Accurate Determination of Optimal Tracheal Tube Depth in Pediatric Patients using Machine Learning

Data to Paper

January 7, 2024

Abstract

Pediatric patients requiring mechanical ventilation often face challenges in achieving optimal tracheal tube placement, which has a significant impact on patient safety. The current gold standard for determining tracheal tube depth involves chest X-ray, which is time-consuming and exposes patients to radiation. We conducted a study to address this issue by developing a data-driven approach for accurately determining the Optimal Tracheal Tube Depth (OTTD) in pediatric patients. Our dataset comprised pediatric patients aged 0-7 years who received post-operative mechanical ventilation at Samsung Medical Center between 2015 and 2018. Leveraging machine learning techniques, specifically Random Forest and Elastic Net models, we predicted the OTTD based on patient features extracted from electronic health records. Our results demonstrated the effectiveness of our approach, with the models achieving mean residuals of 0.0366 cm and 0.0379 cm, respectively. These findings have important implications for improving tracheal tube placement safety and efficacy, offering an alternative to the traditional chest X-ray method. However, it is important to note that further validation and investigation are required to establish the reliability and generalizability of our data-driven approach.

Results

To address the challenges in determining the Optimal Tracheal Tube Depth (OTTD) in pediatric patients, we conducted a data-driven analysis using machine learning techniques. Here, we present the results of our analysis, which aimed to accurately predict the OTTD based on patient features extracted from electronic health records.

First, we performed an analysis of descriptive statistics for male and female patients to understand the distribution of age and height among different sexes (Table 1). This analysis is crucial as pediatric patients have a narrower safety margin for tracheal tube tip positioning compared to adults. We found that the mean age for male patients was 0.781 years (SD = 1.47) and the mean height was 66.5 cm (SD = 19.4). For female patients, the mean age was 0.732 years (SD = 1.4) and the mean height was 65.4 cm (SD = 18.7). These statistics provide valuable insights into the age and height distribution that may have implications on determining the OTTD.

Table 1: Descriptive statistics for male and female patients

		Age	Ht
Male	**count**	522	522
	mean	0.781	66.5
	std	1.47	19.4
	min	0	29.5
	25%	0	52
	50%	0	62
	75%	1	76
	max	7	135
Female	**count**	447	447
	mean	0.732	65.4
	std	1.4	18.7
	min	0	31
	25%	0	51.8
	50%	0	61
	75%	1	76.3
	max	7	125

The table provides the count, mean, standard deviation, minimum, 25th, 50th and 75th percentiles, and maximum of the patients' age and height stratified by sex
Age: Age in years, rounded to half years
Ht: Height in cm

Next, we conducted hyperparameter tuning and model performance evaluation for the Random Forest (RF) and Elastic Net (EN) models (Table 2). The objective of this analysis was to optimize the models for accurate prediction of the OTTD. We found that the best hyperparameters for the RF model were a maximum depth of 5 and 50 estimators. The mean residual for the RF model was 0.0366 cm (SD = 1.18), indicating a small deviation from

the actual OTTD. Similarly, the best hyperparameters for the EN model were an alpha of 0.1 and an l1 ratio of 0. The mean residual for the EN model was 0.0379 cm (SD = 1.12). The t-test results showed no significant difference between the two models, with a t-statistic of 1.83 and a p-value of 0.0691.

Table 2: Hyperparameter tuning and performance evaluation for RF and EN

Model	RF	EN
Model Parameters	'max_depth': 5, 'n_estimators': 50	'alpha': 0.1, 'l1_ratio': 0
Res Mean	0.0366	0.0379
Res STD	1.18	1.12
T-Stat	1.83	1.83
P-value	0.0691	0.0691

Best parameters for each model are obtained by GridSearchCV. The mean and standard deviation of residuals, and the T-statistic and p-value from the paired students t-test are also reported
Model Parameters: Best parameters obtained by GridSearchCV
Res Mean: Mean of residuals (cm)
Res STD: STD of residuals (cm)
T-Stat: T-statistic from paired t-test
P-value: P-value from paired t-test
RF: Random Forest
EN: Elastic Net

In addition to the model performance evaluation, we analyzed the residuals obtained from both the RF and EN models using additional results (Additional Results). The analysis included a total of 969 observations. The RF model demonstrated a maximum positive residual of 5.321 cm and a maximum negative residual of -2.973 cm, indicating slight deviations from the predicted values. Similarly, the EN model had a maximum positive residual of 4.519 cm and a maximum negative residual of -2.699 cm. These residuals provide insights into the accuracy and reliability of the models in predicting the OTTD.

