МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Систем обработки информации и управления»

# ОТЧЕТ

по дисциплине «НИР по обработки и анализу данных »
Тема: «Диагностическая точность методов машинного обучения для нахождения рака молочной железы»

ИСПОЛНИТЕЛЬ: Егоров С.А.
ФИО
группа ИУ5-32М _____
подпись
"__"_____2020 г.

ПРЕПОДАВАТЕЛЬ: Гапанюк Ю.Е.
ФИО
_____
подпись
"__"_____2020 г.

Москва - 2020

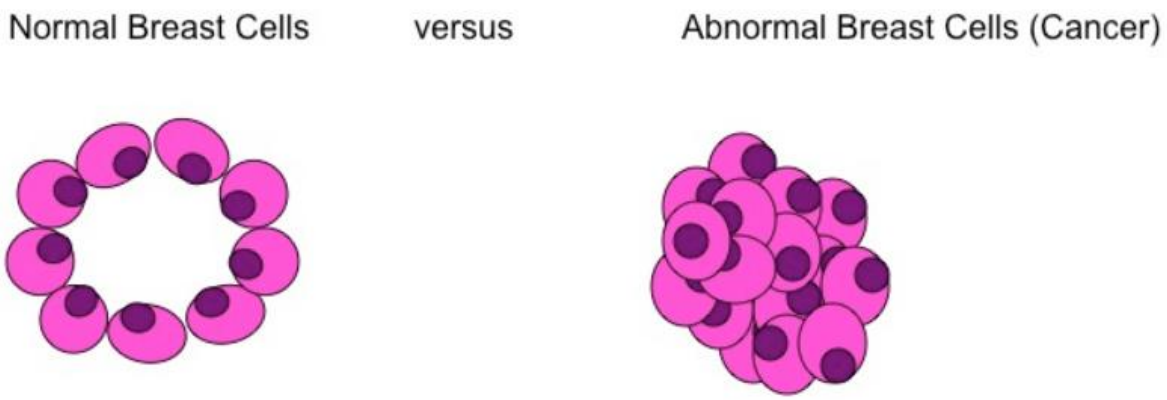
**Оглавление**

## Введение

Во всем мире рак молочной железы является наиболее распространенным видом рака у женщин и вторым по уровню смертности. Диагноз рака молочной железы ставится, когда обнаруживается аномальная опухоль (при самоанализе или рентгенографии) или крошечное пятнышко кальция (на рентгенограмме). После обнаружения подозрительной опухоли врач проведет диагностику, чтобы определить, является ли она злокачественной и, если да, то распространилась ли она на другие части тела.

Рак молочной железы — это заболевание, при котором клетки молочной железы выходят из-под контроля. Существуют различные виды рака молочной железы. Вид рака молочной железы зависит от того, какие клетки в груди превращаются в рак[1].

Можно выделить два вида рака:




1) Нормальные клетки (Normal Breast Cells) – выглядят почти одинаково ( клетки находится в одном и том же месте внутри каждой клетки) и имеют чёткую структуру.
2) Аномальные клетки (Abnormal Breast Cells) – выглядят по разному, клетки дезорганизованы и налезают друг на друга.

В этом исследование попробуем предсказать Рак молочной железы, используя шесть алгоритмов:

1) Логистическая регрессия;

2) SVM (Метод опорных векторов);
3) Наивный байесовский классификатор;
4) Дерево решений;
5) Случайный лес;
6) Метод К-ближайших соседей.

Для сравнения этих методов будем использовать матрицу ошибок.

## Описание набора данных

Этот набор данных по раку молочной железы был получен из больниц Висконсинского университета в Мэдисоне от доктора Уильяма Х. Уолберга [2] и состоит из описания характеристик опухоли и диагноза: злокачественная или доброкачественная это опухоль.

Набор данных включает в себя следующие колоноки:

- mean_radius (среднее значение расстояний от центра до точек по периметру)
- mean_texture (стандартное отклонение значений шкалы серого цвета)
- mean_perimeter (периметр образования)
- mean_area (площадь образования)
- mean_smoothness (локальное изменение длины радиуса)
- diagnosis (1-злокачественные, 0-доброкачественные)

## Предварительная обработка данных

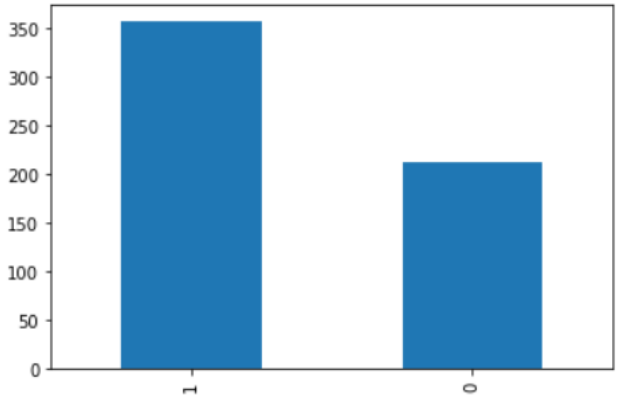
Получим основные характеристики набора данных:

```
Dataset : (569, 6)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   mean_radius      569 non-null    float64
 1   mean_texture     569 non-null    float64
 2   mean_perimeter   569 non-null    float64
 3   mean_area        569 non-null    float64
 4   mean_smoothness  569 non-null    float64
 5   diagnosis        569 non-null    int64
dtypes: float64(5), int64(1)
memory usage: 26.8 KB
```

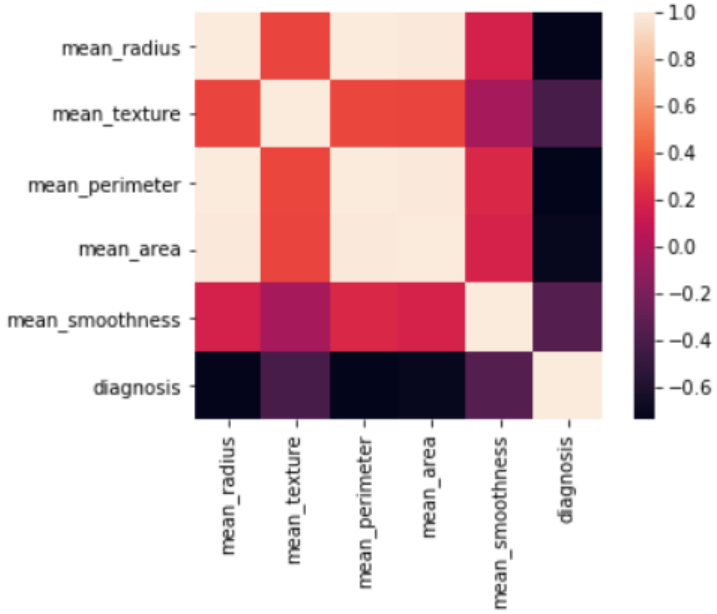
Данные можно сразу использовать, так как они все находятся в нужных форматах и не имеют пропусков. Пример первых 10 строк набора данных:

| | mean_radius | mean_texture | mean_perimeter | mean_area | mean_smoothness | diagnosis |
|---|---|---|---|---|---|---|
| **0** | 17.99 | 10.38 | 122.80 | 1001.0 | 0.11840 | 0 |
| **1** | 20.57 | 17.77 | 132.90 | 1326.0 | 0.08474 | 0 |
| **2** | 19.69 | 21.25 | 130.00 | 1203.0 | 0.10960 | 0 |
| **3** | 11.42 | 20.38 | 77.58 | 386.1 | 0.14250 | 0 |
| **4** | 20.29 | 14.34 | 135.10 | 1297.0 | 0.10030 | 0 |
| **5** | 12.45 | 15.70 | 82.57 | 477.1 | 0.12780 | 0 |
| **6** | 18.25 | 19.98 | 119.60 | 1040.0 | 0.09463 | 0 |
| **7** | 13.71 | 20.83 | 90.20 | 577.9 | 0.11890 | 0 |
| **8** | 13.00 | 21.82 | 87.50 | 519.8 | 0.12730 | 0 |
| **9** | 12.46 | 24.04 | 83.97 | 475.9 | 0.11860 | 0 |

В наборе присутствуют 317 описаний доброкачественных опухолей и 212 злокачественных.

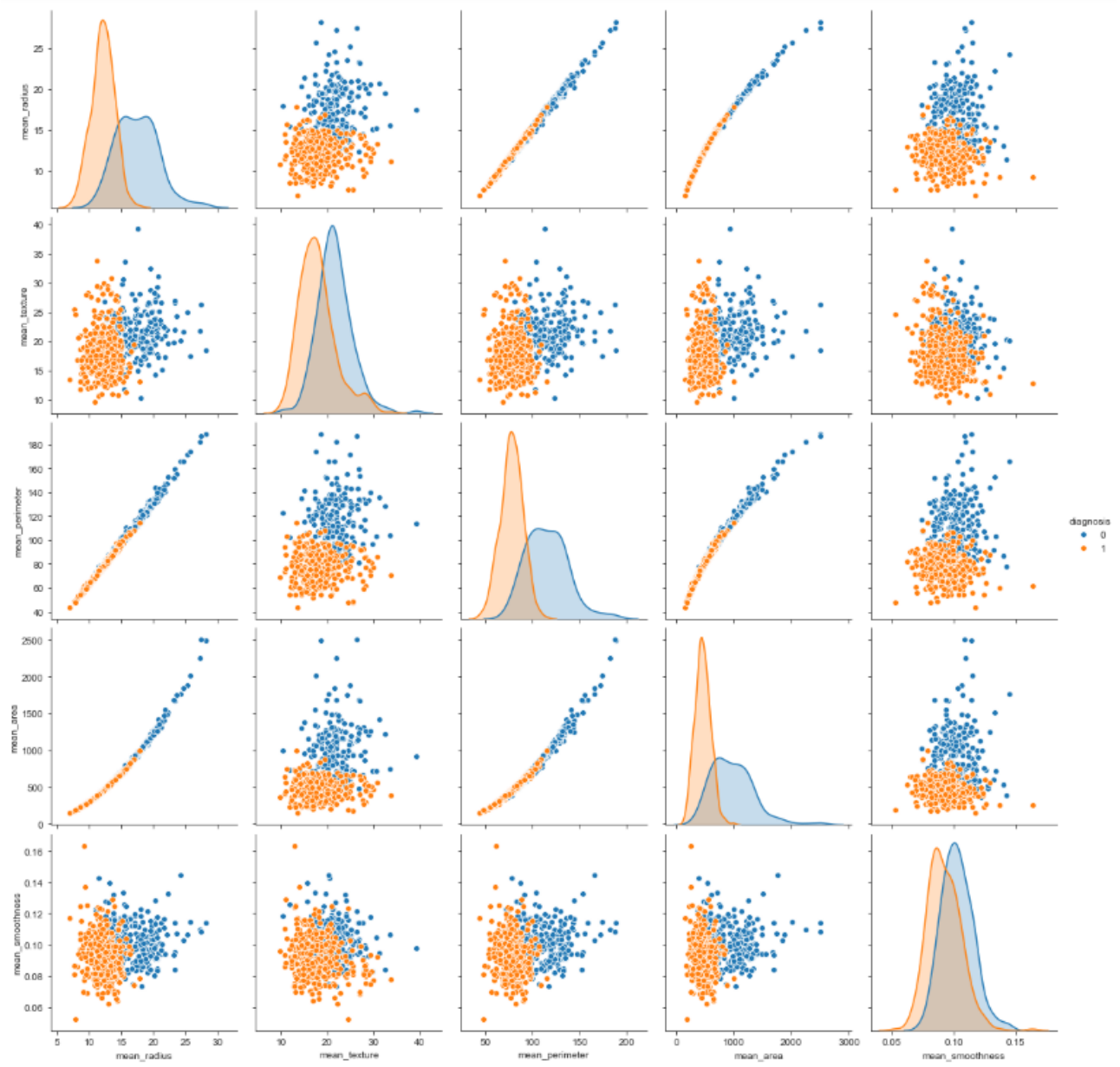


Матрица корреляции:




Как вы можете видеть выше, мы получаем тепловую карту корреляции между переменными. Цветовая палитра сбоку представляет собой величину корреляции между переменными. Более светлый оттенок представляет собой высокую корреляцию.

Корреляция между параметрами разделённые по виду диагноза:




Далее выберем набор обучающих данных, которые будут вводиться в алгоритм машинного обучения, чтобы убедиться, что алгоритм классификации обучения сможет хорошо работать на других данных. Для этого исследования будем использовать выборку размером 20%, предположив в ней идеальное соотношение между обучением и тестированием.

```
X train shape:  (455, 5)
Y train shape:  (455,)
X test shape:  (114, 5)
Y test shape:  (114,)
```

## 1. Логистическая регрессия

Логистическая регрессия — это метод, который может быть применен к задачам бинарной классификации. В этом методе использует логистическую функцию или сигмовидную функцию, которая представляет собой S-образную кривую, которая может принимать любое действительное значение и присваивать ему значение между 0 и 1.

Таким образом, логистическая регрессия моделирует вероятность класса по умолчанию (вероятность того, что вход (X) принадлежит классу по умолчанию (Y=1) ).

$$P(X)=P(Y=1|X)$$

Для того чтобы сделать прогноз вероятности, используется логистическая функция, которая позволяет нам получить логарифмические коэффициенты.

Таким образом, модель представляет собой линейную комбинацию входных данных, но эта линейная комбинация относится к логарифмическим коэффициентам класса по умолчанию.

Создадим экземпляр модели, устанавливающий значения по умолчанию. Укажем обратную величину силы регуляризации равной 10. Обучим модель логистической регрессии на обучающей выборки, а затем применил такую модель к тестовым данным, и найдём среднюю точность.

```
# Определяем модель
logreg = LogisticRegression(C=10)

# Тренируем модель
logreg.fit(X_train, Y_train)

# Прогнозируем целевые значения
Y_predict1 = logreg.predict(X_test)
```

```
#Точность
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, Y_predict1)

print('Средняя оценка точности: {0:0.2f}'.format(
      average_precision))
```

```
Средняя оценка точности: 0.90
```

## 2. SVM (Метод опорных векторов)

SVM (Метод опорных векторов) показал быстрое распространение в течение последних лет. Постановка задачи обучения для SVM соответствует некоторой неизвестной и нелинейной зависимости $y=f(x)$ между некоторым многомерным входным вектором $x$ и скалярным выходом $y$ . Примечательно, что нет никакой информации о совместных вероятностных функциях, поэтому необходимо проводить обучение, используя свободное распределение.

Единственная доступная информация – это набор обучающих данных

$$D=(x_i,y_i)\in X\times Y,\ i=1,l$$

где $l$ обозначает количество пар обучающих данных и, следовательно, равен размеру набора обучающих данных $D$, кроме того, $y_i$ обозначается как $d_i$ , где $d$ обозначает желаемое (целевое) значение. Следовательно, SVM относятся к контролируемым методам обучения.

С точки зрения классификационного подхода, цель SVM состоит в том, чтобы найти гиперплоскость в N-мерном пространстве, которая четко классифицирует точки данных. Таким образом, гиперплоскости являются границами принятия решений, которые помогают классифицировать точки данных.

Обучим модель и найдём её среднюю точность.

```
# Определяем модель
svmcla = OneVsRestClassifier(BaggingClassifier(SVC(C=10,kernel='rbf',random_state=9, probability=True),
                                               n_jobs=-1))

# Тренируем модель
svmcla.fit(X_train, Y_train)

# Прогнозируем целевые значения
Y_predict2 = svmcla.predict(X_test)
```

```
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, Y_predict2)

print('Средняя оценка точности: {0:0.2f}'.format(
      average_precision))
```

```
Средняя оценка точности: 0.87
```

## 3. Наивный байесовский классификатор

Наивный байесовский классификатор — это вероятностный классификатор, основанный на теореме Байеса с сильными допущениями независимости между признаками.

