Breast Cancer Prediction

Group 17: Nathan Han, Zeqing Li, Siyan Wu, Li Zhu

Introduction

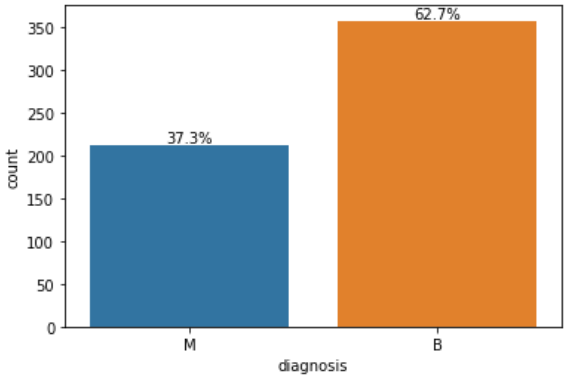
We analyzed breast cancer data, with diagnoses as labels and tumor features as variables. Our goal is to predict whether tumors are malignant. We also studied the following questions:

Which and how many variables to include?
Which variable contributes the most?
Which model predicts the best?

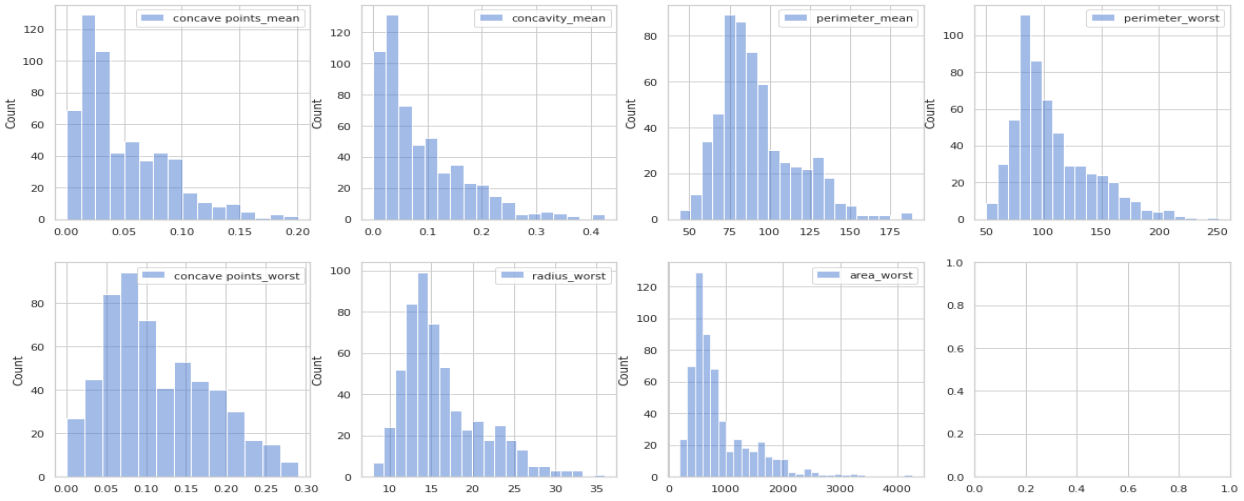
We adopted PCA and random forest to transform data and used grid search and f-beta to select models.
The top 7 features selected by the Random Forest with logistic regression are the best model. The most contributing feature is concave_points_mean.

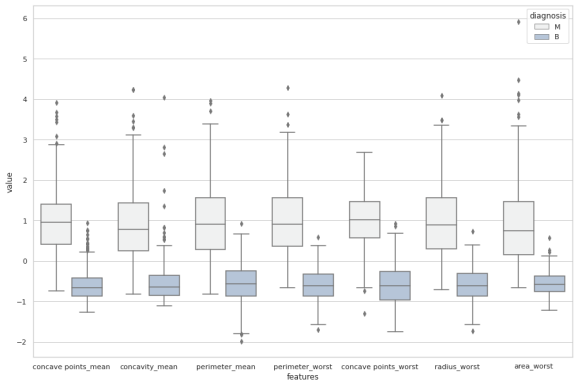
Dataset Information

We adopted Breast Cancer Wisconsin Diagnostic Data Set from UCI Machine Learning Repository. It includes 569 lines with 31 features. Diagnoses are used as data labels, with malignant as one and benign as 0. The data is balanced.



The remaining 30 features are “mean”, “standard error” and “worst” data. In the selected data graphed below, concave_points_worst is the largest value for concave portions of the contour; texture_se is the SE for the standard deviation of gray-scale values.
We see right-skewed distributions. Most points are scattered left, while some long-tailed ones are to the right.



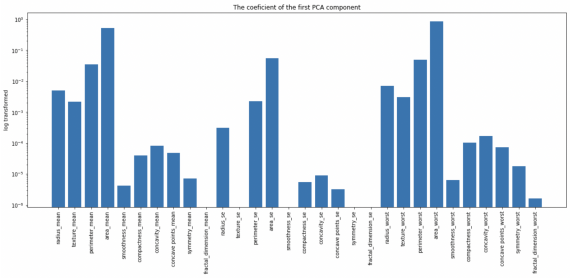


The boxplots with diagnosis labels showed that most outliers are malignant.
Since 30 features are too complex to fit models, we apply data transformation.

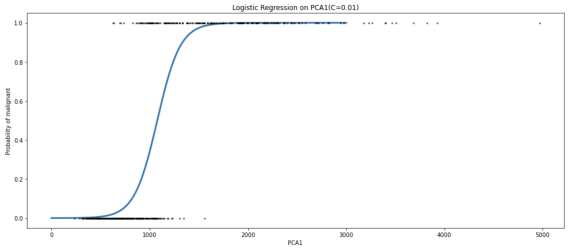
Data Transformation

PCA

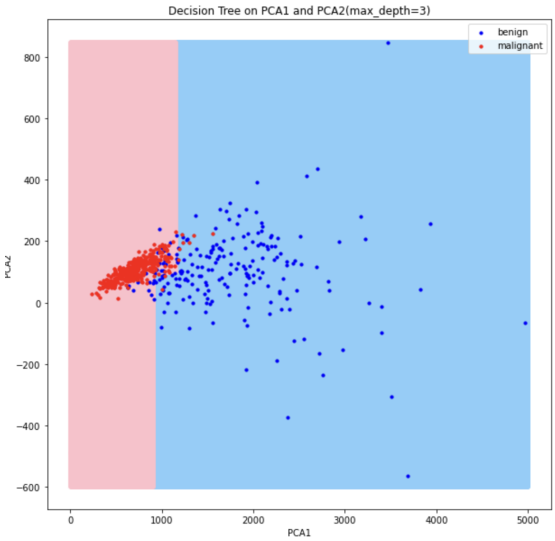
We used PCA to generate four components. Since the sum of the first two explained variance ratios is over 0.99, we use one or two components. The following graph shows the coefficients of the first PCA component. The ten features that are over 10^-3 are more significant than others.



We fitted logistic regression with C = 0.01 on the PCA1 and made a graph as follows.



We fitted the decision tree on training data with the max_depth is 3 on two PCA components and plotted all data. Only a few points are outliers.



The model was trained on test data. The model fits very well through plotting, and the vertical dividing line makes a small change on PCA2.

Random Forest

sklearn.feature_selection.RFECV

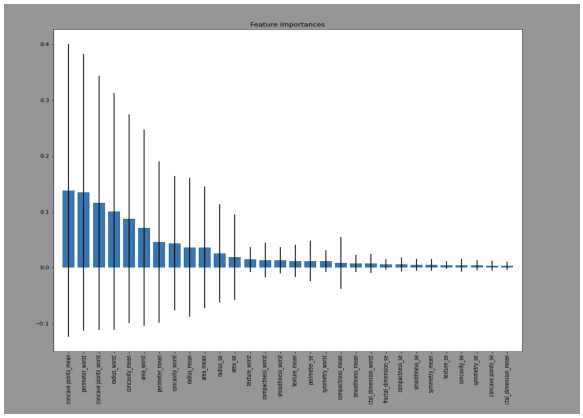
class sklearn.feature_selection.**RFECV**(*estimator*, *, *step=1*, *min_features_to_select=1*, *cv=None*, *scoring=None*, *verbose=0*, *n_jobs=None*, *importance_getter='auto'*) [source]

Recursive feature elimination with cross-validation to select features.

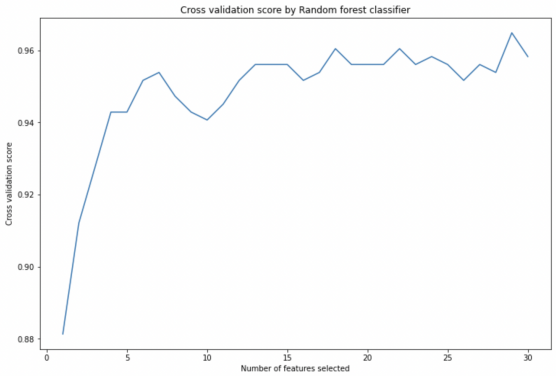
See glossary entry for cross-validation estimator.

Read more in the User Guide.

Because features from PCA have no actual meaning, we use the random forest to make feature selection. With RFECV class, we can run 100 default times of the decision tree to get the mean importance of features.



We visualized feature importances ranking from the highest to the lowest, getting from the random forest. The blue bar is the mean of feature importance scores, while the black line is the standard error. The standard error is significant because of each decision tree's unbalanced feature selection.



When feature numbers(x) are equal to or greater than seven, the cross-validation score does not change much. Therefore, it shows the top 7 features are the most important.

Hyperparameter Tuning

In hyperparameter tuning, we used grid search and F-beta score for the scoring strategy.

Grid Search

We include linear and RBF SVM, logistic regression, decision tree, and KNN models in the grid search.

PCA

By applying grid search on one and two PCA components, the best score is 0.947, and the model is Logistic Regression(C=0.01) with one component.

