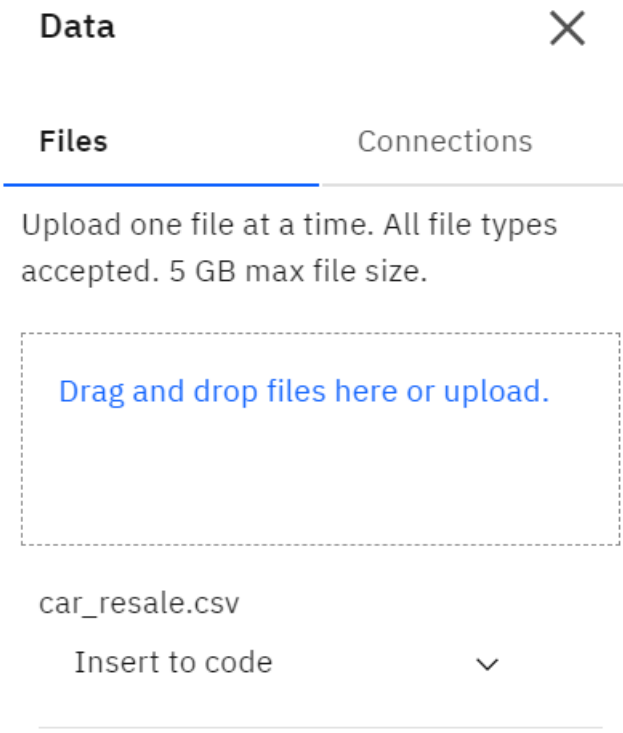
Development Phase
Train the model on IBM cloud

Date	18 November 2022
Team ID	PNT2022TMID21553
Project Name	Project – Car Resale Value Prediction



In [1]:
```
import pandas as pd
import numpy as np
import matplotlib as plt
from sklearn.preprocessing import LabelEncoder
import pickle
```

READ THE DATASETS

In [2]:

```
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 includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='0cEMgCpub2nZF3LK07mkqLs1luADFC07vVBCeF5JGpVe',
    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-blg0jnxocswfh1'
object_key = 'car_resale.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__, body )

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

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/IPython/core/interactiveshell.py:3457: DtypeWarning: Columns (11) have mixed types.
Specify dtype option on import or set low_memory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

Out[2]:

	dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	mod
0	24-03-2016 11:52	Golf_3_1.6	privat	Angebot	480.0	test	NaN	1993.0	manuell	0.0	g
1	24-03-2016 10:58	A5_Sportback_2.7_Tdi	privat	Angebot	18300.0	test	coupe	2011.0	manuell	190.0	Na
2	14-03-2016 12:52	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800.0	test	suv	2004.0	automatik	163.0	grar
3	17-03-2016 16:54	GOLF_4_1_4__3TÜRER	privat	Angebot	1500.0	test	kleinwagen	2001.0	manuell	75.0	g
4	31-03-2016 17:25	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600.0	test	kleinwagen	2008.0	manuell	69.0	fab

In [3]:

```
df.tail()
```

Out[3]:

	dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gear
371534	14-03-2016	Suche_t4___vito_ab_6_sitze	privat	Angebot	2200.0	test	NaN	2005.0	

CLEANING THE DATASET

In [4]:
```
#different sellers
print(df.seller.value_counts())
```

```
privat      371534
gewerblich       3
golf             1
Name: seller, dtype: int64
```

In [5]:
```
#remove the seller 'gewerblich'
df[df.seller != 'gewerblich']
```

Out[5]:

	dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gear
0	24-03-2016 11:52	Golf_3_1.6	privat	Angebot	480.0	test	NaN	1993.0	mar
1	24-03-2016 10:58	A5_Sportback_2.7_Tdi	privat	Angebot	18300.0	test	coupe	2011.0	mar
2	14-03-2016 12:52	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800.0	test	suv	2004.0	auton
3	17-03-2016 16:54	GOLF_4_1_4__3TÜRER	privat	Angebot	1500.0	test	kleinwagen	2001.0	mar
4	31-03-2016	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600.0	test	kleinwagen	2008.0	mar

In [6]:
```
#all entries of column 'seller' are same
#drop the column 'seller'
df = df.drop('seller', 1)
```

```
/tmp/wsuser/ipykernel_408/776328260.py:3: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argu
ment 'labels' will be keyword-only
  df = df.drop('seller', 1)
```

In [7]:
```
#different offer types
print(df.offerType.value_counts())
```

