notebook

December 2, 2022

1 Used Car Prices Prediction in Python

1.1 Business Problem

A used car dealer is expanding and has hired a large number of junior salespeople. Although promising, these junior employees have difficulties pricing used cars that arrive at the dealership. Sales have declined 18% in recent months. The sales team have reached out to the data science team to get help with this problem.

1.1.1 Project Tasks

- Predict the price that a used car should be listed at based on features of the car?
- Based on success criteria of "It is known that cars that are more than £1500 from the estimated price will not sell", try to make predictions within this range.

1.1.2 Data Validation

This data set has 6738 rows, 9 columns. I have validated all variables and I have not made any changes after validation. All the columns are just as described in the data dictionary:

- model: character, 18 possible values
- year: numeric, from 1998 to 2020
- price: numeric, from 850 to 59995
- transmission: character, 4 categories
- mileage: numeric, from 2 to 174419
- fuelType: character, 4 categories
- tax: numeric, from 0 to 565
- mpg: numeric, from 2.8 to 235
- engineSize: numeric, 16 possible values

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.style as style
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression # For Multiple-Linear␣
 ↪Regression Model
from sklearn.tree import DecisionTreeRegressor # For Desicion Tree Regression␣
 ↪Model
from sklearn.ensemble import RandomForestRegressor # For Random Forest␣
 ↪Regresion Model
from xgboost import XGBRegressor # For XGBoost Regression Modle
from sklearn.model_selection import GridSearchCV # For hyperparameters tuning
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import PowerTransformer
from sklearn.metrics import r2_score,mean_squared_error
plt.style.use('ggplot')
```

[2]:
```
car = pd.read_csv('car.csv')
car.head()
```

[2]:

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	GT86	2016	16000	Manual	24089	Petrol	265	36.2	2.0
1	GT86	2017	15995	Manual	18615	Petrol	145	36.2	2.0
2	GT86	2015	13998	Manual	27469	Petrol	265	36.2	2.0
3	GT86	2017	18998	Manual	14736	Petrol	150	36.2	2.0
4	GT86	2017	17498	Manual	36284	Petrol	145	36.2	2.0

[3]:
```
#validate any missing value in dataset
car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6738 entries, 0 to 6737
```

```
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   model         6738 non-null   object
 1   year          6738 non-null   int64
 2   price         6738 non-null   int64
 3   transmission  6738 non-null   object
 4   mileage       6738 non-null   int64
 5   fuelType      6738 non-null   object
 6   tax           6738 non-null   int64
 7   mpg           6738 non-null   float64
 8   engineSize    6738 non-null   float64
dtypes: float64(2), int64(4), object(3)
memory usage: 473.9+ KB
```

[4]:
```
#validate any negative values in numeric variables
car.describe()
```

[4]:
```
              year         price        mileage          tax          mpg  \
count  6738.000000   6738.000000    6738.000000  6738.000000  6738.000000
mean   2016.748145  12522.391066   22857.413921    94.697240    63.042223
std       2.204062   6345.017587   19125.464147    73.880776    15.836710
min    1998.000000    850.000000       2.000000     0.000000     2.800000
25%    2016.000000   8290.000000    9446.000000     0.000000    55.400000
50%    2017.000000  10795.000000   18513.000000   135.000000    62.800000
75%    2018.000000  14995.000000   31063.750000   145.000000    69.000000
max    2020.000000  59995.000000  174419.000000   565.000000   235.000000

