# 3D Printer Material Prediction Using Watson Auto AI

## Final Report

Submitted By: **Vaibhav**

Internship Title: **RSIP Career Basic ML 159**

Project ID : **SPS_PRO_305**

# Table of Contents

# APPENDIX

## A. Source code

# 1. INTRODUCTION

## 1.1. Overview

3D printing materials are usually called by their traditional names such as ABS, nylon and etc. They are available in majority, but we have to be aware that many of the 3D printing materials only mimic true thermoplastics. We need to choose the right material to get a better printed object. Choosing the right material allows us to improve the shape, quality and function of our 3d printed part. Hence, selection of the correct 3D printing material is highly essential. To identify the type of material required after a 3D model is designed is a complicated task. The aim of the study is to determine the material which will be perfectly suitable for the given use case. We have a dataset in which there are eleven setting parameters and one output parameters. Based on these input parameters we have to predict the best material for model. This model will predict whether to use ABS or PLA.

## 1.2. Purpose

The purpose of this project is to provide an output to the users based on the input parameters. The output will tell the users, which material is best suitable for their needs. We will use machine learning techniques to classify the the materials as ABS or PLA based on the input parameters.We are building a IBM Watson AutoAI Machine Learning to predict the material. We are developing a web application which is built using node red service. We make use of the scoring end point to give user input values to the deployed model. The model prediction is then showcased on User Interface. This model is to predict the best material to be used for building 3D models.

# 2. LITERATURE SURVEY

## 2.1. Existing Problem

**Choosing the right material for 3D printing a part is getting complex day by day. Choosing the right material allows us to improve the shape, quality and function of our 3d printed part. Hence, selection of the correct 3D printing material is highly essential. To identify the type of material required after a 3D model is designed is a complicated task. We have input parameters based on which we predict which material will be best suitable for our needs.**

## 2.2. Proposed Solution

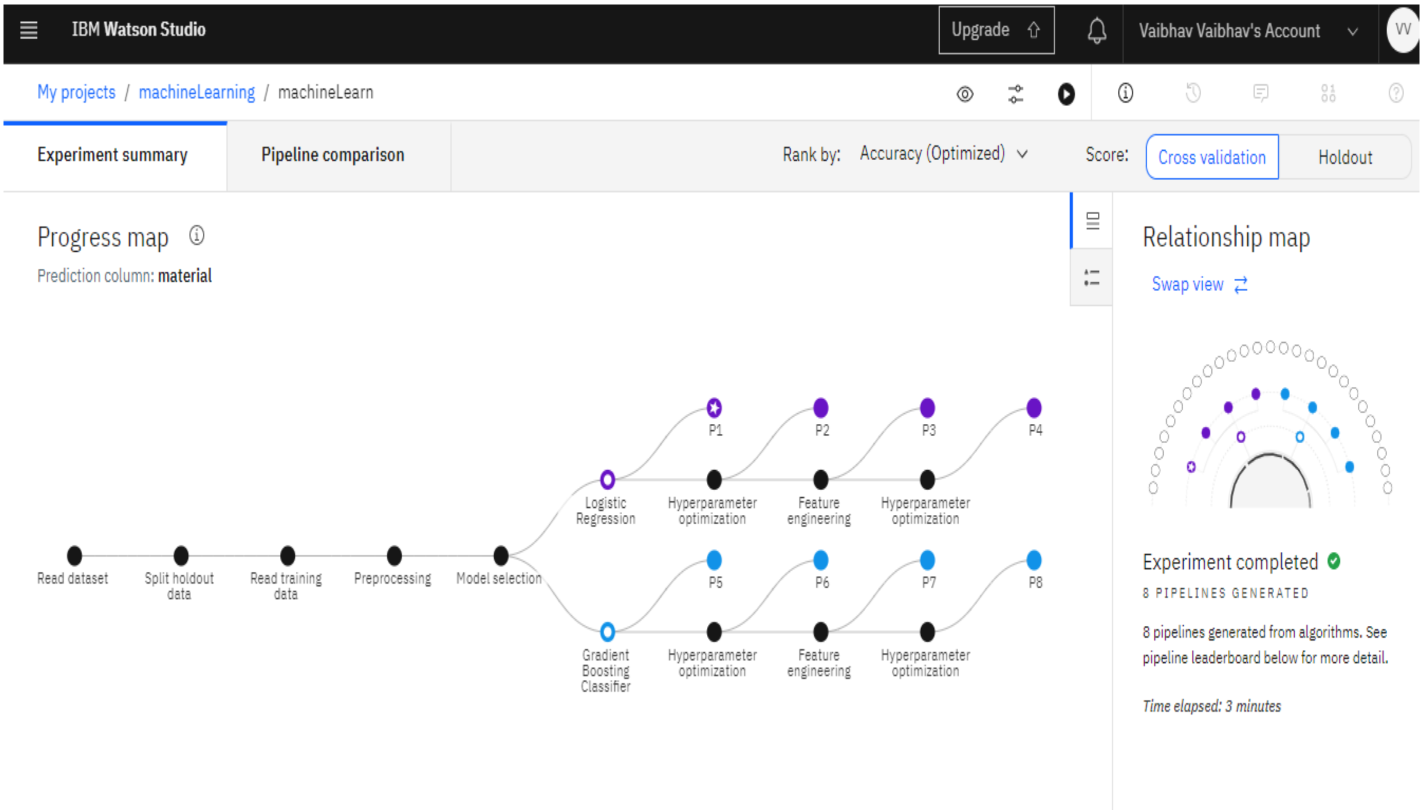
**With the use of Machine Learning Model, there will be no limitation of the complexity increasing number of variables. This model can train and test the given parameters and with the best performing machine learning model it can effortlessly predict the best material suitable for 3D printing an object with much**

**higher accuracy than traditional methods.**

**For making we will use Watson Studios Auto AI Experiment feature. We just have to input the data and Auto AI will generate the model according to it. Then we can deploy the model and use Node Red to make a web application.**

## 3. THEORITICAL ANALYSIS

### 3.1 Block diagram




## 3.2 Hardware / Software designing

## For Auto AI solution:

- **Strategy: matching the problem with the solution.**

- **Dataset preparation and pre-processing. Data collection.**

- **Adding Dataset to the Watson Machine Learning.**

- **Doing Auto AI analysis to find out the best model.**

- **Model deployment.**

- **Making Node Red flow.**

- **Deploying the machine learning model through that Flow Application.**

**For own ipynb Notebook solution:**

- **Strategy: matching the problem with the solution.**

- **Dataset preparation and pre-processing. Data collection. Data visualization.**

**Labelling. Data selection. Data pre-processing. Data**

**transformation.**

- **Dataset splitting into train data and test data.**

- **Modelling. Model training. Model evaluation and testing. Improving predictions with ensemble methods.**

- **Model deployment.**

- **Making Node Red flow.**

- **Deploying the machine learning model through that Flow Application.**

## 4. EXPERIMENTAL INVESTIGATIONS

**This dataset comes from research by TR/Selcuk University Mechanical Engineering department.**

**The aim of the study is to determine how much of the adjustment parameters in 3d printers affect the print quality, accuracy and strenght. Where there are eleven setting parameters and one measured output parameters.**

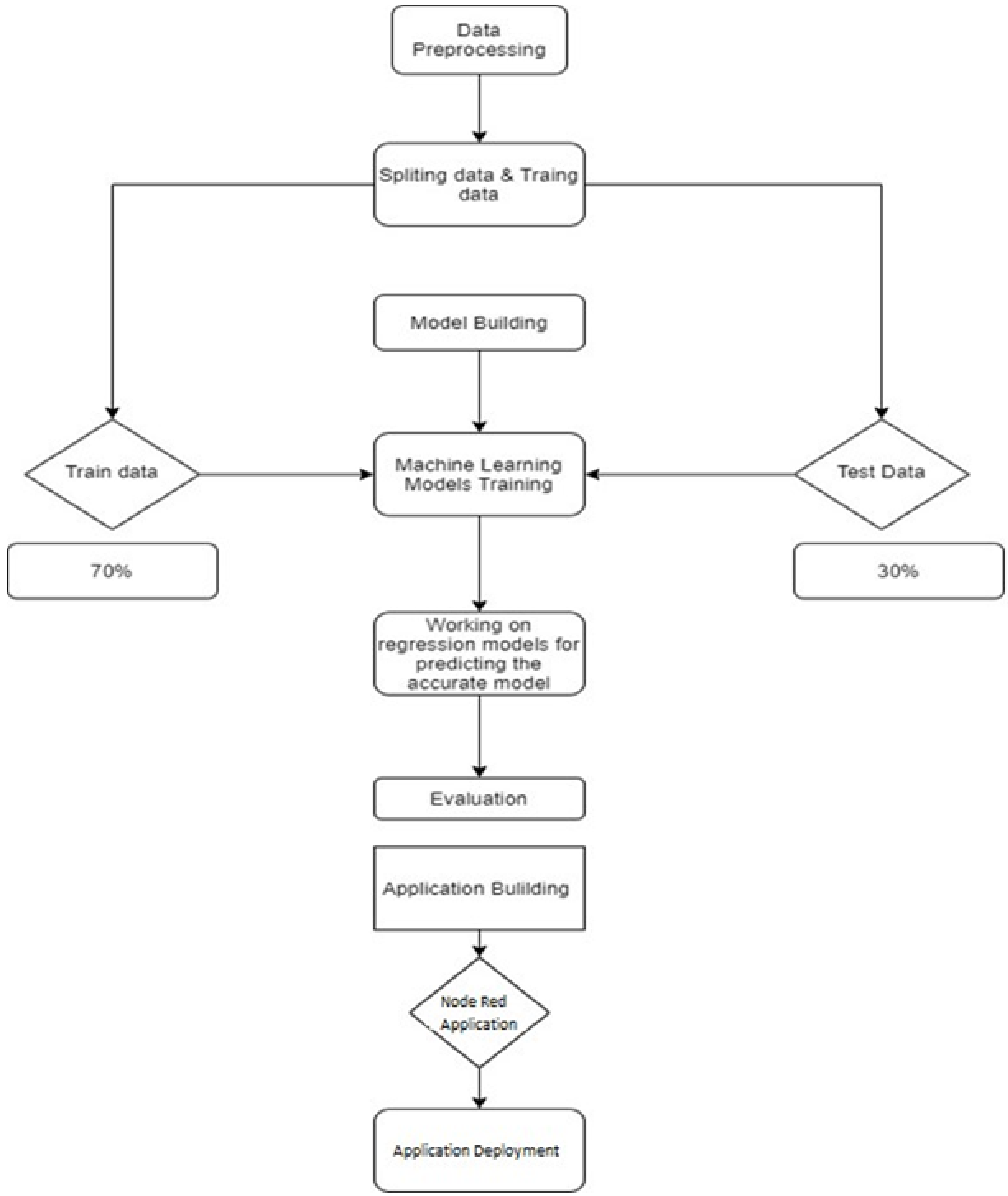
**Content**

**Setting Parameters:**

- **Layer Height (mm)**
- **Wall Thickness (mm)**
- **Infill Density (%)**
- **Infill Pattern ()**
- **Nozzle Temperature (Cº)**

