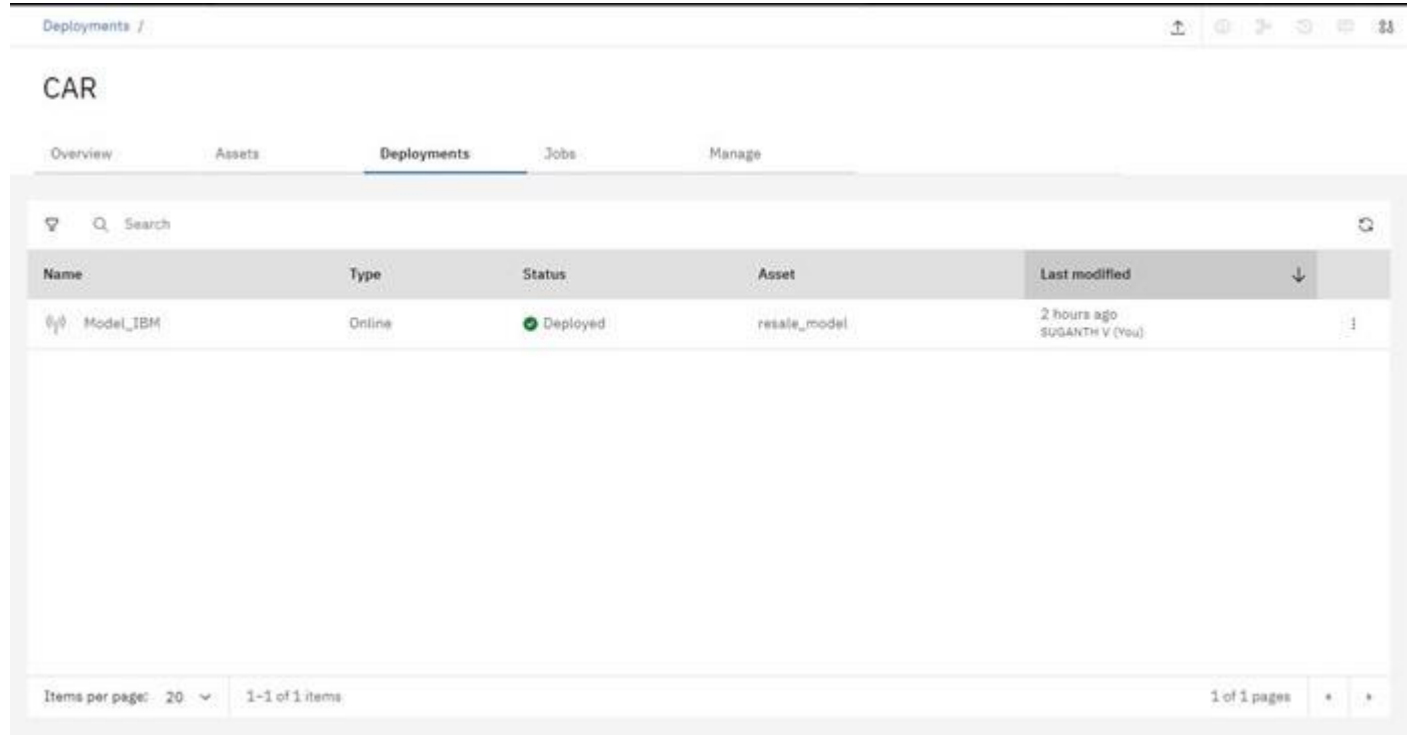


TRAIN THE ML MODEL ON IBM

Team ID	PNT2022TMID15858
Project Name	Car Resale value Prediction

TRAIN THE ML MODEL ON IBM



Deployments /				
CAR				
Overview	Assets	Deployments	Jobs	Manage
🔍 Search				
Name	Type	Status	Asset	Last modified
Model_IBM	Online	Deployed	resale_model	2 hours ago SUGANTH V (You)
Items per page: 20 1-1 of 1 items 1 of 1 pages				

```
import pandas as pd
import numpy as np
import matplotlib as plt
from sklearn.preprocessing import LabelEncoder
import pickle
print("IMPORTED REQUIRED LIBRARIES")
# df = pd.read_csv("C:/Users/MUGUNTHAN/Desktop/IBM/Data/autos.csv", header=0 , sep=',',
,encoding='Latin1',low_memory=False)
# df.head()
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3
import io
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='DT151-IL0017uhnUGwXyhG_Eort5gohoW6XJTNoT3RKk',
    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-yuhtzmzidi0ka1p' object_key
= 'autos.csv'
```

```
body = cos_client.get_object(Bucket=bucket,Key=object_key)
df = pd.read_csv((io.BytesIO(body['Body'].read())) , header=0 , sep=',',encoding='Latin1',low_memory=False)
df.head()
# df = pd.read_csv("C:/Users/MUGUNTHAN/Desktop/IBM/Data/autos.csv", header=0 , sep=',',
,encoding='Latin1',low_memory=False)
# df.head() import
os, types import
pandas as pd
from botocore.client import Config
import ibm_boto3
import io
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='DT15l-IL0017uhnUGwXyhG_Eort5gohoW6XJTNoT3RkK',
    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-yuhtzmzidi0ka1p' object_key
= 'autos.csv'
```

```
body = cos_client.get_object(Bucket=bucket,Key=object_key)
df = pd.read_csv((io.BytesIO(body['Body'].read())) , header=0 , sep=',',encoding='Latin1',low_memory=False)
df.head() print(df.seller.value_counts()) df[df.seller !='gewerblich']
df=df.drop('seller',axis=1)

print(df.offerType.value_counts())
df[df.offerType !='Gesuch']
df=df.drop('offerType',axis=1) print(df.shape)
df=df[(df.powerPS>50) & (df.powerPS<900)] print(df.shape)
df=df[(df.yearOfRegistration>=1950)&(df.yearOfRegistration<2022)] print(df.shape)
df.drop(['name','abtest','dateCrawled','nrOfPictures','lastSeen','postalCode','dateCreated'], axis='columns',inplace=True)
new_df=df.copy()
