Four Classification Algorithms' Accuracy in Predicting Clinically Important Brain Injury

Teresa M. Vail

Data Science Tools 2 – COMP 4448

Nov. 14, 2022

Introduction – Purpose and Significance of Project

Traumatic brain injury (TBI) is one of the leading causes of death and disability in children around the world; in the U.S. alone, approximately 7,400 deaths, over 600,000 emergency department (ED) visits and over 60,000 hospital admissions of children between the age of zero to 18 years old were caused annually by head trauma (Zhu, et. al, 2014). According to CDC in their report to Congress in 1999, about 1.5 million Americans sustain a TBI each year, and as the cumulative result of past traumatic brain injuries, an estimated 5.3 million Americans are living with a permanent TBI-related disability today, including myself (www.cdc.gov). People who suspect they have concussion may get treated in emergency rooms, urgent care centers, etc., if their symptoms are mild enough. Those who have more severe symptoms, however, may be admitted to a hospital or be required to get an imaging scan – typically cranial computed tomography (CT) - of their brain (Nigrovic, et. al, 2012). There are risks to getting CT scans (e.g., increased lifetime risk of lethal malignancy); therefore, the medical team must balance the pros and cons of diagnosing the patient with clinically important TBI with a CT scan.

Thus, the purpose of this project is to understand the performance of Random Forest, K-Nearest Neighbors (K-NN), Naïve Bayes, and Logistic Regression in predicting someone will have any acute brain finding revealed on CT (aka clinically important brain injury) given their symptoms and risk factors, which are the categorical input variables. Medically, this project is important for understanding which symptoms and/or risk factors have the greatest impacts on acute brain finding revealed on CT and may help reduce the time assessing and tests conducted before determining to take a CT scan on the brain. Statistically and personally, this will deepen my understanding of the four algorithms used for categorical inputs with a binary output variable, and see which algorithm is the most accurate.

Research Question

Which of the four algorithms – Random Forest, K-NN, Naïve Bayes, and Logistic Regression – is the most accurate and which of the four algorithms fares the worst in predicting whether someone will have any acute brain finding revealed on CT given their symptoms and risk factors?

Description of Dataset

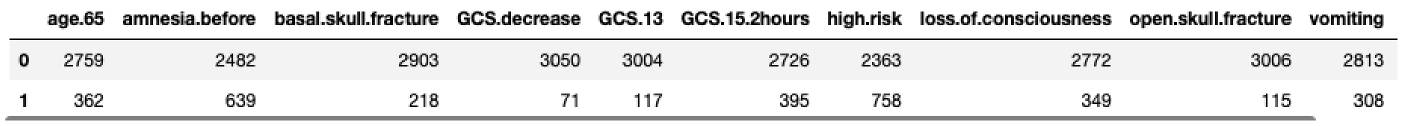
The dataset, called "headInjury", is one of the standard R datasets that can be pulled in R Studio. I've downloaded a csv file from GitHub (see Appendix A for documentation and general website). There are 3121 rows and 11 columns, and the data were simulated according to a simple logistic regression model to match roughly the clinical characteristics of a sample of individuals who suffered minor head injuries. The output variable is clinically important brain injury ("clinically.important.brain.injury"), and it is binary (e.g., 0 = not present, 1 = present). All ten input variables are categorical and binary; they include: age.65, amnesia.before (amnesia before impact), basal.skull.fracture, GCS.decrease (Glasgow Coma Scale decrease), GCS.13 (initial Glasgow Coma Scale), GCS.15.2hours (Glasgow Coma Scale after 2 hours), high.risk (assessed by clinician as high risk for neurological intervention), loss.of.consciousness, open.skull.fracture, and vomiting. The more in-depth descriptions of each input variable and notes on the Glasgow Coma Scale can be found in Appendix A.

Data Preprocessing

There are no missing data in any of the variables of the dataset in the csv file downloaded from the internet. All values of the variables were verified to be 0 or 1, which meant that all variables are indeed categorical and binary. This also meant that the data did not need to be transformed. I checked the data types of the variables to make sure that they meet my expectations of being numeric.

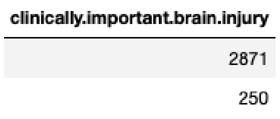
Exploratory Data Analyses

Since all variables are categorical and binary, I checked for distribution of all variables. I did not check for the descriptive stats (e.g., mean, median, max, min, etc.) since they are not feasible.



	age.65	amnesia.before	basal.skull.fracture	GCS.decrease	GCS.13	GCS.15.2hours	high.risk	loss.of.consciousness	open.skull.fracture	vomiting
0	2759	2482	2903	3050	3004	2726	2363	2772	3006	2813
1	362	639	218	71	117	395	758	349	115	308

Table 1: count of all input variables



clinically.important.brain.injury
2871
250

Table 2: count of target variable (for “0” value in top row, for “1” value in bottom row)

All variables have much higher counts of “0” than of “1”. The variable “GCS.decrease” has the highest proportion of “0” value (see Figure 1), and the variable “high.risk” has the highest proportion of “1” value (see Figure 2). All other countplots can be found in Appendix B.

