Predicting Movie Success Using Machine Learning

Zachary Szentimrey
zszentim@uoguelph.ca
University of Guelph
Guelph, Ontario

Rylee Thompson
rylee@uoguelph.ca
University of Guelph
Guelph, Ontario

Adesh Kadambi
akadambi@uoguelph.ca
University of Guelph
Guelph, Ontario

ABSTRACT

The motion picture industry, generating over billions of dollars every year in the United States alone, is comprised of film production companies, film studios, cinematography, animation, screenwriting, pre-production, post production, actors, and film directors. With so many moving parts, being able to predict whether a movie will be profitable or earn an Academy Award prior to release is integral for budgeting and advertising to ensure movie studios do not waste resources. This paper explores the possibility of accomplishing this task using all the freely available metadata available prior to movie release on websites like IMDb. A principal component analysis (PCA) was used for dimensionality reduction and feature selection. Several machine learning techniques were then employed (Generalized Linear Regression, Decision Tree Regression, Gaussian Naive Bayes Classification, and Support Vector Machine Classification) using features like actors, director, Metacritic rating, IMDb rating, run time, and movie genre. It was found that decision trees were better at predicting movie box office success than the linear regression model, and both classification models used had high accuracy. However, the accuracy was found to be inflated due to the data points being heavily skewed towards 0 Oscar wins.

*ENGG*3130 '19, April 2019, Guelph, Canada*

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00

KEYWORDS

IMDb, generalized linear regression, support vector machine, Gaussian naive Bayes, decision tree, features, web scraping, prediction, machine learning

ACM Reference Format:
Zachary Szentimrey, Rylee Thompson, and Adesh Kadambi. 2019. Predicting Movie Success Using Machine Learning. In *Proceedings of ENGG*3130 Project (ENGG*3130 '19).* University of Guelph, Guelph, Canada, 6 pages.

1 INTRODUCTION / PROBLEM DESCRIPTION

As of 2018, the global box office is worth $41.7 billion, with Hollywood contributing upwards of $11 billion every year [5]. Being able to predict a movie's success would be a powerful tool for movie studios to distinguish between movies they would like to pursue and invest in versus movies that would not turn a profit. With the vast amount of data published on the Internet, and especially the freely available metadata found on websites like IMDb, this paper explores the possibility of taking advantage of these resources to make viable predictions.

For the purpose of this paper, **movie success** is defined with respect to two different parameters. The first measure of success is the revenue generated in the US box office. Movies generate income from a number or revenue streams including box office sales, streaming, rentals, and purchases to name a few. By isolating revenue to one revenue stream (i.e. US box office sales), comparisons between movies are made easier and more accurate. The second measure of success is the how many Academy Awards the movie will win. Both of these parameters will be predicted using the metadata available on IMDb.

2 RELATED WORK

There has been much research in the area of movie success in recent years. This research is typically centred around finding proper techniques of determining success. With that being said, many different techniques have been used to extract important information numerically from previous movies to predict future movie success.

Lee *et al.* were able to predict box office success as a classification problem. This was done by discretizing the box office performance variable (dependent variable) into six classes [4]. They determined the range for each class by interviewing industry experts [4]. The movie database used in the study scraped movies released between October 25, 2012 to December 31, 2014 the Korean Film Council web page [4]. An important consideration to note is the dataset being comprised of only the top 400 movies. This structure was chosen because they believed including movies beyond the top 400 can lead to large improvement in classifications due to a large amount of films outside the top 400 usually considered to be flops, skewing the results [4]. They chose very unique features for their data including observing a movie's genre, whether the movie has a sequel or not, movie buzz (number of comments made about the movie before release), whether the movie is based on previous stories or is original and the actors involved in the film [4]. This group tried to include many social factors which is an important aspect of movie success.

In addition, Lee *et al.* built seven different machine learning algorithms to build models. They used *adaptive tree boosting, gradient tree boosting, linear discriminant, logistic regression, neural networks, random forests,* and *support vector classifiers*[4]. After determining the performance of each model, it was found that gradient tree boosting had the best accuracy with 88.3% and support vector classification performed the worst with 28.7% accuracy [4].

