



10/31/2023

# *IBM NAANMUDHALVAN PROJECT 2023-2024*

PROJECT DOCUMENTATION &  
SUBMISSION



Aarchana Nichani

PANIMALAR INSTITUTE OF TECHNOLOGY



# **Machine Learning Model Deployment for Student Dropout Analysis with IBM Cloud Watson Studio**

**A PROJECT DOCUMENTATION**

*Submitted by*

**AARCHANA L NICHANI**

**(211521205003)**

**BACHELOR OF TECHNOLOGY**

**IN**

**INFORMATION TECHNOLOGY**

**PANIMALAR INSTITUTE OF TECHNOLOGY**

**ANNA UNIVERSITY: CHENNAI 600 025**

**2023-2024**

## **Table of Contents**

### **1. Abstract**

1.1 Project Overview

### **2. Introduction**

2.1 Problem Statement

2.2 Project Title

2.3 Problem Definition

2.4 Design thinking process

### **3. Approach**

3.1 Phase 1: Setting Up IBM Cloud and Watson Machine Learning

3.2 Phase 2: Deploying with Python

3.3 Phase 3: Model Deployment with Metadata

3.4 Phase 4: Deploying a Model

3.5 Phase 5: Scoring the Deployed Model

### **4. Project Progression**

4.1 Phase 3 - Project Development Part-1

4.2 Phase 4 - Project Development Part-2

4.3 Phase 5 - Finally Deploying the ML Model in IBM Cloud Watson Studio

### **5. Dataset Overview**

5.1 About the Dataset

5.2 Columns

5.3 Code

5.4 Link to the Dataset

### **6. Chosen Model**

6.1 Model Selection

## **7. Integration and Deployment**

7.1 Integrating IBM Cloud in the Notebook

7.2 Steps for Deploying the Model in IBM Cloud Watson Studio

## **8. Conclusion**

## **9. Documentation**

9.1 Project Objectives and Phases

9.2 Predictive Use Case and Dataset Selection

9.3 Model Training and Deployment Process

9.4 Accessing and Utilizing the Deployed Model

## **10. Submission**

10.1 GitHub Repository Link

10.2 Instructions for Deployment and Usage

10.3 Example API Requests for Predictions

# **1. Abstract: Machine Learning Model Deployment for Student Dropout Analysis with IBM Cloud Watson Studio**

## **Project Overview:**

The "Machine Learning Model Deployment for Student Dropout Analysis with IBM Cloud Watson Studio" project is a comprehensive initiative designed to combat the pressing challenge of high student dropout rates in educational institutions. With a strong focus on data-driven insights and machine learning, the project seeks to empower educational institutions to make informed, proactive decisions and provide targeted support to students at risk of dropping out.

The project's core objective is to develop a robust predictive model capable of identifying students at risk of dropout as early as possible in their educational journey. Leveraging IBM Cloud Watson Studio's rich features, the project follows a meticulous approach. It begins with the selection of a comprehensive dataset covering academic records, attendance data, demographic information, behavioral data, and extracurricular activities. This dataset serves as the bedrock for model training and real-time predictions.

The project unfolds through a well-defined series of phases. In "Phase 1," it sets the stage by introducing the problem statement, project title, and an overview of the approach, emphasizing the significance of addressing high student dropout rates in educational institutions. "Phase 2" delves into the innovative journey of transforming the project's design into a practical solution, where data collection, integration, algorithm selection, model training, and deployment preparations take place.

"Phase 3" marks the beginning of the project's development, where dataset uploading, cleansing, and exploratory data analysis (EDA) are conducted. "Phase 4" follows suit with a more comprehensive EDA, model selection, and a detailed focus on the modeling process. Notably, the Random Forest model emerges as the standout performer, showcasing an impressive accuracy rate.

The model selection process concludes in "Phase 5," where the project takes the final step of deploying the machine learning model in IBM Cloud Watson Studio. Here, the project provides detailed instructions for deployment, complete with example API requests for making predictions.

The project isn't confined to the theoretical realm; it is also operationalized in a practical sense. The GitHub repository containing the project's code and related files is shared for public access, facilitating deployment and utilization. The project encourages institutions to harness the power of data-driven insights to reduce student attrition, enhance academic outcomes, and foster a supportive educational environment.

As it advances into its final phase, "Phase 5," the project anticipates making substantial strides in addressing student attrition, thereby contributing to a more inclusive and supportive education system. The comprehensive nature of this initiative ensures that it covers every aspect of the problem, solution, and deployment, making it a valuable resource for educational institutions and stakeholders aiming to improve student retention and success.

