## PREDICTING COVID-19 PANDEMIC USING MACHINE LEARNING

***Project submitted in partial fulfillment of the requirements for the award of the degree of***

# BACHELOR OF TECHNOLOGY

### IN

**COMPUTER SCIENCE AND ENGINEERING**

**BY**

**T.AKHIL (18C91A0599)**

**T.DIVYASRI (18C91A05A0)**

**V.PRAVALYA (18C91A05B1)**

**Under the Esteemed guidance of**

**Mr.RAVINDHER M.Tech**

Assistant Professor



### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**HOLY MARY INSTITUTE OF TECHNOLOGY & SCIENCE**

**(COLLEGE OF ENGINEERING)**

***(Approved by AICTE New Delhi, Permanently Affiliated to JNTU Hyderabad, Accredited by NAAC with ‘A’ Grade)***

**Bogaram (V), Keesara (M), Medchal District -501 301. - 11**

### 2021 - 2022

**HOLY MARY INSTITUTE OF TECHNOLOGY & SCIENCE**

#### (COLLEGE OF ENGINEERING)

***(Approved by AICTE New Delhi, Permanently Affiliated to JNTU Hyderabad, Accredited by NAAC with ‘A’ Grade)***

**Bogaram (V), Keesara (M), Medchal Dist-501301.**

### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



**CERTIFICATE**

This is to certify that the mini project entitled “PREDICTING COVID-19 PANDEMIC USING MACHINE LEARNING” is being submitted by T.AKHIL (18C91A0599), T.DIVYASRI (18C91A05A0), V.PRAVALYA(18C91A05B1), in Partial fulfillment of the academic requirements for the award of the degree of Bachelor of Technology in “COMPUTER SCIENCE AND ENGINEERING” HOLY MARY INSTITUTE OFTECHNOLOGY & SCIENCE, JNTU Hyderabad during the year 2021-2022.

**INTERNAL GUIDE HEAD OF THE DEPARTMENT**

Mr. RAVINDER M.Tech DR .B.NARSIMHA M.Tech, Ph.D.

Assistant Professor Professor & HoD

Dept. of Computer Science & Engineering. Dept. of Computer Science & Engineering

### EXTERNAL EXAMINER

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, who’s constant guidance and encouragement crowns all effort with success.

I take this opportunity to express my profound gratitude and deep regards to My Guide **Mr. RAVINDER, Assistant Professor**, Dept. of Computer Science & Engineering, Holy Mary Institute of Technology & Science for his / her exemplary guidance, monitoring and constant encouragement throughout the project work.

My special thanks to **Dr. B. Narsimha, Head of the Department**, Dept. of Computer Science & Engineering, Holy Mary Institute of Technology & Science who has given an immense support throughout the course of the project.

I also thank to **Dr. P. Bhaskara Reddy,** the **honorable Director** of my college Holy Mary Institute of Technology & Science for providing me the opportunity to carry out this work.

At the outset, I express my deep sense of gratitude to the beloved **Chairman A. Siddartha Reddy** of **Holy Mary Institute of Technology & Science**, for giving me the opportunity to complete my course of work

I am obliged to **staff members** of Holy Mary Institute of Technology & Science for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of my assignment.

Last but not the least I thank **ALMIGHTY** and My **Parents**, and **Friends** for their constant encouragement without which this assignment would not be possible.

**T.AKHIL (18C91A0599)**

**T.DIVYASRI (18C91A05A0)**

**V.PRAVALYA (18C91A05B1)**

**DECLARATION**

This is to certify that the work reported in the present project titled **“PREDICTION OF COVID-19 PANDEMIC USING MACHINE LEARNING”** is a record of

work done by me in the Department of Computer Science & Engineering, Holy Mary Institute of Technology and Science.

No part of the thesis is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text the reported are based on the project work done entirely by me not copied from any other source.

**T.AKHIL (18C91A0599)**

**T.DIVYASRI (18C91A05A0)**

**V.PRAVALYA (18C91A05B1)**

CONTENTS

ABSTRACT

Name of the Chapter Page No.

### 1.INTRODUCTION

PROBLEM STATEMENT----------------------------------------------------------8

MOTIVATION----------------------------------------------------------------------12

EXISTING SYSTEM---------------------------------------------------------------14

PROPOSED SYSTEM-------------------------------------------------------------15

**2.LITERATURE SURVEY**

RESEARCH QUESTIONS---------------------------------------------------------16

PREDICTION MODELS---------------------------------------------------------- 16 APPLICATIONS ------------------------------------------------------------------- 17

SUMMARY--------------------------------------------------------------------------20

**3.SOFTWARE REQUIREMENTS SPECIFICATIONS**

SOFTWARE REQUIREMENTS---------------------------------------------------------22

HARDWARE REQUIREMENTS------------------------------------------------22

# 4. SYSTEM DESIGN

## DIAGRAM--------------------------------------------------------------------------23

# 5.IMPLEMENTATION

## MODULES-------------------------------------------------------------------------24

## SAMPLE CODE-------------------------------------------------------------------55

## SOFTWARE DESCRIPTION---------------------------------------------------41

### 6.SYSTEM TESTING

## TESTS----------------------------------------------------------------56

# 7.RESULTS----------------------------------------------------------57

# 8.RESULTSCREEN SHOTS-------------------------------------58

# 9.CONCLUSION----------------------------------------------------64

# 10.BIBLIOGRAPHY-----------------------------------------------66

# ABSTRACT :

In this project we will building a machine learning model using python to discuss the spread of corona virus as well as certain steps taken by governments of countries such as india,china,south korea,and Italy.The covid 19 is on the spread it basically goes through four stages.The outbreak of corona virus is obviously developing in to a major international crisis and lending to influence the most important aspects of our daily lifes.The objective of this project is to come up with a strong machine learning module which predicts how the virus could spread across different states in regions the goal of this is to come up with something that predicts the spread of the virus in the next week.This model is written in Jupiter and its being executed using kolob.

**PREDICTING COVID-19 USING MACHINE LEARNING**

# INTRODUCTION

## PROBLEM STATEMENT:

Machine Learning is a subset of artificial intelligence which focuses mainly on machine learning from their experience and making predictions based on its experience.

**What does it do**? It enables the computers or the machines to make data-driven decisions rather than being explicitly programmed for carrying out a certain task. These programs or algorithms are designed in a way that they learn and improve over time when are exposed to new data. As you know, we are living in the world of humans and machines. The Humans have been evolving and learning from their past experience since millions of years. On the other hand, the era of machines and robots have just begun. You can consider it in a way that currently we are living in the primitive age of machines, while the future of machine is enormous and is beyond our scope of imagination.

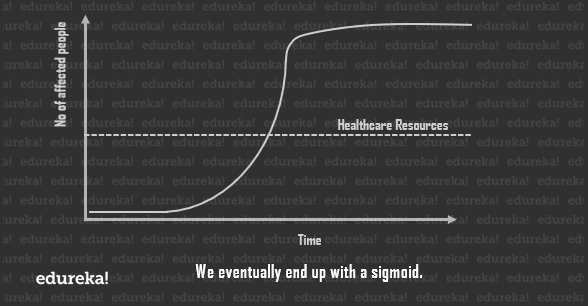
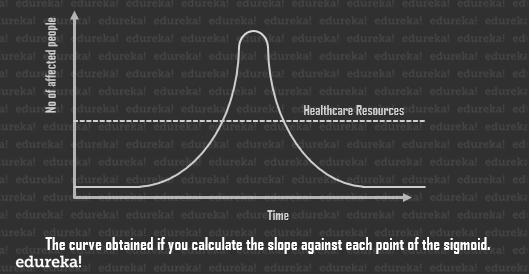
In today’s world, these machines or the robots have to be programmed before they start following your instructions. But what if the machine started learning on their own from their experience, work like us, feel like us, do things more accurately than us? These things sound fascinating, Right? Well, just remember this is just the beginning of the new era.

Corona Virus disease (COVID-19) is an infectious disease caused by a newly discovered virus, which emerged in Wuhan, China in December of 2019.Most people infected with the COVID-19 virus will experience mild to moderate respiratory illness and recover without requiring special treatment.  Older people and those with underlying medical problems like cardiovascular disease, diabetes, chronic respiratory disease, and cancer are more likely to develop serious illness.The COVID-19 virus spreads primarily through droplets of saliva or discharge from the nose when an infected person coughsor sneezes, so you might have heard caution to practice respiratory etiquette (for example, by coughing into a flexed elbow)**How does a Pandemic Work?**

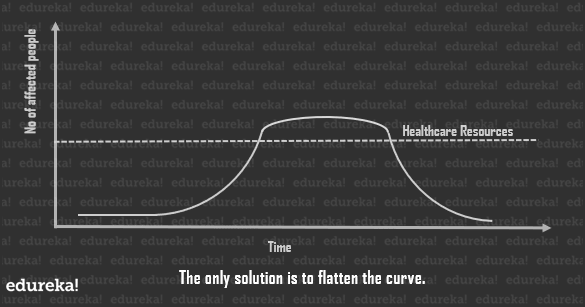
To understand this better let’s look a small riddle.There’s a glass slide held under a microscope which consists of a specific germ. This germ has a property to double every day. So on the first day, there’s one, on the second day there are two, on the third day there four a



nd the fourth day eight, and so on.On the 60th day, the slide is full. So on which day is the slide half full?Day 59. But of course, you knew that But onwhich day, is the slide 1% full? Surprisingly, ***not until the 54th day!*** What it means that the slide goes from being 1% full to 100% in less than a week and hence, displays a property called ***exponential growth***. And this is also how a Pandemic works. The outbreak is fairly unnoticeable in the beginning, then, once it reaches a significant value, the growth to maxima is extremely quickcannot go on forever. The virus will eventually stop finding people to infect and ultimate will slow down the count. This is called ***logistic growth***and the curve is known as a ***sigmoid.***

Now every point in the curve will give you to running total of cases of the current day. But if you delve a little into [***statistics***](https://www.edureka.co/blog/mathematics-for-machine-learning/), you’ll discover that by plotting the slope of each day, you shall get the new cases per day. There are fewer new cases right at the beginning and at the end, with a sharp rise in the stages in between.As you can see, the peak of the curve may greatly overwhelm our healthcare systems, which is the amount of resources available to us for the care of affected individuals at any given point in time.

Since we can’t really help the total number of individuals affected by the pandemic, the best solution is to ***flatten the curve***so as to bring down the total number of cases, at any given point in time, as close to the healthcare line as possible.

This spreads the duration of this whole process a little longer, but since the healthcare system can tend to the number of cases at any given point in time, the casualties are way lower.

***The Solution***

Social Distancing. The logic here is, the virus can’t infect bodies if it cannot find bodies to infect!

World Leaders in all affected countries announced quarantines and lock-downs to keep their folks safe and away from anything or anyone that could infect them, all big social events were postponed and all major sports leagues cancelled as well.

# MOTIVATION:

Conducted for the first time on Iranian patients, this study provided and compared three practical prognostic models using invasive and non-invasive data from the first day of patients’ admission to predict the COVID19 mortality. Furthermore, the prediction power of non-invasive and invasive feature groups was evaluated across the temporal and feature number spectrum to reveal interesting results. Compared with the invasive model, the non-invasive model provided better performances in lower, sparse feature dimensions, pointing to the presence of a significant concentration of prediction information in several non-invasive features. In contrast, a more disperse distribution of prediction information was observed among invasive features. Furthermore, while invasive features were good predictors for imminent expiration, they were outperformed by non-invasive features for a more distant expiration interval.Predicting the trajectory destination of COVID-19 could provide substantial support for decreasing mortality rates.

In a pandemic, rapid disease transmission and high patient load could quickly overload healthcare infrastructures; an overloaded medical system can result in higher mortality rates due to inefficient management of limited medical resources; this issue was highlighted by a study indicating that 30% of Chinese COVID-19 patients died without receiving ventilator support [[2](https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0252384#pone.0252384.ref002)]. Furthermore, strict preventive measures, social isolation, and pandemic distress could lead to the activation of psychological defensive behaviors in patients where they underestimate their symptoms and do not seek immediate medical assistance [[18](https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0252384#pone.0252384.ref018)]. This optimistic bias could be fatal if a patient’s condition suddenly worsens towards a critical stage. The disease has an unpredictable trajectory where the condition of some patients suddenly becomes critical [[3](https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0252384#pone.0252384.ref003)], surprising even the most skilled physicians; this hampers physicians’ performance by limiting their action time window.

Moreover, rapid isolation of patients with high mortality risks is required since these patients carry significantly more viral loads even before their condition becomes critical [[19](https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0252384#pone.0252384.ref019)]. Similar to an early warning system, our models could be a step in alleviating these problems by providing unbiased, rapid prognosis prediction to support resource allocation and decision making.We developed three predictive models using invasive features, non-invasive features, and both. Our joint model provides rapid, accurate predictions using features that are routinely collected upon patient admission, making it implementable even in conditions where imaging or sophisticated laboratory equipment is unavailable.

Our results revealed that non-invasive features displayed an overall good prediction capacity compared with the invasive and joint model ([Fig 3A](https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0252384#pone-0252384-g003)). The distribution of predictive information was dispersed among invasive features and they were better predictors for patient expirations in near future. In contrast, several non-invasive features had significant information concentrations and these features.

**EXISTING SYSTEM:**

Our society is in the era of unbelievable attempts to struggle upon the spread of this life-threatening condition in terms of infrastructure, finance, business, manufacturing, and several other resources. Artificial Intelligence (AI) researchers strengthen their proficiency in developing mathematical paradigms for investigating this pandemic using nationwide distributed data.AI can in principle be used to track and to predict how the COVID-19 disease will spread over time and space. In fact, an AI-based model of HealthMap, at Boston Children’s Hospital (USA), sounded one of the first alarms on 30 December 2019, around 30 minutes earlier than a scientist at the Program for Monitoring Emerging Diseases (PMED) issued an alert (see the discussion in Naudé 2020). For the further tracking and prediction of how COVID-19 will spread, however, AI has so far not been very useful. This is for a number of reasons. This article intends to apply the machine learning models simultaneously with the forecast of expected reachability of the COVID-19 over the nations by using the real-time data from the Johns Hopkins dashboard

The system is unable to accurately predict the deterioration risk on a test set of new patients. It achieves an area under the receiver operating characteristic curve (AUC) of 0.786 (95% CI: 0.745–0.830), and an area under the precision-recall curve (PR AUC) of 0.517 (95% CI: 0.429–0.600) for prediction of deterioration within 96 h. Additionally, its estimated probability of the temporal risk evolution discriminates effectively between patients, and is well-calibrated. The imaging-based model achieves a comparable AUC to two experienced chest radiologists in a reader study, highlighting the potential of our data-driven approach. In order to verify our system’s performance in a real clinical setting, we silently deployed a preliminary version of it at NYU Langone Health during the first wave of the pandemic, demonstrating that it can produce accurate predictions in real-time. Overall, these results strongly suggest that our system is a viable and valuable tool to inform triage of COVID-19 patients. For reproducibility, we published our code and the trained models at <https://github.com/nyukat/COVID-19_prognosis>.

**PROPOSED SYSTEM:**

The proposed method is based on machine learning framework. Logistic Regression algorithm has been

used for predicting the COVID-19 in someone’s body.

APPLIED LINEAR REGRESSION ON ALGORITHM

TRAINING THE DATA

DATA COLLECTION

PROCESSING DATA

The symptoms we have considered as: pneumonia, diabetes,chronic obstructive pulmonary disease, asthma, hypertension, cardiovascular disease, renal disease, obesity, tobacco taking habit, and contact with other covid-19 positive one.

**Step 1**: Collection of Data Set: COVID-19 is currently one of the most focused areas of research area. Different web resources are there with plenty of data set. Kaggle is basically the world’s largest data science community from where the COVID-19 patient data has been collected and analyzed for our model[2].

