In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
%matplotlib inline
mpl.style.use('ggplot')
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
car=pd.read_csv("C:/Users/Naveed/Downloads/quikr_car.csv")
```

In [4]:
```
backup=car.copy
```

In [5]:
```
car=car[car['year'].str.isnumeric()]
```

In [6]:
```
car['year']=car['year'].astype(int)
```

In [7]:
```
car=car[car['Price']!="Ask For Price"]
```

In [8]:
```
car['Price']=car['Price'].str.replace(',','').astype(int)
```

In [9]:
```
car['kms_driven']=car['kms_driven'].str.split().str.get(0).str.replace(',','')
```

In [10]:
```
car=car[car['kms_driven'].str.isnumeric()]
car['kms_driven']=car['kms_driven'].astype(int)
```

In [11]:
```
car=car[~car['fuel_type'].isna()]
car.shape
```

Out[11]:
```
(816, 6)
```

In [12]:
```
car['name']=car['name'].str.split().str.slice(start=0,stop=3).str.join(' ')
```

In [13]:
```
car=car.reset_index(drop=True)
```

In [14]:
```
car.to_csv('Cleaned_Car1.csv')
```

In [15]:
```
car=car[car['Price']<6000000]
```

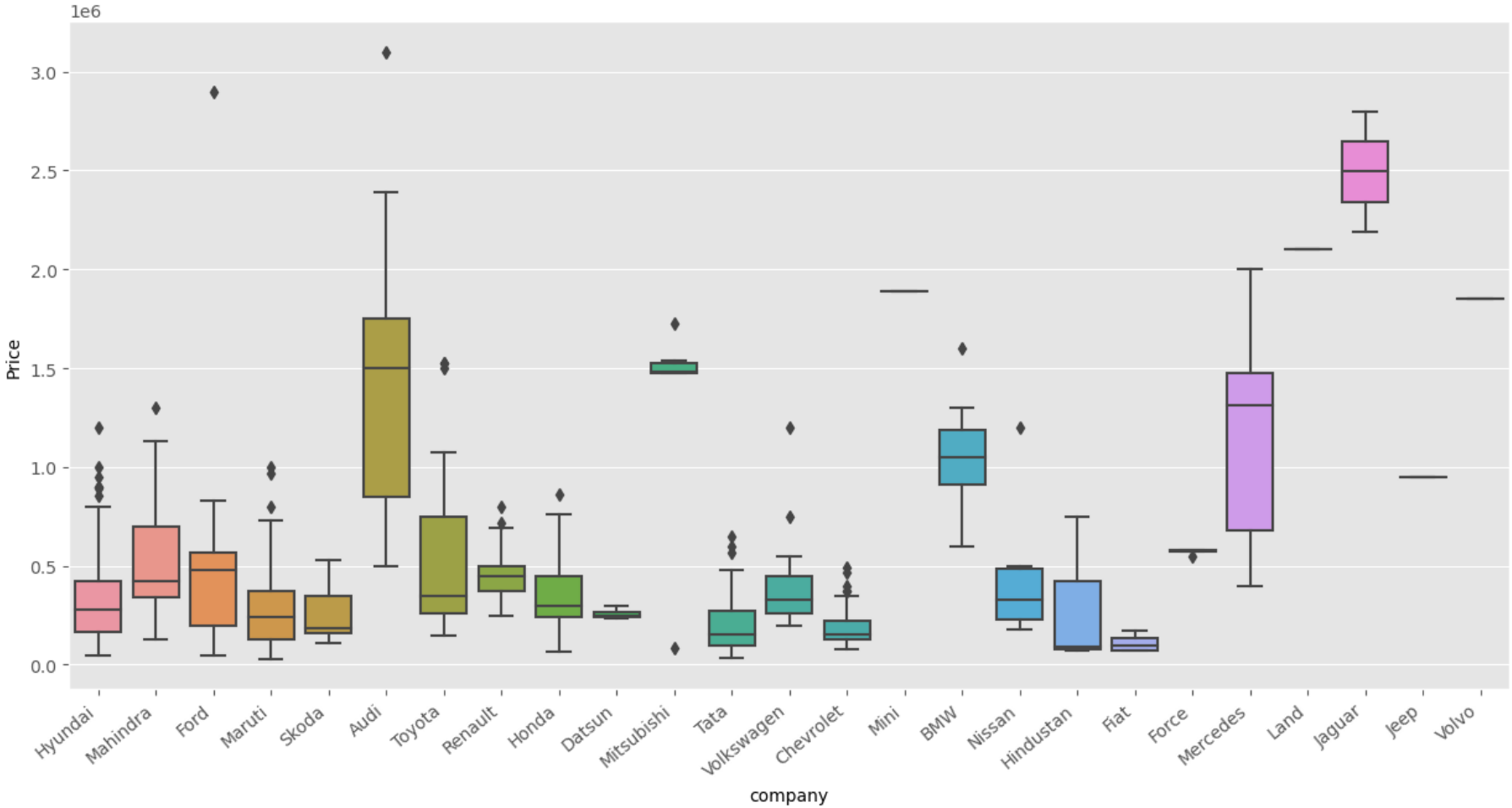
In [16]:
```
car['company'].unique()
```

Out[16]:
```
array(['Hyundai', 'Mahindra', 'Ford', 'Maruti', 'Skoda', 'Audi', 'Toyota',
       'Renault', 'Honda', 'Datsun', 'Mitsubishi', 'Tata', 'Volkswagen',
       'Chevrolet', 'Mini', 'BMW', 'Nissan', 'Hindustan', 'Fiat', 'Force',
       'Mercedes', 'Land', 'Jaguar', 'Jeep', 'Volvo'], dtype=object)
```

In [17]:
```
import seaborn as sns
```

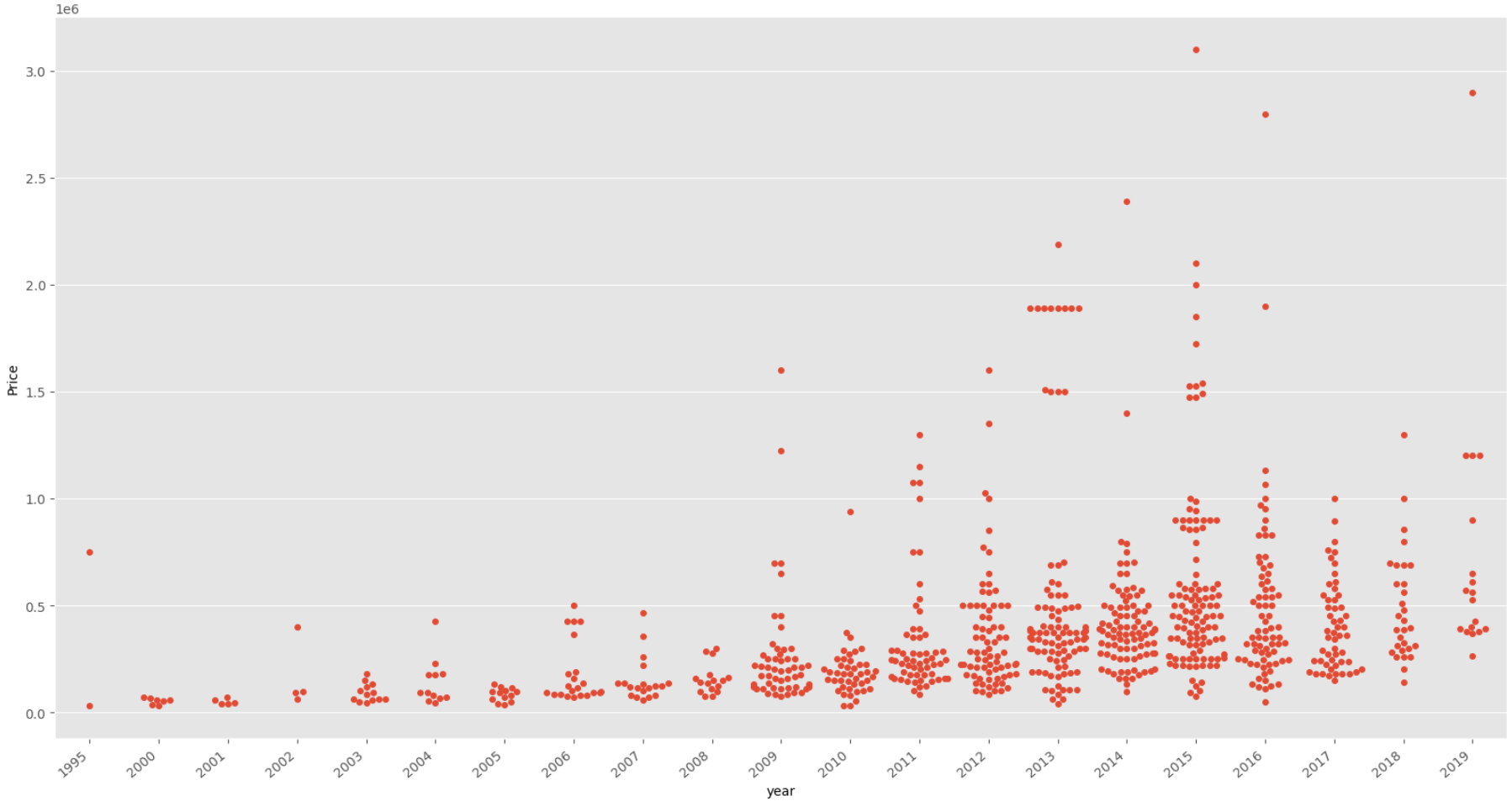
In [18]:
```
