**One Year Life Expectancy Post Thoracic Surgery Using IBM Watson**

**NAME**: SANDU JAI VENKATESH

**BRANCH:** COMPUTER SCIENCE OF ENGINEERING AND TECHNOLOGY

**YEAR**: 2ND YEAR

**ROLL NO**: 18R11A05J1

**COLLEGE**: GEETHANJALI COLLEGE OF ENGINEERING AND TECNOLOGY

**TECHNOLOGY**: MACHINE LAERNING

**TITLE OF THE PROJECT**:

One Year Life Expectancy Post Thoracic Surgery Using IBM Watson

**TECHNOLOGY**: MACHINE LAERNING

Skills Required:Python, Python For Data Analysis, Machine Learning, IBM Cloud, IBM Watson

# INDEX

1. **Introduction** 
   1. **About IBM Watson API**
   2. **Why IBM Watson API**
2. **Literature Survey** 
   1. **Project Literature**
   2. **Introduction To Python**
      1. **Python Technology**
3. **System Analysis And Requirements**
   1. **Software And Hardware Requirements**
      1. **Hardware Requirements**
      2. **Software Requirements**

**3.2 Performance Requirements**

1. **Problem Statement**
2. **Solution**

**Introduction**

Lung cancer is the most common form of cancer world-wide, and the most common cause of cancer death. Radical surgical resection, with or without adjuvant treatment, is still a Prerequisite for cure. In spite of different additional modes of treatment, survival is still poor. It is important to have knowledge of peri- and postoperative mortality (life expectancy) and morbidity (health complications), and also of risk factors prior to surgery, to be able to improve the quality of operative procedures and identify patients running the highest risk. This helps to optimize the patient’s condition, medication and respiratory status before surgery. Furthermore, the operative risks must be considered in relation to the long-term results in order to identify patients who will clearly benefit from surgery.

OBJECTIVES

Preoperative lung function is an independent predictor of long-term survival after lung resection for non-small-cell lung cancer (NSCLC). The extent of resection has an impact on operative mortality, determines postoperative lung function and may influence both overall- and cancer-specific survival. We sought to determine the impact of predicted postoperative (ppo) lung function on long-term survival after lung cancer resection.

METHODS

We previously reported long-term survival analyses for patients who underwent major lung resection for NSCLC 1980–2006. For this study, we calculated ppo spirometry (forced expiratory volume in the first second, FEV1) and diffusing capacity of the lung for carbon monoxide (DLCO) in the same cohort using the functional segment technique or quantitative perfusion scans when available, and updated survival data; missing data were imputed. We assessed the relationship of ppoFEV1 and ppoDLCO to long-term survival using Cox regression.

RESULTS

Of 854 patients, 471 (55%) were men, the mean age was 63 years and median survival was 42 months. At the time of analysis, 70% of patients had died. On regression analysis, all-cause mortality was related to age, stage, performance status, renal function and prior myocardial infarction. Preoperative lung function was marginally associated with mortality [DLCO (10-percentage point decrease): HR (hazard ratio) 1.04, 95% confidence interval (95% CI) 1.00–1.08, P = 0.056; FEV1 (10-percentage point decrease): HR 1.04, 95% CI 1.00–1.09, P = 0.067]. In contrast, ppo lung function was strongly associated with mortality (ppoDLCO: HR 1.06, 95% CI 1.01–1.12, P = 0.024; ppoFEV1: HR 1.06, 95% CI 1.01–1.12, P = 0.031).

CONCLUSIONS

Ppo lung function is strongly associated with long-term survival after major lung resection and is more strongly related to survival than preoperative lung function. Surgeons struggle with challenging decisions about the appropriate extent of resection for early-stage cancer, balancing factors such as operative morbidity/mortality, local recurrence and postoperative quality of life. Ppo lung function and its relation to survival also should be taken into consideration during such deliberations.

* 1. **About IBM Watson API**

Watson comes pre-integrated and pre-trained on a flexible information architecture optimized to accelerate production and deployment of AI. Build models and develop applications to help your business make more accurate predictions, automate processes, interact with users and customers, and augment expertise.