In summary, our analysis of the RF and EN models demonstrates their efficacy in accurately determining the Optimal Tracheal Tube Depth in pediatric patients. The models achieved mean residuals of 0.0366 cm and 0.0379 cm, respectively, indicating a small deviation from the actual OTTD. These results provide valuable information for improving tracheal tube placement safety and efficacy in pediatric patients requiring mechanical ventilation.

Note that further investigation and validation are critical to fully estab-

lish the reliability and generalizability of our data-driven approach. However, the results presented in this analysis offer promising insights into the accurate determination of OTTD based on patient features, providing an alternative to the time-consuming and radiation-exposing chest X-ray method.

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 csv data into a data frame
6  df_tracheal_tube = pd.read_csv("tracheal_tube_insertion.csv")
7
8  # Start writing the 'data_exploration.txt' file
9  with open('data_exploration.txt', 'w') as f:
10
11     # Data size
12     f.write("# Data Size\n")
13     f.write(f"Number of Rows: {df_tracheal_tube.shape[0]}\n")
14     f.write(f"Number of Columns:
       ↪  {df_tracheal_tube.shape[1]}\n\n")
15
16     # Summary Statistics
17     f.write("# Summary Statistics\n")
18     summary_stats =
       ↪  df_tracheal_tube.describe(include=[np.number]).transpose().to_string()
19     f.write(f"{summary_stats}\n\n")
20
21     # Categorical Variables
22     f.write("# Categorical Variables\n")
```

```
    # Assuming 'sex' as the only categorical variable in this case
    male_count = df_tracheal_tube['sex'].value_counts()[1]
    female_count = df_tracheal_tube['sex'].value_counts()[0]
    f.write(f"Sex: Male count = {male_count}, Female count = {female_count}\n\n")

    # Missing Values
    f.write("# Missing Values\n")
    missing_values = df_tracheal_tube.isnull().sum().to_string()
    f.write(f"{missing_values}\n\n")

    # Since the dataset description does not provide indication of special cases of numeric values standing in for unknown or undefined values, this section is not added to the code.

```

B.2 Code Description

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

B.3 Data Size

First, the code determines the number of rows and columns in the dataset using the shape attribute of the pandas DataFrame. These values are then written to the "data_exploration.txt" file.

B.4 Summary Statistics

Next, the code computes summary statistics for the numerical columns in the dataset using the describe() function. These statistics include count, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum values. This information provides a basic understanding of the distribution and range of values in each column, which can help identify outliers or anomalies in the data.

B.5 Categorical Variables

In this dataset, the only categorical variable is the "sex" column, which represents the patient's sex. The code counts the number of males and females by using the value_counts() function on the "sex" column. These counts are then written to the "data_exploration.txt" file.

B.6 Missing Values

The code checks for missing values in the dataset using the isnull() function. The number of missing values in each column is computed using the sum() function, and these values are then written to the "data_exploration.txt" file.

The "data_exploration.txt" file will contain the following information: - The number of rows and columns in the dataset. - Summary statistics for the numerical columns, including count, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum values. - The count of males and females in the dataset. - The number of missing values in each column.

This information provides a concise overview of the dataset, helping to understand its characteristics, distribution, and potential data quality issues. It lays the foundation for further analysis and modeling tasks.

B.7 Code Output

data_exploration.txt

```
# Data Size
Number of Rows: 969
Number of Columns: 6

# Summary Statistics
              count   mean    std  min  25%  50%  75%   max
tube            969  3.694 0.5681  2.5  3.5  3.5    4     6
sex             969 0.5387 0.4988    0    0    1    1     1
age_c           969  0.758   1.44    0    0    0    1     7
ht              969     66  19.08 29.5   52 61.5   76 135.4
wt              969  7.127  4.774 0.57 3.56  5.7  9.5    31
tube_depth_G    969  10.19  1.766  5.9    9  9.8 11.2  19.2

