AI BASED DIABETES PRIDICTION SYSTEM

TRANSFLOW:

Globally, diabetes affects 537 million people, making it the deadliest and the most common non-communicable disease.

Many factors can cause a person to get affected by diabetes, like excessive body weight, abnormal cholesterol level, family history, physical inactivity, bad food habit etc. Increased urination is one of the most common symptoms of this disease. People with diabetes for a long time can get several complications like heart disorder, kidney disease, nerve damage, diabetic retinopathy etc. But its risk can be reduced if it is predicted early.

In this paper, an automatic diabetes prediction system has been developed using a private dataset of female patients in Bangladesh and various machine learning techniques. The authors used the Pima Indian diabetes dataset and collected additional samples from 203 individuals from a local textile factory in Bangladesh.

Feature selection algorithm mutual information has been applied in this work. A semi-supervised model with extreme gradient boosting has been utilized to predict the insulin features of the private dataset.

SMOTE and ADASYN approaches have been employed to manage the class imbalance problem.

The authors used machine learning classification methods, that is, decision tree, SVM, Random Forest, Logistic Regression, KNN, and various ensemble techniques, to determine which algorithm produces the best prediction results. After training on and testing all the classification models, the proposed system provided the best result in the XGBoost classifier with the ADASYN approach with 81% accuracy, 0.81 F1 coefficient and AUC of 0.84.

Furthermore, the domain adaptation method has been implemented to demonstrate the versatility of the proposed system.

The explainable AI approach with LIME and SHAP frameworks is implemented to understand how the model predicts the final results. Finally, a website framework and an Android smartphone application have been developed to input various features and predict diabetes instantaneously.

The private dataset of female Bangladeshi patients and programming codes are available at the following link: https://github.com/tansin-nabil/Diabetes-Prediction-Using-Machine-Learning.

FLOW DIAGRAM:

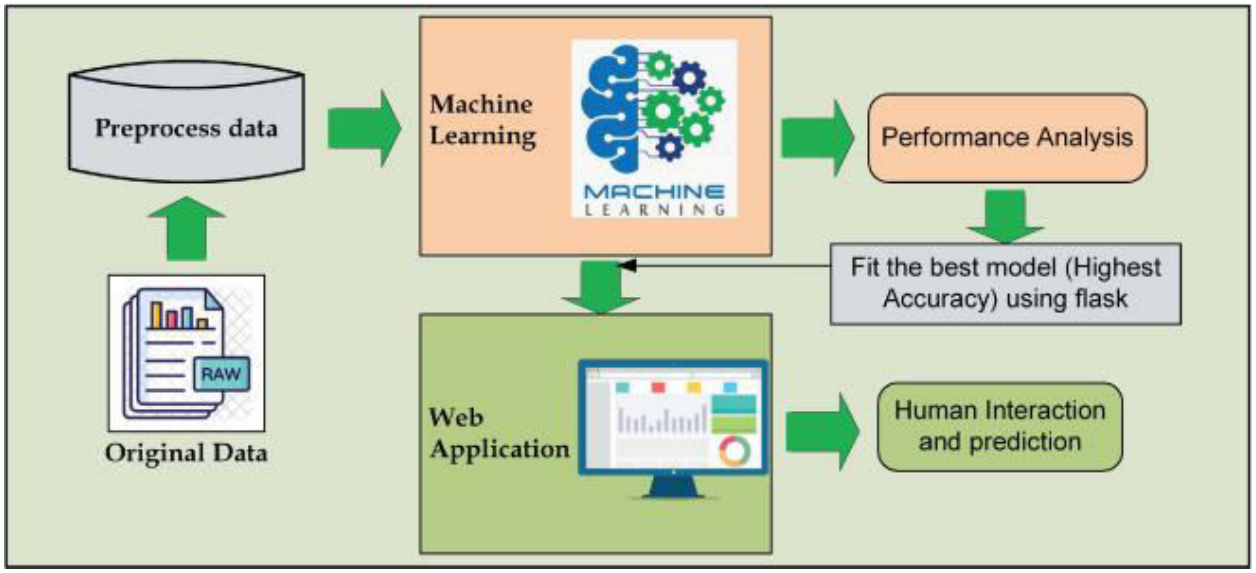
Make sure that the Training and Testing are downloaded and the train.csv, test.csv are put in the dataset folder. Open jupyter notebook and run the code individually for better understanding.

Phase of the proposed ML based diabetes prediction model. In the first phase, every dataset is pre-processed.

In the second stage, the pre-processed datasets are feed into the different machine learning algorithms.

In the third phase, the output of the models is then analyzed using various metrics. In the later phase, the model that provides the highest accuracy is adopted to detect diabetes of any individual and integrated with a web-based application.

This web-based application is developed using Flask of python programming language.



