```
import pandas as pd
```

```
data = pd.read_csv("/content/drive/MyDrive/covid_data_2020-2021.csv.zip")
```

```
data.head()
```

	test_date	cough	fever	sore_throat	shortness_of_breath	head_ache	corona_result	age_60_and_above	gender	test_
0	2021-10-11	0	0	0	0	0	Negative	Yes	female	
1	2021-10-11	0	0	0	0	0	Negative	Yes	male	
2	2021-10-11	0	0	0	0	0	Negative	No	female	
	2021-10-									

```
data.shape
```

```
(5861480, 10)
```

```
data = data.drop('test_date' , axis = 1)
```

Mounting Google Drive...

```
(5861480, 9)
```

Checking duplicates

```
data.duplicated().any()
```

```
True
```

```
data.drop_duplicates()
```

	cough	fever	sore_throat	shortness_of_breath	head_ache	corona_result	age_60_and_above	gender	test_indi
0	0	0	0	0	0	Negative	Yes	female	
1	0	0	0	0	0	Negative	Yes	male	
2	0	0	0	0	0	Negative	No	female	
12	0	0	0	0	0	Negative	No	male	
218	0	0	0	0	0	Positive	No	male	
...	...	...	...	...	...	...	...	...	
5856327	0	0	1	1	1	Positive	No	female	
5856442	0	1	1	1	1	Positive	No	female	
5856817	1	1	1	1	0	Positive	Yes	female	
5856866	1	0	1	1	1	Positive	No	male	
5856959	1	1	0	1	1	Positive	Yes	male	

Mounting Google Drive...

Checking null values

```
data.isnull().any().sum()
```

```
0
```

Checking the number of unique values

```
data['age_60_and_above'].unique()
```

```
array(['Yes', 'No'], dtype=object)
```

```
data['corona_result'].unique()
```

```
array(['Negative', 'Positive'], dtype=object)
```

```
data['gender'].unique()
```

```
array(['female', 'male'], dtype=object)
```

```
data['test_indication'].unique()
```

```
array(['Other', 'Contact with confirmed', 'Abroad'], dtype=object)
```

Since the values are in Strings thats why we have to convert these values to integer format to apply models on it...

Applying Label Encoder on the particular columns to convert them to integer...

```
from sklearn.preprocessing import LabelEncoder
```



```
data['gender'] = k.fit_transform(data['gender'])
```

```
data['corona_result'] = k.fit_transform(data['corona_result'])
```

```
data['test_indication'] = k.fit_transform(data['test_indication'])
```

```
data['age_60_and_above'] = k.fit_transform(data['age_60_and_above'])
```

Now checking the values that are stored in place of the strings.....

```
data.head()
```

	cough	fever	sore_throat	shortness_of_breath	head_ache	corona_result	age_60_and_above	gender	test_indication
0	0	0	0	0	0	0	1	0	2
1	0	0	0	0	0	0	1	1	2
2	0	0	0	0	0	0	0	0	2
3	0	0	0	0	0	0	1	0	2
4	0	0	0	0	0	0	1	0	2

We can clearly see that all the strings are now changed to the 0 and 1 form.....

Our target is Corona Result...Hence we are going to divide the data in x and y

```
x = data.drop('corona_result' , axis=1)
```

Mounting Google Drive...

```
import sklearn
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=1/3,random_state=0)
```

```
model = LogisticRegression(solver='liblinear')

model.fit(x_train , y_train)
```

```
LogisticRegression(solver='liblinear')
```

```
model.score(x_train , y_train)
```

```
0.9084888038932832
```

```
model.score(x_test , y_test)
```

```
0.9083276052588075
```

The trainig score and testing score in LogisticRegreesion Model is 90.84% ...So to increase this score we will now chcek this with another model...

