CSC311H5: Machine Learning Challenge

Date: April 8th, 2024

By Chun-Kai Chen (1006428457),
Arush Awasthi (1005954861),
Shihui Lu (1005955496),
& Vijay Sai Patnaik (1008200732)

Table of Contents

Data Exploration **2**
- General Overview 2
- Exploring Q1: Popularity (Included) 2
- Exploring Q2: Social Media Viralness (Included) 3
- Exploring Q3: Architectural Uniqueness (Included) 4
- Exploring Q4: Enthusiasm for Street Parties (Included) 5
- Exploring Q5: Travel Companion (Included)fg 6
- Exploring Q6: Word Relatability (Included) 8
- Exploring Q7: Average January Temperature (Included) 12
- Exploring Q8: Number of Languages Spoken (Excluded) 13
- Exploring Q9: Number of Fashion Styles (Excluded) 14
- Exploring Q10: Quote (Partially Included) 15
- Input Features & Feature Encoding 18
- Splitting the Data 19

Model Exploration **19**
- K-Nearest Neighbors 19
- Random Forests 19
- Multilayer Perceptron 20
- Naive Bayes 20

Model Choice & Hyperparameters **21**
- Hyperparameter Tuning: KNN 21
- Hyperparameter Tuning: Random Forest 21
- Hyperparameter Tuning: Naive Bayes 23
- Hyperparameter Tuning: MLP 24
- Model Choice 27

Predictions **27**

Workload Distribution **28**
- Chun-Kai Chen 28
- Arush Awasthi 28
- Shihui Lu 28
- Vijay Sai Patnaik 28

Data Exploration

In this section, we will be performing a thorough exploration of the survey responses by carefully examining the statistics and distributions of each survey question. This will offer us insight as to how the data correlates to each city and, ultimately, aid us in choosing which features to include or exclude in the training phase of the machine learning challenge. Additionally, we will also perform feature encoding as well as splitting the data into training, validating, and testing sets in a meaningful manner.

General Overview

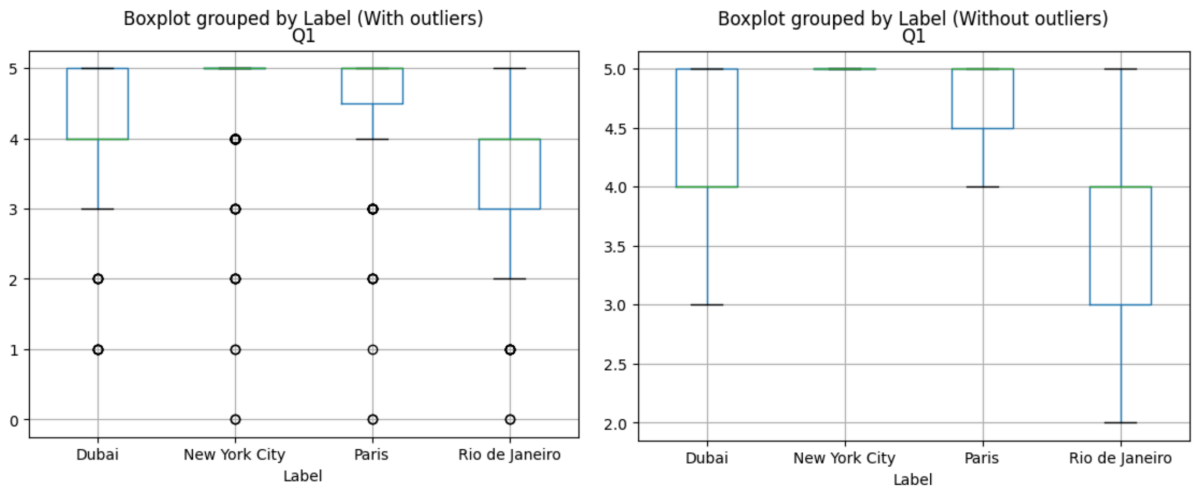
```
### Data points per city ###
Dubai             367
Rio de Janeiro    367
New York City     367
Paris             367
```

Looking at the entire dataset, we see that each city has an equal amount of data points. This is good as it allows us to train, tune, and test the model whilst reducing prediction biases in the model. We will see how in the later subsection ("S*plitting the Data*").

Exploring Q1: Popularity (Included)

Q1 is written as follows:
"From a scale 1 to 5, how popular is this city? (1 is the least popular and 5 is the most popular)"

```
### Statistics for Q1 ###
Dubai:                mean=4.23433242506812,   median=4.0
Rio de Janeiro:       mean=3.5395095367847413, median=4.0
New York City:        mean=4.782016348773842,  median=5.0
Paris:                mean=4.659400544959128,  median=5.0
```

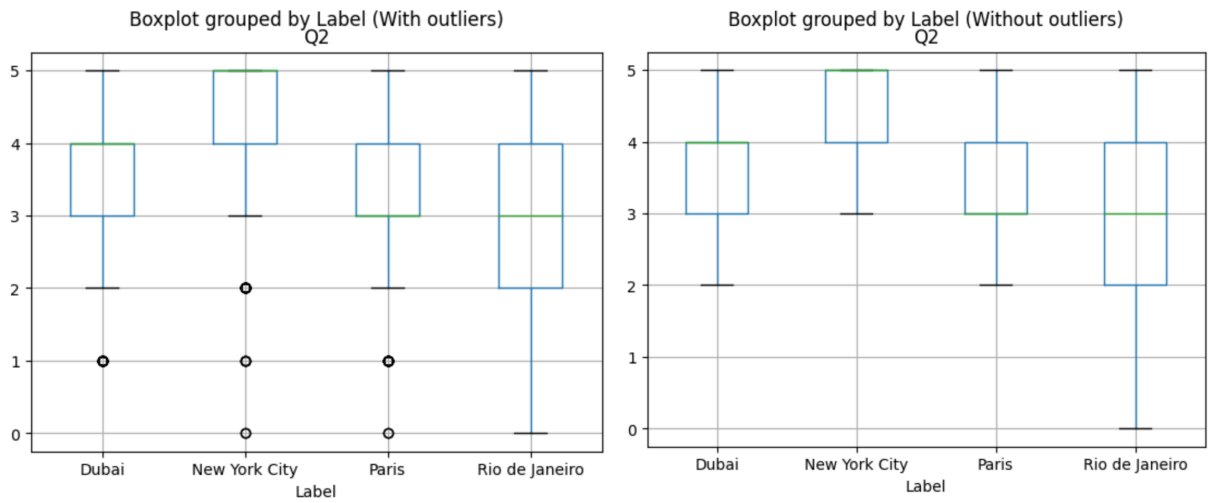


Examining the mean of Q1 reveals that the average popularity of each city hovers approximately around ~4.5, with the exception of Rio de Janeiro, which has a notably lower average popularity of ~3.54. Looking at the boxplots, both with and without the outliers, further illustrates this disparity, showing that the popularity distribution of the other cities are clustered around 4 and tends toward the higher and of the scale, while Rio de Janeiro's popularity distribution is skewed towards the lower end of the rating scale, the median also helps reinforce this idea. Thus, we decided to incorporate Q1 as an input feature in our model as it may aid the model in identifying Rio de Janeiro.

Exploring Q2: Social Media Viralness (Included)

Q2 is written as follows:
"On a scale of 1 to 5, how efficient is this city at turning everyday occurrences into potential viral moments on social media? (1 is the least efficient and 5 is the most efficient)"

```
### Statistics for Q2 ###
Dubai:                  mean=3.4059945504087192, median=4.0
Rio de Janeiro:         mean=2.8419618528610355, median=3.0
New York City:          mean=4.430517711171662,  median=5.0
Paris:                  mean=3.3405994550408717, median=3.0
```

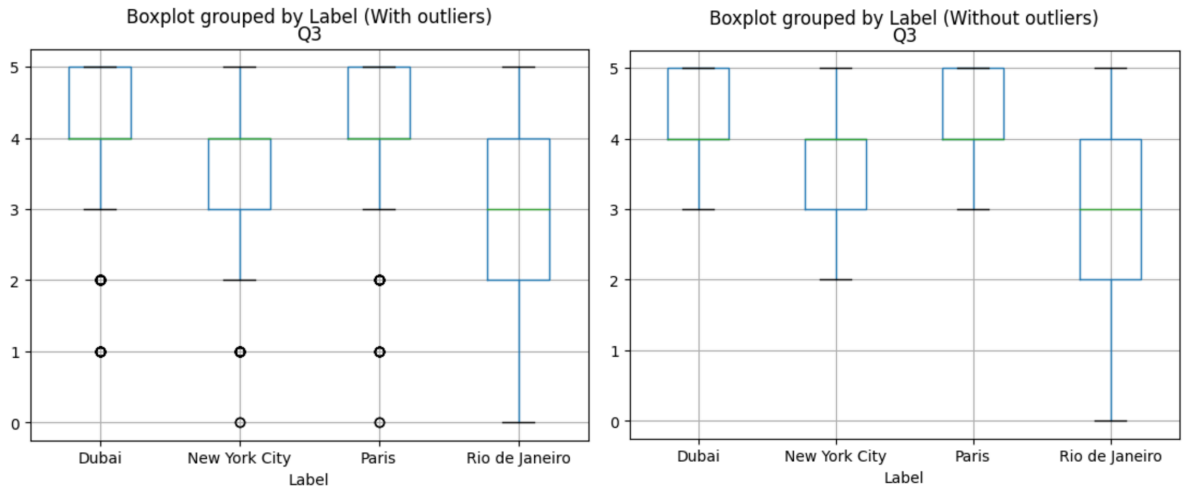


Examining the mean and median of Q2 shows that Dubai and Paris have near identical social media viralness at ~3.37, ranking in the middle out of the four cities, while New York City ranks the highest at ~4.43 and Rio de Janeiro ranks the lowest at ~2.84. Looking at the boxplot, both with and without the outliers, illustrates the same story. We decided to include Q2 as an input feature as it may help the model in identifying New York City as it is correlated with high social media viralness, and at the same time identify Rio de Janeiro as it is correlated with low social media viralness.