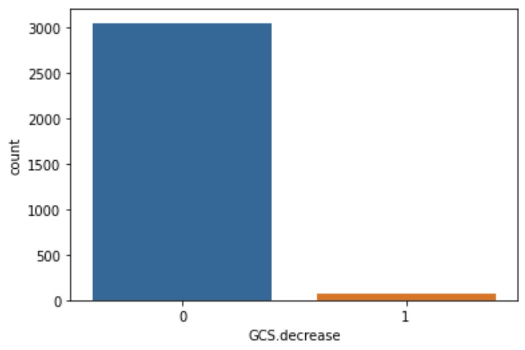


Figure 1: Countplot of GCS.decrease variable

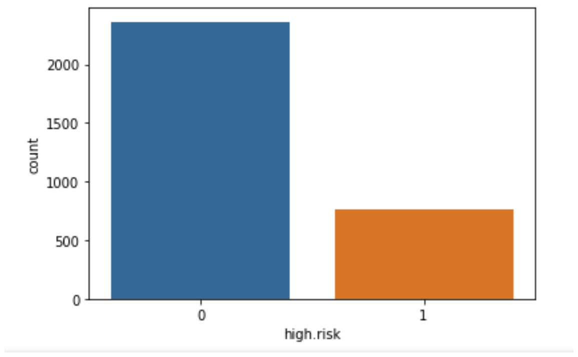


Figure 2: Countplot of high.risk variable

Since there are only 250 instances with clinically important brain injury in the entire dataset, the bars for “clinically.important.brain.injury” are predictably very short in grouped bar charts. The greatest discrepancy between input variable and output variable when input variable has “1” is visible for the “high.risk” variable (see Figure 3), and the smallest discrepancy is for the “GSC.decrease” variable (see Figure 4). This *may* indicate that the “GSC.decrease” variable has the strongest correlation to the “clinically.important.brain.injury” variable and that the “high.risk” variable has the weakest correlation. All other grouped bar charts can be found in Appendix B.

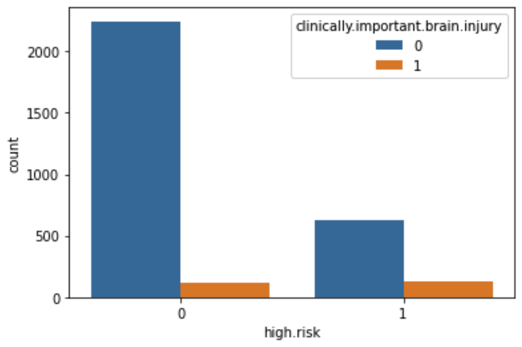


Figure 3: Grouped bar chart of high.risk vs. target variable

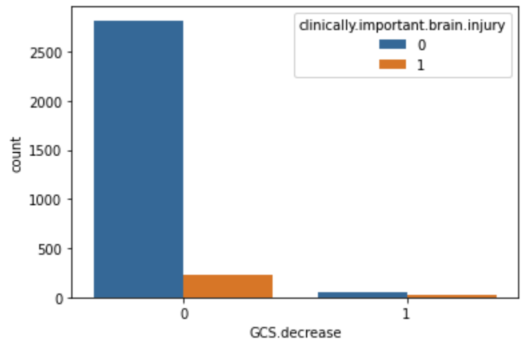


Figure 4: Grouped bar chart of GCS.decrease vs. target variable.

To check my educated guesses, the correlation between variables are found with .corr() in Python, and the heatmap of the correlation is shown here:

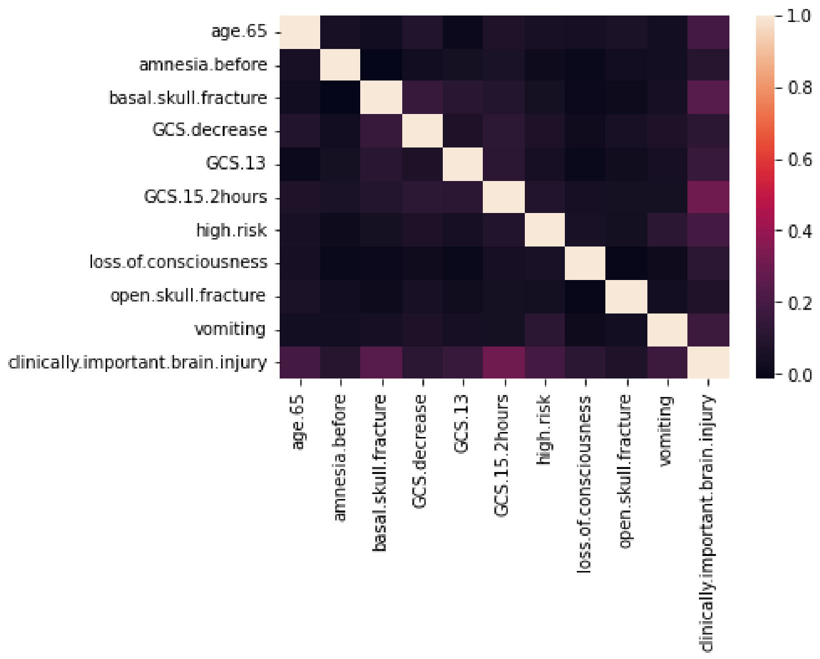
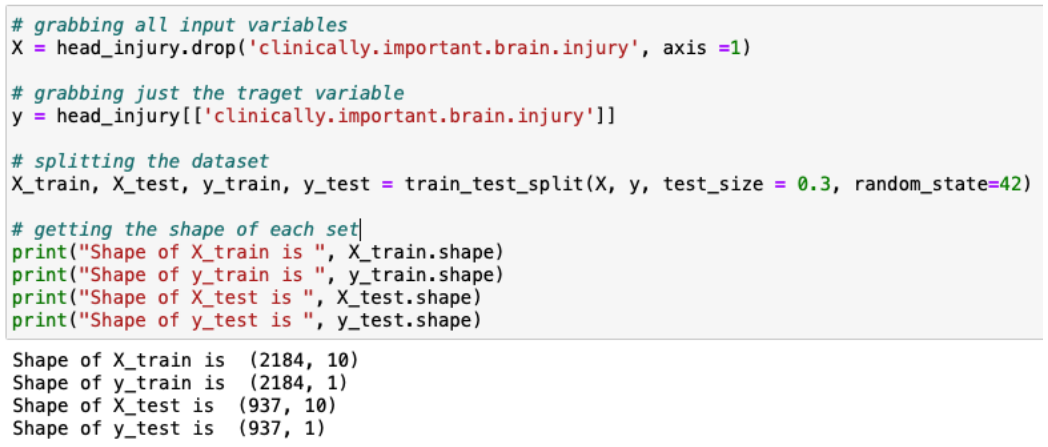


