RISING TO THE RIDE-SHARING CHALLENGE:

DATA SCIENCE TO DRIVE TAXI PROFITABILITY

DERRICK CHAM

Case Interview – Considerations and Approach

Business First, Data Science Second

End-to-End Thinking

Proof-of-Concept Mindset

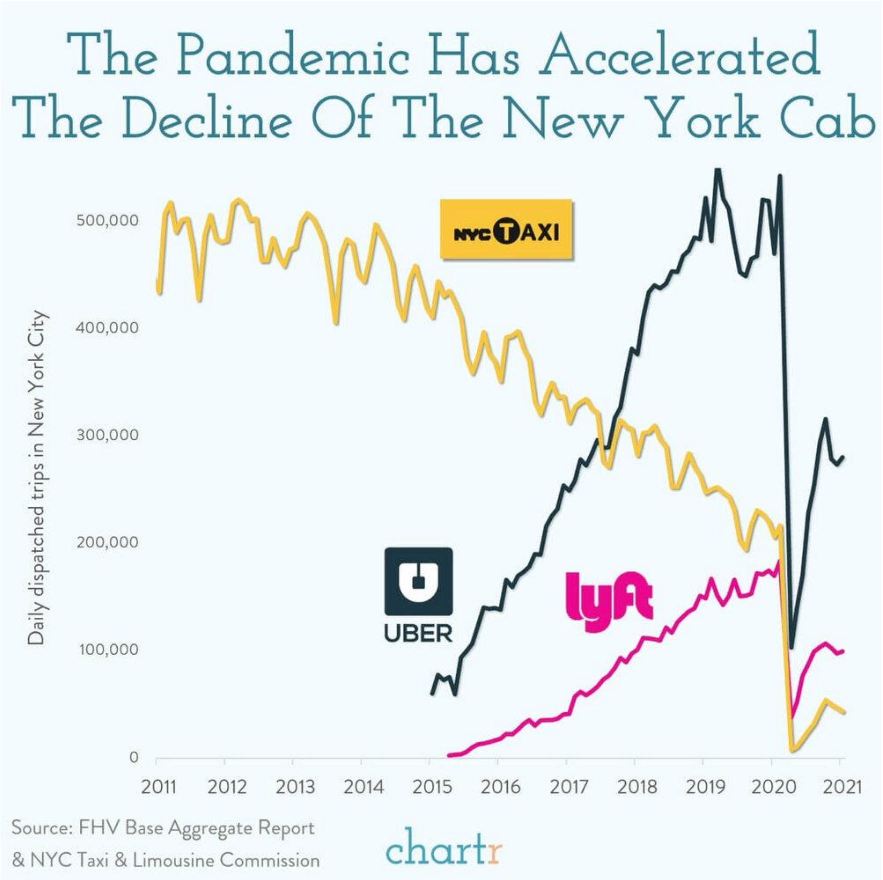
Taxi Industry Faces Strategic Crisis

Losing Riders

- Taxi cab ridership fell from 500,00 trips in 2012, to 230,000 in 2019
- Ride-hailing apps like Uber and Lyft make six times as many trips as yellow cabs

Losing Drivers

- Medallion prices plunged from $1 million in 2013/2014 to between $120,000 and $130,000 today.



But Strong Pandemic Rebound May Indicate Underlying Strengths

- The US recovery daily taxi trips in New York City **surged more than 800%** in April 2021 from a year earlier, while Uber and Lyft only rebounded **220%**

Why Ride-sharing Recovery Lags Behind

- Uber and Lyft struggling to find drivers due to pandemic, pushing prices up to 40% in April 2021
- Though ride-share's baseline prices are consistent, frequent surges causing fares to balloon

Taxi companies should not compete directly in ride bookings

Ride-sharing companies backed by funds with large war chests

Have elite tech talent and large pools of data to optimise booking services

Increasingly have other services on the same platform to keep users hooked

But instead rely on inherent strengths to win back market position

Freedom of Movement, no “Control Tower”

Ensures a base of drivers who will not shift to ride-sharing

Street Hailing

A revenue source inaccessible to ride-sharing platforms

Leading Us to the Problem Statement

How can data science help drivers make more money, more efficiently, without telling them what to do?

Proof-of-Concept: TaxiBuddy Helper App

Key Design Principle:
Enhance Inherent Strengths of Taxi Companies

Key Features:

Preserve **driver freedom** with recommendations, not directives

Maximise market information about most profitable areas for **street hails**

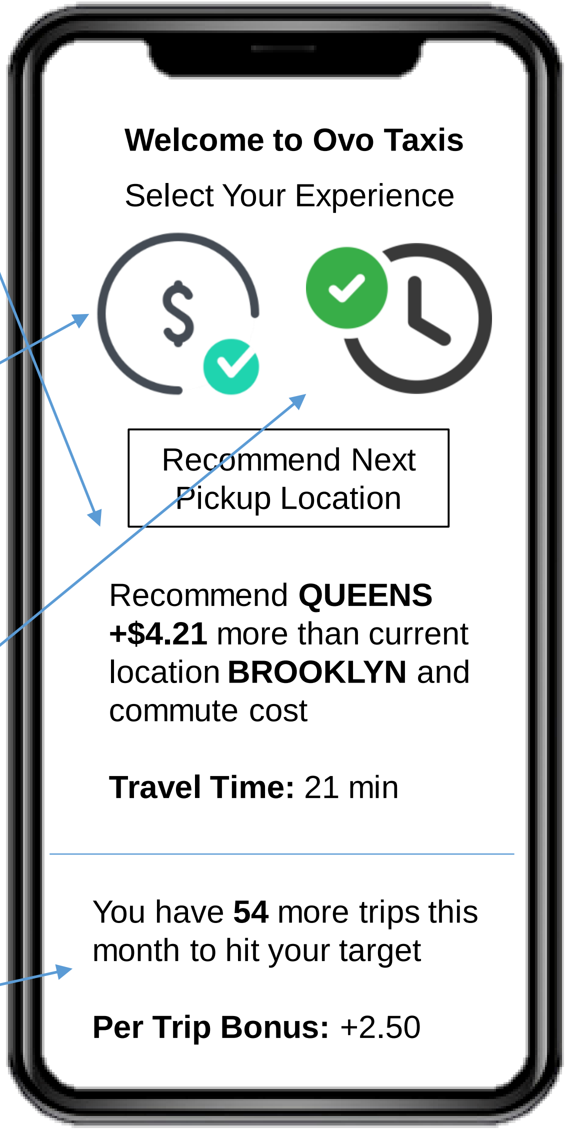
Provide stretch **targets** with clear incentives

Qn 1 & 2: What method to best predict?

