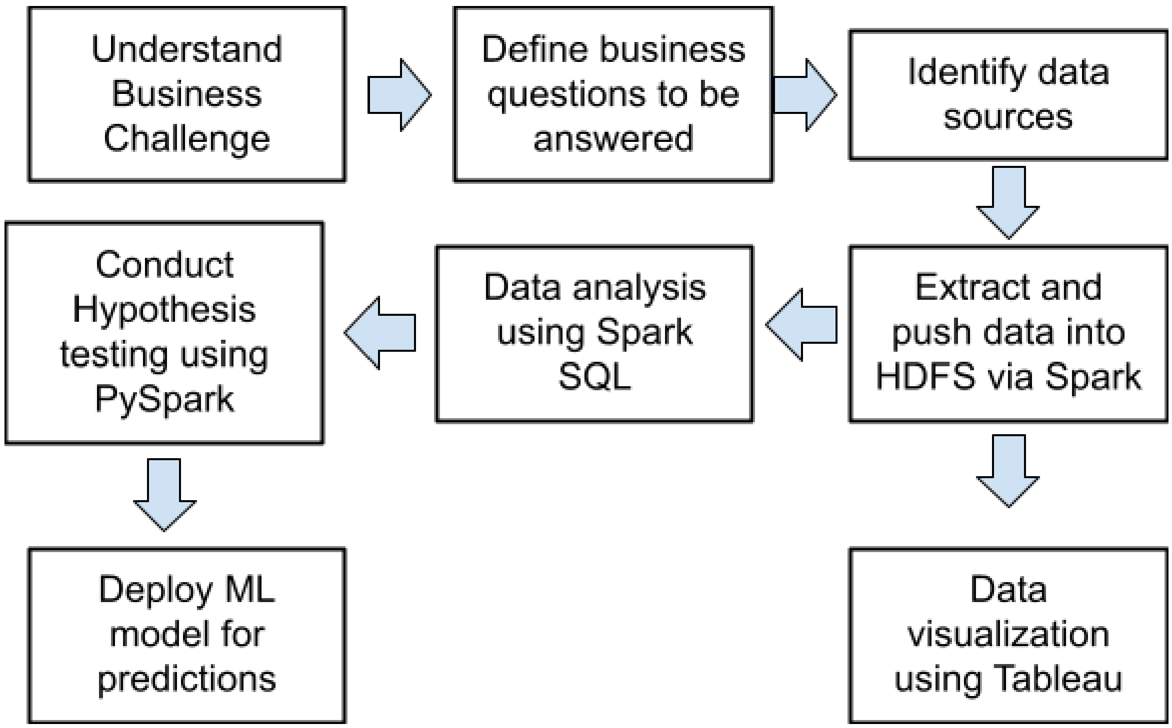
Project methodology



<u>Business Ideas</u>

There are trends that many businesses have relocated their locations recently. When a business relocates, one of the expenses of moving is airfare. The airfare for moving the employees is one of the essential parts to consider. Data suggests companies are moving to 6 major states- Texas, North Carolina, Tennessee, Virginia, Florida and Georgia and hence these are our destination states considered for analysis. Because of this, we would like to determine which airline provides a better option with respect to the price of air tickets. The main ideas of this prediction are three parts:

1. Which airline is appropriate to move the employees?
2. Which quarter is the best time to book the air tickets?
3. Do Non-stop/stop flight differences affect the airfare?

Data source: The Bureau of Transportation Statistics filtered by year 2021

There are 19 Airlines on the dataset:

- HA: Hawaiian Airlines
- AS: Alaska Airlines
- 9E: Endeavor Airlines
- DL: Delta Airlines
- AA: American Airlines
- OO: Skywest Airlines
- C5: CommutAir
- MQ: Envoy Air
- B6: JetBlue
- QX: Horizon Air
- UA: United Airlines
- OH: PSA Airlines
- G7: GoJet Airlines
- PT: Piedmont Airlines
- 3M: Silver Airways
- SY: Sun Country Airlines
- G4: Allegiant Air
- NK: Spirit Airlines
- F9: Frontier Airlines

Data Source

1. Flight price data was identified from the Bureau of Transportation Statistics and filtered by year(i.e. 2021)

Variable	Description
ItinID	*Itinerary ID*
Market_Coupons	*Number of Coupons in the Market*
Year	*Year*
Quarter	*Quarter (1-4)* [Lookup]
Origin	*Origin Airport Code*
Origin_State	*Origin State Name*
Destination	*Destination Airport*
Destination_State	*Destination State Name*
Airport_Group	*List of airports flight travel*
Flight_Change	*Whether a stop-over was present or not*
Flight	*Type of airlines* [Lookup]
Reporting_Flight	*Airline type* [Lookup]
Bulk_Fare	*Bulk Fare Indicator (1=Yes)*
Passengers	*Number of Passengers*
Fare	*Market Fare (ItinYield*MktMilesFlown)*
Distance_Group	*Distance Group, in 500 Mile Intervals* [Lookup]
Itin_Geo_Type	*Itinerary Geography Type* [Lookup]
Mkt_Geo_Type	*Market Geography Type* [Lookup]

