Import biblioteki **TensorFlow** (https://www.tensorflow.org/) z której będziemy korzystali w **uczeniu maszynowym**:

```
import tensorflow as tf
```

Gradient

Możliwe jest wyliczenie gradientu dowolnego wyrażenia różniczkowalnego. Wykorzystujemy do tego metodę **tf.GradientTape()**

Funkcja **jednej zmiennej**:

```
x = tf.Variable(4.0)

with tf.GradientTape() as tape:
    f = x**3                      #definicja funkcji f(x)=x^3
    df_dx = tape.gradient(f, x)  #gradient 'f' ze względu na zmienną 'x'

df_dx.numpy()
```

```
48.0
```

Funkcja **dwóch zmiennych**:

```
x = tf.Variable(4.0)
y = tf.Variable(3.0)

with tf.GradientTape() as tape:
    f = x**3+y**2                          #definicja funkcji f(x,y)=x^3+y^2
    df_dx,df_dy = tape.gradient(f,(x,y))  #gradient 'f' ze względu na zmienną 'x' i ze wzg

print(df_dx)
print(df_dy)
```

```
tf.Tensor(48.0, shape=(), dtype=float32)
tf.Tensor(6.0, shape=(), dtype=float32)
```

Przykład z **prezentacji**:

```
x = tf.Variable(3.0)
y = tf.Variable(2.0)

with tf.GradientTape() as tape:
    f = (x**2)*y                           #definicja funkcji f(x,y)=x^2*y
    df_dx,df_dy = tape.gradient(f,(x,y))  #gradient 'f' ze względu na zmienną 'x' i ze wzg
```

```
print(df_dx)
print(df_dy)
```

```
     tf.Tensor(12.0, shape=(), dtype=float32)
     tf.Tensor(9.0, shape=(), dtype=float32)
```

Trochę skomplikujemy:

```
x = tf.Variable([3.0,2.0])

with tf.GradientTape() as tape:
    f = (x**3)                   #definicja funkcji f(x)=x^3
    df_dx = tape.gradient(f,x)  #gradient 'f' ze względu na zmienną 'x'

print(df_dx)
```

```
     tf.Tensor([27. 12.], shape=(2,), dtype=float32)
```

I jeszcze trochę skomplikujemy:

```
x = tf.Variable([3.0,2.0])
y = tf.Variable([1.0,0.0])

with tf.GradientTape() as tape:
    f = (x**3)+y**2                        #definicja funkcji f(x)=x^3+y^2
    df_dx,df_dy = tape.gradient(f,(x,y))  #gradient 'f' ze względu na zmienną 'x'

print(df_dx)
print(df_dy)
```

```
     tf.Tensor([27. 12.], shape=(2,), dtype=float32)
     tf.Tensor([2. 0.], shape=(2,), dtype=float32)
```

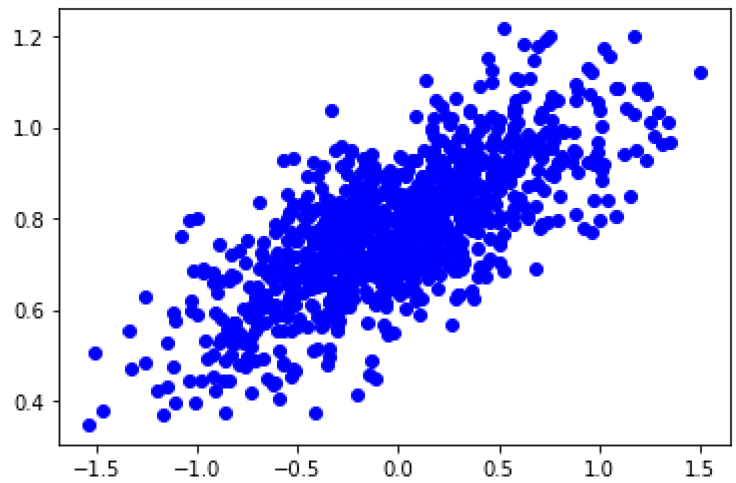
Zmienne mogą być zastąpione przez tensory, wówczas konieczne jest **rejestrowanie wprost** operacji zastosowanych do tych sensorów. Służy do tego metoda **watch()**. W przypkadku zmiennych operacje są rejestrowane automatycznie.

```
x = tf.random.normal([2])
y = tf.random.normal([2])

print(x)
print(y)
print("")

with tf.GradientTape() as tape:
    tape.watch(x)
    tape.watch(y)
    f = (x**3)+y**2                        #definicja funkcji f(x)=x^3+y^2
    df_dx,df_dy = tape.gradient(f,(x,y))  #gradient 'f' ze względu na zmienną 'x'

print(df_dx)
```

```
print(df_dx)
print(df_dy)
```

```
tf.Tensor([ 0.5899918 -1.3989412], shape=(2,), dtype=float32)
tf.Tensor([-1.1552613   0.48971453], shape=(2,), dtype=float32)

tf.Tensor([1.044271 5.871109], shape=(2,), dtype=float32)
tf.Tensor([-2.3105226   0.97942907], shape=(2,), dtype=float32)
```

```
import matplotlib.pyplot as plt
import numpy as np


number_of_points = 1000
x_point = []
y_point = []


a = 0.22
b = 0.78


for i in range(number_of_points):
    x = np.random.normal(0.0,0.5)
    y = (a*x+b)+np.random.normal(0.0,0.1)
    x_point.append(x)
    y_point.append(y)


plt.scatter(x_point,y_point,c='b')
plt.show()
```




```
real_x = np.array(x_point)
real_y = np.array(y_point)
```

