Kamen Zhekov
Vrije Universiteit Brussel
Techniques of AI Project Report

SVM-ASSISTED BREAST CANCER DETECTION

TECHNIQUES OF AI PROJECT 2019

TABLE OF CONTENTS

Contents

PROJECT REPORT

Introduction

PROJECT DESCRIPTION

The goal of the project is to implement a machine-learning assisted algorithm that helps detect whether a patient has breast cancer or not, based on the analysis of characteristics extracted from the 3D models of microcalcifications present in the patient's breast tissue.

IMPLEMENTATION

For the realization of the project, an SVM was chosen as the machine-learning method used to classify the microcalcifications. The model was trained and tested with the given data set of 3562 microcalcifications, each having 150 characteristics.

DIAGNOSING THE PATIENT

Once the algorithm was implemented and well-tested, the program diagnoses the patient using the same characteristics as the ones used in the training data set. The diagnosis is based on the number of microcalcifications detected as malignant, and the SVM's prediction performance in the various tests.

A visual representation is shown, representing the percentages of chance that a patient has cancer, based on the average probability classifications of his microcalcifications.

TESTING THE ALGORITHM

To obtain an algorithm that is to be trusted, various tests are used in order to maximize the test's accuracy. The tests were realised using a patient-by-patient basis, a 10-fold stratified test as well as a 96-fold stratified test (simulating a patient-by-patient test).

Various kernels (linear, 2-poly, 3-poly) and SVM parameters were extensively tested and graphed, in order to find the optimal implementation for the classification.

Particular attention was given to the fact that in disease diagnosing sensitivity is the extent to which actual positives are not overlooked. In this case, what is needed is a highly sensitive test that rarely overlooks an actual positive (for example, showing "nothing bad" despite something bad existing).

Implementation details

OVERVIEW

The chosen implementation for the analysis and classification of microcalcifications is an SVM-assisted algorithm to separate between the two following classes: malignant or benign.

The choice for the algorithm is loosely based on a research paper [1] in which the SVM method is concluded to be the most accurate machine-learning method for breast cancer detection. The paper is only used as a guideline for this project, as they do not use the same type of data for the classification.

SVM is also usually regarded as the first-to-try method for binary classification, as it usually performs really well in that regard.

For the project, Python and scikit-learn are the main tools used for the implementation. The module wraps both liblinear and libsvm, which are regarded as the best tools for SVMs currently available for the language, and it optimizes memory allocations with the wrappings. It is one of the better libraries in terms of performance. It is also open-source and considered well-documented.

The program allows the user to initialize the SVM with either a linear or a polynomial kernel which uses the given error-compensation parameter (C parameter) for the requested predictions. The user needs to input both the training file (training_data.xlsx) and the patient file when initializing the prediction agent.

IMPLEMENTATION

The program is implemented as an agent which parses the given .xlsx files when initialized. The agent can then use various functions to classify microcalcifications and diagnose the patients. The functions are documented, but here is roughly what the program can do:

- Takes an array of data containing microcalcification characteristics and classifies it into benign or malignant.
- Initializes a linear or polynomial support vector machine that can classify directly or with probabilities
- Parse the data from the given .xlsx files in order to be used in training and classification
- Plot the SVM decision boundaries in a 2D plane using PCA feature extraction, but results are disappointing since there is no good way of projecting the data on a 2D plane using only the most important feature
- Plot a K-fold stratified test on the training data, with a varying error-compensation parameter and number of folds.
- Plot the ROC curve on a K-fold stratified test of the training data with a varying error-compensation parameter and number of folds
- Preprocess the data given in the form of .xlsx by normalizing it
- Compute the accuracy of a prediction based on the known classes
- Test the accuracy for the predictions of the testing data set given as a parameter to the agent
- Plot and give a diagnosis for every patient found in the .xlsx file, displaying it at the end

Diagnosing the patient

THE DIAGNOSIS

The program divides the diagnosis in 5 manually adjusted categories based on various tests and the 96-fold ROC results:

- If the patient has an average benign probability of **>= 0.82**, the diagnosis is:

  "Patient healthy, no biopsy needed."

- If the patient has an average benign probability of **>= 0.71**, the diagnosis is:

  "Patient probably healthy, low necessity of biopsy."

- If the patient has an average benign probability of **<= 0.4**, the diagnosis is:

  "Patient probably has cancer, high necessity of biopsy."

- If the patient has an average benign probability of **<= 0.25**, the diagnosis is:

  "Patient has cancer, biopsy needed."

- If the patient has an average benign probability of **>0.4** and **<0.7**, the diagnosis is:

  "Diagnosis inconclusive, biopsy recommended."

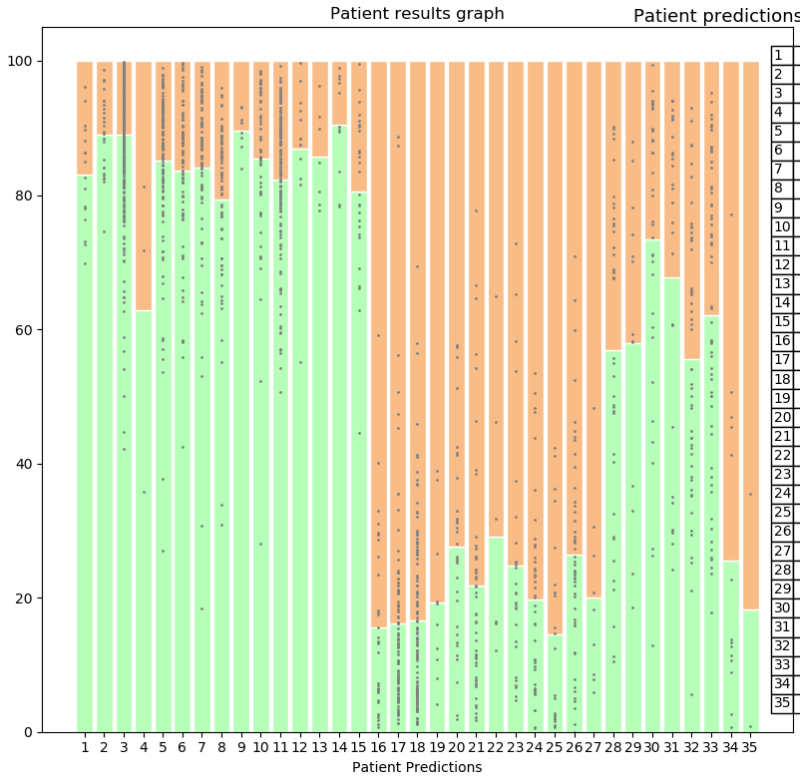
The division does not correspond exactly to the values found during the testing process, because a safety margin was important in this kind of diagnosis. It is better to have a higher chance to misdiagnose a healthy patient as a cancerous one, than to misdiagnose a cancerous patient for a healthy one.

To help with the decision, the program graphs every microcalcification to show their distribution and where they stand in terms of “benign level” and “malignant level”.

INTERPRETING THE RESULTS

The figure below is the default final analysis given by the application. It receives a list of microcalcifications and analyzes the characteristics of each one of them and gives them a probability which places them somewhere on the patient's bar on the chart. Every gray dot represents a patient's microcalcification, the green portion of the bar is the "benign level" of the patient, and the red portion is the "malignant level". The table values are as follows:

[Patient Number | Benign Level | Malignant Level | Diagnosis]



Patient predictions and diagnosis

1	0.83	0.17	Patient healthy, no biopsy needed.
2	0.89	0.11	Patient healthy, no biopsy needed.
3	0.89	0.11	Patient healthy, no biopsy needed.
4	0.629	0.371	Diagnosis inconclusive, biopsy recommended.
5	0.851	0.149	Patient healthy, no biopsy needed.
6	0.836	0.164	Patient healthy, no biopsy needed.
7	0.84	0.16	Patient healthy, no biopsy needed.
8	0.794	0.206	Patient probably healthy, low necessity of biopsy.
9	0.897	0.103	Patient healthy, no biopsy needed.
10	0.854	0.146	Patient healthy, no biopsy needed.
11	0.823	0.177	Patient healthy, no biopsy needed.
12	0.87	0.13	Patient healthy, no biopsy needed.
13	0.857	0.143	Patient healthy, no biopsy needed.
14	0.905	0.095	Patient healthy, no biopsy needed.
15	0.805	0.195	Patient probably healthy, low necessity of biopsy.
16	0.156	0.844	Patient has cancer, biopsy needed.
17	0.162	0.838	Patient has cancer, biopsy needed.
18	0.167	0.833	Patient has cancer, biopsy needed.
19	0.193	0.807	Patient has cancer, biopsy needed.
20	0.277	0.723	Patient probably has cancer, high necessity of biopsy.
21	0.218	0.782	Patient has cancer, biopsy needed.
22	0.291	0.709	Patient probably has cancer, high necessity of biopsy.
23	0.247	0.753	Patient has cancer, biopsy needed.
24	0.198	0.802	Patient has cancer, biopsy needed.
25	0.145	0.855	Patient has cancer, biopsy needed.
26	0.264	0.736	Patient probably has cancer, high necessity of biopsy.
27	0.2	0.8	Patient has cancer, biopsy needed.
28	0.569	0.431	Diagnosis inconclusive, biopsy recommended.
29	0.58	0.42	Diagnosis inconclusive, biopsy recommended.
30	0.734	0.266	Patient probably healthy, low necessity of biopsy.
31	0.677	0.323	Diagnosis inconclusive, biopsy recommended.
32	0.556	0.444	Diagnosis inconclusive, biopsy recommended.
33	0.621	0.379	Diagnosis inconclusive, biopsy recommended.
34	0.256	0.744	Patient probably has cancer, high necessity of biopsy.
35	0.182	0.818	Patient has cancer, biopsy needed.

Fig. 1 – Patient predictions, microcalcifications and diagnosis

Model prediction quality tests

10-FOLD STRATIFIED CROSS VALIDATION TEST

Cross-validation is a statistical method used to estimate the skill of machine learning models. It consists in shuffling the data, splitting it into k groups and for each unique group, taking the group as a test dataset, and taking the remaining groups as a training dataset. The training data is used to fit the model and the testing data is used to test its prediction capabilities. The model is then discarded after evaluation. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. This whole process evaluates the capacity of the model to predict unknown data and is one of the most reliable mechanisms for testing its robustness.

The test is therefore used to evaluate the different kernels and error-compensation parameters, in order to decide which ones are the best for the given problem.

The C parameter tells the SVM optimization to what point to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

The compared kernels were the linear kernel, the 2-poly kernel and the 3-poly kernel, each having been tested with a few hundred different error-compensation parameters (C parameter). The results were in favor of the 2-poly kernel but were still very close. The only notable difference was speed, since training and testing the linear kernel took a lot more time than doing so for the polynomial ones. The following figures represent the tested parameters that appeared optimal.

To read them, read the y-axis as the accuracy of the prediction, meaning the number of microcalcifications predicted as the correct class, and the x-axis as the fold for which the accuracy was tested. The --- orange line is the accuracy mean.

