HOMEWORK 3 – FUNDAMENTALS OF MACHINE LEARNING (CS-UA 473)

ANDREA CARDIEL | SPRING 2025

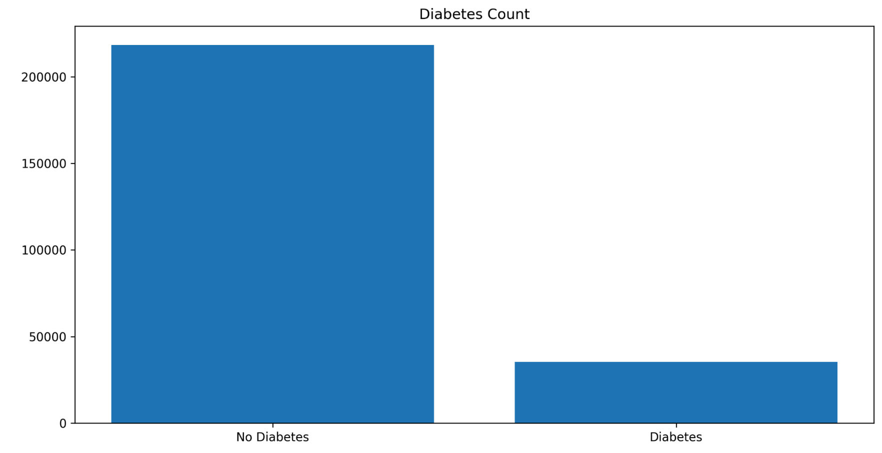
GENERAL METHODOLOGY

To determine the best predictor of diabetes and evaluate model performance using AUC, I began by preprocessing the data. I one-hot encoded the zodiac feature—since it holds no ordinal meaning—while preserving the ordinal structure of features like education level, age bracket, and income bracket. I also standardized numerical features using StandardScaler, which transforms variables to have a mean of 0 and a standard deviation of 1. This ensures that features are on a comparable scale, preventing those with larger ranges from disproportionately influencing the model—a particularly important step for logistic regression and SVM, which are sensitive to feature magnitudes.

Without these transformations, categorical variables might be misinterpreted as ordinal, and large-scale features could dominate the learning process. Although models like Random Forest and AdaBoost do not require such preprocessing, I applied these steps consistently across all models to ensure that performance comparisons reflected differences in predictive ability rather than data preparation.

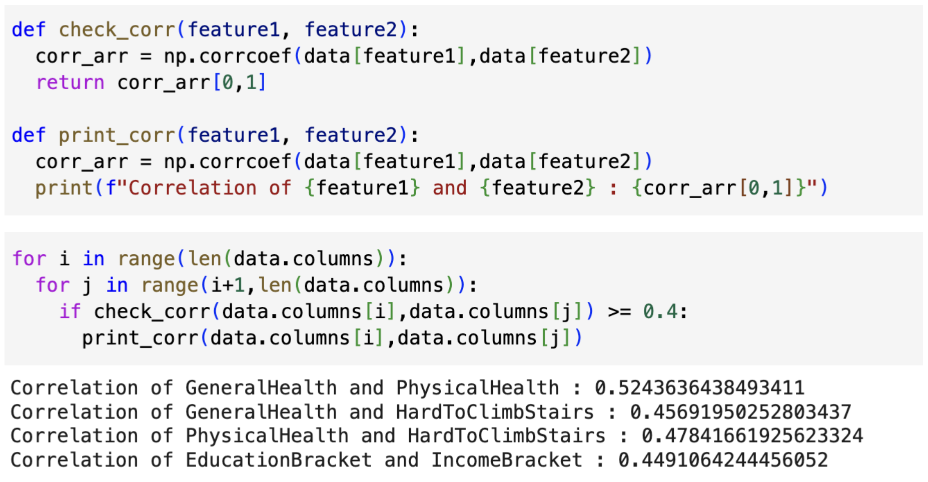
After preprocessing, I split the data into an 80/20 training-test split and evaluated performance using the Area Under the ROC Curve (AUC). An ROC curve plots the false positive rate (x-axis) against the true positive rate (y-axis) at different thresholds, showing the model's ability to distinguish between classes. The diagonal line represents random guessing, and curves closer to the top-left corner indicate better performance.

AUC, therefore, measures how well a model distinguishes between individuals with and without diabetes across classification thresholds. It's especially appropriate in imbalanced classification problems, where accuracy can be misleading. For example, a model that always predicts "no diabetes" could achieve high accuracy if most individuals in the dataset are non-diabetic, while failing to identify actual positive cases. This was definitely a concern in the diabetes dataset, as became apparent when I plotted the number of diabetic vs. non-diabetic cases and observed a noticeably higher number of non-diabetic individuals, reinforcing the importance of using AUC over accuracy as the primary evaluation metric.



AUC accounts for both the true positive rate and false positive rate, making it well-suited for evaluating model performance in contexts where identifying at-risk individuals is critical. An AUC of 1.0 indicates perfect classification—high sensitivity with no false positives—while 0.5 reflects random guessing.

Given its relevance, I also used AUC to assess feature importance. While coefficient magnitudes are often used for this, such measures can be misleading when features are correlated or interact. My EDA supported this concern: for instance, GeneralHealth and PhysicalHealth had a Pearson correlation of 0.52, and both were correlated with HardToClimbStairs. Similarly, EducationBracket and IncomeBracket were correlated (r = 0.45), reflecting real-world socioeconomic ties. While these are not extremely strong correlations, they are moderate enough to suggest potential overlap in predictive power that could distort coefficient-based interpretations.



```python
def check_corr(feature1, feature2):
  corr_arr = np.corrcoef(data[feature1],data[feature2])
  return corr_arr[0,1]

def print_corr(feature1, feature2):
  corr_arr = np.corrcoef(data[feature1],data[feature2])
  print(f"Correlation of {feature1} and {feature2} : {corr_arr[0,1]}")
```

```python
for i in range(len(data.columns)):
  for j in range(i+1,len(data.columns)):
    if check_corr(data.columns[i],data.columns[j]) >= 0.4:
      print_corr(data.columns[i],data.columns[j])
```

```
Correlation of GeneralHealth and PhysicalHealth : 0.5243636438493411
Correlation of GeneralHealth and HardToClimbStairs : 0.45691950252803437
Correlation of PhysicalHealth and HardToClimbStairs : 0.47841661925623324
Correlation of EducationBracket and IncomeBracket : 0.4491064244456052
```

For this reason, I used an AUC-based drop test, which directly measures the impact of each feature on model performance. I removed each feature individually, retrained the model, and recorded the change in AUC. The feature whose removal caused the largest drop was identified as the most important predictor.