plt.subplots(figsize=(15,7))
ax=sns.boxplot(x='company',y='Price',data=car)
ax.set_xticklabels(ax.get_xticklabels(),rotation=40,ha='right')
```

Out[18]:
```
[Text(0, 0, 'Hyundai'),
 Text(1, 0, 'Mahindra'),
 Text(2, 0, 'Ford'),
 Text(3, 0, 'Maruti'),
 Text(4, 0, 'Skoda'),
 Text(5, 0, 'Audi'),
 Text(6, 0, 'Toyota'),
 Text(7, 0, 'Renault'),
 Text(8, 0, 'Honda'),
 Text(9, 0, 'Datsun'),
 Text(10, 0, 'Mitsubishi'),
 Text(11, 0, 'Tata'),
 Text(12, 0, 'Volkswagen'),
 Text(13, 0, 'Chevrolet'),
 Text(14, 0, 'Mini'),
 Text(15, 0, 'BMW'),
 Text(16, 0, 'Nissan'),
 Text(17, 0, 'Hindustan'),
 Text(18, 0, 'Fiat'),
 Text(19, 0, 'Force'),
 Text(20, 0, 'Mercedes'),
 Text(21, 0, 'Land'),
 Text(22, 0, 'Jaguar'),
 Text(23, 0, 'Jeep'),
 Text(24, 0, 'Volvo')]
```


