IIT BOMBAY TECHFEST 2022

WELDRIGHT

Team HA226956

TEAM MEMBERS

Saad Ahmed

Sejal Kotian

Safdar Inamdar

Akshita Mittal

PROJECT FLOWCHART

Understanding of PS and Welding Process

Analyzing and Preprocessing of the Dataset

Implementing various models

Tackled issues and ensembled the models

Building our final ML and Business model

Understanding of the PS

- We aim to build an ML model that effectively predicts defects in the welds given their process parameters so that the loss incurred by the industry due to the defective welds is avoided.
- The end goal of this competition is to enable the Godrej Aerospace team to produce defect-free products every time.

Understanding of the Dataset

	Employee Code	Machine	Production	Order Operation No	Date	Time	Current	Humidity	Temperature	Flow	Job Temp	Voltage	Defect
0	(Office id of employee)	(Welding machine name)	(Project order number)	(This captures the activity to be performed by...	(date of activity)	(Timestamp for the activity)	(In Ampere)	(Relative humidity in %)	(in degree celsius)	(in liters per min (LPM))	(in degree celsius)	(in volts)	(if defect occurred)
1	382617	TWLD23	E15002965	240	2022-09-10 00:00:00	7:32:28:527	1.13	74	23	0.01	29.3	15.2	No Defect
2	382617	TWLD23	E15002965	240	2022-09-10 00:00:00	7:32:29:40	1.82	74	23	0	29.3	0	Tungsten Inclusion
3	382617	TWLD23	E15002965	240	2022-09-10 00:00:00	7:32:29:677	2.91	74	23	0	29.3	0	No Defect
4	382617	TWLD23	E15002965	240	2022-09-10	7:32:30:166	0.04	74	23	0	29.3	0	No Defect

Understanding the DataSet

Of these, the **target column** is the '**Defect**' column which consists of the three classes: '**No Defect**', '**Tungsten Inclusion**', and '**Porosity**'.

Since the data is taken from industrial applications, it is a highly imbalanced dataset with

- **No Defect: 99.3%**
- **Tungsten Inclusion: 0.56%**
- **Porosity: 0.14%**

This is a **Multiclass Imbalanced Data Classification Problem.**

Problems

Some of the problems that were faced and tackled by us while working on the PS

Class Imbalance

Overlap between classes

Class Imbalance

The class imbalance causes the model to be **highly biased towards the majority class** thus performing poorly to give a good result for the minority class.

Since the minority classes(Porosity and Tungsten Inclusion) hold greater value than the majority class, the occurrence of a **false negative** incurs a greater **penalty** than a false positive.

Consequence:

When the bias favors the majority class, **the defective weld is classified as no defect.**
This can cause a **huge loss for the industry!!**

Overlap Between Classes

Problem was majorly with the classes porosity and no defect.
For the pairs nd-ti and p-ti, the classification was performed as follows:

For P-Ti Pair:

BalancedRandomForestClassifier with SmoteTomek upsampling

sampling ratio of 0.75 for p-ti pair

Obtained Recalls of 0.89 and 0.93 respectively

For Nd-Ti Pair:

RandomForestClassifier with **RandomUnderSampling - sampling ratio of 0.15 with**

SmoteTomek - sampling ratio of 0.85

Obained Recalls of 0.84 and 0.87 respectively

Dealing with Overlap

For nd-p pair

Recall - 0.55

Unsatisfactory!!

Either of the one got biased in most algorithms like Random Forest, SVM

Root of the problem = high level of overlap

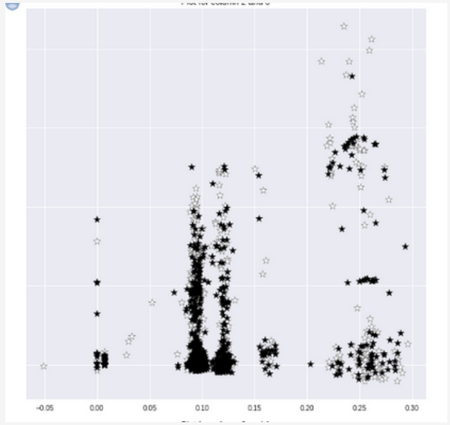
Exploratory Data Analysis

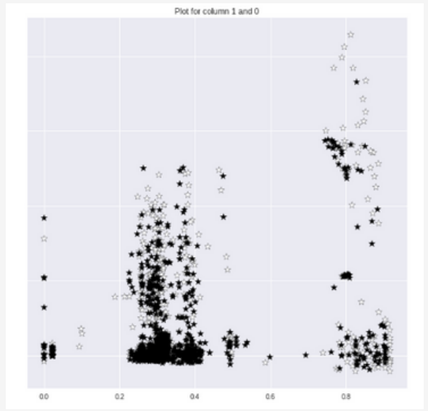
Before going deep into algorithms, we tried to perform **EDA** using various plotting techniques like **scatter plotting** and **pair plotting**.

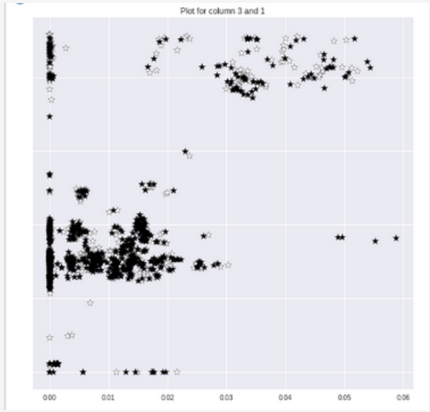
In addition to that, we also did some **dimensionality reduction** techniques like **PCA (Principal Component Analysis)**, **LDA (Linear Discriminant Analysis)** and **T-SNE (T-distributed Stochastic Neighbourhood Embedding)**.

