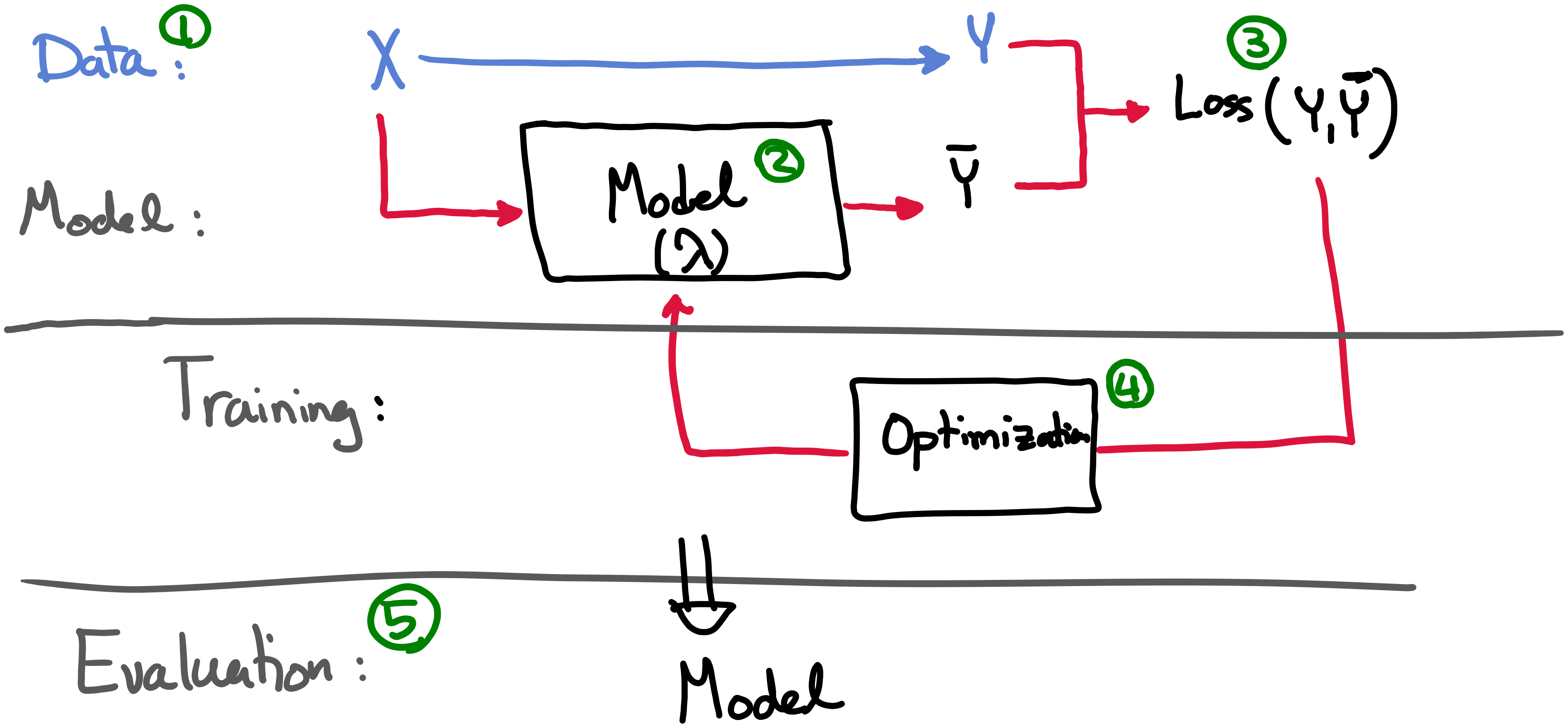
Machine Learning in Physics: Model Evaluation

