

Machine Learning Models for Prediction of Optimal Tracheal Tube Depth in Pediatric Patients Undergoing Mechanical Ventilation

Data to Paper

February 20, 2024

Abstract

Accurate determination of the optimal tracheal tube depth (OTTD) is crucial in pediatric patients undergoing mechanical ventilation to prevent potential complications and ensure patient safety. However, current methods for determining OTTD have limitations in accuracy and efficiency. In this study, we present a machine learning approach using Random Forest and Elastic Net models to predict OTTD based on patient features. Our analysis utilizes a dataset of pediatric patients who received post-operative mechanical ventilation between January 2015 and December 2018 at Samsung Medical Center. The models show promising results with low mean squared residuals, indicating overall predictive accuracy. We also identify optimal hyperparameters through grid search, further improving the models' performance. However, these findings require validation on independent datasets, and predictive accuracy may vary based on specific patient characteristics. Nevertheless, our study provides valuable insights into improving the accuracy and efficiency of determining OTTD in pediatric patients, potentially reducing the risks associated with tracheal tube misplacement and enhancing patient safety.

Results

Commencing with a dataset containing 969 observations, two machine learning models were deployed to predict the optimal tracheal tube depth (OTTD), namely a Random Forest (RF) model and an Elastic Net (EN) model. The hyperparameters for these models were tuned using grid search.

An essential part of our study was assessing the performance of both RF and EN models. We evaluated this by comparing the squared residuals,

which provide a numerical measure of the difference between the model's OTTD predictions and the actual OTTD as determined by chest X-rays. As shown in Table 1, the mean squared residuals for the RF and EN models were 1.47 and 1.24, respectively, indicating that on average, both models achieved similar performance, with the EN model providing somewhat more accurate predictions. In parallel, the standard deviation of the squared residuals was also found to be lower for the EN model.

Table 1: Summary statistics of the squared residuals for test datasets using the RF and EN models.

	Avg. Sq. Residuals	Std. Sq. Residuals
RF	1.47	3.24
EN	1.24	2.32

Avg. Sq. Residuals: Average of the squared residuals

Std. Sq. Residuals: Standard deviation of the squared residuals

RF: Random Forest Machine Learning Model

EN: Elastic Net Machine Learning Model

For thoroughness, we also carefully inspected the influence of hyperparameters on our models. The optimal hyperparameters, which were determined via grid search, are presented in Table 2. For the RF model, the optimal results were observed with a maximum tree depth of 7, a minimum number of samples per leaf of 2, and a forest size of 50 trees. In contrast, the optimal parameters for the EN model were an alpha of 0.001 and an l1_ratio of 1.

Table 2: Optimal hyperparameters for the RF and EN models.

Parameters	
RF	'max_depth': 7, 'min_samples_leaf': 2, 'n_estimators': 50
EN	'alpha': 0.001, 'l1_ratio': 1

RF: Random Forest Machine Learning Model

EN: Elastic Net Machine Learning Model

Parameters: Optimal model parameters found via grid search

A critical aspect of our results pertains to the statistical comparison of the squared residuals between the RF and EN models. We undertook a paired t-test, which yielded a statistic value of 1.797, and the associated p-value equaled 0.07386. While this p-value is greater than typically accepted thresholds for statistical significance (usually p-value < 0.05), it still provides

useful information about the comparative performance of the RF and EN models.

To summarize, both the RF and EN models were found to have comparably effective performances in predicting the OTTD, as evidenced by the squared residuals. The EN model, however, had a slightly lower average of squared residuals. Our use of grid search also allowed us to identify the optimal hyperparameters for both models. Additionally, the paired t-test comparison of residuals from both models failed to confirm a statistically significant difference in their performance.

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
4 # loading the dataset
5 df = pd.read_csv('tracheal_tube_insertion.csv')
6
7 # opening a file to write the summary
8 with open('data_exploration.txt', 'w') as file:
9
10     # Data Size
11     file.write("# Data Size\n")
12     file.write(f"Number of rows: {df.shape[0]}\n")
13     file.write(f"Number of columns: {df.shape[1]}\n\n")
14
15     # Summary Statistics
16     file.write("# Summary Statistics\n")
17     file.write(df.describe().to_string())
18     file.write("\n\n")
19
20     # Categorical Variables
21     file.write("# Categorical Variables\n")
22     # Checking if sex is the only categorical variable
23     categorical_cols =
        ↪ df.select_dtypes(include='object').columns.tolist()
```

```

24     if 'sex' in categorical_cols:
25         file.write('Sex Variable Most common values:\n')
26         file.write(df['sex'].value_counts().to_string())
27     else:
28         file.write('There are no categorical variables in the
29         ↪ dataset.\n')
30     file.write("\n\n")
31
32     # Missing Values
33     file.write("# Missing Values\n")
34     file.write(df.isnull().sum().to_string())
35     file.write('\n')
36
37     # Check for unknown/undefined numeric values if mentioned
38     ↪ in "Description of the Dataset"
39     # But this information was not provided, so I'm skipping
40     ↪ this part.
41
42 file.close()

```

B.2 Code Description

The provided code conducts a data exploration analysis on the given dataset using the pandas library in Python. The aim is to provide a summary of the dataset and identify any missing values or categorical variables.

The code begins by reading the dataset file "tracheal_tube_insertion.csv" and loading it into a pandas DataFrame. It then opens a file called "data_exploration.txt" to write the analysis results.

The analysis consists of several steps:

1. Data Size: The code writes the number of rows and columns in the dataset to the "data_exploration.txt" file.
2. Summary Statistics: The code computes summary statistics for the dataset using the describe() function provided by pandas. These statistics include count, mean, standard deviation, minimum, quartiles, and maximum for each numerical column in the dataset. The resulting summary statistics table is then written to the "data_exploration.txt" file.
3. Categorical Variables: The code checks if there are any categorical variables in the dataset. It does this by identifying columns with the "object" data type, which typically indicate categorical variables. If the 'sex'

variable is identified as categorical, the code writes the counts of each category (0 for female and 1 for male) to the "data_exploration.txt" file. If there are no categorical variables in the dataset, a message is written indicating this.

4. Missing Values: The code checks for missing values in the dataset using the `isnull()` function provided by pandas. It calculates the count of missing values for each column and writes these counts to the "data_exploration.txt" file.

In summary, the code performs a data exploration analysis on the given dataset, providing a summary of the dataset's size, summary statistics, categorical variables (if any), and missing values counts. The results are written to the "data_exploration.txt" file for further reference and analysis.

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 dataset.

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:

```
1
2 # IMPORT
3 import pandas as pd
4 import numpy as np
5 from scipy import stats
6 from sklearn.model_selection import train_test_split,
   ↪ GridSearchCV
7 from sklearn.ensemble import RandomForestRegressor
8 from sklearn.linear_model import ElasticNet
9 from sklearn.metrics import mean_squared_error
10 import pickle
11
12 # LOAD DATA
13 df = pd.read_csv("tracheal_tube_insertion.csv")
14
15 # DATASET PREPARATIONS
16 # No dataset preparations are needed.
17
18 # DESCRIPTIVE STATISTICS
19 # No descriptive statistics table is needed.
20
21 # PREPROCESSING
22 # No preprocessing is needed, because the provided features
   ↪ are already numerical.
23
24 # ANALYSIS
25 X = df[['sex', 'age_c', 'ht', 'wt']]
26 y = df['tube_depth_G']
27
```



```

28 X_train, X_test, y_train, y_test = train_test_split(X, y,
    ↪ test_size=0.2, random_state=42)
29
30 # Create Regressor with Grid Search for RandomForest
31 rf_regressor = RandomForestRegressor(random_state=42)
32 grid_values_rf = {'n_estimators': [50, 100], 'max_depth' :
    ↪ [None, 7, 10], 'min_samples_leaf' : [1, 2]}
33 grid_search_rf = GridSearchCV(rf_regressor, param_grid =
    ↪ grid_values_rf, scoring = 'neg_mean_squared_error', cv =
    ↪ 5)
34 grid_search_rf.fit(X_train, y_train)
35
36 # Create Regressor with Grid Search for ElasticNet
37 en_regressor = ElasticNet(random_state=42)
38 grid_values_en = {'alpha': [0.001, 0.01, 0.1, 1], 'l1_ratio':
    ↪ [0.5, 0.7, 1]}
39 grid_search_en = GridSearchCV(en_regressor, param_grid =
    ↪ grid_values_en, scoring = 'neg_mean_squared_error', cv =5)
40 grid_search_en.fit(X_train, y_train)
41
42 # Predictions
43 rf_predictions =
    ↪ grid_search_rf.best_estimator_.predict(X_test)
44 en_predictions =
    ↪ grid_search_en.best_estimator_.predict(X_test)
45
46 # residuals
47 rf_resid = (rf_predictions - y_test) ** 2
48 en_resid = (en_predictions - y_test) ** 2
49
50 ## Table 1: "Summary statistics of squared residuals for
    ↪ Random Forest and Elastic Net models"
51 table_1 = {'Mean Squared Residuals': [np.mean(rf_resid),
    ↪ np.mean(en_resid)],
52           'Std. Dev. Squared Residuals': [np.std(rf_resid),
    ↪ np.std(en_resid)]}
53 labels = ['Random Forest', 'Elastic Net']
54
55 df1 = pd.DataFrame(table_1, index=labels)
56 df1.to_pickle('table_1.pkl')

```

```

57
58 # Perform paired t-test
59 ttest_results = stats.ttest_rel(rf_resid, en_resid)
60
61 ## Table 2: "Best hyperparameters for Random Forest and
↪ Elastic Net models"
62 table_2 = {'Random Forest':
63 ↪ [str(grid_search_rf.best_params_)],
64 ↪ 'Elastic Net': [str(grid_search_en.best_params_)]}
65
66 labels = ['Hyperparameters']
67
68 df2 = pd.DataFrame(table_2, index=labels)
69 df2.to_pickle('table_2.pkl')
70
71 # SAVE ADDITIONAL RESULTS
72 additional_results = {
73 'Total number of observations': len(df),
74 'RF model MSE': mean_squared_error(y_test, rf_predictions),
75 'EN model MSE': mean_squared_error(y_test, en_predictions),
76 'Paired t-test statistic': ttest_results.statistic,
77 'Paired t-test p-value': ttest_results.pvalue
78 }
79
80 with open('additional_results.pkl', 'wb') as f:
81     pickle.dump(additional_results, f)

```

C.2 Code Description

The provided code performs data analysis to determine the optimal tracheal tube depth (OTTD) for pediatric patients who require mechanical ventilation. The analysis is conducted using a dataset that includes features extracted from patient electronic health records and their corresponding OTTD values determined by chest X-ray.

First, the code imports the necessary libraries and loads the dataset into a pandas DataFrame. No dataset preparations or descriptive statistics tables are required as the provided dataset is already clean and numerical.

Next, the code splits the dataset into input features (X) and the target variable (y). The input features consist of the patient's sex, age, height, and weight, while the target variable is the OTTD. The dataset is then split into

training and testing sets with a ratio of 80:20.

Two regression models, Random Forest and Elastic Net, are used for the analysis. For each model, a grid search is performed to find the best hyperparameters that minimize the mean squared error (MSE). The Random Forest model is optimized for the number of estimators, maximum depth, and minimum samples leaf, while the Elastic Net model is optimized for the alpha parameter and the l1 ratio. The best hyperparameters for each model are stored in a DataFrame and saved as a pickle file.

After training the models, predictions are made on the testing set using the best estimator from each model. The squared residuals between the predicted values and the actual OTTD values are calculated for both models. The summary statistics of the squared residuals, including the mean and standard deviation, are stored in a DataFrame and saved as a pickle file.

Furthermore, a paired t-test is performed to determine if there is a statistically significant difference between the squared residuals of the Random Forest and Elastic Net models. The t-test statistics and p-value are calculated and saved in the `additional_results.pkl` file along with other relevant information, such as the total number of observations and the MSE for both models.

In summary, the code performs data analysis on the pediatric patient dataset to determine the OTTD using Random Forest and Elastic Net regression models. It finds the best hyperparameters for each model, calculates squared residuals, performs a paired t-test, and saves the results for further analysis and interpretation.

C.3 Code Output

`table_1.pkl`

	Mean Squared Residuals	Std. Dev. Squared Residuals
Random Forest	1.466384	3.240586
Elastic Net	1.241655	2.316675

`table_2.pkl`

	Random Forest
Elastic Net	
Hyperparameters	<code>{'max_depth': 7, 'min_samples_leaf': 2, 'n_estimators': 50}</code> <code>{'alpha': 0.001, 'l1_ratio': 1}</code>

additional_results.pkl

```
{
    'Total number of observations': 969,
    'RF model MSE': 1.466,
    'EN model MSE': 1.242,
    'Paired t-test statistic': 1.797,
    'Paired t-test p-value': 0.07386,
}
```

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 typing import Any, Dict, Optional, Tuple
5 from my_utils import to_latex_with_note, format_p_value,
6     ↪ is_str_in_df, split_mapping, AbbrToNameDef
7
8 # PREPARATION FOR ALL TABLES
9 shared_mapping: AbbrToNameDef = {
10     'Mean Squared Residuals': ('Avg. Sq. Residuals', 'Average
11     ↪ of the squared residuals'),
12     'Std. Dev. Squared Residuals': ('Std. Sq. Residuals',
13     ↪ 'Standard deviation of the squared residuals'),
14     'Random Forest': ('RF', 'Random Forest Machine Learning
15     ↪ Model'),
16     'Elastic Net': ('EN', 'Elastic Net Machine Learning
17     ↪ Model'),
18     'Hyperparameters': ('Parameters', 'Optimal model
19     ↪ parameters found via grid search'),
20 }
21
22 # TABLE 1:
23 df1 = pd.read_pickle('table_1.pkl')
24
25 # RENAME ROWS AND COLUMNS
```

```

20 mapping_table_1 = {k: v for k, v in shared_mapping.items() if
    ↪ is_str_in_df(df1, k)}
21 abbrs_to_names, legend = split_mapping(mapping_table_1)
22 df1 = df1.rename(columns=abbrs_to_names, index=abbrs_to_names)
23
24 # Save as latex:
25 to_latex_with_note(
26     df1,
27     'table_1.tex',
28     caption="Summary statistics of the squared residuals for test
    ↪ datasets using the RF and EN models.",
29     label='table:summary_statistics',
30     legend=legend)
31
32 # TABLE 2:
33 df2 = pd.read_pickle('table_2.pkl')
34
35 # Transpose df2 to make it narrower
36 df2 = df2.T
37
38 # RENAME ROWS AND COLUMNS
39 mapping_table_2 = {k: v for k, v in shared_mapping.items() if
    ↪ is_str_in_df(df2, k)}
40 abbrs_to_names, legend = split_mapping(mapping_table_2)
41 df2 = df2.rename(columns=abbrs_to_names, index=abbrs_to_names)
42
43 # Save as latex:
44 to_latex_with_note(
45     df2,
46     'table_2.tex',
47     caption="Optimal hyperparameters for the RF and EN models.",
48     label='table:hyperparameters',
49     legend=legend)
50

```

D.2 Provided Code

The code above is using the following provided functions:

```

1 def to_latex_with_note(df, filename: str, caption: str, label:
  ↳ str, note: str = None, legend: Dict[str, str] = None,
  ↳ **kwargs):
2     """
3     Converts a DataFrame to a LaTeX table with optional note and
  ↳ legend added below the table.
4
5     Parameters:
6     - df, filename, caption, label: as in `df.to_latex`.
7     - note (optional): Additional note below the table.
8     - legend (optional): Dictionary mapping abbreviations to full
  ↳ names.
9     - **kwargs: Additional arguments for `df.to_latex`.
10
11     Returns:
12     - None: Outputs LaTeX file.
13     """
14
15 def format_p_value(x):
16     returns "{:.3g}".format(x) if x >= 1e-06 else "<1e-06"
17
18 def is_str_in_df(df: pd.DataFrame, s: str):
19     return any(s in level for level in getattr(df.index,
  ↳ 'levels', [df.index]) + getattr(df.columns, 'levels',
  ↳ [df.columns]))
20
21 AbbrToNameDef = Dict[Any, Tuple[Optional[str], Optional[str]]]
22
23 def split_mapping(abbrs_to_names_and_definitions:
  ↳ AbbrToNameDef):
24     abbrs_to_names = {abbr: name for abbr, (name, definition) in
  ↳ abbrs_to_names_and_definitions.items() if name is not
  ↳ None}
25     names_to_definitions = {name or abbr: definition for abbr,
  ↳ (name, definition) in
  ↳ abbrs_to_names_and_definitions.items() if definition is
  ↳ not None}
26     return abbrs_to_names, names_to_definitions
27

```

D.3 Code Output

table_1.tex

```
\begin{table}[h]
\caption{Summary statistics of the squared residuals for test datasets using the
        RF and EN models.}
\label{table:summary_statistics}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrr}
\toprule
& Avg. Sq. Residuals & Std. Sq. Residuals \\
\midrule
\textbf{RF} & 1.47 & 3.24 \\
\textbf{EN} & 1.24 & 2.32 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Avg. Sq. Residuals}: Average of the squared residuals
\item \textbf{Std. Sq. Residuals}: Standard deviation of the squared residuals
\item \textbf{RF}: Random Forest Machine Learning Model
\item \textbf{EN}: Elastic Net Machine Learning Model
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_2.tex

```
\begin{table}[h]
\caption{Optimal hyperparameters for the RF and EN models.}
\label{table:hyperparameters}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{ll}
\toprule
& Parameters \\
\end{tabular}}
\end{threeparttable}
```

```

\midrule
\textbf{RF} & {'max\_depth': 7, 'min\_samples\_leaf': 2, 'n\_estimators': 50} \\
\textbf{EN} & {'alpha': 0.001, 'l1\_ratio': 1} \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{RF}: Random Forest Machine Learning Model
\item \textbf{EN}: Elastic Net Machine Learning Model
\item \textbf{Parameters}: Optimal model parameters found via grid search
\end{tablenotes}
\end{threeparttable}
\end{table}

```