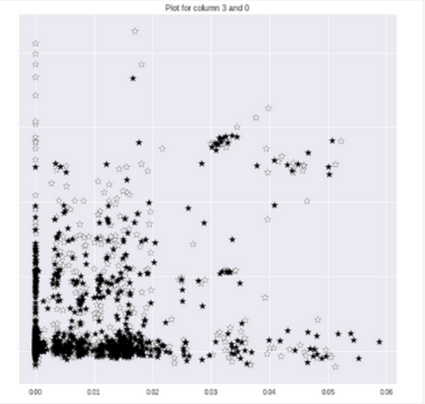
EDA – Scatter Plotting

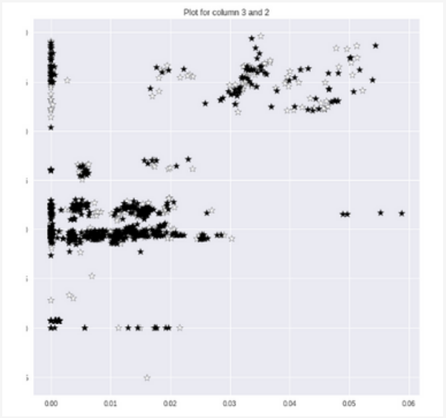
Scatter Plotting

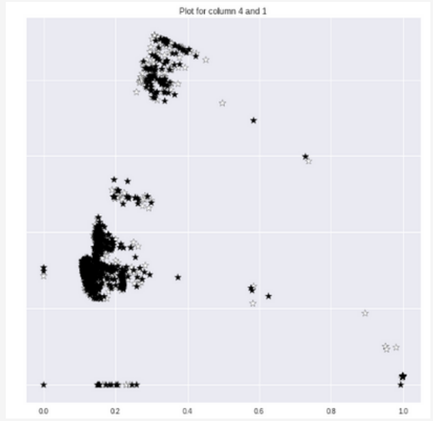


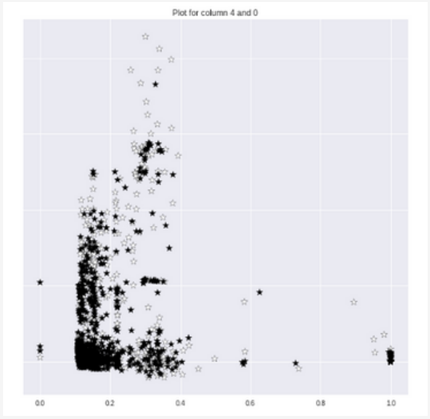


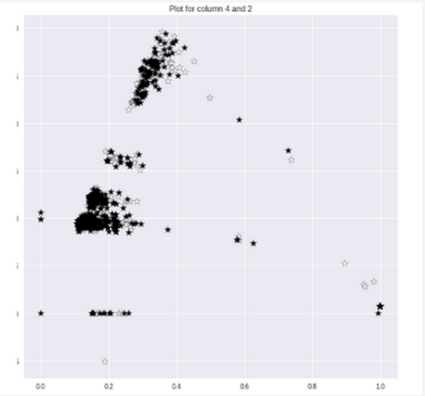






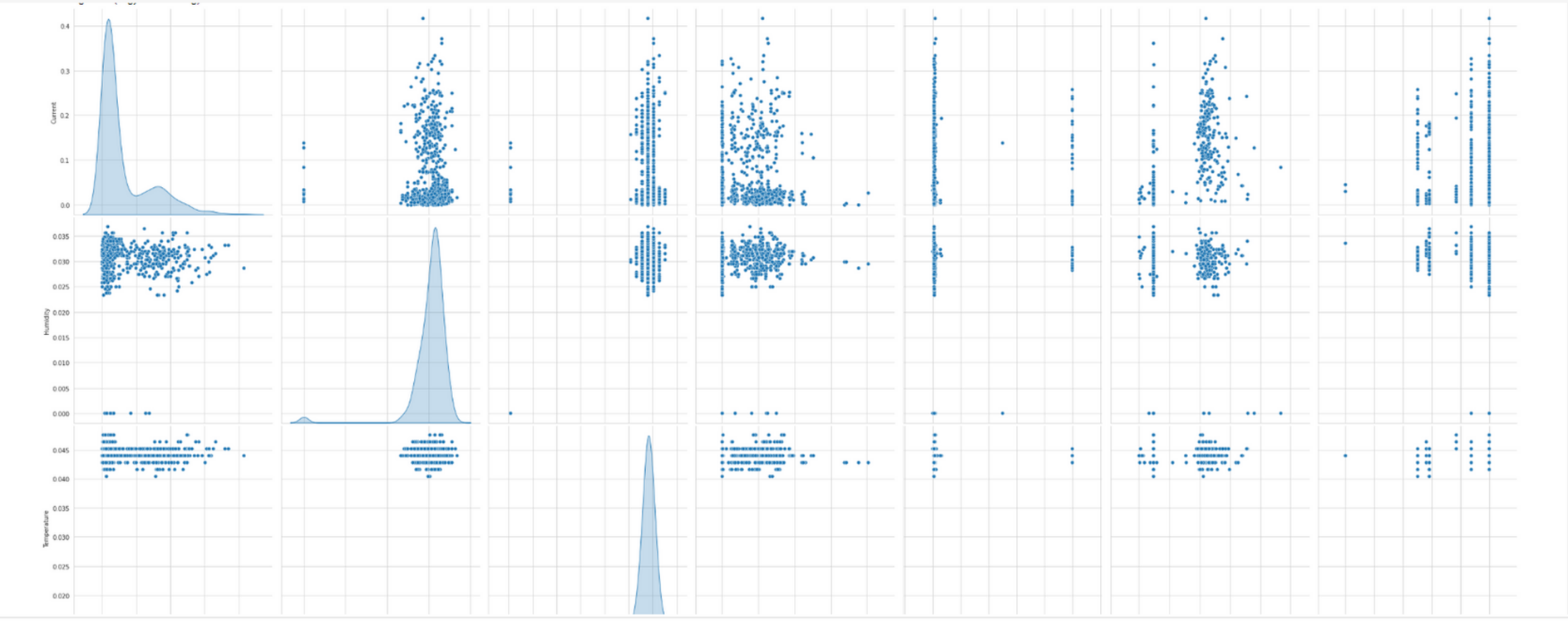






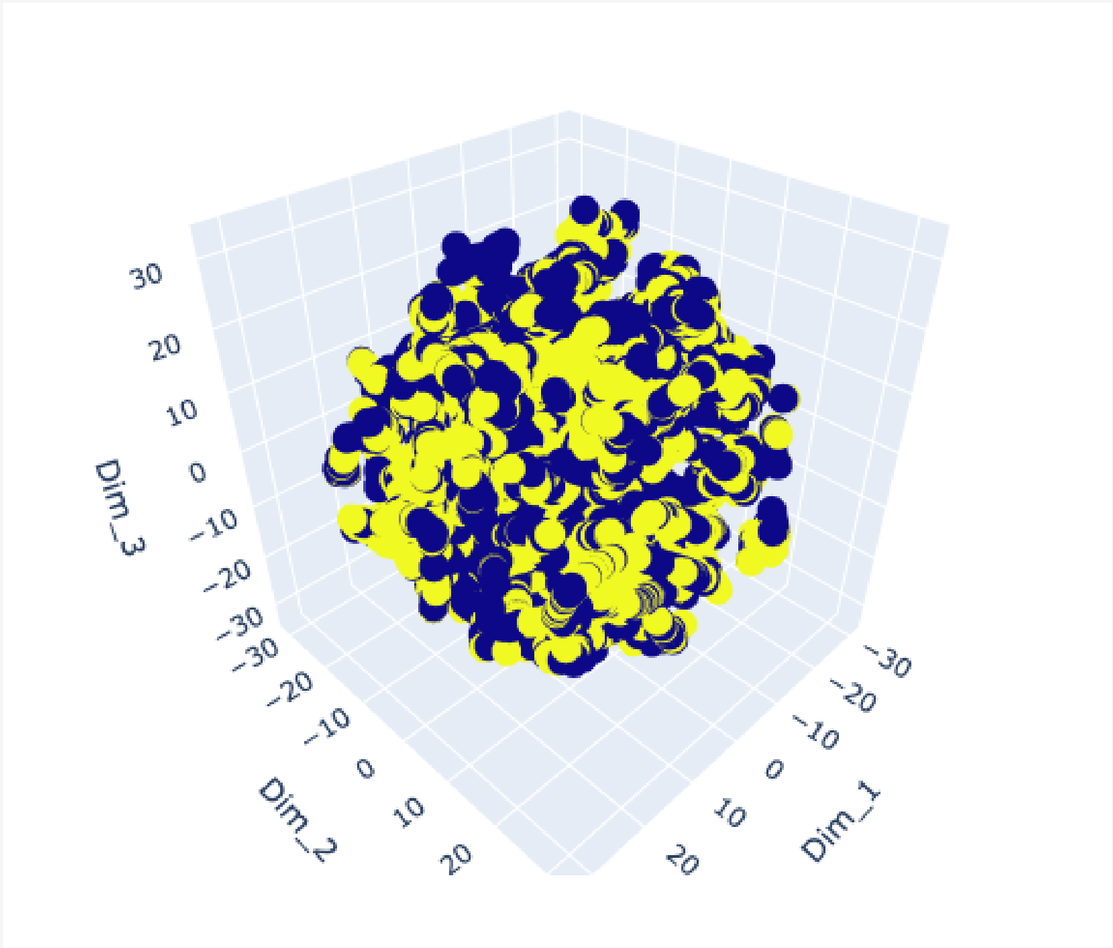
EDA – Pair Plotting

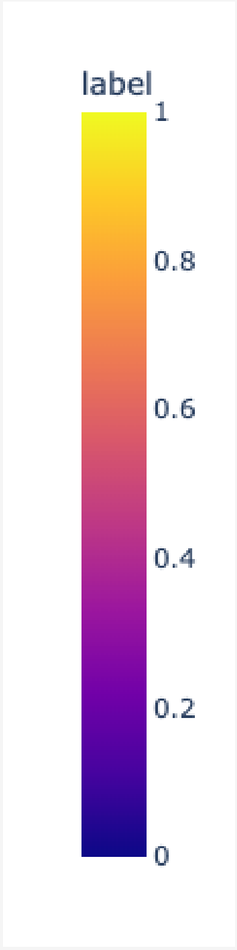
Pair Plotting



Dimensionality reduction – T-SNE for Nd-P pair

T-SNE (T-distributed Stochastic Neighbourhood Embedding)





Yellow: P

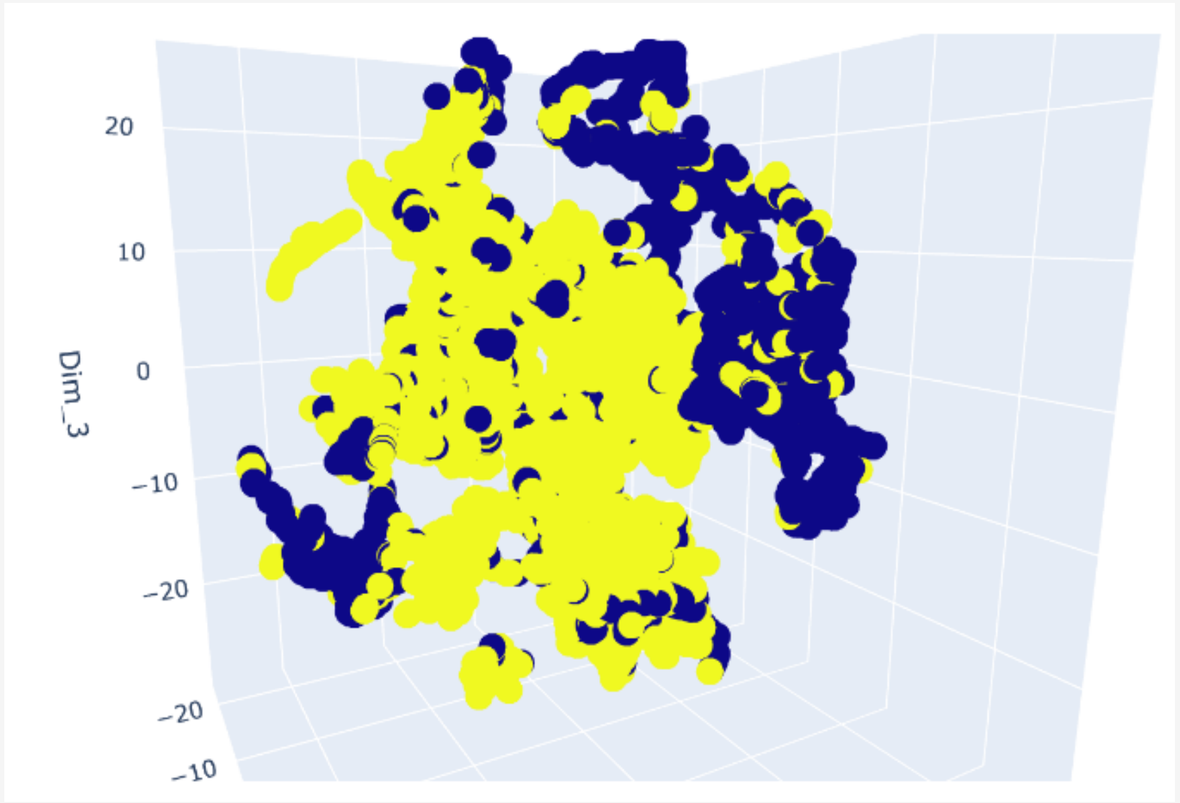
Blue:Nd

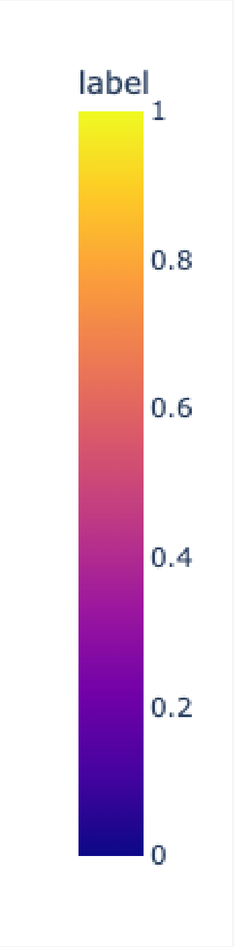
T-SNE justified the root cause for the mishappening:

- Nd-P pairs were completely mixed as shown in the diagram

Dimensionality reduction - T-SNE for Nd-Ti pair

T-SNE (T-distributed Stochastic Neighbourhood Embedding)





Dimension 1:
Dimension 2:
Dimension 3:

Yellow: Ti
Blue: Nd

T-SNE justified the root cause for the mishappening:

- Nd-Ti points were clearly distinguished

Tackling Overlap between Nd-P pair

Clustering based under sampling

In ClusBus, minority class samples are retained and majority class points are discarded to make a clear distinction.

