

```
In [27]: import pandas as pd
import altair as alt

url = "https://raw.githubusercontent.com/MarkHershey/CompleteTrumpTweetsArch

tweets = pd.read_csv(
    url,
    encoding="utf-8-sig",
    on_bad_lines="skip"
)
tweets.columns = tweets.columns.str.strip()
#print(tweets.columns)
tweets["Date & Time"] = pd.to_datetime(tweets["Time"], errors="coerce")
tweets = tweets.drop(columns=["ID", "Tweet URL", "Time"])
tweets = tweets.set_index("Date & Time")
tweets.head(10)
```

Out[27]:

Date & Time	Tweet Text
2017-01-20 06:31:00	"It all begins today! I will see you at 11:00...
2017-01-20 11:54:00	"We will bring back our jobs. We will bring b...
2017-01-20 11:55:00	"We will follow two simple rules: BUY AMERICA...
2017-01-20 11:58:00	"It is time to remember that...https://www.fa...
2017-01-20 12:13:00	"TO ALL AMERICANS https://www.facebook.com/Do...
2017-01-21 05:53:00	"A fantastic day and evening in Washington D...
2017-01-22 06:47:00	"Watched protests yesterday but was under the...
2017-01-23 05:38:00	"Busy week planned with a heavy focus on jobs...
2017-01-24 05:11:00	"Will be meeting at 9:00 with top automobile ...
2017-01-24 10:58:00	"A photo delivered yesterday that will be dis...

```
In [28]: daytime = tweets.between_time("08:01", "16:00")
evening = tweets.between_time("16:01", "00:00")
overnight = tweets.between_time("00:01", "08:00")

print(f"The number of tweets in the 'Daytime' category is: {len(daytime)}")
print(f"The number of tweets in the 'Evening' category is: {len(evening)}")
print(f"The number of tweets in the 'Overnight' category is: {len(overnight)}")

print("Just by the distribution of tweets it is surprisingly evenly spread;
```

The number of tweets in the 'Daytime' category is: 3325
 The number of tweets in the 'Evening' category is: 3653
 The number of tweets in the 'Overnight' category is: 3706
 Just by the distribution of tweets it is surprisingly evenly spread; with the most tweets happening overnight.

```
In [29]: tw = pd.Series(tweets.index.strftime('%m-%d'), index=tweets.index)      # Cha

spring = tweets.loc[tw.between('04-01', '06-30')]      # April
summer = tweets.loc[tw.between('07-01', '09-30')]      # July 1
autumn = tweets.loc[tw.between('10-01', '12-31')]      # October
winter = tweets.loc[tw.between('01-01', '03-31')]      # January

print(f"The number of tweets in the 'Spring' category is: {len(spring)}")
print(f"The number of tweets in the 'Summer' category is: {len(summer)}")
print(f"The number of tweets in the 'Autumn' category is: {len(autumn)}")
print(f"The number of tweets in the 'Winter' category is: {len(winter)}")
print("")
print("Judging by the distribution of tweets throughout the seasons, the fre
```

The number of tweets in the 'Spring' category is: 2520
 The number of tweets in the 'Summer' category is: 3148
 The number of tweets in the 'Autumn' category is: 3067
 The number of tweets in the 'Winter' category is: 1949

Judging by the distribution of tweets throughout the seasons, the frequency decays in the winter and spring and peaks in the summer

```
In [30]: tweet_time = pd.DataFrame({
    "Time": ["daytime", "evening", "overnight"],
    "Count": [len(daytime), len(evening), len(overnight)]
})

tweet_time
```

```
Out[30]:
```

	Time	Count
0	daytime	3325
1	evening	3653
2	overnight	3706

```
In [31]: tweet_season = pd.DataFrame({
    "Season": ["spring", "summer", "autumn", "winter"],
    "Count": [len(spring), len(summer), len(autumn), len(winter)]
})

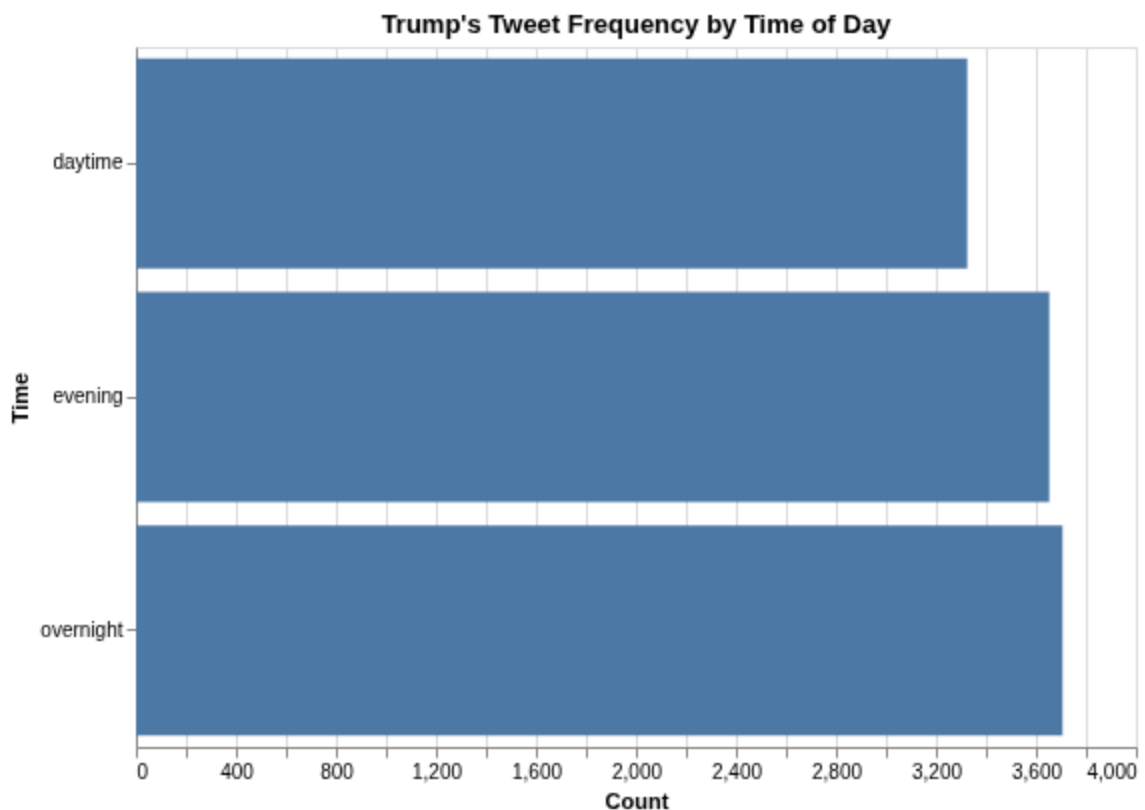
tweet_season
```

Out[31]:

	Season	Count
0	spring	2520
1	summer	3148
2	autumn	3067
3	winter	1949

```
In [120... timeBars = alt.Chart(tweet_time).mark_bar().encode(  
    y = "Time:N",  
    x = "Count:Q",  
).properties(  
    title="Trump's Tweet Frequency by Time of Day",  
    width=500,  
    height=350)  
  
timeBars
```

Out[120...



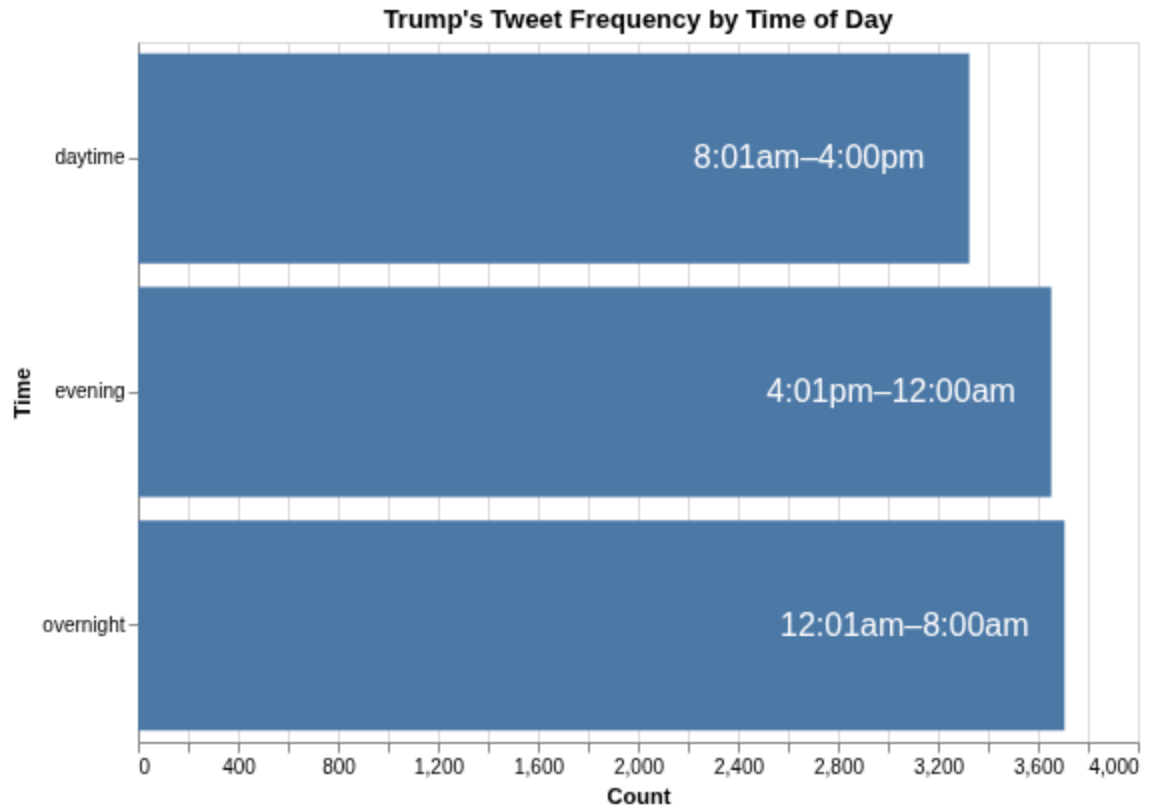
```
In [121... range_text = (  
    alt.Chart(label_df)  
    .transform_calculate(mid="datum.Count / 2")  
    .mark_text(  
        align="center",  
        baseline="middle",  
        color="white",  
        fontSize=16,  
        dx=-80  
    )  
)
```

```

        .encode(
            y="Time:N",
            x="Count:Q",
            text="range:N"
        )
    )
timeBars + rangeText

```

Out[121]...

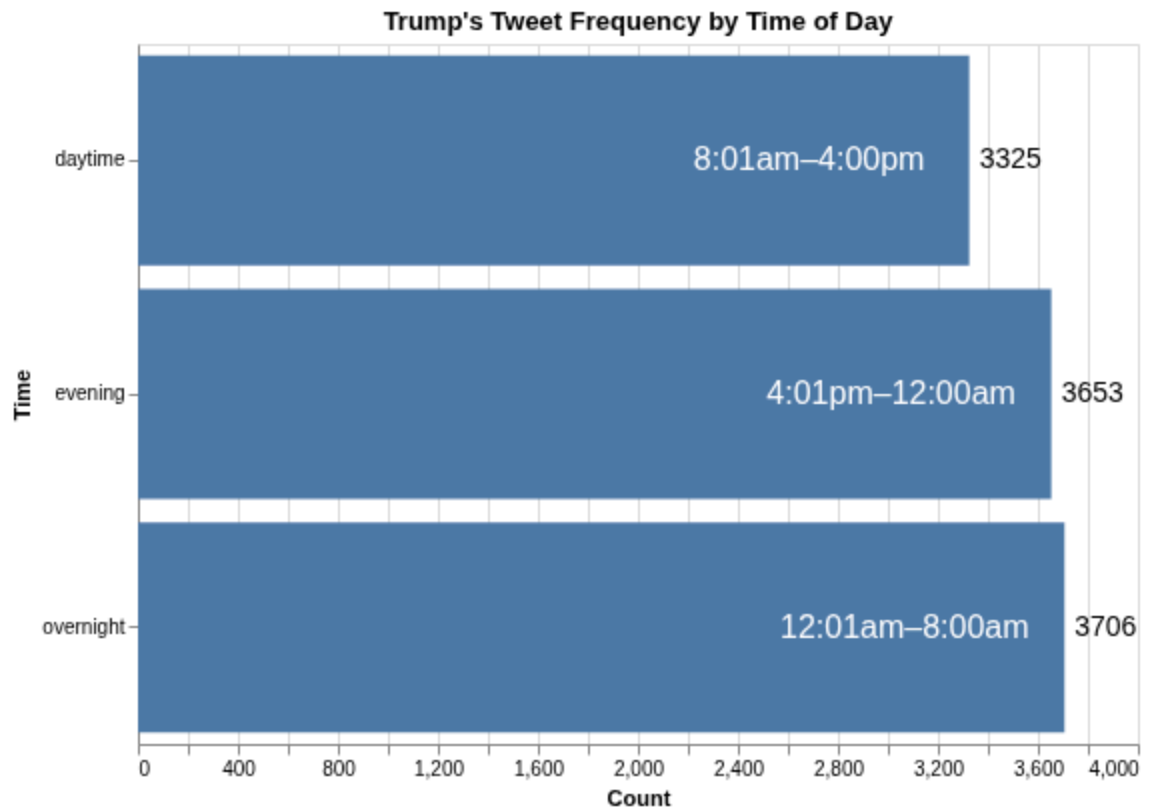


```

In [122]... countText = alt.Chart(labelDf).markText(
    align="left",
    baseline="middle",
    dx=5,
    color="black",
    fontSize=14
).encode(
    y="Time:N",
    x="Count:Q",
    text="Count:Q"
)
timeBars + rangeText + countText

```

Out[122...



```
In [124... final_chart = (time_bars + range_text + count_text).properties(padding={"ri...  
final_chart
```

Out[124...

