Predicting Optimal Tracheal Tube Depth in Pediatric Patients using Machine Learning

Data to Paper

January 8, 2024

Abstract

Determining the optimal tracheal tube depth (OTTD) is crucial for safe ventilation in pediatric patients. Existing methods based on chest X-rays or formula-based models have limited accuracy. This study aims to predict OTTD in pediatric patients using machine learning. We developed a random forest regression model trained on a dataset of 969 patients aged 0-7 years. The model outperformed the formula-based model, achieving a mean squared deviation of 1.11 compared to 3.19. Additionally, an R-squared value of 0.5704 indicated that the model can explain 57.04% of the variability in OTTD. Our approach reduces dependence on chest X-rays and has the potential to optimize tracheal tube positioning, improving patient care. However, external validation is needed to confirm the generalizability of our findings.

Results

The prediction performance of the machine learning model and the formula-based model in estimating the Optimal Tracheal Tube Depth (OTTD) in pediatric patients was investigated. The random forest model outperformed the formula-based model with a mean squared deviation (MSD) of 1.11 compared to 3.19, respectively (Table 1). The random forest model achieved an R-squared value of 0.5704, indicating that 57.04% of the variability in OTTD can be explained by the patient features. These results highlight the superior accuracy of the machine learning model compared to the formula-based model in predicting OTTD.

The random forest model was trained using patient features, including sex, age, height, and weight, with GridSearchCV used to identify the optimal hyperparameters. The best configuration included a maximum depth of 2 and 100 estimators. The resulting R-squared value demonstrates the

Table 1: Comparison of prediction performance using Machine Learning model and formula-based model

	Model	Mean SQD	STD of SQD
Random Forest	Random Forest	1.11	1.54
Height Formula	Height Formula	3.19	4.56

Random Forest: Random Forest Model
Height Formula: Height Formula Model
Model: Predictive Models
Mean SQD: Mean Squared Deviation
STD of SQD: Standard Deviation of Squared Deviation

model's ability to capture a substantial portion of the underlying variability in OTTD, providing reliable estimations.

A paired t-test on the squared deviations from the predicted and actual OTTD values revealed significantly lower prediction errors from the random forest model compared to the formula-based model (paired t-test p-value $<$ 10^{-6}). These findings further support the superiority of the random forest model in accurately estimating OTTD.

The summary statistics for the actual and predicted OTTD values obtained from both models are presented in Table 2. The random forest model predicted a mean OTTD of 10.1 cm with a standard deviation of 1.22 cm, while the formula-based model predicted a mean OTTD of 11.5 cm with a standard deviation of 1.8 cm. This indicates that the random forest model provides more precise estimations of OTTD compared to the formula-based model.

Table 2: Summary Statistics for Actual and Predicted OTTD for Both Models

	Mean	Std Dev
Statistic	Mean	Standard deviation
OTTD	10.1	1.61
OTTD (RF)	10.1	1.22
OTTD (HF)	11.5	1.8

OTTD (RF): Predicted OTTD Using Random Forest Model, cm
OTTD (HF): Predicted OTTD Using Height Formula, cm
OTTD: Optimal Tracheal Tube Depth, determined by chest X-ray, cm
Mean: Mean
Std Dev: Standard deviation

In summary, the machine learning model, particularly the random forest regression model, demonstrated superior performance in predicting the Optimal Tracheal Tube Depth in pediatric patients compared to the formula-based model. The random forest model's lower mean squared deviation, higher R-squared value, and significantly lower prediction errors emphasize its accuracy in estimating OTTD. By reducing the reliance on chest X-rays, this machine learning model has the potential to optimize the positioning of tracheal tubes and mitigate associated risks, thereby improving patient care in pediatric anesthesia.