Figure 5: The heatmap of correlation between all variables of the head injury dataset

The heatmap indicates that there are very weak correlations between input variables themselves, and that the variable “GCS.15.2hours” actually have the strongest correlation to the target variable with 0.310139, and the variable “open.skull.fracture” has the weakest correlation to the target variable with 0.073860.

Data Splitting/Partitioning

The dataset is split with two-way partition: training set, which is used to develop the model, and the test set, which is used to evaluate the performance of the model. I used the recommended proportions, which is 70% for training and 30% for test, and I set the “random_state” to 42 so I can get consistent results every time I run my Jupyter Notebook.



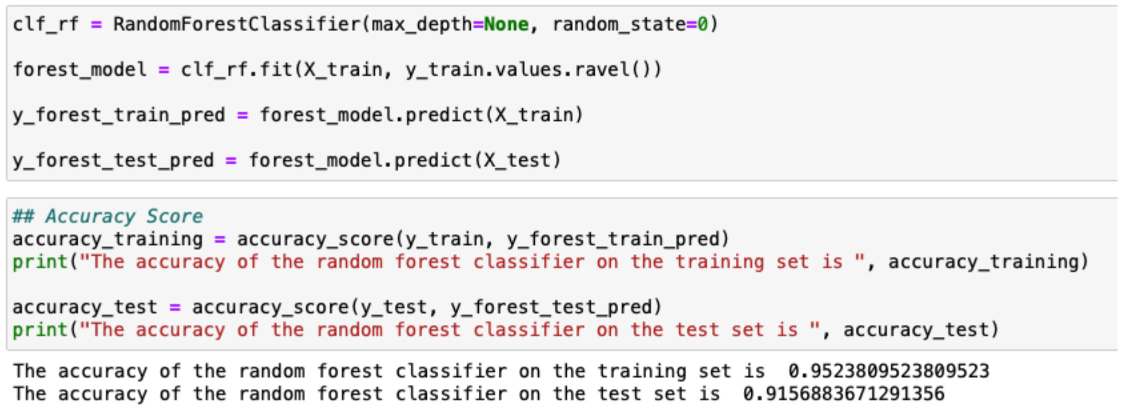
```
# grabbing all input variables
X = head_injury.drop('clinically.important.brain.injury', axis =1)

# grabbing just the traget variable
y = head_injury[['clinically.important.brain.injury']]

# splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state=42)

# getting the shape of each set
print("Shape of X_train is ", X_train.shape)
print("Shape of y_train is ", y_train.shape)
print("Shape of X_test is ", X_test.shape)
print("Shape of y_test is ", y_test.shape)
```

```
Shape of X_train is  (2184, 10)
Shape of y_train is  (2184, 1)
Shape of X_test is  (937, 10)
Shape of y_test is  (937, 1)
```

Figure 6: code snippet for data splitting/partitioning

Model Building, Selection, and Evaluation

Built-in packages from scikit-learn were used to build the four classification models: Random Forest Classifier, K-Nearest Neighbors Classifier, Naïves Bayes, and Logistic Regression. For the Naïves Bayes model, Bernoulli was selected since the predictors are Boolean variables. The four mentioned algorithms were selected because they are designed to classify the instances into the target categories and can work with categorical and binary input variables. The code snippet for each algorithm follows this structure in general: call the algorithm with default values, fit the algorithm with X_train and y_train values, predict the y_train and y_test values, and calculate the accuracy score of the training and test scores.



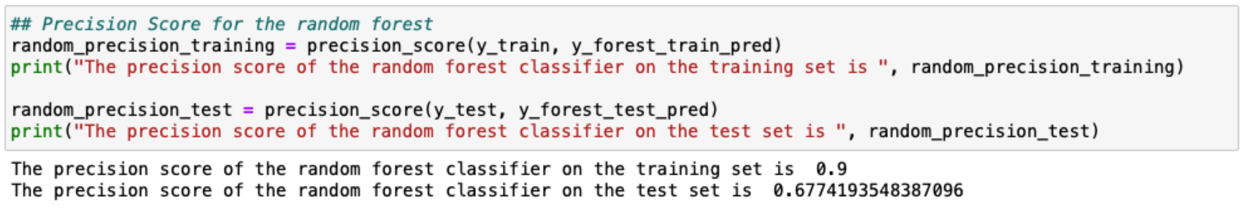
```
clf_rf = RandomForestClassifier(max_depth=None, random_state=0)

forest_model = clf_rf.fit(X_train, y_train.values.ravel())

y_forest_train_pred = forest_model.predict(X_train)

y_forest_test_pred = forest_model.predict(X_test)
```

```
## Accuracy Score
accuracy_training = accuracy_score(y_train, y_forest_train_pred)
print("The accuracy of the random forest classifier on the training set is ", accuracy_training)

accuracy_test = accuracy_score(y_test, y_forest_test_pred)
print("The accuracy of the random forest classifier on the test set is ", accuracy_test)
```

```
The accuracy of the random forest classifier on the training set is  0.9523809523809523
The accuracy of the random forest classifier on the test set is  0.9156883671291356
```

Figure 7: Code snippet for Random Forest Classifier's algorithm set-up and accuracy.