Developer tools that make it easy to incorporate conversation, language, and search into your applications. Watson gives you access to detailed developer resources that help you get started fast, including documentation and SDKs on GitHub.

**IBM Watson Studio** is an integrated environment designed to make it easy to develop, train, manage models, and deploy AI-powered applications and is a SaaS solution delivered on the **IBM** Cloud. It is evolving Data Science Experience on **IBM** Cloud with lot of new features to build AI applications.

* 1. **Why IBM Watson API**

IBM Watson® Studio helps data scientists and analysts prepare data and at scale across any cloud. With its open, flexible multicloud architecture, Watson Studio provides capabilities that empower businesses to simplify enterprise data science and AI:

Automate AI lifecycle management with AutoAI

Visually prepare and build models with IBM

Build models using images with IBM Watson and texts with IBM Watson Natural

Deploy and run models through one-click integration with Machine learning

Manage and monitor models through integration with IBM Watson OpenScale

**2.LITERATURE SURVEY**

**2.1 PROJECT LITERATURE**

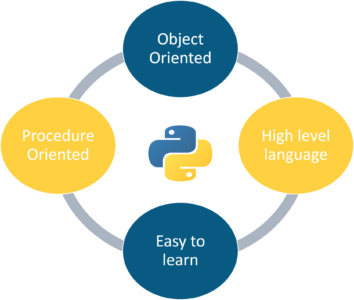
Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

**2.2. INTRODUCTION TO PYTHON**

Python is a popular programming language, Python is a high-level, interpreted, interactive and object oriented-scripting language. Python is Interpreted Python is Interactive Python is Object-Oriented Python is Beginner's Language Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python's feature highlights include: Easy-to-learn Easy-to-read Easy-to-maintain A broad standard library Interactive Mode Portable Extendable Databases GUI Programming Scalable

Python language is a high level language that can be characterized by the following buzzwords.

* Object-Oriented
* Scripting
* Rapid Prototyping
* Steering
* Rapid Application Development
* Productivity



With most programming languages, you either compile or interpret a program so that you can run it on your computer.

* **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Python enables programs to be written compactly and readably. Programs written in Python are typically much shorter than equivalent C, C++, or Java programs, for several reasons:

• the high-level data types allow you to express complex operations in a single statement;

• statement grouping is done by indentation instead of beginning and ending brackets;

• no variable or argument declarations are necessary

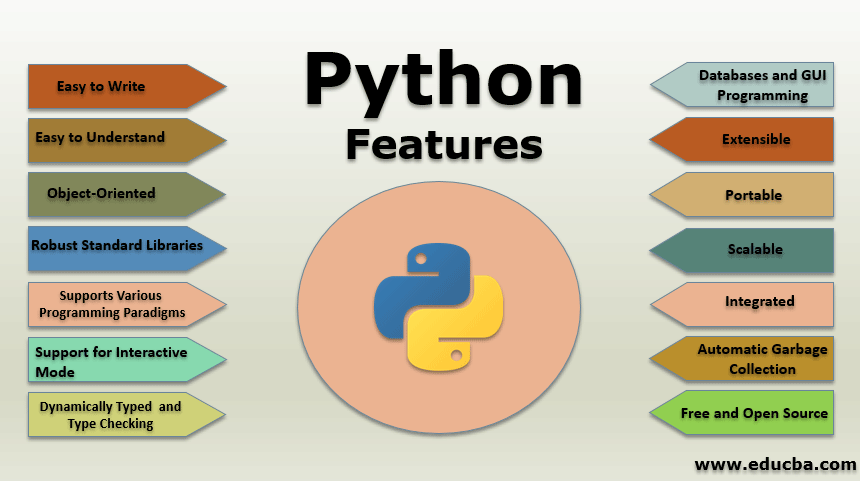
## **Python Features**

Python's features include −

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
* **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases** − Python provides interfaces to all major commercial databases.
* **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* It supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

****

**2.2.1 Python Technology**

Python is everywhere. You may not even realize how widespread it is. Most of all, [Python is easy to learn, clear to read, and simple to write in](https://stxnext.com/ebooks/introduction-python-tech-managers/). This speeds up development without sacrificing reliability or scalability. It can be used in the following :

