In [144...
```
test_data = test_value.copy
test_data = test_data.set_i
test_data.head()
```

Out[144...

building_id	geo_level_1_id	geo_lev
300051	17	
99355	6	
890251	22	
745817	26	
421793	17	

In [145...
```
test_data_num = test_data.s
test_data_cat = test_data.s
```

In [146...
```
test_data_num = test_data_n
```

In [147...
```
test_data_cat = pd.get_dumm
```

In [148...
```
new_test_data = pd.concat([
```

In [149...
```
scaler = StandardScaler()
scaled_test = new_test_data
data_train = df_new.copy()
scaled_col = ['geo_level_1_
              'count_floors
              'height_perce

scaled_train = data_train[s
scaler.fit(scaled_train)
feature_scaled_test = scale
feature_scaled_test = scale
scaled_test[scaled_col] = f
data_test = scaled_test.cop
```

In [150...
```
data_test.head()
```

```
0.30      0.39      4979
             2      0.60
0.88      0.71     29703
             3      0.56
0.20      0.29     17439

    accuracy
0.59     52121
   macro avg      0.58
0.46      0.47     52121
weighted avg      0.58
0.59      0.54     52121
```


```
Train F1 Score (Micro) for Lo
gistic Regression Algorithm i
s : 0.591514773599386
Test F1 Score (Micro) for Log
istic Regression Algorithm is
: 0.5943861399435928
```

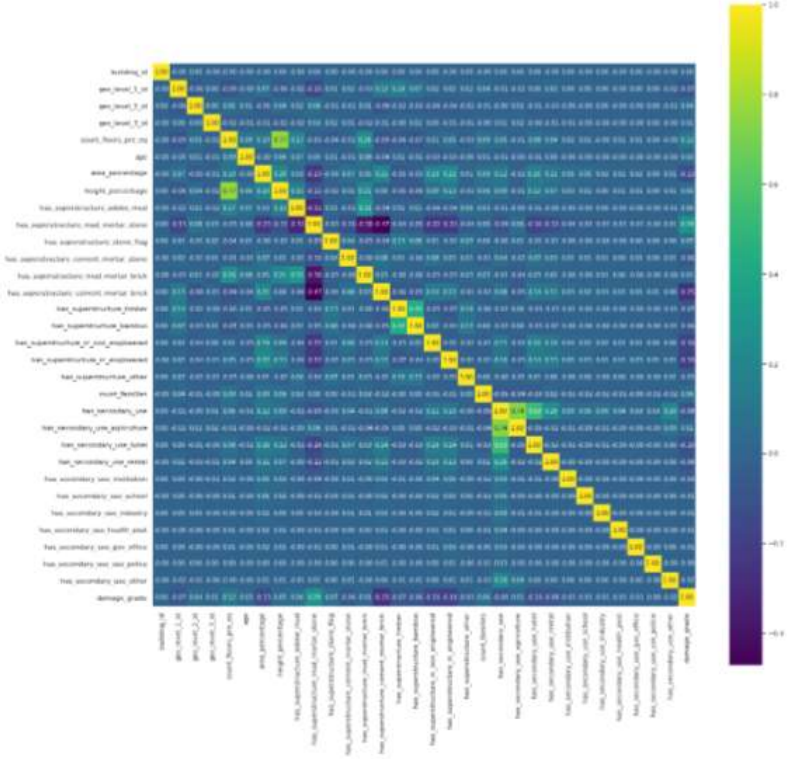
Decision Tree

In [124...

```
model_2 = DecisionTreeClass
model_2.fit(X_train, y_trai
pred_2 = model_2.predict(X_
train_pred_2 = model_2.pred

print(classification_report
#make_confusion_matrix(pred
train_score_2 = f1_score(y_
score_2 = f1_score(y_test,
print('Train F1 Score (Micr
print('Test F1 Score (Micro

with open('model_2', 'wb')
    pickle.dump(model_2, fi
```

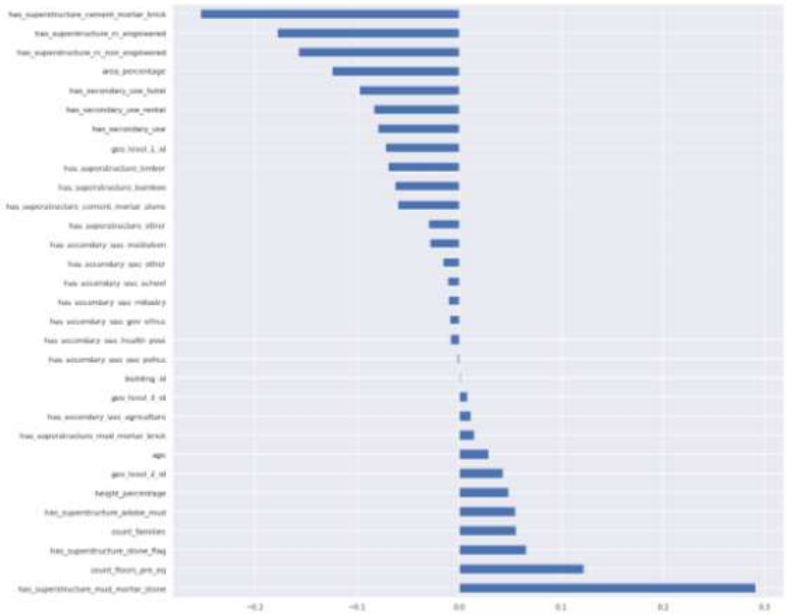
```
              precision    re
```

```
plt.show()
```



In [84]:

```
plt.figure(figsize=(16,16))

corr_matrix['damage_grade'].
plt.show()
```



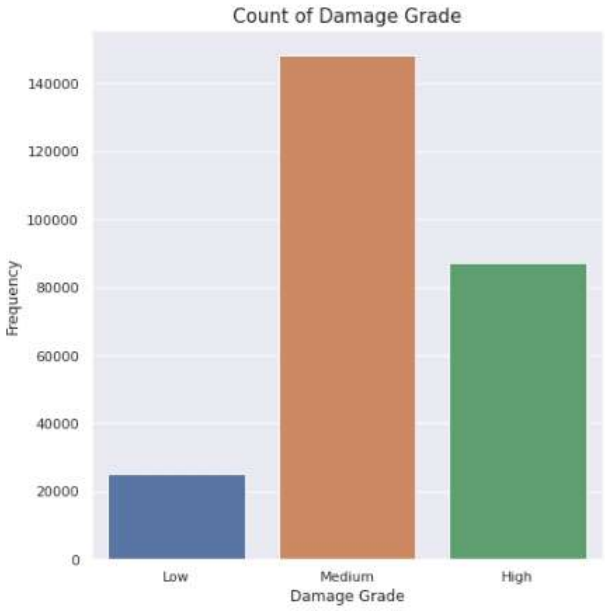
EDA (Recommendation

In [20]:
```
train['damage_grade'] = tra
train['damage_grade'] = tra
```

In [21]:
```
train['damage_grade'].uniqu
```

Out[21]:
```
array(['High', 'Medium', 'L
ow'], dtype=object)
```

In [22]:
```
plt.figure(figsize=(7,7))

sns.set_theme()
sns.countplot(train['damage
plt.xlabel('Damage Grade',
plt.ylabel('Frequency', fon
plt.title('Count of Damage
plt.tight_layout()
plt.show()
```



In [23]:
```
print('Jumlah Bangunan deng
print('Jumlah Bangunan deng
print('Jumlah Bangunan deng
print('')
print('Persentase Bangunan
print('Persentase Bangunan
print('Persentase Bangunan
```

```
Jumlah Bangunan dengan tingka
t kerusakan rendah : 25124
Jumlah Bangunan dengan tingka
```

In [12]:
```
train.isnull().sum()
```

Out[12]:
```
building_id
0
geo_level_1_id
0
geo_level_2_id
0
geo_level_3_id
0
count_floors_pre_eq
0
age
0
area_percentage
0
height_percentage
0
land_surface_condition
0
foundation_type
0
roof_type
0
ground_floor_type
0
other_floor_type
0
position
0
plan_configuration
0
has_superstructure_adobe_mu
d                 0
has_superstructure_mud_mort
ar_stone          0
has_superstructure_stone_fl
ag                0
has_superstructure_cement_m
ortar_stone       0
has_superstructure_mud_mort
ar_brick          0
has_superstructure_cement_m
ortar_brick       0
has_superstructure_timber
0
has_superstructure_bamboo
0
```

In [7]:
```
train_label.head()
```

Out[7]:

	building_id	damage_grade
0	802906	3
1	28830	2
2	94947	3
3	590882	2
4	201944	3

In [8]:
```
train_value.head()
```

Out[8]:

	building_id	geo_level_1_id	geo_
0	802906	6	
1	28830	8	
2	94947	21	
3	590882	22	
4	201944	11	

In [9]:
```
test_value.head()
```

Out[9]:

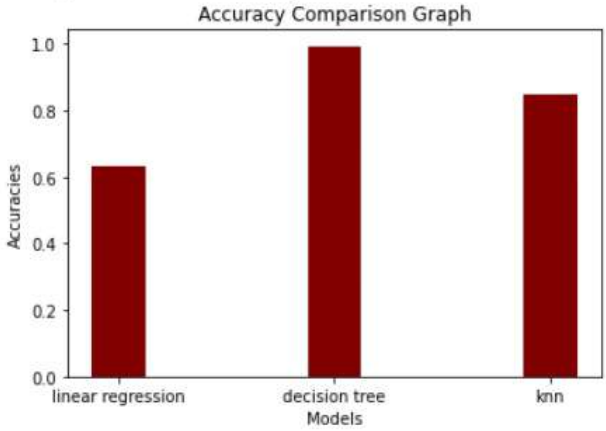
	building_id	geo_level_1_id	geo_
0	300051	17	
1	99355	6	
2	890251	22	
3	745817	26	
4	421793	17	

Description Data

Penjelasan Label

```
        width = 0.25)
plt.xlabel("Models")
plt.ylabel("Accuracies")
plt.title("Accuracy Compar
```

Out[455… Text(0.5, 1.0, 'Accuracy Comparison Graph')

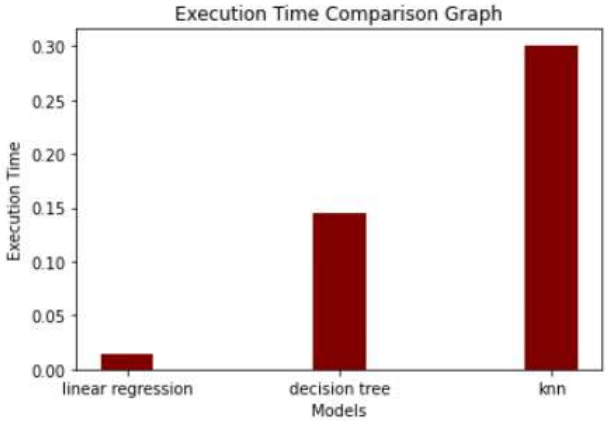


2. Execution Time

In [456…

```
times = [t1,t2,t3]
plt.bar(models, times, col
        width = 0.25)
plt.xlabel("Models")
plt.ylabel("Execution Time
plt.title("Execution Time
```

Out[456… Text(0.5, 1.0, 'Execution Time Comparison Graph')

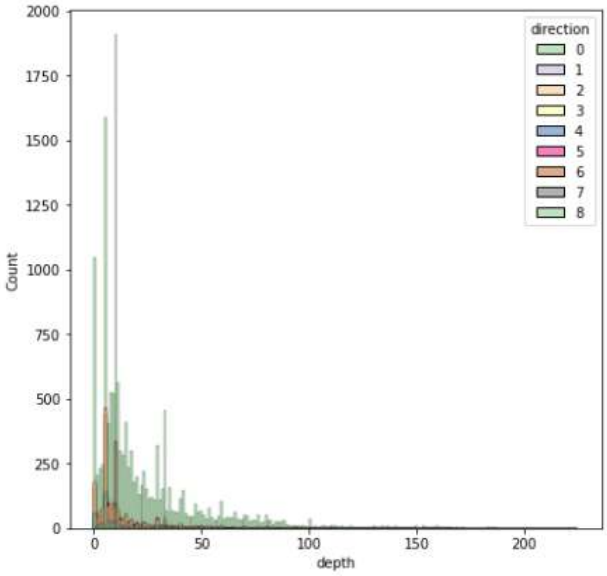


In [ ]:

```
px.scatter(df, x='richter'
```

In [412...

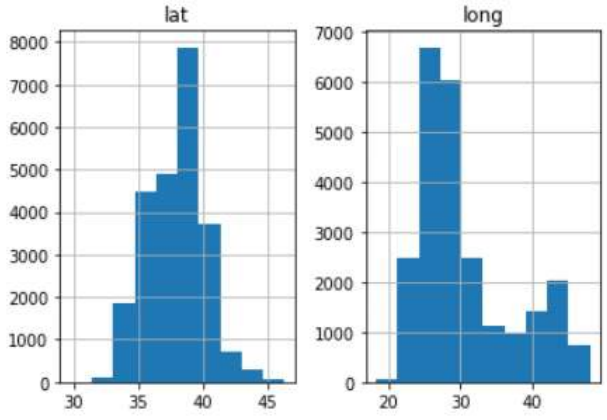
```
plt.figure(figsize=(7,7))
sns.histplot(data=df, x='d
plt.show()
```



In [413...

```
plt.figure(figsize=(7,7))
df[['lat','long']].hist()
plt.show()
```

```
<Figure size 504x504 with 0 A
xes>
```



In [414...

```
plt.figure(figsize=(15,10)
sns.countplot(df.xm)
```

```
/usr/local/lib/python3.8/dist
```

```
        padding: 5px;
        position: absolute;
        top: 10px;
        z-index: 5;
      }
    </style>
  </head>

  <body>
        <table border=0 cellpadding="1" style=
            <tr>
              <td><h1><p style="text-align:cen
            </tr>
        </table>


    <div id="map"></div>
    <script>

      var map, heatmap;

      function initMap() {
        map = new google.maps.Map(document.get
          zoom: 1.5,
          center: {lat: 0, lng: 0},
          mapTypeId: 'roadmap'
        });

        heatmap = new google.maps.visualizatio
          data: getPoints(),
          map: map
        });
      }

      function toggleHeatmap() {
        heatmap.setMap(heatmap.getMap() ? null
      }

      function changeGradient() {
        var gradient = [
          'rgba(0, 255, 255, 0)'
```

```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=devic
    <meta charset="UTF=8">
    <title>Predicting Earthquakes</title>
    <script src="//ajax.googleapis.com/ajax/li
  <link rel="stylesheet" href="//netdna.bootst
    <link rel="stylesheet" href="//netdna.boot
    <script src="//netdna.bootstrapcdn.com/boo

    <script async defer
        src="https://maps.googleapis.com/maps/
    </script>

    <style>
      /* Always set the map height explicitly
       * element that contains the map. */
      #map {
        height: 50%;
        width: 700px;
      }
      /* Optional: Makes the sample page fill
      html, body {
        height: 100%;
        margin: 0;
        padding: 0;
      }
      #floating-panel {
        position: absolute;
        top: 10px;
        left: 25%;
        z-index: 5;
        background-color: #fff;
        padding: 5px;
        border: 1px solid #999;
        text-align: center;
        font-family: 'Roboto','sans-serif';
        line-height: 30px;
        padding-left: 10px;
```

Out[24]:

	date	place	latitude
75	2020-08-18	Hawaii	19.282855
76	2020-08-18	Indonesia	-4.186166
77	2020-08-18	Nevada	38.177047
78	2020-08-18	Philippines	7.507142
79	2020-08-18	Puerto Rico	17.974638

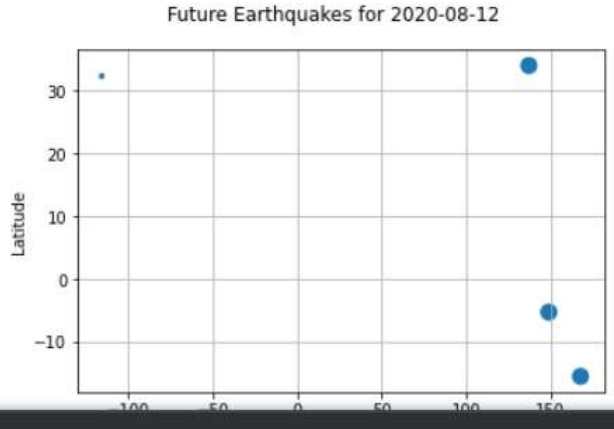
In [25]:

```
import datetime as dt
# convert date to proper fo
days = list(set([d for d in
days.sort()

# Predict NaN outcome value
predict_day=days[2]
predict_day
```

Out[25]: '2020-08-14'

In [26]:

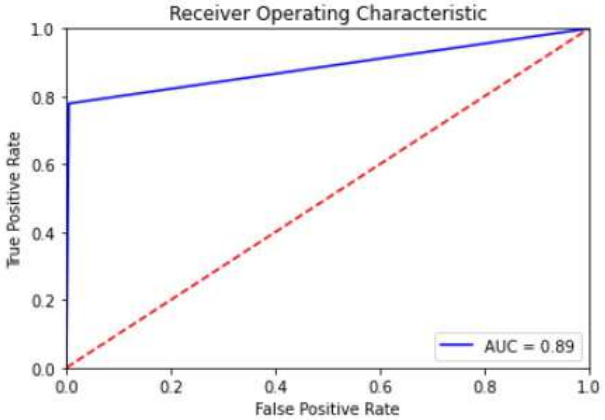
```
# place, date, lat and long
for i in range(0,7):
    live_set_tmp = live_set
    plt.scatter(live_set_tm
    plt.suptitle('Future Ea
    plt.xlabel('Longitude')
    plt.ylabel('Latitude')
    plt.grid()
    plt.show()
```



