Understand the problem

WINNING A KAGGLE COMPETITION IN PYTHON

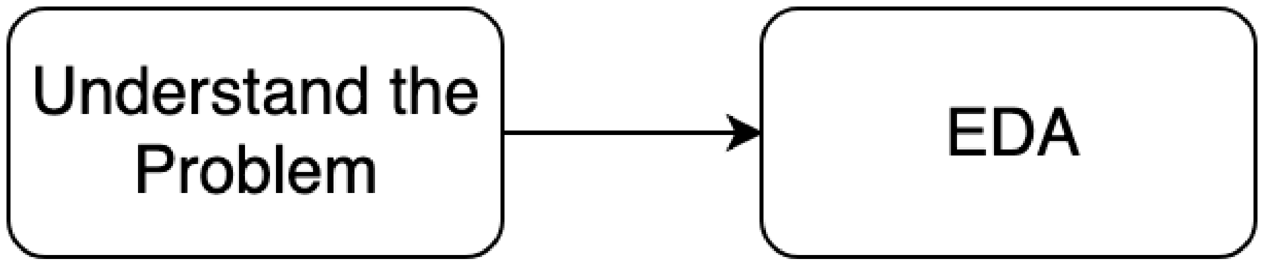
Yauhen Babakhin
Kaggle Grandmaster

DataCamp

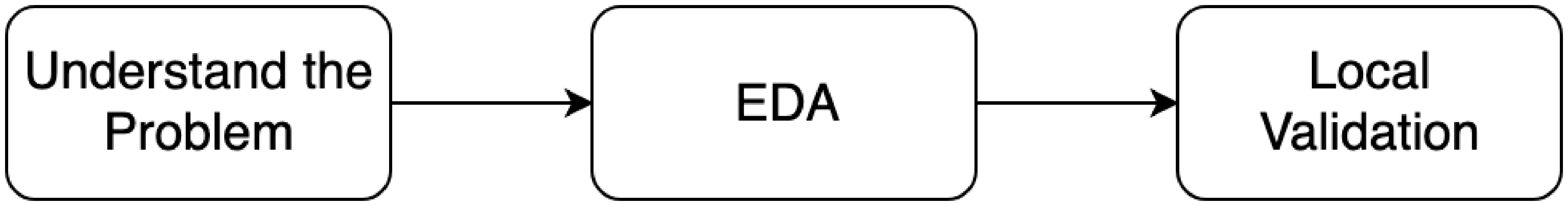
Solution workflow

Understand the Problem

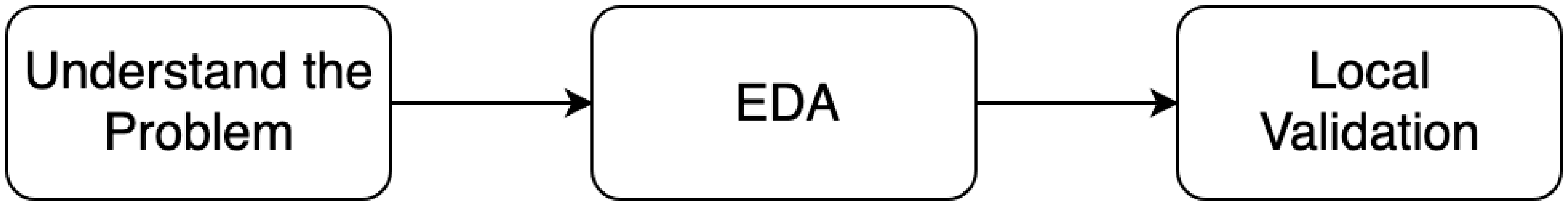
Solution workflow



Solution workflow



Solution workflow



Understand the problem

- **Data type:** tabular data, time series, images, text, etc.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0

Understand the problem

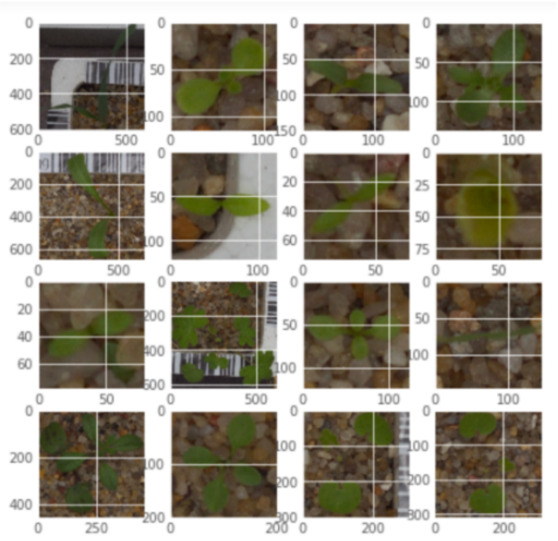
- **Data type:** tabular data, time series, images, text, etc.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0

Understand the problem

- **Data type:** tabular data, time series, images, text, etc.

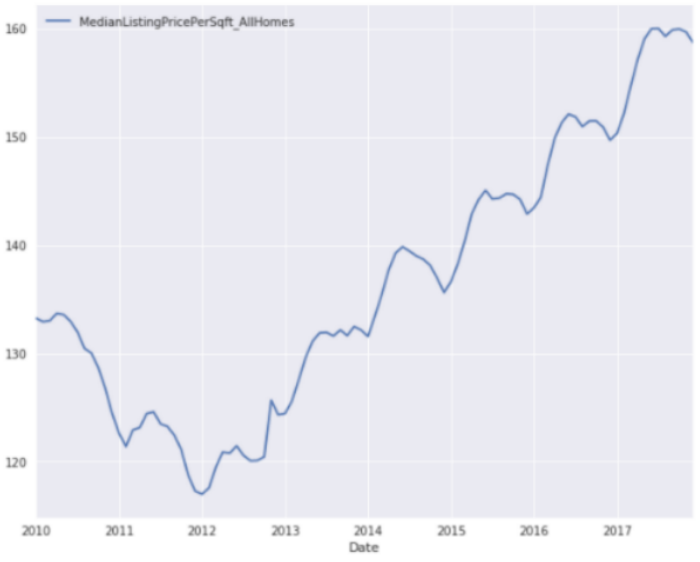
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0