Sadegh Raeisi

Supervised: Ingredients

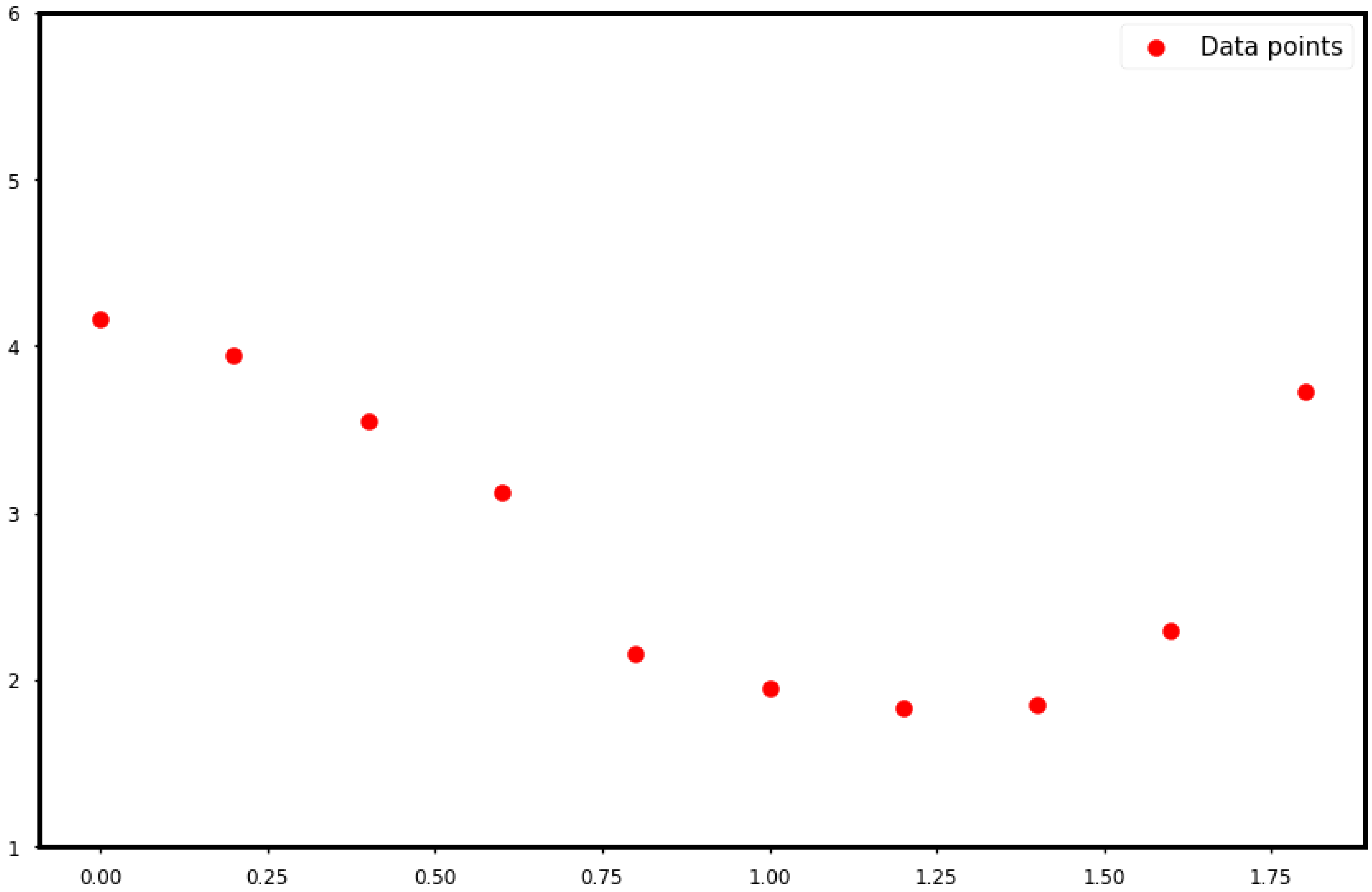


Outline

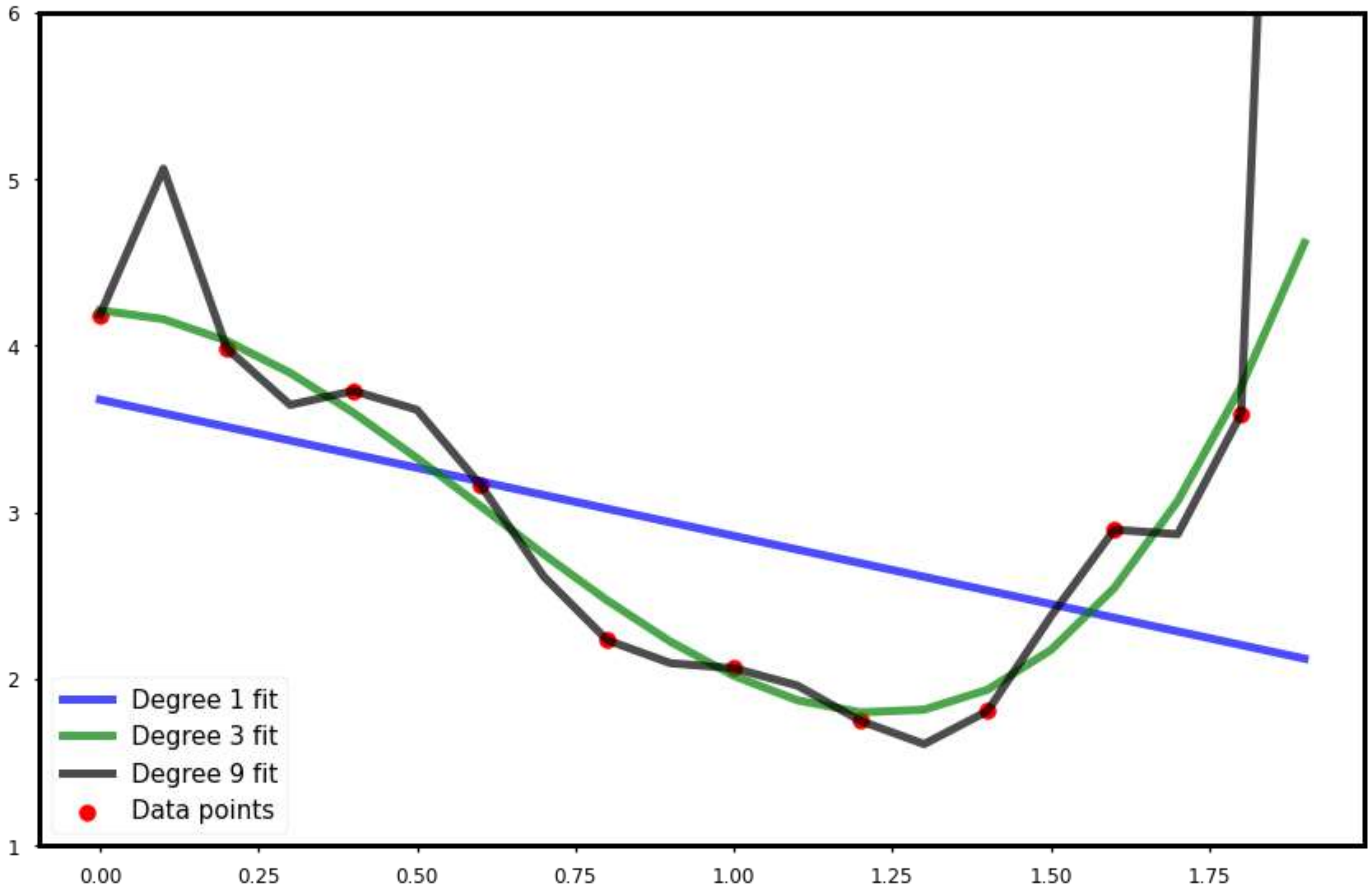
- What’s a good model?
- Bias and Variance
- Metrics
- Model Tuning

