Exploring Machine Learning Methods for Stock Ratings Classification

Ruining Feng, Bowen Gu, Jin Qin, Yichuan Zhang

February 2024

1 Introduction

This project investigates the capability of machine learning methods to forecast companies' ESG ratings. With the growing emphasis on sustainable investment, ESG ratings have become a cornerstone for investors and stakeholders in making informed investment choices. Utilizing advanced machine learning techniques, specifically Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Support Vector Machine (SVM), and K Nearest Neighbours (KNN). this study scrutinizes the efficacy of various financial indicators in predicting a company's future ESG rating.

In 2016, Plbennikov et. al shows that ESG rating is correlated with the performace of corporate bond [5]. In 2019, Giese et. al shows that the ESG have great impact on equity valuation, risk, and performance [4]. Given that a company's rating often reflects its past performance, our analysis aims to pioneer a method for anticipatively determining potential shifts in ESG ratings before they are officially updated by rating agencies. This forward-looking approach seeks to equip investors with early insights, enabling proactive decision-making in the realm of sustainable investing.

2 Data Exploration

This data was aggregated from Bloomberg by Augustud Kasper, Koontze Jang, WaiLok Ho, and Andrew Garrido. The data combines the ESG ratings for stocks in Russell 3000 index in Spring 2022 whose ESG score is available and P/E ratio smaller than 200.

The dataset under consideration is characterized by an imbalance in MSCI ESG ratings, as shown in Figure 1. This is particularly notable at the 'CCC' level, indicating a concentration of companies categorized as ESG laggards. This imbalance may reflect on the rating system's criteria. The financial metrics included in the dataset comprise one categorical and six numerical variables. Among these, variables such as Return on Assets (ROA LF), Return on Equity (ROE LF), Price to Earnings (P/E), and Dividend Yield (Dvd Est Yld) exhibit a right-skewed distribution. This skewness suggests that while a majority of companies cluster towards lower values, a few outliers significantly outperform. This should be considered when doing preprocessing before fitting the model.

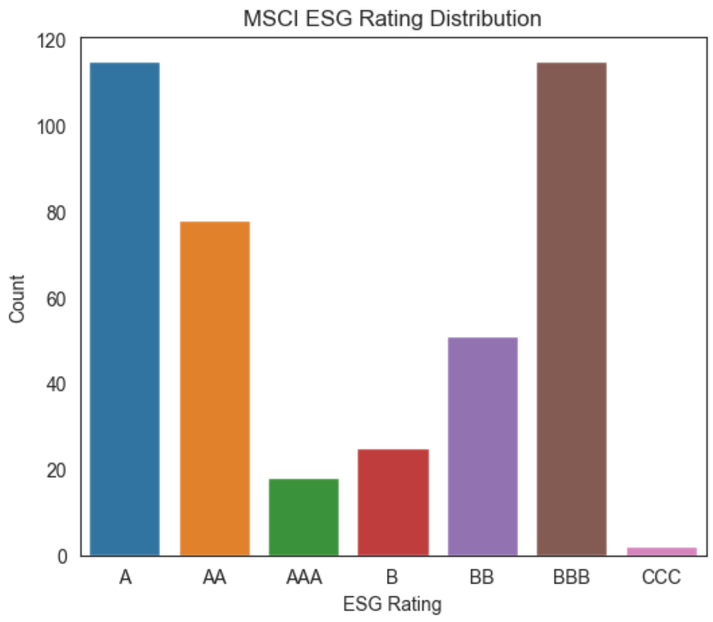


Figure 1: MSCI ESG rating distribution for each class

3 LDA and QDA

LDA is an approach used in supervised machine learning to find a linear combination of features that characterizes or separates two or more classes of objects. LDA assumes that the conditional probability density functions of different classes follow Gaussian distributions different mean and same covariance matrix: $p(x \mid y=k) \sim \mathcal{N}(\mu_k, \Sigma)$. We use the Bayes estimator $\widehat{\mu}_k = \frac{1}{n_k}\sum_{i:y_i=k} x_i$, $\widehat{\Sigma} = \frac{1}{n}\sum_{k=1}^{K}\sum_{i:y_i=k}(x_i-\widehat{\mu}_k)(x_i-\widehat{\mu}_k)^\top$. Then we classifiy using the decision function

$$\widehat{h}(x) = \arg\max_k \log(\widehat{\pi}_k) - \frac{1}{2}(x-\widehat{\mu}_k)^\top \widehat{\Sigma}^{-1}(x-\widehat{\mu}_k)$$

