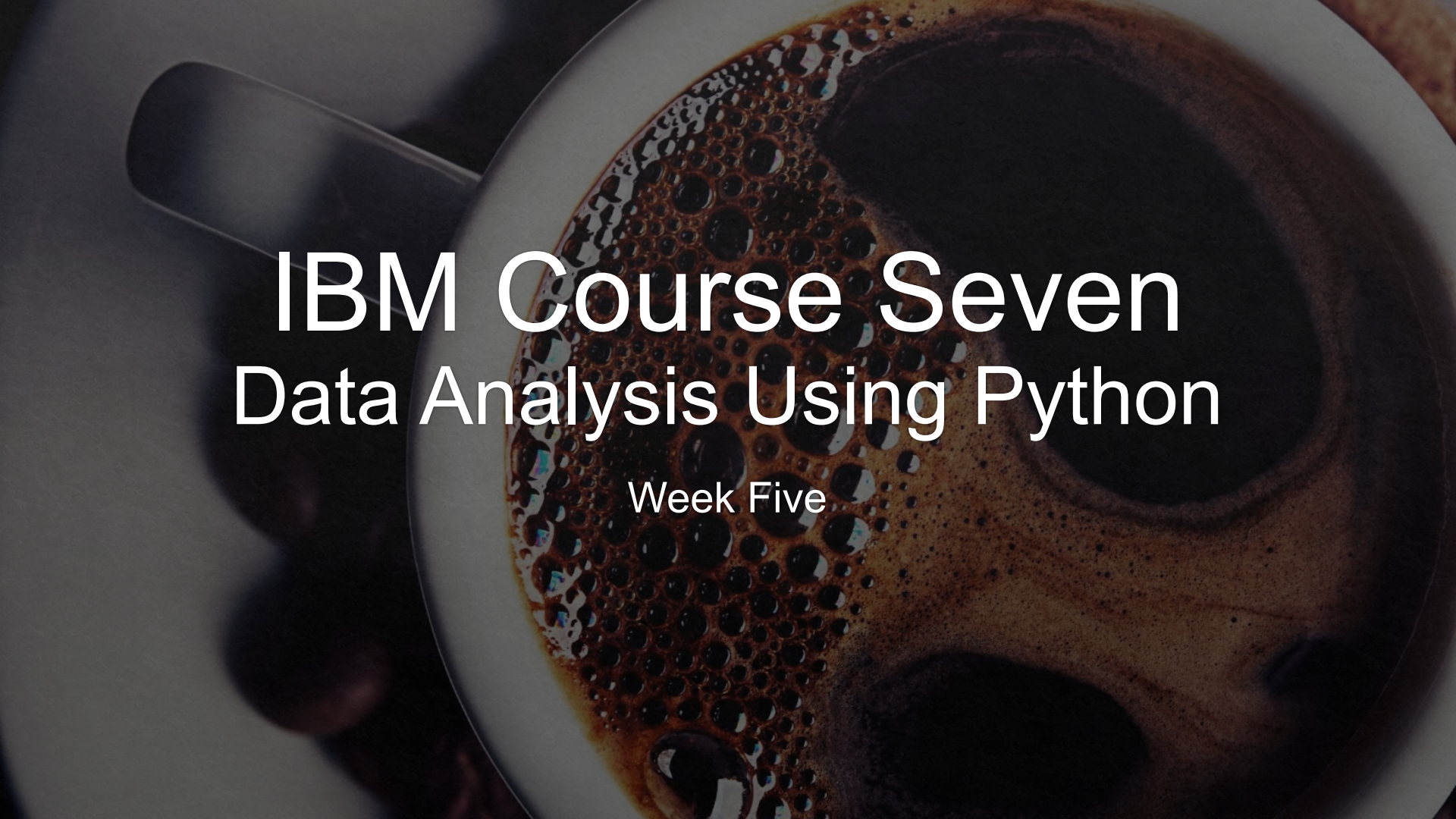


IBM Course Seven
Data Analysis Using Python

Week Five

Objectives

Demonstrate the process of model refinement

Demonstrate the process of ridge regression

Demonstrate the use of Grid search in Python

Demonstrate the process of model evaluation in Python

Demonstrate the handling of fitting, underfitting, overfitting and model selection in Python

