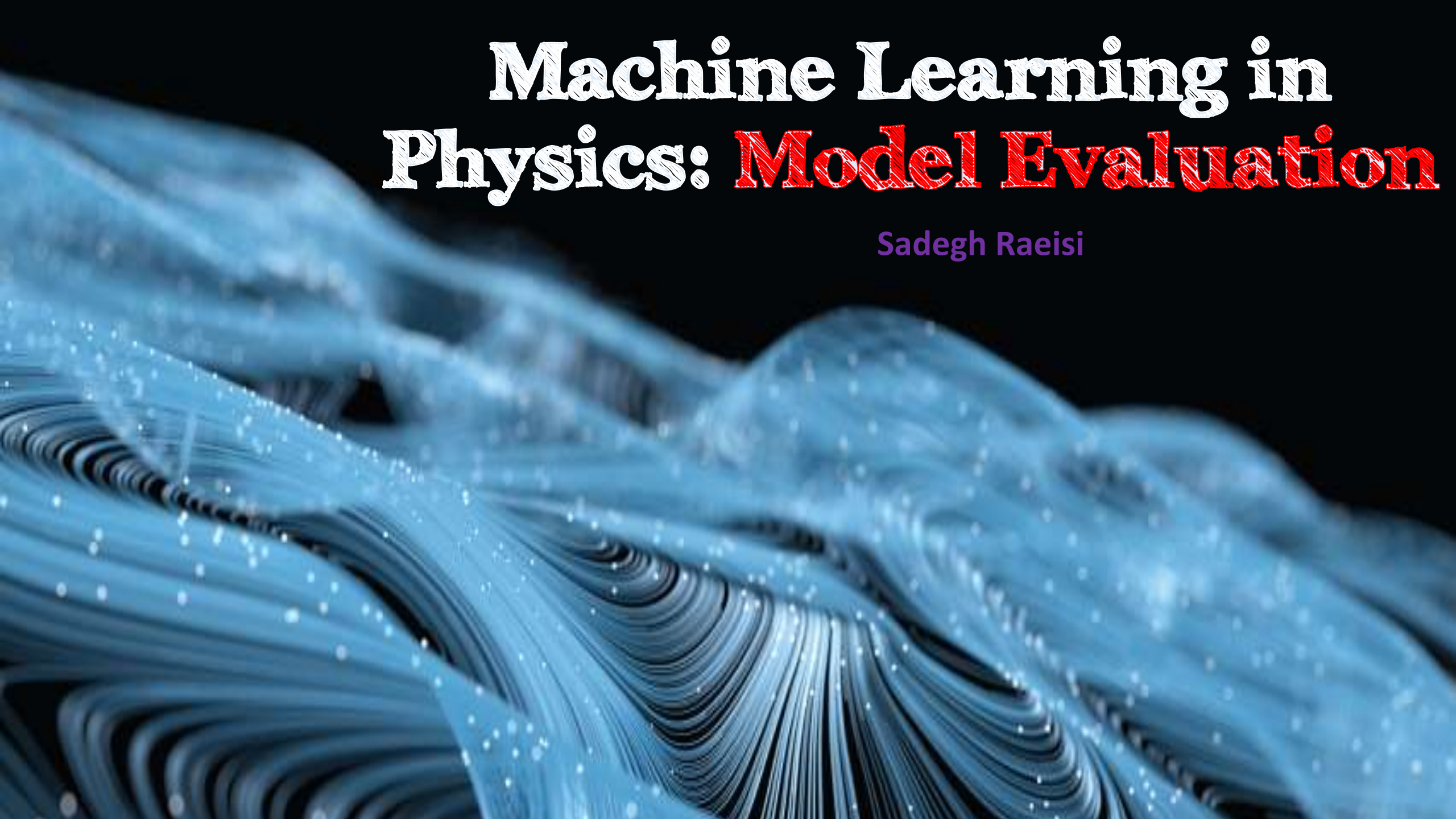
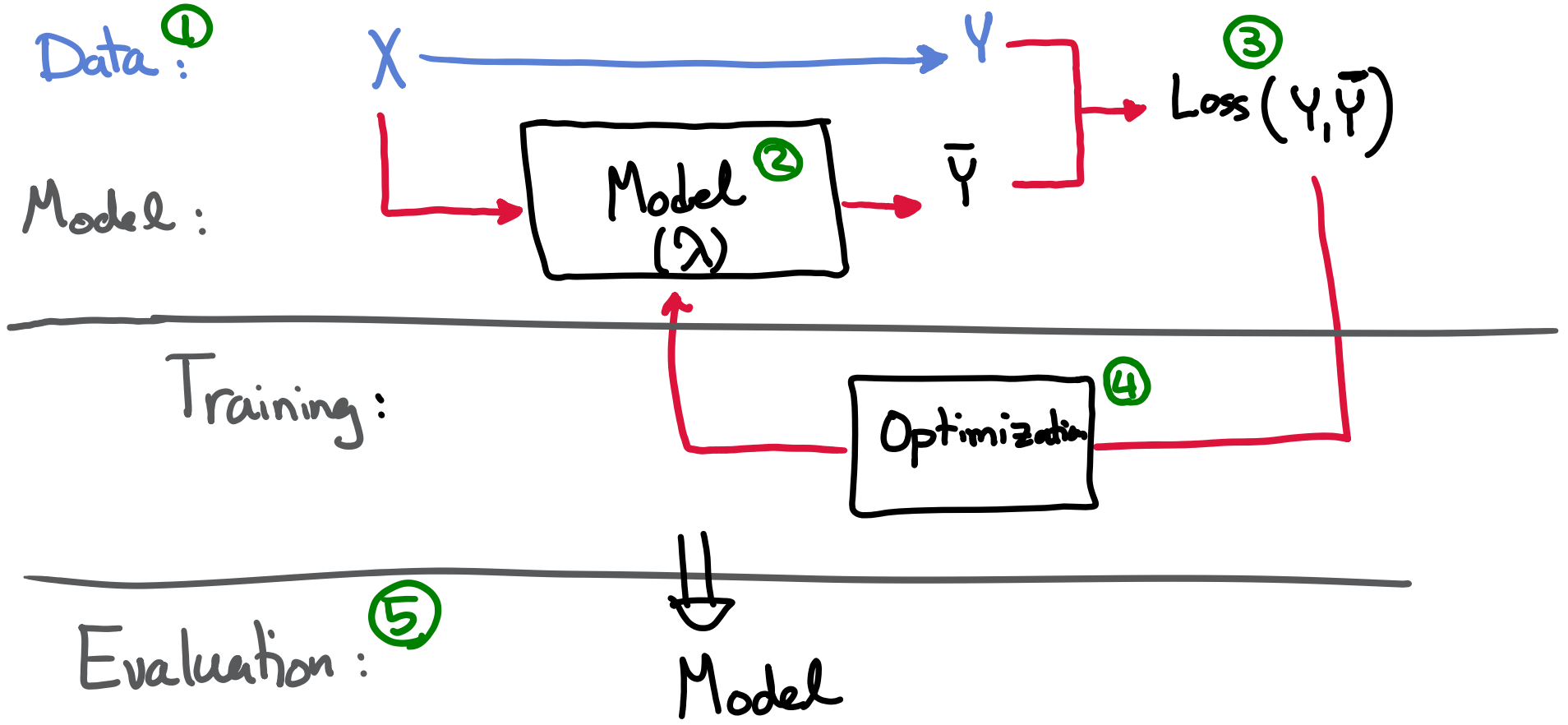
Machine Learning in Physics: Model Evaluation

