Steps

- Frame the problem
- Data exploration
- Preprocess and feature selection
- Model development
- Final evaluation

Frame the problem

- What is your goal? Classification? Regression?
- What performance measure should you use?

Data Exploration

- After splitting your dataset (train/val/test?)
- Plot data
- See correlations
- Observe ranges of data

Preprocess and feature selection

Remember look only at the training data! Perform the same transformation on the validation/test

- Get a list of things to try (see if it improves or makes it worse):
  - Remove duplicate/unneeded features
  - Remove outliers
  - Drop rows with missing values or impute missing values (mean imputation, learn the value using regression, etc.)
  - Feature encoding by transforming categorical variables into numerical variables (e.g., nominal: on hot encoding, ordinal: label encoder)
  - Feature normalization (scaling)
  - Feature engineering: with domain knowledge, common transformations
  - Dimensionality reduction (PCA). We will discuss this topic in class.
  - If needed, deal with an imbalanced dataset

Model the dataset

- For each of your three approaches:
    1. Fit the model using:
        - different options you developed in the preprocessing stage or during the learning process
        - different hyper-parameter choices (regularization, etc.)
    2. Evaluate the fitted model with your criteria (accuracy, recall, precision, f1 score, etc., …).
    3. See how you could improve your model.

Experiment!

Finally

- Use your test set.

Read more at: https://neptune.ai/blog/life-cycle-of-a-machine-learning-project

1. Working with text Previously posted
2. Working with unbalanced datasets Previously posted
3. Hyperparameter tuning
4. Very will briefly discuss diagnosing errors

Ack!!! There are too many parameters to tune

sklearn's [RandomizedSearchCV] and [GridSearchCV] are classes for parameter tuning that methodically builds and evaluates different combinations of parameters as a grid

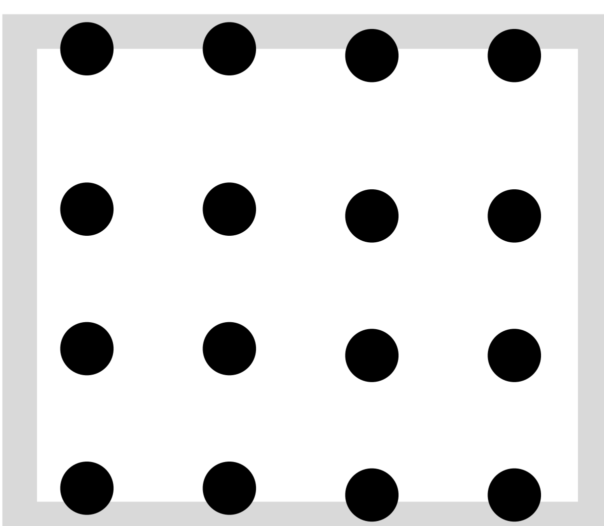
Two main approaches

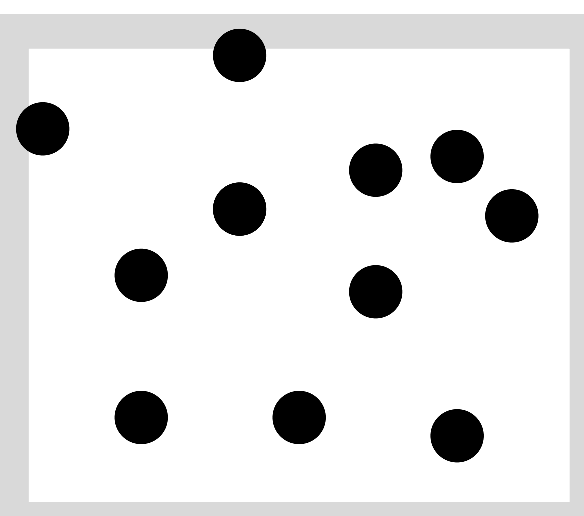
Heuristics (not rules!)

If you have n hyperparameters, the search space is all the different values the n different hyperparameters can take. You can think of this as an n-dimensional volume.

