```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
%matplotlib inline
mpl.style.use('ggplot')
```

"@hidden_cell" is not an allowed annotation - allowed values include [@param, @title, @markdown].

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It incluc
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='fafCRw8nYLmybxQIleCNZTTgAh1UeXsbWA5F308_03qz',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'carresalevalueprediction-donotdelete-pr-2twhxsxnqpdh0j'
object_key = 'quikr_car.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, bc

df_data_1 = pd.read_csv(body)
df_data_1.head()
```

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
2	Maruti Suzuki Alto 800 Vxi	Maruti	2018	Ask For Price	22,000 kms	Petrol
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol

```
df_data_1.shape
```

```
(892, 6)
```

```
df_data_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 892 entries, 0 to 891
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        892 non-null    object
 1   company     892 non-null    object
 2   year        892 non-null    object
 3   Price       892 non-null    object
 4   kms_driven  840 non-null    object
 5   fuel_type   837 non-null    object
dtypes: object(6)
memory usage: 41.9+ KB
```

```
backup = df_data_1.copy()
```

```
df_data_1=df_data_1[df_data_1['year'].str.isnumeric()]
```

```
df_data_1['year']=df_data_1['year'].astype(int)
```

```
df_data_1=df_data_1[df_data_1['Price']!='Ask For Price']
```

```
df_data_1['Price']=df_data_1['Price'].str.replace(',','').astype(int)
```

```
df_data_1['kms_driven']=df_data_1['kms_driven'].str.split().str.get(0).str.replace(',','')
```

```
df_data_1=df_data_1[df_data_1['kms_driven'].str.isnumeric()]
```

```
df_data_1['kms_driven']=df_data_1['kms_driven'].astype(int)
```

```
df_data_1=df_data_1[~df_data_1['fuel_type'].isna()]
```

```
df_data_1.shape
```

```
(816, 6)
```

```
df_data_1['name']=df_data_1['name'].str.split().str.slice(start=0,stop=3).str.join(' ')
```

```
df_data_1=df_data_1.reset_index(drop=True)
```

```
df_data_1
```

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
812	Tata Indica V2	Tata	2009	110000	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
814	Tata Zest XM	Tata	2018	260000	27000	Diesel

```
df_data_1.to_csv('Cleaned_df_data_1_data.csv')
```

816 rows × 6 columns

```
df_data_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        816 non-null    object
 1   company     816 non-null    object
 2   year        816 non-null    int64 
 3   Price       816 non-null    int64 
 4   kms_driven  816 non-null    int64 
 5   fuel_type   816 non-null    object
dtypes: int64(3), object(3)
memory usage: 38.4+ KB
```

```
df_data_1.describe(include='all')
```

	name	company	year	Price	kms_driven	fuel_type
count	816	816	816.000000	8.160000e+02	816.000000	816
unique	254	25	NaN	NaN	NaN	3

