Diabetes Prediction: Classification Model

YorkU

CSML1000 Machine Learning in Business Context - Blended Live Online Fall 2024

Project #1

20th October, 2024

Arslan Abakarov
Daniel (Jee Hwan) Lee
Karen Krucik, MBA(Oxon)
Aayush
Mathi Mahalingam

Abstract

Diabetes is a chronic health condition that affects the body's ability to regulate blood sugar level, leading to serious health complications if left unmanaged. It is one of the most prevalent diseases globally, with rising incidence rates driven by factors such as poor diet. Early diagnosis and effective management are critical in reducing the risk of complications such as cardiovascular disease, kidney failure, and nerve damage. Public health initiatives are increasingly focused on prevention, early detection, and lifestyle interventions to mitigate the growing burden of diabetes on healthcare systems.

Background

The Public Health Authority of Canada is launching a campaign to identify high-risk individuals for diabetes, aiming to improve early detection and optimize healthcare resources. The York University student team is developing a predictive model to identify key diabetes risk factors. This data-driven model will improve the precision of risk assessments, enabling more effective targeting of preventive population interventions and reducing healthcare costs.

Objective

The objective is to develop a supervised binary classification model to predict diabetes risk in individuals, with the model's accuracy exceeding 95%. This level of accuracy will allow the Public Health Authority to effectively deploy the model for targeted preventative measures, leading to improved early detection of at-risk indivduals. By meeting this performance threshold, the model will improve early detection and enhance resource allocation for diabetes prevention.

Data Analysis

The dataset used to build the diabetes prediction model comprises medical and demographic information from 100,000 patient samples, along with their diabetes status. It includes 8 key features that serve as predictors for diabetes, offering valuable insights for healthcare professionals to assess and predict diabetes risk in individuals. These features are crucial in identifying patterns and risk factors associated with the disease. Sourced from Kaggle, the dataset provides useful information for developing a highly accurate predictive model, ultimately supporting more effective diabetes detection and prevention strategies in healthcare settings.

Description of features:

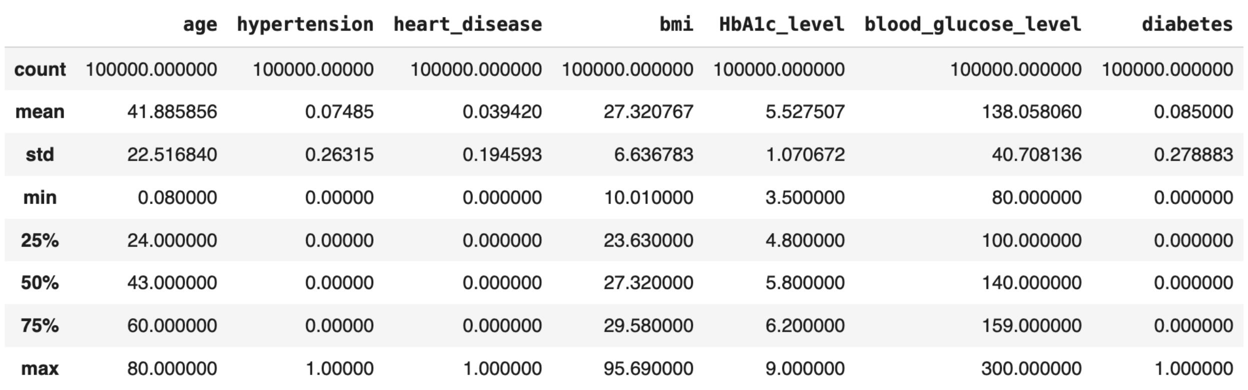
Column Name	**Impact**
Gender	Men are believed to be at a higher risk for Type 2 diabetes compared to women, based on hormonal differences and fat distribution.
Age	45 years is the threshold after which risk of developing type 2 diabetes is increased. Insulin resistance increment also is related to aging.
Hypertension	High blood pressure (hypertension) affects insulin resistance which impacts type 2 diabetes
Heart Disease	Heart disease is a high influencer for diabetes, diabetes causes vascular damage which increases the risk of heart disease.

Smoking History	Smoking increases risk of cardiovascular disease, it increases risk of type 2 diabetes that can worsen complications in those already diagnosed.
Body Mass Index	Higher the BMI, higher insulin resistance. Body weight affects BMI.
HbA1c Level	HbA1c (glycated hemoglobin) levels are indicators of average blood sugar levels over the past 2-3 months. Higher HbA1c levels increase risk of diabetes-related complications.
Blood glucose level	Raised blood glucose levels are a primary indicator of diabetes. High levels can lead to diabetes

Data Exploration

The initial exploration of the dataset will focus on identifying trends and correlations between the features and diabetes risk. Key steps will include analyzing the data for missing values, outliers, and any potential imbalances between positive and negative cases (patients with and without diabetes). This process will ensure the dataset is clean, well-structured, and ready for model development. ‘

The following table provides statistical analysis of the data set, it assists in understanding distribution of value:

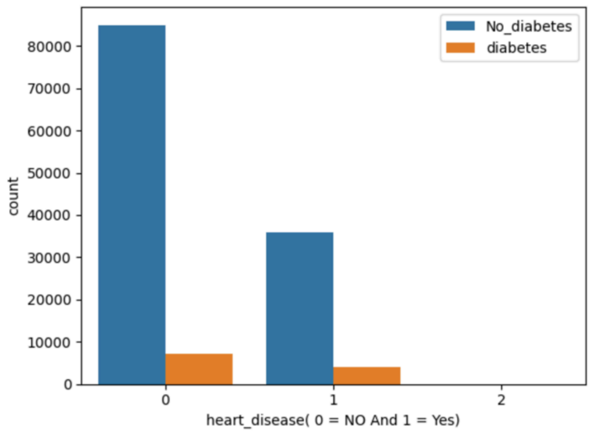


	age	hypertension	heart_disease	bmi	HbA1c_level	blood_glucose_level	diabetes
count	100000.000000	100000.00000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000
mean	41.885856	0.07485	0.039420	27.320767	5.527507	138.058060	0.085000
std	22.516840	0.26315	0.194593	6.636783	1.070672	40.708136	0.278883
min	0.080000	0.00000	0.000000	10.010000	3.500000	80.000000	0.000000
25%	24.000000	0.00000	0.000000	23.630000	4.800000	100.000000	0.000000
50%	43.000000	0.00000	0.000000	27.320000	5.800000	140.000000	0.000000
75%	60.000000	0.00000	0.000000	29.580000	6.200000	159.000000	0.000000
max	80.000000	1.00000	1.000000	95.690000	9.000000	300.000000	1.000000

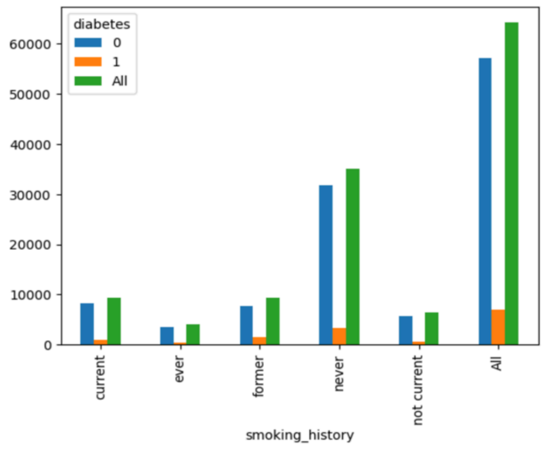
** Smoking history is not part of the statistical analysis chart as it comprises 5 different string values.*

- Age: Age distribution ranges from infants to old adults, with median age being 43
- Hypertension: It is present in around 7.5% population in the dataset
- Heart disease: It only affect around 4% individuals from the dataset
- BMI: Majority of the population seems to have overweight range
- Diabetes: 8.5% population from dataset is confirmed to have diabetes

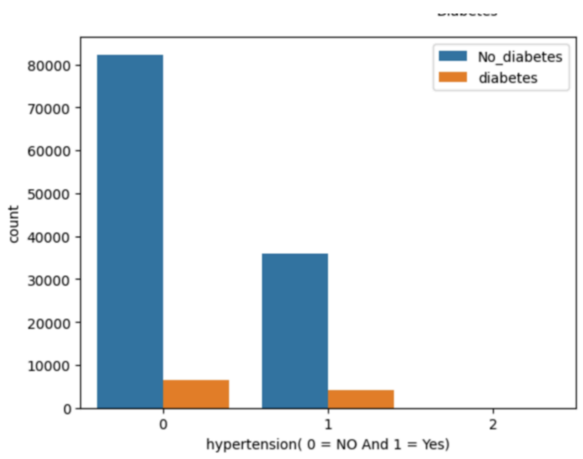
The following graphs indicate representation of features across the data set. In this exploration, we cannot suggest strong correlation between each feature and the risk of having diabetes:



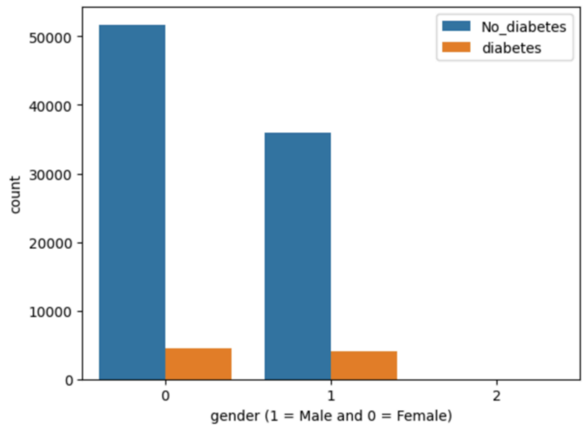
Patients having Heart Disease with or Without Diabetes



