**DOKUZ EYLÜL UNIVERSITY**

**ENGINEERING FACULTY**

**DEPARTMENT OF COMPUTER ENGINEERING**

**CME 4403**

**INTRODUCTION TO MACHINE LEARNING**

**BREAST CANCER PREDICTION**

**by**

**Şeniz AKBULUT**

**Dr.Özge KART**

**İzmir 2021**

**ABSTRACT**

Breast cancer (BC) is one of the most common cancers among women worldwide, representing the majority of new cancer cases and cancer-related deaths according to global statistics, making it a significant public health problem in today’s society.

The early diagnosis of BC can improve the prognosis and chance of survival significantly, as it can promote timely clinical treatment to patients. Further accurate classification of benign tumors can prevent patients undergoing unnecessary treatments. Thus, the correct diagnosis of BC and classification of patients into malignant or benign groups is the subject of much research. Because of its unique advantages in critical features detection from complex BC datasets, machine learning (ML) is widely recognized as the methodology of choice in BC pattern classification and forecast modelling.

Classification and data mining methods are an effective way to classify data. Especially in medical field, where those methods are widely used in diagnosis and analysis to make decisions.

In this draft report, the dataset to be used for breast cancer prediction and the features of this data will be mentioned.

# CHAPTER ONE

INTRODUCTION

Breast cancer is the most common cancer in women. According to current information, one out of every eight to nine women gets breast cancer at some point in their life.

Today, with early diagnosis and early treatment, it has been reached to the point that breast cancer ceases to be a fatal disease.

Mammography. The most important screening test for breast cancer is the mammogram. A mammogram is an X-ray of the breast. It can detect breast cancer up to two years before the tumor can be felt by you or your doctor.

Women age 40–45 or older who are at average risk of breast cancer should have a mammogram once a year.Women at high risk should have yearly mammograms along with an MRI starting at age 30.

The economic and social values of Breast Cancer Diagnosis (BCD) are very high. As a result, the problem has attracted many researchers in the area of computational intelligence recently.

This machine learning project uses a dataset that can help determine the likelihood that a breast tumor is malignant or benign. Various factors are taken into consideration, including the lump's thickness, number of bare nuclei, and mitosis.

Using the Breast Cancer Wisconsin (Diagnostic) dataset that contains all these data, we can create a classifier that can help diagnose patients and predict the likelihood of a breast cancer.

* 1. **So** **Some Risk Factors for Breast Cancer**

The following are some of the known risk factors for breast cancer. However, most cases of breast cancer cannot be linked to a specific cause.

**Age.** The chance of getting breast cancer increases as women age. Nearly 80 percent of breast cancers are found in women over the age of 50.

**Personal history of breast cancer.** A woman who has had breast cancer in one breast is at an increased risk of developing cancer in her other breast.

**Family history of breast cancer.** A woman has a higher risk of breast cancer if her mother, sister or daughter had breast cancer, especially at a young age (before 40). Having other relatives with breast cancer may also raise the risk.

**Genetic factors.** Women with certain genetic mutations, including changes to the BRCA1 and BRCA2 genes, are at higher risk of developing breast cancer during their lifetime. Other gene changes may raise breast cancer risk as well.

**Childbearing and menstrual history.** The older a woman is when she has her first child, the greater her risk of breast cancer. Also at higher risk are:

* Women who menstruate for the first time at an early age (before 12)
* Women who go through menopause late (after age 55)
* Women who’ve never had children

methodology will be given in details about the user groups and the users perspective and solution techniques.

# CHAPTER TWO

dataset

1. **Basic Information About Dataset**

I will use the UCI Machine Learning Repository for breast cancer dataset.  
<http://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+%28diagnostic%29>

The dataset used in this story is publicly available and was created by Dr. William H. Wolberg, physician at the University Of Wisconsin Hospital at Madison, Wisconsin, USA. To create the dataset Dr. Wolberg used fluid samples, taken from patients with solid breast masses and an easy-to-use graphical computer program called Xcyt, which is capable of perform the analysis of cytological features based on a digital scan. The program uses a curve-fitting algorithm, to compute ten features from each one of the cells in the sample, than it calculates the mean value, extreme value and standard error of each feature for the image, returning a 30 real-valuated vector.

**2.2 Dataset Summary**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Data Set Characteristics:** | Multivariate | **Number of Instances:** | 569 | **Area:** | Life |
| **Attribute Characteristics:** | Real | **Number of Attributes:** | 32 | **Date Donated** | 1995-11-01 |
| **Associated Tasks:** | Classification | **Missing Values?** | No | **Number of Web Hits:** | 1495927 |

**Data Set Characteristics**

Multivariate data consist of individual measurements that are acquired as a function of more than two variables, for example, kinetics measured at many wavelengths and as a function of temperature, or as a function of pH, or as a function of initial concentrations, and so forth, of the reacting solutions.

**Attribute Characteristics**

Ordinal:

Quantitative in nature; numeric codings may not necessarily be ordinal (quantitative). Further analysis. discrete (integer or real) values , vary within a fixed range of values.

**Associated Tasks**

Classification predictive modeling involves assigning a class label to input examples. Binary classification refers to predicting one of two classes and multi-class classification involves predicting one of more than two classes.

**Number of Instances**

There are 569 lines (samples) in the data set.

**Number Of Attributes**

Each row has 32 columns (attributes).

1. ID number 2) Diagnosis (M = malignant, B = benign) 3–32)

Ten real-valued features are computed for each cell nucleus:

1. radius (mean of distances from center to points on the perimeter)

2. texture (standard deviation of gray-scale values)

3. perimeter

4. area

5. smoothness (local variation in radius lengths)

6. compactness (perimeter² / area — 1.0)

7. concavity (severity of concave portions of the contour)

8. concave points (number of concave portions of the contour)

9. symmetry

10. fractal dimension (“coastline approximation” — 1)

The mean, standard error and “worst” or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius.

All feature values are recoded with four significant digits.

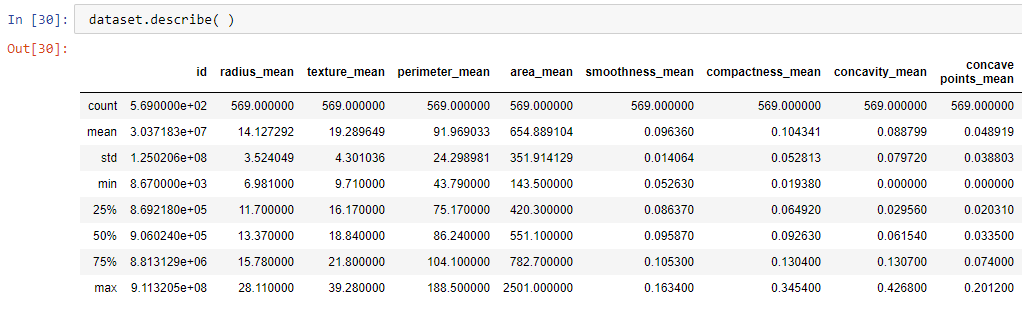
**Missing Values**

There is no missing value in this data set.

# CHAPTER three

data qualıty

A data quality report includes tabular reports that describe the characteristics of each feature in an ABT using standard statistical measures of central tendency and variation.



**3.1 List of Continuous Features**

radius\_mean , texture\_mean , perimeter\_mean , area\_mean , smoothness\_mean , compactness\_mean , concavity\_mean , concave points\_mean , symmetry\_mean , fractal\_dimension\_mean , radius\_se , texture\_se , perimeter\_se , area\_se , smoothness\_se , compactness\_se , concavity\_se , concave points\_se , symmetry\_se , fractal\_dimension\_se , radius\_worst , texture\_worst , perimeter\_worst , area\_worst , smoothness\_worst , compactness\_worst , concavity\_worst , concave points\_worst , symmetry\_worst , fractal\_dimension\_worst

**3.2 List of Categorical Features**

Diagnosis

**3.3 Data Quality Report for Categorical Features**

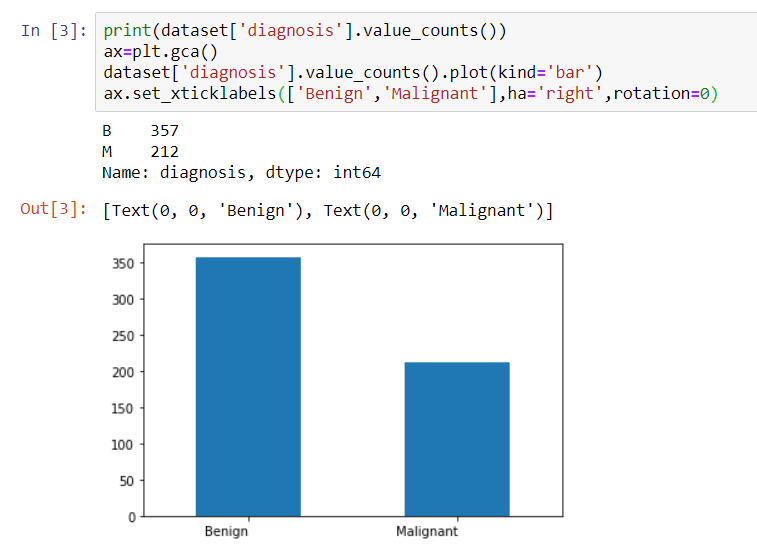
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature** | **Count** | **%Miss** | **Card.** | **Mode** | **ModeFreq** | **Mode%** | **2ndMode** | **2ndModeFreq** | **2ndMode%** |
| Diagnostic | 569 | 0.0 | 3 | B | 357 | 62,74 | M | 212 | 37,26 |

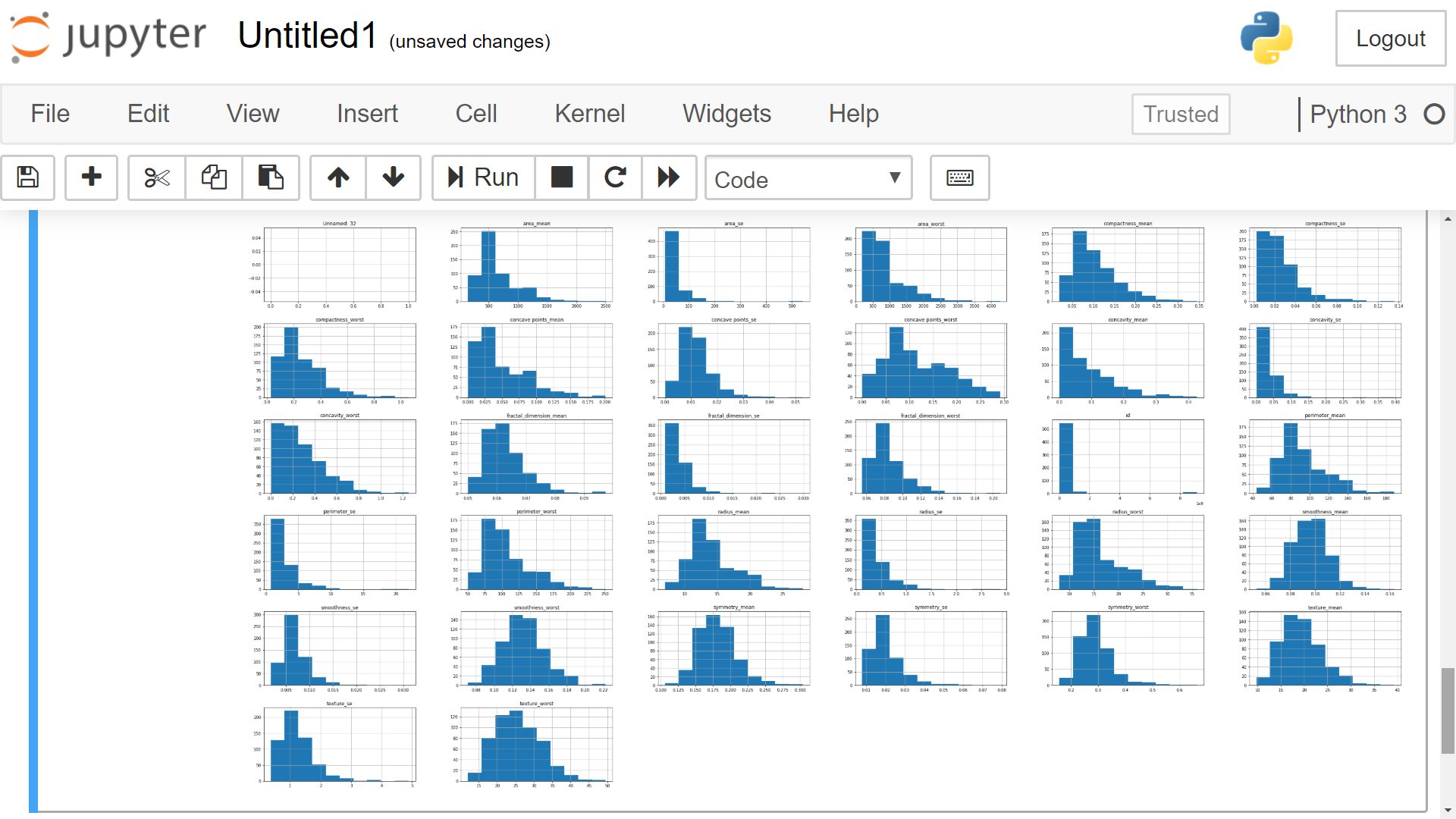
**3.4 Data Quality Report for Continuous Features**