Sadegh Raeisi

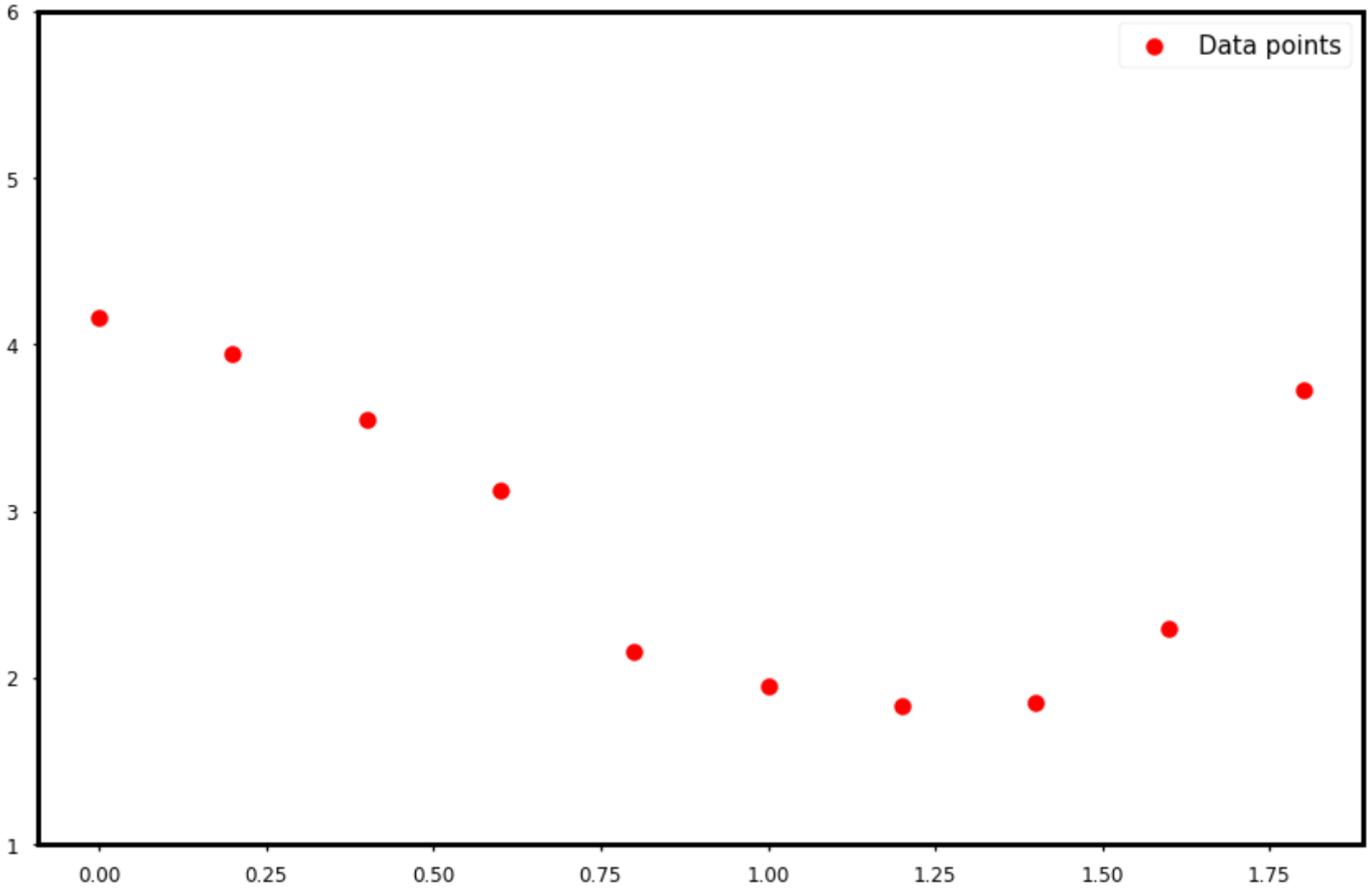


Supervised: Ingredients