The work done by Meenakshi *et al.* showed how to predict the box office success using classification techniques for Bollywood movies. The researchers separated the data into separate training and test data sets with three classes; Hit, Flop and Average [6]. These labels were assigned using the website Box Office India from a data set of over 3,000,000. The features they used includes actors, actresses, composer, genre, director, producer, and music director [6]. They used these features as they were most readily available in the IMDb database file *ratings.list.gz*. After cleaning the data and creating a database of features, Meenakshi *et al.* used two different classification techniques which include *K-means cluster and decision trees*. The K-means technique was used to determine which feature impact the success of the movies the most. It does this by clustering the movies into 3 classes using the Euclidean measure of distance and tries to reduce this distance for each movie based on the clusters. The team found that the genre feature makes a large impact on movie success. After, they used decision trees to try and predict movie success and compare with the actual result of the movie [6]. They found there model not be a good predictor of movie success and decided that factors including word of mouth and blog posts can play a large role in movie success [6].

Finally, Subramaniyaswamy *et al.* used many different features in order to create models that can predict box office success. Their definition of success is the amount of money a movie makes in comparison to its budget. They used data from many different kinds of media including BoxOfficeMojo, Wikipedia and Youtube [10]. The features scraped included opening date, domestic gross, international gross, trailer views, cast and crew, genre, Wikipedia views, and Rotten Tomatoes score.

In the end, only 138 movies were scraped, which is small compared to most other researchers. After obtaining the features, they classified each movie as either low, medium, or big budget. Success for each class was defined differently, where low budget movies were considered successful if revenue exceeded budget by a factor of 2 [6, 10]. Similarly, the factors for medium and big budget movies were 2.5 and 3 respectively with anything lower being considered a failure [6, 10]. Both linear regression and SVM classification methods were employed - achieving an accurate classification rate of 56.52% [6].

From this, it can be stated that most papers define success as box office revenue. This paper hopes to accurately predict US box office revenue using similar techniques as seen in literature (i.e. GLM). Web scraping was used in the majority of these papers to create the datasets. In order to make this whole process more efficient, dimensionality reduction can be used to determine the features that have the highest effect on the variance in the data. This paper also hopes to extend previous work by accurately classify Academy Award wins using similar techniques like SVM and Naive Bayes classification.

3 METHODOLOGY

As suggested in the problem description, success was defined as a combination of the revenue generated in the US Box Office and how many Academy Awards the movie will win. Both of these parameters will be predicted using the metadata available prior to movie release. To accomplish this task, machine learning models for both regression and classification will be used. The workflow that was incorporated (Figure 1) involves web scraping from IMDb, preprocessing the data, performing feature selection, and then training

the model. A 90/10 training-test split was used for regression and a 80/20 training-test split was used for classification.

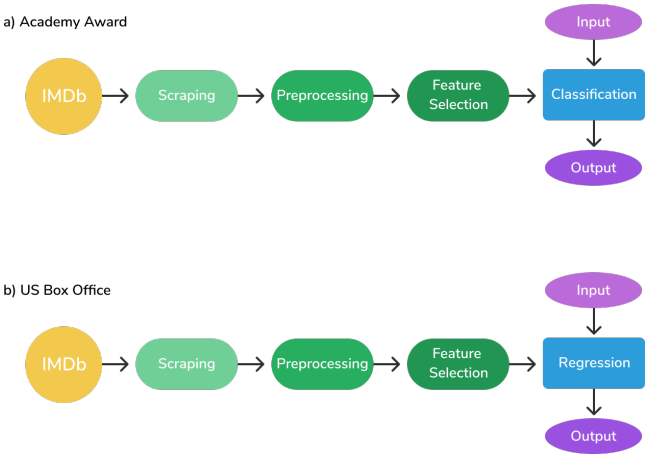

Figure 1: Flow diagram of methodology for (a) Academy Awards through classification and (b) US Box Office through regression.

3.1 Web Scraping

The Python library, *BeautifulSoup4*, was used to parse the HTML content and scrape data from IMDb. Over 6000 movies were scraped in total through this method, retrieving the following information for each movie.

- Actors involved
- Director
- Genre
- Run time
- Box office gross
- Metacritic rating
- IMDb rating
- Parental rating

If a movie was missing any of these parameters, they were immediately dropped from the list in order to prevent empty values from populating the dataset.

