

ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



PROJE RAPORU

(BLM3522-A) BULUT BİLİŞİM VE UYGULAMALARI

Perizat SAGYNBEKOVA (21290895)

Derda SİNA GÜNAY (20291274)

Rüstem TUKHBETOV (21291001)

GitHub (<https://github.com/Impasbaa/BulutBilisim>)

YouTube (<https://youtu.be/MvMmtNjSvO0>)

12.05.2025

PROJE 2: AKILLI VERİ ANALİTİĞİ VE MAKİNE ÖĞRENMESİ UYGULAMASI

Problem: Makine Öğrenimi API'si (Application Programming Interface)

Backend: Python

Makine Öğrenmesi Kütüphaneleri: Scikit-learn

Veritabanı: Veri, doğrudan bir CSV dosyasından alınıyor

Bulut Platformu: AWS (Amazon Web Services) - Sagemaker

Veri Seti: Iris

AMAÇ

Bu projenin amacı, makine öğrenmesi ile oluşturulmuş bir modeli kullanarak tahmin işlemlerini gerçekleştirebilen bir sistem kurmaktır. Bu sistemde model, AWS SageMaker üzerinde barındırılmakta ve bu modele veri gönderip sonuç almak için AWS Lambda fonksiyonu kullanılmaktadır. Lambda fonksiyonu, bir API Gateway ile bağlantılıdır. Böylece kullanıcılar, oluşturulan API üzerinden veri göndererek tahmin sonucunu alabilmektedir.

Proje sayesinde, eğitilmiş bir modeli doğrudan internet üzerinden erişilebilecek şekilde yayınlama, bu modeli Lambda ve API Gateway ile bağlayarak otomatik çalışan bir sistem oluşturma hedeflenmiştir. Bu yapı sayesinde hem sunucusuz (serverless) bir ortamda çalışılmış hem de ölçeklenebilir ve düşük maliyetli bir çözüm elde edilmiştir.

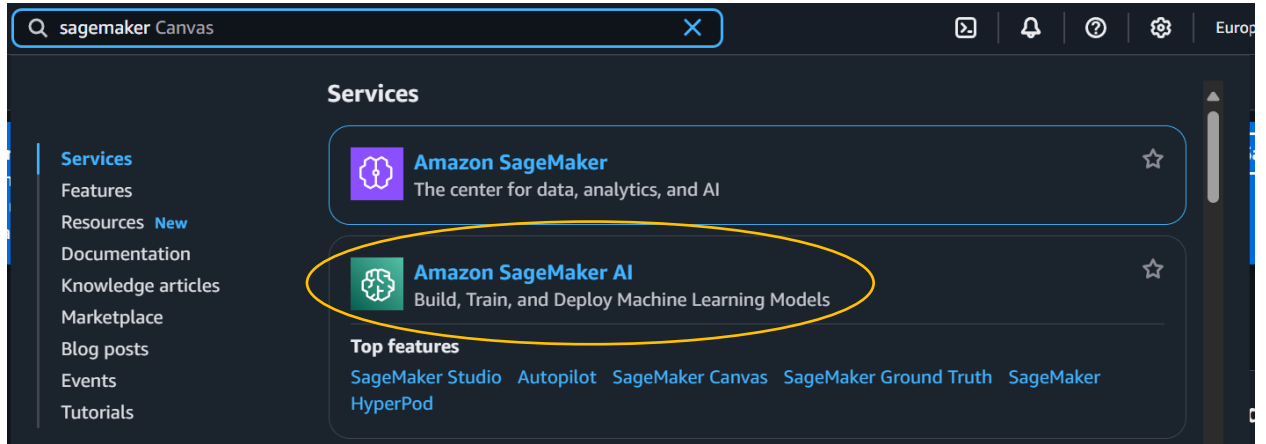
GİRİŞ

Bu projede, gerçek bir veri kümesi üzerinde çalışarak bir makine öğrenmesi modeli geliştirme süreci gerçekleştirilmiştir. Modelin eğitimi, Amazon SageMaker gibi bulut tabanlı bir platform üzerinde gerçekleştirilmiş ve ardından yine bulut ortamında modelin dağıtımı (deployment) sağlanmıştır. Böylece model yalnızca eğitim amacıyla değil, aynı zamanda canlı ortamlarda da kullanılabilir hale getirilmiştir.

Veri ön işleme, analiz ve model eğitimi adımlarının ardından, veriden anlamlı bilgiler çıkarma ve belirli girişlere karşılık tahminlerde bulunma gibi temel makine öğrenmesi hedeflerine ulaşılmıştır. Ayrıca, Lambda ve API Gateway servisleri kullanılarak modelin web üzerinden erişilebilir olması sağlanmış ve uçtan uca bir makine öğrenmesi çözümü tamamlanmıştır.

Bu çalışma sayesinde hem makine öğrenmesi algoritmalarının pratikte nasıl kullanıldığı hem de bulut bilişim teknolojilerinin bu süreçlerdeki önemi konusunda değerli deneyimler kazanılmıştır.

