Data Science: Machine Learning: Scikit

CPSC 501: Advanced Programming Techniques
Fall 2022

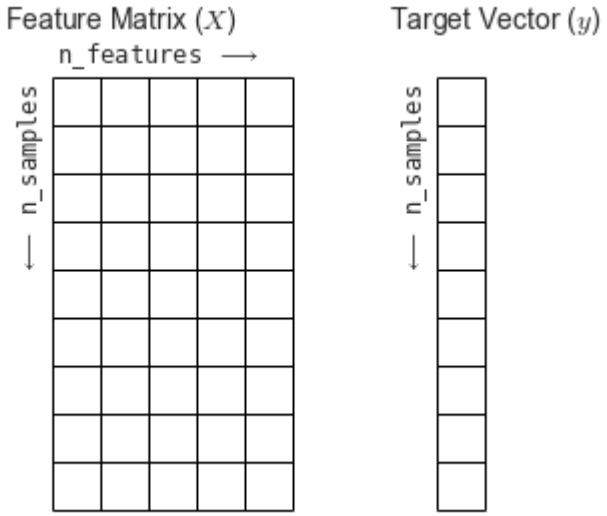
Jonathan Hudson, Ph.D
Assistant Professor (Teaching)
Department of Computer Science
University of Calgary

Thursday, October 13, 2022

UNIVERSITY OF CALGARY

scikit-learn

- scikit-learn is one of most common machine learning libraries
- Contains many, many boilerplate algorithms that take a couple lines of code of plug into data science operation
- Also a lot of syntax is mappable between different models to the point very different structural ones can often be swapped with a simple copy paste of one name
- Most models work on this structure
- Have array of data
  - split into input on left and output on right
- Apply model generally to find a relationship (left -> right)

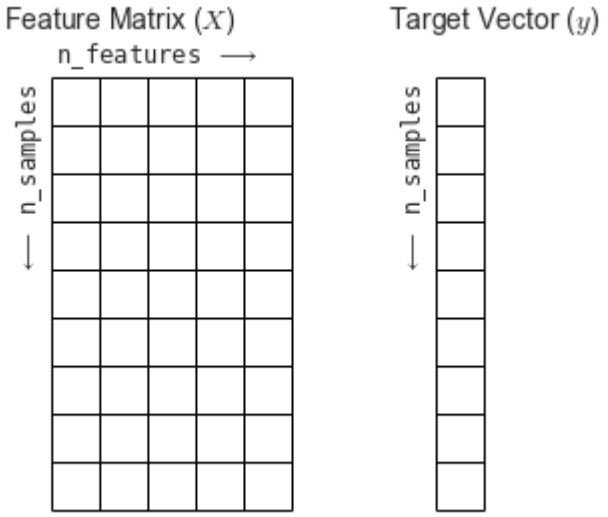


Design

- Consistency: comm interface
- Inspection: expose parameters as public attributes
- Limited object hierarchy: only use standard arrays, numpy, pandas
- Composition: make use of sequence of standard algorithms, instead of making groupings
- Sensible defaults: have common defaults so only advanced usage needs setup

