MAKALAH

ANALISIS PREDIKTIF KARAKTERISTIK BANGUNAN TAHAN GEMPA MENGGUNAKAN XGBOOST CLASSIFIER

Disusun oleh:

Thesion Marta Sianipar	Tech Savants
Rendika Nurhartanto Suharto	Tech Savants
Ahmad Mu'min Faisal	Tech Savants

Disusun guna berpartisipasi dalam Babak Penyisihan
JOINTS Data Competition 2023

FAKULTAS TEKNOLOGI INFORMASI DAN BISNIS
INSTITUT TEKNOLOGI TELKOM SURABAYA

2023

DAFTAR ISI

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Gempa bumi adalah salah satu bencana alam yang menimbulkan banyak kerusakan, baik dari segi korban jiwa maupun dari segi kerusakan material. Dalam artikel yang diterbitkan oleh Reuters, 10 dari 12 bencana alam dengan akibat kerusakan paling tinggi di abad ke-21 adalah gempa bumi. Bahkan, gempa Haiti pada tahun 2010 mengakibatkan tewasnya kira-kira 316.000 jiwa serta rusaknya 80.000 bangunan di kota Port-au-Prince dan sekitarnya [1]. Dua kerugian tersebut memiliki relasi, dimana sebagian besar korban berada di bangunan yang ambruk pada saat gempa [2]. Selain itu, jatuhnya korban juga berkorelasi dengan kerusakan ruang interior saat terjadinya gempa [3].

Berangkat dari hal tersebut, maka jatuhnya korban jiwa pada saat gempa dapat diminimalisir dengan cara melakukan analisis terhadap karakteristik-karakteristik bangunan yang memiliki tingkat kerusakan rendah. Kemudian, kesimpulan yang didapat dari analisis tersebut dapat dijadikan standar untuk pendirian bangunan untuk antisipasi kerusakan yang lebih besar apabila terjadi gempa lagi di kemudian hari. Model yang sama juga dapat digunakan untuk memprediksi tingkat kerusakan bangunan yang sudah ada sebagai apabila terjadi gempa dengan kekuatan dan kedalaman yang sama di tempat tersebut.

1.2 Rumusan Masalah

1. Bagaimana karakteristik-karakteristik bangunan yang memiliki tingkat kerusakan rendah setelah terkena gempa?
2. Bagaimana memilih model yang tepat untuk memprediksi tingkat kerusakan bangunan berdasarkan fitur-fiturnya apabila terjadi gempa dengan kekuatan dan kedalaman yang sama di kemudian hari?
3. Berapa akurasi dari hasil prediksi yang didapatkan dari model yang telah dipilih?

1.3 Tujuan

1. Mencari karakteristik-karakteristik bangunan yang memiliki tingkat kerusakan rendah setelah terkena gempa.
2. Memilih model yang tepat untuk memprediksi tingkat kerusakan bangunan berdasarkan fitur-fiturnya apabila terjadi gempa dengan kekuatan dan kedalaman yang sama di kemudian hari.
3. Mengetahui akurasi dari hasil prediksi yang didapatkan dari model yang telah dipilih.

1.4 Manfaat

1. Manfaat bagi masyarakat adalah hasil analisis prediktif ini dapat digunakan untuk perbaikan standar pendirian bangunan yang lebih tahan gempa berdasarkan karakteristik-karakteristiknya.

2. Manfaat bagi ilmu pengetahuan adalah makalah ini dapat digunakan sebagai referensi bagi para peneliti lain dan sebagai sumbangsih untuk ilmu pengetahuan dalam pencarian metode untuk antisipasi kerusakan bencana alam.

BAB II
METODE

1.5 Perangkat Lunak

Perangkat-perangkat lunak yang digunakan dalam mengolah dan memprediksi dataset adalah sebagai berikut:

Bahasa Pemrograman	Python v3.9.16
Package Manager	Conda v22.11.1
Package	jupyter, pandas, numpy, scikit-learn, xgboost, pickle, matplotlib, seaborn, cudatoolkit
Perangkat Lunak lainnya	Microsoft Excel

