DATA 301: Linear Regression review
Ordinary Least Squares (OLS)

Topics

Training Overview
OLS Linear Regression – outline
OLS Linear Regression - scikit-learn

Training a Model - Soon

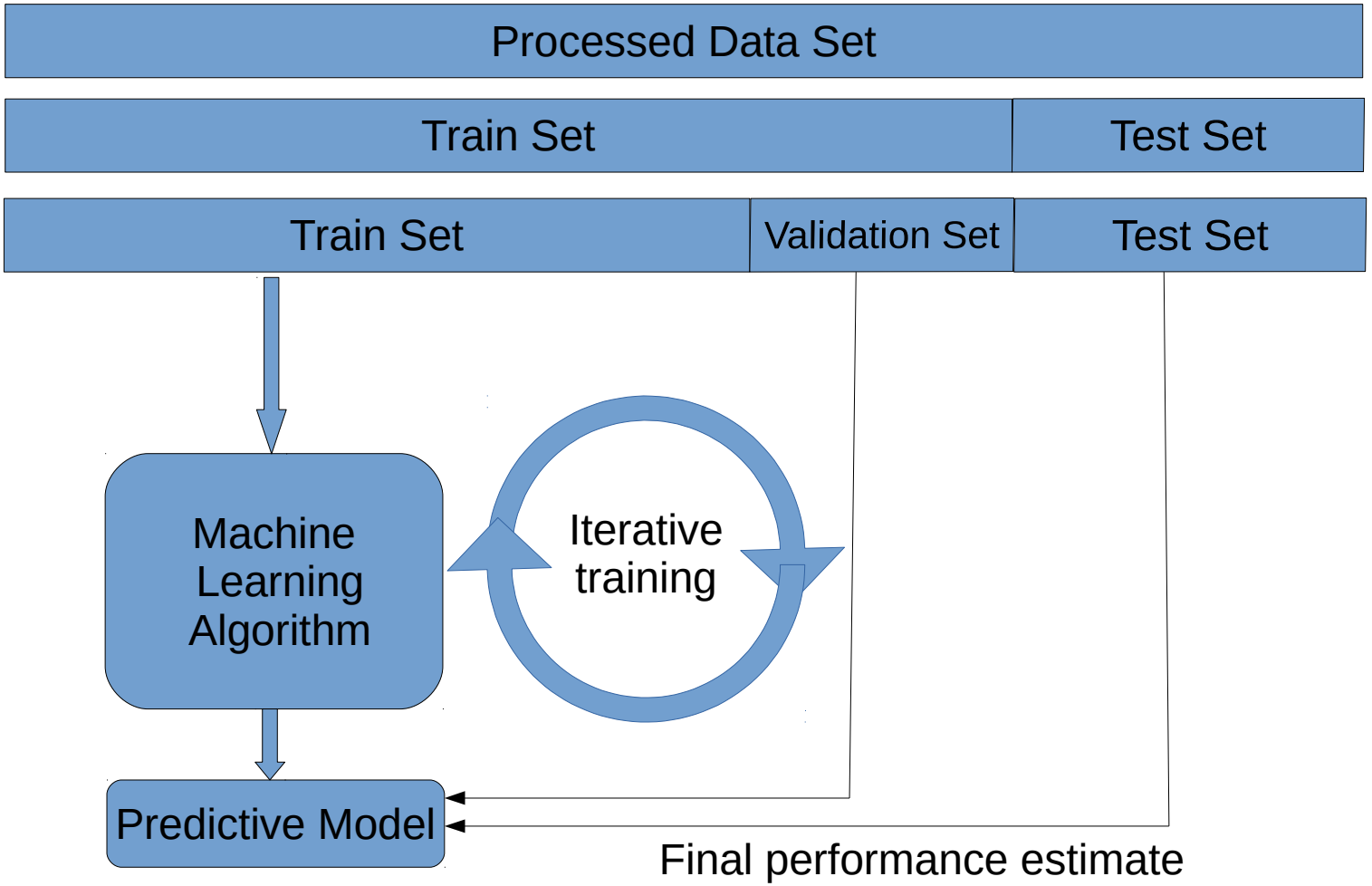


Figure from 'Python Machine Learning' by Sebastian Raschka

Training a Model - Soon

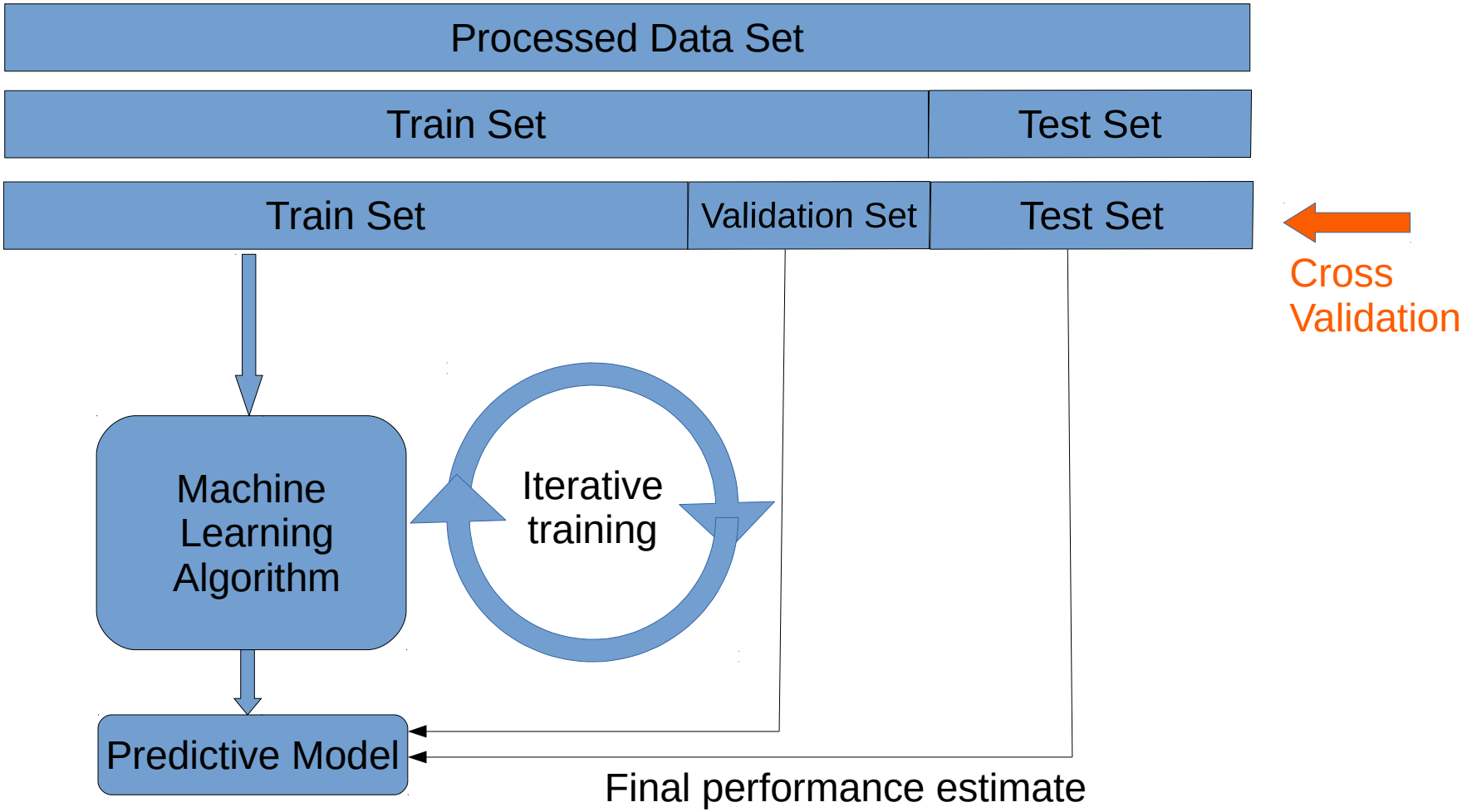