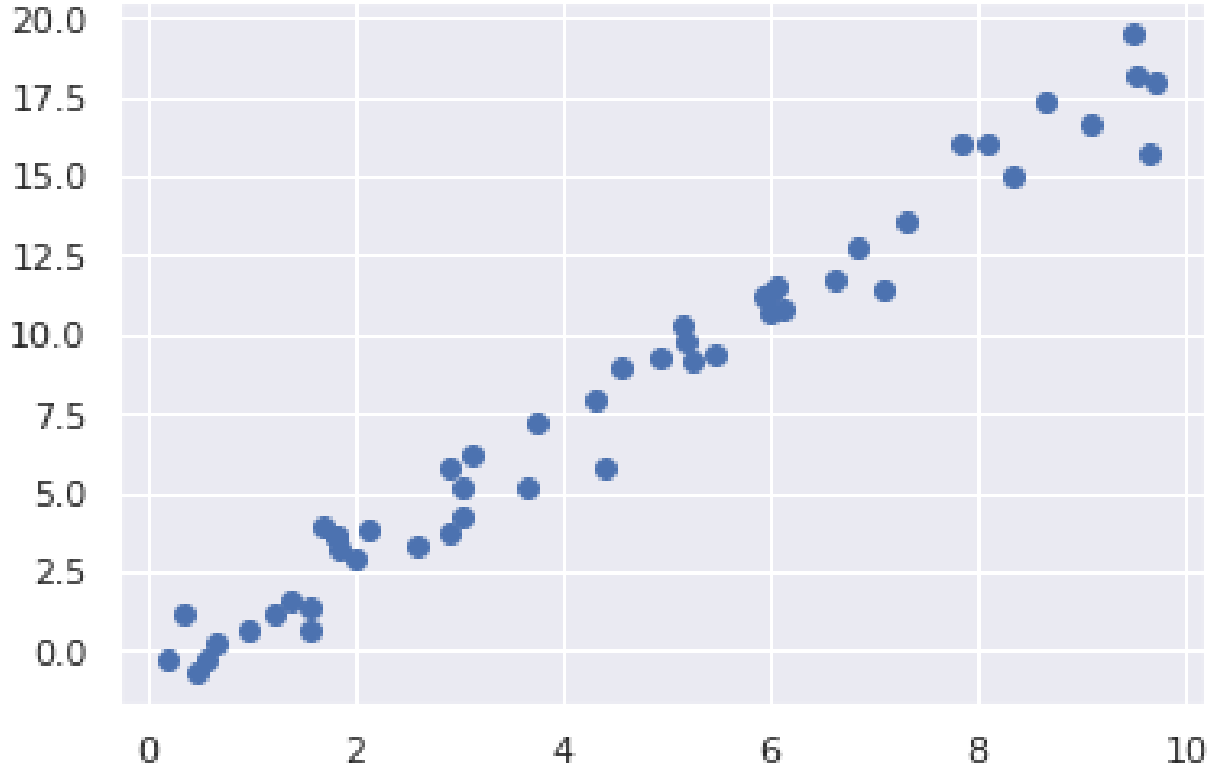
Process of Estimator API

- Choose model and import
- Choose parameters
- Arrange data into feature/target
- Call fit() on the estimator model
- Apply model to new data
  - predict for supervised learning
  - predict/transform for unsupervised learning



Example linear regression

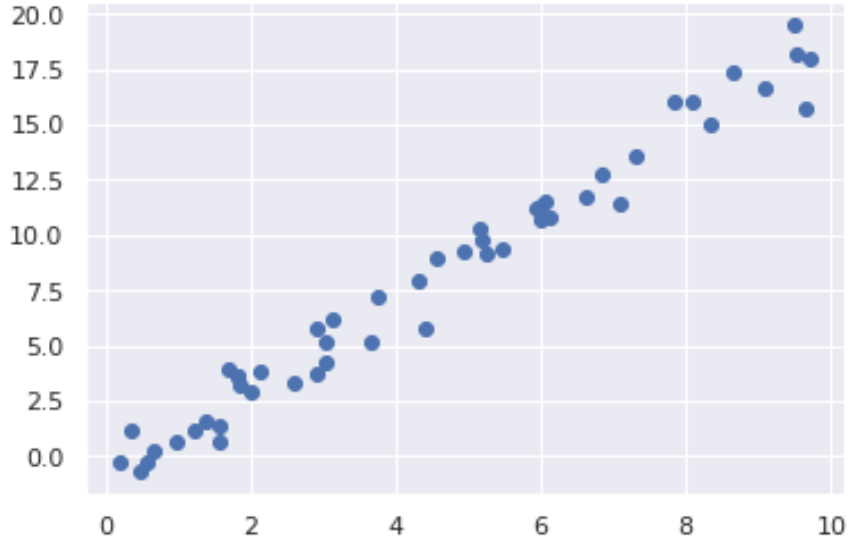
```
import matplotlib.pyplot as plt
import numpy as np

rng = np.random.RandomState(42)
x = 10 * rng.rand(50)
y = 2 * x - 1 + rng.randn(50)
plt.scatter(x, y);
```



