Machine Learning Driven Prognostics Approach to Solar Panel Quality Inspections

Final Year Project Report

Tang Ho Yin
55695318

Department of Advanced Design and Systems Engineering
City University of Hong Kong

Content

Introduction

This report aims to thoroughly present its readers with the details of a project aiming to design a machine learning system in order to improve the solar panel quality inspection business process of quality assurance and technical compliance consulting firm specializing in solar photovoltaics, namely Sinovoltaics. This system may act as a part of a bigger decision support system (DSS), described detailly in the paper "Designing a System for Data-driven Risk Assessment of Solar Projects", which is a system to assess technical risks associated with photovoltaic (PV) projects using a data-driven approach. (Umair, Zwetsloot, Luk, Shim & Kostromin, 2021)

The solar panel quality inspection process of the solar panel assembly line or manufacturing plant performed by Sinovoltaics mainly consists of having inspectors performing a series of tests and rating the line or plant based on industrial standards. Although such practice has been effective for years, it is considered both cost- and time-consuming due to the man-hours involved in onsite inspections. Also, onsite inspections mean an extended time is needed to study a manufacturer. Machine learning has been field and utilized to achieve automation and improve business processes in many sectors. With the historical records of inspections being available, which may act as data, introducing machine learning as a means to reduce man-hours and improve the solar panel quality inspection process may redeem as feasible.

Therefore, in this paper, I will explore the possibility of utilizing machine learning techniques in the solar panel quality inspection process, with a purposed model presented, through studying the case of Sinovoltaics.

Literature Review

CRISP-DM

The Cross Industry Standard Process for Data Mining is the process that shows how data science works in real-world situations with six stages. CRISP-DM functions similarly to collecting guardrails designed by individuals in planning, organizing, and implementing data science (or deep learning) projects. The life cycle model is comprised of six stages, with axes indicating the most significant and common links between them. The steps are not sequential. Each stage in the process begins with a thorough understanding of unique goals and needs. Models of the process, such as CRISP-DM, should aid the effort and be strengthened by agile methodologies (Schröer, Kruse & Gómez, 2021). Indeed, most projects alternate between stages as necessary. Forecasting possible context changes are vital for data mining initiatives (Martínez-Plumed et al., 2017). Unexpected changes in the environment might lead to enormous extra costs and, in the worst situation, entail initiating a new task from the ground up. Data mining is a specialized area rooted in analytics, deep learning, and application systems. The advancement of strategies, methods, and systems, combined with the increased data availability and the increased complexity of multiple projects, has been so substantial over the last century that research methods have become highly significant to utilize all of the possible results effectively.

MICE and Random Forest

Multiple Imputation by Chained Equations (MICE) is a consistent and insightful technique for resolving missing information in records. MICE has developed a systematic strategy for resolving missing data in the statistical literature (Azur, Stuart, Frangakis & Leaf, 2011). The statistical uncertainties inherent in the imputations are accounted for by generating several imputations rather than one imputation. Additionally, the technique of the chained equation is very versatile since it can manage parameters of various types and complexity such as boundaries or survey skipped sequences. The approach fills in missing data in a dataset using an iterative series of prediction models (Mantas, Castellano, Moral-García & Abellán, 2018). Each iteration imputes each variable in the dataset using other variables. Repeat these repetitions until convergence seems to have been attained. The random forest method is a classification technique that extensively uses multiple decision trees. In order to make an uncorrelated forest of trees, it uses bagging and feature randomization to make each tree. This makes the majority forecast more accurate than any single tree's forecast. It is an approach for fine-grained supervised classification that utilizes bagging and random feature selection to generate a series of decision trees with controlled variance. Random forest reduces the risk of overfitting by aggregating numerous regression trees and combining the results from various trees to get more accurate estimates. A downside of random forests is that the models are complicated and hard to understand (Shah et al., 2014). However, this may not bring negative impacts on imputation. Random forests can also be skewed in some cases. Forecasts of continuous variables near the extremes of their range tend to be less extreme than those of continuous variables. A research team has done a simulation study based on CALIBER data. The study shows differences in efficiency and confidence intervals (Shah et al., 2014). MICE with a random forest gave more accurate estimates than MICE with a parametric model, despite only a slight amount. The MICE with random forest had a minor average variance between imputations. MICE's confidence intervals with parametric were approximately 93% to 95% covered. Additionally, the mean widths of confidence intervals were smaller when MICE with the random forest was used rather than parametric. However, coverage was equivalent or more extensive when MICE was combined with random forest. One of the potential reasons for the efficiency advantages of random forest MICE is that it produces adequate use of existing data by accepting predictor nonlinearities. Random forest MICE was less biased than parametric MICE in

simulations with an interaction between predictor variables, yet the substantive model. In conclusion, random forest imputation would be applied for imputing complex epidemiologic data sets with missing data.

Variable and Method	Bias[a] of Log HR	z Score for Bias[b]	SD of Estimated Log HR	Mean Length of 95% CI	Coverage of 95% CI, %	Between-Imputation Variance
Neutrophils (10^9 cells/L), per doubling						
Full data	0.002	0.43	0.158	0.564	92.2	
Complete record[c]	−0.045	−2.67	0.533	1.677	90.1	
MICE normal	−0.038	−5.15	0.232	0.883	93.4	0.0243
MICE PMM	−0.042	−5.68	0.230	0.889	93.4	0.0245
missForest	−0.266	27.72	0.303	0.781	63.2	0.0014
MICE RF 10 trees	−0.024	−4.55	0.165	0.798	97.9	0.0143
Lymphocytes (10^9 cells/L), per doubling						
Full data	−0.007	−1.23	0.155	0.526	91.6	
Complete record[c]	−0.087	−5.87	0.464	1.544	89.8	
MICE normal	0.001	0.13	0.202	0.759	93.2	0.0157
MICE PMM	0.006	0.99	0.205	0.768	92.4	0.0162
missForest	−0.190	−22.21	0.270	0.724	72.5	0.0011
MICE RF 10 trees	0.003	0.56	0.156	0.727	97.8	0.0109
Hemoglobin, per g/dL						
Full data	−0.004	−1.99	0.057	0.202	91.6	
Complete record[c]	−0.022	−3.91	0.180	0.593	90.8	
MICE normal	−0.007	−2.73	0.076	0.279	92.6	0.0019
MICE PMM	−0.004	−1.47	0.077	0.279	92.7	0.0019
missForest	−0.056	−19.96	0.089	0.255	77.3	0.0001
MICE RF 10 trees	−0.010	−5.61	0.059	0.261	97.2	0.0012

Abbreviations: CALIBER, Cardiovascular Disease Research using Linked Bespoke Studies and Electronic Records; CI, confidence interval; HR, hazard ratio; MICE, multivariate imputation by chained equations; PMM, predictive mean matching; RF 10 trees, random forest with 10 trees; SD, standard deviation.

[a] Bias was measured relative to estimates from analysis of the full data set (data set C) (Web Table 2).

[b] The *z* score is defined as the mean bias of the estimate divided by the empirical standard error from simulations, and it should lie approximately within the interval (−2, +2).

[c] Results for complete records were based on the 986 samples for which it was possible to estimate hazard ratios for all parameters.

