

IBM ASSIGNMENT 4

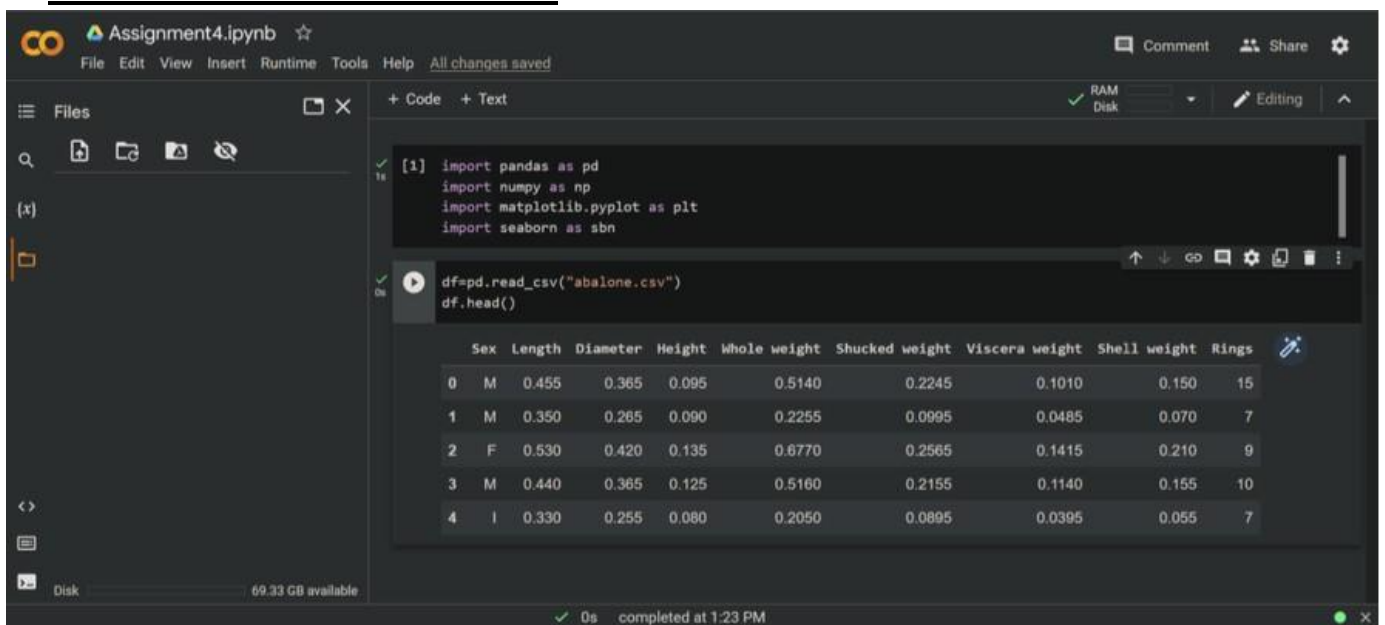
Name: V.Siva chandran

Register No: 961819104081

CHALLENGE:

Abalone Age Prediction

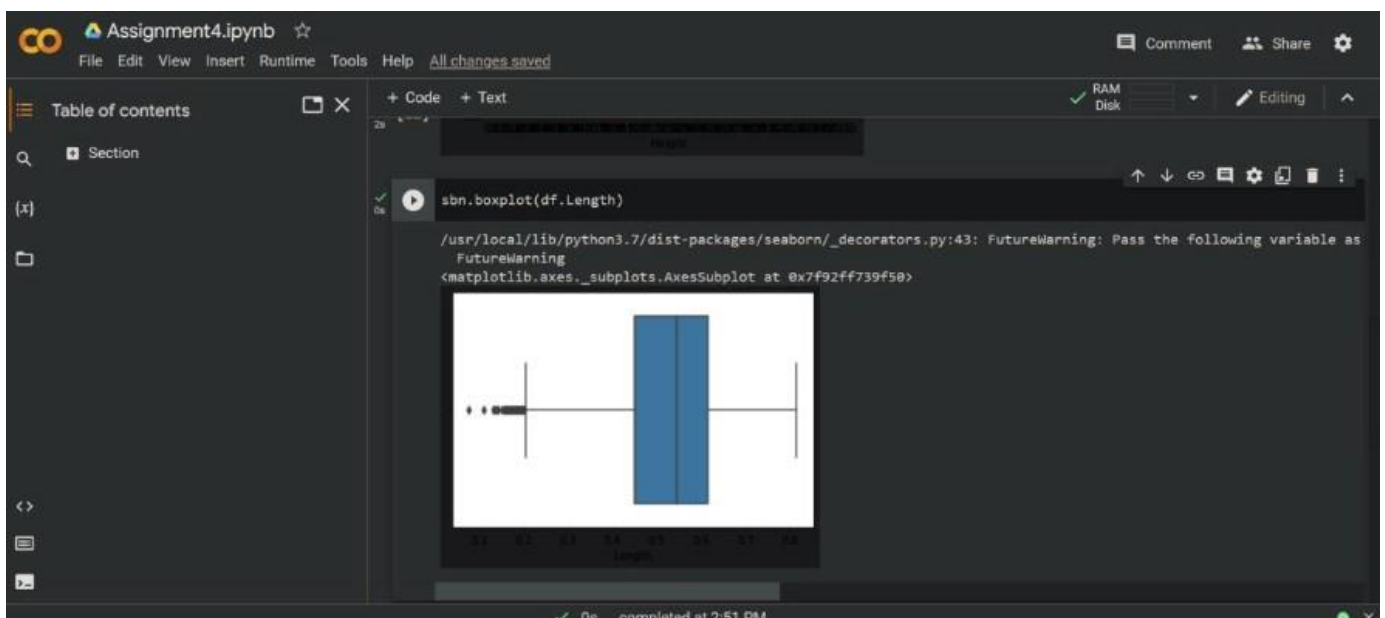
LOADING THE DATASET:

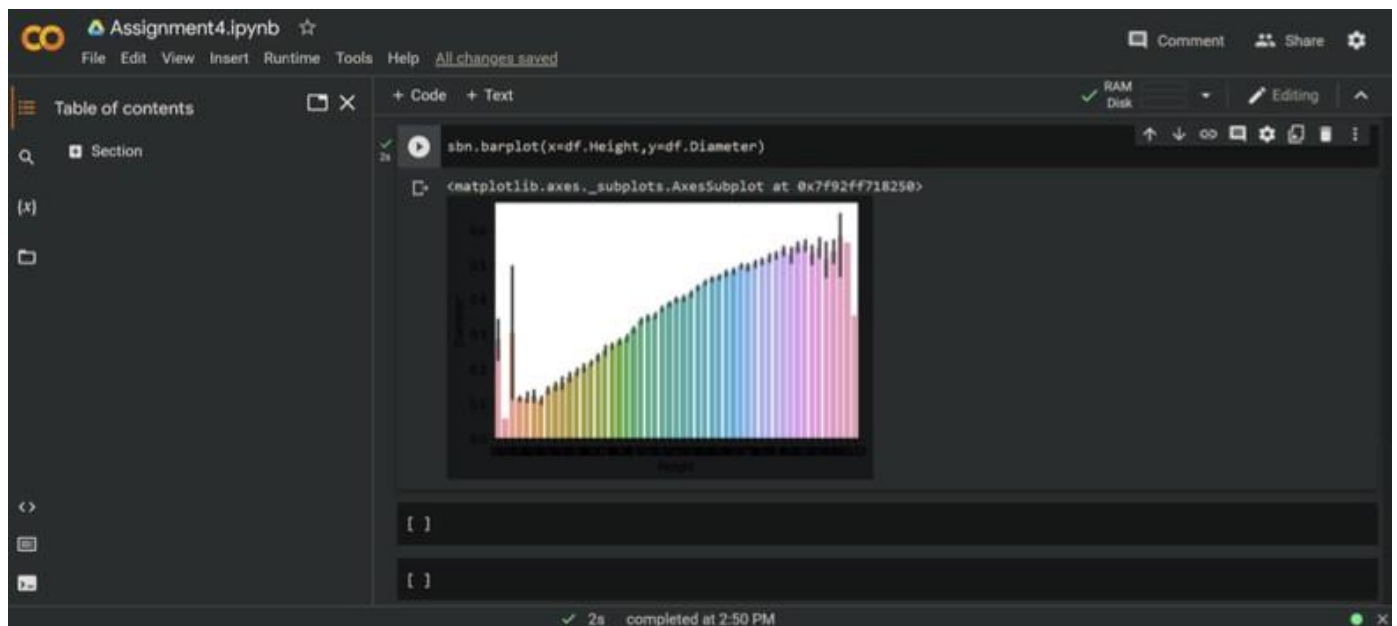


VISUALIZATIONS:

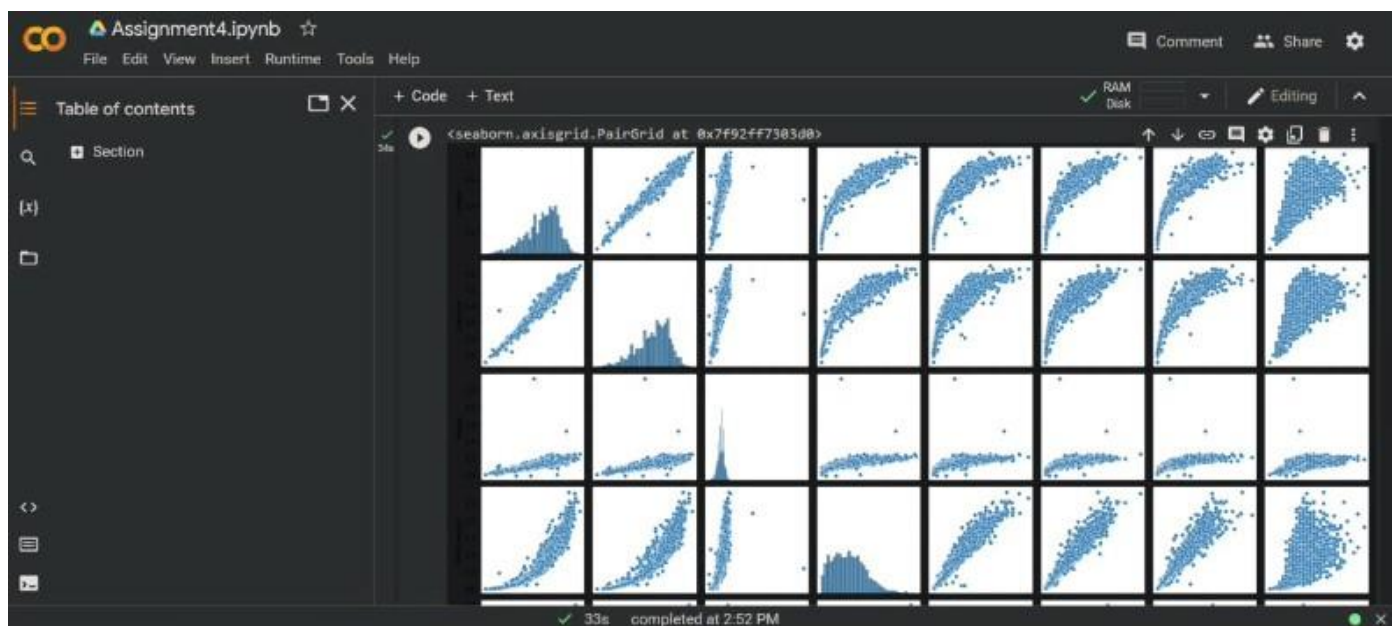
Univariate Analysis

Bi-Variate Analysis





Multi-Variate Analysis



Perform descriptive Analysis on datasets

This screenshot shows a Jupyter Notebook titled 'Assignment4.ipynb'. The left sidebar contains a 'Table of contents' and a search bar. The main area displays three code cells:

- Cell [15]: `df['Length'].mode()` returns a Series with values 0.550 and 0.625, dtype: float64.
- Cell [17]: `df['Height'].mean()` returns the value 0.13951639932966242.
- Cell [20]: `df.count()` returns a Series showing counts for various columns: Sex (4177), Length (4177), Diameter (4177), Height (4177), Whole weight (4177), Shucked weight (4177), Viscera weight (4177), Shell weight (4177), and Rings (4177), dtype: int64.

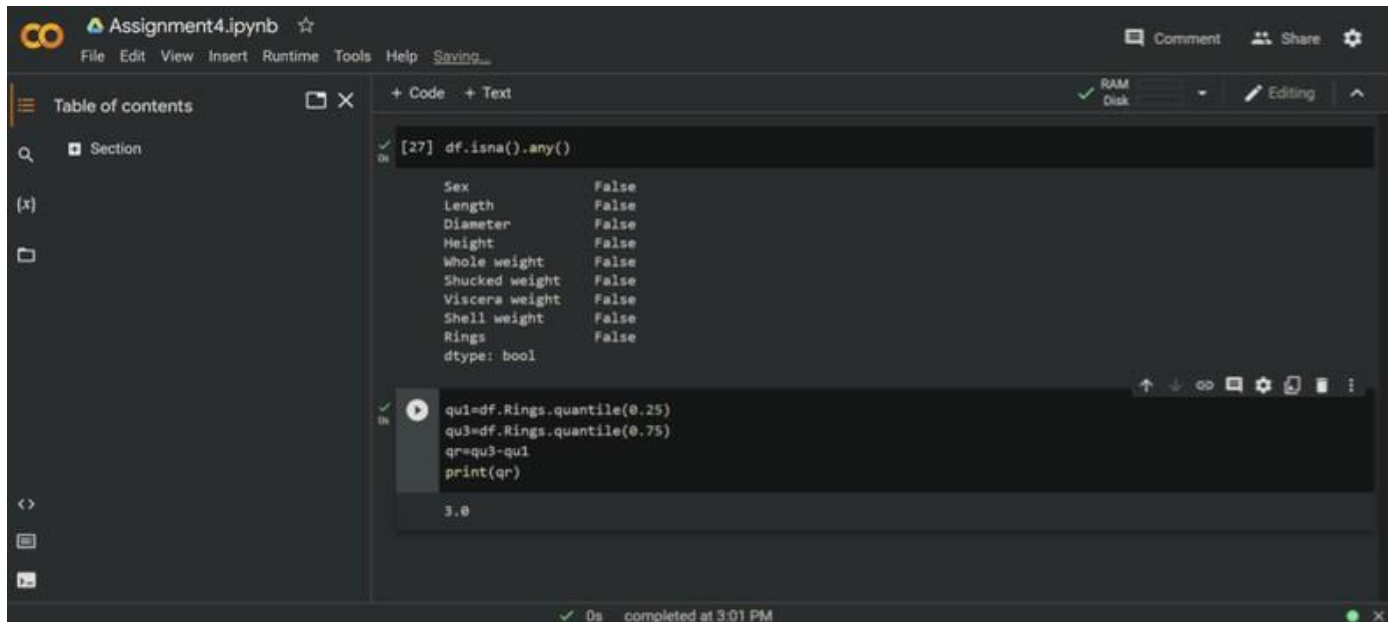
The status bar at the bottom indicates '0s completed at 2:56 PM'.

