Higher Technology Institute

10th of ramadan city

Checkmate Breast Cancer

Graduation Project

CSC400

SEP/JAN(2018/2019)

Menna Allah Anwer Ali (42015113)
Amira Ahmed Hafez (42015141)
Sarah Muhammed Abdelaziz(42015049)
Nada Alaa Muhammed (42014129)
Muhammed Esam Omar (42015091)

Supervised by: Dr/Eid Abd Elhakem

Acknowledgment

First and formost , I am very gratefull To Allah.

This Thesis would not have been possible Without Support and Encouragement of supervisor **Doctor/Eid Abdelhakem** and **Doctor/Rania Ragab.**

.

Abstract

.

Checkmate breast cancer disk top application, The project aims to shed light on the diseases that women face and neglect, where in some cases the life of some people may end due to lack of correct diagnosis, accuracy and speed diagnosis.
This project will help doctors in the of diagnosis and high accuracy and thus the speed of treatment and this is what the cancer patient needs. And rapid identification if this tumor is malignant or benign.

Key Terms

PyQt	Python QT
ER Diagram	Entity Relation Diagram.
UML	Unified Modeling Language.
DMBS	Data Base Management System.
SQL	Structure Query Language.
SQlite	C-Language Liabrary that implement SQL data base.
QML	Module provides a frame network for developing application and liabrary with the QML language.

List of content

List of figures

Chapter One

Introduction

1.1 Breast Cancer.

Breast cancer starts when cells in the breast begin to grow out of control. These cells usually form a tumor that can often be seen on an x-ray or felt as a lump. The tumor is malignant (cancer) if the cells can grow into (invade) surrounding tissues or spread (metastasize) to distant areas of the body. Breast cancer occurs almost entirely in women, but men can get breast cancer, too.

Cells in nearly any part of the body can become cancer and can spread to other areas. To learn more about cancer and how all cancers start and spread, see Cancer Basics.

Figure 1.1 Breast cancer ribbon

1.1.1 Where breast cancer starts:

Breast cancers can start from different parts of the breast. Most breast cancers begin in the ducts that carry milk to the nipple (ductal cancers). Some start in the glands that make breast milk (lobular cancers). There are also other types of breast cancer that are less common.

A small number of cancers start in other tissues in the breast. These cancers are called sarcomas and lymphomas and are not really thought of as breast cancers.

Although many types of breast cancer can cause a lump in the breast, not all do. Many breast cancers are found on screening mammograms which can detect cancers at an earlier stage, often before they can be felt, and before symptoms develop. There are other symptoms of breast cancer you should watch for and report to a health care provider.

It's also important to understand that most breast lumps are benign and not cancer (malignant). Non-cancerous breast tumors are abnormal growths, but they do not spread outside of the breast and they are not life threatening. But some benign breast lumps can increase a woman's risk of getting breast cancer. Any breast lump or change needs to be checked by a health care professional to determine if it is benign or malignant (cancer) and if it might affect your future cancer risk.

1.1.2 breast cancer spreads:

Breast cancer can spread when the cancer cells get into the blood or lymph system and are carried to other parts of the body.

The lymph system is a network of lymph (or lymphatic) vessels found throughout the body that connects lymph nodes (small bean-shaped collections of immune system cells). The clear fluid inside the lymph vessels, called lymph, contains tissue by-products and waste material, as well as immune system cells. The lymph vessels carry lymph fluid away from the breast. In the case of breast cancer, cancer cells can enter those lymph vessels and start to grow in lymph nodes. Most of the lymph vessels of the breast drain into:

- Lymph nodes under the arm (axillary nodes)
- Lymph nodes around the collar bone (supraclavicular [above the collar bone] and infraclavicular [below the collar bone] lymph nodes)
- Lymph nodes inside the chest near the breast bone (internal mammary lymph nodes)

1.1.3 What is the project?

The project aims to shed light on the diseases that women face and neglect, where in some cases the life of some people may end due to lack of correct diagnosis, accuracy and speed diagnosis.
This project will help doctors in the of diagnosis and high accuracy and thus the speed of treatment and this is what the cancer patient needs. And rapid identification if this tumor is malignant or benign.

1.1.4 Problem Description:

In the past few years doctor's diagnosis for human's Breast Cancer have been leading a lot of them stray and causing massive casualties, the primary target of this project is to eliminate this problem by using technology to make a very precise and safe diagnosis for different types of breast cancer by making a desktop app that takes numerical data from the diffuse optical tomography and entering the data into the program to identify if the cancer is benign or malignant (in other words Cancerous).

1.1.5 General Information about breast cancer:

- The ratio of female to male breast cancer is approximately 100:1 and this means that there is little awareness among men, and even among physicians, regarding the occurrence of breast cancer in males.

- Male breast cancer accounts for 1% of all breast cancer cases, and men tend to be diagnosed at an older age than women (mean age is about 67 years). Several risk factors have been identified, such as genetic and hormonal abnormalities.

- Younger women usually don't think about getting breast cancer After all, under 7% of all breast cancer cases happen in women under 40. But it can happen at any age, and it's important to be aware of your risk factors, regardless of your age.

1.1.6 Tumor:

A tumor is a mass of abnormal tissue. There are two types of breast cancer tumors: those that are non-cancerous, or 'benign', and those that are cancerous, which are 'malignant'.

1.1.7 Benign Tumors:

When a tumor is diagnosed as benign, doctors will usually leave it alone rather than remove it. Even though these tumors are not generally aggressive toward surrounding tissue, occasionally they may continue to grow, pressing on organs and causing pain or other problems. In these situations, the tumor is removed, allowing pain or complications to subside.

1.1.8 Malignant tumors:

Malignant tumors are cancerous and aggressive because they invade and damage surrounding tissue. When a tumor is suspected to be malignant, the doctor will perform a biopsy to determine the severity or aggressiveness of the tumor.

1.1.9 Metastatic cancer:

Metastatic cancer is when cancer cells of a malignant tumor spread to other parts of the body, usually through the lymph system, and form a secondary tumor.

1.1.10 Tumor Grades:

Tumor grading is a system used to classify a malignant breast cancer tumor based upon the severity of the mutation and the likelihood that it will spread. The breast cancer cells are examined under a microscope to determine, among other factors, how closely the breast cancer cells resemble the healthy cells (called the histologic grade) and the shape and size of the tumor cells' nuclei (called the nuclear grade) as well as how rapidly those cells divide and multiply.

When dealing with breast cancer, tumors are often graded based on a scale of one to three indicating how aggressive the cancerous cells are:

- Low grade – Well-diffentiated
- Intermediate grade – Moderately differentiated
- High grade – Poorly differentiated

1.2 Software usage.

1.2.1 python:

Python is a high-level programming language, with applications in numerous areas, including web programming, scripting, scientific computing, and artificial intelligence.

It is very popular and used by organizations such as Google, NASA, the CIA, and Disney

1.2.2 Qt:

Qt is a cross-platform development framework written in C++
Can be used in several programming languages through bindings

- Ruby
- Java
- Perl
- Python → PyQt

The Qt Toolkit is a collection of classes for various purposes :

- Database management
- XML
- WebKit
- Multimedia
- Networking

For desktop, mobile and embedded development

- Used by more than 350,000 commercial and open source developers
- Backed by Qt consulting, support and training
- Trusted by over 6,500 companies worldwide

1.2.3 PyQt library:

What is PyQt?

PyQt is a GUI widgets toolkit. It is a Python interface for Qt, one of the most powerful, and popular cross-platform GUI libraries. PyQt is a blend of Python programming language and the Qt library.
PyQt is used to write all kinds of GUI applications, from accounting applications, to visualization tools used by scientists and engineers.

PyQt is compatible with all the popular operating systems including Windows, Linux, and Mac OS. It is dual licensed, available under GPL as well as commercial license

Python is probably the easiest to learn and nicest scripting language in widespread use, and Qt is probably the best library for developing GUI applications. The combination of Python and Qt, "PyQt", makes it possible to develop applications on any supported platform and run them unchanged on all the supported platforms—for example, all modern versions of Windows, Linux, Mac OS X, and most Unix-based systems. No compilation is required, and no source code changes to adapt to different operating systems are required thanks to Qt abstracting away the platform-specific details. We only have to copy the source file or files to a target machine that has both Python and PyQt installed and the application will run.

Why PyQt?

Qt is more than a GUI toolkit. It includes abstractions of network sockets, threads, Unicode, regular expressions, SQL databases, SVG, OpenGL, XML, a fully functional web browser, a help system, a multimedia framework, as well as a rich collection of GUI widgets.
Qt classes employ a signal/slot mechanism for communicating between objects that is type safe but loosely coupled making it easy to create re-usable software components.
Qt also includes Qt Designer, a graphical user interface designer. PyQt is able to generate Python code from Qt Designer. It is also possible to add new GUI controls written in Python to Qt Designer.
Python is a simple but powerful object-orientated language. Its simplicity makes it easy to learn, but its power means that large and complex

applications can be created. Its interpreted nature means that Python programmers are very productive because there is no

edit/compile/link/run development cycle.

1.2.4 The Machine Learning Library we use is :

Scikit learn :Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language.[3] 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.

The Software Environment we use

Jupyter Notebook: Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebooks documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.
Jupyter notebooks document can be converted to a number of open standard output formats (HTML, presentation
slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through 'Download As' in the web interface, via the nbconvert library or 'jupyter nbconvert' command line interface in a shell.
To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the file and display it to the user.
Jupyter Notebook provides a browser-based REPL built upon a number of popular open-source libraries.

1.2.5 Machine Learning with scikit-Learn:

In general, a learning problem considers a set of n samples of data and then tries to predict properties of unknown data. If each sample is more than a single number and, for instance, a multi-dimensional entry (aka multivariate data), it is said to have several attributes or features.

Learning problems fall into a few categories:

- supervised learning, in which the data comes with additional attributes that we want to predict,This problem can be either:
- classification: samples belong to two or more classes and we want to learn from already labeled data how to predict the class of unlabeled data. An example of a classification problem would be handwritten digit recognition, in which the aim is to assign each input vector to one of a finite number of discrete categories. Another way to think of classification is as a discrete (as opposed to continuous) form of supervised learning where one has a limited number of categories and for each of the n samples provided, one is to try to label them with the correct category or class.
- regression: if the desired output consists of one or more continuous variables, then the task is called regression. An example of a regression problem would be the prediction of the length of a salmon as a function of its age and weight.
- unsupervised learning, in which the training data consists of a set of input vectors x without any corresponding target values. The goal in such problems may be to discover groups of similar examples within the data, where it is called clustering, or to determine the distribution of data within the input space, known as density estimation, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization

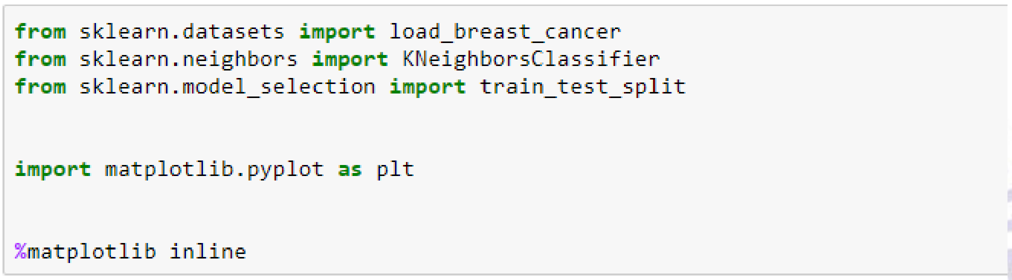
1.2.6 Training set and testing set:

Machine learning is about learning some properties of a data set and then testing those properties against another data set. A common practice in machine learning is to evaluate an algorithm by splitting a data set into two. We call one of those sets the training set, on which we learn some properties; we call the other set the testing set, on which we test the learned properties.

Loading an example dataset.

scikit-learn comes with a few standard datasets, for instance the iris and digits datasets for classification and the boston house prices dataset for regression.

In the following, we start a Python interpreter from our shell and then load the iris and digits datasets. Our notational convention is that $ denotes the shell prompt while >>> denotes the Python interpreter prompt:

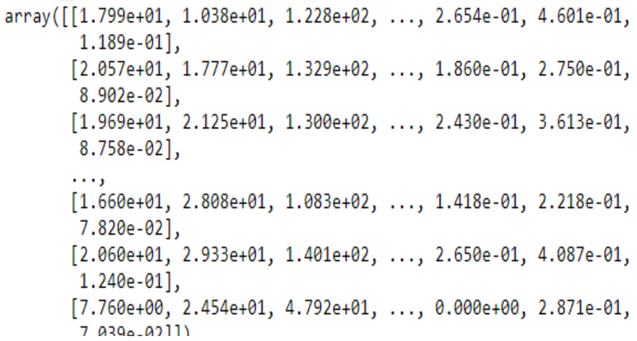


```
from sklearn.datasets import load_breast_cancer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split


import matplotlib.pyplot as plt


%matplotlib inline
```

Figure 1.2 training code

A dataset is a dictionary-like object that holds all the data and some metadata about the data. This data is stored in the .data member, which is a n_samples, n_features array. In the case of supervised problem, one or more response variables are stored in the .target member. More details on the different datasets can be found in the dedicated section.

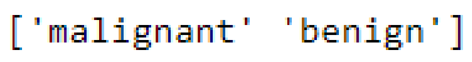
For instance, in the case of the digits dataset, digits.data gives access to the features that can be used to classify the digits samples



```
array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
        1.189e-01],
       [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
        8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
        8.758e-02],
       ...,
       [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
        7.820e-02],
       [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
        1.240e-01],
       [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
        7.039e-02]])
```

Figure 1.3 testing code

and digits.target gives the ground truth for the digit dataset, that is the number corresponding to each digit image that we are trying to learn:

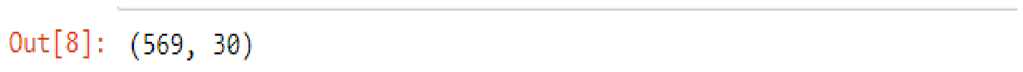


```
['malignant' 'benign']
```

Figure 1.4 malignant and benign

1.2.7 Shape of the data arrays:

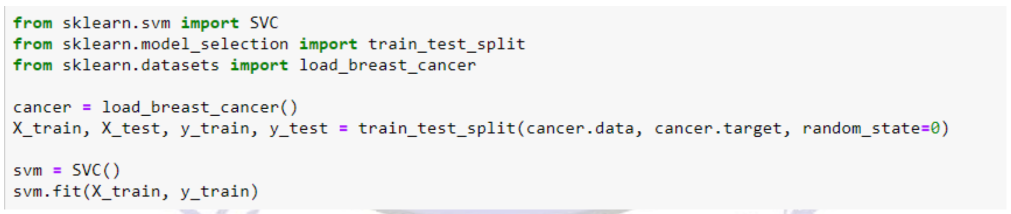
The data is always a 2D array, shape (n_samples, n_features), although the original data may have had a different shape. In the case of the digits, each original sample is a dataset of shape (569, 30) and can be accessed using:



```
Out[8]: (569, 30)
```

In the case of the digits dataset, the task is to predict, given a dataset, which digit it represents. We are given samples of each of the two possible classes (malignant and begnin) on which we *fit* an estimator to be able to *predict* the classes to which unseen samples belong.

But before training the dataset we split it by 80%-20% into train and test data by train_test_split function in skitlearn and calculate the accuracy of the train data and the accuracy of the test data to avoid the over fitting and to have a data to test our model by it.



```
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_breast_cancer

cancer = load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer.target, random_state=0)

svm = SVC()
svm.fit(X_train, y_train)
```

Figure 1.5 train test

In scikit-learn, an estimator for classification is a Python object that implements the methods fit(X, y) and predict(T).

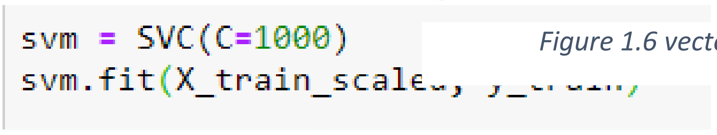
An example of an estimator is the class sklearn.svm.SVC, which implements

Support vector classification:

- Family of machine-learning algorithms that are used for mathematical and engineering problems including for example handwriting digit recognition, object recognition, speaker identification, face detections in images and target detection.

- SVM performs classification by constructing an N-dimensional hyper plane that optimally separates the data into two categories.

For now, we will consider the estimator as

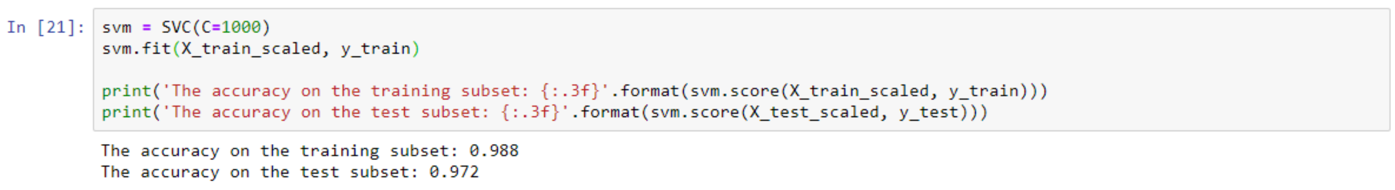


```
svm = SVC(C=1000)
svm.fit(X_train_scaled, y_train)
```

Figure 1.6 vector classification

1.2.8 Choosing the parameters of the model:

In this example, we set the value of gamma manually. To find good values for these parameters, we can use tools such as grid search and cross validation.
The clf (for classifier) estimator instance is first fitted to the model; that is, it must *learn* from the model. This is done by passing our training set to the fit method. For the training set, we'll use all the digits from our dataset.



```
In [21]: svm = SVC(C=1000)
         svm.fit(X_train_scaled, y_train)

         print('The accuracy on the training subset: {:.3f}'.format(svm.score(X_train_scaled, y_train)))
         print('The accuracy on the test subset: {:.3f}'.format(svm.score(X_test_scaled, y_test)))

         The accuracy on the training subset: 0.988
         The accuracy on the test subset: 0.972
```

Figure 1.7 data set parameter

1.3 Analysis.

1.3.1 Use case

Main use case:

