Project Stage 4

Group Members:
Phil Martinkus and Abhijeet Kamble

Merging the Walmart and Amazon Data Sets:
To begin with, the two datasets we merged contained information about laptops sold at Walmart and Amazon. The features described the laptops and included the screen size, RAM, hard drive capacity and more. Each dataset contained just over 3000 tuples.

After the matching process we took a look at the matches and found that tuples from the Walmart dataset matched up with many tuples from the Amazon dataset. Considering this and the fact that the Walmart data was generally cleaner, especially in the Battery Life column, we decided to prefer the values in the Walmart table when available. Additionally, there were many more unique tuples matched in the Amazon data set than the Walmart data.

To give a more detailed explanation of how exactly we merged tuples, we first found all tuples in the Amazon data that matched up with each tuple from the Walmart data. Then, for each tuple from the Walmart dataset, we merged it with all of its matched tuples from the Amazon data. For most of the attributes we just used the value from the Walmart tuple if it was not null. In the case that the value was null in the Walmart tuple, we took the most common value for that attribute in all of the matching Amazon tuples. We used this approach for Name, Brand, Screen Size, RAM, Hard Drive Capacity, Processor Type, Operating System and Battery Life. The exceptions to this policy are the Price and Processor Speed attributes. For the Price, we took the average of all the matched tuples. Since the Processor Speed column was cleaner in the Amazon table, we chose the most common value of the matched Amazon tuples for that feature.

Statistics on the Merged Table:
The final features in the merged table were ID, Name, Price, Brand, Screen Size, RAM, Hard Drive Capacity, Processor Type, Processor Speed, Operating System and Battery Life. In the end, this table contains 4,162 tuples. Below are some example tuples from the merged data:

ID	Name	Price	Brand	Screen Size	RAM	Hard Drive Capacity	Processor Type	Processor Speed	Operating System	Battery Life
0	HP Flyer Red...	299	HP	15.6 in	4 GB	500 GB	Intel Pentium	2.16 GHz	Windows 10	4.5 hours
2	HP Stream...	199	HP	11.6 in	4 GB	32 GB	Intel Celeron	1.6 GHz	Windows 10	10 h
4	HP Black Licorice...	329	HP	15.6 in	4 GB	500 GB	AMD A-Series	2.20 GHz	Windows 10	
5	HP 15-bw032WM...	569	HP	15.6 in	12 GB	1 GB	7th Generation Intel...	2.50 GHz	Windows 10	9.00 h

Data Analysis:
We wanted to explore how each brand of laptop is different from the others. We completed two seprate analysis tasks to learn more about the brands. First we uses a bit of OLAP style exploration to get an idea of how the brands compare to each other. Then, we attempted to train a classifier to predict a laptop brand from just its features.

OLAP Style Exploration:
The following table shows the results of a group by on the Brand and some aggregations on some of the numeric features in the merged table. We took a closer look at the table and found some interesting results.

	ID	Price	Screen Size (Numeric)	RAM (Numeric)	Hard Drive Capacity (Numeric)	Processor Speed (Numeric)	Battery Life (Numeric)
	count	mean	mean	mean	mean	mean	mean
Brand							
ASUS	381	744.020395	15.090237	9.768293	729.685301	2.012891	7.163934
Acer	560	608.746000	13.845962	7.569170	417.422164	2.129517	7.203390
Apple	559	657.938945	13.459546	6.081481	345.389791	2.092437	9.921656
Dell	559	568.354168	14.431223	8.565996	419.961945	2.430898	8.287794
HP	1120	436.927536	14.999099	6.748988	522.975207	2.192193	7.223881
Lenovo	983	563.939959	14.803406	9.247337	591.687124	2.179495	5.675904

Quantity: In the above table, we can see that this table contains the most HP and Lenovo laptops at 1118 and 979 laptops respectively. In the middle, Acer, Apple, and Dell all have between five and six hundred laptops in the table and we have the fewest data points for ASUS at just under 400.

Price: The most expensive brand, on average, is ASUS at nearly $750 per laptop with Apple in second at about $660. The brand with the cheapest laptops is HP at under $450 per laptop.

