Call Center Data Analysis

August 9, 2023

[43]:
```
import pandas as pd
import numpy as np
import random
import scipy
from scipy.stats import chi2_contingency
import sorted_months_weekdays
import sort_dataframeby_monthorweek
import seaborn as sns
import matplotlib.pyplot as plt
from prettytable import PrettyTable
from sklearn.linear_model import LinearRegression
%matplotlib inline
```

0.1 Analyzing the data from the Sleuth Goose call center data

[44]:
```
#read in the data

df = pd.read_csv('SleuthGoose_Call_Data.csv')
df.head(2)
```

[44]:

	Unnamed: 0	date	time	call_id	agent_id	answer_time_minute	answer_time_second	talk_time_minute	talk_time_second	escalation	abandoned_num	abandoned	csat	csat_value
0	1	1/9/2022	4:53 PM	2186	212	1	43	26	54	False	0	False	False	NaN
1	1	1/9/2022	2:09 PM	3563	211	2	43	51	4	False	0	False	True	0.0

[46]:
```
# copy of the datafile to remove all calls with wait times under 5 minutes
df1 = df.copy()

#creating binary column to indicate call wait times over 5 minutes and one for␣
 ↪escalations
#I just felt it would be easier to count these
```

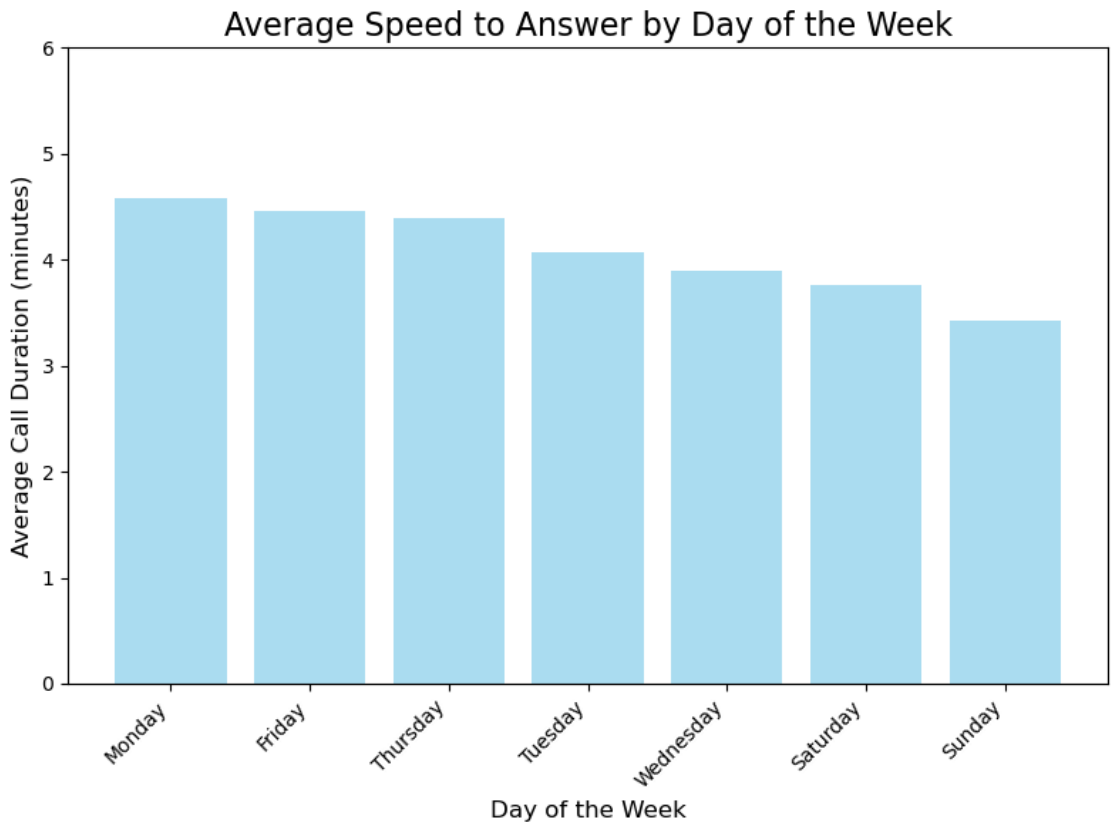
```
df1['over_five'] = np.where(df1['answer_time_minute'] < 5, 0, 1)
df1['escalation_num'] = np.where(df1['escalation'] ==False, 0, 1)


#create day_name column from the date
df1['date'] = pd.to_datetime(df1['date'])
df1['day_name'] = df1['date'].dt.day_name()


#create new copy of datafile
df2 = df1.copy()
df2.tail()
```

