DSC 478 Final Project

Group Members: Matthew Soria, Jack Leniart, Francisco Lozano

Project Proposal

We proposed to create a classification model that would identify smoker's from a kaggle dataset that contains bio signals related to their health.

We chose this problem because we recognized the potential ability to be used in healthcare. One application of our model could be used by epidemiologists - attempting to understand population health trends/information - without directly having access to whether or not a person was a smoker.

DM Tasks

1. Preprocessing
2. Exploratory Data Analysis
3. Create benchmark model
4. Model creation (all team members created one)
    a. Evaluation - choose the best model
5. Create an application for the model

Our Dataset

- Smoker Status Prediction using Bio-Signals (kaggle.com)
  - URL: https://www.kaggle.com/datasets/gauravduttakiit/smoker-status-prediction-using-biosignals/data
- Our dataset, Smoker Status Prediction using Bio-Signals, contains 22 features and 2 classes (smoker not smoker). It's a total of ~ 3 MB.
- Our classification model will aim to predict values for the smoking variable. Our dataset contains 24,666 instances of nonsmokers and 14,318 instances of smoker

Preprocessing

- In our preprocessing step we checked for missing data and any categorical fields that needed to be transformed.
- We normalized the data using scikit-learn’s MinMax scaler.
- We split our data into train/test using an 80/20 split respectively.
- Finally, we saved the data to be used by future steps when applicable.

Exploratory Data Analysis

- We analyzed the data types, descriptive statistics, distributions, and correlations of the variables in the dataset.
- We identified five categorical variables (including our target variable)
- The descriptive statistics showed significant differences in the ranges of values of some of the variables
  - This confirmed our decision to normalize (scale) the data
- Most of the variables had a normal distribution
- There were some correlations we observed that made sense intuitively
  - Positive correlations between: height and weight, weight and waist, systolic and relaxation (both heart rate measures), cholesterol and LDL, and AST and ALT (both glutamic oxaloacetic transaminase types)
  - Negative correlations between HDL (“good” cholesterol) and: weight, waist, and triglyceride (a type of fat found in the blood)

Classification Models

- Decision Tree (Benchmark)
- Xgboost
- K-Nearest Neighbors
- SVM
- Combination of models

Decision Tree (Benchmark Model)

- We created this benchmark model to compare our other models to in order to see if we were improving classification accuracy.
- We started by creating a Decision Tree model with no parameters (using the defaults), but that lead to overfitting.
- To combat overfitting, we then used gridsearch to find the best parameters for the Decision Tree.
- The resulting model had a train/test accuracy of 73%.

Xgboost (created by Matt)

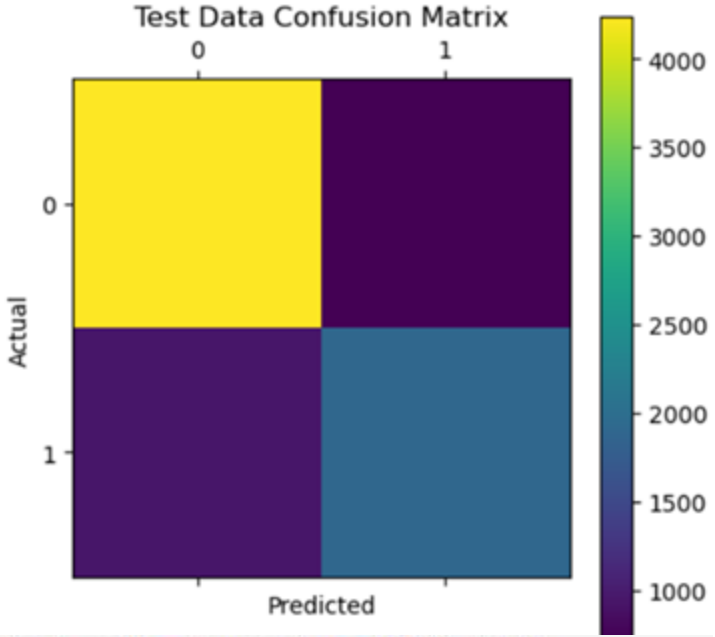
- I started with no parameters and got 87% and 77% accuracy for train and test data respectively.
- However, when looking at feature importance, the height parameter was by far the most important.
- I removed height and used GridSearch to find the best parameters.
  - This lead to a model with 83%/76% accuracy.
- Finally, I attempted feature selection and found that hearing (right), hearing(left), and urine protein could be removed.
  - This lead to a model with 85%/77% accuracy, which was the best accuracy recorded.
- Feature selection and gridsearch lead to the best model for Xgboost.

K-Nearest Neighbors (created by Jack)

- First, I created a baseline model using the default parameters
  - It had a 72% accuracy when ran against the test data
- Then, I used GridSearch to identify the best parameters for a KNN model
  - The best parameters were: n_neighbors = 82 , weights = “distance”
- I ran my own cross validation on the training data using all of the same possible combinations of parameters
  - That confirmed that the best parameters were the ones identified by GridSearch
  - The model with the best parameters had a 77.5% accuracy in cross-validation
- Next, I tried feature reduction with Principal Component Analysis
  - The first 8 components captured 91.6% of the variance in the dataset
- I created another KNN model using the best parameters and fit the model to the transformed training dataset
  - This model had a 70% accuracy when ran against the transformed test data

Best KNN Model

- I found the best model used parameters:
  - N_neighbors = 82
  - Distance = “weights”
- This model had 79% accuracy when run against the test data
- All of the KNN models tested were better at predicting non-smokers (0) than smokers (1)
  - This is likely because our dataset had much more observations of non-smokers

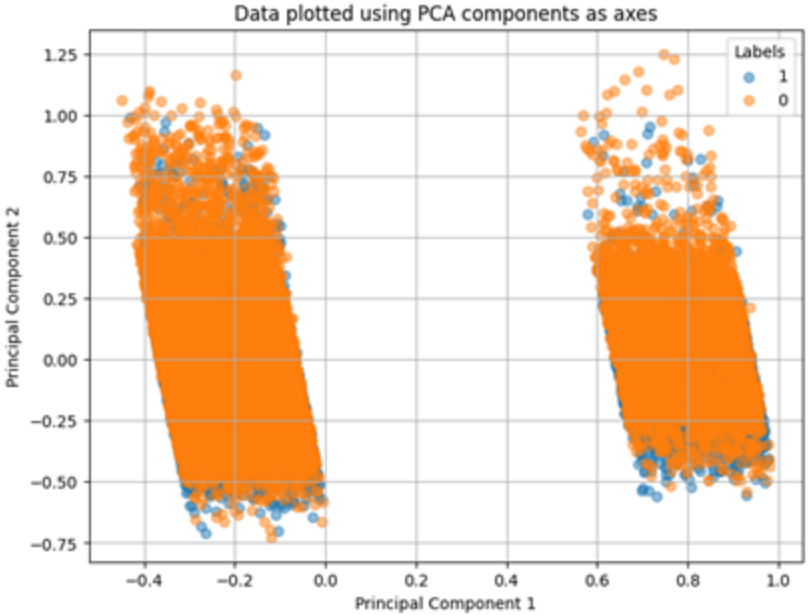



```
best_test_preds = knn_best.predict(X_test)
print(classification_report(y_test, best_test_preds))
```

	precision	recall	f1-score	support
0	0.82	0.85	0.84	4960
1	0.72	0.67	0.70	2837
accuracy			0.79	7797
macro avg	0.77	0.76	0.77	7797
weighted avg	0.79	0.79	0.79	7797

SVM (created by Francisco)

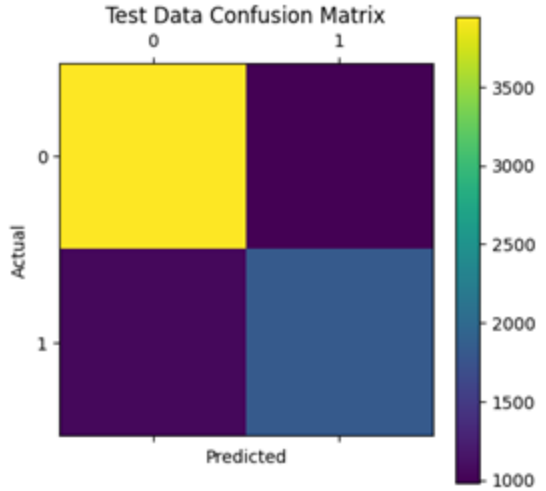
- Linear SVM
  - baseline Linear SVM, yielded a validation accuracy of 73%, comparable to our benchmark model.
  - attempts at feature reduction led to a decrease in accuracy, prompting exploration of alternative kernels.
- RBF SVM
  - promising initial results with a baseline accuracy
  - yielded validation accuracy of 74% without signs of overfitting



Best SVM Model

- The RBF SVM model demonstrated superior performance, particularly in predicting non-smokers, evident from the confusion matrix.
  - Training accuracy: 0.757
  - Test accuracy: 0.74

	precision	recall	f1-score	support
non-smoker	0.79	0.80	0.80	4933
smoker	0.65	0.64	0.64	2864
accuracy			0.74	7797
macro avg	0.72	0.72	0.72	7797
weighted avg	0.74	0.74	0.74	7797



Individual Model Evaluation

- All three models (KNN, RBF SVM, XGBoost) outperformed our benchmark model in terms of accuracy.
- KNN exhibited the highest accuracy, while XGBoost showed slight overfitting.
- The SVM and XGBoost models highlighted distinct feature importance patterns.

