K-FOLD CROSS VALIDATION

Now, we will cover K-fold cross validation. In the next few slides, we will breakdown the cross-validation process line-by-line.

```
from sklearn.model_selection import KFold
import statistics

features = ['TV', 'Radio', 'Newspaper']
X = advertising_df[features]
y = advertising_df['Sales']

# Split the data into ten set
kf = KFold(n_splits=10)

# Create linear regression object
linear_reg = linear_model.LinearRegression()

# List of RMSE for each fold
k_fold_RMSE = []

# Iterate through each fold and calculate the RMSE for each fold
for train_index, test_index in kf.split(X):

    # Extract the training and test data
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    # Fit model
    linear_reg_model = linear_reg.fit(X_train, y_train)
    y_pred = linear_reg_model.predict(X_test)

    #Calculate RMSE for the fold and append it
    RMSE = mean_squared_error(y_test, y_pred, squared=False)
    k_fold_RMSE.append(RMSE)

print('The RMSE for each fold is:', k_fold_RMSE)
print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

1. Separate dataset into features & outcomes
2. Generate k-fold indexes for Train & Test set
3. Split dataset into Train/Test sets for the particular iteration
4. Train model & generate predictions for model evaluation

K-FOLD CROSS VALIDATION

1 Separate dataset into features & outcomes

```
1  from sklearn.model_selection import KFold
2  import statistics
3
4  features = ['TV', 'Radio', 'Newspaper']
5  y = advertising_df['Sales']
6  X = advertising_df[features]
7
8  # Split the data into ten set
9  kf = KFold(n_splits=10)
10
11 # Create linear regression object
12 linear_reg = linear_model.LinearRegression()
13
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
```

Slice the original dataset.
X is a DataFrame of all the **features**.
Y is a series with all **outcomes**.

Create a KFold object which helps split the data into 10 folds later

Create the model object

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

Pass the **features** to the KFold object which produces the train/test indexes

2 Generate k-fold indexes for Train & Test set (details next slide)

* For the **`Kfold()`** function, there are optional parameters **`shuffle=True, random_state=__`** which we recommend for you to use in your personal projects.

K-FOLD CROSS VALIDATION

```
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
19
20     # Extract the training and test data
21     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
22     y_train, y_test = y.iloc[train_index], y.iloc[test_index]
23
24     # Fit model
25     linear_reg_model = linear_reg.fit(X_train, y_train)
26     y_pred = linear_reg_model.predict(X_test)
27
28     #Calculate RMSE for the fold and append it
29     RMSE = mean_squared_error(y_test, y_pred, squared=False)
30     k_fold_RMSE.append(RMSE)
31
32 print('The RMSE for each fold is:', k_fold_RMSE)
33 print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

2 Generate k-fold indexes for Train & Test set

- Each iteration, kf.split(X) returns us 2 lists:
  - The first list comprises 9 folds of the DataFrame **row index** (we store these indices in **`train_index`**)
  - The second list comprises 1-fold of the DataFrame **row index** (we store these indices in **`test_index`**)
- Observe that in iteration 1, the chosen test index were 0 – 20, and in iteration 2, it was 20 – 39. This carries on until iteration 10, where **all the 200 rows of data would have a chance at being the test data.**

What `kf.split(X)` does

Iteration 1

```
[ 20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37
  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55
  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73
  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91
  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109
 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145
 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163
 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181
 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199]
```

train_index (9 folds)

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
```

test_index (1 fold)

Iteration 2

```
[  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
  18  19  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55
  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73
  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91
  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109
 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145
 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163
 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181
 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199]
```

train_index (9 folds)

```
[20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39]
```

test_index (1 fold)

Iteration 10

K-FOLD CROSS VALIDATION

```
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
19
20     # Extract the training and test data
21     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
22     y_train, y_test = y.iloc[train_index], y.iloc[test_index]
23
24     # Fit model
25     linear_reg_model = linear_reg.fit(X_train, y_train)
26     y_pred = linear_reg_model.predict(X_test)
27
28     #Calculate RMSE for the fold and append it
29     RMSE = mean_squared_error(y_test, y_pred, squared=False)
30     k_fold_RMSE.append(RMSE)
31
32 print('The RMSE for each fold is:', k_fold_RMSE)
33 print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

3 Split dataset into Train/Test sets for the particular iteration

Let's focus on Iteration 1 now. We use **`iloc`** to filter indexes from kf.split(X). Details next slide.

