Diabetes Risk Analysis and Population Segmentation

Using CDC Diabetes Health Indicators Survey Data

Alexander Young
University of Warwick (WMG): MSc Applied AI

March 8, 2026

GitHub Repository:
`https://github.com/Ayoung-14/Diabetes-Risk-Analysis-and-Population-Segmentation`

Abstract

This report summarizes a public health analytics project that (i) estimates diabetes risk using supervised learning and (ii) identifies population risk segments using unsupervised learning. Results are presented with an emphasis on interpretability, methodological rigor, and actionable public health insights.

Contents

1 Executive Summary

Key findings. Models separate *diabetes* from *no diabetes* well, but *prediabetes* remains the hardest class because its feature profile overlaps with both neighboring groups. Across EDA, association rules, clustering (K-Means, DBSCAN, Hierarchical), and supervised learning, elevated risk is most consistently linked to poorer self-rated general health (`GenHlth`), older age, cardiometabolic comorbidities (notably `HighBP` and `HighChol`), and higher BMI. Unsupervised segments reflect distinct burden/vulnerability profiles, and post-hoc checks show meaningful differences in diabetes prevalence across clusters.

Practical implications. Outputs are intended for *screening and triage* rather than diagnosis. Calibrated probabilities and decision thresholds can prioritize follow-up testing and outreach, while clusters and co-occurrence patterns can inform segment-specific interventions (clinical risk management vs. prevention and access support). A two-stage workflow (at-risk screening, then diabetes vs. prediabetes stratification) aligns evaluation with public health objectives, and explainability methods (coefficients, permutation importance, SHAP, LIME) support transparency.

Limitations and uncertainty. Prediabetes is intrinsically ambiguous in this survey feature space and is therefore more prone to misclassification near class boundaries. Results also inherit limitations of self-reported, cross-sectional survey data, so predictions should be treated as risk signals to guide resource allocation and confirmatory clinical assessment rather than as diagnostic decisions.

2 Business Understanding

The objective is to help a public health research institute understand and address diabetes risk in the adult population using national survey data.

Two complementary analytical goals guide the work:

1. **Predict diabetes risk (classification).** Use supervised learning to rank individuals by risk using demographic, lifestyle, and self-reported health indicators. Predicted probabilities are interpreted as *screening signals* to support triage and follow-up testing, not as diagnosis.

2. **Identify risk segments (clustering).** Use unsupervised learning to group respondents into descriptive profiles based on shared behaviors and risk-factor patterns. These clusters are not disease labels; they support targeted, segment-specific prevention and outreach planning.

Together, these approaches provide individual-level risk stratification and population-level insight to support evidence-based public health planning.

3 Data Understanding

3.1 Dataset Overview

The CDC Diabetes Health Indicators dataset contains responses from over 250 000 adults in a national health survey. Predictors capture demographics, health behaviors, and self-reported health status. The target variable encodes diabetes status: no diabetes, prediabetes, and diabetes.

3.2 Data Loading & Initial Inspection

The dataset was loaded and checked for basic integrity (dimensions, numeric encodings, and target mapping) before analysis. Although provided in analysis-ready form with no missing values, exact duplicate rows were identified and removed to avoid inflating the effective sample size and biasing downstream evaluation (particularly under stratified train/validation/test splitting). Summary statistics and range checks were also reviewed to confirm that key variables fall within expected survey bounds.

```
Shape Before Dropping Duplicates: (253680, 22)
Shape After Dropping Duplicates: (229781, 22)

 Index(['Diabetes_012', 'HighBP', 'HighChol', 'CholCheck', 'BMI', 'Smoker',
       'Stroke', 'HeartDiseaseorAttack', 'PhysActivity', 'Fruits', 'Veggies',
       'HvyAlcoholConsump', 'AnyHealthcare', 'NoDocbcCost', 'GenHlth',
       'MentHlth', 'PhysHlth', 'DiffWalk', 'Sex', 'Age', 'Education',
       'Income'],
      dtype='str')
```

	Diabetes_012	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	HeartDiseaseorAttack	PhysActivity	Fruits	...	AnyHealthcare	NoDocbcCost	GenHlth	MentHlth	PhysHlth	DiffWalk	Sex	Age	Education	Income
0	0.0	1.0	1.0	1.0	40.0	1.0	0.0	0.0	0.0	0.0	...	1.0	0.0	5.0	18.0	15.0	1.0	0.0	9.0	4.0	3.0
1	0.0	0.0	0.0	0.0	25.0	1.0	0.0	0.0	1.0	0.0	...	0.0	1.0	3.0	0.0	0.0	0.0	0.0	7.0	6.0	1.0
2	0.0	1.0	1.0	1.0	28.0	0.0	0.0	0.0	0.0	1.0	...	1.0	1.0	5.0	30.0	30.0	1.0	0.0	9.0	4.0	8.0
3	0.0	1.0	0.0	1.0	27.0	0.0	0.0	0.0	1.0	1.0	...	1.0	0.0	2.0	0.0	0.0	0.0	0.0	11.0	3.0	6.0
4	0.0	1.0	1.0	1.0	24.0	0.0	0.0	0.0	1.0	1.0	...	1.0	0.0	2.0	3.0	0.0	0.0	0.0	11.0	5.0	4.0

5 rows × 22 columns

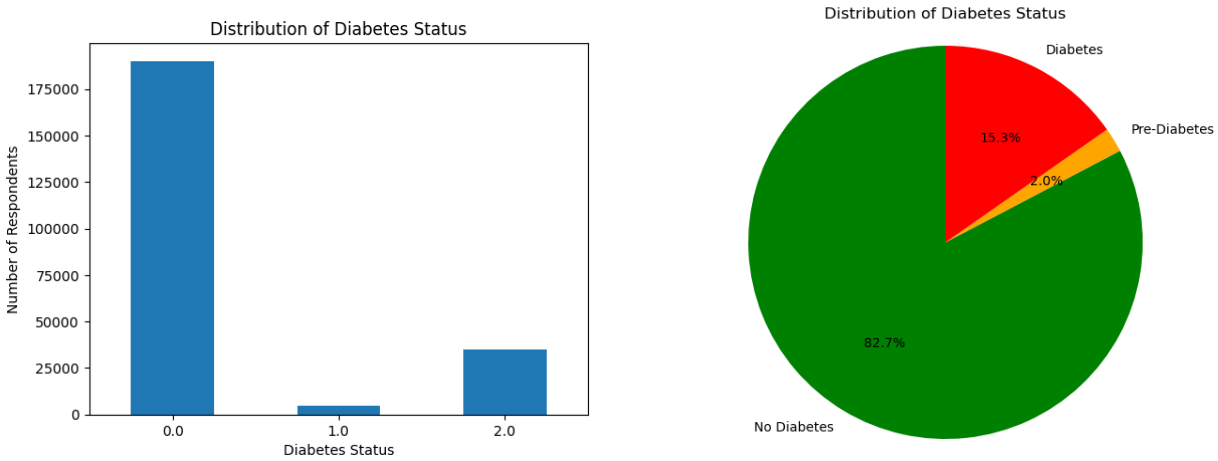
Figure 1: Initial data integrity checks and deduplication summary.

3.3 Target Variable and Class Imbalance

The outcome variable is `Diabetes_012`, a three-class label encoding self-reported diabetes status:

- 0 = No diabetes (or diabetes only during pregnancy)
- 1 = Prediabetes
- 2 = Diabetes

After removing exact duplicate rows, the working dataset contains 229 781 respondents. The class distribution is strongly imbalanced: 190 055 in class 0 (no diabetes), 4629 in class 1 (prediabetes), and 35 097 in class 2 (diabetes), with prediabetes comprising only about 2% of observations. Several predictor variables are also unevenly distributed (many are binary/low-prevalence indicators), so both the target and parts of the feature space are skewed toward the most common outcomes and responses. As a result, modeling decisions should explicitly account for imbalance to avoid solutions that primarily reflect majority patterns.



(a) Class counts (bar chart).

(b) Class proportions (pie chart).

Figure 2: Distribution of `Diabetes_012` in the deduplicated dataset.

3.4 Feature Groups and Variable Definitions

The working dataset contains 21 predictor variables derived from a national health survey. All predictors are already numerically encoded (binary indicators or ordinal categories), with BMI as a continuous measure and `MentHlth`/`PhysHlth` as bounded count variables (0–30 days). Following the initial integrity checks, all features were retained as provided (no missing values were observed and types were consistently numeric), and the final target encoding for `Diabetes_012` remains 0 = no diabetes (or diabetes only during pregnancy), 1 = prediabetes, and 2 = diabetes.

For interpretability, features are grouped into three broad categories:

- **Demographics (background context):** `Age` (ordinal age group), `Sex` (binary), `Education` (ordinal 1–6), and `Income` (ordinal 1–8). These variables describe population subgroups and socioeconomic context.
- **Lifestyle/behaviors (modifiable factors):** `PhysActivity`, `Smoker`, `HvyAlcoholConsump`, `Fruits`, and `Veggies` (all binary). These capture reported health behaviors relevant to prevention and intervention.
- **Health status, medical history, and access (risk burden):** `BMI` (continuous); health history: `HighBP`, `HighChol`, `CholCheck`, `Stroke`, and `HeartDiseaseorAttack` (binary); self-rated/burden: `GenHlth` (ordinal 1–5), plus `MentHlth` and `PhysHlth` (0–30 days); function: `DiffWalk` (binary); access: `AnyHealthcare` and `NoDocbcCost` (binary). Together, these features reflect risk, perceived health burden, and barriers to care.

These groupings are used throughout the EDA and modeling sections to support coherent interpretation (e.g., separating potentially modifiable behaviors from clinical burden and access-to-care proxies) while keeping the prediction task aligned with the project objective of risk screening rather than clinical diagnosis.

4 Exploratory Data Analysis (EDA)

This section summarizes key patterns in the data that inform feature engineering, modeling choices, and public health interpretation. The analysis focuses on (i) baseline distributions and prevalence rates, (ii) how diabetes burden varies across age, BMI, and socioeconomic status, and (iii) which behavioral and health-indicator features show the clearest separation between outcome groups. Throughout, results are treated as descriptive (not causal) given the cross-sectional, self-reported survey design, and emphasis is placed on patterns that translate into actionable screening and outreach priorities.

4.1 Summary Statistics

To establish a baseline for EDA and modeling, summary statistics were computed for the deduplicated dataset. All variables are numerically encoded and contain no missing values, so summaries were computed directly. For binary variables, the mean corresponds to prevalence; for ordinal scales, summaries reflect the distribution over coded categories; and for continuous or bounded-count variables, percentiles are reported alongside mean and standard deviation.

Table 1: Summary Statistics for key variables (deduplicated dataset; $n = 229\,781$).

Variable	Mean	SD	Min	P25	Median	P75	Max
Diabetes_012	0.326	0.725	0	0	0	0	2
BMI	28.686	6.786	12	24	27	32	98
GenHlth	2.601	1.065	1	2	3	3	5
Age	8.087	3.094	1	6	8	10	13
HighBP	0.454	0.498	0	0	0	1	1
HighChol	0.442	0.497	0	0	0	1	1
HeartDiseaseorAttack	0.103	0.304	0	0	0	0	1
Stroke	0.045	0.207	0	0	0	0	1
PhysActivity	0.733	0.442	0	0	1	1	1
Smoker	0.466	0.499	0	0	0	1	1
NoDocbcCost	0.093	0.290	0	0	0	0	1

4.2 Feature Distributions

To understand variable scale and shape before modeling, distributions were examined selectively by feature type. Many predictors are binary (0/1), while `GenHlth` and `Age` are discrete ordinal scales. Continuous `BMI` shows wider spread, and bounded-count measures (`MentHlth`, `PhysHlth`; 0–30 days) are strongly skewed and zero-inflated, motivating interpretation via percentiles and high-burden thresholds rather than means alone.

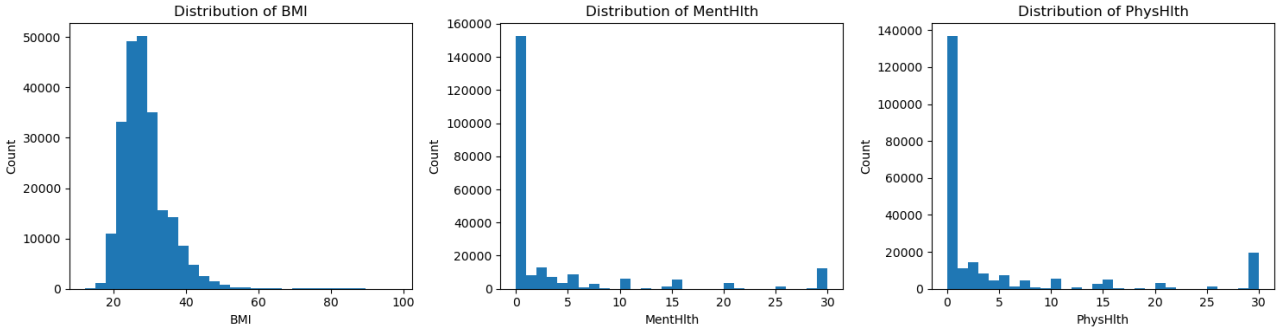
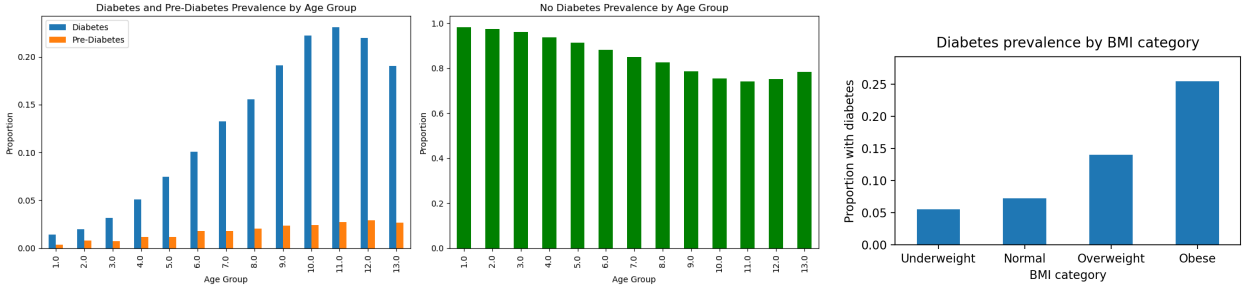


Figure 3: Selected feature distributions: `BMI`, `MentHlth`, and `PhysHlth`.

4.3 Diabetes Prevalence by Age and BMI

Diabetes prevalence increases with both age and BMI. In Figure 4, panel (a) shows prevalence across age groups (with the complementary no-diabetes pattern), and panel (b) summarizes prevalence across standard BMI categories. For context, respondent counts and diabetes case counts are also provided in Figures 57 and 58 (Appendix Section A). Together, these views highlight where prevalence is concentrated, supporting screening prioritization and targeted prevention.



(a) Prevalence by age group (and complementary no-diabetes pattern). (b) Prevalence across BMI categories.

Figure 4: Diabetes prevalence patterns by age and BMI.

4.4 Demographics vs. Diabetes

Diabetes prevalence varies across demographics. The strongest gradient is by age, with higher-coded `Age` groups showing substantially higher prevalence. Differences are also observed across `Income` and `Education`, consistent with socioeconomic context shaping baseline health and access to prevention and care. These results are descriptive (not causal) and may reflect confounding (e.g., age and burden correlating with socioeconomic position), but they still help identify higher-burden subpopulations for targeted screening and outreach.

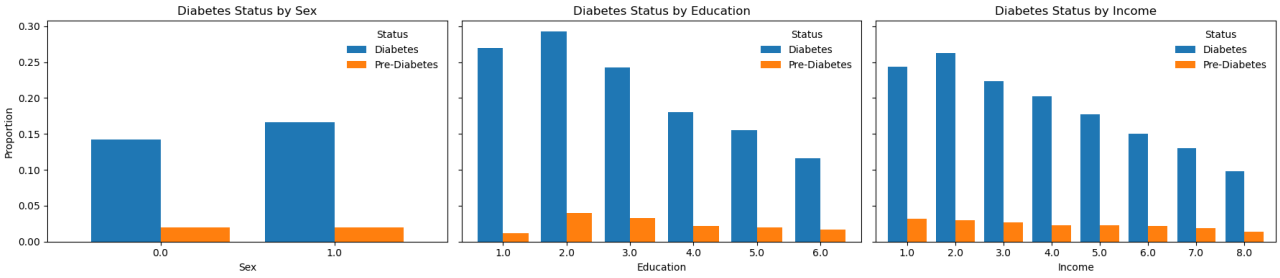


Figure 5: Diabetes prevalence across demographic and socioeconomic groups.

4.5 Lifestyle Factors vs. Diabetes

Lifestyle and behavioral variables differ across diabetes groups, though results are descriptive given the cross-sectional, self-reported survey design. Overall, respondents with diabetes show a less favorable behavioral profile (notably lower reported physical activity and differences in diet proxies). Smoking remains common across all groups, indicating a prevention opportunity that extends beyond diabetes status.

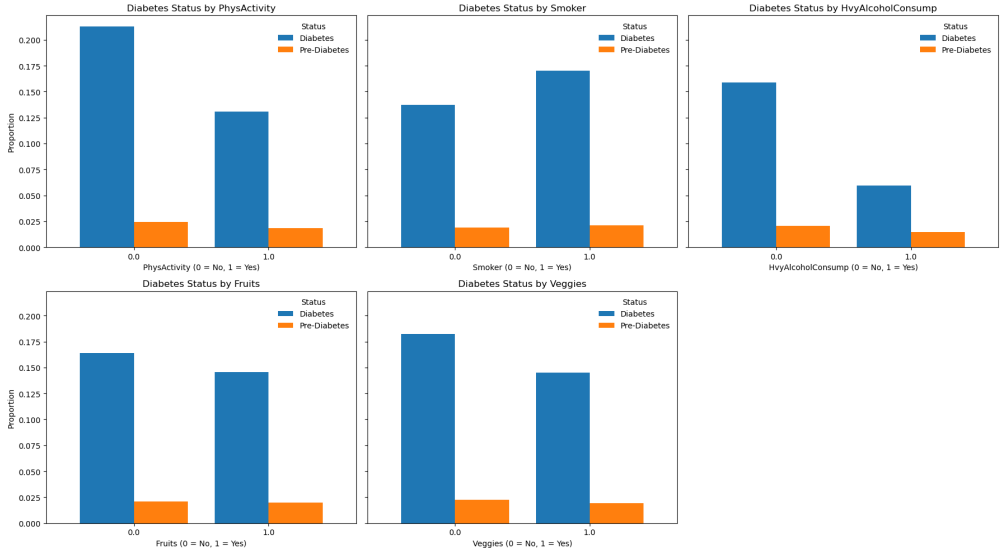


Figure 6: Lifestyle and behavioral indicators by diabetes status.