Таким образом, с помощью теоремы Байеса^

$$\mathrm{P(X|У)} = \frac{(\mathrm{P}(Y|X)P(X))}{P(Y)}$$

Мы можем найти вероятность X, учитывая, что изменение произошло. Здесь Y — это доказательство, а X-гипотеза. Предположение, сделанное здесь, состоит в том, что наличие одного конкретного признака не влияет на другой. Поэтому его называют наивным. В этом случае мы предположим, что значения взяты из гауссовского распределения, и поэтому мы рассматриваем Гауссовское наивное байесовское распределение.

```
# Определяем модель
nbcla = GaussianNB()

# Тренируем модель
nbcla.fit(X_train, Y_train)

# Прогнозируем целевые значения
Y_predict3 = nbcla.predict(X_test)
```

```
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, Y_predict3)

print('Средняя оценка точности: {0:0.2f}'.format(
      average_precision))
```

```
Средняя оценка точности: 0.88
```

## 4. Дерево решений

Дерево решений — это древовидная структура, подобная блок-схеме, в которой внутренний узел представляет объект, ветвь представляет правило принятия решения, а каждый конечный узел представляет результат.

Дерево решений анализирует набор данных для построения набора правил или вопросов, которые используются для прогнозирования класса, то есть цель дерева решений состоит в создании модели, которая предсказывает значение целевой переменной путем изучения простых правил принятия решений, выведенных из особенностей данных. В этом смысле дерево решений выбирает наилучший атрибут, используемый для разделения записей, Преобразуя этот атрибут в узел принятия решений и разделяя набор данных на меньшие подмножества, чтобы, наконец, начать построение дерева, повторяя этот процесс рекурсивно.

Обучим модель и найдём её среднюю точность.

```
# Определяем модель
dtcla = DecisionTreeClassifier(random_state=9)

# Тренируем модель
dtcla.fit(X_train, Y_train)

# Прогнозируем целевые значения
Y_predict4 = dtcla.predict(X_test)
```

```
:
#Точность
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, Y_predict4)

print('Средняя оценка точности: {0:0.2f}'.format(
      average_precision))
```

```
Средняя оценка точности: 0.86
```

## 5. Случайный лес

Основываясь на предыдущем методе классификации, случайный лес — это контролируемый алгоритм обучения, который создает лес случайным образом. Этот лес представляет собой набор деревьев решений, в большинстве случаев, обученных с помощью бэггинга. Основная идея бэггинга заключается в том, что классификаторы не исправляют ошибки друг друга, а компенсируют их при голосовании. Каждое дерево строится по следующему алгоритму:

1) Пусть N-число тестовых случаев, M-число переменных в классификаторе.
2) Пусть m-число входных переменных, используемых для определения решения в данном узле; m<M .
3) Выберите обучающий набор для этого дерева и используйте остальные тестовые случаи для оценки ошибки.
4) Для каждого узла дерева случайным образом выберите m переменных, на основе которых будет приниматься решение. Вычислите наилучшее разбиение обучающего набора из m переменных.

Для предсказания новый случай толкается вниз по дереву. Затем ему присваивается метка терминального узла, где он заканчивается. Этот процесс повторяется всеми деревьями в сборке, и метка, которая получает наибольшее количество инцидентов, сообщается как прогноз.

Возьмём количество деревьев в лесу равное 100, обучим модель и найдём точность данной модели:

```
# Определяем модель
rfcla = RandomForestClassifier(n_estimators=100,random_state=9,n_jobs=-1)

# Тренируем модель
rfcla.fit(X_train, Y_train)

# Прогнозируем целевые значения
Y_predict5 = rfcla.predict(X_test)
```

```python
#Точность
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, Y_predict5)

print('Средняя оценка точности: {0:0.2f}'.format(
      average_precision))
```

```
Средняя оценка точности: 0.91
```

## 6. Метод К-ближайших соседей

К-ближайшие соседи — это метод, который хранит все доступные случаи и классифицирует новые случаи на основе меры сходства (например, функции расстояния). Этот метод непараметрический, так как нет никаких предположений для распределения базовых данных, и он ленив, так как не нуждается в какой-либо обучающей точечной модели данных. Все обучающие данные используются на этапе тестирования. Это делает обучение быстрее, а фазу тестирования медленнее и более дорогостоящей.

В этом методе число соседей k обычно нечетное число, если число классов равно 2. Для нахождения ближайших похожих точек найдите расстояние между точками, используя такие меры расстояния, как Евклидово расстояние, расстояние Хэмминга, манхэттенское расстояние и расстояние Минковского.

