CS205 - Healthcare Analytics: Final Report

Allen Nikka
Department of Computer Science
University of California, Los Angeles
Los Angeles, CA 90095
allennikka7@g.ucla.edu

Abstract

Cancer is the second leading cause of death globally, and is responsible for an estimated 9.6 million deaths in 2018, with enormous economic impacts in addition to the human costs it incurs[1]. In this paper, I present a novel framework for pre-processing and analyzing survey data in order to predict cancer or malignancy in a given patient. The model proposed uses a combination of imputation, tree and variance based feature selection methodologies, and random forest based machine learning models to identify the 19 most important features in predicting cancer or malignancy, and achieves accuracy of 84.4% on data supplied by the CDC [2019a] NHANES survey.

1 Feature Selection and Evaluation

1.1 Options for feature selection

Given the task of building a model to predict positive test results for cancer or malignancy, several options were initially considered. The first decision concerned how to select the features that the machine learning algorithm (or ensemble of algorithms) would use. These options, broadly speaking, were the following:

1. Use various feature analysis tools, such as threshold measures, statistical dependency measures, and L-based feature selection in a batched manner on the entire dataset.
2. Consult outside expert resources, select relevant features manually, and then perform the feature selection tools mentioned in option (1) above to prune this feature set.

While the first option initially seemed more sound, input from more experienced practitioners in the art, research on industry standard methodologies indicating the potentially negative impact of irrelevant features[2], and the practical fact that the resulting data frames would be totally unmanageable made option two a more reasonable choice and thus it was selected.

1.2 Background research

The next step was collecting relevant, reliable information to help inform the selection of features from the CDC [2019a] NHANES database. The first step undertaken was to use the CDC [2019b] USCS (United States Cancer Statistics) database to determine which factors are likely relevant to positive diagnoses. The following observations were made based on the data visualizations provided by this database to find features demonstrating high variance conditioned on the target, cancer:

[1] World Health Organization, Cancer Fact Sheet: https://www.who.int/news-room/fact-sheets/detail/cancer

[2] Raheel Shaikh, *Feature Selection Techniques in Machine Learning with Python*: https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

Risk List A:

1. Geographic location
2. Race/ethnicity
3. Age
4. Combined sex and race
5. Socioeconomic status

In order to further expand the list of relevant features, another expert data source was consulted. In this case, the National Institutes of Health's (NIH) National Cancer Institue's published list of risk factors for cancer[3]. According to the information there, the following are all environmental and health-related features that influence cancer diagnoses:

Risk List B:

1. Age
2. Alcohol
3. Cancer-Causing Substances[4]
4. Chronic Inflammation
5. Diet
6. Hormones
7. Immunosuppression
8. Infectious Agents
9. Obesity
10. Radiation
11. Sunlight
12. Tobacco

Clearly, the above list has some overlap with the first but concerns itself with more direct cancer-causing agents than the CDC USCS GIS database's indirect agents. This might imply that there might be interdependence (i.e. higher covariance when conditioned on our target feature: cancer), which can be elucidated using the aforementioned feature selection techniques.

1.3 Feature Selection

Given the results of the background research (above), the next logical step involved selecting features from the NHANES database that were reflective of the categories outlined therein. The NHANES database is separated into Demographics Data, Dietary Data, Examination Data, Laboratory Data, and Limited Access Data, each of which likely contains data relevant to one more of the relevant categories. Thus in determining which data to select, the first 5 subcategories (as limited access data use is restricted) were examined, with the aim of selecting between 50 and 70 total features that are each clearly related to one of the aforementioned cancer risk factors. These will then be pared down to 25 to 35 features based on the feature selection methodologies subsequently employed.

Based on the aforementioned considerations, the following features were selected from each of the five categories:

1.3.1 Demographic Features

Variable Name	Variable Description
RIAGENDR	Participant gender
RIDAGEYR	Age in years at screening
RIDRETH3	Reported Race
DMDEDUC2	Education level - Adults 20+
INDHHIN2	Annual household income
INDFMIN2	Annual family income
INDFMPIR	Ratio of family income to poverty

These features were selected based on several important data points from the background research, indicating the importance of gender, age, race, and socioeconomic status as risk factors for a positive cancer diagnosis. Specifically, the features selected here fall into the categories of: (3a, 1b) Age, (4a) Combined Sex and Race, and (5a) Socioeconomic status.

[3] NIH National Cancer Institute, Risk Factors for Cancer: `https://www.cancer.gov/about-cancer/causes-prevention/risk`

[4] NIH National Cancer Institute, Cancer-Causing Substances in the Environment: `https://www.cancer.gov/about-cancer/causes-prevention/risk/substances`

1.3.2 Dietary Features

Variable Name	Variable Description
DR1TALCO	Alcohol consumed (g)
DR1TKCAL	Kcal consumed
DR1TSUGR	Sugar Consumed (g)
DRQSDT1	Weight loss /Low-Calorie Diet
DR2TALCO	Alcohol consumed (g)
DR2TKCAL	Kcal consumed
DR2TSUGR	Sugar Consumed (g)

These features were selected based on the obesity and diet components unearthed in the background research section, which indicate that not only current dietary intake, but dietary intake trends and habits, are likely predictive of positive cancer diagnosis. Specifically, the features selected here fall into the categories of: (2b) Alcohol, (5b) Diet, and (9b) Obesity.

1.3.3 Examination Features

Variable Name	Variable Description
BMXWT	Weight (kg)
BMXHT	Standing Height (cm)
BMXWAIST	Waist Circumference (cm)
BPXCHR	60-second heart rate
BPXPLS	60-second pulse
BPXPULS	Pulse regular or irregular
BPXSY1	Systolic BP in mm Hg
BPXDI1	Diastolic BP in mm Hg
BMXBMI	BMI

These features were selected based on the chronic inflammation and obesity components of the background research section. Specifically, the features selected here fall into the categories of: (4b) Chronic Inflammation, (5b) Diet, and (9b) Obesity.

1.3.4 Laboratory Features

Variable Name	Variable Description
LBXBPB	Blood lead (ug/dL)
LBXBCD	Blood cadmium (ug/L)
LBXBSE	Blood selenium (ug/L)
LBXBMN	Blood manganese (ug/L)
URXUHG	Urine Mercury (ng/mL)
LBXTC	Total Cholesterol

These features were selected based on the hormones and cancer-causing substances components of the background research, which should be indicative of positive cancer diagnosis. Specifically, the features selected here fall into the categories of: (3b) Cancer-Causing Substances, (6b) Hormones, (9b) Obesity, and (10b) Radiation.

1.3.5 Questionnaire Features

Variable Name	Variable Description
MCQ080	Doctor ever said you were overweight
MCQ365a	Doctor told you to lose weight
MCQ365b	Doctor told you to exercise
MCQ365d	Doctor told you to reduce fat/calories
MCQ365c	Doctor told you to reduce salt in diet
MCQ370a	Are you now controlling or losing weight
MCQ370b	Are you now increasing exercise
MCQ370c	Are you now reducing salt in diet
MCQ370d	Are you now reducing fat in diet
INDFMMPI	Family monthly poverty level index
IND310	Total savings/cash assets for the family
DBQ700	How healthy is the diet
DBD895	# of meals not home prepared
DBD900	# of meals from fast food or pizza place
CBQ540	Used nutrition info to choose fast foods
CBQ585	Used nutrition info in restaurant
ALQ130	Avg # alcoholic drinks/day - past 12 mos
ALQ141Q	# days have 4/5 drinks - past 12 mos
PAD645	Minutes walk/bicycle for transportation per day (average)
PAQ677	Past wk # days cardiovascular exercise
SLD012	Sleep hours
SLQ050	Ever told doctor had trouble sleeping?
SMQ621	Cigarettes smoked in entire life
SMQ905	How many days used an e-cigarette?
SMD480	In past week # days person smoked inside
SMDANY	Used any tobacco product last 5 days?
SMQ020	Smoking
SMD030	Smoking
ALQ101	Alcohol Consumption
ALQ120Q	Alcohol Consumption

These features were selected based on several of the aforementioned risk factors, including: obesity, diet, tobacco and alcohol use, and sunlight. This data is especially valuable as it may provide insight into subject habits over the long term (in lieu of any self-report bias that may exist in this questionnaire data) which is extremely valuable in determining potential cancer state according to background research. Specifically, the features selected here fall into the categories of: (5a) Socioeconomic Status, (1b) Alcohol, (5b) Diet, (9b) Obesity, and (11b) Sunlight, (12b) Tobacco.

1.4 Feature Evaluation

After pre-processing, as described in the next section, the first step taken in evaluating the features was calculating their relative importance. This is a useful first evaluation test as it can directly communicate the relevance of a given feature on predicting the target, and necessary as the selected 75 features above total 168 unique features after one-hot pre-processing has been applied, thus necessitating evaluation and pruning.

1.4.1 Tree-based selection

There exists a large body of research, including the results of Genuer et al. [2010], demonstrating the efficacy of feature selection using importance derived from Tree-based classifiers. The specific classifier that was used for this pre-processing task is the Extra Trees (or Extremely Randomized Trees) Classifier, first proposed by Geurts et al. [2006]. It was used as implemented in the Extra-TreesClassifier in the Python scikit-learn library. The theoretical underpinning of this classifier that makes them an attractive option for feature importance ranking is threefold:

Number of Averaged ExtraTreesClassifiers	Number of Estimators per ExtraTreesClassifier	Accuracy	F1	Precision	Recall
2	50	0.8275	0.8263	0.8368	0.8275
5	50	0.8350	0.8342	0.8414	0.8350
10	50	0.8205	0.8197	0.8267	0.8205
5	20	0.8210	0.8197	0.8301	0.8210
5	50	0.8350	0.8342	0.8414	0.8350
5	**100**	0.8380	0.8371	0.8455	0.8380

Table 1: Calculating the number of classifiers to average output from, and the number of estimators per classifier.

Importance Threshold	Accuracy	F1	Precision	Recall
0.5	0.8370	0.8360	0.8450	0.8370
0.6	0.8380	0.8371	0.8455	0.8380
0.7	0.8170	0.8158	0.8255	0.8170

Table 2: Calculating the importance threshold for the ExtraTreesClassifier importance vector that maximized model output

1. The classifier is computationally efficient, so we can run it on the entirety of our input data. This is essential because after one-hot encoding of categorical features, we have 168 features, which would become difficult to manage with an inefficient classifier.
2. The ExtraTreesClassifier is essentially a Tree-based ensemble classifier in which there is strong randomization for both attribute and cut-point choice while splitting a tree node (hence the efficiency), allowing it to perform better with noisy input data (i.e. the un-evaluated features).
3. As a tree-based ensemble, the ExtraTreesClassifier can handle both one-hot categorical data and real-valued, continuous data simultaneously. This is critical, as our dataset contains both.

To use the ExtraTreesClassifier as a feature selector, it was simply fit to the entire dataset and target vector 2 times. Each time, the resulting unit vector of importance values, corresponding to the each feature, was extracted. After all runs, the importance values were averaged per feature and sorted to provide the expected importance of each feature.

In order to tune the ExtraTreesClassifier, the hyper-parameters were modified, keeping all other variables in the procedure constant, and quantified by the accuracy of a Random Forest Classifier upon the output. The results are described in Table 1.4.1 and Table 2.

Using the three resulting tuned hyperparameters for the ExtraTreesClassifier, the top 31 features were selected from the feature set, wich are displayed, along with their respective importance score (ranging from 0 to 1) in Figure 1.

1.4.2 Principal Component Analysis

The other methodology employed for feature evaluation was Principal Component Analysis (PCA). This methodology essentially computes the covariance matrix of the supplied data and then returns the n features with the highest variance, where n is supplied by the experimenter/user. This allowed for further dimensionality reduction of the dataset, by essentially computing the variance of each feature and allowing us to maintain a specific percentage of the original variance represented in the data while removing some irrelevant features.

Using the scikit-learn package and it's included PCA functionality, the behavior was slightly modified. Retrieving the explained_variance_ attribute from the PCA function, we get a vector containing the proportion of the total dataset variance contained in each feature. We can then keep the top n features corresponding to the desired fraction of maintained variance σ^2. This is another hyper-parameter

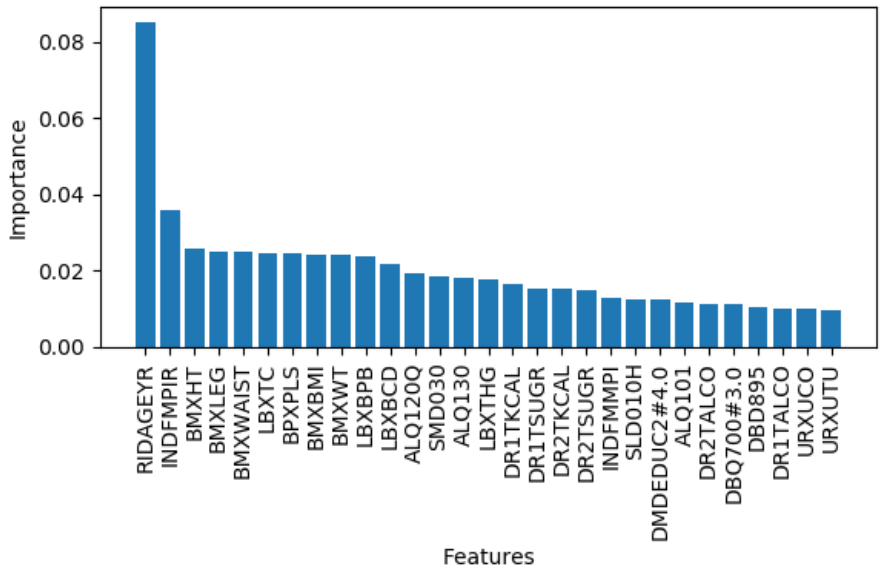

Figure 1: Top 31 Features resulting from tuned ExtraTreesClassifier ranking

Retained Variance	Accuracy	F1	Precision	Recall
0.75	0.8300	0.8292	0.8363	0.8300
0.85	0.8320	0.8311	0.8394	0.8320
0.95	0.8235	0.8226	0.8301	0.8235

Table 3: Calculating optimal parameter for retained variance in features dropped after PCA analysis

that must be tuned, and was done in a similar manner to the tuning of the ExtraTreesClassifier's hyper-parameters. Namely, the σ^2 was modified, keeping all other variables in the procedure constant, and quantified by the accuracy of a Random Forest Classifier upon the output. The results are described in Table 3 to achieve the optimal $\sigma^2 = 0.85$.

Using the resultant retained variance value, additional features could be removed. Because PCA is sensitive to the scale of data, the data was first separated into one-hot encoded categorical features, and real-valued features, respectively containing 2 and 29 of the remaining 31 features after the feature pruning based on the output of the ExtraTreesClassifier. These data were separately applied to the PCA function, and the resulting feature variances for the categorical and real-valued features that were computed are produced below in Figures 2 and 3 respectively.

With this proportion of the total variance explained by each feature computed, and the optimal retained variance computed to be 0.85 via hyperparameter tuning, only the features representing the top 85% of the variance were retained. This reduced the feature sets to 2 categorical and 17 real-valued features, which amount to the final features to be used in the model. The features that were retained are described in Table 4.

2 Pre-processing and Feature Engineering

2.1 Feature Pre-processing

Pre-processing of features occurred in 2 primary categories: real valued continuous features and categorical features. The data collected from the NHANES database contained both types of data, each of which had to be processed differently to ensure that the data was usable with subsequent feature engineering and selection.

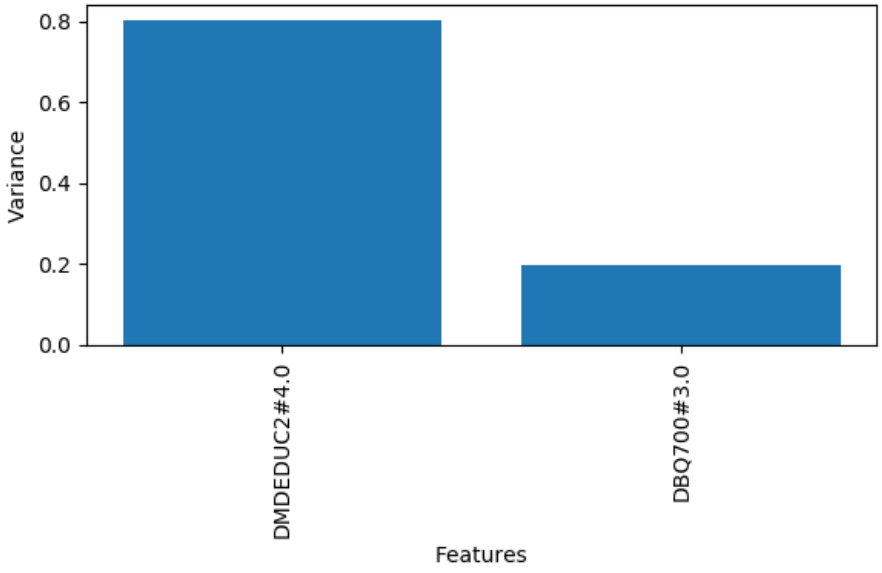

Figure 2: Proportion of variance of in dataset represented by categorical features

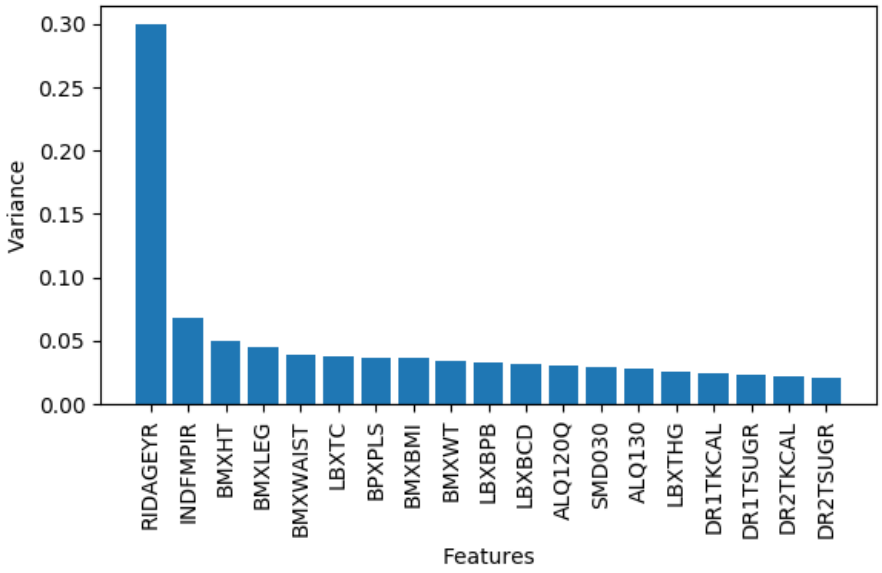

Figure 3: Proportion of variance of in dataset represented by real-valued features

Code	Description	Data Category
RIDAGEYR	Participant gender	Demographic
INDFMPIR	Ratio of family income to poverty	Demographic
BMXHT	Standing Height (cm)	Examination
BMXLEG	Upper Leg Length (cm)	Examination
BMXWAIST	Waist Circumference (cm)	Examination
LBXTC	Total Cholesterol(mg/dL)	Laboratory
BPXPLS	60-second pulse	Examination
BMXBMI	Body Mass Index (kg/m**2)	Examination
BMXWT	Weight (kg)	Examination
LBXBPB	Blood lead (ug/dL)	Laboratory
LBXBCD	Blood cadmium (ug/L)	Laboratory
ALQ120Q	How often drink alcohol over past 12 mos	Questionnaire
SMD030	Age started smoking cigarettes regularly	Questionnaire
ALQ130	Avg # alcoholic drinks/day - past 12 mos	Questionaire
LBXTHG	Blood mercury, total (ug/L)	Laboratory
DR1TKCAL	Kcal consumed	Dietary
DR1TSUGR	Sugar Consumed (g)	Dietary
DMDEDUC2#4.0	Education level - Adults 20+: Some college or AA degree	Demographic
DBQ700#3.0	How healthy is the diet: Good	Questionaire

Table 4: Final features resulting from feature selection methodologies.

2.1.1 Real-Valued Continuous Features

Several different techniques were considered for these features, informed by the work of Kumar Singh et al. [2015], primarily **Z-scoring** and **linear scaling** to unit range as they demonstrated the strongest results in the aforementioned work and are commonly employed in industry in addition to academia. First, however, all real valued, continuous features were imputed. That is, all missing values for a particular feature were replaced with the mean value for that feature f, μ_f, calculated as:

$$\frac{1}{N}\sum_{n=1}^{N} f_n = \mu_f$$

In **Z-scoring**, each element in the feature vector f is normalized to have mean of 0 and standard deviation of 1 as follows:

$$f = \frac{f - \mu_f}{\sigma_f}$$

Where σ_f is computed as:

$$\frac{1}{N-1}\sum_{n=1}^{N}(f_n - \mu_f)^2 = \sigma_f^2$$

Linear scaling to unit range was done by computing the following for every element of the feature vector f:

$$f_n = \frac{f_n - min(f)}{max(f) - min(f)}$$

2.1.2 Categorical Features

The categorical features encountered in the NHANES dataset were subject to one-hot encoding. That is, one single categorical feature with n categories was transformed into n features each of which is discretized to be either a 1 (i.e. yes) or 0 (i.e. no).

The potential downside of this approach is that the amount of information contained in these categorical features will be disproportionate, as discerned by some of the feature engineering principles used in this work (primarily Principal Component Analysis). This is because, when compared to continuous variables that fall within a range, they can consistently fall at either end of their 'range', simulating extremely high variance. This was corrected by performing PCA on continuous and categorical features separately.

2.2 Feature Engineering

2.2.1 Application of Pre-Processing Functions

While one-hot encoding is the most viable approach to pre-processing the categorical data, several viable options existed for pre-processing of the continuous data. Among these were the following:

1. Apply z-scoring only
2. Apply linear scaling only
3. Apply z-scoring, followed by linear scaling
4. Apply linear scaling, followed by z scoring
5. Apply no pre-processing function (other than mean imputation)

The efficacy of each of the aforementioned methodologies was tested in a similar method to the testing of the feature selection methods, by determining the average accuracy of our Random Forest classifier over 10 runs while only varying the above pre-processor function. The results are produced below, and while there isn't a very significant dispersion, the best results seem to be achieved using only Z-scoring, so it was selected moving forward.

Method	**Accuracy**
Z-Scoring	0.826
Linear Scaling	0.818
Z-Scoring, then Linear Scaling	0.799
Linear Scaling, then Z-Scoring	0.816
Just Imputing	0.799

Table 5: Pre-Processing Method efficacies

3 Predictive Modeling

3.1 Model Selection

Once the data had been selected and pre-processed, the last requisite step was selecting a model to then modify with the objective of producing the best accuracy scores in predicting the cancer target. Several types of models were considered based on previous research in the field, the particular strengths and weaknesses of the model, and their applicability to the single class classification task at hand, as is described below.

3.1.1 Logistic regression

Logistic regression was examined as one potential algorithmic framework, as the task at hand was binary classification, for which logistic regression is designed to work. It is a widely used technique because it is very efficient, does not require too many computational resources, it can easily be interpreted, it does not require input features to be scaled, it does not require significant tuning, it is easy to regularize, and it outputs well-calibrated predicted probabilities.

Conversely, the model also has several disadvantages. First, it is very sensitive to ordering of data and prone to overfitting, which is circumvented by the high number of permutations we perform in input test and train data, as well as strict test-train-validation data separation. Another than must be considered is that it performs significantly better in the absence of noisy or irrelevant data, a

Trial number	Penalty	Solver	C-value	Mean Test Score
16	l1	liblinear	0.233572	0.8305
21	l2	liblinear	1.623777	0.8296
22	l1	liblinear	4.281332	0.8290
24	l1	liblinear	11.28838	0.8290
18	l1	liblinear	0.615848	0.8290
33	l2	liblinear	545.5595	0.8287
34	l1	liblinear	1438.45	0.8287
37	l2	liblinear	3792.69	0.8287
35	l2	liblinear	1438.45	0.8287
36	l1	liblinear	3792.69	0.8287

Table 6: Top 10 cross-validation testing scores for Logistic Regression classifier

roadblock that the previous sections' feature engineering and selection methodologies allow us to mostly circumvent[5].

The logistic regressor has three primary hyper-parameters that need to be tuned to achieve better results on the test dataset, which are penalty, c-value, and the solver. Penalty controls whether an $l1$ (absolute value) or $l2$ (squared euclidean) distance is used in the measurements, C controls the inverse regularization strength (which balances weighting of samples with maintaining a simpler model), and the solver which controls specific algorithm to use in the optimization problem. The results of a 5-fold cross validation tuning of the aforementioned hyper parameters is described in Table 6, ordered in descending order of performance (i.e. most performant model first).

3.1.2 Decision Tree

Decision tree is among the two simpler models tested in this work (the other being Logistic regression), as there are several distinct advantages to using this type of model, in addition to work done by Oh and Park [2004] showing the extremely high efficacy in predicting cancer-related decision making using Decision tree models.

The advantages of the Decision tree model, other than proven success in similar classification tasks, are that they are easy to understand and interpret, function well with categorical and non-categorical data, and can easily be combined with other techniques or used in ensemble methods. This being accounted for, there are also some marked pitfalls to avoid using this model. Namely, they are unstable and can have large changes in structure due to small changes in the data, and they can be relatively inaccurate compared to other models with similar data.

Both these concerns can be addressed (at least partially) by using the tree as a weak learner in a larger ensemble classifier, such as AdaBoost or a Random Forest, as described in subsequent sections. However, in using these ensemble methods, we lose much (if not all) of the ease of understanding of this type of model. Finally, the trees can bias towards categorical variables with larger numbers of levels, which we circumvent by one-hot encoding our categorical variables so they all have only two binary levels.

The hyperparameters to tune for the Decision Tree classifier can include several different criteria, however because the package offered by scikit-learn offers methods within the classifier to pick the optimal parameter for many of these criteria (including: splitting strategy for nodes of the tree, max depth of the tree, minimum number of samples to consider per split, etc.), the hyperparameter examined was the decision function to measure the quality of a split. The examined criteria were "gini" for the Gini impurity and "entropy" for the information gain, and the results of the 5-fold cross validation to optimize this parameter are displayed in Table 7.

3.1.3 Support Vector Machine

Support vector machines (or SVM's) represent a significantly more powerful learning model than the aforementioned Decision trees and Logistic regression models. Moreover, they have been shown

[5] *The Logistic Regression Algorithm*, TEILEN MIT, `https://machinelearning-blog.com/2018/04/23/logistic-regression-101/`

Trial Number	Criterion	Mean Test Score
41	entropy	0.7817
40	gini	0.7786

Table 7: All cross-validation testing scores for Decision Tree Classifier

Trial Number	Kernel	Gamma	C	Mean Test Score
46	linear	auto	0.01	0.8312
48	linear	scale	0.01	0.8312
58	linear	auto	10	0.8296
60	linear	scale	10	0.8296
56	linear	scale	1	0.8293
54	linear	auto	1	0.8293
57	rbf	scale	1	0.8290
52	linear	scale	0.1	0.8290
50	linear	auto	0.1	0.8290
55	rbf	auto	1	0.8234

Table 8: Top 10 cross-validation testing scores for Support Vector Machine

to have strong predictive power in similar cancer prediction problems, as exemplified by the work of Huang et al. [2017] in using SVM's and SVM ensembles to predict breast cancer incidence with high accuracy, a task similar to the one at hand in this work. The advantages conferred by SVM's are manifold, but those most pertinent to the application at hand are that it's regularization parameter helps prevent overfitting on the data, the kernel trick allows us to generate 'expert knowledge' about the high dimensional data without needing to deeply understand how it is working, and that it can build complex decision boundaries (unlike the previous two weaker classifiers).

It does, however, come with its own drawbacks. Chiefly, choosing good values of hyerparameters for the model is quite difficult, as the effects of different kernel functions, C, and γ values are fairly unintuitive. To circumvent these shortcomings, the hyperparameter tuning of the SVM is critical, and will be accomplished using a 5-fold cross-validated exhaustive grid search, searching over hyperparameter value ranges that have shown to be effective in previous work in cancer detection, such as the work of Huang et al. [2017] and others.

The key hyperparameters that needed to be tuned for the SVM include the choice of kernel function, and the values of Gamma (γ) and C. C specifies the float penalty parameter of the SVM, i.e. how 'hard' the decision boundary created by the classifier is. The kernel function essentially specifies the function used to efficiently transform the input data into an high dimensional space within which the SVM can learn it's optimal decision boundary, such as the linear and radial basis function kernels tested here. Lastly, The γ term is a free parameter of the radial basis function kernel, and controls the amount of variance in the Gaussian distribution described by the kernel, which was passed the either the 'auto' or 'scale' functions. Auto sets $\gamma = \frac{1}{number features}$, whereas 'scale' sets $\gamma = \frac{1}{number features \times variance(featureset)}$. The results are described in Table 8.

3.1.4 AdaBoost Classifier

AdaBoost, as described by Freund and Schapire [1997], is the first ensemble algorithm considered. It essentially functions by combining weak learners (in this case decision stumps, which are simply decision trees with only a single split). It combines these weak learners by consistently re-weighting them based on difficulty to classify particular examples within the dataset.

With this understanding, it becomes clear that AdaBoost can potentially help overcome some of the shortcomings of decision trees, the weak learner it is based upon. The primary problems that are solved are the trees' instability (large changes in structure due to small changes in the data), and inaccuracy by essentially creating a 'weighted-average' of the trees, and avoiding the pitfalls of any one tree.

Trial Number	Number of Estimators	Algorithm	Mean Test Score
68	30	SAMME.R	0.8271
69	40	SAMME.R	0.8243
70	50	SAMME.R	0.8243
67	80	SAMME	0.8225
66	70	SAMME	0.8221
72	70	SAMME.R	0.8215
71	60	SAMME.R	0.8197
64	50	SAMME	0.8187
65	60	SAMME	0.8184
73	80	SAMME.R	0.8181

Table 9: Top 10 cross-validation testing scores for the AdaBoost Classifier

The only significant disadvantage of AdaBoost, or nearly any other ensemble method, is the increased computational complexity and cost associated with both training and evaluation of data. However, since the goal of the work described herein is not on-line, and has no particularly latency or computational complexity restrictions, we can all but ignore this disadvantage.

The hyperparameters that needed to be tuned for the AdaBoost classifier were the number of estimators used in the ensemble and the algorithm used for the re-weighting steps. For the prior, a sweep of values was performed, while the algorithms 'SAMME' and 'SAMME.R' were considered for the latter. According to the scikit-learn documentation, the primary difference between the two options is that the SAMME.R algorithm typically converges faster than SAMME, achieving a lower test error with fewer boosting iterations[6]. The results of the 5-fold cross validation are described in Table 9.

