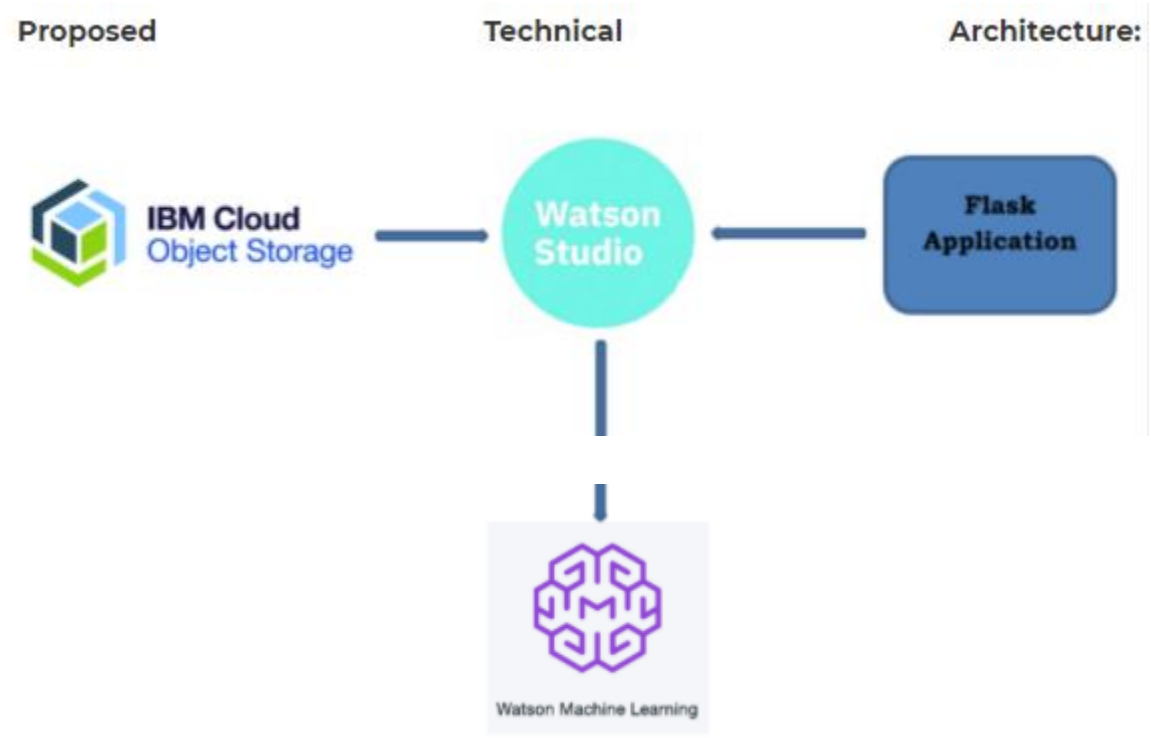
Diabetics Prediction System Based On Life Style

INTRODUCTION

In this, we need to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

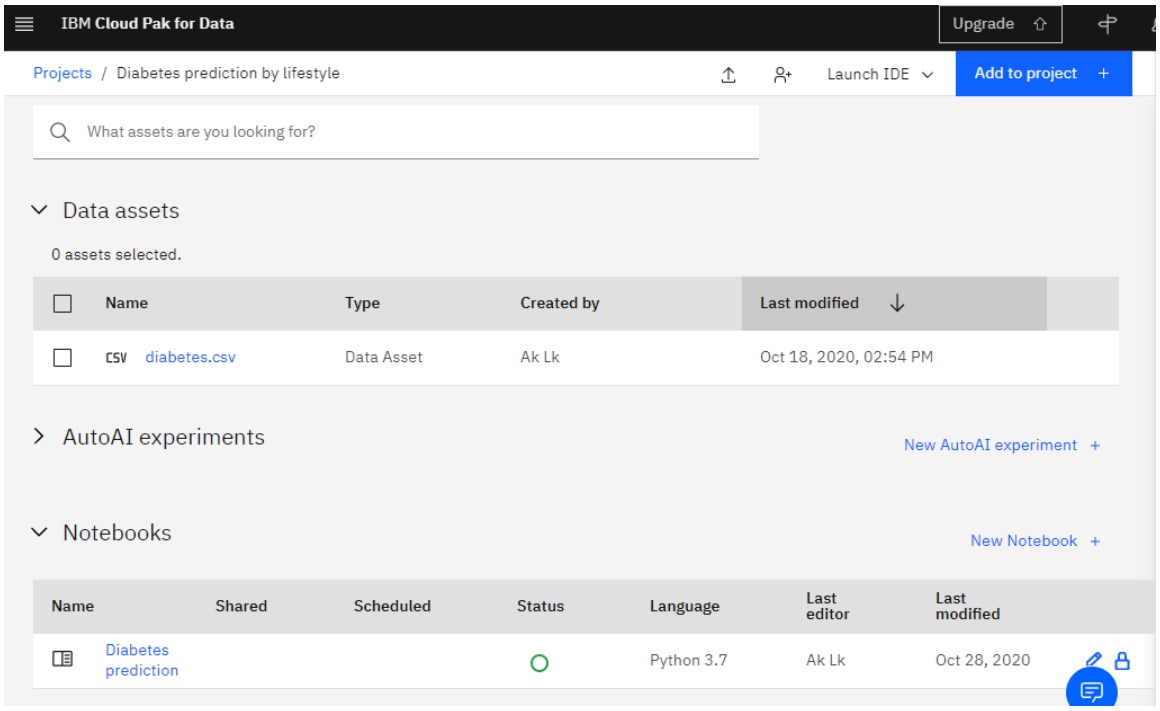
The datasets consist of several medical predictor variables and one target variable, Diabetes. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

BLOCK DIAGRAM



METHODOLOGY

1. Create a project in IBM Watson studio and Add Notebook and diabetes.csv file



2. Import the necessary libraries and dataset.
3. Visualize the data. Check for any null values.
4. Even though there are no null values there are 0's in some predictor variables which has to be replaced by the mean of values.
5. Once the feature scaling is done, split it into train and test set.
6. Train the model using classification algorithms.

	Logistic Regression	KNN	Naive Bayes	Decision Tree Classifier	Random Forest Classifier	Support Vector Machine
accuracy	0.707792	0.75974	0.694805	0.62987	0.714286	0.701299

7. Identify the best model using the accuracy score.
8. Obtain the Watson machine learning credentials

```
wml_credentials = {
                      "apikey":"IQ3jJ4f7rD-vhat0JKKuoQ38J85Opxi2MRRQ8Z0_xE99",
                      "url": "https://us-south.ml.cloud.ibm.com"
}
from ibm_watson_machine_learning import APIClient
client = APIClient(wml_credentials)
```

```
bestmodel = pipeline
```

```
MODEL_NAME = "Diabetes Prediction Model"
Deployment_name ="Model Deployment"
best_model = "Knn model"
```

```
software_spec_id = client.software_specifications.get_id_by_name("scikit-learn_0.22-py3.6")
software_spec_id
```

```
'154010fa-5b3b-4ac1-82af-4d5ee5abbc85'
```

```
model_props = {
    client.repository.ModelMetaNames.NAME : "{}".format(MODEL_NAME),
    client.repository.ModelMetaNames.TYPE : "scikit-learn_0.22",
     client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_id
}
```

9. Create a pickle file for deployment.

```
import pickle
pickle.dump( pipeline, open( "model.pkl", 'wb') )
```

```
!tar -zcvf model.tar.gz model.pkl
```

```
model.pkl
```

```
client.set.default_space('519f4181-a557-472f-825c-862a9a9fb67f')
```

```
'SUCCESS'
```

```
published_model_details = client.repository.store_model(model = "model.tar.gz", meta_props = model_props, training_data = X_trai
n, training_target = y_train)
```

```
published_model_details
```

```
{'entity': {'software_spec': {'id': '154010fa-5b3b-4ac1-82af-4d5ee5abbc85',
   'name': 'scikit-learn_0.22-py3.6'},
  'type': 'scikit-learn_0.22'},
 'metadata': {'created_at': '2020-10-28T13:57:39.992Z',
  'id': '486d9509-b039-4f82-8e14-b950dcb0faf1',
  'modified_at': '2020-10-28T13:57:42.121Z',
  'name': 'Diabetes Prediction Model',
  'owner': 'IBMid-5500060365',
  'space_id': '519f4181-a557-472f-825c-862a9a9fb67f'}}
```

```
model_uid=client.repository.get_model_uid(published_model_details)
model_uid
```

```
'486d9509-b039-4f82-8e14-b950dcb0faf1'
```

```
meta_props = {
        client.deployments.ConfigurationMetaNames.NAME: "Model Deployment",
        client.deployments.ConfigurationMetaNames.ONLINE: {}
    }

created_deployment = client.deployments.create(artifact_uid=model_uid , meta_props=meta_props)
```

```
#######################################################################################

Synchronous deployment creation for uid: '486d9509-b039-4f82-8e14-b950dcb0faf1' started

#######################################################################################


initializing
ready


-----------------------------------------------------------------------------------------------
Successfully finished deployment creation, deployment_uid='357eab0c-64c9-4860-bc69-651fce092b82'
-----------------------------------------------------------------------------------------------
```

