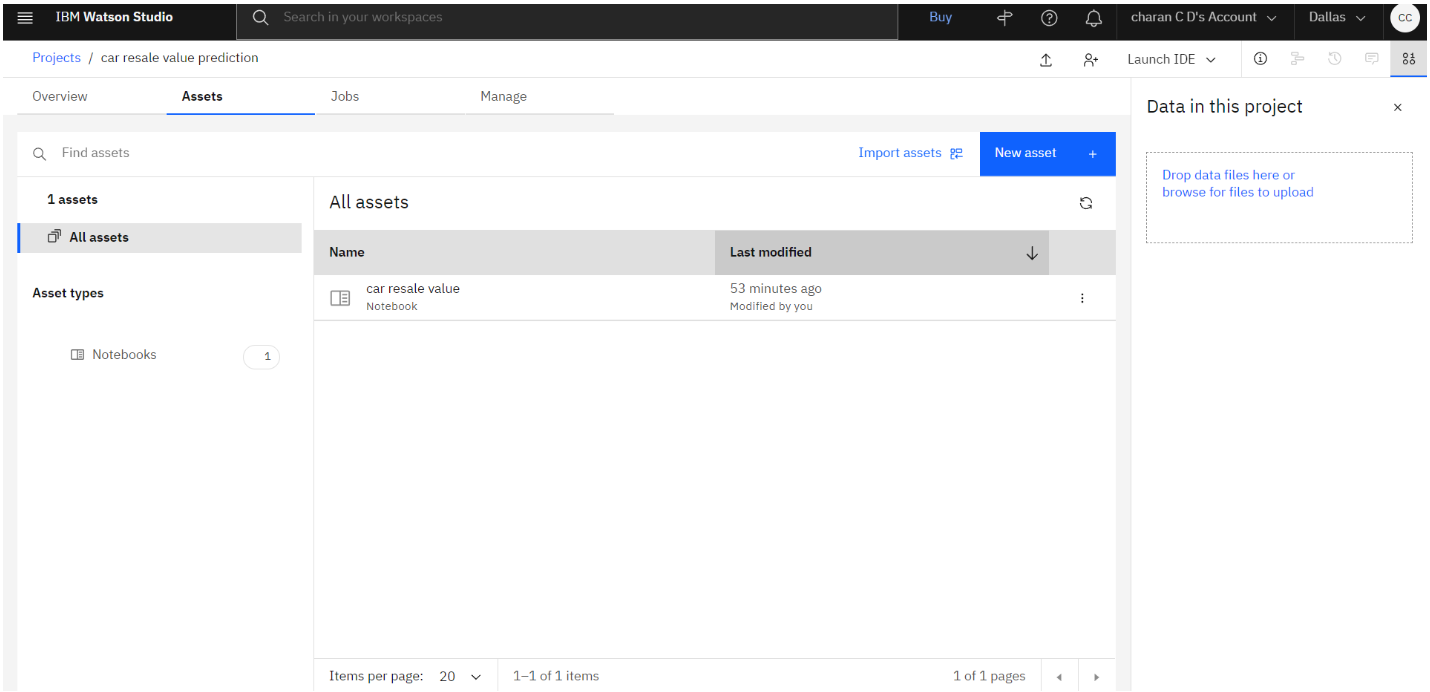
TRAIN THE ML MODEL ON IBM

Team ID	**PNT2022TMID15174**
Project Name	**Car Resale value Prediction**

TRAIN THE ML MODEL ON IBM

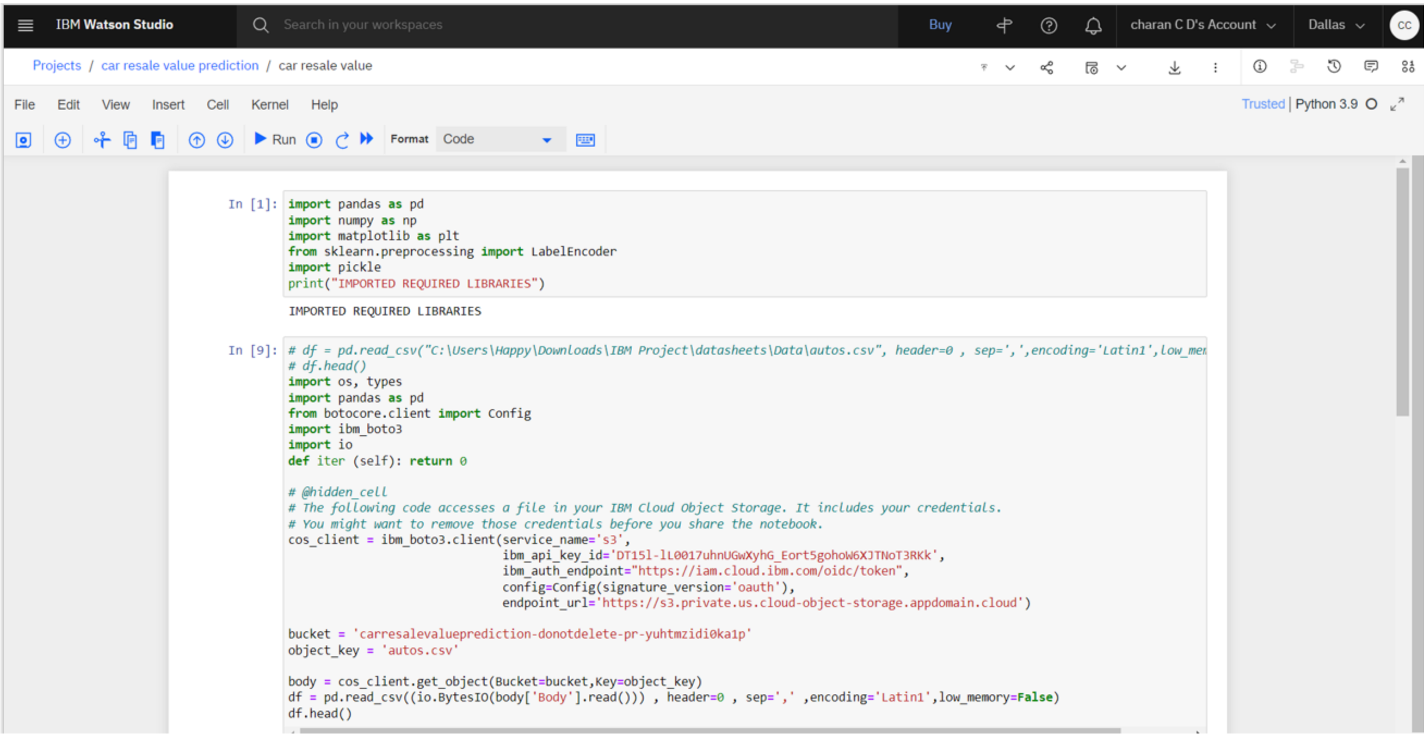


All assets

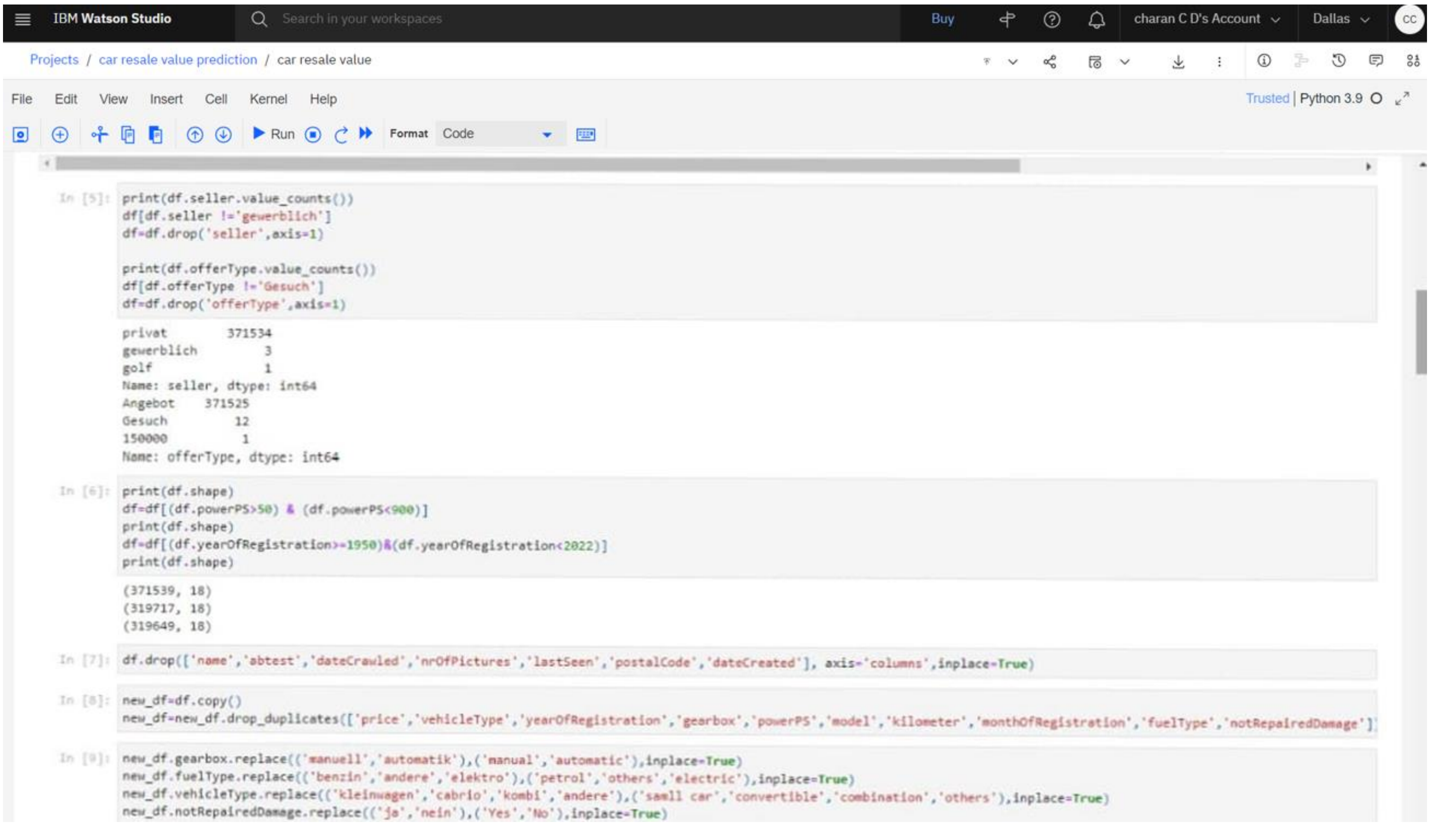
Name	Last modified
car resale value Notebook	53 minutes ago Modified by you

```
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/SUGARANJAN/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-lL0017uhnUGwXyhG_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-yuhtmzidi0ka1p'
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/SUGARANJAN/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-lL0017uhnUGwXyhG_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-yuhtmzidi0ka1p'
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'),('samll 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+'.npy'),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'))
```