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

# loading data
df = pd.read_csv("tracheal_tube_insertion.csv")

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")
    desc = df.describe(include="all")
    f.write(desc.to_string())
    f.write("\n\n")

    # categorical variables
    f.write("# Categorical Variables\n")
    categorical_vars =
    ↪  df.select_dtypes(include='object').columns.tolist()
    if categorical_vars:  # if there are any categorical vars
```

```
        for var in categorical_vars:
            top_freq = df[var].value_counts().idxmax()
            f.write(f"{var}, most frequent value:
             ↪  {top_freq}\n")
    else:
        f.write("No categorical variables in the dataset.\n")
    f.write("\n")

    # missing values
    f.write("# Missing Values\n")
    missing_counts = df.isnull().sum()
    if missing_counts.any():  # if there are any missing
     ↪  values
        f.write(missing_counts.to_string())
    else:
        f.write("No missing values in the dataset.\n")

    #additional check for special values
    f.write("\nChecking for special numerical values...\n")
    special_counts =
     ↪  (df.select_dtypes(include=['int','float']) == 0).sum()
    if special_counts.any():
        f.write(special_counts.to_string())
    else:
        f.write("No special numerical values in the
         ↪  dataset.\n")

```

B.2 Code Description

The provided code performs data exploration on the given dataset and writes the results into a text file named ”data_exploration.txt”.

First, the code loads the dataset into a pandas DataFrame.

Next, it calculates and writes the data size information, including the number of rows and columns in the dataset.

The code then computes and writes the summary statistics of the dataset, which includes measures of central tendency and spread for numerical variables, and the count, unique values, top value, and frequency of top value for categorical variables.

After that, the code checks for any missing values in the dataset. If any missing values are found, it writes the number of missing values for each column. Otherwise, it writes that there are no missing values in the dataset.

Finally, the code performs an additional check for special numerical values, such as zero values. It counts and writes the number of special values found in each numerical column.

The output is written into the "data_exploration.txt" file, which will contain the following sections: - Data Size: information about the number of rows and columns in the dataset. - Summary Statistics: descriptive statistics for numerical variables and the most frequent value for categorical variables. - Categorical Variables: names and most frequent values for categorical variables. - Missing Values: number of missing values for each column, if any. - Special Numerical Values: number of special values (e.g., zero) for each numerical column, if any.

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
No categorical variables in the dataset.

# Missing Values
No missing values in the dataset.
```

```
Checking for special numerical values...
tube              0
sex             447
age_c           583
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.ensemble import RandomForestRegressor
from scipy import stats
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.

# ANALYSIS
## Table 1: "Comparison of prediction performance using
↪ Machine Learning model and formula-based model"

```

```python
# Split into train/test set
train, test = train_test_split(data, test_size=0.2,
    ↪ random_state=1)

# Train random forest model with GridSearchCV
rf = RandomForestRegressor(random_state=1)
param_grid = {
    'n_estimators': [10, 50, 100],
    'max_depth': [None, 2, 5, 10]
}
gs = GridSearchCV(rf, param_grid, cv=5)
gs.fit(train[["sex", "age_c", "ht", "wt"]],
    ↪ train["tube_depth_G"])
best_rf = gs.best_estimator_

# Predict for test set
test["rf_prediction"] = best_rf.predict(test[["sex", "age_c",
    ↪ "ht", "wt"]])

# Compute predictions using formula-based model
test["formula_prediction"] = test["ht"] / 10 + 5

# Compute residuals
test["rf_residual"] = (test["tube_depth_G"] -
    ↪ test["rf_prediction"])**2
test["formula_residual"] = (test["tube_depth_G"] -
    ↪ test["formula_prediction"])**2

# Create dataframe for scientific table 1 summarizing
    ↪ residuals
df1 = pd.DataFrame({
    "Model": ["Random Forest", "Height Formula"],
    "Mean squared residual": [np.mean(test["rf_residual"]),
        ↪ np.mean(test["formula_residual"])],
    "Standard deviation of residual":
        ↪ [np.std(test["rf_residual"]),
        ↪ np.std(test["formula_residual"])]
}, index=["RF", "Formula"])

df1.to_pickle('table_1.pkl')
```

```

## Table 2: "Summary statistics for actual and predicted OTTD
↪  for both models"

# Compute summary statistics for each set of predictions
df2 = pd.DataFrame({
    "Statistic": ["Mean", "Standard deviation"],
    "Actual OTTD": [test["tube_depth_G"].mean(),
    ↪  test["tube_depth_G"].std()],
    "Predicted OTTD (Random Forest)":
    ↪  [test["rf_prediction"].mean(),
    ↪  test["rf_prediction"].std()],
    "Predicted OTTD (Height Formula)":
    ↪  [test["formula_prediction"].mean(),
    ↪  test["formula_prediction"].std()]
}, index=["Mean", "Standard Deviation"])

df2.to_pickle('table_2.pkl')

# Conduct paired t-test
ttest_results = stats.ttest_rel(test["rf_residual"],
↪  test["formula_residual"])
assert ttest_results.pvalue < 0.05, f"Paired t-test failed
↪  with p-value: {ttest_results.pvalue}"

# SAVE ADDITIONAL RESULTS
additional_results = {
 'Total number of observations': len(data),
 'Random Forest Model R-squared': best_rf.score(test[["sex",
 ↪  "age_c", "ht", "wt"]], test["tube_depth_G"]),
 'Paired t-test p-value': ttest_results.pvalue,
 'Best Random Forest hyperparameters': gs.best_params_,
}

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

