DTSC 2301 Spring 2025 Homework #2

Turn in your assignment via Gradescope

Due 1/24/25, 11:59pm

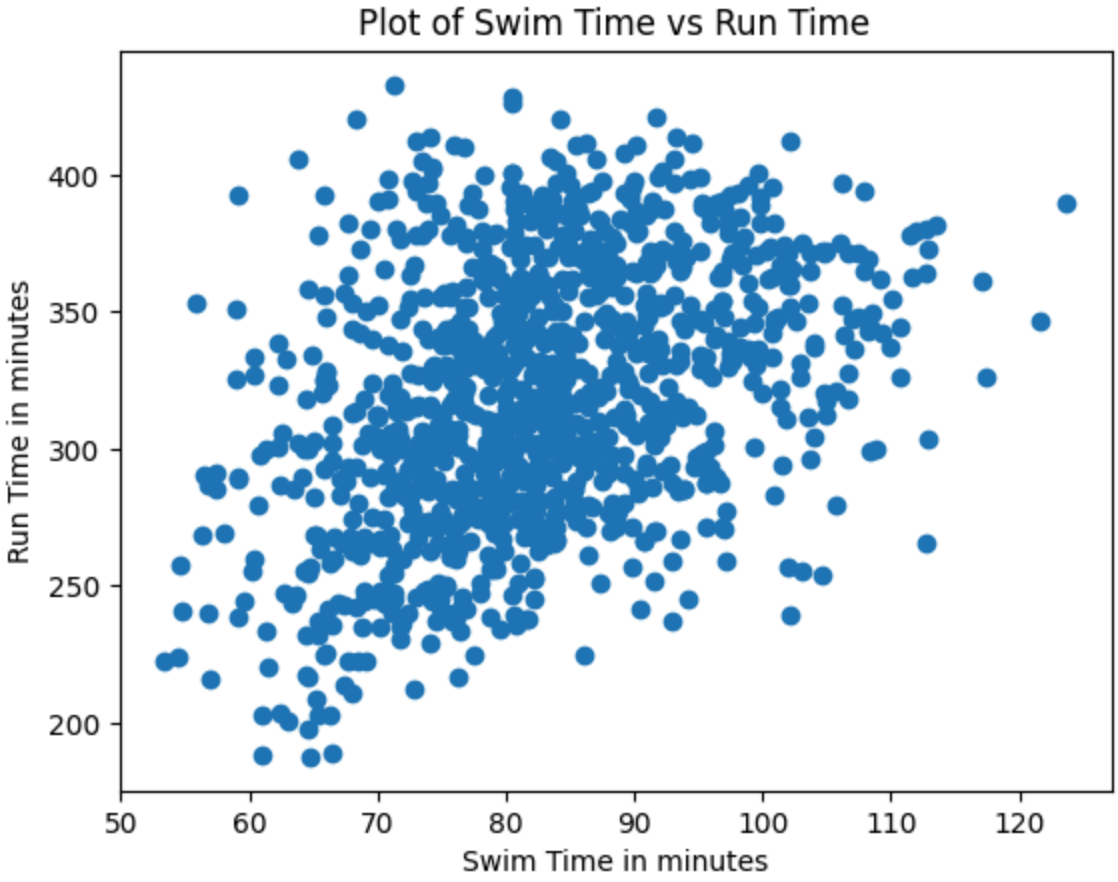
For this assignment you may *not* use any generative AI and you may only use python commands and code we used in class.

Question 1

In the file `https://webpages.charlotte.edu/mschuck1/classes/DTSC2301/Data/Ironman1819.csv', there is data on female finishers of the 2018 and 2019 Ironman Triathlon in Lake Placid, NY. We will focus on the times to complete the three elements of the triathlon, swimming (Swim.Time), biking (Bike.Time) and running (Run.Time). The units for each of these features is minutes. The overall time (including transitions from on element to the next) is found in *Overall.Time*. Create a scatterplot for *Swim.Time* as a predictor for *Run.Time*. As a note it is generally not a good idea in python to make feature/variables have a '.' in them.