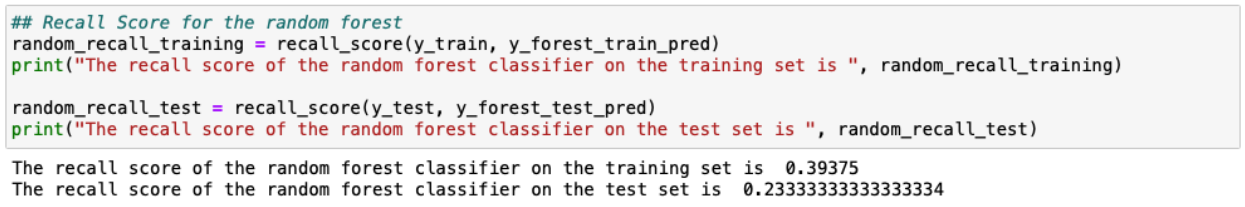
Because the head injury dataset is set in the medical world, the number of false negatives is more costly; this means that the patients are not getting the treatment that they need. I decided to include both precision, which is a measurement of true positives out of all the positive predicted, and recall, which is the true positive rate. We want to have high precision and recall. However, because we don't want to miss any patient, recall is more important than precision.



```
## Precision Score for the random forest
random_precision_training = precision_score(y_train, y_forest_train_pred)
print("The precision score of the random forest classifier on the training set is ", random_precision_training)

random_precision_test = precision_score(y_test, y_forest_test_pred)
print("The precision score of the random forest classifier on the test set is ", random_precision_test)
```

```
The precision score of the random forest classifier on the training set is  0.9
The precision score of the random forest classifier on the test set is  0.6774193548387096
```

Figure 8: Code snippet for Random Forest Classifier's precision score

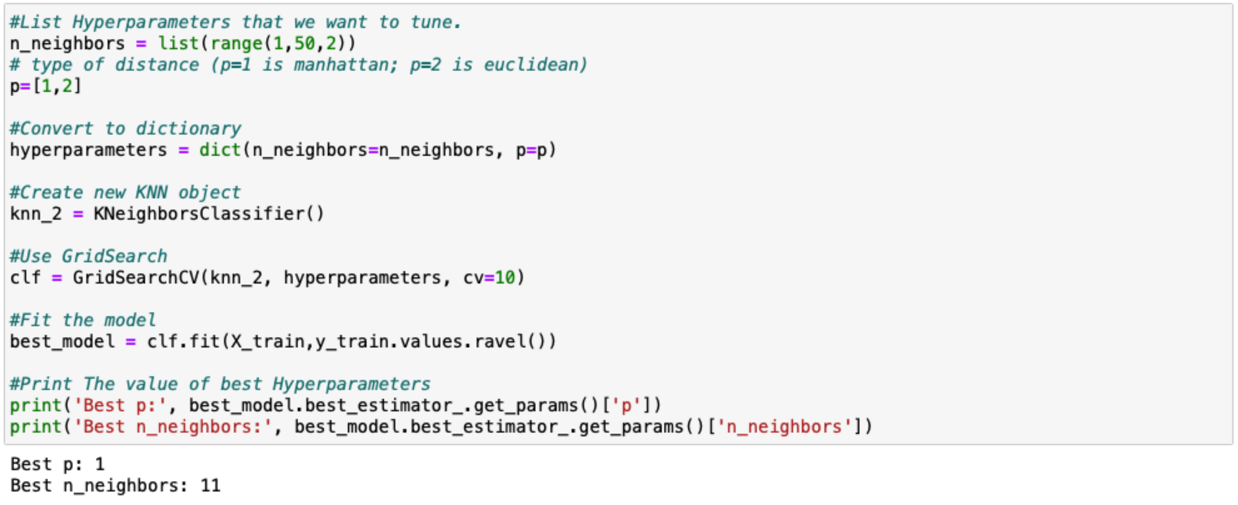


```
## Recall Score for the random forest
random_recall_training = recall_score(y_train, y_forest_train_pred)
print("The recall score of the random forest classifier on the training set is ", random_recall_training)

random_recall_test = recall_score(y_test, y_forest_test_pred)
print("The recall score of the random forest classifier on the test set is ", random_recall_test)
```

```
The recall score of the random forest classifier on the training set is  0.39375
The recall score of the random forest classifier on the test set is  0.23333333333333334
```

Figure 9: Code snippet for Random Forest Classifier's recall score

After the initial set-up of each algorithm, I ran the Grid Search with selected parameters to tune the models. The selected parameters for Random Forest Classifier were "max_depth" with range from 1 to 20 and “max_features” with a list of [0:2, 0:4, 0.6, 0.8]; “n_neighbors” with a range from odd numbers from 1 to 50 and “p” with a list of [1,2] for K-NN (p is the type of distance); “alpha” with a list of [1, 0.1, 0.01, 0.001, 0.001, 0.0001] for Bernoulli Naïve Bayes; and “penalty” = [‘l1’, ‘l2’, ‘elasticnet’] and “C” = np.logspace(-4, 4, 20) for Logistic Regression.



```
#List Hyperparameters that we want to tune.
n_neighbors = list(range(1,50,2))
# type of distance (p=1 is manhattan; p=2 is euclidean)
p=[1,2]

#Convert to dictionary
hyperparameters = dict(n_neighbors=n_neighbors, p=p)

#Create new KNN object
knn_2 = KNeighborsClassifier()

#Use GridSearch
clf = GridSearchCV(knn_2, hyperparameters, cv=10)

#Fit the model
best_model = clf.fit(X_train,y_train.values.ravel())

#Print The value of best Hyperparameters
print('Best p:', best_model.best_estimator_.get_params()['p'])
print('Best n_neighbors:', best_model.best_estimator_.get_params()['n_neighbors'])
```

```
Best p: 1
Best n_neighbors: 11
```