[46]:

	Unnamed: 0	date	time	call_id	agent_id	answer_time_minute	answer_time_second	talk_time_minute	talk_time_second	escalation	abandoned_num	abandoned	csat	csat_value	over_five	escalation_num	day_name
4611	7	2022-09-24	1:18 PM	2966	216	6	59	0	27	False	0	False	False	NaN	1	0	Saturday
4612	7	2022-09-24	6:36 PM	4971	210	3	31	53	52	False	0	False	True	0.0	0	0	Saturday
4613	7	2022-09-24	2:46 PM	2967	213	8	41	14	5	False	0	False	False	NaN	1	0	Saturday
4614	7	2022-09-24	8:19 AM	2969	214	2	54	54	17	False	0	False	False	NaN	0	0	Saturday
4615	7	2022-09-24	5:12 PM	4970	214	2	14	19	53	False	0	False	True	0.0	0	0	Saturday

[39]:
```
# Calculate the total call duration (in minutes) for each call and take the␣
 ↪average by day of the week
df1['total_answer_time'] = (df1['answer_time_minute'] * 60 +␣
 ↪df1['answer_time_second']) / 60
average_duration_by_day = df1.groupby('day_name')['total_answer_time'].mean()
```

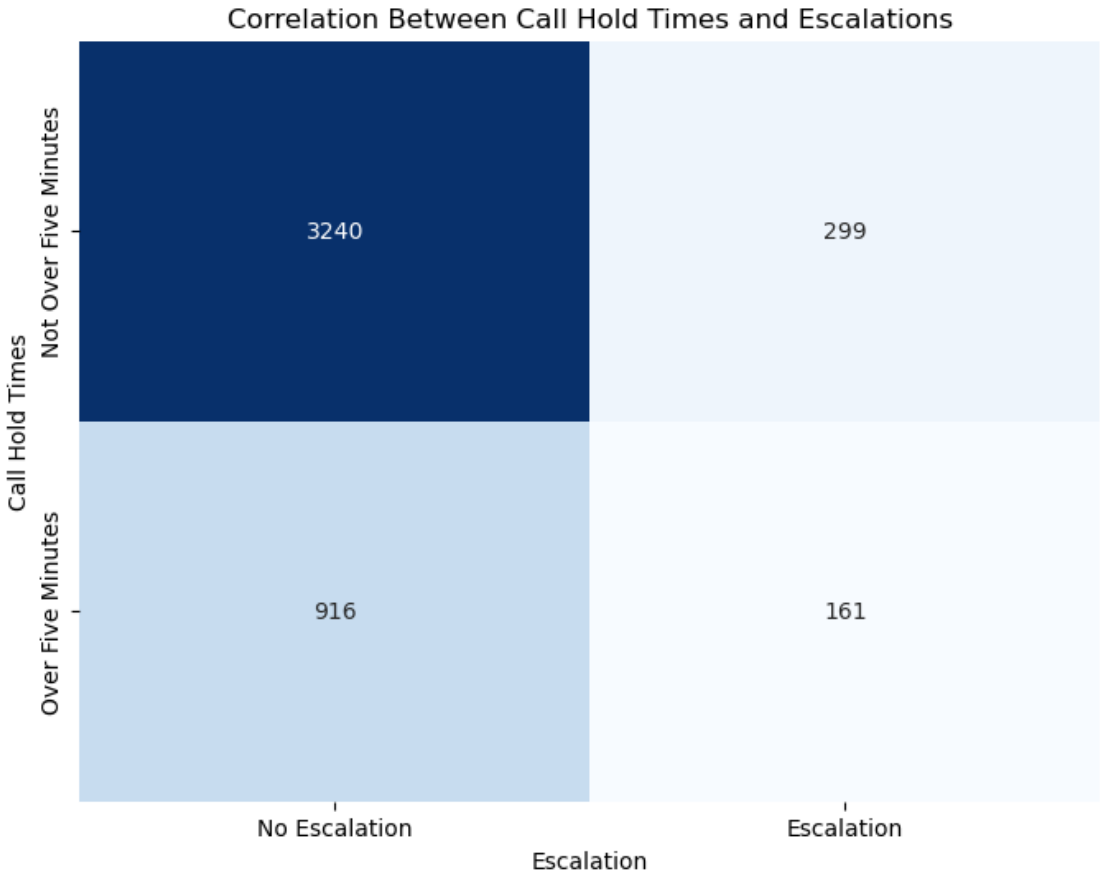
```
# Order the days of the week based on the average call duration in descending␣
 ↪order
ordered_days = average_duration_by_day.sort_values(ascending=False).index

# Plot the ordered days (optional)
plt.figure(figsize=(8, 6))
plt.bar(ordered_days, average_duration_by_day[ordered_days], color='skyblue',␣
 ↪alpha=0.7)
plt.xlabel('Day of the Week', fontsize=12)
plt.ylabel('Average Call Duration (minutes)', fontsize=12)
plt.title('Average Speed to Answer by Day of the Week', fontsize=16)
plt.ylim(0, 6)  # Set the y-axis limit to 6

plt.xticks(rotation=45, ha='right', fontsize=10)
plt.tight_layout()
plt.show()

# Print the ordered days with their corresponding total call duration
print('AVERAGE SPEED TO ANSWER IN MINUTES BY DAY')
for day in ordered_days:
    print(f"{day}: {average_duration_by_day[day]} minutes")
```



```
AVERAGE SPEED TO ANSWER IN MINUTES BY DAY
Monday: 4.582945736434109 minutes
Friday: 4.459156378600823 minutes
Thursday: 4.393874007936508 minutes
Tuesday: 4.068227593152065 minutes
Wednesday: 3.8897260273972605 minutes
Saturday: 3.7647073052733426 minutes
Sunday: 3.423613271124935 minutes
```