```
df_data_1=df_data_1[df_data_1['Price']<6000000]
```

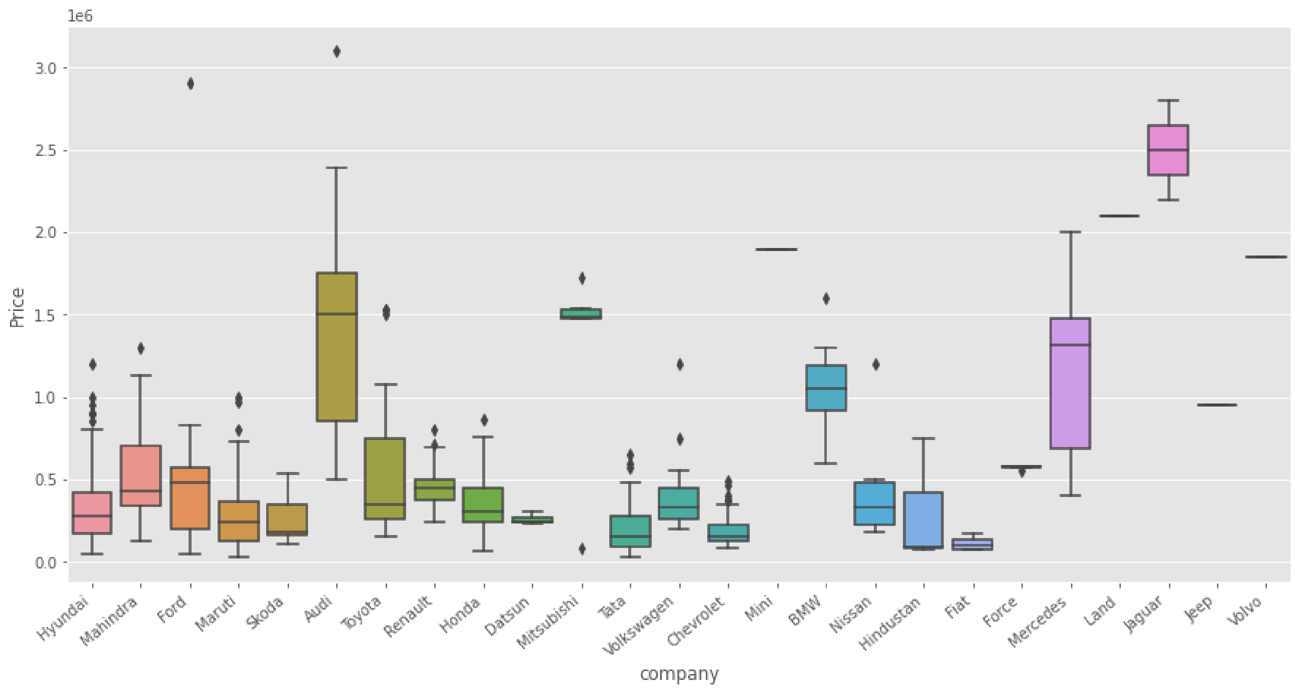
```
df_data_1['company'].unique()
```

```
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)
```

25%	NaN	NaN	2010.000000	1.750000e+05	27000.000000	NaN

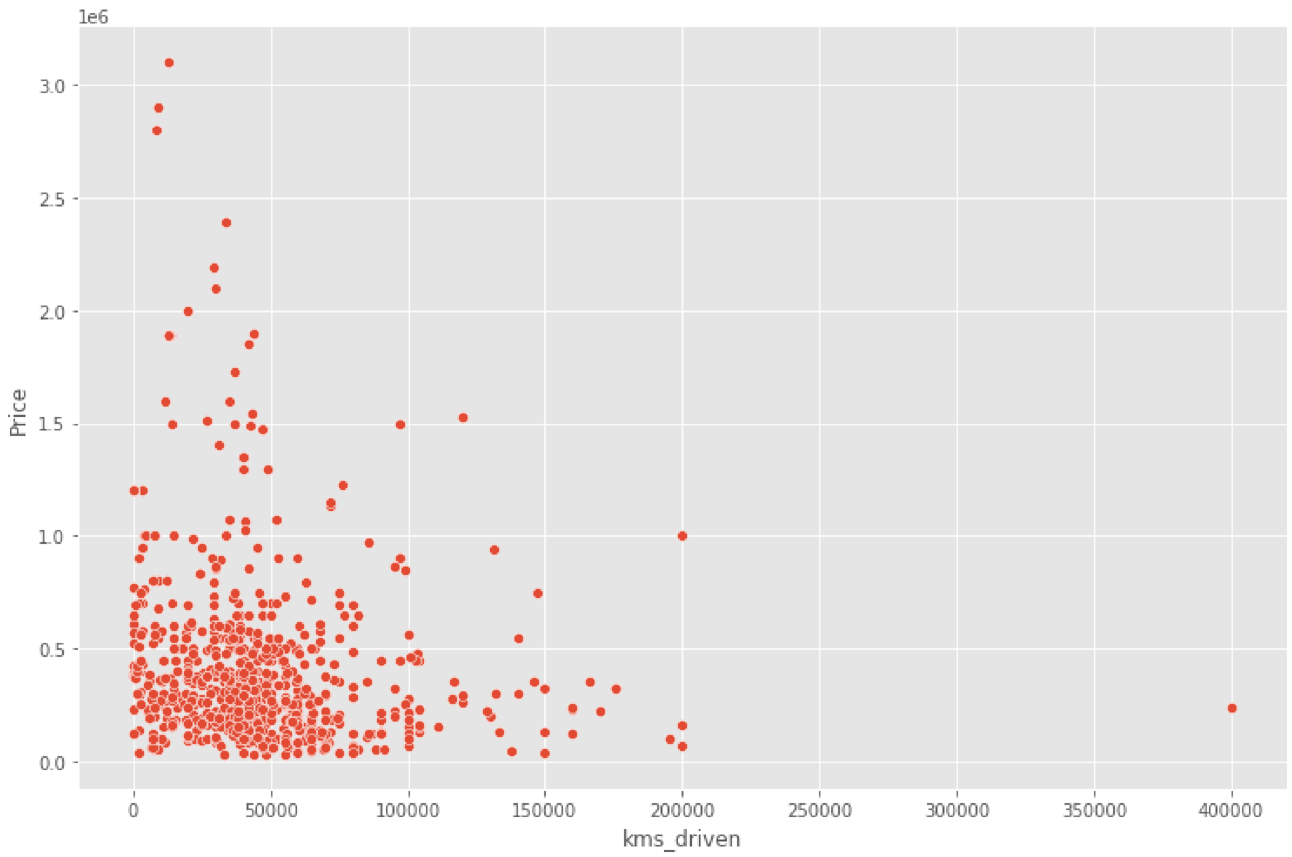
```
import seaborn
```

