Isim Soyisim: Shiraz Amadu Bello

Öğrenci Numarası: 200023474

Python'da Basit Doğrusal Regresyon (Sıfırdan)

Basit doğrusal gerileme, ortaokul veya liseden aşina olabileceğiniz bir kavramdır. Hiç bir eğim ve kesişme veya **y = mx + b** duyduysanız, basit doğrusal gerilemeyi zaten öğrendiniz!

Basit Doğrusal Regresyon Nedir?

Basit doğrusal regresyon, iki değişken arasında bir ilişki bulmak ve tahminlerde bulunmak içinve istatistiksel bir yöntemdir. Kullanılan iki değişken genellikle y ve xolarak gösterilir. Bağımsız değişken veya bağımlı değişkeni tahmin etmek için kullanılan değişken x olarak gösterilir. Bağımlı değişken veya **sonuç/çıktı** yolarak gösterilir. Basit bir doğrusal regresyon **modelin en uygun** regresyon çizgisi bir üretecektir. Bir veri saçılım grafiği aracılığıyla en uygun çizgiyi çizmeyi duymuş olabilirsiniz. Örneğin, yılların deneyiminin maaşları nasıl etkilediğini gösteren bir dağılım arsamız olduğunu varsayalım. Trendi tahmin etmek için bir çizgi çizdiğinizi hayal edin.

Kullanacağımız basit doğrusal regresyon denklemi aşağıda yazılmıştır. Sabit, y kesişim noktasıdır **(β0)** veya regresyon çizgisinin y ekseninde başlayacağı yerdir. Beta katsayısı **(β1)** eğimdir ve bağımsız değişken ile bağımlı değişken arasındaki ilişkiyi açıklar. Katsayı pozitif veya negatif olabilir ve bağımsız değişkendeki her 1 birimlik değişim için bağımlı değişkendeki değişim derecesidir.

Yi=B0+B1.x

Örneğin, diyelim ki **y = 2 + 0.5x** regresyon denklemimiz var. Bağımsız değişkendeki her 1 birimlik artış için (x), bağımlı değişkende (y) 0,50 artış olacaktır.

Python Kullanarak Basit Doğrusal Regresyon

Bu örnekte, Kaggle'dan maaş verilerini kullanıyor olacağız. Veriler iki sütun, yılların deneyimi ve ilgili maaşdan oluşur. Verilere buradanulaşabilirsiniz. İlk olarak, bu analiz için ihtiyaç duyacağımız Python paketlerini içe aktaracağız. İhtiyacımız olan tek şey

a)NumPy= matematik hesaplamalarına yardımcı olmak için
b)Pandas= verileri depolamak ve işlemek için
c)Matplotlib=verileri çizmek için

In [18]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [19]:

```python
# Veri Okuma, ve ayırma
data = pd.read_csv('./DataSet/Salary_Data.csv')
x = data['YearsExperience']
y = data['Salary']
```

In [20]:

```python
data.head()# Veriyi incelemek. Ilk 5 satırını göstermek
```

Out[20]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

In [21]:

```python
data.tail() #Verinin Son 5 satırını göstermek
```

Out[21]:

	YearsExperience	Salary
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

In [22]:

```python
data.describe()#Verinin İstatistik değerini yansıtmak
```

Out[22]:

	YearsExperience	Salary
count	30.000000	30.000000
mean	5.313333	76003.000000
std	2.837888	27414.429785
min	1.100000	37731.000000
25%	3.200000	56720.750000
50%	4.700000	65237.000000
75%	7.700000	100544.750000
max	10.500000	122391.000000

Regresyon Satırını Hesaplama

Tüm günümüzü doğrusal regresyon çizgisinin eğimini ve kesişimini tahmin ederek geçirebilirken, neyse ki bu hesaplamaları hızlı bir şekilde yapmak için kullanabileceğimiz formüller var. Verilerin eğim **β1'ini** tahmin etmek için aşağıdaki formülü kullanacağız:

In [23]:

```python
def linear_regression(x, y):
    N = len(x)
    x_mean = x.mean() #x'lerin Ortamasi
    y_mean = y.mean() #Y'nin Ortalamasi

    B1_num = ((x - x_mean) * (y - y_mean)).sum() #Pay'in Formulu
    B1_den = ((x - x_mean)**2).sum() #Payda'nin Formulu
    B1 = B1_num / B1_den # Verılen Eğimin Formulu

    B0 = y_mean - (B1*x_mean) # Kesişim Noktası

    reg_line = 'y = {} + {}β'.format(B0, round(B1, 3))

    return (B0, B1, reg_line)
```

Regresyon Çizgisinin Ne Kadar İyi Uyduğunu Hesaplama

Regresyon hattımızın verilere ne kadar iyi uyduğunu belirlemek için, genellikle yalnızca **R** olarak adlandırılan korelasyon katsayısını ve **R^2 (R kare)** olarak bilinen belirleme katsayısını hesaplamak istiyoruz.

Belirleme Katsayısı (R^2) — **0** ile **1** arasındaki değerlerle bağımsız değişken (x) ile açıklanan varyans yüzdesi. Kare bir değer olduğundan negatif olamaz. Örneğin, **R^2 = 0,81** ise, bu size x'in **y**'deki farkın **% 81**'ini açıkladığını söyler. **"Fitin iyiliği"** olarak da bilinir.

Korelasyon Katsayısı (R)— İki değişken arasındaki ilişki veya korelasyon derecesi (bu durumdax ve y). R, 1'e eşit değerlerle -1 ile 1 arasında değişebilir, yani mükemmel pozitif korelasyon ve -1'e eşit değerler mükemmel bir negatif korelasyon anlamına gelir.

Pearson'ın korelasyon katsayısının formülü aşağıdadır:

$$r = \frac{N\sum XY - (\sum X \sum Y)}{\sqrt{\left[N\sum x^2 - (\sum x)^2\right]\left[N\sum y^2 - (\sum y)^2\right]}}$$

Bu formülü Python koduna dönüştürmemiz gerekecek. Pearson'ın korelasyon katsayısını hesapladığımızda, belirleme katsayısını elde etmek için kare oluşturabiliriz. Gözlem sayısını (verilerdeki satırlar) **N** değişkeninde tekrar depolamamız gerekecek. Daha sonra, formülü iki bölüme ayıracağız: pay ve payda. Daha sonra korelasyon katsayısını geri verebiliriz.

Korelasyon Katsayinin Kodu

In [24]:

```python
def corr_coef(x, y):
    N = len(x)
    #Formulun Payi ve paydasi hesaplamak
    num = (N * (x*y).sum()) - (x.sum() * y.sum())
    den = np.sqrt((N * (x**2).sum() - x.sum()**2) * (N * (y**2).sum() - y.sum()**2))
    R = num / den #Korelasyon Katsayiyi Hesaplama
    return R
```

Bu işlevleri verilerimize uygulayarak, sonuçları yazdırabiliriz:

Linear Regresyonu, R ve R^2 Fonksiyonlarını uygulamak

In [25]:

```python
B0, B1, reg_line = linear_regression(x, y)
print('Regression Line: ', reg_line)
R = corr_coef(x, y)
print('Correlation Coef.: ', R)
print('"Goodness of Fit": ', R**2)
```

```
Regression Line:  y = 25792.20019866869 + 9449.962β
Correlation Coef.:  0.97824161848876
"Goodness of Fit":  0.9569566641435087
```