Figure 10: Code snippet for GridSearchCV on K-NN

After each Grid Search, I would use the best parameters to tune the algorithms. I then refit the data to the tuned models to get the accuracy, precision, and recall of these tuned models. Because the Bernoulli Naïves Bayes was already tuned to the best alpha (the default alpha of 1), I didn’t tune the algorithm further. Thus, there are no numbers to report for the tuned Bernoulli Naïves Bayes model.

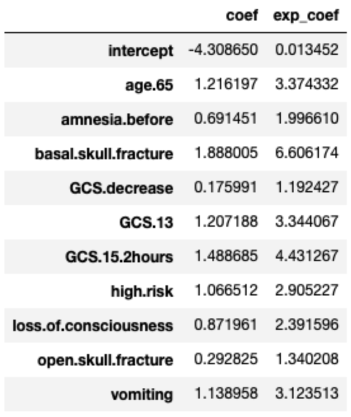
Algorithm	***Accuracy (Training)*** **Accuracy (Test)**	***Recall (Training)*** **Recall (Test)**	***Precision (Training)*** **Recall (Test)**
Random Forest	*0.9524* **0.9157**	*0.3938* **0.2333**	*0.9* **0.6774**
Random Forest, tuned	*0.9455* **0.9200**	*0.2625* **0.2**	*0.9767* **0.8571**
K-NN	*0.9396* **0.9146**	*0.2188* **0.1667**	*0.8333* **0.75**
K-NN, tuned	*0.9341* **0.9104**	*0.1188* **0.0778**	*0.8636* **0.875**

Bernoulli Naïve Bayes	*0.9382* **0.9157**	*0.3188* **0.3667**	*0.6623* **0.6**
Logistic Regression	*0.9391* **0.9200**	*0.25* **0.2778**	*0.7547* **0.7143**
Logistic Regression, tuned	*0.94* **0.9189**	*0.2688* **0.2889**	*0.7544* **0.6842**

Figure 11: Table of accuracy, recall and precision metrics for each algorithm. Italicized numbers are for training sets and bold numbers are for test sets, which are used to determine how well the model generalizes to new examples.

In Figure 11, there are very little overfitting between training and test sets for accuracy metrics for all algorithms (tuned and untuned). Tuned Random Forest Classifier and Logistic Regression are tied for the highest accuracy rate with 92%, and tuned K-NN model ranked the worst with the accuracy rate of 91.04%, which is still high and incredibly close to the best models. The tuned K-NN model fared the worst when it comes to recall with only 0.0778; the Bernoulli Naïves Bayes is the best model for recall with its high score of 0.3667. Precision, which is the least important metrics here, fares the best in the tuned K-NN model with the value of 0.875 and the worst in Bernoulli Bayes with the value of 0.6.

Finally, I extracted the values of coefficients of the logistic regression model to see which input variables have the most impact on the target variable. According to Figure 12 below, the "basal.skull.fracture" variable has the strongest impact on the target variable with the coefficient of 1.88805. This means that if every other factor is held constant and the "basal.skull.fracture" is increased by 1 (e.g., go from 0 to 1 since it is binary), the log odds of being diagnosed with the clinically important brain injury would increase by 6.606174. The input variable with the smallest coefficient is the "open.skull.fracture" with the value of 0.292825.



	coef	exp_coef
intercept	-4.308650	0.013452
age.65	1.216197	3.374332
amnesia.before	0.691451	1.996610
basal.skull.fracture	1.888005	6.606174
GCS.decrease	0.175991	1.192427
GCS.13	1.207188	3.344067
GCS.15.2hours	1.488685	4.431267
high.risk	1.066512	2.905227
loss.of.consciousness	0.871961	2.391596
open.skull.fracture	0.292825	1.340208
vomiting	1.138958	3.123513

Figure 12: Table of the coefficients of the Logistic Regression model. First column is the input variables/index, the second column is the coefficients, and the second column is the exponent of the coefficient.

Conclusion

The best algorithms for predicting the clinically important brain injury based on the accuracy metrics are the tuned Random Forest Classifier and untuned Logistic Regression models with the accuracy rate of 92%. However, the worst accuracy rate of all algorithms was 91.04%, which means that there is very little difference among the algorithms for accuracy. Based on the recall metrics, the Bernoulli Naïve Bayes is the best algorithm. However, I would use a bit of caution since it happens to have the worst precision metrics of all algorithms.

Since the accuracy metrics are really close for all four algorithms, I would recommend using both accuracy and recall in finding the best algorithm. Therefore, the best algorithm would be the untuned Logistic Regression. This is not surprising, since the dataset is based on a logistic regression of clinical data. Finally, the Logistic Regression coefficients yielded insights on which variables have the most impacts on the target variables (highest coefficient/most impact: “basal.skull.fracture”; lowest coefficient/least impact: “open.skull.fracture”).

The biggest lessons learned here are that the accuracy isn’t everything, and be prepared to investigate beyond accuracy, especially if it seems especially high. I recommend using a back-up metrics in case of ties in future cases, and to use untuned Logistic Regression for this dataset.

