Improving Prediction of Optimal Tracheal Tube Depth in Pediatric Patients Undergoing Mechanical Ventilation

Data to Paper

January 7, 2024

Abstract

Tracheal tube misplacement is a significant concern in pediatric patients undergoing mechanical ventilation, necessitating accurate determination of the optimal tracheal tube depth (OTTD) to prevent complications. Traditional methods, such as chest X-rays, are time-consuming and entail radiation exposure. Here, we aimed to develop and compare predictive models for OTTD using non-invasive approaches. We analyzed a dataset of 969 pediatric patients who underwent post-operative mechanical ventilation. Our Random Forest model, incorporating patient characteristics such as sex, age, height, and weight, demonstrated superior precision over the formula-based model commonly used in clinical practice. The Random Forest model exhibited a lower Root Mean Square Error (RMSE) of 1.21 compared to 1.85 for the formula-based model. By accurately estimating OTTD, these non-invasive predictive models offer potential for reducing reliance on chest X-rays and associated risks. However, further refinement and validation are necessary to ensure clinical applicability and safety. Implementation of these models in clinical practice may enhance outcomes in pediatric patients undergoing mechanical ventilation.

Results

Our analysis was based on a dataset of 969 clinical observations, each representing a pediatric patient who underwent mechanical ventilation post-operatively at Samsung Medical Center. A Random Forest model was developed to predict the optimal tracheal tube depth (OTTD), determined via chest X-rays, using non-invasive parameters including patient's sex, age, height, and weight.

To understand the capabilities of our Random Forest model, we first compared its residuals with those of a common formula-based model, instinctively appealed to clinicians, and thereby quantified its relative predictive accuracy. Indeed, a paired sample t-test was performed on the squared residuals of the two models to identify any significant difference in their predictive performance. Our analysis indicated that the Random Forest model significantly outperformed the formula-based model in predicting OTTD (Table 1). The p-value was lower than the commonly accepted threshold ($p < 10^{-6}$), implying strong evidence against the null hypothesis, i.e., no difference in the performance of the two models.

Table 1: Test of difference in predictive power between formula-based model

	Model	t-statistic	p-value
Model 1	Random Forest	-6.02	$<10^{-6}$
Model 2	Formula-based Model	-6.02	$<10^{-6}$

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

Subsequently, the Root Mean Square Error (RMSE), a popular measure of the concentration of data, was calculated to gain insights into the predictive power of the models in absolute terms. Remarkably, the Random Forest model exhibited a lower RMSE of 1.21 as compared to the formula-based model which exhibited a noticeably higher RMSE value of 1.85 (Table 2). This further underscored the superiority of the Random Forest model in accurately predicting the OTTD.

Table 2: Root mean square error of formula-based model and the Random Forest model

	Model	RMSE
Model 1	Random Forest	1.21
Model 2	Formula-based Model	1.85

Model 1: Random Forest Model
Model 2: Height Formula-based Model
RMSE: Root Mean Square Error of the model predictions

Finally, the optimal parameters for the Random Forest model were obtained using the concept of Grid Search. More specifically, a total of 200 estimators were combined in the final model while the minimum samples

split was determined to be 8. This showed that the model was non-trivial and it learned from a substantial amount of features and residuals to make accurate predictions.

Taking these analyses together, the Random Forest model demonstrated improved precision over the traditional formula-based model in predicting the optimal tracheal tube depth. An RMSE of 1.21, significantly lower than that of the formula-based model, further emphasized this superiority, underlining its potential role in addressing the criticality of accurately determining OTTD in pediatric patients undergoing 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:

```
1
2  import pandas as pd
3  import numpy as np
4
5  # load the dataset
6  df = pd.read_csv('tracheal_tube_insertion.csv')
7
8  # Open file for writing the exploration results
9  with open("data_exploration.txt", "w") as f:
10
11     # Check the size of our data
12     f.write("# Data Size\n")
13     f.write(str(df.shape) + "\n\n")
14
15     # Summary statistics of all or key variables
16     f.write("# Summary Statistics\n")
17     f.write(str(df.describe()) + "\n\n")
18
19     # Categorical Variables
20     f.write("# Categorical Variables\n")
21     f.write(str(df['sex'].value_counts()) + "\n\n")
22
23     # missing and special values
24     f.write("# Missing Values\n")
```

```
    f.write(str(df.isnull().sum()) + "\n\n")

```

B.2 Code Description

The provided code performs data exploration on the given dataset of pediatric patients who received post-operative mechanical ventilation after undergoing surgery. The goal of data exploration is to understand the structure and characteristics of the dataset.

First, the code loads the dataset using the 'read_csv' function from the pandas library.

