Wisconsin Diagnostic Breast Cancer Dataset Project Report

Youyang Han Jan 1st, 2017

1. Definition

1.1 Project Overview

When I searched datasets in Kaggle, I focused on the subject in health industry for making combination of machine learning and biology and make efforts to improve the better understanding of disease. Finally, I find the newest released dataset subset from classical Wisconsin Diagnostic Breast Cancer Dataset. I think this dataset meet the needs both for myself and Udacity.

This dataset contains mostly numerical features which describe characteristics of different picture qualities and only categorical feature of diagnosis. This is easy to identify target connection that only exists between features and diagnosis. What is more, people are concerning the accuracy of cancer sensing for the great price for miss decision.

This dataset is one image dataset of Wisconsin Diagnostic Breast Cancer (WDBC) which comes from UCI Machine Learning. It is originally created in November 1995. Creators of WDBC are Dr. William H. Wolberg, W. Nick Street and Olvi L. Mangasarian.
They worked in University of Wisconsin, Madison. Donor of this dataset is Nick Street.

This database is also available through the UW CS ftp server:
ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

1.2 Problem Statement

Each feature describes one of characteristics of cell images. All features are statistically computed from image space while lack meaning of connection to real diagnosis. To explore the dataset, one is expected to contribute to find relationship of features and diagnosis by using different machine learning methods. This can be viewed as binary classification problem by definition.

To explore dataset and solve this problem, I would like to separate process into 3 different categorical of method. First, I do not know if there is linear or non-linear relationship between features and diagnosis. So I will try both method to compare the benchmark accuracy provided by the dataset description. Second, I will try to improve the performance of method in condition. Finally, I will try to apply deep learning method on this problem to see if there is any space for improvement.

From the description of dataset, best predictive accuracy obtained using one separating plane in the 3-D space of Worst Area, Worst Smoothness and Mean Texture. Estimated accuracy 97.5% using repeated 10-fold cross validations. Classifier has correctly diagnosed 176 consecutive new patients as of November 1995. With exploring of dataset, I will apply different method and try to train and optimize the "best" model.

1.3 Metrics

Accuracy is our starting point. It is the number of correct predictions made divided by the total number of predictions made, multiplied by 100 to turn it into a percentage[1].

A clean and unambiguous way to present the prediction results of a classifier is to use a use a **confusion matrix**. For a binary classification problem the table has 2 rows and 2 columns. Across the top is the observed class labels and down the side are the predicted class labels. Each cell contains the number of predictions made by the classifier that fall into that cell.

	Positive	**Negative**
Positive	True Positive	False Positive
Negative	False Negative	True Negative

In this case, a perfect classifier would correctly predict 357 no recurrence and 212 recurrence which would be entered into the top left cell no recurrence/no recurrence (**True Negatives**) and bottom right cell recurrence/recurrence (**True Positives**).

Incorrect predictions are clearly broken down into the two other cells. **False Negatives** which are recurrence that the classifier has marked as no recurrence. We do not have any of those. **False Positives** are no recurrence that the classifier has marked as recurrence.

This is a useful table that presents both the class distribution in the data and the classifiers predicted class distribution with a breakdown of error types.

In a problem where there is a large class imbalance, a model can predict the value of the majority class for all predictions and achieve a high classification accuracy, the problem is that this model is not useful in the problem domain.This is called the **Accuracy Paradox**. For problems like, this additional measures are required to evaluate a classifier.

Precision is the number of True Positives divided by the number of True Positives and False Positives. Put another way, it is the number of positive predictions divided by the total number of positive class values predicted. It is also called the Positive Predictive Value (PPV).A low precision can also indicate a large number of False Positives.

Recall is the number of True Positives divided by the number of True Positives and the number of False Negatives. Put another way it is the number of positive predictions divided by the number of positive class values in the test data. It is also called Sensitivity or the True Positive Rate. Recall can be thought of as a measure of a classifiers completeness. A low recall indicates many False Negatives.

The **F1 Score** is the 2*((precision *recall)/(precision+recall)). It is also called the F Score or the F Measure. Put another way, the F1 score conveys the balance between the precision and the recall.

In summary, I choose accuracy, precision and F1 score to evaluate the performance of the model prediction.

2. Analysis

2.1 Data Exploration

This dataset are linearly separable using all 30 input features. Also, there are 2 predicting fields, diagnosis: B = benign, M = malignant. Total numbers of attributes: 32 (ID, diagnosis, 30 real-valued input features). Total number of instances: 569.

Features of this dataset are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. A few of the images can be found at http://www.cs.wisc.edu/~street/images/

Separating plane described above was obtained using Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree Construction Via Linear Programming." Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society, pp. 97-101, 1992], a classification method which uses linear programming to construct a decision tree. Relevant features were selected using an exhaustive search in the space of 1-4 features and 1-3 separating planes.

The actual linear program used to obtain the separating plane in the 3-dimensional space is that described in: [K. P. Bennett and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992,

23-34].

Training set: A set of examples used for learning, that is to fit the parameters of the classifier[fn20:]. **Here**

Validation set: A set of examples used to tune the parameters of a classifier, for example to choose the number of hidden units in a neural network.

Test set: A set of examples used only to assess the performance of a fully specified classifier.

Attribute Information:
1) ID number
2) Diagnosis (M = malignant, B = benign)
3)-32):Ten real-valued features are computed for each cell nucleus:
a) radius (mean of distances from center to points on the perimeter)
b) texture (standard deviation of gray-scale values)
c) perimeter
d) area
e) smoothness (local variation in radius lengths)
f) compactness ($perimeter^2$ / area - 1.0)
g) concavity (severity of concave portions of the contour)
h) concave points (number of concave portions of the contour)
i) symmetry
j) fractal dimension ("coastline approximation" - 1)

Specific description of several features:
The mean, standard error and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features.
For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius.

In detail, there is no missing attribute values. All feature values are recoded with four significant digits. Diagnosis class distribution are 357 benign 212 malignant.

For instance of dataset: data[0:5] with all features:

	id	diagnosis
0	842302	M
1	842517	M
2	84300903	M
3	84348301	M
4	84358402	M

Fig.1 Id numbers and diagnosis.

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean	_dimension_mean
0	17.99	10.38	122.8	1001	0.1184	0.2776	0.3001	0.1471	0.2419	0.07871
1	20.57	17.77	132.9	1326	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667
2	19.69	21.25	130	1203	0.1096	0.1599	0.1974	0.1279	0.2069	0.05999
3	11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414	0.1052	0.2597	0.09744
4	20.29	14.34	135.1	1297	0.1003	0.1328	0.198	0.1043	0.1809	0.05883
	radius_se	texture_se	perimeter_se	area_se	smoothness_se	compactness_se	concavity_se	concave points_se	symmetry_se	al_dimension_se
0	1.095	0.9053	8.589	153.4	0.006399	0.04904	0.05373	0.01587	0.03003	0.006193
1	0.5435	0.7339	3.398	74.08	0.005225	0.01308	0.0186	0.0134	0.01389	0.003532
2	0.7456	0.7869	4.585	94.03	0.00615	0.04006	0.03832	0.02058	0.0225	0.004571
3	0.4956	1.156	3.445	27.23	0.00911	0.07458	0.05661	0.01867	0.05963	0.009208
4	0.7572	0.7813	5.438	94.44	0.01149	0.02461	0.05688	0.01885	0.01756	0.005115
	radius_worst	texture_worst	perimeter_worst	area_worst	smoothness_worst	compactness_worst	concavity_worst	concave points_worst	symmetry_worst	_dimension_worst
0	25.38	17.33	184.6	2019	0.1622	0.6656	0.7119	0.2654	0.4601	0.1189
1	24.99	23.41	158.8	1956	0.1238	0.1866	0.2416	0.186	0.275	0.08902
2	23.57	25.53	152.5	1709	0.1444	0.4245	0.4504	0.243	0.3613	0.08758
3	14.91	26.5	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.173
4	22.54	16.67	152.2	1575	0.1374	0.205	0.4	0.1625	0.2364	0.07678

Fig.2 30 features of dataset.

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean	_dimension_mean
count	569	569	569	569	569	569	569	569	569	569
mean	14.127292	19.289649	91.969033	654.889104	0.09636	0.104341	0.088799	0.048919	0.181162	0.062798
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.07972	0.038803	0.027414	0.00706
min	6.981	9.71	43.79	143.5	0.05263	0.01938	0	0	0.106	0.04996
25%	11.7	16.17	75.17	420.3	0.08637	0.06492	0.02956	0.02031	0.1619	0.0577
50%	13.37	18.84	86.24	551.1	0.09587	0.09263	0.06154	0.0335	0.1792	0.06154
75%	15.78	21.8	104.1	782.7	0.1053	0.1304	0.1307	0.074	0.1957	0.06612
max	28.11	39.28	188.5	2501	0.1634	0.3454	0.4268	0.2012	0.304	0.09744
	radius_se	texture_se	perimeter_se	area_se	smoothness_se	compactness_se	concavity_se	concave points_se	symmetry_se	al_dimension_se
count	569	569	569	569	569	569	569	569	569	569
mean	0.405172	1.216853	2.866059	40.337079	0.007041	0.025478	0.031894	0.011796	0.020542	0.003795
std	0.277313	0.551648	2.021855	45.491006	0.003003	0.017908	0.030186	0.00617	0.008266	0.002646
min	0.1115	0.3602	0.757	6.802	0.001713	0.002252	0	0	0.007882	0.000895
25%	0.2324	0.8339	1.606	17.85	0.005169	0.01308	0.01509	0.007638	0.01516	0.002248
50%	0.3242	1.108	2.287	24.53	0.00638	0.02045	0.02589	0.01093	0.01873	0.003187
75%	0.4789	1.474	3.357	45.19	0.008146	0.03245	0.04205	0.01471	0.02348	0.004558
max	2.873	4.885	21.98	542.2	0.03113	0.1354	0.396	0.05279	0.07895	0.02984
	radius_worst	texture_worst	perimeter_worst	area_worst	smoothness_worst	compactness_worst	concavity_worst	concave points_worst	symmetry_worst	_dimension_worst
count	569	569	569	569	569	569	569	569	569	569
mean	16.26919	25.677223	107.261213	880.583128	0.132369	0.254265	0.272188	0.114606	0.290076	0.083946
std	4.833242	6.146258	33.602542	569.356993	0.022832	0.157336	0.208624	0.065732	0.061867	0.018061
min	7.93	12.02	50.41	185.2	0.07117	0.02729	0	0	0.1565	0.05504
25%	13.01	21.08	84.11	515.3	0.1166	0.1472	0.1145	0.06493	0.2504	0.07146
50%	14.97	25.41	97.66	686.5	0.1313	0.2119	0.2267	0.09993	0.2822	0.08004
75%	18.79	29.72	125.4	1084	0.146	0.3391	0.3829	0.1614	0.3179	0.09208
max	36.04	49.54	251.2	4254	0.2226	1.058	1.252	0.291	0.6638	0.2075

Fig.3 Statistics description of 30 features.

As we can see from above, features are all numerical feature whose numbers ranges from 0 to 4254. That is quite a large difference as the characteristics are not essentially related to each other. Since I will see the relationship between features and determine to standardize data into same level for better understanding of possible features. What is more, calculation and accuracy will be benefited from standardization.

2.2 Exploratory Visualization

First I choose to see if there are obvious imbalance in whole dataset. A heat map is a graphical representation of data where the individual values contained in a matrix are represented as colors.

Fractal maps and tree maps both often use a similar system of color-coding to represent the values taken by a variable in a hierarchy.Heat maps originated in 2D displays of the values in a data matrix. Larger values were represented by small dark gray or black squares (pixels) and

smaller values by lighter squares.

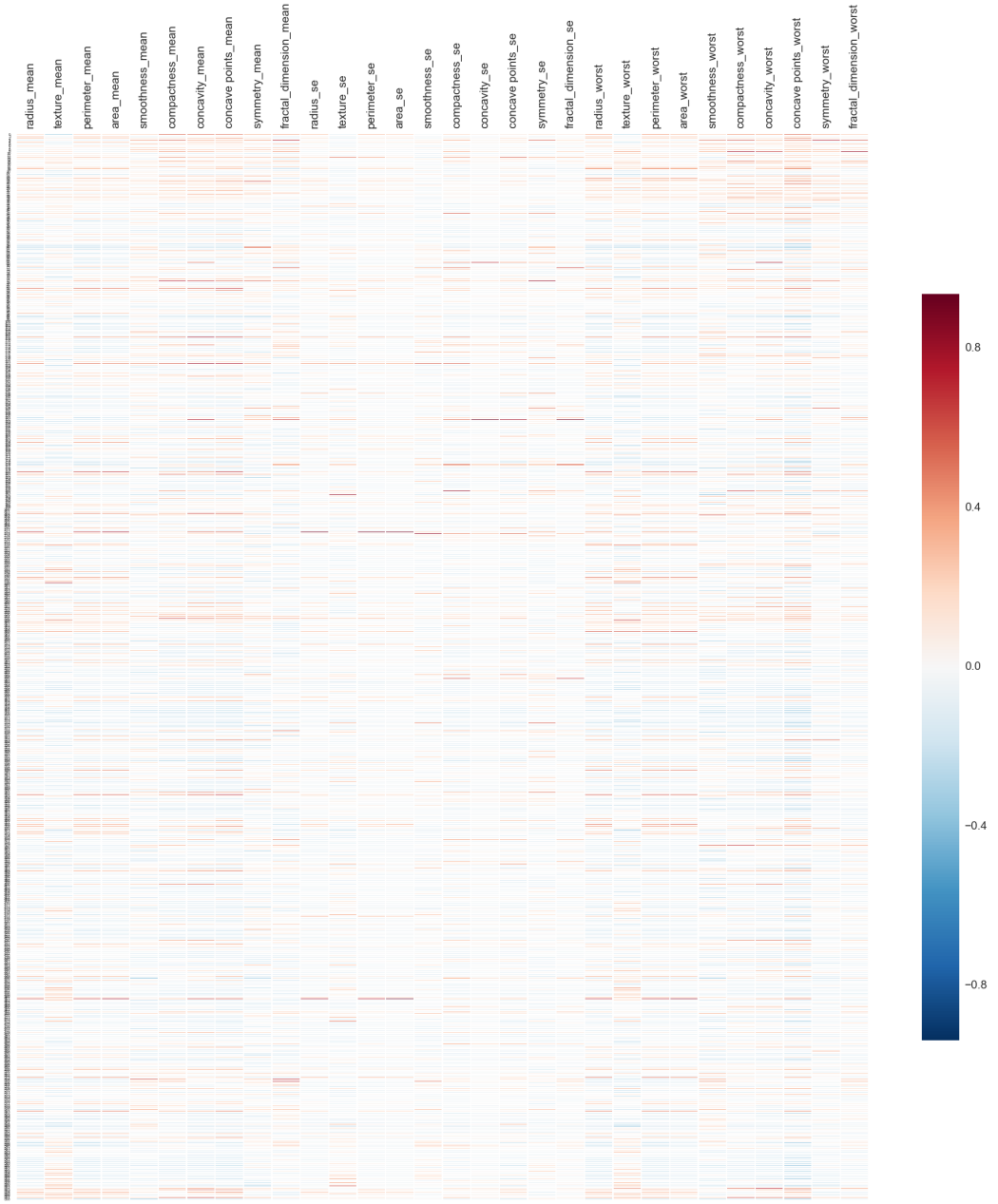


Fig.4 Heatmap of all data shows some imbalance in dataset.

As the map showed, there are no large imbalance for data inside each column to whole dataset, however there are some features do contain large imbalance in dataset which may be unsuitable for further analysis. I used outlier detection and removed those z-score larger than 3 since it may affect actual model we built[2].

Second, I will see if there are irrelative features to other features. In statistics, dependence or association is any statistical relationship, whether causal or not, between two random variables or two sets of data. Correlation is any of a broad class of statistical relationships involving dependence, though in common usage it most often refers to the extent to which

two variables have a linear relationship with each other.

The correlation matrix of n random variables X1, ..., Xn is the n × n matrix whose i,j entry is corr(xi, xj). Consequently, each is necessarily a positive-semidefinite matrix.
The correlation matrix is symmetric because the correlation between xi and xj is the same as the correlation between xj and xi.

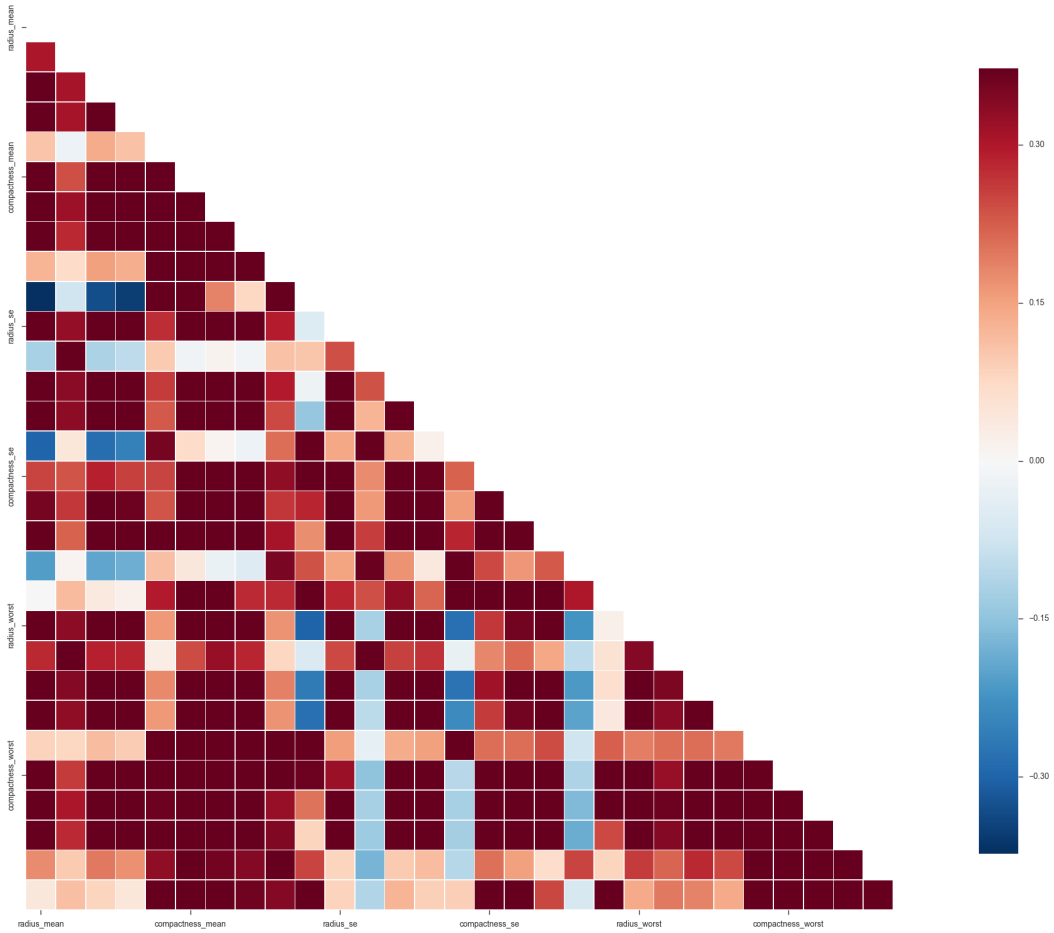


Fig.5 Correlation map of 30 features.

As map shows that most features have weak connection to other features. Only several features are relatively independent to others. Since I will try to reduce useless features to conduct feature selection at data processing.

Using correlation map is not enough for me to determine if features are suitable for classification since correlation map only show linear relationship between features. I used pair plot since it shows the location of two diagnosis in a single unit map after I used MinMaxScaler function for each feature. Pair Plot for 30 features and each 10 features are all listed in the supplementary.

The pair plot shows that indeed there are indeed several features are linearly correlated but most features can be separated by a 2-D hyperplane. So that I should try both linear and non-linear method and also try to find some different ways to handle 30 features at one model.

2.3 Algorithms and Techniques

Binary classification is the task of classifying the elements of a given set into two groups on the basis of a classification rule. Instancing a decision whether an item has or not some qualitative property, some specified characteristic[3].

Normalization of ratings means adjusting values measured on different scales to a notionally common scale, often prior to averaging. In more complicated cases, normalization may refer to more sophisticated adjustments where the intention is to bring the entire probability distributions of adjusted values into alignment[4].

In the field of machine learning, **linear classifiers** identify by making a classification decision based on the value of a linear combination of the characteristics. An object's characteristics are also known as feature values and are typically presented to the machine in a vector called a feature vector. Such classifiers work well for practical problems such as document classification, and more generally for problems with many variables, reaching accuracy levels comparable to non-linear classifiers while taking less time to train and use[5].

In machine learning, this problem can be defined as **binary classification**. However, there are linear and non-linear methods to solve this problem. In this question, I will use 4 common linear methods to see if we can find solutions at first. Then I choose non-linear method to see the accuracy. Finally, I apply powerful deep learning method of TensorFlow comes from Google to get the best model I can find.

Logistic regression, is a linear model for classification rather than regression. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function[6].

The implementation of logistic regression in scikit-learn can be accessed from class LogisticRegression. This implementation can fit binary, One-vs- Rest, or multinomial logistic

regression with optional L2 or L1 regularization.

Stochastic Gradient Descent is a simple yet very efficient approach to discriminative learning of linear classifiers under convex loss functions such as Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning[7].

SGD has been successfully applied to large-scale and sparse machine learning problems often encountered in text classification and natural language processing. Given that the data is sparse, the classifiers easily scale to problems with more than 10^5 training examples and more than 10^5 features.

Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features[8].

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represents classification rules.

In decision analysis a decision tree and the closely related influence diagram are used as a visual and analytical decision support tool, where the expected values (or expected utility) of competing alternatives are calculated.

Ensemble learning methods combine a set of models to produce an estimator that has better predictive performance than its individual components. A **random forest** is a collection of decision trees that have been trained on randomly selected subsets of the training instances and explanatory variables.

Support vector machinesare a set of supervised learning methods used for classification, regression and outliers detection. SVM requires that each data instance is represented as a vector of real numbers[9].

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side

of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features[10].

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of P(xi \mid y).In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering.

Besides the supposition of linear correlation, I also apply non-linear method without supposition.

In the context of machine learning,**hyperparameter optimization**or **model selection** is the problem of choosing a set of hyperparameters for a learning algorithm, usually with the goal of optimizing a measure of the algorithm's performance on an independent data set. Often **cross-validation** is used to estimate this generalization performance. Hyperparameter optimization contrasts with actual learning problems, which are also often cast as optimization problems, but optimize a loss function on the training set alone. In effect, learning algorithms learn parameters that model/reconstruct their inputs well, while hyperparameter optimization is to ensure the model does not overfit its data by tuning, e.g., regularization[11].

The traditional way of performing hyperparameter optimization has been **grid search** which is simply an exhaustive searching through a manually specified subset of the hyperparameter space of a learning algorithm. A grid search algorithm must be guided by some performance metric, typically measured by cross-validation on the training set or evaluation on a held-out validation set.

Since the parameter space of a machine learner may include real-valued or unbounded value spaces for certain parameters, **manually set bounds and discretization** may be necessary before applying grid search[12].

2.4 Benchmark

As mentioned before, there is one of the best predictive accuracy obtained using one separating plane in the 3-D space of Worst Area, Worst Smoothness and Mean Texture. Estimated accuracy 97.5% using repeated 10-fold cross validations. Classifier has correctly

diagnosed 176 consecutive new patients as of November 1995.

3. Methodology

3.1 Data Preprocessing

Import data from file of breast cancer file with pandas package which provides method 'pd.read_csv' to read csv file.

Rename each column with labels and features. Since I already know most features have weak connections, feature selection is reserved for further refinement of analyzing. Giving 30 features their names can also help me select meaningful features based on prior knowledge if I have.

Raw data shape is (569, 33) which means 569 rows and 33 imported features all have been read into form of dataframe. But there are **2 labels** and**actually only 30 feature** in description and I manually delete one wrongly read column from file.

Watch and reshape the form of data to show the exact what dataset contains. This is necessary process for continuing analyze data.

Normalize features by removing the mean and scaling to unit variance. Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using the transform method.

Outliers are detected and removed and there are 495 data left. Since there are 30 features and only 70 data removed if it contains any outliers in all features. Actually it is indeed small and can be accepted since on average there are 2 data removed because outliers in each features.

3.2 Implementation

General models without tunning

Load and applyeach algorithm. After I know the potential correlative connection between all features, I apply linear method first to see if it is best fit this dataset. What is more, I already find there is no data imbalance in dataset so I can standardize all features for less necessary calculation and more easy understandable meaning from distribution of percentile.

LogisticRegression, **SGDClassifier**, **SVC** and **GaussianNB** are all applied based on their

default setting. Based on the different theory of each classifier, results give direction for further improvement.

```
Accuracy of LR:  0.959568733154
             precision    recall  f1-score   support

     Benigh       0.99      1.00      0.99        85
  Malignant       1.00      0.97      0.99        39

avg / total       0.99      0.99      0.99       124

Accuracy of SGDC:  0.946091644205
             precision    recall  f1-score   support

     Benigh       0.94      1.00      0.97        85
  Malignant       1.00      0.87      0.93        39

avg / total       0.96      0.96      0.96       124

Accuracy of SVC:  0.940700808625
             precision    recall  f1-score   support

     Benigh       0.92      1.00      0.96        85
  Malignant       1.00      0.82      0.90        39

avg / total       0.95      0.94      0.94       124

Accuracy of GB:  0.940700808625
             precision    recall  f1-score   support

     Benigh       0.95      0.98      0.97        85
  Malignant       0.95      0.90      0.92        39

avg / total       0.95      0.95      0.95       124

Accuracy of DT:  1.0
             precision    recall  f1-score   support

     Benigh       0.96      0.96      0.96        85
  Malignant       0.92      0.92      0.92        39

avg / total       0.95      0.95      0.95       124
```

Fig6. Apply of five basic machine learning models with default parameters.

As the figure showed, we get the accuracy, precision, recall, and F1 score of each algorithm. None of algorithms are better than benchmark of previous study and Decision-Tree is obviously over-fitting. I will try to explore a basic linear TensorFlow method to see if there are more interpretable results.

3.3 Refinement

As the result table showed, logistic regression and support vector machine is obviously under the benchmark while decision tree is over-fitting. So I decide tuning hyperparameters of those three models, since logistic regression and decision tree are based on linear or non-linear supposition respectively and while support vector machine can choose either linear or non-linear kernel.

I apply grid search on each of three algorithm to tune related hyperparameter to optimize performance and find best numbers which suit this problem.

In grid search, I set 'scoring' to 'f1' which is most suitable for binary problem[13]. The 'verbose' is equal to 1 since this provide enough information for further presentation.

3.3.1 Logistic Regression refinement

Regularization in the fields of machine learning and inverse problems, refers to a process of introducing additional information in order to solve an ill-posed problem or to prevent overfitting[14].

Regularization. https://en.wikipedia.org/wiki/Regularization_%28mathematics%29#Regularization_in_statistics_and_machine_learning

In logistic regression, I should use regularization on minimizing error of training models. As this is a binary classification problem, I set 'solver' as default which is 'liblinear' and choose tune penalty and C value.