Effect sizes (differences in proportions) are more informative here than visual separation alone. For screening and intervention, these variables are best interpreted as modifiable levers that complement clinical risk management, with smoking and alcohol indicators helping flag groups that may benefit from targeted counseling.

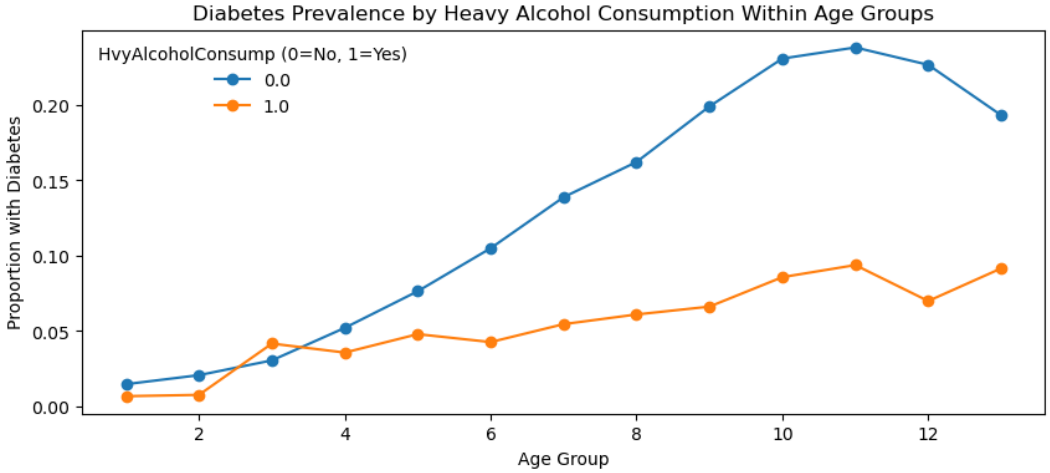


Figure 7: Lifestyle and behavioral indicators by diabetes status (II).

4.6 Health Indicators vs. Diabetes

Health-status indicators show the clearest separation across outcome groups. Respondents with diabetes have higher prevalence of negative health indicators (notably `HighBP` and `HighChol`) and related history markers (`HeartDiseaseorAttack`, `Stroke`), alongside worse self-rated health (`GenHlth`) and more functional limitation (`DiffWalk`), reflecting cumulative burden rather than modifiable behavior alone.

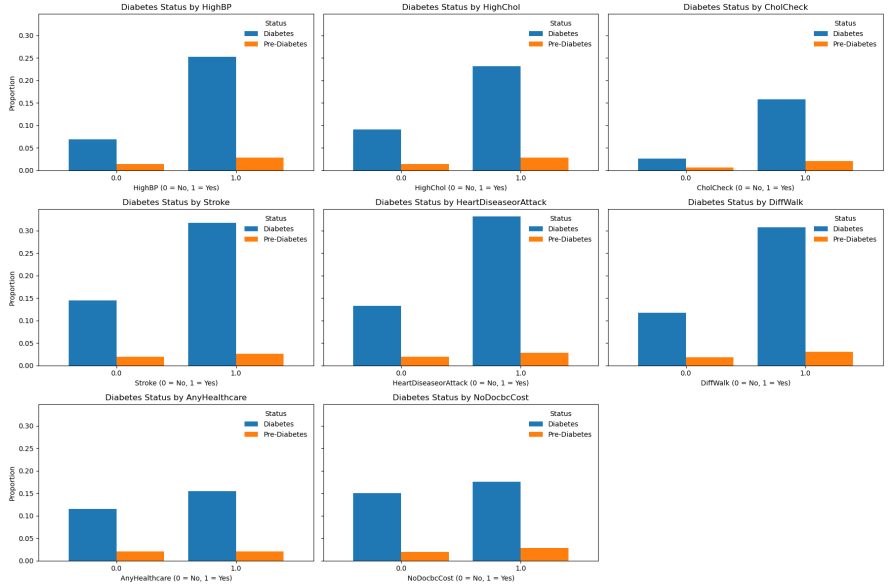


Figure 8: Comorbidity and self-reported health indicators by diabetes status (I).

Bounded-count measures (`MentHlth`, `PhysHlth`) also shift upward for respondents with diabetes. Because these variables are bounded (0–30) and typically zero-inflated, interpretation is better supported by percentiles and a high-burden threshold (e.g., ≥ 14 days) than by means alone. These features add screening signal, but likely overlap with other correlated burden indicators.

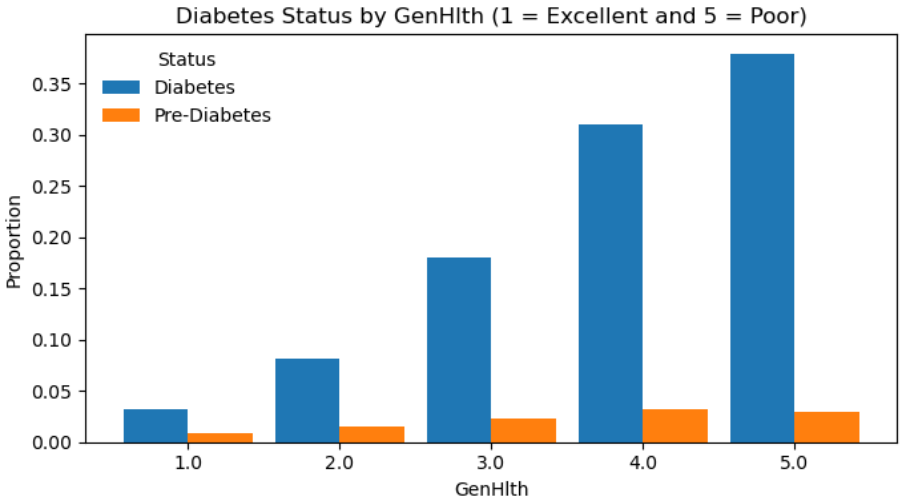
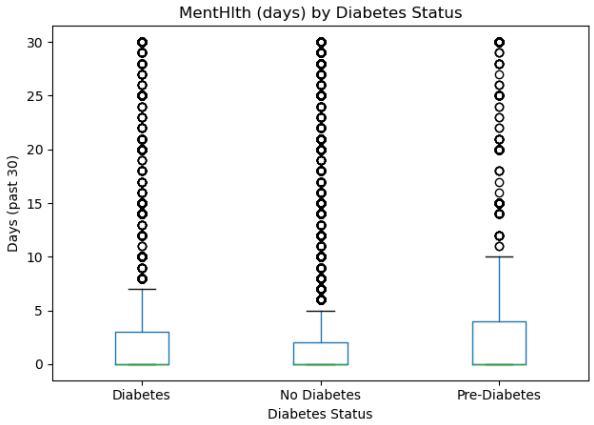


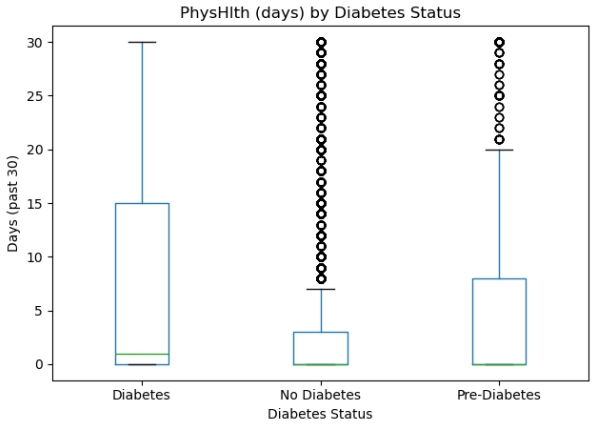
Figure 9: Self-reported burden and functional limitation indicators by diabetes status (II).

4.6.1 Distribution-Aware Summaries: `MentHlth` and `PhysHlth`

`MentHlth` and `PhysHlth` record the number of days in the past 30 when mental or physical health was "not good." These bounded counts (0–30) are typically right-skewed and zero-inflated, so mean comparisons can be misleading. Therefore emphasis is on medians, IQRs, and selected percentiles, alongside full distribution views. The diabetes group shows a clear shift toward higher values, consistent with greater day-to-day burden.



(a) Distribution-aware view of `MentHlth` by diabetes status.



(b) Distribution-aware view of `PhysHlth` by diabetes status.

Figure 10: Distribution-aware summaries of mental and physical health burden (0–30 days) by diabetes status.

Table 2: Distribution-aware summaries of `MentHlth` and `PhysHlth` (0–30 days) by diabetes status.

Diabetes status	n	Ment med	Phys med	Ment IQR	Phys IQR	Ment 0%	Phys 0%
Diabetes	35097	0.0	1.0	3.0	15.0	0.659743	0.470040
No diabetes	190055	0.0	0.0	2.0	3.0	0.665670	0.620405
Prediabetes	4629	0.0	0.0	4.0	8.0	0.638151	0.533377

4.6.2 High-Burden Thresholds (14+ days)

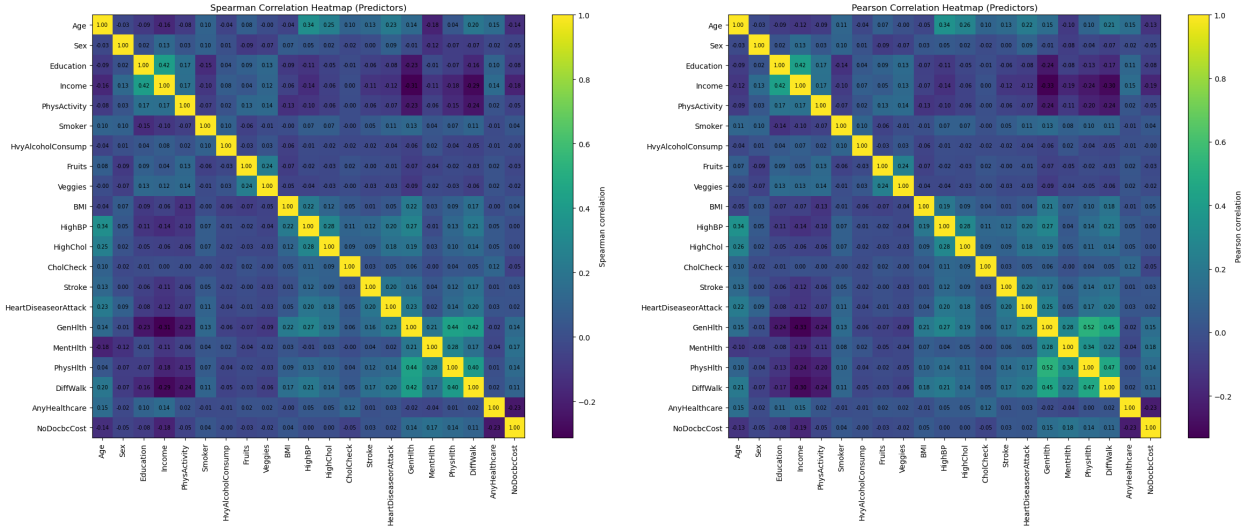
To capture sustained impairment, "high-burden" is defined as reporting 14 or more days in the past 30 when mental health or physical health was not good. This creates two clear binary indicators that are easy to compare across diabetes groups and are less sensitive to skew and many zero responses.

Table 3: Prevalence of high-burden mental and physical health days (14+ days) by diabetes status.

Diabetes status	MentHlth 14+ (%)	PhysHlth 14+ (%)
Diabetes	14.7	26.7
No diabetes	9.8	12.3
Prediabetes	14.5	20.4

4.7 Correlation, Redundancy, and Multivariate Checks

Correlations are used to check redundancy and guide scaling/interpretation (not feature selection), with dependencies clustering into indicator risk (e.g., `HighBP`/`HighChol`), burden/functional, and access/socio-economic status bundles.



(a) Spearman. (b) Pearson.

Figure 11: Predictor correlation structure (deduplicated dataset).

Implications for modeling. Redundancy is expected in population-health survey features and mainly affects interpretation: linear-model coefficients should be read as *conditional* effects, clustering requires scaling so a few higher-variance variables do not dominate distances, and explainability is most stable when framed around coherent bundles rather than single predictors.

Outcome association and redundancy diagnostics. To summarize which predictors align most strongly with diabetes status while acknowledging overlap, the following are reported: (i) ranked associations with the outcome (Spearman/Pearson) and (ii) VIF diagnostics for multicollinearity. These checks inform interpretation and pipeline design.

Table 4: Outcome association and redundancy diagnostics (top features).

(a) Top outcome associations.

Feature	Spearman	Pearson
GenHlth	0.2809	0.2849
HighBP	0.2619	0.2620
BMI	0.2267	0.2120
DiffWalk	0.2097	0.2106
HighChol	0.2048	0.2033
Age	0.1837	0.1846

(b) VIF (redundancy).

Feature	VIF
GenHlth	1.716
PhysHlth	1.595
DiffWalk	1.512
Income	1.431
Age	1.353
HighBP	1.301

Nonlinear dependence (complementary view). Because correlation can miss threshold-like relationships, mutual information (MI) is reported as a model-agnostic dependency measure. MI concentrates on the same small set of burden/health-indicator variables, reinforcing the EDA conclusion that a coherent "health-burden" signal dominates.

Table 5: Mutual information with the outcome

Feature	MI
GenHlth	0.041788
HighBP	0.035211
BMI	0.028975
Age	0.023068
HighChol	0.020995
DiffWalk	0.019303

Interaction and confounding checks (appendix). Additional multivariate views (Age×BMI gradients, burden interactions with `HighBP`/`DiffWalk`, and an age-standardized check for `HvyAlcoholConsump`) are included in Appendix Section A.2 to keep the main narrative focused.

4.8 EDA Takeaways

Across this survey feature space, diabetes prevalence is driven most strongly by broad clinical burden rather than any single behavior. Age and BMI show clear upward gradients, and health indicators (especially `GenHlth` plus indicators such as `HighBP`, `HighChol`, and cardiovascular-history markers) co-occur more often among respondents with diabetes.

Socioeconomic differences are also visible: prevalence varies across `Income` and `Education`, while sex differences are modest. Lifestyle variables trend in expected directions but are weaker separators in univariate views and are best framed as modifiable intervention levers rather than diagnostic proxies. Finally, correlation checks suggest related feature clusters, motivating grouped interpretation and leakage-safe preprocessing for both supervised risk scoring and clustering.

5 Data Preprocessing & Feature Engineering

This section summarizes the end-to-end preparation pipeline: data-quality checks and deduplication, confirmation of missingness and survey encodings, the scaling strategy for supervised vs. unsupervised methods, the small set of engineered high-burden indicators, and the leakage-safe preprocessing pipelines. It concludes with the stratified train/validation/test split used to support model selection and an unbiased final evaluation.

5.1 Data Quality, Missingness, and Feature Encoding/Scaling Plan

After loading the CDC Diabetes Health Indicators dataset, exact duplicate rows were removed (survey snapshots; not repeated measurements) to avoid inflating the effective sample size and to prevent duplicate leakage across train/validation/test splits. After deduplication, the working dataset contains $n = 229\,781$ respondents.

Missingness. All 22 columns (`Diabetes_012` plus 21 predictors) are complete after deduplication, so no imputation is performed.

Encoding. Predictors are already numeric survey encodings: mostly binary indicators (0/1), ordinal scales (`GenHlth`, `Age`, `Education`, `Income`), and count/continuous measures (`BMI`, `MentHlth`, `PhysHlth`). They are used as-is.

Scaling. Scale-sensitive supervised models standardize only continuous/count variables (e.g., `BMI`, `MentHlth`, `PhysHlth`) within training folds, leaving binary/ordinal features unchanged. For distance-based clustering, all features are scaled to comparable ranges so `BMI` and 0–30 day burden measures do not dominate distances. All scaling parameters are fit on training data only and applied to validation/test to avoid leakage.

5.2 Engineered Features

Feature engineering was kept intentionally limited to preserve interpretability in a dataset dominated by binary and ordinal survey indicators. The only added variables convert the zero-inflated 0–30

day burden measures into sustained high-burden flags: `MentHlth_14plus` and `PhysHlth_14plus`, defined as 1 when the respondent reports ≥ 14 days of poor mental or physical health in the past 30 days. The 14-day cutoff (at least half of the month) emphasises persistent impairment, reduces sensitivity to long-tailed day counts, and yields a simple feature that is easy to explain and use in downstream models.

```python
# Zero-heavy burden flags (0 vs >0 days)
df["MentHlth_gt0"] = (df["MentHlth"] > 0).astype(int)
df["PhysHlth_gt0"] = (df["PhysHlth"] > 0).astype(int)

# High-burden flags aligned with the EDA "14+ days" threshold
df["MentHlth_14plus"] = (df["MentHlth"] >= 14).astype(int)
df["PhysHlth_14plus"] = (df["PhysHlth"] >= 14).astype(int)

# build X after feature engineering
X = df.drop(columns=[target_col]).copy()

# Add engineered flags to binary feature list
binary_cols = binary_cols + ["MentHlth_gt0", "PhysHlth_gt0", "MentHlth_14plus", "PhysHlth_14plus"]
```

Figure 12: Engineered high-burden indicators derived from `MentHlth` and `PhysHlth` (14+ days).

