Deliverable 2 - Group 1

PROJECT ARTIFACT REPOSITORY

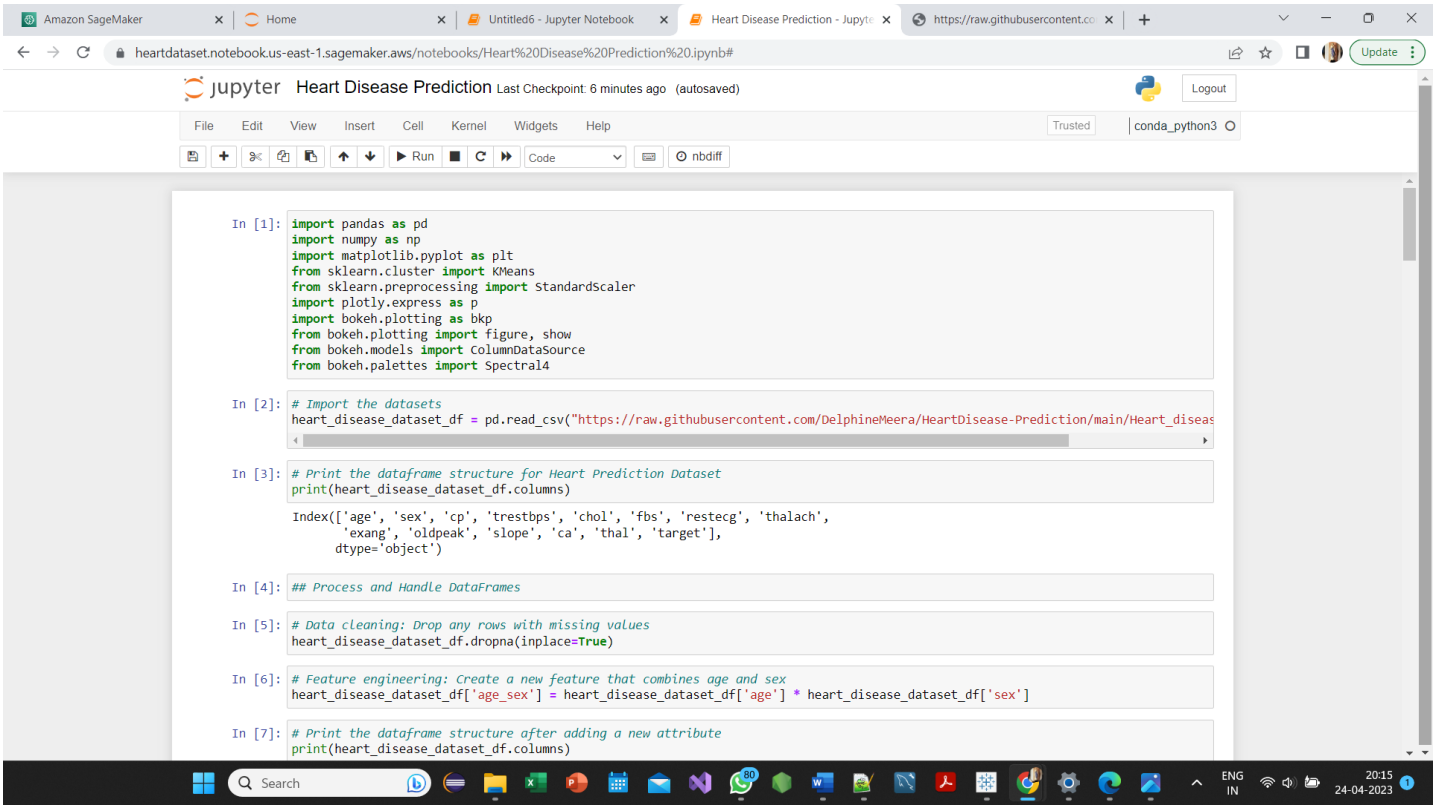
Our project work can be found in the public repository that has been created on Github.
Link to the Repository - https://github.com/DelphineMeera/HeartDisease-Prediction

FIRST CHOICE - HEART DISEASE PREDICTION

DATA UNDERSTANDING

a. Exploring Data Analysis

Dataset - https://www.kaggle.com/datasets/ritwikb3/heart-disease-cleveland
Amazon S3 Bucket - s3://heartdiseasedataset-bucket
Exploratory Data Analysis was done using AWS Sagemaker.



Amazon SageMaker | Home | Untitled6 - Jupyter Notebook | Heart Disease Prediction - Jupyter | https://raw.githubusercontent.com

heartdataset.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Disease%20Prediction%20.ipynb#

jupyter **Heart Disease Prediction** Last Checkpoint: 6 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | conda_python3

Run | Code | nbdiff

In [7]:
```
# Print the dataframe structure after adding a new attribute
print(heart_disease_dataset_df.columns)
```

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target', 'age_sex'],
      dtype='object')
```

In [8]:
```
# Create a correlation matrix of the features
corr_matrix = heart_disease_dataset_df.corr()
# Select the top 5 most correlated features with the target variable 'target'
top_features = corr_matrix.nlargest(6, 'target')['target'].index
df_top = heart_disease_dataset_df[top_features]
```

In [9]:
```
# Display the top 5 most correlated features with the target variable 'target'
print(top_features)
```

```
Index(['target', 'thal', 'ca', 'exang', 'oldpeak', 'cp'], dtype='object')
```

In [10]:
```
# Display the top 5 most correlated features with the target value
print(df_top)
```

```
     target  thal  ca  exang  oldpeak  cp
0         0     2   0      0      2.3   0
1         1     1   3      1      1.5   3
2         1     3   2      1      2.6   3
3         0     1   0      0      3.5   2
4         0     1   0      0      1.4   1
..      ...   ...  ..    ...      ...  ..
298       1     3   0      0      1.2   0
299       1     3   2      0      3.4   3
300       1     3   1      1      1.2   3
301       1     1   1      0      0.0   1
302       0     1   0      0      0.0   2

[303 rows x 6 columns]
```

Search | ENG IN | 20:15 24-04-2023

Amazon SageMaker | Home | Untitled6 - Jupyter Notebook | Heart Disease Prediction - Jupyter | https://raw.githubusercontent.com

heartdataset.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Disease%20Prediction%20.ipynb#

jupyter **Heart Disease Prediction** Last Checkpoint: 6 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | conda_python3

Run | Code | nbdiff

In [11]:
```
# Print the first 5 entries of the Heart Disease Prediction dataset
print(heart_disease_dataset_df.head(5))
```

```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   63    1   0       145   233    1        2      150      0      2.3      2
1   67    1   3       160   286    0        2      108      1      1.5      1
2   67    1   3       120   229    0        2      129      1      2.6      1
3   37    1   2       130   250    0        0      187      0      3.5      2
4   41    0   1       130   204    0        2      172      0      1.4      0

   ca  thal  target  age_sex
0   0     2       0       63
1   3     1       1       67
2   2     3       1       67
3   0     1       0       37
4   0     1       0        0
```

In [12]:
```
# Describe the Heart Disease Prediction dataset
heart_disease_dataset_df.describe()
```

Out[12]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0
mean	54.438944	0.679868	2.158416	131.689769	246.693069	0.148515	0.990099	149.607261	0.326733	1.039604	0.600660	0.663366	1.8
std	9.038662	0.467299	0.960126	17.599748	51.776918	0.356198	0.994971	22.875003	0.469794	1.161075	0.616226	0.934375	0.9
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	1.0
25%	48.000000	0.000000	2.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	0.000000	0.000000	1.0
50%	56.000000	1.000000	2.000000	130.000000	241.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	1.0
75%	61.000000	1.000000	3.000000	140.000000	275.000000	0.000000	2.000000	166.000000	1.000000	1.600000	1.000000	1.000000	3.0
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	3.000000	3.0

Search | ENG IN | 20:15 24-04-2023

Amazon SageMaker | Home | Untitled6 - Jupyter Notebook | Heart Disease Prediction - Jupyte | https://raw.githubusercontent.co

heartdataset.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Disease%20Prediction%20.ipynb#

Update

jupyter **Heart Disease Prediction** Last Checkpoint: 6 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

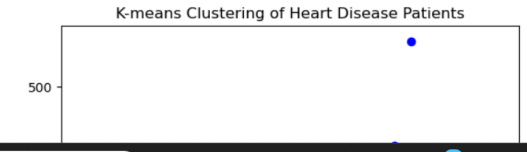
Trusted | conda_python3

Run | Code | nbdiff

In [13]:
```
## Dashboards
```

In [14]:
```
# Drop the target column from the dataset
X = heart_disease_dataset_df.drop('target', axis=1)

# Scale the features using StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
kmeans = KMeans(n_clusters=2, n_init=10, random_state=42)
clusters = kmeans.fit_predict(X_scaled)

# Add the clusters as a new column in the dataset
heart_disease_dataset_df['cluster'] = clusters

# Plot the scatter plot with the clusters colored by target
plt.scatter(heart_disease_dataset_df['age'][heart_disease_dataset_df['target'] == 0], heart_disease_dataset_df['chol'][heart_dise
plt.scatter(heart_disease_dataset_df['age'][heart_disease_dataset_df['target'] == 1], heart_disease_dataset_df['chol'][heart_dise

# Add axis labels and title
plt.xlabel('Age')
plt.ylabel('Cholesterol')
plt.title('K-means Clustering of Heart Disease Patients')#Histogram of Age
```

Out[14]: Text(0.5, 1.0, 'K-means Clustering of Heart Disease Patients')



Search | ENG IN | 20:15 24-04-2023

Amazon SageMaker | Home | Untitled6 - Jupyter Notebook | Heart Disease Prediction - Jupyte | https://raw.githubusercontent.co

heartdataset.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Disease%20Prediction%20.ipynb#

Update

jupyter **Heart Disease Prediction** Last Checkpoint: 6 minutes ago (autosaved)

Logout

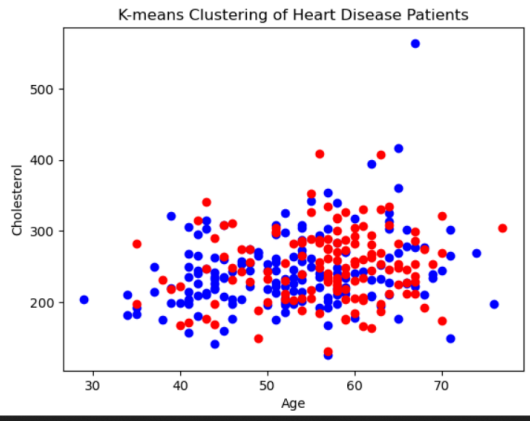
File Edit View Insert Cell Kernel Widgets Help

Trusted | conda_python3

Run | Code | nbdiff

```
plt.scatter(heart_disease_dataset_df['age'][heart_disease_dataset_df['target'] == 0], heart_disease_dataset_df['chol'][heart_dise
plt.scatter(heart_disease_dataset_df['age'][heart_disease_dataset_df['target'] == 1], heart_disease_dataset_df['chol'][heart_dise

# Add axis labels and title
plt.xlabel('Age')
plt.ylabel('Cholesterol')
plt.title('K-means Clustering of Heart Disease Patients')#Histogram of Age
```

Out[14]: Text(0.5, 1.0, 'K-means Clustering of Heart Disease Patients')



Search | ENG IN | 20:16 24-04-2023

Amazon SageMaker | Home | Untitled6 - Jupyter Notebook | Heart Disease Prediction - Jupyte | https://raw.githubusercontent.co

heartdataset.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Disease%20Prediction%20.ipynb#

Update

jupyter **Heart Disease Prediction** Last Checkpoint: 6 minutes ago (autosaved)

Logout

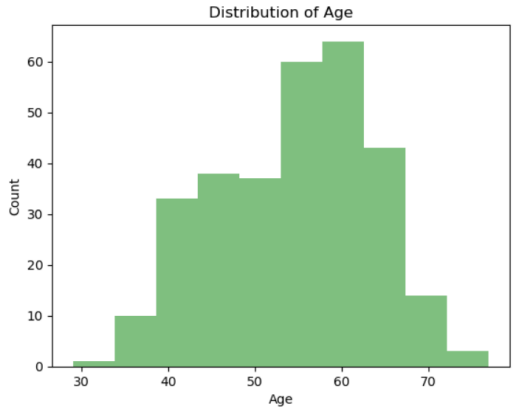
File Edit View Insert Cell Kernel Widgets Help

Trusted | conda_python3

Run | Code | nbdiff

In [15]:
```
#Histogram of Age
```

In [16]:
```
plt.hist(heart_disease_dataset_df['age'], bins=10, alpha=0.5, color='green')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



Search | ENG IN | 20:16 24-04-2023

Amazon SageMaker | Home | Untitled6 - Jupyter Notebook | Heart Disease Prediction - Jupyte | https://raw.githubusercontent.co

heartdataset.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Disease%20Prediction%20.ipynb#

Update

jupyter **Heart Disease Prediction** Last Checkpoint: 6 minutes ago (autosaved)

Logout

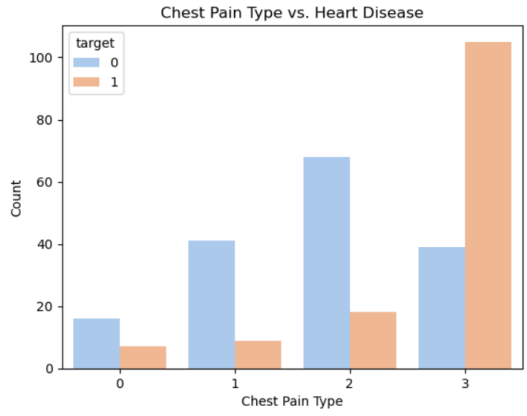
File Edit View Insert Cell Kernel Widgets Help

Trusted | conda_python3

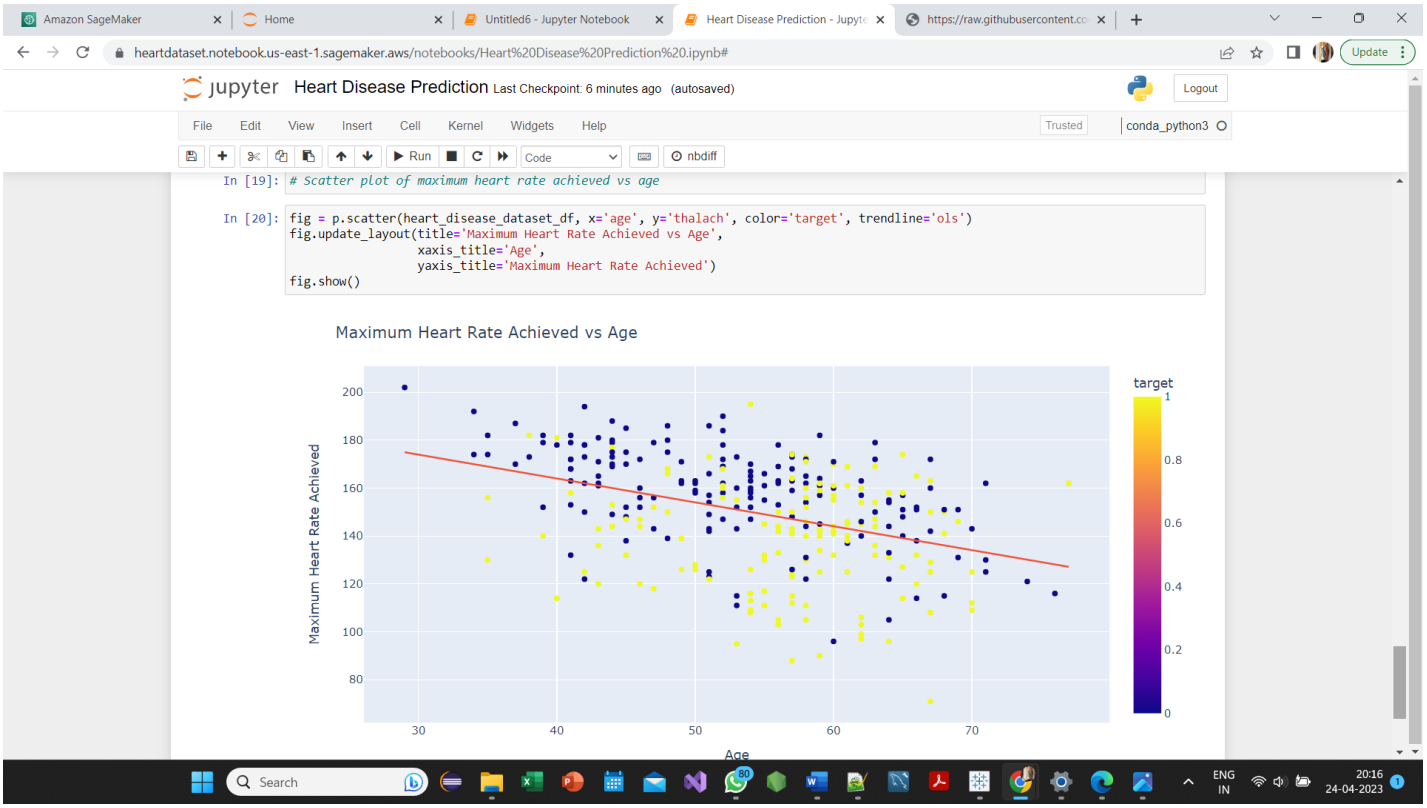
Run | Code | nbdiff

In [17]:
```
# Bar plot of chest pain type
```

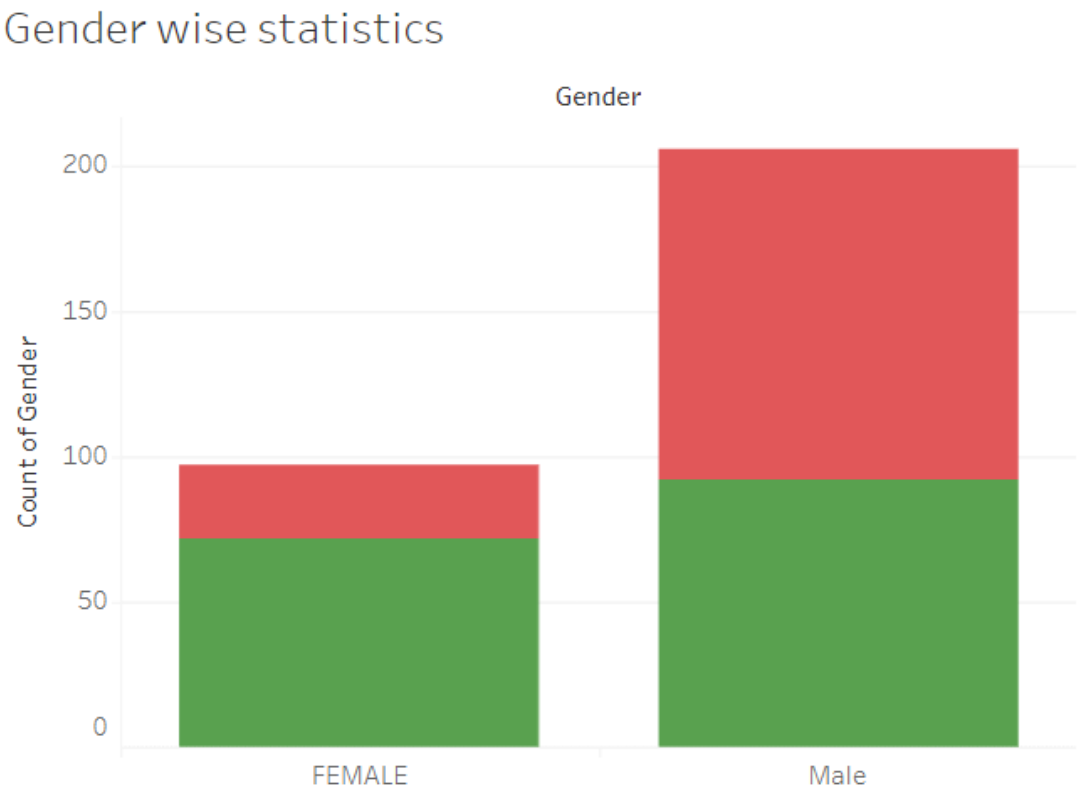
In [18]:
```
import seaborn as sns
sns.countplot(x='cp', hue='target', data=heart_disease_dataset_df, palette='pastel')
plt.title('Chest Pain Type vs. Heart Disease')
plt.xlabel('Chest Pain Type')
plt.ylabel('Count')
plt.show()
```



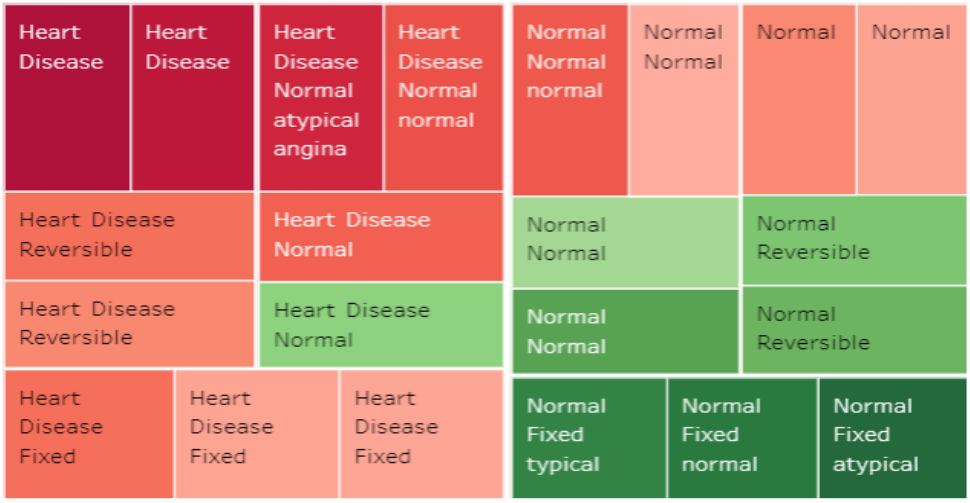
Search | ENG IN | 20:16 24-04-2023


Amazon SageMaker
Home
Untitled6 - Jupyter Notebook
Heart Disease Prediction - Jupyt
https://raw.githubusercontent.co
heartdataset.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Disease%20Prediction%20.ipynb#
Update
jupyter Heart Disease Prediction Last Checkpoint: 6 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
conda_python3
Run
Code
nbdiff
In [19]: # Scatter plot of maximum heart rate achieved vs age
In [20]: fig = p.scatter(heart_disease_dataset_df, x='age', y='thalach', color='target', trendline='ols')
fig.update_layout(title='Maximum Heart Rate Achieved vs Age',
xaxis_title='Age',
yaxis_title='Maximum Heart Rate Achieved')
fig.show()
Maximum Heart Rate Achieved vs Age
Maximum Heart Rate Achieved
200
180
160
140
120
100
80
30
40
50
60
70
Age
target
1
0.8
0.6
0.4
0.2
0
Search
ENG
IN
20:16
24-04-2023

b. Dashboard


Gender wise statistics
Gender
Count of Gender
200
150
100
50
0
FEMALE
Male

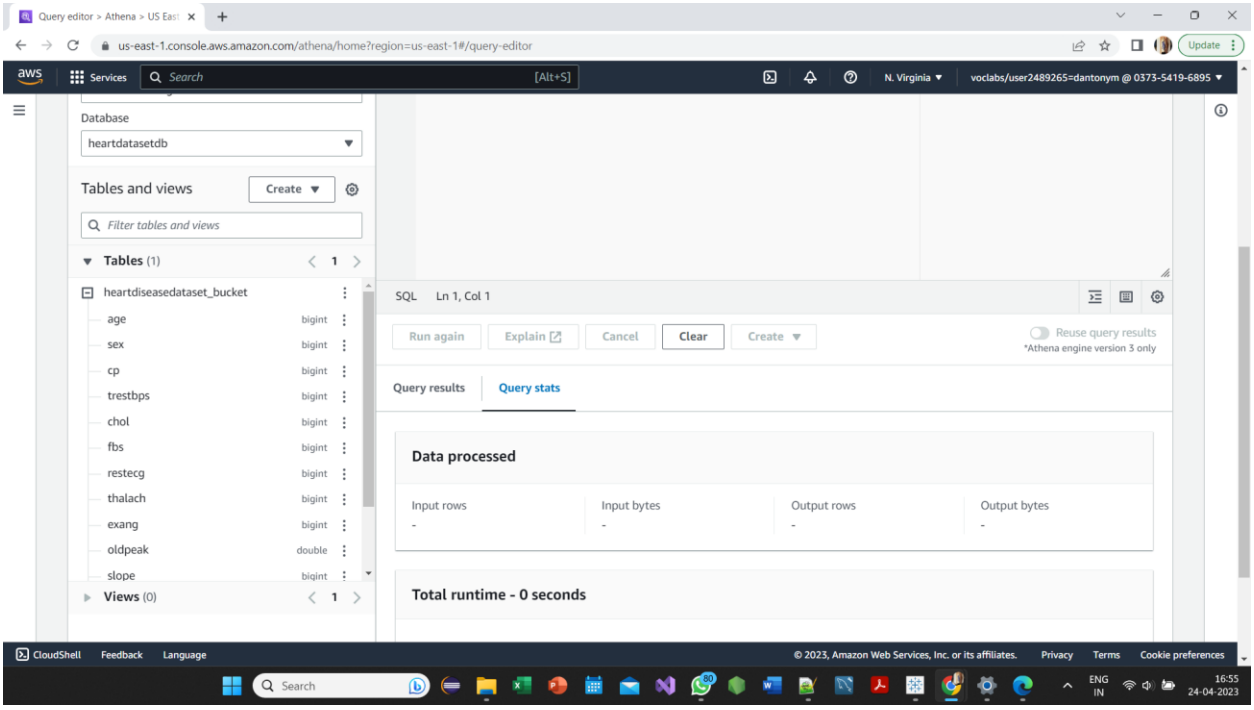
Effect of multiple conditions on Heart disease

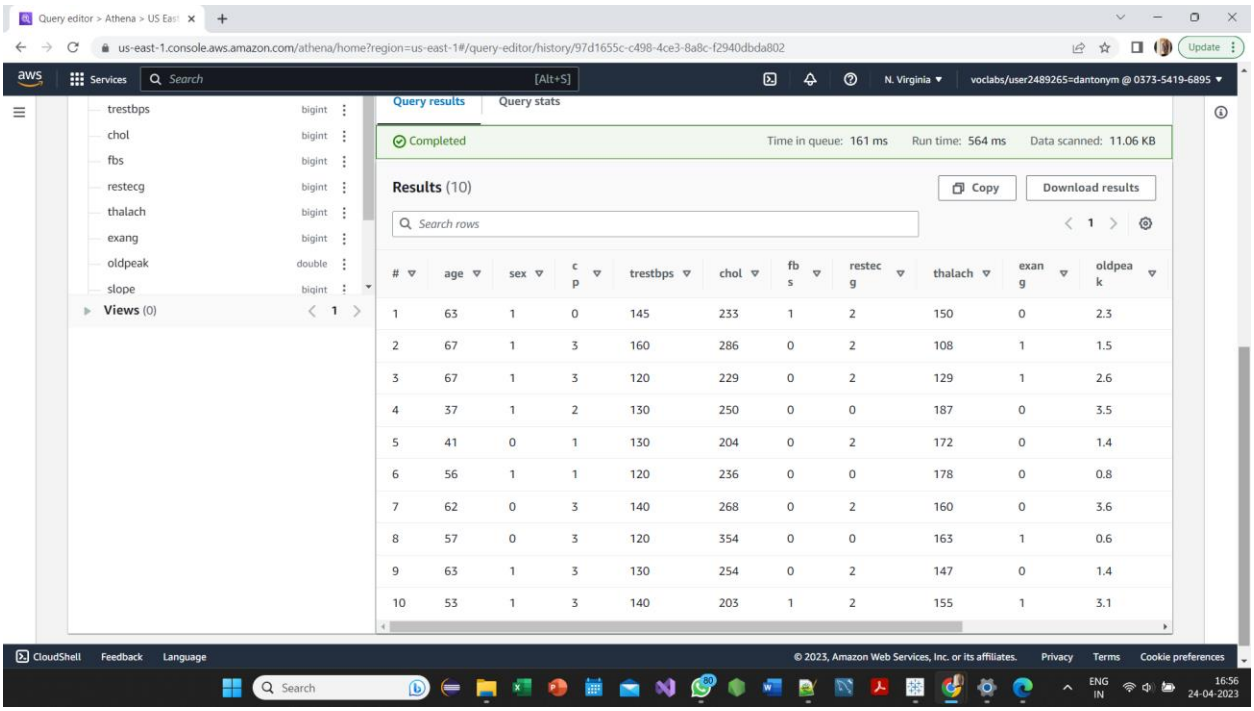


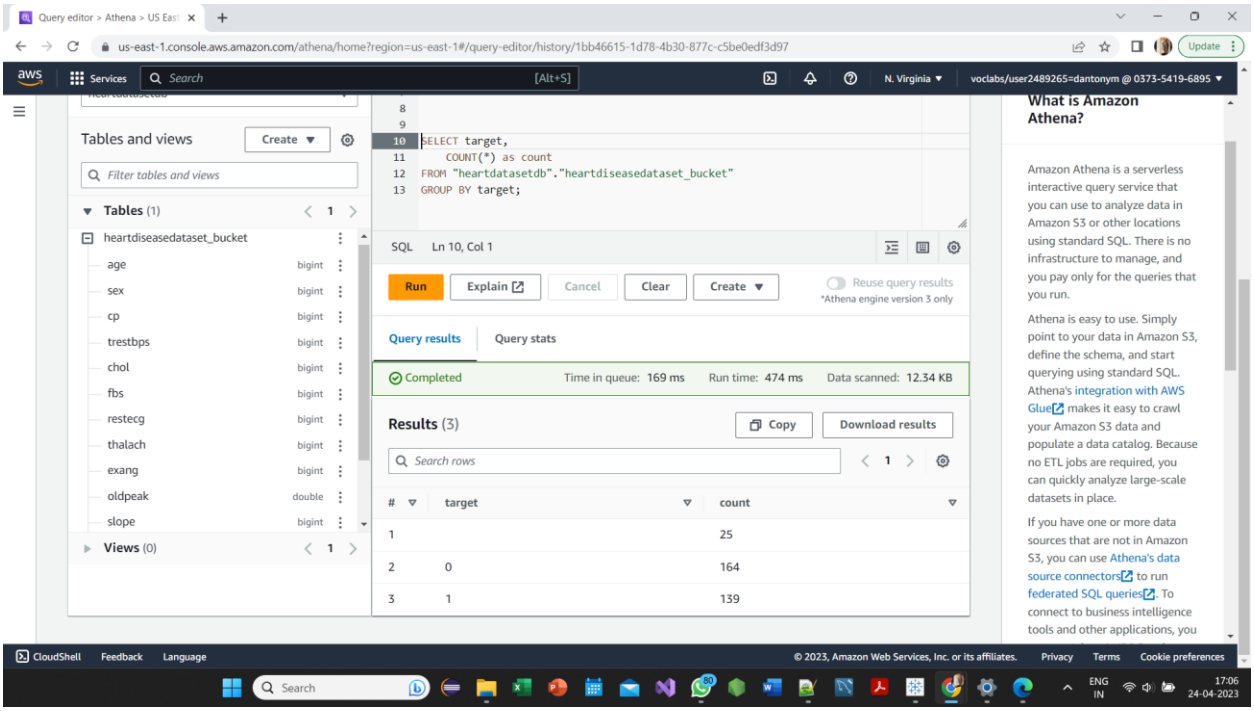
Slope of ST segment

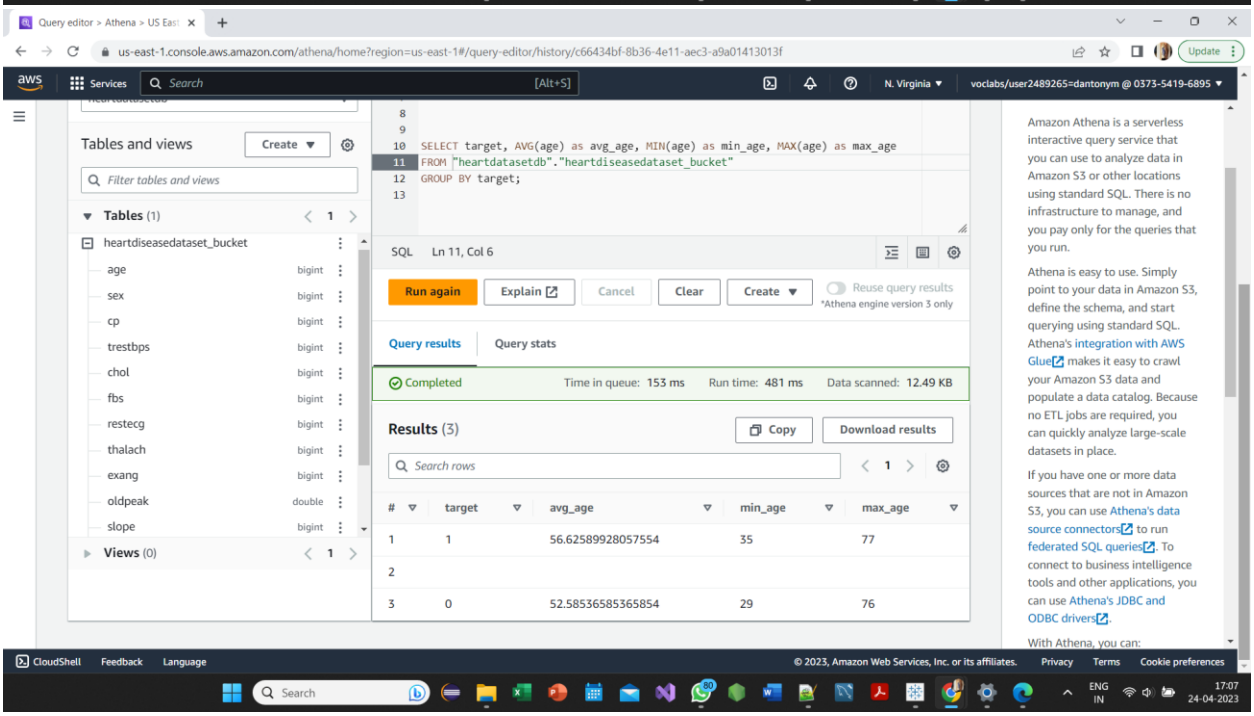
DATA PREPARATION

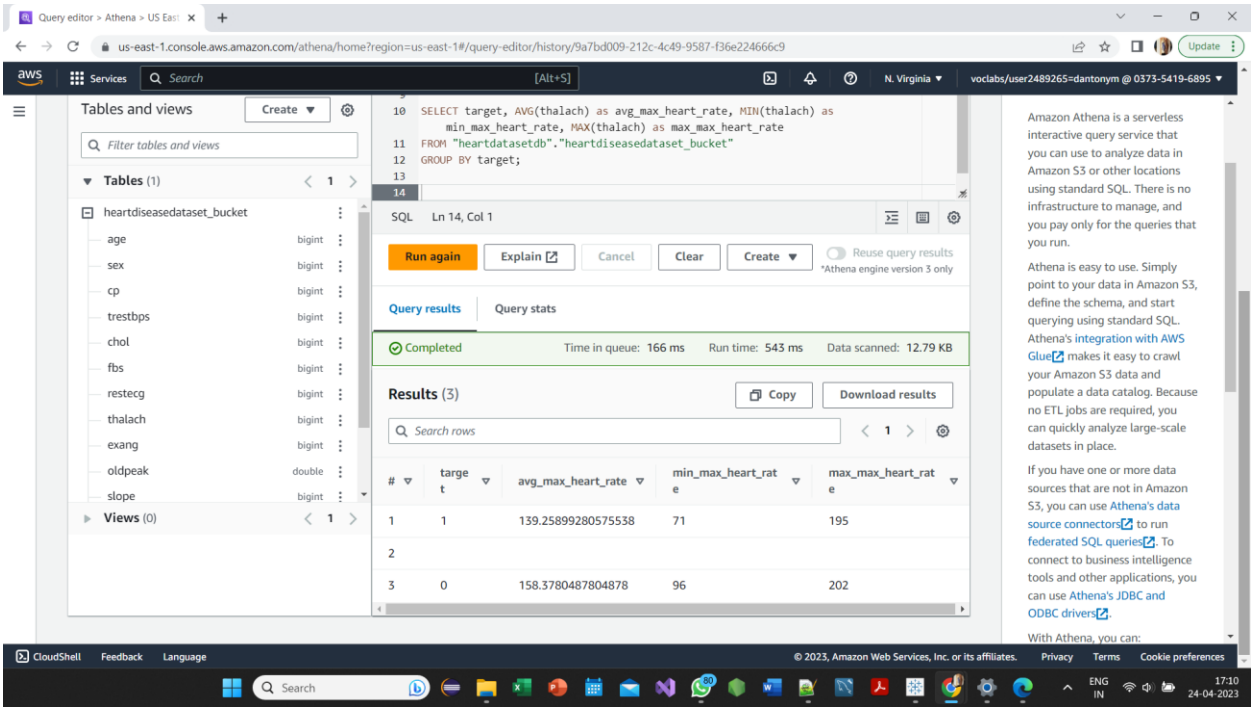
Data preparation was conducted throughout the project. Data preparation involved activities like adding new features to predict the target variable, encoding categorical variables into numeric values, checking for any outliers or any missing values in the data.

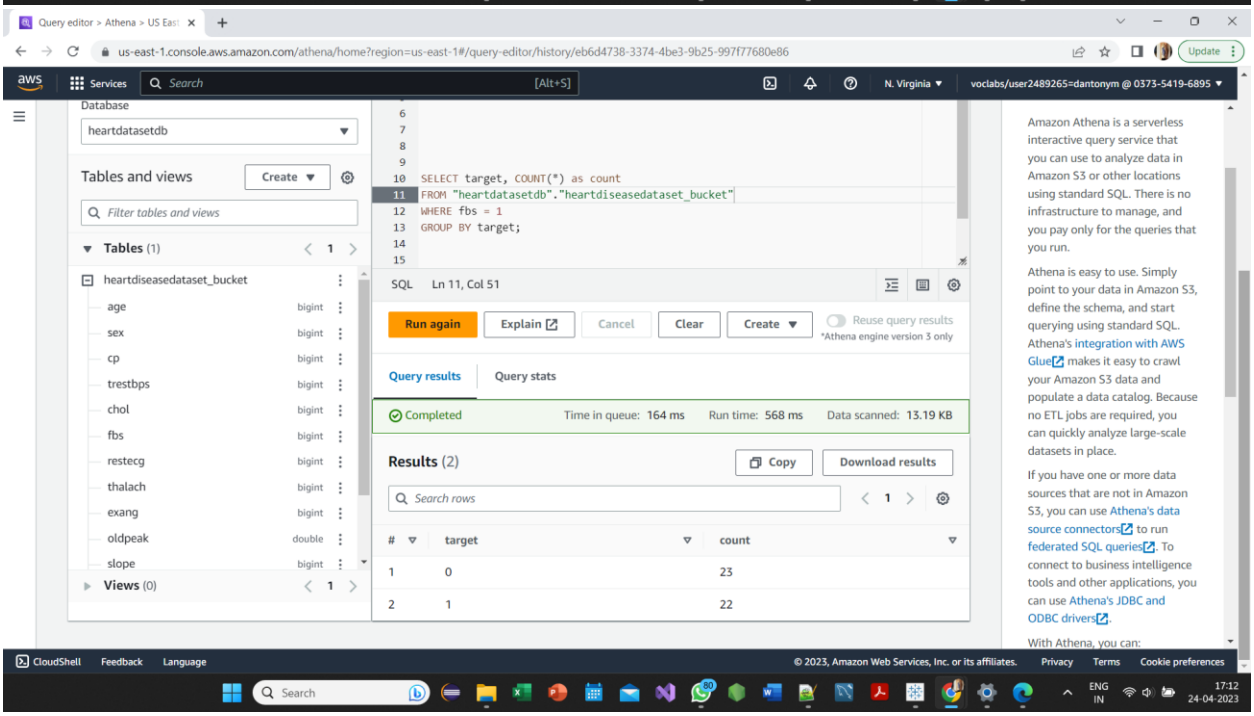


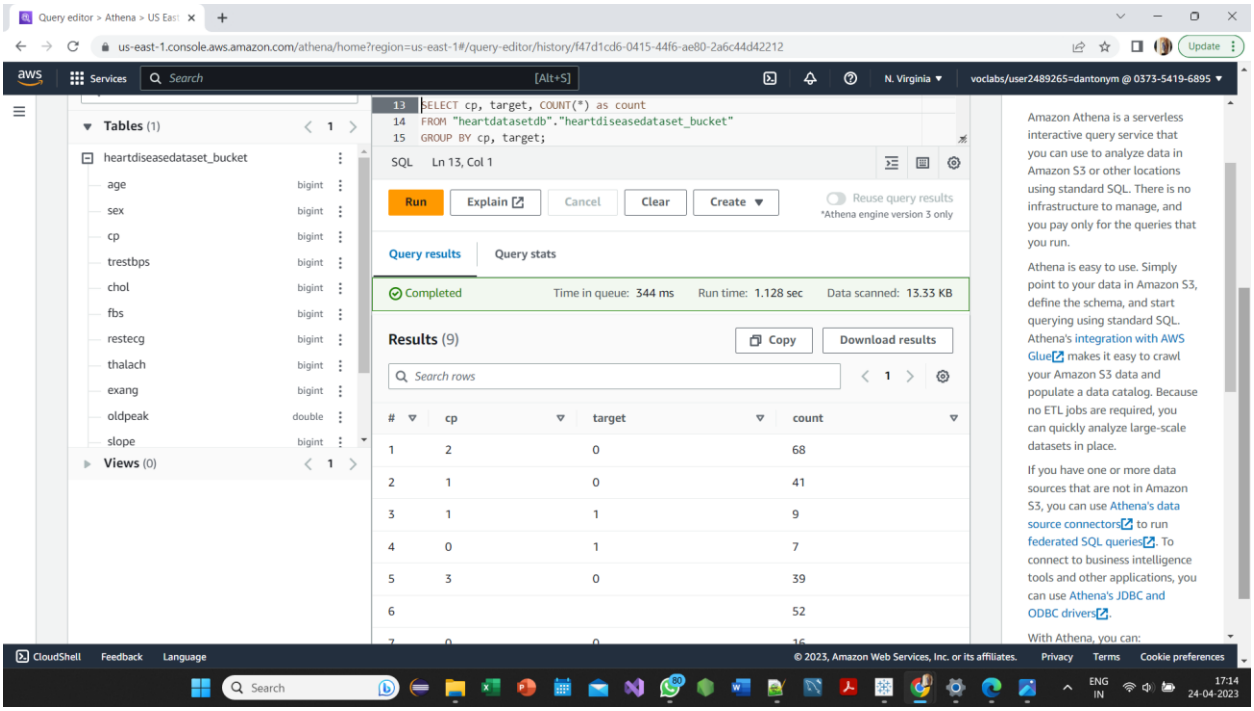


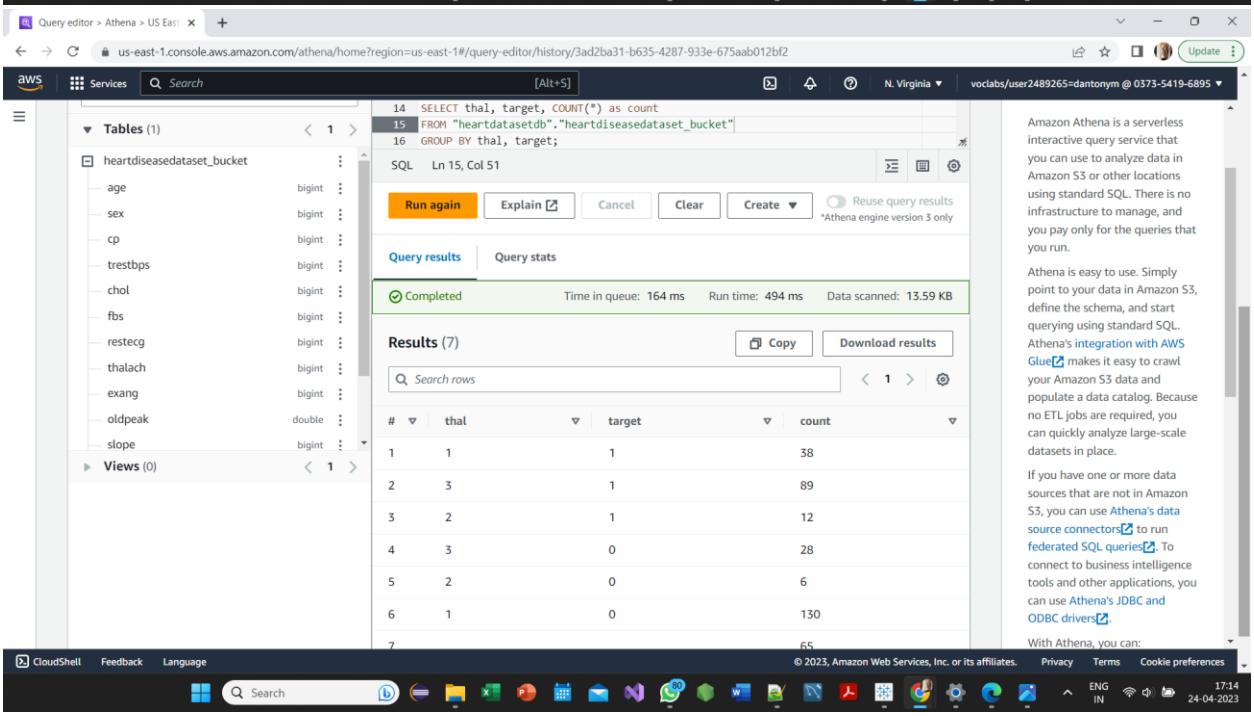

Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/1bb46615-1d78-4b30-877c-c5be0edf3d97
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
8
9
10 SELECT target,
11 COUNT(*) as count
12 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
13 GROUP BY target;
SQL Ln 10, Col 1
Run
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 169 ms
Run time: 474 ms
Data scanned: 12.34 KB
Results (3)
Copy
Download results
Search rows
target count
1 25
2 0 164
3 1 139
What is Amazon Athena?
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:06
24-04-2023

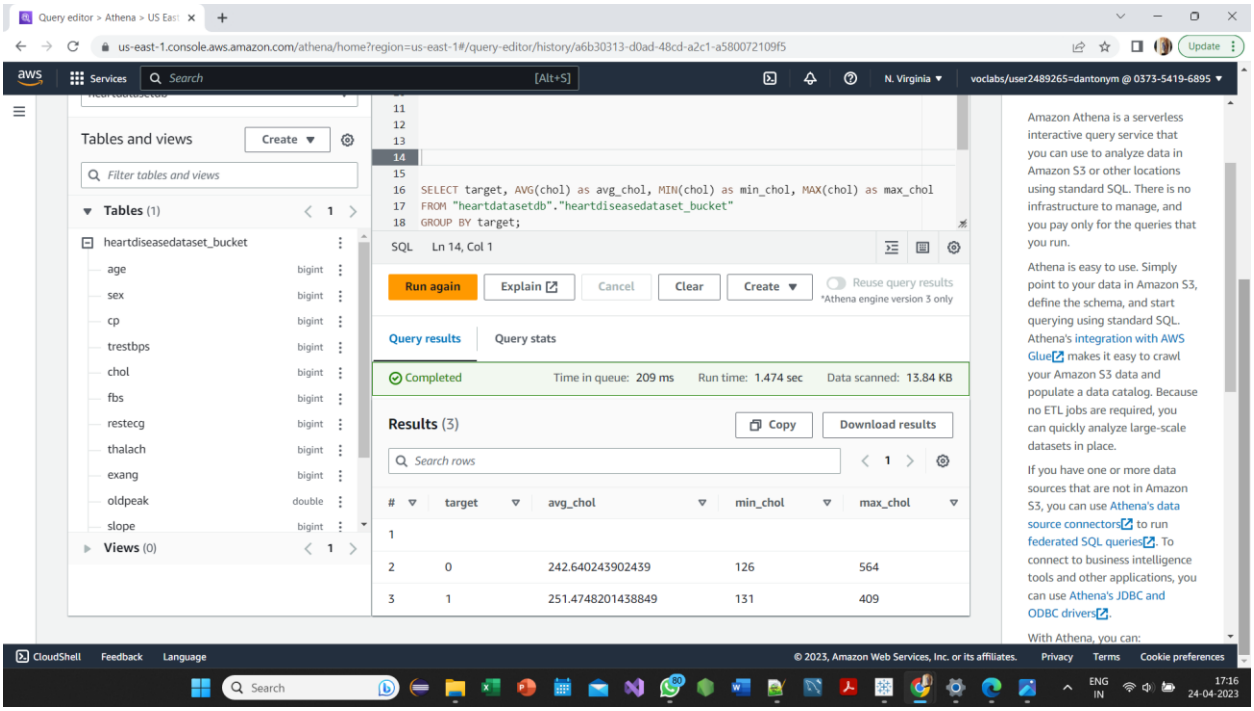

Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/c66434bf-8b36-4e11-aec3-a9a01413013f
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
8
9
10 SELECT target, AVG(age) as avg_age, MIN(age) as min_age, MAX(age) as max_age
11 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
12 GROUP BY target;
13
SQL Ln 11, Col 6
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 153 ms
Run time: 481 ms
Data scanned: 12.49 KB
Results (3)
Copy
Download results
Search rows
target avg_age min_age max_age
1 1 56.62589928057554 35 77
2
3 0 52.58536585365854 29 76
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:07
24-04-2023

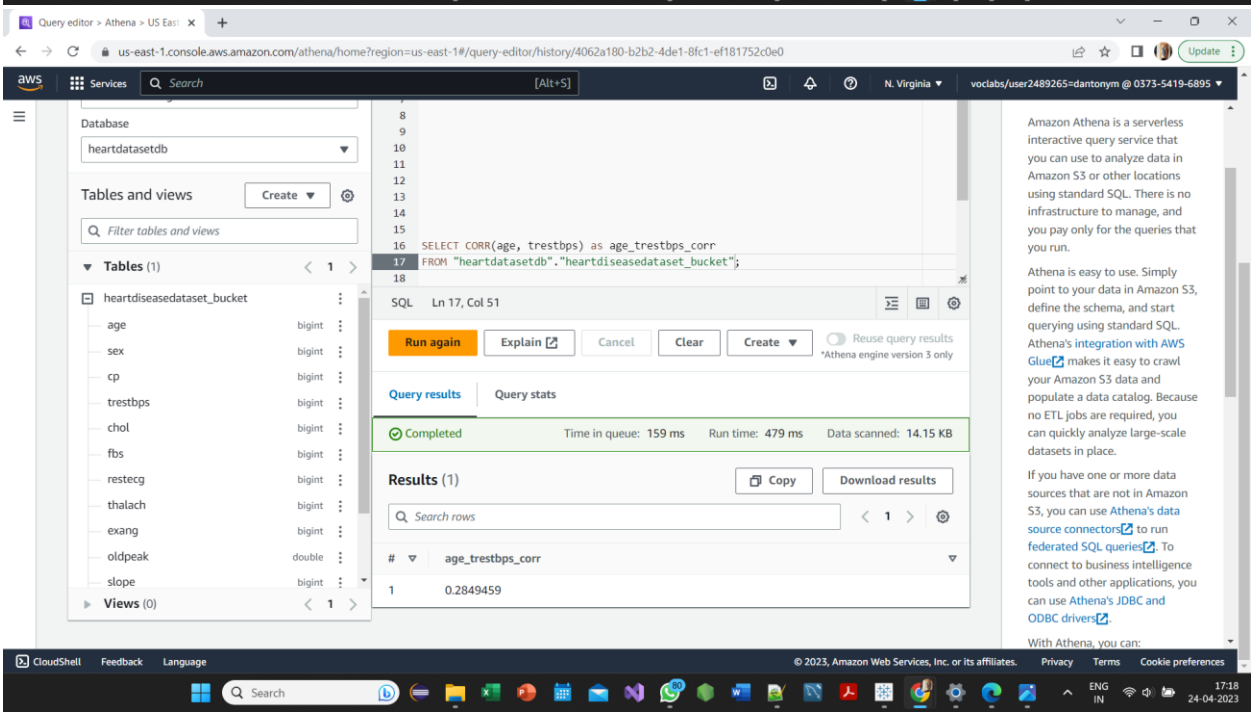

Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/9a7bd009-212c-4c49-9587-f36e224666c9
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
10 SELECT target, AVG(thalach) as avg_max_heart_rate, MIN(thalach) as
min_max_heart_rate, MAX(thalach) as max_max_heart_rate
11 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
12 GROUP BY target;
13
14
SQL Ln 14, Col 1
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 166 ms
Run time: 543 ms
Data scanned: 12.79 KB
Results (3)
Copy
Download results
Search rows
target avg_max_heart_rate min_max_heart_rate max_max_heart_rate
1 1 139.25899280575538 71 195
2
3 0 158.3780487804878 96 202
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:10
24-04-2023

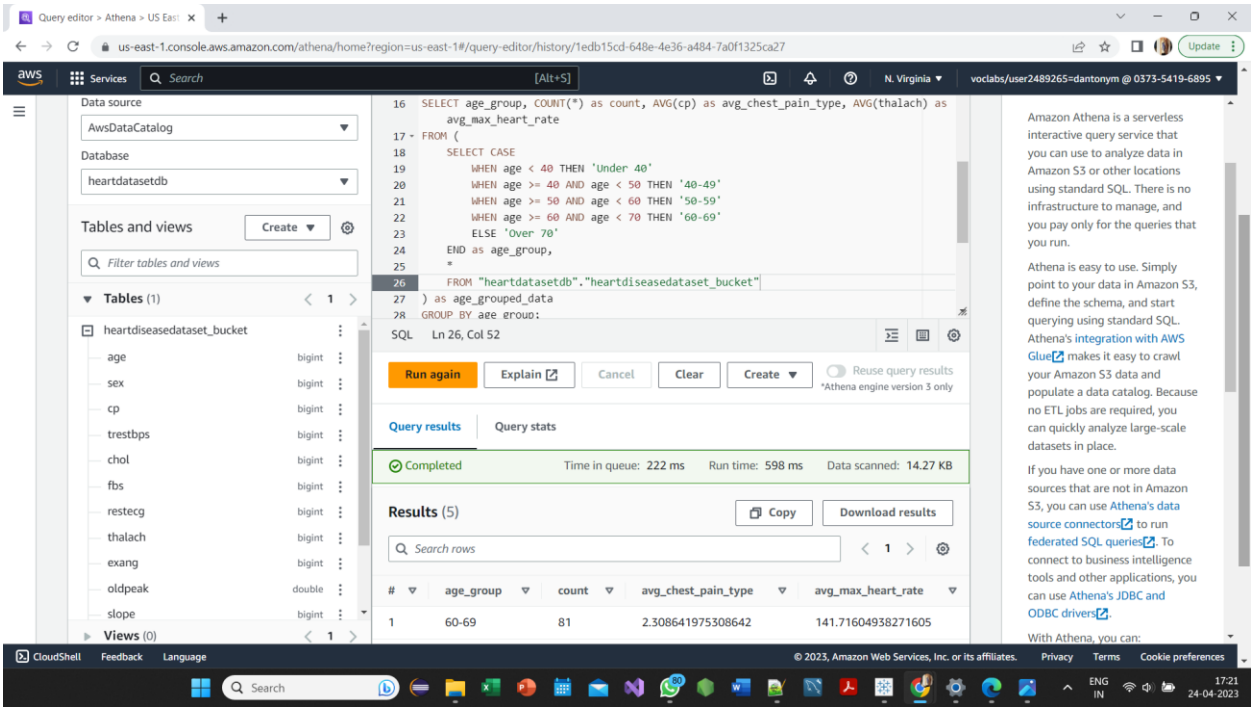

Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/eb6d4738-3374-4be3-9b25-997f77680e86
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Database
heartdatasetdb
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
6
7
8
9
10 SELECT target, COUNT(*) as count
11 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
12 WHERE fbs = 1
13 GROUP BY target;
14
15
SQL Ln 11, Col 51
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 164 ms
Run time: 568 ms
Data scanned: 13.19 KB
Results (2)
Copy
Download results
Search rows
target count
1 0 23
2 1 22
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:12
24-04-2023

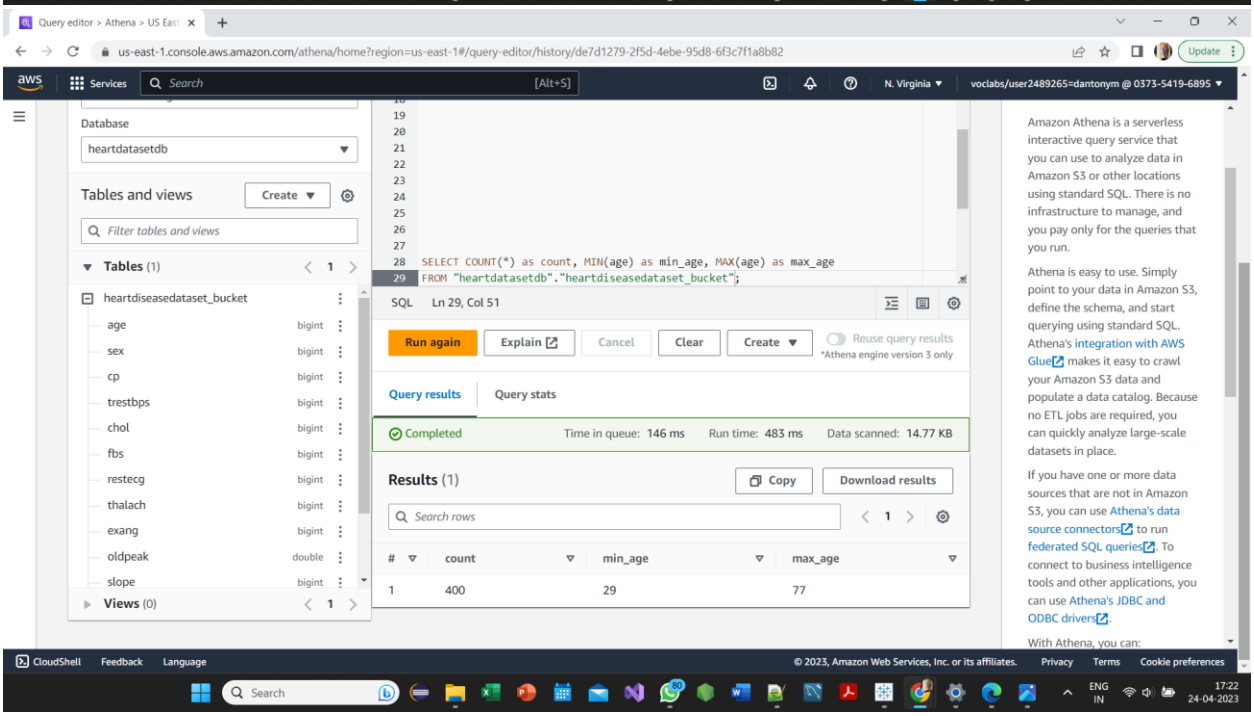

Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/f47d1cd6-0415-44f6-ae80-2a6c44d42212
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
13 SELECT cp, target, COUNT(*) as count
14 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
15 GROUP BY cp, target;
SQL Ln 13, Col 1
Run
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 344 ms
Run time: 1.128 sec
Data scanned: 13.33 KB
Results (9)
Copy
Download results
Search rows
cp target count
1 2 0 68
2 1 0 41
3 1 1 9
4 0 1 7
5 3 0 39
6 52
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:14
24-04-2023


Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/3ad2ba31-b635-4287-933e-675aab012bf2
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
14 SELECT thal, target, COUNT(*) as count
15 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
16 GROUP BY thal, target;
SQL Ln 15, Col 51
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 164 ms
Run time: 494 ms
Data scanned: 13.59 KB
Results (7)
Copy
Download results
Search rows
thal target count
1 1 1 38
2 3 1 89
3 2 1 12
4 3 0 28
5 2 0 6
6 1 0 130
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:14
24-04-2023


Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/a6b30313-d0ad-48cd-a2c1-a58007210 9f5
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
11
12
13
14
15
16 SELECT target, AVG(chol) as avg_chol, MIN(chol) as min_chol, MAX(chol) as max_chol
17 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
18 GROUP BY target;
SQL Ln 14, Col 1
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 209 ms
Run time: 1.474 sec
Data scanned: 13.84 KB
Results (3)
Copy
Download results
Search rows
target avg_chol min_chol max_chol
1
2 0 242.640243902439 126 564
3 1 251.4748201438849 131 409
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG IN
17:16
24-04-2023


Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/4062a180-b2b2-4de1-8fc1-ef181752c0e0
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Database
heartdatasetdb
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
8
9
10
11
12
13
14
15
16 SELECT CORR(age, trestbps) as age_trestbps_corr
17 FROM "heartdatasetdb"."heartdiseasedataset_bucket";
18
SQL Ln 17, Col 51
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 159 ms
Run time: 479 ms
Data scanned: 14.15 KB
Results (1)
Copy
Download results
Search rows
age_trestbps_corr
1 0.2849459
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG IN
17:18
24-04-2023


Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/1edb15cd-648e-4e36-a484-7a0f1325ca27
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Data source
AwsDataCatalog
Database
heartdatasetdb
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
16 SELECT age_group, COUNT(*) as count, AVG(cp) as avg_chest_pain_type, AVG(thalach) as
avg_max_heart_rate
17 FROM (
18 SELECT CASE
19 WHEN age < 40 THEN 'Under 40'
20 WHEN age >= 40 AND age < 50 THEN '40-49'
21 WHEN age >= 50 AND age < 60 THEN '50-59'
22 WHEN age >= 60 AND age < 70 THEN '60-69'
23 ELSE 'Over 70'
24 END as age_group,
25 *
26 FROM "heartdatasetdb"."heartdiseasedataset_bucket"
27) as age_grouped_data
28 GROUP BY age group;
SQL Ln 26, Col 52
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 222 ms
Run time: 598 ms
Data scanned: 14.27 KB
Results (5)
Copy
Download results
Search rows
age_group count avg_chest_pain_type avg_max_heart_rate
1 60-69 81 2.308641975308642 141.71604938271605
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
17:21
24-04-2023


Query editor > Athena > US East
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/de7d1279-2f5d-4ebe-95d8-6f3c7f1a8b82
Update
Services
Search
[Alt+S]
N. Virginia
voclabs/user2489265=dantonym @ 0373-5419-6895
Database
heartdatasetdb
Tables and views
Create
Filter tables and views
Tables (1)
heartdiseasedataset_bucket
age bigint
sex bigint
cp bigint
trestbps bigint
chol bigint
fbs bigint
restecg bigint
thalach bigint
exang bigint
oldpeak double
slope bigint
Views (0)
28 SELECT COUNT(*) as count, MIN(age) as min_age, MAX(age) as max_age
29 FROM "heartdatasetdb"."heartdiseasedataset_bucket";
SQL Ln 29, Col 51
Run again
Explain
Cancel
Clear
Create
Reuse query results
*Athena engine version 3 only
Query results
Query stats
Completed
Time in queue: 146 ms
Run time: 483 ms
Data scanned: 14.77 KB
Results (1)
Copy
Download results
Search rows
count min_age max_age
1 400 29 77
Amazon Athena is a serverless interactive query service that you can use to analyze data in Amazon S3 or other locations using standard SQL. There is no infrastructure to manage, and you pay only for the queries that you run.
Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL. Athena's integration with AWS Glue makes it easy to crawl your Amazon S3 data and populate a data catalog. Because no ETL jobs are required, you can quickly analyze large-scale datasets in place.
If you have one or more data sources that are not in Amazon S3, you can use Athena's data source connectors to run federated SQL queries. To connect to business intelligence tools and other applications, you can use Athena's JDBC and ODBC drivers.
With Athena, you can:
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
17:22
24-04-2023

SECOND CHOICE - FATHOMNET 2023

Shifting seas, shifting species: Out-of-sample detection in the deep ocean
https://www.kaggle.com/competitions/fathomnet-out-of-sample-detection

---

DATA UNDERSTANDING

Data Organization
According to the website where the data is originally sourced (https://fathomnet.org/fathomnet/#/) the dataset consists of 84,454 images. Each image has a varying number of sea creatures included, oftentimes having more than one. In the images, there are 290 unique organisms and as such there are 290 categories. The dataset also includes 20 "supercategories" that the 290 categories fall into.

According to the data's Kaggle page: *"The training and evaluation data are split across an 800-meter depth threshold: all training data is collected from 0-800 meters, evaluation data comes from the whole 0-1300 meter range".*

Below is a sample of the .csv file containing the image names and their labels/categories: *"Each line indicates an image by its id and a list of corresponding categories present in the frame.*

```
id, categories
4a7f2199-772d-486d-b8e2-b651246316b5, [1.0]
3bddedf6-4ff8-4e81-876a-564d2b03b364, "[1.0, 9.0, 11.0, 88.0]"
3f735021-f5de-4168-b139-74bf2859d12a, "[1.0, 37.0, 51.0, 119.0]"
130e185f-09c5-490c-8d08-641c4cbf6e54, "[1.0, 51.0, 119.0]"
```

Each category also belongs to one of 20 semantic supercategories:

```
['Worm', 'Feather star', 'Sea cucumber', 'Squat lobster', 'Fish',
        'Soft coral', 'Urchin', 'Sea star', 'Sea fan', 'Sea pen',
        'Barnacle', 'Eel', 'Glass sponge', 'Shrimp', 'Black coral',
        'Anemone', 'Sea spider', 'Gastropod', 'Crab', 'Stony coral']"
```

Downloading the Data
The method to download the dataset is provided on the Kaggle competition linked at the top of this document. They provide multiple files that are necessary to download the data. Once downloaded, the files will all come organized in a single directory and should not be changed.

The ***demo_download.ipynb*** file should be run to download the images, making a subdirectory in the downloaded folder.

Of the rest of the files, the most important for us would be the ***train.csv*** located in the ***multilabel_classification*** directory. This .csv file contains the IDs of the dataset's images and their corresponding labels, which we will need to utilize to train the model.

DASHBOARD

https://public.tableau.com/app/profile/kameron.nanthanolath/viz/6100Dashboard/Dashboard1?publish=yes

DATA PREPARATION

Since this dataset consists of images, our data preparation will be geared towards augmenting and modifying our images to help the model's learning process.

- Verify Data
    - Although the data comes from a reliable source, we should ensure that all the image samples are good for training. This will probably entail running through the dataset and checking for labeling errors.
- Data Augmentation
    - Many data augmentations exist to artificially increase the size of the dataset, which we should avoid in this situation where the dataset size is sufficiently large.
    - In this case, the goal is to classify organisms in deep-sea environments we should augment the data such that all images simulate a deep-sea environment to help the model learn through the deep-sea noise.
    - With enough time, it may also be useful to use a GAN to modify images to appear more deep-sea by training it to distinguish between deep-sea and not-deep-sea images
- Image Normalization
    - Scaling the pixel values in the images into the range [0, 1]. Exists in most Python ML libraries as a function (i.e. Pytorch's .ToTensor() function automatically normalizes the data)
- Image Resizing
    - Using the dashboard we created to analyze the data, we can see that the images have varying dimensions and as such should all be resized to the same dimensions.
- Embeddings
    - It may also be worthwhile to use a denoising autoencoder to compress the images into a smaller dimension vector to use as the model's input to improve training speeds.