3.1.5 Random Forest Classifier

The final model considered was the Random Forest, an ensemble classifier first described by Breiman [2001]. The classifier is essentially a combination of Decision Tree classifiers in which each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest.

A significant advantage of this ensemble classifier, as compared to the previously mentioned AdaBoost classifier, is that even when using a random selection of features to split each node, it is able to produce error rates that compare favorably to AdaBoost, which must perform a somewhat cumbersome computation at each re-weighting. Because of this fact, the random forest is more robust with respect to noise. This latter feature serves as a good stop-gap in the work described here, wherein there have been several attempts to reduce noise from the data via the previously mentioned feature engineering, selection, and preprocessing that may not have been totally sufficient to remove noisy data.

The hyperparameters that were chosen to tune for the Random Forest Classifier were whether the model bootstrapped, the number of estimators (see: internal decision trees) to use, number of features to consider on each split of an internal tree, the minimum number of samples to weigh per leaf node, and the minimum number of samples to weigh per split of the internal trees. In this case, bootstrapping simply refers to whether bootstrap samples are used when building trees. In the False case, the whole dataset is used to build each internal tree. The results of the 5-fold cross validation are displayed in Table 10.

3.1.6 Model Comparison

To perform the computations of the previous Model Selection sections, the scikit-learn functions GridSearchCV and Pipeline were employed. The GridSearchCV allows for a grid search (i.e. exhaustive search) over all possible permutations of a given set of hyperparameters and models. Moreover, it allows for seamless k-fold cross validation of the resulting training and testing scores

[6] scikit-learn documentation, *sklearn.ensemble.AdaBoostClassifier*, `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html`

Trial Number	Bootstrapping On	Number of Estimators	Max Features per Split	Min Samples per Leaf	Min Samples per Split	Mean Test Score
635	TRUE	90	0.78	12	5	0.8427
664	TRUE	80	0.8	10	5	0.8420
661	TRUE	100	0.8	10	3	0.8417
528	TRUE	130	0.73	10	3	0.8417
577	TRUE	110	0.75	10	5	0.8417
623	TRUE	130	0.78	10	5	0.8417
680	TRUE	90	0.8	12	5	0.8414
642	TRUE	110	0.78	12	7	0.8414
411	TRUE	100	0.6	12	5	0.8414
597	TRUE	110	0.75	12	7	0.8414

Table 10: Top 10 cross-validation testing scores for the Random Forest Classifier

Classifier	Train Score		Test Score		
	Mean	Std Dev.	Mean	Std Dev.	Rank
Random Forest	0.9021	0.0033	0.8427	0.0216	1
SVM	0.8362	0.0038	0.8312	0.0160	406
Logistic Regression	0.8333	0.0039	0.8305	0.0161	431
AdaBoost	0.8473	0.0014	0.8271	0.0186	599
Decision Tree	1.0000	0.0000	0.7817	0.0081	686

Table 11: Top 5 fold cross-validation training and testing scores for each classifier

of the models, allowing for statistically and methodologically sound selection of hyperparameters. Combining this function with a Pipeline object allowed for a significantly increased computation speed, as the Pipeline object allows for tasks using the same resources to be executed in parallel. In this case, the shared resource were folds of the validation dataset, which allowed this total computation to be completed in just under 30 minutes, rather that several hours via a less efficient approach.

Based on the above results of the previous section, it is clear that the random forest classifier performed best and was thus selected as the model that would ultimately be used with the final data selection previously. The results of the overall five fold cross validation, broken down by the best performer for each classifier, is described in Table 11. It should be noted when looking at rank that there were a total of 704 permutations of search conditions tested, and that the best performing SVM was still less performant than almost all Random Forest classifiers tested, which is why trial numbers and rank numbers have such large gaps between them.

table

3.2 Model Performance

Model performance was quantified by training the model with a balanced training set, and then testing it with a smaller, disjoint testing set, of 5000 and 1000 data-points respectively. The results of this test can be seen tabulated in Table 12 in terms of the most common indices of machine learning model performance characterization. Of note is the the high value of recall at 0.844. Recall is defined as follows:

$$recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

In the particular application of this work, the prediction of cancer and malignancies, a false negative (i.e. saying a patient is not likely to have cancer or malignancy, when in fact they are) is perhaps the worst possible outcome, as it can lead to the most potentially dire health consequences for the patient. In this regard, a higher ratio of true positives to the sum of true positives and false negatives is indicative of fewer false positives in the model's results, which by the previous logic is a highly desirable trait in our model.

Accuracy	F1-Score	Precision	Recall
0.844	0.844	0.848	0.844

Table 12: Performance metrics of the final Random Forest classifier

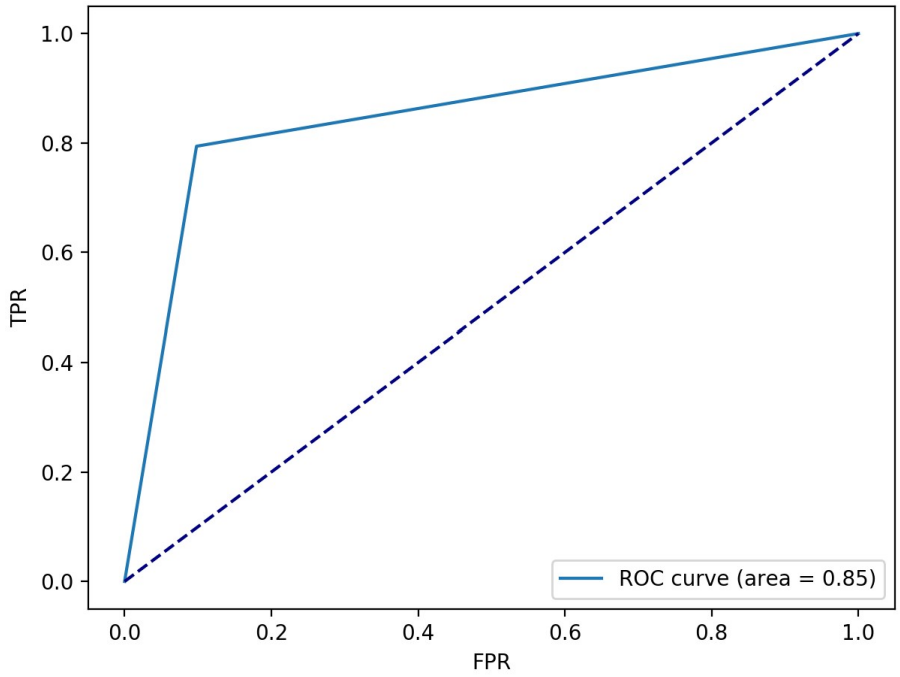

Figure 4: Receiver Operating Characteristic for the final Random Forest classifier

Also detailed in Figure 4 is the Receiver Operating Characteristic (ROC) curve, graphing the false positive rate (FPR) against the true positive rate (TPR) of the model, with the dashed blue line demonstrating the ROC curve of a random guess classifier. Also displayed on Figure 4 is the Area Under ROC (AUROC) measurement, for which 1 represents perfect performance, 0 represents the worst possible performance, and 0.5 represents random guess performance. In this regard, the AUROC value of 0.85 achieved by the model is again indicative of a rather strong performance in the given task.

4 Conclusion

In conclusion, this work has shown that using a novel combination of relatively simple techniques allows us to create a highly performant cancer and malignancy prediction model with a relatively small subset of the originally examined data.

With the goal of predicting whether a given individual would be diagnosed with cancer or malignancy, we began by using expert resources to inform what types of data were useful to collect from the CDC NHANES database with regard to cancer diagnoses. Subsequently, 75 features were selected, which were expanded to 168 unique features when categorical features were separated into unique, one-hot encoded binary features and z-scored real-valued features.

With these 168 features, the importance information averaged from several averaged Extra Trees Classifiers was used to remove the bottom 40% of features in terms of usefulness in predicting the cancer outcome. To finish the feature selection, PCA was then applied to the remaining data to retain only the features that accounted for the top 85% of the variance in the remaining data set.

Finally, several models were examined and the Random Forest Classifier selected as the most performant of the hyperparameter-tuned models examined, and used to evaluate the resultant testing and training datasets to achieve the results reported in the last section, namely an AUROC of 0.85 and accuracy of 0.844.

Perhaps the two most interesting and important conclusions of this work are the following:

1. Of the 75 initial features, selected, the highest performance was achieved with a very small subset of 19 features, as described in section 1.4.2.
2. With this relatively short list of features, a model yielding very high accuracy and recall could be produced using standard methods.

4.0.1 Future Research

Future extensions of this research can look to build upon the existing framework to predict specific types of cancer, or validate the cancer-risk of the features identified as important to predicting the target label in this work.

References

Centers for disease control, nhanes survey 2015-2016, 2019a. URL https://wwwn.cdc.gov/nchs/nhanes/continuousnhanes/default.aspx?BeginYear=2015.

Uscs data visualizations, 2019b. URL https://gis.cdc.gov/Cancer/USCS/DataViz.html.

Robin Genuer, Jean-Michel Poggi, and Christine Tuleau-Malot. Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225 – 2236, 2010. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2010.03.014. URL http://www.sciencedirect.com/science/article/pii/S0167865510000954.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, Apr 2006. ISSN 1573-0565. doi: 10.1007/s10994-006-6226-1. URL https://doi.org/10.1007/s10994-006-6226-1.

Bikesh Kumar Singh, Kesari Verma, and A S. Thoke. Investigations on impact of feature normalization techniques on classifiers performance in breast tumor classification. *International Journal of Computer Applications*, 116:11–15, 04 2015. doi: 10.5120/20443-2793.

Hyo-Sook Oh and Hyeoun-Ae Park. Decision tree model of the treatment-seeking behaviors among korean cancer patients. *Cancer Nursing*, 27(4), 2004. ISSN 0162-220X. URL https://journals.lww.com/cancernursingonline/Fulltext/2004/07000/Decision_Tree_Model_of_the_Treatment_Seeking.1.aspx.

Min-Wei Huang, Chih-Wen Chen, Wei-Chao Lin, Shih-Wen Ke, and Chih-Fong Tsai. Svm and svm ensembles in breast cancer prediction. *PLOS ONE*, 12(1):1–14, 01 2017. doi: 10.1371/journal.pone.0161501. URL https://doi.org/10.1371/journal.pone.0161501.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139, 1997. ISSN 0022-0000. doi: https://doi.org/10.1006/jcss.1997.1504. URL http://www.sciencedirect.com/science/article/pii/S002200009791504X.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001. ISSN 1573-0565. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.