

Fraud Detection



AŞAMALAR

1. Veri Setini İnceleme
2. Veri Setindeki Kategorik/Sayısal Analizi
3. Feature Engineering Yöntemlerini Uygulama
4. Veriyi Train/Test Olarak Ayırma
5. Model Seçimi
6. Model Eğitimi
7. Verilen Test Verisi ile Olasılıkları Bulma

Fraud Detection Amacı

- Bu mücadelenin amacı, dolandırıcılık vakalarını tespit etmek amacıyla perakendeci ortaklarından birinden gelen sepet verilerini işlemenin ve analiz etmenin en iyi yolunu bulmaktır. Bu sepet verilerini kullanarak dolandırıcı müşterilerin tespit edilmesi ve gelecekte reddedilmesi gerekir.



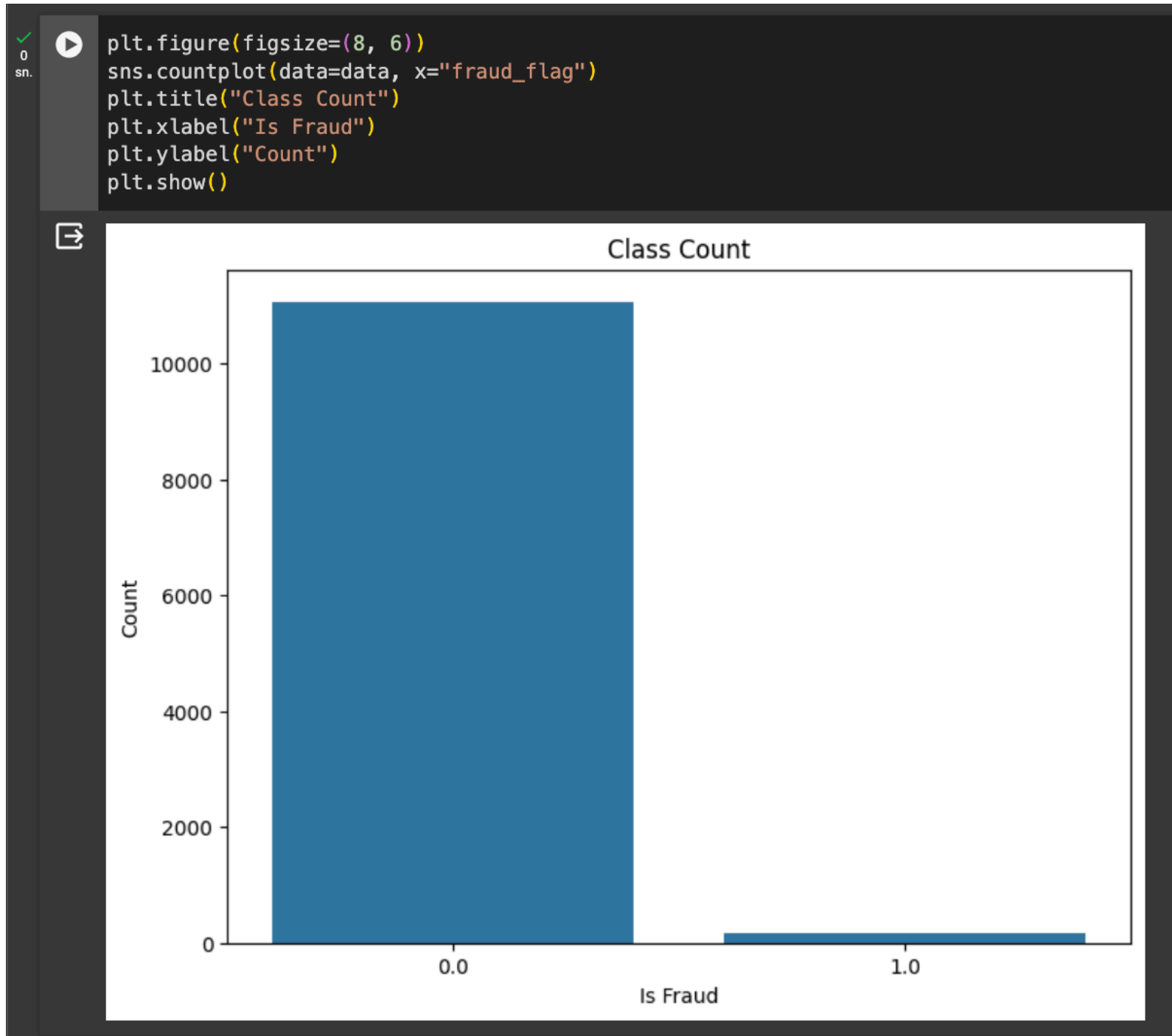
VERI ANALIZI

✓
0
sn.


```
[43] train_df.head(50)
```

[illegible]

VERİ ANALİZİ





VERİ ANALİZİ

✓ 0 sn.  data=pd.read_csv("train_dataset.csv")
data.head()

	ID	item1	item2	item3	item4	item5	item6	item7	item8	item9	.
0	79815	COMPUTER PERIPHERALS ACCESSORIES	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	22598	BEDROOM FURNITURE	BEDROOM FURNITURE	SERVICE	SERVICE	NaN	NaN	NaN	NaN	NaN	
2	63665	LIVING DINING FURNITURE	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	31312	COMPUTERS	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	30742	COMPUTERS	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

5 rows x 147 columns

✓ 0 sn.  data.dtypes

	ID	int64
	item1	object
	item2	object
	item3	object
	item4	object
		...
	Nbr_of_prod_purchas22	float64
	Nbr_of_prod_purchas23	float64
	Nbr_of_prod_purchas24	float64
	Nb_of_items	float64
	fraud_flag	float64
	Length: 147, dtype: object	

VERİ ANALİZİ

✓

0 sn.

▶

data.describe()

	ID	cash_price1	cash_price2	cash_price3	cash_price4	cash_price5	cash_price6	cash_price7	cash_price8
count	11235.000000	11235.000000	5460.000000	1545.000000	591.000000	289.000000	190.000000	131.000000	100.000000
mean	57339.146862	1094.867290	197.044872	205.400647	182.208122	192.660900	156.047368	131.656489	112.470000
std	33585.074232	704.817302	405.999025	419.205023	383.608176	363.381623	279.960188	210.986539	190.393820
min	19.000000	3.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	27911.500000	649.000000	7.000000	7.000000	10.000000	17.000000	13.000000	21.000000	12.000000
50%	56995.000000	949.000000	40.000000	44.000000	46.000000	59.000000	45.000000	56.000000	46.500000
75%	86498.500000	1349.000000	140.000000	188.000000	169.000000	190.000000	149.000000	125.000000	111.250000
max	115983.000000	14999.000000	4999.000000	5999.000000	5198.000000	2399.000000	1549.000000	1280.000000	1349.000000

8 rows × 53 columns

VERİ ANALİZİ

✓
0
sn.

```
[5] data.isnull().sum()
```

```
ID          0
item1        0
item2       5775
item3       9690
item4      10644
...
Nbr_of_prod_purchas22  11229
Nbr_of_prod_purchas23  11230
Nbr_of_prod_purchas24  11230
Nb_of_items           1
fraud_flag            1
Length: 147, dtype: int64
```

✓
0
sn.



```
data.columns
```

```
Index(['ID', 'item1', 'item2', 'item3', 'item4', 'item5', 'item6', 'item7',
      'item8', 'item9',
      ...,
      'Nbr_of_prod_purchas17', 'Nbr_of_prod_purchas18',
      'Nbr_of_prod_purchas19', 'Nbr_of_prod_purchas20',
      'Nbr_of_prod_purchas21', 'Nbr_of_prod_purchas22',
      'Nbr_of_prod_purchas23', 'Nbr_of_prod_purchas24', 'Nb_of_items',
      'fraud_flag'],
      dtype='object', length=147)
```


VERİ ANALİZİ

✓
1
sn. [7] train_df = pd.read_csv('train_dataset.csv')
test_df = pd.read_csv('submission_data_x.csv')

✓
0
sn. [8] print("Training Dataset Shape:", train_df.shape)

Training Dataset Shape: (74230, 147)

✓
0
sn. [9] print("Test Dataset Shape:", test_df.shape)

Test Dataset Shape: (18558, 146)

✓
0
sn. [10] print("Train Columns:", train_df.columns)

Train Columns: Index(['ID', 'item1', 'item2', 'item3', 'item4', 'item5', 'item6', 'item7',
'item8', 'item9',
...,
'Nbr_of_prod_purchas17', 'Nbr_of_prod_purchas18',
'Nbr_of_prod_purchas19', 'Nbr_of_prod_purchas20',
'Nbr_of_prod_purchas21', 'Nbr_of_prod_purchas22',
'Nbr_of_prod_purchas23', 'Nbr_of_prod_purchas24', 'Nb_of_items',
'fraud_flag'],
dtype='object', length=147)

Veri Setindeki Kategorik/Sayısal Analizi

- Kategorik değişkenleri stringe dönüştürdük çünkü bu değerler daha sonra kategorik kodlama teknikleri kullanılarak sayısal değerlere dönüştürülebilir.

✓
12
sn.

```
[13] # Tüm kategorik değişkenleri string e dönüştürün
      categorical_features = train_df.select_dtypes(include=['object']).columns

      train_df[categorical_features] = train_df[categorical_features].astype(str)

      test_df[categorical_features] = test_df[categorical_features].astype(str)
```

Veri Setindeki Kategorik/Sayısal Analizi

- Pipeline sayesinde, eksik değerleri doldurma ve one-hot encoding gibi işlemleri içeren bir dizi dönüştürücüyü bir araya getirdik. Bu şekilde, bu dönüştürücüleri kullanarak kategorik değişkenlerin işlenmesi otomatikleştirilir ve makine öğrenimi modellerine giriş olarak sağlanabilir.

```
✓ 0 [14] X = train_df.drop(columns=['fraud_flag'])  
sn. y = train_df['fraud_flag']
```

```
✓ 0 [15] numeric_features = X.select_dtypes(include=['int64', 'float64']).columns  
sn. numeric_transformer = Pipeline(steps=[  
    ('imputer', SimpleImputer(strategy='mean')),  
    ('scaler', StandardScaler())  
])
```

```
✓ 0 [16] categorical_transformer = Pipeline(steps=[  
sn.     ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),  
     ('onehot', OneHotEncoder(handle_unknown='ignore'))  
])
```

- XGBoost, gradient boosting algoritmasını kullanarak yüksek performanslı bir sınıflandırma modeli oluşturduk.

```
✓ 0 [17] preprocessor = ColumnTransformer(  
sn.   transformers=[  
      ('num', numeric_transformer, numeric_features),  
      ('cat', categorical_transformer, categorical_features)  
    ])
```

```
✓ 1 [18] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)  
sn.
```

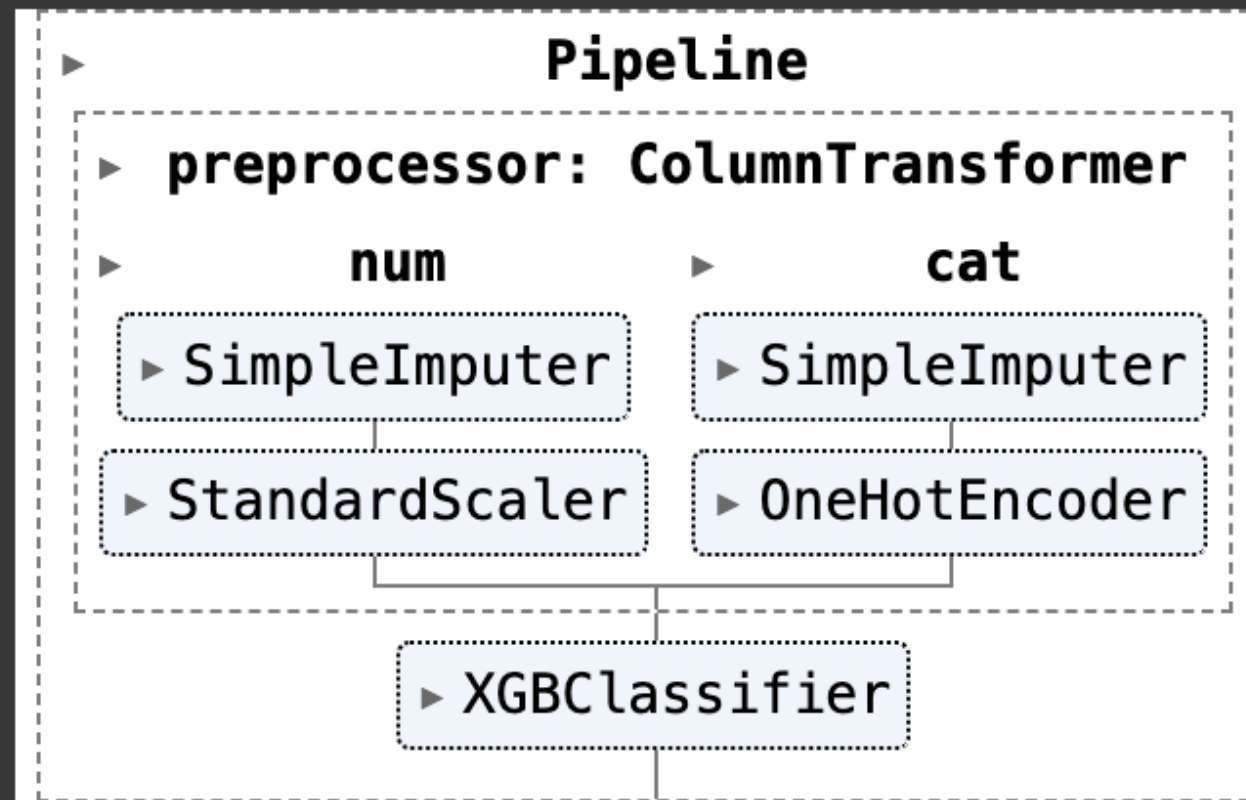
```
✓ 0 [29] # Modeli Tanımla  
sn.   model = Pipeline(steps=[('preprocessor', preprocessor),  
                              ('classifier', xgb.XGBClassifier(random_state=42))])
```

MODEL EĞİTİMİ


✓
16
sn.



```
# Modeli eđit  
model.fit(X_train, y_train)
```



Tahmin yapma

```
✓ 1 [41] # Test verisi üzerinde tahmin etme  
sn. test_pred_proba = model.predict_proba(test_df)[: , 1]  
  
✓ 0 [34] sample_submission = pd.read_csv('sample_submission.csv')  
sn.  
  
✓ 0 [35] sample_submission["fraud_flag"] = test_pred_proba  
sn.  
  
✓ 0  sample_submission.to_csv("sample_submission.csv", index=False)  
sn.
```

SONUÇ ÇIKTISI

sample_submission.csv ✕

ID	fraud_flag
38100	0.0014317476
13409	0.0009608214
56447	0.0026939092
70271	0.013678441
11531	0.0066271005
8721	0.0031703115
113775	0.053094175
80530	0.0041563823
105181	0.0067137484
13388	0.071035035
113870	0.0012535434
101582	0.019269105
15470	0.018728366
13608	0.0050902897
60937	0.008407673
88154	0.003920259
112605	0.0032920607
4227	0.0014409991
79347	0.00078795914
7992	0.0024331885
72041	9.1209404e-05
64093	0.00031991678

Teşekkürler

HAZIRLAYANLAR:

ZEYNEP KICIKOĞLU

YÜSRA KAPLAN