To visualize my findings for each model, I used two key plots:

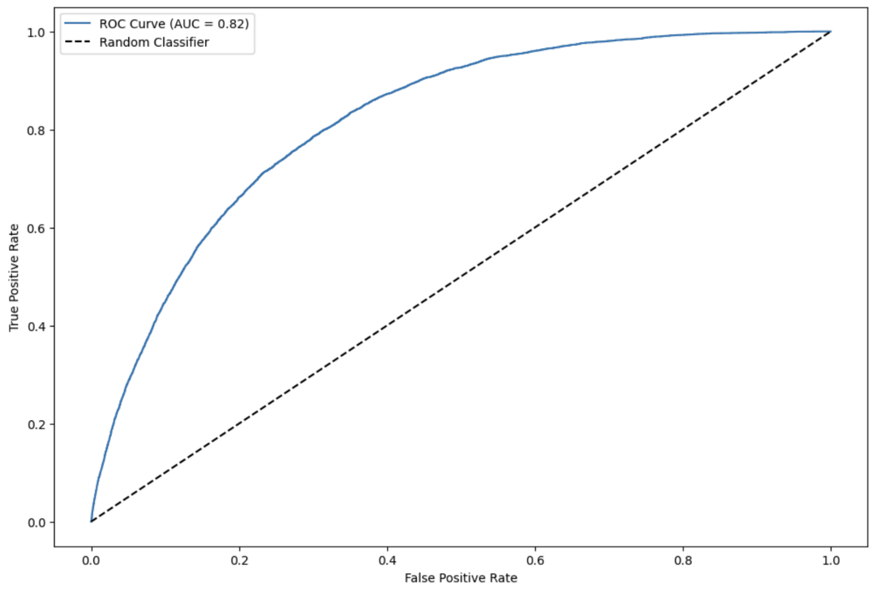
- **ROC Curve:** This plots the false positive rate (x-axis) against the true positive rate (y-axis) at different thresholds, showing the model's ability to distinguish between classes. The diagonal line represents random guessing, and curves closer to the top-left corner indicate better performance.
- **Feature Importance AUC Drop Plot:** This bar plot displays the drop in AUC when each feature is removed. Features with larger drops are more critical to model performance.

This overall approach—thoughtful preprocessing, AUC-based evaluation, and performance-driven feature analysis—formed the foundation for comparing models and identifying the most important predictors of diabetes in each.

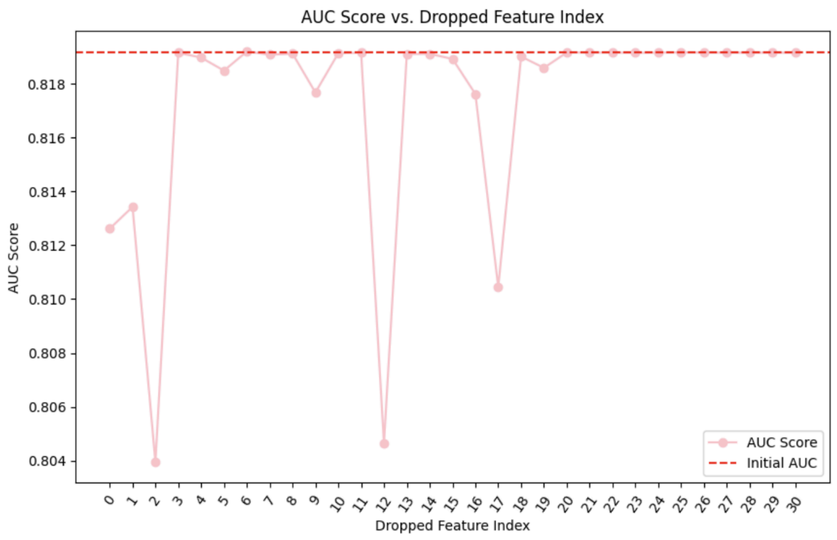
1. LOGISTIC REGRESSION MODEL

To evaluate the best predictor and AUC of a logistic regression model on this dataset, I applied the methods discussed and justified as described in the *General Methodology* section–the standardization of numerical features in preprocessing being particularly important to this model–and using an AUC-based drop test to identify the most important predictor.

With respect to the model itself, I used scikit-learn's LogisticRegression with the 'liblinear' solver–which is efficient for binary classification and performs well on small to medium-sized datasets like this one–and a maximum of 1000 iterations to ensure convergence. After training the model, I evaluated its performance on the test set using the AUC and obtained an **AUC of 0.82. This suggests that the model distinguishes well between diabetic and non-diabetic individuals, assigning higher risk scores to the correct group about 82% of the time.** The ROC curve shows this visually: the model's curve (blue) lies well above the random classifier baseline (dashed), hugging the top-left corner, which reflects high true positive rates and low false positive rates across thresholds.



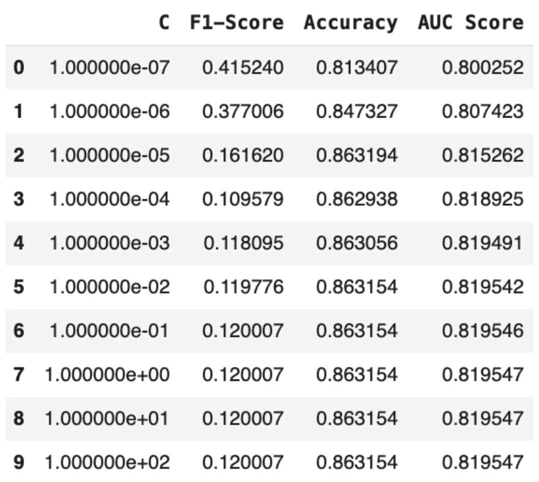
To assess feature importance, I performed an AUC-based drop test as discussed in the *General Methodology* section and visualized results in the AUC drop plot, where each pink point represents the model's AUC with one feature removed, and the red dashed line marks the baseline AUC using all features. The largest drop occurred when feature index 2 was excluded, indicating that it was the most important predictor of diabetes in the logistic regression model.



The results show that the logistic regression model performs well overall (AUC = 0.82) and that feature index 2, **BMI, is the most influential predictor of diabetes in this model**. This means that this particular feature contributed most to the model's ability to correctly classify individuals. In a healthcare setting, identifying such key features can guide targeted interventions, screenings, or further study. The drop-test approach also demonstrated that not all features are equally valuable, and means that potential interventions might also explore the other features that decrease performance like general health (feature index 12).