For our specific question, we apply several key preprocessing and analysis steps designed to refine the data for optimal model performance. 'GICS Sector', was transformed using one-hot encoding, which is essential for incorporating nominal categorical data into our models, ensuring that the models do not impose an artificial ordinal relationship where none exists. Moreover, to address potential skewness in the financial metrics, we applied a log transformation to specific features, including 'ROA LF', 'ROE LF', 'P/E', and 'Dividend Yield'. This step is crucial as LDA models that assume normally distributed input data. The final accuracy rate of LDA is: 25%

3.1 QDA with Principal Component Analysis

QDA is closely related to LDA, where it is assumed that the measurements from each class are normally distributed. On the other hand, QDA makes no assumption that the covariance matrix of each class is identical. We instead model $p(x \mid y=k) \sim \mathcal{N}(\mu_k, \Sigma_k)$. Using similar estimates $\widehat{\mu}_k = \frac{1}{n_k}\sum_{i:y_i=k} x_i, \widehat{\Sigma}_k = \frac{1}{n_k}\sum_{i:y_i=k}(x_i - \widehat{\mu}_k)(x_i - \widehat{\mu}_k)^T$, we have the decision function

$$\hat{h}(x) = \arg\max_k \log(\widehat{\pi}_k) - \frac{1}{2}\log\left|\widehat{\Sigma}_k\right| - \frac{1}{2}\widehat{\mu}_k^\top \widehat{\Sigma}_k^{-1}\widehat{\mu}_k + \widehat{\mu}_k^\top \widehat{\Sigma}_k^{-1}x - \frac{1}{2}x^\top \widehat{\Sigma}_k^{-1}x$$

After preprocessing, we applied QDA to the transformed dataset. Since QDA allows for each class to have its own covariance matrix, it offers a more flexible decision boundary at the cost of increased risk of overfitting. To solve this problem, PCA was employed to address the high dimensionality of our dataset. We aim to strike a balance between model flexibility and the prevention of overfitting by reducing the number of features. By concentrating on the principal components, we not only reduce the complexity of the models but also potentially uncover the underlying structure of the data that is most relevant for predicting ESG ratings. The final accuracy rate of QDA after PCA is 32%.

The outcome of our analysis indicates that QDA outperforms LDA in predicting ESG ratings with a 7 percentage points higher accuracy. This suggests that the flexibility of QDA, with its allowance for each class to have its own covariance matrix, provides a better fit for the complexity inherent in the relationship between the predictors and ESG ratings. The superior performance of QDA underscores the importance of accommodating the distinct variance-covariance structures across different ESG rating predictors, which may not be fully captured by the more restrictive assumptions of LDA.

4 Support Vector Machine (SVM)

In this section, we explore the use of SVM model. The SVM model was chosen due to its robustness and versatility in multi-class classification problems. Specifically, we included one-hot encoding to convert categorical variables, such as 'GICS Sector', into a binary format suitable for SVM, ensuring no artificial ordinal relationships were imposed. Additionally, feature scaling was applied to standardize the range of our continuous features, thereby enhancing the SVM's performance by equalizing the influence of each feature.

4.1 Kernel Trick and Parameter Optimization

We utilized the powerful kernel trick to transform the data into a higher-dimensional space, allowing for the separation of data points that are not linearly separable in the original space. The kernel types explored included `linear`, `polynomial`, `RBF`, and `sigmoid`, each providing different complexities and boundaries to classify the data points effectively. Grid search optimization was employed to fine-tune the hyperparameters, particularly the penalty parameter 'C', which controls the trade-off between achieving a low-error classifier and a classifier that is less sensitive to outliers, and 'gamma', which defines the influence of individual training samples on the decision boundary. After using the grid search method, we conclude that the best parameters are {'classifier_C': 1, 'classifier_gamma': 'auto', 'classifier_kernel': 'sigmoid'}. Since the dataset is not large, there is a higher possibility of overfitting. Therefore, a lower C value is preferred in this setting.

