Can A Patient’s Health Metrics Predict Their Smoker Status?

By: Emerie Danson, Sam Mathieu, Eve Dean

Image credit to Ralf Kunze (Pixabay)

When patients visit the doctor’s office, one question they may be asked is whether or not they smoke. Smoking can increase the risk of lung and cardiovascular diseases. A dataset from the South Korean government, hosted on Kaggle, contains 26 columns of health metrics for 55,692 people, including their smoker status. Other variables provided include factors such as height, weight, cholesterol, and blood pressure. We aimed to use the health metrics in this dataset to predict the smoker status of each patient. By visualizing the dataset and building predictive models, we can better understand the relationship between a person’s smoker status and different aspects of their health.

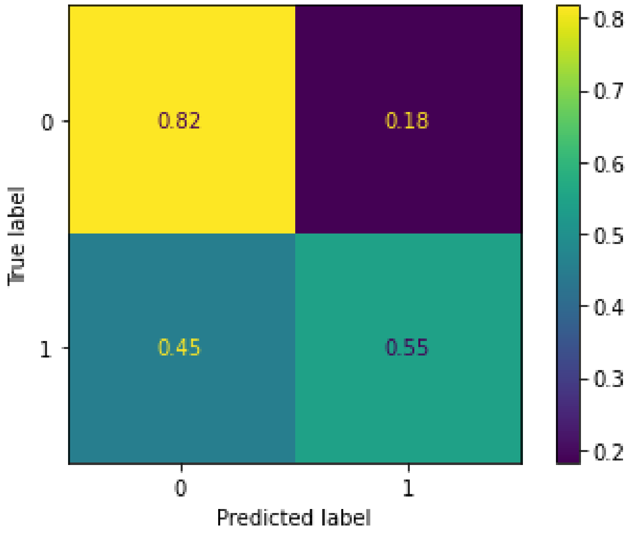
Data exploration

An initial look at our data showed that only 37% of the individuals in the dataset were smokers. Additionally men made up 64% of the dataset, and 45 year olds were significantly overrepresented. All the continuous variables were normally distributed. Since the higher proportion of non-smokers in the dataset could cause the models to predict that patients are nonsmokers at an inappropriately high rate, we balanced the data so that it included an equal amount of smokers and non-smokers. Though we did not use this balanced dataset in our initial models, we did use it in the final version of the models used for ensembling.

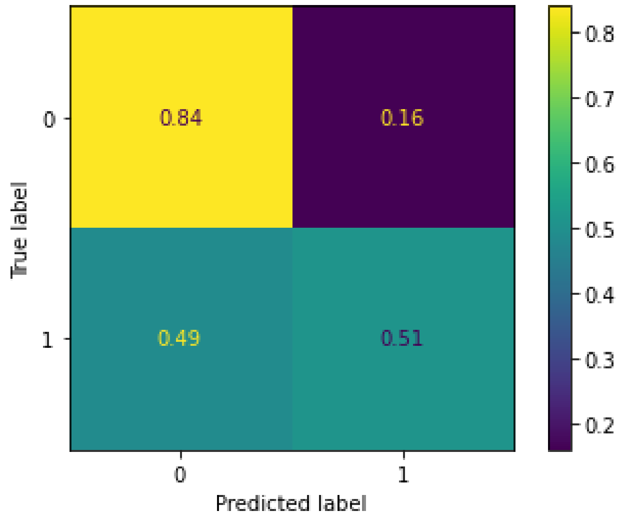
Predictive models

One of the first steps in building a decent predictive model is to split the data into training and testing sets. If a model receives every entry in the dataset at once to learn from, there is a risk of 'overfitting' to the data. The model may rely on 'memorizing' the correct outcomes for the initial data, and would be ineffective at making predictions for any new data received. We randomly pulled 70% of the rows from our dataset to use as our training set, and the remaining 30% of the rows would serve as a testing set. This way, models have a lot of the data available to learn from, but we can have some entries left over to get unbiased predictions for, and ensure our models can make accurate predictions.

The first type of model we built was a logistic regression model. This was a baseline model using all the physiological data in the dataset, and it was unbalanced. For the initial logistic regression model, we got a decent starting accuracy of 0.72. However, the true positive rate for predicting smoker status was 0.55 which was fairly low. This left room for implementing new variables into our model so that we may increase the true positive rate.

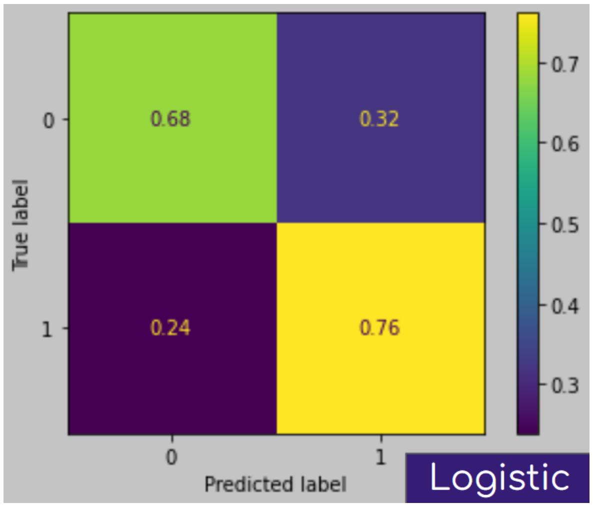


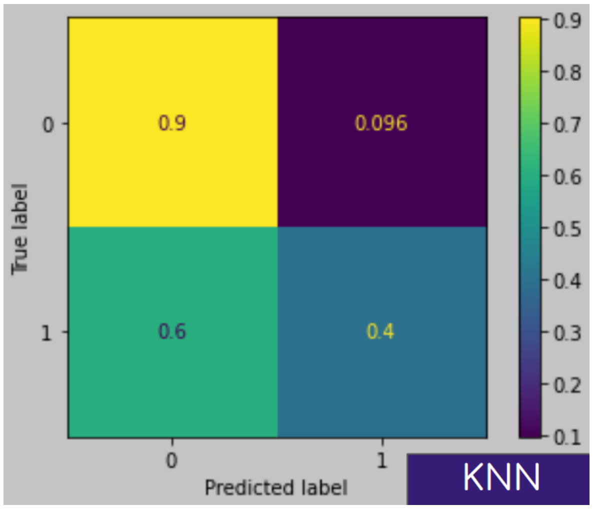
Next, we built a k nearest neighbors classifier model. Similarly to the initial logistic regression model, the initial KNN model's accuracy was decent at 0.72 but had a fairly low TPR of 0.51.



To see if we could increase the TPR of both of these models, we used feature engineering to make new, informative features from the existing columns. These new columns included BMI (using the height and weight columns), UPC ratio (using the urine protein and urine creatinine columns), HDL, and LDL cholesterol because they are important health variables relating to smoking. When looking at these new columns, the majority of the dataset fell within a normal or “healthy” range.

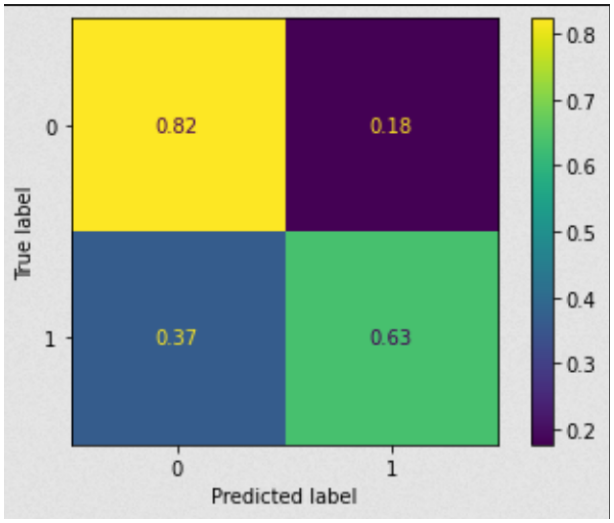
After adding these variables to the models, TPR of the logistic model increased to 0.76 but the TPR for the KNN model decreased to 0.40.





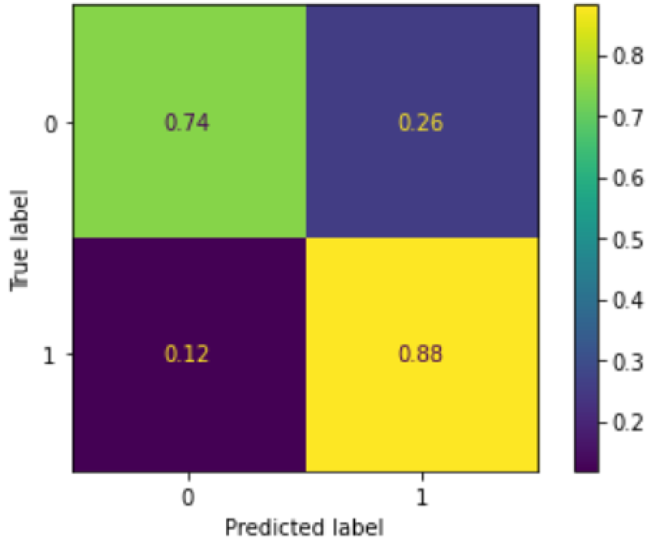
We then looked at the top coefficients for the logistic regression to see what variables have the greatest impact on predicting if someone was a smoker or not. We used a cutoff of -0.05 and +0.05. This left us with the variables hemoglobin, gender, UPC ratio, tartar, and dental caries. By only including these variables, this increased the TPR for the logistic regression to 0.96 but decreased the TPR for the nearest neighbors to 0.39, so we experienced another tradeoff.