We are now going to use Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier()
```



```
RandomForestClassifier()

rfc.score(x_train , y_train)
```

```
0.9153727314068061
```

```
rfc.score(x_test , y_test)
```

```
0.9149515284618341
```

Since the Random Forest Classifier increases the score but still the score is not as good as we want...

Now we are going to use the Naive Bayes Classifier

```
from sklearn.naive_bayes import GaussianNB
```

```
gnb = GaussianNB()
```

```
gnb.fit(x_train , y_train)
```

```
GaussianNB()
```

```
gnb.score(x_train , y_train)
```

```
0.8960378518768171
```

```
gnb.score(x_test , y_test)
```

```
0.8957456315221358
```

Mounting Google Drive...

Oh my God the score is decreasing now...

Since the highest score is still 91.53% therefore we will now try XGBoost Classifier

```
import xgboost as xgb
xgboost = xgb.XGBClassifier(use_label_encoder=True, eval_metric='mlogloss')
```

```
xgboost.fit(x_train , y_train)
```

```
XGBClassifier(eval_metric='mlogloss', use_label_encoder=True)
```

```
xgboost.score(x_train , y_train)
```

```
0.9143626621913461
```

```
xgboost.score(x_test , y_test)
```

```
0.9139688416630541
```

Since after using XGBoost we are getting only 91.39% of accuracy thats why we can say that this is the highest accuracy of the data given....

We cannot achieve more accuracy on that...

Now I am going to make some charts for visualization purposes...

```
print("Prediction on Train Data_Set")
yp_train = model.predict(x_train)
data = pd.DataFrame({'Actual':y_train,'Predicted':yp_train})
data.head(10)
```

Mounting Google Drive...

Prediction on Train Data_Set

	Actual	Predicted
4152091	0	0
1697871	0	0
1114541	0	0
934297	0	0

```
import matplotlib.pyplot as plt
plt.figure(figsize=(10,10))
data["Actual"][:25,].plot.line()
data["Predicted"][:25,].plot.line()
plt.legend()
plt.show()
```


```
import seaborn as sns

print('Error Exists in training data')
fig = plt.figure()
sns.distplot((y_train - yp_train), bins = 20)
fig.suptitle('Error Terms', fontsize = 20)
plt.xlabel('Errors', fontsize = 18)
```

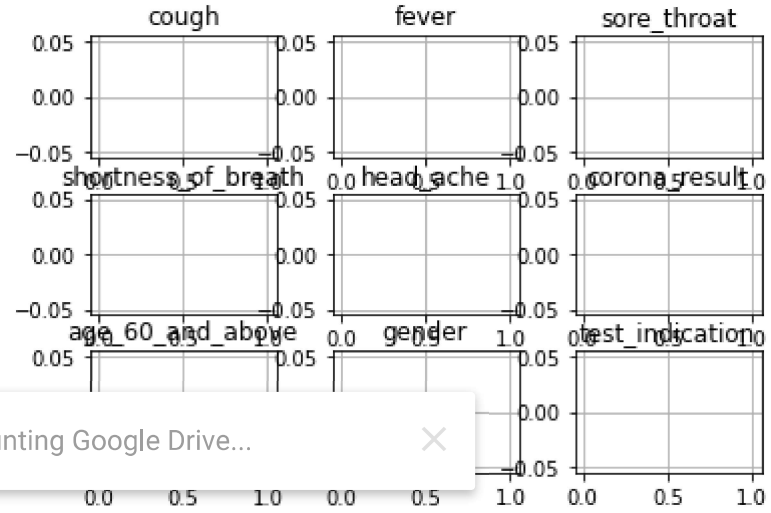
Mounting Google Drive...

```
     Error Exists in training data
```

```
df = pd.DataFrame(data, columns = ['cough', 'fever',
                                   'sore_throat', 'shortness_of_breath',
                                   'head_ache', 'corona_result' , 'age_60_and_above' , 'gender' , 'test_indication'] )
```

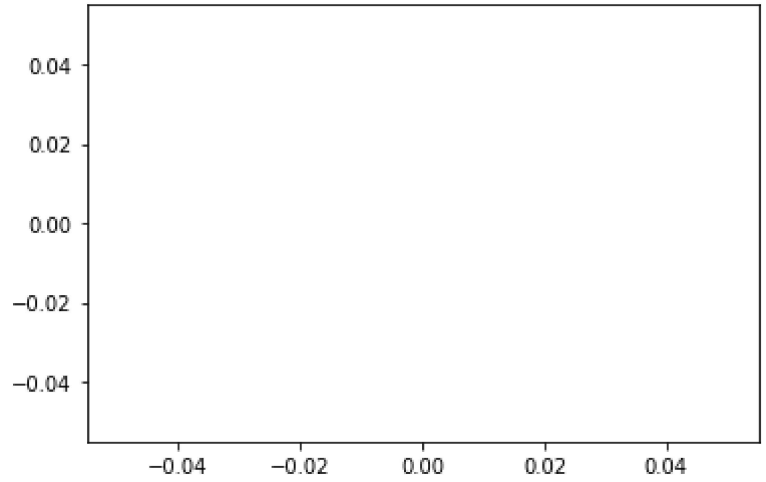
```
df.hist()
```

```
     array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f3172c93710>,
             <matplotlib.axes._subplots.AxesSubplot object at 0x7f3172c9b050>,
             <matplotlib.axes._subplots.AxesSubplot object at 0x7f3172c05310>],
            [<matplotlib.axes._subplots.AxesSubplot object at 0x7f3172bb9910>,
             <matplotlib.axes._subplots.AxesSubplot object at 0x7f3172bf0f10>,
             <matplotlib.axes._subplots.AxesSubplot object at 0x7f3172b34550>],
            [<matplotlib.axes._subplots.AxesSubplot object at 0x7f3172b6abd0>,
             <matplotlib.axes._subplots.AxesSubplot object at 0x7f3172b2d150>,
             <matplotlib.axes._subplots.AxesSubplot object at 0x7f3172b2d190>]],
           dtype=object)
```




```
plt.show()
```

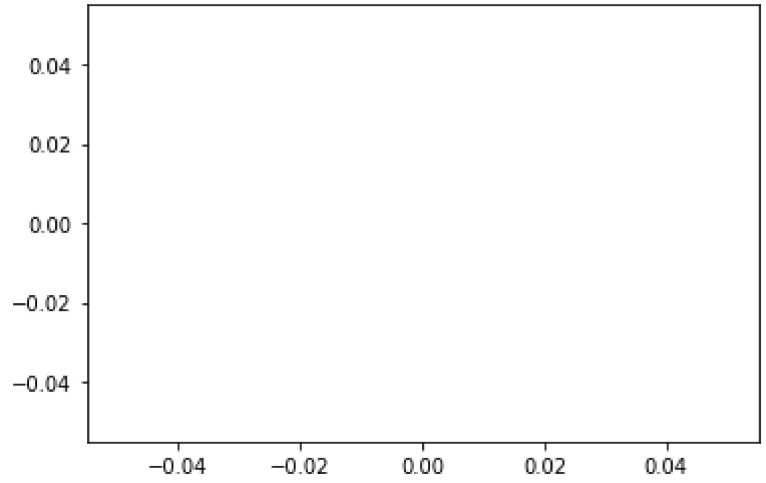
```
plt.scatter(df['corona_result'], df['gender'])
plt.show()
```




```
plt.scatter(df['corona_result'], df['age_60_and_above'])
plt.show()
```

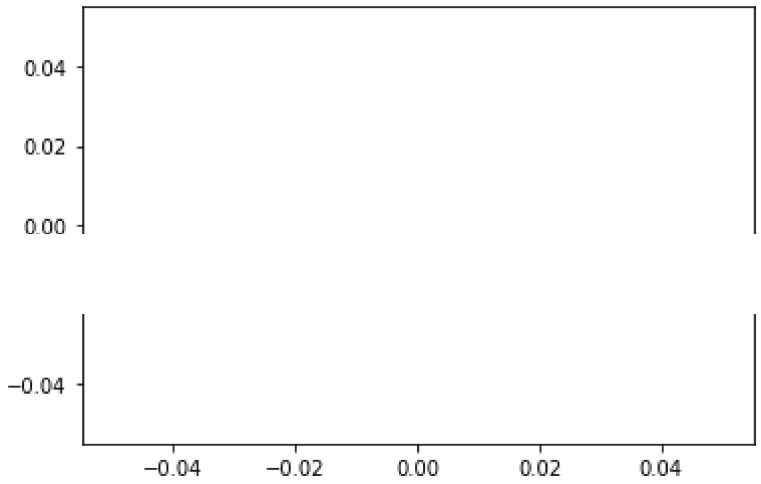

```
plt.scatter(df['cough'], df['test_indication'])
plt.show()
```


```
plt.scatter(df['cough'], df['fever'])
plt.show()
```



Mounting Google Drive...

```
], df['gender'])
```



Mounting Google Drive...