Exploring Q3: Architectural Uniqueness (Included)

Q3 is written as follows:
"Rate the city's architectural uniqueness from 1 to 5, with 5 being a blend of futuristic wonder and historical charm."

```
### Statistics for Q3 ###
Dubai:                  mean=4.0272479564032695, median=4.0
Rio de Janeiro:         mean=3.0790190735694822, median=3.0
New York City:          mean=3.4523160762942777, median=4.0
Paris:                  mean=4.201634877384196,  median=4.0
```

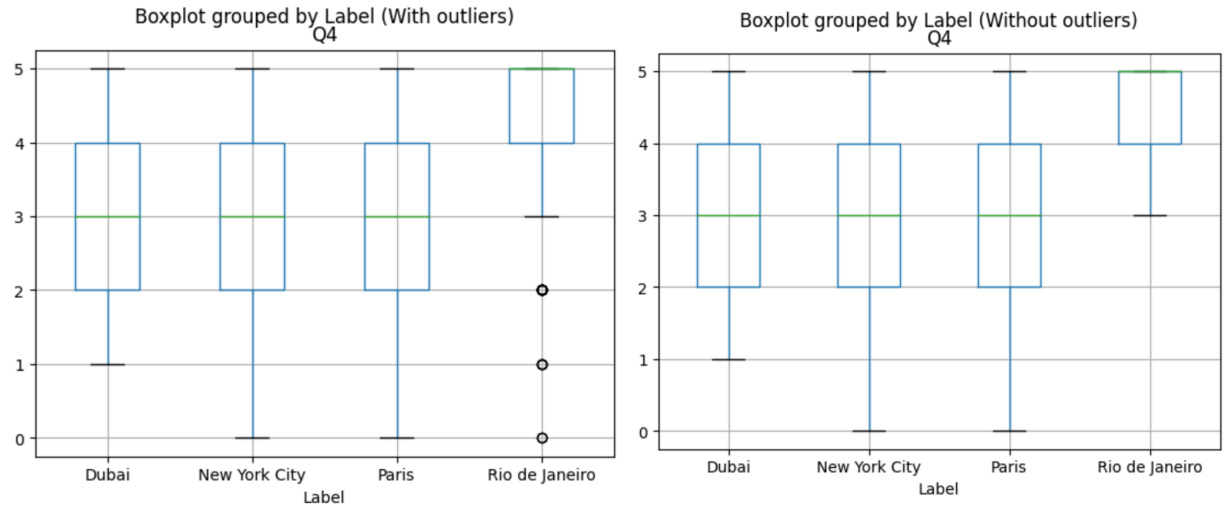


Examining the mean of Q3 shows that Dubai and Paris are very similar in architectural uniqueness, both ranking high with Paris being the highest and Dubai following close behind. We also observe that New York City ranks third with an average of ~3.45 and Rio de Janeiro ranking the lowest with an average rating of ~3. Taking a closer look at the median and the boxplots, reveals that Dubai, New York City, and Paris are distributed very similarly with the exception of Rio de Janeiro, distributing lower on the rating scale. Meaning Rio de Janeiro may be correlated with lower rating in architectural uniqueness. With this, we've decided to include Q3 as a feature to aid the model in identifying Rio de Janeiro.

Exploring Q4: Enthusiasm for Street Parties (Included)

Q4 is written as follows:
"Rate the city's enthusiasm for spontaneous street parties on a scale of 1 to 5, with 5 being the life of the celebration."

```
### Statistics for Q4 ###
Dubai:                mean=2.768392370572207,  median=3.0
Rio de Janeiro:       mean=4.433242506811989,  median=5.0
New York City:        mean=3.2752043596730247, median=3.0
Paris:                mean=3.035422343324251,  median=3.0
```



Examining the mean and median of Q4 shows a strong correlation between enthusiasm for street parties and Rio de Janeiro. We see that Dubai, New York City, and Paris have average ratings of enthusiasm for street parties centered at around 3, while Rio de Janeiro is centered much higher at around 4-5. Looking at the boxplots, both with and without the outliers, further cements this idea. We can observe that Dubai, New York City, and Paris have nearly identical distributions while Rio de Janeiro has a distribution centered much higher on the scale as well as being skewed towards the higher end. Thus, we've decided to include Q4 as an input feature as it appears to be a strong feature in identifying Rio de Janeiro.

Exploring Q5: Travel Companion (Included)fg

For this question we will break it down into its subcomponents, whether or not the respondent will travel to a given city accompanied by a certain companion.

Q5-1: *"If you were to travel to this city, would your partner be likely to be with you?"*

```
### Statistics for Q5-1 ###
Dubai:                  proportion=0.5340599455040872
Rio de Janeiro:         proportion=0.5831062670299727
New York City:          proportion=0.6348773841961853
Paris:                  proportion=0.8610354223433242

### Partner Count (Out of 367) ###
 Dubai: 196, Rio de Janeiro: 214, New York City: 233, Paris: 316
```

Looking at the proportions listed above, we see that Paris has an overwhelmingly high proportion of people being accompanied by their partner (86%) compared to all other cities (with proportions hovering around ~58%), showing strong correlation between Paris and the presence of a partner. With this discovery, we've decided to include this subcomponent as an input feature to aid the

model in identifying Paris.

Q5-2: *"If you were to travel to this city, would your friend be likely to be with you?"*

```
### Statistics for Q5-2 ###
Dubai:                  proportion=0.7683923705722071
Rio de Janeiro:         proportion=0.8310626702997275
New York City:          proportion=0.7874659400544959
Paris:                  proportion=0.5013623978201635

### Friend Count (Out of 367) ###
 Dubai: 282, Rio de Janeiro: 305, New York City: 289, Paris: 184
```

Looking at the proportions here, we see that Dubai, Rio de Janeiro, and New York City have fairly high proportions of respondents being accompanied by their friends (~77%, ~83, and ~79% respectively), with the exception of Paris having a low proportion (~50%). This illustrates that the absence of a friend may be correlated to Paris. Thus, we've decided to include this subcomponent as an input feature since it may aid the model in identifying Paris.

Q5-3: *"If you were to travel to this city, would your siblings be likely to be with you?"*

```
### Statistics for Q5-3 ###
Dubai:                  proportion=0.3460490463215259
Rio de Janeiro:         proportion=0.3188010899182561
New York City:          proportion=0.5613079019073569
Paris:                  proportion=0.3242506811989101

### Siblings Count (Out of 367) ###
 Dubai: 127, Rio de Janeiro: 117, New York City: 206, Paris: 119
```

Looking at the proportions listed above, we see that New York City has the highest proportion of respondents being accompanied by their siblings at ~56% compared to all other cities (~34%, ~32%, and ~32% for Dubai, Rio de Janeiro, and Paris respectively). Thus, we've chosen to include this subcomponent as an input feature for aiding the model in identifying New York City.

Q5-4: *"If you were to travel to this city, would your co-worker be likely to be with you?"*

```
### Statistics for Q5-4 ###
Dubai:                  proportion=0.23978201634877383
Rio de Janeiro:         proportion=0.08719346049046321
New York City:          proportion=0.5776566757493188
Paris:                  proportion=0.11444141689373297

### Co-Worker Count (Out of 367) ###
```

```
Dubai: 88, Rio de Janeiro: 32, New York City: 212, Paris: 42
```

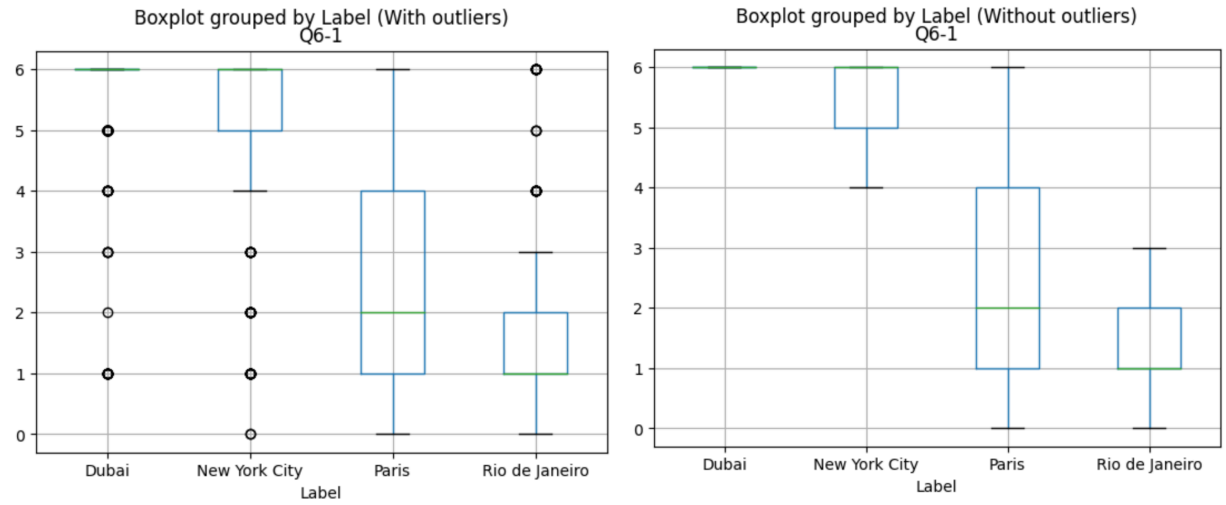
Looking at the proportions listed above, we see New York City as the highest proportions of respondents accompanied by co-workers at ~58%, followed by Dubai at ~24%, Rio de Janeiro at ~9%, and Paris at ~11%. With this we can see that New York City is correlated with high proportions, thus we will include this subcomponent as an input feature for aiding the model in identifying New York City.

