Amazon Flex Data Science & Analytics Project

By Juan Moctezuma-Flores

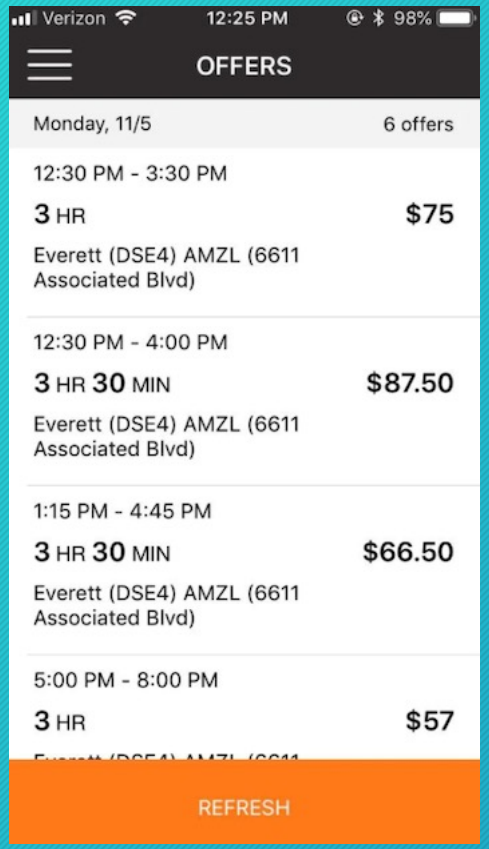
Contents

What is Amazon Flex?

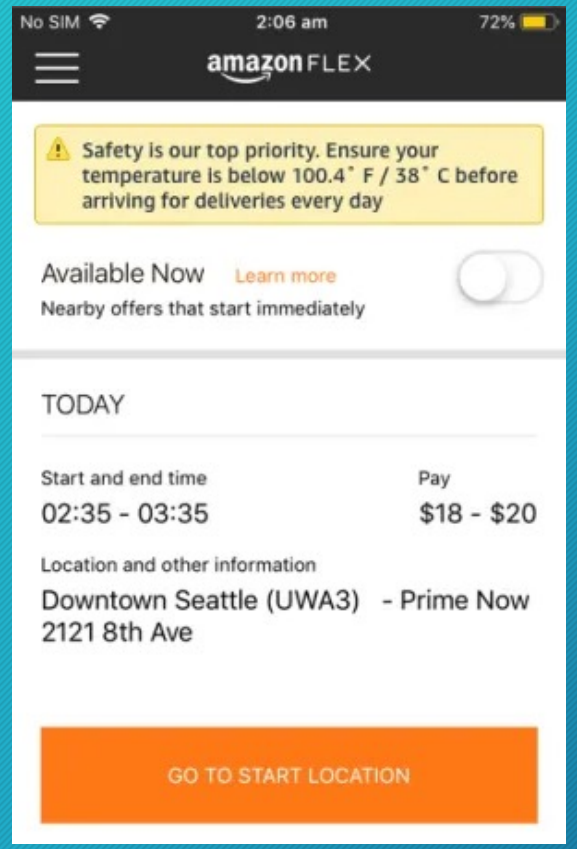
- Amazon's Flex program hires qualified individuals as delivery partners or independent contractors to deliver Amazon orders to homes, residential complexes, Amazon Lockers, small businesses and apartment buildings located in nearby neighborhoods or districts.
- Every contractor uses his/her vehicle to work the available block that he/she selects throughout Amazon Flex's mobile application. Once the block and starting point is chosen in the *Job Offers* section, the delivery partner must show up to the selected warehouse to load the customers' orders.
- Routes / delivery locations associated with each block are randomly assigned to contractors once these individuals arrive to the loading zone.

Amazon Flex's Mobile App Icon.

How Does This Work?



Step 1: You select your shift in the mobile App's Offers' section.



Step 2: Go to the warehouse and get checked-in by an associate.

Step 3: You scan your route and packages, then load the orders and start delivering.

Step 4: You return any undelivered products to the warehouse at the end of the block if necessary.

What is This Project About?

- This is a technical Data Science and Analytics project based on Amazon Flex's logistics.
- The main objectives for this project are the following:
  - A) Define one Key Performance Indicator (KPI) based on several factors affecting on-time deliveries and the contractor's performance; and generate KPI metrics.
  - B) Create an ETL Data Pipeline that creates an output (clean data file) based on the project's raw data (information manually compiled by author).
  - C) Find how much money should Amazon-Flex Delivery Partners driving 4-cylinder vehicles spend on gas before working shifts or blocks for Amazon.
  - D) Use statistics and data science (or machine learning) techniques with Python. Results are displayed throughout analytics dashboard (visual summary).
  - E) Use Python to create reports and graphs or plots in Jupyter Notebooks.

amazon FLEX

Facts About the Data Sources

- Every evening block (shift) performed during this project began around 5:00 PM and intended to last 3 hours or less.
- The term *Raw Data* stands for every bit of information that was manually compiled into a spreadsheet. The Excel document consist of 31 columns (from A to AE) that gets filled after a block / shift / route is completed.
- Please note that every row represents a block.
- Raw data for this project originate from the following sources:

A) Author's vehicle odometer – for mileage calculations.

B) Amazon Flex's mobile application – for delivery data related.

C) Google Maps – for approximate distance (in miles) calculation.

D) Gas station pump's dashboard (or printed receipts) – for fuel data.

E) Amazon Flex Summary – for weekly performance and delivery data.

How is Raw Data Organized? (Part I)

- The following represents the columns A – D *(From AMAZON-FLEX_RAW_DATA)*. Information related with these headers is stored on Amazon's platform:

  A. DATE: Day in which the block was performed.

  B. WAREHOUSE: Shift's starting point where packages are scanned and loaded or collected.

  C. START_TIME: Starting point for selected blocks or shifts.

  D. SHIFT_DURATION_HR: How many hours did the shift last according to the contractor.

How is Raw Data Organized? (Part II)

- The following represents the columns E – H *(From AMAZON-FLEX_RAW_DATA)*. Information related to these headers is directly related to vehicle and mileage data:

E. OPERATING_VEHICLE: Car used during block.

F. STARTING_MILEAGE: Vehicle's mileage recorded at starting time.

G. FINAL_MILEAGE: Vehicle's mileage recorded after the block is completed.

H. ROUTE_MILEAGE: Difference subtracted between ENDING_MILEAGE minus STARTING_MILEAGE.

A) Modern Car Dashboard.

How is Raw Data Organized? (Part III)

- The following represents the columns I and J *(From AMAZON-FLEX_RAW_DATA)*. Information from these headers is directly related to destination data :

I. NEIGHBORHOOD(S): Delivery destination(s). Contractors are guided to a set of addresses (by the App) within the different areas / districts / neighborhoods.

J. WHS-NBHD_MIN-APPROX-DIST: Warehouse (WHS) to neighborhood (NBHD) minimum (MIN) approximate (APPROX) distance (DIST) in miles. Google Maps is used to estimate the distance from the starting point to the first delivery neighborhood.

A) San Diego Downtown Area.

How is Raw Data Organized? (Part IV)

- The following represents the columns K – Q *(From AMAZON-FLEX_RAW_DATA)*. Information from these headers is directly related to fuel data:

K. FUEL_TOTAL: Fuel expense in USD that occurred before the block.

L. FUEL_GAL: Volume of gas (in gallons) injected to contractor's vehicle proportional to FUEL_TOTAL.

M. PRICE_PER_GAL: Price per gallon in USD recorded from specific gas station fuel category.

N. OIL_COMPANY: Gas station's brand.

O. GAS_CATEGORY: Gas choice from specific station. For instance, regular gas vs premium.

P. GAS_STATION_ADDRESS: Address within warehouse's range.

Q. GS-WHS_DIST: Exact distance (in miles) on Google Maps between gas station (GS) and warehouse (WHS).

How is Raw Data Organized? (Part V)

- The following represents the columns R – T *(From AMAZON-FLEX_RAW_DATA)*. Information from these headers is directly related to package data:

R. TOTAL_PKGS: Number of packages provided by Amazon's warehouse Associates. Package number is recorded on Amazon's performance report sent via e-mail.

S. ON-TIME_PKGS: Number of packages delivered on-time. Data gets recorded on Amazon's performance report sent via e-mail. Clients confirm arrival through their customer app as well.

T. RETURNED_PKGS: Number of packages returned to the warehouse. This situation is not common, but reasons could vary. Some examples are being unable to access a building, the Amazon Locker is full, it's simply too late at night, or customer is absent during arrival and requested his/her unattended order to be returned to the warehouse. Data gets recorded on Amazon's performance report sent via e-mail.

How is Raw Data Organized? (Part VI)

- The following represents the columns U & V *(From AMAZON-FLEX_RAW_DATA)*. Information from these headers is directly related to package data:

U. LATE_PKGS: Number of packages that surpassed the estimated delivery (or product arrival) time frame. Data gets recorded on Amazon's performance report sent via e-mail.

V. MISSING_ORDERS: This case applies only for packages that were delivered but the client reports his/her product as a no-show. In other words, packages that were most likely stolen. Data gets recorded by Amazon.

A) Packages being delivered.

How is Raw Data Organized? (Part VII)

- The following represents the columns W – Z *(From AMAZON-FLEX_RAW_DATA)*. Data is directly related to calculated performance ratios:

W. ON-TIME_RATIO: This percentage comes from the number of on-time packages divided by the total number of given packages (minus the returned number of packages). Returned packages do not get included.

X. PDD_FAIL_RATIO: Promise Delivery Date (PDD) fail percentage applies only for returned packages, which is divided by the total number of given packages.

Y. LATE-PKGE_RATIO: This percentage comes from the number of late packages divided by the total number of given packages (minus the returned number of packages). Returned packages do not get included.

Z. RCVD-PKGE_RATIO: This percentage comes from the number of received packages (on-time or late) minus both returned packages or missing orders divided by the total number of given packages.

How is Raw Data Organized? (Part VIII)

- The following represents the columns AA – AE *(From AMAZON-FLEX_RAW_DATA)*. Information from these headers is directly related to other performance parameters:

AA. ON-TIME_ARR (Y/N): Did the contractor (author) arrived early to the warehouse? 'Y' stands for yes and 'N' for no.

AB. TRAFFIC (Y/N): Was there traffic that day? 'Y' stands for yes and 'N' for no.

AC. WHS_DELAY (Y/N): Was there a delay on behalf of the warehouse's (WHS) personnel? 'Y' stands for yes and 'N' for no.

AD. BAD_WEATHER (Y/N): Was there rain, fog, hail? Weather conditions may affect the contractor's performance.

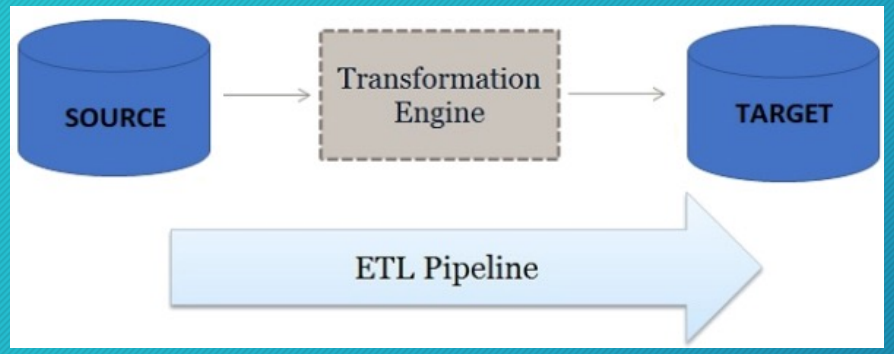
'Y' stands for yes and 'N' for no.

AE. OTHER_DELAYS (Y/N): Was there any other inconvenience that affected performance? 'Y' stands for yes and

'N' for no.

How is Raw Data Processed? (Part I)

- An Extract Transform Load (ETL) Pipeline is used during this project, and it was made with Python 3 within the Jupyter Notebook platform.
- The Raw Data File (*Amazon-Flex_RDF.csv*) is initially stored in a folder, then the script extracts the data from the spreadsheet (input), cleans or transforms some columns and loads the cleaned data into a new CSV file in the 'output' folder. The output file is named *Amazon-Flex_CDF.csv* (Clean Data File).
- The ETL runs occasionally after more data gets compiled.

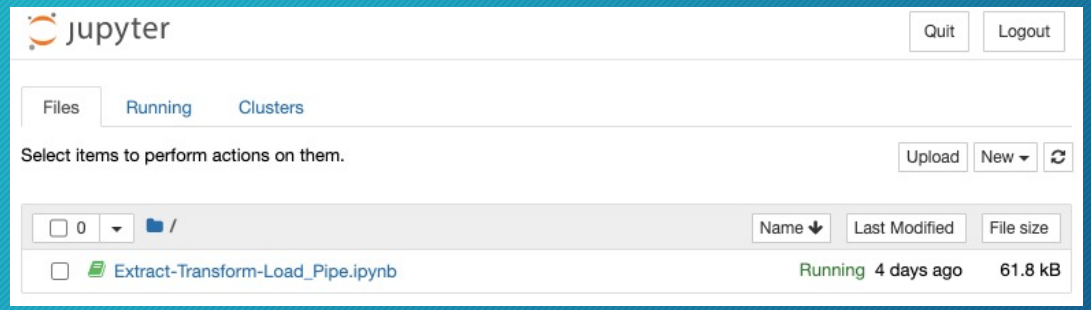


A) Visual definition of an ETL Pipeline – Definition by Databricks.

B) An ETL pipe is an abstract concept unlike an actual pipeline.

How is Raw Data Processed? (Part II)

- The ETL pipeline consists of one file. Each stage consists of several lines of code. The 'Extract' stage uses Pandas library to read the raw data. The cleaning or 'transforming' stage, for instance, removes '$' from fuel related columns, changed some headers and replaced columns containing 'YES' or 'NO' values with binary numbers. The 'loading' stage simply moves clean data to another file.
- The 'source' is the raw data csv file, the 'transformation engine' is the Python script or pipeline and the 'target' is the cleaned data csv file.



A) The ETL Pipeline gets ran locally on Jupyter Notebook.

How is Raw Data Processed? (Part III)

```
In [5]: df.dtypes

Out[5]: DATE                          object
        WAREHOUSE                     object
        START_TIME                    object
        SHIFT_DURATION_HR            float64
        OPERATING_VEHICLE             object
        STARTING_MILEAGE               int64
        FINAL_MILEAGE                  int64
        ROUTE_MILEAGE                  int64
        NEIGHBORHOOD(S)               object
        WHS-NBHD_MIN-APPROX-DIST     float64
        FUEL_TOTAL                    object
        FUEL_GAL                     float64
        PRICE_PER_GAL                 object
        OIL_COMPANY                   object
        GAS_CATEGORY                  object
        GAS_STATION_ADDRESS           object
        GS-WHS_DIST                  float64
        TOTAL_PKGS                     int64
        ON-TIME_PKGS                   int64
        RETURNED_PKGS                  int64
        LATE_PKGS                      int64
        MISSING_ORDERS                 int64
        ON-TIME_RATIO                 object
        PDD_FAIL_RATIO                object
        LATE-PKGE_RATIO               object
        RCVD-PKGE_RATIO               object
        ON-TIME_ARR (Y/N)             object
        TRAFFIC (Y/N)                 object
        WHS_DELAY (Y/N)               object
        BAD_WEATHER (Y/N)             object
        OTHER_DELAYS (Y/N)            object
        dtype: object
```

A) Data types of Raw Data File. Each name represents a column.

```
In [22]: df_cleaned.head(n=50)
```

Out[22]:

RETURNED_PKGS	LATE_PKGS	MISSING_ORDERS	ON-TIME_RATIO	PDD_FAIL_RATIO	LATE-PKGE_RATIO	RCVD-PKGE_RATIO	ON_TIME_ARR	NO_TRAFFIC	NO_WHS_DELAY
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
2	4	0	0.870968	0.060606	0.129032	0.939394	1	1	0
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
1	0	0	1.000000	0.032258	0.000000	0.967742	1	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	0	1	1
0	0	0	1.000000	0.000000	0.000000	1.000000	1	1	1

B) The screenshot displays some of the transformed / cleaned data. These reflect the same results as the new CSV in the 'output' folder. Not all columns required processing.

REST API

- A Representational State Transfer (REST) Application Programming Interface (API) was created (in Python 3) to reflect the results from the cleaned data on the web or browser.
- This API is NOT part of the of the project's ETL and it's publicly not available. However, a PDF and one JSON file representing how the API looks like and its results, respectively, are provided on the project's Github repository.
- The REST API was created in Visual Studio Code, unlike the ETL pipeline, which was built on Jupyter Notebook. Why? Because Visual Studio can run it via a local server.


```
{
    "0": {
        "DATE": "1/2/21",
        "WAREHOUSE": "DSD3 (National City)",
        "START_TIME": "5:00 PM",
        "SHIFT_DURATION_HR": 3,
        "OPERATING_VEHICLE": "Sedan (Jetta VW)",
        "STARTING_MILEAGE": 94918,
        "FINAL_MILEAGE": 94939,
        "ROUTE_MILEAGE": 21,
        "NEIGHBORHOOD(S)": "BONITA / NATIONAL CITY",
        "WHS-NBHD_MIN-APPROX-DIST": 4.5,
        "FUEL_TOTAL": 11.04,
        "FUEL_GAL": 3.085,
        "PRICE_PER_GAL": 3.5789999999999997,
        "OIL_COMPANY": "Shell Oil Company",
        "GAS_CATEGORY": "V-Power (Premium)",
        "GAS_STATION_ADDRESS": "3230 National City Blvd",
        "GS-WHS_DIST": 1.3,
        "TOTAL_PKGS": 54,
        "ON-TIME_PKGS": 54,
        "RETURNED_PKGS": 0,
        "LATE_PKGS": 0,
        "MISSING_ORDERS": 0,
        "ON-TIME_RATIO": 1,
        "PDD_FAIL_RATIO": 0,
        "LATE-PKGE_RATIO": 0,
        "RCVD-PKGE_RATIO": 1,
        "ON-TIME_ARR (Y/N)": 1,
        "TRAFFIC (Y/N)": 1,
```


C) 1st item on project's REST API has an index of zero.

KPI - Introduction

- A Key Performance Indicator (KPI) is a quantifiable measure used to evaluate the success of a company, employee, etc. (*definition by Lexico*).
- The author's KPI is aligned to one 'key' business objective, which is to measure the success (as percentage) of on-time deliveries during each work block.
- This KPI takes unpredictable factors into account too, because uncertainty may negatively impact the driver's or contractor's performance. Why? Customers expect their package(s) to arrive on-time but most of them are not aware of the external factors that directly affect the delivery partner's job. For example, if there was a car accident and the contractor must drive through heavy road congestion the order(s) might arrive late.

KPI's Definition - NOTE

- Please refer to the AMAZON-FLEX_RAW_DATA file (RDF) for a better understanding of terms, naming convention, concepts or abbreviations.
- Variables plugged into the Key Performance Indicator (KPI) are directly linked to the RDF (columns W, Z, and AA – AE). Headers for cols. AA – AE are renamed on *Amazon-Flex_CDF.csv* (Clean Data File).
- The KPI's definition and formula are defined by the author and not by Amazon. The KPI's report uses data from *Amazon-Flex_CDF.csv*.

W	Z	AA	AB	AC	AD	AE
ON-TIME_RATIO	RCVD-PKGE_RATIO	ON-TIME_ARR (Y/N)	TRAFFIC (Y/N)	WHS_DELAY (Y/N)	BAD_WEATHER (Y/N)	OTHER_DELAYS (Y/N)
100%	100%	YES	NO	NO	NO	NO
100%	100%	YES	NO	NO	NO	NO
87%	94%	YES	NO	YES	NO	NO
100%	100%	YES	NO	NO	NO	NO
100%	100%	YES	NO	NO	YES	NO
100%	100%	YES	YES	YES	NO	NO

A) Amazon-Flex_RDF.xlsx contains the manually compiled data set for this project. The defined KPI uses data from each row from the columns displayed on the image.

KPI's Definition (Part I)

- The author of this project decided to define one Key Performance Indicator (KPI) for Amazon contractors, as the average of the following:
  A. Column AA (ON-TIME_ARR) is represented by 1 (True) or 0 (False).
     i. Reliability. Did the contractor arrive on-time to the warehouse? If yes, then 1 will be added into the KPI 'formula'.
  B. Column AB – AE data, in which 1 is True and 0 is False.
     i. Favorable road traffic conditions are labelled as NO_TRAFFIC on the Clean Data File (CDF). Did traffic cause late-deliveries? If there was NO traffic, then 1 will be added to the KPI.
     ii. Amazon warehouse's on-time dispatch are labelled as NO_WHS_DELAY on the CDF. Were the Associates on-time? If there were NO delays at the warehouse, then 1 will be added to the KPI.
     iii. Favorable weather conditions are labelled as NO_BAD_WEATHER on the CDF. Was there rain, hail, fog during the block or shift? If weather was NOT an inconvenience, then 1 will be added to the KPI.
     iv. The absence of delays recorded are labelled as NO_OTHER_DELAYS on the CDF. Were there any other incidents? If there were NO delays or any other inconvenience, then 1 will be added to the KPI.

KPI's Definition (Part II)

C. Column W (ON-TIME_RATIO) data, where the decimal (result) gets added to the KPI.
   i. Ratio of on-time delivered packages divided by total number of packages (boxes or envelopes) that were assigned during that shift. Returned packages get excluded or subtracted from the denominator.

D. Column Z (RCVD-PKGE_RATIO) data, where the decimal (result) gets added to the KPI.
   i. Ratio of received packages (confirmed by client) minus both returned items or missing orders divided by the total number of packages (boxes or envelopes) that were assigned during that shift. On-time or late deliveries are excluded from the equation.

- Regarding bullet points C and D, ratios can't be higher that 1 or 100%.
- Column X and Y (PDD_FAIL_RATIO and LATE-PKGE_RATIO) data doesn't get added to the formula. Why? mathematically, on-time and received-package ratios already take data from column X & Y into account.

KPI's Mathematical Definition (Part I)

- The KPI formula is not designed to measure monthly results, but individual outcomes (blocks). The reason being that each shift or block may vary in route, number of packages, and even the number of times worked per month may not be the same, therefore averages wouldn't weigh the same.
- *See slides 13 & 14 for the columns' name reference.*
- The defined KPI formula (average) results in a decimal and it's a unitless number. The highest output can only be 1 or 100% success.
- The following variables represent the numerator and denominator for each row's ratios (where N is equal to some integer or natural number):
  i. ON-TIME_PKGS = N_{OTP}
  ii. TOTAL_PKGS = N_{TOT}
  iii. RETURNED_PKGS = N_{RP}
  iv. MISSING_ORDERS = N_{MO}

KPI's Mathematical Definition (Part II)

- *See slides 13 & 14 for the columns' name reference.*
- The following variables represent the numerical input from each row (*Please review N variables from previous slide if necessary*) :
  i. ON-TIME_RATIO = $Y_{OTR} = (N_{OTP}) / (N_{TOT} - N_{RP})$ which is less than or equal to (<=) 1
  ii. RCVD-PKGE_RATIO = $Y_{RPR} = (N_{TOT} - (N_{RP} + N_{MO})) / (N_{TOT})$ which is <= 1
  iii. ON-TIME_ARR = X_{OTA} = 1 or 0
  iv. NO_TRAFFIC = X_{NT} = 1 or 0
  v. NO_WHS_DELAY = X_{NWD} = 1 or 0
  vi. NO_BAD_WEATHER = X_{NBW} = 1 or 0
  vii. NO_OTHER_DELAYS = X_{NOD} = 1 or 0
- Note that every Y's & X's (variables) subscript represents the initial letters of every column's name that will be summed in the KPI. Letter 'N' in every subscript stands for 'No', for instance, X_{NT} where NT stands for 'No-Traffic'.

KPI's Mathematical Definition (Part III)

- *See slides 23 & 24 for the columns' name reference.*
- Every X variable included can have a value of 0 or 1. Y variables can have a value ranging from 0 to 1; Y can be a decimal number. The sum is divided by 7 since we are looking at the average that defines how efficient the contractor performed on 1 block or shift. The delivery success KPI formula is defined as the following:

$$KPI = [(X_{NBW} + X_{NOD} + X_{NT} + X_{NWD} + X_{OTA} + Y_{OTR} + Y_{RPR}) \div 7] \times 100\%$$

where $Y_{RPR} = (N_{TOT} - (N_{RP} + N_{MO})) / (N_{TOT}) <= 1$

where $Y_{OTR} = (N_{OTP}) / (N_{TOT} - N_{RP}) <= 1$

where $X = 1$ or $X = 0$

KPI Example (Part I)

- During 01/02/2021, 54 out of 54 packages were delivered on-time on Bonita, CA. Every client confirmed that they all received their order. The contractor arrived on-time, there was no traffic, no warehouse delays, no bad weather, and no other delays. *See AMAZON-FLEX_RAW_DATA File – row 2 for reference.*
- Therefore, the following numbers get plugged in:
  - TOTAL_PKGS = $N_{TOT} = 54$
  - ON-TIME_PKGS = $N_{OTP} = 54$
  - RETURNED_PKGS = $N_{RP} = 0$
  - MISSING_ORDERS = $N_{MO} = 0$
  - $Y_{OTR} = (N_{OTP}) / (N_{TOT} - N_{RP}) = (54) / (54 - 0) = 1$
  - $Y_{RPR} = (N_{TOT} - (N_{RP} + N_{MO})) / (N_{TOT}) = (54 - (0 + 0)) / (54) = 1$
  - $X_{NBW} = X_{NOD} = X_{NT} = X_{NWD} = X_{OTA} = Y_{OTR} = Y_{RPR} = 1$

KPI Example (Part II)

