# MACHINE LEARNING

## DIGITAL ASSESSMENT – 1

APRIL 9, 2023


**DEEKSHITH REDDY B. U**

**20MIC0058**

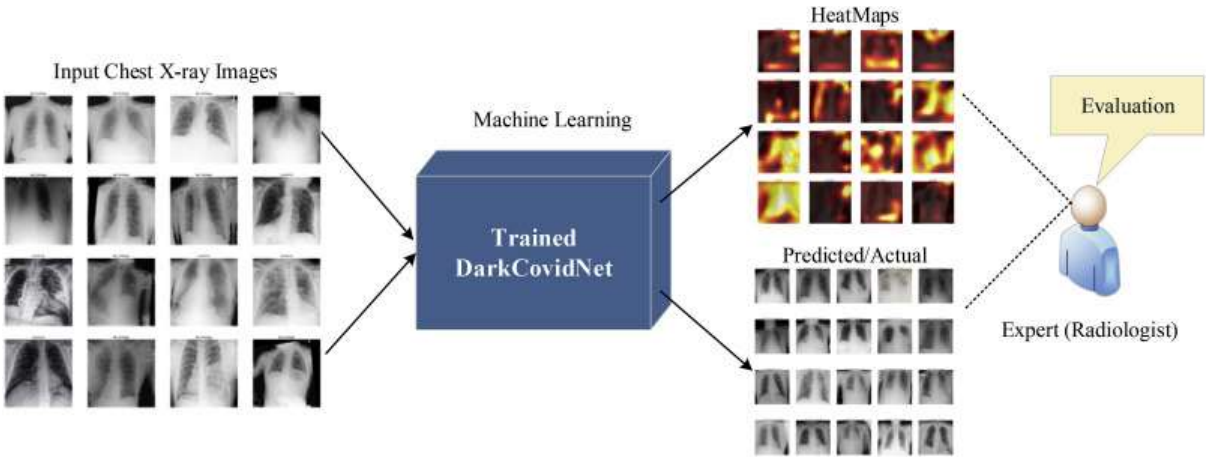
# DA DOCUMENT

## 1. Write about the project and its functionalities

## Covid-19 detection Using Machine Learning

The COVID-19 detection project using machine learning is designed to predict whether a person is infected with the COVID-19 virus based on their chest X-ray image. The project uses a deep learning model to classify chest X-ray images as either COVID-19 positive or negative.

The project first collects a dataset of chest X-ray images from COVID-19 positive and negative patients. The dataset is then preprocessed to normalize the images and remove any noise. The preprocessed images are then used to train a deep learning model, such as a convolutional neural network (CNN), using supervised learning.




The trained model is then used to predict whether a given chest X-ray image belongs to a COVID-19 positive or negative patient. The project can be deployed as a web application or mobile app, where users can upload their chest X-ray images and receive a prediction of whether they are infected with the virus.

The project aims to provide a faster and more accurate method for detecting COVID-19 infections, especially in areas where testing is limited or not readily available. However, it is important to note that the model's predictions should not replace medical diagnosis and treatment.

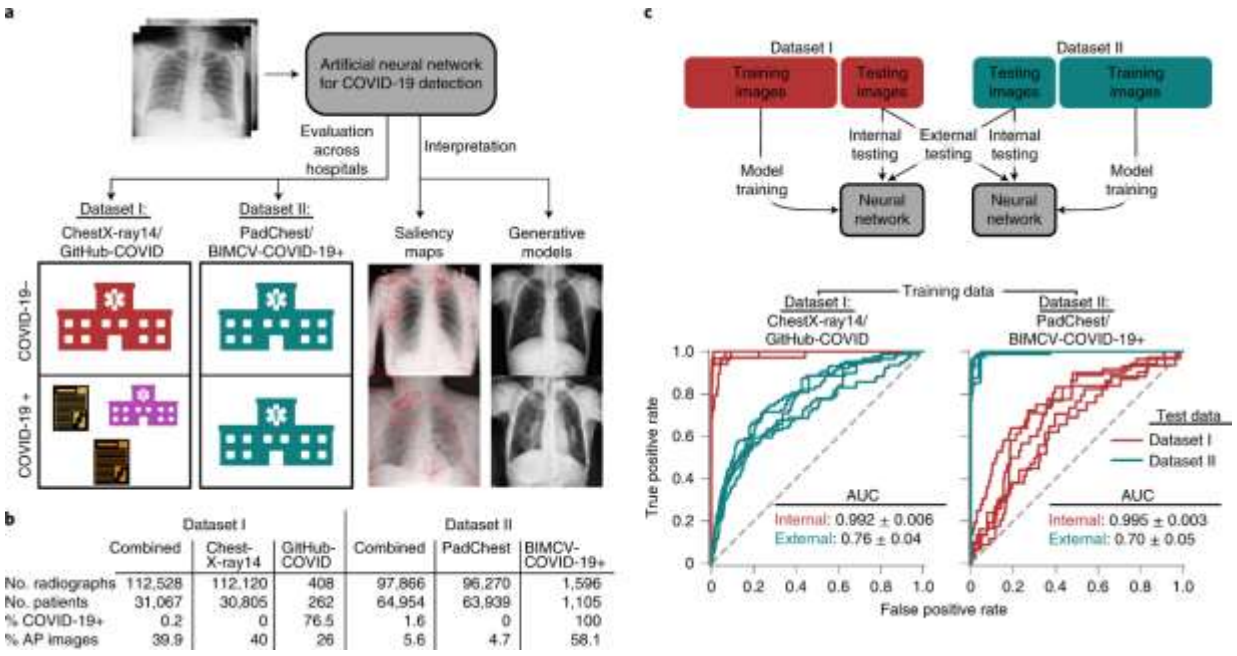
## COVID-19 Detection System using Machine Learning Classification Algorithms

An infected person shows symptoms within 2-14 days. According to W.H.O the symptoms and signs of moderate to severe conditions are dry cough, fatigue and fever while in severe cases dyspnea, fever and fatigue may occur.