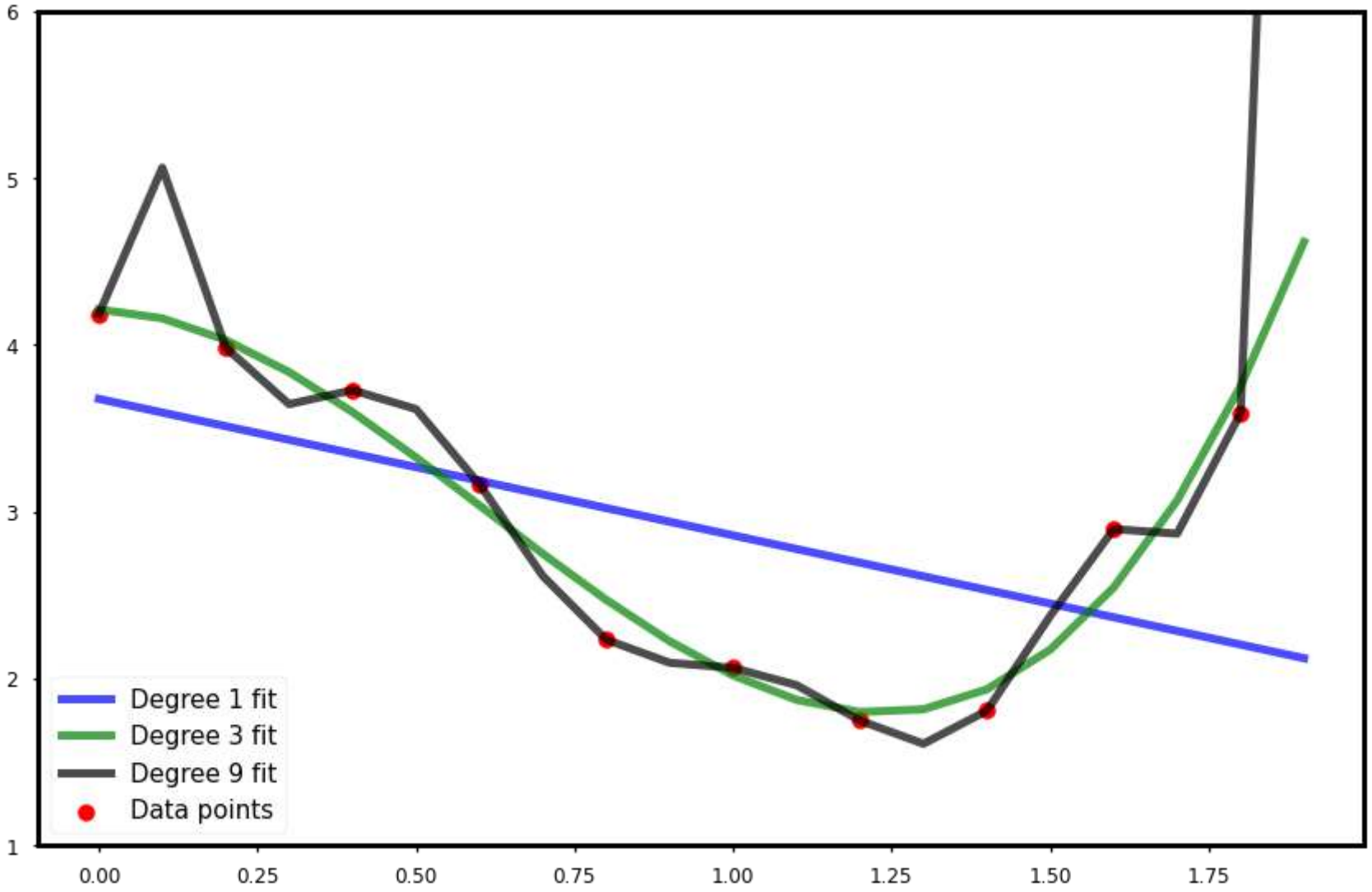


A good model

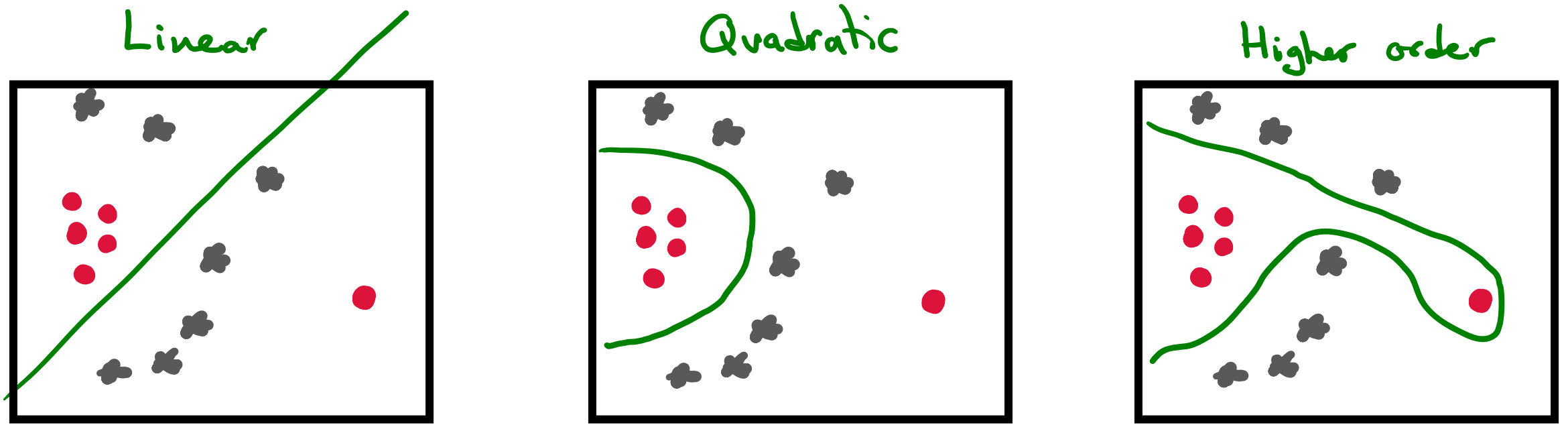
A good fit vs a good model



A good fit vs a good model

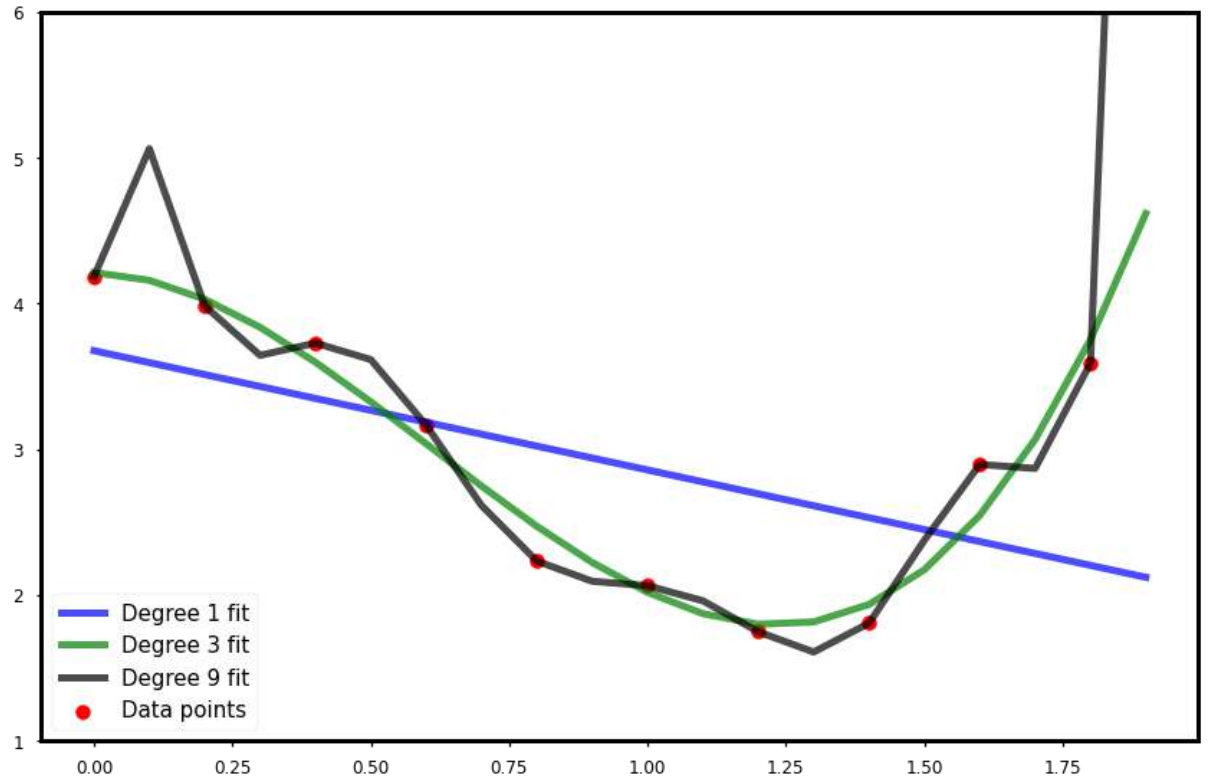


A good fit vs a good model

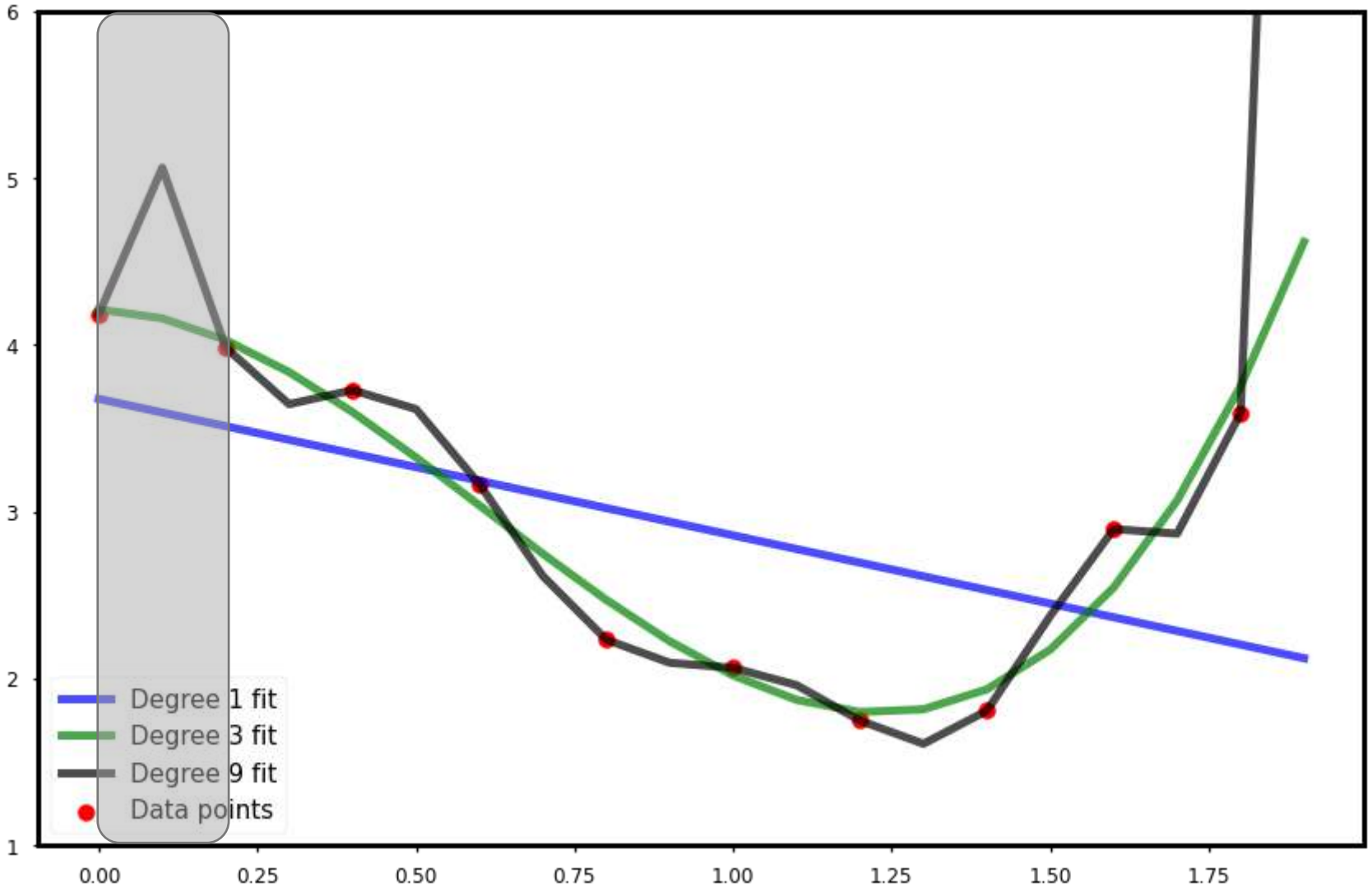


What’s the problem?

How can we solve it?

