Hands on ML Chapter 2 End to End Project

California Real Estate Prices

Our task is to use California census data to build a model of housing prices in the state.

In [1]:
```
### Fetch the data
import os
import tarfile
import urllib

DOWNLOAD_ROOT = "https://raw.githubusercontent.com/ageron/handson-ml2/master/"
HOUSING_PATH = os.path.join("datasets", "housing")
HOUSING_URL = DOWNLOAD_ROOT + "datasets/housing/housing.tgz"

def fetch_housing_data(housing_url=HOUSING_URL, housing_path=HOUSING_PATH):
    os.makedirs(housing_path, exist_ok=True)
    tgz_path = os.path.join(housing_path, "housing.tgz")
    urllib.request.urlretrieve(housing_url, tgz_path)
    housing_tgz = tarfile.open(tgz_path)
    housing_tgz.extractall(path=housing_path)
    housing_tgz.close()
```

In [2]:
```
fetch_housing_data()
```

In [3]:
```
### Load the data
import pandas as pd

# This function returns a pandas DataFrame object containing all the data.
def load_housing_data(housing_path=HOUSING_PATH):
    csv_path = os.path.join(housing_path, "housing.csv")
    return pd.read_csv(csv_path)
```

In [4]:
```
# Lets take a look at the top five rows
housing = load_housing_data()
housing.head()
# Each row represents one district
```

Out[4]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0

In [5]:
```
# Quick description of the data
housing.info()
# Notice total_bedrooms has 20433 features -> missing 207 districts. We will n
eed to fix this later.
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

In [6]:
```
# Notice ocean_proximity is an 'object' which in this case is most likely tex
t.
housing["ocean_proximity"].value_counts()
```

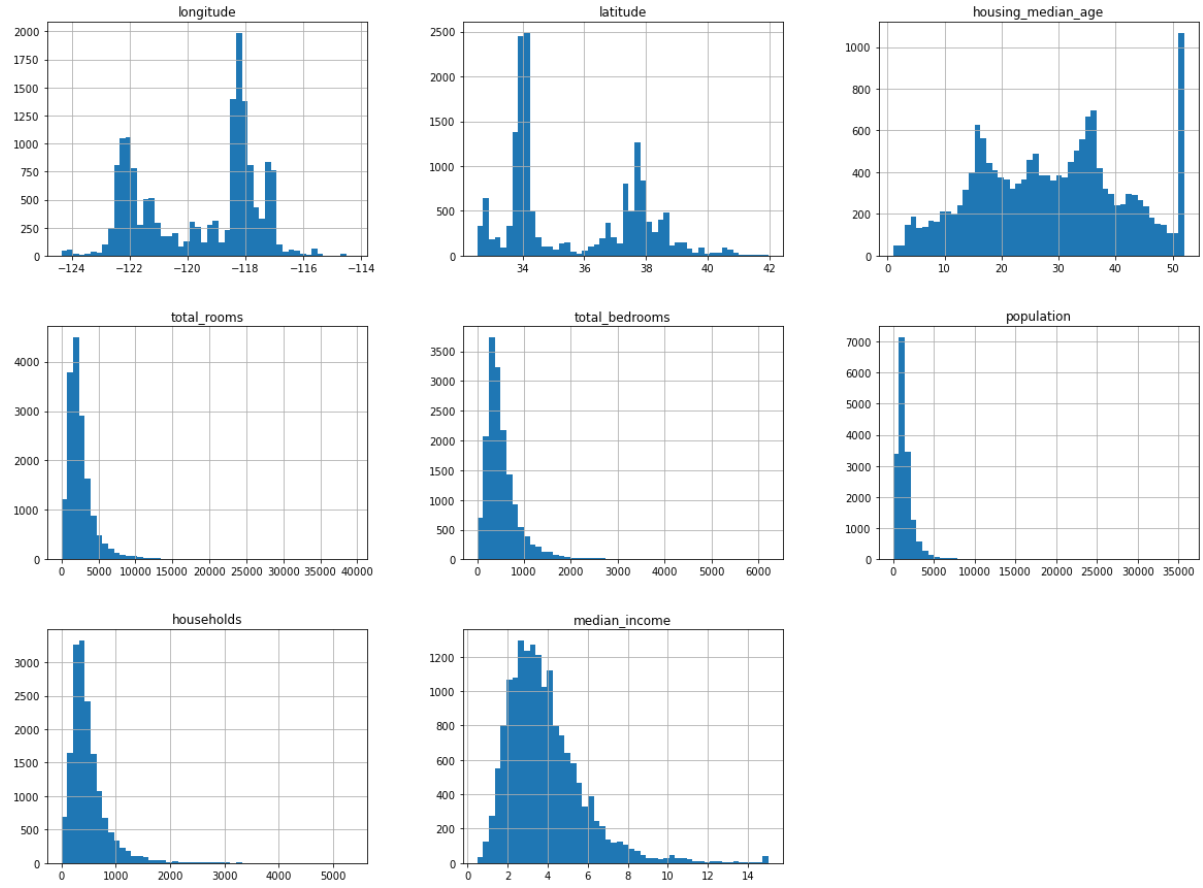
Out[6]:
```
<1H OCEAN     9136
INLAND        6551
NEAR OCEAN    2658
NEAR BAY      2290
ISLAND           5
Name: ocean_proximity, dtype: int64
```

In [7]:
```
# Describe shows a summary of the numerical attributes.
housing.describe()
```

Out[7]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	popul
count	20640.000000	20640.000000	20640.000000	20640.000000	20433.000000	20640.00
mean	-119.569704	35.631861	28.639486	2635.763081	537.870553	1425.47
std	2.003532	2.135952	12.585558	2181.615252	421.385070	1132.46
min	-124.350000	32.540000	1.000000	2.000000	1.000000	3.00
25%	-121.800000	33.930000	18.000000	1447.750000	296.000000	787.00
50%	-118.490000	34.260000	29.000000	2127.000000	435.000000	1166.00
75%	-118.010000	37.710000	37.000000	3148.000000	647.000000	1725.00
max	-114.310000	41.950000	52.000000	39320.000000	6445.000000	35682.00

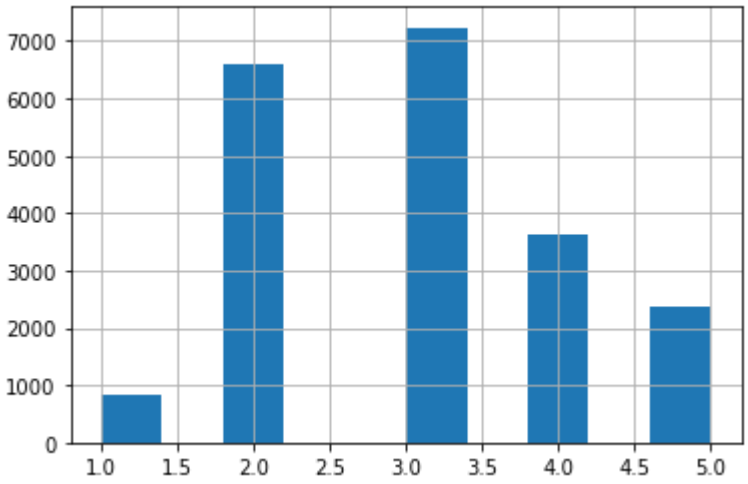
In [49]:
```
# We can also plot a histogram of each numerical attribute to get a feel for o
ur data
%matplotlib inline
# ^ Jupyter notebook command for inline matplotlib
import matplotlib.pyplot as plt
housing.hist(bins=50, figsize=(20,15))
plt.show()
# Notice:
# 1. The median income does not look like it is in USD. The number represents
 roughly 3~$30,000 etc.
#    It is also capped between 0.5-15.0 (4.999-15.0001)
# 2. The housing median age and median house value are also capped.
#    The median house value being capped may be a serious problem as that is w
hat we are trying to predict.
# 3. The attributes all have very different scales.
# 4. Many of the histograms are tail-heavy, ideally we want a more bell-shaped
distribution.
```



Create a Test Set

In [9]:
```
### Create a Test Set
# We are going to use stratified sampling based on median income by creating a
n income_category attribute.
# First we need to create the income categories as follows:
import pandas as pd
import numpy as np
housing["income_cat"] = pd.cut(housing["median_income"],
                               bins=[0.,1.5,3.0,4.5,6.,np.inf],
                               labels=[1,2,3,4,5])
housing["income_cat"].hist()
```

Out[9]: <AxesSubplot:>



In [10]:
```
# This is a possible implementation of splitting the data into a training set
 and test set using the row
# numbers as unique id's so the test set doesn't change on each iteration of t
he model.

#from zlib import crc32

#def test_set_check(identifier, test_ratio):
#    return crc32(np.int64(identifier)) & 0xffffffff < test_ratio * 2**32

#def split_train_test_by_id(data, test_ratio, id_column):
#    ids = data[id_column]
#    in_test_set = ids.apply(lambda id_: test_set_check(id_, test_ratio))
#    return data.loc[~in_test_set], data.loc[in_test_set]

# adds an Index column
#housing_with_id = housing.reset_index()
#train_set, test_set = split_train_test_by_id(housing_with_id, 0.2, "index")

#housing_with_id["id"] = housing["longitude"] * 1000 + housing["latitude"]
#train_set, test_set = split_train_test_by_id(housing_with_id, 0.2, "id")

## We are going to use the following implementation using Scikit-Learn's funct
ions to split the dataset into
## multiple subsets.
from sklearn.model_selection import train_test_split

train_set, test_set = train_test_split(housing, test_size=0.2, random_state=42
)

from sklearn.model_selection import StratifiedShuffleSplit

split = StratifiedShuffleSplit(n_splits=1, test_size=0.2, random_state=42)
for train_index, test_index in split.split(housing, housing["income_cat"]):
    strat_train_set = housing.loc[train_index]
    strat_test_set = housing.loc[test_index]

# Check if this worked correctly by looking at the income category proportions
in the test set
strat_test_set["income_cat"].value_counts() / len(strat_test_set)
```

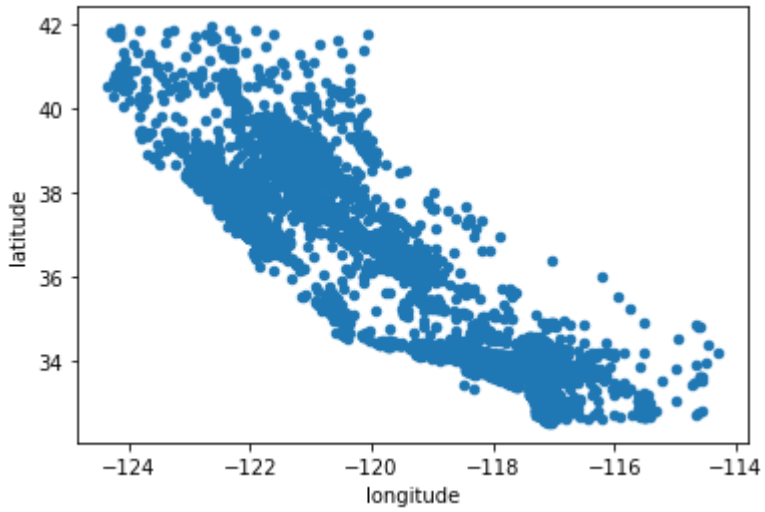
Out[10]:
```
3    0.350533
2    0.318798
4    0.176357
5    0.114583
1    0.039729
Name: income_cat, dtype: float64
```

In [11]:
```
# Remove income_cat attribute so the data is back to its original state
for set_ in (strat_train_set, strat_test_set):
    set_.drop("income_cat", axis=1, inplace=True)
```

Discover and Visualize the Data to Gain Insights

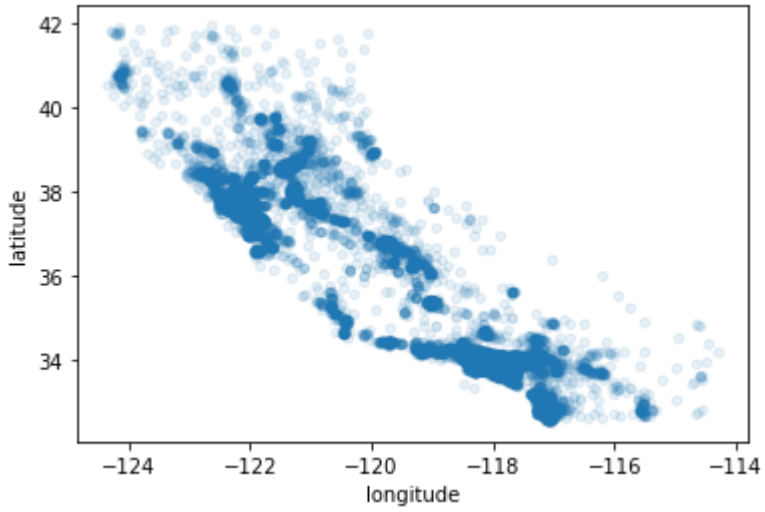
In [12]:
```
housing = strat_train_set.copy() # We are only working with the training set f
rom here on out.
housing.plot(kind="scatter", x="longitude", y="latitude")
```

Out[12]: <AxesSubplot:xlabel='longitude', ylabel='latitude'>



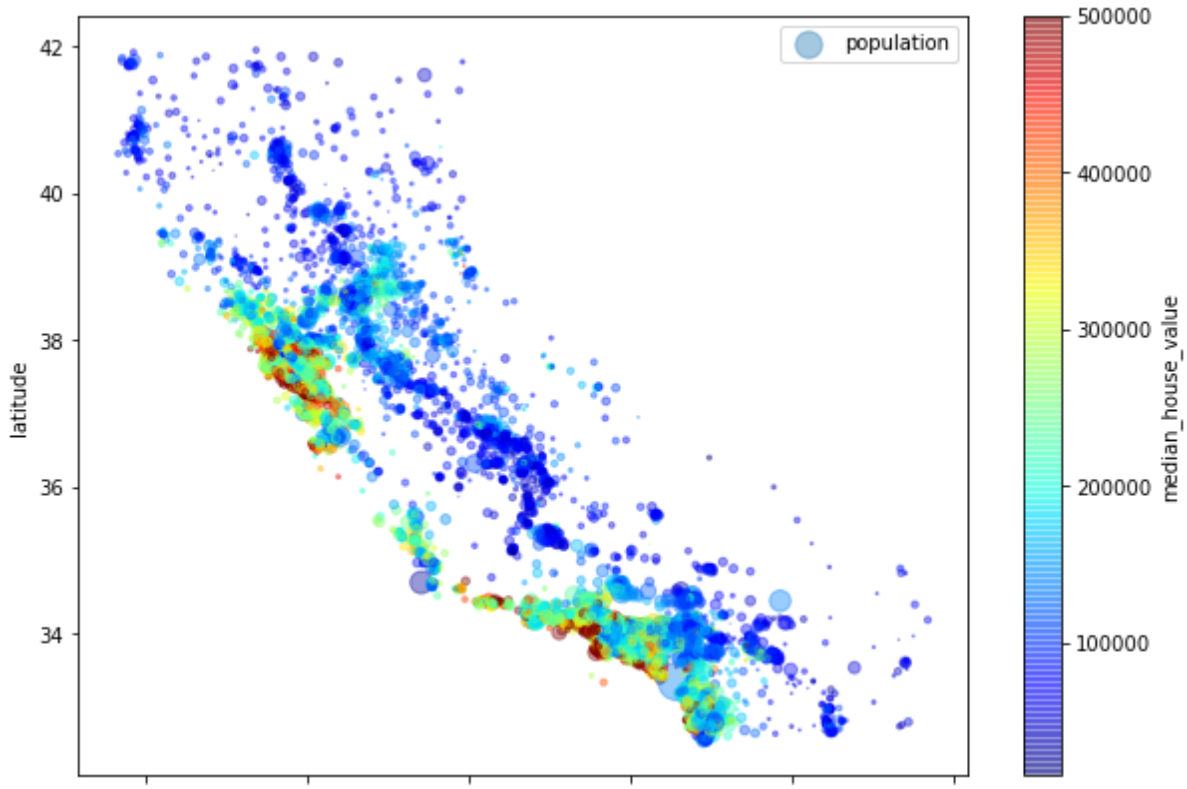
In [13]:
```
housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.1)
```

Out[13]: <AxesSubplot:xlabel='longitude', ylabel='latitude'>



In [14]:
```
housing.plot(kind="scatter",
             x="longitude",
             y="latitude",
             alpha=0.4,
             s=housing["population"]/100,
             label="population",
             figsize=(10,7),
             c="median_house_value",
             cmap=plt.get_cmap("jet"),
             colorbar=True,
            )
plt.legend()
```

Out[14]: <matplotlib.legend.Legend at 0x1b14d025760>

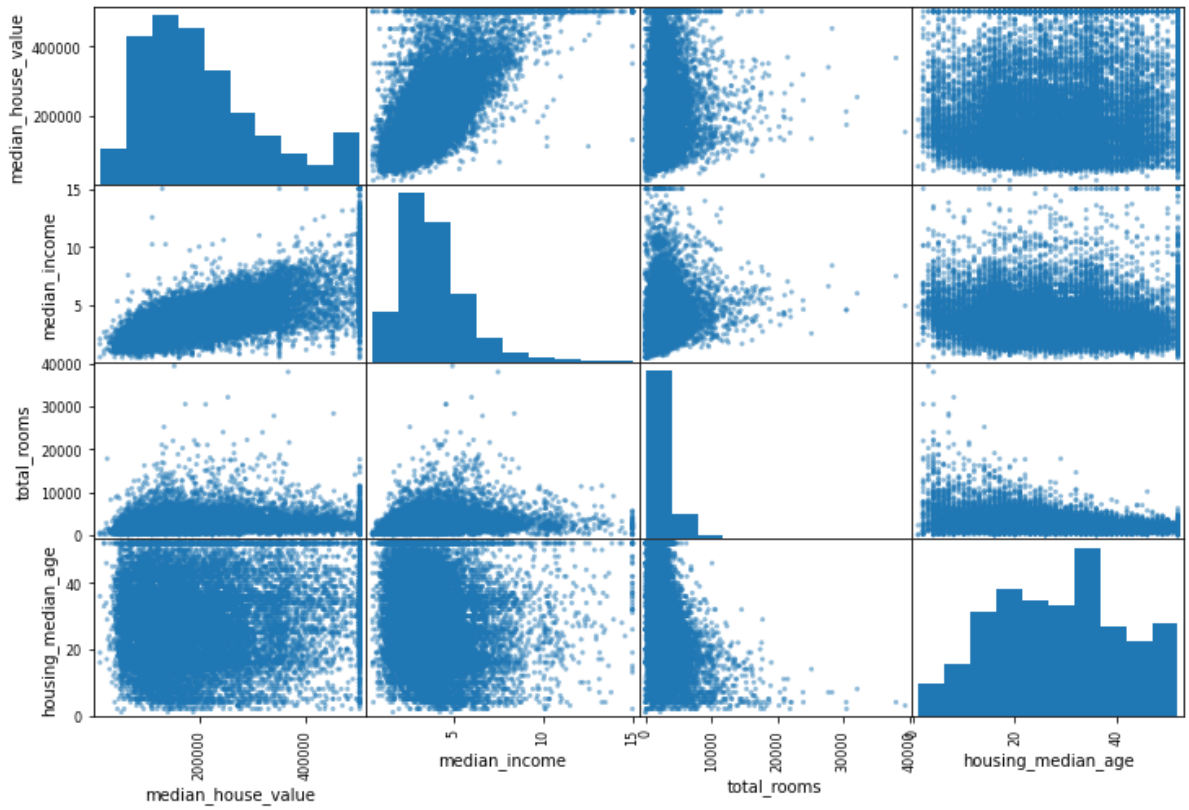


In [15]:
```
## Looking for Correlations
# Compute the standard correlation coefficient (Pearson's r) between every pai
r of attributes
corr_matrix = housing.corr()