1e6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Price
Hyundai
Mahindra
Ford
Maruti
Skoda
Audi
Toyota
Renault
Honda
Datsun
Mitsubishi
Tata
Volkswagen
Chevrolet
Mini
BMW
Nissan
Hindustan
Fiat
Force
Mercedes
Land
Jaguar
Jeep
Volvo
company

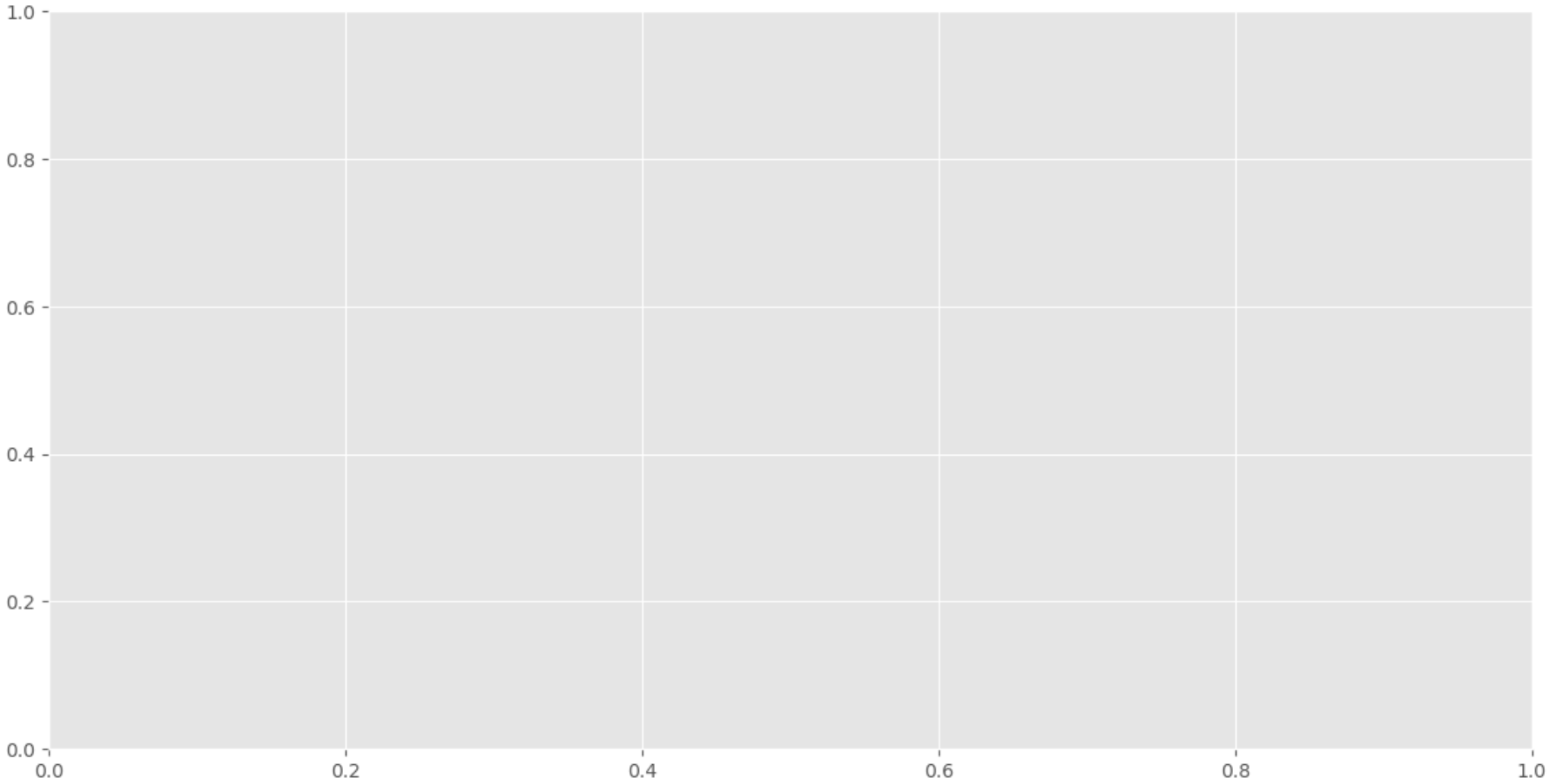
In [19]:
```
plt.subplots(figsize=(20,10))
ax=sns.swarmplot(x='year',y='Price',data=car)
ax.set_xticklabels(ax.get_xticklabels(),rotation=40,ha='right')
```

Out[19]:
```
[Text(0, 0, '1995'),
 Text(1, 0, '2000'),
 Text(2, 0, '2001'),
 Text(3, 0, '2002'),
 Text(4, 0, '2003'),
 Text(5, 0, '2004'),
 Text(6, 0, '2005'),
 Text(7, 0, '2006'),
 Text(8, 0, '2007'),
 Text(9, 0, '2008'),
 Text(10, 0, '2009'),
 Text(11, 0, '2010'),
 Text(12, 0, '2011'),
 Text(13, 0, '2012'),
