Ensemble Methods

Stanford University
DATASCI / STATS 112

Revisiting the Bordeaux Wine Data

```
import pandas as pd

df_bordeaux = pd.read_csv(
    "http://dlsun.github.io/pods/data/bordeaux.csv",
    index_col="year")
X_train = df_bordeaux.loc[:1980, ["summer", "har", "win"]]
y_train = df_bordeaux.loc[:1980, "price"]
X_test = df_bordeaux.loc[1981:, ["summer", "har", "win"]]
```

Suppose we have trained two machine learning models on this data.

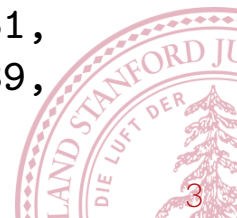
Two Models

```
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsRegressor
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import cross_val_score

model1 = make_pipeline(
    StandardScaler(),
    KNeighborsRegressor(n_neighbors=2, metric="euclidean"))
model1.fit(X_train, y_train)
model1.predict(X_test)
```

```
array([36. , 33.5, 73.5, 33.5, 48. , 14.5, 30.5, 73.5, 48.5, 48.5, 51.5])
```

```
from sklearn.linear_model import LinearRegression

model2 = LinearRegression()
model2.fit(X_train, y_train)
model2.predict(X_test)
```

```
array([39.3953522 , 54.45701963, 66.57148197, 30.818935  , 54.43684661,
       19.77585036, 33.95796178, 63.14084833, 72.47782861, 76.43960489,
       50.58871732])
```



How to combine these predictions?

One approach is to average the predictions. This is called **voting**.

```
(model1.predict(X_test) + model2.predict(X_test)) / 2
```

```
array([37.6976761 , 43.97850981, 70.03574099, 32.1594675 , 51.21842331,
       17.13792518, 32.22898089, 68.32042417, 60.48891431, 62.46980245,
       51.04435866])
```

Another approach is to train *another* machine learning model on top of the predictions (from the individual models) to produce one overall prediction. This is called **stacking**.

```
def get_predictions(X):
  return pd.DataFrame({"model 1": model1.predict(X),
                       "model 2": model2.predict(X)})
stacker = LinearRegression()
stacker.fit(X=get_predictions(X_train), y=y_train)
stacker.predict(X=get_predictions(X_test))
```

```
array([39.13084336, 41.94228209, 80.74488751, 34.11722177, 54.54780579,
       13.93522838, 32.54693204, 79.60922379, 60.95492636, 62.26641749,
       56.31826047])
```

Methods for combining predictions from machine learning models are called **ensemble methods**.

How do we evaluate ensemble models?

In order to evaluate ensemble models, we need to cross-validate the entire process:

1. Fit each individual models to the training set.
2. Fit a final model on the predictions on the training set (for stacking).
3. Make predictions on the validation set.

Scikit-learn provides `VotingRegressor` and `StackingRegressor` that are estimators you can pass into `cross_val_score`.

Original Models:

```
[-cross_val_score(model, X_train, y_train,
                  scoring="neg_mean_squared_error",
                  cv=4).mean() for model in [model1, model2]]
```

```
[288.63690476190476, 251.73652195521595]
```

Ensemble Models:

```
from sklearn.ensemble import VotingRegressor
voter = VotingRegressor([("Model 1", model1),
                         ("Model 2", model2)])
-cross_val_score(voter, X_train, y_train,
                 scoring="neg_mean_squared_error",
                 cv=4).mean()
```

```
240.54681582178412
```

```
from sklearn.ensemble import StackingRegressor
stacker = StackingRegressor([("Model 1", model1),
                             ("Model 2", model2)],
                            final_estimator=LinearRegression())
-cross_val_score(stacker, X_train, y_train,
                 scoring="neg_mean_squared_error",
                 cv=4).mean()
```

```
285.9842338179883
```

Your Turn

- Find a partner. (If there are an odd number of people in section, then you may need to form a group of 3.)
- Work on ensembling your best models from Assignment 4 in the Assignment 5 Colab.
- Use cross-validation to see if the ensemble model is better than your individual models.
- Feel free to complete Assignment 5 later with a different partner in another section. This activity is just designed to get you started.