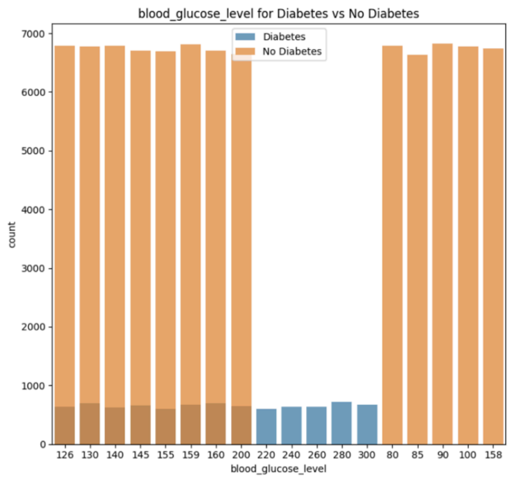
Smoking History of Patients With or Without Diabetes



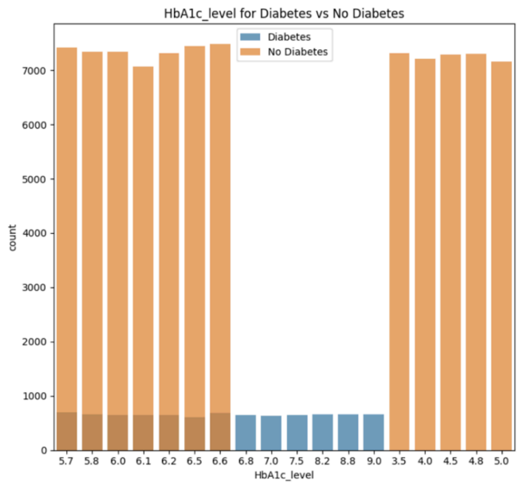
Patients Having Hypertensions With or Without Diabetes



Male and Female Ratio With or Without Diabetes



Blood Glucose Level of Patients With or Without Diabetes



HbA1c Level of Patients With or Without Diabetes

Data Preparation and Feature Engineering

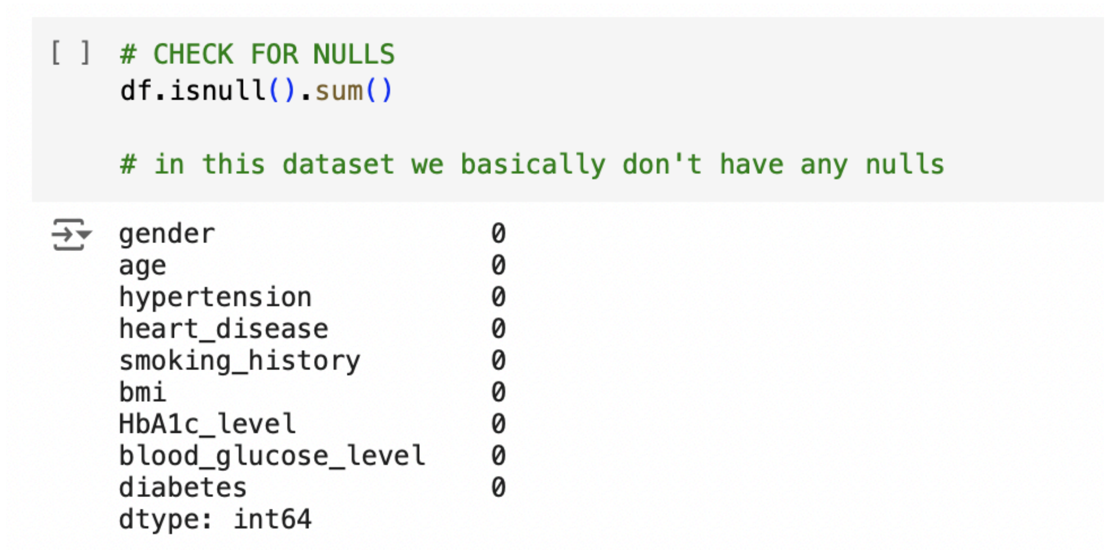
Feature engineering helps to enhance the quality and relevance of the data, which ultimately leads to better-performing models. For this diabetes prediction model, the goal is to achieve high predictability on unseen data and effective feature engineering plays a key role in that.

Key steps in this process include:

- Checking for missing values
- Checking for outliers
- Encoding categorical variables
- Managing duplicates
- Data correlation and observations
- Addressing class imbalance

Checking for missing values

To ensure the quality of the dataset, the **isnull()** function from the Pandas library will be used to check for missing values. Missing or empty values can significantly reduce the model's performance by introducing inaccuracies and bias in predictions.



```
[ ] # CHECK FOR NULLS
    df.isnull().sum()

    # in this dataset we basically don't have any nulls
```

```
gender               0
age                  0
hypertension         0
heart_disease        0
smoking_history      0
bmi                  0
HbA1c_level          0
blood_glucose_level  0
diabetes             0
dtype: int64
```

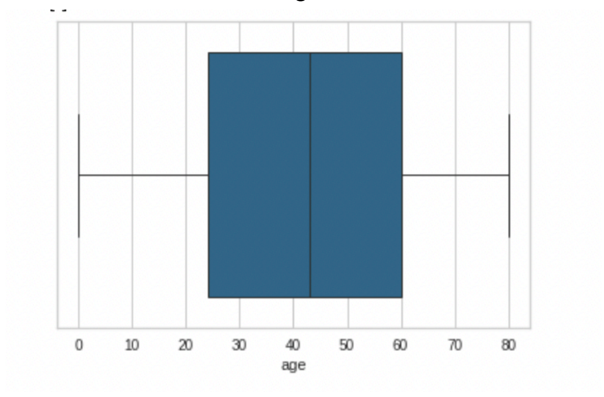
In the code above, we check for missing values, and as demonstrated, none were found. Therefore, no additional processing was required to handle missing data.

Further, during data exploration, we identified 18 entries where the gender was labeled as "other." Given the extremely small number of these instances, we decided to remove these entries during data preparation. This decision helps maintain data consistency and avoids introducing noise that could affect the model’s performance.

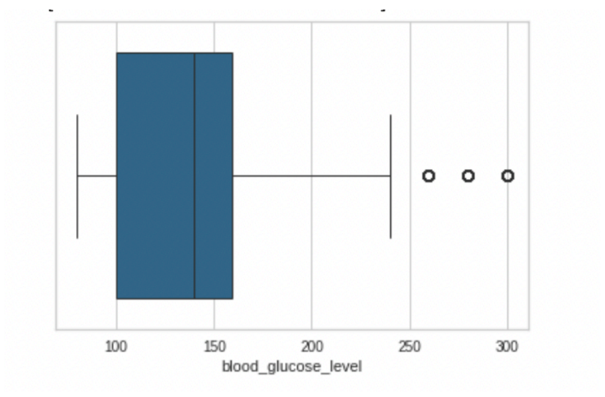
Checking for Outliers

As a next step, we focus on identifying outliers, which can negatively impact model training and performance by skewing predictions. The box plots below illustrate the presence of outliers in the following features: Age, BMI, HbA1c_level and blood_glucose_level.

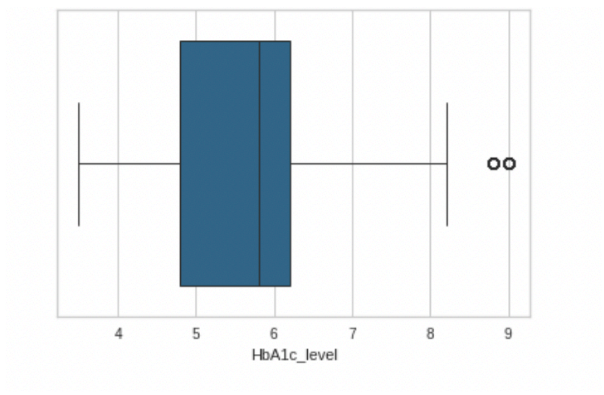
Age



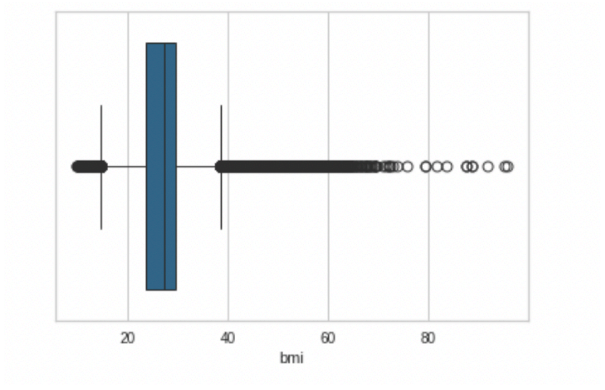
Blood Glucose



HbA1c Level



BMI

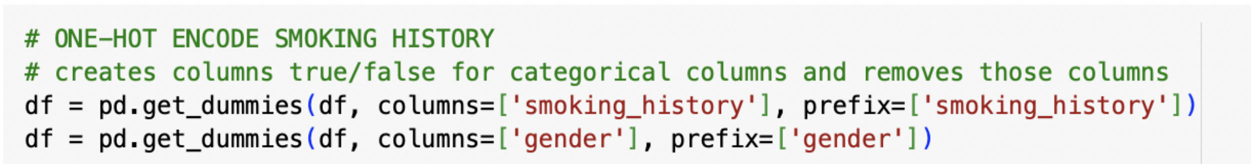


In the Box Plots, there are clear indications of outliers. However, it is not immediately clear whether the outliers are due to any errors in data entry or other causes. BMI is a highly skewed data feature, thus we applied log transformation to more effectively mitigate the data skew. The log transformation is, arguably, the most popular among the different types of transformations used to transform skewed data to approximately conform to normality.[1] HbA1c and Blood Glucose levels were not as skewed, therefore we 'Capped' the dataset to disregard the outliers.

