

# Google Colab (Google Colaboratory) ile Makine Öğrenmesi ve Derin Öğrenme Sklearn & Keras Test Ortamı Hazırlanması

---

FERDİ GÜL –

[contact@ferdigul.com](mailto:contact@ferdigul.com)

# İÇİNDEKİLER

---

## 1. Colab Nedir?

## 2. Geliştirme Ortamının Hazırlanması

2.1 Google Drive'da çalışma klasörü oluşturma

2.2 Google Drive'da çalışma klasörü oluşturma:

2.3 Yeni Notebook Ortamının Kurulması

a. Colab Hesabına Giriş

b. Yeni Notebook hesabı oluşturma

c. Google Drive Çalışma ortamı Colab Bağlaması

2.4 GPU Ayarlaması

2.5 Run the sample

## 3. Projeye Hazır Hale Getirme

## 4. SKLEARN Kütüphanesi ile Doğrusal Regresyon Analizi

## 5. KERAS Convolutional Neural Network (CNN) ile MNIST Rakım Tanıma Problemini Çözme

5.1 !python komutu ile çalıştırma

5.2 Jupyter Notebook ile çalıştırma

## KAYNAKÇA

# 1. Colab

---



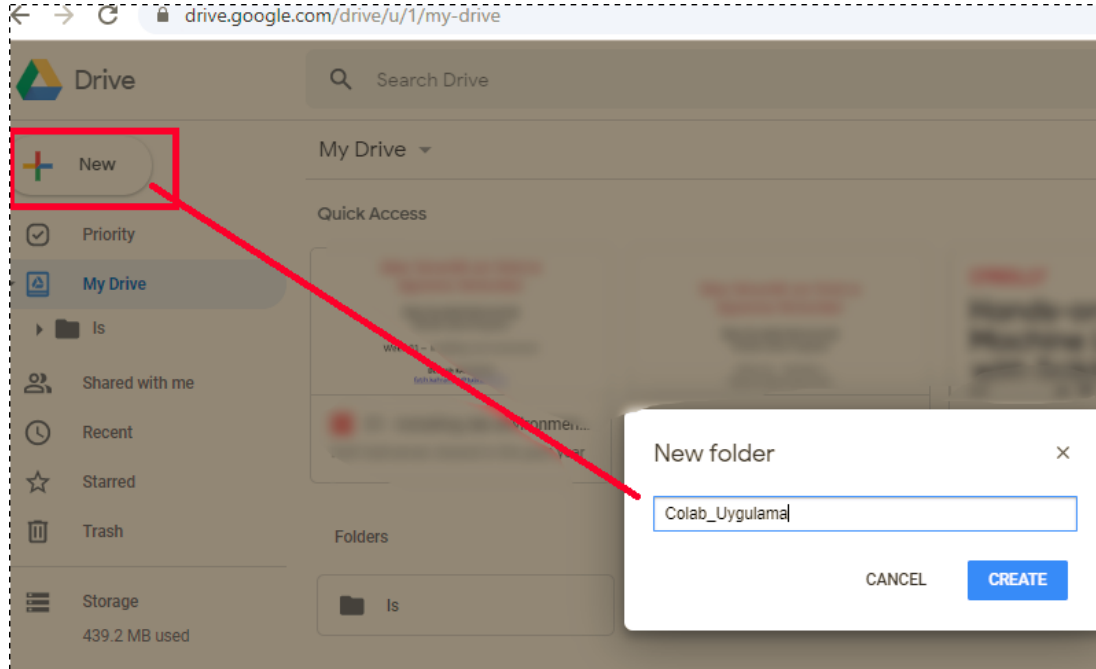
- Ücretsiz Araştırma Topluluğu
- Chrome, Firefox, Safari
- Jupyter Notebook
- Tesla K80 GPU – Free
- Python2.7 ve python3.6

