Improved Determination of Optimal Tracheal Tube Depth in Pediatric Patients Using Machine Learning Models

Data to Paper

January 8, 2024

Abstract

Accurate determination of the optimal tracheal tube depth (OTTD) is critical for safe mechanical ventilation in pediatric patients. However, existing methods have limitations, leading to misplaced tube tips and potential complications. In this study, we analyze a dataset of pediatric post-operative mechanical ventilation cases, incorporating patient features from electronic health records and OTTD determined by chest X-ray. We compare the performance of a random forest model and a formula-based model in predicting OTTD. Our results demonstrate that the random forest model, leveraging patient sex, age, height, and weight, outperforms the formula-based model, significantly improving the precision of OTTD determination. These findings highlight the importance of accurately determining OTTD in pediatric patients, which can reduce complications and enhance patient safety during mechanical ventilation. However, the study has limitations due to the relatively small dataset, necessitating further validation on larger cohorts to enhance the generalizability of our findings. Additionally, future research can explore additional patient characteristics and refine the predictive model to enhance the accuracy of OTTD determination in pediatric patients.

Results

Optimal Tracheal Tube Depth (OTTD) is a critical parameter to ensure safe mechanical ventilation in pediatric patients. With pediatric patients having a shorter tracheal length than adults, inaccurate OTTD determination can lead to misplacement of tracheal tube tips and cause grave complications– such as hypoxia, atelectasis, hypercarbia, and pneumothorax–endangering

the safety of post-operative care in pediatric patients. Therefore, the need for accurate predictive models for OTTD determination is unequivocal.

In our quest for preferential models, we compared the performance of a random forest model with a traditional formula-based model, finding compelling evidence towards the superior accuracy of the random forest model (Table 1). The formula-based model, which primarily uses age and height as determining features, showed a Mean Squared Error (MSE) of 3.42. On the other hand, the random forest model, which integrates a wider set of features, including patient sex, age, height, and weight, minimized MSE to 1.55. A paired t-test provided further evidence of this superiority, indicating the difference in performance to be statistically significant with a p-value less than 10^{-6}.

Table 1: Comparison of Mean Squred Error of Two Models: Random Forest and Formula-Based model

	Model	MSE
0	Random Forest	1.55
1	Formula-Based	3.42
T-test p-value	$<10^{-6}$	

Comparison table showing how well each model performed with respect to Mean Squared Error (MSE)
MSE: Mean Square Error of the Model
Model: Model employed for achieving OTTD ('Random Forest' or 'Formula-Based')

The random forest model achieved this performance after optimization over varied parameters by implementing a grid search process (Table 2). The model showed the optimal parameters to be a maximum of 4 features and 30 estimators. This optimization was guided by 5-fold cross-validation to ensure the robustness and reliability of our model.

Table 2: Best parameters used for the Random Forest model

	Max Features	Number of Estimators
Best Parameters	4	30

Table showing the optimal parameters for the Random Forest model as found by Grid Search
Max Features: The max number of features considered for splitting a node
Number of Estimators: The number of base estimators in the ensemble

We conducted this study with a sizable number of observations amount-

ing to 969. Consequently, the overall accuracy of our optimized regression model for determining optimal tracheal tube depth was computed to be 0.5497.

In essence, the results conspicuously favor the use of the random forest model, integrating multiple patient features, for more accurately determining OTTD in pediatric patients. The study submits that the application of machine learning algorithms can significantly contribute to precise tracheal tip placements, hence aiding better post-operative care in pediatric patients.

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  data = pd.read_csv('tracheal_tube_insertion.csv')
7
8  # prepare the summary statistics
9  summary_stats = data.describe(include='all')
10
11 # count the missing/unknown or undefined values
12 missing_values = data.isnull().sum()
13
14 # count distinct values for categorical variables
15 sex_counts = data['sex'].value_counts()
16
17 # count special numeric values
18 special_numeric_values = data['tube'].value_counts()
19
20 # count total number of rows and columns
21 num_rows = len(data)
22 num_columns = len(data.columns)
23
24 # write to a txt file
```

```
with open('data_exploration.txt', 'w') as file:
    file.write('# Data Size\n')
    file.write('Number of Rows: {}\n'.format(num_rows))
    file.write('Number of Columns:
    ↪  {}\n\n'.format(num_columns))

    file.write('# Summary Statistics\n')
    file.write('{}\n\n'.format(summary_stats))

    file.write('# Categorical Variables\n')
    file.write('sex:\n{}\n\n'.format(sex_counts.to_string()))

    file.write('# Missing Values\n')
    file.write('{}\n\n'.format(missing_values.to_string()))

    file.write('# Special Numeric Values\n')