Finally, we built a decision tree model. This kind of model works by ‘asking questions’ about the data to split it up into different groups, and making predictions based on these. At first, decision trees were created with train test split data, but without balancing or the additional four variables from feature engineering. Decision trees have several hyperparameters that require finetuning. Max depth is the number of questions that can be asked. Min sample split is how many rows are required for the model to ask the next question. Min sample leaf is how many ‘bins’ the data should be grouped in at the end. The hyperparameters for the most accurate tree from this initial build were max depth of 25, min sample split of 2, and min sample leaf of 2. The tree had an accuracy score of .75, and was better at predicting true negatives (nonsmokers) than true positives (smokers).



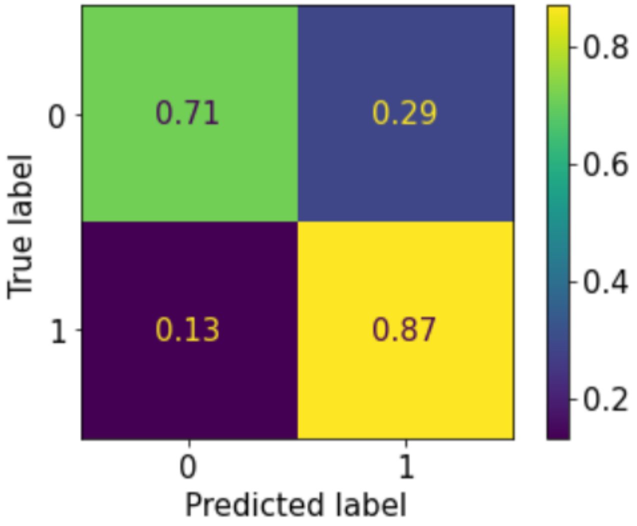
Decision trees were generated again using balanced data and the new features. The ID variable was also dropped, since this is essentially the index number of the patient and should not be used to make predictions. The best hyperparameters were max depth of 10, min sample split of 50, and min sample leaf of 10, with an accuracy of 0.71. This model was better at predicting true positives. The accuracy likely decreased due to the model no longer having access to patient IDs.

Gradient boosting is a method to generate many decision trees that learn from the previous trees’ mistakes. Two hyperparameters were fine tuned. Number of estimators is how many trees should be made, which should be high enough to make a lot of trees, but a low enough amount for the program to take a reasonable amount of time to run. We used 200 estimators. The other hyperparameter is learning rate, which determines how drastically each tree changes from the previous one. Setting this too low will result in the model not ‘learning enough’, but too high may cause it to ‘overshoot’ the most accurate configuration. The best learning rate was found to be 0.075. A max depth of 10 was used based on the previous decision tree tuning. The gradient boosting model had an accuracy score of 0.774, and improved the score of the true negative rate without sacrificing the high score of the true positive rate.



Ensembling

To combine our model types and increase the accuracy and TPR of our model, we used ensembling on the logistic regression, nearest neighbors model, and gradient boosted decision trees. We used a voting classifier with soft voting to make predictions. The soft voting method predicts the class using the argmax of the sums of the predicted probabilities. Soft voting is useful when combining well calibrated classifiers, which is what we used in this model. The soft voting classifier had an overall accuracy of 0.74, and AUROC score of 0.79, and the confusion matrix shows the model was 0.87 effective at predicting true smokers.



Conclusion

Moving through several iterations of these models allowed us to make better predictions by balancing the data, fine tuning parameters for each model, and including informative features. The most important factors as decided by our models were hemoglobin, gender, and UPC ratio, which further highlights the value of including informative features in these models. Although the soft voting classifier was more effective than just the logistic regression and nearest neighbors models on their own, the gradient boosted decision tree was ultimately the best of the models that we made. If we wanted to further improve these models, we could develop more informative features and split the models into different age groups to account for the interaction of age with health metrics.

References:

https://www.cdc.gov/tobacco/data_statistics/fact_sheets/health_effects/effects_cig_smoking/index.htm