```
plt.subplots(figsize=(15,7))
ax=seaborn.boxplot(x='company',y='Price',data=df_data_1)
ax.set_xticklabels(ax.get_xticklabels(),rotation=40,ha='right')
plt.show()
```



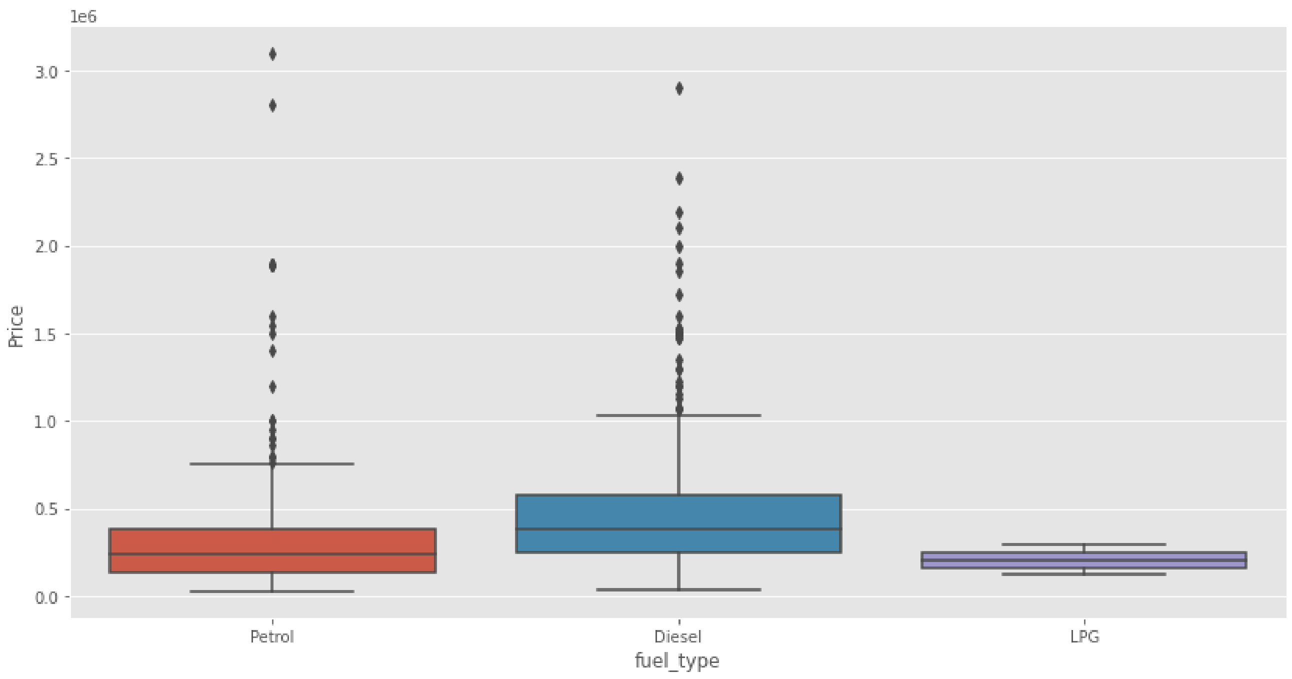
```
seaborn.relplot(x='kms_driven',y='Price',data=df_data_1,height=7,aspect=1.5)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f1034b9d0d0>
```



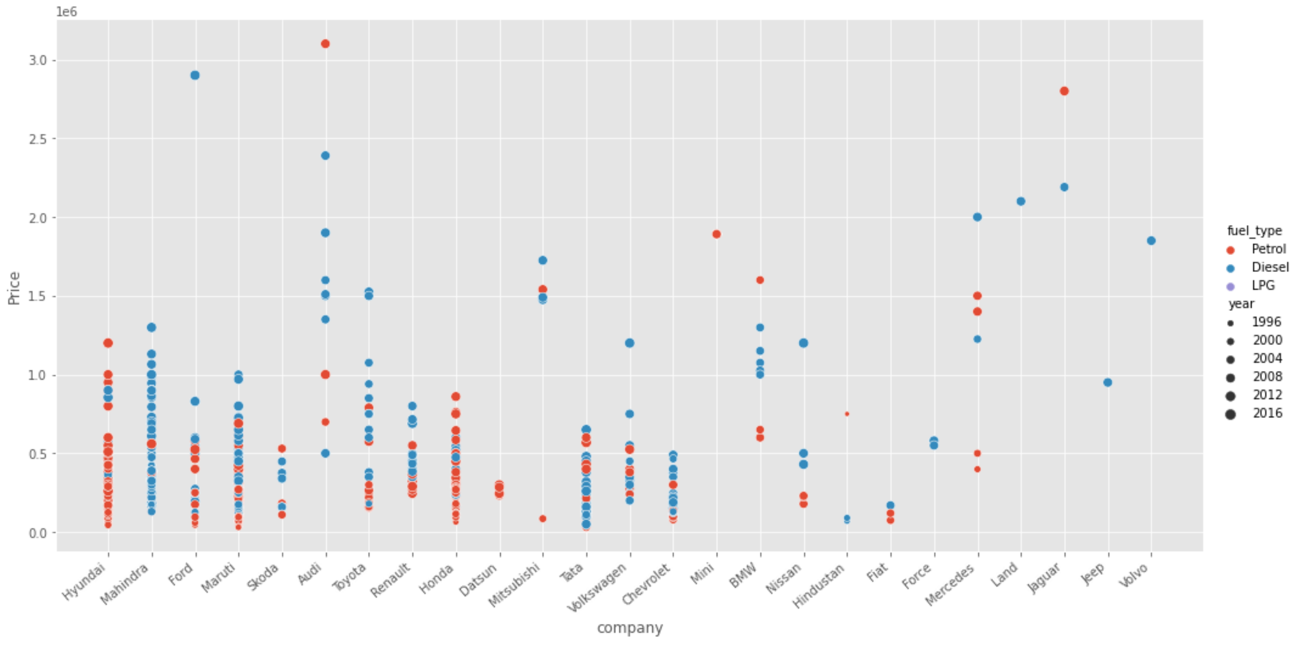
```
plt.subplots(figsize=(14,7))
seaborn.boxplot(x='fuel_type',y='Price',data=df_data_1)
```

```
<AxesSubplot:xlabel='fuel_type', ylabel='Price'>
```



```
ax=seaborn.relplot(x='company',y='Price',data=df_data_1,hue='fuel_type',size='year',height
ax.set_xticklabels(rotation=40,ha='right')
```

```
<seaborn.axisgrid.FacetGrid at 0x7f1034d4a7f0>
```



```
X=df_data_1[['name','company','year','kms_driven','fuel_type']]
y=df_data_1['Price']
```

```
X
```

	name	company	year	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	40	Diesel

```
y.shape
```

```
(815,)
```

4	Ford Figo	Ford	2012	41000	Diesel

```
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)
```

```
from sklearn.linear_model import LinearRegression
```

813	Toyota Corolla Altis	Toyota	2009	132000	Petrol

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline
```

815 rows × 5 columns

```
from sklearn.metrics import r2_score
```

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

```
OneHotEncoder()
```

```
column_trans=make_column_transformer((OneHotEncoder(categories=ohe.categories_),['name','c
                                    remainder='passthrough')
```

```
lr=LinearRegression()
```

```
pipe=make_pipeline(column_trans,lr)
```

```
pipe.fit(X_train,y_train)
```

```
Pipeline(steps=[('columntransformer',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('onehotencoder',
                                                  OneHotEncoder(categories=
[array(['Audi\xa0A3\xa0Cabriolet', 'Audi\xa0A4\xa01.8',
       'Audi\xa0A4\xa02.0', 'Audi\xa0A6\xa02.0', 'Audi\xa0A8',
       'Audi\xa0Q3\xa02.0', 'Audi\xa0Q5\xa02.0', 'Audi\xa0Q7',
       'BMW\xa03\xa0Series', 'BMW\xa05\xa0Series', 'BMW\xa07\xa0...

array(['Audi', 'BMW', 'Chevrolet', 'Datsun', 'Fiat', 'Force', 'Ford',
       'Hindustan', 'Honda', 'Hyundai', 'Jaguar', 'Jeep', 'Land',
       'Mahindra', 'Maruti', 'Mercedes', 'Mini', 'Mitsubishi', 'Nissan',
       'Renault', 'Skoda', 'Tata', 'Toyota', 'Volkswagen', 'Volvo'],
      dtype=object),
     array(['Diesel', 'LPG', 'Petrol'], dtype=object)]),
                                                  ['name', 'company',
                                                   'fuel_type'])])),
                     ('linearregression', LinearRegression())])
```

```
y_pred=pipe.predict(X_test)
```

```
r2_score(y_test,y_pred)
```

```
     0.7342105166205257
```

```
scores=[]
for i in range(1000):
    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))
```

```
np.argmax(scores)
```

```
     655
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=np.argmax(sc
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)
```

```
     0.920087093218515
```

```
import pickle
```

```
pickle.dump(pipe,open('LinearRegressionModel.pkl','wb'))
```

```
pipe.steps[0][1].transformers[0][1].categories[0]
```

```
     array(['Audi\xa0A3\xa0Cabriolet', 'Audi\xa0A4\xa01.8',
            'Audi\xa0A4\xa02.0', 'Audi\xa0A6\xa02.0', 'Audi\xa0A8',
            'Audi\xa0Q3\xa02.0', 'Audi\xa0Q5\xa02.0', 'Audi\xa0Q7',
            'BMW\xa03\xa0Series', 'BMW\xa05\xa0Series', 'BMW\xa07\xa0Series',
            'BMW\xa0X1', 'BMW\xa0X1\xa0sDrive20d', 'BMW\xa0X1\xa0xDrive20d',
            'Chevrolet\xa0Beat', 'Chevrolet\xa0Beat\xa0Diesel',
            'Chevrolet\xa0Beat\xa0LS', 'Chevrolet\xa0Beat\xa0LT',
            'Chevrolet\xa0Beat\xa0PS', 'Chevrolet\xa0Cruze\xa0LTZ',
            'Chevrolet\xa0Enjoy', 'Chevrolet\xa0Enjoy\xa01.4',
            'Chevrolet\xa0Sail\xa01.2', 'Chevrolet\xa0Sail\xa0UVA',
