Predicting Airbnb New User Booking Destination

COGS 118B

Angela Zhang, Kyle Sung, Mark Choe, Nicholas Martz, Preston Wong

Introduction and Motivation

Airbnb users have over 190 countries to choose from for their first booking. If we can accurately predict which country a new user will book their first trip in, it would allow Airbnb to provide more personalized advertisements through emails, on their homepage, and from third-party advertisers. They could also provide coupons to entice potential customers and decrease the amount of time customer's wait before booking. This problem came from a competition Airbnb personally hosted on Kaggle.

Related Work and Literature

1. Research paper on "Key Factors Affecting the Price of Airbnb Listings"

- Global regression models (GLM)
  - OLS regression
  - quantile regression
  - Can investigate factors on Airbnb's listing price
- Also used predictor variables related to price determinants
- Difference is that this model ignoring existence of local variations; added a Geometric weighted regression model to include that
- Interesting thing that they looked at was how different parts of a city ---> downtown, suburban, metro areas affected how the prices were listed, and they used the model to predict the dependent variable (price)
- Key Difference is that they did not add in a lot of variables → Gender, Language, etc.
- Focused on Distance and Reviews, while accounting for age

Related Work and Literature (cont.)

2. Interview with the 2nd Place Winner of the Airbnb Kaggle Contest:

- Goal was the same: predict Airbnb new user booking destination
- Key difference is that they used 1,312 features and we used 19 features (They one-hot encoded all the different categorical features)
- Otherwise, it was a similar process of cleaning up the data (abnormal values/outliers) and one-hot encoding the different categorical features we chose to use in order to create a predictive model
- The final model used Boosting, which we did not use as one of our models since it was out of our scope

Methods

Data Cleaning

Dataset: www.kaggle.com/c/airbnb-recruiting-new-user-bookings/data

- **16 total columns**
    - id: user id
    - date_account_created: the date account creation
    - Timestamp_first_active: timestamp of first activity
    - Date_first_booking: date of first booking
    - Gender
    - Age
    - Signup_method
    - Signup_flow: page a user came to signup from
    - Language
    - Affiliate_channel: what kind of paid marketing
    - Affiliate_provider: where the marketing is e.g. google, craigslist, other
    - First_affiliate_tracked: the first marketing the user interacted with before signing up
    - Signup_app
    - First_device_type
    - First_browser
    - Country_destination: target variable to predict

Data Cleaning (cont.)

- We dropped columns that are not useful to our prediction and analysis
- These are the final features we have kept
- From those features, we then turn non-numerical values including NaN into numerical representations

	date_account_created	date_first_booking	gender	age	language	country_destination
0	2010-06-28	NaN	-unknown-	NaN	en	NDF
1	2011-05-25	NaN	MALE	38.0	en	NDF
2	2010-09-28	2010-08-02	FEMALE	56.0	en	US
3	2011-12-05	2012-09-08	FEMALE	42.0	en	other
4	2010-09-14	2010-02-18	-unknown-	41.0	en	US

Cleaning our features

- Gender
  - Fill -unknown- values with NaN values
  - One-hot encode categories (Male, Female, Other)
- Age
  - Drop outlier values
    - Ages older than 90 years old
    - Ages younger than 14 years old
  - Replace NaN values with the average age (36)
- Language
  - if language is specified as english (en) then value in column is set to 1 (True), 0 (False) if otherwise.

Cleaning our features (cont.)

- Country_destination
  - Map countries to a corresponding numerical value
- Date = date_first_booking - date_account_created
  - Time between date account created and first booking
  - NaT values in this column are replaced with an arbitrary value of -1 to numerically indicate
- Month_first_booked
  - One-hot encoded all months including missing NaT values