1.6 Dataset

Data awal yang akan diolah terdiri dari 2 dataset berekstensi csv (*comma separated values*), yaitu train set yang memiliki nama file `train.csv` dan test set yang memiliki nama file `test.csv`. Tabel train set terdiri dari 25 kolom dan 722814 baris. Dataset ini masih belum ternormalisasi dan terdapat nilai hilang (*missing values*). Sedangkan, tabel test set terdiri dari 24 kolom dan 242082 baris. Tabel ini memiliki kolom-kolom yang sama dengan tabel train set, kecuali kolom `damage_grade` yang tidak ada pada tabel test set. Seperti pada tabel sebelumnya, data-data pada tabel ini juga masih belum ternormalisasi, namun tidak terdapat nilai hilang.

```
train_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 722815 entries, 0 to 722814
Data columns (total 25 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   Unnamed: 0                      722815 non-null  int64
 1   floors_before_eq (total)        390009 non-null  object
 2   old_building                    483611 non-null  float64
 3   plinth_area (ft^2)              301607 non-null  object
 4   height_before_eq (ft)           390009 non-null  float64
 5   land_surface_condition          421209 non-null  object
 6   type_of_foundation              483611 non-null  object
 7   type_of_roof                    301607 non-null  object
 8   type_of_ground_floor            390009 non-null  object
 9   type_of_other_floor             421209 non-null  object
 10  position                        410809 non-null  object
 11  building_plan_configuration     421209 non-null  object
 12  technical_solution_proposed     46801 non-null   object
 13  legal_ownership_status          598013 non-null  object
 14  has_secondary_use               525211 non-null  float64
 15  type_of_reinforcement_concrete  431609 non-null  float64
 16  residential_type                452411 non-null  object
 17  no_family_residing              577213 non-null  object
 18  public_place_type               722815 non-null  object
 19  industrial_use_type             608413 non-null  object
 20  govermental_use_type            473211 non-null  object
 21  flexible_superstructure         660415 non-null  object
 22  wall_binding                    660415 non-null  float64
 23  wall_material                   494011 non-null  float64
 24  damage_grade                    722815 non-null  float64
dtypes: float64(7), int64(1), object(17)
memory usage: 137.9+ MB
```

(a) train.csv

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 242082 entries, 0 to 242081
Data columns (total 24 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   id                              242082 non-null  int64
 1   floors_before_eq (total)        242082 non-null  object
 2   old_building                    242082 non-null  int64
 3   plinth_area (ft^2)              242082 non-null  object
 4   height_before_eq (ft)           242082 non-null  int64
 5   land_surface_condition          242082 non-null  object
 6   type_of_foundation              242082 non-null  object
 7   type_of_roof                    242082 non-null  object
 8   type_of_ground_floor            242082 non-null  object
 9   type_of_other_floor             242082 non-null  object
 10  position                        242082 non-null  object
 11  building_plan_configuration     242082 non-null  object
 12  technical_solution_proposed     242082 non-null  object
 13  legal_ownership_status          242082 non-null  object
 14  has_secondary_use               242082 non-null  float64
 15  type_of_reinforcement_concrete  242082 non-null  int64
 16  residential_type                242082 non-null  object
 17  no_family_residing              242082 non-null  object
 18  public_place_type               242082 non-null  object
 19  industrial_use_type             242082 non-null  object
 20  govermental_use_type            242082 non-null  object
 21  flexible_superstructure         242082 non-null  object
 22  wall_binding                    242082 non-null  int64
 23  wall_material                   242082 non-null  int64
dtypes: float64(1), int64(6), object(17)
memory usage: 44.3+ MB
```

(b) test.csv

Gambar 1: Informasi mengenai tabel train set dan test set

Pada tahap selanjutnya, train set (`train.csv`) akan dipergunakan untuk melakukan analisis data eksplorasi (EDA), pelatihan model prediksi, sekaligus evaluasi model melalui pemisahan

dataset (*train-test splitting*).

1.7 Algoritma

1.7.1 *Frequent Category Imputation*

Frequent Category Imputation/Mode Imputation merupakan algoritma imputasi untuk mengisi nilai dengan menggunakan nilai yang paling banyak muncul (modus). Algoritma imputasi ini akan digunakan untuk melakukan imputasi pada kolom-kolom kategorikal yang memiliki persentasi nilai hilang kurang dari 10%.

1.7.2 *K-Nearest Neighbour Imputation*

K-Nearest Neighbour Imputation merupakan salah satu algoritma imputasi fitur pada training set yang memanfaatkan algoritma *K-Nearest Neighbour*. Algoritma imputasi ini menggunakan rataan (*mean*) jarak Euclidean dari k tetangga terdekat dari nilai yang hilang untuk melakukan imputasi terhadap nilainya yang hilang [4]. Jarak Euclidean dari dua titik (x, y) dan (a, b) pada sebuah koordinat Cartesius dapat dihitung menggunakan:

$$dist((x, y), (a, b)) = \sqrt{(x - a)^2 + (y - b)^2}$$

1.7.3 *Gradient Boosting Machines*

Gradient Boosting Machines merupakan salah satu algoritma *ensemble tree* yang dapat digunakan untuk masalah regresi maupun klasifikasi. Algoritm ini dimulai dengan menghasilkan pohon klasifikasi awal dan terus menyesuaikan pohon baru melalui minimalisasi fungsi kerugian (*loss function*) [5].

1.8 *Data Cleaning*

Pembersihan data diawali dengan menghapus baris-baris yang memiliki banyak nilai hilang. Sebelum melakukan penghapusan, maka perlu ditentukan batasan berapa banyak kolom minimal dalam suatu baris agar bisa dihapuskan.

```
[11]: count_unique_nan = dict(train_data.isnull().sum(axis=1).value_counts())
      for key in sorted(count_unique_nan):
          print(f"{key}\t {count_unique_nan[key]}")
```

```
0       46801
1       254806
3       88402
6       20800
7       10400
10      10400
11      20802
12      20800
13      10400
15      10400
16      31200
17      52002
18      20800
19      10400
20      52002
22      62400
```

Gambar 2: Melakukan pemeriksaan setiap kemungkinan kekosongan kolom dan banyak barisnya.

Dilihat dari gambar di atas, maka dapat ditentukan bahwa setiap baris yang memiliki sebanyak lebih dari 30% kolom yang kosong dapat dihapus.

```
[12]: list_of_deleted_rows = []
      for i in range(len(train_data)):
          if train_data.loc[[i]].isna().sum().sum() > 7:
              list_of_deleted_rows.append(i)

[13]: list_of_deleted_rows[:10]

[13]: [5, 12, 13, 14, 29, 32, 36, 37, 44, 47]

[14]: train_data.drop(list_of_deleted_rows, axis=0, inplace=True)
```

Gambar 3: Menghapus baris yang memiliki lebih dari 30% kolom yang kosong.

Setelah melakukan penghapusan baris, maka perlu diperiksa ulang berapa banyak nilai kosong yang ada pada setiap kolom. Pada pemeriksaan ini, didapatkan bahwa kolom `technical_solution_proposed` sebagian besar datanya kosong sehingga perlu dihapus. Selain itu, terdapat kolom `plinth_area (ft2)` dan `type_of_roof` yang memiliki jumlah nilai kosong yang cukup signifikan (28,39%), serta beberapa kolom lain yang memiliki nilai kosong di bawah 10%. Kolom-kolom tersebut akan dinormalisasi (untuk kolom yang belum ternormalisasi) terlebih dahulu sebelum dilakukan imputasi.

```
[17]: train_data.isnull().sum()*100/len(train_data)

[17]: floors_before_eq (total)          7.407249
      old_building                      0.000000
      plinth_area (ft^2)               28.394930
      height_before_eq (ft)             7.407249
      land_surface_condition            0.000000
      type_of_foundation                0.000000
      type_of_roof                     28.394930
      type_of_ground_floor              7.407249
      type_of_other_floor               0.000000
      position                          2.469083
      building_plan_configuration       0.000000
      technical_solution_proposed      88.888889
      legal_ownership_status            0.000000
      has_secondary_use                 0.000000
      type_of_reinforcement_concrete    0.000000
      residential_type                  0.000000
      no_family_residing                0.000000
      public_place_type                 0.000000
      industrial_use_type               0.000000
      govermental_use_type              0.000000
      flexible_superstructure           0.000000
      wall_binding                      0.000000
      wall_material                     0.000000
      damage_grade                      0.000000
      dtype: float64
```

Gambar 4: Persentase nilai kosong pada setiap kolom setelah melakukan penghapusan baris.

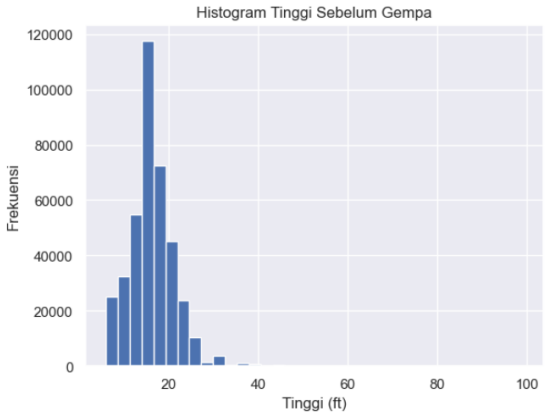
Untuk kolom yang sudah ternormalisasi seperti kolom `industrial_use_type` dan memiliki nilai kosong sebanyak kurang dari 10%, maka dapat langsung dilakukan imputasi berdasarkan nilai paling banyak muncul (*frequent category imputation*). Namun, untuk kolom string yang belum normal seperti kolom `flooars_before_eq (total)`, maka perlu dinormalisasi dulu dengan cara mengubahnya menjadi bentuk numerik sesuai dengan substring yang terkandung di dalamnya. Sebagai contoh, string ”one floor” dapat diubah menjadi 1 karena mengandung substring ”one” atau string ”fifth” dapat diubah menjadi 5 karena ”fifth” merupakan substring dari data tersebut.

```
[19]: train_data['floors_before_eq (total)'] = train_data['floors_before_eq (total)'].astype(str).str.lower()

