Predictive Modeling of Diabetes Risk Using Health Behavior Indicators

- Name: Frank Ofosu
- Course: Clinical Decision Support Modeling
- Instructor: Prof. Guy Hembroff

Introduction

- Diabetes affects over 34 million Americans and is a leading cause of morbidity, mortality, and healthcare costs. This project develops an artificial intelligence-driven Clinical Decision Support (CDS) system that predicts diabetes risk using health behavior indicators from the CDC Behavioral Risk Factor Surveillance System (BRFSS 2015).

- **Key Achievement:** XGBoost model achieved 82% ROC-AUC and 86% accuracy, outperforming baseline logistic regression and Random Forest classifiers.

Problem Statement

- Diabetes affects millions of adults and remains a major cause of morbidity, mortality, and rising healthcare costs.
- Traditional screening methods often rely only on clinical or laboratory measures, missing key behavioral and lifestyle risk factors.
- As a result, many high-risk individuals remain undiagnosed or identified too late for effective prevention.
- There is a critical need for an accurate, interpretable, and scalable model that can predict diabetes risk using self-reported health behavior indicators.
- This project aims to address this gap by developing machine learning models evaluated on both balanced and real-world imbalanced datasets to support early detection and inform clinical decision-making

Literature Review

Research consistently demonstrates that **behavioral and lifestyle indicators** such as obesity, hypertension, physical inactivity, and poor nutrition are key predictors of diabetes.

- **Rahman et al. (2023)** reported improved predictive performance and clinician trust when combining ML models with explainable AI, using SHAP to show feature contributions.
- **Ullah & Lee (2024)** demonstrated that ensemble models such as Random Forest and XGBoost outperform traditional linear models in diabetes risk stratification.
- **Xie et al. (2019)** used BRFSS data to successfully develop accurate diabetes prediction models, validating its suitability for population-level analytics.
- **Wang et al. (2024)** highlighted the importance of optimized behavioral features for diabetes questionnaires and risk scoring systems.

Gaps This Project Addresses

Integration of *both* balanced and real-world imbalanced datasets.

Resampling (SMOTE) + model tuning.

Focus on interpretability for CDS use cases.

End-to-end framework suitable for healthcare deployment.

Dataset Description

BRFSS 2015 dataset (CDC) with 400,000+ responses

- Includes health behaviors, chronic conditions, and lifestyle indicators

Datasets Used:

- Balanced Binary Dataset (equal diabetes / no diabetes)
- Imbalanced Binary Dataset (real-world distribution)

Feature Categories:

- Clinical: BMI, HighBP, HighChol
- Behavioral: Smoking, Physical Activity, Diet
- Socioeconomic: Education, Income
- Self-Reported Health: General, Physical, Mental Health

- Features strongly correlate with diabetes risk

EDA

Datasets checked for missing values – none detected

- Balanced dataset: 50/50 diabetes distribution
- Imbalanced dataset reflects real-world majority class
- Class distribution

Key Variable Patterns:

- Higher BMI, poor General Health, HighBP, HighChol --higher diabetes risk
- Behavioral factors (diet, smoking, activity) show meaningful variation

Correlations:

- Strong positive: BMI, HighBP, HighChol
- Negative associations: Physical Activity, Fruit/Veg intake

```
# Split into train/test
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)
print(f"Train shape: {X_train.shape}, Test shape: {X_test.shape}")
```

```
Train shape: (202944, 21), Test shape: (50736, 21)
```

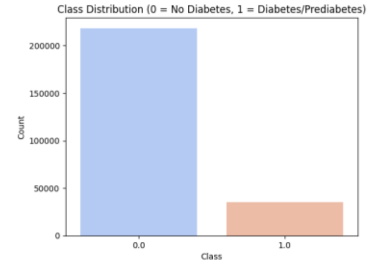
```
#Handle Imbalance

from imblearn.over_sampling import SMOTE

sm = SMOTE(random_state=42)
X_res, y_res = sm.fit_resample(X_train, y_train)

print("After SMOTE:", y_res.value_counts())
```

```
After SMOTE: Diabetes_binary
0    174667
1    174667
Name: count, dtype: int64
```

```
print(df.columns)
```