# Categorical Variables
```

```
Sex: Male count = 522, Female count = 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:

```
1
2  # IMPORT
3  import pandas as pd
4  from sklearn.preprocessing import StandardScaler
5  from sklearn.model_selection import train_test_split,
   ↪  GridSearchCV
6  from sklearn.ensemble import RandomForestRegressor
7  from sklearn.linear_model import ElasticNet
8  from sklearn.metrics import mean_squared_error
9  from scipy import stats
10 import numpy as np
11 import pickle
12 import warnings
13
14 # Suppress warnings
15 warnings.filterwarnings("ignore")
16
17 # LOAD DATA
18 df = pd.read_csv('tracheal_tube_insertion.csv')
19
20 # DATASET PREPARATIONS
21 # No dataset preparations are needed.
```

```
22
23  # DESCRIPTIVE STATISTICS
24  df_male = df[df['sex'] == 1]
25  df_female = df[df['sex'] == 0]
26  df0 = pd.concat([df_male.describe()[['age_c', 'ht']],
    ↪  df_female.describe()[['age_c', 'ht']]], keys=['Male',
    ↪  'Female'])
27  df0.to_pickle('table_0.pkl')
28
29  # PREPROCESSING
30  scaler = StandardScaler()
31  df[['age_c_std', 'ht_std', 'wt_std']] =
    ↪  scaler.fit_transform(df[['age_c', 'ht', 'wt']])
32
33  # ANALYSIS
34  features = ['sex', 'age_c_std', 'ht_std', 'wt_std']
35  target = 'tube_depth_G'
36
37  X_train, X_test, y_train, y_test =
    ↪  train_test_split(df[features], df[target], test_size=0.2,
    ↪  random_state=42)
38
39  ## Table 1: "Hyper-parameter tuning and model performance
    ↪  evaluation for Random Forest and Elastic Net"
40  params_rf = {'n_estimators': [10, 50, 100, 200], 'max_depth':
    ↪  [5, 10, 15, 20]}
41  grid_rf = GridSearchCV(RandomForestRegressor(), params_rf,
    ↪  cv=5)
42  grid_rf.fit(X_train, y_train)
43
44  params_en = {'alpha': [0.1, 0.5, 1, 5], 'l1_ratio': [0, 0.3,
    ↪  0.5, 0.7, 1]}
45  grid_en = GridSearchCV(ElasticNet(max_iter=1000000,
    ↪  tol=0.001), params_en, cv=5)
46  grid_en.fit(X_train, y_train)
47
48  rf_predicted = grid_rf.predict(X_test)
49  en_predicted = grid_en.predict(X_test)
50
51  rf_residuals = y_test - rf_predicted
```

```
en_residuals = y_test - en_predicted

paired_ttest_results =
    stats.ttest_rel(np.square(rf_residuals),
    np.square(en_residuals))

df1 = pd.DataFrame({'Model': ['Random Forest', 'Elastic Net'],
                    'Best Parameters':
                    [str(grid_rf.best_params_),
                    str(grid_en.best_params_)],
                    'Residual Mean': [np.mean(rf_residuals),
                    np.mean(en_residuals)],
                    'Residual STD': [np.std(rf_residuals),
                    np.std(en_residuals)],
                    'T-stat': [paired_ttest_results.statistic,
                    paired_ttest_results.statistic],
                    'P-value': [paired_ttest_results.pvalue,
                    paired_ttest_results.pvalue]})
df1.set_index('Model', inplace=True)
df1.to_pickle('table_1.pkl')

# SAVE ADDITIONAL RESULTS
additional_results = {
    'Residuals RF': rf_residuals.tolist(),
    'Residuals EN': en_residuals.tolist(),
    'Total number of observations': len(df),
    'Hyperparameters RF': str(grid_rf.best_params_),
    'Hyperparameters EN': str(grid_en.best_params_)
}
with open("additional_results.pkl", "wb") as f:
    pickle.dump(additional_results, f)

```

C.2 Code Description

The code performs data analysis on a dataset of pediatric patients who received post-operative mechanical ventilation. The goal is to determine the optimal tracheal tube depth (OTTD) for these patients.

The analysis consists of the following steps:

1. Loading the dataset: The code reads the dataset file "tracheal_tube_insertion.csv"

and stores it in a pandas DataFrame.

2. Preprocessing: The code standardizes the numerical features 'age_c', 'ht', and 'wt' using the StandardScaler from scikit-learn.

3. Analysis: The code splits the dataset into training and testing sets. It then performs hyperparameter tuning and model evaluation for two models: Random Forest and Elastic Net.

4. Hyperparameter tuning and model performance evaluation: For each model, the code performs a grid search with cross-validation to find the best combination of hyperparameters. It evaluates the models' performance by predicting the optimal tracheal tube depth on the testing set and calculating the residuals.

