MODEL EVALUATION AND REFINEMENT

- MODEL EVALUATION
- UNDER / OVER FITTING
- MODEL SELECTION
- RIDGE REGRESSION
- GRID SEARCH

HOW CAN YOU BE CERTAIN YOUR MODEL WORKS IN THE REAL WORLD AND PERFORMS OPTIMALLY???

MODEL EVALUATION

- IN-SAMPLE EVALUATION: HOW WELL OUR MODEL WILL FIT THE DATA USED TO TRAIN IT

HOW TO KNOW HOW WELL IT PREDICTS NEW DATA?

- DIVIDE DATA

TRAIN DATA ↑ LARGER ≈ 70% AND TEST DATA ↑ SMALLER ≈ 30% (OUT OF SAMPLE)

AFTER TESTING THE MODEL WE SHOULD USE ALL THE DATA TO TRAIN THE MODEL TO GET THE BEST PERFORMANCE

FUNCTION train_test_split()
- Split data into random train and test subsets

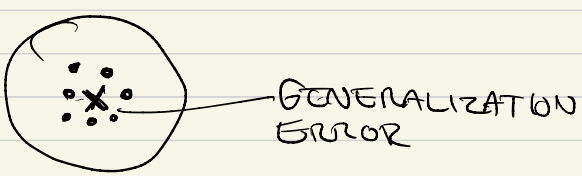
```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test =
    train_test_split(X_data, y_data,
                     test_size = 0.3,
                     random_state = 0)
```

GENERALIZATION ERROR
- HOW WELL OUR MODEL DOES AT PREDICTING PREVIOUSLY UNSEEN DATA

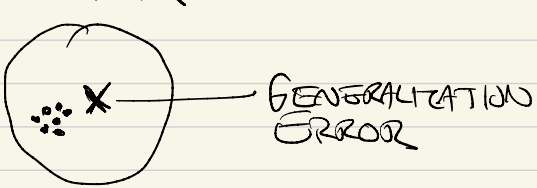
- LOTS OF TRAINING DATA

ACCURACY ↑ PRECISION ↓



- VERY FEW TRAINING DATA

ACCURACY ↓ PRECISION ↑



CROSS VALIDATION (OUT OF SAMPLE EVALUATION)

- DATASETS SPLIT INTO K-EQUAL GROUPS
  - FOLDS
- MOST COMMON EVAL METRICS
- MORE EFFECTIVE USE OF DATA
  (EACH OBS IS USED FOR BOTH TRAINING AND TEST)

AVERAGE RESULT
OF THE ERROR

EVALUATION METRICS
DEPENDS ON THE MODEL

R-SQUARED

HOW TO APPLY CROSS VALIDATION

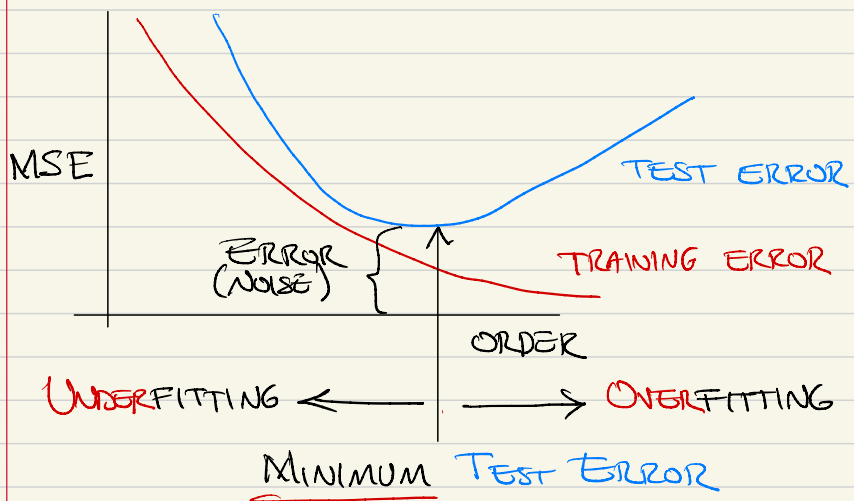
```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(lr, x_data, y_data, cv=3)
np.mean(scores)
```
(lr: model)

```
yhat = cross_val_predict(lrze, x_data, y_data,
                         cv=3)
```

