Predicting Health

Mining Patient Data To Gain Invaluable Insights

Anthony Le
CSPB 4502 Summer 2024
University of Colorado Boulder
Boulder, CO, USA
anle4634@colorado.edu

Abstract

This project aimed to develop a predictive model for patient outcomes using a comprehensive healthcare dataset related to COVID-19.

Using my model, I sought to answer the following questions:

- Can I predict whether a patient will need intubation?
- Can I predict whether a patient will need to be admitted to the ICU?
- Can I predict the mortality outcome of a patient?

Evaluating my model yielded fairly good results in terms of accuracy and other related metrics that boasted my confidence in utilizing my model to make predictions.

Analysis of my model also yielded interesting findings related to feature correlation and importance of certain features over others when it came to making predictions.

Introduction

The healthcare industry is constantly seeking ways to improve patient care and operational efficiency. Predictive modeling in healthcare analytics offers a promising avenue for achieving these goals. By leveraging large datasets and advanced machine learning techniques, predictive models can help healthcare providers anticipate patient outcomes, estimate treatment costs, predict hospitalization durations and more, helping augment the patient experience and lessen any uncertainty the patient may have about their prognosis.

For my data mining project, I aim to build a predictive model using a COVID-19 dataset (further explanation and analysis of this dataset in a further section). Given the severity of COVID-19, there were three questions that came to mind that I felt a predictive model would be best suited to help answer that I thought was critical to address:

- Can I predict whether a patient will need intubation?
    - This prediction can aid in resource planning and patient prioritization, helping to identify patients at risk of requiring intubation.
    - Healthcare providers can ensure they are adequately and readily equipped with the necessary resources and equipment to be able to prioritize patients that may require immediate intervention.

- Can I predict whether a patient will need to be admitted to the ICU?
    - Predicting ICU admission is crucial in the way of capacity management and early intervention
    - Accurate predictions can help hospitals make rapid decisions to

manage ICU beds more effectively and intervene early to strongly improve patient outcomes and optimize resource allocation within the hospital.

- Can I predict the mortality outcome of a patient?
    - Forecasting mortality outcomes can help providers develop effective treatment strategies and, similar to answering intubation needs, help develop patient prioritization.
    - Understanding which patients are at a higher risk of mortality allows providers to quickly identify which conditions a patient comes in with as being a higher indicator for worse mortality and understanding those indicators can help ensure providers develop tailored treatment plans rather than just a general one that doesn't account for all patients properly.

By leveraging predictive, healthcare facilities can enhance their operation efficiency, improve patient outcomes, and make informed decisions that benefit both patients and healthcare systems.

Answering these questions will help me build insight into the growing field of healthcare analytics and have a better understanding of the mechanics of predictive models that can help aid in providing powerful and useful insights for both healthcare providers and patients alike.

Related Work

The field of healthcare analytics has undergone significant transformation due to advancements in machine learning and artificial intelligence. These technologies have helped enable more precise and predictive models, augmenting healthcare providers' ability to offer personalized and efficient care.

I have identified three pieces of literature that cover different studies and initiatives that have helped me align my approach towards this project.

1 **AI models predicting patient response to immunotherapy**

GE Healthcare & Vanderbilt University Medical Center (VUMC) partnered together to develop AI models that can be utilized to predict how a patient would respond to immunotherapy. This study demonstrated the potential machine learning has in being able to help develop personalized treatment plans for patients, with this study diving more into patients undergoing cancer treatment. The predictive models developed in this study utilized a variety of patient-specific data that ranged from genetic information and more to help improve the accuracy of the predictions being made by the models.

This study underscores the importance of using detailed and personalized data and how it can help in predictive modeling. For my project, I can identify the benefits of incorporating data related to patient demographic and other healthcare indicators as variables that can help provide a good basis to train my model on. The success of the approach taken by the team involved in this study informs my decision to use a wide range of patient-specific features and to not

2 **Cloud-Based Predictive Modeling in the UK**

A small team in the UK was able to develop a cloud-based predictive modeling solution in a

relatively timely manner to enhance healthcare services. This initiative aimed to improve operational efficiency and patient care by leveraging scalable, cloud-based technologies.