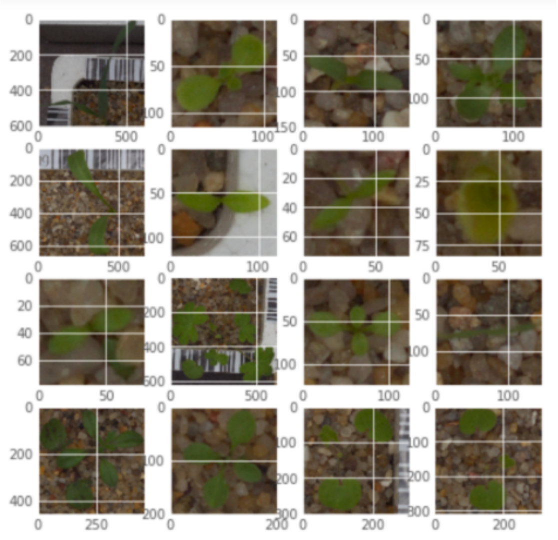


Understand the problem

- **Data type:** tabular data, time series, images, text, etc.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0





- **Problem type:** classification, regression, ranking, etc.
- **Evaluation metric:** ROC AUC, F1-Score, MAE, MSE, etc.

Metric definition

```
# Some classification and regression metrics
from sklearn.metrics import roc_auc_score, f1_score, mean_squared_error
```

$$RMSLE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\log(y_i + 1) - \log(\hat{y}_i + 1))^2}$$

```
import numpy as np

def rmsle(y_true, y_pred):
    diffs = np.log(y_true + 1) - np.log(y_pred + 1)
    squares = np.power(diffs, 2)
    err = np.sqrt(np.mean(squares))
    return err
```

Let's practice!

WINNING A KAGGLE COMPETITION IN PYTHON

DataCamp

Initial EDA

WINNING A KAGGLE COMPETITION IN PYTHON

Yauhen Babakhin
Kaggle Grandmaster

DataCamp

Goals of EDA

- Size of the data
- Properties of the target variable
- Properties of the features
- Generate ideas for feature engineering

Two sigma connect: rental listing inquiries

Problem statement

Predict the popularity of an apartment rental listing

Target variable

interest_level

Problem type

Classification with 3 classes: 'high', 'medium' and 'low'

Metric

Multi-class logarithmic loss

EDA. Part I

```
# Size of the data
twosigma_train = pd.read_csv('twosigma_train.csv')
print('Train shape:', twosigma_train.shape)

twosigma_test = pd.read_csv('twosigma_test.csv')
print('Test shape:', twosigma_test.shape)
```

```
Train shape: (49352, 11)
Test shape: (74659, 10)
```

EDA. Part I

```
print(twosigma_train.columns.tolist())
```

```
['id', 'bathrooms', 'bedrooms', 'building_id', 'latitude', 'longitude',
'manager_id', 'price', 'interest_level']
```

```
twosigma_train.interest_level.value_counts()
```

```
low       34284
medium    11229
high       3839
```

EDA. Part I

```
# Describe the train data
twosigma_train.describe()
```

```
        bathrooms      bedrooms      latitude     longitude         price
count 49352.00000  49352.000000  49352.000000  49352.000000  4.935200e+04
mean      1.21218      1.541640     40.741545    -73.955716  3.830174e+03
std       0.50142      1.115018      0.638535      1.177912  2.206687e+04
min       0.00000      0.000000      0.000000   -118.271000  4.300000e+01
25%       1.00000      1.000000     40.728300    -73.991700  2.500000e+03
50%       1.00000      1.000000     40.751800    -73.977900  3.150000e+03
75%       1.00000      2.000000     40.774300    -73.954800  4.100000e+03
max      10.00000      8.000000     44.883500      0.000000  4.490000e+06
```

EDA. Part II

```python
import matplotlib.pyplot as plt
plt.style.use('ggplot')
```

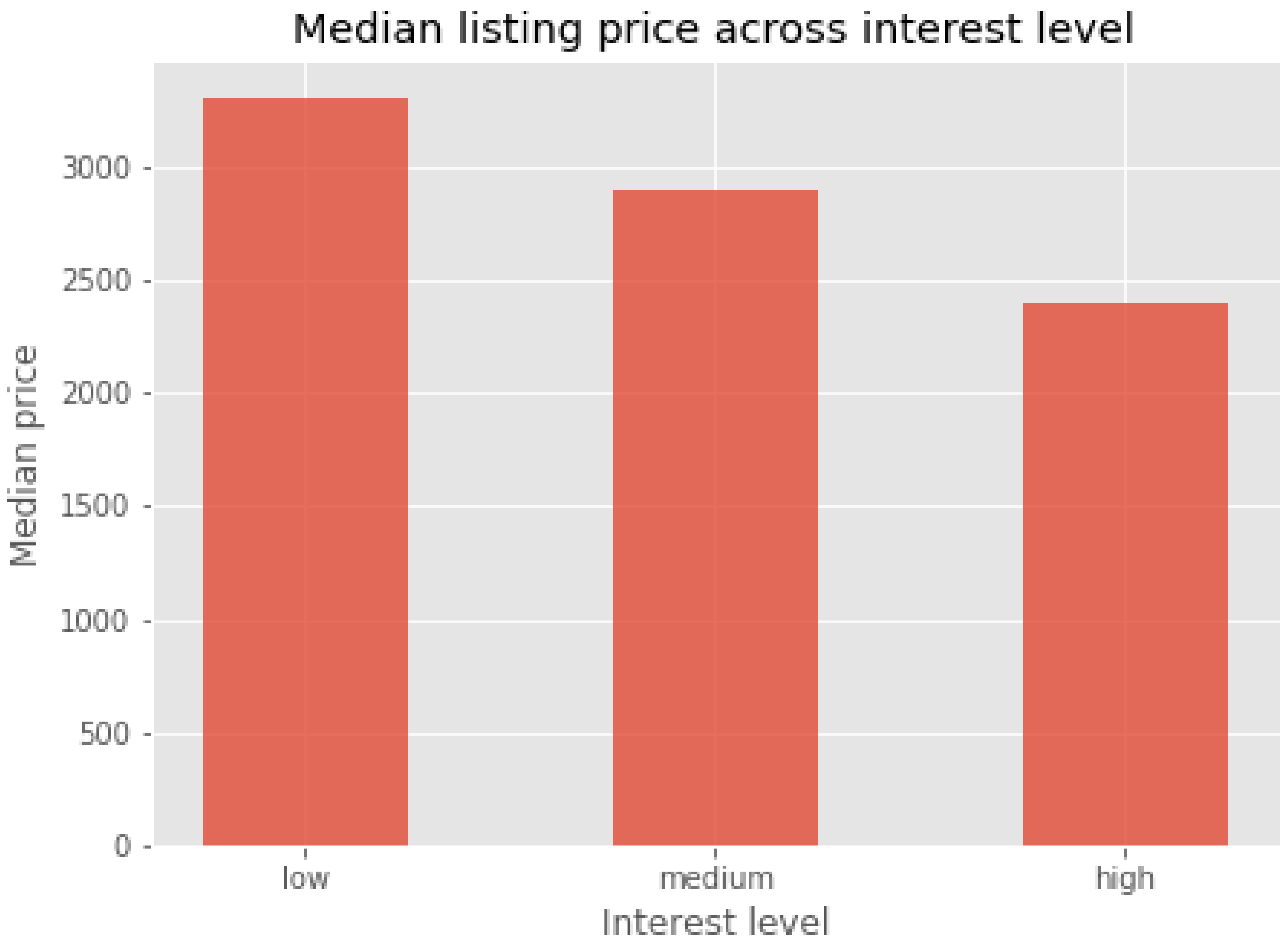
```python
# Find the median price by the interest level
prices = twosigma_train.groupby('interest_level', as_index=False)['price'].median()
```

