Accurate Prediction of Optimal Tracheal Tube Depth in Pediatric Patients

Data to Paper

February 20, 2024

Abstract

Pediatric patients requiring respiratory support often experience complications due to misplaced tracheal tube tips, emphasizing the critical need to accurately determine the Optimal Tracheal Tube Depth (OTTD). We conducted a comprehensive analysis using a dataset of pediatric patients aged 0-7 years who underwent surgery and required post-operative mechanical ventilation. The dataset includes patient characteristics and OTTD determined by chest X-ray. By developing a machine learning model incorporating sex, age, height, and weight, we achieved superior predictive power compared to formula-based models. Our model significantly outperformed the formula-based model in predicting the OTTD, as demonstrated by lower mean squared residuals. These findings highlight the effectiveness of incorporating patient characteristics in optimizing tracheal tube placement. However, further refinement and validation are required to enhance precision and safety. Our study contributes to improving patient care and clinical outcomes in pediatric patients requiring respiratory support. Future research should focus on expanding the model with additional patient parameters and conducting prospective studies to validate its effectiveness.

Results

To determine the optimal tracheal tube depth (OTTD) in pediatric patients, we conducted a comprehensive analysis using a dataset of pediatric patients aged 0-7 years who underwent surgical procedures and required post-operative respiratory support. The dataset included 969 patients, and for each patient, the OTTD was determined by chest X-ray.

First, to understand the descriptive statistics of height and age stratified by sex, we calculated the mean and standard deviation for each variable.

As shown in Table 1, the mean height was 65.4 cm (SD=18.7) for females and 66.5 cm (SD=19.4) for males. The mean age was 0.732 years (SD=1.4) for females and 0.781 years (SD=1.47) for males. These findings provide important insights into the patient characteristics in our dataset.

Table 1: Descriptive statistics of Height and Age stratified by Sex

	Height		Age	
	mean	std	mean	std
Female	65.4	18.7	0.732	1.4
Male	66.5	19.4	0.781	1.47

Height: Height(cm)
Age: Age(years)

Next, we compared the predictive power of a machine learning (ML) model with a formula-based model in determining the OTTD. We used a Random Forest model and performed hyperparameter tuning to optimize the ML model's performance. The ML model showed a mean squared residual of 1.49, significantly outperforming the formula-based model, which had a mean squared residual of 3.81 (see Table 2). The paired t-test, with a p-value $< 10^{-6}$, confirmed the statistically significant superiority of the ML model. These results highlight the effectiveness of incorporating patient characteristics such as sex, age, height, and weight in predicting the OTTD.

Table 2: Comparison of predictive power: ML Model vs. Formula-based Model

Model	Mean Squared Residual	p-value
ML Model	1.49	$<10^{-6}$
Formula-based Model	3.81	-

ML Model: Machine Learning Model

Finally, we examined the RF model's performance using additional metrics. The RF model achieved a train score of 0.6642 and a test score of 0.5803. The best parameters for the RF model were found to be a max depth of 5, min samples split of 5, and 100 estimators. These results demonstrate the model's ability to generalize well on unseen data and its reasonable predictive accuracy.

Taken together, these results highlight the superiority of the ML model

over the formula-based model in predicting the OTTD in pediatric patients. Our findings emphasize the importance of incorporating patient characteristics into the decision-making process for tracheal tube depth determination. The RF model, with its optimized parameters, provides a valuable tool for accurately determining the OTTD in pediatric patients requiring respiratory support.

A Data Description

Here is the data description, as provided by the user:

```
Rationale: Pediatric patients have a shorter tracheal length than adults;
    therefore, the safety margin for tracheal tube tip positioning is narrow.
Indeed, the tracheal tube tip is misplaced in 35%{50% of pediatric patients and
    can cause hypoxia, atelectasis, hypercarbia, pneumothorax, and even death.
Therefore, in pediatric patients who require mechanical ventilation, it is
    crucial to determine the Optimal Tracheal Tube Depth (defined here as `OTTD`,
    not an official term).

Note: For brevity, we introduce the term `OTTD` to refer to the "optimal
    tracheal tube depth". This is not an official term that can be found in the
    literature.

Existing methods: The gold standard to determine OTTD is by chest X-ray, which
    is time-consuming and requires radiation exposure.
Alternatively, formula-based models on patient features such as age and height
    are used to determine OTTD, but with limited success.

The provided dataset focus on patients aged 0-7 year old who received post-
    operative mechanical ventilation after undergoing surgery at Samsung Medical
    Center between January 2015 and December 2018.
For each of these patients, the dataset provides the OTTD determined by chest
    X-ray as well as features extracted from patient electronic health records.


1 data file:

"tracheal_tube_insertion.csv"
The csv file is a clean dataset of 969 rows (patients) and 6 columns:

Tube:
#1 `tube` - "tube ID", internal diameter of the tube (mm) [Included only for the
    formula-based model; Do not use as a machine-learning model feature]

Model features:
#2 `sex` - patient sex (0=female, 1=male)
#3 `age_c` - patient age (years, rounded to half years)
```

```
#4 `ht` - patient height (cm)
#5 `wt` - patient weight (kg)

Target:
#6 `tube_depth_G` - Optimal tracheal tube depth as determined by chest X-ray (in
    cm)
```

B Data Exploration

B.1 Code

The Data Exploration was carried out using the following custom code:

```
1
2  import pandas as pd
3  import numpy as np
4
5  # Load the data
6  df = pd.read_csv('tracheal_tube_insertion.csv')
7
8  # Open our output file
9  with open('data_exploration.txt', mode='w') as f:
10
11     # Data Size
12     shape = df.shape
13     f.write('# Data Size\n')
14     f.write(f'Rows: {shape[0]}, Columns: {shape[1]}\n\n')
15
16     # Summary Statistics
17     f.write('# Summary Statistics\n')
18     desc = df.describe(include='all')
19     f.write(f'{desc}\n\n')
20
21     # Check categorical variables
22     f.write('# Categorical Variables\n')
23     categorical = df.select_dtypes(include=['object'])
24     for name in categorical.columns:
```

```
        top_5 = df[name].value_counts().nlargest(5)
        f.write(f'{name}: \n{top_5}\n\n')

    # Missing Values
    f.write('# Missing Values\n')
    total_missing = df.isnull().sum().sum()
    f.write(f'Total missing/undefined values: {total_missing}\n')

    # Check for special numeric values stand for unknown/undefined if any
    f.write('Unknown/Undefined numeric values:\n')
    for col in df.select_dtypes(include=[np.number]):
        unknowns = df[col].isin([-999, -9999]).sum()
        if unknowns > 0:
            f.write(f'{col}: {unknowns} Unknown/Undefined values\n')

    # Count of each sex
    f.write('\n# Sex Distribution\n')
    sex_count = df['sex'].value_counts()
    f.write(f'Males: {sex_count[1]}, Females: {sex_count[0]}\n')

```

B.2 Code Description

The provided code performs data exploration on a dataset of pediatric patients who received post-operative mechanical ventilation after surgery. The exploration is conducted to gain insights into the dataset and understand its characteristics.

Initially, the code loads the dataset into a pandas DataFrame.

Next, the code generates the following analysis steps and writes the results into the "data_exploration.txt" file:

1. Data Size: The code determines the number of rows and columns in the dataset and writes this information to the file.

2. Summary Statistics: The code calculates summary statistics for each column of the dataset, including count, mean, standard deviation, minimum, quartiles, and maximum values. These statistics provide a broad overview of the distribution and ranges of the numerical features in the dataset.

3. Categorical Variables: The code identifies the categorical variables in the dataset and determines the top 5 most frequent values for each categorical variable. This analysis step helps understand the prevalence of different categories within each variable.

4. Missing Values: The code counts the total number of missing or undefined values in the dataset and writes this information to the file. This step allows for an assessment of the completeness of the dataset and the potential need for handling missing values.

5. Unknown/Undefined Numeric Values: The code checks for any specific numeric values that are used to represent unknown or undefined data. It identifies columns with such values and writes the column names and the count of unknown/undefined values to the file. This analysis helps identify any special codes used to represent missing or invalid data.

6. Sex Distribution: The code calculates the count of each sex category in the dataset. Specifically, it determines the number of males and females and writes these counts to the file. This analysis provides an understanding of the gender distribution in the dataset.

Overall, the code performs comprehensive data exploration on the provided dataset, providing valuable information about its size, summary statistics, categorical variables, missing values, unknown/undefined numeric values, and sex distribution. The results of this exploration are written to the "data_exploration.txt" file for reference and further analysis.

B.3 Code Output

data_exploration.txt

```
# Data Size
Rows: 969, Columns: 6

# Summary Statistics
          tube    sex  age_c    ht    wt  tube_depth_G
count      969    969    969   969   969           969
mean     3.694 0.5387  0.758    66 7.127         10.19
std     0.5681 0.4988   1.44 19.08 4.774         1.766
min        2.5      0      0  29.5  0.57           5.9
25%        3.5      0      0    52  3.56             9
50%        3.5      1      0  61.5   5.7           9.8
75%          4      1      1    76   9.5          11.2
max          6      1      7 135.4    31          19.2
```

```
# Categorical Variables
# Missing Values
Total missing/undefined values: 0
Unknown/Undefined numeric values:

# Sex Distribution
Males: 522, Females: 447
```

C Data Analysis

C.1 Code

The Data Analysis was carried out using the following custom code:

```
1
2  # IMPORT
3  import pandas as pd
4  import numpy as np
5  from scipy import stats
6  import statsmodels.api as sm
7  from sklearn.ensemble import RandomForestRegressor
8  from sklearn.model_selection import train_test_split,
   ↪  GridSearchCV
9  import pickle
10
11 # LOAD DATA
12 data = pd.read_csv('tracheal_tube_insertion.csv')
13
14 # DATASET PREPARATIONS
15 # No dataset preparations are needed.
16
17 # DESCRIPTIVE STATISTICS
18 ## Table 0: "Descriptive statistics of height and age
   ↪  stratified by sex"
19 df0 = data.groupby('sex').agg({'ht':['mean', 'std'],
   ↪  'age_c':['mean', 'std']})
20 df0.index = ['Female', 'Male']
21 df0.to_pickle('table_0.pkl')
```

```python

# PREPROCESSING
# No preprocessing is needed.

# ANALYSIS
## Table 1: "Comparison of predictive power: ML model vs. Formula-based model"
# Define target and features
target = data['tube_depth_G']
features = data[['sex', 'age_c', 'ht', 'wt']]

# split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.3, random_state=0)

# Define the Random Forest model
rf = RandomForestRegressor()

# define parameters for GridSearchCV
param_grid = {'n_estimators': [50, 100], 'max_depth': [None, 5, 10], 'min_samples_split': [2, 5]}

# Perform hyperparameter tuning
grid = GridSearchCV(rf, param_grid, cv=5)
grid.fit(X_train, y_train)

# Predict with ML model
ml_preds = grid.predict(X_test)

# Predict with formula-based model
fb_preds = X_test['ht'] / 10 + 5

# Compute square residuals
ml_sq_res = np.square(ml_preds - y_test)
fb_sq_res = np.square(fb_preds - y_test)

# Perform paired t-test
t_results = stats.ttest_rel(ml_sq_res, fb_sq_res)

# Create Table 1's DataFrame
```

```
df1 = pd.DataFrame({'Model': ['ML Model', 'Formula-based Model'],
                    'Mean Squared Residual': [ml_sq_res.mean(), fb_sq_res.mean()],
                    'p-value': [t_results.pvalue, '-']})

df1.set_index('Model', inplace=True)
df1.to_pickle('table_1.pkl')

# SAVE ADDITIONAL RESULTS
rf_train_score = grid.score(X_train, y_train)
rf_test_score = grid.score(X_test, y_test)
best_params = grid.best_params_
additional_results = {
 'RF train score': rf_train_score,
 'RF test score': rf_test_score,
 'RF best parameters': best_params
}

with open('additional_results.pkl', 'wb') as f:
    pickle.dump(additional_results, f)
```