A good model

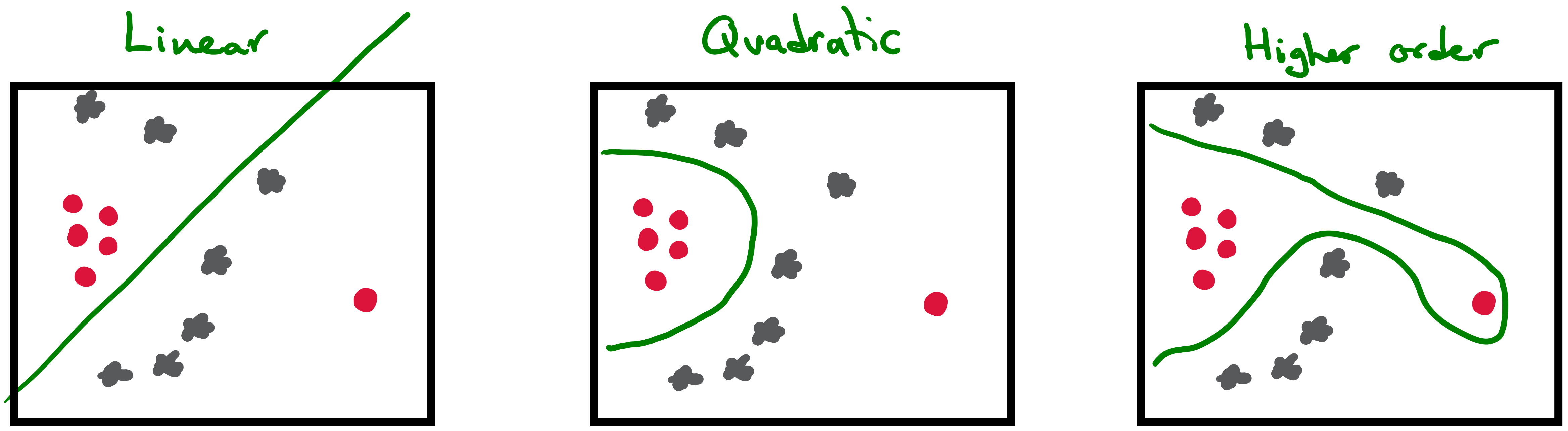
A good fit vs a good model



A good fit vs a good model

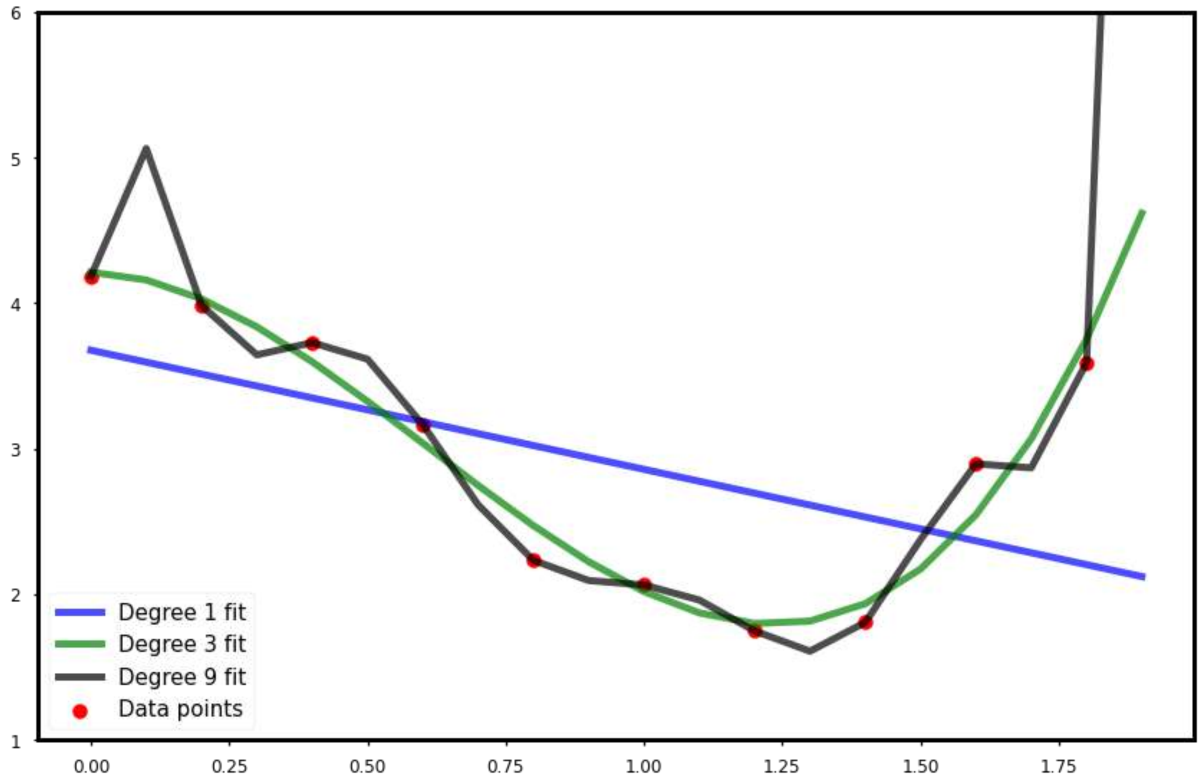


A good fit vs a good model

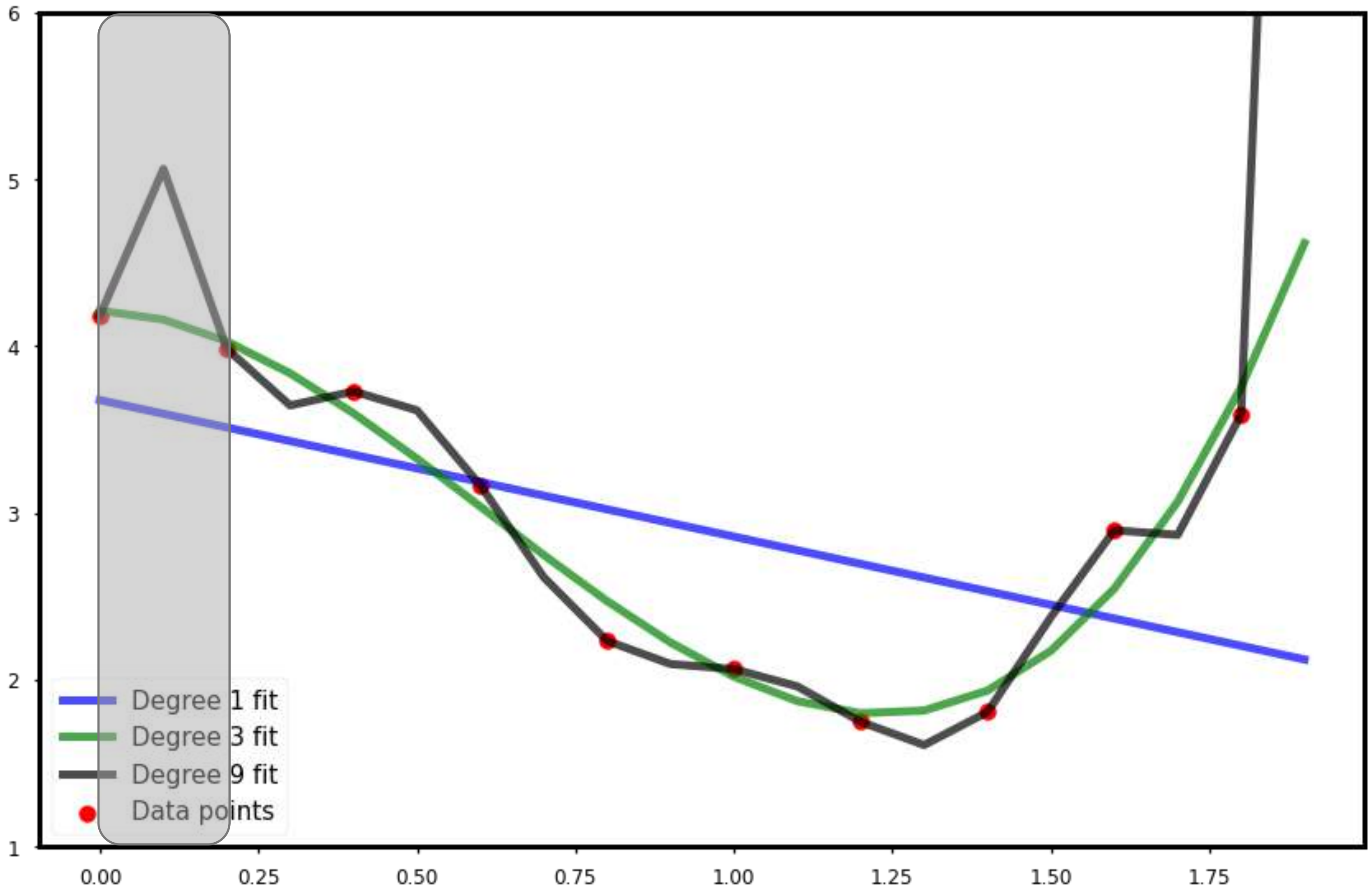


What’s the problem?

How can we solve it?

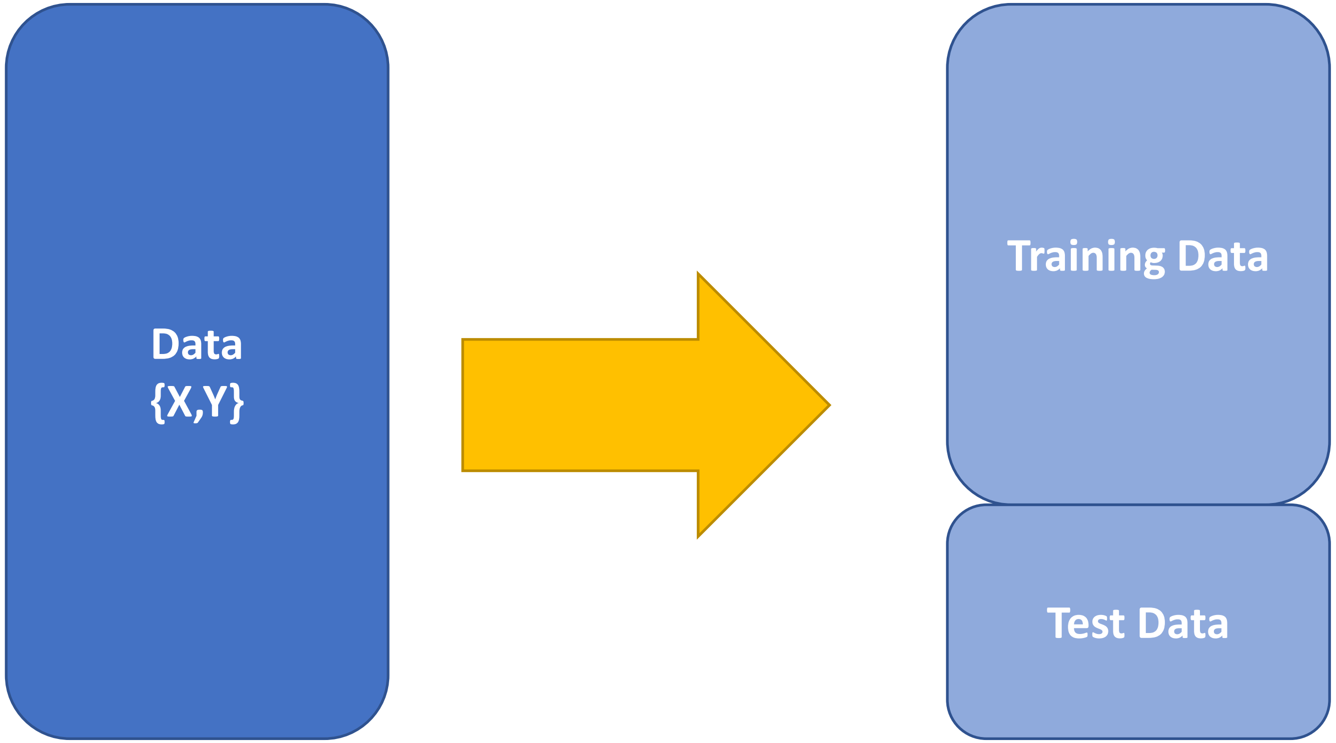


Good fit vs Good Prediction



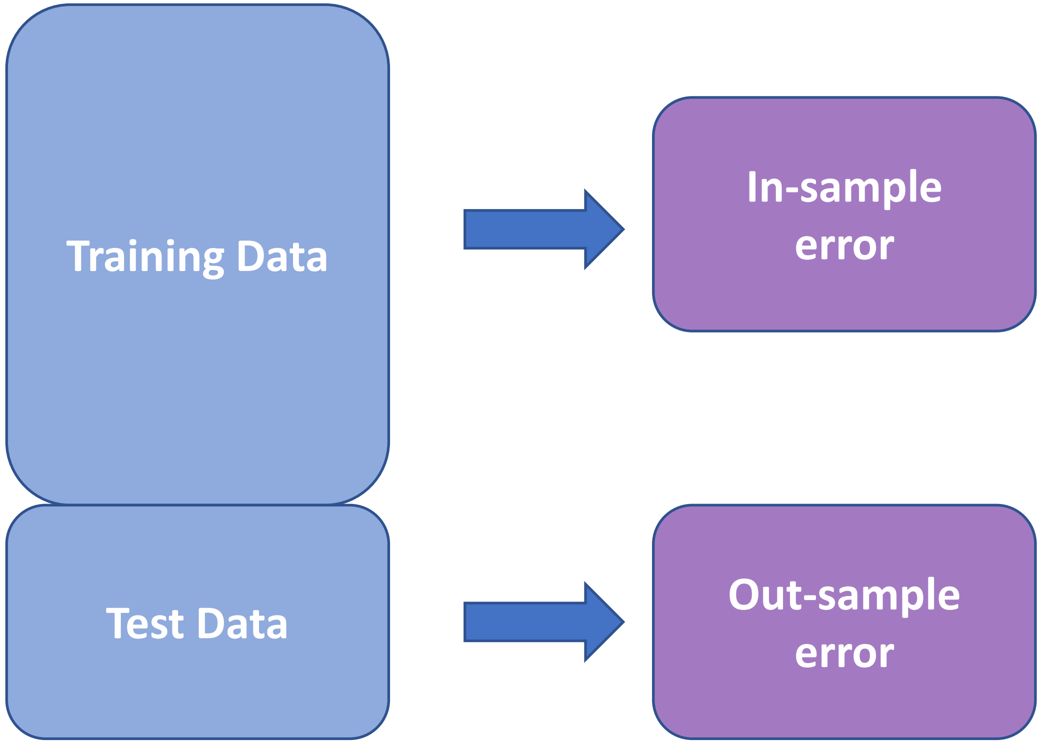
How can check the prediction power of a model?

- In-sample vs out sample error



Bias vs Variance

Paper

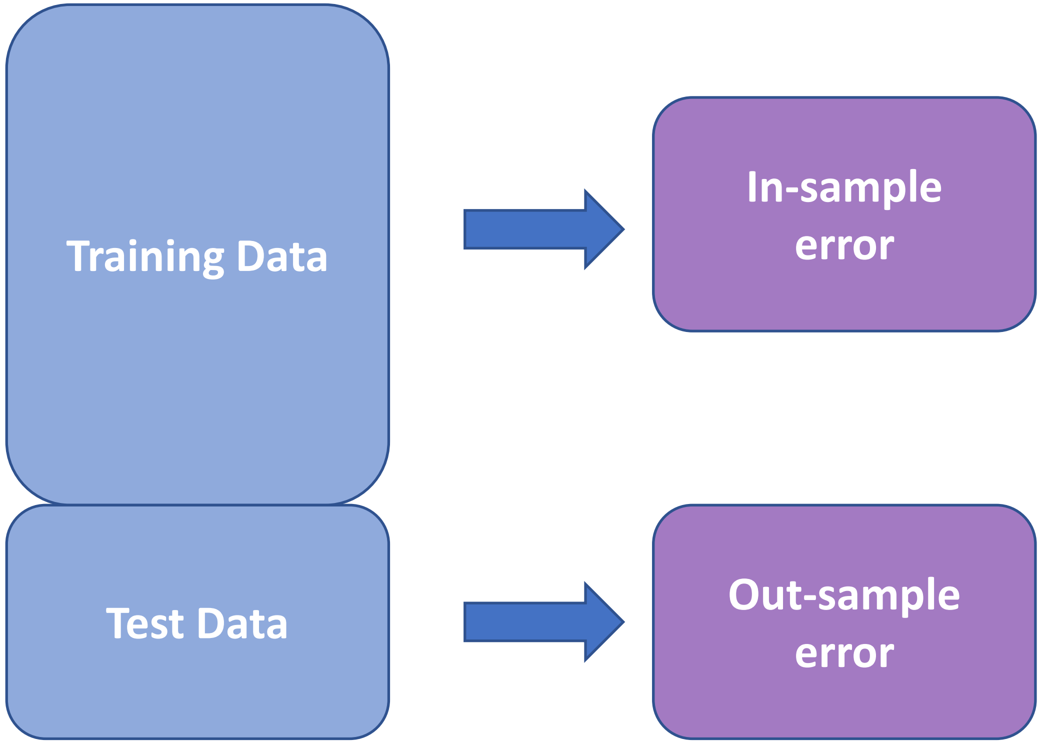


What do they depend on?

How can we reduce them?

Which one do we care more about?

Bias and Variance



What do they depend on?

How can we reduce them?

Which one do we care more about?

Bias and Variance: more detail

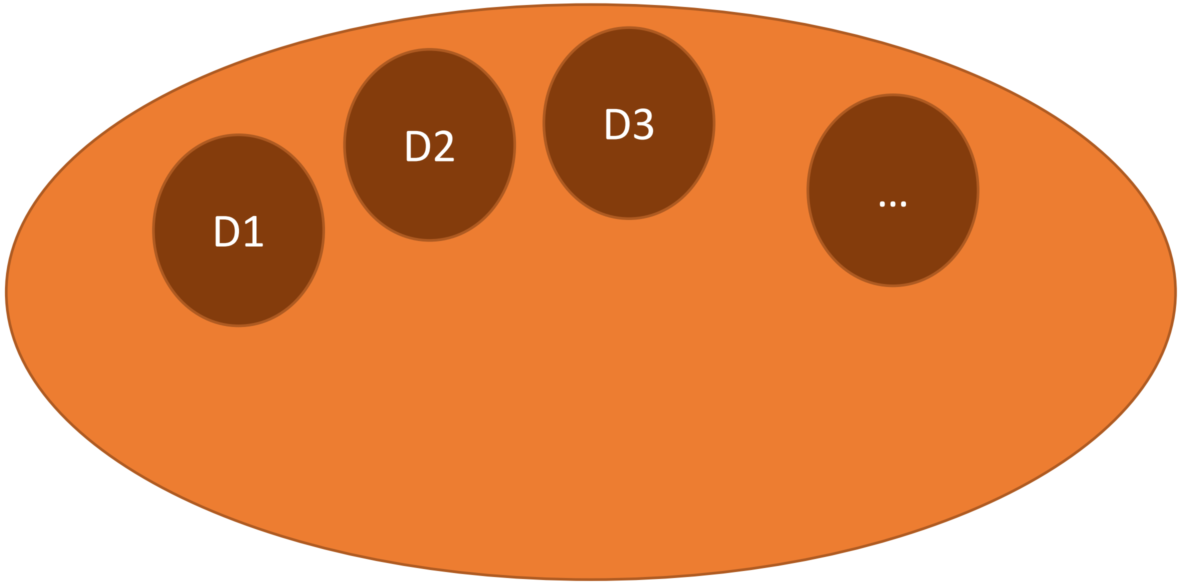
$$\mathcal{L}(Y,\bar{Y}) = \sum_i \left(Y^i - f_w(X^i)\right)^2$$

This depends on the Data.

$$\mathcal{L}_{D_i}(Y,\bar{Y})$$

$$\mathbb{E}_D[\mathcal{L}(Y,\bar{Y})]$$

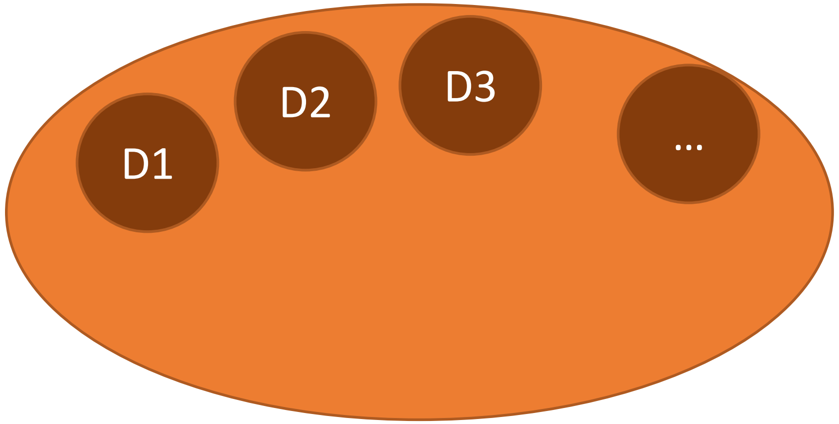
All the possible data



Mehta, Pankaj, et al. "A high-bias, low-variance introduction to machine learning for physicists." Physics Reports (2019).

