

CMSC 210: Assignment #2

Supervised Learning

Building the Classifiers

In the rapidly evolving healthcare landscape, the "Digital Health Adoption and Usage" survey presents invaluable insights from 57 respondents concerning their telehealth experience. In the realm of computer science, this survey offers an exceptional opportunity to apply machine learning classifiers, i.e., Decision Trees, K-Nearest Neighbor, and Naive Bayes algorithms, to comprehend the intricate factors impacting an individual's overall utilization of digital health services and tools to manage their health and wellness. In this assignment, the proponents examined each classifier's performance across distinct sections of the survey data and delved into an essential aspect of overfitting, which is crucial for ensuring the reliability and generalizability of results. Moreover, this takes off from the result in Assignment #1, where most data cleaning tasks were done, plus additional pre-processing tasks such as dropping rows with NA values, scaling, one-hot encoding, and 70%-30% train test split (random_state = 42) to develop the models.

Decision Tree Classifier

The first classifier examined in this assignment is the Decision Tree (DT) model. A DT classifier was developed based on the following attributes from our survey data, namely:

- (q1) Target Attribute: Level of Digital Health Adoption (Low, Moderate, and High)
- Other Attributes:
 - (age) Age Range
 - (q9) Trust in Telehealth (scale 1-5)
 - (q10) Digital Health Literacy Score (score 0-100)
 - (q13) Self-Reported Health Improvement due to Digital Health (None, Minor, Moderate, Significant)
 - (q14) Accessibility of Telehealth Services (Limited, Adequate, Good, Excellent)

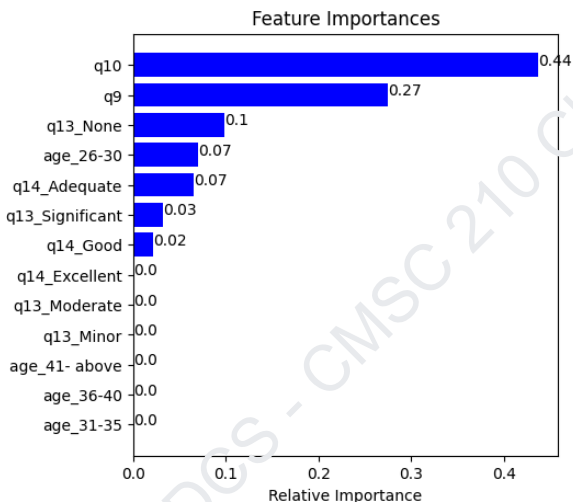
Here, the proponents employed two approaches in building the classifiers. First, via a straightforward approach, using a decision tree with default parameters (full-grown tree). This served as a baseline to better understand the model without performing parameter tuning. Second, the proponents applied hyperparameter tuning to determine the model's "best" parameters within the training dataset. This optimizes the DT's performance by achieving the highest possible accuracy in predicting the target variable. Selecting the values for tuning in the parameter grid was based on the baseline results and are given as follows:

```
param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 1, 2, 3, 4, 5, 6, 7, 8, 9],
    'min_samples_split': [2, 4, 5],
    'min_samples_leaf': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
}
```

Results of the two approaches employed are provided in Table 1. The table summarizes the parameters used, the accuracy of training and test datasets, essential features, and the tree diagram for the models. Please note that the

confusion matrix and classification report of the “best” model will be discussed in the “Comparing of Clusters” section in this write-up along with the other “best” model of the other classification algorithms used in this Assignment (see Page 10).

Table 1. Decision Tree Diagram results.

