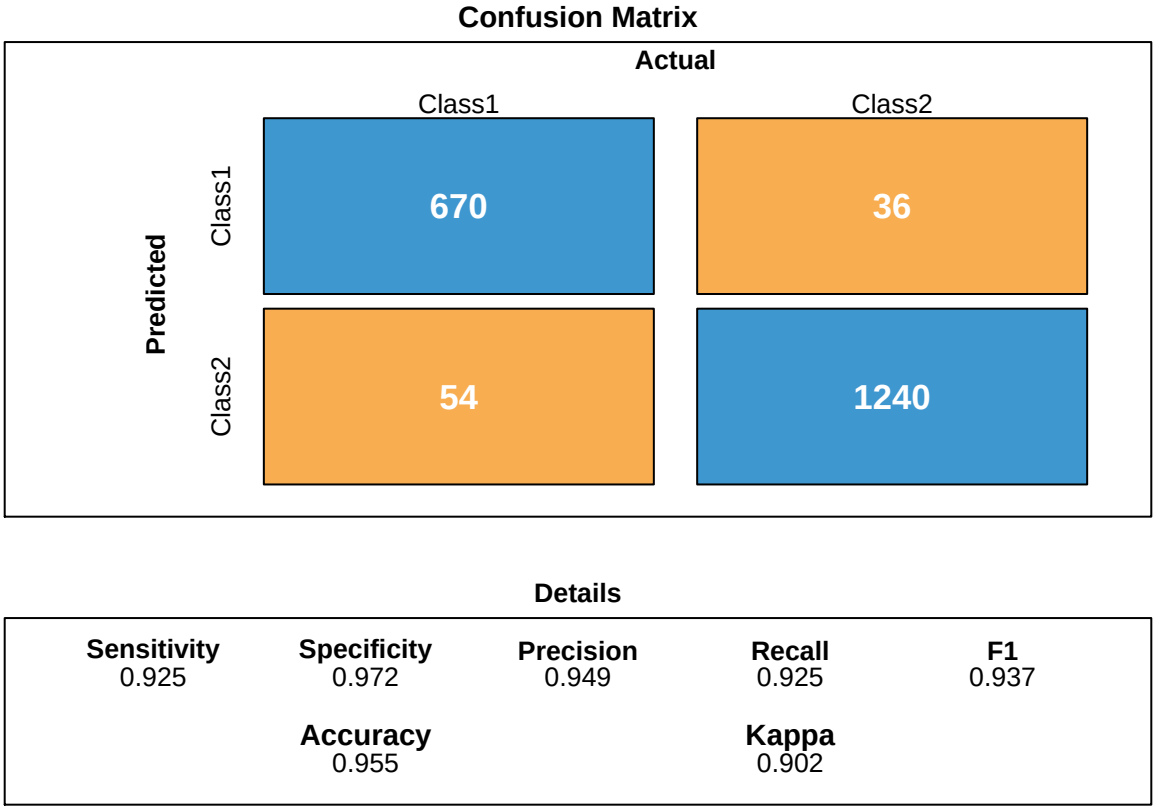
```
##             Sensitivity : 0.9254
##             Specificity : 0.9718
##          Pos Pred Value : 0.9490
##          Neg Pred Value : 0.9583
##              Prevalence : 0.3620
##          Detection Rate : 0.3350
##    Detection Prevalence : 0.3530
##       Balanced Accuracy : 0.9486
##
##        'Positive' Class : stable
##
```

Conclusion

The following confusion matrix shared on www.stackoverflow.com (2017) summarizes the outcome of the Support Vector Machines with Radial Basis Function Kernel model on the Electrical Grid Stability Simulated data set. The ability to make accurate predictions will be a key factor in implementing decentralized electrical grids while maintaining the required reliability. This analysis shows that machine learning algorithms can be implemented within energy management systems to make operators aware of disrupting events, such as surges in either supply or demand.

Figure 5

Illustration of the confusion matrix output



Future work includes further model testing on large data sets which represent readings from actual grids. The four-node star electrical grid presented here represents a very small load and supply sample. The density plot of the stability attribute showed relatively small variances across the 11 different predictors. Future studies should include data sets with disruptions from real world events in order to build confidence in implementing a reliable decentralized electrical grid.

References

Arzamasov, V. (2018). Electrical Grid Stability Simulated Data data set. Retrieved from https://archive.ics.uci.edu/ml/datasets/Electrical+Grid+Stability+Simulated+Data+

Arzamasov, V., Böhm, K., & Jochem, P. (2018). Address at IEEE International Conference. Communications, Control, and Computing Technologies for Smart Grids (SmartGridConn), Section V-A. Towards Concise Models of Grid Stability. Retrieved from https://dbis.ipd.kit.edu/download/DSGC_simulations.pdf

Dua, D. & Graff, C. (2019). UCI Machine Learning Repository. School of Information and Computer Science, University of California.

Kuhn, M. (2019). The caret Package. Retrieved from http://topepo.github.io/caret/index.html

Overleaf. (2020). Help documentation. Retrieved from https://www.overleaf.com/learn

Stack Overflow. (2017). R How to Visualize Confusion Matrix Using the Caret Package. Edited by Cybernetic. Retrieved from https://stackoverflow.com/questions/23891140/r-how-to-visualize-confusion-matrix-using-the-caret-package

The Comprehensive R Archive Network. (n.d.). A Short Introduction to the caret Package. Retrieved from https://cran.r-project.org/web/packages/caret/vignettes/caret.html

The Comprehensive R Archive Network. (n.d.). The YAML Fieldguide. Retrieved from https://cran.r-project.org/web/packages/ymlthis/vignettes/yaml-fieldguide.html