Example linear regression

Our model is a line of best fit

Used to predict future values

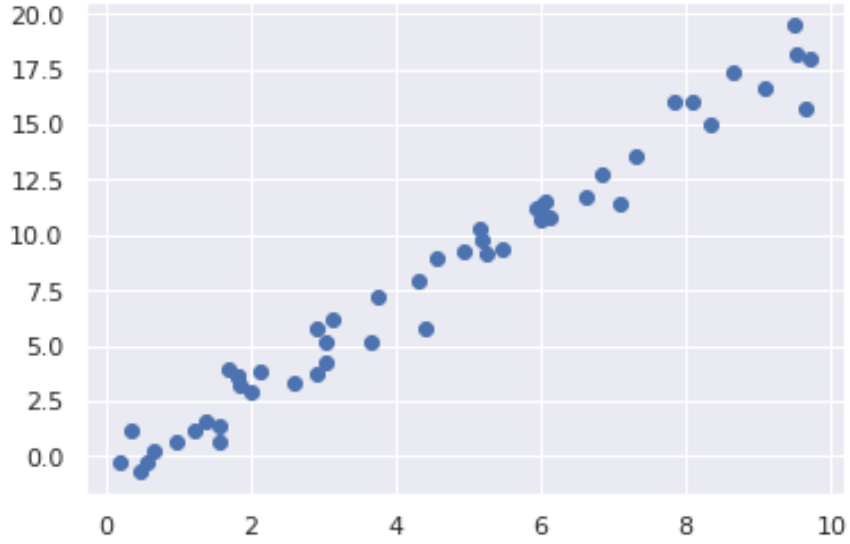



```
from sklearn.linear_model import LinearRegression
```

Example hyper parameters (not necessary)

1. fit for the offset (i.e., y-intercept)?
2. model to be normalized?
3. preprocess our features to add model flexibility?
4. degree of regularization?
5. how many model components?

Example linear regression



Our model is a line of best fit

Used to predict future values

```
from sklearn.linear_model import LinearRegression
```

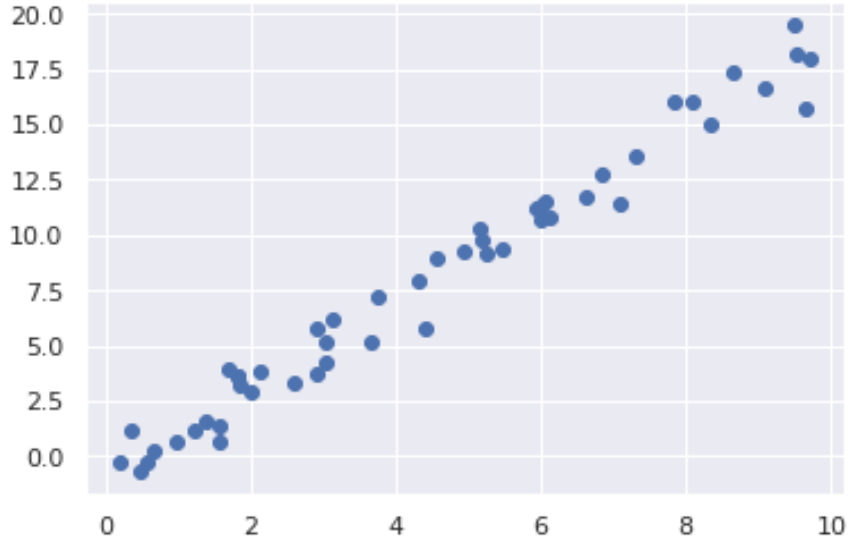
```
model = LinearRegression(fit_intercept=True)
```

or (defaults will be filled for us like or we can put them in)