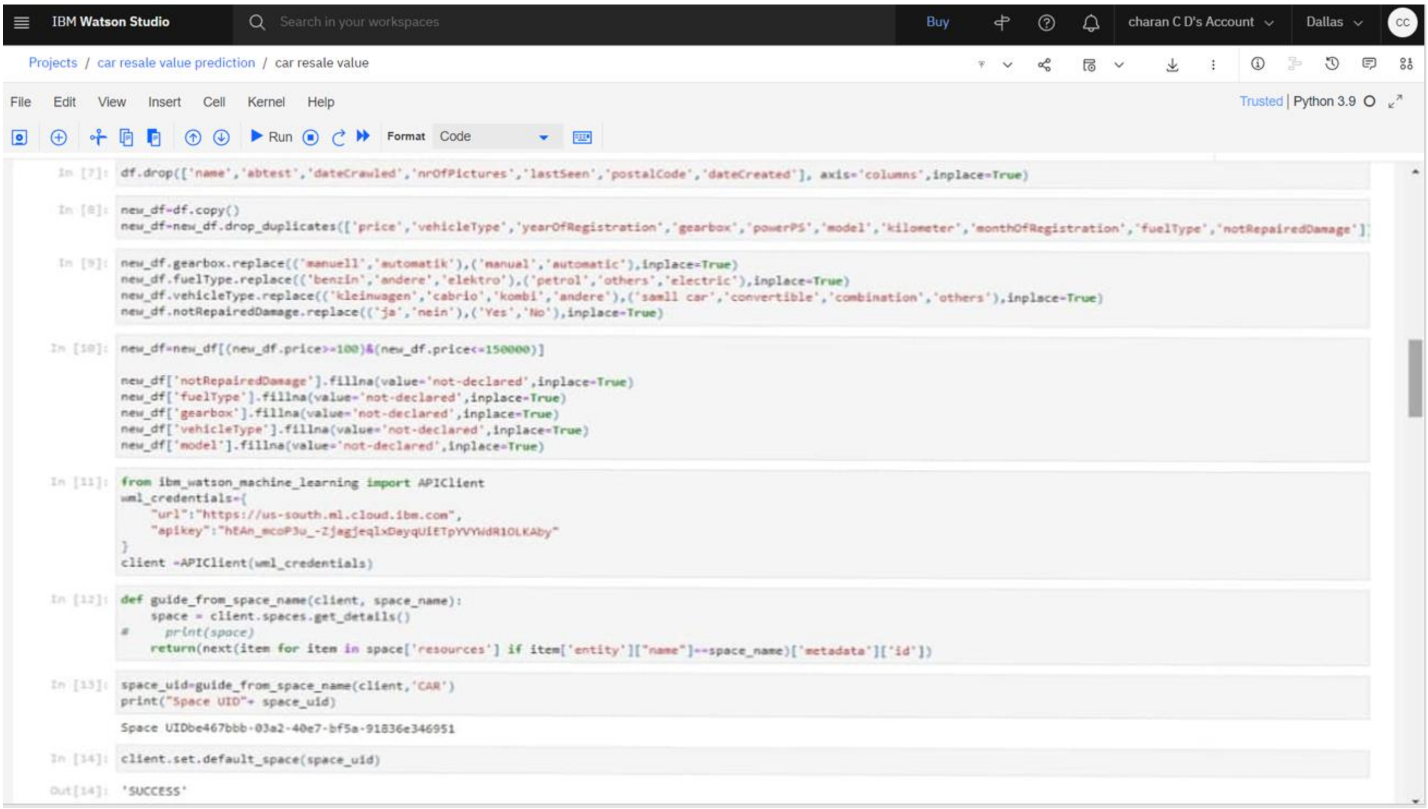
```
model_details = client.repository.store_model(model=regressor,meta_props={
    client.repository.ModelMetaNames.NAME: "resale_model",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid,
    client.repository.ModelMetaNames.TYPE: "scikit-learn_1.0"
})
model_id = client.repository.get_model_id(model_details)
model_id
X_train[0]
regressor.predict([[2012.0, 179.0, '1500000', 12.0, 0, 0, 30, 1, 1, 4]])
```


