Cervical Cancer Report

Cervical cancer is the second most common type of cancer that affects women around the world. Especially in developed countries. As with all cancers, early detection offers the best chance for successful treatment, so the ability to use behavioral science, which does not require expensive testing, can have a positive effect on initial diagnosis.

Data overview

This dataset consist of 18 variables that are provided by the seven indicators:

- Intention
- Attitude
- Subjective Norm
- Perception
- Motivation
- Social Support
- Empowerment

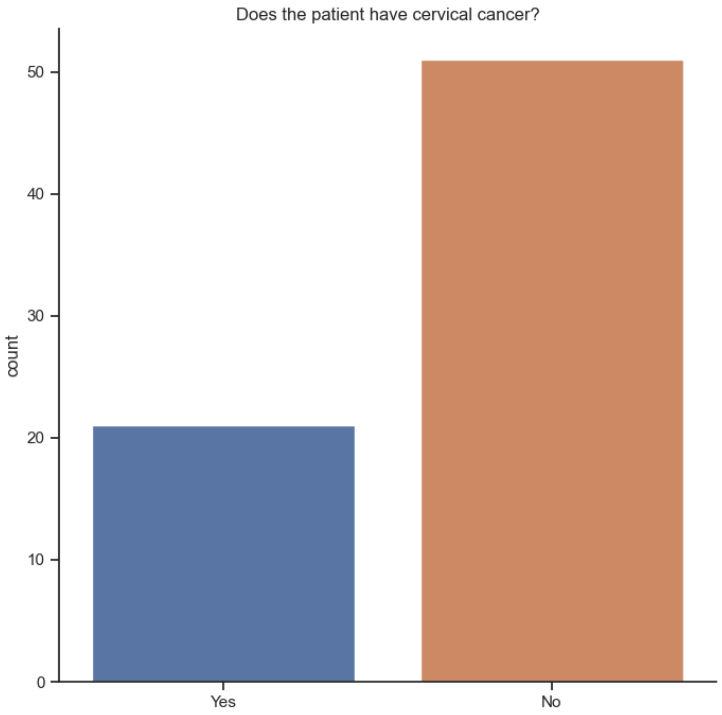
and dependent variable which describes whether respondent has cervical cancer (*1=has cervical cancer, 0=no cervical cancer*):

1. behavior eating
2. behavior personalHygiene
3. intention aggregation
4. intention commitment
5. attitude consistency
6. attitude spontaneity
7. norm significantPerson
8. norm fulfillment
9. perception vulnerability
10. perception severity
11. motivation strength
12. motivation willingness
13. socialSupport emotionality
14. socialSupport appreciation

15. socialSupport instrumental
16. empowerment knowledge
17. empowerment abilities
18. empowerment desires
19. **ca cervix**

Number of respondents: 72

Missing Data: 0



29.17% cases with cancer

Two classification algorithms were used in the study:

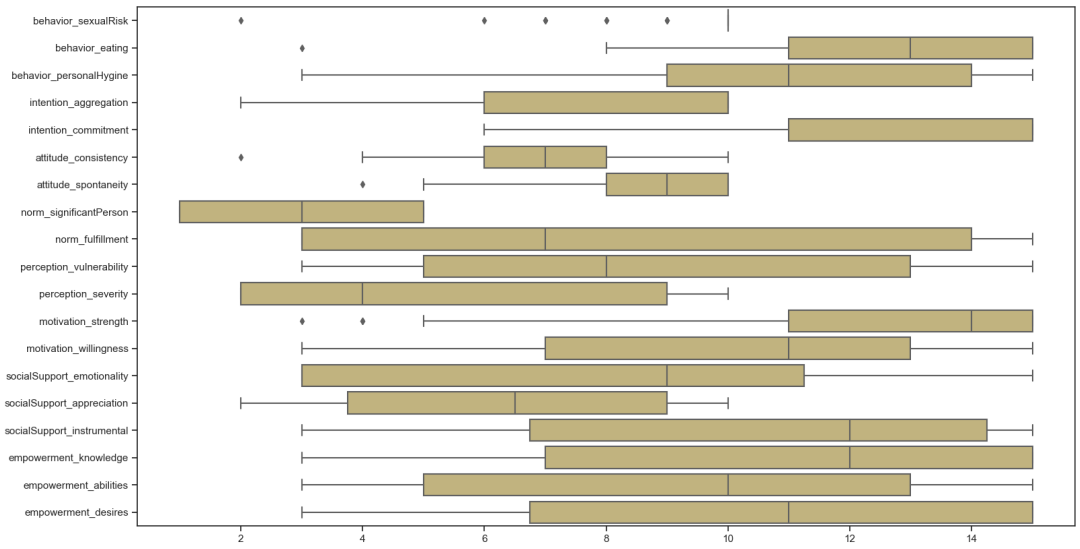
- Naive Bayes (NB)
- Logistic Regression (LR)