People with other illnesses such as asthma, diabetes, and heart disease are at greater risk of contracting the virus and may become seriously ill. A system which can be used to detect the virus has become necessary due to the rapid spread of the virus, killing hundreds of thousands of people.

Machine learning classification algorithms, data sets and machine learning software are essential tools for designing the COVID-19 predictive model. This project aims to compare different machine learning algorithms like K-nearest neighbors, Random forest and Naive Bayes with respect to their accuracies and then use the best one among them to develop a system which predicts whether a person has COVID or not using the data provided to the model

The Covid-19 detection project using machine learning aims to provide a tool for the early detection of Covid-19 based on medical imaging data. Some of the functionalities of the project are:



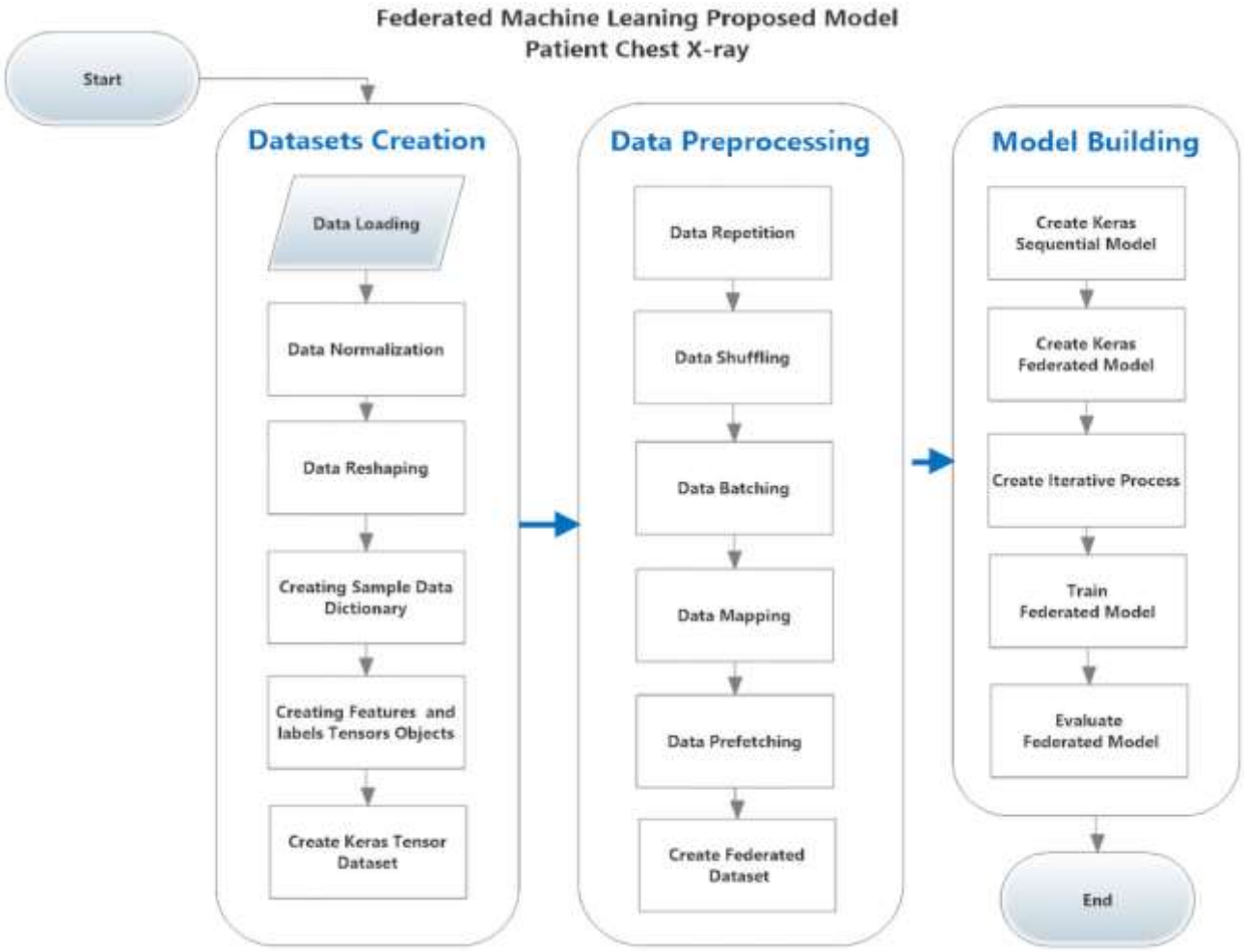

The Covid-19 detection project using machine learning aims to provide a tool for the early detection of Covid-19 based on medical imaging data. Some of the functionalities of the project are:

1. **Data preprocessing**: The project preprocesses the medical imaging data, which involves image resizing, normalization, and segmentation, to make it suitable for the machine learning algorithms.
2. **Feature extraction**: The project extracts relevant features from the preprocessed medical images using feature extraction techniques like Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), and Convolutional Neural Networks (CNN).
3. **Machine learning models**: The project uses various machine learning models like Random Forest, Support Vector Machines (SVM), and Convolutional Neural Networks (CNN) to train on the extracted features to classify the medical images as Covid-19 positive or negative.
4. **Testing**: The trained models are tested on the testing dataset to evaluate the performance of the model.
5. **Deployment**: The best-performing model is deployed as a web application where medical practitioners can upload medical images

of patients to predict the presence of Covid-19. The web application provides the predictions along with the probability of the prediction

One case study of COVID-19 detection using machine learning was conducted by a team of researchers in India. They used a dataset of chest X-ray images from COVID-19 positive and negative patients and applied various machine learning algorithms such as Convolutional Neural Networks (CNNs) and Support Vector Machines (SVMs) to classify the images as positive or negative for COVID-19.