Comparisons Between Methods of Handling Missing Data in 1,000 Samples with Continuous Variables. Missing at Random in a Pattern Similar to That of the Original Data Set (Missingness Mechanism 1). (Shah et al., 2014)

Variable and Method	Bias[a] of Log HR	z Score for Bias[b]	SD of Estimated Log HR	Mean Length of 95% CI	Coverage of 95% CI, %	% Falsely Classified[c]
Previous myocardial infarction						
Full data	0.006	1.22	0.154	0.587	94.2	0
MICE logistic	−0.013	−2.46	0.168	0.682	95.5	29.6
missForest	0.002	0.27	0.179	0.625	91.8	17.3
MICE RF 10 trees	−0.020	−4.21	0.149	0.662	97.3	28.5
Diabetes mellitus						
Full data	0.010	2.30	0.156	0.592	93.7	0
MICE logistic	0.016	3.21	0.171	0.685	95.7	32.0
missForest	0.014	2.73	0.182	0.627	90.8	19.7
MICE RF 10 trees	−0.021	−4.25	0.149	0.668	97.5	30.7
Previous stroke						
Full data	0.005	0.86	0.198	0.707	94.0	0
MICE logistic	−0.005	−0.58	0.207	0.828	95.5	17.9
missForest	0.004	0.65	0.211	0.763	92.9	8.4
MICE RF 10 trees	−0.011	−1.79	0.183	0.808	97.9	16.7
Peripheral arterial disease						
Full data	0.016	2.59	0.199	0.730	93.6	0
MICE logistic	−0.002	−0.21	0.218	0.858	94.8	15.5
missForest	0.028	4.18	0.223	0.788	91.9	7.0
MICE RF 10 trees	0.005	0.94	0.192	0.834	97.1	14.5
Heart failure						
Full data	0.015	2.47	0.191	0.653	91.7	0
MICE logistic	0.015	2.22	0.207	0.759	93.8	14.6
missForest	0.001	0.08	0.216	0.696	89.4	7.2
MICE RF 10 trees	−0.034	−5.78	0.190	0.746	95.5	13.7
Smoking status: current vs. never						
Full data	0.019	2.62	0.264	0.969	93.9	0
MICE logistic	0.023	2.65	0.292	1.092	94.0	52.4
missForest	−0.036	−3.56	0.308	1.062	91.5	35.0
MICE RF 10 trees	−0.098	−12.92	0.237	1.072	95.5	50.0
Smoking status: former vs. never						
Full data	0.011	1.66	0.247	0.908	93.6	0
MICE logistic	−0.008	−0.82	0.266	1.022	94.1	52.4
missForest	0.045	5.34	0.270	0.980	93.2	35.0
MICE RF 10 trees	−0.060	−8.81	0.212	1.000	97.1	50.0

Abbreviations: CALIBER, Cardiovascular Disease Research using Linked Bespoke Studies and Electronic Records; CI, confidence interval; HR, hazard ratio; MICE, multivariate imputation by chained equations; RF 10 trees, random forest with 10 trees; SD, standard deviation.

[a] Bias was measured relative to estimates from analysis of the full data set (data set C) (Web Table 2).

[b] The *z* score is defined as the mean bias of the estimate divided by the empirical standard error from simulations, and it should lie approximately within the interval (−2, +2).

[c] Percentage of imputed values that were different from the "true" (observed) missing value.

Comparisons Between Methods of Handling Missing Data in 1,000 Samples with Categorical Variables. Missing Completely at Random (Missingness Mechanism 2). (Shah et al., 2014).

Adaptive synthetic sampling

The main idea behind adaptive synthetic sampling is to use a weighted distribution to generate more synthetic data for difficult-to-learn minority class instances than easier-to-learn minority class examples (He et al., 2008). ADASYN is a technique that makes minority data instances easier or more challenging to learn depending on their distributions. Minority samples would be easier to learn when compared to the more artificial data made for minority class models. Additionally, the ADASYN algorithm may be customized to support applications requiring incremental learning. Most contemporary unbalanced learning techniques presuppose the availability of relevant data samples throughout the training phase. These reasons become why ADASYN is generally applied in networking, security, surveillance, Web mining, and sensor application. An adaptive ADASYN algorithm can efficiently be applied in the corresponding field and reduce bias from the imbalanced data.

Methodology

In the study of Sinovoltaics's quality inspection process, I will primarily follow a standardized process model commonly used in data mining, namely "cross-industry standard process for data mining", or "CRISP-DM" in short. Under this model, I will dissect the study of the business process into 6 major phases, namely business understanding, data understanding, data preparation, data modeling, evaluation, and deployment. The details and results of each phase will be discussed as a standalone chapter in this paper.

6 major phases in CRISP-DM (Jensen, 2012)

The source of data and information on Sinovoltaics's quality inspection process is provided by Sinovoltaics. All data used in this paper, structured or unstructured, is derived from the original data provided by Sinovoltaics.

Business Understanding

Sinovoltaics's quality inspection process primarily follows the following way.

A quality consultation on a solar panel manufacturing line or manufacturing plant will be commenced under the request of clients. As a part of the consultation, quality inspection will take place depending on the service type, which includes, audit reports of factories, production monitoring (During Production Inspection, DuPRO), pre-shipment inspection, packing loading and supervision, customized quality inspection, and solar farm inspection. (Umair, Zwetsloot, Luk, Shim & Kostromin, 2021)

The firm will dispatch inspector(s) to the manufacturing site or other premises on some occasions to perform inspections, depending on the service type.

Onsite inspections consist of conducting different tests on reference samples. The actual tests to be conducted depends on the service type, which may include product construction conformity check, visual inspection, electroluminescence (EL) imaging, mechanical data conformity check, label/serial number check, insulation test, and I-V measurement (flash test).

Commonly Seen Tests and Checks

<u>Visual Inspection</u>

Visual evaluation of the product defects. Present defects will be classified into recoverable, which consist of defects such as too tightly strapped cables or glue marks, and non-recoverable, which consist of defects such as broken cells or scratches on cells. All defects will be classified as critical, major, or minor.

Electroluminescence (EL) Imaging

Electroluminescence imaging is performed to detect micro defects such as micro cracks and inactive cells. All defects will be classified as critical, major, or minor.

I-V Measurement Test

I-V measurement test checks if the output power rating matches the quoted rating.

Mechanical Data Conformity Check

This check is performed to check the physical conformity of the product. For example: size, cable length, weight, etc.

Label/Serial Number Check

Consist of rating label and serial number rubbing test, and frame and junction box sealing test.

Insulation Test

Insulation test is conducted to test the insulation resistance of the product.

Product Construction Conformity

The check on whether the product is constructed as in the quote.