Bias and Variance: more detail

All the possible data



$$\mathbb{E}_D[\mathcal{L}(Y, \bar{Y})]$$

$$= \sum_i \mathbb{E}_D\left(Y^i - \mathbb{E}_D\left(f_w(X^i)\right) + \mathbb{E}_D\left(f_w(X^i)\right) - f_w(X^i)\right)^2$$

$$= \underbrace{\sum_i \left(Y^i - \mathbb{E}_D\left(f_w(X^i)\right)\right)^2}_{\textbf{Bias^2}} + \underbrace{\sum_i \mathbb{E}_D\left(\mathbb{E}_D\left(f_w(X^i)\right) - f_w(X^i)\right)^2}_{\textbf{Variance}}$$

Mehta, Pankaj, et al. "A high-bias, low-variance introduction to machine learning for physicists." Physics Reports (2019).

For a good model

Training Data → In-sample error

Test Data → Out-sample error

$$\mathbb{E}_D[\mathcal{L}(Y,\bar{Y})]$$

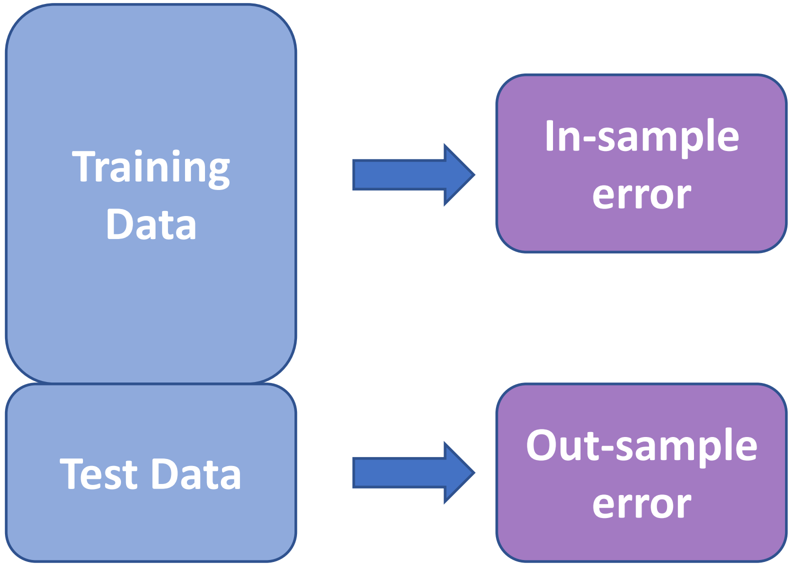
$$= \underbrace{\sum_i \left(Y^i - \mathbb{E}_D\left(f_w(X^i)\right)\right)^2}_{\text{Bias^2}} + \underbrace{\sum_i \mathbb{E}_D\left(\mathbb{E}_D\left(f_w(X^i)\right) - f_w(X^i)\right)^2}_{\text{Variance}}$$

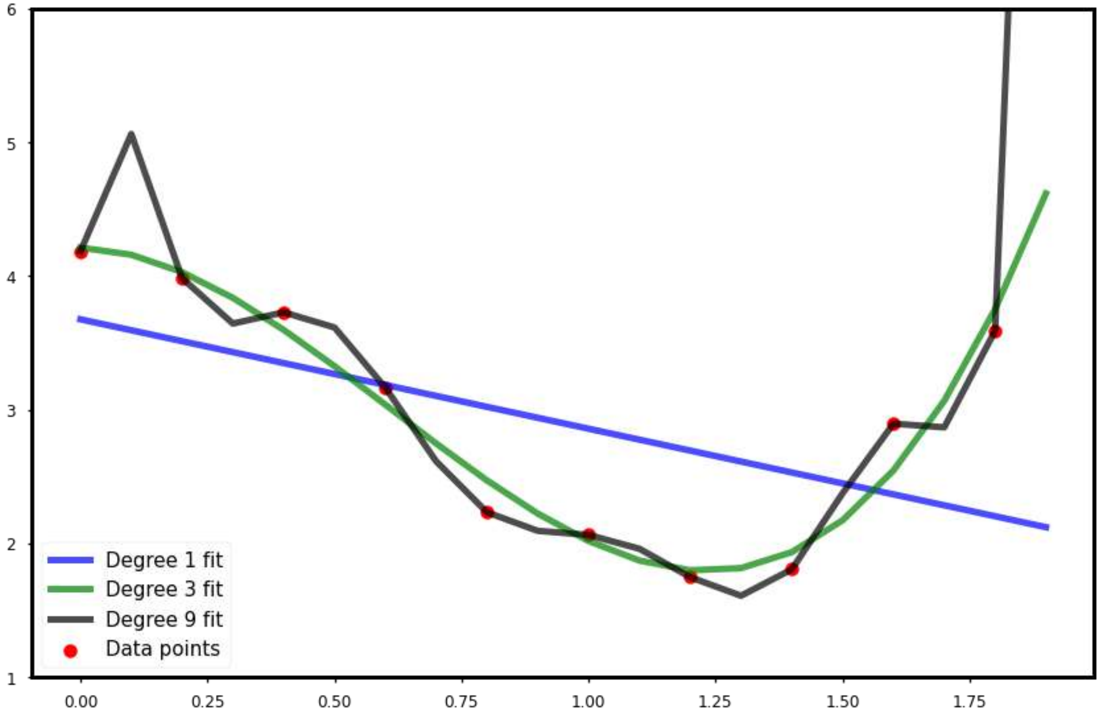
Bias^2 → Low

Variance → Low

How can we reduce bias?

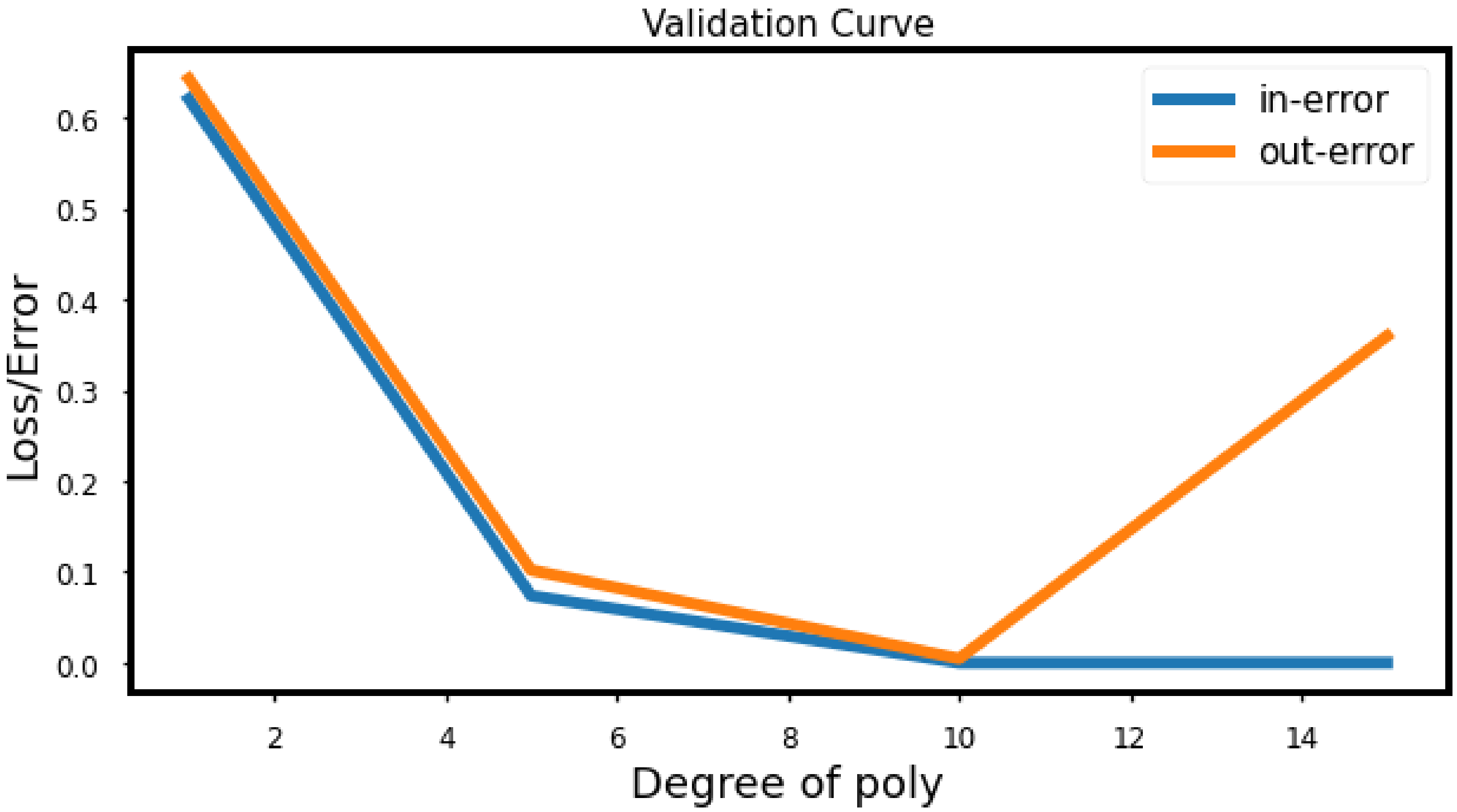
How can we reduce variance?



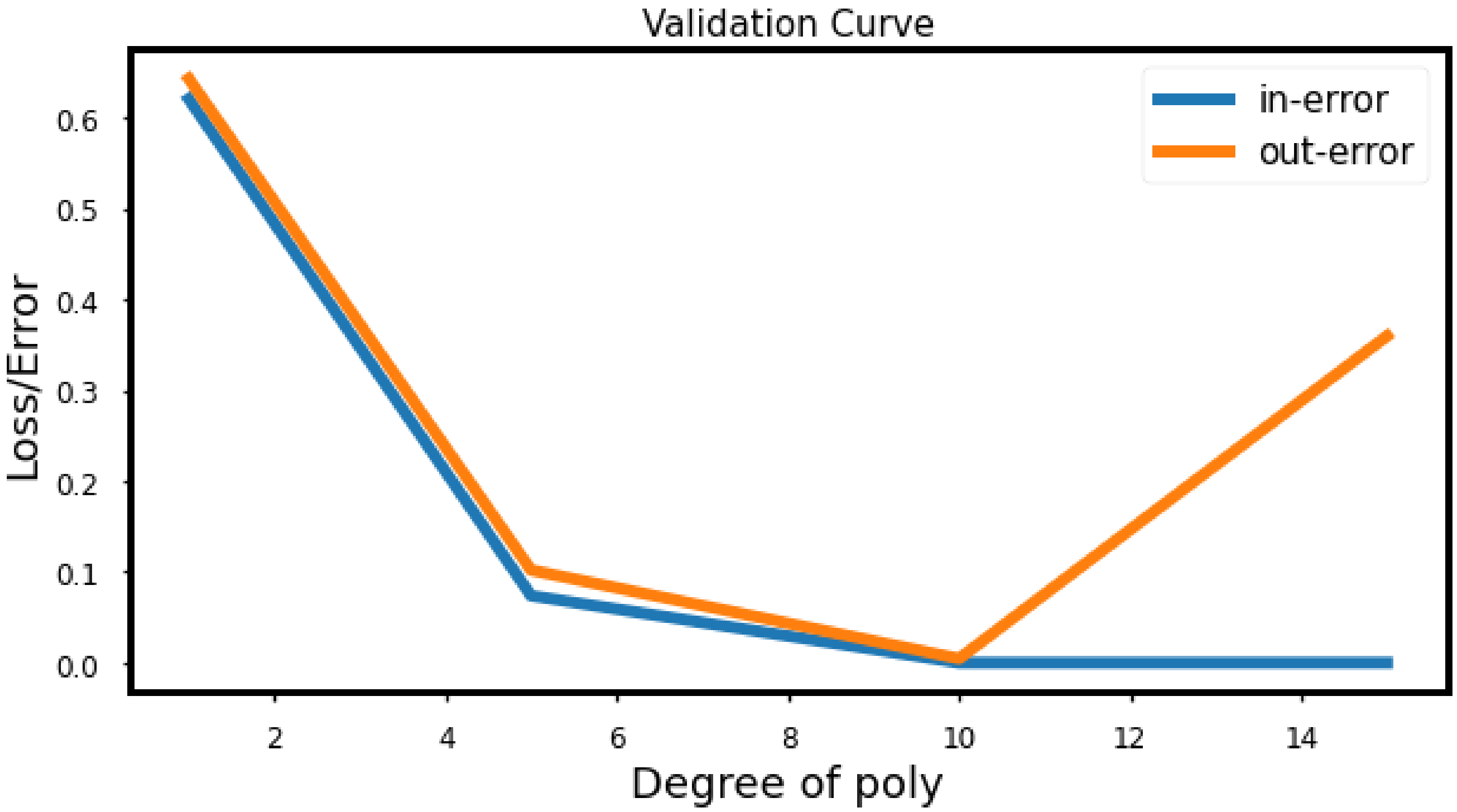


Check the code!

Validation curve

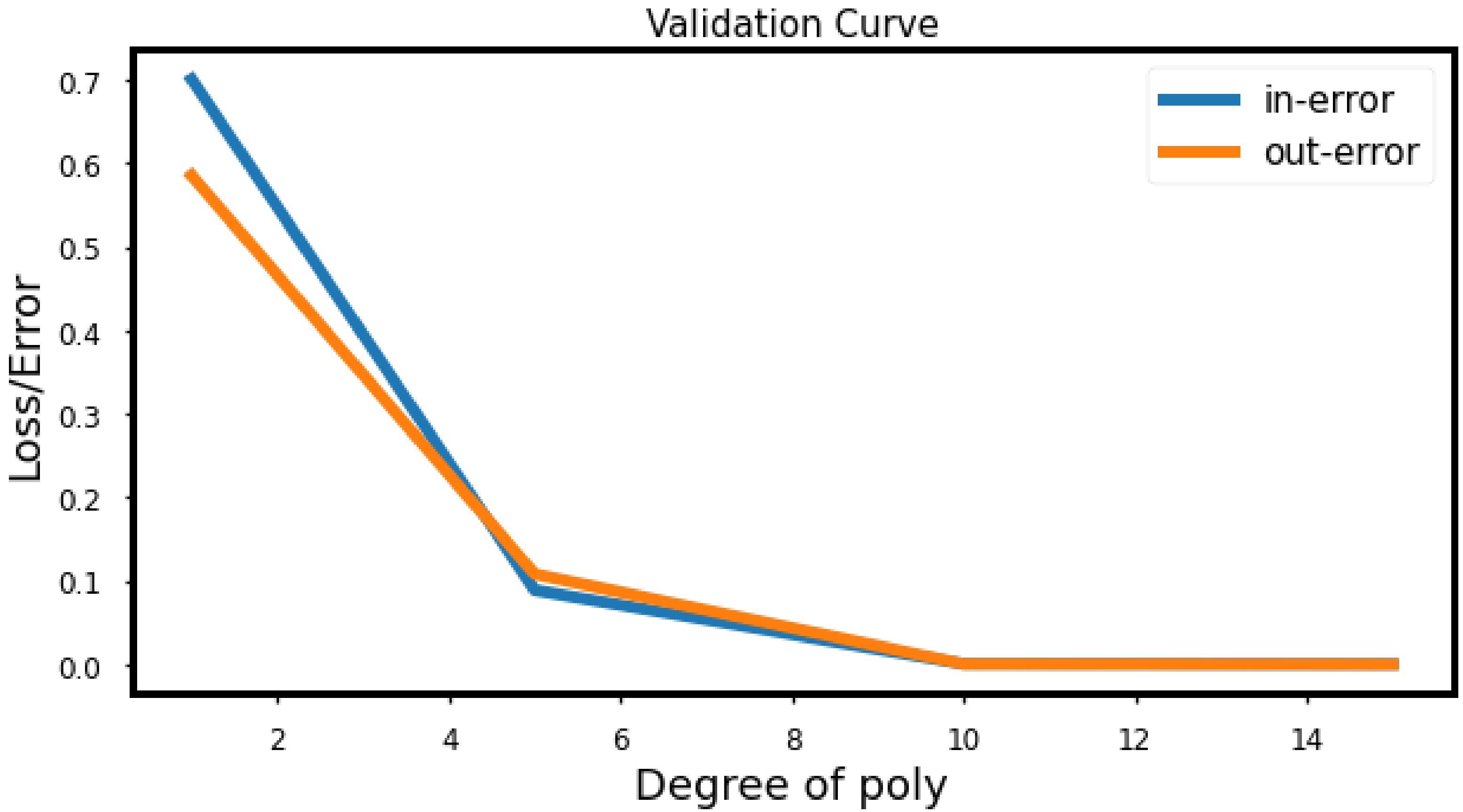


How much complexity?