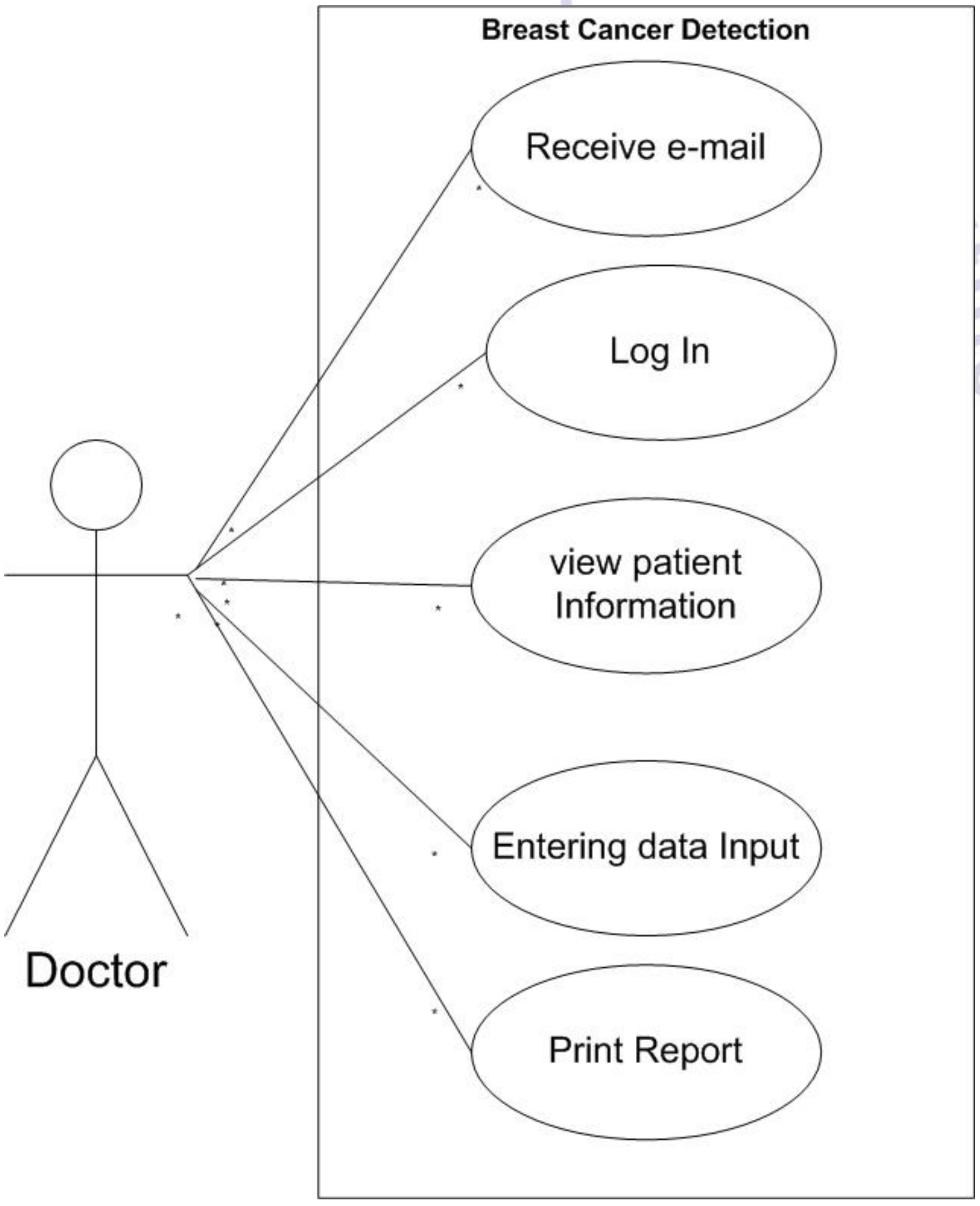


Figure 1.8 : Doctor Use case

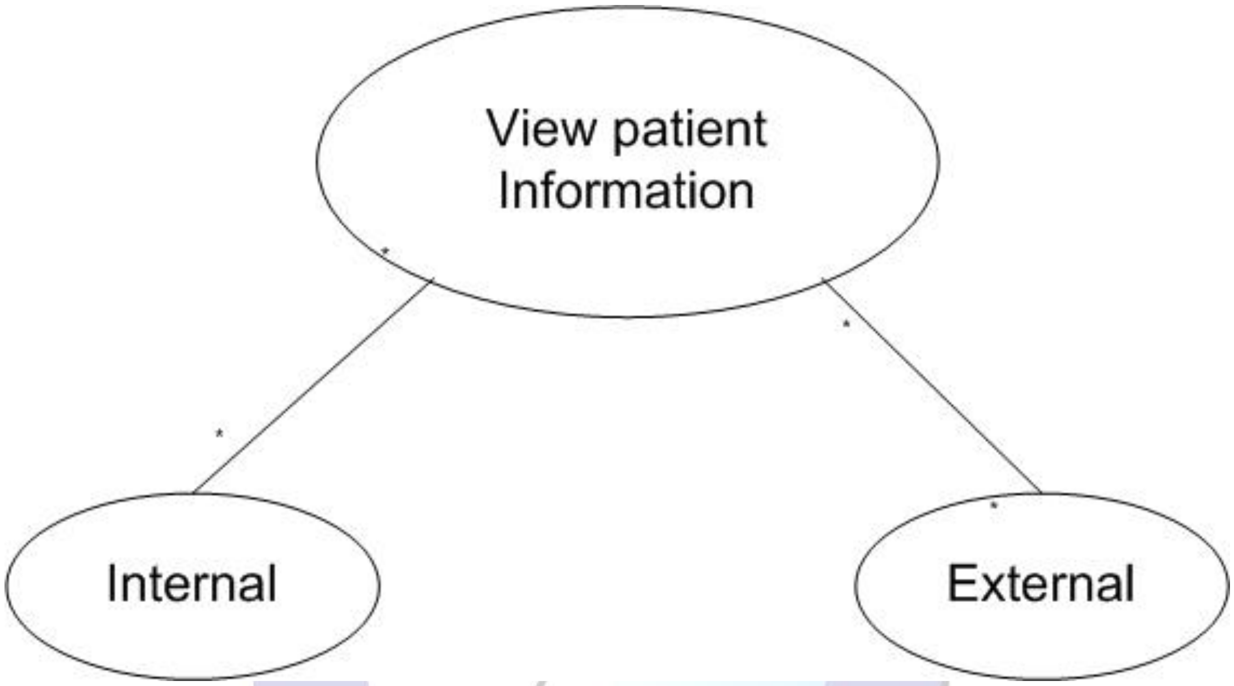
System	Checkmate Breast Cancer Application
Use-case	Receive e-mail
Actor	Doctor
Description	The doctor receive e-mail from admin and log in through him
Stimulus	Receive e-mail
Response	Log in to the application

System	Checkmate Breast Cancer Application
Use-case	Login
Actor	Doctor
Description	The doctor write the username and password to log in to application
Stimulus	enter the name and password
Response	It opens a pateint information window

System	Checkmate Breast Cancer Application
Use-case	View patient information
Actor	Doctor
Description	A page divided by two parts of the first part, Enternal, and the second part is External
Stimulus	The first part enter the SSN patient to viw patient history , and second page create new patiant.
Response	Its open history of patient window.

System	Checkmate Breast Cancer Application
Use-case	Entering data Information
Actor	Doctor
Description	The doctor entering the data information of patient (sample, mammography, FNA, blood analysis)
Stimulus	The doctor inserts the numbers from the mammography or sample or FNA or blood analysis.
Response	determined whether it is benign or malignant

System	Checkmate Breast Cancer Application
Use-case	Print Report
Actor	Doctor
Description	Written in which patient data and explanation of the condition of disease and final diagnosis
Stimulus	After you write the data by clicking Print Report
Response	The report is print


View patient Information
*
*
*
*
Internal
External

Main use case :

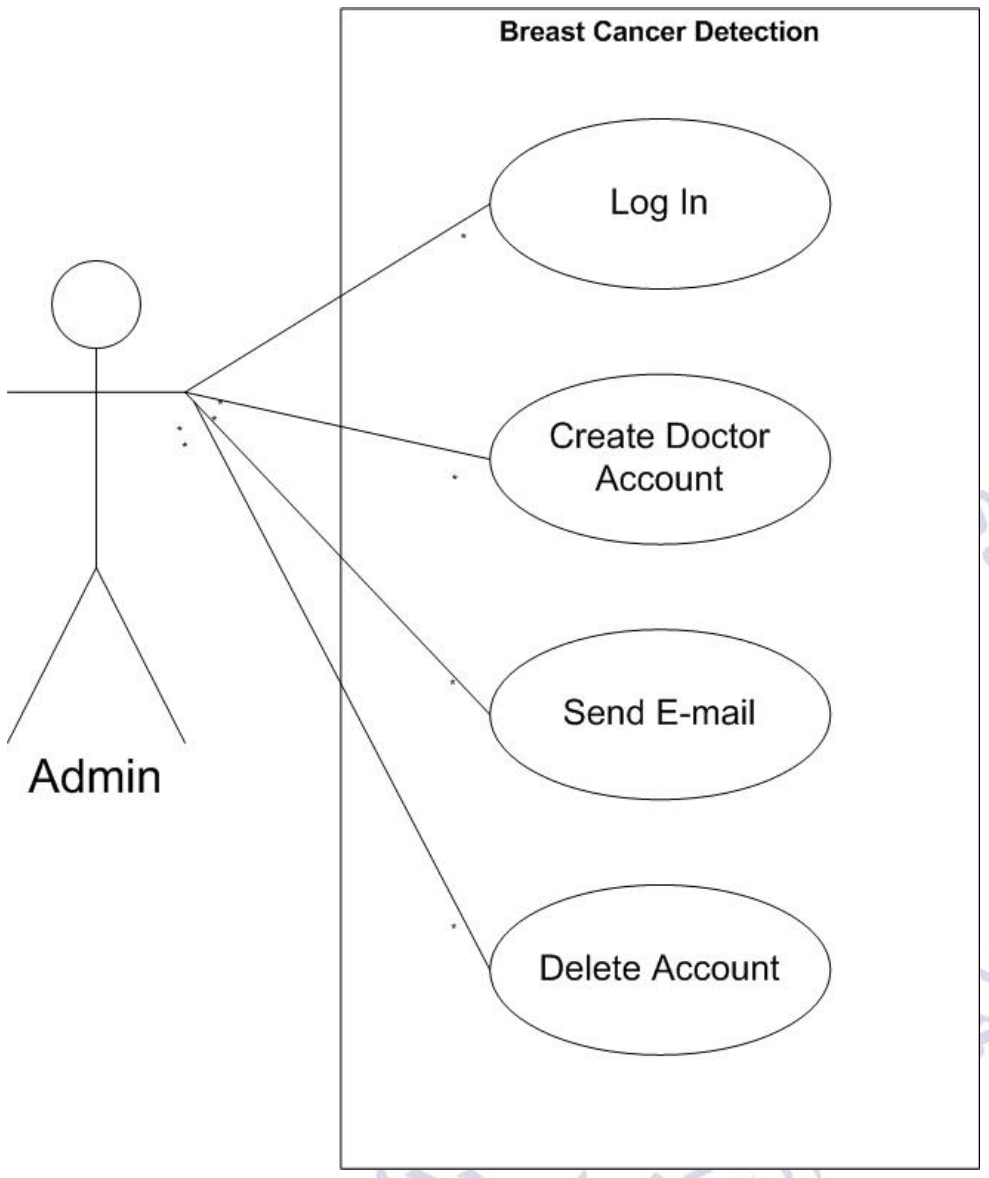


Figure 1.9 : Admin Use case

Sub diagrams of use case:

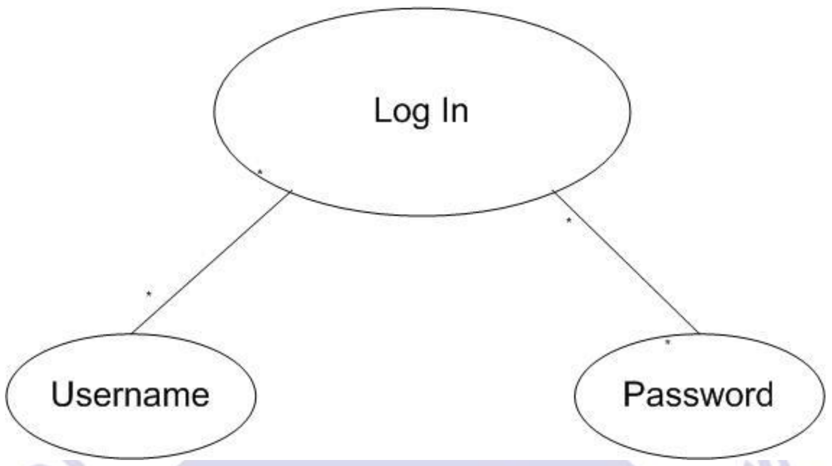
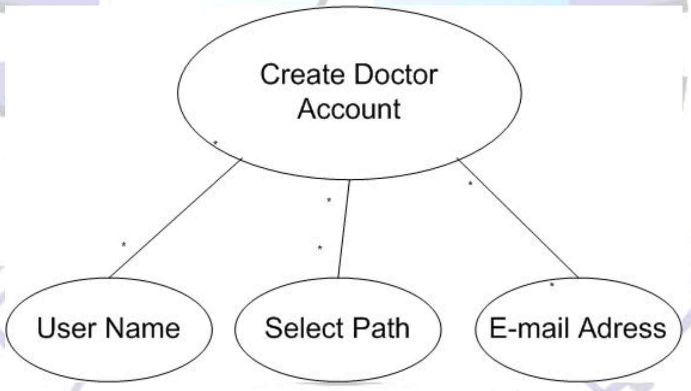


Figure 1.10 : Sub-diagram "Log in"



Figurer 1.11 : Sub-diagram "Create Doctor Account"

System	Checkmate Breast Cancer Application
Use-case	Login
Actor	Admin
Description	A page consists of a user name and password of the admin
Stimulus	Admin enter the user name and password
Response	Open the private registration page of the doctor

System	Checkmate Breast Cancer Application
Use-case	Create Account
Actor	Admin
Description	It consists of 2 part , the first is creat account doctor And the seconde list of accounts doctor.
Stimulus	The admin entre the name, password , e-mail, type of path doctor.
Response	The account is send to the doctor

System	Checkmate Breast Cancer Application
Use-case	Send Email
Actor	Admin
Description	Admin send the account for the doctor
Stimulus	The doctor receiving the account
Response	Login to the program and deal with the program

System	Checkmate Breast Cancer Application
Use-case	Delete Account
Actor	Admin
Description	Admin can delete the doctor account .
Stimulus	Admin select the account who wants to delete it
Response	delete the account

1.3.2 ER Diagram:

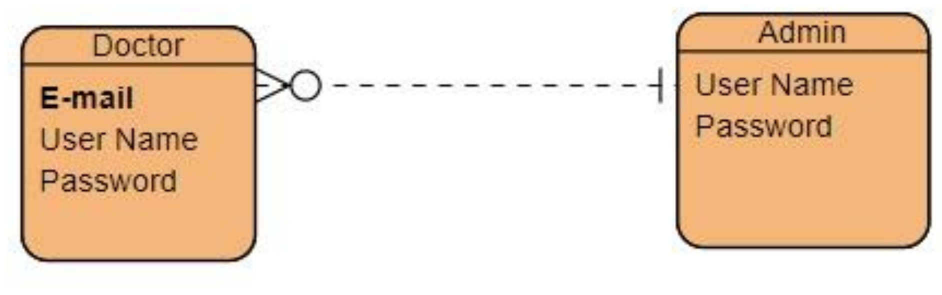


Figure 1.12 ER Diagram

1.3.3UML:

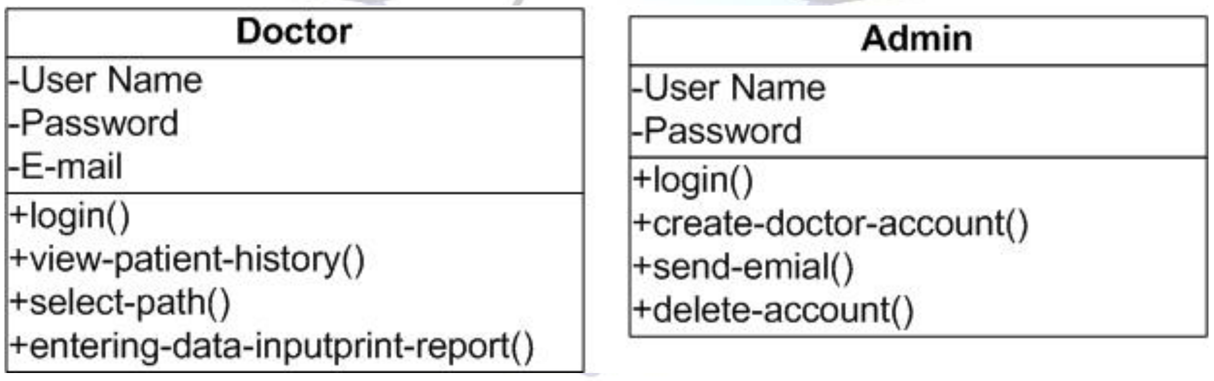


Figure 1.13 UML

1.3.4 Context:

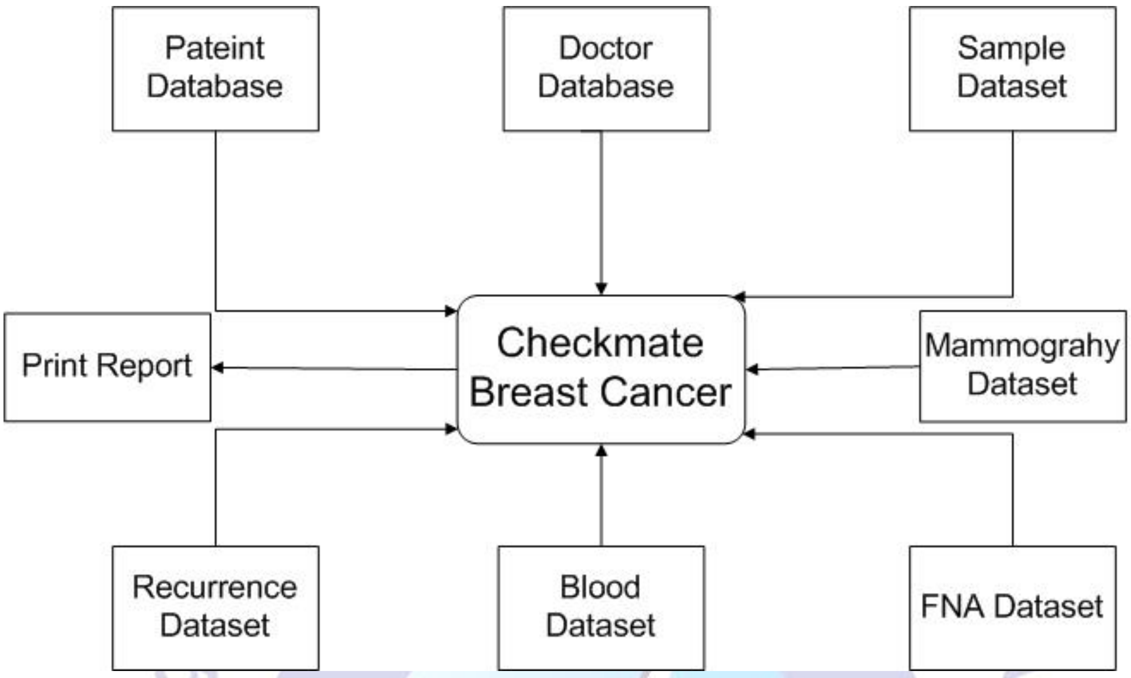
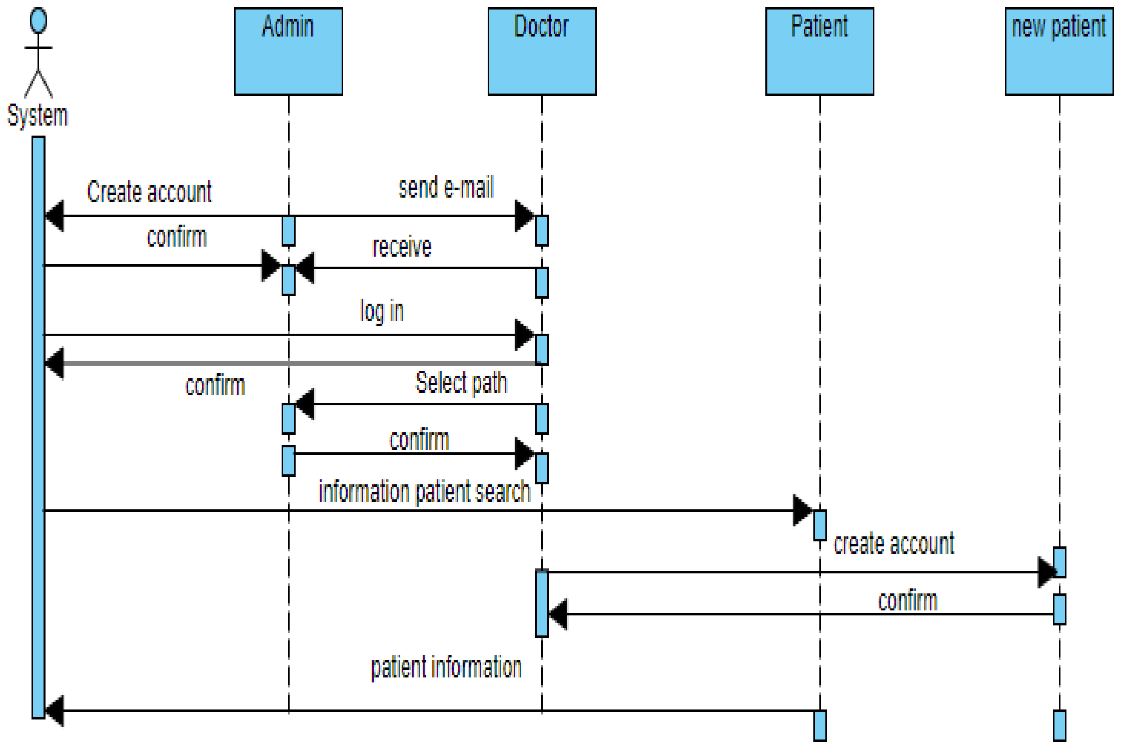


Figure 1.14 Context



Chapter Two

Data base

2.1 Introduction

2.1.1 Data base :

Database is a systematic collection of data. Databases support storage of data. Databases make data management easy. Let's discuss few examples.