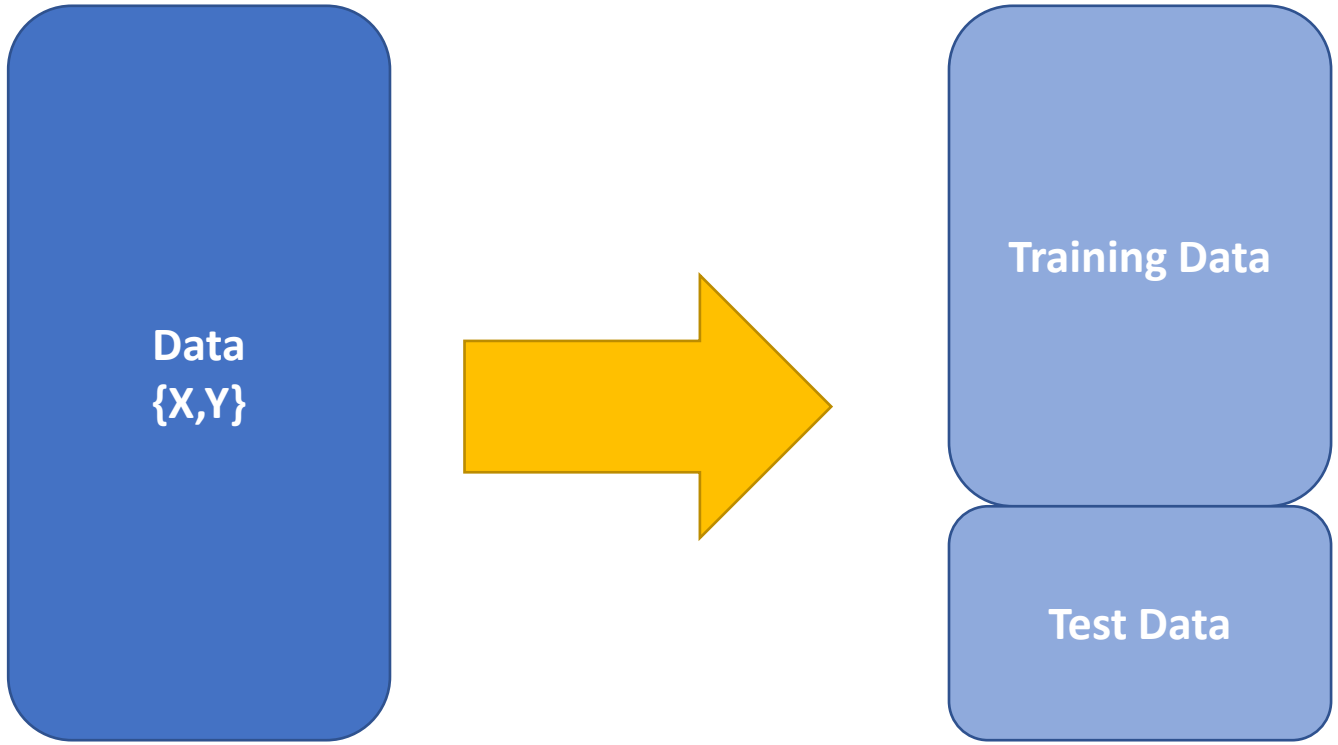


Good fit vs Good Prediction



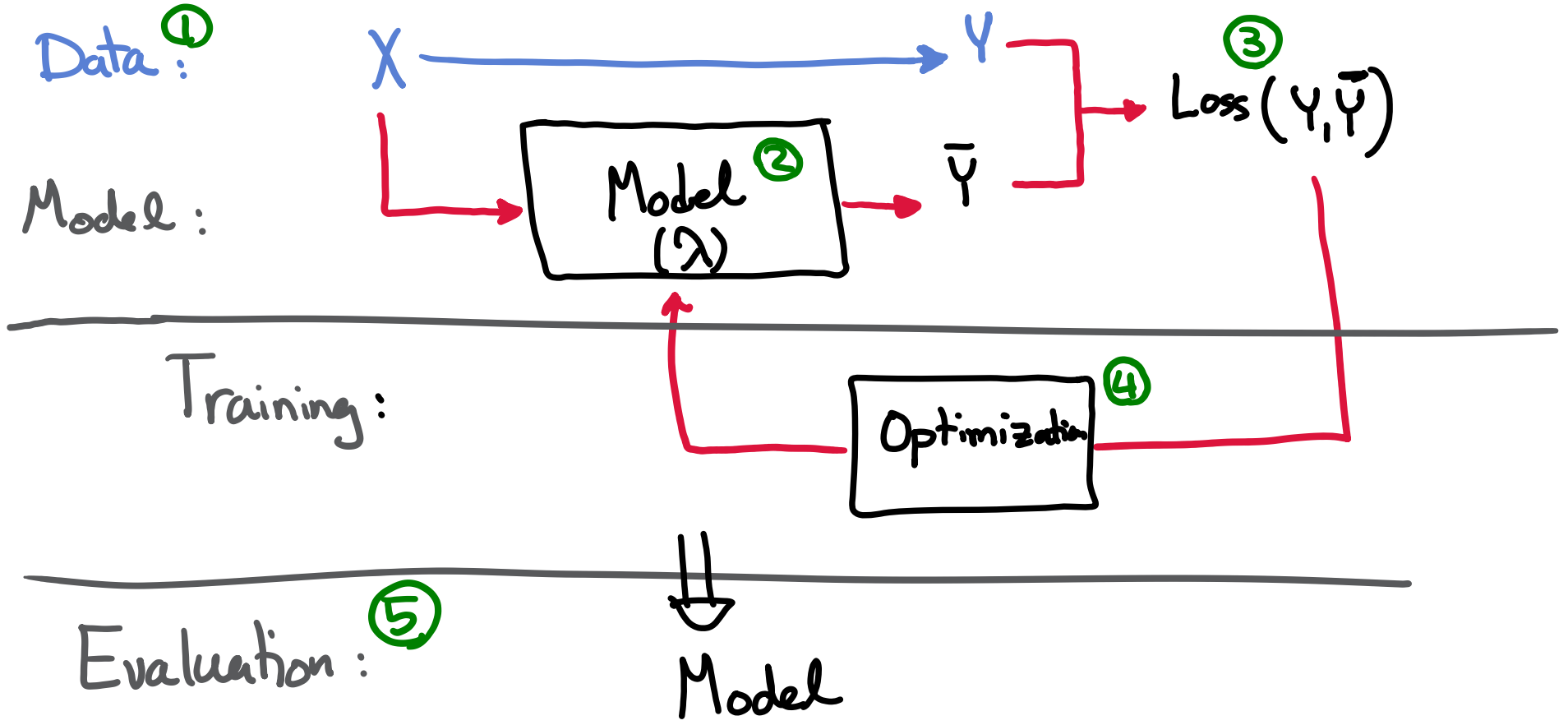
How can check the prediction power of a model?

