Supervised machine learning in example of breast cancer

Autors: Suljić Anisa & Merzić Naida
Index numbers: IB180102 & IB180008
Faculty: Faculty of information technology
University: University "Džemal Bijedić" of Mostar
City and country: Mostar, Bosnia and Herzegovina
Date: January, 2022.
Emails: anisa.suljic@edu.fit.ba & naida.merzic@edu.fit.ba

***Abstract*—This document contains basic information about supervised machine learning, types of supervised machine learning and some algorithms of these types. We will get acquainted with the support vector machine, K-nearest neighbor and logistic regression algorithms in example breast cancer. We will see characteristics of these algorithms, how they work in concrete example and compare their results to find the best algorithm to solve problem of malignant tumor prediction.**

Keywords—algorithm, classification, regression, model, prediction

I. WHAT IS SUPERVISED LEARNING?

Supervised learning is a well-known subcategory of machine learning and artificial intelligence and it is often called supervised machine learning. It is the science of giving computers the ability to learn to make decisions from data without being explicitly programmed. Algorithms that are part of supervised machine learning are designed to learn by example. How its function relates to its name which originates from the idea that training this type of algorithm is like having a teacher supervise the entire process. Datasets which are used are there to train algorithms to classify data or predict outcomes accurately. Input data which is fed into the model adjusts up to the level while the model has been fitted appropriately, across validation process. When training this type of algorithm, the data that is represented as training data will consist of inputs paired with the correct outputs. The algorithm is, during the training process, searching for patterns in the data that correlate with the desired outputs. What is happening after the training process is that a supervised learning algorithm is taking in new unseen inputs and determining which label the new inputs will be classified as based on prior training data. Supervised machine learning is used to help the organizations solve real-world problems at scale, such as classifying spam in a separate folder from your inbox. To yield the desired output, supervised learning uses a training set, which includes inputs and outputs, to teach models. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.

Supervised learning has two types:

A. Classification

B. Regression.

A. Classification

Classification is a process of categorizing a given set of data into classes. The classes are also named target, label or categories. It can be performed on both structured and unstructured data. The process starts with predicting the class of given data points. In general, with classification, supervised machine learning creates models that distinguish between the classes that are present in the dataset they are induced from. Solution for classification problems is in a numerous amount of algorithms. To choose the right solution is to choose the correct algorithm, but that depends on the data and the situation you have. In our case we decided to use two classification algorithms, **Support Vector Machine** and **K-Nearest Neighbor**.

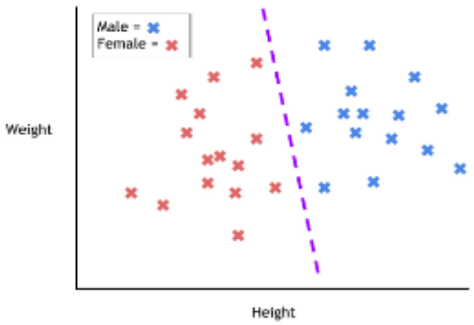

Figure 1. An example of a classification into men and women based on height and weight

B. Regression

Regression is represented in a process where the model attempts to find the important relationship between dependent and independent variables. What regression algorithm predicts is a continuous number such as sales, income, and test scores.

The equation for basic linear regression:

$$\hat{y} = w[0] * x[0] + w[1] * x[1] + \cdots + w[i] * x[i] + b$$

Where x[i] is the feature(s) for the data and where w[i] and b are parameters which are developed during training.