# Look at how much each attribute correlates with the median house value
corr_matrix["median_house_value"].sort_values(ascending=False)
```

Out[15]:
```
median_house_value    1.000000
median_income         0.687160
total_rooms           0.135097
housing_median_age    0.114110
households            0.064506
total_bedrooms        0.047689
population           -0.026920
longitude            -0.047432
latitude             -0.142724
Name: median_house_value, dtype: float64
```

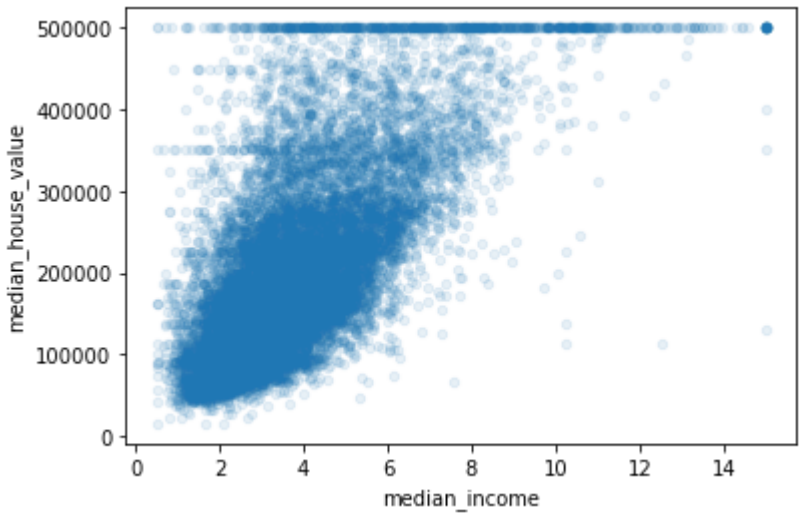
In [16]:
```
# Scatter matrix plots used to plot every numerical attribute against every ot
her numerical attribute,
# plus a histogram of each numberial attribute
from pandas.plotting import scatter_matrix

# Here are just a few since 11*11=121 plots
attributes=["median_house_value", "median_income", "total_rooms", "housing_med
ian_age"]
scatter_matrix(housing[attributes], figsize=(12,8))
```

Out[16]:
```
array([[<AxesSubplot:xlabel='median_house_value', ylabel='median_house_valu
e'>,
        <AxesSubplot:xlabel='median_income', ylabel='median_house_value'>,
        <AxesSubplot:xlabel='total_rooms', ylabel='median_house_value'>,
        <AxesSubplot:xlabel='housing_median_age', ylabel='median_house_valu
e'>],
       [<AxesSubplot:xlabel='median_house_value', ylabel='median_income'>,
        <AxesSubplot:xlabel='median_income', ylabel='median_income'>,
        <AxesSubplot:xlabel='total_rooms', ylabel='median_income'>,
        <AxesSubplot:xlabel='housing_median_age', ylabel='median_income'>],
       [<AxesSubplot:xlabel='median_house_value', ylabel='total_rooms'>,
        <AxesSubplot:xlabel='median_income', ylabel='total_rooms'>,
        <AxesSubplot:xlabel='total_rooms', ylabel='total_rooms'>,
        <AxesSubplot:xlabel='housing_median_age', ylabel='total_rooms'>],
       [<AxesSubplot:xlabel='median_house_value', ylabel='housing_median_ag
e'>,
        <AxesSubplot:xlabel='median_income', ylabel='housing_median_age'>,
        <AxesSubplot:xlabel='total_rooms', ylabel='housing_median_age'>,
        <AxesSubplot:xlabel='housing_median_age', ylabel='housing_median_ag
e'>]],
      dtype=object)
```



In [17]:
```
# median_income to "median_house_value" seems to be the most promising correla
tion
housing.plot(kind="scatter", x="median_income", y="median_house_value", alpha=
0.1)
# Notice also that the price cap at 500,000 is clearly visible
# Notice also that there is another line around 350,000
# We may want to remove these data quirks to prevent our model from reproducin
g them
```

Out[17]: <AxesSubplot:xlabel='median_income', ylabel='median_house_value'>



In [18]:
```
## Create new attribute combinations
housing["rooms_per_household"] = housing["total_rooms"] / housing["households"
]
housing["bedrooms_per_room"] = housing["total_bedrooms"] / housing["total_room
s"]
housing["population_per_household"] = housing["population"] / housing["househo
lds"]

# check out the new combinations
corr_matrix = housing.corr()
corr_matrix["median_house_value"].sort_values(ascending=False)
```

Out[18]:
```
median_house_value          1.000000
median_income               0.687160
rooms_per_household         0.146285
total_rooms                 0.135097
housing_median_age          0.114110
households                  0.064506
total_bedrooms              0.047689
population_per_household   -0.021985
population                 -0.026920
longitude                  -0.047432
latitude                   -0.142724
bedrooms_per_room          -0.259984
Name: median_house_value, dtype: float64
```

Prepare the Data for ML Learning Algorithms

In [21]:
```
housing = strat_train_set.drop("median_house_value", axis=1) # Note that drop
() makes a copy
housing_labels = strat_train_set["median_house_value"].copy()

## Data cleaning
# take care of missing attributes
#housing.dropna(subset=["total_bedrooms"]) # option 1 drop corresponding distr
icts
#housing.drop("total_bedrooms", axis=1) # option 2 drop the whole attribute
#median = housing["total_bedrooms"].median() # option 3 set missing attributes
to the mean
#housing["total_bedrooms"].fillna(median, inplace=True)

# option 3 using scikit learn
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy="median")
housing_numerical_attributes = housing.drop("ocean_proximity", axis=1)
imputer.fit(housing_numerical_attributes)

# compute median of each attribute
imputer.statistics_
housing_numerical_attributes.median().values

# transform the training set by replacing missing values with the learned valu
es
X = imputer.transform(housing_numerical_attributes)

# If you want to turn it back into a pandas dataframe object...
housing_transformed = pd.DataFrame(X, columns=housing_numerical_attributes.col
umns, index=housing_numerical_attributes.index)
# ^ Note this is just to demonstrate, we don't use 'housing_transformed' agai
n.
housing_transformed.head()
```

Out[21]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	housel
17606	-121.89	37.29	38.0	1568.0	351.0	710.0	
18632	-121.93	37.05	14.0	679.0	108.0	306.0	
14650	-117.20	32.77	31.0	1952.0	471.0	936.0	
3230	-119.61	36.31	25.0	1847.0	371.0	1460.0	
3555	-118.59	34.23	17.0	6592.0	1525.0	4459.0	1

In [22]:
```
## Handling text and categorical attributes
housing_categorical_attributes = housing[["ocean_proximity"]]
housing_categorical_attributes.head(10)
```

Out[22]:

	ocean_proximity
17606	<1H OCEAN
18632	<1H OCEAN
14650	NEAR OCEAN
3230	INLAND
3555	<1H OCEAN
19480	INLAND
8879	<1H OCEAN
13685	INLAND
4937	<1H OCEAN
4861	<1H OCEAN

In [23]:
```
from sklearn.preprocessing import OrdinalEncoder
ordinal_encoder = OrdinalEncoder()
housing_categorical_attributes_encoded = ordinal_encoder.fit_transform(housing
_categorical_attributes)
```

In [24]:
```
housing_categorical_attributes_encoded[:10]
```

Out[24]:
```
array([[0.],
       [0.],
       [4.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.]])
```

In [25]:
```
ordinal_encoder.categories_
```

Out[25]:
```
[array(['<1H OCEAN', 'INLAND', 'ISLAND', 'NEAR BAY', 'NEAR OCEAN'],
       dtype=object)]
```

In [26]:
```
from sklearn.preprocessing import OneHotEncoder
cat_encoder = OneHotEncoder()
housing_categorical_attributes_1hot = cat_encoder.fit_transform(housing_catego
rical_attributes)
housing_categorical_attributes_1hot
```

Out[26]:
```
<16512x5 sparse matrix of type '<class 'numpy.float64'>'
        with 16512 stored elements in Compressed Sparse Row format>
```

In [27]:
```
housing_categorical_attributes_1hot.toarray()
```

Out[27]:
```
array([[1., 0., 0., 0., 0.],
       [1., 0., 0., 0., 0.],
       [0., 0., 0., 0., 1.],
       ...,
       [0., 1., 0., 0., 0.],
       [1., 0., 0., 0., 0.],
       [0., 0., 0., 1., 0.]])
```

In [28]:
```
cat_encoder.categories_
```

Out[28]:
```
[array(['<1H OCEAN', 'INLAND', 'ISLAND', 'NEAR BAY', 'NEAR OCEAN'],
       dtype=object)]
```

In [29]:
```
## Custom Transformers

# Here is an example of a small transformer class that adds the combined attri
butes we created above

# The 'add_bedrooms_per_room' hyperparameter will allow us to easily find out
 if
# adding this attribute helps the ML algorithm or not.

from sklearn.base import BaseEstimator, TransformerMixin

rooms_index, bedrooms_index, population_index, households_index = 3, 4, 5, 6 #
The column in X

class CombinedAttributesAdder(BaseEstimator, TransformerMixin):
    def __init__(self, add_bedrooms_per_room = True): # no *args or **kargs
        self.add_bedrooms_per_room = add_bedrooms_per_room

    def fit(self, X, y=None):
        return self # nothing else to do

    def transform(self, X, y=None):
        rooms_per_household = X[:, rooms_index] / X[:, households_index]
        population_per_household = X[:, population_index] / X[:, households_in
dex]
        if self.add_bedrooms_per_room:
            bedrooms_per_room = X[:, bedrooms_index] / X[:, rooms_index]
            return np.c_[X, rooms_per_household, population_per_household, bed
rooms_per_room]
        else:
            return np.c_[X, rooms_per_household, population_per_household]

attr_adder = CombinedAttributesAdder(add_bedrooms_per_room=False)
housing_extra_attribs = attr_adder.transform(housing.values)
```

In [30]:
```
## Feature Scaling

# We perform this below as part of the pipeline
```

In [31]:
```
## Transformation Pipelines

# Here is a small pipeline for the numerical attributes

from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler

numerical_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='median')),
    ('attribs_adder', CombinedAttributesAdder()),
    ('std_scaler', StandardScaler()), # Notice here we are doing the feature s
caling using sklearn StandardScaler
])

housing_numerical_attributes_transformed = numerical_pipeline.fit_transform(ho
using_numerical_attributes)
# ^ Note that we don't use this result, we use the numerical pipeline below in
the full pipeline
```

In [32]:
```
# Full pipeline to apply all the transformations to the housing data

from sklearn.compose import ColumnTransformer

numerical_attribs = list(housing_numerical_attributes)
categorical_attribs = ["ocean_proximity"]

full_pipeline = ColumnTransformer([
    ('num', numerical_pipeline, numerical_attribs),
    ('cat', OneHotEncoder(), categorical_attribs),
])

# ^ This is really the 'data transformation pipeline'

housing_prepared = full_pipeline.fit_transform(housing)
```

Select and Train a Model

In [34]:
```
# Linear Regression Model
from sklearn.linear_model import LinearRegression

# Train the model
linear_reg = LinearRegression()
linear_reg.fit(housing_prepared, housing_labels)

# Test the Linear Regression Model
some_data = housing.iloc[:5]
some_labels = housing_labels.iloc[:5]
some_data_prepared = full_pipeline.transform(some_data)
print('Predictions:', linear_reg.predict(some_data_prepared))
print('Labels:', list(some_labels))
# These are not very accurate
```

```
Predictions: [210644.60459286 317768.80697211 210956.43331178  59218.98886849
 189747.55849879]
Labels: [286600.0, 340600.0, 196900.0, 46300.0, 254500.0]
```

In [35]:
```
# Lets measure the RMSE on the whole training set.
from sklearn.metrics import mean_squared_error
housing_predictions = linear_reg.predict(housing_prepared)
linear_mse = mean_squared_error(housing_labels, housing_predictions)
linear_rmse = np.sqrt(linear_mse)
print(linear_rmse)
# These results are not very good, high error on the training set means the mo
del is underfitting the training data.
# Either:
# the features do not provide enough info to make a good prediction,
# or we need to constrain or the model (regularize it),
# or the model isn't powerful enough.
```

```
68628.19819848923
```

In [36]:
```
# Let's try a more powerful model.
# Decision Tree Regressor
from sklearn.tree import DecisionTreeRegressor

# Train the model
decision_tree_reg = DecisionTreeRegressor()
decision_tree_reg.fit(housing_prepared, housing_labels)

# Test the model
housing_predictions = decision_tree_reg.predict(housing_prepared)
decision_tree_mse = mean_squared_error(housing_labels, housing_predictions)
decision_tree_rmse = np.sqrt(decision_tree_mse)
print(decision_tree_rmse)
# The error is showing up as zero, this can't be right.
# Low error (in this case zero error) on the training set means the model is o
verfitting the training data.
```

```
0.0
```

In [37]:
```
# k-fold cross-validation
from sklearn.model_selection import cross_val_score
scores = cross_val_score(decision_tree_reg, housing_prepared, housing_labels,
scoring="neg_mean_squared_error", cv=10)
decision_tree_rmse_scores = np.sqrt(-scores)

def display_scores(scores):
    print("Scores:", scores)
    print("Mean:", scores.mean())
    print("Standard deviation:", scores.std())

display_scores(decision_tree_rmse_scores)
# We can see the decision tree regressor actually performed worse than linear
 regression from how badly it is overfitting.
```

```
Scores: [67960.42510836 67376.42270742 70741.27154365 69176.31261707
 70590.30119684 74088.07056824 70523.86371319 71638.65086703
 78298.33579674 70620.43607771]
Mean: 71101.4090196249
Standard deviation: 2990.111267490658
```

In [38]:
```
# Lets compute the cross validation scores for linear regression model just to
be safe.
linear_reg_scores = cross_val_score(linear_reg, housing_prepared, housing_labe
ls, scoring="neg_mean_squared_error", cv=10)
linear_reg_rmse_scores = np.sqrt(-linear_reg_scores)
display_scores(linear_reg_rmse_scores)
# We have confirmed that the decision tree regressor model is overfitting so m
uch its performance is worse.
```

```
Scores: [66782.73843989 66960.118071   70347.95244419 74739.57052552
 68031.13388938 71193.84183426 64969.63056405 68281.61137997
 71552.91566558 67665.10082067]
Mean: 69052.46136345083
Standard deviation: 2731.6740017983434
```

In [39]:
```
# Let's try one more model
# Random Forest Regressor
# - This works by training many Decision Trees on random subsets of the featur
es, then averaging out their predictions.

