Machine learning models for accurate determination of tracheal tube depth in pediatric patients

Data to Paper

January 8, 2024

Abstract

Accurate determination of the optimal tracheal tube depth (OTTD) is crucial in pediatric patients undergoing mechanical ventilation to prevent serious complications. However, current methods for determining OTTD are either time-consuming or have limited success. This study addresses the need for a data-driven approach to determine OTTD in pediatric patients. We conducted a comprehensive analysis using a dataset of pediatric patients aged 0-7 years who received post-operative mechanical ventilation, with features extracted from their electronic health records. We applied random forest regression and elastic net regression models to predict the OTTD. Our results demonstrate the good performance of both models, with the elastic net regression model outperforming the random forest regression model in terms of accuracy. Importantly, these models utilize patient features such as age, sex, height, and weight as predictors, offering a faster and potentially more accurate alternative to the existing methods. However, validation of these models in a larger cohort is necessary before implementing them in clinical practice. Accurate determination of OTTD has the potential to greatly improve patient safety and outcomes in pediatric mechanical ventilation.

Results

We first conducted a descriptive analysis of the patient characteristics in our dataset, yielding key insights about the patient demographics and the variation in OTTD values (Table 1). Our cohort included 969 pediatric patients aged 0-7 years. The mean and standard deviation (SD) of patient age was 0.758 years and 1.44 years respectively, while the mean height was 66 cm (SD=19.1 cm) and the mean weight was 7.13 kg (SD=4.77 kg). As shown in Table 2, the mean OTTD, as determined by chest X-ray, was 10.2 cm (SD=1.77 cm).

Table 1: Descriptive Statistics of Age, Sex, Height, Weight, and OTTD

	Sex	Age	Height	Weight	OTTD
mean	0.539	0.758	66	7.13	10.2
std	0.499	1.44	19.1	4.77	1.77

Sex: Patient sex (0=female, 1=male)
Age: Patient age (years, rounded to half years)
Height: Patient height (cm)
Weight: Patient weight (kg)
OTTD: Optimal tracheal tube depth as determined by chest X-ray (in cm)

Table 2: Descriptive Statistics of Age, Sex, Height, Weight, and OTTD

	Sex	Age	Height	Weight	OTTD
mean	0.539	0.758	66	7.13	10.2
std	0.499	1.44	19.1	4.77	1.77

Sex: Patient sex (0=female, 1=male)
Age: Patient age (years, rounded to half years)
Height: Patient height (cm)
Weight: Patient weight (kg)
OTTD: Optimal tracheal tube depth as determined by chest X-ray (in cm)

Building on these patient features, we ventured to construct predictive models for OTTD using machine learning approaches. The performance of the random forest regression model is presented in Table 3, which shows a Mean Squared Error (MSE) of 1.95, demonstrating its predictive ability. However, the elastic net regression model outperformed the random forest regression model, achieving an MSE of 1.33 as detailed in Table 4.

Table 3: Random Forest Model Performance: Mean Square Error

	RF MSE
Random Forest	1.95

RF MSE: Random Forest model - Mean Square Error

In order to confirm the observed differences in the performance of the two models, we performed a paired-sample t-test on the squared prediction errors

Table 4: Elastic Net Model Performance: Mean Square Error

	EN MSE
Elastic Net	1.33

EN MSE: Elastic Net model - Mean Square Error

of both models. The t- statistic was found to be 3.752, with a corresponding p-value < 0.000212, thus establishing a statistically significant difference in the performance of the two models.

In summary, our results underscore the potential of machine learning models, particularly the elastic net model, in increasing the precision of OTTD prediction in pediatric patients receiving mechanical ventilation. Utilising features like sex, age, height, and weight, these models open up opportunities for safer and more effective OTTD determination in clinical practice.

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
df = pd.read_csv('tracheal_tube_insertion.csv')

# Open the summary file
with open('data_exploration.txt', 'w') as f:

    # Data size
    f.write('# Data Size\n')
    f.write(f'Number of rows: {df.shape[0]}\n')
    f.write(f'Number of columns: {df.shape[1]}\n\n')

    # Summary statistics
    f.write('# Summary Statistics\n')
    summary_statistics = df.describe(include=[np.number])
    summary_statistics.to_string(f)
    f.write('\n')

    # Categorical variables
    categorical_variables = df.select_dtypes(include=['bool',
     ↪  'object'])

    # If there is any categorical variable
    if not categorical_variables.empty:
        f.write('# Categorical Variables\n')
        for var in categorical_variables.columns:
            f.write(f'{var} value counts:\n')

             ↪  f.write(f'{categorical_variables[var].value_counts()}\n')
        f.write('\n')

    # If no categorical variable is available
    else:
        f.write('# Categorical Variables\n')
        f.write('There are no categorical variables in the
         ↪  data.\n\n')

    # Missing values
    f.write('# Missing Values\n')
    missing_values = df.isnull().sum()
    missing_values.to_string(f)
    f.write('\n')

```

B.2 Code Description

The provided code performs a basic data exploration of the dataset contained in the ”tracheal_tube_insertion.csv” file.

Firstly, it loads the dataset into a Pandas DataFrame.

Then, it writes a summary of the data size into the ”data_exploration.txt” file, including the number of rows and columns in the dataset.

Next, it calculates and writes summary statistics for the numerical variables in the dataset, such as count, mean, standard deviation, minimum, and maximum values. These statistics provide an overview of the data distribution and central tendency.

If the dataset contains any categorical variables, the code writes the value counts for each categorical variable into the ”data_exploration.txt” file. This provides information on the frequency of each category in the dataset.

Lastly, the code determines and writes the number of missing values for each variable in the dataset. This information is useful for identifying any data gaps or inconsistencies that may need to be addressed during analysis.

These exploratory analysis steps help researchers gain initial insights into the dataset and understand its general characteristics, such as its size, variable types, statistical properties, and missing data. These insights can guide further data cleaning, preprocessing, or feature selection steps in the research process.

B.3 Code Output

data_exploration.txt

```
# Data Size
Number of rows: 969
Number of 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
There are no categorical variables in the data.

# Missing Values
tube            0
sex             0
age_c           0
ht              0
wt              0
tube_depth_G    0
```

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.linear_model import ElasticNetCV
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import scipy.stats as stats
import pickle

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

# DATASET PREPARATIONS
# No dataset preparations are needed.

# DESCRIPTIVE STATISTICS
desc_stats = data[['sex', 'age_c', 'ht', 'wt', 'tube_depth_G']].describe().loc[['mean', 'std']]
desc_stats.to_pickle('table_0.pkl')

# PREPROCESSING
# No preprocessing is needed, because all the data features are already numeric.

# ANALYSIS
## Table 1: "Descriptive statistics of age, sex, height, weight, and OTTD"
df1 = desc_stats
df1.to_pickle('table_1.pkl')

# Model preparation
features = data[['sex', 'age_c', 'ht', 'wt']]
target = data['tube_depth_G']
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.3, random_state=0)

# RF model
rf_model = RandomForestRegressor(random_state=1)
```

```python
rf_model.fit(X_train, y_train)
rf_pred = rf_model.predict(X_test)

## Table 2: "RF model performance: MSE"
df2 = pd.DataFrame({'RF_MSE': [mean_squared_error(y_test, rf_pred)]}, index=['Random Forest'])
df2.to_pickle('table_2.pkl')

# EN model
en_model = ElasticNetCV(cv=5, l1_ratio=[.1, .5, .7, .9, .95, .99, 1])
en_model.fit(X_train, y_train)
en_pred = en_model.predict(X_test)

## Table 3: "EN model performance: MSE"
df3 = pd.DataFrame({'EN_MSE': [mean_squared_error(y_test, en_pred)]}, index=['Elastic Net'])
df3.to_pickle('table_3.pkl')

# Hypothesis test
t_test_result = stats.ttest_rel((rf_pred - y_test) ** 2, (en_pred - y_test) ** 2)

# SAVE ADDITIONAL RESULTS
additional_results = {
    'Total number of observations': len(data),
    't_stat for hypothesis test': t_test_result.statistic,
    'p_val for hypothesis test': t_test_result.pvalue
}
with open('additional_results.pkl', 'wb') as f:
    pickle.dump(additional_results, f)

```

C.2 Code Description

The code performs a data analysis to determine the optimal tracheal tube depth (OTTD) for pediatric patients who require mechanical ventilation. The analysis is based on a dataset containing patient information such as sex, age, height, weight, and the OTTD determined by chest X-ray.

The code starts by loading the dataset and computing descriptive statistics for the features and the target variable (OTTD). The statistics, including mean and standard deviation, are saved in a pickle file for further reference.

Next, the code prepares the data for analysis by splitting it into training and test sets. Two regression models, Random Forest (RF) and Elastic Net (EN), are then trained using the training set. The models are used to predict the OTTD for the test set.

The performance of each model is evaluated by calculating the mean squared error (MSE) between the predicted and actual values of OTTD. The RF and EN models' MSE values are saved in a pickle file for comparison.

To test the hypothesis that there is a difference in performance between the RF and EN models, a t-test is performed on the squared error values for the two models. The t-statistic and p-value of the test are computed and saved in the "additional_results.pkl" file.

Overall, the code conducts a data analysis to determine the optimal tracheal tube depth for pediatric patients using regression models, evaluates the performance of the models, and performs a hypothesis test to compare the models' performance. The results are saved in pickle files for further analysis and reference.

C.3 Code Output

table_0.pkl

```
           sex     age_c         ht        wt  tube_depth_G
mean  0.538700  0.757998  66.000516  7.126687     10.189474
std   0.498758  1.440271  19.081267  4.774186      1.766052
```

table_1.pkl

```
           sex     age_c         ht        wt  tube_depth_G
mean  0.538700  0.757998  66.000516  7.126687     10.189474
std   0.498758  1.440271  19.081267  4.774186      1.766052
```

table_2.pkl

```
                RF_MSE
Random Forest  1.95202
```

table_3.pkl

```
                EN_MSE
Elastic Net   1.329487
```

additional_results.pkl

```
{
    'Total number of observations': 969,
    't_stat for hypothesis test': 3.752                ,
    'p_val for hypothesis test': 0.000212,
}
```

D LaTeX Table Design

D.1 Code

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

```python

# IMPORT
import pandas as pd
from typing import Dict, Any, Optional, Tuple

# import
from my_utils import to_latex_with_note, format_p_value, is_str_in_df, split_mapping, AbbrToNameDef

# PREPARATION FOR ALL TABLES
AbbrToNameDef = Dict[Any, Tuple[Optional[str], Optional[str]]]

shared_mapping: AbbrToNameDef = {
 'sex': ('Sex', 'Patient sex (0=female, 1=male)'),
 'age_c': ('Age', 'Patient age (years, rounded to half years)'),
 'ht': ('Height', 'Patient height (cm)'),
 'wt': ('Weight', 'Patient weight (kg)'),
 'tube_depth_G': ('OTTD', 'Optimal tracheal tube depth as determined by chest X-ray (in cm)'),
 'RF_MSE': ('RF MSE', 'Random Forest model - Mean Square Error'),
 'EN_MSE': ('EN MSE', 'Elastic Net model - Mean Square Error')
}

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

# RENAME ROWS AND COLUMNS
mapping = {k: v for k, v in shared_mapping.items() if is_str_in_df(df, k)}
abbrs_to_names, legend = split_mapping(mapping)
df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(
 df, 'table_0.tex',
 caption="Descriptive Statistics of Age, Sex, Height, Weight, and OTTD",
 label='table:stat_desc_0',
 note="",
 legend=legend)

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

# RENAME ROWS AND COLUMNS
mapping = {k: v for k, v in shared_mapping.items() if is_str_in_df(df, k)}
abbrs_to_names, legend = split_mapping(mapping)
df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(
 df, 'table_1.tex',
 caption="Descriptive Statistics of Age, Sex, Height, Weight, and OTTD",
 label='table:stat_desc_1',
 note="",
 legend=legend)

# TABLE 2:
```

```
df = pd.read_pickle('table_2.pkl')

# RENAME ROWS AND COLUMNS
mapping = {k: v for k, v in shared_mapping.items() if
↪  is_str_in_df(df, k)}
abbrs_to_names, legend = split_mapping(mapping)
df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(
 df, 'table_2.tex',
 caption="Random Forest Model Performance: Mean Square Error",
 label='table:rf_model_performance',
 note="",
 legend=legend)

# TABLE 3:
df = pd.read_pickle('table_3.pkl')

# RENAME ROWS AND COLUMNS
mapping = {k: v for k, v in shared_mapping.items() if
↪  is_str_in_df(df, k)}
abbrs_to_names, legend = split_mapping(mapping)
df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(
 df, 'table_3.tex',
 caption="Elastic Net Model Performance: Mean Square Error",
 label='table:en_model_performance',
 note="",
 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}
 return abbrs_to_names, names_to_definitions

```

D.3 Code Output

table_0.tex

```
\begin{table}[h]
\caption{Descriptive Statistics of Age, Sex, Height, Weight, and OTTD}
\label{table:stat_desc_0}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrrrrr}
\toprule
 & Sex & Age & Height & Weight & OTTD \\
\midrule
\textbf{mean} & 0.539 & 0.758 & 66 & 7.13 & 10.2 \\
\textbf{std} & 0.499 & 1.44 & 19.1 & 4.77 & 1.77 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Sex}: Patient sex (0=female, 1=male)
\item \textbf{Age}: Patient age (years, rounded to half years)
\item \textbf{Height}: Patient height (cm)
\item \textbf{Weight}: Patient weight (kg)
\item \textbf{OTTD}: Optimal tracheal tube depth as determined by chest X-ray
    (in cm)
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_1.tex

```
\begin{table}[h]
\caption{Descriptive Statistics of Age, Sex, Height, Weight, and OTTD}
\label{table:stat_desc_1}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrrrrr}
\toprule
```

```
 & Sex & Age & Height & Weight & OTTD \\
\midrule
\textbf{mean} & 0.539 & 0.758 & 66 & 7.13 & 10.2 \\
\textbf{std} & 0.499 & 1.44 & 19.1 & 4.77 & 1.77 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Sex}: Patient sex (0=female, 1=male)
\item \textbf{Age}: Patient age (years, rounded to half years)
\item \textbf{Height}: Patient height (cm)
\item \textbf{Weight}: Patient weight (kg)
\item \textbf{OTTD}: Optimal tracheal tube depth as determined by chest X-ray
    (in cm)
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_2.tex

```
\begin{table}[h]
\caption{Random Forest Model Performance: Mean Square Error}
\label{table:rf_model_performance}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lr}
\toprule
 & RF MSE \\
\midrule
\textbf{Random Forest} & 1.95 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{RF MSE}: Random Forest model - Mean Square Error
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_3.tex

```
\begin{table}[h]
\caption{Elastic Net Model Performance: Mean Square Error}
\label{table:en_model_performance}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lr}
\toprule
 & EN MSE \\
\midrule
\textbf{Elastic Net} & 1.33 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{EN MSE}: Elastic Net model - Mean Square Error
\end{tablenotes}
\end{threeparttable}
\end{table}
```