A 10-fold cross validation was applied for each of them, and the results obtained were as follows:

- NB:
  - Accuracy: 91.67%
  - AUC: 0.96
- LR:
  - Accuracy: 87.50%
  - AUC: 0.97

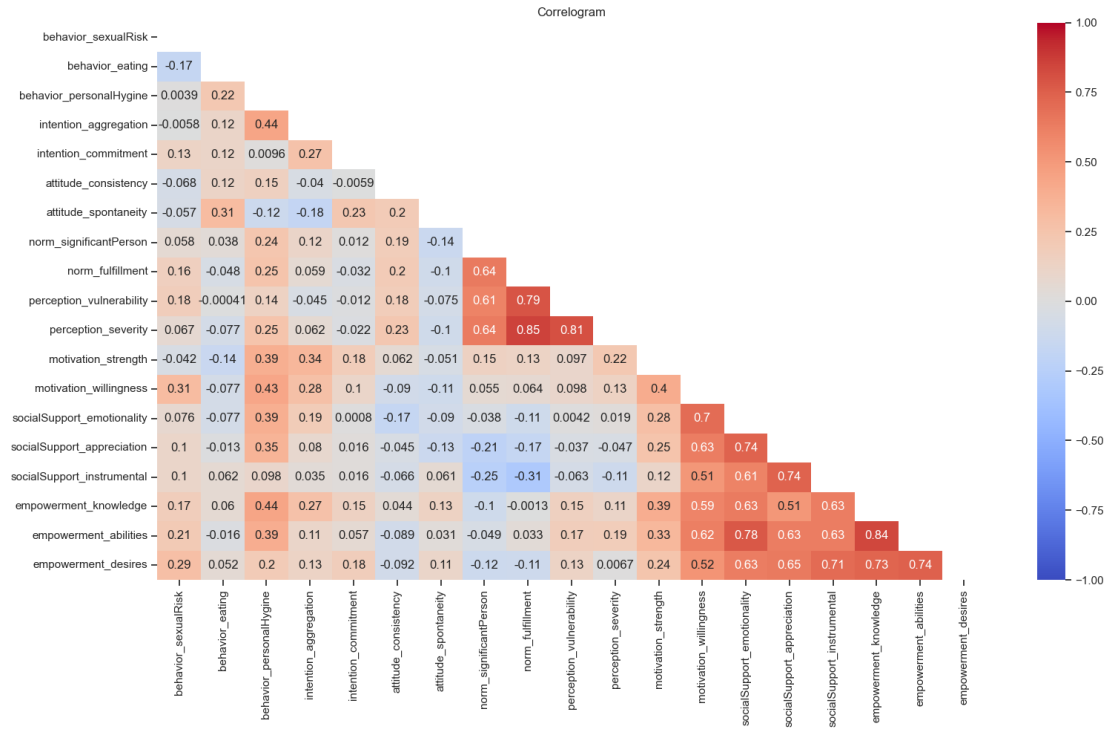
All the data types are numeric (int64).

Coefficient of variation is high in most of the cases expect sexualRisk, eating, commitment, spontaneity where are below 20.
sexualRisk has very high kurtosis (25.87) and negative skewness(-4.73) which indicate presence of outliers among this variable. Two outliers detection has been used ($1.5 \cdot \text{IQR}$ and $3.0 \cdot \text{IQR}$) and both of them showed presence of outliers in this particular variable.



boxplot for all variables

Correlation among variables is relatively strong considering they came from behavioral studies.



At the end **KMO test** was done to decide if Principal Component Analysis can be use for plotting the results. This statistic represents the degree to which each observed variable is predicted, without error, by the other variables in the dataset. In general, a KMO < 0.6 is considered inadequate. In our case we got 0.724, thus we can try PCA.

More numerical information you can fine here.

Basic model from Sklearn

Code below was used for building models from article (*base on the information the article provides*) with the results I got:

- **LR**: 0.95 Accuracy $\pm$ 0.15
- **NB**: 0.94 Accuracy $\pm$ 0.13

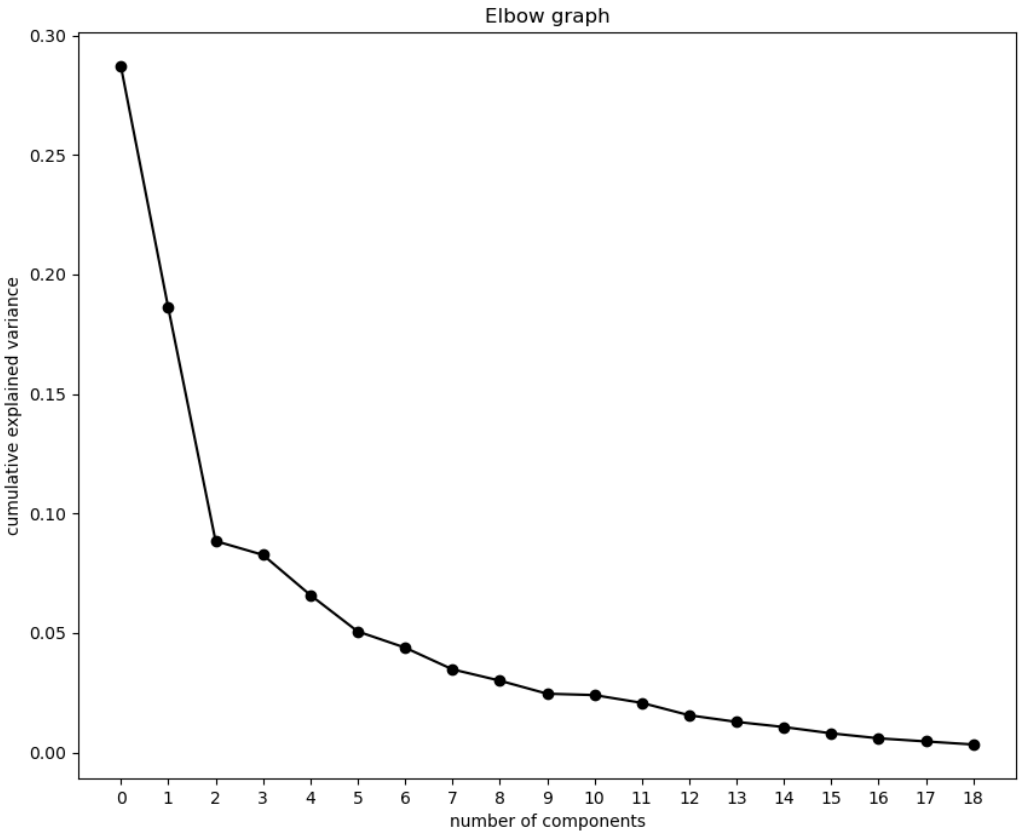
```
models = {"LR": LogisticRegression(),
                  "NB": GaussianNB()}

# Lists for results stores
names = []
results = []
sc = 'accuracy'

# Loop through models
for name, model in models.items():
    pipe = Pipeline([
        ("preProc", StandardScaler()),
            ("model", model)
    ])

    cv_res = cross_val_score(pipe, X, y, cv=10, scoring=sc)
    results.append(cv_res)
    names.append(name)
    print("%s:␣%f␣(%f)" % (name, cv_res.mean(), cv_res.std()))
```

So basically the results Sklearn provided are better then the one obtained by researchers under the same condition.

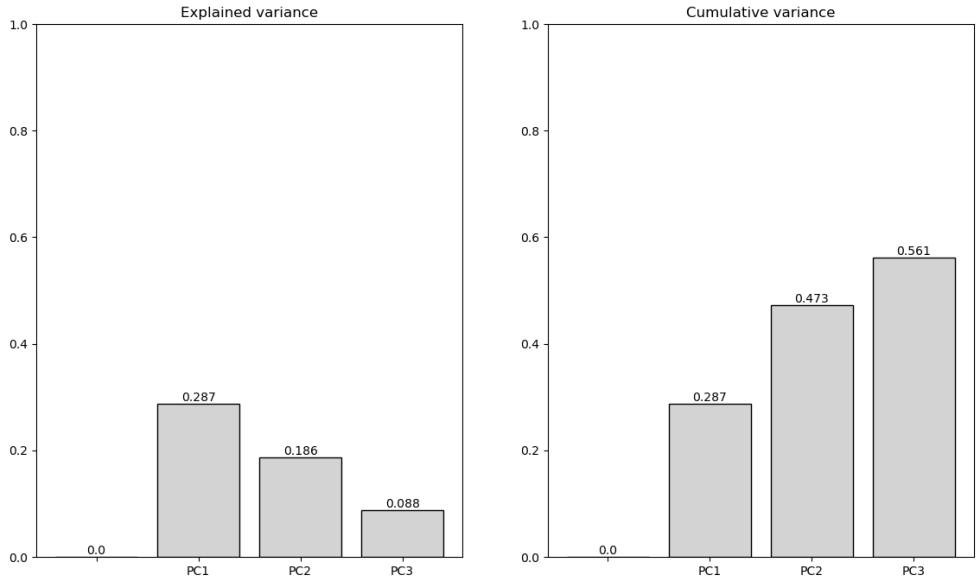
Principal Component Analysis

Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process [Source].

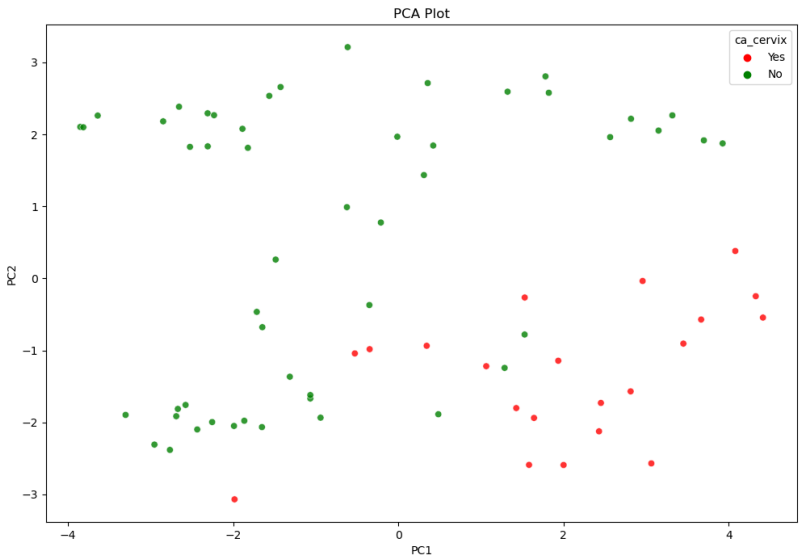
KMO test result let us assumed that PCA make sens, thus we generated elbow graph for choosing the principle components.