 Text(14, 0, '2013'),
 Text(15, 0, '2014'),
 Text(16, 0, '2015'),
 Text(17, 0, '2016'),
 Text(18, 0, '2017'),
 Text(19, 0, '2018'),
 Text(20, 0, '2019')]
```

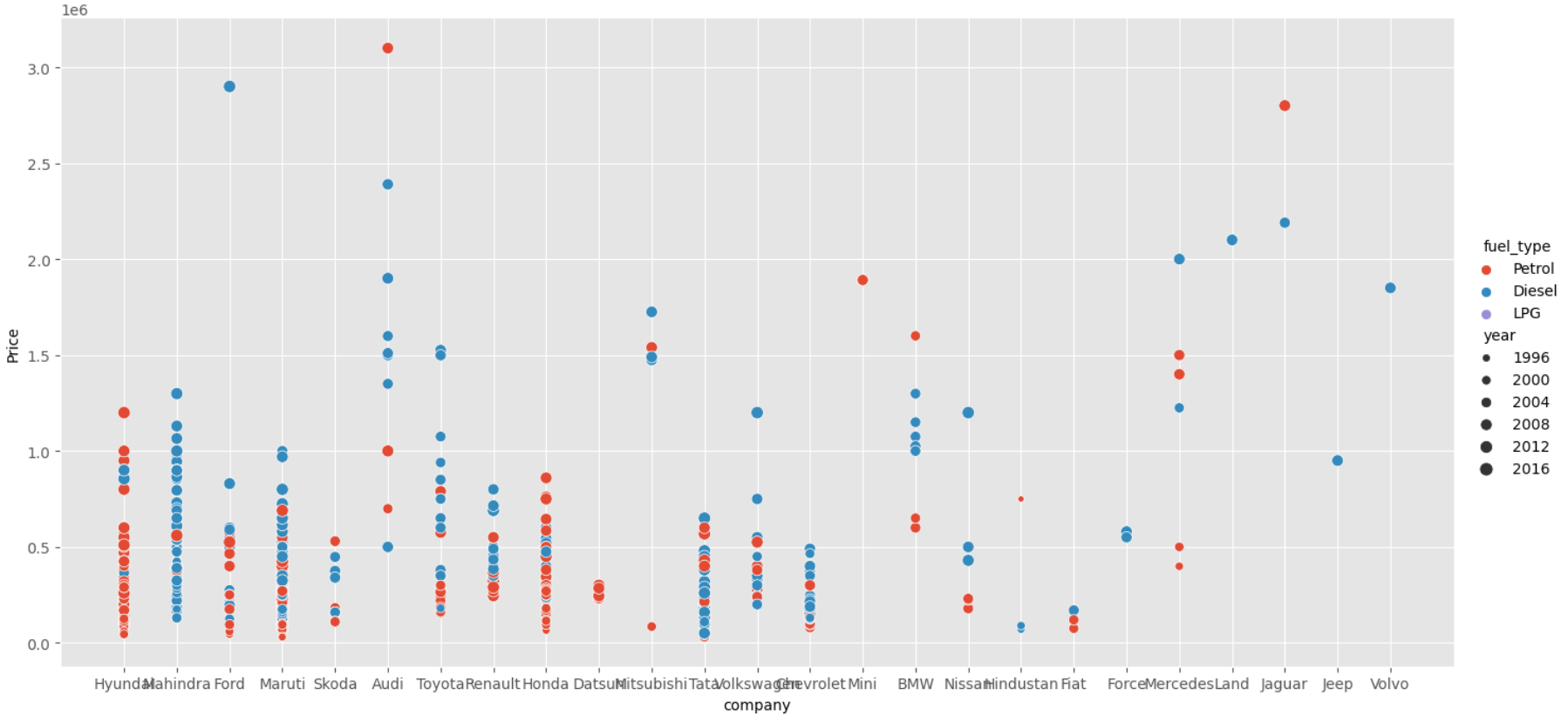

1e6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Price
1995
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
year

In [20]:
```
plt.subplots(figsize=(14,7))
```

Out[20]:
```
(<Figure size 1400x700 with 1 Axes>, <Axes: >)
```



In [21]:
```
ax=sns.relplot(x='company',y='Price',data=car,hue='fuel_type',size='year',height=7,aspect=2)
```



In [22]:
```
X=car[['name','company','year','kms_driven','fuel_type']]
y=car['Price']
```

In [23]:
```
y.shape
```

Out[23]:
```
(815,)
```

In [24]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)
```