An online telephone directory would definitely use database to store data pertaining to people, phone numbers, other contact details, etc.

Your electricity service provider is obviously using a database to manage billing , client related issues, to handle fault data, etc.

Let's also consider the facebook. It needs to store, manipulate and present data related to members, their friends, member activities, messages, advertisements and lot more.

We can provide countless number of examples for usage of databases .

2.1.2 Database Management System (DBMS):

Database Management System (DBMS) is a collection of programs which enables its users to access database, manipulate data, reporting / representation of data .

It also helps to control access to the database.

Database Management Systems are not a new concept and as such had been first implemented in 1960s.

Charles Bachmen's Integrated Data Store (IDS) is said to be the first DBMS in history.

With time database technologies evolved a lot while usage and expected functionalities of databases have been increased immensely.

Types of DBMS:

Navigational DBMS
Relational DBMS
SQL DBMS
Object Oriented DBMS

Figure 2.1 tupes of DBMS

There are 4 major types of DBMS. Let's look into them in detail:

Hierarchical - this type of DBMS employs the "parent-child" relationship of storing data. This type of DBMS is rarely used nowadays. Its structure is like a tree with nodes representing records and branches representing fields. The windows registry used in Windows XP is an example of a hierarchical database. Configuration settings are stored as tree structures with nodes.

Network DBMS - this type of DBMS supports many-to many relations. This usually results in complex database structures. RDM Server is an example of a database management system that implements the network model.

Relational DBMS - this type of DBMS defines database relationships in form of tables, also known as relations. Unlike network DBMS, RDBMS does not support many to many relationships.Relational DBMS usually have pre-defined data types that they can support. This is the most popular DBMS type in the market. Examples of relational database management systems include MySQL, Oracle, and Microsoft SQL Server database.

Object Oriented Relation DBMS - this type supports storage of new data types. The data to be stored is in form of objects. The objects to be stored in the database have attributes (i.e. gender, ager) and methods that define what to do with the data. PostgreSQL is an example of an object oriented relational DBMS.

2.1.3 SQL:

Structured Query language (SQL) pronounced as "S-Q-L" or sometimes as "See-Quel" is actually the standard language for dealing with Relational Databases.

SQL programming can be effectively used to insert, search, update, delete database records.

That doesn't mean SQL cannot do things beyond that.

In fact it can do lot of things including, but not limited to, optimizing and maintenance of databases.

Relational databases like MySQL Database, Oracle, Ms SQL server, Sybase, etc uses SQL! How to use sql syntaxes?

SQL syntaxes used in these databases are almost similar, except the fact that some are using few different syntaxes and even proprietary SQL syntaxes.

SQL Example

```
SELECT * FROM Members WHERE Age > 30
```

2.2 SQLIte.

2.2.1 Definition:

Is a relational database management system contained in a C programming library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program.

SQLite is ACID-compliant and implements most of the SQL standard, generally following PostgreSQL syntax. However, SQLite uses a dynamically and weakly typed SQL syntaxthat does not guarantee the domain integrity. This means that one can, for example, insert a string into an column defined as an integer. SQLite will attempt to convert data between formats where appropriate, the string "123" into an integer in this case, but does not guarantee such conversions, and will store the data as-is if such a conversion is not possible.

SQLite is a popular choice as embedded database software for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems.

2.2.2 Design:

Unlike client–server database management systems, the SQLite engine has no standalone processes with which the application program communicates. Instead, the SQLite library is linked in and thus becomes an integral part of the application program. Linking may be static or dynamic. The application program uses SQLite's functionality through simple function calls, which reduce latency in database access: function calls within a single process are more efficient than inter-process communication.

SQLite stores the entire database (definitions, tables, indices, and the data itself) as a single cross-platform file on a host machine. It implements this simple design by locking the entire database file during writing. SQLite read operations can be multitasked, though writes can only be performed sequentially.

Due to the server-less design, SQLite applications require less configuration than client-server databases. SQLite is called *zero-conf* because it does not require service management (such as startup scripts) or access control based on GRANT and passwords. Access control is handled by means of file system permissions given to the database file itself. Databases in client-server systems use file system permissions which give access to the database files only to the daemon process.

Another implication of the serverless design is that several processes may not be able to write to the database file. In server-based databases, several writers will all connect to the same daemon, which is able to handle its locks internally. SQLite on the other hand has to rely on file-system locks. It has less knowledge of the other processes that are accessing the database at the same time. Therefore, SQLite is not the preferred choice for write-intensive deployments. However, for simple queries with little concurrency, SQLite performance profits from avoiding the overhead of passing its data to another process.

SQLite uses PostgreSQL as a reference platform. "What would PostgreSQL do" is used to make sense of the SQL standard. One major deviation is that, with the exception of primary keys, SQLite does not enforce type checking; the type of a value is dynamic and not strictly constrained by the schema (although the schema will trigger a conversion

.

2.2.3Feature:

SQLite implements most of the SQL-92 standard for SQL but it lacks some features. For example, it partially provides triggers, and it cannot write to views (however it provides INSTEAD OF triggers that provide this functionality). While it provides complex queries, it still has limited ALTER TABLE function, as it cannot modify or delete columns.

SQLite uses an unusual type system for an SQL-compatible DBMS; instead of assigning a type to a column as in most SQL database systems, types are assigned to individual values; in language terms it is *dynamically typed*. Moreover, it is *weakly typed* in some of the same ways that Perl is: one can insert a string into an integer column (although SQLite will try to convert the string to an integer first, if the column's preferred type is integer). This adds flexibility to columns, especially when bound to a dynamically typed scripting language. However, the technique is not portable to other SQL products. A common criticism is that SQLite's type system lacks the data integrity mechanism provided by statically typed columns in other products. The SQLite web site describes a "strict affinity" mode, but this feature has not yet been added. However, it can be implemented with constraints like `CHECK(typeof(x)='integer')`. Tables normally include a hidden *rowid* index column which gives faster access.[17] If a database includes an Integer Primary Key column SQLite will typically optimize it by treating it as an alias for *rowid*, causing the contents to be stored as a strictly typed 64-bit signed integer and changing its behavior to be somewhat like an auto-incrementing column. Future versions of SQLite may include a command to introspect whether a column has behavior like that of *rowid* to differentiate these columns from weakly-typed, non-autoincrementing Integer Primary Keys.

SQLite with full Unicode function is optional.

Several computer processes or threads may access the same database concurrently. Several read accesses can be satisfied in parallel. A write access can only be satisfied if no other accesses are currently being serviced. Otherwise, the write access fails with an error code (or can automatically be retried until a configurable timeout expires). This concurrent access situation would change when dealing with temporary tables. This restriction is relaxed in version 3.7 when write-ahead logging (WAL) is turned on enabling concurrent reads and writes.

SQLite version 3.7.4 first saw the addition of the FTS4 (full text search) module, which features enhancements over the older FTS3 module. FTS4 allows users to perform full text searches on documents similar to how search engines search webpages. Version 3.8.2 added support for creating tables without rowid, which may provide space and performance improvements. Common table expressions support was added to SQLite in version 3.8.3.

In 2015, with the *json1 extension* and new subtype interfaces, SQLite version 3.9 introduced JSON content managing.

2.2.4 Notable users:

MiddlewareEdit

ADO.NET adapter, initially developed by Robert Simpson, is maintained jointly with the SQLite developers since April 2010.

ODBC driver has been developed and is maintained separately by Christian Werner. Werner's ODBC driver is the recommended connection method for accessing SQLite from OpenOffice.org.

COM (ActiveX) wrapper making SQLite accessible on Windows to scripted languages such as JScript and VBScript. This adds SQLite database capabilities to HTML Applications (HTA).

XULRunner uses SQLite

Web browsersEdit

The browsers Google Chrome, Opera, Safari and the Android Browser all allow for storing information in, and retrieving it from, a SQLite database within the browser, using the Web SQL Database technology, although this is rapidly becoming deprecated (namely superseded by IndexedDB).

Mozilla Firefox and Mozilla Thunderbird store a variety of configuration data (bookmarks, cookies, contacts etc.) in internally managed SQLite databases. Until the advent of Firefox version 57 ("Firefox Quantum"), there is a third-party add-on that uses the code supporting this functionality to provide a user interface for managing arbitrary SQLite databases.

Several third-party add-ons can make use of JavaScript APIs to manage SQLite databases.

Web application frameworksEdit

- Bugzilla
- Django's default database management system
- Drupal
- Trac
- Ruby on Rails' default database management system
- web2py
- Laravel

VariousEdit

- Adobe Systems uses SQLite as its file format in Adobe Photoshop Lightroom, a standard database in Adobe AIR, and internally within Adobe Reader.
- Evernote uses SQLite to store its local database repository in Windows.
- Skype
- The Service Management Facility, used for service management within the Solaris and OpenSolaris operating systems
- Flame (malware)
- BMW IDrive Sat Nav system

Operating systemsEdit

- Blackberry's BlackBerry 10 OS
- Symbian OS
- Nokia's Maemo
- Google's Android
- Linux Foundation's MeeGo
- LG's webOS
- NetBSD
- FreeBSD where starting with 10-RELEASE version in January 2014, it is used by the core package management system.
- illumos
- Oracle Solaris 10 where the Service Management Facility database is serialized for booting.
- Apple adopted it as an option in macOS's Core Data API from the original implementation in Mac OS X 10.4 onwards, and also for administration of videos and songs, and in iOS for storage of text messages on the iPhone.

Chapter Three

PyQt

3.1 PyQt.

3.1.1 Python (programming language):

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.[26] In July 2018, Van Rossum stepped down as the leader in the language community.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open sourcesoftware and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

3.1.2 Python Libraries:

Python's large standard library, commonly cited as one of its greatest strengths, provides tools suited to many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals, manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications (for example, the Web Server Gateway Interface (WSGI) implementation `wsgiref` follows PEP 333), but most modules are not. They are specified by their code, internal documentation, and test suites (if supplied). However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of March 2018, the Python Package Index (PyPI), the official repository for third-party Python software, contains over 130,000 packages with a wide range of functionality, including:

- Graphical user interfaces
- Web frameworks
- Multimedia
- Databases
- Networking
- Test frameworks
- Automation
- Web scraping
- Documentation
- System administration
- Scientific computing
- Text processing
- Image processing

3.1.3 About PyQt:

PyQt is one of the most popular Python bindings for the Qt cross-platform C++ framework. PyQt developed by Riverbank Computing Limited. Qt itself is developed as part of the Qt Project. PyQt provides bindings for Qt 4 and Qt 5. PyQt is distributed under a choice of licences: GPL version 3 or a commercial license.

is available in two editions: PyQt4 which will build against Qt 4.x and 5.x and PyQt5 which will only build against 5.x. Both editions can be built for Python 2 and 3. PyQt contains over 620 classes that cover graphical user interfaces, XML handling, network communication, SQL databases, Web browsing and other technologies available in Qt.

The latest iteration of PyQt is v5.11.3. It fully supports Qt 5.11.2.

PyQt4 runs on Windows, Linux, Mac OS X and various UNIX platforms. PyQt5 also runs on Android and iOS.

3.1.4 Why PyQt:

- PyQt brings together the Qt C++ cross-platform application framework and the cross-platform interpreted language Python.
- Qt is more than a GUI toolkit. It includes abstractions of network sockets, threads, Unicode, regular expressions, SQL databases, SVG, OpenGL, XML, a fully functional web browser, a help system, a multimedia framework, as well as a rich collection of GUI widgets.
- Qt classes employ a signal/slot mechanism for communicating between objects that is type safe but loosely coupled making it easy to create re-usable software components.
- Qt also includes Qt Designer, a graphical user interface designer. PyQt is able to generate Python code from Qt Designer. It is also possible to add new GUI controls written in Python to Qt Designer.
- Python is a simple but powerful object-orientated language. Its simplicity makes it easy to learn, but its power means that large and complex applications can be created. Its interpreted nature means that Python programmers are very productive because there is no edit/compile/link/run development cycle.
- Much of Python's power comes from its comprehensive set of extension modules providing a wide variety of functions including HTTP servers, XML parsers, database access, data compression tools and, of course, graphical user interfaces. Extension modules are usually implemented in either Python, C or C++. Using tools such as SIP it is relatively straight forward to create an extension module that encapsulates an existing C or C++ library. Used in this way, Python can then become the glue to create new applications from established libraries.
- PyQt combines all the advantages of Qt and Python. A programmer has all the power of Qt, but is able to exploit it with the simplicity of Python.

3.1.5 What is PyQt used for:

PyQt is a binding for QT framework. Qt is written in C++, So with PyQt you can use all QT classes and methods with Python languages without knowing C++.

Mainly, you can develop desktop applications with PyQT.

PyQt brings together the Qt C++ cross-platform application framework and the cross-platform interpreted language Python.

3.1.6 PyQt Desktop Apps with Python:

Make Desktop applications with Python Pyqt:

PyQt can be challenging to learn, in this course we'll make it easy for you. The course starts with the basics like buttons, labels, signals and slots. Then it dives deeper and teaches you how to use designer,

Make several projects including:

- a text editor,
- web browser
- bmi calculator.
- You will be able to *make your own desktop apps with Python PyQt!*

3.2 PyQt5.

3.2.1 Introduction to pyqt5:

This is the reference guide for PyQt5 5.11.1. PyQt5 is a set of Python bindings for v5 of the Qt application framework from The Qt Company.

Qt is a set of C++ libraries and development tools that includes platform independent abstractions for graphical user interfaces, networking, threads, regular expressions, SQL databases, SVG, OpenGL, XML, user and application settings, positioning and location services, short range communications (NFC and Bluetooth), web browsing, 3D animation, charts, 3D data visualisation and interfacing with app stores. PyQt5 implements over 1000 of these classes as a set of Python modules.

PyQt5 supports the Windows, Linux, UNIX, Android, macOS and iOS platforms.

PyQt does not include a copy of Qt. You must obtain a correctly licensed copy of Qt yourself. However, binary wheels of the GPL version of PyQt5 are provided and these include a copy of the appropriate parts of the LGPL version of Qt..

PyQt5 is built using the SIP bindings generator. SIP must be installed in order to build and use PyQt5.

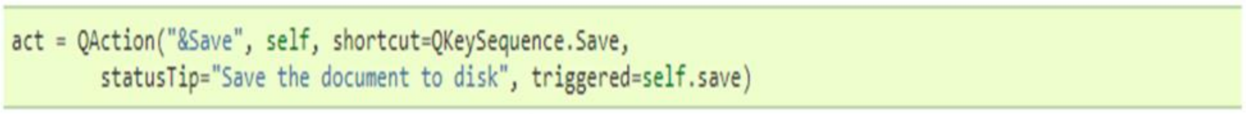
Earlier versions of Qt are supported by PyQt4.

3.2.2 Differences Between PyQt4 and PyQt5:

- PyQt5 is not compatibile with PyQt4 (although experience shows that the effort in porting applications from PyQt4 to PyQt5 is not great). This section describes the main differences between the two.
- Unlike PyQt4, PyQt5 supports the definition of properties, signals and slots in classes not sub-classed from QObject (i.e. in mixins).
- PyQt4's QtGui module has been split into PyQt5's QtGui, QtPrintSupport and QtWidgets modules.
- In PyQt4, QSet was implemented as a list in Python v2 and a set in Python v3. In PyQt5 QSet is always implemented as a set.

3.2.3 Support for Qt Properties:

- PyQt5 does not support the setting and getting of Qt properties as if they were normal instance attributes. This is because the name of a property often conflicts with the name of the property's getter method.
- However, PyQt5 does support the initial setting of properties using keyword arguments passed when an instance is created. For example:

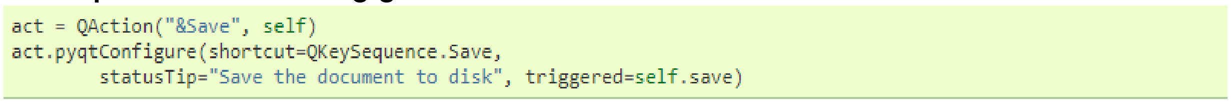


```
act = QAction("&Save", self, shortcut=QKeySequence.Save,
        statusTip="Save the document to disk", triggered=self.save)
```

Figure3.1 support for qt

The example also demonstrates the use of a keyword argument to connect a signal to a slot.

- PyQt5 also supports setting the values of properties (and connecting a signal to a slot) using the pyqtConfigure() method. For example, the following gives the same results as above:



```
act = QAction("&Save", self)
act.pyqtConfigure(shortcut=QKeySequence.Save,
        statusTip="Save the document to disk", triggered=self.save)
```

Figure 3.2 act for qt

3.2.4Integrating Python and QML:

Qt includes QML as a means of declaratively describing a user interface and using JavaScript as a scripting language within it. It is possible to write complete standalone QML applications, or to combine them with C++. PyQt5 allows QML to be integrated with Python in exactly the same way. In particular:

- Python types that are sub-classed from Object can be registered with QML.
- Instances of registered Python types can be created and made available to QML scripts.
- Instances of registered Python types can be created by QML scripts.
- Singleton instances of registered Python types can be created automatically by a QML engine and made available to QML scripts.
- QML scripts interact with Python objects through their properties, signals and slots.
- Python properties, signals and slots can be given revision numbers that only those implemented by a specific version are made available to QML.

3.2.5Registering Python Types:

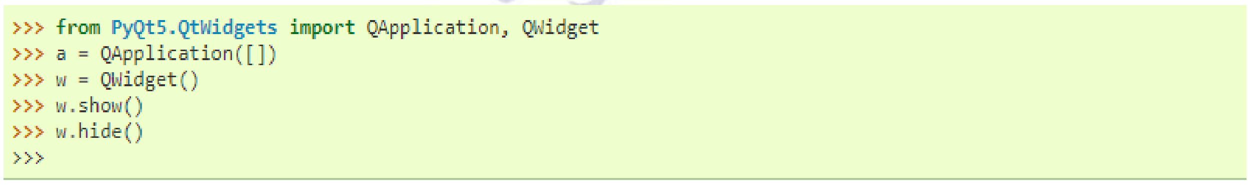
- Registering Python types with QML is done in the same way is it is done with C++ classes, i.e. using the qmlRegisterType(),qmlRegisterSingletonType(), qmlRegisterUncreatableType() and qmlRegisterRevision() functions.
- In C++ these are template based functions that take the C++ class, and sometimes a revision, as template arguments. In the Python implementation these are simply passed as the first arguments to the respective functions.

3.2.6 Using Qt Designer:

- Qt Designer is the Qt tool for designing and building graphical user interfaces. It allows you to design widgets, dialogs or complete main windows using on-screen forms and a simple drag-and-drop interface. It has the ability to preview your designs to ensure they work as you intended, and to allow you to prototype them with your users, before you have to write any code.
- Qt Designer uses XML .ui files to store designs and does not generate any code itself. Qt includes the uic utility that generates the C++ code that creates the user interface. Qt also includes the QUiLoader class that allows an application to load a .ui file and to create the corresponding user interface dynamically.
- PyQt5 does not wrap the QUiLoader class but instead includes the uic Python module. Like QUiLoader this module can load.ui files to create a user interface dynamically. Like the uic utility it can also generate the Python code that will create the user interface. PyQt5's pyuic5 utility is a command line interface to the uic module. Both are described in detail in the following sections.

3.2.7 Using PyQt5 from the Python Shell:

- PyQt5 installs an input hook (using PyOS_InputHook) that processes events when an interactive interpreter is waiting for user input. This means that you can, for example, create widgets from the Python shell prompt, interact with them, and still being able to enter other Python commands.
- For example, if you enter the following in the Python shell:



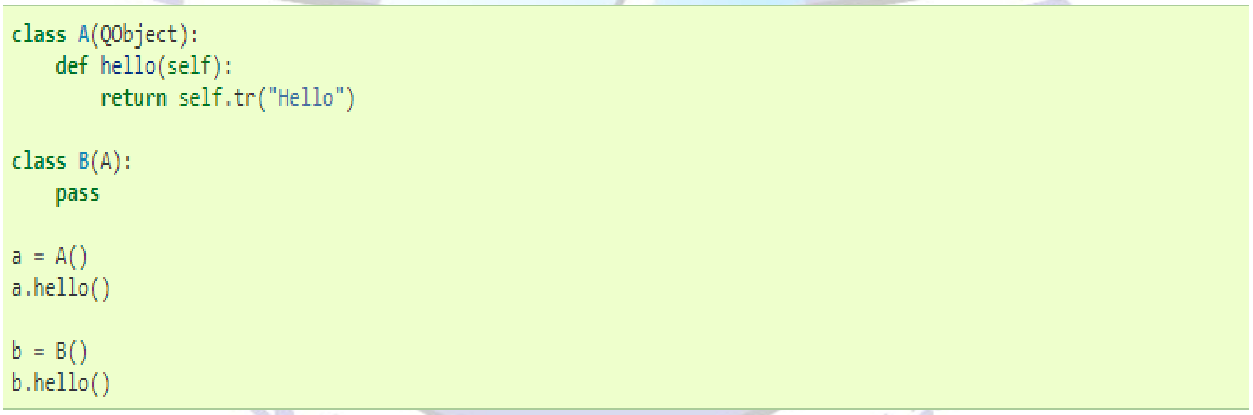
```
>>> from PyQt5.QtWidgets import QApplication, QWidget
>>> a = QApplication([])
>>> w = QWidget()
>>> w.show()
>>> w.hide()
>>>
```

Figure 3.3 pyqt5 python

- The widget would be displayed when w.show() was entered and hidden as soon as w.hide() was entered.
- The installation of an input hook can cause problems for certain applications (particularly those that implement a similar feature using different means). The QtCore module contains .

3.2.8 Differences Between PyQt5 and Qt:

- Qt implements internationalisation support through the QTranslator class, and the translate() and tr() methods. Usuallytr() is used to obtain the correct translation of a message. The translation process uses a message context to allow the same message to be translated differently. In Qt tr() is actually generated by moc and uses the hardcoded class name as the context. On the other hand, translate() allows the context to be specified explicitly.
- Unfortunately, because of the way Qt implements tr() it is not possible for PyQt5 to exactly reproduce its behaviour. The PyQt5 implementation of tr() uses the class name of the instance as the context. The key difference, and the source of potential problems, is that the context is determined dynamically in PyQt5, but is hardcoded in Qt. In other words, the context of a translation may change depending on an instance's class hierarchy. For example:

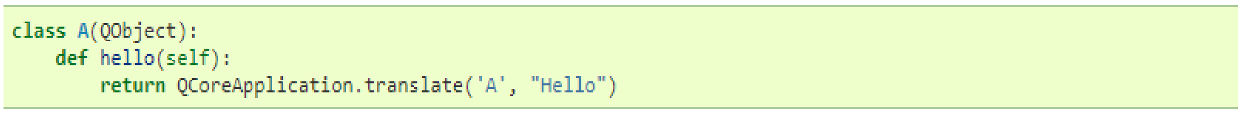


```
class A(QObject):
    def hello(self):
        return self.tr("Hello")

class B(A):
    pass

a = A()
a.hello()

b = B()
b.hello()
```

Figure 3.4 pyqt5

In the above the message is translated by a.hello() using a context of A, and by b.hello() using a context of B. In the equivalent C++ version the context would be A in both cases.

The PyQt5 behaviour is unsatisfactory and may be changed in the future. It is recommended that translate() be used in preference to tr(). This is guaranteed to work with current and future versions of PyQt5 and makes it much easier to share message files between Python and C++ code. Below is the alternative implementation of A that uses translate():



```
class A(QObject):
    def hello(self):
        return QCoreApplication.translate('A', "Hello")
```

Figure 3.5 PyQt4

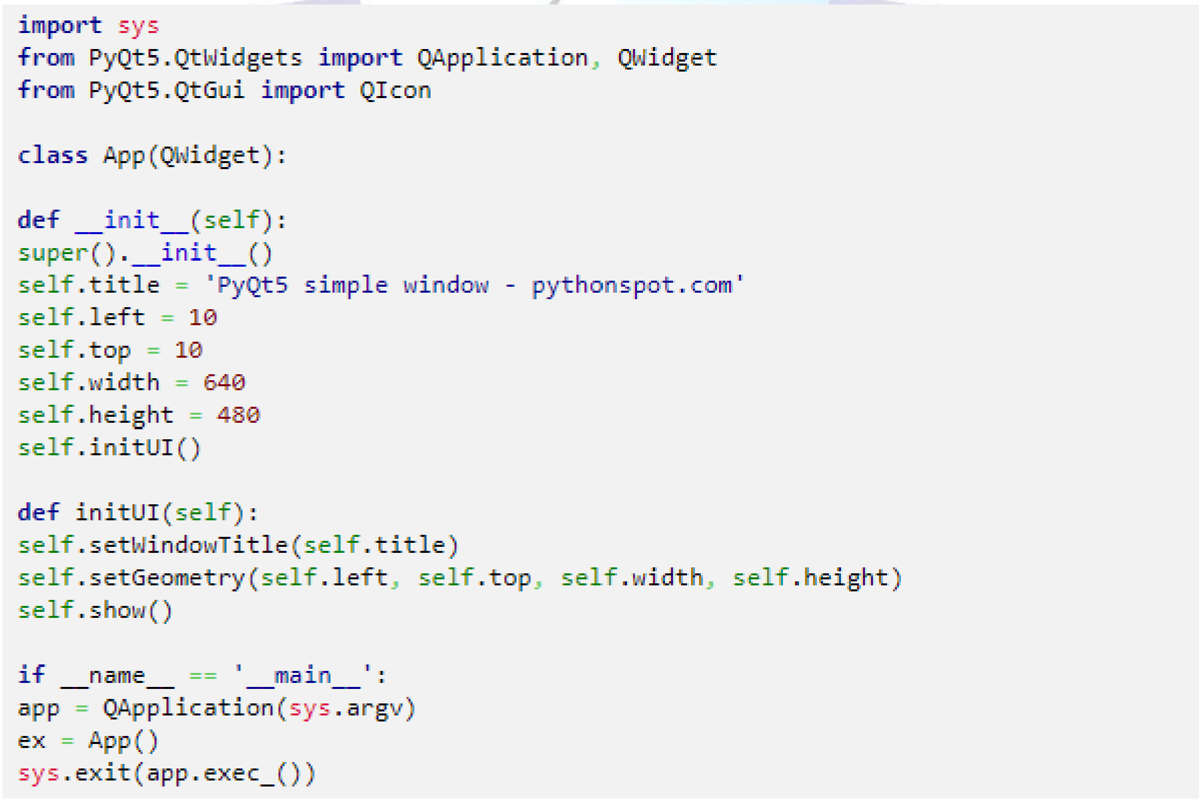
3.2.9 The PyQt5 Extension API:

Python API:

- The Python part of the API is accessible via the QtCore module and is typically used by an extension module's equivalent of PyQt5's configure.py.
- The API consists of PyQt5.QtCore.PYQT_CONFIGURATION which is a dict that describes how PyQt5 was configured. At the moment it contains a single value called sip_flags which is a string containing the -t and -x flags that were passed to the sip executable by configure.py. Other extension modules must use the same flags in their configuration.
- This information is also provided by SIP v4's sipconfig module. However this module will not be implemented by SIP v5.

3.2.10 PyQt5 window:

- You can create a PyQT5 window using the code below:

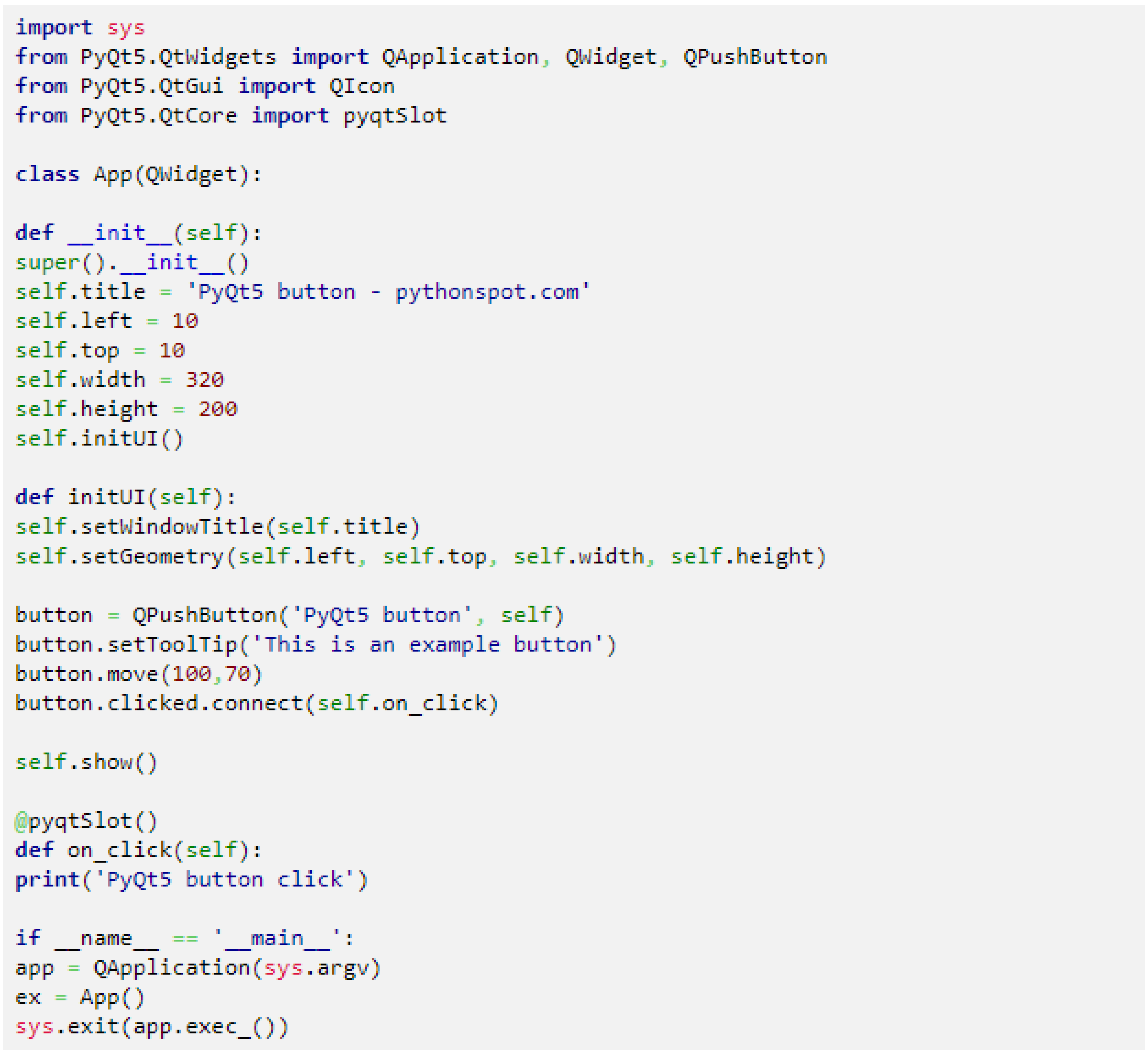


```
import sys
from PyQt5.QtWidgets import QApplication, QWidget
from PyQt5.QtGui import QIcon

class App(QWidget):

def __init__(self):
super().__init__()
self.title = 'PyQt5 simple window - pythonspot.com'
self.left = 10
self.top = 10
self.width = 640
self.height = 480
self.initUI()

def initUI(self):
self.setWindowTitle(self.title)
self.setGeometry(self.left, self.top, self.width, self.height)
self.show()

if __name__ == '__main__':
app = QApplication(sys.argv)
ex = App()
sys.exit(app.exec_())
```

Figure 3.6 pyqt5 window

3.2.11 PyQt5 buttons:

- PyQt5 supports buttons using the QPushButton class. This class is inside the PyQt5.QtWidgets group. The button can be created by calling the constructor QPushButton with the text to display as parameter.

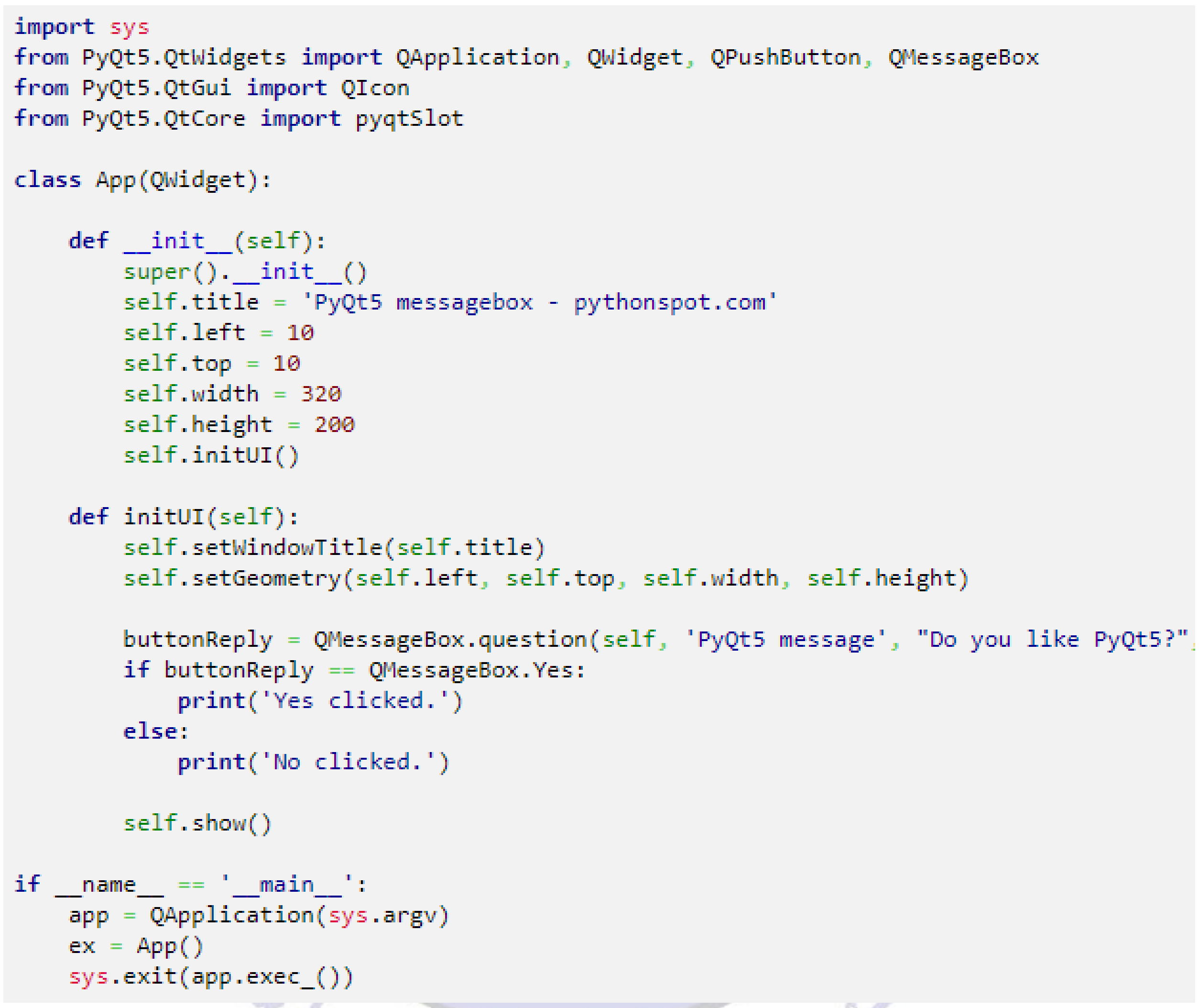


```
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton
from PyQt5.QtGui import QIcon
from PyQt5.QtCore import pyqtSlot

class App(QWidget):

def __init__(self):
super().__init__()
self.title = 'PyQt5 button - pythonspot.com'
self.left = 10
self.top = 10
self.width = 320
self.height = 200
self.initUI()

def initUI(self):
self.setWindowTitle(self.title)
self.setGeometry(self.left, self.top, self.width, self.height)

button = QPushButton('PyQt5 button', self)
button.setToolTip('This is an example button')
button.move(100,70)
button.clicked.connect(self.on_click)

self.show()

@pyqtSlot()
def on_click(self):
print('PyQt5 button click')

if __name__ == '__main__':
app = QApplication(sys.argv)
ex = App()
sys.exit(app.exec_())
```

Figure 3.7 pyqt5 Button

3.2.12 PyQt5 messagebox:

To show a messagebox we need to import **QMessageBox**



```python
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QMessageBox
from PyQt5.QtGui import QIcon
from PyQt5.QtCore import pyqtSlot

class App(QWidget):

    def __init__(self):
        super().__init__()
        self.title = 'PyQt5 messagebox - pythonspot.com'
        self.left = 10
        self.top = 10
        self.width = 320
        self.height = 200
        self.initUI()

    def initUI(self):
        self.setWindowTitle(self.title)
        self.setGeometry(self.left, self.top, self.width, self.height)

        buttonReply = QMessageBox.question(self, 'PyQt5 message', "Do you like PyQt5?",
        if buttonReply == QMessageBox.Yes:
            print('Yes clicked.')
        else:
            print('No clicked.')

        self.show()

if __name__ == '__main__':
    app = QApplication(sys.argv)
    ex = App()
    sys.exit(app.exec_())
```

Figure 3.8 pyqt5 messagebox

Chapter Four

Machine Learning

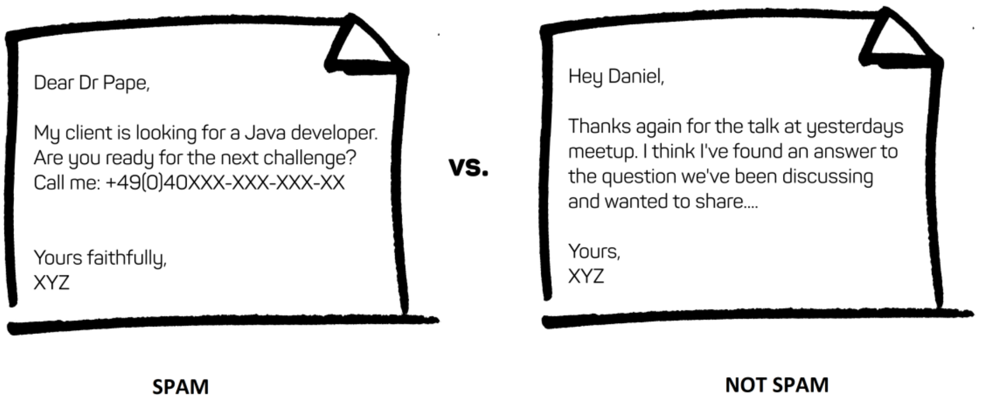
4.1 Introduction

"We Don't have Better Algorithms we just have more data"
-Peter Norvig (Director of research ,Google)

Machine learning is about extracting knowledge from data. It is a research field at the intersection of statistics, artificial intelligence, and computer science and is also known as predictive analytics or statistical learning. The application of machine learning methods has in recent years become ubiquitous in everyday life. From auto- matic recommendations of which movies to watch, to what food to order or which products to buy, to personalized online radio and recognizing your friends in your photos, many modern websites and devices have machine learning algorithms at their core. When you look at a complex website like Facebook, Amazon, or Netflix, it is very likely that every part of the site contains multiple machine learning models. Outside of commercial applications, machine learning has had a tremendous influence on the way data-driven research is done today. The tools introduced in this book have been applied to diverse scientific problems such as understanding stars, finding distant planets, discovering new particles, analyzing DNA sequences, and providing personalized cancer treatments. Your application doesn't need to be as large-scale or world-changing as these examples in order to benefit from machine learning

Why Machine Learning? In the early days of "intelligent" applications, many systems used handcoded rules of "if " and "else" decisions to process data or adjust to user input. Think of a spam filter whose job is to move the appropriate incoming email messages to a spam folder. You could make up a blacklist of words that would result in an email being marked as spam.

This would be an example of using an expert-designed rule system to design an "intelligent" application. Manually crafting decision rules is feasible for some applications, particularly those in which humans have a good understanding of the process to model.



Dear Dr Pape,

My client is looking for a Java developer. Are you ready for the next challenge? Call me: +49(0)40XXX-XXX-XXX-XX

Yours faithfully,
XYZ

Hey Daniel,

Thanks again for the talk at yesterdays meetup. I think I've found an answer to the question we've been discussing and wanted to share....

Yours,
XYZ

vs.
SPAM
NOT SPAM

Figure 4.1 spam &not spam

However, using handcoded rules to make decisions has two major disadvantages:

- The logic required to make a decision is specific to a single domain and task. Changing the task even slightly might require a rewrite of the whole system.
- Designing rules requires a deep understanding of how a decision should be made by a human expert. One example of where this hand coded
  approach will fail is in detecting faces in images.

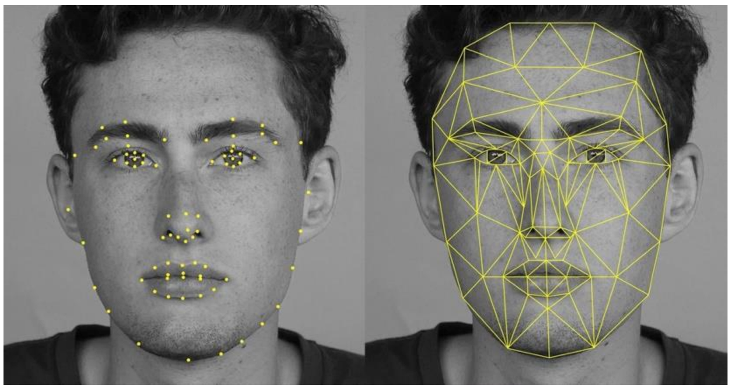


Figure 4.2 detect faces images

Today, every smartphone can detect a face in an image. However, face detection was an unsolved problem until as recently as 2001.

The main problem is that the way in which pixels (which make up an image in a computer) are "perceived" by the computer is very different from how humans perceive a face.

This difference in representation makes it basically impossible for a human to come up with a good set of rules to describe what constitutes a face in a digital image.

"A baby learns to crawl, walk and then run. We are in the crawling stage when it comes to applying machine learning."

-Dave Waters

University of Oxford Associate Professor of Metamorphic

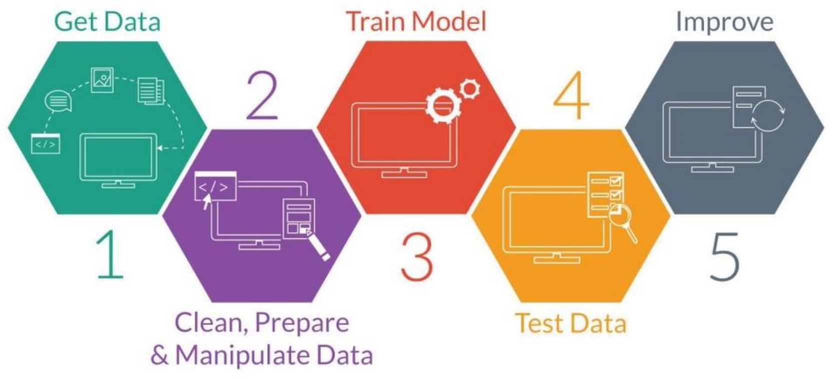


Figure 4.3 data handling

Many people imagine that data science is mostly machine learning and that data scientists mostly build and train and tweak machine-learning models all day long. (Then again, many of those people don't actually know what machine learning is.) In fact, data science is mostly turning business problems into data problems and collecting data and understanding data and cleaning data and formatting data, after which machine learning is almost an afterthought.

Even so, it’s an interesting and essential afterthought that you pretty much have to know about in order to do data science Using machine learning, however, simply presenting a program with a large collection of images of faces is enough for an algorithm to determine what characteristics are needed to identify a face. Problems Machine Learning Can Solve The most successful kinds of machine learning algorithms are those that automate decision-making processes by generalizing from known examples. In this setting, which is known as supervised learning, the user provides the algorithm with pairs of inputs and desired outputs, and the algorithm finds a way to produce the desired out- put given an input. In particular, the algorithm is able to create an output for an input it has never seen before without any help from a human. Going back to our example of spam classification, using machine learning, the user provides the algorithm with a large number of emails (which are the input), together with information about whether any of these emails are spam (which is the desired output). Given a new email, the algorithm will then produce a prediction as to whether the new email is spam. Machine learning algorithms that learn from input/output pairs are called supervised learning algorithms because a “teacher” provides supervision to the algorithms in th create a dataset that includes the desired outcome, machine learning will likely be able to solve your problem. Examples of supervised machine learning tasks include: Identifying the zip code from handwritten digits on an envelope Here the input is a scan of the handwriting, and the desired output is the actual digits in the zip code. To create a dataset for building a machine learning model, you need to collect many envelopes. Then you can read the zip codes yourself and store the digits as your desired outcomes. Determining whether a tumor is benign based on a medical image Here the input is the image, and the output is whether the tumor is benign. To create a dataset for building a model, you need a database of medical images.

You also need an expert opinion, so a doctor needs to look at all of the images and decide which tumors are benign and which are not. It might even be necessary to do additional diagnosis beyond the content of the image to determine whether the tumor in the image is cancerous or not. Detecting fraudulent activity in credit card transactions Here the input is a record of the credit card transaction, and the output is whether it is likely to be fraudulent or not. Assuming that you are the entity distributing the credit cards, collecting a dataset means storing all transactions and recording if a user reports any transaction as fraudulent.

An interesting thing to note about these examples is that although the inputs and out- puts look fairly straightforward, the data collection process for these three tasks is vastly different. While reading envelopes is laborious, it is easy and cheap. Obtaining medical imaging and diagnoses, on the other hand, requires not only expensive machinery but also rare and expensive expert knowledge, not to mention the ethical concerns and privacy issues. In the example of detecting credit card fraud, data col- lection is much simpler. Your customers will provide you with the desired output, as they will report fraud. All you have to do to obtain the input/output pairs of fraudulent and no fraudulent activity is wait. Unsupervised algorithms are the other type of algorithm that we will cover in this book. In unsupervised learning, only the input data is known, and no known output data is given to the algorithm. While there are many successful applications of these methods, they are usually harder to understand and evaluate. Examples of unsupervised learning include: Identifying topics in a set of blog posts If you have a large collection of text data, you might want to summarize it and find prevalent themes in it. You might not know beforehand what these topics are, or how many topics there might be. Therefore, there are no known outputs. Why Machine Learning? | 3 Segmenting customers into groups with similar preferences given a set of customer records,

You might want to identify which customers are similar, and whether there are groups of customers with similar preferences. For a shopping site, these might be “parents,” “bookworms,” or “gamers.” Because you don’t know in advance what these groups might be, or even how many there are, you have no known outputs. Detecting abnormal access patterns to a website to identify abuse or bugs, it is often helpful to find access patterns that are different from the norm. Each abnormal pattern might be very different, and you might not have any recorded instances of abnormal behavior. Because in this example you only observe traffic, and you don’t know what constitutes normal and abnormal behavior, this is an unsupervised problem. For both supervised and unsupervised learning tasks, it is important to have a representation of your input data that a computer can understand. Often it is helpful to think of your data as a table. Each data point that you want to reason about (each email, each customer, each transaction) is a row, and each property that describes that data point (say, the age of a customer or the amount or location of a transaction) is a column. You might describe users by their age, their gender, when they created an account, and how often they have bought from your online shop. You might describe the image of a tumor by the grayscale values of each pixel, or maybe by using the size, shape, and color of the tumor. Each entity or row here is known as a sample (or data point) in machine learning, while the columns—the properties that describe these entities—are called features. Later in this book we will go into more detail on the topic of building a good representation of your data, which is called feature extraction or feature engineering. You should keep in mind, however, that no machine learning algorithm will be able to make a prediction on data for which it has no information. For example, if the only feature that you have for a patient is their last name, no algorithm will be able to predict their gender. this information is simply not contained in your data. If you add another feature that contains the patient's first name, you will have much better luck, as it is often possible to tell the gender by a person's first name.

Machine learning is an integral part of many commercial applications and research projects today, in areas ranging from medical diagnosis and treatment to finding your friends on.

4.2 Gathering The Data (Asking the right Question)

"I can't make bricks without clay".

-Arthur Conan Doyle

Quite possibly the most important part in the machine learning process is under- standing the data you are working with and how it relates to the task you want to solve.

"In God we trust, all others bring data."

-William Edwards Deming was an statistician professor

> We live in a world that's drowning in data. Websites track every user's every click. Your smartphone is building up a record of your location and speed every second of every day. "Quantified selfers" wear pedometers-on-steroids that are ever recording their heart rates, movement habits, diet, and sleep patterns. Smart cars collect driving habits, smart homes collect living habits, and smart marketers collect purchasing habits. The Internet itself represents a huge graph of knowledge that contains (among other things) an enormous cross-referenced encyclopedia; domain-specific databases about movies, music, sports results, pinball machines, memes, and cocktails; and too many government statistics (some of them nearly true!) from too many governments to wrap your head around. Buried in these data are answers to countless questions that no one's ever thought to ask. In this book, we'll learn how to find them.

It will not be effective to randomly choose an algorithm and throw your data at it.

It is necessary to understand what is going on in your dataset before you begin building a model.

Each algorithm is different in terms of what kind of data and what problem setting it works best for.

While you are building a machine learning solution, you should answer, or at least keep in mind, the following questions:

- What question(s) am I trying to answer? Do I think the data collected can answer that question?
- What is the best way to phrase my question(s) as a machine learning problem?
- Have I collected enough data to represent the problem I want to solve?
- What features of the data did I extract, and will these enable the right predictions?
- How will I measure success in my application?
- How will the machine learning solution interact with other parts of my research or business product? In a larger context, the algorithms and methods in machine learning are only one part of a greater process to solve a particular problem, and it is good to keep the big picture in mind at all times. Many people spend a lot of time building complex machine learning solutions, only to find out they don't solve the right problem.

When going deep into the technical aspects of machine learning , it is easy to lose sight of the ultimate goals. While we will not discuss the ques- tions listed here in detail,

we still encourage you to keep in mind all the assumptions that you might be making, explicitly or implicitly, when you start building machine learning models.

4.3 Preparing the Data(Data Cleaning)

"Big Data is not about the Big data"

-Gray king, Harvard University

making point that the data is plentiful ,but the real value is in analytics.

There is a Rule say that

"data will never be in the format the I need"

Data scientists spend a large amount of their time cleaning datasets and getting them down to a form with which they can work. In fact, a lot of data scientists argue that the initial steps of obtaining and cleaning data constitute 80% of the job.

Therefore, if you are just stepping into this field or planning to step into this field, it is important to be able to deal with messy data, whether that means missing values, inconsistent formatting, malformed records, or nonsensical outliers.

Cleaning and preparing data is a critical first step in any machine learning project. In this blog post, Dataquest student Daniel Oseitakes us through examining a dataset, selecting columns for features, exploring the data visually and then encoding the features for machine learning.
After first reading about Machine Learning on Quora in 2015, Daniel became excited at the prospect of an area that could combine his love of Mathematics and Programming. After reading this article on how to learn data science, Daniel started following the steps, eventually joining Dataquest to learn Data Science with us in in April 2016.
We'd like to thank Daniel for his hard work, and generously letting us publish this post. This walkthrough uses Python 3.5 and Jupyter notebook.

4.3.1 Understanding the Data:

Before you start working with data for a machine learning project, it is vital to understand what the data is, and what we want to achieve. Without it, we have no basis from which to make our decisions about what data is relevant as we clean and prepare our data.

4.3.2 Loading The Data Into Pandas:

We've downloaded our dataset and named it Breastcancer.csv, but now we need to load it into a pandas DataFrame to explore it.
To ensure that code run fast for us, we need to reduce the size of Breastcancer.csv by doing the following:

- Remove the first line: It contains extraneous text instead of the column titles. This text prevents the dataset from being parsed properly by the pandas library.
- Remove the 'desc' column: it contains a long text explanation for the loan.
- Remove the 'url' column: it contains a link to each on Lending Club which can only be accessed with an investor account.
- Removing all columns with more than 50% missing values: This allows us to move faster since don't need to spend time trying to fill these values.

We'll also name the filtered dataset CleanningBreastcancer.csv and later at the end of this section save it as CleanningBreastcancer.csv to keep it separate from the raw data. This is good practice and makes sure we have our original data in case we need to go back and retrieve any of the original data we're removing.

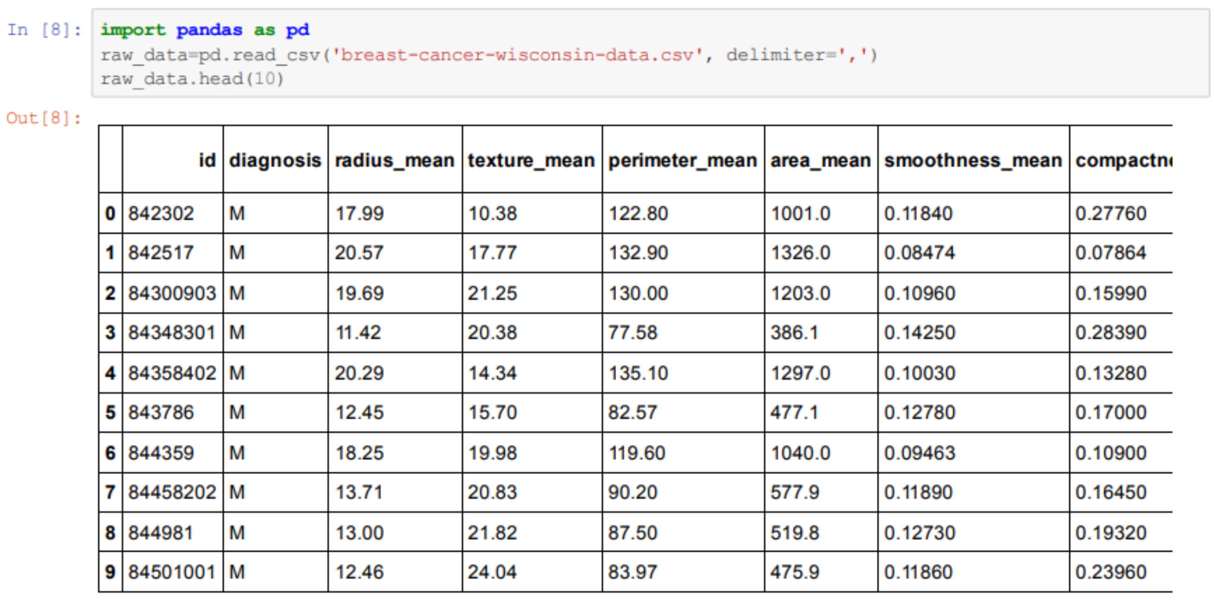
4.4 Preparing the Features for Machine Learning:

In this section, we'll prepare the `CleanningBreastcancer.csv` data for machine learning. We'll focus on handling missing values, converting categorical columns to numeric columns and removing any other extraneous columns.
We need to handle missing values and categorical features before feeding the data into a machine learning algorithm, because the mathematics underlying most machine learning models assumes that the data is numerical and contains no missing values. To reinforce this requirement, scikit-learn will return an error if you try to train a model using data that contain missing values or non-numeric values when working with models like linear regression and logistic regression.