Definicja błędu:

```
x = tf.constant([1.0, 2.0, 3.0, 4.0])
tf.reduce_mean(x)
```

```
<tf.Tensor: shape=(), dtype=float32, numpy=2.5>
```

```
def loss_fn(real_y, pred_y):
    return tf.reduce_mean((real_y - pred_y)**2)
```

```
import random
a = tf.Variable(random.random())
b = tf.Variable(random.random())
```

```
Loss = []
epochs = 50

for _ in range(epochs):
  with tf.GradientTape() as tape:
    pred_y = a * real_x + b
    loss = loss_fn(real_y, pred_y)
    Loss.append(loss.numpy())

  dloss_da, dloss_db = tape.gradient(loss,(a, b))

  a.assign_sub(0.1*dloss_da)
  b.assign_sub(0.1*dloss_db)
```

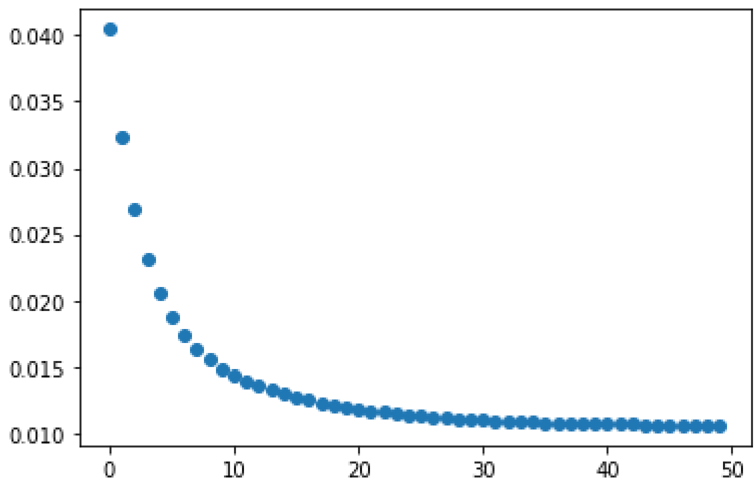
```
np.max(Loss),np.min(Loss)
```

```
(0.040465012, 0.010614281)
```

```
print(a.numpy())
print(b.numpy())
```

