Kidney Disease Prediction and Risk Stratification Through Machine Learning

Amanda D. Hanway

Northwest Missouri State University, Maryville MO 64468, USA
S547016@nwmissouri.edu

Abstract. Kidney disease affects millions of Americans and often goes undiagnosed. In the absence of medical care, data science can be leveraged to alert individuals who are at risk of developing the disease. This paper details the process of developing and evaluating machine learning models to predict the disease based on associated risk factors. Three machine learning models were trained using 2021 survey data collected by state health departments in combination with Centers for Disease Control & Prevention. Each model was run using four different sets of features. The XGBoost algorithm outperformed Random Forest and SGD using the features: diabetes, angina_coronary_heart_disease, high_blood_pressure, and age_group. This model achieved an F1 score of 80.98. The finalized model was applied to 2019 survey data to generate disease predictions and stratify individuals into risk levels. The survey data and model output were imported to a SQL database for storage. Data visualizations were developed in Tableau to summarize the results, which showed that 30% of the respondents were at risk of developing the disease. People aged 65 years and older were at the highest risk with a median risk score of 62.

Keywords: data science · machine learning · disease prediction · classification algorithm

1 Introduction

Kidney disease is a leading cause of death in the United States[5] affecting an estimated 37 million adults[6]. Kidney disease is a condition in which the organs lose their ability to filter the blood, resulting in an accumulation of waste. This may increase an individual's risk of developing other conditions, such as stroke, heart disease, heart failure, and lead to early death[6]. The disease reaches chronic designation after the kidneys experience dysfunction for a period of three months[17]. Left untreated, Chronic Kidney Disease (CKD) can lead to kidney failure, requiring dialysis treatments or a kidney transplant, as the disease progresses to End Stage Kidney Disease (ESKD)[5]. 90% of people living with CKD may be unaware they have the disease[6].

Kidney disease may develop comorbid to other chronic conditions. CKD develops most frequently as a result of diabetes or high blood pressure[11]. 44%

of new cases can be attributed to diabetes[12], while an estimated 1 in 5 adults with hypertension may have the disease. Other risk factors include advanced age, family history, heart disease, and obesity[6], while "social factors, such as lower income and related factors of food insecurity and poorer access to quality health care, are also associated"[7].

Although early CKD may have no symptoms, it can be detected through blood or urine tests. "Early detection and treatment...are the keys to keeping kidney disease from progressing to kidney failure"[17]. Starting treatment in the early stages of CKD can improve health outcomes while decreasing treatment costs associated with disease progression and comorbidities. In 2019, CKD and ESKD treatments for Medicare beneficiaries totaled $124.5 billion[5].

Many factors may influence a person's ability or willingness to visit a health care provider for testing or regular check-ups. Expenses associated with medical tests, office visits, and insurance co-pays may deter an individual from making an appointment. According to a 2020 survey, 22% of respondents avoided medical care due to cost[15]. Other causes of health care avoidance include distance to medical facilities, time commitment, and psychological impediments[2].

In the absence of medical testing and oversight, alternative approaches can be leveraged to alert individuals about their risk of developing kidney disease. This project utilizes data science techniques to predict the disease before diagnosis.

The rest of this paper is organized as follows. Section 2 outlines the data science methodology. Section 3 reviews results and analysis. Section 4 presents conclusions and future work.

1.1 Goals of the Project

The main goal of this project is to develop a machine learning model to predict kidney disease. The model will generate a risk score and assign a risk level to individuals based on their associated risk factors.

A secondary goal of this project is to demonstrate a use case for the machine learning model. The final model will be run on a new dataset to generate predictions. The model's output will be stored in a SQL database, where it will then be connected to Tableau as a reporting platform.

2 Methodology

The methodology for this project involved four distinct phases as detailed in Figure 1.