Here's an outline of what we'll be doing in this stage:

- Handle Missing Values
- Investigate Categorical Columns
- Convert Categorical Columns To Numeric Features
- Map Ordinal Values To Integers
- Encode Nominal Values As Dummy Variables

First though, let's load in the data from last section's final output:



```
In [8]: import pandas as pd
        raw_data=pd.read_csv('breast-cancer-wisconsin-data.csv', delimiter=',')
        raw_data.head(10)

Out[8]:
```

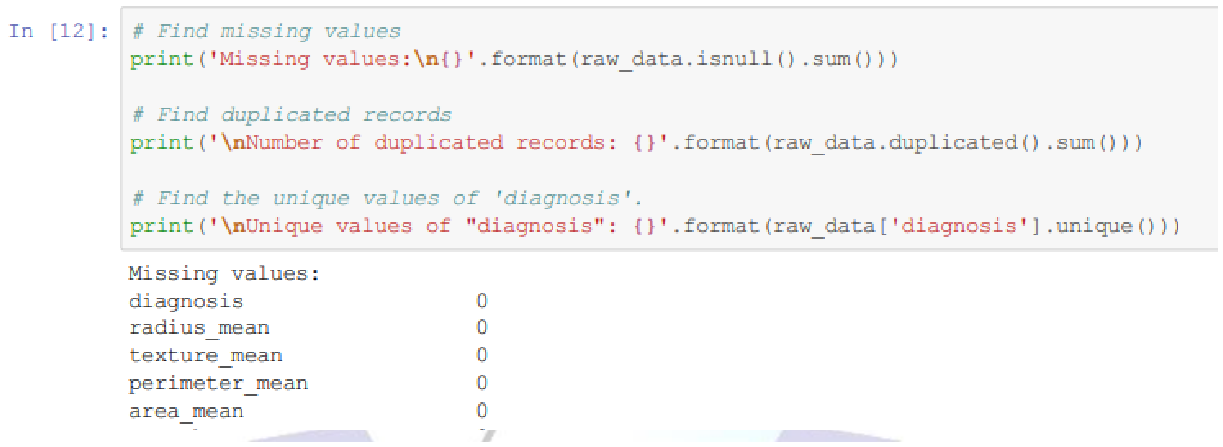
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactn
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280
5	843786	M	12.45	15.70	82.57	477.1	0.12780	0.17000
6	844359	M	18.25	19.98	119.60	1040.0	0.09463	0.10900
7	84458202	M	13.71	20.83	90.20	577.9	0.11890	0.16450
8	844981	M	13.00	21.82	87.50	519.8	0.12730	0.19320
9	84501001	M	12.46	24.04	83.97	475.9	0.11860	0.23960

Figure 4.4 feature of FNA

4.4.1 Handle Missing Values:

Let's compute the number of missing values and determine how to handle them. We can return the number of missing values across the DataFrame by:

- First, use the Pandas DataFrame method isnull() to return a DataFrame containing Boolean values:
- True if the original value is null
- False if the original value isn't null
- Then, use the Pandas DataFrame method sum() to calculate the number of null values in each column.



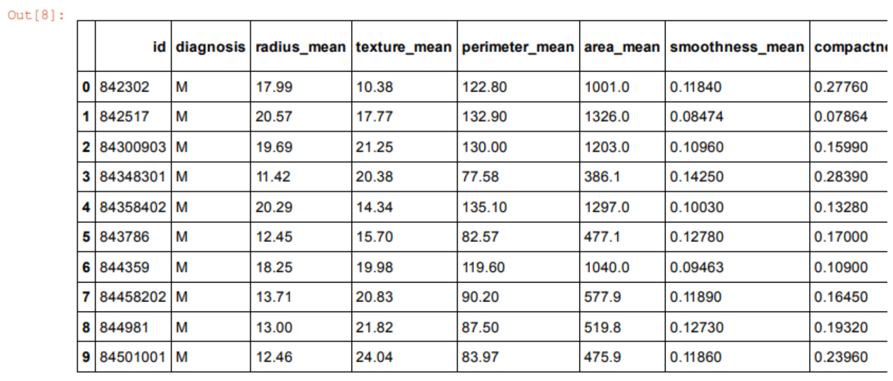
```
In [12]: # Find missing values
         print('Missing values:\n{}'.format(raw_data.isnull().sum()))

         # Find duplicated records
         print('\nNumber of duplicated records: {}'.format(raw_data.duplicated().sum()))

         # Find the unique values of 'diagnosis'.
         print('\nUnique values of "diagnosis": {}'.format(raw_data['diagnosis'].unique()))

         Missing values:
         diagnosis                0
         radius_mean              0
         texture_mean             0
         perimeter_mean           0
         area_mean                0
```

Figure 4.5 code of handling missing value

4.4.2 Dropping Columns in a Data Frame:

Often, you'll find that not all the categories of data in a dataset are useful to you. For example, you might have a dataset containing features about cancer and there is a columns not useful.

In this case, the id and unnamed: 32 categories are not important to you. Retaining these unneeded categories will take up unnecessary space and potentially also bog down runtime.



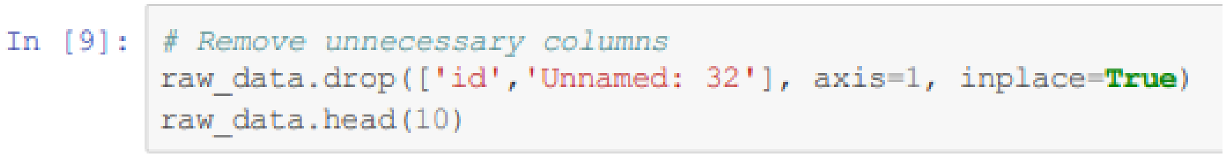
Out[8]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactn
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280
5	843786	M	12.45	15.70	82.57	477.1	0.12780	0.17000
6	844359	M	18.25	19.98	119.60	1040.0	0.09463	0.10900
7	84458202	M	13.71	20.83	90.20	577.9	0.11890	0.16450
8	844981	M	13.00	21.82	87.50	519.8	0.12730	0.19320
9	84501001	M	12.46	24.04	83.97	475.9	0.11860	0.23960

Figure 4.6 unnecessary data

From a Data Frame with the drop() function. Let's look at a simple example where we drop a number of columns from a Data Frame.

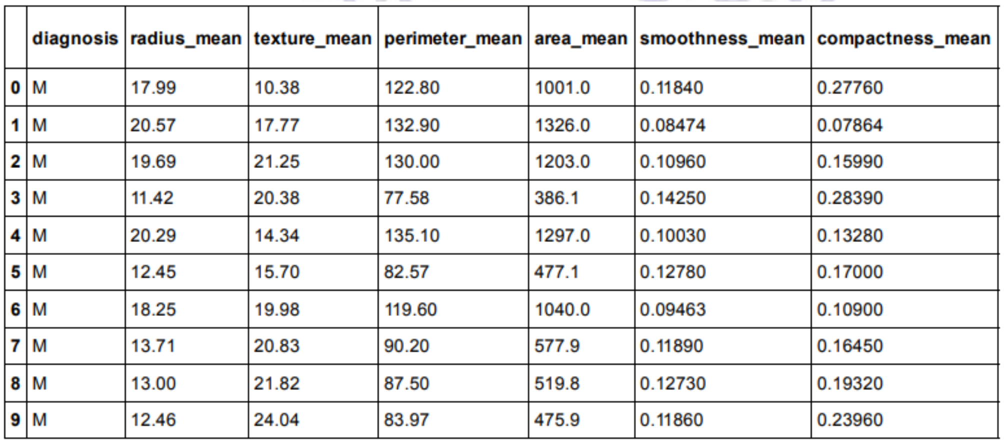
First, let's create a Data Frame out of the CSV file 'BL-Flickr-Images-Book.csv'. In the examples below, we pass a relative path to pd.read_csv, meaning that all of the datasets are in a folder named Datasets in our current working directory:



```
In [9]: # Remove unnecessary columns
        raw_data.drop(['id','Unnamed: 32'], axis=1, inplace=True)
        raw_data.head(10)
```

Figure 4.7 drob unnecessary data

When we look at the first five entries using the head() method, we can see that the id and the unnamed:32 removed.



	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
0	M	17.99	10.38	122.80	1001.0	0.11840	0.27760
1	M	20.57	17.77	132.90	1326.0	0.08474	0.07864
2	M	19.69	21.25	130.00	1203.0	0.10960	0.15990
3	M	11.42	20.38	77.58	386.1	0.14250	0.28390
4	M	20.29	14.34	135.10	1297.0	0.10030	0.13280
5	M	12.45	15.70	82.57	477.1	0.12780	0.17000
6	M	18.25	19.98	119.60	1040.0	0.09463	0.10900
7	M	13.71	20.83	90.20	577.9	0.11890	0.16450
8	M	13.00	21.82	87.50	519.8	0.12730	0.19320
9	M	12.46	24.04	83.97	475.9	0.11860	0.23960

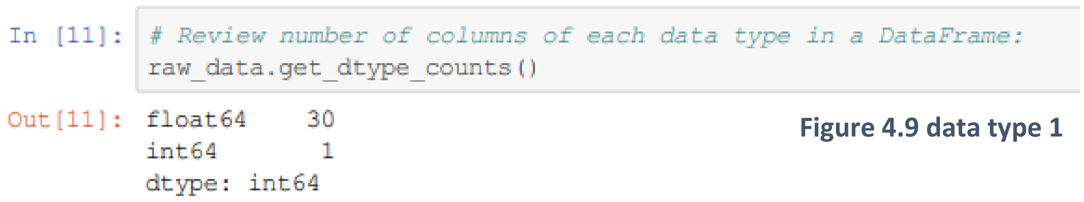
Figure 4.8 feature of FNA 1

4.4.3 Tidying up Fields in the Data:

So far, we have removed unnecessary columns and changed the index of our Data Frame to something more sensible. In this section, we will clean specific columns and get them to a uniform format to get a better understanding of the dataset and enforce consistency. In particular, we will be cleaning Date of Publication and Place of Publication.

Upon inspection, all of the data types are currently the object dtype, which is roughly analogous to str in native Python.

It encapsulates any field that can't be neatly fit as numerical or categorical data. This makes sense since we're working with data that is initially a bunch of messy strings:

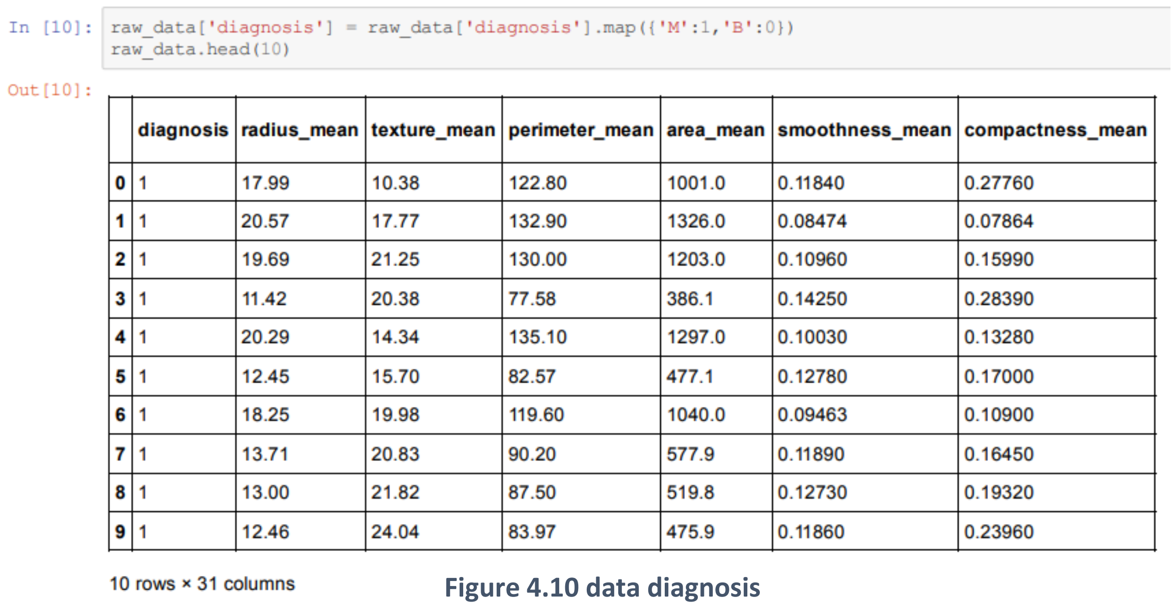


```
In [11]: # Review number of columns of each data type in a DataFrame:
         raw_data.get_dtype_counts()

Out[11]: float64    30
         int64       1
         dtype: int64
```

Figure 4.9 data type 1

4.4.4 Convert Categorical Columns to Numeric Features

First, let's understand the two types of categorical features we have in our dataset and how we can convert each to numerical features:



```
In [10]: raw_data['diagnosis'] = raw_data['diagnosis'].map({'M':1,'B':0})
         raw_data.head(10)
```

Out[10]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
0	1	17.99	10.38	122.80	1001.0	0.11840	0.27760
1	1	20.57	17.77	132.90	1326.0	0.08474	0.07864
2	1	19.69	21.25	130.00	1203.0	0.10960	0.15990
3	1	11.42	20.38	77.58	386.1	0.14250	0.28390
4	1	20.29	14.34	135.10	1297.0	0.10030	0.13280
5	1	12.45	15.70	82.57	477.1	0.12780	0.17000
6	1	18.25	19.98	119.60	1040.0	0.09463	0.10900
7	1	13.71	20.83	90.20	577.9	0.11890	0.16450
8	1	13.00	21.82	87.50	519.8	0.12730	0.19320
9	1	12.46	24.04	83.97	475.9	0.11860	0.23960

10 rows × 31 columns

Figure 4.10 data diagnosis

4.5 Machine Learning and Selection the Algorithm:

I am always ready to learn although I do not always like being taught.

-Winston Churchill

supervised machine learning is one of the most commonly used and successful types of machine learning. supervised learning is used whenever we want to predict a certain outcome from a given input, and we have examples of input/output pairs. We build a machine learning model from these input/output pairs, which comprise our training set. Our goal is to make accurate predictions for new, never-before-seen data. Super- vised learning often requires human effort to build the training set, but afterward automates and often speeds up an otherwise laborious or infeasible task. Classification and Regression There are two major types of supervised machine learning problems, called classification and regression. In classification, the goal is to predict a class label, which is a choice from a predefined list of possibilities.. Classification is sometimes separated into binary classification, which is the special case of distinguishing between exactly two classes, and multiclass classification, which is classification between more than two classes. You can think of binary classification as trying to answer a yes/no question. Classifying emails as either spam or not spam is an example of a binary classification problem. In this binary classification task, the yes/no question being asked would be “Is this email spam?” 25 1 We ask linguists to excuse the simplified presentation of languages as distinct and fixed entities. In binary classification we often speak of one class being the posi- tive class and the other class being the negative class. Here, positive doesn’t represent having benefit or value, but rather what the object of the study is. So, when looking for spam, “positive” could mean the spam class. Which of the two classes is called positive is often a subjective matter, and specific to the domain. The iris example, on the other hand, is an example of a multiclass classification prob- lem. Another example is predicting what language a website is in from the text on the website. The classes here would be a pre-defined list of possible languages.

For regression tasks, the goal is to predict a continuous number, or a floating-point number in programming terms (or real number in mathematical terms).

Predicting a person's annual income from their education, their age, and where they live is an example of a regression task. When predicting income, the predicted value is an amount, and can be any number in a given range. Another example of a regression task is predicting the yield of a corn farm given attributes such as previous yields, weather, and number of employees working on the farm. The yield again can be an arbitrary number. An easy way to distinguish between classification and regression tasks is to ask whether there is some kind of continuity in the output. If there is continuity between possible outcomes, then the problem is a regression problem. Think about predicting annual income. There is a clear continuity in the output. Whether a person makes $40,000 or $40,001 a year does not make a tangible difference, even though these are different amounts of money; if our algorithm predicts $39,999 or $40,001 when it should have predicted $40,000, we don't mind that much. By contrast, for the task of recognizing the language of a website (which is a classifi- cation problem), there is no matter of degree. A website is in one language, or it is in another. There is no continuity between languages, and there is no language that is between English and French.1 Generalization, Overfitting, and Underfitting In supervised learning, we want to build a model on the training data and then be able to make accurate predictions on new, unseen data that has the same characteris- tics as the training set that we used. If a model is able to make accurate predictions on unseen data, we say it is able to generalize from the training set to the test set. We want to build a model that is able to generalize as accurately as possible. Supervised Learning 2 In the real world, this is actually a tricky problem. While we know that the other customers haven't bought a boat from us yet, they might have bought one from someone else, or they may still be saving and plan to buy one in the future. Usually we build a model in such a way that it can make accurate predictions on the training set. If the training and test sets have enough in common, we expect the model to also be accurate on the test set. However, there are some cases where this can go wrong. For example, if we allow ourselves to build very complex models, we can always be as accurate as we like on the training set. Let's take a look at a made-up example to illustrate this point.

Say a novice data scien- tist wants to predict whether a customer will buy a boat, given records of previous boat buyers and customers who we know are not interested in buying a boat.2 The goal is to send out promotional emails to people who are likely to actually make a purchase, but not bother those customers who won't be interested. Suppose we have the customer records shown in Example data about customers Age Number of cars owned Owns house Number of children Marital status Owns a dog Bought a boat 66 1 yes 2 widowed no yes 52 2 yes 3 married no yes 22 0 no 0 married yes no 25 1 no 1 single no no 44 0 no 2 divorced yes no 39 1 yes 2 married yes no 26 1 no 2 single no no 40 3 yes 1 married yes no 53 2 yes 2 divorced no yes 64 2 yes 3 divorced no no 58 2 yes 2 married yes yes 33 1 no 1 single no no After looking at the data for a while, our novice data scientist comes up with the fol- lowing rule: "If the customer is older than 45, and has less than 3 children or is not divorced, then they want to buy a boat." When asked how well this rule of his does, our data scientist answers, "It's 100 percent accurate!" And indeed, on the data that is in the table, the rule is perfectly accurate. There are many possible rules we could come up with that would explain perfectly if someone in this dataset wants to buy a boat. No age appears twice in the data, so we could say people who are 66, 52, 53, or Generalization, Overfitting, and Underfitting | 27 3 And also provably, with the right math. 58 years old want to buy a boat, while all others don't. While we can make up many rules that work well on this data, remember that we are not interested in making pre- dictions for this dataset; we already know the answers for these customers. We want to know if new customers are likely to buy a boat. We therefore want to find a rule that will work well for new customers, and achieving 100 percent accuracy on the training set does not help us there. We might not expect that the rule our data scientist came up with will work very well on new customers. It seems too complex, and it is sup- ported by very little data. For example, the "or is not divorced" part of the rule hinges on a single customer. The only measure of whether an algorithm will perform well on new data is the eval- uation on the test set. However, intuitively3 we expect simple models to generalize better to new data. If the rule was "People older than 50 want to buy a boat," and this would explain the behavior of all the customers, we would trust it more than the rule involving children and marital status in addition to age. Therefore, we always want to find the simplest model. Building a model that is too complex for the amount of information we have, as our novice data scientist did, is called overfitting.