The results showed that the CNN model achieved the highest accuracy of 98.77% in detecting COVID-19 from X-ray images, demonstrating the potential of machine learning for accurate and efficient diagnosis of COVID-19



## 2. Your observation on the project and possible additional functionalities

# OBSERVATION

### A. Data Collection

As the WHO has declared the Coronavirus pandemic as a health emergency, researchers and hospitals have provided open access to data related to the epidemic. We procured a data set from kaggle.com and it has 5434 × 21 rows of columns. This dataset contains 20 variables that could be determinants in the prediction of COVID-19, as well as one class attribute that defines if COVID-19 is found.

### B. Data Preprocessing

The process of converting raw data into a comprehensible format is known as data preprocessing. Real-world data may have noise, missing values, or be in an incompatible format that prevents it from being directly used in machine learning models. Preprocessing of data is an essential step in which we clean the data and make it compatible.

The main steps in data preprocessing are as follows:

**1) Removing features:** we can conclude that wearing masks and sanitization from the market are two features that have only one value which is 'no' as they don't affect our predictions we can simply just drop those columns off our dataset.

**2) Encoding Categorical Data:** Labeling Coding is a popular form of code flexible code management for categories. In this process, each label is given a whole number based on alphabetical order. All attributes of our

dataset are of ‘yes’ or ‘no’ type so we have used label encoding to convert it to 0 and 1 for the model to understand the dataset better. Table 4 shows the dataset after applying label encoding.

**3) Splitting the Dataset:** The next stage in machine learning data preprocessing is to split the dataset. A machine learning model’s dataset should be split into two parts: training and testing. This means that we use 80% of the data to train the model while keeping the remaining 20% for testing. We take all the 20 independent attributes into x and the dependent column ‘COVID-19’ into y as we aim to predict if the patient is COVID positive or not.

| | missing_values | percent_missing % |
|---|---|---|
| Breathing Problem | 0 | 0.0 |
| Fever | 0 | 0.0 |
| Dry Cough | 0 | 0.0 |
| Sore throat | 0 | 0.0 |
| Running Nose | 0 | 0.0 |
| Asthma | 0 | 0.0 |
| Chronic Lung Disease | 0 | 0.0 |
| Headache | 0 | 0.0 |
| Heart Disease | 0 | 0.0 |
| Diabetes | 0 | 0.0 |
| Hyper Tension | 0 | 0.0 |
| Fatigue | 0 | 0.0 |
| Gastrointestinal | 0 | 0.0 |
| Abroad travel | 0 | 0.0 |
| Contact with COVID Patient | 0 | 0.0 |
| Attended Large Gathering | 0 | 0.0 |
| Visited Public Exposed Places | 0 | 0.0 |
| Family working in Public Exposed Places | 0 | 0.0 |
| Wearing Masks | 0 | 0.0 |
| Sanitization from Market | 0 | 0.0 |
| COVID-19 | 0 | 0.0 |

# IMPLEMENTATION

With the growth of computer technology, predictive mod eling is changing. We are now able to make predictable modeling more efficient, and less expensive than before. In our project, we use various classification algorithms to predict and use a grid search CV to find the most advanced solution for each algorithm. Some of the categorization algorithms that have been employed include:

The implementation of a COVID-19 detection system using machine learning involves several steps:

**A. Logistic Regression**

Logistic regression is a data categorization technique that uses machine learning. This algorithm, models the odds of the potential outcomes of a single experiment using a logistic function. The easiest way to understand the influence of numerous independent factors on a single outcome variable is to use logistic regression, which was designed for this purpose. In general, the algorithm calculates the probability of belonging to a particular class. We have two classes here, y=0,1.

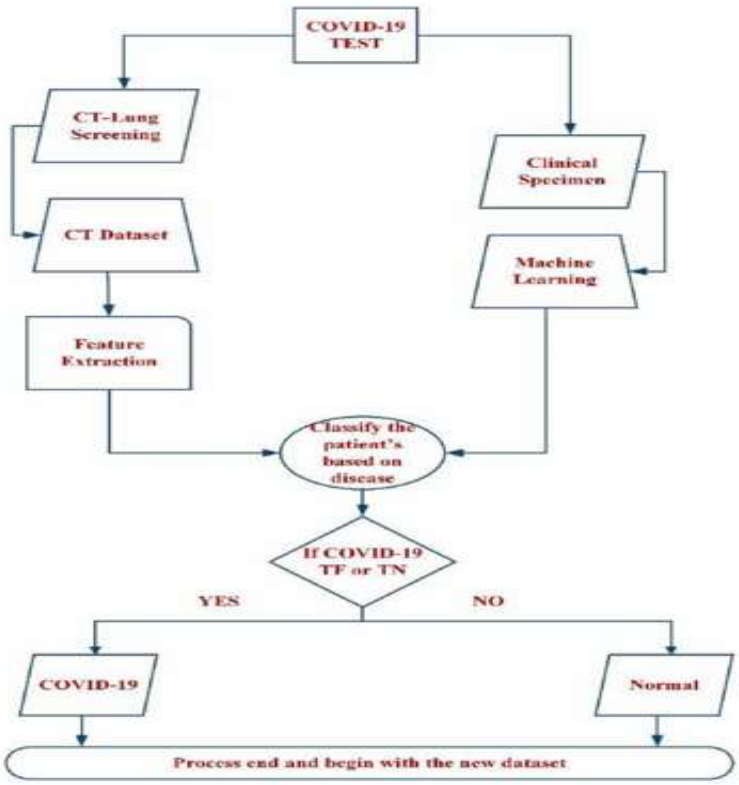
**B. K Nearest Neighbours**

The oldest supervised machine learning algorithm for classification is KNN, which classifies a given instance according to the majority of categories among its k-nearest neighbours in the dataset. The distance between the item to be categorized and every other item in the data set is calculated by the algorithm.