---

## 2. Geliştirme Ortamının Hazırlanması

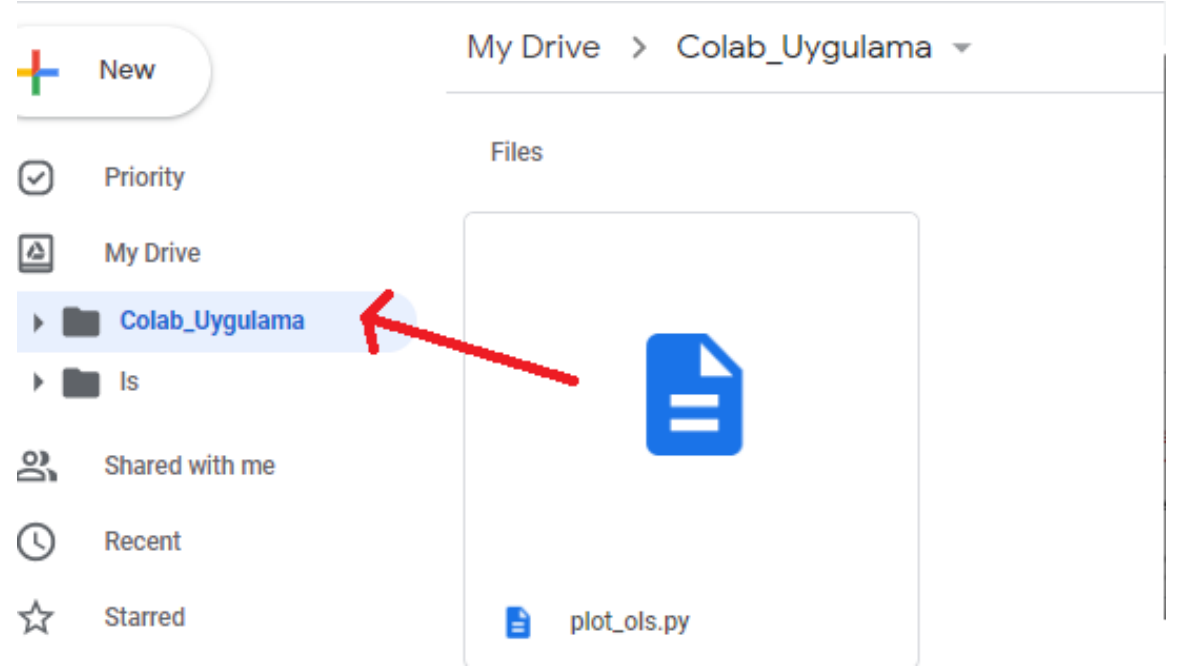
# 2. Geliştirme Ortamının Hazırlanması

## 2.2 Google Drive'da Çalışma Klasörü Oluşturma:



## 2.2 Çalışma Dosyalarımızın Drive Ortamına Yüklenmesi

[https://drive.google.com/file/d/15w4C2zP9PGTDjCFA5Cr9\\_NjY3ahjY4\\_-/\\_view?usp=sharing](https://drive.google.com/file/d/15w4C2zP9PGTDjCFA5Cr9_NjY3ahjY4_-/_view?usp=sharing)

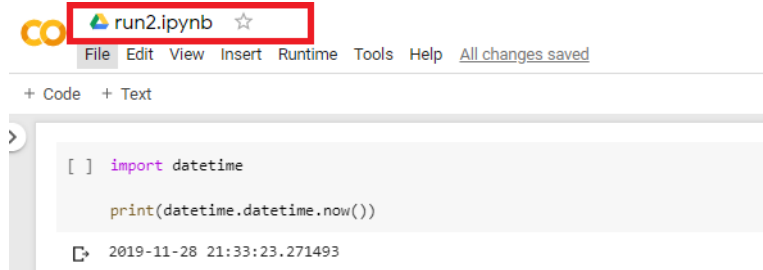


# 2. Geliştirme Ortamının Hazırlanması

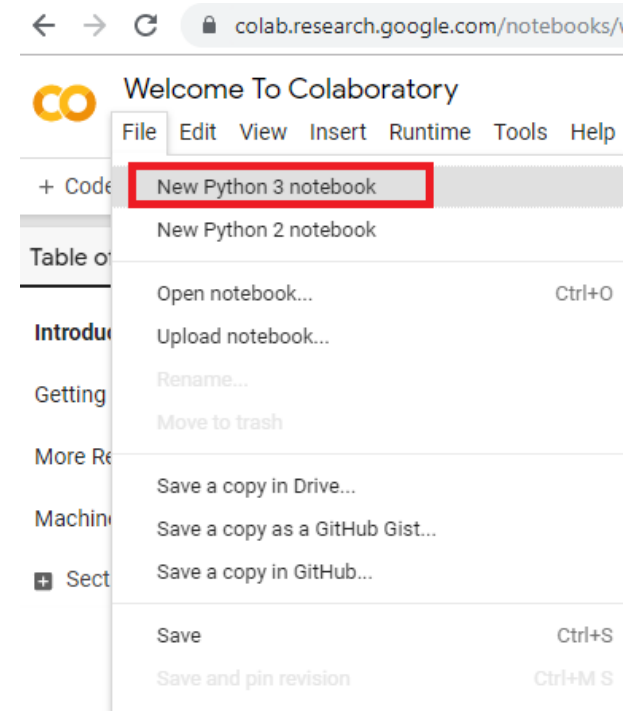
## 2. Yeni Notebook Ortamının Kurulması

### a. Colab Hesabına Giriş:

#Link: <http://colab.research.google.com/>



### b. Yeni Notebook Oluşturma



# 2. Geliştirme Ortamının Hazırlanması

## 2. Yeni Notebook Ortamının Kurulması

### c. Google Drive Çalışma Ortamı Colab Bağlantısı

The image illustrates the process of connecting Google Colab to Google Drive. It consists of three main parts:

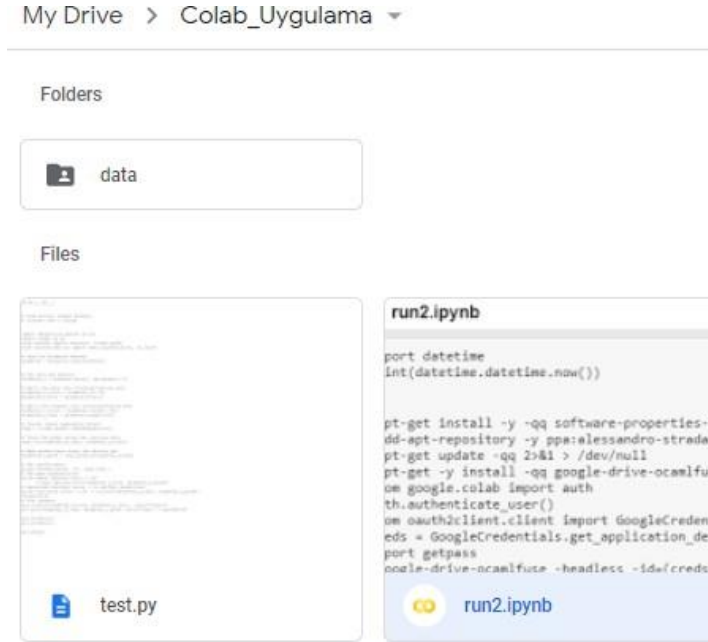
- Colab Interface:** The top left shows the Colab interface with the 'File' menu open. The 'Locate in Drive' option is highlighted with a red box.
- Context Menu:** The middle part shows a context menu for the file 'run2.ipynb'. The 'Move to' option is highlighted with a red box.
- My Drive Window:** The bottom right shows a 'My Drive' window. The 'Colab\_Uygulama' folder is highlighted with a red box.

