```
# Importing Libraries:
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


# for displaying all feature from dataset:
pd.pandas.set_option('display.max_columns', None)



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

def __iter__(self): return 0


# 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='L98UVbsm7b3_7zIa2z6yS-mEkgRaYiCuS_WER6MMiN-E',
    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 = 'ckdprediction-donotdelete-pr-uosytmsjavw7pn'
object_key = 'kidney_disease.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 )

dataset = pd.read_csv(body)
```

```
# Top 5 records:
dataset.head()
```

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wc	rc	htn
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	121.0	36.0	1.2	NaN	NaN	15.4	44	7800	5.2	yes
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	NaN	18.0	0.8	NaN	NaN	11.3	38	6000	NaN	no
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	423.0	53.0	1.8	NaN	NaN	9.6	31	7500	NaN	no
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	117.0	56.0	3.8	111.0	2.5	11.2	32	6700	3.9	yes
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	106.0	26.0	1.4	NaN	NaN	11.6	35	7300	4.6	no

```
# Dropping unneccsary feature :
dataset = dataset.drop('id', axis=1)
```

```
# Shape of dataset:
dataset.shape
```

```
(400, 25)
```

```
# Cheaking Missing (NaN) Values:
dataset.isnull().sum()
```

```
age                9
bp                12
sg                47
al                46
su                49
```

```
rbc               152
pc                 65
pcc                 4
ba                  4
bgr                44
bu                 19
sc                 17
sod                87
pot                88
hemo               52
pcv                70
wc                105
rc                130
htn                 2
dm                  2
cad                 2
appet               1
pe                  1
ane                 1
classification      0
dtype: int64
```

```
# Description:
dataset.describe()
```

	age	bp	sg	al	su	bgr	bu	sc	sod	pot

```
# Datatypes:
dataset.dtypes
```

```
age               float64
bp                float64
sg                float64
al                float64
su                float64
rbc                object
pc                 object
pcc                object
ba                 object
bgr               float64
bu                float64
sc                float64
sod               float64
pot               float64
hemo              float64
pcv                object
wc                 object
rc                 object
htn                object
dm                 object
cad                object
appet              object
pe                 object
ane                object
classification     object
dtype: object
```

```
dataset.head()
```

age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wc	rc	htn	dm	ca
48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	121.0	36.0	1.2	NaN	NaN	15.4	44	7800	5.2	yes	yes	n
7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	NaN	18.0	0.8	NaN	NaN	11.3	38	6000	NaN	no	no	n
62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	423.0	53.0	1.8	NaN	NaN	9.6	31	7500	NaN	no	yes	n

```
dataset['rbc'].value_counts()
```

```
normal      201
abnormal     47
Name: rbc, dtype: int64
```

```
dataset['rbc'] = dataset['rbc'].replace(to_replace = {'normal' : 0, 'abnormal' : 1})
```

```
dataset['pc'].value_counts()
```

```
normal      259
abnormal     76
Name: pc, dtype: int64
```

```
dataset['pc'] = dataset['pc'].replace(to_replace = {'normal' : 0, 'abnormal' : 1})
```

```
dataset['pcc'].value_counts()
```

```
notpresent    354
present        42
Name: pcc, dtype: int64
```

```
dataset['pcc'] = dataset['pcc'].replace(to_replace = {'notpresent':0,'present':1})
```

```
dataset['ba'].value_counts()
```

```
notpresent    374
present        22
Name: ba, dtype: int64
```

```
dataset['ba'] = dataset['ba'].replace(to_replace = {'notpresent':0,'present':1})
```

```
dataset['htn'].value_counts()
```

```
no     251
yes    147
Name: htn, dtype: int64
```

```
dataset['htn'] = dataset['htn'].replace(to_replace = {'yes' : 1, 'no' : 0})
```

```
dataset['dm'].value_counts()
```

```
no       258
yes      134
\tno       3
\tyes      2
 yes       1
Name: dm, dtype: int64
```

```
dataset['dm'] = dataset['dm'].replace(to_replace = {'\tyes':'yes', ' yes':'yes', '\tno':'no'})
```

```
dataset['dm'] = dataset['dm'].replace(to_replace = {'yes' : 1, 'no' : 0})
```

```
dataset['cad'].value_counts()
```

```
no      362
yes      34
\tno      2
Name: cad, dtype: int64
```

```
dataset['cad'] = dataset['cad'].replace(to_replace = {'\tno':'no'})
```

```
dataset['cad'] = dataset['cad'].replace(to_replace = {'yes' : 1, 'no' : 0})
```

```
dataset['appet'].unique()
```

```
array(['good', 'poor', nan], dtype=object)
```

```
dataset['appet'] = dataset['appet'].replace(to_replace={'good':1,'poor':0,'no':np.nan})
```

```
dataset['pe'].value_counts()
```

```
no     323
yes     76
Name: pe, dtype: int64
```

```
dataset['pe'] = dataset['pe'].replace(to_replace = {'yes' : 1, 'no' : 0})
```

```
dataset['ane'].value_counts()
```

```
no     339
yes     60
Name: ane, dtype: int64
```

```
dataset['ane'] = dataset['ane'].replace(to_replace = {'yes' : 1, 'no' : 0})
```

```
dataset['classification'].value_counts()
```

```
ckd       248
notckd    150
ckd\t       2
Name: classification, dtype: int64
```

```
dataset['classification'] = dataset['classification'].replace(to_replace={'ckd\t':'ckd'})


dataset["classification"] = [1 if i == "ckd" else 0 for i in dataset["classification"]]


dataset.head()
```

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wc	rc	htn	dm	cad	appet	pe	ar
0	48.0	80.0	1.020	1.0	0.0	NaN	0.0	0.0	0.0	121.0	36.0	1.2	NaN	NaN	15.4	44	7800	5.2	1.0	1.0	0.0	1.0	0.0	0
1	7.0	50.0	1.020	4.0	0.0	NaN	0.0	0.0	0.0	NaN	18.0	0.8	NaN	NaN	11.3	38	6000	NaN	0.0	0.0	0.0	1.0	0.0	0
2	62.0	80.0	1.010	2.0	3.0	0.0	0.0	0.0	0.0	423.0	53.0	1.8	NaN	NaN	9.6	31	7500	NaN	0.0	1.0	0.0	0.0	0.0	1
3	48.0	70.0	1.005	4.0	0.0	0.0	1.0	1.0	0.0	117.0	56.0	3.8	111.0	2.5	11.2	32	6700	3.9	1.0	0.0	0.0	0.0	1.0	1
4	51.0	80.0	1.010	2.0	0.0	0.0	0.0	0.0	0.0	106.0	26.0	1.4	NaN	NaN	11.6	35	7300	4.6	0.0	0.0	0.0	1.0	0.0	0

```
# Datatypes:
dataset.dtypes
```

```
age              float64
bp               float64
sg               float64
al               float64
su               float64
rbc              float64
pc               float64
pcc              float64
ba               float64
bgr              float64
bu               float64
sc               float64
sod              float64
pot              float64
hemo             float64
```

```
pcv                object
wc                 object
rc                 object
htn               float64
dm                float64
cad               float64
appet             float64
pe                float64
ane               float64
classification      int64
dtype: object
```

```
dataset['pcv'] = pd.to_numeric(dataset['pcv'], errors='coerce')
dataset['wc'] = pd.to_numeric(dataset['wc'], errors='coerce')
dataset['rc'] = pd.to_numeric(dataset['rc'], errors='coerce')
```

```
# Datatypes:
dataset.dtypes
```

```
age               float64
bp                float64
sg                float64
al                float64
su                float64
rbc               float64
pc                float64
pcc               float64
ba                float64
bgr               float64
bu                float64
sc                float64
sod               float64
pot               float64
hemo              float64
pcv               float64
wc                float64
rc                float64
htn               float64
dm                float64
```

```
cad               float64
appet             float64
pe                float64
ane               float64
classification      int64
dtype: object
```

```
# Description:
dataset.describe()
```

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	
count	391.000000	388.000000	353.000000	354.000000	351.000000	248.000000	335.000000	396.000000	396.000000	356.000000	381.0(
mean	51.483376	76.469072	1.017408	1.016949	0.450142	0.189516	0.226866	0.106061	0.055556	148.036517	57.4:
std	17.169714	13.683637	0.005717	1.352679	1.099191	0.392711	0.419431	0.308305	0.229351	79.281714	50.5(
min	2.000000	50.000000	1.005000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	22.000000	1.5(
25%	42.000000	70.000000	1.010000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	99.000000	27.0(
50%	55.000000	80.000000	1.020000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	121.000000	42.0(
75%	64.500000	80.000000	1.020000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	163.000000	66.0(
max	90.000000	180.000000	1.025000	5.000000	5.000000	1.000000	1.000000	1.000000	1.000000	490.000000	391.0(

```
# Cheaking Missing (NaN) Values:
dataset.isnull().sum().sort_values(ascending=False)
```

```
rbc               152
rc                131
wc                106
pot                88
sod                87
pcv                71
pc                 65
```

```
hemo              52
su                49
sg                47
al                46
bgr               44
bu                19
sc                17
bp                12
age                9
ba                 4
pcc                4
htn                2
dm                 2
cad                2
appet              1
pe                 1
ane                1
classification     0
dtype: int64
```

```
dataset.columns
```

```
Index(['age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr', 'bu',
       'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc', 'htn', 'dm', 'cad',
       'appet', 'pe', 'ane', 'classification'],
      dtype='object')
```

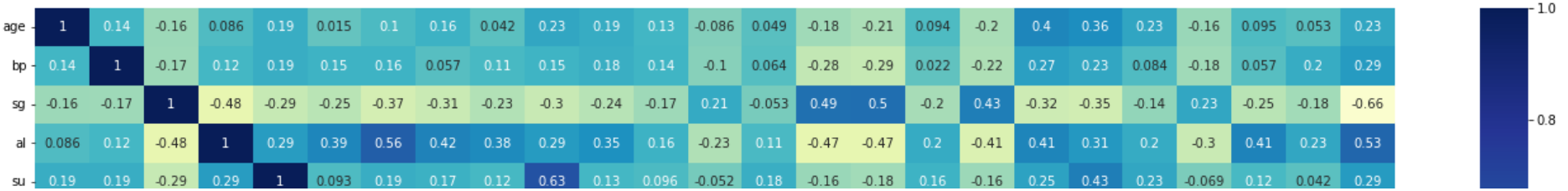
```
features = ['age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr', 'bu',
            'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc', 'htn', 'dm', 'cad',
            'appet', 'pe', 'ane']
```

```
for feature in features:
    dataset[feature] = dataset[feature].fillna(dataset[feature].median())
```

```
dataset.isnull().any().sum()
```

```
0
```

```
plt.figure(figsize=(24,14))
sns.heatmap(dataset.corr(), annot=True, cmap='YlGnBu')
plt.show()
```




```
dataset.drop('pcv', axis=1, inplace=True)
```


```
dataset.head()
```

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	wc	rc	htn	dm	cad	appet	pe	ane	cla
0	48.0	80.0	1.020	1.0	0.0	0.0	0.0	0.0	0.0	121.0	36.0	1.2	138.0	4.4	15.4	7800.0	5.2	1.0	1.0	0.0	1.0	0.0	0.0	
1	7.0	50.0	1.020	4.0	0.0	0.0	0.0	0.0	0.0	121.0	18.0	0.8	138.0	4.4	11.3	6000.0	4.8	0.0	0.0	0.0	1.0	0.0	0.0	
2	62.0	80.0	1.010	2.0	3.0	0.0	0.0	0.0	0.0	423.0	53.0	1.8	138.0	4.4	9.6	7500.0	4.8	0.0	1.0	0.0	0.0	0.0	1.0	
3	48.0	70.0	1.005	4.0	0.0	0.0	1.0	1.0	0.0	117.0	56.0	3.8	111.0	2.5	11.2	6700.0	3.9	1.0	0.0	0.0	0.0	1.0	1.0	
4	51.0	80.0	1.010	2.0	0.0	0.0	0.0	0.0	0.0	106.0	26.0	1.4	138.0	4.4	11.6	7300.0	4.6	0.0	0.0	0.0	1.0	0.0	0.0	

```
# Target feature:
sns.countplot(dataset['classification'])
```

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as
  warnings.warn(
<AxesSubplot:xlabel='classification', ylabel='count'>
```

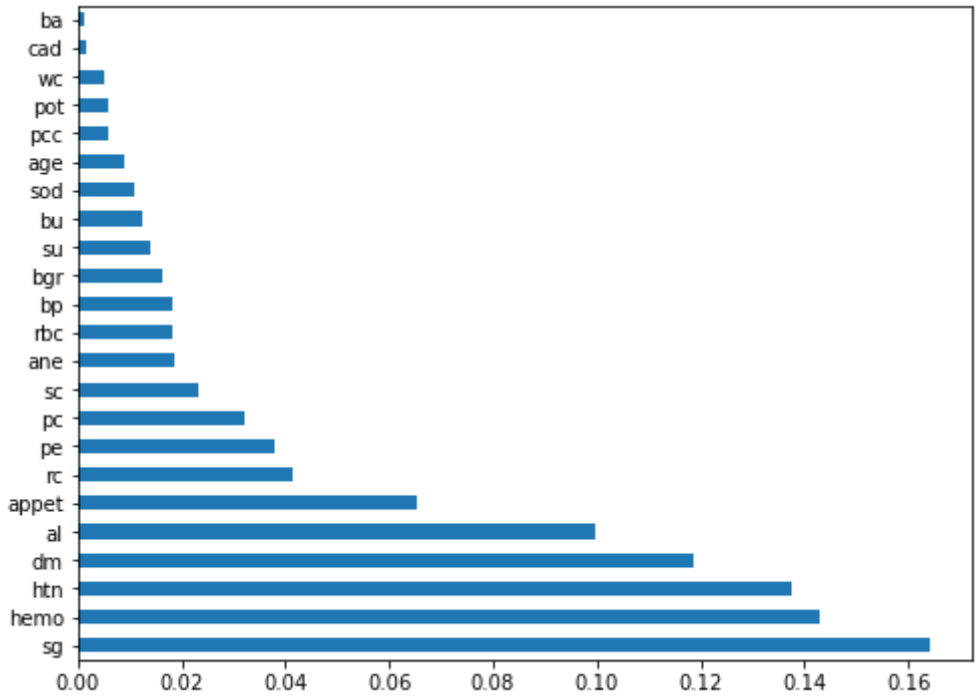

```
# Independent and Dependent Feature:
X = dataset.iloc[:, :-1]
y = dataset.iloc[:, -1]
```

```
X.head()
```

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	wc	rc	htn	dm	cad	appet	pe	ane
0	48.0	80.0	1.020	1.0	0.0	0.0	0.0	0.0	0.0	121.0	36.0	1.2	138.0	4.4	15.4	7800.0	5.2	1.0	1.0	0.0	1.0	0.0	0.0
1	7.0	50.0	1.020	4.0	0.0	0.0	0.0	0.0	0.0	121.0	18.0	0.8	138.0	4.4	11.3	6000.0	4.8	0.0	0.0	0.0	1.0	0.0	0.0
2	62.0	80.0	1.010	2.0	3.0	0.0	0.0	0.0	0.0	423.0	53.0	1.8	138.0	4.4	9.6	7500.0	4.8	0.0	1.0	0.0	0.0	0.0	1.0
3	48.0	70.0	1.005	4.0	0.0	0.0	1.0	1.0	0.0	117.0	56.0	3.8	111.0	2.5	11.2	6700.0	3.9	1.0	0.0	0.0	0.0	1.0	1.0
4	51.0	80.0	1.010	2.0	0.0	0.0	0.0	0.0	0.0	106.0	26.0	1.4	138.0	4.4	11.6	7300.0	4.6	0.0	0.0	0.0	1.0	0.0	0.0

```
# Feature Importance:
from sklearn.ensemble import ExtraTreesClassifier
import matplotlib.pyplot as plt
model=ExtraTreesClassifier()
model.fit(X,y)

plt.figure(figsize=(8,6))
ranked_features=pd.Series(model.feature_importances_,index=X.columns)
ranked_features.nlargest(24).plot(kind='barh')
plt.show()
```




```
ranked_features.nlargest(8).index
```

```
Index(['sg', 'hemo', 'htn', 'dm', 'al', 'appet', 'rc', 'pe'], dtype='object')
```

```
X = dataset[['sg', 'htn', 'hemo', 'dm', 'al', 'appet', 'rc', 'pc']]
X.head()
```

```
X.tail()
```

	sg	htn	hemo	dm	al	appet	rc	pc
395	1.020	0.0	15.7	0.0	0.0	1.0	4.9	0.0
396	1.025	0.0	16.5	0.0	0.0	1.0	6.2	0.0
397	1.020	0.0	15.8	0.0	0.0	1.0	5.4	0.0
398	1.025	0.0	14.2	0.0	0.0	1.0	5.9	0.0
399	1.025	0.0	15.8	0.0	0.0	1.0	6.1	0.0

```
y.head()
```

```
0    1
1    1
2    1
3    1
4    1
Name: classification, dtype: int64
```

```
# Train Test Split:
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y, test_size=0.3, random_state=33)
```

```
print(X_train.shape)
print(X_test.shape)
```

```
(280, 8)
(120, 8)
```

```
# Importing Performance Metrics:
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

```
# RandomForestClassifier:
from sklearn.ensemble import RandomForestClassifier
RandomForest = RandomForestClassifier()
RandomForest = RandomForest.fit(X_train,y_train)

# Predictions:
y_pred = RandomForest.predict(X_test)

# Performance:
print('Accuracy:', accuracy_score(y_test,y_pred))
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
Accuracy: 0.975
[[55  3]
 [ 0 62]]
              precision    recall  f1-score   support

           0       1.00      0.95      0.97        58
           1       0.95      1.00      0.98        62

    accuracy                           0.97       120
   macro avg       0.98      0.97      0.97       120
weighted avg       0.98      0.97      0.97       120
```

```
# AdaBoostClassifier:
from sklearn.ensemble import AdaBoostClassifier
AdaBoost = AdaBoostClassifier()
AdaBoost = AdaBoost.fit(X_train,y_train)

# Predictions:
y_pred = AdaBoost.predict(X_test)

# Performance:
print('Accuracy:', accuracy_score(y_test,y_pred))
```

```
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
Accuracy: 0.975
[[55  3]
 [ 0 62]]
              precision    recall  f1-score   support

           0       1.00      0.95      0.97        58
           1       0.95      1.00      0.98        62

    accuracy                           0.97       120
   macro avg       0.98      0.97      0.97       120
weighted avg       0.98      0.97      0.97       120
```

```
# GradientBoostingClassifier:
from sklearn.ensemble import GradientBoostingClassifier
GradientBoost = GradientBoostingClassifier()
GradientBoost = GradientBoost.fit(X_train,y_train)

# Predictions:
y_pred = GradientBoost.predict(X_test)

# Performance:
print('Accuracy:', accuracy_score(y_test,y_pred))
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
Accuracy: 0.975
[[55  3]
 [ 0 62]]
              precision    recall  f1-score   support

           0       1.00      0.95      0.97        58
           1       0.95      1.00      0.98        62

    accuracy                           0.97       120
   macro avg       0.98      0.97      0.97       120
```

```
weighted avg       0.98      0.97      0.97       120
```

```
import pickle
pickle.dump(RandomForest, open("chronic_kidney_disease_prediction_model.pkl", 'wb'))
```

```
#install necessary libraries for ibm deployment
!pip install -U 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: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learn
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-lear
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-ma
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
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-lear
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learni
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-lea
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learn
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-m
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
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-s
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from i
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.2
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm-watso
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from reques
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from importlib-metadata->ib
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from packagi
```

```
from ibm_watson_machine_learning import APIClient
import json
import numpy as np
```

```
wml_credentials={
    "apikey":"1ljScmIG2pnHq56Rj-Gs46EJ1U_LXOyFAGtFyWd7Ca6V",
    "url":"https://us-south.ml.cloud.ibm.com"
}
```

```
wml_client=APIClient(wml_credentials)
```

```
wml_client.spaces.list()
```

```
Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
------------------------------------  ---------  ------------------------
ID                                    NAME       CREATED
c6205b3d-f891-4d0f-a099-beb38bc6dd82  CKD_space  2022-11-14T16:37:31.378Z
------------------------------------  ---------  ------------------------
```

```
SPACE_ID="c6205b3d-f891-4d0f-a099-beb38bc6dd82"
```

```
wml_client.set.default_space(SPACE_ID)
```

```
'SUCCESS'
```

```
wml_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
autoai-kb_3.1-py3.7           632d4b22-10aa-5180-88f0-f52dfb6444d7  base
pytorch-onnx_1.7-py3.8        634d3cdc-b562-5bf9-a2d4-ea90a478456b  base
------------------------------  ------------------------------------  ----
Note: Only first 50 records were displayed. To display more use 'limit' parameter.
```

```
#then we need to save and deploy our model
import sklearn
sklearn.__version__
```

```
'1.0.2'
```

```
MODEL_NAME='Demo_ckd'
DEPLOYMENT_NAME='CKD_predict'
DEMO_MODEL=RandomForest
```

```
#WE NEED TO SET DEFAULT PYTHON VERSION
software_spec_uid=wml_client.software_specifications.get_id_by_name('runtime-22.1-py3.9')
```

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

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

```
model_details
```

```
      {'entity': {'hybrid_pipeline_software_specs': [],
        'label_column': 'classification',
        'schemas': {'input': [{'fields': [{'name': 'sg', 'type': 'float64'},
            {'name': 'htn', 'type': 'float64'},
            {'name': 'hemo', 'type': 'float64'},
            {'name': 'dm', 'type': 'float64'},
            {'name': 'al', 'type': 'float64'},
            {'name': 'appet', 'type': 'float64'},
            {'name': 'rc', 'type': 'float64'},
            {'name': 'pc', 'type': 'float64'}],
           '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': '2022-11-14T17:05:42.393Z',
        'id': 'bd42b039-02cd-47f2-89e8-b40dc77c3585',
        'modified_at': '2022-11-14T17:05:47.177Z',
        'name': 'Demo_ckd',
        'owner': 'IBMid-66300432UH',
        'resource_key': '804f00d2-f193-49c7-a320-25b2ba2446f9',
        'space_id': 'c6205b3d-f891-4d0f-a099-beb38bc6dd82'},
       'system': {'warnings': []}}
```

```
model_id=wml_client.repository.get_model_id(model_details)
```

```
model_id
```

```
     'bd42b039-02cd-47f2-89e8-b40dc77c3585'
```

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

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

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

Synchronous deployment creation for uid: 'bd42b039-02cd-47f2-89e8-b40dc77c3585' 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='435db472-41e3-47a9-9b94-9bb7d6130a88'
------------------------------------------------------------------------------------------------
```