This article highlights the scalability of predictive models and the importance of factoring and leveraging various technologies, such as cloud-based technologies, to help build out a successful solution. Although my project will be executed primarily locally, keeping in mind the principles of scalability and efficient resource management is crucial. The large size of the dataset of interest requires efficient handling and processing - finding and researching various technologies and libraries out there will help build a good focus on robust data preprocessing and model optimization techniques to utilize for the project

3 Predictive Analytics and Social Determinants

Research has shown that social determinants of health, from socioeconomic status to education and even access to healthcare, significantly impact patient outcomes. Predictive analytics can help identify those determinants and their overall effects, allowing for more targeted interventions and allow proactive measures to be taken to help ensure patients have success in navigating finding the care they deserve. This study demonstrated how incorporating social factors into predictive models can lead to more comprehensive and effective healthcare solutions.

Incorporating a wide range of factors, including social determinants, can be crucial for building accurate and meaningful predictive models. Incorporating a dataset that includes demographic variables and other relevant health indicators can help build and ensure a holistic approach to prediction - this study in particular informs my selection process, emphasizing the need to consider various determinants of health outside of what one would typically anticipate.

By building on the methodologies and findings of these studies, I aim to develop a robust predictive model that leverages my dataset of interest - incorporating personalized patient data, scalable processing techniques, and a comprehensive feature set can help enable me to address key healthcare questions and gain better insight on how contributions are built within the field of healthcare analytics.

Dataset

Given the nature of HIPAA (Health Insurance Portability and Accountability Act) regulations, it can be very difficult to obtain dataset that has patient electronic health record (EHR) - in this case, a lot of professionals that are want to explore healthcare analytics outside of a hospital setting typically employ the use of datasets that has numerous synthetically generated patient EHRs. Despite the synthetic nature of these datasets, they can still offer invaluable insights into healthcare analysis and patient records. However, that isn't to say that there doesn't exist datasets that contain data and information gathered from real patients - typically health surveys and datasets related to disease statistics are generated from real patients but presented in a way that no discernable information can be utilized to identify a patient.

Keeping that in mind, the dataset I chose to work with for this project is supplied by the Mexican government and was uploaded to Kaggle. This dataset contains 1,048,576 unique patients and 21 unique features. This dataset represents a comprehensive collection of COVID-19 related patient information, including various attributes

related to health conditions, patient demographics, and patient outcomes.

A breakdown of the features is as follows:

- Sex: 1 indicates female and 2 indicates male
- Age: how old is the patient
- Classification: covid-19 test findings with values 1-3 representing patients diagnosed with covid-19 in different degrees of severity and values >= 4 indicates a patient not being a carrier of covid-19 or that the test conducted was inconclusive
- Patient type: describes the type of care a patient received with 1 indicated the patient returned home and 2 indicates the patient was hospitalized
- Pneumonia: 1 indicates they have pneumonia and 2 indicates they do not
- Pregnancy: 1 indicates they are pregnant and 2 indicates they are not
- Diabetes: 1 indicates they have diabetes and 2 indicates they do not
- COPD: 1 indicates they have chronic obstructive pulmonary disease and 2 indicates they do not
- Asthma: 1 indicates they have asthma and 2 indicates they do not
- Inmsupr: 1 indicates they are immunosuppressed and 2 indicates they are not
- Hypertension: 1 indicates they have hypertension and 2 indicates they do not
- Cardiovascular: 1 indicates they have a heart or blood vessel related disease and 2 indicates they do not
- Renal chronic: 1 indicates they have chronic renal disease and 2 indicates they do not
- Other disease: 1 indicates they have some other unspecified disease and 2 indicates they do not
- Obesity: 1 indicates they are obese and 2 indicates they are not
- Tobacco: 1 indicates they are a tobacco user and 2 indicates they are not
- USMER: describes which floor the unit the patient was treated in
- Medical Unit: describes the type of institution of the National Health System provided the care for the patient
- Intubed: 1 indicates they were placed on a ventilator and 2 indicates they were not
- ICU: 1 indicates they were admitted into the ICU and 2 indicates they were not
- Date Died: A valid date indicates the patient died and an invalid date indicates the patient is alive
- For all of these features, additional values of 97, 98, or 99 may populate them - this value is used as an indicator for missing data