3.2 Database Manipulation

Since IMDb does not contain Academy Award wins in an easy to scrape format, a Kaggle dataset was used to populate the fields [1]. This data was stored in a nested dictionary, where the first key is the year, the second key is the name of the star (actor or director), and the value is the number of Oscars won in that year by the star. This data provides a way of numerically assigning a weight (or rating) to actors and directors featured in each movie, which is required in order for a machine learning model to be used. The major challenge that stemmed from this was that Oscar wins should only include wins for movies up until that point; future Oscar wins by a star cannot count for movies made in the past. In order to solve this, the previous dictionary was used to create another nested dictionary of the same form, however the value is now the number of combined Oscars the star has won up until the given year. This running total dictionary was created using the function below.

```
def getRunningTotal(Dict):

    totalDict = {}
    totalDict[startYear] = Dict[startYear]

    for year in range(startYear + 1, endYear + 1):
        totalDict[year] = {}
        totalDict[year].update(totalDict[year - 1]) #update
            dictionary with previous year to keep a running
            total each year

        for key, value in Dict[year].items():
            try:
                totalDict[year][key] += 1 #if the actor is
                    already present increment by 1
            except:
                totalDict[year][key] = 1 #set it to 1 if not
                    already present
    return totalDict
```

3.3 Data Pre-Processing

Before building a predictive model, it is important to normalize and numerate the data. For example, the genre data scrapped is a string and thus cannot be used in a machine learning model. To solve this, a unique number from 0 to the total number of genres was assigned to each genre in a dictionary. This converts the data into numbers such that it is easy to use in a predictive model. Non-numerical features are converted to numerical data using the same process as above, and all features are then normalized. The goal of normalization is to use a common scale for numeric columns without losing information in the values or distorting the difference values (range). This normalization will prevent some features such as the year (which is in the range of thousands) from overpowering other features such as Metascore (which is in the range of ones) when evaluating the co-variance matrix values in the principle component analysis (PCA).

A PCA is used to reduce dimensionality by removing unnecessary features which do not add much variance to the data. It is performed by taking the covariance of the feature matrix which shows how each feature relates with one another. After that, the eigenvalues are found for the covariance matrix and these eigenvalues are listed in descending order from the largest eigenvalue. Typically in PCA, only the top k number of features are taken which represent around 95% or more of the variance. This means that the features with the k largest eigenvalues are kept while the other features are removed.

3.4 Models

A model is just an abstraction of reality that provides an approximation of some relatively more complex phenomenon. Models are generally classified as deterministic or probabilistic. In a deterministic model, the systems outcomes and responses are precisely defined, often by a set of equations (i.e. Ohm's Law or ideal gas law) [7]. In a probabilistic model, the system responses exhibit variability

because the model either contains random elements or is impacted in some way by random forces [7].

3.4.1 Generalized Linear Model (GLM) Regression. As the name suggests, the GLM is a flexible generalization of ordinary linear regression that can be viewed as a unification of linear and nonlinear regression models that incorporates a family of normal and non-normal response distributions [3, 7]. It has three components, the linear predictor (Equation 1), the link function that describes how the mean depends on the linear predictor (Equation 2), and the variance function that describes how the variance depends on the mean (Equation 3), where the dispersian parameter, ϕ, is a constant [3, 7].

$$\eta_i = \beta_0 + \beta_1 x_{1i} + ... + \beta_k x_{ki} \tag{1}$$

$$g(\mu_i) = \eta_i \tag{2}$$

$$Var(Y_i) = \phi V(\mu) \tag{3}$$

3.4.2 Decision Tree Regression. Decision trees are created using two main steps: induction and pruning. Induction involves (1) determining the best feature in the data set to split the data on, (2) splitting the data set into subsets (or nodes) that contain the possible values for the best feature, and (3) recursively generating new tree nodes until maximum accuracy is reached while minimizing the number of nodes in order to create the tree [8]. For a regression tree, simple squared error is used as the cost function (Equation 4) where Y is the ground truth and $\hat{Y}$ is the predicted value.

$$E = \sum (Y - \hat{Y})^2 \tag{4}$$

Due to the nature of training decision trees, they can be prone to over-fitting. Thus, setting the correct value for the minimum number of instances per node can be difficult task. Therefore it is generally acceptable to lower the minimum, and then using tree pruning to remove unnecessary splits [8].