Regresyon Çizgisini Çizme

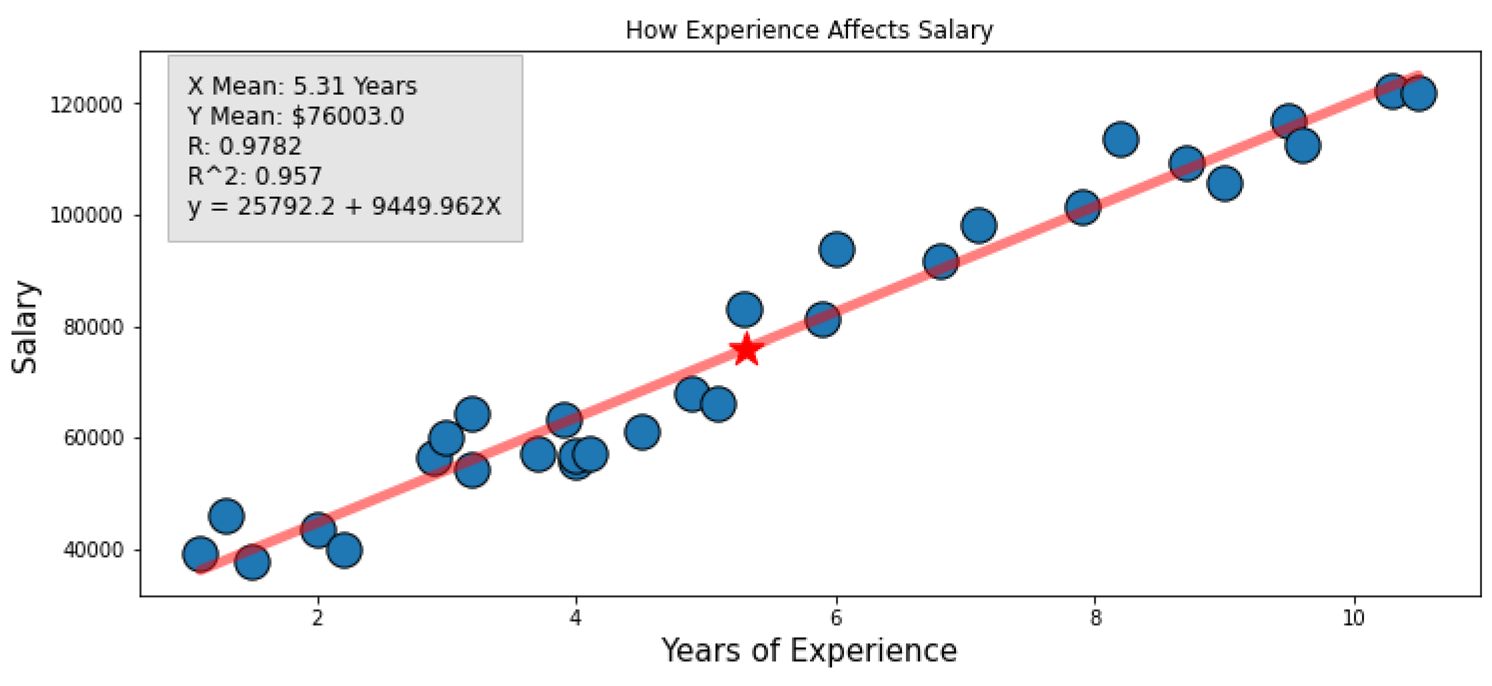
Bu bölüm tamamen isteğe bağlıdır ve sadece eğlence içindir. Matplotlib'i kullanarak, elde ettiğimiz regresyon çizgisini artık verilerimizle çizebiliriz.

In [26]:

```python
plt.figure(figsize=(12,5))
plt.scatter(x, y, s=300, linewidths=1, edgecolor='black')
text = '''X Mean: {} Years
Y Mean: ${}
R: {}
R^2: {}
y = {} + {}X'''.format(round(x.mean(), 2),
                       round(y.mean(), 2),
                       round(R, 4),
                       round(R**2, 4),
                       round(B0, 3),
                       round(B1, 3))
plt.text(x=1, y=100000, s=text, fontsize=12, bbox={'facecolor': 'grey', 'alpha': 0.2, 'pad': 10})
plt.title('How Experience Affects Salary')
plt.xlabel('Years of Experience', fontsize=15)
plt.ylabel('Salary', fontsize=15)
plt.plot(x, B0 + B1*x, c = 'r', linewidth=5, alpha=.5, solid_capstyle='round')
plt.scatter(x=x.mean(), y=y.mean(), marker='*', s=10**2.5, c='r') # average point
```

Out[26]:

```
<matplotlib.collections.PathCollection at 0x1e863f34f70>
```



Çıkan denkleme göre yeni tecrube yıllara bağlı olarak Maaşı Tahmin Etmek

Artık regresyon çizgisi hesaplamalarımızı, karşılaştığımız yeni verilerle tahminlerde bulunmak için kullanabiliriz

In [27]:

```python
#Maaşı Tahmin eden  Fonksiyonu
def predict(B0, B1, new_x):
    y = B0 + B1 * new_x
    return y
```

In [28]:

```python
def determine_your_salary(years_of_experience):
    print(
        '''{0} yillın tecrubelisi olduğun için {1} maaş alırsın"
        '''.format(years_of_experience,round(predict(B0,B1,years_of_experience),3))
          )
```

Bazı Maaşları Hesaplanması

In [29]:

```python
#20 yıılık tecrubesinin maaşı
determine_your_salary(20)
```

```
20 yillın tecrubelisi olduğun için 214791.447 maaş alırsın"
```

In [30]:

```python
#15 yıılık tecrubesinin maaşı
determine_your_salary(15)
```

```
15 yillın tecrubelisi olduğun için 167541.635 maaş alırsın"
```

In [31]:

```python
#1 yıılık tecrubesinin maaşı
determine_your_salary(1)
```

```
1 yillın tecrubelisi olduğun için 35242.163 maaş alırsın"
```

In [32]:

```python
#0 yıılık tecrubesinin maaşı
determine_your_salary(0)
```

```
0 yillın tecrubelisi olduğun için 25792.2 maaş alırsın"
```