1. Data
2. Preprocessing
3. Machine Learning Model
4. Application

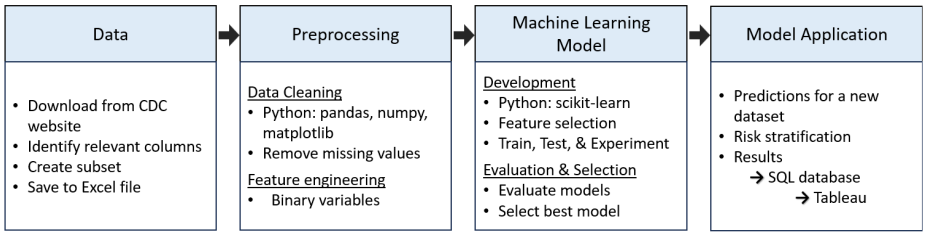

Fig. 1. Methodology

2.1 Data

The data for this project were sourced from the Centers for Disease Control and Prevention (CDC) website. I downloaded the data file containing 438,693 records and 303 columns in SAS transport (.xpt) format. No data scraping techniques were required for collection due to the static file format.

The file includes survey responses to a variety of questions from the 2021 Behavioral Risk Factor Surveillance System (BRFSS)[4], a telephone survey conducted by state health departments with assistance from CDC[9]. Each row represents one respondent. Each column represents the respondent's answer to each question. The data attributes reflect survey respondents' reported behavioral risk factors and medical history.

2.2 Preprocessing

The data first required preprocessing in preparation for a machine learning model. Although the survey data were provided in a normalized format, I further cleaned the dataset by removing unneeded attributes and handling missing values. I transformed some data elements into a more usable format.

I read the full .xpt file into a pandas dataframe using the Python programming language and then created a subset of relevant columns. I included questions identified to be most relevant to kidney disease based on research, including those related to diabetes, high blood pressure, heart disease, and obesity as primary risk factors[5], as well as other questions related to the respondent's environmental and lifestyle factors. I referenced the BRFSS codebook[8] for the questions and value definitions when selecting the columns. I saved this subset to a new Excel file.

2.2.1 Exploratory Data Analysis

I then undertook exploratory data analysis (EDA) to explore and understand my working subset of 438,693 rows and 31 columns. EDA involves generating descriptive or summary statistics and data visualizations to better understand a dataset. This process can reveal patterns or relationships among data elements,

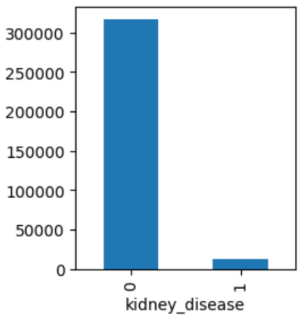

Fig. 2. Kidney Disease Responses

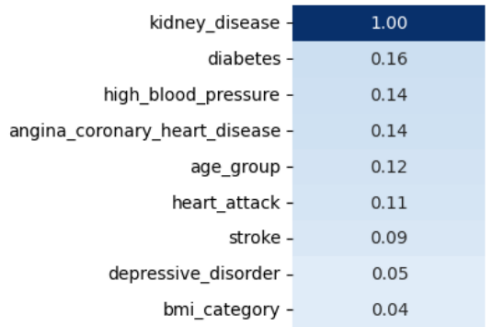

Fig. 3. Top 8 Attributes Correlated to Kidney Disease (Pearson correlation coefficients)

identify value formats and distributions, and illuminate underlying issues such as missing data. EDA is essential in a data science project because it establishes a foundation for further work with the dataset. Taking steps to understand the data before embarking on an analysis or machine learning project ensures the techniques applied are appropriate for the data and its results are justifiable[14].

I first generated descriptive and summary statistics about the dataset to gain an understanding of its overall makeup. I used the df.keys(), df.head(), df.shape(), df.describe() and df.info() functions to review descriptive statistics. I found that the values of my selected columns were in float64 format. I plotted histograms of each column to review the distribution of values. I found that some values were categorical while others were binary, there were slightly more females than males in the data, higher numbers of the older population groups compared to the younger groups, and the kidney disease responses were imbalanced with the majority indicating no history of kidney disease as shown in Figure 2. I used df.isnull().sum() to identify columns with missing values and their counts. Eighteen of my selected columns had missing values, while several other columns had values that denoted a missing answer. I calculated the correlation of kidney disease to its risk factors using df.corr() to identify the top factors to consider as features for the model (Figure 3).

Through EDA techniques, I identified the steps needed to clean and transform the dataset in preparation for the model. Notably, I found there were missing data in my file that I would need to either remove or impute. Sometimes, a value was used to denote a missing answer, and I would need to remove or impute those values as well. I would need to change the format or re-categorize some columns' values into binary answers. The kidney disease responses were imbalanced and I would need to resample the dataset.