[1]https://pmc.ncbi.nlm.nih.gov/articles/PMC4120293/#:~:text=The%20log%20transformation%20is%2C%20arguably,normal%20or%20near%20normal%20distribution.

One-Hot Encoding

In our dataset, Gender and Smoking History are not numerical values, therefore we had to apply One-Hot Encoding.



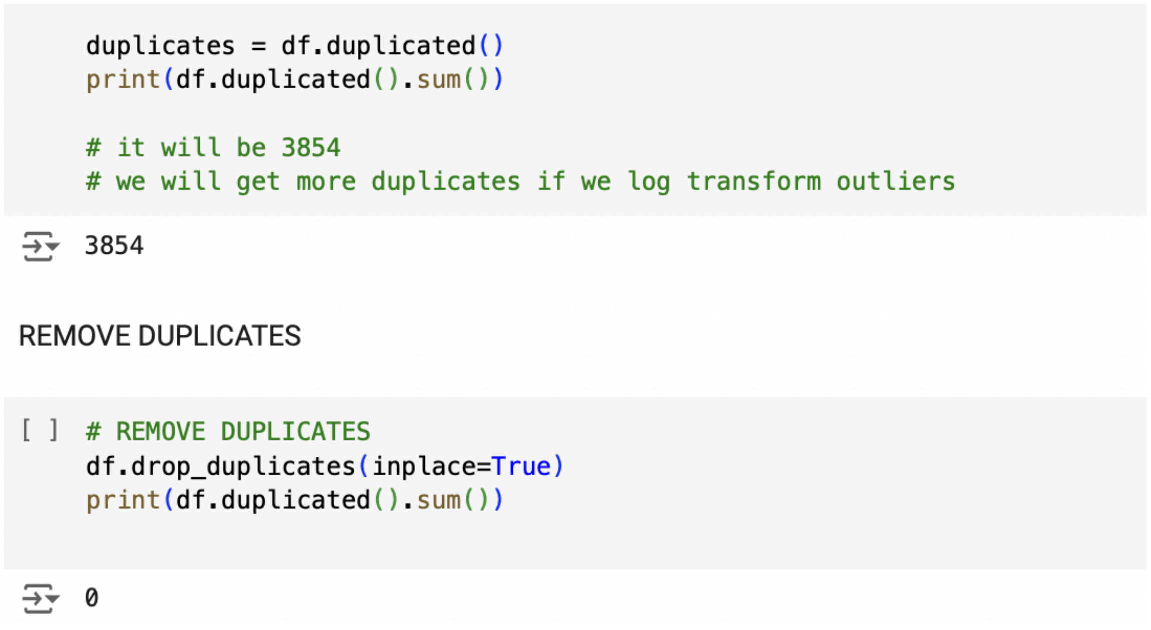
```
# ONE-HOT ENCODE SMOKING HISTORY
# creates columns true/false for categorical columns and removes those columns
df = pd.get_dummies(df, columns=['smoking_history'], prefix=['smoking_history'])
df = pd.get_dummies(df, columns=['gender'], prefix=['gender'])
```

This created additional columns for each value when the feature was non-numeric (in this instance categorical). This enables the categorical features to be provided to the algorithm.

Managing Duplicates

As part of our data preparation, we removed duplicates. We removed duplicates after the transformations listed above, as the duplicate total increased during those processes.

After transformations the duplication total was 3854. In the code below, we removed these, reducing them to zero.



```
duplicates = df.duplicated()
print(df.duplicated().sum())

# it will be 3854
# we will get more duplicates if we log transform outliers
```

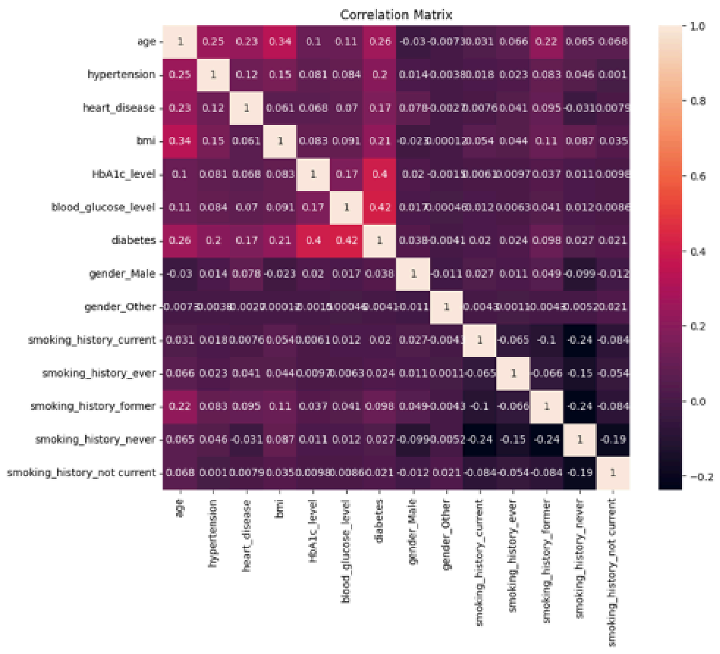
```
3854
```

REMOVE DUPLICATES

```
[ ] # REMOVE DUPLICATES
    df.drop_duplicates(inplace=True)
    print(df.duplicated().sum())
```

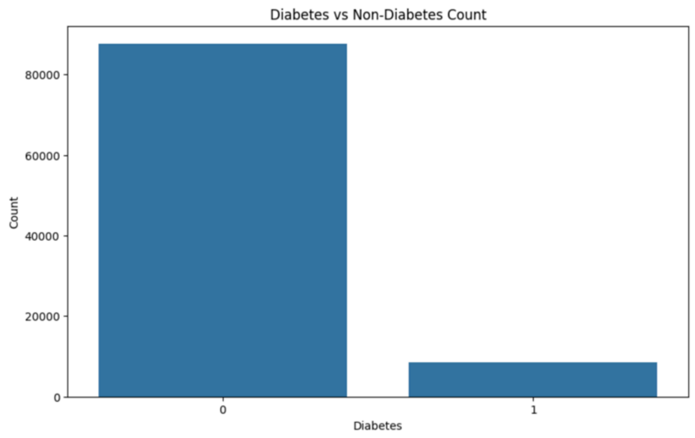
```
0
```

Data Correlation and Observations



In the heat map above, correlation and observations drawn from our dataset, we see that there is very low correlation between Gender and Diabetes Further, we also see low data correlation between certain types of Smoking History (i.e. Smoking ‘Ever’ , Smoking ‘Current’, Smoking ‘Former’, and Smoking ‘Not Current’) and Diabetes. Because of this low correlation, these features should be explored further for the Feature Engineering purposes.

Imbalanced dataset



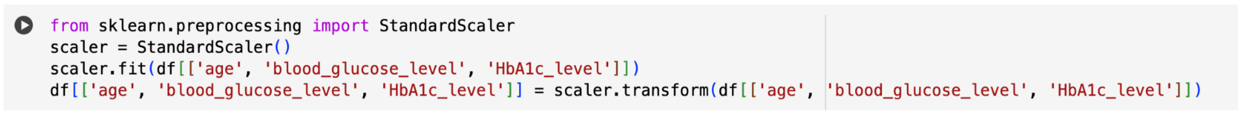
```
# CLASS PROPORTIONS
class_proportions = df['diabetes'].value_counts(normalize=True)
print(class_proportions)

diabetes_values = pd.Series(df['diabetes']).value_counts()
print(diabetes_values)
```

```
diabetes
0    0.915
1    0.085
Name: proportion, dtype: float64
diabetes
0    91500
1     8500
Name: count, dtype: int64
```

- Diabetes class proportions in the dataset indicate 91.5% population not having diabetes and only 8.5% population with diabetes
- Since there is only small proportion of entries with Diabetes, it can lead to biased models

Feature Scaling

Features like BMI, HbA1c Level, and Blood Glucose Level might have different scales. They were standardized for better model performance.



```python
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(df[['age', 'blood_glucose_level', 'HbA1c_level']])
df[['age', 'blood_glucose_level', 'HbA1c_level']] = scaler.transform(df[['age', 'blood_glucose_level', 'HbA1c_level']])
```

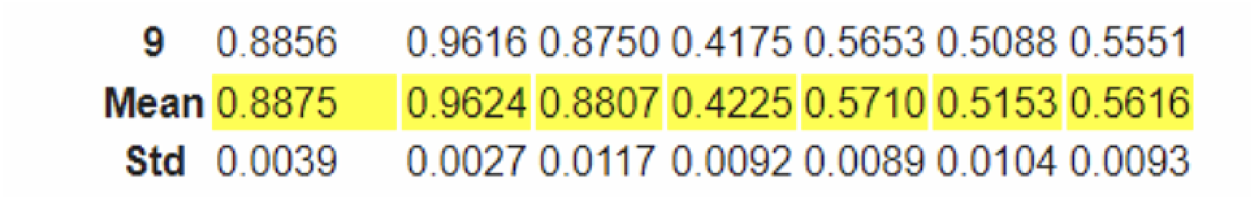
In the above, Age, Blood Glucose Level, and HbA1c Levels were scaled.

This enabled the features to have a more similar range of values (a more similar scale) without losing the implications of varying values and would lead to more useful training outcomes.

