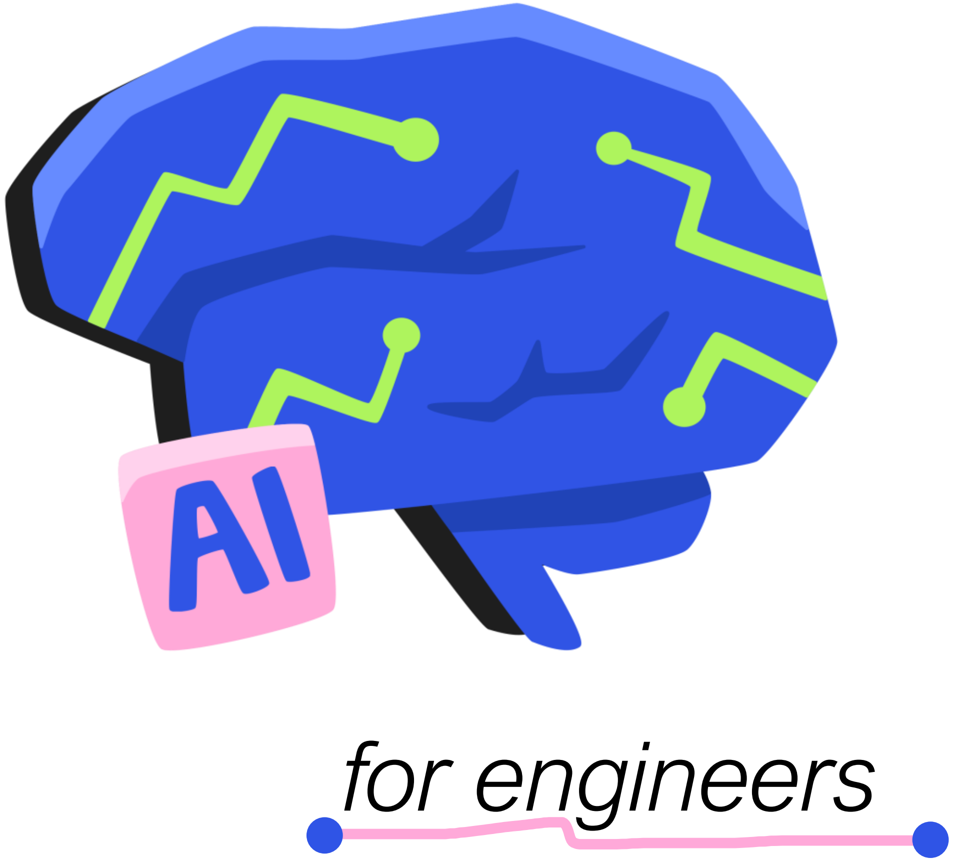


# AI for Engineers:

## Introduction to Python Libraries for Machine Learning

**Mina Farmanbar**

Associate Professor, University of Stavanger

**Python** is a popular and powerful programming language among data scientists.

There are a lot of modules and libraries already implemented in Python, that can make your life much easier

# NumPy

- A math library to work with N-dimensional arrays in Python.

- It enables you to do computation efficiently and effectively.

- For example, for working with arrays, dictionaries, functions, datatypes and working with images you need to know NumPy.

# SciPy

- SciPy is a collection of numerical algorithms and domain specific toolboxes, including signal processing, optimization, statistics and much more.

- SciPy is a good library for scientific and high-performance computation

# pandas

- Pandas library is a very high-level Python library that provides high performance easy to use data structures.
- It has many functions for data importing, manipulation and analysis.
- In particular, it offers data structures and operations for manipulating numerical tables and timeseries.

scikit learn

- SciKit Learn is a collection of algorithms and tools for machine learning which is our focus here and which you'll learn to use within this course.