# Train the model
from sklearn.ensemble import RandomForestRegressor
random_forest_reg = RandomForestRegressor()
random_forest_reg.fit(housing_prepared, housing_labels)

# Test the model on the training set
housing_predictions = random_forest_reg.predict(housing_prepared)
random_forest_mse = mean_squared_error(housing_labels, housing_predictions)
random_forest_rmse = np.sqrt(random_forest_mse)
print(random_forest_rmse)
```

```
18663.248064432224
```

In [55]:
```
# This looks a lot better, let's look at the cross validation scores
```

In [40]:
```
# Test the random forest model on the validation set
scores = cross_val_score(random_forest_reg, housing_prepared, housing_labels,
scoring="neg_mean_squared_error", cv=10)
random_forest_rmse_scores = np.sqrt(-scores)

display_scores(random_forest_rmse_scores)
```

```
Scores: [49568.3417132  47315.08712098 49854.13492773 52209.914778
 49528.74754482 53516.14223867 48688.10995098 47887.98583279
 52876.88005253 50272.84261808]
Mean: 50171.81867777926
Standard deviation: 1977.9470620493362
```

In [57]:
```
# Notice that the forest_rmse (training set error) is much lower than the scor
es from the cross-validation.
# This means the model is still overfitting the training set.

# Possible solutions for overfitting include:
# simplify the model,, constraint it (regularization), or get a lot more train
ing data
```

In [54]:
```
# Let's try the Support Vector Machine Regressor model
from sklearn.svm import SVR

# Train the model
svm_reg = SVR(kernel="linear")
svm_reg.fit(housing_prepared, housing_labels)

# Test the model on the training set
housing_predictions = svm_reg.predict(housing_prepared)
svm_mse = mean_squared_error(housing_labels, housing_predictions)
svm_rmse = np.sqrt(svm_mse)
print(svm_rmse)
```

```
111094.6308539982
```

In [58]:
```
# This looks pretty bad, let's look at the cross validaiton scores
```

In [59]:
```
# Test the SVM model on the validation set
scores = cross_val_score(svm_reg, housing_prepared, housing_labels, scoring="n
eg_mean_squared_error", cv=10)
svm_reg_rmse_scores = np.sqrt(-scores)

display_scores(svm_reg_rmse_scores)
```

```
Scores: [105342.09141998 112489.24624123 110092.35042753 113403.22892482
 110638.90119657 115675.8320024  110703.56887243 114476.89008206
 113756.17971227 111520.1120808 ]
Mean: 111809.84009600841
Standard deviation: 2762.393664321567
```

In [60]:
```
# This still looks pretty awful. Let's try a different SVM Kernel and mess aro
und with the gamma, C, and epsilon values

# gamma {'scale', 'auto'} is the Kernel coefficient for 'rbf', 'poly', and 'si
gmoid'
# C {float, default = 1.0} is the regularization parameter
# epsilon {float, default = 0.1} is the epsilon in the SVR model

# See https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html f
or details

# default: svm_reg = SVR(kernel="rbf", gamma='scale', C=1.0, epsilon=0.1)
```

In [61]:
```
# Support Vector Machine Regressor - Continued

# Train the model
svm_reg = SVR(kernel="rbf", gamma='scale', C=3.0, epsilon=0.1)
svm_reg.fit(housing_prepared, housing_labels)

# Test the model on the training set
housing_predictions = svm_reg.predict(housing_prepared)
svm_mse = mean_squared_error(housing_labels, housing_predictions)
svm_rmse = np.sqrt(svm_mse)
print(svm_rmse)

# Test the SVM model on the validation set
scores = cross_val_score(svm_reg, housing_prepared, housing_labels, scoring="n
eg_mean_squared_error", cv=10)
svm_reg_rmse_scores = np.sqrt(-scores)

display_scores(svm_reg_rmse_scores)
```

```
117846.60742277169
Scores: [110779.4889748  118876.25647598 116314.6702655  119791.93555743
 116947.42903547 121672.7246191  116952.82703015 120842.75274804
 119700.49689316 117370.94945319]
Mean: 117924.95310528048
Standard deviation: 2931.9050857954962
```

In [62]:
```
# The SVM results do not look good, maybe this isn't the best model for this t
ask.
# We could still use GridSearch to mess around with the gamma, C, and epsilon
 values for further exploration,
# see exericse solutions below
```

In [63]:
```
## Save Models:
import joblib

joblib.dump(linear_reg, "linear_regression_model.pkl") # Save the Linear Regression Model
joblib.dump(decision_tree_reg, "decision_tree_regressor_model.pkl") # Save the Decision Tree Model
joblib.dump(random_forest_reg, "random_forest_regressor_model.pkl") # Save the Random Forest Model
joblib.dump(svm_reg, "support_vector_machine_regressor_model.pkl") # Save the Support Vector Machine Model

## Load Models:
# lin_reg_model_loaded = joblib.load("lin_reg_model.pkl") # Load the Linear Regression Model
```

Out[63]:
```
['support_vector_machine_regressor_model.pkl']
```

Fine-Tune Your Model

Grid Search CV to experiment with hyperparameter combinations on RandomForestRegressor

In [81]:
```
from sklearn.model_selection import GridSearchCV

param_grid = [
    {
        'n_estimators': [3, 10, 30, 100],
         'max_features': [2, 4, 6, 8, 10]
    },
    {
        'bootstrap': [False],
        'n_estimators': [3, 10, 30, 100],
        'max_features': [2, 3, 4, 6, 8, 10]
    },
]

random_forest_reg = RandomForestRegressor()

grid_search = GridSearchCV(random_forest_reg, param_grid, cv=5, scoring='neg_mean_squared_error', return_train_score=True)

grid_search.fit(housing_prepared, housing_labels)
```

Out[81]:
```
GridSearchCV(cv=5, estimator=RandomForestRegressor(),
             param_grid=[{'max_features': [2, 4, 6, 8, 10],
                          'n_estimators': [3, 10, 30, 100]},
                         {'bootstrap': [False],
                          'max_features': [2, 3, 4, 6, 8, 10],
                          'n_estimators': [3, 10, 30, 100]}],
             return_train_score=True, scoring='neg_mean_squared_error')
```

In [82]:
```
# Explore best parameter combinations:
print(grid_search.best_params_)
```

```
{'bootstrap': False, 'max_features': 6, 'n_estimators': 100}
```

In [83]:
```
print(grid_search.best_estimator_)
```

```
RandomForestRegressor(bootstrap=False, max_features=6)
```

In [84]:
```
# Grid Search Evaluation Scores:
cvres = grid_search.cv_results_
for mean_score, params in zip(cvres["mean_test_score"], cvres["params"]):
    print(np.sqrt(-mean_score), params)
```

```
63211.03853672407 {'max_features': 2, 'n_estimators': 3}
56034.5147579656 {'max_features': 2, 'n_estimators': 10}
52928.42291110197 {'max_features': 2, 'n_estimators': 30}
51964.88145085111 {'max_features': 2, 'n_estimators': 100}
60530.25010864527 {'max_features': 4, 'n_estimators': 3}
52844.69735283786 {'max_features': 4, 'n_estimators': 10}
50865.44552614693 {'max_features': 4, 'n_estimators': 30}
49731.23013985973 {'max_features': 4, 'n_estimators': 100}
59475.05098749464 {'max_features': 6, 'n_estimators': 3}
52608.673755902215 {'max_features': 6, 'n_estimators': 10}
49955.112006627714 {'max_features': 6, 'n_estimators': 30}
49265.20432866328 {'max_features': 6, 'n_estimators': 100}
59066.05982549296 {'max_features': 8, 'n_estimators': 3}
52433.81728973791 {'max_features': 8, 'n_estimators': 10}
50045.26008148687 {'max_features': 8, 'n_estimators': 30}
49409.23410830111 {'max_features': 8, 'n_estimators': 100}
58915.383393140415 {'max_features': 10, 'n_estimators': 3}
52168.0389061894 {'max_features': 10, 'n_estimators': 10}
50162.13332003433 {'max_features': 10, 'n_estimators': 30}
49557.03223877481 {'max_features': 10, 'n_estimators': 100}
62719.56086662608 {'bootstrap': False, 'max_features': 2, 'n_estimators': 3}
53807.18477738074 {'bootstrap': False, 'max_features': 2, 'n_estimators': 10}
51953.27604104431 {'bootstrap': False, 'max_features': 2, 'n_estimators': 30}
50833.915404223924 {'bootstrap': False, 'max_features': 2, 'n_estimators': 10
0}
61183.1618441438 {'bootstrap': False, 'max_features': 3, 'n_estimators': 3}
52676.95256494929 {'bootstrap': False, 'max_features': 3, 'n_estimators': 10}
50150.466695685405 {'bootstrap': False, 'max_features': 3, 'n_estimators': 3
0}
49651.39479392064 {'bootstrap': False, 'max_features': 3, 'n_estimators': 10
0}
58213.654152968265 {'bootstrap': False, 'max_features': 4, 'n_estimators': 3}
52057.20332564132 {'bootstrap': False, 'max_features': 4, 'n_estimators': 10}
49640.096935294656 {'bootstrap': False, 'max_features': 4, 'n_estimators': 3
0}
48834.2497568854 {'bootstrap': False, 'max_features': 4, 'n_estimators': 100}
56989.99649358026 {'bootstrap': False, 'max_features': 6, 'n_estimators': 3}
51216.24188644808 {'bootstrap': False, 'max_features': 6, 'n_estimators': 10}
49050.437979995026 {'bootstrap': False, 'max_features': 6, 'n_estimators': 3
0}
48537.192628134784 {'bootstrap': False, 'max_features': 6, 'n_estimators': 10
0}
57679.48106694756 {'bootstrap': False, 'max_features': 8, 'n_estimators': 3}
51160.97696595444 {'bootstrap': False, 'max_features': 8, 'n_estimators': 10}
49219.96743746264 {'bootstrap': False, 'max_features': 8, 'n_estimators': 30}
48668.13727939909 {'bootstrap': False, 'max_features': 8, 'n_estimators': 10
0}
57739.9865698914 {'bootstrap': False, 'max_features': 10, 'n_estimators': 3}
51456.748143697485 {'bootstrap': False, 'max_features': 10, 'n_estimators': 1
0}
50003.27761288673 {'bootstrap': False, 'max_features': 10, 'n_estimators': 3
0}
49362.923772717426 {'bootstrap': False, 'max_features': 10, 'n_estimators': 1
00}
```

Randomized Search CV to experiment with hyperparameter combinations on RandomForestRegressor

In [85]:
```
# This is used when you need to randomly sample large hyperparamter spaces

from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import randint

param_distributions = {
    'n_estimators': randint(low=1, high=200),
    'max_features': randint(low=1, high=8),
}

random_forest_reg = RandomForestRegressor(random_state=42)

random_search = RandomizedSearchCV(random_forest_reg,
                                   param_distributions=param_distributions,
                                   n_iter=10,
                                   cv=5,
                                   scoring='neg_mean_squared_error',
                                   random_state=42)

random_search.fit(housing_prepared, housing_labels)
```

Out[85]:
```
RandomizedSearchCV(cv=5, estimator=RandomForestRegressor(random_state=42),
                   param_distributions={'max_features': <scipy.stats._distn_i
nfrastructure.rv_frozen object at 0x000001B112224C40>,
                                        'n_estimators': <scipy.stats._distn_i
nfrastructure.rv_frozen object at 0x000001B11247EF40>},
                   random_state=42, scoring='neg_mean_squared_error')
```

In [100]:
```
# Explore best random search parameter combinations:
print(random_search.best_params_)
```

```
{'max_features': 7, 'n_estimators': 180}
```

In [87]:
```
print(random_search.best_estimator_)
```

```
RandomForestRegressor(max_features=7, n_estimators=180, random_state=42)
```

In [88]:
```
# Random Search Evaluation Scores:
cvres = random_search.cv_results_
for mean_score, params in zip(cvres["mean_test_score"], cvres["params"]):
    print(np.sqrt(-mean_score), params)
```

```
49150.70756927707 {'max_features': 7, 'n_estimators': 180}
51389.889203389284 {'max_features': 5, 'n_estimators': 15}
50796.155224308866 {'max_features': 3, 'n_estimators': 72}
50835.13360315349 {'max_features': 5, 'n_estimators': 21}
49280.9449827171 {'max_features': 7, 'n_estimators': 122}
50774.90662363929 {'max_features': 3, 'n_estimators': 75}
50682.78888164288 {'max_features': 3, 'n_estimators': 88}
49608.99608105296 {'max_features': 5, 'n_estimators': 100}
50473.61930350219 {'max_features': 3, 'n_estimators': 150}
64429.84143294435 {'max_features': 5, 'n_estimators': 2}
```

In [89]:
```
# In this case, the Random Search didn't find better parameters than the Grid
 Search
```

Analyze the Best Models and Their Error

In [90]:
```
# Let's look at the relative importance of each attribute for making accurate
 predictions
feature_importances = grid_search.best_estimator_.feature_importances_
random_tree_reg_feature_importances = feature_importances # Note we need this
 later in Exercise 3
extra_attribs = ["rooms_per_household","pop_per_hhold","bedrooms_per_room"]
cat_encoder = full_pipeline.named_transformers_["cat"]
cat_one_hot_attribs = list(cat_encoder.categories_[0])
attributes = numerical_attribs + extra_attribs + cat_one_hot_attribs
print(sorted(zip(feature_importances, attributes), reverse=True))

# With this info we can see the relative importance of each attribute for maki
ng an accurate prediction
# and which are less useful that we should maybe drop in the future.
# Ex. it appears that the <1H OCEAN category is the only useful ocean_proximit
y category
```

```
[(0.32820383909931383, 'median_income'), (0.143077997543386, 'INLAND'), (0.10
804589057862227, 'pop_per_hhold'), (0.08119879731256048, 'longitude'), (0.080
087908597283, 'bedrooms_per_room'), (0.07499945447052753, 'latitude'), (0.054
69985129854927, 'rooms_per_household'), (0.04263139319651873, 'housing_median
_age'), (0.016806938836952563, 'total_rooms'), (0.016565035217440644, 'popula
tion'), (0.01564709777088628, 'households'), (0.015601740074349257, 'total_be
drooms'), (0.01310521958261995, '<1H OCEAN'), (0.005020450659939618, 'NEAR OC
EAN'), (0.004234867399445356, 'NEAR BAY'), (7.351836160527607e-05, 'ISLAND')]
```

Evaluate Your System on the Test Set

In [47]:
```
final_model = grid_search.best_estimator_

X_test = strat_test_set.drop("median_house_value", axis=1)
y_test = strat_test_set["median_house_value"].copy()

X_test_prepared = full_pipeline.transform(X_test)

final_predictions = final_model.predict(X_test_prepared)

final_mse = mean_squared_error(y_test, final_predictions)
final_rmse = np.sqrt(final_mse)
print(final_rmse)
# The final_rmse (generalization error) may not be enough to convince you to l
aunch your model.
```

```
46372.31934924472
```

In [48]:
```
# In order to have an idea of how precise this estimate is, let us do the foll
owing:

# Compute a 95% confidence interval for the generalization of the error to get
an idea of how precise our estimate is
from scipy import stats
confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
generalizedError = np.sqrt(stats.t.interval(confidence, len(squared_errors) -
1, loc=squared_errors.mean(), scale=stats.sem(squared_errors)))
print(generalizedError)
# We see that our system is not better than the experts' price estimates, but
 it may still be good enough to launch
# to free up more of their time for other productive tasks.
```

```
[44377.82362471 48284.49827834]
```

Exerice Solutions

1. Try SVM Regressor with various hyper parameters

In [69]:
```
param_grid = [
        {
            'kernel': ['linear'],
            'C': [10., 30., 100., 300., 1000., 3000., 10000., 30000.0]
        },
        {
            'kernel': ['rbf'],
            'C': [1.0, 3.0, 10., 30., 100., 300., 1000.0],
            'gamma': [0.01, 0.03, 0.1, 0.3, 1.0, 3.0]
        },
    ]

svm_reg = SVR()
grid_search = GridSearchCV(svm_reg, param_grid, cv=5, scoring='neg_mean_square
d_error', verbose=2)
grid_search.fit(housing_prepared, housing_labels)
```

```
Fitting 5 folds for each of 50 candidates, totalling 250 fits
[CV] C=10.0, kernel=linear ...........................................