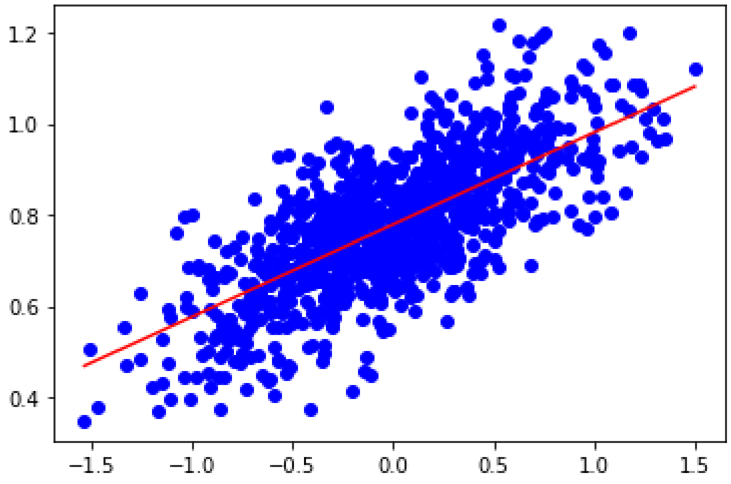
```
0.20191799
0.7786026
```

```
plt.scatter(np.arange(epochs),Loss)
plt.show()
```




```
max = np.max(x_point)
```

```
min = np.min(x_point)


X = np.linspace(min, max, num=10)
plt.plot(X,a.numpy()*X+b.numpy(),c='r')
plt.scatter(x_point,y_point,c="b")
plt.show()
```



zad.1 a)

```
x = tf.Variable(4.0)

with tf.GradientTape() as tape:
    f = ((2*x) ** 2) + 3*x + 4
    df_dx = tape.gradient(f, x)

df_dx.numpy()
```

```
35.0
```

b)

```
x = tf.Variable(4.0)
y = tf.Variable(5.0)

with tf.GradientTape() as tape:
    f = ((2*x) ** 3) + ((3*y) ** 2) + 4
    df_dx, df_dy = tape.gradient(f, (x,y))

print(df_dx)
print(df_dy)
```

```
tf.Tensor(384.0, shape=(), dtype=float32)
tf.Tensor(90.0, shape=(), dtype=float32)
```

c)

```
x = tf.constant([1.0, 3.0])
y = tf.constant([2.0, 1.0])

print(x)
print(y)
print("")

with tf.GradientTape() as tape:
    tape.watch(x)
    tape.watch(y)
    f = ((2*x) ** 3) + ((3*y) ** 2) + 4
    df_dx,df_dy = tape.gradient(f,(x,y))

print(df_dx)
print(df_dy)
```

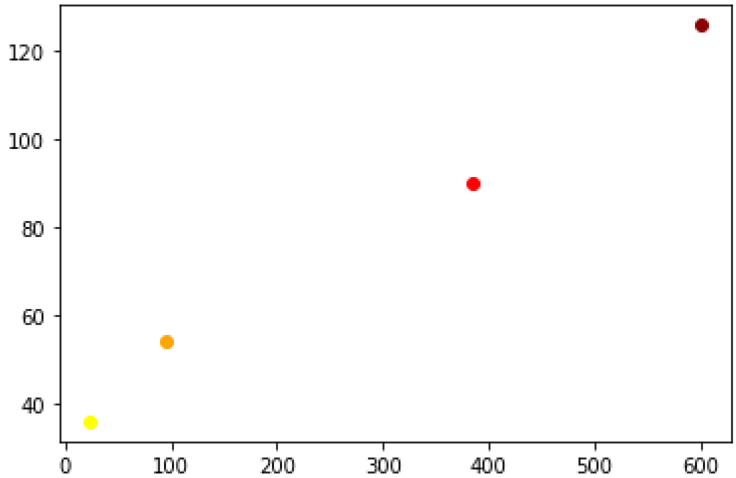
```
tf.Tensor([1. 3.], shape=(2,), dtype=float32)
tf.Tensor([2. 1.], shape=(2,), dtype=float32)

tf.Tensor([ 24. 216.], shape=(2,), dtype=float32)
tf.Tensor([36. 18.], shape=(2,), dtype=float32)
```

zad.2

```
import matplotlib.pyplot as plt

# 1) (4,5)
x = tf.Variable(4.0)
y = tf.Variable(5.0)

with tf.GradientTape() as tape:
    f = (2*x) ** 3+(3*y) ** 2+4
    df_dx,df_dy = tape.gradient(f, (x,y))

print(df_dx)
print(df_dy)
plt.scatter(df_dx, df_dy, c = "red")
print("-------------------------------")

# 2) (2,3) -> decrease
x = tf.Variable(2.0)
y = tf.Variable(3.0)

with tf.GradientTape() as tape:
    f = (2*x) ** 3+(3*y) ** 2+4
    df_dx,df_dy = tape.gradient(f, (x,y))

print(df_dx)
print(df_dy)
plt.scatter(df_dx, df_dy, c = "orange")
print("-------------------------------")

# 3) (1,2) -> decrease
x = tf.Variable(1.0)
```

```
y = tf.Variable(2.0)

with tf.GradientTape() as tape:
    f = (2*x) ** 3+(3*y) ** 2+4
    df_dx,df_dy = tape.gradient(f, (x,y))

print(df_dx)
print(df_dy)
plt.scatter(df_dx, df_dy, c = "yellow")
print("-------------------------------")
#plt.show()

# 4) (5,7) -> increase
x = tf.Variable(5.0)
y = tf.Variable(7.0)

with tf.GradientTape() as tape:
    f = (2*x) ** 3+(3*y) ** 2+4
    df_dx,df_dy = tape.gradient(f, (x,y))

print(df_dx)
print(df_dy)
plt.scatter(df_dx, df_dy, c = "#8b0000") # dark red color

plt.show()
```