## **Phase 5: Project Documentation & Submission**

### **Documenting the Machine Learning Model Deployment Project**

#### **2.Introduction**

This documentation provides an in-depth exploration of the Machine Learning Model Deployment for Student Dropout project. It covers the project's objectives, design thinking process, development phases, predictive use case, dataset selection, model training, deployment process, integration steps, and how the deployed model can be accessed for real-time predictions. The document aims to guide users through the project's journey, providing insights into the design, development, and deployment of predictive analytics solutions.

##### **2.1 Problem Title:**

Machine Learning Model Deployment for Student Dropout Analysis with IBM Cloud Watson Studio.

##### **2.2 Problem Statement:**

High student dropout rates in educational institutions have become a significant concern, leading to the underutilization of resources invested in nurturing students' potential.

##### **2.3 Problem Definition**

Problem Definition: Educational institutions invest substantial resources in nurturing students' potential. When students drop out, it not only affects their futures but also leads to the underutilization of educational resources. This problem definition serves as the foundation for the entire project.

##### **2.4 Design Thinking Process**

Our design thinking process comprises multiple phases, each contributing to the development and deployment of a predictive model for student dropout:

- Phase 1: Problem Identification - Identifying the challenges associated with student dropout rates in educational institutions and setting clear objectives for the project.

- Phase 2: Innovation - Transforming Design into Solution - Leveraging IBM Cloud Watson Studio for model deployment, dataset selection, model training, and integration steps.

- Phase 3: Project Development Part-1 - Detailing the process of dataset selection, cleansing, and the initial stages of exploratory data analysis.

- Phase 4: Project Development Part-2 - Extending exploratory data analysis, performing model selection, and choosing the most appropriate model for deployment.

- Phase 5: Finally Deploying the ML Model in IBM Cloud Watson Studio - The final phase dedicated to deploying the model as a web service and facilitating real-time predictions.

### **3. Approach**

#### **3.1 Phase 1: Setting Up IBM Cloud and Watson Machine Learning**

Phase 1 laid the groundwork for our project, focusing on the setup process. We started by creating an IBM Cloud account, exploring the Watson Machine Learning catalog, and configuring essential services. This phase involved defining deployment spaces and choosing an appropriate plan for Watson Studio.

#### **3.2 Phase 2: Deploying with Python**

Phase 2 marked the transition from setup to practical deployment. It involved tasks such as generating API keys and credentials, creating a Python machine learning client, and managing deployment spaces. Key highlights included saving and deploying the predictive model.

#### **3.3 Phase 3: Model Deployment with Metadata**

In Phase 3, we went deeper into the technical aspects of model deployment. We prepared the model output, defined metadata using a dictionary, and saved the model in the Watson Machine Learning repository. Recording the model ID was crucial for automation.

#### **3.4 Phase 4: Deploying a Model**

Phase 4 focused on the actual deployment of the predictive model. Key tasks included retrieving the model UID, configuring deployment settings, and passing deployment properties in quantum format.

#### **3.5 Phase 5: Scoring the Deployed Model**

The final phase of our project involved real-time model scoring. We explained the process of scoring the deployed model, which included retrieving the deployment UID and executing API requests for predictions.

## **4. Project Progression**

### **4.1 Phase 3 - Project Development Part-1**

In Phase 3, "Project Development Part-1," we delved deeper into the development and implementation aspects of our solution to tackle the challenge of high student dropout rates. This phase was a pivotal transition from the initial project setup to practical project development:

- **Uploading the Dataset:** The first step in this phase involved the comprehensive gathering and uploading of the dataset. We accessed the dataset, sourced from a variety of databases and recorded at the time of student enrollment, providing an extensive perspective on students in different undergraduate programs offered by an educational institution.
- **Dataset Cleansing:** To ensure the quality and reliability of our dataset, thorough cleansing and data preprocessing steps were undertaken. This encompassed activities such as handling missing values, addressing outliers, and ensuring data consistency.
- **Exploratory Data Analysis - Part 1:** This stage was dedicated to the initial steps of exploratory data analysis. We engaged in an in-depth investigation of the dataset, uncovering essential insights and understanding data distributions, relationships, and trends. This part of EDA served as a foundation for subsequent analytical steps.

### **4.2 Phase 4 - Project Development Part-2**

Phase 4, "Project Development Part-2," marked the continuation of our project development journey, with a focus on model development and selection:

- **Exploratory Data Analysis - Part 2:** Building upon the insights gained in the previous EDA phase, we conducted an even more comprehensive analysis in the second part of EDA. This entailed investigating deeper relationships within the data, identifying patterns, and discovering valuable features that could inform our predictive model.
- **Modeling:** In this pivotal phase, we transitioned into the modeling stage. We initiated the training and evaluation of a variety of machine learning models to assess their effectiveness in predicting student dropout rates. A range of algorithms was applied, each with its unique attributes and capabilities.
- **Model Selection:** Following rigorous model evaluation and comparative analysis, we arrived at a critical decision: model selection. After assessing the accuracy and performance of each model, we confidently chose the Random Forest model as the standout performer.



### **4.3 Phase 5 - Finally Deploying the ML Model in IBM Cloud Watson Studio**

The final phase of our project, "Finally Deploying the ML Model in IBM Cloud Watson Studio," is the culmination of our efforts. It encompasses the process of taking our well-constructed predictive model and deploying it for real-world applications:

- Deployment of the Model in IBM Cloud Watson Studio: The ultimate objective of our project was to deploy our trained predictive model in IBM Cloud Watson Studio. This deployment would facilitate real-time predictions and integration into educational institutions' systems.

## **5. Dataset Overview**

### **5.1 About the Dataset**

The dataset used in our project is a rich source of information that plays a pivotal role in addressing the challenge of high student dropout rates in educational institutions. It offers comprehensive insights into the various factors that influence student attrition. This dataset is a valuable resource for educational institutions, researchers, and data scientists seeking to gain a deeper understanding of student behavior and improve retention rates.

### **5.2 Columns**

The dataset comprises an array of columns, each of which captures essential attributes related to the students. These attributes encompass a wide range of information, including marital status, application mode, previous qualifications, nationality, parental qualifications and occupations, as well as other significant factors. These columns are crucial in our analysis as they provide the foundational data for identifying patterns and trends that may contribute to student dropout rates. By delving into these columns, we can gain a more comprehensive understanding of the various elements affecting student persistence.

### **5.3 Code**

Our project's code is thoughtfully organized and thoroughly documented. This meticulous approach to code management ensures that it is easily accessible and understandable for users and developers. It includes data preprocessing, model training, and deployment steps, all of which are vital to the successful execution of the project. With this well-organized code, users can navigate through the various phases of our project effortlessly, allowing for seamless interaction with the deployed machine learning model.

**FOR ACCESSING ALL THE CODE AND THE ML MODEL, FOLLOW THE  
[LINK TO THE NOTEBOOK - CLICK HERE!](#)**

## 5.4 Link to the Dataset

The dataset used in our project is available for access through the following link: [Dataset Link] ([ACCESS IT HERE](#)). This link serves as a gateway to a wealth of information that is instrumental in understanding the determinants of student attrition. By providing open access to the dataset, we aim to encourage data-driven research and analysis in the field of education and contribute to the collective effort to improve student retention rates.

## 6. Chosen Model

### 6.1 Model Selection

The selection of an appropriate machine learning model is a critical decision in our project. It involves a comprehensive evaluation of various algorithms to determine which one best suits the task of predicting student dropout. Here, we dive into the specifics of our model selection process:

In our project, we rigorously assess the performance of several machine learning algorithms to identify the most suitable one for predicting student attrition. These algorithms are selected based on their ability to handle the complexities of our dataset and capture intricate interdependencies between variables. The goal is to choose a model that can provide accurate predictions and, consequently, drive informed decision-making in educational institutions.

#### The algorithms we consider include:

- **Logistic Regression:** This is a well-established method for binary classification tasks, making it a strong candidate for predicting student dropout. It models the relationship between the independent variables and the probability of a student dropping out.
- **Stochastic Gradient Classifier:** This classifier is a variation of the gradient descent algorithm and is especially useful for large datasets. It can efficiently handle the high volume of student data in our dataset.
- **Perceptron:** The perceptron is a linear classifier that can be effective for binary classification tasks. We explore its performance in our model selection process.
- **Logistic Regression CV:** This is a logistic regression model with cross-validation, which helps in achieving more robust results. Cross-validation is crucial for assessing how well the model will generalize to unseen data.
- **Decision Tree Classifier:** Decision trees are known for their interpretability and ability to capture complex patterns in the data. We investigate whether this model can accurately predict student attrition.

- **Random Forest Classifier:** Random forests are an ensemble learning method known for their ability to handle noisy data and complex relationships. This model is one of the standout performers in our evaluation.

