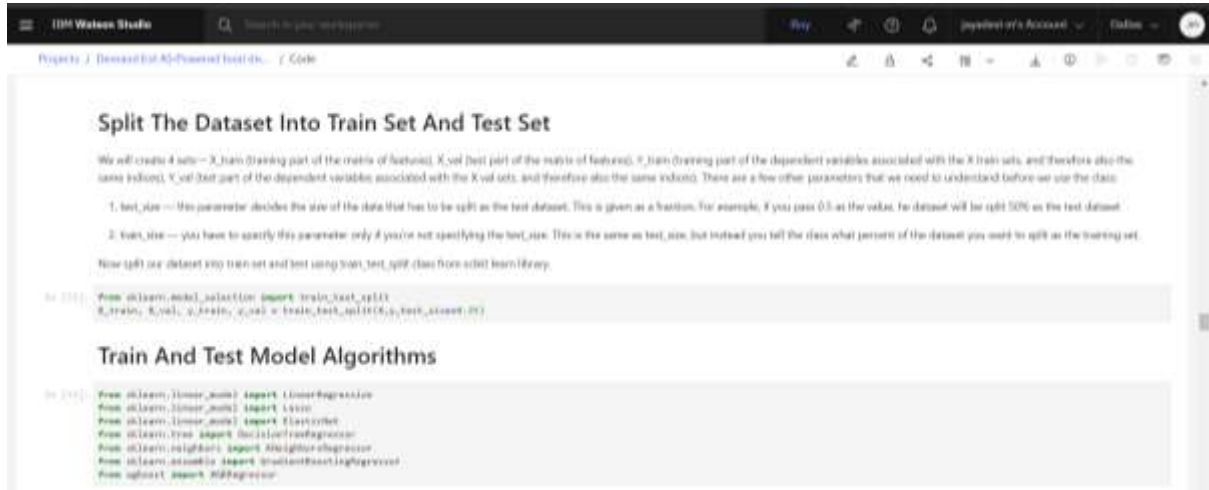


# MODEL BUILDING

## Train and test model algorithms



The screenshot shows a Jupyter Notebook in IBM Watson Studio. The notebook is titled "Split The Dataset Into Train Set And Test Set". It contains a text block explaining the process of splitting a dataset into training and testing sets. It mentions that the dataset is split into two parts:  $X_{train}$  (training part of the matrix of features),  $X_{val}$  (test part of the matrix of features),  $y_{train}$  (training part of the dependent variables associated with the  $X_{train}$  sets), and  $y_{val}$  (test part of the dependent variables associated with the  $X_{val}$  sets). It also mentions that there are a few other parameters that we need to understand before we use the data.

1.  $test\_size$  — This parameter decides the size of the data that has to be split as the test dataset. This is given as a fraction. For example, if you pass 0.3 as the value, the dataset will be split 70% as the training set.

2.  $train\_size$  — You have to specify this parameter only if you're not specifying the  $test\_size$ . This is the same as  $test\_size$ , but instead you tell the class what percent of the dataset you want to split as the training set.


Now split our dataset into train set and test using `train_test_split` class from `skit-learn` library.

```
In [17]: from sklearn.model_selection import train_test_split
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.3)
```

The notebook also has a section titled "Train And Test Model Algorithms" which contains a list of algorithms to be used for training and testing the model.

```
In [18]: from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Logistic
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import r2_score
```

## Model Evaluation



The screenshot shows a Jupyter Notebook in IBM Watson Studio. The notebook is titled "Model Evaluation". It contains a text block explaining the process of evaluating a model. It mentions that we are going to use  $X_{train}$  and  $y_{train}$  obtained above in `train_test_split` section to train our regression model. We're using the `fit` method and passing the parameters as shown below. Firstly, we need to check to see how well our model is performing on the test data.

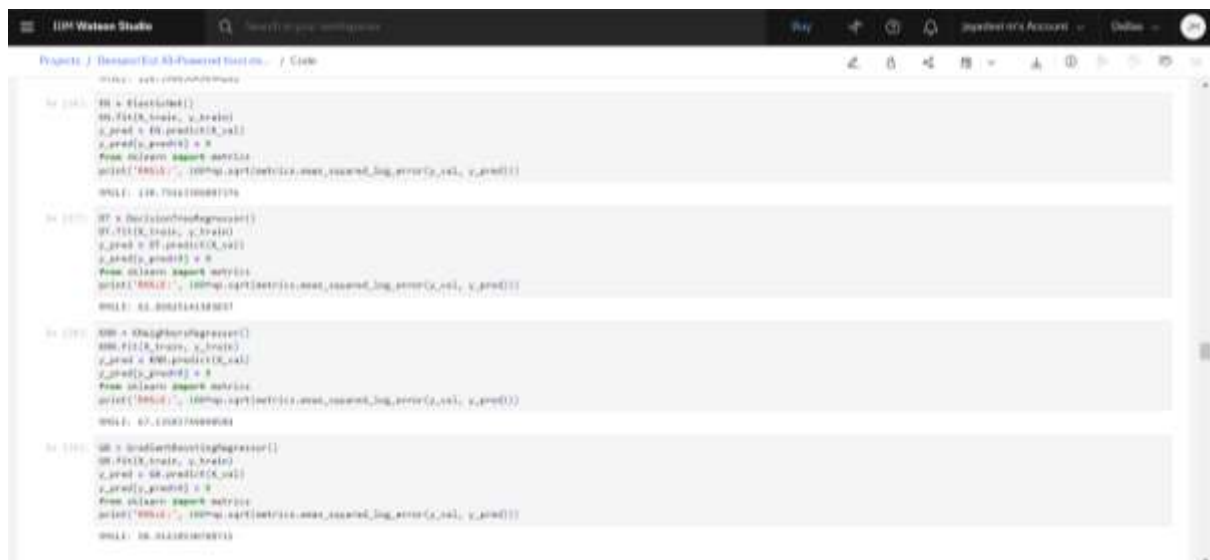
Regression Evaluation Metrics: RMSE (Root Mean Square Error) RMSE is the square root of the averaged squared difference between the target value and the value predicted by the model. It is preferred more in some cases because the errors are first squared before averaging which poses a high penalty on large errors. This implies that RMSE is useful when large errors are undesired.

For testing the model we use the below method.

```
In [19]: R0 = RidgeRegressor()
R0.fit(X_train, y_train)
y_pred = R0.predict(X_val)
y_pred[y_pred] = 0
from sklearn.metrics import
print('RMSE: ', 100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))
RMSE: 68.407190779988
```

```
In [20]: L = Logistic()
L.fit(X_train, y_train)
y_pred = L.predict(X_val)
y_pred[y_pred] = 0
from sklearn.metrics import
print('RMSE: ', 100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))
RMSE: 128.50804484410
```

```
In [21]: R1 = RidgeClassifier()
R1.fit(X_train, y_train)
y_pred = R1.predict(X_val)
y_pred[y_pred] = 0
from sklearn.metrics import
print('RMSE: ', 100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))
```



## Save the Model



## Predicting the output using the model





```
jupyter Code Last Checked: 14 hours ago (unsaved changes)
File Edit View Insert Cell Kernel Windows Help Not Trusted Python 3 (system)

In [ ]: software_spec_id = client.software_specifications.get_id_by_name("default_py3.8")
software_spec_id

Out[145]: 'ab0e180-f32c-503c-a702-4f23aaf77194'

In [ ]: model_details = client.repository.store_model(model = m0, meta_params={
    client.repository.MetadataName.META_TOOL_DEMAND_FORECASTING_DEPLOYMENT,
    client.repository.MetadataName.TYPES:'skits-learn_0.20',
    client.repository.MetadataName.SOFTWARE_SPEC_ID:software_spec_id
})

model_id = client.repository.get_model_id(model_details)
#this method is deprecated, please use get_model_id()

In [ ]: model_details

Out[150]: {'city': ['hybrid_pipeline_software_specs'],
 'software_spec_id': {'id': 'ab0e180-f32c-503c-a702-4f23aaf77194',
 'name': 'default_py3.8',
 'type': 'skits-learn_0.20',
 'metadata': {'created_at': '2022-02-15T17:50:07.127Z',
 'id': 'bf051ab5-055f-404d-85ab-115a2f088079',
 'modified_at': '2022-02-15T17:50:11.507Z',
 'name': 'tool_demand_forecasting_deployment',
 'owner': 'BRIID-6660108WA',
 'resource_key': '72b5f51af-9f00-4073-b257-1f44b309acc0',
 'space_id': '7dfe1f8e-d805-4a0f-ad85-f5d036c79d2'},
 'system': {'warnings': []}}

In [ ]: model_id

Out[150]: 'bf051ab5-055f-404d-85ab-115a2f088079'

In [ ]: client.connections.list_data_source_types()
```

NAME	DATASOURCE_ID	TYPE	STATUS
Informa	029e011e-fa73-d800-b7d2-1a01a390a1c9	database	active
postgresql-ibmcloud	0a6ed1bf-510c-a0f8-a090-4a39a308c11	database	active
googlecloudstorage	0b0f000a-ba08-4702-a070-c120f1108f75	file	active
awsdata	05c5820a-0020-0507-b10a-c71a47e7608c	database	active
salesforce	00047010-07f4a-0015-0020-c05f13a17a45	database	active
datacube-ibmcloud	0bf00a0d-c7c3-d2c3-b770-00030232a400	database	active
cosmos	0c41740-2572-110a-0707-2a720e0812f5	file	active
rdbe-datasage	0c023d10-b006-3042-a071-7703553c3904	database	active
mysql-cosmos	0c023d10-b006-3042-a071-7703553c3904	database	active
hive	0f02f0eb-0070-4020-a100-070000131a0d	database	active
cognos-analytics	11f30293-a1c1-4043-b007-0a022f454294	file	active
cassandra-datasage	12362631-d3f5-ad05-0002-c702a00400d1	generic	active
blazex-ibmcloudstorage	103a0711-4075-4a19-b000-295c4f403117	file	active
elasticsearch	100071a0-0a05-0030-0004-a00000000000	file	active
unipharmacy-datasage	113a0e00-0a2d-1270-b00d-0000001f7f75	database	active
data	27510100-b7d2-4072-0511-150a0a197040	generic	active
azurefilestorage	2a704fa1-c770-0007-0071-a3c3d5f5a0a1	file	active
bigsql	3a000504-f13a-d700-b0c3-f990a0a0000a	database	active
newlake	2f1c132f-b50c-a045-b004-df012a1178a5	database	active
redshift	31100000-f00c-4100-b00a-0070422fa1d0f	database	active
dbserver	370c10f7-a075-470a-ba07-70a00c000011	database	active
db	30a7a0f0-e15a-d100-0000-0a00013a3040	database	active
salesforce-datasage	3a000010-2500-0070-07c2-5f0000000015	generic	active
http	42101204-000f-a00a-b0c0-1c0a0a20f000	file	active
cloudant	4a000020-0010-ad00-a0c0-000c3a3a7004	file	active
sqlserver	4a000020-0010-ad00-a0c0-000c3a3a7004	database	active
ipfsnode	40070002-fac2-4702-0901-a0101c1a0004	database	active
db	00a00002-250f-0210-a000-0a0000f00a01	database	active
cloudobjectstorage	40f300d1-0000-0000-0000-b0f05720c0f9	file	active
ibmcloud	500010f0-001f-a0f2-a0f1-c10000000000	database	active
dropbox	5070000c-fa01-d107-a00a-a1000a100010	file	active
netezza-datasage	60a00010-0010-0070-0140-90a0f7250000	database	active
starburstlake	60a00000-07c4-b0c3-0000-00000001f700	file	active
googlecloud	600c2003-3003-0300-0400-f0d0f0f30f7a	file	active
sales	0070a0f0-0000-a000-a000-a0000a000000	database	active

NAME	DATASOURCE_ID	TYPE	STATUS
spine	00000010-0000-0000-0000-000000000000	generic	active
ibmcloud	00007000-0000-0020-0070-720000700000	file	active
sqlserver	010c0100-0000-1100-00c1-0002ac120000	database	active
capofata	02000110-0000-0000-0000-000000000000	generic	active
arby	02000110-0000-0000-0000-000000000000	database	active
hdfs-analyticengine	00000010-0000-0000-0000-000000000000	file	active
oracle-amazon	00000000-0000-0000-0000-000000000000	database	active
db	00000000-0000-0000-0000-000000000000	database	active
megadot-ibmcloud	00000000-0000-0000-0000-000000000000	database	active
bigquery	00000000-0000-0000-0000-000000000000	database	active
postgresql-amazon	00000000-0000-0000-0000-000000000000	database	active
dbserver	00000000-0000-0000-0000-000000000000	database	active
teradata	00000000-0000-0000-0000-000000000000	database	active
oracle	00000000-0000-0000-0000-000000000000	database	active
ibm	00000000-0000-0000-0000-000000000000	file	active
azureblobstorage	00000000-0000-0000-0000-000000000000	file	active
mysql-amazon	00000000-0000-0000-0000-000000000000	database	active
tableau	00000000-0000-0000-0000-000000000000	file	active
amazon	00000000-0000-0000-0000-000000000000	file	active
amazon	00000000-0000-0000-0000-000000000000	database	active
mysql	00000000-0000-0000-0000-000000000000	database	active
hdfs-apache	00000000-0000-0000-0000-000000000000	file	active
netezza	00000000-0000-0000-0000-000000000000	database	active
datawarehouse	00000000-0000-0000-0000-000000000000	database	active
megadot	00000000-0000-0000-0000-000000000000	database	active
dbserver	00000000-0000-0000-0000-000000000000	database	active
tableau	00000000-0000-0000-0000-000000000000	generic	active
cassandra	00000000-0000-0000-0000-000000000000	database	active
dashdb	00000000-0000-0000-0000-000000000000	database	active
redshift-ibm	00000000-0000-0000-0000-000000000000	generic	system
ftp	00000000-0000-0000-0000-000000000000	file	active
db-datasage	00000000-0000-0000-0000-000000000000	database	active
oracle-datasage	00000000-0000-0000-0000-000000000000	generic	active
postgresql	00000000-0000-0000-0000-000000000000	database	active
postgresql	00000000-0000-0000-0000-000000000000	database	active
tableau-datasage	00000000-0000-0000-0000-000000000000	generic	active
airbnb	00000000-0000-0000-0000-000000000000	database	active