**Step 2**: Data Processing: The most important part of machine learning based application is processing of data. To prepare an accurate machine learning application data must be proper processed before feeding it to the system. Python machine learning frame work has been used for this data processing phase in the proposed model. Followings are the python modules used for our application: import pandas as pd importnumpy as np importstatsmodels.api as sm importscipy.stats as st importmatplotlib.pyplot as plt importseaborn as sn fromsklearn.metrics import confusion\_matrix importmatplotlib.mlab as mlab For processing the data, it has to be loaded first: d=pd.read\_csv("d: \Research\COVId-19")

# LITERATURE SURVEY

# Research Questions

The wide spread of the COVID-19 pandemic has resulted in illness and loss of life on a global scale. Research teams have worked on various models to understand the spread of the virus and make data-driven predictions. For the purpose of this SLR, we articulated a research question (RQ) to help focus on the main issue. The motivation and RQs of this study were as follows:Motivation: To identify methods, techniques, models that support the prediction of COVID-19 infections.

RQ1: What factors support the prediction of COVID-19 infections?

RQ2: What methods and techniques are followed in data-driven modeling for predicting COVID-19 infections?

# Prediction Models

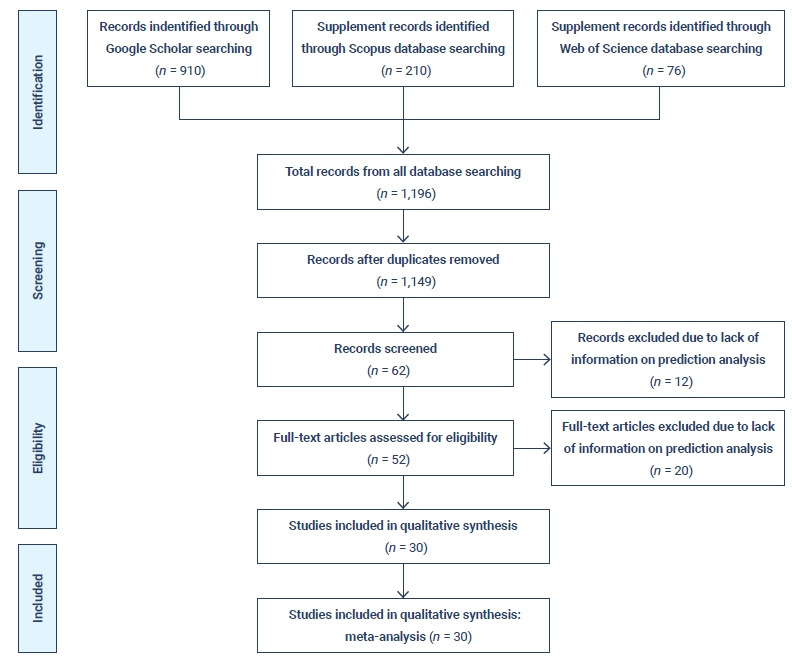
A prediction model is a method of becoming aware of a future scenario beforehand based on available data. Predictive modeling mainly uses statistics to predict outcomes [[18](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b18-j-phrp-2021-0100)]. Forecasting in the COVID-19 pandemic allows medical professionals to better manage facilities and to validate the use of medical and financial resources. It is essential to systematically assess the predictive outcomes of 1 or more prediction models in order to analyze the prediction accuracy of a framework across different study populations, ecosystems, and locations and to assess the need for further developments or improvements of a model [[19](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b19-j-phrp-2021-0100)]. In this paper, we present a systematic review and analysis of these models as presented in the literature.

# Related Works

Coronaviruses are among the main pathogens that predominantly affect the human respiratory system. The focus of the literature review was, therefore, to outline the predominant variables and methodology used in studies related to the spread of the virus. People with prevalent illnesses such as diabetes, hypertension, diabetes, stroke, heart, or kidney failure, as well as elderly people with impaired immune systems, are at an increased risk of infection [[20](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b20-j-phrp-2021-0100)]. Closed areas with low ventilation and airflow may increase the risk of infection. The spread of the virus is believed to occur through respiratory droplets from coughing and sneezing, as with other respiratory viruses, including influenza virus and rhinoviruses. Aerosol transmission is also possible in case of protracted exposure to elevated aerosol concentrations in closed spaces [[21](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b21-j-phrp-2021-0100)].

Several reports have defined a series of variables in terms of quarantine facilities, laboratory testing facilities, and healthcare capability, contributing to state preparedness to fight the pandemic. The most important and successful of these factors must be explored as an urgent solution to the pandemic. The availability of open data sets corresponding to different variables helps to accelerate studies and forge cooperation [[22](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b22-j-phrp-2021-0100)]. Environmental factors, such as pollution and basic sanitation, were considered in some studies. Several studies have also taken into consideration deaths due to COVID-19 and other demographic information [[23](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b23-j-phrp-2021-0100),[24](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b24-j-phrp-2021-0100)]. Other studies and theories have pointed to comorbidities as a key factor in the number of COVID-19 cases [[25](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b25-j-phrp-2021-0100),[26](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b26-j-phrp-2021-0100)]. Without considering comorbidities, fatalities may be mistakenly interpreted as exclusively COVID-19 deaths. Researchers from many universities in the USA have successfully predicted COVID-19 deaths. One such study was conducted at Columbia University and the CDC (2020), in which “death” was used as an exponential function and a social distance parameter prediction was made using a susceptible-exposed-infectious-removed (SEIR) meta-population model.

Since the very beginning of the COVID-19 pandemic, numerous researchers have attempted to construct statistical models of the COVID-19 pandemic, as can be seen from a primary review of existing models. There are several differences in scope, assumptions, forecasts, the effects of interventions, and their impact on health services [[27](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/#b27-j-phrp-2021-0100)]. A PRISMA flow diagram based on the identification of studies from various databases, screening, and the eligibility and inclusion criteria is presented in [Figure 4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/figure/f4-j-phrp-2021-0100/).

[[](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/figure/f4-j-phrp-2021-0100/)](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8408413/figure/f4-j-phrp-2021-0100/" \t "figure)

# APPLICATIONS:

# Search Strategy

The research question was: “What are the applications of machine learning techniques and their performances in COVID-19 diagnosis using X-ray images?”. The search of the present review was based on the PICO elements, which were as follows**P (Problem/Patient/Population):** Patients' CT scans and Chest X-rays.• **I (Intervention/Indicator):** Machine/deep learning models for diagnosis of Covid-19 patients• **C (Comparison):** Ground truth or reference standards• **O (Outcome):** Performance measurements including accuracy, AUC score, sensitivity, and specificity.

In other words, we were looking for publications that evaluated the performance of any machine learning or deep learning approaches based on inclusion and exclusion criteria. Studies that used other types of medical image modalities (e.g., ultrasound images) were excluded. An electronic search was conducted on PubMed, Google Scholar, Scopus, Embase, arXiv, and medRxiv for finding the relevant literature. Duplicate studies were removed. Studies that were cited within the retrieved papers were reviewed for finding missing studies. For identifying proper journal papers and conference proceedings, investigtitle and abstracts based on inclusion and exclusion criteria independently. Finally, considering the inclusion and exclusion criteria, investigators identified the eligible publications in tindependently.

### Inclusion Criteria

The following inclusion criteria were used in the selection of the articles: (1) Studies that applied machine learning or deep learning algorithms, (2) Studies that evaluated the measurement of model outcomes in comparison with ground truth or gold standards, and (3) Studies that used algorithms to analyze radiographic images (CT scan, Chest X-ray, etc.).

### Exclusion Criteria

The following studies were excluded: (1) Studies that used any machine learning or deep learning approaches for problems not directly related to the COVID-19 imaging, (2) Studies that used other artificial intelligence techniques or classic computer vision approaches, (3) Studies that did not provide a clear explanation of the machine learning or deep learning model that was used to solve their problem, and (4) Review studies. The latter were excluded as we did not aim to review the data on an original level without any second-hand interpretations (summation, inferences, etc.).

# SUMMARY

The pandemic of COVID-19 has affected the entire globe. It has spread in more than 85 countries as of Apr. 2020. Scientists have made every effort to find solutions to it; according to claims by the United States and India, some vaccines have been made that are being trialed. The use of computers by scientists for early prediction has been widespread. A lot of research is taking place using ML to combat COVID-19. This chapter can be used by different researchers to learn how ML can be employed to forecast not only this situation but also other cases. The chapter specifically used the ARIMA method of time to forecast the stability and growth of COVID-19. Many countries have seen high totals of deaths owing to COVID-19. It is believed that the performance of the model can be improved or the model can give more accurate data if more datasets are available. The model gives results on the basis of data developed by information given by health agencies. Thus, forecasting may not be 100% accurate, but it can surely be used as a corrective measure. For future work further enhancement can be done by combining new factors and algorithms with ARIMA to get more accurate results.

The COVID-19 pandemic outbreak has devastated the whole world and lead to a state of worldwide health emergency. Several efforts have been performed to combat this pandemic. In this study, we aimed to explore the impact of vital signs, chronic disease, preliminary clinical data, and demographic features to predict the mortality and survival of the COVID-19 patients using supervised machine learning algorithms. Due to the reduced mortality risk of the COVID-19 cases, the dataset suffers from data imbalance. SMOTE technique was used to alleviate the data imbalance. The results showed that random forest outperformed the other models using 10-fold cross-validation. Grid search technique was applied for parameter optimization. The study achieved the accuracy of 0.952 and AUC of 0.99. Despite the significant outcome achieved from this proposed model, there is still a need for improvement. The models need to be validated using multiple datasets. Furthermore, in the future, we will incorporate and explore the impact of other clinical features and laboratory results that were identified as significant in the previous studies.

# SOFTWARE REQUIREMENT SPECIFICATION

SOFTWARE REQUIREMENTS:

Operating system : Windows 10.

Coding Language : Python

Editor : JUPYTER

HARDWARE REQUIREMENTS :

System : Intel i 5Core.11thgen

Hard Disk : 80GB.

Monitor : 15”LED

Input Devices : Keyboard, Mouse

Ram : 8G

## SYSTEM DESIGN

COVID- 19 DATASET

PREPROCESSING

ANALYSIS

DATA SPLITTING

TESTING

APPLYING ML ALGORITHMS

OUTPUT:

CLASSIFYING COVID19 CONFIRMED CASES

# ****Analysing the Outbreak of COVID 19 using Machine Learning****

We need a strong model that predicts how the virus could spread across different states and regions. The goal of this task is to build a model that predicts the spread of the virus in the next 7 days.***The model was built on a***[***test dataset***](https://bit.ly/2yhHrCm)***updated till April,’20. But you can access the source to these datasets at the ‘John Hopkins University Coronavirus Resource Centre’ which gets updated on a daily basis, so you can run this model for the date you prefer.***

### Tasks to be performed:

1. Analysing the present condition in India
2. Is this trend similar to Italy/South Korea/ Wuhan
3. Exploring the world wide data
4. Forecasting the worldwide COVID-19 cases using Prophet

Before we begin with the model, let’s first import the libraries that we need. Consider this a **Step 0** if you may.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | # importing the required libraries  **import** pandas as pd  # Visualisation libraries  **import** matplotlib.pyplot as plt  **%**matplotlib inline  **import** seaborn as sns  **import** plotly.express as px  **import** plotly.graph\_objects as go  **import** folium  **from** folium **import** plugins  # Manipulating the default plot size  plt.rcParams['figure.figsize'] **=** 10, 12  # Disable warnings  **import** warnings  warnings.filterwarnings('ignore') |

1. In here we import a few important libraries that we shall use throughout the model. PANDAS is an extremely fast and flexible data analysis and manipulation tool and allows you to allow you to store and manipulate tabular data. We also import visualisation libraries such as MATPLOTLIB, SEABORN and ***PLOTLY***. And finally, we determine the default plot size and disable warnings in our module.

### ****Part 1: Analysing the present condition in India****

So, how did it actually start in India?

The first **COVID-19**case was reported on 30th January 2020 when a student arrived in Kerala, India from Wuhan, China. Just in the next 2 days, Kerela reported 2 more cases. For almost a month, no new cases were reported in India, however, on 2nd March 2020, five new cases of coronavirus were reported in Kerala again and since then the cases have only been rising.

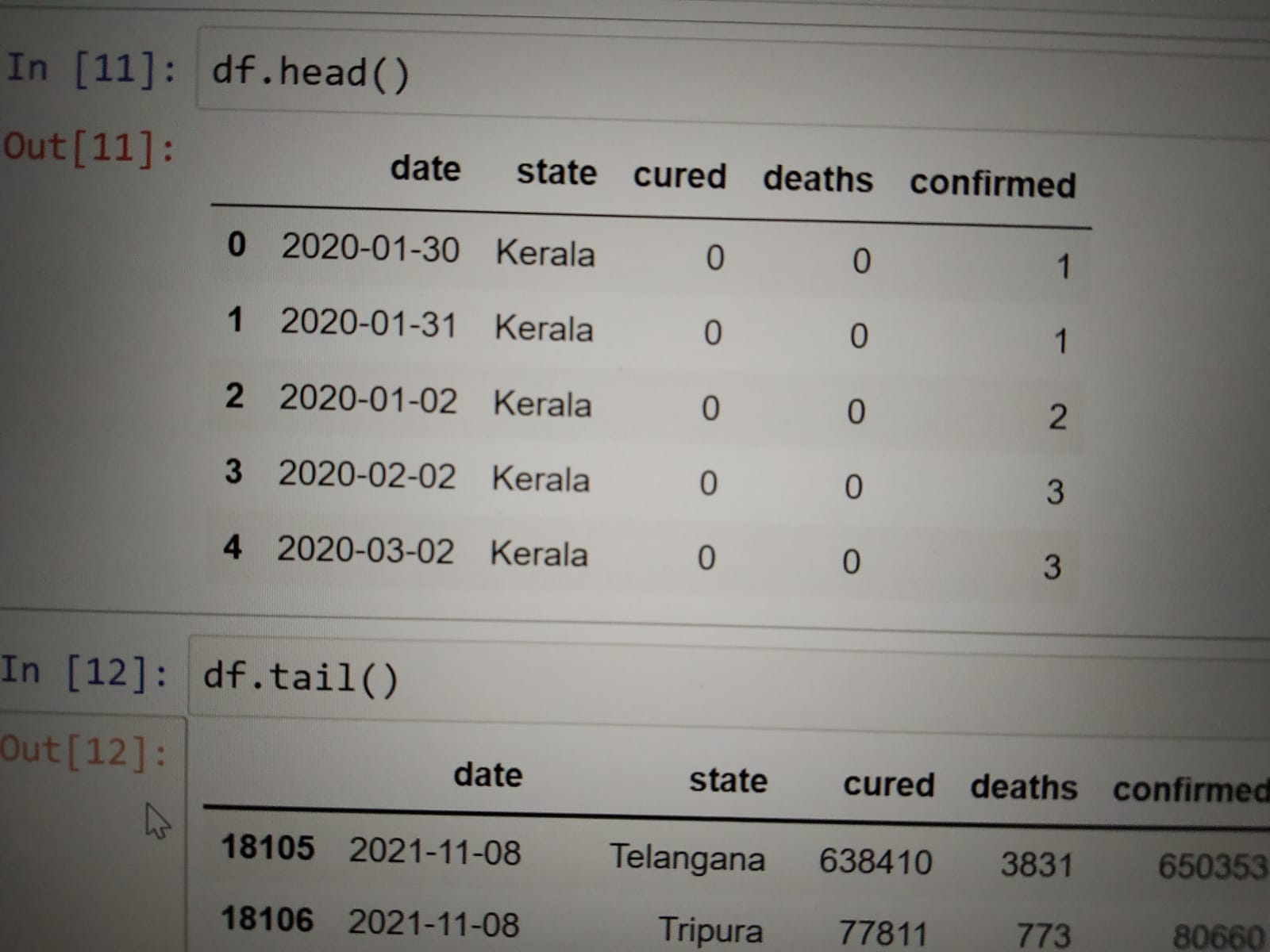
### ****Reading the Datasets****

First, we’re going to start out by reading our datasets by creating a data frame using Pandas.