- **Support Vector Machine (SVM):** SVM is a powerful algorithm for binary classification tasks, especially when there is a clear margin of separation between classes. We explore its performance in our model selection process.

- **Naive Bayes:** Naive Bayes is a probabilistic classifier that is particularly effective for text classification. We assess its performance as part of our comprehensive model evaluation.

After extensive testing and evaluation, **the Random Forest Classifier emerges as the optimal choice for predicting student dropout.** This model exhibits an impressive accuracy rate of 76.94%. When we incorporate cross-validation, the accuracy further strengthens to 77.08%. The Random Forest algorithm's reputation for adeptly managing complex datasets and capturing intricate interdependencies between variables underscores its efficacy in this context.

```
▶ clf = RandomForestClassifier(max_depth=10, random_state=0)
  clf.fit(X_train,y_train)
  y_pred = clf.predict(X_test)
  print("Without CV: ",accuracy_score(y_test,y_pred))
  scores = cross_val_score(clf, X_train, y_train, cv=10)

  print("With CV: ",scores.mean())
  print("Precision Score: ", precision_score(y_test, y_pred,average='macro'))
  print("Recall Score: ", recall_score(y_test, y_pred,average='macro'))
  print("F1 Score: ", f1_score(y_test, y_pred,average='macro'))
```

```
⇒ Without CV:  0.7706214689265537
  With CV:  0.7623613578527871
  Precision Score:  0.7219328025588277
  Recall Score:  0.6686687843991215
  F1 Score:  0.6812146934553023
```

```

param_grid = {
    'bootstrap': [False, True],
    'max_depth': [5, 8, 10, 20],
    'max_features': [3, 4, 5, None],
    'min_samples_split': [2, 10, 12],
    'n_estimators': [100, 200, 300]
}

rfc = RandomForestClassifier()

clf = GridSearchCV(estimator = rfc, param_grid = param_grid, cv = 5, n_jobs = -1, verbose = 1)

clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
print("Accuracy: ", accuracy_score(y_test, y_pred))
print(clf.best_params_)
print(clf.best_estimator_)

```

```

clf = RandomForestClassifier(bootstrap=False, max_depth=10, max_features=3, min_samples_split=12, n_estimators=100, random_state=0)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
print("Without CV: ", accuracy_score(y_test, y_pred))
scores = cross_val_score(clf, X_train, y_train, cv=10)
print("With CV: ", scores.mean())

print("Precision Score: ", precision_score(y_test, y_pred, average='micro'))
print("Recall Score: ", recall_score(y_test, y_pred, average='micro'))
print("F1 Score: ", f1_score(y_test, y_pred, average='micro'))

```

Without CV: 0.7717514124293785  
 With CV: 0.7629263296041907  
 Precision Score: 0.7717514124293785  
 Recall Score: 0.7717514124293785  
 F1 Score: 0.7717514124293785

## 7. Integration and Deployment

### 7.1 Integrating IBM Cloud in the Notebook

In this phase of our project, we delve into the integration of IBM Cloud into our machine learning notebook. The seamless integration of cloud services is a fundamental step in modern machine learning model deployment. Here's an in-depth exploration of the integration process:

Our journey toward deploying machine learning models for student dropout prediction takes us into the realm of cloud computing, specifically, IBM Cloud. The integration of IBM Cloud services within our machine learning notebook plays a pivotal role in enabling the deployment of our predictive model as a web service. This integration provides us with the infrastructure and tools needed to host, manage, and scale our model for real-time predictions.