5.3 Preprocessing Pipelines

This subsection describes the end-to-end preprocessing used for clustering and classification, emphasizing (i) excluding the target from unsupervised segmentation, (ii) scaling choices driven by distance sensitivity, and (iii) leakage prevention by fitting transformations on training data only.

5.3.1 Unsupervised Learning Pipeline (Clustering)

Clustering is used to identify *population segments* in the predictor space, so `Diabetes_012` is excluded from the clustering matrix (diabetes/prediabetes prevalence is computed *post hoc* for cluster profiling).

Because K-Means/DBSCAN/hierarchical methods rely on distances, all predictors are scaled before clustering so variables on larger ranges do not dominate. In practice, `BMI`, `MentHlth`, and `PhysHlth` are standardized and the full predictor matrix is transformed to a comparable scale, then the same scaled matrix is used across clustering methods to enable fair comparison. Cluster solutions are assessed using internal separation metrics where applicable (e.g., silhouette), sensitivity to random seeds/initialization, and interpretability of cluster profiles.

Figures 13 and 14 summarize the unsupervised preprocessing pipeline and the resulting modeling matrix used for clustering.

```python
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler, RobustScaler
from sklearn.pipeline import Pipeline

ohe_unsup = OneHotEncoder(handle_unknown="ignore", sparse_output=False) # return numpy array, not sparse matrix

preprocess_unsupervised = ColumnTransformer(
    transformers=[
        ("bin", "passthrough", binary_cols), # retained as 0's/1's
        ("cat", ohe_unsup, cat_cols),
        ("num", StandardScaler(), continuous_cols + ordinal_cols),  # BMI + GenHlth (# potential alternative consideration is MinMax countinuous_cols + ordinal_cols + count_cols) removing robust
        ("cnt", RobustScaler(), count_cols),                # Ment/Phys days
    ],
    remainder="drop", # drop any columns not explicitly listed in the transformers above
    verbose_feature_names_out=False # keep output feature names clean (no transformer prefixes like "cat__Age_1.0")
)
```

Figure 13: Unsupervised preprocessing pipeline for clustering (overview).

```
X_unsup = preprocess_unsupervised.fit_transform(X)

print("Clustering matrix shape:", X_unsup.shape)

feature_names = preprocess_unsupervised.get_feature_names_out()
print("First 25 feature names:", feature_names[:25])
```

```
[187] ✓ 0.1s

··· Clustering matrix shape: (229781, 53)
First 25 feature names: ['HighBP' 'HighChol' 'CholCheck' 'Smoker' 'Stroke' 'HeartDiseaseorAttack'
 'PhysActivity' 'Fruits' 'Veggies' 'HvyAlcoholConsump' 'AnyHealthcare'
 'NoDocbcCost' 'DiffWalk' 'Sex' 'MentHlth_gt0' 'PhysHlth_gt0'
 'MentHlth_14plus' 'PhysHlth_14plus' 'Age_1.0' 'Age_2.0' 'Age_3.0'
 'Age_4.0' 'Age_5.0' 'Age_6.0' 'Age_7.0']
```

Figure 14: Clustering feature matrix after excluding `Diabetes_012` and applying scaling.

5.3.2 Supervised Learning Pipeline (Classification)

For supervised risk prediction, preprocessing is applied in a leakage-safe way: all transformations are fit on the training data only and then applied unchanged to validation/test. Binary and ordinal survey encodings are kept in their coded form, while scale-sensitive variables (`BMI`, `MentHlth`, `PhysHlth`) and any engineered burden indicators are standardized so models like logistic regression are not driven by raw magnitude differences. Imbalance mitigation (class weights and/or resampling) is performed within the training pipeline, and predicted probabilities can be calibrated/thresholded for screening use.

```python
# One-hot encoder for categorical-style survey codes
ohe_sup = OneHotEncoder(handle_unknown="ignore", sparse_output=False)  # ignore unseen categories at test time

# Preprocessor A (Scaled): for models that benefit from scaling (e.x., linear models, KNN)
# Standardize continuous/ordinal-ish features; robust-scale zero-heavy day counts.
preprocess_scaled = ColumnTransformer(
    transformers=[
        ("bin", "passthrough", binary_cols),                # 0/1 indicators + engineered flags
        ("cat", ohe_sup, cat_cols),                         # one-hot: Age, Education, Income, BMI_cat
        ("num", StandardScaler(), continuous_cols + ordinal_cols),  # BMI + GenHlth
        ("cnt", RobustScaler(), count_cols),                # MentHlth, PhysHlth (0-30, zero-heavy)
    ],
    remainder="drop",               # drop any columns not explicitly listed above
    verbose_feature_names_out=False   # cleaner names (no 'cat__', 'num__' prefixes)
)

# Preprocessor B (Unscaled): for models that don't require feature scaling (e.x., tree-based models)
# Same encoding + engineered flags, but leave numeric values unscaled for tree-based models.
preprocess_unscaled = ColumnTransformer(
    transformers=[
        ("bin", "passthrough", binary_cols),
        ("cat", ohe_sup, cat_cols),
        ("num", "passthrough", continuous_cols + ordinal_cols + count_cols),  # keep raw BMI/GenHlth/MentHlth/PhysHlth
    ],
    remainder="drop",
    verbose_feature_names_out=False
)
```

Figure 15: Supervised learning pipeline (overview).

5.4 Train/Validation/Test Split

The dataset was split into disjoint train, validation, and test sets using stratification on `Diabetes_012` so that the strong class imbalance (especially the small prediabetes class) is preserved in each split. The training set is used to fit preprocessing and model parameters, the validation set is reserved for model selection (hyperparameters, calibration choice, and decision-threshold tuning), and the test set is held out for a single final estimate of generalization. To avoid leakage, any preprocessing and imbalance-handling steps are fit on training data only (or within training folds) and then applied to validation/test data after fitting.

```python
from sklearn.model_selection import train_test_split

supervised_df = df.copy()

# first split: train vs temp (stratified)
training_data, temp_df = train_test_split(
    supervised_df,
    test_size=0.30,     # 70% train, 30% temp
    stratify=supervised_df[target_col],
    random_state=42,
    shuffle=True
)

# second split: temp -> val vs test (stratified)
validation_data, testing_data = train_test_split(
    temp_df,
    test_size=0.50,     # 15% val, 15% test
    stratify=temp_df[target_col],
    random_state=42,
    shuffle=True
)
```

Figure 16: Train/validation/test split strategy.

6 Unsupervised Learning

This section uses unsupervised methods to (i) mine co-occurring risk-factor patterns and (ii) segment the population into interpretable groups for targeted public health intervention.

6.1 Unsupervised Pattern Mining: Co-occurring Risk Factor Rules (Apriori)

Apriori identifies combinations of behaviours and health indicators that frequently appear together. The goal is to surface common "risk profiles" that can support later cluster interpretation.

6.1.1 Item Matrix Construction

For association rule mining, each respondent is represented as a set of binary *items* in a sparse matrix. Binary survey indicators are mapped directly to items (e.g., `HighBP=1`, `HighChol=1`). Ordered and continuous measures are first grouped into interpretable public-health categories before binarisation (e.g., BMI bands; coarse age, education, and income levels) so that rules correspond to meaningful profiles rather than arbitrary numeric cutoffs. The zero-heavy burden measures `MentHlth` and `PhysHlth` are additionally summarised with high-burden flags using a 14+ day threshold to capture

sustained impairment. The resulting respondent-by-item boolean table is then used as input to frequent itemset mining and rule generation.

```
# Turn the unsupervised preprocessed array back into a DataFrame for Apriori
X_unsup_df = pd.DataFrame(X_unsup, columns=feature_names, index=X.index)

# Columns that are not valid "items" for Apriori (continuous / scaled numeric outputs)
non_item_cols = ["BMI", "GenHlth", "MentHlth", "PhysHlth"]

item_cols = [c for c in X_unsup_df.columns if c not in non_item_cols]

# Build item matrix (boolean True/False is what mlxtend expects)
item_matrix = X_unsup_df[item_cols].astype(bool)
```

Figure 17: Construction of the binary item matrix used for Apriori association rule mining.

6.1.2 Frequent Itemsets

Frequent itemsets were mined from the binary item matrix to identify common co-occurring risk-factor patterns in the survey population. To improve interpretability and avoid flooding results with trivial patterns, near-universal items (prevalence $> 90\%$) were removed (`CholCheck`, `AnyHealthcare`), and ultra-rare items (prevalence $< 1\%$) were removed (`Education_1.0`). Itemsets were then mined with minimum support $= 0.05$ and maximum size $= 4$, yielding 1,139 frequent itemsets.

```python
from mlxtend.frequent_patterns import apriori

# Drop items that are "almost always true" (e.x AnyHealthcare)
item_prev = item_matrix.mean().sort_values(ascending=False)

drop_thresh = 0.90  # items present in >90% of respondents add little differentiation
drop_items = item_prev[item_prev > drop_thresh].index.tolist()

item_matrix_ap = item_matrix.drop(columns=drop_items)

print("Dropped near-universal items (prevalence > {:.0%}):".format(drop_thresh), drop_items)
print("Item matrix shape after drop:", item_matrix_ap.shape)

# drop ultra-rare items to speed up Apriori
# Anything <1% can never appear in 5%+ itemsets anyway.
low_prev_thresh = 0.01

# list which ultra-rare items are dropped
rare_items = item_matrix_ap.mean()[lambda s: s < low_prev_thresh].index.tolist()
print("Dropped ultra-rare items (prevalence < {:.0%}):".format(low_prev_thresh), rare_items)

keep_items = item_matrix_ap.mean()[lambda s: s >= low_prev_thresh].index
item_matrix_ap = item_matrix_ap[keep_items]

print("Item matrix shape after rare-drop:", item_matrix_ap.shape)

# Mine frequent itemsets with interpretability constraints
min_support = 0.05
max_len = 4  # keep itemsets readable and reduce combinatorial explosion

freq_itemsets = apriori(
    item_matrix_ap,
    min_support=min_support,
    use_colnames=True,
    max_len=max_len
).sort_values("support", ascending=False)

print("Frequent itemsets found:", freq_itemsets.shape[0])
display(freq_itemsets.head(15))

freq_itemsets["itemset_size"] = freq_itemsets["itemsets"].apply(len)
display(freq_itemsets["itemset_size"].value_counts().sort_index())
```

Figure 18: Top frequent itemsets under the selected minimum support threshold.

The highest-support patterns are dominated by broadly prevalent lifestyle indicators (e.g., `Veggies`, `PhysActivity`, `Fruits`) and common risk flags mentioned previously (e.g., `HighBP`, `HighChol`). As expected, support decreases as more conditions are combined: the mined set contains 38 singletons, 233 pairs, 486 triples, and 382 four-itemsets. This motivates focusing downstream association-rule generation on additional strength criteria (e.g., confidence/lift), since frequency alone can reflect baseline prevalence rather than meaningful dependence.

Table 6: Examples of frequent itemsets with highest support (filtered item matrix).

Support	Itemset
0.794813	{Veggies}
0.733355	{PhysActivity}
0.612966	{Fruits}
0.607074	{Veggies, PhysActivity}
0.535014	{Veggies, Fruits}
0.476506	{PhysActivity, Fruits}
0.465661	{Smoker}
0.454441	{HighBP}
0.441760	{HighChol}
0.439231	{Sex}
0.424243	{Veggies, PhysActivity, Fruits}
0.404315	{PhysHlth_gt0}
0.384901	{Education_6.0}
0.367315	{Veggies, Smoker}
0.354925	{BMI_cat_Overweight}

6.1.3 Association Rules and Filtering

Association rules were generated from frequent itemsets by enumerating antecedent $\Rightarrow$ consequent splits and computing *support* (prevalence), *confidence* ($P(\text{consequent} \mid \text{antecedent})$), and *lift* (enrichment over the consequent base rate). Rules were filtered to keep patterns that are both common enough to matter and meaningfully enriched (lift > 1), then deduplicated to remove near-equivalent variants and retain a compact, interpretable rule set.

```
from mlxtend.frequent_patterns import association_rules

# Convert frequent itemsets -> association rules (broad pass; filtering comes next)
rules = association_rules(
    freq_itemsets,
    metric="lift",
    min_threshold=1.0
)

# Quick look (prioritize stronger + more reliable rules)
rules_preview = (
    rules.sort_values(["lift", "confidence", "support"], ascending=False)
        .reset_index(drop=True)
)

display(
    rules_preview[["antecedents", "consequents", "support", "confidence", "lift"]].head(10)
)
```

Figure 19: Association-rule workflow (rule generation and filtering).

After filtering, the strongest rules consistently link mobility limitation and non-zero physical-health burden (`DiffWalk`, `PhysHlth_gt0`) with high physical-health burden (`PhysHlth_14plus`) and common cardiometabolic indicators (`HighBP`, `HighChol`) or smoking. High lift values indicate these co-occurrences are far more frequent than expected from marginal prevalence alone, supporting the EDA theme that "burden bundles" dominate over isolated behaviors.

Table 7: Preview of association rules (support, confidence, lift).

Antecedent	Consequent	Support	Confidence	Lift
{PhysHlth_14plus, HighBP}	{PhysHlth_gt0, DiffWalk}	0.058991	0.655623	4.814164
{PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus, HighBP}	0.058991	0.433164	4.814164
{PhysHlth_14plus, MentHlth_gt0}	{PhysHlth_gt0, DiffWalk}	0.051419	0.645628	4.740777
{PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus, MentHlth_gt0}	0.051419	0.377560	4.740777
{MentHlth_gt0, PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus}	0.051419	0.682672	4.662911
{PhysHlth_14plus}	{MentHlth_gt0, PhysHlth_gt0, DiffWalk}	0.051419	0.351208	4.662911
{PhysHlth_14plus, HighChol}	{PhysHlth_gt0, DiffWalk}	0.052524	0.634776	4.661091
{PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus, HighChol}	0.052524	0.385677	4.661091
{PhysHlth_14plus, Smoker}	{PhysHlth_gt0, DiffWalk}	0.053738	0.630934	4.632874
{PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus, Smoker}	0.053738	0.394593	4.632874

```
# Filter strong rules
strong_rules = rules[
    (rules["support"] >= 0.05) & # rule occurs in at least 5% of respondents
    (rules["lift"] > 3.5) & # antecedant+consequent appear together >3.5x more than expected by chance
    (rules["confidence"] > 0.60) # consequent happens in >60% of cases where antecedent is present
].sort_values("lift", ascending=False) # sort by strongest association first

print("Rules after filtering:", strong_rules.shape[0])
strong_rules[["antecedents", "consequents", "support", "confidence", "lift"]]
```

Figure 20: Filtered rule set (visual summary).

For reporting, a focus was on the highest-lift rules that are still operationally common (support $\approx$ 5–9%). These rules emphasize that severe physical-health burden (`PhysHlth_14plus`) is strongly associated with functional limitation and broader burden context (including cardiometabolic indicators), providing concise, actionable co-occurrence profiles.

Table 8: Filtered association rules (selected high-lift examples).

Antecedent	Consequent	Support	Confidence	Lift
{PhysHlth_14plus, HighBP}	{PhysHlth_gt0, DiffWalk}	0.058991	0.655623	4.814164
{PhysHlth_14plus, MentHlth_gt0}	{PhysHlth_gt0, DiffWalk}	0.051419	0.645628	4.740777
{MentHlth_gt0, PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus}	0.051419	0.682672	4.662911
{PhysHlth_14plus, HighChol}	{PhysHlth_gt0, DiffWalk}	0.052524	0.634776	4.661091
{PhysHlth_14plus, Smoker}	{PhysHlth_gt0, DiffWalk}	0.053738	0.630934	4.632874
{PhysHlth_gt0, DiffWalk, Smoker}	{PhysHlth_14plus}	0.053738	0.663479	4.531815
{HighChol, PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus}	0.052524	0.645815	4.411169
{PhysHlth_gt0, HighBP, DiffWalk}	{PhysHlth_14plus}	0.058991	0.637313	4.353090
{PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus}	0.086456	0.634838	4.336191
{Veggies, PhysHlth_gt0, DiffWalk}	{PhysHlth_14plus}	0.063082	0.628496	4.292869

6.1.4 Key Co-occurrence Patterns

The filtered rules above summarize the most stable co-occurring "risk profiles" in the dataset, dominated by high physical-health burden (`PhysHlth_14plus`), mobility limitation (`DiffWalk`), and the stacking of chronic-condition markers (`HighBP`, `HighChol`, `Smoker`). These patterns provide an interpretable reference point for the next step: population segmentation via clustering.

6.2 Population Segmentation (Clustering)

Population segmentation identifies recurring "risk profiles" that may not be obvious from single-variable summaries, the goal is to connect clusters back to the earlier APriori co-occurance paterns and profile each cluster for context towards diabetes.

6.2.1 Method Rationale and Setup

Inputs are standardised to prevent large-scale variables dominating distance computations, and PCA is used only as a low-dimensional representation for visualisation. Model selection emphasises three criteria: (i) **separation** via silhouette scores where applicable, (ii) **stability** under repeated initialisations/seeds, and (iii) **interpretability** of the resulting clusters in terms of coherent health and sociodemographic patterns.

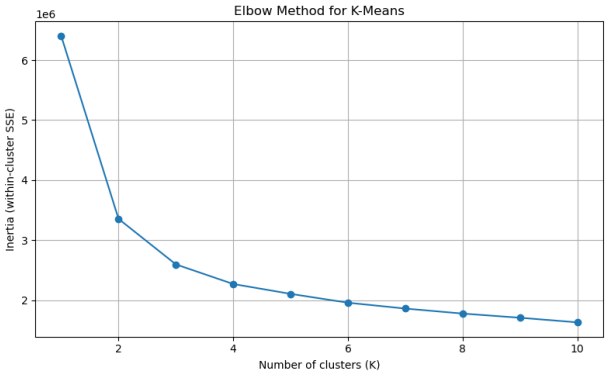
6.2.2 K-Means: Selecting k, Stability, and Profiling

K-Means is used as the primary baseline for population segmentation because it yields compact clusters that are straightforward to profile and compare across candidate values of k.

