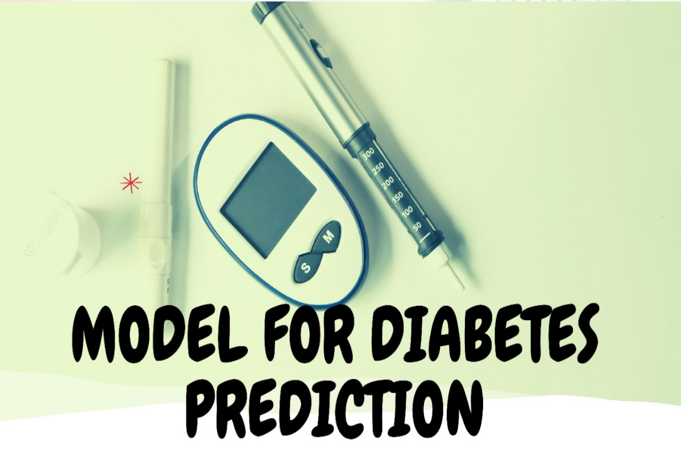
BVRIT HYDERABAD College of Engineering for Women

Department of Information Technology

Team 1
A.S.Anusha - 18WH1A1201
M.V.Harini- 18WH1A1233
D.Harshitha - 18WH1A1253
K.Divya- 18WH1A1258

Under the guidance of :
Guide Name: Dr P.Kayal
Designation: Associate Professor


300
250
200
150
100
50
S
M
MODEL FOR DIABETES
PREDICTION

Agenda

- Abstract
- Introduction
- Problem Definition
- Literature Survey
- About Dataset
- Design of Project
- Flow of the Model
- Stage-I Partial Implementation
- Stage-II Extendend Design
- Results and Predictions
- References

Abstract

Diabetes mellitus is a chronic disease characterized by hyperglycemia. It may cause many complications.In this project, we focus on building machine learning models to determine whether a patient admitted to an ICU has been diagnosed with a particular type of diabetes, Diabetes Mellitus.Since the dataset has 371 attributes,we use feature engineering to optimise the data and then predict diabetes mellitus using various ML algorithms like decision trees,random forests and LightGBM.

Introduction

- Diabetes mellitus (DM), is a group of metabolic diseases in which there are high blood sugar levels over a prolonged period.
- According to the growing morbidity in recent years, in 2040, the world's diabetic patients will reach 642 million.
- We focused on model to determine whether a patient admitted to an ICU has been diagnosed with a particular type of diabetes, Diabetes Mellitus.

Problem Definition

- The constant hyperglycemia of diabetes is related to long-haul harm, brokenness, and failure of various organs, particularly the eyes, kidneys, nerves, heart, and veins.
- The objective of this project is to make use of significant features, design a prediction algorithm using Machine learning and find the optimal classifier to give the closest result comparing to clinical outcomes.
- The proposed method aims to focus on selecting the attributes that ail in early detection of Diabetes Mellitus using Predictive analysis.

Literature Survey

Title	Outcome
Analysis of diabetes mellitus for early prediction using optimal features selection. Author: N. Sneha & Tarun Gangil	The point of this examination is to the finding of diabetes illness, which is a standout amongst the most vital infections in the restorative field utilizing Generalized Discriminant Analysis (GDA) and Least Square Support Vector Machine
Predicting Diabetes Mellitus With Machine Learning Techniques. Author: Quan Zou	Principal component analysis (PCA) and minimum redundancy maximum relevance (mRMR) to reduce the dimensionality. The results showed that prediction with random forest could reach the highest accuracy when all the attributes were used.
Prediction of Diabetes using Classification Algorithms Author: Deepti Sisodia and Dilip Singh Sisodia	Decision Tree, SVM and Naive Bayes are used in this experiment to detect diabetes at an early stage. Experiments are performed on Pima Indians Diabetes Database (PIDD) which is sourced from UCI machine learning repository.

Literature Survey

Title	Outcome
Machine Learning Based Diabetes Classification and Prediction for Healthcare Applications Author: Umair Muneer Butt,Sukumar Letchumanan, Mubashir Ali	In this paper for diabetes classification and early-stage identification, three different classifiers have been employed, i.e., random forest (RF), multilayer perceptron (MLP), and logistic regression (LR).
Research on Diabetes Prediction Method Based on Machine Learning Author: Jingyu Xue1st,a, Fanchao Min	In this paper,Support Vector Machine (SVM), Naive Bayes classifier and LightGBM to train on the actual data of 520 diabetic patients and potential diabetic patients aged 16 to 90. Through comparative analysis of classification and recognition accuracy, the performance of support vector machine is the best.
Diabetes prediction model based on an enhanced deep neural network Author: Huaping Zhou, Raushan Myrzashova	In this paper the model is mainly built using the hidden layers of a deep neural network to prevent overfitting.It showed much accuracy with DLPD (Deep Learning for Predicting Diabetes) model. The best training accuracy of the diabetes type data set is 94.02174%.

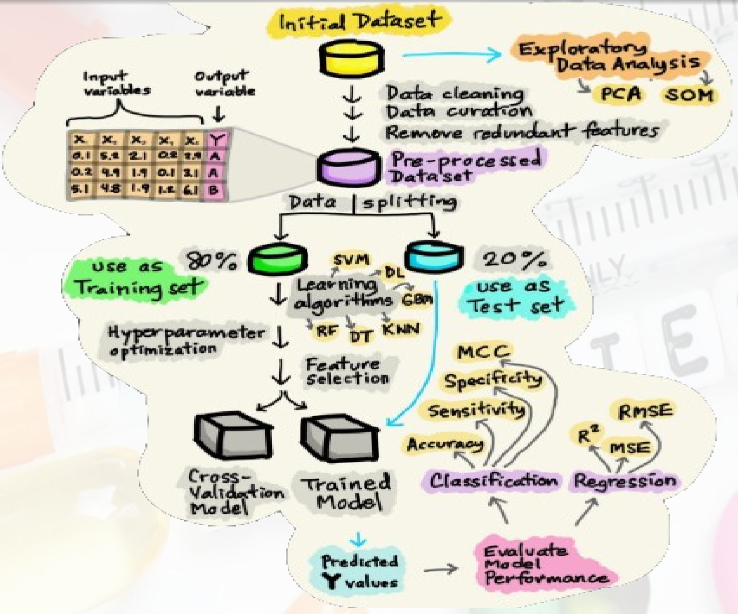
Literature Survey

Title	Outcome
LGBM Classifier based Technique for Predicting Type-2 Diabetes Author: B. Shamreen Ahamed, Dr. Meenakshi Sumeet Arya	The author has used the existing PIMA Indian Dataset for diabetes prediction and detection using LGBM Algorithm. Accuracy is 95.20% Therefore by using the LGBM classifiers, we can develop a data model for diabetes detection and prediction
Prediction of Gestational Diabetes Based on LightGBM Author: Fan Hou,ZhiXiang Cheng	This paper says that it is important to identify the risk of diabetes. Artificial intelligence assisted diabetes genetic risk prediction rematch is selected to construct the LightGBM prediction model and compare with Random Forest and XGBoost on the ROC curve.The results show that the AUC of LightGBM is 85.2%.

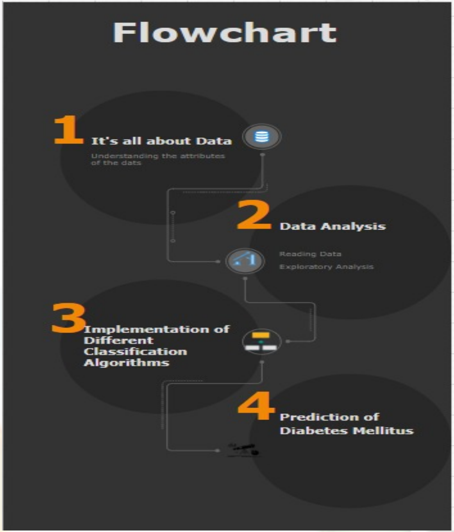
About Dataset

- The dataset consists of 1,30,157 rows and 317 columns with different data types like int, float, string etc...
- **Splitting of Data**
  - 80% of data is used for training
  - 20% of data is used for testing.

Design of the project



Flow of the Model



Stage-I Partial Implementation

- Dropping unnecessary columns

```
df_train.drop('Unnamed: 0',axis=1,inplace=True)
print('Training data shape: ', df_train.shape)
df_train.head()
```

```
Training data shape:  (130157, 180)
```

	encounter_id	hospital_id	age	bmi	elective_surgery	ethnicity	gender	height	hospital_admit_source
0	214826	118	68.0	22.732803	0	Caucasian	M	180.3	Floo
1	246060	81	77.0	27.421875	0	Caucasian	F	160.0	Floo
2	276985	118	25.0	31.952749	0	Caucasian	F	172.7	Emergency Departmen
3	262220	118	81.0	22.635548	1	Caucasian	F	165.1	Operating Roon
4	201746	33	19.0	NaN	0	Caucasian	M	188.0	Na

5 rows × 180 columns

Stage-I Partial Implementation

- Finding missing data

```
missing_values_train = missing_values_table(df_train)
missing_values_train[:20].style.background_gradient(cmap='Greens')
```

```
Your selected dataframe has 180 columns.
There are 160 columns that have missing values.
```

	Missing Values	% of Total Values
h1_bilirubin_min	119861	92.100000
h1_bilirubin_max	119861	92.100000
h1_albumin_max	119005	91.400000
h1_albumin_min	119005	91.400000
h1_lactate_max	118467	91.000000
h1_lactate_min	118467	91.000000
h1_pao2fio2ratio_min	113397	87.100000
h1_pao2fio2ratio_max	113397	87.100000
h1_arterial_ph_max	107849	82.900000
h1_arterial_ph_min	107849	82.900000
h1_arterial_pco2_min	107666	82.700000

Stage-I Partial Implementation

- Preprocessing Data

```
train_copy = df_train.copy()
test_copy = df_test.copy()
```

```
train_copy['source'] = 0
test_copy['source'] = 1
```

```
all_data = pd.concat([train_copy, test_copy], axis=0, copy=True)
del train_copy
del test_copy
gc.collect()
```

```
3264
```

```
all_data.drop('encounter_id',axis=1,inplace=True)
```

```
df_train['hospital_id'].isin(df_test['hospital_id']).value_counts()
```

```
False    130157
Name: hospital_id, dtype: int64
```

```
all_data.drop('hospital_id',axis=1,inplace=True)
```

Stage-I Partial Implementation

- Encoding Categorical Data

```python
objList = all_data.select_dtypes(include = "object").columns
print (objList)


# Create a label encoder object
le = LabelEncoder()
for feat in objList:
    all_data[feat] = le.fit_transform(all_data[feat].astype(str))

print (all_data.info())
```

```
Index(['ethnicity', 'gender', 'hospital_admit_source', 'icu_admit_source',
       'icu_stay_type', 'icu_type'],
      dtype='object')
<class 'pandas.core.frame.DataFrame'>
Int64Index: 140391 entries, 0 to 10233
Columns: 179 entries, age to source
dtypes: float64(158), int64(21)
memory usage: 192.8 MB
None
```

Stage-I Partial Implementation

- Light GBM Predictions

```
[ ] lgb_predictions = lgb_model.predict_proba(test)[:, 1]
    lgb_predictions

array([0.04808896, 0.11977739, 0.12393632, ..., 0.08444076, 0.01472031,
       0.01385323])
```

Stage-II Extendend Design

- Dropping Identical Columns

```
[ ]  cols_to_drop = []
     for i, col_1 in enumerate(all_data.columns):
         for col_2 in all_data.columns[(i+1):]:
             if all_data[col_1].equals(all_data[col_2]):
                 print(f"{col_1} and {col_2} are identical.")
                 cols_to_drop.append(col_2)

     all_data.drop(cols_to_drop, axis=1, inplace = True)

     paco2_apache and paco2_for_ph_apache are identical.
```

```
[ ]  print(cols_to_drop)

     ['paco2_for_ph_apache']
```

```
[ ]  all_data.shape

     (140391, 172)
```

Stage-II Extended Design

- Outlier Detection

```
[ ] def subset_by_iqr(df, column, whisker_width=1.5):
        q1 = df[column].quantile(0.25)
        q3 = df[column].quantile(0.75)
        iqr = q3 - q1
        filter = (df[column] >= q1 - whisker_width*iqr) & (df[column] <= q3 + whisker_width*iqr)
        return df.loc[filter]


    for feature in all_data.columns:
        cleaned_train_data = subset_by_iqr(train_data, feature, whisker_width=1.5)

    cleaned_train_data.shape

    (117191, 131)
```

Stage-II Extended Design

- Correlation Implementation

```
[ ]  def get_correlation(data, threshold):
         corr_col = set()
         corrmat = data.corr()
         for i in range(len(corrmat.columns)):
             for j in range(i):
                 if abs(corrmat.iloc[i, j])> threshold:
                     colname = corrmat.columns[i]
                     corr_col.add(colname)
         print(corrmat)
         return corr_col
```

```
[ ]  corr_features = get_correlation(all_data, 0.80)
     len(corr_features)
```

Stage-II Extended Design

- Removing Correlated Data

```
[ ] print(corr_features)

{'d1_hematocrit_min', 'd1_mbp_max', 'd1_sysbp_noninvasive_min', 'icu_stay_type_transfer', 'd1_wbc_mi

[ ] all_data_uncorr = all_data.drop(labels=corr_features, axis = 1)
    print('original size of data: ',all_data.shape)
    print('After removing co related features: ',all_data_uncorr.shape)

original size of data:  (127425, 131)
After removing co related features:  (127425, 103)
```

Stage-II Extended Design

- Getting Train and Test Data

```
[ ] data = all_data_uncorr[all_data_uncorr.train_data==1].drop(['train_data'], axis =1)
    print(data.shape)
    print(all_data_uncorr.shape)
    x = data.drop(['diabetes_mellitus'], axis =1)
    y = data['diabetes_mellitus']
    x_train,x_val,y_train,y_val = train_test_split(x,y,test_size=0.2, random_state = 40)
    test = all_data_uncorr[all_data.train_data==0].drop(['train_data', 'diabetes_mellitus'], axis =1)
    print(test.shape)
    print(x.shape)

    (117191, 102)
    (127425, 103)
    (10234, 101)
    (117191, 101)
```

Stage-II Extended Design

- LightGBM Implementation

```
[ ]
lgb = lgb.LGBMClassifier(silent=False)
lgb.fit(x_train, y_train)
y_pred_lgb=lgb.predict(x_val)

print("Accuracy:",accuracy_score(y_val, y_pred_lgb))
print("Precision:",precision_score(y_val, y_pred_lgb))
print("Recall:",recall_score(y_val, y_pred_lgb))
print("F1 score:",f1_score(y_val, y_pred_lgb))

y_pred_proba = lgb.predict_proba(x_val)[:,1]
print("AUC score:",roc_auc_score(y_val, y_pred_proba))
plot_confusion_matrix(lgb, x_val, y_val)
```

```
Accuracy: 0.8376210589188958
Precision: 0.685752688172043
Recall: 0.4917116422513493
F1 score: 0.5727436012572968
AUC score: 0.8638601228724088
```

Stage-II Extended Design

- Important Features By LightGBM

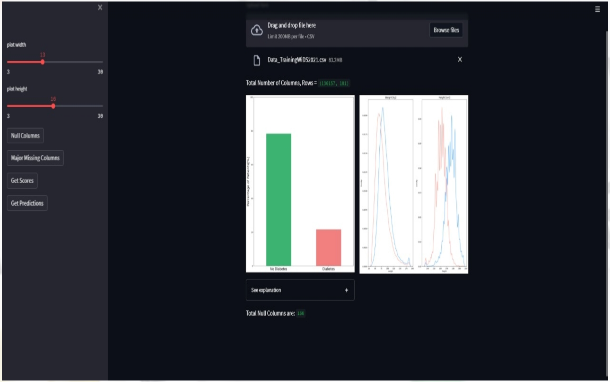
```
feature_importance_values = lgb.feature_importances_
feature_importances = pd.DataFrame({'feature': features, 'importance': feature_importance_val
print(feature_importances)
feature_importances_sorted = plot_feature_importances(feature_importances)
```

```
                   feature  importance
0                      age         161
1                      bmi         196
2        elective_surgery           0
3                   height          26
4                   icu_id         332
..                     ...         ...
96    icu_type_Cardiac ICU           2
97           icu_type_MICU           1
98   icu_type_Med-Surg ICU           4
99      icu_type_Neuro ICU           4
100          icu_type_SICU           0

[101 rows x 2 columns]
```