```
Angebot   371525
Gesuch        12
150000         1
Name: offerType, dtype: int64
```

In [8]:
```
#remove the offertype 'Gesuch'
df[df.offerType != 'Gesuch']
```

Out[8]:

	dateCrawled	name	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	p
0	24-03-2016 11:52	Golf_3_1.6	Angebot	480.0	test	NaN	1993.0	manuell	
1	24-03-2016 10:58	A5_Sportback_2.7_Tdi	Angebot	18300.0	test	coupe	2011.0	manuell	
2	14-03-2016 12:52	Jeep_Grand_Cherokee_"Overland"	Angebot	9800.0	test	suv	2004.0	automatik	

In [9]:
```
#column 'offerType' has same entires
#drop the column 'offerType'
df = df.drop('offerType', 1)
```

```
/tmp/wsuser/ipykernel_408/939542170.py:3: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except fo
ment 'labels' will be keyword-only
  df = df.drop('offerType', 1)
```

In [10]:
```
print(df.shape)
```

```
(371539, 18)
```

In [11]:
```
#remove cars having power less than 50p and greater than 900p
df = df[(df.powerPS > 50) & (df.powerPS < 900)]
print(df.shape)
```

```
(319717, 18)
```

In [12]:
```
#remove cars with year of registration before 1950 and after 2017
df = df[(df.yearOfRegistration >= 1950) & (df.yearOfRegistration < 2017)]
print(df.shape)
```

```
(309179, 18)
```

In [13]:
```
#remove columns that are not relevant
df.drop(['name', 'abtest', 'dateCrawled', 'nrOfPictures', 'lastSeen', 'postalCode', 'dateCreated'], axis='columns', inplace=True)
```

In [14]:
```
#creating a copy of the dataframe and remove the duplicates in the columns
new_df = df.copy()
```

In [15]:
```
#clean the dataset of German words and replace with proper English words
new_df.gearbox.replace(('manuell', 'automatik'), ('manual', 'automatic'), inplace=True)
new_df.fuelType.replace(('benzin', 'andere', 'elektro'), ('petrol', 'others', 'electric'), inplace=True)
new_df.vehicleType.replace(('kleinwagen', 'cabrio', 'kombi', 'andere'), ('small  car', 'convertible', 'combination', 'others'), inplace=True)
new_df.notRepairedDamage.replace(('ja', 'nein'), ('Yes', 'No'), inplace=True)
```

In [16]:
```
#Outlier Removal
new_df = new_df[(new_df.price >= 100) & (new_df.price <= 150000)]
```

In [17]:
```
#Fill the not declared values of the columns as NaN using fillna function
new_df['notRepairedDamage'].fillna(value='not-declared', inplace=True)
new_df['fuelType'].fillna(value='not-declared', inplace=True)
new_df['gearbox'].fillna(value='not-declared', inplace=True)
new_df['vehicleType'].fillna(value='not-declared', inplace=True)
new_df['model'].fillna(value='not-declared', inplace=True)
```

In [18]:
```
#save the dataframe as csv
new_df.to_csv('car_resale_preprocessed.csv')
```

In [19]:
```
#label encode the categorical data
labels = ['gearbox', 'notRepairedDamage', 'model', 'brand', 'fuelType', 'vehicleType']

mapping = {}
for i in labels:
  mapping[i] = LabelEncoder()
  mapping[i].fit(new_df[i])
  tr = mapping[i].transform(new_df[i])
  np.save(str('classes'+i+'.npy'), mapping[i].classes_)
```

In [20]:
```
#'labeled' dataframe contains the final data

labelled = new_df[ ['price', 'yearOfRegistration', 'powerPS', 'kilometer', 'monthOfRegistration'] + [x+"_labels" for x in labels]]
print(labelled.columns)
```

```
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
       'model_labels', 'brand_labels', 'fuelType_labels',
       'vehicleType_labels'],
      dtype='object')
```

SPLITTING DATA INTO INDEPENDENT AND DEPENDENT VARIABLES

In [21]:
```
#split price and other data into Y and X respectively
Y = labelled.iloc[:, 0].values
X = labelled.iloc[:, 1:].values
Y = Y.reshape(-1, 1)
```

In [22]:
```
#split dataset into train and test dataset
from sklearn.model_selection import cross_val_score, train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=3)
```

MODEL BUILDING

CHOOSE THE APPROPRIATE MODEL AND CHECK THE METRICS OF THE MODELS

In [23]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
```

In [24]:
```
regressor = RandomForestRegressor(n_estimators=1000, max_depth=10, random_state=34)

regressor.fit(X_train, np.ravel(Y_train, order='C'))
```

Out[24]:
```
RandomForestRegressor(max_depth=10, n_estimators=1000, random_state=34)
```

In [25]:
```
pred_1 = regressor.predict(X_test)

print(r2_score(Y_test, pred_1))
```

```
0.8396847388211943
```

In [26]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [27]:
```
ds = DecisionTreeClassifier(max_depth=5000, max_features=0.9, max_leaf_nodes=5000, random_state=2, splitter='best')

ds.fit(X_train,  np.ravel(Y_train, order='C'))
```

Out[27]:
```
DecisionTreeClassifier(max_depth=5000, max_features=0.9, max_leaf_nodes=5000,
                       random_state=2)
```

In [28]:
```
pred_3 =ds.predict(X_test)

print(r2_score(Y_test, pred_3))
```

```
0.6753189087840161
```

In [29]:
```
file_name = 'resale_model.pkl'
pickle.dump(regressor, open(file_name, 'wb'))
```

DEPLOY MODEL IN IBM CLOUD

In [30]:
```
!pip install ibm_watson_machine_learning
```

```
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learn
ing) (2.11.0)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_lea
rning) (1.3.4)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learni
ng) (4.8.2)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.26.7)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.8.9)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.26.0)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (21.3)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2022.9.2
4)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==
2.11.*->ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->i
bm_watson_machine_learning) (0.10.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*-
>ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==
2.11.0->ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.8.2)
```

In [42]:
```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
   "url": "https://us-south.ml.cloud.ibm.com",
   "apikey": "GxGc70sFN0c3WjkhCyutIq8zsCOhhQ0MrznbSeQ8aTw0"
}
client = APIClient(wml_credentials)
```

In [43]:
```
#create deployment space
def guid_from_space_name(client, space_name):
   space = client.spaces.get_details()
   return(next(item for item in space['resources'] if item['entity']["name"] == space_name)['metadata']['id'])
```

In [44]:
```
#create deployment space as 'new space'
space_uid = guid_from_space_name(client, 'models')
print(space_uid)
```

```
8f2d2037-436b-4453-aa67-acae3cf33557
```

In [45]:
```
#make the created space as default space
client.set.default_space(space_uid)
```

Out[45]: 'SUCCESS'

In [35]:
```
#view client software specifications
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              2e8ef57d-2687-4b7d-acce-01f94976dac1 base
```

In [56]:
```
software_spec_uid = client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
software_spec_uid
```

Out[56]: '12b83a17-24d8-5082-900f-0ab31fbfd3cb'

In [57]:
```
import sklearn
sklearn.__version__
```

Out[57]: '1.0.2'

In [58]:
```
#store the model in the deployment space
MODEL_NAME = 'CAR_RESALE_PREDICTION'
DEPLOYMENT_NAME = 'models'
DEMO_MODEL = regressor
```

In [59]:
```
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
}
```

In [54]:
```
import json
```

In [60]:
```
model_details = client.repository.store_model(
  model = DEMO_MODEL,
  meta_props = model_props,
  training_data = X_train,
  training_target = Y_train
)
```

In [61]:
```
model_details
```

Out[61]:

```
{'entity': {'hybrid_pipeline_software_specs': [],
  'label_column': 'l0',
  'schemas': {'input': [{'fields': [{'name': 'f0', 'type': 'float'},
     {'name': 'f1', 'type': 'float'},
     {'name': 'f2', 'type': 'str'},
     {'name': 'f3', 'type': 'float'},
     {'name': 'f4', 'type': 'int'},
     {'name': 'f5', 'type': 'int'},
     {'name': 'f6', 'type': 'int'},
     {'name': 'f7', 'type': 'int'},
     {'name': 'f8', 'type': 'int'},
     {'name': 'f9', 'type': 'int'}],
    'id': '1',
    'type': 'struct'}],
   'output': []},
  'software_spec': {'id': '12b83a17-24d8-5082-900f-0ab31fbfd3cb',
   'name': 'runtime-22.1-py3.9'},
  'type': 'scikit-learn_1.0'},
 'metadata': {'created_at': '2022-11-18T15:42:06.299Z',
  'id': '4934f7cd-2b1a-4154-b6a4-71d5b742b357',
  'modified_at': '2022-11-18T15:42:49.056Z',
```


In [62]:
```
model_id = client.repository.get_model_id(model_details)
model_id
```

Out[62]:

```
'4934f7cd-2b1a-4154-b6a4-71d5b742b357'
```


In [65]:
```
deployment_props = {
  client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
  client.deployments.ConfigurationMetaNames.ONLINE: {}
}
```

In [66]:
```
deployment = client.deployments.create(
  artifact_uid=model_id,
  meta_props=deployment_props
)
```

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

Synchronous deployment creation for uid: '4934f7cd-2b1a-4154-b6a4-71d5b742b357' started

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


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


------------------------------------------------------------------------------------------------
Successfully finished deployment creation, deployment_uid='8397b34a-34dc-40fe-9160-49934a2e3720'
------------------------------------------------------------------------------------------------
```

The model is deployed on the IBM cloud.

The dataset is stored in the cloud storage.