Amazon SageMaker ve Notebook Instance:



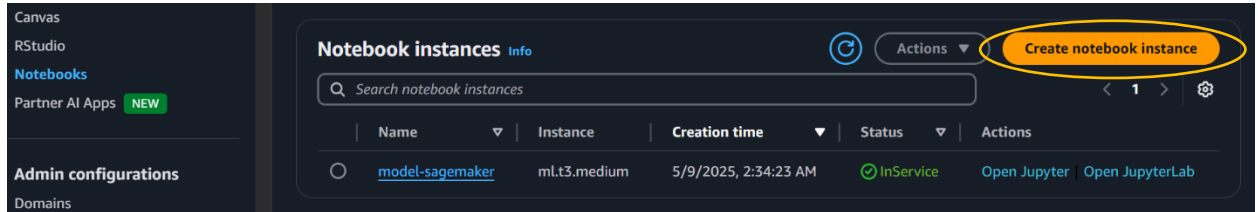
Şekil 1. Amazon SageMaker AI Hizmeti

AWS SageMaker, makine öğrenmesi projelerini baştan sona yönetmemizi sağlayan bir bulut hizmetidir.

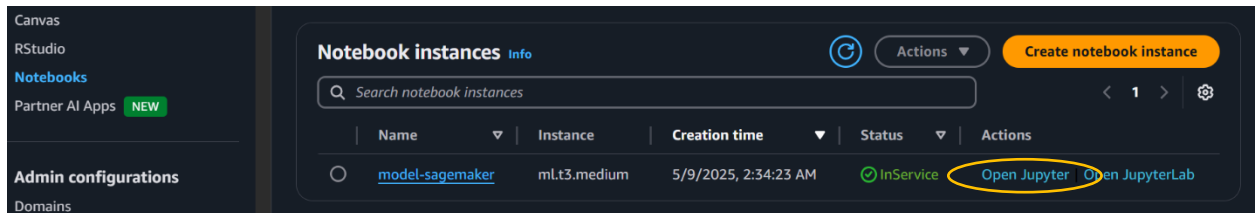
- Veri hazırlama
- Model Eğitme
- Hiperparametre Optimizasyonu
- Model Dağıtım

İşlemlerinin hepsini AWS SageMaker hizmetiyle yapmamız mümkündür.

Proje kapsamında amacımız bir model eğitip tahmin işlemleri gerçekleştirebildiğimiz bir sistem kurmak ve model dağıtımını yapmak olduğu için AWS SageMaker hizmetini kullanmaya karar verdik.



Şekil 2. Notebook Instance Oluşturmak.

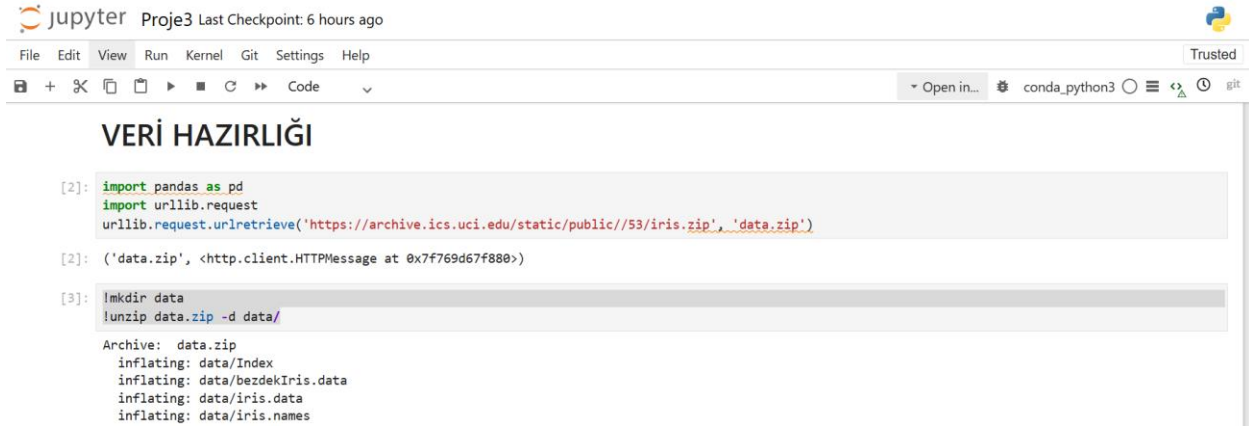


Şekil 3. Jupyter Notebook

Amazon SageMaker'ın en önemli özelliklerinden birisi içerisinde notebook instance oluşturma imkânımız olmasıdır.

Notebook instance oluşturmaya AWS'nin avantajlarından yararlanarak kullanabildiğimiz bir jupyter notebook oluşturmak olarak ifade edebiliriz, bu sayede bulut üzerinde veri seti yükleme, veri işleme ve model eğitme işlemlerini yapabileceğimiz bir ortama sahip olabiliyoruz.