2. SVM

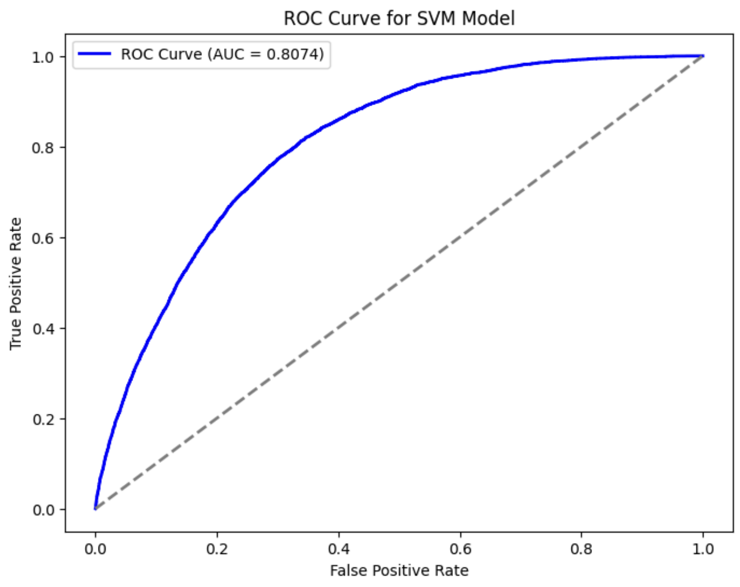
To evaluate the best predictor of diabetes and assess model performance using a Support Vector Machine (SVM), I followed the general methodology outlined in the *General Methodology* section, including preprocessing (one-hot encoding and standardization), model evaluation using AUC, and feature importance analysis through an AUC-based drop test. Since SVMs are sensitive to feature scale, standardization was especially important here. Before training the final model, I optimized the C-value, which controls the balance between model complexity and misclassification. I trained LinearSVC models across a wide range of C-values (from 1e-7 to 1e+2), using 5-fold cross-validation. I evaluated each model using F1 score, accuracy, and AUC. F1 score was used during cross-validation because it balances precision and recall, which is particularly important in imbalanced classification problems like this one. I selected C = 1e-6, which produced the best combination of AUC and F1-score, and used it to train the final model.



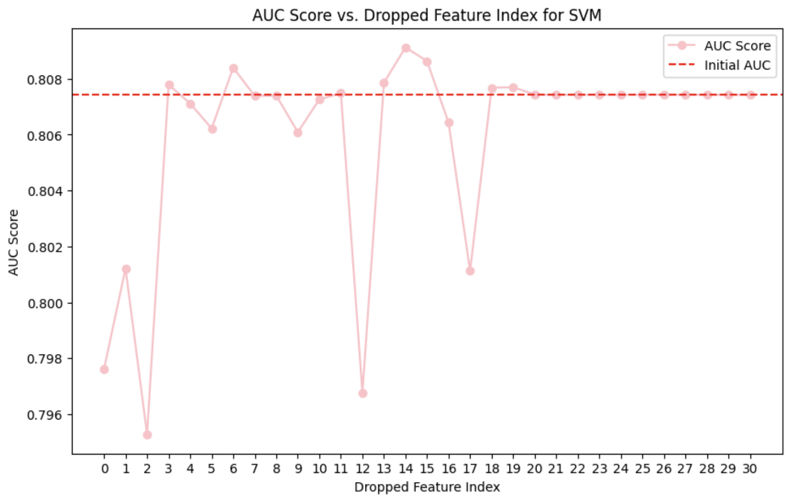
	C	F1-Score	Accuracy	AUC Score
0	1.000000e-07	0.415240	0.813407	0.800252
1	1.000000e-06	0.377006	0.847327	0.807423
2	1.000000e-05	0.161620	0.863194	0.815262
3	1.000000e-04	0.109579	0.862938	0.818925
4	1.000000e-03	0.118095	0.863056	0.819491
5	1.000000e-02	0.119776	0.863154	0.819542
6	1.000000e-01	0.120007	0.863154	0.819546
7	1.000000e+00	0.120007	0.863154	0.819547
8	1.000000e+01	0.120007	0.863154	0.819547
9	1.000000e+02	0.120007	0.863154	0.819547

SVMs are especially effective in high-dimensional spaces but are highly sensitive to hyperparameters like C- which is critical when the data is not perfectly linearly separable, as is often the case with real-world health data. Tuning C helped the model manage the trade-off between margin size and classification error. I used F1 score rather than accuracy during cross-validation to ensure the model performed well across both false positives and false negatives, which is crucial in identifying at-risk individuals in imbalanced datasets. After training the model, I used an AUC-based drop test to determine which features contributed most to the model's predictive performance. I also calculated coefficient-based feature importances, which though contrary to my generalized methodology, was done because of my own personal curiosities brought about by the similar SVM and logistic regression AUCs. I sought an added layer of interpretability and model validation.

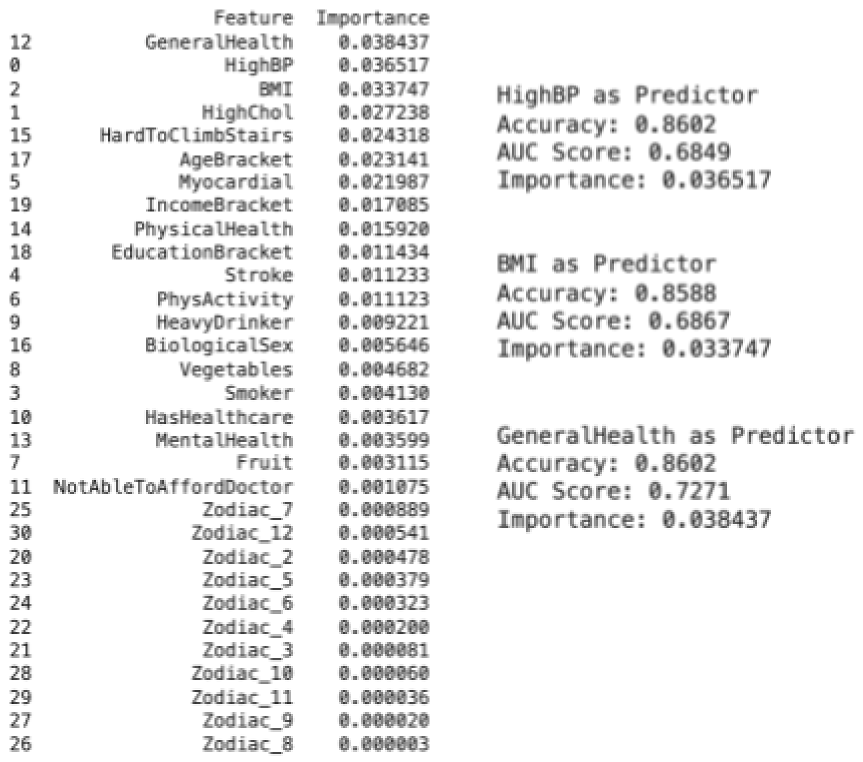
The SVM model (with C = 1e-6) achieved an **AUC of 0.81. This suggests that the model distinguishes well between diabetic and non-diabetic individuals, assigning higher risk scores to the correct group about 81% of the time.** The ROC curve shows the model's true positive rate vs. false positive rate across thresholds. The blue curve lies well above the random classifier baseline (diagonal), indicating strong discriminatory performance.