File>"Located in Drive"

## 2. Geliştirme Ortamının Hazırlanması

### 23 Yeni Notebook Ortamının Kurulması

#### SON GÖRÜNÜM



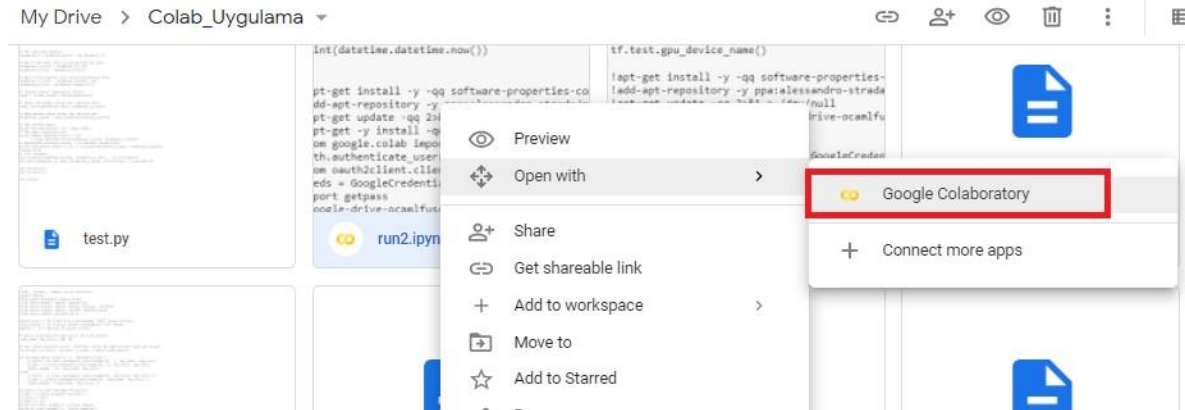
Google drive üzerinde projemizin olduğu dizine gittiğimizde; oluşacak son görünüm yukarıdaki gibi oluşması beklenmektedir.



# 2. Geliştirme Ortamının Hazırlanması

## 2.4 GPU Ayarlaması

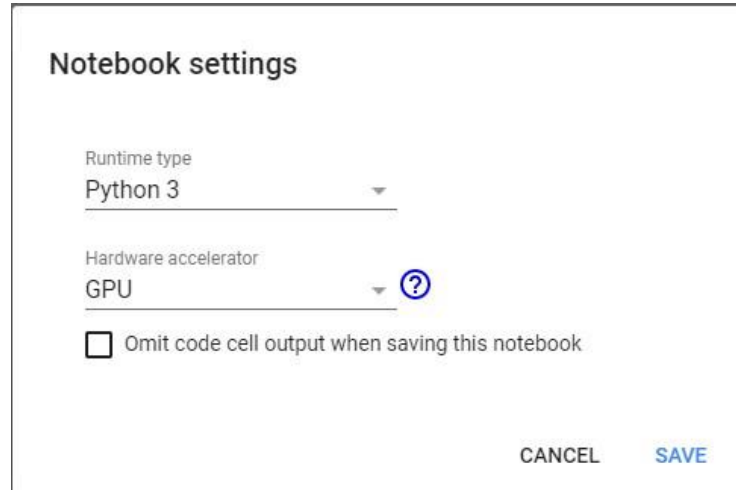
İlk önce **run2.ipynb**>**Open with**>**Google Colaboratory** diyerek notebookumuzu CoLab üzerinde açabiliriz.



## 2. Geliştirme Ortamının Hazırlanması

### 2.4 GPU Ayarlaması

Ardından **Edit>Notebook Settings** özelliği ile ücretsiz bize sunulan GPU hizmetinden yararlanabiliriz.



Notebook settings

Runtime type  
Python 3

Hardware accelerator  
GPU

☐ Omit code cell output when saving this notebook

CANCEL SAVE



# Dökümanı kaydettiğimizde, kod hücrelerindeki çıktıların atılıp atılmayacağını bize söylemektedir.

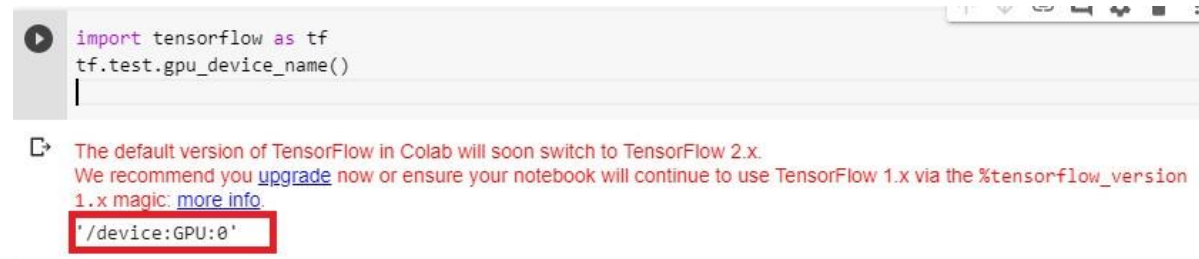
☐ Omit code cell output when saving this notebook

## 2. Geliştirme Ortamının Hazırlanması

### 24 GPU Ayarlaması

#### GPU Kontrol Etme

```
import tensorflow as tf  
tf.test.gpu_device_name()
```



The screenshot shows a Jupyter Notebook interface. The top part is a code cell with the following Python code:

```
import tensorflow as tf  
tf.test.gpu_device_name()  
|
```

Below the code cell is an output area. It contains a message from TensorFlow:

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.  
We recommend you [upgrade](#) now or ensure your notebook will continue to use TensorFlow 1.x via the %tensorflow\_version 1.x magic: [more info](#)

Below the message, the output of the code is displayed: `"/device:GPU:0"`. This output is highlighted with a red rectangular box.