IBM Watson Studio
Search in your workspaces
Buy
charan C D's Account
Dallas
CC
Projects / car resale value prediction / car resale value
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format Code

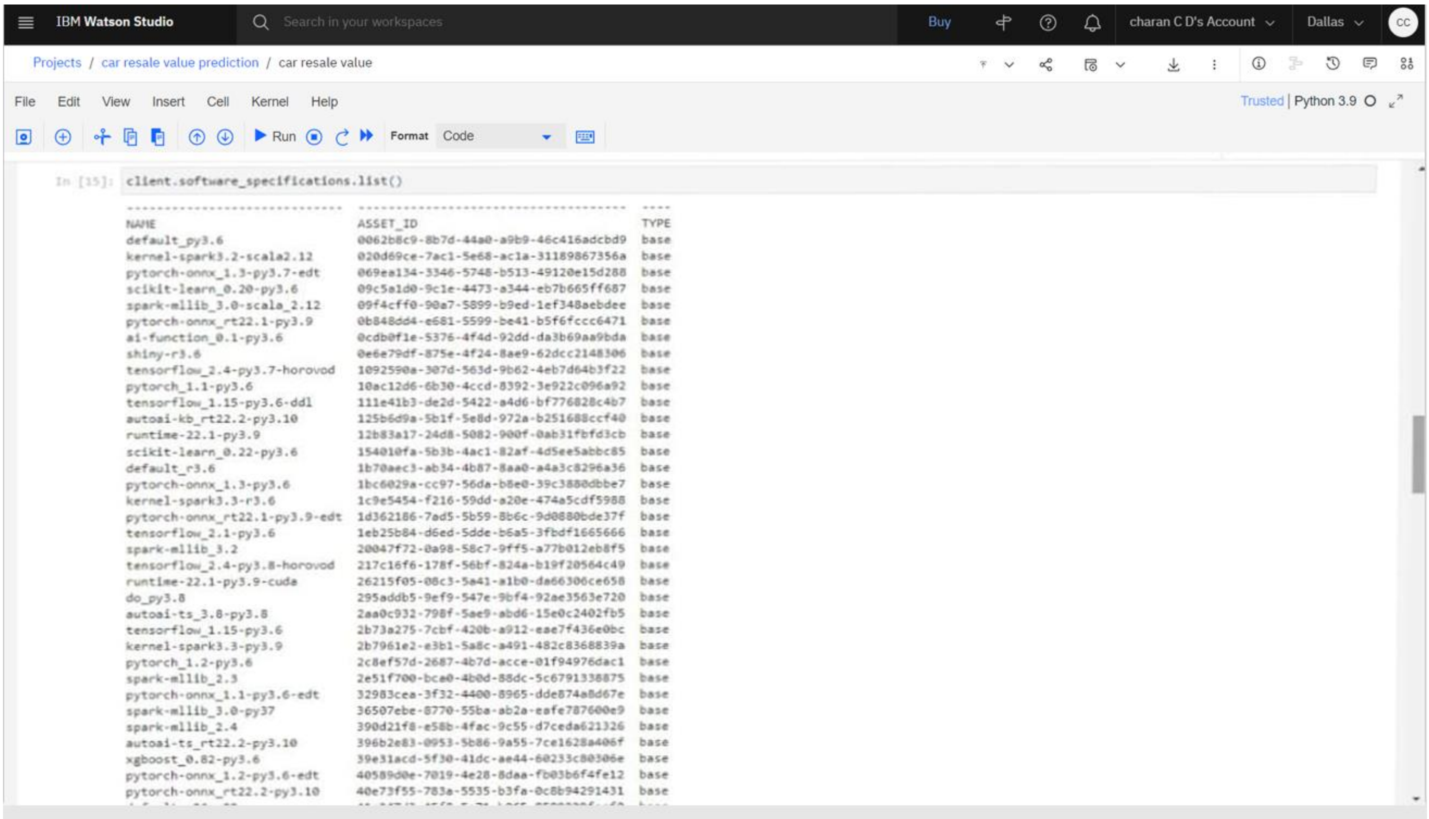
```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib as plt
        from sklearn.preprocessing import LabelEncoder
        import pickle
        print("IMPORTED REQUIRED LIBRARIES")

        IMPORTED REQUIRED LIBRARIES

In [9]: # df = pd.read_csv("C:\Users\Happy\Downloads\IBM Project\datasheets\Data\autos.csv", header=0 , sep=',',encoding='Latin1',low_mem
        # 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-1L0017uhnUGwXyhG_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-yuhtmzidi0ka1p'
        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()
```