The AUC drop plot shows how performance changed when each feature was individually removed. The red dashed line represents the original AUC (0.81). **The largest performance drop occurred when feature index 2 (BMI) was removed, identifying it as the most important feature within the full model.**



Coefficient-based feature importance showed that GeneralHealth, HighBP, and BMI had the highest absolute weights. To further explore this, I tested each of these features individually in the model. While GeneralHealth had the highest standalone AUC (0.7271), BMI caused the greatest drop in model performance when removed.



```
                  Feature  Importance
12          GeneralHealth    0.038437
0                  HighBP    0.036517
2                     BMI    0.033747
1                HighChol    0.027238
15       HardToClimbStairs   0.024318
17             AgeBracket    0.023141
5              Myocardial    0.021987
19          IncomeBracket    0.017085
14         PhysicalHealth    0.015920
18       EducationBracket    0.011434
4                  Stroke    0.011233
6            PhysActivity    0.011123
9            HeavyDrinker    0.009221
16          BiologicalSex    0.005646
8              Vegetables    0.004682
3                  Smoker    0.004130
10          HasHealthcare    0.003617
13           MentalHealth    0.003599
7                   Fruit    0.003115
11  NotAbleToAffordDoctor    0.001075
25               Zodiac_7    0.000889
30              Zodiac_12    0.000541
20               Zodiac_2    0.000478
23               Zodiac_5    0.000379
24               Zodiac_6    0.000323
22               Zodiac_4    0.000200
21               Zodiac_3    0.000081
28              Zodiac_10    0.000060
29              Zodiac_11    0.000036
27               Zodiac_9    0.000020
26               Zodiac_8    0.000003
```

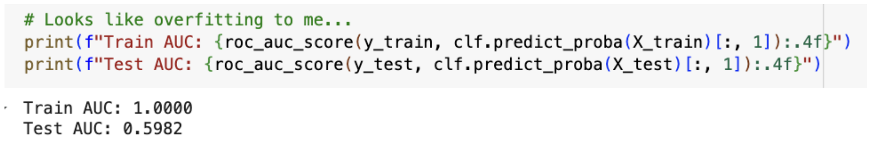
```
HighBP as Predictor
Accuracy: 0.8602
AUC Score: 0.6849
Importance: 0.036517

BMI as Predictor
Accuracy: 0.8588
AUC Score: 0.6867
Importance: 0.033747

GeneralHealth as Predictor
Accuracy: 0.8602
AUC Score: 0.7271
Importance: 0.038437
```

The SVM model demonstrated strong predictive performance, with an **AUC of 0.8074.** While GeneralHealth was the most predictive on its own, BMI emerged as the most important feature in the context of the full model, based on the drop in AUC. This suggests that **BMI contributes most to the model's overall ability to correctly classify cases**, even if it's not the best standalone predictor. The agreement between the SVM and logistic regression models—both identifying BMI and GeneralHealth as top features—adds confidence to the robustness of these predictors.

At the same time, these findings raise interesting questions about the relationship between BMI and GeneralHealth. Since GeneralHealth is a self-reported, subjective measure, it likely reflects broader perceptions of well-being that may implicitly include BMI, blood pressure, and other health conditions. The fact that both features rank highly across models suggests there may be some redundancy or interaction, and further investigation could clarify whether GeneralHealth provides independent predictive value or simply reflects a combination of more objective measures like BMI and HighBP. Understanding these relationships would be valuable for both model refinement and health communication.

3. SINGLE DECISION TREE

To evaluate the best predictor of diabetes using a single decision tree, I trained a baseline DecisionTreeClassifier on the preprocessed dataset following the same methodology used for other models. Initially, I had thought to use the entropy criterion rather than Gini for better class discrimination and only later on realized my flawed thinking once I had actually *fully read* the article I was consulting. I evaluated the model using AUC and accuracy and visualized performance with a ROC curve. I also applied an AUC-based drop test to assess feature importance, where each feature was removed one at a time to observe its impact on AUC. The initial model showed signs of overfitting: while it achieved a perfect training AUC of 1.0, the test AUC was only 0.5982, indicating poor generalization.

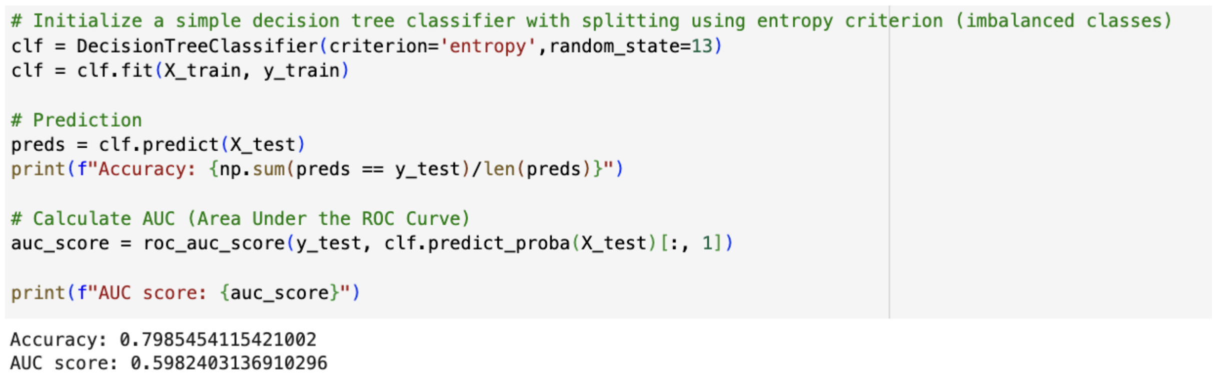

```
# Looks like overfitting to me...
print(f"Train AUC: {roc_auc_score(y_train, clf.predict_proba(X_train)[:, 1]):.4f}")
print(f"Test AUC: {roc_auc_score(y_test, clf.predict_proba(X_test)[:, 1]):.4f}")
```

```
Train AUC: 1.0000
Test AUC: 0.5982
```

To address this, I performed basic hyperparameter tuning using RandomizedSearchCV to adjust parameters such as max_depth, min_samples_split, and min_samples_leaf. I then retrained the model using the best-found parameters and re-evaluated its performance.

Decision trees are powerful yet highly flexible models, which makes them prone to overfitting, especially when left unconstrained. The model's perfect fit to the training data but poor test performance clearly signaled overfitting. To improve generalization, I tuned the model's complexity via regularization-based parameters that limit tree depth and control how splits occur. I used RandomizedSearchCV for faster tuning across a modest candidate set. I also ran an AUC-based drop test to consistently evaluate feature importance across models. Additionally, I observed the structure of the trained tree, particularly the root node split, to understand which feature the model prioritized first based on impurity reduction—measured by Gini impurity.

The untuned model had an **accuracy of 0.7985** and an **AUC of 0.5982** on the test set. As shown in the **ROC curve**, the model's predictions were only slightly better than random guessing. This high accuracy but low AUC was another indicator of potential overfitting.

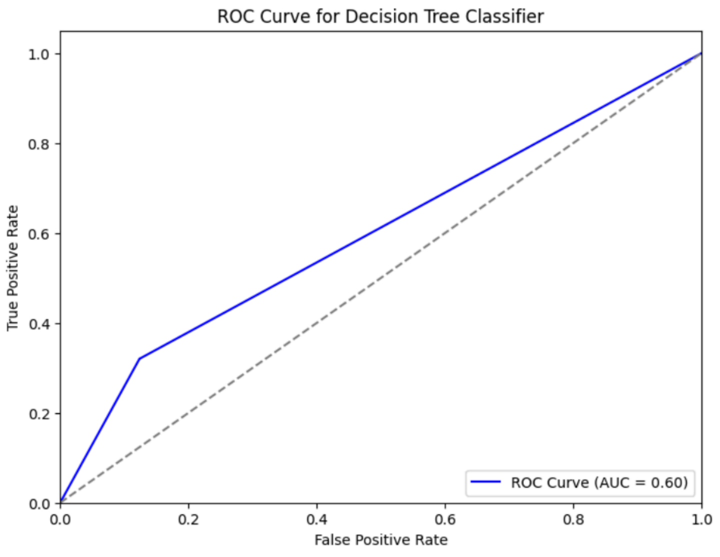


```
# Initialize a simple decision tree classifier with splitting using entropy criterion (imbalanced classes)
clf = DecisionTreeClassifier(criterion='entropy',random_state=13)
clf = clf.fit(X_train, y_train)

# Prediction
preds = clf.predict(X_test)
print(f"Accuracy: {np.sum(preds == y_test)/len(preds)}")

# Calculate AUC (Area Under the ROC Curve)
auc_score = roc_auc_score(y_test, clf.predict_proba(X_test)[:, 1])

print(f"AUC score: {auc_score}")
```

```
Accuracy: 0.7985454115421002
AUC score: 0.5982403136910296
```

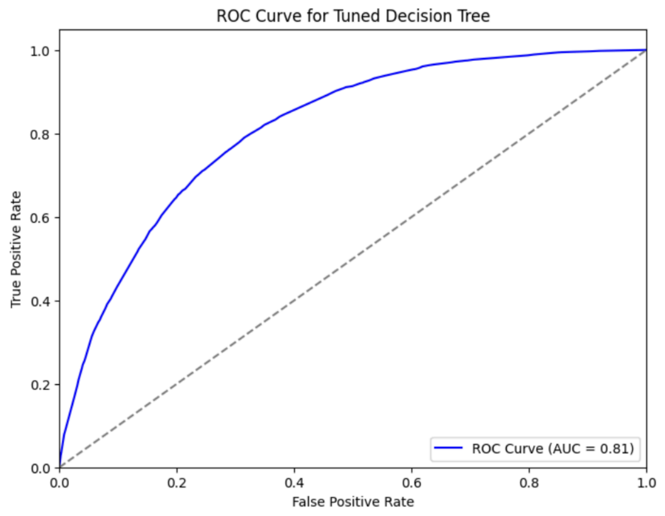
After tuning, performance improved significantly: the model achieved a **test AUC of 0.8113** and **accuracy of 0.8641**. The ROC curve now showed a clear separation between classes. Notably, in both the untuned and tuned trees, the root node split was based on HighBP. This indicates that the tree found HighBP to be the most informative feature for making an initial split.

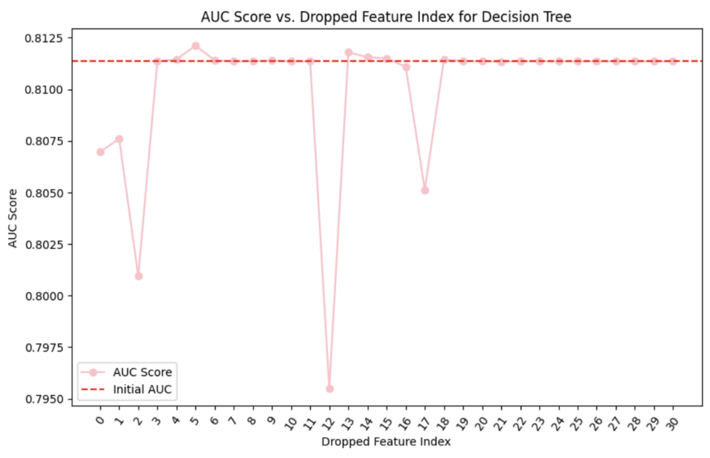
The **AUC drop plot** confirmed that most features had little impact on performance individually, with the exception of **feature index 12 (General Health)**, whose removal led to the largest drop.