## 2. Geliştirme Ortamının Hazırlanması

### 25 Run the sample:

```
▶ from datetime import datetime

a=datetime.now()

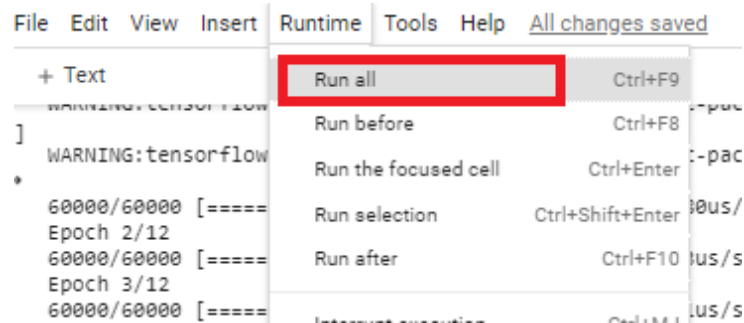
print(a)
```

📄 2019-11-28 18:15:37.336756

**Not-1:** Yeni Kod Bloğu ya da yorum satırı eklemek için sol üstteki bölümleri kullanabiliriz:

+ Code + Text

**Not-2:** Notebookta sadece blokları çalıştırmak için “**Ctrl+Enter**” diyebiliriz ya da **Runtime>Run all** ile tüm blokları adım adım çalıştırabiliriz.



---

## 3. Projeye Hazır Hale Getirme

# 3. Projeye Hazır Hale Getirme

## Part I

**# Install a Drive FUSE wrapper.**

**# <https://github.com/astrada/google-drive-ocamlfuse>**

```
!apt-get install -y -qq software-properties-common python-software-properties module-init-tools
```

```
!add-apt-repository -y ppa:alessandro-strada/ppa 2>&1 > /dev/null
```

```
!apt-get update -qq 2>&1 > /dev/null
```

```
!apt-get -y install -qq google-drive-ocamlfuse fuse
```

## Part II

**# Generate auth tokens for**

**Colab** from google.colab

```
import auth
```

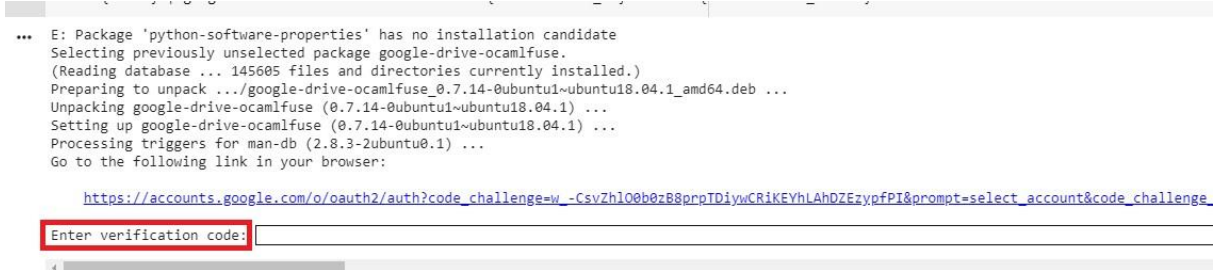
```
auth.authenticate_user()
```

# 3. Projeye Hazır Hale Getirme

## Part III

### # Generate creds for the Drive FUSE library.

```
from oauth2client.client import GoogleCredentials
creds =
GoogleCredentials.get_application_default() import
getpass
!google-drive-ocamlfuse -headless -id={creds.client_id} -secret={creds.client_secret} < /dev/null 2>&1 | grep
URL vcode = getpass.getpass()
!echo {vcode} | google-drive-ocamlfuse -headless -id={creds.client_id} -secret={creds.client_secret}
```



## Part IV


### # Create a directory and mount Google Drive using that directory.

```
!mkdir -p drive
!google-drive-ocamlfuse drive
```



Oturum aç

Lütfen bu kodu kopyalayın, uygulamanıza geçin ve kodu oraya yapıştırın:

4/twG17dK\_5FfIgrM4iHS4JQn4vEe\_tk1lU7F8JcTdeHUzI9wF   
SHUGQrk

# 3. Projeye Hazır Hale Getirme

## Part V

```
print ('Files in Drive:') !ls drive/
```

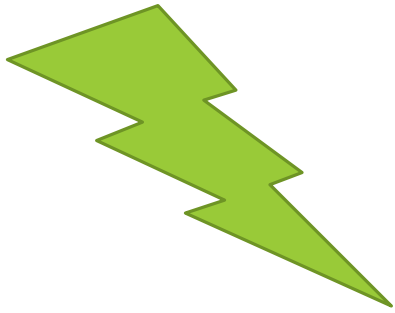
```
[34] print ('Files in Drive:')  
      !ls drive/
```

```
↳ Files in Drive:  
   'Colab Notebooks'  Colab_Uygulama  foo.txt  ls
```



# 3. Projeye Hazır Hale Getirme

File pathimize sys kütüphanesinden yararlanarak projemizde göstermemiz önemlidir.