Next we will look at some of the features of each laptop to see if the more expensive brands justify their price with better products. If the more expensive brands have higher values for Screen Size, RAM, Hard Drive Capicity, Processor Speed, and/or Battery Life, then it would make sense that their laptops are more expensive.

Screen Size: ASUS, HP, and Lenovo have the largest screens with a mean of 15, or nearly 15, inches. Apple, on the other hand has the smallest average screen size at under 13.5 inches.

RAM: ASUS again tops the list along with Lenovo at over 9 GB of RAM for their average laptop. Apple and HP are at the bottom with an average RAM capacity of 6 and 6.7 respectively.

Hard Drive Capacity: ASUS has a substantial lead in average Hard Drive Capacity at over 700 GB with the closest brand, Lenovo, at under 600 GB. Dell and Apple generally have the lowest Hard Drive Capicties at about 420 GB per laptop.

Processor Speed: The fastest brand is Dell at over 2.4 GHz with the others all hovering between 2 and 2.2 GHz.

Battery Life: Lastly, Apple is the brand with the longest battery life at almost 10 hours with Lenovo by far the worst at under 6 hours of battery life.

Overall, ASUS and Apple sold their laptops for the highest prices. We found that ASUS generally justified the higher price because they seemed to sell higher end laptops on average. They were at the top, or at least near it, in nearly every category and it shows us that ASUS tends to only sell higher quality products. On the other hand, Apple’s products were on the lower end of most categories. It shows that, based on these features alone, Apple laptops are probably not worth the asking price. This however, does not consider some of the other features, such as the OS for example, that may give customers a reason to pay extra for the Apple brand.

Lenovo, Acer, and Dell were in the middle of the pack for price. Lenovo in particular seemed to be one of the best deals for the money out of any of the brands. Their laptops generally were towards the top of most categories despite the relatively modest price and it seems to indicate that Lenovo laptops are a bargain for their quality. Dell was by far the best laptop when it comes to processor speed, so anyone looking for a fast laptop at a decent price should take a look at Dell laptops.

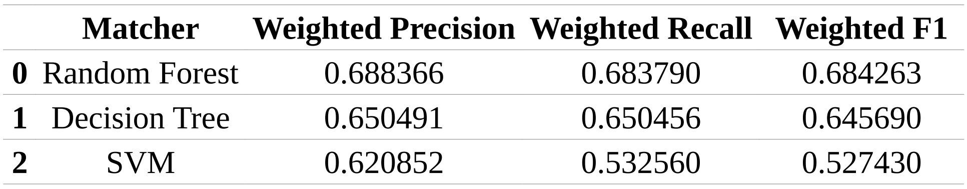
Finally, HP laptops had, on average, the lowest price, but they tended to be towards the middle for the other categories. This shows that anyone on a budget can get a pretty good quality laptop at an affordable price if they look at HP laptops.

Brand Classification:

After running our OLAP query, we wanted to investigate if it would be possible to classify the laptops based on these observations. Many technological products tend to seem relatively the same to the casual observer. What really is the difference between a Dell and Lenovo laptop? We think that if an ML algorithm can successfully classify the laptops, then it would show that there are significant differences in the brands.

We created some numeric features for the classification from our original features in the merged tables. We removed the non-numeric components of the string values and then converted the fields to floats. In the end, we used the Price, Screen Size, RAM, Hard Drive Capacity, Processor Speed, and Battery Life as our features to train and test the model. We then had to impute the missing features with the average value for each column. Next we were able to split the data into training and evaluation sets.

After our data was pre-processed, we ran cross validation for Random Forest, Decision Tree, and SVM classifiers. The results after cross validation are shown in the table below:



	Matcher	Weighted Precision	Weighted Recall	Weighted F1
0	Random Forest	0.688366	0.683790	0.684263
1	Decision Tree	0.650491	0.650456	0.645690
2	SVM	0.620852	0.532560	0.527430

Since predicting the brand is a multiclass problem, the precision, recall, and f1 values are calculated using a weighted average where the weights are based on the support for each feature. The results above show that the Random Forest Classifier was the best for this particular data set.