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worke
rs.

[CV] ............................ C=10.0, kernel=linear, total=   3.6s
[CV] C=10.0, kernel=linear ...........................................

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    3.5s remaining:    0.
0s
```

```
[CV] ........................... C=10.0, kernel=linear, total=   3.6s
[CV] C=10.0, kernel=linear .....................................
[CV] ........................... C=10.0, kernel=linear, total=   3.5s
[CV] C=10.0, kernel=linear .....................................
[CV] ........................... C=10.0, kernel=linear, total=   3.5s
[CV] C=10.0, kernel=linear .....................................
[CV] ........................... C=10.0, kernel=linear, total=   3.5s
[CV] C=30.0, kernel=linear .....................................
[CV] ........................... C=30.0, kernel=linear, total=   3.5s
[CV] C=30.0, kernel=linear .....................................
[CV] ........................... C=30.0, kernel=linear, total=   3.5s
[CV] C=30.0, kernel=linear .....................................
[CV] ........................... C=30.0, kernel=linear, total=   3.7s
[CV] C=30.0, kernel=linear .....................................
[CV] ........................... C=30.0, kernel=linear, total=   3.6s
[CV] C=30.0, kernel=linear .....................................
[CV] ........................... C=30.0, kernel=linear, total=   3.6s
[CV] C=100.0, kernel=linear ....................................
[CV] ........................... C=100.0, kernel=linear, total=   3.5s
[CV] C=100.0, kernel=linear ....................................
[CV] ........................... C=100.0, kernel=linear, total=   3.5s
[CV] C=100.0, kernel=linear ....................................
[CV] ........................... C=100.0, kernel=linear, total=   3.6s
[CV] C=100.0, kernel=linear ....................................
[CV] ........................... C=100.0, kernel=linear, total=   3.5s
[CV] C=100.0, kernel=linear ....................................
[CV] ........................... C=100.0, kernel=linear, total=   3.5s
[CV] C=300.0, kernel=linear ....................................
[CV] ........................... C=300.0, kernel=linear, total=   3.6s
[CV] C=300.0, kernel=linear ....................................
[CV] ........................... C=300.0, kernel=linear, total=   3.6s
[CV] C=300.0, kernel=linear ....................................
[CV] ........................... C=300.0, kernel=linear, total=   3.7s
[CV] C=300.0, kernel=linear ....................................
[CV] ........................... C=300.0, kernel=linear, total=   3.6s
[CV] C=300.0, kernel=linear ....................................
[CV] ........................... C=300.0, kernel=linear, total=   3.5s
[CV] C=1000.0, kernel=linear ...................................
[CV] ........................... C=1000.0, kernel=linear, total=   3.7s
[CV] C=1000.0, kernel=linear ...................................
[CV] ........................... C=1000.0, kernel=linear, total=   3.7s
[CV] C=1000.0, kernel=linear ...................................
[CV] ........................... C=1000.0, kernel=linear, total=   3.7s
[CV] C=1000.0, kernel=linear ...................................
[CV] ........................... C=1000.0, kernel=linear, total=   3.8s
[CV] C=1000.0, kernel=linear ...................................
[CV] ........................... C=1000.0, kernel=linear, total=   3.7s
[CV] C=3000.0, kernel=linear ...................................
[CV] ........................... C=3000.0, kernel=linear, total=   4.0s
[CV] C=3000.0, kernel=linear ...................................
[CV] ........................... C=3000.0, kernel=linear, total=   3.9s
[CV] C=3000.0, kernel=linear ...................................
[CV] ........................... C=3000.0, kernel=linear, total=   4.1s
[CV] C=3000.0, kernel=linear ...................................
[CV] ........................... C=3000.0, kernel=linear, total=   4.1s
[CV] C=3000.0, kernel=linear ...................................
[CV] ........................... C=3000.0, kernel=linear, total=   3.9s
```

```
[CV] C=10000.0, kernel=linear ........................................
[CV] ......................... C=10000.0, kernel=linear, total=   5.3s
[CV] C=10000.0, kernel=linear ........................................
[CV] ......................... C=10000.0, kernel=linear, total=   5.4s
[CV] C=10000.0, kernel=linear ........................................
[CV] ......................... C=10000.0, kernel=linear, total=   5.4s
[CV] C=10000.0, kernel=linear ........................................
[CV] ......................... C=10000.0, kernel=linear, total=   5.0s
[CV] C=10000.0, kernel=linear ........................................
[CV] ......................... C=10000.0, kernel=linear, total=   4.7s
[CV] C=30000.0, kernel=linear ........................................
[CV] ......................... C=30000.0, kernel=linear, total=   8.5s
[CV] C=30000.0, kernel=linear ........................................
[CV] ......................... C=30000.0, kernel=linear, total=   8.6s
[CV] C=30000.0, kernel=linear ........................................
[CV] ......................... C=30000.0, kernel=linear, total=   9.1s
[CV] C=30000.0, kernel=linear ........................................
[CV] ......................... C=30000.0, kernel=linear, total=   8.7s
[CV] C=30000.0, kernel=linear ........................................
[CV] ......................... C=30000.0, kernel=linear, total=   7.6s
[CV] C=1.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.03, kernel=rbf, total=   5.9s
[CV] C=1.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=1.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=1.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.1, kernel=rbf, total=   6.0s
[CV] C=1.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=1.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=1.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=1.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=1.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=1.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=1.0, gamma=0.3, kernel=rbf ....................................
```

```
[CV] ..................... C=1.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=1.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=1.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=1.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=1.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=1.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=1.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=1.0, kernel=rbf, total=   5.5s
[CV] C=1.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=3.0, kernel=rbf, total=   6.2s
[CV] C=1.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=3.0, kernel=rbf, total=   6.1s
[CV] C=1.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=3.0, kernel=rbf, total=   6.1s
[CV] C=1.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=3.0, kernel=rbf, total=   6.2s
[CV] C=1.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=1.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=3.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.01, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.03, kernel=rbf, total=   5.9s
[CV] C=3.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.03, kernel=rbf ...................................
[CV] .................... C=3.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.1, kernel=rbf, total=   5.8s
[CV] C=3.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=3.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=3.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=3.0, gamma=0.1, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.1, kernel=rbf, total=   6.0s
[CV] C=3.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=3.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.3, kernel=rbf, total=   5.8s
```

```
[CV] C=3.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=3.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=3.0, gamma=0.3, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=3.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=3.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=3.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=3.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=3.0, gamma=1.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=3.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=3.0, kernel=rbf, total=   6.4s
[CV] C=3.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=3.0, kernel=rbf, total=   6.4s
[CV] C=3.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=3.0, kernel=rbf, total=   6.4s
[CV] C=3.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=3.0, kernel=rbf, total=   6.4s
[CV] C=3.0, gamma=3.0, kernel=rbf ....................................
[CV] ..................... C=3.0, gamma=3.0, kernel=rbf, total=   6.4s
[CV] C=10.0, gamma=0.01, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.01, kernel=rbf, total=   6.2s
[CV] C=10.0, gamma=0.01, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.01, kernel=rbf, total=   6.2s
[CV] C=10.0, gamma=0.01, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.01, kernel=rbf, total=   6.2s
[CV] C=10.0, gamma=0.01, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.01, kernel=rbf, total=   6.2s
[CV] C=10.0, gamma=0.01, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.01, kernel=rbf, total=   6.2s
[CV] C=10.0, gamma=0.03, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.03, kernel=rbf, total=   6.2s
[CV] C=10.0, gamma=0.03, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.03, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.03, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.03, kernel=rbf ..................................
[CV] .................. C=10.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.1, kernel=rbf ...................................
[CV] ................... C=10.0, gamma=0.1, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.1, kernel=rbf ...................................
[CV] ................... C=10.0, gamma=0.1, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.1, kernel=rbf ...................................
[CV] ................... C=10.0, gamma=0.1, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.1, kernel=rbf ...................................
[CV] ................... C=10.0, gamma=0.1, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.1, kernel=rbf ...................................
[CV] ................... C=10.0, gamma=0.1, kernel=rbf, total=   6.1s
[CV] C=10.0, gamma=0.3, kernel=rbf ...................................
```

```
[CV] .................... C=10.0, gamma=0.3, kernel=rbf, total=   5.9s
[CV] C=10.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=0.3, kernel=rbf, total=   6.0s
[CV] C=10.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=0.3, kernel=rbf, total=   6.0s
[CV] C=10.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=0.3, kernel=rbf, total=   5.9s
[CV] C=10.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=0.3, kernel=rbf, total=   5.9s
[CV] C=10.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=10.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=10.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=10.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=10.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=10.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=10.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=10.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=10.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=10.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=10.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=30.0, gamma=0.01, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.01, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.01, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.01, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.01, kernel=rbf, total=   6.2s
[CV] C=30.0, gamma=0.01, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.01, kernel=rbf, total=   6.2s
[CV] C=30.0, gamma=0.03, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.03, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.03, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.03, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.03, kernel=rbf ..................................
[CV] ................... C=30.0, gamma=0.03, kernel=rbf, total=   6.1s
[CV] C=30.0, gamma=0.1, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.1, kernel=rbf, total=   6.0s
[CV] C=30.0, gamma=0.1, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.1, kernel=rbf, total=   6.0s
[CV] C=30.0, gamma=0.1, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.1, kernel=rbf, total=   6.0s
[CV] C=30.0, gamma=0.1, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.1, kernel=rbf, total=   6.0s
```

```
[CV] C=30.0, gamma=0.1, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.1, kernel=rbf, total=   6.0s
[CV] C=30.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=30.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=30.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.3, kernel=rbf, total=   5.9s
[CV] C=30.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.3, kernel=rbf, total=   5.9s
[CV] C=30.0, gamma=0.3, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=0.3, kernel=rbf, total=   5.9s
[CV] C=30.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=30.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=30.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=30.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=30.0, gamma=1.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=30.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=30.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=30.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=30.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=30.0, gamma=3.0, kernel=rbf ...................................
[CV] .................... C=30.0, gamma=3.0, kernel=rbf, total=   6.4s
[CV] C=100.0, gamma=0.01, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=100.0, gamma=0.01, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=100.0, gamma=0.01, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=100.0, gamma=0.01, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=100.0, gamma=0.01, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.01, kernel=rbf, total=   6.1s
[CV] C=100.0, gamma=0.03, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=100.0, gamma=0.03, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=100.0, gamma=0.03, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=100.0, gamma=0.03, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=100.0, gamma=0.03, kernel=rbf .................................
[CV] .................. C=100.0, gamma=0.03, kernel=rbf, total=   6.0s
[CV] C=100.0, gamma=0.1, kernel=rbf ..................................
[CV] ................... C=100.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=100.0, gamma=0.1, kernel=rbf ..................................
[CV] ................... C=100.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=100.0, gamma=0.1, kernel=rbf ..................................
```

```
[CV] .................. C=100.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=100.0, gamma=0.1, kernel=rbf ................................
[CV] .................. C=100.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=100.0, gamma=0.1, kernel=rbf ................................
[CV] .................. C=100.0, gamma=0.1, kernel=rbf, total=   5.9s
[CV] C=100.0, gamma=0.3, kernel=rbf ................................
[CV] .................. C=100.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=100.0, gamma=0.3, kernel=rbf ................................
[CV] .................. C=100.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=100.0, gamma=0.3, kernel=rbf ................................
[CV] .................. C=100.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=100.0, gamma=0.3, kernel=rbf ................................
[CV] .................. C=100.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=100.0, gamma=0.3, kernel=rbf ................................
[CV] .................. C=100.0, gamma=0.3, kernel=rbf, total=   5.8s
[CV] C=100.0, gamma=1.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=100.0, gamma=1.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=100.0, gamma=1.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=100.0, gamma=1.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=100.0, gamma=1.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=100.0, gamma=3.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=100.0, gamma=3.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=100.0, gamma=3.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=100.0, gamma=3.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=100.0, gamma=3.0, kernel=rbf ................................
[CV] .................. C=100.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=300.0, gamma=0.01, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=300.0, gamma=0.01, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=300.0, gamma=0.01, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=300.0, gamma=0.01, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=300.0, gamma=0.01, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.01, kernel=rbf, total=   6.0s
[CV] C=300.0, gamma=0.03, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.03, kernel=rbf, total=   5.8s
[CV] C=300.0, gamma=0.03, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.03, kernel=rbf, total=   5.9s
[CV] C=300.0, gamma=0.03, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.03, kernel=rbf, total=   5.8s
[CV] C=300.0, gamma=0.03, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.03, kernel=rbf, total=   5.8s
[CV] C=300.0, gamma=0.03, kernel=rbf ...............................
[CV] ................. C=300.0, gamma=0.03, kernel=rbf, total=   5.9s
[CV] C=300.0, gamma=0.1, kernel=rbf ................................
[CV] .................. C=300.0, gamma=0.1, kernel=rbf, total=   5.8s
```

```
[CV] C=300.0, gamma=0.1, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.1, kernel=rbf, total=   5.8s
[CV] C=300.0, gamma=0.1, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.1, kernel=rbf, total=   5.8s
[CV] C=300.0, gamma=0.1, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.1, kernel=rbf, total=   5.8s
[CV] C=300.0, gamma=0.1, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.1, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=0.3, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=0.3, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=0.3, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=0.3, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=0.3, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=1.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=1.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=1.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=1.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=1.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=300.0, gamma=3.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=300.0, gamma=3.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=300.0, gamma=3.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=300.0, gamma=3.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=300.0, gamma=3.0, kernel=rbf ..................................
[CV] ................... C=300.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=1000.0, gamma=0.01, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.01, kernel=rbf, total=   5.8s
[CV] C=1000.0, gamma=0.01, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.01, kernel=rbf, total=   5.8s
[CV] C=1000.0, gamma=0.01, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.01, kernel=rbf, total=   5.8s
[CV] C=1000.0, gamma=0.01, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.01, kernel=rbf, total=   5.9s
[CV] C=1000.0, gamma=0.01, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.01, kernel=rbf, total=   5.8s
[CV] C=1000.0, gamma=0.03, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.03, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.03, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.03, kernel=rbf, total=   5.8s
[CV] C=1000.0, gamma=0.03, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.03, kernel=rbf, total=   5.8s
[CV] C=1000.0, gamma=0.03, kernel=rbf ................................
[CV] ................. C=1000.0, gamma=0.03, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.03, kernel=rbf ................................
```

```
[CV] ................. C=1000.0, gamma=0.03, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.1, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.1, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.1, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.1, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.1, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.1, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.1, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.1, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.1, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.1, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.3, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.3, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.3, kernel=rbf, total=   5.6s
[CV] C=1000.0, gamma=0.3, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=0.3, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.3, kernel=rbf, total=   5.6s
[CV] C=1000.0, gamma=0.3, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=0.3, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=1.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=1.0, kernel=rbf, total=   5.6s
[CV] C=1000.0, gamma=1.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=1.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=1.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=1.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=1.0, kernel=rbf, total=   5.7s
[CV] C=1000.0, gamma=3.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=1000.0, gamma=3.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=1000.0, gamma=3.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=1000.0, gamma=3.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=3.0, kernel=rbf, total=   6.3s
[CV] C=1000.0, gamma=3.0, kernel=rbf .................................
[CV] .................. C=1000.0, gamma=3.0, kernel=rbf, total=   6.3s

[Parallel(n_jobs=1)]: Done 250 out of 250 | elapsed: 23.8min finished
```

Out[69]:
```
GridSearchCV(cv=5, estimator=SVR(),
             param_grid=[{'C': [10.0, 30.0, 100.0, 300.0, 1000.0, 3000.0,
                                10000.0, 30000.0],
                          'kernel': ['linear']},
                         {'C': [1.0, 3.0, 10.0, 30.0, 100.0, 300.0, 1000.0],
                          'gamma': [0.01, 0.03, 0.1, 0.3, 1.0, 3.0],
                          'kernel': ['rbf']}],
             scoring='neg_mean_squared_error', verbose=2)
```

2. Try replacing GridSearchCV with RandomizedSearchCV

In [71]:
```
# Let's do this for the SVM Regressor

from scipy.stats import expon, reciprocal

# see https://docs.scipy.org/doc/scipy/reference/stats.html
# for `expon()` and `reciprocal()` documentation and more probability distribu
tion functions.

param_distribs = {
        'kernel': ['linear', 'rbf'],
        'C': reciprocal(20, 200000),
        'gamma': expon(scale=1.0),
    }

svm_reg = SVR()
random_search = RandomizedSearchCV(svm_reg,
                                   param_distributions=param_distribs,
                                   n_iter=50,
                                   cv=5,
                                   scoring='neg_mean_squared_error',
                                   verbose=2,
                                   random_state=42)

random_search.fit(housing_prepared, housing_labels)
```

```
Fitting 5 folds for each of 50 candidates, totalling 250 fits
[CV] C=629.782329591372, gamma=3.010121430917521, kernel=linear ......

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worke
rs.

[CV]  C=629.782329591372, gamma=3.010121430917521, kernel=linear, total=   3.
6s
[CV] C=629.782329591372, gamma=3.010121430917521, kernel=linear ......

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    3.5s remaining:    0.
0s
```

```
[CV]  C=629.782329591372, gamma=3.010121430917521, kernel=linear, total=   3.
6s
[CV] C=629.782329591372, gamma=3.010121430917521, kernel=linear ......
[CV]  C=629.782329591372, gamma=3.010121430917521, kernel=linear, total=   3.
6s
[CV] C=629.782329591372, gamma=3.010121430917521, kernel=linear ......
[CV]  C=629.782329591372, gamma=3.010121430917521, kernel=linear, total=   3.
7s
[CV] C=629.782329591372, gamma=3.010121430917521, kernel=linear ......
[CV]  C=629.782329591372, gamma=3.010121430917521, kernel=linear, total=   3.
7s
[CV] C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf ......
[CV]  C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf, total=   7.
3s
[CV] C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf ......
[CV]  C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf, total=   7.
5s
[CV] C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf ......
[CV]  C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf, total=   7.
4s
[CV] C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf ......
[CV]  C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf, total=   7.
5s
[CV] C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf ......
[CV]  C=26290.206464300216, gamma=0.9084469696321253, kernel=rbf, total=   7.
6s
[CV] C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf .....
[CV]  C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf, total=
6.0s
[CV] C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf .....
[CV]  C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf, total=
6.0s
[CV] C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf .....
[CV]  C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf, total=
6.0s
[CV] C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf .....
[CV]  C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf, total=
6.0s
[CV] C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf .....
[CV]  C=84.14107900575871, gamma=0.059838768608680676, kernel=rbf, total=
6.0s
[CV] C=432.37884813148855, gamma=0.15416196746656105, kernel=linear ..
[CV]  C=432.37884813148855, gamma=0.15416196746656105, kernel=linear, total=
3.5s
[CV] C=432.37884813148855, gamma=0.15416196746656105, kernel=linear ..
[CV]  C=432.37884813148855, gamma=0.15416196746656105, kernel=linear, total=
3.5s
[CV] C=432.37884813148855, gamma=0.15416196746656105, kernel=linear ..
[CV]  C=432.37884813148855, gamma=0.15416196746656105, kernel=linear, total=
3.7s
[CV] C=432.37884813148855, gamma=0.15416196746656105, kernel=linear ..
[CV]  C=432.37884813148855, gamma=0.15416196746656105, kernel=linear, total=
3.6s
[CV] C=432.37884813148855, gamma=0.15416196746656105, kernel=linear ..
[CV]  C=432.37884813148855, gamma=0.15416196746656105, kernel=linear, total=
3.6s
[CV] C=24.17508294611391, gamma=3.503557475158312, kernel=rbf .......
```

```
[CV]  C=24.17508294611391, gamma=3.503557475158312, kernel=rbf, total=   6.6s
[CV] C=24.17508294611391, gamma=3.503557475158312, kernel=rbf ........
[CV]  C=24.17508294611391, gamma=3.503557475158312, kernel=rbf, total=   6.6s
[CV] C=24.17508294611391, gamma=3.503557475158312, kernel=rbf ........
[CV]  C=24.17508294611391, gamma=3.503557475158312, kernel=rbf, total=   6.7s
[CV] C=24.17508294611391, gamma=3.503557475158312, kernel=rbf ........
[CV]  C=24.17508294611391, gamma=3.503557475158312, kernel=rbf, total=   6.6s
[CV] C=24.17508294611391, gamma=3.503557475158312, kernel=rbf ........
[CV]  C=24.17508294611391, gamma=3.503557475158312, kernel=rbf, total=   6.7s
[CV] C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf ...
[CV]  C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf, total=
5.8s
[CV] C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf ...
[CV]  C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf, total=
5.8s
[CV] C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf ...
[CV]  C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf, total=
5.8s
[CV] C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf ...
[CV]  C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf, total=
5.8s
[CV] C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf ...
[CV]  C=113564.03940586245, gamma=0.0007790692366582295, kernel=rbf, total=
5.9s
[CV] C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf ......
[CV]  C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf, total=   5.
8s
[CV] C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf ......
[CV]  C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf, total=   5.
8s
[CV] C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf ......
[CV]  C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf, total=   5.
8s
[CV] C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf ......
[CV]  C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf, total=   5.
8s
[CV] C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf ......
[CV]  C=108.30488238805073, gamma=0.3627537294604771, kernel=rbf, total=   5.
8s
[CV] C=21.344953672647435, gamma=0.023332523598323388, kernel=linear .
[CV]  C=21.344953672647435, gamma=0.023332523598323388, kernel=linear, total=
3.6s
[CV] C=21.344953672647435, gamma=0.023332523598323388, kernel=linear .
[CV]  C=21.344953672647435, gamma=0.023332523598323388, kernel=linear, total=
3.6s
[CV] C=21.344953672647435, gamma=0.023332523598323388, kernel=linear .
[CV]  C=21.344953672647435, gamma=0.023332523598323388, kernel=linear, total=
3.5s
[CV] C=21.344953672647435, gamma=0.023332523598323388, kernel=linear .
[CV]  C=21.344953672647435, gamma=0.023332523598323388, kernel=linear, total=
3.6s
[CV] C=21.344953672647435, gamma=0.023332523598323388, kernel=linear .
[CV]  C=21.344953672647435, gamma=0.023332523598323388, kernel=linear, total=
3.5s
[CV] C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf ......
[CV]  C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf, total=   5.
7s
```

```
[CV] C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf ......
[CV]  C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf, total=   5.
7s
[CV] C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf ......
[CV]  C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf, total=   5.
7s
[CV] C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf ......
[CV]  C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf, total=   5.
7s
[CV] C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf ......
[CV]  C=5603.270317432516, gamma=0.15023452872733867, kernel=rbf, total=   5.
6s
[CV] C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf .....
[CV]  C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf, total=  1
5.3s
[CV] C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf .....
[CV]  C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf, total=  1
6.3s
[CV] C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf .....
[CV]  C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf, total=  1
8.7s
[CV] C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf .....
[CV]  C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf, total=  1
4.9s
[CV] C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf .....
[CV]  C=157055.10989448498, gamma=0.26497040005002437, kernel=rbf, total=  1
6.6s
[CV] C=27652.464358739708, gamma=0.2227358621286903, kernel=linear ...
[CV]  C=27652.464358739708, gamma=0.2227358621286903, kernel=linear, total=
7.9s
[CV] C=27652.464358739708, gamma=0.2227358621286903, kernel=linear ...
[CV]  C=27652.464358739708, gamma=0.2227358621286903, kernel=linear, total=
8.3s
[CV] C=27652.464358739708, gamma=0.2227358621286903, kernel=linear ...
[CV]  C=27652.464358739708, gamma=0.2227358621286903, kernel=linear, total=
8.7s
[CV] C=27652.464358739708, gamma=0.2227358621286903, kernel=linear ...
[CV]  C=27652.464358739708, gamma=0.2227358621286903, kernel=linear, total=
7.9s
[CV] C=27652.464358739708, gamma=0.2227358621286903, kernel=linear ...
[CV]  C=27652.464358739708, gamma=0.2227358621286903, kernel=linear, total=
7.0s
[CV] C=171377.39570378003, gamma=0.628789100540856, kernel=linear ....
[CV]  C=171377.39570378003, gamma=0.628789100540856, kernel=linear, total=  3
4.6s
[CV] C=171377.39570378003, gamma=0.628789100540856, kernel=linear ....
[CV]  C=171377.39570378003, gamma=0.628789100540856, kernel=linear, total=  2
6.6s
[CV] C=171377.39570378003, gamma=0.628789100540856, kernel=linear ....
[CV]  C=171377.39570378003, gamma=0.628789100540856, kernel=linear, total=  3
3.2s
[CV] C=171377.39570378003, gamma=0.628789100540856, kernel=linear ....
[CV]  C=171377.39570378003, gamma=0.628789100540856, kernel=linear, total=  2
8.6s
[CV] C=171377.39570378003, gamma=0.628789100540856, kernel=linear ....
[CV]  C=171377.39570378003, gamma=0.628789100540856, kernel=linear, total=  2
2.8s
```

```
[CV] C=5385.293820172355, gamma=0.18696125197741642, kernel=linear ...
[CV]  C=5385.293820172355, gamma=0.18696125197741642, kernel=linear, total=
4.3s
[CV] C=5385.293820172355, gamma=0.18696125197741642, kernel=linear ...
[CV]  C=5385.293820172355, gamma=0.18696125197741642, kernel=linear, total=
4.4s
[CV] C=5385.293820172355, gamma=0.18696125197741642, kernel=linear ...
[CV]  C=5385.293820172355, gamma=0.18696125197741642, kernel=linear, total=
4.5s
[CV] C=5385.293820172355, gamma=0.18696125197741642, kernel=linear ...
[CV]  C=5385.293820172355, gamma=0.18696125197741642, kernel=linear, total=
4.3s
[CV] C=5385.293820172355, gamma=0.18696125197741642, kernel=linear ...
[CV]  C=5385.293820172355, gamma=0.18696125197741642, kernel=linear, total=
4.4s
[CV] C=22.59903216621323, gamma=2.850796878935603, kernel=rbf ........
[CV]  C=22.59903216621323, gamma=2.850796878935603, kernel=rbf, total=   6.1s
[CV] C=22.59903216621323, gamma=2.850796878935603, kernel=rbf ........
[CV]  C=22.59903216621323, gamma=2.850796878935603, kernel=rbf, total=   6.1s
[CV] C=22.59903216621323, gamma=2.850796878935603, kernel=rbf ........
[CV]  C=22.59903216621323, gamma=2.850796878935603, kernel=rbf, total=   6.3s
[CV] C=22.59903216621323, gamma=2.850796878935603, kernel=rbf ........
[CV]  C=22.59903216621323, gamma=2.850796878935603, kernel=rbf, total=   6.2s
[CV] C=22.59903216621323, gamma=2.850796878935603, kernel=rbf ........
[CV]  C=22.59903216621323, gamma=2.850796878935603, kernel=rbf, total=   6.3s
[CV] C=34246.75194632794, gamma=0.3632878599687583, kernel=linear ....
[CV]  C=34246.75194632794, gamma=0.3632878599687583, kernel=linear, total=
9.2s
[CV] C=34246.75194632794, gamma=0.3632878599687583, kernel=linear ....
[CV]  C=34246.75194632794, gamma=0.3632878599687583, kernel=linear, total=
9.1s
[CV] C=34246.75194632794, gamma=0.3632878599687583, kernel=linear ....
[CV]  C=34246.75194632794, gamma=0.3632878599687583, kernel=linear, total=
9.8s
[CV] C=34246.75194632794, gamma=0.3632878599687583, kernel=linear ....
[CV]  C=34246.75194632794, gamma=0.3632878599687583, kernel=linear, total=
9.3s
[CV] C=34246.75194632794, gamma=0.3632878599687583, kernel=linear ....
[CV]  C=34246.75194632794, gamma=0.3632878599687583, kernel=linear, total=
8.5s
[CV] C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf .......
[CV]  C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf, total=   5.8
s
[CV] C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf .......
[CV]  C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf, total=   5.7
s
[CV] C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf .......
[CV]  C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf, total=   5.8
s
[CV] C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf .......
[CV]  C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf, total=   5.7
s
[CV] C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf .......
[CV]  C=167.7278956080511, gamma=0.2757870542258224, kernel=rbf, total=   5.8
s
[CV] C=61.54360542501371, gamma=0.6835472281341501, kernel=linear ....
[CV]  C=61.54360542501371, gamma=0.6835472281341501, kernel=linear, total=
```

```
3.5s
[CV] C=61.54360542501371, gamma=0.6835472281341501, kernel=linear ....
[CV]  C=61.54360542501371, gamma=0.6835472281341501, kernel=linear, total=
3.5s
[CV] C=61.54360542501371, gamma=0.6835472281341501, kernel=linear ....
[CV]  C=61.54360542501371, gamma=0.6835472281341501, kernel=linear, total=
3.5s
[CV] C=61.54360542501371, gamma=0.6835472281341501, kernel=linear ....
[CV]  C=61.54360542501371, gamma=0.6835472281341501, kernel=linear, total=
3.6s
[CV] C=61.54360542501371, gamma=0.6835472281341501, kernel=linear ....
[CV]  C=61.54360542501371, gamma=0.6835472281341501, kernel=linear, total=
3.5s
[CV] C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf .......
[CV]  C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf, total=   5.8
s
[CV] C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf .......
[CV]  C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf, total=   5.8
s
[CV] C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf .......
[CV]  C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf, total=   5.8
s
[CV] C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf .......
[CV]  C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf, total=   5.7
s
[CV] C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf .......
[CV]  C=98.73897389920914, gamma=0.4960365360493639, kernel=rbf, total=   5.8
s
[CV] C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf ......
[CV]  C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf, total=   5.
7s
[CV] C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf ......
[CV]  C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf, total=   5.
7s
[CV] C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf ......
[CV]  C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf, total=   5.
7s
[CV] C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf ......
[CV]  C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf, total=   5.
7s
[CV] C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf ......
[CV]  C=8935.505635947808, gamma=0.37354658165762367, kernel=rbf, total=   5.
7s
[CV] C=135.76775824842434, gamma=0.838636245624803, kernel=linear ....
[CV]  C=135.76775824842434, gamma=0.838636245624803, kernel=linear, total=
3.6s
[CV] C=135.76775824842434, gamma=0.838636245624803, kernel=linear ....
[CV]  C=135.76775824842434, gamma=0.838636245624803, kernel=linear, total=
3.5s
[CV] C=135.76775824842434, gamma=0.838636245624803, kernel=linear ....
[CV]  C=135.76775824842434, gamma=0.838636245624803, kernel=linear, total=
3.6s
[CV] C=135.76775824842434, gamma=0.838636245624803, kernel=linear ....
[CV]  C=135.76775824842434, gamma=0.838636245624803, kernel=linear, total=
3.6s
[CV] C=135.76775824842434, gamma=0.838636245624803, kernel=linear ....
[CV]  C=135.76775824842434, gamma=0.838636245624803, kernel=linear, total=
```