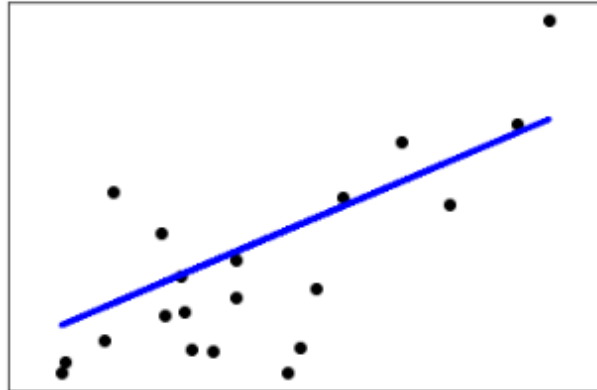


```
import sys  
sys.path.insert(0, 'drive/Colab_Uygulama')
```

# 3. Projeye Hazır Hale Getirme

Colab proje klasörümüz içindeki “plot\_linear.py” kodlarını çalıştırdığımızda ise aşağıdaki gibi çizim elde ettik:

```
↳ Automatically created module for IPython interactive environment  
Coefficients:  
[938.23786125]  
Mean squared error: 2548.07  
Variance score: 0.47
```



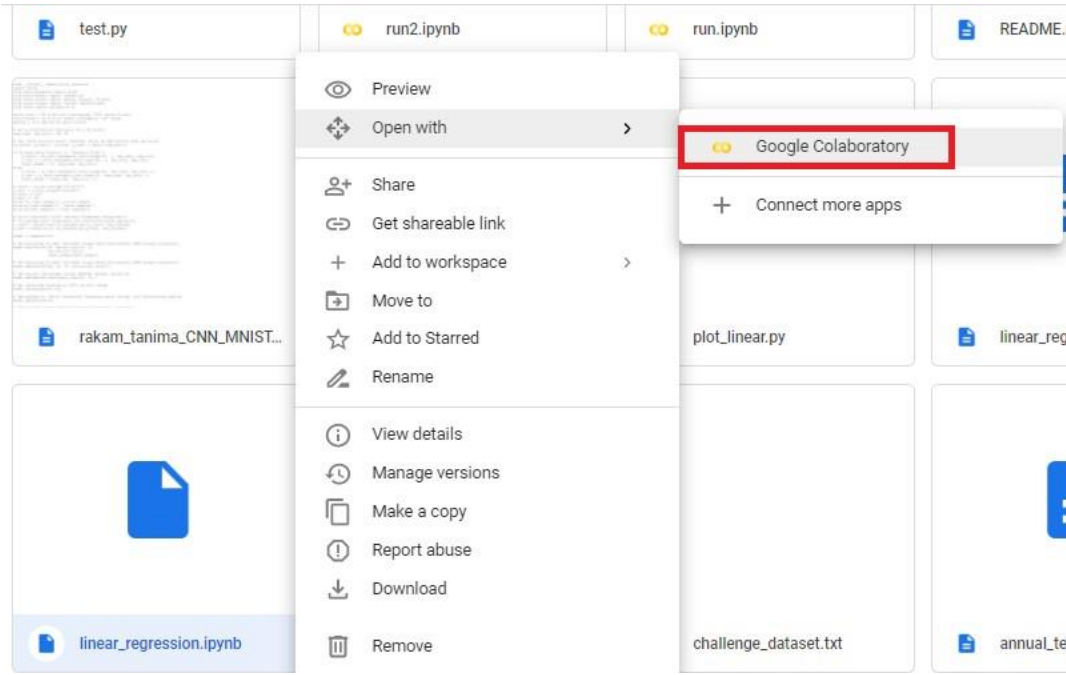
---

## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

### #Githublink:

[https://github.com/justmarkham/scikit-learn-videos/blob/master/06\\_linear\\_regression.ipynb](https://github.com/justmarkham/scikit-learn-videos/blob/master/06_linear_regression.ipynb)



**06\_linear\_regression.ipynb** isimli dosyayı “**linear\_regression.ipynb**” olarak değiştirip Colab Proje klasörümüzün içine attıktan sonra githubdan indirdiğimiz “data” klasöründe aynı şekilde driverda aynı dizinde yer alacak şekilde import edebiliriz.

Bu sefer notebook olarak çalıştırmak için “**linear\_regression.ipynb**” isimli dosyaya sağ tıklayıp Colab üzerinde açabiliriz.

GPU kullanmak istersek yine aynı şekilde **Edit>Notebook Settings**

[https://colab.research.google.com/drive/1AQ0wfbdOe5hkLOeqGA\\_P5BZwv-zoJ3gQ?authuser=1#scrollTo=GiwFIP5bVaeN](https://colab.research.google.com/drive/1AQ0wfbdOe5hkLOeqGA_P5BZwv-zoJ3gQ?authuser=1#scrollTo=GiwFIP5bVaeN)

## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

---

Sonuçlarımızı visualize etmek için;

**!pip install seaborn**

Şeklinde seaborn kütüphanesini yüklü değilse yüklemeliyiz.

## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

```
[ ] # Install a Drive FUSE wrapper.  
# https://github.com/astrada/google-drive-ocamlfuse  
  
!apt-get install -y -qq software-properties-common python-software-properties module-init-tools  
!add-apt-repository -y ppa:alessandro-strada/ppa 2>&1 > /dev/null  
!apt-get update -qq 2>&1 > /dev/null  
!apt-get -y install -qq google-drive-ocamlfuse fuse
```

❏ E: Package 'python-software-properties' has no installation candidate  
.....

```
[ ] # Generate auth tokens for Colab  
  
from google.colab import auth  
auth.authenticate_user()
```

```
[ ] # Generate creds for the Drive FUSE library.  
  
from oauth2client.client import GoogleCredentials  
creds = GoogleCredentials.get_application_default()  
import getpass  
!google-drive-ocamlfuse -headless -id={creds.client_id} -secret={creds.client_secret} < /dev/null 2>&1 | grep URL  
vcode = getpass.getpass()  
!echo {vcode} | google-drive-ocamlfuse -headless -id={creds.client_id} -secret={creds.client_secret}
```

❏ .....

## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

```
[ ] # Create a directory and mount Google Drive using that directory.  
!mkdir -p drive
```

```
[ ] !google-drive-ocamlfuse drive
```

```
[ ] !pip install seaborn
```

```
[ ] print ('Files in Drive:')  
!ls drive/
```

```
Files in Drive:  
'Colab Notebooks' Colab_Uygulama foo.txt ls
```

```
import sys  
sys.path.insert(0, 'drive/Colab_Uygulama')  
  
# conventional way to import pandas  
import pandas as pd
```

## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

```
[ ] # Create a directory and mount Google Drive using that directory.  
!mkdir -p drive
```

```
[ ] !google-drive-ocamlfuse drive
```

```
[ ] !pip install seaborn
```

```
[ ] print ('Files in Drive:')  
!ls drive/
```

```
Files in Drive:  
'Colab Notebooks' Colab_Uygulama foo.txt ls
```

File pathimize `sys` kütüphanesinden yararlanarak projemizde göstermemiz önemlidir.

```
import sys  
sys.path.insert(0, 'drive/Colab_Uygulama')  
  
# conventional way to import pandas  
import pandas as pd
```



## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

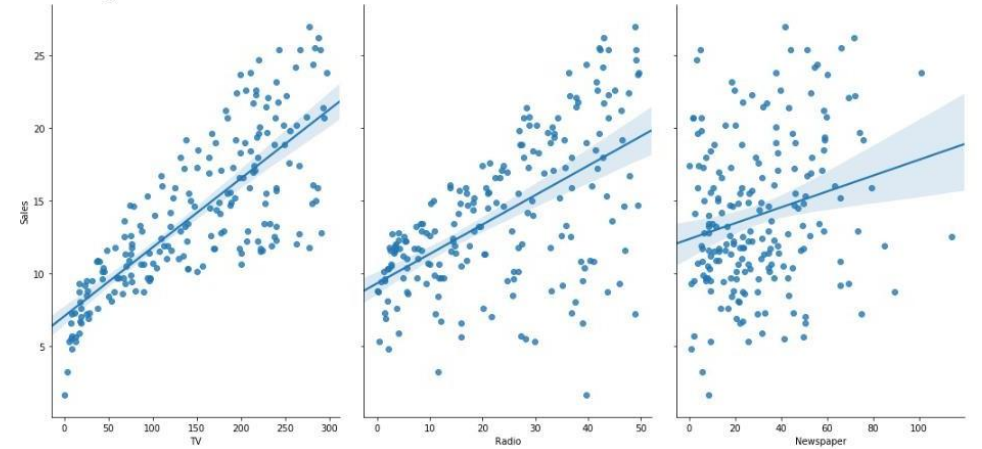
Gör üd ü ğ ü gibi “data/Advertising.csv” dataset dosyamız okunabildi.

```
# read CSV file from the 'data' subdirectory using a relative path
data = pd.read_csv('drive/Colab_Uygulama/data/Advertising.csv', index_col=0)

# display the first 5 rows
data.head()
```

	TV	Radio	Newspaper	Sales
1	230.1	37.8	69.2	22.1
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9

Seaborn kütüphanesi ile TV, Radio, Newspaper  
Öz niteliklerini visualize ettiğimizde aşağıdaki  
gibi sonuç olacaktır.



## 4. SKLEARN kütüphanesi ile Doğrusal Regresyon Analizi

Sklearn kütüphanesinin doğrusal regresyon analizi için aşağıdaki kütüphane import adımını unutmamamız gerekmektedir:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)
```

Modelimizin coefficients değerleri için aşağıdakileri girebiliriz.

▼ Interpreting model coefficients

```
[48] # print the intercept and coefficients
print(linreg.intercept_)
print(linreg.coef_)
```

```
2.87696662231793
[0.04656457 0.17915812 0.00345046]
```

```
[49] # pair the feature names with the coefficients
list(zip(feature_cols, linreg.coef_))
```

```
[('TV', 0.046564567874150295),
 ('Radio', 0.17915812245088839),
 ('Newspaper', 0.0034504647111804343)]
```



Prediction ımızın RMSE yani Root Mean Squared Error değeri ise aşağıdaki sonucu verecektir. En son.

```
# compute the RMSE of our predictions
print(np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
1.3879034699382888
```

RMSE'in güzelliği ise, MSE'den daha popülerdir ve "y" unitlerde yorumlanabilir olma özelliğidir.

## 5. KERAS Convolutional Neural Network (CNN) ile MNIST Rakım Tanıma Problemini Çözme

### 5.1 !python Komutu ile Çalıştırma

**run2.ipynb** daha önce oluşturduğumuz notebook üzerinde çalıştırabiliriz.

“**rakam\_tanisma\_CNN\_MNIST.py**” dosyasını driverımıza ekledikten sonra Colab üzerinde çalıştırmak için şu komutu girebiliriz.