Exploring Q6: Word Relatability (Included)

Similar to Q5, we are going to break Q6 into its subcomponents, each subcomponent being the relatability of the word in regards to each city.

Q6-1: *"Rank how relatable "skyscrapers" is to this city."*

```
### Statistics for Q6-1 ###
Dubai:                  mean=5.539509536784741, median=6.0
Rio de Janeiro:         mean=1.7411444141689374, median=1.0
New York City:          mean=5.130790190735695, median=6.0
Paris:                  mean=2.6594005449591283, median=2.0
```

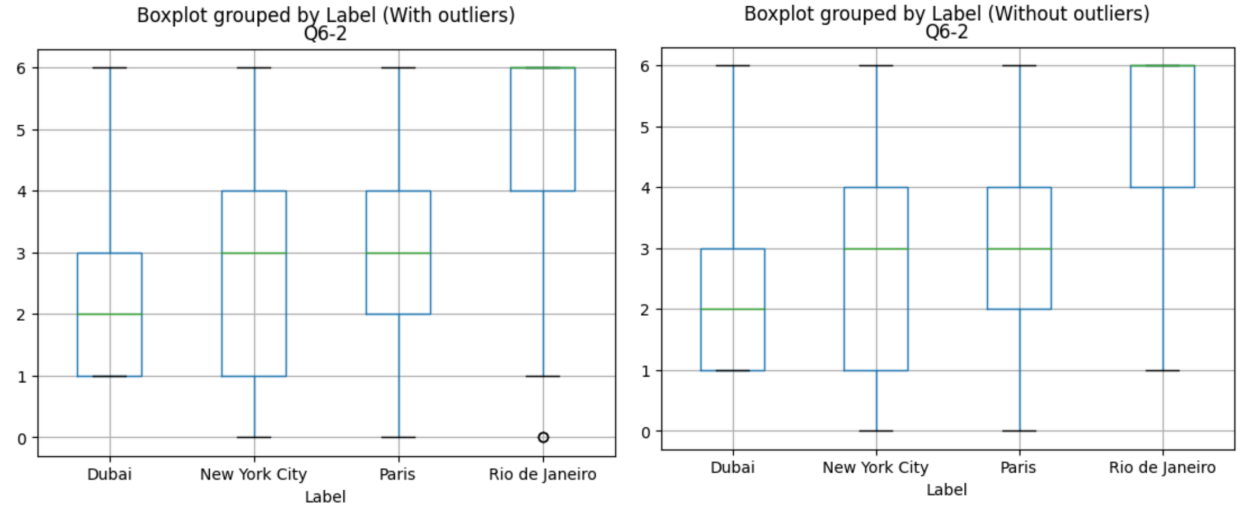


Looking at the mean and median as shown above, we can observe a strong correlation between high relatability of "skyscrapers" and Dubai, as well as, New York City. Inversely, we see that Rio de Janeiro and Paris are strongly correlated with low relatability to "skyscrapers". Looking at the boxplots, both with and without outliers, further confirms this idea. With this information, we've decided to include this subcomponent as an input feature as it may help the model narrow down predictions into two groups, either being Dubai or New York City, or Rio de Janeiro or Paris.

Q6-2: *"Rank how relatable 'sport' is to this city."*

```
### Statistics for Q6-2 ###
```

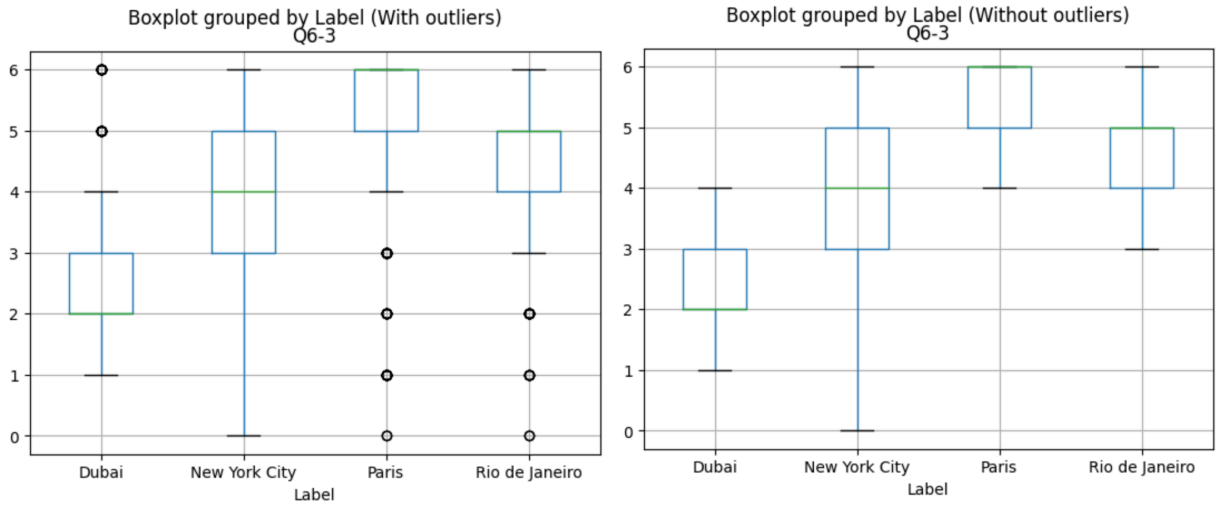
```
Dubai:                  mean=2.482288828337875, median=2.0
Rio de Janeiro:         mean=5.0190735694822886, median=6.0
New York City:          mean=2.749318801089918, median=3.0
Paris:                  mean=2.9182561307901906, median=3.0
```



Looking at the mean and median, we can observe that Rio de Janeiro is correlated with high relatability to "sport" while other cities rank much lower (around 2-3). Looking at the boxplots also illustrates the same idea. In fact, we see that the other cities are distributed similarly while Rio de Janeiro is distributed much more uniquely (centered and distributed much higher). Thus, we've decided to include Q6-2 to aid the model in identifying Rio de Janeiro.

Q6-3: *"Rank how relatable "art and music" is to this city."*

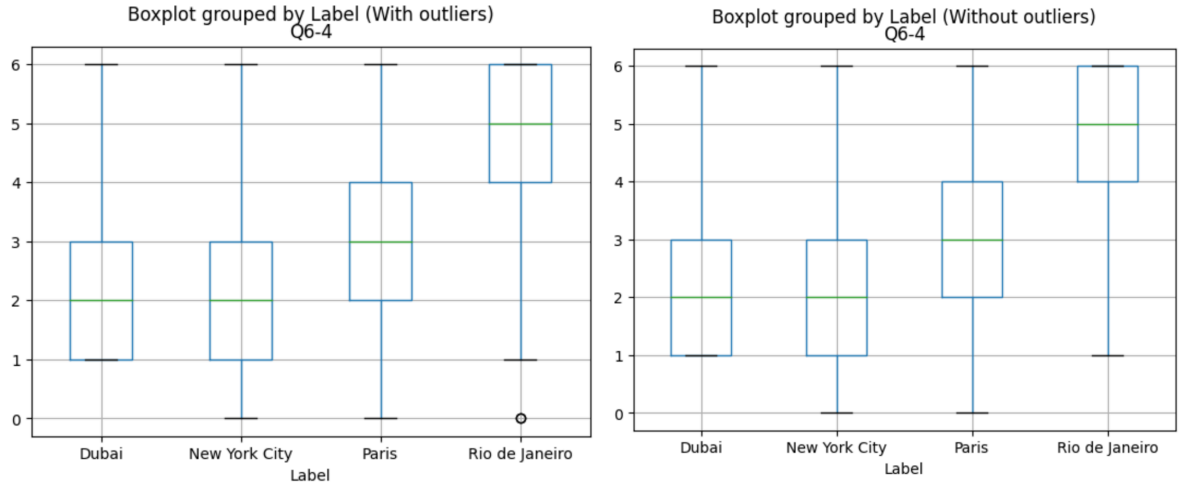
```
### Statistics for Q6-3 ###
Dubai:                  mean=2.5885558583106265, median=2.0
Rio de Janeiro:         mean=4.446866485013624, median=5.0
New York City:          mean=3.683923705722071, median=4.0
Paris:                  mean=5.15258855585831, median=6.0
```



Looking at the mean and median, we see that each city has a relatively distinct relatability to “art and music”. Looking at the boxplots, with and without outliers, we also observe that the distributions of each city appear to be relatively unique to each other (centered differently as well as being skewed differently). Thus, we’ve decided to include Q6-3 as it may help the model in identifying individual cities from each other.

Q6-4: *“Rank how relatable “carnival” is to this city.”*

```
### Statistics for Q6-4 ###
Dubai:                  mean=2.5068119891008176, median=2.0
Rio de Janeiro:         mean=4.839237057220709, median=5.0
New York City:          mean=2.3896457765667574, median=2.0
Paris:                  mean=2.869209809264305, median=3.0
```

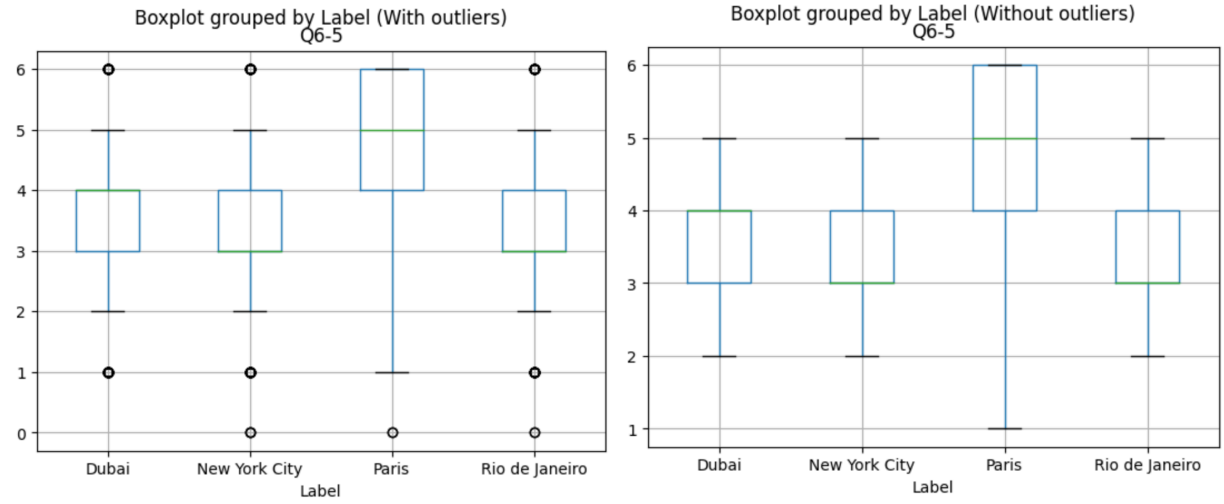


Looking at the mean and median reveals that Rio de Janeiro is correlated with high relatability to “carnival” compared to the other cities. Looking at the boxplot, both with and without outliers, further reinforces this idea. We can observe that Dubai, New York City, and Paris are similarly distributed and centered close to each other while Rio de Janeiro stands out by being centered much

higher. With this, we've decided to include Q6-4 as an input feature as it may prove useful for the model in identifying Rio de Janeiro.

Q6-5: *"Rank how relatable "cuisine" is to this city."*

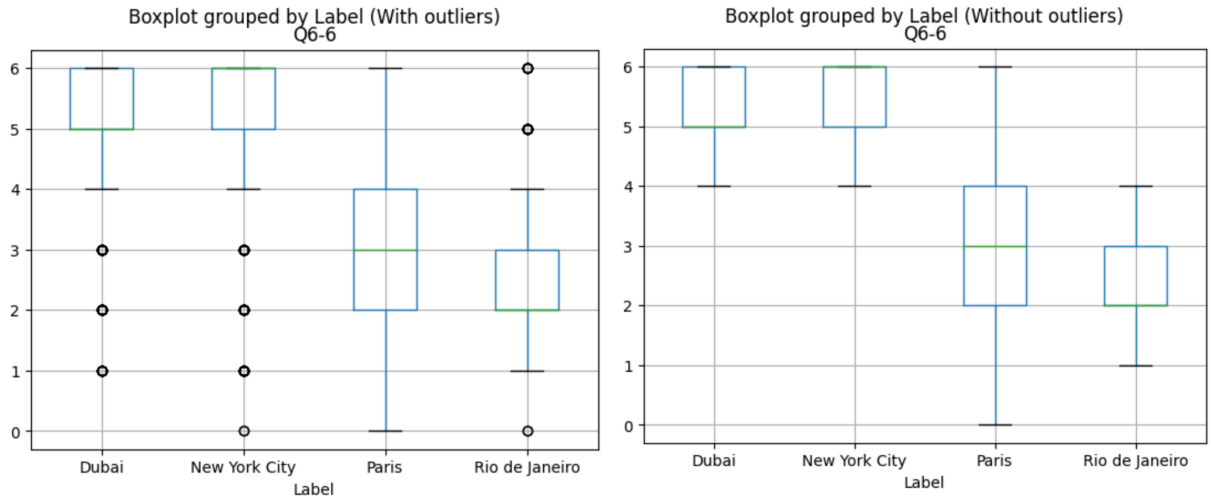
```
### Statistics for Q6-5 ###
Dubai:                  mean=3.449591280653951, median=4.0
Rio de Janeiro:         mean=3.3569482288828336, median=3.0
New York City:          mean=3.3896457765667574, median=3.0
Paris:                  mean=4.8038147138964575, median=5.0
```



Looking at the mean and median, we see that the relatability of "cuisine" is very similar across Dubai, Rio de Janeiro, and New York City, with the exception of Paris, ranking much higher than the aforementioned cities. Looking at the boxplots, both with and without outliers, illustrate the same pattern. We see that Dubai, New York City, and Rio de Janeiro are almost identically distributed around the same center with Paris distributed much more uniquely (centered much higher with a slightly wider variance). This shows correlation between high relatability to "cuisine" and Paris. Thus, we've decided to include Q6-5 as an input feature to help train the model in identifying Paris.

Q6-6: *"Rank how relatable "economic" is to this city."*

```
### Statistics for Q6-6 ###
Dubai:                  mean=4.880108991825613, median=5.0
Rio de Janeiro:         mean=2.32425068119891, median=2.0
New York City:          mean=4.994550408719346, median=6.0
Paris:                  mean=3.2779291553133514, median=3.0
```

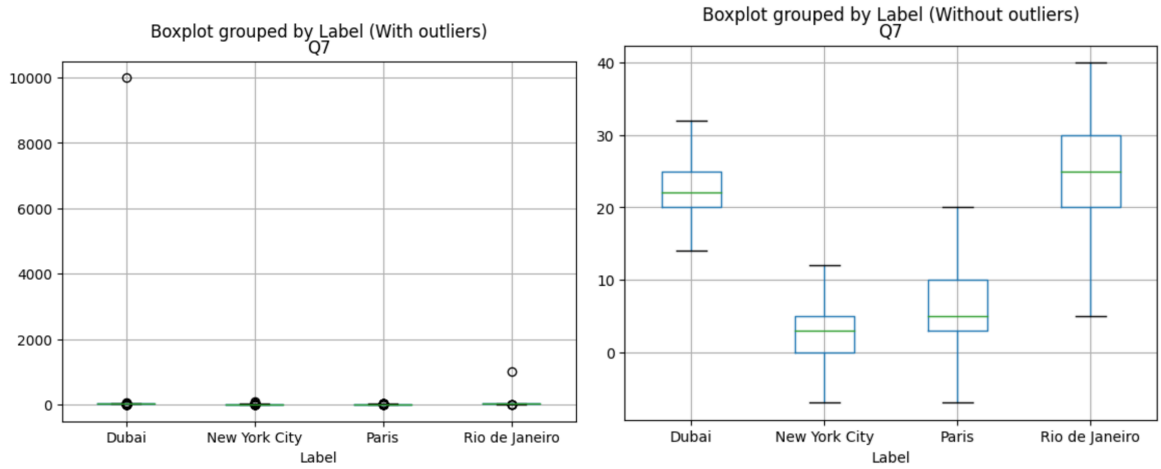


Looking at the mean and median, we immediately notice that Dubai and New York City have a high relatability to “economic”, while Paris and Rio de Janeiro both relate much less. Investigating the boxplots, both with and without the outliers, shows that Dubai and New York City are distributed similarly, as with Paris and Rio de Janeiro. Revealing that high relatability to “cuisine” may be correlated with Dubai and New York City, while lower relatability may correlate with Paris and Rio de Janeiro. Thus, we’ve decided to include Q6-6 as an input feature as it may help the model narrow down the prediction into either Dubai or New York City, or alternatively, Paris or Rio de Janeiro.

Exploring Q7: Average January Temperature (Included)

Q7 is written as follows:
“In your opinion, what is the average temperature of this city over the month of January? (Specify your answer in Celsius)”

```
### Statistics for Q7 ###
Dubai:                  mean=49.438692098092645, median=22.0
Rio de Janeiro:         mean=27.010899182561307, median=25.0
New York City:          mean=3.3583106267029974, median=3.0
Paris:                  mean=6.863760217983652, median=5.0
```

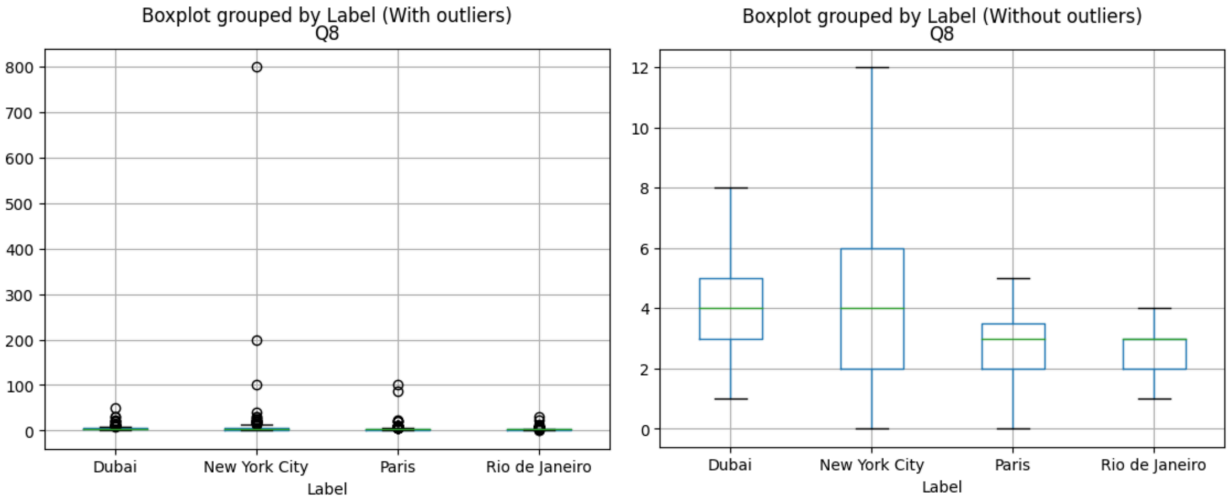


Looking at the mean and median as shown above, we immediately notice that Dubai has the highest temperature in January followed by Rio de Janeiro. We also see that compared to Dubai and Rio de Janeiro, New York City and Paris have relatively low temperatures. Looking at the boxplot, we noticed that there are outliers that are skewing the data. After removing the outliers, we see a better representation of the distributions. We see that the high temperatures correlate with Dubai and Rio de Janeiro, while low temperatures correlate with New York City and Paris. Thus, we've decided to include this question as an input feature as it may help the model narrow down predictions into two groups, Dubai and Rio de Janeiro, or alternatively, New York City and Paris.