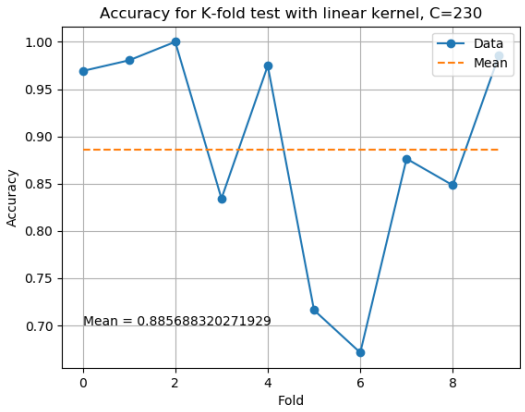


Fig. 1 - Linear 10-fold test

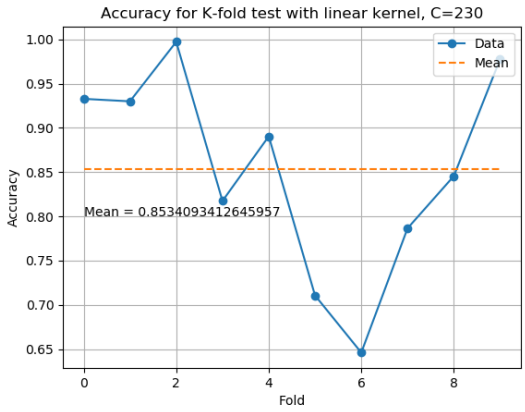


Fig. 2 - Linear 10-fold test with shuffled data

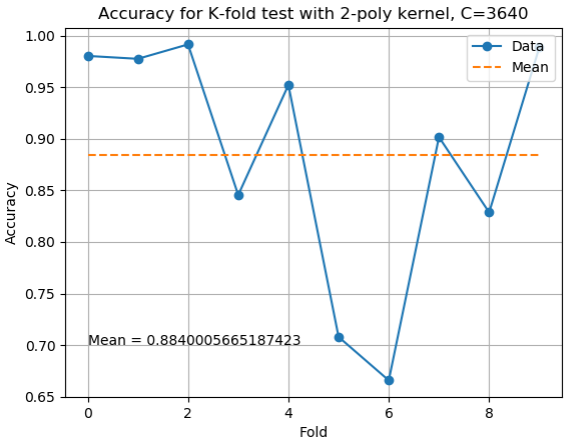


Fig. 3 - 2-poly 10-fold test

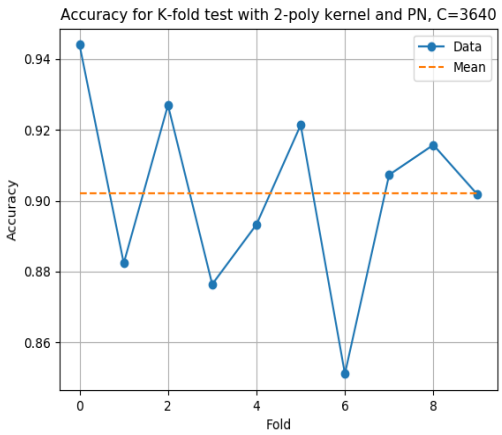


Fig. 4 – 2-poly 10-fold test with shuffled data

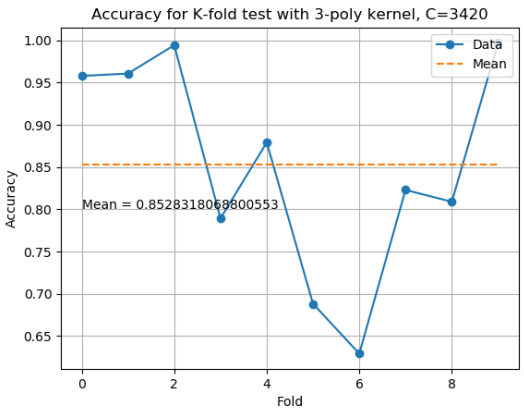


Fig. 5 - 3-poly 10-fold test

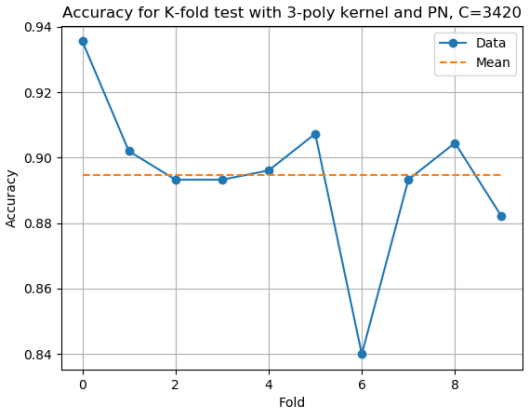


Fig. 6 – 3-poly 10-fold test with shuffled data

As you can see in the above graphs, the 2-poly tests have the highest success ratio, and it is why those are the parameters used for the predictions.