Over/Under Fitting and Model Selection

Under Fitting → Too Many Errors

Over Fitting → Predicts the Noise instead of the Function

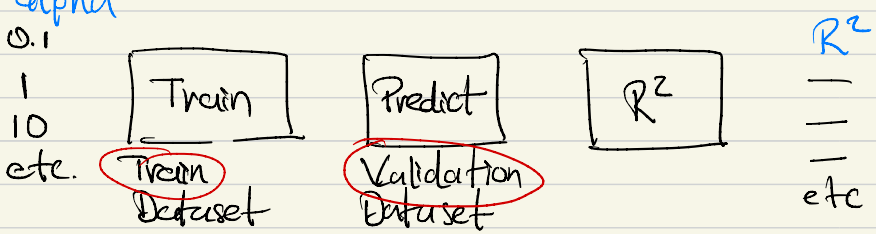


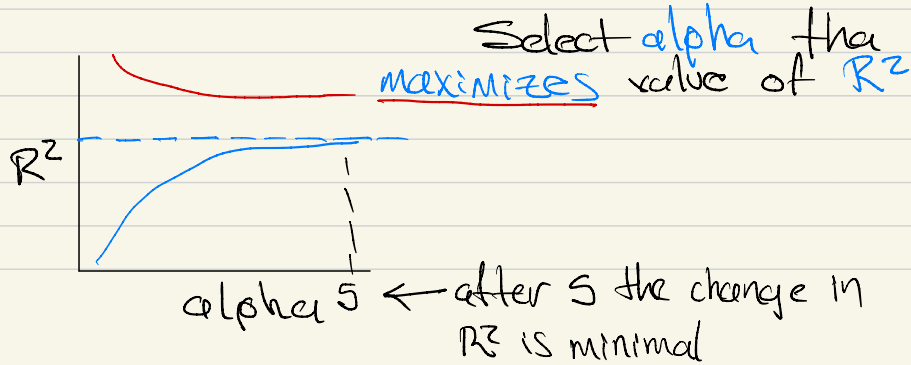
RIDGE REGRESSION

- RR PREVENTS OVERFITTING, THAT IS A BIG PROBLEM WHEN HAVING MULTIPLE INDEPENDENT VARIABLES

- CONTROLS THE MAGNITUDE OF THE POLYNOMIAL COEFFICIENTES USING THE PARAMETER ALPHA

```
from sklearn.linear_model import Ridge
RidgeModel = Ridge(alpha=0.1)
RidgeModel.fit(X, y)
Yhat = RidgeModel.predict(X)
```





GRID SEARCH

- ALLOW TO SCAN THROUGH MULTIPLE FREE PARAMETERS WITH FEW LINES OF CODE

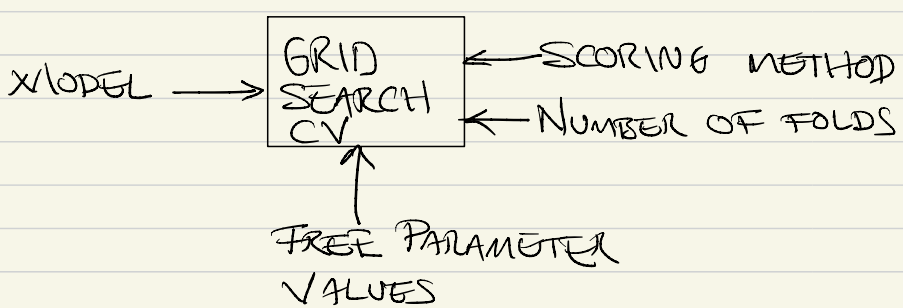
  HYPERPARAMETRES SUCH AS ALPHA (HP)

- ITERATES OVER THE HYPERPARAMETERS USING CROSS-VALIDATION CALLED GRID SEARCH

- GRID SEARCH TAKES THE MODEL OR OBJECTS TO TRAIN AND DIFFERENT VALUES OF HPs, IT THEN CALCULATES THE MSE OR R^2 FOR EACH HP, ALLOWING TO CHOOSE THE BEST VALUES

- EACH ITERATION PRODUCES AN ERROR, WE SELECT THE ONE THAT MINIMIZES IT

- IT USES THE 3 SETS: TRAINING, VALIDATION AND TEST SET.



```
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV

parameters1 = [{'alpha': [0.001, 0.01, 0.1, 1, 10, 100]}]

RR = Ridge()  #object creation

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

Grid1.fit(x_data, y_data)

Grid1.best_estimator_

scores = Grid1.cv_results_
scores['mean_test_score']
```

R^2 IS THE DEFAULT SCORE

ANOTHER OPTION

```
parameters1 = [{'alpha': [1, 10, 100], 'normalize': [True, False]}]
```

OUTPUT

Alpha	1	10	100	
True	R_1	R_2	R_3	} RESULT VALUES
False	R_4	R_5	R_6	