new_df=new_df.drop_duplicates(['price','vehicleType','yearOfRegistration','gearbox','powerPS','model','kilometer','monthOfRegistration','fuelType','notRepairedDamage'])
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)
new_df=new_df[(new_df.price>=100)&(new_df.price<=150000)]

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) from
ibm_watson_machine_learning import APIClient wml_credentials={
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"hEAn_mcoP3u_-ZjagjeqlxDayqUiETpYVYWdR1OLKAby"
} client =APIClient(wml_credentials) def
guide_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_name)['metadata']['id'])
space_uid=guide_from_space_name(client,'CAR') print("Space UID"+ space_uid)
client.set.default_space(space_uid) client.software_specifications.list()
software_spec_uid = client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
software_spec_uid print(new_df)
labels=['gearbox','notRepairedDamage','model','brand','fuelType','vehicleType']

mapper={} for
i in labels:
    mapper[i]=LabelEncoder()
mapper[i].fit(new_df[i])
tr=mapper[i].transform(new_df[i])
    np.save(str('classes'+i+'.numpy'),mapper[i].classes_)
print(i,":",mapper[i])
    new_df.loc[:, i+ '_labels']=pd.Series(tr,index=new_df.index)

labeled = new_df[['price','yearOfRegistration','powerPS','kilometer','monthOfRegistration']+[x+" _labels" for x in
labels]] print(labeled.columns) Y=labeled.iloc[:,0].values
X=labeled.iloc[:,1:].values

Y=Y.reshape(-1,1)
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) from
sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
regressor = RandomForestRegressor(n_estimators = 1000,max_depth = 10,random_state = 34)

regressor.fit(X_train, np.ravel(Y_train,order='C')) y_pred
= regressor.predict(X_test)
print(r2_score(Y_test,y_pred))
filename='resale_model.sav'
pickle.dump(regressor,open(filename,'wb'))
model_details = client.repository.store_model(model=regressor,meta_props={
client.repository.ModelMetaNames.NAME: "resale_model",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid,
client.repository.ModelMetaNames.TYPE: "scikit-learn_1.0"
})
model_id = client.repository.get_model_id(model_details)
model_id X_train[0]
regressor.predict([[2012.0, 179.0, '1500000', 12.0, 0, 0, 30, 1, 1, 4]])

```