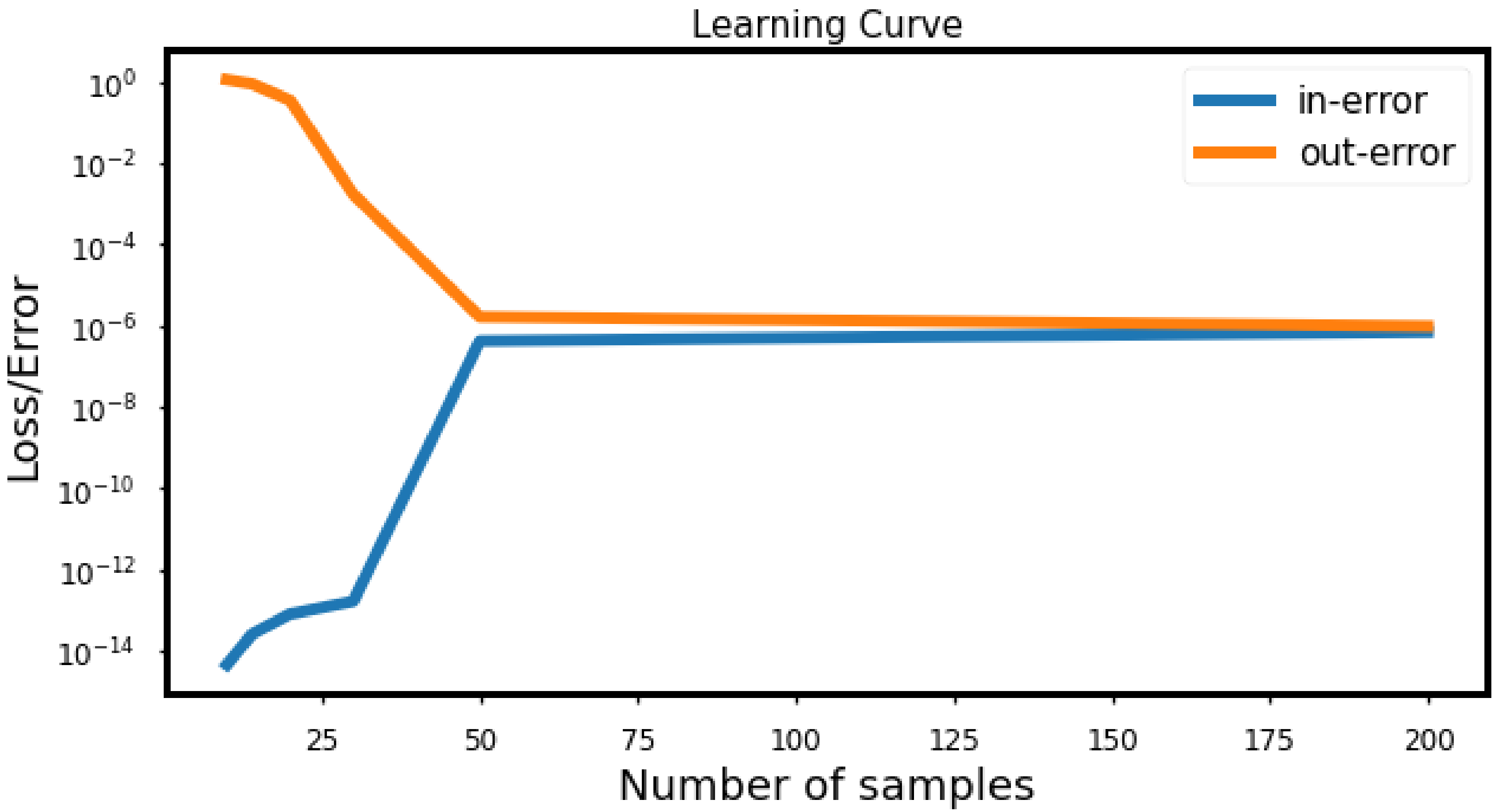


How much data?

Do we have enough data?



Learning curve



Regularization

$$\mathcal{L}(Y,\bar{Y}) = \sum_i \left(Y^i - f_w(X^i)\right)^2 \quad \boldsymbol{+\alpha\|w\|_l}$$

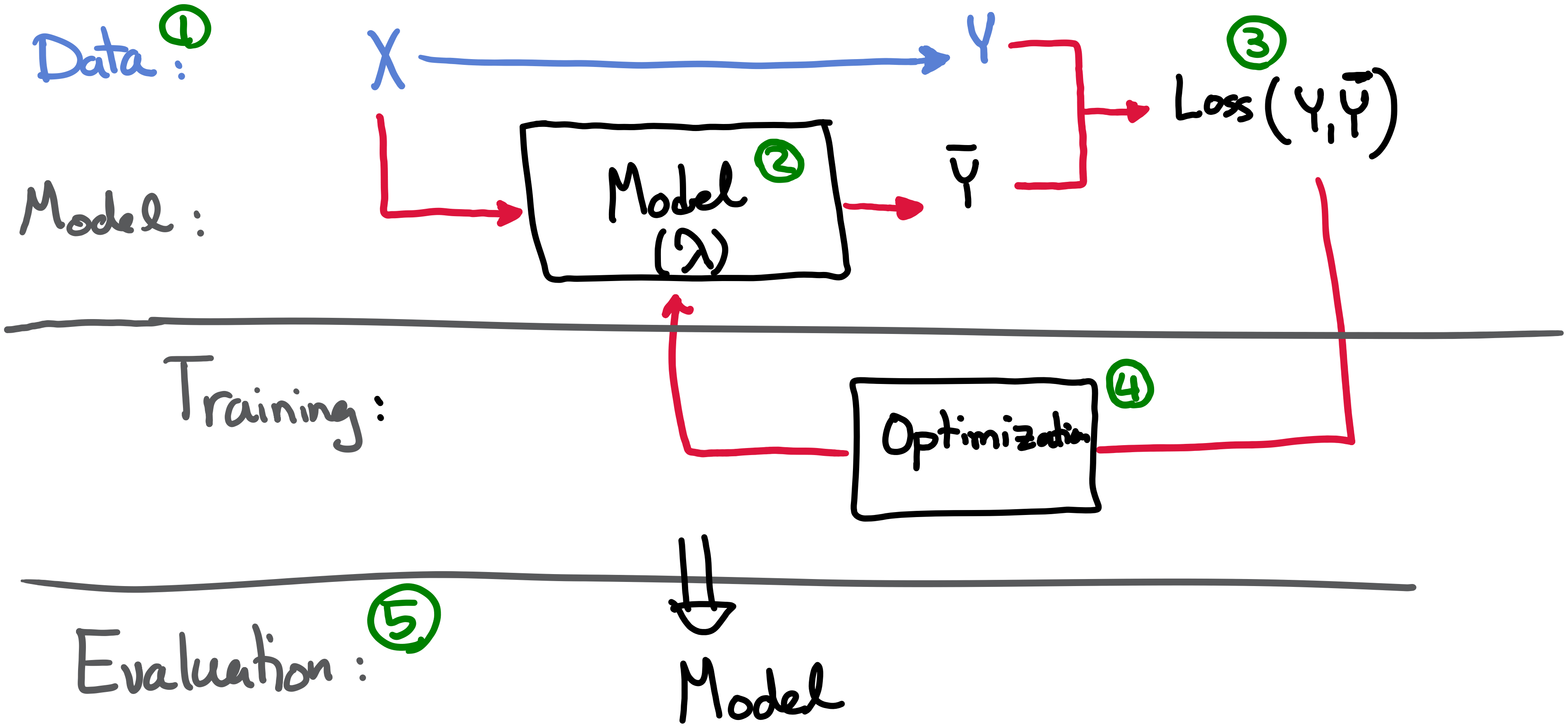
L2: $\|\boldsymbol{w}\|_2 = \sum_i \boldsymbol{w}_i^2$