[40]:
```python
# Calculate the percentage of calls with escalations
escalations = df2['escalation_num'].sum()
percent_escalations = (escalations / calls) * 100

print("Total Escalations: " + str(escalations) + "   Percentage of Escalations:␣
 ↪" + "{:.2f}".format(percent_escalations) + "%")

# Create a pivot table to analyze the relationship between over_five and␣
 ↪escalations
pivot_table = df2.pivot_table(index='over_five', columns='escalation_num',␣
 ↪values='call_id', aggfunc='count', fill_value=0)
pivot_table.rename(index={0: 'Not Over Five Minutes', 1: 'Over Five Minutes'},␣
 ↪columns={0: 'No Escalation', 1: 'Escalation'}, inplace=True)

# Calculate the percentage of escalations for calls over five minutes and calls␣
 ↪under five minutes
percent_escalations_over_five = (pivot_table.loc['Over Five Minutes',␣
 ↪'Escalation'] / pivot_table.loc['Over Five Minutes'].sum()) * 100
percent_escalations_under_five = (pivot_table.loc['Not Over Five Minutes',␣
 ↪'Escalation'] / pivot_table.loc['Not Over Five Minutes'].sum()) * 100

print("Percentage of Escalations for Calls Over Five Minutes: " + "{:.2f}".
 ↪format(percent_escalations_over_five) + "%")
print("Percentage of Escalations for Calls Under Five Minutes: " + "{:.2f}".
 ↪format(percent_escalations_under_five) + "%")

# Visualize the correlation between over_five and escalations
plt.figure(figsize=(8, 6))
sns.heatmap(pivot_table, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.title('Correlation Between Call Hold Times and Escalations')
plt.xlabel('Escalation')
plt.ylabel('Call Hold Times')
plt.show()
```

```
Total Escalations: 460   Percentage of Escalations: 9.97%
Percentage of Escalations for Calls Over Five Minutes: 14.95%
Percentage of Escalations for Calls Under Five Minutes: 8.45%
```



[41]:
```
# Contingency table for the chi-square test
contingency_table = pd.crosstab(df2['over_five'], df2['escalation_num'])

# Chi-square test
chi2, p_value, dof, expected = chi2_contingency(contingency_table)

print("Chi-square value:", chi2)
print("p-value:", p_value)
print("Degrees of freedom:", dof)
print("Expected frequencies table:")
print(pd.DataFrame(expected, index=['Not Over Five Minutes', 'Over Five␣
 ↪Minutes'], columns=['No Escalation', 'Escalation']))
```

```
Chi-square value: 38.164111759703005
p-value: 6.503767274372569e-10
Degrees of freedom: 1
Expected frequencies table:
                       No Escalation  Escalation
Not Over Five Minutes     3186.32669   352.67331
Over Five Minutes          969.67331   107.32669
```

[42]:
```
table = PrettyTable()

table.field_names = ["Test", "Value"]

table.add_row(["Chi-Square", round(chi2, 3)])
table.add_row(["P-value", round(p_value, 5)])
table.add_row(["Degrees of Freedom", dof])

print("\n### Results of chi-square test for escalations vs hold times\n")

print(table)

print("\n*p < 0.05 indicates statistical significance*")
```

```
### Results of chi-square test for escalations vs hold times

+--------------------+--------+
|        Test        | Value  |
+--------------------+--------+
|     Chi-Square     | 38.164 |
|      P-value       |  0.0   |
| Degrees of Freedom |   1    |
+--------------------+--------+

*p < 0.05 indicates statistical significance*
```