Figure from 'Python Machine Learning' by Sebastian Raschka

Training a Model - Today

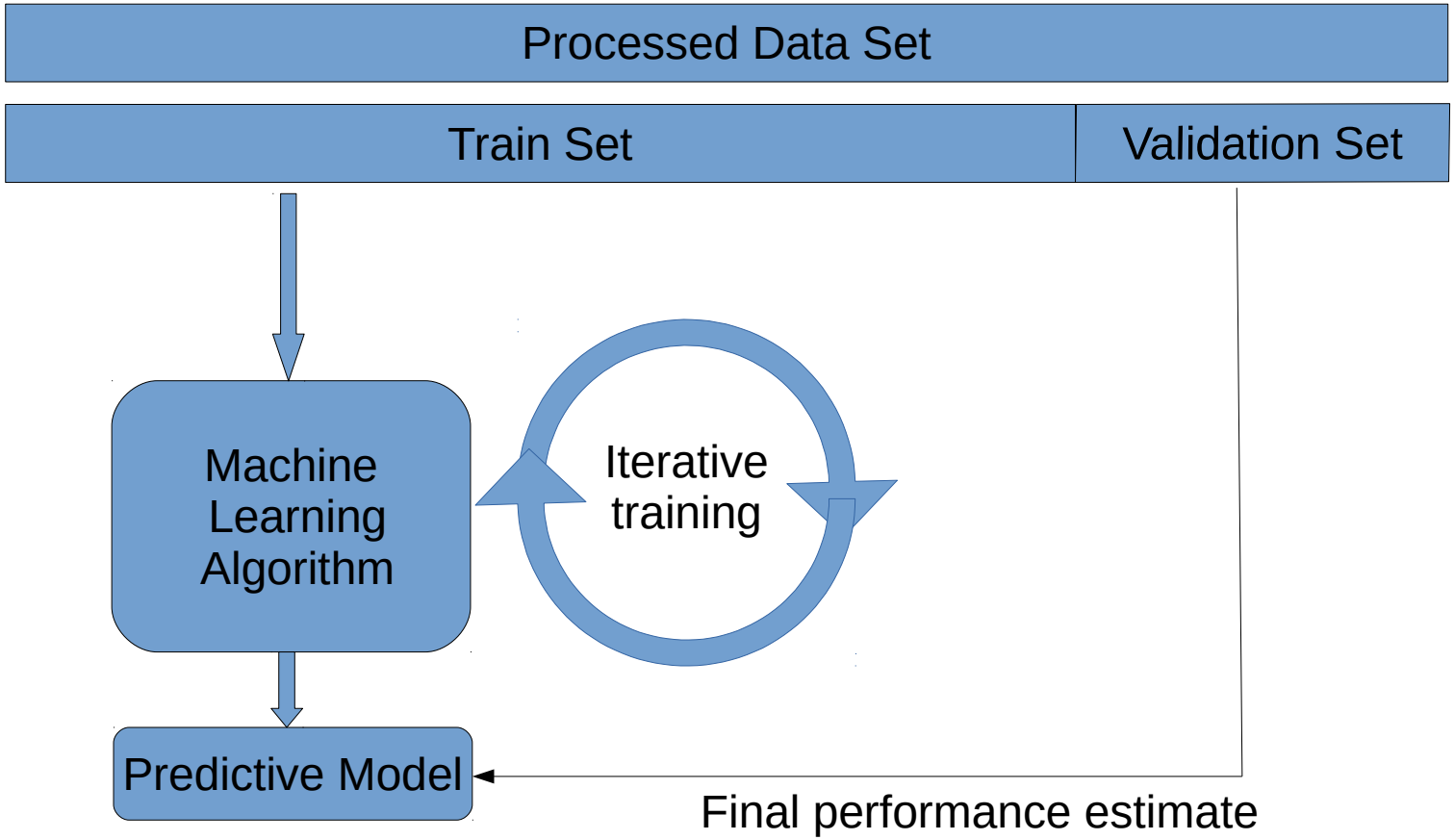
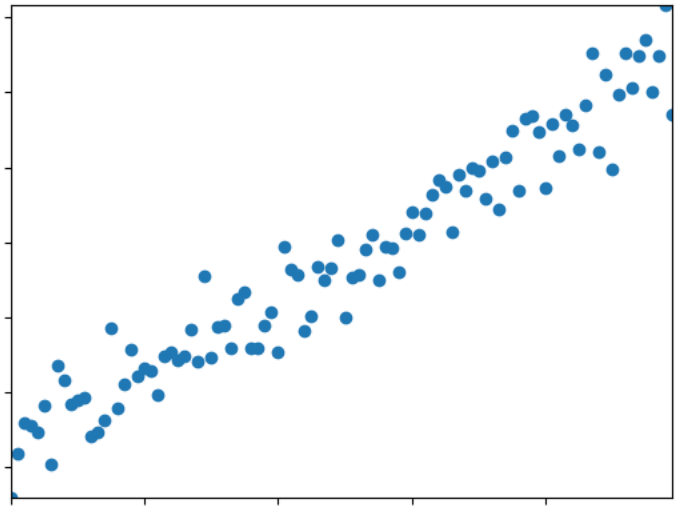