IBM Watson Studio
Search in your workspaces
Buy
charan C D's Account
Dallas
CC
Projects / car resale value prediction / car resale value
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format Code

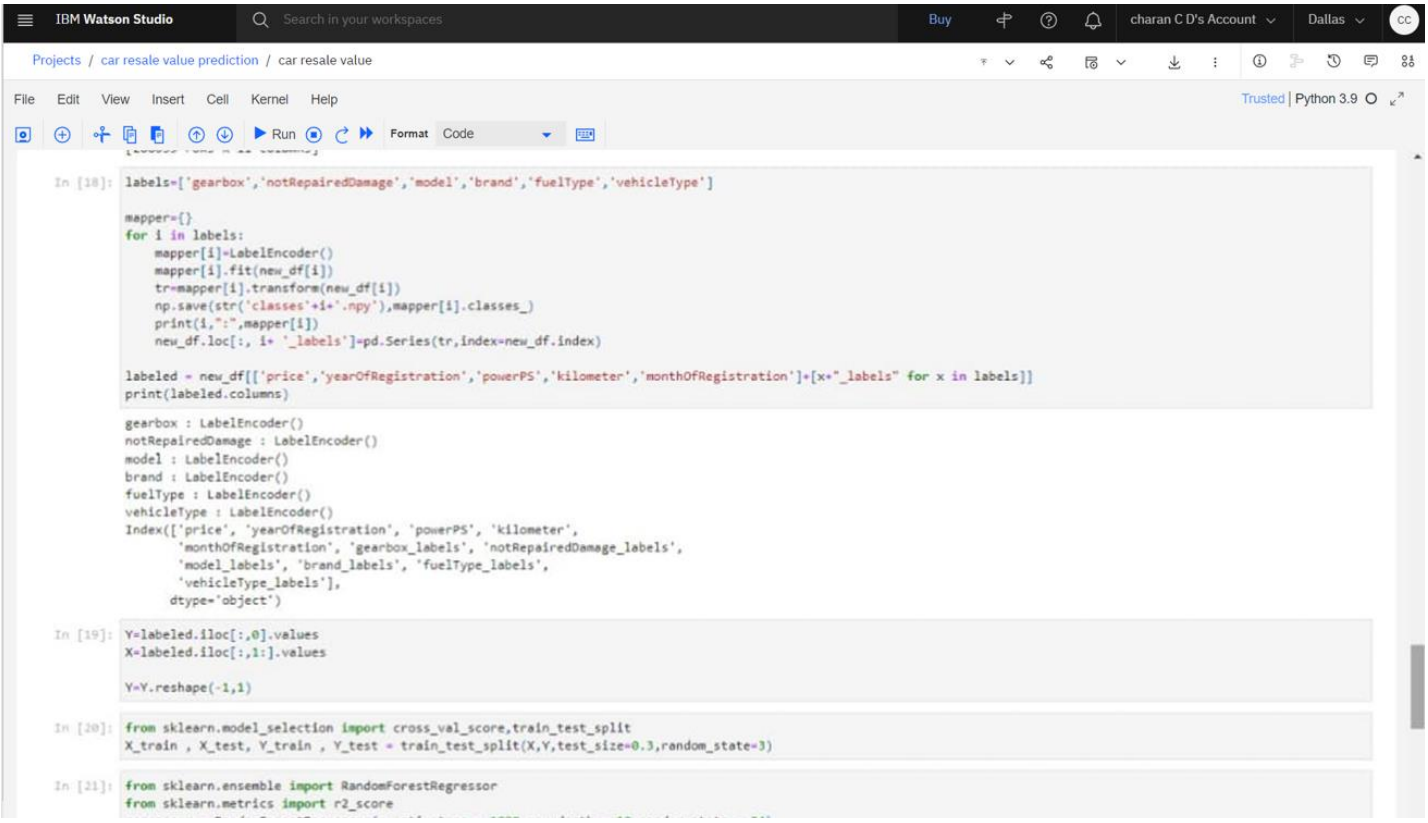
```
In [5]: print(df.seller.value_counts())
        df[df.seller !='gewerblich']
        df=df.drop('seller',axis=1)

        print(df.offerType.value_counts())
        df[df.offerType !='Gesuch']
        df=df.drop('offerType',axis=1)

        privat        371534
        gewerblich         3
        golf               1
        Name: seller, dtype: int64
        Angebot    371525
        Gesuch         12
        150000          1
        Name: offerType, dtype: int64

In [6]: 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)

        (371539, 18)
        (319717, 18)
        (319649, 18)

In [7]: df.drop(['name','abtest','dateCrawled','nrOfPictures','lastSeen','postalCode','dateCreated'], axis='columns',inplace=True)

In [8]: new_df=df.copy()
        new_df=new_df.drop_duplicates(['price','vehicleType','yearOfRegistration','gearbox','powerPS','model','kilometer','monthOfRegistration','fuelType','notRepairedDamage'])

In [9]: 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'),('samll car','convertible','combination','others'),inplace=True)
        new_df.notRepairedDamage.replace(('ja','nein'),('Yes','No'),inplace=True)
```


