Predictive Analysis

Matthew E. Heino

Data Mining I

Predictive Analysis

Introduction

The purpose of this paper is to create a decision tree that will be able to answer a question that is of interest to the organization. The organization in this paper is a medical institution. So the appropriate question will be of a medical nature. The competencies that this paper seeks to address are the ability to postulate a question that is of interest to the organization, be able to define the variables that are required to answer, the question, assess the validity and the accuracy of the model, and be able to propose a course of action. Each of the competencies will be explored throughout the course of this paper.

Background

The background is that this analysis is composed of a supplied CSV file. Within this file, there are 10000 observations and 50 features or columns. Each of these rows has information about a patient. The row contains both personal information like address information and other personal information. The dataset contains a patient's medical history like do they have high blood pressure or other medical history. While there are fifty columns within the dataset, not all of these features will be used in the creation of the model.

Part I. Organizational Research Question

This section will cover a discussion of the proposed research question. There will be a brief discussion of one of the goals that the research question hopes to answer with the proposed model.

A1. The Research Question. The question that would be of interest to the organization, “Is it possible to determine if what is the total number of days spent in the hospital using certain features?" This question will make it possible to see what features can be used to make predictions and what conclusions the organization can make about the available observations.

A2. A Goal of the Data Analysis. The primary goal of this analysis is to determine the total time spent in the hospital using a simple decision tree regression and a few select features from the dataset. This can be accomplished by looking at the past data observations that are included in the data file. This question will look at certain features that are in the dataset.

Part II. Justification of the Method

In this section, there will be a justification offered as to why a decision tree would be an appropriate method for answering the question that was proposed in Section A1. There will be a discussion about one of the assumptions that are associated with a decision tree. The final subsection will go over the packages that will be used in the cleaning of the data and the creation of the decision tree model.

B1. Justification for Using a Decision Tree. Why should we use a decision tree as the basis for answering the research question? A decision tree is a simple way to evaluate the outcomes for a problem – the question that was proposed in section A. The decision tree is a machine-learning algorithm that can handle variables that are both categorical and numeric. We are interested in using the decision tree to find a numeric answer to the question that was proposed in a previous section. We will use a regressor version of the decision. The reason for this is that we are looking for a numeric value as opposed to a binary value.

How does the decision tree work? The decision works by splitting the dataset into smaller and smaller subsets. These subsets are then used to help construct the tree. This division of the data into subsets will eventually lead to a tree that is composed of decision nodes and leaf nodes.

The subsets are composed of or contain instances that are similar in value (Decision Tree - Regression, n.d.).

The decision nodes are composed of values that the predicted target will be tested against. Based on the outcome of the test the testing will continue if there are other nodes in the path or the decision will be made there. The actual decision is referred is the value of the leaf node. No further tests are performed at this point. The leaf node indicates that the decision has run its course and there are no longer any decision nodes to traverse (Decision Tree – Regression, n.d.).

The actual splitting of the dataset is based on the attributes that compose the data. For example, using gender and complication risk. There will be a path created for each of the subcategories within each of the categorical values. This will result in a path to the eventual leaf node.

When the leaf node is finally reached there will be a value assigned to the target value. This value will be the average (if there is more than one instance) of all the instances that match the path that has been traversed to arrive at the leaf node. This will be the target variable's value (Decision Tree – Regression, n.d.).

B2. Assumption of Decision Trees. There are a few assumptions that are associated with a Decision Tree. The one that will discussed here is:

- The records that will compose the decision tree are “distributed recursively” based on the value of the attributes (Vishalmendekarhere, 2021).
    - This assumption is based on the idea that the tree is built based on updates to information as each member of the dataset is compared to the node. As the data is iterated through there will be a look at a metric that will help determine the amount of

information that is contained within the attribute. Metrics that help to determine the selection of the attributes or the nodes are Information Gain and the Gini Index. These metrics can be used to create nodes using the attributes with the highest score. These nodes with the highest score will make up the higher node levels of the tree. This will be done recursively until all the attributes have been added at the appropriate level of the tree (Saxena & Saxena, 2017).

B3. Libraries Used. The programming language that was chosen for this assessment was Python. It is composed of many libraries that will be beneficial in creating the decision tree and analyzing its performance. The table below shows the libraries that were used in the creation of the model along with a brief description of what their purpose is.