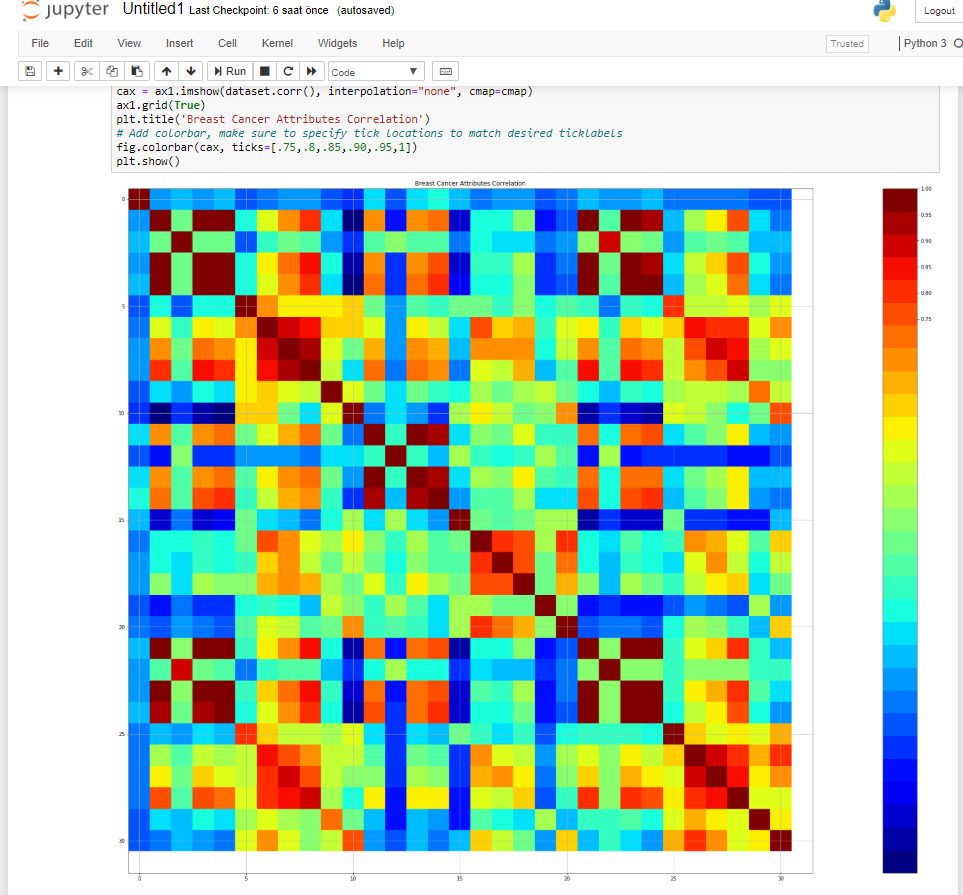
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature** | **Count** | **%Miss** | **Card.** | **Min** | **1st Qrt** | **Mean** | **Median** | **3rd Qrt** | **Max** | **Std.Dev.** |
| radius\_mean | 569 | 0.0 | 457 | 6,981 | 11,7 | 14,12729 | 13,37 | 15,78 | 28,11 | 3,520951 |
| texture\_mean | 569 | 0.0 | 480 | 9,71 | 16,17 | 19,28965 | 18,84 | 21,8 | 39,28 | 4,297255 |
| perimeter\_mean | 569 | 0.0 | 523 | 43,79 | 75,17 | 91,96903 | 86,24 | 104,1 | 188,5 | 24,27762 |
| area\_mean | 569 | 0.0 | 540 | 143,5 | 420,3 | 654,8891 | 551,1 | 782,7 | 2501 | 351,6048 |
| smoothness\_mean | 569 | 0.0 | 475 | 0,05263 | 0,08637 | 0,09636 | 0,09587 | 0,1053 | 0,1634 | 0,014052 |
| compactness\_mean | 569 | 0.0 | 538 | 0,01938 | 0,06492 | 0,104341 | 0,09263 | 0,1304 | 0,3454 | 0,052766 |
| concavity\_mean | 569 | 0.0 | 538 | 0 | 0,02956 | 0,088799 | 0,06154 | 0,1307 | 0,4268 | 0,07965 |
| concave points\_mean | 569 | 0.0 | 543 | 0 | 0,02031 | 0,048919 | 0,0335 | 0,074 | 0,2012 | 0,038769 |
| symmetry\_mean | 569 | 0.0 | 433 | 0,106 | 0,1619 | 0,181162 | 0,1792 | 0,1957 | 0,304 | 0,02739 |
| fractal\_dimension\_mean | 569 | 0.0 | 500 | 0,04996 | 0,0577 | 0,062798 | 0,06154 | 0,06612 | 0,09744 | 0,007054 |
| radius\_se | 569 | 0.0 | 541 | 0,1115 | 0,2324 | 0,405172 | 0,3242 | 0,4789 | 2,873 | 0,277069 |
| texture\_se | 569 | 0.0 | 520 | 0,3602 | 0,8339 | 1,216853 | 1,108 | 1,474 | 4,885 | 0,551163 |
| perimeter\_se | 569 | 0.0 | 534 | 0,757 | 1,606 | 2,866059 | 2,287 | 3,357 | 21,98 | 2,020077 |
| area\_se | 569 | 0.0 | 529 | 6,802 | 17,85 | 40,33708 | 24,53 | 45,19 | 542,2 | 45,45101 |
| smoothness\_se | 569 | 0.0 | 548 | 0,001713 | 0,005169 | 0,007041 | 0,00638 | 0,008146 | 0,03113 | 0,003 |
| compactness\_se | 569 | 0.0 | 542 | 0,002252 | 0,01308 | 0,025478 | 0,02045 | 0,03245 | 0,1354 | 0,017892 |
| concavity\_se | 569 | 0.0 | 534 | 0 | 0,01509 | 0,031894 | 0,02589 | 0,04205 | 0,396 | 0,03016 |
| concave points\_se | 569 | 0.0 | 508 | 0 | 0,007638 | 0,011796 | 0,01093 | 0,01471 | 0,05279 | 0,006165 |
| symmetry\_se | 569 | 0.0 | 499 | 0,007882 | 0,01516 | 0,020542 | 0,01873 | 0,02348 | 0,07895 | 0,008259 |
| fractal\_dimension\_se | 569 | 0.0 | 546 | 0,000895 | 0,002248 | 0,003795 | 0,003187 | 0,004558 | 0,02984 | 0,002644 |
| radius\_worst | 569 | 0.0 | 458 | 7,93 | 13,01 | 16,26919 | 14,97 | 18,79 | 36,04 | 4,828993 |
| texture\_worst | 569 | 0.0 | 512 | 12,02 | 21,08 | 25,67722 | 25,41 | 29,72 | 49,54 | 6,140854 |
| perimeter\_worst | 569 | 0.0 | 515 | 50,41 | 84,11 | 107,2612 | 97,66 | 125,4 | 251,2 | 33,573 |
| area\_worst | 569 | 0.0 | 545 | 185,2 | 515,3 | 880,5831 | 686,5 | 1084 | 4254 | 568,8565 |
| smoothness\_worst | 569 | 0.0 | 412 | 0,07117 | 0,1166 | 0,132369 | 0,1313 | 0,146 | 0,2226 | 0,022812 |
| compactness\_worst | 569 | 0.0 | 530 | 0,02729 | 0,1472 | 0,254265 | 0,2119 | 0,3391 | 1,058 | 0,157198 |
| concavity\_worst | 569 | 0.0 | 540 | 0 | 0,1145 | 0,272188 | 0,2267 | 0,3829 | 1,252 | 0,208441 |
| concave points\_worst | 569 | 0.0 | 493 | 0 | 0,06493 | 0,114606 | 0,09993 | 0,1614 | 0,291 | 0,065675 |
| symmetry\_worst | 569 | 0.0 | 501 | 0,1565 | 0,2504 | 0,290076 | 0,2822 | 0,3179 | 0,6638 | 0,061813 |
| fractal\_dimension\_worst | 569 | 0.0 | 536 | 0,05504 | 0,07146 | 0,083946 | 0,08004 | 0,09208 | 0,2075 | 0,018045 |

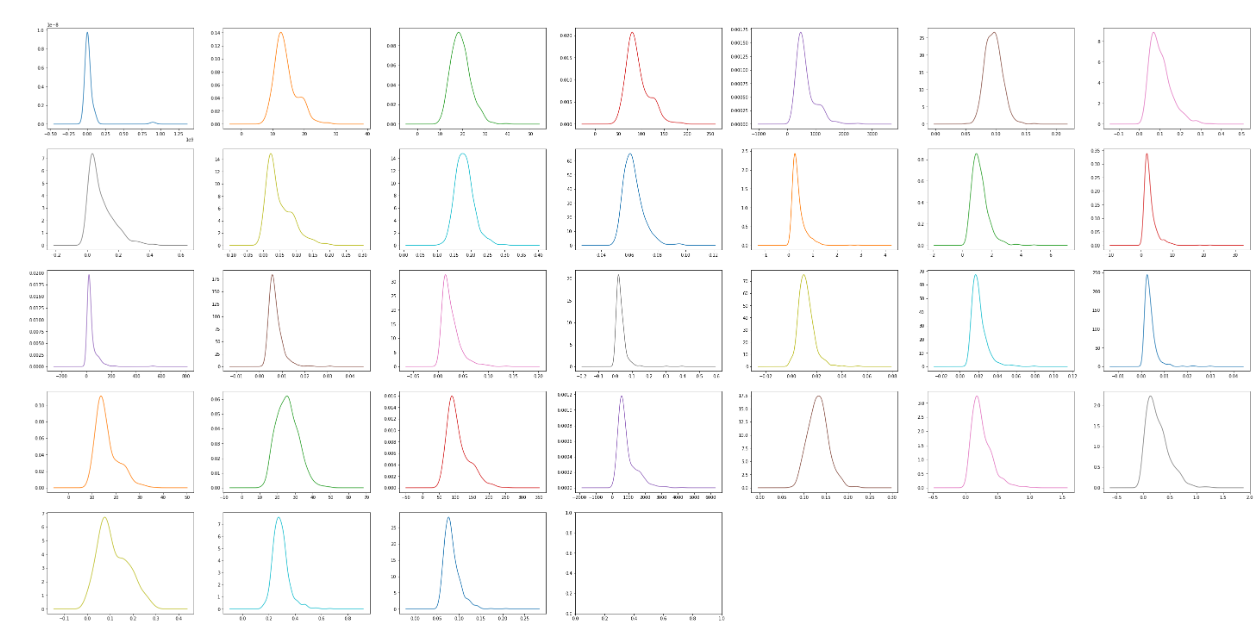
# CHAPTER FOUR

vısualızatıon









# CHAPTER FIVE

PRE PROCESSING

**Missing Data**

There is no missing value in the data set used, so it is not necessary to handle it.



**Noisy Data**

There are many areas in the data set that will not be used in the estimation mechanism. Generally, various factors such as tuber thickness, number of bare nuclei and mitosis are taken into consideration in breast cancer analyzes. These areas are important for us. The areas that will not be used have been simplified.

**Splitting Dataset**

The data we use is usually split into training data and test data. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. We have the test dataset (or subset) in order to test our model’s prediction on this subset.

I will do this using SciKit-Learn library in Python using the train\_test\_split method.

