Backorder Prediction

Pradyoth Singenahalli Prabhu
Data Science
University of Massachusetts
Dartmouth, Dartmouth MA - 02747
Student ID - 02071847
psingenahalliprabhu@umassd.edu

Varun W R
Data Science
University of Massachusetts
Dartmouth, Dartmouth MA - 02747
Student ID - 02070206
wvarun@umassd.edu

Bharath Anand
Data Science
University of Massachusetts
Dartmouth, Dartmouth MA - 02747
Student ID - 02044023
banand@umass.edu

***Abstract* — This End-to-End Machine Learning case study addresses the challenge of predicting backorders for out-of-stock products in a manufacturing company. The study covers the problem statement, data collection, pre-processing, and feature engineering techniques used to prepare the data for modeling. The study evaluates and compares the performance of different machine learning models, including Logistic Regression, Random Forest, and LightGBM, and discusses hyperparameter tuning and ensemble techniques to improve model accuracy. The study presents a web application that uses the trained model to predict the probability of backorders for new products. The study emphasizes the importance of data pre-processing, feature engineering, model selection, and evaluation for accurate backorder prediction.**

Keywords—Logistic Regression, Random Forest, LightGBM, Feature engineering, Hyper-parameter tuning, Ensemble techniques, Web application.

I. Introduction

Backorder prediction is a crucial challenge faced by manufacturing companies that strive to maintain optimal inventory levels while meeting customer demand. To address this problem, many companies have started using machine learning techniques. This paper presents an end-to-end machine learning case study that focuses on predicting backorders for out-of-stock products. The study covers data collection, pre-processing, and feature engineering techniques to prepare the data for machine learning models.

The study compares the performance of different machine learning models, evaluates their accuracy using a test dataset, and provides insights into the impact of different features on backorders. Ensemble techniques and hyper-parameter tuning are also explored to improve model accuracy. The study concludes with the development of a web application that uses the trained machine learning model to predict the probability of backorders for new products. The findings of this case study provide a comprehensive guide for solving the backorder prediction problem using machine learning techniques.

Backorders can have a significant impact on customer satisfaction, business revenue, and costs. Failure to address backorders can lead to customer dissatisfaction and loss of confidence, which can ultimately result in declining revenue. However, by predicting backorders in advance, companies can manage their inventory effectively, prevent demand-supply issues, and maintain customer trust.

Therefore, the development of a machine learning model for backorder prediction can be a game-changer for businesses, helping them make informed decisions and take proactive measures to prevent backorders. This, in turn, can lead to improved customer satisfaction, increased revenue, and reduced costs. In summary, this case study provides valuable insights into using machine learning techniques for backorder prediction and highlights the potential benefits for manufacturing companies.

A. Business Objectives and Constraints

1) Objectives:

Using the provided historical data, the primary objective will be to anticipate whether a product would experience a backorder in advance.

2) Constraints:

a) *Low-latency requirement:* Here, the business can wait for the model to finish running for a few seconds or for several hours. Thus, there is no rigid requirement for latency.

b) *Interpretability is important:* The model needs to be transparent so that the business can reverse its choices.

II. Literature Review

A. Imbalanced Class Problem

Supervised learning faces the issue of imbalanced datasets, where instances of a particular class are much lower than those of other classes. This problem is common in real-world scenarios, such as medical diagnosis and remote sensing.

Standard learning classifiers primarily focus on accuracy, leading to the disregard of minority classes, treating them as noise. This challenge is further complicated by small groups of positive samples, overlapping classes, and insufficient examples of minority classes.

To tackle this issue, researchers have developed three techniques: internal, external, and cost-sensitive learning frameworks. Each method addresses the problem differently.

Internal approaches modify existing classifier learning algorithms to favor learning positive classes. External approaches adjust class distributions at the data level before applying classifiers. Cost-sensitive learning frameworks use data transformations and algorithm-level adaptation by considering costs during the training process.

To evaluate the performance of predictive learning systems developed using imbalanced methods frameworks, specific assessment metrics are necessary, which will be further explored later in this section.