#READING DATASETS

df=pd.read\_csv(‘covid\_19\_india.csv’,parse\_dates=[‘Date’],dayfirst=True)

df.head()



### ****1.2 Analysing COVID19 Cases in India****

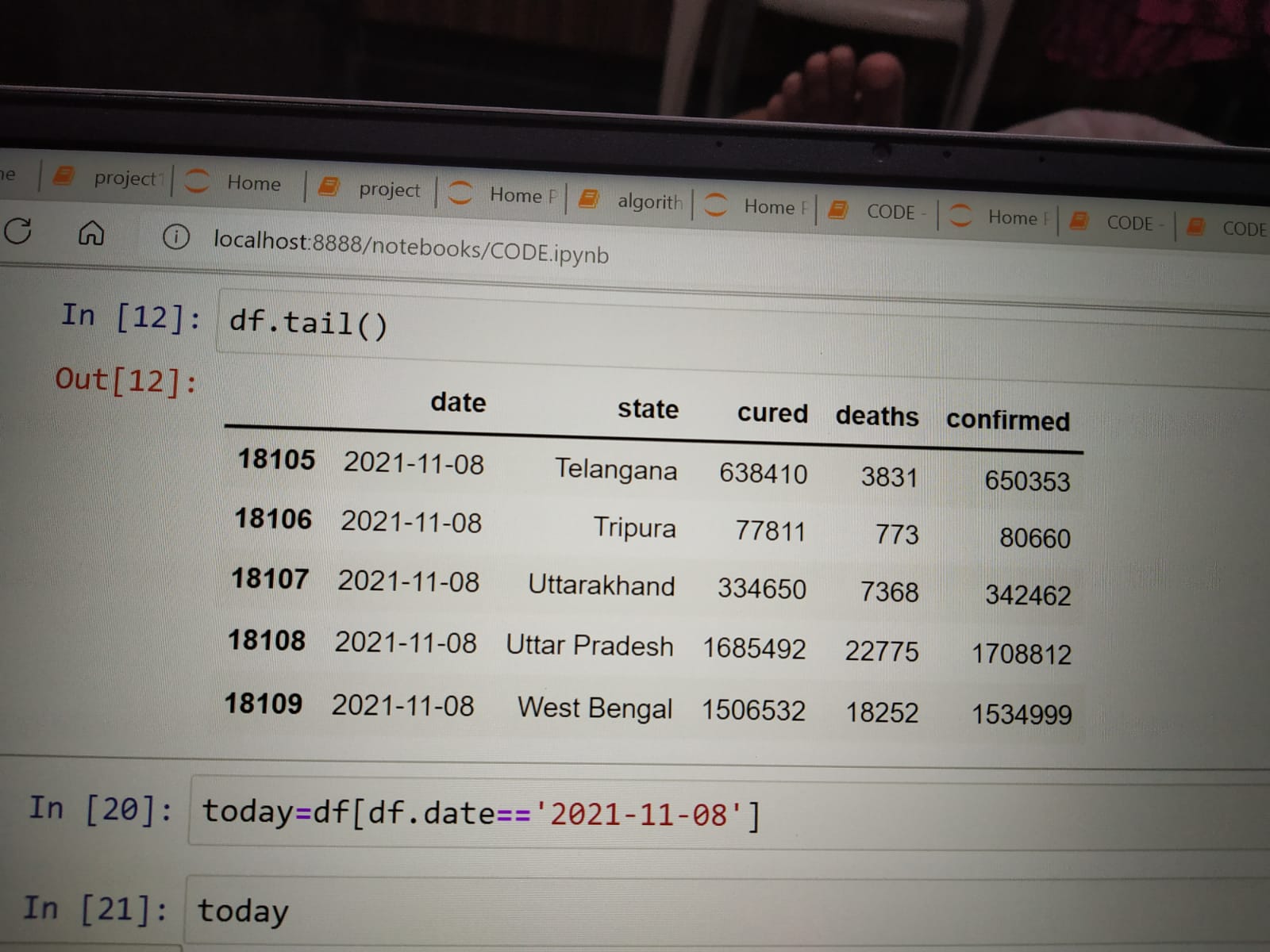
So, here we’re going to play around with the data frame

df=df[[‘Date’,’State/UnionTerritory’,’Cured’,’Deaths’,’Confirmed’]]

df.columns=[‘date’,’state’,’cured’,’deaths’,’confirmed’]

df.head()

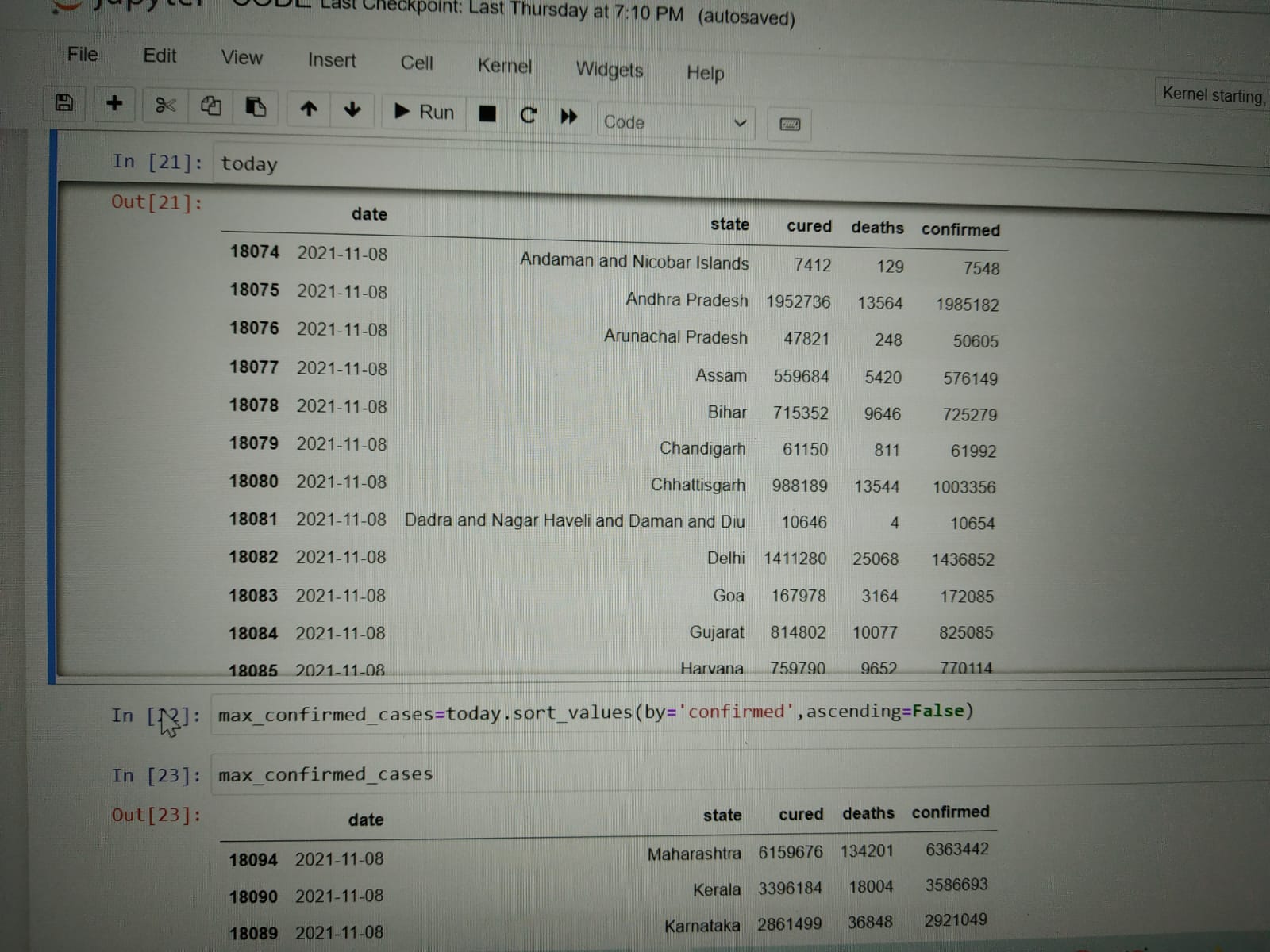
df.tail()



ANALSYING THE CASES BASED ON LATEST DATE:

today=df[df.date==’2020-08-11’]

today



By observing this states have confirmed highest number of cases in india

Max\_death\_cases=today\_sort\_values(by=”deaths”,ascending=False)

Max\_death\_cases

Top\_states\_death=max\_death\_cases[0:5]

Sns.set(rc={‘figure.figsize’:(15,10)})

Sns.barplot(x=”state”,y=”deaths”,data=top\_states\_death,hue=”state”)

Plt.show()

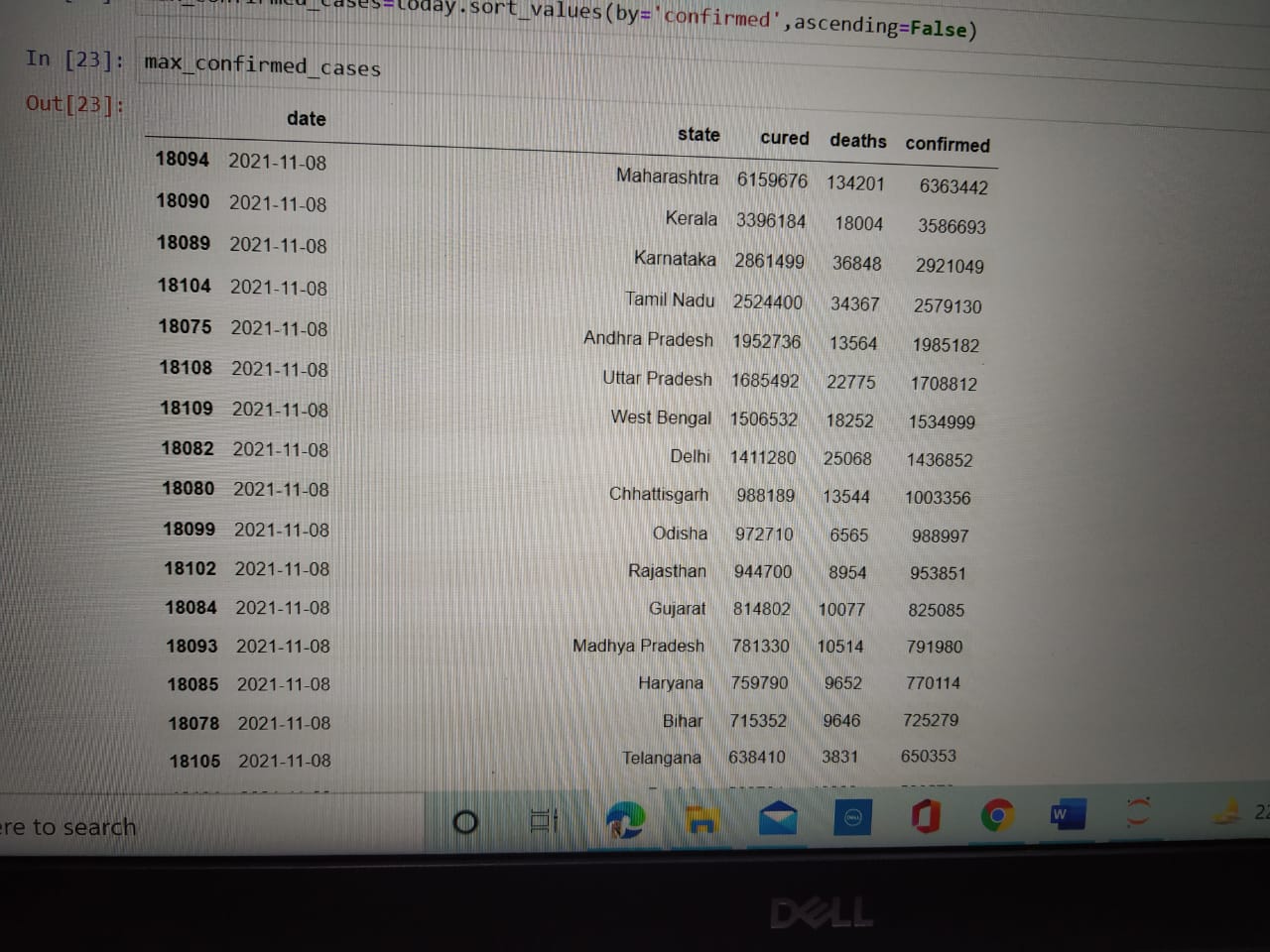
Max\_cured\_cases=today.sort\_values(by=”cured”,ascending=False)