EDA. Part II

```
# Draw a barplot
fig = plt.figure(figsize=(7, 5))
plt.bar(prices.interest_level, prices.price, width=0.5, alpha=0.8)

# Set titles
plt.xlabel('Interest level')
plt.ylabel('Median price')
plt.title('Median listing price across interest level')

# Show the plot
plt.show()
```


Median listing price across interest level
Median price
3000
2500
2000
1500
1000
500
0
low
medium
high
Interest level

Let's practice!

WINNING A KAGGLE COMPETITION IN PYTHON

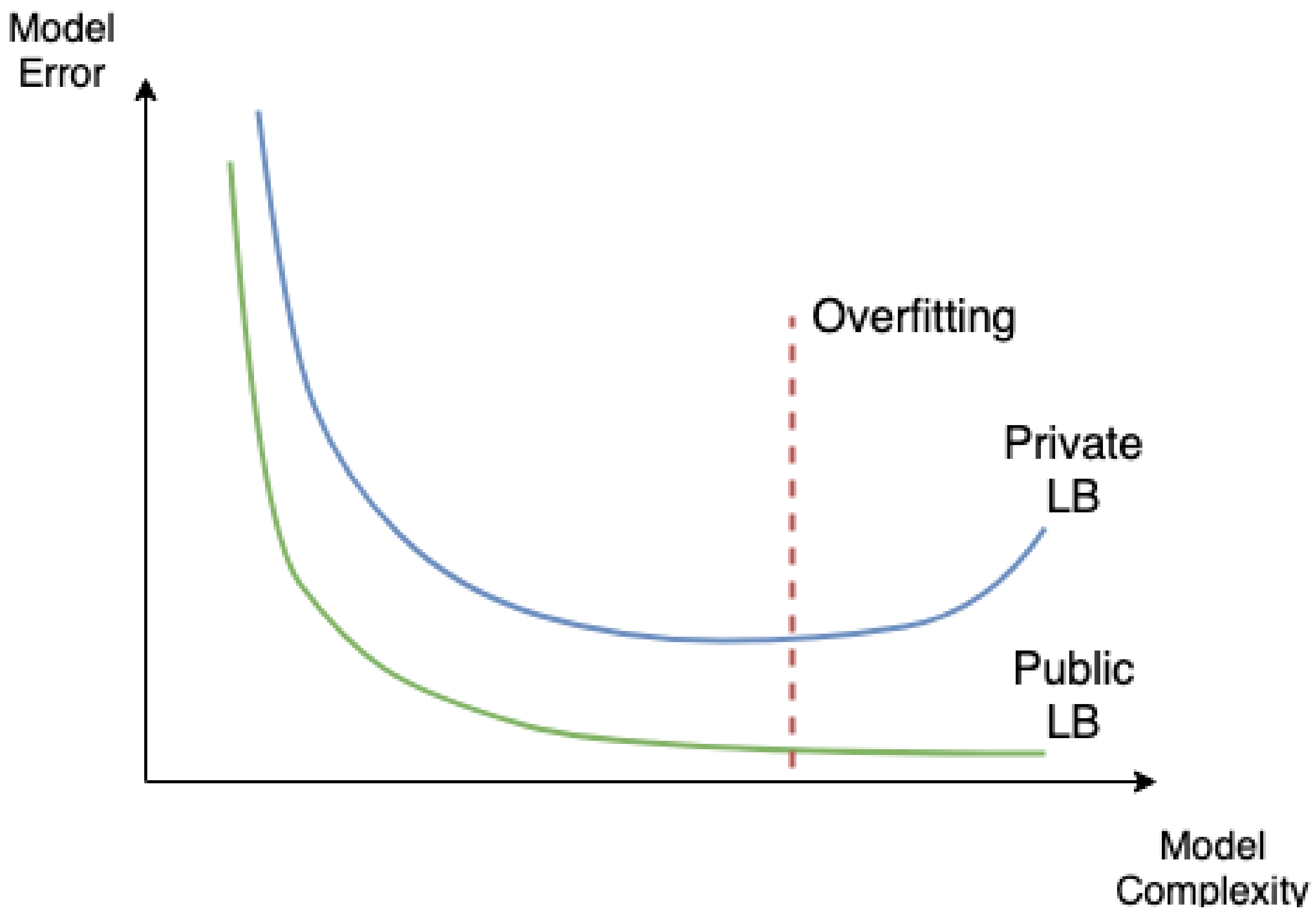
Local validation

WINNING A KAGGLE COMPETITION IN PYTHON

Yauhen Babakhin
Kaggle Grandmaster

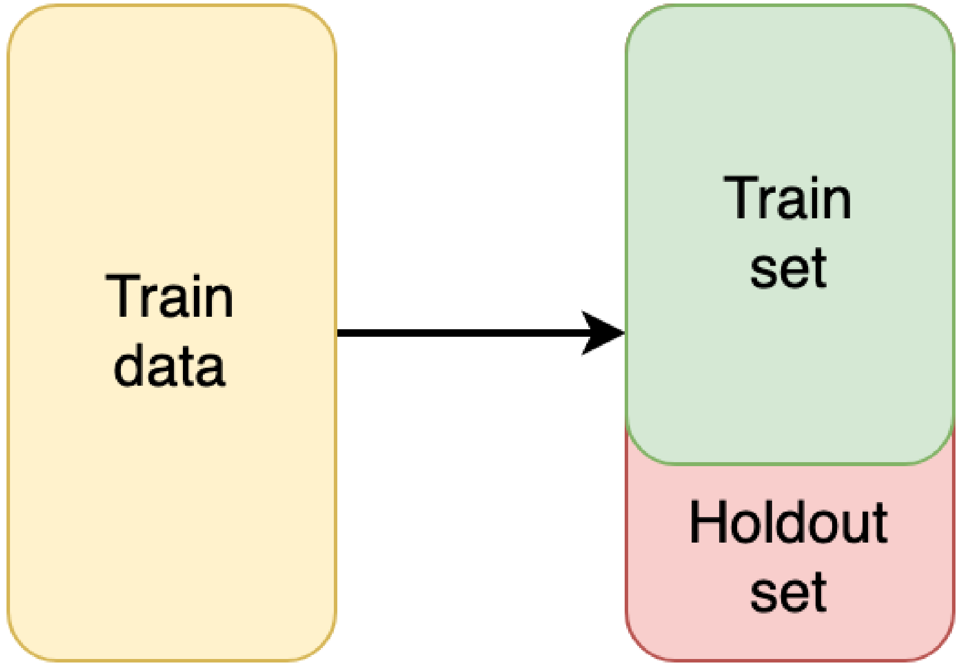
DataCamp

Motivation

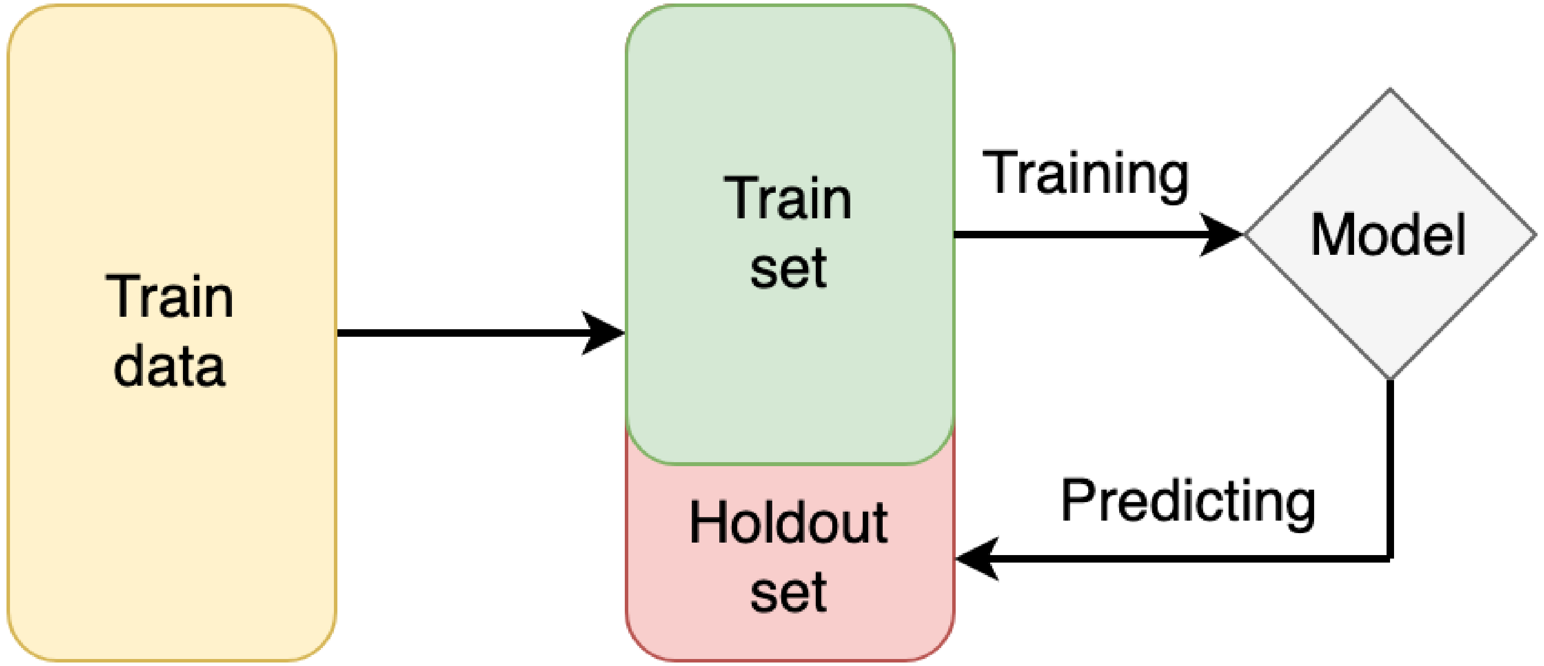


Model Error
Overfitting
Private LB
Public LB
Model Complexity