Oftentimes, medical data is contained in various spreadsheets to cover different features and can be more complicated to work with when multiple files are related to a singular dataset of interest. However, this particular dataset is all contained within a singular CSV file, making it very easy and approachable to explore and manipulate for my project. Additionally, all of the features of the dataset (with the exception of Age which is numerical data) can be identified as categorical data; information sorted into predefined groups or classes.

Workflow

Setting out to attempt to build out a predictive model was quite the undertaking, especially given that prior to this course, most of what I knew was in passing from various news articles

whenever the media leans into the phenomenon. To ensure that I would be able to deliver something of substance with this project, I made sure to methodically plan out the entire flow of the project. Here is what the process looked like for this project:

1 Data Collection and Processing

As mentioned before, the dataset that will be utilized for this project is a COVID-19 dataset covering cases in Mexico that was published by the Mexican government, collected. After publication, the dataset was then also uploaded to Kaggle, which is where I downloaded the dataset from. The dataset is contained within a singular CSV file - this makes it easy to approach and offers a lot of flexibility in being able to easily manipulate the data using various Python libraries, in particular I utilized Pandas a lot. After downloading the data and setting up my local environment on my machine, I created a Jupyter Notebook to be able to better break up the entirety of the process of data preprocessing.

The process first began with importing the Pandas library to get access to a variety of different modules and functions to help aid in the process of diving into the data. The dataset was loaded into the notebook as a DataFrame. To ensure the dataset was loaded correctly, I used the pandas.head() function to verify that the notebook could read and return the first 5 elements of the DataFrame. Once this was confirmed, I was good to go to write various lines of code to help aid in the process of walking through the data to identify things of interest to look out for or keep in mind when I go through the process of cleaning the data to be suitable for the predictive model that will be built later. In particular, pandas.shape and pandas.columns.values were helpful to be able to generate a quick snapshot of the dataset - shape allowed quick feedback of how many records I would be working with in this dataset while pandas.columns.values was particularly helpful in returning an array that contains all of the features or attributes of the dataset that each record would have a value for (a breakdown of the 21 features can be found and described in the Dataset section above). To conclude the walkthrough of the data, I wrote a simple for loop to iterate through each of the columns of the dataset and return the total count for each value that could be found for that particular feature - this was a great way to be able to see a general distribution of values of the dataset.

This dataset walkthrough provided clear insights of things that needed to be addressed:

- An entire feature was misspelled within the dataset; there is a column ‘HIPERTENSION’ would should be corrected and renamed as ‘HYPERTENSION’
- As mentioned from before, values of 97, 98, or 99 are utilized as indicators for missing data for that particular feature. There were a variety of different features in the dataset that had a count for these values
- For records where patients were male (‘SEX’ = 2), they had missing data related to pregnancy (‘PREGNANT’ = 97, 98, or 99). To keep the values consistent with records where patients were female (‘SEX’ = 1), I would need to address and change these values to be ‘PREGNANT’ = 2 to reflect not pregnant.
    - Assumption is being made that for records where the patients were male, they are biologically male and cannot be pregnant
- Additionally, for records where patients were female (‘SEX’ = 1), there were some records where they had missing data related to pregnancy. Unlike before with records with male patients, I cannot just assume that they are not pregnant, I will address the values to be NaN to later

perform data imputation over to generate values for this feature
    - Assumption is being made that for records where the patients were female, they are biologically female and are capable of getting pregnant
- For records where patients were sent home ('PATIENT_TYPE' = 1), they had missing data related to two features; if they were placed on a ventilator ('INTUBED' = 97, 98, or 99) and if they were admitted to the ICU ('ICU' = 97,98, or 99). For these features, I will go and change the value to be 2 to reflect that they were not placed on a ventilator nor were they admitted to the ICU
    - Assumption is being made that for records where the patient was sent home, it was such that they were sent home because they did not need an escalation of care to be placed on a ventilator or admitted to the ICU and was able to administer self-care and treatment at home
- For records related to different medical conditions ('PNEUMONIA', 'DIABETES', 'COPD', 'ASTHMA', 'INMSUPR', etc.), there was missing data related to these features. Similar to the case of female patients and missing pregnancy data, I will go and change these values to be NaN, as there is no reasonable assumption that can be made where these patients are incredibly healthy and don't have these medical conditions. Replacing these values with NaN values will set me up to apply imputation over these values.
- I can generate a new feature 'ALIVE' that takes into account the values found for the feature 'DATE_DIED". Using the same value as the other Boolean features (1 denotes yes and 2 denotes no), for patient records that have a valid date, they will have a value of 2 associated with the new feature 'ALIVE' and for patient records that does not have a valid date (9999-99-99), they will have a value of 1 associated with the new feature 'ALIVE'.
- Although generally, values of 97, 98, or 99 are utilized to denote missing data for a particular feature, I need to be careful during the cleaning process to factor in that these values are valid for the feature 'AGE'. It can be very easy to forget this and accidentally introduce NaN values to a feature that has all valid data

With a clear vision ahead of the features I need to dive into and the values that need to be addressed, I wrote a code to print a simple report about just how many values permeated the dataset as being missing data. An initial count yielded the following numbers:

	97	98	99
INTUBED	848544	0	7325
PNEUMONIA	0	0	16003
AGE	135	124	86
PREGNANT	523511	3754	0
DIABETES	0	3338	0
COPD	0	3003	0
ASTHMA	0	2979	0
INMSUPR	0	3404	0

HYPERTENSION	0	3104	0
OTHER_DISEASE	0	5045	0
CARDIOVASCULAR	0	3076	0
OBESITY	0	3032	0
RENAL_CHRONIC	0	3006	0
TOBACCO	0	3220	0
ICU	848544	0	7488

From this report, we can see that there is a lot of missing data spread amongst these features; omitted from this table is reports related to features that had no missing data. There are a lot of values that need to be addressed. The process of cleaning here is fairly straight-forward. First, I utilized pandas.copy() to create a new DataFrame that will store all of the cleaned data - this is so I can easily reference back to the original DataFrame if I need to without going through the process of reading and loading the original CSV again. After the copy was created, the easiest value that needed addressing was correcting the spelling of the feature 'HIPERTENSION' - pandas.rename() made this very easy to change the name from 'HIPERTENSION' to 'HYPERTENSION'. The new function, pandas.loc(), was utilized a lot to help write code that could easily access various features that needed to be addressed and was handled with 1 to 2 lines of code. For the cleaning related to values found in the features related to medical conditions, I created an array that contained all of the features and wrote a loop to iterate through this new array and using pandas.loc, I can go and address the values of interest. For the creation of the new feature 'ALIVE', I first wrote code to convert the values in 'DATE_DIED' to datetime format. Using these newly converted values, I utilize numpy.where to insert new values to 'ALIVE' based on if a valid or invalid date was being parsed and read. Last step was dropping the 'DATE_DIED' feature from the newly created DataFrame as this was something not needed since we encoded that feature into our new feature 'ALIVE'. As mentioned before, I need to ensure that the feature 'AGE' was left untouched as the values 97, 98, and 99 represented valid ages that patients could have and were not in fact missing data in this context.