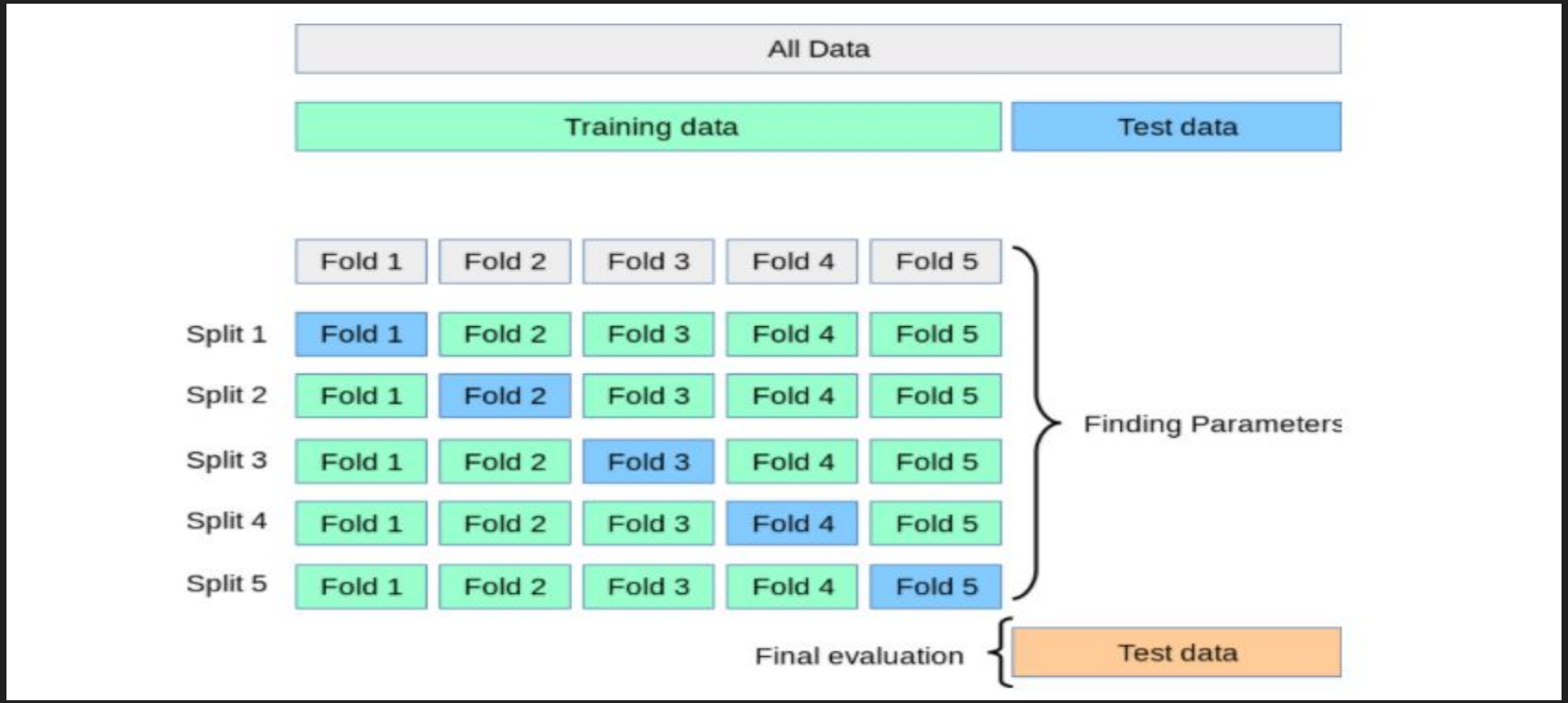
What is test data, train data and validation data?

Training Dataset: The sample of data used to fit the model.

Validation Dataset: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.