Overfitting occurs when you fit a model too closely to the particularities of the training set and obtain a model that works well on the training set but is not able to generalize to new data. On the other hand, if your model is too simple—say, "Everybody who owns a house buys a boat"—then you might not be able to capture all the aspects of and vari- ability in the data, and your model will do badly even on the training set. Choosing too simple a model is called underfitting. The more complex we allow our model to be, the better we will be able to predict on the training data. However, if our model becomes too complex, we start focusing too much on each individual data point in our training set, and the model will not gener- alize well to new data. There is a sweet spot in between that will yield the best generalization performance. This is the model we want to find. The trade-off between overfitting and underfitting of model complexity against training and test accuracy Relation of Model Complexity to Dataset Size It's important to note that model complexity is intimately tied to the variation of inputs contained in your training dataset: the larger variety of data points your data- set contains, the more complex a model you can use without overfitting. Usually, col- lecting more data points will yield more variety, so larger datasets allow building more complex models. However, simply duplicating the same data points or collect- ing very similar data will not help. Going back to the boat selling example, if we saw 10,000 more rows of customer data, and all of them complied with the rule "If the customer is older than 45, and has less than 3 children or is not divorced, then they want to buy a boat," we would be much more likely to believe this to be a good rule than when it was developed using only the 12 rows. Having more data and building appropriately more complex models can often work wonders for supervised learning tasks. In this book, we will focus on working with datasets of fixed sizes. In the real world, you often have the ability to decide how much data to collect, which might be more beneficial than tweaking and tuning your model. Never underestimate the power of more data. Supervised Machine Learning Algorithms We will now review the most popular machine learning algorithms and explain how they learn from data and how they make predictions. We will also discuss how the concept of model complexity plays out for each of these models, and provide an over- Supervised Machine Learning Algorithms .

Discussing all of them is beyond the scope of the book, and we refer you to the scikit-learn documentation for more details. view of how each algorithm builds a model. We will examine the strengths and weak-nesses of each algorithm, and what kind of data they can best be applied to. We will also explain the meaning of the most important parameters and options.4 Many algorithms have a classification and a regression variant, and we will describe both. It is not necessary to read through the descriptions of each algorithm in detail, but understanding the models will give you a better feeling for the different ways machine learning algorithms can work. This chapter can also be used as a reference guide, and you can come back to it when you are unsure about the workings of any of the algorithms. Some Sample Datasets We will use several datasets to illustrate the different algorithms. Some of the datasets will be small and synthetic (meaning made-up), designed to highlight particular aspects of the algorithms. Other datasets will be large, real-world examples. An example of a synthetic two-class classification dataset is the forge dataset, which has two features. The following code creates a scatter plot visualizing all of the data points in this dataset. The plot has the first feature on the x-axis and the second feature on the y-axis. As is always the case in scatter plots, each data point is represented as one dot. The color and shape of the dot indicates its class:

4.6 Find the available algorithms:

Now that you a clear understanding of where you stand, you can identify the algorithms that are applicable and practical to implement using the tools at your disposal. Some of the factors affecting the choice of a model are:

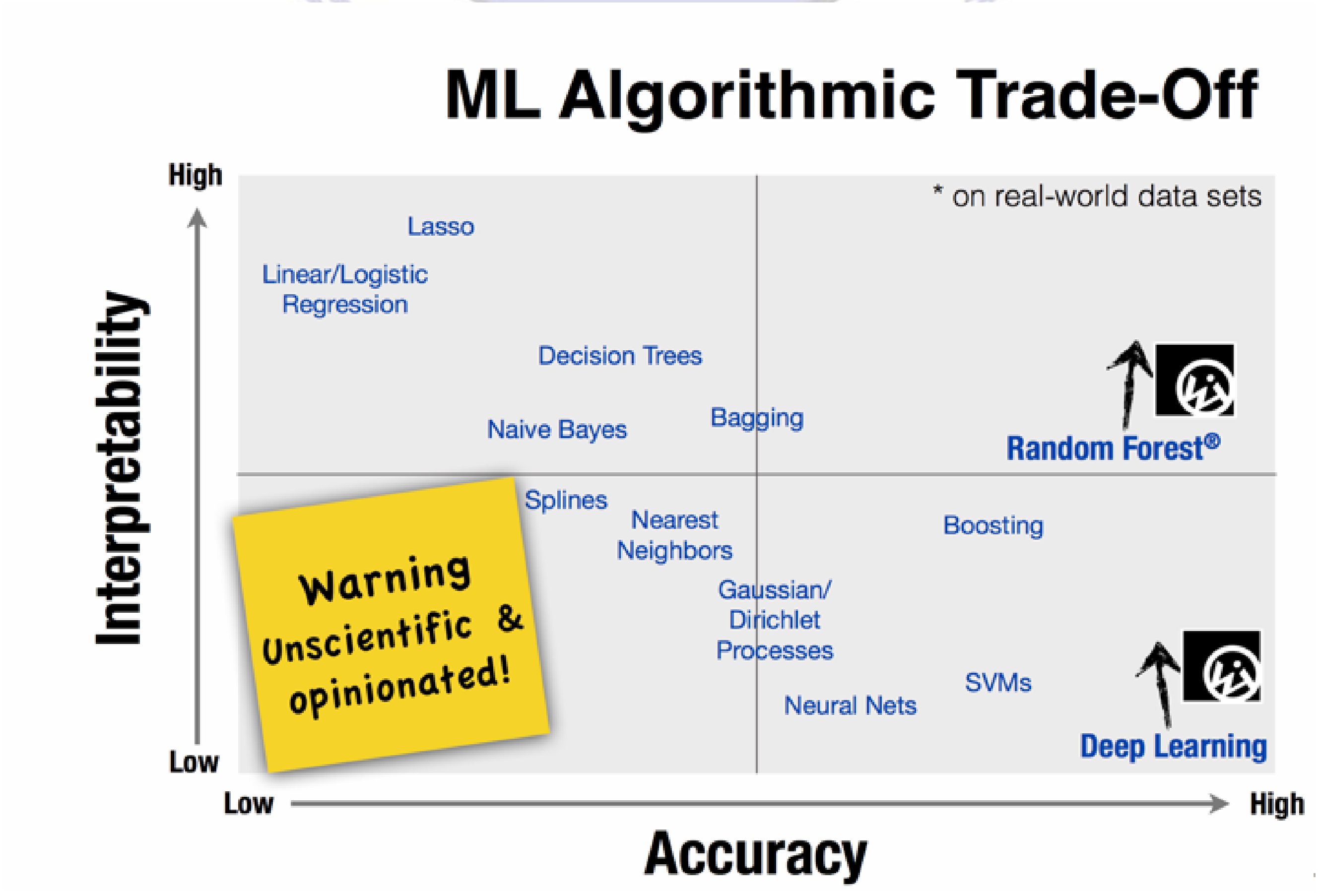
- Whether the model meets the business goals
- How much pre processing the model needs
- How accurate the model is
- How explainable the model is

- How fast the model is: How long does it take to build a model, and how long does the model take to make predictions.
- How scalable the model is An important criteria affecting choice of algorithm is model complexity. Generally speaking, a model is more complex .
- It relies on more features to learn and predict (e.g. using two features vs ten features to predict a target)
- It relies on more complex feature engineering (e.g. using polynomial terms, interactions, or principal components)
- It has more computational overhead (e.g. a single decision tree vs. a random forest of 100 trees).

Besides this, the same machine learning algorithm can be made more complex based on the number of parameters or the choice of some hyperparameters. For example,

- A regression model can have more features, or polynomial terms and interaction terms.
- A decision tree can have more or less depth.

Making the same algorithm more complex increases the chance of overfitting.



- Predicting sales of particular product next month
- Impact of blood alcohol content on coordination

Figure 4.11 ML algorithm 1

- Predict monthly gift card sales and improve yearly revenue projections

4.6.1 Logistic Regression:

Logistic regression performs binary classification, so the label outputs are binary. It takes linear combination of features and applies non-linear function (sigmoid) to it, so it's a very small instance of neural network.

Logistic regression provides lots of ways to regularize your model, and you don't have to worry as much about your features being correlated, like you do in Naive Bayes. You also have a nice probabilistic interpretation, and you can easily update your model to take in new data, unlike decision trees or SVMs. Use it if you want a probabilistic framework or if you expect to receive more training data in the future that you want to be able to quickly incorporate into your model. Logistic regression can also help you understand the contributing factors behind the prediction, and is not just a black box method.

Logistic regression can be used in cases such as:

- Predicting the Customer Churn
- Credit Scoring & Fraud Detection
- Measuring the effectiveness of marketing campaigns

4.6.2 Decision trees:

Single trees are used very rarely, but in composition with many others they build very efficient algorithms such as Random Forest or Gradient Tree Boosting.

Decision trees easily handle feature interactions and they're non-parametric, so you don't have to worry about outliers or whether the data is linearly separable. One disadvantage is that they don't support online learning, so you have to rebuild your tree when new examples come on. Another disadvantage is that they easily overfit, but that's where ensemble methods like random forests (or boosted trees) come in.

Decision Trees great job in practice. A good bet if want something fast and easy that performs pretty well. Its main disadvantage is that it can't learn interactions between features.

Naive Bayes can be used in real-world applications such as:

- Sentiment analysis and text classification
- Recommendation systems like Netflix, Amazon
- To mark an email as spam or not spam
- Face recognition

4.6.3 Random Forest:

Random Forest is an ensemble of decision trees. It can solve both regression and classification problems with large data sets. It also helps identify most significant variables from thousands of input variables. Random Forest is highly scalable to any number of dimensions and has generally quite acceptable performances. Then finally, there are genetic algorithms, which scale admirably well to any dimension and any data with minimal knowledge of the data itself, with the most minimal and simplest implementation being the microbial genetic algorithm. With Random Forest however, learning may be slow (depending on the parameterization) and it is not possible to iteratively improve the generated models

Random Forest can be used in real-world applications such as:

- Predict patients for high risks
- Predict parts failures in manufacturing
- Predict loan defaulters

4.6.4 Neural networks:

Neural Networks take in the weights of connections between neurons . The weights are balanced, learning data point in the wake of learning data point . When all weights are trained, the neural network can be utilized to predict the class or a quantity, if there should arise an occurrence of regression of a new input data point. With Neural networks, extremely complex models can be trained and they can be utilized as a kind of black box, without playing out an unpredictable complex feature engineering before training the model. Joined with the "deep approach" even more unpredictable models can be picked up to realize new possibilities. E.g. object recognition has been as of late enormously enhanced utilizing Deep Neural Networks. Applied to unsupervised learning tasks, such as feature extraction, deep learning also extracts features from raw images or speech with much less human intervention.

On the other hand, neural networks are very hard to just clarify and parameterization is extremely mind boggling. They are also very resource and memory intensive.

Scikit Cheat Sheet

Scikit learning has put a very indepth and well explained flow chart to help you choose the right algorithm that I find very handy.

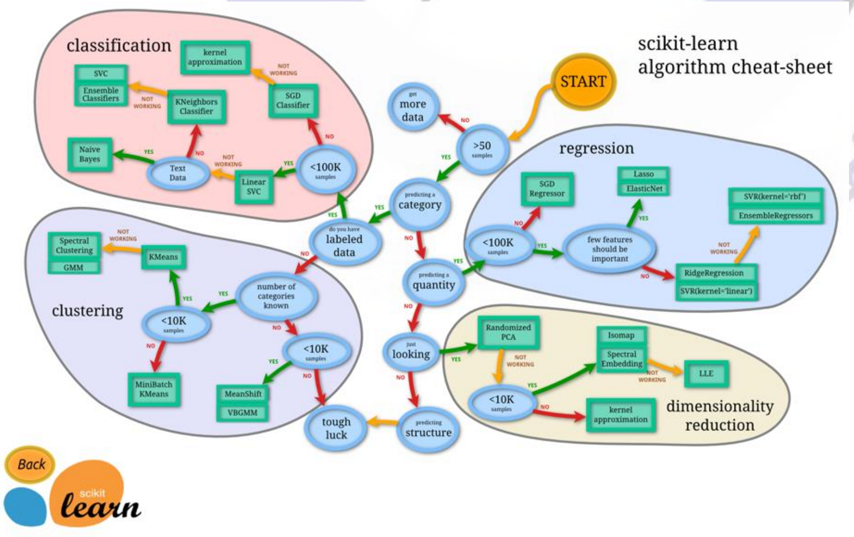
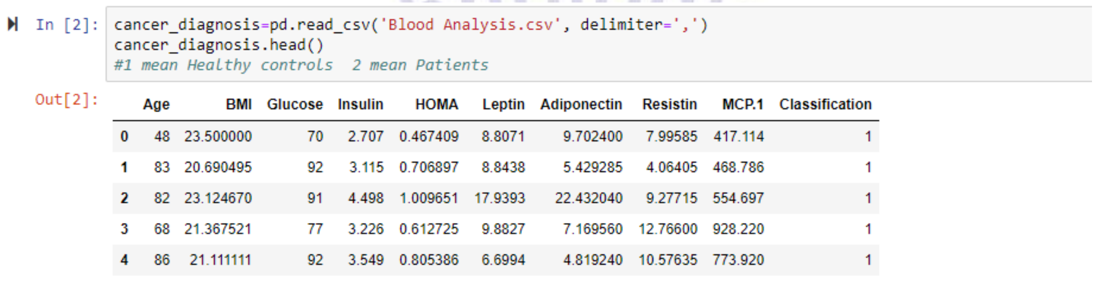


Figure 4.12 neural network algorithm 1

4.6.5 The Algorithms we use and their accuracy:

In blood analysis dataset the goal of using this data was to develop and assess a prediction model which can potentially be used as a biomarker of breast cancer.



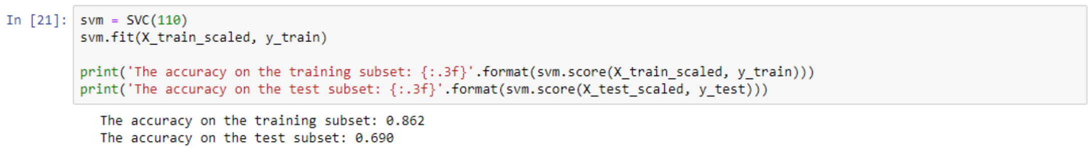
```
In [2]: cancer_diagnosis=pd.read_csv('Blood Analysis.csv', delimiter=',')
        cancer_diagnosis.head()
        #1 mean Healthy controls  2 mean Patients
```

Out[2]:

	Age	BMI	Glucose	Insulin	HOMA	Leptin	Adiponectin	Resistin	MCP.1	Classification
0	48	23.500000	70	2.707	0.467409	8.8071	9.702400	7.99585	417.114	1
1	83	20.690495	92	3.115	0.706897	8.8438	5.429285	4.06405	468.786	1
2	82	23.124670	91	4.498	1.009651	17.9393	22.432040	9.27715	554.697	1
3	68	21.367521	77	3.226	0.612725	9.8827	7.169560	12.76600	928.220	1
4	86	21.111111	92	3.549	0.805386	6.6994	4.819240	10.57635	773.920	1

Figure 4.13 feature of blood analysis

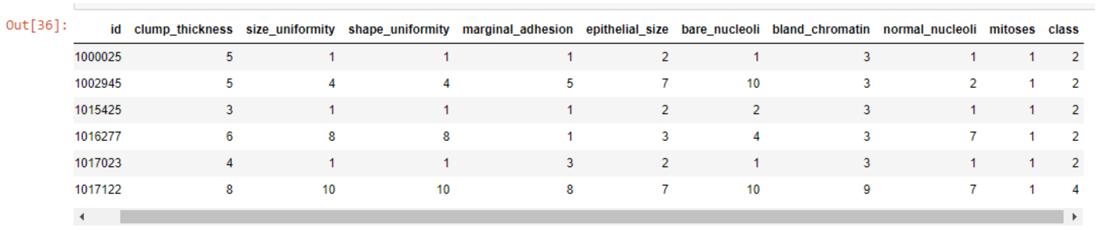
We use SVM algorithm to training this data it give a highly accuracy.



```
In [21]: svm = SVC(110)
         svm.fit(X_train_scaled, y_train)

         print('The accuracy on the training subset: {:.3f}'.format(svm.score(X_train_scaled, y_train)))
         print('The accuracy on the test subset: {:.3f}'.format(svm.score(X_test_scaled, y_test)))

The accuracy on the training subset: 0.862
The accuracy on the test subset: 0.690
```

Figure 4.14 accuraccy of blood analysis data 1

In sample dataset a breast biopsy provides a sample of tissue that doctors use to identify and diagnose abnormalities in the cells that make up breast lumps, other unusual breast changes, or suspicious or concerning findings on a mammogram or ultrasound. The lab report from the breast biopsy can help determine whether you need additional surgery or other treatment.

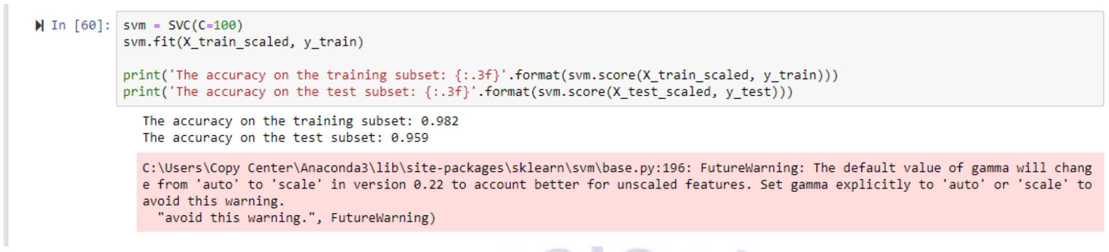


Out[36]:

id	clump_thickness	size_uniformity	shape_uniformity	marginal_adhesion	epithelial_size	bare_nucleoli	bland_chromatin	normal_nucleoli	mitoses	class
1000025	5	1	1	1	2	1	3	1	1	2
1002945	5	4	4	5	7	10	3	2	1	2
1015425	3	1	1	1	2	2	3	1	1	2
1016277	6	8	8	1	3	4	3	7	1	2
1017023	4	1	1	3	2	1	3	1	1	2
1017122	8	10	10	8	7	10	9	7	1	4

Figure 4.15 accuracy of sample

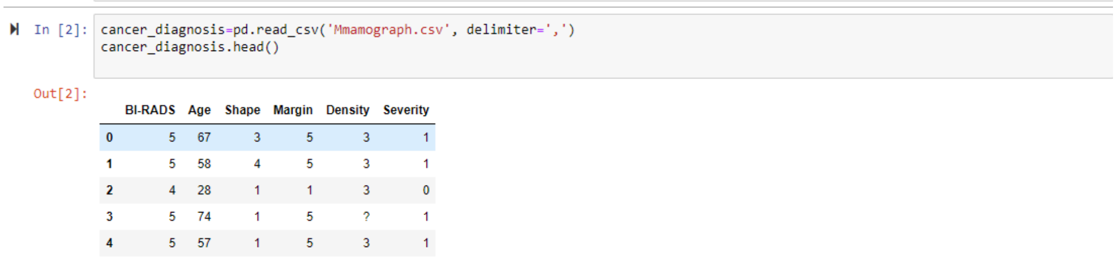
We use SVM algorithm to training this data it give a highly accuracy.



```
In [60]: svm = SVC(C=100)
         svm.fit(X_train_scaled, y_train)

         print('The accuracy on the training subset: {:.3f}'.format(svm.score(X_train_scaled, y_train)))
         print('The accuracy on the test subset: {:.3f}'.format(svm.score(X_test_scaled, y_test)))

The accuracy on the training subset: 0.982
The accuracy on the test subset: 0.959

C:\Users\Copy Center\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will chang
e from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
  "avoid this warning.", FutureWarning)
```

Figure 4.16 accuracy of sample 1

In mammographic dataset the mammograms don't prevent breast cancer, but they can save lives by finding breast cancer as early as possible. ... Finding breast cancers early with mammography has also meant that many more women being treated for breast cancer are able to keep their breasts.