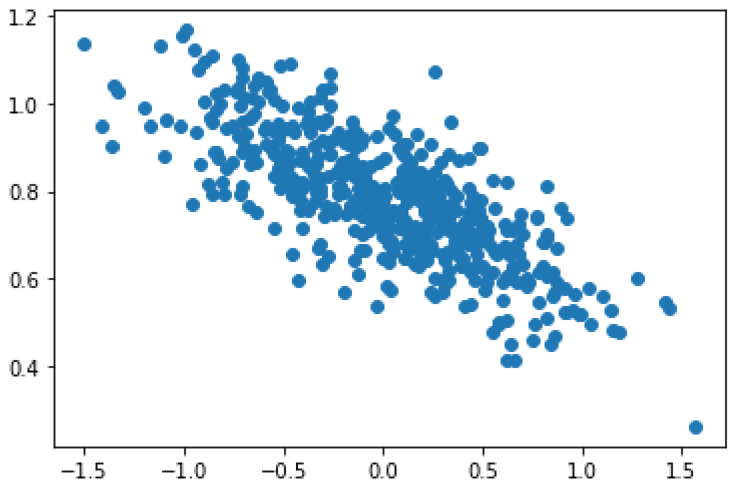
```
tf.Tensor(384.0, shape=(), dtype=float32)
tf.Tensor(90.0, shape=(), dtype=float32)
-------------------------------
tf.Tensor(96.0, shape=(), dtype=float32)
tf.Tensor(54.0, shape=(), dtype=float32)
-------------------------------
tf.Tensor(24.0, shape=(), dtype=float32)
tf.Tensor(36.0, shape=(), dtype=float32)
-------------------------------
tf.Tensor(600.0, shape=(), dtype=float32)
tf.Tensor(126.0, shape=(), dtype=float32)
```



zad.3 tr

```
import matplotlib.pyplot as plt
import numpy as np
```

```
number_of_points = 500
x_point = []
y_point = []

a = 0.22
b = 0.78

for i in range(number_of_points):
    x = np.random.normal(0.0,0.5)
    y = (-a*x+b)+np.random.normal(0.0,0.1)
    x_point.append(x)
    y_point.append(y)

plt.scatter(x_point,y_point,c='tab:blue')
plt.show()
```




```
real_x = np.array(x_point)
real_y = np.array(y_point)

# Definicja błędu:

x = tf.constant([1.0, 2.0, 3.0, 4.0])
tf.reduce_mean(x)

def loss_fn(real_y, pred_y):
    return tf.reduce_mean((real_y - pred_y)**2)

import random
a = tf.Variable(random.random())
b = tf.Variable(random.random())

Loss = []
epochs = 50

for _ in range(epochs):
  with tf.GradientTape() as tape:
    pred_y = a * real_x + b
    loss = loss_fn(real_y, pred_y)
    Loss.append(loss.numpy())
```

```
  dloss_da, dloss_db = tape.gradient(loss,(a, b))

  a.assign_sub(0.1*dloss_da)
  b.assign_sub(0.1*dloss_db)

np.max(Loss),np.min(Loss)

print(a.numpy())
print(b.numpy())

plt.scatter(np.arange(epochs),Loss)
plt.show()

max = np.max(x_point)
min = np.min(x_point)

X = np.linspace(min, max, num=10)
plt.plot(X,a.numpy()*X+b.numpy(),c='r')
plt.scatter(x_point,y_point,c="b")
plt.show()
```

```
-0.18407144
0.7833143
```