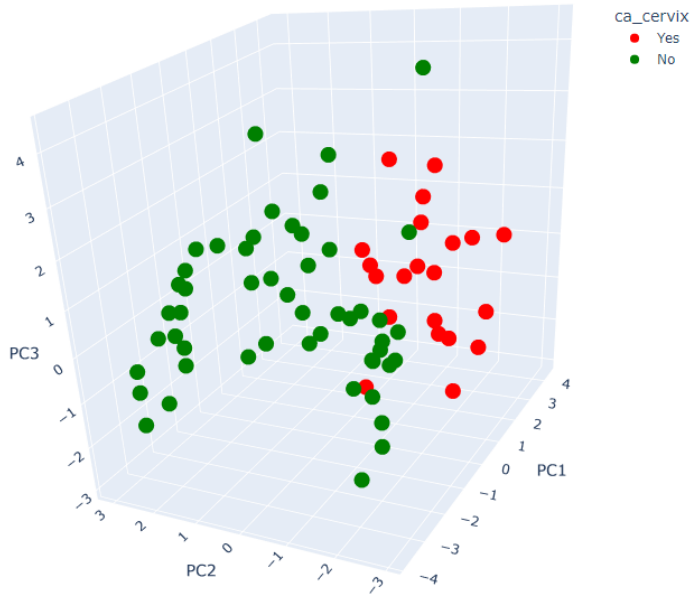


In order to plot our problem, we had to limit the number of components to a maximum of three, which gave us the variance values shown below.



As we can see cumulative variance from three components is relatively low (< 0.6) but the 2D and 3D plots doesn't look bad:





We can see that the cervical cancer cases are close to each other and we can fairly correctly separate these cases using the curve.

Components importance show us which variable contribute the most in given component. The larger the value, the greater the contribution of the variable.

```
Explained variance
[0.28708527 0.18606619 0.08846312]

Components importance
                              PC1       PC2       PC3
behavior_sexualRisk          0.103321  0.065606  0.250468
behavior_eating              0.001861  0.016548  0.512038
behavior_personalHygine      0.218474  0.179417  0.278398
intention_aggregation        0.127109  0.076997  0.335119
intention_commitment         0.059120  0.002435  0.363442
attitude_consistency         0.029147  0.161184  0.289885
attitude_spontaneity         0.013585  0.083331  0.402996
norm_significantPerson       0.018169  0.428449  0.015374
norm_fulfillment             0.004231  0.491977  0.066659
perception_vulnerability     0.057986  0.446174  0.119794
perception_severity          0.057285  0.479074  0.072913
motivation_strength          0.194467  0.128065  0.171868
motivation_willingness       0.345067  0.040238  0.069938
socialSupport_emotionality   0.363981  0.057046  0.108047
socialSupport_appreciation   0.343294  0.111393  0.108998
socialSupport_instrumental   0.320679  0.177210  0.071219
empowerment_knowledge        0.367179  0.005210  0.123677
empowerment_abilities        0.380708  0.011558  0.060318
empowerment_desires          0.352274  0.070239  0.020125
```

For PC1: motivation, social support and empowerment, for PC2: subjective norm and perception and for PC3: intention, behavioral and attitude has the largest share, respectively.

For code look here.

Tuning models in Sklearn

Because Sklearn provide many tuning options for models we decide to improve our previous results and show important metrics, the ones didn't show in study, like recall and precision. We also changed the thresholds base on which model assigns class label.

Splitting the data

We shouldn't *fit* and *predict* on the same data because model know our data and can perform much better then in reality on data it doesn't know. `train_test_split` was used to perform splitting dataset into training and testing with `test_size=0.25` and `stratify=y` with means 25% of our dataset was used for checking model performance (*model didn't learned on this cases*) and - cause we have imbalance dataset - we provided adequate number of cancer cases into test and train datasets.