Citations

Zhu H, Gao Q, Xia X, Xiang J, Yao H, Shao J. Clinically-important brain injury and CT findings in pediatric mild traumatic brain injuries: a prospective study in a Chinese reference hospital. Int J Environ Res Public Health. 2014 Mar 26;11(4):3493-506. Doi: 10.3390/ijerph110403493. PMID: 24675642; PMCID: PMC4025027.

Nigrovic LE, Lee LK, Hoyle J, et al. Prevalence of Clinically Important Traumatic Brain Injuries in Children With Minor Blunt Head Trauma and Isolated Severe Injury Mechanisms. *Arch Pediatr Adolesc Med.*2012;166(4):356–361. Doi:10.1001/archpediatrics.2011.1156

Appendix A

Dataset:

Documentation for Minor Head Injury (Simulated) Data:

https://vincentarelbundock.github.io/Rdatasets/doc/DAAG/headInjury.html

General website for the dataset with csv file and documentation:

https://vincentarelbundock.github.io/Rdatasets/datasets.html

In-depth Descriptions of Variables:

- age.65
  - age factor (0 = under 65, 1 = over 65)
- amnesia.before
  - amnesia before impact (less than 30 minutes = 0, more than 30 minutes = 1)
- basal.skull.fracture
  - (0 = no fracture, 1 = fracture)
- GCS.decrease
  - Glasgow Coma Scale decrease (0 = no deterioration, 1 = deterioration)
- GCS.13
  - Initial Glasgow Coma Scale (0 = '13', 1 = '13')
- GCS.15.2hours
  - Glasgow Coma Scale after 2 hours (0 = not '15', 1 = '15')
- high.risk
  - assessed by clinician as high risk for neurological intervention (0 = not high risk, 1 = high risk)
- loss.of.consciousness
  - (0 = conscious, 1 = loss of consciousness)
- open.skull.fracture
  - (0 = no fracture, 1 = fracture)
- vomiting
  - (0 = no vomiting, 1 = vomiting)
- clinically.important.brain.injury (Note, this is the output variable)
  - any acute brain finding revealed on CT (0 = present, 1 = present)

Notes on Glasgow Coma Scale:

According to the National Library of Medicine, the Glasgow Coma Scale (GCS) is used to objectively describe the extent of impaired consciousness according to three aspects of responsiveness: eye-opening, motor, and verbal responses. The total Coma Score has values between three and 15, three being the worst and 15 being the highest/best. The scale can be used in children older than 5 years with no modification and is as followed:

Best eye response (4)

1. No eye opening
2. Eye opening to pain
3. Eye opening to sound
4. Eyes open spontaneously

Best verbal response (5)

1. No verbal response
2. Incomprehensible sounds
3. Inappropriate words
4. Confused
5. Orientated

Best motor response (6)

1. No motor response
2. Abnormal extension to pain
3. Abnormal flexion to pain
4. Withdrawal from pain
5. Localizing pain
6. Obeys commands