Penalty includes 'l1' and 'l2'.**L1-norm**is also known as least absolute deviations (LAD), least absolute errors (LAE). It is basically minimizing the sum of the absolute differences (S) between the target value (Yi) and the estimated values (f(xi))[15]:

$$S = \sum_{i=1}^{n} |y_i - f(x_i)|.$$

L2-norm is also known as least squares. It is basically minimizing the sum of the square of the differences (S) between the target value (Yi) and the estimated values (f(xi):

$$S = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

C is inverse of regularization strength and must be a positive float. Like in support vector machines, smaller values specify stronger regularization.

```
Best score: 0.980
Best parameters set:
        clf__C: 5
        clf__penalty: 'l2'
Accuracy: 0.991935483871
Precision: 0.988372093023
Recall: 1.0
             precision    recall  f1-score   support

          0       1.00      0.97      0.99        39
          1       0.99      1.00      0.99        85

avg / total       0.99      0.99      0.99       124
```

Fig.7 Refined results of logistic regression

Combine penalty and C together and apply grid search, I get the results which is 0.992 and that is much better than benchmark.

3.3.2 Decision Tree refinement and tree ensembles

In decision tree, there are multiple parameters. First, I set 'criterion' as 'gini' since all features are continuous attributes and 'gini' will tend to find the largest class, and 'entropy' tends to find groups of classes that make up ~50% of the data[16].

http://paginas.fe.up.pt/~ec/files_1011/week%2008%20-%20Decision%20Trees.pdf; http://paginas.fe.up.pt/~ec/files_1011/week%2008%20-%20Decision%20Trees.pdf

Then I decide to apply other hyperparameters including max_depth, min_samples_split and min_samples_leaf. All the explanation can be found in official documentation. I set those three range less than 10 or 20 since whole dataset is small and indeed it is good for choosing. The max_features are default using all 30 features in dataset.

```
Best score: 0.942
Best parameters set:
        clf__max_depth: 15
        clf__min_samples_leaf: 1
        clf__min_samples_split: 1
Accuracy: 0.951612903226
Precision: 0.964705882353
Recall: 0.964705882353
             precision    recall  f1-score   support

          0       0.92      0.92      0.92        39
          1       0.96      0.96      0.96        85

avg / total       0.95      0.95      0.95       124
```

Fig.8 Refined results of decision tree

Since accuracy of decision tree is 0.952 which is not better than benchmark, I try to apply this in ensemble learning to build random forest.

A **random forest** is a collection of decision trees that have been trained on randomly selected subsets of the training instances and explanatory variables. Random forests usually make predictions by returning the mode or mean of the predictions of their constituent trees; scikit-learn's implementations return the mean of the trees' predictions[17].

Beside three parameters already in grid search, I set n_estimators from 3 to 25 which is appropriate for the scale of this dataset.

```
Best score: 0.974
Best parameters set:
        clf__max_depth: 6
        clf__min_samples_leaf: 1
        clf__min_samples_split: 4
        clf__n_estimators: 10
Accuracy: 0.967741935484
Precision: 0.965517241379
Recall: 0.988235294118
             precision    recall  f1-score   support

          0       0.97      0.92      0.95        39
          1       0.97      0.99      0.98        85

avg / total       0.97      0.97      0.97       124
```

Fig.9 Refined results of random forest

Still, the accuracy of random forest is 0.968 which is also lower than benchmark. Since both non-linear method return results which are not better than benchmark and linear method, I do suppose this problem is fit for linear method.

3.3.3 Support Vector Machine refinement

Since support vector machine can be set different kernels and apply linear or non-linear method, I try both ways to see performance on this problem.

The **gamma** parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

The **C** parameter trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors.

I set those two parameters exponentially with several numbers inside the range. The reason for the exponential grid is that both C and gamma are scale parameters that act multiplicatively,

so doubling gamma is as likely to have roughly as big an effect (but in the other direction) as halving it. This means that if we use a grid of approximately exponentially increasing values, there is roughly the same amount of "information" about the hyper-parameters obtained by the evaluation of the model selection criterion at each grid point[18].

First, I set kernel as 'rbf'. In general, the **RBF kernel** is a reasonable first choice. This kernel nonlinearly maps
samples into a higher dimensional space so it, can handle the case when the relation between class labels and attributes is nonlinear. Furthermore, the linear kernel is a special case of RBF since the linear kernel with a penalty parameter C has the same performance as the RBF kernel with some parameters C and gamma[19].

```
Best score: 0.986
Best parameters set:
        clf__C: 10
        clf__gamma: 1
Accuracy: 0.983870967742
Precision: 0.977011494253
Recall: 1.0
             precision    recall  f1-score   support

          0       1.00      0.95      0.97        39
          1       0.98      1.00      0.99        85

avg / total       0.98      0.98      0.98       124
```

Fig.10 Refined results of SVM whit RBF kernel

Accuracy is 0.984 which is better than benchmark.

Secondly, I set kernel as 'linear' since SVM with linear kernel is usually comparable with logistic regression[20]. Indeed, in this dataset its performance is similar to logistic regression while I just need to tune C only.

```
Best score: 0.982
Best parameters set:
        clf__C: 25
Accuracy: 0.991935483871
Precision: 0.988372093023
Recall: 1.0
             precision    recall  f1-score   support

          0       1.00      0.97      0.99        39
          1       0.99      1.00      0.99        85

avg / total       0.99      0.99      0.99       124
```

Fig.11 Refined results of SVM whit linear kernel

This results is very similar to results of logistic regression and it is also 0.992 which is best now.

4. Results

4.1 Model Evaluation and Validation

In this section, the final model and any supporting qualities should be evaluated in detail. It should be clear how the final model was derived and why this model was chosen. In addition, some type of analysis should be used to validate the robustness of this model and its solution, such as manipulating the input data or environment to see how the model's solution is affected (this is called sensitivity analysis). Questions to ask yourself when writing this section:

In logistic regression and SVM, I get three results which are better than benchmark 0.975 by applying grid search with different combination of hyperparameters each model need and cross validation.

From the beginning of this program, I clear all the outliers and then normalized all features since it is necessary for some model optimization. Also, I split dataset into training and testing set by function of cross validation and I do got better performance with increase of 'cv' numbers which could split dataset into more subsets.

In each model, the iteration is the default times which is 100. That do keep robustness of model and parameters.

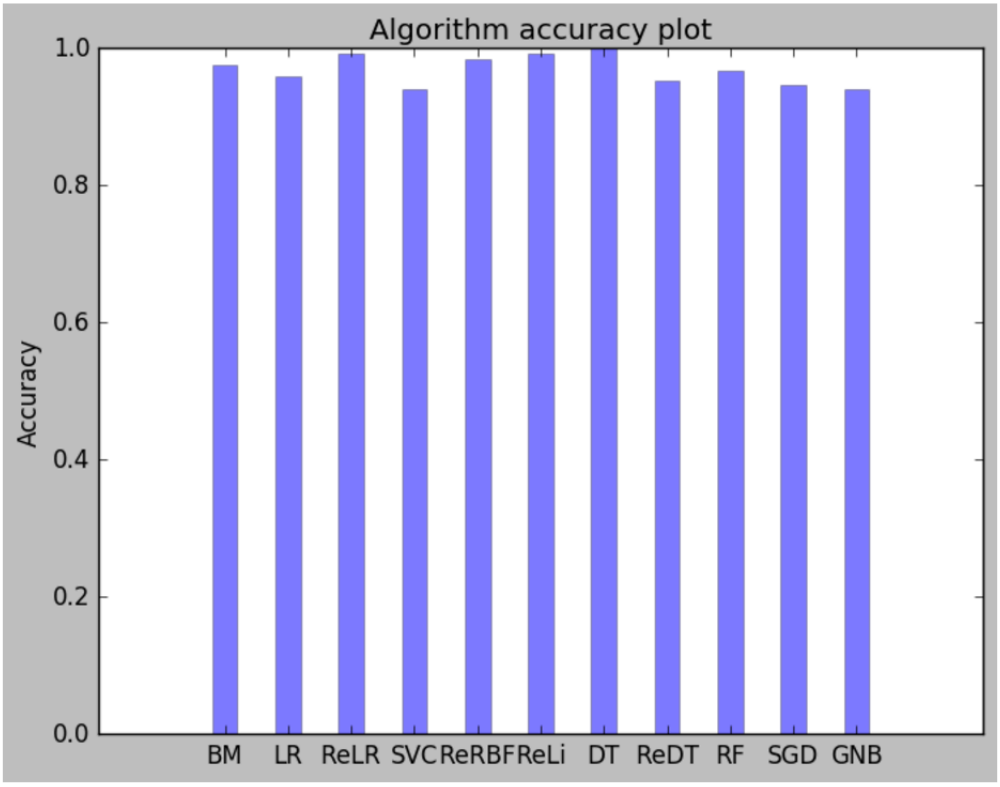


Fig.12 Accuracy of each model in plot

4.2 Justification

In summary, I conclude all accuracy a table:

	Model	Accuracy
0	ReSVClinear	0.992
1	ReLR	0.992
2	SVC	0.941
3	DT	1.000
4	ReDT	0.952
5	GaussianNB	0.941
6	Benchmark	0.975
7	ReSVCrbf	0.984
8	RF	0.968
9	LR	0.960
10	SGD	0.946

Fig.13 Accuracy of each model in table

As table showed that, there are 3 method is effectively superior to the benchmark the dataset provided, 0.975. The winning method are refined logistic regression, refined SVC with rbf kernel and refined SVC with linear kernel. Highest accuracy comes from refined logistic regression and refined SVC with linear kernel both are 0.984.

5. Conclusion

5.1 Reflection

Interesting aspects:

> Although features are only different aspects of characteristics of cancer cell images, accuracy suppose there are some linear relationship between all features and diagnosis.
>
> Besides the suppose of linear and non-linear relationship, SVC with linear kernel give similar performance like logistic regression although they operate in different ways.
>
> I use cross validation in 3 fold to omit unnecessary calculation. If I increase number of fold to 5, decision tree model can give better performance even better than benchmark.

Difficult aspects:

> I need to understand the background of each algorithm for what is in count of calculation.
>
> Understanding meaning of parameters should based on math knowledge

> Even classic algorithm can produce better performance than TensorFlow.

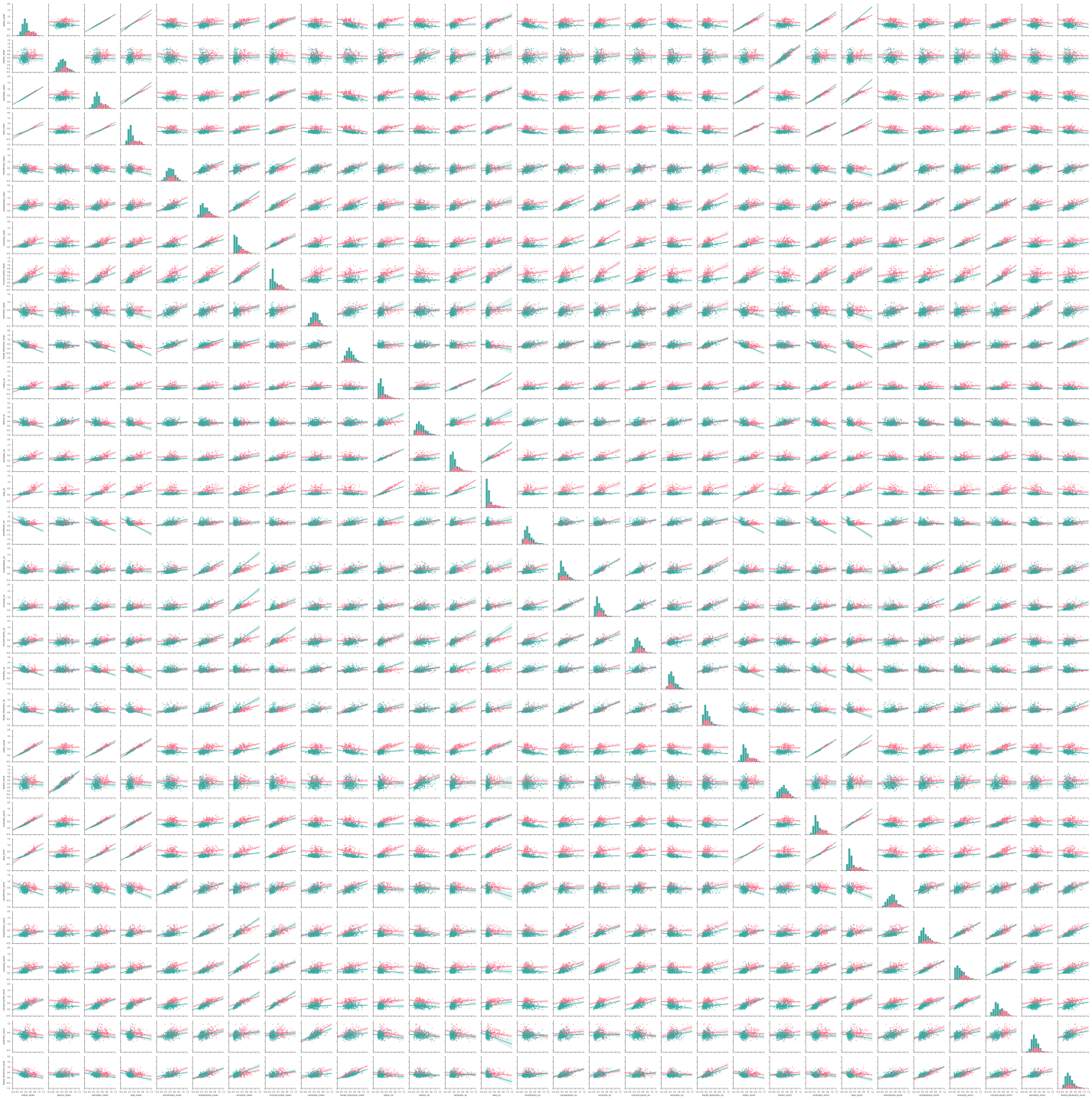
5.2 Improvement

In this problem, there may be more algorithm to apply on this dataset while I just use several of them. What is more, there are different parameters to define the process algorithm could process and I could try more.

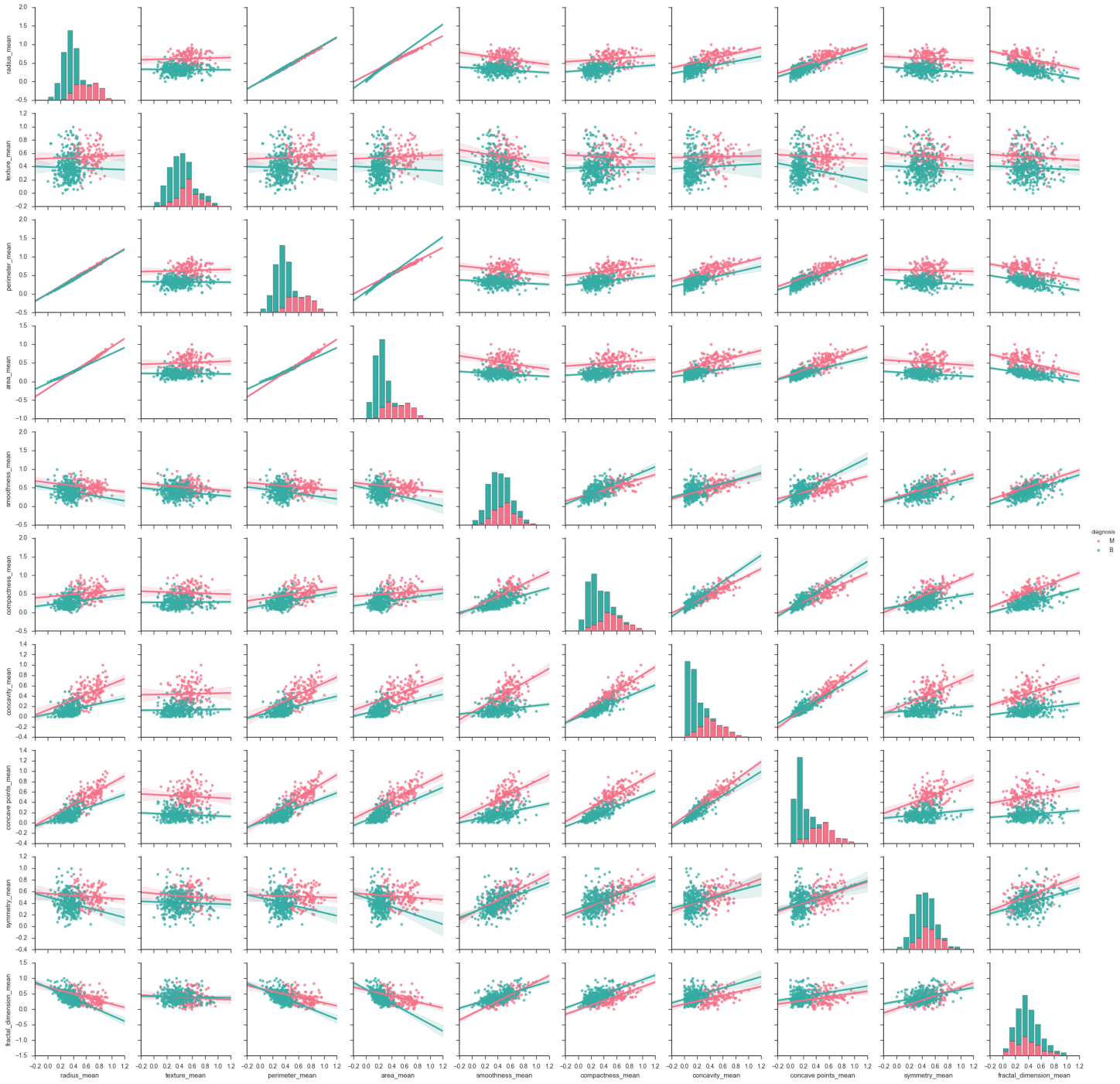
To define this problem is binary classification, one can use limited method. However, if anyone try to find the special connection of features between features or to select features and combine them, it is still another problem to solve.

In summary, I successfully apply classic algorithm with grid search and cross validation on this dataset and produce a perfect match with accuracy 0.992. I suppose this scale of data is enough to follow a patient and give accurate advice though the larger dataset is the better model will be. Also, there are a bright future to train model with larger dataset and get what I can know to support health industry future development to make a better world.

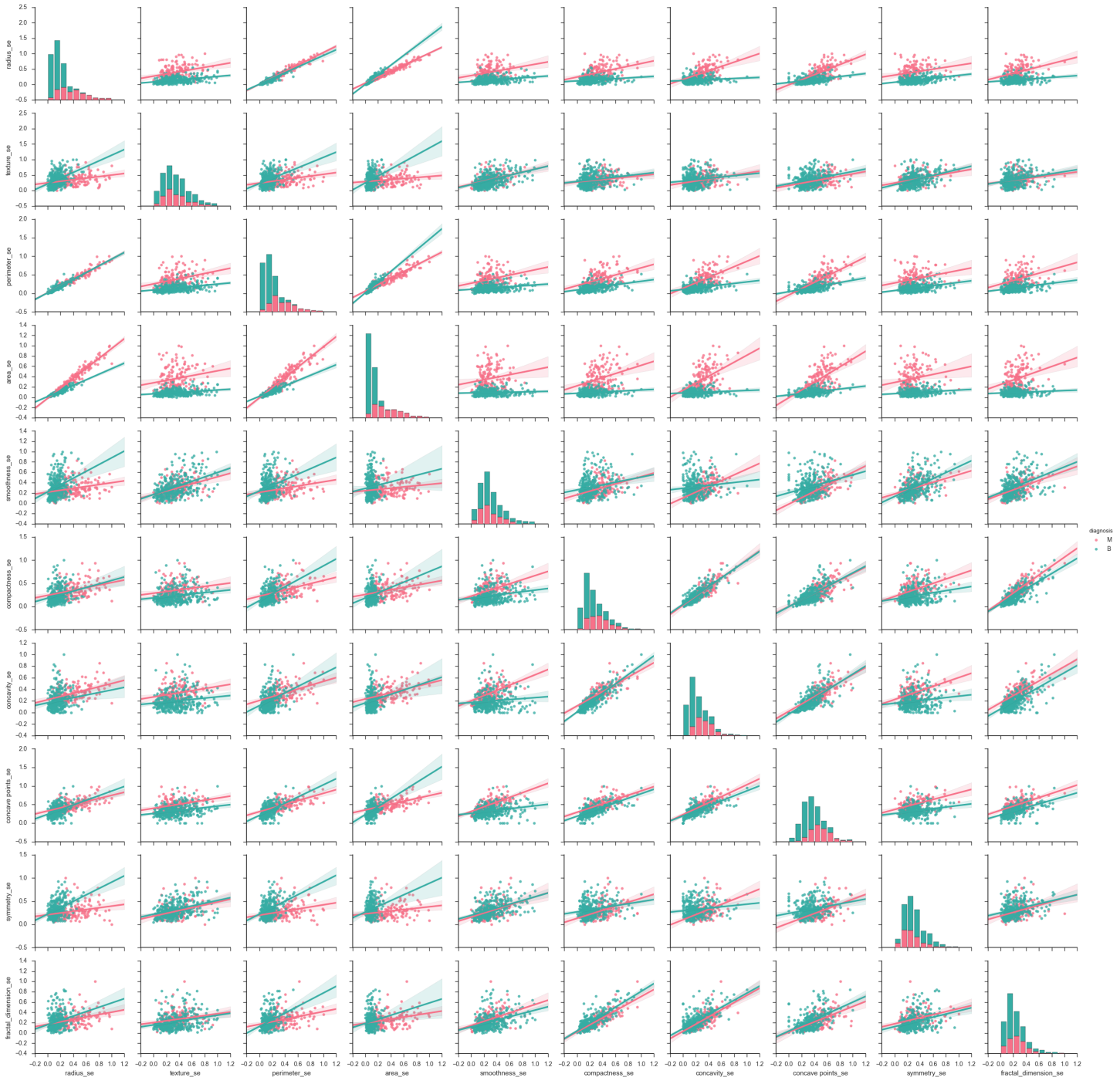
6.Supplementary



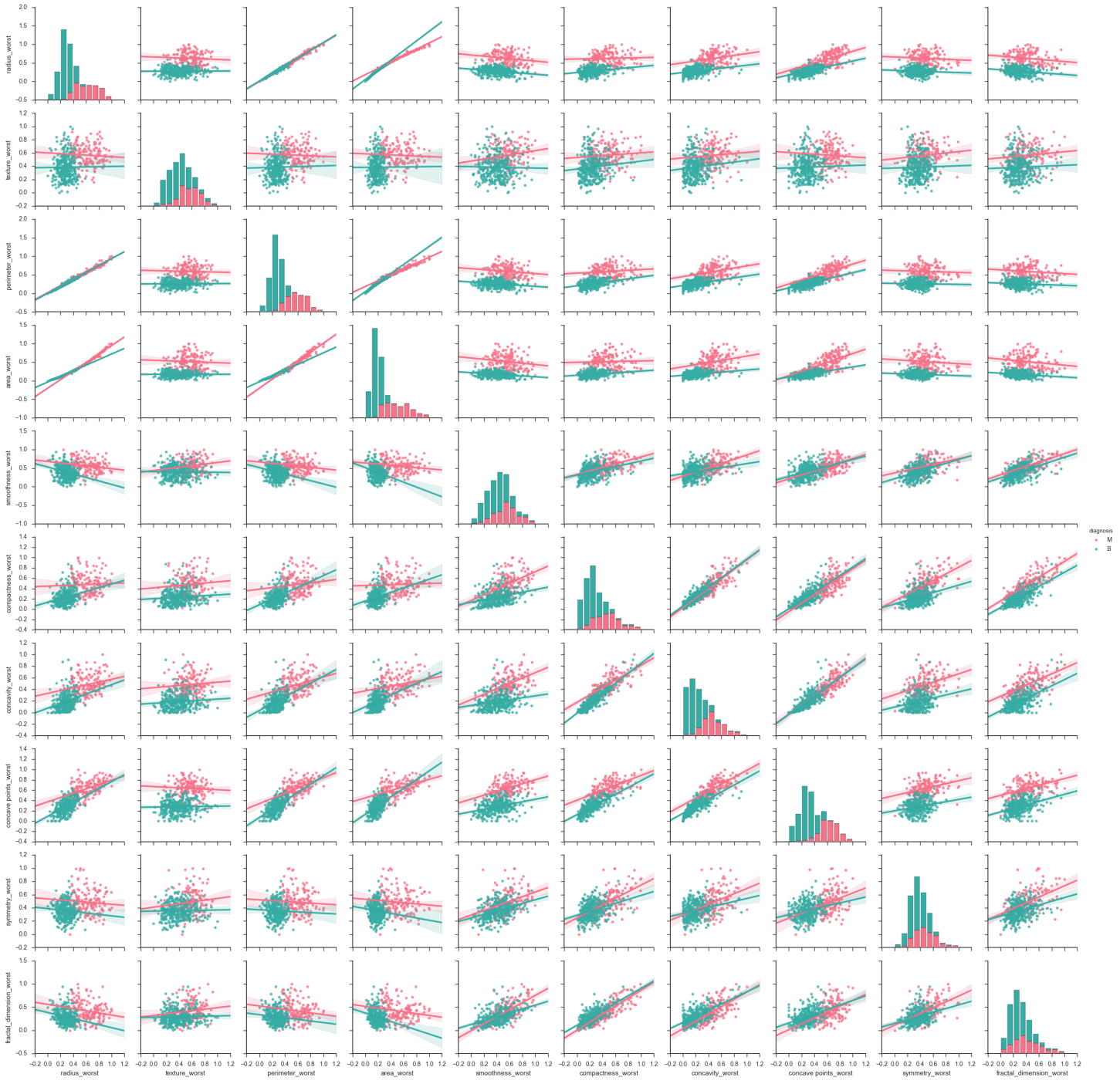
Sfig.1 Outliers_removed_30 features



Sfig.2 Outliers_removed_first_10 features



Sfig.3 Outliers_removed_second_10 features



Sfig.4 Outliers_third_10 features

---

1. Classification Accuracy is Not Enough: More Performance Measures You Can Use ↩

2. Outlier https://en.wikipedia.org/wiki/Outlier; http://stackoverflow.com/questions/23199796/detect-and-exclude-outliers-in-pandas-dataframe ↩

3. Binary classification, https://en.wikipedia.org/wiki/Binary_classification ↩

4. Normalization https://en.wikipedia.org/wiki/Normalization_(statistics)#cite_note-Dodge-1 ↩

5. Y. Yang, X. Liu, "A re-examination of text categorization", Proc. ACM SIGIR Conference, pp. 42–49, (1999). ↩

6. sklearn.linear_model.LogisticRegressi, http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression ↩

7. Stochastic Gradient Descent, http://scikit-learn.org/stable/modules/sgd.html ↩

8. Decision Trees, http://scikit-learn.org/stable/modules/tree.html ↩

9. Support vector machine, https://en.wikipedia.org/wiki/Support_vector_machine ↩

10. Naive Bayes, http://scikit-learn.org/stable/modules/naive_bayes.html ↩

11. Hyperparameter optimization. https://en.wikipedia.org/wiki/Hyperparameter_optimization ↩

12. Chin-Wei Hsu, Chih-Chung Chang and Chih-Jen Lin (2010). http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf ↩

13. Model evaluation: quantifying the quality of predictions. http://scikit-learn.org/stable/modules/model_evaluation.html#scoring-parameter ↩

14.

15. Differences between the L1-norm and the L2-norm. http://www.chioka.in/differences-between-the-l1-norm-and-the-l2-norm-least-absolute-deviations-and-least-squares/ ↩

16.

17. Master machine learning with scikit-learn. Gavin Hackeling. ↩

18. http://stats.stackexchange.com/questions/81537/gridsearch-for-svm-parameter-estimation ↩

19. http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf ↩

20. Where is it best to use svm with linear kernel?. http://stackoverflow.com/questions/20566869/where-is-it-best-to-use-svm-with-linear-kernel ↩