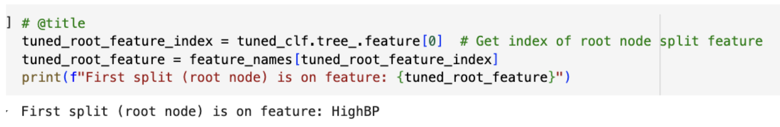


```
Accuracy: 0.8641004415011038
AUC score: 0.8113471453110448
```

Show code







```
# @title
tuned_root_feature_index = tuned_clf.tree_.feature[0]  # Get index of root node split feature
tuned_root_feature = feature_names[tuned_root_feature_index]
print(f"First split (root node) is on feature: {tuned_root_feature}")
```

```
First split (root node) is on feature: HighBP
```

The results highlight the importance of regularization in decision trees, where even a simple model can overfit if left unconstrained. After tuning, the decision tree performed comparably to other models like SVM and logistic regression. The consistent use of HighBP as the root node suggests that it provided the largest immediate reduction in impurity (Gini impurity), likely because its binary nature allowed for a clean and informative first split. However, the AUC drop test identified GeneralHealth as the most important feature overall, meaning its removal had the greatest negative effect on the model's ability to classify cases correctly.

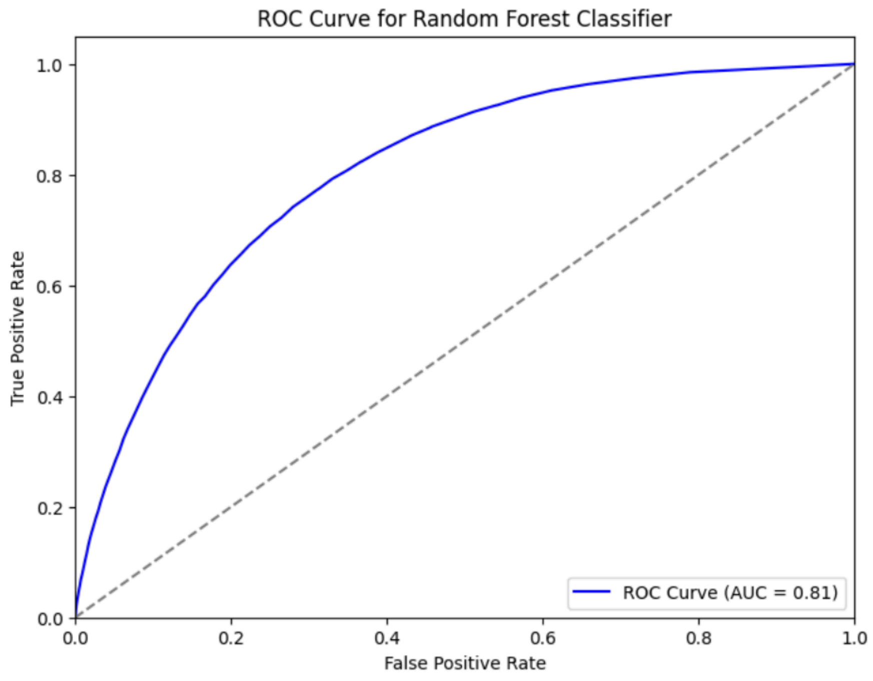
This distinction underscores the value of combining multiple approaches to feature interpretation. While HighBP was the best local discriminator at the root, GeneralHealth contributed more broadly across the entire tree. **Therefore, while HighBP was the most influential early split, GeneralHealth was the most important predictor in terms of overall model performance.** This contrasts with the findings from logistic regression and SVM, where BMI was consistently identified as the most important predictor across both coefficient-based and AUC-drop approaches. The difference here may reflect how tree-based models prioritize features that create early, highly pure splits, while linear models capture cumulative, global influence more effectively. Interestingly, my earlier analysis of SVM feature importance caused me to suggest a possible connection between GeneralHealth and BMI, so I again wonder if GeneralHealth may be acting as a proxy for BMI (or vice versa) in the decision tree, capturing similar information in a way that the tree structure finds more directly useful.

4. RANDOM FOREST

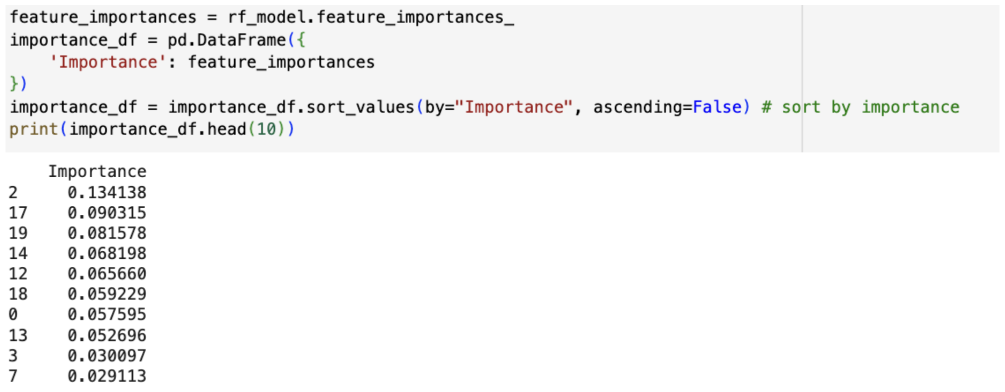
To evaluate diabetes prediction using a Random Forest model, I trained a RandomForestClassifier with 100 decision trees (n_estimators=100) on the preprocessed dataset. I introduced additional randomness by setting max_samples=0.5 and max_features=0.5, so each tree trained on a different subset of samples and features. After training, I evaluated the model using both AUC and accuracy, plotted an ROC curve to visualize performance, and assessed feature importance using two complementary methods: (1) scikit-learn's built-in .feature_importances_ based on impurity reduction, and (2) an AUC-based drop test, where I removed each feature individually and observed the resulting change in AUC.

Random Forests are powerful ensemble models that reduce overfitting by aggregating predictions from many diverse decision trees. They handle complex, nonlinear relationships well, making them well-suited for structured medical data. I used AUC as the primary evaluation metric to account for the dataset's class imbalance.For feature importance, I combined two methods: the built-in Gini-based importances, which measure how much each feature reduces impurity across trees, and the AUC-based drop test, which directly quantifies each feature's impact on overall model performance. Comparing both methods gave a more robust and interpretable understanding of which predictors mattered most.

The **Random Forest model achieved an AUC of 0.81** and an accuracy of 0.8625 on the test set, indicating strong predictive performance. The ROC curve showed that the model consistently achieved a high true positive rate while keeping the false positive rate low, consistent with its high AUC score.



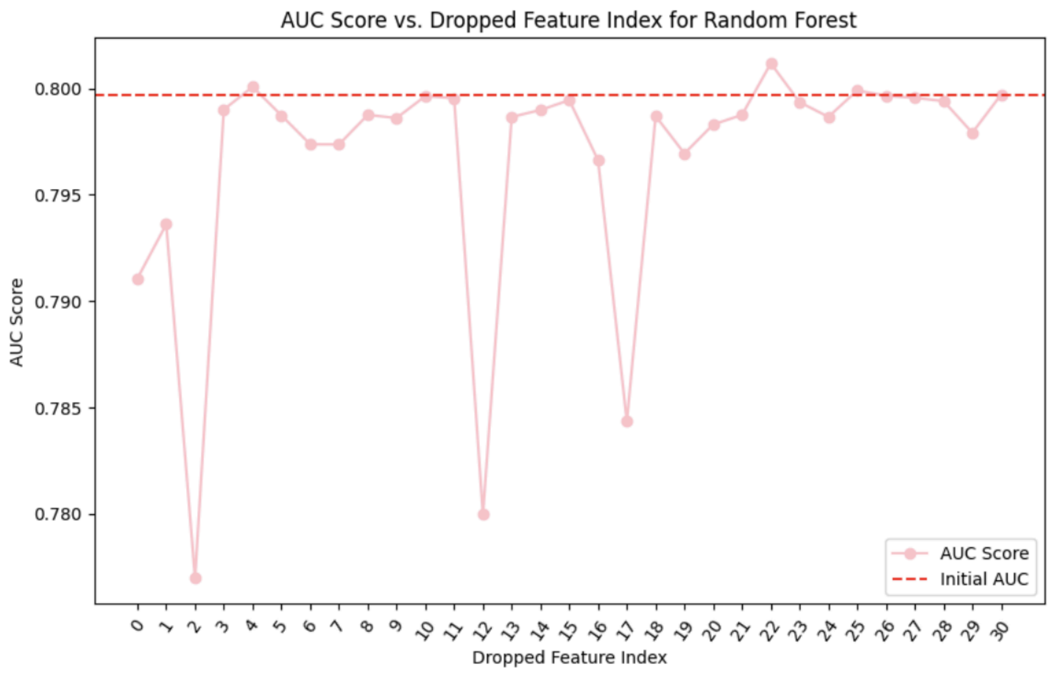
The **Gini-based feature importances identified BMI (feature index 2) as the most important feature overall, with an importance score of 0.1341**. Other highly ranked features included BiologicalSex (17), IncomeBracket (19), PhysicalHealth (14), and GeneralHealth (12).



```
feature_importances = rf_model.feature_importances_
importance_df = pd.DataFrame({
    'Importance': feature_importances
})
importance_df = importance_df.sort_values(by="Importance", ascending=False) # sort by importance
print(importance_df.head(10))
```

```
    Importance
2     0.134138
17    0.090315
19    0.081578
14    0.068198
12    0.065660
18    0.059229
0     0.057595
13    0.052696
3     0.030097
7     0.029113
```

The AUC-based drop test further supported these results: removing BMI caused the largest drop in AUC, confirming it as the feature with the most significant overall impact on model performance. Smaller drops were observed when features like GeneralHealth and BiologicalSex were removed, suggesting they also contribute meaningfully to the model's accuracy, but to a lesser extent than BMI.



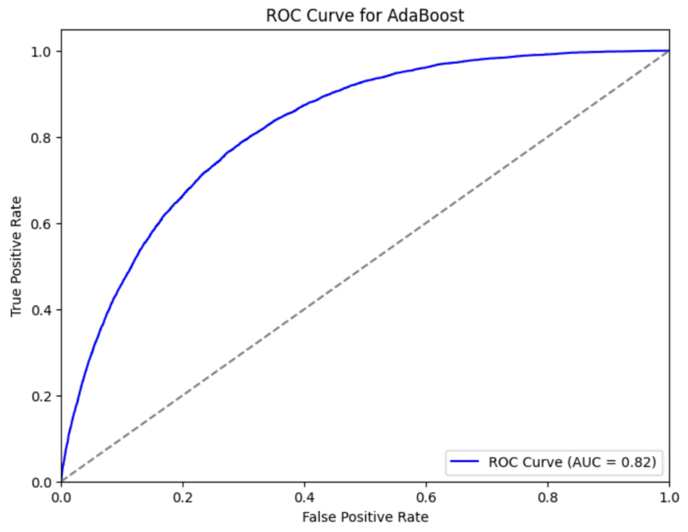
The Random Forest model performed comparably to the best-performing models in this study, with a **strong AUC (0.81)** and high accuracy. Importantly, both feature importance methods identified **BMI as the most important predictor of diabetes**—echoing the results from logistic regression, SVM, and decision trees. While HighBP and GeneralHealth were also relevant, BMI had the most consistent and significant impact on model performance. Additionally, the emergence of features like BiologicalSex and IncomeBracket as relatively important in the Random Forest suggests that tree-based models may be capturing nonlinear interactions or threshold effects that linear models miss. These insights reinforce the value of ensemble models in structured healthcare prediction tasks, and further confirm BMI's central role in diabetes risk across a variety of modeling techniques.

5. ADABOOST

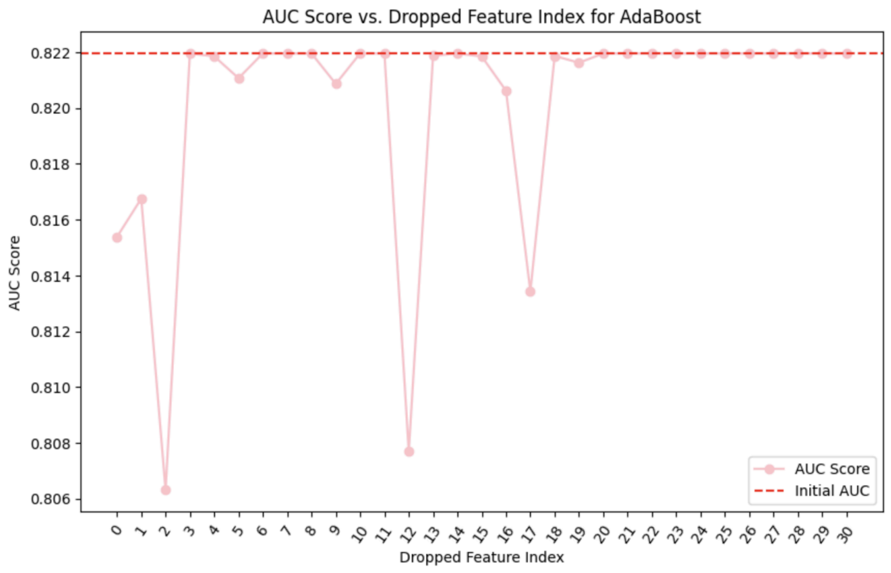
To evaluate diabetes prediction using an AdaBoost model, I trained an AdaBoostClassifier using decision stumps (i.e., decision trees with max_depth=1) as weak learners. The model was trained on the same preprocessed dataset used for previous models, and evaluated using AUC and accuracy, again, in accordance with the same generalized approach.

