CSC311 Machine Learning Challenge Report*

Siqi Fei Runshi Zhang Mark A Stevens Adelina Patlatii

April 8, 2024

Machine learning models are largely dependent on the types of features they use. The survey's mix of categorical, continuous, and free-response features meant that our model had to be versatile, but still powerful. We investigated Decision Trees, MLPs, and Logistic Regression. We decided to use a multi-class logistic regression model to meet these needs. predict future inputs. We tuned our model and found the ideal to be an inverse-regularization strength of c=1 and n-iter=60. We predict our model will have an accuracy of 90.5%. This prediction came from our model's performance on the test dataset, which had never been seen by any of the models until all hyper-parameter tuning had been completed.

Table of contents

*Code and data are available at: https://github.com/FXXFERMI/CSC311_ML.gitt.

1 Introduction

Our dataset comes from the lecture survey. We wanted to explore what families of models would perform best on this dataset. We investigated our models using the sklearn package, which allowed for quick yet relevant modifications to our models.

This paper is organized into four sections: Data, Model, Prediction and Discussion. Our data section explores the distributions of data, and describes generally how we chose to represent the data for our models. The Model section discusses how our data was modified and what assumptions we made on our data to apply 3 different models. Also, our Model Choice & Hyper-Parameter Tuning section describes our findings from applying these 3 models, and how we chose our final model. Our Prediction section discusses how we expect our model to perform in the future, and a small discussion on the types of data-sets we expect our model to be proficient or deficient in predicting.

2 Data

2.1 Data Measurement

To handle irregular inputs for a given datapoint, we chose to normalize the data. We could have chosen to remove the entire datapoint, or remove the irregular feature from the datapoint. Because our model will be applied on future datapoints which may also have irregular inputs, we chose to normalize, so that our model will have some “experience” handling this type of input.

A further description of the mechanism of normalization is included in the description of each question. We chose to use the median for discrete/categorical inputs, the mean for continuous inputs, and question handling for the questions involving ranking and free-response.

- Q1-4: Median
- Q5 & Q6: Question handling
- Q7-9: Mean
- Q10: Question Handling

2.1.1 Q1

Question 1 involved assigning a popularity score to each city, ranging from 1 to 5. The graph below illustrates the distribution of scores for each city.

The vast majority of responders have assigned a maximum score of 5 to the cities Paris and New York, while the cities Dubai and Rio de Janeiro were perceived as less popular. The response to this question constituted one of our features. Since the raw data is in a format which can be used directly to train the model, no further modifications to the representation of this feature were performed. As mentioned before, the outliers were replaced with the median of the responses, as the popularity feature is categorical - it is one of the numbers in the set $\{1, 2, 3, 4, 5\}$. A mean output could return a non-discrete output, which would cause the normalized datapoint to still vary from a typical datapoint.

Furthermore, we won’t have outliers due to extreme values in this question since it is a categorical question.

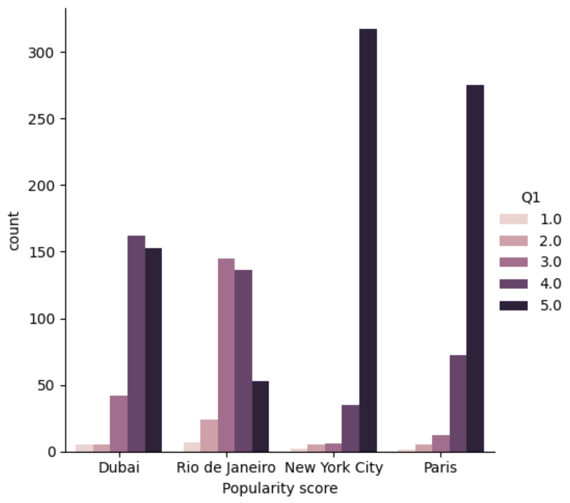


Figure 1: Question 1 Data Distribution

2.1.2 Q2

Question 2 asked the participants to rate how efficient each city is at turning everyday occurrences into potential viral moments on social media, on a scale from 1 to 5.

The Figure 2 below illustrates the correlation between the popularity feature and the feature referring to a city's likelihood of turning an everyday occurrence into a viral moment. Note the positive correlation between the two features, which can be explained by the fact that responders who perceived a city as more popular (a larger score in Q1) are more likely to assign a higher score to the city's efficiency at making everyday occurrences more known to the media (large Q2). The correlation coefficient between these two features was found to be $0.45 > 0$, indicating a positive correlation as expected.

According to Figure 3, only the responses for New York show a distinct trend, with the majority believing that the city is likely to turn daily occurrences into viral moments. Given the similarity between the graphs for the other three cities, we believe that this question did not contribute significantly to the model's predictions. Just like question 1, this question constituted its own feature, and no additional reformatting was performed on it, as it could be used directly for the model's training. We replaced the missing inputs with the median, as this feature is categorical.

Because this is a categorical question, we won't have outliers caused by extreme values.

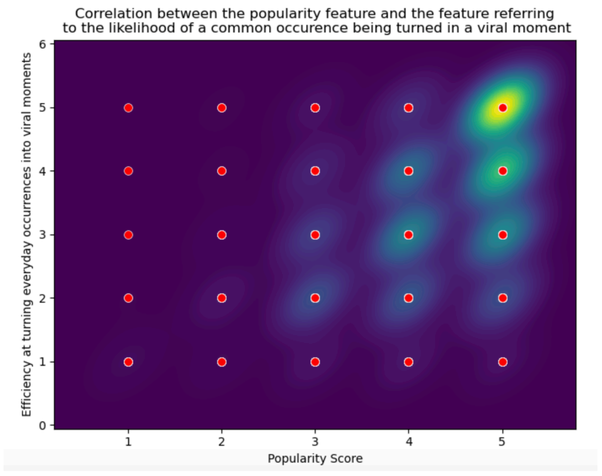


Figure 2: Q2 Correlation

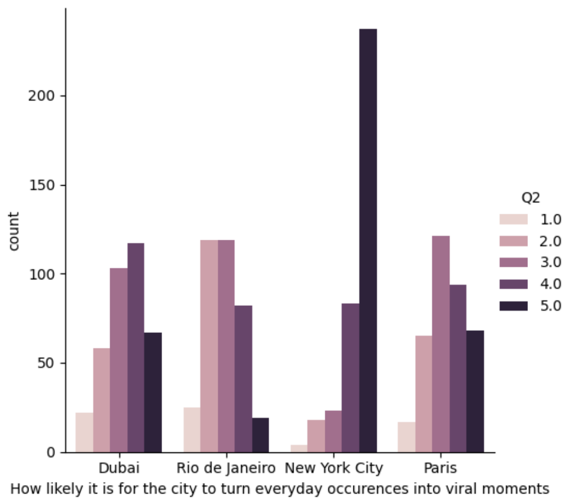


Figure 3: Q2 Count of Responses

2.1.3 Q3

Question 3 asked the participants to rate the architectural uniqueness on a scale from 1 to 5, with 5 indicating a perfect blend of futuristic wonder and historical charm.

From Figure 4 we can see that Paris stands out with a strong preference towards the highest score of architectural uniqueness, suggesting a consensus on its distinctive blend of historical and futuristic architecture. Dubai shows a trend towards higher scores as well, indicating that many people see its architecture as unique, but there's less agreement on the highest rating compared to Paris. New York City has a more balanced distribution of scores, with no single score overwhelmingly favored, suggesting a mixed opinion on its architectural uniqueness. And, Rio de Janeiro has the fewest high scores, indicating that fewer respondents find its architecture to uniquely blend the futuristic and historical, compared to the other cities surveyed. Similar to prior questions, this question can be directly incorporated as an individual feature within the model without the need for transformation. We replaced the missing values with the median, as this feature is categorical.