Feature selection

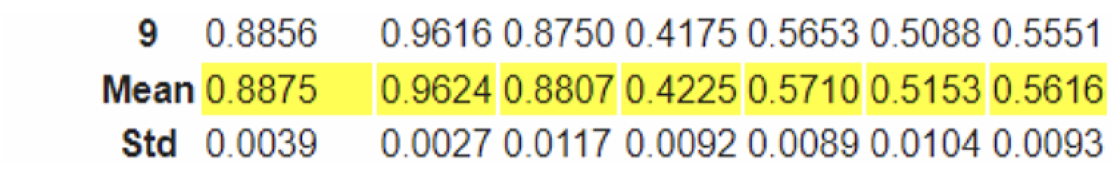
For the current project, we decided to use the full list of features and then test it against a reduced set of features by excluding low correlation features out of correlation matrix plot. After looking at the models' performance, we found that the full set of features actually provides the same results as a reduced set of features. So, we decided to go with a full set of features that we are going to train the final model on.

Reduced set of features



9	0.8856	0.9616	0.8750	0.4175	0.5653	0.5088	0.5551
Mean	0.8875	0.9624	0.8807	0.4225	0.5710	0.5153	0.5616
Std	0.0039	0.0027	0.0117	0.0092	0.0089	0.0104	0.0093

Full set of features



9	0.8856	0.9616	0.8750	0.4175	0.5653	0.5088	0.5551
Mean	0.8875	0.9624	0.8807	0.4225	0.5710	0.5153	0.5616
Std	0.0039	0.0027	0.0117	0.0092	0.0089	0.0104	0.0093

Training The Model

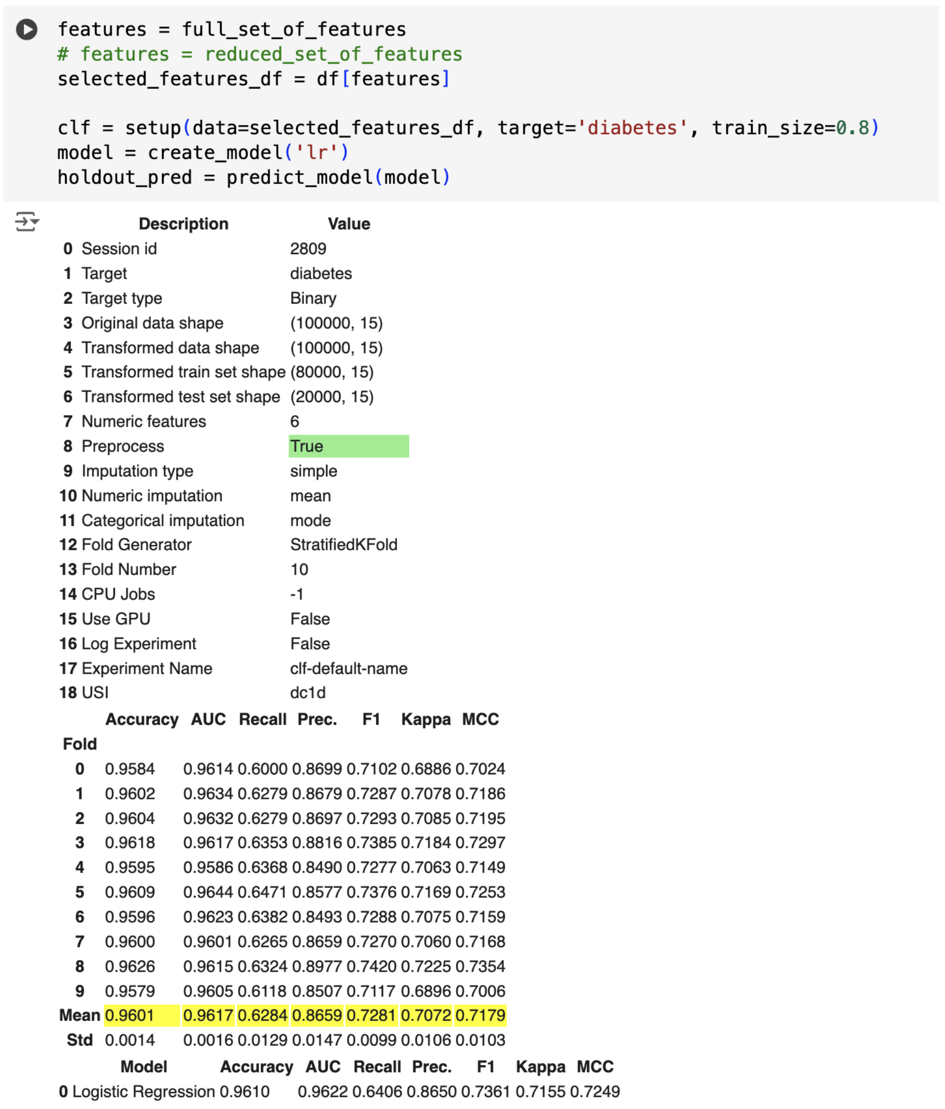
Baseline Model

Logistic Regression is often used for binary classification. As a base model, it is often judged as providing a good balance between simplicity, interpretability and performance.

To train our data, we divided our data into 80% seen and 20% unseen data.

```
clf = setup(data=selected_features_df, target='diabetes', train_size=0.8)
model = create_model('lr')
```

To begin, we started with a full set of Features.



```
features = full_set_of_features
# features = reduced_set_of_features
selected_features_df = df[features]

clf = setup(data=selected_features_df, target='diabetes', train_size=0.8)
model = create_model('lr')
holdout_pred = predict_model(model)
```

```
       Description                 Value
 0  Session id                    2809
 1  Target                        diabetes
 2  Target type                   Binary
 3  Original data shape           (100000, 15)
 4  Transformed data shape        (100000, 15)
 5  Transformed train set shape   (80000, 15)
 6  Transformed test set shape    (20000, 15)
 7  Numeric features              6
 8  Preprocess                    True
 9  Imputation type               simple
10  Numeric imputation            mean
11  Categorical imputation        mode
12  Fold Generator                StratifiedKFold
13  Fold Number                   10
14  CPU Jobs                      -1
15  Use GPU                       False
16  Log Experiment                False
17  Experiment Name               clf-default-name
18  USI                           dc1d
       Accuracy  AUC    Recall Prec.  F1     Kappa  MCC
Fold
   0   0.9584    0.9614 0.6000 0.8699 0.7102 0.6886 0.7024
   1   0.9602    0.9634 0.6279 0.8679 0.7287 0.7078 0.7186
   2   0.9604    0.9632 0.6279 0.8697 0.7293 0.7085 0.7195
   3   0.9618    0.9617 0.6353 0.8816 0.7385 0.7184 0.7297
   4   0.9595    0.9586 0.6368 0.8490 0.7277 0.7063 0.7149
   5   0.9609    0.9644 0.6471 0.8577 0.7376 0.7169 0.7253
   6   0.9596    0.9623 0.6382 0.8493 0.7288 0.7075 0.7159
   7   0.9600    0.9601 0.6265 0.8659 0.7270 0.7060 0.7168
   8   0.9626    0.9615 0.6324 0.8977 0.7420 0.7225 0.7354
   9   0.9579    0.9605 0.6118 0.8507 0.7117 0.6896 0.7006
Mean   0.9601    0.9617 0.6284 0.8659 0.7281 0.7072 0.7179
 Std   0.0014    0.0016 0.0129 0.0147 0.0099 0.0106 0.0103
         Model              Accuracy  AUC    Recall Prec.  F1     Kappa  MCC
0 Logistic Regression 0.9610    0.9622 0.6406 0.8650 0.7361 0.7155 0.7249
```

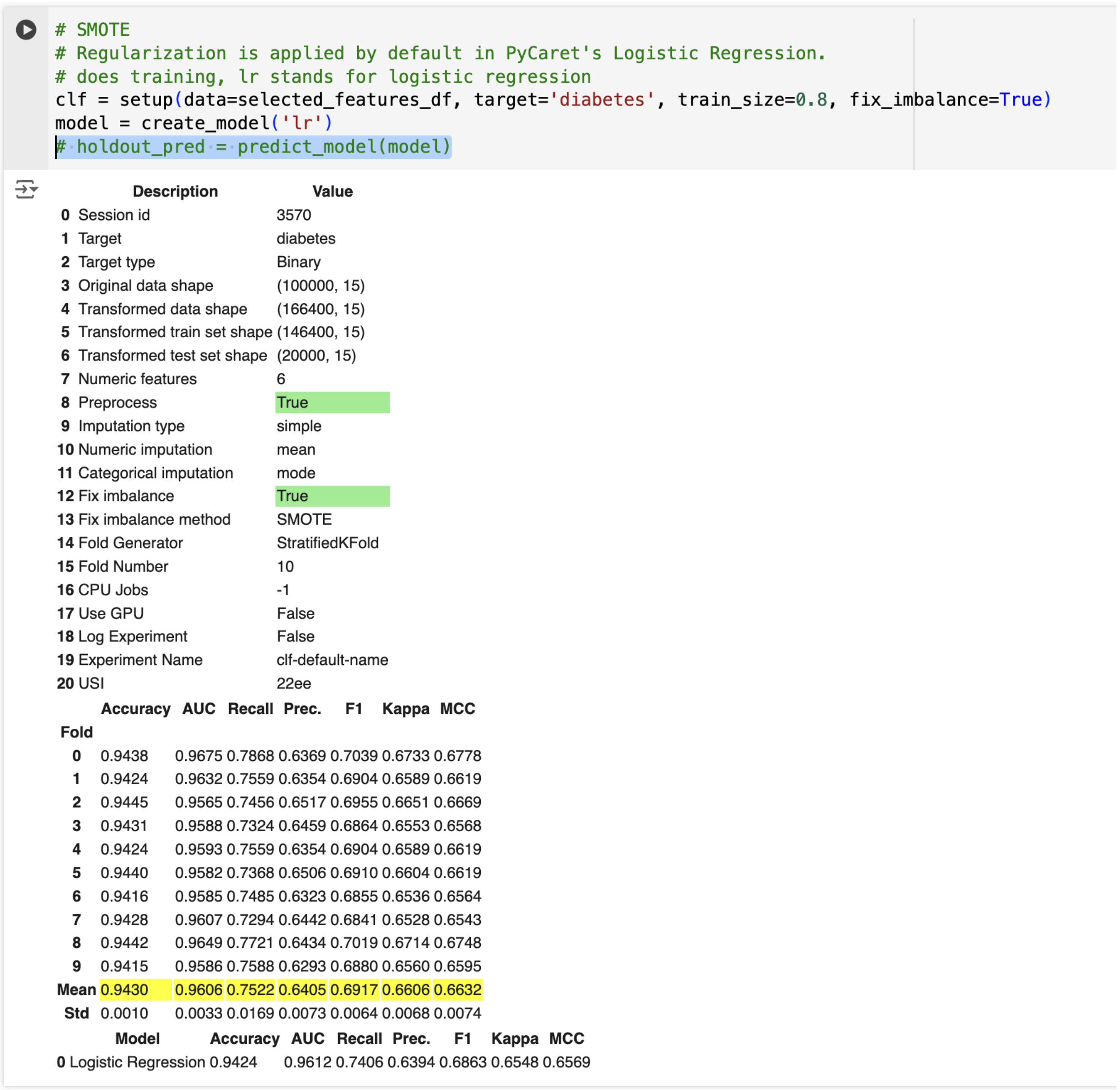
As above, we can see that the Accuracy is 0.96, AUC is 0.96, Recall is 0.64 and Precision is 0.86. However, this may also be indicative of the underlying problem with an unbalanced data set.

