Tugas Individu 1
Data Preparation

Pendahuluan

Menurut George Fuechsel, terdapat sebuah istilah berupa "*Garbage In, Garbage Out*" yang berarti kualitas dari *output* akan sangat ditentukan oleh kualitas *input*-nya. Dalam proses analisis data dan *decision making*, kita menginginkan hasil atau *output* yang semaksimal mungkin. Dalam mendukung hasil yang maksimal, sebaiknya dilakukan *data preparation* terlebih dahulu. *Data preparation* bertujuan untuk mengolah suatu data mentah menjadi data yang berkualitas. Dalam prosesnya terdapat beberapa hal yang umum dilakukan seperti *data cleaning*, *data reduction*, dan *data transformation*. Dalam tugas kali ini, mahasiswa diminta untuk melakukan *data preparation* terhadap *dataset home loan*.

Deskripsi Data

Menampilkan sebagian data dengan menggunakan head() dan tail(). Dari data tersebut dapat dilihat terdapat beberapa column seperti loan_status, loan_amnt, int_rate, grade, emp_length, home_ownership, annual_inc, term.

1. loan_status
   Column ini menunjukkan data terkait status peminjaman. Value data yang diinginkan: Current, Charged off, Default, Fully Paid, In Grace Period, Late (16-30 days), Late (31-120 days)
2. loan_amnt
   Column ini menunjukkan jumlah pinjaman yang dimiliki oleh record. loan_amnt memiliki tipe data integer.
3. int_rate
   Column ini menunjukkan tingkat atau persentase suku bunga dari pinjaman. int_rate memiliki tipe data folat.
4. grade
   Column ini menunjukkan tingkat pekerjaan peminjam. Value data pada column ini terdiri dari A, B, C, D, E, F, G.

5. emp_length

   Column ini menunjukkan lama durasi kerja peminjam. Value pada column ini terdiri dari < 1 year, 1 year, 2 years, 3 years, 4 years, 5 years, 6 years, 7 years, 8 years, 9 years, 10+ years.

6. home_ownership

   Column ini menunjukkan data terkait status kepemilikan tempat tinggal. Value data yang diinginkan: MORTGAGE, RENT, OWN, ANY.

7. annual_inc

   Column ini menunjukkan data terkait jumlah pendapatan tahunan peminjam. Column ini masih memiliki tipe data object (seharusnya berbentuk numeric).

8. term

   Column ini menunjukkan jangka waktu peminjaman. Value pada column ini hanya dapat memiliki nilai 30 bulan atau 60 bulan.

```
loan_dataset.head()
```

```
  loan_status  loan_amnt  int_rate  ... home_ownership annual_inc         term
0     Current       2500     13.56  ...           RENT      55000   36 months
1     Current      30000     18.94  ...       MORTGAGE      90000   60 months
2     Current       5000     17.97  ...       MORTGAGE      59280   36 months
3     Current       4000     18.94  ...       MORTGAGE      92000   36 months
4     Current      30000     16.14  ...       MORTGAGE      57250   60 months

[5 rows x 8 columns]
```

```
loan_dataset.tail()
```

```
    loan_status  loan_amnt  ...  annual_inc         term
149994             Current       9000  ...       53000   36 months
149995     In Grace Period       7000  ...       34000   36 months
149996             Current      25525  ...       76000   36 months
149997  Late (31-120 days)      25000  ...       80000   60 months
149998             Current      18000  ...      156000   36 months

[5 rows x 8 columns]
```

Dari loan_dataset.shape(), kita mendapatkan informasi bahwa dataset ini memiliki 149999 row dengan 8 total column.

```
loan_dataset.shape()
```

```
(149999, 8)
```

```
loan_dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 149999 entries, 0 to 149998
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   loan_status     149999 non-null  object
 1   loan_amnt       149999 non-null  int64
 2   int_rate        149999 non-null  float64
 3   grade           149999 non-null  object
 4   emp_length      136331 non-null  object
 5   home_ownership  149998 non-null  object
 6   annual_inc      149999 non-null  object
 7   term            149999 non-null  object
dtypes: float64(1), int64(1), object(6)
memory usage: 5.7+ MB
None
```

```
loan_dataset.describe()
```

```
           loan_amnt       int_rate
count  149999.000000  149999.000000
mean    16020.048300      12.908296
std     10138.235301       5.127500
min      1000.000000       6.000000
25%      8000.000000       8.460000
50%     14000.000000      11.800000
75%     21987.500000      16.140000
max     40000.000000      30.990000
```

Data Preparation

1. Melakukan import package

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

2. Load data

```
loan_dataset = pd.read_csv('data-t1.csv’, na_values = “n/a”, low_memory = False)
loan_dataset.head()
```

Muncul error pada line 52431 dan 131201 karena adanya data pada kolom 9 (I)

```
Traceback (most recent call last):
File "pandas\_libs\parsers.pyx", line 2070, in
pandas._libs.parsers.raise_parser_error
```

```
pandas.errors.ParserError: Error tokenizing data. C error: Expected 8
fields in line 52431, saw 9
```

	A	B	C	D	E	F	G	H	I	J
52428	Current	3500	16.91	C	10+ years	RENT	82000	36 months		
52429	Current	40000	14.47	C	5 years	MORTGAG	47150	60 months		
52430	Current	15000	16.91	C	10+ years	OWN	72000	60 months		
52431	Current	9500	6.46	A	10+ years	MORTGAG	138831		36 months	
52432	Current	8000	13.56	C	2 years	MORTGAG	80000	36 months		
52433	In Grace P	12000	10.72	B	10+ years	MORTGAG	140000	36 months		

Terdapat beberapa cara yang dapat dilakukan untuk menangani hal seperti ini:

- Manual
- Memberikan limit column data yang akan diproses

  Hal ini menyebabkan data yang diambil hanya yang berada pada column index 0-7 dan menganggap data yang kosong sebagai null

  ```
  loan_dataset = pd.read_csv('data-t1.csv',low_memory = False, names =
  list(range(0,8)))
  ```

- Melakukan skipping line

  ```
  loan_dataset = pd.read_csv('data-t1.csv', low_memory = False,
  error_bad_lines = False)
  ```

  ```
  b'Skipping line 52431: expected 8 fields, saw 9\nSkipping line
  131201: expected 8 fields, saw 9\n'
  ```

Pada kali ini saya menggunakan metode manual karena hanya terdapat 2 line yang error sebagai bentuk efisiensi. Jika menggunakan metode kedua maka dapat digunakan data pada column 8 row 52431 dan 131201 akan bernilai null.

3. Check Missing Value

   Dapat diperhatikan bahwa terdapat cell yang memiliki nilai “n/a”, nilai ini tidak dapat langsung di cek menggunakan .isnull tetapi karena sudah dilakukan na_values = “n/a” pada saat load data, maka semua nilai “n/a” akan dianggap sebagai NA atau NaN.

```
loan_dataset.isnull().sum(axis = 0))
```

```
loan_status           0
loan_amnt             0
int_rate              0
grade                 0
emp_length        13668
home_ownership        1
annual_inc            0
```

```
term                 0
dtype: int64
```

Kemudian kita berniat merubah nilai NA/NaN ini menjadi berisi, maka dilakukan transformasi dengan menggunakan nilai **modus** pada emp_length karena merupakan sebuah kata sehingga tidak bisa menggunakan mean atau standar deviasi.

```
loan_dataset['emp_length'].fillna(loan_dataset['emp_length'].mode()[0],
inplace = True)
```

```
loan_dataset['home_ownership'].fillna(loan_dataset['home_ownership'].mode
()[0], inplace = True)
```

Dapat dilihat pada kode diatas parameter (inplace = true) pada fillna bertujuan untuk tidak mereturn dataframe baru, yaitu melakukan perubahan langsung pada loan_dataset (dataframe lama).

```
loan_status        0
loan_amnt          0
int_rate           0
grade              0
emp_length         0
home_ownership     0
annual_inc         0
term               0
dtype: int64
```

Missing value telah berhasil kita transformasi.

4. Check Inconsistent Data

   Jika diperhatikan dari dataset, terdapat data yang tidak sesuai dengan data-data lainnya seperti pada contoh gambar dibawah. Oleh karena itu, akan dilakukan pengecekkan untuk setiap column

212	Current	15000	15.02	C	10+ years	MORTGAG	175000	36 months
213	18-Dec	15000	13.56	C	Patient Finance"	4 years	MORTGAG	60 months
214	Current	20000	14.47	C	4 years	OWN	52000	36 months
215	Current	20400	10.33	B	10+ years	MORTGAG	140000	60 months
216	Current	20000	12.98	B	< 1 year	RENT	40000	60 months
217	Fully Paid	8425	27.27	E	3 years	MORTGAG	450000	36 months
218	Current	21000	16.91	C	< 1 year	OWN	68000	36 months
219	Current	10000	14.47	C	8 years	MORTGAG	143000	36 months
220	Current	15000	15.02	C	10+ years	MORTGAG	42000	36 months
221	Current	24000	13.56	C	10+ years	MORTGAG	75000	60 months
222	18-Dec	15000	14.47	C	Solutions Management"	9 years	RENT	60 months

Terdapat pola dimana column emp_length yang berpindah ke column home_ownership dan column home_ownership yang berpindah pada annual_inc sehingga perlu dilakukan pemindahan dulu data-data tersebut ke kolum aslinya

```
loan_dataset['emp_length'] =
np.where((loan_dataset['home_ownership'].str.contains('years')),
loan_dataset['home_ownership'], loan_dataset['emp_length'])
```

```
loan_dataset['home_ownership'] =
np.where((loan_dataset['annual_inc'].str.contains('MORTGAGE|RENT|OWN')),loan_data
set['annual_inc'], loan_dataset['home_ownership'])
```

Pada kode diatas, dilakukan pemindahan value column home_ownership ke column emp_length jika pada column home_ownership ditemukan substring ‘years’ dan pemindahan value column annual_inc ke column home_ownership jika ditemukan substring ‘MORTGAGE’ atau ‘RENT’ atau ‘OWN’ atay ‘ANY’.

Setelah melakukan pembetulan pola kesalahan posisi data yang diulang, saya mencoba melihat lagi apakah masih ada data yang tidak sesuai.

```
print(loan_dataset['home_ownership'].unique())
```

```
['RENT' 'MORTGAGE' 'OWN' 'ANY' ' Planning and Accountab"' ' Events "'
 ' Regional Manager"' 'MOTGAGE' ' Construction Supervisor"' ' GS 13-3"'
 ' Ethics and Compliance"' ' carpenter"' ' HR"' ' IT Mgr"' ' NC and TN"'
 ' Videographer"' ' and Communications "' ' and Implant Dentist"' '
cesc"'
 ' utah"' 'Server"' ' Sales"' ' Director of C"' 'MORGAGE' ' Supervisor"'
 'installation"' 'maintenance"' ' and Wellness"' ' "' ' security "'
 ' Associate Prof"' ' coach"' ' starbuck' ' Pension & Comp"'
 ' Brand Ambassador' '  Rare Disor"' ' store manager"' ' entertainer"'
 ' master tech"' ' welder"']
```

Ternyata masih terdapat data yang typo yaitu ‘MOTGAGE’ dan ‘MORGAGE’ dan sisanya tidak sesuai. Kita akan mulai dengan memperbaiki typo pada column home_ownership:

```
loan_dataset.loc[loan_dataset['home_ownership'].str.contains('MOTGAGE|MOR
GAGE'), 'home_ownership'] = 'MORTGAGE'
```

Kemudian memperbaiki data-data lain yang tidak sesuai dengan menggunakan transformasi berdasarkan modus

```
loan_dataset.loc[~loan_dataset['home_ownership'].isin(['MORTGAGE','RENT',
'OWN','ANY']), 'home_ownership'] =
loan_dataset['home_ownership'].mode()[0]
```

Setelah dua kode diatas, opsi/data dari home_ownership menjadi seperti ini.

```
dtype: int64
['RENT' 'MORTGAGE' 'OWN' 'ANY']
```

Kemudian kita melakukan pembersihan data seperti ini kepada column column lainnya:

//emp_length

```
{'10+ years': 59153, '< 1 year': 18326, '2 years': 12705, '3 years': 11792, '1 year': 10395, '5 years': 9075, '4 years': 8809, '6 years': 6345, '7 years': 5112, '8 years': 4804, '9 years': 3255, ' Business Development"': 7, ' Marketing"': 5, ' Operations"': 5, ' Human Resources"': 4, ' Client Services"': 2, ' Program Management"': 2, ' RN"': 2, ' Sr"': 2, ' Professional Services"': 2, ' VP"': 2, ' FP&A"': 2, ' Communications"': 2, ' Accounting Policy Analyst"': 1, ' Online Media"': 1, ' Public Relation"': 1, ' Social Media"': 1, ' 1st Lt"': 1, … }
```

Dapat diperhatikan bahwa dalam data tersebut terdapat banyak data yang tidak sesuai, sehingga kita hanya akan mengambil data dari yang paling awal (10+ years) hingga data angka terakhir (9 years) dan sisanya akan dihiraukan.

```
loan_dataset = loan_dataset[loan_dataset.groupby('emp_length').emp_length.transform('count') > 3225]
```

```
['10+ years' '6 years' '4 years' '< 1 year' '2 years' '9 years' '5 years'
 '3 years' '7 years' '1 year' '8 years']
```

//loan_status

```
['Current' 'Fully Paid' 'Dec-18' 'Late (31-120 days)' 'In Grace Period'
 'Charged Off' 'Late (16-30 days)' 'Nov-18' 'Oct-18' 'Sep-18' 'Fulli
Paid' 'Full Paid' 'Curren' 'Curent']
```

Dilihat dari value data, banyak data yang typo seperti Fully Paid menjadi Fulli Paid, Full Paid serta Current yang menjadi Curren dan Curent.

```
loan_dataset.loc[loan_dataset['loan_status'].str.contains('Curren|Curent'), 'loan_status'] = 'Current'
```

```
loan_dataset.loc[loan_dataset['loan_status'].str.contains('Fulli Paid|Full Paid'), 'loan_status'] = 'Fully Paid'
```

```
['Current' 'Fully Paid' 'Dec-18' 'Late (31-120 days)' 'In Grace Period'
 'Charged Off' 'Late (16-30 days)' 'Nov-18' 'Oct-18' 'Sep-18']
```

Terdapat data-data lain yang bentuknya date seperti Nov-18, Oct-18, Sep-18. Kita akan memperbaiki data tersebut dengan mentransformasinya menggunakan modus.

```
loan_dataset.loc[loan_dataset['loan_status'].str.contains('Nov-18|Oct-18|
Sep-18'), 'loan_status'] = loan_dataset['loan_status'].mode()[0]
```

```
['Current' 'Fully Paid' 'Dec-18' 'Late (31-120 days)' 'In Grace Period'
 'Charged Off' 'Late (16-30 days)']
```

//annual_inc

```
loan_dataset.loc[loan_dataset['annual_inc'].str.contains('MORTGAGE|RENT|O
WN|ANY'), 'annual_inc'] = 0
```

5. Change data type to simplest form

   Karena semua term berbentuk "[angka] months", kita dapat mengubah column ini sehingga menjadi angka atau integer saja dengan menghilangkan substring months dan kemudian mengubah tipe data menjadi integer.

```
loan_dataset['term'] = loan_dataset['term'].map(lambda x:
x.rstrip('months'))
```

```
loan_status        object
loan_amnt           int64
int_rate          float64
grade              object
emp_length         object
home_ownership     object
annual_inc         object
term                int64
dtype: object
```

Kita juga akan merubah emp_length dengan menghilangkan "years".

```
loan_dataset['emp_length'] = loan_dataset['emp_length'].map(lambda x:
x.rstrip('years'))
```