In [25]:
```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline
from sklearn.metrics import r2_score
```

In [26]:
```
ohe=OneHotEncoder()
ohe.fit(X[['name','company','fuel_type']])
```

Out[26]:
```
OneHotEncoder
OneHotEncoder()
```

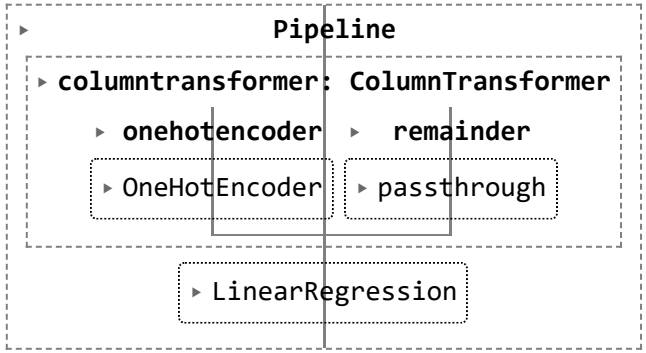
In [27]:
```
column_trans=make_column_transformer((OneHotEncoder(categories=ohe.categories_),['name','company','fuel_type']),
 remainder='passthrough')
```

In [28]:
```
lr=LinearRegression()
```

In [29]:
```
pipe=make_pipeline(column_trans,lr)
```

In [30]:
```
pipe.fit(X_train,y_train)
```

Out[30]:



In [31]:
```
y_pred=pipe.predict(X_test)
```

In [34]:
```
r2_score(y_test,y_pred)
```

Out[34]:
```
0.6540033632627376
```

In [39]:
```
scores=[]
for i in range(100):
    X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=i)
    lr=LinearRegression()
    pipe=make_pipeline(column_trans,lr)
    pipe.fit(X_train,y_train)
    y_pred=pipe.predict(X_test)
    scores.append(r2_score(y_test,y_pred))
```

In [40]:
```
np.argmax(scores)
```

Out[40]:
```
56
```

In [41]:
```
scores[np.argmax(scores)]
```

Out[41]:
```
0.8598801022532936
```

In [42]:
```
pipe.predict(pd.DataFrame(columns=X_test.columns,data=np.array(['Maruti Suzuki Swift','Maruti',2019,100,'Petrol']).res
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=np.argmax(scores))
```

In [43]:
```
lr=LinearRegression()
pipe=make_pipeline(column_trans,lr)
pipe.fit(X_train,y_train)
y_pred=pipe.predict(X_test)
r2_score(y_test,y_pred)
```

Out[43]:
```
0.8598801022532936
```

In [44]:
```
import pickle
```

In [45]:
```
pickle.dump(pipe,open('LinearRegressionModel.pkl','wb'))
```

In [46]:
```
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
import pandas as pd
import pickle
```

In [47]:
```
model = pickle.load(open('LinearRegressionModel.pkl', 'rb'))
def predict_price():
    try:
        name = name_entry.get()
        company = company_entry.get()
        year = int(year_entry.get())
        kms_driven = int(kms_entry.get())
        fuel_type = fuel_type_var.get()
 # Make a prediction using the model
        prediction = model.predict(pd.DataFrame(columns=['name', 'company', 'year', 'kms_driven', 'fuel_type'],
        data=[[name, company, year, kms_driven, fuel_type]]))
 # Display the predicted price
        result_label.config(text=f"Predicted Price: {prediction[0]:,.2f} INR")
    except Exception as e:
        messagebox.showerror("Error", f"An error occurred: {str(e)}")
```

In [48]:
```
root = tk.Tk()
root.title("Car Price Prediction")
root.geometry("1000x1000")
# Create and pack GUI elements
name_label = ttk.Label(root, text="Car Name:")
name_entry = ttk.Entry(root, width=30)
company_label = ttk.Label(root, text="Company:")
company_entry = ttk.Entry(root, width=30)
year_label = ttk.Label(root, text="Year:")
year_entry = ttk.Entry(root, width=30)
kms_label = ttk.Label(root, text="Kilometers Driven:")
kms_entry = ttk.Entry(root, width=30)
fuel_type_label = ttk.Label(root, text="Fuel Type:")
fuel_type_var = tk.StringVar()
fuel_type_combobox = ttk.Combobox(root, textvariable=fuel_type_var, values=['Petrol', 'Diesel', 'CNG', 'LPG'])
predict_button = ttk.Button(root, text="Predict Price", command=predict_price)
result_label = ttk.Label(root, text="Predicted Price: ")
# Arrange elements using the grid layout
name_label.grid(row=0, column=0, pady=5, padx=5, sticky="e")
name_entry.grid(row=0, column=1, pady=5, padx=5)
company_label.grid(row=1, column=0, pady=5, padx=5, sticky="e")
company_entry.grid(row=1, column=1, pady=5, padx=5)
year_label.grid(row=2, column=0, pady=5, padx=5, sticky="e")
year_entry.grid(row=2, column=1, pady=5, padx=5)
kms_label.grid(row=3, column=0, pady=5, padx=5, sticky="e")
kms_entry.grid(row=3, column=1, pady=5, padx=5)
fuel_type_label.grid(row=4, column=0, pady=5, padx=5, sticky="e")
fuel_type_combobox.grid(row=4, column=1, pady=5, padx=5)
predict_button.grid(row=5, column=0, columnspan=2, pady=10)
result_label.grid(row=6, column=0, columnspan=2, pady=5)
# Start the GUI event loop
root.mainloop()
```

In [ ]: