Files
main
Go to file t
1. Basic Data Explore
2. Machine Learning Model
3. Model Validation
4. Underfitting and Overfitting
5. Random Forest Algorithm
6. Machine Learning Competitions
Muhammad Habibulloh - Intro t...
README.md
Documentation • Share feedback

In [4]:
```
import pandas as pd

# Path file dataset home-data-for-ml-course/train.csv
iowa_file_path = '../input/home-data-for-ml-course/train.csv'

# Membaca file (read.csv) dan menyimpan datanya dalam variabel home_data
home_data = pd.read_csv(iowa_file_path)

# Memanggil baris di bawah ini tanpa argumen untuk memeriksa bahwa Anda telah memuat data dengan benar
step_1.check()
```

Correct

In [5]:
```
import pandas as pd

iowa_file_path = '../input/home-data-for-ml-course/train.csv'
home_data = pd.read_csv(iowa_file_path)
step_1.check()
```

Correct

Step 2: Review The Data

Use the command you learned to view summary statistics of the data. Then fill in variables to answer the following questions

In [6]:
```
# Cetak statistik ringkasan pada baris berikutnya
print(home_data.describe())
```

Kaggle-Intro-to-Machine-Learni | Downloads

github.com/habibumhd/Kaggle-Intro-to-Machine-Learning/blob/main/2.%20Machine%20Learning%20Model/exercise-your-first-machine-learning-model.ip...

Files

main

Go to file t

- 1. Basic Data Explore
- 2. Machine Learning Model
- 3. Model Validation
- 4. Underfitting and Overfitting
- 5. Random Forest Algorithm
- 6. Machine Learning Competitions
- Muhammad Habibulloh - Intro t...
- README.md

Documentation • Share feedback

Kaggle-Intro-to-Machine-Learning / 2. Machine Learning Model / exercise-your-first-machine-learning-model.ipynb ↑ Top

Preview | Code | Blame | 1 lines (1 loc) · 15.8 KB | Code 55% faster with GitHub Copilot | Raw

Step 1: Specify Prediction Target

Select the target variable, which corresponds to the sales price. Save this to a new variable called `y`. You'll need to print a list of the columns to find the name of the column you need.

In [4]:
```
# Mencetak daftar kolom dalam dataset
print(home_data.columns)
```

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

In [5]:
```
y = home_data['SalePrice']

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

Type here to search

ENG 15:27 08/11/2023

Exercises

Step 1: Split Your Data

Use the `train_test_split` function to split up your data.

Give it the argument `random_state=1` so the `check` functions know what to expect when verifying your code.

Recall, your features are loaded in the DataFrame **X** and your target is loaded in **y**.

In [2]:
```
from sklearn.model_selection import train_test_split

# Memecah data menjadi data pelatihan dan data validasi
train_X, val_X, train_y, val_y = train_test_split(X, y, random_state=1)

# Memeriksa jawaban Anda
step_1.check()
```

Correct

In [ ]:
```
# The lines below will show you a hint or the solution.
# step_1.hint()
# step_1.solution()
```

Step 2: Specify and Fit the Model

Step 1: Compare Different Tree Sizes

Write a loop that tries the following values for *max_leaf_nodes* from a set of possible values.

Call the *get_mae* function on each value of max_leaf_nodes. Store the output in some way that allows you to select the value of `max_leaf_nodes` that gives the most accurate model on your data.

In [5]:

```
from sklearn.metrics import mean_absolute_error
from sklearn.tree import DecisionTreeRegressor

# Fungsi untuk menghitung MAE
def get_mae(max_leaf_nodes, train_X, val_X, train_y, val_y):
    model = DecisionTreeRegressor(max_leaf_nodes=max_leaf_nodes, random_state=0)
    model.fit(train_X, train_y)
    predictions = model.predict(val_X)
    mae = mean_absolute_error(val_y, predictions)
    return mae

# Kumpulan nilai max_leaf_nodes yang mungkin
candidate_max_leaf_nodes = [5, 25, 50, 100, 250, 500]

# Inisialisasi variabel untuk menyimpan MAE terbaik dan best_tree_size
best_mae = float('inf')
best_tree_size = None

# Loop untuk mencari nilai max_leaf_nodes terbaik
for max_leaf_nodes in candidate_max_leaf_nodes:
    current_mae = get_mae(max_leaf_nodes, train_X, val_X, train_y, val_y)
    if current_mae < best_mae:
        best_mae = current_mae
        best_tree_size = max_leaf_nodes
```

Files
main
Go to file t
1. Basic Data Explore
2. Machine Learning Model
3. Model Validation
4. Underfitting and Overfitting
5. Random Forest Algorithm
6. Machine Learning Competitions
Muhammad Habibulloh - Intro t...
README.md

Step 1: Use a Random Forest

In [2]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error

# Mendefinisikan model Random Forest dengan random_state=1
rf_model = RandomForestRegressor(random_state=1)

# Melatih model Random Forest
rf_model.fit(train_X, train_y)

# Menghitung MAE dari model Random Forest pada data validasi
rf_val_predictions = rf_model.predict(val_X)
rf_val_mae = mean_absolute_error(val_y, rf_val_predictions)

# Mencetak MAE model Random Forest pada data validasi
print("Validation MAE for Random Forest Model: {}".format(rf_val_mae))

step_1.check()
```

```
Validation MAE for Random Forest Model: 21857.15912981083
```

Correct

In [ ]:
```
# The lines below will show you a hint or the solution.
# step_1.hint()
# step_1.solution()
```

So far, you have followed specific instructions at each step of your project. This helped learn key ideas and build your first model, but now you know enough to try things on your own.

The code cell above trains a Random Forest model on `train_X` and `train_y`.

Use the code cell below to build a Random Forest model and train it on all of `X` and `y`.

In [7]:
```
# Membuat model Random Forest baru
rf_model_on_full_data = RandomForestRegressor(random_state=1)

# Melatih model rf_model_on_full_data dengan seluruh data pelatihan
rf_model_on_full_data.fit(X, y)
```

Out[7]:
```
RandomForestRegressor(random_state=1)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Now, read the file of "test" data, and apply your model to make predictions.

In [8]:
```
# Path ke file yang akan digunakan untuk prediksi
test_data_path = '../input/test.csv'

# Membaca file data pengujian menggunakan pandas
test_data = pd.read_csv(test_data_path)

# Membuat test_X yang berasal dari data pengujian tetapi hanya mencakup kolom yang Anda gunakan untuk prediksi
test_X = test_data[features]

# Membuat prediksi yang akan Anda kirimkan
test_preds = rf_model_on_full_data.predict(test_X)
```

Before submitting, run a check to make sure your `test_preds` have the right format.