```
!python3 drive/Colab_Uygulama/rakam_tanima_CNN_MNIST.py
```

Ardından outputumuz aşağıdaki gibi olması beklenmektedir:

12 adımda gerçekleşen işlemler listelendi. Test kaybı ve doğruluğumuz listelenmiştir.

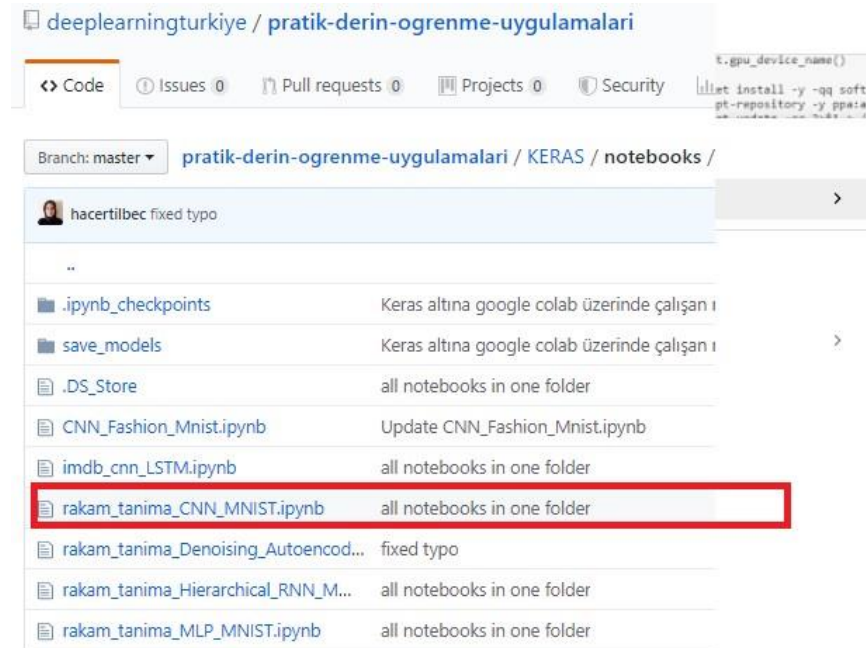
```
2019-11-28 22:55:44.564331: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcudnn.so.7
60000/60000 [=====] - 6s 108us/step - loss: 0.2631 - acc: 0.9185 - val_loss: 0.0582 - val_acc: 0.9820
Epoch 2/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0908 - acc: 0.9728 - val_loss: 0.0416 - val_acc: 0.9857
Epoch 3/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0688 - acc: 0.9796 - val_loss: 0.0362 - val_acc: 0.9870
Epoch 4/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0557 - acc: 0.9834 - val_loss: 0.0330 - val_acc: 0.9882
Epoch 5/12
60000/60000 [=====] - 4s 68us/step - loss: 0.0471 - acc: 0.9858 - val_loss: 0.0292 - val_acc: 0.9901
Epoch 6/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0415 - acc: 0.9872 - val_loss: 0.0260 - val_acc: 0.9906
Epoch 7/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0384 - acc: 0.9882 - val_loss: 0.0264 - val_acc: 0.9901
Epoch 8/12
60000/60000 [=====] - 4s 67us/step - loss: 0.0337 - acc: 0.9894 - val_loss: 0.0287 - val_acc: 0.9900
Epoch 9/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0324 - acc: 0.9904 - val_loss: 0.0261 - val_acc: 0.9907
Epoch 10/12
60000/60000 [=====] - 4s 67us/step - loss: 0.0291 - acc: 0.9909 - val_loss: 0.0239 - val_acc: 0.9920
Epoch 11/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0282 - acc: 0.9912 - val_loss: 0.0241 - val_acc: 0.9915
Epoch 12/12
60000/60000 [=====] - 4s 66us/step - loss: 0.0263 - acc: 0.9916 - val_loss: 0.0249 - val_acc: 0.9917
Test loss: 0.024907596350954554
Test accuracy: 0.9917
```

## 5. KERAS Convolutional Neural Network (CNN) ile MNIST Rakam Tanıma Problemini Çözme

### 5.2 Jupyter Notebook ile Çalıştırma

Notebook olarak çalıştırmak istiyorsak yine 3. Konuda olduğu gibi aynı adımları izleyebiliriz.

<https://github.com/deeplearningturkiye/pratik-derin-ogrenme-uygulamalari/tree/master/KERAS/notebooks>

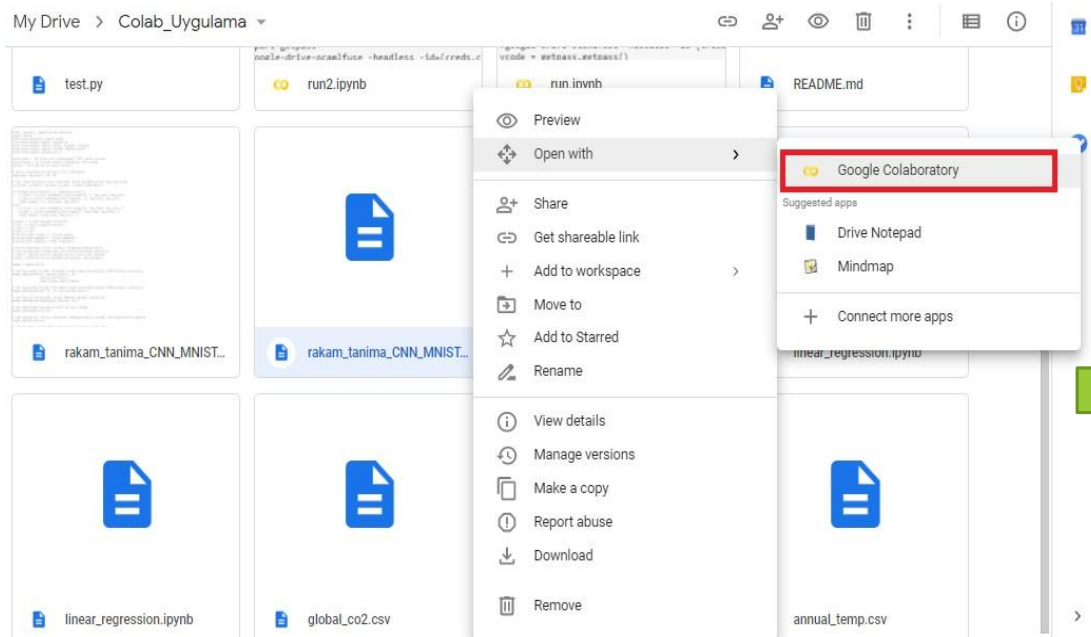


## 5. KERAS Convolutional Neural Network (CNN) ile MNIST Rakam Tanıma Problemini Çözme

### 5.2 Jupyter Notebook ile

#### Hazır CoLab Ortamı

<https://colab.research.google.com/drive/1IYydl3gSz1sKdxrKUclBrFm6TC8zbM6P?authuser=1#scrollTo=KU0NBitONcNg> Githubtaki CNN ile MNIST rakam tanıma notebook'u indirilip 3. bölümdeki gibi colab projemizde açalım:



## 5. KERAS Convolutional Neural Network (CNN) ile MNIST Rakam Tanıma Problemini Çözme

### 5.2 Jupyter Notebook ile

