

A short horizontal bar with a teal-to-orange gradient.

DSCI 551 Midterm Progress Report

Early Stage Diabetes Risk Prediction





Project Title

Early Stage Diabetes Risk Prediction

Topic

Healthcare

Dataset

[https://archive.ics.uci.edu/ml/datasets/Early+stage+diabetes+risk+prediction+dataset.](https://archive.ics.uci.edu/ml/datasets/Early+stage+diabetes+risk+prediction+dataset)



Motivation

1

Diabetes is a chronic (long lasting) health condition that affects how your body turns food into energy. As there isn't a cure yet for diabetes, diagnosing it at an early stage is essential to be clinically treated.

2

Due to the presence of a relatively long asymptomatic phase, early detection is a difficult task and hence we feel that if data science is used for this purpose it would be of immense social value.

3

Since diabetes is linked to the lifestyle of a person, based on a simple questionnaire we can predict whether a person is diabetic or not. This questionnaire includes critical questions like what is the age and gender of the person, whether the person is experiencing excessive urination or not, or is he or she experiencing excessive thirst or not and so on which help us predict whether the person is diabetic or not.



Project Overview

Data: For this project we will need a labelled dataset which is open sourced. While looking to satisfy these requirements we found a dataset where data was collected using direct questionnaires and diagnosis results from the patients in the Sylhet Diabetes Hospital in Sylhet, Bangladesh. The dataset contains the signs and symptoms of newly diabetic or would be a diabetic patient.

Machine Learning Model: We plan on using this publicly available dataset which has a list of carefully chosen 16 questions along with the corresponding diagnosis results to fit a carefully chosen and designed supervised machine learning classification algorithm like the random forest, naive bayes, neural networks and logistic regression .

Web App (Browser): To complement this we will also be developing a web based User Interface (UI) where the user will be able to answer those 16 questions himself and see what our model predicts corresponding to his inputs (whether he is diabetic or not). Apart from this, the UI will also have insightful data visualizations where we explore the most common features associated with diabetic risk. We are planning to do this with the help of streamlit.

Progress in meeting the goals set out in the proposal



Project Definition and Planning

- Got a relatively clean dataset without missing values. (Using the link submitted in the project proposal)
- Explored mongoDB and firebase and chose firebase as it is better suitable for small-scale applications like ours.
- Explored flask and streamlit and found streamlit to be more suitable for our purpose as we did not want to build highly customized interfaces and wanted a basic interface. Additionally streamlit also provides a pythonic way of implementation which we are most comfortable with.