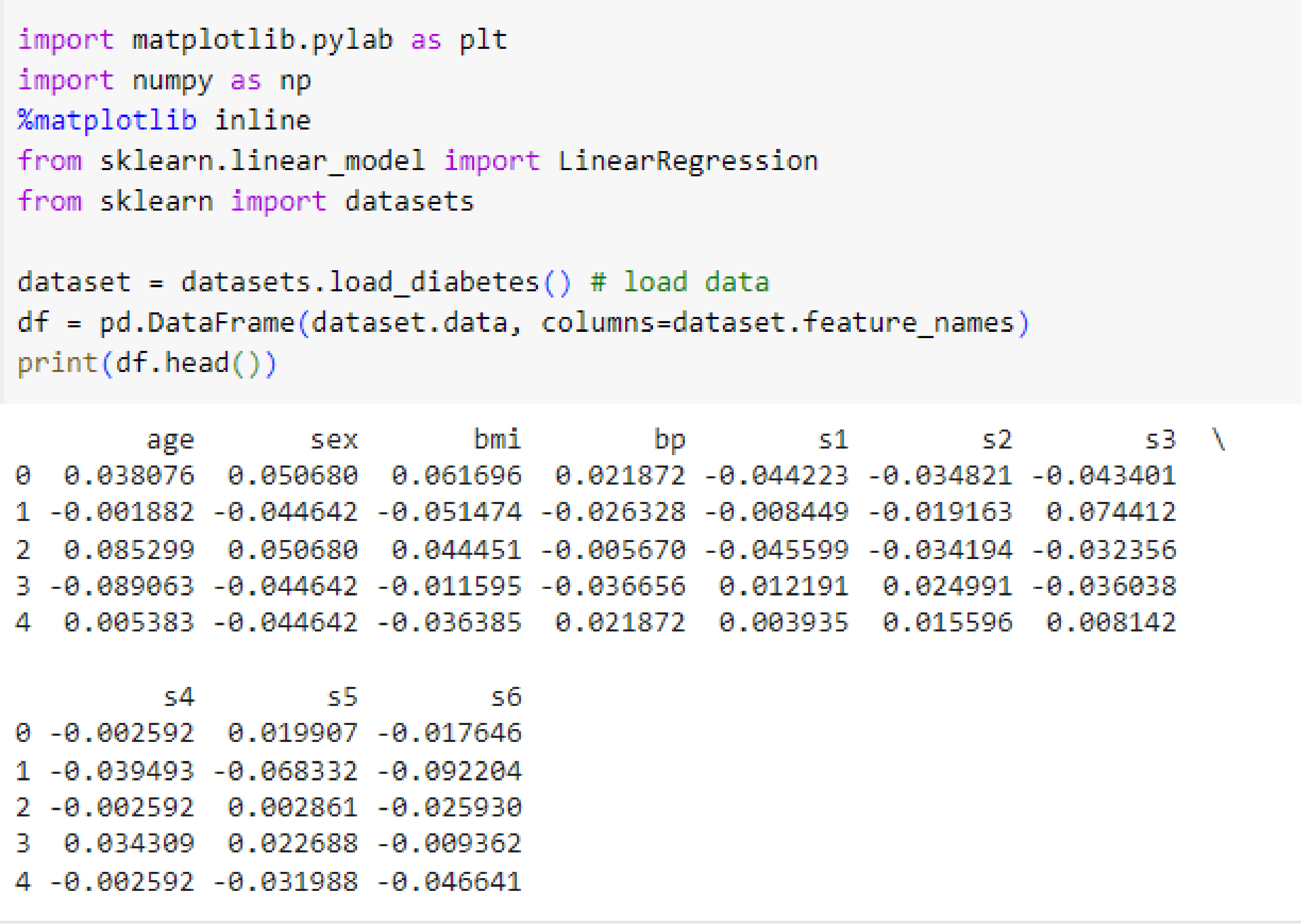
- It has most of the classification, regression and clustering algorithms, and it's designed to work with a Python numerical and scientific libraries: NumPy and SciPy.

# Scikit-learn features and functions

- Scikit-learn comes with several inbuilt datasets such as the iris dataset, house prices dataset, diabetes dataset, etc.
- Sklearn provided the functionality to split the dataset for training and testing
- With the help of sklearn, we can easily implement the machine learning techniques
- Finaly we can save

scikit learn

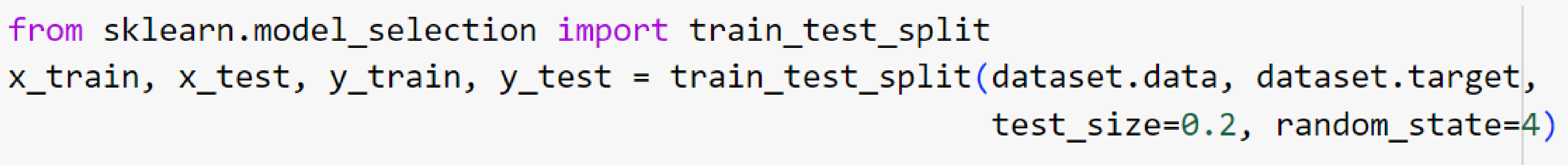
# Datasets



```
import matplotlib.pylab as plt
import numpy as np
%matplotlib inline
from sklearn.linear_model import LinearRegression
from sklearn import datasets

dataset = datasets.load_diabetes() # load data
df = pd.DataFrame(dataset.data, columns=dataset.feature_names)
print(df.head())
```

```
        age       sex       bmi        bp        s1        s2        s3  \
0  0.038076  0.050680  0.061696  0.021872 -0.044223 -0.034821 -0.043401
1 -0.001882 -0.044642 -0.051474 -0.026328 -0.008449 -0.019163  0.074412
2  0.085299  0.050680  0.044451 -0.005670 -0.045599 -0.034194 -0.032356
3 -0.089063 -0.044642 -0.011595 -0.036656  0.012191  0.024991 -0.036038
4  0.005383 -0.044642 -0.036385  0.021872  0.003935  0.015596  0.008142

         s4        s5        s6
0 -0.002592  0.019907 -0.017646
1 -0.039493 -0.068332 -0.092204
2 -0.002592  0.002861 -0.025930
3  0.034309  0.022688 -0.009362
4 -0.002592 -0.031988 -0.046641
```