**C. Random Forest**

This classifier is a meta-estimator that adapts to decision trees on the dataset's different sub-samples and utilizes the average to increase the model's predicted accuracy and control over-fitting. In most circumstances, this random forest classifier seems to be more accurate than decision trees, and it also minimizes overfitting. At the Random Forest level, average over all the trees is the final feature importance. The feature's importance sum value on each tree is numerically calculated and divided by the total number of trees:

The implementation of a COVID-19 detection system using machine learning requires expertise in data science, machine learning, and software engineering. The accuracy of the model depends on the quality of the data and the effectiveness of the machine learning algorithms used. Therefore, it is essential to have a robust data collection process, data pre-processing techniques, and a suitable machine learning model to achieve accurate results.




# Implementation of the K-Nearest Neighbors algorithm for COVID-19 detection in Python:

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy_score

**# Load the dataset**

data = pd.read_csv('covid_data.csv')

**# Split the dataset into training and testing sets**

X_train, X_test, y_train, y_test = train_test_split(data.drop('COVID', axis=1), data['COVID'], test_size=0.2, random_state=42)

```
# Train the K-Nearest Neighbors classifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
# Make predictions on the testing set
y_pred = knn.predict(X_test)
# Evaluate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)
```

## Possible additional functionalities

1. **Real-time monitoring:** The system can be integrated with IoT devices and sensors to collect real-time data on vital signs such as temperature, heart rate, and respiratory rate. This data can be analyzed by the machine learning model to detect potential Covid-19 cases and provide alerts to healthcare providers.

2. **Contact tracing**: The system can be used to trace the contacts of infected individuals and alert them to get tested. This can help to prevent the spread of the virus and limit the number of new cases.

3. **Vaccine distribution**: The system can be used to analyze data on vaccine distribution and prioritize areas that are most in need. This can help to ensure that vaccines are distributed fairly and efficiently.

4. **Predictive analytics:** The system can be used to analyze data on Covid-19 cases, hospitalizations, and deaths to predict future trends and patterns. This can help healthcare providers and policymakers to make informed decisions on resource allocation and public health interventions.

5. **Multilingual support:** The system can be made available in multiple languages to ensure that people from different backgrounds and cultures can access the service. This can help to promote equity in healthcare and reduce disparities in Covid-19 detection and treatment.

# 3. Additional functionalities

## a. Copy the main part of your functionality code and sample screenshot.

```
covid_data = pd.read_csv("Covid Dataset.csv")
```

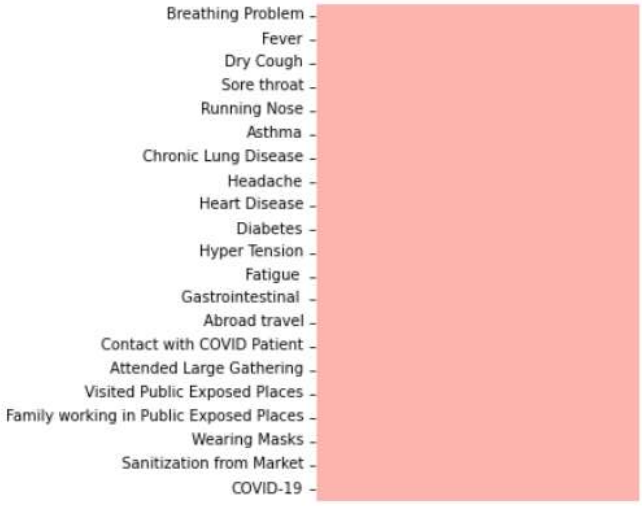
| | Breathing Problem | Fever | Dry Cough | Sore throat | Running Nose | Asthma | Chronic Lung Disease | Headache | Heart Disease | Diabetes | Hyper Tension | Fatigue | Gastrointestinal | Abroad travel | Contact with COVID Patient | Attended Large Gathering | Visited Public Exposed Places | Fa wor Pu Exp Pl |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Yes | Yes | Yes | Yes | Yes | No | No | No | No | Yes | Yes | Yes | Yes | No | Yes | No | Yes | |
| 1 | Yes | Yes | Yes | Yes | No | Yes | Yes | Yes | No | No | No | Yes | No | No | No | Yes | Yes | |
| 2 | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | Yes | No | Yes | Yes | Yes | No | No | No | |
| 3 | Yes | Yes | Yes | No | No | Yes | No | No | Yes | Yes | No | No | No | Yes | No | Yes | Yes | |
| 4 | Yes | Yes | Yes | Yes | Yes | No | Yes | Yes | Yes | Yes | Yes | No | Yes | No | Yes | No | Yes | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 5429 | Yes | Yes | No | Yes | Yes | Yes | Yes | No | No | No | No | Yes | Yes | No | No | No | No | |
| 5430 | Yes | Yes | Yes | No | Yes | Yes | No | Yes | No | Yes | Yes | Yes | No | No | No | No | No | |
| 5431 | Yes | Yes | Yes | No | No | No | No | No | Yes | No | Yes | No | No | No | No | No | No | |
| 5432 | Yes | Yes | Yes | No | Yes | No | No | Yes | Yes | No | No | No | No | No | No | No | No | |
| 5433 | Yes | Yes | Yes | No | Yes | Yes | No | Yes | No | Yes | Yes | Yes | No | No | No | No | No | |