C.2 Code Description

The provided code conducts an analysis on a dataset related to determining the optimal tracheal tube depth (OTTD) in pediatric patients who require mechanical ventilation. The dataset consists of 969 patient samples.

The analysis consists of the following steps:

Step 1: Loading the dataset The code reads the dataset file "tracheal_tube_insertion.csv" using the pandas library and stores it in a DataFrame variable called "data".

Step 2: Descriptive Statistics The code calculates descriptive statistics on the height and age of the patients stratified by sex. It groups the data by sex and computes the mean and standard deviation of height and age for female and male patients separately. The results are saved in a pickled DataFrame named "table_0.pkl".

Step 3: Preprocessing No preprocessing steps are required for this analysis.

Step 4: Analysis The analysis involves comparing the predictive power of

a machine learning (ML) model against a formula-based model. The target variable is the tracheal tube depth determined by chest X-ray (tube_depth_G), and the feature variables are sex, age, height, and weight.

The code splits the dataset into training and testing sets using the train_test_split function from the sklearn library. It then defines a Random Forest model and performs hyperparameter tuning using GridSearchCV to find the best combination of parameters (n_estimators, max_depth, and min_samples_split).

Next, the code predicts the tube depth using both the ML model and the formula-based model. For the formula-based model, the tube depth is estimated based on the patient's height divided by 10 and added to 5.

The code computes the mean squared residuals for both the ML model and the formula-based model by taking the square of the difference between the predicted tube depth and the actual tube depth.

Finally, the code performs a paired t-test to compare the mean squared residuals of the two models and saves the results in a pickled DataFrame named "table_1.pkl". Additionally, it saves the training and testing scores of the ML model, as well as the best parameters found during the hyperparameter tuning, in a pickled dictionary named "additional_results.pkl".

The "additional_results.pkl" file contains the following information: - 'RF train score': The R-squared score of the Random Forest model on the training set. - 'RF test score': The R-squared score of the Random Forest model on the testing set. - 'RF best parameters': The best combination of hyperparameters found during the hyperparameter tuning process.

This analysis provides insights into the predictive power of a machine learning model compared to a formula-based model for determining the optimal tracheal tube depth in pediatric patients requiring mechanical ventilation.

C.3 Code Output

table_0.pkl

	ht		age_c	
	mean	std	mean	std
Female	65.400447	18.701462	0.731544	1.402500
Male	66.514368	19.403722	0.780651	1.472808

table_1.pkl

	Mean Squared Residual	p-value
Model		
ML Model	1.488980	1.101e-12
Formula-based Model	3.805471	-

additional_results.pkl

```
{
    'RF train score': 0.6642           ,
    'RF test score': 0.5803           ,
    'RF best parameters': {'max_depth': 5, 'min_samples_split': 5,
    'n_estimators': 100},
}
```