10.Deployment created

```
iam_token = client.wml_token
```

```
created_deployment
```


```
{'entity': {'asset': {'id': '486d9509-b039-4f82-8e14-b950dcb0faf1'},
  'custom': {},
  'deployed_asset_type': 'model',
  'hardware_spec': {'id': 'Not_Applicable', 'name': 'S', 'num_nodes': 1},
  'name': 'Model Deployment',
  'online': {},
  'space_id': '519f4181-a557-472f-825c-862a9a9fb67f',
  'status': {'message': {'text': 'Scikit-learn 0.22/XGBoost 0.90 framework with Python3.6 for Watson Machine Learning is deprec
ated and will be removed on January 20, 2021. Use Scikit-learn 0.23/XGBoost 0.90 with default_py3.7 instead. For details, see h
ttps://dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/pm_service_supported_frameworks.html',
     'level': 'warning'},
    'online_url': {'url': 'https://us-south.ml.cloud.ibm.com/ml/v4/deployments/357eab0c-64c9-4860-bc69-651fce092b82/prediction
s'},
    'state': 'ready'}},
 'metadata': {'created_at': '2020-10-28T13:57:49.578Z',
  'id': '357eab0c-64c9-4860-bc69-651fce092b82',
  'modified_at': '2020-10-28T13:57:49.578Z',
  'name': 'Model Deployment',
  'owner': 'IBMid-5500060365',
  'space_id': '519f4181-a557-472f-825c-862a9a9fb67f'}}
```


11.Reference deploy the model

```
deployment_href = client.deployments.get_href(created_deployment)
deployment_href
```

```
'/ml/v4/deployments/357eab0c-64c9-4860-bc69-651fce092b82'
```

```
deployment_id = client.deployments.get_uid(created_deployment)
deployment_id
```

```
'357eab0c-64c9-4860-bc69-651fce092b82'
```

12.Score the deployed model

```
import urllib3, requests, json
```

```
#To score the deployed model construct the model payload, following the schema of the model.
fields = ['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedigreeFunction','Age']
values = [0,137,40,35,168,43.1,2.288,33]
scoring_payload = {client.deployments.ScoringMetaNames.INPUT_DATA: [{"fields": fields, "values": [values]}]}
print(json.dumps(scoring_payload, indent=2))
```

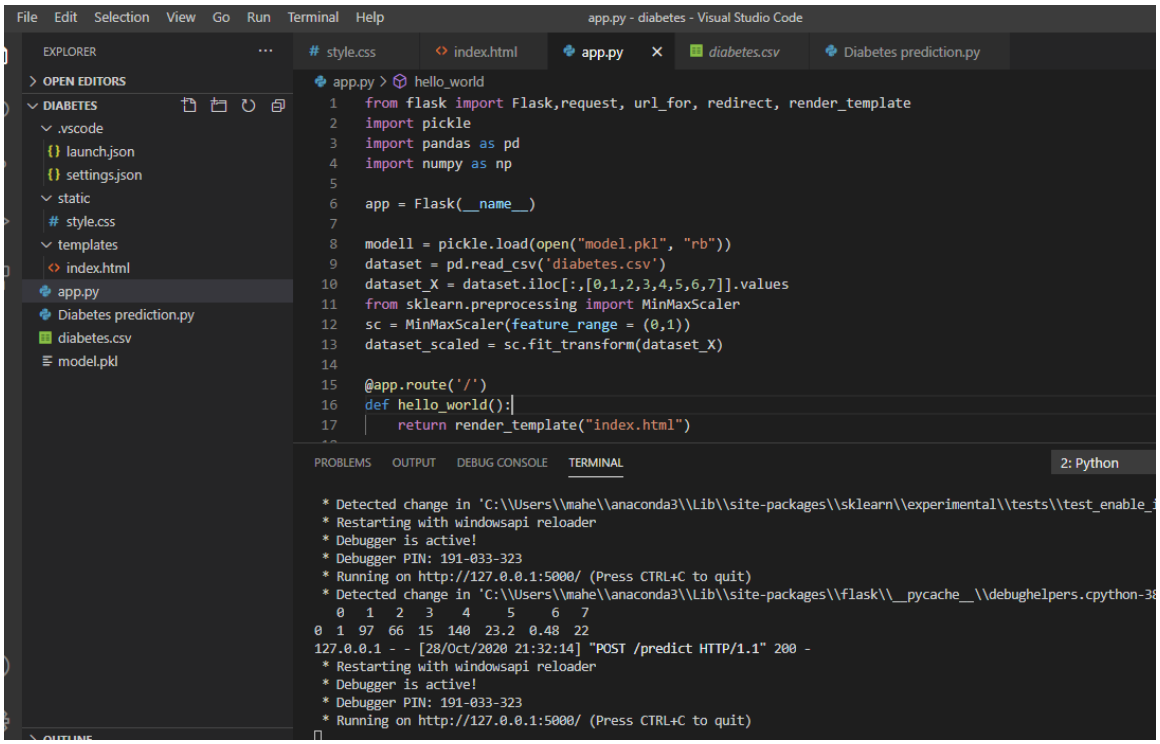
```
{
  "input_data": [
    {
      "fields": [
        "Pregnancies",
        "Glucose",
        "BloodPressure",
        "SkinThickness",
        "Insulin",
        "BMI",
        "DiabetesPedigreeFunction",
        "Age"
      ],
      "values": [
        [
          0,
          137,
          40,
          35,
          168,
          43.1,
          2.288,
          33
        ]
      ]
```