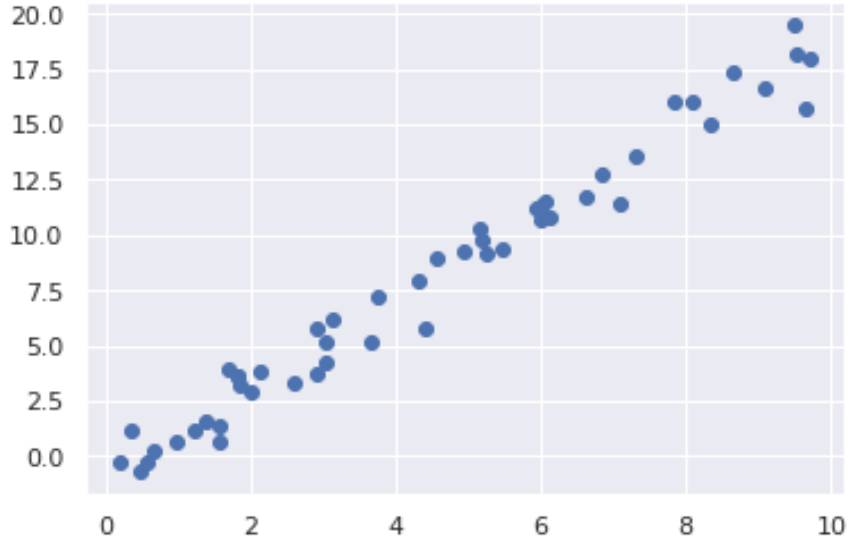
```
model = LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

Example linear regression

```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)

#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
```



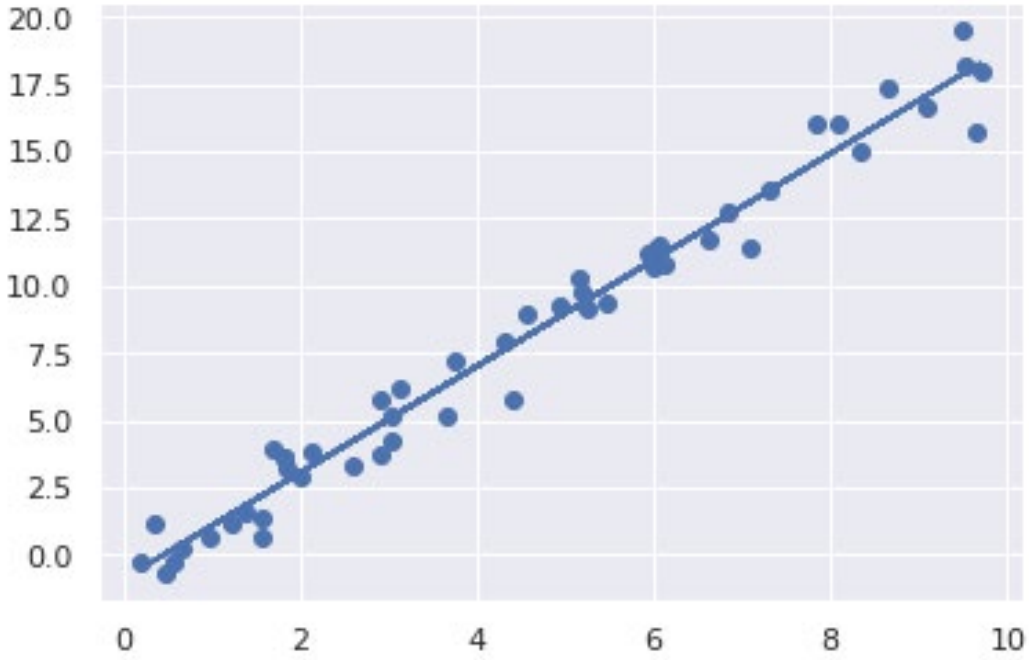
Example linear regression