```
3.5s
[CV] C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf ......
[CV]  C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf, total= 1.5m
in
[CV] C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf ......
[CV]  C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf, total= 1.1m
in
[CV] C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf ......
[CV]  C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf, total= 1.0m
in
[CV] C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf ......
[CV]  C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf, total= 1.3m
in
[CV] C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf ......
[CV]  C=151136.20282548846, gamma=1.4922453771381408, kernel=rbf, total= 1.3m
in
[CV] C=761.4316758498783, gamma=2.6126336514161914, kernel=linear ....
[CV]  C=761.4316758498783, gamma=2.6126336514161914, kernel=linear, total=
3.7s
[CV] C=761.4316758498783, gamma=2.6126336514161914, kernel=linear ....
[CV]  C=761.4316758498783, gamma=2.6126336514161914, kernel=linear, total=
3.6s
[CV] C=761.4316758498783, gamma=2.6126336514161914, kernel=linear ....
[CV]  C=761.4316758498783, gamma=2.6126336514161914, kernel=linear, total=
3.7s
[CV] C=761.4316758498783, gamma=2.6126336514161914, kernel=linear ....
[CV]  C=761.4316758498783, gamma=2.6126336514161914, kernel=linear, total=
3.6s
[CV] C=761.4316758498783, gamma=2.6126336514161914, kernel=linear ....
[CV]  C=761.4316758498783, gamma=2.6126336514161914, kernel=linear, total=
3.6s
[CV] C=97392.81883041795, gamma=0.09265545895311562, kernel=linear ...
[CV]  C=97392.81883041795, gamma=0.09265545895311562, kernel=linear, total=
18.4s
[CV] C=97392.81883041795, gamma=0.09265545895311562, kernel=linear ...
[CV]  C=97392.81883041795, gamma=0.09265545895311562, kernel=linear, total=
17.8s
[CV] C=97392.81883041795, gamma=0.09265545895311562, kernel=linear ...
[CV]  C=97392.81883041795, gamma=0.09265545895311562, kernel=linear, total=
29.9s
[CV] C=97392.81883041795, gamma=0.09265545895311562, kernel=linear ...
[CV]  C=97392.81883041795, gamma=0.09265545895311562, kernel=linear, total=
18.9s
[CV] C=97392.81883041795, gamma=0.09265545895311562, kernel=linear ...
[CV]  C=97392.81883041795, gamma=0.09265545895311562, kernel=linear, total=
15.5s
[CV] C=2423.0759984939164, gamma=3.248614270240346, kernel=linear ....
[CV]  C=2423.0759984939164, gamma=3.248614270240346, kernel=linear, total=
4.0s
[CV] C=2423.0759984939164, gamma=3.248614270240346, kernel=linear ....
[CV]  C=2423.0759984939164, gamma=3.248614270240346, kernel=linear, total=
4.1s
[CV] C=2423.0759984939164, gamma=3.248614270240346, kernel=linear ....
[CV]  C=2423.0759984939164, gamma=3.248614270240346, kernel=linear, total=
3.9s
[CV] C=2423.0759984939164, gamma=3.248614270240346, kernel=linear ....
[CV]  C=2423.0759984939164, gamma=3.248614270240346, kernel=linear, total=
```

```
4.1s
[CV] C=2423.0759984939164, gamma=3.248614270240346, kernel=linear ....
[CV]  C=2423.0759984939164, gamma=3.248614270240346, kernel=linear, total=
3.8s
[CV] C=717.3632997255095, gamma=0.3165604432088257, kernel=linear ....
[CV]  C=717.3632997255095, gamma=0.3165604432088257, kernel=linear, total=
3.7s
[CV] C=717.3632997255095, gamma=0.3165604432088257, kernel=linear ....
[CV]  C=717.3632997255095, gamma=0.3165604432088257, kernel=linear, total=
3.6s
[CV] C=717.3632997255095, gamma=0.3165604432088257, kernel=linear ....
[CV]  C=717.3632997255095, gamma=0.3165604432088257, kernel=linear, total=
3.6s
[CV] C=717.3632997255095, gamma=0.3165604432088257, kernel=linear ....
[CV]  C=717.3632997255095, gamma=0.3165604432088257, kernel=linear, total=
3.6s
[CV] C=717.3632997255095, gamma=0.3165604432088257, kernel=linear ....
[CV]  C=717.3632997255095, gamma=0.3165604432088257, kernel=linear, total=
3.6s
[CV] C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf .......
[CV]  C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf, total=   6.6
s
[CV] C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf .......
[CV]  C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf, total=   6.6
s
[CV] C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf .......
[CV]  C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf, total=   6.6
s
[CV] C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf .......
[CV]  C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf, total=   6.6
s
[CV] C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf .......
[CV]  C=4446.667521184072, gamma=3.3597284456608496, kernel=rbf, total=   6.5
s
[CV] C=2963.564121207815, gamma=0.15189814782062885, kernel=linear ...
[CV]  C=2963.564121207815, gamma=0.15189814782062885, kernel=linear, total=
3.9s
[CV] C=2963.564121207815, gamma=0.15189814782062885, kernel=linear ...
[CV]  C=2963.564121207815, gamma=0.15189814782062885, kernel=linear, total=
4.2s
[CV] C=2963.564121207815, gamma=0.15189814782062885, kernel=linear ...
[CV]  C=2963.564121207815, gamma=0.15189814782062885, kernel=linear, total=
4.2s
[CV] C=2963.564121207815, gamma=0.15189814782062885, kernel=linear ...
[CV]  C=2963.564121207815, gamma=0.15189814782062885, kernel=linear, total=
4.0s
[CV] C=2963.564121207815, gamma=0.15189814782062885, kernel=linear ...
[CV]  C=2963.564121207815, gamma=0.15189814782062885, kernel=linear, total=
3.9s
[CV] C=91.64267381686706, gamma=0.01575994483585621, kernel=linear ...
[CV]  C=91.64267381686706, gamma=0.01575994483585621, kernel=linear, total=
3.4s
[CV] C=91.64267381686706, gamma=0.01575994483585621, kernel=linear ...
[CV]  C=91.64267381686706, gamma=0.01575994483585621, kernel=linear, total=
3.5s
[CV] C=91.64267381686706, gamma=0.01575994483585621, kernel=linear ...
[CV]  C=91.64267381686706, gamma=0.01575994483585621, kernel=linear, total=
```

```
3.5s
[CV] C=91.64267381686706, gamma=0.01575994483585621, kernel=linear ...
[CV]  C=91.64267381686706, gamma=0.01575994483585621, kernel=linear, total=
3.6s
[CV] C=91.64267381686706, gamma=0.01575994483585621, kernel=linear ...
[CV]  C=91.64267381686706, gamma=0.01575994483585621, kernel=linear, total=
3.4s
[CV] C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf .....
[CV]  C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf, total=
5.9s
[CV] C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf .....
[CV]  C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf, total=
6.0s
[CV] C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf .....
[CV]  C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf, total=
6.0s
[CV] C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf .....
[CV]  C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf, total=
6.0s
[CV] C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf .....
[CV]  C=24547.601975705915, gamma=0.22153944050588595, kernel=rbf, total=
5.9s
[CV] C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf ......
[CV]  C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf, total=   5.
9s
[CV] C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf ......
[CV]  C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf, total=   5.
9s
[CV] C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf ......
[CV]  C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf, total=   5.
9s
[CV] C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf ......
[CV]  C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf, total=   5.
8s
[CV] C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf ......
[CV]  C=22.76927941060928, gamma=0.22169760231351215, kernel=rbf, total=   5.
8s
[CV] C=16483.850529752886, gamma=1.4752145260435134, kernel=linear ...
[CV]  C=16483.850529752886, gamma=1.4752145260435134, kernel=linear, total=
5.8s
[CV] C=16483.850529752886, gamma=1.4752145260435134, kernel=linear ...
[CV]  C=16483.850529752886, gamma=1.4752145260435134, kernel=linear, total=
6.2s
[CV] C=16483.850529752886, gamma=1.4752145260435134, kernel=linear ...
[CV]  C=16483.850529752886, gamma=1.4752145260435134, kernel=linear, total=
6.3s
[CV] C=16483.850529752886, gamma=1.4752145260435134, kernel=linear ...
[CV]  C=16483.850529752886, gamma=1.4752145260435134, kernel=linear, total=
6.4s
[CV] C=16483.850529752886, gamma=1.4752145260435134, kernel=linear ...
[CV]  C=16483.850529752886, gamma=1.4752145260435134, kernel=linear, total=
5.5s
[CV] C=101445.66881340064, gamma=1.052904084582266, kernel=rbf .......
[CV]  C=101445.66881340064, gamma=1.052904084582266, kernel=rbf, total=  29.2
s
[CV] C=101445.66881340064, gamma=1.052904084582266, kernel=rbf .......
[CV]  C=101445.66881340064, gamma=1.052904084582266, kernel=rbf, total=  27.8
```

```
s
[CV] C=101445.66881340064, gamma=1.052904084582266, kernel=rbf .......
[CV]  C=101445.66881340064, gamma=1.052904084582266, kernel=rbf, total=  36.3
s
[CV] C=101445.66881340064, gamma=1.052904084582266, kernel=rbf .......
[CV]  C=101445.66881340064, gamma=1.052904084582266, kernel=rbf, total=  37.9
s
[CV] C=101445.66881340064, gamma=1.052904084582266, kernel=rbf .......
[CV]  C=101445.66881340064, gamma=1.052904084582266, kernel=rbf, total=  31.2
s
[CV] C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf .......
[CV]  C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf, total=  12.1
s
[CV] C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf .......
[CV]  C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf, total=  12.2
s
[CV] C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf .......
[CV]  C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf, total=  12.0
s
[CV] C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf .......
[CV]  C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf, total=  13.6
s
[CV] C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf .......
[CV]  C=56681.80859029545, gamma=0.9763011917123741, kernel=rbf, total=  12.9
s
[CV] C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf .......
[CV]  C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf, total=   5.7
s
[CV] C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf .......
[CV]  C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf, total=   5.7
s
[CV] C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf .......
[CV]  C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf, total=   5.7
s
[CV] C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf .......
[CV]  C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf, total=   5.7
s
[CV] C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf .......
[CV]  C=48.15822390928914, gamma=0.4633351167983427, kernel=rbf, total=   5.7
s
[CV] C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf .......
[CV]  C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf, total=   5.7
s
[CV] C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf .......
[CV]  C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf, total=   5.7
s
[CV] C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf .......
[CV]  C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf, total=   5.6
s
[CV] C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf .......
[CV]  C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf, total=   5.6
s
[CV] C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf .......
[CV]  C=399.7268155705774, gamma=1.3078757839577408, kernel=rbf, total=   5.6
s
[CV] C=251.14073886281363, gamma=0.8238105204914145, kernel=linear ...
[CV]  C=251.14073886281363, gamma=0.8238105204914145, kernel=linear, total=
```

```
3.5s
[CV] C=251.14073886281363, gamma=0.8238105204914145, kernel=linear ...
[CV]  C=251.14073886281363, gamma=0.8238105204914145, kernel=linear, total=
3.6s
[CV] C=251.14073886281363, gamma=0.8238105204914145, kernel=linear ...
[CV]  C=251.14073886281363, gamma=0.8238105204914145, kernel=linear, total=
3.5s
[CV] C=251.14073886281363, gamma=0.8238105204914145, kernel=linear ...
[CV]  C=251.14073886281363, gamma=0.8238105204914145, kernel=linear, total=
3.5s
[CV] C=251.14073886281363, gamma=0.8238105204914145, kernel=linear ...
[CV]  C=251.14073886281363, gamma=0.8238105204914145, kernel=linear, total=
3.5s
[CV] C=60.17373642891687, gamma=1.2491263443165994, kernel=linear ....
[CV]  C=60.17373642891687, gamma=1.2491263443165994, kernel=linear, total=
3.5s
[CV] C=60.17373642891687, gamma=1.2491263443165994, kernel=linear ....
[CV]  C=60.17373642891687, gamma=1.2491263443165994, kernel=linear, total=
3.5s
[CV] C=60.17373642891687, gamma=1.2491263443165994, kernel=linear ....
[CV]  C=60.17373642891687, gamma=1.2491263443165994, kernel=linear, total=
3.5s
[CV] C=60.17373642891687, gamma=1.2491263443165994, kernel=linear ....
[CV]  C=60.17373642891687, gamma=1.2491263443165994, kernel=linear, total=
3.5s
[CV] C=60.17373642891687, gamma=1.2491263443165994, kernel=linear ....
[CV]  C=60.17373642891687, gamma=1.2491263443165994, kernel=linear, total=
3.4s
[CV] C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf ......
[CV]  C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf, total=   5.
8s
[CV] C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf ......
[CV]  C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf, total=   5.
8s
[CV] C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf ......
[CV]  C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf, total=   5.
8s
[CV] C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf ......
[CV]  C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf, total=   5.
8s
[CV] C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf ......
[CV]  C=15415.161544891856, gamma=0.2691677514619319, kernel=rbf, total=   5.
8s
[CV] C=1888.9148509967113, gamma=0.739678838777267, kernel=linear ....
[CV]  C=1888.9148509967113, gamma=0.739678838777267, kernel=linear, total=
3.8s
[CV] C=1888.9148509967113, gamma=0.739678838777267, kernel=linear ....
[CV]  C=1888.9148509967113, gamma=0.739678838777267, kernel=linear, total=
3.9s
[CV] C=1888.9148509967113, gamma=0.739678838777267, kernel=linear ....
[CV]  C=1888.9148509967113, gamma=0.739678838777267, kernel=linear, total=
3.8s
[CV] C=1888.9148509967113, gamma=0.739678838777267, kernel=linear ....
[CV]  C=1888.9148509967113, gamma=0.739678838777267, kernel=linear, total=
3.8s
[CV] C=1888.9148509967113, gamma=0.739678838777267, kernel=linear ....
[CV]  C=1888.9148509967113, gamma=0.739678838777267, kernel=linear, total=
```

```
3.7s
[CV] C=55.53838911232773, gamma=0.578634378499143, kernel=linear .....
[CV]  C=55.53838911232773, gamma=0.578634378499143, kernel=linear, total=
3.5s
[CV] C=55.53838911232773, gamma=0.578634378499143, kernel=linear .....
[CV]  C=55.53838911232773, gamma=0.578634378499143, kernel=linear, total=
3.5s
[CV] C=55.53838911232773, gamma=0.578634378499143, kernel=linear .....
[CV]  C=55.53838911232773, gamma=0.578634378499143, kernel=linear, total=
3.5s
[CV] C=55.53838911232773, gamma=0.578634378499143, kernel=linear .....
[CV]  C=55.53838911232773, gamma=0.578634378499143, kernel=linear, total=
3.5s
[CV] C=55.53838911232773, gamma=0.578634378499143, kernel=linear .....
[CV]  C=55.53838911232773, gamma=0.578634378499143, kernel=linear, total=
3.5s
[CV] C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf ......
[CV]  C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf, total=   5.
6s
[CV] C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf ......
[CV]  C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf, total=   5.
6s
[CV] C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf ......
[CV]  C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf, total=   5.
6s
[CV] C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf ......
[CV]  C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf, total=   5.
7s
[CV] C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf ......
[CV]  C=26.714480823948186, gamma=1.0117295509275495, kernel=rbf, total=   5.
6s
[CV] C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear ...
[CV]  C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear, total=
4.3s
[CV] C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear ...
[CV]  C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear, total=
4.1s
[CV] C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear ...
[CV]  C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear, total=
4.2s
[CV] C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear ...
[CV]  C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear, total=
4.1s
[CV] C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear ...
[CV]  C=3582.0552780489566, gamma=1.1891370222133257, kernel=linear, total=
4.0s
[CV] C=198.7004781812736, gamma=0.5282819748826726, kernel=linear ....
[CV]  C=198.7004781812736, gamma=0.5282819748826726, kernel=linear, total=
3.5s
[CV] C=198.7004781812736, gamma=0.5282819748826726, kernel=linear ....
[CV]  C=198.7004781812736, gamma=0.5282819748826726, kernel=linear, total=
3.5s
[CV] C=198.7004781812736, gamma=0.5282819748826726, kernel=linear ....
[CV]  C=198.7004781812736, gamma=0.5282819748826726, kernel=linear, total=
3.6s
[CV] C=198.7004781812736, gamma=0.5282819748826726, kernel=linear ....
[CV]  C=198.7004781812736, gamma=0.5282819748826726, kernel=linear, total=
```