The code then proceeds to analyze the dataset and write the results into a file named "data_exploration.txt".

The code starts by determining the size of the dataset using the 'shape' attribute of the pandas DataFrame. The result, which represents the number of rows and columns in the dataset, is written into the file.

Next, the code computes and writes summary statistics of the dataset using the 'describe' function. This provides information such as count, mean, standard deviation, minimum, and maximum values for each numerical variable in the dataset.

The code then focuses on the categorical variable 'sex' and computes the frequency count of different categories using the 'value_counts' function. The result, which shows the number of male and female patients, is written into the file.

Finally, the code checks for missing values in the dataset using the 'isnull' function. It computes the sum of missing values for each variable and writes the result into the file. This step helps identify any data points that are incomplete or have missing information.

Overall, the code provides a comprehensive exploration of the dataset by analyzing its size, summary statistics, categorical variables, and missing values. The results are written into the "data_exploration.txt" file for further examination and interpretation.

B.3 Code Output

data_exploration.txt

```
# Data Size
(969, 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
sex
1    522
0    447
Name: count, dtype: int64

# Missing Values
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:

```
1
2
3 # IMPORT
4 import pandas as pd
5 from sklearn.model_selection import train_test_split
6 from sklearn.ensemble import RandomForestRegressor
7 from sklearn.metrics import mean_squared_error
```

```python
from scipy.stats import ttest_rel
import numpy as np
from sklearn.model_selection import GridSearchCV
import pickle

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

# DATASET PREPARATIONS
# No dataset preparations are needed.

# DESCRIPTIVE STATISTICS
# No descriptive statistics table is needed.

# PREPROCESSING
# No preprocessing is needed, because the dataset is ready to be used by the machine learning model.

# ANALYSIS

## Table 1: Test of difference in predictive power between formula-based model

# Separate features and target
features = data[['sex', 'age_c', 'ht', 'wt']].values
target = data['tube_depth_G'].values

# Train test split
features_train, features_test, target_train, target_test = train_test_split(
    features, target, test_size=0.2, random_state=42)

rf = RandomForestRegressor(random_state=42)
param_grid = {
    'n_estimators': [50, 100, 200],
    'min_samples_split': [2, 4, 8],
}
grid_search = GridSearchCV(estimator = rf, param_grid = param_grid)

```

```
grid_search.fit(features_train, target_train)
best_rf = grid_search.best_estimator_

# Prediction
rf_predictions = best_rf.predict(features_test)

# Compute residuals for RF model
rf_squared_residuals = (rf_predictions - target_test) ** 2

# Compute Formula-based model predictions for test set
formula_predictions_test = features_test[:, 2] / 10 + 5

# Compute residuals for Formula-based model
formula_squared_residuals = (formula_predictions_test -
↪  target_test) ** 2

# Perform paired t-test
ttest_res = ttest_rel(rf_squared_residuals,
↪  formula_squared_residuals)

table_1 = pd.DataFrame({
    'Model': ['Random Forest', 'Formula-based Model'],
    't-statistic': [ttest_res.statistic, ttest_res.statistic],
    'p-value': [ttest_res.pvalue, ttest_res.pvalue]
}, index=['Model 1', 'Model 2'])

# SAVE DATAFRAME
table_1.to_pickle('table_1.pkl')

## Table 2: RMSE of Formula-based model and the Random Forest
↪  model

rf_rmse = np.sqrt(mean_squared_error(target_test,
↪  rf_predictions))
formula_rmse = np.sqrt(mean_squared_error(target_test,
↪  formula_predictions_test))

table_2 = pd.DataFrame({
    'Model': ['Random Forest', 'Formula-based Model'],
    'RMSE': [rf_rmse, formula_rmse]
```

```
}, index=['Model 1', 'Model 2'])

table_2.to_pickle('table_2.pkl')

# SAVE ADDITIONAL RESULTS
additional_results = {
    'Total number of observations': len(data),
    'Best Random Forest parameters': grid_search.best_params_
}