Selecting k (inertia elbow). Candidate solutions were first compared using the inertia elbow diagnostic. The code and resulting plot are shown in Figure 21, supporting $k = 4$ as a practical balance between structure and interpretability.

```python
from sklearn.cluster import KMeans

inertia = []

for k in k_range:
    km = KMeans(
        n_clusters=k,
        n_init=10, # more stable than default single init
        random_state=42
    )

    km.fit(X_unsup)
    inertia.append(km.inertia_)

plt.figure(figsize=(8, 5))
plt.plot(list(k_range), inertia, marker="o")
plt.xlabel("Number of clusters (K)")
plt.ylabel("Inertia (within-cluster SSE)")
plt.title("Elbow Method for K-Means")
plt.grid(True)
plt.tight_layout()
plt.show()
```

(a) Code: inertia elbow diagnostic.



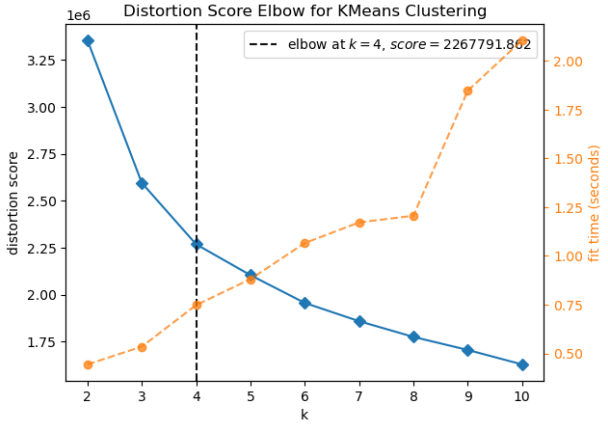
(b) Inertia elbow plot.

Figure 21: K-Means k selection using inertia.

Cross-checking k (Yellowbrick distortion). Because elbow interpretation can be subjective, k was cross-checked using Yellowbrick's distortion-based elbow diagnostic. The code and output in Figure 22 corroborate the choice of $k = 4$.

```python
from yellowbrick.cluster import KElbowVisualizer

model = KMeans(
    n_init=10,
    random_state=42
)

visualizer = KElbowVisualizer(
    model,
    k=(2,11),
    force_model=True # <- bypass Yellowbrick's clusterer check
)

visualizer.fit(X_unsup)
visualizer.show()
plt.rcParams['axes.prop_cycle'] = plt.cycler(color=plt.cm.tab10.colors)
plt.show()
```

(a) Code: Yellowbrick distortion elbow.



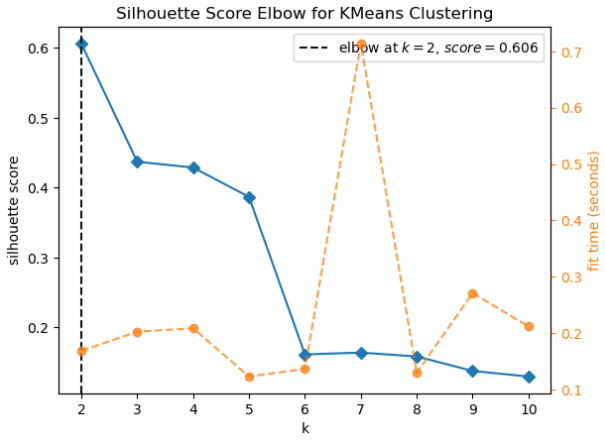
(b) Distortion elbow plot.

Figure 22: K-Means k cross-check using Yellowbrick distortion.

Silhouette structure (separation check). Silhouette diagnostics were then used to confirm that the selected $k = 4$ solution remains viable in terms of separation. The code and Yellowbrick silhouette visualiser are shown in Figure 23.

```python
from sklearn.metrics import silhouette_score

k_star = 4
km4 = KMeans(
    n_clusters=k_star,
    n_init=10,
    random_state=42
)

labels4 = km4.fit_predict(X_viz)

sil4 = silhouette_score(X_viz, labels4)
print(f"Silhouette score at k={k_star} (sample={X_viz.shape[0]}): {sil4:.4f}")
```

(a) Code: silhouette evaluation.



(b) Silhouette visualiser.

Figure 23: Silhouette diagnostics supporting the selected $k = 4$ solution.

Cluster sizes (selected $k = 4$). After fixing $k = 4$, cluster sizes were inspected to ensure segments are interpretable and not dominated by tiny fragments; Figure 24 and Table 9 summarise the resulting proportions.

```python
k_final = 4

kmeans_final = KMeans(
    n_clusters=k_final,
    n_init=10,
    random_state=42
)

labels_final = kmeans_final.fit_predict(X_unsup)

# Cluster sizes
cluster_sizes = pd.Series(labels_final).value_counts().sort_index()
cluster_props = (cluster_sizes / cluster_sizes.sum()).round(4)

display(pd.DataFrame({
    "n": cluster_sizes,
    "proportion": cluster_props
}))
```

Figure 24: Cluster sizes ($k = 4$).

Table 9: Cluster sizes for $k = 4$.

Cluster	n	Proportion
0	176,322	0.7673
1	21,569	0.0939
2	15,161	0.0660
3	16,729	0.0728

Stability (ARI across seeds). Stability was evaluated by re-running K-Means across multiple random seeds and measuring agreement with Adjusted Rand Index (ARI). Figure 25 reports high agreement (mean ≈ 0.984, std ≈ 0.019), indicating the $k = 4$ structure is not a single-initialisation artefact.

```python
from sklearn.metrics import adjusted_rand_score

seeds = [0, 1, 2, 3, 4]
labels_list = []

for s in seeds:
    km = KMeans(n_clusters=k_final, n_init=10, random_state=s)
    labels_list.append(km.fit_predict(X_unsup))

# Pairwise ARI across seeds (higher = more consistent partition)
aris = []
for i in range(len(labels_list)):
    for j in range(i + 1, len(labels_lis
        aris.append(adjusted_rand_score(labels_list[i], labels_list[j]))

print(f"Stability (ARI) across seeds: mean={np.mean(aris):.3f}, std={np.std(aris):.3f}")
```

(variable) labels_list: list

[199] ✓ 3.4s

```
Stability (ARI) across seeds: mean=0.984, std=0.019
```

Figure 25: Stability across seeds using ARI (mean ≈ 0.984, std ≈ 0.019).

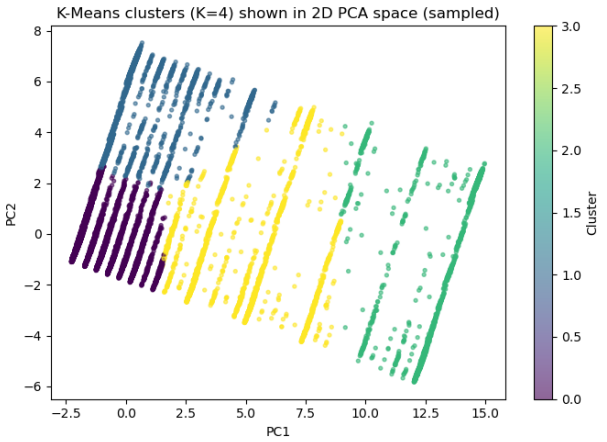
PCA projection (visualisation only). For visual confirmation only, clusters were projected into a low-dimensional PCA space. The code and resulting projection plot are shown in Figure 26.

```python
from sklearn.decomposition import PCA

# sample for plotting
rng = np.random.default_rng(42)
n = X_unsup.shape[0]
plot_n = 30000
idx = rng.choice(n, size=min(plot_n, n), replace=False)

X_plot = X_unsup[idx]
labels_plot = labels_final[idx]  # labels from fitted kmeans_final

# 2D projection
pca = PCA(n_components=2, random_state=42)
X_2d = pca.fit_transform(X_plot)

print("Explained variance (PC1, PC2):", pca.explained_variance_ratio_)
print("Total (PC1+PC2):", pca.explained_variance_ratio_.sum())

plt.figure(figsize=(7, 5))
plt.scatter(X_2d[:, 0], X_2d[:, 1], c=labels_plot, s=8, alpha=0.6)
plt.title("K-Means clusters (K=4) shown in 2D PCA space (sampled)")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.colorbar(label='Cluster')
plt.tight_layout()
plt.show()
```

[201] ✓ 0.3s

```
Explained variance (PC1, PC2): [0.57919177 0.16566183]
Total (PC1+PC2): 0.7448536053322843
```

(a) Code: PCA projection.



(b) PCA projection ($k = 4$).

Figure 26: PCA projection of K-Means clusters (visualisation only).

Profiling (top "high" features per cluster). Clusters were profiled using within-cluster means for continuous/count variables and within-cluster prevalences for binary indicators. Figure 27 shows the profiling code/output, and Table 10 lists the highest-mean features for each cluster. These profiles are descriptive segments intended for planning and targeting, not diagnostic labels.

```
feature_names = preprocess_unsupervised.get_feature_names_out()

X_unsup_df = pd.DataFrame(X_unsup, columns=feature_names)
X_unsup_df["cluster"] = labels_final

cluster_profile = X_unsup_df.groupby("cluster").mean().T

# For each cluster, show the top "high" features (most defining in that cluster)
top_n = 12
for c in sorted(X_unsup_df["cluster"].unique()):
    print(f"\nCluster {c}: top features by mean value")
    display(cluster_profile[c].sort_values(ascending=False).head(top_n).to_frame("mean"))
```

Figure 27: Code/output: extracting top within-cluster "high" features ($k = 4$).

Table 10: Top features by within-cluster mean (K-Means, $k = 4$).

(a) Cluster 0

Feature	Mean
CholCheck	0.957152
AnyHealthcare	0.949121
Veggies	0.804670
PhysActivity	0.774050
Fruits	0.624341
Sex	0.459109
Smoker	0.438414
HighBP	0.426918
HighChol	0.418660
Education_6.0	0.412779
BMI_cat_Overweight	0.366137
Income_8.0	0.346451

(b) Cluster 1

Feature	Mean
PhysHlth	6.226471
GenHlth	1.011654
PhysHlth_gt0	1.000000
PhysHlth_14plus	0.995410
CholCheck	0.979600
AnyHealthcare	0.956743
Veggies	0.774120
MentHlth	0.696509
HighBP	0.614354
Fruits	0.613102
Smoker	0.559924
DiffWalk	0.555102

(c) Cluster 2

Feature	Mean
MentHlth	14.353044
PhysHlth	3.921031
MentHlth_14plus	1.000000
MentHlth_gt0	1.000000
CholCheck	0.959897
GenHlth	0.937385
AnyHealthcare	0.917354
PhysHlth_gt0	0.748302
Veggies	0.730295
Smoker	0.605435
PhysActivity	0.556823
PhysHlth_14plus	0.549304

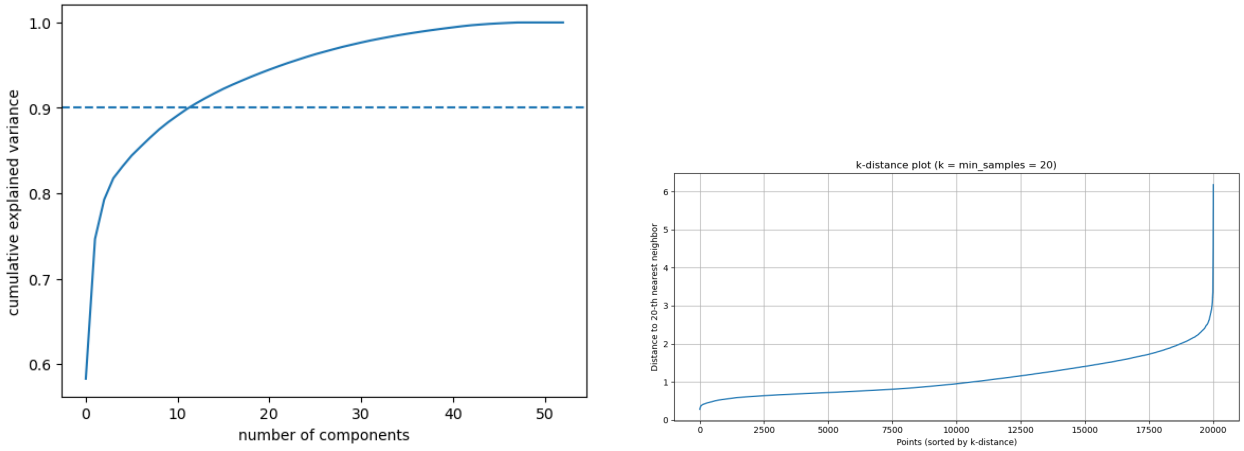
(d) Cluster 3

Feature	Mean
MentHlth	6.659035
PhysHlth	1.679867
MentHlth_gt0	1.000000
CholCheck	0.958455
AnyHealthcare	0.926236
Veggies	0.776077
PhysActivity	0.696754
PhysHlth_gt0	0.605535
Fruits	0.561958
MentHlth_14plus	0.559627
Smoker	0.504633
HighChol	0.460876

6.2.3 DBSCAN: PCA Reduction and Parameter Selection

DBSCAN was evaluated as a complementary segmentation method because it can identify *dense cores* in the predictor space while explicitly labeling low-density observations as *noise*. This is useful in survey health data, where some respondents may reflect atypical or heterogeneous risk-factor combinations rather than stable, policy-actionable segments. The workflow therefore focuses on (i) making distance computations more meaningful and (ii) selecting parameters that yield interpretable cluster cores without collapsing into a single cluster.

PCA reduction before DBSCAN. DBSCAN depends on neighborhood distances, which can become unreliable in high-dimensional mixed-type feature spaces (many binary indicators alongside continuous/ordinal variables). To improve the geometry for density estimation, features are first standardized and then projected into a low-dimensional PCA space. This step reduces noise, mitigates distance concentration, and enables visual inspection of density structure, while retaining the dominant gradients captured by the standardized survey indicators.



(a) PCA space used for DBSCAN. (b) `eps` diagnostic (k-distance view).

Figure 28: DBSCAN setup and parameter guidance in PCA space.

Selecting `eps` and `min_samples`. Parameter selection targets a neighborhood radius (`eps`) that captures coherent dense regions without (i) fragmenting the sample into many tiny clusters with high noise, or (ii) merging structure into one dominant cluster. A k-distance plot provides an initial `eps` region near the knee, and a local sweep then checks how the number of clusters, the noise rate, and separation among non-noise points behave around that region. Throughout, silhouette is computed on *non-noise* points only and is only meaningful when DBSCAN finds at least two clusters.

Table 11: DBSCAN `eps` sweep near the knee in PCA space (`min_samples`=20). Silhouette is computed on non-noise points only.

ϵ	`min_samples`	# clusters	Noise (%)	Silhouette (non-noise)
1.500000	20	6	10.960	0.422566
1.571429	20	6	8.815	0.502639
1.642857	20	7	7.200	0.567715
1.714286	20	4	5.870	0.607216
1.785714	20	5	4.835	0.584683
1.857143	20	3	3.815	0.585558
1.928571	20	5	2.820	0.422451
2.000000	20	5	1.755	0.416119
2.071429	20	4	1.090	0.433313
2.142857	20	3	0.815	0.452064
2.214286	20	2	0.615	0.358926
2.285714	20	1	0.430	–
2.357143	20	1	0.315	–
2.428571	20	1	0.265	–
2.500000	20	1	0.205	–

Interpreting the sweep (clusters vs. noise). The sweep shows the expected DBSCAN trade-off: smaller `eps` values produce more clusters but higher noise, while larger `eps` values steadily reduce noise and eventually collapse into a single cluster. A practically useful region is where DBSCAN yields multiple clusters with a manageable noise rate and reasonable separation among non-noise points (here, the knee neighborhood yields several viable settings with modest noise and moderate-to-strong silhouette on the retained cores). The noise label is interpreted substantively: it isolates low-density, heterogeneous respondents rather than forcing every observation into a centroid-defined segment.

Relationship to K-Means (segmentation utility). DBSCAN provides a different lens than K-Means: it emphasizes only the *most coherent* dense regions and explicitly acknowledges a residual set that may not belong to any stable cluster. In contrast, K-Means is better suited when the goal is a compact, fully partitioned set of segments for profiling and downstream reporting. DBSCAN is therefore used here as a robustness/structure check, confirming that meaningful dense cores exist, rather than as the primary segmentation used for population-wide stratification.

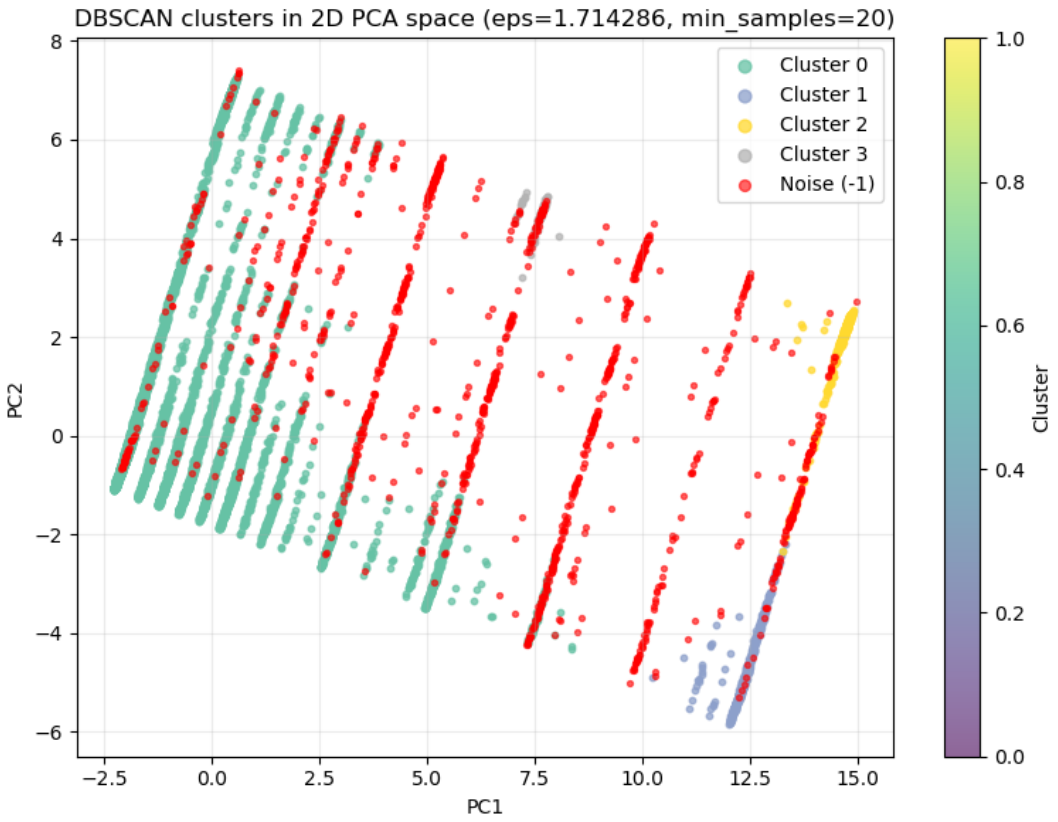


Figure 29: DBSCAN summary: dense cores vs. noise (comparison view).