B. Assessment Metrics

In order to develop an effective predictive model, it is important to carefully select evaluation metrics that are appropriate for the task at hand. When working with binary classification problems, the confusion matrix is a widely used tool for analyzing the performance of a model. The matrix displays the true positive, true negative, false positive, and false negative outcomes for each class, allowing for a detailed assessment of the accuracy of the model. The information

provided by the confusion matrix can be used to calculate a variety of different evaluation metrics, such as precision, recall, and F1 score, which can be used to compare the performance of different models. By selecting the right evaluation metrics and carefully analyzing the output of the confusion matrix, researchers can develop more accurate and effective predictive models.

TABLE I. CONFUSION MATRIX IN A BINARY CLASSION PROBLEM

	Positive Prediction	**Negative Prediction**
Positive Class	True Positive (TP)	False Negative (FN)
Negetive Class	False Positive (FP)	True Negative (TN)

Accuracy (Equation 1) is a widely used empirical measure for classification but is not suitable for imbalanced datasets. This is because accuracy fails to distinguish between correctly classified examples from different classes, leading to ambiguous results.

$$x = \frac{\mathrm{T_p} + T_n}{\mathrm{T_p} + F_n + F_p + T_n} \quad (1)$$

In imbalanced problems, specific metrics have been proposed to consider the class distribution. Precision, defined by (2), measures the accuracy of an estimator when predicting the positive class, while recall (3) indicates its ability to find all the positive samples.

$$\mathrm{P} = \frac{\mathrm{T_p}}{\mathrm{T_p} + \mathrm{F_p}} \quad (2)$$

$$\mathrm{R} = \frac{\mathrm{T_p}}{\mathrm{T_p} + \mathrm{F_n}} \quad (3)$$

III. MATERIALS AND METHODS

A. Dataset

This paper examines an imbalanced real-world dataset that is accessible on Kaggle's "Can You Predict Product Backorders?" competition. Table II provides an overview of the dataset's characteristics, including the number of attributes, positive and negative classes, number of samples, and imbalance ratio.

TABLE II. DATASET SUMMARY

Dataset	**Atts**	**Yes**	**No**	**Total**
Train	23	11293	1676567	1687861
Test	23	2688	239387	242076

Attributes are as follows:

- sku: Unique ID of product
- national_inv: Current inventory level of the component
- lead_time: Transit time for product (if available)
- in_transit_qty: Quantity of product in transit from source
- forecast_month: Forecast sales for the next 3 , 6 and 9 months.
- sales_month: Sales quantity for the prior 1, 3, 6 and 9 month time period.
- min_bank: Minimum recommended amount in stock
- potential_issue: Source issue for part identified.
- pieces_past_due: Parts overdue from source.
- perf_month_avg: Source performance for prior 6 month and 12-month period.
- local_bo_qty: Amount of stock orders overdue.
- deck_risk: Part risk flag.
- oe_constraint: Part risk flag.
- ppap_risk: Part risk flag.
- stop_auto_buy: Part risk flag.
- rev_stop: Part risk flag.
- went_on_backorder: Product actually went on backorder. This is the target value.

https://www.kaggle.com/datasets/ztrimus/backorder-dataset

B. Transformations

When working with features in a predictive model, standardization can be used to ensure that they are all scaled to the same level. However, if the original features are highly skewed and overlapping for both classes, it may be necessary to come up with new features to aid in classification. This approach can be particularly useful in dealing with imbalanced datasets where one class is significantly more prevalent than the other. By creating new features, the predictive model can better differentiate between the different classes, leading to improved accuracy and performance.

In our dataset, the distribution of raw features is highly skewed, indicating that there is a significant imbalance in the number of instances of each class. To address this, we applied standardization to scale down the features to the same level. However, we found that the original features still exhibit high skewness and overlap between the two classes. In order to improve classification performance, we decided to create new features that might better capture the underlying patterns in the data. By doing so, we aim to improve the ability of our model to differentiate between the classes and achieve better overall performance.