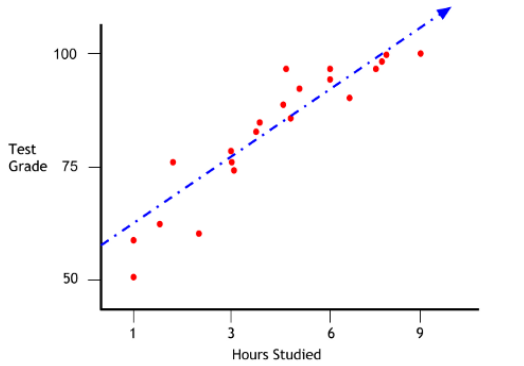

Figure 2. An example of a logistic regression based on studying hours which shows probability of passing an exam

There are many diverse types of regression algorithms, but the one we will use is **Logistic Regression.**

II. EXPLANATION OF ALGORITMS

A. Support vector machine algorithm

This algorithm which belongs to supervised machine learning algorithms can be used for classification or regression problems. A technique called the kernel trick is way how it is functioning. It transforms our data and then finds an optimal boundary between the possible outputs, based on those transformations. In general, after complex data transformation and after our definition of labels or outputs, it figures out way to separate our data.

B. K-nearest neighbor algorithm

K-nearest neighbor algorithm is the second one who can handle both classification and regression problems as well. This type of algorithm works with similarities which are close. In general, it assumes how similar things are near to each other. K is the value of how many similar neighbors we analyze. Some of the first steps of KNN is loading the data and initializing K to our chosen number of neighbors, then comes calculating distance between the query example and the current example from the data, adding the distance and the index of the example to an ordered collection etc. The way how KNN function is quite simple to implement and the algorithm is versatile.

C. Logistic regression algorithm

Logistic Regression is a classification algorithm in machine learning that uses one or more independent variables to determine an outcome. For this type of algorithm stands the rule that it will have only two outcomes. Also, it is the best one when it comes to binary classification, it can be classified as its disadvantage because it assumes that the data is free of missing values and assumes that the predictors are independent of each other. But it can be classified as its advantage because it quantitatively explains the factors leading to classification.

Logistic Regression is used for categorical variables like 0/1, True/False, Yes/No, etc.

Use cases:

- *Identifying risk factors for diseases*
- *Word classification*
- *Weather Prediction*
- *Voting Applications*

III. CASE STUDY : BREAST CANCER

As we can see this case study is about breast cancer. We all know how serious this disease is and how much the prediction about it can be useful and important for medicine, doctors and sick people. There are two types of cancer, malignant and benign. So, what we wanted to achieve with this project is creating a model that will predict if the tumor is malignant or benign, so that doctors know how to treat the tumor.

Dataset we are using is derived from Fine needle aspiration. FNA is a type of biopsy procedure, a thin needle is inserted into an area of abnormal appearing tissue or body fluid. As with other types of biopsies, the sample collected during fine needle aspiration can help make a diagnosis or rule out conditions such as cancer. In the dataset we have a column named diagnosis which is represented with 0 if the tumor is malignant and with 1 if it is benign.

At the very beginning what we are doing is the data collection process. Once we have the dataset we want to use, the data collection process is completed. Then comes the data process, which means dragging the raw data through some preprocessing steps till it comes to machine learning model. Once we have done that, we need to split data into training and testing data so the purpose of this is we will train our machine learning model using the training data and then we will evaluate model using the test data. To make better and more accurate prediction models, we will use three algorithms and compare their results using accuracy score, recall score, precision score and F1 score. We will use a logistic regression algorithm, support vector machine algorithm and K-nearest neighbor algorithm because these algorithms are the best when it comes to binary classification so considering that our data is based on the fact if the tumor is malignant or benign, that is the 0 and 1 values.

A. *Data Understanding*

When we have breast cancer dataset, we must analyze them to know how much data we are working with and what useful information that data can give us. In the figure below are shown data which we will use in continuation.

```
[12] print(data_frame)

     mean radius  mean texture  ...  worst symmetry  worst fractal dimension
0          17.99         10.38  ...          0.4601                  0.11890
1          20.57         17.77  ...          0.2750                  0.08902
2          19.69         21.25  ...          0.3613                  0.08758
3          11.42         20.38  ...          0.6638                  0.17300
4          20.29         14.34  ...          0.2364                  0.07678
..           ...           ...  ...             ...                      ...
564        21.56         22.39  ...          0.2060                  0.07115
565        20.13         28.25  ...          0.2572                  0.06637
566        16.60         28.08  ...          0.2218                  0.07820
567        20.60         29.33  ...          0.4087                  0.12400
568         7.76         24.54  ...          0.2871                  0.07039

[569 rows x 30 columns]
```

Figure 3. Summary view of the data set

There are 569 rows and 30 columns, which means that we have 569 instances, medical records of people who have or have had cancer, that represent different values of 30 health parameters important for prediction if cancer is malignant or benign. Columns in predictive analytics are called features, and except 30 descriptive features there is one target feature called **'label'** and represents diagnosis. There are ten real-valued features that are computed for each cell nucleus and those are the first ten. Another ten features are standard error and the last ten features are "worst" or largest (mean of the three largest values). In figure 4 we can see a list of all features and descriptions of individual features.

No.	Feature name	Description
1	Mean radius	Mean of distances from the center to points on the perimeter cell
2	Mean texture	The standard deviation of grayscale values
3	Mean perimeter	Perimeter of cell
4	Mean area	Area of cell
5	Mean smoothness	Local variation in radius lengths
6	Mean compactness	Perimeter ^ 2/area—1.0
7	Mean concavity	The severity of concave portions of the contour
8	Mean concave points	Number of concave portions of the contour
9	Mean symmetry	Symmetry
10	Mean fractal dimension	Coastline approximation"—1
11	Radius error	—
12	Texture error	—
13	Perimeter error	—
14	Area error	—
15	Smoothness error	—
16	Compactness error	—
17	Concavity error	—
18	Concave points error	—
19	Symmetry error	—
20	Fractal dimension error	—
21	Worst radius	—
22	Worst texture	—
23	Worst perimeter	—
24	Worst area	—
25	Worst smoothness	—
26	Worst compactness	—
27	Worst concavity	—
28	Worst concave points	—
29	Worst symmetry	—
30	Worst fractal dimension	—
31	label	Diagnosis -> M = malignant = 1B = benign = 0

Figure 4. Features and their explanation

Valuable information is how many of the 569 are malignant and how many are benign. After we counted all instances by target feature, we got the information that there is a more benign diagnosis than malignant. In this dataset there are 357 benign and 212 malignant cases or 37,5% malignant and 62,5% benign cases, so in the next figure we will see the graphic representation of malignant and benign cases.

```
[ ] #checking the distribution of Target Variable
    data_frame['label'].value_counts()

    1    357
    0    212
    Name: label, dtype: int64
```

- 1 ---> Benign
- 0 ---> Malignant

```
[ ] #number of each Outcome (diabetic-1 and non-diabetic-0)
    sns.countplot(x = 'label',data = data_frame)

    <matplotlib.axes._subplots.AxesSubplot at 0x7f1e7b0bba10>
```

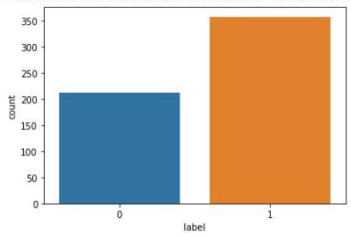

Figure 5. Graphical representation of the number of instances with the diagnosis of malignant and benign tumors

It is good practice to visualize data because it helps to understand data and to explain the data to another person. Here is used Panda's visualization based on "matplotlib" library to find data distribution of first ten features.

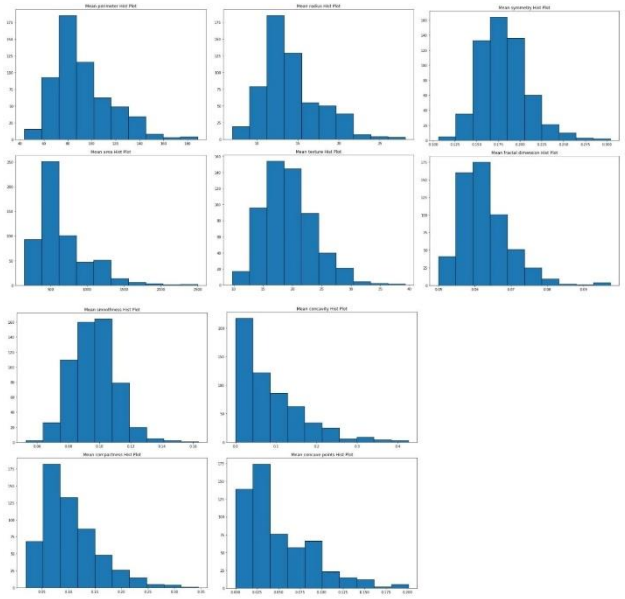

Figure 6. Histograms for first ten features

B. *Data Preparation*

Before using these data, we must check if there are some data that have null values or have data type which is not compatible with other data types. The reason this is important is to check the validity of the data we intend to work with. Data that has a null value is unusable and should be avoided. We can remove such data, but it also means the loss of data and thus information. Also depending on the context of the data we can replace them with the value 0 or take the arithmetic mean of the previous and next values in that column. In situations when we have a data type that differs from all other data types or we do not expect this column to be that type of data because it is unusable, it is necessary to convert this data into the type with which it is

most appropriate to work, which of course depends on the data context. Breast cancer dataset is well made, so we do not have deviations. There are no null values and it is represented in list below. In the second column in figure 7 is count of all null values from a specific column. Also, all data are of numerical type, float64 and int64, as it should be and it suits us perfectly because we are not forced to do conversions to numerical data types. And this is also visible in figure 7.

```
data_frame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 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   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
 30  label                    569 non-null    int64
```

Figure 7. The data type of each column and count of null values of each column

The next step before we split data into training and test set is to standardize data. Data standardization is the process of bringing data into a uniform format that allows research, analyze, utilize the data and to compare scores between diverse types of variables.

```
X = data_frame.drop(columns='label', axis=1)
Y = data_frame['label']

print(X)
```

```
     mean radius  mean texture  ...  worst symmetry  worst fractal dimension
0          17.99         10.38  ...          0.4601                  0.11890
1          20.57         17.77  ...          0.2750                  0.08902
2          19.69         21.25  ...          0.3613                  0.08758
3          11.42         20.38  ...          0.6638                  0.17300
4          20.29         14.34  ...          0.2364                  0.07678
..           ...           ...  ...             ...                      ...
564        21.56         22.39  ...          0.2060                  0.07115
565        20.13         28.25  ...          0.2572                  0.06637
566        16.60         28.08  ...          0.2218                  0.07820
567        20.60         29.33  ...          0.4087                  0.12400
568         7.76         24.54  ...          0.2871                  0.07039
```

Figure 8. Non-standardized data

```
Scaler=StandardScaler()
```

```
Scaler.fit(X)
```

```
StandardScaler()
```

```
Standardized_data=Scaler.transform(X)
```

```
print(Standardized_data)
```

```
[[ 1.09706398 -2.07333501  1.26993369 ...  2.29607613  2.75062224
   1.93701461]
 [ 1.82982061 -0.35363241  1.68595471 ...  1.0870843  -0.24388967
   0.28118999]
 [ 1.57988811  0.45618695  1.56650313 ...  1.95500035  1.152255
   0.20139121]
 ...
 [ 0.70228425  2.0455738   0.67267578 ...  0.41406869 -1.10454895
  -0.31840916]
 [ 1.83834103  2.33645719  1.98252415 ...  2.28998549  1.91908301
   2.21963528]
 [-1.80840125  1.22179204 -1.81438851 ... -1.74506282 -0.04813821
  -0.75120669]]
```

