RESULTS

Team: GROUP 2

Title: Crop yield management and prediction by analyzing environmental data

Members:

1. Arun Pratap Tomar - 11652000
2. Namratha Modalavalasa - 11652002
3. Padmaja Kodati - 11647780
4. Rikhita Koganti - 11641222

PySpark:

1.
```
df1 =
spark.read.format("com.dtabricks.sparks.csv").option("mode","DROPMAL
FORMED").option("header",
True).option("inferschema",True).csv("crop.csv")
df1.show()
```

```
df1.show()
```

```
+--------------------+-------------+----+-----------+--------------------+-------+----------+
|               State|District_Name|YEAR|     Season|                Crop|   Area|Production|
+--------------------+-------------+----+-----------+--------------------+-------+----------+
|Andaman and Nicob...|     NICOBARS|2000|Kharif     |           Arecanut| 1254.0|    2000.0|
|Andaman and Nicob...|     NICOBARS|2000|Kharif     |Other Kharif pulses|    2.0|       1.0|
|Andaman and Nicob...|     NICOBARS|2000|Kharif     |                Rice|  102.0|     321.0|
|Andaman and Nicob...|     NICOBARS|2000|Whole Year |              Banana|  176.0|     641.0|
|Andaman and Nicob...|     NICOBARS|2000|Whole Year |           Cashewnut|  720.0|     165.0|
|Andaman and Nicob...|     NICOBARS|2000|Whole Year |            Coconut |18168.0|    6.51E7|
|Andaman and Nicob...|     NICOBARS|2000|Whole Year |          Dry ginger|   36.0|     100.0|
|Andaman and Nicob...|     NICOBARS|2000|Whole Year |           Sugarcane|    1.0|       2.0|
|Andaman and Nicob...|     NICOBARS|2000|Whole Year |        Sweet potato|    5.0|      15.0|
|Andaman and Nicob...|     NICOBARS|2000|Whole Year |             Tapioca|   40.0|     169.0|
|Andaman and Nicob...|     NICOBARS|2001|Kharif     |           Arecanut| 1254.0|    2061.0|
|Andaman and Nicob...|     NICOBARS|2001|Kharif     |Other Kharif pulses|    2.0|       1.0|
|Andaman and Nicob...|     NICOBARS|2001|Kharif     |                Rice|   83.0|     300.0|
|Andaman and Nicob...|     NICOBARS|2001|Whole Year |           Cashewnut|  719.0|     192.0|
|Andaman and Nicob...|     NICOBARS|2001|Whole Year |            Coconut |18190.0|   6.443E7|
|Andaman and Nicob...|     NICOBARS|2001|Whole Year |          Dry ginger|   46.0|     100.0|
|Andaman and Nicob...|     NICOBARS|2001|Whole Year |           Sugarcane|    1.0|       1.0|
|Andaman and Nicob...|     NICOBARS|2001|Whole Year |        Sweet potato|   11.0|      33.0|
|Andaman and Nicob...|     NICOBARS|2002|Kharif     |                Rice|  189.2|     510.84|
|Andaman and Nicob...|     NICOBARS|2002|Whole Year |           Arecanut| 1258.0|    2083.0|
+--------------------+-------------+----+-----------+--------------------+-------+----------+
only showing top 20 rows
```

2.
```
joined_df = df1.join(df2, ["State","YEAR"], "inner")
joined_df.show()
```

```
joined_df.show()
```

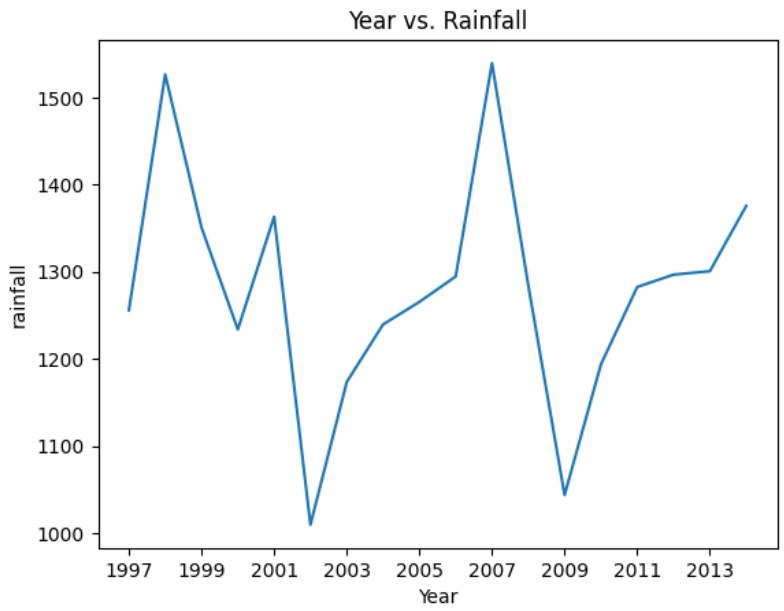
```
+----------------+----+-------------+----------+-------------+------+----------+-----+------+------+------+------+------+------+------+------+------+
|           State|YEAR|District_Name|    Season|         Crop|  Area|Production|  JAN|   FEB|   MAR|   APR|   MAY|   JUN|   JUL|   AUG|   SEP|   OCT|
+----------------+----+-------------+----------+-------------+------+----------+-----+------+------+------+------+------+------+------+------+------+
|ARUNACHAL PRADESH|2004|        ANJAW|Kharif    |        Maize|2453.0|    2674.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Kharif    |         Rice|3342.0|    1914.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Kharif    |Small millets| 532.0|     495.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Rabi      |        Wheat|  30.0|      60.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Whole Year | Dry chillies|  13.0|      18.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Whole Year |   Dry ginger|  60.0|     420.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Whole Year |       Potato|  86.0|     688.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Whole Year |    Sugarcane|   5.0|      50.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2004|        ANJAW|Whole Year |     Turmeric|   8.0|      20.0|38.00| 39.10|175.50|210.20|298.70|402.90|654.30|243.00|278.50|184.80|
|ARUNACHAL PRADESH|2005|        ANJAW|Kharif    |        Maize|3000.0|    2700.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Kharif    |         Rice|3400.0|    5000.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Kharif    |Small millets| 585.0|     525.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Rabi      |        Wheat|  32.0|      65.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Whole Year | Dry chillies|  20.0|      28.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Whole Year |   Dry ginger|  65.0|     455.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Whole Year |       Potato|  90.0|     720.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Whole Year |    Sugarcane|   7.0|      70.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2005|        ANJAW|Whole Year |     Turmeric|  10.0|      25.0|48.40|167.60|229.50|195.30|179.80|269.30|430.80|400.00|243.60|139.30|
|ARUNACHAL PRADESH|2006|        ANJAW|Kharif    |        Maize|3234.0|    3719.0| 6.00|103.70| 63.30|202.70|321.70|520.40|382.20|227.60|263.20| 77.20|
|ARUNACHAL PRADESH|2006|        ANJAW|Kharif    |         Rice|4060.0|    3004.0| 6.00|103.70| 63.30|202.70|321.70|520.40|382.20|227.60|263.20| 77.20|
+----------------+----+-------------+----------+-------------+------+----------+-----+------+------+------+------+------+------+------+------+------+
```

3.
```
from pyspark.sql.functions import  avg

# Assume the DataFrame is called `df` and contains a column `value` and a column `date` with the format 'yyyy-mm-dd'
df_avg_by_state = df_merged.groupBy('State').agg(avg('Rainfall').alias('avg_rainfall')
,avg('Production').alias('avg_production')).orderBy('State')

# Show the resulting DataFrame
df_avg_by_state.show()
df_avg_by_state_pd = df_avg_by_state.toPandas()
```

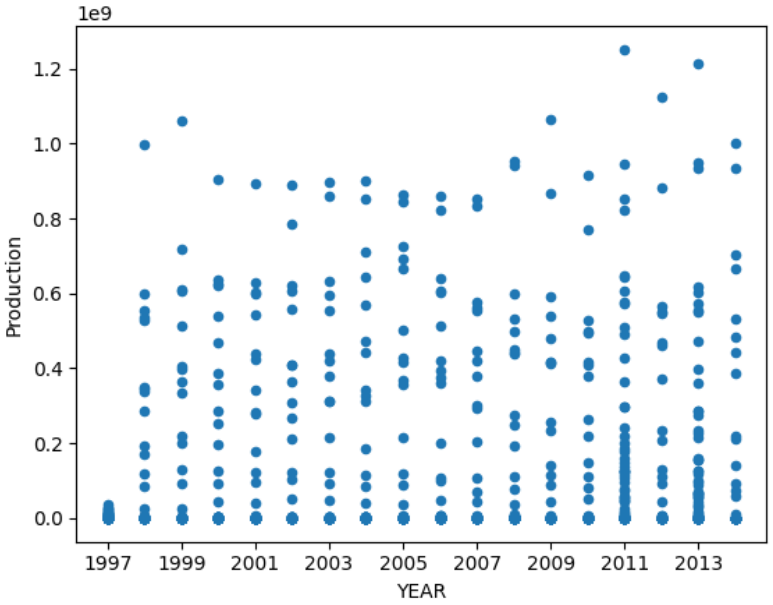
```
[ ] from pyspark.sql.functions import  avg

    # Assume the DataFrame is called `df` and contains a column `value` and a column `date` with the format 'yyyy-mm-dd'
    df_avg_by_state = df_merged.groupBy('State').agg(avg('Rainfall').alias('avg_rainfall'),avg('Production').alias('avg_production')).orderBy('State')

    # Show the resulting DataFrame
    df_avg_by_state.show()
    df_avg_by_state_pd = df_avg_by_state.toPandas()
```

```
+----------------+------------------+--------------------+
|           State|      avg_rainfall|      avg_production|
+----------------+------------------+--------------------+
|ARUNACHAL PRADESH|2504.1995679497627|  2681.3016110019653|
|           BIHAR| 1161.739888800616|   19417.37822719085|
|     CHHATTISGARH| 1274.152236436557|   9736.873843557098|
| HIMACHAL PRADESH|1134.0299518845038|    7249.66148208469|
|        JHARKHAND|1190.7740126382369|   8513.224131121651|
|          KERALA| 2841.861957286988|2.4451672589482916E7|
|          PUNJAB| 499.1636936652942|  186568.56538339166|
|       TAMIL NADU| 840.6875470584001|   910330.3971754861|
|     UTTARAKHAND|1444.4504289216177|  27394.270466321243|
+----------------+------------------+--------------------+
```

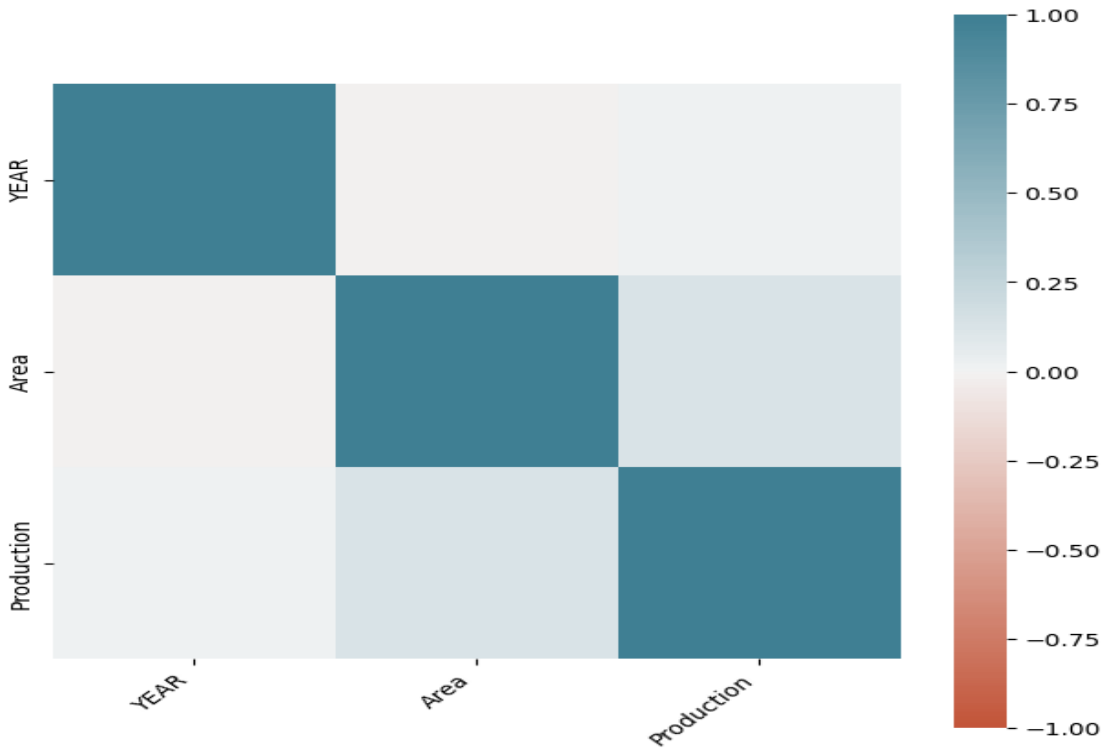
4.
```
import matplotlib.pyplot as plt
from pyspark.sql.functions import col

# select the columns to plot
x = df_avg_by_year.select(col("Year")).collect()
y = df_avg_by_year.select(col("avg_rainfall")).collect()

# create a scatter plot
plt.plot(x, y)

# add axis labels and a title
plt.xticks(range(1997, 2015, 2), [str(year) for year in range(1997,
2015, 2)])
plt.xlabel("Year")
plt.ylabel("rainfall")
plt.title("Year vs. Rainfall")

# show the plot
plt.show()
```



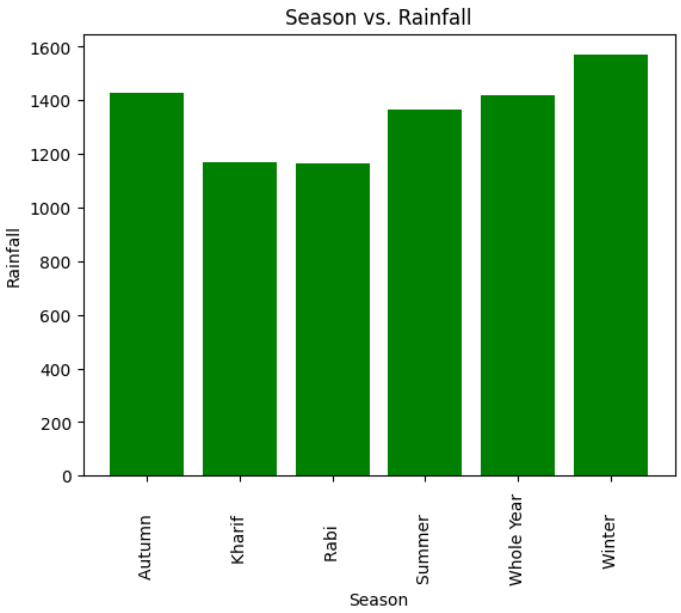
5.
```
import pandas as pd
import matplotlib.pyplot as plt

# create a scatter plot
df.plot.scatter(x='YEAR', y='Production')

# set the x-axis tick locations and labels
plt.xticks(range(1997, 2015, 2), [str(year) for year in range(1997,
2015, 2)])

# show the plot
plt.show()
```



6.
```
import matplotlib.pyplot as plt
import seaborn as sns
corr = df_updated.corr()
plt.subplots(figsize=(7.5,7.5))
ax = sns.heatmap(corr, vmin=-1, vmax=1,
center=0,cmap=sns.diverging_palette(20, 220, n=200),square=True)
ax.set_xticklabels(ax.get_xticklabels(),rotation=45,horizontalalignm
ent='right')
```



7.
```
import matplotlib.pyplot as plt
from pyspark.sql.functions import col

# select the columns to plot
x = df_avg_by_season_pd["Season"]
y= df_avg_by_season_pd["avg_rainfall"]
#y = df.select(col("Production")).collect()

# create a plot
plt.bar(x, y, color = 'green')

# add axis labels and a title
#plt.xticks(range(1997, 2015, 2), [str(year) for year in range(1997,
2015, 2)])
plt.xlabel("Season")
plt.ylabel("Rainfall")
plt.title("Season vs. Rainfall")
plt.xticks(rotation=90)
# show the plot
plt.show()
```



8. **create table rainfall_district (State text,district text, JAN decimal, FEB decimal, MAR decimal, APR decimal, MAY decimal, JUN decimal, JUL decimal, AUG decimal, SEP decimal, OCT decimal, NOV decimal, DEC decimal, ANNUAL decimal, Jan_Feb decimal, Mar_May decimal, Jun_Sep decimal, Oct_Dec decimal, PRIMARY KEY(State,district, ANNUAL));**

```
b96859b\cassandra\io\asyncorereactor.py", line 335, in create_timer
 File "C:\Cassandra\apache-cassandra-3.11.14\bin\..\lib\cassandra-driver-internal-only-3.11.0-bb96859b.zip\cassandra-driver-3.11.0-bb
96859b\cassandra\io\asyncorereactor.py", line 373, in close
 ttributeError: 'NoneType' object has no attribute 'add_timer'
   cls._loop.add_timer(timer)
A ttributeError: 'NoneType' object has no attribute 'add_timer'
 ttributeError: 'NoneType' object has no attribute 'add_timer'
A   cls._loop.add_timer(timer)
AttributeError: 'NoneType' object has no attribute 'add_timer'
   self._connection.close()
    cls._loop.add_timer(timer)
ttributeError: 'NoneType' object has no attribute 'add_timer'
ttributeError: 'NoneType' object has no attribute 'add_timer'
A    AsyncoreConnection.create_timer(0, partial(asyncore.dispatcher.close, self))
AttributeError: 'NoneType' object has no attribute 'add_timer'
ttributeError: 'NoneType' object has no attribute 'add_timer'
  File "C:\Cassandra\apache-cassandra-3.11.14\bin\..\lib\cassandra-driver-internal-only-3.11.0-bb96859b.zip\cassandra-driver-3.11.0-b
b96859b\cassandra\io\asyncorereactor.py", line 373, in close
 File "C:\Cassandra\apache-cassandra-3.11.14\bin\..\lib\cassandra-driver-internal-only-3.11.0-bb96859b.zip\cassandra-driver-3.11.0-bb
96859b\cassandra\io\asyncorereactor.py", line 335, in create_timer
     cls._loop.add_timer(timer)
   AsyncoreConnection.create_timer(0, partial(asyncore.dispatcher.close, self))
A ttributeError: 'NoneType' object has no attribute 'add_timer'
 File "C:\Cassandra\apache-cassandra-3.11.14\bin\..\lib\cassandra-driver-internal-only-3.11.0-bb96859b.zip\cassandra-driver-3.11.0-bb
96859b\cassandra\io\asyncorereactor.py", line 335, in create_timer
    cls._loop.add_timer(timer)
AttributeError: 'NoneType' object has no attribute 'add_timer'
Processed: 4116 rows; Rate:     490 rows/s; Avg. rate:     954 rows/s
4090 rows imported from 1 files in 4.313 seconds (0 skipped).
cqlsh:crop_prediction> create table rainfall_district (State text,district text, JAN decimal, FEB decimal, MAR decimal, APR decimal,
MAY decimal, JUN decimal, JUL decimal, AUG decimal, SEP decimal, OCT decimal, NOV decimal, DEC decimal, ANNUAL decimal, Jan_Feb decim
al, Mar_May decimal, Jun_Sep decimal, Oct_Dec decimal, PRIMARY KEY(State,district, ANNUAL));
```

9. Select state,district_name,year, season from crop where State = 'Andhra Pradesh' and district_name = 'VISAKHAPATANAM' and Year = 1997 allow filtering;

```
cqlsh:crop_prediction> select state,district_name,year, season from crop where State = 'Andhra Pradesh' and district_name = 'VISAKHAP
ATANAM' and Year = 1997 allow filtering;

 state          | district_name  | year | season
----------------+----------------+------+------------
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Kharif
 Andhra Pradesh | VISAKHAPATANAM | 1997 | Rabi

(21 rows)
```

10. Select subdivision, year, annual as rainfall from rainfall_subdivision where annual > 4000 allow filtering;

```
cqlsh:crop_prediction> select subdivision, year, annual as rainfall from rainfall_subdivision where annual > 4000 allow filtering;

 subdivision            | year | rainfall
------------------------+------+----------
      COASTAL KARNATAKA | 1933 |  4139.50
      COASTAL KARNATAKA | 1946 |  4041.40
      COASTAL KARNATAKA | 1959 |  4330.80
      COASTAL KARNATAKA | 1961 |  5553.90
      COASTAL KARNATAKA | 1962 |  4017.70
      COASTAL KARNATAKA | 1970 |  4042.00
      COASTAL KARNATAKA | 1975 |  4300.00
      COASTAL KARNATAKA | 1982 |  4090.90
      COASTAL KARNATAKA | 1990 |  4120.40
      COASTAL KARNATAKA | 1994 |  4636.30
      COASTAL KARNATAKA | 2013 |  4229.40
           KONKAN & GOA | 1958 |  4000.20
     ARUNACHAL PRADESH | 1918 |  5486.30
     ARUNACHAL PRADESH | 1919 |  4693.90
     ARUNACHAL PRADESH | 1920 |  4106.70
     ARUNACHAL PRADESH | 1921 |  5691.40
     ARUNACHAL PRADESH | 1922 |  4187.20
     ARUNACHAL PRADESH | 1924 |  4450.50
     ARUNACHAL PRADESH | 1925 |  4439.80
     ARUNACHAL PRADESH | 1927 |  5253.20
     ARUNACHAL PRADESH | 1928 |  5272.70
     ARUNACHAL PRADESH | 1929 |  4320.20
     ARUNACHAL PRADESH | 1930 |  4334.10
     ARUNACHAL PRADESH | 1931 |  4872.10
     ARUNACHAL PRADESH | 1932 |  4850.90
     ARUNACHAL PRADESH | 1934 |  4575.70
     ARUNACHAL PRADESH | 1935 |  4173.70
```

11. Select subdivision,year, avg(annual) from rainfall_subdivision group by subdivision,year allow filtering;

```
cqlsh:crop_prediction> select state, avg(annual) as average_rainfall from rainfall_district group by state allow filtering;

 state                       | average_rainfall
-----------------------------+------------------
 ANDAMAN And NICOBAR ISLANDS |           2911.4
               UTTAR PRADESH |            955.3
                     TRIPURA |           2479.1
                   MEGHALAYA |           3682.9
                       DELHI |            747.1
                   KARNATAKA |           1194.6
                       ASSAM |           2454.4
                    HIMACHAL |           1371.6
                   RAJASTHAN |            581.4
             JAMMU AND KASHMIR |           1016.6
                MADHYA PRADESH |           1032.3
                     JHARKHAND |           1303.5
                       BIHAR |           1200.5
                     HARYANA |            614.6
               ANDHRA PRADESH |            945.1
                      ORISSA |           1466.1
                     MANIPUR |           2496.6
                 LAKSHADWEEP |             1600
                     GUJARAT |            924.3
                    NAGALAND |           1940.7
                  CHATISGARH |           1286.4
               DAMAN AND DUI |           1535.7
                  MAHARASHTRA |           1278.6
             ARUNACHAL PRADESH |           2927.4
                         GOA |           3278.5
                      SIKKIM |           2838.4
                  PONDICHERRY |           1378.4
            DADAR NAGAR HAVELI |           2374.1
```

12.
```
from sklearn.model_selection import train_test_split
train_x, test_x, train_y, test_y = train_test_split(X, y,
test_size=0.3, random_state=42)
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(criterion="entropy", max_depth=3)
model.fit(train_x, train_y)
predicted = model.predict(test_x)
model.score(test_x, test_y)
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(test_y, predicted)
cm
from sklearn.metrics import roc_auc_score
probabilities = model.predict_proba(test_x)
roc_auc_score(test_y, probabilities[:, 1])
from sklearn.metrics import precision_score

train_predictions = model.predict(train_x)
precision_score(train_y, train_predictions)
from sklearn.metrics import recall_score

recall_score(train_y, train_predictions)
from sklearn.metrics import roc_curve
```

```
fpr, tpr, _ = roc_curve(test_y, probabilities[:, 1])
plt.plot(fpr, tpr)
plt.plot([0, 1], [0, 1], color='grey', lw=1, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
```

```
[53] X = df[['YEAR','Rainfall','Area']]
     y = df[['Perf']]
```

```
[54] from sklearn.model_selection import train_test_split
     train_x, test_x, train_y, test_y = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
[69] from sklearn.tree import DecisionTreeClassifier
     model = DecisionTreeClassifier(criterion="entropy", max_depth=3)
     model.fit(train_x, train_y)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', max_depth=3)
```

```
[70] predicted = model.predict(test_x)
     model.score(test_x, test_y)

0.916963418582065 4
```

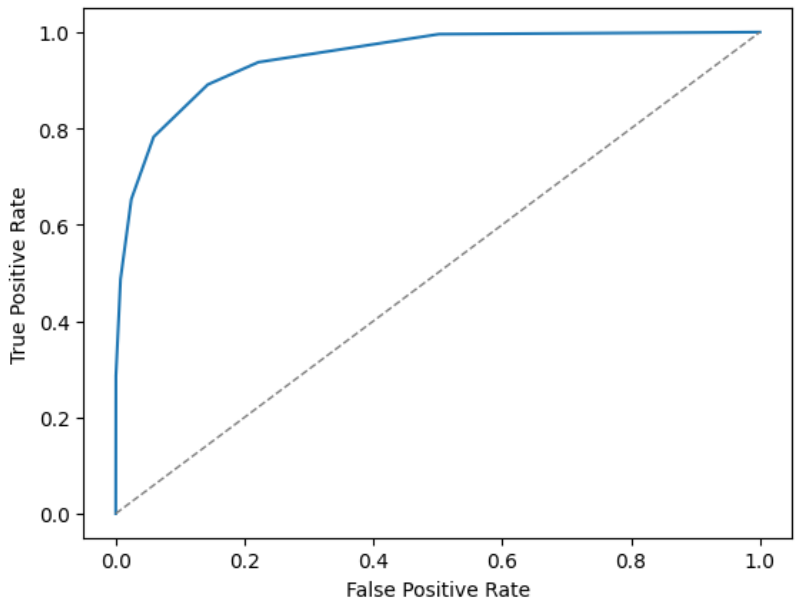
```
[71] from sklearn.metrics import confusion_matrix
     cm=confusion_matrix(test_y, predicted)
     cm

array([[14789,   366],
       [ 1173,  2206]])
```

```
[72] from sklearn.metrics import roc_auc_score
     probabilities = model.predict_proba(test_x)
     roc_auc_score(test_y, probabilities[:, 1])

0.9475774401423037
```

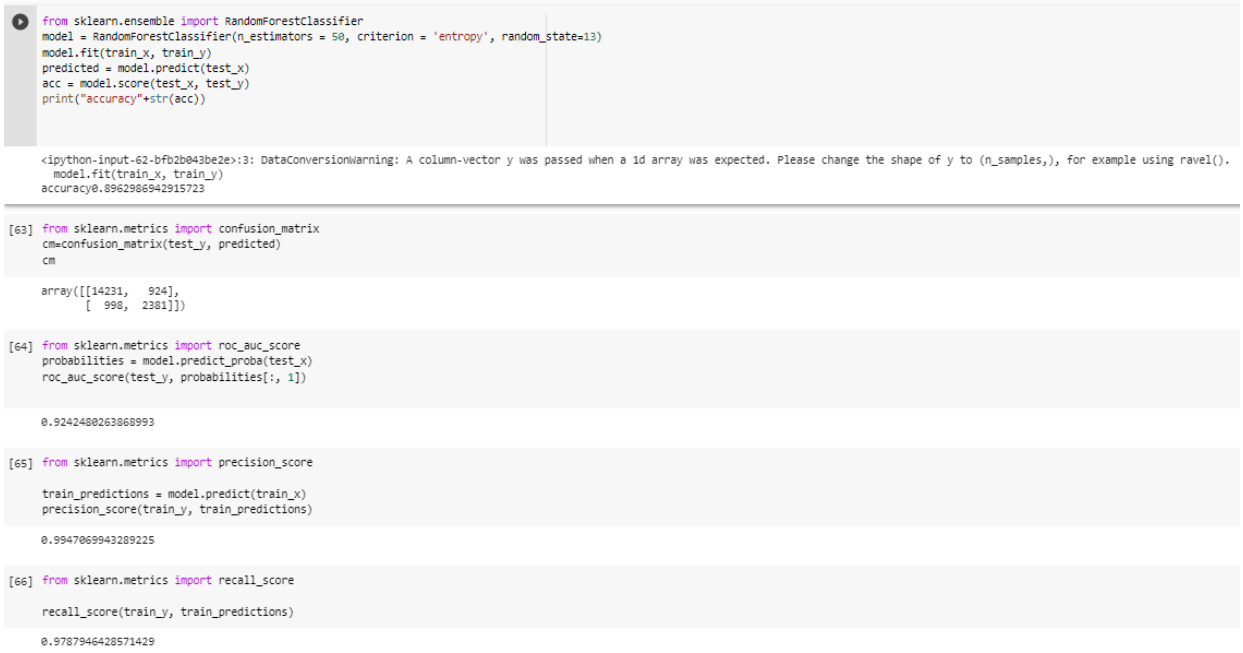
```
[73] from sklearn.metrics import precision_score

     train_predictions = model.predict(train_x)
     precision_score(train_y, train_predictions)

0.8690825987593862
```

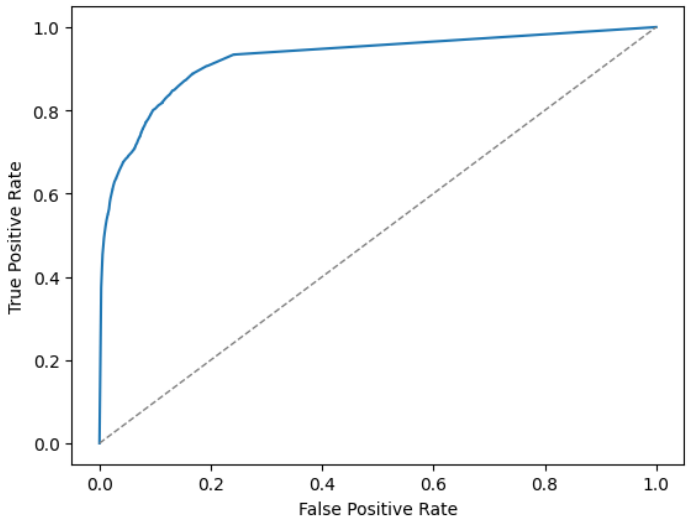
```
[74] from sklearn.metrics import recall_score

     recall_score(train_y, train_predictions)

0.660218253968254
```

ROC Curve:



13.
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators = 50, criterion =
'entropy', random_state=13)
model.fit(train_x, train_y)
predicted = model.predict(test_x)
model.score(test_x, test_y)
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(test_y, predicted)
cm
from sklearn.metrics import roc_auc_score
probabilities = model.predict_proba(test_x)
roc_auc_score(test_y, probabilities[:, 1])
from sklearn.metrics import precision_score
train_predictions = model.predict(train_x)
precision_score(train_y, train_predictions)
from sklearn.metrics import recall_score
recall_score(train_y, train_predictions)
from sklearn.metrics import roc_curve
fpr, tpr, _ = roc_curve(test_y, probabilities[:, 1])
plt.plot(fpr, tpr)
plt.plot([0, 1], [0, 1], color='grey', lw=1, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
```

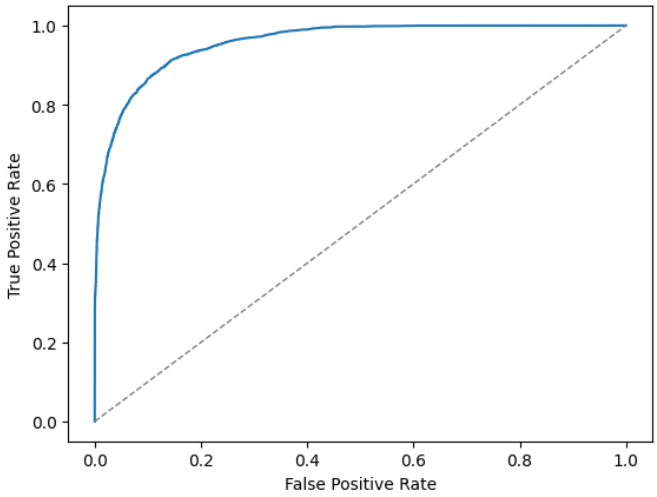




14.

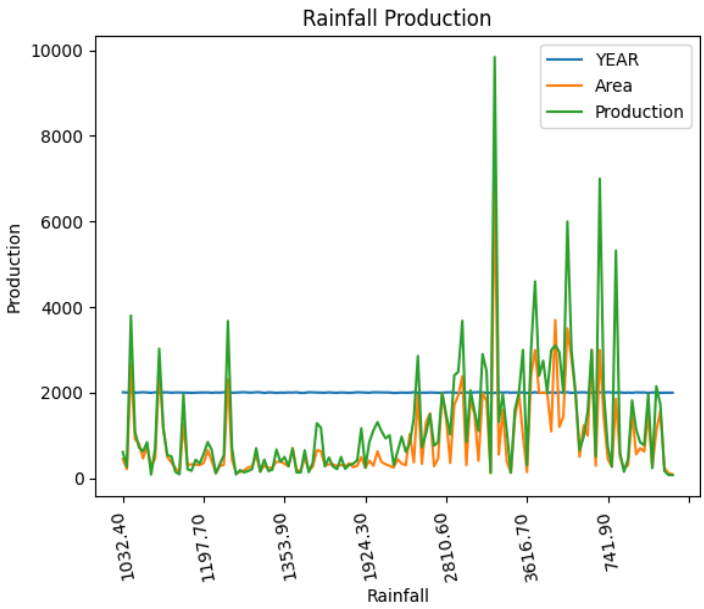
```
from sklearn.ensemble import GradientBoostingClassifier
model = GradientBoostingClassifier(random_state=13)
model.fit(train_x, train_y)
predicted = model.predict(test_x)
model.score(test_x, test_y)
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(test_y, predicted)
cm
from sklearn.metrics import roc_auc_score
probabilities = model.predict_proba(test_x)
roc_auc_score(test_y, probabilities[:, 1])
from sklearn.metrics import precision_score
train_predictions = model.predict(train_x)
precision_score(train_y, train_predictions)
from sklearn.metrics import recall_score
recall_score(train_y, train_predictions)
from sklearn.metrics import roc_curve
fpr, tpr, _ = roc_curve(test_y, probabilities[:, 1])
plt.plot(fpr, tpr)
plt.plot([0, 1], [0, 1], color='grey', lw=1, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
```

```
[68] from sklearn.ensemble import GradientBoostingClassifier
     model = GradientBoostingClassifier(random_state=13)
     model.fit(train_x, train_y)
     predicted = model.predict(test_x)
     acc = model.score(test_x, test_y)
     print("accuracy"+str(acc))

     /usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_gb.py:437: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
       y = column_or_1d(y, warn=True)
     accuracy0.9199309377360526

[69] from sklearn.metrics import confusion_matrix
     cm=confusion_matrix(test_y, predicted)
     cm

     array([[14639,   516],
            [  968,  2411]])

[70] from sklearn.metrics import roc_auc_score
     probabilities = model.predict_proba(test_x)
     roc_auc_score(test_y, probabilities[:, 1])

     0.9574578228777135

[71] from sklearn.metrics import precision_score

     train_predictions = model.predict(train_x)
     precision_score(train_y, train_predictions)

     0.8401562273976566

[72] from sklearn.metrics import recall_score

     recall_score(train_y, train_predictions)

     0.7202380952380952
```



15. Group by plot

```
perf_by_state=df.groupby('Rainfall').median()
plot_by_state=perf_by_state.plot(title="Rainfall
Production",rot=100)
plot_by_state.set_xlabel('Rainfall')
plot_by_state.set_ylabel('Production')
```


Rainfall Production
YEAR
Area
Production
10000
8000
6000
4000
2000
0
Production
1032.40
1197.70
1353.90
1924.30
2810.60
3616.70
741.90
Rainfall