# Data Splitting

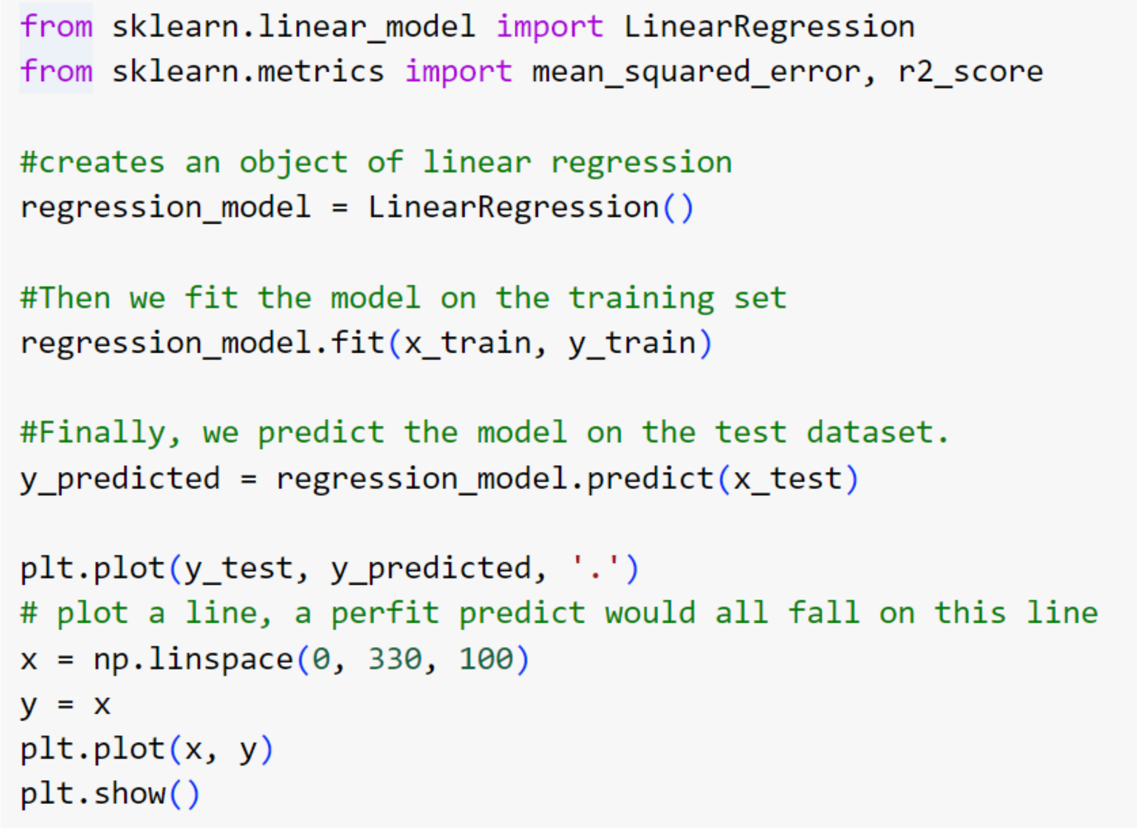


```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(dataset.data, dataset.target,
                                                    test_size=0.2, random_state=4)
```