Veri Hazırlığı:



```
[2]: import pandas as pd
import urllib.request
urllib.request.urlretrieve('https://archive.ics.uci.edu/static/public/53/iris.zip', 'data.zip')

[2]: ('data.zip', <http.client.HTTPMessage at 0x7f769d67f880>)

[3]: !mkdir data
!unzip data.zip -d data/

Archive: data.zip
  inflating: data/Index
  inflating: data/bezdekIris.data
  inflating: data/iris.data
  inflating: data/iris.names
```

Şekil 4. Veri Setini Çekme İşlemi

Bu aşamada:

- Urllib kütüphanesini kullanarak yukarıdaki görselde gözüken web sitesine istekte gönderdik ve model eğitmek için kullanacak olduğumuz iris veri setini indirdik.
- İndirdiğimiz veri setini unzip komutunu kullanarak ayıkkladık.

```
[57]: # veriyi okumak
df = pd.read_csv('data/iris.data', header=None)
# data = pd.read_csv('data/iris.data', header = None)

# sayısal değerlere dönüştürmek
data[4] = data[4].replace('Iris-setosa', 0)
data[4] = data[4].replace('Iris-virginica', 1)
data[4] = data[4].replace('Iris-versicolor', 2)

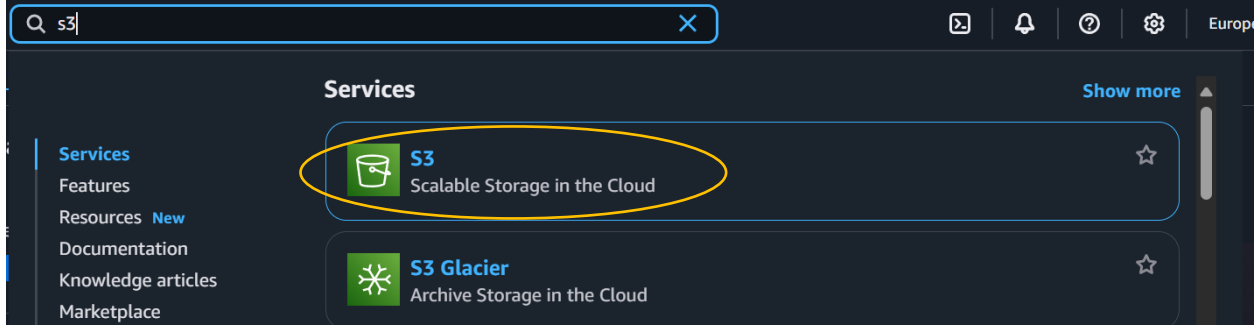
# karıştırmak
data = data.sample(frac = 1).reset_index(drop = True)

# bölmek (eğitim ve doğrulama veri kümeleri)
# %80 eğitim veri seti / %20 doğrulama veri seti
train_data = data[:120]
val_data = data[120:]
```

Şekil 5. Veri Ön İşleme Süreci

- Model eğitmeden önce veri setini inceledik ve gerekli veri ön işleme işlemlerini gerçekleştirerek eğitim ve doğrulama veri setlerini oluşturduk.

S3 ve Bucket Düzenlemeleri(Depolama Düzenlemeleri):



Şekil 6. S3'e Giriş

Amazon S3(Simple Storage Service), AWS'nin nesne tabanlı depolama hizmetidir.

Eğitim ve doğrulama dosyalarımızı SageMaker'ın eğitim işlerinin doğrudan okuyabileceği şekilde buluta yüklemek model eğitim sürecini sorunsuz yürütmemizi sağlayacak.

S3'e girerek "sagemaker-kurma-ve-dagıtma-modeli" isminde bir bucket oluşturduk.

(Bucket AWS S3'te kullandığımız klasör benzeri bir yapıdır.)

VERİNİN S3'E TAŞINMASI

```
[58]: import boto3
      bucket_name = 'sagemaker-kurma-ve-dagıtma-modeli'

      data = pd.concat([data[4], data.drop(columns=[4])], axis=1)
      train_data.to_csv('data.csv', header = False, index = False)
      key = 'data/train/data'
      url = 's3://{}/{}'.format(bucket_name, key)
      boto3.Session().resource('s3').Bucket(bucket_name).Object(key).upload_file('data.csv')

      val_data.to_csv('data.csv', header = False, index = False)
      key = 'data/val/data'
      url = 's3://{}/{}'.format(bucket_name, key)
      boto3.Session().resource('s3').Bucket(bucket_name).Object(key).upload_file('data.csv')
```

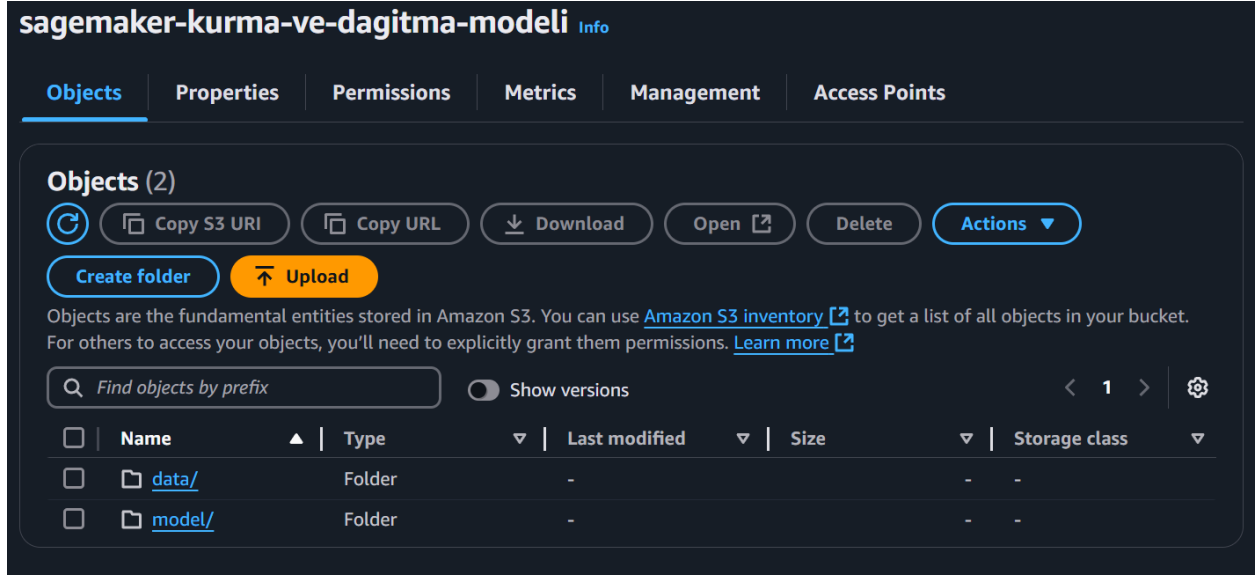
[05/11/25 16:38:10] INFO Found credentials from IAM Role: BaseNotebookInstanceEc2InstanceRole credentials.py:1132