13. Run the score function

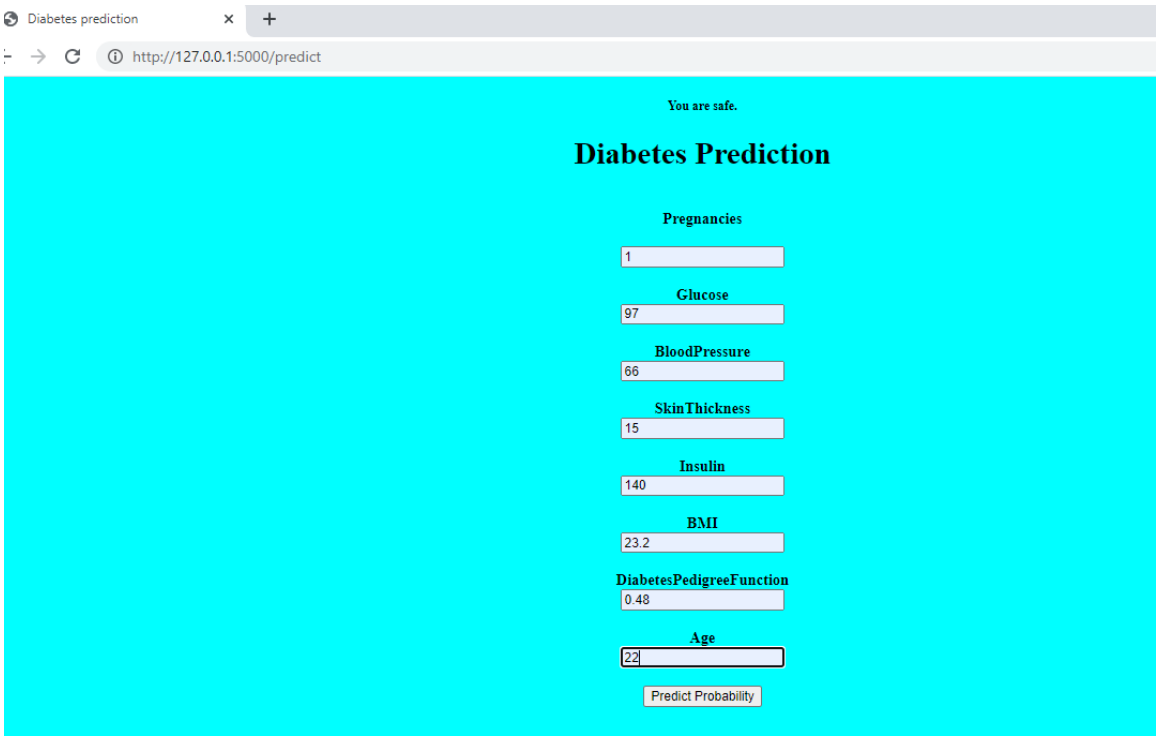
```
predictions = client.deployments.score(deployment_id, scoring_payload) #Run the score function to generate the prediction.
predictions
```

```
{'predictions': [{'fields': ['prediction', 'probability'],
   'values': [[1.0, [0.16666666666666666, 0.8333333333333334]]]}]}
```

14. Build Flask Application using the pickle file.



15. Generate an html file and enter the values.



For the given values, it shows that the person is safe.

CONCLUSIONS AND FUTURE SCOPE

Among the machine learning models K Nearest neighbors was chosen as the best one and deployed. Hyper parameter tuning can be done and then checked for a better machine learning model so that the prediction is will be more accurate than the existing one which has an accuracy score of 75%.