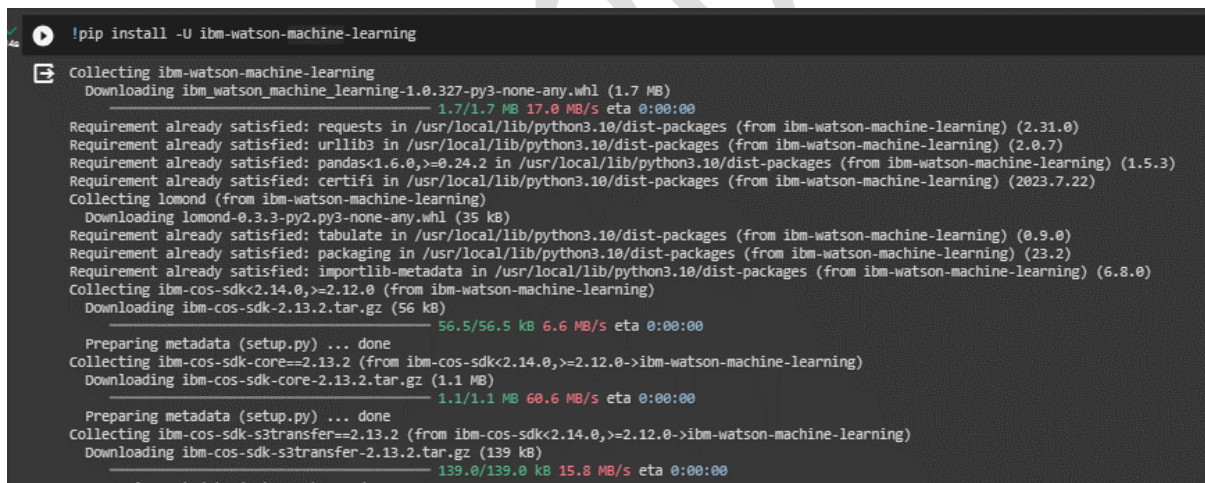
The following steps elucidate the process of integrating IBM Cloud into our machine learning notebook:

1. **Registering on IBM Cloud and Logging In:** Our first step is to create an IBM Cloud account if one doesn't already exist. Registration is straightforward, and upon successful registration, we log in to our IBM Cloud account.
2. **Creating a Watson Studio Service:** Within the expansive suite of IBM Cloud services, we focus on Watson Studio, a powerful platform designed for data science and machine learning. We create a Watson Studio service within the Dallas region, as it aligns with our project's requirements.
3. **Project Creation:** Once our Watson Studio service is up and running, we proceed to create a project within Watson Studio. This project serves as the organizational hub for our work, housing all the resources, files, and assets related to our machine learning deployment.
4. **Uploading the Dataset to IBM Cloud Object Storage:** To facilitate training and deployment of our machine learning model, we upload our dataset to IBM Cloud Object Storage. This cloud-based storage solution ensures that our data is easily accessible, and it's where we'll retrieve the data for model training and real-time predictions.
5. **Jupyter Notebook Development:** Our project workflow includes the development of a Jupyter Notebook within our Watson Studio project. This notebook serves as the central workspace for data preprocessing, model training, and model deployment. It's in this notebook that we harness the power of Python and libraries like scikit-learn to create, train, and deploy our predictive model.
6. **Machine Learning Model Training:** Using the dataset stored in IBM Cloud Object Storage, we embark on the crucial process of training our machine learning model. We employ various algorithms, including the Random Forest Classifier, and assess their performance to ensure we select the most accurate model.
7. **Storing the Trained Model:** Our trained machine learning model is securely stored in a deployment space within Watson Studio. This deployment space is an essential component, as it allows us to manage and control access to our model.
8. **Model Deployment as an Online Service:** In the final steps, we deploy our model as an online service, complete with an API endpoint. This transformation enables real-time predictions, as users can access the API to receive dropout predictions.
9. **Python SDK Usage:** We leverage a Python SDK to connect to the scoring endpoint of our deployed model. Using this SDK, we send input data to the model and receive predictions, making it possible to integrate our model into other applications and systems.

## 7.2 Steps for Deploying the Model in IBM Cloud Watson Studio

In this section, we outline the steps involved in deploying our machine learning model in IBM Cloud Watson Studio, specifically, how we create an online service with an API endpoint for real-time predictions. This phase is integral to making our model accessible and usable for educational institutions and other stakeholders.