L1: $\|\boldsymbol{w}\|_1 = \sum_i |\boldsymbol{w}_i|$

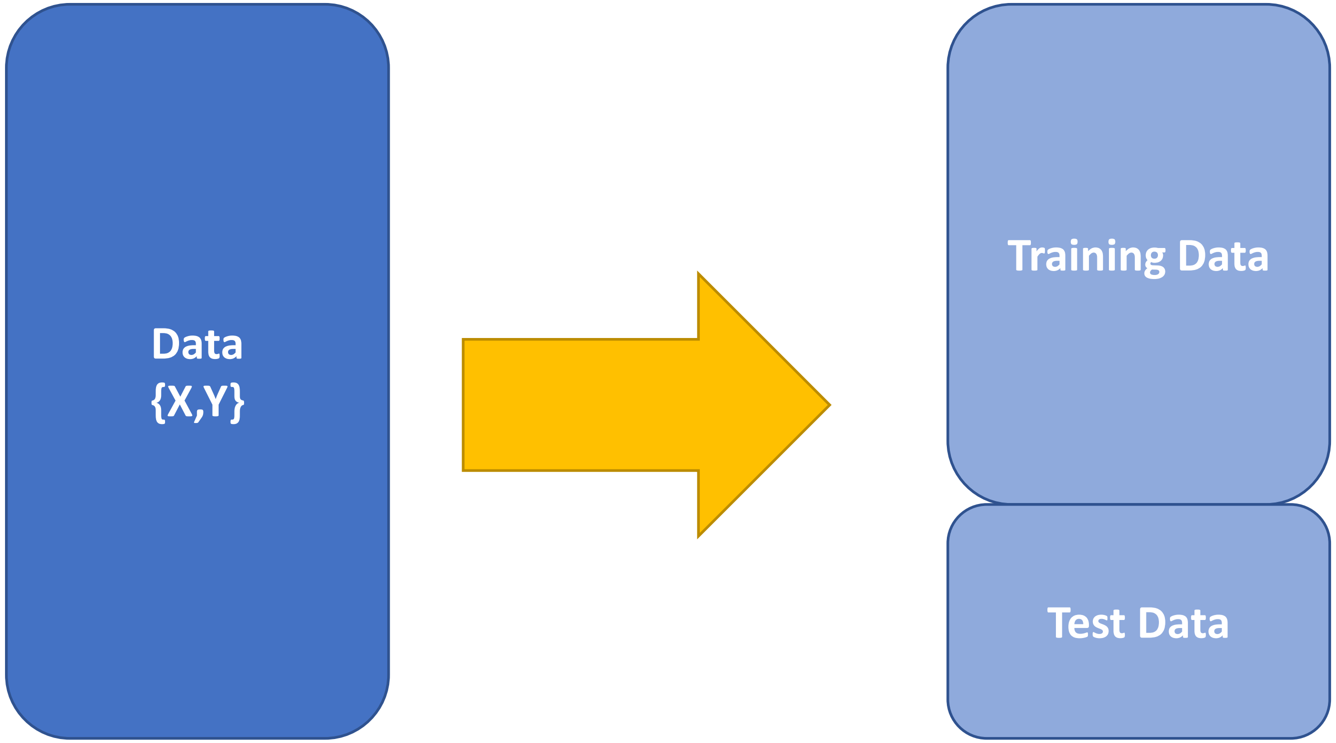
Metrics

Recap

Supervised: Ingredients

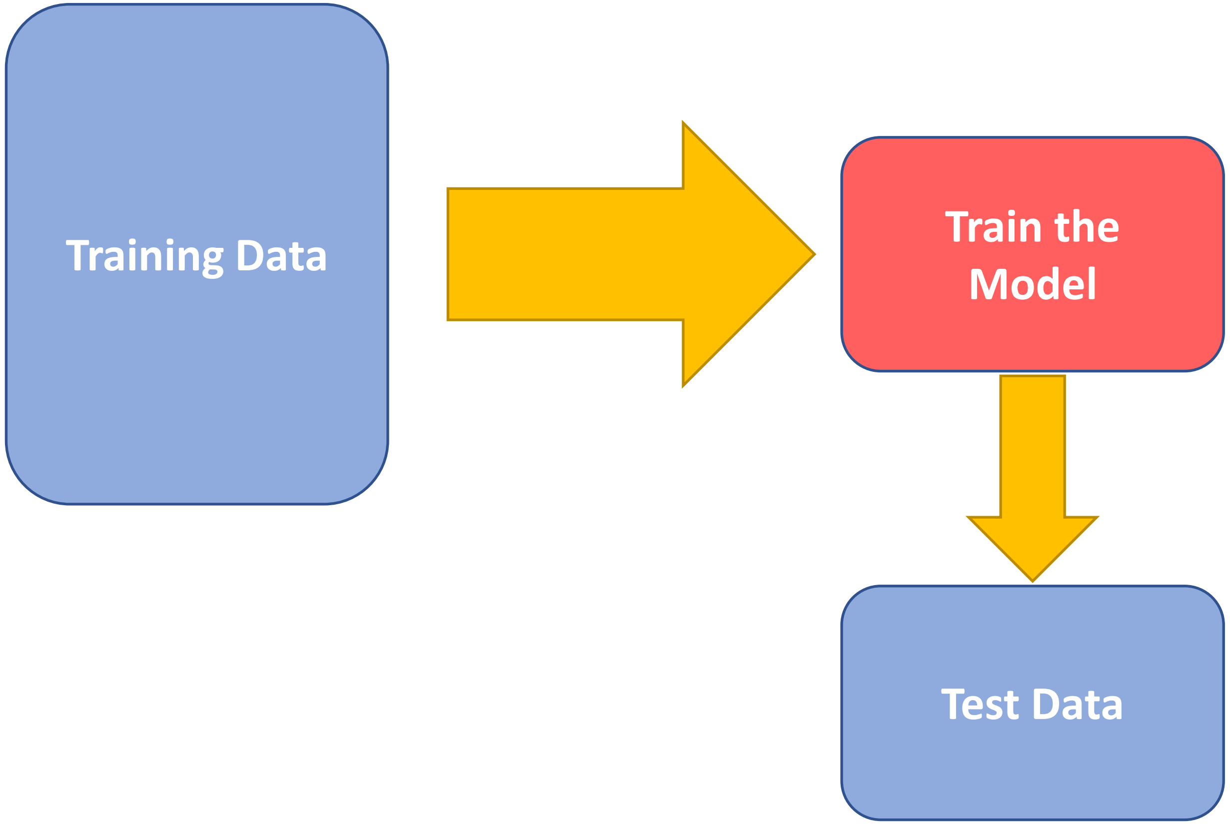


Code




```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X , Y, random_state=0)
```

Code




```
from sklearn.linear_model import SGDClassifier

clf = SGDClassifier()
clf.fit(X_train, Y_train)
```

```
y_predict = clf.predict(X_test)
error = np.abs(Y_test - y_predict).sum() / len(Y_test)
```

Code: full pipeline

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X , Y, random_state=0)
```

```
from sklearn.linear_model import SGDClassifier
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler, PolynomialFeatures

### Training the model
clf_pipeline= Pipeline([('scaler', StandardScaler() ),
                        ('p_transformer', PolynomialFeatures(degree = 3)),
                        ('clf', SGDClassifier())])
clf_pipeline.fit(X_train,Y_train)
```

```
### Testing the model
y_predict = clf_pipeline.predict(X_test)
out_error = np.abs(Y_test - y_predict).sum() / len(Y_test)
in_error = np.abs(Y_train - clf_pipeline.predict(X_train)  ).sum() / len(Y_train)
```