AdaBoost builds a strong classifier by sequentially combining many weak learners, adjusting their weights based on prior errors. This makes it particularly effective for problems with subtle patterns or noisy features. I used decision stumps as base estimators, which is common in AdaBoost, since the algorithm compensates for their simplicity through many boosting rounds. I used AUC as the main evaluation metric to maintain consistency across models and account for the class imbalance. I also used an AUC-based drop test (rather than model-internal importances) to keep feature importance assessment consistent across all models, especially since AdaBoost's internal importances aren't as interpretable when built from stumps.

The AdaBoost model achieved an **AUC of 0.82**, matching the best-performing models in the comparison. The ROC curve showed a smooth upward curve toward the top-left corner, indicating strong discriminative ability.



The AUC-based drop test identified feature index 2 (**BMI) as the most important predictor:** its removal caused the largest drop in AUC. Other features such as GeneralHealth (index 12) and BiologicalSex (index 17) also led to moderate performance drops, suggesting they played supporting roles in boosting rounds.



Overall, AdaBoost achieved top-tier performance, with an **AUC of 0.82,** while the AUC drop test once again confirmed **BMI as the most important feature** — consistent with nearly every other model tested. AdaBoost's reliance on simple rules makes it especially informative when a feature like BMI causes a large drop: it means the model heavily depends on even basic splits involving that feature. Supporting features like GeneralHealth and BiologicalSex also contributed meaningfully, but no other feature matched BMI’s impact. These findings reinforce BMI’s central role in diabetes prediction and highlight AdaBoost as a strong, interpretable ensemble method, particularly useful when simplicity and performance must be balanced.

EXTRA CREDIT

A. BEST MODEL FOR THIS DATASET

Of all the models evaluated, AdaBoost achieved the highest AUC (0.82), matching only logistic regression in predictive performance. However, AdaBoost also successfully identified important features like BMI and GeneralHealth using simple, interpretable decision stumps. Unlike the single decision tree, AdaBoost did not overfit. Compared to logistic regression and SVM, it was more capable of capturing nonlinear relationships and interactions between features, which were particularly important in this dataset. While Random Forest also performed well (AUC = 0.81), AdaBoost consistently delivered top performance with less complexity and more stable feature rankings.

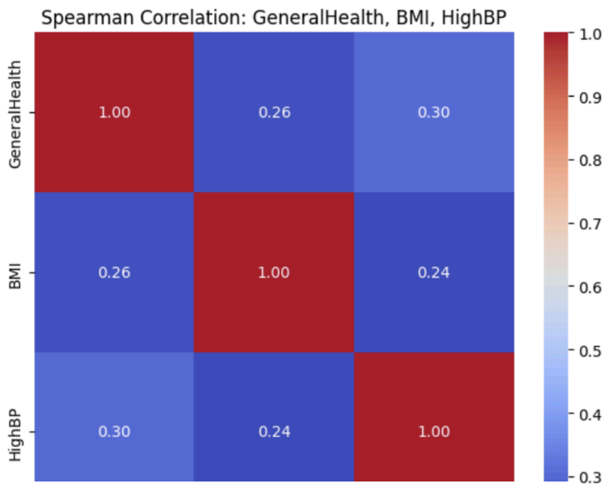
Based on performance, interpretability, and robustness, AdaBoost is the best model for predicting diabetes in this dataset. It combines the strengths of both linear and tree-based models—achieving top predictive accuracy while still highlighting consistent and meaningful predictors. AdaBoost's success suggests that combining many simple rules is especially effective in this setting, where multiple health-related variables contribute to risk in subtle, interacting ways. The model generalizes well without requiring extensive tuning and consistently surfaced BMI as the most important predictor, further reinforcing its central role across all analyses.

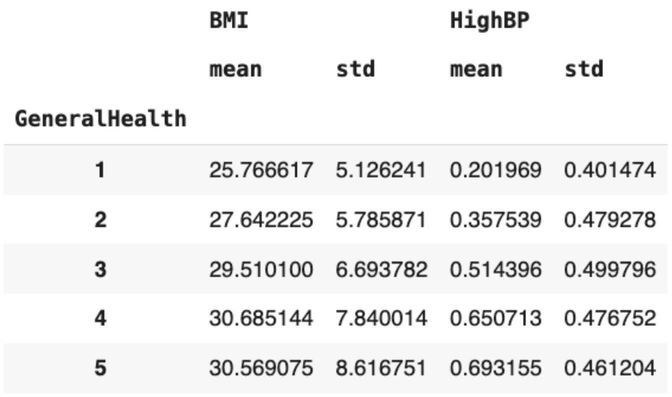
B. ADDITIONAL INSIGHT

One non-obvious insight I found in this dataset is that self-rated GeneralHealth may act as a nonlinear summary of multiple underlying risk factors — even though it doesn’t appear strongly correlated with any single one. Initially, the Pearson correlations between GeneralHealth and objective health metrics like BMI and HighBP were low, which might suggest it's not a particularly informative feature. But when I switched to Spearman correlation, which measures monotonic relationships rather than strictly linear ones, the results were similarly modest– yet still didn’t capture the full picture.

What revealed the real insight was grouping the data by GeneralHealth level: as self-rated health worsened, both BMI and the probability of high blood pressure increased steadily and nonlinearly. For instance, average BMI rose from 25.8 to 30.6, and HighBP prevalence from 20% to nearly 70%.

This gap between weak correlation coefficients and strong ordinal group trends suggests that GeneralHealth encodes something more complex — likely reflecting a threshold-based or latent perception of health that integrates multiple risk factors. It may be a compressed self-assessment of overall risk, and its





	BMI		HighBP	
	mean	std	mean	std
GeneralHealth				
1	25.766617	5.126241	0.201969	0.401474
2	27.642225	5.785871	0.357539	0.479278
3	29.510100	6.693782	0.514396	0.499796
4	30.685144	7.840014	0.650713	0.476752
5	30.569075	8.616751	0.693155	0.461204

predictive strength in models likely comes from capturing interactions or cumulative effects that no single variable alone expresses well.