Inspection Report

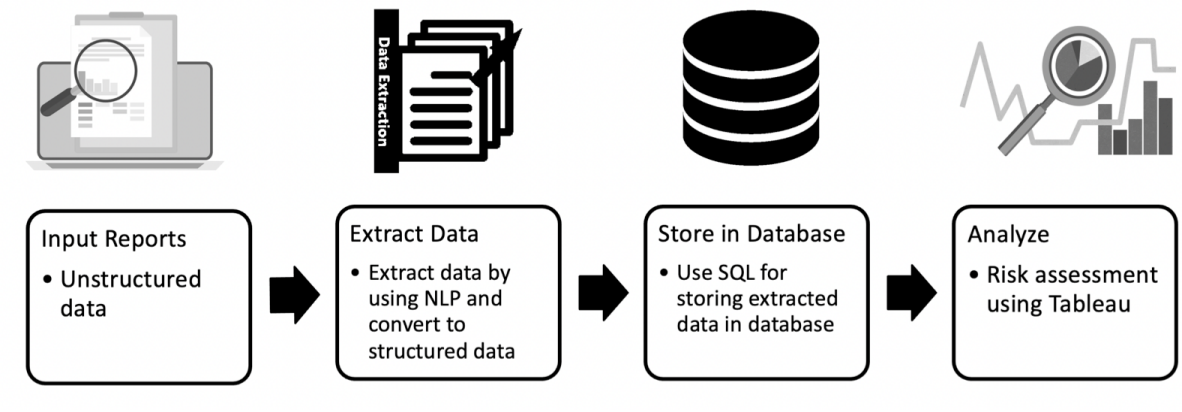
Inspector(s), based on the findings in the inspection, will rate the inspection target as pass or fail, or pass with limitations.

All the information of a quality inspection, including background information of the inspection, onsite inspection findings, and the overall inspection result will be recorded in a report, which format is to an extent standardized. The reports are subjected to data extraction as a part of the decision support system (DSS) (Umair, Zwetsloot, Luk, Shim & Kostromin, 2021), and the data extraction will serve as the primary source of data in this project.

Data Understanding

Primary Source of Data

The data used in the studies in this paper is the product of the DSS. The workflow of the DSS comes into 4 main steps. (Umair, Zwetsloot, Luk, Shim & Kostromin, 2021)



Workflow of the DSS (Umair, Zwetsloot, Luk, Shim & Kostromin, 2021)

The first step of the DSS workflow is to pre-process and input the report. The pre-processing of the reports allows them to be sorted into different formats and categories, in order to be fed into the data extraction algorithms used in the next step.

The next step is to extract data from the organized and pre-processed reports. As the reports come in different categories, different data extraction algorithms are designed for each category respectively. With the algorithms, unstructured data will be structured and be ready to be stored in the database.

The third step is to store the structured data in the SQL database, which is where the primary source of the data I used in this project is stored.

There are 303 reports stored in the database, which implies that only 303 samples can be derived from it. The detailed structure of the data stored in the database is illustrated in *Table 1*.

Proposed data solution

As the information from the reports can be divided into 2 categories, those that require an onsite inspection to obtain, and those that are already available before the onsite inspection, it is beneficial to the firm and the client if a pre-emptive estimation of the inspection result is available before the inspection is commenced.

Therefore, I propose to build a machine learning classifier to classify the overall inspection result based on information available before the inspection, with the objective to offer a prediction of the onsite inspection result with acceptable precision. It is believed this classifier will be able to act as an integrated part of the DSS and assist the firm to provide the clients an early insight into the inspection subject.

Two models, one trained with pre-available information only and one trained with full information will be built. The full information model will be acting as a side-by-side comparison to the pre-available information only model.

Data Preparation

Data Structuring

Although data in the database has already been structured once by the data extraction algorithms, further structuring is needed to fit the machine learning model that will be used.

In the progress of data structuring, I firstly determined and discard information that are not relevant or too difficult to be structured (see *Table 3*). Several processing methods (see *Table* 2) will then be used to structure the data. There are 24 features in total after structuring. In *Table 2*, the features that will be used in the pre-available information only model is marked with green, and the target class is marked with white. The full information model will use all features.

Imputation

The data point with no data in the "final result" column, which is the target of prediction, is dropped as they cannot be used to train the classifier or to be used in the test set. This resulted in a reduced sample size of 296.

Ratio of missing data can reach as high as 19.8% in some features

As there is a considerable number of null values, which represent missing data, and the sample size is considerably small (296 samples), discarding samples with null value would be a sub-optimal option, and thus data imputation is needed. As discussed in the literature review, we will utilize a random forest-based MICE algorithm for this endeavor.

The open-source Python package for random forest-based MICE, *Miceforest*, will be used. This package is an implementation of the MICE and random forest approach discussed in the literature review section. The link to its GitHub repository can be found in the appendix.

```
# Using MICE imputation to handle missing value

#convert object column to category, required by miceforest


# Create kernel.
kernel = mf.MultipleImputedKernel(
  data,
  save_all_iterations=True,
  datasets = 5
)


kernel.mice(iterations=5, boosting='gbdt', min_sum_hessian_in_leaf=0.01)
```

Implementation of Miceforest

Train-test Split

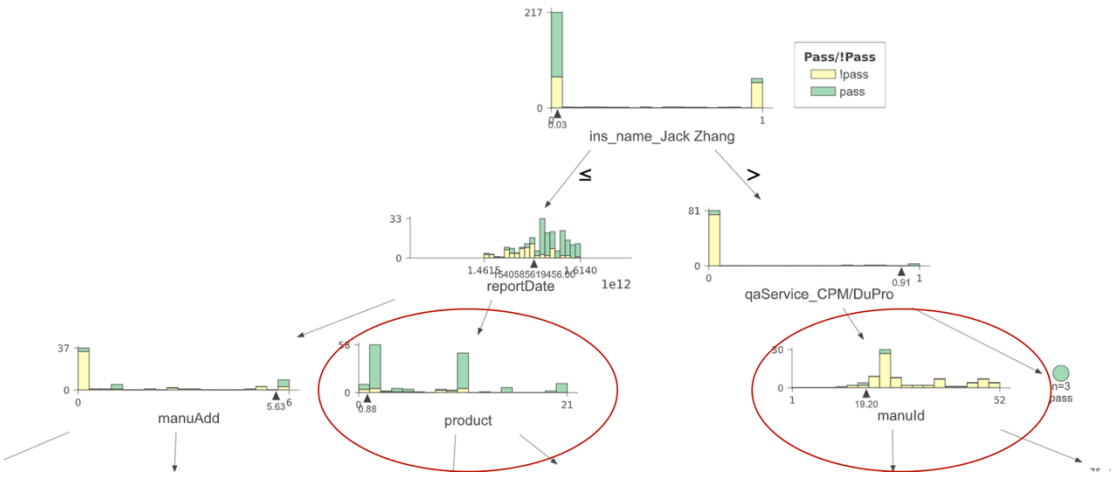
After the imputation is done, the data is then split into a train set and a test set with a ratio of 9 to 1. The reason to perform the splitting after the imputation is that imputing the test set as well can mimic the situation in the production environment where the user does not have all the input needed for the classifier and need to use the imputation kernel to impute some of the values.

Feature Encoding

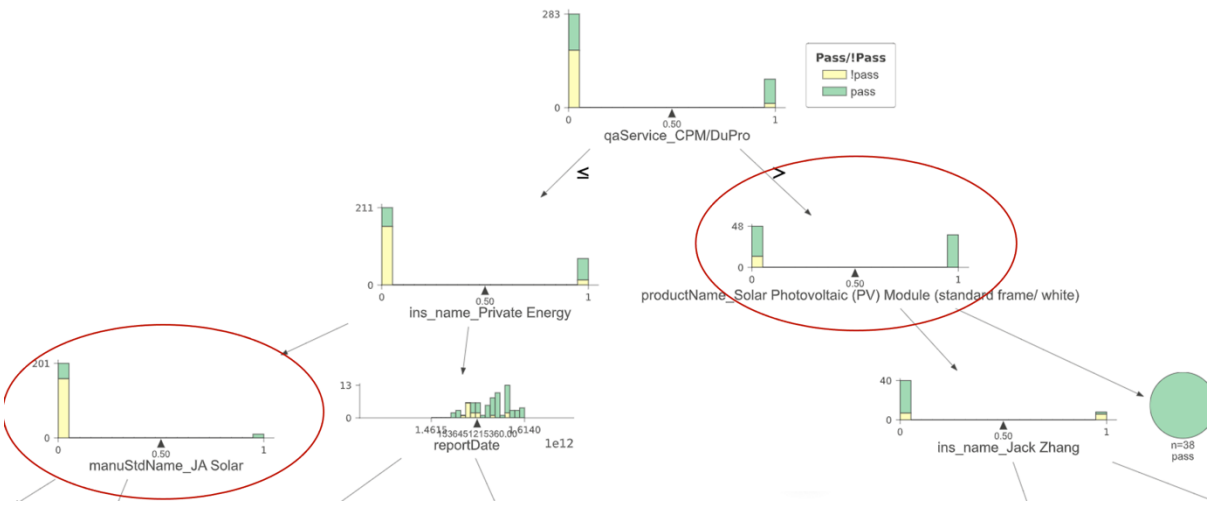
Once the imputation of missing data is completed, we should encode the categorical features into numerical features. There are 2 possible methods to perform this process, label encoding and dummy encoding. Label encoding will assign each category in a categorical feature an integer based on the alphabetical ordering. Dummy encoding will create a dummy feature for each category in a categorical feature and assign either 1 or 0 to a datapoint as its value in the feature based on whether that datapoint has this category as its value in that

feature. The characteristic of label encoding implies that it will give an order to the categories, regardless of whether the feature is ordinal or not.

Dummy encoding also gives higher interpretability to decision tree visualization due to the binary characteristic of the dummy features it creates.



Label encoded features in a visualized decision tree

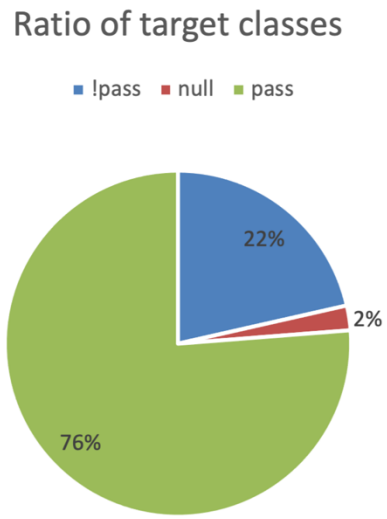


Dummy encoded features in a visualized decision tree

Based on all these factors, dummy encoding is used to encode the categorical features. The size of the feature space after dummy encoding has been expanded to 64, from 24. Although 64 features for 303 data points may cause poor performance to some machine learning models, for example, neural networks, the XGBoost model we use can offset this problem by boosting. More will be explained later in the data modeling section.

Oversampling

There is a large imbalance between the target class ‘pass’ and ‘not pass’. Out of all 303 samples in the original dataset, 231 of them belong to the pass class and 65 of them belong to the not pass class, while 7 of them are missing data.



Ratio of target classes of the original dataset

An imbalanced dataset could cause the classifier to heavily favor the majority class and overlook the minority class, if the classifier uses accuracy as its scoring method, since choosing the majority class as the prediction has a much higher probability of being correct and raising accuracy. This would lead to the inability to identify the minority class.

To overcome this challenge, we utilized adaptive synthetic sampling (ADASYN) discussed in the literature review to oversample the minority class, in order to make the dataset balance again.

Before performing the oversampling, we must first split the dataset into a train set and a test set, as the test set used for the evaluation of the model should only contain original samples and not samples artificially created through the oversampling technique. The dataset will be stratified based on the ratio of the target class to make sure the ratio is the same in

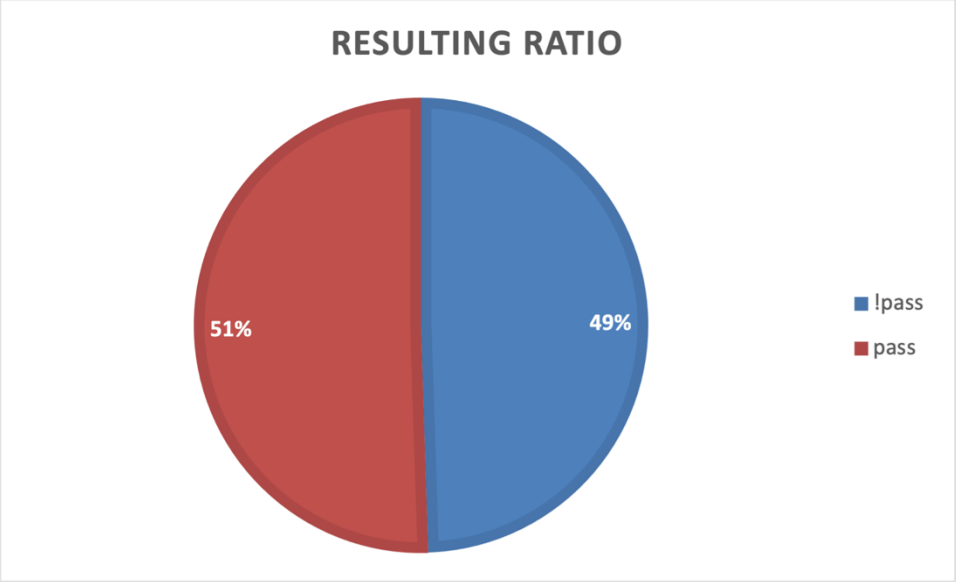
both the train set and the test set. The train set to test set ratio will be 0.85 to 0.15. Both train set and test set will then have their predictor and target variables separated.

We will utilize the open-source python package *Imblearn* to perform the adaptive synthetic sampling, on the train set only. It contains an implementation of the adaptive synthetic sampling approach discussed in the literature review section. The link to its GitHub repository can be found in the appendix.

```
# ADASYN
def myADA(X_train, X_test, y_train, y_test):
    from imblearn.over_sampling import ADASYN

    oversample = ADASYN()
    X_train=np.array(X_train)
    X_test=np.array(X_test)
    X_balanced, y_balanced = oversample.fit_resample(X_train, y_train)
    #X_test_balanced, y_test_balanced = oversample.fit_resample(X_test, y_test)
    df_X_balanced = pd.DataFrame(X_balanced)
    df_y_balanced = pd.DataFrame(y_balanced)
    #viewInExcel(df_y_balanced, "training_y.xlsx", "main")
    return X_balanced, y_balanced, X_test, y_test
```

Implementation of Adaptive Synthetic Sampling in Imblearn



Resulting class ratio after oversampling

Data Modeling

Classifier

XGBoost is used as the classifier in this project.

XGBoost in general utilized an optimized gradient boosting algorithm applied on decision trees. Gradient boosted trees ensemble multiple swallow decision trees, which are regarded as weak learners, to become a strong learner. This algorithm provides a function to rank features based on their importance rather than using them indiscriminately. (Chen et al, 2016)

It is believed that XGBoost is highly suitable for our situation given its robustness and adaptability. We will utilize the open-source python package XGBoost to implement the classification. The link to its GitHub repository can be found in the appendix.

Hyperparameters Tuning

The performance of the XGBoost classifier is highly affected by the settings of the hyperparameters. The most important hyperparameters in XGBoost include max depth, subsample, learning rate, minimum sum of instance weight needed in a child which may curb overfitting, and col-sample by tree which adds in randomness.

To discover the best set of hyperparameters, a brute-force search algorithm called grid search can be implemented. By manually entering a series of possible parameters options, the grid search algorithm will automatically find out the best combination given a scoring criterion. We will therefore implement grid search to optimize several parameters.

```
params = {
    'min_child_weight': [0.1, 0.2, 0.3, 0.5,1 ], #default 1
    'colsample_bytree': [0.01,0.05, 0.2,1],#default 1, ratio
    'max_depth': [2, 4, 6,8], #default 6
    'eta': [0.15,0.3,0.45], #default 0.3
    'subsample': [0.8,0.9,1],
    }
```

Grid of Parameters

```
In [4]: runcell(6, '/Users/alex/Desktop/ML_software/main_wrangling&modeling.py')
best params:  {'colsample_bytree': 0.2, 'eta': 0.3, 'max_depth': 8, 'min_child_weight':
0.1, 'subsample': 0.8}
```

Best Parameters for Pre-available Information Only Model

```
In [6]: runcell(6, '/Users/alex/Desktop/ML_software/main_wrangling&modeling.py')
best params:  {'colsample_bytree': 1, 'eta': 0.15, 'max_depth': 8, 'min_child_weight':
0.1, 'subsample': 1}
```

Best Parameters for Full Information Only Model

Threshold Tuning Mechanism

To better reduce the risk and increase the utility of the model, a threshold mechanism that allows users to adjust the threshold of accepting a sample as a pass is implemented. The threshold act as the minimum number that the probability of "pass", outputted by the XGBoost classifier, needs to have to be predicted as "pass" by the model.

This mechanism allows users to sacrifice "not pass" prediction precision for "pass" prediction precision, which lowers the number of false-positive and thus lower potential risk, or vice versa.

Model Evaluation and Interpretation

Performance Metrics

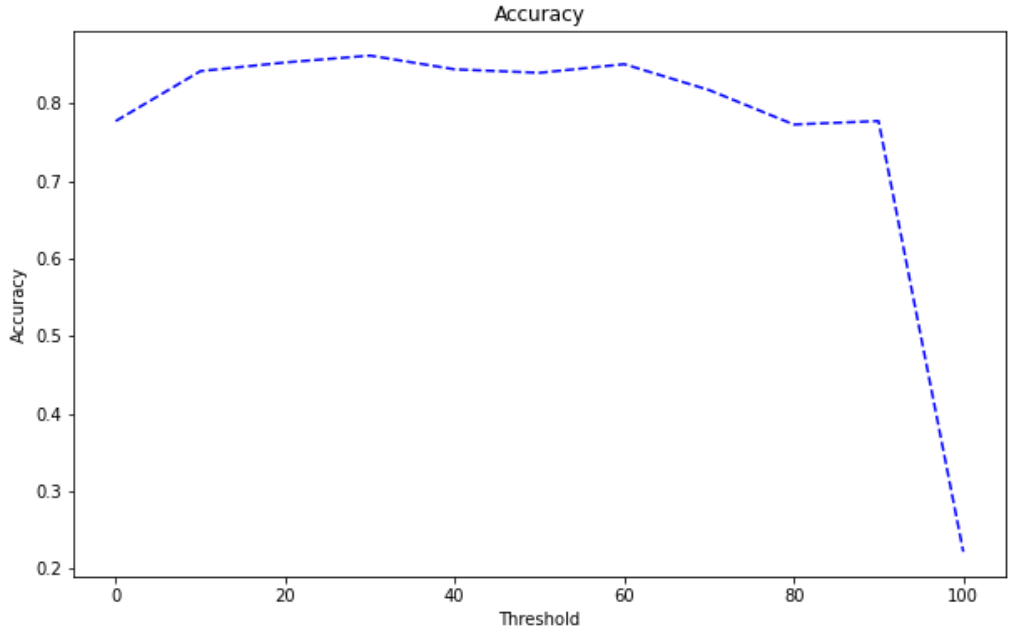
Accuracy and precision under different target classes are used to measure the performance of both models. The reason to use precision under different target classes alongside accuracy is that the precision of a class represents the probability of the prediction being correct when that class is outputted. All performances are measured with the result of models running on the test set dataset.