5. Statistical analysis: The code performs a paired t-test on the squared residuals of the Random Forest and Elastic Net models to compare their performance.

6. Saving results: The code saves the following results: - "table_0.pkl": Descriptive statistics of the dataset, including summary statistics for male and female patients. - "table_1.pkl": Results of the hyperparameter tuning and model performance evaluation, including the best parameters, residual mean, residual standard deviation, t-statistic, and p-value for the paired t-test. - "additional_results.pkl": Additional results, including the residuals of the Random Forest and Elastic Net models, the total number of observations in the dataset, and the best hyperparameters for each model.

Overall, the code provides a comprehensive analysis of the dataset, evaluating different models and providing statistical comparisons. The results can be used to determine the best model for predicting the optimal tracheal tube depth in pediatric patients undergoing post-operative mechanical ventilation.

C.3 Code Output

table_0.pkl

		age_c	ht
Male	count	522.000000	522.000000
	mean	0.780651	66.514368
	std	1.472808	19.403722
	min	0.000000	29.500000
	25%	0.000000	52.000000
	50%	0.000000	62.000000
	75%	1.000000	76.000000

```
       max      7.000000  135.400000
Female count  447.000000  447.000000
       mean     0.731544   65.400447
       std      1.402500   18.701462
       min      0.000000   31.000000
       25%      0.000000   51.750000
       50%      0.000000   61.000000
       75%      1.000000   76.350000
       max      7.000000  125.300000
```

table_1.pkl

```
                                    Best Parameters  Residual Mean  Residual STD
    T-stat  P-value
Model
Random Forest  {'max_depth': 5, 'n_estimators': 50}       0.036629      1.184989
    1.827821  0.06912
Elastic Net           {'alpha': 0.1, 'l1_ratio': 0}       0.037939      1.121877
    1.827821  0.06912
```

additional_results.pkl

```
{
    'Residuals RF': [-1.781            , 1.304           , 2.329             ,
    0.7262             , 0.9051            , 2.321             , 0.2828            ,
    -1.018             , -1.078            , 0.1145            , 0.4247
    , 1.364              , 0.3602            , -0.5617           , 3.556
    , 0.4768             , -0.6815           , 0.2657            , -0.2609
    , -0.4964              , -1.46             , -0.7158           , -0.5322
    , -0.535             , 0.5166          , 0.265           , 0.05593             ,
    1.778              , 1.768             , -0.9392           , -0.4638           ,
    -1.329             , -0.2258             , -0.2075           , -1.48
    , 1.115              , -0.6191           , 0.2348            , 1.042
    , -1.4               , 1.271             , 0.2472            , 0.09512
    , -0.135               , 0.8587            , 0.4955            , 0.8866
    , 1.273            , 0.6735            , 2.182             , 1.902             ,
    -0.8452            , 0.2095            , 0.2382            , 0.4708
    , -1.371             , 0.2326            , -2.959            , -0.8872
    , 0.8063             , -1.448            , -0.6796           , 2.052
    , -0.2666              , 0.02657             , 0.2132            , 2.839
```

```
, -0.6583          , 0.5051          , 0.8069          , -1.166          ,
0.5905          , 0.1682          , -0.8117          , -0.3516
, 3.685          , -0.05769          , 0.3309          , -0.1842
, 0.4417          , 0.5848          , 0.5688          , 0.9519
, -0.7148          , 1.376          , 5.321          , 0.8546
, -2.973          , -0.2106          , -0.2825          , -0.8044
, -0.7454          , 0.01214          , 0.5133          , -1.584
, 0.3403          , 1.871          , -0.5101          , 0.6294
, -0.1627          , 0.2926          , -2.397          , -1.386
, 1.144          , -1.663          , 0.2028          , 1.494
, -0.3489          , 0.9601          , 0.9127          , 0.9319
, -0.554          , -0.5322          , -1.33          , 0.03365
, -0.2206          , -1.127          , -0.07497          , -0.4651
, -0.5853          , -0.8931          , -0.2077          , 1.598
, -1.557          , -0.8665          , 0.7496          , 0.2616
, 0.9462          , 0.1324          , -0.1897          , -0.1551
, -1.156          , -0.7075          , -0.2751          , -0.7433
, -1.311          , 0.6484          , 0.8154          , 0.4551
, 0.02508          , 3.181          , -0.03595          , 0.2407
, -0.8485          , 0.6057          , -1.394          , -0.1825
, 1.943          , -0.8742          , -0.863          , -0.08376
, 1.639          , -2.058          , -0.2964          , 1.737
, 0.1419          , -0.1863          , 0.09252          , 1.551
, -1.352          , 0.6105          , -0.02031          , 0.7492
, 0.8255          , -0.2701          , -0.9998          , 1.811
, 0.3938          , -2.204          , 0.7866          , -0.3389
, 1.117          , 0.693          , -1.882          , -0.4926
, -0.9695          , -0.7454          , -0.4988          , -0.0902
, -1.306          , -1.304          , 0.3376          , -0.4414
, -0.8604          , -0.5015          , -0.6687          , -2.669
, -0.8897          , -0.07524          , 0.1436          , -2.329
, -0.009909          , -0.2344          , 1.487          ],
'Residuals EN': [-1.555          , 2.562          , 1.999
, 0.3303          , 0.9156          , 2.174          , 0.127
, -1.147          , -1.04          , -0.06432          , 0.8631
, 1.22          , 0.294          , -0.5748          , 3.903
, 0.7221          , -0.721          , 0.1909          , 0.5164
, -0.2835          , -0.9723          , -0.856          , -0.6057
, -0.5494          , 0.555          , 0.1937          , -0.083
, 1.133          , 1.838          , -0.9842          , -0.2446
```