2.2.2 Data Cleaning & Feature Engineering

Next I cleaned the data and engineered features for the model. I deleted rows with missing values, as well as values where the answer was "Don't know/Refused/Missing", "Not asked", or similar. I chose to delete rows with missing values rather than impute their value because the size of the dataset allowed for deletion. I renamed the columns to be more clearly recognizable. I converted the column values to binary where possible for use in a classification model. I again extracted a smaller subset of the most-relevant columns and saved the cleaned dataset containing 330,344 records and 16 attributes to an Excel file for further review. The final attributes and their definitions are shown in Table 1.

Table 1. Attribute Definitions

Attribute:	**Definition:**
age_group	Respondent's imputed age in six groups
angina_coronary_heart_disease	Reported history of angina or coronary heart disease
bmi_category	Respondent's computed body mass index category
current_or_former_smoker	Reported history of cigarette smoking
depressive_disorder	Reported history of depressive disorder
diabetes	Reported history of diabetes
exercise_daily	Reported exercise in past 30 days
fruit_consumed_daily	Reported fruit consumption 1 or more per day
heart_attack	Reported history of heart attack
heavy_alcohol_drinker	Reported alcohol consumption: adult men having more than 14 drinks per week and adult women having more than 7 drinks per week
high_blood_pressure	Reported history of high blood pressure
kidney_disease	Reported history of kidney disease
male	Respondent's sex (male/female)
race	Respondent's imputed race/ethnicity
stroke	Reported history of stroke
vegetable_consumed_daily	Reported vegetable consumption 1 or more per day

2.3 Machine Learning Model

Following the steps outlined in Figure 4, I split the data into training and testing datasets, determined features, resampled the training data, then trained, tested, and evaluated machine learning models using different sets of features. I first experimented with several different models, including Random Forest, XGBoost, Linear SGD, ADA Boosted, Neural Net, and SVC. I found that most gave similar results, while SVC was not efficient for my large dataset. Given this, I chose three models for further exploration: Random Forest, XGBoost, and Linear SGD. The models' source code and results can be viewed on GitHub[1]. A description of the three models follows.

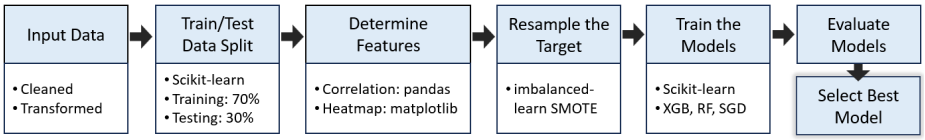

Fig. 4. Machine Learning Model Steps

1. Random Forest Models are built from many Decision Tree models, which use if-then-else rules for learning and prediction[19]. They employ a technique known as "bagging" (bootstrap aggregation) in which random samples are chosen from the entire dataset to independently train a forest of decision trees[22]. The final prediction of a Random Forest model is achieved by averaging the prediction of all individual decision tree models in the ensemble[20].
2. XGBoost (Extreme Gradient Boosting) Models utilize decision trees in combination with a gradient boosting framework. In gradient boosting, each decision tree is created by way of correcting for errors in prior trees. The final prediction of an XGBoost model comes from a strong tree formed by learning from its weaker predecessors.[1]
3. Linear SGD (Stochastic Gradient Descent) Models employ a linear Support Vector Machine (SVM) format using a gradient descent framework. SVM models make predictions by plotting values as coordinates then determining on which side of a class-dividing hyperplane the coordinates fall[18]. In gradient descent, the model arbitrarily chooses a starting position, then moves in the direction of the fastest decrease of the cost function, or negative gradient. With SGD, the gradients are calculated using random samples of the larger dataset. The final prediction of a Linear SGD model is a reflection of combining an SVM algorithm with SGD-based learning[21].

These models are supervised machine learning models for classification applications. Supervised models learn patterns and relationships by training on an

[1] https://github.com/mandi1120/kidney_disease_prediction

input dataset. The models take in labeled data containing one or more Features (independent variables) to learn from, in order to predict a specific Target (dependent variable). In a classification application, the model's intent is to predict a categorical outcome based on information gathered through training[10].