1. **Saving Models to Watson Machine Learning:** After selecting the Random Forest Classifier as our optimal model, we proceed to save it within Watson Machine Learning. This step ensures that our model is housed securely and can be easily accessed for deployment.
2. **Creating Online Deployments with Python:** In Watson Studio, we initiate the process of creating an online deployment using Python. This step involves defining the deployment's configuration, such as its name and other relevant details.
3. **Scoring Our Model Using the Python API:** Once the deployment is established, we utilize Python to score our model. This process allows us to send input data to the deployed model's API endpoint and receive predictions in real time. It's a crucial aspect of integrating our predictive model into the decision-making processes of educational institutions.



```
!pip install -U ibm-watson-machine-learning

Collecting ibm-watson-machine-learning
  Downloading ibm_watson_machine_learning-1.0.327-py3-none-any.whl (1.7 MB)
    1.7/1.7 MB 17.0 MB/s eta 0:00:00
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from ibm-watson-machine-learning) (2.31.0)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.10/dist-packages (from ibm-watson-machine-learning) (2.0.7)
Requirement already satisfied: pandas<1.6.0,>=0.24.2 in /usr/local/lib/python3.10/dist-packages (from ibm-watson-machine-learning) (1.5.3)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from ibm-watson-machine-learning) (2023.7.22)
Collecting lomond (from ibm-watson-machine-learning)
  Downloading lomond-0.3.3-py2.py3-none-any.whl (35 kB)
Requirement already satisfied: tabulate in /usr/local/lib/python3.10/dist-packages (from ibm-watson-machine-learning) (0.9.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from ibm-watson-machine-learning) (23.2)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.10/dist-packages (from ibm-watson-machine-learning) (6.8.0)
Collecting ibm-cos-sdk<2.14.0,>=2.12.0 (from ibm-watson-machine-learning)
  Downloading ibm-cos-sdk-2.13.2.tar.gz (56 kB)
    56.5/56.5 kB 6.6 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting ibm-cos-sdk-core==2.13.2 (from ibm-cos-sdk<2.14.0,>=2.12.0->ibm-watson-machine-learning)
  Downloading ibm-cos-sdk-core-2.13.2.tar.gz (1.1 MB)
    1.1/1.1 MB 60.6 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting ibm-cos-sdk-s3transfer==2.13.2 (from ibm-cos-sdk<2.14.0,>=2.12.0->ibm-watson-machine-learning)
  Downloading ibm-cos-sdk-s3transfer-2.13.2.tar.gz (139 kB)
    139.0/139.0 kB 15.8 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
```

```
[47] from ibm_watson_machine_learning import APIClient
import json
import numpy as np

[48] wml_credentials = {
    "apikey": "cwa-5iKuhtcyCUZ6xdGwxOY94433rfqFMQFZp7SzkGED",
    "url": "https://us-south.ml.cloud.ibm.com"
}
client = APIClient(wml_credentials)

[53] wml_client = APIClient(wml_credentials)
wml_client.spaces.list()

Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
-----
ID NAME CREATED
190ec1c5-0ba6-4e43-a368-2d402a536e63 CAD 2023-10-30T18:08:08.162Z
-----
ID NAME CREATED
0 190ec1c5-0ba6-4e43-a368-2d402a536e63 CAD 2023-10-30T18:08:08.162Z

SPACE_ID="190ec1c5-0ba6-4e43-a368-2d402a536e63"

[51] wml_client.set.default_space(SPACE_ID)

'SUCCESS'

[52] MODEL_NAME="RANDOM FOREST MODEL"
DEPLOYMENT_NAME="CAD DEPLOYMENT"
BEST_MODEL=best_model
```

## 8. Conclusion

In the culmination of our project, we arrive at a significant juncture where we reflect upon the entire journey of deploying a machine learning model to address the pressing issue of student dropout rates in educational institutions. This section serves as a comprehensive conclusion, encapsulating the key takeaways, achievements, and the broader impact of our project.

Our journey commenced with a clear objective: to develop a predictive model that empowers educational institutions to proactively identify students at risk of dropping out. Student dropout rates have long been a substantial concern, with far-reaching implications, from the underutilization of educational resources to the potential disruption of students' academic and career paths. To address this challenge, we embarked on a data-driven quest that leverages machine learning and cloud computing.

## 9. Documentation

In this section, we provide a thorough documentation of our machine learning model deployment project, delving into the critical aspects that have steered our project from inception to deployment.

## 9.1 Project Objectives and Phases

Our project's core objective is to tackle the issue of student attrition in educational institutions through the strategic application of machine learning. We have meticulously structured our project into five distinct phases, each playing a pivotal role in the overall success of our endeavor:

- Phase 1: Setting Up IBM Cloud and Watson Machine Learning: This initial phase involves the essential task of authenticating and setting up an IBM Cloud account, which serves as the foundation for our entire project. We delve into the intricacies of navigating the IBM Watson Machine Learning catalog, selecting third-party services, and establishing deployment spaces for model saving and deployment.
- Phase 2: Deploying with Python: In this phase, we transition to the practical application of deploying our model with Python. We detail the process of setting up API keys and credentials for various regions, creating a Python machine learning client, and listing deployment spaces. Additionally, we offer insights into saving and deploying the model, specifying crucial parameters for successful deployment.
- Phase 3: Model Deployment with Metadata: Our journey then proceeds to the critical phase of model deployment, where we emphasize the importance of model metadata. We guide you through the process of saving the model in the Watson Machine Learning repository, meticulously specifying model metadata using a dictionary, and passing essential keyword parameters. The documentation here also highlights the significance of model ID for automation purposes.
- Phase 4: Deploying a Model: The next phase focuses on the mechanics of deploying a model saved in a deployment space. We explore the process of obtaining the model's UID, specifying deployment underprops, and using the command `wml_client.deployment.create`` for deployment. The ``artifact_uid`` keyword parameter is discussed in detail, and the documentation highlights the crucial role of deployment properties in quantum format.
- Phase 5: Scoring the Deployed Model: The final phase, Phase 5, encapsulates the process of scoring the deployed model. We offer insights into scoring the model, using the deployment UID, and sending data from a pandas data frame. The `wml_client.deployments.get_uid`` and payload are explained in context, providing a clear understanding of how to make API requests for real-time predictions.

## 9.2 Predictive Use Case and Dataset Selection

Our project centers around a predictive use case dedicated to identifying students at risk of dropping out. To empower this use case effectively, we have meticulously selected a dataset that offers a comprehensive perspective on students enrolled in diverse undergraduate programs. This dataset encompasses demographic details, socioeconomic factors, academic performance metrics, and more, making it a robust foundation for our predictive model.



### **9.3 Model Training and Deployment Process**

We shine a spotlight on the critical phases of model training and deployment that underpin our project's success. Model training encompasses data cleaning, feature engineering, data splitting, model selection, and the pivotal task of training the model to recognize patterns and make accurate predictions. The documentation underscores the importance of the IBM Cloud Watson Studio's deployment capabilities, allowing for scalability, accessibility, and real-time predictions.

### **9.4 Accessing and Utilizing the Deployed Model**

Once our model is deployed, we provide a comprehensive guide on how it can be accessed and effectively utilized for real-time predictions. This section explores the integration of APIs, the creation of a user-friendly dashboard for educators and administrators, and the implementation of automated alerts for high-risk students. The documentation emphasizes the model's role in facilitating data-driven decisions and personalized interventions, ultimately fostering student success. Additionally, we highlight the importance of meticulous documentation for knowledge transfer and future enhancements.

## **10. Submission**

In this section, we outline the key elements of our project submission, ensuring that all essential components are readily accessible for evaluation and utilization.

### **10.1 GitHub Repository Link**

To facilitate easy access to our project code and associated files, we have established a GitHub repository. The repository serves as a central hub for all project-related resources, making it convenient for reviewers, collaborators, and users to explore our work. You can find the GitHub repository at the following link: [\[GitHub Repository Link- Click here\]](#)

### **10.2 Instructions for Deployment and Usage**

Deploying and utilizing our machine learning model as a web service is a streamlined process that we aim to simplify further. The following instructions are designed to guide users through the deployment and usage of the model, enabling them to harness its predictive capabilities effectively:

1. **Register on IBM Cloud:** To initiate the deployment process, users are required to create an IBM Cloud account if they don't have one. Registration is a straightforward process and lays the foundation for deploying our model.

2. **Access Watson Studio:** Upon logging into your IBM Cloud account, navigate to the IBM Watson Studio service in the Dallas region. This is where the core of our machine learning projects is managed.
3. **Create a Project:** Within Watson Studio, create a new project to efficiently organize your work. This project will serve as the hub for your dataset, model, and deployment spaces.
4. **Upload the Dataset:** Our model relies on specific datasets for training and deployment. Users are encouraged to upload the provided dataset to IBM Cloud Object Storage, which will be used for these crucial phases.
5. **Develop a Jupyter Notebook:** To train the machine learning model, a Jupyter Notebook is an invaluable tool. Within your project, you can create a Jupyter Notebook for data preprocessing, model training, and deployment.
6. **Train the Model:** Using Python and libraries like scikit-learn, proceed to train the machine learning model. The Notebook provides a structured environment for this process.
7. **Store the Trained Model:** Once the model is trained, store it in a deployment space within Watson Studio. This step is essential for making the model accessible via an API endpoint.
8. **Deploy the Model:** Deploying the model as an online service with an API endpoint is a pivotal step. The model becomes accessible for real-time predictions.
9. **Use the Python SDK:** To connect to the scoring endpoint, send input data, and receive predictions, it is recommended to use the Python SDK. This facilitates the interaction with the deployed model.
10. **Integrate Flask:** Consider integrating the Flask web framework to create a user interface. This interface will display model predictions and offer a user-friendly experience for accessing the predictive capabilities of the model.