In [12]:

```
from sklearn.metrics import
from sklearn.metrics import
from sklearn.metrics import
from sklearn.metrics import
from sklearn.metrics import

print(roc_auc_score(y_test,

fpr, tpr, _ = roc_curve(y_t
roc_auc = auc(fpr, tpr)
print('AUC:', np.round(roc_

plt.title('Receiver Operati
plt.plot(fpr, tpr, 'b', lab
plt.legend(loc = 'lower rig
plt.plot([0, 1], [0, 1],'r-
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive R
plt.xlabel('False Positive
plt.show()


print("Confusion Matrix: \n
print("\nRecall 'TP/TP+FN'
```

```
0.8869826795052952
AUC: 0.887
```




```
Confusion Matrix:
 [[3595   18]
 [  42  148]]

Recall 'TP/TP+FN' =  0.778947
3684210526
```

Adaboost Random Forest Classifier

In [3]:
```
df_features.head()
```

Out[3]:

	date	depth	mag	place
0	2020-07-27	76.37	4.2	Papua New Guinea
1	2020-07-28	88.76	4.3	Papua New Guinea
2	2020-07-29	83.01	4.4	Papua New Guinea
3	2020-07-30	10.00	4.9	Papua New Guinea
4	2020-07-31	195.88	4.3	Papua New Guinea

In [4]:
```
engine =create_engine('sqli
df_predict = pd.read_sql_ta
```

In [5]:
```
# Live data to be predicted
#Hence NaN outcome that has

df_predict.head()
```

Out[5]:

	date	depth	mag	place
0	2020-08-03	140.81	4.4	Papua New Guinea
1	2020-08-04	172.48	5.4	Papua New Guinea
2	2020-08-05	94.65	5.0	Papua New Guinea
	2020-			Papua

```
  box-sizing: content-box;
}
input[type="search"]::-webkit-search-cancel
input[type="search"]::-webkit-search-decora
  -webkit-appearance: none;
}
fieldset {
  border: 1px solid #c0c0c0;
  margin: 0 2px;
  padding: 0.35em 0.625em 0.75em;
}
legend {
  border: 0;
  padding: 0;
}
textarea {
  overflow: auto;
}
optgroup {
  font-weight: bold;
}
table {
  border-collapse: collapse;
  border-spacing: 0;
}
td,
th {
  padding: 0;
}
/*! Source: https://github.com/h5bp/html5-b
@media print {
  *,
  *:before,
  *:after {
    background: transparent !important;
    box-shadow: none !important;
    text-shadow: none !important;
  }
  a,
  a:visited {
```

```
  margin: 1em 40px;
}
hr {
  box-sizing: content-box;
  height: 0;
}
pre {
  overflow: auto;
}
code,
kbd,
pre,
samp {
  font-family: monospace, monospace;
  font-size: 1em;
}
button,
input,
optgroup,
select,
textarea {
  color: inherit;
  font: inherit;
  margin: 0;
}
button {
  overflow: visible;
}
button,
select {
  text-transform: none;
}
button,
html input[type="button"],
input[type="reset"],
input[type="submit"] {
  -webkit-appearance: button;
  cursor: pointer;
}
button[disabled],
```

```
}
audio,
canvas,
progress,
video {
  display: inline-block;
  vertical-align: baseline;
}
audio:not([controls]) {
  display: none;
  height: 0;
}
[hidden],
template {
  display: none;
}
a {
  background-color: transparent;
}
a:active,
a:hover {
  outline: 0;
}
abbr[title] {
  border-bottom: 1px dotted;
}
b,
strong {
  font-weight: bold;
}
dfn {
  font-style: italic;
}
h1 {
  font-size: 2em;
  margin: 0.67em 0;
}
mark {
  background: #ff0;
  color: #000;
```

Code Blame

```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8" />

<title>ETL_USGS_EarthQuake</title>

<script src="https://cdnjs.cloudflare.com/a
<script src="https://cdnjs.cloudflare.com/a



<style type="text/css">
    /*!
*
* Twitter Bootstrap
*
*/
/*!
 * Bootstrap v3.3.7 (http://getbootstrap.co
 * Copyright 2011-2016 Twitter, Inc.
 * Licensed under MIT (https://github.com/t
 */
/*! normalize.css v3.0.3 | MIT License | gi
html {
  font-family: sans-serif;
  -ms-text-size-adjust: 100%;
  -webkit-text-size-adjust: 100%;
}
body {
  margin: 0;
}
article,
aside,
details,
figcaption,
figure,
footer
```