96-FOLD STRATIFIED CROSS VALIDATION TEST

In a more "practical" approach, a "per-patient" k-fold stratified cross validation test was used. Since the dataset contains 96 patients, doing a 96-fold test simulates the prediction that the SVM would do with an average patient containing an average amount of microcalcifications. It is therefore an interesting measure of the accuracy of the algorithm.

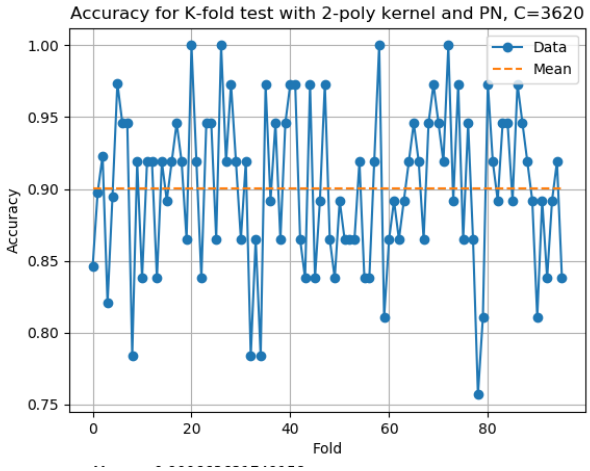


Fig. 7 - 2-poly 96-fold test

The mean is roughly 0.90, which is very high for the prediction model. It also doesn't reflect the same results as the per-patient tests realised.

PER-PATIENT ACCURACY TEST

In order to test the supposed performance of the model in real-life conditions, half the training set was used as training (stratified), and the other half was used to test the predictions of the model. The predictions were split by patient, since this is what would happen if a real doctor tried to use the application.

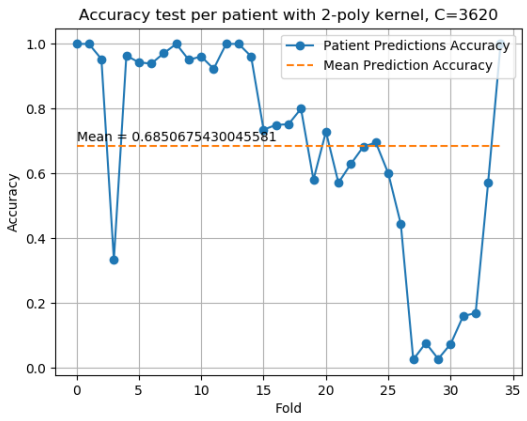


Fig. 8 - 2-poly per-patient test with C=3620

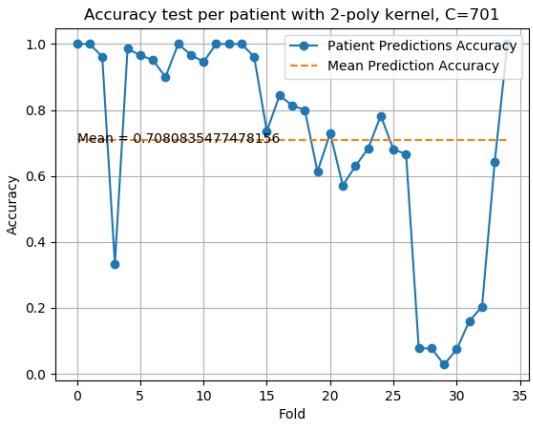


Fig. 9 – 2-poly per-patient test with C=701

The mean is lower than average, which is understandable given the big drop in accuracy from patient 27 to patient 32. The 6 cases are apparently very hard for the algorithm to predict, since it predicts only a few microcalcifications as cancerous, which would indicate nothing to worry about, even though the patient has cancer. The model is also trained using only half the data, which of course lowers its accuracy.