Methods such as Clustering Based Under Sampling are used to deal with class overlap for highly skewed datasets as data-cleaning techniques that remove minority classes are not desirable since they can cause the loss of highly informative data.

Dealing with Overlap – Binary over Multiclass Classification

Goal:

- Club Porosity and Tungsten Inclusion into one class named 'Defect' creating a Binary Classification Problem yet again!
- Obtain a good recall for the 'Defect' and 'No Defect' classes.
- On the prediction of a defect, use the P-Ti Classifier to classify the type of defect.

Dealing with Overlap – Binary over Multiclass Classification

Now using the transformed dataset (into defect and no defect classes), we ran several algorithms especially the **imbalanced_ensemble** algorithms:

- `RUSBoostClassifier`
- `EasyEnsembleClassifier`
- `EasyEnsembleClassifier with KNN as base estimator`
- `SelfPacedEnsembleClassifier`
- `SelfPacedEnsembleClassifier with KNN as base estimator`
- `BalancedRandomForestClassifier with different RUS and Smote strategies`

Dealing with Overlap – Binary over Multiclass Classification

Range of Recalls for both the classes = 0.73-0.85

We then saved around 9 different models with slightly different metrics for the sake of ensembling.
Post ensembling we achieved a very good recall on test set for both the defect and no defect classes.

BalancedRandomForestClassifier for P-Ti

BalancedRandomForestClassifier with **SmoteTomek upsampling** with a **sampling ratio of 0.75** for p-ti pair with recalls of 0.89 and 0.93 respectively

	precision	recall	f1-score	support
Porosity	0.77	0.89	0.83	103
Tungsten Inclusion	0.97	0.93	0.95	415
accuracy			0.92	518
macro avg	0.87	0.91	0.89	518
weighted avg	0.93	0.92	0.93	518

Multiclass classification metrics

BalancedRandomForestClassifier for Nd-P-Ti as multiclass classification With RandomUnderSampling with sampling ratio = minority class

	precision	recall	f1-score	support
No Defect	1.00	0.48	0.65	82104
Porosity	0.00	0.44	0.00	114
Tungsten Inclusion	0.03	0.83	0.07	461
accuracy			0.48	82679
macro avg	0.34	0.58	0.24	82679
weighted avg	0.99	0.48	0.64	82679

Multiclass classification metrics

BalancedRandomForestClassifier for Nd-P-Ti as multiclass classification With RandomUnderSampling with sampling ratio = minority class

	precision	recall	f1-score	support
No Defect	1.00	0.48	0.65	82104
Porosity	0.00	0.44	0.00	114
Tungsten Inclusion	0.03	0.83	0.07	461
accuracy			0.48	82679
macro avg	0.34	0.58	0.24	82679
weighted avg	0.99	0.48	0.64	82679

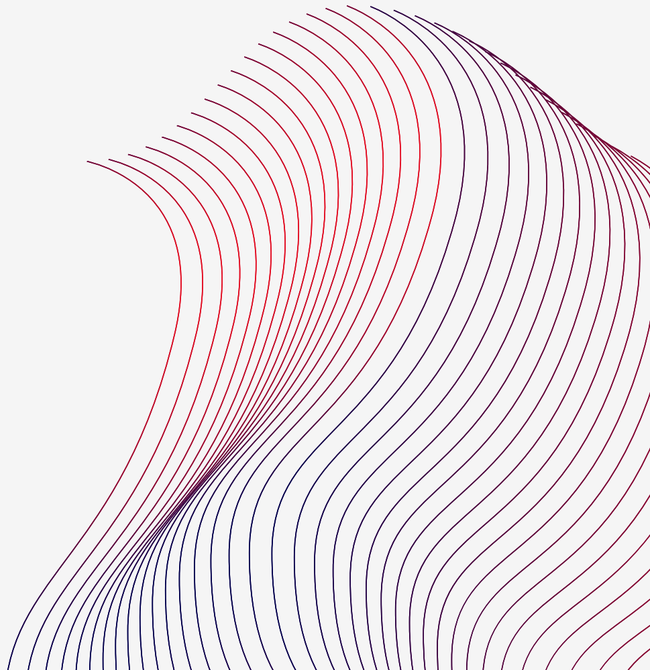
Multiclass classification metrics

OneVsRest Classifier with RUS (Minority class) using SVM as estimator

	precision	recall	f1-score	support
No Defect	1.00	0.19	0.31	82105
Porosity	0.00	0.40	0.00	114
Tungsten Inclusion	0.01	0.93	0.02	461
accuracy			0.19	82680
macro avg	0.34	0.51	0.11	82680
weighted avg	0.99	0.19	0.31	82680

RUSBoostClassifier

	precision	recall	f1-score	support
No Defect	1.00	0.39	0.56	82105
Porosity	0.00	0.48	0.00	114
Tungsten Inclusion	0.03	0.83	0.06	461
accuracy			0.39	82680
macro avg	0.34	0.57	0.21	82680
weighted avg	0.99	0.39	0.56	82680



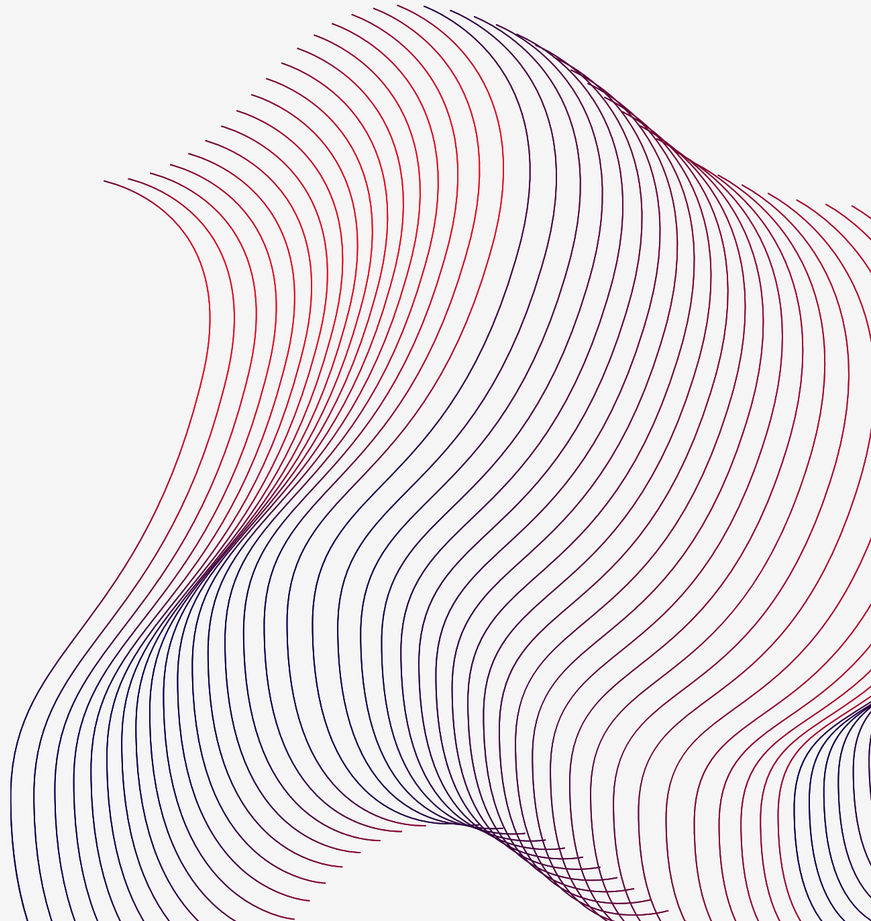
Multiclass classification metrics

EasyEnsembleClassifier with KNN as base estimator

	precision	recall	f1-score	support
No Defect	1.00	0.42	0.59	82105
Porosity	0.00	0.46	0.00	114
Tungsten Inclusion	0.02	0.82	0.04	461
accuracy			0.42	82680
macro avg	0.34	0.57	0.21	82680
weighted avg	0.99	0.42	0.58	82680

SelfPacedEnsembleClassifier

	precision	recall	f1-score	support
No Defect	1.00	0.34	0.51	82105
Porosity	0.00	0.56	0.00	114
Tungsten Inclusion	0.03	0.92	0.05	461
accuracy			0.35	82680
macro avg	0.34	0.61	0.19	82680
weighted avg	0.99	0.35	0.51	82680



Multiclass classification metrics

SelfPacedEnsembleClassifier with KNN as base estimator

	precision	recall	f1-score	support
No Defect	1.00	0.33	0.50	82105
Porosity	0.00	0.49	0.00	114
Tungsten Inclusion	0.02	0.86	0.04	461
accuracy			0.33	82680
macro avg	0.34	0.56	0.18	82680
weighted avg	0.99	0.33	0.49	82680

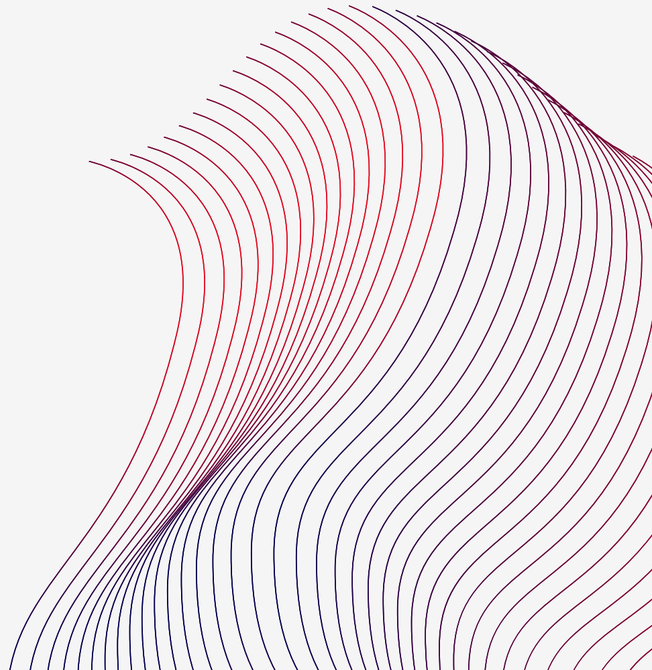
Defect/No Defect classification metrics

XGBoost with RUS (sampling ratio = 1)

	precision	recall	f1-score	support
Defect	0.03	0.74	0.06	575
No Defect	1.00	0.85	0.92	82105
accuracy			0.85	82680
macro avg	0.52	0.79	0.49	82680
weighted avg	0.99	0.85	0.91	82680

EasyEnsembleClassifier

	precision	recall	f1-score	support
Defect	0.02	0.79	0.04	575
No Defect	1.00	0.74	0.85	82105
accuracy			0.74	82680
macro avg	0.51	0.76	0.44	82680
weighted avg	0.99	0.74	0.84	82680



Defect/No Defect classification metrics

SelfPacedEnsembleClassifier

	precision	recall	f1-score	support
Defect	0.03	0.74	0.06	575
No Defect	1.00	0.84	0.91	82105
accuracy			0.84	82680
macro avg	0.51	0.79	0.48	82680
weighted avg	0.99	0.84	0.90	82680

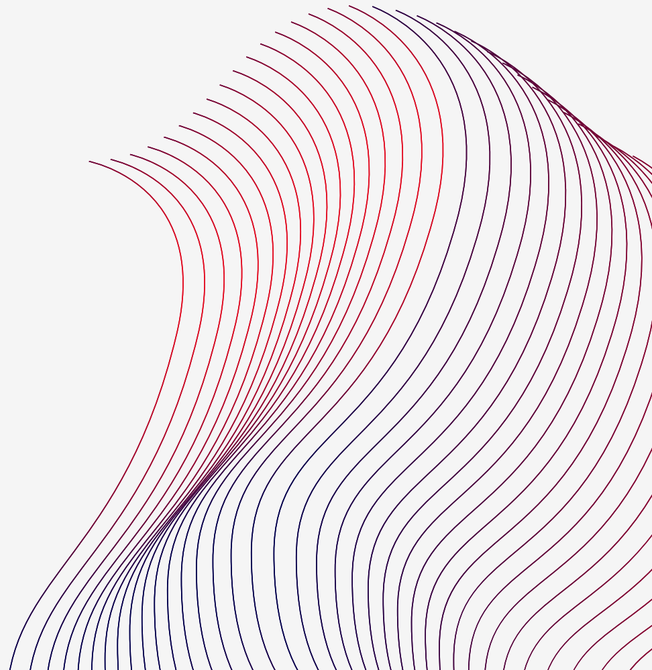
RUSBoostCassifier

	precision	recall	f1-score	support
Defect	0.03	0.75	0.05	575
No Defect	1.00	0.82	0.90	82105
accuracy			0.82	82680
macro avg	0.51	0.78	0.48	82680
weighted avg	0.99	0.82	0.89	82680

Defect/No Defect classification metrics

BalancedRandomForestClassifier
with RUS = 1

	precision	recall	f1-score	support
Defect	0.03	0.71	0.05	575
No Defect	1.00	0.82	0.90	82105
accuracy			0.82	82680
macro avg	0.51	0.77	0.47	82680
weighted avg	0.99	0.82	0.89	82680



Defect/No Defect classification metrics

BalancedRandomForestClassifier
with RUS = 1

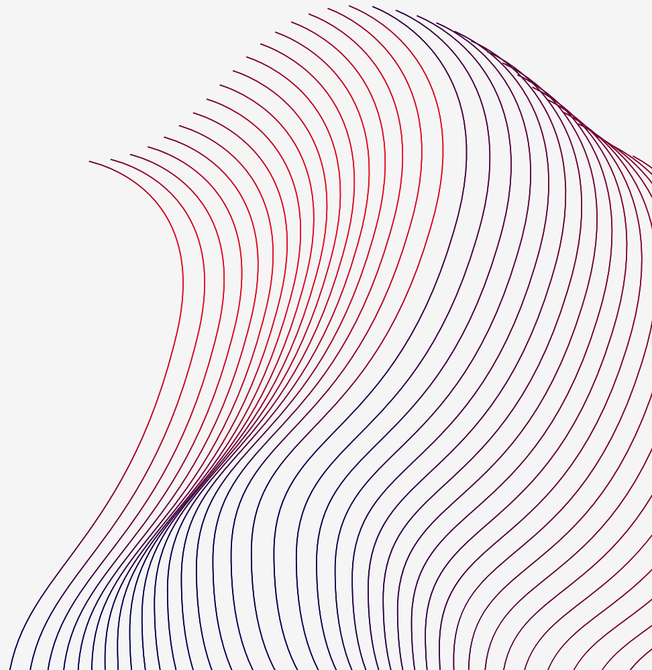
	precision	recall	f1-score	support
Defect	0.03	0.71	0.05	575
No Defect	1.00	0.82	0.90	82105
accuracy			0.82	82680
macro avg	0.51	0.77	0.47	82680
weighted avg	0.99	0.82	0.89	82680

FINAL MODEL

Defect/No Defect classification metrics

	precision	recall	f1-score	support
Defect	0.04	0.98	0.08	575
No Defect	1.00	0.84	0.91	82105
accuracy			0.84	82680
macro avg	0.52	0.91	0.49	82680
weighted avg	0.99	0.84	0.91	82680

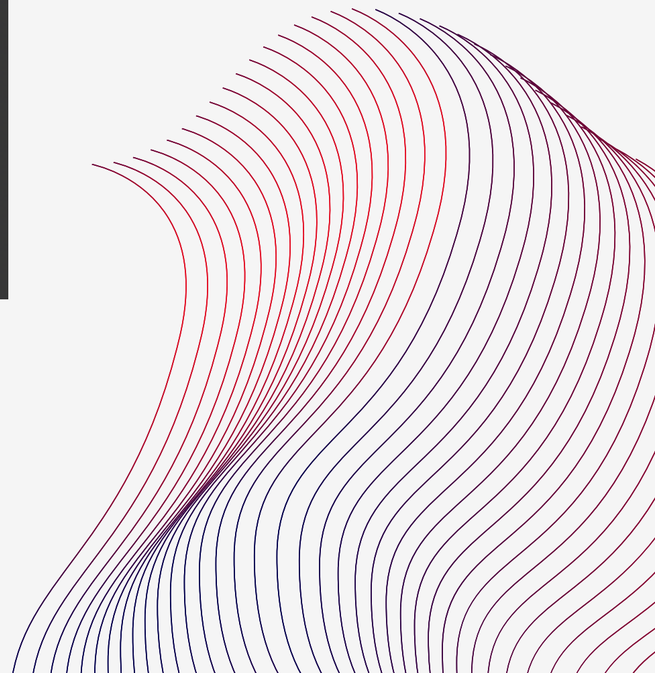
	precision	recall	f1-score	support
No Defect	1.00	0.84	0.91	82105
Porosity	0.03	0.86	0.07	114
Tungsten Inclusion	0.04	0.94	0.08	461
accuracy			0.84	82680
macro avg	0.36	0.88	0.35	82680
weighted avg	0.99	0.84	0.91	82680



FINAL MODEL

Defect/No Defect classification metrics

	precision	recall	f1-score	support
Defect	0.04008	0.97565	0.07699	575
No Defect	0.99980	0.83634	0.91079	82105
accuracy			0.83731	82680
macro avg	0.51994	0.90600	0.49389	82680
weighted avg	0.99312	0.83731	0.90500	82680

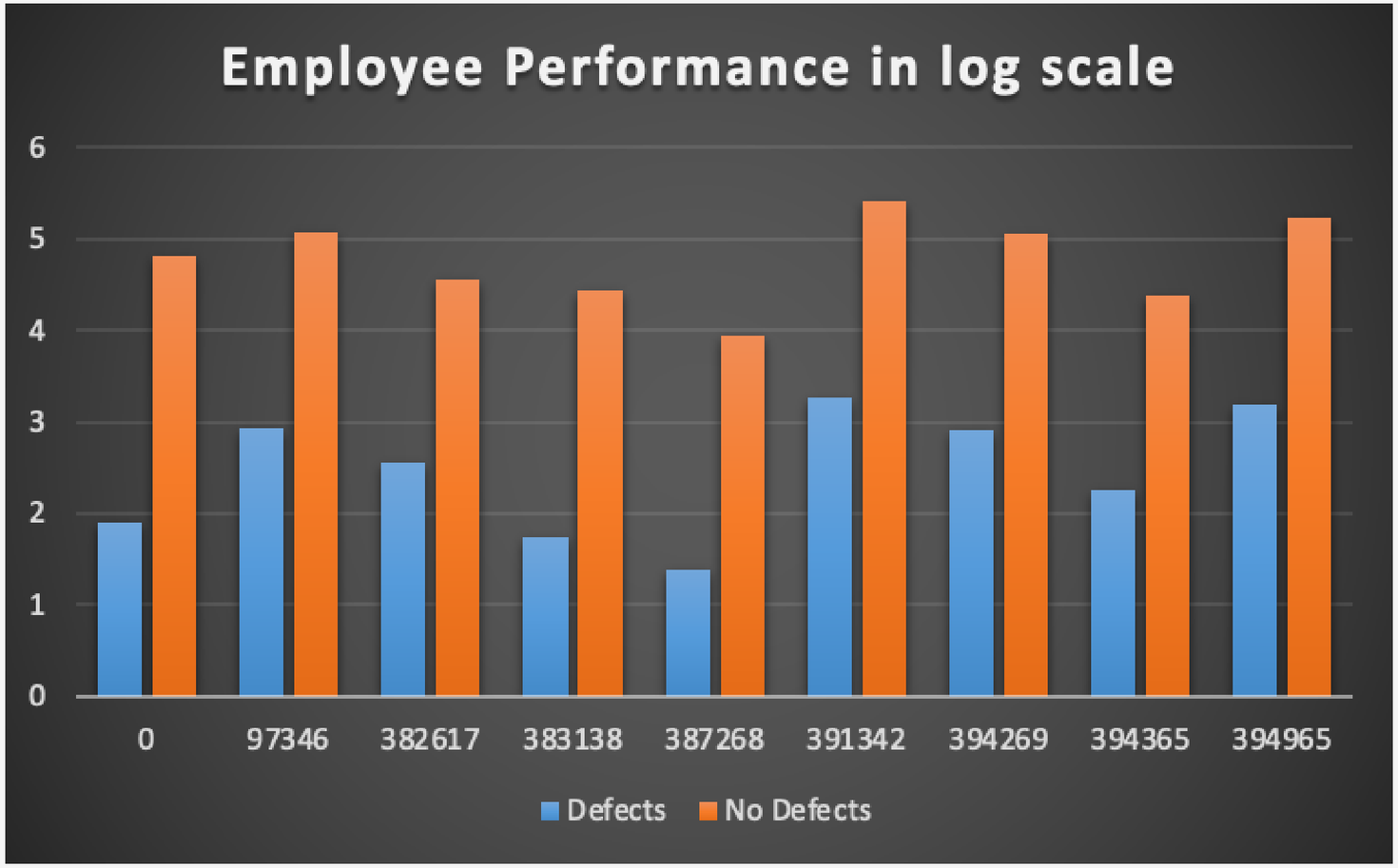


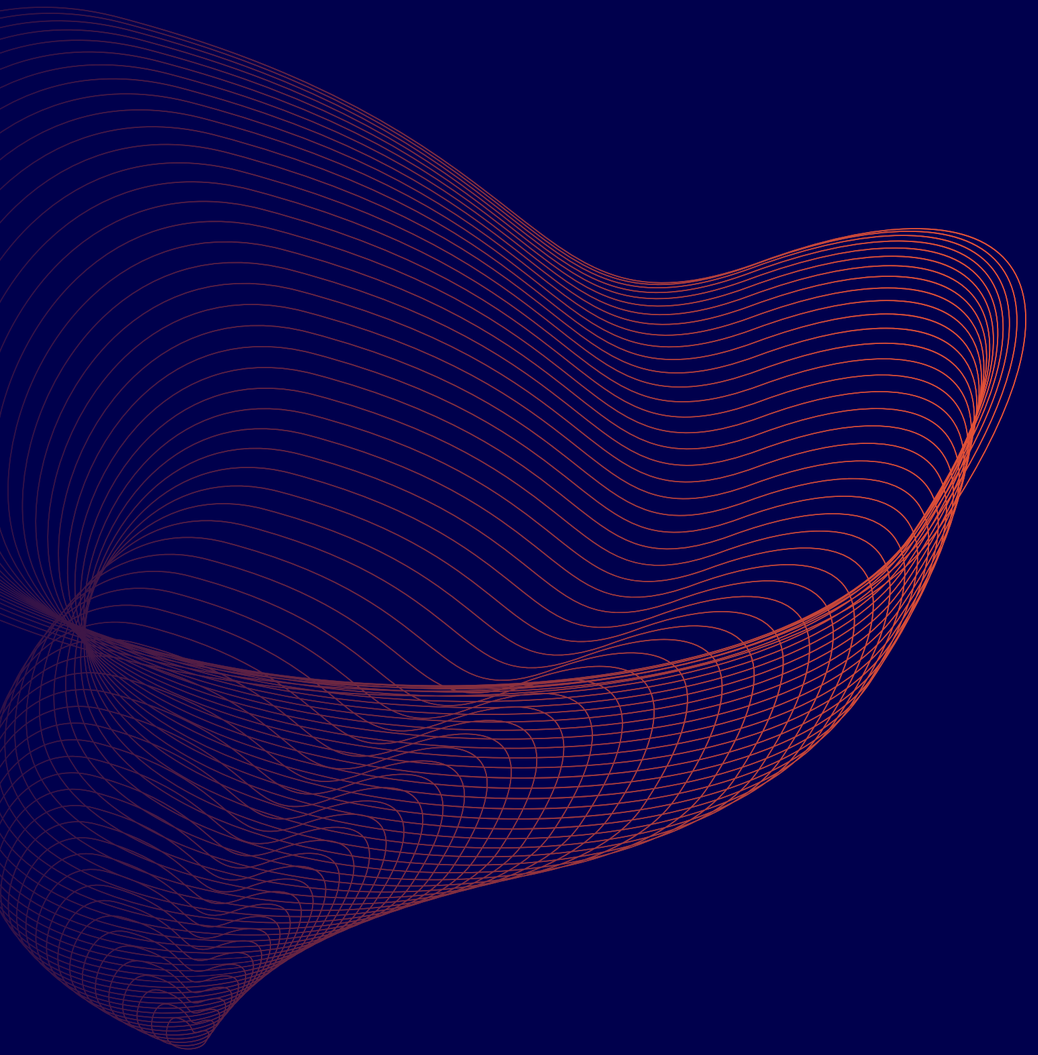
Impact of The Model in Industry

Consider a batch of **1000 units** of any high precision machining unit that is manufactured.
Based on the ratio of defects from the given dataset, one can deduce the number of defective pieces from this batch must be around 7-8.

By the use of the model, from a dataset which has 8 defects, the model will classify around 200 of them as defects. However, due to the extremely high recall, it is ensured that every defect is classified correctly.

Employee Performance

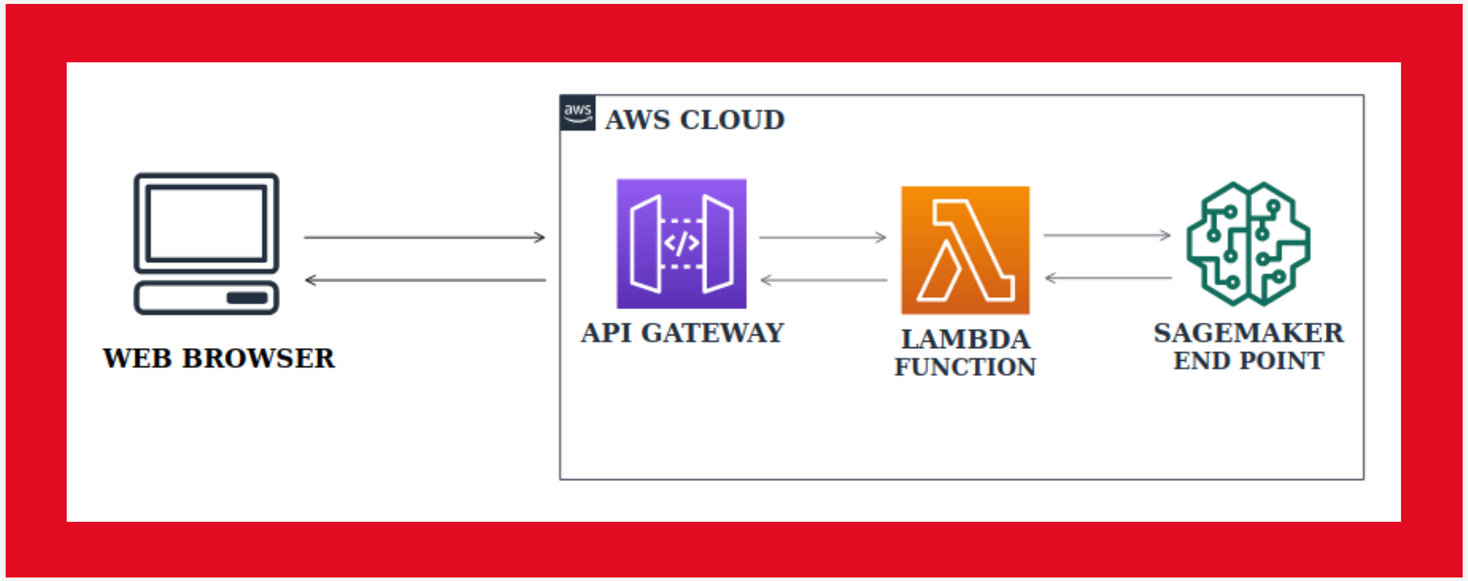




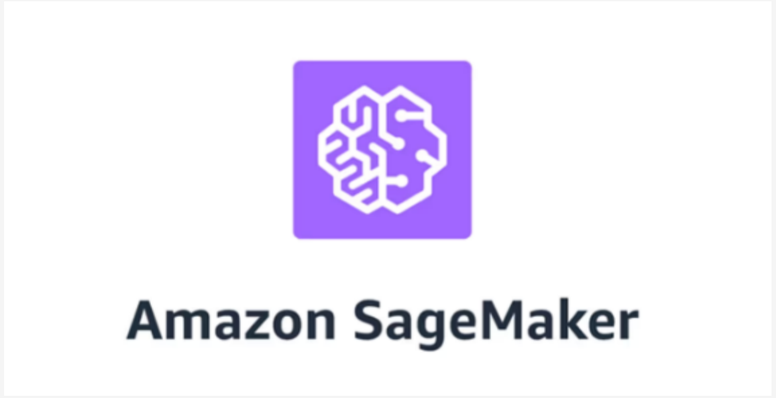
BUSINESS MODEL

MODEL DEPLOYMENT

The **Cost of Ownership** of the Project will be the cost of deploying the project onto an existing production environment which, on feeding the model, returns the output that can be used in the industry for decision-making.



AWS SAGEMAKER



Specifications

- Amazon EC2 instance, Asia Pacific(Mumbai) Region
- Server: c6gn.2xlarge
- Operating System - Linux
- $0.254 - On-demand hourly cost
- vCPUs - 8
- RAM- 6GB
- Memory(GiB) - 8 GiB
- Network Performance - Upto 10GigaBit

Considering the current AWS SageMaker Pricing for the above specifications, the cost is $0.254 per hour.

Thus, TCO for the Model is which is 45,301.81 Rs per annum (considering the utilization of 6hrs per day)

Return of Investment

Model : Pre-production Method

NDT : Post - production Method

Assumed size of the Batch = **1000 units**

Average Cost of Weld = $350 per job

Average number of Defective Welds produced per thousand Welds = 8

ROI = (Net Savings / Cost of Investment) x 100

Return of Investment

Model : Pre-production Method

NDT : Post-production Method

The precision for **No defect** is 0.99980

Precision = Correctly predicted as *No Defect* / Total Predicted *No Defect*

Incorrect predictions are 1 - 0.99980 = 0.0002

Which is merely 0.02%

Only 0.02% chance that the model will predict a Defective piece as a No defect piece.

ROI = (Net Savings / Cost of Investment) x 100

Return of Investment

If the model predicts that a certain set of parameters will give a No Defect Unit, we can safely skip the cost for NDT for such units.

