Data Mining - Lab 05 - Feature Engineering

Nguyễn Đức Vũ Duy - 18110004

```
#Import libraries
# !pip install featuretools
import featuretools as ft
import numpy as np
import pandas as pd
```

```
#Import dataset
path='https://raw.githubusercontent.com/duynguyenhcmus/Repository/main/HocKy_2/KhaiThacDuLieu/BigMartSales.csv'
df=pd.read_csv(path,header=0)
df.head()
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Est
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	

```
#Check the null value
df.isna().sum()
```

```
Item_Identifier               0
Item_Weight                1463
Item_Fat_Content              0
Item_Visibility               0
Item_Type                     0
Item_MRP                      0
Outlet_Identifier             0
Outlet_Establishment_Year     0
Outlet_Size                2410
Outlet_Location_Type          0
Outlet_Type                   0
Item_Outlet_Sales             0
dtype: int64
```

```
#Deal with null value, fill with mean and mode
df.Item_Weight.fillna(df.Item_Weight.mean(),inplace=True)
df.Outlet_Size.fillna('Medium',inplace=True)
df.isna().sum()
```

```
Item_Identifier            0
Item_Weight                0
Item_Fat_Content           0
Item_Visibility            0
Item_Type                  0
Item_MRP                   0
Outlet_Identifier          0
Outlet_Establishment_Year  0
Outlet_Size                0
Outlet_Location_Type       0
Outlet_Type                0
Item_Outlet_Sales          0
dtype: int64
```

```
# dictionary to replace the categories
fat_content_dict = {'Low Fat':0, 'Regular':1, 'LF':0, 'reg':1, 'low fat':0}

#Preprocessing Item_Fat_Content to have only 2 values
df['Item_Fat_Content'] = df['Item_Fat_Content'].replace(fat_content_dict, regex=True)
df.head()
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Est
0	FDA15	9.30	0	0.016047	Dairy	249.8092	OUT049	
1	DRC01	5.92	1	0.019278	Soft Drinks	48.2692	OUT018	
2	FDN15	17.50	0	0.016760	Meat	141.6180	OUT049	
3	FDX07	19.20	1	0.000000	Fruits and Vegetables	182.0950	OUT010	
4	NCD19	8.93	0	0.000000	Household	53.8614	OUT013	

```
#Combine Item_Identider and Outlet_Identifier for Id
df['id'] = df['Item_Identifier'] + df['Outlet_Identifier']
df.drop(['Item_Identifier'], axis=1, inplace=True)
df.head()
```

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_I
0	9.30	0	0.016047	Dairy	249.8092	
1	5.92	1	0.019278	Soft Drinks	48.2692	
2	17.50	0	0.016760	Meat	141.6180	
3	19.20	1	0.000000	Fruits and Vegetables	182.0950	
4	8.93	0	0.000000	Household	53.8614	

```
# creating and entity set 'es'
es = ft.EntitySet(id = 'sales')

# adding a dataframe
es.entity_from_dataframe(entity_id = 'bigmart', dataframe = df, index = 'id')
```

```
Entityset: sales
  Entities:
    bigmart [Rows: 8523, Columns: 12]
  Relationships:
    No relationships
```

```
es.normalize_entity(base_entity_id='bigmart', new_entity_id='outlet', index = 'Outlet_Identifier',
additional_variables = ['Outlet_Establishment_Year', 'Outlet_Size', 'Outlet_Location_Type', 'Outlet_Type'])
```

```
Entityset: sales
  Entities:
    bigmart [Rows: 8523, Columns: 8]
    outlet [Rows: 10, Columns: 5]
  Relationships:
    bigmart.Outlet_Identifier -> outlet.Outlet_Identifier
```

```
#Build Deep Feature Synthesis
feature_matrix, feature_names = ft.dfs(entityset=es,
target_entity = 'bigmart',
max_depth = 2,
verbose = 1)
```

```
Built 44 features
Elapsed: 00:00 | Progress: 100%|██████████
```

```
#Print the built features
feature_names
```

```
[<Feature: Item_Weight>,
 <Feature: Item_Fat_Content>,
 <Feature: Item_Visibility>,
 <Feature: Item_Type>,
 <Feature: Item_MRP>,
 <Feature: Outlet_Identifier>,
 <Feature: Item_Outlet_Sales>,
 <Feature: outlet.Outlet_Establishment_Year>,
 <Feature: outlet.Outlet_Size>,
```

```
<Feature: outlet.Outlet_Location_Type>,
<Feature: outlet.Outlet_Type>,
<Feature: outlet.COUNT(bigmart)>,
<Feature: outlet.MAX(bigmart.Item_Fat_Content)>,
<Feature: outlet.MAX(bigmart.Item_MRP)>,
<Feature: outlet.MAX(bigmart.Item_Outlet_Sales)>,
<Feature: outlet.MAX(bigmart.Item_Visibility)>,
<Feature: outlet.MAX(bigmart.Item_Weight)>,
<Feature: outlet.MEAN(bigmart.Item_Fat_Content)>,
<Feature: outlet.MEAN(bigmart.Item_MRP)>,
<Feature: outlet.MEAN(bigmart.Item_Outlet_Sales)>,
<Feature: outlet.MEAN(bigmart.Item_Visibility)>,
<Feature: outlet.MEAN(bigmart.Item_Weight)>,
<Feature: outlet.MIN(bigmart.Item_Fat_Content)>,
<Feature: outlet.MIN(bigmart.Item_MRP)>,
<Feature: outlet.MIN(bigmart.Item_Outlet_Sales)>,
<Feature: outlet.MIN(bigmart.Item_Visibility)>,
<Feature: outlet.MIN(bigmart.Item_Weight)>,
<Feature: outlet.MODE(bigmart.Item_Type)>,
<Feature: outlet.NUM_UNIQUE(bigmart.Item_Type)>,
<Feature: outlet.SKEW(bigmart.Item_Fat_Content)>,
<Feature: outlet.SKEW(bigmart.Item_MRP)>,
<Feature: outlet.SKEW(bigmart.Item_Outlet_Sales)>,
<Feature: outlet.SKEW(bigmart.Item_Visibility)>,
<Feature: outlet.SKEW(bigmart.Item_Weight)>,
<Feature: outlet.STD(bigmart.Item_Fat_Content)>,
<Feature: outlet.STD(bigmart.Item_MRP)>,
<Feature: outlet.STD(bigmart.Item_Outlet_Sales)>,
<Feature: outlet.STD(bigmart.Item_Visibility)>,
<Feature: outlet.STD(bigmart.Item_Weight)>,
<Feature: outlet.SUM(bigmart.Item_Fat_Content)>,
<Feature: outlet.SUM(bigmart.Item_MRP)>,
<Feature: outlet.SUM(bigmart.Item_Outlet_Sales)>,
<Feature: outlet.SUM(bigmart.Item_Visibility)>,
<Feature: outlet.SUM(bigmart.Item_Weight)>]
```

```
#Reset Index id
feature_matrix = feature_matrix.reindex(index=df['id'])
feature_matrix = feature_matrix.reset_index()
```

```
# !pip install catboost  #Install catboost module
from catboost import CatBoostRegressor

#Deal with categorical_features
categorical_features = np.where(feature_matrix.dtypes == 'object')[0]

for i in categorical_features:
    feature_matrix.iloc[:,i] = feature_matrix.iloc[:,i].astype('str')

feature_matrix=feature_matrix.drop(['id','Outlet_Identifier'],axis=1)
```

```
#Set X,y for train and test
X=feature_matrix.drop(['Item_Outlet_Sales'],axis=1)
y=feature_matrix['Item_Outlet_Sales']
```

```
#Take the categorical features for catboost regressor
categorical_features=np.where(X.dtypes == 'object')[0]
```

```
#Import sklearn train_test_split
from sklearn.model_selection import train_test_split

#Train test split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=11)

#Import Model CatBoostRegressor
model_cat = CatBoostRegressor(iterations=100, learning_rate=0.3, depth=6, eval_metric='RMSE', random_seed=7)

# training model
model_cat.fit(X_train, y_train, cat_features=categorical_features, use_best_model=True)

```

```
41:     learn: 1030.1586799     total: 249ms    remaining: 343ms
42:     learn: 1027.7633587     total: 255ms    remaining: 338ms
43:     learn: 1025.9416202     total: 259ms    remaining: 330ms
44:     learn: 1025.8268245     total: 263ms    remaining: 322ms
45:     learn: 1024.3383395     total: 268ms    remaining: 314ms
46:     learn: 1023.3220822     total: 272ms    remaining: 307ms
47:     learn: 1021.4891271     total: 277ms    remaining: 300ms
48:     learn: 1021.0208181     total: 281ms    remaining: 292ms
49:     learn: 1018.2120202     total: 286ms    remaining: 286ms
50:     learn: 1017.7335576     total: 290ms    remaining: 278ms
```

```
51:     learn: 1016.3034868     total: 294ms    remaining: 272ms
52:     learn: 1015.3244956     total: 299ms    remaining: 265ms
53:     learn: 1015.1850965     total: 303ms    remaining: 258ms
54:     learn: 1014.1988355     total: 307ms    remaining: 251ms
55:     learn: 1013.4727730     total: 311ms    remaining: 245ms
56:     learn: 1011.7189740     total: 316ms    remaining: 238ms
57:     learn: 1009.6441888     total: 321ms    remaining: 232ms
58:     learn: 1008.5578876     total: 325ms    remaining: 226ms
59:     learn: 1008.0844024     total: 329ms    remaining: 219ms
60:     learn: 1006.5223130     total: 334ms    remaining: 213ms
61:     learn: 1005.0531805     total: 338ms    remaining: 207ms
62:     learn: 1004.5666356     total: 342ms    remaining: 201ms
63:     learn: 1002.3030740     total: 346ms    remaining: 195ms
64:     learn: 999.5444003      total: 358ms    remaining: 193ms
65:     learn: 998.8751923      total: 362ms    remaining: 187ms
66:     learn: 997.9191382      total: 367ms    remaining: 181ms
67:     learn: 996.9384581      total: 372ms    remaining: 175ms
68:     learn: 996.5011975      total: 376ms    remaining: 169ms
69:     learn: 994.5480787      total: 381ms    remaining: 163ms
70:     learn: 993.3536244      total: 386ms    remaining: 157ms
71:     learn: 993.1308054      total: 390ms    remaining: 152ms
72:     learn: 990.6092292      total: 397ms    remaining: 147ms
73:     learn: 988.9809912      total: 403ms    remaining: 141ms
74:     learn: 987.1268538      total: 407ms    remaining: 136ms
75:     learn: 986.8282039      total: 411ms    remaining: 130ms
76:     learn: 985.5408791      total: 416ms    remaining: 124ms
77:     learn: 983.2273717      total: 420ms    remaining: 119ms
78:     learn: 980.4510435      total: 425ms    remaining: 113ms
79:     learn: 977.6806357      total: 429ms    remaining: 107ms
80:     learn: 975.3693844      total: 433ms    remaining: 102ms
81:     learn: 974.3514031      total: 438ms    remaining: 96.1ms
82:     learn: 971.8028735      total: 443ms    remaining: 90.7ms
83:     learn: 971.7225832      total: 451ms    remaining: 85.9ms
84:     learn: 970.7266211      total: 455ms    remaining: 80.4ms
85:     learn: 970.5151928      total: 459ms    remaining: 74.8ms
86:     learn: 966.9976882      total: 464ms    remaining: 69.3ms
87:     learn: 966.6366310      total: 468ms    remaining: 63.8ms
88:     learn: 964.7046928      total: 473ms    remaining: 58.4ms
89:     learn: 963.9837055      total: 477ms    remaining: 53ms
90:     learn: 962.8713035      total: 481ms    remaining: 47.6ms
91:     learn: 962.0253709      total: 486ms    remaining: 42.2ms
92:     learn: 960.7920787      total: 490ms    remaining: 36.9ms
93:     learn: 958.4015399      total: 494ms    remaining: 31.6ms
94:     learn: 957.1179485      total: 499ms    remaining: 26.3ms
95:     learn: 955.9136327      total: 503ms    remaining: 21ms

96:     learn: 955.2905956      total: 507ms    remaining: 15.7ms
97:     learn: 954.2338117      total: 512ms    remaining: 10.4ms
98:     learn: 952.5613248      total: 516ms    remaining: 5.21ms
99:     learn: 950.9960696      total: 521ms    remaining: 0us
```

```
1 #Model evaluation on test set
2 model_cat.score(X_test,y_test)
```

```
0.5811597077929681
```

```
1 #Model best Root Mean Square Error
2 model_cat.best_score_
```

```
{'learn': {'RMSE': 950.9960695654398}}
```

```
1 #Get the first 10 most importance features
2 feature_importance=pd.DataFrame(np.concatenate((model_cat.feature_importances_.reshape(-1,1),np.array(X.columns).resha
3 feature_importance_sort=feature_importance.sort_values(by=0,ascending=False)
4 feature_importance_sort.head(10)[1]
```

```
4                                    Item_MRP
22     outlet.MIN(bigmart.Item_Outlet_Sales)
17    outlet.MEAN(bigmart.Item_Outlet_Sales)
39     outlet.SUM(bigmart.Item_Outlet_Sales)
2                           Item_Visibility
0                               Item_Weight
9                     outlet.COUNT(bigmart)
37     outlet.SUM(bigmart.Item_Fat_Content)
3                                  Item_Type
30     outlet.SKEW(bigmart.Item_Visibility)
Name: 1, dtype: object
```

✓ 0s completed at 6:39 PM