Building robust classifier by combining all models:

```
# Training and testing SVM Classifier

svm_model = SVC()

svm_model.fit(X_train, y_train)

preds = svm_model.predict(X_test)
```

```
print(f"Accuracy on train data by SVM Classifier\
: {accuracy_score(y_train,
svm_model.predict(X_train))*100}")

print(f"Accuracy on test data by SVM Classifier\
: {accuracy_score(y_test, preds)*100}")
cf_matrix = confusion_matrix(y_test, preds)
plt.figure(figsize=(12,8))
sns.heatmap(cf_matrix, annot=True)
plt.title("Confusion Matrix for SVM Classifier on Test Data")
plt.show()

# Training and testing Naive Bayes Classifier
nb_model = GaussianNB()
nb_model.fit(X_train, y_train)
preds = nb_model.predict(X_test)
print(f"Accuracy on train data by Naive Bayes Classifier\
: {accuracy_score(y_train, nb_model.predict(X_train))*100}")
```

```
print(f"Accuracy on test data by Naive Bayes Classifier\
: {accuracy_score(y_test, preds)*100}")
cf_matrix = confusion_matrix(y_test, preds)
plt.figure(figsize=(12,8))
sns.heatmap(cf_matrix, annot=True)
plt.title("Confusion Matrix for Naive Bayes Classifier on Test Data")
plt.show()

# Training and testing Random Forest Classifier
rf_model = RandomForestClassifier(random_state=18)
rf_model.fit(X_train, y_train)
preds = rf_model.predict(X_test)
print(f"Accuracy on train data by Random Forest Classifier\
: {accuracy_score(y_train, rf_model.predict(X_train))*100}")

print(f"Accuracy on test data by Random Forest Classifier\
: {accuracy_score(y_test, preds)*100}")
```

cf_matrix = confusion_matrix(y_test, preds)

plt.figure(figsize=(12,8))

sns.heatmap(cf_matrix, annot=True)

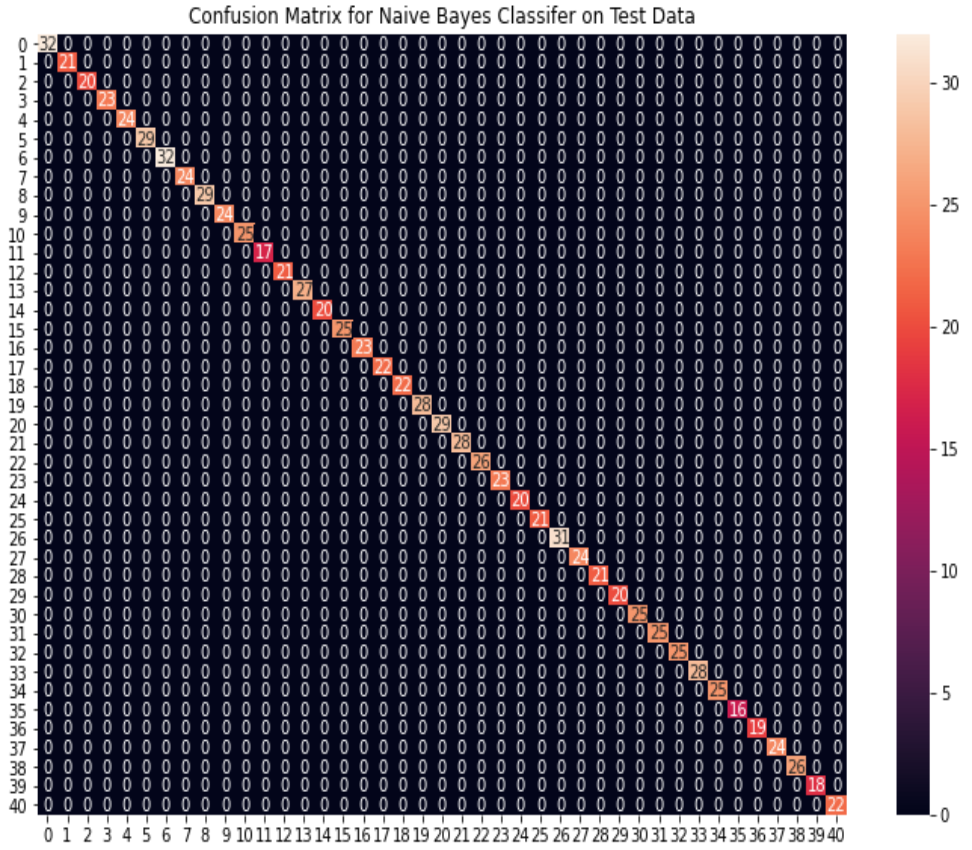
plt.title("Confusion Matrix for Random Forest Classifier on Test Data")

plt.show()

Output:

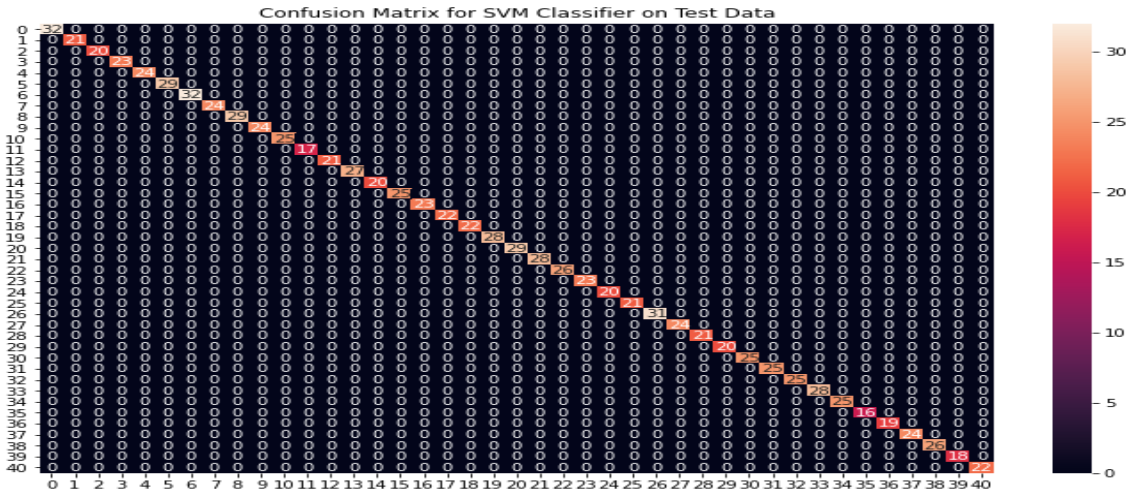
```
Accuracy on train data by SVM Classifier: 100.0

Accuracy on test data by SVM Classifier: 100.0
```

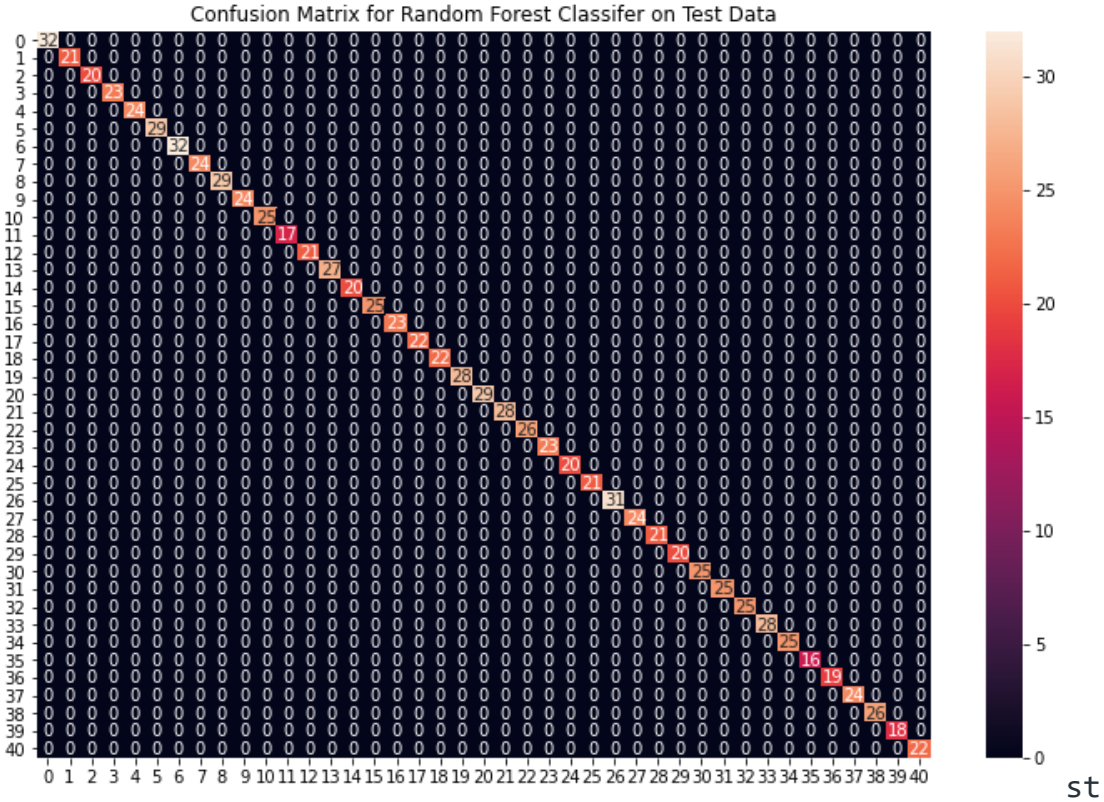


Accuracy on train data by Naive Bayes Classifier: 100.0

Accuracy on test data by Naive Bayes Classifier: 100.0



Accuracy on train data by Random Fore



st

Classifier: 100.0

Accuracy on test data by Random Forest Classifier: 100.0

From the above confusion matrices, we can see that the models are performing very well on the unseen data. Now we will be training the models on the whole train data present in the dataset that we downloaded and then test our combined model on test data present in the dataset.

REPORT:

All the above instruction are installed and executed successfully.

Project By :

Name : C . Thiyanadurai

Dept :CSE III YEAR

Reg no :621421104054

College code: 6214

Group :IBM – Group 5