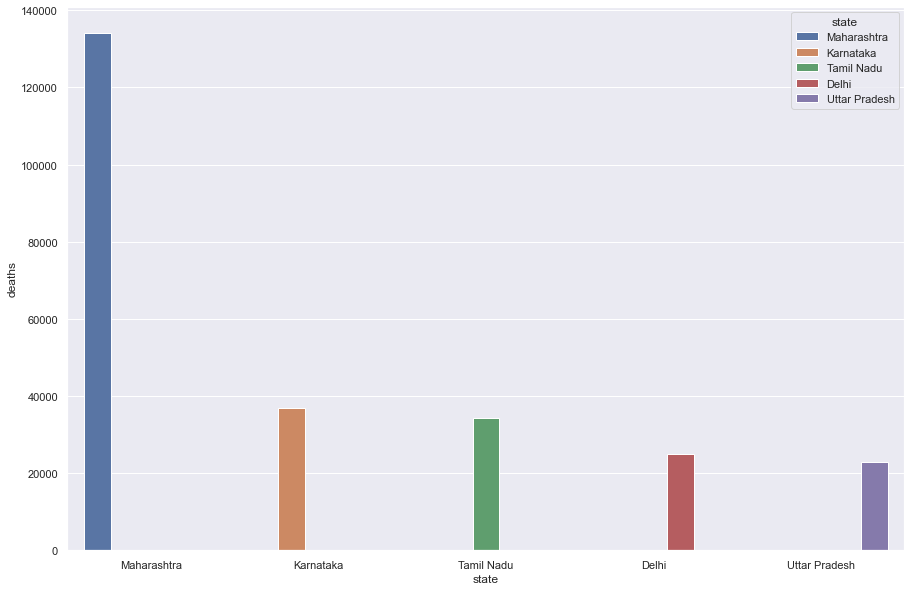
Max\_cured\_cases

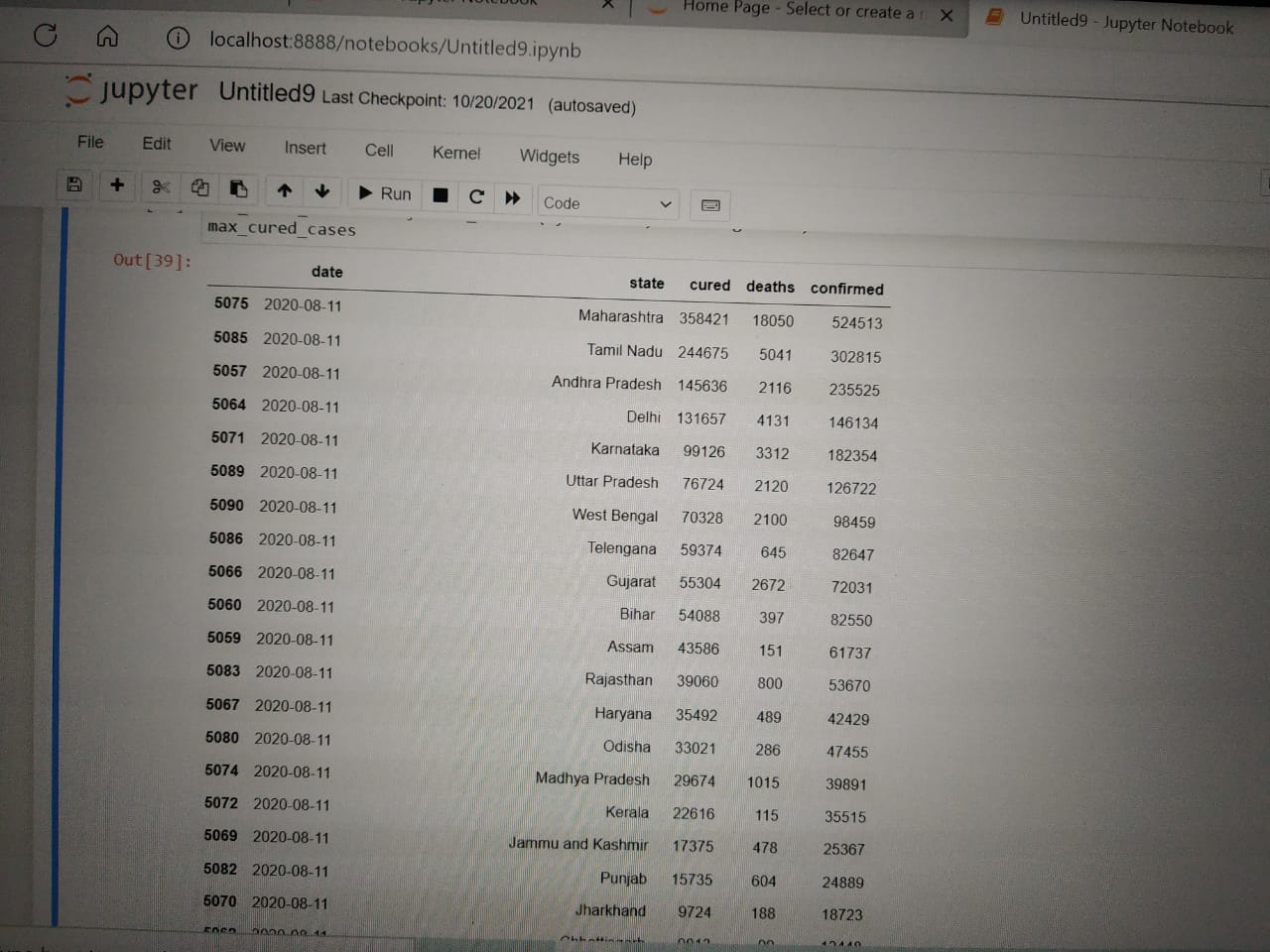
Top\_states\_cured=max\_cured\_cases[0:6]

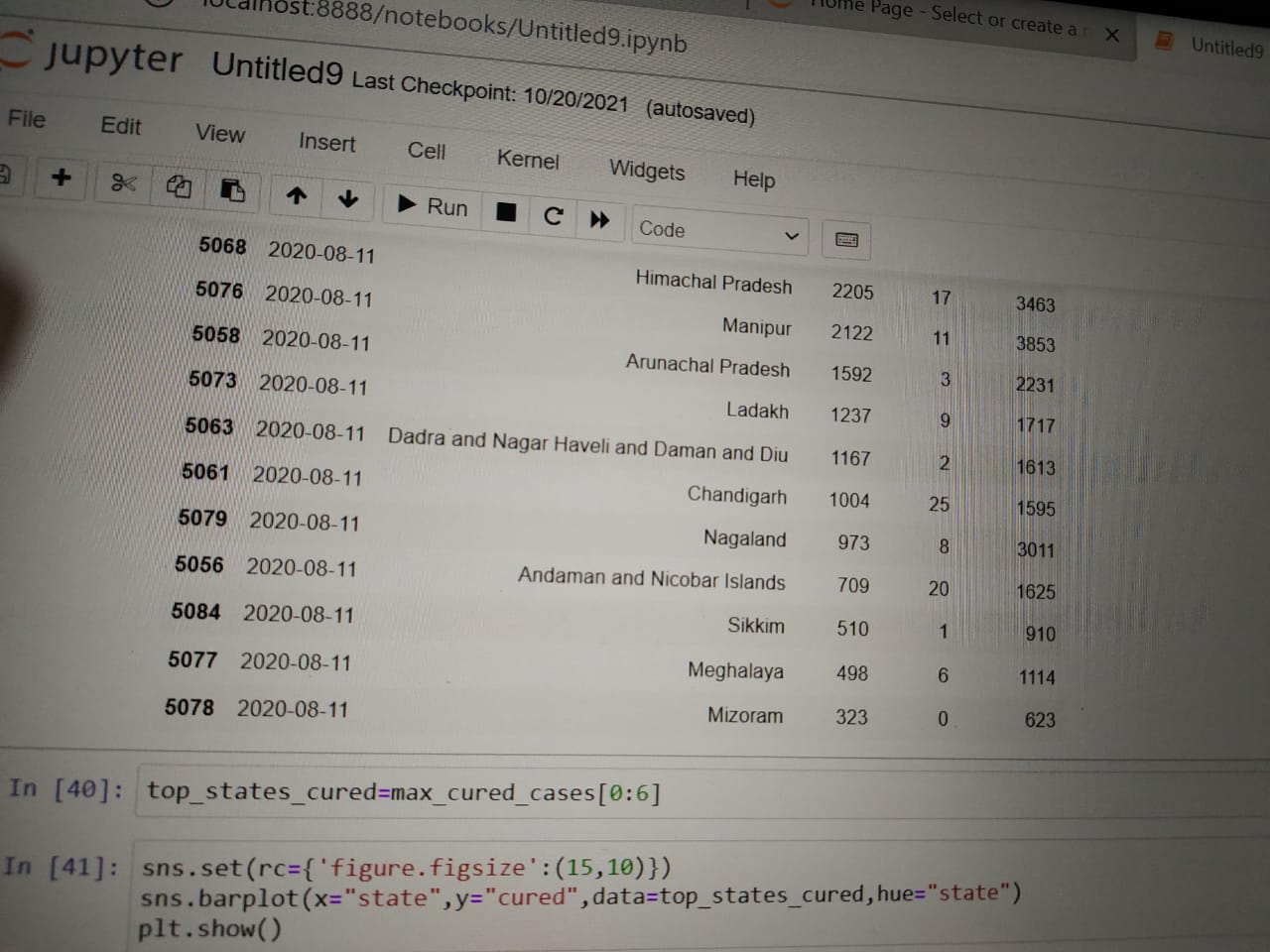
Sns.set(rc={‘figure.figsize’:(15,10)})

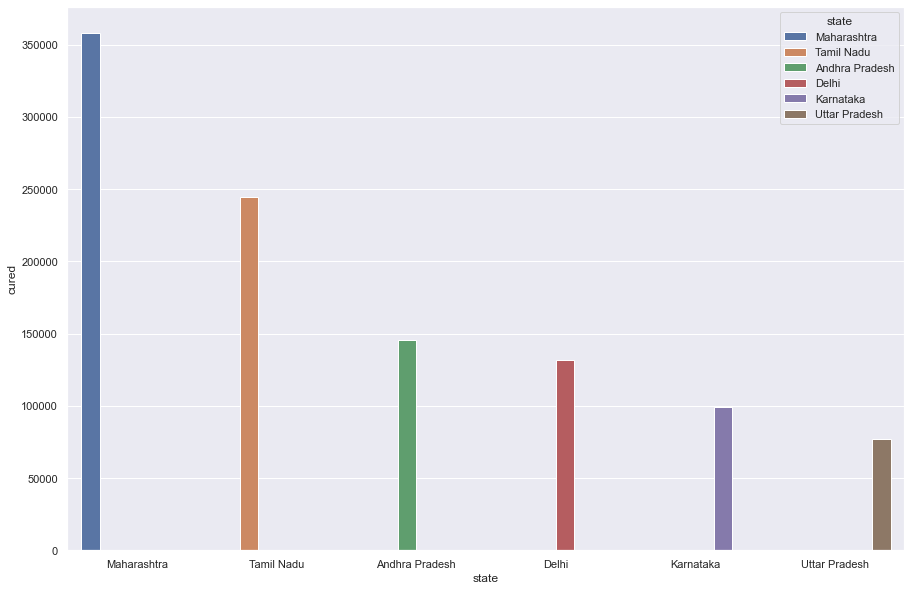
Sns.barplot(x=”state”,y=”cured”,data=top\_states\_cured,hue=”state”)











Max\_confirmed\_cases=today.sort\_values(by=”confirmed”,ascending=False)

Max\_confirmed\_cases

Top\_states\_confirmed=max\_confirmed\_cases[0:5]

Sns.set(rc={‘figure.figsize’:(15,10)})

Sns.barplot(x=”state”,y=”confirmed”,data=top\_states\_confirmed,hue=”state”)

Plt.show()

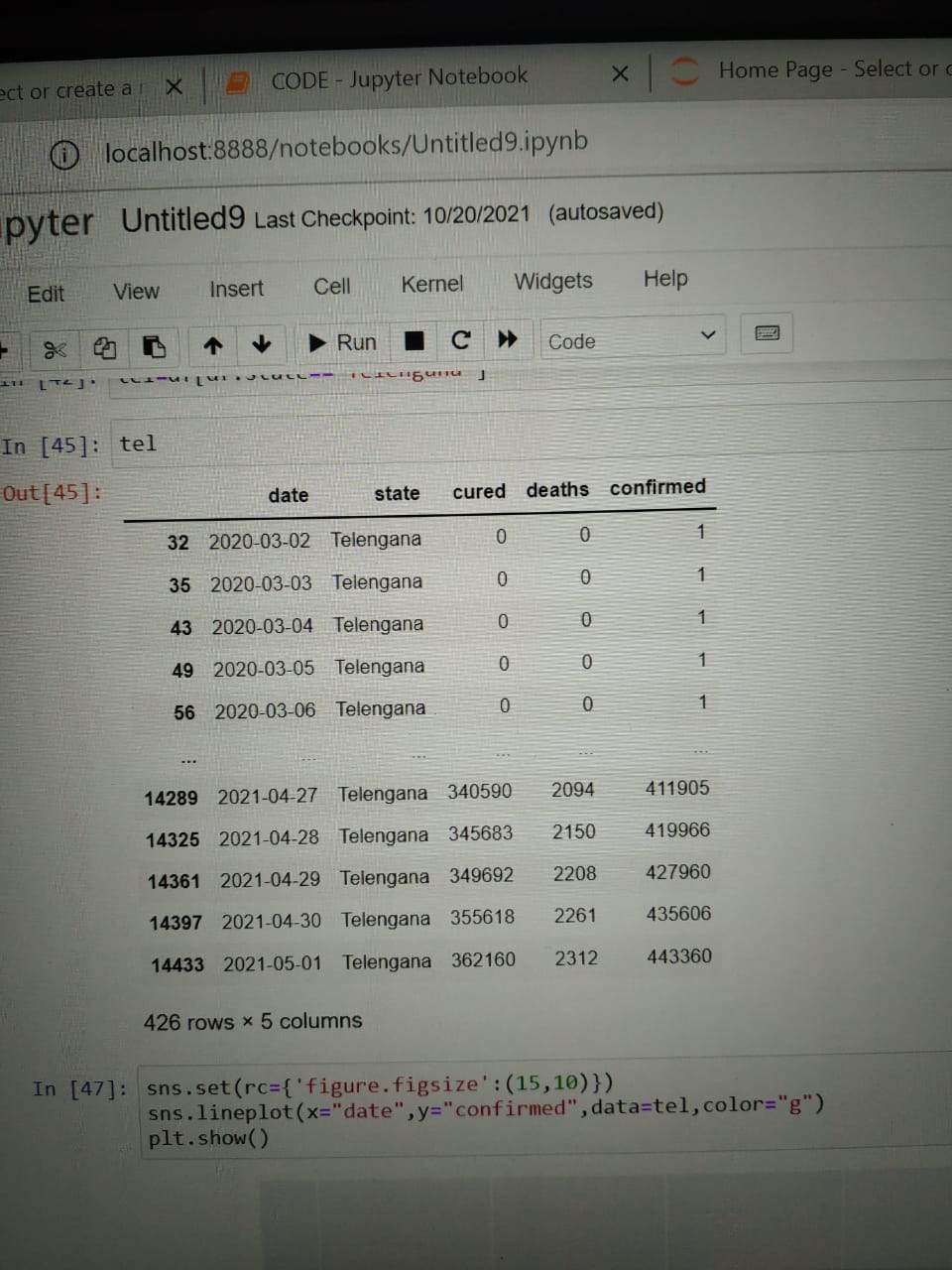
tel=df[df.state==’Telangana’]

tel

sns.set(rc={‘figure.figsize’:(15,10)})

sns.lineplot(x=”date”,y=”confirmed”,data=tel,color=”g”)

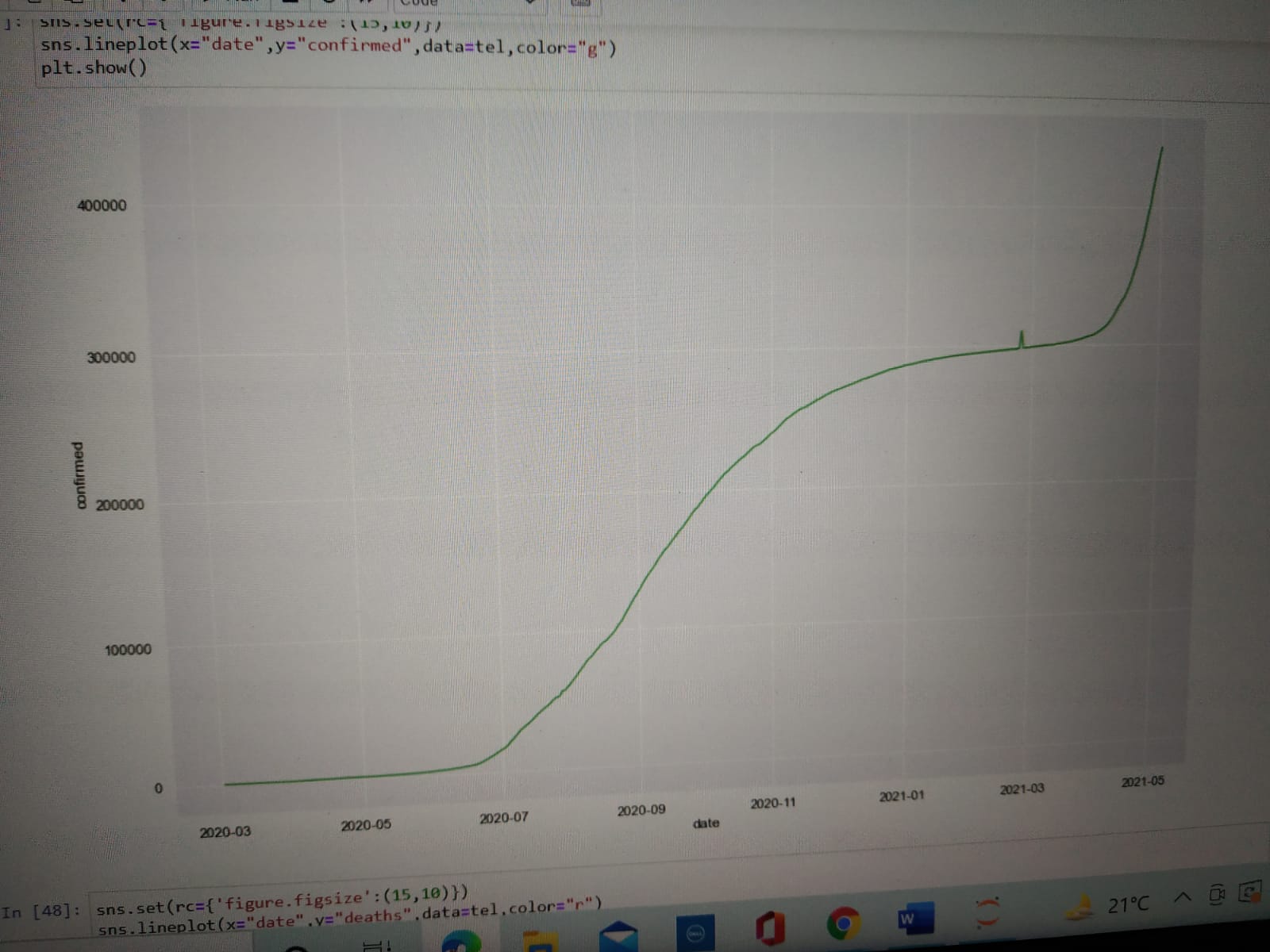
plt.show()