6.2.4 Hierarchical Clustering: Dendrogram and Cluster Cut

Hierarchical clustering provides a complementary segmentation lens by showing a *nested* similarity structure. Rather than forcing a fixed partition, the dendrogram reveals how respondents merge as the distance threshold increases, which helps assess whether the feature space supports a small number of stable, interpretable macro-segments.

Setup (PCA space + Ward linkage). Clustering was performed in the same reduced PCA space used elsewhere to improve distance behavior in a mixed, high-dimensional survey feature set. Ward linkage was used to encourage compact, variance-minimizing groups, supporting downstream profiling and comparison.

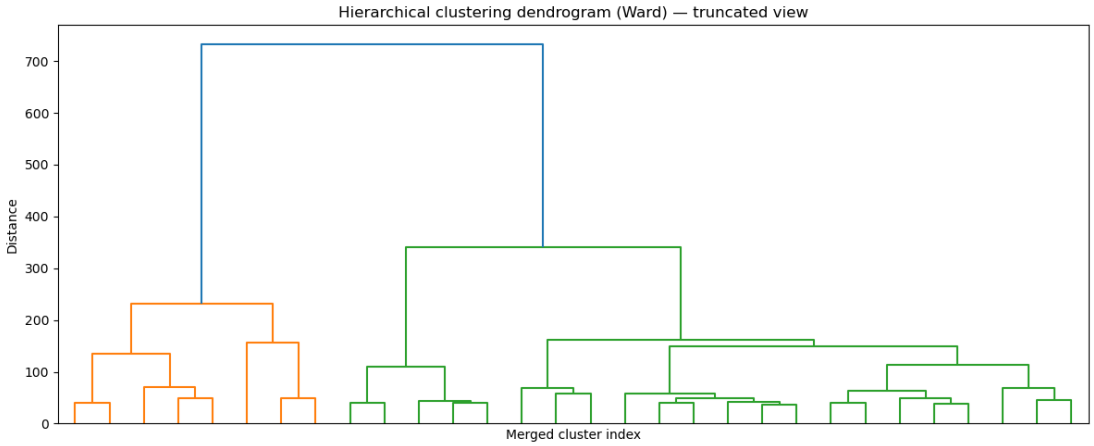
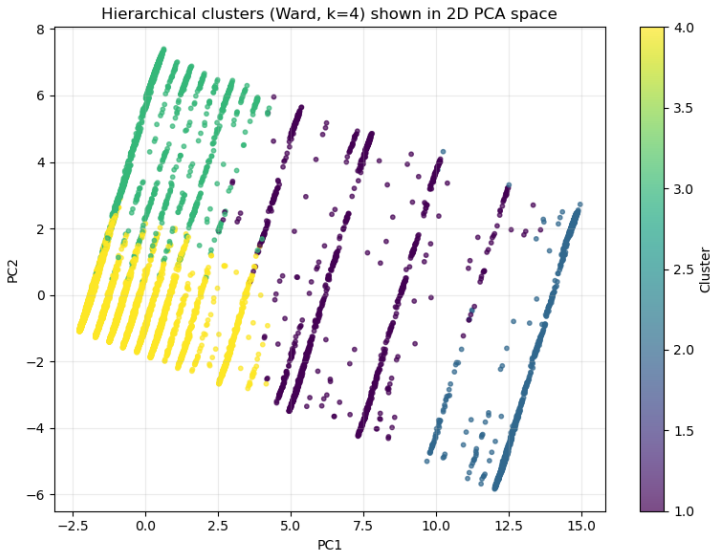


Figure 30: Ward-linkage dendrogram (truncated view).

Choosing the cut (practical $k = 4$ solution). A cut producing $k = 4$ clusters was selected to balance interpretability against overly fragmented micro-groups. This yields one large baseline segment alongside three smaller, higher-contrast groups, which is useful for profiling and outcome comparison.

Cluster	n
1	1,152
2	1,169
3	1,918
4	15,761

(a) Cluster sizes ($k = 4$; $n = 20{,}000$ sample).



(b) Cluster assignments under the selected cut ($k = 4$) in PCA space.

Figure 31: Hierarchical clustering ($k = 4$): cluster sizes and PCA visualization.

Post-hoc validation with diabetes prevalence. To connect segments to the study outcome *without* using labels to form clusters, diabetes status was evaluated *after* clustering via within-cluster outcome proportions. The largest cluster serves as a lower-risk baseline (highest no-diabetes share), while the three smaller clusters show substantially higher diabetes prevalence. Prediabetes remains comparatively rare across clusters but is modestly elevated in higher-risk segments.

Table 12: Diabetes outcome distribution by hierarchical cluster (post-hoc validation; $n = 20{,}000$ sample).

Cluster	n	No diabetes (%)	Prediabetes (%)	Diabetes (%)
1	1,152	79.2535	2.4306	18.3160
2	1,169	72.9683	3.5928	23.4388
3	1,918	72.6277	2.9718	24.4004
4	15,761	85.4514	1.7385	12.8101

7 Supervised Learning: Diabetes Risk Modeling (3-Class)

7.1 Baseline Logistic Regression

Logistic regression was used as the baseline because it is simple, stable for mostly binary/ordinal survey predictors, and produces interpretable coefficients. Predicted probabilities are treated as screening-style risk scores (for ranking/thresholding rather than diagnosis). This provides an auditable benchmark for evaluating whether more complex models meaningfully improve detection, especially for the harder prediabetes class.

Table 13: Baseline Logistic Regression classification report (Validation vs. Test).

Table 14: Validation

Class	Precision	Recall	F1	Support
0	0.85	0.98	0.91	28,508
1	0.00	0.00	0.00	695
2	0.56	0.18	0.28	5,264
Accuracy			0.83	34,467
Macro avg	0.47	0.39	0.39	34,467
Weighted avg	0.79	0.83	0.79	34,467
Balanced accuracy				0.3864
Overall ROC-AUC				0.7719

Table 15: Test

Class	Precision	Recall	F1	Support
0	0.85	0.97	0.91	28,509
1	0.00	0.00	0.00	694
2	0.56	0.19	0.28	5,265
Accuracy			0.84	34,468
Macro avg	0.47	0.39	0.40	34,468
Weighted avg	0.79	0.84	0.79	34,468
Balanced accuracy				0.3878
Overall ROC-AUC				0.7724

7.2 Imbalance Handling

The target distribution is highly imbalanced, with *prediabetes* as a rare class. Imbalance handling is therefore necessary to avoid models that achieve high overall accuracy by primarily learning the majority (no diabetes) pattern. Because the intended use is *screening support*, minority-class recall and probability ranking for at-risk groups are prioritized, and false negatives are typically more costly than false positives.

Accordingly, multiple imbalance-mitigation strategies were evaluated:

- **Class weighting**: reweights the loss so minority classes contribute more during training while preserving the full dataset.
- **Random under-sampling (RUS)**: reduces the majority-class volume to create a more balanced training set, trading information for improved decision boundaries and faster training.
- **Imbalance-aware boosting (e.g., RUSBoost)**: combines boosting with repeated under-sampling so successive learners focus on hard-to-classify minority cases rather than being dominated by the majority class.

All sampling operations were applied *within* the training data only (after the train/validation/test split) to avoid leakage. Model comparisons emphasize minority-class recall and the precision–recall trade-off, reflecting triage rather than diagnosis.

```python
from imblearn.under_sampling import RandomUnderSampler
from imblearn.over_sampling import RandomOverSampler
from imblearn.combine import SMOTETomek, SMOTEENN
from imblearn.over_sampling import SMOTE  # use with caution in one-hot/binary feature spaces

samplers = {
    "none": None, # no resampling (baseline)

    # Undersampling (primary focus) downsample majority class
    "rus_auto_1to1": RandomUnderSampler(random_state=RNG, sampling_strategy="auto"),  # aggressive stress test (downsamps all classes to class-1 size)
    "rus_3to1": RandomUnderSampler(random_state=RNG, sampling_strategy=rus_3to1_strategy),
    "rus_5to1": RandomUnderSampler(random_state=RNG, sampling_strategy=rus_5to1_strategy),

    # Oversampling by duplication (comparison) (can lead to overfitting, especially with many binary features)
    "ros": RandomOverSampler(random_state=RNG),

    # Hybrid cleanup methods (use with caution)
    # These can help by removing borderline majority points after limited oversampling.
    "smote_tomek": SMOTETomek(random_state=RNG),
    "smote_enn": SMOTEENN(random_state=RNG),

    # Synthetic oversampling (use with caution *Probably leasat ideal for this data*)
    "smote": SMOTE(random_state=RNG),
}
```

Figure 32: Imbalance-handling strategies evaluated (weighting and sampling-based approaches).

7.3 Evaluation Metrics

Because the target is highly imbalanced (especially the rare *prediabetes* class), overall accuracy can be misleading: a classifier may score well by predicting the majority *no diabetes* class while missing minority cases. Error is therefore assessed with class-aware metrics: per-class **precision**, **recall**, and **F1** (prioritizing minority-class recall for screening), **macro-averaged** scores and **balanced accuracy** to reduce majority-class dominance, and **one-vs-rest ROC–AUC** to evaluate probability ranking across thresholds. The **precision–recall trade-off** for at-risk classes is also considered when choosing operating thresholds.

7.4 Nonlinear Models (XGBoost, RUSBoost)

Tree-based ensembles were evaluated to test whether nonlinear interactions among survey indicators (e.g., combinations of age, BMI, health indicator flags, and self-rated health) improve separation beyond a linear decision boundary. In this dataset, the main difficulty is consistently the *prediabetes* class: its feature profile overlaps substantially with both *no diabetes* and *diabetes*, so gains from additional model complexity are not guaranteed. The nonlinear models below are therefore interpreted as *risk stratification* tools, with emphasis on class-aware metrics (balanced accuracy and macro OvR ROC-AUC) rather than overall accuracy alone.

XGBoost with Random Under-Sampling. XGBoost (gradient-boosted decision trees) was used as a strong nonlinear baseline. To counter class imbalance, a RandomUnderSampler (1:1 auto) was applied during training to reduce majority-class dominance. The resulting models increased recall for the minority classes relative to a naive majority predictor, but performance remained constrained by the inherent overlap of prediabetes with adjacent classes. Validation and test results were broadly consistent, suggesting limited overfitting under the selected settings.

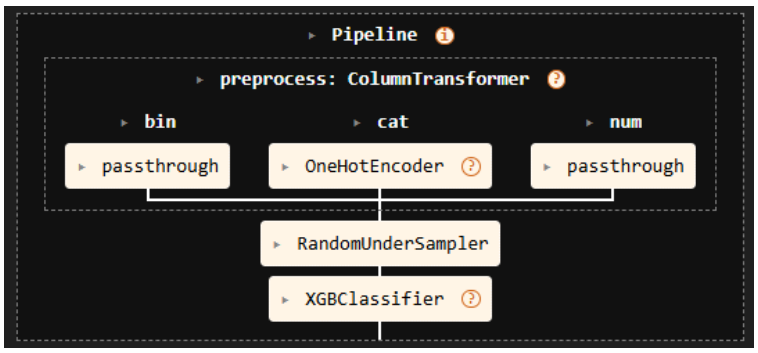


Figure 33: XGBoost evaluation summary (under-sampled training): diagnostic view used for comparison against the linear baseline.

Table 16: XGBoost + RandomUnderSampler (1:1 auto): classification report summary.

(a) Validation

Class	Precision	Recall	F1	Support
0	0.94	0.58	0.72	28,508
1	0.03	0.36	0.05	695
2	0.34	0.53	0.41	5,264
Accuracy			0.57	34,467
Macro avg	0.44	0.49	0.39	34,467
Weighted avg	0.83	0.57	0.66	34,467

Metric	Value
Balanced accuracy	0.4901
ROC-AUC (macro OvR)	0.7240

(b) Test

Class	Precision	Recall	F1	Support
0	0.94	0.58	0.72	28,509
1	0.03	0.38	0.06	694
2	0.33	0.52	0.40	5,265
Accuracy			0.57	34,468
Macro avg	0.43	0.49	0.39	34,468
Weighted avg	0.83	0.57	0.66	34,468

Metric	Value
Balanced accuracy	0.4929
ROC-AUC (macro OvR)	0.7229

RUSBoost. RUSBoost was tested as an imbalance-aware alternative that integrates repeated random under-sampling into the boosting procedure. The goal is to improve minority-class recall without synthetic oversampling. In practice, RUSBoost produced a similar overall pattern: stronger minority recall than an unadjusted classifier, but continued difficulty in cleanly isolating prediabetes due to feature overlap. Performance was stable between validation and test, reinforcing that the limiting factor is primarily separability in the available survey indicators rather than variance from training.

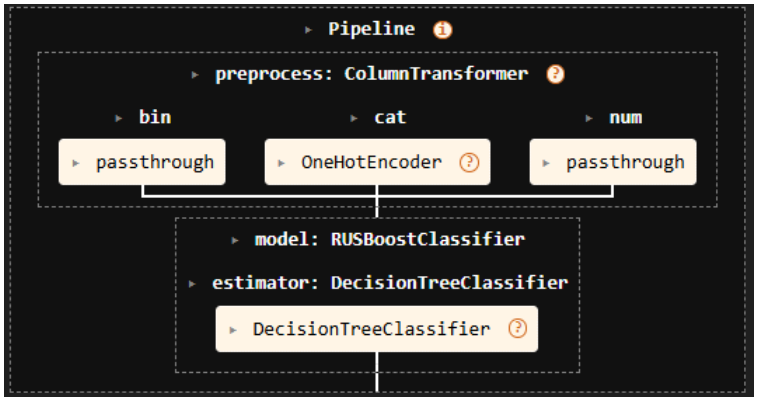


Figure 34: RUSBoost evaluation summary: diagnostic view for minority-sensitive performance comparison.

Table 17: RUSBoost: classification report summary.

(a) Validation

Class	Precision	Recall	F1	Support
0	0.94	0.64	0.76	28,508
1	0.03	0.23	0.05	695
2	0.34	0.60	0.44	5,264
Accuracy			0.63	34,467
Macro avg	0.44	0.49	0.42	34,467
Weighted avg	0.83	0.63	0.70	34,467

Metric	Value
Balanced accuracy	0.4898
ROC-AUC (macro OvR)	0.7320

(b) Test

Class	Precision	Recall	F1	Support
0	0.94	0.64	0.76	28,509
1	0.03	0.22	0.05	694
2	0.35	0.60	0.44	5,265
Accuracy			0.63	34,468
Macro avg	0.44	0.49	0.42	34,468
Weighted avg	0.83	0.63	0.70	34,468

Metric	Value
Balanced accuracy	0.4900
ROC-AUC (macro OvR)	0.7362

Interpretation for model selection. Across both nonlinear approaches, there were no significant gains over the linear models as they were still not able to resolve problems within the prediabetes class. Given the similar held-out performance, the linear model remains attractive as a primary reference: it is simpler to audit, easier to calibrate, and aligns naturally with the project goal of screening-oriented risk estimates rather than deterministic diagnosis. In other words, the limiting factor appears to be feature separability in the survey data, not a lack of nonlinear model capacity.

7.5 Model Comparison Summary and Decision Point

Across linear and nonlinear models (with multiple imbalance-handling strategies), results consistently indicate that the feature space contains *usable risk-ranking signal* but does not support *prediabetes* as a reliable hard class in a single 3-class decision rule: when prediabetes recall becomes non-zero, precision collapses, reflecting substantial overlap with both neighboring classes. Collapsing prediabetes into diabetes would blur clinically distinct states and add label noise, while dropping prediabetes would reduce the task to detecting existing diabetes and undermine the preventive screening goal. Accordingly, the modeling strategy shifts to a two-stage workflow that mirrors practice: (Stage 1) a sensitivity-first screening model predicts `AtRisk` (1 if `Diabetes_012` $\in \{1, 2\}$, else 0) and outputs calibrated probabilities with an explicit operating threshold; (Stage 2) a follow-on classifier is applied only to Stage-1 positives to stratify `Diabetes (2)` vs `Prediabetes (1)`, estimating severity without competing against the dominant no-diabetes class.

8 Two-Stage Modeling Strategy

8.1 Stage 1: At-Risk Screening (Binary)

Stage 1 converts the three-class task into a binary *screening* problem: flag respondents who are *at risk* (prediabetes or diabetes) versus *no diabetes*. This reduces the hardest ambiguity (prediabetes overlap) and supports a sensitive triage rule where false negatives are more costly than false positives.

Binary target definition. Using `Diabetes_012`, the label is set to `AtRisk=1` for values `1` (prediabetes) or `2` (diabetes), and `AtRisk=0` for `0` (no diabetes). Pooling creates a cleaner screening boundary while preserving the prevention-oriented goal of identifying elevated risk.