[20]: for i in range(len(train_data)):
          if ("1" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("one" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("f
              train_data.loc[i, 'floors_before_eq (total)'] = 1
          if ("2" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("two" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("s
              train_data.loc[i, 'floors_before_eq (total)'] = 2
          if ("3" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("three" in str(train_data.loc[i, 'floors_before_eq (total)'])) or (
              train_data.loc[i, 'floors_before_eq (total)'] = 3
          if ("4" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("four" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("
              train_data.loc[i, 'floors_before_eq (total)'] = 4
          if ("5" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("five" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("
              train_data.loc[i, 'floors_before_eq (total)'] = 5

          if ("6" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("six" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("s
              train_data.loc[i, 'floors_before_eq (total)'] = 6
          if ("7" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("seven" in str(train_data.loc[i, 'floors_before_eq (total)'])) or (
              train_data.loc[i, 'floors_before_eq (total)'] = 7
          if ("8" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("eight" in str(train_data.loc[i, 'floors_before_eq (total)'])) or (
              train_data.loc[i, 'floors_before_eq (total)'] = 8
          if ("9" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("nine" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("
              train_data.loc[i, 'floors_before_eq (total)'] = 9
          if ("10" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("ten" in str(train_data.loc[i, 'floors_before_eq (total)'])) or ("
              train_data.loc[i, 'floors_before_eq (total)'] = 10
```

Gambar 5: Mengatasi salah satu kolom bertipe data string yang belum ternormalisasi.

Untuk kolom numerik yang memiliki nilai hilang seperti kolom `height_before_eq (ft)`, maka perlu dilakukan pemeriksaan terlebih dahulu mengenai distribusi datanya. Setelah diperiksa, didapatkan bahwa distribusinya berbentuk *skewness* positif (data tidak berdistribusi normal) sehingga perlu dilakukan imputasi menggunakan median (*median imputation*) karena mean sudah tidak bisa lagi merepresentasikan pusat distribusi datanya.

```
[70]: plt.hist(train_data['height_before_eq (ft)'], bins=35)
      plt.title('Histogram Tinggi Sebelum Gempa')
      plt.xlabel('Tinggi (ft)')
      plt.ylabel('Frekuensi')
      plt.show()
```




```
[71]: skewness = train_data['height_before_eq (ft)'].skew()
      print("Skewness kolom 'height_before_eq':", skewness)

      Skewness kolom 'height_before_eq': 2.6022487047328484

[72]: train_data['height_before_eq (ft)'] = train_data['height_before_eq (ft)'].fillna(train_data['height_before_eq (ft)'].median())
```

Gambar 6: Imputasi berdasarkan median untuk data yang tidak berdistribusi normal.

Pada tahap ini, masih ada kolom yang memiliki nilai yang hilang, yaitu kolom `plinth_area (ft$\hat{2}$)` yang akan diimputasi pada tahap berikutnya, yaitu tahap *feature engineering*.

1.9 *Feature Engineering*

Sebelum melakukan *feature engineering*, matriks fitur (X) perlu dipisahkan dengan target-nya (y) karena dalam *feature engineering* hanya berfokus pada pengolahan fitur agar dapat dijadikan training set yang cocok dalam pembuatan model prediksi. Kemudian, dalam matrix fitur X, dilakukan pemisahan lagi antara kolom-kolom numerik dan kategorikal karena akan menggunakan metode *encoding* yang berbeda. Perlu diperhatikan bahwa kolom `plinth_area (ft$\hat{2}$)` masih bertipe *object* karena masih mengandung nilai kosong.

```
[241]: X = train_data.drop('damage_grade', axis=1)
       y = train_data['damage_grade']

[242]: column_type_dict = dict(X.dtypes)
       categorical_features = []
       numerical_features = []
       for key, value in column_type_dict.items():
           if str(value) == "category":
               categorical_features.append(str(key))
           else:
               numerical_features.append(str(key))

       categorical_features, numerical_features
```

```
[242]: (['land_surface_condition',
          'type_of_foundation',
          'type_of_roof',
          'type_of_ground_floor',
          'type_of_other_floor',
          'position',
          'building_plan_configuration',
          'legal_ownership_status',
          'residential_type',
          'no_family_residing',
          'public_place_type',
          'industrial_use_type',
          'governmental_use_type',
          'flexible_superstructure'],
         ['floors_before_eq (total)',
          'old_building',
          'plinth_area (ft^2)',
          'height_before_eq (ft)',
          'has_secondary_use',
```

Gambar 7: Pemisahan matriks fitur X dan target y, dilanjutkan pemisahan kolom-kolom numerik dengan kategorikal pada matriks fitur X.

Setelah pemisahan tersebut, maka kolom-kolom yang berjenis kategorikal dilakukan *label encoding*. Encoding jenis ini dilakukan untuk membuat dimensi fitur tetap sama untuk menghemat biaya komputasi.

```
[243]: from sklearn.preprocessing import LabelEncoder
       encoder = LabelEncoder()

[244]: for col in categorical_features :
           X[col] = encoder.fit_transform(X[col])

[245]: # concat
       X_encoded = pd.concat([X[numerical_features].copy(), X[categorical_features]], axis=1)
       X_encoded
```

[245]:

	floors_before_eq (total)	old_building	plinth_area (ft^2)	height_before_eq (ft)	has_secondary_use	type_of_reinforcement_concrete	wall_binding	wall_material	land_surface_condition	type_of_fou
0	2	1	256	22	0	0	0	0	0	
1	3	3	985	18	0	0	5	2	0	
2	2	7	NaN	14	0	0	5	2	0	
3	2	18	185	15	0	0	5	2	0	
4	2	22	290	17	0	0	5	2	0	
...	...	...	...	...	...	...	...	...	...	
421204	2	32	NaN	12	0	0	1	0	2	
421205	3	45	NaN	18	0	0	5	2	1	
421206	3	72	NaN	21	0	0	5	1	0	
421207	1	22	NaN	6	0	0	5	2	0	

Gambar 8: Label encoding untuk kolom-kolom berjenis kategorikal.

BAB III
ANALISIS DATA EKSPLORASI DAN MODEL PREDIKSI

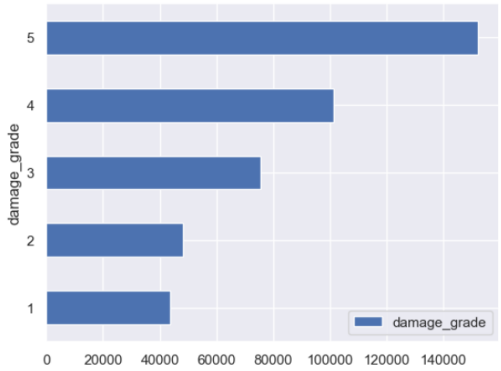
1.10 Analisis Data Eksplorasi

Analisis Data Eksplorasi dilakukan sebelum melakukan *feature engineering* karena label-label asli masih diperlukan dalam melakukan analisis.

1.10.1 Analisis Univariat

Dalam analisis ini, didapatkan bahwa pada saat gempa, sebagian besar dari bangunan yang rusak mengalami tingkat kerusakan yang lebih parah.

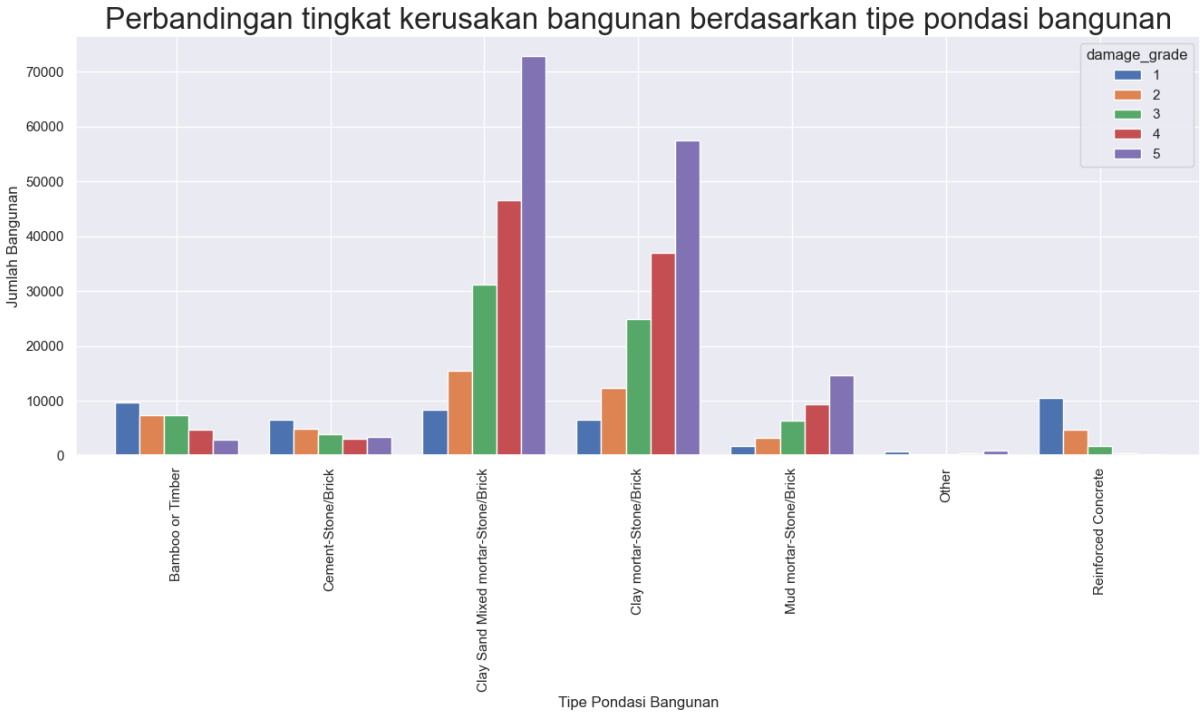
```
[220]: train_data.groupby('damage_grade')[['damage_grade']].count().plot(kind='barh')
[220]: <Axes: ylabel='damage_grade'>
```



Gambar 9: Banyak bangunan yang rusak pada setiap tingkat kerusakan.

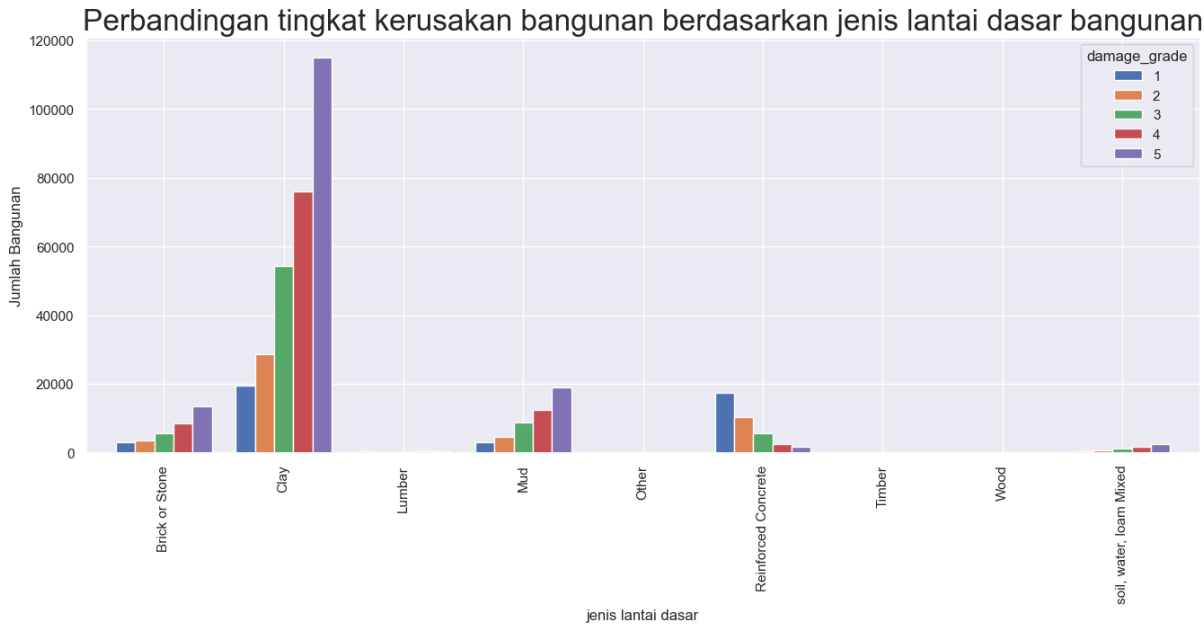
1.10.2 Analisis Bivariat

Kemudian, ketika melakukan analisis mengenai kolom `type_of_foundation` dengan `damage_grade`, informasi menarik yang dapat diambil dari data tersebut adalah bahwa `type_of_foundation` yang menggunakan Clay Sand Mixed mortar-Stone/Brick memiliki jumlah bangunan yang mengalami `damage_grade` 5 yang paling banyak dibandingkan dengan `type_of_foundation` lainnya. Selain itu, bangunan dengan foundation Reinforced Concrete memiliki jumlah bangunan yang mengalami `damage_grade` 1 yang paling banyak dibandingkan dengan `type_of_foundation` lainnya. Selain itu, bangunan dengan foundation Mud mortar-Stone/Brick memiliki jumlah bangunan yang mengalami `damage_grade` 4 dan 5 yang paling sedikit dibandingkan dengan `type_of_foundation` lainnya.

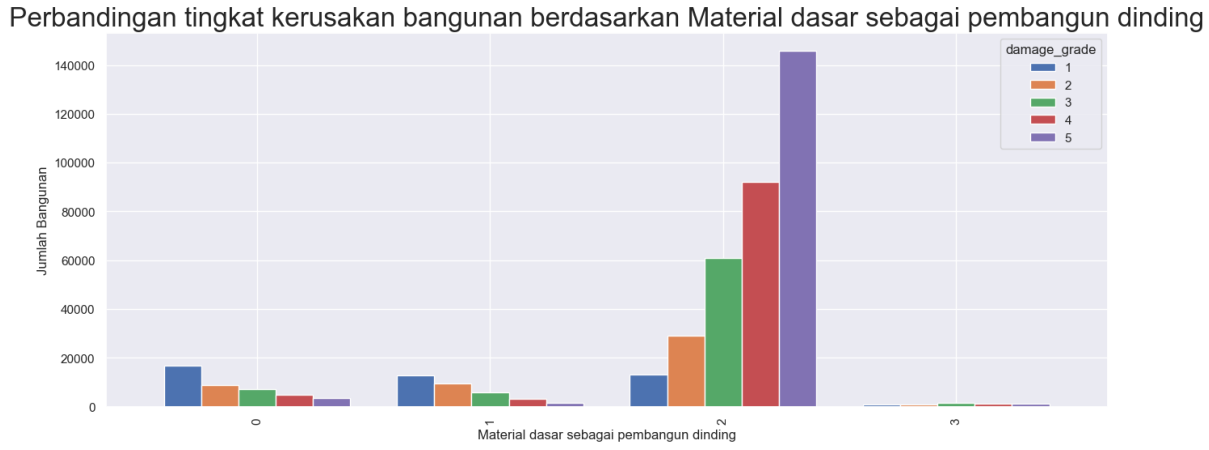


Informasi menarik yang dapat diambil data ini adalah bahwa bangunan yang memiliki tingkat kerusakan paling tinggi (damage grade tingkat 5, 4, dan 3) didominasi oleh bangunan yang menggunakan tipe pondasi Clay Sand Mixed mortar-Stone/Brick, Clay mortar-Stone/Brick, Mud

mortar-Stone/Brick. Sedangkan, bangunan yang menggunakan tipe pondasi Reinforced Concrete, Bamboo of Timber, dan Cement-Stone/Brick lebih banyak mengalami kerusakan tingkat rendah (1, 2, 3) dibandingkan tingkat kerusakan lainnya. Dalam hal ini, dapat disimpulkan bahwa bangunan-bangunan yang menggunakan tipe pondasi Reinforced Concrete, Bamboo of Timber, dan Cement-Stone/Brick lebih tahan gempa karena memiliki tingkat kerusakan yang lebih rendah



Selain itu, pada damage grade 1, dinding dengan material 0 (mud brick/stone) memiliki jumlah bangunan yang paling banyak mengalami kerusakan dibandingkan dengan material dinding lainnya. Sedangkan pada damage grade 2, dinding dengan material 2 (concrete) memiliki jumlah bangunan yang paling banyak mengalami kerusakan dibandingkan dengan material dinding lainnya.



1.11 Model Prediksi

Dalam pembuatan model prediksi, semua fitur pada test set harus dipastikan identik dengan fitur yang ada pada train set. Sehingga, *preprocessing* yang sama perlu dilakukan untuk test set. Setelah dilakukan preprocessing, maka objek model XGBoost Classifier dibuat dengan opsi memilih GPU untuk akselerasi komputasi dan parameter-parameter yang akan di-*tuning* menggunakan GridSearchCV.

```
[271]: params = {
           'gpu_id' : [1],
           'tree_method' : ['gpu_hist'],
           'predictor' : ["gpu_predictor"],
       #    "booster" : ["gbtree","gblinear","dart"],
           #'eta':np.arange(0,1,0.1),
           "gamma": range(0,2),
           "max_depth": range(2,4),
           'min_child_weight':range(1,5),
           #"subsample": np.arange(0.5,1,0.1),
           #"colsample_bytree": np.arange(0.5,1,0.1),
           #"learning_rate": [0.1, 0.01, 0.05],
           #"reg_lambda": [0, 1, 10],
           #"scale_pos_weight": [1, 3, 5],
       }

       xgb_cl = xgb.XGBClassifier(params)

       C:\Users\rendi\anaconda3\lib\site-packages\xgboost\core.py:617: FutureWarning: Pass `objective` as keyword args.
         warnings.warn(msg, FutureWarning)
```

Gambar 10: Instansiasi objek XGB Classifier dengan parameter yang mungkin.

Kemudian, *hyperparameter tuning* dilakukan menggunakan GridSearchCV untuk menelusuri parameter-parameter yang ditentukan sehingga dapat menentukan parameter terbaik dari penelurusan tersebut.

```
4.11. Membuat objek gridsearchCV

[272]: grid_search = GridSearchCV(xgb_cl, params, n_jobs=-1, cv=3,  scoring = "accuracy")

4.12. melakukan training data dengan gridsearchCV

[273]: import time

       Start = time.time()

       grid_search.fit(X, y)
       score_df = pd.DataFrame(grid_search.cv_results_)

       Stop = time.time()

       RentangWaktu = Stop - Start
       print(RentangWaktu)

       [23:49:29] WARNING: C:\buildkite-agent\builds\buildkite-windows-cpu-autoscaling-group-i-07593ffd91cd9da33-1\xgboost\xgboost-ci-windows\src\learner.cc:347: Only 1 GPUs are visible, setting `gpu_id` to 0
       89.07099986076355

4.13. cek untuk mean test 5 tertinggi

[274]: score_df.nlargest(5,"mean_test_score")

[274]:
```

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_gamma	param_gpu_id	param_max_depth	param_min_child_weight	param_predictor	param_tree_method	params	split0_test_score	split1_test_score	split2_t
4	13.604999	0.102948	0.459666	0.040714	0	1	3	1	gpu_predictor	gpu_hist	{'gamma': 0, 'gpu_id': 1, 'max_depth': 3, 'min...	0.444834	0.445247	

Gambar 11: *hyperparameter tuning* menggunakan GridSearchCV.

Setelah itu, parameter terbaik setelah penelusuran GridSearchCV diambil untuk melakukan pelatihan model XGB Classifier. Setelah model dilatih, prediksi dilakukan dari data test set (`test.csv`) kemudian dieksport menjadi `sample_submission.csv` untuk dilakukan evaluasi di Kaggle menggunakan metrics *f1-score*.

```
4.14. Mengambil best params-nya

[275]: grid_search.best_params_

[275]: {'gamma': 0,
        'gpu_id': 1,
        'max_depth': 3,
        'min_child_weight': 1,
        'predictor': 'gpu_predictor',
        'tree_method': 'gpu_hist'}

[276]: final_Xgb_cl = grid_search.best_estimator_

       preds = final_Xgb_cl.predict(test_data)

[277]: preds.shape

[277]: (242082,)

[278]: len(preds)

[278]: 242082

4.15. konvert hasil predict dari label encoding menjadi format seperti damage_grade (y/target)

[2...  preds = preds.tolist()

       for i in range (len(preds)):
           preds[i] = preds[i] + 1

       preds = pd.DataFrame(pd.Series(preds)).reset_index()
       preds = preds.rename(columns = {"index":"id",0:"damage_grade"})
       preds.to_csv("data/sample_submission.csv")
```

Gambar 12: Pelatihan dan prediksi model XGB Classifier.

BAB IV
KESIMPULAN

4.1 Hasil dan Pembahasan

Melalui analisis data eksplorasi, didapatkan bahwa lebih banyak bangunan yang mengalami tingkat kerusakan yang berat daripada tingkat kerusakan yang lebih rendah. Di antara banyaknya bangunan tersebut, tentu saja terdapat **struktur-struktur yang mempengaruhi sedemikian sehingga suatu bangunan dengan struktur tertentu lebih tahan kerusakan saat gempa** daripada bangunan lainnya. Struktur-struktur tersebut di antaranya adalah:

1. tipe pondasi: *reinforced concrete*, bambi atau timber, dan *cement-stone/brick*.
2. jenis lantai dasar bangunan: *reinforced concrete*.
3. jenis beton bertulang: tipe 1
4. bahan perekat dinding: bahan tipe 0 dan tipe 2.
5. material dinding: tipe 0 dan tipe 1.

Kemudian, evaluasi model XGB Classifier dalam Kaggle Competition menggunakan 65% dari test data dan metrics *f1-score* menghasilkan akurasi 0.31614.

4.2 Kesimpulan

Melalui makalah ini, dapat disimpulkan bahwa terdapat struktur-struktur yang membuat bangunan lebih tahan gempa daripada bangunan dengan struktur-struktur lainnya. Sehingga, hasil dari analsis data eksplorasi dapat digunakan untuk menentukan struktur bangunan yang lebih tahan gempa pada saat membangun bangunan untuk antisipasi gempa dengan kekuatan dan kedalaman yang sama di kemudian hari. Namun, pembuatan model prediksi menggunakan XGB Classifier masih belum mencapai hasil dan performa yang maksimal. Sehingga, disarankan untuk membuat model prediksi menggunakan algoritma lain dan *hyperparameter tuning* yang lebih baik lagi.

DAFTAR PUSTAKA

[1] Reuters, "Factbox: Turkey earthquake and some of the worst natural disasters of this century," 2023. [Online]. Available: https://www.reuters.com/business/environment/turkey-quake-other-major-natural-disasters-this-century-2023-02-09/

[2] L. Hengjian, M. Kohiyama, K. Horie, N. Maki, H. Hayashi, and S. Tanaka, "Building damage and casualties after an earthquake," *Natural Hazards*, vol. 29, pp. 387–403, 2003. [Online]. Available: http://www.jstor.org/stable/26058899

[3] F. Aiko, S. Robin, O. Yutaka, and S. Emily, "Analytical study on vulnerability functions for casualty estimation in the collapse of adobe buildings induced by earthquake," *Bulletin of Earthquake Engineering*, vol. 8, no. 2, pp. 451–479, 2010. [Online]. Available: https://doi.org/10.1007/s10518-009-9156-z

[4] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman, "Missing value estimation methods for dna microarrays," *Bioinformatics*, vol. 17, no. 6, pp. 520–525, 2001.

[5] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Front Neurorobots*, vol. 7, no. 21, pp. 1 – 21, 2013.

LAMPIRAN

1.1 Biodata Ketua Tim

1.	Nama Lengkap	Thesion Marta Sianipar
2.	Jenis Kelamin	Perempuan
3.	Program Studi	S1 Sains Data
4.	NIM	1206210004
5.	Perguruan Tinggi	Institut Teknologi Telkom Surabaya
6.	Tempat, Tanggal Lahir	Pematang Sianipar, 4 Desember 2002
7.	E-mail	thesion.jambi2018@gmail.com
8.	Nomor Telepon/HP	082198287359

1.2 Biodata Anggota 1

1.	Nama Lengkap	Rendika Nurhartanto Suharto
2.	Jenis Kelamin	Laki-laki
3.	Program Studi	S1 Sains Data
4.	Nama NIM	1206210011
5.	Perguruan Tinggi	Institut Teknologi Telkom Surabaya
6.	Tempat, Tanggal Lahir	Sleman, 22 Oktober 2003
7.	E-mail	rendikarendi96@gmail.com
8.	Nomor Telepon/HP	081998396441

1.3 Biodata Anggota 2

1.	Nama Lengkap	Ahmad Mu'min Faisal
2.	Jenis Kelamin	Laki-laki
3.	Program Studi	S1 Informatika
4.	Nama NIM	1203210101
5.	Perguruan Tinggi	Institut Teknologi Telkom Surabaya
6.	Tempat, Tanggal Lahir	Nganjuk, 22 Juni 2003
7.	E-mail	ahmad.faisalewy@gmail.com
8.	Nomor Telepon/HP	0895365037183