Обучим модель и найдём её среднюю точность.

```
# Определяем модель
kncla = KNeighborsClassifier(n_neighbors=5,n_jobs=-1)

# Тренируем модель
kncla.fit(X_train, Y_train)

# Прогнозируем целевые значения
Y_predict6 = kncla.predict(X_test)
```

```
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, Y_predict6)

print('Средняя оценка точности: {0:0.2f}'.format(
      average_precision))
```

```
Средняя оценка точности: 0.87
```

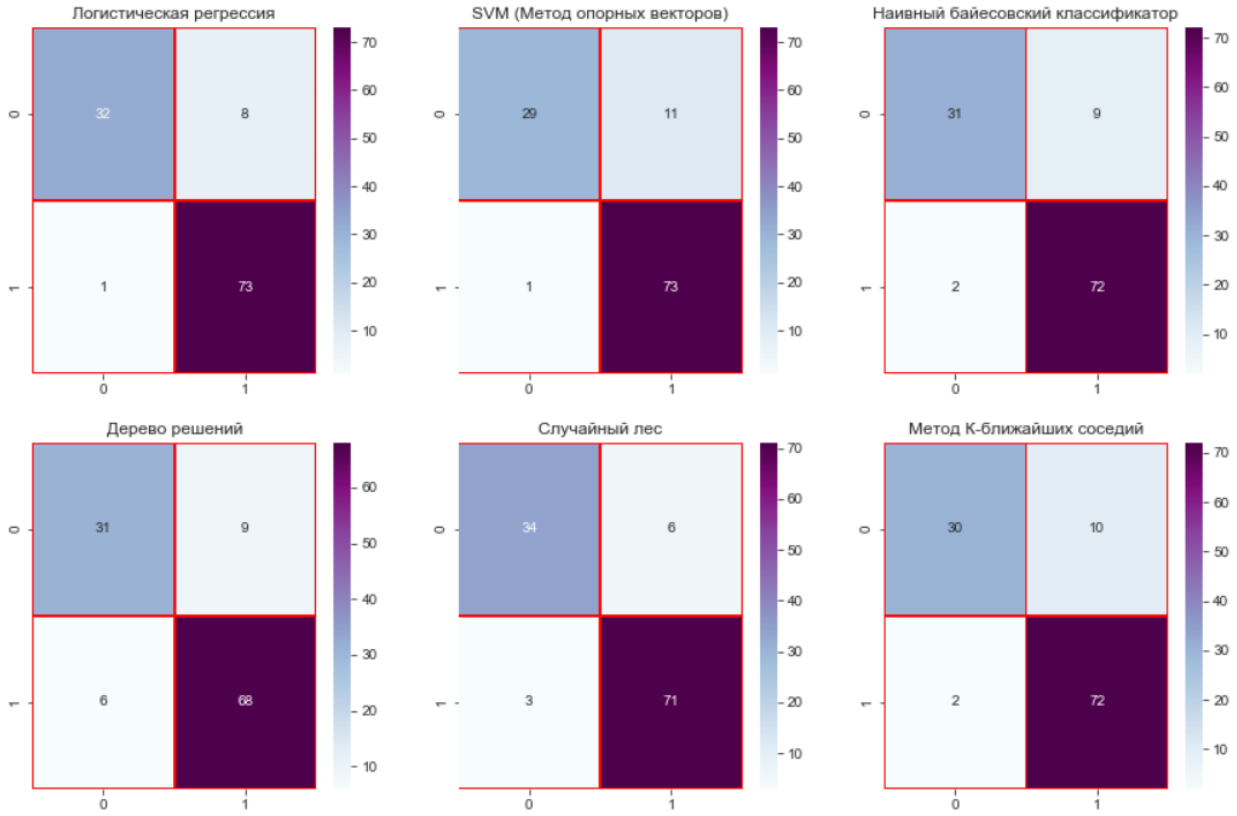
## Сравнение методов классификации

Как писалось в начале работы сравнивать данные методы машинного обучения для диагностирования рака молочной железы будем при помощи матриц ошибок.