To remedy the distortion potentially caused by an imbalance data set, we have applied the Synthetic Minority Oversampling Technique (SMOTE) Technique.

Applied SMOTE Technique

SMOTE technique addresses the imbalance in the dataset by oversampling minority classes to create a more balanced dataset. It reduces bias towards the majority class. Thus, the SMOTE technique helps address the difficulties created by imbalance data sets.

The usage of SMOTE is demonstrated as follows:



```
# SMOTE
# Regularization is applied by default in PyCaret's Logistic Regression.
# does training, lr stands for logistic regression
clf = setup(data=selected_features_df, target='diabetes', train_size=0.8, fix_imbalance=True)
model = create_model('lr')
# holdout_pred = predict_model(model)
```

	Description	Value
0	Session id	3570
1	Target	diabetes
2	Target type	Binary
3	Original data shape	(100000, 15)
4	Transformed data shape	(166400, 15)
5	Transformed train set shape	(146400, 15)
6	Transformed test set shape	(20000, 15)
7	Numeric features	6
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Fix imbalance	True
13	Fix imbalance method	SMOTE
14	Fold Generator	StratifiedKFold
15	Fold Number	10
16	CPU Jobs	-1
17	Use GPU	False
18	Log Experiment	False
19	Experiment Name	clf-default-name
20	USI	22ee

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Fold							
0	0.9438	0.9675	0.7868	0.6369	0.7039	0.6733	0.6778
1	0.9424	0.9632	0.7559	0.6354	0.6904	0.6589	0.6619
2	0.9445	0.9565	0.7456	0.6517	0.6955	0.6651	0.6669
3	0.9431	0.9588	0.7324	0.6459	0.6864	0.6553	0.6568
4	0.9424	0.9593	0.7559	0.6354	0.6904	0.6589	0.6619
5	0.9440	0.9582	0.7368	0.6506	0.6910	0.6604	0.6619
6	0.9416	0.9585	0.7485	0.6323	0.6855	0.6536	0.6564
7	0.9428	0.9607	0.7294	0.6442	0.6841	0.6528	0.6543
8	0.9442	0.9649	0.7721	0.6434	0.7019	0.6714	0.6748
9	0.9415	0.9586	0.7588	0.6293	0.6880	0.6560	0.6595
Mean	0.9430	0.9606	0.7522	0.6405	0.6917	0.6606	0.6632
Std	0.0010	0.0033	0.0169	0.0073	0.0064	0.0068	0.0074

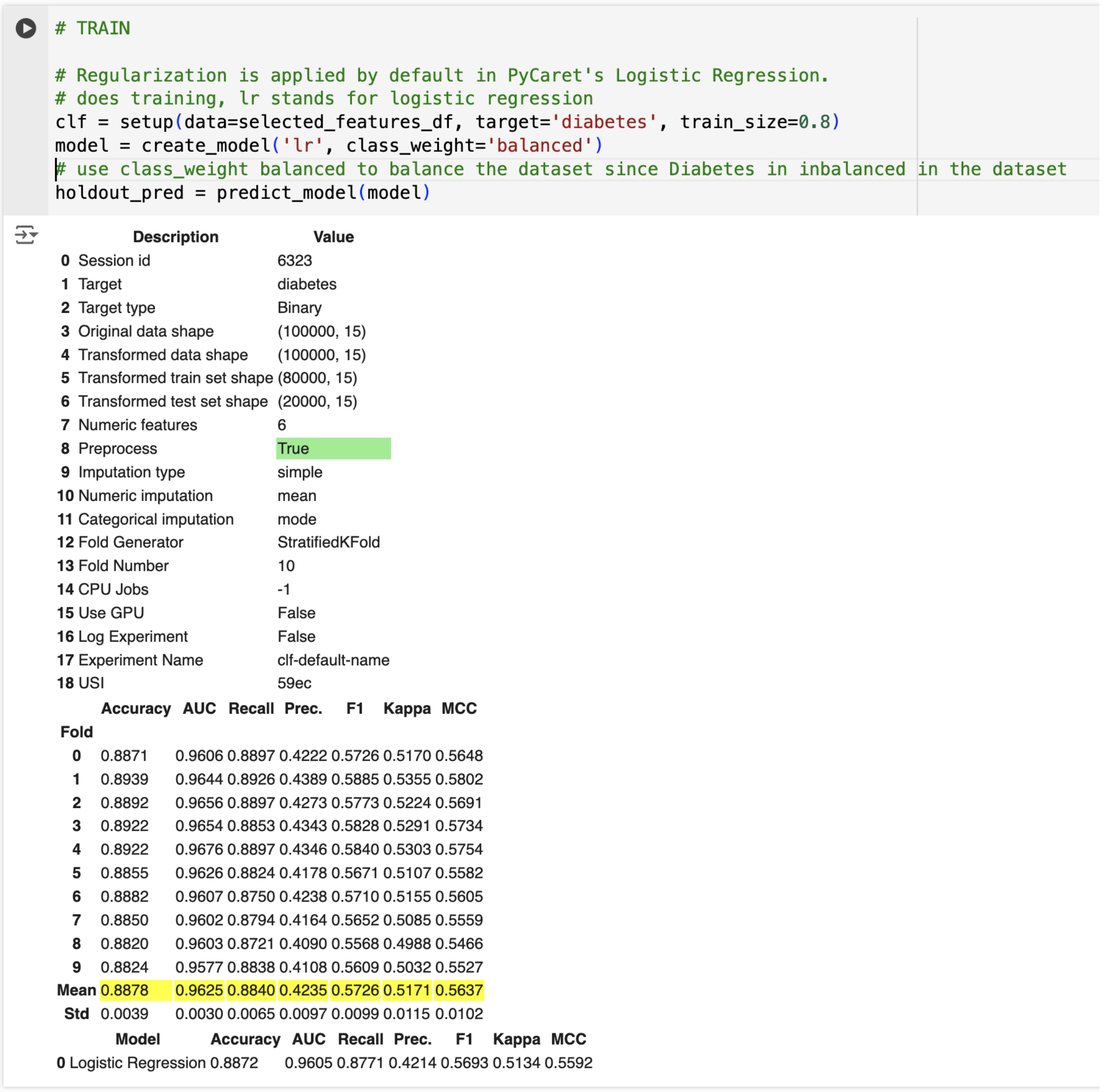
	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Logistic Regression	0.9424	0.9612	0.7406	0.6394	0.6863	0.6548	0.6569

As in the previous image, we can see that the Accuracy with SMOTE is 0.94, AUC is 0.96, Recall is 0.75 and Precision is 0.64.

Thus, using SMOTE to rebalance the dataset produces lower outcomes with Accuracy, Recall and Precision.

To redress the data imbalance, we can also try to train our data with the Class Weight Balancing. This is demonstrated below.

Applied Class Weight Balancing



```python
# TRAIN

# Regularization is applied by default in PyCaret's Logistic Regression.
# does training, lr stands for logistic regression
clf = setup(data=selected_features_df, target='diabetes', train_size=0.8)
model = create_model('lr', class_weight='balanced')
# use class_weight balanced to balance the dataset since Diabetes in inbalanced in the dataset
holdout_pred = predict_model(model)
```

```
   Description                 Value
 0 Session id                  6323
 1 Target                      diabetes
 2 Target type                 Binary
 3 Original data shape         (100000, 15)
 4 Transformed data shape      (100000, 15)
 5 Transformed train set shape (80000, 15)
 6 Transformed test set shape  (20000, 15)
 7 Numeric features            6
 8 Preprocess                  True
 9 Imputation type             simple
10 Numeric imputation          mean
11 Categorical imputation      mode
12 Fold Generator              StratifiedKFold
13 Fold Number                 10
14 CPU Jobs                    -1
15 Use GPU                     False
16 Log Experiment              False
17 Experiment Name             clf-default-name
18 USI                         59ec
```

```
      Accuracy  AUC    Recall Prec.  F1     Kappa  MCC
Fold
   0  0.8871    0.9606 0.8897 0.4222 0.5726 0.5170 0.5648
   1  0.8939    0.9644 0.8926 0.4389 0.5885 0.5355 0.5802
   2  0.8892    0.9656 0.8897 0.4273 0.5773 0.5224 0.5691
   3  0.8922    0.9654 0.8853 0.4343 0.5828 0.5291 0.5734
   4  0.8922    0.9676 0.8897 0.4346 0.5840 0.5303 0.5754
   5  0.8855    0.9626 0.8824 0.4178 0.5671 0.5107 0.5582
   6  0.8882    0.9607 0.8750 0.4238 0.5710 0.5155 0.5605
   7  0.8850    0.9602 0.8794 0.4164 0.5652 0.5085 0.5559
   8  0.8820    0.9603 0.8721 0.4090 0.5568 0.4988 0.5466
   9  0.8824    0.9577 0.8838 0.4108 0.5609 0.5032 0.5527
Mean  0.8878    0.9625 0.8840 0.4235 0.5726 0.5171 0.5637
 Std  0.0039    0.0030 0.0065 0.0097 0.0099 0.0115 0.0102
```

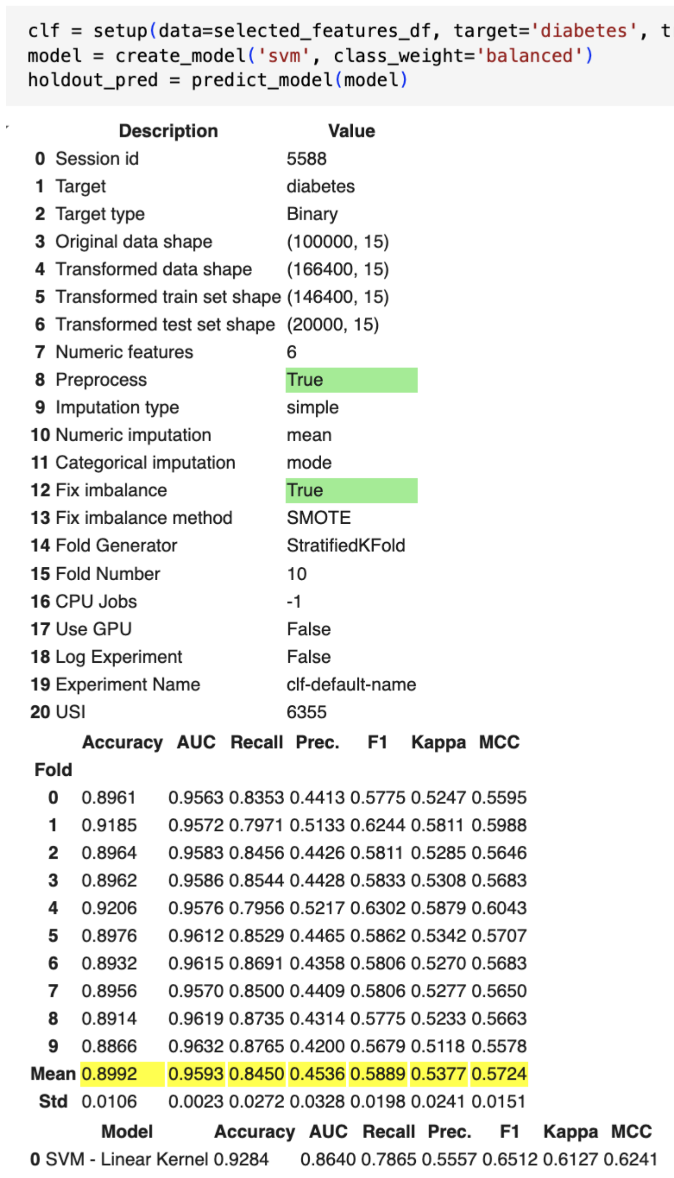
```
  Model               Accuracy  AUC    Recall Prec.  F1     Kappa  MCC
0 Logistic Regression 0.8872    0.9605 0.8771 0.4214 0.5693 0.5134 0.5592
```

Using the Class Weight Balance method we can see that the Accuracy is 0.88, AUC is 0.96, Recall is 0.88 and Precision is 0.42.

Generalising between the Class Weight and SMOTE rebalancing method for this dataset, we discern no significant difference in the outcomes. However, Class weight gave us better results than SMOTE when considering Recall performance metric, so we decided to continue with this Class Weight Balance method.

Support Vector Machine Model

With Logistic Regression as our base model with applied class weight balance, we applied same parameters for the SVM model to compare its performance.



```
clf = setup(data=selected_features_df, target='diabetes', train_size=0.8, fix_imbalance=True)
model = create_model('svm', class_weight='balanced')
holdout_pred = predict_model(model)
```

	Description	Value
0	Session id	5588
1	Target	diabetes
2	Target type	Binary
3	Original data shape	(100000, 15)
4	Transformed data shape	(166400, 15)
5	Transformed train set shape	(146400, 15)
6	Transformed test set shape	(20000, 15)
7	Numeric features	6
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Fix imbalance	True
13	Fix imbalance method	SMOTE
14	Fold Generator	StratifiedKFold
15	Fold Number	10
16	CPU Jobs	-1
17	Use GPU	False
18	Log Experiment	False
19	Experiment Name	clf-default-name
20	USI	6355

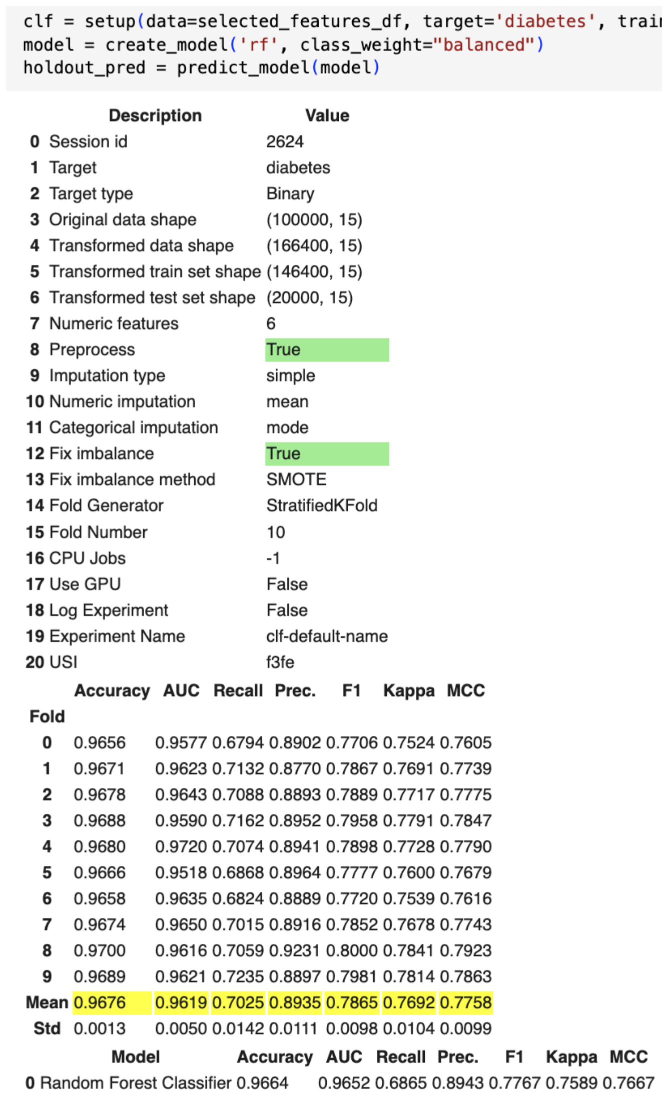
	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Fold							
0	0.8961	0.9563	0.8353	0.4413	0.5775	0.5247	0.5595
1	0.9185	0.9572	0.7971	0.5133	0.6244	0.5811	0.5988
2	0.8964	0.9583	0.8456	0.4426	0.5811	0.5285	0.5646
3	0.8962	0.9586	0.8544	0.4428	0.5833	0.5308	0.5683
4	0.9206	0.9576	0.7956	0.5217	0.6302	0.5879	0.6043
5	0.8976	0.9612	0.8529	0.4465	0.5862	0.5342	0.5707
6	0.8932	0.9615	0.8691	0.4358	0.5806	0.5270	0.5683
7	0.8956	0.9570	0.8500	0.4409	0.5806	0.5277	0.5650
8	0.8914	0.9619	0.8735	0.4314	0.5775	0.5233	0.5663
9	0.8866	0.9632	0.8765	0.4200	0.5679	0.5118	0.5578
Mean	0.8992	0.9593	0.8450	0.4536	0.5889	0.5377	0.5724
Std	0.0106	0.0023	0.0272	0.0328	0.0198	0.0241	0.0151

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	SVM - Linear Kernel	0.9284	0.8640	0.7865	0.5557	0.6512	0.6127	0.6241

Based on the results, we can see that the Accuracy is 0.89, AUC is 0.95, Recall is 0.84 and Precision is 0.45. Recall value has gone down while training the model using SVM.

Random Forest Model

Following are the performance metric for model trained using random forest algorithm:



```
clf = setup(data=selected_features_df, target='diabetes', train_size=0.8, fix_imbalance=True)
model = create_model('rf', class_weight="balanced")
holdout_pred = predict_model(model)
```

	Description	Value
0	Session id	2624
1	Target	diabetes
2	Target type	Binary
3	Original data shape	(100000, 15)
4	Transformed data shape	(166400, 15)
5	Transformed train set shape	(146400, 15)
6	Transformed test set shape	(20000, 15)
7	Numeric features	6
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Fix imbalance	True
13	Fix imbalance method	SMOTE
14	Fold Generator	StratifiedKFold
15	Fold Number	10
16	CPU Jobs	-1
17	Use GPU	False
18	Log Experiment	False
19	Experiment Name	clf-default-name
20	USI	f3fe

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Fold							
0	0.9656	0.9577	0.6794	0.8902	0.7706	0.7524	0.7605
1	0.9671	0.9623	0.7132	0.8770	0.7867	0.7691	0.7739
2	0.9678	0.9643	0.7088	0.8893	0.7889	0.7717	0.7775
3	0.9688	0.9590	0.7162	0.8952	0.7958	0.7791	0.7847
4	0.9680	0.9720	0.7074	0.8941	0.7898	0.7728	0.7790
5	0.9666	0.9518	0.6868	0.8964	0.7777	0.7600	0.7679
6	0.9658	0.9635	0.6824	0.8889	0.7720	0.7539	0.7616
7	0.9674	0.9650	0.7015	0.8916	0.7852	0.7678	0.7743
8	0.9700	0.9616	0.7059	0.9231	0.8000	0.7841	0.7923
9	0.9689	0.9621	0.7235	0.8897	0.7981	0.7814	0.7863
Mean	0.9676	0.9619	0.7025	0.8935	0.7865	0.7692	0.7758
Std	0.0013	0.0050	0.0142	0.0111	0.0098	0.0104	0.0099

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Random Forest Classifier	0.9664	0.9652	0.6865	0.8943	0.7767	0.7589	0.7667

We can see that the Accuracy is 0.96, AUC is 0.96, Recall is 0.70 and Precision is 0.89. Recall value has drastically reduced down to 0.70 when Random Forest algorithm was consumed for training the model.

Rationale for Performance Metric Choice

After reviewing the models, we have to decide which metric we will judge their performance on. Three primary metrics we will discuss include Recall, Precision and Accuracy. Different metrics are best applied to data with different characteristics and outcome objectives.

1. Recall (Sensitivity) is Likely Most Important:

- In a public health campaign, the primary objective is often to capture as many at-risk individuals as possible. Missing individuals who might have diabetes (false negatives) would mean they will not receive further information, testing, or follow-up. The goal is to maximize awareness and encourage testing or lifestyle changes in as many potentially affected individuals as possible.
- Public Health Focus: Even if some false positives (people without diabetes) are incorrectly identified, this might not be a huge concern because these individuals can simply be advised to monitor their health or receive further screening, which is generally less harmful than missing someone with potential risk.
- Best Metric: Recall should be the primary focus. We want to ensure that your campaign reaches and identifies as many individuals as possible who are at risk of diabetes, even if it means some individuals who are not at risk are also targeted.

2. Precision (Less Important in a Public Health Campaign):

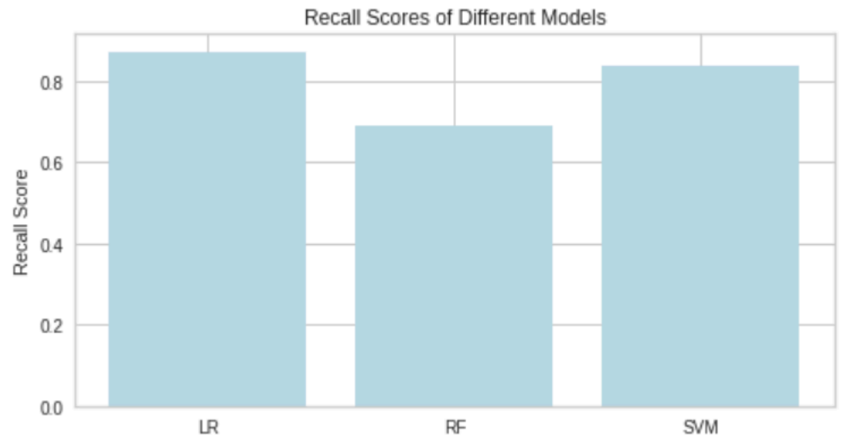
- Precision is less critical in this context because false positives (e.g., identifying someone as at risk of diabetes when they're not) may lead to minimal negative consequences. In public health, the cost of a false positive is often just increased engagement (e.g., suggesting a check-up or providing information), which is typically much less severe than the consequences of missing someone at risk.
- Public Health Focus: The downside of false positives is low compared to the benefits of identifying at-risk individuals who might otherwise go unnoticed.
- Metric Weight: Precision can be less emphasized here, as the goal is broad outreach rather than high accuracy of positive predictions.

3. Accuracy (Still Useful, but Less Emphasized):

- While accuracy gives a general measure of how often the campaign correctly identifies individuals, in an imbalanced setting (if the majority of the population does not have diabetes), accuracy can be misleading. The campaign's success should be judged primarily on its ability to capture at-risk individuals (recall), not just the overall correct predictions.
- Public Health Focus: Since false negatives are the primary concern (i.e., missing people who are at risk), accuracy alone wouldn't fully reflect the effectiveness of the campaign in terms of public health outcomes.
- Metric Weight: Accuracy is a secondary measure, as it doesn't fully capture the public health goal of reaching as many at-risk individuals as possible.

Recall as Primary Performance Metric

```
[0.8724, 0.6912, 0.8382]
```



Since this is a public health campaign, we want to capture as many at-risk individuals as possible. Our goal is to maximize awareness and encourage testing or lifestyle changes in as many potentially affected individuals as possible. Identifying individuals with positive conditions is of more importance than misclassifying those without the condition.

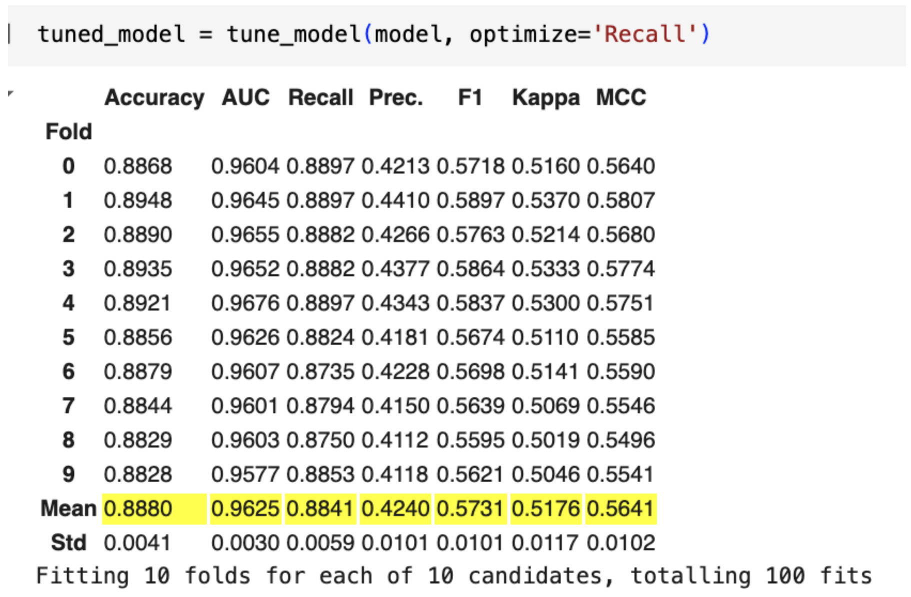
Since Recall measures the proportion of correctly predicted positive cases from all the actual positive cases, it ensures capturing the truly diabetic individuals. Therefore we chose the model with highest Recall performance, which is the logistic regression model as depicted in the comparison chart above.

Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction technique used to simplify large datasets by transforming them into fewer variables (called principal components) while retaining most of the original variance. PCA is not required in the case of this dataset as it is not a high dimensional dataset.

