In [7]:
```
import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

In [8]:
```
df =pd.read_csv("CarPricesPrediction.csv")
df
```

Out[8]:

	Unnamed: 0	Make	Model	Year	Mileage	Condition	Price
0	0	Ford	Silverado	2022	18107	Excellent	19094.75
1	1	Toyota	Silverado	2014	13578	Excellent	27321.10
2	2	Chevrolet	Civic	2016	46054	Good	23697.30
3	3	Ford	Civic	2022	34981	Excellent	18251.05
4	4	Chevrolet	Civic	2019	63565	Excellent	19821.85
...	...	...	...	...	...	...	...
995	995	Nissan	Camry	2010	149032	Excellent	24548.50
996	996	Chevrolet	F-150	2014	20608	Excellent	26969.70
997	997	Ford	Altima	2016	109851	Good	20507.55
998	998	Toyota	Silverado	2010	11704	Good	31414.90
999	999	Nissan	Silverado	2017	128390	Excellent	18580.60

1000 rows × 7 columns

In [25]:
```
df.drop("Unnamed: 0",axis=1,inplace=True)
df
```

Out[25]:

	Make	Model	Year	Mileage	Condition	Price
0	Ford	Silverado	2022	18107	Excellent	19094.75
1	Toyota	Silverado	2014	13578	Excellent	27321.10
2	Chevrolet	Civic	2016	46054	Good	23697.30
3	Ford	Civic	2022	34981	Excellent	18251.05
4	Chevrolet	Civic	2019	63565	Excellent	19821.85
...	...	...	...	...	...	...
995	Nissan	Camry	2010	149032	Excellent	24548.50
996	Chevrolet	F-150	2014	20608	Excellent	26969.70
997	Ford	Altima	2016	109851	Good	20507.55
998	Toyota	Silverado	2010	11704	Good	31414.90
999	Nissan	Silverado	2017	128390	Excellent	18580.60

1000 rows × 6 columns

In [21]:
```
X = df.drop(columns=['Price'])
y = df['Price']
```

Out[21]:

	Unnamed: 0	Make	Model	Year	Mileage	Condition	Price
0	0	Ford	Silverado	2022	18107	Excellent	19094.75
1	1	Toyota	Silverado	2014	13578	Excellent	27321.10
2	2	Chevrolet	Civic	2016	46054	Good	23697.30
3	3	Ford	Civic	2022	34981	Excellent	18251.05
4	4	Chevrolet	Civic	2019	63565	Excellent	19821.85
...	...	...	...	...	...	...	...
995	995	Nissan	Camry	2010	149032	Excellent	24548.50
996	996	Chevrolet	F-150	2014	20608	Excellent	26969.70
997	997	Ford	Altima	2016	109851	Good	20507.55
998	998	Toyota	Silverado	2010	11704	Good	31414.90
999	999	Nissan	Silverado	2017	128390	Excellent	18580.60

In [10]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42
```

In [11]:
```
numeric_features = ['Year', 'Mileage']
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])
```