This screenshot shows the same Jupyter Notebook at a later stage. The left sidebar is identical. The main area displays three more code cells:

- Cell [23]: `df['Shell weight'].sum()` returns the value 997.5964999999999.
- Cell [24]: `df['Rings'].product()` returns the value 0.
- Cell [25]: `df['Whole weight'].max()` returns the value 2.8255.

Below the code cells, there is a toolbar with icons for undo, redo, copy, paste, and other actions, followed by an empty input field and a list icon. The status bar at the bottom indicates '0s completed at 2:59 PM'.

Checking for missing values and deal with them , Finding the outliers and replace them outliers



The screenshot shows a Jupyter Notebook interface with the file 'Assignment4.ipynb'. The left sidebar contains a 'Table of contents' and a search bar. The main area displays two code cells. The first cell, labeled '[27]', contains the code `df.isna().any()`, which has been executed to show the presence of missing values across various columns. The second cell contains code to calculate the interquartile range (IQR) for the 'Rings' variable: `qu1=df.Rings.quantile(0.25)`, `qu3=df.Rings.quantile(0.75)`, `qr=qu3-qu1`, and `print(qr)`. The output of this cell is the value `3.0`. The status bar at the bottom indicates the notebook is 'completed at 3:01 PM'.

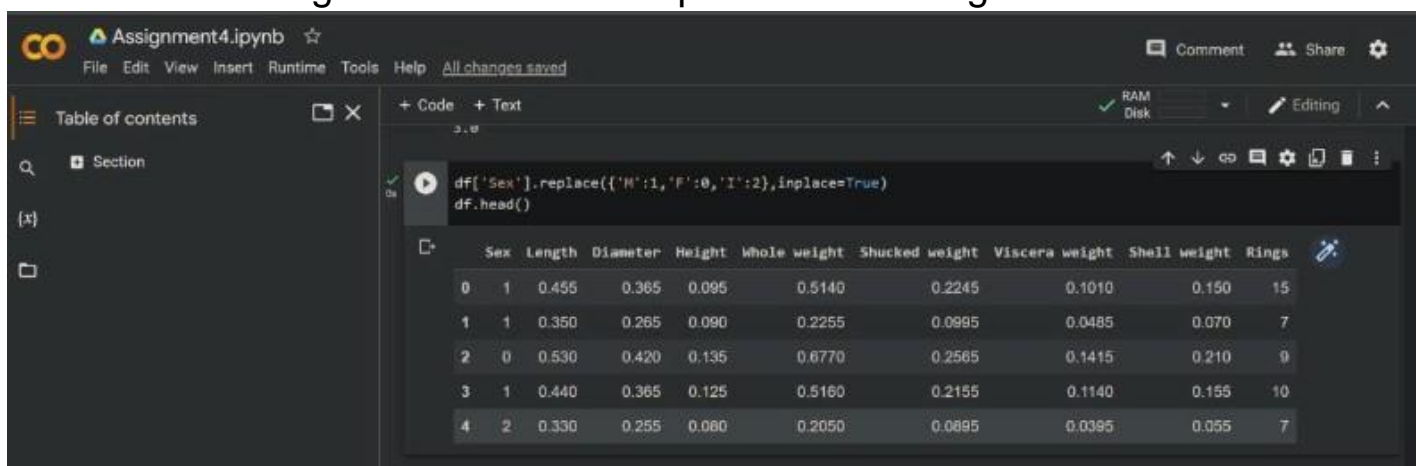
```
[27] df.isna().any()

Sex          False
Length       False
Diameter     False
Height       False
Whole weight False
Shucked weight False
Viscera weight False
Shell weight False
Rings        False
dtype: bool
```

```
qu1=df.Rings.quantile(0.25)
qu3=df.Rings.quantile(0.75)
qr=qu3-qu1
print(qr)

3.0
```

Check for categorical columns and perform encoding



This screenshot shows the next step in the data preprocessing workflow. The code cell contains `df['Sex'].replace({'M':1,'F':0,'I':2},inplace=True)` to convert the categorical 'Sex' variable into numerical values, followed by `df.head()` to preview the data. The output is a preview of the first five rows of the DataFrame. The status bar shows 'All changes saved'.

```
df['Sex'].replace({'M':1,'F':0,'I':2},inplace=True)
df.head()
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	1	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	1	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	2	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

Split the data into dependent and independent variables, Scale the independent variable

The screenshot shows a Jupyter Notebook titled "Assignment4.ipynb". The left sidebar contains a "Table of contents" and a search bar. The main area displays two code cells. The first cell, labeled [33], contains the following code:

```
x=df.iloc[:, :-1].values
y=df.iloc[:, -1].values
```

The second cell, labeled [34], contains the following code:

```
from sklearn.preprocessing import StandardScaler
std=StandardScaler()
x=std.fit_transform(x)
x
```

The output of the second cell is a large NumPy array of standardized features. The status bar at the bottom indicates "completed at 3:06 PM".

Split the data into training and testing

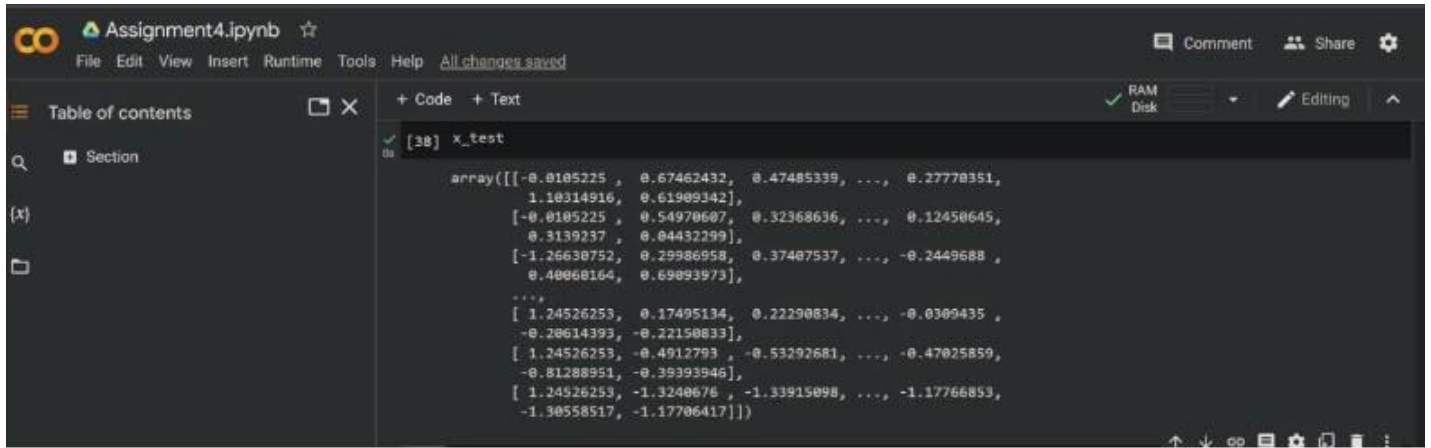
The screenshot shows the same Jupyter Notebook titled "Assignment4.ipynb". The left sidebar is identical. The main area displays two code cells. The first cell, labeled [35], contains the following code:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

The second cell, labeled [36], contains the following code:

```
x_train
```

The output of the second cell is a large NumPy array of training features. The status bar at the bottom indicates "completed at 3:18 PM".



Assignment4.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Table of contents

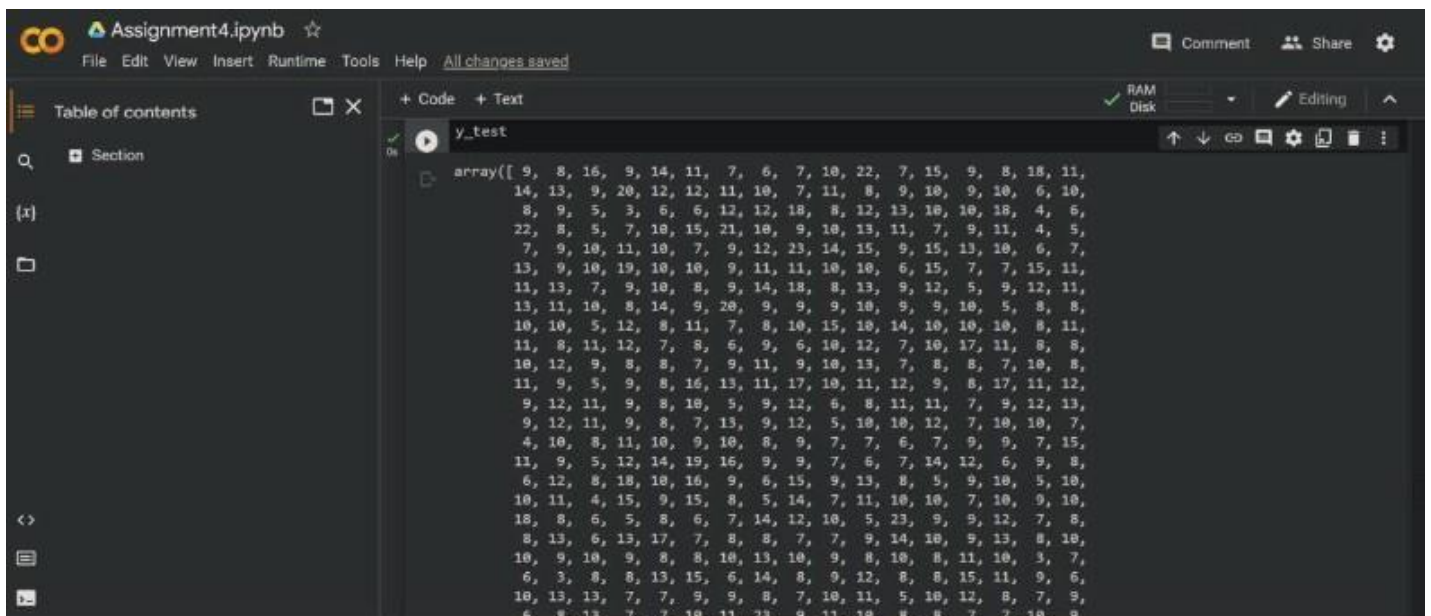
Section

+ Code + Text

RAM Disk

Editing

```
[38] x_test
array([[ -0.0105225,  0.67462432,  0.47485339, ...,  0.27770351,
         1.10314916,  0.61909342],
       [ -0.0105225,  0.54970607,  0.32368636, ...,  0.12450645,
         0.3139237,   0.04432299],
       [ -1.26630752,  0.29986958,  0.37407537, ..., -0.2449688,
         0.40060164,  0.69093973],
       ...,
       [ 1.24526253,  0.17495134,  0.22290834, ..., -0.0309435,
        -0.20614393, -0.22150833],
       [ 1.24526253, -0.4912793, -0.53292681, ..., -0.47025859,
        -0.81288951, -0.39393946],
       [ 1.24526253, -1.3240676, -1.33915098, ..., -1.17766853,
        -1.30558517, -1.17706417]])
```



Assignment4.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Table of contents

Section

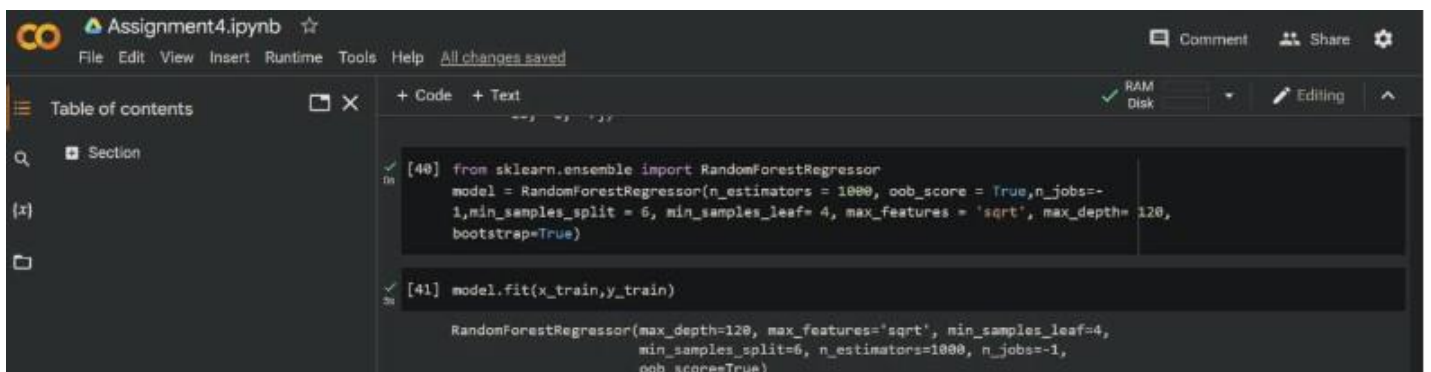
+ Code + Text

RAM Disk

Editing

```
y_test
array([ 9,  8, 16,  9, 14, 11,  7,  6,  7, 10, 22,  7, 15,  9,  8, 18, 11,
        14, 13,  9, 20, 12, 12, 11, 10,  7, 11,  8,  9, 10,  9, 10,  6, 10,
         8,  9,  5,  3,  6,  6, 12, 12, 18,  8, 12, 13, 10, 10, 18,  4,  6,
        22,  8,  5,  7, 10, 15, 21, 10,  9, 10, 13, 11,  7,  9, 11,  4,  5,
         7,  9, 10, 11, 10,  7,  9, 12, 23, 14, 15,  9, 15, 13, 10,  6,  7,
        13,  9, 10, 19, 10, 10,  9, 11, 11, 10, 10,  6, 15,  7,  7, 15, 11,
        11, 13,  7,  9, 10,  8,  9, 14, 18,  8, 13,  9, 12,  5,  9, 12, 11,
        13, 11, 10,  8, 14,  9, 20,  9,  9,  9, 10,  9,  9, 10,  5,  8,  8,
        10, 10,  5, 12,  8, 11,  7,  8, 10, 15, 10, 14, 10, 10, 10,  8, 11,
        11,  8, 11, 12,  7,  8,  6,  9,  6, 10, 12,  7, 10, 17, 11,  8,  8,
        10, 12,  9,  8,  8,  7,  9, 11,  9, 10, 13,  7,  8,  8,  7, 10,  8,
        11,  9,  5,  9,  8, 16, 13, 11, 17, 10, 11, 12,  9,  8, 17, 11, 12,
         9, 12, 11,  9,  8, 10,  5,  9, 12,  6,  8, 11, 11,  7,  9, 12, 13,
         9, 12, 11,  9,  8,  7, 13,  9, 12,  5, 10, 10, 12,  7, 10, 10,  7,
         4, 10,  8, 11, 10,  9, 10,  8,  9,  7,  7,  6,  7,  9,  9,  7, 15,
        11,  9,  5, 12, 14, 19, 16,  9,  9,  7,  6,  7, 14, 12,  6,  9,  8,
         6, 12,  8, 18, 10, 16,  9,  6, 15,  9, 13,  8,  5,  9, 10,  5, 10,
        10, 11,  4, 15,  9, 15,  8,  5, 14,  7, 11, 10, 10,  7, 10,  9, 10,
        18,  8,  6,  5,  8,  6,  7, 14, 12, 10,  5, 23,  9,  9, 12,  7,  8,
         8, 13,  6, 13, 17,  7,  8,  8,  7,  7,  9, 14, 10,  9, 13,  8, 10,
        10,  9, 10,  9,  8,  8, 10, 13, 10,  9,  8, 10,  8, 11, 10,  3,  7,
         6,  3,  8,  8, 13, 15,  6, 14,  8,  9, 12,  8,  8, 15, 11,  9,  6,
        10, 13, 13,  7,  7,  9,  9,  8,  7, 10, 11,  5, 10, 12,  8,  7,  9,
         6,  8, 13,  7,  7, 10, 11, 23,  9, 11, 10,  8,  8,  7,  7, 10,  9,
```

Build the model, Train the Model Test the Model



Assignment4.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Table of contents

Section

+ Code + Text

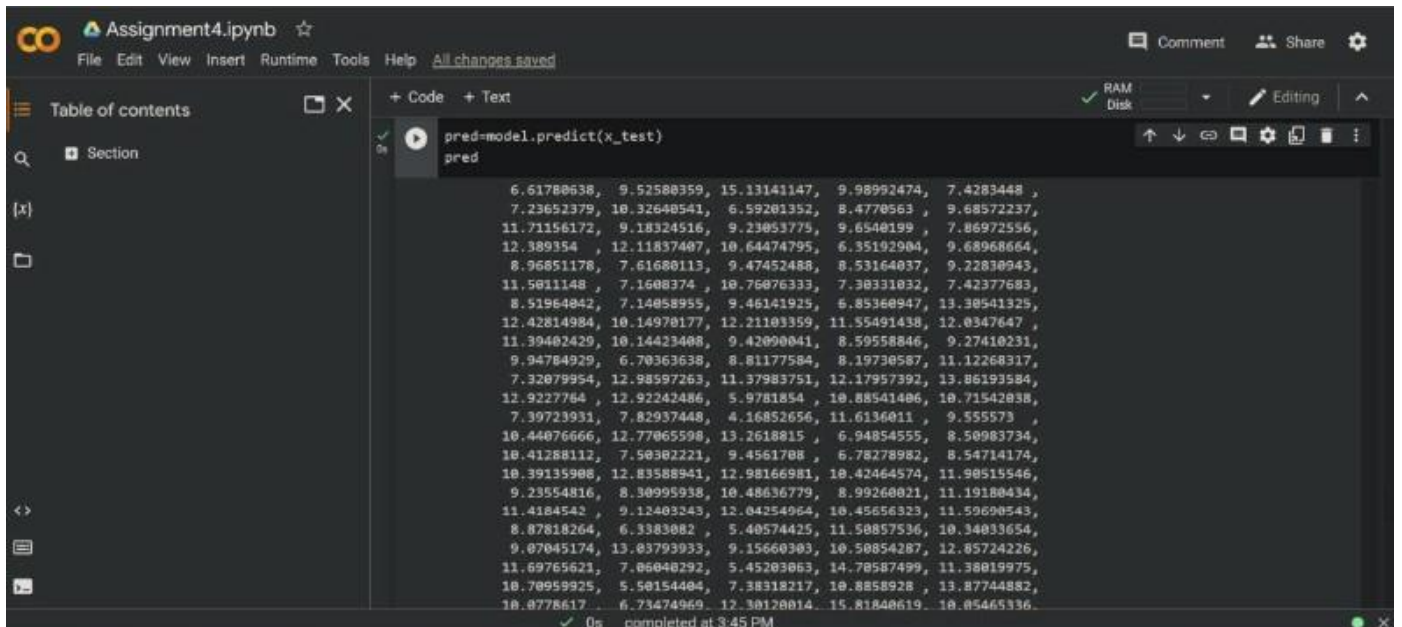
RAM Disk

Editing

```
[40] from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(n_estimators = 1000, oob_score = True, n_jobs=-1, min_samples_split = 6, min_samples_leaf = 4, max_features = 'sqrt', max_depth = 120, bootstrap=True)

[41] model.fit(x_train, y_train)

RandomForestRegressor(max_depth=120, max_features='sqrt', min_samples_leaf=4, min_samples_split=6, n_estimators=1000, n_jobs=-1, oob_score=True)
```

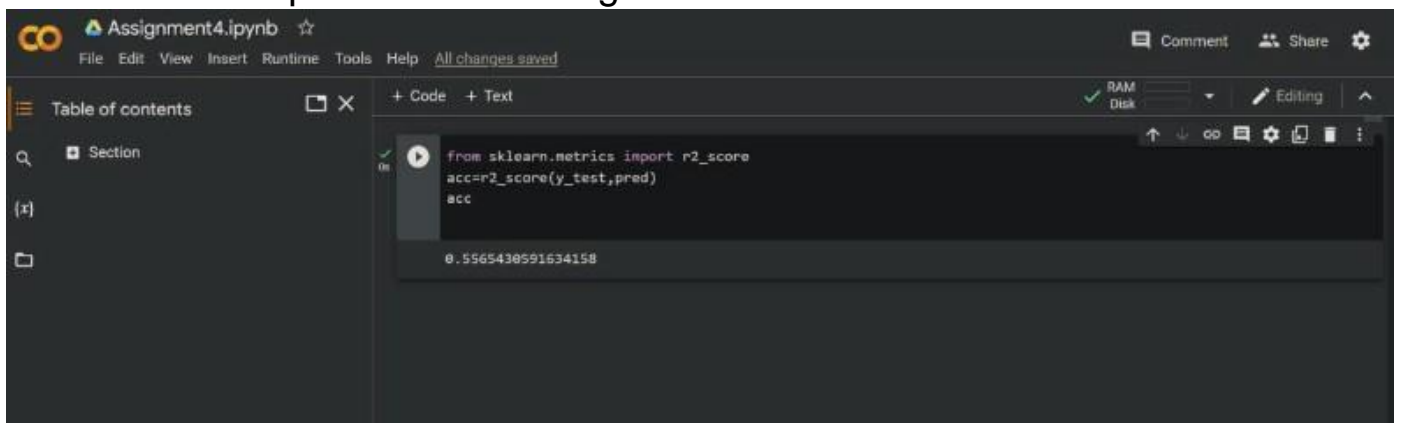
The screenshot shows a Jupyter Notebook titled "Assignment4.ipynb". The left sidebar contains a "Table of contents" and a "Section" dropdown. The main area is in "Code" mode, displaying a single code cell with the following code:

```
pred=model.predict(x_test)
pred
```

The output of the code cell is a large array of 50 numerical values, representing predictions. The values are displayed in a grid-like format, with 10 columns and 5 rows. The first row of values is: 6.61780638, 9.52580359, 15.13141147, 9.98992474, 7.4283448, 7.23652379, 10.32640541, 6.59201352, 8.4770563, 9.68572237, 11.71156172, 9.18324516, 9.23053775, 9.6540199, 7.86972556, 12.389354, 12.11837407, 10.64474795, 6.35192904, 9.68968664, 8.96851178, 7.61680113, 9.47452488, 8.53164037, 9.22830943, 11.5011148, 7.1608374, 10.76076333, 7.30331032, 7.42377683, 8.51964042, 7.14058955, 9.46141925, 8.85360947, 13.30541325, 12.42814984, 10.14970177, 12.21103359, 11.55491438, 12.0347647, 11.39402429, 10.14423408, 9.42090041, 8.59558846, 9.27410231, 9.94784929, 6.70363638, 8.81177584, 8.19730587, 11.12268317, 7.32079954, 12.98597263, 11.37983751, 12.17957392, 13.86193584, 12.9227764, 12.92242486, 5.9781854, 10.88541406, 10.71542038, 7.39723931, 7.82937448, 4.16852656, 11.6136011, 9.555573, 10.44076666, 12.77065598, 13.2618015, 6.94854555, 8.50983734, 10.41288112, 7.50302221, 9.4561708, 6.78278982, 8.54714174, 10.39135908, 12.83588941, 12.98166981, 10.42464574, 11.90515546, 9.25554816, 8.30995938, 10.48636779, 8.99260021, 11.19180434, 11.4184542, 9.12403343, 12.04254064, 10.45656323, 11.59690543, 8.07818264, 6.3383082, 5.40574425, 11.50857536, 10.34033654, 9.07045174, 13.03793933, 9.15660303, 10.50854287, 12.85724226, 11.69765621, 7.06040292, 5.45203063, 14.70587499, 11.38019975, 10.70959925, 5.50154404, 7.38318217, 10.8858928, 13.87744882, 10.8778617, 6.73474969, 12.30120014, 15.81840619, 10.05465336.

The status bar at the bottom indicates "completed at 3:45 PM".

Measure the performance using metrics



The screenshot shows the same Jupyter Notebook interface. The code cell now contains the following code:

```
from sklearn.metrics import r2_score
acc=r2_score(y_test,pred)
acc
```

The output of the code cell is the R-squared score: 0.5565430591634158.