INFO Found credentials from IAM Role: BaseNotebookInstanceEc2InstanceRole credentials.py:1132

Şekil 7. AWS S3'e Veri Taşıma

- Boto3 kütüphanesiyle AWS S3'e bağlandık.
- bucket_name değişkeniyle hedef bucket'ı belirledik, data üzerinde sütun düzenlemesi yaptıktan sonra train_data ve val_data'yı data.csv dosyasına kaydettik.

- “data/train/data” ve “data/val/data” anahtarlarıyla S3’teki bucket’ımıza yükleme işlemini gerçekleştirdik.
- Boto3, IAM rolü üzerinden otomatik kimlik doğrulama sağladı.



Şekil 8. Data Klasörü

Ekran görüntülerimizi süreci tamamladıktan sonra aldığımız için model klasörü de gözüktüyor ancak normalde bu aşamadayken sadece data klasörüne sahiptik.

Docker ve Eğtilecek Modelin Şablonu:

MODEL OLUŞUMU

```
[48]: import sagemaker
      from sagemaker.amazon.amazon_estimator import get_image_uri
      from sagemaker import get_execution_role
      from sagemaker import image_uris

      key = 'model/xgb_model'
      s3_output_location = 's3://{}/{}'.format(bucket_name, key)

      container = image_uris.retrieve(
          framework='xgboost',
          region=boto3.Session().region_name,
          version='1.7-1')
      xgb_model = sagemaker.estimator.Estimator(
          # image_uri=container,
          get_image_uri(boto3.Session().region_name, 'xgboost'),
          role = get_execution_role(),
          train_instance_count = 1,
          train_instance_type = 'ml.m5.xlarge',
          train_volume_size = 5,
          output_path = s3_output_location,
          sagemaker_session = sagemaker.Session())

      xgb_model.set_hyperparameters(
          max_depth = 5,
          eta = 0.2,
          gamma = 4,
          min_child_weight = 6,
          # silent = 0,
          verbosity = 1,
          objective = 'multi:softmax',
          num_class = 3,
          num_round = 10)
```

```
[05/11/25 16:18:07] INFO Ignoring unnecessary instance type: None. image_uris.py:530

WARNING The method get_image_uri has been renamed in sagemaker>=2. deprecations.py:34
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.

INFO Ignoring unnecessary instance type: None. image_uris.py:530

WARNING train_instance_count has been renamed in sagemaker>=2. deprecations.py:34
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.

WARNING train_instance_type has been renamed in sagemaker>=2. deprecations.py:34
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.

WARNING train_volume_size has been renamed in sagemaker>=2. deprecations.py:34
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```

Şekil 9. Docker ve Estimator

Sagemaker'ın içinde bulunan Estimator, eğitim sürecini tanımladığımız bir python nesnesidir.

- Container oluşturarak stabil bir çalışma ortamı kurguladık ve model eğitiminde hangi algoritmanın kullanılacağını tanımladık.
- Estimator ile oluşturduğumuz Container ortamında hangi ayarlarla model eğitimini yapacağımızı belirttik.

Bu aşamada Hangi algoritma, hangi veri, kaç tane ve ne tip sunucu kullanılacağı, hiperparametreler gibi tanımlamalarımızı yaptık.

Model Eğitim ve Dağıtım Süreci:

MODEL EĞİTİMİ

```
[49]: import numpy as np
      labels = data[4]
      unique_labels = sorted(np.unique(labels))
      label_mapping = {label: idx for idx, label in enumerate(unique_labels)}
      data[4] = np.vectorize(label_mapping.get)(labels)
      np.sort(np.unique(data[4]))

[49]: array([0, 1, 2])

[53]: import pandas as pd

      df_train = pd.read_csv('data.csv', header=None)
      print("Unique labels in training set:", df_train[0].unique())

      df_val = pd.read_csv('data.csv', header=None)
      print("Unique labels in validation set:", df_val[0].unique())

Unique labels in training set: [6.4 6.3 5.4 6.7 5.5 7.7 5.2 5.8 4.8 4.9 4.6 6.2 5.1 5.7 6.1 6.9 6. 5.6
 5. 6.6 4.4]
Unique labels in validation set: [6.4 6.3 5.4 6.7 5.5 7.7 5.2 5.8 4.8 4.9 4.6 6.2 5.1 5.7 6.1 6.9 6. 5.6
 5. 6.6 4.4]

[60]: from sagemaker.inputs import TrainingInput

      train_data = 's3://{}/{}/{}'.format(bucket_name, 'data/train')
      val_data = 's3://{}/{}/{}'.format(bucket_name, 'data/val')

      # train_channel = sagemaker.session.s3_input(train_data, content_type = 'text/csv')
      train_channel = TrainingInput(train_data, content_type='text/csv')
      # val_channel = sagemaker.session.s3_input(val_data, content_type = 'text/csv')
      val_channel = TrainingInput(val_data, content_type='text/csv')

      data_channels = {'train': train_channel, 'validation': val_channel}
      xgb_model.fit(inputs = data_channels)

[5]#011train-merror:0.008333#011validation-merror:0.066667
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 0 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 4 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 4 extra nodes, 2 pruned nodes, max_depth=2
[6]#011train-merror:0.008333#011validation-merror:0.066667
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 0 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 4 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 4 extra nodes, 2 pruned nodes, max_depth=2
[7]#011train-merror:0.008333#011validation-merror:0.066667
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 0 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 4 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 4 extra nodes, 2 pruned nodes, max_depth=2
[8]#011train-merror:0.016667#011validation-merror:0.066667
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 4 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 4 extra nodes, 2 pruned nodes, max_depth=2
[8]#011train-merror:0.016667#011validation-merror:0.066667
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 0 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 2 extra nodes, 4 pruned nodes, max_depth=1
[16:40:41] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 4 extra nodes, 4 pruned nodes, max_depth=2
[9]#011train-merror:0.008333#011validation-merror:0.066667

2025-05-11 16:41:00 Completed - Training job completed
Training seconds: 89
Billable seconds: 89

[51]: df = pd.read_csv('data.csv')
      print(df.head())
      print(df.iloc[:, 0].unique())

      6.4 3.2 5.3 2.3 1
0 6.3 3.3 4.7 1.6 2
1 5.4 3.0 4.5 1.5 2
2 6.7 3.3 5.7 2.1 1
3 5.5 2.6 4.4 1.2 2
4 7.7 2.8 6.7 2.0 1
[6.3 5.4 6.7 5.5 7.7 6.4 5.2 5.8 4.8 4.9 4.6 6.2 5.1 5.7 6.1 6.9 6. 5.6
 5. 6.6 4.4]
```

Şekil 10. Model Eğitimi

Birkaç veri düzenleme işlemi yaptık ve modelimizi eğittik


```
MODELİN DAĞITIMI

[61]: xgb_predictor = xgb_model.deploy(initial_instance_count = 1,
                                         instance_type = 'ml.m5.xlarge')

[05/11/25 16:45:47] INFO Creating model with name: xgboost-2025-05-11-16-45-47-848 session.py:4094

[05/11/25 16:45:48] INFO Creating endpoint-config with name xgboost-2025-05-11-16-45-47-848 session.py:6019

INFO Creating endpoint with name xgboost-2025-05-11-16-45-47-848 session.py:4841

-----!
```

Şekil 11. Model Dağıtımı

Gerçek zamanlı bir tahmin sunucusu kurduk (Endpoint oluşturduk).

Bu sayede dışarıdan istek atılıp, tahmin sonucunun elde edilebileceği sistemi oluşturmuş olduk.

Name	Modified	File Size
data	5 hours ago	
Proje3.ipynb	10 seconds ago	50 KB
data.csv	5 hours ago	540 B
data.zip	7 hours ago	3.7 KB
train_data.csv	5 hours ago	2.1 KB
val_data.csv	5 hours ago	540 B

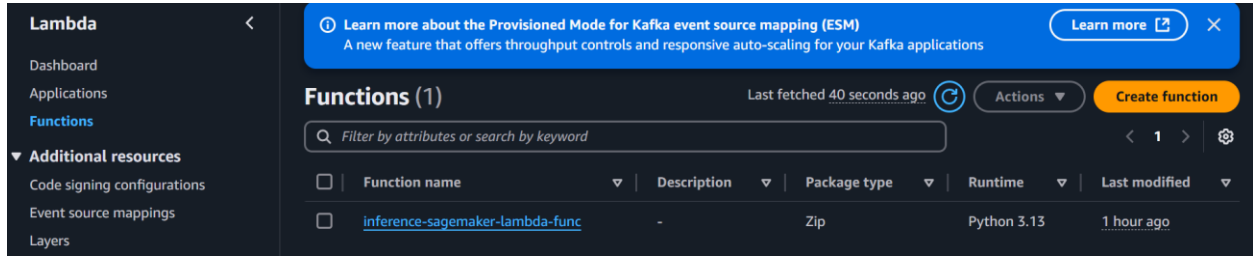
Şekil 12. Oluşan Dosya ve Klasörler

Training Plans	Endpoints
Inference	<input type="text" value="Search endpoints"/>
Compilation jobs	
Marketplace model packages	
Models	
Endpoint configurations	
Endpoints	
Batch transform jobs	
Shadow tests	
Inference Recommender	

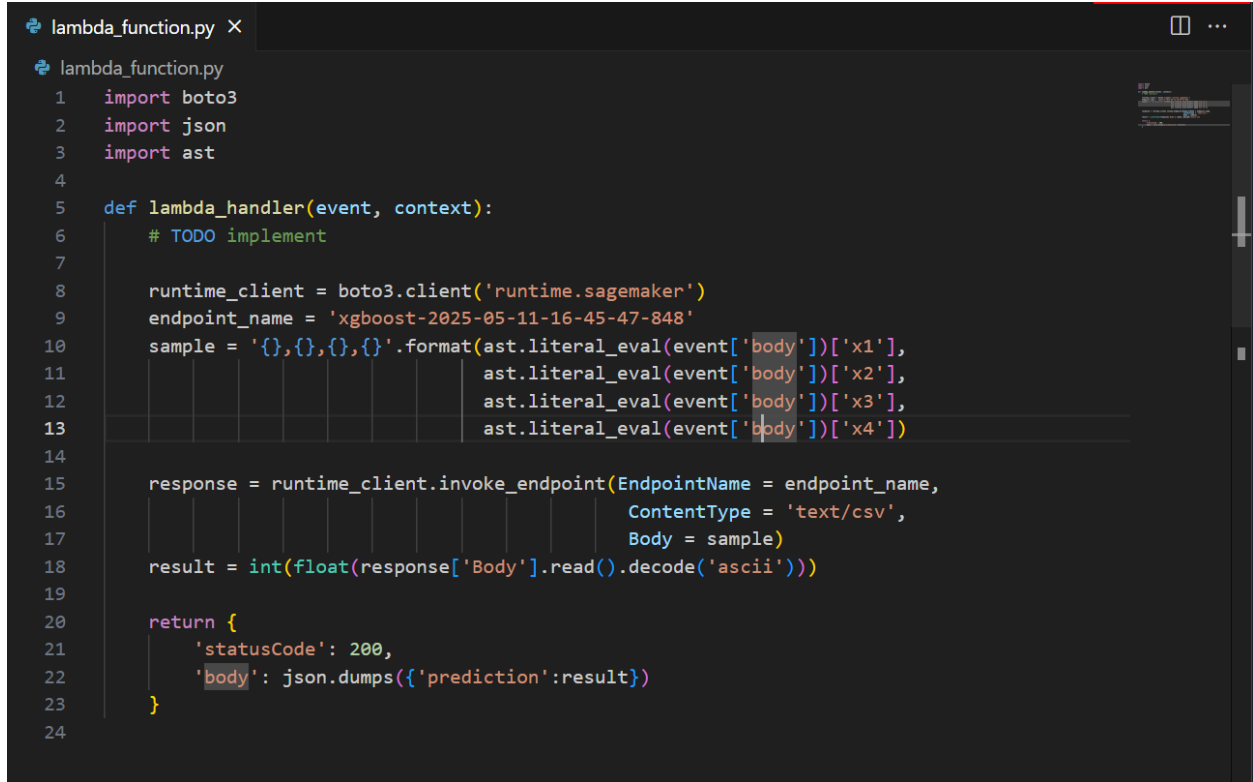
Name	ARN	Creation time	Status	Last updated
xgboost-2025-05-11-16-45-47-848	arn:aws:sagemaker:eu-north-1:992757690971:endpoint/xgboost-2025-05-11-16-45-47-848	5/11/2025, 7:45:49 PM	InService	5/11/2025, 7:49:57 PM

Şekil 13. İki Aşama Önce Oluşturduğumuz Endpoint

API Gateway Düzenlemeleri ve Lambda Fonksiyonu:

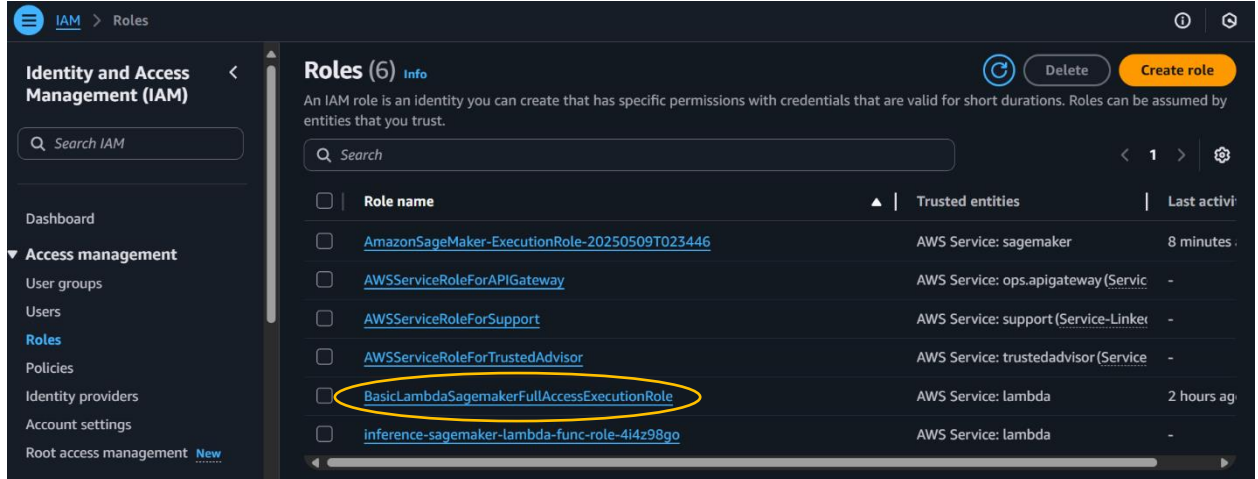


Şekil 14. Oluşturulan Lambda Fonksiyonu



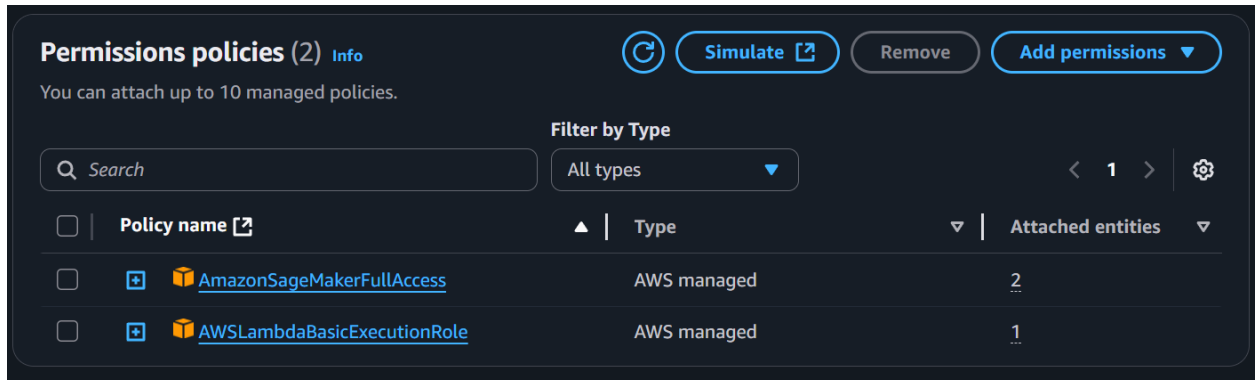
Şekil 15. Lambda Fonksiyonunun Kodu

Lambda Fonksiyonu yazdığımız kod sayesinde gelen HTTP isteklerinin nasıl yönetileceğini belirler. Hemen isteği alır eğittiğimiz modelin olduğu endpointe yönlendirir, tahmin sonucunu alır uygun formata çevirir ve istemciye tahmin sonucunu geri gönderir. Ayrıca sunucuyu sadece istek atıldığı zamanlarda aktifleştirerek maliyetleri ve enerji tüketimini azaltır.



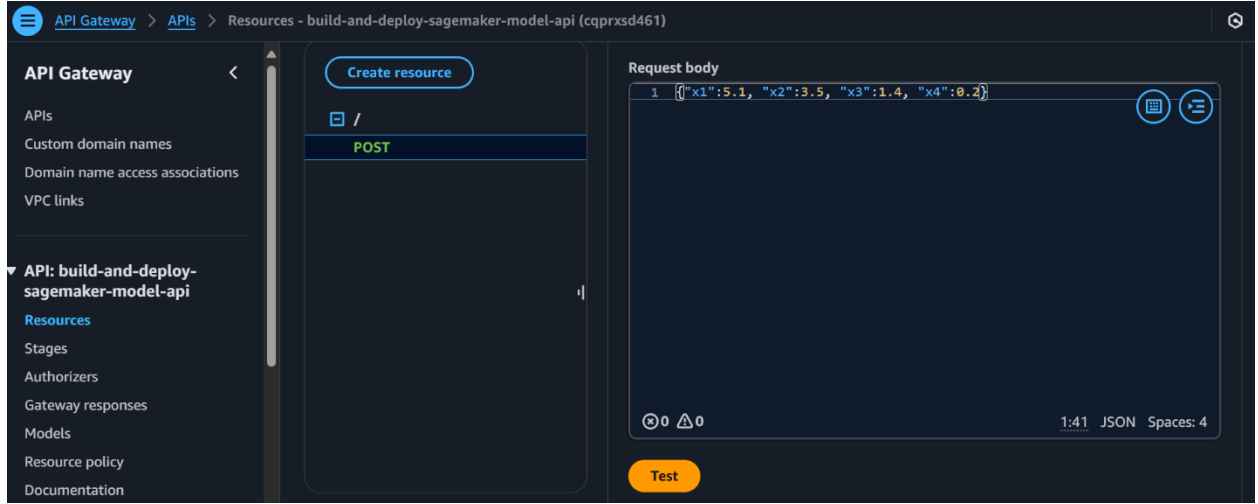
Şekil 16. IAM Rol Oluşturma

Lambda kodu çalıştığında SageMaker endpoint'ine istek atabilsin, S3'ten veri okuyup CloudWatch'a log yazabilsin diye bu tam erişim yetkilerine sahip rolü oluşturduk.



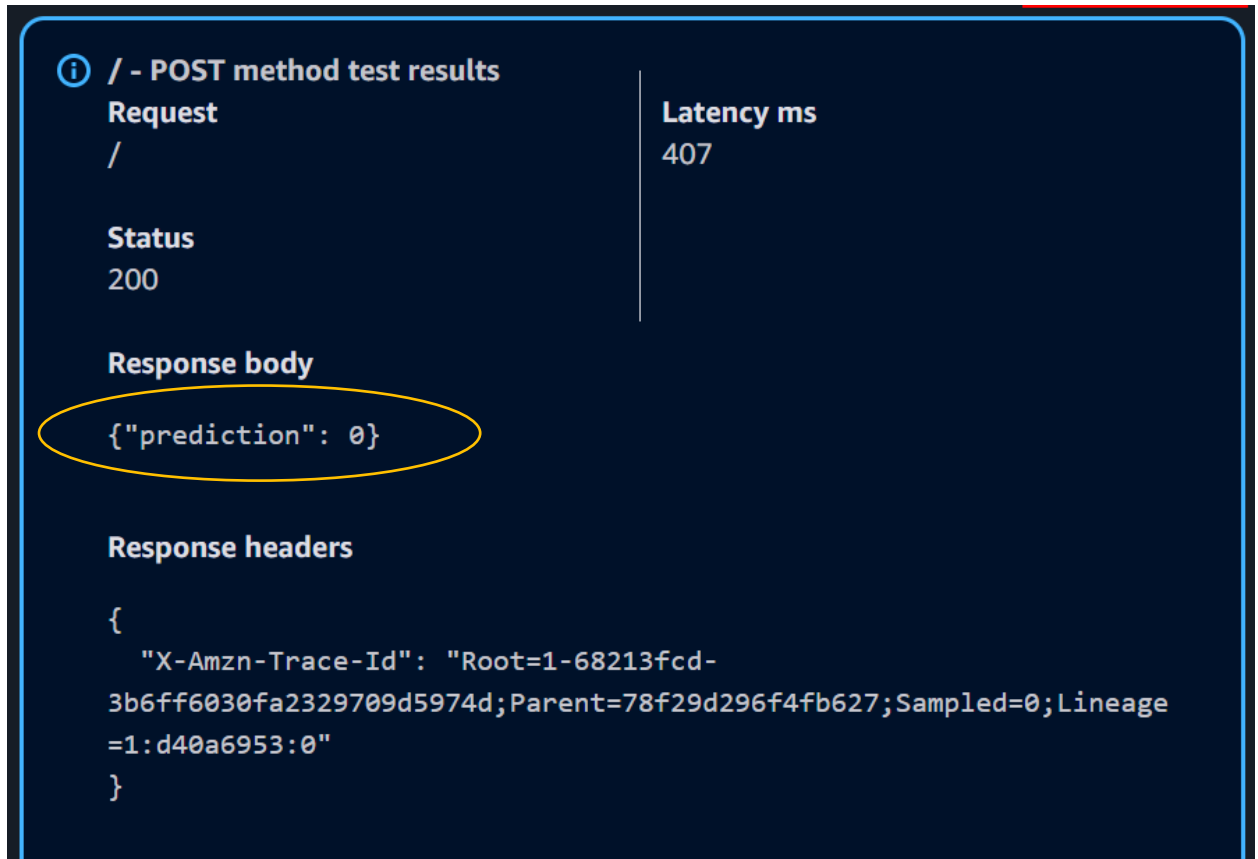
Şekil 17. Rol için Seçilen Politikalar

Uygulama:

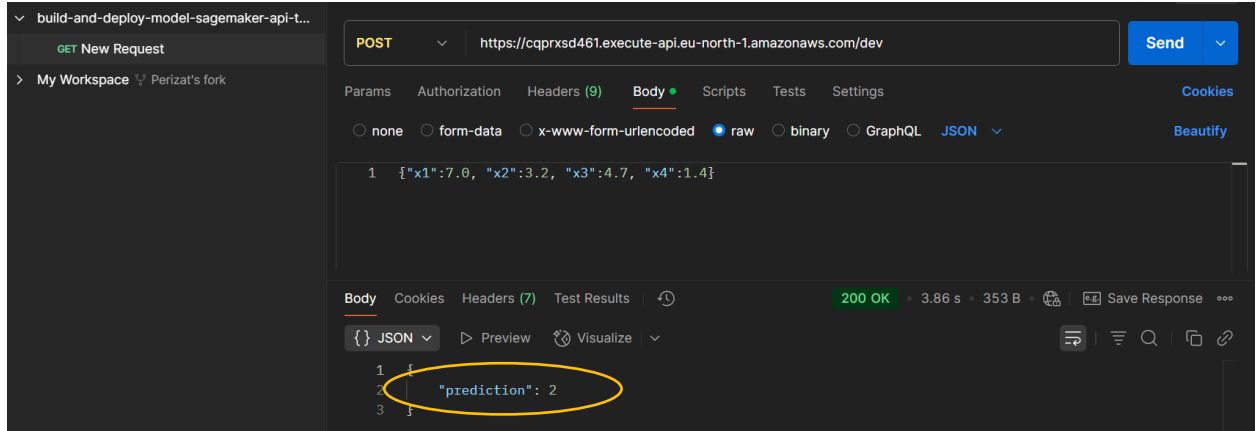


Şekil 18. API Gateway POST Metodu

- Oluşturduğumuz endpoint'e istek attık.



Şekil 19. Tahmin Çıktısı



Şekil 20. All-In-One API Platformu (Postman) Üzerinde Deneme

Sonuç olarak, bulut ortamında dağıtık çalışabilen ve kendisine istek gönderildiğinde tahmin sonuçları döndürebilen bir model geliştirdik.

KAYNAKLAR

1. **Veri Seti:** <https://archive.ics.uci.edu/dataset/53/iris>
2. **XGBoost Hiperparametreleri:** https://docs.aws.amazon.com/sagemaker/latest/dg/xgboost_hyperparameters.html