- Train dataset: 54 rows
- Test dataset: 18 rows

Logistic Regression

In statistics, the logistic model (*or logit model*) is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc. Each object being detected in the image would be assigned a probability between 0 and 1, with a sum of one. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable [Source].

```
pipe_lr = Pipeline([
    ("preProc", StandardScaler()),
        ("model", LogisticRegression())
])

model_1 = GridSearchCV(
    estimator=pipe_lr,
        param_grid={'model__class_weight': [{0: 1, 1: v} for v in
                                            np.linspace(1, 10, 20)],
                                'model__C': np.logspace(-4, 4, 20)},
        cv=10,
        n_jobs=-1
)

model_1.fit(X_train, y_train)
y_pred_m1 = model_1.best_estimator_.predict(X_test)
y_proba_m1 = model_1.best_estimator_.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_proba_m1)
```

For converge purpose we used `StandardScaler`: $z = \frac{X-\mu}{\sigma}$ and for finding the best parameters we used `GridSearchCV` with 10-fold cross validation. The chosen parameters for tuning was: class weight which describe importance of selected class (*weight = 1 is default*) and C which describe inverse of regularization strength.

Best parameters:

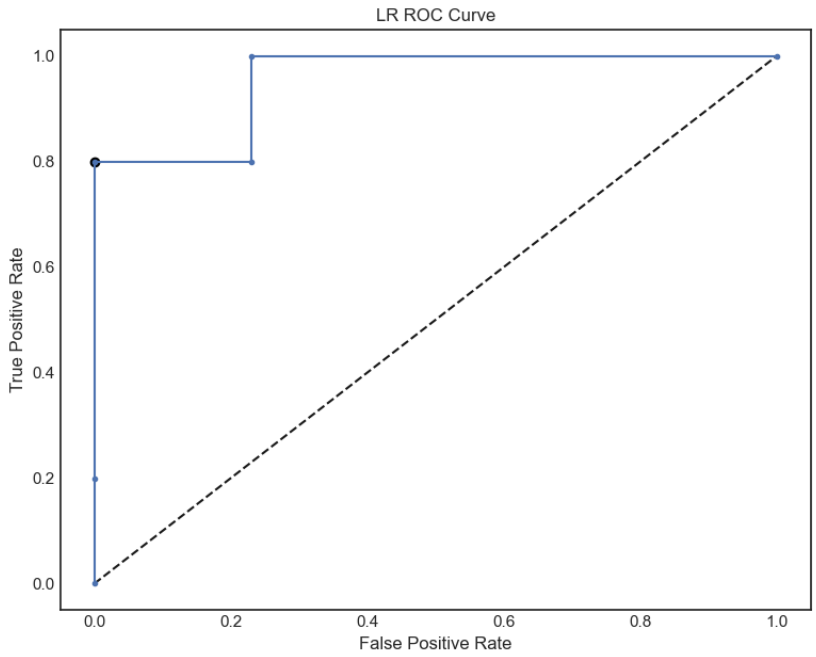
- weight of class 1 (patient has cancer): 1.473684
- C: 0.0885867

Best model accuracy: 0.94667

Below are results obtained on **test dataset**:

```
Test data results

Classification report
              precision    recall  f1-score   support

           0       0.92      0.85      0.88        13
           1       0.67      0.80      0.73         5

    accuracy                           0.83        18
   macro avg       0.79      0.82      0.80        18
weighted avg       0.85      0.83      0.84        18


AUC score
0.823076923076923

Accuracy score
0.8333333333333334
```

ROC curve is presented below.



If we use **G-Mean** for finding the best threshold for our case we can improve obtained results. G-Mean is a metric for imbalanced classification that, if optimized, will seek a balance between the sensitivity and the specificity. Below are results obtained on **test dataset** after changed threshold:

```
Test data results with changed threshold

Best Threshold=0.686739, G-Mean=0.894

Classification report
              precision    recall  f1-score   support

           0       0.93      1.00      0.96        13
           1       1.00      0.80      0.89         5

    accuracy                           0.94        18
   macro avg       0.96      0.90      0.93        18
weighted avg       0.95      0.94      0.94        18


AUC score
0.9

Accuracy score
0.9444444444444444
```

As we can see the results are much better from previous.