Figure from 'Python Machine Learning' by Sebastian Raschka

Training a Model

Ex. Linear regression iteratively estimates w terms in this equation

$$\hat{y}(w, x) = w_0 + w_1 x_1 + \ldots + w_p x_p$$



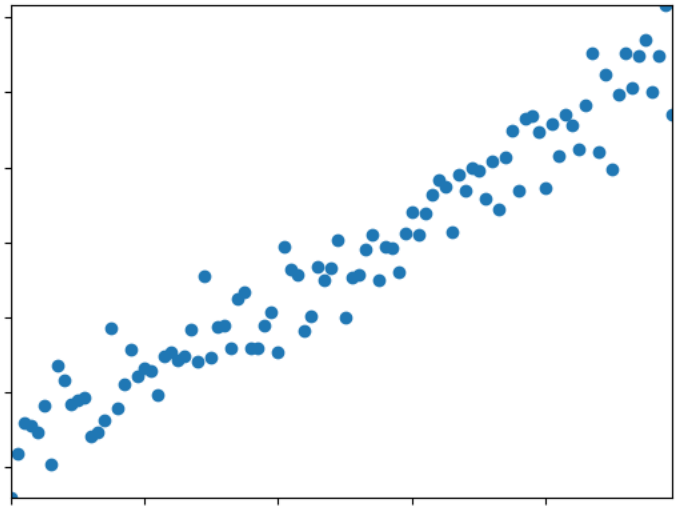
Training a Model

Ex. Linear regression iteratively estimates w terms in this equation

$$\hat{y}(w, x) = w_0 + w_1 x_1 + \ldots + w_p x_p$$

By reducing the error between actual and predicted values using this equation

$$\min_w ||Xw - y||_2^2$$



Training a Model

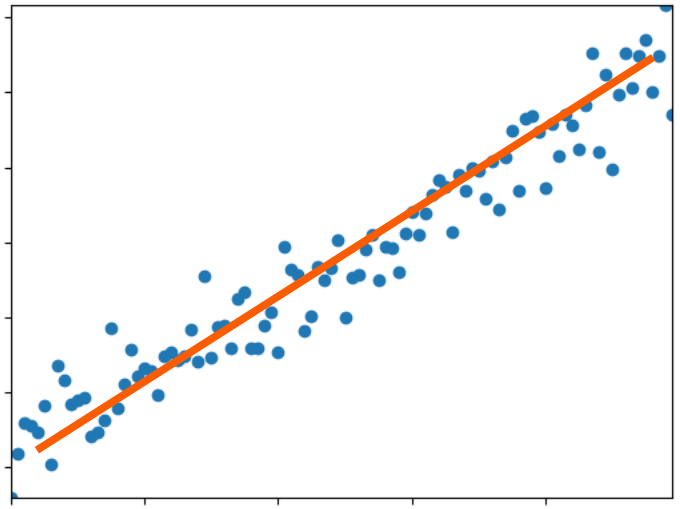
Ex. Linear regression iteratively estimates w terms in this equation

$$\hat{y}(w, x) = w_0 + w_1 x_1 + \ldots + w_p x_p$$

By reducing the error between actual and predicted values using this equation

$$\min_w ||Xw - y||_2^2$$

To generate a best fit line



Training a Model

Ex. Linear regression- In scikit learn;

```
#split into training and test
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=42)
```

Training a Model

Ex. Linear regression- In scikit learn;

```
#split into training and test
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=42)
```

```
from sklearn import linear_model
reg = linear_model.LinearRegression()
reg.fit(X=X_train, y=y_train);
```