Defining the Target (At-Risk vs No Diabetes)

```
# Stage 1 target: At-Risk vs No Diabetes
# At-Risk = 1 if Diabetes_012 in {1,2}, else 0

y_train_risk = (y_train != 0).astype(int)
y_val_risk   = (y_val   != 0).astype(int)
y_test_risk  = (y_test  != 0).astype(int)

print("At-risk prevalence (train):", y_train_risk.mean().round(4))
print("At-risk prevalence (val):  ", y_val_risk.mean().round(4))
print("At-risk prevalence (test): ", y_test_risk.mean().round(4))
```

✓ 0.0s

```
At-risk prevalence (train): 0.1729
At-risk prevalence (val):   0.1729
At-risk prevalence (test):  0.1729
```

Figure 35: Stage 1 setup: pooled at-risk target and resulting class balance.

Modeling protocol and decision rule. A logistic regression baseline is first fit without imbalance correction to establish a lower bound. The final Stage 1 model follows a strict train/validation/test protocol: hyperparameters are tuned on validation data (Optuna), probability calibration is selected on validation diagnostics (Brier score and log loss), and a single screening threshold is chosen on calibrated validation probabilities using a balanced-accuracy sweep. The test set is held out until the end and is not used for tuning, calibration choice, or threshold selection.

```
    # Objective: mostly balanced accuracy, slight ROC-AUC tie-breaker
    score = bal_acc + 0.10 * roc

    # Store diagnostics
    trial.set_user_attr("bal_acc_val_at_0p5", float(bal_acc))
    trial.set_user_attr("roc_auc_val", float(roc))

    return score

study_s1 = optuna.create_study(direction="maximize")
study_s1.optimize(objective_stage1_bal_acc, n_trials=60, show_progress_bar=False)

best_s1 = study_s1.best_trial
print("Best score:", best_s1.value)
print("Best params:", best_s1.params)
print("VAL Balanced Acc @ 0.5:", best_s1.user_attrs["bal_acc_val_at_0p5"])
print("VAL ROC-AUC:", best_s1.user_attrs["roc_auc_val"])
```

Figure 36: Stage 1 Optuna Hyperparameter Tuning.

Evaluation emphasis (screening performance). Because the positive class is less prevalent, overall accuracy is not sufficient for screening evaluation. Stage 1 comparisons focus on (i) recall for the at-risk class (coverage), (ii) the precision–recall trade-off (follow-up burden), (iii) balanced accuracy (accounts for class imbalance), and (iv) ROC–AUC (threshold-independent ranking). The final model is retrained on Train+Val, recalibrated, and the threshold is re-selected to remain aligned with the updated probability scale before the one-time test evaluation.

```
[I 2026-02-15 18:36:02,705] Trial 57 finished with value: 0.8187370995074076 and parameters: {'C': 0.2198453775325886
[I 2026-02-15 18:36:03,039] Trial 58 finished with value: 0.8185267747243319 and parameters: {'C': 0.0138838594742436
[I 2026-02-15 18:36:04,019] Trial 59 finished with value: 0.8184917021493371 and parameters: {'C': 1.1038598992028648
Best score: 0.8192613621286076
Best params: {'C': 0.07945616658667748, 'solver': 'liblinear', 'fit_intercept': True, 'tol': 0.0006453172864536886}
VAL Balanced Acc @ 0.5: 0.738133963238295
VAL ROC-AUC: 0.8112739889031246
```

Figure 37: Stage 1 Best Parameters from Optuna.

```python
from sklearn.calibration import CalibratedClassifierCV
from sklearn.frozen import FrozenEstimator
from sklearn.metrics import brier_score_loss, log_loss

eps = 1e-12

# Freeze the already-refit tuned Stage-1 classifier
frozen_s1 = FrozenEstimator(risk_clf_best)

# Define calibrators (cv=None gives "prefit"-style behavior in newer sklearn)
cal_s1_sigmoid = CalibratedClassifierCV(
    estimator=frozen_s1,
    method="sigmoid",
    cv=None
)

cal_s1_isotonic = CalibratedClassifierCV(
    estimator=frozen_s1,
    method="isotonic",
    cv=None
)

# Fit calibration mappings on VAL ONLY
cal_s1_sigmoid.fit(X_val, y_val_risk)
cal_s1_isotonic.fit(X_val, y_val_risk)

# Calibrated probabilities on VAL
p_val_risk_sig = cal_s1_sigmoid.predict_proba(X_val)[:, 1]
p_val_risk_iso = cal_s1_isotonic.predict_proba(X_val)[:, 1]

# Calibrated probabilities on TEST (store only; do not tune on TEST)
p_test_risk_sig = cal_s1_sigmoid.predict_proba(X_test)[:, 1]
p_test_risk_iso = cal_s1_isotonic.predict_proba(X_test)[:, 1]

# Calibration diagnostics (VAL)
def calib_metrics(y, p, name):
    p = np.clip(p, eps, 1 - eps)
    return {
        "model": name,
        "brier": brier_score_loss(y, p),
        "log_loss": log_loss(y, p),
        "mean_p": float(np.mean(p)),
    }

(
    calib_metrics(y_val_risk, p_val_risk_raw, "raw"),
    calib_metrics(y_val_risk, p_val_risk_sig, "sigmoid"),
    calib_metrics(y_val_risk, p_val_risk_iso, "isotonic"),
)
```

Figure 38: Stage 1 calibrating probabilities.

```python
thresholds = np.linspace(0.0, 1.0, 2001)

scores = []
for t in thresholds:
    y_pred = (p_val_risk_iso >= t).astype(int)
    scores.append(balanced_accuracy_score(y_val_risk, y_pred))

best_i = int(np.argmax(scores))
best_threshold_s1 = float(thresholds[best_i])
best_bal_acc_s1 = float(scores[best_i])

print(f"Chosen threshold (Stage 1): {best_threshold_s1:.4f}")
print(f"Balanced accuracy @ threshold: {best_bal_acc_s1:.3f}")
```

✓ 3.0s

```
Chosen threshold (Stage 1): 0.1705
Balanced accuracy @ threshold: 0.739
```

Figure 39: Stage 1 threshold selection

Table 18: Final Stage 1 (Test) performance after tuning, isotonic calibration, and threshold re-selection.

Class	Precision	Recall	F1-score	Support
0 (Not at-risk)	0.94	0.70	0.80	28,509
1 (At-risk)	0.35	0.77	0.48	5,959
Accuracy			0.72	34,468
Macro avg	0.64	0.74	0.64	34,468
Weighted avg	0.84	0.72	0.75	34,468
Threshold (Stage 1)				0.1770
Balanced accuracy				0.7377
ROC–AUC				0.8101
Confusion matrix			$\begin{bmatrix} 20054 & 8455 \\ 1359 & 4600 \end{bmatrix}$	

Interpretation and link to Stage 2. Stage 1 provides a calibrated risk score plus a fixed decision rule for triage, not diagnosis. Respondents flagged as at-risk are passed to Stage 2, where the model focuses only on distinguishing diabetes versus prediabetes without competing against the dominant no-diabetes class.

8.2 Stage 2: Diabetes vs Prediabetes Stratification

Stage 2 is applied only to respondents who are already in the *at-risk* group and focuses on the harder boundary: distinguishing *diabetes* from *prediabetes*. Removing the dominant *no diabetes* class concentrates model capacity on the diabetes–prediabetes separation and makes the task operationally closer to follow-up triage rather than population-wide diagnosis.

Stage 2 population and label. Stage 2 filters each split to rows where `Diabetes_012` is `1` (prediabetes) or `2` (diabetes). The binary label is then recoded so `1` corresponds to diabetes and `0` corresponds to prediabetes. This produces a focused classifier that estimates severity among already-elevated risk cases, without competing against the majority class from the original dataset.

```
def subset_stage2(X, y):
    mask = (y != 0)                 # keep only rows where y in {1,2}
    X_s2 = X.loc[mask]              # subset features to those rows
    y_s2 = (y.loc[mask] == 2).astype(int)  # diabetes=1, prediabetes=0
    return X_s2, y_s2, mask

X_train_s2, y_train_s2, _ = subset_stage2(X_train, y_train)
X_val_s2,   y_val_s2,   _ = subset_stage2(X_val, y_val)
X_test_s2,  y_test_s2,  _ = subset_stage2(X_test, y_test)

print("Stage-2 prevalence (train) diabetes:", y_train_s2.mean())
print("Stage-2 prevalence (val)   diabetes:", y_val_s2.mean())
print("Stage-2 prevalence (test)  diabetes:", y_test_s2.mean())
print("Stage-2 sample sizes:", len(y_train_s2), len(y_val_s2), len(y_test_s2))
```

✓ 0.0s

```
Stage-2 prevalence (train) diabetes: 0.8834867663981588
Stage-2 prevalence (val)   diabetes: 0.8833696929014936
Stage-2 prevalence (test)  diabetes: 0.8835375062930022
Stage-2 sample sizes: 27808 5959 5959
```

Figure 40: Stage 2 setup: restricting to the at-risk cohort and defining the diabetes vs. prediabetes target.

Modeling protocol and thresholding. Stage 2 follows the same strict protocol as Stage 1: an unweighted logistic regression baseline is established first, then imbalance is handled with Random Under-Sampling (1:1) inside the training pipeline. Hyperparameters are tuned with Optuna on validation performance, probabilities are calibrated (sigmoid vs isotonic) using validation-only fits, and a decision threshold is selected by sweeping thresholds to maximize balanced accuracy on calibrated validation probabilities. The test set is held out until the final one-time evaluation.

```
    # Objective: mostly balanced accuracy, slight ROC-AUC tie-breaker
    score = bal_acc + 0.10 * roc

    # Store diagnostics
    trial.set_user_attr("bal_acc_val_at_0p5", float(bal_acc))
    trial.set_user_attr("roc_auc_val", float(roc))

    return score

# Run the study
study_s2 = optuna.create_study(direction="maximize")
study_s2.optimize(objective_s2, n_trials=60, show_progress_bar=False)

best_s2 = study_s2.best_trial
print("Best score:", best_s2.value)
print("Best params:", best_s2.params)
print("VAL Balanced Acc @ 0.5:", best_s2.user_attrs["bal_acc_val_at_0p5"])
print("VAL ROC-AUC:", best_s2.user_attrs["roc_auc_val"])
```

Figure 41: Stage 2 Optuna Study

Evaluation focus Because diabetes is still the majority class within the at-risk cohort, accuracy alone is not sufficient. Reporting emphasizes balanced accuracy, ROC–AUC (ranking), and the class-wise precision–recall trade-off, with particular attention to diabetes recall (missed diabetes

cases) versus the operational burden created by false positives.

```
[I 2026-02-15 18:36:16,346] Trial 56 finished with value: 0.6541080722048501 and parameters: {'C': 0.0025772850432047
[I 2026-02-15 18:36:16,394] Trial 57 finished with value: 0.6615097253504187 and parameters: {'C': 0.0342756529181731
[I 2026-02-15 18:36:16,437] Trial 58 finished with value: 0.6605713575036628 and parameters: {'C': 0.0793475158728055
[I 2026-02-15 18:36:16,496] Trial 59 finished with value: 0.6597949694955282 and parameters: {'C': 0.1464251565389769
Best score: 0.6644054088036562
Best params: {'C': 0.01488502895804253, 'solver': 'liblinear', 'fit_intercept': False, 'tol': 0.0003086309204002868}
VAL Balanced Acc @ 0.5: 0.6018863025081455
VAL ROC-AUC: 0.6251910629551071
```

Figure 42: Stage 2 Best Parameters from Optuna.

```python
eps = 1e-12

# Freeze the already-refit tuned Stage-2 classifier
frozen_s2 = FrozenEstimator(s2_clf_best)

# Define calibrators (cv=None = "prefit-style" in newer sklearn)
cal_s2_sigmoid = CalibratedClassifierCV(
    estimator=frozen_s2,
    method="sigmoid",
    cv=None
)

cal_s2_isotonic = CalibratedClassifierCV(
    estimator=frozen_s2,
    method="isotonic",
    cv=None
)

# Fit calibration mappings on Stage-2 VAL ONLY
cal_s2_sigmoid.fit(X_val_s2, y_val_s2)
cal_s2_isotonic.fit(X_val_s2, y_val_s2)

# Calibrated probabilities on Stage-2 VAL
p_val_s2_sig = cal_s2_sigmoid.predict_proba(X_val_s2)[:, 1]
p_val_s2_iso = cal_s2_isotonic.predict_proba(X_val_s2)[:, 1]

# Calibrated probabilities on Stage-2 TEST (store only; don't optimize on TEST)
p_test_s2_sig = cal_s2_sigmoid.predict_proba(X_test_s2)[:, 1]
p_test_s2_iso = cal_s2_isotonic.predict_proba(X_test_s2)[:, 1]

def calib_metrics(y, p, name):
    p = np.clip(p, eps, 1 - eps)
    return {
        "model": name,
        "brier": brier_score_loss(y, p),
        "log_loss": log_loss(y, p),
        "mean_p": float(np.mean(p)),
    }

(
    calib_metrics(y_val_s2, p_val_s2_raw, "raw"),
    calib_metrics(y_val_s2, p_val_s2_sig, "sigmoid"),
    calib_metrics(y_val_s2, p_val_s2_iso, "isotonic"),
)
```

Figure 43: Stage 2 probability calibration for diabetes vs. prediabetes.

Role in the two-stage system. Stage 2 errors are conditional: it only refines labels for cases that Stage 1 already flagged as at-risk. In contrast, Stage 1 false negatives are irrecoverable because those cases never reach Stage 2. This reinforces the screening-first design: Stage 1 prioritizes coverage, and Stage 2 adds severity structure among flagged cases.

```python
thresholds = np.linspace(0.0, 1.0, 2001)

scores = []
for t in thresholds:
    y_hat = (p_val_s2_cal >= t).astype(int)
    scores.append(balanced_accuracy_score(y_val_s2, y_hat))

best_i = int(np.argmax(scores))
best_threshold_s2 = float(thresholds[best_i])
best_bal_acc_s2 = float(scores[best_i])

print(f"Chosen threshold (Stage 2): {best_threshold_s2:.4f}")
print(f"Balanced accuracy @ threshold: {best_bal_acc_s2:.3f}")
```

✓ 1.3s

```
Chosen threshold (Stage 2): 0.8615
Balanced accuracy @ threshold: 0.606
```

Figure 44: Stage 2 Threshold decisions

Table 19: Final Stage-2 (Test) performance after threshold selection.

Class	Precision	Recall	F1-score	Support
0	0.16	0.58	0.25	694
1	0.92	0.60	0.73	5,265
Accuracy			0.60	5,959
Macro avg	0.54	0.59	0.49	5,959
Weighted avg	0.83	0.60	0.67	5,959
Threshold (Stage 2)				0.8825
Balanced accuracy				0.5896
ROC–AUC				0.6246
Confusion matrix			$\begin{bmatrix} 400 & 294 \\ 2091 & 3174 \end{bmatrix}$	

8.3 Tuning, Calibration, Threshold Selection

Both stages follow the same evaluation discipline: tuning and all decision-setting are performed on the validation split, and the test split is used once for final reporting. Hyperparameters are tuned with Optuna by fitting candidate pipelines on the training data and selecting the best configuration on validation performance.

Calibration. Because outputs are used as *risk scores* for triage, predicted probabilities are calibrated after tuning. Sigmoid and isotonic calibration are compared on the validation set using proper scoring rules (Brier score and log loss) and calibration diagnostics; the calibration method that best matches validation reliability is then fixed for the final pipeline.

Threshold selection and re-selection. After calibration, each stage's decision threshold is chosen by sweeping candidate thresholds on the calibrated validation probabilities and selecting the threshold that maximises balanced accuracy (used to reduce the influence of class imbalance). The final model is then retrained on Train+Validation using the selected hyperparameters, re-calibrated, and the threshold is *re-selected* on the updated probability scale before the one-time test evaluation. This re-selection step is required because calibration and refitting can shift the probability distribution, so the earlier threshold may no longer correspond to the same operating point.

Operational emphasis. Stage 1 thresholding prioritises sensitivity to avoid missed at-risk cases (irrecoverable under hard gating), while Stage 2 thresholding prioritises within-cohort discrimination between diabetes and prediabetes under a different base rate.

8.4 Final 3-Class Assembly and Evaluation

The final predictor composes the two calibrated, thresholded binary stages into a single three-class output on the full test set. Stage 1 first flags *at-risk* cases (prediabetes or diabetes) versus *no diabetes*. Stage 2 is then applied only to the Stage 1 positives to stratify *prediabetes* versus *diabetes*. This *hard-gated* design makes Stage 1 errors irrecoverable: any missed at-risk case is permanently assigned to class 0.

```
# Stage 1: calibrated risk probability on ALL test rows
p_risk_test = p_test_risk_final  # already computed: final_cal_s1.predict_proba(X_test)[:, 1]
at_risk_hat = (p_risk_test >= FINAL_THRESHOLD_S1).astype(int)  # 1=at-risk, 0=no diabetes

# Default prediction: class 0 (no diabetes)
final_pred_3class = np.zeros(len(at_risk_hat), dtype=int)

# Stage 2 only for those predicted at-risk by Stage 1
idx_at_risk = np.where(at_risk_hat == 1)[0]

if idx_at_risk.size > 0:
    X_test_at_risk = X_test.iloc[idx_at_risk]  # keep DF for ColumnTransformer
    p_diab_at_risk = final_cal_s2.predict_proba(X_test_at_risk)[:, 1]  # P(diabetes | at-risk)
    diabetes_hat = (p_diab_at_risk >= FINAL_THRESHOLD_S2).astype(int)  # 1=diabetes, 0=prediabetes

    # Map to multiclass: 1=prediabetes, 2=diabetes
    final_pred_3class[idx_at_risk] = np.where(diabetes_hat == 1, 2, 1)

print("Two-stage (tuned+calibrated) HARD-GATED vs TRUE multiclass labels (TEST)")
print(classification_report(y_test, final_pred_3class, digits=3))
print("Confusion matrix:")
print(confusion_matrix(y_test, final_pred_3class))
print("Balanced accuracy (3-class):", balanced_accuracy_score(y_test, final_pred_3class))
```

Figure 45: Hard-gated two-stage assembly: Stage 1 screens at-risk cases; Stage 2 stratifies diabetes vs. prediabetes within the screened subset.