4.2 SVM Model Evaluation

We observe that the SVM models with RBF and sigmoid kernels have the best performance, achieving an accuracy rate of approximately 32.4%. On the other hand, the SVM model with a polynomial kernel performs the worst, with an accuracy rate of 25.5%. This could be due to the fact that the true decision boundary in the dataset is not polynomial. As a result, the polynomial kernel might fail to fit the data well as it tries to map the original features into a polynomial feature space.

Regarding the best performance (32.4%), it is certainly not an ideal result. However, given the following two facts, the performance actually makes sense. First, we have a small dataset—the total number of samples in this dataset is only 506. This is extremely small in the field of machine learning, and thus will no doubt drag down the performance of any model in this setting. In fact, if a model has a really high accuracy rate in this case, it is very likely that the model is overfitting the data. Secondly, there are seven classes to predict. If we were to guess randomly, the probability of guessing the right class is around 14.3%. As a result, a performance of 32.4% is already much better than random guessing, proving the usefulness of our model.

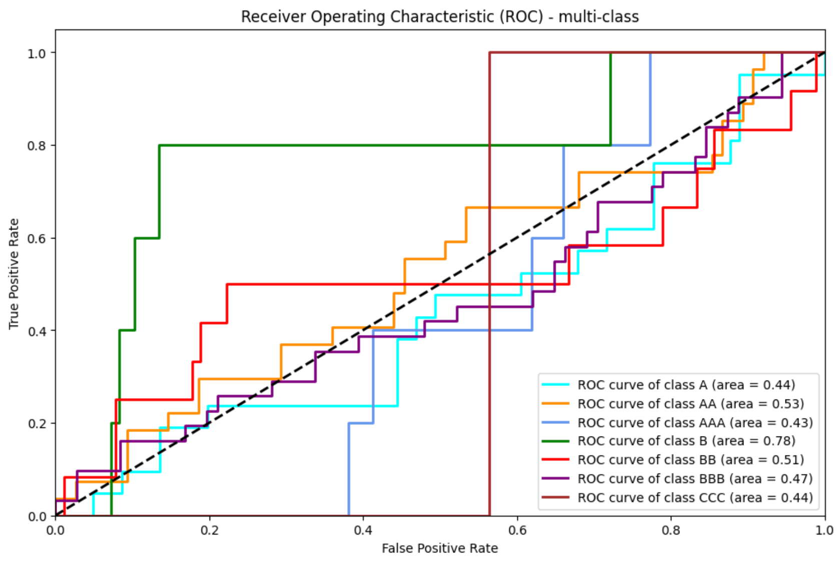


Figure 2: ROC Curves for different stock ratings using SVM Classification

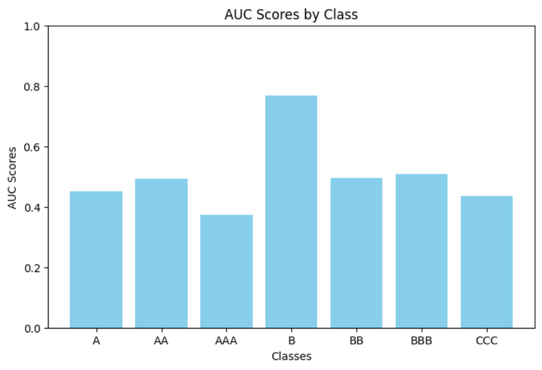
As shown in Figure 2, the ROC curves appear as staircases with pronounced steps, a characteristic likely due to a limited number of samples. With a larger sample size, we would expect the ROC curves to become smoother. Despite the step-like ROC curves, it is apparent that the model performs optimally when classifying bonds with a B rating. The performance on other bond ratings is comparable. This pattern may be a particular feature of this dataset. It suggests that for a bond to be classified as either ”too good” or ”too bad,” it must exhibit multiple specific features. Conversely, a B rating, being intermediate, might be easier to classify due to fewer stringent classification criteria.