2.3.1 Development

The first step in the model development phase was to split the data into a training set and a test set. I used a 70/30 split which reserved 70% of the data for training and 30% for testing. The training set would be used to train the model, while the testing set would be preserved to test and evaluate the performance of the trained model on an unseen dataset.

Next I selected features for the model. By reviewing the correlation heatmap generated earlier (Figure 3), I determined six features with the greatest correlation to kidney disease. I updated the training and test datasets to use the following features: diabetes, high_blood_pressure, angina_coronary_heart_disease, age_group, heart_attack, and stroke.

Because the kidney_disease responses were imbalanced, I then applied a resampling technique to the training set only, using the imbalanced-learn Synthetic Minority Over-sampling Technique (SMOTE). This method over-samples the minority class until it is balanced with the majority class[16]. In this case, SMOTE increased the count of the True responses until the total matched the count of False responses as shown in Figure 5.

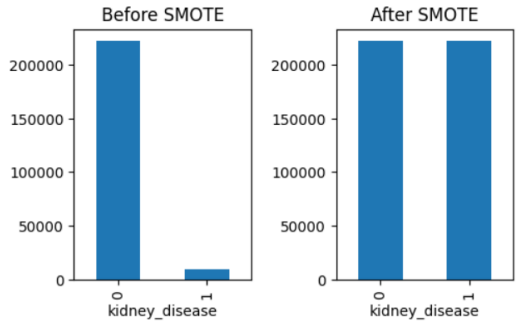

Fig. 5. Results of Synthetic Minority Over-sampling Technique (SMOTE)

I trained the Random Forest, XGBoost, and Linear SGD models using the features and the target as input. I ran the models on the train and test sets four times each, using the different combinations of features shown in Table 2.

Table 2. Feature Sets

Features:	Run 1	Run 2	Run 3	Run 4
diabetes	x	x	x	x
high_blood_pressure	x	x	x	x
angina_coronary_heart_disease	x	x	x	x
age_group		x	x	x
heart_attack			x	x
stroke				x

2.3.2 Evaluation

I generated a classification report and a confusion matrix for each run. The classification report displays metrics to evaluate the model. A confusion matrix classifies the predictions against the actuals, labeled as True Positive Predictions (TP), False Positive Predictions (FP), True Negative Predictions (TN) and False Negative Predictions (FN). I compiled the metrics into Figure 6 and the confusion matrixes into Figure 7 for evaluation.

The results of Run 1 indicate the models were overfitting the training data. The XGBoost and Random Forest accuracy metric decreased from 69.5 on the training set to 57.7 for the testing set, while the SGD model decreased from 68.2 on the training set to 62.1 on the test set. Runs 2, 3, and 4 showed similar accuracy scores between the training and testing data for both XGBoost and Random Forest, indicating the overfitting was resolved by the inclusion of additional features. The SGD model achieved the same results for all four runs.

Next I compared the results for Runs 2, 3, and 4 using the F1 score. The F1 score is calculated as a weighted average of the precision and recall values, where precision measures the ratio of True Positives to all the Positives (TP/(TP+FP)) and recall measures the True Positive Rate (TP/(TP+FN))[13]. On the test data, the Random Forest and XGBoost results were identical, while the Linear SGD model did slightly worse overall. Comparing the F1 scores for each run for each model, the Run 2 feature set performed best for the Random Forest and XGBoost models with an F1 of 80.98.

The predictions from Run 2 can be seen in Figure 7. XGBoost predicted 2,615 TP and 69,219 TN values, and 26,114 FP and 1,156 FN values. Similar results can be seen for Random Forest. While the SGD model predicted a higher number of TP (2,830), it also predicted a higher number of FP values (36,556).

Based on the evaluation of the accuracy, F1, and predictions, I selected the XGBoost model using the Run 2 feature set as the best fit model for my project. Although XGBoost and Random Forest achieved similar scores, XGBoost ran considerably faster than Random Forest. I re-ran the XGBoost model with the selected features and saved the trained model to a pickle file.

