PREDICTIVE ANALYTICS

ALY6020, FALL 2020

CARDIOVASCULAR DISEASE DATA ANALYSIS

CRN: 71717

SUBMITTED BY:

VISHRUTA MAKHIJA (001084592)

EMAIL ID:

makhija.v@northeastern.edu

SUBMITTED TO: PROF. JUSTIN GROSZ

NORTHEASTERN UNIVERSITY, COLLEGE OF PROFESSIONAL STUDIES

PREDICTIVE ANALYTICS

We have used the Cardiovascular disease dataset for our final week's group project. The aim of the project is to clean data and prepare it to further construct predictive models in order to determine whether a person has cardiovascular disease or not. The cardiovascular data that we are working on has 70,000 data points and 13 columns having dependent variable as cardio (which tells if a person has cardiovascular disease or not and is marked as 1 and 0 respectively) and independent variable as age, gender, height, weight, diastolic pressure (ap_lo), systolic pressure (ap_hi) etc.

DATA PREPARATION AND ANALYSIS

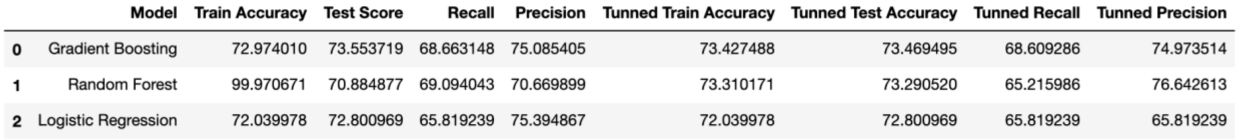
We first performed the data cleaning. We observed that there were no missing values in the data but there were a lot of inaccuracies in the dataset. There were values in the height column that were not practically possible, such as 55 cm and 250 cm. There were mistaken values in the ap_lo and ap_hi columns. Also, there were rows having diastolic pressure greater than the systolic pressure, which is also incorrect. We removed the above-mentioned inaccuracies step by step and obtained clean data having 63321 data points. We then observed the distribution of our dependent variable cardio to be balanced with 50% of the data marked as 0 and 50% marked as 1. Lastly, we changed the unit of age variable from days to years by dividing all the values in the age column by 365 and also removed the Id column as it did not provide any meaningful insights. We also performed a simple exploratory data analysis. From the exploratory data analysis, we observed that the number of female records is less than the male records but the distribution of the males and females for presence of cardiovascular disease is almost the same. We also found that majority of individuals fell into the usual categoric of cholesterol and glucose, which suggests that very few individuals are listed as having above usual levels of cholesterol and glucose.

PREDICTIVE MODELS

We have performed three different predictive models namely Logistic Regression, Random Forest and Gradient Boosting in order to determine the presence of Cardiovascular Disease. We have used the logistic model for our predictive models as it works well when the target variables are binary. Apart from logistic model, tree-based models are considered to be among the best supervised learning algorithms. Tree-based models facilitate high precision, stability and simple interpretation of the predictive models. They are also capable of mapping non-linear relationships very well. They can be modified to solve any problem in hand be it classification or regression.

We first observed the accuracies on the train data set. The Random Forest model gave us the maximum accuracy of 99%, followed by Gradient Boosting and Logistic model with accuracies of 73% and 72% respectively. To further check our model performance, we checked the accuracies on the test dataset and observed that the Gradient Boosting model had the maximum accuracy of 73%, followed by Logistic and Random Forest having 72% and 70% respectively. The difference between train and test accuracy is almost zero for Gradient boosting and Logistic model, but this difference is very large for the Random Forest model which implies that the model is highly overfitted and will not perform well when new data is introduced to it.

As mentioned in last week's report, in addition to running more predictive models, we also suggested to perform hyperparameter tuning to find optimized results for the various parameters of a model, in order to increase the accuracy and reduce the overfitting. We used the GridSearchCV for hyperparameter tuning. GridSearchCV is a feature in the library of model selection of sklearn. It helps to loop through predetermined hyperparameters and match your training set with your model. We can then pick the best parameters from the mentioned list of hyperparameters. GridSearchCV is performed to identify the optimum values for a given model. After getting the optimum values, we then predicted the three models with their best parameters and displayed the metrices to compare them.



