FNA Breast Cancer Diagnosis with Machine Learning

Introduction

Fine needle aspiration is a type of biopsy procedure where a thin needle is inserted into an area of ab normal-appearing tissue or body fluid.

As with other types of biopsies, the sample collected during fine needle aspiration can help make a d iagnosis or rule out conditions such as cancer.

Generally, pathologists use information from the collected abnormal sample to determine if the abno rmality is cancerous or not and if it's not they determine how close it is to becoming cancerous. Ther e is also a grade(1-3) that determines the severity of the tumor where each grade lies in a score range based on how similar the tumor is to a normal cell.

After the sample is tested/processed, a doctor comes to a conclusion about the tumor considering a l ot of factors from the pathology report. Some of these features include the properties of cell nuclei w hich are studied under microscope and these are the features used in the following analysis.

The notion of this exercise is to build a prediction model that would best predict the diagnosis of a given FNA sample. In that regard, we were given a set of 30 features that represent attributes like radius, symmetry, texture among others. Each of the properties like area are repeated thrice once for mean, Standard error and Worst (because the tissue sample is sliced into thin layers to be observed under microscope we get multiple areas for one tissue and hence the need for aggregation). Most of these attributes are readily understood as important but there's one that isn't very obvious, Fractal Dimension or Fdimension for short. Cell tissues are physical structures that repeat themselves and this repetition makes them fractal(any geometric shape that extends/ consists of similar structures multiple times like a snowflake).

Various methods have been used to identify suitable candidates for the predictive model. An analysis of variances of the original whole column vs variance in each group when split by Diagnosis, Logistic Regression, Forward Selection, Recursive Feature Elimination with respect to multiple algorithms, and penalty variants(Lasso and Ridge). To keep the report concise the extensive work on the aforementioned is only included in part, however, important results are all included for reference.

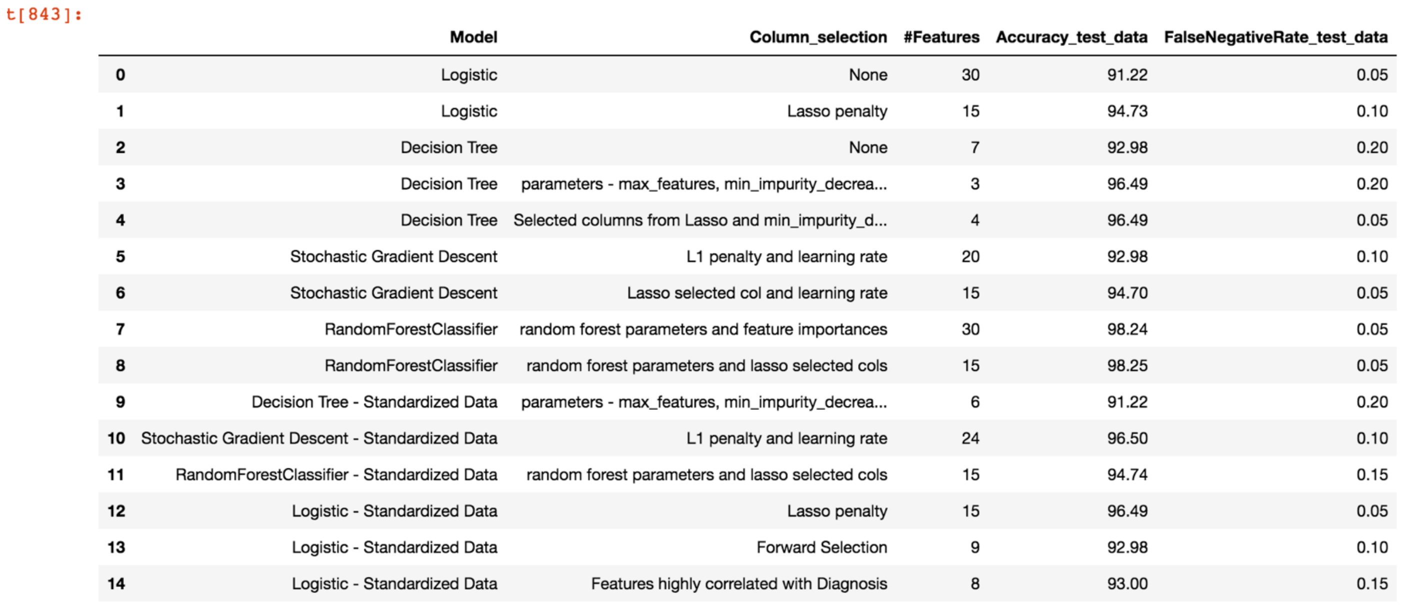
From a multitude of resources, it was found that Fractal Dimension is a measure that had high expectations to differentiate cancerous cells from the others and hence a considerable amount of time was spent on looking at the interactions not only between Fdimension (M, SE and W) and Diagnosis but for the others as well. Not so surprisingly, there's a substantial overfit between FdimensionSE and Diagnosis and hence had to be removed from the Logistic Model. Moreover, the LogisticRegression variant of Statsmodels also identified this relation as too good. Most of the other models included a variant F dimension (M or W) but not SE which proved that FdimensionSE isn't as great for a prediction model.

Any given model is expected to solve the problem well and we would most definitely like to measure the effectiveness of the model in terms of some performance metrics. As such, the most common metric for a classification problem is Accuracy. But, considering the fact that this data is related to health we need to decide on the metric carefully. It is known that accuracy isn't a great metric when the classes are not very well balanced. For this dataset the classes are not drastically imbalanced (62.7 for B and 37.3 for M) but still we would want to reduce the number of false negatives so as to identify as many people as possible – that have a Malignant tumor. I have tried to reduce the false negatives as much as possible while trying to maintain a high accuracy rate and hence both accuracy and false negative rate are the performance metrics for this exercise

FNA Breast Cancer Diagnosis with Machine Learning

For the purposes of this report, analysis on Logistic Regression only started out as a means to determine the ideal candidates for a predictive model but it proved to be far more than that for the dataset in question.

Below attached is the results summary of the most effective models, information regarding which will be discussed further in this report.

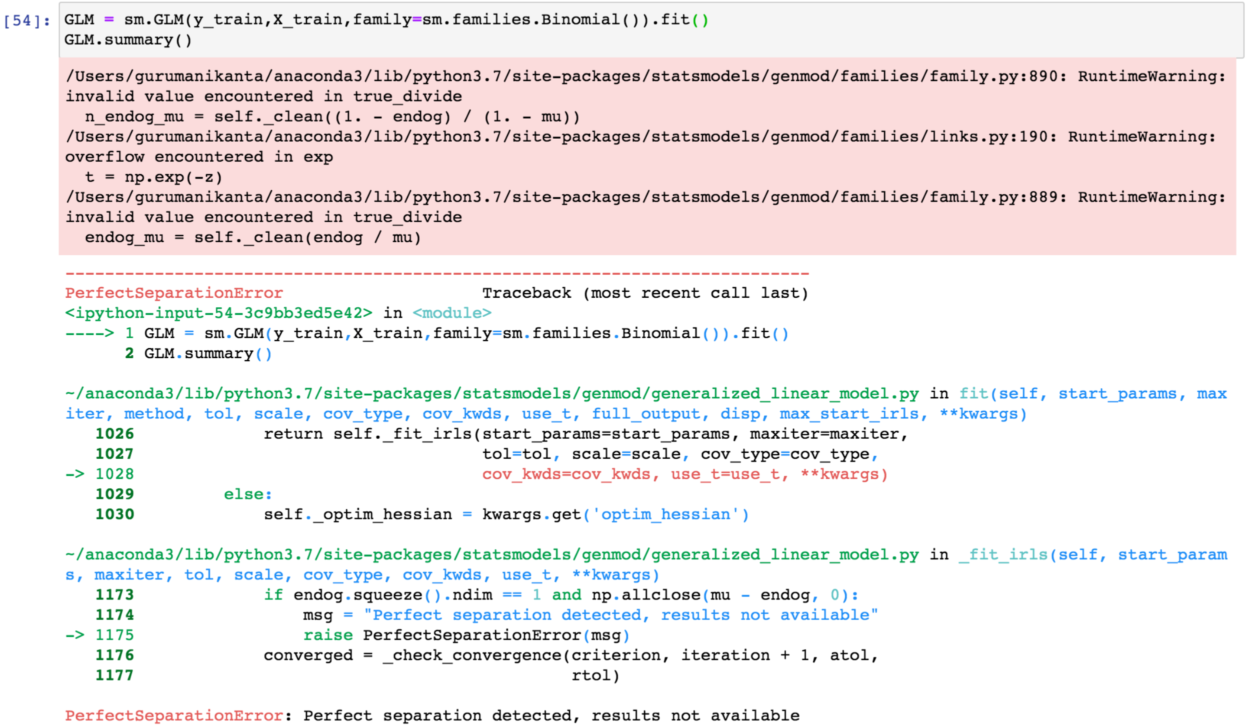


t[843]:

	Model	Column_selection	#Features	Accuracy_test_data	FalseNegativeRate_test_data
0	Logistic	None	30	91.22	0.05
1	Logistic	Lasso penalty	15	94.73	0.10
2	Decision Tree	None	7	92.98	0.20
3	Decision Tree	parameters - max_features, min_impurity_decrea...	3	96.49	0.20
4	Decision Tree	Selected columns from Lasso and min_impurity_d...	4	96.49	0.05
5	Stochastic Gradient Descent	L1 penalty and learning rate	20	92.98	0.10
6	Stochastic Gradient Descent	Lasso selected col and learning rate	15	94.70	0.05
7	RandomForestClassifier	random forest parameters and feature importances	30	98.24	0.05
8	RandomForestClassifier	random forest parameters and lasso selected cols	15	98.25	0.05
9	Decision Tree - Standardized Data	parameters - max_features, min_impurity_decrea...	6	91.22	0.20
10	Stochastic Gradient Descent - Standardized Data	L1 penalty and learning rate	24	96.50	0.10
11	RandomForestClassifier - Standardized Data	random forest parameters and lasso selected cols	15	94.74	0.15
12	Logistic - Standardized Data	Lasso penalty	15	96.49	0.05
13	Logistic - Standardized Data	Forward Selection	9	92.98	0.10
14	Logistic - Standardized Data	Features highly correlated with Diagnosis	8	93.00	0.15

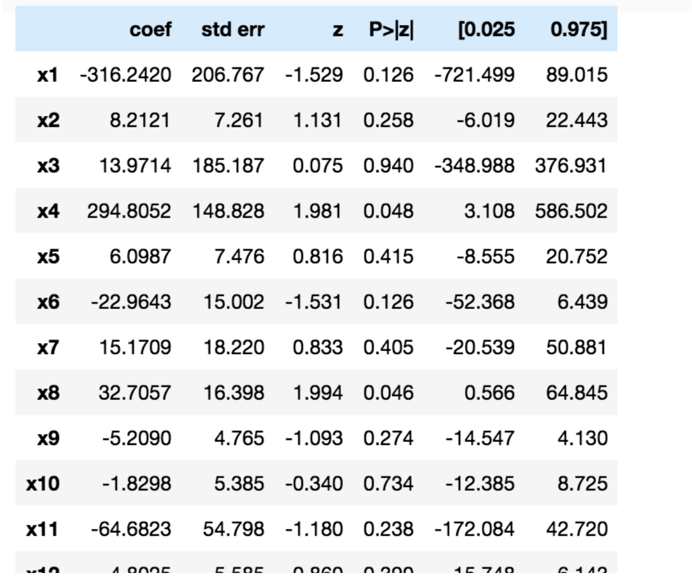
Model Development and Results

The Process started out with no preprocessing or feature selection. A Logistic Model had been built on the raw data as is. There weren't any null/missing values hence requirement for cleaning was also mitigated.
Data was split into training and testing at 0.85 and 0.15
The above training resulted in a PerfectSeperationError



```
[54]: GLM = sm.GLM(y_train,X_train,family=sm.families.Binomial()).fit()
      GLM.summary()

/Users/gurumanikanta/anaconda3/lib/python3.7/site-packages/statsmodels/genmod/families/family.py:890: RuntimeWarning:
invalid value encountered in true_divide
  n_endog_mu = self._clean((1. - endog) / (1. - mu))
/Users/gurumanikanta/anaconda3/lib/python3.7/site-packages/statsmodels/genmod/families/links.py:190: RuntimeWarning:
overflow encountered in exp
  t = np.exp(-z)
/Users/gurumanikanta/anaconda3/lib/python3.7/site-packages/statsmodels/genmod/families/family.py:889: RuntimeWarning:
invalid value encountered in true_divide
  endog_mu = self._clean(endog / mu)

---------------------------------------------------------------------------
PerfectSeparationError                    Traceback (most recent call last)
<ipython-input-54-3c9bb3ed5e42> in <module>
----> 1 GLM = sm.GLM(y_train,X_train,family=sm.families.Binomial()).fit()
      2 GLM.summary()

~/anaconda3/lib/python3.7/site-packages/statsmodels/genmod/generalized_linear_model.py in fit(self, start_params, max
iter, method, tol, scale, cov_type, cov_kwds, use_t, full_output, disp, max_start_irls, **kwargs)
   1026                 return self._fit_irls(start_params=start_params, maxiter=maxiter,
   1027                                       tol=tol, scale=scale, cov_type=cov_type,
-> 1028                                       cov_kwds=cov_kwds, use_t=use_t, **kwargs)
   1029         else:
   1030             self._optim_hessian = kwargs.get('optim_hessian')

~/anaconda3/lib/python3.7/site-packages/statsmodels/genmod/generalized_linear_model.py in _fit_irls(self, start_param
s, maxiter, tol, scale, cov_type, cov_kwds, use_t, **kwargs)
   1173             if endog.squeeze().ndim == 1 and np.allclose(mu - endog, 0):
   1174                 msg = "Perfect separation detected, results not available"
-> 1175                 raise PerfectSeparationError(msg)
   1176             converged = _check_convergence(criterion, iteration + 1, atol,
   1177                                            rtol)

PerfectSeparationError: Perfect separation detected, results not available
```

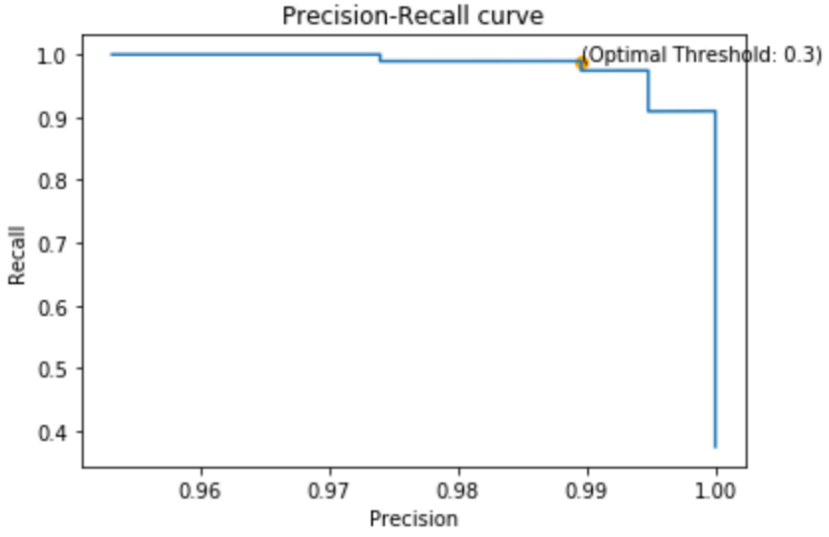
An iterative method similar to forward selection has been employed which adds one feature at a time into the dataframe and tries to train the model, once the column that resulted in the above error is identified it wouldn't be included in the next steps. That being said, the perfect separation error doesn't arise with SkLearn's Logistic model

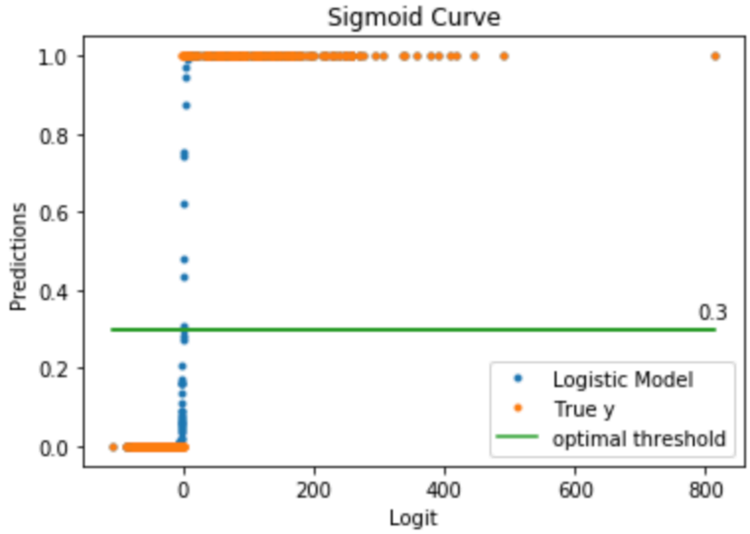


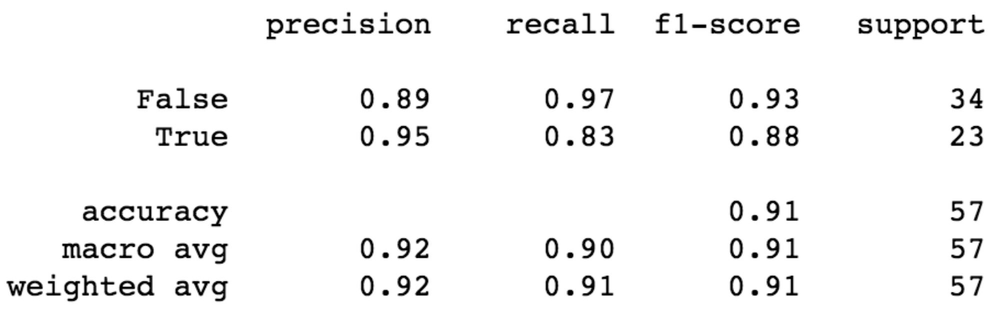
	coef	std err	z	P>|z|	[0.025	0.975]
x1	-316.2420	206.767	-1.529	0.126	-721.499	89.015
x2	8.2121	7.261	1.131	0.258	-6.019	22.443
x3	13.9714	185.187	0.075	0.940	-348.988	376.931
x4	294.8052	148.828	1.981	0.048	3.108	586.502
x5	6.0987	7.476	0.816	0.415	-8.555	20.752
x6	-22.9643	15.002	-1.531	0.126	-52.368	6.439
x7	15.1709	18.220	0.833	0.405	-20.539	50.881
x8	32.7057	16.398	1.994	0.046	0.566	64.845
x9	-5.2090	4.765	-1.093	0.274	-14.547	4.130
x10	-1.8298	5.385	-0.340	0.734	-12.385	8.725
x11	-64.6823	54.798	-1.180	0.238	-172.084	42.720
x12	-4.8025	5.585	-0.860	0.390	-15.748	6.143

Though the model was successfully trained it had it’s shortcomings: Drastic values for coefficients, around 90% accuracy, great false negative rate on test set and too many columns. So the next part focuses on identifying suitable candidate columns for the final model.

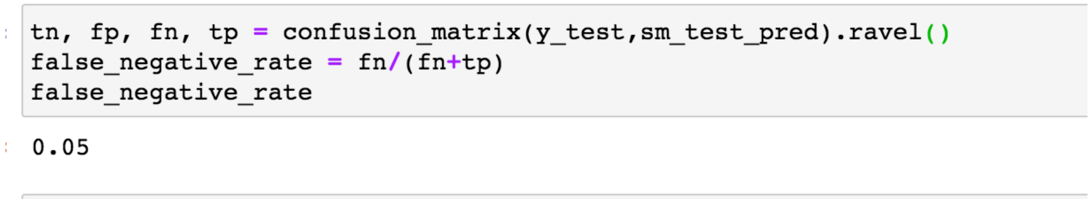
- Logistic Regression:
    - Without any preprocessing and column selection the logistic model there hasn’t been any overfit
    - accuracy = 91.22% but the false negative rate was great at 5%
    - The problem was that there isn’t great balance between accuracy and fnr
    - As seen below, the optimal threshold was calculated using the precision-recall curve since we’re interested in maintaining a balance
    - The sigmoid curve for this model is also attached along with the separator at optimal thereshold
    - The precision recall table is also included which shows a great level of precision for positive class and that’s a plus since the concentration is to reduce fnr







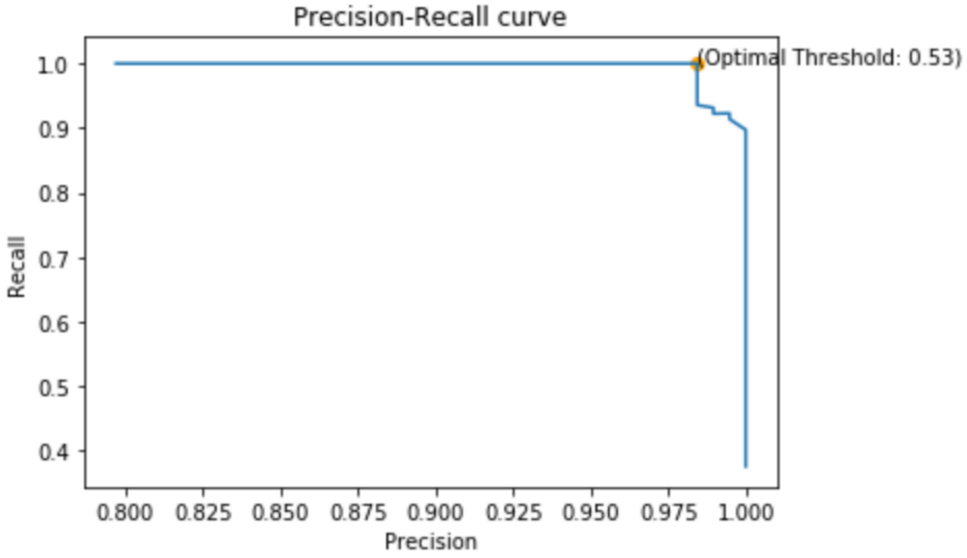
```
              precision    recall  f1-score   support

       False       0.89      0.97      0.93        34
        True       0.95      0.83      0.88        23

    accuracy                           0.91        57
   macro avg       0.92      0.90      0.91        57
weighted avg       0.92      0.91      0.91        57
```

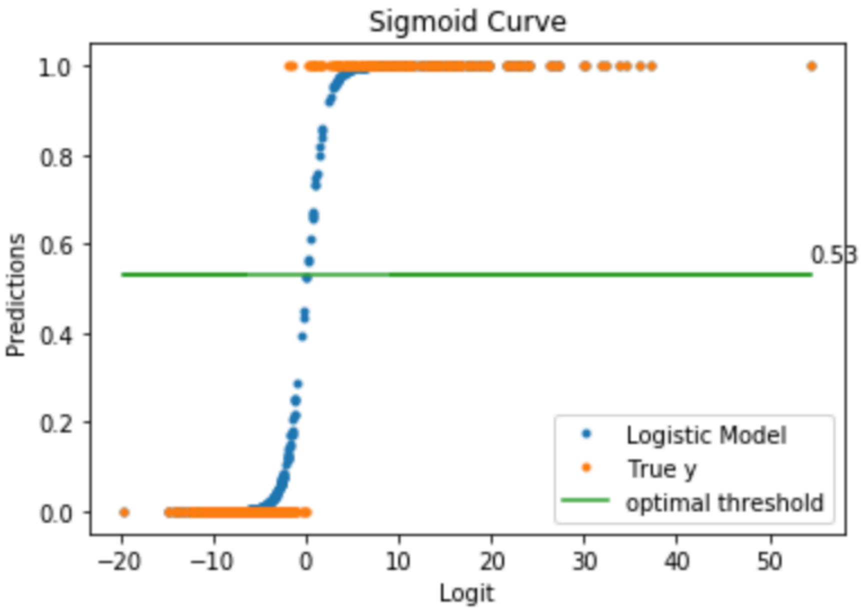


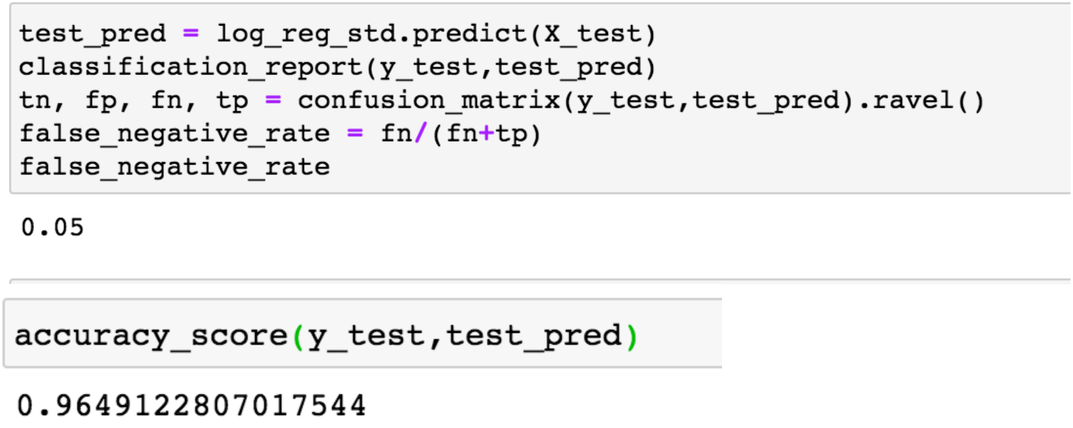
```
tn, fp, fn, tp = confusion_matrix(y_test,sm_test_pred).ravel()
false_negative_rate = fn/(fn+tp)
false_negative_rate
```

```
0.05
```

