# Customer Churn Predictions Classifier App Deployment Process

1. **Introduction**

Deploying machine learning models into production environments is crucial in leveraging their predictive capabilities. This report focuses on the deployment process for a Customer Churn Predictions Classifier trained using PyTorch. The deployment process involves saving the trained model, preparing the data preprocessing steps, writing the application script, creating an interface for user interaction, managing dependencies, and setting up a virtual environment.

1. **Model Deployment Steps**
2. Saving the Trained Model: The first step in the deployment process is to save the trained Customer Churn Predictions Classifier model. This is achieved by executing the following code in Colab: “torch.save(churn\_predictor.state\_dict(), '/content/drive/My Drive/deep\_learning/churn\_predictor\_model.pth')”. This code saves the model's state dictionary to a specified file path.
3. Preprocessing Data: Alongside the trained model, preprocessing steps are crucial for preparing input data for predictions. This involves saving a preprocessor object that handles tasks such as OneHotEncoding for categorical data and standardizing numerical variables using StandardScaler. The preprocessor object is saved using the following: with open('/content/drive/My Drive/deep\_learning/preprocessor.pkl', 'wb') as f: pickle.dump(preprocessor, f)”. This code serializes the preprocessor object using Python's pickle module and saves it to a specified file path.
4. Application Script: The flask application script is written Using Visual Studio Code (VS Code) to integrate the trained model and preprocessing steps. The script is essential for loading the model, preprocessing input data, and making predictions. It encapsulates the functionality of the deployed application.
5. User Interface: HTML files are created to interact with the Flask application to provide an intuitive interface for users. These HTML files contain upload fields, buttons, and result displays. The interface facilitates user interaction with the deployed model, allowing them to upload data and receive predictions.
6. Managing Dependencies: The application relies on specific dependencies to function correctly. To ensure seamless deployment, a list of dependencies is compiled and saved in a requirements.txt file. This file is generated using the pipreqs tool, which automatically detects and lists dependencies based on the Python script's imports.
7. Setting Up Virtual Environment: A virtual environment is created to isolate the application's dependencies from other Python projects. This ensures the application runs with its specific set of dependencies without conflicts. The virtual environment is created using the following commands: “python -m venv envname". Then activate the environment using "envname\Scripts\activate”. Once activated, the virtual environment allows for the installation of dependencies using the requirements.txt file: “pip install -r requirements.txt”

Conclusion:

Deploying machine learning models involves a series of steps, from saving the trained model to setting up a user interface and managing dependencies. This report has provided a detailed overview of the deployment process for a Customer Churn Predictions Classifier, highlighting the importance of each step in ensuring the successful operation of the deployed application. By following these steps and best practices, developers can effectively deploy machine learning models into production environments, enabling their utilization for real-world applications.