* Artificial Intelligence with Python
* Bio python
* Bokeh
* Beautiful Soup
* Concurrency in python
* Cryptography
* Jupyter
* Jython
* Logistic Regression in Python
* Matplotlib
* Machine Learning with Python
* NumPy
* Object Oriented Python
* PyGTK
* PyQt
* Pycharm
* Python MYSQL

## List of Python Web Frameworks:

### 1. Full-Stack Frameworks

This type of framework acts as a one-stop solution for fulfilling all the developers’ necessary requirements. Form validation, form generators, and template layouts are the components that are commonly included in the full-stack frameworks.

* CubicWeb
* [Django](https://mindmajix.com/python-django-training)
* Giotto
* Pylons Framework
* Pyramid
* TurboGears
* Web2Py



### 2. Microframeworks

These frameworks are also called non-full stack frameworks. These are lightweight in nature because they do not offer more components as a full-stack framework. The developers who want to use this framework must put a lot of effort in coding and also in adding additional requirements manually.

* Bottle
* CherryPy
* Dash
* Falcon
* Flask
* Hug
* MorePath
* Pycnic

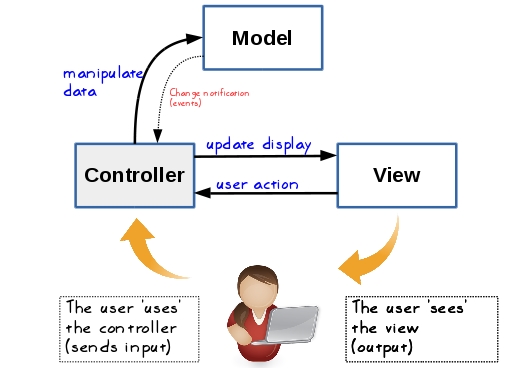
### 3. Asynchronous Framework

It is also a microframework that mainly allows us in handling a huge set of concurrent connections. This asynchronous framework is mainly built for Python and it also uses the asyncio library.

* AIOHTTP
* Growler
* Sanic
* Tornado

**2.2.2 MVC Architecture**

MVC is a widely used software architectural pattern in GUI-based applications. It has three components, namely a **model** that deals with the business logic, a **view** for the user interface, and a **controller** to handle the user input, manipulate data, and update the view. The following is a simplified schematic that shows the basic interactions between the various components:



### Model:

It consists of pure application logic, which interacts with the database. It includes all the information to represent data to the end user.

### View:

View represents the HTML files, which interact with the end user. It represents the model’s data to user.

### Controller:

It acts as an intermediary between view and model. It listens to the events triggered by view and queries model for the same.

**2.2.3 Tkinter**

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit

## Tkinter Widgets

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

There are currently 15 types of widgets in Tkinter. We present these widgets as well as a brief description in the following table –