	TRAIN				TEST			
Model	**Accuracy**	**F1**	**Precision**	**Recall**	**Accuracy**	**F1**	**Precision**	**Recall**
Run 1								
XGBoost	69.51	68.99	70.89	69.51	57.73	69.84	95.28	57.73
Random Forest	69.51	68.99	70.89	69.51	57.73	69.84	95.28	57.73
Linear SGD	68.28	68.13	68.62	68.28	62.16	73.43	94.95	62.16
Run 2								
XGBoost	71.50	71.49	71.51	71.50	72.49	80.98	94.96	72.49
Random Forest	71.50	71.49	71.51	71.50	72.49	80.98	94.96	72.49
Linear SGD	68.28	68.13	68.62	68.28	62.16	73.43	94.95	62.16
Run 3								
XGBoost	71.67	71.67	71.67	71.67	71.81	80.51	95.00	71.81
Random Forest	71.67	71.67	71.67	71.67	71.81	80.51	95.00	71.81
Linear SGD	68.27	68.13	68.61	68.27	62.16	73.43	94.95	62.16
Run 4								
XGBoost	72.04	72.04	72.05	72.04	71.01	79.95	95.04	71.01
Random Forest	72.04	72.04	72.05	72.04	71.01	79.95	95.04	71.01
Linear SGD	68.27	68.12	68.61	68.27	62.16	73.43	94.95	62.16

Fig. 6. Metrics for Train and Test Data

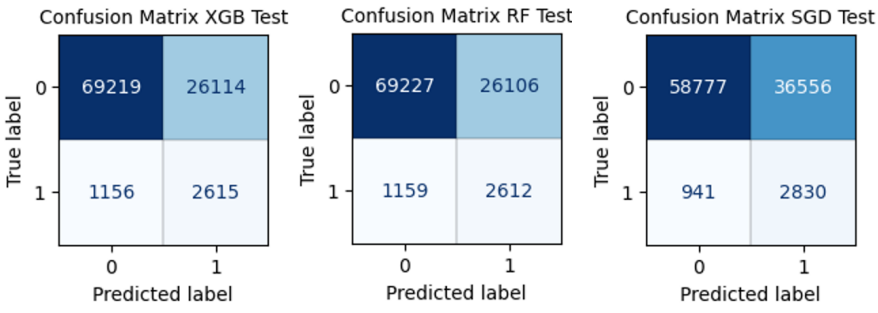

Fig. 7. Confusion Matrix for Test Data with Run 2 Feature Set

2.4 Application

Following the steps shown in Table 8, I applied the machine learning model to a new dataset then developed visualizations to summarize the results.

I first loaded a new dataset into a python dataframe. I used 2019 BRFSS data, which I downloaded from the CDC website[3] in the same SAS file format as before. Because the file included the same attributes found in my original 2021 BRFSS dataset, I applied similar cleaning and feature engineering steps in preparation for running the model.

I then loaded the final XGBoost model from the .pkl file and ran it on the 2019 data to generate predictions. 409,225 respondents were assigned a disease prediction as True or False. I extracted the True prediction probability score calculated by the model as a proxy for a kidney disease risk score. I stratified the risk scores into risk levels using the ranges shown in Table 3. I loaded the 2019 BRFSS data and the model prediction output into a PostgreSQL database for storage.

Finally, I wrote a SQL query to load the data into Tableau, where I developed a dashboard to visualize kidney disease predictions and statistics for the 2019 survey respondents. The dashboard can be viewed on the Tableau Public website[2].

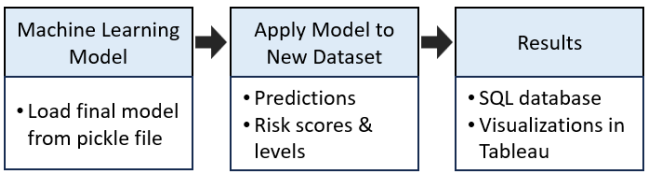

Fig. 8. Model Application Steps

Table 3. Risk Stratification

Level:	Range:
High	75-100
Moderate	50-74
Low	25-49
Minimal	0-24

[2] https://public.tableau.com/app/profile/amanda.hanway/viz/BRFSS2019KidneyDiseasePredictions/Dashboard