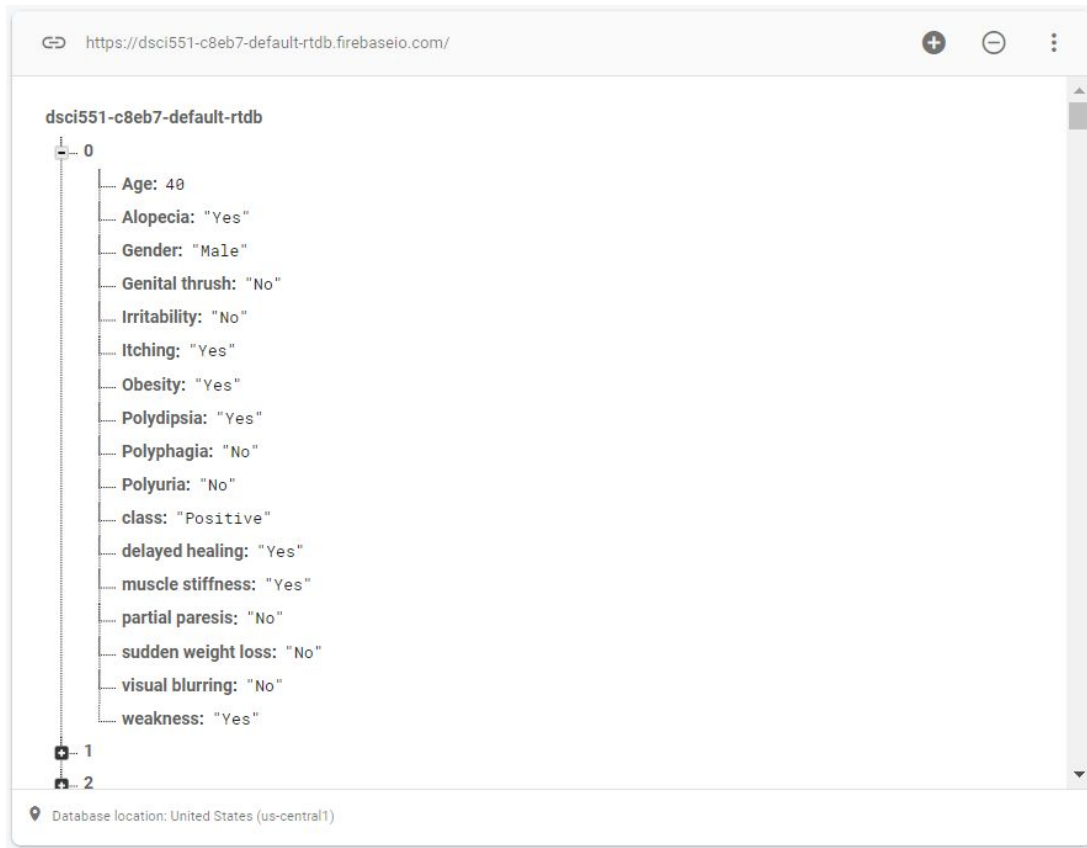
Project Execution (Cloud Database)

- Set up the Firebase successfully without any hassles
- Loaded the data into Firebase

Progress in meeting the goals set out in the proposal

Project Execution (Cloud Database)

- Snapshot:



Progress in meeting the goals set out in the proposal



Project Execution (ML Models)

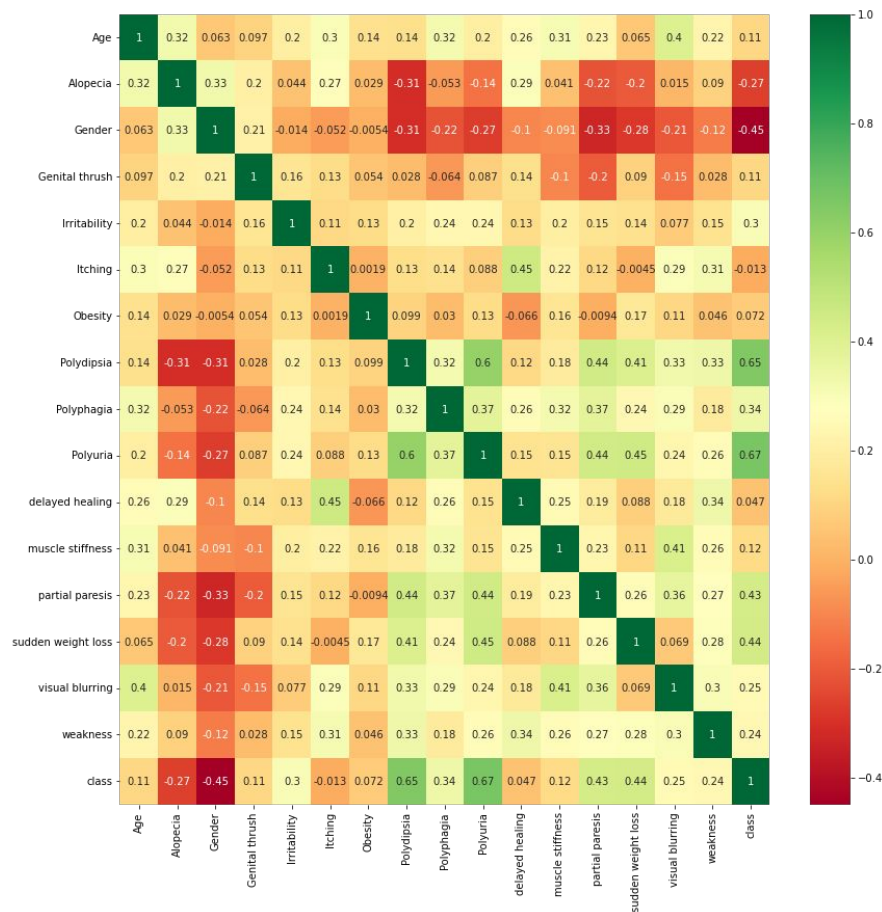
- Assigned values to categorical features and split the data into train and test samples to build and configure the model. We also checked for outliers in the dataset by plotting a boxplot and found out that there were not many outliers and we need not remove any samples.
- We plotted the correlation between all features and found out that there were not any features which were highly correlated and hence looking at the heatmap we could conclude that the features are more or less independent.
- Plotted the value counts for all categorical variables to check if any of them are imbalanced or not. (1:10 was the imbalance ratio we had in mind). Since all variables were balanced we proceeded to use them normally.
- To gauge the feature importance for our data we plotted the barplot and violin plot. In the bar plot we plotted the average value of each feature for a given value of class. If the feature is to be more important according to this plot then we expect to see stark differences in the averages for both class 0 and class 1. In the violin plot along with the box plot, we plotted the kernel density estimate. If the feature is to be more important according to this plot then we expect to see stark differences in the density distribution for each feature belonging to class 0 and class 1. We found out that features like polydipsia and polyuria are the best features followed by sudden weight loss and partial paresis.