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Future Scalling**

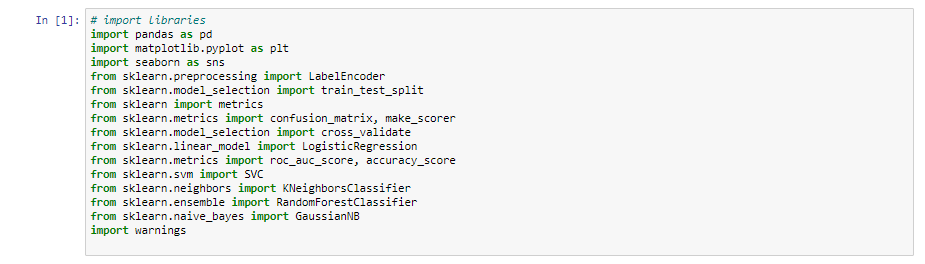
Most of the times, your dataset will contain features highly varying in magnitudes, units and range. But since, most of the machine learning algorithms use Eucledian distance between two data points in their computations. We need to bring all features to the same level of magnitudes. This can be achieved by scaling. This means that you’re transforming your data so that it fits within a specific scale, like 0–100 or 0–1.

I will use StandardScaler method from SciKit-Learn library.

# CHAPTER SIX

IMPLEMENTATION

**1.Libraries**



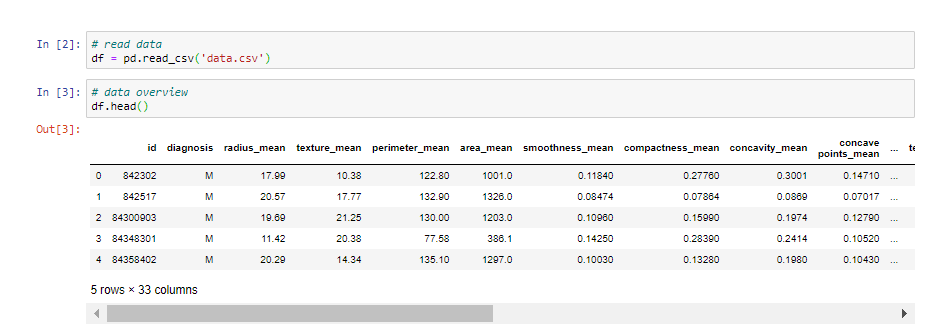
**Pandas:** Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

**Mathplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

**Seaborn:** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

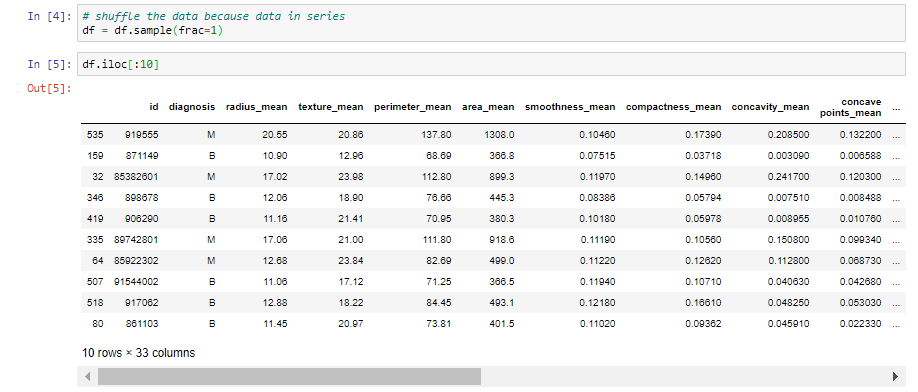
**Sklearn:** Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language.It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

**2. Reading and Viewing the Dataset**



**Kaggle** allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges. The "Breast Cancer Wisconsin (Diagnostic) Data Set" published on kaggle was used in the project. This data set is available to researchers as a .csv extension. With the use of Dataset.head(), we can view the first 5 rows of the data set to see that the data is kept in order.

**3. Collation of Data**



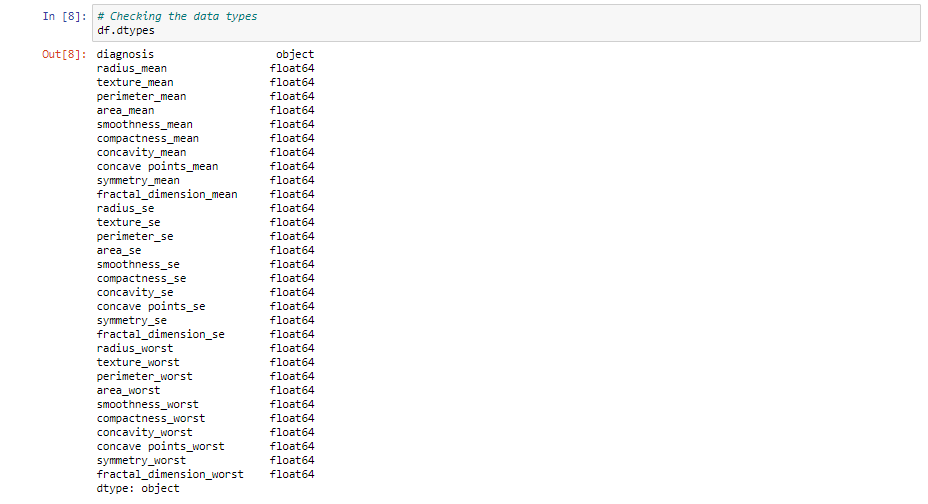
Since the dataset is created in a regular manner, the data in the dataset is mixed to display the result in the most complex case. We then take 10 samples from the blended data to classify on it.

**4. Cleaning the Dataset**



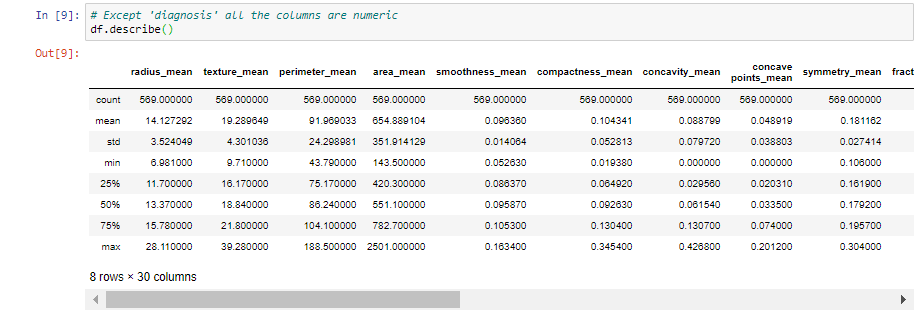
There is no need for the 32 numbered unnamed column and the id column in the data set. For this reason, the data set has been simplified by removing it from the data set. As a result, we have 31 columns and 569 rows of data.

**5. Data Type Classification and Control**

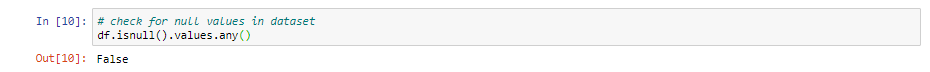


The data types of the existing columns in the data set are checked. While the Diagnosis column receives string(object) values encoded as 'B' and 'M', all other columns hold float type values.

**6. Generate Descriptive Statistics**

  
Descriptive statistics include those that summarize the central tendency, dispersion and shape of a dataset’s distribution, excluding NaN values.

Analyzes both numeric and object series, as well as DataFrame column sets of mixed data types. The output will vary depending on what is provided.

**6. Checking and Clearing Missing Values** 

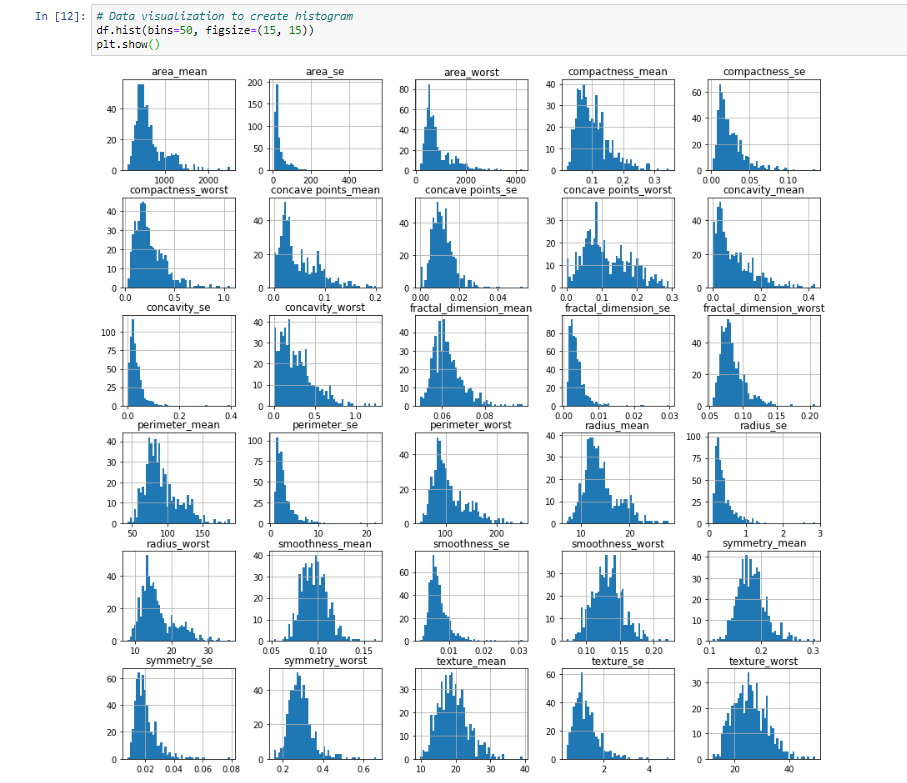
There are no null values in the data set used.

**7.Class Label Frequency Calculation**



Diagnosis is a categorical feature. Basically, it divides the data into two categories, 'B' and 'M'. Before creating a histogram, class label frequency calculation should be done at the category level.

**8.Data Visualization**



Data visualization is the representation of data or information in a graph, chart, or other visual format. It communicates relationships of the data with images. This is important because it allows trends and patterns to be more easily seen. With the rise of big data upon us, we need to be able to interpret increasingly larger batches of data. Machine learning makes it easier to conduct analyses such as predictive analysis, which can then serve as helpful visualizations to present.

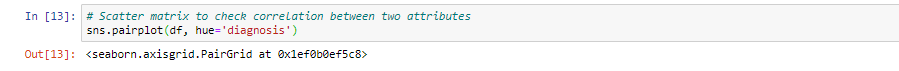
For this reason, histogram graphs of all the features in the data set were created.

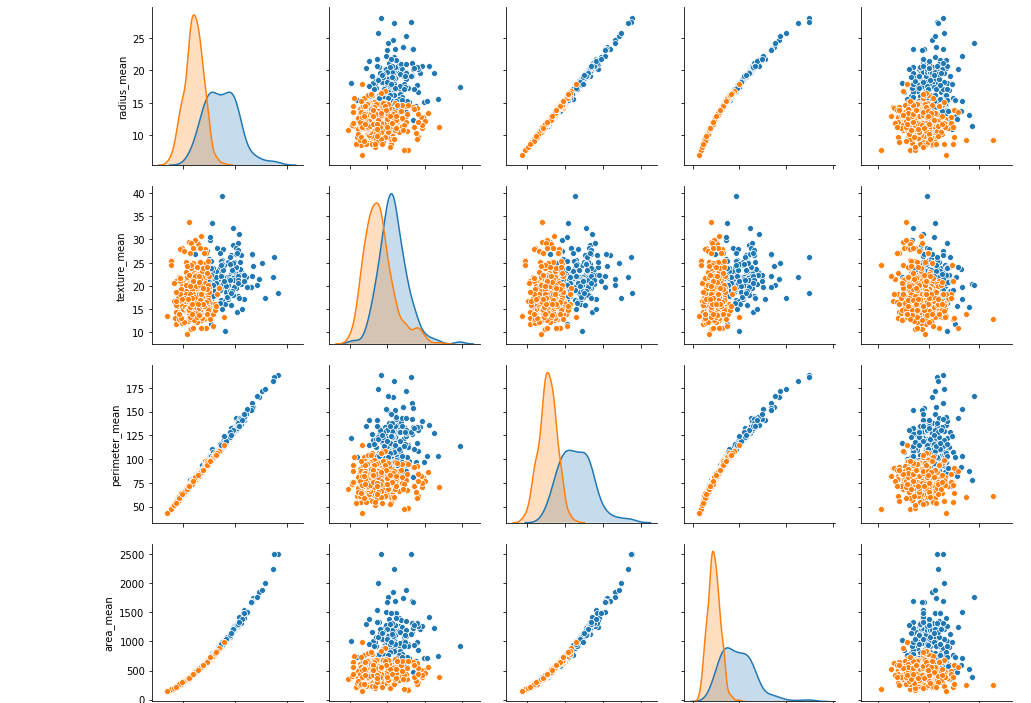
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | radius\_mean |  | | texture\_mean |  | | perimeter\_mean |  | | area\_mean |  | | smoothness\_mean | | | compactness\_mean | | | concavity\_mean |  | | concave points\_mean | | | symmetry\_mean |  | | fractal\_dimension\_mean | | | radius\_se |  | | texture\_se |  | | perimeter\_se |  | | area\_se |  | | smoothness\_se |  | | compactness\_se |  | | concavity\_se |  | | concave points\_se | | | symmetry\_se |  | | fractal\_dimension\_se | | | radius\_worst |  | | texture\_worst |  | | perimeter\_worst |  | | area\_worst |  | | smoothness\_worst | | | compactness\_worst | | | concavity\_worst |  | | concave points\_worst | | | symmetry\_worst |  | | fractal\_dimension\_worst | | | 🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪  🡪 | Exponential  Unimodal (skewed right)  Exponential  Normal (Unimodal)  Unimodal (skewed right)  Exponential  Multimodal  Normal (Unimodal)  Exponantial  Unimodal (skewed right)  Exponential  Exponential  Unimodal (skewed right)  Exponential  Unimodal (skewed right)  Unimodal (skewed right)  Exponential  Multimodal  Unimodal (skewed right)  Unimodal (skewed right)  Multimodal  Multimodal  Normal (Unimodal)  Exponential  Unimodal (skewed right)  Exponential  Unimodal (skewed right)  Unimodal (skewed right)  Normal (Unimodal)  Normal (Unimodal)  Normal (Unimodal)  Unimodal (skewed right) |

🡪 **Know The Data**

**9. Correlation Check Between Features**

Correlation explains how one or more variables are related to each other. These variables can be input data features which have been used to forecast our target variable. Correlation, statistical technique which determines how one variables moves/changes in relation with the other variable. It gives us the idea about the degree of the relationship of the two variables. It’s a bi-variate analysis measure which describes the association between different variables. In most of the business it’s useful to express one subject in terms of its relationship with others.

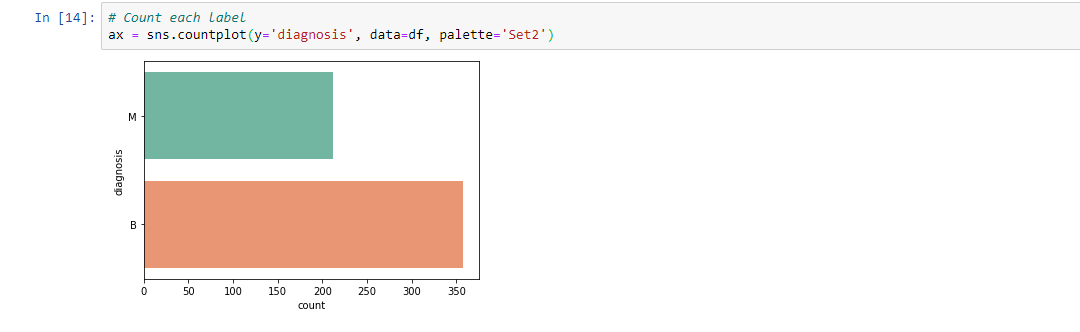


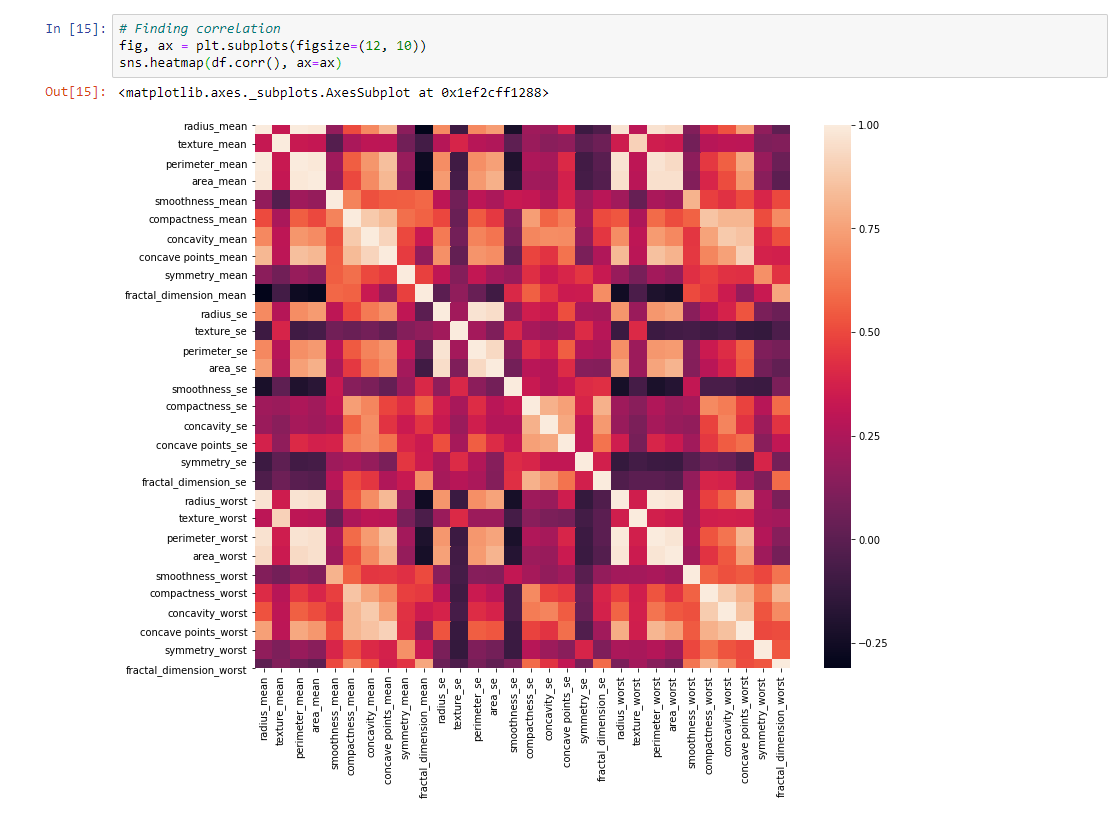


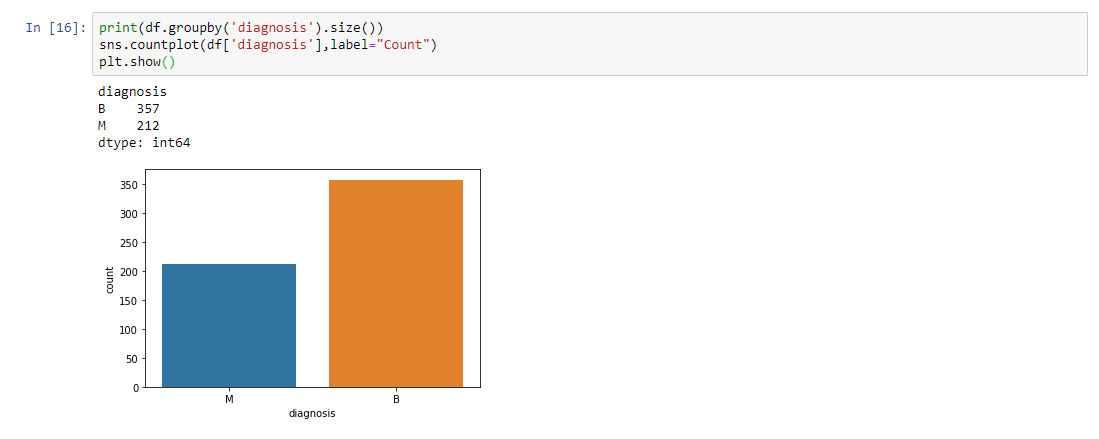
Japon sürgülü kapısı, pencere, bina, insanlar içeren bir resim

Açıklama otomatik olarak oluşturuldu

**10. Visualization of the Class Label Frequency Calculation**



**11. Visualization of the Correlation Check Between Features** 



**12.Visualization with Boxplot**



A boxplot is a standardized way of displaying the distribution of data based on a five number summary (“minimum”, first quartile (Q1), median, third quartile (Q3), and “maximum”). It can tell you about your outliers and what their values are. It can also tell you if your data is symmetrical, how tightly your data is grouped, and if and how your data is skewed.

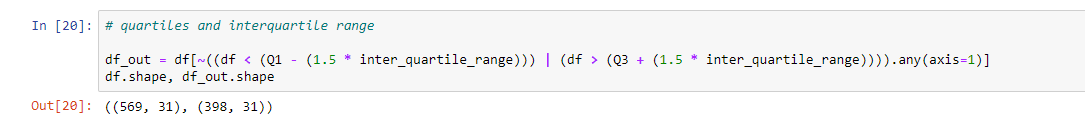
For some distributions/datasets, you will find that you need more information than the measures of central tendency (median, mean, and mode).

You need to have information on the variability or dispersion of the data. A boxplot is a graph that gives you a good indication of how the values in the data are spread out. Although boxplots may seem primitive in comparison to a histogram or density plot, they have the advantage of taking up less space, which is useful when comparing distributions between many groups or datasets.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

**13. Defining and Calculating Quartile Values**





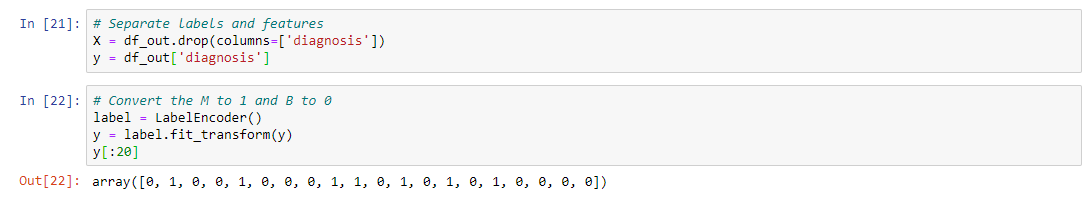
**Median (Q2/50th Percentile):** the middle value of the dataset.

**First quartile (Q1/25th Percentile):** the middle number between the smallest number (not the “minimum”) and the median of the dataset.

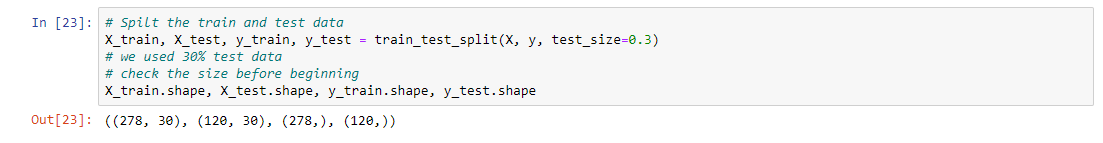
**Third quartile (Q3/75th Percentile): the** middle value between the median and the highest value (not the “maximum”) of the dataset.

**Interquartile range (IQR):** 25th to the 75th percentile.

**14. Convert Categorical Feature to Numeric Value**



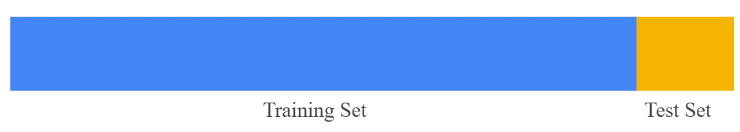
**15. Separation of Target Feature and Descriptive Features**



training set—a subset to train a model.

test set—a subset to test the trained model.