Exploring Q8: Number of Languages Spoken (Excluded)

Q8 is written as follows:
"How many different languages might you overhear during a stroll through the city?"

```
Statistics for Q8
Dubai:                  mean=4.566757493188011, median=4.0
Rio de Janeiro:         mean=3.016348773841962, median=3.0
New York City:          mean=7.912806539509536, median=4.0
Paris:                  mean=3.5967302452316074, median=3.0
```

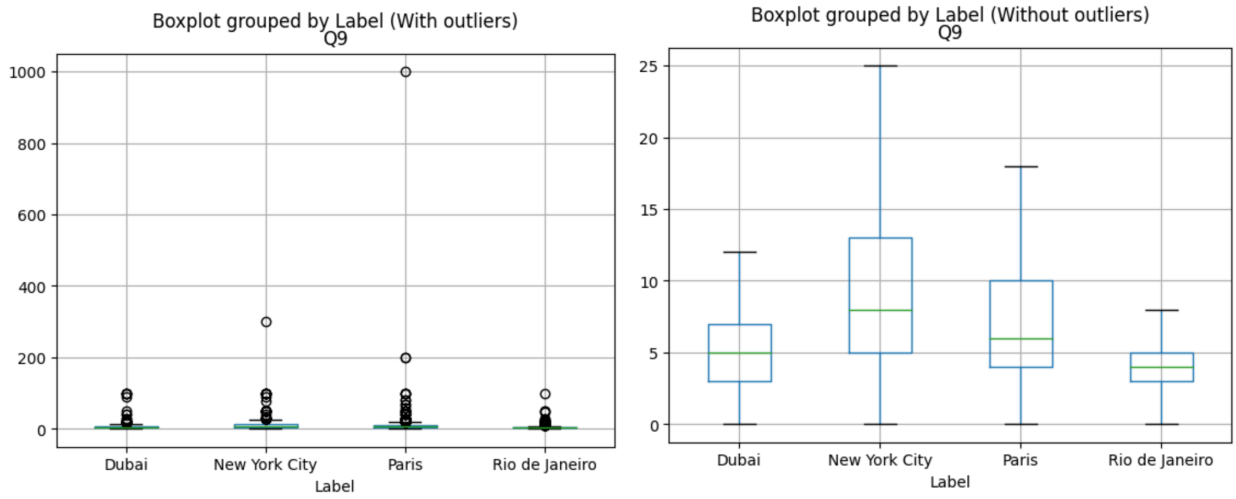


Looking at the statistics above, we see the survey results indicated that New York City has the highest number of languages spoken. However, taking a closer look at the boxplot with outliers, shows that the mean is skewed by a large outlier. Removing the outliers reveals that the distributions are centered and varied very similarly (especially given the range of the responses). Because of this we've decided to exclude this survey question as it may not provide the model with meaningful information in identifying cities and possibly introducing noise instead.

Exploring Q9: Number of Fashion Styles (Excluded)

Q9 is written as follows:
"How many different fashion styles might you spot within a 10-minute walk in the city?"

```
Statistics for Q9
Dubai:                  mean=7.114441416893733, median=5.0
Rio de Janeiro:         mean=5.76566757493188, median=4.0
New York City:          mean=12.811989100817438, median=8.0
Paris:                  mean=14.2425068119891, median=6.0
```



Looking at the mean as shown above, we see that New York City and Paris have the highest average fashion styles. However taking a closer look at the data using boxplots, we see that outliers exist to skew the distribution for both cities. Removing the outliers shows us that the distributions of responses from each city are centered and distributed relatively similarly given the range of the response. With this we've decided to exclude this question from the input feature as it may not provide useful information and potentially introduce noise to the model.

Exploring Q10: Quote (Partially Included)

Q10 is written as follows:
"What quote comes to mind when you think of this city?"
For this field of our data, we have a more complex structure than a rating or a number. Since we are unable to extract the full intended meaning of a sentence programmatically, we look at the presence of individual words. This leads us to assessing two criteria when trying to figure out how useful data from this feature could be:

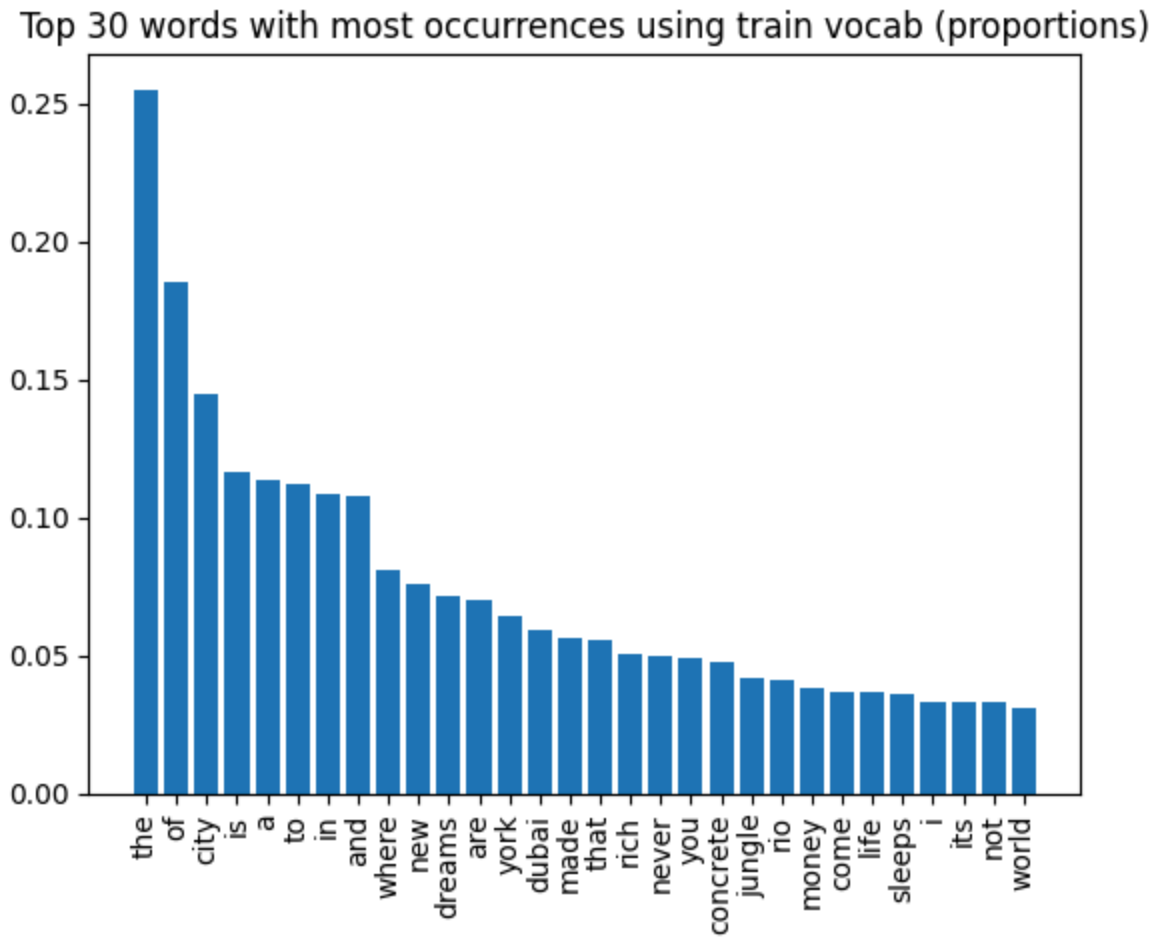
1. How strong of an indication does each word provide for a category?

   The presence of a word providing a strong indication for a certain category is trivially helpful.

2. How prevalent is this word

   Even if we have a strong indicator of a word, we won't be able to leverage its strength if it does not appear in many sentences

We start by looking at the proportion of sentences each word is present in. Below is a graph of the top 30 most frequent words:



The next step would be to see how much of an indication a word provides for each category. We hypothesize that non-noun words (and maybe some verbs) will not signify much, because they must occur in any grammatically valid sentence (like “the”, “is”, “a” etc.) We start with a quick analysis of “the”, since it is not only the most prevalent word, but also by a considerate margin:

```
34% entries that were classified as Dubai have the word 'the' in their quote.
32% entries that were classified as New York have the word 'the' in their quote.
32% entries that were classified as Paris have the word 'the' in their quote.
25% entries that were classified as Rio have the word 'the' in their quote.
```

We can see that the word “the” provides just as much information for one category as it does for another. So in the overall calculation, it does not “influence” much. This logic can be generalized to other “grammatical constructs” that do not carry any special meaning as well.