Furthermore, we won't have outliers in this question since it is a categorical question.

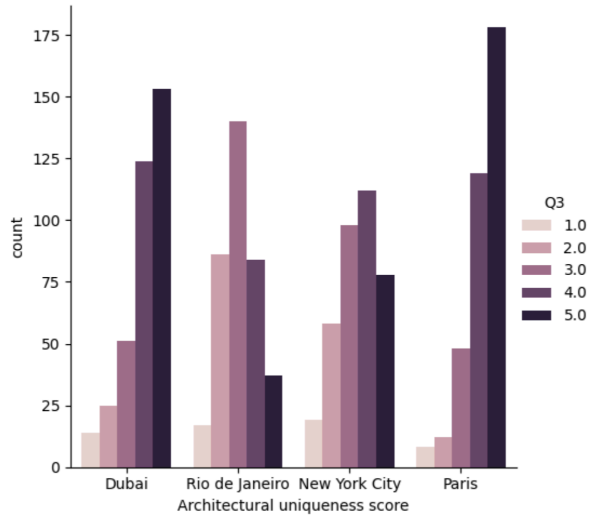


Figure 4: Q3 Count of Responses

2.1.4 Q4

Question 4 asks the participants to rate the city's enthusiasm for spontaneous street parties on a scale of 1 to 5, with 5 being the life of the celebration.

According to Figure 5 below, Rio de Janeiro stands out with the highest enthusiasm for spontaneous street parties, while Paris and New York City are perceived to have moderate enthusiasm, with Dubai leaning towards the lower end of the scale. As Paris, New York City, and Dubai show similar patterns in enthusiasm, this might indicate that for these cities, the question contributes less to differentiating between them in a predictive model. However, for Rio de Janeiro, the question shows significant variance, indicating it may be a strong predictor for this city in a model. To be mentioned that, in this specific context, high values for Rio de Janeiro reflect true cultural characteristics rather than an outlier. The city's enthusiasm for street parties can be added to the final data set as an individual feature without any further data processing, for the same reason as the first three questions. Its missing values were replaced with the median.

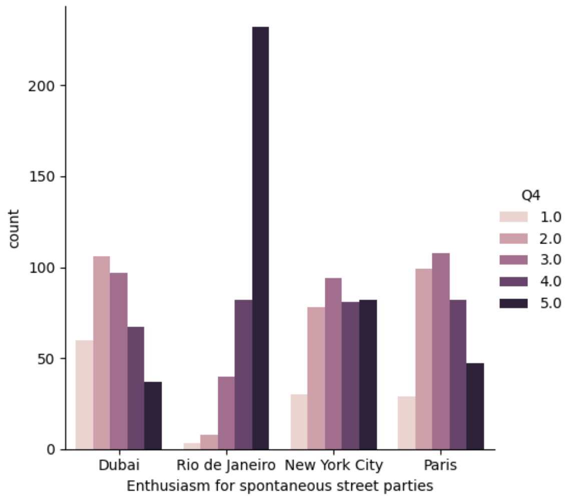


Figure 5: Q4 Count of Responses

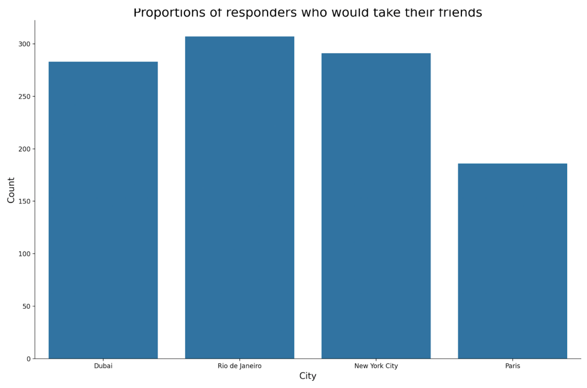
2.1.5 Q5

Question 5 asked us to think of the person who would most likely be with us if we were to travel to a particular city. We decided to represent this question as 4 features: a "friends" feature, a "partner" feature, a "siblings" feature, and a "co-workers" feature. If a responder i selected a certain person corresponding to feature j to be taken on a trip to a given city, then the entry in the data table on the ith row and jth column is assigned the number 1; otherwise, the entry is assigned number 0.

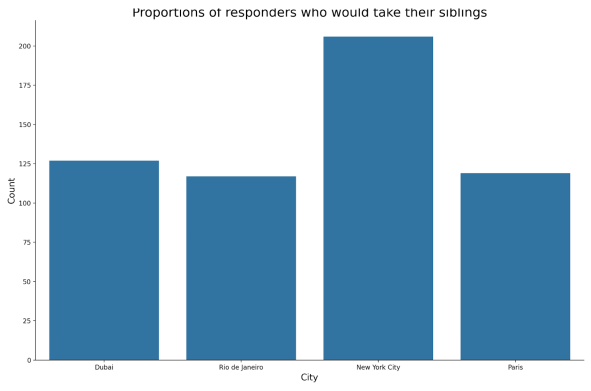
To handle missing values, we first calculate the majority value for each feature across the entire dataset.

For example, if the majority of responders include 'Friends' in their answers, then we will use 1 to represent the 'friends' feature as a replacement for missing values. Conversely, if the majority of responders do not include 'Siblings' in their answers, we will assign 0 to the 'siblings' feature to replace the missing values.

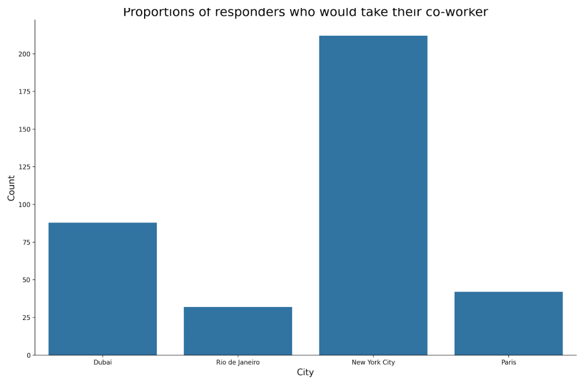
The Figure 6 below corresponding to the “co-worker” feature is characterized by high variance across the four cities, with a significant number of responders who would take their co-worker to New York City. As such, we believe that the “co-worker” feature will play a bigger role in the model’s training and final prediction. Worth noting are the number of responders who would take their siblings to New York City, and the number of responders who would take their partners to Paris. However, the graphs referring to the proportions who would take their friends, their partner, and their siblings to a given city show a relatively uniform distribution for three out of four cities, thus the model will likely not derive much information from the features “friends”, “partner”, and “siblings”.



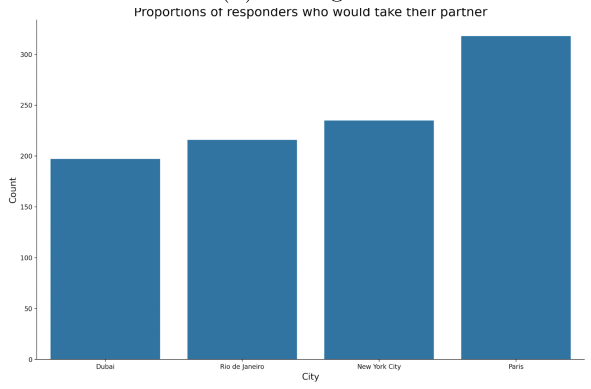
(a) Friends



(b) Siblings



(c) Co-Worker



(d) Partner

Figure 6: Question 5 proportion of responses

2.1.6 Q6

Question 6 asked "Rank the following words from the least to most relatable to this city. Each area should have a different number assigned to it. (1 is the least relatable and 6 is the most relatable)" with categories: **Skyscrapers, Sport, Art & Music, Carnival, Cuisine, and Economic**.

When exploring the data, we found that some individuals selected the same number for multiple categories. Thus, we chose to turn responses into a 6-dimensional vector, each value representing a response for that category. For example, a response of "Skyscrapers=>6,Sport=>4,Art and Music=>2,Carnival=>1,Cuisine=>3,Economic=>5"

Would become [6 4 2 1 3 5].

Similarly, some individuals would not respond for a certain category, but would respond to other categories. We've chosen to display any "non-responses" as a -1, but our mode will choose to normalize those values to the median for that category, as this is a categorical variable.

An initial approach we used for Q6 was to include 6 indicator variables. For example, a response of 6 in Art & Music would look like [[...], [...], [0, 0, 0, 0, 0, 1, 0], [...], [...], [...]], where an extra 0 was added to the end to indicate an invalid response. This approach was ultimately abandoned, as an indicator variable approach works better for responses which do not have a component of "intensity". An example would be white=1, non-white=2 for a survey response.

These graphs show the frequency counts of responses per ranking category.

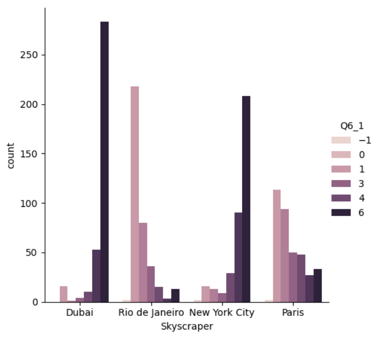
Figure 7a illustrates that a response of 6 in the "Skyscraper" category very likely means the response came from New York or Dubai, while a response of 1, 2, or 3 very likely means the response came from Rio or Paris.

Figure 7b illustrates that a response of 6 in the "Sport" category very likely means that the response came from Rio, while a response of 1 very likely means that the response did not come from rio.

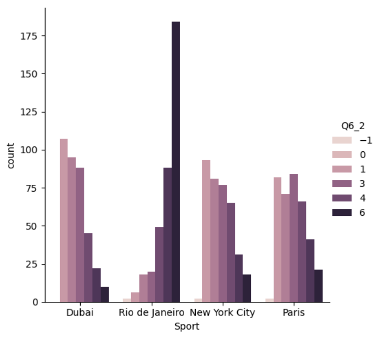
Figure 7c shows the perceived connection between "Art and Music" and Paris. A response of 6 means the response likely belongs to Paris, while a response of 1 means the response likely belongs to Dubai.

Figure 7d shows the connection between "Carnival" and Rio. We see that a response of 6 in this category most likely belongs to Rio, while a response of 1, 2, or 3 likely does not.

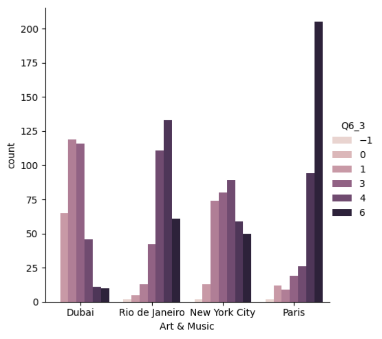
Figure 7e illustrates the connection between "Cuisine" and Paris. A response of 6 in this category very likely means the response came from Paris. It should be noted that Rio specifically had a high proportion of responses of 3 for this category. A response of 1 or 6 likely did not come from Rio, while a response of 2, 3, or 4, could have.



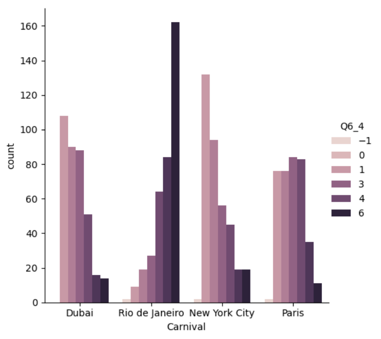
(a) Skyscraper



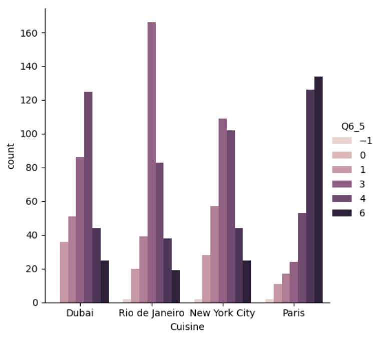
(b) Sport



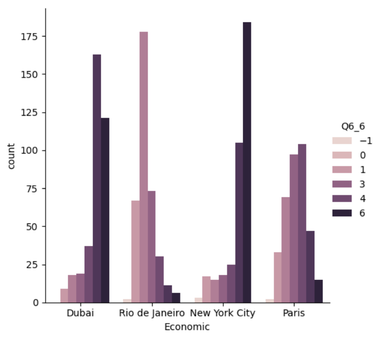
(c) Art and Music



(d) Carnival



(e) Cuisine



(f) Economic

Figure 7: Question 6 frequency counts of responses

Figure 7f shows the responses for the “Economic” category. A response of 6 likely came from New York or Dubai. It should be noted again that Rio had a very large number of responses for the 2 category. This means that a response of 6, 5, or 4 likely did not come from Rio, while a response of 1, 2, or 3 likely did.

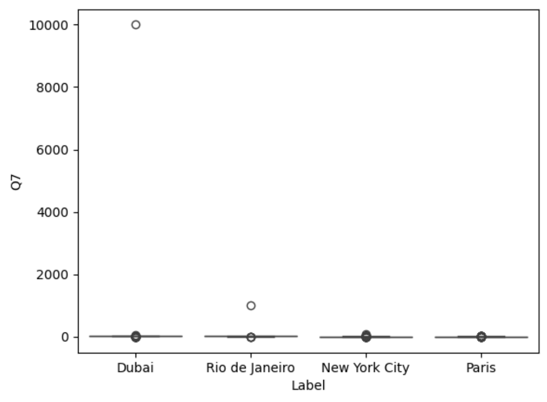
Ultimately, it should be noted that these categories were originally supposed to be a ranking, and not interpreted as a scalar. This ranking would have made responses mutually exclusive, and allowed for more creative interpretations of the data. That being said, this scalar approach still yielded powerful trends.

2.1.7 Q7

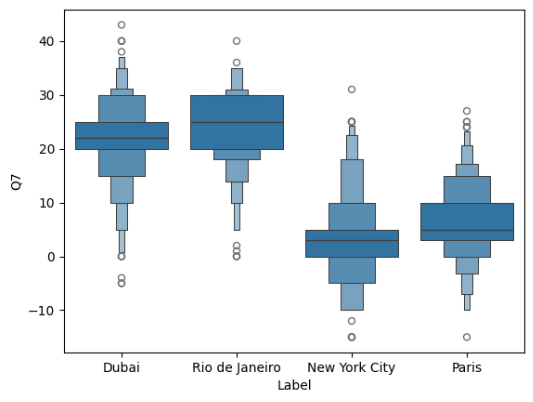
Question 7 asked “In your opinion, what is the average temperature of this city over the month of January? (Specify your answer in Celsius)”

Figure 8a illustrates the need for data normalization, as there are 1 or 2 datapoints which exceed 1000 degrees. We chose to change these data points to the mean value. This was done to preserve the distributions of the data. That being said, an alternative approach would be to “flatten” the data points to some maximum and minimum value. For example, we could have chosen to flatten any datapoint above 42 to just be 42, and any datapoint below -5 to be -5. This would have allowed some information to still be obtained from the datapoint, but was ultimately omitted to preserve the distributions of the data.