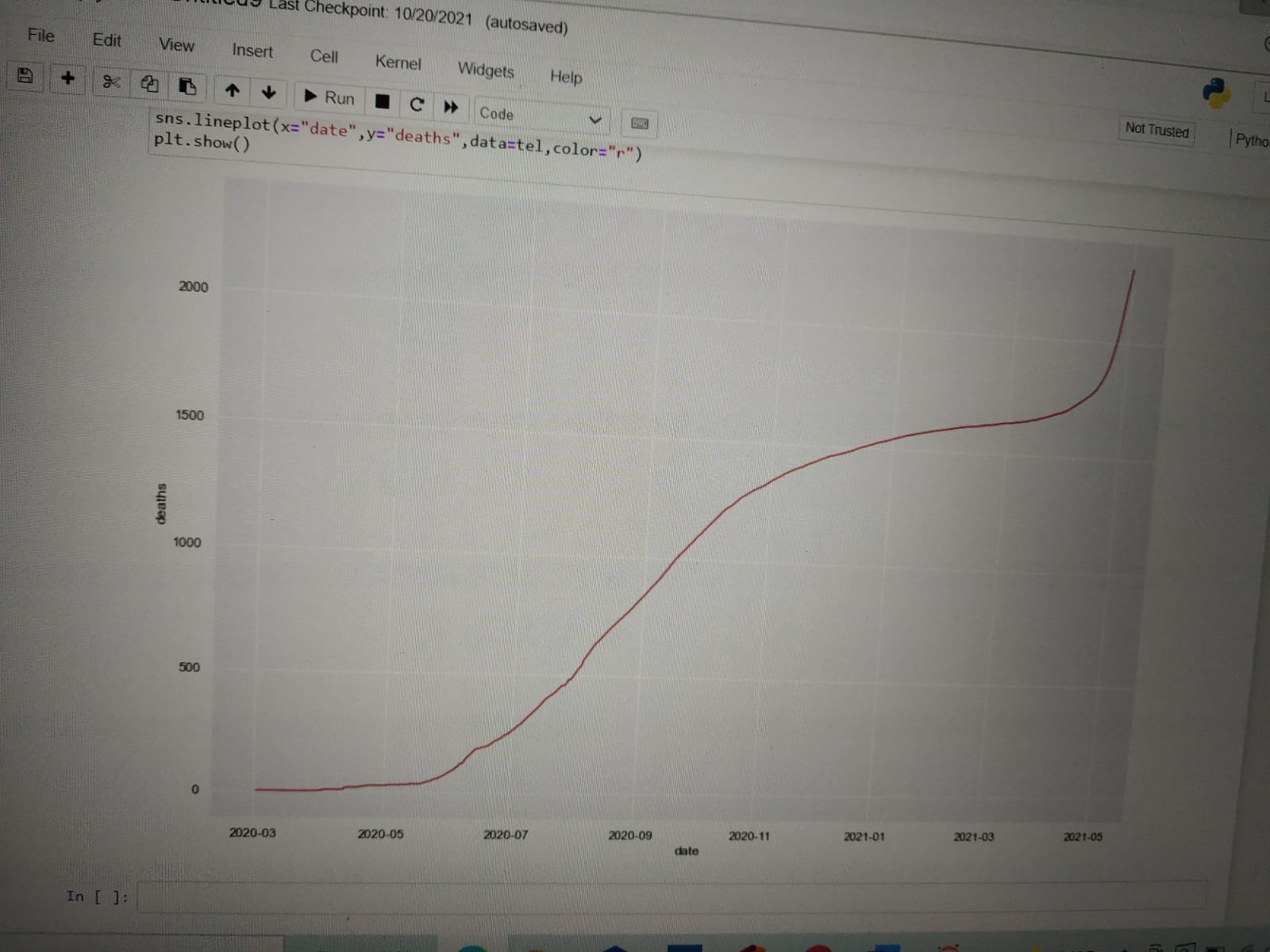


Sns.set(rc={‘figure.figsize’:(15,10)})

Sns.lineplot(x=”date”,y=”deaths”,data=tel,color=”r”)

Plt.show()





### ****Forecasting Total Number of Cases Worldwide****

In this segment, we’re going to generate a week ahead forecast of confirmed cases of COVID-19 using***Prophet***, with specific prediction intervals by creating a base model both with and without tweaking of seasonality-related parameters and additional regressorsProphet is open source software released by Facebook’s Core Data Science team. It is available for download on CRAN and PyPI.

We use Prophet, a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

* **Accurate and fast:**Prophet is used in many applications across Facebook for producing reliable forecasts for planning and goal setting. Facebook finds it to perform better than any other approach in the majority of cases. It fit models in [Stan](https://mc-stan.org/), so that you get forecasts in just a few seconds.
* **Fully automatic:**Get a reasonable forecast on messy data with no manual effort. Prophet is robust to outliers, missing data, and dramatic changes in your time series.
* **Tunable forecasts:**The Prophet procedure includes many possibilities for users to tweak and adjust forecasts. You can use human-interpretable parameters to improve your forecast by adding your domain knowledge
* **Available in R or Python:**Facebook has implemented the Prophet procedure in R and Python. Both of them share the same underlying Stan code for fitting. You can use whatever language you’re comfortable with to get forecasts.

|  |  |
| --- | --- |
| 1  2  3  4 | **from** fbprophet **import** Prophet  confirmed **=** df.groupby('Date').sum()['Confirmed'].reset\_index()  deaths **=** df.groupby('Date').sum()['Deaths'].reset\_index()  recovered **=** df.groupby('Date').sum()['Recovered'].reset\_index() |

The input to Prophet is always a data frame with two columns: **ds** and **y**. The **ds (datestamp)** column should be of a format expected by Pandas, ideally YYYY-MM-DD for a date or YYYY-MM-DD HH:MM:SS for a timestamp. The y column must be numeric and represents the measurement we wish to forecast.

|  |  |
| --- | --- |
| 1  2  3  4 | confirmed.columns **=** ['ds','y']  #confirmed['ds'] = confirmed['ds'].dt.date  confirmed['ds'] **=** pd.to\_datetime(confirmed['ds'])  confirmed.tail() |

### ds versus y - covid 19 machine learning - edureka

### ****Forecasting Confirmed COVID-19 Cases Worldwide with Prophet (Base model)****

Generating a week ahead forecast of confirmed cases of COVID-19 using Prophet, with a 95% prediction interval by creating a base model with no tweaking of seasonality-related parameters and additional regressors.

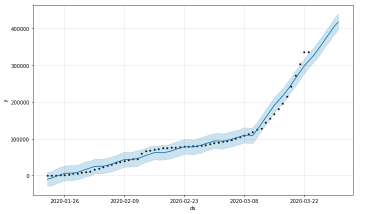
|  |  |
| --- | --- |
| 1  2  3  4 | m **=** Prophet(interval\_width**=**0.95)  m.fit(confirmed)  future **=** m.make\_future\_dataframe(periods**=**7)  future.tail() |

The **predict** method will assign each row in future a predicted value which it names **yhat**. If you pass on historical dates, it will provide an in-sample fit. The **forecast object** here is a new data-frame that includes a column yhat with the forecast, as well as columns for components and uncertainty intervals.

|  |  |
| --- | --- |
| 1  2  3 | #predicting the future with date, and upper and lower limit of y value  forecast **=** m.predict(future)  forecast[['ds', 'yhat', 'yhat\_lower', 'yhat\_upper']].tail() |

You can plot the forecast by calling the Prophet.plot method and passing in your forecast data frame.

|  |  |
| --- | --- |
| 1 | confirmed\_forecast\_plot **=** m.plot(forecast) |



|  |  |
| --- | --- |
| 1 | confirmed\_forecast\_plot **=**m.plot\_components(forecast) |

### prophet forecast by day - covid 19 machine learning - edureka

The novel coronavirus disease 2019 (COVID-19) pandemic caused by the SARS-CoV-2 continues to pose a critical and urgent threat to global health. The outbreak in early December 2019 in the Hubei province of the People’s Republic of China has spread worldwide. As of October 2020, the overall number of patients confirmed to have the disease has exceeded 39,500,000, in >180 countries, though the number of people infected is probably much higher. More than 1,110,000 people have died from COVID-19[1](https://www.nature.com/articles/s41746-020-00372-6#ref-CR1).

This pandemic continues to challenge medical systems worldwide in many aspects, including sharp increases in demands for hospital beds and critical shortages in medical equipment, while many healthcare workers have themselves been infected. Thus, the capacity for immediate clinical decisions and effective usage of healthcare resources is crucial. The most validated diagnosis test for COVID-19, using reverse transcriptase polymerase chain reaction (RT-PCR), has long been in shortage in developing countries

First take a data set from a Kaggle university that it contains all the covid cases with date and time .then process the data and analyse the data how many cases are there and the data can be identified through the analysing and test the data by using some packages that can be imported from python this data can be processed using machine learning algorithms called linear algorithm which is a supervised learning and we can observe the number of cases in states across india by machine learning algorithms.

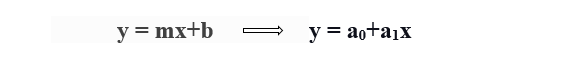
**Linear Regression** is a machine learning algorithm based on **supervised learning**. It performs a **regression task**. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number

of independent variables being used.  
Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.  
In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Linear regression can be further divided into two types of the algorithm:

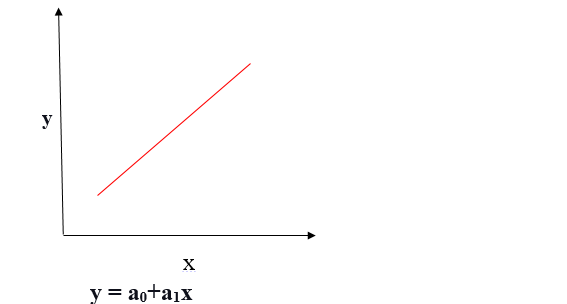
**Simple Linear Regression:**  
If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.

* **Multiple Linear regression:**  
  If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.
* *To calculate best-fit line linear regression uses a traditional slope-intercept form.*



**y= Dependent Variable.  
  
x= Independent Variable.  
  
a0= intercept of the line.  
  
a1 = Linear regression coefficient.**

the dependent variable expands on the Y-axis and the independent variable progress on X-axis, then such a relationship is termed a Positive linear relationship.



# IMPLEMENTATION

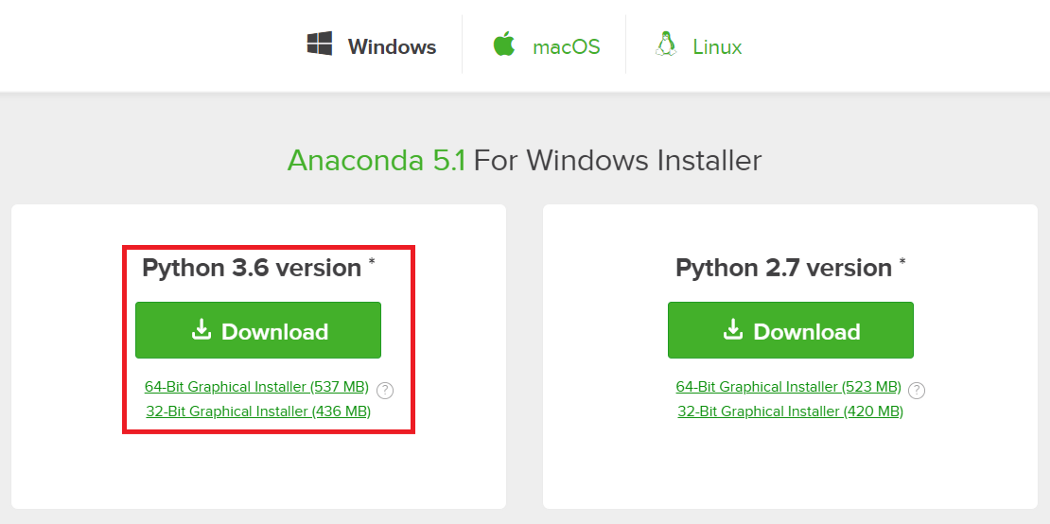
Environmental setup

# Step 1: Download Anaconda

In this step, we will download the Anaconda Python package for your platform.

Anaconda is a free and easy-to-use environment for scientific Python.

* 1.Install Anaconda (Python 3.6 version)



I am using Windows you can choose according to your OS.

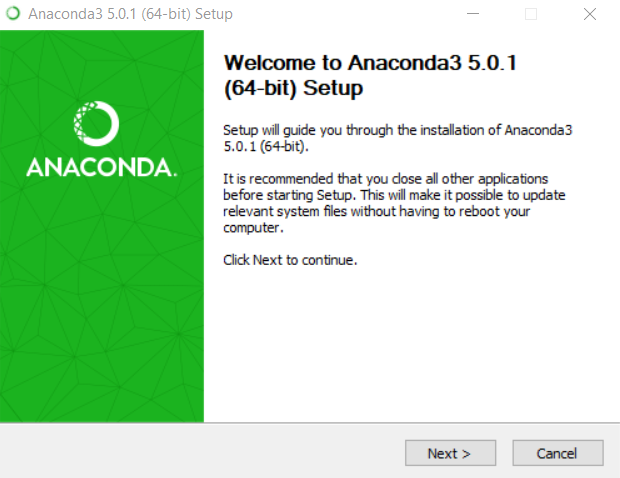
Step 2: Install Anaconda

In this step, we will install the Anaconda Python software on your system.14

Installation is very easy and quick once you download the setup. Open the setup and follow the wizard instructions.

**#Note:** It will automatically install Python and some basic libraries with it.

It might take 5 to 10 minutes or some more time according to your system.



# Step 3: Update Anaconda

Open Anaconda Prompt to type the following command(s). Don’t worry Anaconda Prompt works the same as cmd.

**conda update conda  
conda update –all 16**

# Step 4: Install CUDA Toolkit & cuDNN

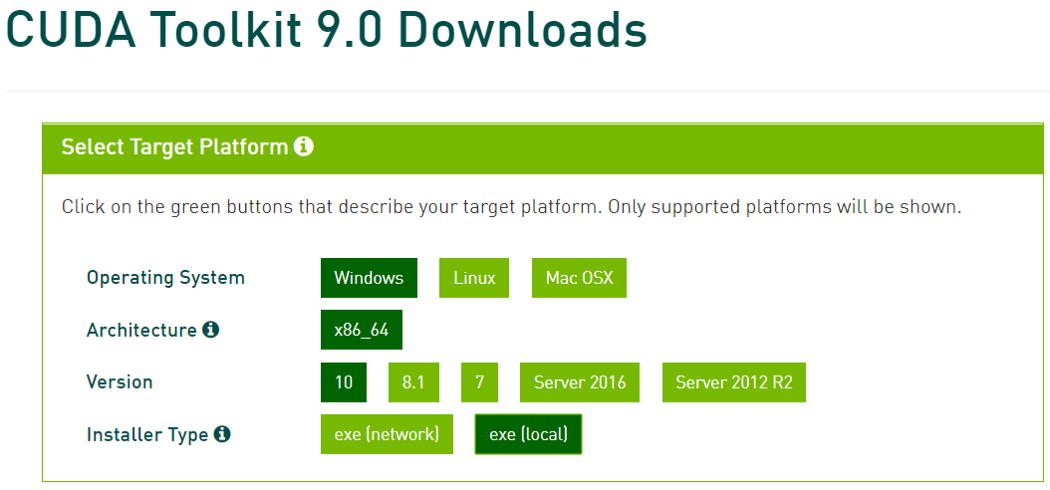
1. Install CUDA Toolkit 9.0 or 8.0 [Download](https://developer.nvidia.com/cuda-downloads)

Choose your version depending on your Operating System and GPU.

