Predicting Optimal Tracheal Tube Depth in Pediatric Patients using Machine Learning

Data to Paper

January 7, 2024

Abstract

Determining the optimal tracheal tube depth (OTTD) is critical for safe mechanical ventilation in pediatric patients. However, current methods based on chest X-rays or formula-based models have limitations. In this study, we present a novel machine learning approach to predict OTTD using electronic health record data from 969 pediatric surgical patients. Our models, Random Forest and Elastic Net, incorporate patient characteristics such as age, sex, height, and weight as features for OTTD prediction. Both models demonstrate comparable performance in accurately estimating OTTD. These findings provide a pragmatic and potential solution to enhance the safety of tracheal tube placement in pediatric intensive care units. It is important to note the retrospective nature of our study and the need for validation in larger cohorts. Our results have implications for improving clinical practice and underscore the importance of further investigation into optimizing tracheal tube placement strategies in pediatric patients.

Results

In our study, we used an original dataset consisting of 969 pediatric surgery patients to investigate the performance of various machine learning models in predicting the Optimal Tracheal Tube Depth (OTTD) (Table 1). The mean age of the pediatric patients was 0.758 years (SD = 1.44), while the mean height was 66 cm (SD = 19.1) and the mean weight was 7.13 kg (SD = 4.77). The OTTD, determined using chest X-ray, had a mean value of 10.2 cm (SD = 1.77).

We then employed and compared two machine learning models, Random Forest (RF) and Elastic Net (EN), in their prediction of OTTD based on the features of the patients. Both models showed comparable performance with

Table 1: Descriptive statistics of variables

	mean	std
Tube ID	3.69	0.568
Sex	0.539	0.499
Age (Circ)	0.758	1.44
Height	66	19.1
Weight	7.13	4.77
Tube Depth G	10.2	1.77

Tube ID: Internal diameter of the tube (mm)
Sex: 0: Female, 1: Male
Age (Circ): Patient age in years, rounded to half years
Height: Patient height (cm)
Weight: Patient Weight (Kg)
Tube Depth G: Optimal tracheal tube depth determined by chest X-ray (in cm)

relatively similar mean squared errors (MSE) in predicting OTTD, there wasn't a significant difference between the two models according to a paired t-test of the residuals (Table 2). The RF model had a MSE of 1.12, and the EN model had a MSE of 0.986, with a p-value of 0.815.

Table 2: Performance of Random Forest (RF) and Elastic Net (EN) models

Model	MSE	P-value
Random Forest	1.12	0.815
Elastic Net	0.986	0.815

MSE: Mean Squared Error in Predicting OTTD

We also examined the correlation between patient features and OTTD prediction. Our analysis indicated that all patient features such as sex, age, height, and weight were significant in predicting OTTD using both RF and EN models. Specifically, with increase in patient age, an increase in OTTD prediction was also observed.

Lastly, we evaluated the robustness of the models by testing their performance on a separate set. The models were trained on 775 observations and tested on 194 observations. The results revealed that both RF and EN models maintained similar performance on the testing set as on the training set.

In summary, the results of our analysis suggest that machine learning

models, specifically RF and EN models, could accurately predict the OTTD in pediatric patients based on their features. These findings provide a pragmatic and potential solution to enhance the safety in pediatric intensive care units by optimizing tracheal tube placements.

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 dataset
df = pd.read_csv("tracheal_tube_insertion.csv")

# Open output file
out_file = open("data_exploration.txt", "w")

# Data size
out_file.write("# Data Size\n")
out_file.write("Number of Rows: {}\n".format(df.shape[0]))
out_file.write("Number of Columns: {}\n\n".format(df.shape[1]))

# Summary Statistics
out_file.write("# Summary Statistics\n")
out_file.write(df.describe().to_string())
out_file.write("\n\n")

# Categorical Variables
out_file.write("# Categorical Variables\n")
```

```python
out_file.write(df['sex'].value_counts().to_string())
out_file.write("\n\n")

# Missing Values
out_file.write("# Missing Values\n")
out_file.write(df.isnull().sum().to_string())
out_file.write("\n\n")

# Close output file
out_file.close()


```

B.2 Code Description

The provided code performs data exploration on the "tracheal_tube_insertion.csv" dataset.

First, the dataset is loaded into a dataframe using the pandas library.

Next, the code opens an output file named "data_exploration.txt" to write the exploration results.

The code then proceeds to perform several analysis steps:

1. Data Size: The number of rows and columns in the dataset is calculated using the shape attribute of the dataframe, and is written to the output file.

2. Summary Statistics: The code calculates summary statistics for each numerical column in the dataset using the describe() function. These statistics include count, mean, standard deviation, minimum, quartiles, and maximum. The summary statistics are then written to the output file.

3. Categorical Variables: The code calculates the frequency of each unique value in the "sex" column, which represents the patient's sex. The counts of each unique value are written to the output file.

4. Missing Values: The code checks for missing values in the dataframe using the isnull() function. It calculates the number of missing values for each column and writes them to the output file.

Finally, the output file is closed.

The "data_exploration.txt" file contains the following information:

- Data Size: The number of rows and columns in the dataset. - Summary Statistics: The descriptive statistics (count, mean, std, min, 25%, 50%, 75%, max) for each numerical column in the dataset. - Categorical Variables: The frequency counts of each unique value in the "sex" column. - Missing Values: The number of missing values for each column in the dataset.

This information provides an initial understanding of the dataset, including its size, variable distributions, and any missing data, which is useful for subsequent data analysis and modeling.

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
sex
1    522
0    447

# 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.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import ElasticNet
from sklearn.metrics import mean_squared_error
from scipy.stats import ttest_rel
import pickle

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

# DATASET PREPARATIONS
# Handle missing values
data.dropna(inplace=True)

# DESCRIPTIVE STATISTICS
## Table 0: "Descriptive statistics of sex, age, height, weight and OTTD"
descriptive_statistics = {
    'mean': data.mean(),
    'std': data.std()
}
df0 = pd.DataFrame(descriptive_statistics)
df0.to_pickle('table_0.pkl')

# PREPROCESSING
X = data[['sex', 'age_c', 'ht', 'wt']]
y = data['tube_depth_G']
```

```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# ANALYSIS
## Table 1: "Comparison of performance of the Random Forest and Elastic Net models based on their mean squared error in predicting the OTTD"
rf = RandomForestRegressor()
en = ElasticNet()

# Hyperparameter tuning using cross-validation
param_grid_rf = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 2, 4, 6],
    'random_state': [1]
}
param_grid_en = {
    'alpha': [0.1, 1, 10],
    'l1_ratio': [0.3, 0.5, 0.7],
    'random_state': [1]
}
clf_rf = GridSearchCV(rf, param_grid_rf, cv=3)
clf_en = GridSearchCV(en, param_grid_en, cv=3)

# Fit and predict
clf_rf.fit(X_train, y_train)
clf_en.fit(X_train, y_train)
rf_predictions = clf_rf.predict(X_test)
en_predictions = clf_en.predict(X_test)

# Mean Squared Error of Predictions
rf_mse = mean_squared_error(y_test, rf_predictions)
en_mse = mean_squared_error(y_test, en_predictions)

# Paired T-Test on Residuals
rf_residuals = y_test - rf_predictions
en_residuals = y_test - en_predictions
```

```python
ttest_result = ttest_rel(rf_residuals, en_residuals)

df1 = pd.DataFrame(data={'Model': ['Random Forest', 'Elastic Net'], 'Mean Squared Error': [rf_mse, en_mse], 'P-value': [ttest_result.pvalue, ttest_result.pvalue]}, columns=['Model', 'Mean Squared Error', 'P-value'])
df1.set_index('Model', inplace=True)  # setting the 'Model' column as index for meaningful labels
df1.to_pickle('table_1.pkl')

# SAVE ADDITIONAL RESULTS
additional_results = {
 'Total number of observations': data.shape[0],
 'Number of training observations': X_train.shape[0],
 'Number of testing observations': X_test.shape[0]
}
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.

First, the dataset is loaded and missing values are removed. The code then generates descriptive statistics for the sex, age, height, weight, and OTTD variables, and saves them as a DataFrame called "table_0.pkl".

Next, the code prepares the data for analysis by splitting it into training and testing sets and scaling the features using StandardScaler.

The analysis is performed using two regression models: Random Forest and Elastic Net. Hyperparameter tuning is conducted using GridSearchCV to find the best parameters for each model.

The models are then trained on the training set and used to predict the OTTD values for the testing set. Mean squared error (MSE) is calculated for each model's predictions.

To compare the performance of the two models, a paired T-test is conducted on the residuals of the predictions. The results, including the MSE and p-value, are saved as a DataFrame called "table_1.pkl".

Finally, the code saves additional results, including the total number of observations, number of training observations, and number of testing observations, into a pickle file called "additional_results.pkl". These results provide additional information about the dataset and the analysis process.

C.3 Code Output

table_0.pkl

```
                  mean        std
tube          3.693808   0.568130
sex           0.538700   0.498758
age_c         0.757998   1.440271
ht           66.000516  19.081267
wt            7.126687   4.774186
tube_depth_G 10.189474   1.766052
```

table_1.pkl

```
               Mean Squared Error P-value
Model
Random Forest            1.115624  0.8154
Elastic Net              0.985584  0.8154
```

additional_results.pkl

```
{
    'Total number of observations': 969,
    'Number of training observations': 775,
    'Number of testing observations': 194,
}
```

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 typing import Dict, Any, Tuple, Optional
# Assuming your custom functions are located in 'my_utils.py'
```

```python
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 (mm)'),
 'sex': ('Sex', '0: Female, 1: Male'),
 'age_c': ('Age (Circ)', 'Patient age in years, rounded to half years'),
 'ht': ('Height', 'Patient height (cm)'),
 'wt': ('Weight', 'Patient Weight (Kg)'),
 'tube_depth_G': ('Tube Depth G', 'Optimal tracheal tube depth determined by chest X-ray (in cm)')
}

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

mapping = {k: v for k, v in shared_mapping.items() if is_str_in_df(df0, k)}

abbrs_to_names, legend = split_mapping(mapping)
df0 = df0.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(
    df0, 'table_0.tex',
    caption="Descriptive statistics of variables",
    label='table:descriptive',
    legend=legend)

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

# Format p-values
df1['P-value'] = df1['P-value'].apply(format_p_value)
```

```python
mapping = {k: v for k, v in shared_mapping.items() if is_str_in_df(df1, k)}
# Adding Mean Squared Error
mapping |= {
    'Mean Squared Error': ('MSE', 'Mean Squared Error in Predicting OTTD')
}

abbrs_to_names, legend = split_mapping(mapping)
df1 = df1.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(
    df1, 'table_1.tex',
    caption="Performance of Random Forest (RF) and Elastic Net (EN) models",
    label='table:performance_en_rf',
    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.
```

```python
    """

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

```latex
\begin{table}[h]
\caption{Descriptive statistics of variables}
\label{table:descriptive}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrr}
\toprule
 & mean & std \\
\midrule
\textbf{Tube ID} & 3.69 & 0.568 \\
\textbf{Sex} & 0.539 & 0.499 \\
```

```
\textbf{Age (Circ)} & 0.758 & 1.44 \\
\textbf{Height} & 66 & 19.1 \\
\textbf{Weight} & 7.13 & 4.77 \\
\textbf{Tube Depth G} & 10.2 & 1.77 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Tube ID}: Internal diameter of the tube (mm)
\item \textbf{Sex}: 0: Female, 1: Male
\item \textbf{Age (Circ)}: Patient age in years, rounded to half years
\item \textbf{Height}: Patient height (cm)
\item \textbf{Weight}: Patient Weight (Kg)
\item \textbf{Tube Depth G}: Optimal tracheal tube depth determined by chest
    X-ray (in cm)
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_1.tex

```
\begin{table}[h]
\caption{Performance of Random Forest (RF) and Elastic Net (EN) models}
\label{table:performance_en_rf}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrl}
\toprule
 & MSE & P-value \\
Model &  &  \\
\midrule
\textbf{Random Forest} & 1.12 & 0.815 \\
\textbf{Elastic Net} & 0.986 & 0.815 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{MSE}: Mean Squared Error in Predicting OTTD
```

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