Generate Data (Make Regression)

Sklearn.datasets.make_regression(n_sample=100,n_features=100,*,n_informative=10,n_targets=1,bias=0.

n_samples int, default=100 The number of samples.

n_features int, default=100 The number of features.

n_informative int,default=10 The number of informative features, i.e.the number of features used to build the linear model used to generate the output.

n_targets int, default=1 The number of regression targets,i.e. the dimension of the y output vector associated with a sample. By default,the output is a scalar.

bias float,default=0.0 The bias term in the underlying linear model.

effective_rank int, default=None if not None:The approximate number of singular vectors required to explain most of the input data by linear combinations.Using this kind of singular spectrum in the input allows the generator to reproduce the corrections often observed in practice.

if None: The input set is well conditioned, centered and gaussian with unit variance.

tail_strength float default=0.5 The relative importance of the fat noisy tail of the singular values profile if effective_rank is not None.

noise float,default=0.0 the standard deviation of the gaussian noise applied to the output.

shuffle bool, default=False if True shuffle the samples and the features.

coef bool, default=False if True, the coefficients of the underlying linear model are returned.

random_state int, RandomState instance or None, default=None Determines random number generation for dataset creation.Pass an int for reproducible output across multiple function calls.

In [1]:

```
from sklearn.datasets import make_regression
```

In [2]:

```
X,y,w = make_regression(n_samples=1000,n_features=1,coef=True,bias=100,noise=10,random_stat
```

In [3]:

```
X,y = make_regression(n_samples=1000,n_features=1,coef=False,bias=100,noise=10,random_state
```

In [4]:

```
X.shape, y.shape
```

Out[4]:

```
((1000, 1), (1000,))
```

In [5]:

```
w
```

Out[5]:

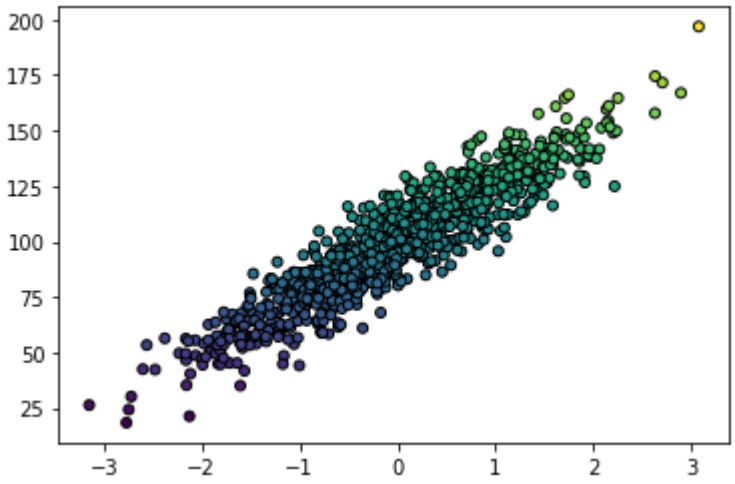
```
array(24.96405085)
```

In [6]:

```
import matplotlib.pyplot as plt
```

In [7]:

```
plt.scatter(X,y,marker="o",c=y,s=25,edgecolor="k");
```



In [ ]:

```

```