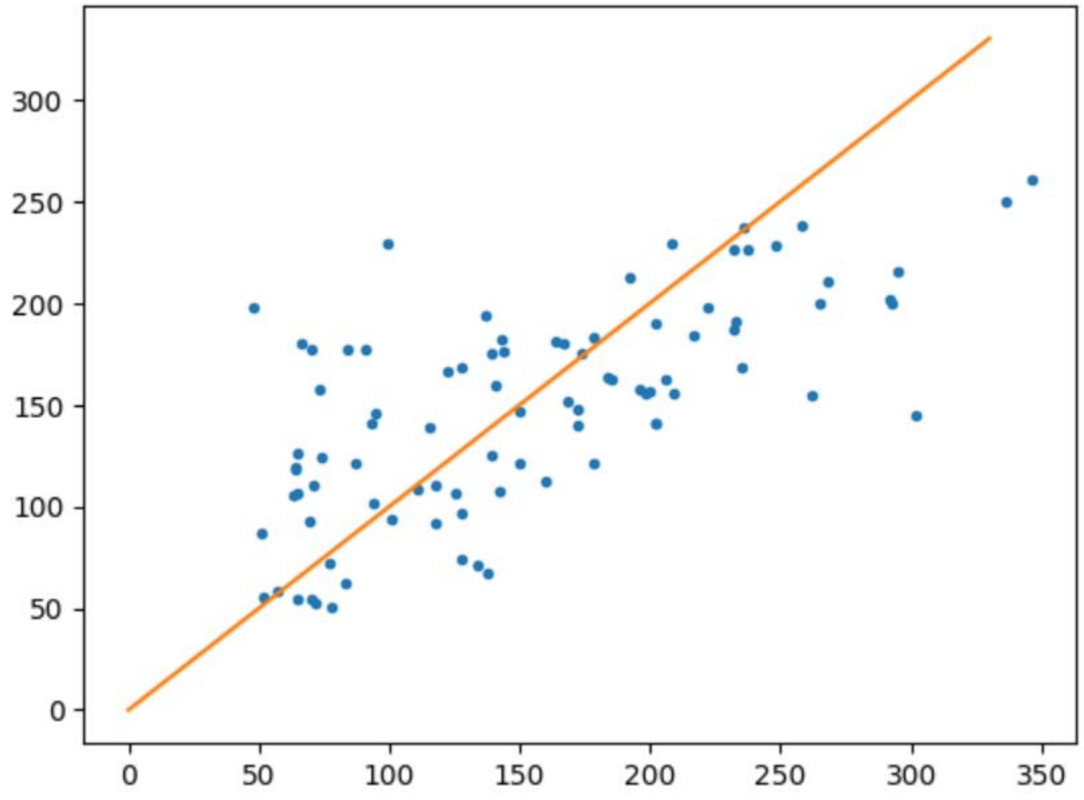
With the help of **train_test_split**, we have split the dataset such that the train set has 80% and the test set has 20% data.

scikit
learn

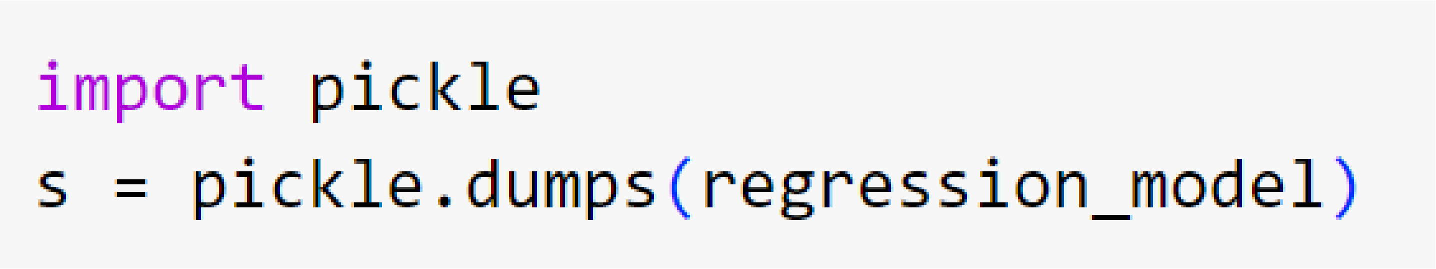
# Linear Regression



```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

#creates an object of linear regression
regression_model = LinearRegression()

#Then we fit the model on the training set
regression_model.fit(x_train, y_train)

#Finally, we predict the model on the test dataset.
y_predicted = regression_model.predict(x_test)

plt.plot(y_test, y_predicted, '.')
# plot a line, a perfit predict would all fall on this line
x = np.linspace(0, 330, 100)
y = x
plt.plot(x, y)
plt.show()
```



# Save the model



```
import pickle
s = pickle.dumps(regression_model)
```

# Summary

- You may find all or some of these machine-learning terms confusing but don't worry, we'll talk about all of these topics in the following modules.
- The most important point to remember is that the entire process of a machine learning task can be done simply in a few lines of code using SciKit Learn.
- And of course, it needs much more coding if you use pure Python programming to implement all of these tasks.

# Thank you!

Mina Farmanbar

mina.farmanbar@uis.no