3.4.3 Gaussian Naive Bayes Classification. Naive Bayes is a simple and effective machine learning classifier that makes classifications using the Maximum A Posteriori estimation in a Bayesian setting. A Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature (Equation 5), an assumption that is hardly ever true in practice but still yields excellent classification results [2]. Naive Bayes is particularly effective on large data sets with high dimensional data.

$$p(x|y) = \prod_i p(x_i|y) \tag{5}$$

Incorporating the assumption into Bayes theorem yields:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} = \frac{\prod_i p(x_i|y)p(y)}{p(x)} \tag{6}$$

Given this probability that a data point will fall under a certain class, the Naive Bayes method uses the Maximum A Posteriori method for classification - simply picking the y that has the largest probability given the data point's features [2].

3.4.4 Support Vector Machine Classification. Support vector machines (SVM) are models which find the best hyper-plane (decision boundary) to separate the classes. It does this by finding the best weight values that create the lowest error. It can do this non-linearly by introducing new dimensions which are made of the previous dimensions. The radial basis function kernel (most popular kernel) tries to create decision surfaces by determining the squared Euclidean distance between feature vectors [9]. The discriminant function (decision boundary) learned by an SVM can be seen in Equation 7 which uses the radial basis function kernel seen in Equation 8 [9]. This kernel can be used to obtain practically any decision boundary [9].

$$f(x) = \sum_{i=1}^{n} y_i a_i e^{-\frac{||x_i - x||^2}{2\sigma^2}} \tag{7}$$

$$k_G(x, x') = e^{-\frac{||x_i - x||^2}{2\sigma^2}} \tag{8}$$

4 RESULTS

The cumulative explained variance of the eigenvalue feature representation from the PCA (Figure 2) shows that approximately 95% of the variance in the data can be explained from the first five principal components. In order to speed up the machine learning models, these five features can be used to generate an approximation that is good enough to explain the data.

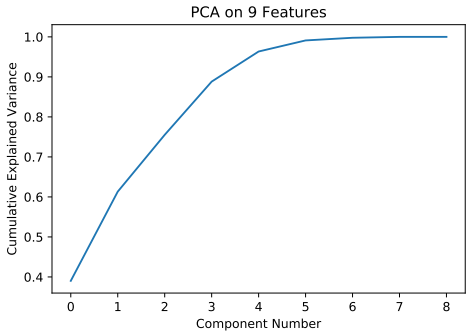

Figure 2: Principle component analysis on the features.

4.1 Predicting a Movie's Gross

A GLM was first constructed to predict box office earnings for each movie as it is simple to understand and easy to build. The accuracy of the model was determined based on a boundary scale for money earned. This means that in order to determine if the predicted earnings is equal to the actual earnings, some range of boundary values is needed; predicting the exact amount of money a movie will earn to the nearest dollar is almost impossible to do. For example, if a movie earned $20,000,000 at the box office and the model predicted it would earn $5,000,000, a threshold range of at least $15,000,000 is needed for the prediction to be considered correct. The accuracy for this model over a range of boundaries is shown in Figure 3.

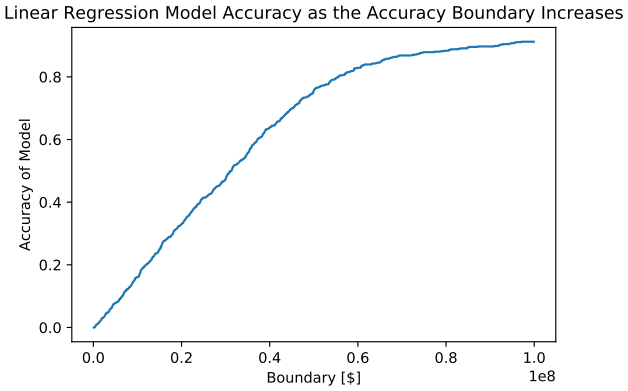

Figure 3: GLM accuracy over a range of boundaries.

In order to see if improvements could be made, a decision tree regression model was used as it may be able to take into account any non-linearities present in the data. The results are analyzed and plotted in the same way as the GLM model to allow for comparisons, and are shown in Figure 4.