434 rows × 21 columns

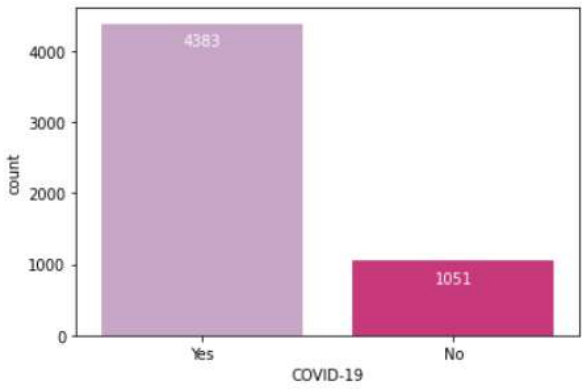
```
covid_data.describe().T
```

| | count | unique | top | freq |
|---|---|---|---|---|
| Breathing Problem | 5434 | 2 | Yes | 3620 |
| Fever | 5434 | 2 | Yes | 4273 |
| Dry Cough | 5434 | 2 | Yes | 4307 |
| Sore throat | 5434 | 2 | Yes | 3853 |
| Running Nose | 5434 | 2 | Yes | 2952 |
| Asthma | 5434 | 2 | No | 2920 |
| Chronic Lung Disease | 5434 | 2 | No | 2869 |
| Headache | 5434 | 2 | Yes | 2736 |
| Heart Disease | 5434 | 2 | No | 2911 |
| Diabetes | 5434 | 2 | No | 2646 |
| Hyper Tension | 5434 | 2 | No | 2771 |
| Fatigue | 5434 | 2 | Yes | 2821 |
| Gastrointestinal | 5434 | 2 | No | 2883 |
| Abroad travel | 5434 | 2 | No | 2983 |
| Contact with COVID Patient | 5434 | 2 | Yes | 2726 |
| Attended Large Gathering | 5434 | 2 | No | 2924 |
| Visited Public Exposed Places | 5434 | 2 | Yes | 2820 |
| Family working in Public Exposed Places | 5434 | 2 | No | 3172 |
| Wearing Masks | 5434 | 1 | No | 5434 |
| Sanitization from Market | 5434 | 1 | No | 5434 |
| COVID-19 | 5434 | 2 | Yes | 4383 |

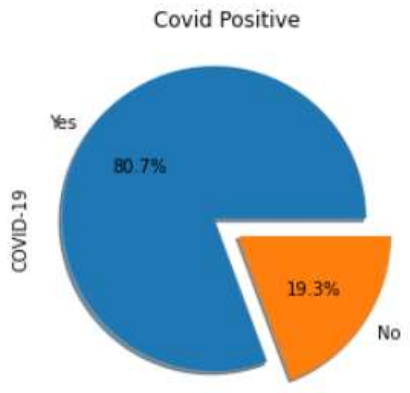
```
sns.heatmap(covid_data.isnull(),yticklabels=False,cbar=False,cmap='Pastel1')
```



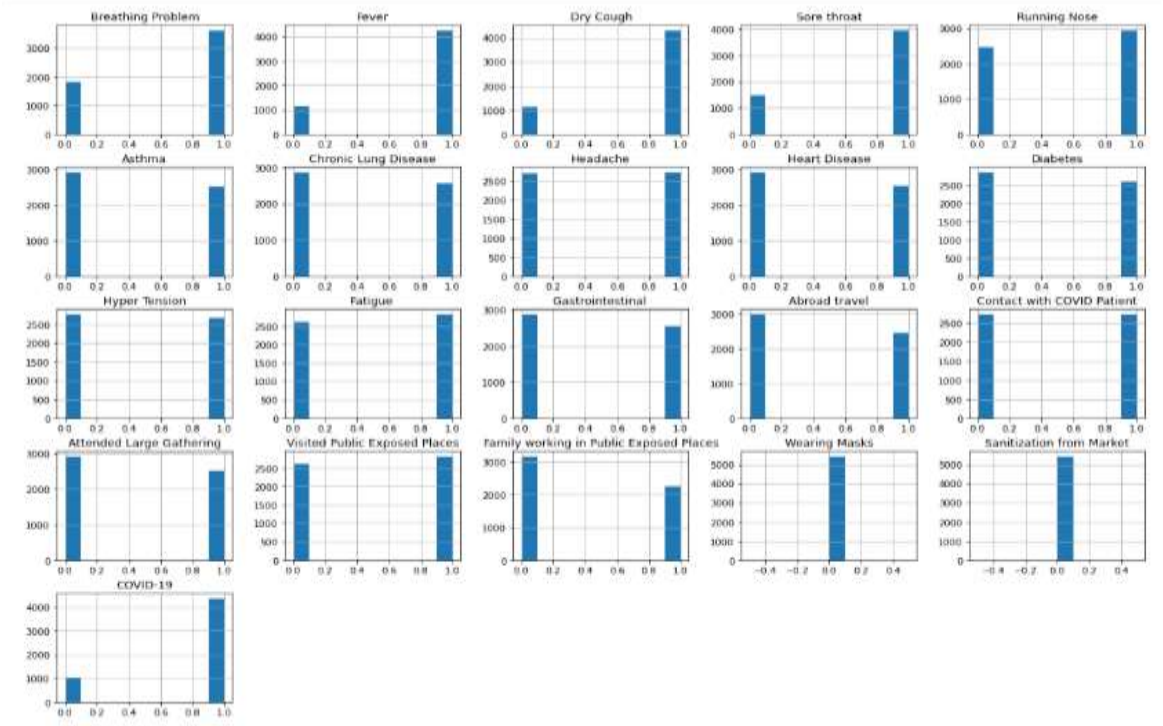

```
ax = sns.countplot(x='COVID-19',data=covid_data, palette="PuRd")
for p in ax.patches:
     ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.4, p.get_height()+100),
ha='center', va='top', color='white', size=10)
plt.show()
```