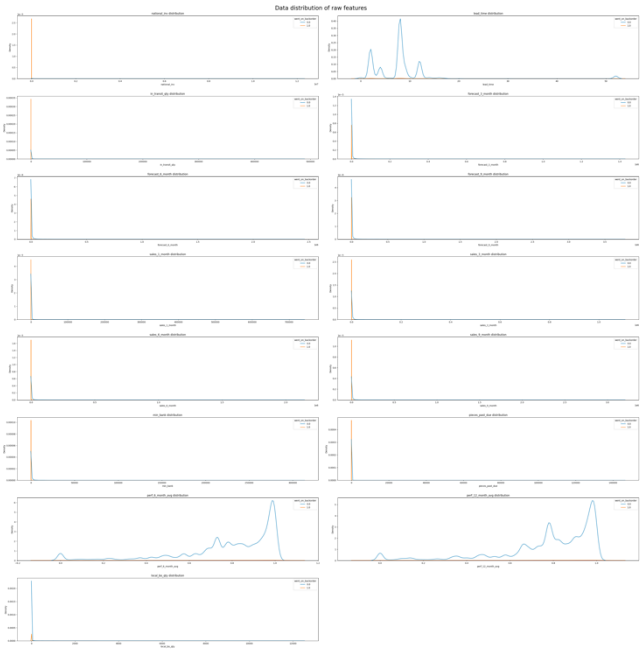

Figure 1 - Distribution of raw features

Transforming features in a dataset can help improve the performance of a machine learning model. We performed various transformations such as addition, multiplication, inverse, square, square root, and logarithmic functions on our dataset. As a result, we observed that the distributions of the transformed features were less skewed and more aligned with

a normal distribution. This normalization of the feature distributions can help to reduce the influence of outliers and improve the model's performance. Additionally, the transformation helped to remove the overlapping regions of the features for both classes, which can aid in better class separation and classification. Therefore, by performing appropriate transformations on the features, we can enhance the model's ability to generalize to unseen data and improve its overall performance.

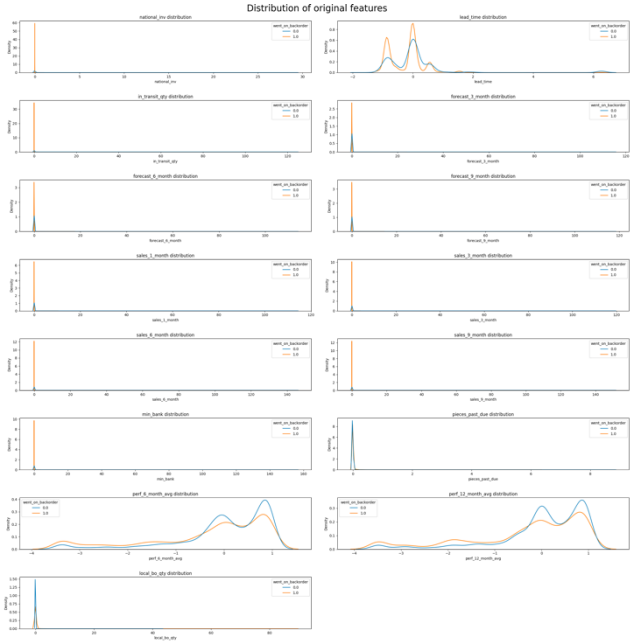

Figure 2 - Distribution of Transformed features

After analyzing the dataset, we have identified the top 10 most important features that will influence the backorder prediction. These features include "local_bo_qty_inv", "lead_time_inv", "in_transit_qty_inv", "local_bo_qty_square", "pieces_past_due_inv", "perf_6_month_avg_square", "perf_12_month_avg_square", "perf_6_month_avg_perf_12_month_avg_mult", "perf_12_month_avg_perf_6_month_avg_mult", "perf_6_month_avg_perf_12_month_avg_add", and "perf_12_month_avg_perf_6_month_avg_add". These columns have been selected based on their correlation with the target variable, as well as their ability to provide valuable information for the prediction model. By incorporating these important features, we can improve the accuracy of the backorder prediction and ensure that the model is making informed decisions based on relevant data.