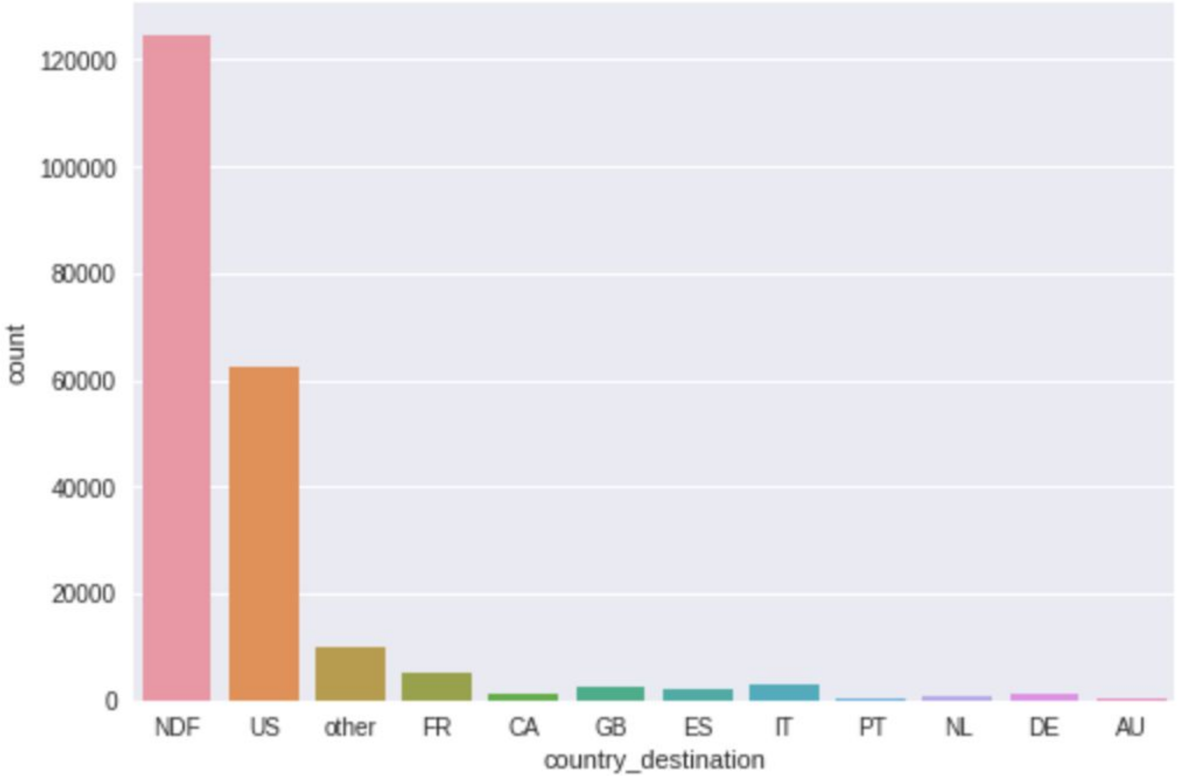
Preprocessed/Cleaned Dataset

	age	language	date	FEMALE	MALE	OTHER	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sept	Oct	Nov	Dec	NaT	country_destination
1	38.0	1	-1.0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	11
3	42.0	1	278.0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
6	46.0	1	3.0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
7	47.0	1	10.0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
8	50.0	1	206.0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

	age	language	date	FEMALE	MALE	OTHER	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sept	Oct	Nov	Dec	NaT	country_destination
213440	24.0	1	-1.0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	11
213441	34.0	1	44.0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	5
213443	36.0	1	13.0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
213445	23.0	1	2.0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
213446	32.0	1	-1.0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	11

Data Exploration

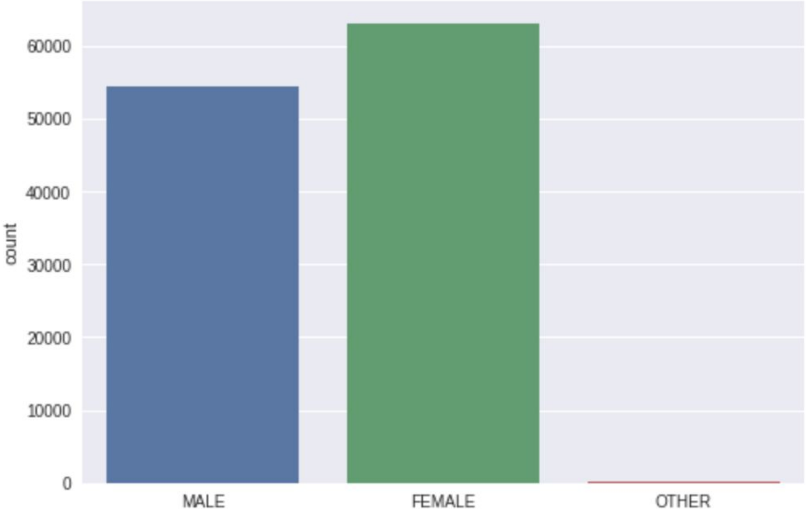
Country_destinations:

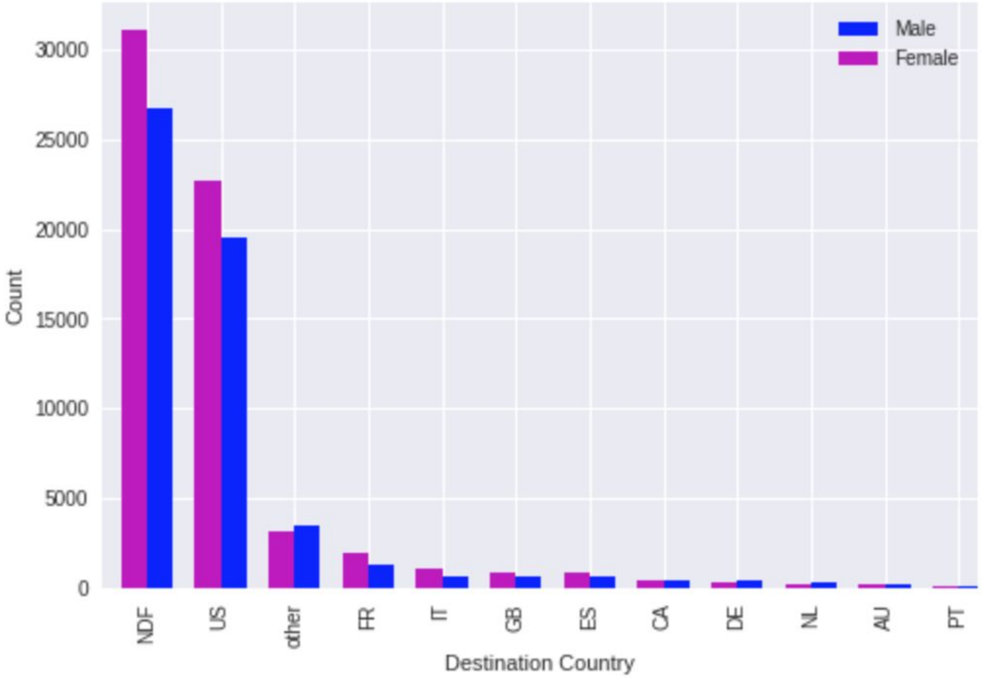


Data Exploration (cont.)

Gender data:

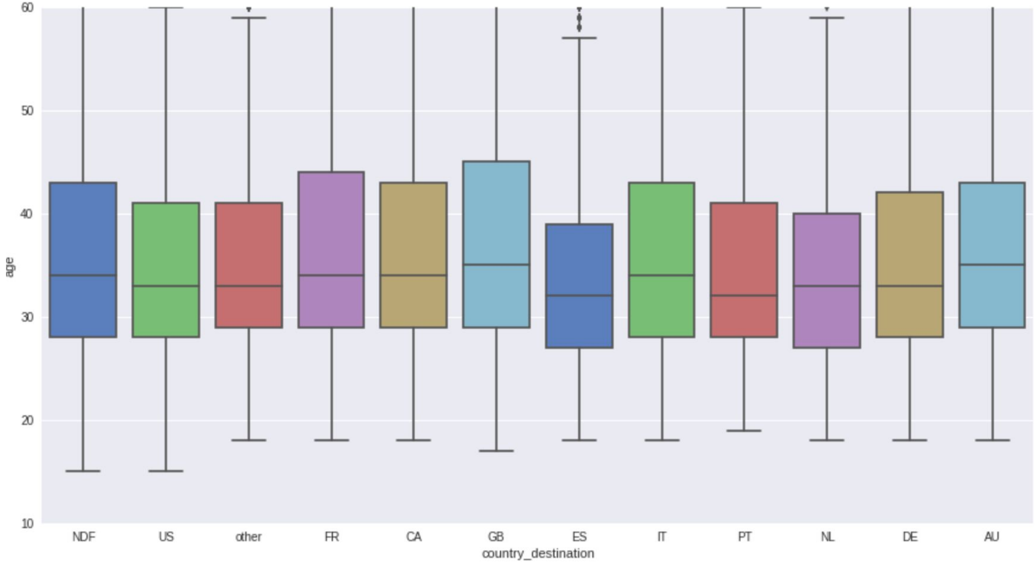
- 63,041 Females
- 54,440 Males

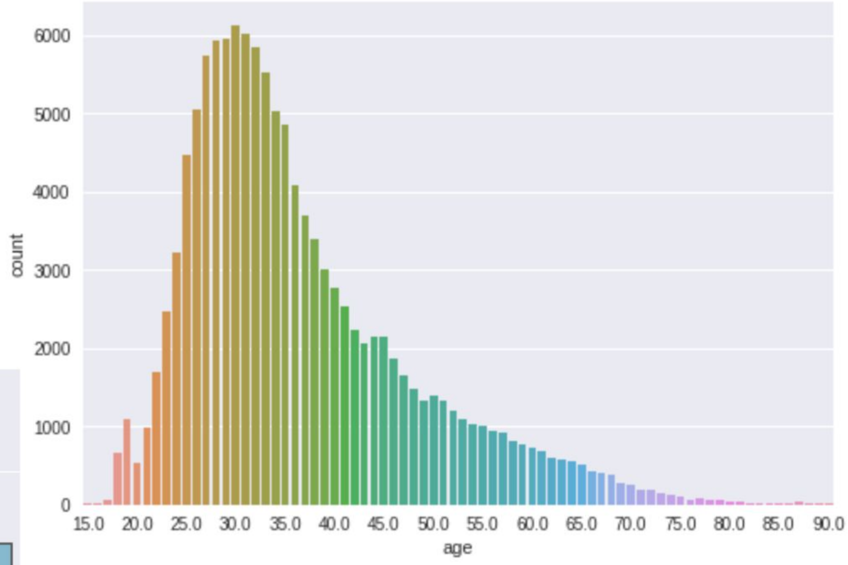




Data Exploration (cont.)

Age data:

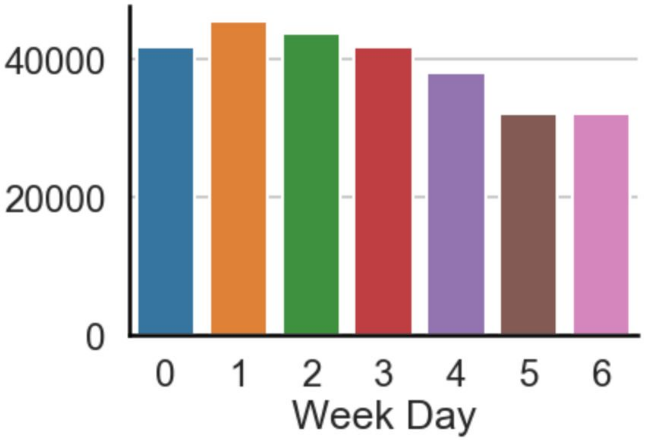


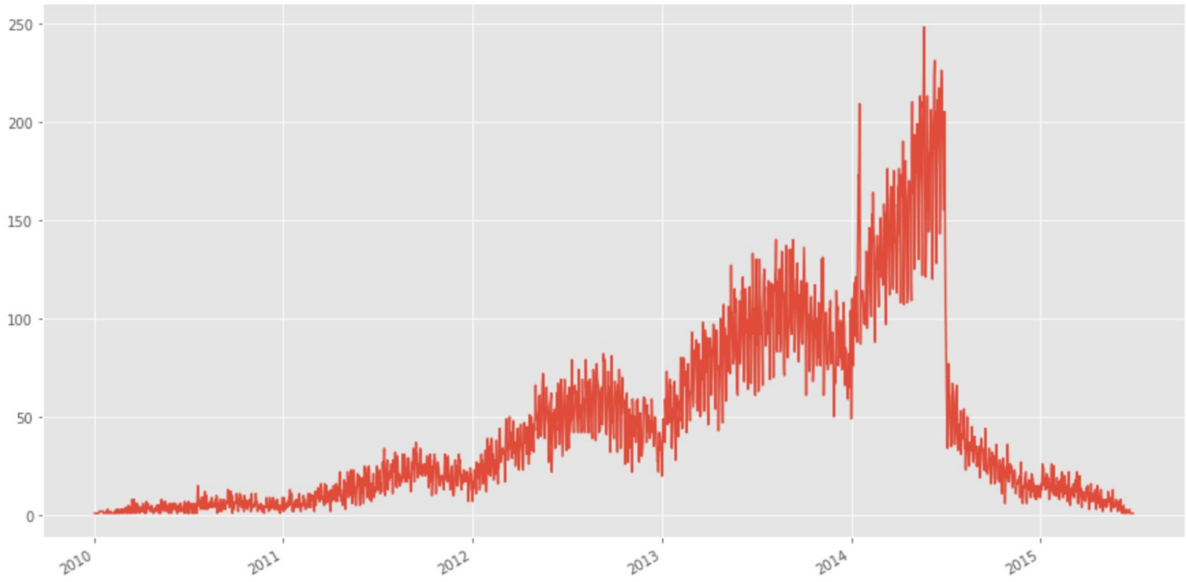



```
count    122861.000000
mean         36.463809
std          11.455232
min          15.000000
25%          28.000000
50%          33.000000
75%          42.000000
max          90.000000
Name: age, dtype: float64
```

Data Exploration (cont.)

Date_first_booked:





General Methods

- 213422 total data points after processing the data
- Goal was to predict the “country_destination” column based on the other features
- Used 5-fold cross-validation or an 80/20 ratio for splitting the data into subsets for training and testing
- Calculated average accuracy and error for each model to compare between them

Logistic Regression

- "Linear Classifier"
- Intuitively straight-forward for multiclass classification
- Simplistic and can be used as a baseline to measure performance of more complex models
- One vs Rest (Training one for each number)

Logistic Regression: Algorithm

- Overview:
  - Logistic Regression essentially converts our probabilities to binary
  - For this problem, comparing confidence probabilities per class
  - Find the highest and label everything accordingly
  - Implemented K-Fold cross validation

Logistic Regression: Predicted

y_pred - NumPy array

	0	1	2	3	4	5	6	7	8	9	10	11
0	0.0871819	0.709788	0.0636518	0.0178339	0.0233748	0.0303788	0.0341382	0.00278414	0.0117512	0.0146383	0.00447001	9.05664e-06
1	0.11533	0.705128	0.037581	0.0206986	0.0224438	0.0360754	0.0279972	0.00223236	0.0112302	0.016693	0.0045789	1.21335e-05
2	3.89849e-05	4.59291e-05	5.45318e-05	4.47956e-05	5.68917e-05	4.35784e-05	4.99309e-05	4.86399e-05	4.49077e-05	4.58972e-05	6.19901e-05	0.999464
3	3.8571e-05	4.92767e-05	4.78966e-05	4.14343e-05	4.73569e-05	4.87551e-05	4.68123e-05	4.98558e-05	4.56848e-05	4.57132e-05	5.51447e-05	0.999483
4	4.77985e-05	4.58811e-05	4.55105e-05	5.02018e-05	4.53418e-05	4.35264e-05	4.87858e-05	4.73716e-05	4.4266e-05	3.89106e-05	4.04755e-05	0.999502
5	3.95001e-05	4.21193e-05	6.39726e-05	4.93088e-05	7.13008e-05	3.79551e-05	5.40555e-05	4.71837e-05	4.3969e-05	4.61243e-05	7.1592e-05	0.999433
6	4.77985e-05	4.58811e-05	4.55105e-05	5.02018e-05	4.53418e-05	4.35264e-05	4.87858e-05	4.73716e-05	4.4266e-05	3.89106e-05	4.04755e-05	0.999502
7	4.77985e-05	4.58811e-05	4.55105e-05	5.02018e-05	4.53418e-05	4.35264e-05	4.87858e-05	4.73716e-05	4.4266e-05	3.89106e-05	4.04755e-05	0.999502
8	0.114562	0.695013	0.0422094	0.021772	0.0259732	0.0346391	0.0300578	0.00221251	0.0112854	0.0172315	0.00503481	9.72761e-06
9	4.96737e-05	4.91596e-05	3.39639e-05	4.81496e-05	3.96585e-05	4.95408e-05	3.34561e-05	4.94864e-05	5.66438e-05	5.99117e-05	5.31172e-05	0.999477
10	4.77985e-05	4.58811e-05	4.55105e-05	5.02018e-05	4.53418e-05	4.35264e-05	4.87858e-05	4.73716e-05	4.4266e-05	3.89106e-05	4.04755e-05	0.999502