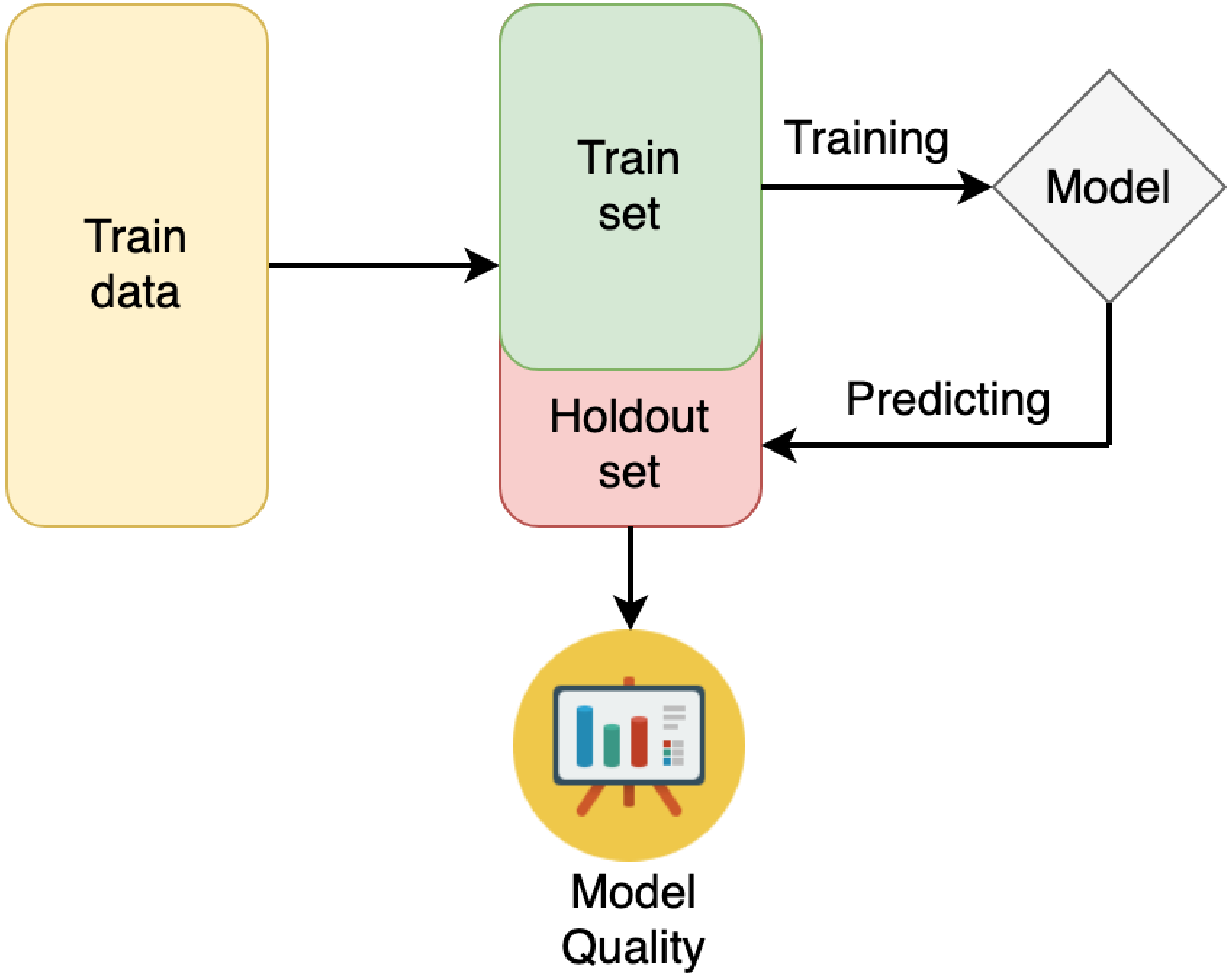
Holdout set



Holdout set

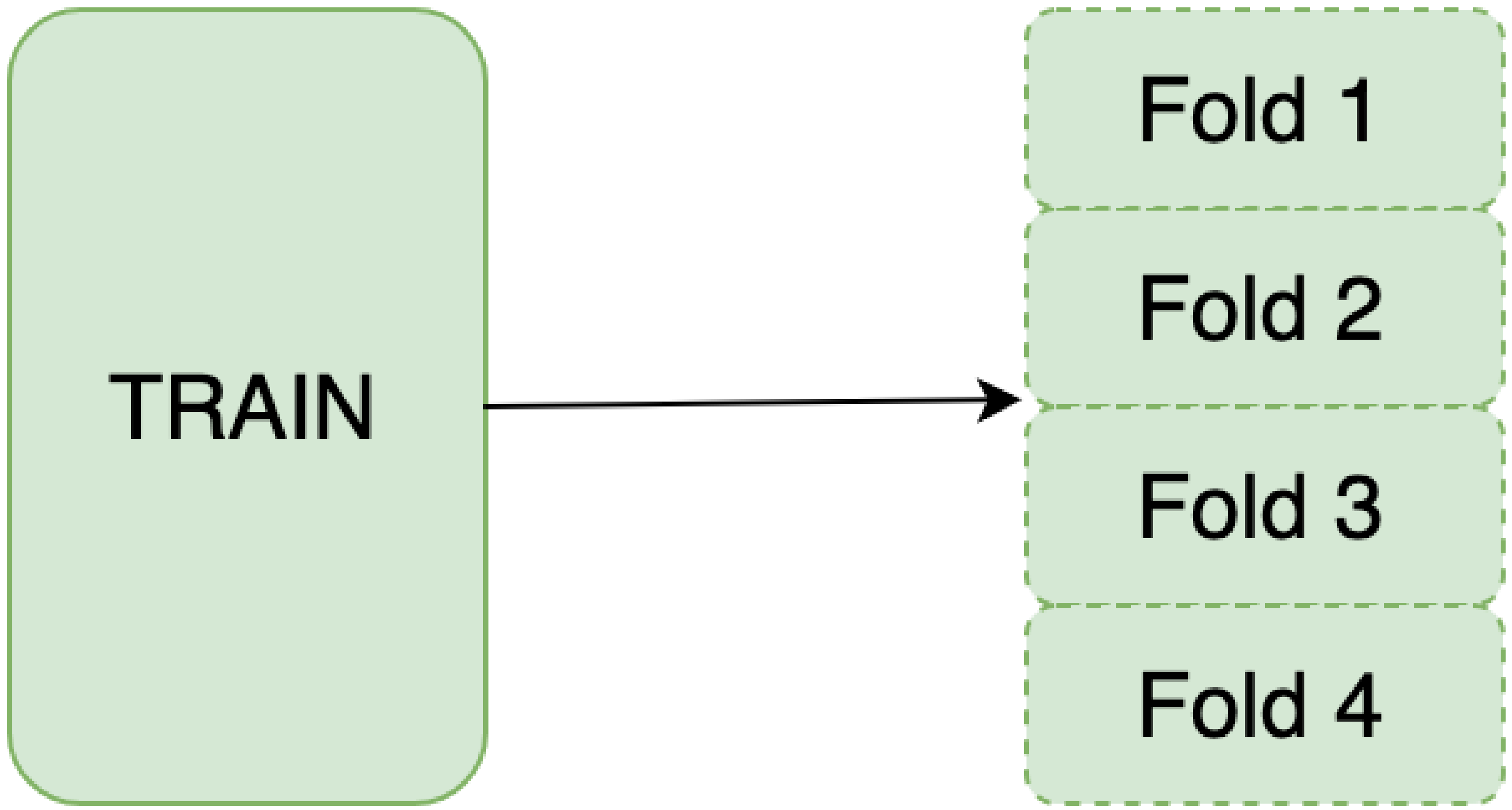

Train
data
Train
set
Holdout
set
Training
Model
Predicting

Holdout set

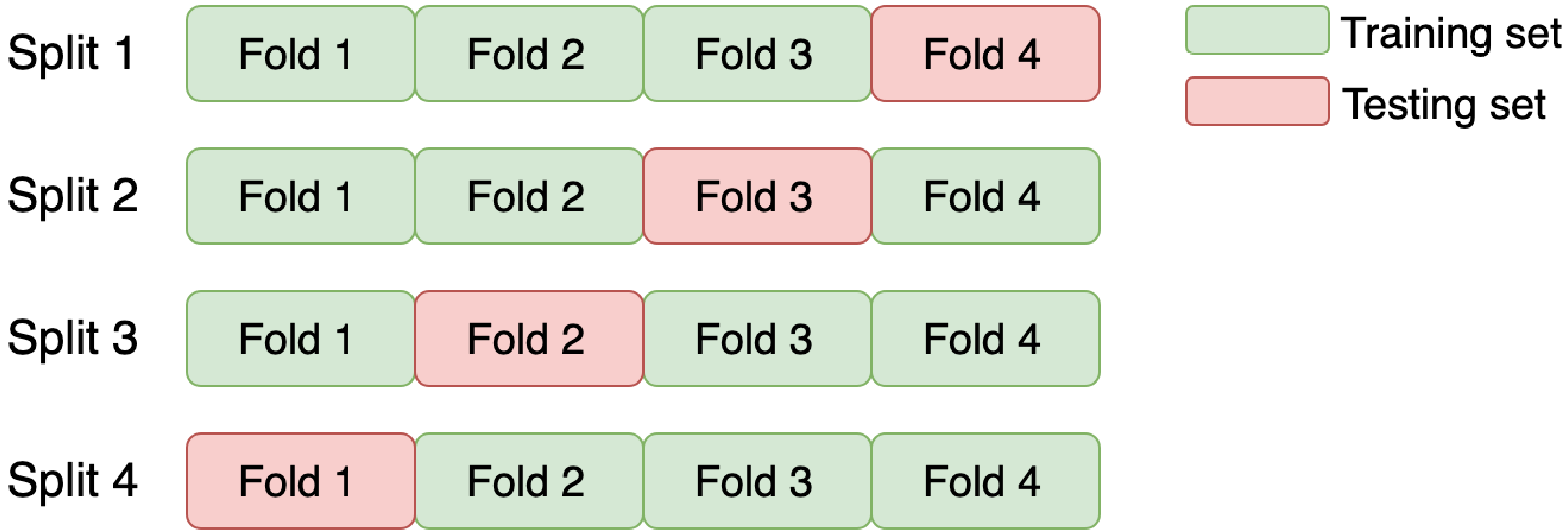


Train
data
Train
set
Training
Model
Holdout
set
Predicting
Model
Quality

K-fold cross-validation



K-fold cross-validation



K-fold cross-validation

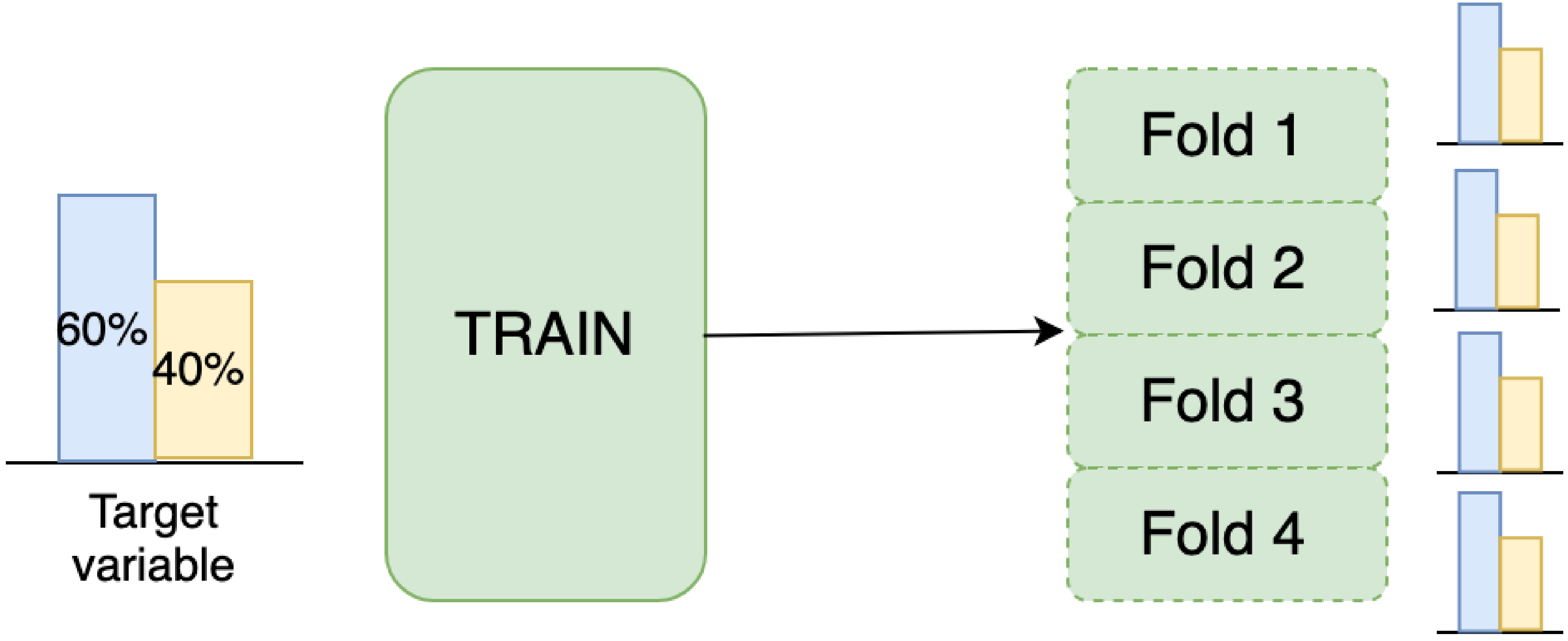
```
# Import KFold
from sklearn.model_selection import KFold
```

```
# Create a KFold object
kf = KFold(n_splits=5, shuffle=True, random_state=123)
```

```
# Loop through each cross-validation split
for train_index, test_index in kf.split(train):

    # Get training and testing data for the corresponding split
    cv_train, cv_test = train.iloc[train_index], train.iloc[test_index]
```

Stratified K-fold



Stratified K-fold

```
# Import StratifiedKFold
from sklearn.model_selection import StratifiedKFold


# Create a StratifiedKFold object
str_kf = StratifiedKFold(n_splits=5, shuffle=True, random_state=123)


# Loop through each cross-validation split
for train_index, test_index in str_kf.split(train, train['target']):
    cv_train, cv_test = train.iloc[train_index], train.iloc[test_index]
```

Let's practice!

WINNING A KAGGLE COMPETITION IN PYTHON

DataCamp

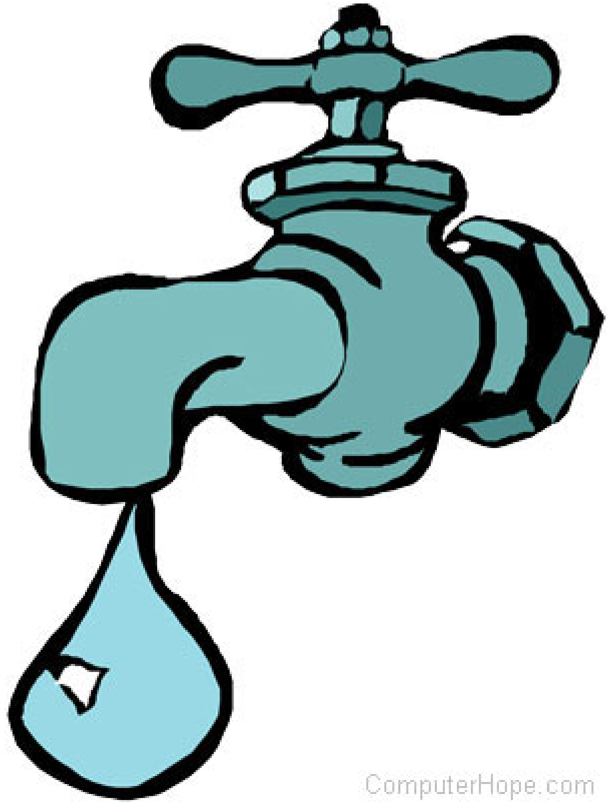
Validation usage

WINNING A KAGGLE COMPETITION IN PYTHON

Yauhen Babakhin
Kaggle Grandmaster

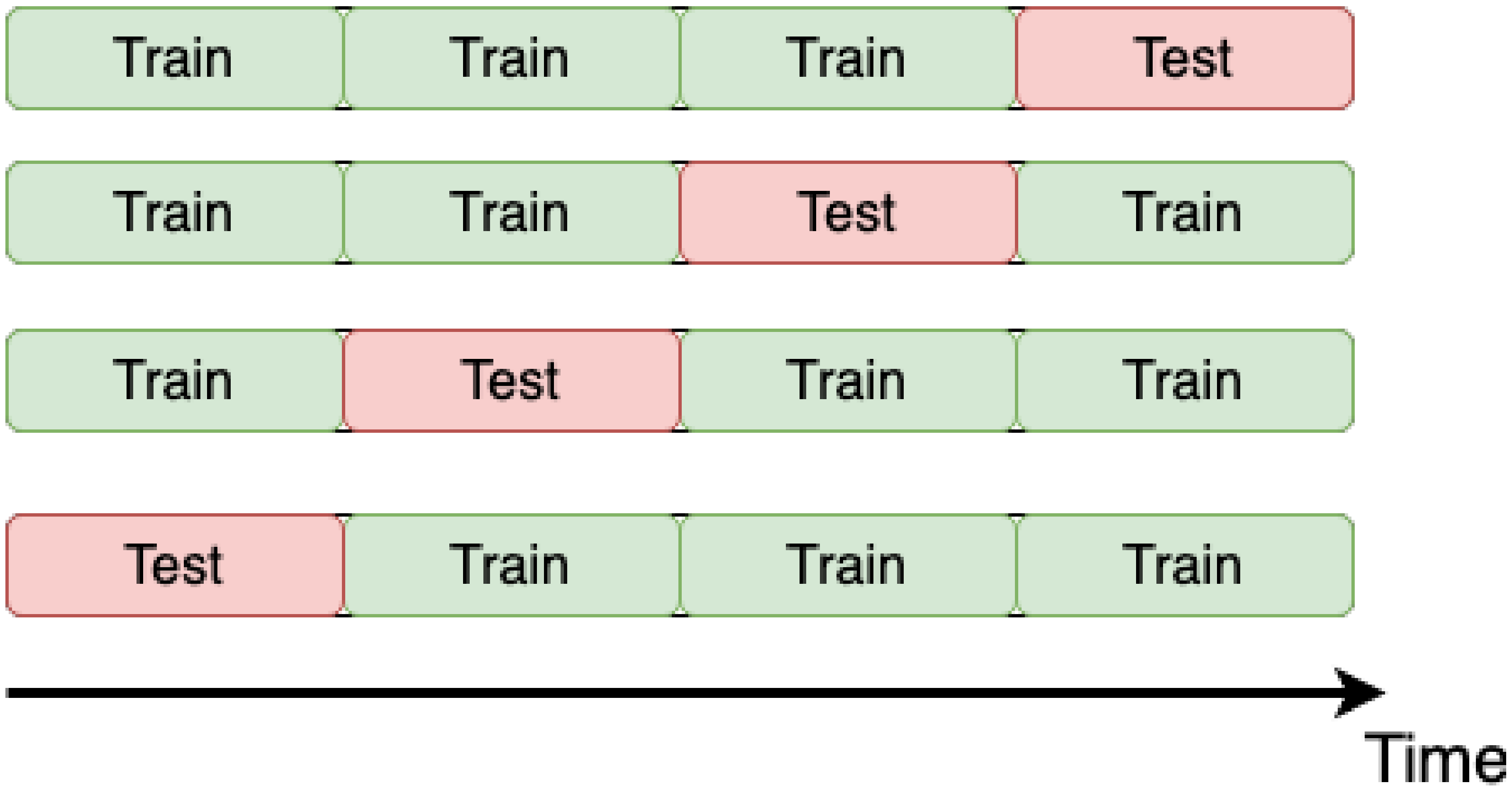
DataCamp

Data leakage

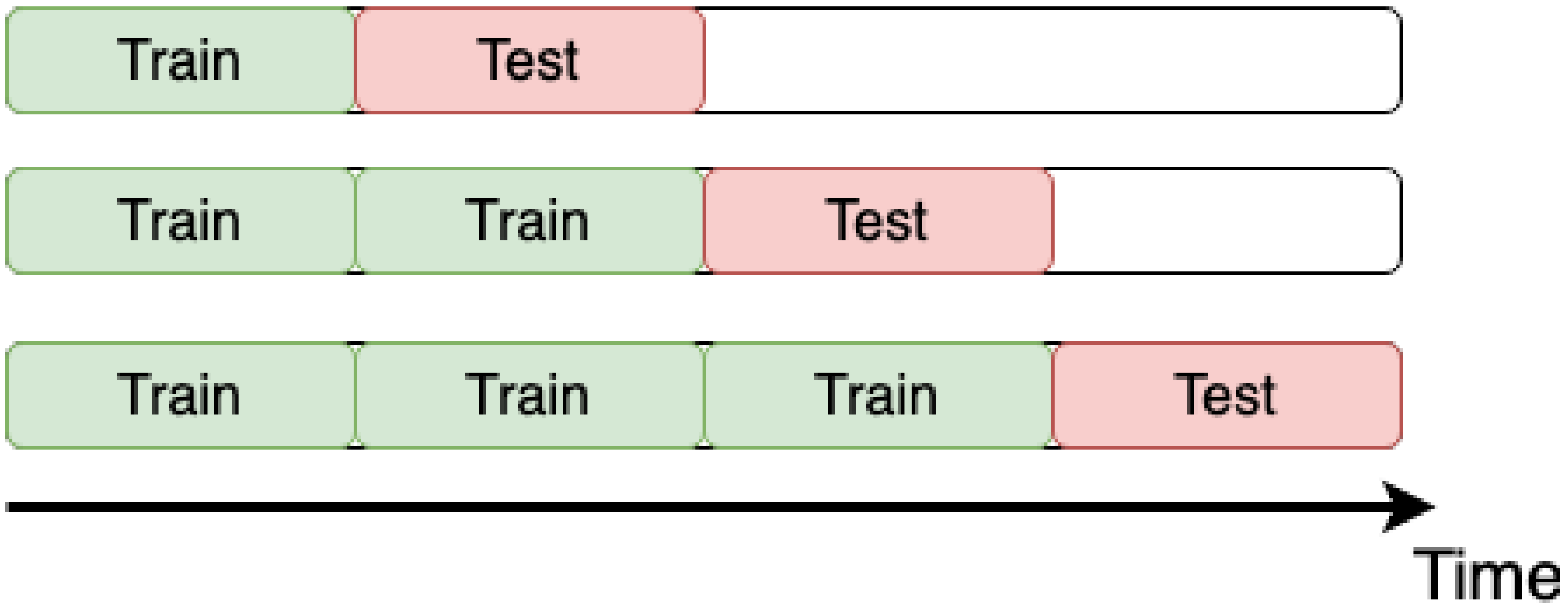


- Leak in **features** – using data that will not be available in the real setting
- Leak in **validation strategy** – validation strategy differs from the real-world situation

Time data


Train
Train
Train
Test
Train
Train
Test
Train
Train
Test
Train
Train
Test
Train
Train
Train
Time

Time K-fold cross-validation



Time K-fold cross-validation

```
# Import TimeSeriesSplit
from sklearn.model_selection import TimeSeriesSplit

# Create a TimeSeriesSplit object
time_kfold = TimeSeriesSplit(n_splits=5)
```

```
