jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

Out[28]: 'List of airlines of the United States - Wikipedia'

In [29]:
```python
soup.a
```

Out[29]:
```
<a class="mw-jump-link" href="#bodyContent">Jump to content</a>
```

In [43]:
```python
links = soup.find_all('a')
```

In [38]:
```python
tables = soup.find_all('table')
```

In [44]:
```python
for link in links:
    print(link['href'])
```

```
#bodyContent
/wiki/Main_Page
/wiki/Wikipedia:Contents
/wiki/Portal:Current_events
/wiki/Special:Random
/wiki/Wikipedia:About
//en.wikipedia.org/wiki/Wikipedia:Contact_us
https://donate.wikimedia.org/wiki/Special:FundraiserRedirector?utm_source=donate&utm_medium=sidebar&utm_campaign=C13_en.wikip
edia.org&uselang=en
/wiki/Help:Contents
/wiki/Help:Introduction
/wiki/Wikipedia:Community_portal
/wiki/Special:RecentChanges
/wiki/Wikipedia:File_upload_wizard
/wiki/Main_Page
/wiki/Special:Search
/w/index.php?title=Special:CreateAccount&returnto=List+of+airlines+of+the+United+States
/w/index.php?title=Special:UserLogin&returnto=List+of+airlines+of+the+United+States
/w/index.php?title=Special:CreateAccount&returnto=List+of+airlines+of+the+United+States
```

jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted

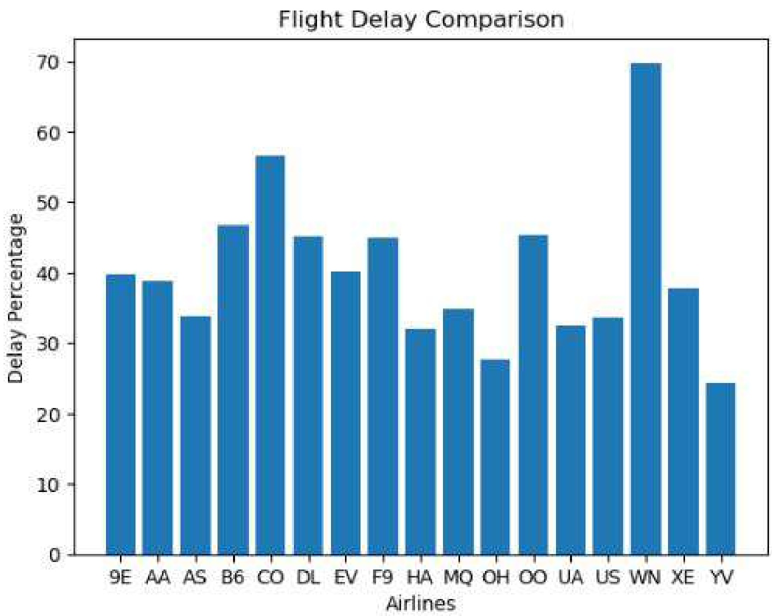
Python 3 (ipykernel)

Run Code

In [42]:

```
# Bar Plot
plt.bar(delay_percentages.index, delay_percentages.values)
plt.xlabel('Airlines')
plt.ylabel('Delay Percentage')
plt.title('Flight Delay Comparison')


plt.show()
```



'WN' is the symbol of SouthWest Airlines and from the above graph we can see that

SouthWest airlines have high percentage of Delay time than other airlines

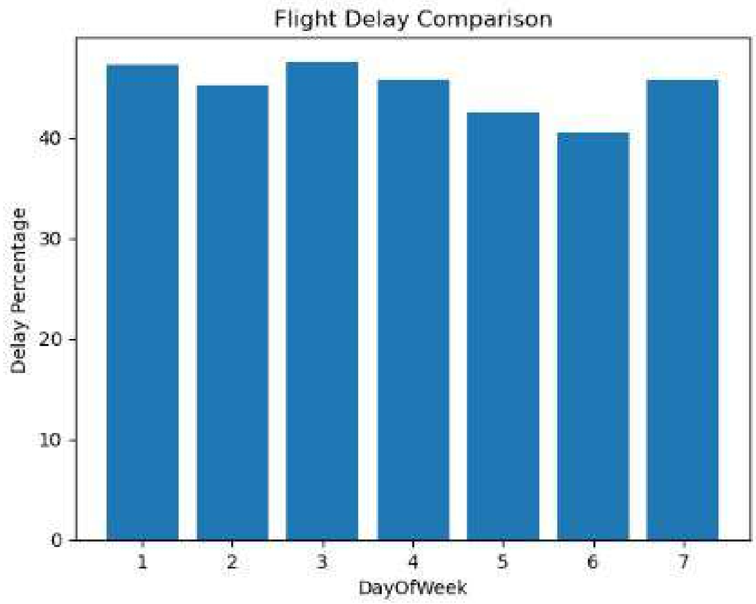
In [43]:

```
delay_week_percentages = airlines.groupby('DayOfWeek')['Delay'].mean() * 100
```

In [44]:

```
# Bar Plot
plt.bar(delay_week_percentages.index, delay_week_percentages.values)
plt.xlabel('DayOfWeek')
plt.ylabel('Delay Percentage')
plt.title('Flight Delay Comparison')

plt.show()
```



20States%20Airlines%20Analysis.ipynb#

jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

```
Long-Distance          559
Name: Travel_Category, dtype: int64
```

In [52]:

```
pd.DataFrame(airlines.groupby(['Airline', 'Travel_Category']).size())
```

Out[52]:

Airline	Travel_Category	0
9E	Short-Distance	20686
	Medium-Distance	0
	Long-Distance	0
AA	Short-Distance	36171
	Medium-Distance	9038
	Long-Distance	149
AS	Short-Distance	8839
	Medium-Distance	2772
	Long-Distance	0
B6	Short-Distance	14720
	Medium-Distance	3391
	Long-Distance	0
CO	Short-Distance	14800
	Medium-Distance	6349
	Long-Distance	166
DL	Short-Distance	40852
	Medium-Distance	11149
	Long-Distance	109
EV	Short-Distance	27973
	Medium-Distance	10
	Long-Distance	0
F9	Short-Distance	5902
	Medium-Distance	551
	Long-Distance	0
HA	Short-Distance	4575
	Medium-Distance	1005
	Long-Distance	0
MQ	Short-Distance	36543
	Medium-Distance	62
	Long-Distance	0
OH	Short-Distance	12505
	Medium-Distance	125
	Long-Distance	0
OO	Short-Distance	40690
	Medium-Distance	284
	Long-Distance	0
UA	Short-Distance	18157
	Medium-Distance	9327
	Long-Distance	135
US	Short-Distance	28063
	Medium-Distance	6437
	Long-Distance	0
WN	Short-Distance	56786

jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

	Long-Distance	0
WN	Short-Distance	88786
	Medium-Distance	7311
	Long-Distance	0
XE	Short-Distance	31072
	Medium-Distance	54
	Long-Distance	0
YV	Short-Distance	13887
	Medium-Distance	58
	Long-Distance	0

In [53]:

```
Long_Duration_Flights = airlines[airlines['Travel_Category'] == 'Long-Distance']
Long_Duration_Flights.shape
```

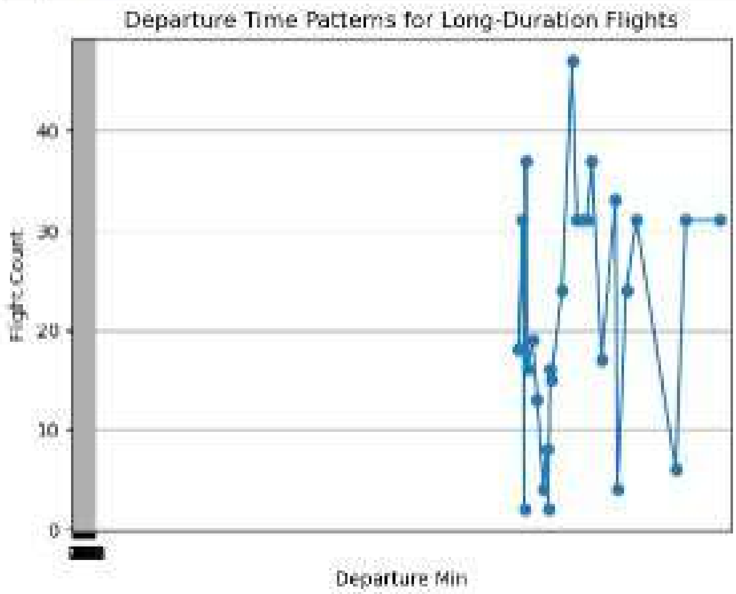
Out[53]: (559, 10)

In [54]:

```
departure_min_counts = Long_Duration_Flights['Length'].value_counts().sort_index()
```

In [55]:

```
plt.plot(departure_min_counts.index, departure_min_counts.values, marker='o')
plt.xlabel('Departure Min')
plt.ylabel('Flight Count')
plt.title('Departure Time Patterns for Long-Duration Flights')
plt.xticks(range(24))  # Set x-axis ticks for each hour
plt.grid(True)
plt.show()
```



Null hypothesis (H0): The duration of a flight does not affect flight delays.

Alternative hypothesis (H1): The duration of a flight affects flight delays.

In [56]:

```
short_delay = airlines[airlines['Travel_Category'] == 'Short-Distance']['Delay']
medium_delay =airlines[airlines['Travel_Category'] == 'Medium-Distance']['Delay']
long_delay =airlines[airlines['Travel_Category'] == 'Long-Distance']['Delay']
```

jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted

Python 3 (ipykernel)

Run Code

Null hypothesis (H0): The duration of a flight does not affect flight delays.

Alternative hypothesis (H1): The duration of a flight affects flight delays.

In [56]:
```
short_delay = airlines[airlines['Travel_Category'] == 'Short-Distance']['Delay']
medium_delay =airlines[airlines['Travel_Category'] == 'Medium-Distance']['Delay']
long_delay =airlines[airlines['Travel_Category'] == 'Long-Distance']['Delay']
```

In [57]:
```
# Perform one-way ANOVA
f_stat, p_value = f_oneway(short_delay, medium_delay, long_delay)
```

In [58]:
```
print("F-Statistic: ", f_stat)
print("P-Value: ", p_value)
```

```
F-Statistic:  352.11904019127104
P-Value:  1.5149282534228687e-153
```

In [59]:
```
# Interpret the results
alpha = 0.05

if p_value < alpha:
    print("Reject the null hypothesis.")
    print("There is a significant difference among the delay groups.")
else:
    print("Fail to reject the null hypothesis.")
    print("There is not enough evidence to conclude a significant difference among the delay groups.")
```

```
Reject the null hypothesis.
There is a significant difference among the delay groups.
```

Null hypothesis (H0): The altitude of the airport does not affect flight delays

There is a significant difference among the delay groups.

Null hypothesis (H0): The altitude of the airport does not affect flight delays.

Alternative hypothesis (H1): The altitude of the airport affects flight delays.

In [60]:
```
t_stat, p_value = stats.ttest_ind((df['Delay']), (df['elevation_ft'].dropna()), equal_var=False)
```

In [61]:
```
print("F-Statistic: ", f_stat)
print("P-Value: ", p_value)
```

```
F-Statistic:  352.11904019127104
P-Value:  0.0
```

In [62]:
```
# Interpret the results
alpha = 0.05

if p_value < alpha:
    print("Reject the null hypothesis.")
    print("There is a significant difference among the delay groups.")
else:
    print("Fail to reject the null hypothesis.")
    print("There is not enough evidence to conclude a significant difference among the delay groups.")
```

```
Reject the null hypothesis.
There is a significant difference among the delay groups.
```

Null hypothesis (H0): The number of runways does not affect flight delays.

Alternative hypothesis (H1): The number of runways affects flight delays.

In [63]:

```
t_stat, p_value = stats.ttest_ind((df['Delay']), (df['closed']), equal_var=False)
```

In [64]:

```
print("F-Statistic: ", f_stat)
print("P-Value: ", p_value)
```

```
F-Statistic:  352.11904019127104
P-Value:  0.0
```

In [65]:

```
# Interpret the results
alpha = 0.05

if p_value < alpha:
    print("Reject the null hypothesis.")
    print("There is a significant difference among the delay groups.")
else:
    print("Fail to reject the null hypothesis.")
    print("There is not enough evidence to conclude a significant difference among the delay groups.")
```

```
Reject the null hypothesis.
There is a significant difference among the delay groups.
```

jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted

Python 3 (ipykernel)

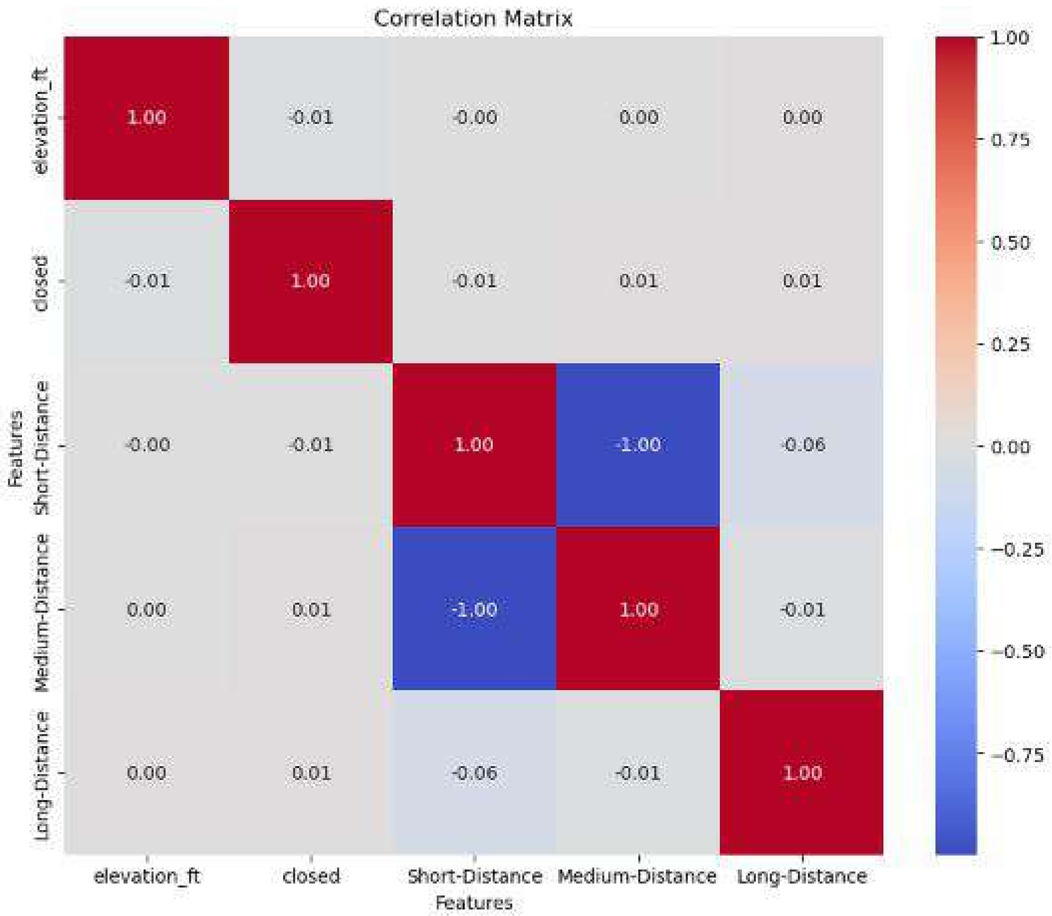
Run Code

In [77]:

```python
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", square=True)

plt.title('Correlation Matrix')
plt.xlabel('Features')
plt.ylabel('Features')

plt.show()
```



jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [119]:
```
X = new_data.drop('Delay', axis=1)
y = new_data['Delay']
```

In [137]:
```
logistic_model = SGDClassifier(loss='log_loss', random_state=42)
logistic_model.fit(X, y)
```

Out[137]:
```
SGDClassifier
SGDClassifier(loss='log_loss', random_state=42)
```

In [122]:
```
decision_tree_model = DecisionTreeClassifier(random_state=42)
decision_tree_model.fit(X, y)
```

Out[122]:
```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

In [123]:
```
n_folds = 5
stratified_kfold = StratifiedKFold(n_splits=n_folds, random_state=42, shuffle=True)
logistic_scores = []
decision_tree_scores = []
```

In [128]:
```
# Perform cross-validation
for train_index, val_index in stratified_kfold.split(X, y):
    # Split the data into training and validation sets
    X_train, X_val = X.iloc[train_index], X.iloc[val_index]
    y_train, y_val = y.iloc[train_index], y.iloc[val_index]
```

In [129]:
```
# Fit the logistic regression model
logistic_model.fit(X_train, y_train)

# Predict and calculate accuracy for logistic regression model
logistic_pred = logistic_model.predict(X_val)
logistic_accuracy = accuracy_score(y_val, logistic_pred)
logistic_scores.append(logistic_accuracy)

# Fit the decision tree model
decision_tree_model.fit(X_train, y_train)

# Predict and calculate accuracy for decision tree model
decision_tree_pred = decision_tree_model.predict(X_val)
decision_tree_accuracy = accuracy_score(y_val, decision_tree_pred)
decision_tree_scores.append(decision_tree_accuracy)
```

jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

```
decision_tree_scores.append(decision_tree_accuracy)
```

In [140]:

```
# Define the hyperparameter grid for logistic regression
logistic_param_grid = {
    'alpha': [0.001, 0.01, 0.1],
    'max_iter': [100, 200, 300]
}

# Define the hyperparameter grid for decision tree
decision_tree_param_grid = {
    'max_depth': [None, 5, 10],
    'min_samples_split': [2, 5, 10]
}

# Create RandomizedSearchCV objects
logistic_search = RandomizedSearchCV(logistic_model, param_distributions=logistic_param_grid, cv=KFold(n_splits=n_folds, shu
decision_tree_search = RandomizedSearchCV(decision_tree_model, param_distributions=decision_tree_param_grid, cv=KFold(n_spli

# Fit the models with hyperparameter tuning
logistic_search.fit(X, y)
decision_tree_search.fit(X, y)

# Get the best hyperparameters for each model
best_logistic_params = logistic_search.best_params_
best_decision_tree_params = decision_tree_search.best_params_
```

```
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:705: ConvergenceWarning: Maximum num
ber of iteration reached before convergence. Consider increasing max_iter to improve the fit.
  warnings.warn(

C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:705: ConvergenceWarning: Maximum num
ber of iteration reached before convergence. Consider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:705: ConvergenceWarning: Maximum num
ber of iteration reached before convergence. Consider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:705: ConvergenceWarning: Maximum num
ber of iteration reached before convergence. Consider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:705: ConvergenceWarning: Maximum num
ber of iteration reached before convergence. Consider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\model_selection\_search.py:306: UserWarning: The total space of parameter
s 9 is smaller than n_iter=10. Running 9 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
```

jupyter United States Airlines Analysis Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
warnings.warn(
```

In [139]:

```python
# Create the SGDClassifier model
sgd_model = SGDClassifier(loss='log', random_state=42)

# Define the hyperparameter grid for SGDClassifier
sgd_param_grid = {
    'alpha': [0.0001, 0.001, 0.01],
    'penalty': ['l1', 'l2'],
    'max_iter': [100, 200, 300]
}

# Create the RandomizedSearchCV object for SGDClassifier
sgd_search = RandomizedSearchCV(sgd_model, param_distributions=sgd_param_grid, cv=KFold(n_splits=n_folds, shuffle=True), ran

# Fit the model with hyperparameter tuning
sgd_search.fit(X, y)

```

```
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:173: FutureWarning: The loss 'log' w
as deprecated in v1.1 and will be removed in version 1.3. Use `loss='log_loss'` which is equivalent.
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:705: ConvergenceWarning: Maximum num
ber of iteration reached before convergence. Consider increasing max_iter to improve the fit.
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:173: FutureWarning: The loss 'log' w
as deprecated in v1.1 and will be removed in version 1.3. Use `loss='log_loss'` which is equivalent.
  warnings.warn(
C:\Users\naray\anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py:705: ConvergenceWarning: Maximum num
ber of iteration reached before convergence. Consider increasing max_iter to improve the fit.
  warnings.warn(
```

Out[139]:

```
RandomizedSearchCV
estimator: SGDClassifier
    SGDClassifier
```

In [152]:

```python
xgb_classifier = XGBClassifier()
```

In [155]:

```python
n_folds = 5
stratified_kfold = StratifiedKFold(n_splits=n_folds, shuffle=True, random_state=42)

# Perform cross-validation using the XGBoost classifier
xgb_scores = cross_val_score(xgb_classifier, X, y, cv=stratified_kfold)
```

File Edit View Insert Cell Kernel Widgets Help

Not Trusted

Python 3 (ipykernel)

Run

Code

▸ SGDClassifier

In [152]:

```
xgb_classifier = XGBClassifier()
```

In [155]:

```
n_folds = 5
stratified_kfold = StratifiedKFold(n_splits=n_folds, shuffle=True, random_state=42)

# Perform cross-validation using the XGBoost classifier
xgb_scores = cross_val_score(xgb_classifier, X, y, cv=stratified_kfold)
```

In [147]:

```
best_logistic_params
```

Out[147]:

```
{'max_iter': 200, 'alpha': 0.01}
```

In [148]:

```
best_decision_tree_params
```

Out[148]:

```
{'min_samples_split': 2, 'max_depth': None}
```

In [144]:

```
best_sgd_params = sgd_search.best_params_
best_sgd_params
```

Out[144]:

```
{'penalty': 'l2', 'max_iter': 100, 'alpha': 0.01}
```

In [150]:

```
logistic_mean_accuracy = sum(logistic_scores) / len(logistic_scores)
decision_tree_mean_accuracy = sum(decision_tree_scores) / len(decision_tree_scores)

print("Logistic Regression Mean Accuracy:", logistic_mean_accuracy)
print("Decision Tree Mean Accuracy:", decision_tree_mean_accuracy)
```

```
Logistic Regression Mean Accuracy: 0.5807574327512978
Decision Tree Mean Accuracy: 0.6493629070316187
```

In [156]:

```
print("XGBoost Classifier Mean Accuracy:", xgb_scores.mean())
```

```
XGBoost Classifier Mean Accuracy: 0.6683181451539499
```

In [ ]:

US Airlines Analysis

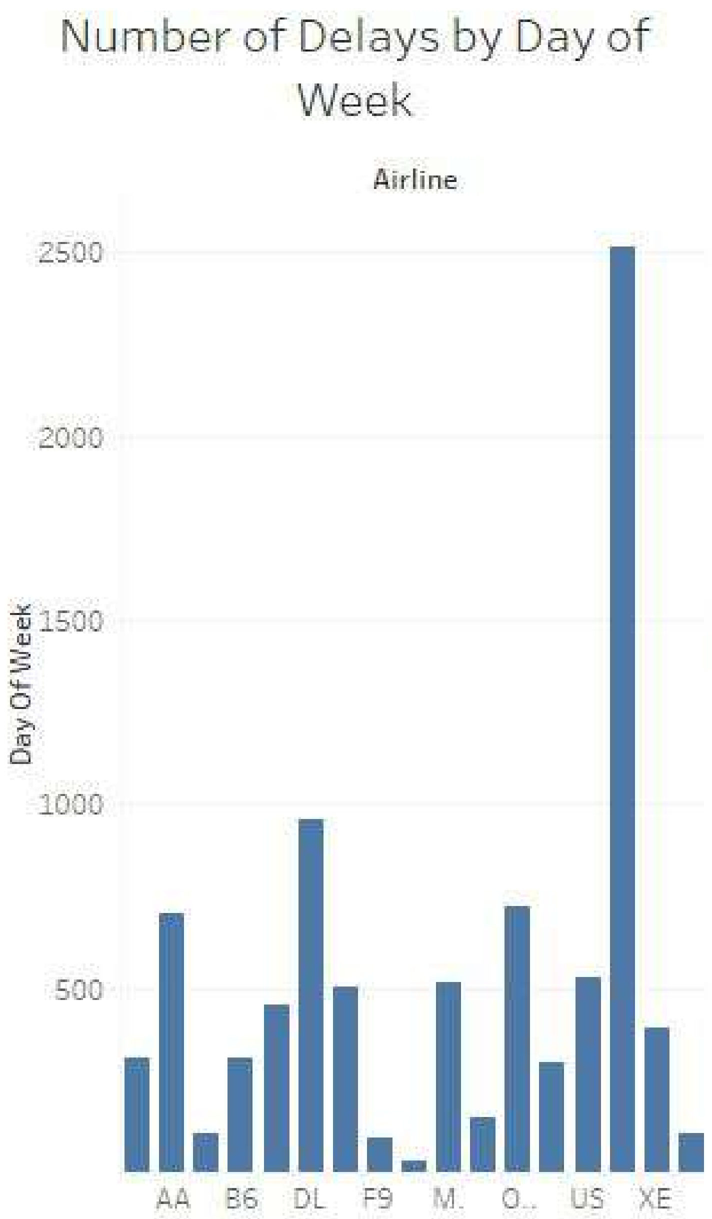
Total Count by Category

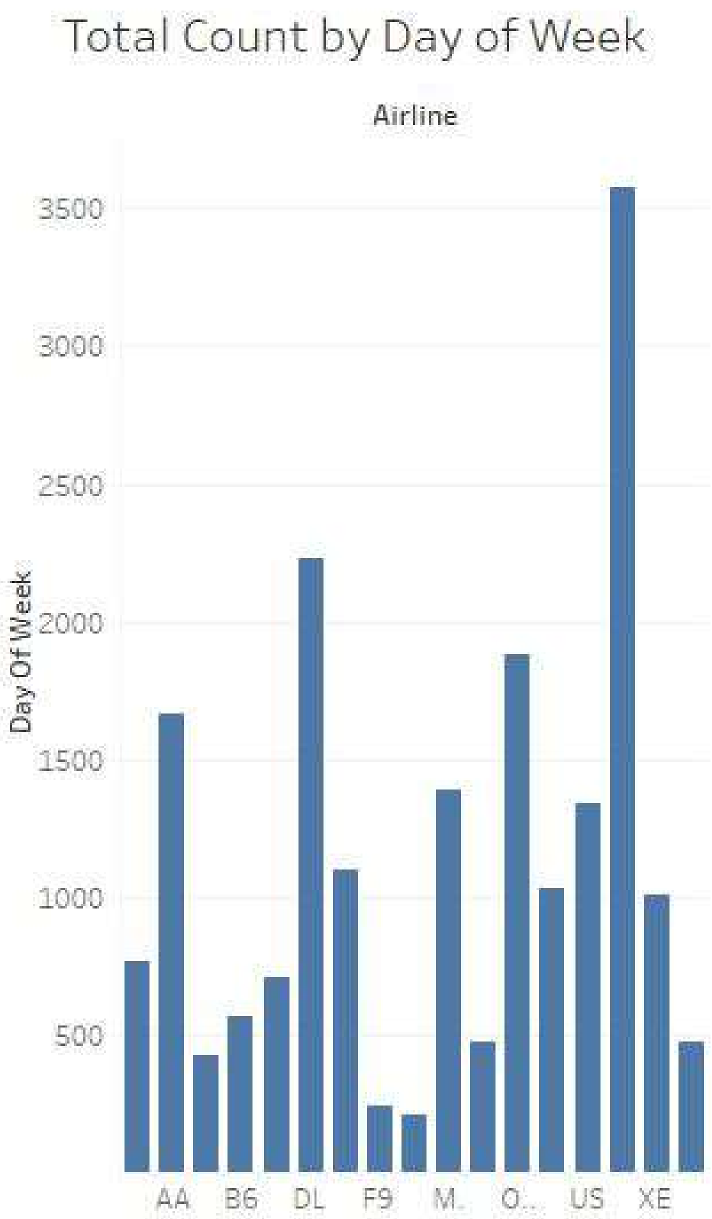
	Travel Category		
Airline	Long-Di..	Mediu..	Short-D..
9E			1,684
AA	8	731	2,935
AS		246	710
B6		242	1,098
CO	6	483	1,131
DL	3	903	4,099
EV		1	2,317
F9		41	445
HA		81	363
MQ		6	3,013
OH		4	1,053
OO		19	4,145
UA	4	815	1,544
US		512	2,455
WN		605	7,213
XE		2	2,451
YV		5	1,011

Number of Delays by Category

	Travel Category		
Airline	Long-Di..	Mediu..	Short-D..
9E			595
AA	4	306	904
AS		84	171
B6		110	494
CO	4	312	521
DL	2	386	1,409
EV		1	815
F9		15	156
HA		29	54
MQ		2	908
OH		1	250
OO		4	1,542
UA	0	219	294
US		197	740
WN		450	4,250
XE		2	689
YV		1	165

US Airlines Analysis





US Airlines Analysis

Types of airports with respect to Airlines

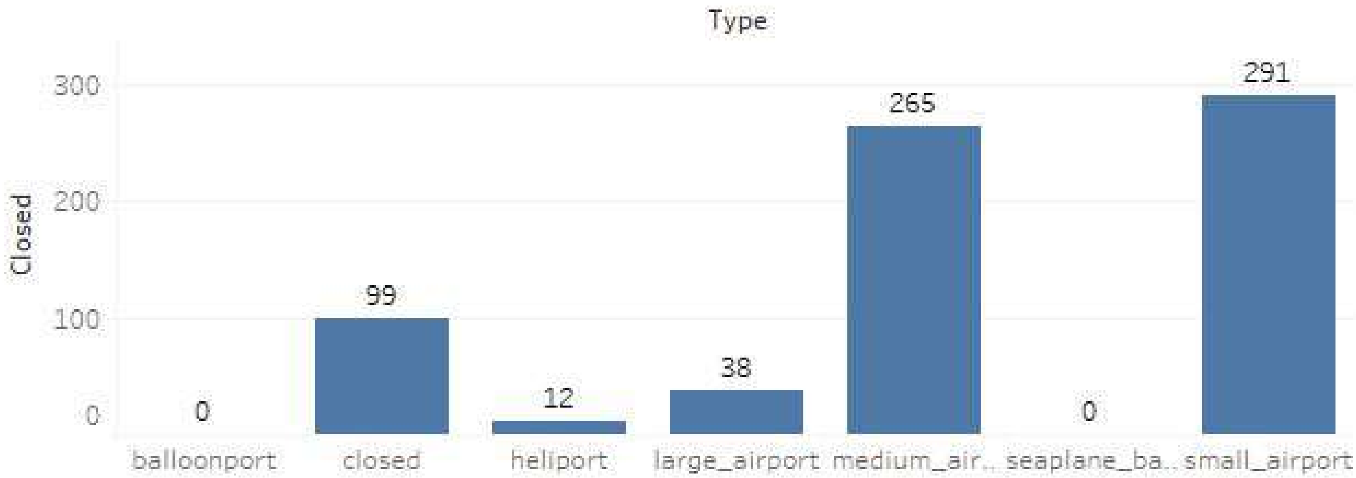
	Airline								
Type	9E	AA	AS	B6	CO	DL	EV	F9	HA
balloonport	2	1		1	1	2			1
closed	101	241	71	78	109	315	136	32	48
heliport	274	634	151	211	276	760	405	71	61
large_airpo..	40	65	17	27	34	83	60	14	16
medium_ai..	258	555	149	181	221	682	342	57	63
seaplane_b..	30	46	8	23	22	112	26	8	13
small_airpo..	979	2,132	560	819	957	3,051	1,349	304	242

Types of airports with respect to Airlines Delay count

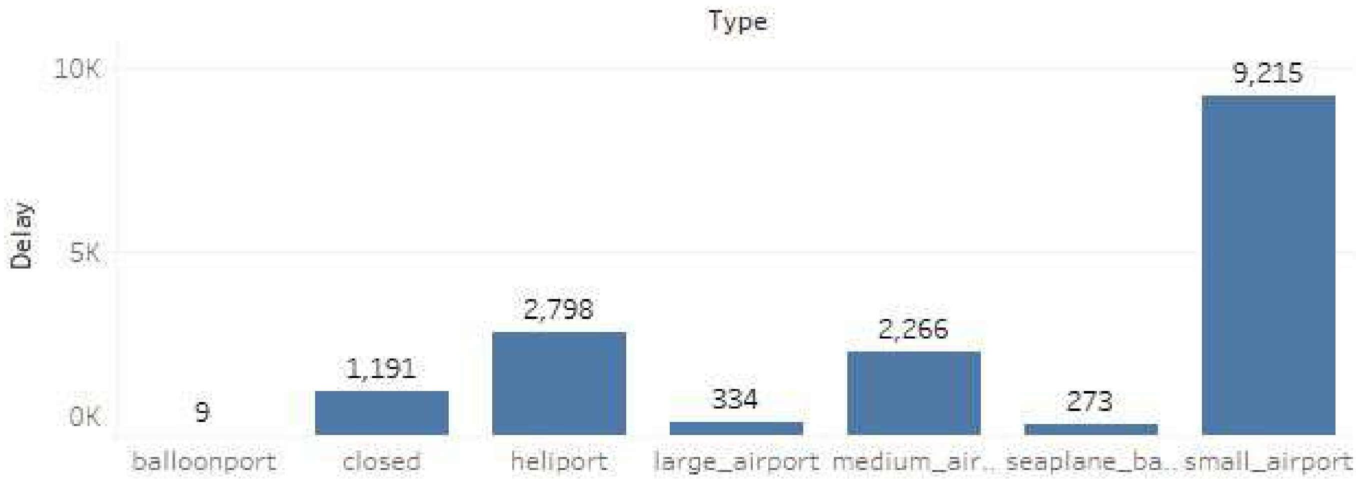
	Airline								
Type	9E	AA	AS	B6	CO	DL	EV	F9	HA
balloonport	0	0		1	1	2			
closed	34	103	16	43	57	137	48	7	
heliport	97	247	35	90	151	280	151	22	1
large_airpo..	14	20	10	8	22	35	10	4	
medium_ai..	85	171	50	80	131	240	149	30	1
seaplane_b..	18	17	1	14	15	44	5	3	
small_airpo..	347	656	143	368	460	1,059	453	105	4

US Airlines Analysis

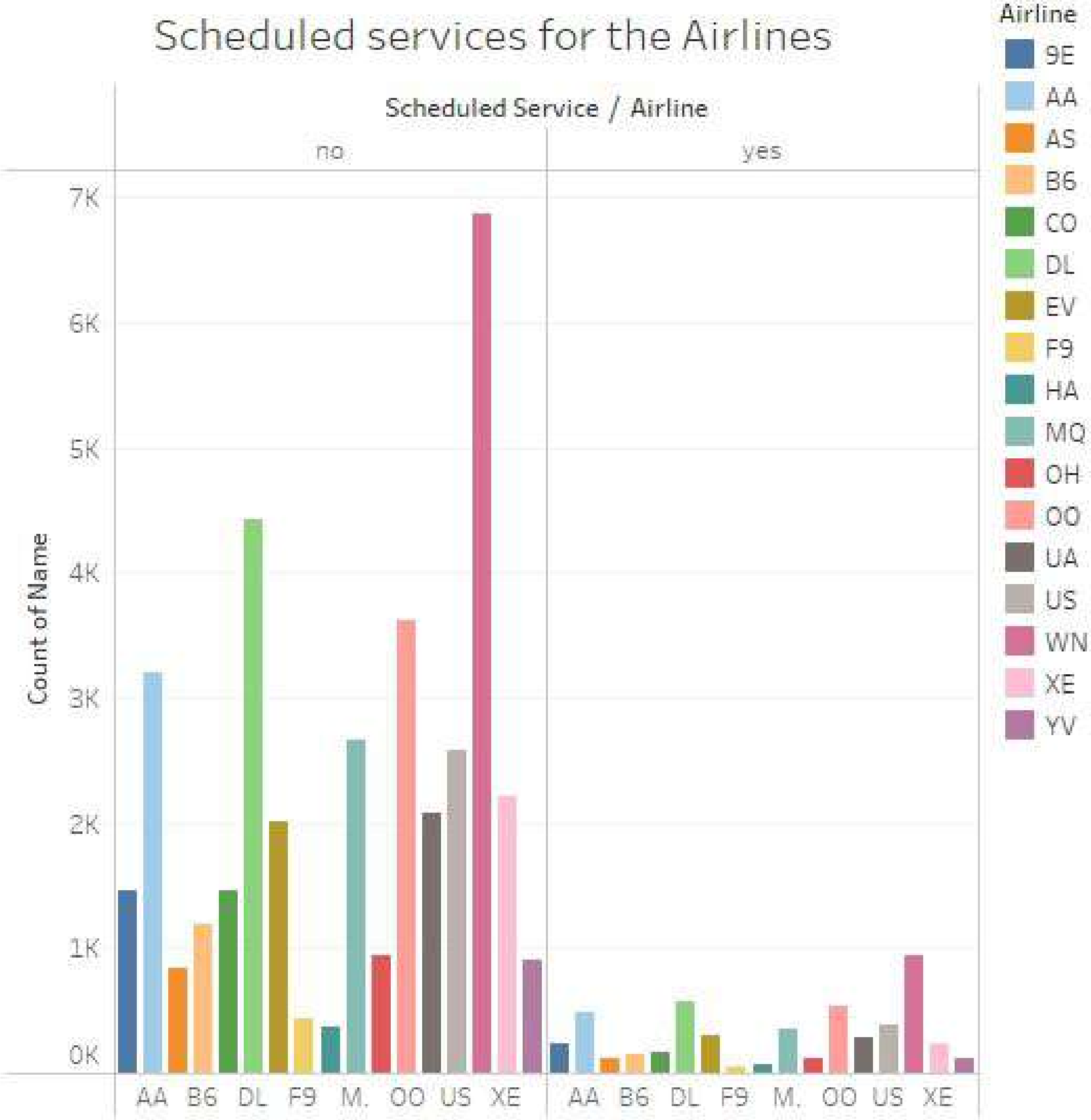
Number of Runways closed with respect to types of Airports



Flight delays with respect to types of Airports



US Airlines Analysis

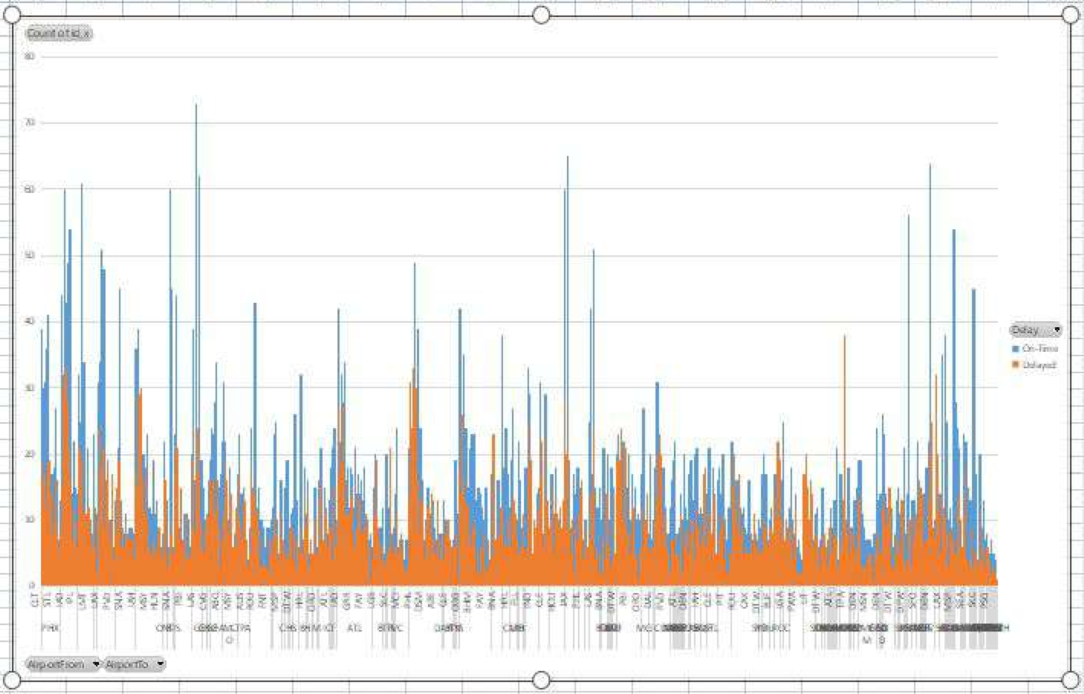


F	G	H	I
	On-time Performance		
	Airline	**Sum of Length**	
	9E	2.91%	
	AA	11.28%	
	AS	3.01%	
	B6	4.06%	
	CO	5.52%	
	DL	13.96%	
	EV	3.81%	
	F9	1.32%	
	HA	0.76%	
	MQ	5.63%	
	OH	2.06%	
	OO	7.05%	
	UA	8.07%	
	US	7.63%	
	WN	16.49%	
	XE	4.79%	
	YV	1.64%	
	Grand Total	**100.00%**	

Percentage of Delayed flights with respective to week

Count of id_x	Delay		
DayOfWeek	On-Time	Delayed	Grand Total
Sunday	1.34%	1.42%	2.77%
Monday	0.97%	0.91%	1.89%
Tuesday	24.50%	20.43%	44.93%
Wednesday	23.68%	9.95%	33.62%
Thursday	9.31%	2.88%	12.19%
Friday	0.74%	0.86%	1.60%
Saturday	1.51%	1.50%	3.00%
Grand Total	**62.05%**	**37.95%**	**100.00%**

Count of Id_x		Delay		
AirportFrom	AirportTo	On-Time	Delayed	Grand Total
PHX		754	410	1164
	CLT	13	9	22
	DFW	39	5	44
	SEA	17	19	36
	MSP	20	11	31
	DTW	14	9	23
	ORD	19	9	28
	ATL	18	3	21
	PDX	14	13	27
	JFK	8	5	13
	IAH	17	5	22
	SLC	30	16	46
	HNL	6	1	7
	MCO	6	2	8
	OGG	3		3
	LAX	31	17	48
	KOA		1	1
	SFO	19	12	31
	MIA	2	1	3
	IAD	2		2
	SMF	12	8	20
	PHL	14	6	20
	LIH		1	1
	DEN	36	12	48
	MEM	7	1	8
	CVG	1	4	5
	YUM	10		10
	MKE	3	1	4
	LAS	41	19	60
	ANC	1		1
	BOS	3	11	14
	LGB	12	3	15
	FLL	3	2	5
	EWR	9	8	17
	DCA	5	1	6
	RDU	3	2	5
	MCI	6	7	13
	SAN	17	19	36
	ONT	14	13	27
	OAK	15	6	21



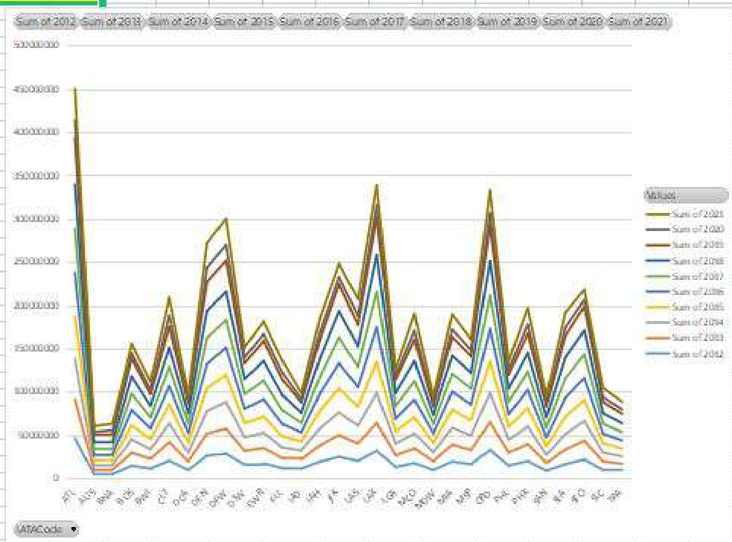
Sheet1 df

I1

Large Hub

Large Hub

	Sum of 2012	Sum of 2013	Sum of 2014	Sum of 2015	Sum of 2016	Sum of 2017	Sum of 2018	Sum of 2019	Sum of 2020	Sum of 2021
4	45798928	45308407	46604273	49340732	50501858	50251964	51865797	53505795	20559866	36676010
5	4606252	4900959	5219982	5797547	6095345	6973115	7921797	8683711	3141505	6666215
6	4797102	5050989	5396958	5715205	6338517	6902771	8017347	8935654	4013995	7594049
7	14293695	14810153	15507561	16290362	17739044	18759742	20006521	20699377	6035452	10909817
8	11186444	11132731	11022200	11738845	12340972	12976554	13371816	13284687	5451355	9253561
9	20033816	21346601	21537725	21913166	21511880	22011251	22281949	24199688	12952869	20900875
10	9462231	9838034	10057794	11242375	11470854	11506310	11367176	11595454	3573489	6731737
11	25799841	25496885	26000591	26280043	28267394	29809097	31362941	33592945	16243216	28645527
12	28022904	29038128	30804567	31589839	31283579	31816933	32821799	35778573	18593421	30005266
13	15599879	15683523	15773941	16255520	16847135	17036092	17436837	18143040	6822324	11517696
14	17055993	17546506	17773405	18684818	19923009	21571198	22797602	23160763	7985474	14514049
15	11445103	11538140	12031860	13061632	14263270	15817043	17612331	17950989	8015744	13598994
16	10816216	10570993	10415948	10363974	10596942	11024306	11621623	11884117	3862658	7227875
17	19039000	18952840	19772087	20595881	20062072	19603731	21157398	21905309	8682558	16242821
18	24520981	25036358	26244928	27782369	29239151	29533154	30620769	31036655	8269819	15273342
19	19959651	19946179	20620248	21857693	22833267	23364393	23795012	24728361	10584059	19160342
20	31326268	32425892	34314197	36351272	39636042	41232432	42624050	42939104	14055777	23663410
21	12818717	13372269	13535372	14319924	14762593	14614802	15058501	15393601	4147116	7827307
22	17159427	16884524	17278608	18759938	20283541	21565448	23202480	24562271	10467728	19618838
23	9436387	9915646	10311996	10830850	11044387	10912074	10678018	10081781	4236603	7680617
24	18987488	19420089	19471466	20986349	20875813	20709225	21021640	21421031	8786007	17500096
25	15943878	16280835	16972678	17634273	18123844	18409704	18361942	19192917	7069720	12211409
26	32171795	32317835	33843426	36305668	37589899	38593028	39873927	40871223	14606034	26350976
27	14589337	14727945	14792339	15101349	14564419	14271243	15292670	16006389	5753239	9820222
28	19560870	19525109	20344867	21351504	20896265	21185458	21622580	22433552	10531436	18940287
29	8686621	8878772	9333152	9985763	10340164	11139933	12174224	12648692	4637856	7836360
30	16121123	16690295	17888080	20148980	21887110	22639124	24024908	25001762	9462411	17430195
31	21284236	21704626	22770783	24190560	25707101	26900048	27790717	27779230	7745057	11725347
32	9579840	9668048	10139065	10634538	11143738	11615954	12226730	12840841	5753239	10795906
33	8218487	8267752	8531561	9150458	9194994	9548580	10368314	10978756	4966775	8847197
34	**518322510**	**526277063**	**544313658**	**574261427**	**595384399**	**612294707**	**638379616**	**661236268**	**257006802**	**455166343**



L45

Medium Hub

IATACode	Sum of 2012	Sum of 2013	Sum of 2014	Sum of 2015	Sum of 2016	Sum of 2017	Sum of 2018	Sum of 2019	Sum of 2020	Sum of 2021_2
SAT	4036625	4005874	4046856	4091389	4179994	4521611	4844427	5022980	1919958	3677643
ABQ	2630574	2477783	2354184	2323883	2341719	2412328	2647269	2641450	868912	1688646
ANC	2249717	2325030	2381826	2525876	2563524	2556188	2642901	2713843	1157301	2184959
BDL	2647610	2681181	2913380	2926047	2982194	3214976	3330734	3323614	1150033	2273259
BOI	1307505	1313741	1378352	1487777	1633507	1777642	1943181	2057750	991241	1809000
BUR	2027203	1918011	1928491	1973897	2077892	2401106	2680240	2988720	1056838	1942417
CHS	1283970	1441415	1339326	1609888	1811695	1945699	2192893	2375968	944640	2015277
CLE	4346941	4375448	3686315	4083476	4205739	4562740	4836680	4894541	1990156	3552402
CMH	3095575	3063822	3115501	3312496	3567864	3765007	4054572	4172067	1577596	2825259
CVG	1937850	2776377	2874684	3056697	3269979	3926158	4269158	4413457	1729395	3050397
DAL	3902628	4023779	4522341	7040921	7554596	7876769	8134848	8408457	3669930	6487563
HNL	9225848	9466995	9463000	9656340	9656340	9743989	9578505	9988678	3126391	5830928
HOU	5043737	5377050	5800726	5937944	6285181	6741870	6937061	7069614	3127178	5560780
IND	3586422	3535015	3605908	3889567	4216766	4376432	4695040	4709183	1989126	3487100
JAX	2579023	2549070	2589198	2716465	2799587	2759067	3118540	3479923	1367501	2425685
MCI	4866850	4836221	4982722	5135127	5391557	5744918	5935131	5759419	2167616	3795290
MEM	3359668	2301003	1800268	1873716	2016089	2102739	2213083	2318442	1015981	1793073
MKE	3710384	3214811	3128607	3229676	3383271	3451544	3546817	3374073	1263365	2231010
MSY	4293624	4576539	4870569	5329696	5569705	6005527	6565482	6717105	2632606	4017147
OAK	4926683	4770716	5069257	5506672	5934639	6530308	6798321	6560230	2271294	4011953
OGG	2861278	2955304	3019338	3220753	3352813	3442189	3572133	3791807	1135141	2933315
OMA	2018738	1975339	2020354	2046155	2127387	2303223	2457087	2455274	1036245	1829912
ONT	2142393	1970538	2037346	2089801	2127387	2247645	2499171	2723002	1237946	2201528
ORF	1651440	1560754	1488114	1515200	1602631	1694329	1846031	1990864	884882	1658024
PBI	2796339	2844507	2926242	3113485	3100624	3166532	3263042	3460429	1518732	2567897
PDX	7142620	7852603	7378760	8340234	9071154	9435473	9940866	9797408	3455877	5759879
PIT	3892338	3812460	3827860	3890677	3986114	4327431	4670033	4715947	1742406	3069259
RDU	4490374	4482016	4673869	4954717	5401714	5851004	6416822	6919429	2337496	4311049
RNO	1683333	1671926	1911572	1669876	1771864	1953028	2048916	2162250	979957	1781785
RSW	3634152	3788870	4025959	4231134	4350650	4461304	4719568	5144467	2947139	5080805
SJC	4077654	4315839	4621003	4885690	5321603	6225148	7140616	7828885	2283186	3619690
SJU	4204478	4103197	4150828	4218785	4343354	4161587	4033412	4590117	2362851	4738725
SMF	4357899	4255145	4384616	4816440	4969366	5460526	6031630	6454413	2710342	4760275
SNA	4381172	4540628	4584147	4945175	5217242	5195047	5317149	5153276	1824836	3807205
STL	6208750	6216104	6108758	6239231	6793076	7371805	7822274	7946986	3041765	5070471
Grand Total	**127603415**	**126975111**	**129510277**	**137925203**	**144978817**	**153717889**	**162745633**	**168123968**	**65512889**	**117849807**