	Model	Train Accuracy	Test Score	Recall	Precision	Tunned Train Accuracy	Tunned Test Accuracy	Tunned Recall	Tunned Precision
0	Gradient Boosting	72.974010	73.553719	68.663148	75.085405	73.427488	73.469495	68.609286	74.973514
1	Random Forest	99.970671	70.884877	69.094043	70.669899	73.310171	73.290520	65.215986	76.642613
2	Logistic Regression	72.039978	72.800969	65.819239	75.394867	72.039978	72.800969	65.819239	65.819239

Figure 1: Performance metrices of the three models before and after hyperparameter tuning.

We created a table with all the performance metrices for our three models before and after hyperparameter tuning. From Figure 1, we observed that the accuracy of the Gradient Boosting model and the Logistic model is almost the same before and after hyperparameter tunning, but the accuracy of the Random Forest model changed drastically. The RF model before tuning was highly over fitted but after tunning it is no longer overfitted.

CONCLUSION

Gradient Boosting algorithm is the most commonly used machine learning algorithm and it provides accuracies that are hard to beat. Unlike Random Forest, Gradient Boosting is an ensemble technique that are built on weak and shallow successive trees where each tree learns from the previous tree and improves whereas Random Forest are built on deep autonomous trees. The Gradient boosting sequentially adds predictors to the ensemble and utilizes the sequencing to fix the preceding predictors to conclude an exact predictor at the end of the method. Gradient boosting attempts to fit a new predictor into the residual error made by the previous predictor by tuning the weights for each inaccurate observation in each iteration. Out of all the three models, Gradient booting can be considered best model for predicting the presence of cardiovascular disease as boosting is an ensemble model that comes with easy to read and understandable algorithms, making it easy to handle the interpretations of the predictions. Also, it is a robust mechanism that restricts over fitting rapidly.

REFERENCES

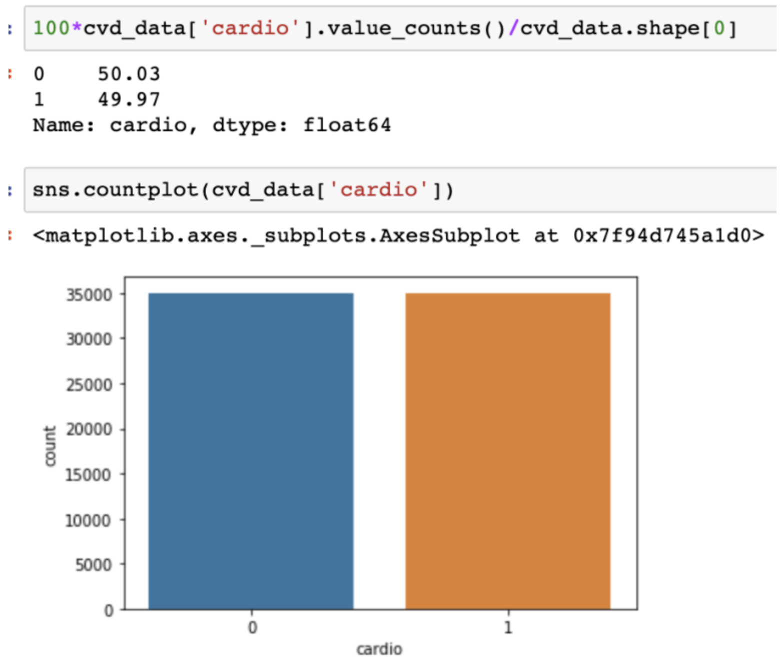
Gradient Boosting Machines · UC Business Analytics R Programming Guide. (2020). Retrieved 22 October 2020, from http://uc-r.github.io/gbm_regression

Python), T. (2020). Tree Based Algorithms | Implementation In Python & R. Retrieved 22 October 2020, from https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/

Accuracy, Precision, Recall or F1?. (2020). Retrieved 22 October 2020, from https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

Boosting - Overview, Forms, Pros and Cons, Option Trees. (2020). Retrieved 22 October 2020, from https://corporatefinanceinstitute.com/resources/knowledge/other/boosting/

APPENDIX



```
100*cvd_data['cardio'].value_counts()/cvd_data.shape[0]
```

```
0    50.03
1    49.97
Name: cardio, dtype: float64
```

```
sns.countplot(cvd_data['cardio'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f94d745a1d0>
```

Figure 1: Distribution of the dependent variable to check for imbalanced data.

OUTLIER TREATMENT

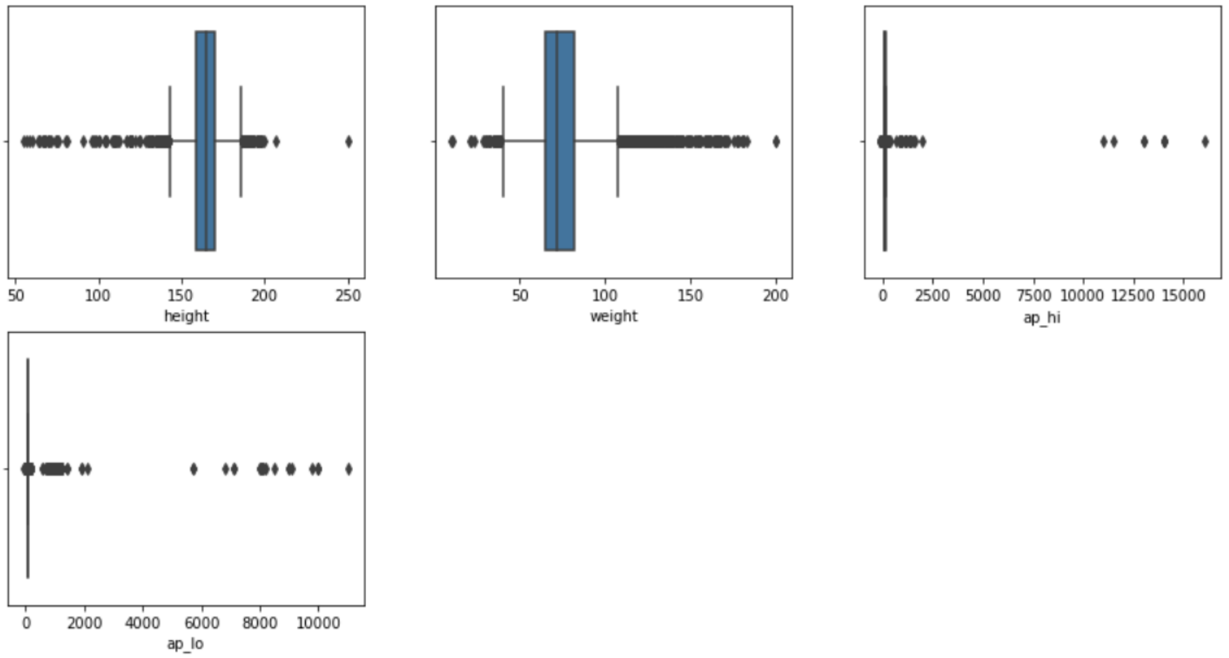


Figure 2: Outlier Detection in the continuous variable columns.