        engineSize
count  6738.000000
mean      1.471297
std       0.436159
min       0.000000
25%       1.000000
50%       1.500000
75%       1.800000
max       4.500000
```

[5]:
```
#validate possible 18 values
car.model.unique()
car.model.nunique()
```

[5]: 18

[6]:
```
#validate year of manufacture from 1998 to 2020
car.year.unique()
```

[6]: array([2016, 2017, 2015, 2020, 2013, 2019, 2018, 2014, 2012, 2005, 2003,
        2004, 2001, 2008, 2007, 2010, 2011, 2006, 2009, 2002, 1999, 2000,
        1998])

[7]:
```
#validate 4 types of transmission
car.transmission.unique()
```

[7]: array(['Manual', 'Automatic', 'Semi-Auto', 'Other'], dtype=object)

[8]:
```
#validate 4 fuel Types
car.fuelType.unique()
```

[8]: array(['Petrol', 'Other', 'Hybrid', 'Diesel'], dtype=object)

[9]:
```
#validate 16 possible values in engineSize
car.engineSize.unique()
car.engineSize.nunique()
```

[9]: 16

1.2 Exploratory Analysis

I have investigated the target variable and features of the car, and the relationship between target variable and features. After the analysis, I decided to apply the following changes to enable modeling:

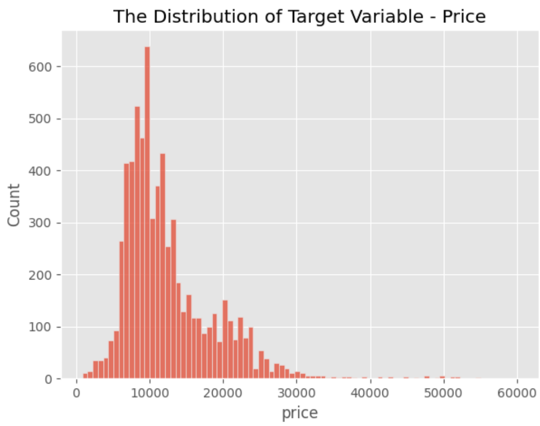
- Price: use log transformation
- Create a new ordinal variable from tax variable

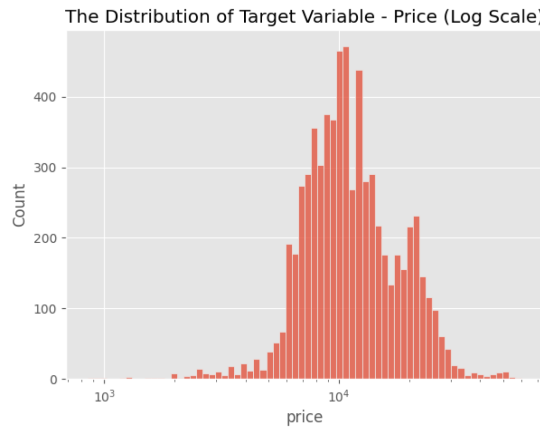
1.2.1 Target Variable - Price

Since we need to predict the price, the price variable would be our target variable. From the histogram on the left below, we can see there is a longer right tail. Therefore, we apply log transformation of the price variable, the distribution on the right below is close to normal distribution.

[10]:
```
### Target Variable - Price
fig, axes = plt.subplots(1,2,figsize=(15,5))
sns.histplot(car.price,ax=axes[0]).set(title='The Distribution of Target␣
↪Variable - Price')
sns.histplot(car.price,log_scale=True,ax=axes[1]).set(title='The Distribution␣
↪of Target Variable - Price (Log Scale)')
```

[10]: [Text(0.5, 1.0, 'The Distribution of Target Variable - Price (Log Scale)')]





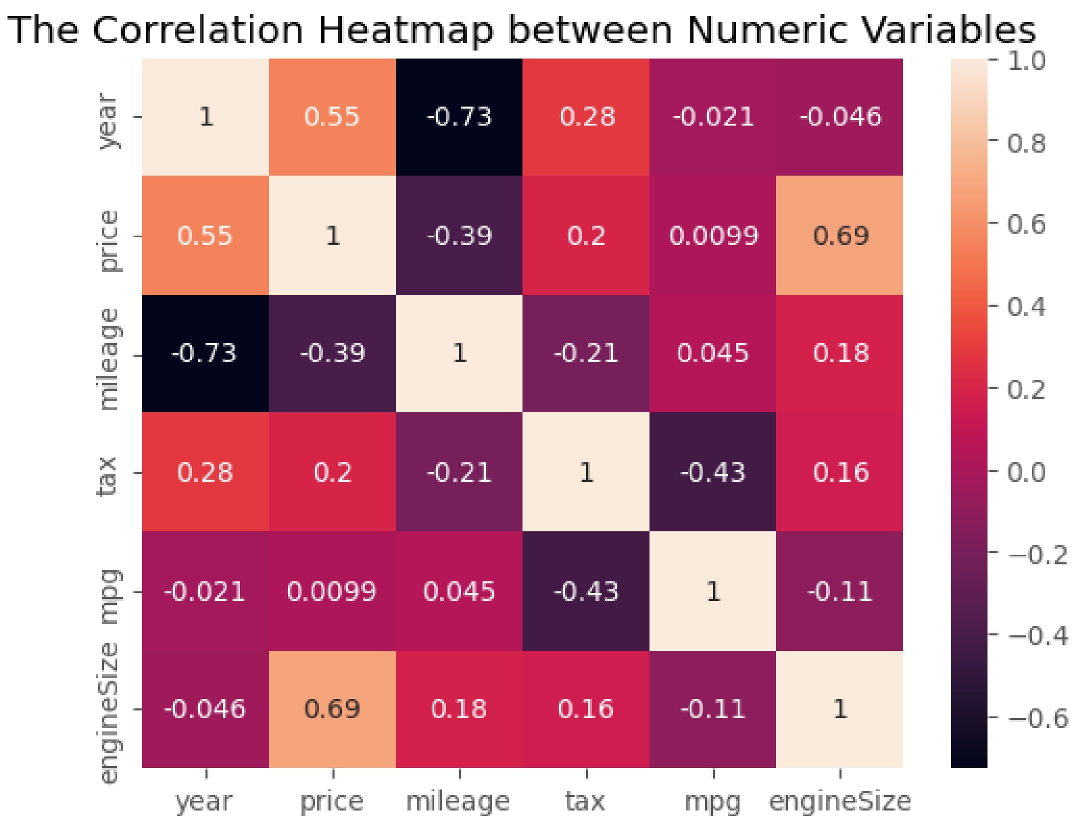
[11]:
```python
# Log transformation for price.
car.price = np.log(car.price)
```

1.2.2 Numeric Variables - Year, Price, Mileage, Tax, mpg, Engine Size

From the heatmap below, we can conclude that there is a moderate linear positive relationship between price in log transformation and engine size and year. Also, there is a moderate to strong linear negative linear negative relationship between year and mileage.

[12]:
```python
sns.heatmap(car.corr(),annot=True).set(title='The Correlation Heatmap between␣
↪Numeric Variables')
```

[12]:
```
[Text(0.5, 1.0, 'The Correlation Heatmap between Numeric Variables')]
```

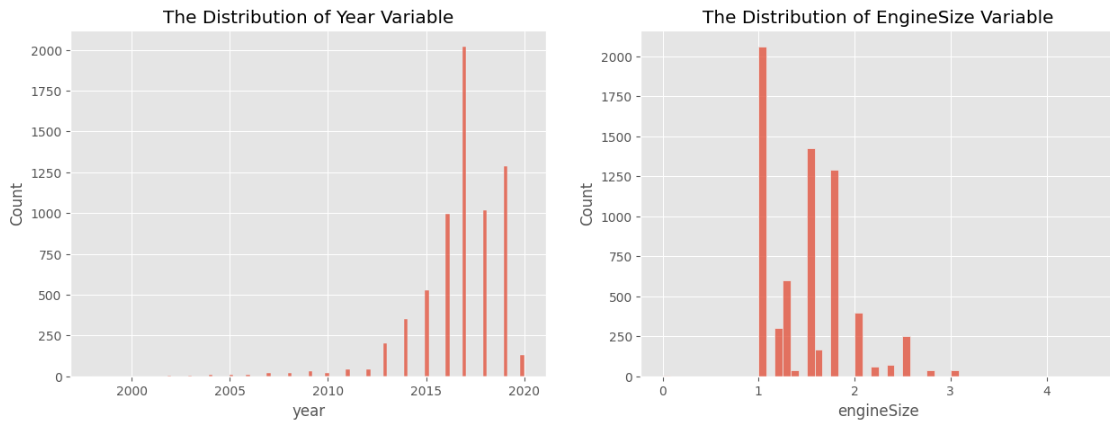


Distribution about year and engine size

Since year and engine size is most related to price, I checked their distribution. From the histogram below, their distribution is skewed.

[13]:
```
fig, axes = plt.subplots(1,2,figsize=(15,5))
sns.histplot(car.year,ax=axes[0]).set(title='The Distribution of Year Variable')
sns.histplot(car.engineSize,ax=axes[1]).set(title='The Distribution of␣
↪EngineSize Variable')
```

[13]:
```
[Text(0.5, 1.0, 'The Distribution of EngineSize Variable')]
```



Relationship between mpg, tax, mileage and price

Since I cannot intepret the linear relationship between mpg, tax and mileage and price variable from the heatmap above, I decided to make scatterplot to further investigate their non-linear relationship. From the scatterplots below, there is linear relationship between mileage and price. Hard to see tell the relationship between price and mpg. There are clusters in the scatterplot between price and tax, so I decided to create a new ordinal variable from the tax variable.

[14]:
```
fig, axes = plt.subplots(1,3,figsize=(15,5))
sns.scatterplot(x = 'mpg', y = 'price', data = car,ax=axes[0]).set(title='Price vs mpg')
sns.scatterplot(x = 'tax', y = 'price', data = car, ax=axes[1]).set(title='Price vs tax')
sns.scatterplot(x = 'mileage',y = 'price', data = car, ax=axes[2]).set(title='Price vs Mileage');
```

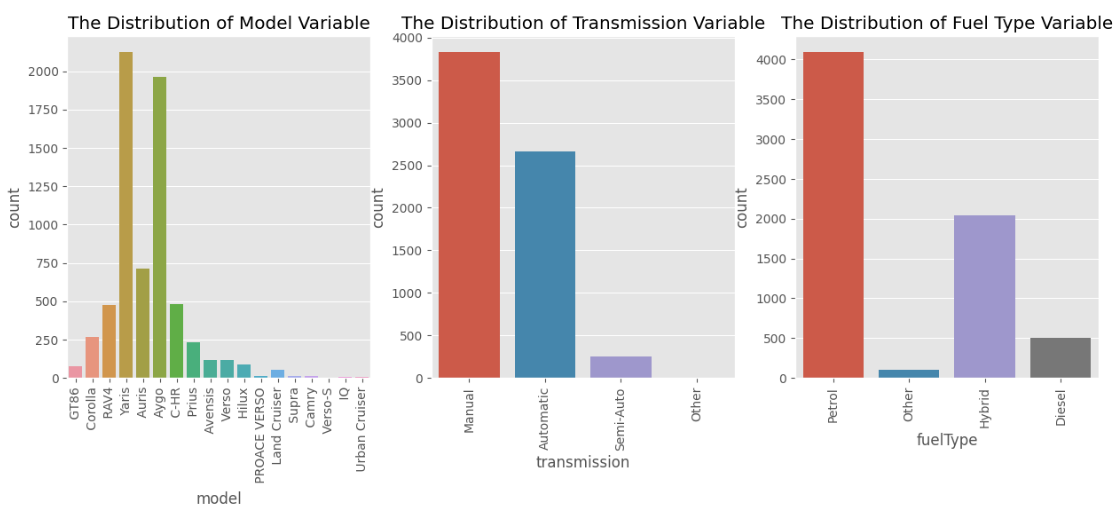
[15]:
```
# Convert tax variable into an ordinal variable
car.loc[(car['tax'] <= 100,'tax')] = 1
```

```
car.loc[((car['tax'] <= 200) & (car['tax'] > 100) ,'tax')] = 2
car.loc[((car['tax'] <= 300) & (car['tax'] > 200) ,'tax')] = 3
car.loc[(car['tax'] > 300 ,'tax')] = 4
```

1.2.3 Categorical Variables - model, transmission, fuelType

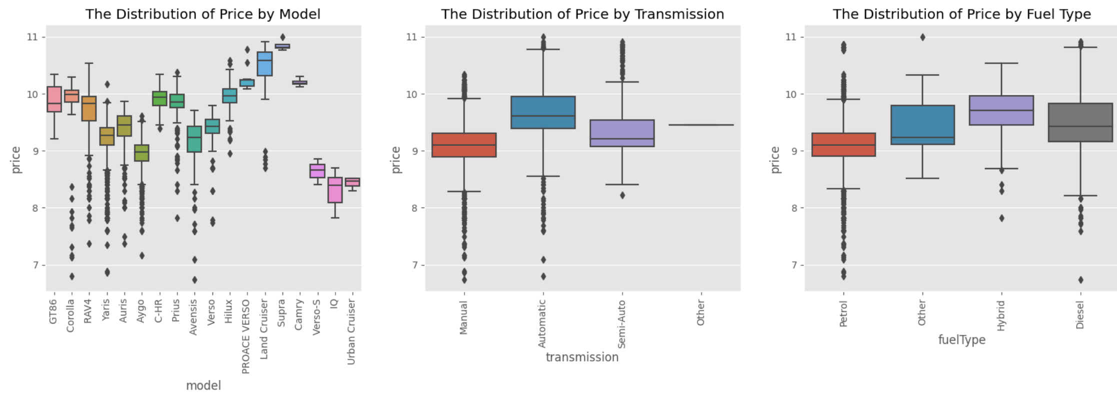
From the bar charts below, we can see the most frequent categories in model, transmission and fuelType variables - Yaris, Manual, Petrol in the dataset. I also investigated their relationship between price. From the boxplots below, we can see there is a difference in distribution of prices among each categories in each variable.

[16]:
```
fig, axes = plt.subplots(1,3,figsize=(15,5))
sns.countplot(x= 'model',data =car,ax=axes[0]).set(title='The Distribution of␣
↪Model Variable')
sns.countplot(x = 'transmission', data = car,ax=axes[1]).set(title='The␣
↪Distribution of Transmission Variable')
sns.countplot(x = 'fuelType', data =car,ax=axes[2]).set(title='The Distribution␣
↪of Fuel Type Variable')
for ax in fig.axes:
    plt.sca(ax)
    plt.xticks(rotation=90);
```



[17]:
```
fig, axes = plt.subplots(1,3,figsize=(20,5))
sns.boxplot(data=car, x='model',y='price',ax=axes[0]).set(title='The␣
↪Distribution of Price by Model')
sns.boxplot(data=car, x='transmission',y='price',ax=axes[1]).set(title='The␣
↪Distribution of Price by Transmission')
sns.boxplot(data=car, x='fuelType',y='price',ax=axes[2]).set(title='The␣
↪Distribution of Price by Fuel Type')
for ax in fig.axes:
```