Untuk efisiensi dan memudahkan dalam proses analisis, kita dapat mengkategorikan data menjadi <1, 1-5, 5-10, >10.

```
def categorizing_data(cell):
    if str(cell) == "10+ ":
        return "> 10"
    elif cell == "< 1 ":
        return "< 1"
    elif cell in ['1 ','2 ', '3 ', '4 ', '5 ']:
        return " 1-5"
    elif cell in ['6 ', '7 ', '8 ', '9 ']:
        return " 6-10"
    else:
```

```
        return "0"

loan_dataset['emp_length'] = loan_dataset.apply(lambda cell:
categorizing_data(cell.emp_length), axis=1)
```

Ubah data annual_inc menjadi integer dibanding dengan menggunakan dtype object seperti saat ini.

```
loan_dataset['annual_inc'] = pd.to_numeric(loan_dataset['annual_inc'])
```

6. Remove outlier data

   Kita harus melakukan perhitungan untuk menentukan IQR yang merupakan hasil dari Q3-Q1, data dianggap outlier jika berada dibawah Q1-1.5*IQR atau berada diatas Q3+1.5*IQR.

```
Q1 = loan_dataset.quantile(0.25)
Q3 = loan_dataset.quantile(0.75)
IQR = Q3 - Q1
```

   Diinginkan untuk meremove data outlier sehingga:

```
loan_dataset = loan_dataset[~((loan_dataset < (Q1 - 1.5 * IQR))
|(loan_dataset > (Q3 + 1.5 * IQR))).any(axis=1)]
loan_dataset.shape
```

Hasil/Temuan

Dari hasil preparasi data, kita telah mendapatkan data yang bersih. Berikut merupakan display sebagian data yang sudah dibersihkan.

```
# get view of data after cleansing
loan_dataset.head()
```

	loan_status	loan_amnt	int_rate	grade	emp_length	home_ownership	annual_inc	term
0	Current	2500	13.56	C	> 10	RENT	55000.0	36
1	Current	30000	18.94	D	> 10	MORTGAGE	90000.0	60
2	Current	5000	17.97	D	6-10	MORTGAGE	59280.0	36
3	Current	4000	18.94	D	> 10	MORTGAGE	92000.0	36
4	Current	30000	16.14	C	> 10	MORTGAGE	57250.0	60

```
loan_dataset.tail()
```

	loan_status	loan_amnt	int_rate	grade	emp_length	home_ownership	annual_inc	term
149994	Current	9000	11.55	B	> 10	RENT	53000.0	36
149995	In Grace Period	7000	6.67	A	> 10	OWN	34000.0	36

```
149996 Current       25525  19.92  D      1-5    RENT   76000.0      36
149997 Late (31-120 days)  25000  10.08  B      < 1    RENT   80000.0      60
149998 Current       18000  23.40  E      1-5    MORTGAGE     156000.0     36
```

Dari hasil preparasi data, ditemukan banyak sekali kesalahan/ketidaksesuaian dalam dataset ini sehingga dataset ini dapat dianggap messy. Dari proses preparasi yang dilakukan, jika hanya dilakukan drop saja untuk setiap data yang bermasalah maka akan dapat mengurangi 15% dari total data atau sekitar 22.731 record. Hal ini berarti terdapat cukup banyak record yang tidak sesuai dengan data yang kita inginkan (clean data).

Kesimpulan

Dari proses data preparation, dapat diketahui mengenai tujuan dari data preparation itu sendiri yaitu bisa jadi data yang kita miliki masih kotor dan belum memenuhi format yang kita inginkan. Data-data yang dipreparasi pada proses diatas dapat dibagi menjadi data yang tidak lengkap (null value), data yang merupakan noise (error/outlier), dan data yang tidak konsisten (berisi value dari column lain, tidak sesuai dengan opsi value yang seharusnya). Proses yang saya lakukan untuk dataset ini hanya sebatas pada data cleansing (mengisi null value, smoothing data noisy, membuang outlier, mengatasi ketidakkonsistenan data) dan data transformation (normalisasi menggunakan modus). Dengan adanya preparasi seperti menghilangkan null value, melakukan pengecekkan kesesuaian value, menghilangkan outlier kita akan mendapatkan data yang bersih atau cukup bersih sehingga meminimalisir kesalahan dalam analisis data.