```
3.5s
[CV] C=198.7004781812736, gamma=0.5282819748826726, kernel=linear ....
[CV]  C=198.7004781812736, gamma=0.5282819748826726, kernel=linear, total=
3.5s
[CV] C=129.8000604143307, gamma=2.8621383676481322, kernel=linear ....
[CV]  C=129.8000604143307, gamma=2.8621383676481322, kernel=linear, total=
3.6s
[CV] C=129.8000604143307, gamma=2.8621383676481322, kernel=linear ....
[CV]  C=129.8000604143307, gamma=2.8621383676481322, kernel=linear, total=
3.5s
[CV] C=129.8000604143307, gamma=2.8621383676481322, kernel=linear ....
[CV]  C=129.8000604143307, gamma=2.8621383676481322, kernel=linear, total=
3.6s
[CV] C=129.8000604143307, gamma=2.8621383676481322, kernel=linear ....
[CV]  C=129.8000604143307, gamma=2.8621383676481322, kernel=linear, total=
3.5s
[CV] C=129.8000604143307, gamma=2.8621383676481322, kernel=linear ....
[CV]  C=129.8000604143307, gamma=2.8621383676481322, kernel=linear, total=
3.5s
[CV] C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf ......
[CV]  C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf, total=   5.
7s
[CV] C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf ......
[CV]  C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf, total=   5.
7s
[CV] C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf ......
[CV]  C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf, total=   5.
7s
[CV] C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf ......
[CV]  C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf, total=   5.
7s
[CV] C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf ......
[CV]  C=288.4269299593897, gamma=0.17580835850006285, kernel=rbf, total=   5.
6s
[CV] C=6287.039489427172, gamma=0.3504567255332862, kernel=linear ....
[CV]  C=6287.039489427172, gamma=0.3504567255332862, kernel=linear, total=
4.5s
[CV] C=6287.039489427172, gamma=0.3504567255332862, kernel=linear ....
[CV]  C=6287.039489427172, gamma=0.3504567255332862, kernel=linear, total=
4.5s
[CV] C=6287.039489427172, gamma=0.3504567255332862, kernel=linear ....
[CV]  C=6287.039489427172, gamma=0.3504567255332862, kernel=linear, total=
4.7s
[CV] C=6287.039489427172, gamma=0.3504567255332862, kernel=linear ....
[CV]  C=6287.039489427172, gamma=0.3504567255332862, kernel=linear, total=
4.6s
[CV] C=6287.039489427172, gamma=0.3504567255332862, kernel=linear ....
[CV]  C=6287.039489427172, gamma=0.3504567255332862, kernel=linear, total=
4.4s
[CV] C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf .......
[CV]  C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf, total=  21.8
s
[CV] C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf .......
[CV]  C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf, total=  25.0
s
[CV] C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf .......
[CV]  C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf, total=  23.5
```

```
s
[CV] C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf .......
[CV]  C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf, total=  24.3
s
[CV] C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf .......
[CV]  C=61217.04421344494, gamma=1.6279689407405564, kernel=rbf, total=  22.4
s
[CV] C=926.9787684096649, gamma=2.147979593060577, kernel=rbf ........
[CV]  C=926.9787684096649, gamma=2.147979593060577, kernel=rbf, total=   5.9s
[CV] C=926.9787684096649, gamma=2.147979593060577, kernel=rbf ........
[CV]  C=926.9787684096649, gamma=2.147979593060577, kernel=rbf, total=   5.9s
[CV] C=926.9787684096649, gamma=2.147979593060577, kernel=rbf ........
[CV]  C=926.9787684096649, gamma=2.147979593060577, kernel=rbf, total=   5.8s
[CV] C=926.9787684096649, gamma=2.147979593060577, kernel=rbf ........
[CV]  C=926.9787684096649, gamma=2.147979593060577, kernel=rbf, total=   5.9s
[CV] C=926.9787684096649, gamma=2.147979593060577, kernel=rbf ........
[CV]  C=926.9787684096649, gamma=2.147979593060577, kernel=rbf, total=   5.9s
[CV] C=33946.157064934, gamma=2.2642426492862313, kernel=linear ......
[CV]  C=33946.157064934, gamma=2.2642426492862313, kernel=linear, total=   9.
2s
[CV] C=33946.157064934, gamma=2.2642426492862313, kernel=linear ......
[CV]  C=33946.157064934, gamma=2.2642426492862313, kernel=linear, total=   9.
0s
[CV] C=33946.157064934, gamma=2.2642426492862313, kernel=linear ......
[CV]  C=33946.157064934, gamma=2.2642426492862313, kernel=linear, total=   8.
3s
[CV] C=33946.157064934, gamma=2.2642426492862313, kernel=linear ......
[CV]  C=33946.157064934, gamma=2.2642426492862313, kernel=linear, total=   9.
4s
[CV] C=33946.157064934, gamma=2.2642426492862313, kernel=linear ......
[CV]  C=33946.157064934, gamma=2.2642426492862313, kernel=linear, total=   8.
5s
[CV] C=84789.82947739525, gamma=0.3176359085304841, kernel=linear ....
[CV]  C=84789.82947739525, gamma=0.3176359085304841, kernel=linear, total=  2
4.1s
[CV] C=84789.82947739525, gamma=0.3176359085304841, kernel=linear ....
[CV]  C=84789.82947739525, gamma=0.3176359085304841, kernel=linear, total=  1
6.8s
[CV] C=84789.82947739525, gamma=0.3176359085304841, kernel=linear ....
[CV]  C=84789.82947739525, gamma=0.3176359085304841, kernel=linear, total=  2
6.2s
[CV] C=84789.82947739525, gamma=0.3176359085304841, kernel=linear ....
[CV]  C=84789.82947739525, gamma=0.3176359085304841, kernel=linear, total=  1
8.8s
[CV] C=84789.82947739525, gamma=0.3176359085304841, kernel=linear ....
[CV]  C=84789.82947739525, gamma=0.3176359085304841, kernel=linear, total=  1
4.3s

[Parallel(n_jobs=1)]: Done 250 out of 250 | elapsed: 37.1min finished
```

Out[71]:
```
RandomizedSearchCV(cv=5, estimator=SVR(), n_iter=50,
                   param_distributions={'C': <scipy.stats._distn_infrastructu
re.rv_frozen object at 0x000001B1001B40D0>,
                                        'gamma': <scipy.stats._distn_infrastr
ucture.rv_frozen object at 0x000001B111B38DC0>,
                                        'kernel': ['linear', 'rbf']},
                   random_state=42, scoring='neg_mean_squared_error',
                   verbose=2)
```

In [73]:
```
print(random_search.best_params_)
print(random_search.best_estimator_)
```

```
{'C': 157055.10989448498, 'gamma': 0.26497040005002437, 'kernel': 'rbf'}
SVR(C=157055.10989448498, gamma=0.26497040005002437)
```

In [ ]:
```
# This time the search found a good set of hyperparameters for the RBF kernel.
# Randomized search tends to find better hyperparameters than grid search in t
he same amount of time.
```

In [74]:
```
# Random Search Evaluation Scores:
negative_mse = random_search.best_score_
rmse = np.sqrt(-negative_mse)
print(rmse)
```

```
54767.960710084146
```

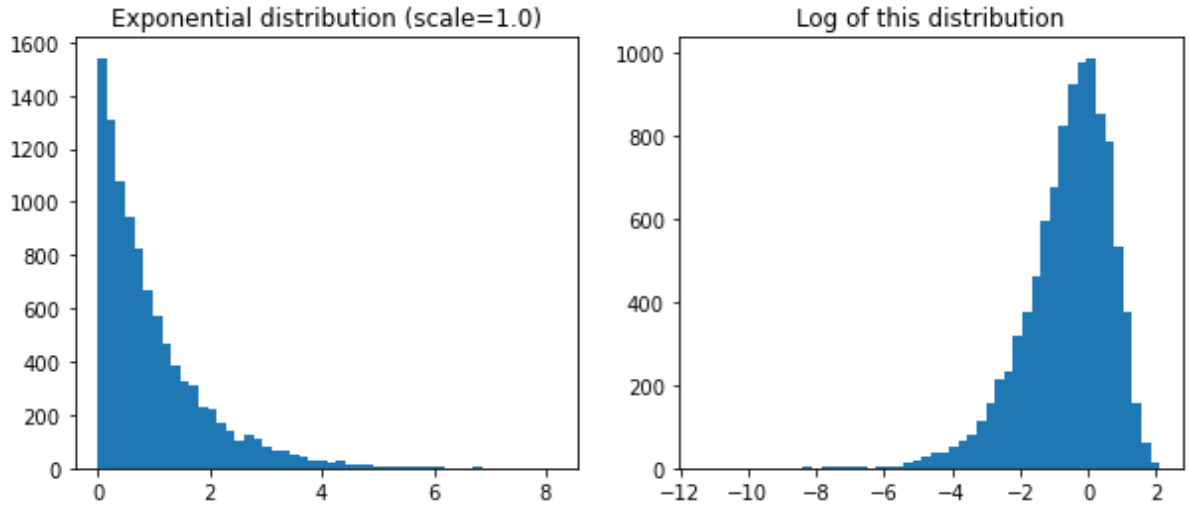
In [ ]:
```
# This is getting a lot closer to the performance of the random tree regresso
r, but it's still not as good.
```

In [75]:
```
# Random Search Evaluation Cross Validation Scores:
cvres = random_search.cv_results_
for mean_score, params in zip(cvres["mean_test_score"], cvres["params"]):
    print(np.sqrt(-mean_score), params)
```

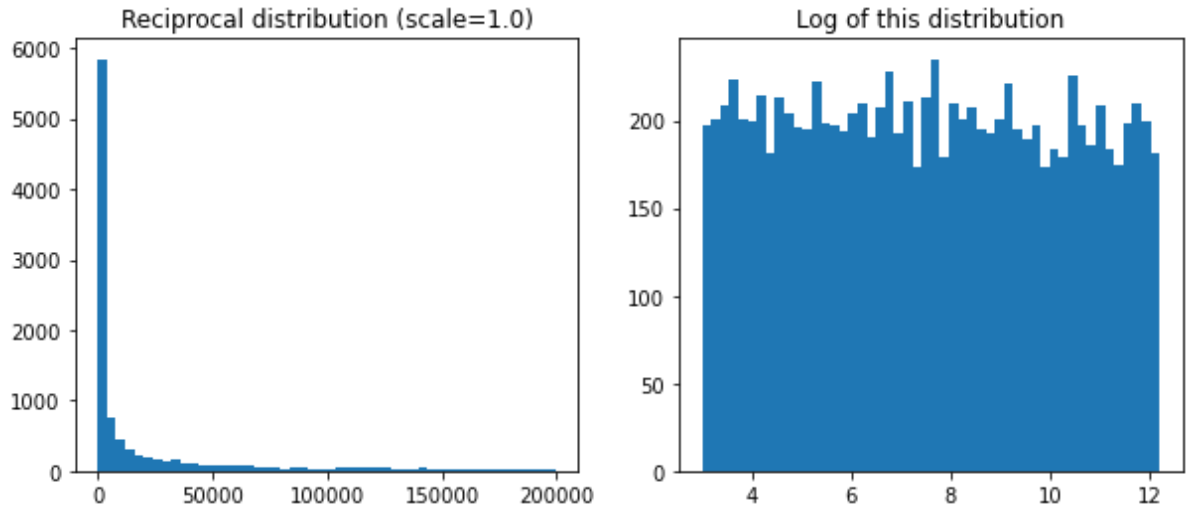
```
70487.7237017999 {'C': 629.782329591372, 'gamma': 3.010121430917521, 'kerne
l': 'linear'}
65152.0053324654 {'C': 26290.206464300216, 'gamma': 0.9084469696321253, 'kern
el': 'rbf'}
100293.25860091094 {'C': 84.14107900575871, 'gamma': 0.059838768608680676, 'k
ernel': 'rbf'}
70574.10617249804 {'C': 432.37884813148855, 'gamma': 0.15416196746656105, 'ke
rnel': 'linear'}
118838.9539854597 {'C': 24.17508294611391, 'gamma': 3.503557475158312, 'kerne
l': 'rbf'}
70029.31697470722 {'C': 113564.03940586245, 'gamma': 0.0007790692366582295,
'kernel': 'rbf'}
107662.7096409836 {'C': 108.30488238805073, 'gamma': 0.3627537294604771, 'ker
nel': 'rbf'}
77563.09297583395 {'C': 21.344953672647435, 'gamma': 0.023332523598323388, 'k
ernel': 'linear'}
62578.81054092114 {'C': 5603.270317432516, 'gamma': 0.15023452872733867, 'ker
nel': 'rbf'}
54767.960710084146 {'C': 157055.10989448498, 'gamma': 0.26497040005002437, 'k
ernel': 'rbf'}
70364.56459025714 {'C': 27652.464358739708, 'gamma': 0.2227358621286903, 'ker
nel': 'linear'}
70356.32992013592 {'C': 171377.39570378003, 'gamma': 0.628789100540856, 'kern
el': 'linear'}
70374.9596621843 {'C': 5385.293820172355, 'gamma': 0.18696125197741642, 'kern
el': 'linear'}
118802.64114676064 {'C': 22.59903216621323, 'gamma': 2.850796878935603, 'kern
el': 'rbf'}
70362.58602083729 {'C': 34246.75194632794, 'gamma': 0.3632878599687583, 'kern
el': 'linear'}
100450.29228566251 {'C': 167.7278956080511, 'gamma': 0.2757870542258224, 'ker
nel': 'rbf'}
72624.98934432027 {'C': 61.54360542501371, 'gamma': 0.6835472281341501, 'kern
el': 'linear'}
111159.67724229675 {'C': 98.73897389920914, 'gamma': 0.4960365360493639, 'ker
nel': 'rbf'}
63169.93892419065 {'C': 8935.505635947808, 'gamma': 0.37354658165762367, 'ker
nel': 'rbf'}
71258.47476709314 {'C': 135.76775824842434, 'gamma': 0.838636245624803, 'kern
el': 'linear'}
62838.44584088588 {'C': 151136.20282548846, 'gamma': 1.4922453771381408, 'ker
nel': 'rbf'}
70478.72525452363 {'C': 761.4316758498783, 'gamma': 2.6126336514161914, 'kern
el': 'linear'}
70352.5125626773 {'C': 97392.81883041795, 'gamma': 0.09265545895311562, 'kern
el': 'linear'}
70403.13959481634 {'C': 2423.0759984939164, 'gamma': 3.248614270240346, 'kern
el': 'linear'}
70476.30827614202 {'C': 717.3632997255095, 'gamma': 0.3165604432088257, 'kern
el': 'linear'}
106956.84165347065 {'C': 4446.667521184072, 'gamma': 3.3597284456608496, 'ker
nel': 'rbf'}
70395.28759239399 {'C': 2963.564121207815, 'gamma': 0.15189814782062885, 'ker
nel': 'linear'}
71759.26815837105 {'C': 91.64267381686706, 'gamma': 0.01575994483585621, 'ker
nel': 'linear'}
57812.89042268699 {'C': 24547.601975705915, 'gamma': 0.22153944050588595, 'ke
```

```
rnel': 'rbf'}
114677.53379803285 {'C': 22.76927941060928, 'gamma': 0.22169760231351215, 'ke
rnel': 'rbf'}
70370.76656627818 {'C': 16483.850529752886, 'gamma': 1.4752145260435134, 'ker
nel': 'linear'}
60052.28891896938 {'C': 101445.66881340064, 'gamma': 1.052904084582266, 'kern
el': 'rbf'}
61481.27261343283 {'C': 56681.80859029545, 'gamma': 0.9763011917123741, 'kern
el': 'rbf'}
114571.2386400452 {'C': 48.15822390928914, 'gamma': 0.4633351167983427, 'kern
el': 'rbf'}
111660.72456364948 {'C': 399.7268155705774, 'gamma': 1.3078757839577408, 'ker
nel': 'rbf'}
70769.46638036826 {'C': 251.14073886281363, 'gamma': 0.8238105204914145, 'ker
nel': 'linear'}
72677.26222787892 {'C': 60.17373642891687, 'gamma': 1.2491263443165994, 'kern
el': 'linear'}
59278.486452716556 {'C': 15415.161544891856, 'gamma': 0.2691677514619319, 'ke
rnel': 'rbf'}
70408.73062596374 {'C': 1888.9148509967113, 'gamma': 0.739678838777267, 'kern
el': 'linear'}
72902.98919438479 {'C': 55.53838911232773, 'gamma': 0.578634378499143, 'kerne
l': 'linear'}
118062.4067893129 {'C': 26.714480823948186, 'gamma': 1.0117295509275495, 'ker
nel': 'rbf'}
70386.9630554019 {'C': 3582.0552780489566, 'gamma': 1.1891370222133257, 'kern
el': 'linear'}
70919.0290566038 {'C': 198.7004781812736, 'gamma': 0.5282819748826726, 'kerne
l': 'linear'}
71312.85405560398 {'C': 129.8000604143307, 'gamma': 2.8621383676481322, 'kern
el': 'linear'}
89483.75607147012 {'C': 288.4269299593897, 'gamma': 0.17580835850006285, 'ker
nel': 'rbf'}
70378.47205891725 {'C': 6287.039489427172, 'gamma': 0.3504567255332862, 'kern
el': 'linear'}
68027.93134796123 {'C': 61217.04421344494, 'gamma': 1.6279689407405564, 'kern
el': 'rbf'}
111760.98691537287 {'C': 926.9787684096649, 'gamma': 2.147979593060577, 'kern
el': 'rbf'}
70362.66297948634 {'C': 33946.157064934, 'gamma': 2.2642426492862313, 'kerne
l': 'linear'}
70352.83535012191 {'C': 84789.82947739525, 'gamma': 0.3176359085304841, 'kern
el': 'linear'}
```

In [76]:
```
# Let's look at the exponential distribution we used, with scale=1.0. Note tha
t some samples are much larger or
# smaller than 1.0, but when you look at the log of the distribution, you can
 see that most values are actually
# concentrated roughly in the range of exp(-2) to exp(+2), which is about 0.1
 to 7.4.

