Diabetes mellitus prediction using IBM Auto AI service

Project Description:

Diabetes mellitus is a chronic disease characterized by hyperglycemia. It may cause many complications. According to the growing morbidity in recent years, in 2040, the world's diabetic patients will reach 642 million, which means that one of the ten adults in the future is suffering from diabetes.

In this project, you need to build a machine learning model that can efficiently discover the rules to predict diabetes mellitus of patients based on the given parameter about their health. The model needs to be deployed in the IBM cloud to get scoring endpoint which can be used as API in web app building. . The model prediction needs to be showcased on User Interface.

Services Used:

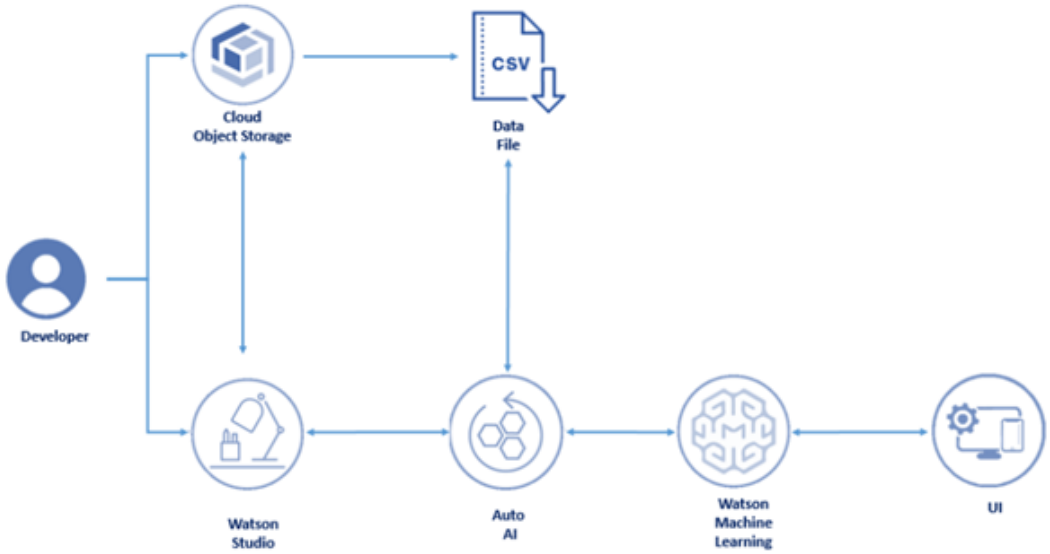
IBM Watson Studio

IBM Watson Machine Learning

Node-RED

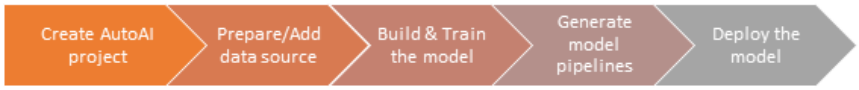
IBM Cloud Object Storage

Architecture:



Process flow:

The following figure depicts the process flow of the project Diabetes mellitus prediction.



Create AutoAI Project:

In IBM Cloud, Watson Studio is used to build the model in the AutoAI. The steps below creates the project and associate addition service for Diabetes mellitus prediction

- Create new project created in Watson Studio
- Add Cloud object storage.
- In the Assets page of the project, chose AUTOAI EXPERIMENT
- Associated Machine learning service instance from Dallas region

Prepare/Add Data source:

The sample data used in this project is Diabetes Prediction which provides the list of features that predicts a person is Diabetic or not. The data source link shown below has 9 columns used for prediction.

https://www.kaggle.com/akhilalexander/diabeticprediction

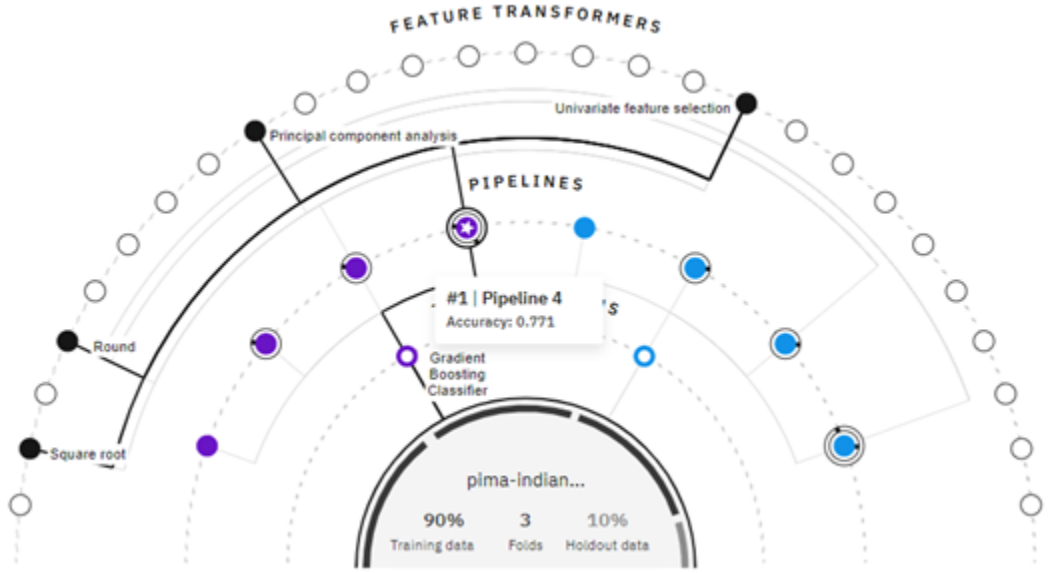
Build and Train the Model:

The data source uploaded and specified the Prediction Column as class in AutoAI. AutoAI analyzes the data and determines that the class column contains 0/1 information, making this data suitable for a binary classification model. The optimized metric for the binary classification is Accuracy.

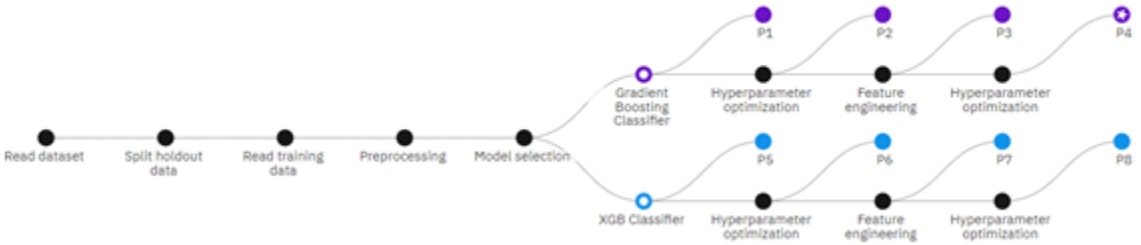
Based on the data source following prediction are provided

- Training data split – 90%
- Initial model tuning iterations: 25
- Feature engineering iterations:60
- Final model tuning iterations: 50

As the model trains, infographic below shows the process of building the pipelines.



Progress Map:



Generate model pipeline

Once the pipeline creation is complete, the ranked pipelines in a leaderboard can be viewed with comparison.

The figure below shows the pipeline leaderboard with comparison of 8 pipelines along with the algorithm,accuracy, average prediction,F1,Log loss, Precision, Recall, ROC/AUC.

Pipeline leaderboard

Rank	Name	Algorithm	Accuracy (Optimized)	Average precisi...	F_1	Log loss	Precision	Recall	ROC AUC
★ 1	Pipeline 4	Gradient Boosting Classifier	0.771	0.708	0.630	0.493	0.724	0.564	0.821
2	Pipeline 3	Gradient Boosting Classifier	0.771	0.725	0.621	0.493	0.734	0.547	0.820
3	Pipeline 2	Gradient Boosting Classifier	0.763	0.699	0.614	0.497	0.712	0.543	0.816
4	Pipeline 7	XGB Classifier	0.761	0.661	0.653	0.567	0.664	0.643	0.802
5	Pipeline 8	XGB Classifier	0.761	0.661	0.653	0.567	0.664	0.643	0.802
6	Pipeline 1	Gradient Boosting Classifier	0.753	0.683	0.621	0.523	0.669	0.581	0.817
7	Pipeline 6	XGB Classifier	0.753	0.652	0.639	0.573	0.649	0.630	0.799
8	Pipeline 5	XGB Classifier	0.751	0.678	0.635	0.528	0.649	0.622	0.808

Gradient Boosting Classifier from the action menu for the pipeline with a rank of 1 saved as model. This saves the pipeline as a Machine Learning asset in the project.

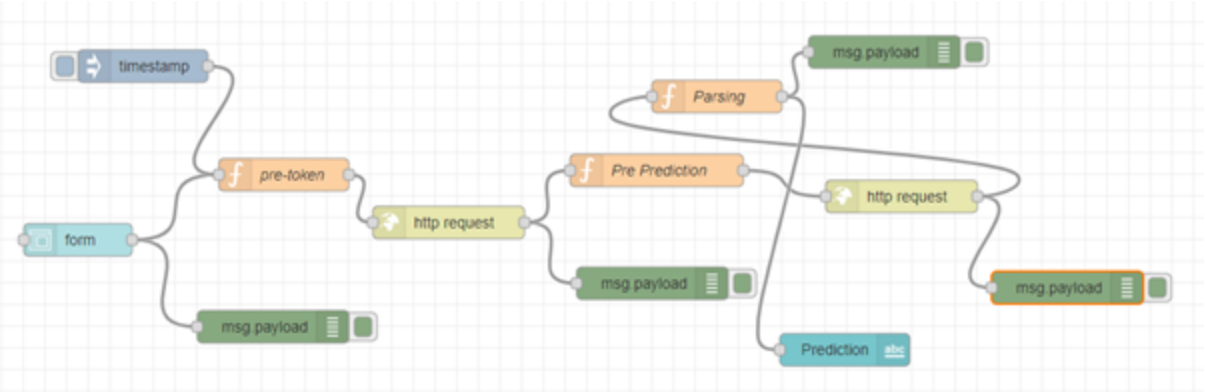
Deploy the trained model

- Initially deployment space is created and the model is deployed.
- Machine learning instance associated with the space.
- The model deployed in the Online Mode and tested.
- Endpoint and API key are generated and saved to integrate the model with the application created in Node Red.

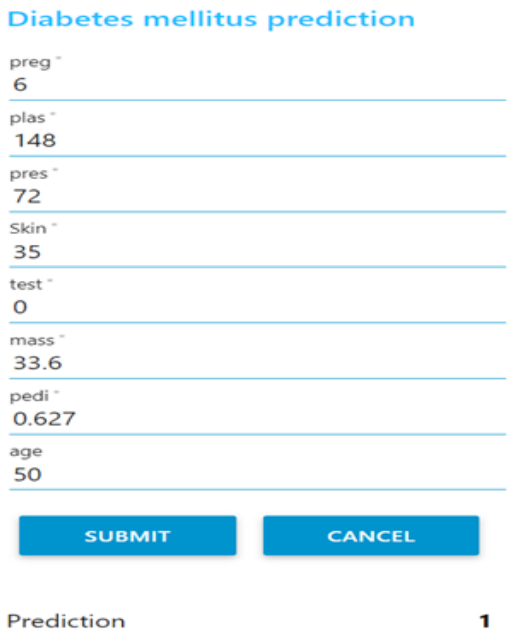
Create and Integrate Node Red to Model:

In this activity, node-red is integrated with the model by calling the API key and scoring end point of the model. The json file for ML Auto AI is uploaded. The following updates are made

- Form updated for the data source elements
- Function Node updated for the input data fields and values.



Finally the model is deployed in the Node Red and produced the following the prediction.



The prediction shows that the person having Diabetic. If the prediction is 0, then the person does not have Diabetic.

Conclusion:

Binary classification with Accuracy as optimized metric are used to train the model for Diabetes mellitus prediction. Upon training on 8 pipelines, Gradient Boosting classifier(pipeline 4) produced accuracy of 0.771 with the Build time 00:00:25, which is saved as model and deployed. The model is further tested and integrated with Node Red service where the result exactly predict the class 0/1 represents No Diabetic/Diabetic.