            'Chevrolet\xa0Spark', 'Chevrolet\xa0Spark\xa01.0',
            'Chevrolet\xa0Spark\xa0LS', 'Chevrolet\xa0Spark\xa0LT',
            'Chevrolet\xa0Tavera\xa0LS', 'Chevrolet\xa0Tavera\xa0Neo',
            'Datsun\xa0GO\xa0T', 'Datsun\xa0Go\xa0Plus',
            'Datsun\xa0Redi\xa0GO', 'Fiat\xa0Linea\xa0Emotion',
            'Fiat\xa0Petra\xa0ELX', 'Fiat\xa0Punto\xa0Emotion',
            'Force\xa0Motors\xa0Force', 'Force\xa0Motors\xa0One',
            'Ford\xa0EcoSport', 'Ford\xa0EcoSport\xa0Ambiente',
            'Ford\xa0EcoSport\xa0Titanium', 'Ford\xa0EcoSport\xa0Trend',
            'Ford\xa0Endeavor\xa04x4', 'Ford\xa0Fiesta',
            'Ford\xa0Fiesta\xa0SXi', 'Ford\xa0Figo', 'Ford\xa0Figo\xa0Diesel',
            'Ford\xa0Figo\xa0Duratorq', 'Ford\xa0Figo\xa0Petrol',
            'Ford\xa0Fusion\xa01.4', 'Ford\xa0Ikon\xa01.3',
            'Ford\xa0Ikon\xa01.6', 'Hindustan\xa0Motors\xa0Ambassador',
            'Honda\xa0Accord', 'Honda\xa0Amaze', 'Honda\xa0Amaze\xa01.2',
            'Honda\xa0Amaze\xa01.5', 'Honda\xa0Brio', 'Honda\xa0Brio\xa0V',
            'Honda\xa0Brio\xa0VX', 'Honda\xa0City', 'Honda\xa0City\xa01.5',
            'Honda\xa0City\xa0SV', 'Honda\xa0City\xa0VX',
            'Honda\xa0City\xa0ZX', 'Honda\xa0Jazz\xa0S', 'Honda\xa0Jazz\xa0VX',
            'Honda\xa0Mobilio', 'Honda\xa0Mobilio\xa0S', 'Honda\xa0WR\xa0V',
            'Hyundai\xa0Accent', 'Hyundai\xa0Accent\xa0Executive',
            'Hyundai\xa0Accent\xa0GLE', 'Hyundai\xa0Accent\xa0GLX',
            'Hyundai\xa0Creta', 'Hyundai\xa0Creta\xa01.6',
            'Hyundai\xa0Elantra\xa01.8', 'Hyundai\xa0Elantra\xa0SX',
            'Hyundai\xa0Elite\xa0i20', 'Hyundai\xa0Eon', 'Hyundai\xa0Eon\xa0D',
            'Hyundai\xa0Eon\xa0Era', 'Hyundai\xa0Eon\xa0Magna',
            'Hyundai\xa0Eon\xa0Sportz', 'Hyundai\xa0Fluidic\xa0Verna',
            'Hyundai\xa0Getz', 'Hyundai\xa0Getz\xa0GLE',
            'Hyundai\xa0Getz\xa0Prime', 'Hyundai\xa0Grand\xa0i10',
            'Hyundai\xa0Santro', 'Hyundai\xa0Santro\xa0AE',
            'Hyundai\xa0Santro\xa0Xing', 'Hyundai\xa0Sonata\xa0Transform',
            'Hyundai\xa0Verna', 'Hyundai\xa0Verna\xa01.4',
            'Hyundai\xa0Verna\xa01.6', 'Hyundai\xa0Verna\xa0Fluidic',
            'Hyundai\xa0Verna\xa0Transform', 'Hyundai\xa0Verna\xa0VGT',
            'Hyundai\xa0Xcent\xa0Base', 'Hyundai\xa0Xcent\xa0SX',
            'Hyundai\xa0i10', 'Hyundai\xa0i10\xa0Era',
            'Hyundai\xa0i10\xa0Magna', 'Hyundai\xa0i10\xa0Sportz',
            'Hyundai\xa0i20', 'Hyundai\xa0i20\xa0Active',
            'Hyundai\xa0i20\xa0Asta', 'Hyundai\xa0i20\xa0Magna',
            'Hyundai\xa0i20\xa0Select', 'Hyundai\xa0i20\xa0Sportz',
            'Jaguar\xa0XE\xa0XE', 'Jaguar\xa0XF\xa02.2',
            'Jeep\xa0Wrangler\xa0Unlimited', 'Land\xa0Rover\xa0Freelander',
            'Mahindra\xa0Bolero\xa0DI', 'Mahindra\xa0Bolero\xa0Power',
            'Mahindra\xa0Bolero\xa0SLE', 'Mahindra\xa0Jeep\xa0CL550',
            'Mahindra\xa0Jeep\xa0MM', 'Mahindra\xa0KUV100',
            'Mahindra\xa0KUV100\xa0K8', 'Mahindra\xa0Logan',
            'Mahindra\xa0Logan\xa0Diesel', 'Mahindra\xa0Quanto\xa0C4',
            'Mahindra\xa0Quanto\xa0C8', 'Mahindra\xa0Scorpio',
```

```
!pip install ibm_watson_machine_learning
```

```
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/si
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/p
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/s
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/si
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/sit
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/s
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/li
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Pyth
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/l
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/s
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-3.9/lib/python3.9/
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3.9
```

```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey": "5166agKSvfJPWAARH8s_Lneamj5Ixktt36YQwC5rLD0w"
}
client = APIClient(wml_credentials)
```

```
def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    #print(space)
    return(next(item for item in space['resources'] if item['entity']["name"] == space_nam
```

```
space_uid = guid_from_space_name(client, 'models')
print(space_uid)
```

```
4add9297-a8a7-452a-a7b0-edcf8ad2fecd
```

```
client.set.default_space(space_uid)
```

```
'SUCCESS'
```

```
client.software_specifications.list()
```

```
-----------------------------  ------------------------------------  ----
NAME                           ASSET_ID                              TYPE
default_py3.6                  0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12      020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt     069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6        09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12     09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9      0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6          0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                     0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod   1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6              10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl      111e41b3-de2d-5422-a4d6-bf776828c4b7  base
autoai-kb_rt22.2-py3.10        125b6d9a-5b1f-5e8d-972a-b251688ccf40  base
runtime-22.1-py3.9             12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6        154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                   1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6         1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
kernel-spark3.3-r3.6           1c9e5454-f216-59dd-a20e-474a5cdf5988  base
```

```
pytorch-onnx_rt22.1-py3.9-edt  1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6           1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2                20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod   217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda        26215f05-08c3-5a41-a1b0-da66306ce658  base
do_py3.8                       295addb5-9ef9-547e-9bf4-92ae3563e720  base
autoai-ts_3.8-py3.8            2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6          2b73a275-7cbf-420b-a912-eae7f436e0bc  base
kernel-spark3.3-py3.9          2b7961e2-e3b1-5a8c-a491-482c8368839a  base
pytorch_1.2-py3.6              2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3                2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt     32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37           36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4                390d21f8-e58b-4fac-9c55-d7ceda621326  base
