Improving Tracheal Tube Placement in Pediatric Patients Undergoing Mechanical Ventilation

Data to Paper

January 8, 2024

Abstract

Pediatric patients undergoing mechanical ventilation face significant risks associated with misplaced tracheal tube tip positioning. Accurately determining the optimal tracheal tube depth (OTTD) is crucial for patient safety. However, existing methods, such as chest X-ray and formula-based models, have limitations in accurately determining OTTD. To address this challenge, we present a comprehensive study utilizing a dataset of pediatric patients aged 0-7 years who received post-operative mechanical ventilation. We employed a machine learning-based Random Forest model and a formula-based model to predict OTTD. Compared to the formula-based model, the Random Forest model demonstrated significantly superior predictive performance, resulting in a lower mean squared error. Our findings underscore the potential of machine learning in optimizing tracheal tube placement in pediatric patients undergoing mechanical ventilation. Furthermore, we identify sex-specific differences in patient characteristics that may impact tracheal tube positioning. It is important to note that our study has limitations, including a small sample size and potential biases in the dataset. Further research is needed to validate our results and establish precise guidelines for tracheal tube placement in pediatric patients. Nonetheless, our study highlights the importance of improving the accuracy of determining OTTD for improving patient safety in pediatric mechanical ventilation.

Results

We first sought to understand the physical characteristics of the pediatric patient cohort under study. Identifying any potential differences in height and age between sexes allowed us to discern patterns that might impact our subsequent analysis. As shown in Table 1, the mean age and height differed between male and female patients, providing insight into diverse characteristics within the dataset. This understanding served as a foundation for the main analysis.

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

	Height		Age	
	Mean	Standard Deviation	Mean	Standard Deviation
sex				
female	65.4	18.7	0.732	1.4
male	66.5	19.4	0.781	1.47

Height: Height in cm
Age: Age in years (rounded to half years)

The central question of our research was to determine the model that most accurately predicts the optimal tracheal tube depth (OTTD) in pediatric patients. We identified two potential models for this purpose: a machine learning based Random Forest model, which can account for complex interactions between variables, and a formula-based model representative of more traditional methods. The calculation of mean squared errors (MSE) for both models was then employed to objectively compare their predictive performance.

Our comparisons (Table 2) revealed that the Random Forest model significantly outperformed the formula-based model, with a considerably lower MSE of 1.43 versus 3.42, respectively. The lower MSE indicates the Random Forest model's superior estimation of OTTD, which is crucial to minimize risk associated with tracheal tube placement. This difference in model performance was statistically significant, as shown by a paired t-test (p-value $<10^{-6}$).

Table 2: Comparison of residuals for Random Forest and Formula-based Models

	Model	Mean Squared Error	p-value
Model 1	Random Forest	1.43	$<10^{-6}$
Model 2	Formula-based Model	3.42	$<10^{-6}$

Model 1: Random Forest Model
Model 2: Formula-based Model

In addition to MSE, we also evaluated the Random Forest model's overall predictive accuracy. Derived from unseen test data, an accuracy score of 0.584 affirms the model's capability in predicting OTTD effectively.

In summary, the Random Forest model's superior performance in predicting OTTD, as evidenced by both the lower MSE and higher overall accuracy, highlights the potential of machine learning based modelling for optimal tracheal tube placement. Such improvements in determining OTTD, in turn, present potential benefits to pediatric patient safety during mechanical ventilation.

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:

```python

import pandas as pd
import numpy as np

# Load the data
data = pd.read_csv('tracheal_tube_insertion.csv')

# Define output file
f = open("data_exploration.txt", "w")

# Measure of the scale of our data (e.g., number of rows,
↪  number of columns)
f.write("# Data Size\n")
f.write(f'Number of rows: {data.shape[0]}\n')
f.write(f'Number of columns: {data.shape[1]}\n')

# Summary statistics of all or key variables
f.write("\n# Summary Statistics\n")
summaryStats = data.describe().transpose()
summaryStats.to_string(f)

# Categorical Variables
f.write("\n# Categorical Variables\n")
sex_counts = data['sex'].value_counts()
```

```python
f.write(f'Sex counts:\n{sex_counts}\n')

# Missing Values
f.write("\n# Missing Values\n")
missing_values_count = data.isnull().sum()
f.write(f'Missing values count:\n{missing_values_count}\n')

# check for any special numeric values that stand for unknown
# if any based on dataset description
f.write("\n# Special Numeric Values\n")
special_values_count = (data < 0).sum()
f.write(f'Special values count:\n{special_values_count}\n')

f.close()

```

B.2 Code Description

The code performs data exploration on the provided dataset, "tracheal_tube_insertion.csv". This dataset contains information about pediatric patients who received post-operative mechanical ventilation.

First, the code loads the dataset using the pandas library. It then proceeds to analyze the data and generate summary statistics.

The code begins by providing the measurement of the scale of the data, including the number of rows and columns in the dataset.

Next, it calculates and writes the summary statistics for all or key variables in the dataset. This helps in understanding the distribution and range of the data, providing measures such as mean, standard deviation, minimum, maximum, and quartiles.

The code also explores categorical variables. In this case, it examines the 'sex' variable by counting the number of occurrences of each category (0=female, 1=male).

To identify any missing values in the dataset, the code checks for the presence of null values and writes the counts of missing values for each variable.

Furthermore, the code checks for any special numeric values that may indicate unknown or erroneous data based on the dataset description. It counts the occurrences of negative values in the dataset.

Lastly, the code writes all the exploratory findings into the "data_exploration.txt" file. This includes the data size, summary statistics, counts for categorical variables, missing values count, and special numeric values count. These findings provide a comprehensive understanding of the dataset's characteristics and any potential issues that may need to be addressed.

B.3 Code Output

data_exploration.txt

```
# Data Size
Number of rows: 969
Number of columns: 6

# Summary Statistics
              count   mean    std  min  25%  50%  75%   max
tube            969  3.694 0.5681  2.5  3.5  3.5    4     6
sex             969 0.5387 0.4988    0    0    1    1     1
age_c           969  0.758   1.44    0    0    0    1     7
ht              969     66  19.08 29.5   52 61.5   76 135.4
wt              969  7.127  4.774 0.57 3.56  5.7  9.5    31
tube_depth_G    969  10.19  1.766  5.9    9  9.8 11.2  19.2
# Categorical Variables
Sex counts:
sex
1    522
0    447
Name: count, dtype: int64

# Missing Values
Missing values count:
tube            0
sex             0
age_c           0
ht              0
wt              0
tube_depth_G    0
dtype: int64

# Special Numeric Values
Special values count:
tube            0
```

```
sex             0
age_c           0
ht              0
wt              0
tube_depth_G    0
dtype: int64
```

C Data Analysis

C.1 Code

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

```python


# IMPORT
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from scipy import stats
from sklearn.metrics import mean_squared_error
import pickle

# LOAD DATA
data = pd.read_csv("tracheal_tube_insertion.csv")

# DATASET PREPARATIONS
# no dataset preparations are needed

# DESCRIPTIVE STATISTICS
# Table 0: "Descriptive statistics of height and age
↪ stratified by sex"
grouped = data.groupby("sex").agg({"ht": ["mean", "std"],
↪ "age_c": ["mean", "std"]})
df0 = grouped.rename(index={0:'female', 1: 'male'})
df0.to_pickle('table_0.pkl')

# PREPROCESSING
# No preprocessing is needed, because all the data are in
↪  final form and we do not need to transform any categorical
↪  variables

# ANALYSIS
# Table 1: Comparison of residuals for two models {
↪  formula-based and Random Forest
y = data['tube_depth_G']

# Preparing X Data
X = data[['sex', 'age_c', 'ht', 'wt']]

# Preparing X_train and X_test
X_train, X_test, y_train, y_test = train_test_split(X, y,
↪  test_size=0.2, random_state=42)

# Random Forest model
rf = RandomForestRegressor(n_estimators=100, max_depth=7)
rf.fit(X_train, y_train)
rf_predicted = rf.predict(X_test)

# Compute residuals for the RF model
rf_residuals = np.square(y_test - rf_predicted)

# Formula-based model
formula_predicted = X_test['ht'] / 10 + 5
# Compute residuals for the formula-based model
formula_residuals = np.square(y_test - formula_predicted)

# Paired t-test of the residuals
paired_t_test = stats.ttest_rel(rf_residuals,
↪  formula_residuals)

# Create dataframe
df1 = pd.DataFrame({
 'Model': ['Random Forest', 'Formula-based Model'],
 'Mean of Squared Residuals': [rf_residuals.mean(),
 ↪  formula_residuals.mean()],
 'Paired t-test p value': [paired_t_test.pvalue,
 ↪  paired_t_test.pvalue]
}, index=['Model 1', 'Model 2'])

df1.to_pickle('table_1.pkl')


# SAVE ADDITIONAL RESULTS
additional_results = {
 'Total number of observations': len(data),
 'Accuracy of Random Forest Regression model':
  ↪  rf.score(X_test, y_test),
}

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

```

C.2 Code Description

The provided code performs an analysis on a dataset of pediatric patients who underwent post-operative mechanical ventilation after surgery. The goal of the analysis is to determine the optimal tracheal tube depth (OTTD) for these patients.

The code begins by loading the dataset of patient information, including features such as sex, age, height, weight, and the OTTD determined by chest X-ray. The data is stored in a pandas DataFrame.

Next, the code calculates descriptive statistics stratified by sex, including mean and standard deviation of height and age. This information is then saved as a pickle file called ”table_0.pkl”.

The code then prepares the data for analysis by separating the features (sex, age, height, and weight) into the input variable X, and the target variable (OTTD) into the output variable y. The data is split into training and testing sets for model evaluation.

The analysis is performed using two models: a Random Forest Regressor and a formula-based model. The Random Forest model is trained on the training data and used to predict the OTTD for the testing data. The residuals (squared differences between the predicted and actual OTTD) are computed.

The formula-based model predicts the OTTD using a simple formula based on height. The residuals for this model are also computed.

A paired t-test is then conducted to compare the residuals of the two models. The results of the t-test, including the mean squared residuals and the p-value, are stored in a DataFrame called "table_1.pkl".

Finally, additional results are saved in a pickle file called "additional_results.pkl". These include the total number of observations in the dataset and the accuracy of the Random Forest model on the testing data.

Overall, this code performs an analysis on pediatric patient data to determine the optimal tracheal tube depth. It compares the performance of a Random Forest model with a formula-based model and provides statistical insights into the accuracy of the models.

C.3 Code Output

table_0.pkl

```
                ht                  age_c
              mean        std        mean       std
sex
female   65.400447  18.701462  0.731544  1.402500
male     66.514368  19.403722  0.780651  1.472808
```

table_1.pkl

```
                       Model  Mean of Squared Residuals Paired t-test p value
Model 1        Random Forest                   1.433608             1.079e-08
Model 2  Formula-based Model                   3.418890             1.079e-08
```

additional_results.pkl

```
{
    'Total number of observations': 969,
    'Accuracy of Random Forest Regression model': 0.584            ,
}
```

D LaTeX Table Design

D.1 Code

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

```

# IMPORT
import pandas as pd
```

```python
from typing import Dict, Tuple, Optional, List, Any
from my_utils import to_latex_with_note, format_p_value,
↪  is_str_in_df, split_mapping, AbbrToNameDef

# PREPARATION FOR ALL TABLES
# Preparing a shared mapping for labels common to all tables
shared_mapping: AbbrToNameDef = {
    'ht': ('Height', 'Height in cm'),
    'age_c': ('Age', 'Age in years (rounded to half years)'),
    'sex': ('Sex', '0: Female, 1: Male'),
}

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

# RENAME ROWS AND COLUMNS
# Preparing mapping for Table 0
mapping0 = {k: v for k, v in shared_mapping.items() if
↪  is_str_in_df(df0, k)}
mapping0['mean'] = ('Mean', None)
mapping0['std'] = ('Standard Deviation', None)
abbrs_to_names, legend = split_mapping(mapping0)
df0 = df0.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Saving as latex
to_latex_with_note(
    df0, 'table_0.tex',
    caption="Descriptive statistics of Height and Age
    ↪  stratified by Sex",
    label='table:t0',
    note=None,
    legend=legend)


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

# FORMAT VALUES
# format p-value
```

```
df1['Paired t-test p value'] = df1['Paired t-test p
↪  value'].apply(format_p_value)

# RENAME ROWS AND COLUMNS
# Preparing mapping for Table1
mapping1 = {k: v for k, v in shared_mapping.items() if
↪  is_str_in_df(df1, k)}
mapping1['Model'] = (None, None)
mapping1['Mean of Squared Residuals'] = ('Mean Squared Error',
↪  None)
mapping1['Paired t-test p value'] = ('p-value', None)
mapping1['Model 1'] = (None, 'Random Forest Model')
mapping1['Model 2'] = (None, 'Formula-based Model')
abbrs_to_names, legend = split_mapping(mapping1)
df1 = df1.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Saving as latex
to_latex_with_note(
    df1, 'table_1.tex',
    caption="Comparison of residuals for Random Forest and
    ↪  Formula-based Models",
    label='table:t1',
    note=None,
    legend=legend)

```

D.2 Provided Code

The code above is using the following provided functions:

```
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}
 return abbrs_to_names, names_to_definitions

```

D.3 Code Output

table_0.tex

```
\begin{table}[h]
\caption{Descriptive statistics of Height and Age stratified by Sex}
\label{table:t0}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrrrr}
\toprule
 & \multicolumn{2}{r}{Height} & \multicolumn{2}{r}{Age} \\
 & Mean & Standard Deviation & Mean & Standard Deviation \\
sex &  &  &  &  \\
\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 in cm
\item \textbf{Age}: Age in years (rounded to half years)
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_1.tex

```
\begin{table}[h]
\caption{Comparison of residuals for Random Forest and Formula-based Models}
\label{table:t1}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{llrl}
\toprule
 & Model & Mean Squared Error & p-value \\
\midrule
\textbf{Model 1} & Random Forest & 1.43 & $<$1e-06 \\
\textbf{Model 2} & Formula-based Model & 3.42 & $<$1e-06 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Model 1}: Random Forest Model
\item \textbf{Model 2}: Formula-based Model
\end{tablenotes}
```

```
\end{threeparttable}
\end{table}
```