    ↪  file.write('tube:\n{}\n'.format(special_numeric_values.to_string()))

```

B.2 Code Description

The provided code performs data exploration on the given dataset using Python. The main objective of this code is to analyze the dataset and provide summary statistics, information on missing values, distinct values for categorical variables, and special numeric values.

First, the code reads the dataset from the "tracheal_tube_insertion.csv" file into a pandas DataFrame.

Next, the code calculates the summary statistics, including count, mean, standard deviation, minimum, and quartile values, for all the variables in the dataset using the describe() function. These summary statistics provide an overview of the distribution and central tendencies of the data.

The code then counts the number of missing, unknown, or undefined values for each variable using the isnull().sum() function. This helps in identifying the completeness of the dataset and any potential data quality issues.

To understand the distribution of categorical variables, the code counts the distinct values of the "sex" variable using the value_counts() function. This provides information about the number of male and female patients in the dataset.

In addition, the code counts the occurrences of each unique value in the "tube" variable, which represents the internal diameter of the tracheal tube. This helps in identifying any special numeric values present in the dataset that may require further investigation.

Finally, the code writes the results of the data exploration analysis into a text file named "data_exploration.txt". The file includes the following information:

- The number of rows and columns in the dataset. - The summary statistics of the dataset. - The distinct values and their frequencies for the "sex" variable. - The count of missing values for each variable. - The count of occurrences for each unique value of the "tube" variable.

This file serves as a comprehensive report of the data exploration process, providing a deeper understanding of the dataset's characteristics, potential data quality issues, and special numeric values.

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:
sex
1    522
0    447
```

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

# Special Numeric Values
tube:
tube
3.5    392
4      211
3      205
4.5    112
5       30
5.5     14
2.5      3
3.8      1
6        1
```

C Data Analysis

C.1 Code

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

```

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

# LOAD DATA
```

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

# DATASET PREPARATIONS
# No dataset preparations are needed

# DESCRIPTIVE STATISTICS
# No descriptive statistics table is needed

# PREPROCESSING
# Assigning descriptive labels to the 'sex' column
df['patient_sex'] = df['sex'].map({0: 'female', 1: 'male'})

# Creating dummy variables
preprocessed_df = pd.get_dummies(df, columns=['patient_sex'])

# ANALYSIS

## Table 1: Comparison of MSE of Two Models - Random Forest and Formula-Based Model

# Dividing into train and test datasets
X = preprocessed_df[['patient_sex_male', 'age_c', 'ht', 'wt']]
y = preprocessed_df['tube_depth_G']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Building and training the random forest model
forest_reg = RandomForestRegressor()
param_grid = [{'n_estimators': [3, 10, 30], 'max_features': [2, 4, 6]}]
grid_search = GridSearchCV(forest_reg, param_grid, cv=5, scoring='neg_mean_squared_error', return_train_score=True)
grid_search.fit(X_train, y_train)

# Assigning the best model from the grid search
model = grid_search.best_estimator_
pred_rf = model.predict(X_test)

# Building and predicting with the formula-based model
pred_f = X_test['ht'] / 10 + 5
```

```python

# Comparing the mean squared error of the two models
rf_mse = mean_squared_error(y_test, pred_rf)
formula_mse = mean_squared_error(y_test, pred_f)

# Performing a paired t-test between the predictions of the two models
ttest_result = ttest_rel(pred_rf, pred_f)

# Creating a dataframe to store MSEs and p-value
df1 = pd.DataFrame({
    'Model': ['Random Forest', 'Formula-Based'],
    'MSE': [rf_mse, formula_mse]
})
df1.loc['T-test p-value'] = [ttest_result.pvalue, '']
df1.to_pickle('table_1.pkl')

## Table 2: Model Parameters and Hyperparameters
df2 = pd.DataFrame(grid_search.best_params_, index=['Best Parameters'])
df2.to_pickle('table_2.pkl')

# SAVE ADDITIONAL RESULTS
additional_results = {
    'Total number of observations': len(df),
    'accuracy of regression model': model.score(X_test, y_test)
}
with open('additional_results.pkl', 'wb') as f:
    pickle.dump(additional_results, f)

```

C.2 Code Description

The code performs the analysis of a dataset containing information about pediatric patients who received post-operative mechanical ventilation after surgery. The main objective is to determine the Optimal Tracheal Tube Depth (OTTD) using a machine learning model based on patient features.

First, the code loads the dataset from the provided "tracheal_tube_insertion.csv" file. The dataset consists of 969 rows (patients) and 6 columns, including the

tracheal tube ID (not used for machine learning), patient sex, age, height, weight, and the optimal tracheal tube depth as determined by chest X-ray.

Next, the code preprocesses the dataset by assigning descriptive labels to the 'sex' column and creating dummy variables for the 'patient_sex' feature. This allows for the inclusion of categorical variables in the machine learning model.

The analysis starts by dividing the preprocessed dataset into training and testing sets. The features used for the analysis are 'patient_sex_male', 'age_c', 'ht', and 'wt', while the target variable is 'tube_depth_G'.

The code then builds and trains a Random Forest Regressor model using the training set. A grid search is performed to find the best combination of hyperparameters, including the number of estimators and the maximum number of features. The best model is selected based on the negative mean squared error (MSE) score.

After training the Random Forest model, its performance is evaluated by predicting the tracheal tube depth on the testing set. The MSE of the Random Forest model is compared to a formula-based model to assess the performance improvement. The formula-based model predicts the tracheal tube depth based on the patient's height using a simple formula.