Python Library	Purpose
math	Allow for math function for square root.
matplotlib. pyplot	Plot the graphs that are used in the assessment
missingno	Create a visual of the missing data.
pandas	For creating the dataframe and providing methods for manipulating the data within the dataframe
from sklearn.feature_selection import SelectKBest, f_regression	Provides the class for selecting the best features from the candidate features.
from sklearn.model_selection import train_test_split, GridSearchCV	Provides the method train_test_split for splitting the data into training and test sets. Provides a GridSearchCV class that allows for the selection of the optimal parameters for the decision

	tree regressor.
from sklearn.metrics import mean_squared_error, r2_score	Provides metrics measuring the accuracy of the decision tree regressor.
from sklearn import tree	For creating a tree from text and creating a visual representation of the tree.
from sklearn.tree import DecisionTreeRegressor	Provides the class that will create the decision tree regressor.

Table 1. Python libraries

Part III. Data Preparation.

In this section, there will be a brief discussion of the goal of pre-processing the data. There will be an identification of the initial variables that will be candidates for inclusion in the model

C1. A Goal of Preprocessing Data. The goal of preprocessing the data is to get the data that is in a state the model can use. The current dataset is composed of data that is in both numeric and string form. The variables that are in string form are the categorical data types. The categorical variables will need to be transformed into their numeric representation using an appropriate method.

C2. Variables Identified. The variables that will be used to create the decision tree are shown in the table below.

	Variable or Column	Independent or Dependent	Data Type	Data Class
1	Age	Independent	Continuous	Quantitative
2	Gender	Independent	Categorical	Qualitative
3	VitD_levels	Independent	Continuous	Quantitative
4	TotalCharge	Independent	Continuous	Quantitative
5	Initial_days	Dependent	Continuous	Quantitative
6	Initial_admin	Independent	Continuous	Quantitative
7	Additional_charges	Independent	Continuous	Quantitative

C3. Steps Used to Prepare Data. The steps that will be used to clean the data are similar to the ones that were used in Task 1. There will be a few changes. The step to look for outliers will not be included. The decision tree is robust to outliers and this indicates that this step is not to be pursued for this type of model. The reason is based on how the decision tree algorithm works. The decision tree algorithm bases its decision to partition on the proportion of the samples and not on the absolute values of the samples (Brown & Myles, 2009).

Scaling and normalization of the data will not be performed. The rationale for not scaling the data is that the splits of the data into the nodes are not affected by scaling or the normalization of the data (Filho, 2023). In the previous task using a K-Nearest Neighbor algorithm was required because the difference in scale may have a bearing on the creation of the decision boundary of the KNN model. The steps that were followed are:

1. Check for duplicates within the data. There were no duplicates found in the dataset. The code for this operation can be found in section C3 Step 2.
2. Check for missing data. This step was performed and there were no missing values found in the dataset. The code for this operation can be found in section C3 Step 2. Look for missing values in the Jupyter Notebook.
3. View the summary statistics for the data. There were no unusual findings in the data set when comes to the statistics. The code for this will be found in C3. Step 3: Summary Statistics for the Chosen Features.
4. Transform the categorical variables into their numeric equivalent. As done in a previous assessment the pandas method get_dummies will be used. The only caveat is that the dummies columns will be included in the dataframe. This conversion will also take care of the conversion of the categorical from object to the integer data type that is required to use the **sklearn** Decision Tree algorithm.

- Select the best features from the candidate features. The features were selected using the same function that was employed in creating the KNN model in a previous assessment. The only change will be the change in an argument. The **SelectKBest** method will use **f_regression**. This is because the target model will be a regression model and not a classification model. The features that were selected for the model are: TotalCharge, Age, Initial_admin_Emergency Admission, Initial_admin_Elective Admission, Gender_Female, Gender_Male, and Initial_days.

5. Output the cleaned data to a CSV file to be read during the analysis section of the assessment. This file will contain both the predictor and the target variables for the creation of the model.

C4. Copy of the Cleaned Data Set. A copy of the cleaned dataset with the selected features and observations can be found in the file named below:

Heino_Cleaned_Medical_Task_2.csv

Part IV. The Analysis

In this section, there will be a discussion of splitting the data into the appropriate sets. A discussion on the analysis technique that was used to create the model. The code will also be found in this section.