Figure 9. Standardized data

After standardization comes data separating on test and train one. Training ones are nothing than the true data and the testing one are the one we are predicting. The training data is used to create the model based on the samples, and the testing data is used to qualify performance.

```
X_train, X_test, Y_train, Y_test = train_test_split(Standardized_data, Y, test_size=0.2, random_state=2)
```

```
print(X.shape, X_train.shape, X_test.shape)
```

```
(569, 30) (455, 30) (114, 30)
```

Figure 10. Dividing the data set into a training and test set

0.2 in the test size means: 80% of data to be training data and 20% of it to be test data. Then we have this random state=2 which is not important but what it does is each time we run this train test split function our data will be splitted in an unusual way, but if we use this random state value as some number it will be splitted in the same way.

On the next two figures we have presented quality report on both standardized and non-standardized data. In general data quality report is used for indicating how reliable a given dataset is.

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	0.062798
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440

Figure 11. Full quality report on non-standardized data

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension
count	5.690000e+02	5.690000e+02	5.690000e+02	5.690000e+02	5.690000e+02	5.690000e+02	5.690000e+02	5.690000e+02	5.690000e+02	5.690000e+02
mean	-3.162887e-15	-6.530609e-15	-7.078891e-16	-8.799835e-16	6.132177e-15	-1.120369e-15	-4.421380e-16	9.732500e-16	-1.971670e-15	-1.453631e-15
std	1.000880e+00	1.000880e+00	1.000880e+00	1.000880e+00	1.000880e+00	1.000880e+00	1.000880e+00	1.000880e+00	1.000880e+00	1.000880e+00
min	-2.029648e+00	-2.229249e+00	-1.984504e+00	-1.454443e+00	-3.112085e+00	-1.610136e+00	-1.114873e+00	-1.261820e+00	-2.744117e+00	-1.819865e+00
25%	-6.893853e-01	-7.259631e-01	-6.919555e-01	-6.671955e-01	-7.109628e-01	-7.470860e-01	-7.437479e-01	-7.379438e-01	-7.032397e-01	-7.226392e-01
50%	-2.150816e-01	-1.046362e-01	-2.359800e-01	-2.951869e-01	-3.489108e-02	-2.219405e-01	-3.422398e-01	-3.977212e-01	-7.162650e-02	-1.782793e-01
75%	4.693926e-01	5.841756e-01	4.996769e-01	3.635073e-01	6.361990e-01	4.938569e-01	5.260619e-01	6.469351e-01	5.307792e-01	4.709834e-01
max	3.971288e+00	4.651889e+00	3.976130e+00	5.250529e+00	4.770911e+00	4.568425e+00	4.243589e+00	3.927930e+00	4.484751e+00	4.910919e+00

Figure 12. Full quality report on standardized data

C. Modeling

Earlier in the text, we announced that we would use three algorithms and compare their results. After dividing the data set into a training set and a test set, we will use the training set to create three prediction models. Below are the codes we used to create the prediction model using logistic regression, support vector machine and K-nearest neighbor algorithms.

```
[ ]  model = LogisticRegression()
```

```
[ ]  # training the Logistic Regression model using Training data
     model.fit(X_train, Y_train)
     LogisticRegression()
```

Figure 13. Logistic regression prediction model

```
[ ]  classifier = svm.SVC(kernel='linear')
```

```
[ ]  classifier.fit(X_train,Y_train)
     SVC(kernel='linear')
```

Figure 14. Support vector machine prediction model

```
[ ]  knn_clf=KNeighborsClassifier(n_neighbors=5)
```

```
[ ]  knn_clf.fit(X_train,Y_train)
     KNeighborsClassifier()
```

Figure 15. K-nearest neighbor prediction model

D. Evaluation

Confusion matrix is a table that is used to describe the performance of a classification model on a set of test data. In confusion matrix there are 4 various states and we will explain them in our example of breast cancer:

1. true positives (TP) – we predicted they have cancer and they do have cancer
2. true negatives (TN) – we predicted they do not have cancer and they do not have cancer
3. false positives (FP) – we predicted they have the cancer, but they do not have the cancer and this is also known as a "Type I error"
4. false negatives (FN) – we predicted they do not have the cancer but they have the cancer and this is also known as a "Type II error"

The best situation would be if we had many TP and TN cases. If we choose which error is more serious and which more affects the prediction between FP and FN, then it is FN or Type II error. Such cases should be as few as possible.