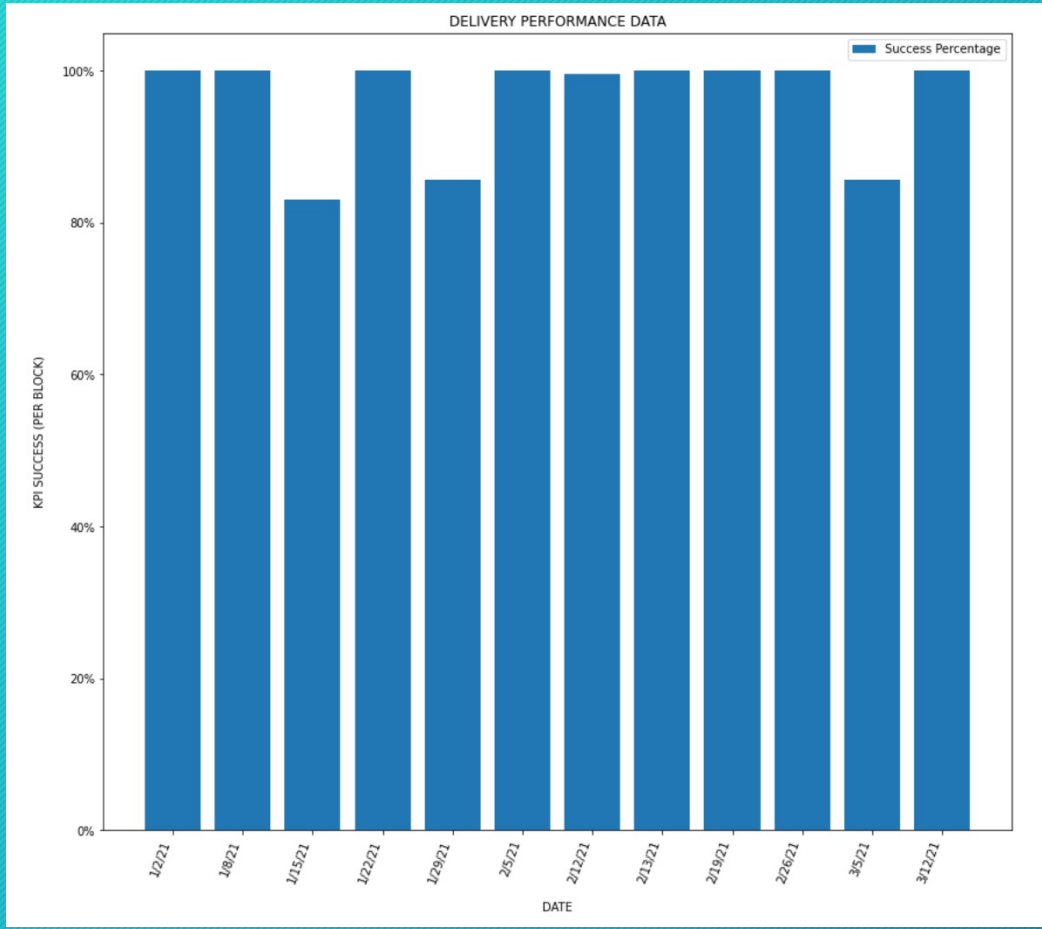
- We now plug in the data into the KPI formula, which is the following:
  $KPI = [(X_{NBW} + X_{NOD} + X_{NT} + X_{NWD} + X_{OTA} + Y_{OTR} + Y_{RPR}) \div 7] \times 100\%$
  - The computation looks like [(7) ÷ 7] x 100% = 100% on-time delivery or success.
  - What's the analysis? There were no external factors that negatively impacted performance during that day. Although 54 packages sounds like a lot, the block was successfully performed!

=AVERAGE(Table2[@[ON-TIME_RATIO]:[NO_OTHER_DELAYS]])

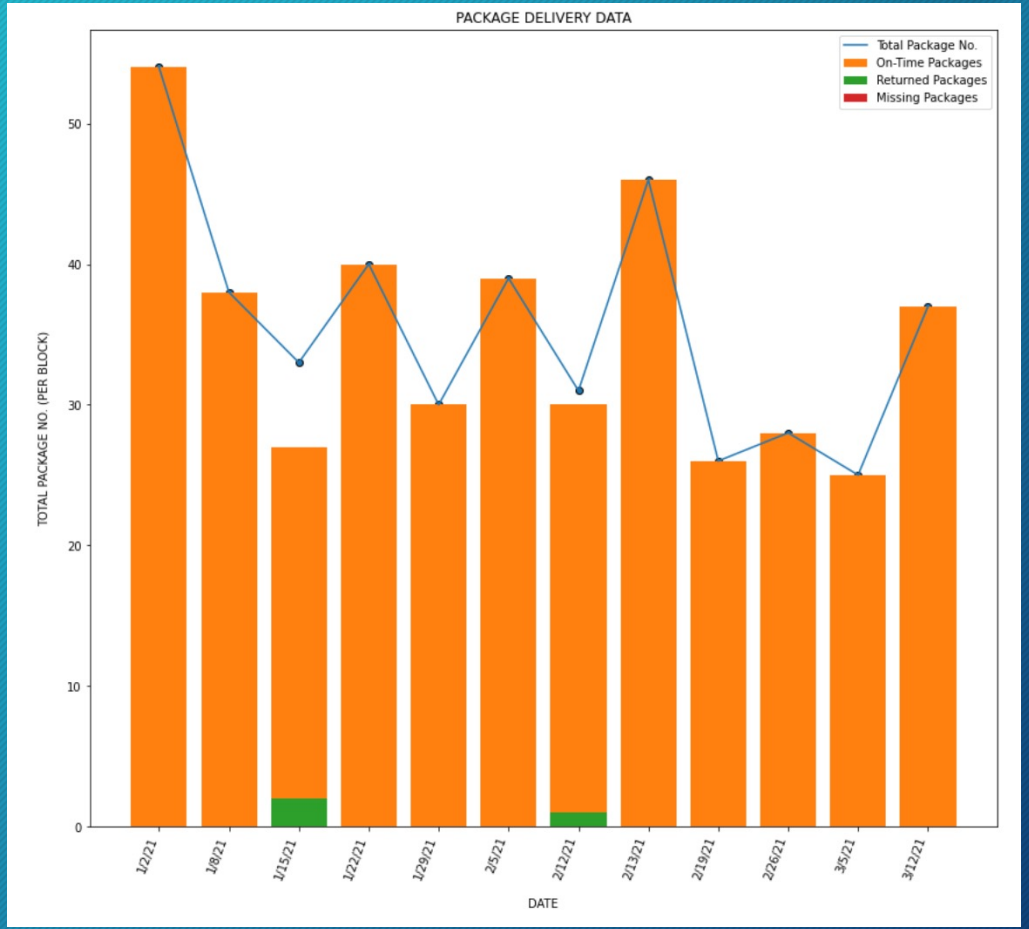
G	H	I	J	K	L	M	N
ON-TIME_RATIO	RCVD-PKGE_RATIO	ON-TIME_ARR	NO_TRAFFIC	NO_WHS_DELAY	NO_BAD_WEATHER	NO_OTHER_DELAYS	KPI_RESULT
1	1	1	1	1	1	1	100%
1	1	1	1	1	1	1	100%
0.87	0.94	1	1	0	1	1	83%
1	1	1	1	1	1	1	100%
1	1	1	1	1	0	1	86%
1	1	1	1	1	1	1	100%
1	0.97	1	1	1	1	1	100%

A) KPI samples during January 2021. The previously mentioned formula was computed in Microsoft Excel with the average formula (red arrow).

KPI Report – Delivery Data



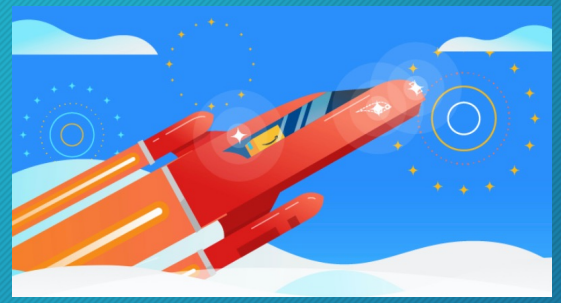
A) KPI results during Jan. 2021 – Mar. 2021; Only 1 block scored less than 85%.



B) Deliveries during Jan. 2021 – Mar. 2021; Most orders are delivered on-time.

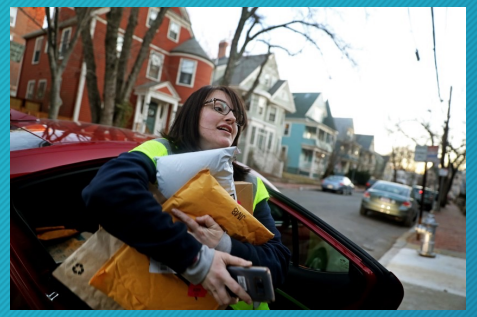
KPI - Conclusion

- The author's Key Performance Indicator (KPI) measures the success percentage of on-time deliveries during each work block.
- This KPI takes unpredictable factors into account too, because uncertainty may affect the driver's or contractor's performance.
- The *Delivery Performance Data* chart from slide #28 reflects that most blocks were completed with a 100% success rate.
- The *Package Delivery Data* chart from slide #28 reflects that most orders were delivered on-time.



Fuel Issue for Delivery Partners (Part I)

- As mentioned on Slide #3, contractors do not know the route beforehand when selecting a block or shift, and typically would spend more gas money than they should in order to avoid stopping at a gas station while working. Every minute counts! therefore making a stop that could been prevented may impact the driver's performance.
- Three gallons of gasoline (approximately) is enough to complete a block in a 4-cylinder vehicle. Gas prices per gallon fluctuate, therefore, fuel expenses will change too.



A) Amazon Flex Contractor.

B) Ordinary vehicle's gas gauge.

Fuel Issue for Delivery Partners (Part II)

IDEAL FUEL REPORT

In [16]:
```
#df_3.head(n=50)
```

In [17]:
```
# Columns '1' to '5' on the following report come from the clean data file (CDF)
# Column titled as 'IDEAL_FUEL_TOTAL' is generated by the value from 'PRICE_PER_GAL' MULTIPLIED BY 3;
# Why 3? Because ideally a 4-cylinder vehicle needs only 3 GALLONS of fuel to complete a block...

# 'IDEAL-ACTUAL_COST_DIFF' is the difference between the actual gas expense during one block MINUS 'Ideal' amount

df_3['IDEAL_FUEL_TOTAL'] = pd.Series([round((val * 3), 2) for val in df_3['PRICE_PER_GAL']], index = df_3.index)
df_3['IDEAL-ACTUAL_COST_DIFF'] = df_3['FUEL_TOTAL'] - df_3['IDEAL_FUEL_TOTAL']
```

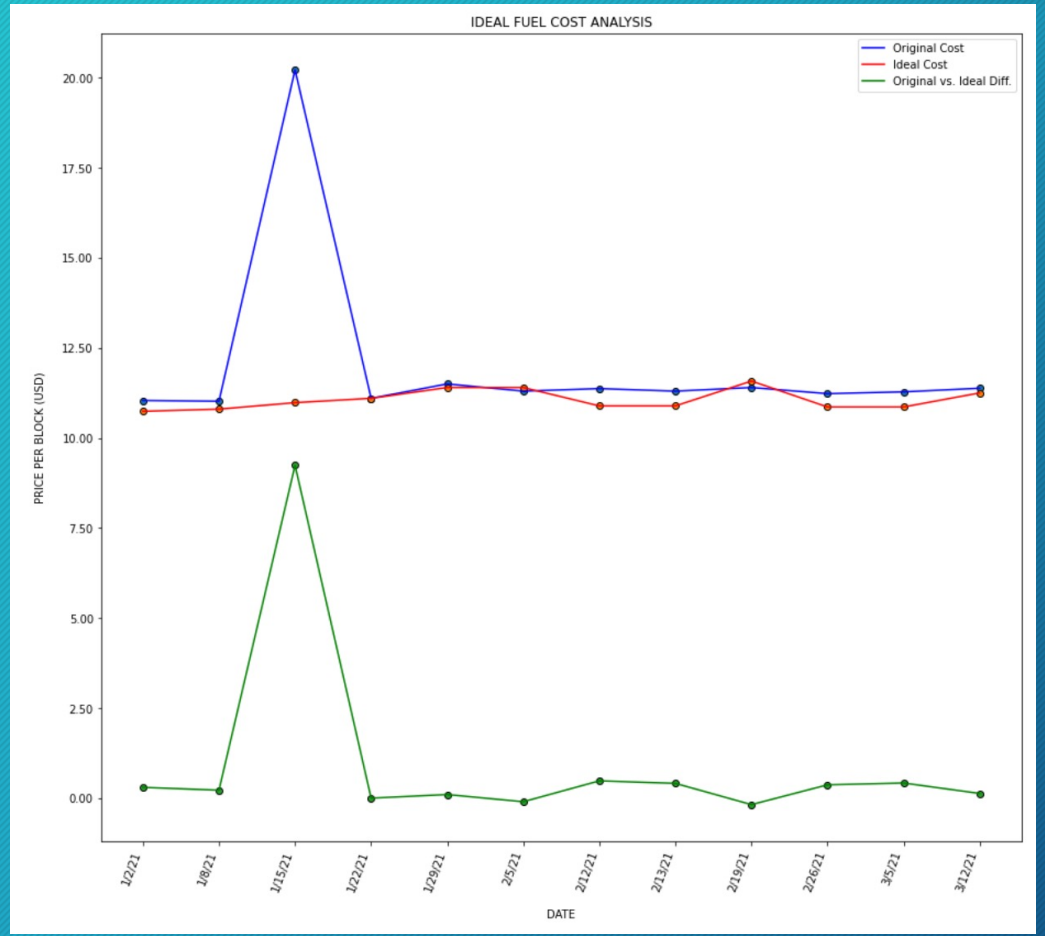
In [18]:
```
df_3.head(n=50)
```

Out[18]:

	DATE	ROUTE_MILEAGE	FUEL_TOTAL	FUEL_GAL	PRICE_PER_GAL	IDEAL_FUEL_TOTAL	IDEAL-ACTUAL_COST_DIFF
0	1/2/21	21	11.04	3.085	3.579	10.74	0.30
1	1/8/21	26	11.02	3.062	3.599	10.80	0.22
2	1/15/21	32	20.22	5.525	3.659	10.98	9.24
3	1/22/21	17	11.10	3.002	3.699	11.10	0.00
4	1/29/21	37	11.50	3.027	3.799	11.40	0.10
5	2/5/21	23	11.30	2.974	3.799	11.40	-0.10
6	2/12/21	49	11.37	3.133	3.629	10.89	0.48
7	2/13/21	31	11.30	3.114	3.629	10.89	0.41
8	2/19/21	31	11.40	2.954	3.859	11.58	-0.18
9	2/26/21	34	11.23	3.103	3.619	10.86	0.37
10	3/5/21	48	11.28	3.117	3.619	10.86	0.42
11	3/12/21	20	11.38	3.035	3.749	11.25	0.13

A) The Ideal Fuel Report contains clean data that originates from the Raw Data File. Please read green letters with '#' on pic.

Fuel Issue for Delivery Partners (Part III)



A) This plot uses data that corresponds to the ideal fuel report. See slide #31.

Fuel Issue Conclusion

- Fuel expenses are unavoidable as an Amazon Delivery Partner. Knowing how to appropriately relocate your resources could allow contractors to save money on gas more efficiently while working.
- The goal of this 'Ideal Fuel Expense Analysis' is to understand / visualize how much money gets spent on gasoline vs. a theoretical price (or value) based on the fluctuant cost of gas per gallon.
- The ideal or theoretical value of money spent on gas is the outcome of calculating the cost of gas (per gallon) during the date of the block multiplied by 3 gallons of gas. The parameter is 3 gallons of gas because that is sufficient for any 4-cylinder car to complete routes.
- Theoretical expenses tend to be lower than the actual expense by monetary differences that are less than $1 USD (outliers are excluded).

Data Science (DS) Techniques

- There are are many statistical and data science (DS) methods that exist and currently get applied in business analytics and different sectors. In this projects we use these techniques to better understand the dataset's insights.
- The following techniques are applied in this project:

A. Time Series Analysis / Forecasting

B. Simple Linear Regression

C. Logistic Regression

D. Statistical Hypothesis Testing

E. Cluster Analysis

A) Book Cover Image: Data Science From Scratch with Python by Richard Wilson.

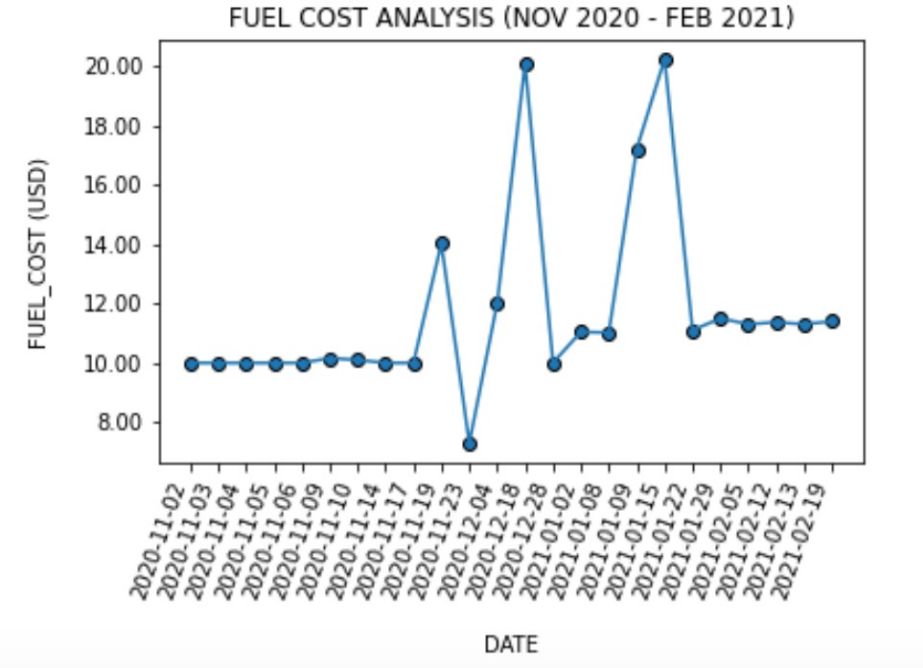
DS – Time Series Analysis (Part I)

- Time series analysis is the statistical technique that deals with time-related data or trend analysis from a set of observations on the value that a variable (fuel cost) takes at different times (*definition by StatisticsSolutions*).
- This method is applied to the project's fuel cost analysis. The objective of this technique is to predict if contractors will have to spend more money of gas expenses before each shift or block.
- Note: Historical data was included in addition to the rows from the Amazon Flex_RDF (Raw Data File). Historical data is defined by the author as the information related to Amazon-Flex's project that was compiled before this project began. Fuel expenses' historical data is NOT included in the Amazon-Flex_RDF document.

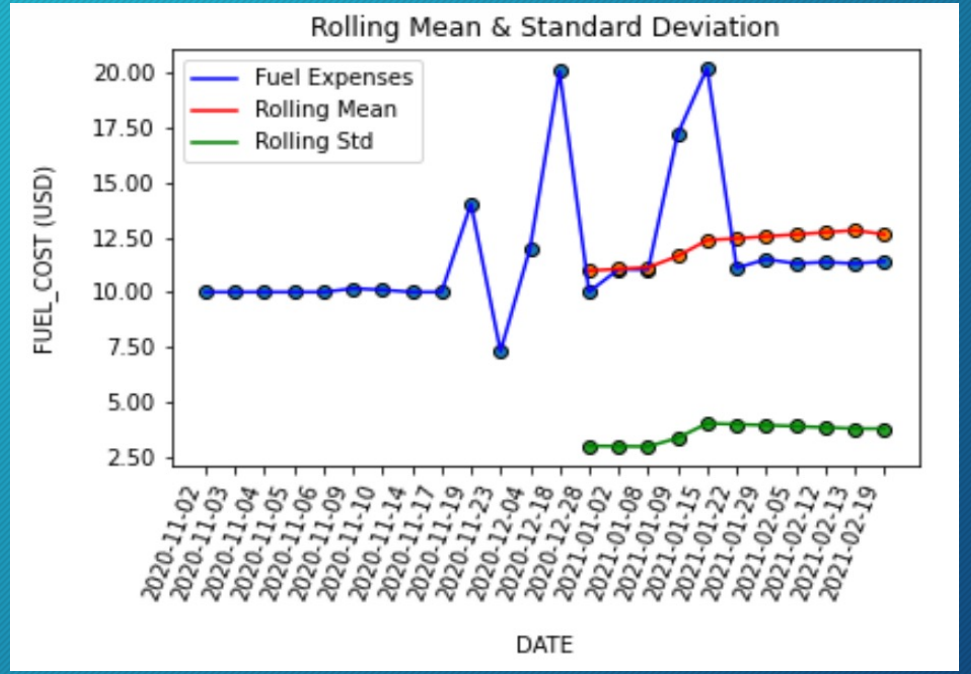
DS – Time Series Analysis (Part II)

- Relevant definitions and concepts:
  A. Scatter plot: Mathematical diagram that uses cartesian coordinates to display values for two variables in the horizontal (x) and vertical (y) axis. Values are typically shown as colored-dots (*definition by Wikipedia*).
  B. Rolling mean: Also known as a moving average, its the unweighted mean of the last *n* values (*definition by Portent*). In this time series model the "last n values" are the 2021 data points because historical data from 2020 is included. The goal of graphing the rolling mean or average is to get rid of any outlier(s) and observe a pattern.
  C. Rolling Standard Deviation (STD): An STD Number is used to tell how measurements for a group are spread out from the average. A low standard deviation means that most of the numbers are close to the average (or mean), while a high standard deviation means that the numbers are more spread out (*definition by Wikipedia*). Therefore, a rolling STD is based on the rolling mean.
  D. Outlier: Point or data value that is significantly or abnormally different (much higher or lower) within a dataset.

DS – Time Series Analysis (Part III)



A) Image of connected scatter plot representing fuel cost data ranging from Nov. 2020 to Feb. 2021 with six outliers. If we carefully look at the graph, we'll see that the fuel expenses' trend tends to increase. This graph is not predicting anything yet.

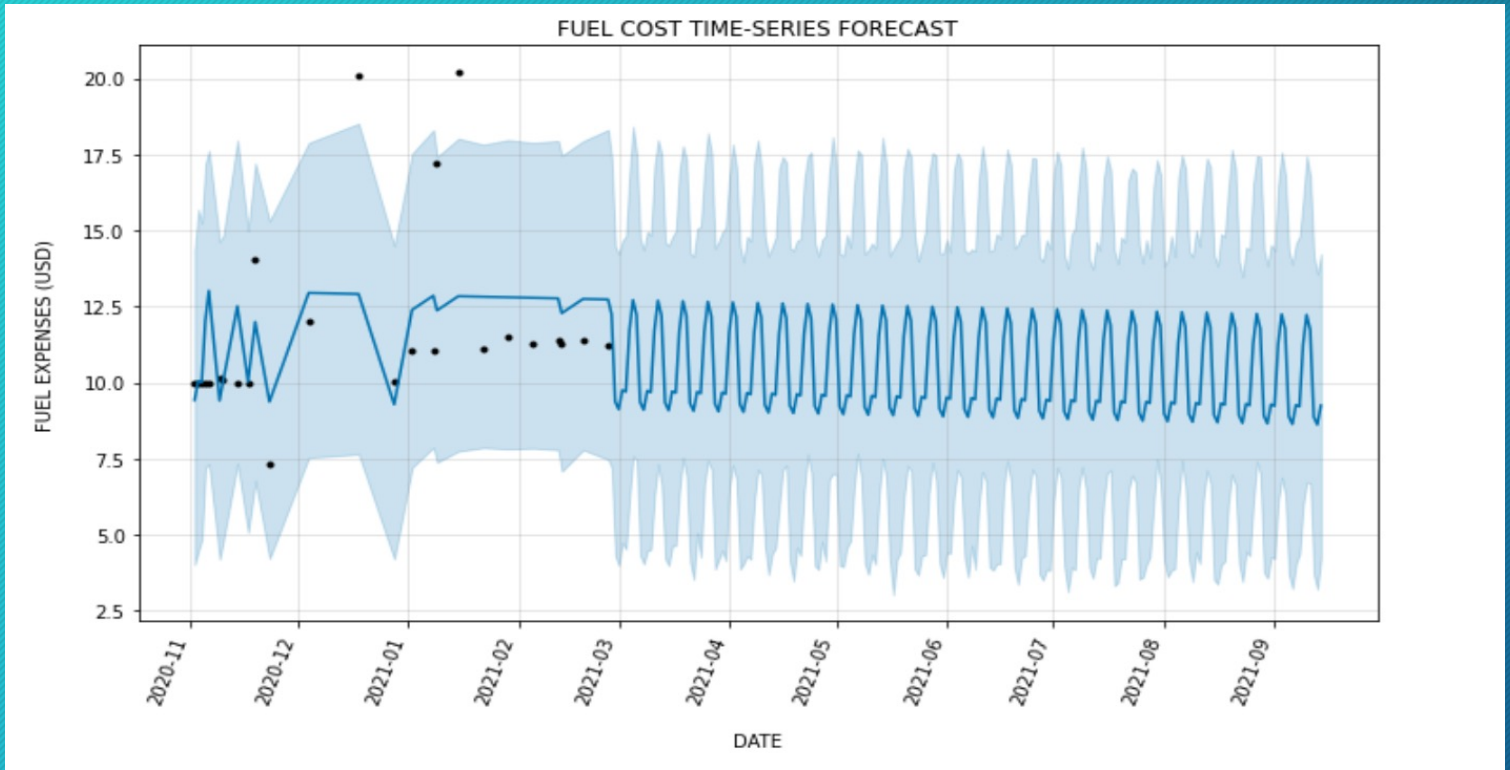


B) Image of connected scatter plots (data ranging from Nov. 2020 to Feb. 2021). Rolling mean (average of 2021 data) and rolling STD show that the fuel expenses' trend tends to increase. No outliers are reflected in the rolling data. Fuel cost dataset is NOT stationary or constant.

DS – Time Series Forecasting (Part IV)

- A time series forecasting technique is used to predict events through a sequence of time by analyzing the trends of the past. This machine learning model assumes that future trends will hold similar historical patterns (*definition by TechTarget*).
- Why can the time series forecast be applied to the author's fuel expenses analysis?
  - We only need data points based on dates and amount spent on 3 gallons of gas pumped into the vehicle before the shift starts.
  - Gas prices fluctuate - in every US state the price per gallon tends to increase. The author injects approximately 3 gallons of fuel required before each shift therefore, it is not possible to have constant gas expenses.
  - A machine learning data model can be "trained" to predict approximate values as more data gets compiled.