Training a Model

Ex. Linear regression- In scikit learn;

```
#split into training and test
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=42)
```

```
from sklearn import linear_model
reg = linear_model.LinearRegression()
reg.fit(X=X_train, y=y_train);
```

```
#predict on new data
y_pred = reg.predict(X_test)
```

Training a Model

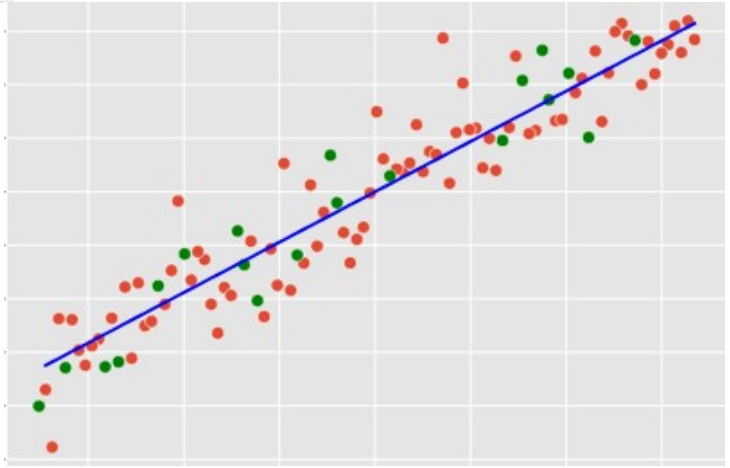
Ex. Linear regression- In scikit learn;

```
#split into training and test
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=42)
```

```
from sklearn import linear_model
reg = linear_model.LinearRegression()
reg.fit(X=X_train, y=y_train);
```

```
#predict on new data
y_pred = reg.predict(X_test)
```

```
#plot points and linear regression line
ax=sns.scatterplot(x=X_train.squeeze(), y=y_train)
ax=sns.scatterplot(x=X_test.squeeze(), y=y_test, color='green')
ax=sns.lineplot(x=X_train.squeeze(), y=reg.predict(X_train),color='blue' )
```



Training a Model

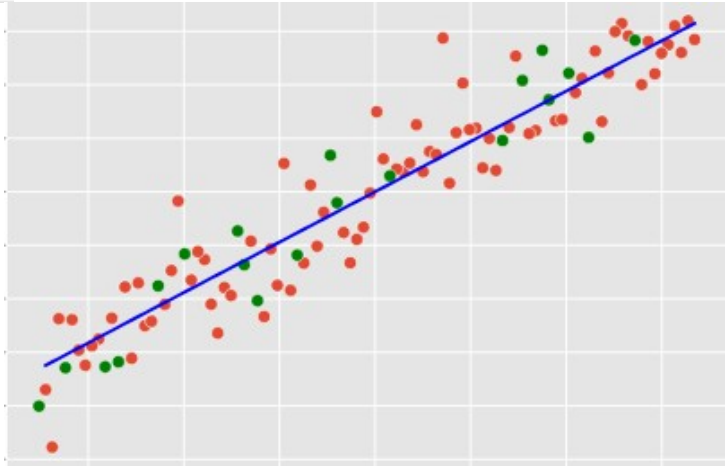
Ex. Linear regression- In scikit learn;

```
#split into training and test
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=42)
```

```
from sklearn import linear_model
reg = linear_model.LinearRegression()
reg.fit(X=X_train, y=y_train);
```

```
#predict on new data
y_pred = reg.predict(X_test)
```

```
#plot points and linear regression line
ax=sns.scatterplot(x=X_train.squeeze(), y=y_train)
ax=sns.scatterplot(x=X_test.squeeze(), y=y_test, color='green')
ax=sns.lineplot(x=X_train.squeeze(), y=reg.predict(X_train),color='blue' )
```



```
from sklearn.metrics import mean_squared_error, r2_score

#mean squared error?
print("Mean squared error for test: %.2f" % mean_squared_error(y_test, y_pred))
print("Mean squared error for train: %.2f" % mean_squared_error(y_train, reg.predict(X_train)))
```

```
Mean squared error for test: 137.19
Mean squared error for train: 118.65
```

Summary

Standardize on train set only
Scikitlearn StandardScaler usage
Scikitlearn LinearRegression