So now our quest is to look for meaningful words. We already have a heuristic in mind (nouns and verbs), but it would be nice to back it up with some numbers. A very trivial choice to make is to use the name of the city itself, since it does appear relatively frequently (note that the vocabulary size is about 1300 words, so being in the top 30 is a good sign). While the name of the city may not always occur in a quote:

```
17% entries that were classified as Dubai have the word 'dubai' in their quote.
24% entries that were classified as New York have the word 'york' in their quote.
17% entries that were classified as Paris have the word 'paris' in their quote.
13% entries that were classified as Rio have the word 'rio' in their quote.
```

It is also important to note that these city names will hardly occur in the quotes for any other city, if at all. Take for example 'Dubai":

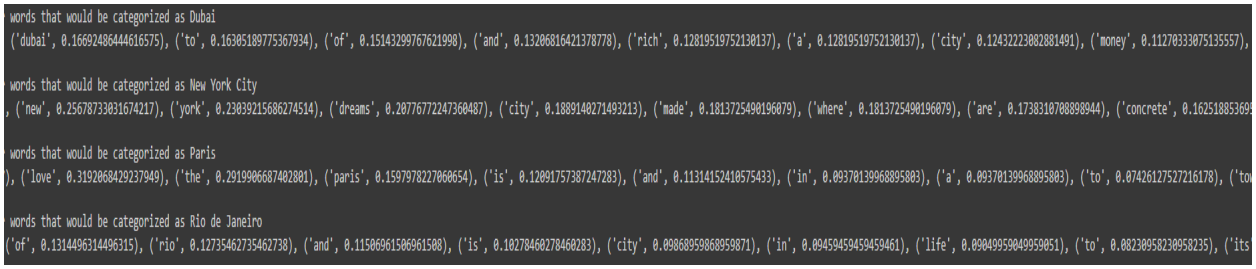
```
0.0% entries that were classified as New York have the word 'dubai' in their quote.
0.0% entries that were classified as Paris have the word 'dubai' in their quote.
0.0% entries that were classified as Rio have the word 'dubai' in their quote.
```

This feature being present can provide a strong indicator that a data item should be categorized as "Dubai". So these words are worth considering a bit more. Now we can look at some other words. Since looking at all of them is too long, I will take words from the top 30 (I also include "love" here despite not appearing in our graph since it was in the top 30 for a bigger sample):

```
The word 'city' is distributed among the four categories as follows:
- Dubai: 15.517241379310345%
- New York City: 24.82758620689655%
- Paris: 46.206896551724135%
- Rio de Janeiro: 13.448275862068964%


The word 'dreams' is distributed among the four categories as follows:
- Dubai: 7.317073170731707%
- New York City: 90.2439024390244%
- Paris: 0.0%
- Rio de Janeiro: 2.4390243902439024%


The word 'love' is distributed among the four categories as follows:
- Dubai: 1.6129032258064515%
- New York City: 4.032258064516129%
- Paris: 87.09677419354838%
- Rio de Janeiro: 7.258064516129033%


The word 'concrete' is distributed among the four categories as follows:
- Dubai: 0.0%
- New York City: 100.0%
- Paris: 0.0%
- Rio de Janeiro: 0.0%


The word 'rich' is distributed among the four categories as follows:
- Dubai: 89.28571428571429%
- New York City: 5.357142857142857%
- Paris: 1.7857142857142856%
- Rio de Janeiro: 3.571428571428571%
```

We see that some of these words have very strong associations with specific cities compared to others. So we can potentially choose just these words in our bag-of-words features and potentially get a greater accuracy over accounting for every single one. To make the choice of which words to

choose specifically (we cannot just use 5 words as above), we created a model for some test data and looked at the conditional probabilities of the words, notably ones that had a 5% chance or greater of being present given a certain category. Below is an excerpt of the words from each category (cropped to show some of the more interesting words):



We then cherry-picked words that were meaningful units (as described previously, nouns), and formed a custom vocabulary. Now instead of looking at the entire quote, we treated the presence of a specific word as an indicator of which category it might belong to. This resulted in us using the final vocabulary of words below:

```
dubai, rich, city, rio, love, dreams, paris, york,
football, tower, concrete, money, baguette, brazil,
carnival, jungle, eiffel, habibi, oil, big, apple
```

We test the change in accuracy later in this report to check the effectiveness of a reduced vocabulary.

Input Features & Feature Encoding

From the data exploration section above, we decided to include Q1-Q7 along with certain vocabularies from Q10. Each survey question is encoded as follow:

Q1: Numerical **popularity** feature, integer from 1 - 5.
Q2: Numerical **social media viralness** feature, integer from 1 - 5.
Q3: Numerical **architectural uniqueness** feature, integer from 1 - 5.
Q4: Numerical **enthusiasm for street parties** feature, integer from 1 - 5.
Q5-1: 1 iff respondent would bring their **partner**, 0 otherwise.
Q5-2: 1 iff respondent would bring their **friend**, 0 otherwise.
Q5-3: 1 iff respondent would bring their **siblings**, 0 otherwise.
Q5-4: 1 iff respondent would bring their **co-worker**, 0 otherwise.
Q6-1: Numerical **skyscraper relatability** feature, integer from 1 - 6.
Q6-2: Numerical **sport relatability** feature, integer from 1 - 6.
Q6-3: Numerical **art and music relatability** feature, integer from 1 - 6.
Q6-4: Numerical **carnival relatability** feature, integer from 1 - 6.
Q6-5: Numerical **cuisine relatability** feature, integer from 1 - 6.
Q6-6: Numerical **economic relatability** feature, integer from 1 - 6.
Q7: Numerical **average January temperature** feature, real number.
Q10: Bag of words, 1 iff the corresponding word is used in the **quote**, 0 otherwise.

Note: In the event of an invalid/missing entry, the value of the response is set to 0. All the questions do not allow for the value 0 to be a response (except for Q7). Thus, the value 0 is especially reserved for invalid/missing entries. This may in turn enable the model to learn even from partially filled surveys and make predictions with them as well.

Splitting the Data

For model exploration and training, we decided to split the data into 3 sets. We allocated 64% of the entire data for the training set, 16% for the validation set, and 20% testing set, as per convention. The splitting of the data is done randomly, and more importantly, stratified. Stratified splitting of the data will ensure proper and equal representation of each city in the training, fine tuning, and accuracy estimation of the model, reducing the bias we may otherwise accidentally encode.

Model Exploration

We explored various machine learning models in order to find the model that best fits the criteria of this machine learning challenge. Namely, we've explored KNN as it is straightforward to understand and has the ability to capture patterns and trends given a small dataset, Random Forest as they are universal function approximators with the capability of capturing patterns and trends in a easy-to-understand manner while reducing the variance, Naive Bayes as the model performs well given a small dataset especially when it contains text, and finally Multilayer Perceptron as it is capable of learning complex/non-linear relationships between features in a flexible fashion.

K-Nearest Neighbors

K-nearest neighbors (KNN) is used to predict the class given training data to work with. KNN decides which training data is the closest match to the test data by calculating distance, for example using Euclidean distance or cosine similarity. We looped through dozens of k-values starting at 1 and moving up. After looping through the k-values, we chose the k-value with the highest test score and kept that as our model hyperparameter. The k-value we found to be the best at fitting the test data is around 5-20.

Random Forests

Ensemble learning method that takes the final result of many decision trees and combines the results in regression tasks and selects the most common result in classification tasks. A greater diversity of decision trees is the backbone of every good random forest model. We used sklearn to find the hyperparameters for Random Forest. The number of decision trees range from 30-60 trees.

Multilayer Perceptron

Neural Networks are foundational to dynamic deep learning. The basic idea is to have a collection of neurons that take in input(s) and return an output. What neural networks can do, through backpropagation, is learn how best to fit the data, which is fundamentally different from every other machine learning model. Although the learning process can take a while, once a neural network fits to training data it fits almost perfectly. We used sklearn's MLPClassifier to find the proper hyperparameters. We decided on a single layer of 96 neurons with the activation function ReLU. Our hyperparameter for lambda after running sklearn was 0.0001.

Naive Bayes

The easiest application of a Naive Bayes model is sentence classification. So instinctively, we began exploration by using the quotes from Question 10 as a standalone feature. To provide further evidence for/against the selection of features from these quotes in the form of choosing specific words, the model was trained with two kinds of vocabularies: the full vocabulary created from the training set, and one that only uses hand-picked words that had higher conditional probabilities for one category compared to another. The condition of a word being picked was that its posterior probability was higher for one category than others (stronger indicator of one category), and that it was a meaningful unit (so a word like "the" which doesn't have meaning by itself would be excluded, and "concrete" would be included). This decision was made when we found studies that successfully increased accuracy and performance of models that were sparse, so we decided to experiment with this idea for our quotes as well.
We specifically look at BernoulliNB, which is a Naive Bayes model from SKLearn that assumes that the underlying distribution of our data is the Bernoulli Distribution. Because of this, we do not work with word counts, but simply whether a word is present or not. A key question when trying to explore this model was whether it could be applied to other features as well, since they carry valuable information. So we attempt to discover binary representations for each of them.

Any real-valued features like Question 7-9 were ruled out, since there was no suitable method to turn these into binary features. Questions 1-4 were ranking based features. Taking Q1 as an example, we could create five different features out of each of the possible options. If one of these columns is a one, then it represents the ranking of this question. The other columns are not useless, since they can be seen as conveying the information "was Q1 rated at this score?". Such a representation can still be seen as independent, as the person answering this question has to think whether each score is worthy to be put, so the decision of the final score is independent from the others in a sense.

Question 5 can simply be seen as four questions packed into one," Would you take X to this place?". These are clearly independent decisions as well. Question 6 involved ranking the relevance of certain words to the city in question. First, each word is independent of others, so they can be split into their own "ranking features" akin to Questions 1-4. Then we can use a similar approach where each ranking position has a feature of its own, giving us binary independent features from the data

of this question as well. So, the Bernoulli Naive Bayes model was tested on just the quote data, as well as these alternate representations to try getting a model which would yield satisfactory results.

Model Choice & Hyperparameters

Hyperparameter Tuning: KNN

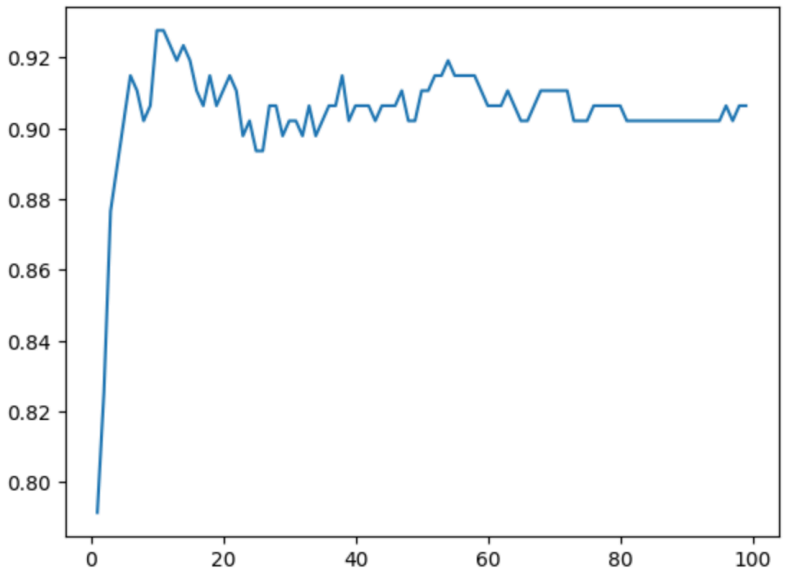
Prior to tuning the hyperparameter, K, we first started by normalizing each input feature. Doing so allows for proper and equal representation of each feature, ensuring one feature does not overpower another. Another benefit to normalizing the data for KNN is the improvement of distance accuracy. Normalizing the data ensures that the distances calculated reflect similarities in the geometry of the data space rather than the differences in the scale of dimensions.

After normalizing the features, we shifted our focus on tuning the hyperparameter K. We tuned the hyperparameter using SKLearn's KNeighborsClassifier model by iterating through different K values, ranging from 1 to 100, recording the training, validation, and test accuracy each time. Repeating the previous step multiple times revealed that the optimal value for K ranges from 1-15 based on the best validation accuracy at each iteration. With the range of optimal K values, we conduct 10 tests for precision, each with randomized data and the best K to fit the data. We discovered that the average validation accuracy of the model is 83% with the average testing accuracy at 80% across the 10 tests.

Hyperparameter Tuning: Random Forest

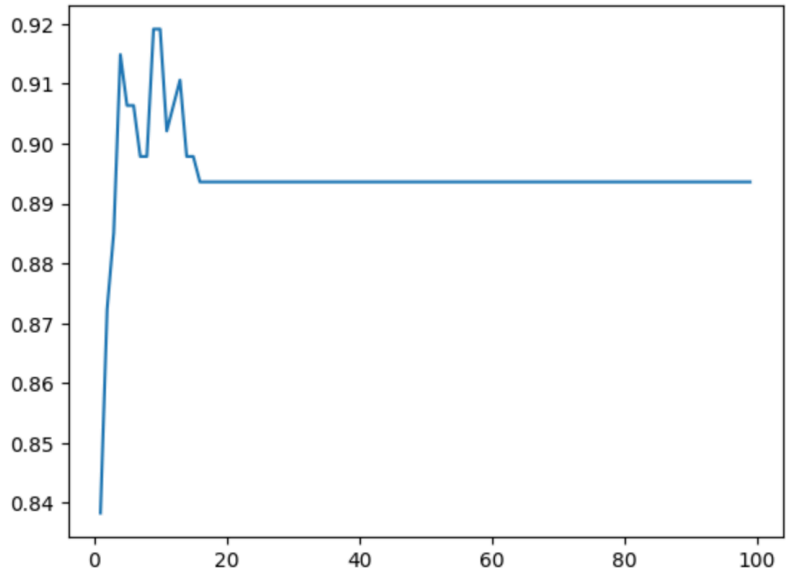
For the Random Forest Model, we've chosen to narrow our focus down to two hyperparameters: n_estimators (number of trees in the random forest) and max_depth (the maximum depth of each tree). We believe these two parameters are the most important in the model as n_estimators has control over the stability and accuracy of the model, while max_depth controls the complexity of the model, deciding the quality of each tree in the forest.

We began by first tuning n_estimators. We conducted multiple tests, each test iterating through different values of n_estimators (1 to 100) to find the range of the most optimal choice. After many tests, we observed the following pattern consistently:



Looking at the graph above illustrates that early on, as n_estimators increases, validation accuracy increases. However, we can see that the model peaks when n_estimators is between the 7-25 range, then it does not show a constantly increasing or decreasing trend. This is most likely the effect of overfitting, which fails to give us further improvements in accuracy. Thus, we've concluded that the optimal range of n_estimators must be around 7-25.

To tune max_depth, we decided to fix n_estimators at 16 (centered on the 7-25 range) in order to isolate its effect and iterate through values of max_depth from 1 to 100. After conducting the test we see the following trend:



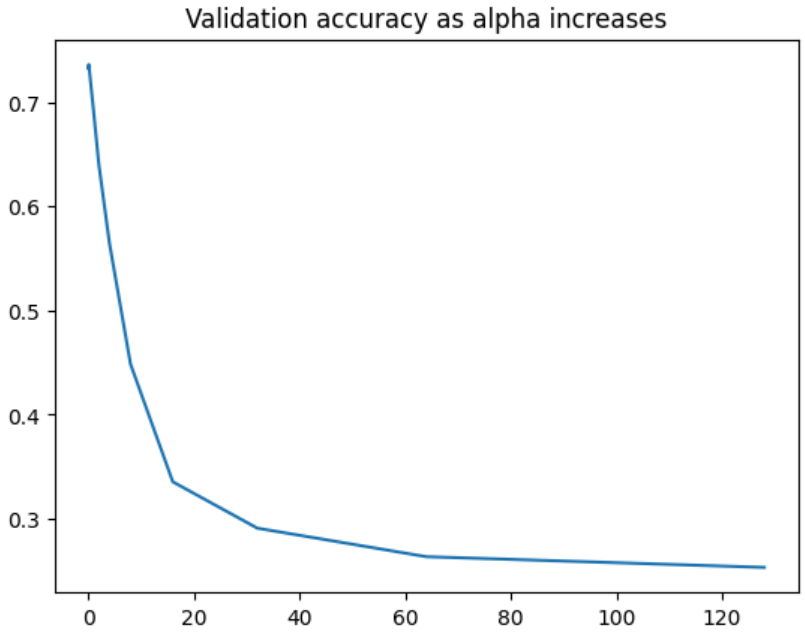
The validation accuracy peaks when max_depth is between 5-15, then becomes constant after 20, which is an indicator of overfitting. Thus, we conclude that the optimal value range for max_depth must be around 5-15

Given the fine tuned ranges of our hyperparameters, 7-25 for n_estimator, 5-15 for max_depth, we conducted 10 tests, each with randomized data and best fit n_estimator and max_depth. The test revealed that across the 10 tests, the Random Forest Model has an average validation accuracy of 92% and an average test accuracy of 88%.

Hyperparameter Tuning: Naive Bayes

Naive Bayes has one hyperparameter we can specify called “alpha”, which is an Additive Smoothing parameter. It can be any positive real number; SKLearn’s default is 1, it can be set to 0 to “turn off” the smoothing, and can be arbitrarily high or close to 0. The first consideration is “how high/low” do we want to go? As alpha grows, the posterior distribution for missing information approaches a uniform distribution, so we can keep increasing alpha till we converge. SKLearn’s documentation recommends that we shouldn’t be too close to 0, so we decided to limit ourselves to 0.01.

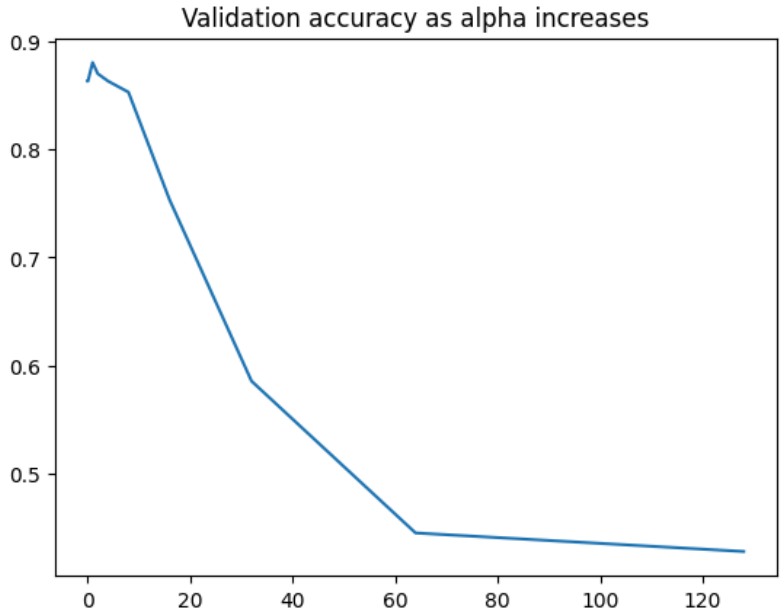
Our process of finding the optimal alpha value was to loop over all possible alpha values as defined by us and choose the one that maximizes validation accuracy. Then, we take an average of validation and test accuracy using 10 BernoulliNB models that are trained on different randomizations of data to showcase what to expect for realistic performance (the test set is kept separate from the training and validation data so it is not included in this randomization and treated as our “unseen data”).



Starting with just our quote data, we observe that we benefit from our alpha being small. This can be explained with an example. Suppose that the word “dubai” is a feature. We observed that it strictly only occurs in data points that would be categorized as Dubai, and would be 0 in any non-Dubai data points. In such a scenario, we would not want to artificially inflate the counts of “dubai” occurring in these other data points.