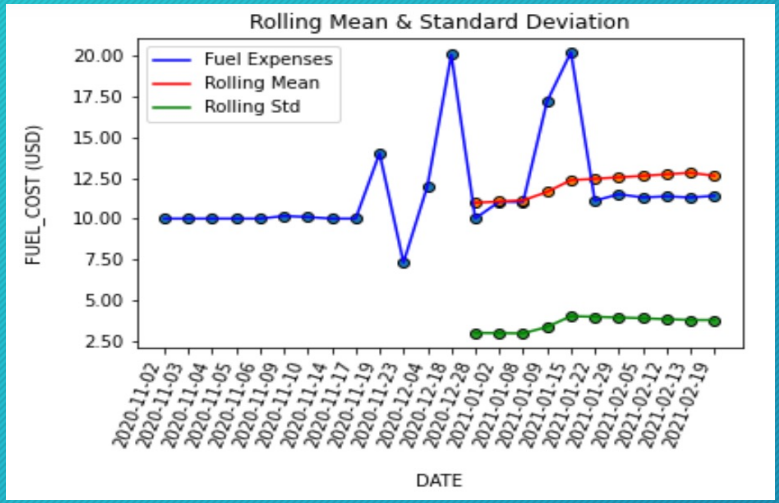
DS – Time Series Forecasting (Part V)



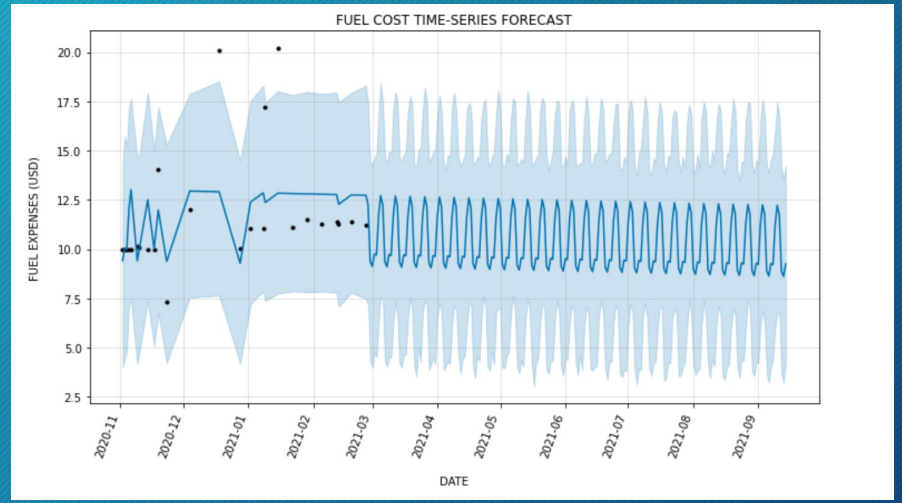
A) Image of fuel expenses time series forecasting (predictive) data model. Data points range from Nov. 2020 – Feb. 2021. The dark blue line represents the prediction values. Since the dataset is not large enough yet to efficiently "train" the model, the graph presents an oscillatory behavior from Mar. 2021 – Sept. 2021. Due to outliers, the machine learning model 'thinks' that fuel costs will always occasionally increase and decrease drastically at any point.

DS – Time Series Conclusion

- The time series analysis for fuel costs reflects a minor increasing trend in gas expenses. There are a few outliers in the data, and it is possible that some might happen again. An increasing trend is expected as gas prices tend to increment in San Diego County, CA.
- The time series forecasting for fuel expenses is a machine learning model that requires a large amount of data points. Outliers in historical data causes the model to predict values to reflect an oscillatory behavior, but the graph's appearance changes as the dataset gets larger.



A) The time series analysis does NOT predict the future.



B) The dark blue line in the time series forecast predicts values.

DS – Simple Linear Regression (Part I)

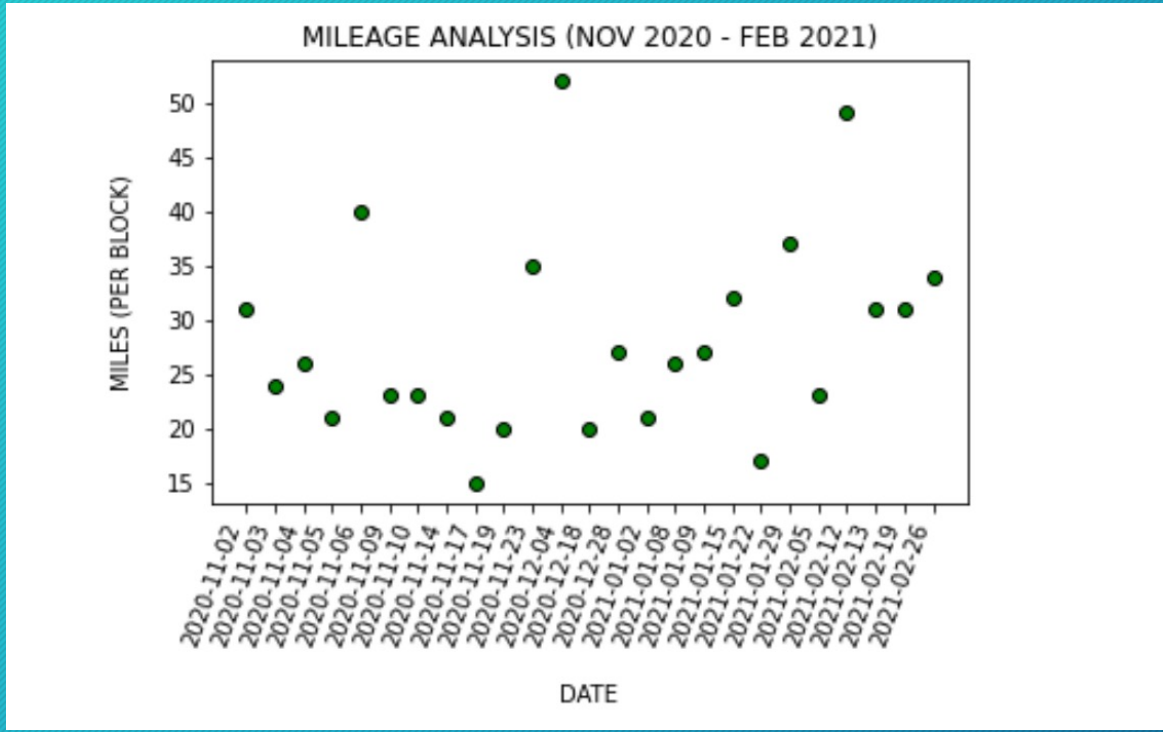
- Regression models are used to estimate the relationship between variables by fitting a line through the observed data. A simple linear regression is used to estimate the relationship between 2 variables by fitting a straight line (*definition by Scribbr*).
- This method is applied to the project's mileage analysis. Amazon contractors cannot predict how much mileage they'll add to their vehicle once they conclude a block, before they arrive to the warehouse.
- This project includes 2 simple linear regression models, the first includes time data and the second models a relationship between mileage and the total package number (per block). Regression is a predictive modelling technique (*definition by Edureka*).

DS – Simple Linear Regression (Part II)

- Historical data was included on the 1st model in addition to the rows from the Amazon-Flex_RDF (Raw Data File). Historical data is defined by the author as the information related to Amazon-Flex's project that was compiled before this project began. Mileage historical data is NOT included in the raw data file.
- The first simple linear regression model (time vs. mileage) uses data ranging from Nov. 2020 – Feb. 2021 because the current dataset is not large enough yet to efficiently "train" the model. The graph's appearance will possibly change as more data gets added.
- Note: Every linear regression modelling technique is a machine learning method. In this case, we are "teaching" the model to predict whether future route mileage will increase or not (*Concept by Movidev*).

DS – Simple Linear Regression (Part III)

- The first simple linear regression technique models time (specific dates) vs. mileage per block.

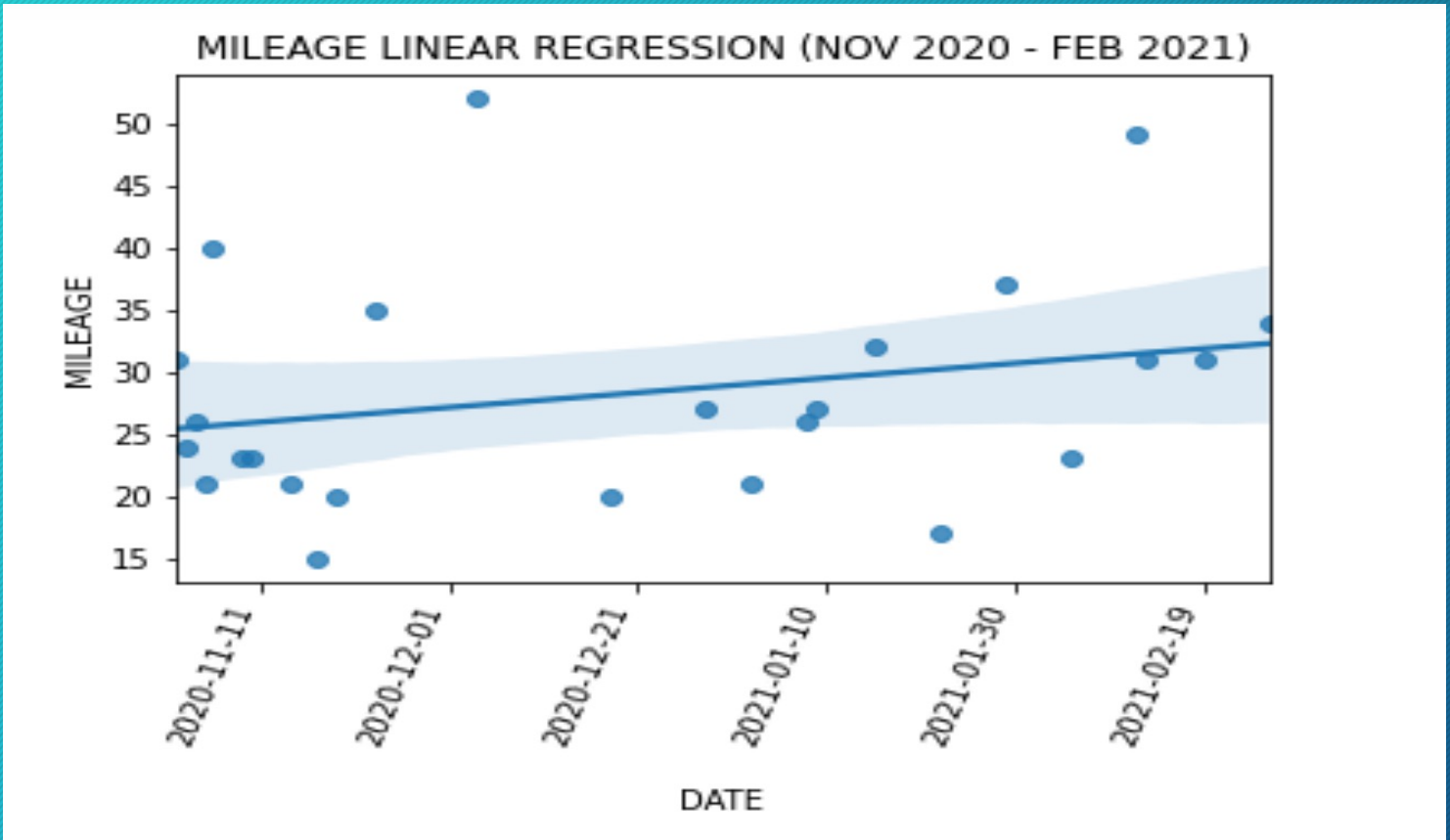


A) This is NOT the linear regression model. This scatter plot includes historical data.

	MILEAGE
count	25.000000
mean	28.240000
std	9.207244
min	15.000000
25%	21.000000
50%	26.000000
75%	32.000000
max	52.000000

B) Mileage analysis' basic statistics.

DS – Simple Linear Regression (Part IV)



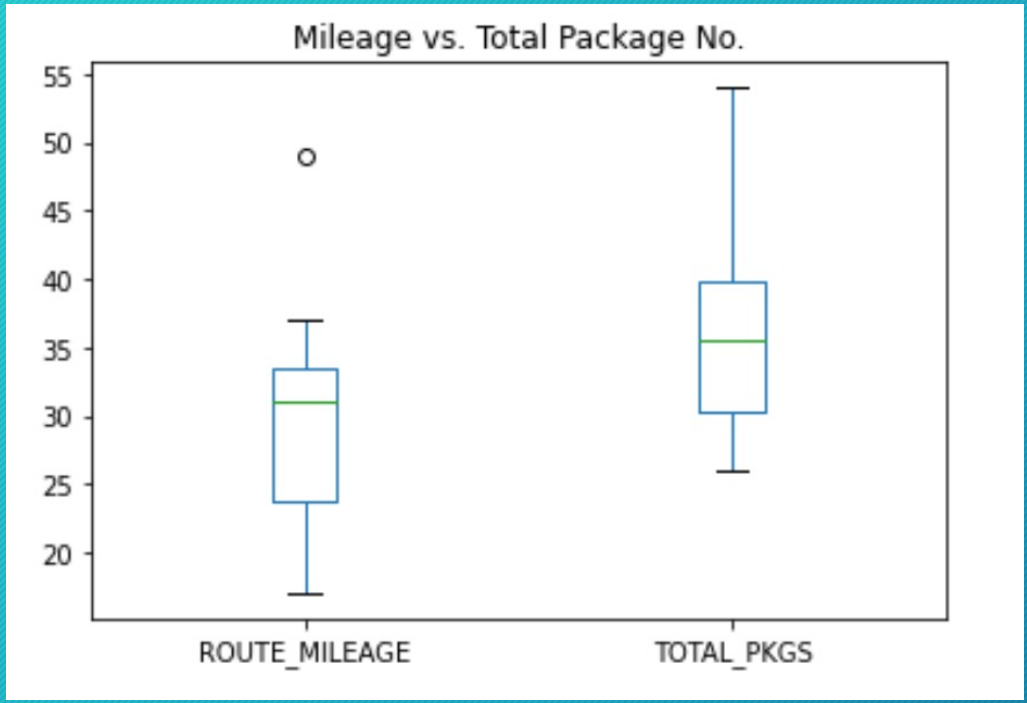
A) First simple linear regression model (time vs. mileage). The dark blue line represents the prediction values. In this case the model assumes that route mileage will increase, however the dataset needs to be larger to be more efficient and behavior (prediction) might change as more data gets compiled.