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

```

C.2 Code Description

The provided code performs the analysis of the dataset "tracheal_tube_insertion.csv" to compare the predictive power and accuracy of a Random Forest model with a formula-based model for determining the optimal tracheal tube depth ('OTTD') in pediatric patients undergoing post-operative mechanical ventilation. The analysis involves the following steps:

1. Data loading: The dataset is read into a pandas DataFrame.

2. Dataset preparations: No specific preparations are needed as the dataset is already cleaned and ready for analysis.

3. Analysis of predictive power: - The features (patient sex, age, height, and weight) and the target variable (optimal tracheal tube depth determined by chest X-ray) are extracted from the dataset. - The dataset is split into training and test sets using an 80-20 ratio. - Random Forest model: - A Random Forest Regressor is trained using the GridSearchCV method to find the best hyperparameters (number of estimators and minimum number of samples to split). - The best-performing Random Forest model is used to make predictions on the test set. - The squared residuals (squared difference between predicted and actual values) are computed for the Random Forest model. - Formula-based model: - Formula-based predictions of tracheal tube depth are calculated using the patient's height. - The squared residuals are computed for the formula-based model. - Paired t-test: - A paired t-test is performed to compare the predictive power of the Random Forest model and the formula-based model. - The t-statistic and p-value are calculated for the t-test. - Table 1: - A table is created with the models' names, t-statistic, and p-value. - The table is saved as a pickle object.

4. Analysis of RMSE: - The root mean squared error (RMSE) for both the Random Forest model and the formula-based model is calculated using the test set. - Table 2: - A table is created with the models' names and corresponding RMSE values. - The table is saved as a pickle object.

5. Additional results: - The total number of observations in the dataset and the best hyperparameters found for the Random Forest model are stored as a dictionary. - The dictionary is saved as a pickle object.

The "additional_results.pkl" file contains a dictionary with the following information: - "Total number of observations": The total number of patients in the dataset. - "Best Random Forest parameters": The best hyperparameters (number of estimators and minimum number of samples to split) found for the Random Forest model during the GridSearchCV process.

These results provide insights into the predictive power and accuracy of the Random Forest model compared to the formula-based model for determining the optimal tracheal tube depth in pediatric patients undergoing post-operative mechanical ventilation.

C.3 Code Output

table_1.pkl

	Model	t-statistic	p-value
Model 1	Random Forest	-6.023673	8.47e-09
Model 2	Formula-based Model	-6.023673	8.47e-09

table_2.pkl

	Model	RMSE
Model 1	Random Forest	1.211588
Model 2	Formula-based Model	1.849024

additional_results.pkl

```
{
    'Total number of observations': 969,
    'Best Random Forest parameters': {'min_samples_split': 8, 'n_estimators':
    200},
}
```

D LaTeX Table Design

D.1 Code

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

```

# IMPORT
import pandas as pd
from my_utils import to_latex_with_note, format_p_value, is_str_in_df, split_mapping, AbbrToNameDef

# PREPARATION FOR ALL TABLES
shared_mapping: AbbrToNameDef = {
    'tube': ('Tube ID', 'Internal diameter of the tube in mm'),
    'sex': ('Sex', 'Patient sex; 0: Female, 1: Male'),
    'age_c': ('Age (years)', 'Patient age in years, rounded to half years'),
    'ht': ('Height (cm)', 'Patient height in cm'),
    'wt': ('Weight (kg)', 'Patient weight in kg'),
    'tube_depth_G': ('Optimal Tracheal Tube Depth (cm)', 'Optimal tracheal tube depth as determined by chest X-ray in cm'),
    'Model 1': ('Model 1', 'Random Forest Model'),
    'Model 2': ('Model 2', 'Height Formula-based Model'),
    'RMSE': ('RMSE', 'Root Mean Square Error of the model predictions')
}

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

# FORMAT VALUES
df['p-value'] = df['p-value'].apply(format_p_value)

# 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)
```

```python

# Save as latex:
to_latex_with_note(
    df, 'table_1.tex',
    caption="Test of difference in predictive power between formula-based model",
    label='table:prediction_models',
    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="Root mean square error of formula-based model and the Random Forest model",
    label='table:rmse_models',
    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_1.tex

```
\begin{table}[h]
\caption{Test of difference in predictive power between formula-based model}
```

```
\label{table:prediction_models}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{llrl}
\toprule
 & Model & t-statistic & p-value \\
\midrule
\textbf{Model 1} & Random Forest & -6.02 & $<$1e-06 \\
\textbf{Model 2} & Formula-based Model & -6.02 & $<$1e-06 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Model 1}: Random Forest Model
\item \textbf{Model 2}: Height Formula-based Model
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_2.tex

```
\begin{table}[h]
\caption{Root mean square error of formula-based model and the Random Forest
    model}
\label{table:rmse_models}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{llr}
\toprule
 & Model & RMSE \\
\midrule
\textbf{Model 1} & Random Forest & 1.21 \\
\textbf{Model 2} & Formula-based Model & 1.85 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
```

```
\item \textbf{Model 1}: Random Forest Model
\item \textbf{Model 2}: Height Formula-based Model
\item \textbf{RMSE}: Root Mean Square Error of the model predictions
\end{tablenotes}
\end{threeparttable}
\end{table}
```