Naive Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features [Source].

```
model_2 = GridSearchCV(
    estimator=CategoricalNB(),
        param_grid={'alpha': np.linspace(1e-10, 4, 60)},
        cv=10,
        n_jobs=-1
)

model_2.fit(X_train, y_train)
y_pred_m2 = model_2.predict(X_test)
y_proba_m2 = model_2.predict_proba(X_test)[:, 1]
fpr2, tpr2, thresholds2 = roc_curve(y_test, y_proba_m2)
```

For Naive Bayes we used *Categorical Naive Bayes* from Sklearn with one parameter to tune: alpha which describe smoothing. It was set to $1.0 \cdot 10^{-10}$ cause 0 is to small for this algorithms and Python set it to $1.0 \cdot 10^{-10}$ automatically anyway.

Best parameters:

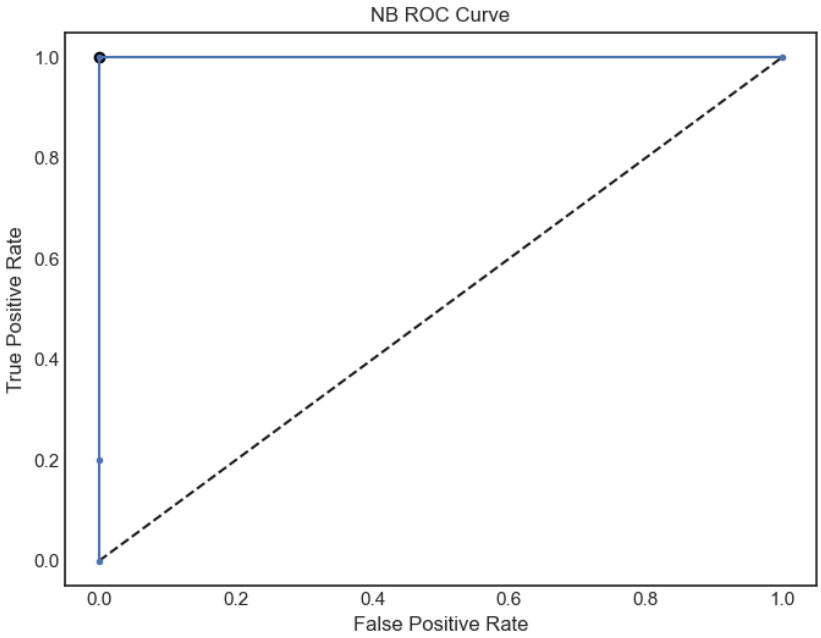
- alpha: 0.0677966

Best model accuracy: 0.87

Below are results obtained on **test dataset**:

```
Test data results

Classification report
              precision    recall  f1-score   support

           0       1.00      0.92      0.96        13
           1       0.83      1.00      0.91         5

    accuracy                           0.94        18
   macro avg       0.92      0.96      0.93        18
weighted avg       0.95      0.94      0.95        18


AUC score
0.9615384615384616

Accuracy score
0.9444444444444444
```

ROC curve is presented below.



Below are results obtained on **test dataset** after changed threshold:

```
Test data results with changed threshold

Best Threshold=0.999951, G-Mean=1.000

Classification report
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00         5

    accuracy                           1.00        18
   macro avg       1.00      1.00      1.00        18
weighted avg       1.00      1.00      1.00        18


AUC score
1.0

Accuracy score
1.0
```

As we can see the results are brilliant - we have reached 100% in all metrics what let us assume that Naive Bayes with changed threshold is the best model for this medical problem.

For code look here.

For results printed from console look here.

Conclusions

Based on the results, we conclude that the best model is the Naive Bayes model learned in 10-fold cross validation with parameter alpha: 0.0677966 and a modified cut-off point (*threshold*).