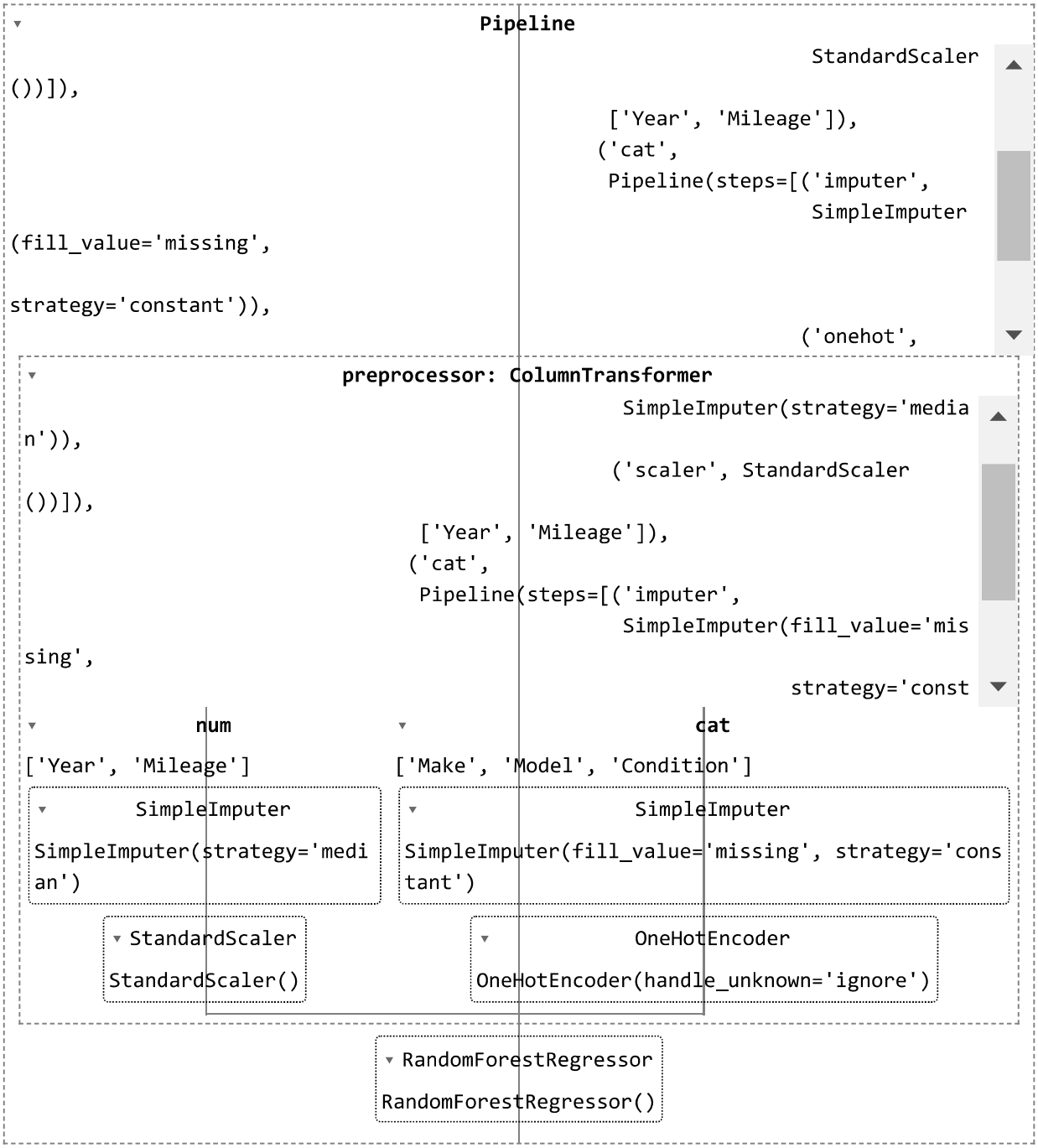
In [12]:
```
categorical_features = ['Make', 'Model', 'Condition']
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))])
```

In [13]:
```
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)])
```

In [14]:
```
regressor = Pipeline(steps=[('preprocessor', preprocessor),
                            ('regressor', RandomForestRegressor())])
```

In [15]:
```
regressor.fit(X_train, y_train)
```

Out[15]:



In [16]:
```
y_pred = regressor.predict(X_test)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)
print(f'Root Mean Squared Error: {rmse}')
print(f'R-squared: {r2}')
```

```
Root Mean Squared Error: 135.27231033182338
R-squared: 0.9990969527286926
```

In [19]:
```
def predict_price(make, model, year, mileage, condition):
    input_data = pd.DataFrame([[year, mileage, make, model, condition]], columns=['Year'
    predicted_price = regressor.predict(input_data)
    return predicted_price[0]
```

In [20]:
```
predicted_price = predict_price('Toyota', 'Camry', 2018, 50000, 'Good')
print(f'Predicted price: ${predicted_price:.2f}')
```

```
Predicted price: $21480.26
```

In [ ]: