CitySense and CabSense

David Rosenberg

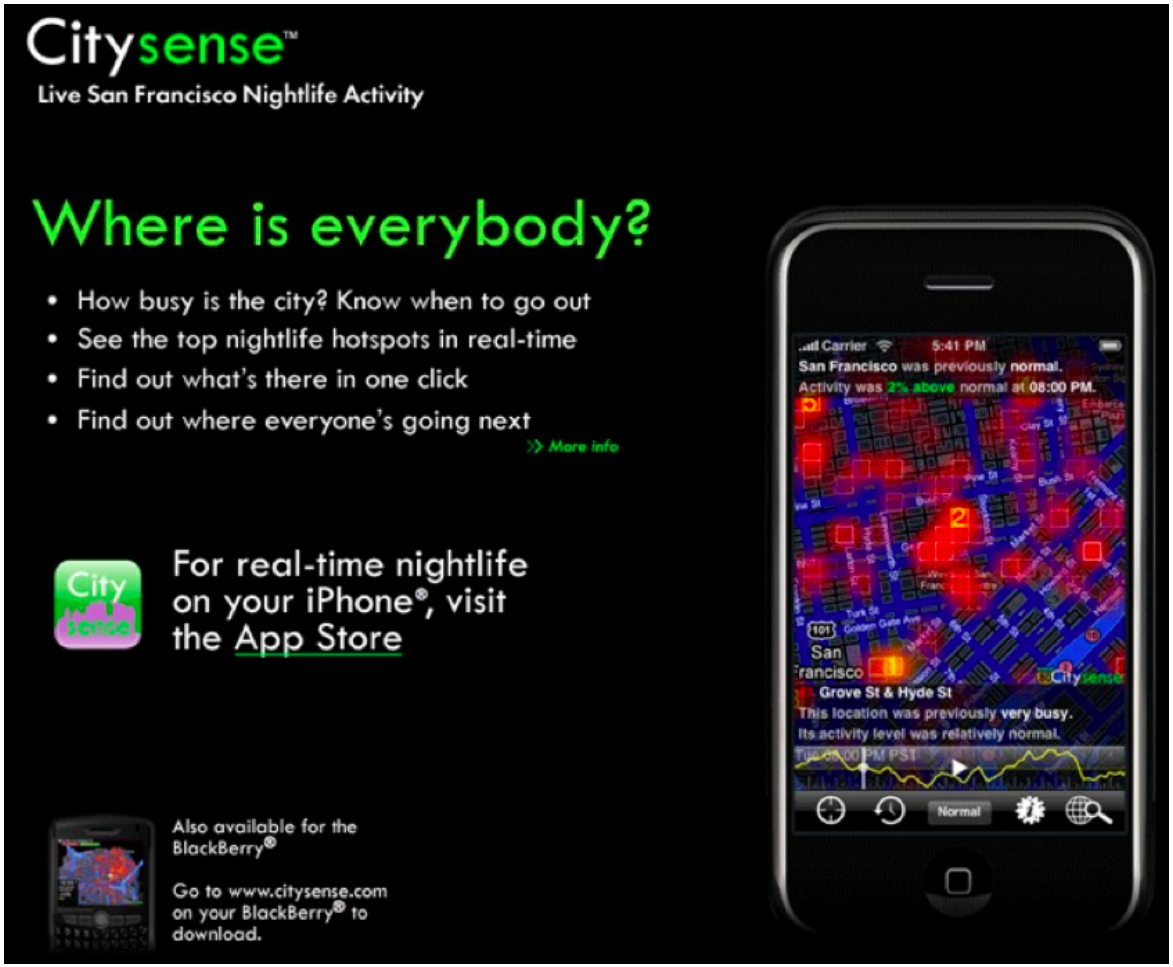
New York University

March 30, 2015

Sense Networks

- Startup company incorporated around 2006.
- Objective: Develop and leverage expertise in **location data** analytics.
- First product was called CitySense (2008).
  - A real-time, data-driven guide to nightlife in San Francisco.

CitySense (2008)



(Sadly, no longer in the App Store.)

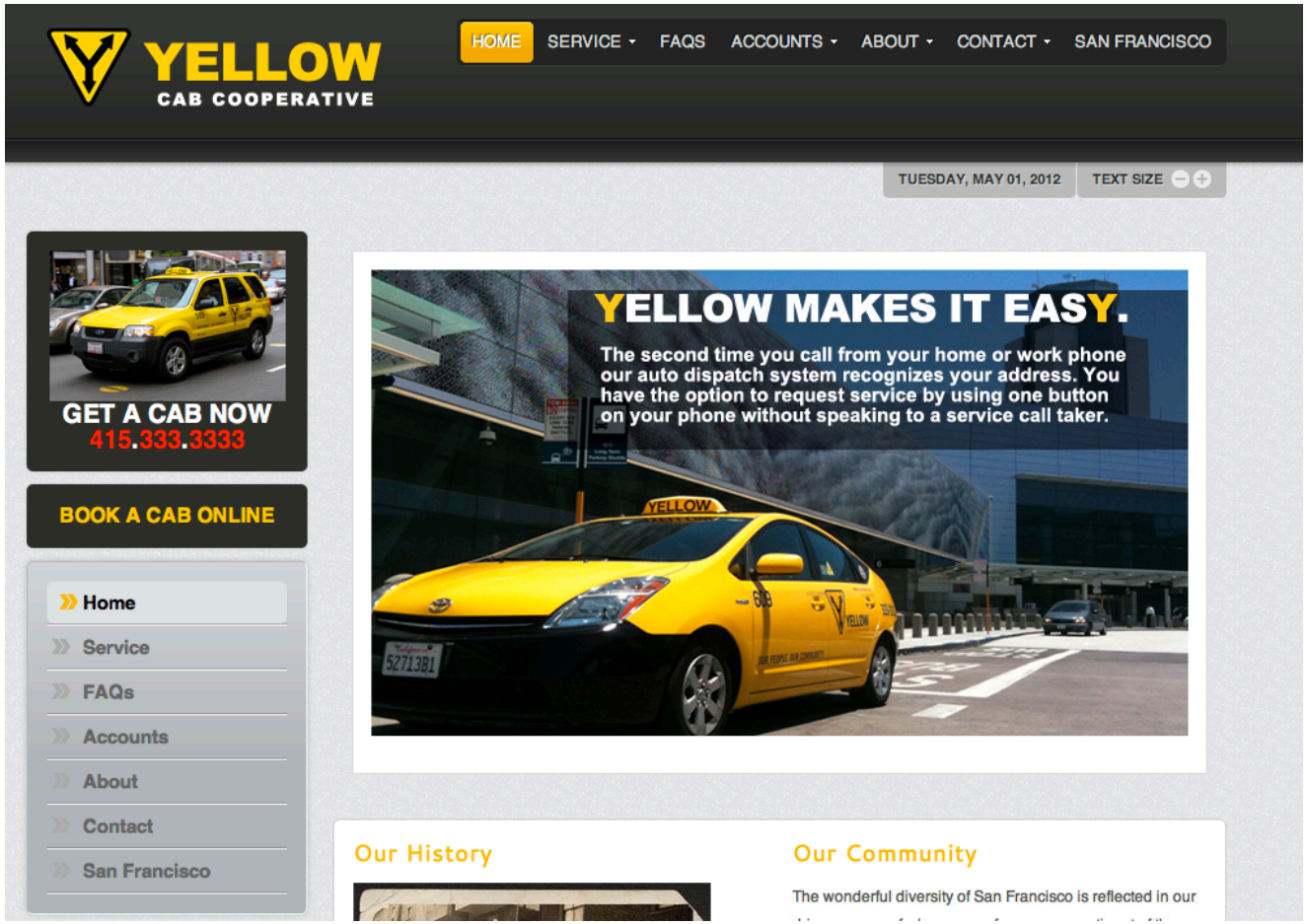
CitySense: Use Cases

Two use cases:

1. I'm new to the city – where does everybody hang out at night?
2. I know the city, but is there anything **special** going on tonight?