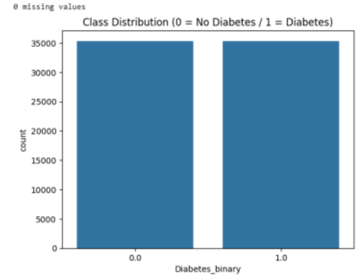
```
Index(['Diabetes_binary', 'HighBP', 'HighChol', 'CholCheck', 'BMI', 'Smoker',
       'Stroke', 'HeartDiseaseorAttack', 'PhysActivity', 'Fruits', 'Veggies',
       'HvyAlcoholConsump', 'AnyHealthcare', 'NoDocbcCost', 'GenHlth',
       'MentHlth', 'PhysHlth', 'DiffWalk', 'Sex', 'Age', 'Education',
       'Income'],
      dtype='object')
```

```
0 missing values
Class Distribution:
 Diabetes_binary
0.0    218334
1.0     35346
Name: count, dtype: int64

Percentage:
 Diabetes_binary
0.0    86.066698
1.0    13.933302
Name: proportion, dtype: float64
```




```
#Explore the dataset
print(df.isna().sum().sum(), "missing values")

# Summary stats
df.describe().T.head(10)

# Check class balance
sns.countplot(x='Diabetes_binary', data=df)
plt.title("Class Distribution (0 = No Diabetes / 1 = Diabetes)")
plt.show()
```

```
0 missing values
```



Methodology

Used machine learning to predict diabetes vs. no diabetes

- Loaded dataset and previewed to inspect structure and quality
- Cleaned and encoded data, then split into train/test sets
- Worked with balanced data and imbalanced data using SMOTE

Models Used:

- Logistic Regression
- Random Forest
- XGBoost

Evaluation:

- ROC-AUC, Accuracy, Precision, Recall, F1-score
- Hyperparameter tuning (GridSearchCV)
- SHAP explainability for feature impact

```
# Logistic Regression Model
logreg = LogisticRegression(max_iter=1000)
logreg.fit(X_train, y_train)
y_pred_logreg = logreg.predict(X_test)
y_prob_logreg = logreg.predict_proba(X_test)[:,1]

# Random Forest Classifier
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
y_prob_rf = rf.predict_proba(X_test)[:,1]

# XGBoost Classifier
xgb = XGBClassifier(eval_metric='logloss')
xgb.fit(X_train, y_train)
y_pred_xgb = xgb.predict(X_test)
y_prob_xgb = xgb.predict_proba(X_test)[:,1]
```

```
, y_test = train_test_split
shape, " Test:", X_test.sha
```

```
: (14139, 21)
```

```
#Handle Imbalance

from imblearn.over_sampling import SMOTE

sm = SMOTE(random_state=42)
X_res, y_res = sm.fit_resample(X_train, y_train)

print("After SMOTE:", y_res.value_counts())
```

```
After SMOTE: Diabetes_binary
0    174667
1    174667
Name: count, dtype: int64
```

```
e balanced binary dataset
ead_csv('/content/diabetes_binary_5050split_health_indicators_BRFSS2015.csv')
 data
ape:", df.shape)
f.head())
```

```
9692, 22)
```

es_binary	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	HeartDiseaseorAttack
0.0	1.0	0.0	1.0	26.0	0.0	0.0	0.0
0.0	1.0	1.0	1.0	26.0	1.0	1.0	0.0
0.0	0.0	0.0	1.0	26.0	0.0	0.0	0.0
0.0	1.0	1.0	1.0	28.0	1.0	0.0	0.0
0.0	0.0	0.0	1.0	29.0	1.0	0.0	0.0

columns

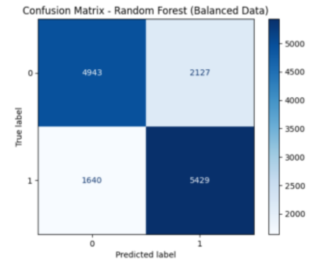
Balanced Dataset Results

- LR: Strong baseline performance
- RF: Better recall for diabetic class
- XGBoost: Best overall AUC and F1

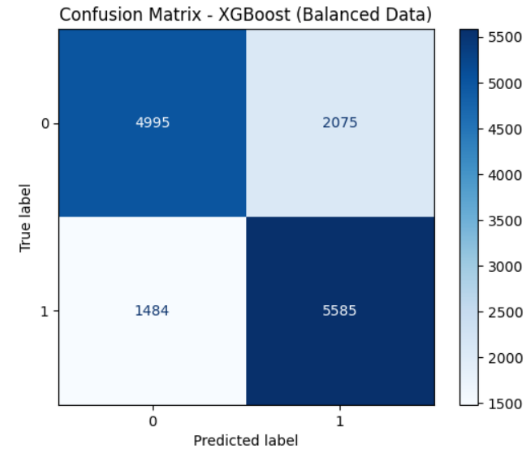
Feature Importance

- Both models agree that health-related features (BMI, General Health, Physical Health, Blood Pressure, Cholesterol) are important.
- Random Forest spreads importance across more features, while XGBoost puts more weight on a few top predictors

```
Random Forest (Balanced Dataset)

Random Forest Performance on BALANCED Test Set
--------------------------------------------------
Accuracy : 0.7335738029563619
Precision: 0.7185018528321864
Recall   : 0.7680011317017966
F1 Score : 0.7424273504273504
AUROC    : 0.8097282935253491
```



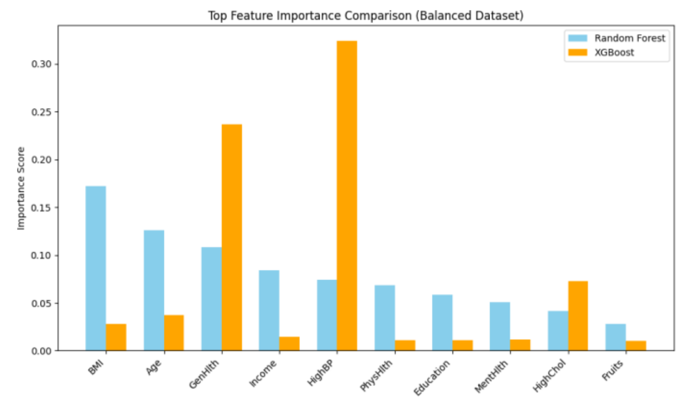

```
XGBoost (Balanced Dataset)

XGBoost Performance on BALANCED Test Set
--------------------------------------------------
Accuracy : 0.7482848857769291
Precision: 0.72911227154047
Recall   : 0.7900693167350403
F1 Score : 0.7583678457464865
AUROC    : 0.8247269839446812
```



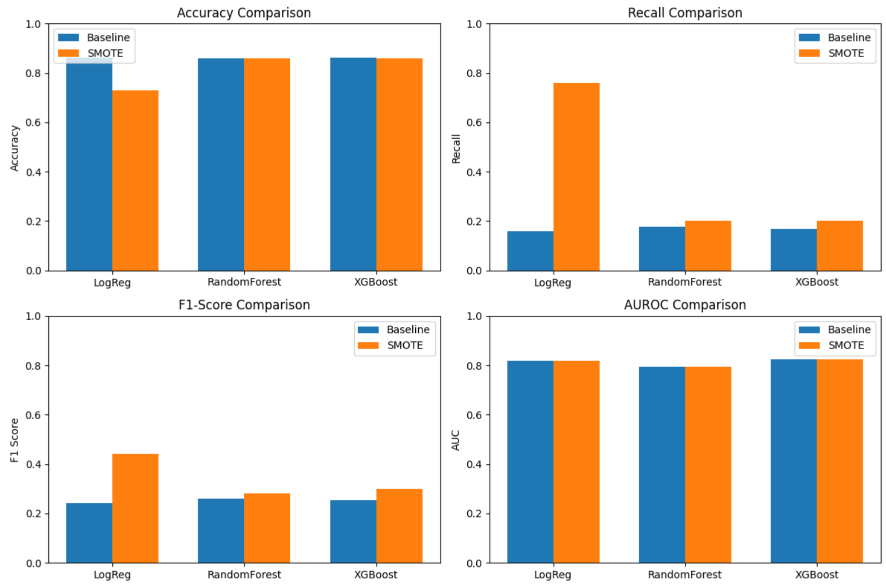

```
Logistic Regression (Balanced Dataset)

Logistic Regression Performance on BALANCED Test Set
--------------------------------------------------
Accuracy : 0.7458094631869298
Precision: 0.7371366179882626
Recall   : 0.7640401754137784
F1 Score : 0.7503473186996388
AUROC    : 0.82322481788425
```

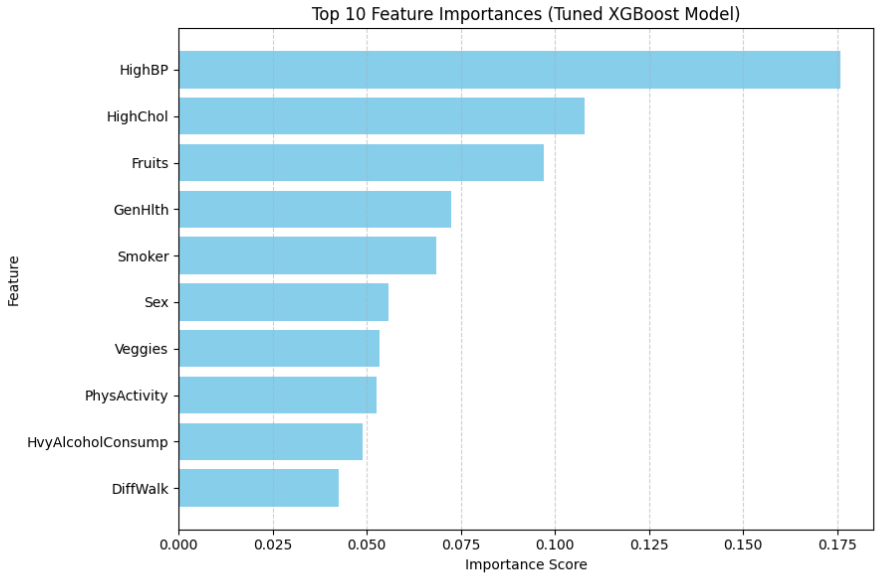

```
# Evaluation metrics
def print_metrics(y_true, y_pred, y_prob):
    print('Accuracy:', accuracy_score(y_true, y_pred))
    print('Precision:', precision_score(y_true, y_pred))
    print('Recall:', recall_score(y_true, y_pred))
    print('F1 Score:', f1_score(y_true, y_pred))
    print('AUROC:', roc_auc_score(y_true, y_prob))

print("Logistic Regression Performance:")
print_metrics(y_test, y_pred_logreg, y_prob_logreg)

print("\nRandom Forest Performance:")
print_metrics(y_test, y_pred_rf, y_prob_rf)

print("\nXGBoost Performance:")
print_metrics(y_test, y_pred_xgb, y_prob_xgb)
```

```
Logistic Regression Performance:
Accuracy: 0.7458094631869298
Precision: 0.7371366179882626
Recall: 0.7640401754137784
F1 Score: 0.7503473186996388
AUROC: 0.82322481788425

Random Forest Performance:
Accuracy: 0.7335738029563619
Precision: 0.7185018528321864
Recall: 0.7680011317017966
F1 Score: 0.7424273504273504
AUROC: 0.8097282935253491

XGBoost Performance:
Accuracy: 0.7482848857769291
Precision: 0.72911227154047
Recall: 0.7900693167350403
F1 Score: 0.7583678457464865
AUROC: 0.8247269839446812
```



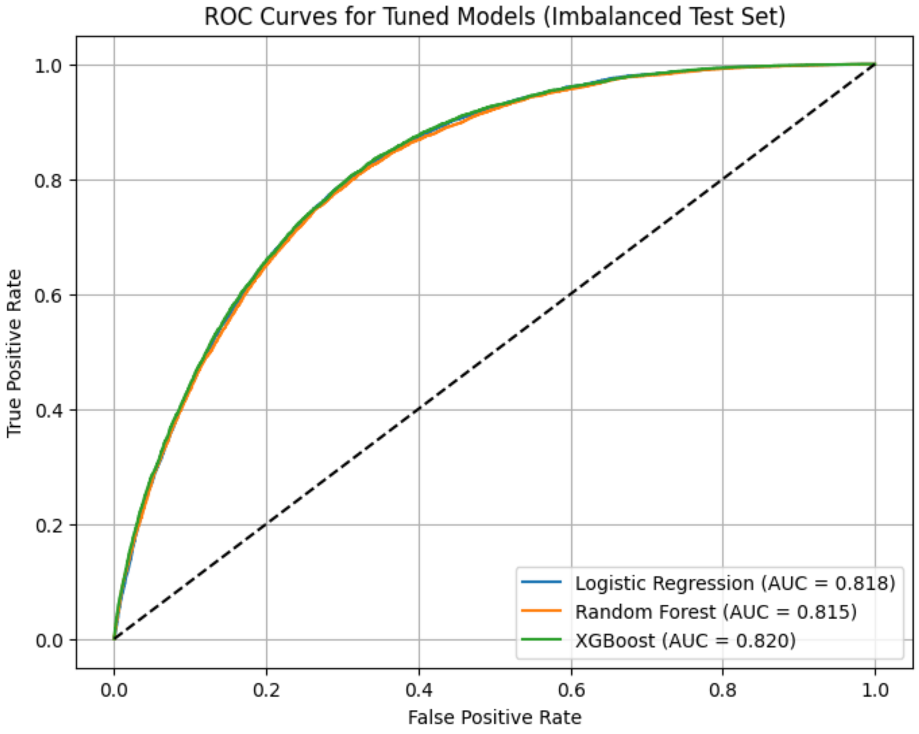
Imbalanced Dataset Results





ROC Curve Comparison

- **XGBoost (AUC = 0.820)** *Best generalization to real-world data.*
- **Logistic Regression (AUC = 0.818)** Very stable, consistent performance.
- **Random Forest (AUC = 0.815)** Slight decline despite strong CV results (overfitting on SMOTE data).



Hyperparameter Tuning

- GridSearchCV used for LR, RF, XGBoost
- Improved calibration
- Enhanced model stability
- Better discrimination across classes
- Random Forest achieved 97%

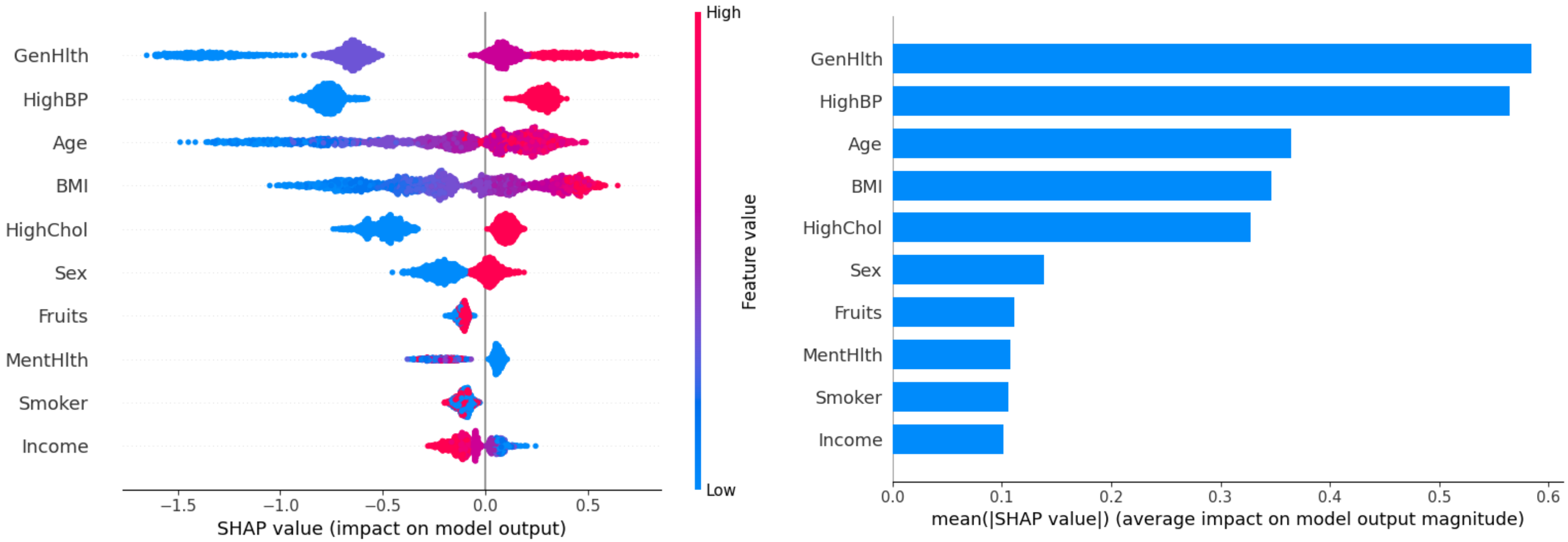
```python
# Logistic Regression Tuning
print("\n Tuning Logistic Regression")
lr_param_grid = {
    'C': [0.01, 0.1, 1, 10, 100],
    'solver': ['liblinear', 'saga']
}
lr_grid_search = GridSearchCV(
    LogisticRegression(max_iter=2000, random_state=42),
    lr_param_grid,
    cv=5,
    scoring='roc_auc',
    n_jobs=-1, verbose=1
)
lr_grid_search.fit(X_res, y_res)

best_lr_smote_tuned = lr_grid_search.best_estimator_
print("Best parameters for Logistic Regression:", lr_grid_search.best_params_)
print("Best cross-validation ROC-AUC for Logistic Regression:", lr_grid_search.best_score_)

# Random Forest Tunining
print("\n Tuning Random Forest")
rf_param_grid = {
    'n_estimators': [100, 200],
    'max_depth': [10, 20],
    'min_samples_split': [2, 5]
}
rf_grid_search = GridSearchCV(
    RandomForestClassifier(random_state=42),
    rf_param_grid,
    cv=5,
```

```
 Tuning Logistic Regression
Fitting 5 folds for each of 10 candidates, totalling 50 fits
Best parameters for Logistic Regression: {'C': 0.01, 'solver': 'saga'}
Best cross-validation ROC-AUC for Logistic Regression: 0.8304776740603959

 Tuning Random Forest
Fitting 5 folds for each of 8 candidates, totalling 40 fits
Best parameters for Random Forest: {'max_depth': 20, 'min_samples_split': 2, 'n_estimators': 200}
Best cross-validation ROC-AUC for Random Forest: 0.9708121103639625

 Tuning XGBoost
Fitting 5 folds for each of 32 candidates, totalling 160 fits
Best parameters for XGBoost: {'colsample_bytree': 0.7, 'learning_rate': 0.05, 'max_depth': 5, 'n_estimators': 200, 'subsample': 0.7}
Best cross-validation ROC-AUC for XGBoost: 0.9600525212024541
```

Model	Best Hyperparameters	Best CV ROC-AUC	Notes
Logistic Regression	C = 0.01, solver = 'saga'	**0.8305**	Performs well but limited by linear decision boundary
Random Forest	n_estimators = 200, max_depth = 20, min_samples_split = 2	**0.9708**	Best-performing tuned model
XGBoost	learning_rate = 0.05, max_depth = 5, n_estimators = 200, subsample = 0.7, colsample_bytree = 0.7	**0.9601**	Excellent performance, competitive with RF

Model Interpretability (SHAP)

- SHAP reveals how each feature influences the model's prediction of diabetes risk.
- Positive SHAP values → increase predicted risk; negative values → decrease predicted risk.
- Colors indicate feature values: **red = high**, **blue = low**.



Conclusion

- Machine learning methods can reliably predict diabetes risk using self-reported behavioral and clinical indicators. SHAP explainability confirms that the model aligns with clinical knowledge, making it suitable for future development into an interpretable clinical decision support tool.

Future Work

Test advanced models (LightGBM, CatBoost) and calibrated probability outputs.

Incorporate additional clinical or behavioral data for improved accuracy.

Explore alternative imbalance strategies and fairness auditing.

Build a deployable CDS tool with interactive SHAP explanations.

Validate predictions with clinicians and real-world patient data.

THANK YOU

References

- Centers for Disease Control and Prevention. (2015). Behavioral Risk Factor Surveillance System (BRFSS) annual data. U.S. Department of Health & Human Services. https://www.cdc.gov/brfss/annual_data/annual_data.htm
- Rahman, S. S., Khatun, F., & Rahman, M. M. (2023). Diabetes prediction using machine learning and explainable AI. Journal of Healthcare Engineering, 2023, 1–12. https://pmc.ncbi.nlm.nih.gov/articles/PMC10107388/
- Ullah, I., & Lee, Y. (2024). Robust diabetic prediction using ensemble machine learning. Scientific Reports, 14, 78519. https://doi.org/10.1038/s41598-024-78519-8
- Xie, Z., Nikolayeva, O., Luo, J., & Li, D. (2019). Building risk prediction models for type 2 diabetes using 2014 BRFSS data. Preventing Chronic Disease, 16, E74. https://doi.org/10.5888/pcd16.190109
- Wang, X., Li, H., & Zhang, Y. (2024). A feature optimization study based on a diabetes risk questionnaire. Frontiers in Public Health, 12, 1328353. https://doi.org/10.3389/fpubh.2024.1328353