Progress in meeting the goals set out in the proposal

Exploratory Data Analysis



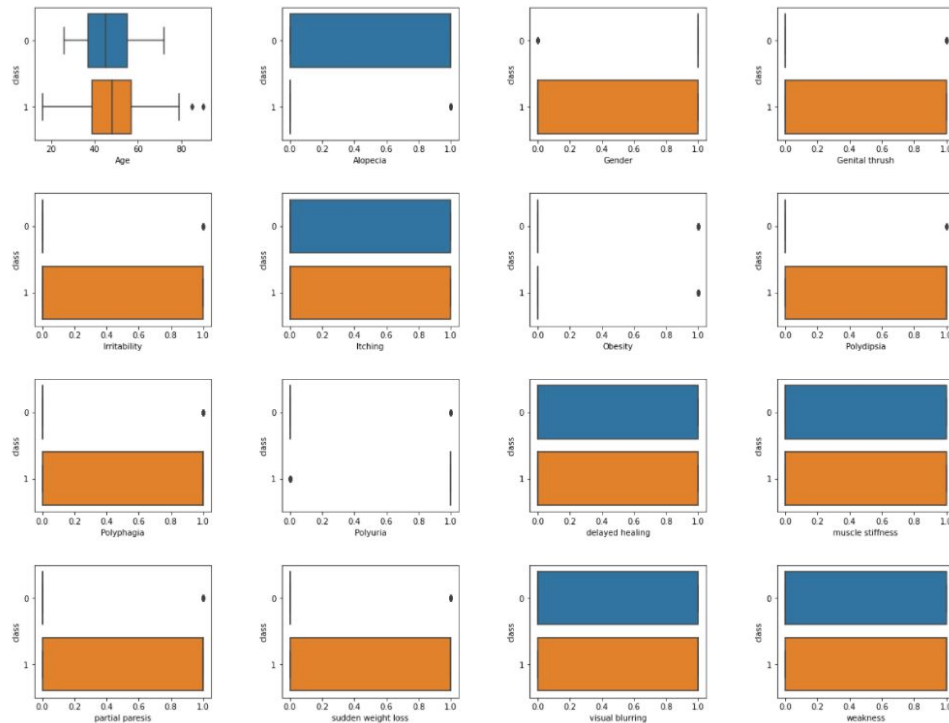
Plot for correlation between the features:



Progress in meeting the goals set out in the proposal

Exploratory Data Analysis

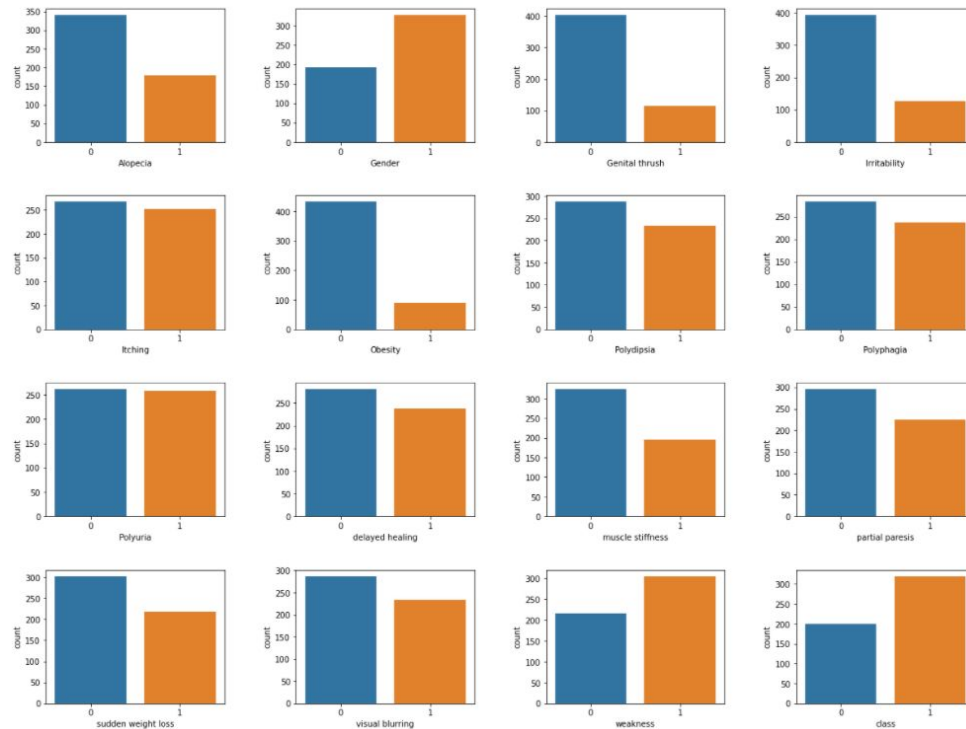
Box plot for all the features to check the distribution of data:



Progress in meeting the goals set out in the proposal

Exploratory Data Analysis

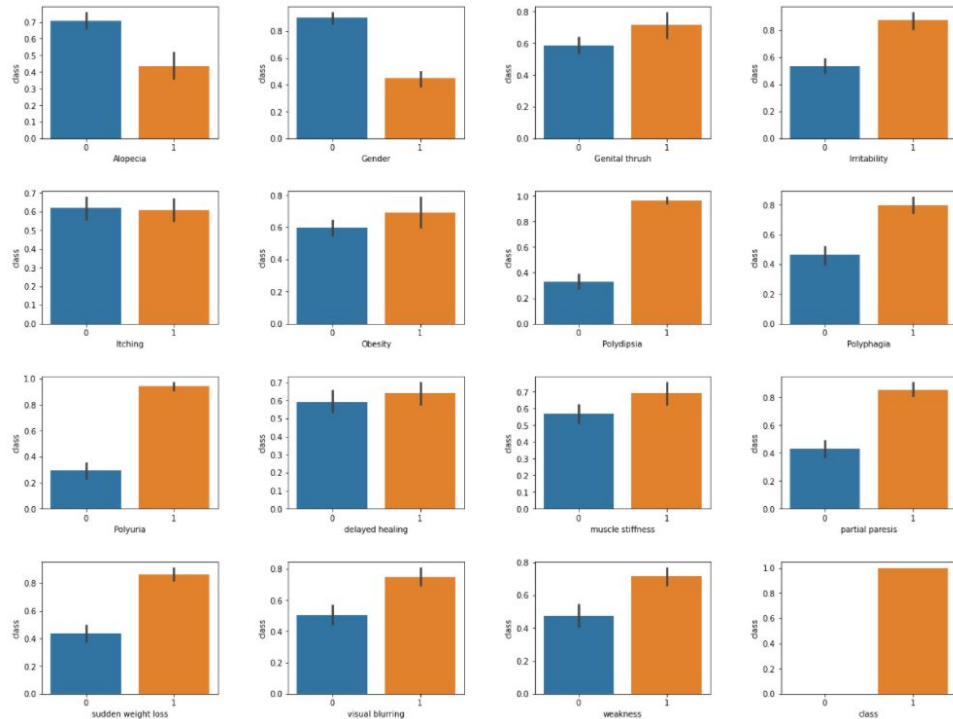
As almost all the features are categorical in nature, we plotted count plot for all the features:



Progress in meeting the goals set out in the proposal

Exploratory Data Analysis

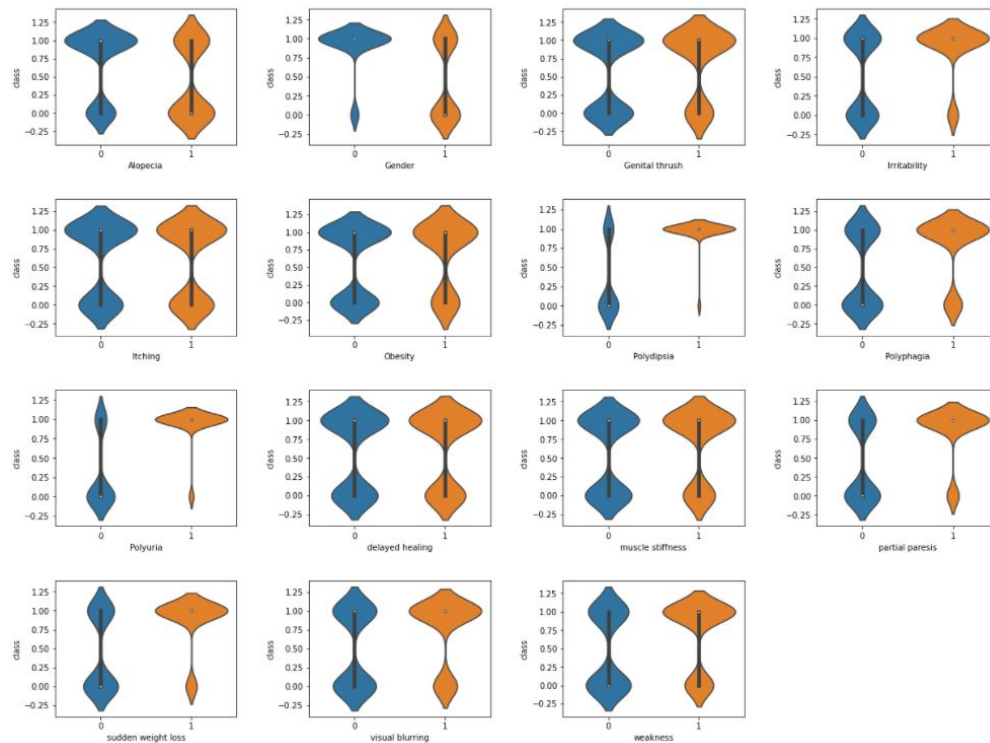
Bar plot for all the features to plot the mean values for each feature:



Progress in meeting the goals set out in the proposal

Exploratory Data Analysis

Violin plot for all the features to plot the combination of box plot and kernel density estimate:



Progress in meeting the goals set out in the proposal




- *Trained multiple machine learning models to understand which model fits the best to our data.*

The details of the models we tried are listed below.

- **Random Forest** - We felt that random forest being a tree based method might perform well on these dataset as we have a lot of categorical variables and hence a tree based based method would work well. For the model of our random forest we used 120 decision trees and fixed the maximum depth of each tree to 3.
- **Logistic regression** - Initially, we used the default scikit parameters for this logistic regression model ('l2' penalty with 'lbfgs' solver). This gave reasonably good results but we still saw scope for improvement and hence went on to try other models.

Progress in meeting the goals set out in the proposal

- 
- **Categorical Naives Bayes** - As our features were more or less independent, we thought of implementing Naive Bayes. Since all the features are categorical in our case, we have used the categorical naive bayes algorithm as it assumes that each feature has its own categorical distribution.
 - **Neural network** - To experiment with deep learning based tasks for this task, we built a simple network with the following architecture:

```
my_network(  
  (l1): Linear(in_features=16, out_features=128, bias=True)  
  (a): ReLU()  
  (l2): Linear(in_features=128, out_features=32, bias=True)  
  (l3): Linear(in_features=32, out_features=1, bias=True)  
  (a_final): Sigmoid()  
  (d): Dropout(p=0.3, inplace=False))
```

We used a high dropout probability of 0.3 to make sure that our network does not overfit the data. We used binary cross entropy loss along with the Adam optimizer with a learning rate of 0.005 to train the network. We trained the model for 20 epochs and used a batch size of 32 to pass our data to the model which was built using the well renowned deep learning framework PyTorch.

Progress in meeting the goals set out in the proposal



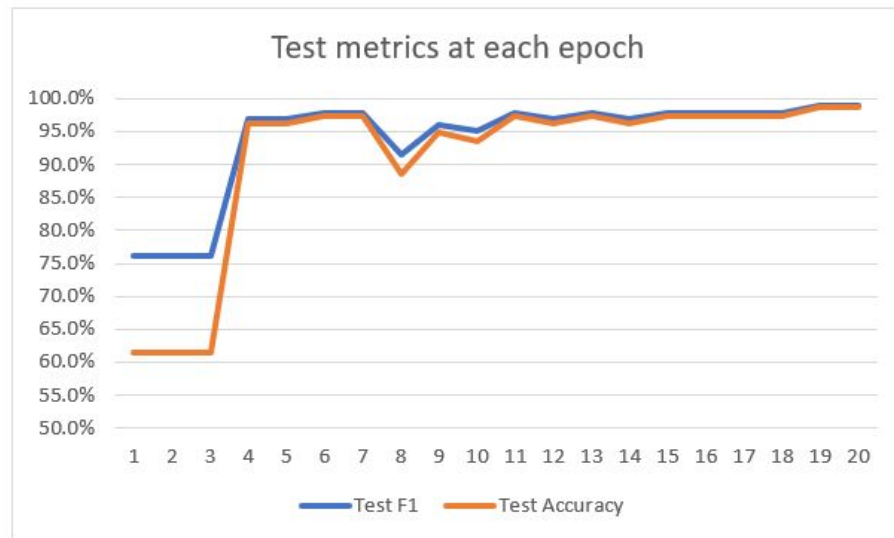
Model Comparison	F1 Score		Accuracy	
ML Models	Train	Test	Train	Test
Random Forest	0.932	0.979	0.916	0.974
Logistic Regression	0.943	0.969	0.930	0.962
Categorical Naives Bayes	0.901	0.979	0.885	0.974
Neural Network	0.927	0.989	0.914	0.987

Neural network outperforms other algorithms when it comes to test accuracy and F1 score

Progress in meeting the goals set out in the proposal

Final ML Model Chosen

We are moving ahead with the neural network based model because on the highest test accuracy and F1 score we were able to achieve.