Results	Model 1: Decision Tree (Default)	Model 2: Hyper Param. Decision Tree (Best)																																																								
Parameter	<ul style="list-style-type: none">Criterion: GiniMax Depth = NoneMin sample leaf = 1Min sample split = 2	<ul style="list-style-type: none">Criterion: EntropyMax Depth = 7Min sample leaf = 1Min sample split = 2																																																								
Accuracy on Training Set	100.00%	94.59%																																																								
Accuracy on Test Set	35.29%	47.06%																																																								
Feature Importance	<div>Feature Importances</div>  <table><thead><tr><th>Feature</th><th>Relative Importance</th></tr></thead><tbody><tr><td>q10</td><td>0.44</td></tr><tr><td>q9</td><td>0.27</td></tr><tr><td>q13_None</td><td>0.1</td></tr><tr><td>age_26-30</td><td>0.07</td></tr><tr><td>q14_Adequate</td><td>0.07</td></tr><tr><td>q13_Significant</td><td>0.03</td></tr><tr><td>q14_Good</td><td>0.02</td></tr><tr><td>q14_Excellent</td><td>0.0</td></tr><tr><td>q13_Moderate</td><td>0.0</td></tr><tr><td>q13_Minor</td><td>0.0</td></tr><tr><td>age_41- above</td><td>0.0</td></tr><tr><td>age_36-40</td><td>0.0</td></tr><tr><td>age_31-35</td><td>0.0</td></tr></tbody></table>	Feature	Relative Importance	q10	0.44	q9	0.27	q13_None	0.1	age_26-30	0.07	q14_Adequate	0.07	q13_Significant	0.03	q14_Good	0.02	q14_Excellent	0.0	q13_Moderate	0.0	q13_Minor	0.0	age_41- above	0.0	age_36-40	0.0	age_31-35	0.0	<div>Feature Importances</div>  <table><thead><tr><th>Feature</th><th>Relative Importance</th></tr></thead><tbody><tr><td>q10</td><td>0.27</td></tr><tr><td>age_26-30</td><td>0.18</td></tr><tr><td>q13_Significant</td><td>0.16</td></tr><tr><td>q9</td><td>0.1</td></tr><tr><td>q13_Minor</td><td>0.09</td></tr><tr><td>age_31-35</td><td>0.09</td></tr><tr><td>q13_Moderate</td><td>0.09</td></tr><tr><td>q14_Adequate</td><td>0.02</td></tr><tr><td>q14_Excellent</td><td>0.0</td></tr><tr><td>q14_Good</td><td>0.0</td></tr><tr><td>q13_None</td><td>0.0</td></tr><tr><td>age_41- above</td><td>0.0</td></tr><tr><td>age_36-40</td><td>0.0</td></tr></tbody></table>	Feature	Relative Importance	q10	0.27	age_26-30	0.18	q13_Significant	0.16	q9	0.1	q13_Minor	0.09	age_31-35	0.09	q13_Moderate	0.09	q14_Adequate	0.02	q14_Excellent	0.0	q14_Good	0.0	q13_None	0.0	age_41- above	0.0	age_36-40	0.0
Feature	Relative Importance																																																									
q10	0.44																																																									
q9	0.27																																																									
q13_None	0.1																																																									
age_26-30	0.07																																																									
q14_Adequate	0.07																																																									
q13_Significant	0.03																																																									
q14_Good	0.02																																																									
q14_Excellent	0.0																																																									
q13_Moderate	0.0																																																									
q13_Minor	0.0																																																									
age_41- above	0.0																																																									
age_36-40	0.0																																																									
age_31-35	0.0																																																									
Feature	Relative Importance																																																									
q10	0.27																																																									
age_26-30	0.18																																																									
q13_Significant	0.16																																																									
q9	0.1																																																									
q13_Minor	0.09																																																									
age_31-35	0.09																																																									
q13_Moderate	0.09																																																									
q14_Adequate	0.02																																																									
q14_Excellent	0.0																																																									
q14_Good	0.0																																																									
q13_None	0.0																																																									
age_41- above	0.0																																																									
age_36-40	0.0																																																									
Tree Diagram																																																										

Discussion

Model 1: Decision Tree (Default). Accuracy on the training set is high at 100%. This indicates that the model fits the training data well. However, the accuracy on the test set is significantly lower at 35.29%. This suggests overfitting, where the model must better generalize to new, unseen data. The feature importance analysis also shows that the (q10) "Digital Health Literacy Score" is the essential feature in making predictions, followed by (q9) "Trust in Telehealth" and (q13) Health Improvement = "None." Moreover, the model is intense and complex (Max Depth = None), which could explain the overfitting. This obscures the rationale behind each leaf node's assignment.

Model 2: Hyper Param. Decision Tree (Best). The training set's accuracy is slightly lower than the default model, at 94.59%, indicating that this model still fits the training data well but may also be prone to over-fitting. The accuracy of the test set, on the other hand, is higher (VS default) at 47.06%, which suggests better generalization to unseen data. This model is intentionally constrained to a maximum depth = 7, which slightly helps improve overfitting, reduces complexity, and better generalizes the data. The feature importance analysis also shows that (q10) "Digital Health Literacy Score" is the essential feature in making predictions, followed by (age) "Age range 26-30." This implies that users with high literacy scores are more likely to be Moderate or High adopters, while ages 26-30 and a "Significant" level of health improvement remains a significant factor.

So, despite the high accuracy on the training set in Model 1, it is prone to overfitting, as evidenced by the significant drop in accuracy on the test set. This is because it is a fully-grown tree that captures noise in the training data. Thus making it less effective on new data. The prominence of the "Digital Health Literacy Score" in this model suggests that individuals with high digital health literacy are classified as High or Moderate adopters. However, the model's lack of interpretability makes it challenging to pinpoint the specific reasons for allocating classes in each leaf node. Model 2, on the other hand, with its relatively controlled depth and minimum samples, generalizes better to unseen data. However, like Model 1, the balance between its training and test accuracy is not reasonable as it still provides traces of over-fitting. In both models, the digital Health Literacy Score and "Trust in Telehealth" feature have high importance in both models, indicating their influence on predicting the level of digital health adoption. These significant features are important as they could help fine-tune the model by focusing more on the top-ranking features or exploring interactions between them for a more accurate prediction.

Therefore, the hyperparameterized model is preferred due to its better generalization performance and controlled complexity. But, while the results of the tuned model are preferred, there are still limitations in the model to overfit because of the following possible reasons:

- Small training dataset. Here, we only have 70% of the 57 survey responses, which is very little. It cannot even be considered as big data. Overfitting happens because if the minimum number of samples required to split a node is too low, the tree might make splits too specific to the training data.
- Complexity and Depth. In Model 2, the tree was already pruned at a certain level; however, it is still quite complex, thus leading to perfectly fitting the training data. It should be noted that when a tree becomes too deep, it may learn noise specific to the training data. Thus losing its ability to generalize to new data.

- Limited Data or Features. Provided that the dataset is already small, it also lacks diversity in its features. This has been observed in Assignment #1, where counts to certain questions are leaning towards “None” or “None Applicable” or similar to the target variable where only three out of the total sample is tagged as “High” adoption. Hence, there is an imbalance in the dataset. Because of these, trees might struggle to capture the true patterns in the data. Hence, they tend to memorize the training data rather than generalize, resulting in overfitting.
- Sensitive to small variations. DTs are highly sensitive to minor fluctuations in the data --- outliers. As shown in Assignment #1, there are responses that the proponent kept because it was believed that they are still part of the entire population of “Digital Health Adopters,” but only a few of them were captured in the survey. The DTs tend to create decision boundaries based on these outliers, leading to overfitting by capturing noise rather than the underlying pattern.

K-Nearest Neighbors Classifier

The second classifier examined in this activity is the K-Nearest Neighbor (KNN) classifier. Unlike in the first model, this was built on a similar dataset but only includes all numeric data (except the Target variable) in the survey, namely:

- (q1) Target Attribute: Level of Digital Health Adoption (Low, Moderate, and High)
- Other Attributes:
 - (q7) Frequency of Telehealth Consultations
 - (q8) Time Spent on Health Applications (in hrs.)
 - (q9) Trust in Telehealth (scale 1-5)
 - (q10) Digital Health Literacy Score (score 0-100)

Here, the proponents employed three approaches in building the model. First, a straightforward approach where a KNN classifier was modeled using default parameters (a simple approach to solving for k was used, $k = \sqrt{n}$ ¹), and attributes were not scaled. The second approach uses the same default parameters, but features are scaled. Since KNN is a distance-based algorithm, it is affected by the range of features. Hence, there is a need to scale. However, for this exercise, we wanted to also show how scaled and unscaled values impact the model's performance and determine a baseline. Lastly, for the third approach, it utilizes hyperparameter tuning to determine the model with the “best” parameters within the training dataset. The hyperparameters used are shown below.

```
KNN_param_grid = {
  'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25],
  'weights': ['uniform', 'distance'],
  'p': [1, 2]
}
```

Critical results for these three approaches are outlined in Table 2, and the Actual and Predicted scatterplot visual using the “best” model of three examples of paired attributes are shown in Table 3. Other approaches aside from

¹ Source:

<https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb#:~:text=The%20optimal%20K%20value%20usually,be%20aware%20of%20the%20outliers.>

those above, such as KNN with Principal Component Analysis (PCA) and Hyper parameterized KNN with PCA, were also made. However, since there were no significant differences in their results, it was not included in this write-up's discussion. For reference, you can find the implementation and result details of the approaches using PCA in the Python source code and Notebook documentation for Assignment #2.

Table 2. K-Nearest Neighbor Results.

Results	Model 1: K-Nearest Neighbor (Default - Unscaled)	Model 2: K-Nearest Neighbor (Default - Scaled)	Model 3: Hyper Param. KNN (Best - Scaled)
Parameters	<ul style="list-style-type: none"> • $K = 7$ • Weights = Uniform • Power = 2 (Euclidean Distance) 		<ul style="list-style-type: none"> • $K = 9$ • Weights = Uniform • Power = 1 (Manhattan Distance)
Accuracy on Training Set	70.27%	70.27%	70.27%
Accuracy on Test Set	52.94%	58.82%	64.71%

Table 3: Actual and Predicted (Hyper Param. KNN (Best - Scaled)) scatterplot visualization of three examples of paired attributes by class.

(q7) Frequency of Telehealth Consultations VS (q8) Time Spent on Health Applications (in hrs.)