Figure 8b displays the distributions by city, after datapoints above 42 were removed. Note that this graph is different from how we actually chose to normalize the data. This graph shows us that Dubai and Rio typically have a higher perceived temperature than cities like New York or Paris.



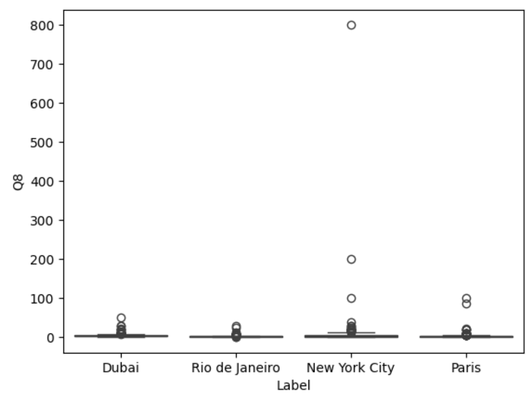
(a) Q7 Outlier



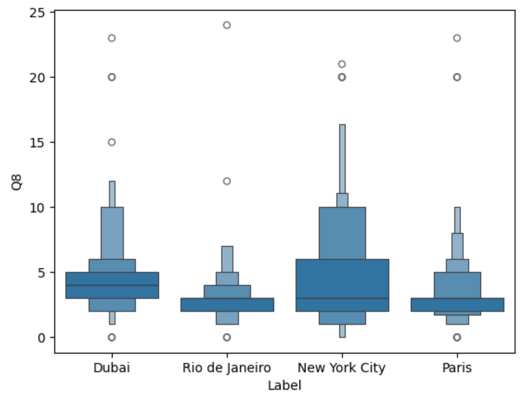
(b) Q7 Distribution

Figure 8: Question 7 Data Distribution

2.1.8 Q8



(a) Q8 Outlier



(b) Q8 Distribution

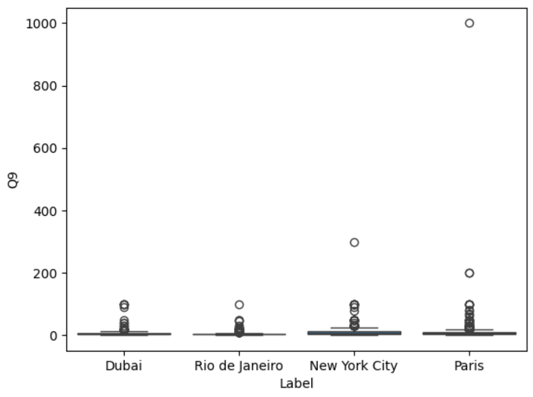
Figure 9: Question 8 Data Distribution

Question 8 asked “How many different languages might you overhear during a stroll through the city?”

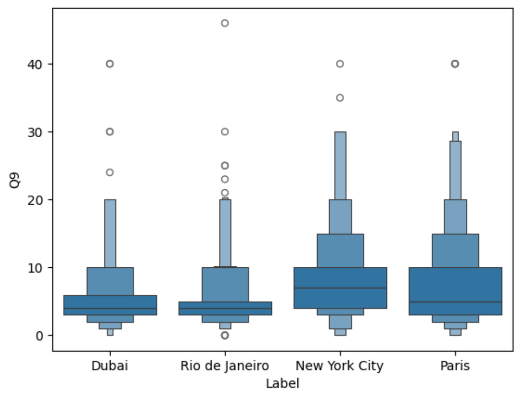
Figure 9a illustrates the need for data normalization, as there are many datapoints which exceed 100 languages. We chose to change these data points to the mean value, again to preserve the distributions of the data.

Figure 9b displays distributions by city, after datapoints above 25 were removed. Note that this graph is different from how we actually chose to normalize the data. This graph allows us to see that a response above 8 typically meant that the city did not come from Rio. However, all cities show significant overlap in their 1st and 3rd quartile ranges. This meant that this feature likely did not provide much insight for the model.

2.1.9 Q9



(a) Q9 Outlier



(b) Q9 Distribution

Figure 10: Question 9 Data Distribution

Question 9 asked "How many different fashion styles might you spot within a 10-minute walk in the city?"

Figure 10a illustrates the need for data normalization, as there are many datapoints which exceed 50 fashion styles. We chose to change these data points to the mean value.

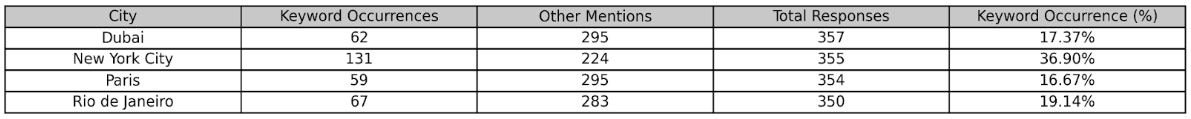
Figure 10b displays distributions by city, after data points above 50 were removed. Note that this graph is different from how we actually chose to normalize the data. This graph indicates that a response above 10 has a slightly larger chance of coming from New York or Paris, but ultimately, because the 25th and 75th quartiles for all cities overlap, it's likely that this feature held little significance in our models' predictions.

2.1.10 Q10

Question 10 asked: "What quote comes to mind when you think of this city?".

For representing this question, we decide to try 2 approaches:

- The first approach is similar to the representation of the movie reviews performed in lab 9. Each quote was split into its individual words. We collected the words present in the quotes into a vocabulary, and this vocabulary was used to create a “bag of words” matrix, where the entry on the ith row and jth column would be assigned the value 1, if the word on position j in the vocabulary appeared in the review i, and 0, otherwise. Consequently, question 10 was split into 1739 features, each corresponding to a word in the vocabulary.
- The second approach differs from the first approach only in the use of 16 of the words in the vocabulary, which we deemed most suggestive in describing the cities. These words are: “dubai”, ”ny”, “new york”, “new york city”, “rio”, “rio de janeiro”, “paris”, “c’est la vie”, “the city of love”, “eiffel”, “apple”, “football”, “soccer”, “rich”, “money”, and “burj khalifa”. This approach was motivated by the fact that the key words play the biggest role in determining whether a certain quote refers to a city. The table below illustrates the proportions occupied by the key words in the quotes assigned to each city:



City	Keyword Occurrences	Other Mentions	Total Responses	Keyword Occurrence (%)
Dubai	62	295	357	17.37%
New York City	131	224	355	36.90%
Paris	59	295	354	16.67%
Rio de Janeiro	67	283	350	19.14%

Figure 11: Q10 Keywords

As the first approach - involving all words in the vocabulary - led to a slightly better performance on the models we have explored, this was the approach with which we decided to move forward. As seen in the graph below, the words which likely also played a significant part in the models’ predictions were the more frequent ones: “the”, “of”, “city”, etc., rather than the keywords, which could explain why the first approach yielded higher accuracy measurements.