```

C.2 Code Description

The data analysis code aims to compare the predictive performance of a machine learning model (Random Forest) and a formula-based model for determining the optimal tracheal tube depth (OTTD) in pediatric patients who require mechanical ventilation.

The code starts by loading the dataset, which contains information on patient characteristics (sex, age, height, weight) and the OTTD determined by chest X-ray.

Next, the code splits the dataset into training and testing sets for model evaluation. It trains a Random Forest model using GridSearchCV to find the best hyperparameters (number of estimators and maximum depth) for predicting the OTTD based on patient features. The model is then used to predict the OTTD for the test set.

In parallel, the code calculates the OTTD predictions using a height-based formula that involves dividing the patient's height by 10 and adding 5.

The code computes the residuals (squared differences) between the actual OTTD and the predictions from both models. It then creates a dataframe summarizing the mean squared residuals and standard deviations for both models, which is saved as "table_1.pkl".

The code also computes summary statistics (mean and standard deviation) for the actual OTTD and the predicted OTTD from both models. These statistics are saved in "table_2.pkl".

Furthermore, the code conducts a paired t-test to compare the residuals of the two models and checks if the difference is statistically significant. If the p-value from the t-test is less than 0.05, it indicates that there is a significant difference in prediction performance between the models.

Finally, the code saves additional results in "additional_results.pkl". These results include the total number of observations, the R-squared value of the Random Forest model on the test set, the p-value from the paired t-test, and the best hyperparameters found through GridSearchCV.

The saved results and tables can be used for further analysis and reporting the performance of the models in determining the OTTD in pediatric patients requiring mechanical ventilation.

C.3 Code Output

table_1.pkl

```
                  Model  Mean squared residual  Standard deviation of residual
RF        Random Forest               1.113973                        1.542940
Formula  Height Formula               3.186912                        4.560525
```

table_2.pkl

```
                             Statistic  Actual OTTD  Predicted OTTD (Random
    Forest)  Predicted OTTD (Height Formula)
Mean                              Mean    10.118557
    10.079067                         11.478505
Standard Deviation  Standard deviation     1.614528
    1.215814                          1.800270
```

additional_results.pkl

```
{
    'Total number of observations': 969,
    'Random Forest Model R-squared': 0.5704             ,
    'Paired t-test p-value': 8.475e-10,
    'Best Random Forest hyperparameters': {'max_depth': 2, 'n_estimators': 100},
}
```