- Forward selection:
  - Forward selection was applied to Logistic model with the elimination criteria of AIC
  - This resulted in a set of 9 features having 99% accuracy on training set
  - The overfit was evident after testing on the test set – accuracy = 92.98% and false negative rate = 10%
  - Forward selection was applied another time after removing features that aren’t correlated or have very low correlation with Diagnosis which once again hadn’t been successful with false negative rate 15%
  - The above procedure has been followed for forward selection in Logistic as well
- Logistic Regression with Lasso Penalty:
  - Lasso Logistic regression had a test accuracy 94.73% and fnr = 10%
  - Lasso Logistic with standardized data had an improvement in the accuracy and fnr at 96.49% and 5%
  - Below are the precision-recall and sigmoid curves







```
test_pred = log_reg_std.predict(X_test)
classification_report(y_test,test_pred)
tn, fp, fn, tp = confusion_matrix(y_test,test_pred).ravel()
false_negative_rate = fn/(fn+tp)
false_negative_rate
```

```
0.05
```

```
accuracy_score(y_test,test_pred)
```

```
0.9649122807017544
```

- Recursive Feature Elimination:
  - Recursive feature elimination had been tried with Logistic, DecisionTree, RandomForest and StochasticGradientDescent
  - The results for each of those models aren't explicitly included since they aren't as effective as the others
  - However, there was a set of columns that remained in all of the four models mentioned above and those were also a part of the other models as well.
  - This could indirectly translate to saying that those attributes are probably the most significant. A snippet – ConcavenessM, ConcavityW, TextureW, SmoothnessW, and RadiusW
- Models with parameter tuning:
  - Decision Trees had the most effective feature elimination. Out of the 5 models built, 4 of them had 7 or less features. 2 of them had 4 or les features.
  - The most interesting aspect was that this reduction in number of features didn't result in worse performance
  - The Best performing decision tree had 96.49% accuracy and 5% with only 4 features which is a contender not just for best DecisionTree model but also the best model from the experiment

  - The parameters that were tuned for Decision tree: min_samples_split, min_samples_leaf, min_impurity_decrease in multiple combinations
  - SGD had also been tuned with parameters of which learning rate played a crucial role, a change of magnitude 0.001 also changed the accuracy and fnr
  - Best model of SGD is almost on par with that of Decision tree but with the number of features being 15
  - Random Forest by far had the best performance in terms of accuracy at 98% on test set with 15 columns which we selected from the Lasso Logistic regression and 5% fnr
  - Random Forest was also trained on multiple variations

Discussions, Limitations and Conclusion

- The discussion on Fractal Dimension as mentioned in the introduction
  - https://www.cancer.gov/about-cancer/diagnosis-staging/prognosis/tumor-grade-fact-sheet
  - https://appmicro.springeropen.com/articles/10.1186/s42649-021-00055-w
  - https://www.researchgate.net/publication/347965778_Detection_of_cancer_stages_through_fractal_dimension_analysis_of_tissue_microarrays_TMA_via_optical_transmission_microscopy
  - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6349609/
- In a hypothetical case where this solution is being used as preliminary step in concluding the state of the cancerous cell, when a patient is diagnosed with Malignant wrongfully the next series of tests would determine the actual conclusion. This justifies the importance of False Negative over all others as far as this exercise is concerned
- The models have not been tried on a huge dataset so there could still be some caveats to consider and hence the scope is only limited to analysis
- There's a lot of scope in improving Decision tree and SGD but before getting there we have still have to dig in deeper to identify any possible overfit for Decision Trees
- Random Forest had by far been the best in terms of metrics but it had more features than other best performing models like Decision tree
- There are more variants for almost all models that can be tried, using techniques like Cross-validation, random search and grid search for parameter tuning, more sophisticated methods like Neural Networks, Gradient Boosting trees and possibly voting classifiers