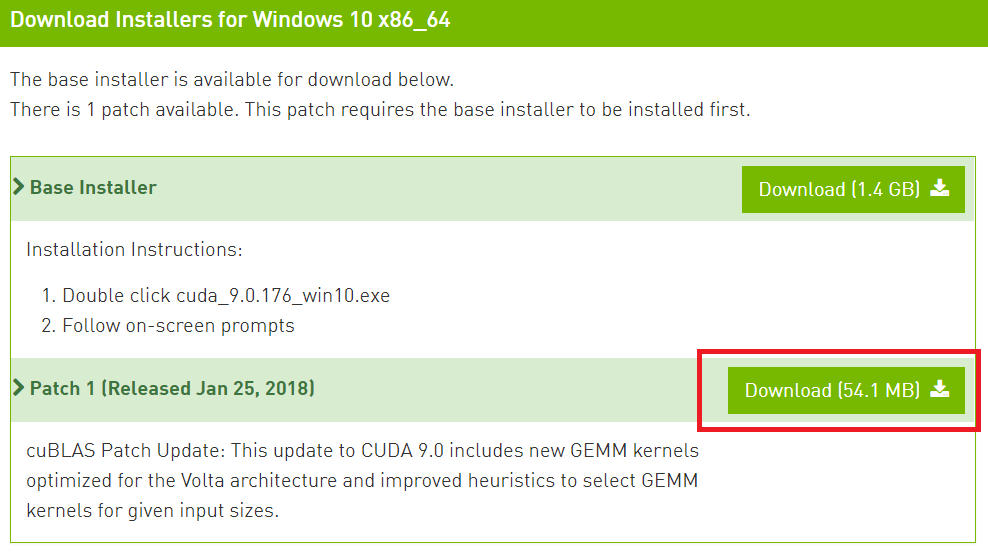
**#Version Support:** Here is a guide to check that if your [version supports](https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html) your Nvidia Graphic Card

For downloading other versions you can follow this link: <https://developer.nvidia.com/cuda-toolkit-archive>

**#Note:** CUDA 9.0 is recommended as TensorFlow is NOT compatible with CUDA Toolkit 9.1 and 9.2 versions. Kindly choose the CUDA version according to your Nvidia GPU version to avoid errors.



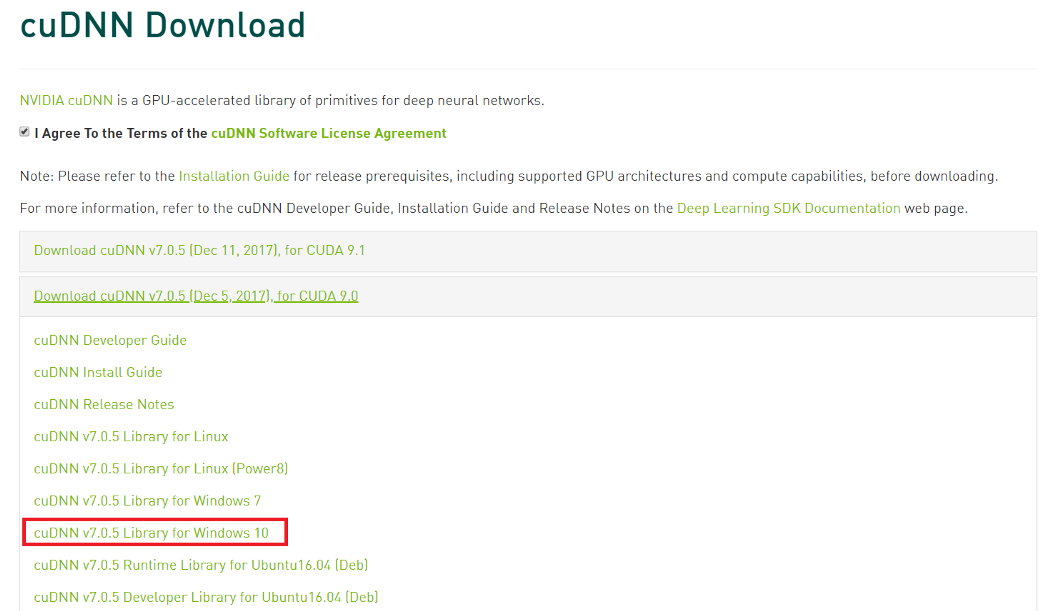
17



**#Note:** People with version 9.0 [Download](https://developer.nvidia.com/cuda-downloads) can also install the given patch in any case of error while proceeding.

2. Download cuDNN [Download](https://developer.nvidia.com/rdp/cudnn-download)

Download the latest version of cuDNN. Choose your version depending on your Operating System and CUDA. Membership registration is required. Don’t worry you can easily create an account using your email.



Put your unzipped folder in C drive as follows:

**C:\cudnn-9.0-windows10-x64-v7**

# Step 5: Add cuDNN into Environment Path

1. Open Run dialogue using (**Win + R)**and run the command **sysdm.cpl**
2. In Window-10 **System Properties**, please select the Tab **Advanced**.
3. Select Environment Variables
4. Add the following path to your Environment.

**C:\cudnn-9.0-windows10-x64-v7\cuda\bin**

Step 6: Create an Anaconda Environment

Here we will create a new anaconda environment for our specific usage so that it will not affect the root of Anaconda. Amazing!! isn’t it?

Open Anaconda Prompt to type the following commands.

1. Create a conda environment named “tensorflow” (you can change the name) by invoking the following command:

**conda create -n tensorflow pip python=3.6**

2. Activate the conda environment by issuing the following command:

**activate tensorflow**  
 (tensorflow)C:> # Your prompt should change

# Step 7: Install Deep Learning Libraries

In this step, we will install Python libraries used for deep learning, specifically: TensorFlow, and Keras.

1. [**TensorFlow**](https://www.tensorflow.org/install/)

TensorFlow is a tool for machine learning. While it contains a wide range of functionality, TensorFlow is mainly designed for deep neural network models.=> For installing TensorFlow, Open Anaconda Prompt to type the following commands.

To install the GPU version of TensorFlow:

C:\> **pip install tensorflow-gpu**

# 20

# PC Hardware Setup

First of all to perform machine learning and deep learning on any dataset, the software/program requires a computer system powerful enough to handle the computing power necessary. So the following is required:

1. **Central Processing Unit (CPU)** — Intel Core i5 6th Generation processor or higher. An AMD equivalent processor will also be optimal.
2. **RAM**— 8 GB minimum, 16 GB or higher is recommended.
3. **Graphics Processing Unit (GPU)** — NVIDIA GeForce GTX 960 or higher. AMD GPUs are not able to perform deep learning regardless. For more information on NVIDIA GPUs for deep learning please visit <https://developer.nvidia.com/cuda-gpus>.
4. **Operating System** — Ubuntu or Microsoft Windows 10. I recommend updating Windows 10 to the latest version before proceeding forward

Module description

In Machine Learning Studio (classic), a module is a building block for creating experiments. Each module encapsulates a specific machine learning algorithm, function, or code library that can act on data in your workspace. The modules are designed to accept connections from other modules, to share and modify data.

The code that runs in each module comes from many sources. These include open source libraries and languages, algorithms developed by Microsoft Research, and tools for working with Azure and other cloud services.

By connecting and configuring modules, you can create a workflow that reads data from external sources, prepares it for analysis, applies machine learning algorithms, and generates results.

When an *experiment* is open in Machine Learning Studio (classic), you can see the complete list of current modules in the navigation pane at left. You drag these building blocks into your experiment, and then connect them to create a complete machine learning workflow, called an experiment.

Sometimes modules are updated to add new functionality, or to remove older code. When this happens, any experiments that you created that use the module continue to run. But the next time you open the experiment, you are prompted to upgrade the module, or to use a different module.

For an example of how to build a complete machine learning experiment, see these tutorials:

* [Develop a predictive solution by using Machine Learning](https://docs.microsoft.com/en-us/azure/machine-learning/classic/tutorial-part1-credit-risk)
* [Create a simple experiment in Machine Learning Studio (classic)](https://docs.microsoft.com/en-us/azure/machine-learning/classic/create-experiment)

## Module categories

To make it easier to find related modules, the machine learning tools in Machine Learning Studio (classic) are grouped by these categories.

### [Data Format conversions](https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-format-conversions)

Use these modules to convert data to one of the formats used by other machine learning tools or formatts

Use these modules to read data and models from cloud data sources, including Hadoop clusters, Azure Table storage, and web URLs. You can also use these modules to write results to storage or to a database.

[Data Transformation](https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-transformation)

Use these modules to prepare data for analysis. You can change data types, flag columns as features or labels, generate features, and scale or normalize data.

[Filter](https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-transformation-filter)

Transform numeric data derived from digital signal processing.

[Learning With Counts](https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-transformation-learning-with-counts)

Use joint probability distributions to build features that compactly describe large datasets.

[Manipulation](https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-transformation-manipulation)

This group provides a variety of tools for data science. For example, you can remove or replace missing values, choose a subset of columns, add a column, or concatenate two datasets.

[Sample and Split](https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-transformation-sample-and-split)

Divide a dataset by criteria or by size, to create training and test sets, or to isolate certain rows.

[Scale and Reduce](https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-transformation-scale-and-reduce)

Transform numerical data.

Software description

Machine learning (ML) is a type of artificial intelligence ([AI](https://searchenterpriseai.techtarget.com/definition/AI-Artificial-Intelligence)) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning [algorithms](https://whatis.techtarget.com/definition/algorithm) use historical data as input to predict new output values.

Machine learning is important because it gives enterprises a view of trends in customer behavior and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies

**Supervised learning:** In this type of machine learning, [data scientists](https://searchenterpriseai.techtarget.com/definition/data-scientist) supply algorithms with labeled training data and define the variables they want the algorithm to assess for correlations. Both the input and the output of the algorithm is specified.

* **Unsupervised learning:** This type of machine learning involves algorithms that train on unlabeled data. The algorithm scans through data sets looking for any meaningful connection. The data that algorithms train on as well as the predictions or recommendations they output are predetermined.
* **Semi-supervised learning:** This approach to machine learning involves a mix of the two preceding types. Data scientists may feed an algorithm mostly labeled [training data](https://searchenterpriseai.techtarget.com/feature/Using-small-data-sets-for-machine-learning-models-sees-growth), but the model is free to explore the data on its own and develop its own understanding of the data set.
* **Reinforcement learning:**Data scientists typically use [reinforcement learning](https://searchenterpriseai.techtarget.com/definition/reinforcement-learning) to teach a machine to complete a multi-step process for which there are clearly defined rules. Data scientists program an algorithm to complete a task and give it positive or negative cues as it works out how to complete a task. But for the most part, the algorithm decides on its own what steps to take along the way.

### How does supervised machine learning work?

Supervised machine learning requires the [data scientist](https://searchbusinessanalytics.techtarget.com/feature/Key-differences-of-a-data-scientist-vs-data-engineer) to train the algorithm with both labeled inputs and desired outputs. Supervised learning algorithms are good for the following tasks:

* **Binary classification:**Dividing data into two categories.
* **Multi-class classification:**Choosing between more than two types of answers.
* **Regression modeling:** Predicting continuous values.
* **Ensembling:** Combining the predictions of multiple machine learning models to produce an accurate prediction.

### Machine Learning Real Examples

Given below are some real examples of ML:

**Example 1:**

If you have used Netflix, then you must know that it recommends you some movies or shows for watching based on what you have watched earlier. Machine Learning is used for this recommendation and to select the data which matches your choice. It uses the earlier data.

**Example 2:**

The second example would be Facebook.

When you upload a photo on Facebook, it can recognize a person in that photo and suggest you, mutual friends. ML is used for these predictions. It uses data like your friend-list, photos available etc. and it makes predictions based on that.

**Example 3:**

The third example is Software, which shows how you will look when you get older. This image processing also uses machine learning.

All these are some examples, that help us to understand, how machine learning is used. ML is similar to AI up to some extent, however, there is a difference between the two. It is related to data mining.

### How does Machine Learning Help us?

It helps through powerful processing.

With the help of machine learning, systems make better decisions, at a high speed and most of the times they are accurate. Using this technique is inexpensive and it can analyze large and complex data

SAMPLE CODE

import pandas as pd

from matplotlib import pyplot as plt

import seaborn as sns

import datetime as dt

import numpy as np

df=pd.read\_csv('covid\_19\_india.csv',parse\_dates=['Date'],dayfirst=True)

df.head()

df=df[['Date','State/UnionTerritory','Cured','Deaths','Confirmed’]]

df.columns=['date','state','cured','deaths','confirmed’]

df.head()

df.tail()

today=df[df.date=='2020-08-11’]

today

max\_death\_cases=today.sort\_values(by="deaths",ascending=False)

max\_death\_cases

top\_states\_confirmed=max\_confirmed\_cases[0:5]

sns.set(rc={'figure.figsize':(15,10)})

sns.barplot(x="state",y="confirmed",data=top\_states\_confirmed,hue="state")

plt.show()

max\_death\_cases=today.sort\_values(by="deaths",ascending=False)

max\_death\_cases

top\_states\_death=max\_death\_cases[0:5]

sns.set(rc={'figure.figsize':(15,10)})

sns.barplot(x="state",y="deaths",data=top\_states\_death,hue="state")

plt.show()

max\_cured\_cases=today.sort\_values(by="cured",ascending=False)

max\_cured\_cases

top\_states\_cured=max\_cured\_cases[0:6]

sns.set(rc={'figure.figsize':(15,10)})

sns.barplot(x="state",y="cured",data=top\_states\_cured,hue="state")

plt.show()

tel=df[df.state=='Telengana’]

tel

sns.set(rc={'figure.figsize':(15,10)})

sns.lineplot(x="date",y="confirmed",data=tel,color="g")

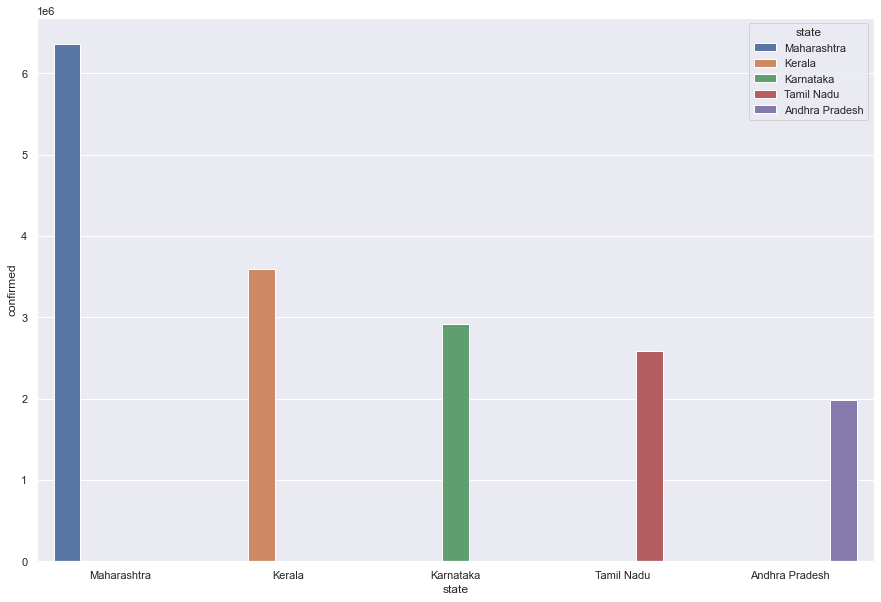
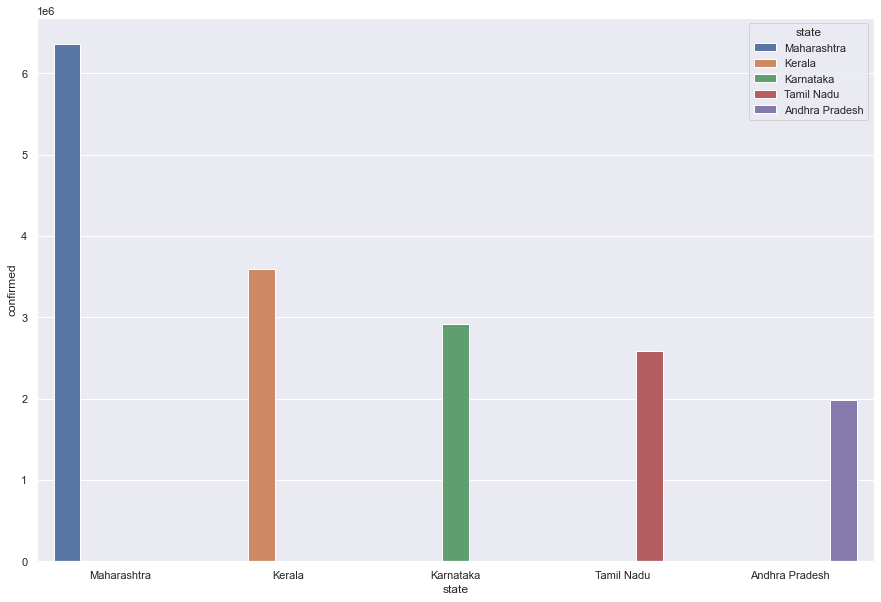
plt.show()

sns.set(rc={'figure.figsize':(15,10)})

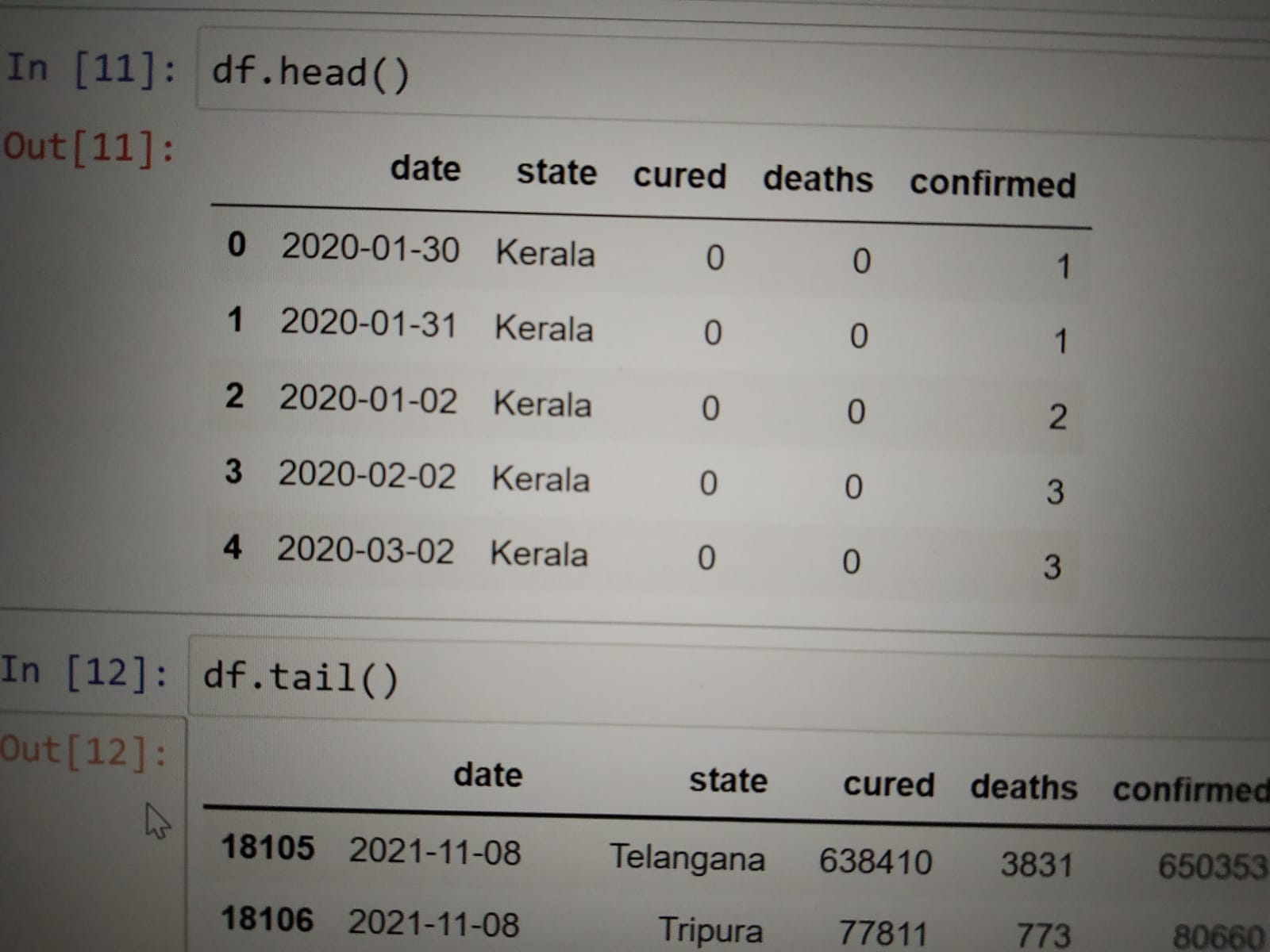
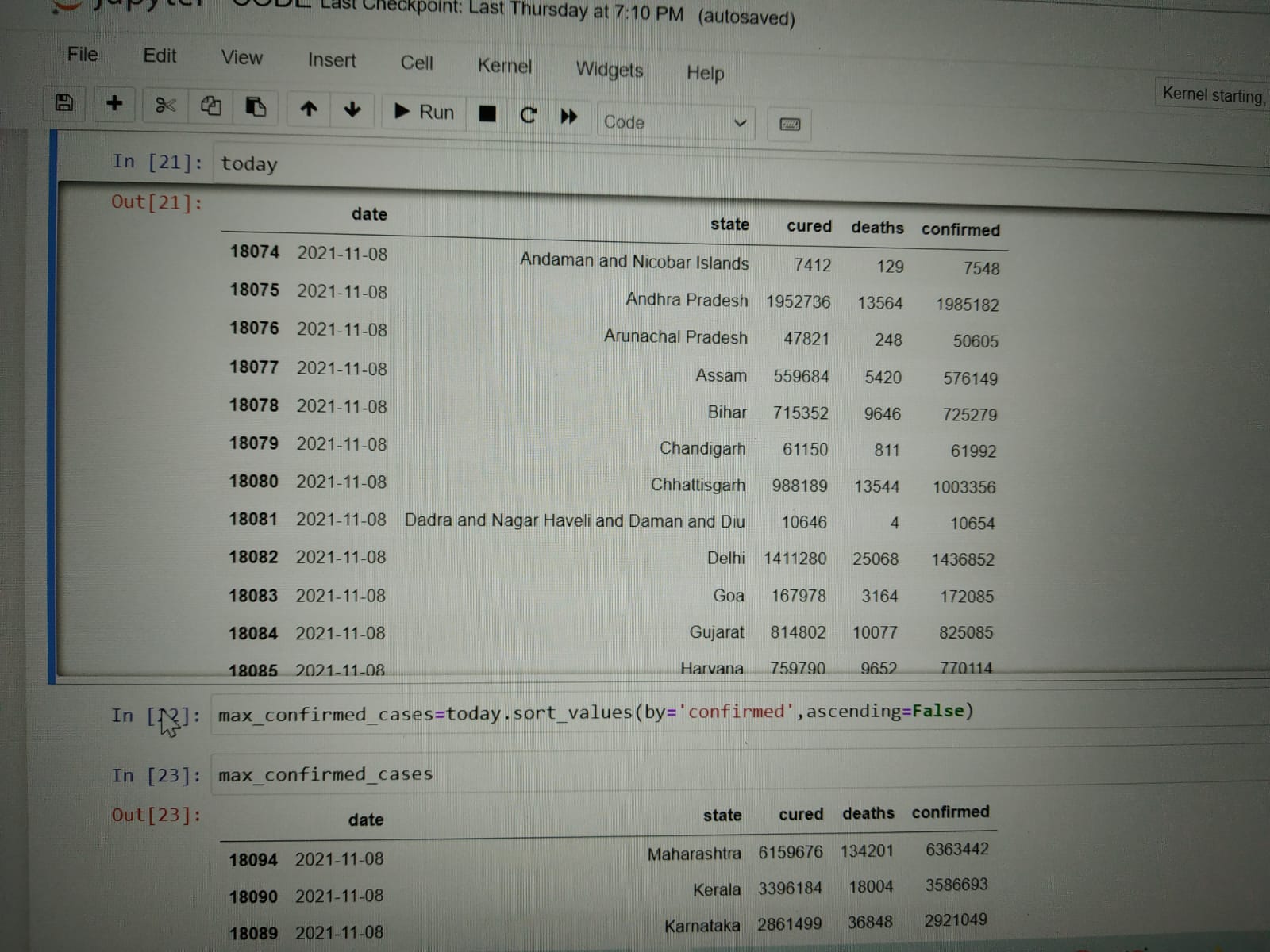
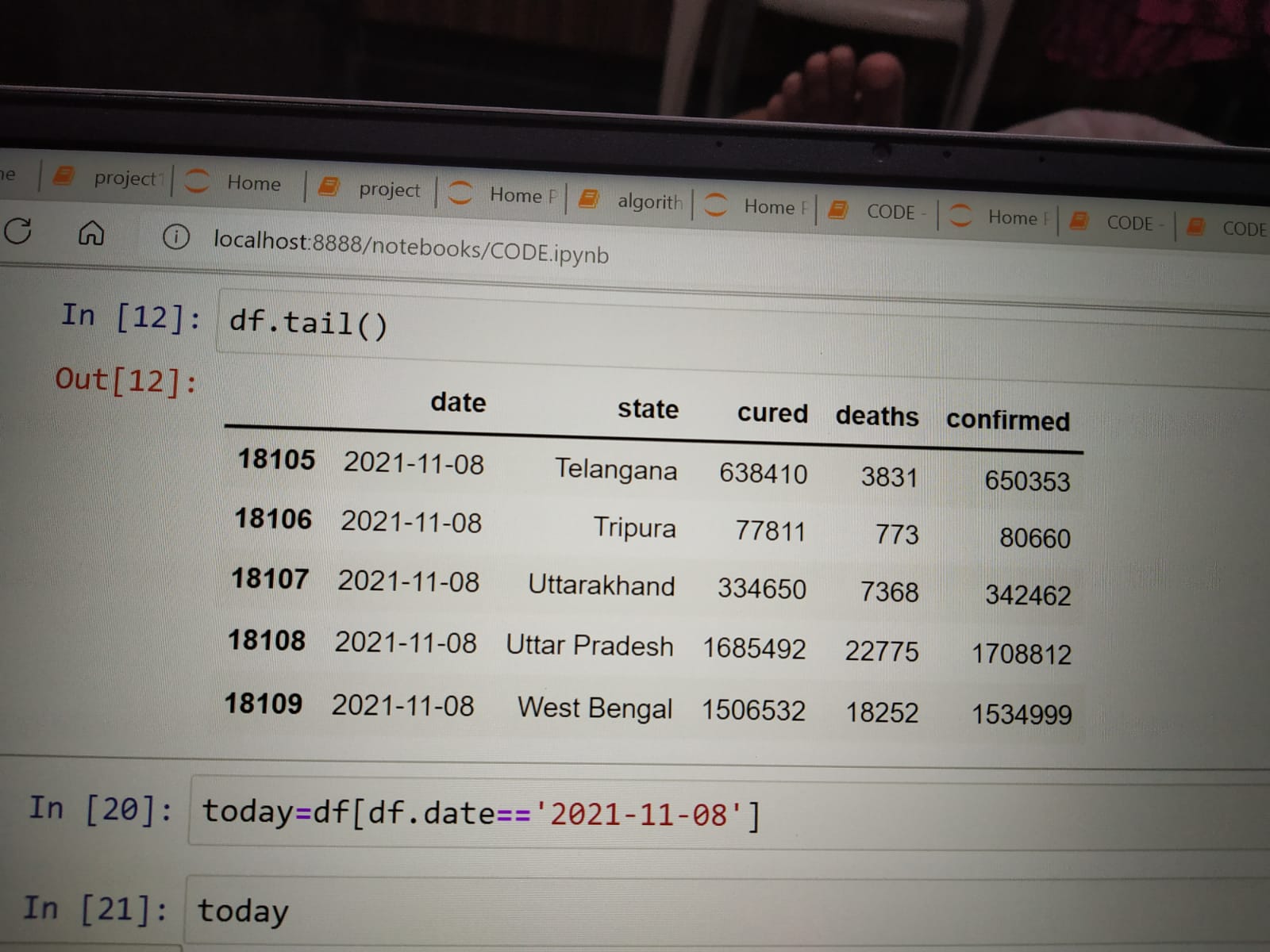
sns.lineplot(x="date",y="deaths",data=tel,color="r")

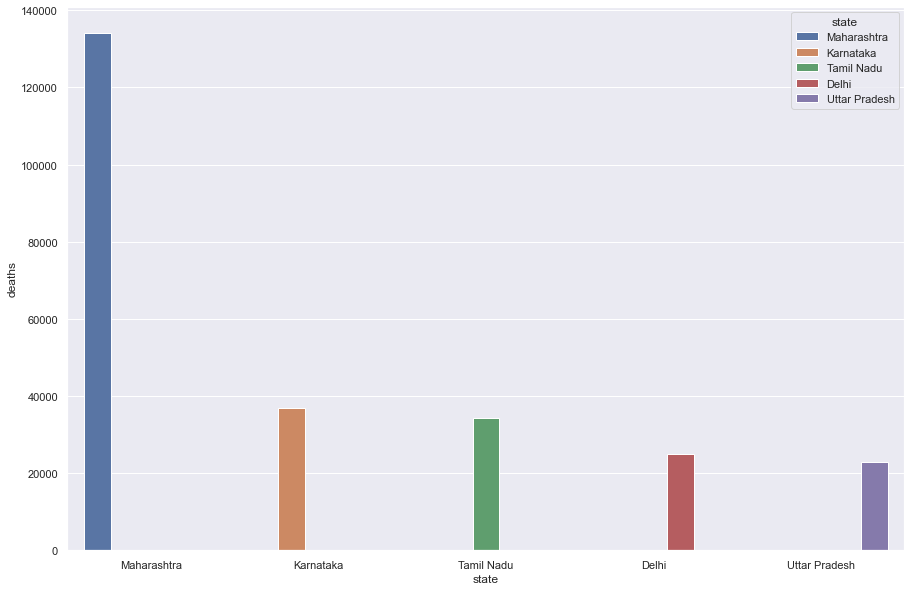
plt.show()

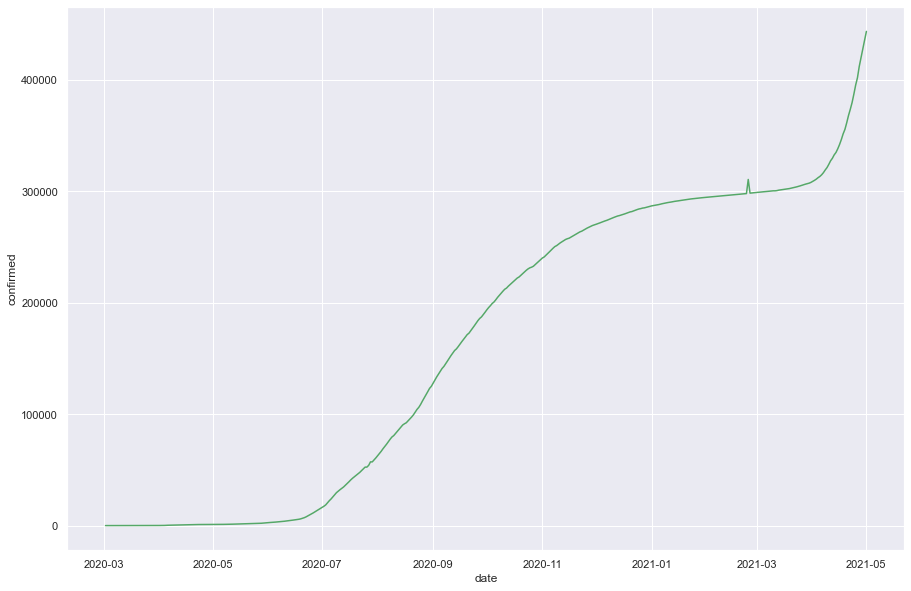
SYSTEM TESTING

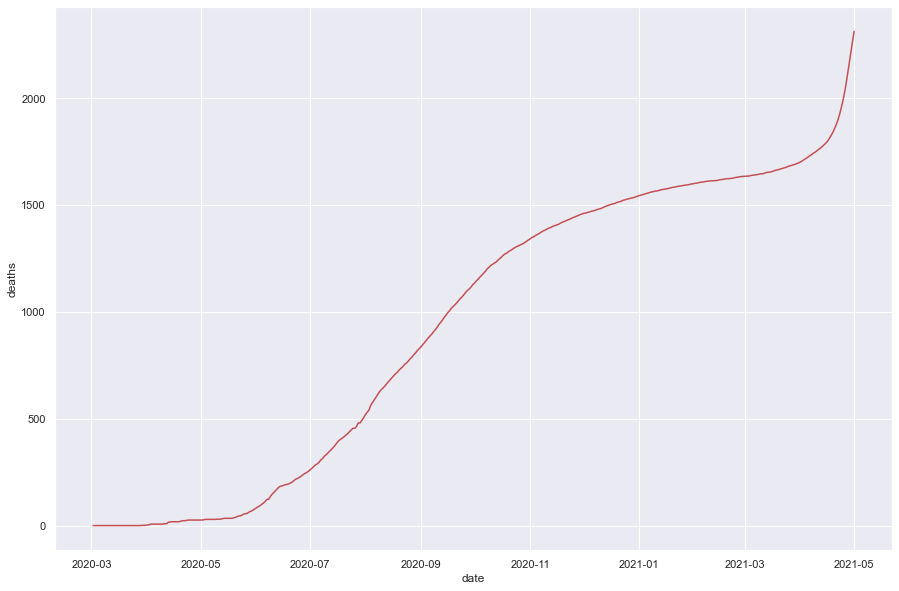


Here we can see that state Maharashtra has highest number of confirmed cases when compared to other states in india and second highest covid case is spread in kerala state it has second highest cases.









# CONCLUSION

A ML-based prediction model can be used as a screening tool to identify patients at risk of imminent ICU transfer within 24 h. This tool could improve the management of hospital resources and patient-throughput planning, thus delivering more effective care to patients hospitalized with COVID-19.

his study focused on the articles that applied machine-learning applications in COVID-19 disease for various purposes with different algorithms, 14 from 16 articles used supervised learning, and only one among them used unsupervised learning another one used both methods supervised and unsupervised learning and both of them shows accurate results. The studies used different machine-learning algorithms in different countries and by different authors but all of them related to the COVID-19 pandemic, (5) of these articles used Logistic regression algorithm, and all of them showed promising results in the COVID-19 health care applications and involvement. While (3) of the articles used artificial neural network (ANN) which also shows successful results, the rest of the 14 articles used different supervised and unsupervised learning algorithms and all of the models showed accurate results. Our conclusion is ML applications in medicine showed promising results with high accuracy, sensitivity, and specificity using different models and algorithms. In general, the paper results explored the supervised learning is more accurate to detect the COVID-19 cases which were above (92%) compare to the unsupervised learning which was mere (7.1%).

The COVID-19 pandemic outbreak has devastated the whole world and lead to a state of worldwide health emergency. Several efforts have been performed to combat this pandemic. In this study, we aimed to explore the impact of vital signs, chronic disease, preliminary clinical data, and demographic features to predict the mortality and survival of the COVID-19 patients using supervised machine learning algorithms. Due to the reduced mortality risk of the COVID-19 cases, the dataset suffers from data imbalance. SMOTE technique was used to alleviate the data imbalance. The results showed that random forest outperformed the other models using 10-fold cross-validation. Grid search technique was applied for parameter optimization. The study achieved the accuracy of 0.952 and AUC of 0.99. Despite the significant outcome achieved from this proposed model, there is still a need for improvement. The models need to be validated using multiple datasets. Furthermore, in the future, we will incorporate and explore the impact of other clinical features and laboratory results that were identified as significant in the previous studies.

Machine Learning Bibliography

Books

* Hastie, Friedman, and Tibshirani, The Elements of Statistical Learning, 2001
* Bishop, Pattern Recognition and Machine Learning, 2006
* Ripley, Pattern Recognition and Neural Networks, 1996
* Duda, Hart, and Stork, Pattern Classification, 2nd Ed., 2002
* Tan, Steinbach, and Kumar, [Introduction to Data Mining](http://www-users.cs.umn.edu/~kumar/dmbook/index.php), Addison-Wesley, 2005.
* Scholkopf and Smola, Learning with Kernels, 2002
* Mardia, Kent, and Bibby, Multivariate Analysis, 1979
* [Computational Statistics](http://www.quantlet.org/mdstat/scripts/csa/html/csahtml.html) (online book)
* Sutton and Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.
* Bertsekas and Tsitsiklis, Neuro-Dynamic Programming, Athena Scientific, 1996.

Other machine learning courses

* [Andrew Ng](http://www.stanford.edu/class/cs229/materials.html)
* [Max Welling](http://www.ics.uci.edu/~welling/classnotes/classnotes.html)

Data repositories

* [UCI Machine Learning Repository](http://archive.ics.uci.edu/ml/)
* [Physionet](http://www.physionet.org/)
* [MNIST Handwritten Digits](http://yann.lecun.com/exdb/mnist/)
* [Handwritten digits, faces, text](http://www.cs.toronto.edu/~roweis/data.html) in Matlab format
* [Medical imaging](http://www.via.cornell.edu/databases/)
* [Image registration](http://www.isi.uu.nl/Research/Databases/GS/)

Background

* [The Matrix Cookbook](http://www.ics.uci.edu/~welling/teaching/KernelsICS273B/MatrixCookBook.pdf) by Kaare Brandt Petersen and Michael Syskind Pedersen.
* [Convex Optimization](http://www.stanford.edu/~boyd/cvxbook/) by Boyd and Vandenberghe

Matlab Software

* [CVX](http://www.stanford.edu/~boyd/cvx/) convex program solver by Stephen Boyd
* [YALMIP](http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php?n=Main.What), a high-level Matlab interface to a variety of convex program solvers, such as SeDuMi
* [SeDuMi](http://sedumi.ie.lehigh.edu/), for solving second order cone programs. Most if not all tractable convex programs can be cast as such.
* [LIBSVM](http://www.csie.ntu.edu.tw/~cjlin/libsvm/), for support vector classification (including multiclass), regression, and one-class classification (novelty detection).

Conferences/Publications

* [NIPS](http://nips.cc/)
* [ICML](http://www.machinelearning.org/icml.html)
* [AISTATS](http://www.aistats.org/)
* [JMLR](http://www.jmlr.org/)
* [MLJ](http://www.springer.com/computer/artificial/journal/10994)
* [TPAMI](http://www.computer.org/tpami/)

Nearest Neighbors

* The primary research area relating to nearest neighbor methods is the problem of storage, data reduction, and rapid calculation of nearest neighbors. A search on "nearest neighbor search" or "condensed nearest neighbors" or "editted nearest neighbors" will return a number of references.
* Theory: Devroye, Gyorfi and Lugosi, A Probabilistic Theory of Pattern Recognition, 1996

Density Estimation

* David Scott, Multivariate Density Estimation, 1992
* Theory: Devroye and Lugosi, Combinatorial Methods in Density Estimation, 2001

Linear methods for classification

* Hastie et al, Bishop, and Duda et al. all have chapters on LDA, logistic regression, and other linear classifiers.

Decision Trees

* The first comprehensive treatment and still a standard reference: Brieman, Friedman, Olshen and Stone, Classification and Regression Trees, 1984
* The other standard reference is Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.
* A somewhat recent survey of research on decision trees: Sreerama K. Murthy: Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey. Data Min. Knowl. Discov. 2(4): 345-389 (1998)
* Ripley has a nice chapter on decision trees -- probably the best place to start.

Error estimation

* [On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation](http://jmlr.csail.mit.edu/papers/volume11/cawley10a/cawley10a.pdf), Gavin C. Cawley, Nicola L. C. Talbot; JMLR 11(Jul):2079-2107, 2010. Discussion of how certain model selection strategies are more biased than others; essential reading if you are doing comparative studies of different machine learning methods.
* [Descent Methods for Tuning Parameter Refinement](http://jmlr.csail.mit.edu/proceedings/papers/v9/lorbert10a/lorbert10a.pdf), Alexander Lorbert, Peter Ramadge ; AISTATS 2010. A natural idea.
* An entry level discussion of the bootstrap, cross-validation, and other error estimates is given in Efron and Tibshirani, An Introduction to the Bootstrap, 1993.

Boosting

* Adaboost was first developed in Freund and Schapire, [A decision-theoretic generalization of on-line learning and an application to boosting](http://www.cs.princeton.edu/~schapire/uncompress-papers.cgi/FreundSc95.ps). Journal of Computer and System Sciences, 55(1):119-139, 1997.
* A simpler proof the Adaboost's weak learning property is given in Robert E. Schapire and Yoram Singer.[Improved boosting algorithms using confidence-rated predictions](http://www.cs.princeton.edu/~schapire/uncompress-papers.cgi/SchapireSi98.ps). Machine Learning, 37(3):297-336, 1999.
* The view of Adaboost as performing functional gradient descent was observed by a number of researchers in the late 90's and early 00's. A representative work is L. Mason, J. Baxter, P. L. Bartlett, and M. Frean. Functional gradient techniques for combining hypotheses. In A. J. Smola, P. L. Bartlett, B. Sch?kopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 221-246. MIT Press, 2000.
* Logitboost was introduced in Friedman, J.H., Hastie, T., Tibshirani, R.: [Additive logistic regression: a statistical view of boosting](http://www-stat.stanford.edu/~hastie/Papers/AdditiveLogisticRegression/alr.pdf). Annals of Statistics **28**, 337-407 (with discussion) (2000).
* [Empirical Bernstein Boosting](http://jmlr.csail.mit.edu/proceedings/papers/v9/shivaswamy10a/shivaswamy10a.pdf), Pannagadatta Shivaswamy, Tony Jebara; AISTATS 2010.
* Many other references to boosting can be found on Robert Schapire's web page.

Support Vector Machines

* The original paper: Corinna Cortes and V. Vapnik, ["Support-Vector Networks''](http://homepage.mac.com/corinnacortes/papers/SVM.ps), Machine Learning, **20**, 1995
* The standard reference: Scholkopf and Smola, Learning with Kernels, 2002
* Algorithms for solving the SVM are discussed E. Osuna, R. Freund, and F. Girosi. "Improved training algorithm for support vector machines." NNSP'97, 1997. <http://citeseer.ist.psu.edu/osuna97improved.html>, and in J. Platt, Fast Training of Support Vector Machines using Sequential Minimal Optimization, in [Advances in Kernel Methods - Support Vector Learning](http://kernel-machines.org/nips97/book.html),  B. Sch?kopf, C. Burges, and A. Smola, eds., MIT Press, 1999.

Clustering

* K-means, EM for Gaussian mixture models, and hierarchical clustering: see the recommended texts, especially Hastie et al., Duda et al., and Bishop (although Bishop doesn't discuss hierarchical clustering). K-means is also known as the Lloyd-Max algorithm in the context of vector quantization.
* EM was originally introduced by Dempster, A. P., Laird, N. M. Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, B, 39, 1-38.
* Spectral clustering: an excellent introduction to spectral clustering is the following: U. Luxburg, ``[A Tutorial on Spectral Clustering](http://www.kyb.mpg.de/publications/attachments/Luxburg07_tutorial_4488%5B0%5D.pdf)," Statistics and Computing 17(4), 395-416 (2007).

Dimensionality reduction

* Principal components analysis: The book by Mardia, Kent and Bibby derives PCA for the ``population" case (the sample case being analagous) for both the maximum orthogonal variance perspective and the least squares linear approximation perspective. Note that PCA is also known as the Karhunen-Loeve transform (KLT).
* Multidimensional scaling: The book by Mardia, Kent and Bibby has a clean and rigorous derivation of classical MDS, associated optimality properties, and connections to PCA. It also discusses nonmetric MDS methods.
* The ``majorization" approach to metric MDS via stress minimization is reviewed and analyzed by Jan de Leeuw, "Convergence of the Majorization Method for Multidimensional Scaling," *Joumal of Classification* 5:163-180 (1988)
* [Isomap](http://www.sciencemag.org/cgi/reprint/290/5500/2319.pdf)
* [Local linear embedding (LLE)](http://www.sciencemag.org/cgi/reprint/290/5500/2323.pdf)
* [Laplacian eigenmaps](http://math.uchicago.edu/~misha/paper1.ps)
* Kernel PCA is covered in the book by Scholkopf and Smola, or see the original paper referenced therein.
* [Manifold learning resource page](http://www.cse.msu.edu/~lawhiu/manifold/)
* Self-organizing maps, principal curves, and independent component analysis (ICA) may be reviewed in Hastie et al.
* Factor analysis is treated in Mardia et al.
* [An Introduction to Variable and Feature Selection](http://www.kernel-machines.org/jmlr/volume3/guyon03a/guyon03a.pdf), an excellent survey and introduction to methods of variable section that appeared in *Journal of Machine Learning Research* 3 (2003) 1157-1182.
* The following article describes extensive simulations for various learning algorithms combined with different feature selection methods, and offers some good intuition: Hua, J., Xiong, Z., Lowey, J., Suh, E., and E. R. Dougherty, [Optimal Number of Features as a Function of Sample Size for Various Classification Rules](http://gsp.tamu.edu/Publications/PDFpapers/pap_Bio_optimal_number.pdf), Bioinformatics, 21, No. 8, 1509-1515, 2005.

Nonlinear regression and Gaussian Processes

* [Introduction to Gaussian Processes](http://www.ics.uci.edu/~welling/teaching/KernelsICS273B/gpB.pdf) by David MacKay.
* [Kernel ridge regression](http://web.eecs.umich.edu/~cscott/past_courses/eecs545f09/bib.html) by Max Welling
* [Support vector regression](http://web.eecs.umich.edu/~cscott/past_courses/eecs545f09/bib.html) by Max Welling
* [Approximations for Binary Gaussian Process Classification](http://www.jmlr.org/papers/volume9/nickisch08a/nickisch08a.pdf) by Hannes Nickisch and Carl Edward Rasmussen.