CitySense: Data Source

- Taxi GPS data for sale in San Francisco



CitySense

- Main Idea: Taxi destinations are a proxy for where people are going.
- Can use taxi data to bootstrap
  - Once we had users, we could use the locations from their phones.
- Taxi feed is **real-time**, so can use it to find those big secret parties.

CitySense

Data Science Strategy

1. Model "typical" behavior of each area of the city.
2. Rank areas with activity levels that are "most unusual".

We'll discuss modeling strategies shortly.

CabSense (2010): Second Product

Business objective

I'm in NYC, and I need a taxi. Where's the best place to find one?

Data Source

NYC Taxi and Limousine Commission provided GPS data from all taxi cabs. However, **not real-time**.

Main Idea

Historical taxi pickup frequency are predictive of future frequencies. Need to model pickup rates based on historical data.

CabSense (2010): Second Product

(Recently updated with fresh data.)

Picture courtesy of Blake Shaw.

Plan for this lecture

- We're going to focus on the CitySense "anomoly detection" problem.
- But we're going to use the NYC taxi pickup data, since we live in NYC.
- Our dataset is from 2009.
- Currently (2015/3/24) you can download 2013 data from `http://chriswhong.com/open-data/foil_nyc_taxi/`
- You can also request data directly from the NYC Taxi and Limousine Commission via the Freedom of Information Law.
  `http://www.nyc.gov/html/tlc/html/passenger/records.shtml`

Predicting Probability Distributions

- So far we've solved decision problems with a two types of action spaces:
  - $\mathcal{A} = \{-1, 1\}$ [hard classification, e.g. decision trees as used in AdaBoost]
  - $\mathcal{A} = \mathbf{R}$ [regression or soft classification]
- Today we consider a third type of action space:

$$\mathcal{A} = \{\text{Probability distributions a space } \mathcal{Y}\}$$

- Why?

The Joy of Probability Distributions

- Output space $\mathcal{Y} = \mathbf{R}$.
- Machine computes conditional probability density on $\mathcal{Y}$ given $x \in \mathcal{X}$:

$$x \mapsto p(y \mid x)$$

- If we know $p(y \mid x)$, we can find a $\hat{y}$ that minimizes any other loss function:
  - For square loss, give the mean of $p(y \mid x)$. [From homework #1]
  - For ℓ_1 loss, give the median of $p(y \mid x)$. [From homework #1]
  - etc.
- Gives idea of the possibilities of knowing the whole distribution.
- In practice, better to directly optimize the loss function of interest.

Probability Distributions for CitySense

- Suppose there are 100 taxi dropoffs
  - in a region R,
  - between 9pm and 10pm.
- Model predicts distribution $p(y \mid x)$ for the number of pickups.
- $\mathbb{P}(y \geqslant 100) = \sum_{y=100}^{\infty} p(y \mid x)$ measures how unusual this event is.
- Decision on what's interesting enough to present to the user goes beyond data science.

Probability Distributions for Prediction Intervals

- Given a probability distribution,
  - it's straightforward to give **prediction intervals**.
- A 95% prediction interval is an interval $[a, b]$ such that

$$\mathbb{P}(y \in [a, b] \mid x) \approx .95$$

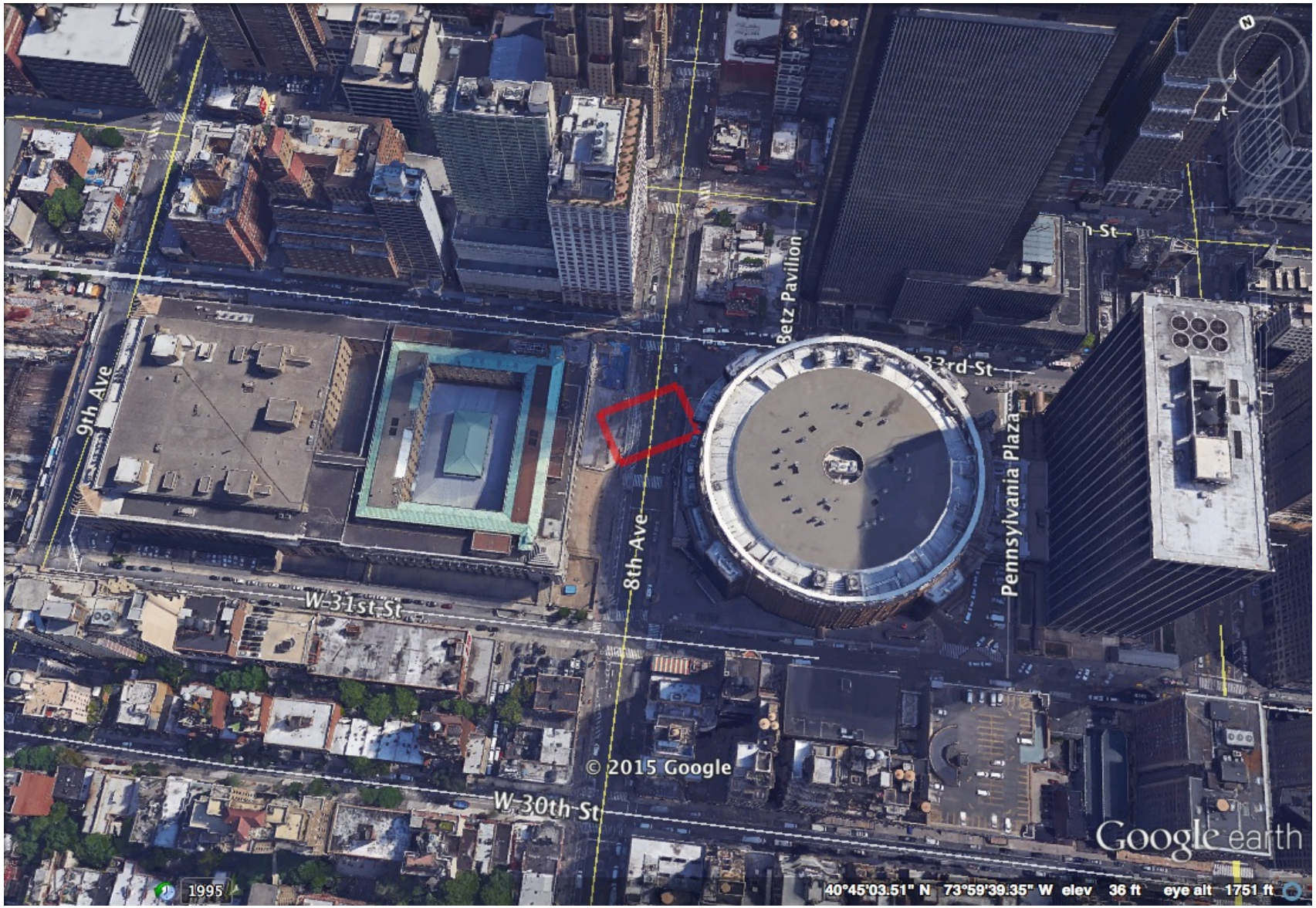
- We can get $[a, b]$ by finding the 2.5% and 97.5% quantiles of $p(y \mid x)$.

The Basic Approach

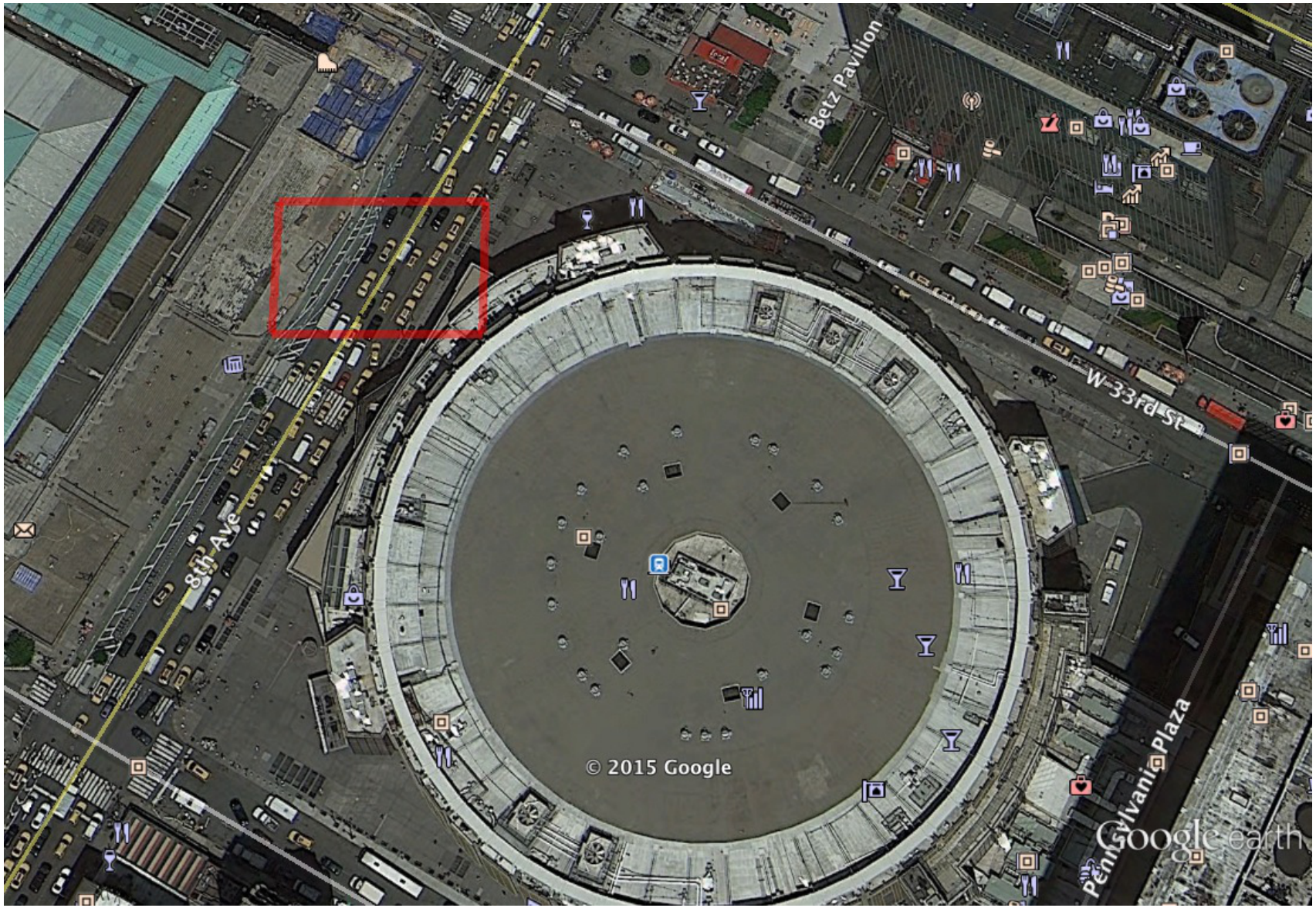
- Raw input is [roughly] continuous in
  - space (lat/lon) and
  - time (seconds since 1970-01-01).
- To make it easier to handle, we partition space and time into buckets.
- Spatial partitioning
  - Divide earth into regularly spaced grid cells.
  - About 400,000 grid cells to cover NYC
- Time partitioning
  - Only consider times at the hour level.
- Aggregate taxi pickup counts at the Grid Cell / Hour level.

Initial data analysis, including aggregation by grid cell and hour, was done by Blake Shaw.

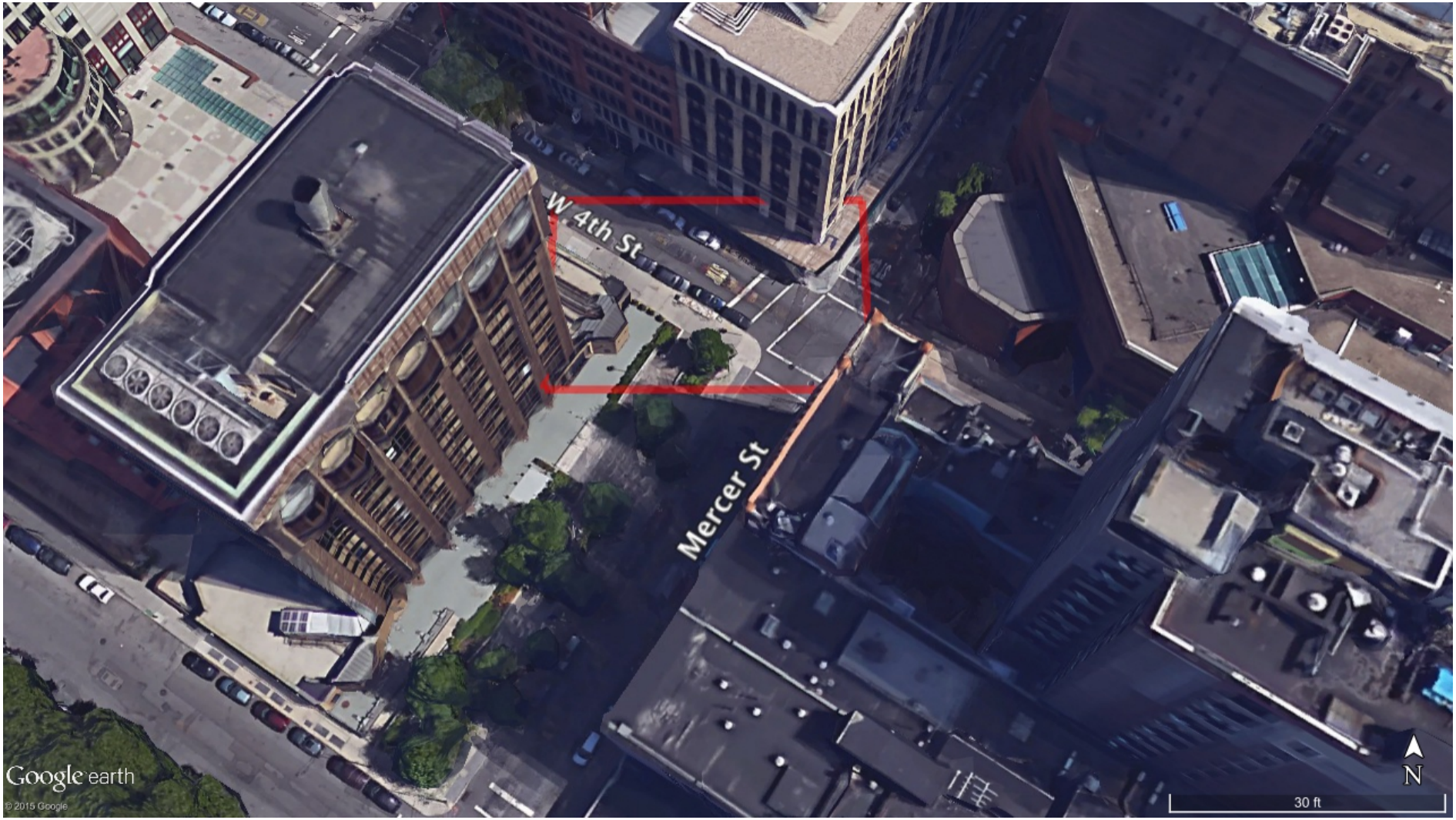
Most Active Grid Cell: Penn Station (Grid ID 7750)



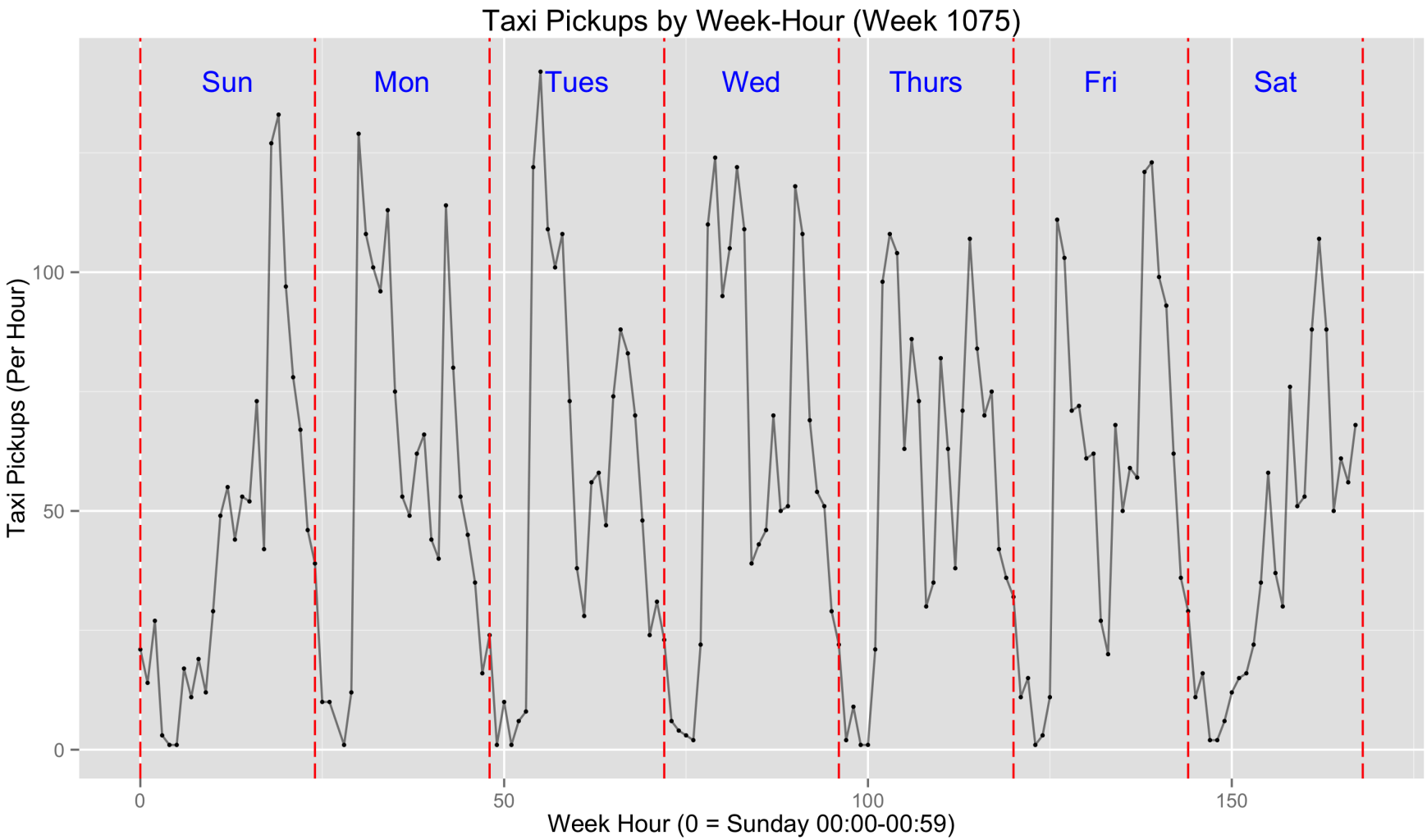
Most Active Grid Cell: Penn Station (Grid ID 7750)



Courant Institute (Grid ID 21272)

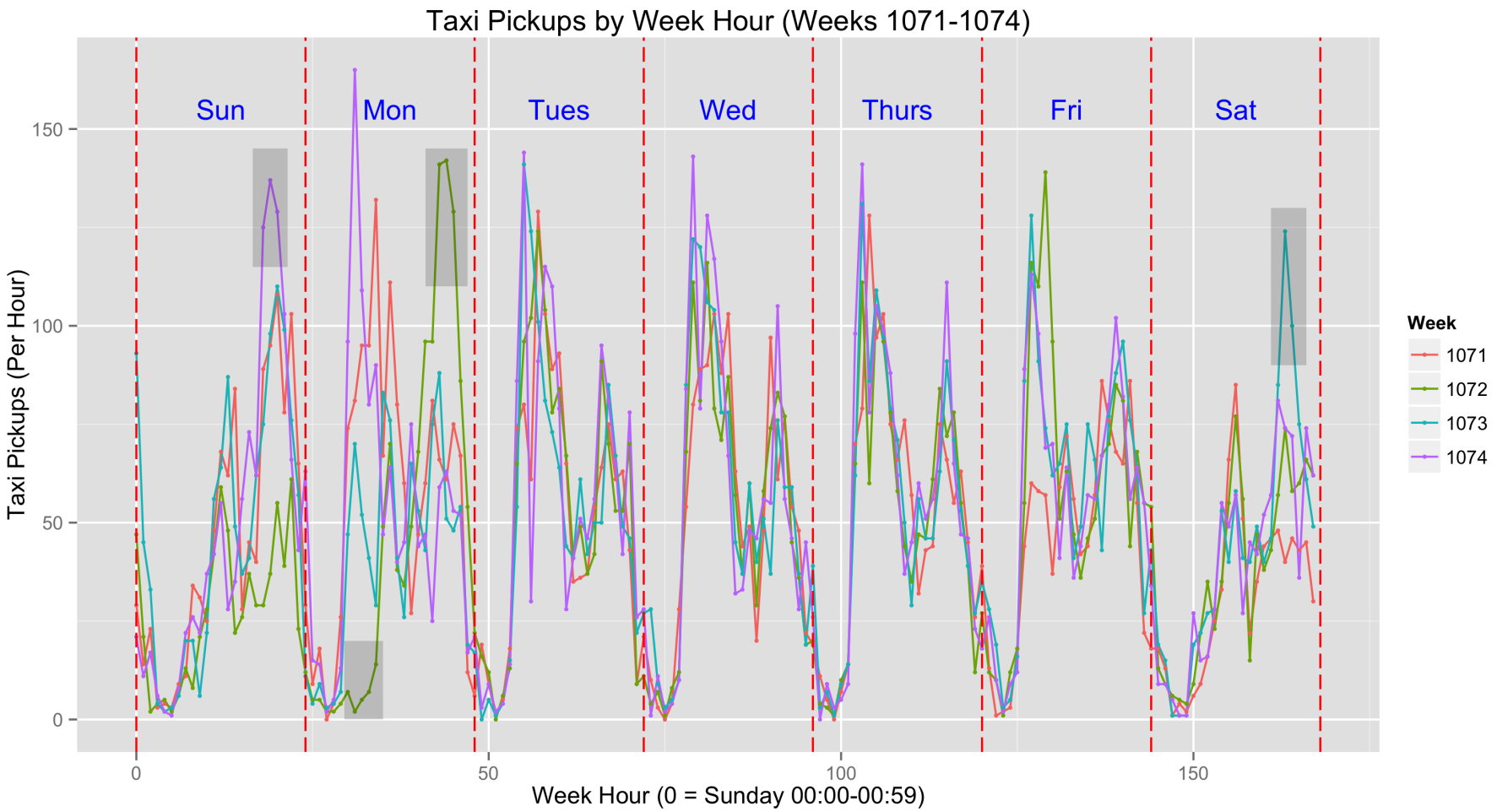


Penn Station (Cell 7750): 1300 Taxi Pickups Per Day



Note difference between weekend and weekday patterns.

Penn Station (Cell 7750): Four Weeks, Some Outliers



Box and Whiskers Plot

- **Box plot** or **box and whiskers** plot concisely summarizes a sample of data.

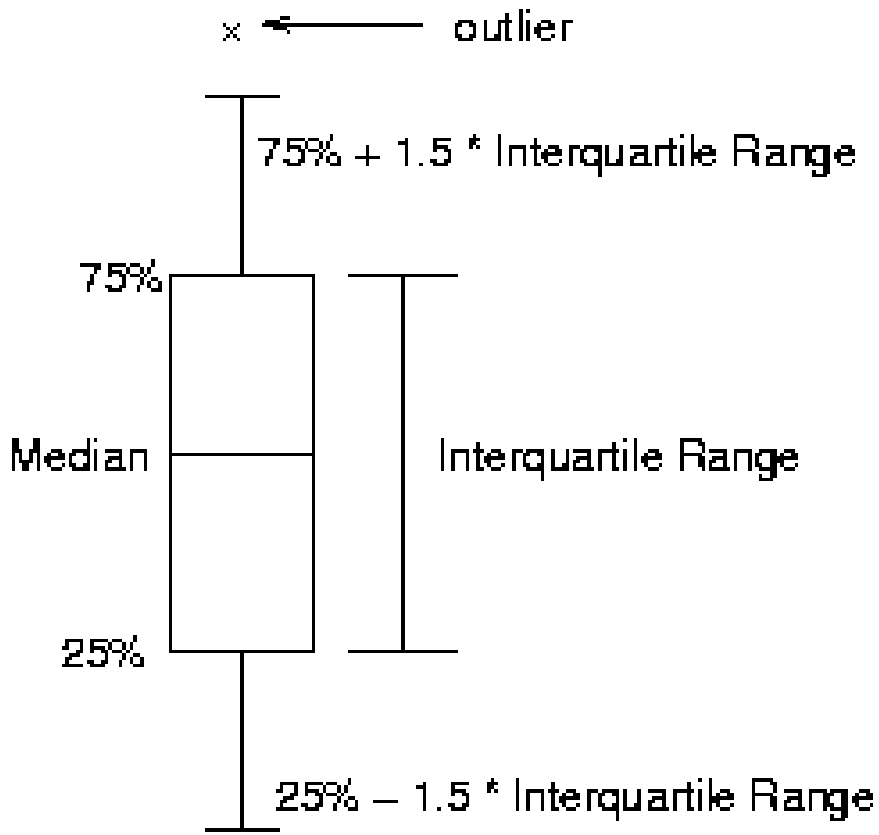
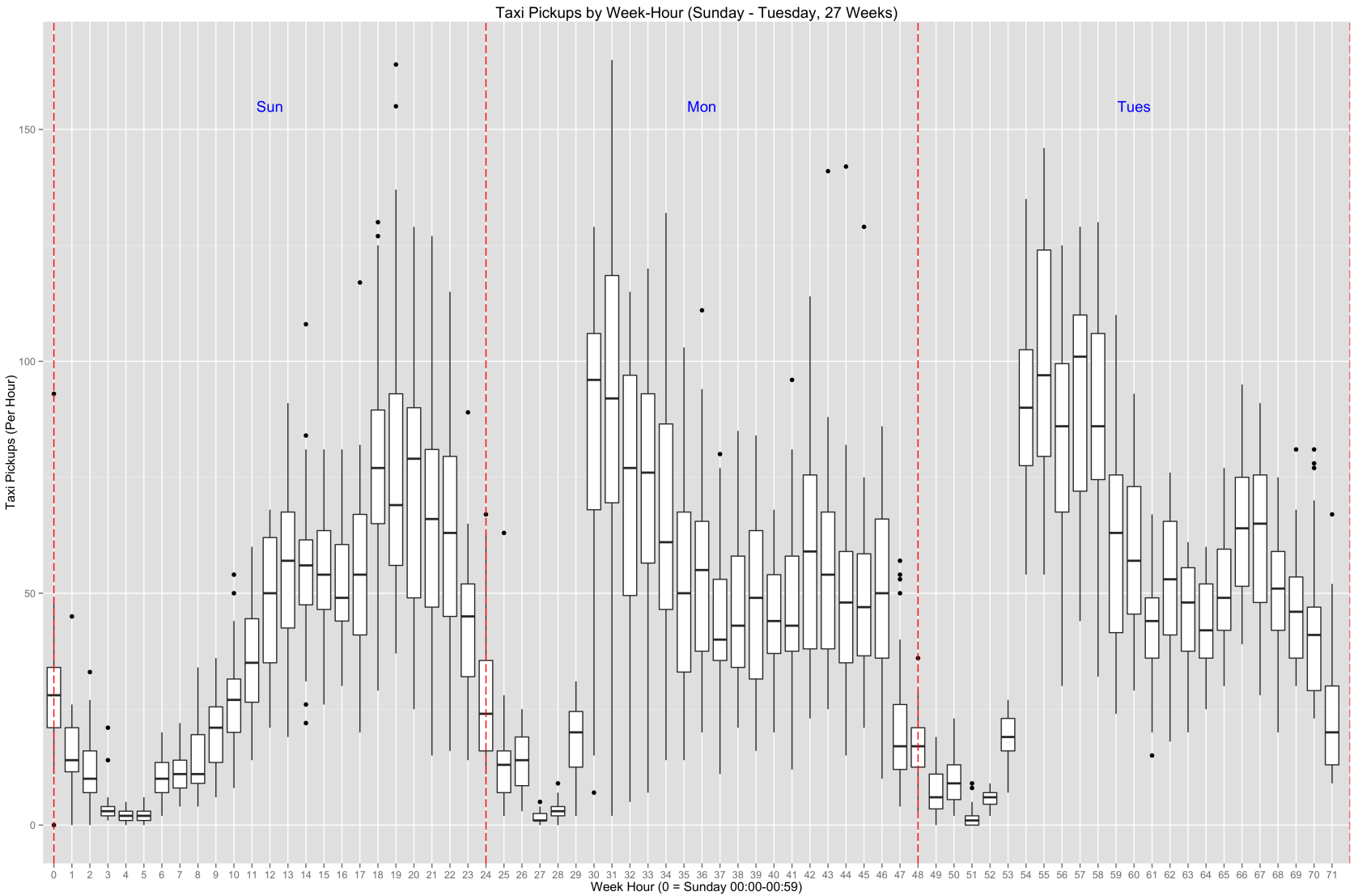
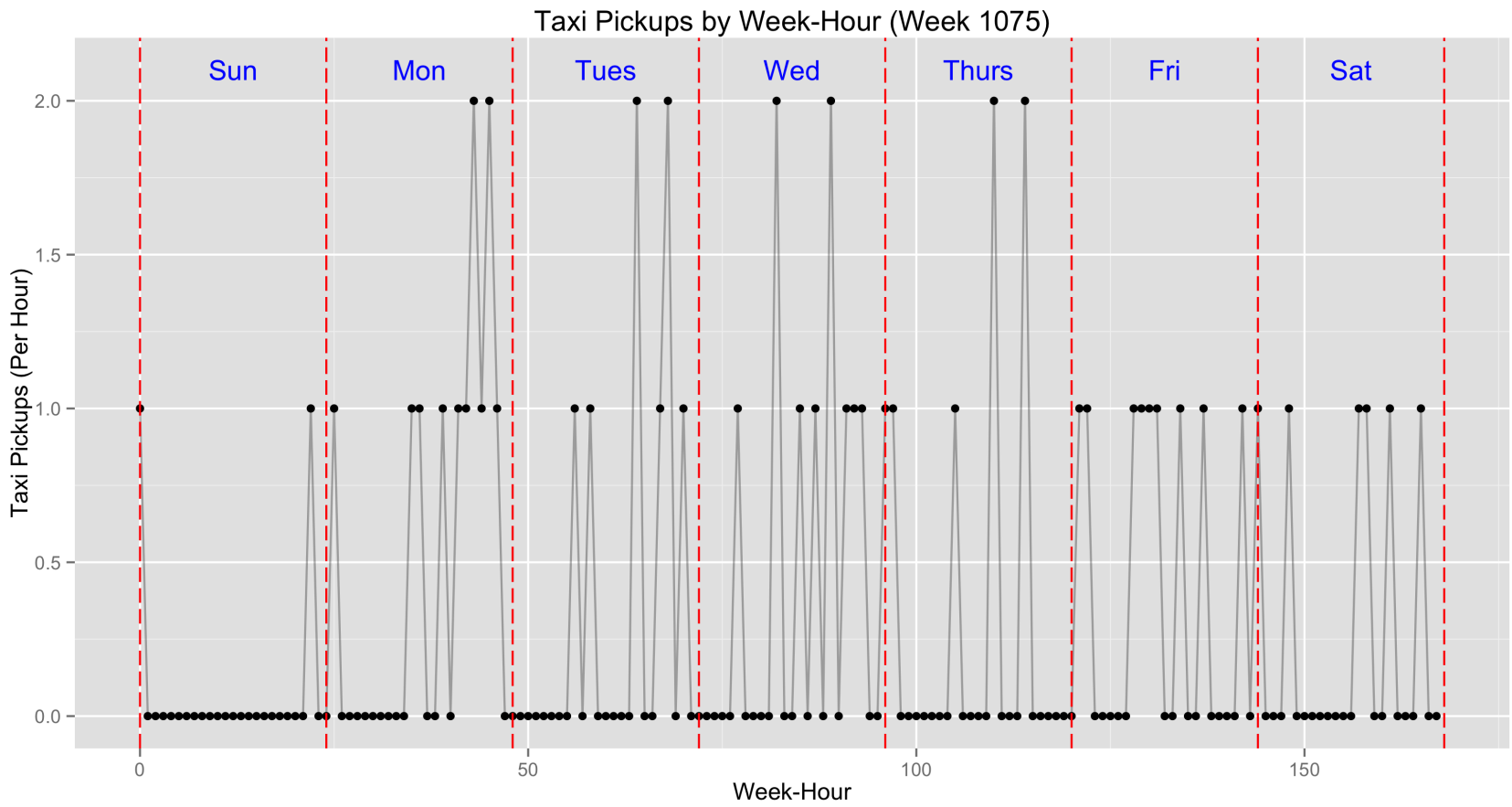


Figure from `http://taps-graph-review.wikispaces.com/Box+and+Whisker+Plots`.

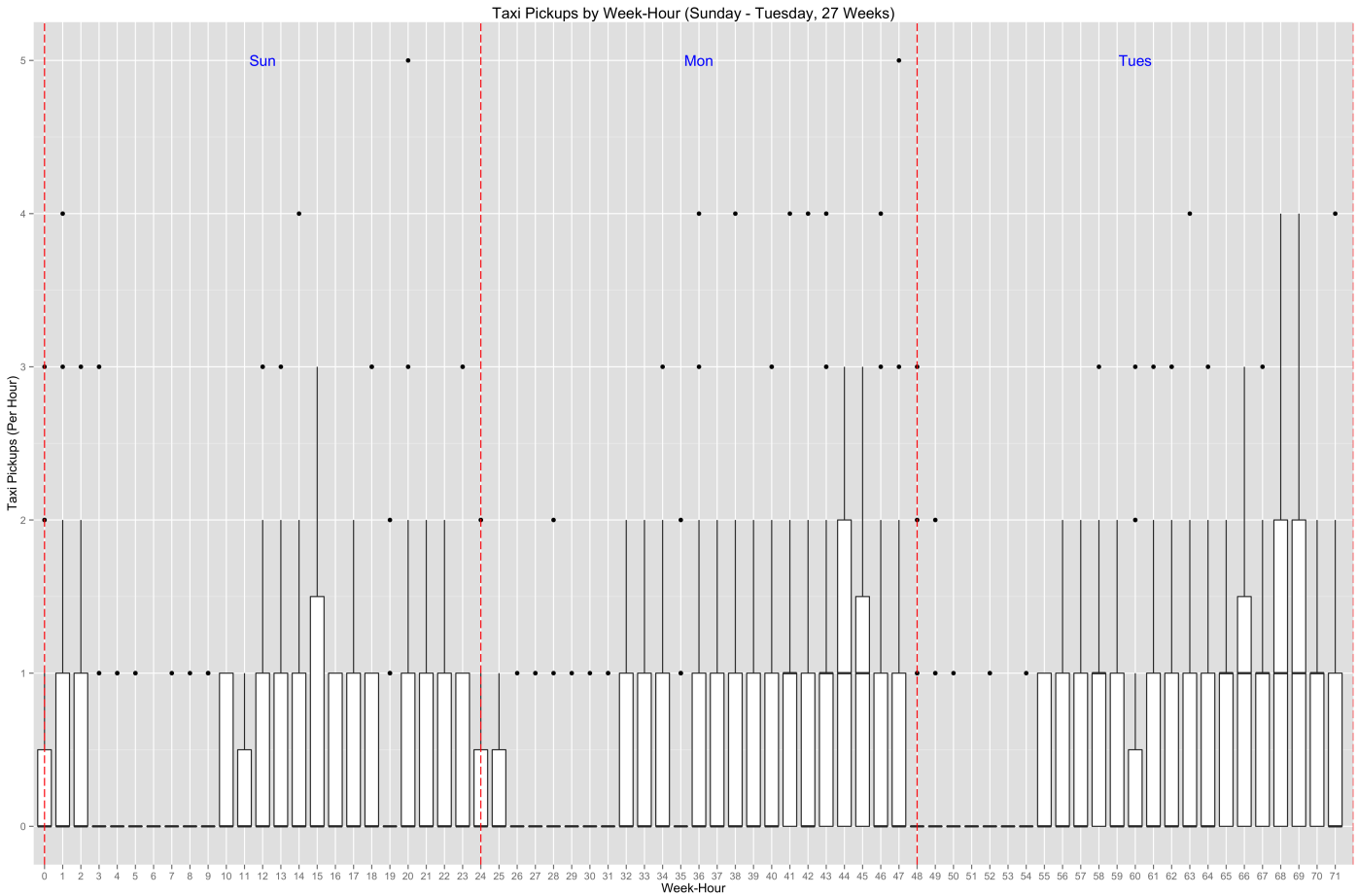
Penn Station: Sunday-Tuesday, 27 Weeks



Courant (Week 1075): 12 Taxi Pickups Per Day



Courant Institute: Sunday-Tuesday, 27 Weeks



Note: At least 25%, sometimes 75%+ of counts are zero.
Box plot clearly shows extreme values (ranging up to 5).

The Prediction Problem

Somebody queries a **grid cell** and a **week-hour**, we tell them what to expect.

- Input space: $\mathcal{X} = \{(g, h) \mid g \in \{1, \ldots, 398245\} \text{ and } h \in \{0, \ldots, 167\}\}$, where
  - g is the grid Cell ID and
  - h is the week-hour
  - Possible future inputs: Holiday? Raining? Special event?
- Action space: $\mathcal{A} = \{\text{Probability distributions on number of pickups}\}$
- Output space: $\mathcal{Y} = \{0, 1, 2, 3, \ldots\}$
  - Actual number of taxi pickups.
- Evaluation? Loss function? We'll come back to these questions...

Setting up the Learning Problem

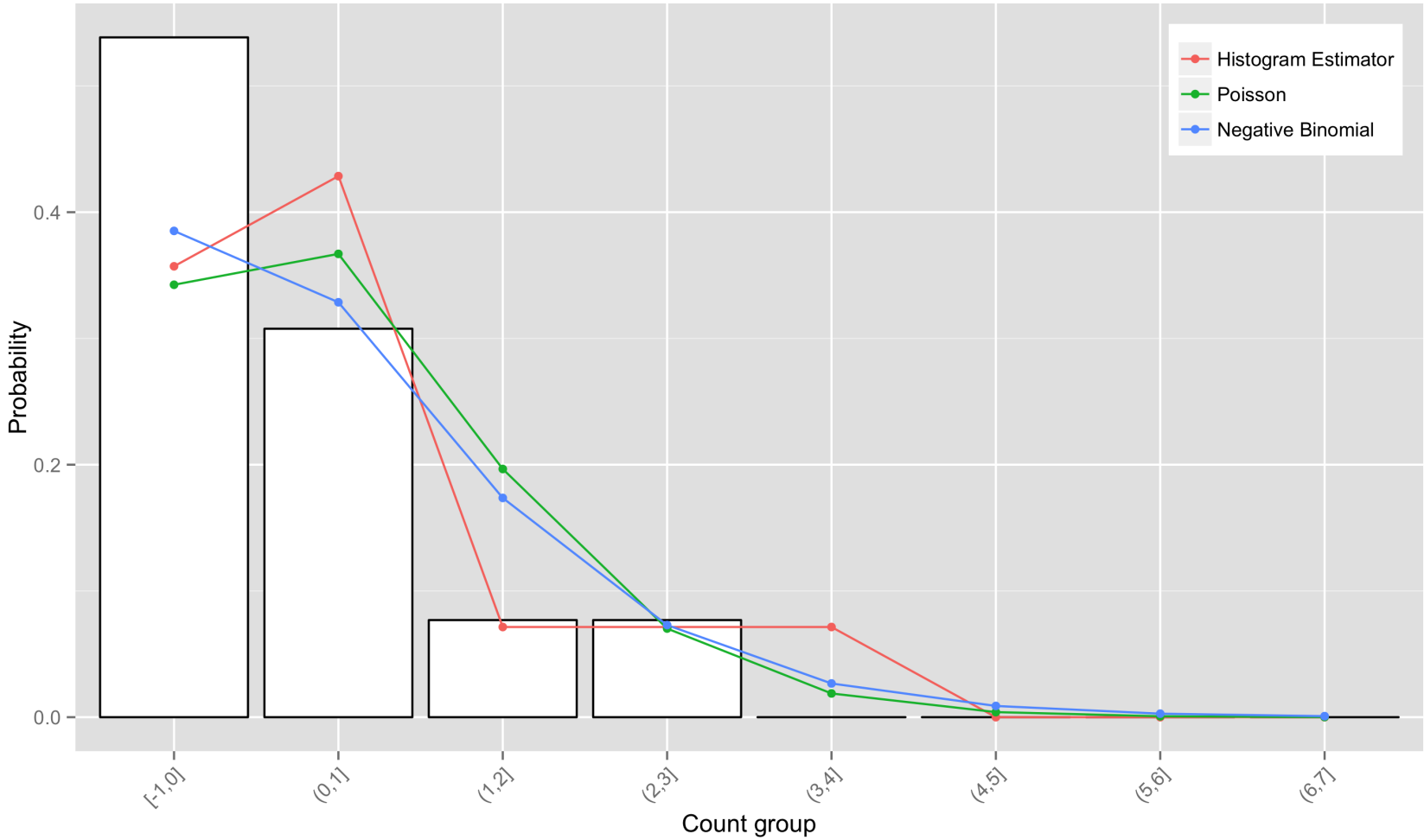
- Labeled data look like:
  - (Grid Cell = 10321, Week Hour = 120) $\mapsto$ Count = 3
  - (Grid Cell = 192001, Week Hour = 6) $\mapsto$ Count = 12
  - (Grid Cell = 1271, Week Hour = 154) $\mapsto$ Count = 0
- How to split the data into a training set and a test set?

Train / Test Splits For Time-Indexed Datasets

- First idea is to randomly split instances into training and test.
- This does not reflect reality at deployment time:
  - Model trained on historical data from before deployment
  - With random split, training examples can occur after test examples.
  - For time series prediction (e.g. stock price prediction),
    - Usually want to predict some amount of time into the future.
    - Training and test data should reflect that application.
- Our approach:
  - First 14 weeks are **training set**.
  - Last 13 weeks are **test set**.

Approach 1: Full Stratification (Courant, Tuesdays 7-8pm)

- Estimate distribution for each grid cell / week hour pair.
- Colored lines are from training. White bars are from test.



Terminology: Stratification and Bucketing

Definition

We say we are **stratifying** if we partition our input space into groups, and treat each group separately. For example, in modeling we would build a separate model for each group, without information sharing across groups.

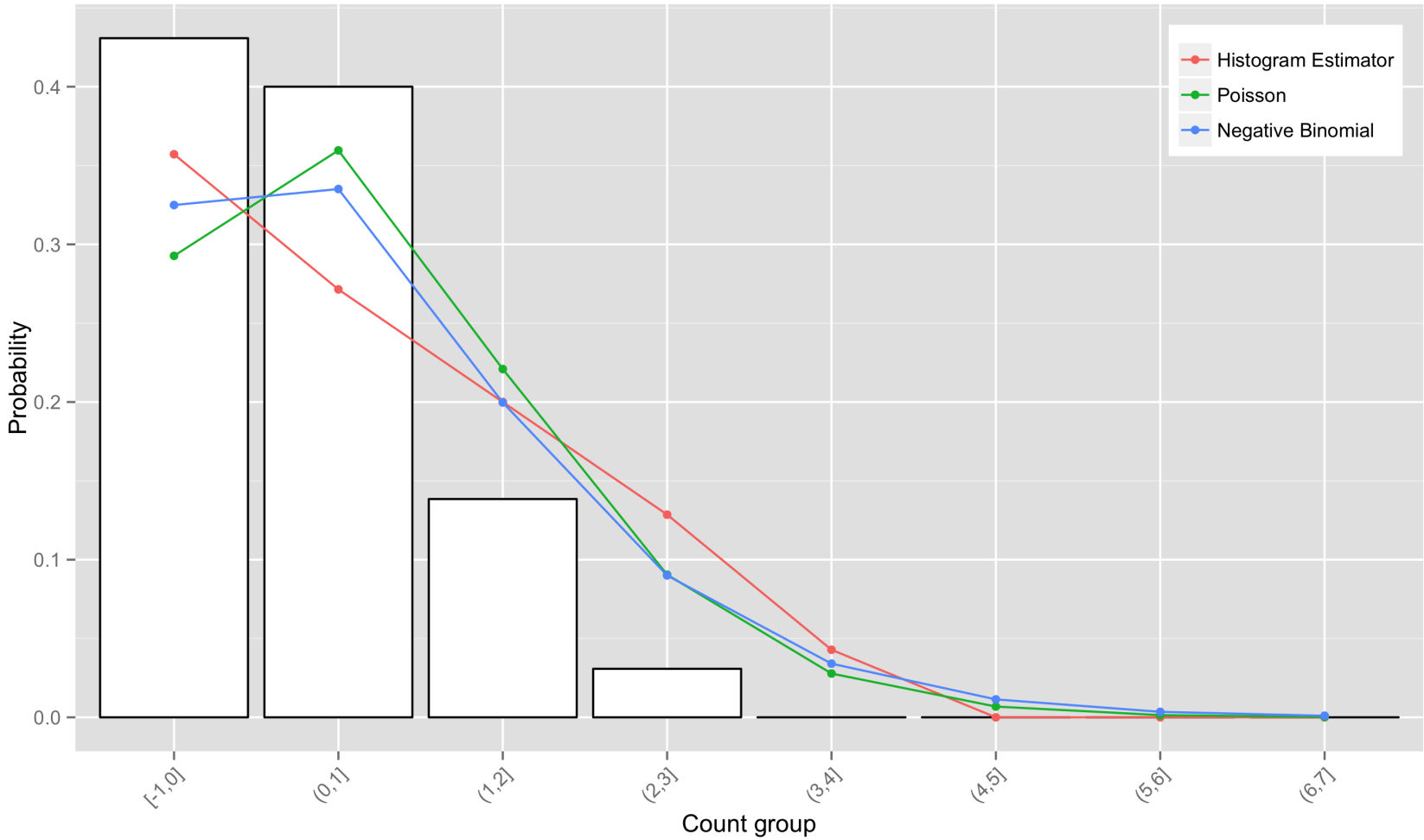
On the other hand,

Definition

We say we are **bucketing** (or **binning**) if we are combining natural groups in the data into a single group, rather than building a separate model for each group. For example, combining all weekdays together would be "bucketing".

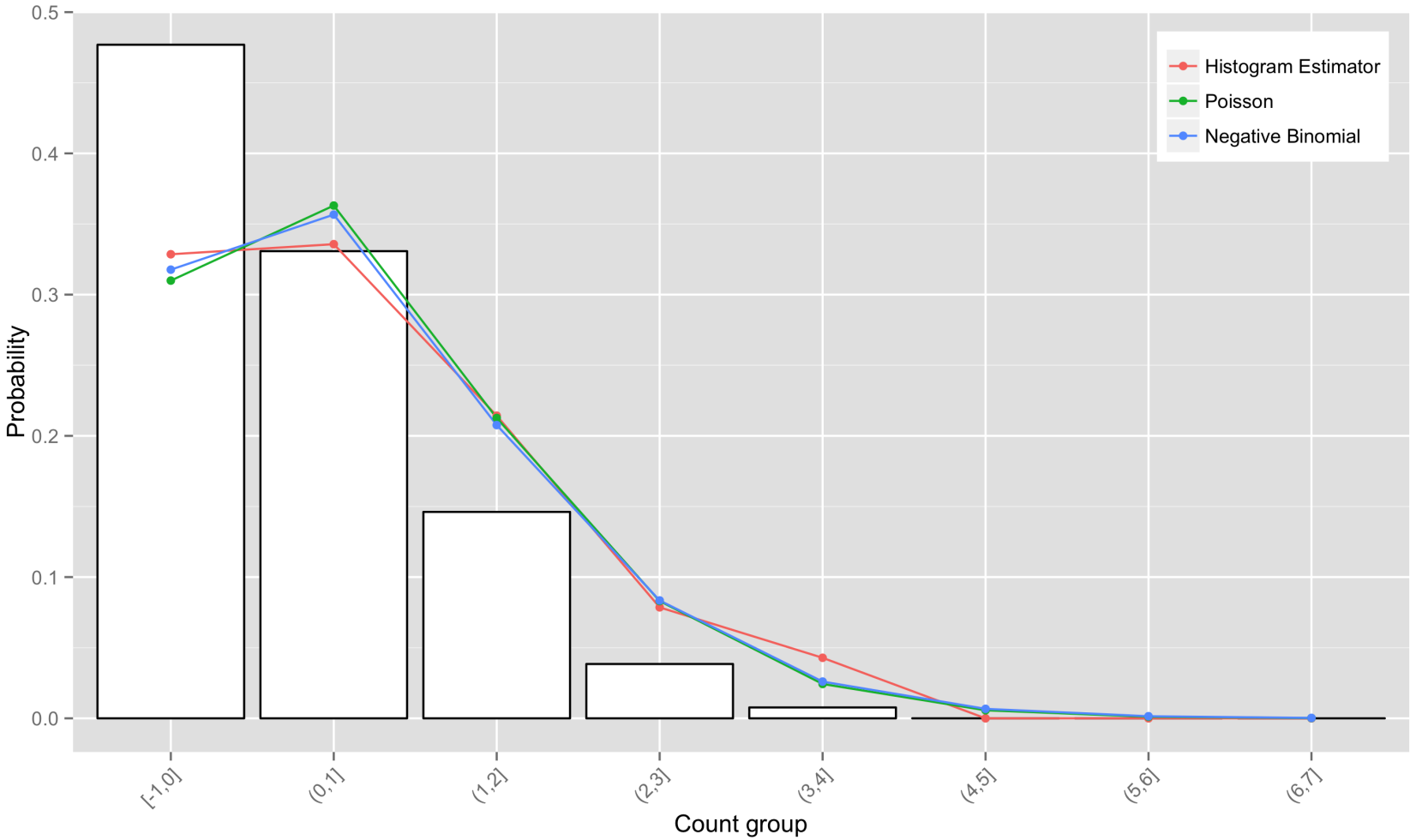
Approach 2: Weekday Bucketing (Courant, M-F 7-8pm)

- Data inspection suggests that day patterns are similar Mon-Fri.



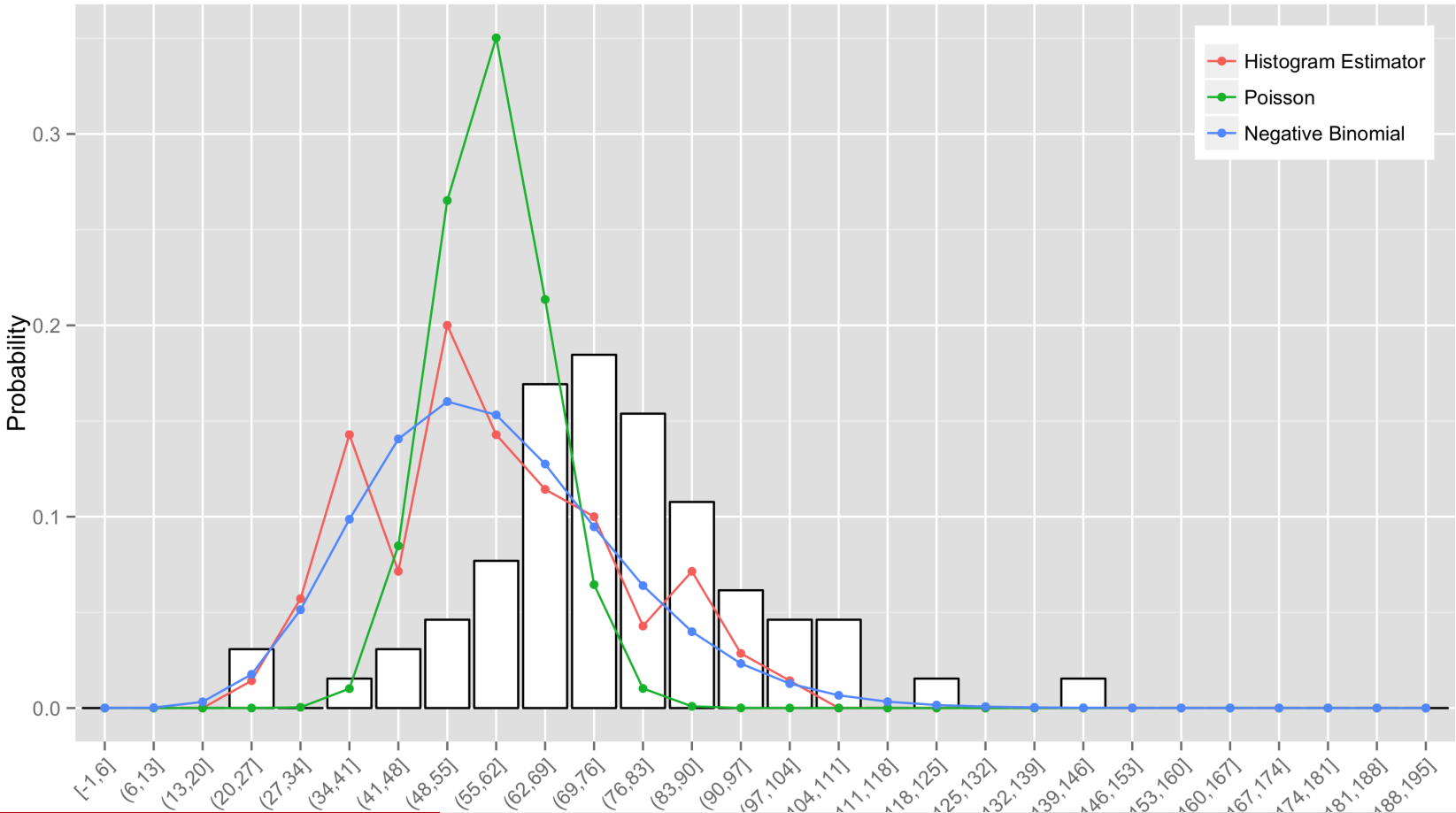
Approach 3: (Courant, M-F 6-8pm)

- Also, 6-7pm looks similar to 7-8pm, so join together



Penn Station, M-F 7-8pm

- Negative binomial fits empirical much better than Poisson. (overdispersion)
- Massive shift between train and test!



The Bias / Variance Tradeoff of Stratification

- With a separate model for every grid cell / week-hour pair, model is highly specific!
- Could capture idiosynchracy of Friday @5pm that we would miss if combining all weekdays.
  - That is, we're minimizing the bias.
- With relatively little data in a particular stratum, estimates will have high variance.
- By "bucketing", or combining stratum:
  - We can reduce variance.
  - It may cost us in bias.
  - By bucketing in a smart way, you can minimize bias increase.

Is there a more convenient way?

- We can tradeoff between bias and variance by varying the stratification and the bucketing.
- It's a great way to start your data analysis.
  - You get a feel for the data and gain some intuition.
- This technique can be used for classification and regression as well.
- Our classification and regression techniques also trade off between bias and variance:
  - We had to choose our features.
  - We had to tune our regularization parameter.
- Can we do something similar for predicting distributions?
- Yes – this is **generalized regression**, where the action space is a distribution rather than a real. A topic for later....