Format Resize Background color

Cancel OK

k-Nearest Neighbors

- Easy to use for multiclass classification
- Factors to consider:
  1. Dimensionality of data: since similarity is calculated by euclidean distance, kNN suffers from the curse of dimensionality
  2. Size of k: small k values over-fit to noise of neighboring data points, but large k values can possibly under-fit and favor dominant classes by smoothing out too much

kNN: Dimensionality of data

- First ran kNN using all predictors, which was 19 total since we one-hot encoded gender and month of first booking
- Dropped all month predictors and the “Other” gender predictor, so there were 5 predictors left: age, language, days between making account and first booking, female and male

kNN: Size of k

- Began by testing a wide range of k values: [5, 10, 25, 100, 461] and comparing error rate/accuracy
- 461 = sqrt(213422) = sqrt(total # data points) which is a commonly used method for choosing k
- Narrowed most accurate model down to between k=25 to k=100, then between k=50 to k=70

kNN: Predicting More Than NDF and US

- Since the data strongly favored the NDF and US, the predicted results using all the data only contained NDF and US values
- Modified the dataset in an effort to predict other countries as well:
    - Removed all NDF data points
    - Removed 20% of the US data points
    - Resulting data was: 76,624 data points with 50,000 US
- Ran model with a few different k values as well

Decision Tree

- Inherently multiclass
- All default parameters
- GridSearchCV function to test max depth values of [1, 3, 5, 7, 9].
- Compared the mean training scores and the mean test scores of the gridsearch.
- Predicted a test accuracy using the optimal depth value, and compared with the other accuracies.

Neural Network - MLP Classifier

- Inherently multiclass
- Default Parameters: “adam” solver, alpha (L2 penalty), hidden units, etc.
  - Non-Default Parameter: adaptive learning rate, shuffled
- GridSearchCV function to test different amounts hidden layers [1, 2].
- Compared the mean training scores and the mean test scores of the gridsearch.
- Predicted a test accuracy using optimal hidden layer, and compared with the other accuracies.

Results

What did you discover? - Logistic Regression

- NDF Baseline = 58.4%
- Logistic Regression 5-fold CV = 88.1%
- Without NDF = 70%
- We expect this to work well because our data could be thought as being linearly separated (US vs NDF)