Qn 3: Maximising Earnings,

Qn 4: Minimising Time, Maintaining Average

Qn 5: Maximising Earnings, 10 Taxis



Using Historical Data to Build TaxiBuddy POC

Dataset Key Facts

- 15m rows, 21 columns
- No significant nulls

Important Variables

- Pickup Time/Date
- Pickup Latitude/Longitude
- Dropoff Latitude/Longitude
- Trip Distance
- Passenger Count
- Fare Amount

Important Contextual Information

- Initial charge: **$2.50**
- Mileage: **40 cents per 1**/5 mile
- Waiting charge: **40 cents** per 120 seconds
- JFK flat fare: $45. (was $35)
- Newark surcharge: $15. (was $10) 4 p.m.–8 p.m. weekday.

Data Processing

Raw Data

Feature Engineering

- Date and Time, Day of Week, Wknd/Weekday
- Speed, Geo distance, Log distance
- Pickup and Dropoff areas (Boroughs, Airports)

Records that met the following criteria were excluded from all subsequent analyses:

- Distance of 0 mile or distance ≥ 50 miles;
- Duration of 0 minute or duration ≥ 200 minutes;
- Average speed ≤ 1 MPH or average speed ≥ 240 MPH;
- Base fare < 2.50 or ≥ 250.00, or tip amount > twice the base fare;
- With invalid longitude or latitude data, or with a trip distance shorter than the geographic distance between pickup and drop-off point by more than 1 mile, or traveled outside the 5 boroughs or 3 airports of NYC.

Analytical Data Set

Dataset for tip prediction model

Further Data Processing

- Dropping all cash payments, negotiated fare

Dataset for fare prediction model

Q1.

In what trips can you confidently use respective means as measures of central tendency to estimate fare, time taken etc.

What Distribution Shapes Allow for Means as Measure of Central Tendency?

Qn 1

Business Considerations

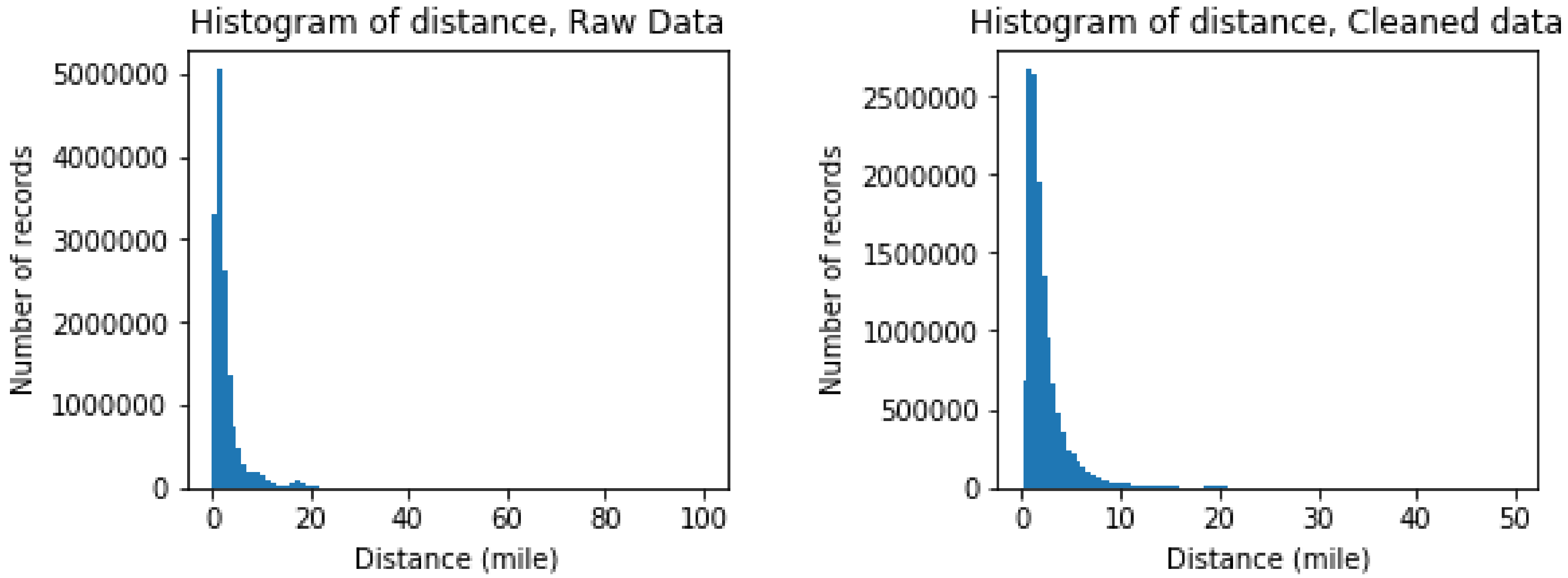
- Means as a measure is easily interpretable and intuitive
- If the data allows it, it is a good option to communicate aggregates of important variables to stakeholders e.g. drivers, riders, investors