D1. Splitting the Data Into Sets. The set will be read from the cleaned dataset that was created in the final section of C. The training and the test dataset were created using the 80/20 rule of thumb. This means that 80% of the data was used to create the training dataset and 20% of the data was used to create the testing data set. This was accomplished using the same method that was employed in Task 1 but without the **stratify** argument. The splitting of the data will use the **train_test_split** method that is found in the sklearn library. This was omitted since the target variable in this model is a continuous and not a categorical one. Including this argument will result in an error.

The dataset is split in this matter to ensure that the data has samples to train on and samples that can be used to test the model. The test data is a sample of the dataset that the model has not been exposed to. This is to make sure that the model can accurately make predictions using data that it was not trained. The split training and test set data can be found in the following:

- **X_train**: *Heino_X_train_Task_2.csv*
- **X_test**: *Heino_X_test_Task_2.csv*
- **y_train**: *Heino_y_train_Task_2.csv*
- **y_test**: Heino_y_test_Task_2.csv

The code can be found in section D1. Split the Cleaned Dataset of the Jupyter Notebook.

D2. Description of the Analysis Technique. To create the model a few steps were followed to create a model that is well-suited for answering the question that was proposed. One step that was discussed in the previous section was to convert the categorical variables to their numeric representations. This was accomplished using the **get_dummies()** method that is found in the pandas library.

The CSV file was read into a data frame that was partitioned into the training and test sets as described in section D1. Before creating an initial decision tree regressor model was created, the model will undergo some hyperparameter tuning using the **GridSearchCV** method. This method will process a list of arguments that are relevant to the decision tree regressor algorithm. The parameters that will undergo tuning are:

- **max_depth**: The maximum number of levels in the tree of the number of splits before a prediction is required. It is, basically, the maximum number of nodes that must be traversed before coming to a decision or prediction (Saini, 2023).
- **min_samples_leaf**: This is the minimum number of samples that are required to be a leaf node. This parameter is to determine whether a node can be a leaf node based on the minimum number of samples (Fraj, 2018).

The GridSearchCV object will then be fitted to training data that was created earlier. This fitting will produce a series of parameters for the model. These parameters will be the optimal values of max_depth and min_samples_leaf.

The next step will be printing the parameters that make the best parameters for the decision tree regressor. This will be accomplished by looking at the **best_params_** attribute of the GridSearchCV object that was created and fitted in the previous step. The values for these parameters are:

• **max_depth:**	10
• **min_samples_leaf:**	0.01

Next, get the best estimator model from the best_estimator attribute. This will contain the best model with the best parameters that were passed to the **GridSearchCV** class. This model can then be used to make predictions.

The predictions were accomplished using **the predict()** method. These predictions can then be compared against the expected values. These values and how they are related to the expected values will be discussed in the next section concerning how "accurate" the predictions are.

There were no further intermediate calculations done in this section. Any calculations that involve the accuracy of the model will be found in section E of the paper and the Jupyter Notebook.

D3. Code for Prediction Analysis: Code for this can be found in section D 2 of the Jupyter Notebook.

Part V. Summary and Implications

This section will contain a discussion about the accuracy and what mean square error (MSE) means concerning the features of the dataset.

E1. Meaning of the Accuracy and Mean Squared Error. To gauge if a model can successfully be used to predict the value of the target. There are a few metrics that can be employed. The two metrics that will be discussed here are mean squared error (MSE) and root mean squared error (RMSE).

The mean squared error is used when the target variable is continuous. This metric does not apply to a binary target variable. The MSE of the model was ~11.12 days before the model was tuned. After tuning the model received a MSE of ~7.11 days. The means square error is a measure of the mean of the squares of the errors. The error in this model is how far off the model's predictions are from the expected values. With tuning we can reduce the error by about four days.

Looking at the coefficient of determination of the tuned model the model received .983 or roughly 98.3%. To interpret this value means that the model captures 98% percent of the variation within the model. Based on this value and previous values it is safe to to assume that the model can be relied upon to predict the initial days a patient will spend based on the values of the predictor features. The code for these calculations can be found in section E1. Meaning of the Accuracy and Mean Squared Error.

E2. Results and Implications of the Analysis. During the course of model creation, there was an employment of various methods to select both the optimal features and tune the parameters of the final model. With SelectKBest there was the ability to pare down the candidate features for the model. This left only features that will have a high likelihood of predicting the value of the target value.

Using the **GridSearchCV** class there was a rudimentary tuning of the model. Although without tuning the model performed very well without it. Without tuning the model received a .983 r-squared score. The optimal parameters for this model were max_depth of 10 and min_samples_leaf size of 0.01.