D LaTeX Table Design

D.1 Code

The LaTeX Table Design was carried out using the following custom code:

```

# IMPORT
import pandas as pd
import pickle
from typing import Dict, Tuple, Any, Optional
from my_utils import to_latex_with_note, format_p_value,
↪  is_str_in_df, split_mapping, AbbrToNameDef

# PREPARATION FOR ALL TABLES
# Define shared mapping
shared_mapping: AbbrToNameDef = {
 'ht': ('Height', 'Height(cm)'),
 'age_c': ('Age', 'Age(years)'),
 'sex': ('Sex', '0: Female, 1: Male'),
 'tube_depth_G': ('OTTD', 'Optimal Tracheal Tube Depth(cm)')
}

# TABLE 0
df0 = pd.read_pickle('table_0.pkl')
```

```python

# Format Values
# No formatting of values in this table.

# Rename Rows and Columns
mapping0 = {k: v for k, v in shared_mapping.items() if
    ↪ is_str_in_df(df0, k)}
abbrs_to_names, legend = split_mapping(mapping0)
df0 = df0.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex
to_latex_with_note(df0, 'table0.tex',
  caption='Descriptive statistics of Height and Age stratified
    ↪ by Sex',
  label='table:descr_stats_height_age',
  legend=legend)

# TABLE 1
df1 = pd.read_pickle('table_1.pkl')

# Format Values
df1['p-value'] = df1['p-value'].apply(format_p_value)

# Rename Rows and Columns
mapping1 = {k: v for k, v in shared_mapping.items() if
    ↪ is_str_in_df(df1, k)}
# Add 'ML Model' to mapping1
mapping1['ML Model'] = ('ML Model', 'Machine Learning Model')
abbrs_to_names, legend = split_mapping(mapping1)
df1 = df1.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex
to_latex_with_note(df1, 'table1.tex',
  caption='Comparison of predictive power: ML Model vs.
    ↪ Formula-based Model',
  label='table:compare_ML_FBM',
  legend=legend)

```

D.2 Provided Code

The code above is using the following provided functions:

```python
def to_latex_with_note(df, filename: str, caption: str, label: str, note: str = None, legend: Dict[str, str] = None, **kwargs):
 """
 Converts a DataFrame to a LaTeX table with optional note and legend added below the table.

 Parameters:
 - df, filename, caption, label: as in `df.to_latex`.
 - note (optional): Additional note below the table.
 - legend (optional): Dictionary mapping abbreviations to full names.
 - **kwargs: Additional arguments for `df.to_latex`.

 Returns:
 - None: Outputs LaTeX file.
 """

def format_p_value(x):
 returns "{:.3g}".format(x) if x >= 1e-06 else "<1e-06"

def is_str_in_df(df: pd.DataFrame, s: str):
 return any(s in level for level in getattr(df.index, 'levels', [df.index]) + getattr(df.columns, 'levels', [df.columns]))

AbbrToNameDef = Dict[Any, Tuple[Optional[str], Optional[str]]]

def split_mapping(abbrs_to_names_and_definitions: AbbrToNameDef):
 abbrs_to_names = {abbr: name for abbr, (name, definition) in abbrs_to_names_and_definitions.items() if name is not None}
 names_to_definitions = {name or abbr: definition for abbr, (name, definition) in abbrs_to_names_and_definitions.items() if definition is not None}
```

```
26  return abbrs_to_names, names_to_definitions
27
```

D.3 Code Output

table0.tex

```
\begin{table}[h]
\caption{Descriptive statistics of Height and Age stratified by Sex}
\label{table:descr_stats_height_age}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrrrr}
\toprule
 & \multicolumn{2}{r}{Height} & \multicolumn{2}{r}{Age} \\
 & mean & std & mean & std \\
\midrule
\textbf{Female} & 65.4 & 18.7 & 0.732 & 1.4 \\
\textbf{Male} & 66.5 & 19.4 & 0.781 & 1.47 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Height}: Height(cm)
\item \textbf{Age}: Age(years)
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table1.tex

```
\begin{table}[h]
\caption{Comparison of predictive power: ML Model vs. Formula-based Model}
\label{table:compare_ML_FBM}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrl}
\toprule
```

```
 & Mean Squared Residual & p-value \\
Model &  &  \\
\midrule
\textbf{ML Model} & 1.49 & $<$1e-06 \\
\textbf{Formula-based Model} & 3.81 & - \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{ML Model}: Machine Learning Model
\end{tablenotes}
\end{threeparttable}
\end{table}
```