DS – Simple Linear Regression (Part V)

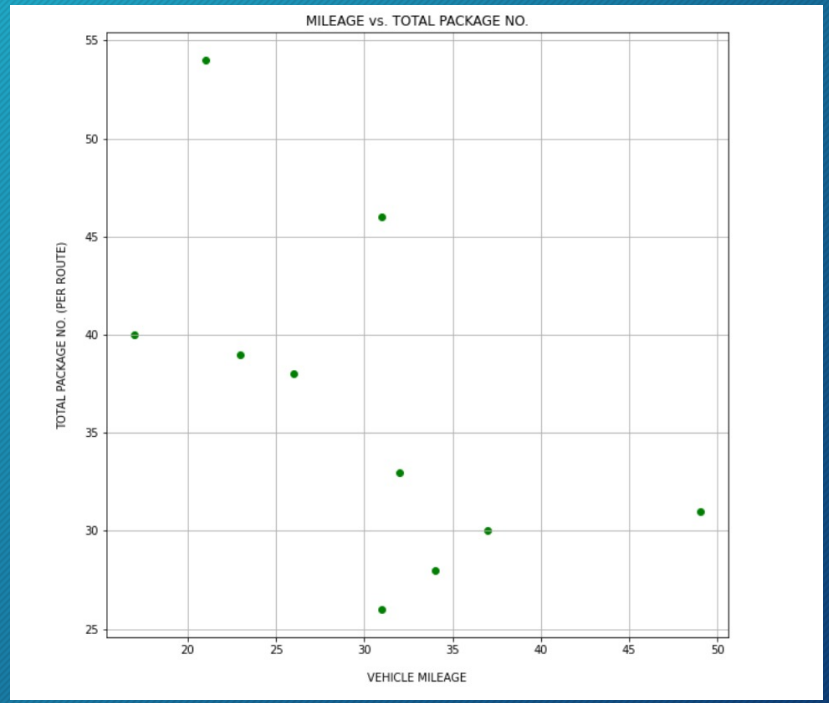
- The 2nd simple linear regression technique models route mileage vs. total package number (per route). Since the current dataset is not large enough yet to efficiently "train" the model, the graph's appearance will possibly change as more data gets compiled.
- Historical data was NOT included on the 2nd model. Total package number per block data wasn't manually compiled prior to this project. Historical data is defined by the author as the information related to Amazon-Flex's project that was compiled before this project began. Only 2021 data is included in the 2nd model.
- Note: Every linear regression modelling technique is a machine learning method. In this case we are "teaching" the model to predict if increasing mileage gets reflected on the given (random) package number in each block (*Concept by Movidev*).

DS – Simple Linear Regression (Part VI)

- The 2nd simple linear regression technique models route mileage vs. total package number per block from January 2021 – February 2021. The following plots are NOT the linear regression models.

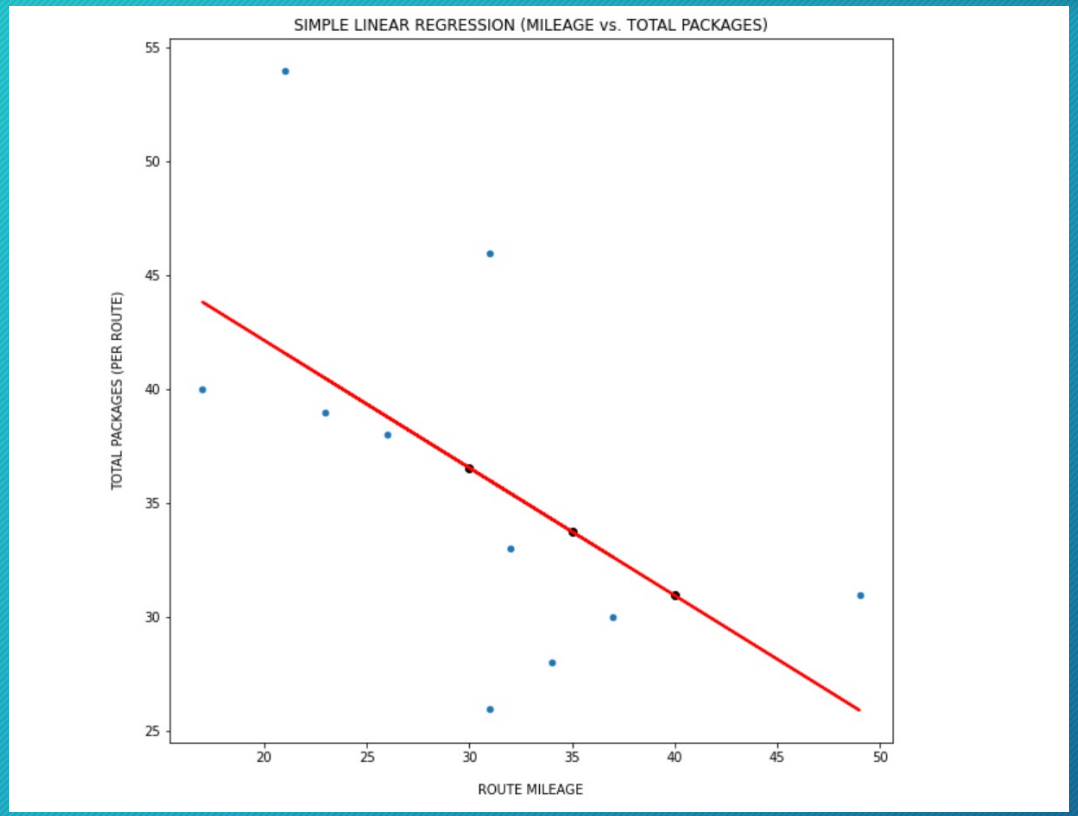


A) This is a box plot comparing route mileage and total package number. Only mileage has an outlier.



B) This scatter plot shows the relationship between car mileage and the given package number in each shift or block.

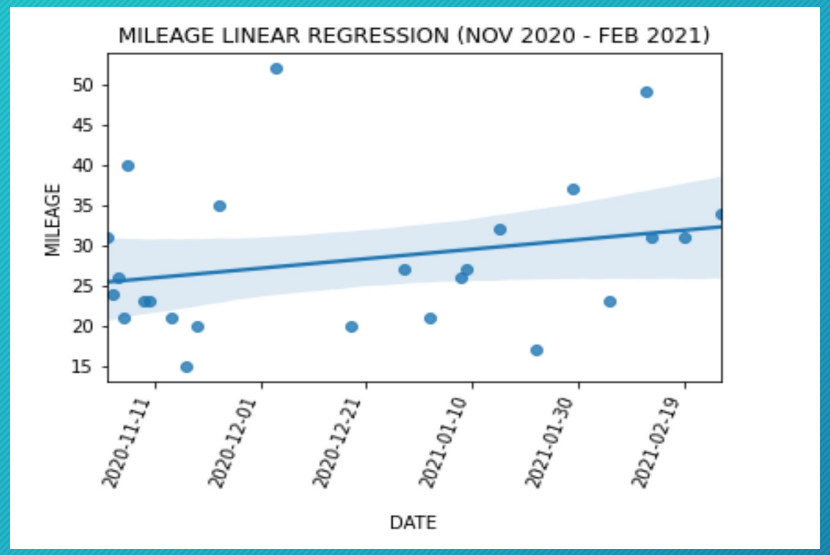
DS – Simple Linear Regression (Part VII)



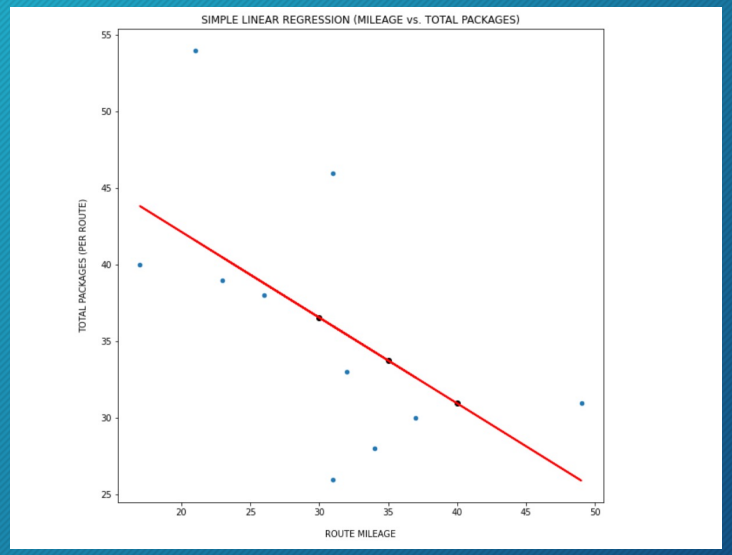
A) 2nd simple linear regression model. The red line represents the prediction values. The black data points intercepting the red line were chosen randomly. The model 'thinks' that as route mileage increases, the package number will decrease. This behavior might change as the dataset gets updated (larger).

DS – Linear Regression Conclusion

- The 1st simple linear regression (dates vs. miles) model predicts mileage will steadily increment in each shift, however the dataset is quite small yet, therefore this behavior might change as more data get compiled.
- The 2nd simple linear regression (miles vs. total packages x block) predicts that the random package number will decrease as the route covers more distance. A larger dataset might prove that blocks with less packages to deliver have longer routes.



A) 1st Linear Regression Model. 2020-2021 dataset.



B) 2nd Regression model. 01/2021 – 02/2021 data.

DS – Logistic Regression (Part I)

- Regression models are used to estimate the relationship between variables by fitting a line to the observed data. Logistic Regression is a type of linear regression (machine learning method) however it is used for classification only (*definition by Ajaytech*). Meaning that you plot one variable (integers or decimals) on the x-axis, and one / zero (true or false) in the y-axis.
- Every plotted value gets the probability of .5 to belong to either category (1 or 0) and these are placed either 'above' the curve or 'below' (it's up to the developer or analyst to determine the meaning of 1 or 0).
- Historical data was NOT included on this model. Historical data is defined by the author as the information related to Amazon-Flex's project that was compiled before this project began. Only 2021 data is included.

DS – Logistic Regression (Part II)

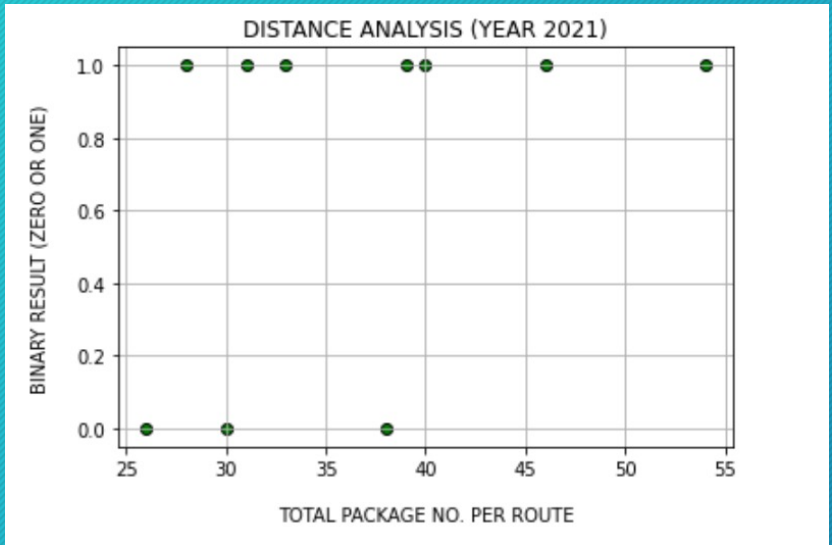
- The application of the logistic regression model was based on a simple question: Do routes or blocks that involve neighborhoods in which the first delivery address is LESS THAN 10 miles from the Amazon warehouse get MORE packages?
- Routes in which the first delivery address was 'relatively' close (< 10 miles) to the warehouse get labelled as '1', but '0' if that's not the case.