4.3 Synthetic Minority Over-sampling Technique (SMOTE)

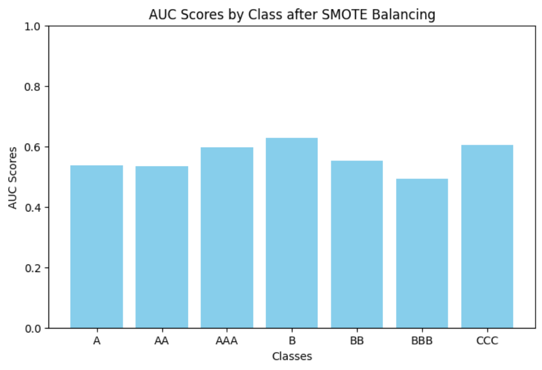
In our dataset, the class distribution for the 'MSCI ESG Rating' shows a significant imbalance. The majority class 'BBB' constitutes approximately 28.85% of the data, while the minority class 'CCC' represents only about 0.59%. This vast disparity indicates that the dataset is highly skewed, which could potentially lead to a classifier that performs well in predicting the majority class but poorly in detecting the minority classes.

To mitigate class imbalance, our model incorporates the Synthetic Minority Over-sampling Technique (SMOTE) [3]. SMOTE generates synthetic samples for underrepresented classes, avoiding overfitting risks associated with mere duplication of instances. By interpolating new samples within the feature space, it bolsters minority class representation, fostering more robust decision-making by the SVM. This balanced approach is designed to enhance the model's predictive accuracy for all classes.

In our model, we combine SMOTE method with the `class_weight` parameter in SVM classifier. This parameter automatically adjusts weights inversely proportional to class frequencies in the input data. The formula used for the 'balanced' mode is: $w_j = \frac{n}{k \cdot n_j}$ where w_j is the weight to be assigned to class j, n is the total number of samples, n_j is the number of samples in class j, and k is the total number of classes.



(a)



(b)

Figure 3: (a): AUC Scores by Class Using SVM; (b) AUC Scores by Class after Addressing Imbalance with SMOTE Using SVM.

Using the combined SMOTE method and class_weight parameter gives us an accuracy of 14.7%, which is lower than our model without using them. However, this does not this approach fails to work. It's important to note that while SMOTE helps balance the class distribution and can improve the classifier's performance on minority classes, it does not guarantee an increase in overall accuracy. In fact, as shown in Figure 3, if we compare the AUC scores for each class before and after we account for the imbalance data, the least represent class CCC (0.59% of total samples)'s AUC score increases to 0.564 from 0.436, and the second least represent class AAA (4.54% of total samples)'s AUC score increases to 0.588 from 0.431. These results show that our approach has led to a better job in predicting classes that are least represented in the total samples.

Despite the robustness of SVMs, the accuracy levels achieved suggest room for further refinement. Future work may involve exploring additional kernel functions, integrating ensemble methods, or employing advanced feature selection techniques such as feature engineering. The increase in dataset's size and the use of advanced optimization algorithms could further enhance the accuracy and generalizability of our SVM model.

5 K Nearest Neighbours (KNN)

In this section, we explore the application of the K-Nearest Neighbors (KNN) for predicting ESG ratings. Let $\mathcal{P} = \{\text{'AAA'}, \text{'AA'}, \text{'A'}, \text{'BBB'}, \text{'BB'}, \text{'B'}, \text{'CCC'}\}$ be the set of classes. Define $\mathrm{KNN}(x)$ the indices of K nearest neighbours of x based on Euclidean distance. We classify $\widehat{h}(x) = \arg\max_{c\in\mathcal{P}} \sum_{i\in\mathrm{KNN}(x)} \mathbb{1}_{y_i=c}$.

The choice of K, the number of neighbours used, is crucial in our model. We will be looking at how standard KNN performs on the ESG ratings data for different values of K. We will then explore a feature weighted KNN model using weights based on the amount of information each feature carries.