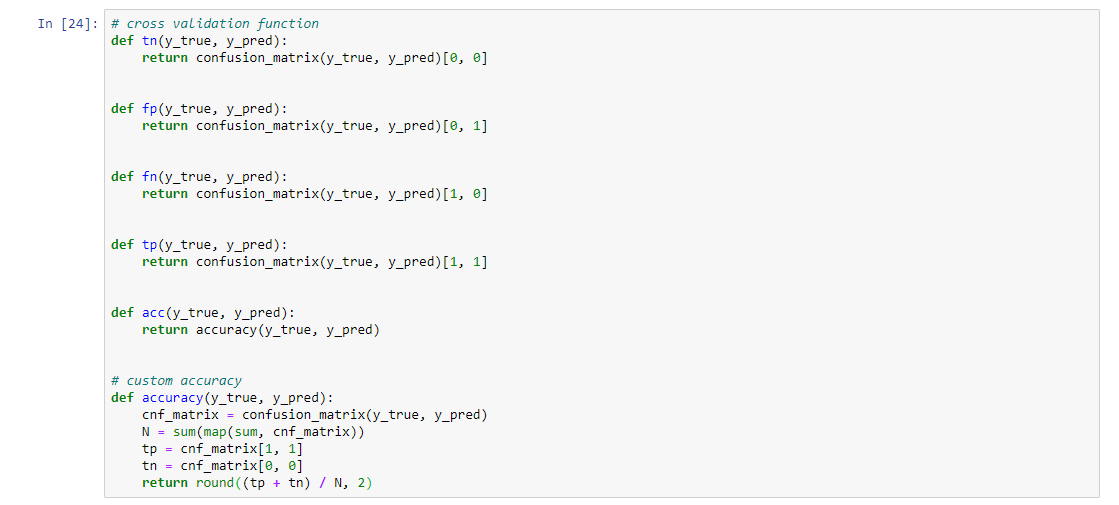
You could imagine slicing the single data set as follows:

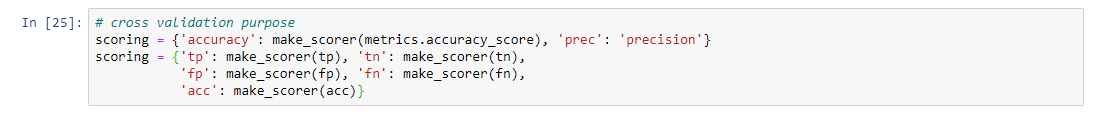


Assuming that your test set meets the preceding two conditions, your goal is to create a model that generalizes well to new data. Our test set serves as a proxy for new data.

**16.Cross Validation**

This process of deciding whether the numerical results quantifying hypothesized relationships between variables, are acceptable as descriptions of the data, is known as validation. Generally, an error estimation for the model is made after training, better known as evaluation of residuals. In this process, a numerical estimate of the difference in predicted and original responses is done, also called the training error. However, this only gives us an idea about how well our model does on data used to train it. Now its possible that the model is underfitting or overfitting the data. So, the problem with this evaluation technique is that it does not give an indication of how well the learner will generalize to an independent/ unseen data set. Getting this idea about our model is known as Cross Validation.





**17.Predictive Modeling Techniques**

Predictive modeling, also called predictive analytics, is a mathematical process that seeks to predict future events or outcomes by analyzing patterns that are likely to forecast future results. The goal of predictive modeling is to answer this question: "Based on known past behavior, what is most likely to happen in the future?

Once data has been collected, the analyst selects and trains statistical models, using historical data. Although it may be tempting to think that big data makes predictive models more accurate, statistical theorems show that, after a certain point, feeding more data into a predictive analytics model does not improve accuracy. The old saying "All models are wrong, but some are useful" is often mentioned in terms of relying solely on predictive models to determine future action. In many use cases, including weather predictions, multiple models are run simultaneously and results are aggregated to create one final prediction. This approach is known as ensemble modeling. As additional data becomes available, the statistical analysis will either be validated or revised.

**Naïve Bayes :**

Naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features (see Bayes classifier). They are among the simplest Bayesian network models, but coupled with kernel density estimation, they can achieve higher accuracy levels.

Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

In the statistics and computer science literature, naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naïve Bayes is not (necessarily) a Bayesian method.

**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, although many more complex extensions exist. In regression analysis, logistic regression[1] (or logit regression) is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labeled "0" and "1". In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names. Analogous models with a different sigmoid function instead of the logistic function can also be used, such as the probit model; the defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each independent variable having its own parameter; for a binary dependent variable this generalizes the odds ratio.

**Support Vector Machine:**

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. Developed at AT&T Bell Laboratories by Vladimir Vapnik with colleagues (Boser et al., 1992, Guyon et al., 1993, Vapnik et al., 1997), SVMs are one of the most robust prediction methods, being based on statistical learning frameworks or VC theory proposed by Vapnik (1982, 1995) and Chervonenkis (1974). Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). SVM maps training examples to points in space so as to maximise the width of the gap between the two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

**KNN:**

In statistics, the k-nearest neighbors algorithm (k-NN) is a non-parametric classification method first developed by Evelyn Fix and Joseph Hodges in 1951, and later expanded by Thomas Cover.It is used for classification and regression. In both cases, the input consists of the k closest training examples in data set. The output depends on whether k-NN is used for classification or regression:

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

k-NN is a type of classification where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically

Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for k-NN classification) or the object property value (for k-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A peculiarity of the k-NN algorithm is that it is sensitive to the local structure of the data.

**Random Forest :**

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

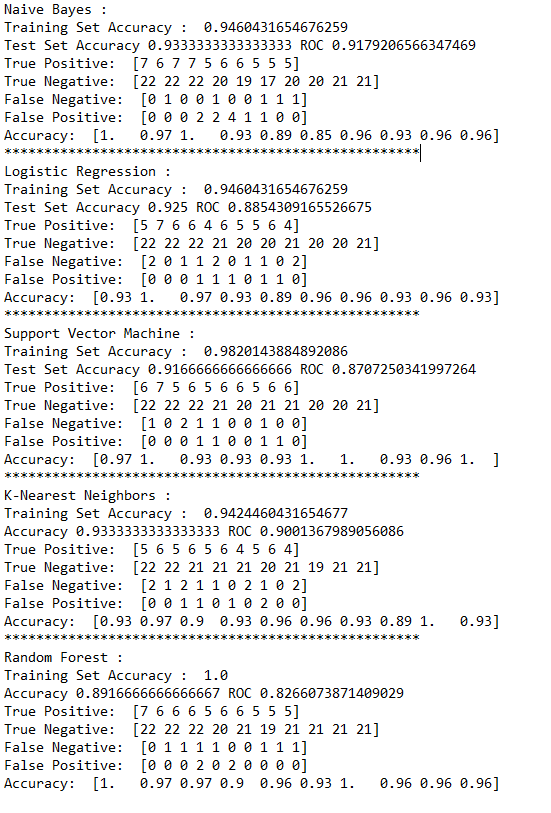
The first algorithm for random decision forests was created in 1995 by Tin Kam Ho using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.). The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho and later independently by Amit and Gemanin order to construct a collection of decision trees with controlled variance.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.



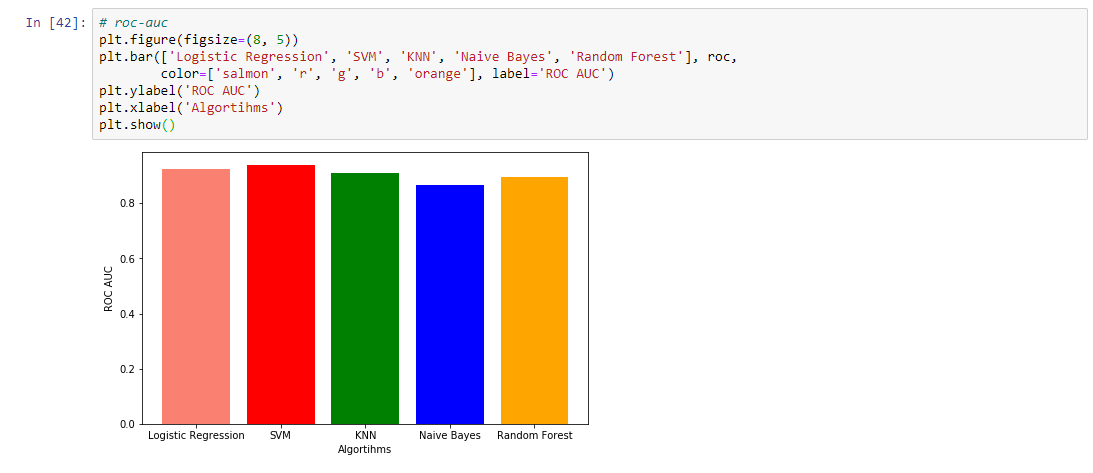




**18. Comparison of Predictive Model Algorithms**



According to the Accuracy Score evaluation, the best algorithm is SVM while the worst algorithm is Naive Bayes. Logistic Regression KNN and Random Forest have almost the same accuracy.



An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

True Positive Rate

False Positive Rate

ROC AUC evaluation results are almost the same as accuracy score evaluation. The best score is SVM and the worst score is Naive Bayes. Other results are very close to each other.

# CHAPTER Seven

code

# import libraries

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.model\_selection import train\_test\_split

from sklearn import metrics

from sklearn.metrics import confusion\_matrix, make\_scorer

from sklearn.model\_selection import cross\_validate

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import roc\_auc\_score, accuracy\_score

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.naive\_bayes import GaussianNB

import warnings

# read data

df = pd.read\_csv('data.csv')

# data overview

df.head()

# shuffle the data because data in series

df = df.sample(frac=1)

df.iloc[:10]

# drop the unnamed and id columns.

df = df.drop(columns=['Unnamed: 32', 'id'])

# no of rows and columns

df.shape

# Checking the data types

df.dtypes

# Except 'diagnosis' all the columns are numeric

df.describe()

# check for null values in dataset

df.isnull().values.any()

# Count class labels

df['diagnosis'].value\_counts()

# Data visualization to create histogram

df.hist(bins=50, figsize=(15, 15))

plt.show()

# Scatter matrix to check correlation between two attributes

sns.pairplot(df, hue='diagnosis')

# Count each label

ax = sns.countplot(y='diagnosis', data=df, palette='Set2')

# Finding correlation

fig, ax = plt.subplots(figsize=(12, 10))

sns.heatmap(df.corr(), ax=ax)

print(df.groupby('diagnosis').size())

sns.countplot(df['diagnosis'],label="Count")

plt.show()

# box plot to check outlier in each category

def boxPlot(dff):

d = dff.drop(columns=['diagnosis'])

for column in d:

plt.figure(figsize=(5, 2))

sns.boxplot(x=column, data=d, palette="colorblind")

boxPlot(df)

# Quartile range

Q1 = df.quantile(0.25)

Q3 = df.quantile(0.75)

inter\_quartile\_range = Q3 - Q1

# quartiles and interquartile range

df\_out = df[~((df < (Q1 - (1.5 \* inter\_quartile\_range))) | (df > (Q3 + (1.5 \* inter\_quartile\_range)))).any(axis=1)]

df.shape, df\_out.shape

# Separate labels and features

X = df\_out.drop(columns=['diagnosis'])

y = df\_out['diagnosis']

# Convert the M to 1 and B to 0

label = LabelEncoder()

y = label.fit\_transform(y)

y[:20]

# Spilt the train and test data

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3)

# we used 30% test data

# check the size before beginning

X\_train.shape, X\_test.shape, y\_train.shape, y\_test.shape

# cross validation function

def tn(y\_true, y\_pred):

return confusion\_matrix(y\_true, y\_pred)[0, 0]

def fp(y\_true, y\_pred):

return confusion\_matrix(y\_true, y\_pred)[0, 1]

def fn(y\_true, y\_pred):

return confusion\_matrix(y\_true, y\_pred)[1, 0]

def tp(y\_true, y\_pred):

return confusion\_matrix(y\_true, y\_pred)[1, 1]

def acc(y\_true, y\_pred):

return accuracy(y\_true, y\_pred)

# custom accuracy

def accuracy(y\_true, y\_pred):

cnf\_matrix = confusion\_matrix(y\_true, y\_pred)

N = sum(map(sum, cnf\_matrix))

tp = cnf\_matrix[1, 1]

tn = cnf\_matrix[0, 0]

return round((tp + tn) / N, 2)

# cross validation purpose

scoring = {'accuracy': make\_scorer(metrics.accuracy\_score), 'prec': 'precision'}

scoring = {'tp': make\_scorer(tp), 'tn': make\_scorer(tn),

'fp': make\_scorer(fp), 'fn': make\_scorer(fn),

'acc': make\_scorer(acc)}

def print\_result(result):

print("True Positive: ", result['test\_tp'])

print("True Negative: ", result['test\_tn'])

print("False Negative: ", result['test\_fn'])

print("False Positive: ", result['test\_fp'])

print("Accuracy: ", result['test\_acc'])

# acc list and rcc list stores the output of each model

acc = []

roc = []

# Naive Bayes

model = GaussianNB()

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

y\_pred\_train = model.predict(X\_train)

ac = accuracy\_score(y\_test, y\_pred)

acc.append(ac)

ac\_train = accuracy\_score(y\_train, y\_pred\_train)

rc = roc\_auc\_score(y\_test, y\_pred)

roc.append(rc)

print("Naive Bayes : ")

print("Training Set Accuracy : ", ac\_train)

print("Test Set Accuracy {0} ROC {1}".format(ac, rc))

# cross validation

result = cross\_validate(model, X\_train, y\_train, scoring=scoring, cv=10)

print\_result(result)

# Logistic Regression

LR = LogisticRegression()

LR.fit(X\_train, y\_train)

LR.score(X\_train, y\_train)

y\_pred = LR.predict(X\_test)

y\_pred\_train = model.predict(X\_train)

# find accuracy

ac = accuracy\_score(y\_test, y\_pred)

acc.append(ac)

ac\_train = accuracy\_score(y\_train, y\_pred\_train)

# Code for ROC\_AUC curve

rc = roc\_auc\_score(y\_test, y\_pred)

roc.append(rc)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("Logistic Regression : ")

print("Training Set Accuracy : ", ac\_train)

print("Test Set Accuracy {0} ROC {1}".format(ac, rc))

# cross validation shows how accurate our model is

result = cross\_validate(LR, X\_train, y\_train, scoring=scoring, cv=10)

print\_result(result)

# Support Vector Machine

model = SVC(gamma='auto', kernel='linear')

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

y\_pred\_train = model.predict(X\_train)

ac = accuracy\_score(y\_test, y\_pred)

acc.append(ac)

ac\_train = accuracy\_score(y\_train, y\_pred\_train)

rc = roc\_auc\_score(y\_test, y\_pred)

roc.append(rc)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("Support Vector Machine : ")

print("Training Set Accuracy : ", ac\_train)

print("Test Set Accuracy {0} ROC {1}".format(ac, rc))

# cross validation

result = cross\_validate(model, X\_train, y\_train, scoring=scoring, cv=10)

print\_result(result)

# KNN

model = KNeighborsClassifier(n\_neighbors=9, n\_jobs=-1)

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

y\_pred\_train = model.predict(X\_train)

ac = accuracy\_score(y\_test, y\_pred)

acc.append(ac)

ac\_train = accuracy\_score(y\_train, y\_pred\_train)

rc = roc\_auc\_score(y\_test, y\_pred)

roc.append(rc)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("K-Nearest Neighbors : ")

print("Training Set Accuracy : ", ac\_train)

print("Accuracy {0} ROC {1}".format(ac, rc))

# cross validation

result = cross\_validate(model, X\_train, y\_train, scoring=scoring, cv=10)

print\_result(result)

# Random Forest

model = RandomForestClassifier(n\_estimators=20, max\_depth=10)

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

y\_pred\_train = model.predict(X\_train)

ac = accuracy\_score(y\_test, y\_pred)

acc.append(ac)

ac\_train = accuracy\_score(y\_train, y\_pred\_train)

rc = roc\_auc\_score(y\_test, y\_pred)

roc.append(rc)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("Random Forest : ")

print("Training Set Accuracy : ", ac\_train)

print("Accuracy {0} ROC {1}".format(ac, rc))

# cross validation

result = cross\_validate(model, X\_train, y\_train, scoring=scoring, cv=10)

print\_result(result)

# Plot the bar graph for accuracy and roc-auc

# accuracy score

plt.figure(figsize=(8, 5))

plt.bar(['Logistic Regression', 'SVM', 'KNN', 'Naive Bayes', 'Random Forest'], acc,

color=['salmon', 'r', 'g', 'b', 'orange'], label='Accuracy')

plt.ylabel('Accuracy Score')

plt.xlabel('Algortihms')

# roc-auc

plt.figure(figsize=(8, 5))

plt.bar(['Logistic Regression', 'SVM', 'KNN', 'Naive Bayes', 'Random Forest'], roc,

color=['salmon', 'r', 'g', 'b', 'orange'], label='ROC AUC')

plt.ylabel('ROC AUC')

plt.xlabel('Algortihms')

plt.show()