expon_distrib = expon(scale=1.)
samples = expon_distrib.rvs(10000, random_state=42)
plt.figure(figsize=(10, 4))
plt.subplot(121)
plt.title("Exponential distribution (scale=1.0)")
plt.hist(samples, bins=50)
plt.subplot(122)
plt.title("Log of this distribution")
plt.hist(np.log(samples), bins=50)
plt.show()
```



In [77]:
```
# The distribution we used for C looks quite different: the scale of the sampl
es is picked from a uniform distribution
# within a given range, which is why the right graph, which represents the log
of the samples, looks roughly constant.
# This distribution is useful when you don't have a clue of what the target sc
ale is:

reciprocal_distrib = reciprocal(20, 200000)
samples = reciprocal_distrib.rvs(10000, random_state=42)
plt.figure(figsize=(10, 4))
plt.subplot(121)
plt.title("Reciprocal distribution (scale=1.0)")
plt.hist(samples, bins=50)
plt.subplot(122)
plt.title("Log of this distribution")
plt.hist(np.log(samples), bins=50)
plt.show()
```



In [78]:
```
# The reciprocal distribution is useful when you have no idea what the scale o
f the hyperparameter should be
# (indeed, as you can see on the figure on the right, all scales are equally l
ikely, within the given range),
# whereas the exponential distribution is best when you know (more or less) wh
at the scale of the hyperparameter should be.
```

3. Try adding a transformer in the preparation pipeline to select only the most import attributes

In [91]:
```
def indices_of_top_k(array, k):
    return np.sort(np.argpartition(np.array(array), -k)[-k:])

class TopFeatureSelector(BaseEstimator, TransformerMixin):
    def __init__(self, feature_importances, k):
        self.feature_importances = feature_importances
        self.k = k
    def fit(self, X, y=None):
        self.feature_indices_ = indices_of_top_k(self.feature_importances, sel
f.k)
        return self
    def transform(self, X):
        return X[:, self.feature_indices_]

# Note that this feature selector assumes you have already computed the featur
e importances
```

In [92]:
```
k = 5
top_k_feature_indices = indices_of_top_k(random_tree_reg_feature_importances,
k)
top_k_feature_indices
```

Out[92]:
```
array([ 0,  7,  9, 10, 12], dtype=int64)
```

In [93]:
```
np.array(attributes)[top_k_feature_indices]
```

Out[93]:
```
array(['longitude', 'median_income', 'pop_per_hhold', 'bedrooms_per_room',
       'INLAND'], dtype='<U19')
```

In [94]:
```
# Let's double check that these are indeed the top k features:
sorted(zip(random_tree_reg_feature_importances, attributes), reverse=True)[:k]
```

Out[94]:
```
[(0.32820383909931383, 'median_income'),
 (0.143077997543386, 'INLAND'),
 (0.10804589057862227, 'pop_per_hhold'),
 (0.08119879731256048, 'longitude'),
 (0.080087908597283, 'bedrooms_per_room')]
```

In [95]:
```
# Now let's create a new pipeline that runs the previously defined preparation
pipeline, and adds top k feature selection:
preparation_and_feature_selection_pipeline = Pipeline([
    ('preparation', full_pipeline),
    ('feature_selection', TopFeatureSelector(random_tree_reg_feature_importanc
es, k))
])
```

In [96]:
```
housing_prepared_top_k_features = preparation_and_feature_selection_pipeline.f
it_transform(housing)
```

In [97]:
```
# To check, let's look at the features of the first 3 instances and compare th
em to the top k features
housing_prepared_top_k_features[0:3]
```

Out[97]:
```
array([[-1.15604281, -0.61493744, -0.08649871,  0.15531753,  0.        ],
       [-1.17602483,  1.33645936, -0.03353391, -0.83628902,  0.        ],
       [ 1.18684903, -0.5320456 , -0.09240499,  0.4222004 ,  0.        ]])
```

In [98]:
```
housing_prepared[0:3, top_k_feature_indices]
```

Out[98]:
```
array([[-1.15604281, -0.61493744, -0.08649871,  0.15531753,  0.        ],
       [-1.17602483,  1.33645936, -0.03353391, -0.83628902,  0.        ],
       [ 1.18684903, -0.5320456 , -0.09240499,  0.4222004 ,  0.        ]])
```

In [99]:
```
# They match, looks good!
```

4. Try creating a single pipeline that does the full data preparation plus the final prediction

In [117]:
```
prepare_and_select_and_predict_pipeline = Pipeline([
    ('preparation', full_pipeline),
    ('feature_selection', TopFeatureSelector(feature_importances, k)),
    ('random_forest', RandomForestRegressor(n_estimators = 100)) # These param
s are from above
])
```

In [118]:
```
prepare_and_select_and_predict_pipeline.fit(housing, housing_labels)
```

Out[118]:
```
Pipeline(steps=[('preparation',
                 ColumnTransformer(transformers=[('num',
                                                  Pipeline(steps=[('imputer',
                                                                   SimpleImpu
ter(strategy='median')),
                                                                  ('attribs_a
dder',
                                                                   CombinedAt
tributesAdder()),
                                                                  ('std_scale
r',
                                                                   StandardSc
aler())]),
                                                  ['longitude', 'latitude',
                                                   'housing_median_age',
                                                   'total_rooms',
                                                   'total_bedrooms',
                                                   'population', 'household
s',
                                                   'median_income']),
                                                 ('cat', OneHotEncoder(...
                 ('feature_selection',
                  TopFeatureSelector(feature_importances=array([8.11987973e-0
2, 7.49994545e-02, 4.26313932e-02, 1.68069388e-02,
       1.56017401e-02, 1.65650352e-02, 1.56470978e-02, 3.28203839e-01,
       5.46998513e-02, 1.08045891e-01, 8.00879086e-02, 1.31052196e-02,
       1.43077998e-01, 7.35183616e-05, 4.23486740e-03, 5.02045066e-03]),
                                     k=5)),
                 ('random_forest', RandomForestRegressor())])
```

In [119]:
```
# Let's try the full pipeline on a few instances:

some_data = housing.iloc[:4]
some_labels = housing_labels.iloc[:4]

print("Predictions:\t", prepare_and_select_and_predict_pipeline.predict(some_d
ata))
print("Labels:\t\t", list(some_labels))
```

```
Predictions:     [264037.   342019.01 209587.    52427.  ]
Labels:          [286600.0, 340600.0, 196900.0, 46300.0]
```

In [120]:
```
# We can see that the full pipeline is working
```

5. Automatically explore some preparation options using GridSearchCV

In [121]:
```
param_grid = [{
    'preparation__num__imputer__strategy': ['mean', 'median', 'most_frequent'
],
    'feature_selection__k': list(range(1, len(feature_importances) + 1))
}]

grid_search_prep = GridSearchCV(prepare_and_select_and_predict_pipeline, param
_grid, cv=5,
                                scoring='neg_mean_squared_error', verbose=2)
grid_search_prep.fit(housing, housing_labels)
```

```
Fitting 5 folds for each of 48 candidates, totalling 240 fits
[CV] feature_selection__k=1, preparation__num__imputer__strategy=mean

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worke
rs.

[CV]  feature_selection__k=1, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=mean

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    1.1s remaining:    0.
0s
```

```
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=1, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=1, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=mean, total
```

```
= 1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=mean, total
=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=median, tot
al=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=2, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=2, preparation__num__imputer__strategy=most_freque
nt, total=   1.2s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=mean, total
=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=mean, total
=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=mean, total
=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=mean, total
=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=mean, total
=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=median, tot
al=   1.9s
```

```
[CV] feature_selection__k=3, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=median, tot
al=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=median, tot
al=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=median, tot
al=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=median, tot
al=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=most_freque
nt, total=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=most_freque
nt, total=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=most_freque
nt, total=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=most_freque
nt, total=   1.9s
[CV] feature_selection__k=3, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=3, preparation__num__imputer__strategy=most_freque
nt, total=   1.9s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=mean, total
=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=mean, total
=   2.3s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=mean, total
=   2.3s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=mean, total
=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=mean, total
=   2.3s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=median, tot
al=   2.3s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=median, tot
al=   2.3s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=median, tot
al=   2.3s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=median
```

```
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=median, tot
al=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=median, tot
al=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=most_freque
nt, total=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=most_freque
nt, total=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=most_freque
nt, total=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=most_freque
nt, total=   2.4s
[CV] feature_selection__k=4, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=4, preparation__num__imputer__strategy=most_freque
nt, total=   2.4s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=mean, total
=   3.1s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=mean, total
=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=mean, total
=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=mean, total
=   3.1s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=mean, total
=   3.1s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=median, tot
al=   3.1s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=median, tot
al=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=median, tot
al=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=median, tot
al=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=median, tot
al=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=most_frequen
t
```

```
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=most_freque
nt, total=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=most_freque
nt, total=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=most_freque
nt, total=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=most_freque
nt, total=   3.0s
[CV] feature_selection__k=5, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=5, preparation__num__imputer__strategy=most_freque
nt, total=   3.0s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=mean, total
=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=mean, total
=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=mean, total
=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=mean, total
=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=mean, total
=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=median, tot
al=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=median, tot
al=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=median, tot
al=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=median, tot
al=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=median, tot
al=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=most_freque
nt, total=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=most_freque
nt, total=   3.6s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=most_frequen
```

```
t
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=most_freque
nt, total=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=most_freque
nt, total=   3.5s
[CV] feature_selection__k=6, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=6, preparation__num__imputer__strategy=most_freque
nt, total=   3.5s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=mean, total
=   4.2s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=mean, total
=   4.2s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=mean, total
=   4.2s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=mean, total
=   4.2s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=mean, total
=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=median, tot
al=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=median, tot
al=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=median, tot
al=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=median, tot
al=   4.2s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=median, tot
al=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=most_freque
nt, total=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=most_freque
nt, total=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=most_freque
nt, total=   4.1s
[CV] feature_selection__k=7, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=most_freque
nt, total=   4.1s
```

```
[CV] feature_selection__k=7, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=7, preparation__num__imputer__strategy=most_freque
nt, total=   4.1s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=mean, total
=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=mean, total
=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=mean, total
=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=mean, total
=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=mean, total
=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=median, tot
al=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=median, tot
al=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=median, tot
al=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=median, tot
al=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=median, tot
al=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=most_freque
nt, total=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=most_freque
nt, total=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=most_freque
nt, total=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=most_freque
nt, total=   4.5s
[CV] feature_selection__k=8, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=8, preparation__num__imputer__strategy=most_freque
nt, total=   4.5s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=mean, total
=   5.2s
```

```
[CV] feature_selection__k=9, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=mean, total
=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=mean, total
=   5.2s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=mean, total
=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=mean, total
=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=median, tot
al=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=median, tot
al=   5.2s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=median, tot
al=   5.2s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=median, tot
al=   5.2s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=median, tot
al=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=most_freque
nt, total=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=most_freque
nt, total=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=most_freque
nt, total=   5.1s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=most_freque
nt, total=   5.2s
[CV] feature_selection__k=9, preparation__num__imputer__strategy=most_frequen
t
[CV]  feature_selection__k=9, preparation__num__imputer__strategy=most_freque
nt, total=   5.2s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=mean, tota
l=   5.8s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=mean, tota
l=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=mean, tota
l=   5.8s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=mean
```

```
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=mean, tota
l=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=mean, tota
l=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=median, to
tal=   5.8s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=median, to
tal=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=median, to
tal=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=median, to
tal=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=median, to
tal=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=most_frequ
ent, total=   5.7s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=most_frequ
ent, total=   5.8s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=most_frequ
ent, total=   5.8s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=most_frequ
ent, total=   5.8s
[CV] feature_selection__k=10, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=10, preparation__num__imputer__strategy=most_frequ
ent, total=   5.7s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=mean, tota
l=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=mean, tota
l=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=mean, tota
l=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=mean, tota
l=   6.4s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=mean, tota
l=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=median, to
```

```
tal=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=median, to
tal=   6.4s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=median, to
tal=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=median, to
tal=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=median, to
tal=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=most_frequ
ent, total=   6.4s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=most_frequ
ent, total=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=most_frequ
ent, total=   6.2s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=most_frequ
ent, total=   6.3s
[CV] feature_selection__k=11, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=11, preparation__num__imputer__strategy=most_frequ
ent, total=   6.2s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=mean, tota
l=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=mean, tota
l=   6.9s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=mean, tota
l=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=median, to
tal=   6.9s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=median, to
tal=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=median, to
tal=   6.9s
```

```
[CV] feature_selection__k=12, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=median, to
tal=   6.9s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=median, to
tal=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=most_frequ
ent, total=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=most_frequ
ent, total=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=most_frequ
ent, total=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=most_frequ
ent, total=   6.8s
[CV] feature_selection__k=12, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=12, preparation__num__imputer__strategy=most_frequ
ent, total=   6.8s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=mean, tota
l=   6.9s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=median, to
tal=   6.9s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=median, to
tal=   6.9s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=median, to
tal=   6.9s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=median, to
tal=   7.0s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=most_freque
```

```
nt
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=most_frequ
ent, total=   7.3s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=most_frequ
ent, total=   7.2s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=most_frequ
ent, total=   7.3s
[CV] feature_selection__k=13, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=13, preparation__num__imputer__strategy=most_frequ
ent, total=   7.2s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=mean, tota
l=   7.3s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=median, to
tal=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=median, to
tal=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=median, to
tal=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=median, to
tal=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=most_frequ
ent, total=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=most_frequ
ent, total=   7.0s
```

```
[CV] feature_selection__k=14, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=most_frequ
ent, total=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=most_frequ
ent, total=   7.0s
[CV] feature_selection__k=14, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=14, preparation__num__imputer__strategy=most_frequ
ent, total=   7.0s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=mean, tota
l=   7.0s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=median, to
tal=   7.0s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=median, to
tal=   7.2s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=most_frequ
ent, total=   7.0s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=most_frequ
```

```
ent, total=   7.0s
[CV] feature_selection__k=15, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=15, preparation__num__imputer__strategy=most_frequ
ent, total=   7.0s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=mean
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=mean, tota
l=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=median
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=median, to
tal=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s
[CV] feature_selection__k=16, preparation__num__imputer__strategy=most_freque
nt
[CV]  feature_selection__k=16, preparation__num__imputer__strategy=most_frequ
ent, total=   7.1s

[Parallel(n_jobs=1)]: Done 240 out of 240 | elapsed: 18.6min finished
```

Out[121]:
```
GridSearchCV(cv=5,
             estimator=Pipeline(steps=[('preparation',
                                        ColumnTransformer(transformers=[('nu
m',
                                                                         Pipe
line(steps=[('imputer',

SimpleImputer(strategy='median')),

('attribs_adder',

CombinedAttributesAdder()),

('std_scaler',

StandardScaler())]),
                                                                         ['lo
ngitude',
                                                                          'la
titude',
                                                                          'ho
using_median_age',
                                                                          'to
tal_rooms',
                                                                          'to
tal_bedrooms',
                                                                          'po
pulation',
                                                                          'ho
useholds',
                                                                          'me
dian_inc...
       5.46998513e-02, 1.08045891e-01, 8.00879086e-02, 1.31052196e-02,
       1.43077998e-01, 7.35183616e-05, 4.23486740e-03, 5.02045066e-03]),
                                                    k=5)),
                                       ('random_forest',
                                        RandomForestRegressor())]),
             param_grid=[{'feature_selection__k': [1, 2, 3, 4, 5, 6, 7, 8, 9,
                                                  10, 11, 12, 13, 14, 15, 1
6],
                          'preparation__num__imputer__strategy': ['mean',
                                                                  'median',
                                                                  'most_frequ
ent']}],
             scoring='neg_mean_squared_error', verbose=2)
```

In [122]:
```
grid_search_prep.best_params_
```

Out[122]:
```
{'feature_selection__k': 8, 'preparation__num__imputer__strategy': 'median'}
```

In [124]:
```
# Above it looks like 'median' is the best imputer strategy and 8 features are
useful.


# In the solutions:
# The best imputer strategy is 'most_frequent' and almost all the features are
useful (15 out of 16)
# The last one (ISLAND) just seems to add noise
```