```
    plt.sca(ax)
    plt.xticks(rotation=90);
```



1.3 Model Fitting & Evaluation

Predicting the price is a regression problem in machine learning. I am choosing the Linear Regression model as a base model, because we can see strong to moderate relationship between some features and target variable. Three comparison models I am choosing are - Decision Tree regression model, because it is easy to interpret with independence from outliers. - Randome Forest regression model, because it is a bagging type of ensemble model, it is easy to interpret and generally could improve the model performance - XGboost regression model, because it is an ensemble model conbined of boosting and bagging, it is fast to run and could improve the model performance.

For the evaluation, I am choosing R squared and RMSE (Root Mean Squared Error) to evaluate the model. R squared measures how well the model fits dependent variables (i.e. features). RMSE measures how much your predicted results deviate from the actual number.

I have also done the hyperparameters for the comparison models and plotted feature importances for each models.

1.3.1 Prepare Data for Modelling

To enable modelling, we chose year,model,transmission,mileage,fuelType,tax,engineSize as features, price as target variables. I also have made the following changes:

- Normalize the numeric features
- Convert the categorical variables into numeric features
- Split the data into a training set and a test set

[47]:
```
labelencoder = LabelEncoder()
car['model'] = labelencoder.fit_transform(car['model'])
car['transmission'] = labelencoder.fit_transform(car['transmission'])
car['fuelType'] = labelencoder.fit_transform(car['fuelType'])
```

[19]:
```
feature_cols = ␣
↪['year','transmission','fuelType','engineSize','tax','model','mileage']
X = car[feature_cols] # Features
y = car['price'] # Target variable
```

[20]:
```
# define the scaler
scaler = PowerTransformer()
# fit and transform the train set
X[['year','engineSize','mileage']] = scaler.
↪fit_transform(X[['year','engineSize','mileage']])
```

[21]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=42)
```

Linear Regression Model

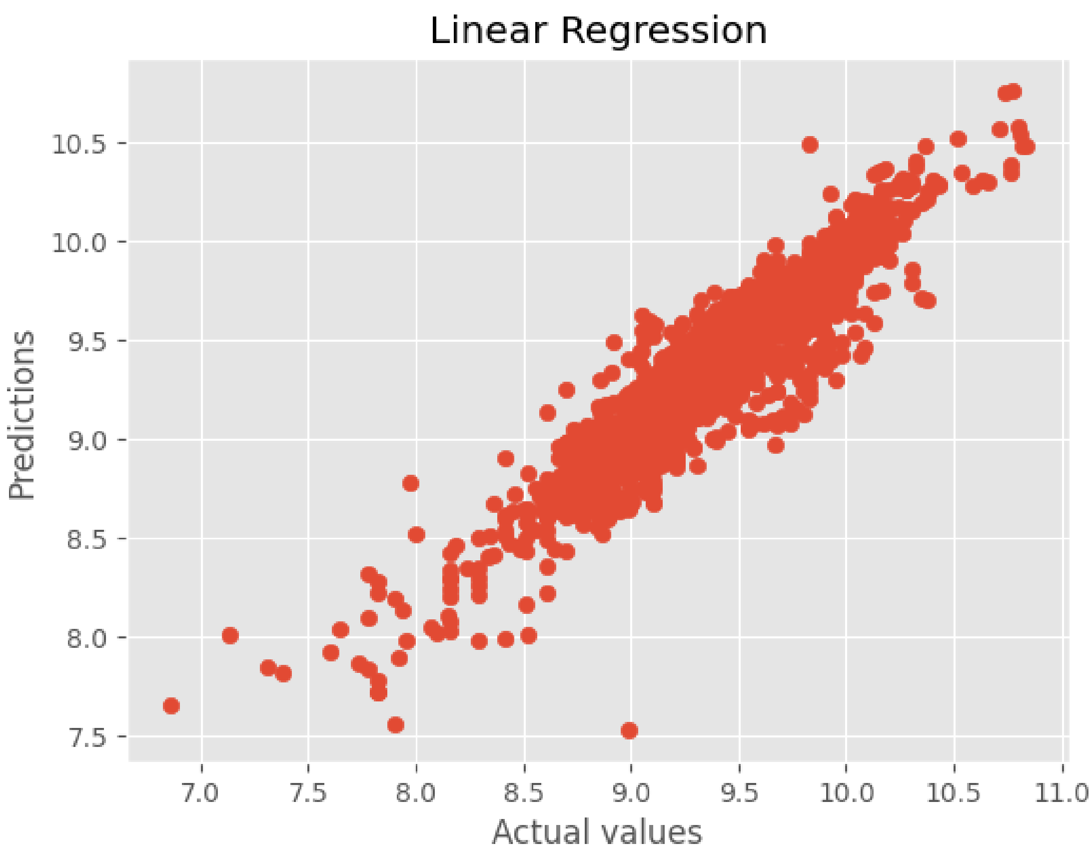
[22]:
```
lr = LinearRegression()
lr.fit(X_train, y_train)
```

[22]:
```
LinearRegression()
```

[23]:
```
y_pred = lr.predict(X_test)
print('Linear Regression r2_score: ',r2_score(y_test,y_pred))
print('Linear Regression Root Mean Squared Error: ',np.
↪sqrt(mean_squared_error(y_test,y_pred)))
```

```
Linear Regression r2_score:  0.8581849472893514
Linear Regression Root Mean Squared Error:  0.17887225612192453
```

[24]:
```
# plot the predicitons and actual values for test data set
plt.scatter(y_test,y_pred)
plt.xlabel("Actual values")
plt.ylabel("Predictions")
plt.title('Linear Regression')
plt.show()
```

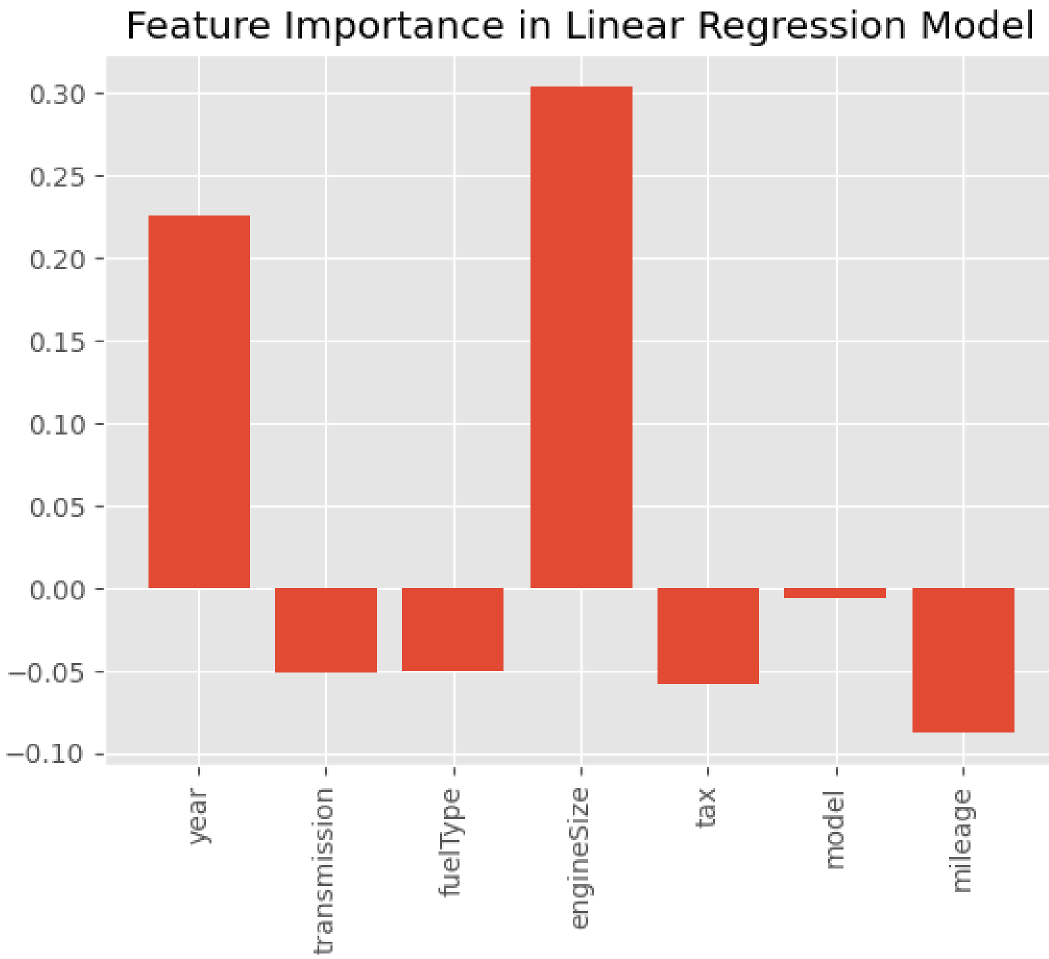


Finding the feature importance

[25]:
```
resultdict = {}
for i in range(len(feature_cols)):
    resultdict[feature_cols[i]] = lr.coef_[i]

plt.bar(resultdict.keys(),resultdict.values())
plt.xticks(rotation='vertical')
plt.title('Feature Importance in Linear Regression Model')
```

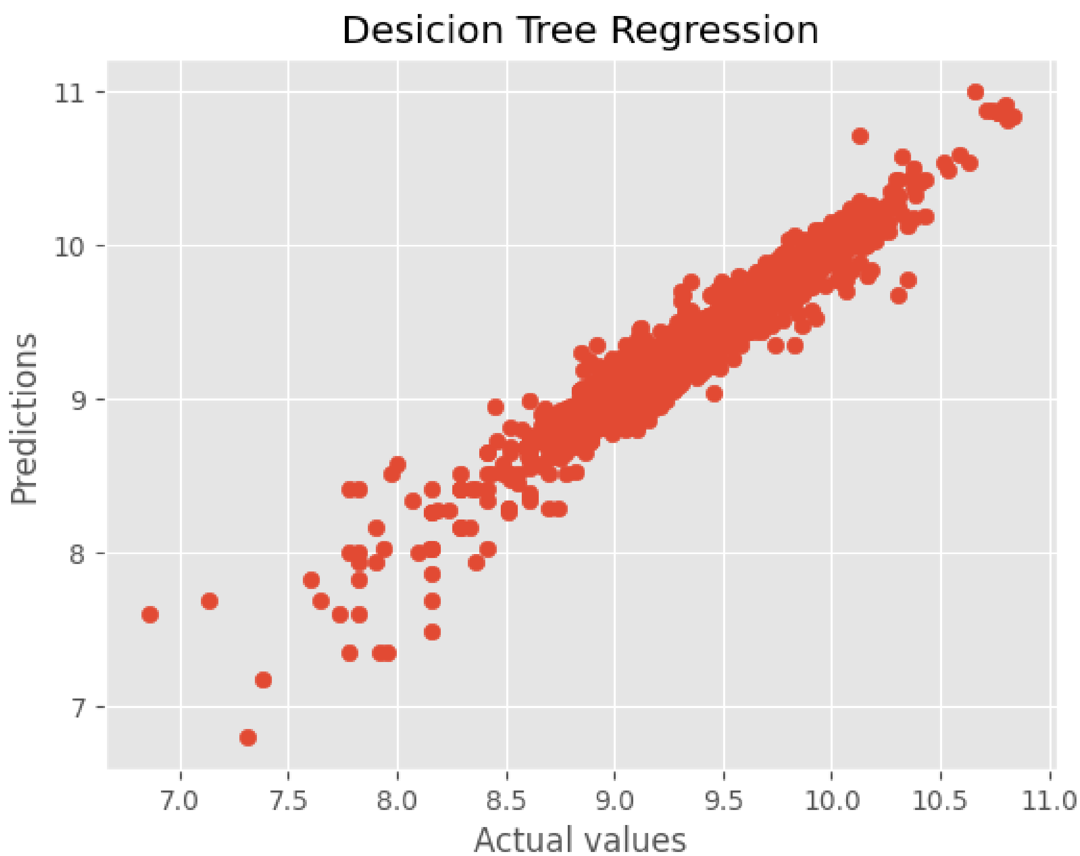
[25]: Text(0.5, 1.0, 'Feature Importance in Linear Regression Model')



Decision Tree Regression Model

[26]:
```
tree = DecisionTreeRegressor(max_depth=12,min_samples_split=2,random_state=42)
tree.fit(X_train,y_train)
y_pred2 = tree.predict(X_test)
```

[27]:
```
# plot the predicitons and actual values for test data set
plt.scatter(y_test,y_pred2)
plt.xlabel("Actual values")
plt.ylabel("Predictions")
plt.title('Desicion Tree Regression')
plt.show()
```




```
[28]: d_r2 = tree.score(X_test, y_test)
print("Decision Tree Regressor R-squared: {}".format(d_r2))

d_mse = mean_squared_error(y_pred2, y_test)
d_rmse = np.sqrt(d_mse)
print("Decision Tree Regressor RMSE: {}".format(d_rmse))
```

```
Decision Tree Regressor R-squared: 0.9378957123254383
Decision Tree Regressor RMSE: 0.11837026281480763
```

Finding the best parameter for Decision Tree Regression Model

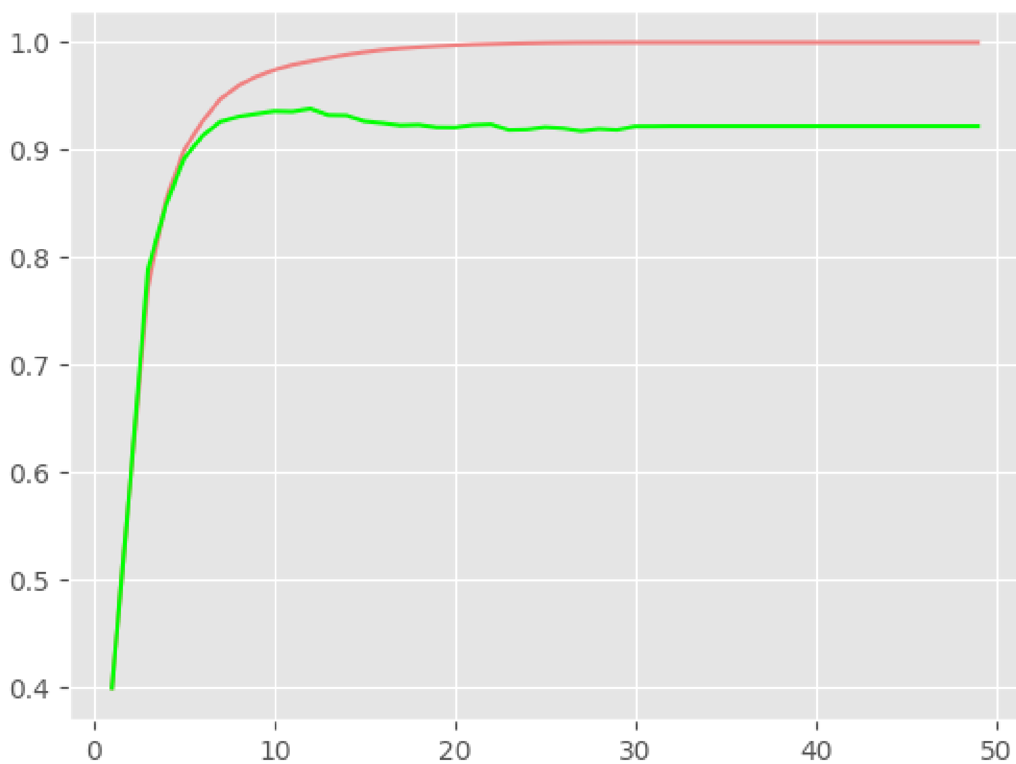
```
[29]: train_score = []
test_score = []
max_score = 0
max_pair = (0,0)

for i in range(1,50):
    tree = DecisionTreeRegressor(max_depth=i,random_state=42)
    tree.fit(X_train,y_train)
```

```
    y_pred = tree.predict(X_test)
    train_score.append(tree.score(X_train,y_train))
    test_score.append(r2_score(y_test,y_pred))
    test_pair = (i,r2_score(y_test,y_pred))
    if test_pair[1] > max_pair[1]:
        max_pair = test_pair

fig, ax = plt.subplots()
ax.plot(np.arange(1,50), train_score, label = "Training R^2",color='lightcoral')
ax.plot(np.arange(1,50), test_score, label = "Testing R^2",color='lime')
print(f'Best max_depth is: {max_pair[0]} \nTesting R^2 is: {max_pair[1]}')
```

```
Best max_depth is: 12
Testing R^2 is: 0.9378957123254383
```



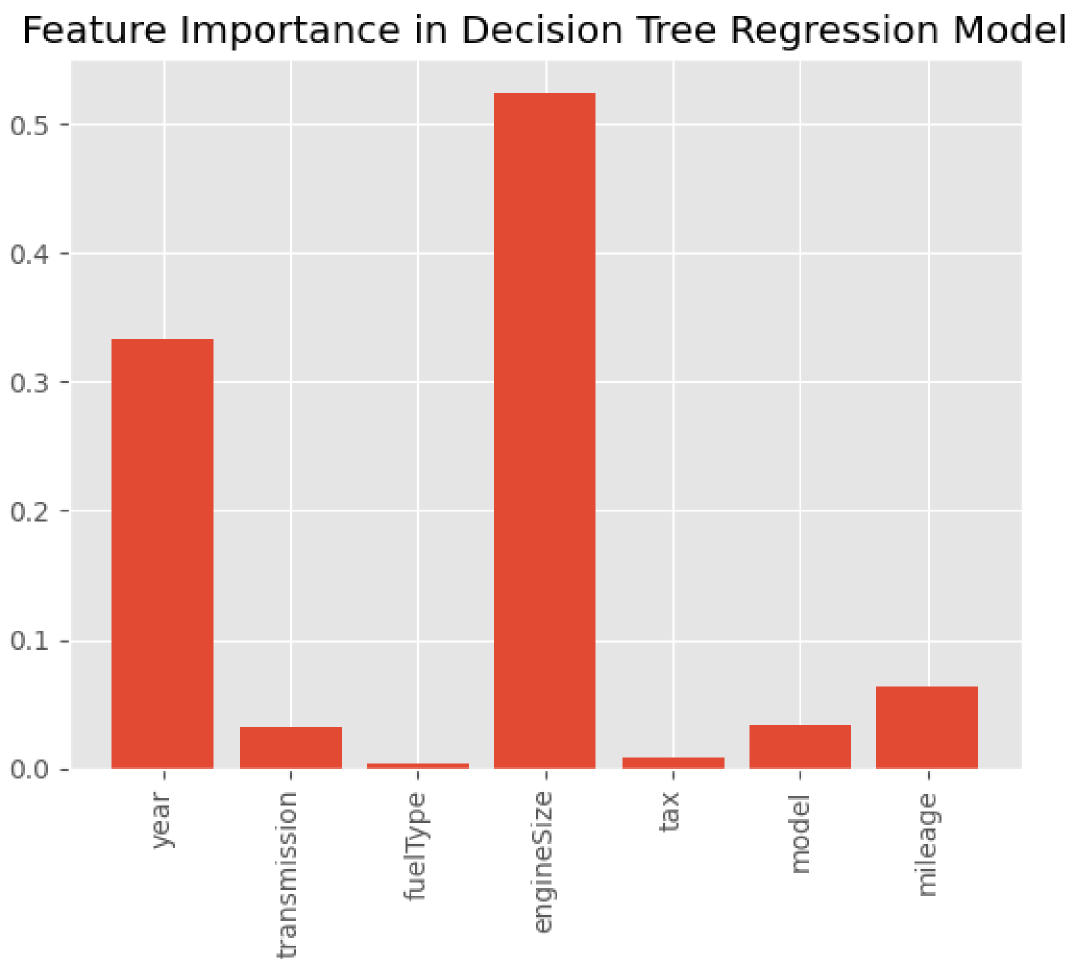
Finding the feature importance

[30]:
```
importance = tree.feature_importances_

f_importance = {}
for i in range(len(feature_cols)):
     f_importance[feature_cols[i]] = importance[i]
```

```
plt.bar(f_importance.keys(),f_importance.values())
plt.xticks(rotation='vertical')
plt.title('Feature Importance in Decision Tree Regression Model')
```

[30]: Text(0.5, 1.0, 'Feature Importance in Decision Tree Regression Model')

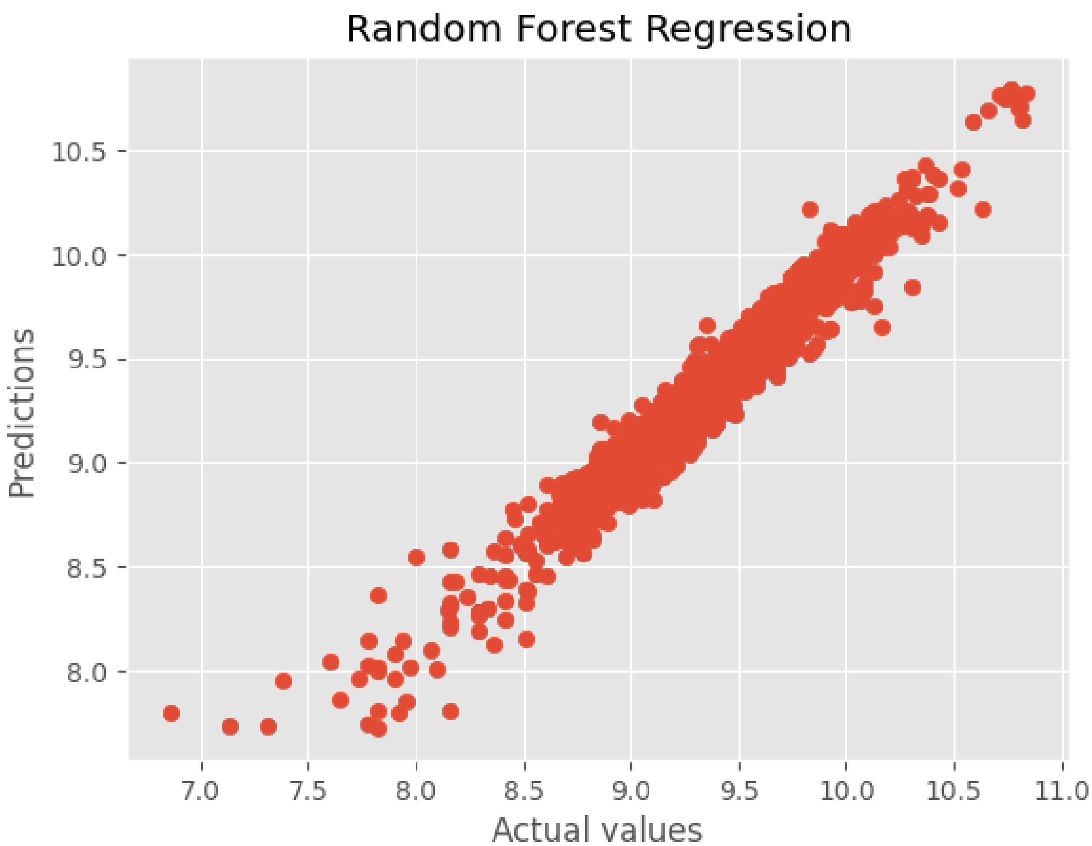


Random Forest Model

[31]:
```
rf = RandomForestRegressor(max_depth = 10, max_features= 3,min_samples_leaf= 3,
 ↪min_samples_split= 8, n_estimators= 1000,random_state=42)
rf.fit(X_train,y_train)
y_pred3 = rf.predict(X_test)
```

[32]:
```
# plot the predicitons and actual values for test data set
plt.scatter(y_test,y_pred3)
plt.xlabel("Actual values")
```

```
plt.ylabel("Predictions")
plt.title('Random Forest Regression')
plt.show()
```



[33]:
```
rf_r2 = rf.score(X_test, y_test)
print("Random Forest Regressor R-squared: {}".format(rf_r2))

rf_mse = mean_squared_error(y_pred3, y_test)
rf_rmse = np.sqrt(rf_mse)
print("Random Forest Regressor RMSE: {}".format(rf_rmse))
```

```
Random Forest Regressor R-squared: 0.9567323628880114
Random Forest Regressor RMSE: 0.09880147057473333
```

Finding the best parameter for Random Forest Regression Model

[34]:
```
# Create the parameter grid
param_grid = {
    'bootstrap': [True],
    'max_depth': [10, 20, 30],
```

```
    'max_features': [2, 3],
    'min_samples_leaf': [3, 4, 5],
    'min_samples_split': [8, 10, 12],
    'n_estimators': [100, 500, 1000]
}
# Create a based model
rf_r = RandomForestRegressor()
# Instantiate the grid search model
grid_search = GridSearchCV(estimator = rf_r, param_grid = param_grid,
                          cv = 3, n_jobs = -1, verbose = 2)
```

[35]:

```
# Fit the grid search to the data(**It could take about 5 mins to run the␣
 ↪gridsearch)
grid_search.fit(X_train, y_train)
grid_search.best_params_
```

```
Fitting 3 folds for each of 162 candidates, totalling 486 fits
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.1s
```

```
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.0s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.0s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
```

```
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.0s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.0s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
```

```
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.1s
```

```
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=10, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.1s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.5s
```

```
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.5s
```

```
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.2s
```

```
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
```

```
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=10, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.4s
```

```
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.2s
```

```
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.9s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.7s
```

```
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.8s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.8s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.6s
```

```
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.5s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.2s
```

```
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.8s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.8s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.3s
```

```
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=20, max_features=3, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
```

```
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.4s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.1s
```

```
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.2s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=500; total time=   1.1s
[CV] END bootstrap=True, max_depth=30, max_features=2, min_samples_leaf=5,
min_samples_split=12, n_estimators=1000; total time=   2.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=8, n_estimators=1000; total time=   2.8s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=500; total time=   1.5s
```

```
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=10, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=3,
min_samples_split=12, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=8, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=500; total time=   1.4s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=10, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=4,
min_samples_split=12, n_estimators=1000; total time=   2.7s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=8, n_estimators=1000; total time=   2.6s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
```

```
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=100; total time=   0.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=500; total time=   1.3s
[CV] END bootstrap=True, max_depth=30, max_features=3, min_samples_leaf=5,
min_samples_split=10, n_estimators=1000; total time=   2.6s
```

[35]:
```
{'bootstrap': True,
 'max_depth': 10,
 'max_features': 3,
 'min_samples_leaf': 3,
 'min_samples_split': 8,
 'n_estimators': 1000}
```

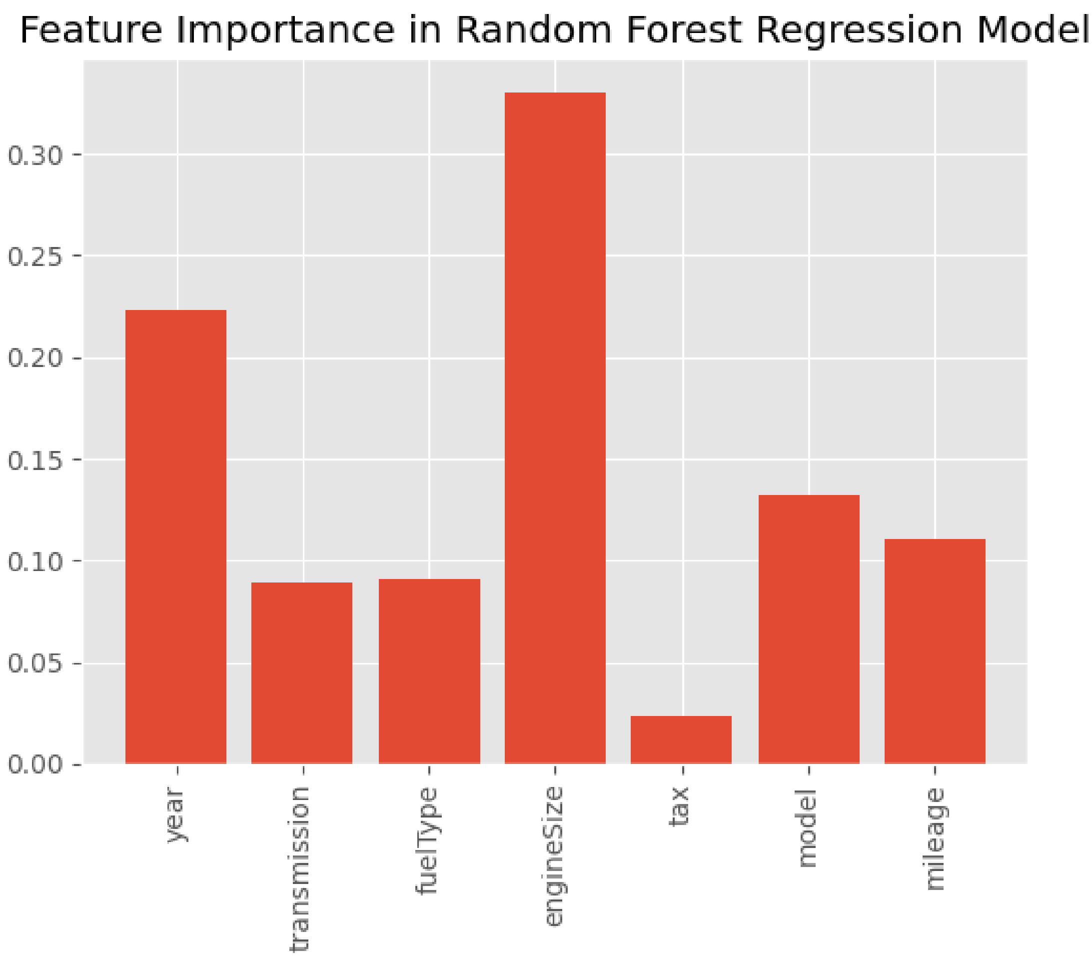
Finding the feature importance

[36]:
```python
rf_importance = rf.feature_importances_

rf_f_importance = {}
for i in range(len(feature_cols)):
     rf_f_importance[feature_cols[i]] = rf_importance[i]

plt.bar(rf_f_importance.keys(),rf_f_importance.values())
plt.xticks(rotation='vertical')
plt.title('Feature Importance in Random Forest Regression Model')
```

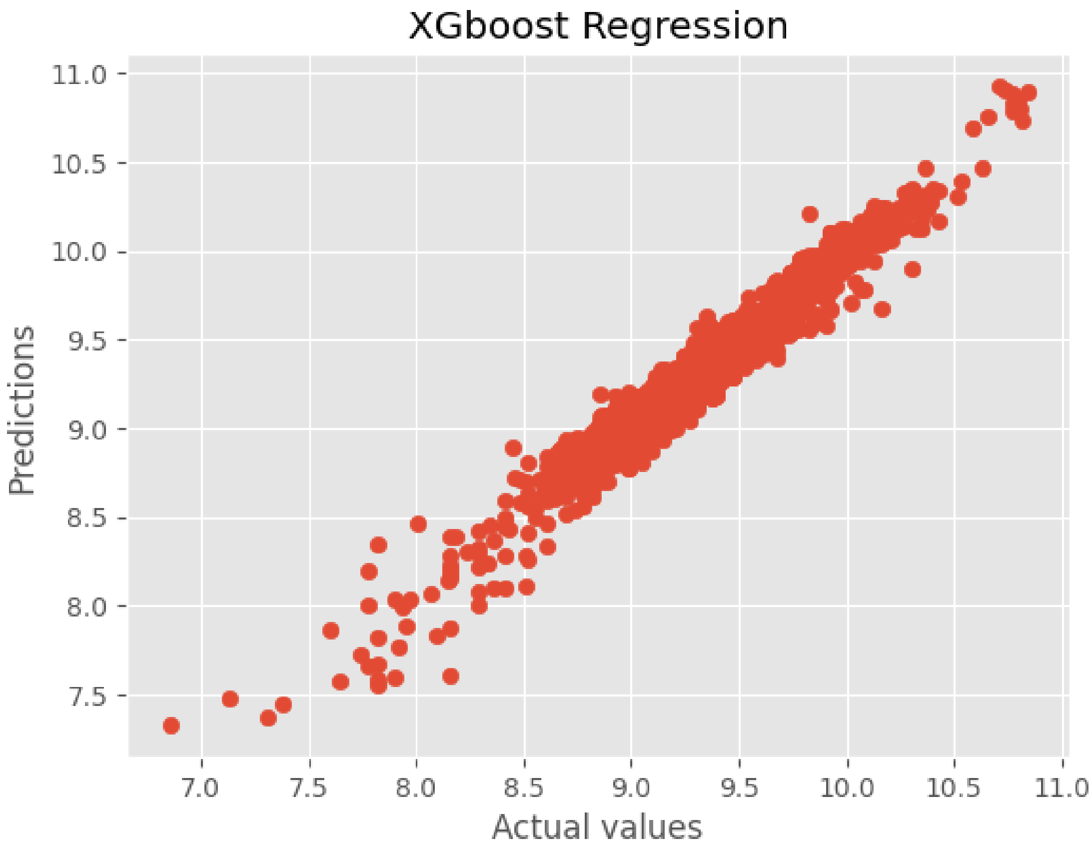
[36]:
```
Text(0.5, 1.0, 'Feature Importance in Random Forest Regression Model')
```



XGBoost Regression Model

[37]:
```
xgb = XGBRegressor(n_estimators=1000, max_depth=3, eta=0.05, subsample=0.7,␣
↪colsample_bytree=0.6,random_state=42)
xgb.fit(X_train,y_train)
y_pred4 = xgb.predict(X_test)
```

[38]:
```
plt.scatter(y_test,y_pred4)
plt.xlabel("Actual values")
plt.ylabel("Predictions")
plt.title('XGboost Regression')
plt.show()
```



[39]:
```
xgb_r2 = xgb.score(X_test, y_test)
print("XGBoost Regressor R-squared: {}".format(xgb_r2))
xgb_mse = mean_squared_error(y_pred4, y_test)
xgb_rmse = np.sqrt(xgb_mse)
print("XGboost Regressor RMSE: {}".format(xgb_rmse))
```

```
XGBoost Regressor R-squared: 0.9617176481502868
XGboost Regressor RMSE: 0.09293538976618351
```

Finding the best parameter for XGboost Regression Model

[40]:
```
# Create the parameter grid.
param_grid1 = {
    'n_estimators': [100,500,1000],
    'max_depth': [3, 9, 15],
    'eta': [0.05, 0.1,0.2],
    'subsample': [0.6, 0.7, 0.8],
    'colsample_bytree': [0.6, 0.8, 0.9]
}
# Create a based model
xgb_r = XGBRegressor()
```

```
# Instantiate the grid search model
grid_search = GridSearchCV(estimator = xgb_r, param_grid = param_grid1,
                          cv = 3, n_jobs = -1, verbose = 1)
```

[41]:
```
# Fit the grid search to the data(**It could take about 12 mins to run the␣
↪gridsearch.)
grid_search.fit(X_train, y_train)
grid_search.best_params_
```

```
Fitting 3 folds for each of 243 candidates, totalling 729 fits
```

[41]:
```
{'colsample_bytree': 0.6,
 'eta': 0.05,
 'max_depth': 3,
 'n_estimators': 1000,
 'subsample': 0.7}
```

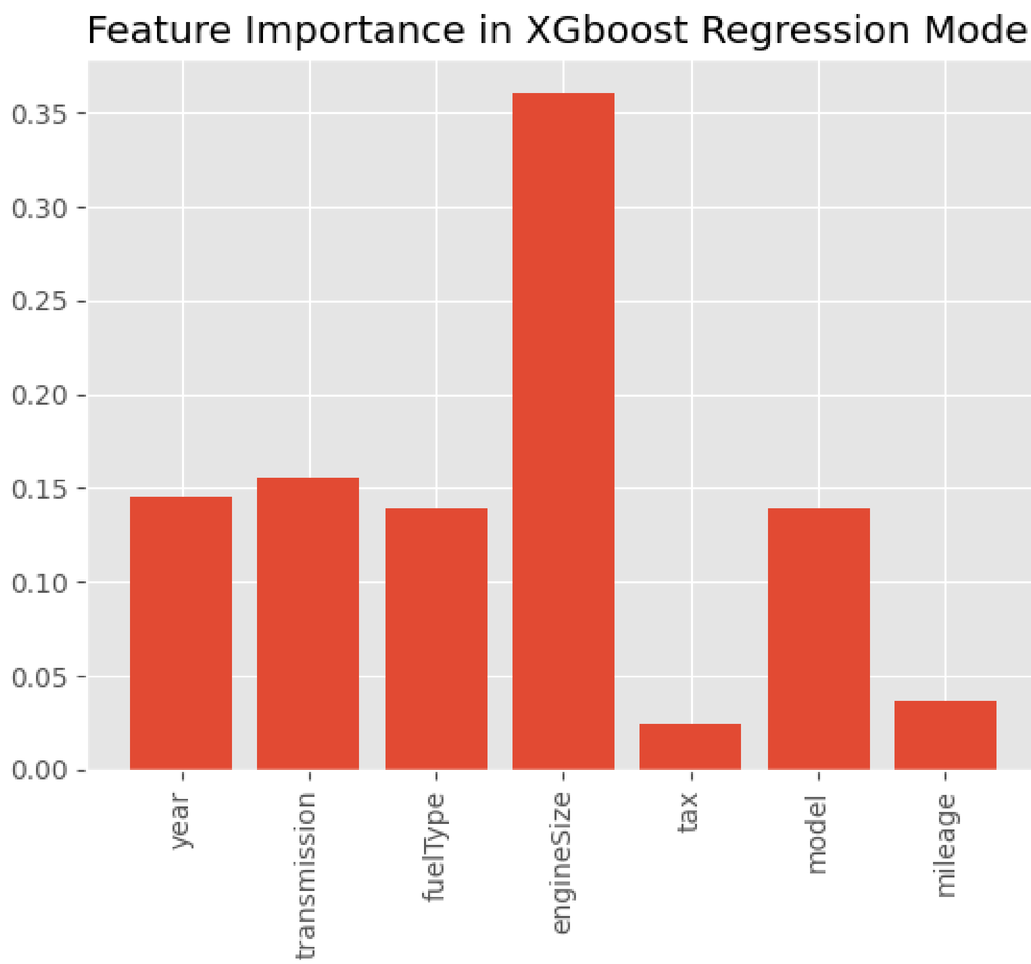
Finding the feature importance

[42]:
```
xgb_importance = xgb.feature_importances_

xgb_f_importance = {}
for i in range(len(feature_cols)):
     xgb_f_importance[feature_cols[i]] = xgb_importance[i]

plt.bar(xgb_f_importance.keys(),xgb_f_importance.values())
plt.xticks(rotation='vertical')
plt.title('Feature Importance in XGboost Regression Model')
```

[42]:
```
Text(0.5, 1.0, 'Feature Importance in XGboost Regression Model')
```



1.3.2 Model Comparison

Metric/Model	Linear Regression	Decision Tree	Random Forest	XGBoost
R-squared	0.8581	0.9379	0.9567	0.9617
RMSE	0.1789	0.1184	0.0988	0.0930

- Linear Regression r2_score: 0.8581849472893514
- Linear Regression Root Mean Squared Error: 0.17887225612192453
- Decision Tree Regressor R-squared: 0.9378957123254383
- Decision Tree Regressor RMSE: 0.11837026281480763
- Random Forest Regressor R-squared: 0.9567323628880114
- Random Forest Regressor RMSE: 0.09880147057473333

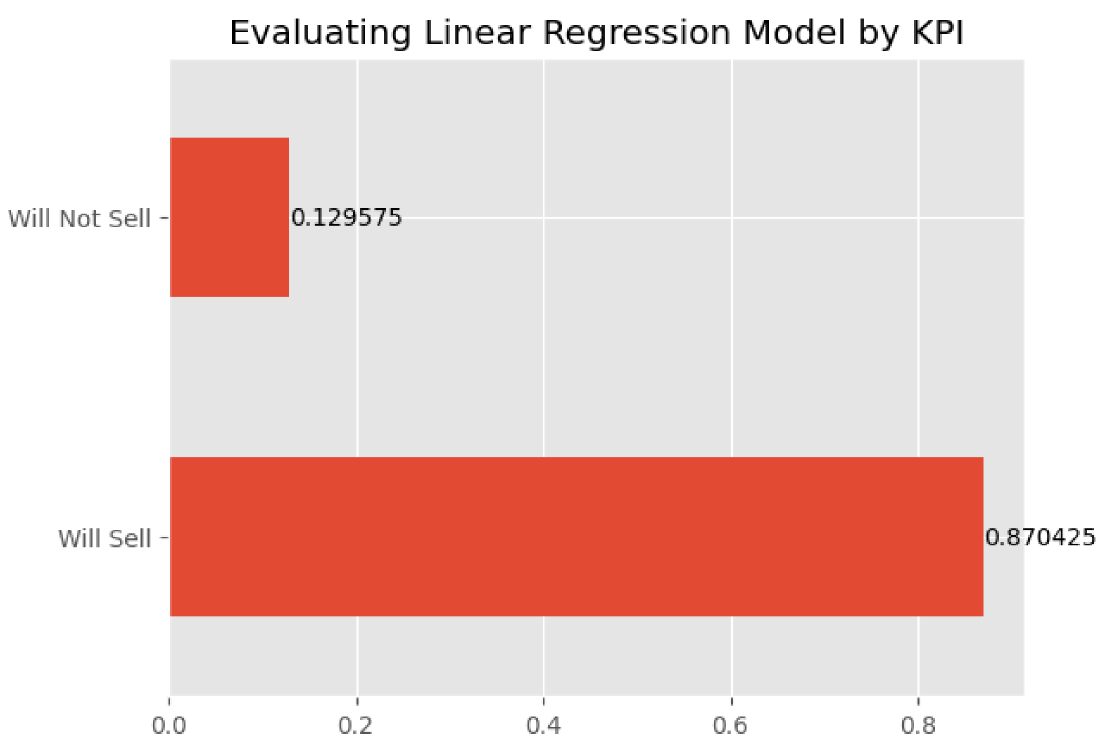
- XGBoost Regressor R-squared: 0.9617176481502868
- XGboost Regressor RMSE: 0.09293538976618351

XGboost model has the highest R-sqaured and lowest RMSE. It means that XGboost model fits the feature best and has lest error in predicting values. However, Decision Tree and Random Forest are also good models, and they are easier to interpret.

1.3.3 Evaluate by Business Criteria

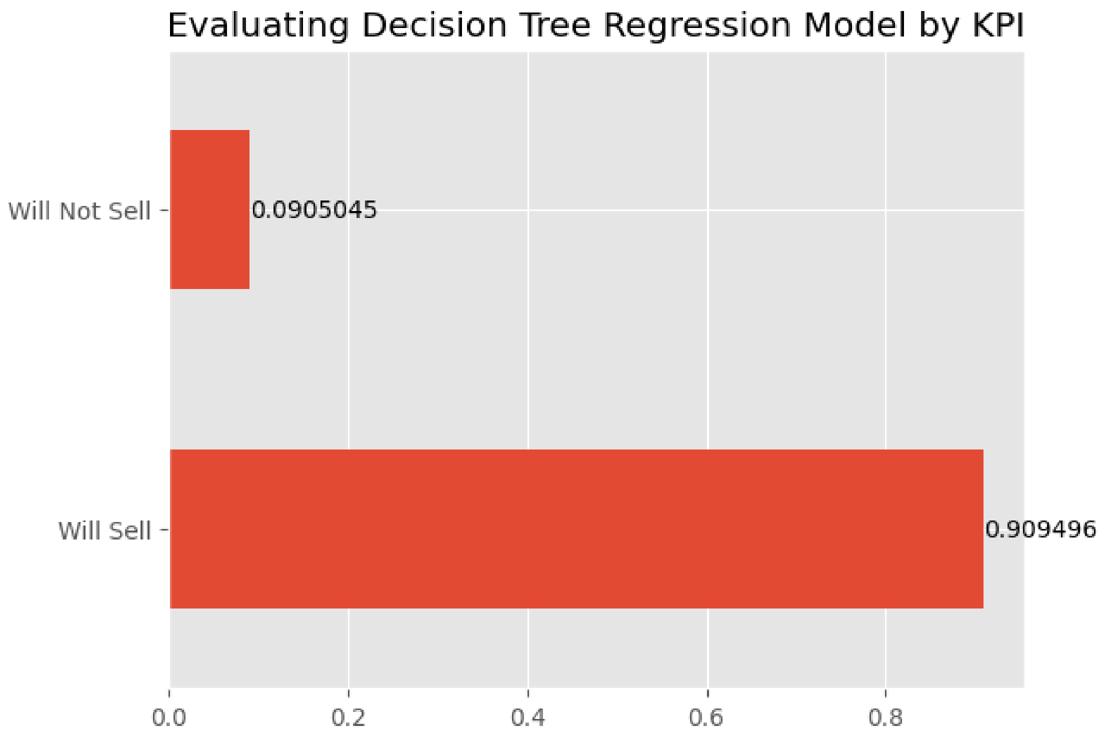
The company wants to avoid prediction out of a range - more than £1500 higher from actual price. Therefore, we would consider using percentage of predictions which predicted price is not more than £1500 higher than actual price as a KPI to compare the 4 models again. The higher the percentage, the better the model performs. 94% of the Random Forest regressor prediction is not more than £1500 higher than actual sell price, while the Linear regression model is 87%, Decision Tree regression model is 91%, and XGboost regression is 93.8%.