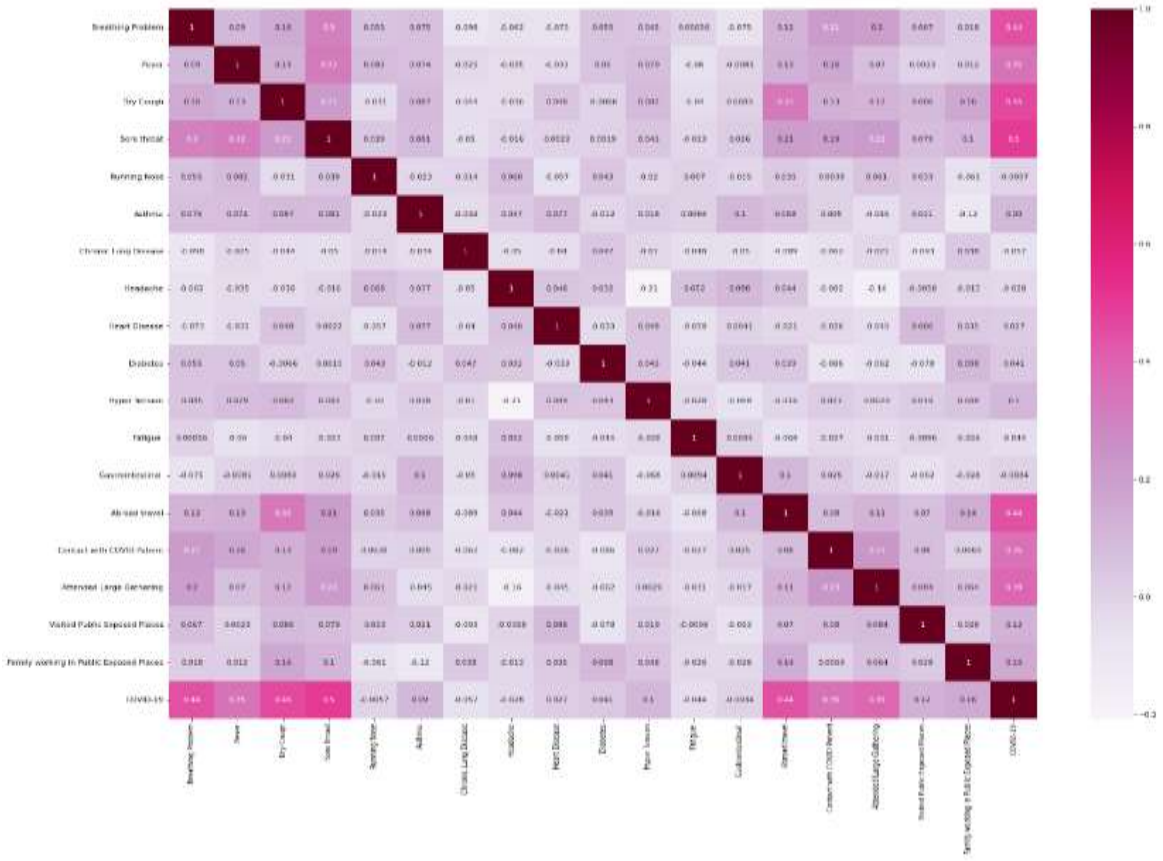

```
covid_data["COVID-
19"].value_counts().plot.pie(explode=[0.1,0.1],autopct='%1.1f%%',shadow=True)
plt.title('Covid Positive');
```



```
covid_data.hist(figsize=(20,15));
```



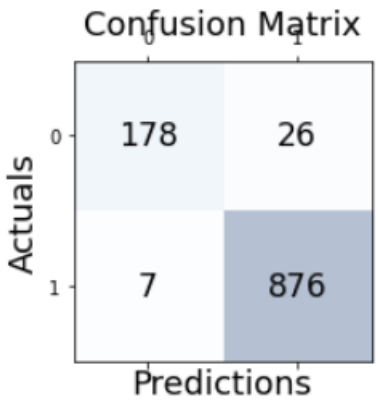

```
plt.figure(figsize=(25,20))
sns.heatmap(covid_data.corr(), annot=True, cmap="PuRd")
```



# LOGISTIC REGRESSION

```
print("LOGISTIC REGRESSION")
start = time.time()
lr = LogisticRegression()
lr.fit(x_train, y_train)
end = time.time()

print_performance2(y_test,lr,'LOGISTIC REGRESSION')
#acc = lr.score(x_train, y_train)*100
#accuracies['LOGISTIC REGRESSION'] = acc
algo_time['LOGISTIC REGRESSION']=end-start
```

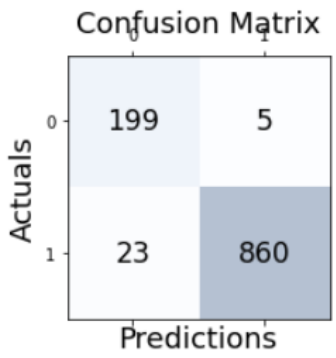
```
LOGISTIC REGRESSION
ROC_AUC value : 93.23107498945218 %

Mean Squared Error : 3.035878564857406 %

R2 score is : 80.08627006861634 %

Accuracy Score : 97.03243616287095 %

Classification Report :
              precision    recall  f1-score   support

           0       0.96      0.87      0.92       204
           1       0.97      0.99      0.98       883

    accuracy                           0.97      1087
   macro avg       0.97      0.93      0.95      1087
weighted avg       0.97      0.97      0.97      1087
```



# K-NEAREST NEIGHBOURS

```
start = time.time()
knn = KNeighborsClassifier()
# assigning the dictionary of variables whose optimium value is to be
retrieved
param_grid = {'n_neighbors' : np.arange(1,50)}
knn_cv = GridSearchCV(knn, param_grid, cv=5)
# training the model with the training data and best parameter
knn_cv.fit(x_train,y_train)
end=time.time()
algo_time['K-NEAREST NEIGHBOURS']=end-start
# finding out the best parameter chosen to train the model
print("The best paramter we have is: {}" .format(knn_cv.best_params_))
# finding out the best score the chosen parameter achieved
print("The best score we have achieved is: {}" .format(knn_cv.best_score_))

print("K-NEAREST NEIGHBOURS")
print_performance2(y_test,knn_cv,'K-NEAREST NEIGHBOURS')
#acc = knn_cv.score(x_train, y_train)*100
#accuracies['K-NEAREST NEIGHBOURS'] = acc
```

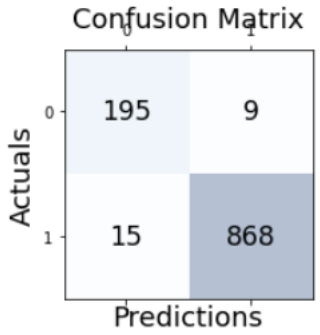
```
K-NEAREST NEIGHBOURS
ROC_AUC value : 97.47213154797593 %

Mean Squared Error : 2.5758969641214353 %

R2 score is : 83.10350187640175 %

Accuracy Score : 98.3666896710375 %

Classification Report :
              precision    recall  f1-score   support

           0       0.90      0.98      0.93       204
           1       0.99      0.97      0.98       883

    accuracy                           0.97      1087
   macro avg       0.95      0.97      0.96      1087
weighted avg       0.98      0.97      0.97      1087
```



# RANDOM FOREST

```
rf_start=time.time()
rfc=RandomForestClassifier(random_state=42)
param_grid = {
    'n_estimators': [200, 500],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [4,5,6,7,8],
    'criterion' :['gini', 'entropy']
}
CV_rfc = GridSearchCV(estimator=rfc, param_grid=param_grid, cv= 5)
CV_rfc.fit(x_train, y_train)
rf_end=time.time()
algo_time['RANDOM FOREST TREE']=rf_end-rf_start
# finding out the best parameter chosen to train the model
print("The best paramter we have is: {}" .format(CV_rfc.best_params_))

# finding out the best score the chosen parameter achieved
print("The best score we have achieved is: {}"
.format(CV_rfc.best_score_*100))
```

```
RANDOM FOREST TREE
ROC_AUC value : 96.94474052361602 %

Mean Squared Error : 2.2079116835326587 %

R2 score is : 85.51728732263008 %

Accuracy Score : 98.38969404186795 %

Classification Report :
              precision    recall  f1-score   support

           0       0.93      0.96      0.94       204
           1       0.99      0.98      0.99       883

    accuracy                           0.98      1087
   macro avg       0.96      0.97      0.96      1087
weighted avg       0.98      0.98      0.98      1087
```



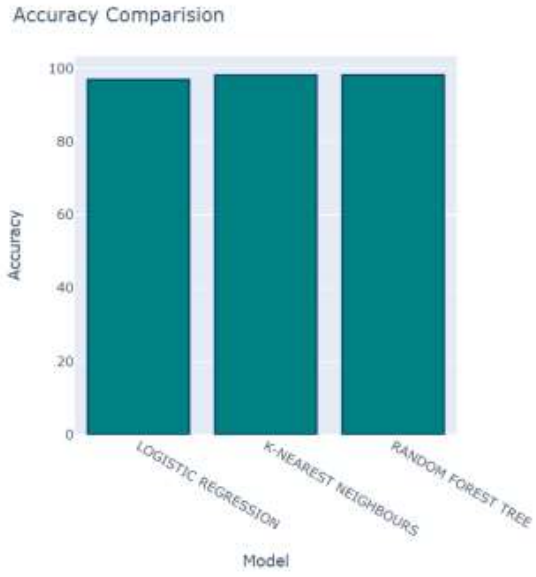
```
patient =
[[Breathing_Problem,Fever,Dry_Cough,Sore_throat,Running_Nose,Asthma,Chronic_Lu
ng_Disease,Headache,Heart_Disease,Diabetes,Hyper_Tension,Fatigue,Gastrointesti
nal,Abroad_travel,Contact_with_COVID_Patient,Attended_Large_Gathering,Visited_
Public_Exposed_Places,Family_working_in_Public_Exposed_Places]]
result = knn_cv.predict(patient)
print("\nResults : ",result)

if result == 1:
    print(Fore.RED + 'You may be affected with COVID-19 virus! Please get
RTPCR test ASAP and stay in Quarantine for 14 days!')
    print()
else :
    print(Fore.GREEN + 'You do not have any symptoms of COVID-19. Stay home!
Stay safe!')
    print()
```

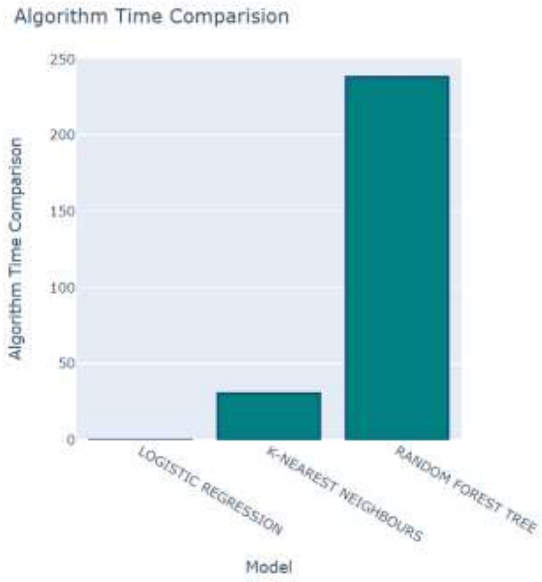
```
COVID PREDICTION BASED ON ML ALGORITHMS
Enter 1 for Yes and 0 for No
Does the patient have breathing problem ? 1
Does the patient have fever ? 1
Does the patient have dry cough ? 1
Does the patient have sore throat ? 0
Does the patient have running nose ? 1
Does the patient have any record of asthma ? 0
Does the patient have any records of chronic lung disease ? 0
Is the patient having headche ? 0
Does the patient have any record of any heart disease ? 0
Does the patient have diabetes ? 1
Does the patient have hyper tension ? 1
Does the patient experience fatigue ? 1
Does the patient have any gastrointestinal disorders ? 0
Has the patient travelled abroad recently ? 0
Was the patient in contact with a covid patient recently ? 0
Did the patient attend any large gathering event recently ? 1
Did the patient visit any public exposed places recently ? 1
Does the patient have any family member working in public exposed places ? 0

Results :  [1]
You may be affected with COVID-19 virus! Please get RTPCR test ASAP and stay in Quarantine for 14 days!

/usr/local/lib/python3.7/dist-packages/sklearn/base.py:446: UserWarning:

X does not have valid feature names, but KNeighborsClassifier was fitted with feature names
```

# RESULT

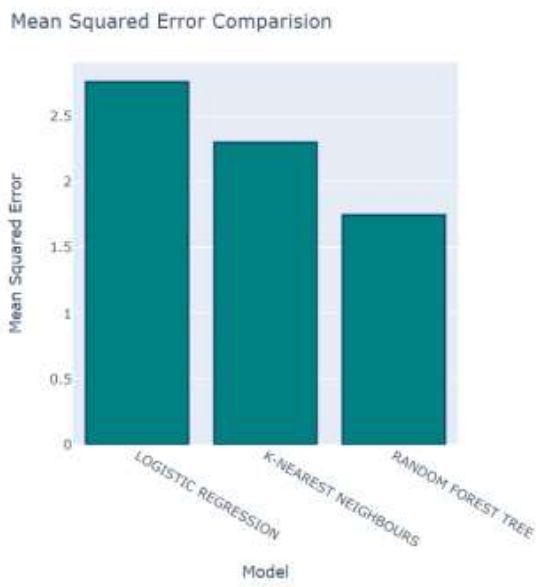
To evaluate the effectiveness of the Machine Learning algorithms applied in this experiment, we decided to adopt the Accuracy, Mean squared error, Precision, Recall and F-Measure which are widely used in domains such as information retrieval, machine learning and other domains that involve binary classification.

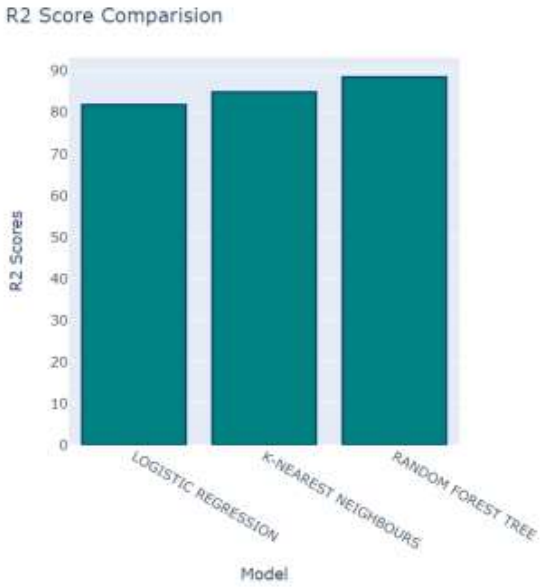
The Random Forest Tree method has the best accuracy of all the algorithms, with a score of 98.39 percent. Additionally, R2 scores, mean squared errors and ROC scores are plotted in the bar chart using red, green and purple respectively. With 98.37 percent accuracy, the KNN is the next most acceptable algorithm to use.




The lowest MSE was attained by the Random Forest algorithm, with 2.207% followed by KNN having 2.57%. R2 scores measure how well the model fits over the data. So here we see that Random Forest have the highest score of 85.51% followed by KNN with 83.1%. Finally, as shown in Figure 10, the time it took to develop the model was taken into account, as this is an important factor in determining the best algorithm for our prediction model. Based on the results, Logistic Regression was the fastest algorithm with 0.05s to train a classifier model.




# CONCLUSION

In conclusion, Covid-19 detection using machine learning is an important tool in the fight against the spread of the virus. It has the potential to accurately predict whether a person has Covid-19 or not, which can aid in early detection and treatment. The implementation of such a system can help in reducing the transmission rate of the virus and ultimately saving lives.

| | Accuracy | MSE | R2 score | ROC score | Running time |
|---|---|---|---|---|---|
| KNN | 98.37% | 2.57 | 83.1 | 98.58 | 24.252 |
| Logistic Regression | 97.03% | 3.036 | 80.086 | 93.23 | 0.038 |
| Random Forest | 98.39% | 2.207 | 85.51 | 97.41 | 213.331 |

TABLE I
COMPARISON OF METRICS FOR KNN, LOGISTIC REGRESSION AND RANDOM FOREST

The results show that the KNN classifier with number of neighbors to be considered equal to 2 is the best machine learning algorithm, having an accuracy of 98.37%, and 0.026 mean absolute error considering the runtime for training. In comparison to other methods, the model takes average time but gives good accuracy.

Extra information or diagnoses from hospital records, persons who contracted the virus, COVID-19 survivors, patients under assessment, or management can all be included for future research. A software which can predict the severity of COVID-19 can indeed be deployed to provide further information about the steps that must be taken and the interventions that should be considered.

The machine learning algorithms used in the project, such as K-nearest neighbors, Random forest, and Naive Bayes, have shown promising results in accurately predicting the Covid-19 infection. However, it is important to note that this system is not a substitute for medical diagnosis and should be used in conjunction with other diagnostic tools. Further research and development are needed to improve the accuracy and efficiency of the system.