PROJECT REPORT

Summary:
Airline on-time performance in United States of America. *Make a smart choice while booking your flight.*

Taking an airplane is one of the most important and efficient ways to travel. However, many travelers have experienced delayed flight. Which airline carriers delayed most often? Which airports have highest probability to make you wait for a long time? Also, which day of week and which month of year are better for your journey without severe delay?

The aim of this presentation is to produce a graphical summary of the airline performance data. Due to the large size of the data set, the author retrieved the recent 11 years' (1998-2008) data for analysis.

All the analyses are made from four different aspects:

- Best airlines to avoid delays.
- Top 25 Airports severely delayed.
- Best time of day/day of week/time of year to fly to minimize delays.
- And Cancellation counts on reasons, monthly basis and airlines.
- Recommendations in terms of the aspects above are made at the end.

Dataset:

Source website: http://stat-computing.org/dataexpo/2009/the-data.html

The data consists of flight arrival and departure details for all commercial flights within the USA, from October 1987 to April 2008. This is a large dataset: there are nearly 120 million records in total, and takes up 1.6 gigabytes of space compressed and 12 gigabytes when uncompressed.

20 years of data in a csv format:

1987, 1988, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008

Other supplement data:

carriers.csv – Listing of carrier codes with full names.
airports.csv – Describes the locations of US airports.
plane-data.csv – Individual plane specification and

Data Screenshot:

Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrie	FlightNum	TailNum	ActualElapse	CRSElapsedT	AirTime	ArrDelay	DepDelay	Origin	Dest	Distance	TaxiIn	TaxiOut
2008	1	3	4	2003	1955	2211	2225	WN	335	N712SW	128	150	116	-14	8	IAD	TPA	810	4	8
2008	1	3	4	754	735	1002	1000	WN	3231	N772SW	128	145	113	2	19	IAD	TPA	810	5	10
2008	1	3	4	628	620	804	750	WN	448	N428WN	96	90	76	14	8	IND	BWI	515	3	17
2008	1	3	4	926	930	1054	1100	WN	1746	N612SW	88	90	78	-6	-4	IND	BWI	515	3	7
2008	1	3	4	1829	1755	1959	1925	WN	3920	N464WN	90	90	77	34	34	IND	BWI	515	3	10
2008	1	3	4	1940	1915	2121	2110	WN	378	N726SW	101	115	87	11	25	IND	JAX	688	4	10
2008	1	3	4	1937	1830	2037	1940	WN	509	N763SW	240	250	230	57	67	IND	LAS	1591	3	7
2008	1	3	4	1039	1040	1132	1150	WN	535	N428WN	233	250	219	-18	-1	IND	LAS	1591	7	7
2008	1	3	4	617	615	652	650	WN	11	N689SW	95	95	70	2	2	IND	MCI	451	6	19
2008	1	3	4	1620	1620	1639	1655	WN	810	N648SW	79	95	70	-16	0	IND	MCI	451	3	6
2008	1	3	4	706	700	916	915	WN	100	N690SW	130	135	106	1	6	IND	MCO	828	5	19
2008	1	3	4	1644	1510	1845	1725	WN	1333	N334SW	121	135	107	80	94	IND	MCO	828	6	8
2008	1	3	4	1426	1430	1426	1425	WN	829	N476WN	60	55	39	1	-4	IND	MDW	162	9	12
2008	1	3	4	715	715	720	710	WN	1016	N765SW	65	55	37	10	0	IND	MDW	162	7	21
2008	1	3	4	1702	1700	1651	1655	WN	1827	N420WN	49	55	35	-4	2	IND	MDW	162	4	10
2008	1	3	4	1029	1020	1021	1010	WN	2272	N263WN	52	50	37	11	9	IND	MDW	162	6	9
2008	1	3	4	1452	1425	1640	1625	WN	675	N286WN	228	240	213	15	27	IND	PHX	1489	7	8
2008	1	3	4	754	745	940	955	WN	1144	N778SW	226	250	205	-15	9	IND	PHX	1489	5	16
2008	1	3	4	1323	1255	1526	1510	WN	4	N674AA	123	135	110	16	28	IND	TPA	838	4	9
2008	1	3	4	1416	1325	1512	1435	WN	54	N643SW	56	70	49	37	51	ISP	BWI	220	2	5
2008	1	3	4	706	705	807	810	WN	68	N497WN	61	65	51	-3	1	ISP	BWI	220	3	7
2008	1	3	4	1657	1625	1754	1735	WN	623	N724SW	57	70	47	19	32	ISP	BWI	220	5	5
2008	1	3	4	1900	1840	1956	1950	WN	717	N786SW	56	70	49	6	20	ISP	BWI	220	2	5
2008	1	3	4	1039	1030	1133	1140	WN	1244	N714CB	54	70	47	-7	9	ISP	BWI	220	2	5
2008	1	3	4	801	800	902	910	WN	2101	N222WN	61	70	53	-8	1	ISP	BWI	220	3	5
2008	1	3	4	1520	1455	1619	1605	WN	2553	N394SW	59	70	50	14	25	ISP	BWI	220	2	7
2008	1	3	4	1422	1255	1657	1610	WN	188	N215WN	155	195	143	47	87	ISP	FLL	1093	6	6
2008	1	3	4	1954	1925	2239	2235	WN	1754	N243WN	165	190	155	4	29	ISP	FLL	1093	3	7
2008	1	3	4	636	635	921	945	WN	2275	N454WN	165	190	147	-24	1	ISP	FLL	1093	5	13
2008	1	3	4	734	730	958	1020	WN	550	N712SW	324	350	314	-22	4	ISP	LAS	2283	2	8
2008	1	3	4	2107	1945	2334	2230	WN	362	N798SW	147	165	134	64	82	ISP	MCO	972	6	7
2008	1	3	4	1008	1005	1234	1255	WN	543	N736SA	146	170	135	-21	3	ISP	MCO	972	5	6
2008	1	3	4	712	710	953	1000	WN	1112	N795SW	161	170	142	-7	2	ISP	MCO	972	5	14
2008	1	3	4	1312	1300	1546	1550	WN	1397	N247WN	154	170	140	-4	12	ISP	MCO	972	7	7
2008	1	3	4	1449	1430	1715	1720	WN	3398	N707SA	146	170	134	-5	19	ISP	MCO	972	6	6
2008	1	3	4	1634	1555	1859	1845	WN	3480	N443WN	145	170	134	14	39	ISP	MCO	972	5	6
2008	1	3	4	831	830	935	955	WN	300	N753SW	124	145	112	-20	1	ISP	MDW	765	5	7
2008	1	3	4	1812	1650	1927	1815	WN	422	N779SW	135	145	118	72	82	ISP	MDW	765	6	11
2008	1	3	4	1127	1105	1235	1230	WN	1837	N704SW	128	145	114	5	22	ISP	MDW	765	9	5

Variable descriptions – Yearly File Columns

Name	Description
1	Year 1987-2008
2	Month 1-12
3	DayofMonth 1-31
4	DayOfWeek 1 (Monday) - 7 (Sunday)
5	DepTime actual departure time (local, hhmm)
6	CRSDepTime scheduled departure time (local, hhmm)
7	ArrTime actual arrival time (local, hhmm)
8	CRSArrTime scheduled arrival time (local, hhmm)
9	UniqueCarrier unique carrier code
10	FlightNum flight number
11	TailNum plane tail number
12	ActualElapsedTime in minutes
13	CRSElapsedTime in minutes
14	AirTime in minutes
15	ArrDelay arrival delay, in minutes
16	DepDelay departure delay, in minutes
17	Origin origin IATA airport code
18	Dest destination IATA airport code
19	Distance in miles
20	TaxiIn taxi in time, in minutes
21	TaxiOut taxi out time in minutes

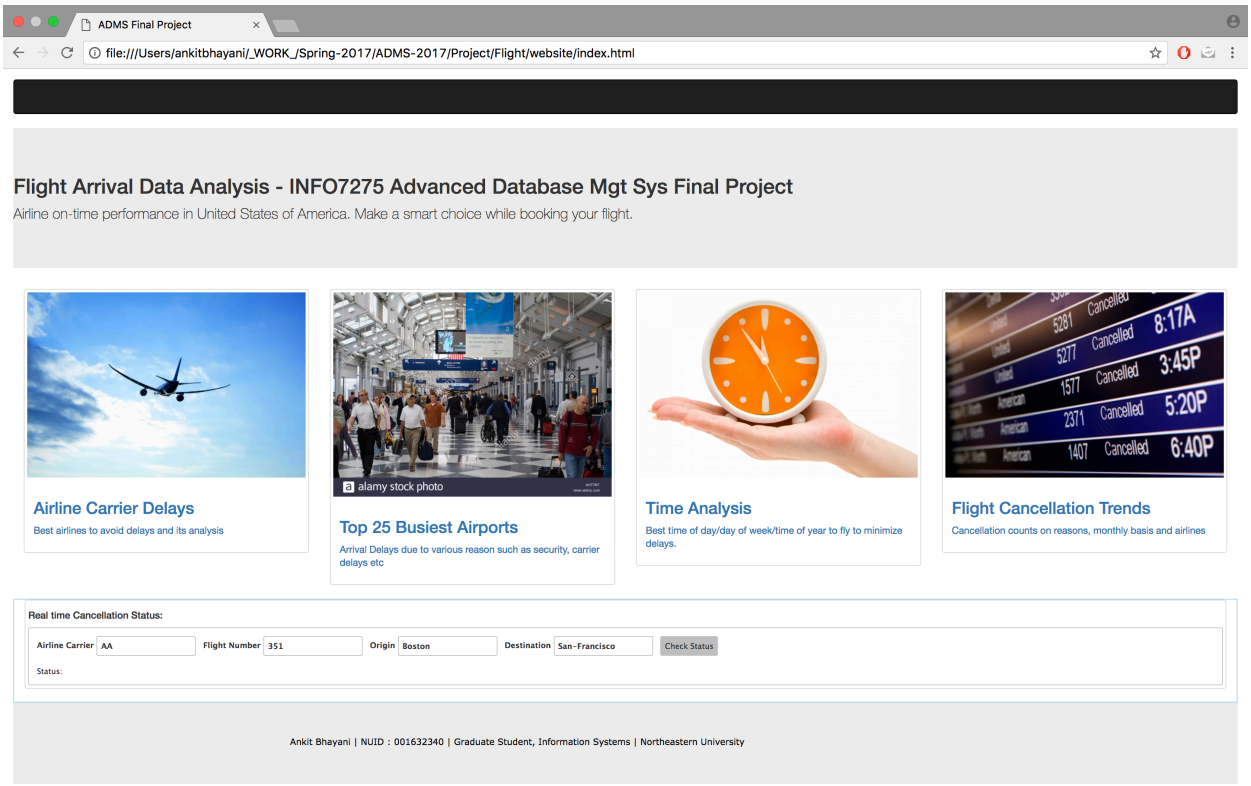
22 Cancelled was the flight cancelled?
23 CancellationCode reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
24 Diverted 1 = yes, 0 = no
25 CarrierDelay in minutes
26 WeatherDelay in minutes
27 NASDelay in minutes
28 SecurityDelay in minutes
29 LateAircraftDelay in minutes

Airports
Airports.csv describes the locations of US airports, with the fields:

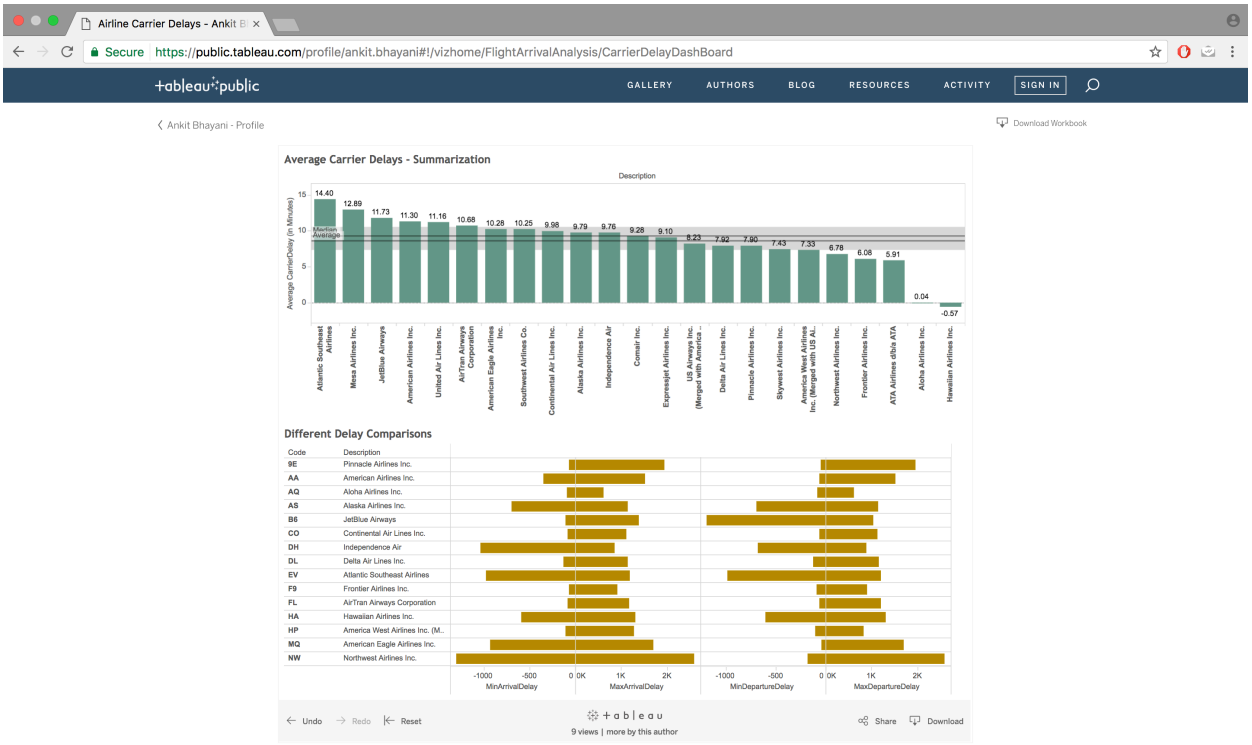
- iata: the international airport abbreviation code
- name of the airport
- city and country in which airport is located.
- lat and long: the latitude and longitude of the airport

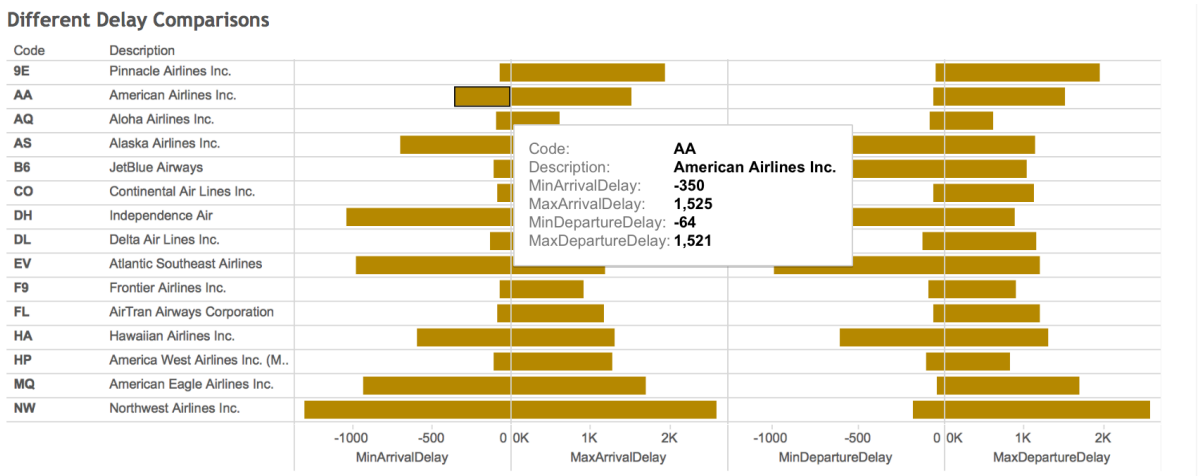
Carrier codes
Listing of carrier codes with full names

SCREENSHOTS and Summary of the Analysis performed:
Index.html – Main page of the website

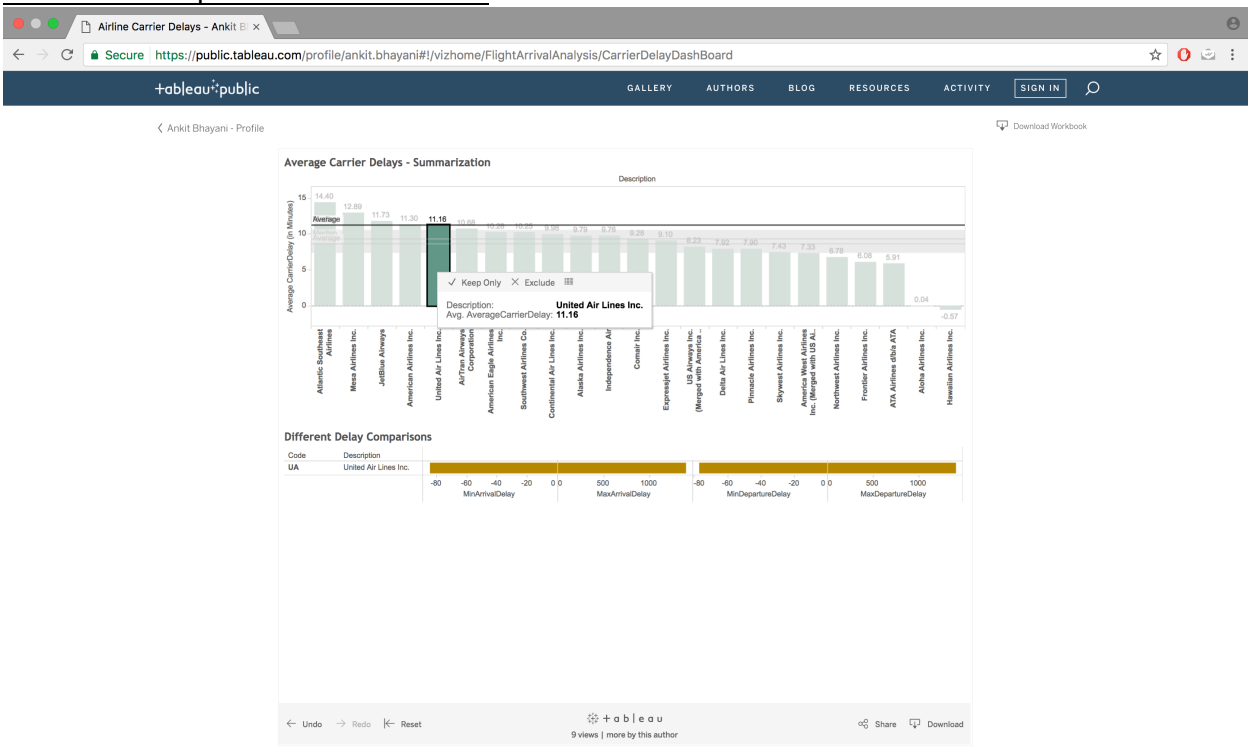


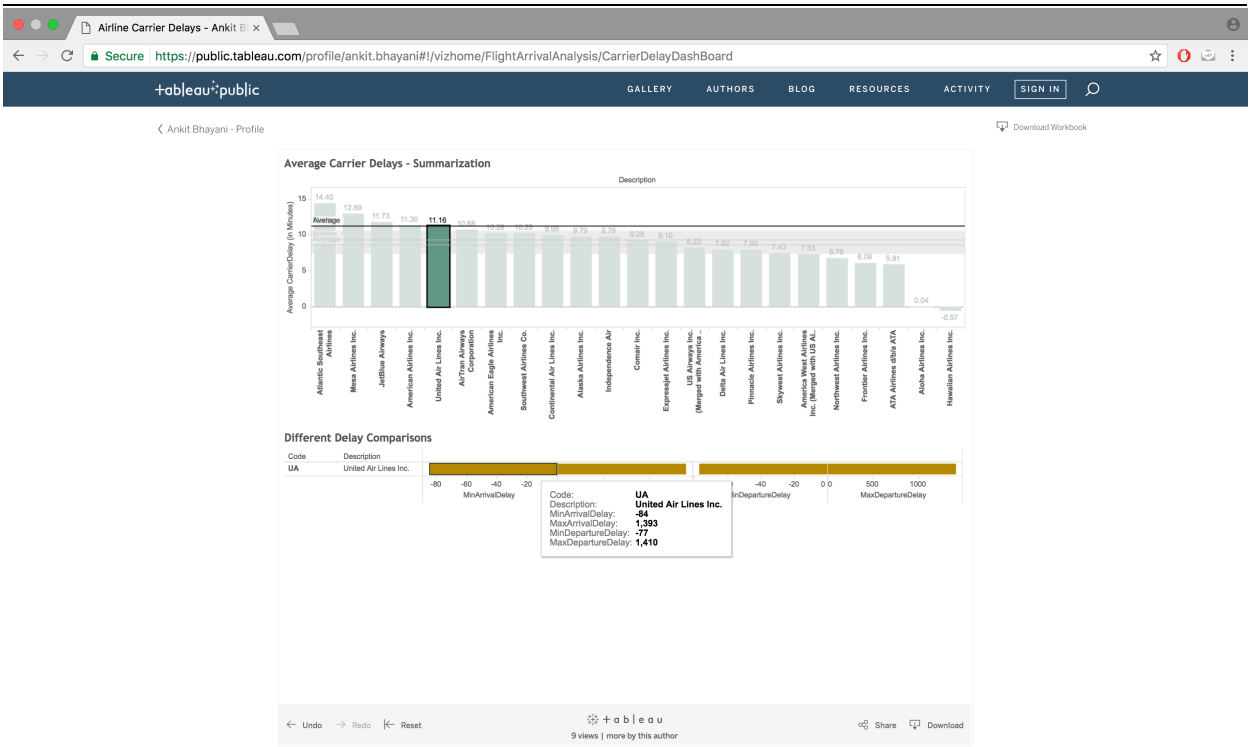
Best airlines to avoid delays: In this analysis, summarization pattern is used to calculate Average Carrier Delays and Min Arrival Delay, Min Arrival Delay, Min Departure Delay and Min Departure Delay (Aggregation Functions grouped on Unique Airline Carrier Code).
https://public.tableau.com/profile/ankit.bhayani#!/vizhome/FlightArrivalAnalysis/CarrierDelayDashBoard





Interactive Graphs on Tableau Public:



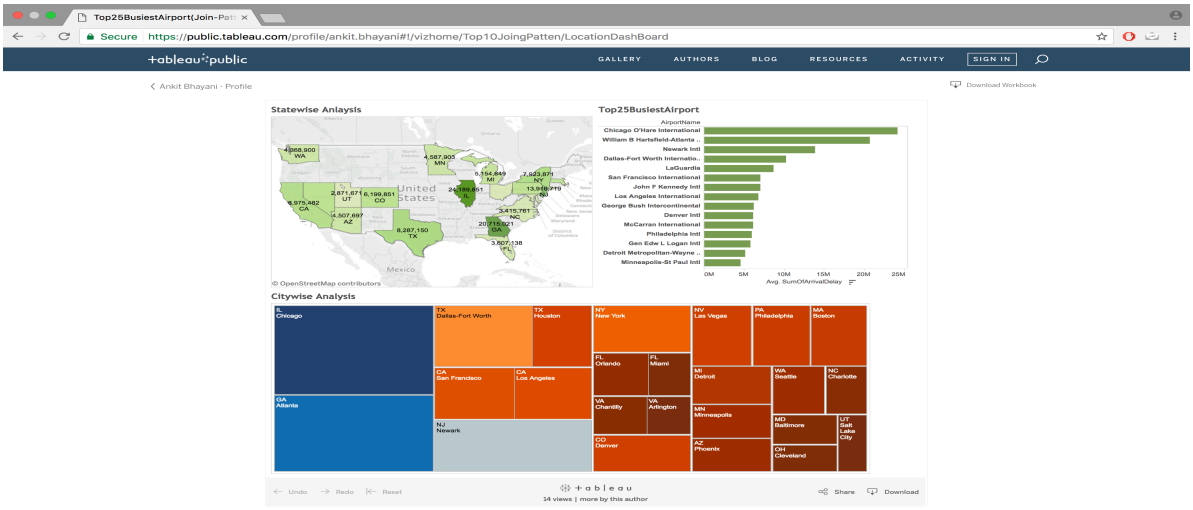


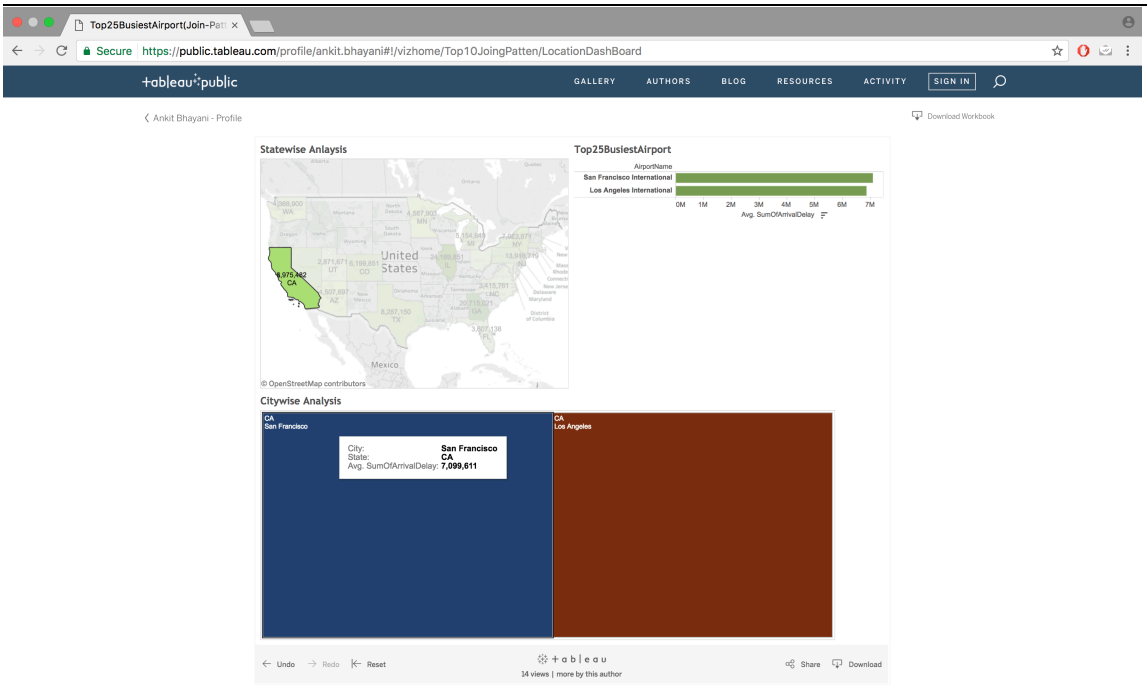
Top 25 Busiest Airport or severely Delayed: In this analysis, Top 10 and Join pattern is used to calculate analysis on State-wise, city wise and to find out the busiest airport.

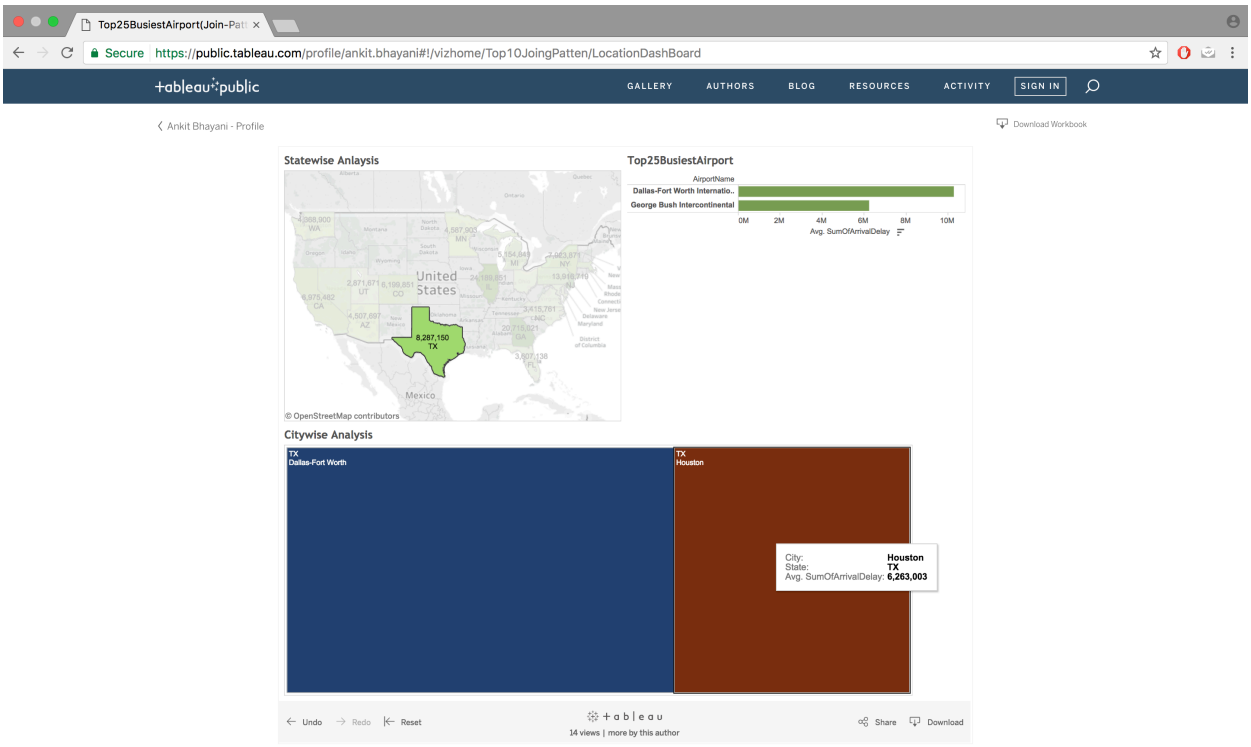
Tableau link:

https://public.tableau.com/profile/ankit.bhayani#!/vizhome/Top10JoingPatten/LocationDashBoard

Please see next page for screenshots


Top25BusiestAirport(Join-Patt
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/Top10JoingPatten/LocationDashBoard
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Statewise Anlaysis
Top25BusiestAirport
AirportName
Chicago O'Hare International
William B Hartsfield-Atlanta ..
Newark Intl
Dallas-Fort Worth Internatio..
LaGuardia
San Francisco International
John F Kennedy Intl
Los Angeles International
George Bush Intercontinental
Denver Intl
McCarran International
Philadelphia Intl
Gen Edw L. Logan Intl
Detroit Metropolitan-Wayne ..
Minneapolis-St Paul Intl
0M 5M 10M 15M 20M 25M
Avg. SumOfArrivalDelay
© OpenStreetMap contributors
Citywise Analysis
IL Chicago
TX Dallas-Fort Worth
TX Houston
NY New York
NV Las Vegas
PA Philadelphia
MA Boston
FL Orlando
FL Miami
CA San Francisco
CA Los Angeles
GA Atlanta
MI Detroit
WA Seattle
NC Charlotte
VA Chantilly
VA Arlington
MN Minneapolis
MD Baltimore
UT Salt Lake City
NJ Newark
CO Denver
AZ Phoenix
OH Cleveland
Undo Redo Reset
+ableau
14 views | more by this author
Share Download


Top25BusiestAirport(Join-Patt
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/Top10JoingPatten/LocationDashBoard
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Statewise Anlaysis
Top25BusiestAirport
AirportName
San Francisco International
Los Angeles International
0M 1M 2M 3M 4M 5M 6M 7M
Avg. SumOfArrivalDelay
© OpenStreetMap contributors
Citywise Analysis
CA San Francisco
CA Los Angeles
City: San Francisco
State: CA
Avg. SumOfArrivalDelay: 7,099,611
Undo Redo Reset
+ableau
14 views | more by this author
Share Download

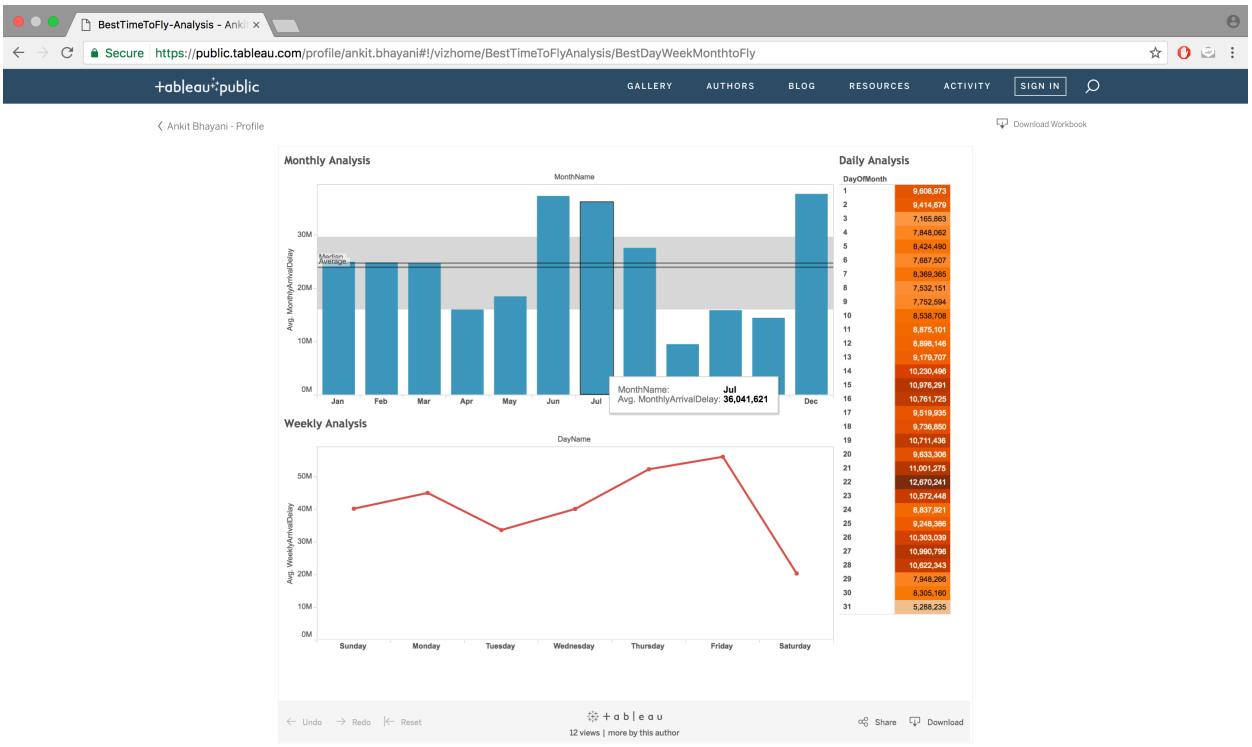

Top25BusiestAirport(Join-Pat
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/Top10JoingPatten/LocationDashBoard
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Statewise Aniaysis
United States
8,287,150
TX
Mexico
© OpenStreetMap contributors
Top25BusiestAirport
AirportName
Dallas-Fort Worth Internatio..
George Bush Intercontinental
0M 2M 4M 6M 8M 10M
Avg. SumOfArrivalDelay
Citywise Analysis
TX
Dallas-Fort Worth
TX
Houston
City: Houston
State: TX
Avg. SumOfArrivalDelay: 6,263,003
Undo Redo Reset
+ableau
14 views | more by this author
Share Download

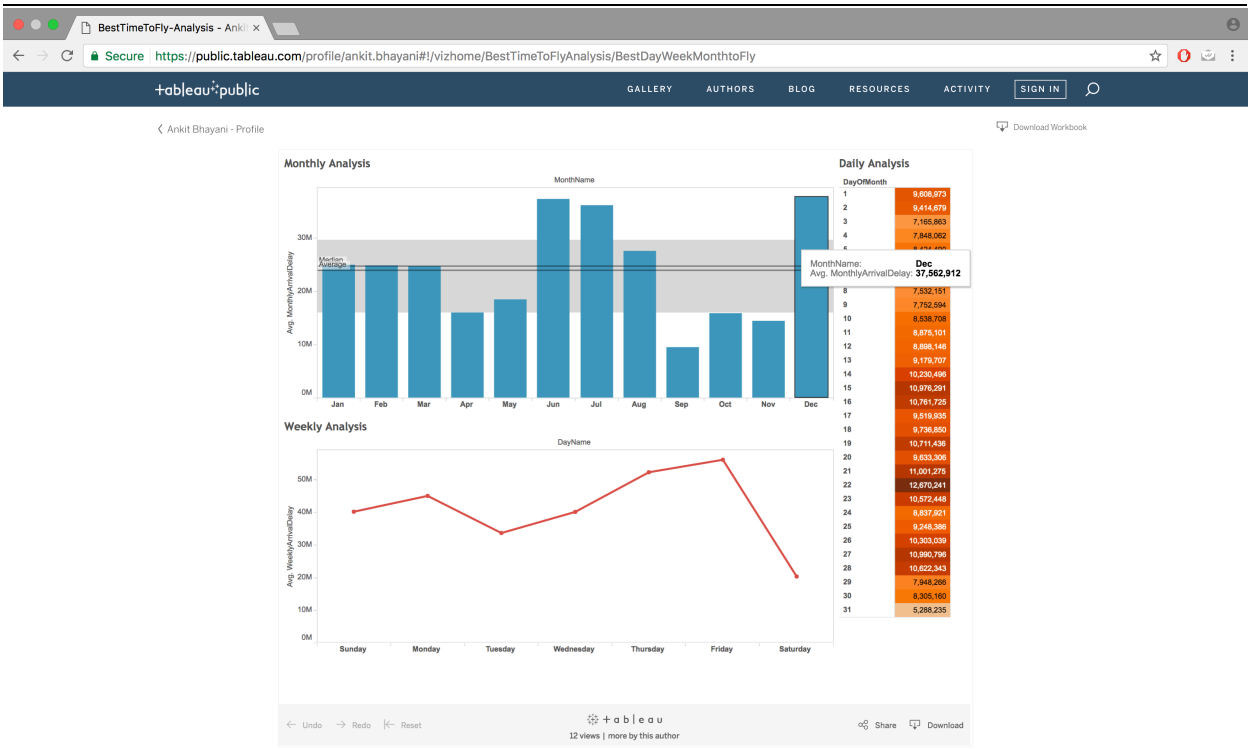
Best time of day/day of week/time of year to fly to minimize delays.: In this analysis, Partitioning pattern is used to calculate monthly, weekly and daily analysis.

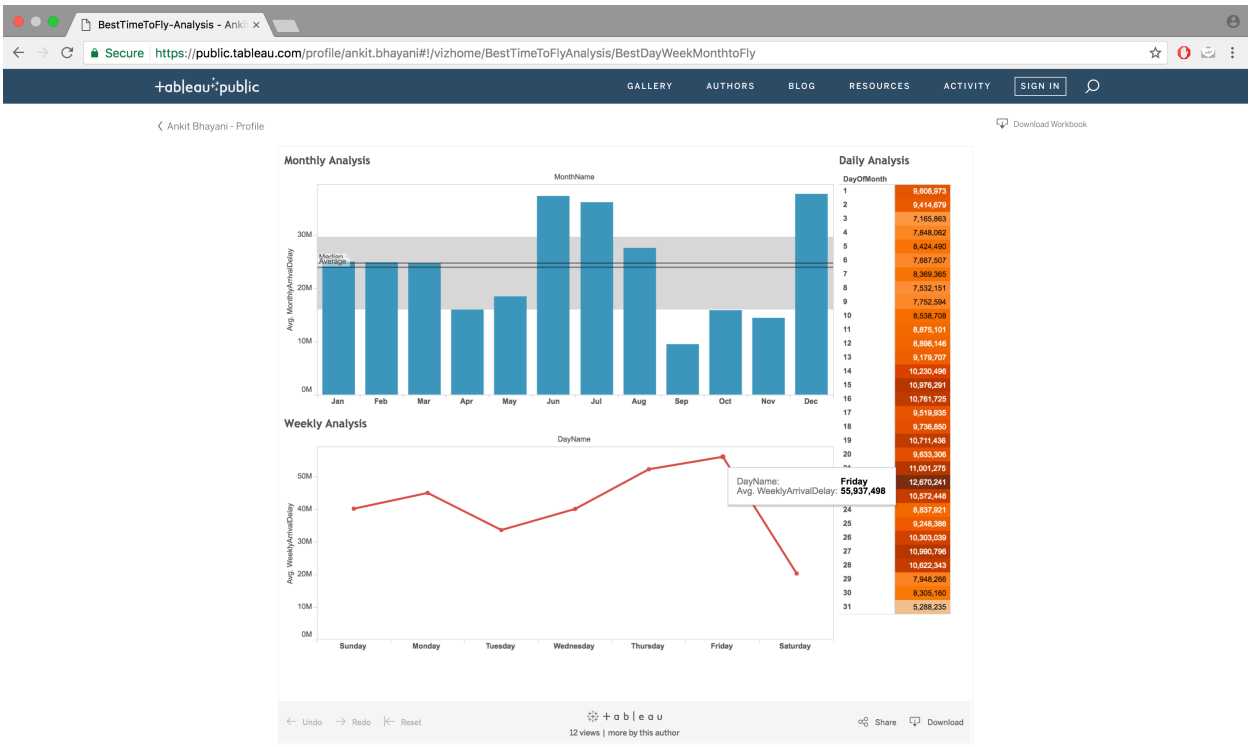
Tableau link:

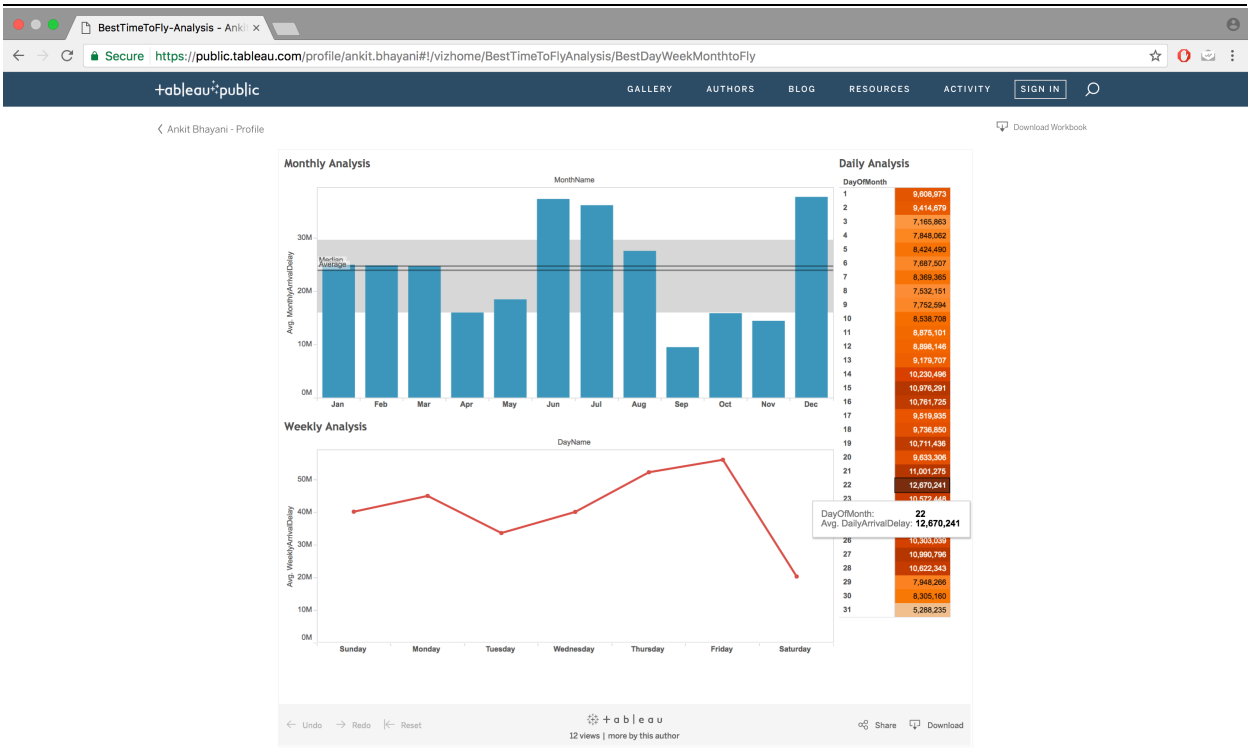
https://public.tableau.com/profile/ankit.bhayani#!/vizhome/BestTimeToFlyAnalysis/BestDayWeekMonthtoFly

Please see next page for screenshots


BestTimeToFly-Analysis - Anki
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/BestTimeToFlyAnalysis/BestDayWeekMonthtoFly
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Analysis
MonthName
Avg. MonthlyArrivalDelay
Median Average
MonthName: Jul
Avg. MonthlyArrivalDelay: 36,041,621
Daily Analysis
DayOfMonth
Weekly Analysis
DayName
Avg. WeeklyArrivalDelay
Sunday Monday Tuesday Wednesday Thursday Friday Saturday
Undo Redo Reset
+ableau
12 views | more by this author
Share Download


BestTimeToFly-Analysis - Anki
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/BestTimeToFlyAnalysis/BestDayWeekMonthtoFly
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Analysis
MonthName
Avg. MonthlyArrivalDelay
Median Average
MonthName: Dec
Avg. MonthlyArrivalDelay: 37,562,912
Daily Analysis
DayOfMonth
Weekly Analysis
DayName
Avg. WeeklyArrivalDelay
Sunday Monday Tuesday Wednesday Thursday Friday Saturday
Undo Redo Reset
+ableau
12 views | more by this author
Share Download


BestTimeToFly-Analysis - Anki
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/BestTimeToFlyAnalysis/BestDayWeekMonthtoFly
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Analysis
MonthName
Avg. MonthlyArrivalDelay
Median Average
30M
20M
10M
0M
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
Daily Analysis
DayOfMonth
1 9,608,973
2 9,414,679
3 7,165,863
4 7,848,062
5 8,424,490
6 7,687,507
7 8,369,365
8 7,532,151
9 7,752,594
10 8,538,708
11 8,875,101
12 8,898,146
13 9,179,707
14 10,230,496
15 10,976,291
16 10,761,725
17 9,519,935
18 9,736,850
19 10,711,436
20 9,633,306
21 11,001,275
22 12,670,241
23 10,572,448
24 8,837,921
25 9,248,386
26 10,303,039
27 10,990,796
28 10,622,343
29 7,948,266
30 8,305,160
31 5,288,235
Weekly Analysis
DayName
Avg. WeeklyArrivalDelay
50M
40M
30M
20M
10M
0M
Sunday Monday Tuesday Wednesday Thursday Friday Saturday
DayName:
Avg. WeeklyArrivalDelay: Friday 55,937,498
Undo Redo Reset
+ableau
12 views | more by this author
Share Download

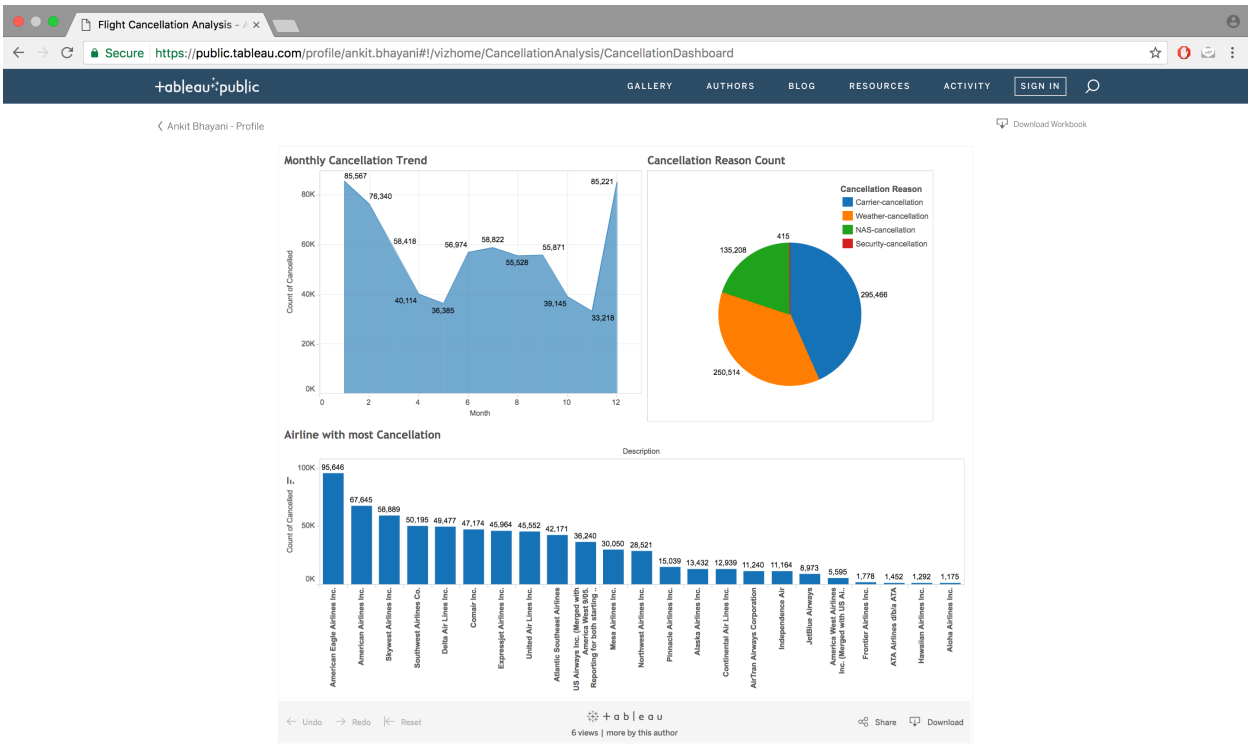

BestTimeToFly-Analysis - Anki
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/BestTimeToFlyAnalysis/BestDayWeekMonthtoFly
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Analysis
MonthName
Avg. MonthlyArrivalDelay
Median Average
30M
20M
10M
0M
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
Daily Analysis
DayOfMonth
1 9,608,973
2 9,414,679
3 7,165,863
4 7,848,062
5 8,424,490
6 7,687,507
7 8,369,365
8 7,532,151
9 7,752,594
10 8,538,708
11 8,875,101
12 8,898,146
13 9,179,707
14 10,230,496
15 10,976,291
16 10,761,725
17 9,519,935
18 9,736,850
19 10,711,436
20 9,633,306
21 11,001,275
22 12,670,241
23 10,572,448
24 8,837,921
25 9,248,386
26 10,303,039
27 10,990,796
28 10,622,343
29 7,948,266
30 8,305,160
31 5,288,235
Weekly Analysis
DayName
Avg. WeeklyArrivalDelay
50M
40M
30M
20M
10M
0M
Sunday Monday Tuesday Wednesday Thursday Friday Saturday
DayOfMonth:
Avg. DailyArrivalDelay: 22 12,670,241
Undo Redo Reset
+ableau
12 views | more by this author
Share Download

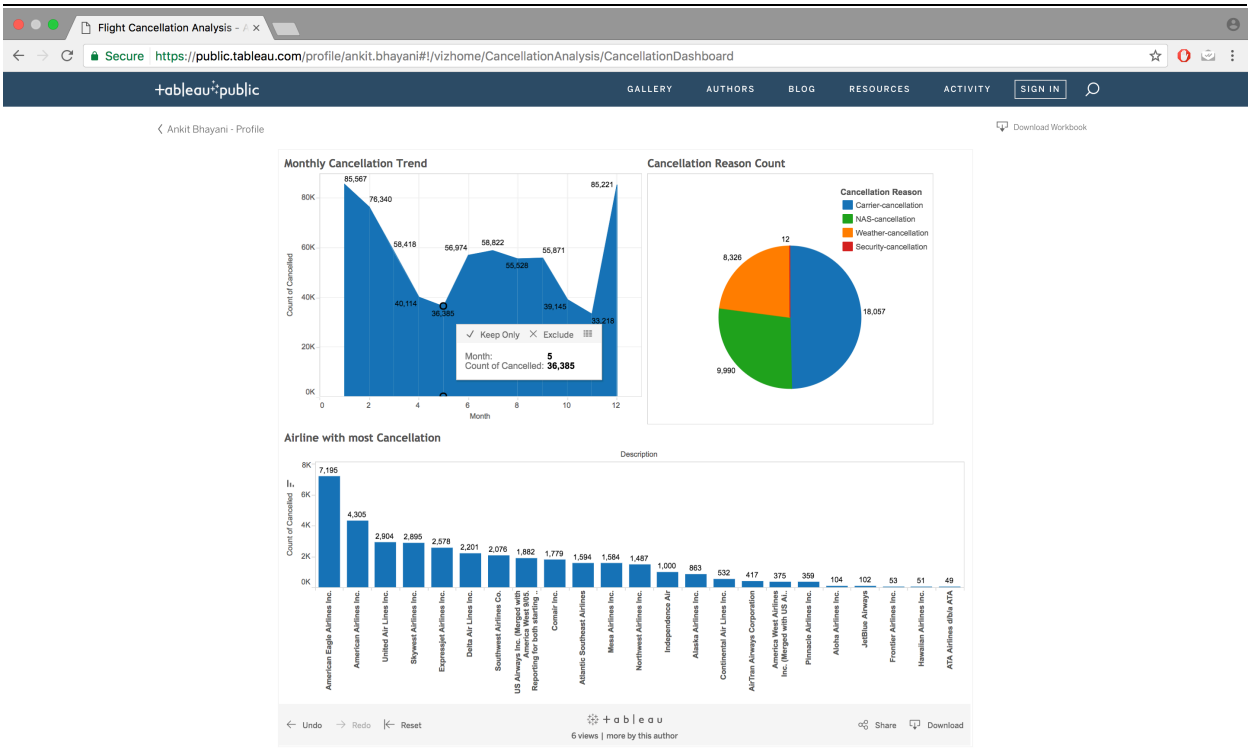
Flight Cancellation Trends.: In this analysis, Binning pattern is used to find out Cancellation counts on reasons, monthly basis and airlines.

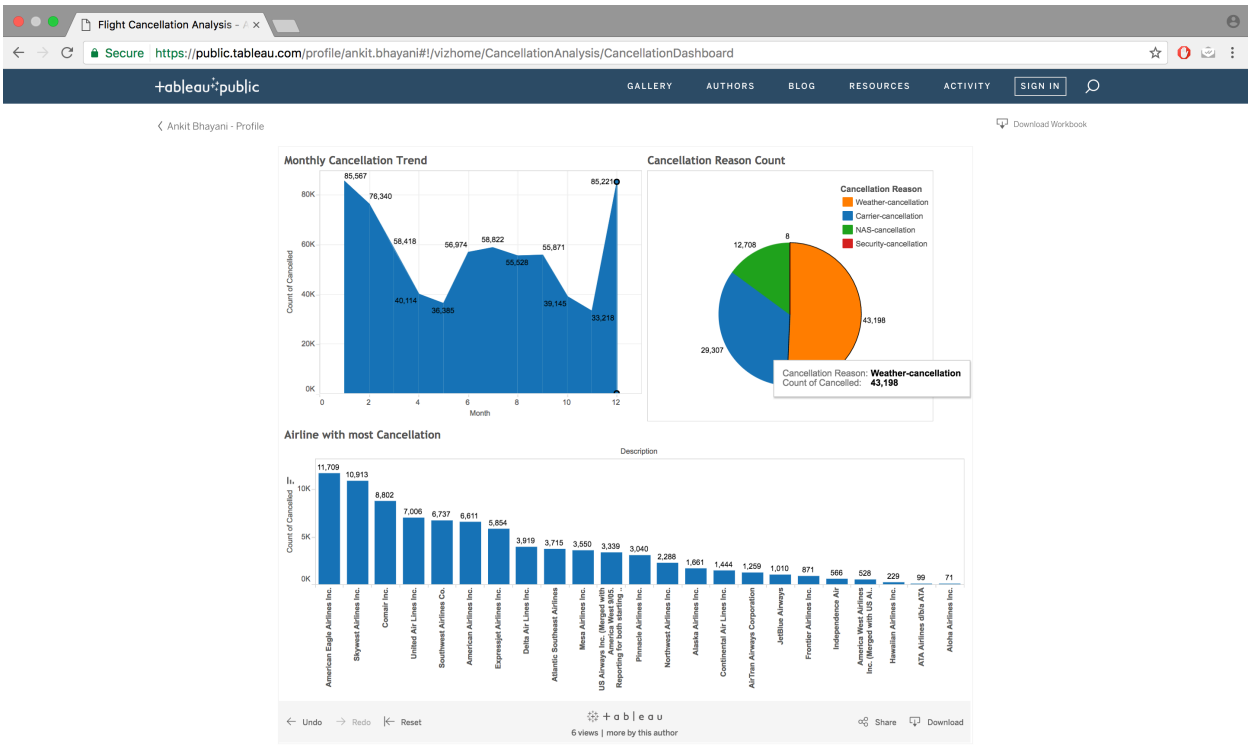
Tableau link:

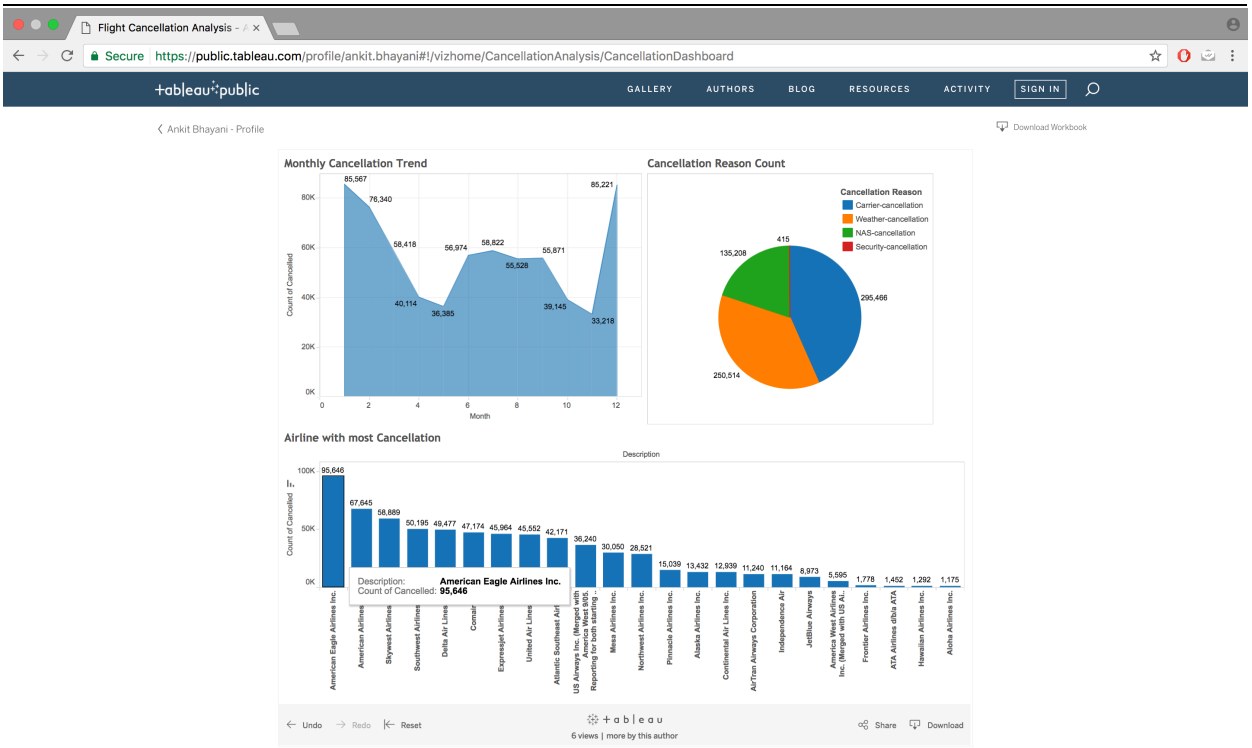
https://public.tableau.com/profile/ankit.bhayani#!/vizhome/CancellationAnalysis/CancellationDashboard

Please see next page for screenshots


Flight Cancellation Analysis
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/CancellationAnalysis/CancellationDashboard
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Cancellation Trend
Count of Cancelled
Month
85,567
78,340
58,418
40,114
36,385
56,974
58,822
55,528
55,871
39,145
85,221
33,218
Cancellation Reason Count
Cancellation Reason
Carrier-cancellation
Weather-cancellation
NAS-cancellation
Security-cancellation
135,208
415
295,466
250,514
Airline with most Cancellation
Description
Count of Cancelled
95,646
67,645
58,889
50,195
49,477
47,174
45,964
45,552
42,171
36,240
30,050
28,521
15,039
13,432
12,939
11,240
11,164
8,973
5,595
1,778
1,452
1,292
1,175
American Eagle Airlines Inc.
American Airlines Inc.
Skywest Airlines Inc.
Southwest Airlines Co.
Delta Air Lines Inc.
Comair Inc.
Expressjet Airlines Inc.
United Air Lines Inc.
Atlantic Southeast Airlines
US Airways Inc. (Merged with America West 9/05. Reporting for both starting
Mesa Airlines Inc.
Northwest Airlines Inc.
Pinnacle Airlines Inc.
Alaska Airlines Inc.
Continental Air Lines Inc.
AirTran Airways Corporation
Independence Air
JetBlue Airways
America West Airlines Inc. (Merged with US Al..
Frontier Airlines Inc.
ATA Airlines d/b/a ATA
Hawaiian Airlines Inc.
Aloha Airlines Inc.
Undo Redo Reset
+ableau
6 views | more by this author
Share Download

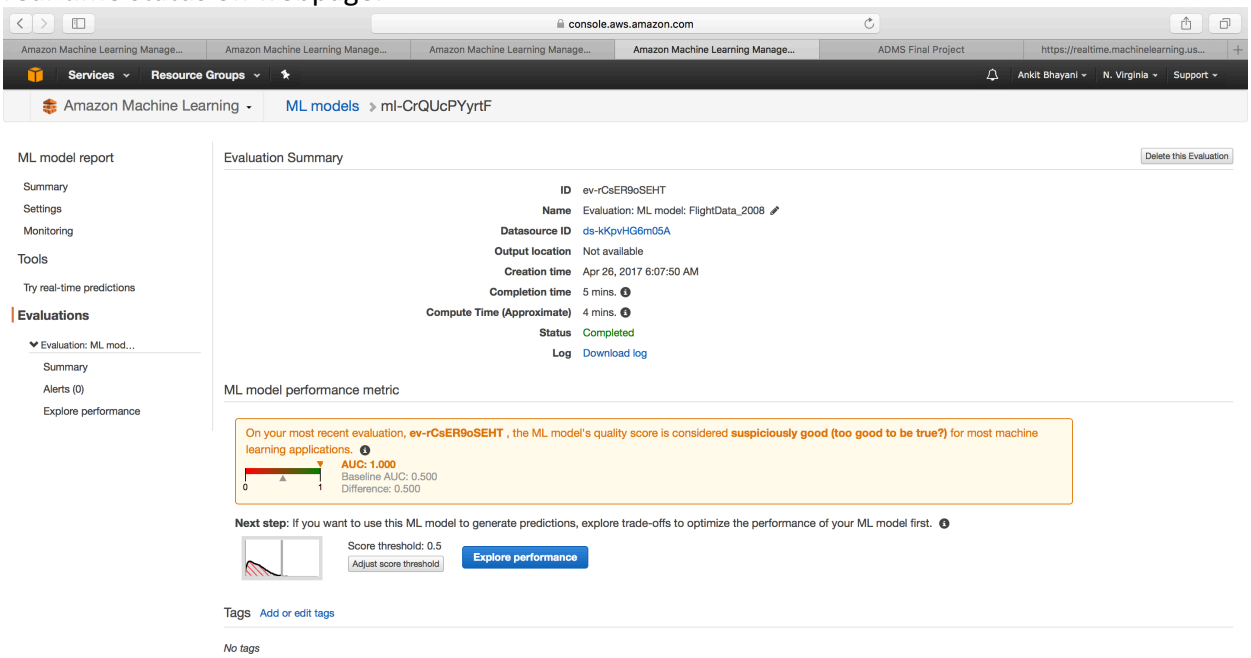

Flight Cancellation Analysis
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/CancellationAnalysis/CancellationDashboard
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Cancellation Trend
Count of Cancelled
Month
85,567
78,340
58,418
40,114
36,385
56,974
58,822
55,528
55,871
39,145
85,221
33,218
Keep Only
Exclude
Month: 5
Count of Cancelled: 36,385
Cancellation Reason Count
Cancellation Reason
Carrier-cancellation
NAS-cancellation
Weather-cancellation
Security-cancellation
8,326
12
18,057
9,990
Airline with most Cancellation
Description
Count of Cancelled
7,195
4,305
2,904
2,895
2,578
2,201
2,076
1,882
1,779
1,594
1,584
1,487
1,000
863
532
417
375
359
104
102
53
51
49
American Eagle Airlines Inc.
American Airlines Inc.
United Air Lines Inc.
Skywest Airlines Inc.
Expressjet Airlines Inc.
Delta Air Lines Inc.
Southwest Airlines Co.
US Airways Inc. (Merged with America West 9/05. Reporting for both starting
Comair Inc.
Atlantic Southeast Airlines
Mesa Airlines Inc.
Northwest Airlines Inc.
Independence Air
Alaska Airlines Inc.
Continental Air Lines Inc.
AirTran Airways Corporation
America West Airlines Inc. (Merged with US Al..
Pinnacle Airlines Inc.
Aloha Airlines Inc.
JetBlue Airways
Frontier Airlines Inc.
Hawaiian Airlines Inc.
ATA Airlines d/b/a ATA
Undo Redo Reset
+ableau
6 views | more by this author
Share Download

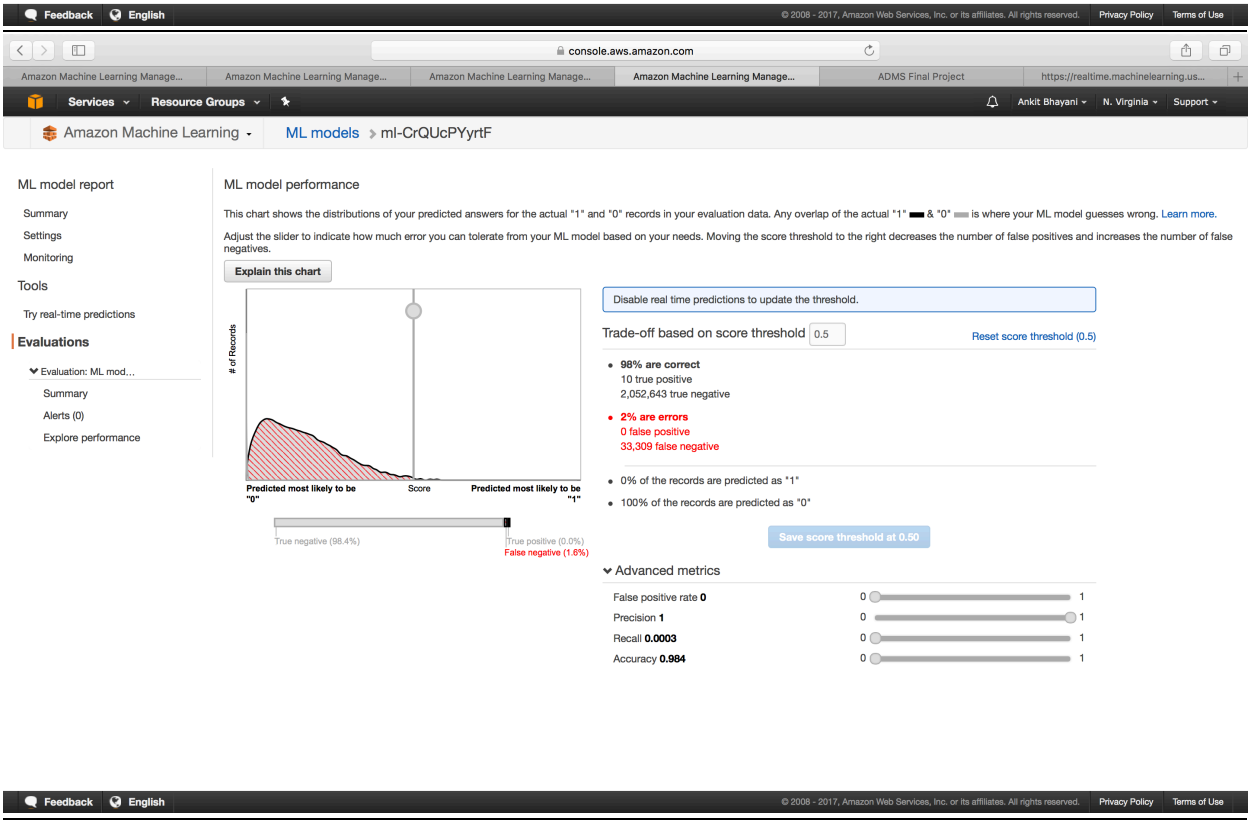

Flight Cancellation Analysis
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/CancellationAnalysis/CancellationDashboard
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Cancellation Trend
Count of Cancelled
Month
85,567
76,340
58,418
40,114
36,385
56,974
58,822
55,528
55,871
39,145
33,218
85,221
Cancellation Reason Count
Cancellation Reason
Weather-cancellation
Carrier-cancellation
NAS-cancellation
Security-cancellation
12,708
8
43,198
29,307
Cancellation Reason: Weather-cancellation
Count of Cancelled: 43,198
Airline with most Cancellation
Description
11,709
10,913
8,802
7,006
6,737
6,611
5,854
3,919
3,715
3,550
3,339
3,040
2,288
1,661
1,444
1,259
1,010
871
566
528
229
99
71
American Eagle Airlines Inc.
Skywest Airlines Inc.
Comair Inc.
United Air Lines Inc.
Southwest Airlines Co.
American Airlines Inc.
Expressjet Airlines Inc.
Delta Air Lines Inc.
Atlantic Southeast Airlines
Mesa Airlines Inc.
US Airways Inc. (Merged with America West 9/05. Reporting for both starting...
Pinnacle Airlines Inc.
Northwest Airlines Inc.
Alaska Airlines Inc.
Continental Air Lines Inc.
AirTran Airways Corporation
JetBlue Airways
Frontier Airlines Inc.
Independence Air
America West Airlines Inc. (Merged with US Al...
Hawaiian Airlines Inc.
ATA Airlines d/b/a ATA
Aloha Airlines Inc.
Undo Redo Reset
+ableau
6 views | more by this author
Share Download

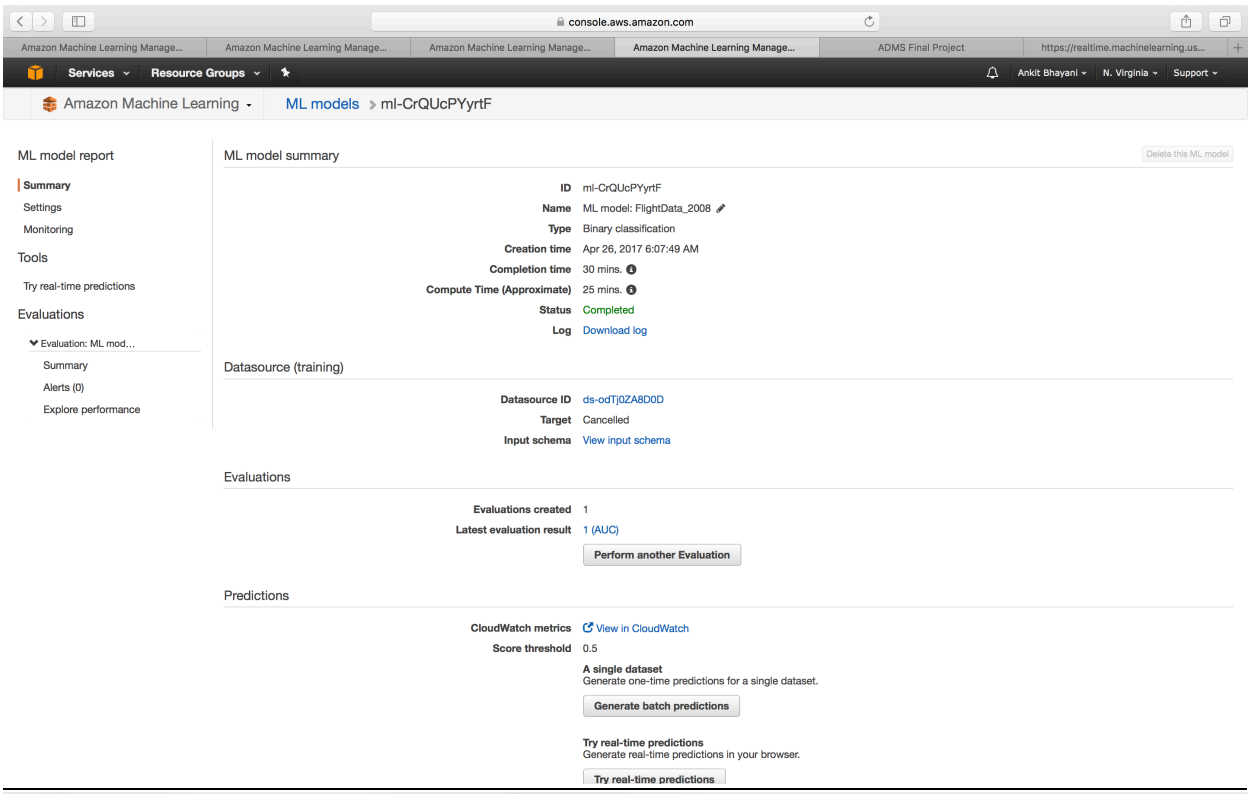

Flight Cancellation Analysis
Secure https://public.tableau.com/profile/ankit.bhayani#!/vizhome/CancellationAnalysis/CancellationDashboard
+ableau public
GALLERY AUTHORS BLOG RESOURCES ACTIVITY SIGN IN
Ankit Bhayani - Profile
Download Workbook
Monthly Cancellation Trend
Count of Cancelled
Month
85,567
76,340
58,418
40,114
36,385
56,974
58,822
55,528
55,871
39,145
33,218
85,221
Cancellation Reason Count
Cancellation Reason
Carrier-cancellation
Weather-cancellation
NAS-cancellation
Security-cancellation
135,208
415
295,466
250,514
Airline with most Cancellation
Description
95,646
67,645
58,889
50,195
49,477
47,174
45,964
45,552
42,171
36,240
30,050
28,521
15,039
13,432
12,939
11,240
11,164
8,973
5,595
1,778
1,452
1,292
1,175
Description: American Eagle Airlines Inc.
Count of Cancelled: 95,646
Undo Redo Reset
+ableau
6 views | more by this author
Share Download

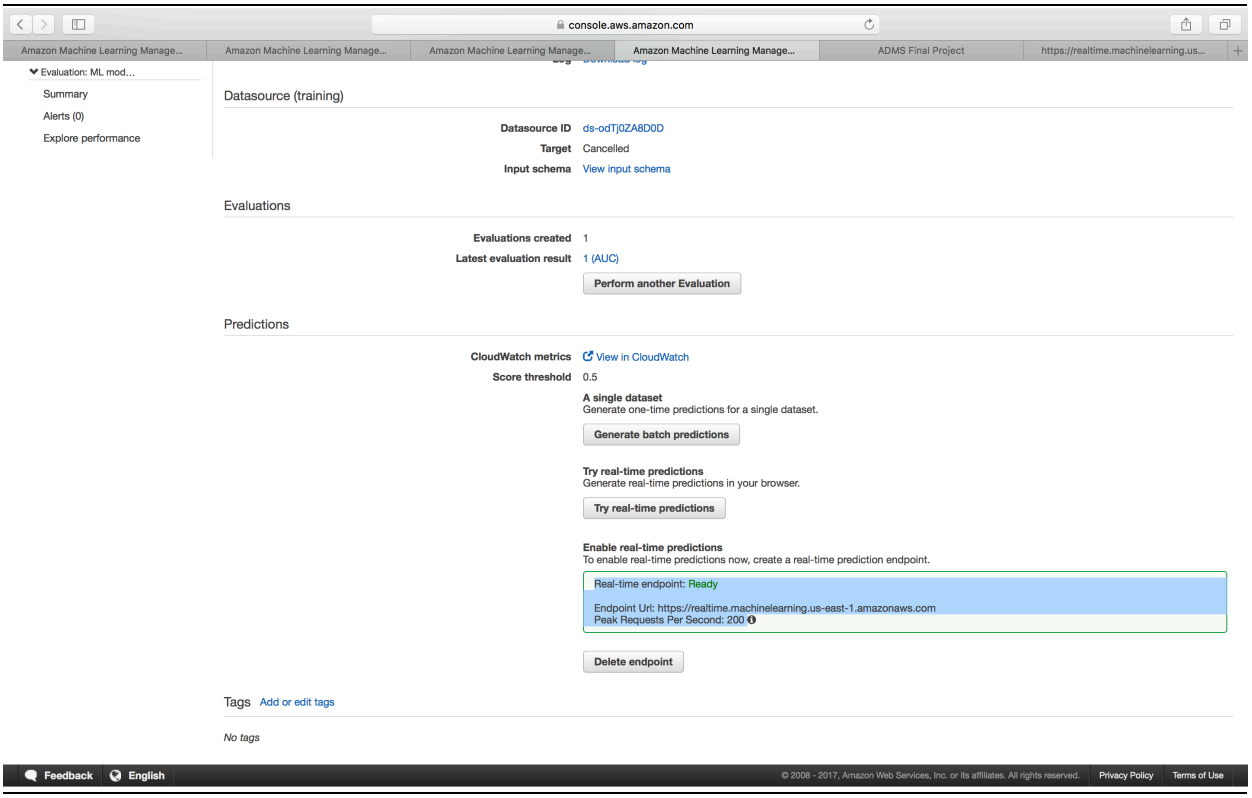
Real Time Flight Cancelled Status Check:

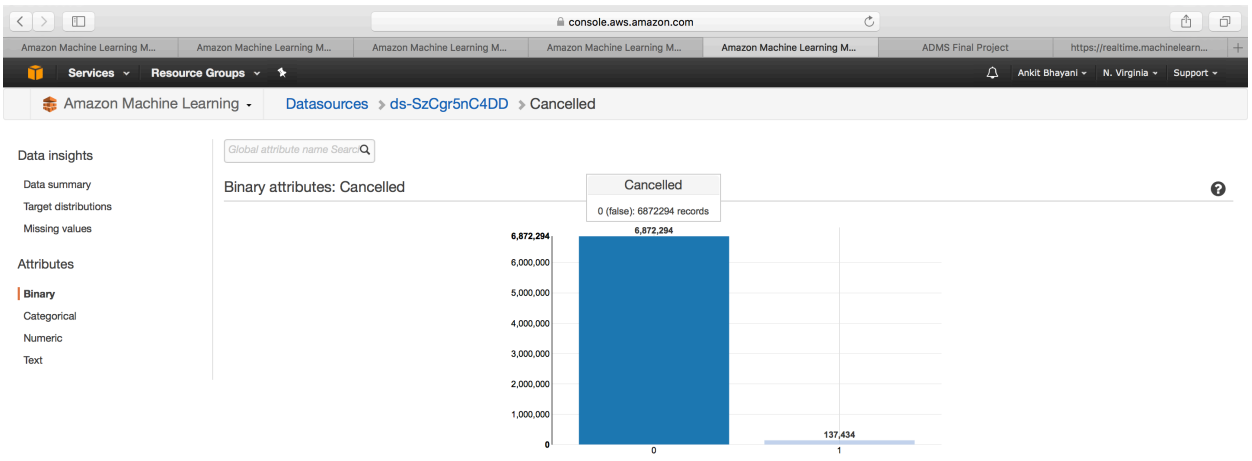
Binary Classification (Machine learning) through Amazon Machine learning and displaying the real-time status on webpage.

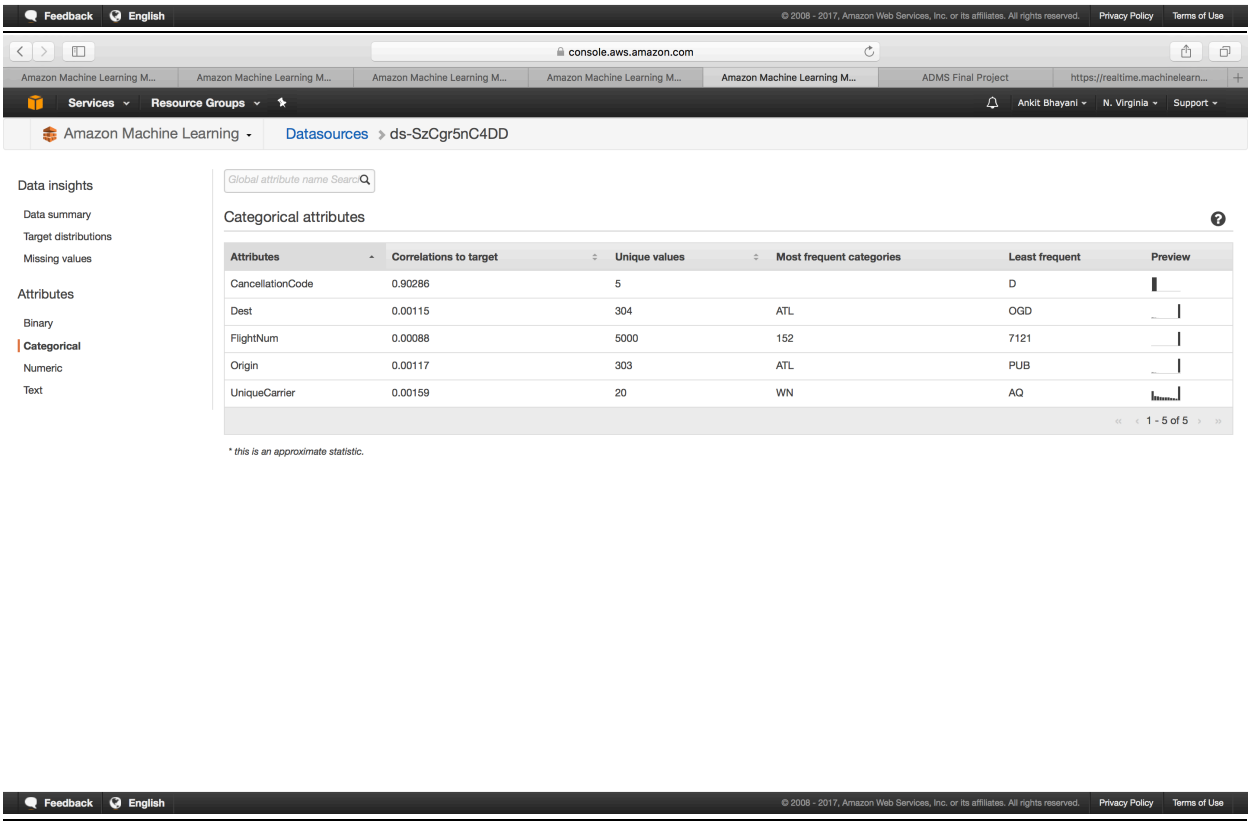





console.aws.amazon.com
Amazon Machine Learning Manage...
Amazon Machine Learning Manage...
Amazon Machine Learning Manage...
Amazon Machine Learning Manage...
ADMS Final Project
https://realtime.machinelearning.us...
Services
Resource Groups
Ankit Bhayani
N. Virginia
Support
Amazon Machine Learning
ML models
ml-CrQUcPYyrtF
ML model report
Summary
Settings
Monitoring
Tools
Try real-time predictions
Evaluations
Evaluation: ML mod...
Summary
Alerts (0)
Explore performance
ML model summary
Delete this ML model
ID ml-CrQUcPYyrtF
Name ML model: FlightData_2008
Type Binary classification
Creation time Apr 26, 2017 6:07:49 AM
Completion time 30 mins.
Compute Time (Approximate) 25 mins.
Status Completed
Log Download log
Datasource (training)
Datasource ID ds-odTj0ZA8D0D
Target Cancelled
Input schema View input schema
Evaluations
Evaluations created 1
Latest evaluation result 1 (AUC)
Perform another Evaluation
Predictions
CloudWatch metrics View in CloudWatch
Score threshold 0.5
A single dataset
Generate one-time predictions for a single dataset.
Generate batch predictions
Try real-time predictions
Generate real-time predictions in your browser.
Try real-time predictions


console.aws.amazon.com
Amazon Machine Learning Manage...
Amazon Machine Learning Manage...
Amazon Machine Learning Manage...
Amazon Machine Learning Manage...
ADMS Final Project
https://realtime.machinelearning.us...
Evaluation: ML mod...
Summary
Alerts (0)
Explore performance
Datasource (training)
Datasource ID ds-odTj0ZA8D0D
Target Cancelled
Input schema View input schema
Evaluations
Evaluations created 1
Latest evaluation result 1 (AUC)
Perform another Evaluation
Predictions
CloudWatch metrics View in CloudWatch
Score threshold 0.5
A single dataset
Generate one-time predictions for a single dataset.
Generate batch predictions
Try real-time predictions
Generate real-time predictions in your browser.
Try real-time predictions
Enable real-time predictions
To enable real-time predictions now, create a real-time prediction endpoint.
Real-time endpoint: Ready
Endpoint Url: https://realtime.machinelearning.us-east-1.amazonaws.com
Peak Requests Per Second: 200
Delete endpoint
Tags Add or edit tags
No tags
Feedback
English
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use


console.aws.amazon.com
Services
Resource Groups
Ankit Bhayani
N. Virginia
Support
Amazon Machine Learning
Datasources
ds-SzCgr5nC4DD
Cancelled
Data insights
Data summary
Target distributions
Missing values
Attributes
Binary
Categorical
Numeric
Text
Global attribute name Search
Binary attributes: Cancelled
Cancelled
0 (false): 6872294 records
6,872,294
6,000,000
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
137,434
0
1


Feedback
English
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
console.aws.amazon.com
Services
Resource Groups
Ankit Bhayani
N. Virginia
Support
Amazon Machine Learning
Datasources
ds-SzCgr5nC4DD
Data insights
Data summary
Target distributions
Missing values
Attributes
Binary
Categorical
Numeric
Text
Global attribute name Search
Categorical attributes
Attributes
Correlations to target
Unique values
Most frequent categories
Least frequent
Preview
CancellationCode 0.90286 5 D
Dest 0.00115 304 ATL OGD
FlightNum 0.00088 5000 152 7121
Origin 0.00117 303 ATL PUB
UniqueCarrier 0.00159 20 WN AQ
1 - 5 of 5
* this is an approximate statistic.
Feedback
English
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

CONCLUSION:

1. Southwest airline is more likely to have a departure delay.
2. Friday has largest probability of delayed flights.
3. December, June and July have largest percentage of delayed flights.
4. Among 50 busiest airports of US, Atlanta, Philadelphia and Chicago-ORD, these three airports have over 85% delayed flights. Ten airports' percentages of delay are over 80%.
5. The travelers should avoid taking Southwest airline's flights, avoid taking off on Friday or in December and avoid choosing larger airports such as Atlanta, Philadelphia and Chicago-ORD in order to reduce the probability to experience delayed flights.

CODE:

Summarization.java

```
package finalproject.summarization;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class Summarization {

	public static class Summarization_Mapper extends Mapper<Object,
Text, Text, CompositeKeyWritable>{

		private Text UniqueCarrier = new Text();

		@Override
		protected void map(Object key, Text value, Mapper<Object,
Text, Text, CompositeKeyWritable>.Context context)
				throws IOException, InterruptedException {

			if (!value.toString().contains("UniqueCarrier")) {
			String[] input = value.toString().split(",");
```

```
                if(!input[14].equalsIgnoreCase("NA") &&
!input[15].equalsIgnoreCase("NA")){

                double carrierDelay = 0;
                if(input[15].equalsIgnoreCase("NA")) carrierDelay=0.0;
                else carrierDelay=Double.parseDouble(input[15]);

                CompositeKeyWritable outTuple= new
CompositeKeyWritable(Integer.parseInt(input[14]),Integer.parseInt(inpu
t[14]),

     Integer.parseInt(input[15]),Integer.parseInt(input[15]),carrierDe
lay,1);
                        UniqueCarrier.set(input[8]);
                        context.write(UniqueCarrier, outTuple);
                }
             }
            }

     }


     public static class Summarization_Reducer extends Reducer<Text,
CompositeKeyWritable, Text, CompositeKeyWritable> {

           @Override
           protected void reduce(Text key,
Iterable<CompositeKeyWritable> values,Reducer<Text,
CompositeKeyWritable, Text, CompositeKeyWritable>.Context context)
                                throws IOException,
InterruptedException {
                    int minArrDel = Integer.MAX_VALUE;
                int maxArrDel = Integer.MIN_VALUE;
                    int minDepDel = Integer.MAX_VALUE;
                int maxDepDel = Integer.MIN_VALUE;
                int count = 0;
                   double sum = 0;


                for (CompositeKeyWritable val : values) {

                   if (val.getArrMinDelay() < minArrDel)
                         minArrDel = val.getArrMinDelay();
```

```
                    if (val.getArrMaxDelay() > maxArrDel)
                        maxArrDel = val.getArrMaxDelay();

                    if (val.getDepMinDelay() < minDepDel)
                        minDepDel = val.getDepMinDelay();

                    if (val.getDepMaxDelay() > maxDepDel)
                        maxDepDel = val.getDepMaxDelay();

                    sum += val.getAverage()*val.getCount();
                        count +=val.getCount();

                }

                context.write(key,new
CompositeKeyWritable(minArrDel, maxArrDel, minDepDel,
maxDepDel,(sum/count),count));
            }

      }


    public static void main( String[] args ){
        System.out.println( "Hello World!" );

        Configuration conf = new Configuration();

           try {
                Job job = Job.getInstance(conf, "stock price high");
                job.setJarByClass(Summarization.class);
                job.setMapperClass(Summarization_Mapper.class);
                job.setCombinerClass(Summarization_Reducer.class);
                job.setReducerClass(Summarization_Reducer.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(CompositeKeyWritable.class);

                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new
Path(args[1]));

                System.exit(job.waitForCompletion(true) ? 0 : 1);
           } catch (Exception e) {
                e.printStackTrace();
           }
```

```
    }
}
```

CompositeKeyWritable.java

```
package finalproject.summarization;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;

import org.apache.hadoop.io.Writable;

public class CompositeKeyWritable implements Writable{

    private int arrMinDelay;
    private int arrMaxDelay;
    private int depMinDelay;
    private int depMaxDelay;
      private double average;
      private long count;

      public CompositeKeyWritable(){}

    public CompositeKeyWritable(int arrMinDelay, int arrMaxDelay, int
depMinDelay,
            int depMaxDelay, double average, int count) {
        this.arrMinDelay = arrMinDelay;
        this.arrMaxDelay = arrMaxDelay;
        this.depMinDelay = depMinDelay;
        this.depMaxDelay = depMaxDelay;
        this.average = average;
        this.count = count;
    }

      public int getArrMinDelay() {
            return arrMinDelay;
      }

      public void setArrMinDelay(int arrMinDelay) {
            this.arrMinDelay = arrMinDelay;
      }
```

```
public int getArrMaxDelay() {
	return arrMaxDelay;
}

public void setArrMaxDelay(int arrMaxDelay) {
	this.arrMaxDelay = arrMaxDelay;
}

public int getDepMinDelay() {
	return depMinDelay;
}

public void setDepMinDelay(int depMinDelay) {
	this.depMinDelay = depMinDelay;
}

public int getDepMaxDelay() {
	return depMaxDelay;
}

public void setDepMaxDelay(int depMaxDelay) {
	this.depMaxDelay = depMaxDelay;
}

public double getAverage() {
	return average;
}

public void setAverage(double average) {
	this.average = average;
}

public long getCount() {
	return count;
}

public void setCount(long count) {
	this.count = count;
}

public void readFields(DataInput in) throws IOException {
	arrMinDelay = in.readInt();
	arrMaxDelay = in.readInt();
```

```
            depMinDelay = in.readInt();
            depMaxDelay = in.readInt();
            average = in.readDouble();
            count = in.readLong();

      }

      public void write(DataOutput out) throws IOException {
            out.writeInt(arrMinDelay);
            out.writeInt(arrMaxDelay);
            out.writeInt(depMinDelay);
            out.writeInt(depMaxDelay);
            out.writeDouble(average);
            out.writeLong(count);
      }

      @Override
      public String toString() {
            return (arrMinDelay +" " +arrMaxDelay +" "+ depMinDelay +"
" + depMaxDelay +" " + average +" "+ count );
      }


}
```

Top10BusiestAirport.java

```
/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package finalproject.top10;

import java.io.IOException;
import java.util.ArrayList;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
```

```
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.MultipleInputs;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author ankitbhayani
 */
public class Top10BusiestAirport {

    public static class Map1 extends Mapper<LongWritable, Text, Text,
IntWritable>{


        @Override
        protected void map(LongWritable key, Text value, Context
context) throws IOException, InterruptedException {

           if (!value.toString().contains("UniqueCarrier")) {

                String row[] = value.toString().split(",");
                if(!row[14].equalsIgnoreCase("NA")){

                      String destAirport = row[17];

                      //String arrDelay = row[14].trim();

                      try {
                          IntWritable busyRating = new
IntWritable(Integer.parseInt(row[14]));
                          context.write(new Text(destAirport),
busyRating);

                      } catch (Exception e) {
                          e.printStackTrace();
                      }
                }

          }
```

```
      }

    }


    public static class Reduce1 extends Reducer<Text, IntWritable,
Text, IntWritable>{

        private IntWritable result = new IntWritable();

        @Override
        protected void reduce(Text key, Iterable<IntWritable> values,
Context context)
                throws IOException, InterruptedException {
            //super.reduce(key, values, context); //To change body of
generated methods, choose Tools | Templates.
            //int count=0;
            int sum=0;
            //double avg=0;

            for(IntWritable val : values){
                sum += val.get();
                //++count;
            }

            //avg=sum/count;
            result.set(sum);
            context.write(key, result);
        }



    }


     public static class Map2 extends Mapper<LongWritable, Text,
IntWritable, Text>{

        //private final static IntWritable sales = new IntWritable();
        //private Text employee = new Text();

        @Override
        protected void map(LongWritable key, Text value, Context
context) throws IOException, InterruptedException {
```

```
            //super.map(key, value, context); //To change body of
generated methods, choose Tools | Templates.

            String row[] = value.toString().split("\\t");
            //IntWritable movieId = new
IntWritable(Integer.parseInt(row[0]));
            //Text airport = new Text(row[0]);

            String rating = row[1].trim();

            try {
                IntWritable ratingg = new
IntWritable(Integer.parseInt(rating));
                context.write(ratingg, new Text(row[0]));

            } catch (Exception e) {
                e.printStackTrace();
            }

        }


    }

    public static class Reduce2 extends Reducer<IntWritable, Text,
Text, IntWritable>{

        private static int count= 25;
        @Override
        protected void reduce(IntWritable key, Iterable<Text> values,
Context context) throws IOException, InterruptedException {

            for(Text val : values){
                if(count>0){
                    context.write(val, key);
                    --count;
                }
                else
                   break;
            }

        }
```

```
    }


    public static class BusyRatingMapper extends Mapper<Object, Text,
Text, Text>{

        private Text outkey= new Text();
        private Text outValue= new Text();

        @Override
         public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            String airportList[] = value.toString().split("\t");
            String airport = airportList[0].replace("\"", "");

            if(airport!=null){
                outkey.set(airport);
                outValue.set("A"+ value.toString().replace("\"", ""));
                context.write(outkey, outValue);
            }
        }
      }

     public static class AiportMapper extends Mapper<Object, Text,
Text, Text>{

        private Text outkey= new Text();
        private Text outValue= new Text();

        @Override
         public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException {

            String aiportList[] = value.toString().split(",");

            if (!value.toString().replace("\"", "").contains("iata"))
{
                 String iata = aiportList[0].replace("\"", "");
                 //System.out.println("Airport nm ->" + iata);
                 if(iata!=null){
                     outkey.set(iata);
                     outValue.set("B"+ value.toString().replace("\"",
```

```
"").replace(",", " "));
                        context.write(outkey, outValue);
                    }
                }
            }
        }


      public static class JoinReducer extends
Reducer<Text,Text,Text,Text> {

            public static final Text EMPTY_TEXT = new Text("");
            private Text tmp = new Text();
            private ArrayList<Text> listA = new ArrayList<Text>();
            private ArrayList<Text> listB= new ArrayList<Text>();
            private String joinType = null;

            @Override
            protected void setup(Context context) throws IOException,
InterruptedException {
                joinType= context.getConfiguration().get("join.type");
            }


            @Override
            public void reduce(Text key, Iterable<Text> values,Context
context)
                    throws IOException, InterruptedException {

                listA.clear();
                listB.clear();

                for(Text val: values){
                    tmp = val;

                    if(tmp.charAt(0) == 'A'){
                        listA.add(new
Text(tmp.toString().substring(1)));
                    }
                    else if(tmp.charAt(0) == 'B'){
                        listB.add(new
Text(tmp.toString().substring(1)));
                    }
                }
```

```
                executeJoin(context);

            }


            private void executeJoin(Context context)
                throws IOException, InterruptedException{
                if(joinType.equalsIgnoreCase("inner")){
                    if(!listA.isEmpty() && !listB.isEmpty()){
                        for(Text A: listA){
                            for(Text B: listB){
                                //System.out.println("ListAB contains
: "+ A + " " + B);
                                context.write(A, B);
                            }
                        }
                    }
                }
            }


     }



     /**
     * @param args the command line arguments
     */
    public static void main(String[] args) throws IOException,
InterruptedException, ClassNotFoundException {

            Configuration conf1 = new Configuration();
            Job job1 = Job.getInstance(conf1, "Chaining");
            job1.setJarByClass(Top10BusiestAirport.class);

            job1.setMapperClass(Map1.class);
            job1.setMapOutputKeyClass(Text.class);
            job1.setMapOutputValueClass(IntWritable.class);


            job1.setReducerClass(Reduce1.class);
            job1.setOutputKeyClass(Text.class);
            job1.setOutputValueClass(IntWritable.class);
```

```
            job1.setCombinerClass(Reduce1.class);

            FileInputFormat.addInputPath(job1, new Path(args[0]));
            FileOutputFormat.setOutputPath(job1, new Path(args[1]));

            boolean firstComplete = job1.waitForCompletion(true);
            boolean secondComplete=false;

            Configuration conf2 = new Configuration();
            Job job2 = Job.getInstance(conf2, "Chaining");



             if(firstComplete) {

                job2.setJarByClass(Top10BusiestAirport.class);
                job2.setMapperClass(Map2.class);
                job2.setMapOutputKeyClass(IntWritable.class);
                job2.setMapOutputValueClass(Text.class);


                job2.setReducerClass(Reduce2.class);
                job2.setOutputKeyClass(Text.class);
                job2.setOutputValueClass(IntWritable.class);

                job2.setSortComparatorClass(SortKeyComparator.class);

                job2.setNumReduceTasks(1);

                FileInputFormat.addInputPath(job2, new Path(args[1]));
                FileOutputFormat.setOutputPath(job2, new
Path(args[2]));

                secondComplete=job2.waitForCompletion(true);

             }

             Configuration conf3 = new Configuration();
             Job job3 = Job.getInstance(conf3, "Join");


             if(secondComplete){

                 job3.setJarByClass(Top10BusiestAirport.class);
```

```
                job3.setMapperClass(BusyRatingMapper.class);
                job3.setMapperClass(AiportMapper.class);
                //job3.setMapOutputKeyClass(Text.class);
                //job3.setMapOutputValueClass(Text.class);
                job3.setReducerClass(JoinReducer.class);

                MultipleInputs.addInputPath(job3, new Path(args[2]),
TextInputFormat.class, BusyRatingMapper.class);
                MultipleInputs.addInputPath(job3, new Path(args[3]),
TextInputFormat.class, AiportMapper.class);
                job3.getConfiguration().set("join.type", "inner");


                job3.setOutputKeyClass(Text.class);
                job3.setOutputValueClass(Text.class);


                FileInputFormat.addInputPath(job3, new
Path(args[2]));
                FileOutputFormat.setOutputPath(job3, new
Path(args[4]));

                 System.exit(job3.waitForCompletion(true)? 0 :1);
            }



    }


}
```

SortKeyComparator.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package finalproject.top10;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.io.WritableComparator;
```

```
/**
 *
 * @author ankitbhayani
 */
public class SortKeyComparator extends WritableComparator{

   protected SortKeyComparator() {
         super(IntWritable.class, true);
   }

   @Override
   public int compare(WritableComparable a, WritableComparable b) {
      IntWritable key1 = (IntWritable) a;
      IntWritable key2 = (IntWritable) b;


         int result = key1.get() < key2.get() ? 1 : key1.get() == key2.get() ? 0 : -1;
      return result;
   }



}
```

```
package finalproject.dailyanalysis;


import java.io.IOException;
//import static java.lang.System.exit;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Partitioner;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author ankitbhayani
```

```
 */
public class DailyTraffic {

  public static class TMapper extends Mapper<Object, Text, Text,
IntWritable>{
    //private final static IntWritable one = new IntWritable(1);
    private Text dayOfTheMonth= new Text();

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        String row[] = value.toString().split(",");
        if (!value.toString().contains("UniqueCarrier")) {
           if(!row[14].equalsIgnoreCase("NA")){
                 dayOfTheMonth.set(row[2]);
                 context.write(dayOfTheMonth, new
IntWritable(Integer.parseInt(row[14])));

             }
        }
    }
  }

  public static class IReducer extends
Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,Context
context)
            throws IOException, InterruptedException {
            int sum=0;

            for (IntWritable val : values){
                sum +=val.get();
            }
            result.set(sum);
            context.write(key, result);
    }
  }

  private static class LPartitioner extends Partitioner<Text,
IntWritable>{

        @Override
```

```
        public int getPartition(Text key, IntWritable value, int
numOfPartitions) {
            //int mon = 0;
            int mon = Integer.parseInt(key.toString());

            /*switch(key.toString()){
                case "Jan" :
                    mon=1;break;
                case "Feb" :
                    mon=2; break;
                case "Mar" :
                    mon=3; break;
                case "Apr" :
                    mon=4; break;
                case "May" :
                    mon=5;break;
                case "Jun" :
                    mon=6; break;
                case "Jul" :
                    mon=7; break;
                case "Aug" :
                    mon=8; break;
                case "Sep" :
                    mon=9; break;
                case "Oct" :
                    mon=10; break;
                case "Nov" :
                    mon=11; break;
                case "Dec" :
                    mon=12; break;

                default :
                    System.out.println("Erroneous Month");
                    exit(0);

            }*/

            return (mon % numOfPartitions);
        }
  }


    public static void main(String[] args) throws IOException,
InterruptedException, ClassNotFoundException {
```

```
            Configuration conf = new Configuration();
            Job job = Job.getInstance(conf, "Daily Partitioning
count");
            job.setJarByClass(DailyTraffic.class);

            job.setMapperClass(TMapper.class);
            job.setMapOutputKeyClass(Text.class);
            job.setMapOutputValueClass(IntWritable.class);

            job.setPartitionerClass(LPartitioner.class);
            job.setNumReduceTasks(31);

            job.setCombinerClass(IReducer.class);
            job.setReducerClass(IReducer.class);
            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);
            FileInputFormat.addInputPath(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));
            System.exit(job.waitForCompletion(true) ? 0 : 1);
    }

}
```

```
package finalproject.partitioning;


import java.io.IOException;
//import static java.lang.System.exit;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Partitioner;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author ankitbhayani
 */
public class MonthlyTraffic {
```

```
  public static class TMapper extends Mapper<Object, Text, Text,
IntWritable>{
    //private final static IntWritable one = new IntWritable(1);
    private Text month= new Text();

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        String row[] = value.toString().split(",");
        if (!value.toString().contains("UniqueCarrier")) {
           if(!row[14].equalsIgnoreCase("NA")){
                 month.set(row[1]);
                 context.write(month, new
IntWritable(Integer.parseInt(row[14])));

             }
        }
    }
  }

  public static class IReducer extends
Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,Context
context)
            throws IOException, InterruptedException {
            int sum=0;

            for (IntWritable val : values){
                sum +=val.get();
            }
            result.set(sum);
            context.write(key, result);
    }
  }

  private static class LPartitioner extends Partitioner<Text,
IntWritable>{

        @Override
        public int getPartition(Text key, IntWritable value, int
numOfPartitions) {
```

```
            //int mon = 0;
            int mon = Integer.parseInt(key.toString());

            /*switch(key.toString()){
                case "Jan" :
                    mon=1;break;
                case "Feb" :
                    mon=2; break;
                case "Mar" :
                    mon=3; break;
                case "Apr" :
                    mon=4; break;
                case "May" :
                    mon=5;break;
                case "Jun" :
                    mon=6; break;
                case "Jul" :
                    mon=7; break;
                case "Aug" :
                    mon=8; break;
                case "Sep" :
                    mon=9; break;
                case "Oct" :
                    mon=10; break;
                case "Nov" :
                    mon=11; break;
                case "Dec" :
                    mon=12; break;

                default :
                    System.out.println("Erroneous Month");
                    exit(0);

            }*/

            return (mon % numOfPartitions);
        }
    }


    public static void main(String[] args) throws IOException,
InterruptedException, ClassNotFoundException {
            Configuration conf = new Configuration();
            Job job = Job.getInstance(conf, "Monthly Partitioning
```

```
count");
            job.setJarByClass(MonthlyTraffic.class);

            job.setMapperClass(TMapper.class);
            job.setMapOutputKeyClass(Text.class);
            job.setMapOutputValueClass(IntWritable.class);

            job.setPartitionerClass(LPartitioner.class);
            job.setNumReduceTasks(12);

            job.setCombinerClass(IReducer.class);
            job.setReducerClass(IReducer.class);
            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);
            FileInputFormat.addInputPath(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));
            System.exit(job.waitForCompletion(true) ? 0 : 1);
    }

}

package finalproject.weeklyanalysis;


import java.io.IOException;
//import static java.lang.System.exit;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Partitioner;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author ankitbhayani
 */
public class WeeklyTraffic {

  public static class TMapper extends Mapper<Object, Text, Text,
```

```
IntWritable>{
    //private final static IntWritable one = new IntWritable(1);
    private Text dayOfTheWeek= new Text();

    public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException {

        String row[] = value.toString().split(",");
        if (!value.toString().contains("UniqueCarrier")) {
           if(!row[14].equalsIgnoreCase("NA")){
                 dayOfTheWeek.set(row[3]);
                 context.write(dayOfTheWeek, new
IntWritable(Integer.parseInt(row[14])));

              }
        }
    }
  }

  public static class IReducer extends
Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,Context
context)
            throws IOException, InterruptedException {
            int sum=0;

            for (IntWritable val : values){
                sum +=val.get();
            }
            result.set(sum);
            context.write(key, result);
    }
  }

  private static class LPartitioner extends Partitioner<Text,
IntWritable>{

        @Override
        public int getPartition(Text key, IntWritable value, int
numOfPartitions) {
            //int mon = 0;
            int mon = Integer.parseInt(key.toString());
```

```
            /*switch(key.toString()){
                case "Jan" :
                    mon=1;break;
                case "Feb" :
                    mon=2; break;
                case "Mar" :
                    mon=3; break;
                case "Apr" :
                    mon=4; break;
                case "May" :
                    mon=5;break;
                case "Jun" :
                    mon=6; break;
                case "Jul" :
                    mon=7; break;
                case "Aug" :
                    mon=8; break;
                case "Sep" :
                    mon=9; break;
                case "Oct" :
                    mon=10; break;
                case "Nov" :
                    mon=11; break;
                case "Dec" :
                    mon=12; break;

                default :
                    System.out.println("Erroneous Month");
                    exit(0);

            }*/

            return (mon % numOfPartitions);
        }
    }


    public static void main(String[] args) throws IOException,
InterruptedException, ClassNotFoundException {
            Configuration conf = new Configuration();
            Job job = Job.getInstance(conf, "Daily Partitioning
count");
            job.setJarByClass(WeeklyTraffic.class);
```

```
            job.setMapperClass(TMapper.class);
            job.setMapOutputKeyClass(Text.class);
            job.setMapOutputValueClass(IntWritable.class);

            job.setPartitionerClass(LPartitioner.class);
            job.setNumReduceTasks(7);

            job.setCombinerClass(IReducer.class);
            job.setReducerClass(IReducer.class);
            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);
            FileInputFormat.addInputPath(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));
            System.exit(job.waitForCompletion(true) ? 0 : 1);
    }

}
```

```
package finalproject.binning;

/*
* To change this license header, choose License Headers in Project
Properties.
* To change this template file, choose Tools | Templates
* and open the template in the editor.
*/

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.MultipleOutputs;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

/**
*
* @author ankitbhayani
```

```
*/
public class FlightCancellationReasonBinning {

public static class TMapper extends Mapper<Object, Text, Text,
NullWritable>{

private MultipleOutputs<Text, NullWritable> mos=null;
//private final static IntWritable one = new IntWritable(1);
//private Text hour= new Text();

@Override
protected void setup(Context context) throws IOException,
InterruptedException {
        mos = new MultipleOutputs(context);
 }

 @Override
 public void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

    String row[] = value.toString().split(",");
    if (!value.toString().contains("UniqueCarrier")) {
        if(!row[21].equalsIgnoreCase("0")){
              String cancellationCode = row[22];

              if(cancellationCode.equalsIgnoreCase("A"))
                  mos.write("bins", value, NullWritable.get(),"Carrier-
cancellation");
              if(cancellationCode.equalsIgnoreCase("B"))
                  mos.write("bins", value, NullWritable.get(),"Weather-
cancellation");
              if(cancellationCode.equalsIgnoreCase("C"))
                  mos.write("bins", value, NullWritable.get(),"NAS-
cancellation");
              if(cancellationCode.equalsIgnoreCase("D"))
                  mos.write("bins", value,
NullWritable.get(),"Security-cancellation");
      }
    }
}

    @Override
    protected void cleanup(Context context) throws IOException,
InterruptedException {
```

```
        mos.close();
    }

}
```

```
public static void main(String[] args) throws IOException,
InterruptedException, ClassNotFoundException {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Cancellation Binning ");
        job.setJarByClass(FlightCancellationReasonBinning.class);

        job.setMapperClass(TMapper.class);


        MultipleOutputs.addNamedOutput(job, "bins",
TextOutputFormat.class, Text.class, IntWritable.class);
        MultipleOutputs.setCountersEnabled(job, true);
        job.setNumReduceTasks(0);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(NullWritable.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);
}

}
```