```
#Install a Drive FUSE wrapper.  
# https://github.com/astrada/google-drive-ocamlfuse  
  
!apt-get install -y -qq software-properties-common python-software-properties module-init-tools  
!add-apt-repository -y ppa:alessandro-strada/ppa 2>&1 > /dev/null  
!apt-get update -qq 2>&1 > /dev/null  
!apt-get -y install -qq google-drive-ocamlfuse fuse
```

```
E: Package 'python-software-properties' has no installation candidate  
Selecting previously unselected package google-drive-ocamlfuse.  
(Reading database ... 145605 files and directories currently installed.)  
Preparing to unpack .../google-drive-ocamlfuse_0.7.14-0ubuntu1~ubuntu18.04.1_amd64.deb ...  
Unpacking google-drive-ocamlfuse (0.7.14-0ubuntu1~ubuntu18.04.1) ...  
Setting up google-drive-ocamlfuse (0.7.14-0ubuntu1~ubuntu18.04.1) ...  
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

```
[ ] # Generate auth tokens for Colab
```

```
from google.colab import auth  
auth.authenticate_user()
```

```
[ ] # Generate creds for the Drive FUSE library.
```

```
from oauth2client.client import GoogleCredentials  
creds = GoogleCredentials.get_application_default()  
import getpass  
!google-drive-ocamlfuse -headless -id={creds.client_id} -secret={creds.client_secret} < /dev/null 2>&1 | grep URL  
vcode = getpass.getpass()  
!echo {vcode} | google-drive-ocamlfuse -headless -id={creds.client_id} -secret={creds.client_secret}
```

**Bu adımları daha önce yapmıştık ancak sıfırdan yaptığımızı varsayarsak gereksinimleri daha önce yüklememiş gibi ilerleyelim:**

## 5. KERAS Convolutional Neural Network (CNN) ile MNIST Rakam Tanıma Problemini Çözme

### 5.2 Jupyter Notebook ile

```
[ ] # Create a directory and mount Google Drive using that directory.
    !mkdir -p drive

[ ] !google-drive-ocamlfuse drive

[ ] # To visualize the dataset
    !pip install seaborn

[ ] print ('Files in Drive:')
    !ls drive/

[ ] import sys
    sys.path.insert(0, 'drive/Colab_Uygulama')
```

Bu adımları daha önce yapmıştık ancak sıfırdan yaptığımızı varsayarsak gereksinimleri daha önce yüklememiş gibi ilerleyelim:

## 5. KERAS Convolutional Neural Network (CNN) ile MNIST Rakam Tanıma Problemini Çözme

### 5.2 Jupyter Notebook ile

```
60000/60000 [=====] - 9s 152us/step - loss: 0.0380 - acc: 0.9883 - val_loss: 0.0300 - val_acc: 0.9909
Epoch 8/12
60000/60000 [=====] - 9s 152us/step - loss: 0.0342 - acc: 0.9900 - val_loss: 0.0322 - val_acc: 0.9900
Epoch 9/12
60000/60000 [=====] - 9s 152us/step - loss: 0.0305 - acc: 0.9905 - val_loss: 0.0276 - val_acc: 0.9912
Epoch 10/12
60000/60000 [=====] - 9s 152us/step - loss: 0.0283 - acc: 0.9914 - val_loss: 0.0254 - val_acc: 0.9919
Epoch 11/12
60000/60000 [=====] - 9s 153us/step - loss: 0.0259 - acc: 0.9922 - val_loss: 0.0290 - val_acc: 0.9907
Epoch 12/12
60000/60000 [=====] - 9s 153us/step - loss: 0.0251 - acc: 0.9922 - val_loss: 0.0277 - val_acc: 0.9918
<keras.callbacks.History at 0x7fb1b162be10>
```

```
# test işlemini gerçekleştirelim ve sonuçları ekrana yazdıralım
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 0.0276763818390541
Test accuracy: 0.9918
```



# KAYNAKÇA

---

Proje Link:

[https://drive.google.com/drive/folders/1hKfz\\_wl0ViCaHaXppu81HBVMdA5hSPNF?usp=sharing](https://drive.google.com/drive/folders/1hKfz_wl0ViCaHaXppu81HBVMdA5hSPNF?usp=sharing)

<https://github.com/justmarkham/scikit-learn-videos>

<https://medium.com/deep-learning-turkiye/google-colab-ile-%C3%BCcretsiz-gpu-kullan%C4%B1m%C4%B1-30fdb7dd822e>

<https://stackoverflow.com/questions/48967757/cant-read-a-file-in-google-colaboratory>

# TEŞEKKÜRLER 😊

---

The logo for Google Colab, featuring the word "colab" in a bold, lowercase, sans-serif font. The letters "co" are yellow, and "lab" is orange. The logo is centered within a black rectangular background.

colab