### **10.3 Example API Requests for Predictions**

To help users kickstart their journey with the deployed model, we provide example API requests for making predictions. These requests serve as practical illustrations of how to interact with the model and obtain real-time predictions. By referring to these examples, users can gain a clear understanding of how to structure requests and leverage the model's capabilities to address their specific use cases.

## AN EXAMPLE API REQUEST IS GIVEN BELOW

```
import requests
import json

# Define the API endpoint for the deployed model
api_endpoint = "https://your-model-endpoint-url-here"

# Define the input data for prediction (You need to structure this based on your dataset columns)
input_data = {
    "Marital Status": "Single",
    "Application Mode": "Online",
    "Application Order": 2,
    "Course": "Computer Science",
    "Daytime/Evening Attendance": "Day",
    "Previous Qualification": "High School Diploma",
    "Nationality": "US",
    "Mother's Qualification": "Bachelor's Degree",
    "Father's Qualification": "Master's Degree",
    "Mother's Occupation": "Engineer",
    "Father's Occupation": "Doctor",
    "Displaced": "No",
    "Educational Special Needs": "No",
    "Debtor": "No",
    "Tuition Fees Up to Date": "Yes",
    "Gender": "Male",
    "Scholarship Holder": "Yes",
    "Age at Enrollment": 20,
    "International": "No",
    "Curricular Units 1st Sem (Credited)": 8,
    "Curricular Units 1st Sem (Enrolled)": 8,
    "Curricular Units 1st Sem (Evaluations)": 7,
    "Curricular Units 1st Sem (Approved)": 6
}

# Send the API request
response = requests.post(api_endpoint, json=input_data)

# Parse the prediction result
result = json.loads(response.text)

# Display the prediction
print("Predicted Dropout Probability:", result["prediction"])
```

The possible output of the API request to the deployed machine learning model for predicting student dropout can vary depending on the specific design and implementation of the model. However, in a typical scenario, the output may consist of a JSON response that provides information about the model's prediction.

Here's an explanation of what the possible output might look like:

```
json
{
  "prediction": 0.72
}
```

1. `"prediction"`: This key in the JSON response typically represents the prediction generated by the machine learning model. In the context of a student dropout prediction model, the value associated with this key can indicate the estimated probability or likelihood of a student dropping out. In the example above, the model predicts a 72% probability of dropout.

The output value, which is the predicted probability of student dropout, can be interpreted as follows:

- If the predicted probability is close to 0 (e.g., 0.05), it suggests a low likelihood of dropout, indicating that the model believes the student is likely to continue their education.
- If the predicted probability is close to 1 (e.g., 0.95), it suggests a high likelihood of dropout, indicating that the model believes the student is at risk of dropping out.
- If the predicted probability is around 0.5 (e.g., 0.50), it suggests an uncertain or neutral prediction, where the model cannot strongly classify the student as a dropout or non-dropout.

This output allows users of the API to make informed decisions or take timely actions based on the model's predictions regarding student dropout, such as providing additional support or interventions for students at higher risk.

It's important to note that the actual format of the output may depend on the model's design and the specific requirements of your machine learning deployment. Therefore, the structure of the output may vary.

### **Innovative techniques employed:**

The innovative techniques used in this project include:

1. Machine Learning and Predictive Analytics: Leveraging machine learning for student dropout prediction.
2. IBM Cloud Watson Studio: Using a cloud-based platform for model development and deployment.
3. Comprehensive Data Integration: Collecting and integrating diverse student data sources.
4. Predictive Model Selection: Choosing the best machine learning algorithm for accurate predictions.
5. Model Deployment as Web Services: Hosting predictive models in the cloud for scalability.

6. Integration with Educational Systems: Creating APIs and a user-friendly dashboard.

7. Automated Alert Mechanisms: Real-time alerts for high-risk students.

8. Documentation and Knowledge Transfer: Maintaining detailed documentation for future use.

### **Conclusion:**

In conclusion, this project represents an innovative approach to addressing the critical issue of high student dropout rates in educational institutions. Leveraging the power of machine learning and IBM Cloud Watson Studio, we have developed a predictive model that empowers educators and administrators to make data-driven decisions and take proactive steps to improve student retention rates. The project's comprehensive documentation, step-by-step deployment process, and integration into educational systems provide a practical solution for enhancing academic success and fostering an inclusive and supportive learning environment. The commitment to technology-driven education remains unwavering, and the future holds the promise of more achievable student success through these data-driven insights.