Оценка точности всех методов:

| | |
|---|---|
| Логистическая регрессия | 0.921053 |
| SVM (Метод опорных векторов) | 0.894737 |
| Наивный байесовский классификатор | 0.903509 |
| Дерево решений | 0.868421 |
| Случайный лес | 0.921053 |
| Метод К-ближайших соседей | 0.894737 |

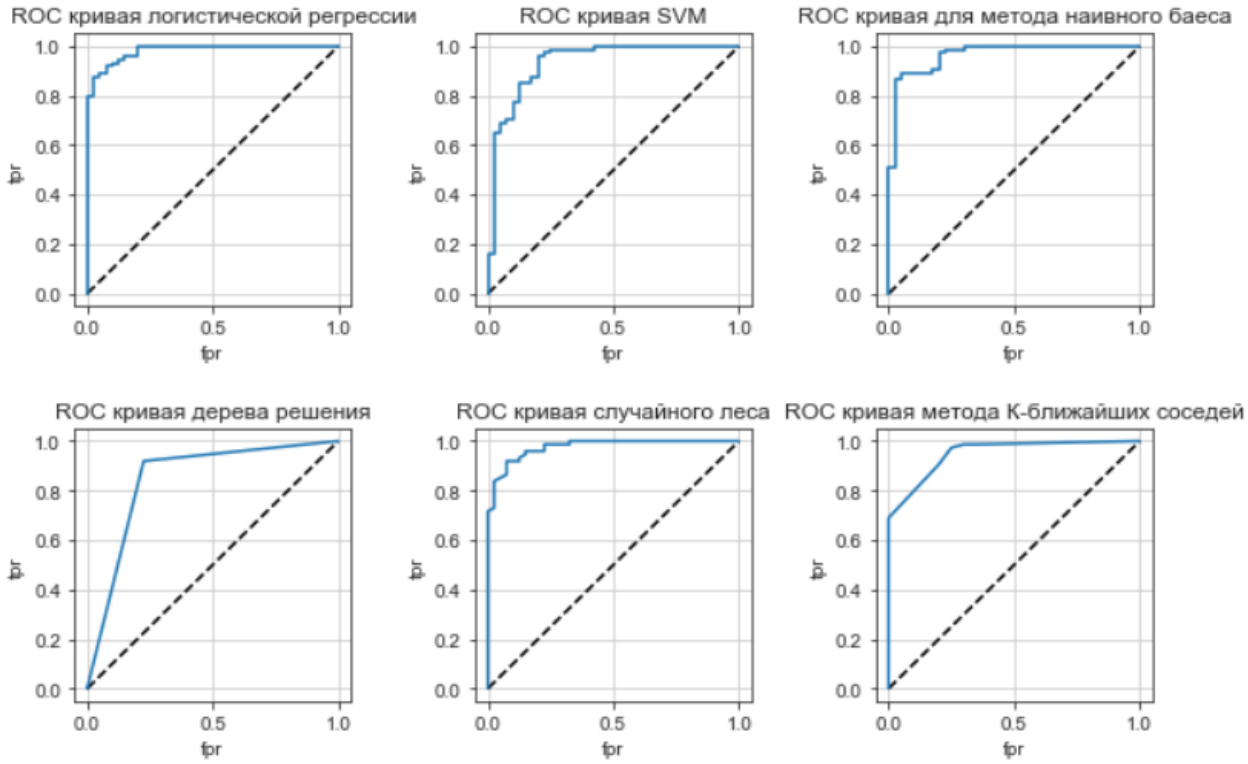
Матрицы ошибок для каждого метода:




Как видно из приведенного выше сравнения методов классификации, мы можем сказать, что точнее в диагностики были методы: «Логистическая регрессия» и «Случайный лес».

## Заключение

Воспользуемся кривой ошибок для оценки бинарной классификации.




Как видно из графиков лучшими методами для бинарной классификации являются «Логистическая регрессия» и «Случайный лес», так как у них AUC (площадь под графиком) больше других

## Список источников литературы


1) What Is Breast Cancer? [Электронный ресурс] Режим доступа: https://www.cdc.gov/cancer/breast/basic_info/what-is-breast-cancer.htm(26.11.2020)
2) Breast Cancer Prediction Dataset [Электронный ресурс] Режим доступа: https://www.kaggle.com/merishnasuwal/breast-cancer-prediction-dataset(26.11.2020)
3) Курс лекций “Методы машинного обучения”[Электронный ресурс] Режим доступа: https://github.com/ugapanyuk/ml_course_2020/wiki/COURSE_MMO(26.11.2020)