|  |  |
| --- | --- |
| 1 | [Button](https://www.tutorialspoint.com/python/tk_button.htm)  The Button widget is used to display buttons in your application. |
| 2 | [Canvas](https://www.tutorialspoint.com/python/tk_canvas.htm)  The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application. |
| 3 | [Checkbutton](https://www.tutorialspoint.com/python/tk_checkbutton.htm)  The Checkbutton widget is used to display a number of options as checkboxes. The user can select multiple options at a time. |
| 4 | [Entry](https://www.tutorialspoint.com/python/tk_entry.htm)  The Entry widget is used to display a single-line text field for accepting values from a user. |
| 5 | [Frame](https://www.tutorialspoint.com/python/tk_frame.htm)  The Frame widget is used as a container widget to organize other widgets. |
| 6 | [Label](https://www.tutorialspoint.com/python/tk_label.htm)  The Label widget is used to provide a single-line caption for other widgets. It can also contain images. |
| 7 | [Listbox](https://www.tutorialspoint.com/python/tk_listbox.htm)  The Listbox widget is used to provide a list of options to a user. |
| 8 | [Menubutton](https://www.tutorialspoint.com/python/tk_menubutton.htm)  The Menubutton widget is used to display menus in your application. |
| 9 | [Menu](https://www.tutorialspoint.com/python/tk_menu.htm)  The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton. |
| 10 | [Message](https://www.tutorialspoint.com/python/tk_message.htm)  The Message widget is used to display multiline text fields for accepting values from a user. |
| 11 | [Radiobutton](https://www.tutorialspoint.com/python/tk_radiobutton.htm)  The Radiobutton widget is used to display a number of options as radio buttons. The user can select only one option at a time. |
| 12 | [Scale](https://www.tutorialspoint.com/python/tk_scale.htm)  The Scale widget is used to provide a slider widget. |
| 13 | [Scrollbar](https://www.tutorialspoint.com/python/tk_scrollbar.htm)  The Scrollbar widget is used to add scrolling capability to various widgets, such as list boxes. |
| 14 | [Text](https://www.tutorialspoint.com/python/tk_text.htm)  The Text widget is used to display text in multiple lines. |
| 15 | [Toplevel](https://www.tutorialspoint.com/python/tk_toplevel.htm)  The Toplevel widget is used to provide a separate window container. |
| 16 | [Spinbox](https://www.tutorialspoint.com/python/tk_spinbox.htm)  The Spinbox widget is a variant of the standard Tkinter Entry widget, which can be used to select from a fixed number of values. |
| 17 | [PanedWindow](https://www.tutorialspoint.com/python/tk_panedwindow.htm)  A PanedWindow is a container widget that may contain any number of panes, arranged horizontally or vertically. |
| 18 | [LabelFrame](https://www.tutorialspoint.com/python/tk_labelframe.htm)  A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts. |
| 19 | [tkMessageBox](https://www.tutorialspoint.com/python/tk_messagebox.htm)  This module is used to display message boxes in your applications. |

* **Standard attributes:**
* Dimensions
* Colors
* Fonts
* Anchors
* Relief styles
* Bitmaps
* Cursors

**3.SYSTEM ANALYSIS AND REQUIREMENTS**

**3.1 FEASIBILITY STUDY:**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**3.1.1 ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### 3.1.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**3.1.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

* 1. **SOFTWARE AND HARDWARE REQUIREMENTS**

**3.2.1 Hardware Requirements:**

* Processor : Core 2 Duo or Higher
* RAM : 1 GB
* Hard disk : 10 GB

**3.2.2 Software Requirements:**

* Operating Systems : Any Graphical OS
* Language : Python
* Database : SQLite
* Tool : Pychram or Microsoft Visual Code

**3.3 PERFORMANCE REQUIREMENTS**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

The system should be able to interface with the existing system

* The system should be accurate
* The system should be better than the existing system

The existing system is completely dependent on the user to perform all the duties.

1. **Problem Statement**

**Description**

Lung cancer is the most common form of cancer world-wide, and the most common cause of cancer death. Radical surgical resection, with or without adjuvant treatment, is still a Prerequisite for cure. In spite of different additional modes of treatment, survival is still poor. It is important to have knowledge of peri- and postoperative mortality (life expectancy) and morbidity (health complications), and also of risk factors prior to surgery, to be able to improve the quality of operative procedures and identify patients running the highest risk. This helps to optimize the patient’s condition, medication and respiratory status before surgery. Furthermore, the operative risks must be considered in relation to the long-term results in order to identify patients who will clearly benefit from surgery.

1. **Solution**

The aim of the project is to examine the operative mortality (life expectancy) and morbidity (Health issues) after lung cancer surgery and to identify factors associated with an adverse Outcome. IBM Watson AutoAI Machine Learning Service is developed to predict the post operative life expectancy of lung cancer patients using the computational methods. These methods were used specifically to predict whether a lung cancer patient will survive one year after he or she has had thoracic surgery. The results of each of the techniques were then measured and compared based on accuracy and performance.. The model is deployed on IBM cloud to get scoring end point which can be used as API in mobile app or web app building. We are developing a web application which is built using node red service. We make use of the scoring end point to give user input values to the deployed model. The model prediction is then showcased on User Interface.