	DATE	WHS-NBHD_MIN-APPROX-DIST	TOTAL_PKGS	DIST_LESS_THAN_TEN
0	2021-01-02	4.5	54	1
1	2021-01-08	13.7	38	0
2	2021-01-15	7.6	33	1
3	2021-01-22	8.7	40	1
4	2021-01-29	13.0	30	0
5	2021-02-05	8.5	39	1
6	2021-02-12	9.3	31	1
7	2021-02-13	6.0	46	1
8	2021-02-19	13.0	26	0
9	2021-02-26	9.3	28	1

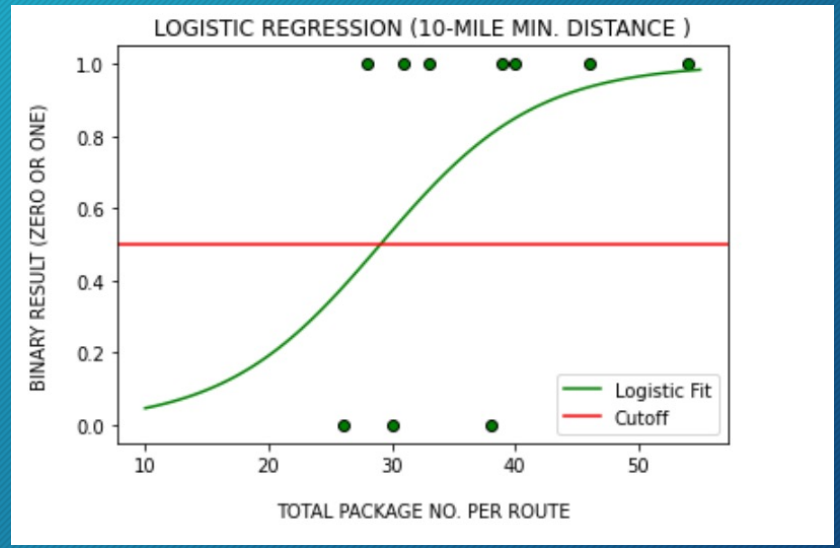
A) Data from DISTANCES_ANALYSIS.csv file. Dataset only contains binary or categorical data on last column.

DS – Logistic Regression (Part III)

- The model's data gets 'trained', and its accuracy gets tested before being plotted. Accuracy will increase as more data gets compiled. The goal of this model is to predict if the total package no. is larger when the warehouse-neighborhood (WHS-NBHD) min. distance is less than 10 miles.



A) This analysis shows that most routes during Jan. and Feb. 2021 didn't surpass the 10-mile condition for min. distance.



B) This regression model is based on Jan. and Feb. 2021 data. The green line predicts that larger package no. converge to '1'.

DS – Logistic Regression Conclusion

- The logistic regression model supports the idea that Amazon's algorithms assign a larger random number of packages to contractors when block routes take place in areas that are relatively close to the warehouse (point of origin). Therefore, it is statistically more likely for Amazon partners to get less packages for routes that are distant from the warehouse.
- The original question that allowed the application of this machine learning technique was: Do routes or blocks that involve neighborhoods in which the first delivery address is LESS THAN 10 miles from the Amazon warehouse get MORE packages?
- The appearance of this model might change as the dataset gets larger. Anyhow, as the total package number increases, the logistic fit or green line (*see previous slide*) converges to '1'. Recall that '1' represents routes that are categorized as those that have a 'min. distance that is less than 10 miles.'

DS – Statistical Hypothesis Testing (Part I)

- Hypothesis testing is used to test the results of surveys, experiments or numerical data by figuring out the odds that your results have happened by chance. However, a null hypothesis needs to be set (by author) to be tested for possible rejection under the assumption that it is true (*definition by Statistics How To*).
- This testing method checks the following:
  A. The probability that most blocks (routes) will cover 2 different neighborhoods on average.
  B. The probability of delivering orders in Downtown Area during any block (hence route).

H_0 = The number of neighborhoods in one random delivery-route is 2.

H_a = The number of neighborhoods in one random delivery-route is NOT 2.

μ = 2 (hypothesis mean in this test).

A) For the first test, we set both our null and alternative hypothesis. We use the 'mew' symbol to set our assumption, which is an average of 2.

H_0 = The probability of working in downtown area during any route is greater than 50%.

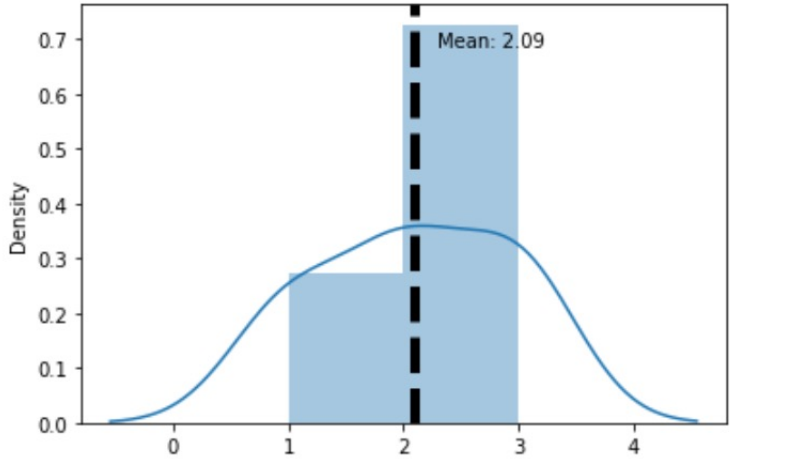
H_a = The probability of working in downtown area during any route is less than or equal to 50%.

P = .5 (P refers to probability in this test).

B) For the second test, we set both our null and alternative hypothesis. Unlike the 1st test, our assumption is a probability that may be greater than .50.

DS – Statistical Hypothesis Testing (Part II)

- Most routes involve a set of addresses that may or may not involve the same zip code. In rare occasions, the block's set of addresses correspond to one single area or neighborhood. In most cases, it's 2 areas per block. This test is categorized as a two-tailed / one sample test. Why '2-tailed'? We initially assume that the average is 2, but there's also a probability of having single-neighborhood blocks or multiple-zone routes. This test only contains 2021 data.

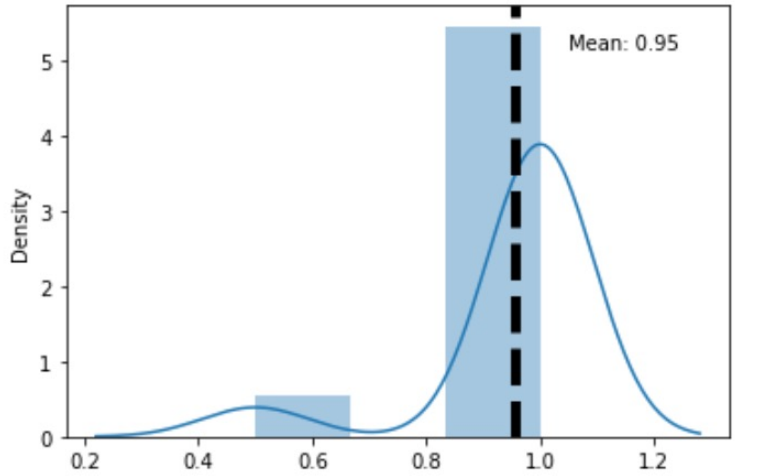


	TOTAL_NBHD_COUNT	NON-DWTN_COUNT	NON-DWTN_EMPIRICAL_PROBABILITY
count	11.000000	11.000000	11.000000
mean	2.090909	2.000000	0.954545
std	0.831209	0.894427	0.150756
min	1.000000	1.000000	0.500000
25%	1.500000	1.000000	1.000000
50%	2.000000	2.000000	1.000000
75%	3.000000	3.000000	1.000000
max	3.000000	3.000000	1.000000

A) Data points on distribution range from Jan. 1st – Mar. 5th. Our Null Hypothesis was correct, the mean or average is 2.09 (2.0 approximately) areas per route.

DS – Statistical Hypothesis Testing (Part III)

- By observation, most blocks rarely include addresses that correspond to downtown (San Diego, CA). This test is categorized as a one-tailed / one sample test. Why '1-tailed'? We initially assume our null hypothesis is the probability of working on downtown being greater than 50%, but in theory we assume in this case that there's NO probability of having less than 50%. Our null hypothesis is expected to get rejected because 95% of blocks do NOT involve downtown. This test only contains 2021 data.



	TOTAL_NBHD_COUNT	NON-DWTN_COUNT	NON-DWTN_EMPIRICAL_PROBABILITY
count	11.000000	11.000000	11.000000
mean	2.090909	2.000000	0.954545
std	0.831209	0.894427	0.150756
min	1.000000	1.000000	0.500000
25%	1.500000	1.000000	1.000000
50%	2.000000	2.000000	1.000000
75%	3.000000	3.000000	1.000000
max	3.000000	3.000000	1.000000

A) Data points on distribution range from Jan. 1st – Mar. 5th. We observe a skewed graph because the odds of getting a block where downtown is involved are very low.

DS – Hypothesis Testing Conclusion

- The first test (2-tailed for one sample) successfully accepts our null hypothesis stating that there's 2 neighborhoods on average on any block (route).
- Unlike the previous test, we reject our null hypothesis stating that there's a chance greater than 50% of making deliveries in downtown. This confirms the fact that working in this area is statistically not likely; meaning that we accept the alternative hypothesis.
- Both tests were based on 2021 data only.

$H_0 : \mu = 2$ ✓

$H_a : \mu \neq 2$

A) We accept our null hypothesis from the 1st test.

$H_0 : P > .50$

$H_a : P \leqslant .50$ ✓

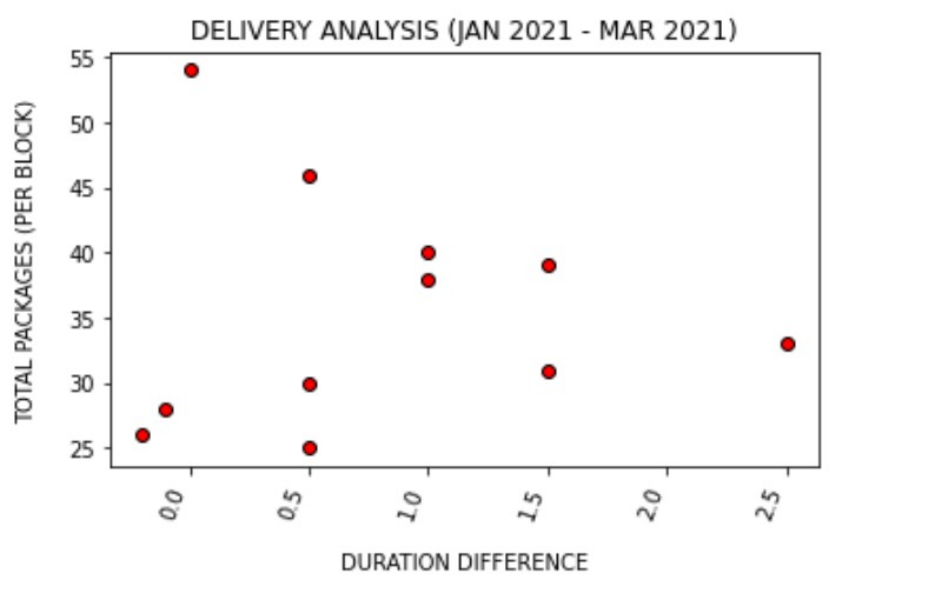
B) We reject our null hypothesis on the 2nd test and accept the alternative hypothesis.

DS – Clustering Analysis (Part I)

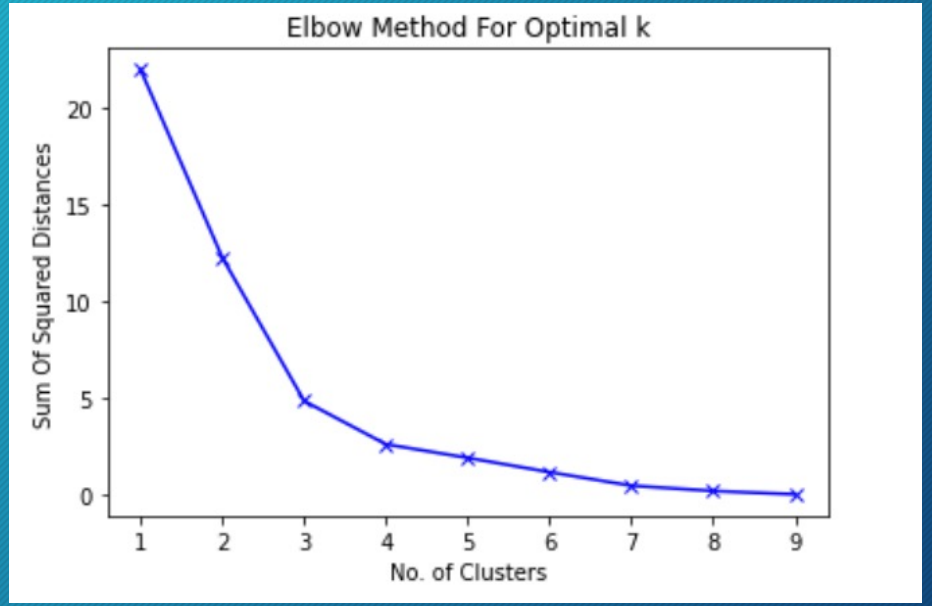
- Clustering means grouping data points (or objects) based on the information found in the data. The objective is to cluster objects into categories based on patterns. This is an 'unsupervised' machine learning technique, which means that the system decides how to categorize data based on complex mathematics (*definition by Edureka*).
- The cluster analysis method categorizes the performance of each block based on the shift's duration difference (actual shift time minus 'ideal' duration – which is 3 hours) vs. total number of given packages per block.
- Historical data was NOT included on this model. Historical data is defined by the author as the information related to Amazon-Flex's project that was compiled before this project began. Only 2021 data is included.

DS – Clustering Analysis (Part II)

- This method requires the Python script to generate two centroids. These are random data points (not from original dataset) that their purpose is to serve as points of reference to be able to classify data.

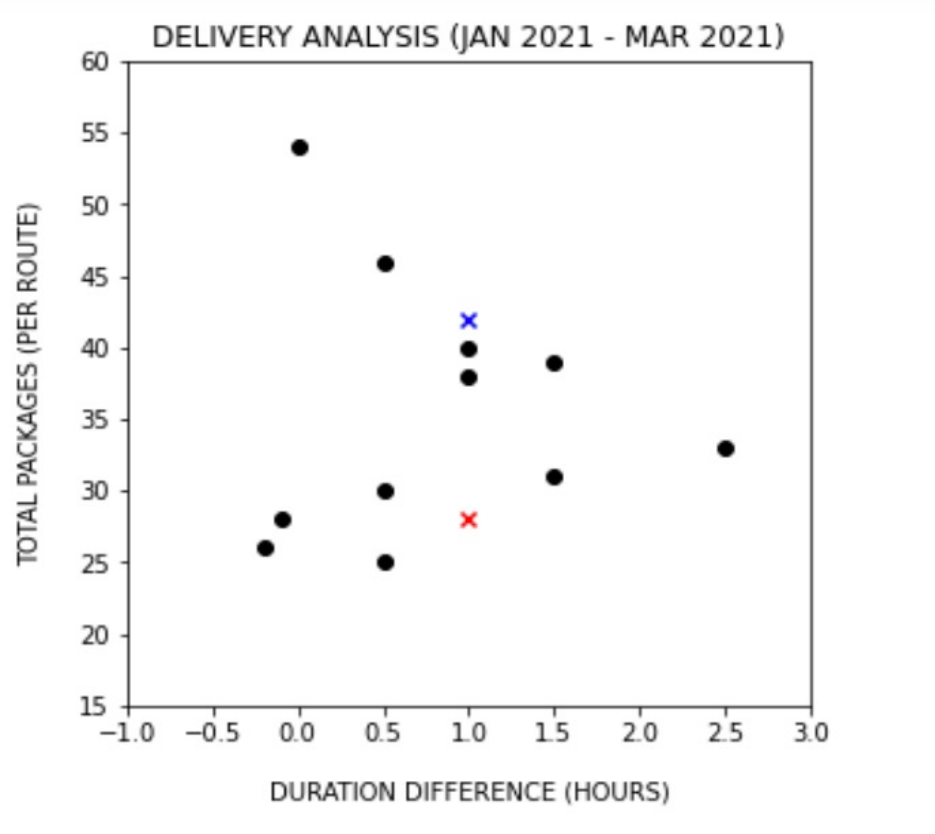


A) Negative time values represent occasions in which deliveries were completed before the assigned schedule was over. Only 2 blocks have been completed before 3 hours in the 1st quarter.

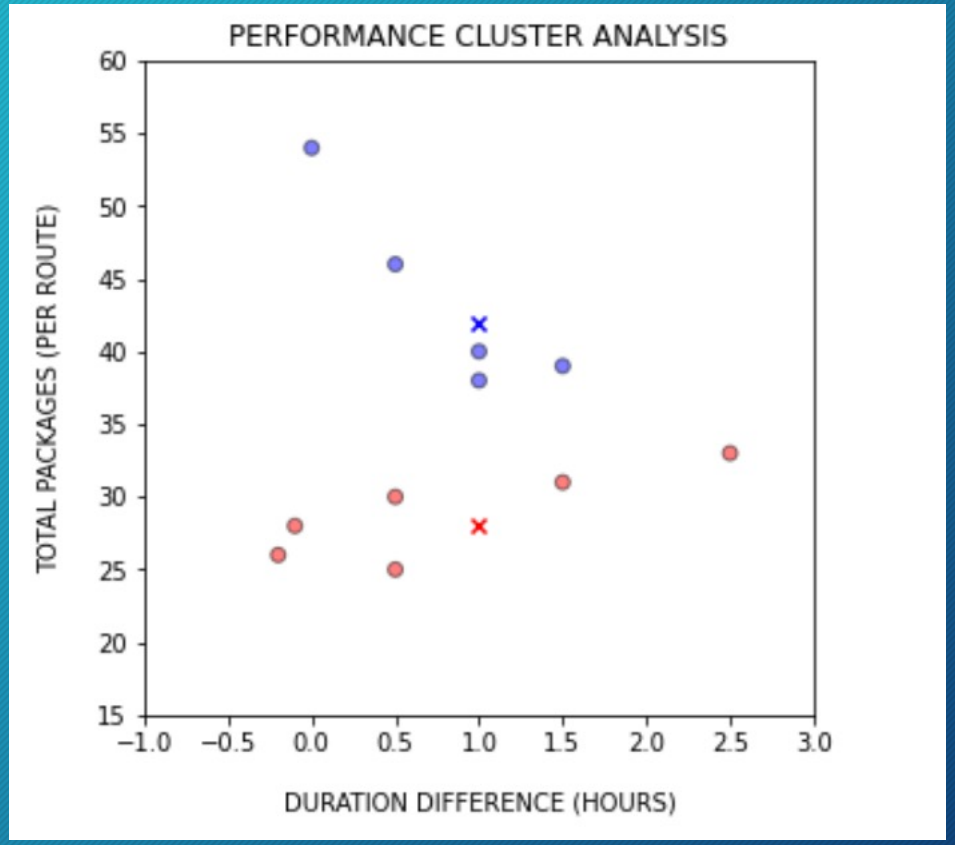


B) The 'Elbow' method is applied to determine the no. of clusters. The 'k' point that resembles an 'elbow joint' is occasionally the ideal no. of centroids that we need to create, but we may pick a smaller value.

DS – Clustering Analysis (Part III)



A) The red and blue cross represent the centroids that were randomly generated and hence assigned in the plot. Black data points represent current data. Since we assigned 2 centroids, we'll get two clusters.



B) Blue data points represent blocks completed with outstanding performance (more deliveries were made relative to red points with similar durations). Red stands for average performance.

DS – Clustering Analysis Conclusion

- The analysis successfully classified 2 different categories. This machine learning technique is considered an unsupervised method. We may control the number of centroids that we need to classify clusters, but the user or developer can't determine how every single data point will be labelled, since the system (or Python's script modules) does.
- How is performance measured in the cluster analysis? The greater the number of delivered packages per block relative to the time difference, the better! For example, the case in which blue or red data points (blocks) had a time difference of 0.5 hours; blue data points mean MORE orders were delivered despite finishing 30 minutes after the schedule was over, unlike those blocks (red points) where LESS products were delivered and still finished half hour later (30 minutes after the block's official schedule).
- NOTE: Completing blocks after the assigned schedule (3 hours) don't necessarily result on late-deliveries or poor performance.

DS Techniques – Conclusion

- There are currently many data science and/or machine learning (supervised or unsupervised) techniques that exist in the field. The author chose 5 relevant methods that matched with questions based on observations, and the raw data's structure.
- Not all aspects (columns) from the original raw data fit every model presented throughout this presentation, therefore careful planning allowed the author to successfully apply legit data science methods.
- The following techniques are applied in the same order as stated below:

A. Time Series Analysis
B. Time Series Forecast
C. Simple Linear Regression
D. Logistics Regression
E. Statistical Hypothesis Testing
F. Clustering Analysis

How Does This Project Help Amazon?

- Insights based on data science techniques and analytics provided by an independent contractor could allow Amazon to gain a different perspective about its Flex Program and find other ways of improvement. For instance, these results might help Amazon's developers to improve its delivery-related algorithms to provide more efficient or practical routes. Another example could be to have a better understanding of the variables that can affect a delivery partner's performance.

A) Amazon package cart.

B) Amazon warehouse's interior.

REFERENCES (Audiovisual Sources)

- "30 Days of Python - Day 17 - Data Science Pipeline with Jupyter, Pandas & FastAPI - Python TUTORIAL." *YouTube*, uploaded by CodingEntrepreneurs, 15 April 2020, www.youtube.com/watch?v=CApCQKuWqBM&list=LL&index=16
- Bhatt, Bhavesh. "T-Test for Comparing Two Group Means in Python." *YouTube*, uploaded by Bhavesh Bhatt, 12 September 2019, www.youtube.com/watch?v=8aaIdXENNJI&list=LL&index=9
- "K Means Clustering Algorithm | K Means Example in Python | Machine Learning Algorithms | Edureka." *YouTube*, uploaded by Edureka, 28 May 2018, www.youtube.com/watch?v=1XqG0kaJVHY&list=LL&index=6&t=1423s
- "Linear Regression Algorithm | Linear Regression in Python | Machine Learning Algorithm | Edureka." *YouTube*, uploaded by Edureka, 26 June 2018, www.youtube.com/watch?v=E5RjzSK0fvY
- "Logistic Regression in Python | Logistic Regression Example | Machine Learning Algorithms | Edureka." *YouTube*, uploaded by Edureka, 29 May 2018, www.youtube.com/watch?v=VCJdg7YBbAQ
- Nguyen Van, Chuc. "How to build a Simple Linear Regression model with Python." *YouTube*, uploaded by Chuc Nguyen Van, 21 September 2018, www.youtube.com/watch?v=fzZ0HO-uz1o&list=LL&index=12
- "Null and Alternate Hypothesis - Statistical Hypothesis Testing - Statistics Course." *YouTube*, uploaded by Math and Science, 20 August 2014, www.youtube.com/watch?v=_QIxt0HmuOo&list=LL&index=10
- Renotte, Nicholas. "Time Series Forecasting with Facebook Prophet and Python in 20 Minutes." *YouTube*, uploaded by Nicholas Renotte, 16 December 2020, www.youtube.com/watch?v=KvLG1uTC-KU&list=LL&index=13&t=31s
- "Time Series Analysis in Python | Time Series Forecasting | Data Science with Python | Edureka." *YouTube*, uploaded by Edureka, 31 May 2018, www.youtube.com/watch?v=e8Yw4alG16Q&t=895s
- Veda, Ashok. "Hypothesis Testing - Statistics for Data Science Part 1." *YouTube*, uploaded by DataMites, 4 February 2020, www.youtube.com/watch?v=_EVrNmgmois&list=LL&index=11

REFERENCES (Text Sources)

- "5 Essential Machine Learning Techniques For Business Applications." *Mobidev.biz*, mobidev.biz/blog/5-essential-machine-learning-techniques. Accessed 5 April 2021.
- Bevans, Rebecca. "An introduction to simple linear regression." *Scribbr*, www.scribbr.com/statistics/simple-linear-regression/#:~:text=What%20is%20simple%20linear%20regression,Both%20variables%20should%20be%20quantitative. Accessed 5 April 2021.
- Brownlee, Jason. *Machine Learning Mastery*, machinelearningmastery.com/. Accessed 5 April 2021.
- "Empirical probability." Wikipedia, Wikimedia Foundation, 20 October 2019, en.wikipedia.org/wiki/Empirical_probability. Accessed 5 April 2021.
- Glen, Stephanie. "Hypothesis Testing." *StatisticsHowTo.com: Elementary Statistics for the rest of us!*, www.statisticshowto.com/probability-and-statistics/hypothesis-testing/. Accessed 5 April 2021.
- "Key Performance Indicator." Lexico by Oxford, 25 September 2020, www.lexico.com/definition/key_performance_indicator. Accessed 4 April 2021.
- "Logistic Regression." *AjayTech*, ajaytech.co/python-logistic-regression/. Accessed 5 April 2021.
- Lurie, Ian. "How do I calculate a rolling average?" *Portent*, www.portent.com/blog/analytics/rolling-averages-math-moron.htm. Accessed 4 April 2020.
- "Scatter plot." Wikipedia, Wikimedia Foundation, 29 March 2021, en.wikipedia.org/wiki/Scatter_plot. Accessed 4 April 2021.
- "Standard deviation." Wikipedia, Wikimedia Foundation, 3 April 2021, en.wikipedia.org/wiki/Standard_deviation. Accessed 4 April 2021.
- "T VALUE TABLE." *T TABLE*, www.ttable.org/. Accessed 5 April 2021.
- "Time Series Analysis." *StatisticsSolutions*, www.statisticssolutions.com/time-series-analysis/. Accessed 5 April 2021.
- "time series forecasting." TechTarget: WhatIs.com, 1 February 2018, whatis.techtarget.com/definition/time-series-forecasting. Accessed 5 April 2021.
- "Understanding K-means Clustering with Examples." *Edureka*, www.edureka.co/blog/k-means-clustering/. Accessed 5 April 2021.

GITHUB LINKS

- PERFORMANCE REPORTS
  - https://github.com/Juan-Moctezuma/Amazon-Flex_Data_Project/tree/main/Amazon_Official_Performance_Evidence/
- DATA ENGINEERING
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_ENGINEERING/ETL_Data_Pipeline/Extract-Transform-Load_Pipe.ipynb
  - https://github.com/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_ENGINEERING/API/API_RESULTS/API_Results.pdf
- DATA SCIENCE & ANALYTICS DASHBOARDS
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_SCIENCE/DASHBOARD/KPI_and_Fuel_Metrics.ipynb
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_SCIENCE/DASHBOARD/Data_Science-Dashboard.ipynb
- DATA SCIENCE TECHNIQUES
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_SCIENCE/DS_TECHNIQUES/Time_Series.ipynb
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_SCIENCE/DS_TECHNIQUES/Linear_Regression.ipynb
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_SCIENCE/DS_TECHNIQUES/Logistic_Regression.ipynb
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_SCIENCE/DS_TECHNIQUES/Hypothesis_Testing.ipynb
  - https://nbviewer.jupyter.org/github/Juan-Moctezuma/Amazon-Flex_Data_Project/blob/main/DATA_SCIENCE/DS_TECHNIQUES/Clustering_Analysis.ipynb

Thank You for Your Attention!

"A brand for a company is like a reputation for a person. You earn reputation by trying to do hard things well." – Jeff Bezos