Underfitting and Overfitting

In this notebook we will do a regression problem and see the effect of model complexity on accuracy.

We will do a polynomial fitting on data

Imports

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

Generating the dataset

We will generate a data using a cosinus function.

In [2]:
```
def true_fun(X):
    return np.cos(1.5 * np.pi * X)
```

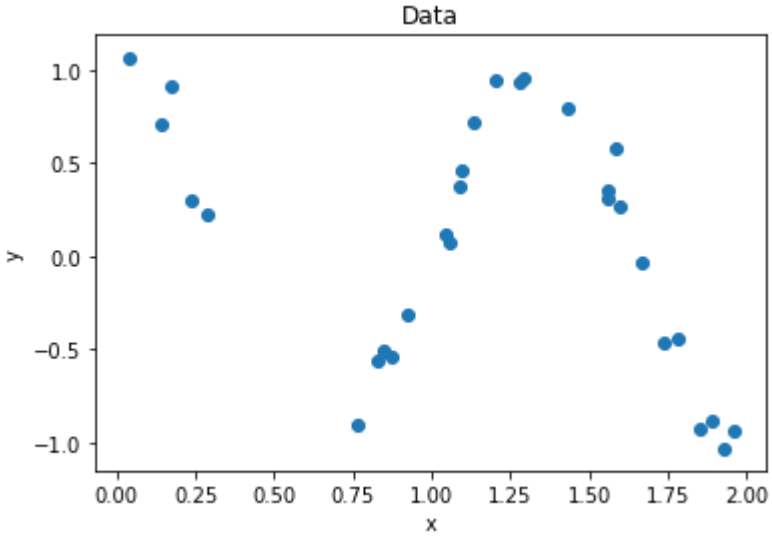
In [3]:
```
np.random.seed(0)
n_samples = 30

X = np.sort(np.random.rand(n_samples)*2)
y = true_fun(X) + np.random.randn(n_samples) * 0.1
```

Visualizing the data

In [9]:
```
plt.title('Data')
plt.scatter(X, y)
plt.xlabel('x')
plt.ylabel('y')
```

Out[9]: Text(0,0.5,u'y')

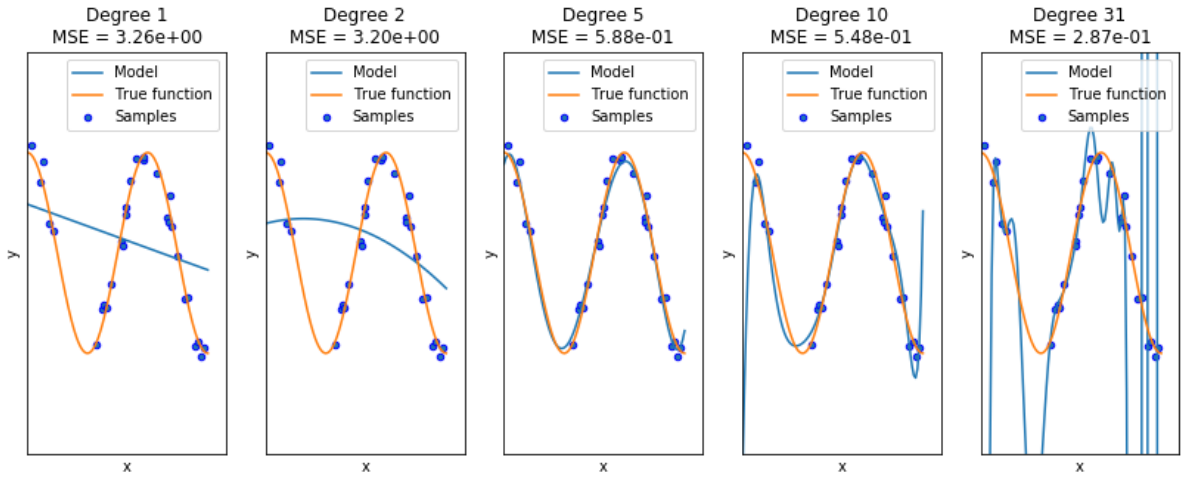


In [10]:
```
from fit_plot import fit_plot
'''This functin plots the true function, the sample,
and the model found with the given degree of the polynomial'''
```

Out[10]: 'This functin plots the true function, the sample,\nand the model found with the given degree of the polynomial'

Testing different degrees

In [11]:
```
degrees = [1, 2, 5, 10, 31]
fit_plot(X, y, true_fun, degrees)
```



Underfitting is when the model is much less complex than reality.

Overfitting is when the model is more complex than reality.

In both cases, the generalization is weak. If we would decide based on the MSE score, we would have picked here the last fit (degree 31).