I. Introduction

I.I. Overview

This project discusses building a system for creating predictions that can be used in different scenarios. It focuses on predicting fraudulent transactions, which can reduce monetary loss and risk mitigation.

I.II. Purpose

This project aims at building a web App which automatically estimates if there is a fraud risk by taking the input values. [1]

II. Literature Survey

II.I. Existing problem

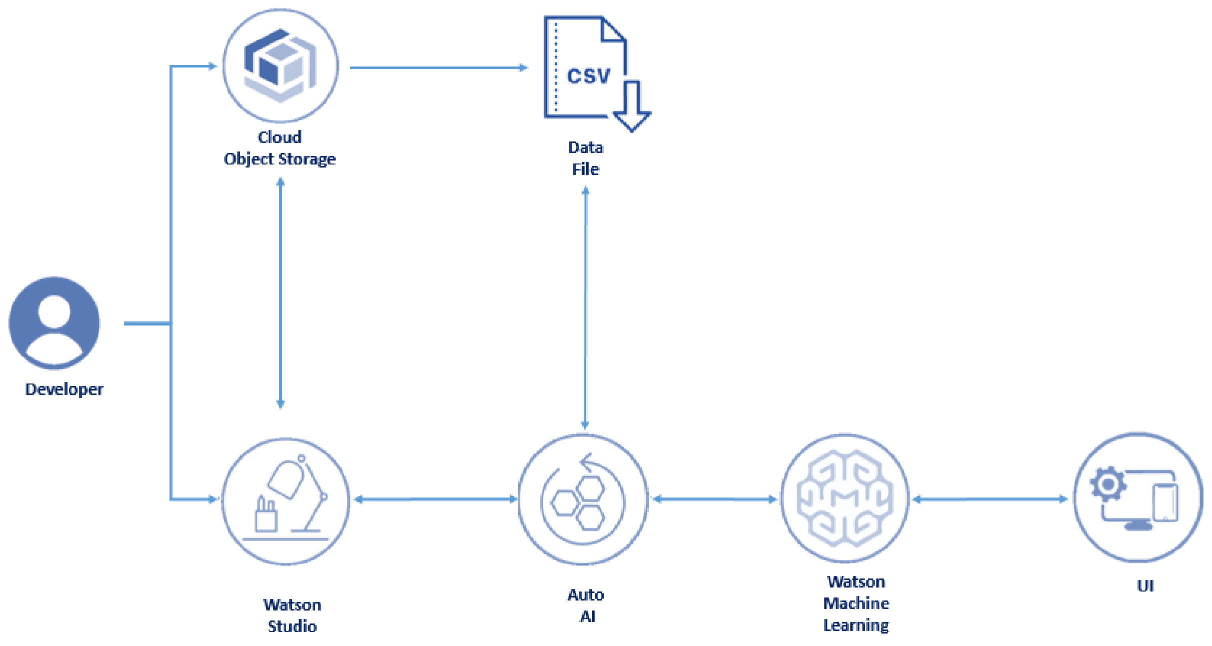
The majority of detection methods combine a variety of fraud detection datasets to form a connected overview of both valid and non-valid payment data to make a decision. This decision must consider IP address, geolocation, device identification, “BIN” data, global latitude/longitude, historic transaction patterns, and the actual transaction information. In practice, this means that merchants and issuers deploy analytically based responses that use internal and external data to apply a set of business rules or analytical algorithms to detect fraud. [2]

II.II. Proposed solution

Using IBM AutoAI, all of the tasks involved in building predictive models for different requirements are automated. The model is generated from a data set that includes the gender, married, dependents, education, self employed, applicant income, co-applicant income, loan amount, loan term, credit history, housing and locality, and predicts the fraud risk.

III. Theoretical analysis

III.I. Block diagram



III.II. Hardware/Software designing

The technologies used in this project are:

- IBM Cloud: A set of cloud computing services for business offered by the information technology company IBM. It combines platform as a service (PaaS) with infrastructure as a service (IaaS).