Test Dataset: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

Split data


All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Split 2
Split 3
Split 4
Split 5
Finding Parameters
Final evaluation
Test data

train_test_split()

X_train,x_test,y_train,y_test = train_test_split(x_data, y_data, test_size = 0.3, randome_state =0)

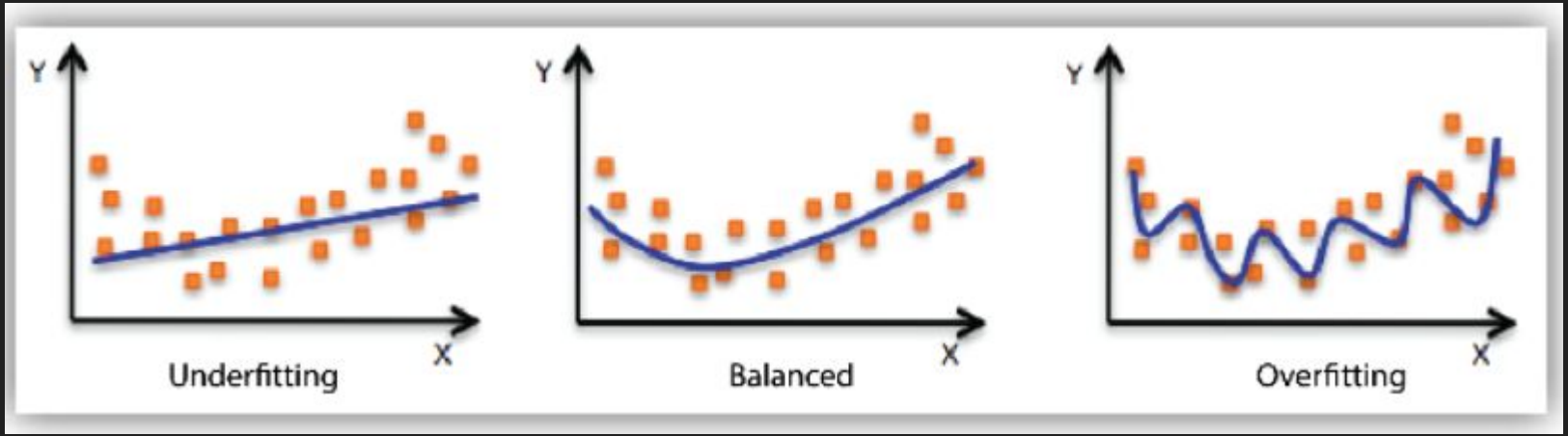
Cross Validation

```
Scores = cross_val_score(lr, x_data, y_data, cv =3)

np.mean(scores)

Yhat = cross_val_predict(lr2e, x_data, y_data, cv = 3)
```

Overfitting and Underfitting



Underfitting

Causes

Trying to create a linear model with non linear data.

Having too little data to build an accurate model

Model is too simple, has too few features

Remedies

Add more features during Feature Selection.

Engineer additional features within the scope of your problem that makes sense.

Overfitting

Causes

The primary cause of models being overfit is that the algorithm captured the “noise” of the data.

Overfitting occurs when the model fits the data too well.

An overfit model shows low bias and high variance.

The model is excessively complicated likely due to redundant features.

Remedies

K-fold cross validation (refer here)

Train with more data

Remove features

Ridge Regression

Prevents Overfitting

From sklearn.linear-model import Ridge

RidgeModel = Ridge(alpha=0.1)

RidgeModel.fit(X,Y)

Yhat = RidgeModel.predict(Y)

Grid Search

Data broken into 3 sets

Training

Validation

Test

```
Parameters =[{‘alpha’ : [1, 10, 100, 1000]}]

From sklearn.linear-model import Ridge

From sklearn.model-selection import GridSearchCV

Parameters1 = [{‘alpha’:[0.001,0.1,1,10]}]

RR=Ridge()

Grid1 = GridSearchCV(RR, parameters1, cv = 4)

Grid1.fit(x_data[[‘Variable 1’ , ‘Variable 2’]], y_data)

Grid1. Best_estimator_

Scores = Grid1.CV_results_

Scores[‘mean_test_score’]
```