Table 20: Two-stage (tuned + calibrated) hard-gated model vs. true three-class labels (Test).

Class	Precision	Recall	F1-score	Support
0 (No Diabetes)	0.936	0.709	0.807	28,509
1 (Prediabetes)	0.037	0.232	0.064	694
2 (Diabetes)	0.363	0.590	0.450	5,265
Accuracy			0.681	34,468
Macro avg	0.445	0.510	0.440	34,468
Weighted avg	0.830	0.681	0.737	34,468
Balanced accuracy (3-class)				0.5101
Confusion matrix		$\begin{bmatrix} 20204 & 3149 & 5156 \\ 248 & 161 & 285 \\ 1140 & 1020 & 3105 \end{bmatrix}$		

Relative to a standard single-model three-class baseline, the two-stage system improves recall for *diabetes* and *no diabetes*, but reduces *prediabetes* recall. This reflects error compounding (Stage 1 gating) and the within-cohort imbalance in Stage 2, which pulls the boundary toward diabetes. Overall, the two-stage pipeline is better suited to diabetes-focused screening and clean identification of class 0, while the single three-class model remains more favourable when prioritising prediabetes sensitivity.

9 Model Interpretation & Explainability

9.1 Global Explanation: Logistic Regression Coefficients

For the multinomial logistic regression, fitted coefficients provide a direct global explanation in the transformed feature space: for each class, positive coefficients increase the log-odds of predicting that class relative to the others, while negative coefficients decrease them. Figure 46 shows the top coefficients (by absolute magnitude) per class.

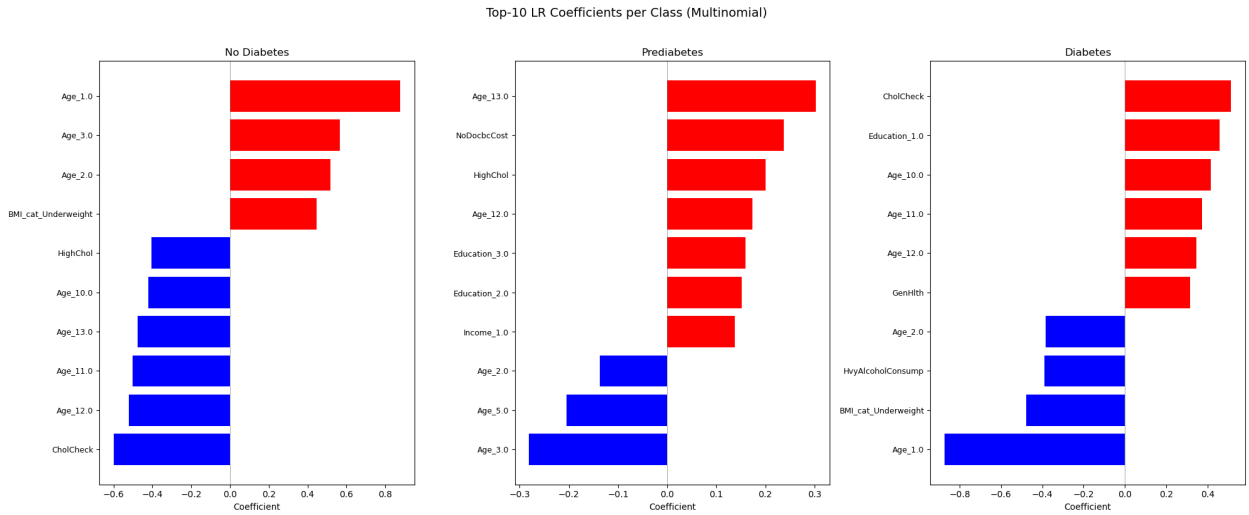


Figure 46: Top-10 multinomial logistic regression coefficients per class (signed global effects).

The coefficient patterns align with medically plausible risk structure. The *No Diabetes* class is supported by younger age bands and low-BMI categories, while older age bands and cardiometabolic indicators reduce its odds. *Prediabetes* loads more strongly on older age and socioeconomic/access signals (e.g., cost barriers and lower education), consistent with a mixed detection-risk signature rather than a clean clinical boundary. *Diabetes* shows the strongest positive loadings on cardiometabolic and health-status indicators (e.g., cholesterol check, poorer self-rated health, and older age), with younger age and low-BMI categories pushing predictions away from diabetes.

9.2 Global Validation: Permutation Importance

Signed coefficients provide a global explanation but can be sensitive to correlated predictors and encoding choices. As a model-agnostic cross-check, permutation importance measures the drop in validation balanced accuracy when each feature is randomly shuffled; larger drops indicate stronger reliance on that feature. Figure 47 and Table 21 show the top features and their stability across repeats.

Table 21: Top-15 permutation importances on validation (Δ balanced accuracy; mean ± std).

Feature	Importance mean	Importance std
GenHlth	0.047951	0.004759
Age	0.028717	0.004640
HighChol	0.018821	0.002389
HighBP	0.017311	0.001710
BMI_cat	0.011568	0.002027
BMI	0.009178	0.002313
CholCheck	0.004251	0.001298
NoDocbcCost	0.004032	0.001761
PhysHlth_14plus	0.003574	0.002467
DiffWalk	0.003091	0.001376
MentHlth_gt0	0.002642	0.002557
PhysActivity	0.002549	0.001681
HvyAlcoholConsump	0.002246	0.002123
MentHlth_14plus	0.002165	0.001033
HeartDiseaseorAttack	0.001959	0.002173

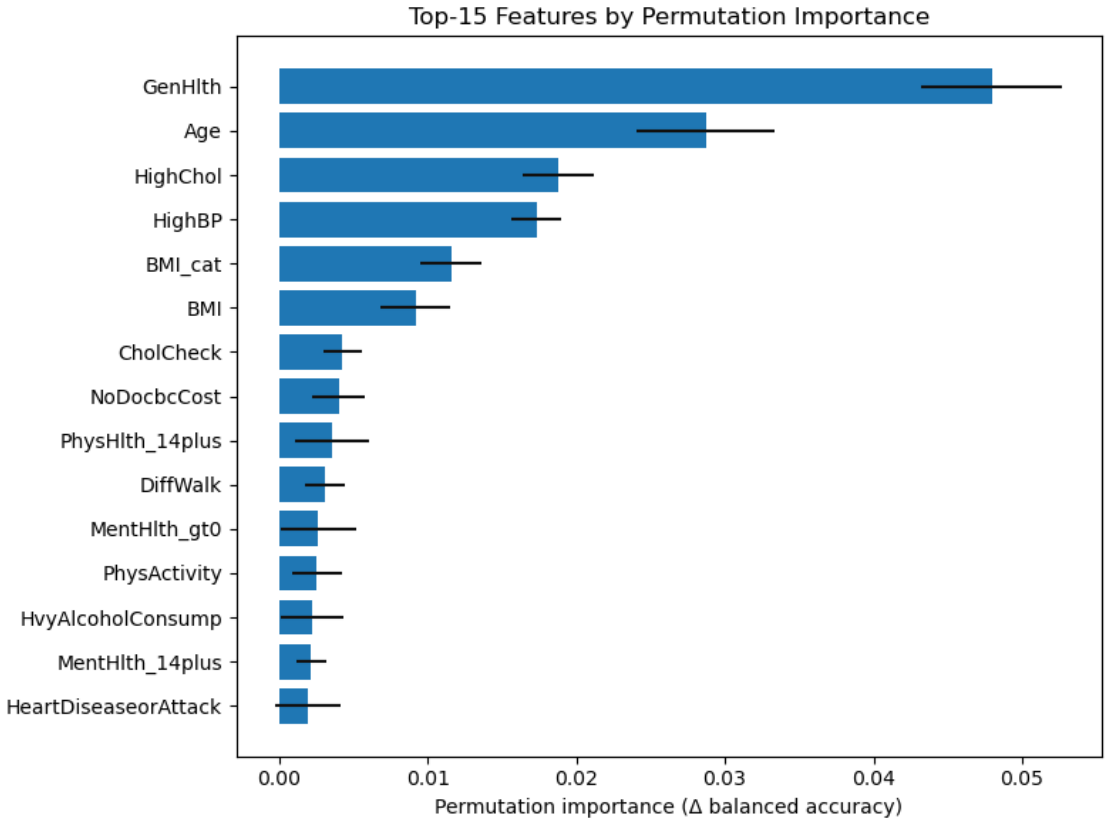


Figure 47: Top permutation importances (validation), measured as Δ balanced accuracy under feature shuffling (mean ± std over repeats).

Permutation importance confirms that `GenHlth` is the dominant signal, followed by `Age` and cardiometabolic risk factors (`HighChol`, `HighBP`, and BMI-related features). The relatively small standard deviations across repeats indicate the ranking is stable rather than a shuffle artifact.

9.3 SHAP (Global + Local)

SHAP explains predictions by assigning each feature an additive contribution (Shapley value) such that contributions sum to the difference between the model output and a baseline expectation. Global SHAP summaries show how features shift predictions across many respondents, while local waterfalls decompose a single case into its main drivers.

Global (beeswarm). Across classes, SHAP confirms the same hierarchy seen in EDA and permutation importance: `GenHlth` and age dominate, followed by cardiometabolic indicators (`HighBP`, `HighChol`, and BMI-related features). The *Prediabetes* class shows smaller, more diffuse effects and relatively greater prominence of socioeconomic and access signals (e.g., `NoDocbcCost`, education, income), consistent with overlap and weaker separability.

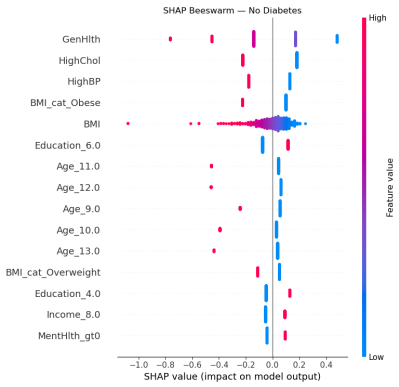


Figure 48: No Diabetes

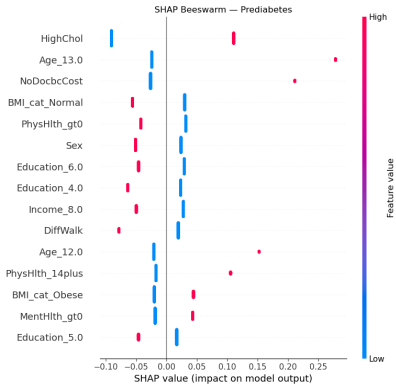


Figure 49: Prediabetes

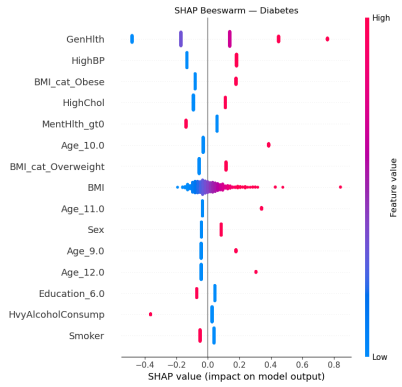


Figure 50: Diabetes

Figure 51: SHAP beeswarm plots (validation sample): distribution of feature contributions for each class. Points to the right increase the probability of the class; colour reflects feature value (low to high).

Local (waterfall). Local SHAP explanations remain consistent with the global view and make misclassifications interpretable. In the illustrated case (true label: No Diabetes), poor self-rated health and cardiometabolic indicators push the prediction toward diabetes, highlighting genuine class overlap in survey space and supporting the use of calibrated probabilities for risk ranking rather than hard diagnosis.

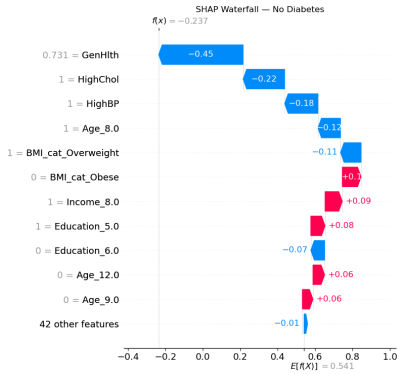


Figure 52: No Diabetes

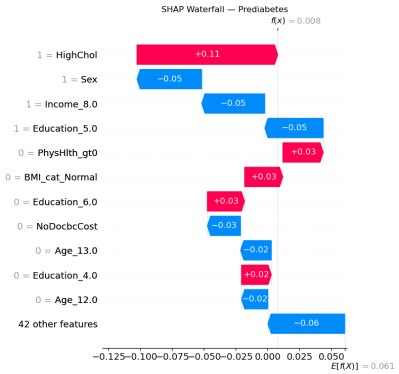


Figure 53: Prediabetes

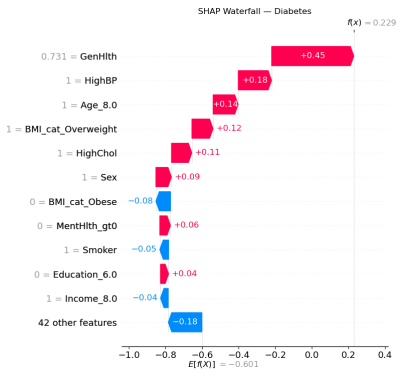


Figure 54: Diabetes

Figure 55: SHAP waterfall plots for one validation instance: additive feature contributions relative to a baseline prediction, shown for each class.

9.4 LIME (Local Explanation)

LIME provides a local explanation by fitting a simple surrogate model around a single respondent using perturbed samples and proximity weights. Because the raw inputs include many encoded categorical indicators, LIME is applied in the model's transformed feature space (after preprocessing), matching the representation used for SHAP.

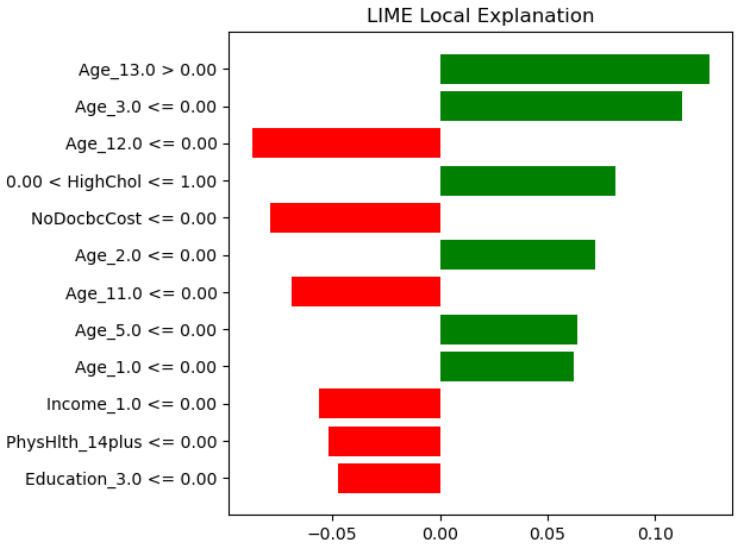


Figure 56: LIME local explanation for one validation instance (true label: No Diabetes). Positive bars support the predicted class; negative bars oppose it.

Table 22: LIME feature contributions for the explained instance (local surrogate weights).

Feature condition	Contribution
Age_13.0 > 0.00	+0.1250
Age_3.0 ≤ 0.00	+0.1124
Age_12.0 ≤ 0.00	-0.0873
0.00 < HighChol ≤ 1.00	+0.0813
NoDocbcCost ≤ 0.00	-0.0785
Age_2.0 ≤ 0.00	+0.0722
Age_11.0 ≤ 0.00	-0.0688
Age_5.0 ≤ 0.00	+0.0638
Age_1.0 ≤ 0.00	+0.0624
Income_1.0 ≤ 0.00	-0.0563
PhysHlth_14plus ≤ 0.00	-0.0518
Education_3.0 ≤ 0.00	-0.0474

For this instance, LIME highlights the same driver groups as SHAP (age-band indicators, cholesterol status, and access/socioeconomic signals), providing cross-method confirmation that the local attributions are robust rather than an artefact of a single explainer.

9.5 Connecting Explanations to EDA and Clustering

The explainability results are consistent with the earlier descriptive and unsupervised findings. The strongest EDA signals (self-rated general health, age, hypertension, high cholesterol, and BMI) also dominate global importance measures and SHAP summaries, indicating that the classifier relies on the most stable risk indicators rather than spurious patterns. The cluster structure shows a similar separation between (i) cardiometabolic and health-burden profiles and (ii) sociodemographic and access-to-care profiles; the supervised models reflect this split, with diabetes driven primarily by health-status and cardiometabolic features, while prediabetes is more strongly associated with socioeconomic and care-access signals (e.g., cost barriers, lower income/education). Local explanations (SHAP/LIME) draw on the same feature groups, and even errors are interpretable (e.g., poor self-rated health with hypertension/high cholesterol pushing predictions toward diabetes). Overall, the alignment across EDA, clustering, and explanations supports a coherent risk-stratification narrative, while reinforcing that prediabetes remains the most overlapping class but carries a meaningful socioeconomic-access signature.

10 Ethical, Practical & Methodological Reflection

This project uses a large national health survey dataset to study diabetes risk patterns and to build predictive models that can support public health planning. Survey responses are not clinical measurements, and observational data can reflect social structure as much as physiology. The goal is therefore risk understanding and population-level prioritization, not individual diagnosis.

Bias in survey-based data. Because the BRFSS-derived features are self-reported, measurement error is expected (recall bias, rounding, and heterogeneous interpretation of questions such as general health or physical activity). Selection effects may also be present, and the dataset version used here does not include the full survey design/weighting information needed for population-representative estimation. Results are interpreted as correlational signals useful for screening and targeting, not as causal effects.

Socioeconomic proxies and fairness risks. Variables such as `Income`, `Education`, and cost-barrier indicators can act as proxies for socioeconomic position. They carry information about risk because structural conditions shape prevention access and timely diagnosis, but naïve deployment can encode inequities into automated decisions. In this report these features are treated as explanatory signals to characterize need; downstream use should be aligned to supportive actions (outreach, screening offers, access improvements), not punitive or exclusionary decisions.

Risk of stigmatization. Risk models can unintentionally stigmatize individuals or communities by implying blame for outcomes. Many strong predictors (age, general health, comorbidity burden, and socioeconomic context) are not simply matters of personal choice, so interpretations focus on patterns and intervention opportunities rather than moral judgment.

Limits of prediction. Predicted labels and probabilities are not equivalent to diagnosis: the target is self-reported diabetes status and inputs are survey indicators rather than clinical lab values. Outputs should be used to prioritize follow-up and confirmatory testing, not to replace clinical assessment. This limitation is most acute for *prediabetes*, which overlaps with both no diabetes and diabetes, motivating calibration and careful threshold selection.

Explicit choices and what more data could enable. Key choices were made to keep the work transparent and reproducible: exact duplicate rows were removed, multiple model families were compared, class imbalance was handled explicitly, and explainability checks were used to confirm reliance on plausible drivers (e.g., `GenHlth`, age, blood pressure, cholesterol, BMI). The analysis does not claim causal effects, does not provide medical advice, and does not present the model as a deployable clinical test; a full fairness audit was also out of scope without stronger subgroup and survey-design support. With richer clinical data, prediction and interpretation would be stronger, and population estimates could be made more defensible with access to the original survey design and weights.

11 Conclusions & Public Health Implications

This analysis combined exploratory profiling, unsupervised clustering, calibrated multi-class classification, and model interpretation to study diabetes risk patterns in a large survey dataset. Survey indicators contain sufficient signal for *population-level* risk stratification: separation is strongest

between *Diabetes* and *No Diabetes*, while *Prediabetes* remains the most ambiguous class and behaves like a continuum between the two. Results should be interpreted as *screening support*, not diagnosis, since both inputs and targets are self-reported rather than clinical tests.

Risk signals were consistent across descriptive analysis and explainability methods. Higher diabetes risk was most strongly associated with poorer self-rated health, older age groups, hypertension, high cholesterol, and higher BMI categories, which also defined higher-burden clusters. Prediabetes showed a distinct secondary signature: socioeconomic vulnerability and care-access barriers (e.g., inability to see a doctor due to cost) appeared more frequently as differentiators, suggesting that elevated survey-reported risk may reflect both underlying health risk and unequal access to prevention and early detection.

Actionable implications are: prioritize outreach for respondents reporting poor general health plus cardiometabolic indicators; treat Prediabetes predictions as an early-warning category that triggers retesting and supportive prevention actions rather than a definitive label; pair prevention with access improvements in groups flagged by socioeconomic and cost-barrier signals; and communicate outputs as calibrated probabilities with transparent explanations to reflect uncertainty.

12 Reproducibility & Limitations

Reproducibility was supported by using fixed random seeds for train/validation/test splitting and models etc. Final model configurations were stored (e.g., as a fixed parameter set) so the selected pipelines can be reconstructed without rerunning full hyperparameter searches. Cluster robustness was checked by rerunning K-means across multiple seeds and reporting agreement, reducing the chance that results reflect a single initialization. Minor run-to-run differences can still occur due to floating-point effects, multi-threaded linear algebra, and library version differences; consistent package versions and execution settings are therefore important for strict replication.

Computation was a practical constraint, particularly for unsupervised learning. The high-dimensional, mostly binary/one-hot feature space increases the cost and fragility of distance-based methods and diagnostics (e.g., silhouette), and iterative procedures (K sweeps, stability reruns, DBSCAN parameter sweeps) multiply runtime. Feasibility required bounded grids for parameter sweeps, reduced representations for visualization (e.g., PCA projections), and a limited number of stability seeds. Interpretation tooling (permutation importance, SHAP, LIME) also adds overhead because it relies on repeated evaluations and sampling.

Prediabetes is intrinsically ambiguous in this dataset, overlapping with both adjacent classes and reflecting detection and access-to-care patterns, so lower separability is expected. Many predictors are binary or coarse ordinal categories, which can blur cluster boundaries and compress distinct profiles into similar indicator patterns. Finally, findings may be specific to the survey design and time period; without external validation (and without full survey design adjustments), transfer to other populations should not be assumed. Feature importance and model associations are correlational and should not be interpreted as causal levers without additional study.

References

abdelazizsami (2026). *CDC Diabetes Health Indicators*. Kaggle. URL: https://www.kaggle.com/datasets/abdelazizsami/cdc-diabetes-health-indicators (visited on 02/16/2026).

Centers for Disease Control and Prevention (2026). *BRFSS Annual Survey Data: 2015*. Centers for Disease Control and Prevention. URL: https://www.cdc.gov/brfss/annual_data/annual_2015.html (visited on 02/16/2026).

DataCamp (2026). *DBSCAN Clustering Algorithm: A Complete Guide*. DataCamp. URL: https://www.datacamp.com/tutorial/dbscan-clustering-algorithm (visited on 02/16/2026).

imbalanced-learn developers (2026). *imbalanced-learn documentation*. imbalanced-learn. URL: https://imbalanced-learn.org/stable/ (visited on 02/16/2026).

scikit-learn developers (2026). *scikit-learn: Machine Learning in Python*. scikit-learn. URL: https://scikit-learn.org/stable/ (visited on 02/16/2026).

SHAP developers (2026). *SHAP documentation*. Read the Docs. URL: https://shap.readthedocs.io/en/latest/ (visited on 02/16/2026).

World Health Organization (2026). *Social determinants of health*. World Health Organization. URL: https://www.who.int/health-topics/social-determinants-of-health#tab=tab_1 (visited on 02/16/2026).

A Additional Plots

A.1 Diabetes Prevalence by Age and BMI: Denominators

These figures provide the respondent counts and diabetes case counts underlying the prevalence curves shown in the main EDA. They are included here to keep the main narrative focused while retaining denominator context for interpretation.

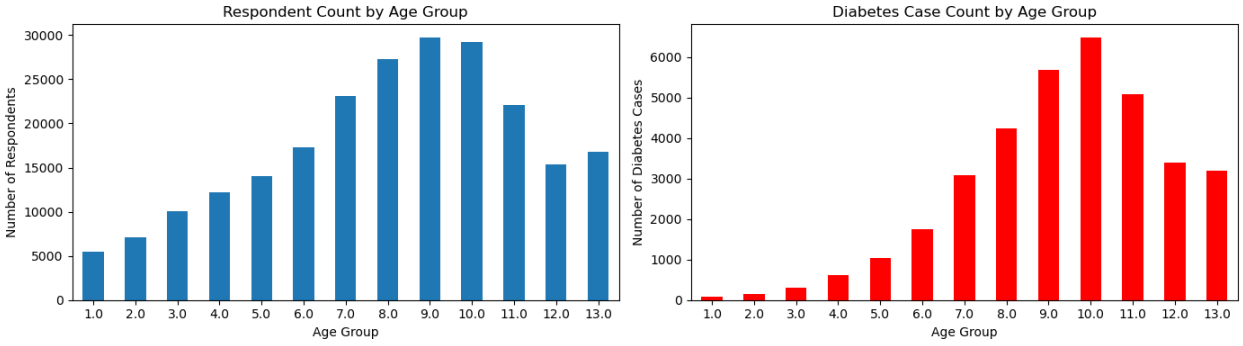


Figure 57: Respondent count and diabetes case count by age group.

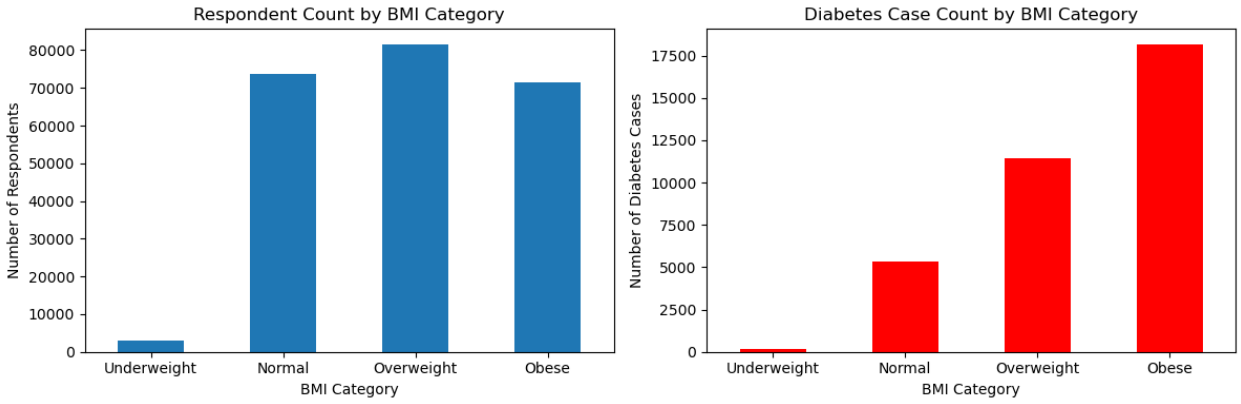


Figure 58: Respondent count and diabetes case count by BMI category.

A.2 Correlation, Redundancy, and Multivariate Checks: Additional Views

This subsection provides supporting multivariate views that complement the main EDA results. The plots summarize an Age×BMI prevalence gradient, selected interactions involving `HighBP`/`DiffWalk`/`GenHlth`, and an age-standardized check for `HvyAlcoholConsump`. These figures are descriptive co-patterns intended to support interpretation, not causal claims.

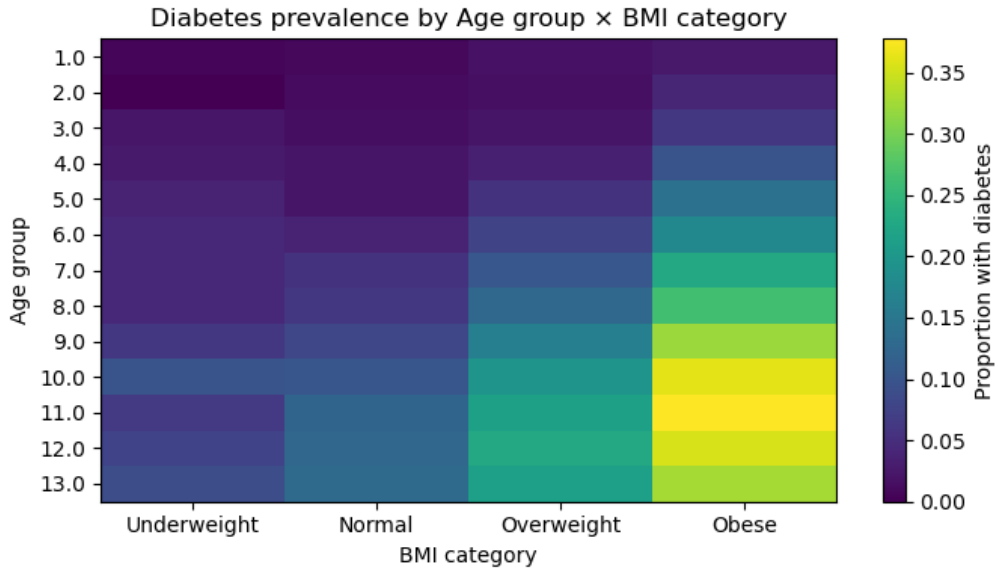
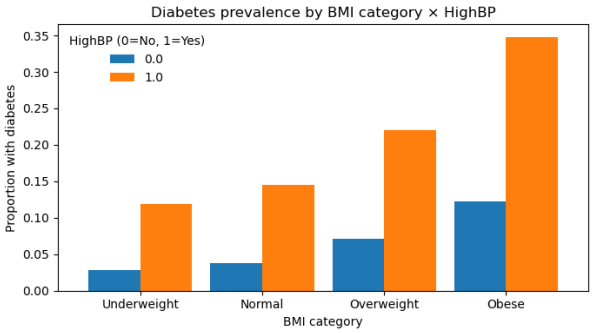
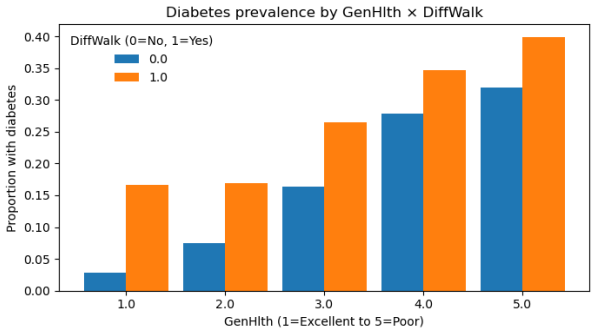


Figure 59: Additional multivariate view: Age×BMI prevalence gradient (supporting diagnostic).



(a) Interaction view (additional diagnostic).



(b) Interaction view (additional diagnostic).

Figure 60: Additional interaction views involving burden/comorbidity indicators (supporting diagnostics).

Table 23: Age-standardized diabetes prevalence by heavy alcohol consumption.

`HvyAlcoholConsump`	**Std. prevalence**
0 (No)	0.158224
1 (Yes)	0.062896

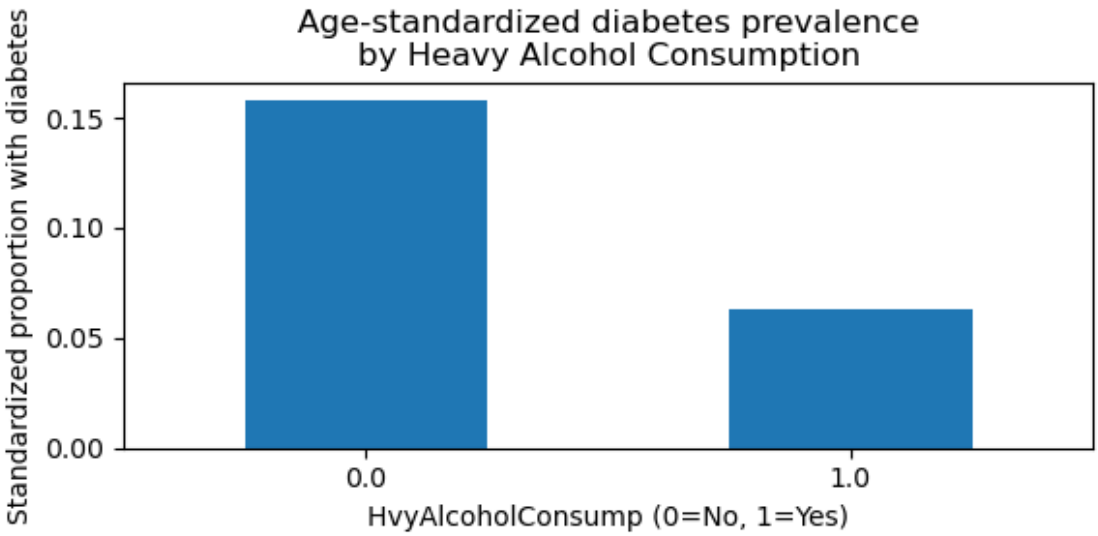


Figure 61: Additional multivariate diagnostic view (supporting check).

Table 24: Within-strata prevalence difference (Heavy=1 minus Heavy=0) by Age group and BMI category.

HvyAlcoholConsump	Age	BMI_cat	diff_1_minus_0
0	1.0	Underweight	-0.006173
1	1.0	Normal	-0.005266
2	1.0	Overweight	0.001620
3	1.0	Obese	-0.027495
4	2.0	Underweight	0.000000
5	2.0	Normal	-0.012079
6	2.0	Overweight	-0.003765
7	2.0	Obese	-0.024211
8	3.0	Underweight	-0.024000
9	3.0	Normal	0.013962

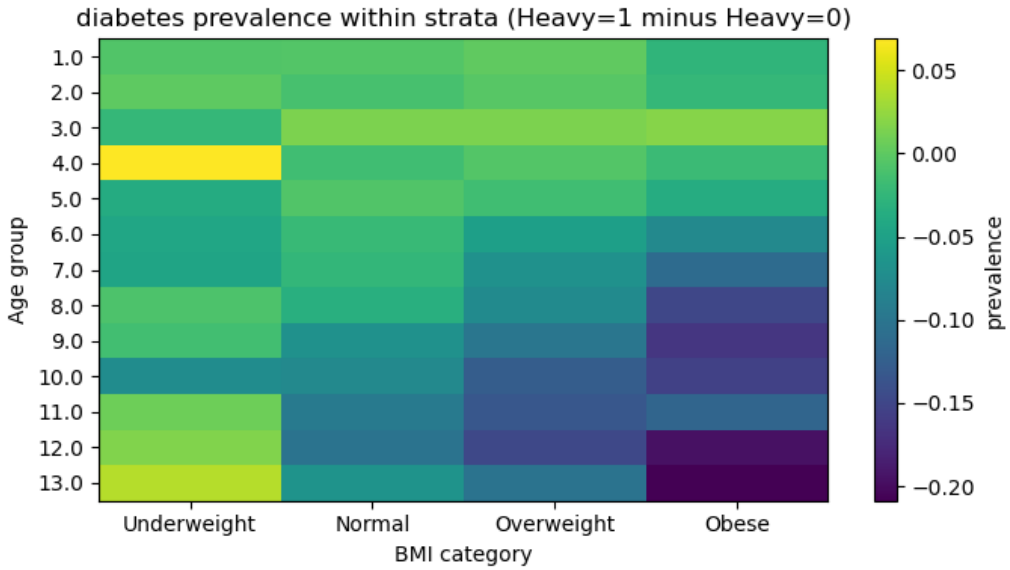


Figure 62: Additional multivariate diagnostic view (supporting check).

Table 25: Class-wise association for HvyAlcoholConsump (one-vs-rest view).

Outcome class	Association
class_0	0.718360
class_1	-0.159417
class_2	-0.807899