Iteration 1

```
[ 20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37
  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55
  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73
  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91
  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109
 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145
 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163
 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181
 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199]
```

train_index (9 folds)

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
```

test_index (1 fold)

K-FOLD CROSS VALIDATION

```
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
19
20     # Extract the training and test data
21     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
22     y_train, y_test = y.iloc[train_index], y.iloc[test_index]
23
24     # Fit model
25     linear_reg_model = linear_reg.fit(X_train, y_train)
26     y_pred = linear_reg_model.predict(X_test)
27
28     #Calculate RMSE for the fold and append it
29     RMSE = mean_squared_error(y_test, y_pred, squared=False)
30     k_fold_RMSE.append(RMSE)
31
32 print('The RMSE for each fold is:', k_fold_RMSE)
33 print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

3

Split dataset into Train/Test sets for the particular iteration

Let's focus on Iteration 1 now. We use **iloc** to filter indexes from kf.split(X).

Iteration 1

Recall that

Train index: **Rows 20 to 200**

Test index: **Rows 0 to 19**

X.iloc [test_index]

Row 0 to 19 of features

X.iloc [train_index]

Row 20 to 200 of features

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

X_test

y_test

X_train

y_train

y.iloc [test_index]

Row 0 to 19 of outcomes

y.iloc [train_index]

Row 20 to 200 of outcomes

K-FOLD CROSS VALIDATION

```
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
19
20     # Extract the training and test data
21     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
22     y_train, y_test = y.iloc[train_index], y.iloc[test_index]
23
24     # Fit model
25     linear_reg_model = linear_reg.fit(X_train, y_train)
26     y_pred = linear_reg_model.predict(X_test)
27
28     #Calculate RMSE for the fold and append it
29     RMSE = mean_squared_error(y_test, y_pred, squared=False)
30     k_fold_RMSE.append(RMSE)
31
32 print('The RMSE for each fold is:', k_fold_RMSE)
33 print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

4 Train model & generate predictions for model evaluation

Iteration 1

Train index: **Rows 20 to 200**

Test index: **Rows 0 to 19**

Model is trained on **9 folds of features and outcomes**

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

X_test y_test X_train y_train

K-FOLD CROSS VALIDATION

```
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
19
20     # Extract the training and test data
21     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
22     y_train, y_test = y.iloc[train_index], y.iloc[test_index]
23
24     # Fit model
25     linear_reg_model = linear_reg.fit(X_train, y_train)
26     y_pred = linear_reg_model.predict(X_test)
27
28     #Calculate RMSE for the fold and append it
29     RMSE = mean_squared_error(y_test, y_pred, squared=False)
30     k_fold_RMSE.append(RMSE)
31
32 print('The RMSE for each fold is:', k_fold_RMSE)
33 print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

4 Train model & generate predictions for model evaluation

Iteration 1

Train index: **Rows 20 to 200**

Test index: **Rows 0 to 19**

Trained model is used to **predict outcomes from 1 fold**, so predictions can be used for evaluation (y_pred are the predicted outcomes)

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

X_test y_test X_train y_train

K-FOLD CROSS VALIDATION

```
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
19
20     # Extract the training and test data
21     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
22     y_train, y_test = y.iloc[train_index], y.iloc[test_index]
23
24     # Fit model
25     linear_reg_model = linear_reg.fit(X_train, y_train)
26     y_pred = linear_reg_model.predict(X_test)
27
28     #Calculate RMSE for the fold and append it
29     RMSE = mean_squared_error(y_test, y_pred, squared=False)
30     k_fold_RMSE.append(RMSE)
31
32 print('The RMSE for each fold is:', k_fold_RMSE)
33 print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

4 Train model & generate predictions for model evaluation

Iteration 1

Train index: **Rows 20 to 200**

Test index: **Rows 0 to 19**

Now we calculate the error of the **predictions** (y_pred) versus the **actual outcomes** (y_test)

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

X_test y_test

X_train y_train

K-FOLD CROSS VALIDATION

```
14 # List of RMSE for each fold
15 k_fold_RMSE = []
16
17 # Iterate through each fold and calculate the RMSE for each fold
18 for train_index, test_index in kf.split(X):
19
20     # Extract the training and test data
21     X_train, X_test = X.iloc[train_index], X.iloc[test_index]
22     y_train, y_test = y.iloc[train_index], y.iloc[test_index]
23
24     # Fit model
25     linear_reg_model = linear_reg.fit(X_train, y_train)
26     y_pred = linear_reg_model.predict(X_test)
27
28     #Calculate RMSE for the fold and append it
29     RMSE = mean_squared_error(y_test, y_pred, squared=False)
30     k_fold_RMSE.append(RMSE)
31
32 print('The RMSE for each fold is:', k_fold_RMSE)
33 print('The average RMSE is:', statistics.mean(k_fold_RMSE))
```

4 Train model & generate predictions for model evaluation

Iteration 1	.....	Iteration 2	.....
Train index: **Rows 20 to 200**		Train index: **Rows 0 - 19, 40 - 200**	
Test index: **Rows 0 to 19**		Test index: **Rows 20 - 39**	

Finally, we append the RMSE of this fold to a list,
… and on to the next iteration where we deal with a new set of train / test indexes!