ADVANCED DATA SCIENCE AND ARCHITECTURE (INFO 7390)

BREAST CANCER DETECTION AND ANALYSIS

Final Project Report

Team 5

DHRUV KANAKIA

AKILAN RAJENDIRAN

Under the guidance of

Sri Krishnamurthy

SUMMARY

This report summarizes the analysis performed on two Breast Cancer Datasets. The links for two are:

1. http://www.cbioportal.org/study?id=brca_metabric#clinical
2. http://www.bcsc-research.org/data/bcsc_data_definitions.html

Using the two datasets we created two use cases:

i. Predict the risk of having Breast Cancer based on the input parameters given by the user.
ii. Classify the Tumor based on different user vitals and cancer cells report.

The problem statement is divided into 4 sections:

1) Data Wrangling

- Web scraping the data from the above links and pre-processing
- Exploratory Data Analysis on Tableau, Python.

2) Dockerizing the process:

- Two docker images for web scraping and uploading the data on Azure Blob.
- Airflow pipeline

3) Building and Evaluating models:

- Prediction using Random Forest, SVM, Neural Network, Logistic Regression
- Classification using Random forest, SVM, Neural Network

4) Creating User interface for Oncologists and Females (User)

- **User:** will be able to check the risk/probability of her having breast cancer based on the prediction model and the input parameters.
- **Oncologist(Doctor):** will be able to classify what type of tumor is it based on classification model.

PART 1: DATA INGESTION AND WRANGLING

1.1. Data Wrangling and Pre-Processing

1. **Breast Cancer Surveillance Consortium Dataset(**The dataset has around 2,392,998 mammograms from women who participated in the study). The dataset consists of a .txt file and it includes information about women during the time of the study like:

Age, Menopause status, density, agegrp, race, bmi, cancer status etc.

The picture attached below shows the web url from which we are Web Scraping the Data.

CHALLENGE:

The challenge was to programmatically get through the login page and extract the dataset. The link to the landing page of the above screenshot : http://www.bcsc-research.org/rfdataset/app2/protected/Logon.aspx?ReturnUrl=%2frfdataset%2fapp2%2fprotected%2frisk_dataset.zip

SOLUTION:

- We tried going pass the login page using request libraries but couldn't get pass it. Finally, we managed to extract the dataset using the mechanical soup library and here's the code for it:

```
url = "http://www.bcsc-research.org/rfdataset/app2/protected/Logon.aspx?ReturnUrl=%2frfdataset%2fapp2%2fprotected%2frisk_dataset.
url2 = "http://www.bcsc-research.org/rfdataset/app2/protected/risk_dataset.zip"
filename='zipfile.zip'
s = requests.Session()
browser = ms.Browser(session = s)
login_page = browser.get(url)
login_form = login_page.soup.find("form",{"id":"form1"})
#print(login_form)
login_form.find("input", {"name":"UserEmail"})["value"] = 'bcsclogin'
login_form.find("input", {"name":"UserPass"})["value"] = 'bcsc_pass'
response = browser.submit(login_form, login_page.url)
login_page2 = browser.get(url2)
with open(os.path.join(final,filename), 'wb') as f:
#                print(link.text)
        for chunk in login_page2.iter_content(chunk_size=1024):
            if chunk: # filter out keep-alive new chunks
                f.write(chunk)
                #print('zip file created')
```

- The file downloaded by the above code was in .zip format and used the zipfile library to extract the zip file. The .txt file present in the zipfile was read using the pandas module, column names were allotted and then the file was written on local as 'Risk.csv'.
- Below is the screenshot which shows how we gave column names to the file replaced the missing values with space characters(because we are going to clean the file using MICE on Azure).

```
names=["menopaus","agegrp","density","race","Hispanic","bmi","agefirst",
                "noOfFirstDegRelativesCan","prevcanProc","lastMamm","surgMeno","hrt","invasive"
                ,"cancerStatus","type","count"]
```

```
import zipfile
for files in glob.glob(os.path.join(final, '*.zip')):
        with zipfile.ZipFile(files) as zip_ref:
            df = pd.read_csv(zip_ref.open('risk.txt'),delim_whitespace=True,names=names)
```

```
a=df.iloc[:, [1,3,4,5,6,7,8,9,10,12]]
a = a.replace(9, '', regex=True)
df.iloc[:, [1,3,4,5,6,7,8,9,10,12]]=a
```

```
df.to_csv(final+'Risk.csv')
print('Risk csv ready')
```

PRE-PROCESSING:

PROBLEM:

Since there are lot of missing values in this file we can neither ignore them nor just fill in mean/mode/random value as it is a patient dataset and nothing can be generalized.

SOLUTION:

After looking at resources online we came across something call DATA IMPUTATION techniques and how missing data can be handled by looking at patterns/ predicting them. Reading about different techniques we found MICE would suit the best for this problem and we implemented it on Azure.

Screenshot shows how the missing value is handled using Azure block Handling missing values(MICE)

2. Breast Cancer (METABRIC, Nature 2012 & Nat Commun 2016:

This dataset consists of 2509 patients who have participated in the study and it includes details like type of cancer, treatments used, gene and tumor information.

We used their library(cgdsr) to web scrape the data using R

The screenshot attached below shows how we are web scraping the data using its WEB API.

```
library(cgdsr)
# Create CGDS object
mycgds = CGDS("http://www.cbioportal.org/")
if ((dir.exists("Data"))==TRUE){
  print('It exists')
}else{
  dir.create("Data")
  print('Created')
}
a<-getClinicalData(mycgds, "brca_metabric_all")
```

Pre-Processing:

There are few categorical columns which needs to be converted into numbers for modeling.

Besides, we have also created manual clusters for age group <=62 and age group >62.

Following is the screenshot for pre processing of this file and storing it in local.

```
desiredData<- a[,c(19,4,5,6,9,13,15,16,14,1,22,23,7,18,24,27,28)]
#####Selecting Deisred Columns from dataset #####
#########Clean Dataframe############
levels(desiredData$CELLULARITY)<-c(2,0,1)
levels(desiredData$CHEMOTHERAPY)<-c(0,1)
levels(desiredData$ER_IHC)<-c(0,1)
levels(desiredData$HER2_STATUS)<-c(0,1)
levels(desiredData$HORMONE_THERAPY)<-c(0,1)
levels(desiredData$INFERRED_MENOPAUSAL_STATE)<-c(1,0)
levels(desiredData$HISTOLOGICAL_SUBTYPE)<-c(1,2,3)
levels(desiredData$OS_STATUS)<-c(0,1)
levels(desiredData$PR_STATUS)<-c(0,1)
levels(desiredData$LATERALITY)<-c(1,2)
levels(desiredData$RADIO_THERAPY)<-c(1,2)
#levels(desiredData$TumorStage)
desiredDataCleanFile<-(desiredData[complete.cases(desiredData), ])
#########Clean Dataframe############
##############Clustering#####################
hist(desiredDataCleanFile$AGE_AT_DIAGNOSIS)
d<-density(desiredDataCleanFile$AGE_AT_DIAGNOSIS)
plot(d)
polygon(d, col="blue", border="black")
dataLessthan60<-subset(desiredDataCleanFile, AGE_AT_DIAGNOSIS <=62)
dataGreaterthan60<-subset(desiredDataCleanFile, AGE_AT_DIAGNOSIS > 62)
nrow(dataLessthan60)
nrow(dataGreaterthan60)
write.csv(desiredDataCleanFile,"Data/DesiredData.csv")
write.csv(dataLessthan60,"Data/dataLessthan60.csv")
write.csv(dataGreaterthan60,"Data/dataGreaterthan60.csv")
```

Part 2: Exploratory Data Analysis

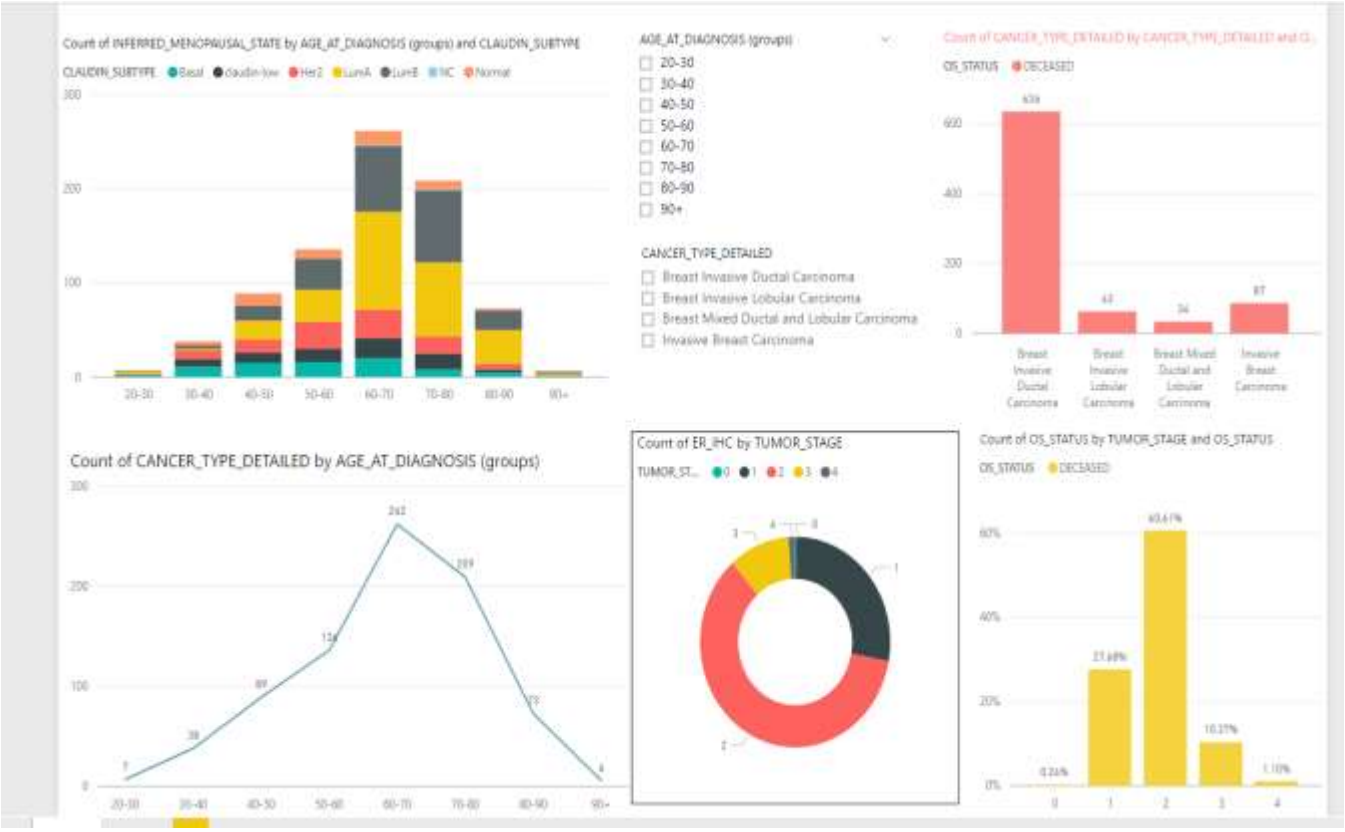
2.1 Analysis – Power BI

EDA on Metabric Dataset:

The first dashboard has 5 visualizations that explore the different aspects in the Metabric Dataset.