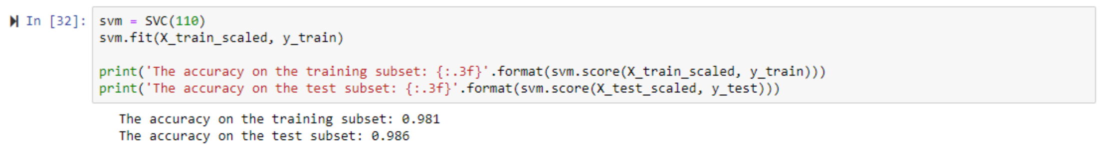


```
In [2]: cancer_diagnosis=pd.read_csv('Mmamograph.csv', delimiter=',')
        cancer_diagnosis.head()

Out[2]:
```

	BI-RADS	Age	Shape	Margin	Density	Severity
0	5	67	3	5	3	1
1	5	58	4	5	3	1
2	4	28	1	1	3	0
3	5	74	1	5	?	1
4	5	57	1	5	3	1

Figure 4.17 feature of mammography data 1

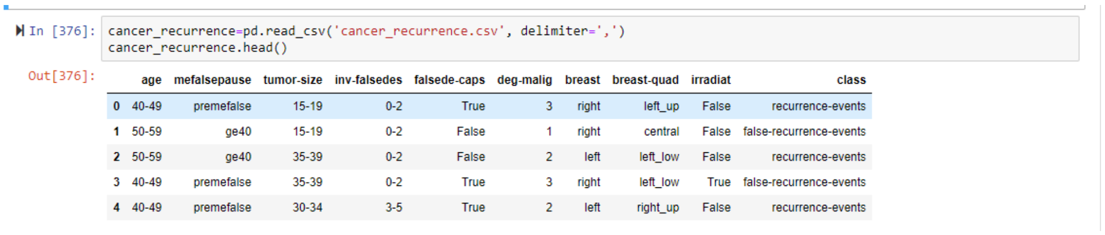
We use SVM algorithm to training this data it give a highly accuracy.



```
In [32]: svm = SVC(110)
         svm.fit(X_train_scaled, y_train)

         print('The accuracy on the training subset: {:.3f}'.format(svm.score(X_train_scaled, y_train)))
         print('The accuracy on the test subset: {:.3f}'.format(svm.score(X_test_scaled, y_test)))

The accuracy on the training subset: 0.981
The accuracy on the test subset: 0.986
```

Figure 4.18 accuracy of mammography

In tumor recurrence dataset this data can be a good guider for physicians in predicting of the breast cancer return.

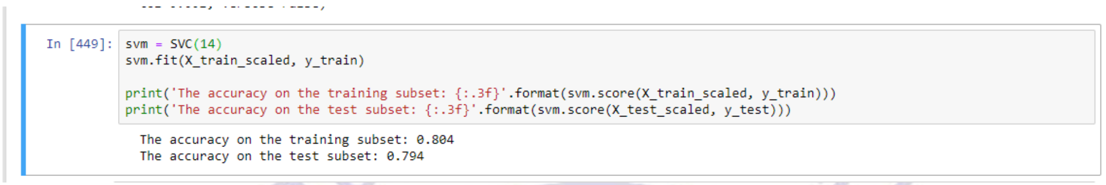


```
In [376]: cancer_recurrence=pd.read_csv('cancer_recurrence.csv', delimiter=',')
          cancer_recurrence.head()
Out[376]:
```

	age	mefalsepause	tumor-size	inv-falsedes	falsede-caps	deg-malig	breast	breast-quad	irradiat	class
0	40-49	premefalse	15-19	0-2	True	3	right	left_up	False	recurrence-events
1	50-59	ge40	15-19	0-2	False	1	right	central	False	false-recurrence-events
2	50-59	ge40	35-39	0-2	False	2	left	left_low	False	recurrence-events
3	40-49	premefalse	35-39	0-2	True	3	right	left_low	True	false-recurrence-events
4	40-49	premefalse	30-34	3-5	True	2	left	right_up	False	recurrence-events

Figure 4.19 feature of recurrency data 1

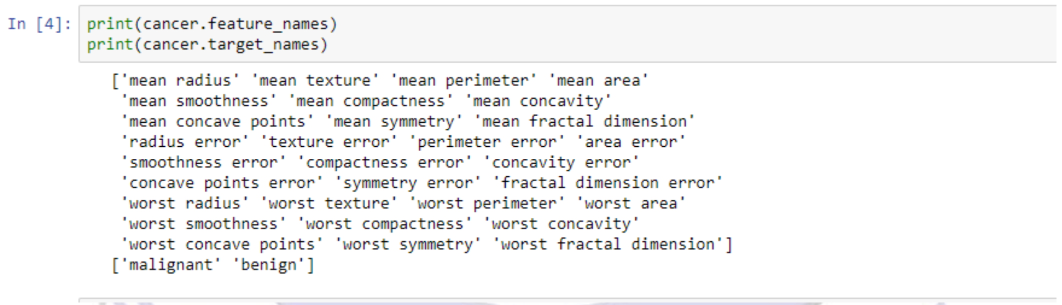
We use SVM algorithm to training this data it give a highly accuracy.



```
In [449]: svm = SVC(14)
          svm.fit(X_train_scaled, y_train)

          print('The accuracy on the training subset: {:.3f}'.format(svm.score(X_train_scaled, y_train)))
          print('The accuracy on the test subset: {:.3f}'.format(svm.score(X_test_scaled, y_test)))

            The accuracy on the training subset: 0.804
            The accuracy on the test subset: 0.794
```

Figure 4.20accuracy od recurrency data 1

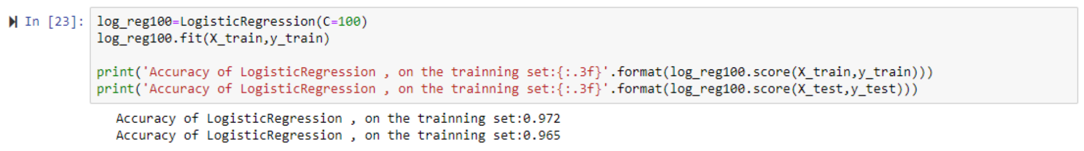
Fine-needle aspiration is a quick way to distinguish between a fluid-filled cyst and a solid mass and, possibly, to avoid a more invasive biopsy procedure.



```
In [4]: print(cancer.feature_names)
        print(cancer.target_names)

          ['mean radius' 'mean texture' 'mean perimeter' 'mean area'
           'mean smoothness' 'mean compactness' 'mean concavity'
           'mean concave points' 'mean symmetry' 'mean fractal dimension'
           'radius error' 'texture error' 'perimeter error' 'area error'
           'smoothness error' 'compactness error' 'concavity error'
           'concave points error' 'symmetry error' 'fractal dimension error'
           'worst radius' 'worst texture' 'worst perimeter' 'worst area'
           'worst smoothness' 'worst compactness' 'worst concavity'
           'worst concave points' 'worst symmetry' 'worst fractal dimension']
          ['malignant' 'benign']
```

Figure 4.21 feature of FNA 1

We use logistic regression algorithm to training this data it give a highly accuracy.



```
In [23]: log_reg100=LogisticRegression(C=100)
         log_reg100.fit(X_train,y_train)

         print('Accuracy of LogisticRegression , on the trainning set:{:.3f}'.format(log_reg100.score(X_train,y_train)))
         print('Accuracy of LogisticRegression , on the trainning set:{:.3f}'.format(log_reg100.score(X_test,y_test)))

           Accuracy of LogisticRegression , on the trainning set:0.972
           Accuracy of LogisticRegression , on the trainning set:0.965
```

Figure 4.22 accuracy of FNA 1

This a Specific purpose app just for a specific customers not for general and we all know that samples from place to another changes according to the status of the people there and their awareness from cancers and others

so we build a function to:

append the new data of the patients to the old samples of our datasets every 100 sample and when it reach to 500 it take the random of the old and the new one .

but because we all know that Algorithms accuracy change according to the number of samples! We have a function to test the accuracy of all Training algorithms and choose the one which have the most accuracy.

And to follow this operation we make our app sending emails with a titles (Checkmate Breast Cancer app) including our status, what algorithm we use and their accuracy when it connected to the internet.

- Machine Learning Course ,Stanford University, Throw Coursera by Andrew NG .
- Understanding Machine Learning with Python by Jurry Kurata throw Plura Sight.
- Data Science from Scratch book by Joel Grus.
- Introduction to Machine Learning with Python book by Andreas C. Müller & Sarah Guido.

4.7 The Dataset we use and their descriptions

4.7.1 Blood Analysis:

Cancer is an open-ended problem till date. It is one of biggest research areas of medical science. There are many types of cancers which are rapidly getting common. We were highly interested to use machine learning models to dive in this dataset and explore about breast cancer predictions.

The goal of using this data was to develop and assess a prediction model which can potentially be used as a biomarker of breast cancer, based on anthropometric data and parameters which can be gathered in routine blood analysis.

In this project, we will use Breast Cancer dataset from University of Coimbra. The data is from archive.ics.uci.edu .

This data consists of 10 variables. The variables are:

Age (years) : Age of the individual.

BMI (kg/m2) : Body mass index of the individual.

Glucose (mg/dL) : Glucose level of the individual.

Insulin (µU/mL) : Insulin level of the individual. Insulin is a hormone made by the pancreas that allows your body to use sugar (glucose) from carbohydrates in the food that you eat for energy or to store glucose for future use.

HOMA : Homeostasis model assessment used to detect insulin resistance and identify patients at high risk of breast cancer development.

Leptin (ng/mL) : Leptin, "the hormone of energy expenditure", is a hormone predominantly made by adipose cells that helps to regulate energy balance by inhibiting hunger. Leptin is opposed by the actions of the hormone ghrelin, the "hunger hormone". Both hormones act on receptors in the arcuate nucleus of the hypothalamus.

Adiponectin (µg/mL) : Adiponectin (also referred to as GBP-28, apM1, AdipoQ and Acrp30) is a protein hormone which is involved in regulating glucose levels as well as fatty acid breakdown. In humans, it is encoded by the ADIPOQ gene and it is produced in adipose tissue.

Resistin (ng/mL) : Resistin also known as adipose tissue-specific secretory factor (ADSF) or C/EBP-epsilon-regulated myeloid-specific secreted cysteine-rich protein (XCP1) is a cysteine-rich adipose-derived peptide hormone that in humans is encoded by the RETN gene.

MCP-1 (pg/dL) : The chemokine (C-C motif) ligand 2 (CCL2) is also referred to as monocyte chemoattractant protein 1 (MCP1) and small inducible cytokine A2. CCL2 is a small cytokine that belongs to the CC chemokine family.

Labels: 1 denotes Healthy controls and 2 denotes Patients.

Plan of action

- Use supervised learning to build a predictive model.
- 80% of the data will be used to train the predictive model, and 20% will be used to test the predictive model. To avoid over-fitting in the model, use cross validation.

- Visualize distributions of training data features using histograms to identify better predictors from the data set.
- Use decision tree classification to build the predictive model.
- Visualize the test data predictions

4.7.2 Mammography:

A mammogram is an x-ray of the breast. While screening mammograms are routinely administered to detect breast cancer in women who have no apparent symptoms, diagnostic mammograms are used after suspicious results on a screening mammogram or after some signs of breast cancer alert the physician to check the tissue. Mammography help with early detection and screening for breast cancer. … A mammography searching for abnormal lesions, benign lumps, or breast cancer is more accurate when performed on women with non-dense breasts, usually older women

How Reliable Are Mammograms For Detecting Cancerous Tumors?

The ability of a mammogram to detect breast cancer may depend on the size of the tumor, the density of the breast tissue, and the skill of the radiologist administering and reading the mammogram. Mammography is less likely to reveal breast tumors in women younger than 50 years than in older women. This may be because younger women have denser breast tissue that appears white on a mammogram. Likewise, a tumor appears white on a mammogram, making it hard to detect.
In this project, we will use Breast Cancer dataset from University of Coimbra. The data is from archive.ics.uci.edu .

This data consists of 10 variables. The variables are:

1. BI-RADS assessment (range from 1 to 5)
2. Age
3. Mass shape (range from 1 to 4)
4. Mass margin (range from 1 to 5)
5. Mass density (range from 1 to 4)
6. Severity: benign=0 or malignant=1

BI–RADS stands for Breast Imaging Reporting and Data System and was established by the American College of Radiology.

BI-RADS is a system that was developed by radiologists for reporting mammogram results using a common language. The radiologist assigns a single digit BI-RADS score (ranging from 0 to 5) when the report of your mammogram is created.

What does BI-RADS 1 mean? BI-RADS 1 means that the mammogram was negative (ie, no cancer) and that you should continue your routine screening.

What does BI-RADS 2 mean? BI-RADS 2 also means that your mammogram was normal (i.e. no cancer), but other findings (such as cysts) are described in the report. You should continue your routine screening.

What does BI-RADS 3 mean? BI-RADS 3 means that your mammogram is probably normal but a repeat mammogram should be completed in 6 months. The chance of breast cancer is approximately 2% in this category. You should make sure that these follow-up mammograms are completed as requested.

What does BI-RADS 4 mean? BI-RADS 4 means that the findings on your mammogram are suspicious and that there is approximately a 23% to 34% chance that this is breast cancer.

What does BI-RADS 5 mean? BI-RADS 5 means that your mammogram results are highly suspicious with a 95% chance of breast cancer.

BI-RADS classifies breast density into four groups:

What does Mass density 1 mean?

Extremely dense: The breasts have a lot of fibrous and glandular tissue.

What does Mass density 2 mean?

Consistent density: The breasts have many areas of fibrous and glandular tissue that are evenly distributed through the breasts. This can make it hard to see small masses in the breast.

What does Mass density 3 mean?

Scattered density: The breasts have quite a bit of fat, but there are a few areas of fibrous and glandular tissue.

What does Mass density 4 mean?

Mostly fatty: The breasts are made up of mostly fat and contain little fibrous and glandular tissue. This means the mammogram would likely show anything that was abnormal

Mass Margin: The margin of a lesion and its relationship with the surrounding tissues (rather than the morphology of the lesion) is emphasized.

What does Mass margin 1 mean?

Circumscribed this is a benign finding

What does Mass margin 2 mean?

Microlobulated this implies a suspicious finding

What does Mass margin 3 mean?

Obscured when the margin is hidden by superimposed fibroglandular tissue

What does Mass margin 4 mean?

Ill-defined this is also a suspicious finding

What does Mass margin 5 mean?

Speculated with radiating lines from mass is a very suspicious finding.

A breast mass appearing with a very irregular or 'random' shape is highly suspicious for breast cancer.

What does Mass shape 1 mean?

round

What does Mass shape 2 mean?

oval

What does Mass shape 3 mean?

lobular

What does Mass shape 4 mean?

Irregular

4.7.3 Sample:

A breast biopsy is a procedure to remove a small sample of breast tissue for laboratory testing.

A breast biopsy is a way to evaluate a suspicious area in your breast to determine whether it is breast cancer. There are several types of breast biopsy procedures.

A breast biopsy provides a sample of tissue that doctors use to identify and diagnose abnormalities in the cells that make up breast lumps, other unusual breast changes, or suspicious or concerning findings on a mammogram or ultrasound. The lab report from the breast biopsy can help determine whether you need additional surgery or other treatment.

Your doctor may recommend a breast biopsy if:

- You or your doctor feels a lump or thickening in your breast, and your doctor suspects breast cancer
- Your mammogram shows a suspicious area in your breast
- An ultrasound scan reveals a suspicious finding
- Your breast MRI reveals a suspicious finding
- You have unusual nipple or areolar changes, including crusting, scaling, dimpling skin or a bloody discharge

In this project, we will use Breast Cancer dataset from University of Coimbra. The data is from archive.ics.uci.edu .

1. Clump Thickness: 1 – 10
2. Uniformity of Cell Size: 1 – 10
3. Uniformity of Cell Shape: 1 – 10
4. Marginal Adhesion: 1 – 10

5. Single Epithelial Cell Size: 1 – 10

6. Bare Nuclei: 1 - 10

7. Bland Chromatin: 1 - 10

8. Normal Nucleoli: 1 - 10

9. Mitoses: 1 - 10

11. Class: (2 for benign, 4 for malignant)

What does clump thickness mean?

Benign cells tend to be grouped in monolayers, while cancerous cells are often grouped in multilayers.

What does Uniformaty of Cell Size mean?

Cancer cells tend to vary in size.

What does Uniformaty of Cell shape mean?

Cancer cells tend to vary in shape.

What does Marginal Adhesion mean?

Normal cells tend to stick together. Cancer cells tends to loose this ability.

What does Single Epithelial Cell Size mean?

It is related to the uniformity mentioned above. Epithelial cells that are significantly enlarged may be a malignant cell.

What does Bare Nuclei mean?

This is a term used for nuclei that is not surrounded by cytoplasm (the rest of the cell). Those are typically seen in benign tumours.

What does Bland Chormatin mean?

In cancer cells the chromatin tend to be more coarse.

What does Normal Nucleoli mean?

In normal cells the nucleolus is usually very small if visible at all. In cancer cells the nucleoli become more prominent

What does Mitoses mean?

Describes the level of mitotic (cell reproduction) activity.

4.7.4 Fine-needle aspiration:

This is the simplest type of breast biopsy and may be used to evaluate a lump that can be felt during a clinical breast exam. For the procedure, you lie on a table. While steadying the lump with one hand, your doctor uses the other hand to direct a very thin needle into the lump.

The needle is attached to a syringe that can collect a sample of cells or

Fluid from the lump.

Fine-needle aspiration is a quick way to distinguish between a fluid-filled cyst and a solid mass and, possibly, to avoid a more invasive biopsy procedure. If, however, the mass is solid, a tissue sample will be obtained

Ten real-valued features are computed for each cell nucleus:
a) Radius (mean of distances from center to points on the perimeter)
b) Texture (standard deviation of gray-scale values)
c) Perimeter
d) Area
e) Smoothness (local variation in radius lengths)
f) Compactness (perimeter^2 / area - 1.0)
g) Concavity (severity of concave portions of the contour)
h) Concave points (number of concave portions of the contour)
i) Symmetry
j) Fractal dimension

4.7.5 tumor recurrence:

The breast cancer is considered as one of the most important and dangerous cancers among women and is known the second factor for the women death. The diagnosis of it means separation of malignant glands from benign tumours.

This paper has been provided by using of the data techniques for increasing diagnosis and predicting it. The data can be a good guider for physicians in predicting of the breast cancer return.

Attribute Information:

1. Age: patient's age at the time of diagnosis

2. Menopause: menopause status of the patient at the time of diagnosis

3. Tumor size: tumor size (in mm).

4. Inv-nodes: range 0 - 39 of axillary lymph nodes showing breast cancer at the time of
Histological examination.

5. Node caps: penetration of the tumor in the lymph node capsule or not.

6. Degree of malignancy: range 1-3 the histological grade of the tumor. That are
Grade: 1 predominantly that consist of cancer cells,
Grade: 2 neoplastic that consist of usual characteristics of cancer cells,
Grade: 3 predominately that consist of cells that are highly affected.

7. Breast: breast cancer may occur in either breast.

8. Breast quadrant: if the nipple consider as a central point the breast may be divided
Into four quadrants.

9. Irradiation: patient's radiation (x-rays) therapy history

Reference

- Machine Learning Course ,Stanford University, Throw Coursera by Andrew NG .
- Understanding Machine Learning with Python by Jurry Kurata throw Plura Sight.
- Data Science from Scratch book by Joel Grus.
- Introduction to Machine Learning with Python book by Andreas C. Müller & Sarah Guido.