Next, we used the Random Forest Classifier to make predictions on the evaluation set. We obtained the following results:

```
Precision Score: 0.677709807418778
Recall Score: 0.6791546589817483
F1 Score: 0.6777767074410274
```

Overall, the scores are a bit low, but we still think that this is compelling evidence that there are some important differences and trends among the different brands. Clearly, the model was able to find some important signals to indicate the brand of a laptop from only its price, RAM, etc. We think this supports our findings from the OLAP analysis that different brands are better at providing different features in their laptops and different brands are able to sell products for a better or worse deal.

Future Work:

If we had more time to continue our analysis, we would want to expand both our OLAP and Classification studies. We believe that the most important thing to improve these studies would be to obtain all of the data found on Walmart and Amazon in the data acquisition step. Since we only were dealing with 3000 tuples from each table, we did not get a full picture about the laptop landscape today. Each of these websites contained data for over 10,000 laptops and we think we could have provided better analysis with better data.

We also think it might have been interesting to add some sort of review data for each laptop. It would be interesting to compare the price for each brand to actual user ratings for the laptops. Maybe HP laptops satisfy customers much better than ASUS laptops despite the gulf in the the prices. It would have been interesting to investigate the relationship between brand and customer satisfaction.

Merging Python Script (merging.py):

```
# Import files and packages
import pandas as pd


# Merge two tables
def merge_tables(A, B, matches):

  # First we will find all of the tuples that have not matches in each table
        data = []

        # Getting non-matches for A
        for tuple in A.itertuples(index=False):
                if not any(tuple.ID == matches.ltable_ID):
                        data.append(tuple)

        # Getting non-matches for B
        for tuple in B.itertuples(index=False):
                if not any(tuple.ID == matches.rtable_ID):
                        data.append(tuple)

        # Get all of the matches for each tuple in the walmart data
        A_matches = {}
        for tuple in matches.itertuples(index=False):

                 # If there is no entry for this id, start a list
                 if tuple.ltable_ID not in A_matches:
                    A_matches[tuple.ltable_ID] = [tuple.rtable_ID]

                 # Otherwise, append this value to the list
                 else:
                    A_matches[tuple.ltable_ID].append(tuple.rtable_ID)

        # Merge each set of matches
        for match in A_matches:
                data.append(merge_tuple(match, A_matches[match], A, B))

        # Create a data frame from the merged tables
        merged_df = pd.DataFrame(data, columns=A.columns)
        return merged_df.drop('Clean Name', axis=1)

# Merge two tuples.
def merge_tuple(ltuple, rtuples, A, B):

   # Prefer Walmart data for most attributes
   feats = ['Name', 'Brand', 'Screen Size', 'RAM', 'Hard Drive Capacity', 'Processor Type',
            'Operating System', 'Operating System', 'Battery Life']
   for feat in feats:

      # If the value for this feature is null in ltuple, get most common value from rtuples
      if A.loc[ltuple][feat] == float('nan'):
         A.loc[ltuple][feat] = most_common(rtuples, feat, B)

   # Merge Processor Speed. Prefer Amazon data for this attribute
   A.loc[ltuple]['Processor Speed'] = most_common(rtuples, 'Processor Speed', B)

   # Merge Price using average
   total = 0
   num = 0

   # Only average non null values
   if A.loc[ltuple]['Price'] != float('nan'):
        total += A.loc[ltuple]['Price']
        num += 1

    for id in rtuples:
        if B.loc[id]['Price'] != float('nan'):
            total += B.loc[id]['Price']
            num += 1

    A.loc[ltuple]['Price'] = total / num

    # Return the merged tuple
    return A.loc[ltuple]


# Return the most common value in tuples for the given feature.
def most_common(tuples, feature, B):
    vals = {}

    # Count occurances of each value for this feature
    for id in tuples:
        if B.loc[id][feature] in vals:
            vals[feature] += 1
        else:
            vals[feature] = 1

    # Return most common value
    max_count = 0
    max_val = None
    for val in vals:
        if vals[val] > max_count:
            max_count = vals[val]
            max_val = val
    return val
```