The model that was used to answer the research question was a decision tree regressor. This regressor sought to find if it could be possible to predict the number of days that a patient would initially stay in the hospital. The model based on two metrics will be able to predict the length of stay accurately. The r-squared score for the model was .989 (98.9%) for the tuned model. This is a very good score. A score that approaches 1 is considered a very good model. The MSE score for the tuned model was ~7.11 days. This means that we will have a difference of only ~2.67 days. This difference is very good. We can use this model to make predictions that will only be off by about 2.67 days.

The model does not seem to need any further tuning as the model is quite suited to predicting a target value. It could investigated as to whether any more features could be removed and get similar results.

E3. A Limitation of the Analysis. A limitation of a decision tree regression a tree will become unstable with changes in the data. This is because the introduction of new data will cause the tree to be regenerated. This tree could be drastically different from the previous tree instance. This change in the tree will result in a change in the output of the tree. This can be overcome by the use of bagging and boosting (Taylor, 2023).

E4. Proposed Course of Action. The course of action that could be pursued is that based on the model's values for the MSE, RMSE, and the r-squared values this model can reliably predict the number of days a patient may stay during their initial visit. For the organization, it may be beneficial to look into the features that compose the final model. The features that were included in the model are:

- TotalCharge
- Age
- Initial_admin (Emergency Admission, Elective Admission, and Observation)
- Gender (Male & Female)

These features can be probed to see if there are avenues of concern that can be addressed. For example, is there a way to make the initial admission more effective at diagnosing a possible illness more quickly by including more appropriate tests or by including more interaction between the patient and the doctor? More questions can be asked, using this as a stepping stone. For example, is the patient's prior history being adequately taken into account when the diagnosis is made and when the patient is to be discharged?

The doctor visits (Doc_visits) could be included as a feature in a future implementation of the model. This could yield insight into whether the number of visits plays a role in the length of stay in the hospital.

Using the features that are currently included will yield a good estimation of a patient's stay, but the model does not seem to require any further tuning of the parameters. In this regard, the model does not need to be augmented.

Part VI. Demonstration

F1. The Panopto Video. The link below is to the demonstration of the working code as well as a brief explanation as to what each section does.

- Link:

Reference

Part VII: Resources

G. Web Resources.

In this section you will find all the citations that were used in the creation of the code. These sources were used in tandem with the resources that were provided by WGU and DataCamp.

Sklearn.tree.export_text. (n.d.). Scikit-learn. Retrieved December 4, 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.tree.export_text.html

Sklearn.tree.plot_tree. (n.d.). Scikit-learn. Retrieved December 4, 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.tree.plot_tree.html

H. In-text Citations.

In this section, you will find the citations that were used to create the written document all code citations will appear in the previous section unless otherwise noted.

Brown, S. D., & Myles, A. J. (2009). Decision tree modeling in classification. In *Elsevier eBooks* (pp. 541–569). https://doi.org/10.1016/b978-044452701-1.00025-9

Decision Tree – Regression. (n.d.). Saedsayad. Retrieved December 4, 2023, from https://saedsayad.com/decision_tree_reg.htm

Filho, M. (2023, March 24). Do decision trees need feature scaling or normalization? *Forecastegy*. Retrieved December 4, 2023, from https://forecastegy.com/posts/do-decision-trees-need-feature-scaling-or-normalization/

Fraj, M. B. (2018, August 14). In Depth: Parameter tuning for Random Forest - All things AI - Medium. *Medium*. Retrieved December 4, 2023, from https://medium.com/all-things-ai/in-depth-parameter-tuning-for-random-forest-d67bb7e920d

Saini, A. (2023, September 13). *Decision Tree Algorithm – a complete guide*. Analytics Vidhya. Retrieved December 4, 2023, from https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/

Saxena, R., & Saxena, R. (2017, April 21). How Decision Tree Algorithm works – Dataaspirant. *Dataaspirant – A Data Science Portal For Beginners*. https://dataaspirant.com/how-decision-tree-algorithm-works/

Taylor, S. (2023, November 21). *Decision tree*. Corporate Finance Institute. Retrieved December 4, 2023, from https://corporatefinanceinstitute.com/resources/data-science/decision-tree/

Vishalmendekarhere. (2021, December 27). It's all about assumptions, pros & cons – the startup - medium. *Medium*. Retrieved December 4, 2023, from https://medium.com/swlh/its-all-about-assumptions-pros-cons-497783cfed2d