9

April 29, 2023

0.1 1 Group Name: Go Bear!

0.2 2 Name: Xiaoke Song

0.3 3 Email: xiaokesong57@gmail.com

0.4 4 Country: born in China, college in the US

0.5 5 College: UC Berkeley

0.6 6 Specialization: Data Science

[1]:
```python
import pandas as pd
```

[16]:
```python
df = pd.read_csv('cust_seg.csv')
df.head(5)
```

```
/var/folders/_0/nmpfpzw134n12j0c0z6jtrw80000gn/T/ipykernel_59787/3036801543.py:1
: DtypeWarning: Columns (16) have mixed types. Specify dtype option on import or
set low_memory=False.
  df = pd.read_csv('cust_seg.csv')
```

[16]:
```
   Unnamed: 0  fecha_dato  ncodpers ind_empleado pais_residencia sexo  age  \
0           0  2015-01-28   1375586            N              ES    H   35
1           1  2015-01-28   1050611            N              ES    V   23
2           2  2015-01-28   1050612            N              ES    V   23
3           3  2015-01-28   1050613            N              ES    H   22
4           4  2015-01-28   1050614            N              ES    V   23

   fecha_alta  ind_nuevo antiguedad  …  ind_hip_fin_ult1 ind_plan_fin_ult1  \
0  2015-01-12        0.0          6  …                 0                 0
1  2012-08-10        0.0         35  …                 0                 0
2  2012-08-10        0.0         35  …                 0                 0
3  2012-08-10        0.0         35  …                 0                 0
4  2012-08-10        0.0         35  …                 0                 0

   ind_pres_fin_ult1 ind_reca_fin_ult1 ind_tjcr_fin_ult1 ind_valo_fin_ult1  \
0                  0                 0                 0                 0
1                  0                 0                 0                 0
2                  0                 0                 0                 0
```

```
3                  0                  0                  0                  0
4                  0                  0                  0                  0

  ind_viv_fin_ult1 ind_nomina_ult1 ind_nom_pens_ult1  ind_recibo_ult1
0                0             0.0                0.0                0
1                0             0.0                0.0                0
2                0             0.0                0.0                0
3                0             0.0                0.0                0
4                0             0.0                0.0                0

[5 rows x 48 columns]
```

[17]:

```
print(df.shape)
```

```
(1000000, 48)
```

[18]:

```
print(df.columns)
```

```
Index(['Unnamed: 0', 'fecha_dato', 'ncodpers', 'ind_empleado',
       'pais_residencia', 'sexo', 'age', 'fecha_alta', 'ind_nuevo',
       'antiguedad', 'indrel', 'ult_fec_cli_1t', 'indrel_1mes', 'tiprel_1mes',
       'indresi', 'indext', 'conyuemp', 'canal_entrada', 'indfall', 'tipodom',
       'cod_prov', 'nomprov', 'ind_actividad_cliente', 'renta',
       'ind_ahor_fin_ult1', 'ind_aval_fin_ult1', 'ind_cco_fin_ult1',
       'ind_cder_fin_ult1', 'ind_cno_fin_ult1', 'ind_ctju_fin_ult1',
       'ind_ctma_fin_ult1', 'ind_ctop_fin_ult1', 'ind_ctpp_fin_ult1',
       'ind_deco_fin_ult1', 'ind_deme_fin_ult1', 'ind_dela_fin_ult1',
       'ind_ecue_fin_ult1', 'ind_fond_fin_ult1', 'ind_hip_fin_ult1',
       'ind_plan_fin_ult1', 'ind_pres_fin_ult1', 'ind_reca_fin_ult1',
       'ind_tjcr_fin_ult1', 'ind_valo_fin_ult1', 'ind_viv_fin_ult1',
       'ind_nomina_ult1', 'ind_nom_pens_ult1', 'ind_recibo_ult1'],
      dtype='object')
```

Problem description: Customer Segmation _____ “XYZ bank wants to roll out Christmas offers to their customers. But Bank does not want to roll out same offer to all customers instead they want to roll out personalized offer to particular set of customers. If they manually start understanding the category of customer then this will be not efficient and also they will not be able to uncover the hidden pattern in the data (pattern which group certain kind of customer in one category). Bank approached ABC analytics company to solve their problem. Bank also shared information with ABC analytics that they don’t want more than 5 group as this will be inefficient for their campaign.”

0.6.1 Data Cleaning and Transformation:

[19]:

```
# Drop any rows with missing values
df.dropna(inplace=True)

# Convert fecha_dato and fecha_alta columns to datetime format
```

```python
df['fecha_dato'] = pd.to_datetime(df['fecha_dato'])
df['fecha_alta'] = pd.to_datetime(df['fecha_alta'])

# Create a new column with the year of the fecha_alta column
df['year_alta'] = df['fecha_alta'].dt.year

# Convert antiguedad column to numeric format and replace -999999 with 0
df['antiguedad'] = pd.to_numeric(df['antiguedad'], errors='coerce')
df['antiguedad'].replace(-999999, 0, inplace=True)

# Create a new column with the difference in months between fecha_dato and fecha_alta
df['months_active'] = (df['fecha_dato'].dt.year - df['fecha_alta'].dt.year) * 12 + (df['fecha_dato'].dt.month - df['fecha_alta'].dt.month)

# Drop the original fecha_alta column
df.drop(columns=['fecha_alta'], inplace=True)

# Reset the index
df.reset_index(drop=True, inplace=True)
```

[ ]:

```python
from sklearn.linear_model import LinearRegression

# Fit a linear regression model
X = df.drop('target', axis=1)
y = df['target']
model = LinearRegression()
model.fit(X, y)

# Identify outliers based on residuals
residuals = y - model.predict(X)
outliers = residuals.abs() > 3 * residuals.std()
df = df[~outliers]
```

[ ]:

```python

```