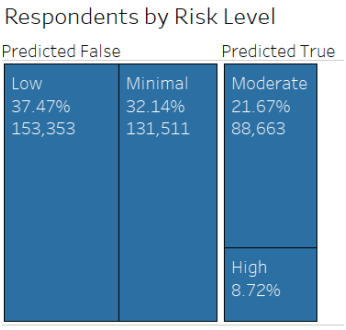

Fig. 9. BRFSS 2019: Respondents by Risk Level

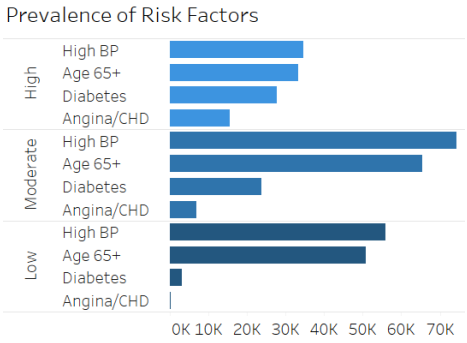

Fig. 10. BRFSS 2019: Prevalence of Risk Factors

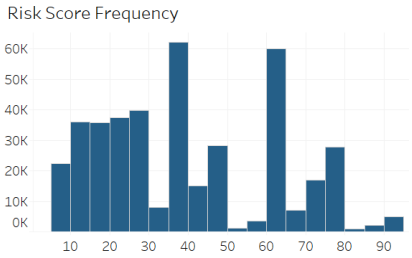

Fig. 11. BRFSS 2019: Risk Score Frequency

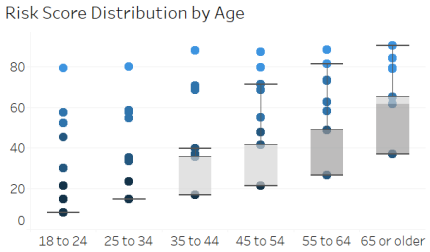

Fig. 12. BRFSS 2019: Risk Score Distribution by Age

3 Results & Analysis

After running the finalized model on the 2019 BRFSS data containing 409,225 respondents, I developed data visualizations in a dashboard form to communicate the results of the model. The dashboard includes a histogram, a box plot, a line chart, a bar chart, and a treemap chart. Each visual contributes a different story point to communicate the outcomes.

The treemap chart in Figure 9 reveals that 30% of the population were predicted to develop kidney disease, while 8.7% of respondents fell into the High Risk category. As shown in the line chart in Figure 13, there is a large uptick for the High or Moderate Risk categories when people reach the age of 65. High blood pressure was the most prevalent risk factor in the Low, Moderate, and High Risk groups as shown in the bar chart in Figure 10. The histogram in Figure 11 illustrates a bimodal risk score distribution, where the scores peak in the ranges of 35-40 and 60-65. The box plot in Figure 12 displays the distribution of scores by age group. Ages 65 and older have the highest median score of 62. An increasing risk score distribution can be seen for ages 35 and over while people under the age of 35 show a lower propensity for developing the disease.

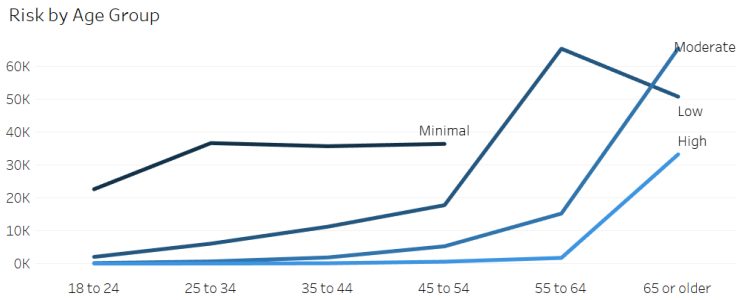

Fig. 13. BRFSS 2019: Risk by Age Group

4 Conclusions & Future Work

The primary goal of this project was to create a machine learning model to predict kidney disease and stratify levels of risk from associated risk factors. This paper detailed the steps followed to develop and evaluate three models - XGBoost, Random Forest and SGD - using different features sets. The XGBoost model was selected as the best-fit model using four features - diabetes, angina_coronary_heart_disease, high_blood_pressure, age_group.

The secondary goal of this project was to apply the model to a new population and create data visualizations on the results. The finalized model was run on the 2019 BRFSS dataset. The output was loaded to a SQL database, then summarized using data visualizations in Tableau.

