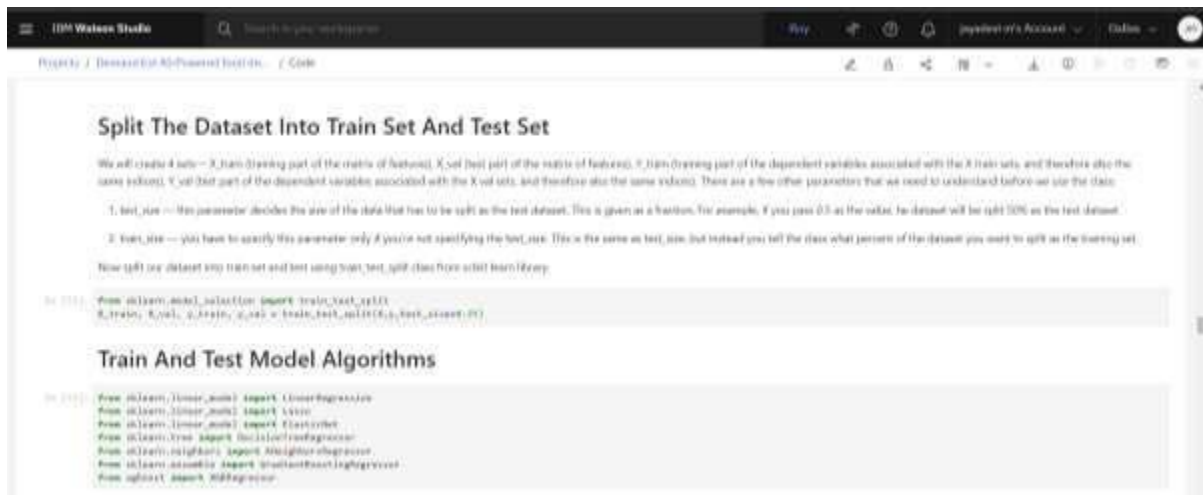


SPRINT-3

MODEL BUILDING

| | |
|---------------|---|
| Team ID | PNT2022TMID14821 |
| Project Title | DemandEst-AI Powered Food Demand Forecaster |

Train and test model algorithms



The screenshot shows a Jupyter Notebook titled "Split The Dataset Into Train Set And Test Set". It contains two sections of code. The first section, "Split The Dataset Into Train Set And Test Set", explains the parameters for `train_test_split` and provides the following code:

```
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 second section, "Train And Test Model Algorithms", lists the models to be trained and tested:

```
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn.linear_model import ElasticNet
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn import metrics
```

Model Evaluation



The screenshot shows a Jupyter Notebook titled "Model Evaluation". It contains three sections of code. The first section, "Model Evaluation", explains the Mean Squared Error (MSE) and provides the following code:

```
MSE = metrics.mean_squared_error(y_val, y_pred_val)
print("MSE: ", MSE)
```

The second section, "Lasso Regression", provides the following code:

```
L = Lasso()
L.fit(X_train, y_train)
y_pred_L = L.predict(X_val)
y_grad_L = 0
from sklearn import metrics
print("MSE: ", metrics.mean_squared_error(y_val, y_pred_L))
```

The third section, "Elastic Net", provides the following code:

```
EN = ElasticNet()
EN.fit(X_train, y_train)
y_pred_EN = EN.predict(X_val)
y_grad_EN = 0
from sklearn import metrics
print("MSE: ", metrics.mean_squared_error(y_val, y_pred_EN))
```

```
Project / DemandEst.RtfDemandEstimator - / Code
# Save the model to a file
save_pickle(model, "model.pkl")

# Load the model from a file
loaded_model = load_pickle("model.pkl")

# Predict the output using the model
X_test = test_data[features].values

# Predict the output using the model
y_pred = loaded_model.predict(X_test)
```

Save the Model

```
Project / DemandEst.RtfDemandEstimator - / Code

Save The Model

pickle is used for serializing and de-serializing Python object structures, also called marshalling or Pickling. Serialisation refers to the process of converting an object in memory to a byte stream that can be stored on disk or sent over a network. Later on, this character stream can then be retrieved and de-serialised back to a Python object here, Df is our decision tree model saving as demand.pkl file. We write binary in bytes.

In [140]: import pickle
          pickle.dump(Df, open("Demand.pkl", "wb"))
```

Predicting the output using the model

```
Project / DemandEst.RtfDemandEstimator - / Code

Predicting The Output Using The Model

Here, we are loading X_test which we are using to test the model to predict the number of orders by giving input to the model built.

In [141]: test_data = pd.read_csv('test_data.csv')
          test_data = test_data.drop('order_id', axis=1)
          test_data = test_data.drop('order_id', axis=1)

          X_test = test_data[features].values
          y_test = test_data['order_id'].values

          # Create a LabelEncoder object
          le = LabelEncoder()
          y_test = le.fit_transform(y_test)

          # Create a LabelEncoder object
          le = LabelEncoder()
          y_test = le.fit_transform(y_test)

          # Create a LabelEncoder object
          le = LabelEncoder()
          y_test = le.fit_transform(y_test)
```



```
jupyter Code (last checkpoint 14 hours ago, some cells changed)
File Edit View Insert Cell Format Help
Python 3 (system)

In [ ]:
tensorflow-1.15-py3.6 2075a275-70b7-4200-a012-5a07f495b08c base
pytorch-1.7-py3.6 2080f570-2087-4970-ae00-01f9a979da13 base
spark-mllib-2.4 2051f780-bc40-d8d0-d8d0-5c579133d875 base
pytorch-orm-1.1-py3.6-soft 32983caa-2f32-4480-8005-d3d87a0b6b7e base
spark-mllib-1.6-py3.7 30a0f9ee-8770-509a-d02a-e976787d0a09 base
spark-mllib-1.4 30a0d1fa-e92b-afac-9c55-d70ada21328e base
xgboost-0.82-py3.6 39031a0d-5f93-41d0-a044-60273530300a base
pytorch-orm-1.2-py3.6-soft 40f0a00e-7010-a020-013a-f0030afafed2 base
default-r10py36 4112d710-05f0-5a71-b055-d500220faf00 base
xgboost-0.82-py3.6 43093010-d0a0-507f-903a-42400a1e0f7f base
spark-mllib-1.4-py3.6 402a0d77-09ee-8c4f-a537-a12d0021099f base
xgboost-0.80-py3.6 41f8d0c2-1303-4c10-05e1-03ac90530d01 base
pytorch-orm-1.1-py3.6 50f9502a-bc10-430b-bc94-d0d0d200c000 base
xgboost-0.82-py3.6 52c57130-007a-370e-0720-49e750a2c0de base
spark-mllib-2.4-scala-2.11 55a70f60-7320-d0e5-9f00-9a0d5a443af1 base
spark-mllib-1.6 5c100ca2-0077-3c20-9d70-f1d4a09ff1e9 base
xgboost-0.82-py3.6 5c2a77fa-0004-5e77-000f-011200000000 base
spss-modeler-18.4 5c3ca07e-307f-d02a-05a3-0b33d1d0e000 base
code-py3.6 5d52320f-c000-507a-02d0-700d7001c040 base
xgboost-0.82-py3.6 61200d22-10aa-5100-00f0-f520f0000007 base
pytorch-orm-1.7-py3.6 53405d0c-0902-50f0-a034-a000a7845000 base
spark-mllib-1.3-py3.6 60d000c3-c000-4f00-000f-00f020000000 base
tensorflow-2.4-py3.7 05e171d7-7000-5200-0a00-f01000000000 base
spss-modeler-18.2 0070d0d0-0000-4117-0000-007000000000 base
pytorch-orm-1.2-py3.6 0f0a000d-20d0-05ff-01e0-b0e000000000 base
spark-mllib-2.4-scala-2.11 00c1e9f0-b000-0170-02cf-0570a2100000 base
spark-mllib-1.4-py3.7 0da00000-b000-0170-0120-000000000000 base
c4pfe-1.0-py3.6 00c00000-d000-4100-0100-00c000000000 base
pytorch-orm-1.2-py3.7 011c0001-0000-5013-0020-000000000000 base
code-py3.6 02c7900e-0011-0000-0707-070000000000 base
tensorflow-1.15-py3.6-hadoop 00c00000-d000-5000-0100-00c000000000 base
hybrid-0.1 0c1c0000-0200-5000-007a-0f51c2700000 base
pytorch-orm-1.1-py3.7 0070000f-0012-500f-010c-391000000000 base
c4pfe-1.0-py3.6 00000000-0000-0000-0000-000000000000 base

Note: only first 30 records were displayed, to display more use 'limit' parameter.
```

```
jupyter Code (last checkpoint 14 hours ago, some cells changed)
File Edit View Insert Cell Format Help
Python 3 (system)

In [ ]:
software_spec_id = client.software_specifications.get_id_by_name("default_py3.6")
software_spec_id

Out[185]: 'ab0e10e-f1ca-502c-0702-0723aaf77194'

In [ ]:
model_details = client.repository.store_model(model = M0, meta_params={
    client.repository.model_metadata.NAME: "Test model for creating deployment",
    client.repository.model_metadata.TYPE: "scikit-learn-0.22",
    client.repository.model_metadata.SOURCE_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[186]: {'entity': {'hybrid_pipeline_software_specs': [],
  'software_specs': [{'id': 'ab0e10e-f1ca-502c-0702-0723aaf77194',
    'name': 'default_py3.6',
    'type': 'scikit-learn-0.22'}]},
  'metadata': {'created_at': '2020-02-15T17:50:02.122Z',
    'id': '1b055a0b-000f-4000-0200-110d2f000000',
    'modified_at': '2020-02-15T17:50:11.500Z',
    'name': 'Test model for creating deployment',
    'owner': '10010-0000010000',
    'resource_key': '700f5240-0f00-4075-0257-1f443000a0c0',
    'spec_id': '700f5240-0f00-4075-0257-1f443000a0c0',
    'system': {'warnings': []}}

In [ ]:
model_id

Out[186]: '1b055a0b-000f-4000-0200-110d2f000000'

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



The image shows a JupyterLab interface. On the left is a file browser pane with a tree view showing a hierarchy of folders and files. The main area on the right displays the contents of the selected folder, 'data', which is a table with 5 columns: Name, Size, Type, Last Modified, and Action. The table lists various files and folders, including 'data', 'data1', 'data2', 'data3', 'data4', 'data5', 'data6', 'data7', 'data8', 'data9', 'data10', 'data11', 'data12', 'data13', 'data14', 'data15', 'data16', 'data17', 'data18', 'data19', 'data20', 'data21', 'data22', 'data23', 'data24', 'data25', 'data26', 'data27', 'data28', 'data29', 'data30', 'data31', 'data32', 'data33', 'data34', 'data35', 'data36', 'data37', 'data38', 'data39', 'data40', 'data41', 'data42', 'data43', 'data44', 'data45', 'data46', 'data47', 'data48', 'data49', 'data50', 'data51', 'data52', 'data53', 'data54', 'data55', 'data56', 'data57', 'data58', 'data59', 'data60', 'data61', 'data62', 'data63', 'data64', 'data65', 'data66', 'data67', 'data68', 'data69', 'data70', 'data71', 'data72', 'data73', 'data74', 'data75', 'data76', 'data77', 'data78', 'data79', 'data80', 'data81', 'data82', 'data83', 'data84', 'data85', 'data86', 'data87', 'data88', 'data89', 'data90', 'data91', 'data92', 'data93', 'data94', 'data95', 'data96', 'data97', 'data98', 'data99', 'data100', 'data101', 'data102', 'data103', 'data104', 'data105', 'data106', 'data107', 'data108', 'data109', 'data110', 'data111', 'data112', 'data113', 'data114', 'data115', 'data116', 'data117', 'data118', 'data119', 'data120', 'data121', 'data122', 'data123', 'data124', 'data125', 'data126', 'data127', 'data128', 'data129', 'data130', 'data131', 'data132', 'data133', 'data134', 'data135', 'data136', 'data137', 'data138', 'data139', 'data140', 'data141', 'data142', 'data143', 'data144', 'data145', 'data146', 'data147', 'data148', 'data149', 'data150', 'data151', 'data152', 'data153', 'data154', 'data155', 'data156', 'data157', 'data158', 'data159', 'data160', 'data161', 'data162', 'data163', 'data164', 'data165', 'data166', 'data167', 'data168', 'data169', 'data170', 'data171', 'data172', 'data173', 'data174', 'data175', 'data176', 'data177', 'data178', 'data179', 'data180', 'data181', 'data182', 'data183', 'data184', 'data185', 'data186', 'data187', 'data188', 'data189', 'data190', 'data191', 'data192', 'data193', 'data194', 'data195', 'data196', 'data197', 'data198', 'data199', 'data200', 'data201', 'data202', 'data203', 'data204', 'data205', 'data206', 'data207', 'data208', 'data209', 'data210', 'data211', 'data212', 'data213', 'data214', 'data215', 'data216', 'data217', 'data218', 'data219', 'data220', 'data221', 'data222', 'data223', 'data224', 'data225', 'data226', 'data227', 'data228', 'data229', 'data230', 'data231', 'data232', 'data233', 'data234', 'data235', 'data236', 'data237', 'data238', 'data239', 'data240', 'data241', 'data242', 'data243', 'data244', 'data245', 'data246', 'data247', 'data248', 'data249', 'data250', 'data251', 'data252', 'data253', 'data254', 'data255', 'data256', 'data257', 'data258', 'data259', 'data260', 'data261', 'data262', 'data263', 'data264', 'data265', 'data266', 'data267', 'data268', 'data269', 'data270', 'data271', 'data272', 'data273', 'data274', 'data275', 'data276', 'data277', 'data278', 'data279', 'data280', 'data281', 'data282', 'data283', 'data284', 'data285', 'data286', 'data287', 'data288', 'data289', 'data290', 'data291', 'data292', 'data293', 'data294', 'data295', 'data296', 'data297', 'data298', 'data299', 'data300', 'data301', 'data302', 'data303', 'data304', 'data305', 'data306', 'data307', 'data308', 'data309', 'data310', 'data311', 'data312', 'data313', 'data314', 'data315', 'data316', 'data317', 'data318', 'data319', 'data320', 'data321', 'data322', 'data323', 'data324', 'data325', 'data326', 'data327', 'data328', 'data329', 'data330', 'data331', 'data332', 'data333', 'data334', 'data335', 'data336', 'data337', 'data338', 'data339', 'data340', 'data341', 'data342', 'data343', 'data344', 'data345', 'data346', 'data347', 'data348', 'data349', 'data350', 'data351', 'data352', 'data353', 'data354', 'data355', 'data356', 'data357', 'data358', 'data359', 'data360', 'data361', 'data362', 'data363', 'data364', 'data365', 'data366', 'data367', 'data368', 'data369', 'data370', 'data371', 'data372', 'data373', 'data374', 'data375', 'data376', 'data377', 'data378', 'data379', 'data380', 'data381', 'data382', 'data383', 'data384', 'data385', 'data386', 'data387', 'data388', 'data389', 'data390', 'data391', 'data392', 'data393', 'data394', 'data395', 'data396', 'data397', 'data398', 'data399', 'data400', 'data401', 'data402', 'data403', 'data404', 'data405', 'data406', 'data407', 'data408', 'data409', 'data410', 'data411', 'data412', 'data413', 'data414', 'data415', 'data416', 'data417', 'data418', 'data419', 'data420', 'data421', 'data422', 'data423', 'data424', 'data425', 'data426', 'data427', 'data428', 'data429', 'data430', 'data431', 'data432', 'data433', 'data434', 'data435', 'data436', 'data437', 'data438', 'data439', 'data440', 'data441', 'data442', 'data443', 'data444', 'data445', 'data446', 'data447', 'data448', 'data449', 'data450', 'data451', 'data452', 'data453', 'data454', 'data455', 'data456', 'data457', 'data458', 'data459', 'data460', 'data461', 'data462', 'data463', 'data464', 'data465', 'data466', 'data467', 'data468', 'data469', 'data470', 'data471', 'data472', 'data473', 'data474', 'data475', 'data476', 'data477', 'data478', 'data479', 'data480', 'data481', 'data482', 'data483', 'data484', 'data485', 'data486', 'data487', 'data488', 'data489', 'data490', 'data491', 'data492', 'data493', 'data494', 'data495', 'data496', 'data497', 'data498', 'data499', 'data500', 'data501', 'data502', 'data503', 'data504', 'data505', 'data506', 'data507', 'data508', 'data509', 'data510', 'data511', 'data512', 'data513', 'data514', 'data515', 'data516', 'data517', 'data518', 'data519', 'data520', 'data521', 'data522', 'data523', 'data524', 'data525', 'data526', 'data527', 'data528', 'data529', 'data530', 'data531', 'data532', 'data533', 'data534', 'data535', 'data536', 'data537', 'data538', 'data539', 'data540', 'data541', 'data542', 'data543', 'data544', 'data545', 'data546', 'data547', 'data548', 'data549', 'data550', 'data551', 'data552', 'data553', 'data554', 'data555', 'data556', 'data557', 'data558', 'data559', 'data560', 'data561', 'data562', 'data563', 'data564', 'data565', 'data566', 'data567', 'data568', 'data569', 'data570', 'data571', 'data572', 'data573', 'data574', 'data575', 'data576', 'data577', 'data578', 'data579', 'data580', 'data581', 'data582', 'data583', 'data584', 'data585', 'data586', 'data587', 'data588', 'data589', 'data590', 'data591', 'data592', 'data593', 'data594', 'data595', 'data596', 'data597', 'data598', 'data599', 'data600', 'data601', 'data602', 'data603', 'data604', 'data605', 'data606', 'data607', 'data608', 'data609', 'data610', 'data611', 'data612', 'data613', 'data614', 'data615', 'data616', 'data617', 'data618', 'data619', 'data620', 'data621', 'data622', 'data623', 'data624', 'data625', 'data626', 'data627', 'data628', 'data629', 'data630', 'data631', 'data632', 'data633', 'data634', 'data635', 'data636', 'data637', 'data638', 'data639', 'data640', 'data641', 'data642', 'data643', 'data644', 'data645', 'data646', 'data647', 'data648', 'data649', 'data650', 'data651', 'data652', 'data653', 'data654', 'data655', 'data656', 'data657', 'data658', 'data659', 'data660', 'data661', 'data662', 'data663', 'data664', 'data665', 'data666', 'data667', 'data668', 'data669', 'data670', 'data671', 'data672', 'data673',