Just like in the previous questions, when processing the data for question 10, we had to take care of missing inputs. Our analysis on the frequency of words in the quotes revealed that the word “without” appeared only once. Since the word “without” is neither a key word, nor does it have a high frequency, it likely made a negligible impact on the models’ predictions. As such, we have made the decision of replacing all of the missing values with the word “without”.

Additionally, to prevent errors when the model encounters unseen words, we hard-code our X_train. This means it will only update the feature matrix based on the vocabulary list used during training, and will ignore any unseen words.

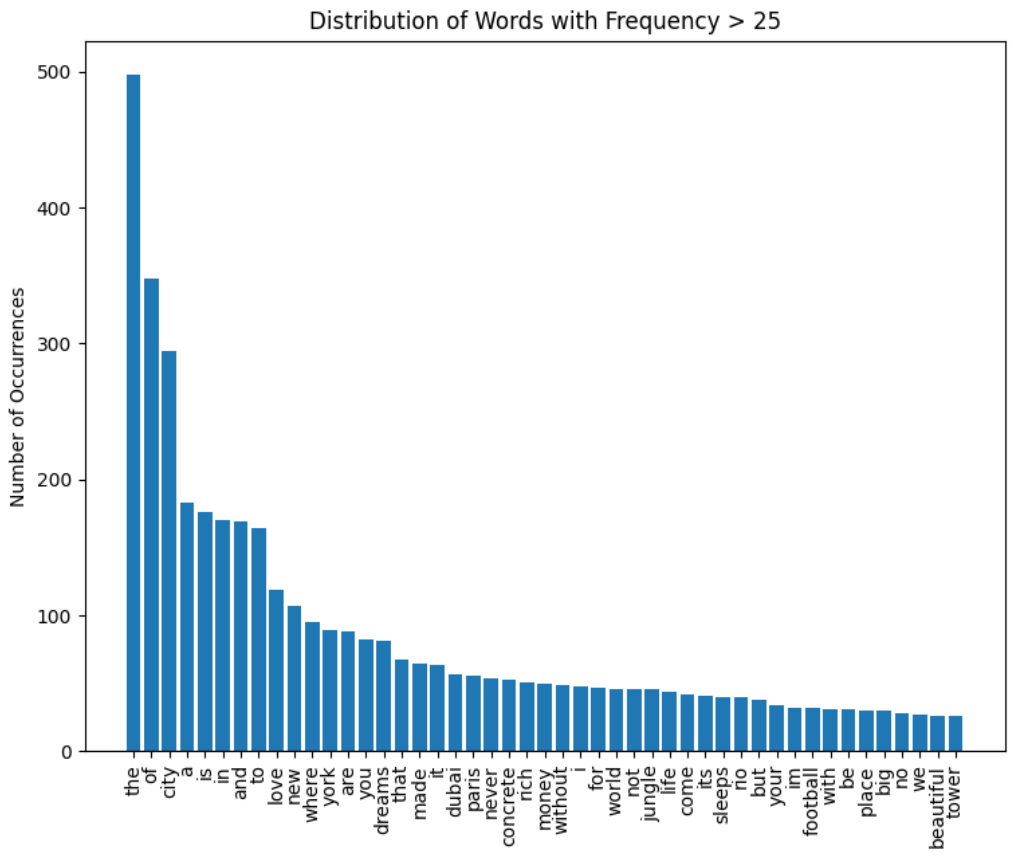


Figure 12: Q10 Bag of Words

2.2 Data Splitting

We chose to use an 80-10-10 training:validation:test split. 80% of our data points were dedicated for training, 10% for validation, and 10% for testing.

2.2.1 K-Fold Cross Validation

Since the dataset only includes 1439 data points, which is a relatively small size for training a Machine Learning Model, we also employed k-fold cross-validation. This technique helps us maximize the use of the available dataset by ensuring that every data point is used for both training and validation at least once.

The process involves the following steps:

- The original dataset of 1439 data points is divided into 'k' equal-sized folds. For example, with k=5, each fold would contain approximately 287 data points.
- The model is trained 'k' times, each using a distinct fold for validation data, and using all other 'k-1' folds as training data. This means that in each iteration, a different fold is held out for validation while the rest are used for training.
- After training the model, it is evaluated on the corresponding validation set. Performance metrics such as accuracy, precision, recall, or F1-score are computed for each iteration.
- Finally, the performance metrics from all 'k' iterations are averaged to obtain a more wholistic estimate of the model's performance. This average performance metric provides a more accurate assessment of how well the model generalizes to unseen data compared to just using a single train-test split.

K-fold cross-validation not only helps in utilizing the available data efficiently but also reduces variance and provides a more robust estimate of the model's performance.

3 Model

3.1 Model Exploration

All of our inputs are scalar values, and our output is a 4-dimensional one-hot vector. Note that some questions, like Q5 and Q10, use indicator variables, other questions use discrete scalars, like Q1-4 and Q6, and other questions use continuous scalars, like Q7-9. Thus, our approach must be able to handle all 3 of these data-types.

3.1.1 Feature Matrix

Through the above data exploration, we now have a Bag of Words with 1739 features, causing X_train to have 1755 features.

- Q1-Q4 each represents 1 feature: `Q1`, `Q2`, `Q3`, `Q4`
- Q5 be reformatted into 4 features: `Co-workers`, `Friends`, `Siblings`, `Partner`
- Q6 be reformatted into 6 features: `Skyscrapers`, `Sport`, `Art and Music`, `Carnival`, `Cuisine`, `Economic`
- Q7-Q9 each represents 1 feature: `Q7`, `Q8`, `Q9`
- Q10 be reformatted into 1739 features based on our Bag of Words

3.2 Decision Trees

We know that decision trees can perfectly model any distribution of data. We chose to investigate decision trees because of this fact. Decision trees are versatile in their ability to use both indicator and scalar features, and were investigated for this reason.

We assigned each city to a separate value from 1-4. This is appropriate for Decision Trees, as these values represent different classes, and not an actual value.

We converted the original categorical output of the decision tree (a value of 1-4), to a 4-dimensional one-hot vector.

3.3 MLP

Multi-layer perceptron models are incredibly versatile, and are able to handle both indicator variables and scalar variables. However, these models tend to be very sensitive to noise. This class of models was chosen to be investigated because of their ability to handle multiple types of inputs.

We chose to use an MLP classifier, as these models lend themselves to producing categorical outputs. Other MLP models could have been used if we wanted to treat our target vector as being continuous, but this would ultimately have led to worse predictions. We assigned each city to a separate value from 1-4, which our model interpreted as being 4 distinct classes.

We used a RELU activation function, as it generally works very well. We were cognisant of the values of weights, and did not notice any weights get "stuck" at a very low negative value.

3.4 Logistic Regression

Finally, Logistic Regression was investigated for its ability to maintain an error signal throughout the activations. That is to say, the model would continue to have an error signal, no matter the weight of the connections.

We chose to use a logistic regression model nested inside a Multi-Output-Classifier. This allowed the Logistic Regression to produce a categorical output instead of a scalar output. We represented our target values as a 4-dimensional one-hot vector.

3.5 Model Choice and Hyper Parameters

3.5.1 Implementation

We used a consistent training, validation, and test set for the hyper-parameter tuning, alongside a consistent ordering of datapoints. The datapoints were shuffled with a consistent seed, meaning that the datapoints were randomly assorted but consistent across comparisons.