autoai-ts_rt22.2-py3.10        396b2e83-0953-5b86-9a55-7ce1628a406f  base
xgboost_0.82-py3.6             39e31acd-5f30-41dc-ae44-60233c80306e  base
pytorch-onnx_1.2-py3.6-edt     40589d0e-7019-4e28-8daa-fb03b6f4fe12  base
pytorch-onnx_rt22.2-py3.10     40e73f55-783a-5535-b3fa-0c8b94291431  base
default_r36py38                41c247d3-45f8-5a71-b065-8580229facf0  base
autoai-ts_rt22.1-py3.9         4269d26e-07ba-5d40-8f66-2d495b0c71f7  base
autoai-obm_3.0                 42b92e18-d9ab-567f-988a-4240ba1ed5f7  base
pmml-3.0_4.3                   493bcb95-16f1-5bc5-bee8-81b8af80e9c7  base
spark-mllib_2.4-r_3.6          49403dff-92e9-4c87-a3d7-a42d0021c095  base
xgboost_0.90-py3.6             4ff8d6c2-1343-4c18-85e1-689c965304d3  base
pytorch-onnx_1.1-py3.6         50f95b2a-bc16-43bb-bc94-b0bed208c60b  base
autoai-ts_3.9-py3.8            52c57136-80fa-572e-8728-a5e7cbb42cde  base
spark-mllib_2.4-scala_2.11     55a70f99-7320-4be5-9fb9-9edb5a443af5  base
spark-mllib_3.0                5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9  base
autoai-obm_2.0                 5c2e37fa-80b8-5e77-840f-d912469614ee  base
spss-modeler_18.1              5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b  base
cuda-py3.8                     5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e  base
runtime-22.2-py3.10-xc         5e8cddff-db4a-5a6a-b8aa-2d4af9864dab  base
autoai-kb_3.1-py3.7            632d4b22-10aa-5180-88f0-f52dfb6444d7  base
-----------------------------  ------------------------------------  ----
Note: Only first 50 records were displayed. To display more use 'limit' parameter.
```

```
software_spec_uid = client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
software_spec_uid
```

```
'12b83a17-24d8-5082-900f-0ab31fbfd3cb'
```

```
MODEL_NAME="Car Resale Value Prediction"
DEPLOYMENT_NAME="models"
DEMO_MODEL=pipe
```

```
model_props={
    client.repository.ModelMetaNames.NAME:MODEL_NAME,
    client.repository.ModelMetaNames.TYPE:'scikit-learn_1.0',
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid

}
```

```
model_details=client.repository.store_model(
model=DEMO_MODEL,
meta_props=model_props,
training_data=X_train,
training_target=y_train
)
```

```
model_id=client.repository.get_model_uid(model_details)
model_id
```

```
    This method is deprecated, please use get_model_id()
    'c3f766ad-de63-41fa-ad92-c01b9bb4754e'
```

```
deployment_props={
    client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    client.deployments.ConfigurationMetaNames.ONLINE:{}
}
```

```
deployment=client.deployments.create(
artifact_uid=model_id,
meta_props=deployment_props
)
```

```
    ##########################################################################################

    Synchronous deployment creation for uid: 'c3f766ad-de63-41fa-ad92-c01b9bb4754e' start

    ##########################################################################################


    initializing
    Note: online_url is deprecated and will be removed in a future release. Use serving_u

    ready


    ------------------------------------------------------------------------------------------
    Successfully finished deployment creation, deployment_uid='7a2f4e5c-0f6f-435c-b1e1-2a
    ------------------------------------------------------------------------------------------
```