

**İstinye Üniversitesi**  
**Mühendislik Fakültesi**  
**Yazılım Mühendisliği Bölümü**  
**YAZ041 – Makine Öğrenmesi Operasyonları**  
**Dönem Proje Ödevi**

**Konu:** Kredi kartı üzerinde yapılan sahteciliğin önüne geçme ve yapılan sahtecilikleri fark etme

**Hazırlayanlar:**

- Mehmet Burak DURDU - 210911059
- Fatih Nadir ÖZDEMİR - 200701052
- Mert YALÇIN - 180701009
- Mehmet Emre BİNGÖL - 210911090

**Dersin Öğretmeni:**

- Alper ÖNER

## Projemizin genel amacı şöyledir:

Bu projedeki amacımız veri kümesi üzerinden kullandığımız metotlar ile oluşabilecek sahte işlemlerin önüne geçmek ve oluşan sahte işlemleri hemen fark edip önlem alabilmektir. Kredi kartı üzerinden yapılan sahtecilik herhangi bir sektör üzerinde hayati bir değere sahip olduğu için bu projeyi gerçekleştirmek istedik. Sistem üzerinden verilen veri kümesi, sahteciliğe karşı korunabiliyor ve bu sahtecilik anından çözülebiliyor ise hedefimize ulaşmış var sayabiliriz.

## Proje datasetimiz:



creditcard.csv

## Projede kullandığımız kütüphaneler:

pandas  
sklearn.metrics  
sklearn.model\_selection  
sklearn.linear\_model  
sklearn.metrics  
mlflow  
mlflow.sklearn  
matplotlib.pyplot  
seaborn  
sklearn.ensemble

## Projenin tasarımı ve anlatımı:

Öncelikle kullanacağımız modülleri kütüphanelerimizden çekiyoruz:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_recall_curve, auc
from sklearn.ensemble import RandomForestClassifier
import mlflow
import mlflow.sklearn
import seaborn as sns
```

Sonrasında database'imizden veriyi alıp işlemeye hazır hale getiriyoruz:

```
df = pd.read_csv("creditcard.csv")
X = df.drop('Class', axis=1)
y = df['Class']
df.info()
```

Ardından elimizdeki veriyi incelemek için modelleme modülleri ile grafikleri (table) oluşturuyoruz:

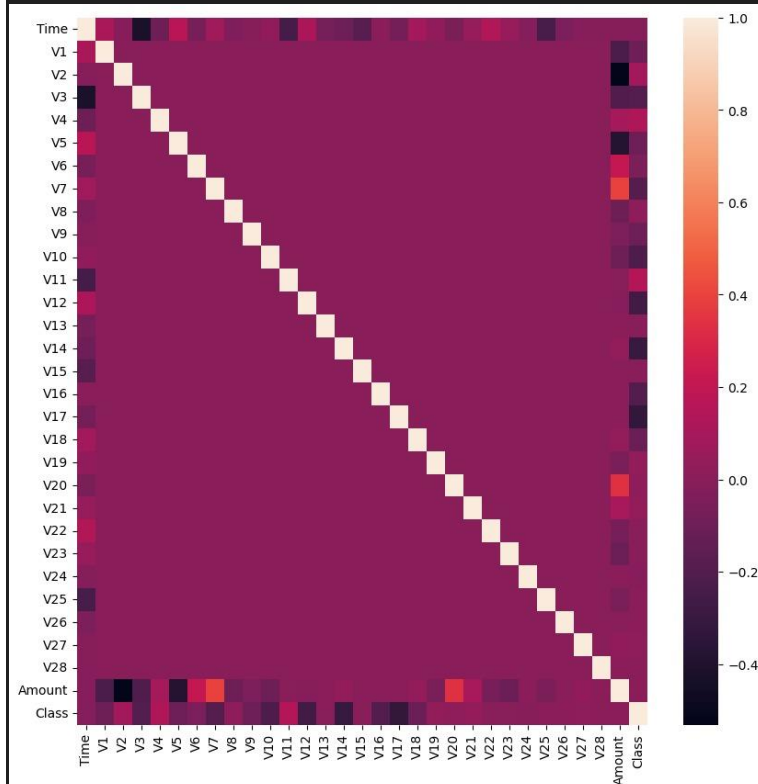
```
df.hist(figsize=(20,20))  
plt.show()
```

#ciktisi:



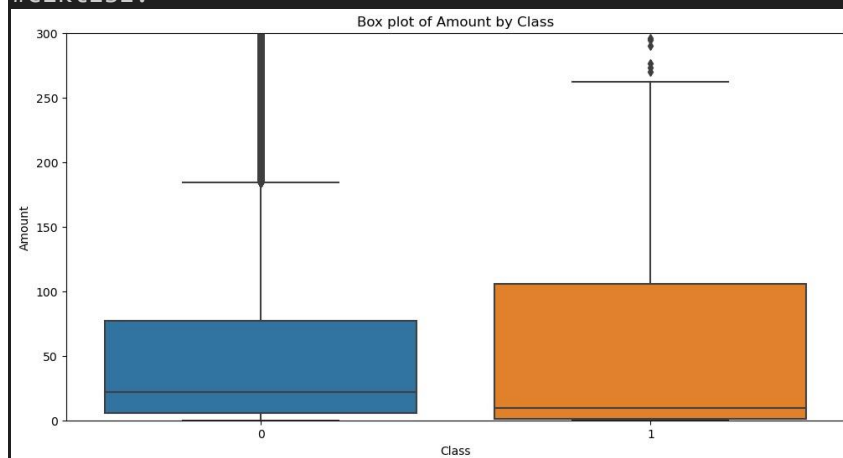