This consistent comparison ensured that all models were given a "fair" chance to succeed, and that any subpar performance was not due to an unlucky or uninformative sampling of data.

All 3 families of models produce a single prediction as the output. This means that we were able to simply find the average number of correct guesses divided by the total number of guesses to compute their accuracy. We used a consistent validation set across all models to ensure the data was not especially "unfair".

We tuned hyper-parameters for each model to accurately compare their performances. We were able to compute the validation accuracy for each model, and used the validation accuracy as our metric for comparison. In general, a higher validation accuracy meant a better model. However, some choices were made to ensure our models would have predictive power and not be over-fit.

3.5.2 Decision Trees

We chose to tune the depth hyper-parameter. The graph includes the accuracy of models with different depths. We chose to use a depth of 3. The decision trees had seemed to converge at a depth of around 3, as the validation accuracy did not significantly improve after this depth. While depth 5 had a higher validation accuracy, we chose depth 3 to prevent any over-fitting.

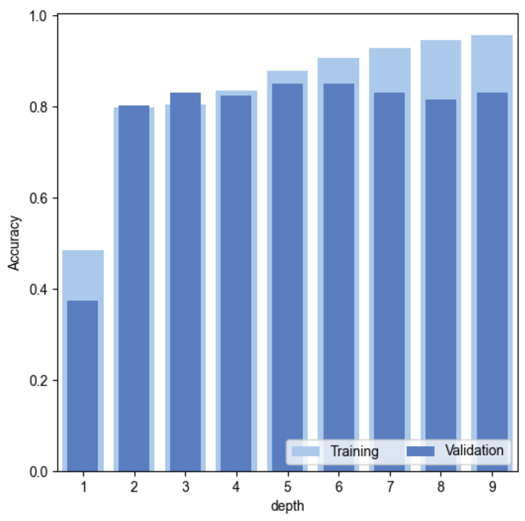


Figure 13: Decision Trees

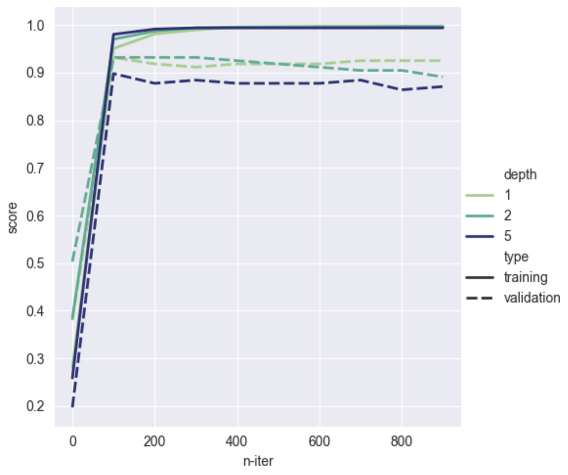
3.5.3 MLP

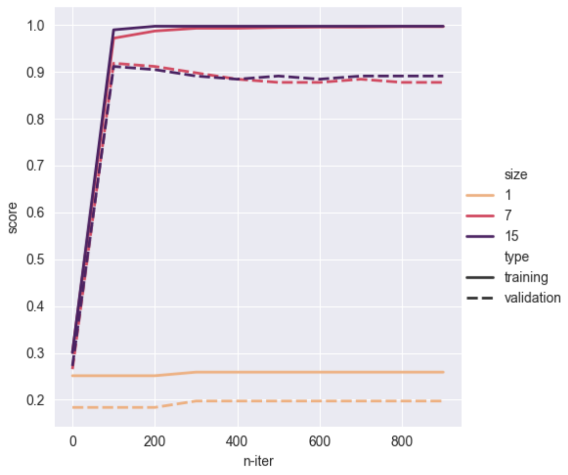
For the Multi-Layer perceptron, we chose to tune 3 hyper-parameters: depth, size, and number of iterations. Size represents the number of nodes in any hidden layer, and depth represents the number of hidden layers. We chose to have a consistent number of hidden nodes in every layer, but we could have chosen to vary the number of hidden nodes per layer. The number of iterations represents how many iterations of training we chose to do on the MLP.

We investigated sizes of [1, 2, 3, 4, 5, 15] and depths of [1, 2, 5]. We tested each possible combination of sizes and depths, but only chose to include a select number of these combinations for clarity. We also plotted approximately 10 datapoints per graph. That is to say, if n-iter has a max value of 1000, we checked n-iter of 1, 100, 200, 300, 400, etc.

Ultimately, we checked n-iter values of [10, 20, 30, ..., 90, 100, 200, ..., 900, 1000]. MLP allows for progressive training, so we took each possible combination of sizes and depths, and plotted the accuracy of the model at each n-iter value.

We first tuned the n-iter (number of iterations) hyper-parameter by assigning very large and very small values to the size and depth hyper-parameters. Doing so yielded the following graphs:

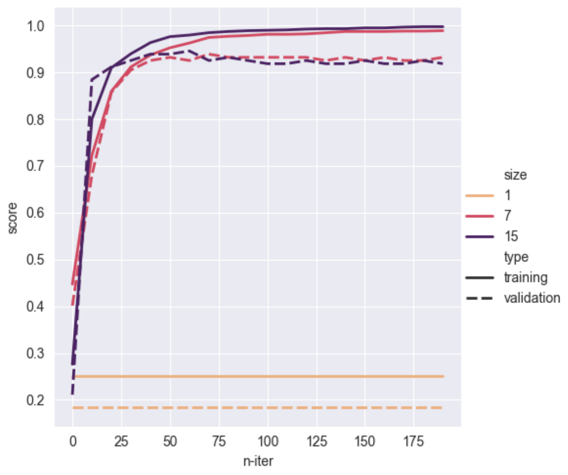


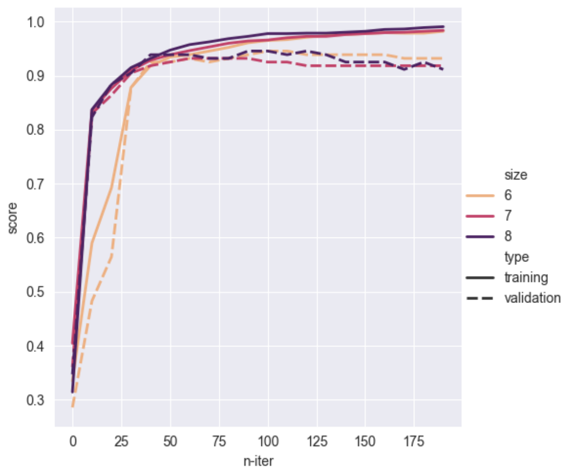


The left plot helps support our claim that regardless of the number of hidden layers, our model will tend to converge by 100 iterations. We used a hidden layer size of 7 in the graph, but tested with more hidden layers.

The right plot helps support our claim that regardless of the number of hidden nodes, our model will tend to converge by 100 iterations.

We then investigated the size parameter, and found the ideal to be 7.