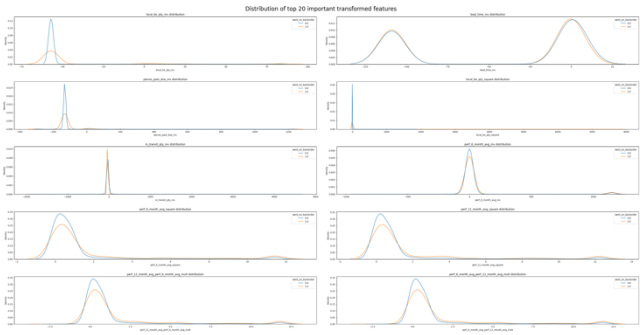

Figure 3 - Most important features that will influence the backorder prediction

Identify applicable funding agency here. If none, delete this text box.

C. Sampling techniques

Random undersampling is a technique used to balance the class distribution of a dataset by reducing the number of instances in the majority class. In other words, it involves randomly selecting a subset of examples from the majority class to match the size of the minority class. This is done to avoid bias in the model towards the majority class, which can be a common issue in imbalanced datasets.

The process of random undersampling involves the following steps: first, the number of instances in the minority class is determined. Then, the same number of instances is randomly selected from the majority class. The resulting dataset will have a balanced distribution of classes. However, it is important to note that this technique can result in loss of information from the majority class, which may impact the overall performance of the model.

Random undersampling can be effective for small datasets or datasets where the majority class has a significantly larger number of instances than the minority class. It can also be used in conjunction with other techniques such as oversampling or hybrid methods to further improve the performance of the model. It is important to evaluate the performance of the model on a separate validation set to ensure that the undersampling did not result in overfitting.

D. Model Utilization

We have utilized two different types of models.

a) Tree based model: Tree-based models are a popular machine learning technique that makes predictions by constructing decision trees. Decision trees are a series of binary splits based on input features that divide the data into smaller and smaller subsets, resulting in a tree-like structure. At each node of the tree, a decision is made based on the selected feature and its value, leading to either a subsequent node or a prediction. One benefit of tree-based models is their interpretability, as they provide a clear visual representation of the decision-making process. Moreover, tree-based models can handle both continuous and categorical data, and are robust to outliers and missing values. However, decision trees tend to overfit the training data, leading to poor generalization performance. To address this issue, ensemble methods such as random forest and gradient boosting are often used, which combine multiple decision trees to improve prediction accuracy and reduce overfitting. Overall, tree-based models are a powerful tool for a wide range of machine learning tasks and are widely used in industry and academia.

i) Decision tree: Decision trees are a popular machine learning method used in building predictive models, including backorder prediction. They work by recursively partitioning the feature space into smaller and smaller sub-regions, based on the value of a particular feature at each node, until a stopping criterion is met. In binary classification, decision trees can be used to predict whether an item will experience a backorder or not. The tree splits on a feature that is most informative in terms of separating the two classes, and the process is repeated at each node until a stopping criterion is met, such as a minimum number of samples per leaf node or a maximum depth of the tree. Decision trees can be simple to interpret and visualize, and can handle both categorical and continuous features.

However, they can be prone to overfitting and may not generalize well to new data.

b) Ensemble models: Ensemble methods combine the predictions of multiple base estimators built with a learning algorithm to enhance the generalization performance compared to a single estimator. Two main approaches are used: bagging, where estimators are trained independently and their prediction mean is assigned to the ensemble; and boosting, where estimators are sequentially built to improve upon the scores of the previous ones. By incorporating sampling techniques, ensembles can combine multiple weak estimators into robust ones that are better at distinguishing between minority and majority classes, making it a useful method to deal with imbalanced datasets.

1) Random Forest: Random Forest is a popular ensemble learning method used for building predictive models. It is a collection of decision trees, where each tree is constructed using a random subset of features and samples from the training data. Random forest algorithm is capable of handling non-linear, high-dimensional data and can capture complex interactions between the features. In the context of building a backorder prediction model, the random forest algorithm can be used to identify the most important features that contribute to the prediction of backorder status. The algorithm can also handle imbalanced data by oversampling the minority class or using class weights. Overall, random forest is a powerful and flexible algorithm that can be effective in building accurate backorder prediction models.

2) LightGBM: LightGBM is a gradient boosting framework that uses tree-based learning algorithms. It is designed to be efficient and can handle large-scale datasets with high-dimensional features. LightGBM uses a leaf-wise approach instead of a level-wise approach to grow the trees, which results in faster training times. Additionally, it uses a histogram-based approach to bin continuous features, which further improves training speed. LightGBM is also capable of handling missing data by treating it as a separate category. Another advantage of LightGBM is that it has built-in support for categorical features, making it easy to handle categorical data. It also allows for early stopping during training to prevent overfitting. Overall, LightGBM is a powerful and efficient tree-based model that can be used for a variety of machine learning tasks, including backorder prediction.

E. Model Selection

In building the backorder prediction model, the training dataset was split into a training set and a validation set in a ratio of 0.90 :0.10, maintaining the original class ratio. Additionally, a separate testing dataset was used for final evaluation. The models were optimized using an exhaustive grid search procedure, evaluating all possible parameter combinations within the defined ranges. The evaluation metrics used were accuracy and recall scores, which were recorded along with the estimator and parameter settings. This approach ensured that the models were tuned for optimal performance and could generalize well to new data. Among the models evaluated was Random Forest, which was found to have high accuracy and recall scores, making it a suitable choice for the backorder prediction task.

TABLE III. PARAMETERS AND RANGES ADOPTED FOR ENSEMBLES

Methods	Parameters	Range
Decision Tree	Criterion	"gini"
	Max_depth	"11"
Random Forest	max_depth	"120"
	n_estimators	"150"
LightGBM	max_depth	"10"
	n_estimators	"500"

F. Web Implimentation

We implemented website backorder prediction using Django framework, which provided a flexible and secure platform for building the web application and integrating the machine learning model with ease. The implementation process involved creating a Django project and app, integrating the machine learning model, and implementing the front-end user interface using HTML and CSS. The use of Django provided a flexible and secure platform for building the web application, and the integration of the machine learning model was straightforward. The final product offers an intuitive and user-friendly interface for predicting backorders, which allows the user to predict the probability of backorder for a given product and provides a reliable solution for managing inventory and meeting customer demand. Our web application allows the user to predict the probability of backorder for a single product or multiple products, as it is designed to accept a CSV file as input, making it a versatile tool for inventory management.

In addition to the aforementioned features, our backorder prediction web application also incorporated a robust pipeline that ensures that the uploaded CSV files undergo the same preprocessing steps as the training data before being fed into the machine learning model. This enables the model to provide accurate and reliable predictions on the uploaded data, taking into account any potential missing or erroneous values. The pipeline also includes feature selection techniques that help to identify the most significant features for predicting backorders. Overall, the incorporation of a pipeline ensures that our web application can efficiently and accurately process uploaded data, making it a valuable tool for inventory management.

Figure 4 - Website Home Page

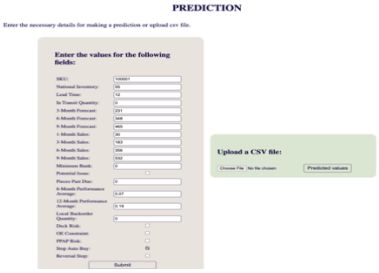

Figure 5 - Website Predict Page

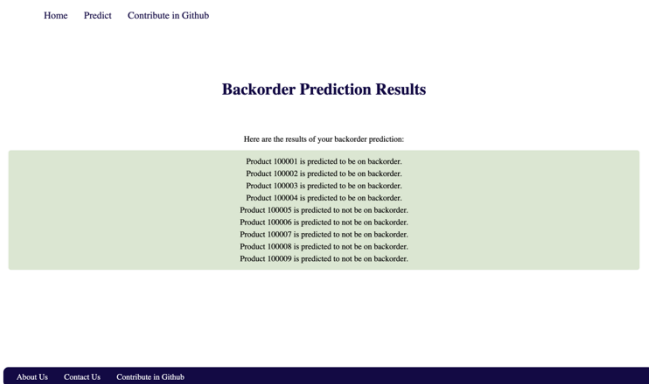

Figure 6 - Website Result Page

IV. Results

In the current study, several machine learning models were applied to develop a predictive system for inventory control. The performance of these models was evaluated based on two key metrics, namely accuracy and recall. The accuracy score represents the proportion of correctly predicted instances out of the total number of instances, while recall measures the proportion of correctly identified positive instances (i.e., items that go on backorder) out of all actual positive instances.

Among the models tested, Random Forest achieved the highest accuracy and recall scores, indicating its superior performance in predicting items that are likely to go on backorder. Decision Tree and LightGBM also performed relatively well. As the accuracies were good, we gave more consideration to recall as our main evaluation metric because our focus was on correctly identifying products that went on backorder.

TABLE IV. Model Accuracies with Recall

Model	Dataset	Accuracy	Recall	Precision
Decision Tree	Train	0.946	0.9651	0.9814
	Valid	0.8818	0.9079	0.8445
	Test	0.8633	0.8065	0.0605
Random Forest	Train	0.9981	0.9982	0.9998
	Valid	0.9194	0.9513	0.9727
	Test	0.9003	0.8077	0.1878
LightGBM	Train	0.9941	0.9954	0.9997
	Valid	0.9221	0.9513	0.9675
	Test	0.9069	0.7939	0.2052

The performance of the models was further evaluated using confusion matrices. The confusion matrix provides a visual representation of the model's performance by showing the number of true positives, false positives, true negatives, and false negatives. Fig4 shows the confusion matrix of the Random Forest. The RF model had the highest true positive rate (i.e., correctly predicted items that went on backorder) and the lowest false positive rate (i.e., incorrectly predicted items that went on backorder), indicating its superior performance in correctly predicting items that are likely to go on backorder and minimize the risk of misidentifying items that are not likely to go on backorder. Table IV gives you a detailed report of the models we have used and corresponding accuracy, recall and precision.

Overall, the results demonstrate the potential of machine learning models for improving inventory control and service levels in companies. Future research could explore the use of different classification learning ensemble and sampling-based algorithms, such as Bagging, Boosting, and Random Subspace methods. Additionally, models could incorporate misclassification costs and a cost-sensitive learning framework to enable cost curves analysis in model design.

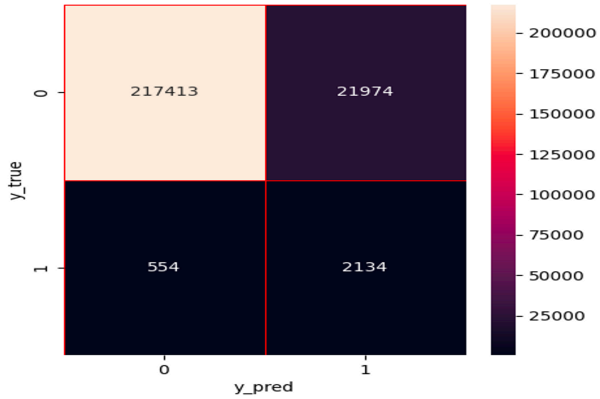

Figure 7 - Confusion Matrix

V. Conclusions

In conclusion, this paper presented a machine learning-based predictive system design for inventory control, which proved to be a feasible and effective method for identifying materials with a high likelihood of shortage in the short-term. The imbalanced class problem was addressed by using specific methods and metrics in model design, and the Random Forest algorithm demonstrated the best performance in terms of accuracy and recall scores. Future work will focus on exploring other classification learning ensembles and sampling-based algorithms, as well as incorporating misclassification costs into the framework for cost curves analysis in model design to potentially improve performance. Overall, the presented approach offers a promising solution for improving inventory management and maintaining optimal inventory levels while meeting customer demand.

References

[1] Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., … others. (2008). Top 10 algorithms in data mining. Knowledge and Information Systems, 14(1), 1–37.

[2] Ho, T. K. (1995). Random decision forests. In Proceedings of 3rd international conference on document analysis and recognition (Vol. 1, pp. 278–282).

[3] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., … Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30, 3146–3154.

[4] Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*(Oct), 2825–2830.

[5] Soledad Galli (2020). Feature Engineering for Machine Learning: A Comprehensive Overview. Train in Data. https://trainindata.medium.com/feature-engineering-for-machine-learning-a-comprehensive-overview-a7ad04c896f8.

[6] Purva Huilgol. (2020). Feature Transformation and Scaling Techniques to Boost Your Model Performance. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/07/types-of-feature-transformation-and-scaling/.

[7] Analytics Vidhya (2020). 10 Techniques to Solve Imbalanced Classes in Machine Learning (Updated 2023). https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/