[43]:
```
X_test['Predicted_price'] = np.round(np.exp(y_pred),0)
X_test['Price'] = np.round(np.exp(y_test),0)
lr_e = X_test
lr_e['Diff'] = lr_e['Predicted_price'] - lr_e['Price']
lr_e['Result'] =  lr_e['Diff'] > 1500
lr_e['Category'] = lr_e['Result'].apply(lambda x: 'Will Not Sell' if x == True␣
↪else 'Will Sell')
ax = lr_e['Category'].value_counts(normalize=True).plot.barh()
ax.bar_label(ax.containers[0])
ax.set_title('Evaluating Linear Regression Model by KPI');
```



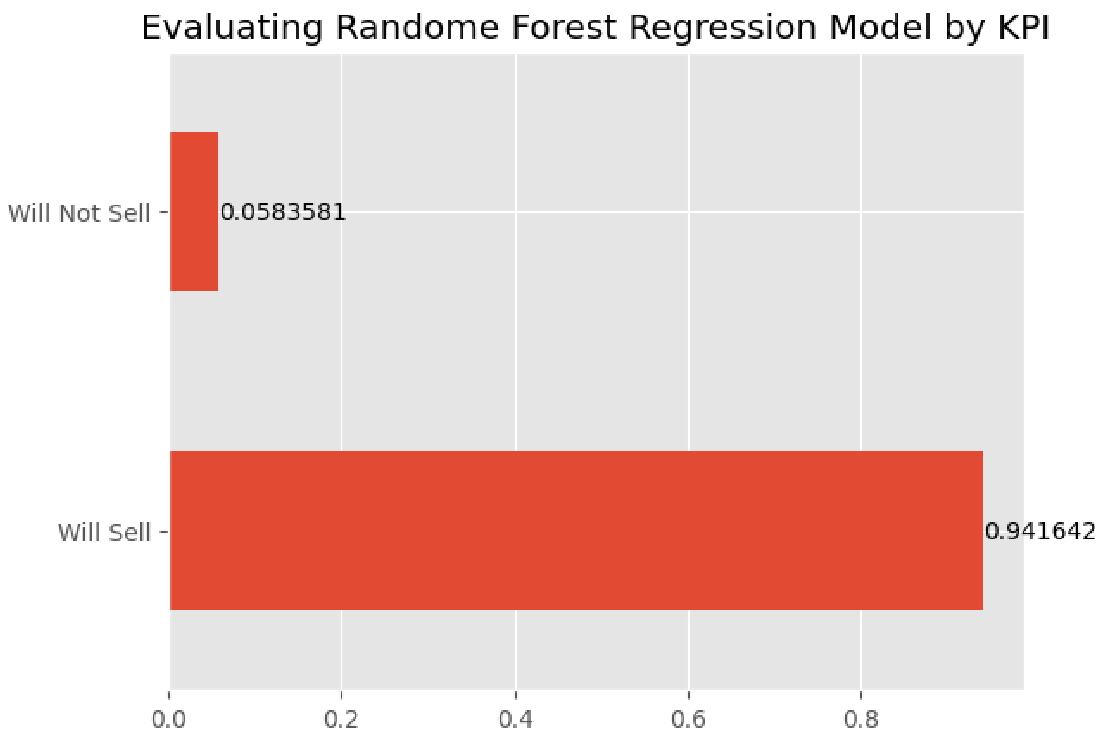
[44]:

```
X_test['Predicted_price'] = np.round(np.exp(y_pred2),0)
X_test['Price'] = np.round(np.exp(y_test),0)
tree_e = X_test
tree_e['Diff'] = tree_e['Predicted_price'] - tree_e['Price']
tree_e['Result'] =  tree_e['Diff'] > 1500
tree_e['Category'] = tree_e['Result'].apply(lambda x: 'Will Not Sell' if x ==␣
↪True else 'Will Sell')
ax = tree_e['Category'].value_counts(normalize=True).plot.barh()
ax.bar_label(ax.containers[0])
ax.set_title('Evaluating Decision Tree Regression Model by KPI');
```

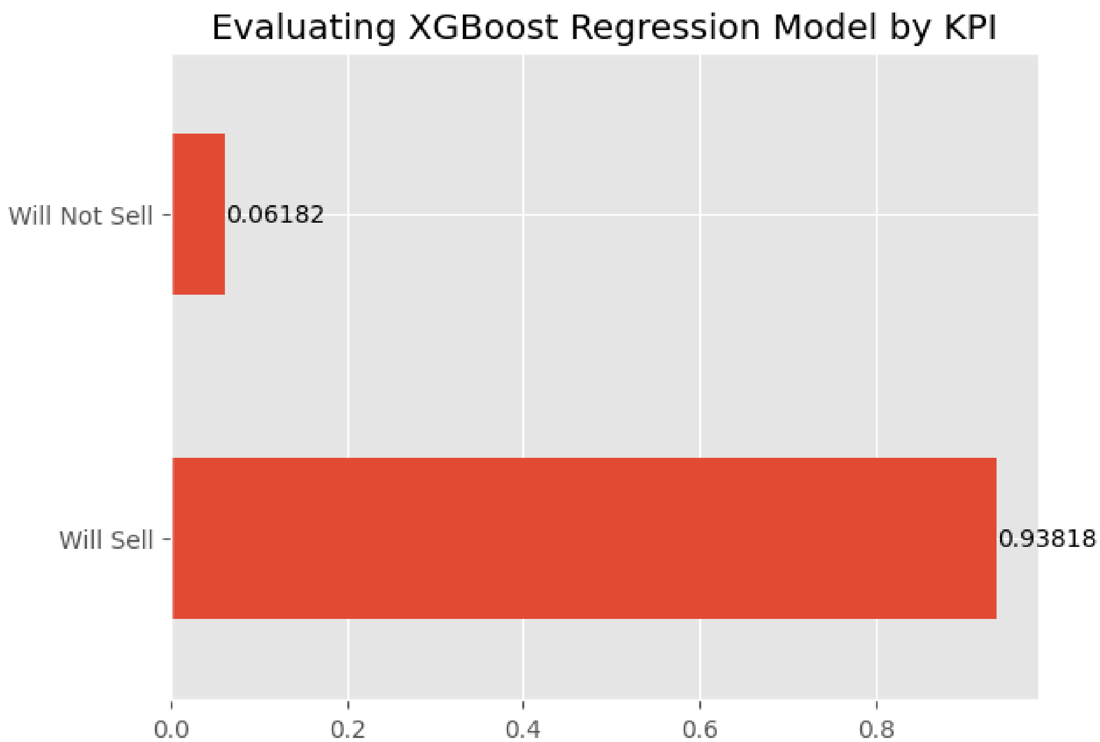


[45]:

```
X_test['Predicted_price'] = np.round(np.exp(y_pred3),0)
X_test['Price'] = np.round(np.exp(y_test),0)
rf_e = X_test
rf_e['Diff'] = rf_e['Predicted_price'] - rf_e['Price']
rf_e['Result'] =  rf_e['Diff'] > 1500
rf_e['Category'] = rf_e['Result'].apply(lambda x: 'Will Not Sell' if x == True␣
↪else 'Will Sell')
ax = rf_e['Category'].value_counts(normalize=True).plot.barh()
ax.bar_label(ax.containers[0])
ax.set_title('Evaluating Randome Forest Regression Model by KPI');
```



[46]:
```
X_test['Predicted_price'] = np.round(np.exp(y_pred4),0)
X_test['Price'] = np.round(np.exp(y_test),0)
xgb_e = X_test
xgb_e['Diff'] = xgb_e['Predicted_price'] - xgb_e['Price']
xgb_e['Result'] = xgb_e['Diff'] > 1500
xgb_e['Category'] = xgb_e['Result'].apply(lambda x: 'Will Not Sell' if x ==␣
 ↪True else 'Will Sell')
ax = xgb_e['Category'].value_counts(normalize=True).plot.barh()
ax.bar_label(ax.containers[0])
ax.set_title('Evaluating XGBoost Regression Model by KPI');
```



1.4 Conclusion

To help junior salespeople better predict the price, we can deploy the Random Forest regression model into production. By implementing this model, about 94% of the prediction will ensure that the cars are very likely to be sold. This will help junior salespeople build their confidence in generating more sales.

To better evaluate whether this model can really help junior salespeople price used cars, I would also recommend A/B testing about using this model to compare two groups of junior salespeople.

To implement and improve the model, I will consider the following steps:

- Looking for best ways to deploy this model in terms of performance and costs. The ideal way is to deploy this machine learning model on edge devices such as mobile and IoT for its convenience and security. However, this might need some work. I will suggest deploying as web services first since it is the easiest way and test the model in newly hired junior salespeople.
- Collecting more data, e.g. time and economic data, the inflation has a huge influence on the car price recently.
- Feature Engineering, e.g reduce the categories in model, create more meaningful features from the variables.