# Sort train by date
train = train.sort_values('date')

# Loop through each cross-validation split
for train_index, test_index in time_kfold.split(train):
    cv_train, cv_test = train.iloc[train_index], train.iloc[test_index]
```

Validation pipeline

```
# List for the results
fold_metrics = []

for train_index, test_index in CV_STRATEGY.split(train):
    cv_train, cv_test = train.iloc[train_index], train.iloc[test_index]

    # Train a model
    model.fit(cv_train)

    # Make predictions
    predictions = model.predict(cv_test)

    # Calculate the metric
    metric = evaluate(cv_test, predictions)
    fold_metrics.append(metric)
```

Model comparison

Fold number	Model A MSE	Model B MSE
Fold 1	2.95	2.97
Fold 2	2.84	2.45
Fold 3	2.62	2.73
Fold 4	2.79	2.83

Overall validation score

```python
import numpy as np

# Simple mean over the folds
mean_score = np.mean(fold_metrics)
```

```python
# Overall validation score
overall_score_minimizing = np.mean(fold_metrics) + np.std(fold_metrics)
# Or
overall_score_maximizing = np.mean(fold_metrics) - np.std(fold_metrics)
```

Model comparison

Fold number	Model A MSE	Model B MSE
Fold 1	2.95	2.97
Fold 2	2.84	2.45
Fold 3	2.62	2.73
Fold 4	2.79	2.83
Mean	**2.80**	**2.75**
Overall	**2.919**	**2.935**

Let's practice!