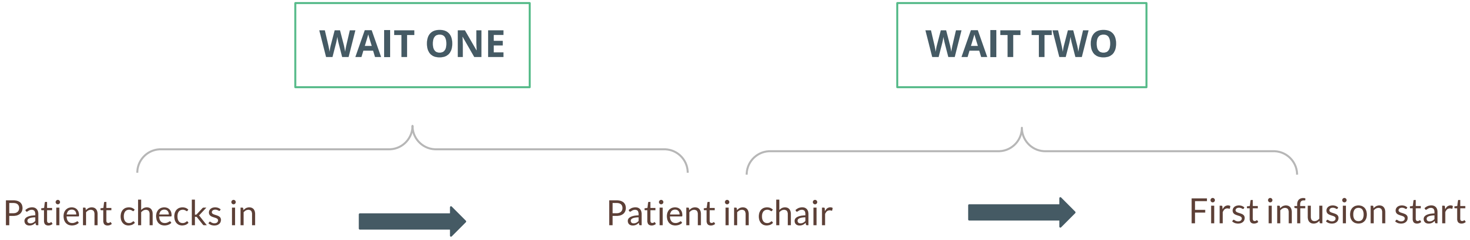
Parsing Infusion Center ABC’s Dataset and Interpreting Wait Times

Soumil Gad

Understanding the Patient Journey

In the current Patient Journey, there are **two** points where the patient is waiting



WAIT ONE

WAIT TWO

Patient checks in → Patient in chair → First infusion start

3 relevant columns for wait time analysis:

- CHECKIN_DTTM (DateTime Patient Checks in)
- CHAIR_START (Time Patient Seated in Chair)
- INFUSION_START (Time First Infusion was Started)

Observations about the Dataset

Major Issues:

* **Affects analysis** of wait time

1. Entries where patient received multiple medications (same appointment ID) have duplicate timestamps
2. Some data required for analysis (check in time, chair start time, infusion start time) is missing
3. Some timestamps are in illogical order (end time comes before start)

Minor Issues:

* **Doesn't affect analysis** of wait time

1. Some data shows patient checking in but not out
2. There was no data for time the patient left chair in any entry
3. Data in Appointment Date Time column is in inconsistent format (MM/DD/YY HH:MM and YYYY-MM-DD HH:MM AM/PM)

Assumptions

1. A wait time is calculated even if there are only two relevant timestamps populated

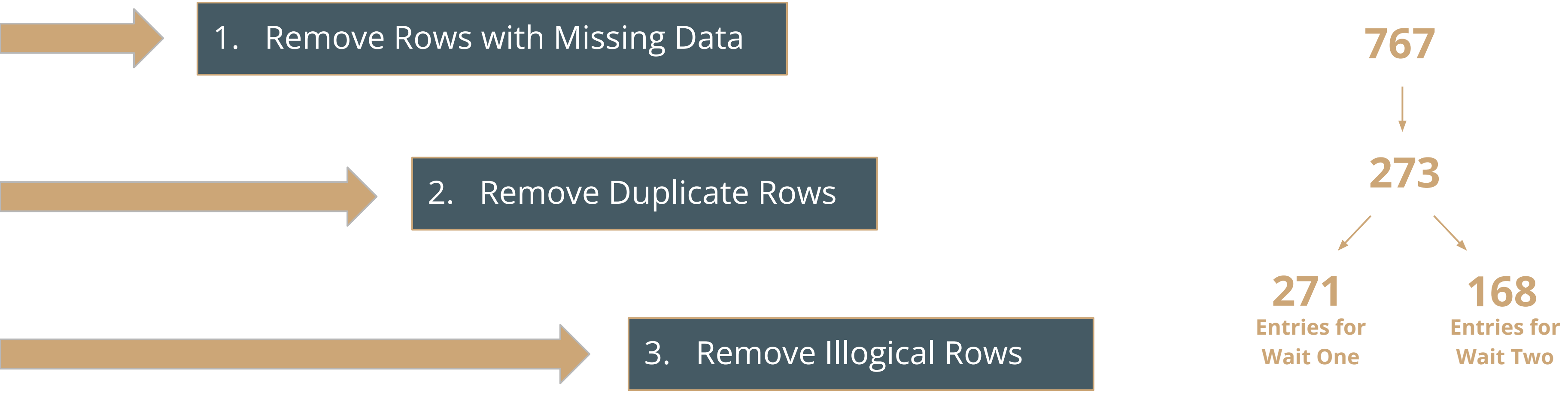
2. Patients wait ends at the start of their first infusion