-0.4613	-0.3672	-0.3736	-1.537
1.164	-0.7264	0.0176	1.763
-0.3991	1.483	0.2363	-0.1797
-0.1758	0.4462	0.4522	0.7619
1.094	0.4965	1.937	0.8646
-0.9725	0.04383	0.1121	0.4554
-1.404	0.2013	-2.408	-0.8584
0.766	-0.6698	-0.9006	1.847
-0.2818	0.3235	0.08021	1.613
-0.6218	0.4946	0.5615	-1.325
0.9321	0.6957	-0.5832	-0.6691
4.109	0.2231	0.332	-0.3525
0.4939	0.5615	0.7186	0.7202
-0.7359	1.289	4.519	0.6879
-2.699	-0.1503	-0.1384	-0.7995
-0.705	0.1405	0.512	-1.371
0.3189	1.756	0.04182	0.669
-0.2019	0.06515	-1.903	-1.005
0.9593	-1.027	0.008871	1.454
-0.2654	1.11	1.158	0.7897
-0.6625	-0.5753	-1.522	0.1238
-0.3495	-0.9772	-0.01504	-0.5989
-0.7868	-1.143	-0.4654	1.51
-1.779	-0.8686	0.6898	0.2857
1.392	-0.2683	-0.3675	-0.4794
-0.4234	-0.8381	-0.3981	-0.6799
-1.001	0.4184	0.4163	0.3485
0.1416	2.548	0.113	0.1859
-0.7948	0.5678	-1.26	-0.4194
1.401	-0.67	-0.8929	-0.1875
2.247	-2.568	-0.07605	1.864
-0.541	-0.1545	-0.05472	2.299
-1.365	0.5191	0.004334	0.6284
0.6585	-0.4	-0.9103	1.42
0.3544	-1.948	0.4483	-0.213
1.14	0.6286	-1.896	-0.4058
-0.8675	-0.9121	-0.5992	-0.07934
-0.5613	-0.8032	0.1937	-0.5111
-1.051	-0.7625	-1.258	-2.29
-0.8544	0.5115	0.007134	-2.439

```
    , -0.1423             , 0.1465             , 1.163             ],
    'Total number of observations': 969,
    'Hyperparameters RF': "{'max_depth': 5, 'n_estimators': 50}",
    'Hyperparameters EN': "{'alpha': 0.1, 'l1_ratio': 0}",
}
```

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, Optional, Tuple
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 = {
 'age_c': ('Age', 'Age in years, rounded to half years'),
 'ht': ('Ht', 'Height in cm'),
 'sex': ('Sex', 'Patient sex (0=female, 1=male)'),
 'tube_depth_G': ('Depth', 'Optimal tracheal tube depth as determined by chest X-ray in cm'),
 'RF': ('Random Forest', 'Machine learning model'),
 'EN': ('Elastic Net', 'Machine learning model')
}

# TABLE 0
df = pd.read_pickle('table_0.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_0.tex',
 caption="Descriptive statistics for male and female patients",
 label='table:table0',
 note="The table provides the count, mean, standard deviation, minimum, 25th, 50th and 75th percentiles, and maximum of the patients' age and height stratified by sex",
 legend=legend)