More on these data as well as a video introduction to them can be found at: https://isle.stat.cmu.edu/SCORE/ironman_triathlon/

In [21]:
```
import pandas as pd
import matplotlib.pyplot as plt
female_ironman =  pd.read_csv("https://webpages.charlotte.edu/mschuck1/classes/DTSC
female_ironman.dropna(inplace=True)
female_ironman.head()
plt.scatter( female_ironman['Swim.Time'],female_ironman['Run.Time'])

plt.xlabel('Swim Time in minutes')
plt.ylabel('Run Time in minutes')
plt.title('Plot of Swim Time vs Run Time')

plt.show()
```



Question 2

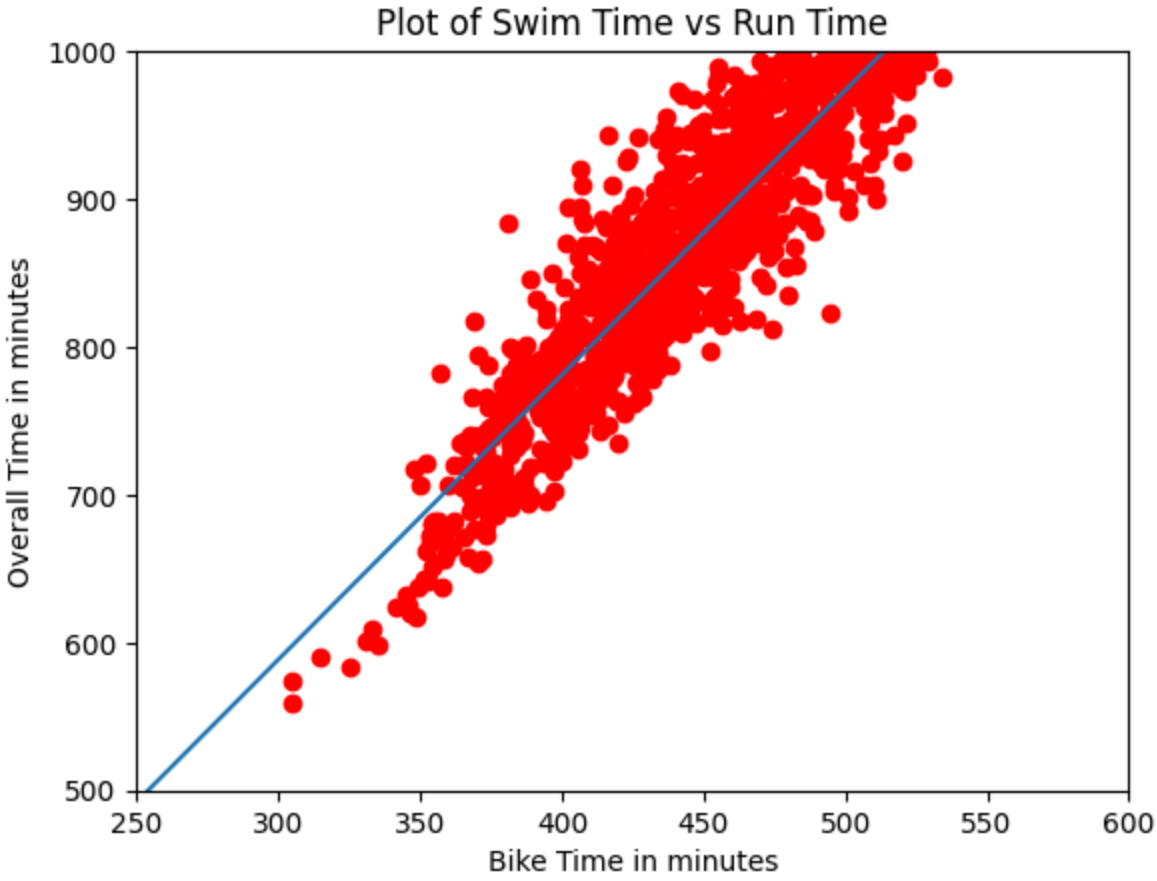
Using the Ironman data from above, fit a regression model that predicts *Overall.Time* from *Bike.Time* and add that line to a scatterplot of those two variables. Interpret the slope and the y-intercept for these data in the context of these data.

In [22]:
```
import numpy as np

y = female_ironman['Overall.Time']

x = female_ironman['Bike.Time']

beta_1, beta_0 = np.polyfit(x, y, deg=1)

print(beta_0)
print(beta_1)
```

```
11.203884734248554
1.92478733704215
```

In [23]:
```
from matplotlib import colors
plt.scatter( female_ironman['Bike.Time'],female_ironman['Overall.Time'], color='red

plt.xlabel('Bike Time in minutes')
plt.ylabel('Overall Time in minutes')
plt.title('Plot of Swim Time vs Run Time')
```

```
xseq = np.linspace(200, 700, num=200)

plt.plot(xseq, 11.2039 + 1.9248*xseq)


plt.xlim(250, 600)
plt.ylim(500,1000)

plt.show()
```



Question 3

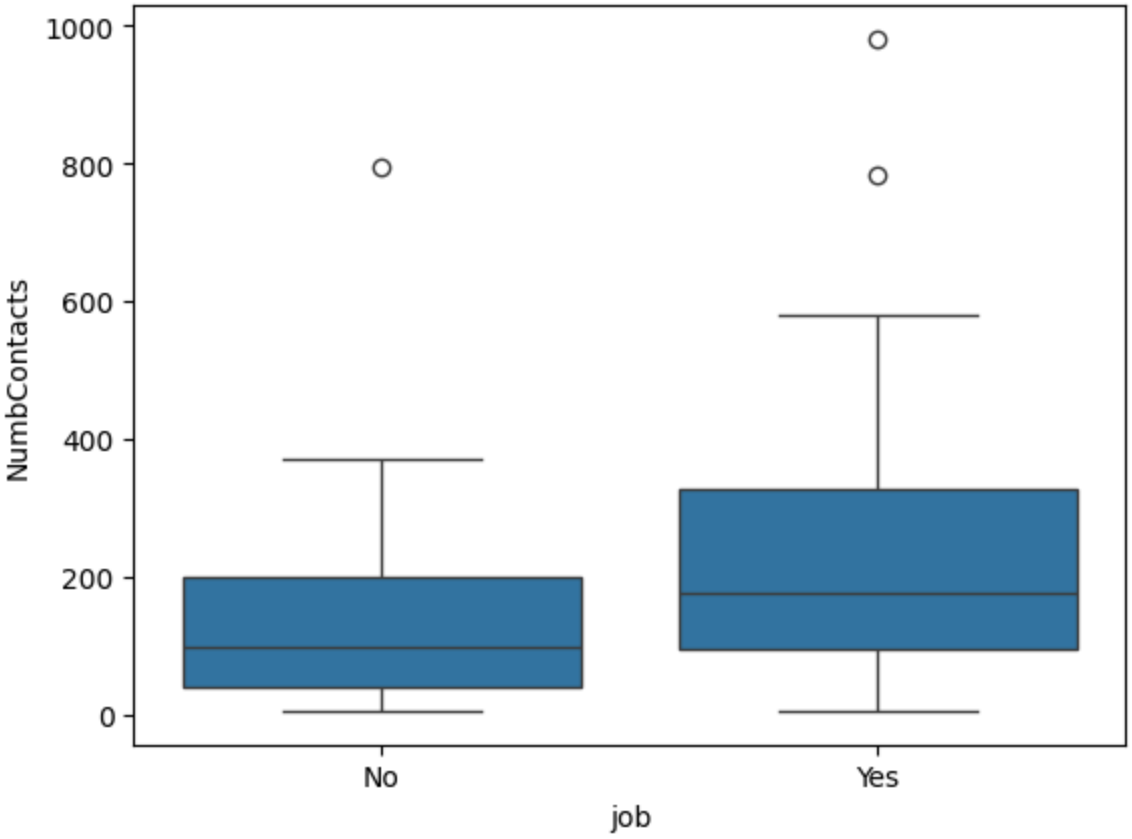
Open the first day survey data, make a new variable that is 1 if the student has a job, either on-campus or off-campus. Call that variable *job*. Then make a side by side boxplot of *job* vs *NumbContacts*. Comment on whether or not there seems to be a relationship between having a job and the number of contacts in their cell phone among DTSC 2301 students.

In [26]:
```
firstday = pd.read_csv("https://webpages.charlotte.edu/mschuck1/classes/DTSC2301/Da
firstday['job'] = firstday.apply(lambda row: 'Yes' if row['CampusJob'] == 'Yes' or

print(firstday.head())
```

```
        Timestamp                                Section Haircut  TimeToCampus  \
0  1/13/2025 9:56  002 Ageenko/Schuckers MWF 905 to 11a      40          2.00
1  1/13/2025 9:56  002 Ageenko/Schuckers MWF 905 to 11a      20          0.25
2  1/13/2025 9:56  002 Ageenko/Schuckers MWF 905 to 11a      22          2.00
3  1/13/2025 9:56  002 Ageenko/Schuckers MWF 905 to 11a      40          0.24
4  1/13/2025 9:56  002 Ageenko/Schuckers MWF 905 to 11a      50          0.00

  CharlotteHome NCHome  Stream_Hours SocialMedia  ClassYear  Exercise  \
0            No    Yes           8.0      TikTok  Sophomore       6.0
1           Yes    Yes          10.0     YouTube  Sophomore       2.0
2            No    Yes          20.0     YouTube  Sophomore       2.0
3           Yes    Yes          10.0      TikTok  Sophomore      21.0
4            No    Yes          20.0     YouTube  Sophomore      12.0

  CampusJob OffCampusJob  NumbSiblings BirthMonth  NumbPiercings  NumbTattoos  \
0        No           No             2  September              0            0
1        No          Yes             1        May              0            0
2        No           No             1   November              0            0
3        No          Yes             1     August              0            2
4        No          Yes             2      April              0            0

  NumbContacts  CellLastDigit ShoeSize  job
0          272              1        9   No
1          314              5      9.5  Yes
2          158              4      9.5   No
3          400              5       12  Yes
4          579              1       11  Yes
```

In [34]:
```
import seaborn as sns
firstday['NumbContacts']=pd.to_numeric(firstday['NumbContacts'],errors="coerce")
sns.boxplot(x="job", y="NumbContacts", data=firstday)
```