To be representative of how a Bernoulli Naive Bayes Model would perform on truly unseen data, we tested the validation and test accuracy using a vocabulary formed only from the training data. While in the real world we would have the ability to capture all words, in our case it is more appropriate to assume that the quotes in the data we will be tested on may have some words never seen before. Using just the quotes, we obtained an average validation accuracy of 72.84%, and a test accuracy of 68.8%.

We also attempted to use this model with Questions 1-6 binarized. The process of this has been explained in the Model section, where we specified how we could apply Naive Bayes to features other than the given quotes. Below is the trend in validation accuracy with changes in alpha:



One thing to be careful about was to check whether the optimal alpha stays the same with all these new features. It did in fact change, to 1. This is plausible because our features don't just represent "presence of words anymore", some of them are different now. Their effect was also significant, giving us an average validation accuracy of 88.53%, and an average test accuracy of 83.40%. Note that there was some variation in the optimal value of alpha, with the most common values being 1 and 0.1. Since 1 is the default, we stuck with it.

Lastly, we used our reduced vocabulary to check for improvements in accuracy. This would help with making stronger inferences as well as serving as a form of dimensionality reduction. We observed a validation accuracy sitting at 90%, and the test accuracy at 88%. So we decided that it would be beneficial to use this smaller vocabulary when taking useful data from the words provided by Question 10.

Hyperparameter Tuning: MLP

For neural networks, there are several hyperparameters for us to consider, such as the number of hidden layers, the number of neurons per hidden layer, the activation function, and the L2

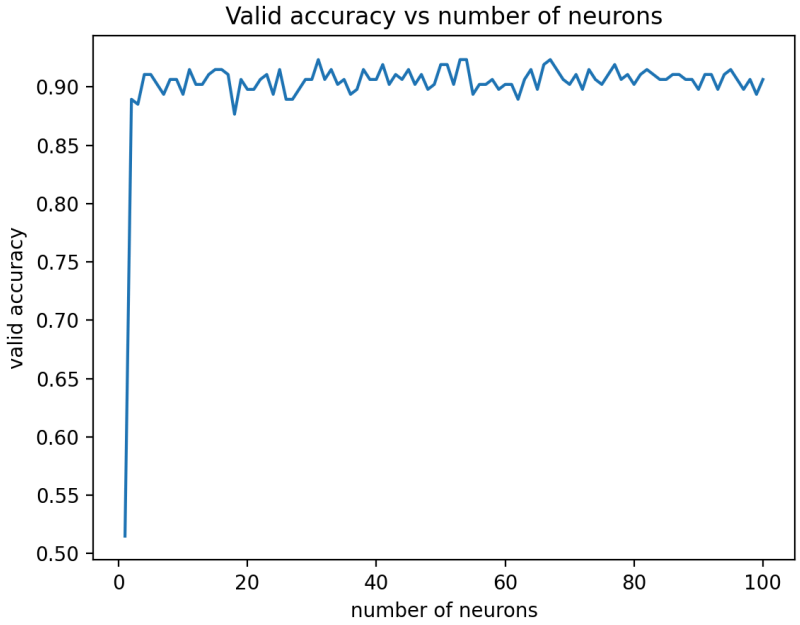
regularization strength (lambda). When tuning our model, we decided to fix the model at one hidden layer, tuning just for the number of neurons in the hidden layer, the activation function, and the L2 regularization strength. The reason we believe one hidden layer is appropriate is because of the Universal Approximation Theorem, which states that "a neural network with one hidden layer can approximate continuous functions on compact sets with any desired precision". Considering that our features are relatively small in size compared with other MLPs, one hidden layer seemed to be suitable following the theorem.

The next place we looked to tune was the activation function to use, being able to choose from the sigmoid, tanh, and ReLU functions. For each activation function, we tested its validation and test accuracy by iterating through different hidden unit sizes (10, 20, …, 100) and averaging the best validation and test accuracy through 10 runs. It is important to note that at each test iteration, all activation functions are using the same training, validation, and test set to ensure a controlled experiment.

```
### Average accuracies over 10 test ###
Average valid relu: 0.9072340425531914
Average test relu: 0.8935374149659863

Average valid tanh: 0.916595744680851
Average test tanh: 0.9040816326530612

Average valid logistic: 0.9025531914893618
Average test logistic: 0.88843537414966
```

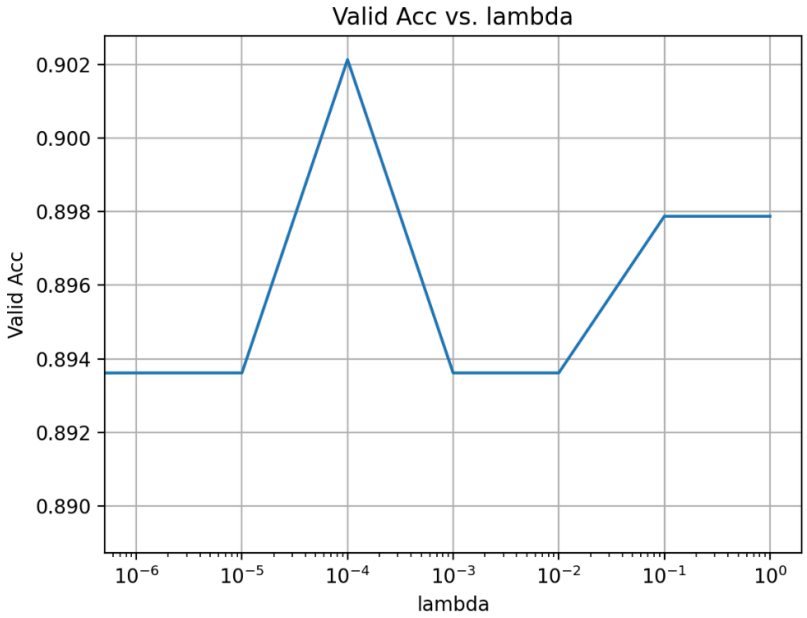
Looking at the result as shown above, we see that tanh has the highest valid and test accuracy averaged over 10 tests. Thus, we've chosen tanh as the activation function for our MLP.

After selecting tanh as the activation function, we began tuning for the hidden unit size. To tune for the optimal hidden unit size, we decided to test the model by varying hidden unit sizes, ranging from 1 to 100.



Looking at the result above, we see no discernible pattern. Thus, to preserve precision without making any incorrect assumptions, we've decided to keep the optimal value of hidden unit size as a range from 1 to 100.

Last, we tuned the L2 regularization strength or lambda, which dictates how much the model penalizes large weights. Large weights can lead to overfitting and decreased numerical stability. By default, the MLPClassifier has lambda = 0.0001, so we experimented with lambda ranging from 10^{-6} to 1, and this is what we found:



This shows that the default lambda at 0.0001 is optimal for our model, and lower values of lambda could've led to overfitting while larger values underfitting, reflected by their lower validation accuracies.

Combining all the tuned hyperparameters above, we are able to get an average validation accuracy of 91% and test accuracy of 89% over 10 tests with randomized datasets.

Model Choice

Looking at the hyperparameter tuning sections above, we have the average test accuracy of each model. Namely, we have KNN with 80% test accuracy, Random Forest with 88% test accuracy, Naive Bayes with 88% test accuracy and MLP with 89% accuracy. Before choosing MLP, we've decided to run the models using the same data sets (training, validation, and test) to ensure that MLP actually performs better than the other models, and not simply due to chance.

```
### Test accuracies given the identical train, validation, and test set ###
KNN test accuracy: 0.8537414965986394
Random Forest test accuracy: 0.8809523809523809
Naive Bayes test accuracy: 0.87
MLP test accuracy: 0.891156462585034
```

The results confirmed MLP was the best choice to make for us. Another justification for the choice of MLP is in its simplicity to implement. Although implementing the model to train is difficult, here we are only making predictions. So the process is relatively straightforward, we used SKLearn to train the weights and biases of the model, exported them to weights.npy and biases.npy, and used them in pred.py to make predictions. The predict function follows the normal flow of a MLP model:

1. Apply first layer weights and biases on the input layer to produce a pre-activated vector
2. Apply tanh to the resulting vector in the previous step to produce an activated vector
3. Applying second layer weights and biases to the activated vector
4. Perform softmax on the previous step's vector to make a prediction.

Note: Feature encoding is done as described in the earlier section.

Predictions

We were able to train and fine tune a MLP model with the following accuracies:

```
### Final model performance ###
Best number of neurons: 100
Best train accuracy: 0.9488817891373802
Best valid accuracy: 0.9191489361702128
```

```
Best test accuracy: 0.9115646258503401
```

We are confident that the model will perform with 91% accuracy as the model was trained using the test set and fine tuned with the validation set, meaning it has never seen the test set. Furthermore, since the data is split in a stratified random fashion, we are confident that the model will have equal precision identifying all cities (Dubai, Rio de Janeiro, New York City, and Paris).

Another thing to mention is that our model does not waste any data by dropping data points with null entries, which means we are as resourceful as we can be with the input to make the best prediction possible. So, no leeway was left for data to be left unused that could help us make predictions (with the exceptions of Questions 8 and 9 whose exclusions we were able to adequately justify).

Workload Distribution

Chun-Kai Chen

Perform data exploration (Q1-Q9), model exploration (KNN, Random forest, and MLP), hyperparameter tuning for all models, and the completion of pred.py. Contributed in data exploration, model exploration, model choice & hyperparameters, and prediction sections of the report.

Arush Awasthi

Data exploration for Q10, model exploration (Naive Bayes), contributed in data exploration of other questions, hyperparameter optimization & selection, model choice, statistical analysis and feature selection decisions.

Shihui Lu

Contributed to data explorations, performed model explorations and hyperparameter tuning for MLP, and helped with the hyperparameter section of the report.

Vijay Sai Patnaik

Worked on the kNN model. Wrote the Model section of the documentation.