5.1 Standard KNN Model

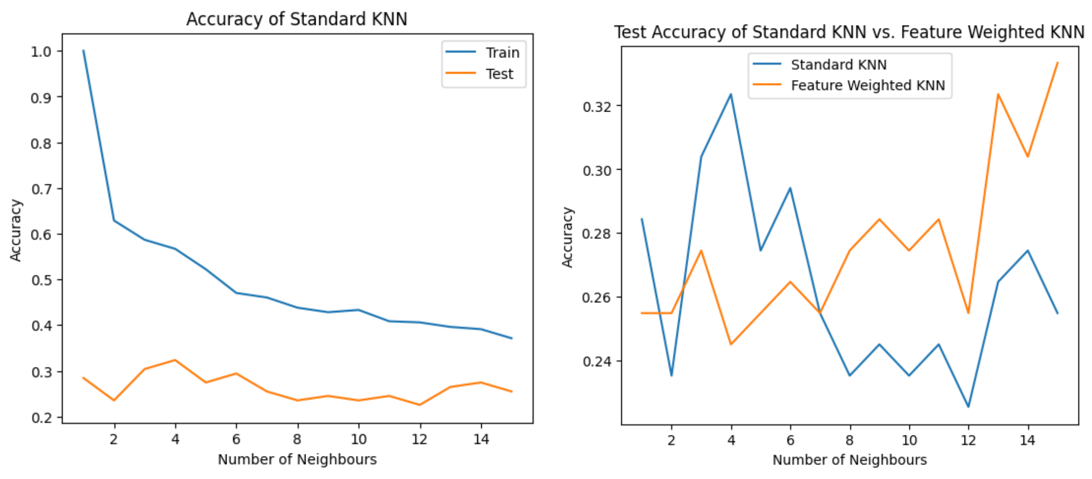
We first fit a standard KNN model to the data, experimenting with different values of K. By varying K, we observe how the accuracy on the training set and test set changes. We should expect a smaller K leads to more complex decision boundaries, potentially overfitting the training data, while a larger K may lead to oversmoothing and underfitting. To prepare the data, we use `pd.get_dummies()` to transform the categorical GICS sector feature. And since the features are clearly of different scale, we use `StandardScaler()` to scale each feature into unit variance.

The training error and test error are presented in Figure 4(a) for $K = 1, 2, \ldots, 15$. We can see a clear gap between the two errors even when we are using $K = 15$. There two possible reasons: one is that the model is still over-fitting at 15 neighbours; the other more plausible reason is that we simply don't have enough data compared to the number of features. This naturally inspires the idea of building a weighted KNN model where we assign different weights to the distance of each feature.

5.2 Feature Weighted KNN

In this section, we develop a feature weighted KNN model based using information gained to measure the importance of each feature. This method was first introduced by Dai & Xu, 2013 [2] and then explored extensively by Chen & Hao, 2017 [1]. Consider dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_i = (x_{i1}, \ldots, x_{iN}) \in \mathbb{R}^N$, $y_i \in \mathcal{P}$. Let $C_{\{i,D\}} = \{(x_j, y_j) : y_j = i \in \mathcal{P}\}$. The information on D is the estimated entropy

$$\mathrm{Info}(D) = -\sum_{i\in\mathcal{P}} \frac{|C_{\{i,D\}}|}{|D|} \log\left(\frac{|C_{\{i,D\}}|}{|D|}\right)$$



(a) (b)

Figure 4: (a). Training error vs. Test error for standard KNN with $K = 1, 2, \dots, 15$; (b). Test error of feature weighted KNN vs. standard KNN with $K = 1, 2, \dots, 15$

Now consider partition D into some subsets indicated by $D_1, D_2, \dots, D_\nu$ based on feature A. The information for feature A is

$$\text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \text{Info}\left(D_j\right) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \sum_{i \in \mathcal{P}} \frac{\left|C_{\{i,D_j\}}\right|}{|D_j|} \log \left(\frac{\left|C_{\{i,D_j\}}\right|}{|D_j|} \right)$$

And we defined the weight for feature A as the information gained

$$w_A = \text{InfoGain}(A) = \sqrt{\text{Info}(D) - \text{Info}_A(D)}$$

Subsequently the weighted distance metric is given by

$$d^w\left(x_i, x_j\right) = \sqrt{\sum_{k=1}^{n} w_k \cdot \left(x_{i,k} - x_{j,k}\right)^2}$$