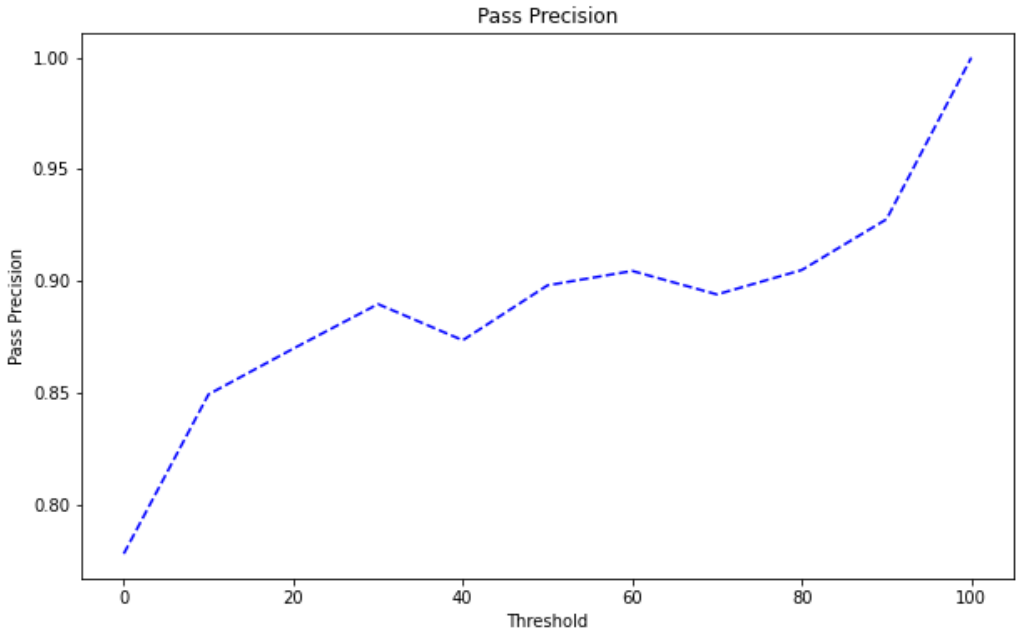
Overall Result

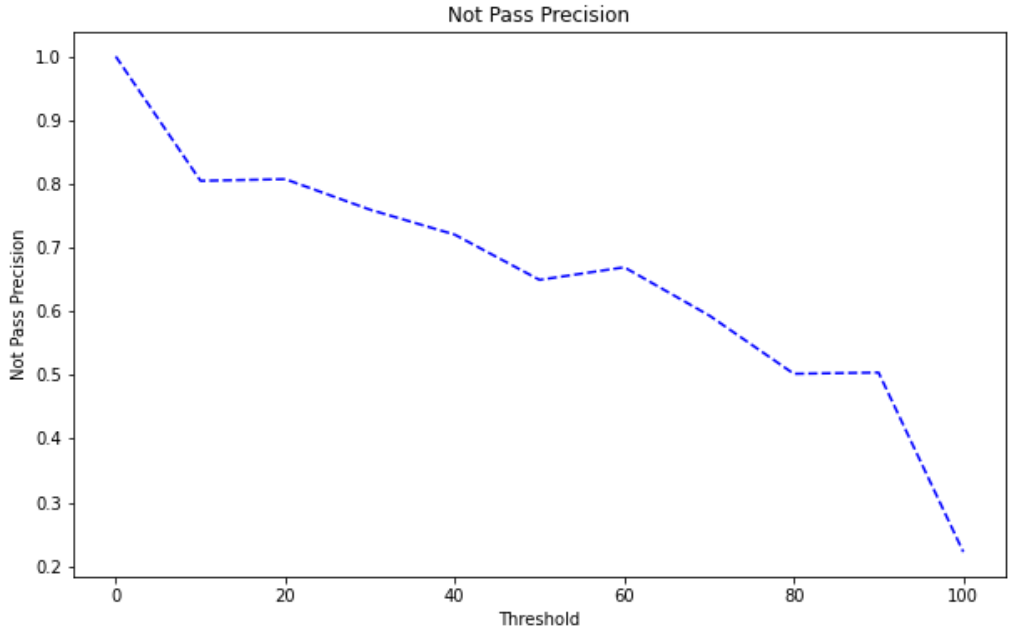
Overall speaking, the pre-available information only model achieves around 80% accuracy, depending on the threshold used. While the full information model achieves around 85% accuracy, depending on the threshold used.

Performance under Different Threshold

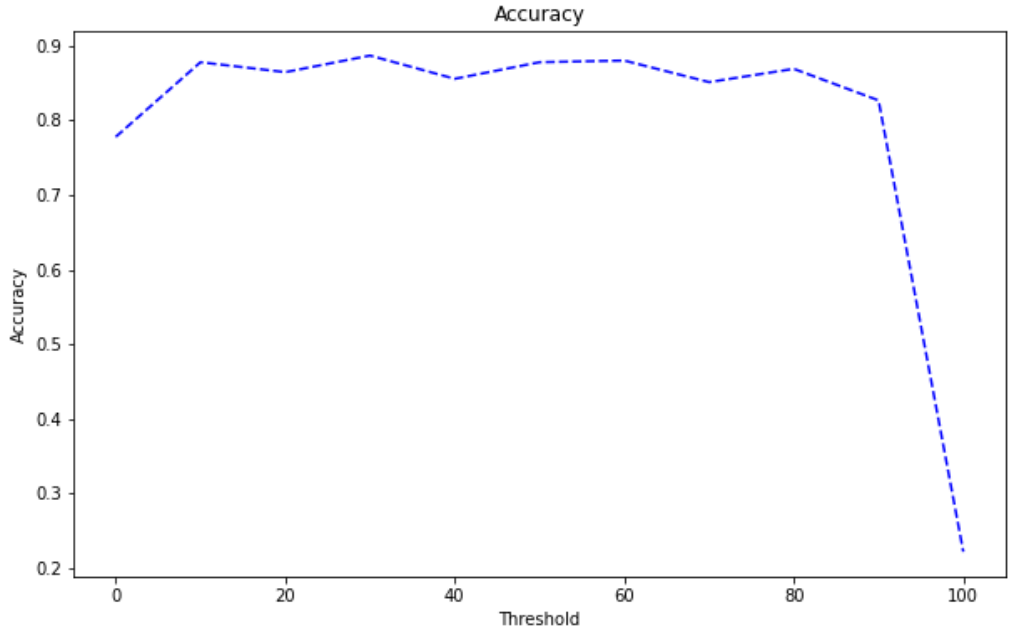
Pre-available Information Only Model

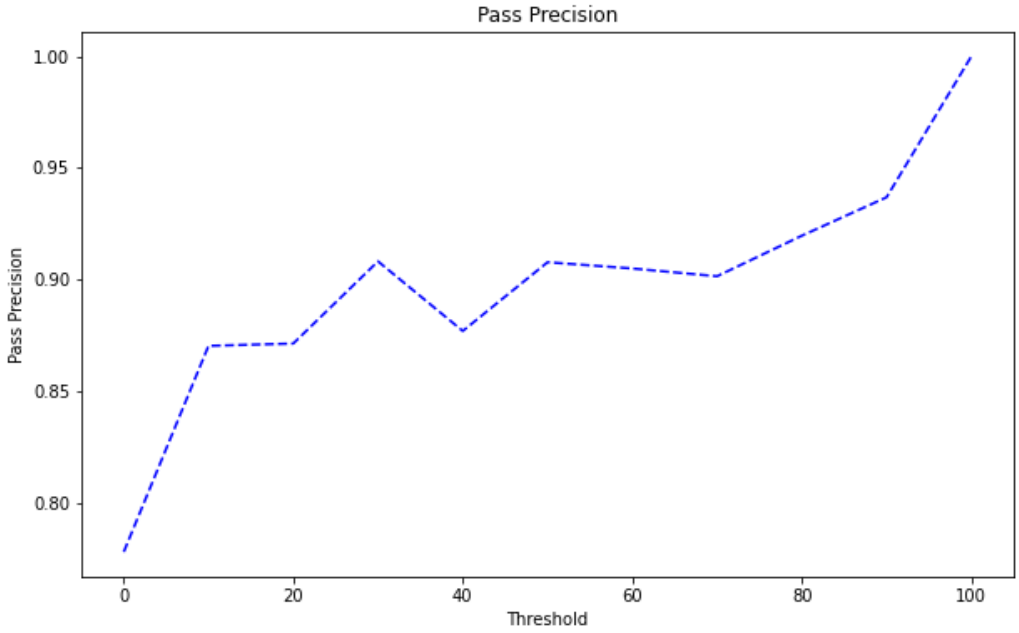


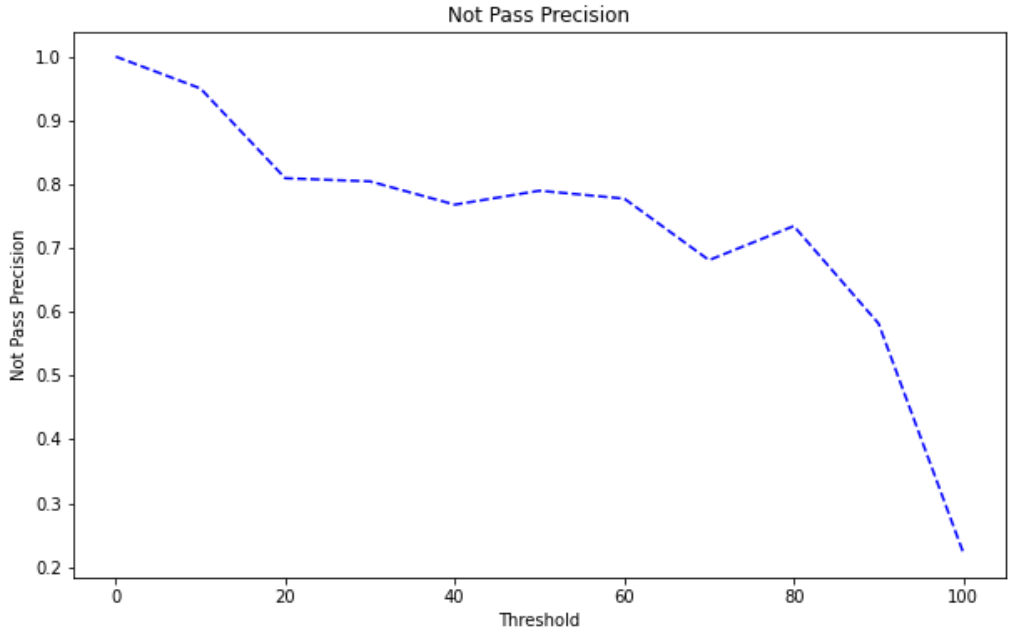




Full Information Model







As we can see in the above figures, the precision of pass and not pass classes are inversely proportional in both models, indicating a tradeoff between the precision of both classes. Nonetheless, one should be aware that even when the threshold is set to a low value, the precision of the pass class is above 0.8 for both models. This is due to the fact that there are way more data points belonging to the pass class than the not pass class.

The selection of threshold should be flexible and done according to the need of each business case. For example, if the client requires a stricter standard, a higher threshold should be used to ensure that when the inspection result is predicted to be pass, the true inspection result has a high probability of being a pass.

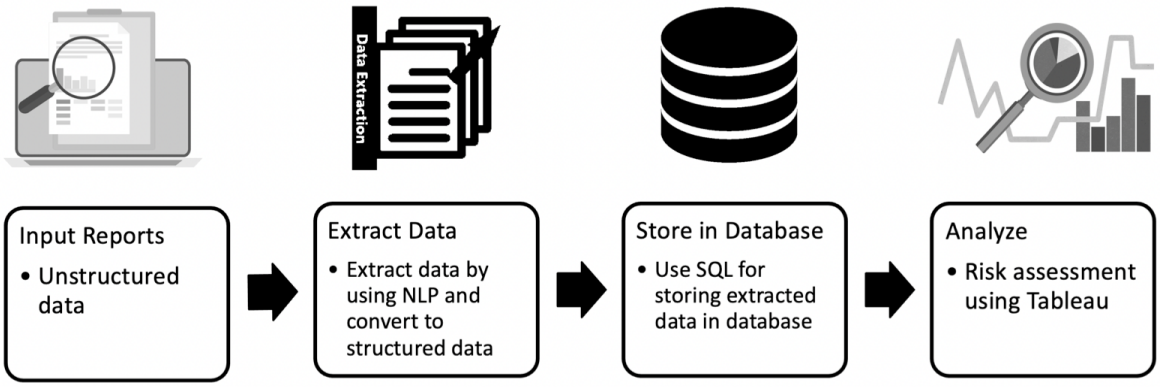
Deployment

Usage in the Firm's Business Process

With the pre-available information-only model, which has above 80% precision for both pass and not passes class under a well-selected threshold, the firm may provide a preemptive evaluation of the manufacturing order for the clients, without having to perform the inspection.

Place in the DSS

As the final step of the DSS workflow is analysis, this machine learning system can act as an integrated part of the DSS as a tool to be used in the analysis step, on top of the risk assessment using Tableau. (Umair, Zwetsloot, Luk, Shim & Kostromin, 2021)



Workflow of the DSS (Umair, Zwetsloot, Luk, Shim & Kostromin, 2021)

Additions Needed for Deployment

For the sake of user-friendliness, user-interface is needed for inputting predictor variables, viewing outputted results, and setting thresholds. The user-interface, however, is

not included as a part of this project, and needs to be built in the future, for the fruits of this project to be fielded for business use.

Maintenance and Monitoring

Constant evaluation of the whole machine learning system is needed after deployment. Any significant drops in model performance should be recorded, and full audit and remediation action shall take place.

Challenges and Risk

Adaptability to changes in feature space

Due to the nature of the data that the model is trained upon, the model may not perform very well when facing novelty in the feature space, for example, *a new manufacturer*, or *a new client*. Therefore, one should use the model with caution when novelty is present in any one of the features.

Conclusion

In this project, I have explored the possibility of applying a machine learning approach to the solar panel quality inspection process through studying the case of Sinovoltaics. A machine learning model with a good accuracy and precision rate was built to predict the solar panel quality inspection result pre-emptively. This model could be useful to enhance Sinovoltaics' business process.

I believe that the machine learning approach could be applied to other solar panel quality inspection processes with an amendment to the machine learning system. Further research on the possibility and detail of it should be done in the future.

Appendix

Reference

Azur, M. J., Stuart, E. A., Frangakis, C., & Leaf, P. J. (2011). Multiple imputation by chained equations: what is it and how does it work? *International Journal of Methods in Psychiatric Research*, 20(1), 40–49. https://doi.org/10.1002/mpr.329

Chen, Tianqi & Guestrin, Carlos. (2016). XGBoost: A Scalable Tree Boosting System. 785-794. 10.1145/2939672.2939785.

Haibo He, Yang Bai, Garcia, E. ., & Shutao Li. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 1322–1328. IEEE. https://doi.org/10.1109/IJCNN.2008.4633969

Kenneth Jensen. (2012). Retrieved 21 Mar. 2022, from https://commons.wikimedia.org/wiki/File:CRISP-DM_Process_Diagram.png.

Martínez-Plumed, F., Contreras-Ochando, L., Ferri, C., Flach, P., Hernández-Orallo, J., Kull, M., Lachiche, N., Ramírez-Quintana, M. J. (2017). CASP-DM: *Context Aware Standard Process for Data Mining* (pp, 1-18).

Mantas, C. J., Castellano, J. G., Moral-García, S., & Abellán, J. (2018). A comparison of random forest based algorithms: random credal random forest versus oblique random forest. *Soft Computing (Berlin, Germany)*, *23*(21), 10739–10754. https://doi.org/10.1007/s00500-018-3628-5

Schröer, C., Kruse, F., & Gómez, J. M. (2021). A Systematic Literature Review on Applying CRISP-DM Process Model. *Procedia Computer Science*, 181, 526–534. https://doi.org/10.1016/j.procs.2021.01.199

Shah, A. D., Bartlett, J. W., Carpenter, J., Nicholas, O., & Hemingway, H. (2014). Comparison of Random Forest and Parametric Imputation Models for Imputing Missing Data Using MICE: A CALIBER Study. *American Journal of Epidemiology*, 179(6), 764–774. https://doi.org/10.1093/aje/kwt312

Z. Umair, I. M. Zwetsloot, L. K. M. Marco, J. Shim and D. Kostromin, "Designing a System for Data-driven Risk Assessment of Solar Projects," IECON 2021 – 47th Annual Conference of the IEEE Industrial Electronics Society, 2021, pp. 1-6, doi: 10.1109/IECON48115.2021.9589414.

Tables

Table 1: Structure of The Data

Table	Column	Corresponding Information Represented	Remarks/ Understanding
report	report_id	unique assigned ID to each quality inspection report	
	report_name	a given name to each quality inspection report	
	report_date	the date which the report is written	
	product	the name of the product involved	
	manu_id	unique assigned ID of the manufacturer involved	
	manu_nonstd_name	original (non-standardised) name of the manufacturer involved	

	manu_addresss	addresss of the manufacturer involved	
	inspection_date	date of the inspection	
	order_quantity	quantity of product ordered by the client	
	standards_evaluation	standards and evaluation basis of each test	
	final_result	overall inspection result, pass / fail / pass with limitation rating mentioned earlier	
	client_id	unique assigned ID of client	
	client_nonstd_name	original (non-standardised) name of the client	
	refsamples_id	unique assigned ID of the origin of the reference sample used	
	final_remarks	final remarks by the inspector(s) on the report	
manufacturer	manu_id	*same as above column with the same name*	
	manu_std_name	standardized name of the manufacturer	
client	client_id	*same as above column with the same name*	
	client_std_name	standardized name of the client	
report__el_inspection	report_id	*same as above column with the same name*	
	el_inspection_id	unique assigned ID of EL inspection check items	
	el_result	result of the EL inspection check item	Includes major and minor failures or none, one check item may have both major and minor failures

report__label_frame_check	report_id	*same as above column with the same name*	
	lf_check_id	unique assigned ID of label/serial number check and frame gap test check items	
	lf_check_result	result of the label/serial number check and frame gap test check item	Includes conformed or not conformed
report__mod_fact_cert	report_id	*same as above column with the same name*	
	mod_fact_cert_id	unique assigned ID of module certifications and factory certifications of the product or factory	
report__performed_test	report_id	*same as above column with the same name*	
	test_id	unique assigned ID of all tests available	
	test_result	result of the corresponding test	Includes conformed, not conformed, partly conformed or not applicable (meaning the tests is not conducted)
report__product_cons_conf	report_id	*same as above column with the same name*	
	pcc_id	unique assigned ID of product construction conformity check items	
	pcc_result	result of product construction conformity check items	Includes approve and not applicable
report__qa_service	report_id	*same as above column with the same name*	
	qaservice_id	unique assigned ID of type of quality assurance service requested	
report__sino_hr	report_id	*same as above column with the same name*	

	sino_hr_id	unique assigned ID of each inspector	There can be more than one inspector involved in one inspection
report__visual_inspection	report_id	*same as above column with the same name*	
	vi_id	unique assigned ID of visual inspection check items	
	vi_result	result of visual inspection check items	Includes major and minor failures or none, one check item may have both major and minor failures
sino_hr	sino_hr_id	*same as above column with the same name*	
	sino_hr_name	name of the inspector	
visual_inspection	vi_id	*same as above column with the same name*	
	vi_type_id	unique assigned ID of 2 types of visual inspection	
	description	detailed description of the visual inspection check items	
vi_type	vi_type_id	*same as above column with the same name*	
	vi_type	2 types of visual inspection	Includes recoverable/ repairable and not recoverable/ repairable
ref_samples	ref_samples_id	*same as above column with the same name*	
	ref_samples_by	origin of the reference samples	
qaservice	qaservice_id	*same as above column with the same name*	
	qaservice_name	name of the qa service	according to Sinovoltaics, service type

			FPSI/PSSI/SPSI and CPM/DuPro can be grouped into 2 types
product_cons_conf	pcc_id	*same as above column with the same name*	
	description	detailed description of the product construction conformity check items	
performed_test	test_id	*same as above column with the same name*	
	description	detailed description of the performed test	Includes tests mentioned earlier, basic checks like quantity check, and some rarely performed tests
mod_fact_cert	certification_id	*same as mod_fact_cert_id*	
	certification_name	name of the certification	
	certification_type_id	type of the certification	
certification_type	certification_type_id	*same as above column with the same name*	
	certification_type_name	define whether it is module or factory certification	
label_frame_check	lf_check_id	*same as above column with the same name*	
	description	detailed description of the label/serial number check and frame gap test check item	
el_inspection	el_inspection_id	*same as above column with the same name*	
	description	detailed description of the EL inspection check item	
cell_tech	report_id	*same as above column with the same name*	
	cell_tech	cell technology used	
aql	report_id	*same as above column with the same name*	

	test_name	name of the accepted quality level test	
	critical_accept	counts or percentage of different degree of acceptance	
	critical_reject		
	major_accept		
	major_reject		
	minor_accept		
	minor_reject		
	remarks	remarks	

Table 2: Processing Methods of Features

Feature	Type	Processing Method
Report Date	numerical data	turning report_date into UNIX time
Product Name	categorical data	not processed
Manufacturer Address	categorical data	acquiring the country of origin of each manufacturer from Sinovoltaics and pairing it with manu_id
Manufacturer Name	categorical data	joining report and manufacturer
Final Result (target)	categorical data	Retaining pass, assigning all the rest to a “not pass” class
QA Service	categorical data	As there can be more than one QA service per sample, this feature is processed by dummy encoding first followed by grouping by report ID.
Cell Tech	categorical data	similar to QA Service, dummy encoding and grouping.

Inspectors	categorical data	similar to QA Service, dummy encoding and grouping.
Client Name	categorical data	joining report and client
EL inspection	numerical data	generalized into major, minor and pass percentage by calculating the percentage of check items with major, minor, or no failures respectively.
Visual Inspection (recoverable)	numerical data	separating visual inspection by recoverable and non-recoverable, and generalized into major, minor and pass percentage by calculating the percentage of check items with major, minor, or no failures respectively.
Visual Inspection (non-recoverable)	numerical data	
Construction Conformity	numerical data	generalized into approve and not applicable by calculating percentage of check items marked as approve and not applicable respectively
Performed Test Result	numerical data	generalized into conformed, not conformed, partly conformed, and not applicable by calculating percentage of test marked as conformed, not conformed, partly conformed and not applicable respectively
Label Conformity	numerical data	generalized into conformed and not conformed by

		calculating its respective percentage
Module Certification	numerical data	generalized into calculating the count of all module certification a sample has

Table 3: Discarded Information

Information	Reason to discard
Report ID	not relevant
Report name	not relevant
Inspection date	not in consistent format, it can be from a single date to a range of days
Order quantity	not in consistent unit, recorded in unit of pieces and power output (MWp)
Standards and evaluation basis	a long sentence that is hard to format
Source of reference sample	an overwhelming majority of samples are from the manufacturer or not recorded, with only 14 out of 303 samples being exceptions
Final remarks	a long sentence that is hard to format
Accepted quality test	not in consistent unit, recorded in both counts and percentage
Factory Certification	not recorded in database

Important Python Package Used

Imblearn: *https://github.com/scikit-learn-contrib/imbalanced-learn*

Miceforest: *https://github.com/AnotherSamWilson/miceforest*

XGBoost: *https://github.com/dmlc/xgboost*