1. Grid search: create a grid on search space.

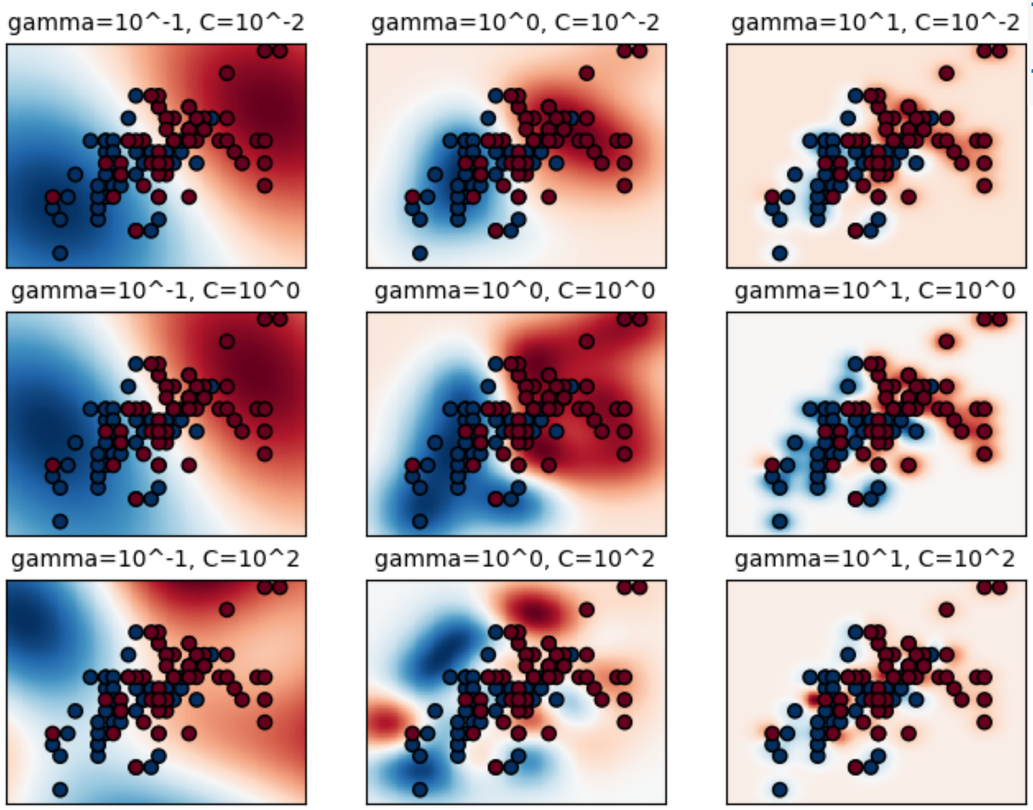
2. Random: randomly select items in the search space.





GridSearchCV

These also need to be imported




```
from sklearn.svm import SVC
from sklearn.model_selection import StratifiedShuffleSpli
from sklearn.model_selection import GridSearchCV

# Train classifiers
#
# For an initial search, a logarithmic grid with basis
# 10 is often helpful. Using a basis of 2, a finer
# tuning can be achieved but at a much higher cost.

C_range = np.logspace(-2, 1, 3)
gamma_range = np.logspace(-1, 1, 3)
param_grid = dict(gamma=gamma_range, C=C_range)
cv = StratifiedShuffleSplit(n_splits=5, test_size=0.2,
grid = GridSearchCV(SVC(), param_grid=param_grid, cv=cv)
grid.fit(X, y)

print("The best parameters are %s with a score of %0.2f"
      % (grid.best_params_, grid.best_score_))
```


Example from http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

Example uses:
C_2d_range = [1e-2, 1, 1e2]
gamma_2d_range = [1e-1, 1, 1e1]

Learn more at:

https://chrisalbon.com/code/machine_learning/model_selection/hyperparameter_tuning_using_random_search/

1. Working with text
2. Working with unbalanced datasets
3. Hyperparameter tuning
4. Very will briefly discuss diagnosing errors