Out[34]:
```
<Axes: xlabel='job', ylabel='NumbContacts'>
```

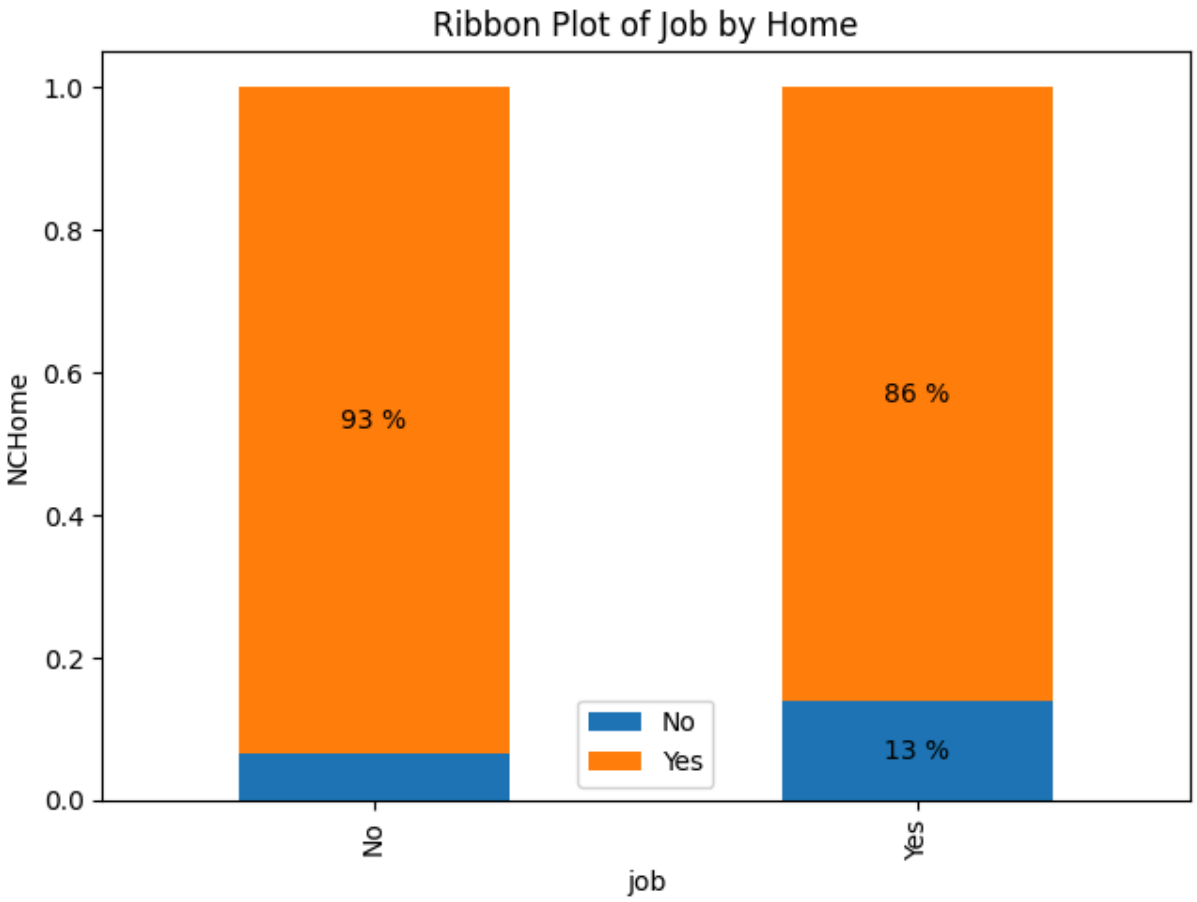


Question 4

For this question, make a ribbon plot with *job* (from the previous question) on the x-axis and *NCHome* (whether or not a student is from North Carolina) on the y-axis. Make the plot with those axes reversed. Explain which plot you prefer and why? Which feature/variable do you think should be the target variable here and why?

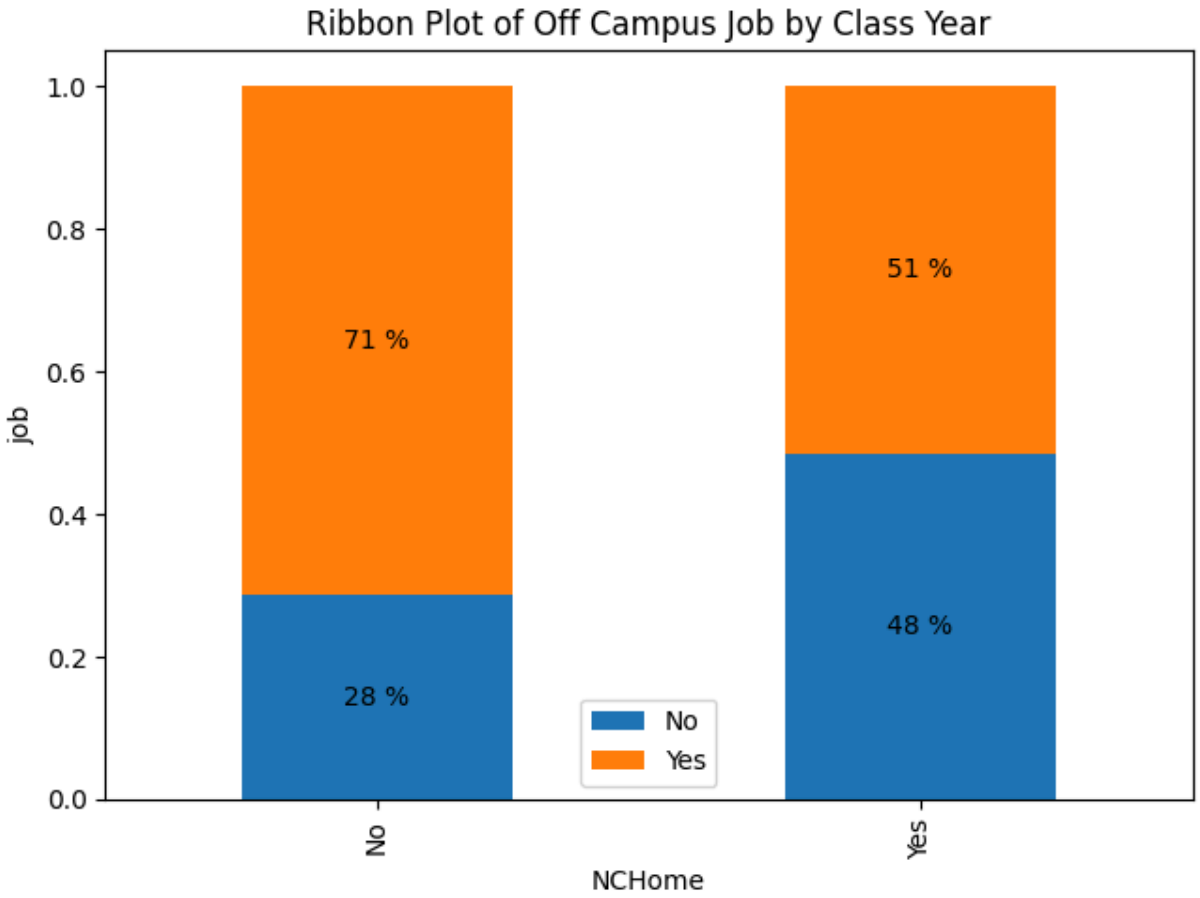
In [37]:
```
df=firstday
x_var, y_var = "job","NCHome"

# create a dataframe of counts
df_grouped = df.groupby(x_var)[y_var].value_counts(normalize=True).unstack(y_var)
# make a bar plot that is stacked
df_grouped.plot.bar(stacked=True)
# add a legend with "best" location
plt.legend(loc="best")
# get the cumulative percents across the groups
for ix, row in df_grouped.reset_index(drop=True).iterrows():
        cumulative = 0
        for element in row:
            if element == element and element > 0.1:
                plt.text(
                    ix,
                    cumulative + element / 2,
                    f"{int(element * 100)} %",
                    va="center",
                    ha="center",
                )
```

```
            cumulative += element
plt.tight_layout()
#bottom_bar = mpatches.Patch(color='lightblue', label='NC Home = Yes')
#plt.legend(handles=[bottom_bar])
plt.ylabel(y_var)
plt.title("Ribbon Plot of Job by Home")
# show the graph
plt.show()
```