A paired t-test is conducted to statistically compare the predictions of the Random Forest model and the formula-based model. The results, including the MSEs and p-value, are stored in a dataframe and saved as "table_1.pkl".

Additionally, the code saves the best parameters obtained from the grid search in a dataframe, which is then saved as "table_2.pkl".

The code also generates additional results. It computes the total number of observations in the dataset and calculates the accuracy of the Random Forest model on the testing set. These results are stored as a dictionary and saved in the "additional_results.pkl" file.

Overall, the code performs a comparison analysis between a Random Forest model and a formula-based model to determine the optimal tracheal tube depth for pediatric patients based on patient features.

C.3 Code Output

table_1.pkl

	Model	MSE
0	Random Forest	1.55146
1	Formula-Based	3.41889

```
T-test p-value      3.777e-56
```

table_2.pkl

```
                 max_features  n_estimators
Best Parameters             4            30
```

additional_results.pkl

```
{
    'Total number of observations': 969,
    'accuracy of regression model': 0.5497    ,
}
```

D LaTeX Table Design

D.1 Code

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

```
1
2  # IMPORT
3  import pandas as pd
4  from my_utils import to_latex_with_note, format_p_value,
   ↪  is_str_in_df, split_mapping, AbbrToNameDef
5
6  # PREPARATION FOR ALL TABLES
7  # Create a shared mapping for labels that are common to all
   ↪  tables
8  shared_mapping: AbbrToNameDef = {
9   'ht': ('Height', 'Patient height in cm'),
10  'wt': ('Weight', 'Patient weight in Kg'),
11  'tube_depth_G': ('OTTD', 'Optimal tracheal tube depth as
    ↪  determined by chest X-ray (in cm)'),
12  'MSE': (None, 'Mean Square Error of the Model'),
13 }
14
15 # TABLE 1
16 df1 = pd.read_pickle('table_1.pkl')
17
18 # FORMAT VALUES
```

```python
# Format p-values using `format_p_value`.
df1.loc['T-test p-value', 'Model'] = format_p_value(df1.loc['T-test p-value', 'Model'])

# RENAME ROWS AND COLUMNS
# Creating a mapping specific for this table using the `shared_mapping`
mapping1 = {k: v for k, v in shared_mapping.items() if is_str_in_df(df1, k)}
mapping1 |= {
 'Model': ['Model', "Model employed for achieving OTTD ('Random Forest' or 'Formula-Based')"],
}
abbrs_to_names1, legend1 = split_mapping(mapping1)
df1 = df1.rename(columns=abbrs_to_names1, index=abbrs_to_names1)

# Save as latex:
to_latex_with_note(
 df1, 'table_1.tex',
 caption="Comparison of Mean Squred Error of Two Models: Random Forest and Formula-Based model",
 label='table:comparison_of_MSE',
 note="Comparison table showing how well each model performed with respect to Mean Squared Error (MSE)",
 legend=legend1)

# TABLE 2
df2 = pd.read_pickle('table_2.pkl')

# RENAME ROWS AND COLUMNS
mapping2 = {k: v for k, v in shared_mapping.items() if is_str_in_df(df2, k)}
mapping2 |= {
 'max_features': ['Max Features', "The max number of features considered for splitting a node"],
 'n_estimators': ['Number of Estimators', "The number of base estimators in the ensemble"]
}
abbrs_to_names2, legend2 = split_mapping(mapping2)
```

```
df2 = df2.rename(columns=abbrs_to_names2,
    index=abbrs_to_names2)

# Save as latex:
to_latex_with_note(
 df2, 'table_2.tex',
 caption="Best parameters used for the Random Forest model",
 label='table:rf_parameters',
 note="Table showing the optimal parameters for the Random
    Forest model as found by Grid Search",
 legend=legend2)

```

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):
```

```python
    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{Comparison of Mean Squred Error of Two Models: Random Forest and
    Formula-Based model}
\label{table:comparison_of_MSE}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lll}
\toprule
 & Model & MSE \\
\midrule
\textbf{0} & Random Forest & 1.55 \\
\textbf{1} & Formula-Based & 3.42 \\
\textbf{T-test p-value} & $<$1e-06 &  \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
```

```
\item Comparison table showing how well each model performed with respect to
    Mean Squared Error (MSE)
\item \textbf{MSE}: Mean Square Error of the Model
\item \textbf{Model}: Model employed for achieving OTTD ('Random Forest' or
    'Formula-Based')
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_2.tex

```
\begin{table}[h]
\caption{Best parameters used for the Random Forest model}
\label{table:rf_parameters}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrr}
\toprule
 & Max Features & Number of Estimators \\
\midrule
\textbf{Best Parameters} & 4 & 30 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item Table showing the optimal parameters for the Random Forest model as found
    by Grid Search
\item \textbf{Max Features}: The max number of features considered for splitting
    a node
\item \textbf{Number of Estimators}: The number of base estimators in the
    ensemble
\end{tablenotes}
\end{threeparttable}
\end{table}
```