Model	Train Score	Test Score
Decision Tree (Benchmark)	0.729	0.73
KNN	0.775	0.79
RBF SVM	0.758	0.74
XGBoost	0.857	0.772

Combination of Models (XGBoost, KNN, SVM)

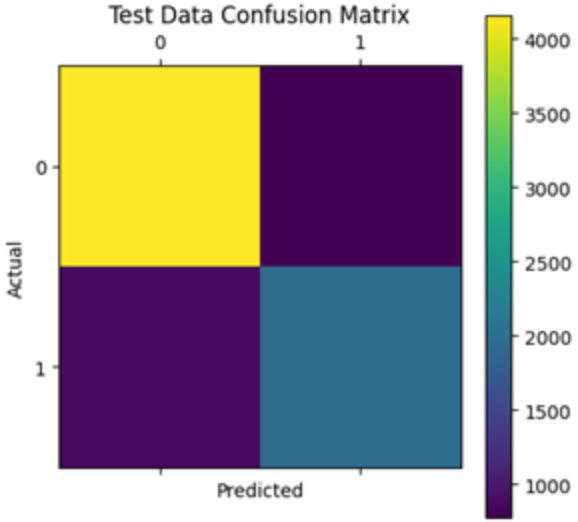
- Recognizing the diverse pattern capturing abilities of individual models
- Two main approaches were considered: Voting Classifier and Stacking Classifier.
- Testing various configurations, we found the Soft Voting Classifier to be the most effective, achieving high accuracy without overfitting.

Model	Train Score	Validation Score
Hard Voting Classifier	0.776	0.788
Soft Voting Classifier	0.783	0.796
Stacking Classifier	0.779	0.794

Final Model Evaluation

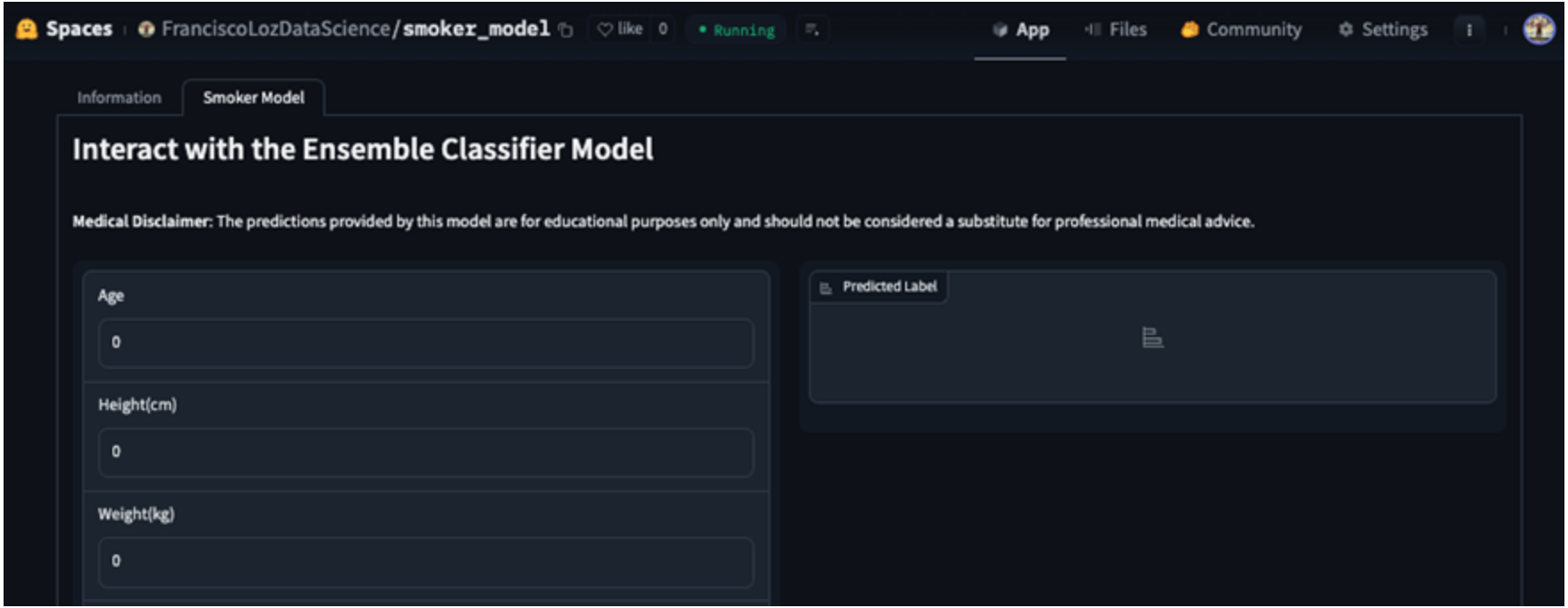
- Upon creating the ensemble classifier, we re-evaluated all models.
- The Soft Voting Classifier continued to outperform individual models, exhibiting superior accuracy and robustness.
- Confusion matrix analysis further supported the effectiveness of the ensemble model, especially in predicting non-smokers.

Model	Train Score	Test Score
Decision Tree (Benchmark)	0.729	0.73
KNN	0.775	0.79
RBF SVM	0.758	0.74
XGBoost	0.857	0.772
Soft Voting Classifier	0.783	0.796



Application

- https://huggingface.co/spaces/FranciscoLozDataScience/smoker_model



Sources/Citations

- The dataset we used for this project was from kaggle.com
  - https://www.kaggle.com/datasets/gauravduttakiit/smoker-status-prediction-using-biosignals/data
- Our project features many functions from the scikit-learn library
  - https://scikit-learn.org/stable/user_guide.html
- We learned about the Profile Report from the ydata_profiling library
  - https://pypi.org/project/ydata-profiling/