After completing the cleaning process up to here, I generated a new report to go through the newly created DataFrame and counted how many values that correlated to missing data still existed - it is important to double check my work as I can't just blindly trust that the code I written completely handled all cases of interest. The report generated the following numbers:

	97	98	99
INTUBED	0	0	7325
AGE	135	124	86
INMSUPR	0	3404	0
ICU	0	0	7488

As before, any features that had no missing data were omitted. From this snapshot, we can see that the new DataFrame was a lot healthier than where I was before the cleaning process. Additionally, the feature 'AGE' did not lose any value count from before the cleaning process to after, indicating that that feature was left

untouched in the process of cleaning the values. However, there are still values that were missed and not completely taken care of. To address this, I just need to run the cleaning process once more over the new DataFrame. This was easily handled with a simple loop that iterates through each column of the DataFrame (while ignoring 'AGE') and replaces the values associated with missing data with NaN. Running a new report generated numbers that indicate that there was no longer any count of values 97, 98, or 99 in any of the features with the exception of 'AGE' would still have the same count after this second step of cleaning.

With cleaning done, I wrote code to give a brief overview into the cleaned DataFrame that uses most of the same code as when I did my initial exploration before all of the cleaning. Additionally, more code was written so that I can generate a quick report on the count of NaN values within the DataFrame along with a quick calculation of how many patient records are affected and have missing values. This report showed that there were a total of 67,777 NaN values and a total of 28,909 unique patient records that had at least one missing value.

With this initial understanding of how many NaN values can be found in the cleaned DataFrame along with how many unique patient records were affected, I could decide how I wanted to handle these missing values.

Given this dataset is comprised of 1,048,576 unique patient records, having only 28,909 unique patient records with missing values was rather miniscule and potentially didn't need to be addressed with imputation and could be easier and more efficient to drop these records from the cleaned DataFrame

To save on time and resources, I found it was in fact more efficient to handle these missing values by dropping the 28,909 affected patient records. Dropping these records resulted in only losing roughly 2.76% of the overall data from the original count of 1,048,576 unique patient records. However, as a way to get practice in imputation, I still wrote a block of code in the notebook that covers filling in these missing values. Given that most of these features were Boolean features and I consider that the missing values were simply missing at random, imputation with the mode of these features can help preserve the overall distribution of values within the feature of interest. Once again however, for the model, I am electing to drop the records rather than to address them with imputation. This would mark the end of the data collection and cleaning process.

2 Model Development

Now that the missing data has been taken care of and I now have a cleaned dataset to work with, I can begin the process of developing a predictive model. The first step is to determine which algorithm I want to utilize for the project.

There are a variety of different algorithms that exist with literature supporting them that can be worth exploring, such as linear regression, random forest, and gradient boosting.

Linear regression models the relationship between a dependent variable and one or more independent variables, assuming a linear relationship between inputs and outputs. This type of model is usually easier to understand, interpret, and implement and can come across as being computationally less intensive compared to other more complex models. These traits can help with making predictions on hospitalization durations or treatment costs where the end output is to estimate a numeric value. However, a shortfall would be that if a

nonlinear relationship exists in our data, the model wouldn't be effective in that case.

Random forest is built upon multiple decision trees that are merged to achieve better accuracy and stability related to making predictions. This algorithm is rather robust and can handle overfitting (the phenomenon where an algorithm can fit too closely to the training model) better than that of a single decision tree. There is some versatility where the algorithm can be used well in tasks related to classification and regression and for the project, can be helpful in predicting treatment costs and patient outcomes through handling complex interactions between features. However, they can be harder to interpret compared to a single decision tree and take up a decent amount of computational resources to run.

Gradient boosting is an algorithm that builds models sequentially and iteratively - each new model that is generated corrects errors made by the previous model and through this process, works to combine the predictions of multiple weaker learners to produce a strong learner that is capable of making good and accurate predictions. This algorithm has a variety of advantages with it that range from offering higher predictive performance to being flexible in the tasks it can handle (similar to random forest). For the project, it can be suitable for predicting patient outcomes and treatment costs where high prediction accuracy is much more paramount. A caveat though is that training time can be rather slow and for larger datasets, this can be a huge slump in the project progression as a hangup.