Based on the results of the model, it can be concluded that age is a top factor in developing kidney disease. People over the age of 65 had a considerably higher risk and a median risk score of 62. People under the age of 45 had a much lower propensity for the disease. Overall, 30% of 409,225 respondents were predicted to be at Moderate or High Risk of developing kidney disease based on risk factors, despite only 4% having reported a history of the disease. This disparity aligns with the research, which indicates a large portion of individuals may not be aware they have the disease.

A limiting factor of this project was the 2021 BRFSS dataset used to train and test the machine learning models. The respondents were predominantly White/Non-Hispanic, and lived in urban counties. It included higher numbers of respondents in older age groups than younger groups. As a result, the models may have given inaccurate predictions for under-represented populations. An area of future work may include seeking samples that are more-representative of all races, ages, and geographic locations, on which to train and re-evaluate the models to improve their accuracy.

References

1. Analytics Vidhya: Introduction to XGBoost Algorithm in Machine Learning, https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/
2. Boykin, April: The Psychology Behind Medical Care Avoidance, https://www.nashvillemedicalnews.com/article/4590/the-psychology-behind-medical-care-avoidance
3. Centers for Disease Control and Prevention: 2019 BRFSS Survey Data and Documentation, https://www.cdc.gov/brfss/annual_data/annual_2019.html
4. Centers for Disease Control and Prevention: 2021 BRFSS Survey Data and Documentation, https://www.cdc.gov/brfss/annual_data/annual_2021.html
5. Centers for Disease Control and Prevention: Chronic Kidney Disease Basics, https://www.cdc.gov/kidneydisease/basics.html
6. Centers for Disease Control and Prevention: Chronic Kidney Disease: Common – Serious – Costly, https://www.cdc.gov/kidneydisease/prevention-risk/CKD-common-serious-costly.html
7. Centers for Disease Control and Prevention: CKD-Related Health Problems, https://www.cdc.gov/kidneydisease/publications-resources/annual-report/ckd-related-health-problems.html
8. Centers for Disease Control and Prevention: LLCP 2021 Codebook Report, https://www.cdc.gov/brfss/annual_data/2021/pdf/codebook21_llcp-v2-508.pdf
9. Centers for Disease Control and Prevention: Survey Data Documentation, https://www.cdc.gov/brfss/data_documentation/index.htm
10. Datacamp: Supervised Machine Learning, https://www.datacamp.com/blog/supervised-machine-learning
11. DaVita Inc.: Living with Comorbidities and Chronic Kidney Disease, https://www.davita.com/education/ckd-life/living-with-comorbidities-and-chronic-kidney-disease#:~:text=Diabetes%20and%20high%20blood%20pressure,and%2For%20high%20blood%20pressure.
12. Healthline: What is kidney disease?, https://www.healthline.com/health/kidney-disease#What-is-kidney-disease?
13. Huilgol, Purva: Precision and Recall — Essential Metrics for Machine Learning (2023 Update), https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/
14. IBM: What is exploratory data analysis?, https://www.ibm.com/topics/exploratory-data-analysis
15. Leonhardt, Megan: Nearly 1 in 4 Americans are skipping medical care because of the cost, https://www.cnbc.com/2020/03/11/nearly-1-in-4-americans-are-skipping-medical-care-because-of-the-cost.html
16. Mirko Stojiljković: ML — Handling Imbalanced Data with SMOTE and Near Miss Algorithm in Python, https://www.geeksforgeeks.org/ml-handling-imbalanced-data-with-smote-and-near-miss-algorithm-in-python/
17. National Kidney Foundation Inc.: How Your Kidneys Work, https://www.kidney.org/kidneydisease/howkidneyswrk
18. Ray, Sunil: Learn How to Use Support Vector Machines (SVM) for Data Science, https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/
19. Scikit-learn: Decision Trees, https://scikit-learn.org/stable/modules/tree.html

20. Scikit-learn: Forests of randomized trees, https://scikit-learn.org/stable/modules/ensemble.html#forest
21. Scikit-learn: Stochastic Gradient Descent, https://scikit-learn.org/stable/modules/sgd.html
22. Sruthi E R: Understand Random Forest Algorithms With Examples (Updated 2023), https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/