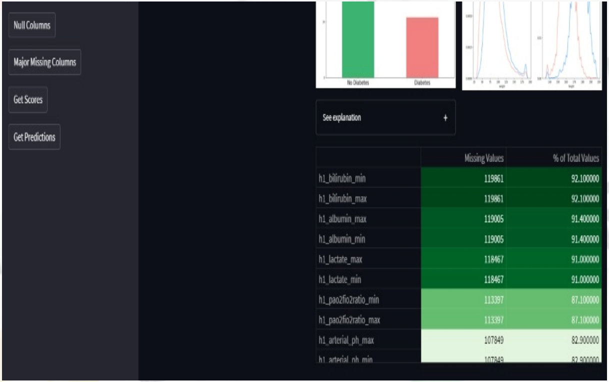
Results and Predictions

- Preprocessing of Train data



Results and Predictions

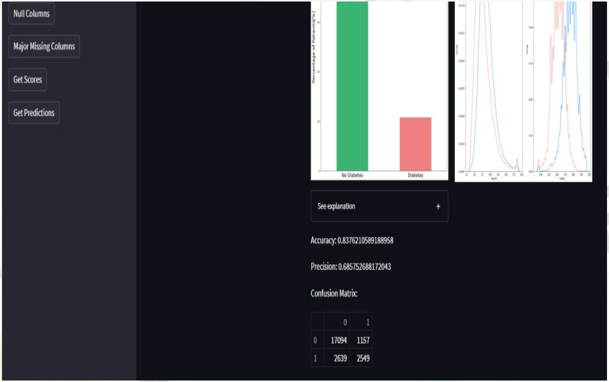
- Major Missing value columns



	Missing Values	% of Total Values
h1_bilirubin_min	119861	92.100000
h1_bilirubin_max	119861	92.100000
h1_albumin_max	119005	91.400000
h1_albumin_min	119005	91.400000
h1_lactate_max	118467	91.000000
h1_lactate_min	118467	91.000000
h1_pao2fio2ratio_min	113397	87.100000
h1_pao2fio2ratio_max	113397	87.100000
h1_arterial_ph_max	107849	82.900000
h1_arterial_ph_min	107849	82.900000

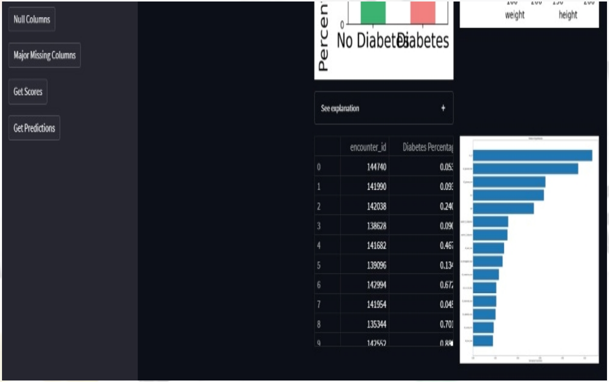
Results and Predictions

- Results and Scores



Results and Predictions

- Predictions



References

- https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0175
- https://www.frontiersin.org/arti cles/10.3389/fgene.2018.00515 /full
- https://www.sciencedirect.com/science/article/pii/S1877050918308548
- https://jwcn-eurasipjournals.springeropen.com/articles/10.1186/s13638-020
- https://www.hindawi.com/journals/jhe/2021/9930985/
- https://iopscience.iop.org/article/10.1088/1742-6596/1684/1/012062/pdf
- https://ejmcm.com/article$_{9}403_{1}5c24bd9c676c28d90c3fc5fad8b42ea.pdf$
- https://dl.acm.org/doi/10.1145/3433996.3434025
- https://towardsdatascience.com/hyperparameters-optimization-526348bb8e2d
- https://medium.com/analytics-vidhya/what-is-correlation-4fe0c6fbed47
- https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e2


THANK YOU.