3. Patients with a ‘cancel date’ that have data recorded still came in and had an infusion

4. If order of time stamps does not match patient journey, they are excluded from analysis

Cleaning the Data

3 steps to prepare data for wait time analysis

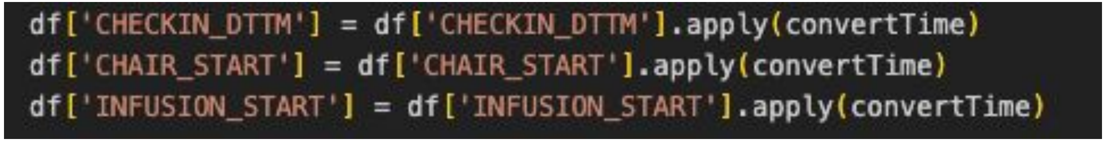


1. Remove Rows with Missing Data

2. Remove Duplicate Rows

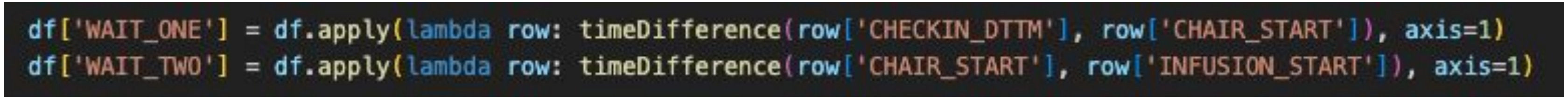
3. Remove Illogical Rows

Deep Dive - Removing Illogical Data



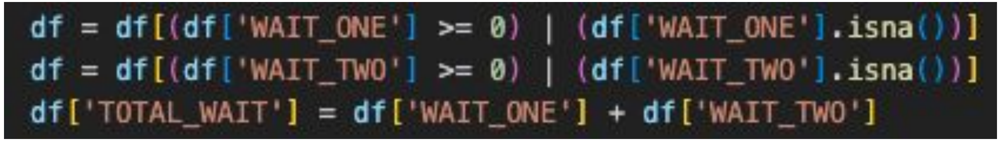
```
df['CHECKIN_DTTM'] = df['CHECKIN_DTTM'].apply(convertTime)
df['CHAIR_START'] = df['CHAIR_START'].apply(convertTime)
df['INFUSION_START'] = df['INFUSION_START'].apply(convertTime)
```

1. Convert all times to datetime.time objects



```
df['WAIT_ONE'] = df.apply(lambda row: timeDifference(row['CHECKIN_DTTM'], row['CHAIR_START']), axis=1)
df['WAIT_TWO'] = df.apply(lambda row: timeDifference(row['CHAIR_START'], row['INFUSION_START']), axis=1)
```

2. Calculate difference in timestamps to determine wait time



```
df = df[(df['WAIT_ONE'] >= 0) | (df['WAIT_ONE'].isna())]
df = df[(df['WAIT_TWO'] >= 0) | (df['WAIT_TWO'].isna())]
df['TOTAL_WAIT'] = df['WAIT_ONE'] + df['WAIT_TWO']
```

3. Eliminate all negative wait times / keep cells where at least one positive wait time is present

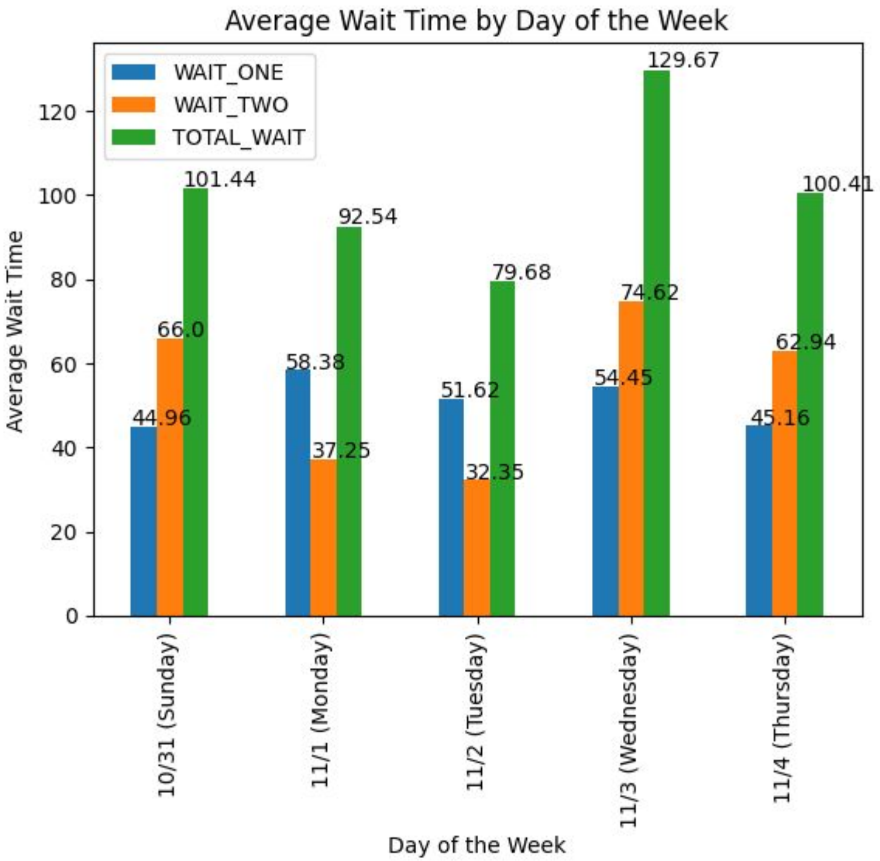
Analyzing Wait Times

On average, patients waited **51** minutes to get seated after checking in

On average, patients waited **56** minutes for infusion to start after being seated

On average, patients spent **102** total minutes waiting at infusion center ABC

Drawing Conclusions



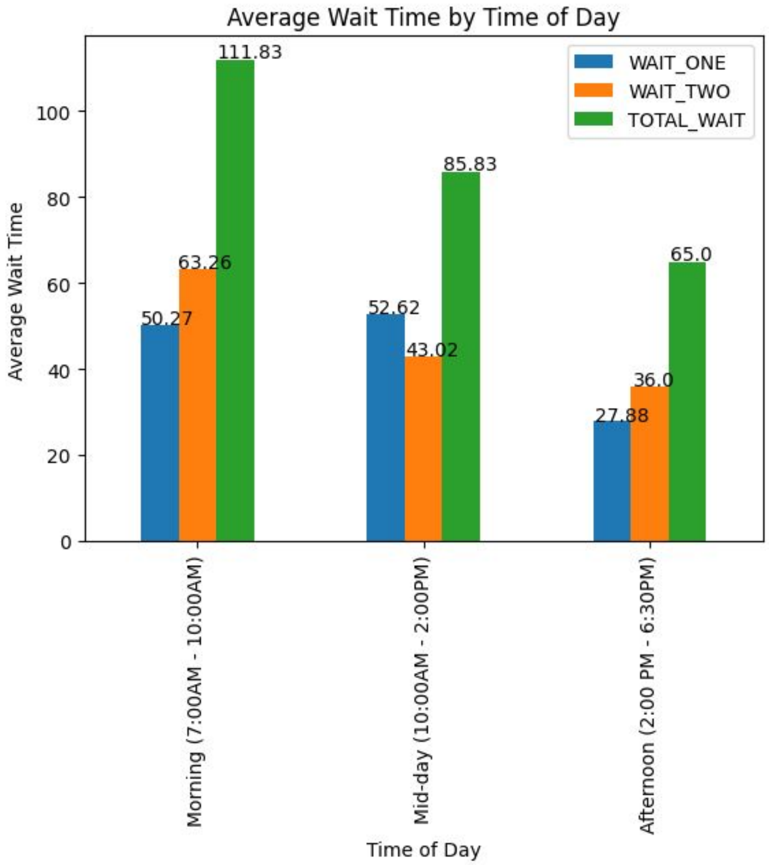
Longest Wait Time on Wednesday

~ 130 minutes spent waiting on average

Shortest Wait Time on Tuesday

~ 80 minutes spent waiting on average

Drawing Conclusions



Wait Time decreases by 42% from morning to afternoon

~ 112 minutes spent waiting on average in morning

~86 minutes spent waiting on average in mid-day

~65 minutes spent waiting on average in afternoon

Impacts of High Wait Time

Unhappy Patients

Loss of customers/revenue

Poor reviews

High Staff Turnover

Short staffed centers can’t utilize all possible chairs

Nurses may need to work longer hours in order to accommodate all patients