Will Your Event Sell Out? Answering a Prediction Problem Using Machine Learning Algorithms

ECON 483: Economic Applications of Machine Learning

Daniel Bathaei

Amanda Muirhead

August 15, 2023

  While by itself it is a piece of worthless paper, a ticket grants individuals access to an event that is otherwise closed to the public. If an event is well-liked, many people will buy tickets for it. If a performer, sports team, or exhibition achieves fame and popularity, more people might want to attend their events than are slots available. Once every ticket for an event has been purchased, that event is sold out. To performers, selling out is a sign of popularity. To producers, it is a sign of a missed opportunity. They could have charged more for their tickets, booked a bigger venue, or offered VIP packages. The ability to accurately anticipate whether an event will hit full capacity can have significant implications on a business’ demand forecasting, market analysis, and revenue optimization, not to mention the subsequent effects on consumer spending and labour demand. Consequently, producers are asking whether a set of variables such as price and location can predict if their event will sell out. To this end, we use machine learning models to predict whether a show will sell out.

  The onset of machine learning in economics, with its capacity to discern patterns from intricate data, has become an indispensable tool in tackling complex prediction problems. Though it has several varying definitions, the study of machine learning boils down to computer programs’ abilities to learn and adapt to new data without the interference of humans (@babenko2021). To predict the likelihood of an event selling out of tickets, we will employ several machine learning techniques, namely Classification Trees, Logistic Regression, Random Forests, Support Vector Machines, and Neural Networks.

# Background and Current Literature

  Event tickets have been in the news a lot lately, namely for the escalating rates at which they are being priced. Event-goers around the world are increasingly lamenting over the rising costs of event tickets, as reports come out of potential price gouging and fees as high as 78% of the original listed price (@whyever2022). While in 2018 it was reported that ticket sales had doubled since the 1990’s (@ bbc), the percentage increase is now estimated to be near triple their pre-Y2K levels (@ john oliver), proving that forecasting the demand for events is already one of many lucrative strategies being employed by artists and ticket-distributing giants such as Ticketmaster. Currently, event tickets sales are a $78bn USD industry, expected to show a growth of 9.7% by 2024 (@ statista).

  To contextualize our research, we studied existing literature surrounding event sell-out and other related quantity demanded predictions. Previous studies have illuminated various factors that contribute to event success, encompassing variables such as ticket pricing, marketing strategies, historical attendance data, and socioeconomic indicators (@Krueger2005).

## Limitations

  While writing this paper, we observed that companies work very hard to conceal any information related to sales volume and quantity of tickets demanded. StubHub has stopped providing businesses and researchers with access to its application programming interface (API) and only provides API services for producers for their own posted tickets(@ <https://developer.stubhub.com/api-reference/sales#tag/Sales>). Ticketmaster has recently removed any indicator of sales volume on their API database. Companies like Pollstar charge considerable fees before providing any information on ticket sales (@ <https://www.pollstar.com/subscribe>). This further solidifies the current importance and relevance of ticket sale information.

# **Methodology**

  We use publicly available resources[[1]](#footnote-24) to prepare the data for our models. Pursuant to our objective, we collected data from SeatGeek using the website’s application programming interface (API). Similar to other websites, SeatGeek has reduced their available sales volume data. However, SeatGeek still provides some insight on sales volume. Although the company has announced their plan to move away from disclosing any sales volume information through their API, this plan is yet to be implemented.

## SeatGeek

  SeatGeek currently provides information regarding the volume of tickets sold by an event, performer, and/or venue. This information is shared through the ‘score’ variable. The ‘score’ variable is a scaled[[2]](#footnote-25) measure of an event, performer, or venue’s ticket sale volume relative to their type. This means that if an event sells more tickets than any other event similar to its type, that event will receive a score of 1. If an event makes no sales, that event will have a score of 0. For each observation we can gather an ‘event score’, ‘venue score’, and a performer score for each performer in that event. This information is highly valuable since it is derived from the quantity of units sold. Conveniently, SeatGeek analyzes the social media following of each event’s performers, and provides a ‘popularity’ value based on how popular an events performers are. This value is also scaled between 0 and 1. Consequently, the variables we gather from SeatGeek are shown on Table 1:

## Data Collection: Setup

  If an event sells out, SeatGeek immediately removes that events ticket price information. This posed a challenge to our objective since we require both ticket pricing information as well as the ‘sold\_out’ status of the event. To overcome this issue, we set up our data collection strategy accordingly. We gathered information on all events for a given week on the Monday of that week. Then, at 9PM of each day, we collected data on all remaining events and whether they had been sold out. We were able to collect an event’s ticket pricing data, as well as information on whether it sold out[[3]](#footnote-27). The fact that our data is gathered in one week in August, our set up implies the presence of heteroskedasticity in the sample especially if we choose to remove “event date” as a regressor.

  The final data set contains 2056 observations with 26 variables (see Table 1). Of those variables, 5 are the scaled scores calculated by SeatGeek. The following table demonstrates

Summary statistic for SeatGeek generated scores.

|  | score | popularity | venue\_score | prim\_perf\_score | performer\_2\_score |
| --- | --- | --- | --- | --- | --- |
|  | Min. :0.0000 | Min. :0.0000 | Min. :0.0000 | Min. :0.0000 | Min. :0.0000 |
|  | 1st Qu.:0.3180 | 1st Qu.:0.6380 | 1st Qu.:0.4800 | 1st Qu.:0.3300 | 1st Qu.:0.0000 |
|  | Median :0.3810 | Median :0.6700 | Median :0.5800 | Median :0.3900 | Median :0.0000 |
|  | Mean :0.4062 | Mean :0.6741 | Mean :0.5792 | Mean :0.4199 | Mean :0.1454 |
|  | 3rd Qu.:0.4662 | 3rd Qu.:0.7130 | 3rd Qu.:0.6900 | 3rd Qu.:0.5100 | 3rd Qu.:0.3300 |
|  | Max. :0.9600 | Max. :0.9600 | Max. :0.9700 | Max. :0.8600 | Max. :0.8000 |

## Data Collection: Preparation[[4]](#footnote-29)

  In its initial form, the data set possesses irrelevant, unique, and over-categorized columns, as well as incomplete rows. We first import and prepare the data using the following R code:

data <- read.csv("../Week1/SoldOutData.csv")  
data <- subset(data,  
 select = c(  
 -id,-venue, -type,  
 -primary\_performer,  
 -performer\_2, -date)) # remove irrelevant and row-unique columns.  
data <- na.omit(data) # omit NA rows.  
  
# lists to declare each column as its specific type of data.  
date\_columns <- c("local\_date", "announce\_date")   
numeric\_columns <- c("score", "popularity", "venue\_score",  
 "listing\_count", "average\_price", "lowest\_price",  
 "highest\_price", "median\_price",  
 "primary\_performer\_score", "performer\_2\_score")  
integer\_columns <- c("n\_performers", "venue\_n\_upcoming\_events",  
 "primary\_performer\_event\_count", "performer\_2\_event\_count")  
factor\_columns <- c("country", "region", "segment")

  The “state” column has 57 categories comprised of U.S. states and Canadian provinces. Several R functions do not have the ability to handle a 57 category column. To solve this problem we use the U.S. Census Bureau’s definition of Regions to combine geographically similar states and reduce the number of categories to 4 regions: “West”, “Midwest”, “Northeast”, and “South” (@ us census). We also placed Canadian provinces in these 4 categories. We then erase the irrelevant portions of the data set and convert each column to its appropriate type.

# Define Regions  
Northeast <- c("CT", "ME", "MA", "NH", "RI", "VT", "NJ",  
 "NY", "PA", "NL", "NS", "PE", "NB", "QC")  
Midwest <- c("IL", "IN", "MI", "OH", "WI", "IA", "KS",  
 "MN", "MO", "NE", "ND", "SD", "ON", "MB")  
South <- c("DE", "FL", "GA", "MD", "NC", "SC", "VA",  
 "DC", "WV", "AL", "KY", "MS", "TN", "AR", "LA", "OK", "TX")  
West <- c("AZ", "CO", "ID", "MT", "NV", "NM", "UT",  
 "WY", "AK", "CA", "HI", "OR", "WA", "BC", "YT", "NT", "NU")  
# Assign Regions  
data$region <- NA  
data$region[data$state %in% Northeast] <- "Northeast"  
data$region[data$state %in% Midwest] <- "Midwest"  
data$region[data$state %in% South] <- "South"  
data$region[data$state %in% West] <- "West"  
data$region <- as.factor(data$region)

# Prediction Models

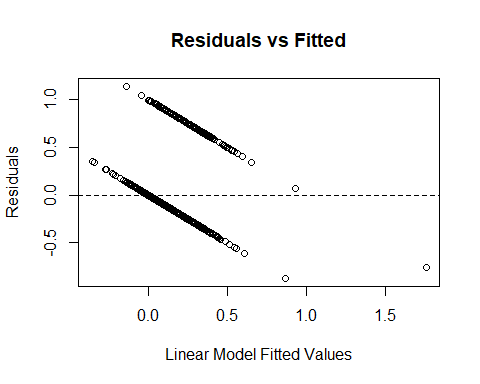
  This paper attempts to solve a classification problem. The dependent variable is binary and can only hold the values 1 and 0. This paper predicts the likelihood of the dependent variable being equal to 1. We employ models that we will answer the following general equation:

  where represents the likelihood of the sold\_out variable being equal to 1, represents a set of covariates in the data, and represents some form of transformation–if any–performed on the covariates.

  In our prediction models, we aim to maximize the variance in the sample’s response variable that the model accounts for in its predictions while minimizing the likelihood of overfitting to the sample. Overfitting is defined as the production of a model that corresponds too closely to a specific sample, thus losing its accuracy in predicting future observations. We combat overfitting using various techniques based on each model we use.

## Linear Model

  First, we examine the residuals of the linear model to better understand the nature of our sample:



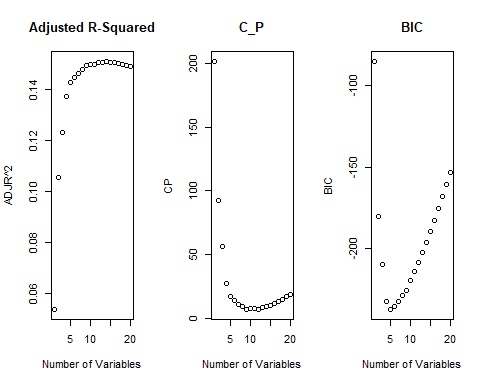
  We observe a clear downward sloping pattern in the Residuals vs. Fitted Values plot. This identifies that the model is inaccurate in capturing the relationship between the variables. Thus, making a linear model inconsistent. However, we may still examine whether selecting a smaller subset of covariates will help reduce overfitting for our non-linear models.

## Best subset selection

  Residual sum of squares (RSS) is defined as the sum of the discrepancies between the actual data and the predictions of a model. In other words, the RSS measures how good the model fits the data. Best subset selection aims to achieve the lowest RSS with the fewest number of variables by creating a penalty term for each extra regressor used in the model. In this paper we separately use the Bayesian Information Criterion (BIC), Adjusted , and as the information criteria for our subset selection:

BESTSUBSET <- regsubsets(sold\_out ~., data = train\_data, nvmax = length(colnames(train\_data)))  
BSM <- summary(BESTSUBSET)  
data.frame(  
 Adj.R2 = which.max(BSM$adjr2),  
 CP = which.min(BSM$cp),  
 BIC = which.min(BSM$bic)  
)

## Adj.R2 CP BIC  
## 1 14 9 5



Best achieved information criteria scores for the optimum combination covariates given different numbers of allowed regressors.

The following R code provided us with the best variables for our prediction models based on each information criteria. Given that that we have a classification problem, we use the logistic model to cross-validate the the accuracy of our Best Subsection Selection. We regressed the sold\_out column 4 separate times. Once on all the available covariates, and once for each set of selected covariates displayed on Table 3.

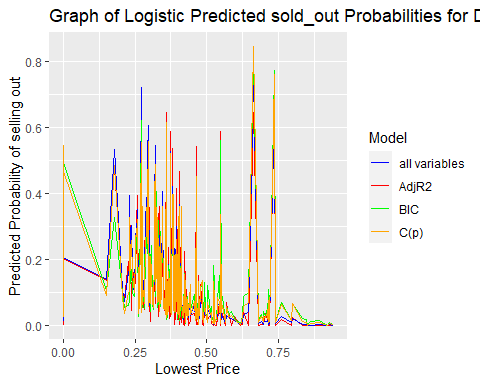
Best selected subsets based on each criterion

| Adj.R2 | Cp | BIC |
| --- | --- | --- |
| local\_date | announce\_date | venue\_score |
| score | n\_performers | lowest\_price |
| announce\_date | venue\_score | median\_price |
| n\_performers | lowest\_price | segmenttheater |
| popularity | median\_price | primary\_performer\_event\_count |
| venue\_score | segmenttheater |  |
| lowest\_price | primary\_performer\_event\_count |  |
| median\_price | primary\_performer\_score |  |
| segmenttheater | performer\_2\_score |  |
| primary\_performer\_event\_count |  |  |
| primary\_performer\_score |  |  |
| performer\_2\_score |  |  |
| regionNortheast |  |  |
| regionWest |  |  |

Based on Table 3, we can observe that SeatGeek’s score covariates hold strong predictive power.

## Logistic Model

  Given that our response variable can only be 0 or 1, we have to assume that probabilities fall within this range. A linear model may predict that an event will sell out with a negative probability, or a probability of higher than 100%. One method used to ensure that the predicted probabilities fall within 0 and 1 is the logistic model. The logistic model leverages the logistic function to constrain its predicted probabilities within the bounded interval of 0 to 1. Employing the logistic model ensures that the continuous probability estimates are confined to this specific range, thus avoiding possible issues that might arise from linear models and providing a more suitable framework for binary response variables.



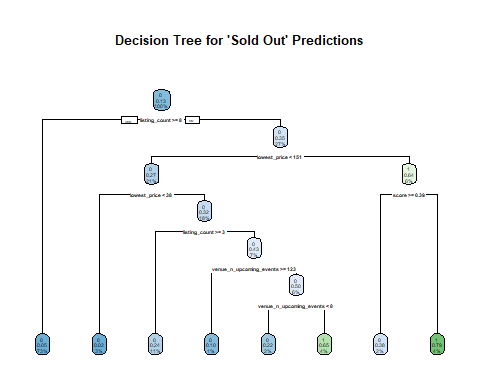
  Upon reviewing the Figure above, we observe that the likelihood of selling out not only does not increase as the score and venue\_score variables for that event increase, we observe that highly popular shows are less likely to sell out.

  However, it is important to note that the Logistic model assumes the independence of errors, absence of multicollinearity, as well as strongly influential outliers, which might make it unsuitable for our data, given that all score variables are derived from overlapping ticket sales information.

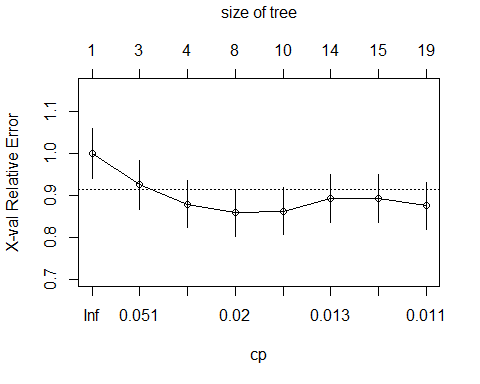
## Classification Trees

  Classification Trees, on the other hand, do not rely on such parametric assumptions, can capture more complex patterns in the data, and can address strong outliers through pruning. In this paper, we first created an unpruned tree. We then used the complexity parameter table to derive the best complexity parameter for pruning. We then tuned the model according to the best complexity parameter. The below R chunk represents these steps[[5]](#footnote-45):

full\_tree <- rpart(sold\_out ~ ., data = train\_data, method = "class")  
cp\_table <- data.frame(full\_tree$cptable)  
best\_cp <- cp\_table[which.min(cp\_table$xerror), "CP"]  
pruned\_tree <- prune(full\_tree, cp = best\_cp)  
rpart.plot(pruned\_tree, main = "Decision Tree for 'Sold Out' Predictions")



  The figure above is derived by minimizing the complexity parameter, as well as the cross-validation error (xerror) in our model. The following graph displays the tree size with the lowest relative error:



rpart-generated Cross Validation Errors for Each Tree Size

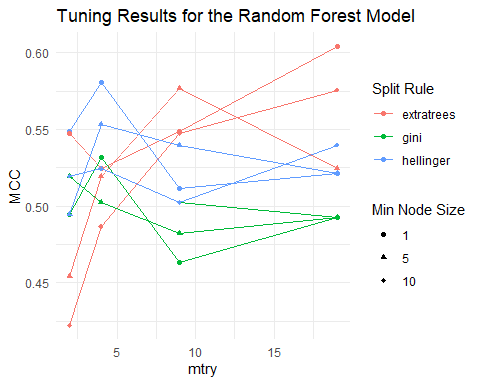
  While one tree may provide insightful information, we may still improve accuracy by using multiple trees and using the majority prediction among those trees.

## Random Forest

  A Random Forest model begins by creating multiple subsets of the data, including some observations many times and allowing the possibility of not using an observation at all. This is known as bootstrap sampling. The model also randomly chooses a set of covariates for each subset, introducing further randomness into the model. This is known as feature sampling. The model then trains a tree based on each subset, creating a forest of classification trees.

  Random Forest models can be tuned to follow specific criteria for how trees split the data, and what the minimum number of observations must be at each terminal node in the tree. In this paper, we utilize all available splitting rules, as well as a range of minimum terminal node sizes in order to find out the best Random Forest model. The following R code represents all tuning choices we utilize for our model:

num\_predictors <- ncol(train\_data) - 1  
mtry\_values <- c(2, floor(sqrt(num\_predictors)), floor(num\_predictors / 2), num\_predictors)  
RDF\_tune\_grid <- expand.grid(mtry = mtry\_values, splitrule = c("gini", "hellinger", "extratrees"), min.node.size = c(1,5, 10))



## Results

We generated several other models for our prediction problem. However, due to their low predictive power, we opted to focus on the ones we did. The following table summarizes the resulting Matthews Correlation Coefficients (MCC) for all models created for our prediction problem:

All models and their corresponding MCC

Model

MCC

Random Forest - Best Model

0.5531944

Unpruned Decision Tree

0.4293986

Pruned Decision Tree

0.3674587

Support Vector Machine

0.2965576

Logit-All Vars

0.2540229

Logit-Adjr2

0.2540229

Logit-Cp

0.2313670

Logit-BIC

0.1186912

# Conclusion

  In our study we analysed the dynamic nature of event popularity, which highlighted the importance of demand forecasting along with industrial organisation and market analysis. Our data sourced from SeatGeek, though boasting intriguing variables such as artist and venue popularity score, may not provide a full representation of the greater population due to potential selection and temporal biases. Another limitation noted early was the restricted nature of some data of interest. The absence of certain predictors and the subsequent quality issues underscored the need for careful consideration when interpreting our findings.

  Additionally, it is important to recognize that the effectiveness of predictive models is intertwined with the parameters chosen during their formulation and training in cross validation. Parametric estimators take into account local data to the observation of interest, which grants us the freedom from restrictive parametric forms. Whereas OLS is a popular example of a global parametric estimator, the functional form of regression functions such as these runs the risk of giving us deficient predictions.

  As is the case with many supervised learning methods, we were subject to some select limitations when applying parameters to our entertainment industry data. Due to these limitations we observed a noticeably lower predictive ability from the logistic regression method, the reason for this being the imposition of the maximum likelihood estimator’s parametric assumptions. Although often an instinctive approach to classification problems with two classes, implementing Support Vector Machines required much more data than we had available to us, and therefore it also produced less than ideal predictive results. The balance between bias and variance, as well as the trade-offs we face in model complexity, necessitates careful consideration to ensure the reliability of our findings. Thus, the method proving to be our best predictor was the pruned random forest, borne from several classification trees.

  In conclusion, our study not only highlighted the multifaceted landscape of event popularity prediction but also underscored the intricate interaction between parameterization and predictive power. As we look ahead, refining and expanding the boundaries of predictive modelling we look forward to adding to the insightful predictions in the dynamic domain of event management.

# References

1. Ethical obligations dictate that we only use publicly available data, even if we use scraping methods. [↑](#footnote-ref-24)
2. [↑](#footnote-ref-25)
3. Here, we note that an event could be sold out prior to its date. While it would be interesting to evaluate how far from its actual date would an event sell out, such a premise is beyond the scope of this paper. Thus, the earliest time that an event reaches “sold\_out” status was ignored. [↑](#footnote-ref-27)
4. We used python throughout the process of sending API requests and managing the data. To replicate the process, please visit <https://github.com/dbathaei/econ483>. [↑](#footnote-ref-29)
5. In the Rmd file, there are available examples to produce a graph for the unpruned decision tree. However, given that our objective is to find the best-pruned, cross-validated tree, that is the only one we displayed in this paper. [↑](#footnote-ref-45)