Random Forest

By applying grid search with the top seven features, we got the best score of 0.965, while the best model is Logistic Regression when C is 100.

Reduce False Positive Rate

sklearn.metrics.fbeta_score

`sklearn.metrics.fbeta_score(y_true, y_pred, *, beta, labels=None, pos_label=1, average='binary', sample_weight=None, zero_division='warn')` [source]

Compute the F-beta score.

The F-beta score is the weighted harmonic mean of precision and recall, reaching its optimal value at 1 and its worst value at 0.

The `beta` parameter determines the weight of recall in the combined score. `beta < 1` lends more weight to precision, while `beta > 1` favors recall (`beta -> 0` considers only precision, `beta -> +inf` only recall).

To reduce the false positive rate, we add make_scorer to adjust fbeta_score into the grid search. The beta parameter determines the recall weight in the combined score, beta less than 1 lends more weight to precision.

- Top 7 features from the random forest:

	Beta=0.7			Beta=0.5		
	classifiers	best_params_	best_score	classifiers	best_params_	best_score
0	SVC()	{'C': 1000, 'kernel': 'linear'}	0.935	SVC()	{'C': 100, 'kernel': 'linear'}	0.905
1	LogisticRegression(max_iter=50000)	{'C': 100}	0.967	LogisticRegression(max_iter=50000)	{'C': 100}	0.979
2	DecisionTreeClassifier(criterion='entropy')	{'max_depth': 5}	0.905	DecisionTreeClassifier(criterion='entropy')	{'max_depth': 3}	0.941
3	KNeighborsClassifier()	{'n_neighbors': 4}	0.948	KNeighborsClassifier()	{'n_neighbors': 4}	0.968

- Top 1 features from PCA:

	Beta=0.7			Beta=0.5		
	classifiers	best_params_	best_score	classifiers	best_params_	best_score
0	SVC()	{'C': 1, 'kernel': 'rbf'}	0.948	SVC()	{'C': 1, 'kernel': 'rbf'}	0.968
1	LogisticRegression(max_iter=500)	{'C': 0.01}	0.948	LogisticRegression(max_iter=500)	{'C': 0.01}	0.968
2	DecisionTreeClassifier(criterion='entropy')	{'max_depth': 1}	0.948	DecisionTreeClassifier(criterion='entropy')	{'max_depth': 1}	0.968
3	KNeighborsClassifier()	{'n_neighbors': 5}	0.905	KNeighborsClassifier()	{'n_neighbors': 5}	0.905

Therefore, we tried beta equals 0.5 and 0.7. From the above two tables, the best model is the logistic model (C:100) with the Top 7 features

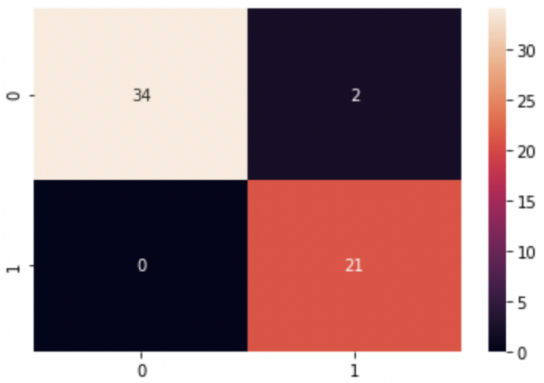
from the random forest with the best score of 0.979 when beta=0.5.

Performance Assessment

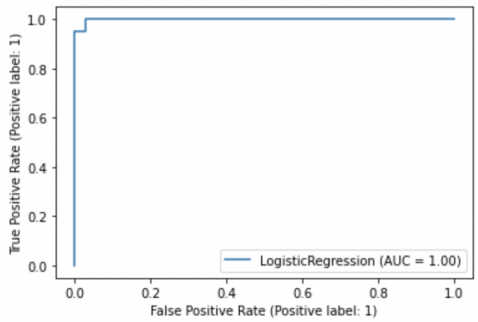
```
Feature selected by PCA:

['area_mean',
 'area_se',
 'area_worst',
 'perimeter_mean',
 'perimeter_se',
 'perimeter_worst',
 'radius_mean',
 'radius_worst',
 'texture_mean',
 'texture_worst']
```

Four features were chosen simultaneously by comparing the features selected by the two methods. They are 'area_worst', 'perimeter_mean', 'perimeter_worst', and 'radius_worst'.




```
precision=0.913, recall=1.0, accuracy=0.965
Area under ROC curve on test data is 0.972.
```



Using Logistic Regression (C: 100) with the top 7 features selected from Random Forest, the accuracy score for test data is 0.965, the precision score is 0.913, and the AUC is 0.972.

Conclusion

The best model is the top 7 features selected by the Random Forest with Logistic Regression (C=100). The feature that affects the most is concave points_mean. We can improve the work by finding a simpler model with fewer features and no false positives but trading off some accuracy.

Contribution:

Member	Proposal	Coding	Presentation	Report
Nathan Han	1	1	1	1
Zeqing Li	1	1	1	1
Siyan Wu	1	1	1	1
Li Zhu	1	1	1	1