Rates that are computed from a confusion matrix are:

Accuracy – tells us how often the classifier is correct

$$average\ class\ accuracy = \frac{1}{|levels(t)|} \sum_{l \in levels(t)} recall_l$$

where levels(t) are the set of levels that the target feature, t, can assume; |levels(t)| is the size of this set; and recall l refers to the recall achieved by a model

Recall – tells us how confident we can be that all the instances with positive target (true) have been found by the prediction model

$$recall = \frac{TP}{(TP + FN)}$$

Precision – when it predicts yes, tells us how often it is correct

$$precision = \frac{TP}{(TP + FP)}$$

F1 score – it is harmonic mean of precision and recall and it defines a model's accuracy on a dataset

$$F_1\ measure = 2 \times \frac{(precision \times recall)}{(precision + recall)}$$

WHAT CAN WE READ FROM CONFUSION MATRIX DIAGRAMS AND RATES COMPUTED FROM CONFUSION MATRIX?

There are two possible predicted classes, **yes** (have the cancer) and no (do not have the cancer). The classifier made a total of 569 predictions. These facts are the same for every confusion matrix, no matter if it is based on logistic regression, SVM or K-nearest neighbor.

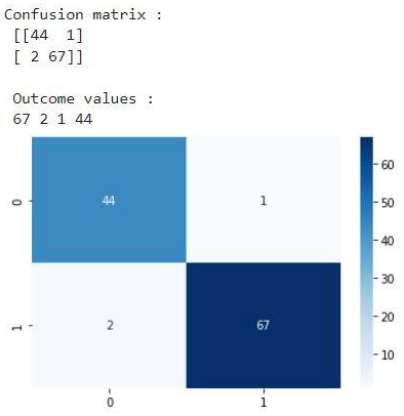

Figure 16. Confusion matrix based on logistic regression prediction model

First confusion matrix diagram is made of data related to logistic regression prediction model, and we can see that out of 114 test cases, the classifier predicted “yes” 68 times and “no” 46 times. In reality, 69 patients in the sample have cancer and 45 patients do not. Count of FN, Type II error is 2.

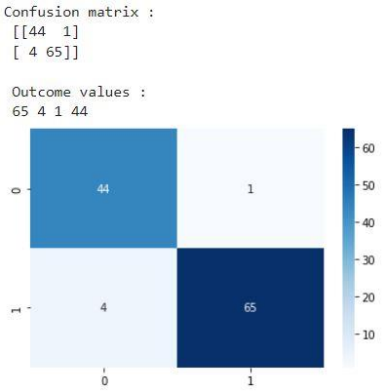

Figure 17. Confusion matrix based on support vector machine prediction model

Next one is based on support vector machine prediction model, and we can see that out of 114 test cases, the classifier predicted “yes” 66 times and “no” 48 times. In reality, 69 patients in the sample have cancer and 45 patients do not. Count of FN, Type II error is 4 and this result in combination with count of FP instances is the worst.

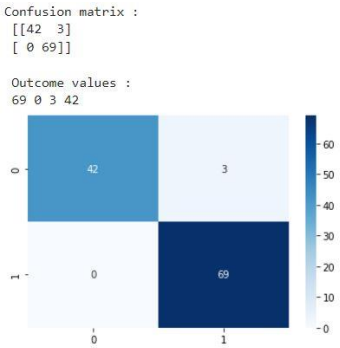

Figure 18. Confusion matrix based on K-nearest neighbor prediction model

The last one is based on K-nearest neighbor algorithm, and we can see that out of 114 test cases, the classifier predicted “yes” 72 times and “no” 42 times. In reality, 69 patients in the sample have cancer and 45 patients do not. Count of FN, Type II error is 0 but count of FP is 3. And these figures show a more favorable prediction than support vector machine figures.