Progress in meeting the goals set out in the proposal

Final ML Model Chosen

Snippets of the code:

```
class my_network(nn.Module):
    def __init__(self, input_size, dropout_required=True):
        super(my_network, self).__init__()
        self.input_size = input_size
        self.dropout_required = dropout_required
        self.l1 = nn.Linear(input_size, 128)
        self.a = nn.ReLU()
        self.l2 = nn.Linear(128, 32)
        self.l3 = nn.Linear(32, 1)
        self.a_final = nn.Sigmoid()
        if dropout_required:
            self.d = nn.Dropout(p=0.3)

    def forward(self, x):
        out = self.l1(x)
        out = self.a(out)
        if self.dropout_required:
            out = self.d(out)
        out = self.l2(out)
        out = self.a(out)
        if self.dropout_required:
            out = self.d(out)
        out = self.l3(out)
        out = self.a_final(out)
        return out
```

```
fls, accs = [], []
def train_net(data_batches, loss_function, optim):
    n_samples = len(data_batches.dataset)
    model_nn.train()
    for batch, (x, y) in enumerate(data_batches):
        x, y = x.to(device).float(), y.to(device).float()
        pred = model_nn(x)
        pred = pred.squeeze()
        loss = loss_function(pred, y)


        optim.zero_grad()
        loss.backward()
        optim.step()

        if batch % 5 == 0:
            loss, done = loss.item(), batch * len(x)
            print(f"Loss={loss} for progress {done}/{n_samples}")

def test_net(data_batches, loss_function):
    n_samples = len(data_batches.dataset)
    n_batches = len(data_batches)
    model_nn.eval()
    test_loss, correct = 0, 0
    tp, tn, fp, fn = 0, 0, 0, 0
    with torch.no_grad():
        for x, y in data_batches:
            x, y = x.to(device).float(), y.to(device).float()
            pred = model_nn(x)
            pred = pred.squeeze()
            test_loss += loss_function(pred, y)
            pred = (pred >= 0.5).int()
            pred, y = pred.cpu().detach().numpy(), y.cpu().detach().numpy()
            cm = confusion_matrix(y, pred)
            tn += cm[0][0]
            fn += cm[1][0]
            tp += cm[1][1]
            fp += cm[0][1]
            correct += ((pred == y).sum()).item()
    test_loss = test_loss / n_batches
    accuracy = correct / n_samples
    precision = tp / (tp + fp)
    recall = tp / (tp + fn)
    f1 = (2 * precision * recall) / (precision + recall)
    fls.append(f1)
    accs.append(accuracy)
    print(f"Test error is {test_loss} ; Accuracy is {accuracy} ; F1 score is {f1}")
```

Challenges Faced



- 
- A decorative horizontal bar with a teal segment on the left and an orange segment on the right.
- Understanding which model to pick for our use case was one of the main challenges we faced. Initially, we had also faced some unexpected convergence issues with our pytorch model even though we expected the best results from it. We solved this issue by increasing the number of epochs and changing the learning rate for our optimizer.
 - Initially we were also using an extremely deep network which led to problems like overfitting and slow prediction. To mitigate this we modified our architecture to make faster predictions and increased the dropout probability to make our model more robust.
 - Also, we experimented with the ordering of dropout and activation and empirically chose to have the dropout layer after the activation function.
 - Additionally since both of us had no prior experience with web development, we faced some troubles deciding the correct framework to use for building the app (we eventually chose streamlit)

Expectation on the on-time completion of the project



- We are on track with the project execution in terms of building a cloud database and finalizing our machine learning model.
- We are currently working on building the web application. We have explored different tools for building web apps and have decided to move ahead with Streamlit.
- We are estimating that building the web application will be done by the week of 4/18 and the project will be ready by the start of week 4/25.



Team

Praveen Iyer | MS Applied Data Science | Email ID - praveeni@usc.edu | USC ID - 5549510111

Tejas Sujit Bharambe | MS Applied Data Science | Email ID - tbharamb@usc.edu | USC ID - 2160849114

Project Plan, Milestones & Team Members' Responsibilities

[illegible]



Relevant Papers

- Likelihood Prediction of Diabetes at Early Stage Using Data Mining Techniques
 - [\[Web Link\]](#) - Computer Vision and Machine Intelligence in Medical Image Analysis. Springer, Singapore, 2020. 113-125
 - Authors and affiliations:
 - M. M. Faniqul IslamEmail
 - Rahatara Ferdousi
 - Sadikur Rahman
 - Humayra Yasmin Bushra
- https://link.springer.com/chapter/10.1007/978-981-16-5655-2_41



Thank you.