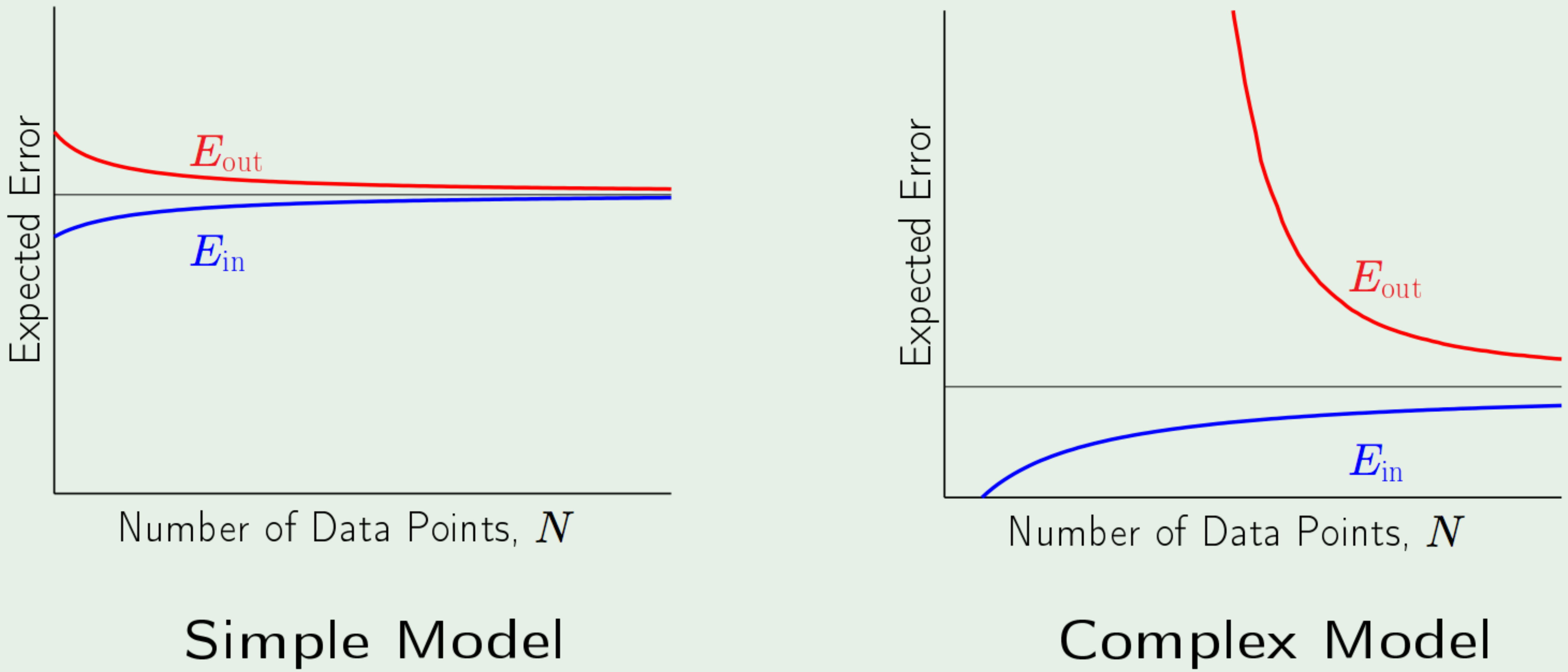
Improvements!

- Ideas:
  - Get more data
  - Add more features
  - perform feature reduction
  - Scale data (bin data, take the log of a feature, standardize, …)
  - Regularize
  - Try a different model
  - Use more/fewer iterations/different learning rate
  - Use a different optimizer

Bias vs Variance

- Run diagnostics
  - Overfitting - big gap between training and validation
  - Underfitting - high training error

The curves


Expected Error
E_{out}
E_{in}
Number of Data Points, N
Simple Model
Expected Error
E_{out}
E_{in}
Number of Data Points, N
Complex Model