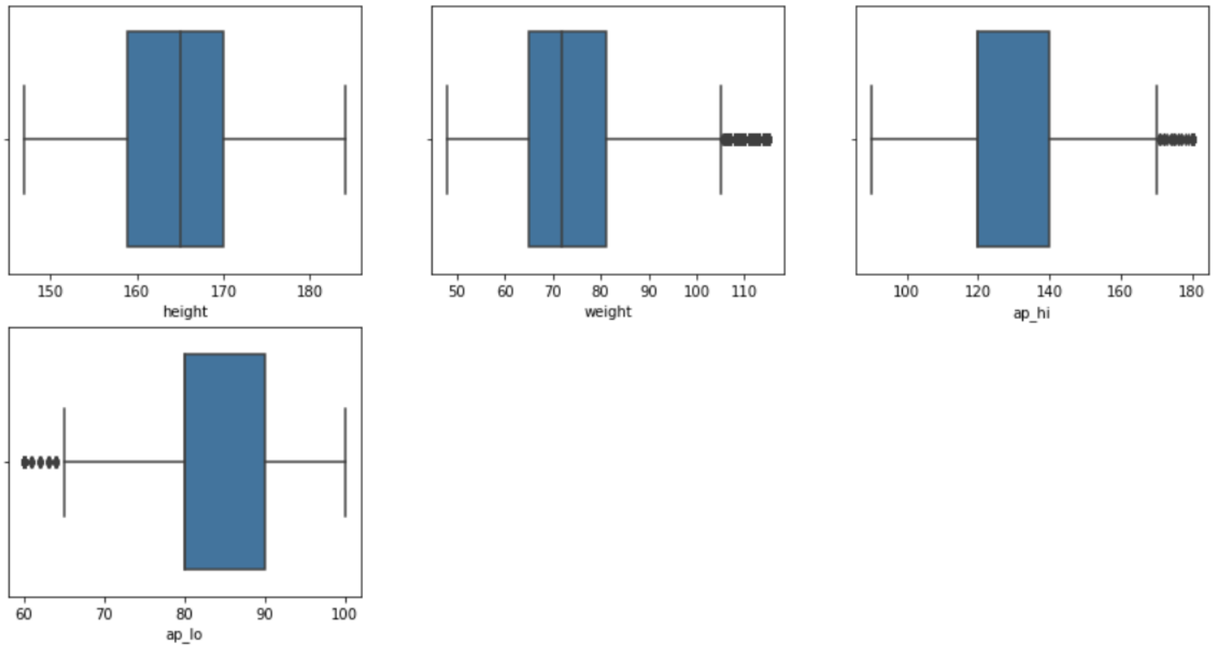
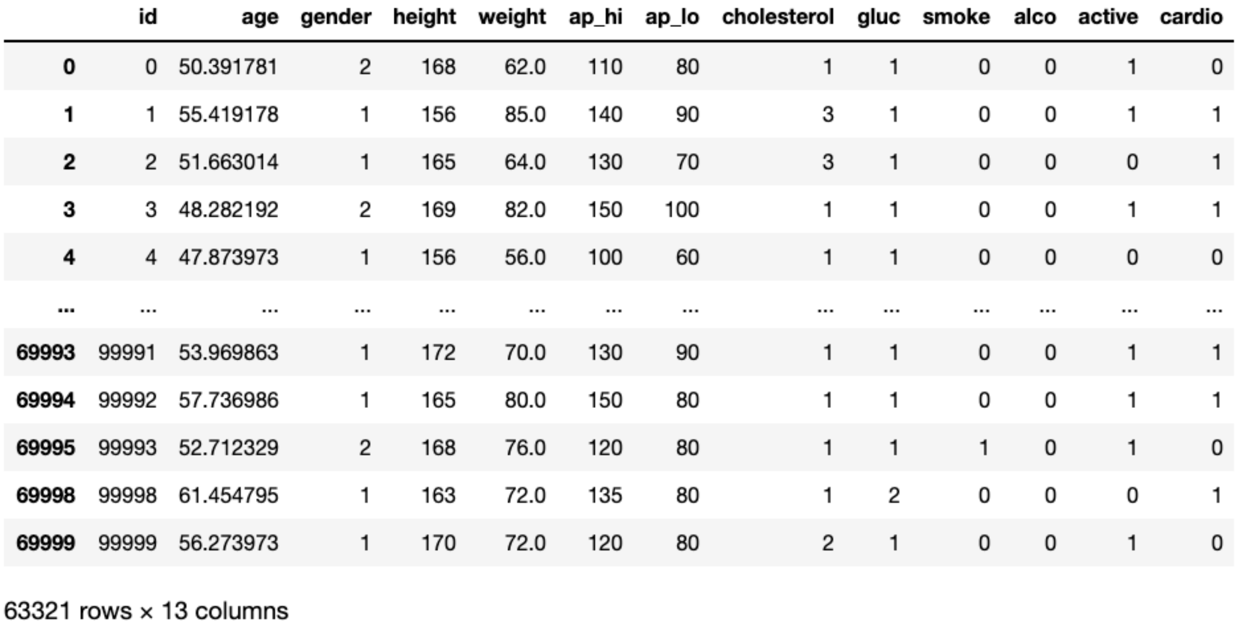


Figure 3: Outlier detection through Boxplots after removing all the outliers.



	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
0	0	50.391781	2	168	62.0	110	80	1	1	0	0	1	0
1	1	55.419178	1	156	85.0	140	90	3	1	0	0	1	1
2	2	51.663014	1	165	64.0	130	70	3	1	0	0	0	1
3	3	48.282192	2	169	82.0	150	100	1	1	0	0	1	1
4	4	47.873973	1	156	56.0	100	60	1	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
69993	99991	53.969863	1	172	70.0	130	90	1	1	0	0	1	1
69994	99992	57.736986	1	165	80.0	150	80	1	1	0	0	1	1
69995	99993	52.712329	2	168	76.0	120	80	1	1	1	0	1	0
69998	99998	61.454795	1	163	72.0	135	80	1	2	0	0	0	1
69999	99999	56.273973	1	170	72.0	120	80	2	1	0	0	1	0

63321 rows × 13 columns

Figure 4: Age column having values in years.