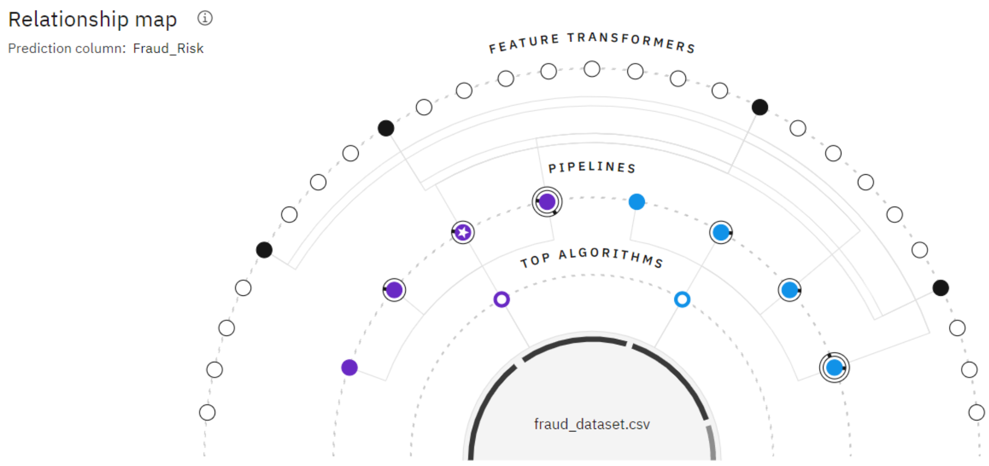
- IBM Watson Studio: IBM's software platform for data science. The platform consists of a workspace that includes multiple collaboration and open-source tools for use in data science.

- IBM Watson Machine Learning: A service that enables you to create, train, and deploy self-learning models using an automated, collaborative workflow.

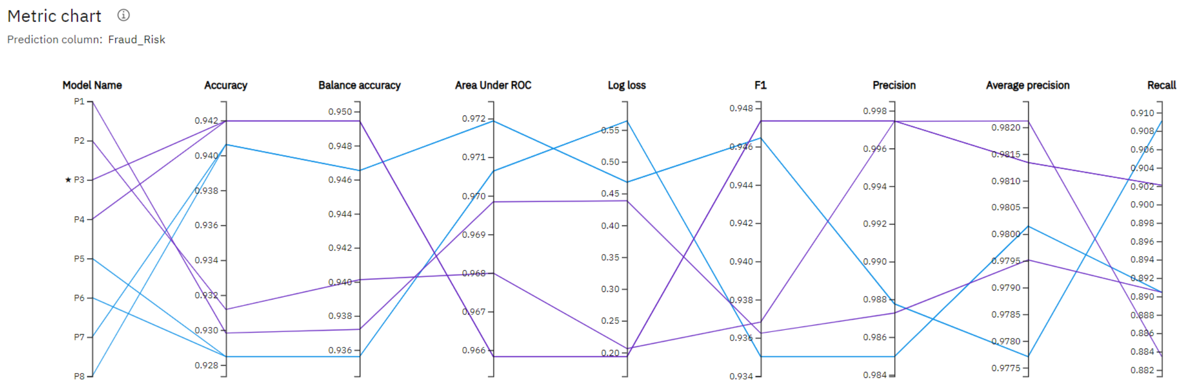
- **Node-RED:** A flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things.

IV. Experimental investigations

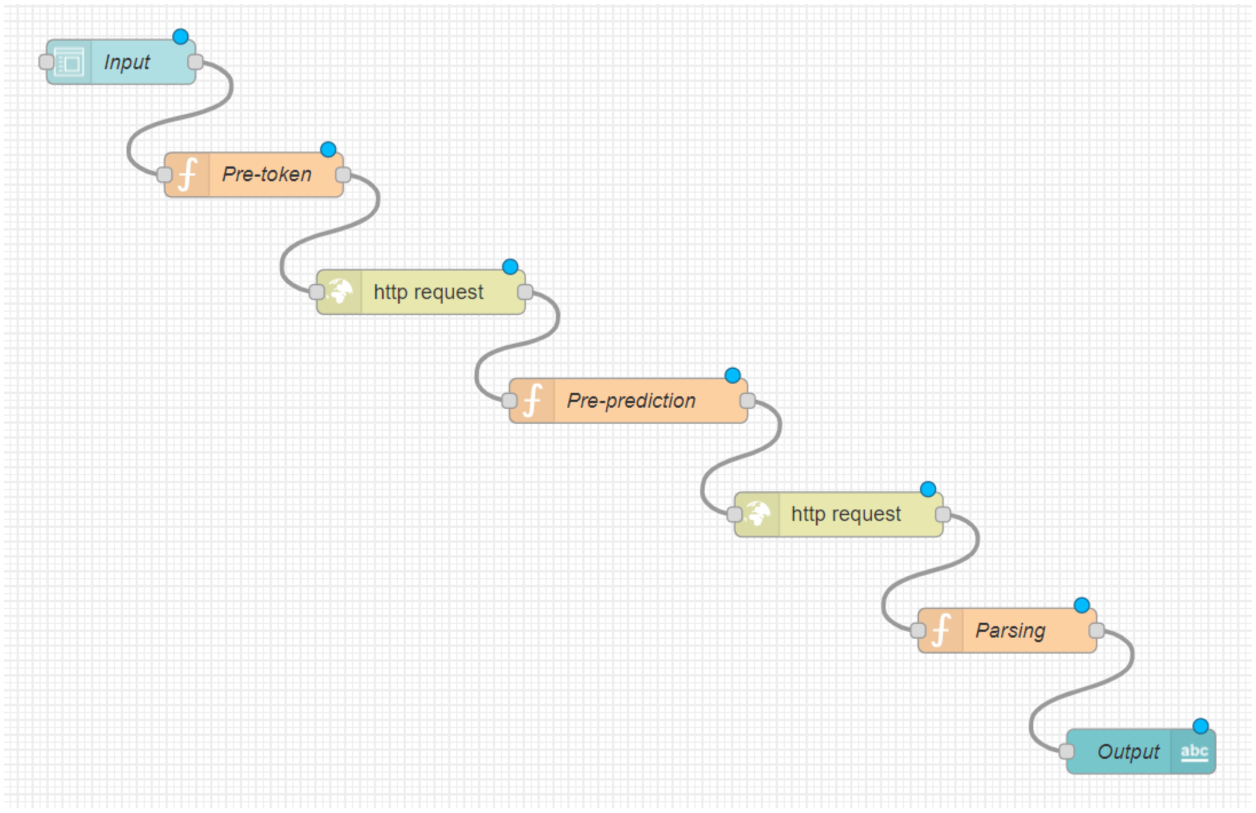
IV. I. Experiment summary



IV. II. Pipeline comparison



V. Flowchart



The main flowchart is composed of:

- Input node: Contains the gender, married, dependents, education, self employed, applicant income, co-applicant income, loan amount, loan term, credit history, housing and locality parameters.

- Pre-token function node:

```
global.set("Gender", msg.payload.Gender)
global.set("Married", msg.payload.Married)
global.set("Dependents", msg.payload.Dependents)
global.set("Education", msg.payload.Education)
global.set("Self_Employed", msg.payload.Self_Employed)
global.set("ApplicantIncome", msg.payload.ApplicantIncome)
global.set("CoapplicantIncome", msg.payload.CoapplicantIncome)
global.set("LoanAmount", msg.payload.LoanAmount)
global.set("Loan_Term", msg.payload.Loan_Term)
global.set("Credit_History_Available",
msg.payload.Credit_History_Available)
global.set("Housing", msg.payload.Housing)
global.set("Locality", msg.payload.Locality)

var apikey = "XfSY87O3aifMDSI0RVQz_HR3Vn5HFEA1WOR098d0IzFZ";
msg.headers = {"content-type": "application/x-www-form-urlencoded"}
msg.payload = {"grant_type": "urn:ibm:params:oauth:grant-type:apikey",
"apikey": apikey}
return msg;
```

- Http request node: Configured with POST method and given the URL address: "https://iam.cloud.ibm.com/identity/token". It returns a parsed JSON object.

- Pre-prediction function node:

```
var Gender = global.get("Gender")
var Married = global.get("Married")
var Dependents = global.get("Dependents")
var Education = global.get("Education")
var Self_Employed = global.get("Self_Employed")
var ApplicantIncome = global.get("ApplicantIncome")
var CoapplicantIncome = global.get("CoapplicantIncome")
var LoanAmount = global.get("LoanAmount")
var Loan_Term = global.get("Loan_Term")
var Credit_History_Available = global.get("Credit_History_Available")
var Housing = global.get("Housing")
var Locality = global.get("Locality")

var token = msg.payload.access_token
var instance_id = "8c3fecba-c13d-4399-8b65-9e6dc300e5dc"
msg.headers = {"Content-type": "application/json", "Authorization":
"Bearer" + token, "ML-Instance-ID": instance_id}
msg.payload = {"input_data": [{"fields": ["Gender", "Married",
"Dependents", "Education", "Self_Employed", "ApplicantIncome",
"CoapplicantIncome", "LoanAmount", "Loan_Term",
"Credit_History_Available", "Housing", "Locality"], "values": [[Gender,
Married, Dependents, Education, Self_Employed, ApplicantIncome,
CoapplicantIncome, LoanAmount, Loan_Term, Credit_History_Available,
Housing, Locality]]}]}
return msg;
```

- Http request node: Configured with POST method and given the model deployment URL address: "https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/8c3fecba-c13d-4399-8b65-9e6dc300e5dc/predictions?version=2021-03-04". It returns a parsed JSON object.

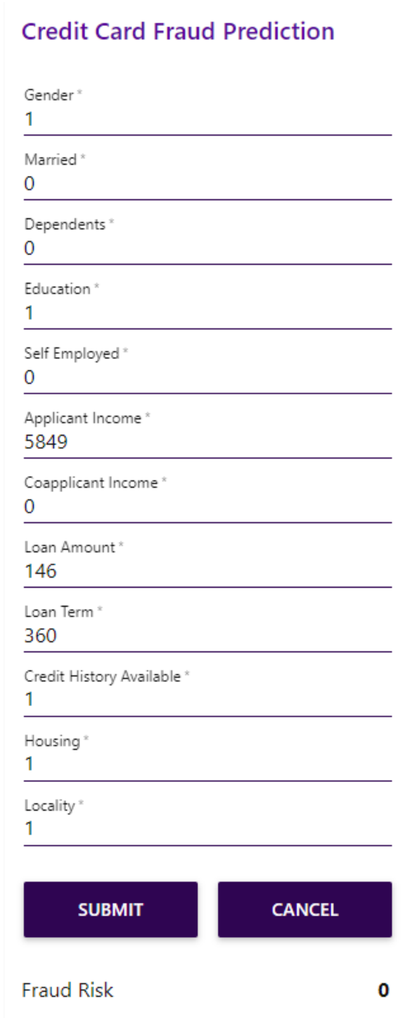
- Parsing function node:

```
msg.payload = msg.payload.predictions[0].values[0][0]
return msg;
```

- Output node: Contains the fraud risk prediction value.

VI. Results

By testing the 1st, 4th and 9th lines of fraud_dataset.cv, we find the same prediction values of the fraud risk, proving the accuracy of the machine learning model.

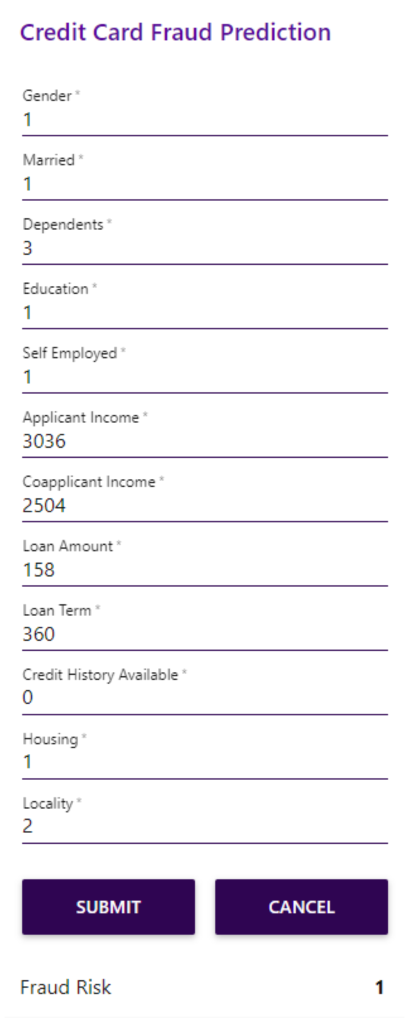


Credit Card Fraud Prediction

Gender *	1
Married *	0
Dependents *	0
Education *	1
Self Employed *	0
Applicant Income *	5849
Coapplicant Income *	0
Loan Amount *	146
Loan Term *	360
Credit History Available *	1
Housing *	1
Locality *	1

SUBMIT CANCEL

Fraud Risk 0

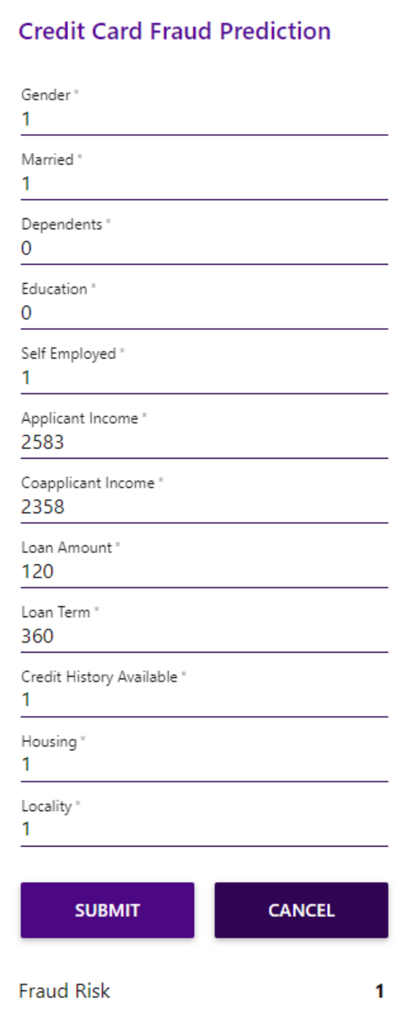


Credit Card Fraud Prediction

Gender *	1
Married *	1
Dependents *	3
Education *	1
Self Employed *	1
Applicant Income *	3036
Coapplicant Income *	2504
Loan Amount *	158
Loan Term *	360
Credit History Available *	0
Housing *	1
Locality *	2

SUBMIT CANCEL

Fraud Risk 1



Credit Card Fraud Prediction

Gender *	1
Married *	1
Dependents *	0
Education *	0
Self Employed *	1
Applicant Income *	2583
Coapplicant Income *	2358
Loan Amount *	120
Loan Term *	360
Credit History Available *	1
Housing *	1
Locality *	1

SUBMIT CANCEL

Fraud Risk 1

VII. Advantages and disadvantages

VII. I. Advantages

- Fast model selection: The top performing models are quickly selected.

- Quick start

- AI lifecycle management: Enforced consistency and repeatability of end-to-end machine learning and AI development. [3]

VII. II. Disadvantages

- Maintenance

- Large data: Working with large files takes time to upload and process especially on limited resources

- Indirect structured data process

VIII. Applications

This solution is adaptable to many fields like healthcare, education, construction, retail etc... that require a high level of precision and aim at optimizing resources, services or sales or reducing risks and losses.

IX. Conclusion

This project is a solution to predicting the risk of fraudulent transactions through credit cards, by using IBM Watson Machine Learning Service and Node-RED.

X. Future scope

Machine learning techniques have been used to detect credit card frauds but no fraud detection systems have been able to offer great efficiency to date. Recent development of deep learning has been applied to solve complex problems in various areas.

Experimental results show great performance of some deep learning methods against traditional machine learning models and imply that the proposed approaches can be implemented effectively for real-world credit card fraud detection systems. [4]

XI. Bibliography

[1] Smartinternz.com. 2021. SmartInternz. [online] Available at: <https://smartinternz.com/ibm-project/83> [Accessed 7 March 2021].

[2] Chuprina, R., 2020. Credit Card Fraud Detection and Prevention: The Complete Guide.

[online] DEV Community. Available at: <https://dev.to/velikachuprina/credit-card-fraud-detection-and-prevention-the-complete-guide-4nn4> [Accessed 7 March 2021].

[3] Syamala, M., 2020. SmartPracticeschool/SPS-6811-Credit-Card-Fraud-Prediction-using-IBM-Auto-AI. [online] GitHub. Available at: <https://github.com/SmartPracticeschool/SPS-6811-Credit-Card-Fraud-Prediction-using-IBM-Auto-AI/blob/main/SPS-6811-Credit.Card.Fraud.Prediction.using.IBM.Auto.AI.pdf?fbclid=IwAR1q3vrdTp_XmlJHFerMyXxaKCFuQfuBQkFhfKPyA0Kfv7EeCFtUuGB1r3M> [Accessed 7 March 2021].

[4] Nguyen, T., Tahir, H., Abdelrazek, M. and Babar, A., 2021. Deep Learning Methods for Credit Card Fraud Detection. [online] arXiv.org. Available at: <https://arxiv.org/abs/2012.03754> [Accessed 7 March 2021].

Appendix

Source code

```
# In[1]:

get_ipython().system('pip install ibm-watson-machine-learning | tail -n 1')
get_ipython().system('pip install -U autoai-libs | tail -n 1')

# In[2]:

# @hidden_cell
from ibm_watson_machine_learning.helpers import DataConnection, S3Connection, S3Location

training_data_reference = [DataConnection(
    connection=S3Connection(
        api_key='f4kPKHTWcZ-s8bANuAxYy2hmZ5rabyc5Y90LNoJM1hRa',
        auth_endpoint='https://iam.bluemix.net/oidc/token/',
        endpoint_url='https://s3.eu-geo.objectstorage.softlayer.net'
    ),
        location=S3Location(
bucket='creditcardfraudpredictionusingibm-donotdelete-pr-qdxryymophsglr',
        path='fraud_dataset.csv'
    )),
]
training_result_reference = DataConnection(
    connection=S3Connection(
        api_key='f4kPKHTWcZ-s8bANuAxYy2hmZ5rabyc5Y90LNoJM1hRa',
        auth_endpoint='https://iam.bluemix.net/oidc/token/',
        endpoint_url='https://s3.eu-geo.objectstorage.softlayer.net'
    ),
    location=S3Location(
bucket='creditcardfraudpredictionusingibm-donotdelete-pr-qdxryymophsglr',
path='auto_ml/7fb173b8-8364-4122-9c74-212c395e91b9/wml_data/0bea83c5-c9d7-4953-adbc-7d9b378a661a/data/automl',
model_location='auto_ml/7fb173b8-8364-4122-9c74-212c395e91b9/wml_data/0bea83c5-c9d7-4953-adbc-7d9b378a661a/data/automl/pre_hpo_d_output/Pipeline1/model.pickle',
training_status='auto_ml/7fb173b8-8364-4122-9c74-212c395e91b9/wml_data/0bea83c5-c9d7-4953-adbc-7d9b378a661a/training-status.json'
    ))

# In[3]:

experiment_metadata = dict(
   prediction_type='classification',
   prediction_column='Fraud_Risk',
   test_size=0.1,
   scoring='accuracy',
    project_id='3f1878b3-49ad-401a-b3a0-3fa8b5375639',
    deployment_url='https://eu-gb.ml.cloud.ibm.com',
    csv_separator=',',
    random_state=33,
    max_number_of_estimators=2,
    daub_include_only_estimators=None,
    training_data_reference=training_data_reference,
    training_result_reference=training_result_reference,
    positive_label=1
)

# In[4]:

api_key = '918Qbg9vw65-heOPvtePjYEDaJ6JmdqprdUZIpp_5-JY'

# In[6]:

wml_credentials = {
    "apikey": api_key,
    "url": experiment_metadata['deployment_url']
}

# In[7]:

from ibm_watson_machine_learning.experiment import AutoAI

pipeline_optimizer = AutoAI(wml_credentials, project_id=experiment_metadata['project_id']).runs.get_optimizer(metadata=experiment_metadata)

# In[8]:

pipeline_optimizer.get_params()

# In[9]:

summary = pipeline_optimizer.summary()
best_pipeline_name = list(summary.index)[0]
```

```
summary

# In[10]:

summary[f"holdout_{experiment_metadata['scoring'].replace('neg_','')}"]
 .plot();

# In[11]:

pipeline_model = pipeline_optimizer.get_pipeline()

# In[12]:

pipeline_optimizer.get_pipeline_details()['features_importance']

# In[13]:

pipeline_model.visualize()

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

# In[15]:

target_space_id = "06a92013-4b5b-44f7-8d77-0d3a5d6cbb40"

from ibm_watson_machine_learning.deployment import WebService
service = WebService(source_wml_credentials=wml_credentials,
                     target_wml_credentials=wml_credentials,

source_project_id=experiment_metadata['project_id'],
                     target_space_id=target_space_id)
service.create(
model=best_pipeline_name,
metadata=experiment_metadata,
deployment_name='Best_pipeline_webservice'
)

# In[16]:
```

```

print(service)

# In[17]:

service.get_params()
```