Technical Considerations

- However, in non-normal distributions, median is preferable

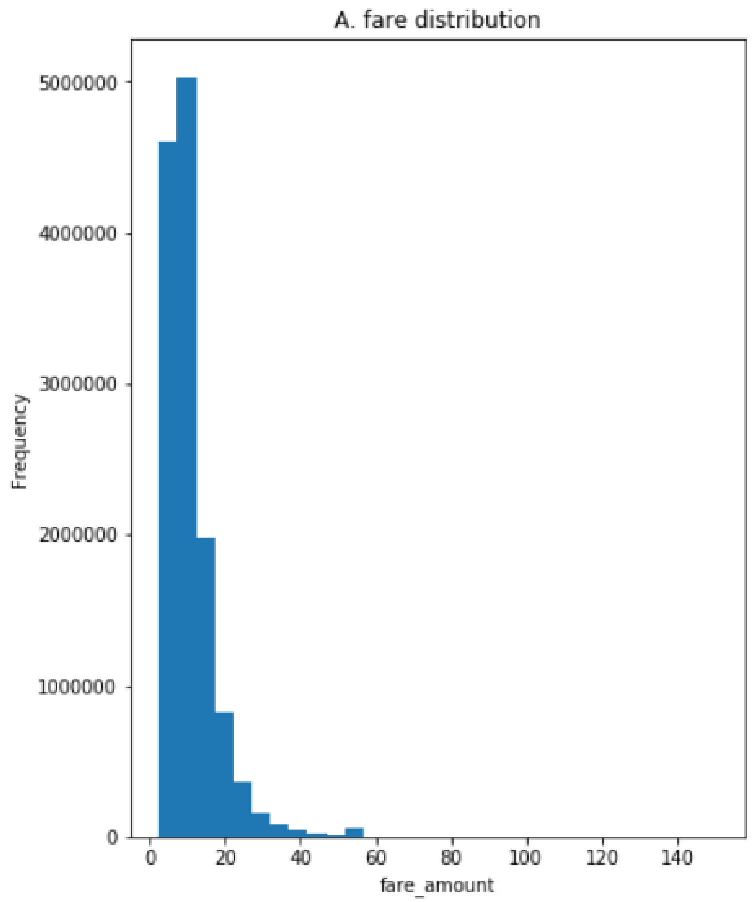
Qn 1

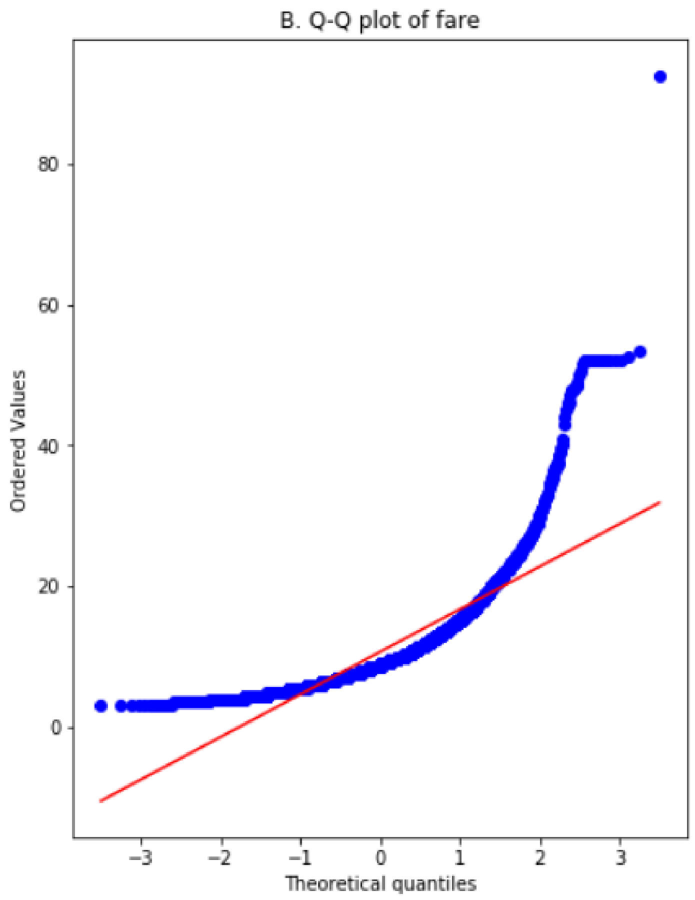
Trips By Distance



Distribution of trip distance is right-skewed. Mean could be inflated by a few long-distance trips. Indicates that median may be a fairer representation of the central tendency of all distances.

Trips by Fare

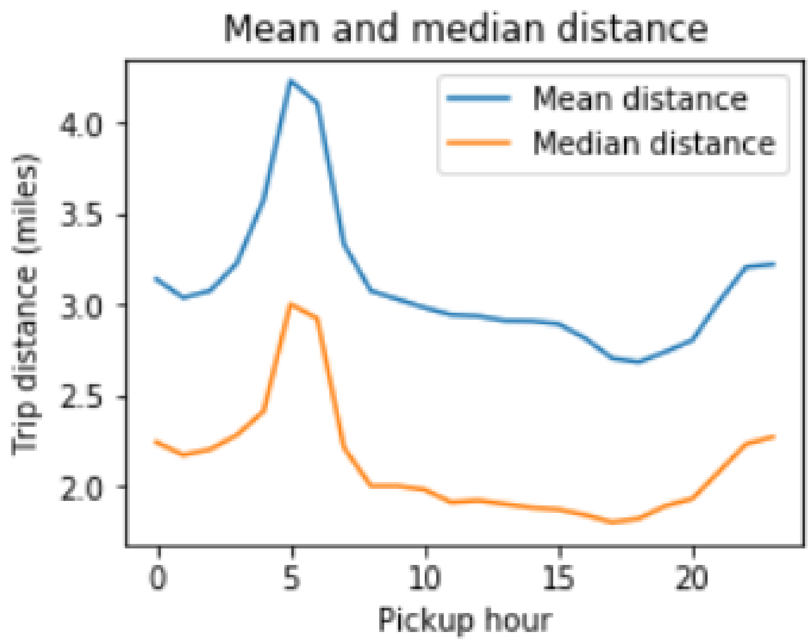


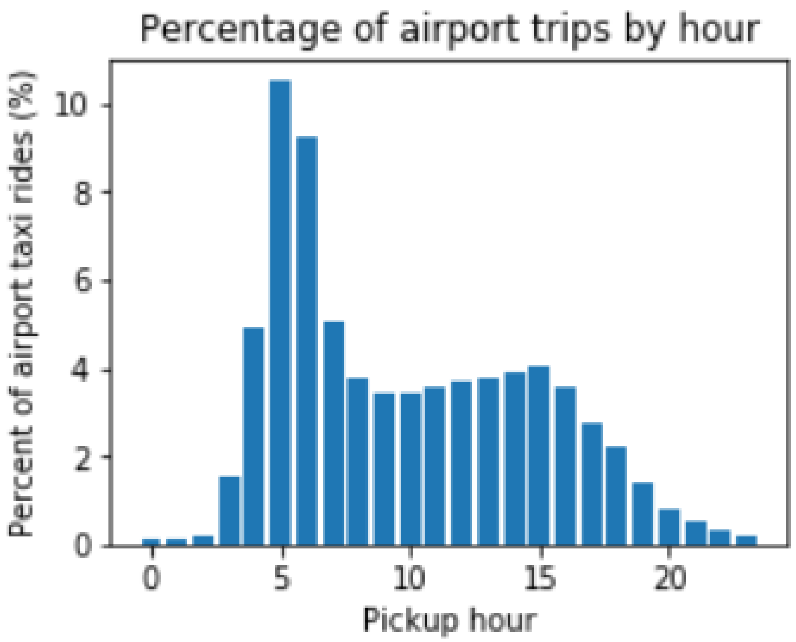


Fare distribution also right skewed, heavily clustered around the $10-$15 range

Trips by Category of Trip (Airport Trips Vs. Non-Airport)

Qn 1

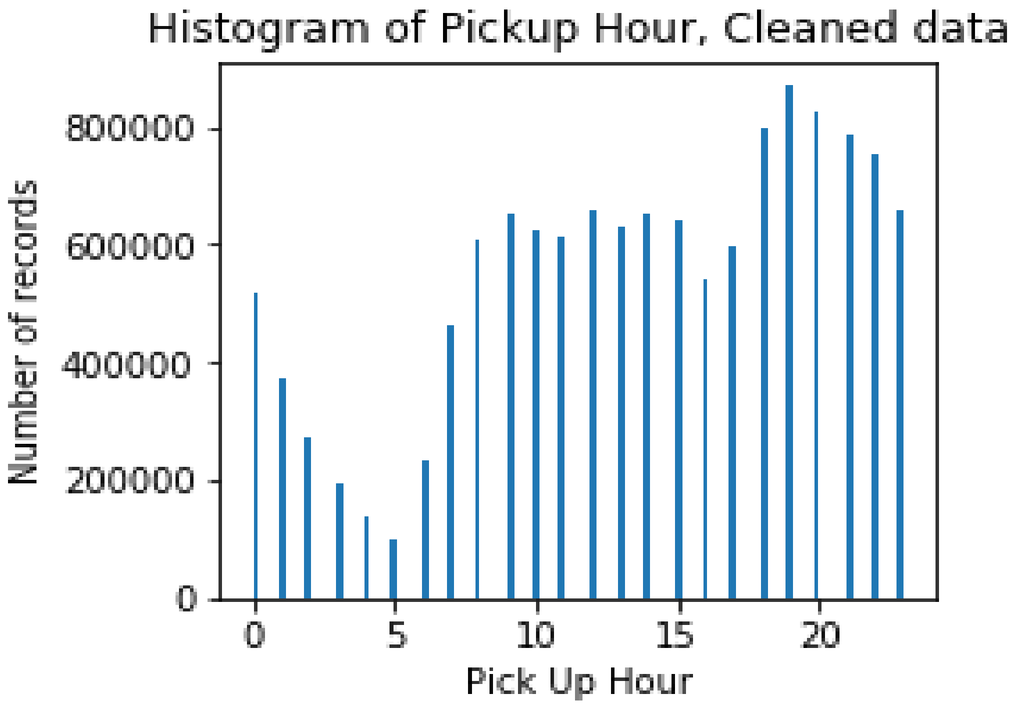


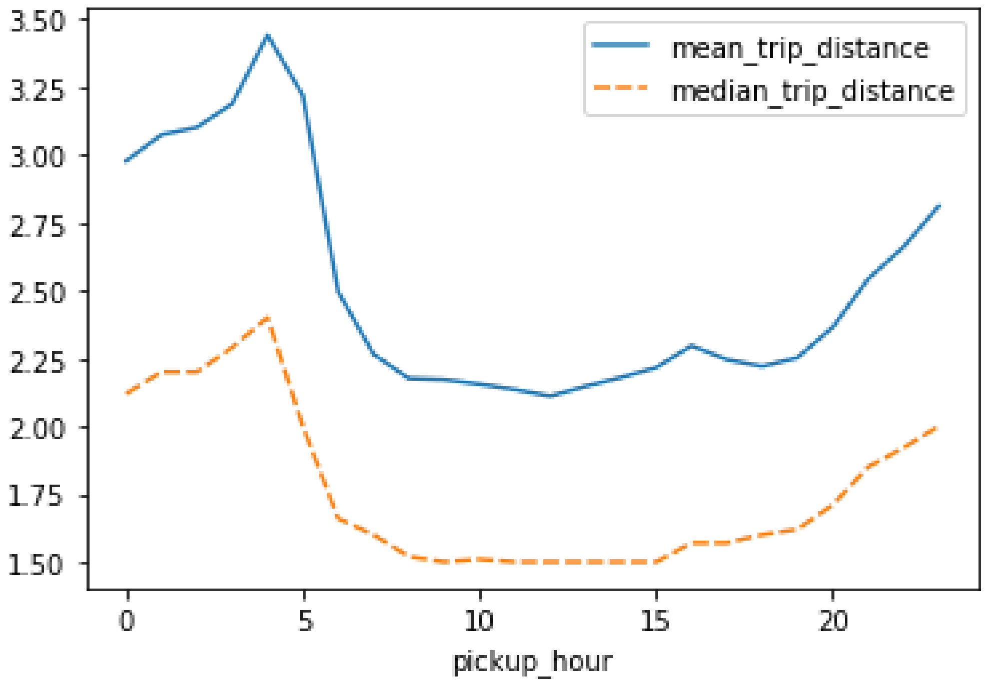


Right skew observed when plotting against counts of different categories of trips (Airport vs non-airport). Once again, mean also consistently higher than the median

Qn 1

Trips by Hours





Non-normal dist. observed when plotting against duration of trips.
Mean also consistently higher than the median

Qn 1

Implications of Non-normal Distribution

- Central Tendency
  - Median is a more reliable measure of central tendency

- Business/Product Implications
  - To give drivers accurate estimations/predictions, will need more powerful methods to overcome lack of normality in the data

Machine learning models should therefore be considered as a method to build fare/tip estimation service in TaxiBuddy

Q2.

Can we build a model to predict fare and tip amount given pick up and drop off coordinates, time of day and week

Qn 2

Building Separate ML Models to Predict

- **Business Considerations**
  - Model error must be acceptably low so that drivers can make the best decisions
  - Final model designed with live service of TaxiBuddy in mind e.g. interpretability, compute load, efficiency of “MLops”
- **Technical Considerations**
  - Intuitively, tip and fare amounts may be driven by different factors, e.g. faster speed and distance respectively
  - Tip amount itself may inversely relate to fees

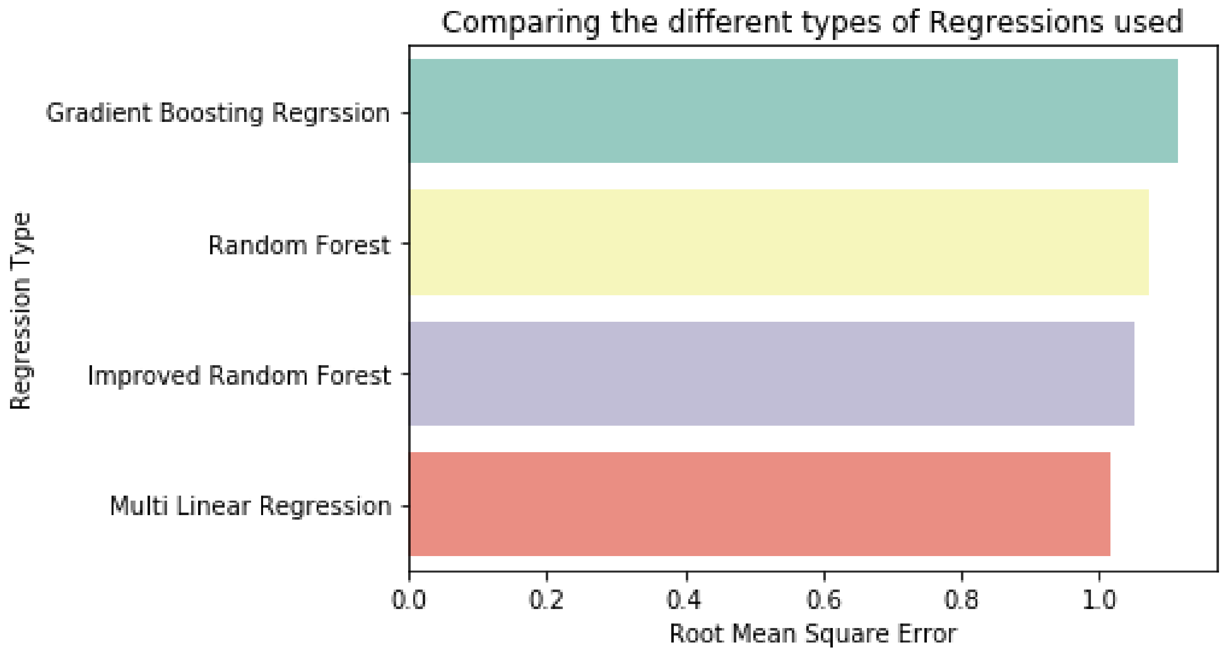
Two separate ML models will be produced and tested to assess for feasibility of tip and fare prediction

Qn 2

Predictive Model for Tip Amount

Results (on lin reg)
- RMSE = 1.02 (very high considering median tip is 1.8
- MAPE: = 45.7

Reasons for Poor Performance
- Lack of information to explain tip behaviour. May be driven more by individual idiosyncrasy, personality. May need more information e.g. rider/driver history, past reviews



Predictive model for tip amount performs poorly, which requires us to diagnose the issue

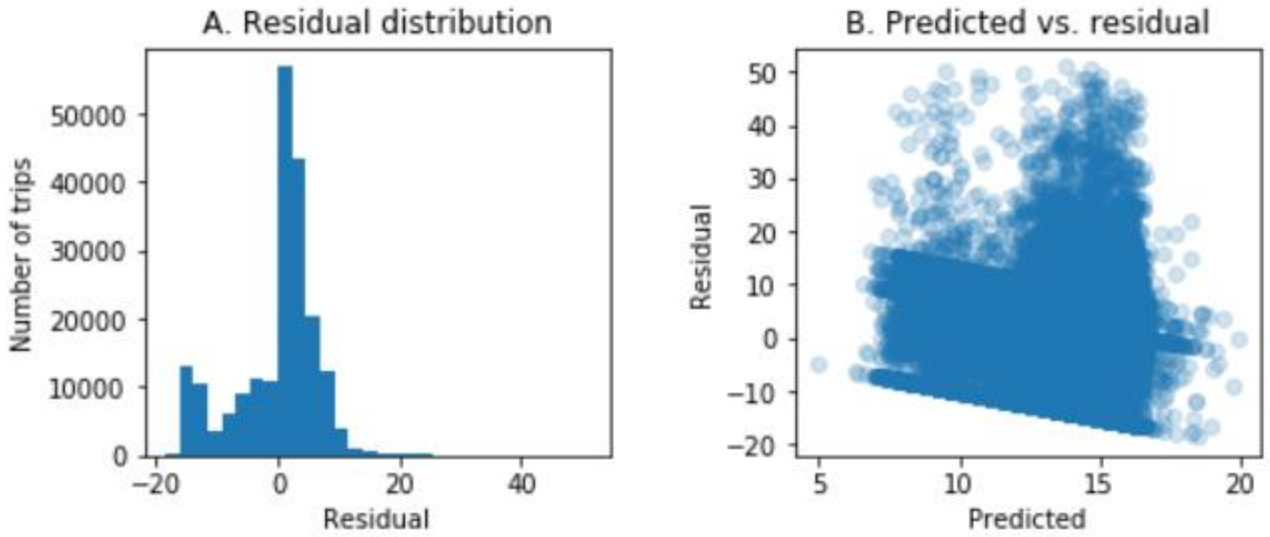
Why is Performance Bad for Tip Prediction Model?

Diagnosis

- Residuals are neither normally distributed nor stable over the predicted values (i.e., violation of homoscedasticity assumption)

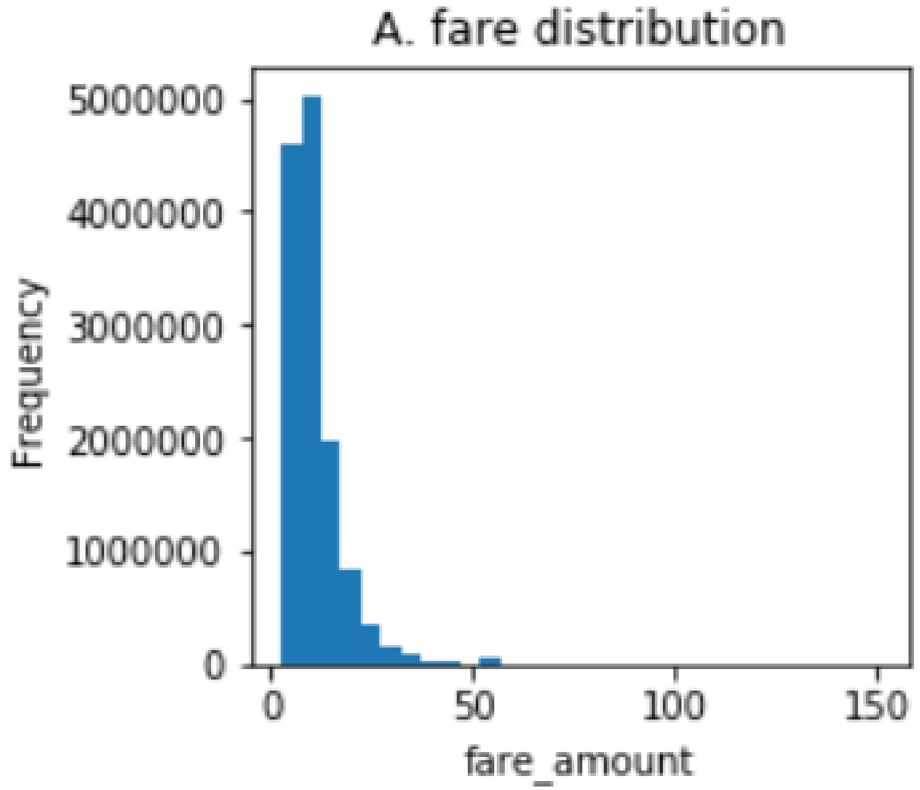
Possible Reasons

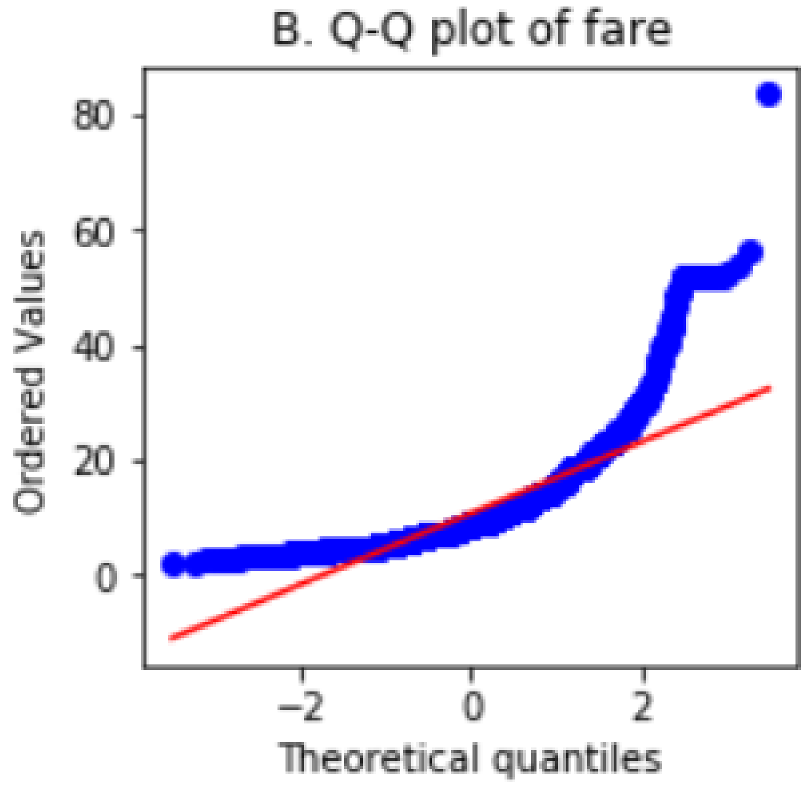
- Key information that are predictive of tipping behavior are not available. E.g. socio-demographic information about the passenger, past tipping rate of passenger, driver's driving style, and car condition, etc.



Not feasible to build a predictive model for tip amounts. Our service will therefore exclude this feature

Predictive Model for Fare Amount





Fare distribution is also non-normal, which may indicate that linear regression methods are less reliable here

Qn 2

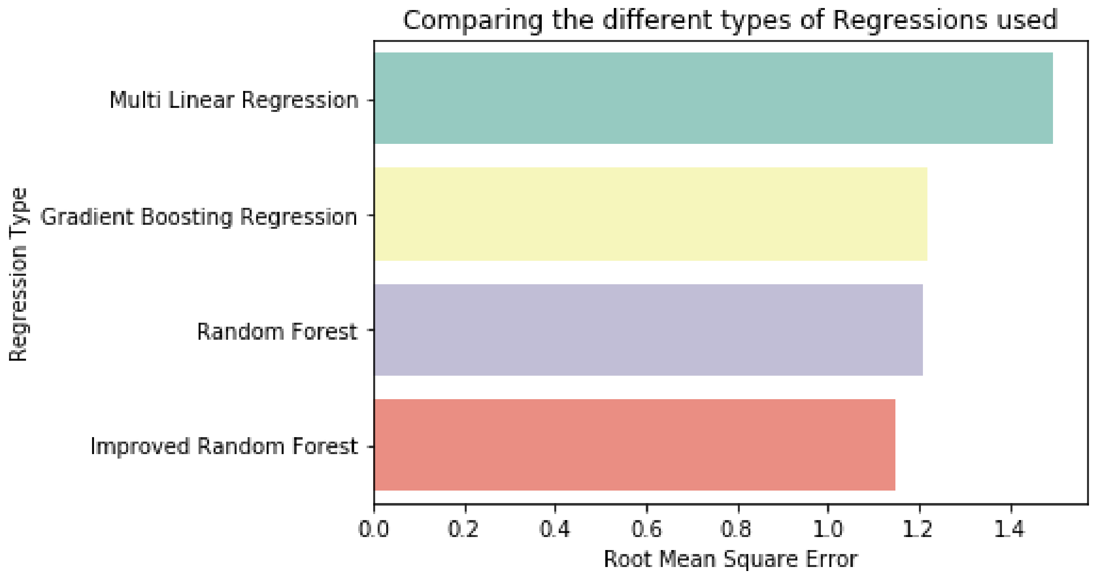
Predictive Model for Fare Amount

Results (on lin reg)

- RMSE = 1.35
- MAPE = 4.6

Important Features

- Dropoff_hour
- Trip_distance
- Pickup_hour
- Certain dropoff destinations



Predictive model for fare amount shows decent performance across a range of regressions types

Qn 2

Fare Model Selection – Balancing Tradeoffs

Business Considerations

- Lin Reg gives slightly more explainability. Able to show increase in mean of DV with every unit increase of IV. Can help design better policies
- At the same time, RMSE diff of -0.4 weakens a service that depends on accuracy

After consideration, we select our tuned RF model as it provides the best RMSE (most accurate) while retaining a degree of explainability

Qn 2

Selected Model for TaxiBuddy

Model

RandomForest (Tuned)

MSE = 1.15

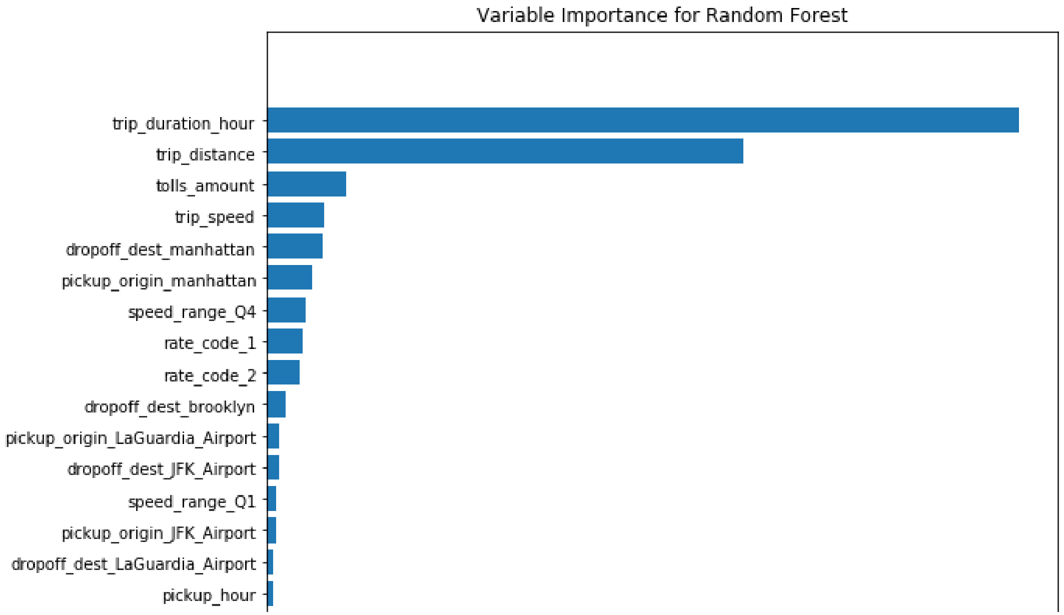
Selected Features

Pickup_hour

Day_of_week (e.g. Monday)

Pickup and Dropoff locations

(one-hot encoded)



Important variables for RF model

We select our RF model given the satisfactory performance and its relative explainability

Q3.

If you were a taxi driver, how would you maximize your earnings in a day?

Qn 3

Optimising a Taxi Driver's Earnings - Considerations

Business Constraints of Implementing Mathematically Optimal Policies from Historical Data

- Unlike ride-sharing, customers' destinations are not known in advance. They may redirect cab too far away from high-density location at required times

Technical Considerations

Friction involved in moving between two locations at any time

- Customers' destinations are not known in advance
- Fixed dispatch schedules may not work due to changing local conditions

Cursory research online shows solutions favouring discovering mathematically optimal passenger-dense locations for specific times. However, they have real-world implementation problems

Qn 3

More Realistic Solution - Bounded Greedy Search

Driver Selects "Money-making" Mode

→ Driver asks for Recommendation app searches for nearest X no of areas

→ App makes call to ML service with updated features, retrieves predicted fare amount for pickups at nearby areas

→ App calculates whether staying or moving is more optimal

→ App recommends where to go, provides expected earnings

Calculates opportunity cost of staying by comparing predicted fares from location X against Y + value of the commute (calculated as the mean trip fare between X and Y – base fare), and selecting the greater one

Design Consideration

Opportunity cost can be valued in different ways

(predicted fare) OR (mean/med historical fare) minus base fare (to account for fact that no passengers make the commute)

Qn 3

Demonstration

Qn 3

Testing Results - Simulation and Backtesting

Programmed entire workflow (viewable in GitHub) to generate

Simulation

- Generate predicted earnings across a range of starting dates, hours worked, and locations
- For example, when initialised starting conditions as May 5, most optimal solution is to start at JFK Airport at 9pm with predicted earnings of $902

Back Testing

- Simulated 2000 drivers with random initial conditions giving predicted median fare of $576 against historical median fare of $377

	start_time	end_time	earnings	hours_worked	start_location
1493	2013-05-09 21:00:00	2013-05-11 22:36:50.000	902.778667	9	JFKAirport
466	2013-05-09 03:00:00	2013-05-10 18:09:05.000	863.465832	8	brooklyn
1331	2013-05-09 03:00:00	2013-05-11 05:10:00.500	846.820064	9	JFKAirport
170	2013-05-09 18:00:00	2013-05-11 17:42:02.000	843.800592	9	NewarkAirport
827	2013-05-09 19:00:00	2013-05-11 19:51:42.500	815.996317	9	manhattan
1520	2013-05-09 00:00:00	2013-05-10 22:52:28.500	807.718736	9	LaGuardiaAirport
1718	2013-05-09 22:00:00	2013-05-11 22:21:06.500	798.154052	9	LaGuardiaAirport
350	2013-05-09 14:00:00	2013-05-11 12:53:14.500	791.931939	9	bronx
53	2013-05-09 05:00:00	2013-05-11 05:54:38.000	780.162989	9	NewarkAirport
1061	2013-05-09 21:00:00	2013-05-11 21:43:03.500	777.642700	9	queens
1366	2013-05-09 07:00:00	2013-05-10 21:12:16.000	771.538983	8	JFKAirport

	pickup_day_type	sum	nunique_drivers	sum_days_worked	nunique_days	day_type_mean_earnings
0	Weekday	1.021748e+08	13270	283922	22	349.985728
1	Weekend	3.896769e+07	13136	100053	8	370.810126

Q4.

If you were a taxi owner, how would you minimize your work time while retaining the average wages earned by a typical taxi in the dataset?

Bounded Greedy Search with Earnings Cut-off

Driver Selects "Time-Saving" Mode

Driver asks for Recommendation app searches for nearest X no of areas

App makes call to ML service with updated features, retrieves predicted fare amount for pickups at nearby areas

App calculates whether staying or moving is more optimal

App recommends where to go, provides expected earnings

App tracks average driver earnings in rolling window of last X months. Displays target

App tracks average driver earnings in rolling window of last X months. Displays target

pickup_day_type	nunique_drivers	sum_days_worked	nunique_days	day_type_mean_earnings
Weekday	13270	283922	22	349.985728
Weekend	13136	100053	8	370.810126

Qn 5

Adding an earnings cut-off to minimise hours

Simulation

- Identify which conditions (starting dates, hours worked, and locations can minimise working hours while maintaining average earnings

start_time	end_time	earning	hours_worked	start_location
11:00:00 AM	4:21:18 PM	373.5269	5	NewarkAirport
2:00:00 PM	5:24:05 PM	374.8103	5	NewarkAirport
6:00:00 PM	11:33:14 PM	378.7932	5	NewarkAirport
8:00:00 AM	12:40:12 PM	375.6762	5	bronx
3:00:00 PM	9:16:25 PM	374.8176	5	bronx
1:00:00 AM	6:27:43 AM	372.2097	5	brooklyn
4:00:00 AM	10:14:43 AM	376.0992	5	manhattan
9:00:00 AM	1:36:10 PM	373.5183	5	manhattan
1:00:00 PM	7:00:30 PM	377.4975	5	manhattan
4:00:00 PM	8:45:10 PM	370.9094	5	manhattan
5:00:00 PM	10:14:59 PM	377.511	5	manhattan
10:00:00 PM	2:53:44 AM	379.98	5	manhattan
10:00:00 AM	1:49:42 PM	378.2357	5	queens

Qn 4

Limitations and Next Steps

Approach Assumptions/Limitations

- Assumes continuous work without breaks
- Currently unable to factor in disjointed sessions in a day by the same driver e.g. working from 9am-11am and then 3pm-6pm, which might actually give more optimal results
- For POC, origin and destinations are large sized areas (boroughs). Going more granular list of locations more realistic

Q5.

If you run a taxi company with 10 taxis, how would you maximize your earnings?

Qn 5

Incentives as the most relevant way to optimise earnings for fleet of drivers

Business Considerations

- Economics of taxi industry - drivers are target oriented
- In real world, optimising 10 taxis is question of driver incentivization, not mathematical optimisation

How to use data science to decide **when** and **how much** to incentivize and **for what**?

Qn 5

Designing Incentive Structures for Different Groups

Approach

- Unsupervised learning to separate groups of drivers based
- PCA to discover differentiating variables – ‘trip count’
- Establish targets e.g. attaining trip counts of 75%/100% percentile levels of the next highest cluster
- Calculate available budget for financial incentives per trip, depending on fixed params e.g. company fee

Earnings Guarantee

	Minimum trip count per week	Guaranteed average fare per trip
Emerald	40	$12.00
Ruby	40	$12.50
Sapphire	60	$13.00
Diamond	60	$13.50

Grab driver reward program

Qn 5

Designing Incentive Structures for Different Groups

Driver	Trip Count	Current Cluster	Target Cluster	Target 100% percentile (trips)	Earnings gap	Incentive budget	Incentive per trip
#1	400	Low	Average	700	$3000	100/300	0.5

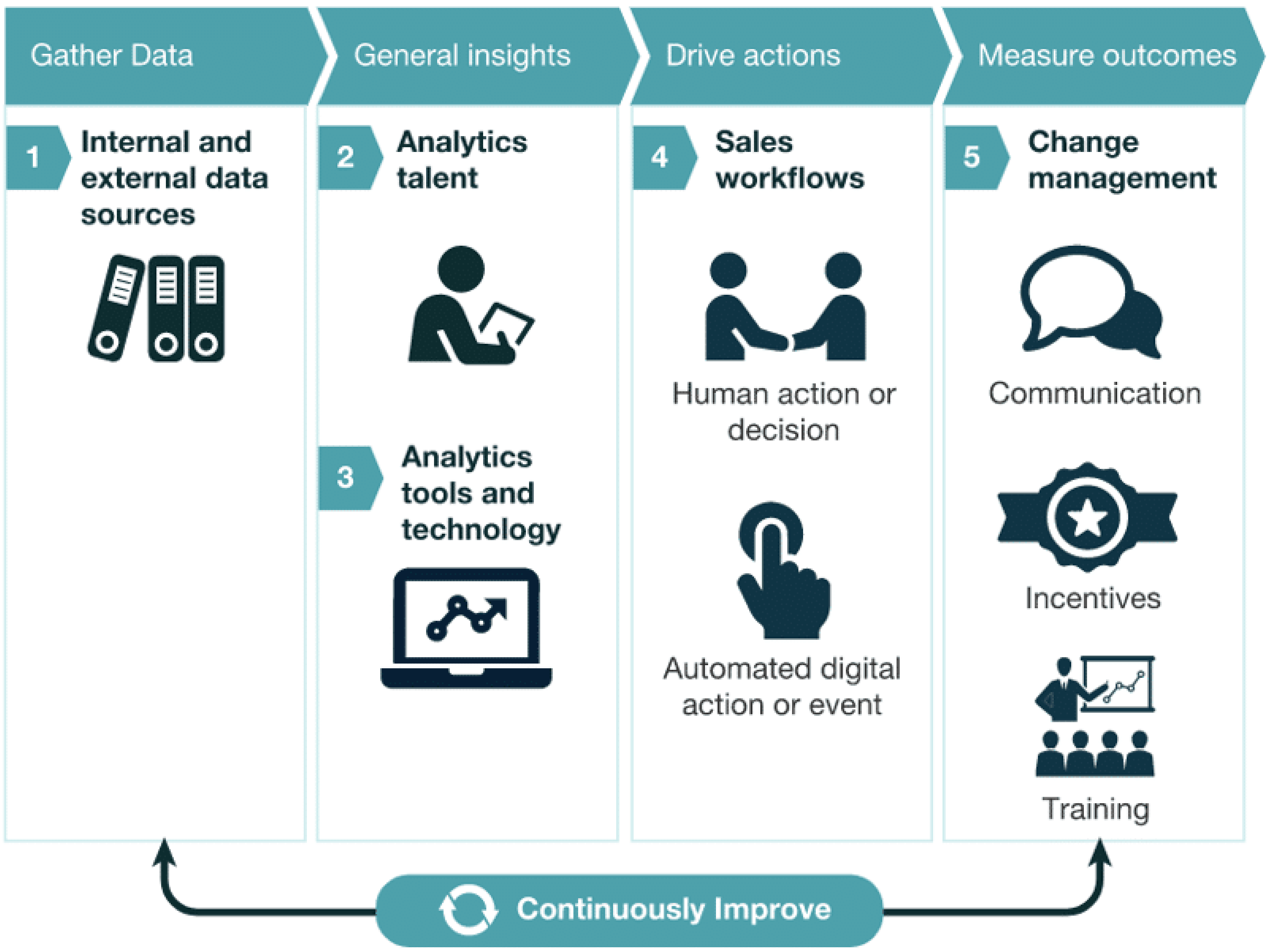
Various Ways to Incentivize

- Incentive per trip
- Ladder of increasing incentives
- Minimum trip incentive

Conclusion – What is Good Data Science?

- Data science must serve business problems with business-ready solutions, while remaining technically robust
- Solutions must be tested and validated, against test sets, historical performance, and the **real world**. Generally, the simpler the solution, the better.
- Data scientists also need to contextualise data and methods in ways that are **relevant** and **friendly** to business users
- Within the short time I had, I tried to demonstrate the above
- Thank you for the opportunity

Annex


Gather Data
General insights
Drive actions
Measure outcomes
1 Internal and external data sources
2 Analytics talent
3 Analytics tools and technology
4 Sales workflows
Human action or decision
Automated digital action or event
5 Change management
Communication
Incentives
Training
Continuously Improve

Features in the dataset:

Field Name	Description
VendorID	A code indicating the TPEP provider that provided the record. 1. Creative Mobile Technologies 2. VeriFone Inc.
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
Pickup_longitude	Longitude where the meter was engaged.
Pickup_latitude	Latitude where the meter was engaged.
RateCodeID	The final rate code in effect at the end of the trip. 1. Standard rate 2. JFK 3. Newark 4. Nassau or Westchester 5. Negotiated fare 6. Group ride
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server. Y= store and forward trip N= not a store and forward trip
Dropoff_longitude	Longitude where the meter was disengaged.
Dropoff_ latitude	Latitude where the meter was disengaged.
Payment_type	A numeric code signifying how the passenger paid for the trip. 1. Credit card 2. Cash 3. No charge 4. Dispute 5. Unknown 6. Voided trip
Fare_amount	The time-and-distance fare calculated by the meter.
Extra	Miscellaneous extras and surcharges. Currently, this only includes. the $0.50 and$ 1 rush hour and overnight charges.
MTA_tax	0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	0.30 improvement surcharge assessed trips at the flag drop. the improvement surcharge began being levied in 2015.
Tip_amount	Tip amount – This field is automatically populated for credit card tips.Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.

Taxi Fare Details

- Initial charge: **$2.50**
- Mileage: **40 cents per 1**/5 mile
- Waiting charge: **40 cents** per 120 seconds
- JFK flat fare: $45. (was $35)
- Newark surcharge: $15. (was $10) 4 p.m.–8 p.m. weekday.

0.9618485052618595 0.9596242273162401

- Fare model all features

```
lm = LinearRegression()
lm.fit(X_train,y_train)
print(lm.score(X_train,y_train))
print(lm.score(X_test,y_test))
```

```
0.9618485052618595
0.9596242273162401
```

```
y_pred = lm.predict(X_test)
lrmse = np.sqrt(metrics.mean_squared_error(y_pred, y_test))
lrmse
```

```
1.3869912173526915
```

Model 2

```
lm = LinearRegression()
lm.fit(X_train,y_train)
print(lm.score(X_train,y_train))
print(lm.score(X_test,y_test))
```

```
0.9564295667586646
0.9555304641506288
```

```
y_pred = lm.predict(X_test)
lrmse = np.sqrt(metrics.mean_squared_error(y_pred, y_test))
lrmse
```

```
1.4564910773151123
```

Model 2

```
: lm = LinearRegression()
  lm.fit(X_train,y_train)
  print(lm.score(X_train,y_train))
  print(lm.score(X_test,y_test))
```

```
0.9564295667586646
0.9555304641506288
```

```
: y_pred = lm.predict(X_test)
  lrmse = np.sqrt(metrics.mean_squared_error(y_pred, y_test))
  lrmse
```

```
: 1.4564910773151123
```