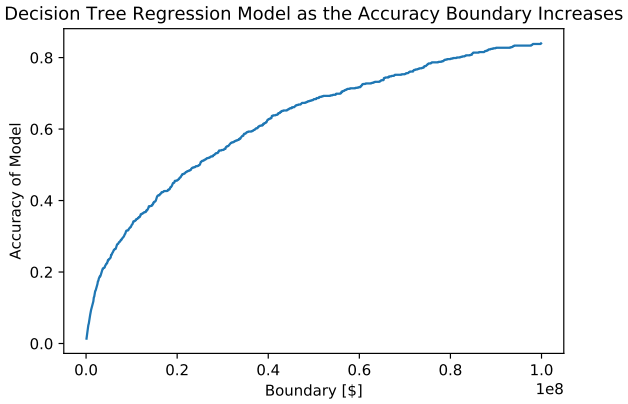

Figure 4: Decision Tree model accuracy over a range of boundaries.

4.2 Predicting Oscars

The first model used is a Gaussian Naive Bayes model which implies independence between features (which may not necessarily be true). This model is used over other Naive Bayes models since the data is continuous in nature from when it was normalized/standardized. Naive Bayes model assumes this data follows a normal distribution and is easy to calculate because of these restrictions. The confusion matrix generated for this model shown in Figure 5.

An SVM classifier was used to see if it would yield better results since they are good at fitting non-linear data using kernel tricks. The specific kernel used was the radial basis function kernel for its success with non-linear data and the confusion matrix was generated (Figure 6).

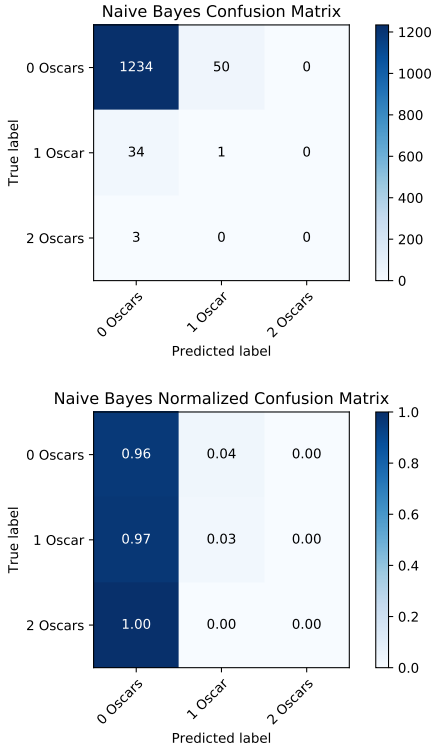

Figure 5: Gaussian Naive Bayes confusion matrices.

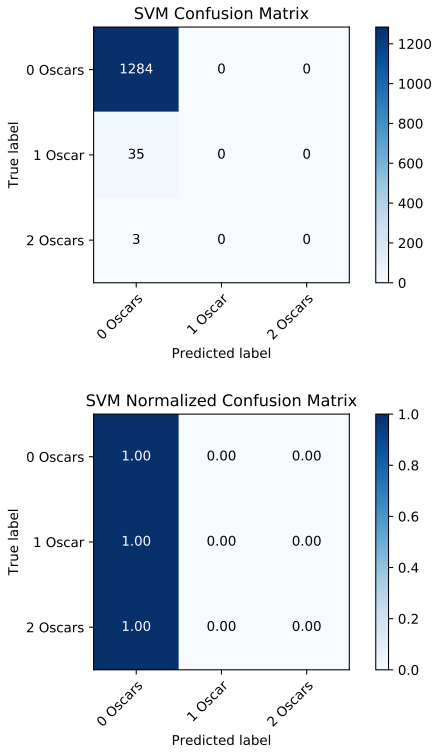

Figure 6: SVM confusion matrices.

5 DISCUSSION

As you can see in Figure 3 and Figure 4, neither model were particularly accurate, with the GLM being slightly more accurate at large boundary ranges, while the reverse is true for smaller ranges. For example, when a boundary threshold of $20,000,000 was used it can be seen that the GLM model and decision tree models had an accuracy of 25% and 40%, respectively. At a boundary range larger than approximately $50,000,000, the GLM outperforms the decision tree model. However, large boundaries do not give much information on whether a movie will be successful or not; the lower money boundaries are more important in predicting success.