- In-sample vs out sample error

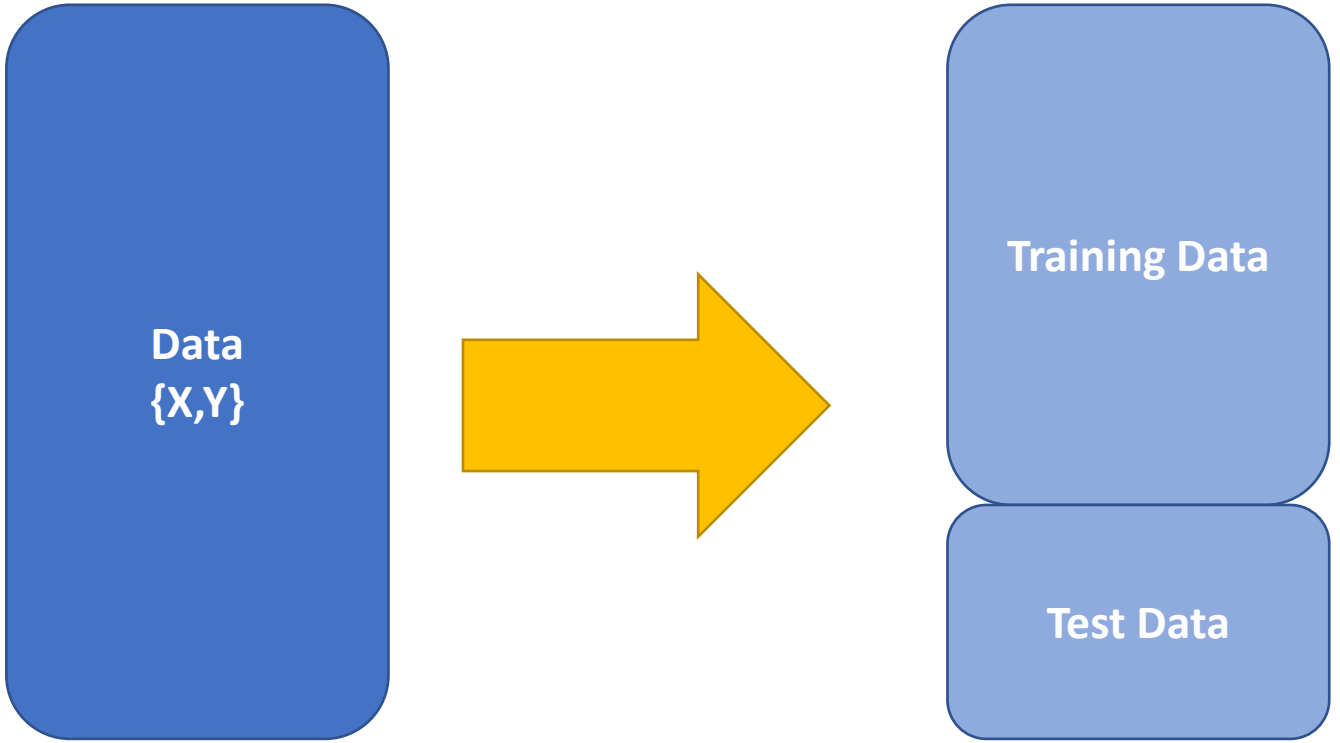


Recap

Supervised: Ingredients

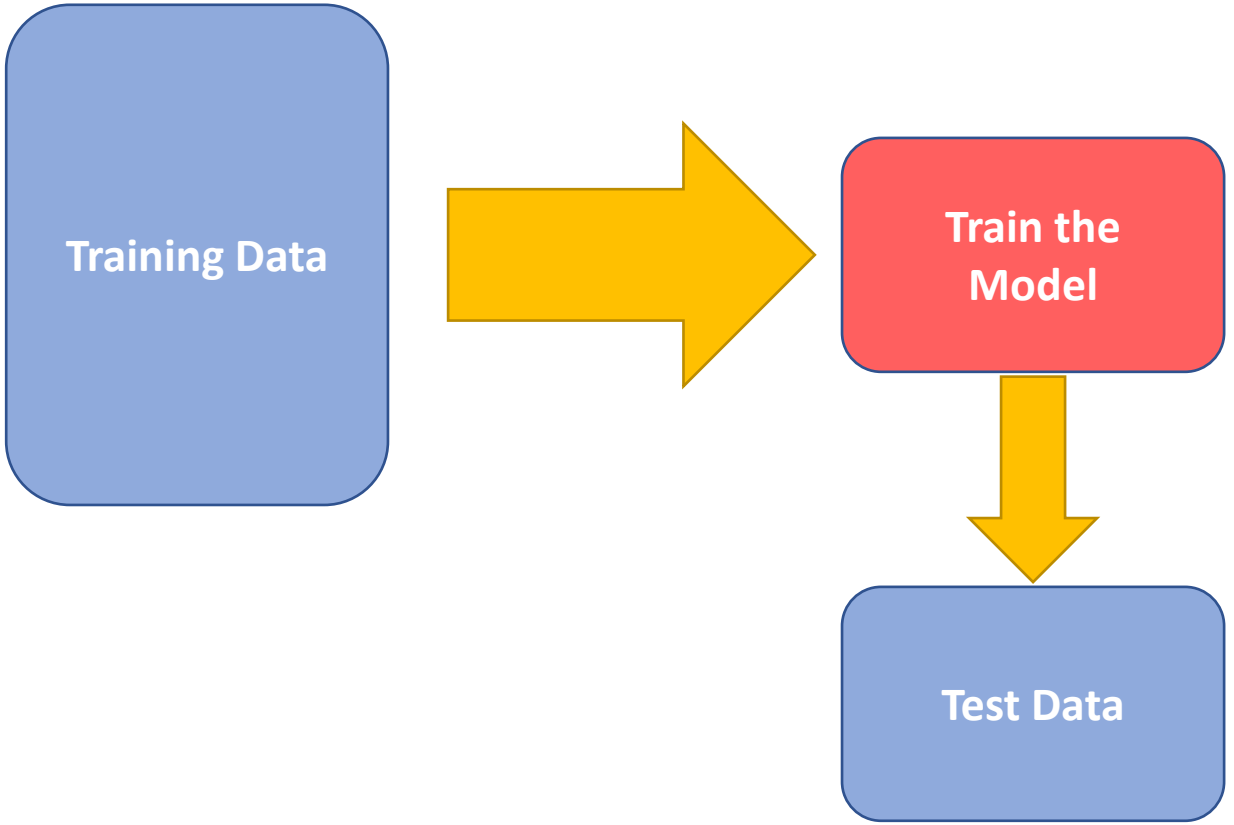


Code




```
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)
```