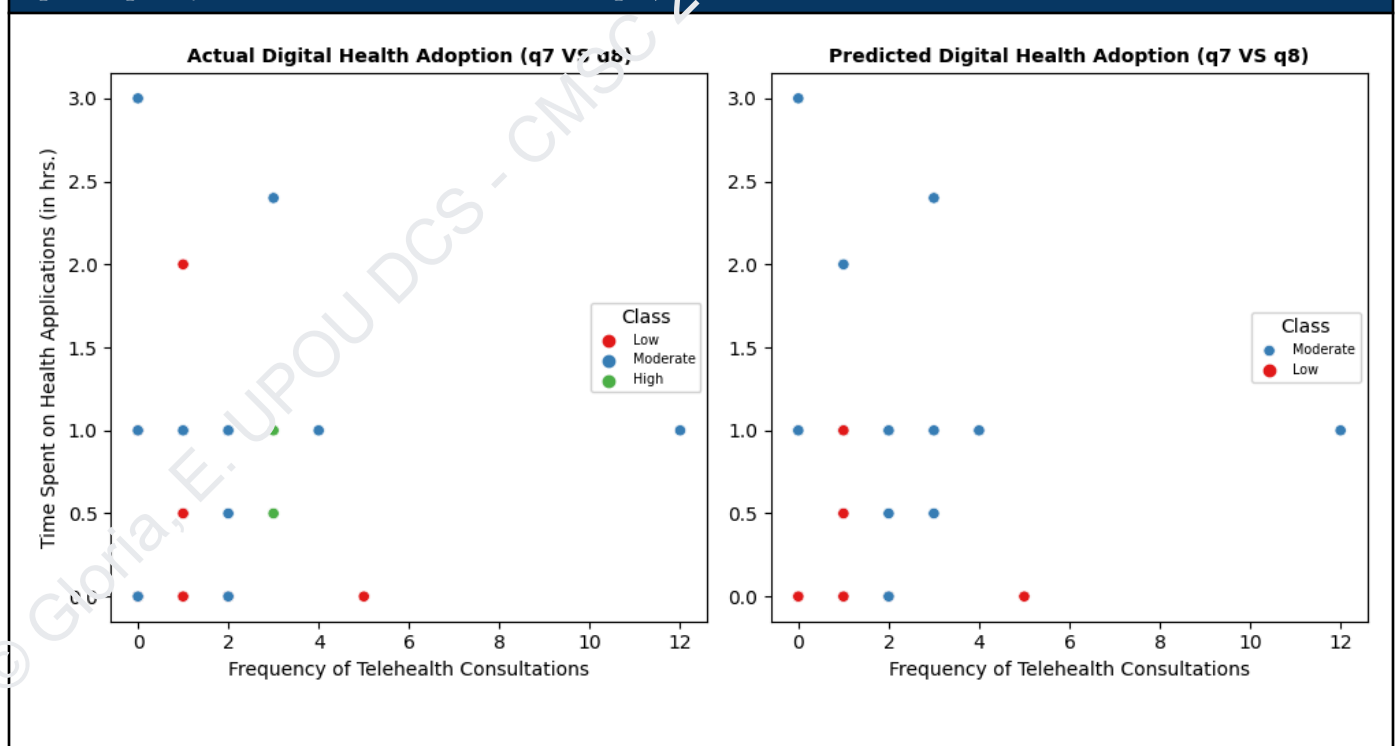
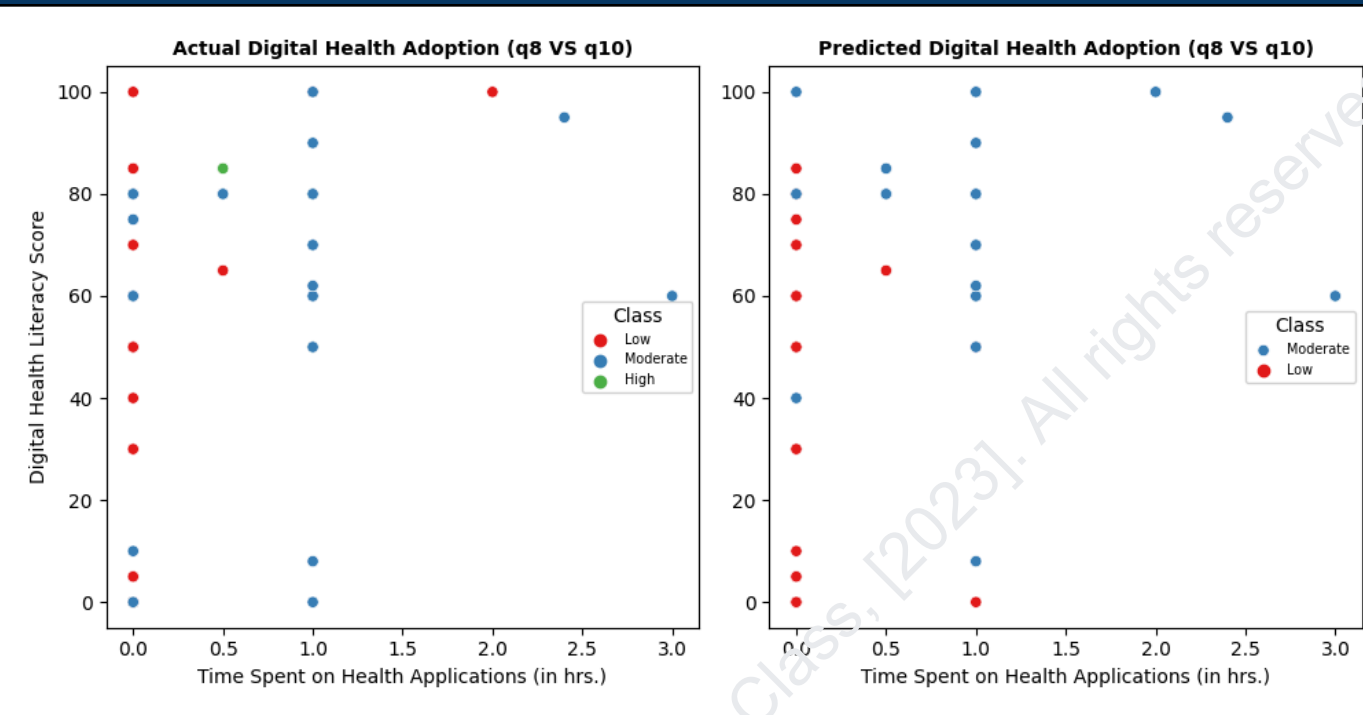
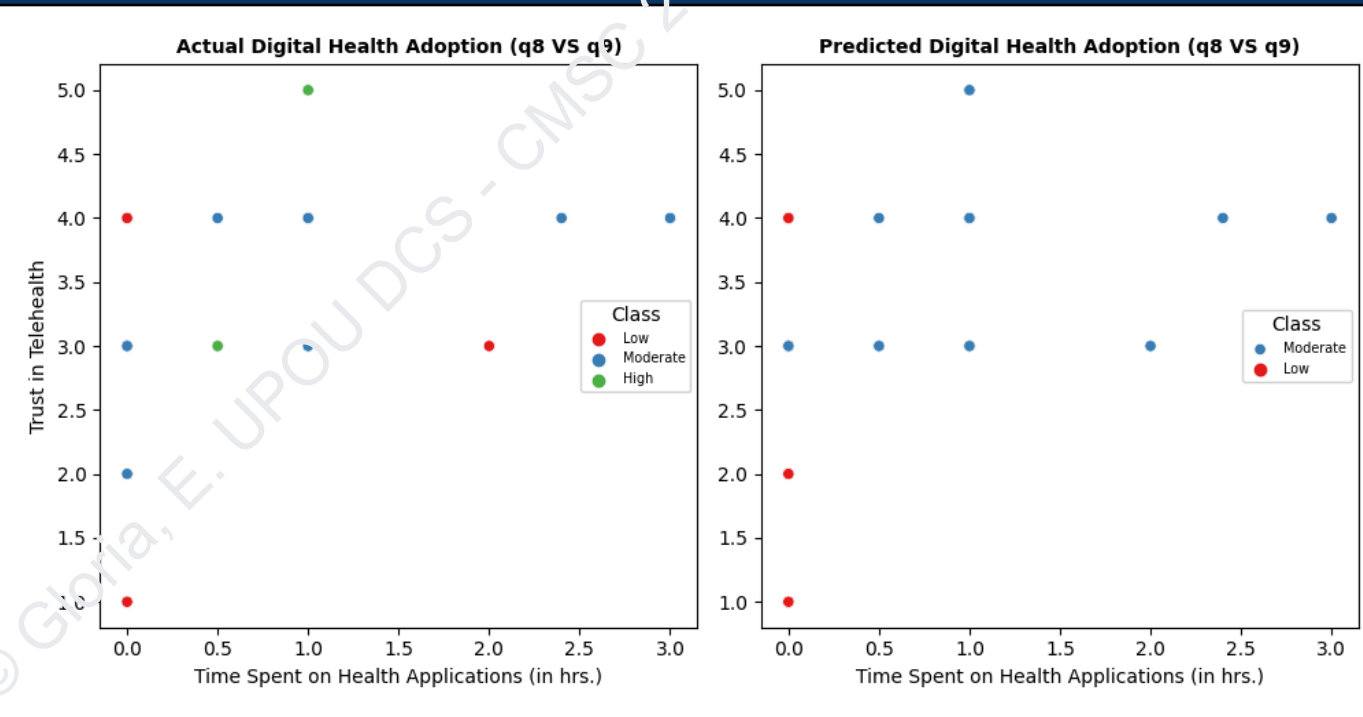