Ultimately, the decision of which algorithm to build with will also take into consideration critical questions I was interested in answering and exploring, as mentioned previously. Given the categorical nature of most of the data in this dataset, I felt it would be most effective to utilize the Random Forest algorithm. There were several compelling reasons that aligned well with the nature of this dataset. Random Forests are effective with handling categorical data with how they handle binary and categorical features without extensive preprocessing. Additionally, Random Forests have additional abilities to provide insights into feature importance that allows me to do analysis over the features of the dataset and identify which patient attributes had more weight in predicting critical outcomes. The primary technique that I felt would be most beneficial for this project was classification - predicting discrete outcomes based on input features; Random Forest is able to handle classification tasks.

For the creation of the model, the sklearn library provides plenty of useful functions that can help expedite and simplify this process. The model development process began with first splitting the dataset into features and targets for each prediction task. Features were defined as all columns with the exception of target variables.

- 'X' contained all features excluding our target variables 'INTUBED', 'ICU', and 'ALIVE'
- 'y_intubed', 'y_icu', and 'y_alive' were our target variables for the our prediction tasks related to intubation needs, ICU admission, and mortality

For each prediction task, the dataset was divided into training and testing sets using an 80:20 split; this split ratio was used so that a large amount of the dataset was used to train and fit the model and learn the underlying patterns and relationships that exist within the data while reserving a small subset of the data to provide an unbiased evaluation of the model's performance. After the completion of this initial setup, I can initialize a Random Forest using RandomForestClassifier() from sklearn.ensemble and do this for all three prediction tasks, which

after the model is created and train, I can make predictions using the remaining 20% of the dataset (given that the dataset was originally split using the 80:20 ratio described previously).

Evaluation and Key Results

To evaluate the performance of my predictive model, the metrics module of the sklearn library provides plenty of useful functions that can calculate a variety of metrics that can provide useful insight to the performance of my model. In particular, these were the following metrics I wanted to generate a report on:

- Accuracy: Measures the proportion of correct predictions made by the model
- Precision: Measures the accuracy of the positive predictions made by the model
- Recall: Measures the ability of the model to identify all relevant cases
- F1 Score: A metric that relates to the harmonic mean of precision and recall that helps provide a singular metric that balances both of these concerns.
- Macro Average: Measures the average performance of the model across all classes, treating each class equally regardless of its frequency in the dataset
- Weighted Average: Measures the model's performance by taking into account the frequency of each class in the dataset, giving more importance to classes with more instances.

Using the metrics module from sklearn, I obtained the following evaluation results:

Intubation Prediction

Class	Precision	Recall	F1-Score	Support
1.0	0.27	0.11	0.15	6,582
2.0	0.97	0.99	0.98	197,352
Accuracy	-	-	-	0.96
Macro Avg	0.62	0.55	0.57	203,934
Weighted Avg	0.95	0.96	0.95	203,934

ICU Admission Prediction

Class	Precision	Recall	F1-Score	Support
1.0	0.25	0.09	0.13	3,252
2.0	0.99	1.00	0.99	200,682
Accuracy	-	-	-	0.98
Macro Avg	0.62	0.54	0.56	203,934
Weighted Avg	0.97	0.98	0.98	203,934

Mortality Prediction

Class	Precision	Recall	F1-Score	Support
1.0	0.97	0.99	0.98	198,155
2.0	0.23	0.08	0.11	5,779

Accuracy	-	-	-	0.97
Macro Avg	0.60	0.53	0.55	203,934
Weighted Avg	0.95	0.97	0.96	203,934

There are a lot of values here to discuss and interpret, but overall, here is a simple breakdown of the evaluation of the model in each of the predictive task:

- Intubation: The model shows high accuracy overall but struggles with predicting intubation cases accurately, as indicated by the low precision and recall values
- ICU Admission: The model performs well overall, but similarly to intubation, it has poor precision and recall for predicting ICU admissions
- Mortality: The model excels in predicting patient mortality with high precision and recall, but particularly struggles with the no-mortality class

The evaluation results provide a nuanced picture of the model's performance, highlighting both its strengths and limitations. On average, the Random Forest models exhibited high accuracy across the dataset, indicating that they effectively predict the outcomes of the prediction tasks of interest - intubation, ICU admission, and mortality. The strong performance is reflected in the high accuracy scores, which suggest that the model is generally proficient in making correct predictions for a majority of the cases.
However, deeper analysis in the results reveals significant weaknesses related to detecting rare but critical events. This challenge is primarily due to the pronounced class imbalance within the dataset. The model performs exceptionally well in predicting the majority class but struggles with identifying the minority class.

To enhance the model's effectiveness and robustness, several steps should be taken for future consideration. Strong consideration should be given towards implementing strategies to balance the class distribution within the dataset. Techniques such as oversampling the minority class or undersampling the majority class can help to better represent critical cases and improve the model's sensitivity. Additionally, adjusting the class weights within the Random Forest algorithm to give more importance to the minority classes can be beneficial. By increasing the weight of the critical events, the model will be incentivized to focus more on accurately predicting these less frequent but important outcomes. Finally, continuously refining the model through the incorporation of new data can help provide regular validation of the model that can help in maintaining the model's relevance and effectiveness over time. Addressing these areas can help the model provide a more balanced and accurate prediction of both frequent and rare events, ultimately leading to better healthcare decision-making and resource allocation.

The journey of developing predictive models, especially in the healthcare domain, has been both enlightening and challenging. Throughout this process, I have gained a deeper understanding of the intricacies involved in selecting, implementing, and refining algorithms to address real-world problems. The process of learning how to manipulate a dataset and develop a workflow related to exploring and cleaning the dataset to have it in a clean and usable state for model training introduced an appreciation for the professionals that do this

work daily. Each stage of model development - from selecting algorithms to addressing data challenges - requires a thoughtful and methodical approach.

Application

The knowledge and insights gained from developing and evaluating predictive models have significant implications for various practical applications, especially within the healthcare domain. These applications can transform how healthcare professionals approach patient care, resource optimization, and help improve overall health outcomes for patients.

Predictive models can assist in the early detection of critical health conditions by identifying patients at high risk for outcomes such as intubation, ICU admission, or mortality as was the case for my model. This can be expanded even further to cover other outcomes such as disease remission or evaluating effectiveness of a drug or treatment plan.

That is just a small snippet of practical applications of predictive modeling - the applications of predictive modeling in healthcare extend far beyond the initial model development and evaluation. By translating the insights gained from actionable strategies, healthcare professionals, administrators, researchers, and more can enhance patient care, optimize resource utilization, and contribute to more effective and personalized healthcare delivery. The potential for predictive modeling to drive improvements across various aspects of healthcare underscores the importance of continued research, innovation, and application of these models in real-world settings where more can be done to help improve patient outcomes.

REFERENCES

[1] Vanderbilt University Medical Center. 2024. GE Healthcare, Vanderbilt Public Data on AI Models Predicting Patient Response to Immunotherapy. Retrieved from https://news.vumc.org/2024/03/05/ge-healthcare-vanderbilt-publish-data-on-ai-models-predicting-patient-response-to-immunotherapy/

[2] Amazon Web Services Public Sector Blog. 2024. One Small Team Created Cloud-Based Predictive Modeling Solution to Improve Healthcare Services in the UK. Retrieved from https://aws.amazon.com/blogs/publicsector/one-small-team-created-cloud-based-predictive-modeling-solution-improve-healthcare-services-uk/.

[3] Health IT Analytics. 2024. Improving Social Determinants of Health with Predictive Analytics. Retrieved from https://healthitanalytics.com/news/improving-social-determinates-of-health-with-predictive-analytics.