The accuracy of the Gaussian Naive Bayes and SVM models was found to be 93.4% and 97.5%, and the confusion matrices can be seen in Figure 5 and Figure 6, respectively. However, this accuracy is heavily inflated as a result of skewed data; a large percentage of movies being released and thus included in our data win do not win an Oscar. If the 0 Oscars class was omitted, the accuracy would be very poor as shown in the confusion matrices. Although the SVM had a higher accuracy, you could warrant that it performed worse; it had an accuracy of 0% when predicting movies that won more than 0 Oscars, and never predicted any movies to actually win an Oscar. This issue with skewed data is a problem that also occurred in one of the related works, and it is important to think about when building machine learning models in the future.

With respect to accuracy, one important consideration that was ignored during the evaluation of the model were Type I and Type II error, commonly referred to as false positive and false negative rate. This is a very important evaluation metric for the use of these models since these errors can result in movie studios rejecting movie ideas based on the model result. In order to test for these, an AUC-ROC curve can be constructed to determine how well the classifier performs, with 1.0 being the perfect classifier and 0.5 being a terrible classifier.

6 CONCLUSION

The intent of this study was to predict movie success using both classification and regression machine learning methods. Between the two initially defined parameters for success, it was found that the classification models for predicting Oscar wins were more accurate (93.4% for Naive Bayes and 97.5% for SVM) than the regression models used for predicting US box office sales (25% for GLM and 40% for decision trees). These results show promise but the low accuracy of the regression models may indicate that there is important predictive data that could not be collected during the scraping process. Additionally, the high accuracy of the classification models can be attributed to the majority of movies not winning an Academy Award. More interesting results could have been obtained by removing a percentage of movies that won 0 Oscars such that the data is not skewed. More analysis is required for both types of models in terms of Type I and Type II error which can be performed using an AUC-ROC curve. Using neural networks or other predictive models could potentially improve accuracy and result in a better model. Also, using features that include social factors such as celebrity influence and number of Tweets referencing a movie could improve the accuracy as done in a related work.

REFERENCES

[1] Academy of Motion Picture Arts and Sciences 2017. The Academy Awards, 1927-2015. https://www.kaggle.com/theacademy/academy-awards.

[2] D.A. Forsyth. 2016. *Applied Machine Learning* (4th. ed.). Illinois Computer Science. http://luthuli.cs.uiuc.edu/~daf/courses/LearningCourse/learning-book-19-jan-small.pdf

[3] Andrew Gelman and Jennifer Hill. 2007. *Data Analysis Using Regression and Multilevel/Hierarchical Models* (1st. ed.). Cambridge University Press.

[4] Kyuhan Lee, Jinsoo Park, Iljoo Kim, and Youngseok Choi. 2018. Predicting movie success with machine learning techniques:ways to improve accuracy. *Information Systems Frontiers* 20, 3 (June 2018), 577–588. https://doi.org/10.1007/s10796-016-9689-z

[5] Dave Mcnary. 2019. 2018 Worldwide Box Office Hits Record as Disney Dominates. (January 2019). https://variety.com/2019/film/news/box-office-record-disney-dominates-1203098075/

[6] K. Meenakshi, G. Maragatham, Neha Agarwal, and Ishitha Ghosh. 2018. A Data mining Technique for Analyzing and Predicting the success of Movie. *Journal of Physics: Conference Series* 1000, 1 (January 2018), 1–9. https://doi.org/10.1088/1742-6596/1000/1/012100

[7] Raymond H. Myers, Douglas C. Montgomery, G. Geoffrey Vining, and Timothy J. Robinson. 2010. *Generalized Linear Models: with Applications in Engineering and the Sciences* (2nd. ed.). Wiley.

[8] James D Malley; Karen G Malley; Sinisa Pajevic. 2011. *Statistical learning for biomedical data* (1st. ed.). Cambridge University Press.

[9] Bernhard Scholkopf, Koji Tsuda, and Jean-Philippe Vert. 2004. *Kernel Methods in Computational Biology* (1st. ed.). The MIT Press.

[10] V. Subramaniyaswamy, M. Viginesh Vaibhav, R. Vishnu Prasad, and R. Logesh. 2017. Predicting Movie Box Office Success using Multiple Regression and SVM. *Proceedings of the International Conference on Intelligent Sustainable Systems* 1, 1 (January 2017), 1–5. https://doi.org/10.1109/ISS1.2017.8389394