```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)


#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
print(f"y = {float(model.intercept_):.2f} + x * {float(model.coef_[0]):.2f}")
```

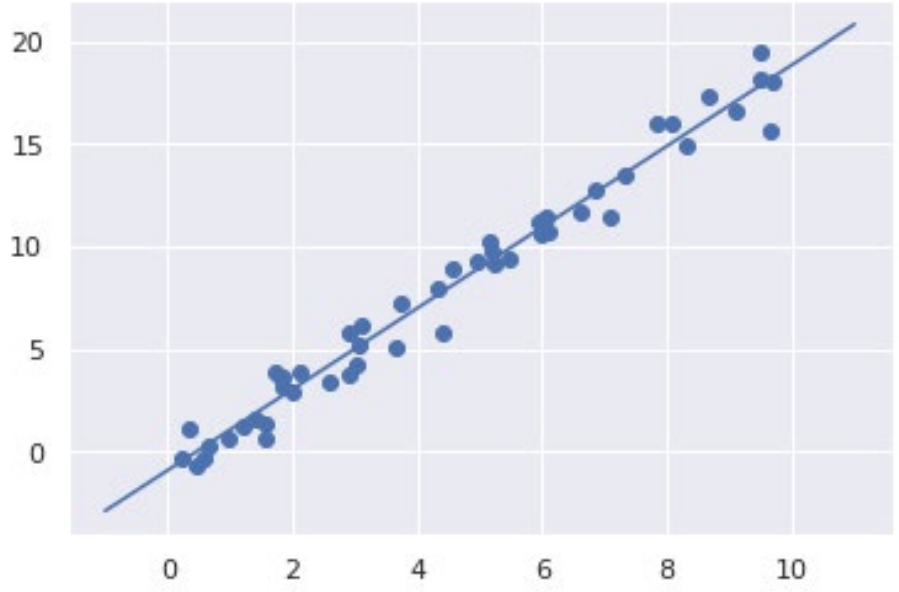
Example linear regression

```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)


#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
print(f"y = {float(model.intercept_):.2f} + x * {float(model.coef_[0]):.2f}")
plt.scatter(x, y);
plt.plot(x, float(model.intercept_)+float(model.coef_[0])*x )
```



Example linear regression

```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)


#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
plt.scatter(x, y);
xfit = np.linspace(-1, 11) [:, np.newaxis]    #Make 2d array across range -1 to 11
yfit = model.predict(Xfit)                    #predict it
plt.plot(xfit, yfit);                         #plot it
```



Available Models

https://scikit-learn.org/stable/user_guide.html

1. Supervised learning

1.1. Linear Models

1.2. Linear and Quadratic Discriminant Analysis

1.3. Kernel ridge regression

1.4. Support Vector Machines

1.5. Stochastic Gradient Descent

1.6. Nearest Neighbors

1.7. Gaussian Processes

1.8. Cross decomposition

1.9. Naive Bayes

1.10. Decision Trees

1.11. Ensemble methods

1.12. Multiclass and multioutput algorithms

1.13. Feature selection

1.14. Semi-supervised learning

1.15. Isotonic regression

1.16. Probability calibration

1.17. Neural network models (supervised)

Available Models

https://scikit-learn.org/stable/user_guide.html

2. Unsupervised learning

2.1. Gaussian mixture models

2.2. Manifold learning

2.3. Clustering

2.4. Biclustering

2.5. Decomposing signals in components (matrix factorization problems)

2.6. Covariance estimation

2.7. Novelty and Outlier Detection

2.8. Density Estimation

2.9. Neural network models (unsupervised)

7. Dataset loading utilities (learning helpers)

7.1. Toy datasets

7.2. Real world datasets

7.3. Generated datasets

7.4. Loading other datasets

Onward to …
neural networks.

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~jwhudson/

UNIVERSITY OF CALGARY