Scaling gamma for per-patient test

While testing out different parameters, the gamma parameter set to "scale" had one of the biggest impacts on per-patient accuracy, while slightly decreasing k-fold test accuracy. For that reason, its use was considered for diagnosing patients, as it seems to have much better results with more difficult cases (as the ones seen above).

However, the probability predictions were pretty bad, so it is not the default mechanism for giving a diagnosis. Perhaps with some tweaking, it can be used for more difficult cases (ones identified as harder to classify) exclusively, but not for this project.

RESULTS INTERPRETATION

Even though the theoretical accuracies are high, when the model faces characteristics which are hard to classify, it has drops in precision. It is therefore wiser to use the probabilities when diagnosing a patient, rather than classifying each microcalcification as benign or malicious.

Considering the fact that difficult cases can be hard to classify, looking at the probability graph for each patient (benign % vs malignant % average over all micros) is more helpful for diagnosing a patient. This is explained in more details in the "Diagnosing the patient" section.

RECEIVER OPERATING CHARACTERISTIC CURVE

One of the more important tests to analyze a binary classification model is the ROC curve, which shows the trade-off between sensitivity (true positives) and specificity (true negatives) - any increase in sensitivity will be accompanied by a decrease in specificity. The area under the ROC curve (AUC) is a measure of the accuracy of the test. The following figure shows the ROC response of different datasets, created from K-fold cross-validation. Taking all these curves, it is possible to calculate the mean AUC and see the variance of the curve when the training set is split into different subsets. This roughly shows how the classifier output is affected by changes in the training data, and how different the splits generated by K-fold cross-validation are from one another.

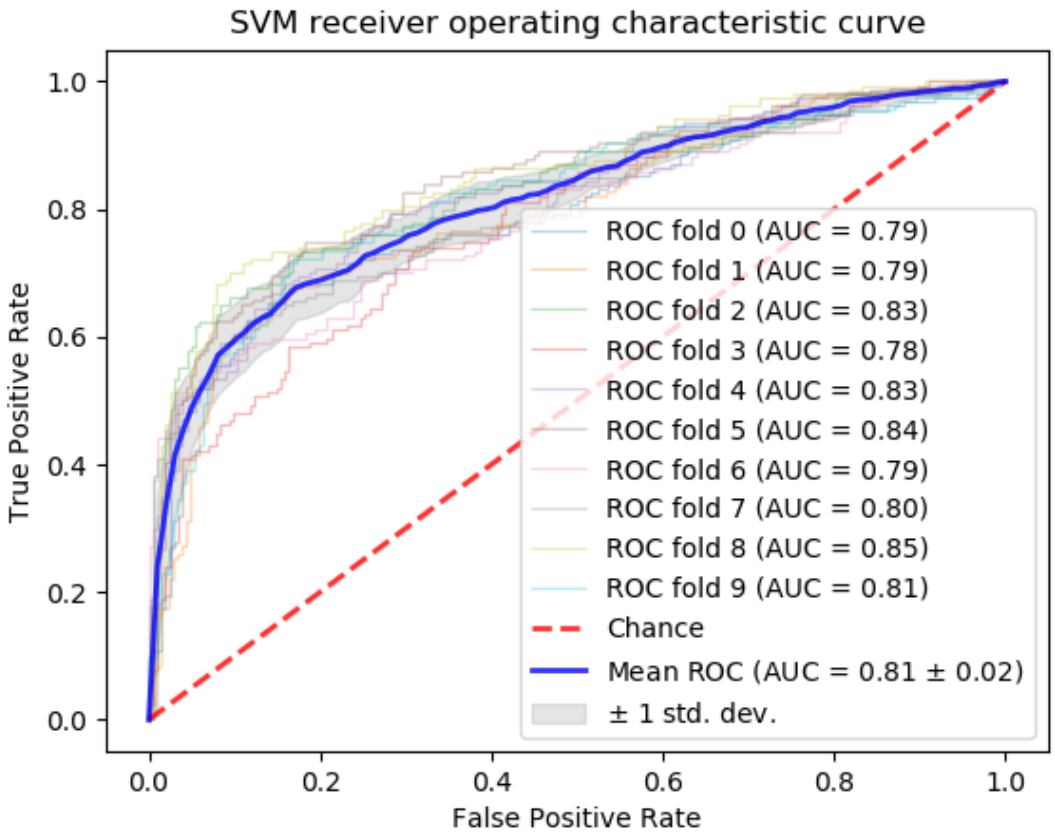


Fig. 10 – Shuffled 10-fold-based ROC curve

We can see in the above graph that the average AUC is 0.81 +/- 0.02, which means that the model should average an accuracy of roughly 80%. The correlation between AUC and accuracy isn't always linear, but the other tests show that here, they are pretty much the same.

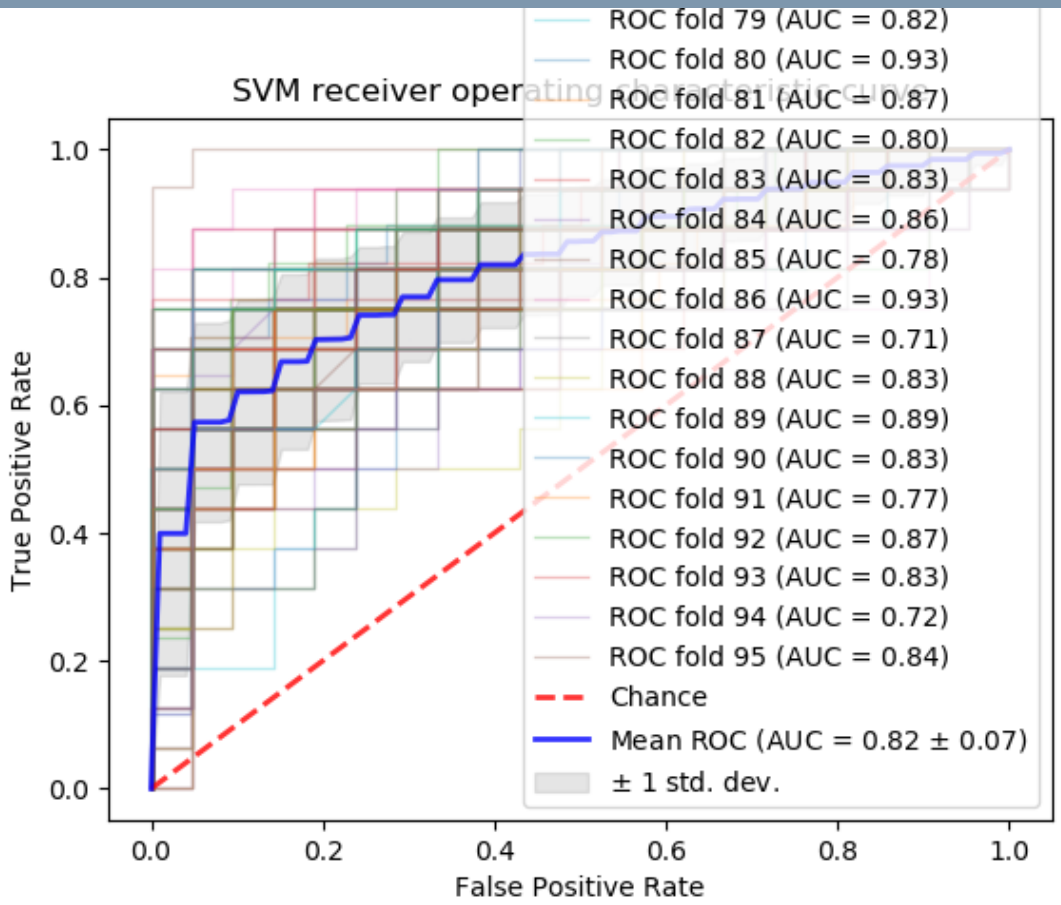


Fig. 11 – Shuffled 96-fold-based ROC curve

In the above figure, the same logic as the 96-fold cross validation test is applied. It simulates the AUC value for the predictions of an average patient.