In order to find a suitable partition for each feature, we look at their financial implications so that the partitions are self-explanatory. For ROE (resp. ROA), we use $\{\text{ROE}_i \le 0\}, \left\{0 < \text{ROE}_i \le \frac{\sum \text{ROE}_i \mathbb{1}_{\text{ROE}_i > 0}}{\sum \mathbb{1}_{\text{ROE}_i > 0}}\right\}$ and $\left\{\text{ROE}_i > \frac{\sum \text{ROE}_i \mathbb{1}_{\text{ROE}_i > 0}}{\sum \mathbb{1}_{\text{ROE}_i > 0}}\right\}$. For alphan and beta, we use $\{\alpha_i \le 0\}$, $\{\alpha_i > 0\}$ and $\{\beta_i \le 1\}$, $\{\beta_i > 1\}$. Fro dividend we use $\{\text{dvd}_i = 0\}$ and $\{\text{dvd}_i > 0\}$. For P/E we split between $(0, 25], (25, 50]$ and $(50, \infty)$. And finally GICS sector naturally partitions into each sector. The weights for each feature are presented in Table 1.

ROE	ROA	Alpha	Beta	P/E	Dvidend	GICS Sector
0.129	0.157	0.101	0.091	0.098	0.088	0.336

Table 1: Feature weights for feature weighted KNN

Now we are ready to apply feature weighted KNN to ESG ratings. The test errors are presented in Figure 4(b) alongside the standard KNN errors for $K = 1, 2, \dots, 15$. We see that feature weighted KNN outperforms standard KNN for $K \ge 5$. This is a decent improvement considering the model has overfitting issues for small K. This implies that putting more weights on features such as 'GICS Sector' better separates the classes. The highest accuracy rate is 33.33%. Given the very limited amount of data and that there are 7 different classes, this is not a bad result. Future work can focus on applying dimension reduction techniques along with feature weighted KNN to produce more accurate classification.

6 Conclusion and Limitation

In this report, we explore using machine learning techniques for predicting ESG ratings. The accuracy rate of each methods is summarized in Table 2. We see that Support Vector Machines (SVM), QDA, and weighted K-Nearest Neighbors (KNN) have relatively higher accuracy on test data, with 32.4%, 32.0%, and 33.3% respectively, indicating the potential of these models in handling the predictive task. The initial underperformance of QDA, attributed to its sensitivity to the feature set's high dimensionality and collinearity, saw notable improvement following the implementation of Principal Component Analysis (PCA) for dimensionality reduction. However, attempts to mitigate model inefficiency due to imbalanced data using Synthetic Minority Over-sampling Technique (SMOTE) were not successful, mainly because SMOTE introduced bias by generating new samples. Meanwhile, standard KNN achieved relatively good results with hyperparameter tuning for suitable k values, and weighted KNN further tailored the model to the data distribution without significantly increasing model complexity.

LDA	QDA	SVM	SMOTE SVM	KNN	Feature Weighted KNN
0.250	0.320	0.324	0.147	0.302	0.333

Table 2: Accuracy rate (%) of each method on classifying ESG ratings

The results emphasizes the critical role of comprehensive data preprocessing, and the strategic application of machine learning algorithms in enhancing model performance. Although we have done a rather complete exploration of standard models with certain extensions, our studies are limited in the scope of feature selection and engineering. Given the very small amount of available ESG ratings data and its high-dimensionality, future study can put more attention into the preprocessing of the data. Other more complex models such as neural network can also be experimented to provide more accurate predictions.

References

[1] Yingjun Chen and Yongtao Hao. A feature weighted support vector machine and k-nearest neighbor algorithm for stock market indices prediction. *Expert Systems with Applications*, 80:340–355, 2017.

[2] Jianhua Dai and Qing Xu. Attribute selection based on information gain ratio in fuzzy rough set theory with application to tumor classification. *Applied Soft Computing*, 13(1):211–221, 2013.

[3] Alberto Fernández, Salvador García, Francisco Herrera, and Nitesh V. Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research*, 61(1):863–905, jan 2018.

[4] Guido Giese, Linda-Eling Lee, Dimitris Melas, Zoltán Nagy, and Laura Nishikawa. Foundations of ESG Investing: How ESG Affects Equity Valuation, Risk, and Performance. *The Journal of Portfolio Management*, 45(5):69–83, 2019.

[5] Simon Polbennikov, Albert Desclée, Lev Dynkin, and Anando Maitra. ESG Ratings and Performance of Corporate Bonds. *The Journal of Fixed Income*, 26(1):21–41, 2016.