2. Quarter wise data was extracted and pooled together using Ubuntu .txt editor

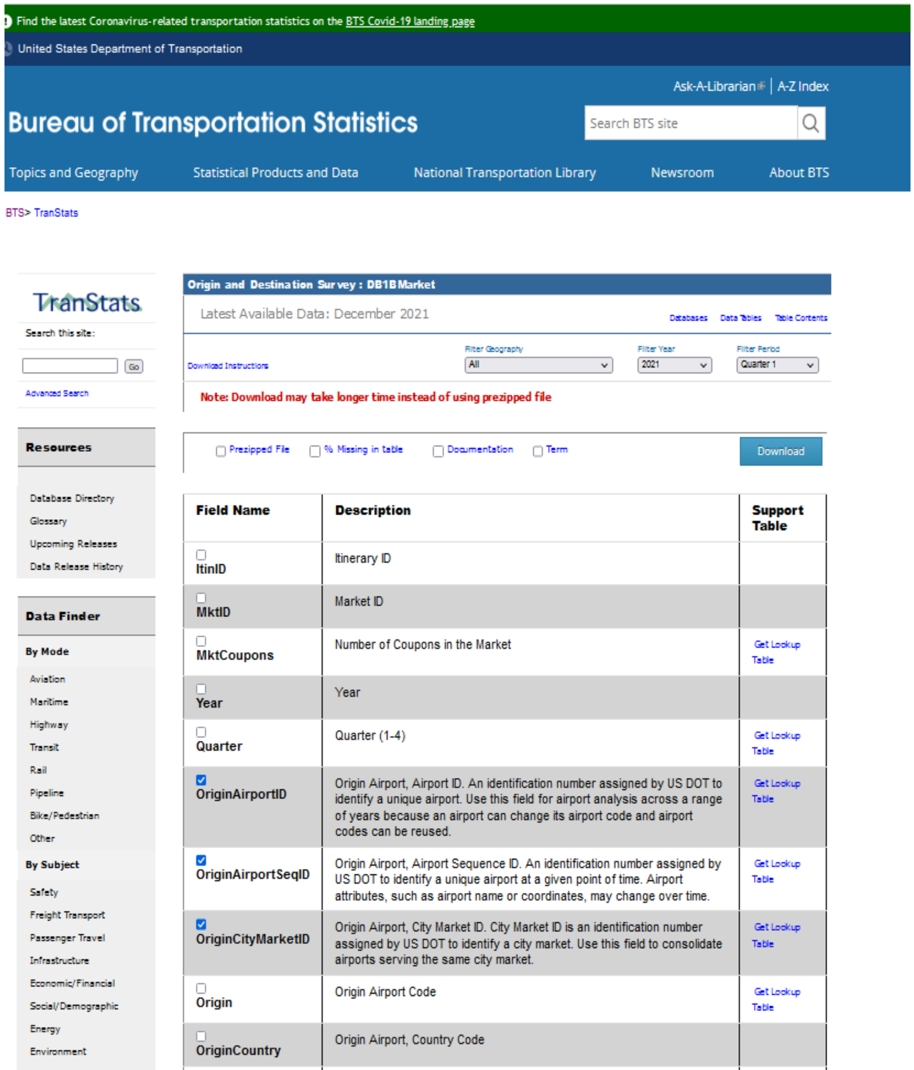


Fig: Data source screenshot

Link of dataset :
https://www.transtats.bts.gov/DL_SelectFields.aspx?gnoyr_VQ=FHK&QO_fu146_anzr=b4vtv0%20n0q%20Qr56v0n6v10%20f748rB

<u>Extract and Push data into HDFS via Spark</u>

Original data was pushed into Spark via the following steps:

1. Activate HDFS and Yarn

```
siddharth-sheth@siddharthsheth-VirtualBox:/usr/share/hadoop$ sbin/start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [siddharthsheth-VirtualBox]
siddharth-sheth@siddharthsheth-VirtualBox:/usr/share/hadoop$ sbin/start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

2. Start Spark session

```
siddharth-sheth@siddharthsheth-VirtualBox:/usr/share/hadoop$ cd
siddharth-sheth@siddharthsheth-VirtualBox:~$ cd /usr/share/spark
siddharth-sheth@siddharthsheth-VirtualBox:/usr/share/spark$ bin/beeline
Beeline version 2.3.9 by Apache Hive
beeline> siddharth-sheth@siddharthsheth-VirtualBox:/usr/share/spark$ sbin/start-ver.shser
starting org.apache.spark.sql.hive.thriftserver.HiveThriftServer2, logging to /usr/share/spark/logs/spark-siddharth-sheth-org.apache.spark.sql.hive.thriftserver.HiveThriftServer2-1-siddharthsheth-Virtual
Box.out
siddharth-sheth@siddharthsheth-VirtualBox:/usr/share/spark$ bin/beeline
Beeline version 2.3.9 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000
Connecting to jdbc:hive2://localhost:10000
Enter username for jdbc:hive2://localhost:10000: siddharth-sheth
Enter password for jdbc:hive2://localhost:10000:
2022-04-20 18:03:51,930 INFO jdbc.Utils: Supplied authorities: localhost:10000
2022-04-20 18:03:51,931 INFO jdbc.Utils: Resolved authority: localhost:10000
Connected to: Spark SQL (version 3.2.1)
Driver: Hive JDBC (version 2.3.9)
Transaction isolation: TRANSACTION_REPEATABLE_READ
```

3. Create original RDD table

<u>Code</u> *: create EXTERNAL TABLE bts_flight(ItinID int, Market_Coupons int, Year int, Quarter int, Origin string, Origin_State string, Destination string, Destination_State string, Airport_Group string, Flight_Change string, Flight string, Reporting_Flight string, Bulk_Fare int, Passengers int, Fare int, Group int, Miles int, Itin_Geo_Type int, Mkt_Geo_Type int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','*
LOCATION'file:///home/siddharth-sheth/Documents/Bigdata/Merged/BTS-Market/bts_flight';

```
0: jdbc:hive2://localhost:10000> select * from bts_flight limit 10;
```

ItinID	Market_Coupons	Year	Quarter	Origin	Origin_State	Destination	Destination_State	Airport_Group	Flight_Change	Flight	Reporting_Flight	Bulk_Fare	Passengers	Fare	Group	Miles	Itin_Geo_Type	Mkt_Geo_Type
2021118	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	1	0	2	545	2	2
2021119	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	2	32	2	545	2	2
2021120	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	6	48	2	545	2	2
2021121	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	3	49	2	545	2	2
2021122	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	1	53	2	545	2	2
2021123	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	3	58	2	545	2	2
2021124	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	1	78	2	545	2	2
2021125	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	5	88	2	545	2	2
2021126	1	2021	1	CAE	South Carolina	FLL	Florida	CAE:FLL	0.00	3M	3M	0	2	32	2	545	2	2
2021126	1	2021	1	FLL	Florida	CAE	South Carolina	FLL:CAE	0.00	3M	3M	0	2	32	2	545	2	2

```
10 rows selected (4.256 seconds)
```

4. Subset data based on identified Southern States(i.e Texas, North Carolina, Virginia, Florida, Tennessee, Georgia)

*<u>Code</u>: create table southern_states as (select * from bts_flight where Destination_State = 'Florida' or Destination_State = 'North Carolina' or Destination_State = 'Texas' or Destination_State = 'Georgia' or Destination_State= 'Tennessee' or Destination_State = 'Virginia')*

```
0: jdbc:hive2://localhost:10000> select * from southern_states limit 10;
+---------+-----------------+-------+----------+---------+---------------+---------------+--------------------+---------------+---------------+---------+-------------------+------------+-------------+--
-----+--------+--------+---------------+---------------+
| ItinID  | Market_Coupons  | Year  | Quarter  | Origin  | Origin_State  | Destination  | Destination_State  | Airport_Group  | Flight_Change  | Flight  | Reporting_Flight  | Bulk_Fare  | Passengers  | F
are  | Group  | Miles  | Itin_Geo_Type  | Mkt_Geo_Type  |
+---------+-----------------+-------+----------+---------+---------------+---------------+--------------------+---------------+---------------+---------+-------------------+------------+-------------+--
-----+--------+--------+---------------+---------------+
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 43          | 1
97   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 7           | 1
98   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 2           | 2
02   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 2           | 2
03   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 2           | 2
04   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 1           | 2
06   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 2           | 2
07   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 7           | 2
08   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 13          | 2
11   | 2      | 545    | 2               | 2             |
| NULL    | 1               | 2021  | 4        | ATL     | Georgia       | PBI           | Florida            | ATL:PBI       | 0.00          | DL      | DL                | 0          | 1           | 2
13   | 2      | 545    | 2               | 2             |
+---------+-----------------+-------+----------+---------+---------------+---------------+--------------------+---------------+---------------+---------+-------------------+------------+-------------+--
-----+--------+--------+---------------+---------------+
10 rows selected (0.38 seconds)
```

Data Analysis using Spark SQL and PySpark

1. Different destination states present

```
        ... 16 more (state=,code=0)
0: jdbc:hive2://localhost:10000> select distinct(Destination_State) from southern_states;
+--------------------+
| Destination_State  |
+--------------------+
| Texas              |
| Georgia            |
| Virginia           |
| North Carolina     |
| Tennessee          |
| Florida            |
+--------------------+
6 rows selected (3.805 seconds)
0: jdbc:hive2://localhost:10000>
```

Query result: Unique states present were identified(Texas, Georgia, Virginia, North Carolina, Tennessee and Florida)

2. Number of records after filtering for southern states

```
0: jdbc:hive2://localhost:10000> select count(*) from southern_states;
+-----------+
| count(1)  |
+-----------+
| 2885168   |
+-----------+
1 row selected (1.634 seconds)
```

Query result: There are 2.88 million flight itinerary information after filtering for Southern States.

3. To understand which distance group flew the highest passengers

Code: select Group,sum(Passengers) from southern_states
group by Group
order by sum(Passengers) DESC;

```
0: jdbc:hive2://localhost:10000> select Group,sum(Passengers) from southern_states
. . . . . . . . . . . . . . . .> group by Group
. . . . . . . . . . . . . . . .> order by sum(Passengers) DESC;
+--------+-----------------+
| Group  | sum(Passengers)  |
+--------+-----------------+
| 2      | 2321492          |
| 3      | 1923647          |
| 1      | 597231           |
| 4      | 563489           |
| 5      | 431318           |
| 6      | 119015           |
| 8      | 28134            |
| 10     | 22206            |
| 7      | 20121            |
| 9      | 10145            |
| 11     | 3950             |
| 12     | 203              |
| 13     | 36               |
| 14     | 13               |
| 18     | 11               |
| 17     | 11               |
| 16     | 5                |
| 15     | 3                |
| 19     | 2                |
+--------+-----------------+
19 rows selected (6.53 seconds)
0: jdbc:hive2://localhost:10000>
```

Query result: Group 2 and 3 have the highest flying passengers among all the distance brackets(i.e most passengers flew in the 500-1500 mile range to the Southern States)
We will now be using this subsetted data for further analysis as we need to control the distance to conduct further analysis and hypothesis testing

4. *Description of airline fare (PySpark output)*

```
In [13]: num_cols = ['MARKET_FARE']
         describe_pd(df2, num_cols)

<ipython-input-12-961d3ff04953>:24: FutureWarning: Sorting because non-concatenation axis is not aligned. A future v
ersion
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=False'.

To retain the current behavior and silence the warning, pass 'sort=True'.

  new_df = pd.concat([spark_describe, percs],ignore_index=True)

Out[13]:
```

	summary	MARKET_FARE
0	count	2885168
1	mean	216.0953635975444
2	stddev	164.3139119398339
3	min	0
4	max	8897
5	25%	120
6	50%	183
7	75%	274

Query result: The minimum, maximum and average airfare for flights in the Southern destinations were observed to be 0, 7851 USD and 199.945 USD respectively. A 0 USD airfare could be a result of multiple coupons applied.

5. ***Airline based avg prices over the year?***

Code: select Reporting_Flight, sum(Passengers),avg(Fare) as TicketPrice from final_df group by Reporting_Flight order by avg(Fare) DESC;

```
0: jdbc:hive2://localhost:10000> select Reporting_Flight, sum(Passengers),avg(Fare) as TicketPrice from final_df group by Reporting_Flight order by avg(Fare) DESC;
+-------------------+-------------------+----------------------+
| Reporting_Flight  | sum(Passengers)   |     TicketPrice      |
+-------------------+-------------------+----------------------+
| 9E                | 161265            | 231.96453738019628   |
| C5                | 40388             | 219.3761904761905    |
| DL                | 994277            | 212.922762756809     |
| AA                | 1917791           | 205.31881246940134   |
| OO                | 34155             | 204.58927071401297   |
| MQ                | 36599             | 198.097919051788     |
| B6                | 630445            | 184.05041927845105   |
| 3M                | 1763              | 179.20077821011674   |
| AS                | 4845              | 178.3798594847775    |
| OH                | 37401             | 174.99048314025967   |
| QX                | 938               | 172.16046966731898   |
| G7                | 3061              | 169.87586964481875   |
| PT                | 10627             | 166.97241165530068   |
| UA                | 10592             | 165.69552697254753   |
| SY                | 10769             | 130.21026761332604   |
| G4                | 92520             | 81.61163026182555    |
| NK                | 172323            | 68.29923686105111    |
| F9                | 85380             | 64.51409991871444    |
+-------------------+-------------------+----------------------+
18 rows selected (3.753 seconds)
```

Query result: 9E,JetBlue,Delta airlines have the highest airfares. American airlines, flew the highest number and had the 5th highest average price.

6. ***To which destination were the average Fare prices highest?***

Code: select Destination,avg(Fare) as AvgTicketPrice from final_df group by Destination order by avg(Fare) DESC LIMIT 20;

```
0: jdbc:hive2://localhost:10000> select Destination,avg(Fare) as AvgTicketPrice from final_df group by Destination order by avg(Fare) DESC LIMIT 20;
+--------------+----------------------+
| Destination  |    AvgTicketPrice    |
+--------------+----------------------+
| MCN          | 398.0                |
| MKL          | 265.0                |
| LRD          | 261.55601907032184   |
| VLD          | 261.3494884221863    |
| SJT          | 260.86106141920095   |
| TYR          | 252.88491547464238   |
| MAF          | 252.67490247074122   |
| ABY          | 251.58743842364532   |
| GGG          | 250.72214580467676   |
| BQK          | 250.38274336283186   |
| ABI          | 248.61902530459233   |
| SPS          | 245.727078891258     |
| ACT          | 245.35737527114966   |
| TRI          | 243.98735341457805   |
| EYW          | 242.91627564633362   |
| LBB          | 242.52781740370898   |
| ROA          | 242.43949044585986   |
| TLH          | 241.15346380863622   |
| MFE          | 237.92847875445995   |
| CRP          | 237.1693109700816    |
+--------------+----------------------+
20 rows selected (2.673 seconds)
```

Query results: Middle Georgia, Valdosta, Laredo, Tricities and San Angelo airport had the highest destination airfare among airports

7. Which airline flew the highest number of passengers in 2021?

Code: select Reporting_Flight,sum(Passengers) from final_df group by Reporting_Flight order by sum(Passengers) DESC;

```
0: jdbc:hive2://localhost:10000> select Reporting_Flight,sum(Passengers) from final_df group by Reporting_Flight order by sum(Passengers) DESC;
+-------------------+------------------+
| Reporting_Flight  | sum(Passengers)  |
+-------------------+------------------+
| AA                | 1917791          |
| DL                | 994277           |
| B6                | 630445           |
| NK                | 172323           |
| 9E                | 161265           |
| G4                | 92520            |
| F9                | 85380            |
| C5                | 40388            |
| OH                | 37401            |
| MQ                | 36599            |
| OO                | 34155            |
| SY                | 10769            |
| PT                | 10627            |
| UA                | 10592            |
| AS                | 4845             |
| G7                | 3061             |
| 3M                | 1763             |
| QX                | 938              |
+-------------------+------------------+
```

Query result: American, Delta and Jetblue airways had the highest flying passengers in 2021 to the Southern States of USA.

8. What distance bracket has the highest avg price range?

Code: select Group, avg(Fare),sum(Passengers) from final_df group by Group having sum(Passengers)> 10000;

```
18 rows selected (3.454 seconds)
0: jdbc:hive2://localhost:10000> select Group, avg(Fare),sum(Passengers) from final_df group by Group having sum(Passengers)> 10000;
+--------+---------------------+------------------+
| Group  |      avg(Fare)      | sum(Passengers)  |
+--------+---------------------+------------------+
| 3      | 211.7201948302179   | 1923647          |
| 2      | 189.74629699048413  | 2321492          |
+--------+---------------------+------------------+
2 rows selected (2.402 seconds)
```

Query results: Group 3 has higher average price, which seems reasonable considering group-3 has a higher distance bracket

9. Effect on price w.r.t Flight change

Code: select Flight_Change,avg(Fare),sum(Passengers) from final_df group by Flight_Change;

```
0: jdbc:hive2://localhost:10000> select Flight_Change,avg(Fare),sum(Passengers) from final_df group by Flight_Change;
+----------------+---------------------+-------------------+
| Flight_Change  |      avg(Fare)      | sum(Passengers)   |
+----------------+---------------------+-------------------+
| 1.00           | 179.98645339285162  | 35319             |
| 0.00           | 200.29697034292025  | 4209820           |
+----------------+---------------------+-------------------+
2 rows selected (3.365 seconds)
```

<u>Query results:</u> A non-stop flight has a slightly higher fare price compared to flights with stoppage by 21 USD

10. *Average ticket prices on flying territory type*

<u>Code</u>: select Itin_Geo_Type,avg(Fare),sum(Passengers) from final_df group by Itin_Geo_Type;

```
        ... 16 more (state=,code=0)
0: jdbc:hive2://localhost:10000> select Itin_Geo_Type,avg(Fare),sum(Passengers) from final_df group by Itin_Geo_Type;
+----------------+---------------------+-------------------+
| Itin_Geo_Type  |      avg(Fare)      | sum(Passengers)   |
+----------------+---------------------+-------------------+
| 1              | 209.91634417455742  | 87929             |
| 2              | 199.81325450368942  | 4157210           |
+----------------+---------------------+-------------------+
2 rows selected (1.91 seconds)
```

11. *Average ticket prices quarter wise*

<u>Code</u>: select Quarter,avg(Fare) from final_df group by Quarter;

```
0: jdbc:hive2://localhost:10000> select Quarter,avg(Fare) from final_df group by Quarter;
+----------+---------------------+
| Quarter  |      avg(Fare)      |
+----------+---------------------+
| 3        | 207.04115731910136  |
| 2        | 202.84659318979578  |
| 4        | 223.47759366462518  |
| 1        | 168.15540691965106  |
+----------+---------------------+
4 rows selected (1.776 seconds)
```

Takeaways from Data Analysis

- Distance group 2 and 3 happen to be the most popular distance bracket of traveling(500-1000 miles)
- Quarter 1 fares were the lowest among the 4 quarters
- An indirect flight has a lower fare than a direct flight
- American airlines, Delta, JetBlue and Endeavor Air were the most popular airlines for people traveling to the Southern States and also had the highest average airfares.

Hypothesis Test

We extracted the data that has the instances of which the Distance group is 2 or 3 from the original data. The reason for limiting the data is to control the distance variable.

	PASSENGERS	MARKET_FARE	MARKET_MILES_FLOWN
PASSENGERS	1.000	-0.101	-0.048
MARKET_FARE	-0.101	1.000	0.263
MARKET_MILES_FLOWN	-0.048	0.263	1.000

The above table shows the correlation between integer variables. We can see the positive correlation between MARKET_MILES_FLOWN and MARKET_FARE. As we indirectly control the distance variable, we can conduct some hypothesis tests.

What is hypothesis testing ?

A statistical hypothesis test is a method of statistical inference used to decide whether the data at hand sufficiently support a particular hypothesis.

Why do we need it ?

For the company, the main issue is to make the largest profit. It means that as the cost becomes smaller, it's better for the company, so we are conducting hypothesis tests for some variable that could affect the MARKET_FARE.

Process:

1. State null and alternate hypothesis
2. Calculate F-score
3. Calculate p score based on F-score and degrees of freedom for corresponding variables
4. Conduct Tukey-Kramer test(if Null hypothesis is rejected) to understand pairwise interactions of independent variables

Hypothesis 1:

- Quarter - Market_Fare

Null hypothesis(H0): There is no difference in air fare between different quarters in 2021(i.e., 1,2,3,4)

Alternate hypothesis(H1) : There is a difference in air fare between different quarters in 2021

Based on the Tableau charts above, many airlines do not have the records in the 2nd, 3rd, 4th quarter. The airlines normally provide the lowest airfare in the 1st quarter.

```
In [6]: df_Quarters = df2.select(col('QUARTER'),col('MARKET_FARE'))

In [7]: df_Quarters.count()
Out[7]: 1861839

In [8]: df_Quarters
Out[8]: DataFrame[QUARTER: string, MARKET_FARE: int]

In [9]: #1. Hypothesis testing for:
        #Null hypothesis(H0):  There is no difference in air fare between different quarters in 2021(i.e., 1,2,3,4)
        #Alternate hypothesis(H1) : There is a difference in air fare between different quarters in 2021

        getAnovaStats(df_Quarters)
Out[9]: (3, 1861835, 13926.350261209976, 0.021947227572958614, 0.021945640094846206)
```

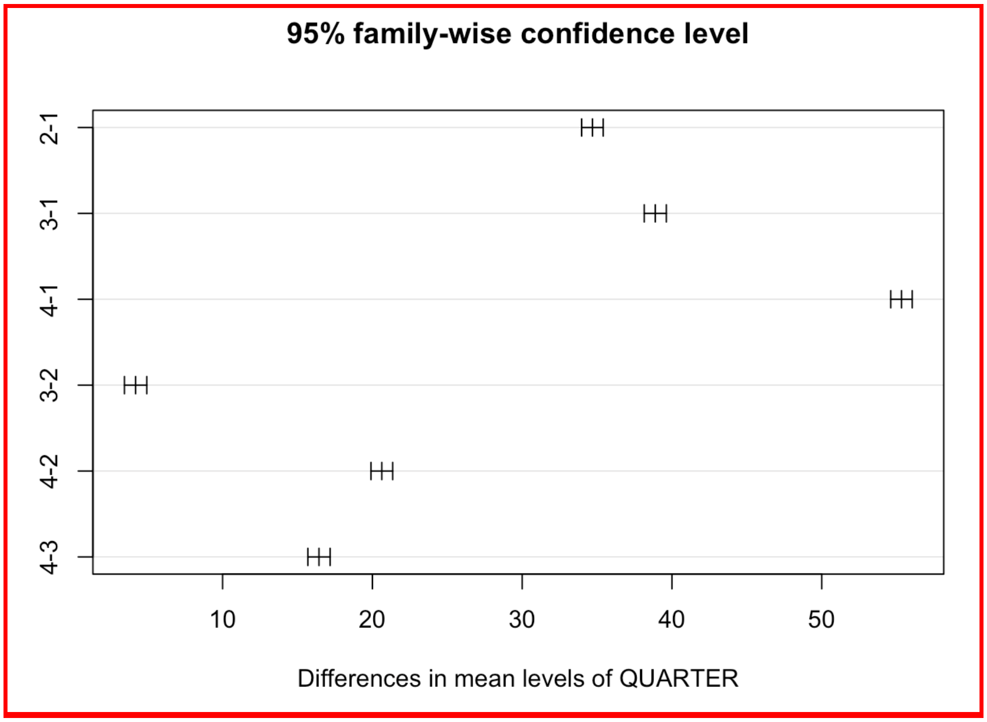
The F-score obtained via ANOVA test 13926.35

Tukey-Kramer test

```
> summary(quarter.aov)
                 Df    Sum Sq   Mean Sq F value Pr(>F)
QUARTER           3 7.855e+08 261842625   13926 <2e-16 ***
Residuals   1861835 3.501e+10     18802
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The result of the One-way ANOVA test shows that there is a difference between the quarters. Now we conduct the Tukey Kramer test to understand pairwise interactions among Quarter variables

```
> tukey.test
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = MARKET_FARE ~ QUARTER, data = df)

$QUARTER
         diff       lwr       upr p adj
2-1 34.691186 33.971625 35.410748     0
3-1 38.885750 38.147975 39.623525     0
4-1 55.322187 54.605947 56.038427     0
3-2  4.194564  3.448611  4.940517     0
4-2 20.631000 19.906340 21.355661     0
4-3 16.436436 15.693687 17.179186     0
```



<u>Inference</u> : Quarter 1 has the lowest airfare among the 4 quarters.

Hypothesis 2:

- TK_CARRIER_CHANGE - MARKET_FARE

Null hypothesis(H0): There is no difference in air fare between stoppage flights vs non-stoppage
Alternate hypothesis(H1) : There is a difference in air fare between stoppage flights v/s non-stoppage

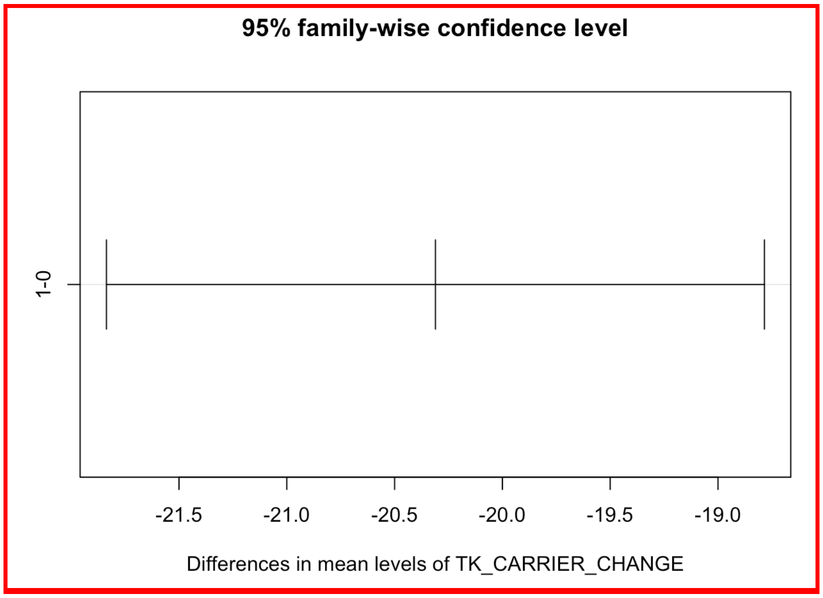
- Change - Unchange
    - Numer of Non-stop and Stop flight and average market fare
    - 0 means non-stop flight
    - 1 means stop flight

```
In [9]: df_StopChange = df2.select(col('TK_CARRIER_CHANGE'), col('MARKET_FARE'))

In [10]: df_StopChange
Out[10]: DataFrame[TK_CARRIER_CHANGE: boolean, MARKET_FARE: int]

In [11]: #4. Hypothesis testing for:
         #Null hypothesis(H0):  There is no difference in air fare between stoppage flights vs non-stoppage
         #Alternate hypothesis(H1) : There is a difference in air fare between stoppage flights v/s non-stoppage
         getAnovaStats(df_StopChange)
Out[11]: (1, 1861837, 680.4881412575361, 0.0003653593298265591, 0.0003648222262403669)
```

F-score from ANOVA test = 680.5

```
> summary(change.aov)
                       Df    Sum Sq  Mean Sq F value Pr(>F)
TK_CARRIER_CHANGE       1 1.308e+07 13076820   680.5 <2e-16 ***
Residuals         1861837 3.578e+10    19217
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> tukey.test
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = MARKET_FARE ~ TK_CARRIER_CHANGE, data = df)

$TK_CARRIER_CHANGE
         diff       lwr      upr p adj
1-0 -20.31052 -21.83653 -18.7845     0
```



Inference : With 95% confidence level, it can be stated that Stoppage flights have higher airfares than Non-stoppage flights.

Hypothesis 3:

Null hypothesis(H0): There is no difference in air fare between 4 airlines
Alternate hypothesis(H1) : There is a difference in air fare between 4 airlines

We especially check for four airlines - AA(American Airline), DL(Delta Airline), B6(JetBlue), 9E(Endeavor Air), which are the four most used airlines in our data.

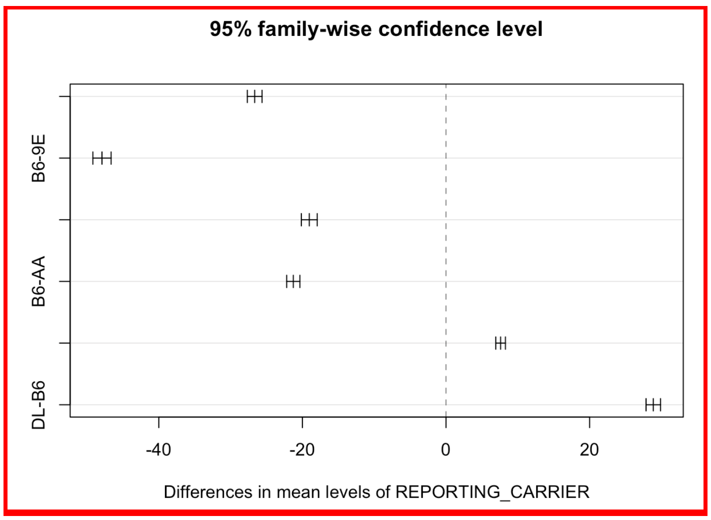
```
In [6]: df_Airline = df2.select(col('REPORTING_CARRIER'), col('MARKET_FARE'))

In [7]: df_Airline
Out[7]: DataFrame[REPORTING_CARRIER: string, MARKET_FARE: int]

In [8]: #3. Hypothesis testing for:
        #Null hypothesis(H0):  There is no difference in air fare between different airlines
        #Alternate hypothesis(H1) : There is a difference in air fare between different airlines
        getAnovaStats(df_Airline)
Out[8]: (17, 1861821, 5546.480993888241, 0.04820287211165421, 0.04819415676134175)
```

F-score obtained from ANOVA test = 5546.48099

```
> summary(reporting_carrier.aov)
                        Df    Sum Sq  Mean Sq F value Pr(>F)    
REPORTING_CARRIER        3 1.982e+08 66052945    3443 <2e-16 ***
Residuals          1631425 3.130e+10    19187                   
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The result of one way anova test shows that there is a difference between 4 airlines.

```
> tukey.test
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = MARKET_FARE ~ REPORTING_CARRIER, data = df2)

$REPORTING_CARRIER
           diff        lwr        upr p adj
AA-9E -26.64572 -27.669005 -25.622445     0
B6-9E -47.91412 -49.184301 -46.643935     0
DL-9E -19.04177 -20.147041 -17.936508     0
B6-AA -21.26839 -22.187580 -20.349207     0
DL-AA   7.60395   6.930784   8.277117     0
DL-B6  28.87234  27.862682  29.882005     0
```



Inference : Jet Blue airlines has the lowest airfare among the popular flights

Machine Learning

Problem Definition and Algorithm

As we conduct the hypothesis tests, we want to check whether we can see a result of a machine learning model that is in line with the results of the hypothesis test. We especially choose the Random Forest Regression model, because we are unsure which variables are important for predicting the airfare. That's the reason why we didn't use the linear regression model. The purpose of conducting Random Forest Regression Model is to see the feature importances.

Dataset preprocessing

```
ITIN_ID 2552089
MARKET_COUPONS 7
YEAR 1
QUARTER 4
ORIGIN 424
ORIGIN_STATE_NM 53
DEST 78
DEST_STATE_NM 6
AIRPORT_GROUP 84513
TK_CARRIER_CHANGE 2
TK_CARRIER_GROUP 350
REPORTING_CARRIER 19
BULK_FARE 2
PASSENGERS 352
MARKET_FARE 2506
DISTANCE_GROUP 19
MARKET_MILES_FLOWN 5240
ITIN_GEO_TYPE 2
MKT_GEO_TYPE 2
```

The above table shows the number of unique values of each column. For making the random forest model, the variables with many types of each category need to be excluded from the model's independent variable.

The excluded variables: ITIN_ID, ORIGIN, ORIGIN_STATE_NM, DEST, TK_CARRIER_GROUP, YEAR , AIRPORT_GROUP, DISTANCE_GROUP

StringIndexer

For encoding the categorical variables, we used the StringIndexer function. This is a label indexer that maps a string column of labels to an ML column of label indices. If the input column is numeric, we cast it to string and index the string values.

```python
import numpy as np
import pandas as pd
from pyspark.sql.functions import *
from pyspark.ml.feature import StringIndexer

#Label Encoding - string data
```

```python
In [6]: si_quarter=StringIndexer(inputCol='QUARTER',outputCol='quarter_index')
        df3=si_quarter.fit(df3).transform(df3)
        df3=df3.drop(col('QUARTER'))
```

Like this way, we converted these variables into String data type: DEST_STATE_NM, REPORTING_CARRIER, DISTANCE_GROUP, ITIN_GEO_TYPE, MKE_GEO_TYPE

Building a Model using Spark Dataframe

```python
In [12]: from pyspark.ml import Pipeline
         from pyspark.ml.regression import RandomForestRegressor
         from pyspark.ml.feature import VectorAssembler
         from pyspark.ml.evaluation import RegressionEvaluator
         from pyspark.ml.tuning import ParamGridBuilder
         from pyspark.ml.tuning import CrossValidator
```

MLlib is Spark's machine learning library. Its goal is to make practical machine learning scalable and easy.

```python
In [13]: from pyspark.sql.functions import *
         df3=df3.withColumnRenamed("MARKET_FARE", "label")
```

For some reason, when fitting the model , we saw an error that the name of the dependent variable is not 'label', so we changed the name of the dependent variable as 'label'.

VectorAssembler

VectorAssembler is a transformer that combines a given list of columns into a single vector column. It is useful for combining raw features and features generated by different feature transformers into a single feature vector, in order to train ML models like logistic regression and decision trees.

```
In [14]: feature_list=[]
         for col in df3.columns:
             if (col =='label'):
                 continue
             else:
                 feature_list.append(col)

         assembler=VectorAssembler(inputCols=feature_list, outputCol='features')
```

For building the machine learning model using pyspark, we need to use vectorAssembler to combine all the values of each column into one list. We rename the output as 'features' column.

```
In [66]: from pyspark.ml.tuning import ParamGridBuilder
         import numpy as np

         paramGrid = ParamGridBuilder()\
         .addGrid(rf.numTrees, [int(x) for x in np.linspace(start=30, stop = 50, num=1)])\
         .addGrid(rf.maxDepth, [int(x) for x in np.linspace(start=15, stop=30, num=3)])\
         .build()

In [67]: from pyspark.ml.tuning import CrossValidator
         from pyspark.ml.evaluation import RegressionEvaluator

         crossval=CrossValidator(estimator=pipeline\
                                 ,estimatorParamMaps=paramGrid\
                                 , evaluator=RegressionEvaluator()\
                                 , numFolds=3)
```

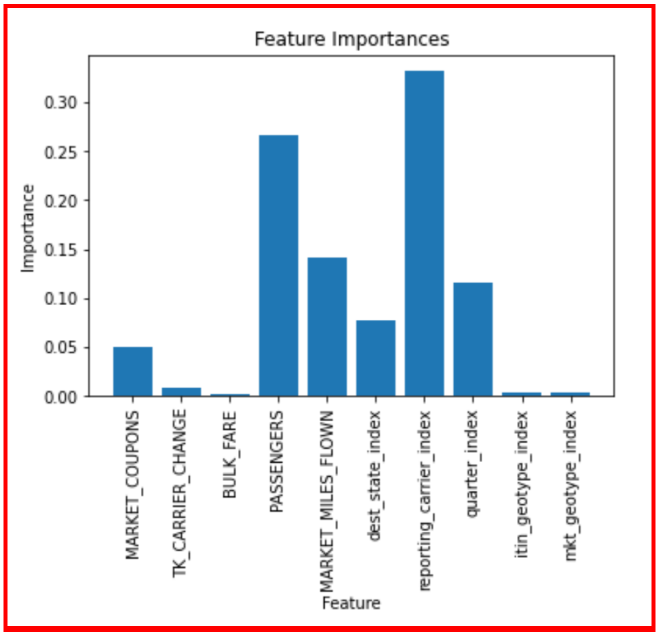
We build a cross validation to get better parameters. The number of the folds are 3.

Fitting the model and Results of the model

```
In [68]: # Splitting the data
         train, test = df3.randomSplit([0.7,0.3])

In [69]: cvModel=crossval.fit(train)

In [70]: predictions=cvModel.transform(test)

In [81]: evaluator=RegressionEvaluator(labelCol='label',predictionCol='prediction',metricName='r2')

In [82]: r2=evaluator.evaluate(predictions)

In [73]: rfPred=cvModel.transform(test)

In [83]: print(r2)
         0.1263582478936628
```

The value of R^2 is only 0.12, so we can say that the quality of the model is not good. There could be many reasons, the exclusion of the variables that have many types of categories could be the one reason for the poor R^2 value.

To check the feature importances using matplotlib, the spark dataframe needs to be converted to pandas dataframe. Since it is costly to convert the whole dataframe into pandas dataframe, we select the columns that we need for visualization then convert using toPandas function: label and prediction column.



Even though the R^2 is low, we can check the importances of each variable. The importance of reporting_carrier_index, which stands for the types of carrier, has the highest feature importance. The variable that ranked second in importance is PASSENGERS. The variable that ranked third is MARKET_MILES_FLOWN, and ranked fourth is quarter_index.

There could be many reasons for the poor quality. One of the reasons could be due to the exclusion of variables that have many types of categorical values. There could be unknown variables that affect airfare other than the remaining variables.

Conclusion

After conducting three hypothesis tests, we could verify some insights, which are directly related to the air fare.

1. There is a difference in airfare between different quarters in 2021.

The result of the Tukey-Kramer test showed that the airfare in the first quarter is the cheapest among all the quarters.

2. There is a difference in airfare between stoppage flights v/s non-stoppage.

The result of the Tukey-Kramer test showed that the airfare of non-stoppage flights is higher than that of stoppage flights.

3. There is a difference in airfare between 4 airlines: AA(American Airline), DL(Delta Airline), B6(JetBlue), 9E(Endeavor Air)

The result of the Tukey-Kramer test showed that using JetBlue Airlines is more economical than using the other three airlines.

Even though the accuracy of the random forest regression model is low, we could find that among the remaining variables, choosing the airline is most important for the airfare, and the distance is important for the airfare too. These facts don't conflict with common sense that we have generally known. Low accuracy could imply that there will be other variables that can be more related to the airfare. If the company wants to know and predict the airfare more accurately, it is necessary to investigate more detailed data.

References

CNBC. (2021, 07 13). America's Top States for Business 2021. *CNBC*. Retrieved from https://www.cnbc.com/2021/07/13/americas-top-states-for-business.html

Genovese, D. (2020, 12 26). Where corporations and consumers moved in 2020. *FOX BUSINESS*. Retrieved from https://www.foxbusiness.com/lifestyle/where-americans-moved-in-2020

PODS. (2020, 05 01). *PODS*. Retrieved from PODS FOR BUSINESS BLOG: https://www.pods.com/business/blog/10-best-cities-us-move-business/

Statistics, B. o. (2021). *Origin and Destination Survey : DB1BTicket.* Washington, DC: Bureau of Transportation Statistics. Retrieved from https://www.transtats.bts.gov/DL_SelectFields.aspx?gnoyr_VQ=FKF&QO_fu146_anzr=b4vtv0+n0q+Qr56v0n6v10+f748rB

More Tableau Tables

https://public.tableau.com/app/profile/chulhwan.kum2107/viz/BigDataProject2/QuarterlyMarketFareperDestinationStates?publish=yes