```
plt.figure(figsize=(10,10))  
sns.heatmap(df.corr())  
plt.show()
```

#ciktisi:



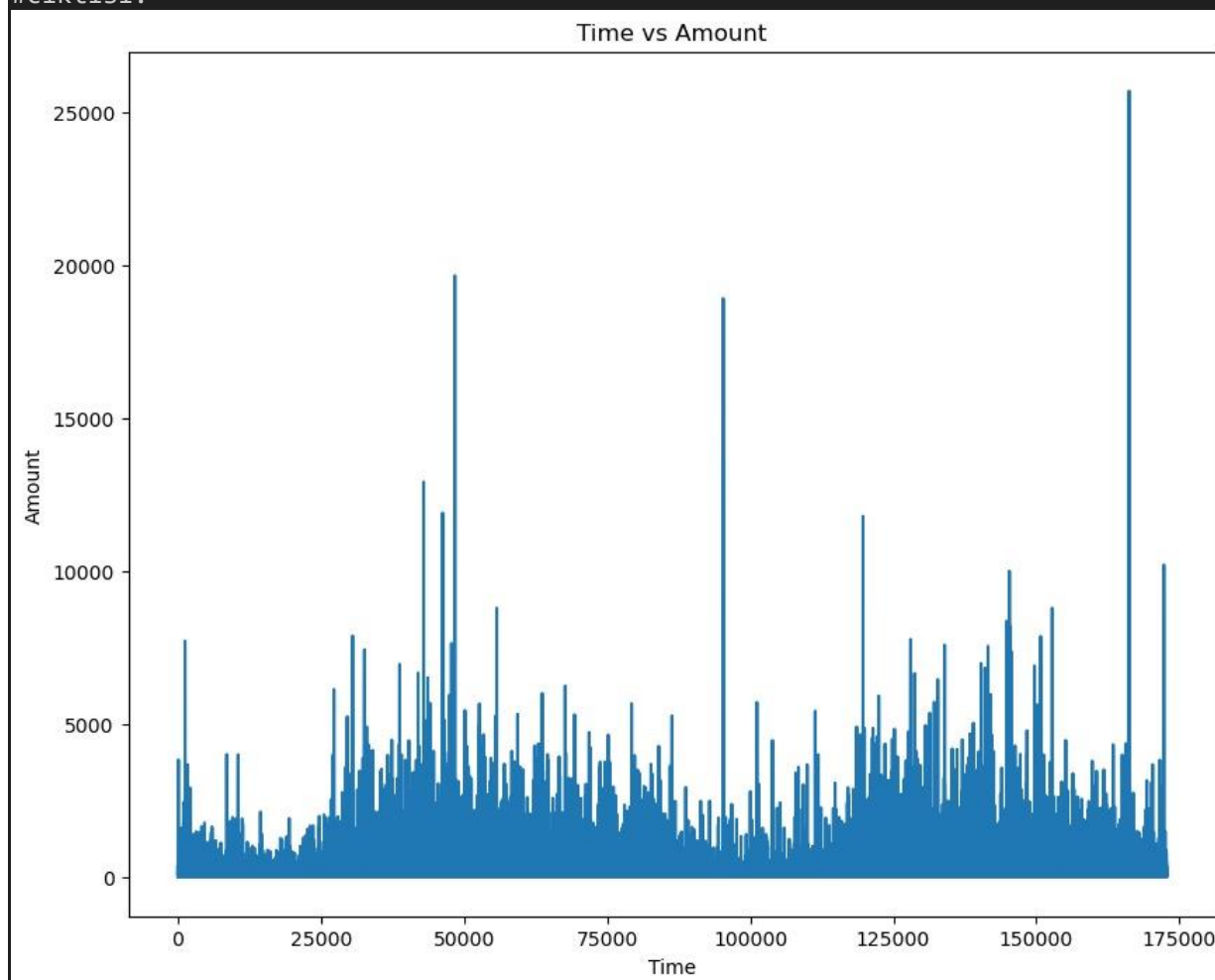
```
plt.figure(figsize=(12,6))
sns.boxplot(x="Class", y="Amount", data=df)
plt.title('Box plot of Amount by Class')
plt.ylim([0, 300])
plt.show()
```

#ciktisi:



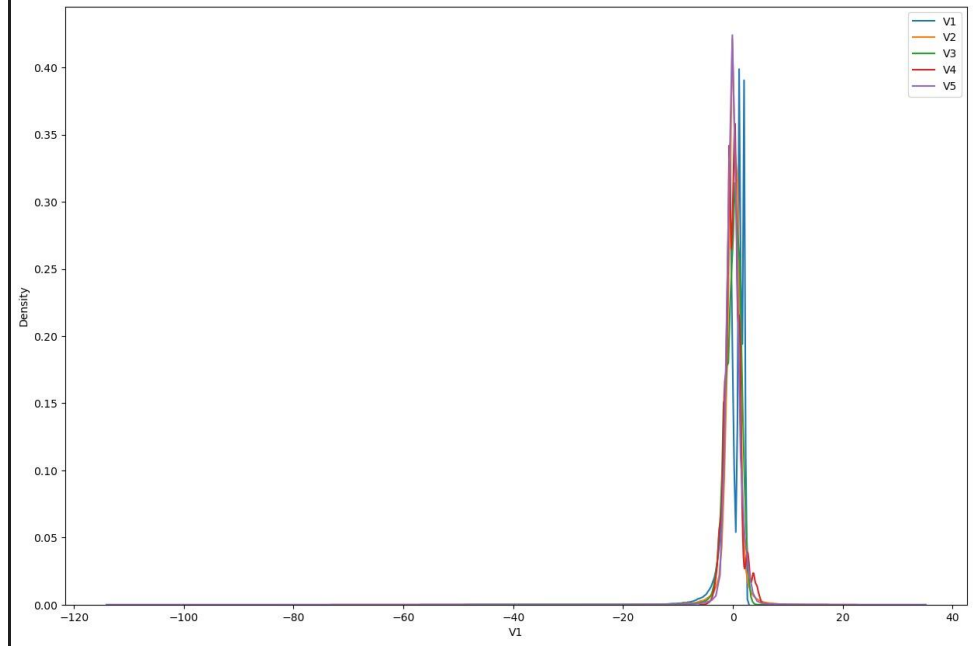
```
plt.figure(figsize=(10,8))
plt.plot(df['Time'], df['Amount'])
plt.title('Time vs Amount')
plt.xlabel('Time')
plt.ylabel('Amount')
plt.show()
```

#ciktisi:



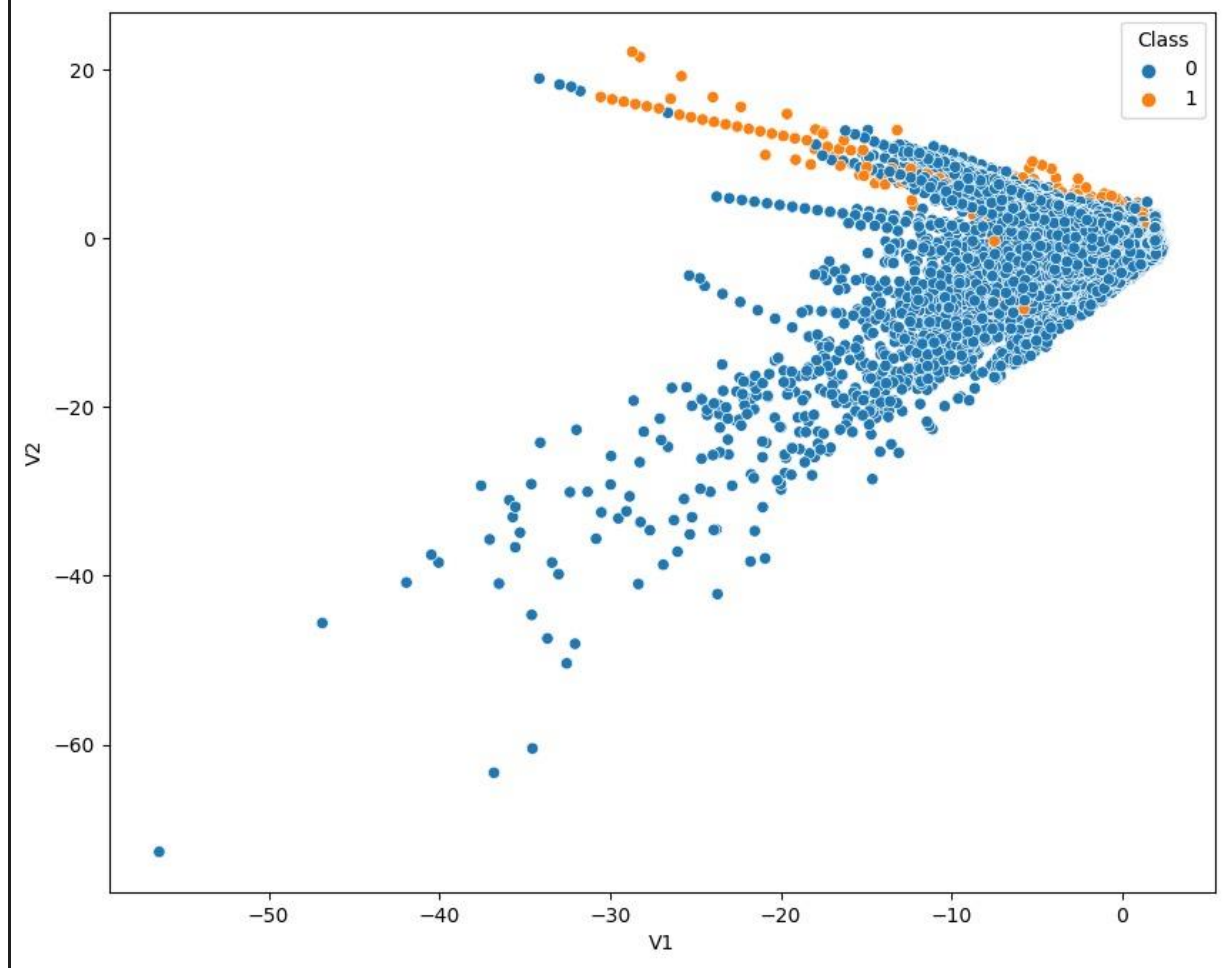
```
plt.figure(figsize=(15,10))
for column in df.columns[1:6]:
    sns.kdeplot(df[column], label=column)
plt.legend()
plt.show()
```

#ciktisi:



```
plt.figure(figsize=(10, 8))
sns.scatterplot(x="V1", y="V2", hue="Class", data=df)
plt.show()
```

#ciktisi:



Kodumuzun devamında tahmin ettireceğimiz değişkenleri belirleyip trainliyoruz:

```
X = df.drop('Class', axis=1)
y = df['Class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Artık mlflow u kullanmaya hazırız elimizde oluşturduğumuz modeller (table) ve eğitmeye hazır bir kod var:

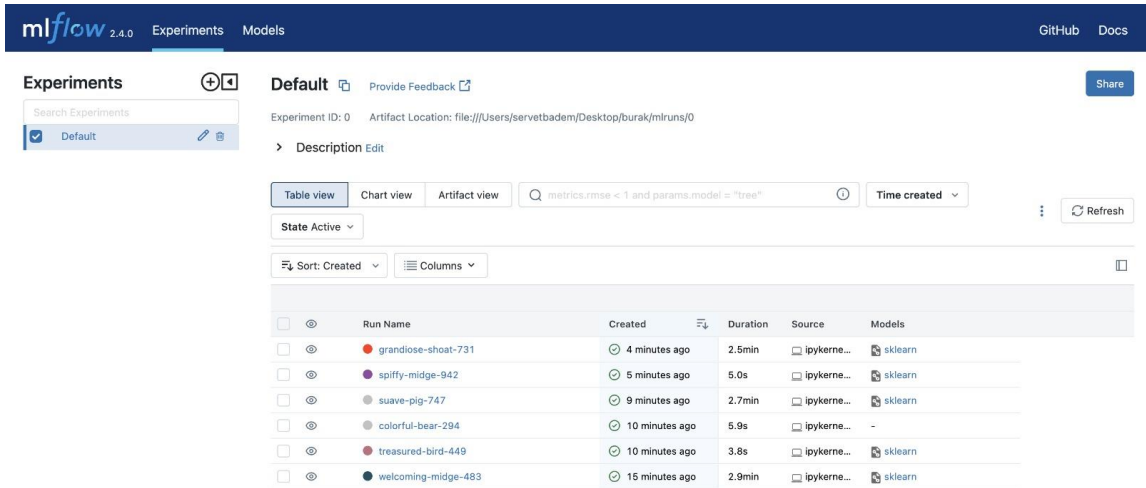
```
mlflow.start_run()

# Modeli oluşturma
model = RandomForestClassifier()

# Modeli eğitme
model.fit(X_train, y_train)

y_scores = model.predict_proba(X_test)[:, 1]
precision, recall, thresholds = precision_recall_curve(y_test, y_scores)
auc_score = auc(recall, precision)
accuracy = model.score(X_test, y_test)
mlflow.log_metric('AUC', auc_score)
mlflow.log_metric("accuracy", accuracy)
mlflow.sklearn.log_model(model, "model")
```

MLflow üzerine işlediğimiz modellerin görünümü:



Run Name	Created	Duration	Source	Models
grandiose-shoot-731	4 minutes ago	2.5min	ipykerne...	sklearn
spiffy-midge-942	5 minutes ago	5.0s	ipykerne...	sklearn
suave-pig-747	9 minutes ago	2.7min	ipykerne...	sklearn
colorful-bear-294	10 minutes ago	5.9s	ipykerne...	-
treasured-bird-449	10 minutes ago	3.8s	ipykerne...	sklearn
welcoming-midge-483	15 minutes ago	2.9min	ipykerne...	sklearn

Son olarak MLflow içinde modellerimizi LogisticRegression yöntemi ile eğitip tahmin ai ımızı tamamlıyoruz:

```
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
y_scores = model.predict_proba(X_test)[:, 1]
precision, recall, thresholds = precision_recall_curve(y_test, y_scores)
auc_score = auc(recall, precision)
accuracy = model.score(X_test, y_test)

mlflow.log_metric('AUC', auc_score)
mlflow.log_metric("accuracy", accuracy)

mlflow.sklearn.log_model(model, "model")

mlflow.end_run()

!mlflow ui
```

## Kaynakça:

<https://coderspace.io>

<https://www.kaggle.com>

<https://mlflow.org/docs/latest/models.html>

<https://chat.openai.com>

<https://www.miuul.com/>

<https://medium.com/deeplearningmadeeasy/mlflow-for-mlops-414be83b33d2>

[MLflow - Modeling | Censius MLOps Tools](#)