Score/Algorithm	Logistic regression %	Support vector machine %	K-nearest neighbor %
Accuracy	0,97	0,96	0,97
Precision	0,99	0,98	0,96
Recall	0,97	0,94	1,00
F1-score	0,98	0,96	0,98

Figure 19. List of scores for all used algorithms

As we know, **accuracy score** tells us how often the classifier is correct, and we can recognize that logistic regression and K-nearest neighbor have score 97% which is better than score of SVM’s accuracy score. This means that these two classifiers will be correct in 97% of cases out of 100%.
In table there is **recall scores** that tells us how confident we can be that all the instances with positive target (true) have been found by the prediction model and based on that recall score result for K-nearest neighbor classifier is 100% which is the best possible result while SVM has the worst result of these three algorithms, 94%.
When it predicts yes, **precision** will tell us how often it is true, and in this case the best precision score had logistic regression, 99%, but SVM’s and K-nearest neighbor do not have those bad results.
And last one score is **F1 score** which is harmonic mean of precision and recall and it defines a model’s accuracy on a dataset. F1 scores for logistic regression and K-nearest neighbor are equal, 98%, and SVM has worse F1 score, 96%, in fact none of these three F1 scores are bad.

Looking at the results from the table in Figure 16, we can conclude that no statistically significant difference was found in the performance of the algorithms. One of the reasons for this outcome may be the simplicity of the problem that comes with the selected dataset and based on that all the previously mentioned methods are applicable. Certainly, the seriousness of the topic should be considered, as well as the fact that this is a medically significant test, so retests are planned in the future. We did not find any algorithm that stands out and whose results differ significantly from other algorithms, so the conclusion is that to solve the problem of malignant tumor prediction it is best to combine all three algorithms because they offer superior performance. There is also no need for some more complex algorithms such as neural networks.

SUMMARY

To sum up, the main problem was to find an ideal algorithm that would help solve the problem of predicting a malignant tumor. The first and second chapters explain supervised machine learning, as the type of machine learning used and its algorithms, logistic regression, SVM and K-nearest neighbor. After a detailed introduction to the theory, the practical application of all the above was done on the example of the prediction of a malignant tumor, where a comparison of the mentioned algorithms tried to reach the one with the best performance.

REFERENCES

[1] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. US, 2020.

[2] P. Boucher, *Artificial intelligence: How does it work, why does it matter, and what can we do about it?*. Brussels, 2020.

[3] D. Petersson, "*What is Supervised Learning?*", SearchEnterpriseAI, 2021. [Online]. Available:https://www.techtarget.com/searchenterpriseai/definition/supervised-learning. [Accessed: 20- Dec- 2021].

[4] "*Supervised and Unsupervised learning - GeeksforGeeks*", GeeksforGeeks, 2021. [Online]. Available:https://www.geeksforgeeks.org/supervised-unsupervised-learning/. [Accessed: 20- Dec- 2021].

[5] M. W. Berry, A. Mohamed and B. Wah Yap, *Supervised and Unsupervised Learning for Data Science*, 1st ed. Springer International Publishing, 2020.

[6] M. Memon, J. Li, A. Haq, M. Memon and W. Zhou, "Breast Cancer Detection in the IOT Health Environment Using Modified Recursive Feature Selection", *Wireless Communications and Mobile Computing*, vol. 2019, pp. 1-19, 2019. Available:https://www.hindawi.com/journals/wcmc/2019/5176705/tab2/. [Accessed 22 December 2021].

[7] "Precision and recall - Wikipedia", *En.wikipedia.org*, 2022. [Online]. Available: https://en.wikipedia.org/wiki/Precision_and_recall. [Accessed: 23- Dec- 2021].