This notebook is an exercise in the Intermediate Machine Learning course. You can reference the tutorial at this link.

As a warm-up, you'll review some machine learning fundamentals and submit your initial results to a Kaggle competition.

Setup

The questions below will give you feedback on your work. Run the following cell to set up the feedback system.

In [ ]:

```
# Set up code checking
import os
if not os.path.exists("../input/train.csv"):
    os.symlink("../input/home-data-for-ml-course/train.csv", "../input/t
rain.csv")
    os.symlink("../input/home-data-for-ml-course/test.csv", "../input/te
st.csv")
from learntools.core import binder
binder.bind(globals())
from learntools.ml_intermediate.ex1 import *
print("Setup Complete")
```

You will work with data from the Housing Prices Competition for Kaggle Learn Users to predict home prices in Iowa using 79 explanatory variables describing (almost) every aspect of the homes.

Ames Housing dataset image

Run the next code cell without changes to load the training and validation features in `X_train` and `X_valid`, along with the prediction targets in `y_train` and `y_valid`. The test features are loaded in `X_test`. (*If you need to review **features** and **prediction targets**, please check out this short tutorial. To read about model **validation**, look here. Alternatively, if you'd prefer to look through a full course to review all of these topics, start here.*)

In [ ]:

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Read the data
X_full = pd.read_csv('../input/train.csv', index_col='Id')
X_test_full = pd.read_csv('../input/test.csv', index_col='Id')

# Obtain target and predictors
y = X_full.SalePrice
features = ['LotArea', 'YearBuilt', '1stFlrSF', '2ndFlrSF', 'FullBath',
'BedroomAbvGr', 'TotRmsAbvGrd']
X = X_full[features].copy()
X_test = X_test_full[features].copy()

# Break off validation set from training data
X_train, X_valid, y_train, y_valid = train_test_split(X, y, train_size=
0.8, test_size=0.2,
                                                      random_state=0)
```

Use the next cell to print the first several rows of the data. It's a nice way to get an overview of the data you will use in your price prediction model.

In [ ]:

```
X_train.head()
```

The next code cell defines five different random forest models. Run this code cell without changes. (*To review **random forests**, look here.*)

In [ ]:

```
from sklearn.ensemble import RandomForestRegressor

# Define the models
model_1 = RandomForestRegressor(n_estimators=50, random_state=0)
model_2 = RandomForestRegressor(n_estimators=100, random_state=0)
model_3 = RandomForestRegressor(n_estimators=100, criterion='mae', rando
m_state=0)
model_4 = RandomForestRegressor(n_estimators=200, min_samples_split=20,
random_state=0)
model_5 = RandomForestRegressor(n_estimators=100, max_depth=7, random_st
ate=0)

models = [model_1, model_2, model_3, model_4, model_5]
```

To select the best model out of the five, we define a function `score_model()` below. This function returns the mean absolute error (MAE) from the validation set. Recall that the best model will obtain the lowest MAE. (*To review **mean absolute error**, look here.*)

Run the code cell without changes.

In [ ]:

```
from sklearn.metrics import mean_absolute_error

# Function for comparing different models
def score_model(model, X_t=X_train, X_v=X_valid, y_t=y_train, y_v=y_vali
d):
    model.fit(X_t, y_t)
    preds = model.predict(X_v)
    return mean_absolute_error(y_v, preds)

for i in range(0, len(models)):
    mae = score_model(models[i])
    print("Model %d MAE: %d" % (i+1, mae))
```

Step 1: Evaluate several models

Use the above results to fill in the line below. Which model is the best model? Your answer should be one of `model_1`, `model_2`, `model_3`, `model_4`, or `model_5`.

In [ ]:

```
from sklearn.ensemble import RandomForestRegressor

# Define the models
model_1 = RandomForestRegressor(n_estimators=50, random_state=0)
model_2 = RandomForestRegressor(n_estimators=100, random_state=0)
model_3 = RandomForestRegressor(n_estimators=100, criterion='mae', rando
m_state=0)
model_4 = RandomForestRegressor(n_estimators=200, min_samples_split=20,
random_state=0)
model_5 = RandomForestRegressor(n_estimators=100, max_depth=7, random_st
ate=0)

models = [model_1, model_2, model_3, model_4, model_5]
```

In [ ]:

```
from sklearn.metrics import mean_absolute_error

# Function for comparing different models
def score_model(model, X_t=X_train, X_v=X_valid, y_t=y_train, y_v=y_vali
d):
    model.fit(X_t, y_t)
    preds = model.predict(X_v)
    return mean_absolute_error(y_v, preds)

for i in range(0, len(models)):
    mae = score_model(models[i])
    print("Model %d MAE: %d" % (i+1, mae))
```

In [ ]:

```
# Fill in the best model
best_model = model_3

# Check your answer
step_1.check()
```

In [ ]:

```
# Lines below will give you a hint or solution code
#step_1.hint()
#step_1.solution()
```

Step 2: Generate test predictions

Great. You know how to evaluate what makes an accurate model. Now it's time to go through the modeling process and make predictions. In the line below, create a Random Forest model with the variable name `my_model`.

In [ ]:

```
# Define a model
my_model = RandomForestRegressor(random_state=0) # Your code here

# Check your answer
step_2.check()
```

In [ ]:

```
# Lines below will give you a hint or solution code
#step_2.hint()
#step_2.solution()
```

Run the next code cell without changes. The code fits the model to the training and validation data, and then generates test predictions that are saved to a CSV file. These test predictions can be submitted directly to the competition!

In [ ]:

```
# Fit the model to the training data
my_model.fit(X, y)

# Generate test predictions
preds_test = my_model.predict(X_test)

# Save predictions in format used for competition scoring
output = pd.DataFrame({'Id': X_test.index,
                       'SalePrice': preds_test})
output.to_csv('submission.csv', index=False)
```

Submit your results

Once you have successfully completed Step 2, you're ready to submit your results to the leaderboard! First, you'll need to join the competition if you haven't already. So open a new window by clicking on this link. Then click on the **Join Competition** button.

join competition image

Next, follow the instructions below:

1. Begin by clicking on the blue **Save Version** button in the top right corner of the window. This will generate a pop-up window.
2. Ensure that the **Save and Run All** option is selected, and then click on the blue **Save** button.
3. This generates a window in the bottom left corner of the notebook. After it has finished running, click on the number to the right of the **Save Version** button. This pulls up a list of versions on the right of the screen. Click on the ellipsis **(...)** to the right of the most recent version, and select **Open in Viewer**. This brings you into view mode of the same page. You will need to scroll down to get back to these instructions.
4. Click on the **Output** tab on the right of the screen. Then, click on the file you would like to submit, and click on the blue **Submit** button to submit your results to the leaderboard.

You have now successfully submitted to the competition!

If you want to keep working to improve your performance, select the blue **Edit** button in the top right of the screen. Then you can change your code and repeat the process. There's a lot of room to improve, and you will climb up the leaderboard as you work.

Keep going

You've made your first model. But how can you quickly make it better?

Learn how to improve your competition results by incorporating columns with **missing values**.

Have questions or comments? Visit the Learn Discussion forum to chat with other Learners.