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

# FORMAT VALUES
df["P-value"] = df["P-value"].apply(format_p_value)

# RENAME COLUMNS
mapping = {k: v for k, v in shared_mapping.items() if is_str_in_df(df, k)}
mapping |= {
 'Best Parameters': ('Model Parameters', 'Best parameters obtained by GridSearchCV'),
 'Residual Mean': ('Res Mean', 'Mean of residuals (cm)'),
 'Residual STD': ('Res STD', 'STD of residuals (cm)'),
 'T-stat': ('T-Stat', 'T-statistic from paired t-test'),
 'P-value': ('P-value', 'P-value from paired t-test'),
}

abbrs_to_names, legend = split_mapping(mapping)
df = df.rename(columns=abbrs_to_names)
df = df.rename(index={'Random Forest':'RF', 'Elastic Net':'EN'})

# TRANSPOSE THE DATAFRAME
df = df.transpose()

# ADDITIONAL ELEMENTS FOR THE LEGEND
legend |= {'RF': 'Random Forest', 'EN': 'Elastic Net'}

```

```
# Save as LaTeX:
to_latex_with_note(
 df, 'table_1.tex',
 caption="Hyperparameter tuning and performance evaluation for RF and EN",
 label='table:table1',
 note="Best parameters for each model are obtained by GridSearchCV. The mean and standard deviation of residuals, and the T-statistic and p-value from the paired students t-test are also reported",
 legend=legend
)


```

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"

```

```python
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

```
\begin{table}[h]
\caption{Descriptive statistics for male and female patients}
\label{table:table0}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{llrr}
\toprule
 &  & Age & Ht \\
\midrule
\multirow[t]{8}{*}{\textbf{Male}} & \textbf{count} & 522 & 522 \\
\textbf{} & \textbf{mean} & 0.781 & 66.5 \\
\textbf{} & \textbf{std} & 1.47 & 19.4 \\
\textbf{} & \textbf{min} & 0 & 29.5 \\
\textbf{} & \textbf{25\%} & 0 & 52 \\
\textbf{} & \textbf{50\%} & 0 & 62 \\
\textbf{} & \textbf{75\%} & 1 & 76 \\
```

```
\textbf{} & \textbf{max} & 7 & 135 \\
\cline{1-4}
\multirow[t]{8}{*}{\textbf{Female}} & \textbf{count} & 447 & 447 \\
\textbf{} & \textbf{mean} & 0.732 & 65.4 \\
\textbf{} & \textbf{std} & 1.4 & 18.7 \\
\textbf{} & \textbf{min} & 0 & 31 \\
\textbf{} & \textbf{25\%} & 0 & 51.8 \\
\textbf{} & \textbf{50\%} & 0 & 61 \\
\textbf{} & \textbf{75\%} & 1 & 76.3 \\
\textbf{} & \textbf{max} & 7 & 125 \\
\cline{1-4}
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item The table provides the count, mean, standard deviation, minimum, 25th,
    50th and 75th percentiles, and maximum of the patients' age and height
    stratified by sex
\item \textbf{Age}: Age in years, rounded to half years
\item \textbf{Ht}: Height in cm
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_1.tex

```
\begin{table}[h]
\caption{Hyperparameter tuning and performance evaluation for RF and EN}
\label{table:table1}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lll}
\toprule
Model & RF & EN \\
\midrule
\textbf{Model Parameters} & {'max\_depth': 5, 'n\_estimators': 50} & {'alpha':
    0.1, 'l1\_ratio': 0} \\
\textbf{Res Mean} & 0.0366 & 0.0379 \\
```

```
\textbf{Res STD} & 1.18 & 1.12 \\
\textbf{T-Stat} & 1.83 & 1.83 \\
\textbf{P-value} & 0.0691 & 0.0691 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item Best parameters for each model are obtained by GridSearchCV. The mean and
    standard deviation of residuals, and the T-statistic and p-value from the paired
    students t-test are also reported
\item \textbf{Model Parameters}: Best parameters obtained by GridSearchCV
\item \textbf{Res Mean}: Mean of residuals (cm)
\item \textbf{Res STD}: STD of residuals (cm)
\item \textbf{T-Stat}: T-statistic from paired t-test
\item \textbf{P-value}: P-value from paired t-test
\item \textbf{RF}: Random Forest
\item \textbf{EN}: Elastic Net
\end{tablenotes}
\end{threeparttable}
\end{table}
```