- The first plot shows different Claudin subtypes over the age group. The claudin low subtype tumors one of the important factors for evaluation of unique biology of structures and heterogeneity of Breast Cancer.
- The second plot shows the count of Different cancer subtypes in this dataset. Our multi class classification is built on this subtype attribute and it was necessary to explore that column.
- Age also played an important role among females getting breast cancer. Thus, the third plot analyzes the age group present in the dataset in the form of bins.
- Just like how we have Claudian subtype, the other important attribute in the dataset is ER_IHC. This is a test that is performed to analyze whether or not the cancer cells have HER2. Based on this information further steps are taken.
- The last plot shows vital information about tumor stages. According to the plot(dataset) tumor stages at level 2 have the highest deceased rate.

The dashboard screenshot is attached below:

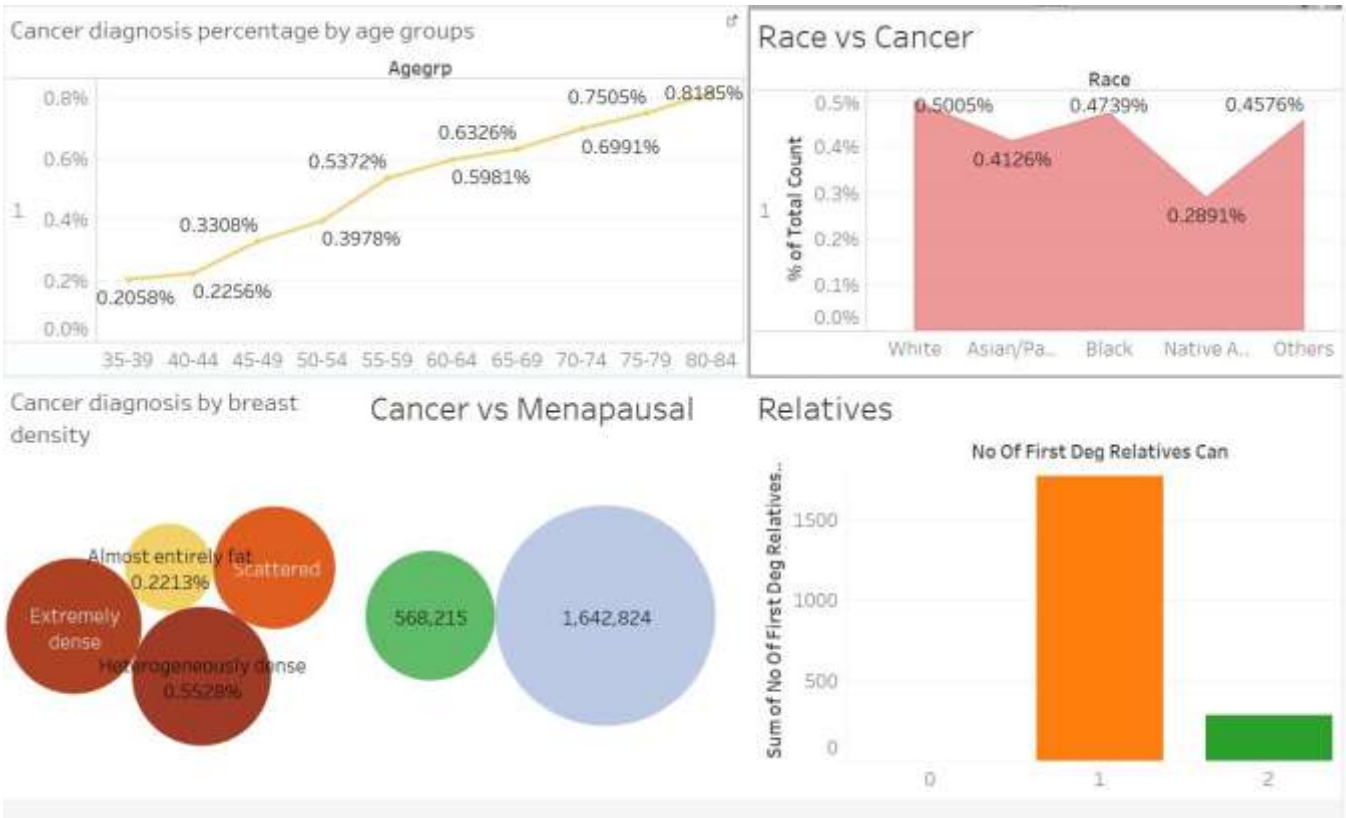


2. The second dashboard

EDA on BCSC Dataset(Power BI):

This dashboard also has five visualizations for exploring the Risk Dataset.

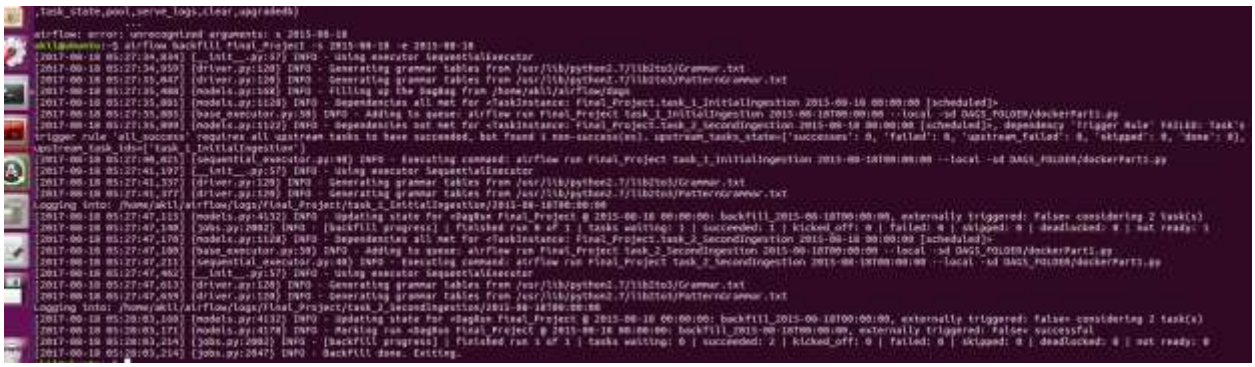
- The first visualization summarizes the percentage of cancer diagnosis over the years. We can clearly see that with increase in age there is high probability of getting cancer.
- The second visualization summarizes the information of race and cancer diagnosis. From, the graph we can clearly see that Whites and Black have the highest percentage of getting bgreast cancer as per dataset.
- The third visualization is the cancer diagnosis visualization which measures the density of breast cancer. From the dataset we can conclude that majority of samples are from Extremely and Heterogeneously data
- The fourth visualization analyzes the menopause status in the dataset where in it indicates number of patients who in are post and pre menopause and are suffering breast cancer.
- The last visualization describes the risk of breast cancer on the basis of number of first degree Relatives

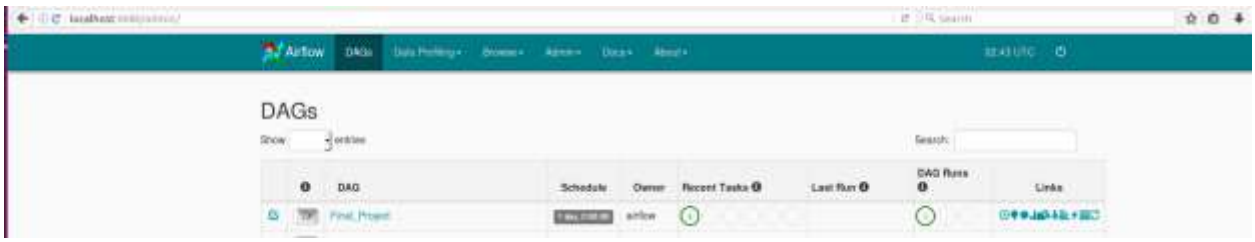


PART II: DOCKERIZING THE PROCESS

PIPELINING THROUGH Airflow

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from datetime import datetime, timedelta
from airflow.operators.python_operator import PythonOperator

default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': datetime(2017, 8, 18),
    'email': ['airflow@airflow.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
    # 'queue': 'bash_queue',
    # 'pool': 'backfill',
    # 'priority_weight': 10,
    # 'end_date': datetime(2017, 8, 18),
}

dag = DAG('Final_Project', default_args=default_args)

# t1 and t2 are examples of tasks created by instantiating operators
t1 = BashOperator(
    task_id='task_1_InitialIngestion',
    bash_command='sudo docker run -i -t r_scraping',
    dag=dag)

t2 = BashOperator(
    task_id='task_2_SecondIngestion',
    bash_command='sudo docker run -i -t akil06/midterm:1.1',
    dag=dag)

t2.set_upstream(t1)
```





DOCKER STEPS AND PROCESS

IMAGE 1:

The first image is a R image that web scrapes Metabric Data, preprocesses it and stores it in data container of the image.

Here's the DOCKERFILE:

```
FROM r-base
RUN mkdir -p /Data
COPY Metabric_Web_Scraping.R /

RUN Rscript -e "install.packages('cgdsr')"

CMD ["Rscript","Metabric_Web_Scraping.R"]
```

Built the docker image using with name r_base

```
$ docker build -t r_base2 .
Sending build context to Docker daemon  25.82MB
Step 1/5 : FROM r-base
 ---> a2a943b96c51
Step 2/5 : RUN mkdir -p /Data
 ---> Using cache
 ---> fa6deb748df4
Step 3/5 : COPY Metabric_Web_Scraping.R /
 ---> Using cache
 ---> f8eab00754ea
Step 4/5 : RUN Rscript -e "install.packages('cgdsr')"
 ---> Using cache
 ---> 8da48d6003d7
Step 5/5 : CMD Rscript Metabric_Web_Scraping.R
 ---> Using cache
 ---> 46f47c3b5ef9
Successfully built 46f47c3b5ef9
Successfully tagged r_base2:latest
SECURITY WARNING: You are building a Docker image from Windows against a non-Wind
check and reset permissions for sensitive files and directories.
```

Pushing the image to the Docker hub. The image name is **akil06/finalproject:image1.2**

```
$ docker push akil06/finalproject:image1.2
The push refers to a repository [docker.io/akil06/finalproject]
cea863567a62: Pushed
10aab11d2885: Layer already exists
f9c651199317: Layer already exists
ee44d7fd47d9: Layer already exists
1b27dda8b3e0: Layer already exists
759042517af4: Layer already exists
b293c96c408c: Layer already exists
3b310fb368a5: Layer already exists
7d33f12cb54f: Layer already exists
af6a2bf56818: Layer already exists
image1.2: digest: sha256:9f0df1e7f19957aaa1a2d2617e480117a6e6e6721dd4d1766fb19030645ceba4 size: 2412
```

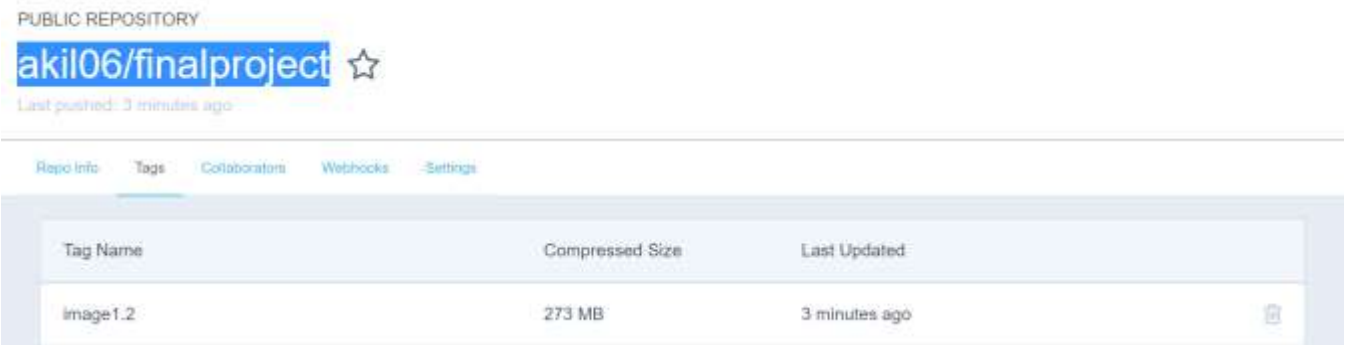


Image can be called using command docker pull **akil06/finalproject:image1.2**

IMAGE 2:

The second image is on python base image. It web scrapes the BCSC data and uploads the files onto blob.

Here's the Docker file:

```
FROM python:latest
RUN mkdir -p /Data
COPY BCSC_DataSet_WebScraping_Uploadig.py /
COPY /Data /Data
COPY config.json /
RUN pip install AzureML
RUN pip install lxml
RUN pip install Mechanicalsoup
RUN pip install Azure

CMD ["python","BCSC_DataSet_WebScraping_Uploadig.py"]
```

Built the image using python base image. Here's the command and screenshot of it.

```
$ docker build -t python .
Sending build context to Docker daemon  25.82MB
Step 1/10 : FROM python:latest
 ---> c0f953e122ee
Step 2/10 : RUN mkdir -p /Data
 ---> Running in 98d4be9c4d38
 ---> ad15e7c0f5bd
Removing intermediate container 98d4be9c4d38
Step 3/10 : COPY BCSC_DataSet_WebScraping_Uploadig.py /
 ---> eac55bddc409
Removing intermediate container b884769e8a8c
Step 4/10 : COPY /Data /Data
 ---> 55b4b815cbc7
Removing intermediate container e69d91659e70
Step 5/10 : COPY config.json /
 ---> 7335009e0446
Removing intermediate container f317a6a2b1bf
Step 6/10 : RUN pip install AzureML
 ---> Running in 7f2c89f441f8
Requirement already satisfied: AzureML in /usr/local/lib/python3.6/site-packages
Requirement already satisfied: requests in /usr/local/lib/python3.6/site-packages (from AzureML)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.6/site-packages (from AzureML)
Requirement already satisfied: pandas in /usr/local/lib/python3.6/site-packages (from AzureML)
Requirement already satisfied: idna<2.7,>=2.5 in /usr/local/lib/python3.6/site-packages (from requests->AzureML)
Requirement already satisfied: urllib3<1.23,>=1.21.1 in /usr/local/lib/python3.6/site-packages (from requests->AzureML)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /usr/local/lib/python3.6/site-packages (from requests->AzureML)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/site-packages (from requests->AzureML)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/site-packages (from python-dateutil->AzureML)
Requirement already satisfied: numpy>=1.7.0 in /usr/local/lib/python3.6/site-packages (from pandas->AzureML)
Requirement already satisfied: pytz>=2011k in /usr/local/lib/python3.6/site-packages (from pandas->AzureML)
 ---> d6db6582fd2c
Removing intermediate container 7f2c89f441f8
Step 7/10 : RUN pip install lxml
 ---> Running in ba25c3ea4c59
Requirement already satisfied: lxml in /usr/local/lib/python3.6/site-packages
 ---> 23f0f5b135b1
Removing intermediate container ba25c3ea4c59
Step 8/10 : RUN pip install Mechanicalsoup
 ---> Running in 189463b4be33
Requirement already satisfied: Mechanicalsoup in /usr/local/lib/python3.6/site-packages
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.6/site-packages (from Mechanicalsoup)
Requirement already satisfied: requests>=2.0 in /usr/local/lib/python3.6/site-packages (from Mechanicalsoup)
Requirement already satisfied: six>=1.4 in /usr/local/lib/python3.6/site-packages (from Mechanicalsoup)
Requirement already satisfied: idna<2.7,>=2.5 in /usr/local/lib/python3.6/site-packages (from requests>=2.0->Mechanicalsoup)
Requirement already satisfied: urllib3<1.23,>=1.21.1 in /usr/local/lib/python3.6/site-packages (from requests>=2.0->Mechanicalsoup)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /usr/local/lib/python3.6/site-packages (from requests>=2.0->Mechanicalsoup)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/site-packages (from requests>=2.0->Mechanicalsoup)
 ---> 32f42217629a
```

Pushed the image on docker hub with tag image2

```
$ docker push akil06/finalproject:image2
The push refers to a repository [docker.io/akil06/finalproject]
f857affcdff2: Pushed
0a55c5845b52: Pushed
38ee8ce09cf4: Pushed
c47e05302988: Pushed
81e4e75a7293: Pushed
c6df71576427: Pushed
c9efb80fc5df: Pushed
c25ee13aaad4: Pushed
db407db37063: Pushed
74b950f60154: Pushed
804792b64572: Pushed
9cfcfc59fd8e: Pushed
8eb56c64a580: Pushed
ac2e71620793: Pushed
20c9fd97b2e7: Pushed
5c1f0257eb15: Pushed
2d32b2aa3df2: Pushed
edd76317b08d: Pushed
2548e7db2a94: Mounted from library/python
325b9d6f2920: Mounted from library/python
815acdffadff: Mounted from library/python
07108d083e01: Mounted from library/python
5616a6292c16: Mounted from library/python
f3ed6cb59ab0: Mounted from library/python
654f45ecb7e3: Mounted from library/python
2c40c66f7667: Mounted from library/python
image2: digest: sha256:6d24a00f3643d2f245a8366541fbaf9c78799e77d110d13a301c32ef31f15e9e size: 6797
```

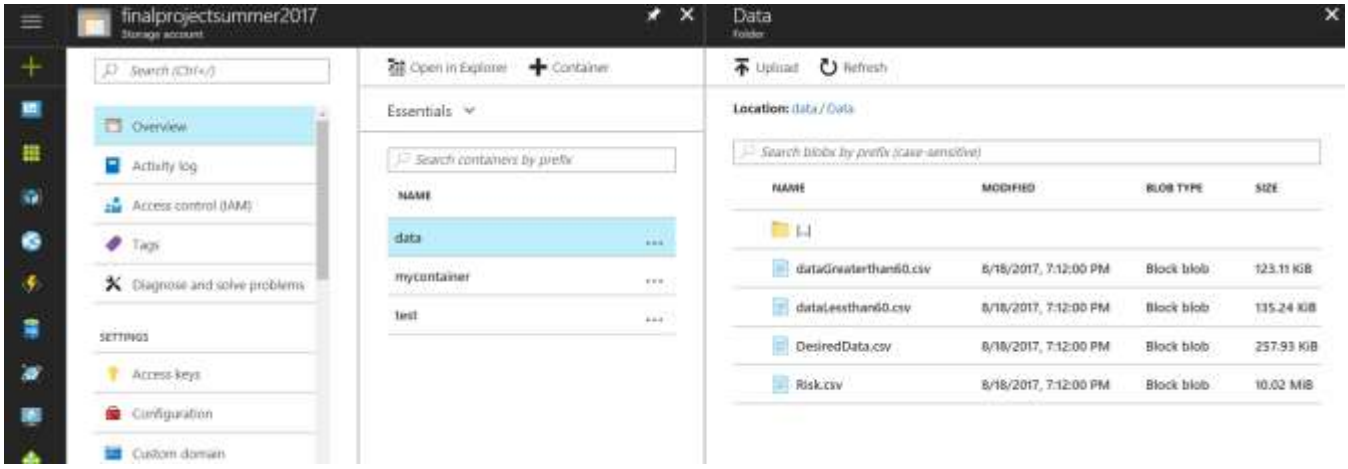
To pull the image you can simply run the command docker pull **akil06/finalproject:image2**

Once, both the images are up and running we can see the files being transferred to the **AZURE BLOB.**

The screenshot shows presence of file on azure blob from where we can directly use the data into Azure Machine learning Studio.

Image showing a folder being created on the Azure blob.

The next picture shows the csv files that got transferred form local to a foreign student. m



PART 3. Building and Evaluating Model

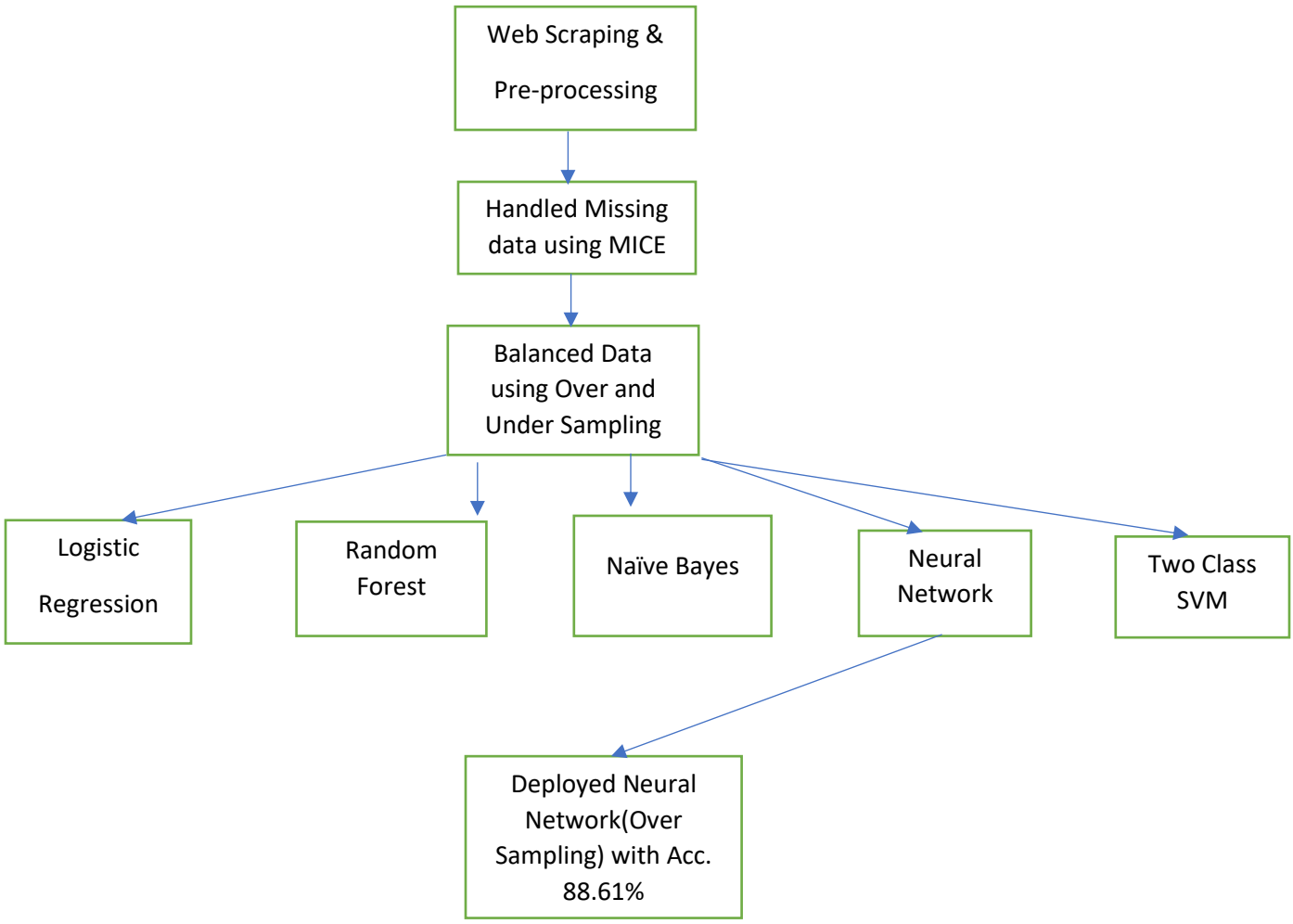
Breast Cancer Risk Assessment

``

What is Breast Cancer Risk Assessment?

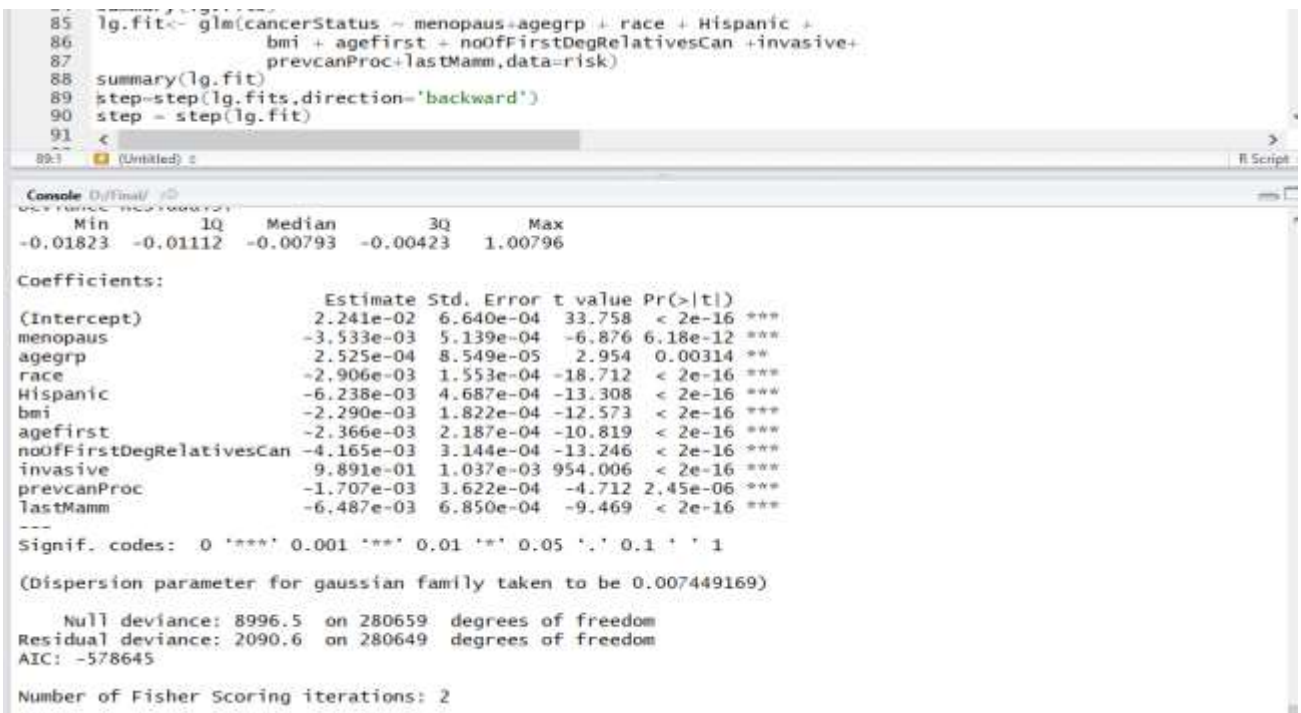
This model is built upon the Breast Cancer Surveillance Consortium (BCSC Dataset).

WORKFLOW:



- Handing Missing values using MICE: As mentioned above in the Pre processing part due to lot of missing values we implemented MICE to fill up the missing values.

- Feature Selection: Using Stepwise regression we selected the features with the highest significance(***). Below is the screenshot which shows the attributes we have selected based on the significance.



```
lg.fit<- glm(cancerStatus ~ menopaus+agegrp + race + Hispanic +
                bmi + agefirst + noOfFirstDegRelativesCan +invasive+
                prevcanProc+lastMamm,data=risk)
summary(lg.fit)
step=step(lg.fits,direction='backward')
step - step(lg.fit)
```

```
     Min       1Q   Median       3Q      Max
-0.01823  -0.01112  -0.00793  -0.00423   1.00796

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)             2.241e-02  6.640e-04  33.758  < 2e-16 ***
menopaus               -3.533e-03  5.139e-04  -6.876 6.18e-12 ***
agegrp                  2.525e-04  8.549e-05   2.954  0.00314 **
race                   -2.906e-03  1.553e-04 -18.712  < 2e-16 ***
Hispanic               -6.238e-03  4.687e-04 -13.308  < 2e-16 ***
bmi                    -2.290e-03  1.822e-04 -12.573  < 2e-16 ***
agefirst               -2.366e-03  2.187e-04 -10.819  < 2e-16 ***
noOfFirstDegRelativesCan -4.165e-03  3.144e-04 -13.246  < 2e-16 ***
invasive                9.891e-01  1.037e-03 954.006  < 2e-16 ***
prevcanProc            -1.707e-03  3.622e-04  -4.712 2.45e-06 ***
lastMamm               -6.487e-03  6.850e-04  -9.469  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.007449169)

    Null deviance: 8996.5  on 280659  degrees of freedom
Residual deviance: 2090.6  on 280649  degrees of freedom
AIC: -578645

Number of Fisher Scoring iterations: 2
```

- Balancing the data: After analyzing at the attribute that we are classifying we see that the attribute is highly imbalanced and we need to balance it for better results.

We used two methods for balancing 1. Over Sampling and 2. Under Sampling.

Below is the screenshot for under sampling.

```
body(predict.lm)
#########Under Sampling ########


#length(lg.fit$coefficients) > lg.fit$rank
(table(risk$cancerStatus))
set.seed(1234)
under_Sample<-ovun.sample(cancerStatus~.,data=risk,method = "under")$data
table(under_Sample$cancerStatus)
table(risk$cancerStatus)

trainData_US=chopFn(under_Sample,0.75)
testData_US=chopFn(under_Sample,0.25)

write.csv(under_Sample, file = "under_Sample.csv")

#########Under Sampling ########
```

Here is the screenshot for Over Sampling .

```
########Over Sampling##############
set.seed(1234)
over_Sample<-ovun.sample(cancerStatus~.,data=risk,method = "over")$data
table(over_Sample$cancerStatus)
head(over_Sample)
write.csv(over_Sample, file = "over_Sample.csv")

trainData_OS=chopFn(over_Sample,0.75)
testData_OS=chopFn(over_Sample,0.25)

head(trainData_OS)
#######Logistic Over Sampling############
```

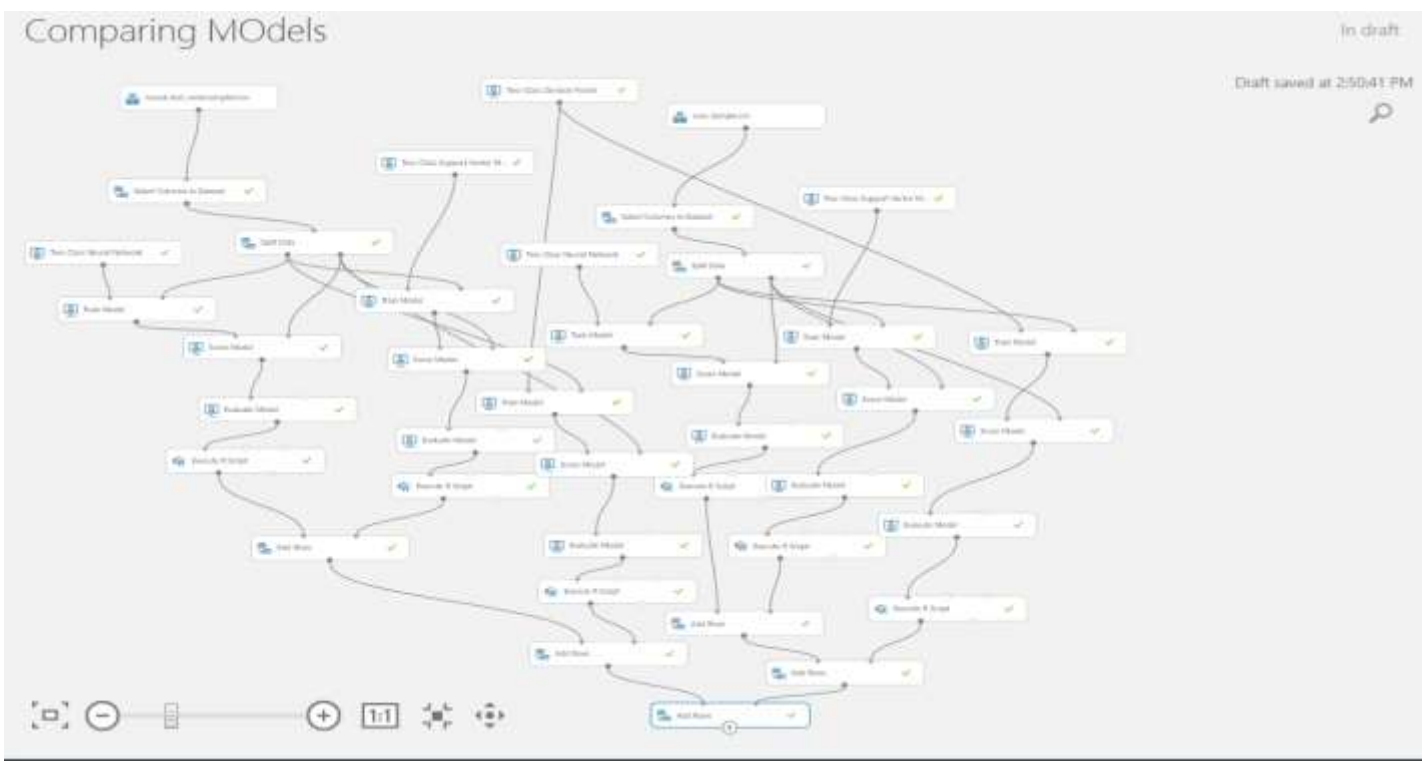
- We implemented RandomForest, Logistic Regression, Naïve Bayes in R and here is the comparison amongst them

```
> model_list <- list(RF_os=rf_OS,RF_US=rf.US,
+                    NB_OS=nb.OS,NB_US=nb.US, Logistic_US=lg.US,LG_OS=lg.fit_OS)
> cm_list <- list(RF_US=CM_rf.US, RF_OS=CM_RF_OS, LR_US=CM_lg.US,LR_OS=CM_LR_OS,NB_US=CM_nb.US,NB_OS=CM_nb.OS)
> cm_list_results <- sapply(cm_list, function(x) x$byClass)
> cm_results_max <- apply(cm_list_results, 1, which.is.max)
> output_report <- data.frame(metric=names(cm_results_max),
+                             best_model=colnames(cm_list_results)[cm_results_max],
+                             value=mapply(function(x,y) {cm_list_results[x,y]},
+                                          names(cm_results_max),
+                                          cm_results_max))
> rownames(output_report) <- NULL
> output_report
                metric best_model     value
1          Sensitivity      LR_OS 1.0000000
2          Specificity      RF_OS 0.9990733
3       Pos Pred Value      RF_OS 0.9992782
4       Neg Pred Value      RF_US 1.0000000
5            Precision      RF_OS 0.9992782
6               Recall      NB_US 1.0000000
7                   F1      RF_US 0.9011240
8           Prevalence      RF_OS 0.6107005
9       Detection Rate      NB_US 0.5092593
10 Detection Prevalence      NB_US 0.6371662
11    Balanced Accuracy      RF_OS 0.9084687
>
```

The above method returns the method which is the best for each testing parameter. Looking at it we realize that Random Forest Over Sampled Model is the best amongst the all.

Now we will compare the Random Forest Over Sampled with Neural network and SVM in Azure ML.

- Comparing RandomForest, Neural networks and SVM in Azure ML



Here's the comparison matrix to compare the test parameters of different models:

Comparing MOdels › Add Rows › Results dataset

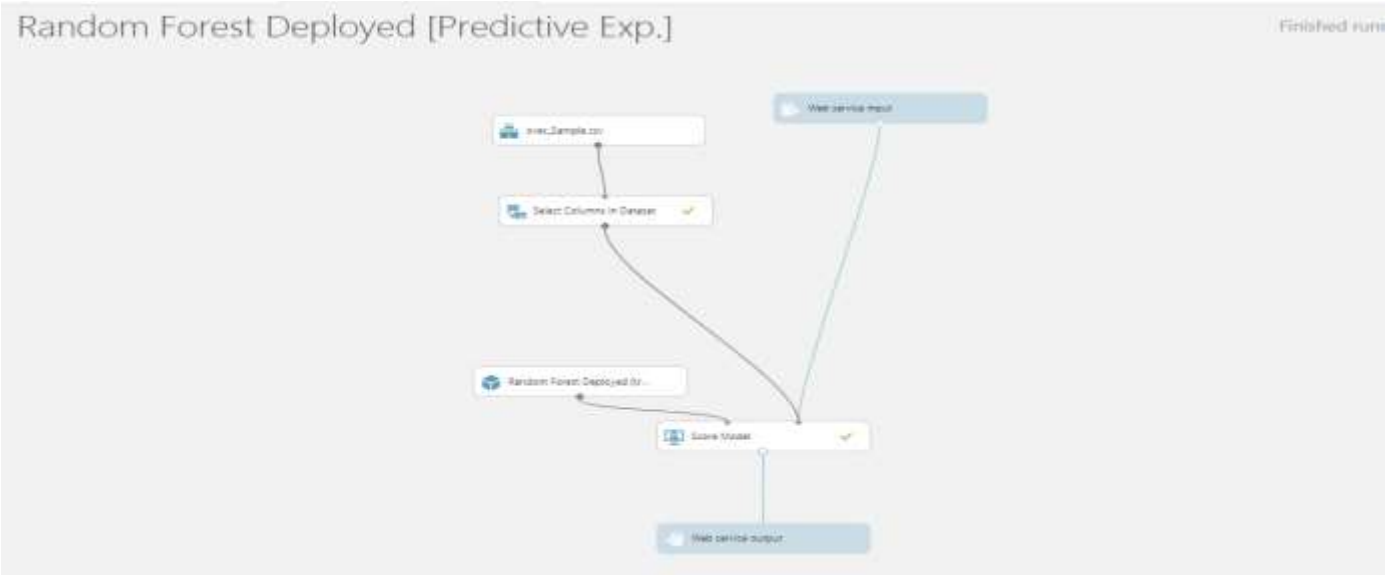
rows 6 columns 8

view as

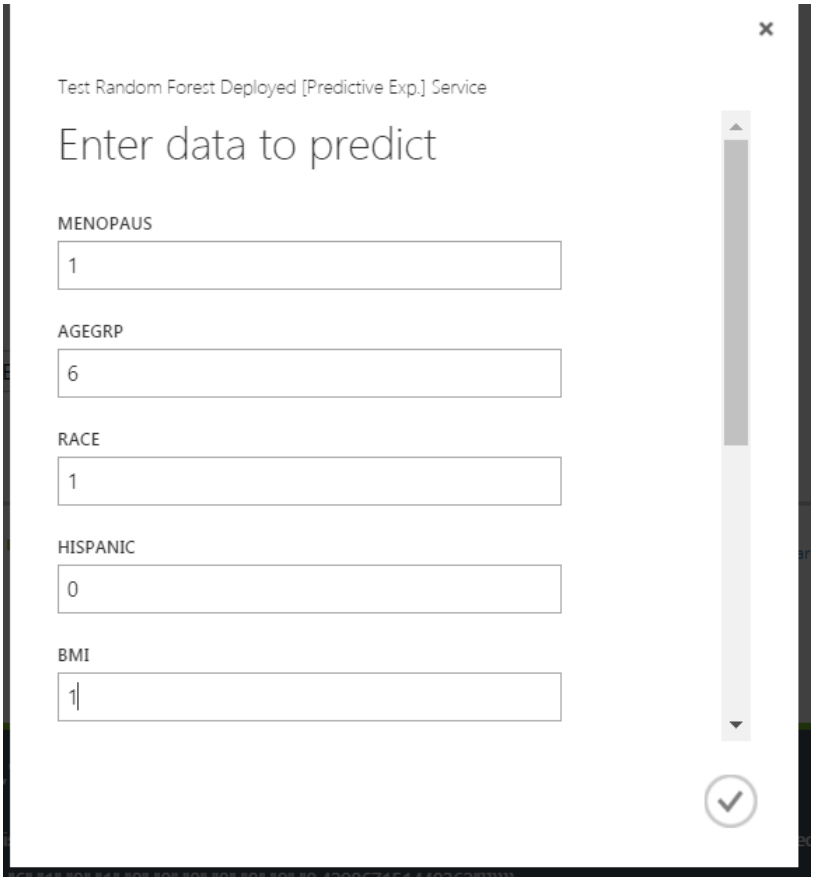
Accuracy	Precision	Recall	F-Score	AUC	Average Log Loss	Training Log Loss	Algorithm
0.879177	0.984848	0.772496	0.865842	0.927579	0.29415	57.560391	NeuralNetwork
0.87832	1	0.758913	0.862934	0.926381	0.2864	58.678589	SVM
0.85347	0.888476	0.811545	0.84827	0.905399	1.304494	-88.210827	RandomForest_UnderSampled
0.886102	0.990527	0.778941	0.872083	0.947678	0.264275	61.872925	NeuralNetwork_OVerSampled
0.886227	1	0.771742	0.871168	0.925324	0.27735	59.986623	SVM_OverSampled
0.89439	0.949564	0.832328	0.88709	0.964856	0.20807	69.981694	RandomForest_OverSampled

Looking at the comparison matrix we can see that RandomForest OverSampled is giving the best results. So, we deploy Random Forest oversampled and use the API of it in our application.

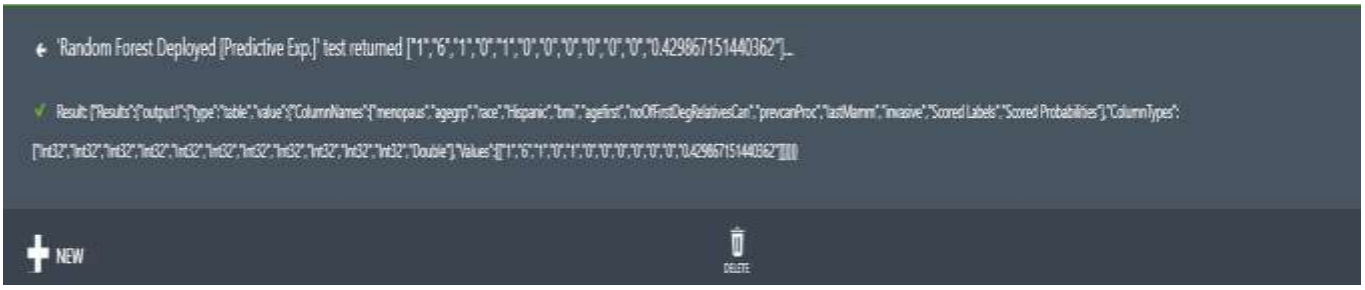
Here's the experiment that shows how we deployed it:



Testing the Model on AzureML:



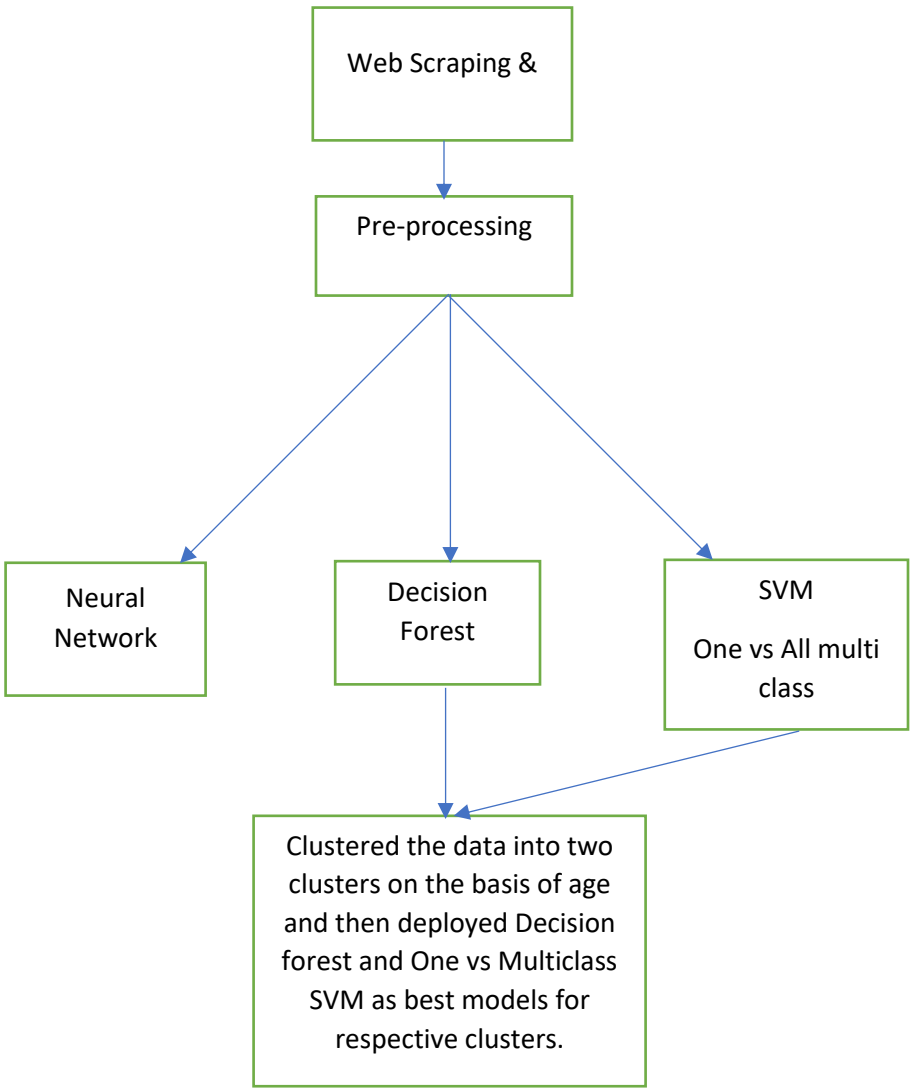
Output:



Classification of Breast cancer subtype:

There are various types of Breast Cancer and it is difficult to classify what type of Cancer it is unless we perform rigorous tests on it. We have come up with a model that can classify the lump into what type of cancer it is based on several input parameters.

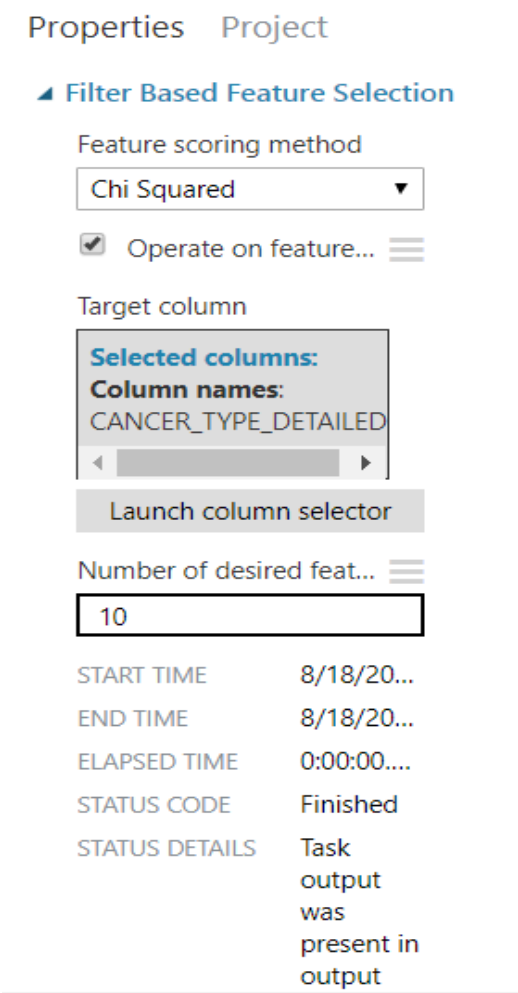
WORKFLOW:

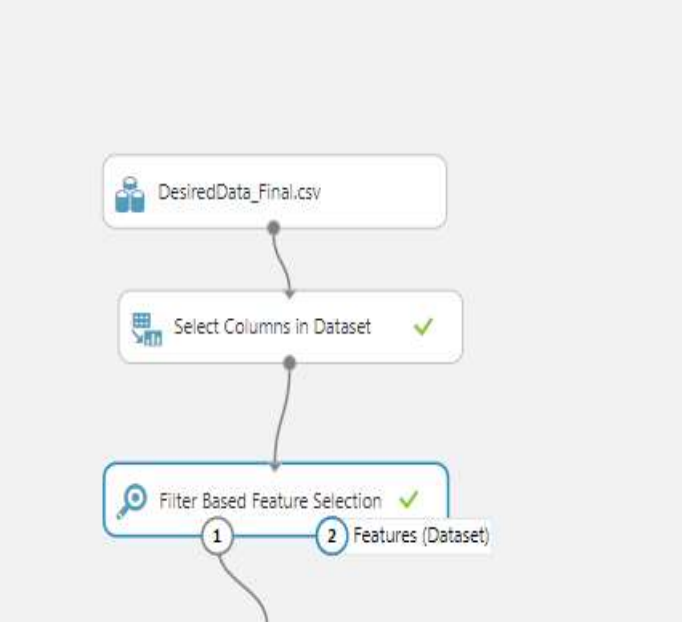


3.1.3 FEATURE SELECTION:

Before proceeding with our models, we have done feature selection using Chi Squared method.

The best features that add to the predictive power of the model will be retained and irrelevant features removed from the model.





We selected 10 features using the Filter Based Feature Selection block. The statistical method used for scoring method was Chi Squared. The features selected are:

MultiClass Classification › Filter Based Feature Selection › Filtered dataset

rows 1171 columns 11

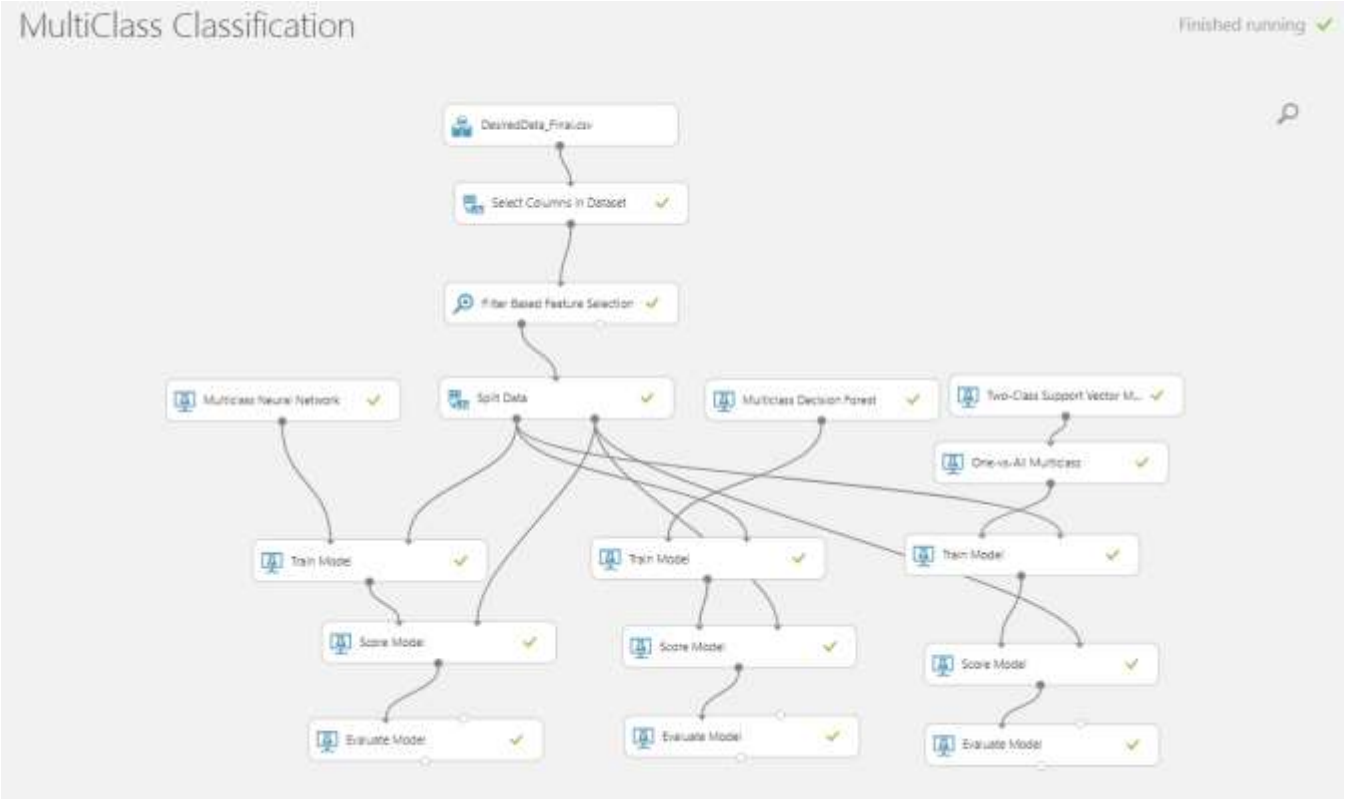
CANCER_TYPE_DETAILED	NPI	COHORT	TUMOR_SIZE	AGE_AT_DIAGNOSIS	TUMOR_STAGE	ER_STATUS	HORMONE_THERAPY	HER2_STATUS	OS_STATUS	PR_STATUS
Breast Invasive Ductal Carcinoma	2.024	1	12	66.11	1	-	1	0	1	0
Breast Invasive Ductal Carcinoma	4.038	1	19	38.86	1	+	1	0	1	0
Breast Invasive Ductal Carcinoma	6.046	1	23	39.3	2	+	1	0	0	1
Invasive Breast Carcinoma	5.054	1	27	45.9	2	+	1	0	1	0
Breast Mixed Ductal and Lobular Carcinoma	5.104	1	52	74.46	3	+	1	0	0	1
Invasive Breast Carcinoma	4.024	1	12	51.45	2	+	1	0	1	0
Breast Invasive Ductal Carcinoma	3.036	3	18	47.61	1	+	1	0	1	0
Breast Invasive Ductal Carcinoma	4.04	1	20	67.88	1	+	1	0	0	0
Breast Invasive Ductal Carcinoma	6.074	1	37	61.89	2	+	1	0	0	1
Breast Invasive Ductal Carcinoma	5.07	1	35	59.07	2	+	1	0	0	1
Breast Invasive Ductal Carcinoma	3.03	2	15	48.07	1	+	0	0	1	1
Breast Invasive Ductal Carcinoma	3.024	1	12	46.66	1	+	1	0	1	1
Breast Invasive Ductal Carcinoma	4.05	1	25	48.47	2	-	0	0	1	0

Machine Learning Algorithms and Output:

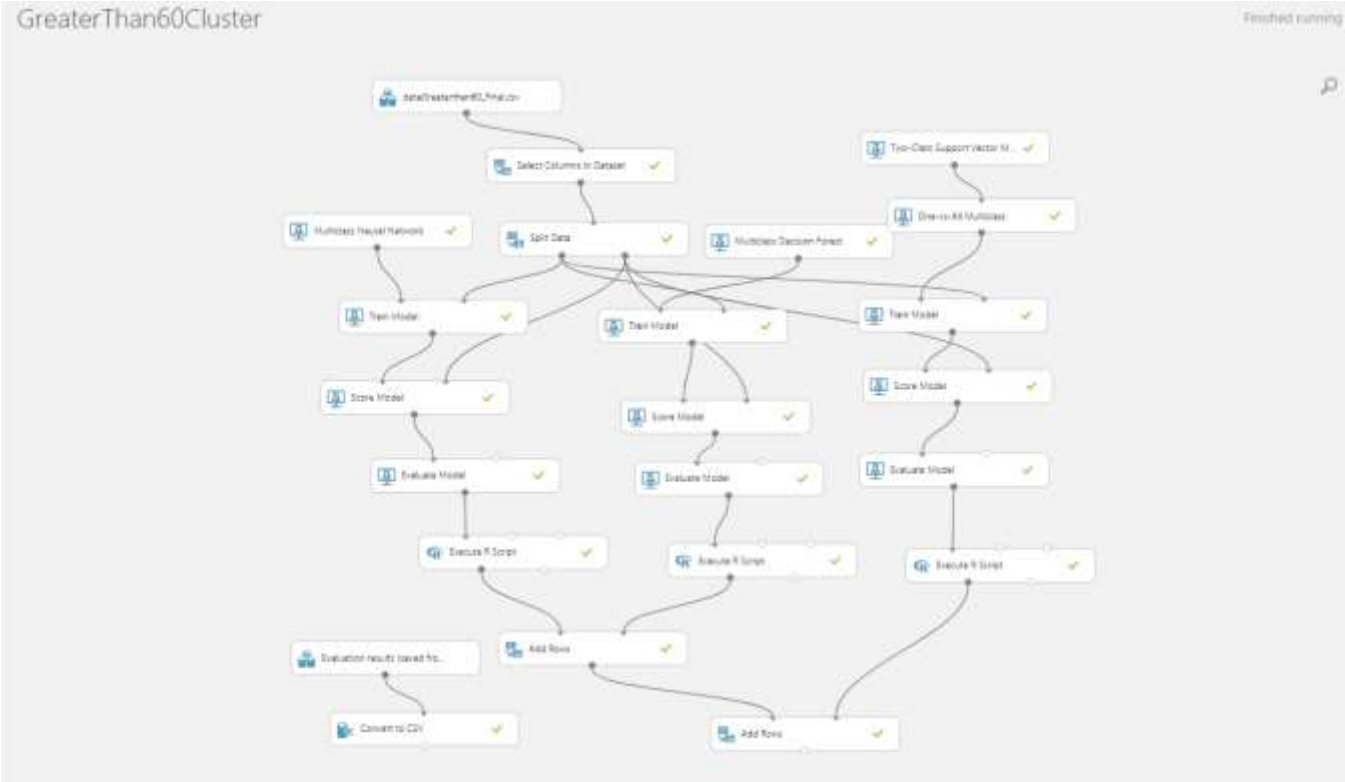
To classify the types of the Breast Cancer with the data of results of patients vitals we found that the age at diagnosis of the patient is normally distributed . So we clustered the dataset into two clusters (age group <= 62 and age group >62) and then build the two models using this clustered data. The algorithms used are:

1. Multi Class Neural Networks
2. Multi Class Decision Forest
3. 1 vs all Multi class SVM

Following is the screenshot attached that shows the three algorithms implemented on the entire dataset:



Following is the screenshot for the cluster with age greater than 62 cluster :



Multiclass Neural networks

GreaterThan60Cluster › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.731884
Average accuracy	0.865942
Micro-averaged precision	0.731884
Macro-averaged precision	NaN
Micro-averaged recall	0.731884
Macro-averaged recall	0.251253

Decision Forest

GreaterThan60Cluster › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.76087
Average accuracy	0.880435
Micro-averaged precision	0.76087
Macro-averaged precision	NaN
Micro-averaged recall	0.76087
Macro-averaged recall	0.260777

1 vs all Multiclass SVM

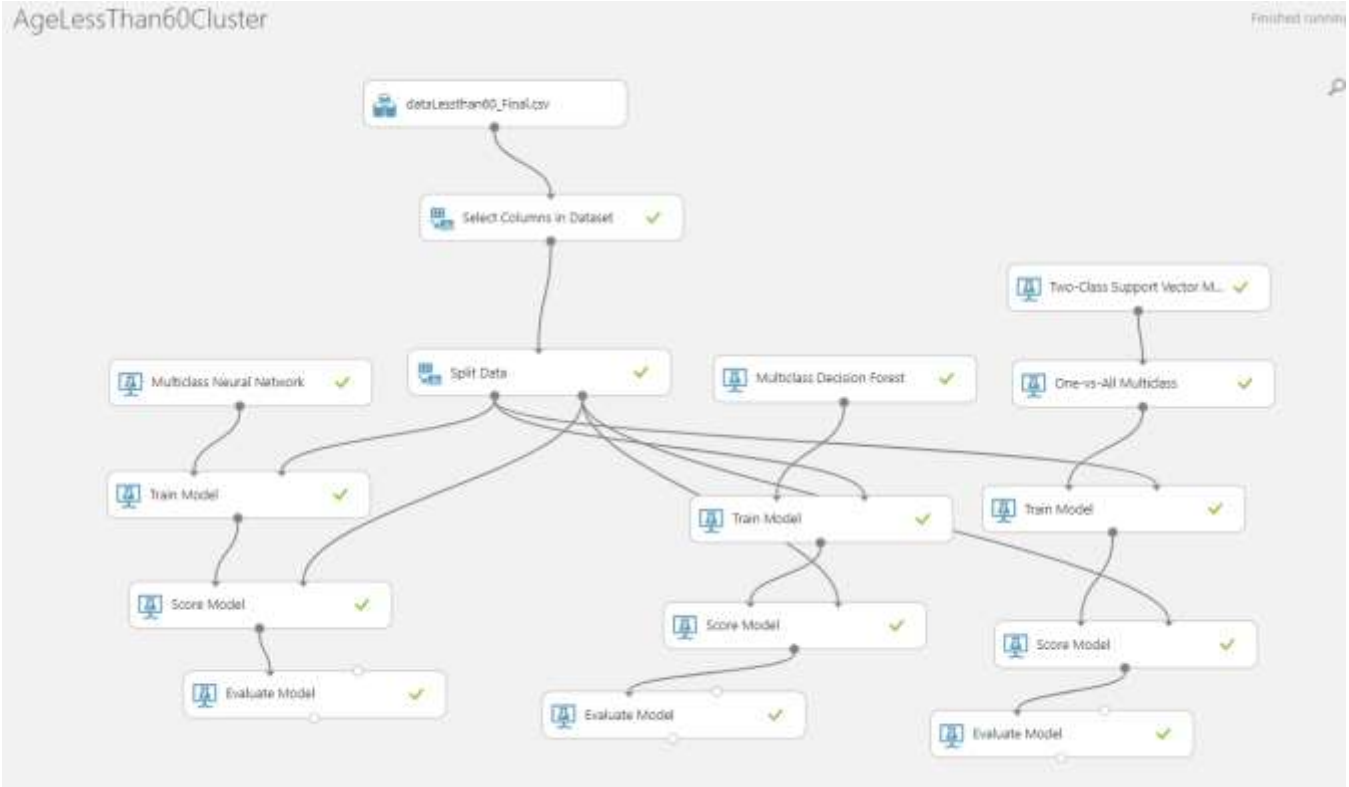
GreaterThan60Cluster › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.76087
Average accuracy	0.880435
Micro-averaged precision	0.76087
Macro-averaged precision	NaN
Micro-averaged recall	0.76087
Macro-averaged recall	0.25

Confusion Matrix

From these three models Decision Forest and SVM models gave similar results but we finally deployed with Decision Forest though either one of the above mentioned algorithms can be used.



Neural network

AgeLessThan60Cluster › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.754839
Average accuracy	0.877419
Micro-averaged precision	0.754839
Macro-averaged precision	NaN
Micro-averaged recall	0.754839
Macro-averaged recall	0.302738

Confusion Matrix

Decision forest

AgeLessThan60Cluster › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.774194
Average accuracy	0.887097
Micro-averaged precision	0.774194
Macro-averaged precision	0.198675
Micro-averaged recall	0.774194
Macro-averaged recall	0.241935

Confusion Matrix

SVM

AgeLessThan60Cluster › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.8
Average accuracy	0.9
Micro-averaged precision	0.8
Macro-averaged precision	NaN
Micro-averaged recall	0.8
Macro-averaged recall	0.25

Confusion Matrix

The accuracy of the SVM gave better accuracy than the other models so we deployed SVM model for this category of cluster.

Model Performance

The model gave good performance in classifying **Invasive Ductal Carcinoma** and **Invasive Lobular Carcinoma** which is widely occurring type of cancer among women even over the years.

But it was not in giving similar results for other two types of cancer. We tried to balance the data using SMOTE and ROSE but the accuracy and precision was getting low when using sampled data.

The app for dealing the two above mentioned use cases is developed in

- Python Flask
- HTML
- CSS
- Java Scripts

And deployed as a Web App in IBM Bluemix

http://breastcancerassesment.mybluemix.net/

The sample screens for the App looks like :

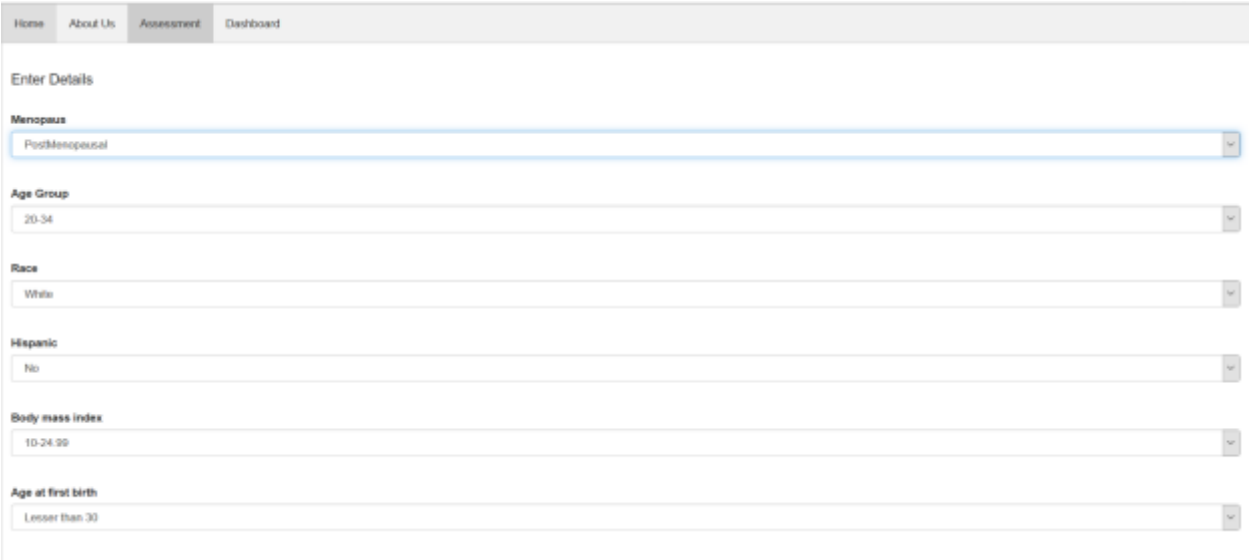
One of the tabs of our application that describes about our app

POWER BI DASHBOARD INTEGRATED WITH THE APP

TABLEAU DASBOARD INTEGRATED WITH THE APP

USE CASE1:

Estimating risk of getting Breast Cancer



OUTPUT:

USE CASE 2:

Classifying the tumor into types of Cancer:

Fill the form to classify the type of breast cancer

Nottingham Prognostic Index

-7

Cellularity

Low

ER Status

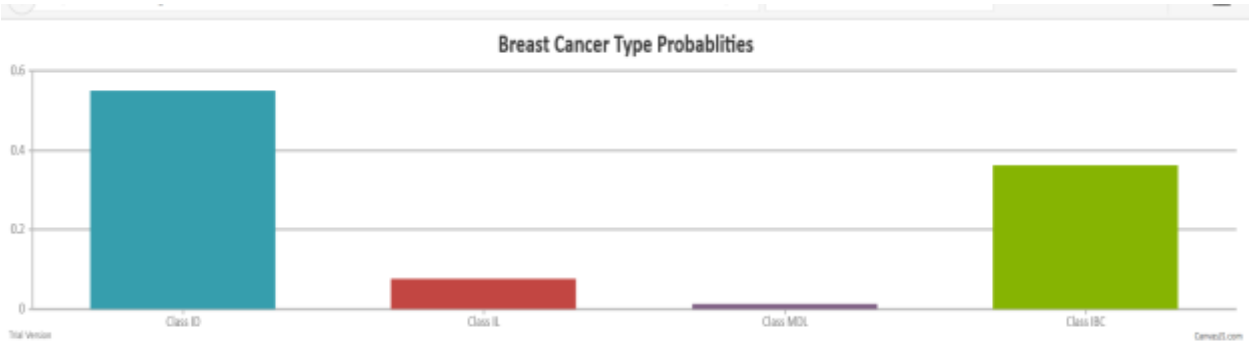
Positive

HER Status

Positive

PR Status

Positive



Click here to go back to main page