In [36]:
```
df=firstday
x_var, y_var = "NCHome","job"

# create a dataframe of counts
df_grouped = df.groupby(x_var)[y_var].value_counts(normalize=True).unstack(y_var)
# make a bar plot that is stacked
df_grouped.plot.bar(stacked=True)
# add a legend with "best" location
plt.legend(loc="best")
# get the cumulative percents across the groups
for ix, row in df_grouped.reset_index(drop=True).iterrows():
        cumulative = 0
        for element in row:
            if element == element and element > 0.1:
                plt.text(
                    ix,
                    cumulative + element / 2,
                    f"{int(element * 100)} %",
                    va="center",
```

```
                    ha="center",
                )
            cumulative += element
plt.tight_layout()
#bottom_bar = mpatches.Patch(color='lightblue', label='NC Home = Yes')
#plt.legend(handles=[bottom_bar])
plt.ylabel(y_var)
plt.title("Ribbon Plot of Home effecting Job")
# show the graph
plt.show()
```



I think in this sceniaro it is better to have 'job on the Y-axis and 'NCHome' on the X-axis, I think it better shows the difference between two demographics. I think we should be targeting the percentage of students who have jobs base off of if their an in-state or out-of-state student.

Question 5

A data scientist developing a hiring model for sales agents discovers a correlation between having a job on-campus or off-campus and the number of contacts in the candidate's cellular phone. Should the algorithm use this relationship to select candidates for job interviews? Write a 80 - 120 word paragraph with your answer to this question. Consider the implications of the ethical fairness and bias principle of including this data in your model.

I think this would be a very poor relationship to use in when selecting candidates. I come to this conculsion simply because of the use of the 'NumbContacts' variable. Students are going to have a very wide range in the number of contacts they have, for a wide range of reasons. Using any algorithm with that variable just won't make any sense when trying to determine who might be the best candidates for their job offering. Using another variable like class year might be a better option, especially if they want a certain age or experience.