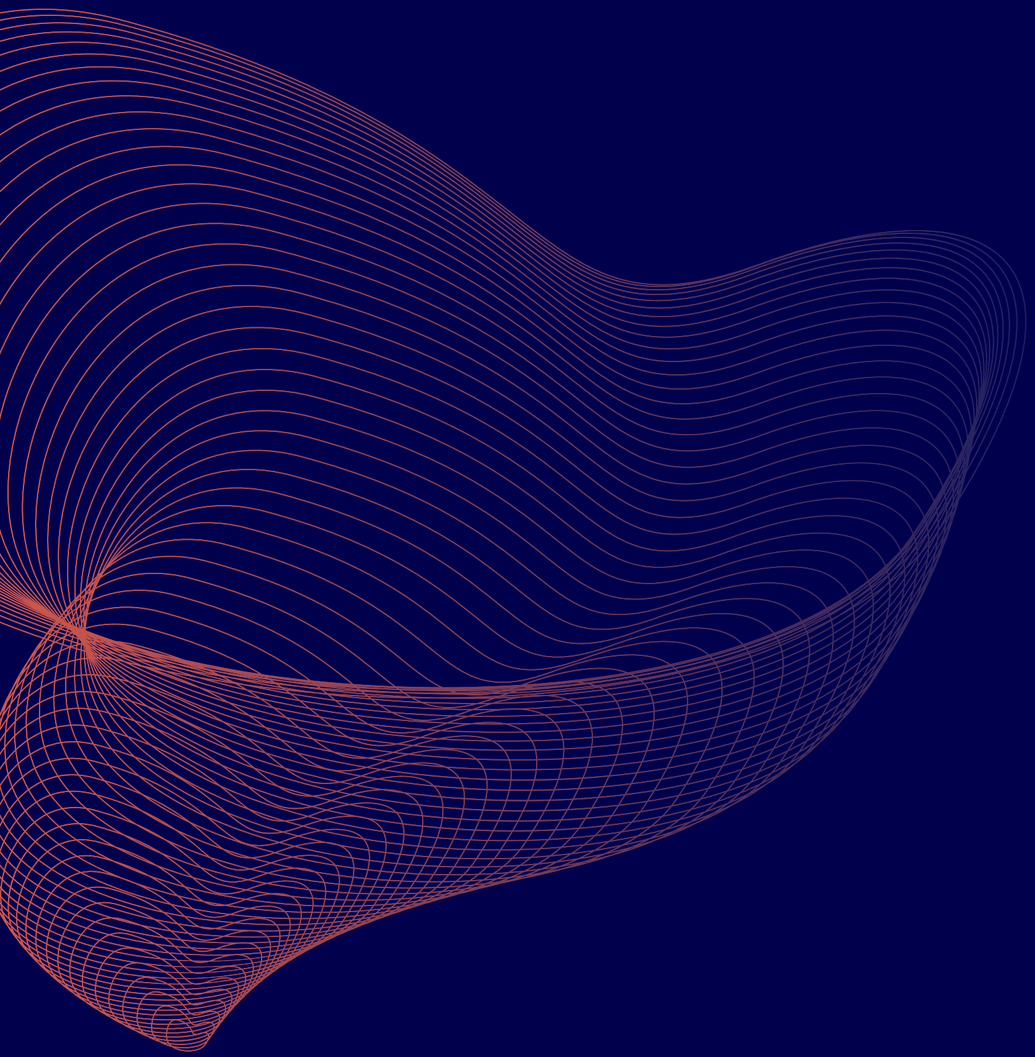
This will be the **major cause** for savings by the utilisation of the model.

Good alternative to the post productive NDT methods for identifying defects.

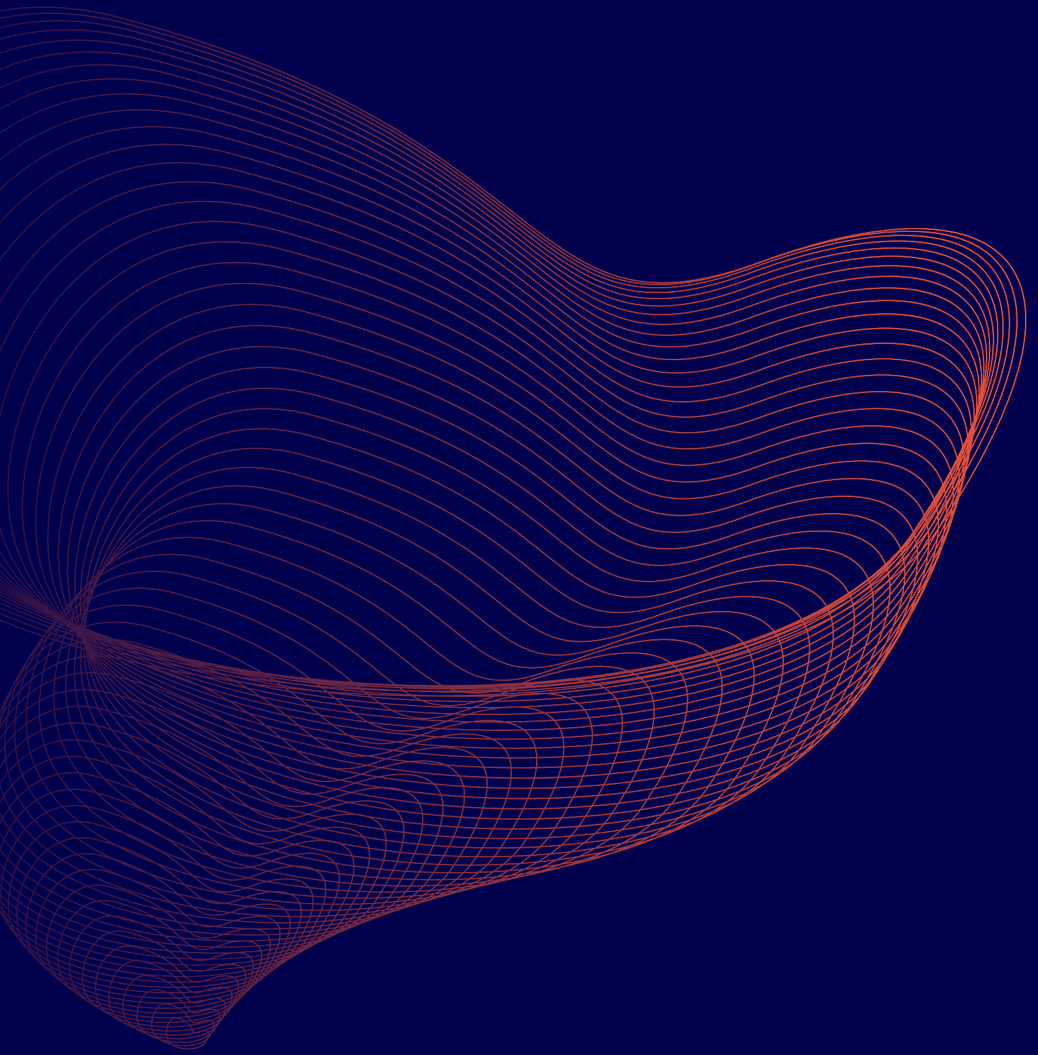
ROI = (Net Savings / Cost of Investment) x 100

Bibliography

https://sentin.ai/en/3-failures-and-catastrophes-in-ndt/



Working of the model with test data



THANK YOU!