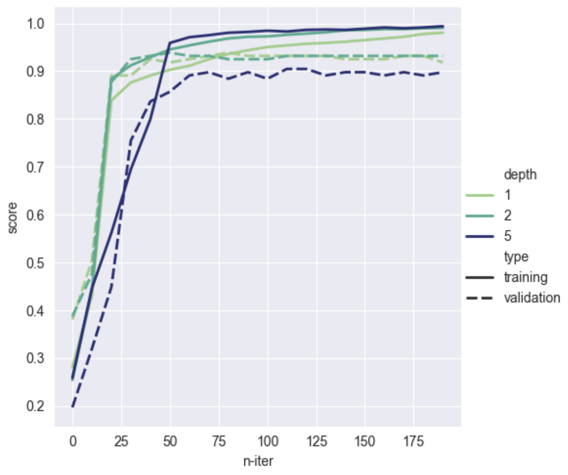


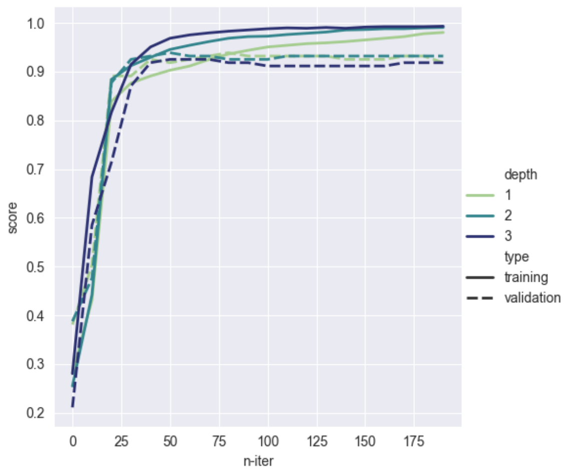
The left plot helps to illustrate that size values much below 7 result in very poor performance, while values much above 7 result in very similar performance.

The right plot helps to show that at 100 iterations, 7 seems to roughly be the point where an increase in the size of a hidden layer would not result in an increase in performance.

Ultimately, we chose to use a hidden layer size of 7, as this size is large enough to produce a high validation accuracy, but small enough to know that we are not overfitting our data.

Next, we tuned the depth of our model.





The left plot shows that depths much above or below 5 result in similar but worse model performance.

The right plot shows that depths of both 1 and 2 show very similar performance, while a depth of 3 begins the point where we stop seeing gains from increased numbers of layers.

We chose to use a hidden layer number of 2, as we believe it could accurately capture the complexity of the dataset, while not resulting in overfitting. We could also have chosen a hidden layer number of 1 as well.

We found that an MLP with 2 hidden layers, each of size 7, resulted in the best validation accuracy.

3.5.4 Logistic Regression

For Multi-class logistic regression, we chose to fit both the number of iterations the model used, and the inverse regularization strength.

When we used the sklearn package, we found that we could not progressively fit our model, and record its performance at different numbers of iterations. Our workaround was to use a "brute force" method, and train an entirely new model for every n-iter we wanted to investigate.

Because we used exactly the same training and validation sets, we found that any given model would perform exactly the same. Thus, our method for plotting n-iter is appropriate.

We also found that sklearn automatically stops training once it detects the model has reached convergence. This means that a value of n-iter of 1,000 would perform exactly the same as a value of n-iter of 10,000, as they would both reach convergence and stop iterating far before reaching 1,000 iterations.

This explains why you will see a flat bar, as sklearn has stopped iterating the model. We detected when sklearn reached convergence, and stopped plotting datapoints, resulting in a line seemingly cutting off. When setting max_iter to a value lower than sklearn has detected convergence, an error is produced. This error can be ignored, as it just indicates that the training has stopped before sklearn has detected the model reaching convergence.

The second hyper-parameter we chose to tune, C, corresponds to the inverse regularization strength. We know that a higher regularization strength correspond with increased weight penalization, which we believe will help to prevent overfitting. C has the inverse property. A lower value of C leads to more regularization, so less overfitting.

We chose to use a single-class logistic regression model to tune hyper-parameters. However, we found that a 4-dimensional vector was required for the output, and thus modified our model to a multi-class logistic regression model. We again performed hyper-parameter tuning, and found that the ideal hyper-parameters were similar for both models.

We chose to test C values of [0.1, 0.5, 1, 2, 3, 5, 10] at iterations {10, 20, ..., 190, 200, 300, 400, ..., 2000}

We first chose to explore n-iter at a large range of C values.

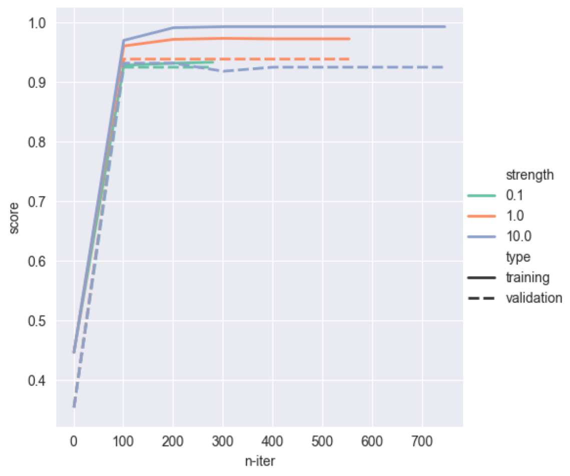


Figure 14: LP-1

Figure 14 supports our claim that converge in a logistic regression model is universally reached by 100 iterations.

This preliminary mapping informed our decisions on where we should look for convergence of our final model.

We then did a more fine-tuned exploration of n-iter around 100.

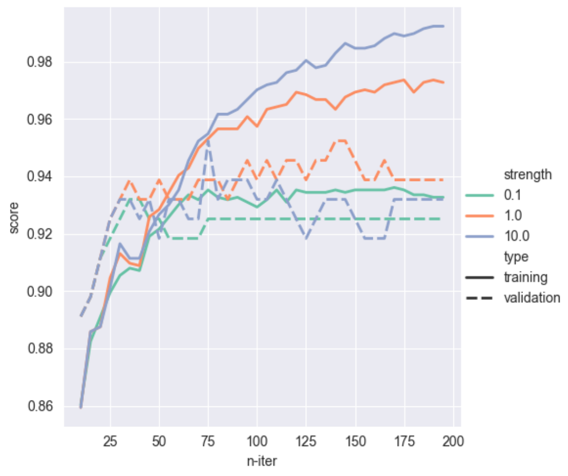


Figure 15: LP-2

Figure 15 supports our claim that convergence is reached by approximately 60 iterations for many values of C. Similarly, values much above and below C=1 result in lower accuracies than that of C=1.

Thus, we explored C values close to C=1

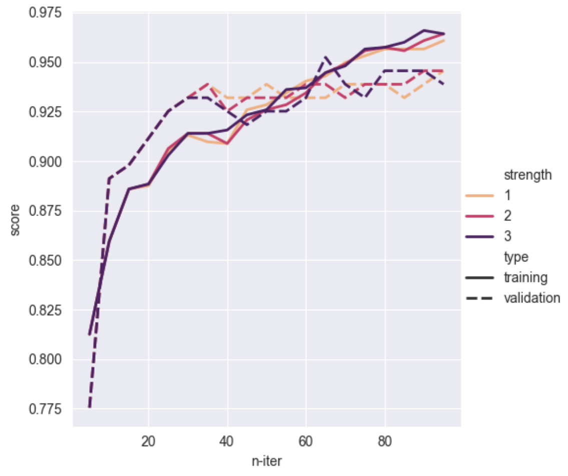
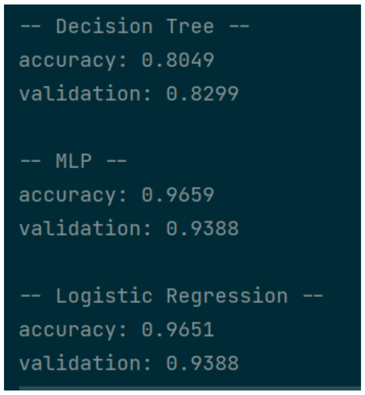


Figure 16: LP-3

Figure 16 supports our claim that c values near c=1 have very little if any impact on the performance of the model. While we chose to ommit the graph, values below c=1 showed accuracies below that of c=1. We chose to keep a c=1 value.