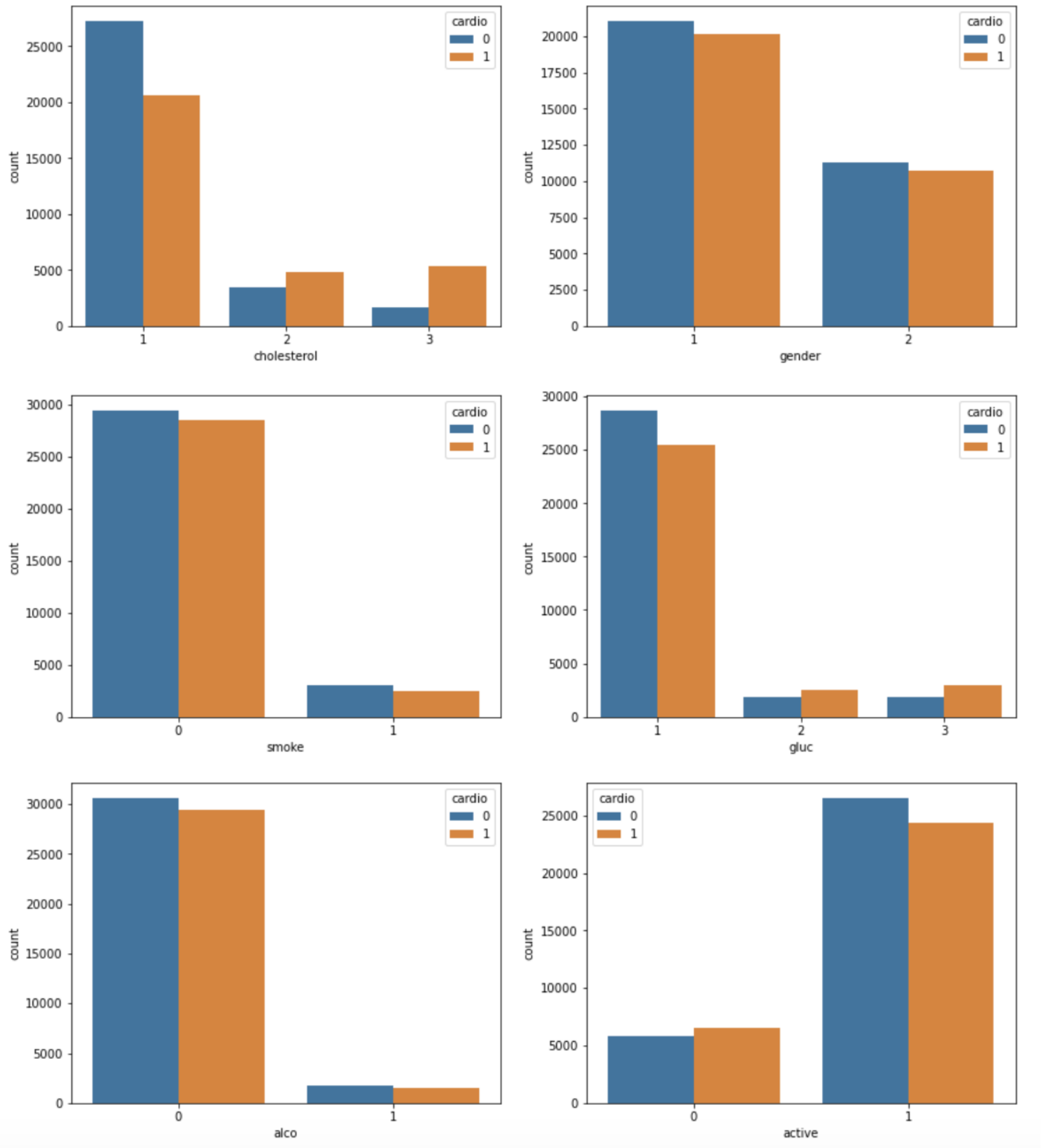


Figure 5: Exploratory data analysis representing the distribution of different variables with respect to cardio feature.