Table 3 continued.

(q8) Time Spent on Health Applications (in hrs.) VS (q10) Digital Health Literacy Score (score 0-100)



(q8) Time Spent on Health Applications (in hrs.) VS (q9) Trust in Telehealth (scale 1-5)



Discussion

Model 1: K-Nearest Neighbor (Default - Unscaled). This model was trained with the default features, meaning no feature scaling was applied, and it used Euclidean distance as the distance metric for finding the nearest neighbors. The training accuracy is relatively high at 70.27%, indicating that the model fits the training data well. However, the accuracy on the test set is significantly lower at 52.94%, suggesting that the model may need to generalize better to new data. Feature scaling might have improved the performance in this case.

Model 2: K-Nearest Neighbor (Default - Scaled). The same KNN model is used in this case, but feature scaling has been applied to the data. The training accuracy remains at 70.27%, indicating that scaling the features did not impact the model's ability to fit the training data. However, the accuracy on the test set has improved to 58.82%, which suggests that scaling the features helped the model generalize better to new data.

Model 3: Hyper Param. KNN (Best - Scaled). For the third model, hyperparameter turning has been performed to find the best parameters for the KNN model. The parameters selected a $k = 9$ (instead of 7), weights are uniformed, and Manhattan distance was used as a metric (VS Euclidean). Despite the parameter changes, the training accuracy remains at 70.27%, indicating that the model still fits the training data well. The accuracy on the test set has improved further to 64.71%, which is the best performance among the tree configurations.

The nature of the KNN algorithm can explain the consistency of training accuracy across different model configurations. In a nutshell, the KNN algorithm is non-parametric, and it does not learn explicit model parameters during training. Instead, it stores the entire training dataset in memory and makes predictions based on the nearest neighbor relative to a distance metric. Since the model uses the same training data in all three configurations, the training accuracy remains the same because it is based on the same data and classes. This is evident because it measures how well the model fits the training data consistently across the different configurations.

In terms of improvement in the test accuracy measure across the different model configurations, this can be due to:

- Scaling. Features on different scales can dominate the distance calculation, leading to suboptimal results.
- Hyperparameter tuning. Optimizes performance by selecting the “best” combination of hyperparameters to improve the model's performance.
- Combining scaling and parameter tuning. Applying both methods helped improve the model's generalization of new, unseen data.

In summary, feature scaling and hyperparameter turning have been found to improve the KNN model's performance. The choice of distance metric (Euclidean or Manhattan) and the number of k neighbors can also significantly impact the model's ability to predict unseen data. The model with scaled features and tune hyperparameters outperforms the other two configurations regarding test accuracy. However, further analysis

and evaluation metrics may still be needed to determine the overall suitability of the model for this classification task. It is also noticeable that in Table 3, the model finds it difficult to predict class = “High”.

Naive-Bayes Classifier

The third and last classifier examined in this assignment is the Naive-Bayes (NB) classifier. Similar to the first method examined, the model was built on the same dataset that includes the following attributes, namely:

- (q1) Target Attribute: Level of Digital Health Adoption (Low, Moderate, and High)
- Other Attributes:
 - (age) Age Range
 - (q9) Trust in Telehealth (scale 1-5)
 - (q10) Digital Health Literacy Score (score 0-100)
 - (q13) Self-Reported Health Improvement due to Digital Health (None, Minor, Moderate, Significant)
 - (q14) Accessibility of Telehealth Services (Limited, Adequate, Good, Excellent)

There are different types of NB classifiers. However, in this assignment, the proponent opted to use a Gaussian NB algorithm since this method is particularly well-suited for general continuous or real-valued feature-based problems. The features also adhere to the independence assumption, which is true within each class.

Now, similar to the first two classifiers examined, the same approach of using a default NB method and a hyperparameterized NB model was developed and compared. The results of these models are shown in Table 4, and a sample manual computation (based on the lecture notes) of NB with $X = (\text{Age } 26 - 30; \text{Trust Rating} = 5; \text{Literacy Score} = 80, \text{Health Improvement} = \text{Moderate}, \text{Accessibility} = \text{Adequate})$ is provided in Table 5.

Table 4. Naive Bayes Results.

Results	Model 1: Naive-Bayes (Default)	Model 2: Hyper Param. Naive-Bayes (Best)
Parameters	Variance Smoothing: 1e-09	Alpha: 0.001
Accuracy on Training Set	56.76%	64.86%
Accuracy on Test Set	41.18%	64.71%

Table 5. Sample computation when $X = (\text{Age } 26 - 30; \text{Trust Rating} = 5; \text{Literacy Score} = 80, \text{Health Improvement} = \text{Moderate}, \text{Accessibility} = \text{Adequate})$.

Digital Health Adoption	(age) Age Range [26-30]	(q9) Trust Rating in Telehealth [5]	(q10) Digital Health Literacy Score [80]	(q13) Health Improvement due to Digital Health [Moderate]	(q14) Accessibility of Telehealth [Adequate]
C1 = Low (22) $P(C1) = 22/54$	11 $P(\text{age} C1) = 11/22$	0 $P(q9 C1) = 0/22$	1 $P(q10 C1) = 1/22$	2 $P(q13 C1) = 2/22$	5 $P(q14 C1) = 5/22$

Table 5 continued.

Digital Health Adoption	(age) Age Range [26-30]	(q9) Trust Rating in Telehealth [5]	(q10) Digital Health Literacy Score [80]	(q13) Health Improvement due to Digital Health [Moderate]	(q14) Accessibility of Telehealth [Adequate]
C2 = Moderate (29) $P(C2) = 29/54$	13 $P(\text{age} C2) = 13/29$	1 $P(q9 C2) = 1/29$	9 $P(q10 C2) = 9/29$	12 $P(q13 C2) = 12/29$	13 $P(q14 C2) = 13/29$
C3 = High (3) $P(C3) = 3/54$	3 $P(\text{age} C3) = 3/3$	1 $P(q9 C3) = 1/3$	1 $P(q10 C3) = 1/3$	1 $P(q13 C3) = 1/3$	0 $P(q14 C3) = 0/3$

Solving for which class X belongs as shown below:

- $P(X | \text{Digital Health Adoption} = \text{"Low"}) = 11/22 \times 0/22 \times 1/22 \times 2/22 \times 5/22 = 0$
 $\circ P(X | \text{Digital Health Adoption} = \text{"Low"}) * P(\text{Digital Health Adoption} = \text{"Low"}) = 0 * 22/54 = 0$
- $P(X | \text{Digital Health Adoption} = \text{"Moderate"}) = 13/29 \times 1/29 \times 9/29 \times 12/29 \times 13/29 = 0.00088985751$
 $\circ P(X | \text{Digital Health Adoption} = \text{"Moderate"}) * P(\text{Digital Health Adoption} = \text{"Moderate"}) =$
 $0.00088985751 * 29/54 = 0.00047788644$
- $P(X | \text{Digital Health Adoption} = \text{"High"}) = 3/3 \times 1/3 \times 1/3 \times 1/3 \times 0/3 = 0$
 $\circ P(X | \text{Digital Health Adoption} = \text{"High"}) * P(\text{Digital Health Adoption} = \text{"High"}) = 0 * 3/54 = 0$

Therefore, X belongs to class (Digital Health Adoption = "Moderate"). The individuals would have a moderate level of adoption towards digital health.

Discussion

Model 1: Naive-Bayes (Default). This model was trained with a very low variance smoothing parameter at 1e-09. Variance smoothing is used to overcome the problem of zero variance --- having the same values in the features and for computing a non-zero division of the Gaussian distribution. However, this extremely low variance smoothing might not address the aforementioned problem effectively and could lead to over-fitting. The training accuracy is 56.76%, indicating that the model fits more than 50% of the training data (moderate fit) but might indicate a potential overfitting, especially considering the low variance. With a test accuracy of 41.15%, the model demonstrates a drop in performance from the training accuracy. The discrepancy between training and testing accuracies hints at overfitting, where the model fails to generalize well to new data.

Model 2: Hyper Param. Naive-Bayes (Best). In the tuned model, the variance smoothing parameter was adjusted to 0.001. This higher value strikes a better balance between addressing zero variance issues and preventing overfitting. This model also achieved a higher training accuracy of 64.86% compared to the default model. This indicates an improved fit to the training data without significant overfitting concerns. With a test accuracy of 64.71%, the hyperparameterized model demonstrates a substantially improved performance compared to the default model. The much smaller gap between training and test accuracies indicates enhanced generalizability, suggesting a reduction in overfitting.

In summary, the default model shows signs of overfitting, with a significant gap between training and test accuracies. Its low variance smoothing might not effectively address the issue of zero variance, leading to poor generalization. In contrast, the tuned model displays a reduced gap between training and test accuracies. Adjusting the variance smoothing parameter improved the model's generalization, resulting in a more balanced and reliable performance. Thus, the tuned model outperforms the default model by a considerable margin on the test set, indicating its better generalization ability. It strikes a better balance between fitting the training data and making accurate predictions on new data.

Moreover, it was also observed in the manual computation for NB that there are probabilities equal to zero, which is influenced by the small sample size and imbalance in the dataset. This limitation is observed across the different classifiers examined in this assignment and should be highlighted as an issue for further improvement.

Comparing the “Best” Classifiers

Table 6. Performance comparison of the three “best” classifiers.

Results	Hyper Param. Decision Tree (Best)	Hyper Param. KNN (Best Scaled)	Hyper Param. Naive-Bayes Classifier (Best)
Accuracy on Training Set	94.59%	70.27%	64.86%
Accuracy on Test Set	47.06%	64.71%	64.71%
Classification Report	<pre> Classification Report: precision recall f1-score support High 0.0000 0.0000 0.0000 1 Low 0.4286 0.5000 0.4615 6 Moderate 0.5556 0.5000 0.5263 10 accuracy 0.3200 0.3333 0.4706 17 macro avg 0.3280 0.3333 0.3295 17 weighted avg 0.4781 0.4706 0.4725 17 </pre>	<pre> Classification Report: precision recall f1-score support High 0.0000 0.0000 0.0000 1 Low 0.6000 0.5000 0.5455 6 Moderate 0.6667 0.8000 0.7273 10 accuracy 0.4222 0.4333 0.6471 17 macro avg 0.4222 0.4333 0.4242 17 weighted avg 0.6039 0.6471 0.6203 17 </pre>	<pre> Classification Report: precision recall f1-score support High 0.0000 0.0000 0.0000 1 Low 0.6667 0.3333 0.4444 6 Moderate 0.6429 0.9000 0.7500 10 accuracy 0.4365 0.4111 0.6471 17 macro avg 0.4365 0.4111 0.3981 17 weighted avg 0.6134 0.6471 0.5980 17 </pre>
Confusion Matrix			
Pros	<ul style="list-style-type: none"> • Easy interpretability • Provided insights into influential features • Works well with non-linear relationship in data 	<ul style="list-style-type: none"> • Stable model fit due to consistency in training accuracy • Better generalization due to scaling and tuning 	<ul style="list-style-type: none"> • Reduced overfitting concerns. • Aligned training and test accuracies. • Better generalization capacity • Works well with small data

Table 6. continued.

Results	Hyper Param. Decision Tree (Best)	Hyper Param. KNN (Best - Scaled)	Hyper Param. Naive-Bayes Classifier (Best)
Cons	<ul style="list-style-type: none"> • Susceptible to overfitting • Lack of interpretability due to Tree complexity and noise in the training data • Emphasis on influential features suggests a bias towards these variables, potentially limiting its ability to generalize accurately. • Low precision to determine class = "High" 	<ul style="list-style-type: none"> • Consistency, however, also indicates a good fit leaning towards training data. • Further evaluation metrics may be needed to assess overall suitability. • Low precision to determine class = "High" 	<ul style="list-style-type: none"> • Sensitive to outliers • Zero probability issue as shown in the manual computation • Low precision to determine class = "High"

Table 7. Cross-validation results of the three "best" classifiers.

Cross Validation Method		Hyper Param. Decision Tree (Best)	Hyper Param. KNN (Best - Scaled)	Hyper Param. Naive-Bayes Classifier (Best)
Standard	<i>Scores</i>	[0.55, 0.45, 0.55, 0.55, 0.4]	[0.82, 0.64, 0.64, 0.27, 0.7]	[0.73, 0.55, 0.55, 0.45, 0.4]
	<i>Avg. Score %</i>	50%	61%	53%
Stratified	<i>Scores</i>	[0.36, 0.55, 0.55, 0.64, 0.2]	[0.82, 0.45, 0.73, 0.55, 0.7]	[0.82, 0.6, 0.64, 0.73, 0.5]
	<i>Avg. Score %</i>	46%	65%	61%
Shuffle-split	<i>Scores</i>	[0.47, 0.41, 0.35, 0.65, 0.65]	[0.65, 0.59, 0.41, 0.59, 0.71]	[0.65, 0.47, 0.35, 0.65, 0.71]
	<i>Avg. Score %</i>	51%	59%	56%

Which is the best-performing classifier?

In summary, based on the results in Tables 6 & 7, both the Hyper Param. KNN and NB classifiers demonstrate better generalization compared to Hyper Param. DT. While the DT model's emphasis on specific features contributes to its overfitting, both the KNN and NB models show potential for improved generalization through their respective tuned models. However, given the consistency in KNN's performance and the reduced gap between training and test accuracies in the NB model, these models seem less prone to overfitting than the DT model.

Before performing cross-validation, the Hyper Param. NB model strikes a better balance between fitting the training data and generalizing it to new data. But, Precision, Recall and F1 scores for most classes are relatively higher for the Hyper Param. KMM model. Table 7 also supports that the KNN model outperforms the other two models. Aside from demonstrating a reasonable level of accuracy, the cross-validation results are consistent and better than chance. This indicates the Hyper Param. KNN model is performing reasonably well across different data splits. Thus, making KNN the preferred model of choice.

However, when determining the best-performing classifier for “Digital Health Adoption” among Filipinos, we need to lay out the specific goals and constraints of this problem in general. It should also be considered that while the Hyper Param. KNN is the preferred choice, there are limitations in the model, such as small and unbalanced datasets, limited features, and sensitivity to outliers. Therefore, further analysis and evaluation metrics, as well as addressing overfitting concerns, may be necessary to make a final decision about the best classifier for this specific task. Addressing data limitations and using other machine-learning methods would also be necessary to supplement this decision.

Which one of them is prone to overfitting?

Among the “best” models in Table 5, the Decision Tree classifier appears to be more prone to overfitting than the other models. The telltale sign is the discrepancy between its high training accuracy (94.59%) and the significantly lower test accuracy (47.06%). This discrepancy indicates that the model fits very well with the training data but is not generalizing effectively to new data. The controlled complexity achieved by limiting the tree’s depth slightly helps to mitigate overfitting. However, the drop in test accuracy implies that it may still capture noise in the training data, hindering its performance on new data. Further reasons for the overfitting of the DT model are discussed in detail on page 3.