IBM Watson Studio
Search in your workspaces
Buy
charan C D's Account
Dallas
CC
Projects / car resale value prediction / car resale value
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format
Code

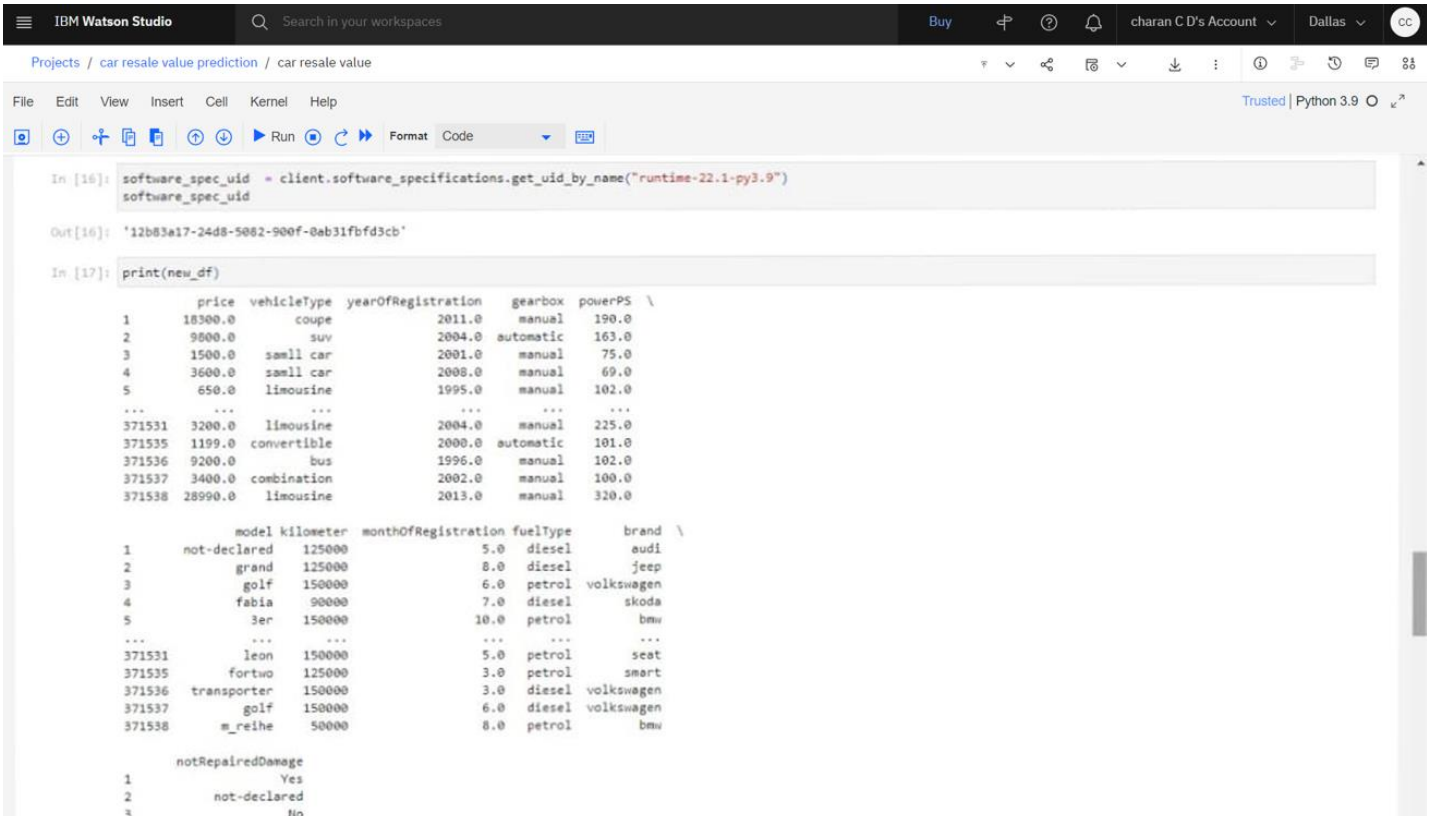
```
In [7]: df.drop(['name','abtest','dateCrawled','nrOfPictures','lastSeen','postalCode','dateCreated'], axis='columns',inplace=True)

In [8]: new_df=df.copy()
        new_df=new_df.drop_duplicates(['price','vehicleType','yearOfRegistration','gearbox','powerPS','model','kilometer','monthOfRegistration','fuelType','notRepairedDamage'])

In [9]: 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'),('samll car','convertible','combination','others'),inplace=True)
        new_df.notRepairedDamage.replace(('ja','nein'),('Yes','No'),inplace=True)

In [10]: 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)

In [11]: 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)

In [12]: 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'])

In [13]: space_uid=guide_from_space_name(client,'CAR')
         print("Space UID"+ space_uid)

         Space UIDbe467bbb-03a2-40e7-bf5a-91836e346951

In [14]: client.set.default_space(space_uid)

Out[14]: 'SUCCESS'
```


IBM Watson Studio
Search in your workspaces
Buy
charan C D's Account
Dallas
CC
Projects / car resale value prediction / car resale value
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format
Code

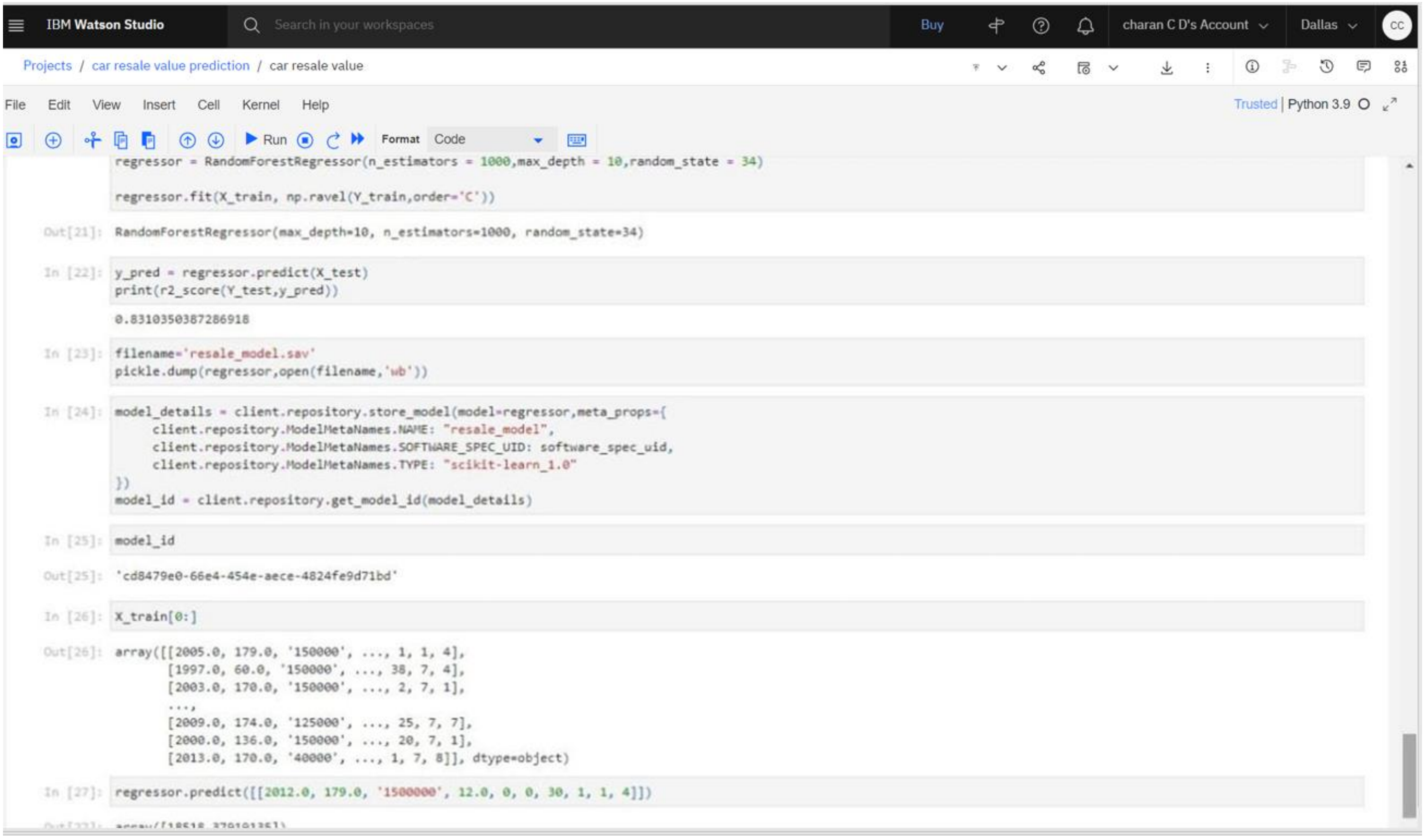
```
In [15]: client.software_specifications.list()

         -----------------------------  ------------------------------------  ----
         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
```


IBM Watson Studio
Search in your workspaces
Buy
charan C D's Account
Dallas
CC
Projects / car resale value prediction / car resale value
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format Code

```
In [18]: 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+'.npy'),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)

         gearbox : LabelEncoder()
         notRepairedDamage : LabelEncoder()
         model : LabelEncoder()
         brand : LabelEncoder()
         fuelType : LabelEncoder()
         vehicleType : LabelEncoder()
         Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
                'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
                'model_labels', 'brand_labels', 'fuelType_labels',
                'vehicleType_labels'],
               dtype='object')

In [19]: Y=labeled.iloc[:,0].values
         X=labeled.iloc[:,1:].values

         Y=Y.reshape(-1,1)

In [20]: 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)

In [21]: from sklearn.ensemble import RandomForestRegressor
         from sklearn.metrics import r2_score
```


IBM Watson Studio
Search in your workspaces
Buy
charan C D's Account
Dallas
CC
Projects / car resale value prediction / car resale value
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format Code

```
In [16]: software_spec_uid  = client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
         software_spec_uid

Out[16]: '12b83a17-24d8-5082-900f-0ab31fbfd3cb'

In [17]: print(new_df)

                  price  vehicleType  yearOfRegistration    gearbox  powerPS  \
         1      18300.0        coupe              2011.0     manual    190.0
         2       9800.0          suv              2004.0  automatic    163.0
         3       1500.0    samll car              2001.0     manual     75.0
         4       3600.0    samll car              2008.0     manual     69.0
         5        650.0    limousine              1995.0     manual    102.0
         ...        ...          ...                 ...        ...      ...
         371531  3200.0    limousine              2004.0     manual    225.0
         371535  1199.0  convertible              2000.0  automatic    101.0
         371536  9200.0          bus              1996.0     manual    102.0
         371537  3400.0  combination              2002.0     manual    100.0
         371538 28990.0    limousine              2013.0     manual    320.0

                       model kilometer  monthOfRegistration fuelType       brand  \
         1      not-declared    125000                  5.0   diesel        audi
         2             grand    125000                  8.0   diesel        jeep
         3              golf    150000                  6.0   petrol  volkswagen
         4             fabia     90000                  7.0   diesel       skoda
         5               3er    150000                 10.0   petrol         bmw
         ...             ...       ...                  ...      ...         ...
         371531         leon    150000                  5.0   petrol        seat
         371535       fortwo    125000                  3.0   petrol       smart
         371536  transporter    150000                  3.0   diesel  volkswagen
         371537         golf    150000                  6.0   diesel  volkswagen
         371538      m_reihe     50000                  8.0   petrol         bmw

                notRepairedDamage
         1                    Yes
         2           not-declared
         3                     No
```



IBM Watson Studio
Search in your workspaces
Buy
charan C D's Account
Dallas
CC
Projects / car resale value prediction / car resale value
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format
Code

```
        regressor = RandomForestRegressor(n_estimators = 1000,max_depth = 10,random_state = 34)

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

Out[21]: RandomForestRegressor(max_depth=10, n_estimators=1000, random_state=34)

In [22]: y_pred = regressor.predict(X_test)
         print(r2_score(Y_test,y_pred))

         0.8310350387286918

In [23]: filename='resale_model.sav'
         pickle.dump(regressor,open(filename,'wb'))

In [24]: 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)

In [25]: model_id

Out[25]: 'cd8479e0-66e4-454e-aece-4824fe9d71bd'

In [26]: X_train[0:]

Out[26]: array([[2005.0, 179.0, '150000', ..., 1, 1, 4],
                [1997.0, 60.0, '150000', ..., 38, 7, 4],
                [2003.0, 170.0, '150000', ..., 2, 7, 1],
                ...,
                [2009.0, 174.0, '125000', ..., 25, 7, 7],
                [2000.0, 136.0, '150000', ..., 20, 7, 1],
                [2013.0, 170.0, '40000', ..., 1, 7, 8]], dtype=object)

In [27]: regressor.predict([[2012.0, 179.0, '1500000', 12.0, 0, 0, 30, 1, 1, 4]])

Out[27]: array([18518.37010135])
```