- **Bed Temperature (Cº)**
- **Print Speed (mm/s)**
- **Material () (output parameter)**
- **Fan Speed (%)**
- **Roughness (µm)**
- **Tension (ultimate) Strenght (MPa)**
- **Elongation (%)**

## 5. FLOWCHART


Data Preprocessing
Spliting data & Traing data
Model Building
Train data
Machine Learning Models Training
Test Data
70%
30%
Working on regression models for predicting the accurate model
Evaluation
Application Bulilding
Node Red Application
Application Deployment

## 6. RESULT

**Based on the 11 inputs entered by the user, the model predicts the best material for 3D printing an object. And gives the output according to the entries in the Node red application.**

## 7. ADVANTAGES & DISADVANTAGES

### 7.1. Advantages

- **Unlike traditional methods there is no wastage of test samples.**

- **Higher accuracy can reduces errors in wrong selection of material.**

- **Reduce the cost of finding out best material for 3D printing an object.**

- **Easy user interface with straight forward**

**prediction.**

## **7.2. Disadvantages**

- **The model is limited to predict the material for only those materials which have exactly 11 compositions in their mixture.**

- **The input parameters need to be correctly examined before the prediction is made.**

## **8. APPLICATIONS**

- **It can be used to predict the best material suitable for 3D printing an object that is made using several parameters.**

- **Implementable on the website.**

- **Can also be made into a phone app. 9.**

## CONCLUSION

**Since anyone who is getting a part 3D printed does not want to waste resources and wants to obtain a reliable product. Our application helps in predicting the best material for 3d printing their object based on the past data.**

## 10. FUTURE SCOPE

**With this model now engineers would be able to determine the best material for 3D printing an object. Based on this many would be able to advise which material to use for 3D printing an object based on the given input parameters. This model can predict the outcome with many different inputs within seconds. The model will save a lot of time. Experiment cost is also reduced which creates a bigger opportunity in cost effectiveness work.**

## 11. BIBILOGRAPHY APPENDIX

https://smartbridge.teachable.com/courses/843009/lectures/17692123

https://smartinternz.com/Student/workspace/3430

https://www.kaggle.com/afumetto/3dprinter

## SOURCE CODE:

### Setup

**Before you use the sample code in this notebook, you must perform the following setup tasks:**

**watson-machine-learning-client uninstallation of the old client**

**watson-machine-learning-client-V4 installation**

**autoai-libs installation/upgrade**

**lightgbm or xgboost installation/downgrade if they are needed**

**!pip uninstall watson-machine-learning-client -y**

**!pip install -U watson-machine-learning-client-V4**

**!pip install -U autoai-libs**

**AutoAI experiment metadata**

**This cell defines COS credentials required to retrieve AutoAI pipeline.**

```
# @hidden_cell

from watson_machine_learning_client.helpers import DataConnection, S3Connection, S3Location



training_data_reference = [DataConnection(

    connection=S3Connection(

api_key='UdiAe8QAEa8b_n87nNRr3VQIYJXdFV2JBe72yWozlkj5',

auth_endpoint='https://iam.bluemix.net/oidc/token/',
```

```
endpoint_url='https://s3-api.us-geo.objectstorage.softlayer.net'

    ),

        location=S3Location(

bucket='machinelearning-donotdelete-pr-xm9cbxhxqrda9g',

        path='data.csv'

    ))

]

training_result_reference = DataConnection(
```

```
    connection=S3Connection(

api_key='UdiAe8QAEa8b_n87nNRr3VQIYJXdFV2JBe72yWozlkj5',

auth_endpoint='https://iam.bluemix.net/oidc/token/',

endpoint_url='https://s3-api.us-geo.objectstorage.softlayer.net'

    ),

    location=S3Location(

bucket='machinelearning-donotdelete-pr-xm9cbxhxqrda9g',
```

**path='auto_ml/5c905225-d804-4e15-ade0-2811f8a26545/wml_data/42c0fd67-4a26-472b-8484-6059fe0e3e9c/data/automl',**

**model_location='auto_ml/5c905225-d804-4e15-ade0-2811f8a26545/wml_data/42c0fd67-4a26-472b-8484-6059fe0e3e9c/data/automl/pre_hpo_d_output/Pipeline1/model.pickle',**

**training_status='auto_ml/5c905225-d804-4e15-ade0-2811f8a26545/wml_data/42c0fd67-4a26-472b-8484-6059fe0e3e9c/training-status.json'**

**))**

**Following cell contains input parameters provided to run the AutoAI experiment in Watson Studio**

```
experiment_metadata = dict(
  prediction_type='classification',
  prediction_column='material',
  test_size=0.1,
  scoring='accuracy',
  csv_separator=',',
  excel_sheet=0,
  max_number_of_estimators=2,
```

```
   training_data_reference = training_data_reference,

   training_result_reference =
training_result_reference)
```

```
pipeline_name='Pipeline_1'
```

**Pipeline inspection**

**In this section you will get the trained pipeline model from the AutoAI experiment and inspect it.**

**You will see pipeline as a pythone code, graphically visualized and at the end, you will perform a local test.**

**Get historical optimizer instance**

**The next cell contains code for retrieving fitted optimizer.**

```
from watson_machine_learning_client.experiment
import AutoAI
```

```
optimizer =
AutoAI().runs.get_optimizer(metadata=experiment_m
etadata)
```

## Get pipeline model

The following cell loads selected AutoAI pipeline model. If you want to get pure scikit-learn pipeline specify as_type='sklearn' parameter. By default enriched scikit-learn pipeline is returned as_type='lale'.

```
pipeline_model =
optimizer.get_pipeline(pipeline_name=pipeline_name)
```

## Preview pipeline model as python code

In the next cell, downloaded pipeline model could be previewed as a python code.

You will be able to see what exact steps are involved in model creation.

```
pipeline_model.pretty_print(combinators=False, ipython_display=True)
```

## Visualize pipeline model

Preview pipeline model stages as graph. Each node's name links to detailed description of the stage.

pipeline_model.visualize()

Read training and holdout data

Retrieve training dataset from AutoAI experiment as pandas DataFrame.

training_df, holdout_df = optimizer.get_data_connections()[0].read(with_holdout_split=True)

train_X = training_df.drop([experiment_metadata['prediction_c

**olumn']], axis=1).values**

**train_y = training_df[experiment_metadata['prediction_column']].values**

**test_X = holdout_df.drop([experiment_metadata['prediction_column']], axis=1).values**

**y_true = holdout_df[experiment_metadata['prediction_column']].values**

**Test pipeline model locally**

**Note: you can chose the metric to evaluate the model**

**by your own, this example contains only a basic scenario.**

```
from sklearn.metrics import accuracy_score

predictions = pipeline_model.predict(test_X)

score = accuracy_score(y_true=y_true,
y_pred=predictions)

print('accuracy_score: ', score)
```

**Pipeline refinery and testing (optional)**

**In this section you will learn how to refine and retrain the best pipeline returned by AutoAI. It can be performed by:**

**modifying pipeline definition source code**

**using lale library for semi-automated data science**

**Note: In order to run this section change following cells to 'code' cell.**

**Pipeline definition source code**

**Following cell lets you experiment with pipeline**

definition in python, e.g. change steps parameters.

It will inject pipeline definition to the next cell.

```
pipeline_model.pretty_print(combinators=False,
ipython_display='input')
```

Lale library

Note: This is only an exemplary usage of lale package. You can import more different estimators to refine downloaded pipeline model.

**Import estimators**

**from sklearn.linear_model import LogisticRegression as E1**

**from sklearn.tree import DecisionTreeClassifier as E2**

**from sklearn.neighbors import KNeighborsClassifier as E3**

**from lale.lib.lale import Hyperopt**

**from lale.operators import TrainedPipeline**

**from lale import wrap_imported_operators**

**from lale.helpers import**

**import_from_sklearn_pipeline**

**wrap_imported_operators()**

**Pipeline decomposition and new definition**

**In this step the last stage from pipeline is removed.**

**prefix = pipeline_model.remove_last().freeze_trainable()**

**prefix.visualize()**

**new_pipeline = prefix >> (E1 | E2 | E3)**

**new_pipeline.visualize()**

**New optimizer hyperopt configuration and training**

**This section can introduce other results than the original one and it should be used by more advanced users.**

**New pipeline is re-trained by passing train data to it and calling fit method.**

**hyperopt = Hyperopt(estimator=new_pipeline, cv=3, max_evals=20)**

```
fitted_hyperopt = hyperopt.fit(train_X, train_y)

hyperopt_pipeline = fitted_hyperopt.get_pipeline()

new_pipeline =
hyperopt_pipeline.export_to_sklearn_pipeline()

predictions = new_pipeline.predict(train_X)

predictions = new_pipeline.predict(test_X)

refined_score = accuracy_score(y_true=y_true,
y_pred=predictions)

print('accuracy_score: ', score)

print('refined_accuracy_score: ', refined_score)
```

**Deploy and Score**

**In this section you will learn how to deploy and score pipeline model as webservice using WML instance.**

**Connect to WML client in order to create deployment**

**Action: Next you will need credentials for Watson Machine Learning and training run_id:**

**go to Cloud catalog resources list**

**click on Services and chose Machine Learning service.**

**Once you are there**

**click the Service Credentials link on the left side of the screen**

**click to expand specific credentials name.**

**copy and paste your WML credentials into the cell below**

**Take in mind that WML Service instance should be the same as used to generate this notebook.**

```
wml_credentials = {

  "apikey": "",
```

```
  "iam_apikey_description": "",

  "iam_apikey_name": "",

  "iam_role_crn": "r",

  "iam_serviceid_crn": "",

  "instance_id": "",

  "url": ""

}
```


**Create deployment**

**Action: If you want to deploy refined pipeline please**

change the pipeline_model to new_pipeline.

If you prefer you can also change the deployment_name.

from watson_machine_learning_client.deployment import WebService

service = WebService(wml_credentials)

service.create(

```
    model=pipeline_model,
    metadata=experiment_metadata,
    deployment_name=f'{pipeline_name}_webservice'
)
```

Deployment object could be printed to show basic information:

```
print(service)
```

To be able to show all available information about deployment use .get_params() method:

```
service.get_params()
```

Score webservice

You can make scoring request by calling score() on deployed pipeline.

```
predictions = service.score(payload=holdout_df.drop([experiment_metadata['prediction_column']], axis=1).iloc[:10])

predictions
```

If you want to work with the webservice in external Python application you can retrieve the service object

**by:**

**initialize service by service = WebService(wml_credentials)**

**get deployment_id by service.list() method**

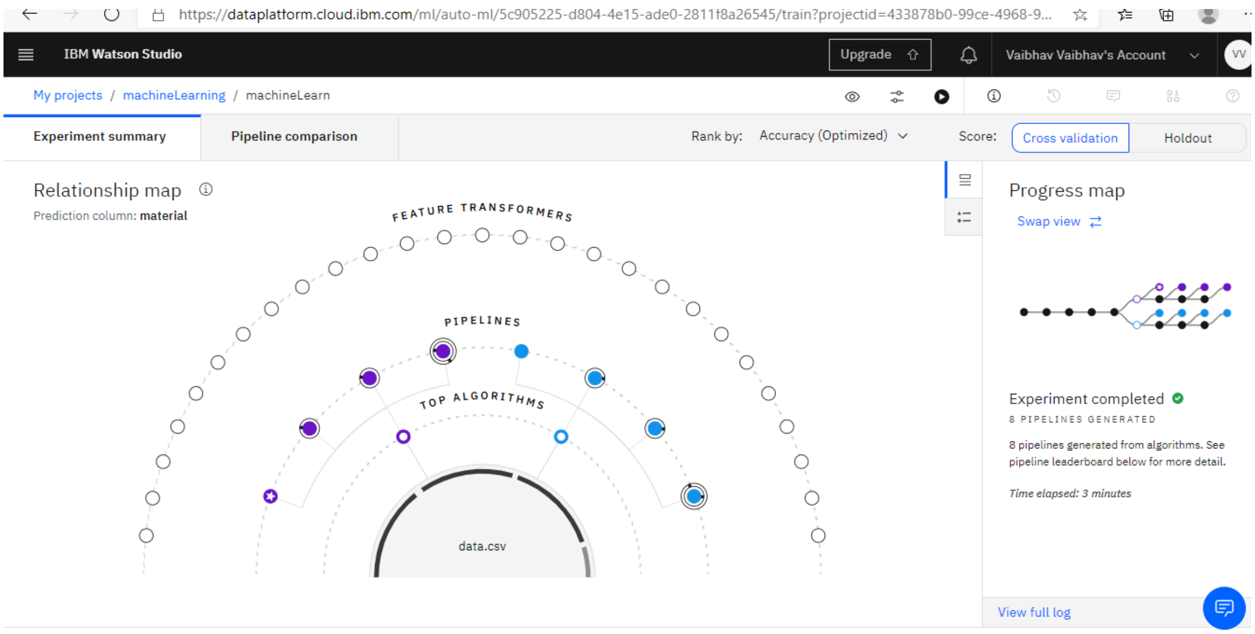
**get webservice object by service.get('deployment_id') method**

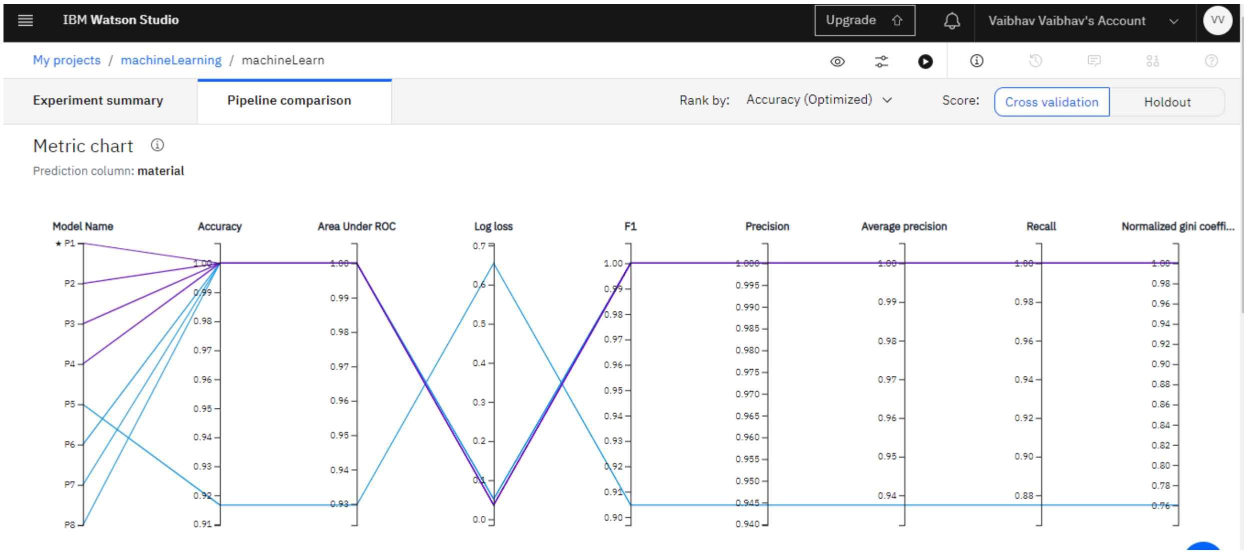
**After that you can call service.score() method.**

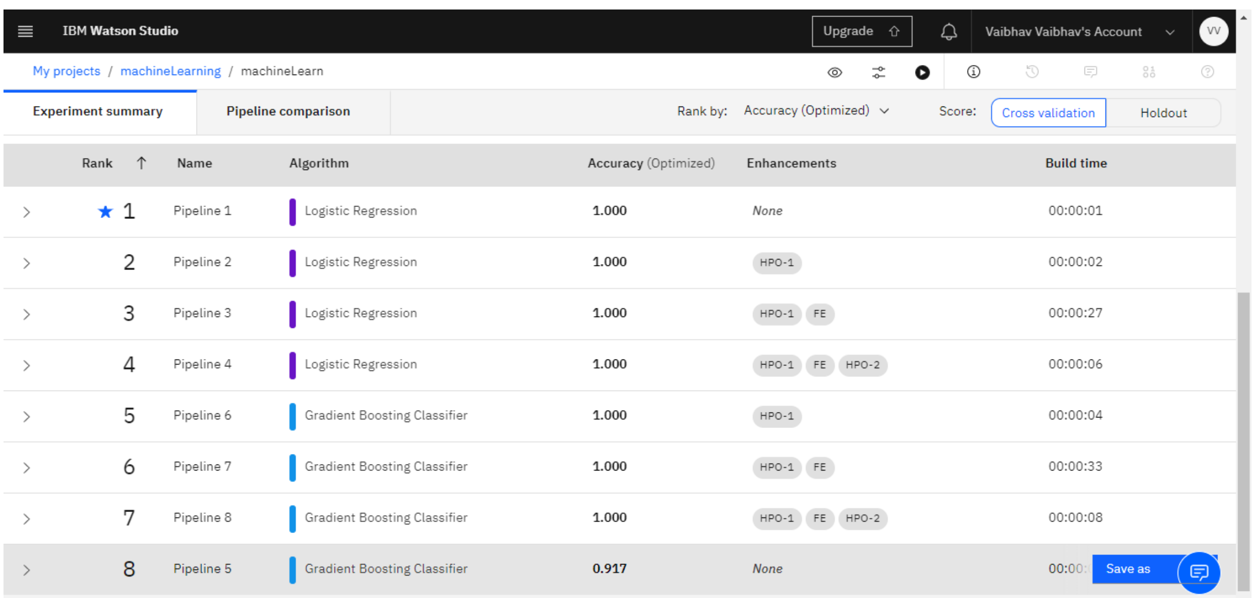
**Delete deployment**

## You can delete an existing deployment by calling service.delete().

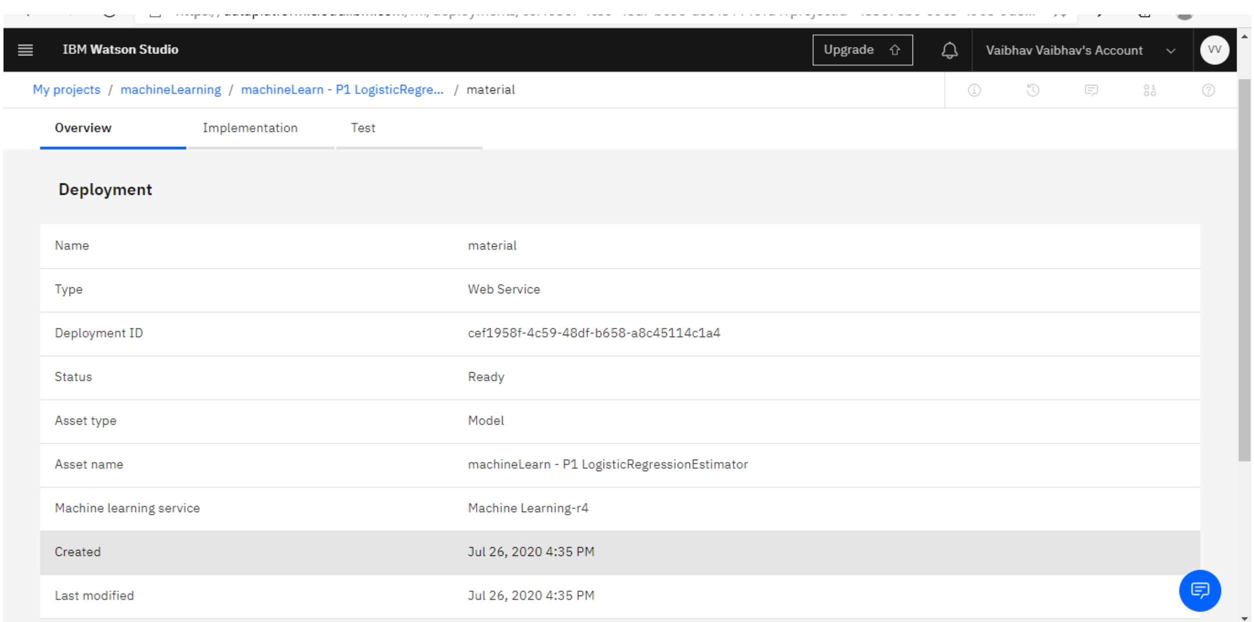
## UI SCREENSHOTS:

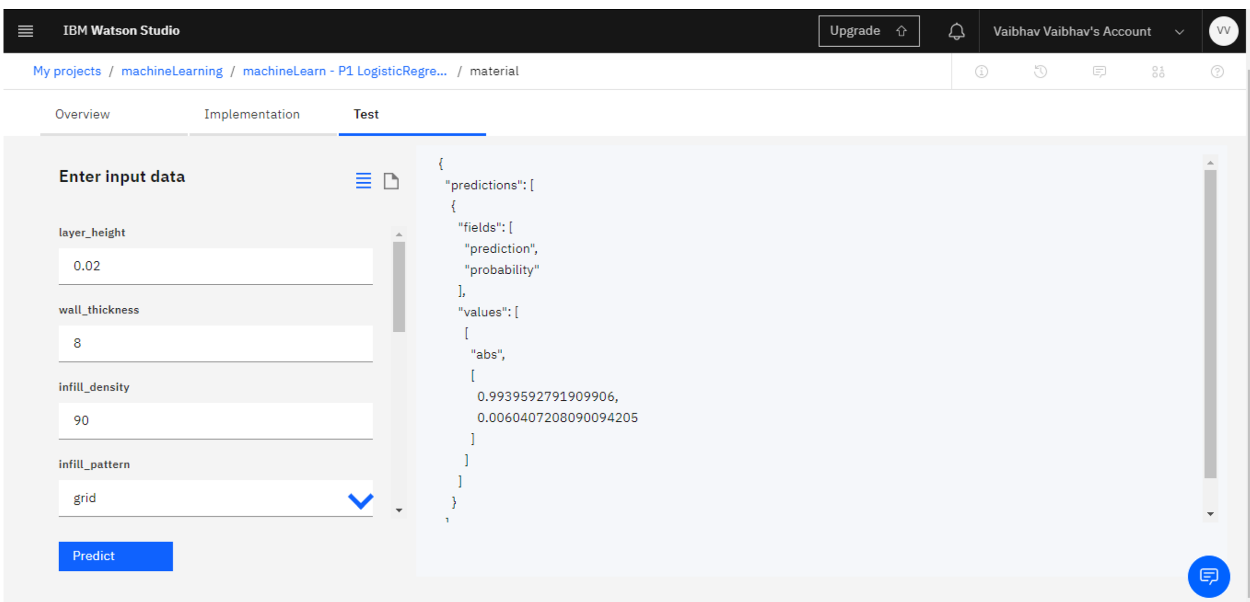


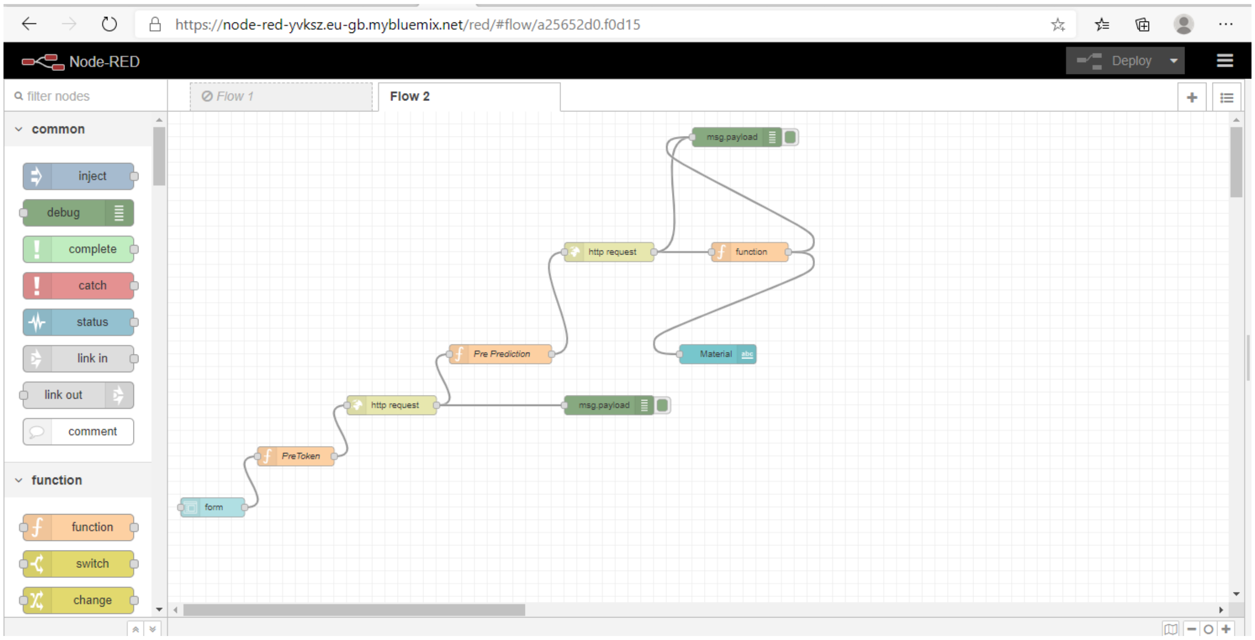


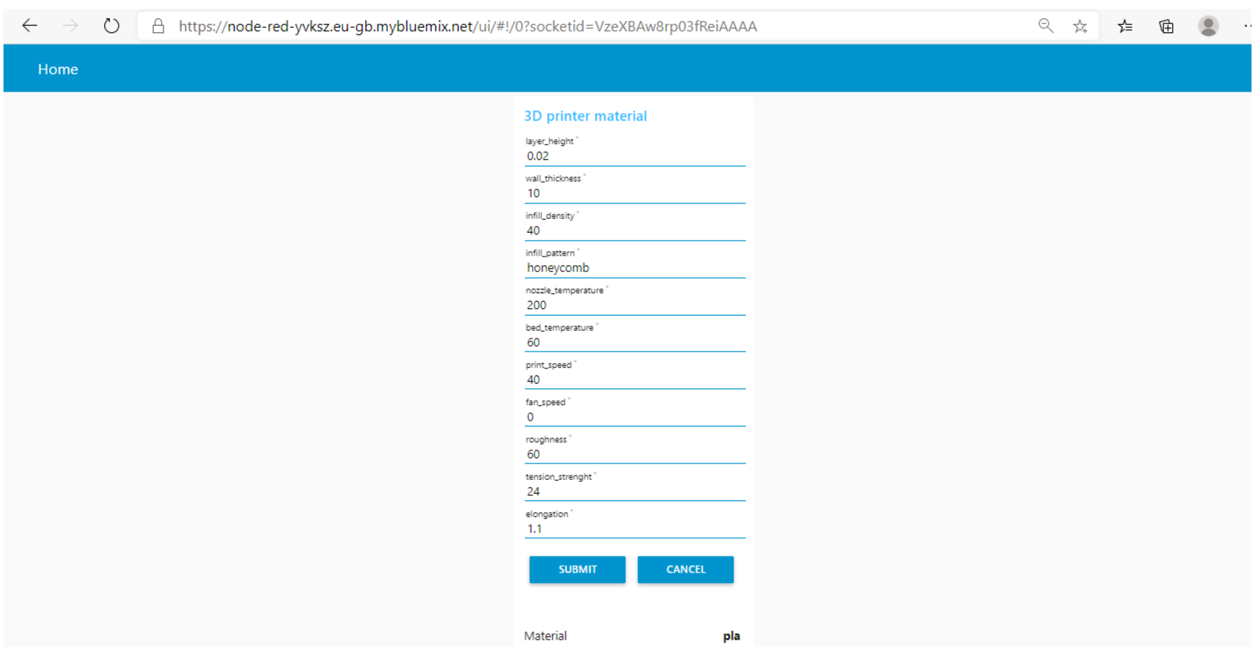




| Rank | Name | Algorithm | Accuracy (Optimized) | Enhancements | Build time |
|---|---|---|---|---|---|
| ★ 1 | Pipeline 1 | Logistic Regression | 1.000 | None | 00:00:01 |
| 2 | Pipeline 2 | Logistic Regression | 1.000 | HPO-1 | 00:00:02 |
| 3 | Pipeline 3 | Logistic Regression | 1.000 | HPO-1 FE | 00:00:27 |
| 4 | Pipeline 4 | Logistic Regression | 1.000 | HPO-1 FE HPO-2 | 00:00:06 |
| 5 | Pipeline 6 | Gradient Boosting Classifier | 1.000 | HPO-1 | 00:00:04 |
| 6 | Pipeline 7 | Gradient Boosting Classifier | 1.000 | HPO-1 FE | 00:00:33 |
| 7 | Pipeline 8 | Gradient Boosting Classifier | 1.000 | HPO-1 FE HPO-2 | 00:00:08 |
| 8 | Pipeline 5 | Gradient Boosting Classifier | 0.917 | None | 00:00: |


IBM Watson Studio
Upgrade
Vaibhav Vaibhav's Account
VV
My projects / machineLearning / machineLearn - P1 LogisticRegre... / material
Overview
Implementation
Test
Deployment
Name
material
Type
Web Service
Deployment ID
cef1958f-4c59-48df-b658-a8c45114c1a4
Status
Ready
Asset type
Model
Asset name
machineLearn - P1 LogisticRegressionEstimator
Machine learning service
Machine Learning-r4
Created
Jul 26, 2020 4:35 PM
Last modified
Jul 26, 2020 4:35 PM


IBM Watson Studio
Upgrade
Vaibhav Vaibhav's Account
VV
My projects / machineLearning / machineLearn - P1 LogisticRegre... / material
Overview
Implementation
Test
Enter input data
layer_height
0.02
wall_thickness
8
infill_density
90
infill_pattern
grid
Predict
{
"predictions": [
{
"fields": [
"prediction",
"probability"
],
"values": [
[
"abs",
[
0.9939592791909906,
0.0060407208090094205
]
]
]
}


https://node-red-yvksz.eu-gb.mybluemix.net/red/#flow/a25652d0.f0d15
Node-RED
Deploy
filter nodes
Flow 1
Flow 2
common
inject
debug
complete
catch
status
link in
link out
comment
function
function
switch
change
msg.payload
http request
function
Pre Prediction
Material
http request
msg.payload
PreToken
form


https://node-red-yvksz.eu-gb.mybluemix.net/ui/#!/0?socketid=VzeXBAw8rp03fReiAAAA
Home
3D printer material
layer_height *
0.02
wall_thickness *
10
infill_density *
40
infill_pattern *
honeycomb
nozzle_temperature *
200
bed_temperature *
60
print_speed *
40
fan_speed *
0
roughness *
60
tension_strenght *
24
elongation *
1.1
SUBMIT
CANCEL
Material
pla