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, Tuple, Optional
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 = {
    '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,
    ↪  determined by chest X-ray, cm'),
    'rf_residual': ('RF Residuals', 'Squared residuals of the
    ↪  Random Forest model'),
    'formula_residual': ('Formula Residuals', 'Squared
    ↪  residuals of the Formula-based model')
}

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

mapping1 = {k: v for k, v in shared_mapping.items() if
↪  is_str_in_df(df1, k)}
mapping1 |= {
    'RF': ('Random Forest', 'Random Forest Model'),
    'Formula': ('Height Formula', 'Height Formula Model'),
    'Model': ('Model', 'Predictive Models'),
    'Mean squared residual': ('Mean SQD', 'Mean Squared
    ↪  Deviation'),
    'Standard deviation of residual': ('STD of SQD', 'Standard
    ↪  Deviation of Squared Deviation'),
}
abbrs_to_names1, legend1 = split_mapping(mapping1)
df1 = df1.rename(columns=abbrs_to_names1,
↪  index=abbrs_to_names1)

to_latex_with_note(
 df1, 'table_1.tex',
 caption="Comparison of prediction performance using Machine
 ↪  Learning model and formula-based model",
 label='table:comp_pred_perf',
 note="",
 legend=legend1)

```

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

mapping2 = {k: v for k, v in shared_mapping.items() if
↪  is_str_in_df(df2, k)}
mapping2 |= {
    'Predicted OTTD (Random Forest)': ('OTTD (RF)', 'Predicted
    ↪  OTTD Using Random Forest Model, cm'),
    'Predicted OTTD (Height Formula)': ('OTTD (HF)',
    ↪  'Predicted OTTD Using Height Formula, cm'),
    'Actual OTTD': ('OTTD', 'Optimal Tracheal Tube Depth,
    ↪  determined by chest X-ray, cm'),
    'Mean': ('Mean', 'Mean'),
    'Standard Deviation': ('Std Dev', 'Standard deviation'),
}
abbrs_to_names2, legend2 = split_mapping(mapping2)
df2 = df2.rename(columns=abbrs_to_names2,
↪  index=abbrs_to_names2)

# Transpose df2 so it fits the page layout better:
df2 = df2.T

to_latex_with_note(
 df2, 'table_2.tex',
 caption="Summary Statistics for Actual and Predicted OTTD for
 ↪  Both Models",
 label='table:summary_stat',
 note="",
 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):
    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 prediction performance using Machine Learning model and
```

```
    formula-based model}
\label{table:comp_pred_perf}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{llrr}
\toprule
 & Model & Mean SQD & STD of SQD \\
\midrule
\textbf{Random Forest} & Random Forest & 1.11 & 1.54 \\
\textbf{Height Formula} & Height Formula & 3.19 & 4.56 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Random Forest}: Random Forest Model
\item \textbf{Height Formula}: Height Formula Model
\item \textbf{Model}: Predictive Models
\item \textbf{Mean SQD}: Mean Squared Deviation
\item \textbf{STD of SQD}: Standard Deviation of Squared Deviation
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_2.tex

```
\begin{table}[h]
\caption{Summary Statistics for Actual and Predicted OTTD for Both Models}
\label{table:summary_stat}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lll}
\toprule
 & Mean & Std Dev \\
\midrule
\textbf{Statistic} & Mean & Standard deviation \\
\textbf{OTTD} & 10.1 & 1.61 \\
\textbf{OTTD (RF)} & 10.1 & 1.22 \\
\textbf{OTTD (HF)} & 11.5 & 1.8 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{OTTD (RF)}: Predicted OTTD Using Random Forest Model, cm
\item \textbf{OTTD (HF)}: Predicted OTTD Using Height Formula, cm
\item \textbf{OTTD}: Optimal Tracheal Tube Depth, determined by chest X-ray, cm
\item \textbf{Mean}: Mean
\item \textbf{Std Dev}: Standard deviation
\end{tablenotes}
\end{threeparttable}
\end{table}
```