Ultimately, we found n-iter=60 and c=1 to be the optimal hyper-parameters for this logistic regression model. Note that c=1 is the default value, so is omitted from our final code.

3.6 Model Comparison



```
-- Decision Tree --
accuracy: 0.8049
validation: 0.8299

-- MLP --
accuracy: 0.9659
validation: 0.9388

-- Logistic Regression --
accuracy: 0.9651
validation: 0.9388
```

Figure 17: Model Comparison

We found Decision Trees produced the lowest validation accuracy, while MLP and logistic Regression tied to produce the highest.

Ultimately, we chose to use logistic Regression, as it was tied for the highest validation accuracy, but modeled a less-complex underlying trend than the MLP. We chose to use the model with less complexity to avoid overfitting.

In the future, it may be helpful to split the data into 4 sets: Training, Tuning, Comparing, and Testing. Essentially, the validation set would be split into two categories: Tuning and Comparing. The tuning set would be used to tune the hyper-parameters for all models, and the comparing set would be used as a 'pseudo-test set' to estimate the model's performance on a set of data it had never seen before. We would then use the actual test-set on the final model to predict its future performance.

3.7 Model Implementation

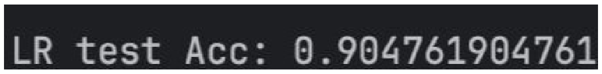
Briefly speaking, the model implementation consists of the following steps: we first import the optimal estimators obtained through the training and hyperparameter tuning of our model as a csv file. Then, the main function of our pred.py file - `predict_all` - cleans the data set in the way described by the "Data exploration" section of our report. Lastly, we generate predictions for each data point of the cleaned data, using the imported estimators. Here is an overview of the functions used in pred.py to achieve this:

- `predict_all(filename)`: This function first creates the data set on which the prediction is to be made. It loads the data and processes it, question by question, taking care of the missing values. Note that for the purpose of processing question 10, we introduced two other helper functions: `clean_text(text)`, which aids in the cleaning of the quotes, and `make_bow(data, vocab)`, which generates the bag of words matrix. The processed questions are merged together into a data set X_test_df, which will be used to make the prediction. As a final step, the function `predict_all` iterates through `X_test_df`, and calls the function predict to make a prediction for each data point. The predictions for each individual point is stored in a list predictions, which is returned as the final output.

- `predict(x)`: This function is responsible for predicting the most likely city for each individual data point. This function first generates the probabilities for the four cities using the helper function `pred(x, model_est)` called on the input data point and the imported estimators. Out of the obtained probabilities, we choose the largest one, and return its corresponding label.
- `pred(x, w)`: This function generates probabilities for an input data point. We multiply the weights of our estimators by the values of the datapoints. These weights are distinct for every city. We then sum all products together for each city, and return 4 values. These 4 values are converted to a probability distribution via the `softmax(z)` function. This probability distribution is returned.

4 Prediction

We estimate our model will have an accuracy of 90.5%. This estimate comes from the percentage of correct classifications of our test set. This test set has only been used at the very end of our project, meaning no training or hyper-parameter tuning involved this dataset.

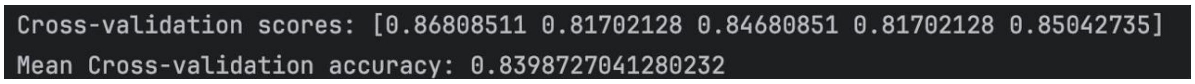


```
LR test Acc: 0.904761904761
```

Figure 18: LP Test Accuracy

4.1 Estimates on quality

It is important to note that our k-fold cross-validation indicates a mean-accuracy of 84%. This is **not** our point estimate of our model's performance, but should be taken into account when considering the accuracy of our model's performance.



```
Cross-validation scores: [0.86808511 0.81702128 0.84680851 0.81702128 0.85042735]
Mean Cross-validation accuracy: 0.8398727041280232
```

Figure 19: LP Cross Validation Accuracy

We expect our model to yield a high performance on data collected from the same source it was trained on. That is to say, we expect our model to perform well on data provided by students taking CSC311 at UTM.

We expect our model to perform poorly on data taken from a new source. Instructors and TAs may tend to respond differently to the provided questions than students, which could cause our model to produce an inaccurate prediction. A student who has never been to Dubai may estimate the average temperature to be 50C, while an instructor could have lived a longer time and gained more information, and produced an estimate of 35C. Trends drawn from student responses may not extend to trends from instructor responses.

5 Discussion

5.1 Future Improvements

We could improve our estimate by investigating further ML models, including Gaussian Discriminant Analysis, or Naive Bayes models.

We could combine all 3 of our models using an ensemble method. This would involve training each model (Logistic Regression, Decision Tree, and MLP) independently. Each model would "cast a vote" for a city, and we would predict the city which had the most votes. This method has been shown to be very effective when each model is very unique. Thus, having many different families of models could be an effective approach.

We could also use a technique called Mixture of Experts to combine all 3 types of models we've created. In theory, each model would independently be trained. We would then add an additional model (called a gating model), which would take the 3 predictions (alongside the original datapoint) as an input, and produce an output. This gating model is hoped to be effective at determining when a given model will perform well, and choosing to selectively listen to that output.

6 Workload Distribution

Initially, we thought we would be able to develop 4 separate models, and create a single hybrid model. We decided to split the workload in the following way: Adelina took up the data exploration for the first four questions, Rodney decided to explore the fifth question, Mark was responsible for questions 6 to 9, and Fermi took up the 10th question. The splitting was motivated by the types of the questions (i.e., categorical, numerical etc.), as well as how they would be represented in the final data set. As we were advised to switch to a simpler approach late into our project, we abandoned the hybrid model plan and decided to move forward with a single model applied on the entire data set. For the new implementation plan, we distributed the work in the following way: Fermi and Rodney were responsible for the data processing and choosing the models which to be explored; Mark and Adelina held responsibility for the hyperparameter tuning of the three models, and the writing of the report. While we realize this is not ideal, and it would be good to have us all collaborate on a single model, we simply ran out of time and compensated.

Appendix

Survey Questions:

Answer the following questions about the city of: ***Dubai / Rio de Janeiro / New York City / Paris***

Q1

From a scale 1 to 5, how popular is this city? (1 is the least popular and 5 is the most popular)

Q2

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)

Q3

Rate the city's architectural uniqueness from 1 to 5, with 5 being a blend of futuristic wonder and historical charm.

Q4

Rate the city's enthusiasm for spontaneous street parties on a scale of 1 to 5, with 5 being the life of the celebration.

Q5

If you were to travel to this city, who would be likely with you? (Choose from Co-worker, Friends, Partner, Siblings)

Q6

Rank the following words from the least to most relatable to this city. Each area should have a different number assigned to it. (1 is the least relatable and 6 is the most relatable)

Q7

In your opinion, what is the average temperature of this city over the month of January? (Specify your answer in Celsius)

Q8

How many different languages might you overhear during a stroll through the city?

Q9

How many different fashion styles might you spot within a 10-minute walk in the city?

Q10

What quote comes to mind when you think of this city?