Appendix B

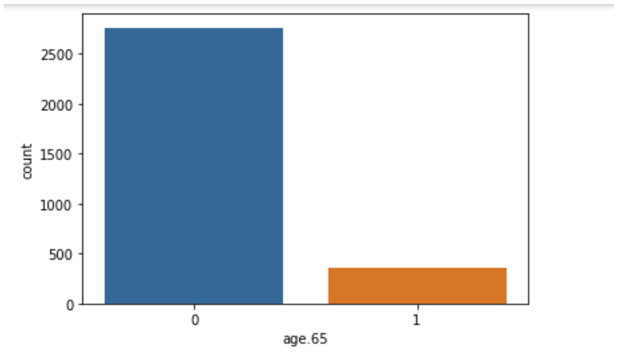


Figure B.1: Countplot of age.65 variable

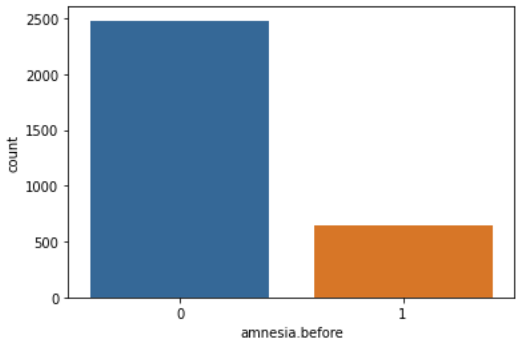


Figure B.2: Countplot of amnesia.before variable

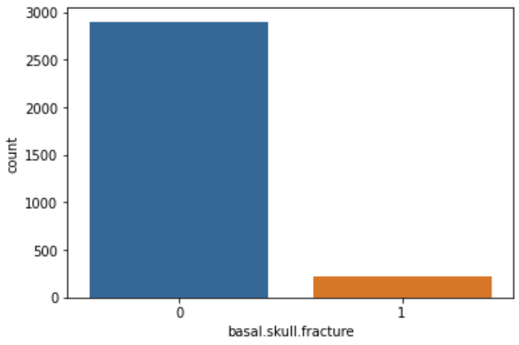


Figure B.3: Countplot of basal.skull.fracture variable

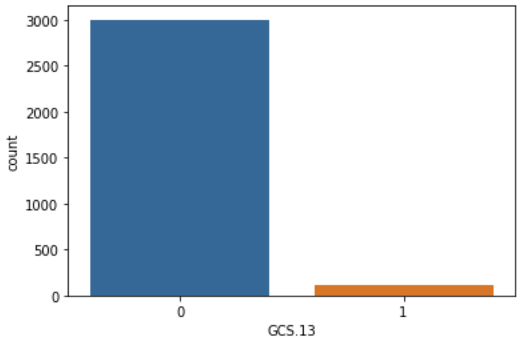


Figure B.4: Countplot of GCS.13 variable

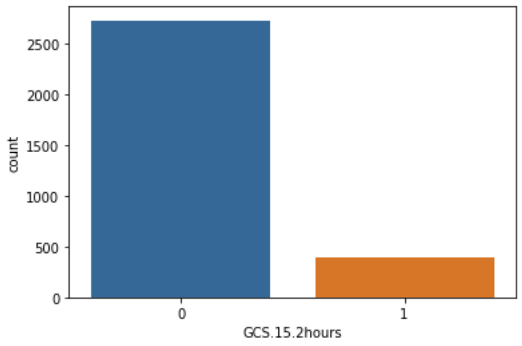


Figure B.6: Countplot of GCS.15.2hours variable

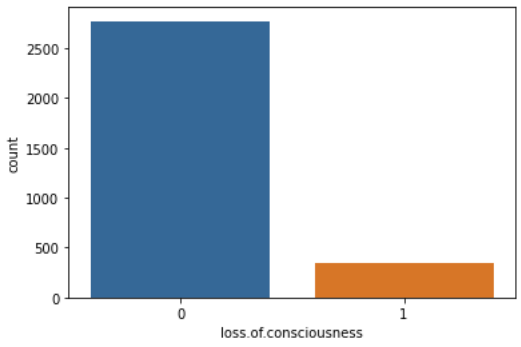


Figure B.7: Countplot of loss.of.consciousness variable

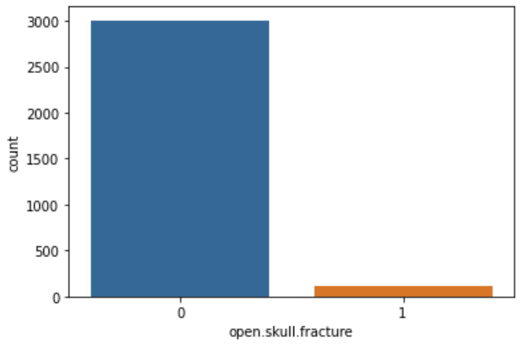


Figure B.8: Countplot of open.skull.fracture variable

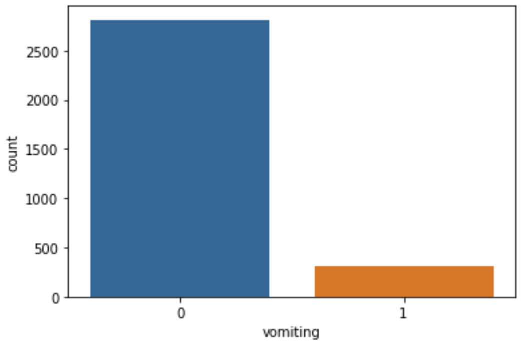


Figure B.9: Countplot of vomiting variable

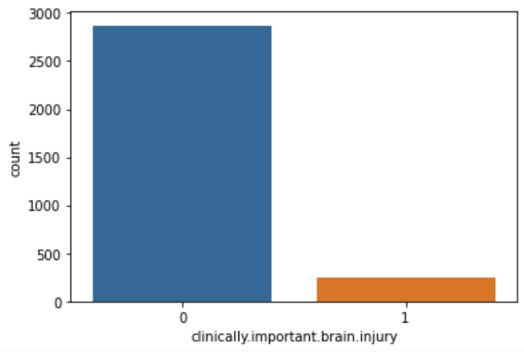


Figure B.10: Coutplot of target variable

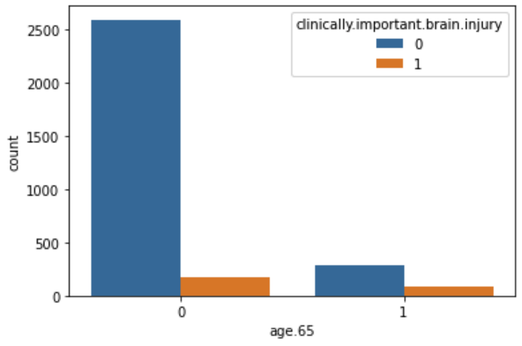


Figure B.11: Grouped bar chart of age.65 vs. target variable

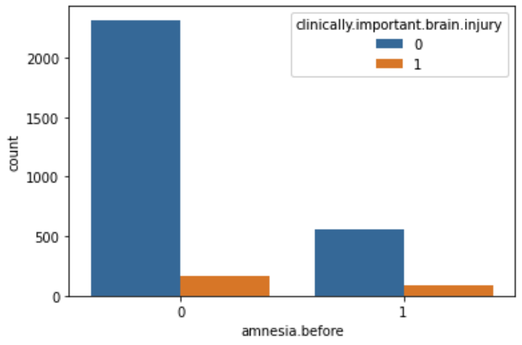


Figure B.12: Grouped bar chart of amnesia.before vs. target variable

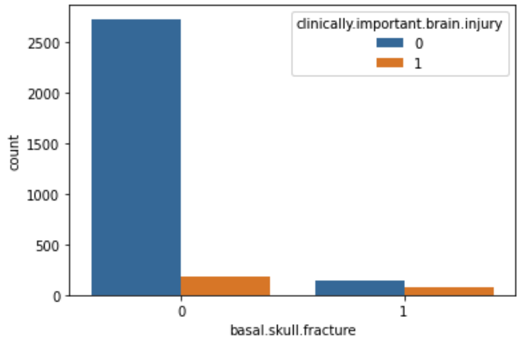


Figure B.13: Grouped bar chart of basal.skull.fracture vs. target variable

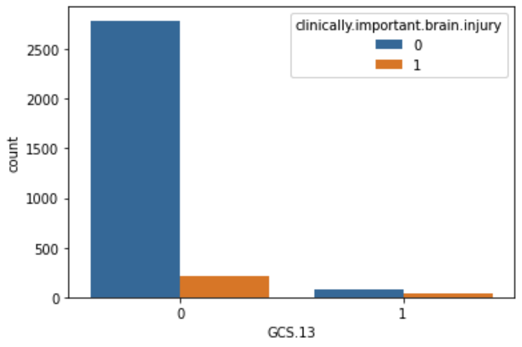


Figure B.14: Grouped bar chart of GCS.13 vs. target variable

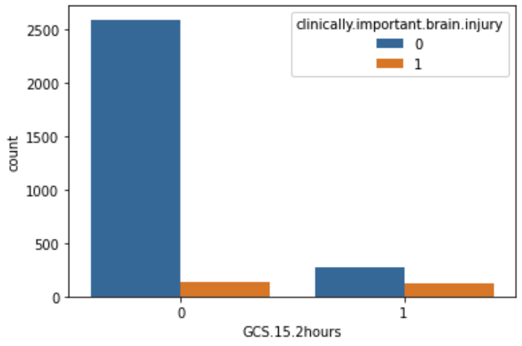


Figure B.15: Grouped bar chart of GCS.15.2hours vs. target variable

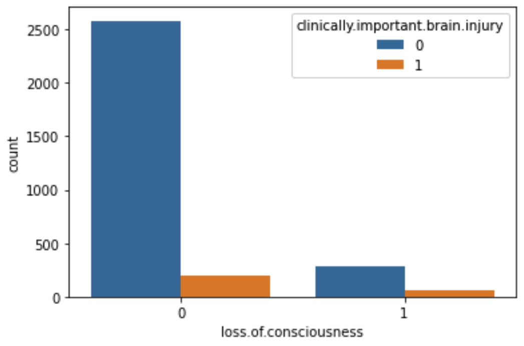


Figure B.16: Grouped bar chart of loss.of.consciousness vs. target variable

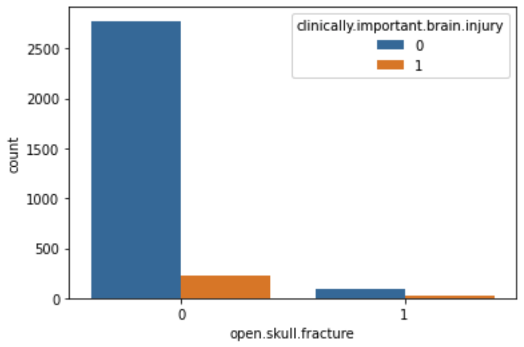


Figure B.17: Grouped bar chart of open.skull.fracture vs. target variable

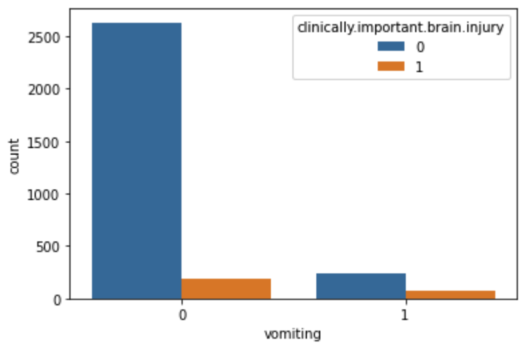


Figure B.18: Grouped bar chart of vomiting vs. target variable

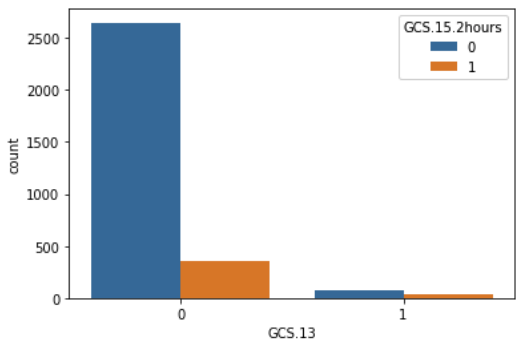


Figure B.19: Grouped bar chart of GCS.13 vs. GCS.15.2hours

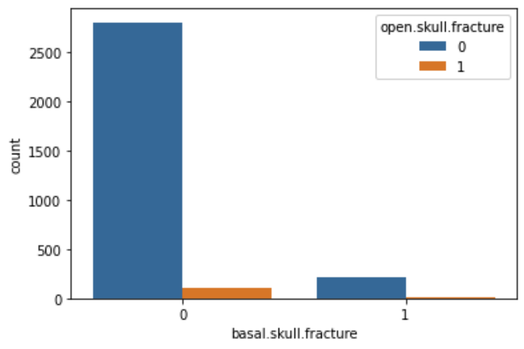


Figure B.20: Grouped bar chart of basal vs. open skull fractures

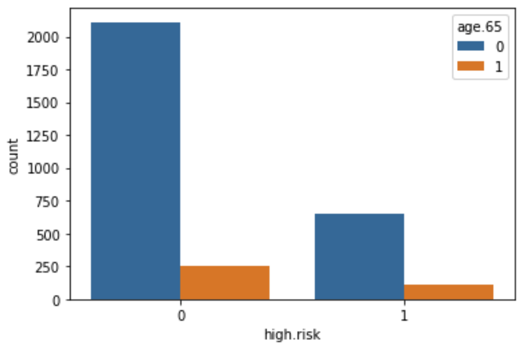


Figure B.21: Grouped bar chart of high.risk vs. age.65