Tuning The Model

Since the Logistic Regression model performed the best for Recall being the primary performance metric, we chose it to tune the model and further optimize its performance for Recall.



```
tuned_model = tune_model(model, optimize='Recall')
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Fold							
0	0.8868	0.9604	0.8897	0.4213	0.5718	0.5160	0.5640
1	0.8948	0.9645	0.8897	0.4410	0.5897	0.5370	0.5807
2	0.8890	0.9655	0.8882	0.4266	0.5763	0.5214	0.5680
3	0.8935	0.9652	0.8882	0.4377	0.5864	0.5333	0.5774
4	0.8921	0.9676	0.8897	0.4343	0.5837	0.5300	0.5751
5	0.8856	0.9626	0.8824	0.4181	0.5674	0.5110	0.5585
6	0.8879	0.9607	0.8735	0.4228	0.5698	0.5141	0.5590
7	0.8844	0.9601	0.8794	0.4150	0.5639	0.5069	0.5546
8	0.8829	0.9603	0.8750	0.4112	0.5595	0.5019	0.5496
9	0.8828	0.9577	0.8853	0.4118	0.5621	0.5046	0.5541
Mean	0.8880	0.9625	0.8841	0.4240	0.5731	0.5176	0.5641
Std	0.0041	0.0030	0.0059	0.0101	0.0101	0.0117	0.0102

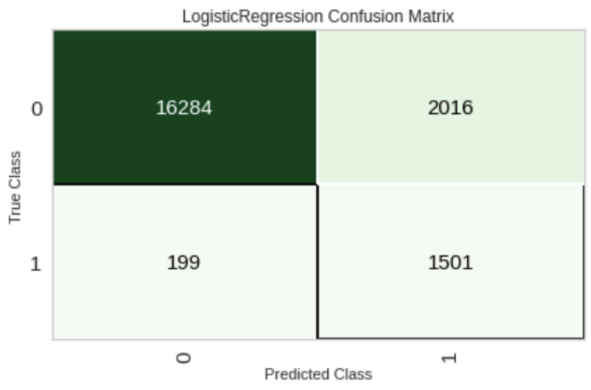
```
Fitting 10 folds for each of 10 candidates, totalling 100 fits
```

As depicted in the image above we can see that the tuned model for logistic regression does not have any major impact on the Recall score.

Model Evaluation

Validation curves can help identify whether the model is overfitting or underfitting.

Confusion matrix



True Negative: Number of people correctly predicted as not having diabetes. Count: 16 284	**False Positive:** Number of people incorrectly predicted as having diabetes when they don't. Count: 2016
False Negative: Number of people incorrectly predicted as not having diabetes when they do. Count: 199	**True Positive:** Number of people correctly predicted as having diabetes. Count: 1501

Often when tuning models, the engineer has to balance between Recall and Precision. In our case, with the large health impact of diabetes, and as previously discussed, we erred towards a high level of false positives and a lower level of false negatives.

In this, we were successful.

Diabetes is a potentially fatal condition. So, the implications of **not** being accurately flagged as having a high probability of developing diabetes when this is the case, exceeds the implications of **being** flagged as having a high probability of developing diabetes, but the individual does not. As evidenced above, while we had 2016 false positives, we had a much lower level of false negatives : 199.

This is in line with our desired approach.

Learning Curve

Observations

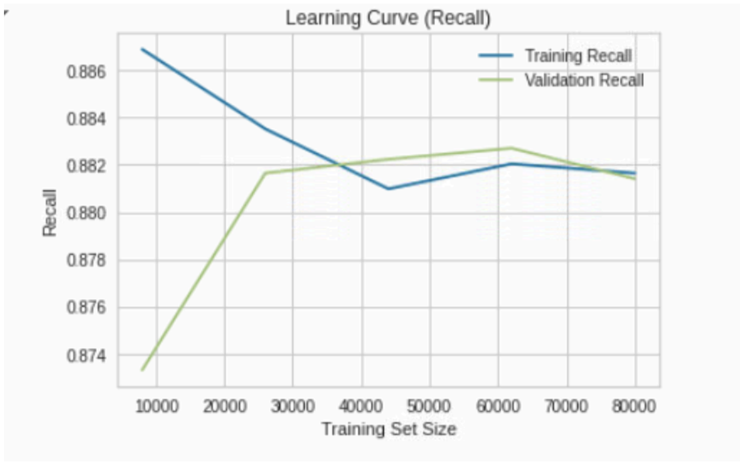
The observations from the learning curves are discussed here.

- **Initial Improvement:** As the training set size grows initially, both the training and validation recall scores increase. This is a good sign, as it means our model is learning from the data.

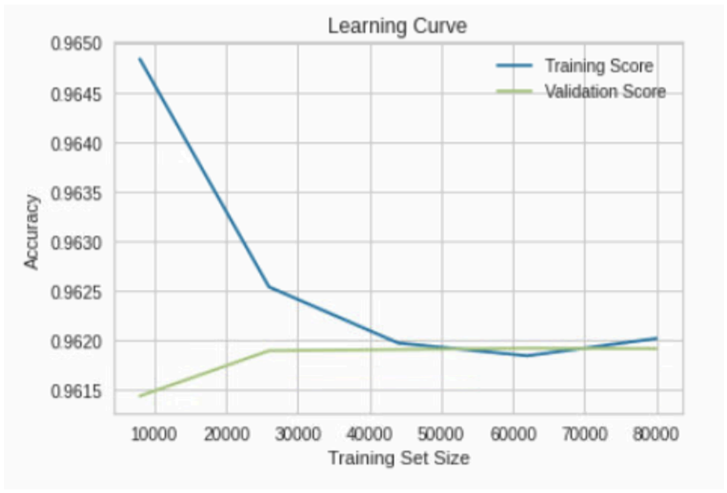
- **Plateauing:** After a certain training size, the validation recall might start to plateau or even slightly decrease. This suggests that adding more data might not significantly improve the recall further. It's possible we have reached a point of diminishing returns in terms of data collection.

- **Recall Gap:** The gap between the training and validation recall curves indicates a degree of overfitting, particularly with respect to identifying positive cases. It means the model might be memorizing some patterns in the training data that don't generalize to unseen examples.

As the training size increases, training declines and validation increases. At some point, the model may become over fit.



This Recall learning curve suggests that our logistic regression model is well-trained and generalizes appropriately. If we are looking for further improvement, we decided we might need to focus on model complexity, feature engineering, or alternative algorithms, as adding more data likely won't make a significant difference.



Summary

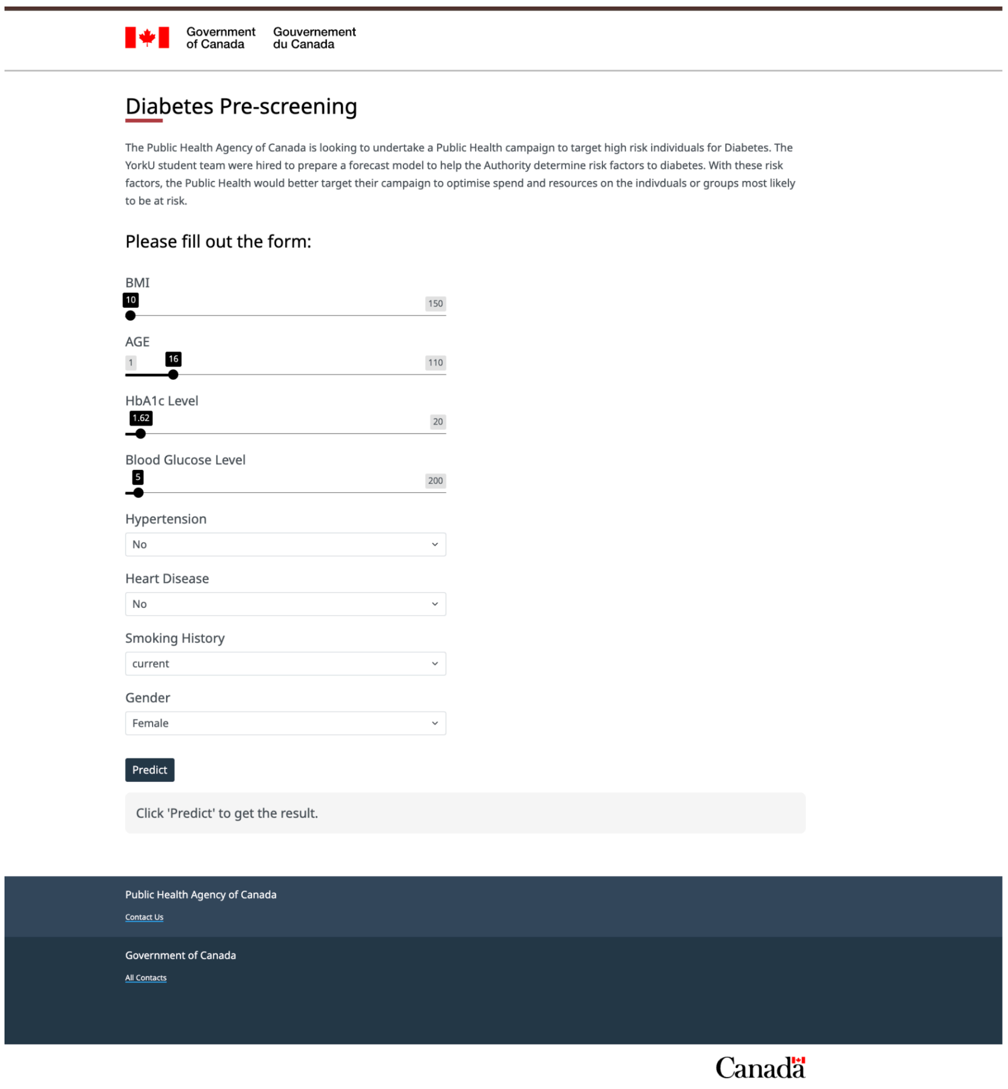
- The above model shows stable generalization, and the training and validation scores are converging, indicating low overfitting.
- The model's learning has plateaued, so we may want to consider tuning hyperparameters or using a different algorithm for better performance.
-

Model Deployment

The final chosen model will be deployed for real-time predictions, allowing the Public Health Authority to use the insights for targeting individuals in the diabetes prevention campaign.

The model was deployed using Shinyapps.io (www.shinyapps.io). Its url may be found here:

https://aabakarov.shinyapps.io/diabetes_prediction/



Government of Canada Gouvernement du Canada

Diabetes Pre-screening

The Public Health Agency of Canada is looking to undertake a Public Health campaign to target high risk individuals for Diabetes. The YorkU student team were hired to prepare a forecast model to help the Authority determine risk factors to diabetes. With these risk factors, the Public Health would better target their campaign to optimise spend and resources on the indivduals or groups most likely to be at risk.

Please fill out the form:

BMI: 10 (range to 150)

AGE: 16 (range 1 to 110)

HbA1c Level: 1.62 (range to 20)

Blood Glucose Level: 5 (range to 200)

Hypertension: No

Heart Disease: No

Smoking History: current

Gender: Female

Predict

Click 'Predict' to get the result.

Public Health Agency of Canada

Contact Us

Government of Canada

All Contacts

Canada

Its GitHub repo is public and may be found here:

https://github.com/ArslanAbakarov/diabetes_prediction/tree/master

This application is simple to understand and use by Health Care staff.

The link to the video presentation of the document and device may be found here

Conclusion

In a Public Health campaign where the goal is to raise awareness and identify potential at-risk individuals, the most important metric is Recall. We want to ensure that your campaign reaches the largest proportion of potentially affected individuals, even if it results in some false positives. Precision is less critical in this context, and accuracy, while useful, might not be the best reflection of our campaign's success.