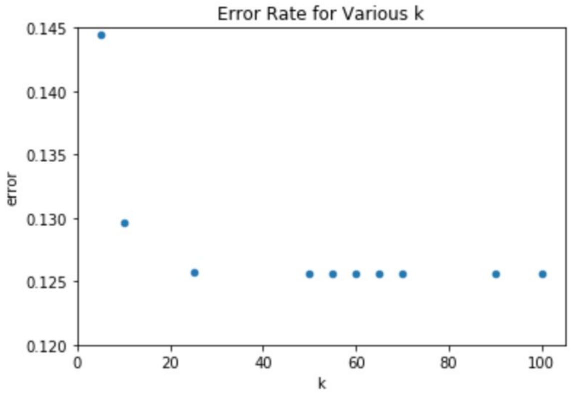
What did you discover? - kNN

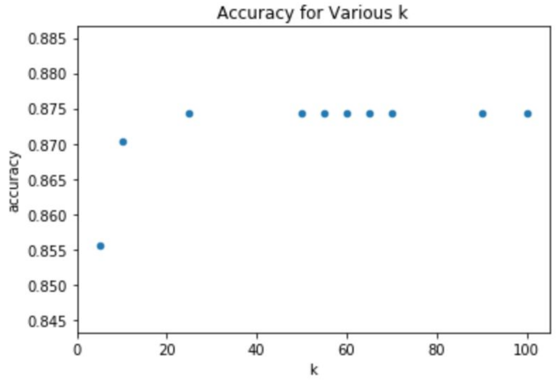
- Using less features actually gave us better accuracy, which matches the fact that higher dimensions could suffer from the curse of dimensionality and be less accurate
- With k=60:

	19 Predictors	**5 Predictors**
Error rate	0.12674	0.12533
Accuracy	0.87325	0.87466

What did you discover? - kNN (cont.)

	k	error	accuracy
0	5	0.144524	0.855476
1	10	0.129624	0.870376
2	25	0.125688	0.874312
3	50	0.125618	0.874382
8	55	0.125618	0.874382
9	60	0.125618	0.874382
10	65	0.125618	0.874382
4	70	0.125618	0.874382
5	90	0.125641	0.874359
6	100	0.125641	0.874359
7	461	0.125829	0.874171





- The optimal k value was between 50 and 70, which all resulted in the same error and accuracy up to the 6th decimal place
- The error and accuracy begin to be similar around k=25, so if we wanted to save time we could use k=25 as well

What did you discover? - kNN (cont.)

-Modifying our data by removing NDF data points and using less US data points gave us results that predicted other countries with lower k values, but the accuracy actually decreased

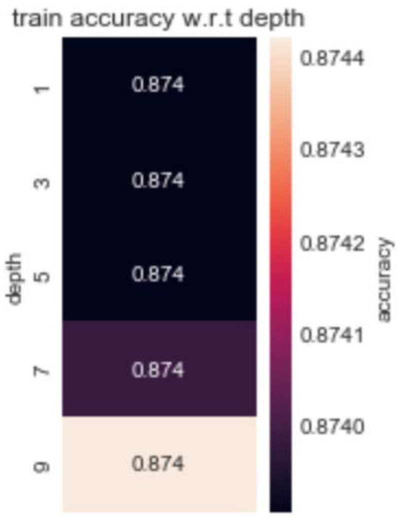
-This means that the different country predictions are most likely due to noise rather than better predictions

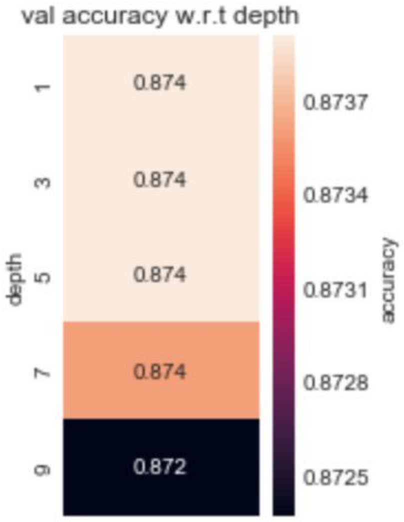
```
Using k=5 we predict countries: {0, 1, 2, 3, 4, 5, 6, 8, 9}
Using k=10 we predict countries: {0, 1, 2, 5, 6}
Using k=30 we predict countries: {0, 1}
Using k=60 we predict countries: {1}
Using k=461 we predict countries: {1}
```

	k	error	accuracy
0	5	0.392421	0.607579
1	60	0.367984	0.632016
2	100	0.352957	0.647043
3	461	0.352761	0.647239
4	1000	0.352761	0.647239

What did you discover? - Decision Tree

- There is little to no difference between the max depths





Testing Accuracy

w/ Optimal Max Depth (1):

0.8829799695443364

Credit to UCSD's 118A instructional staff for the heatmap function

What did you discover? - MLP Classifier

- There was little difference found by adding a hidden layer.
- Training Accuracy:
  - 1 Hidden Layer: 0.87204581
  - 2 Hidden Layers: 0.8722889
- Validation Accuracy from Training Data:
  - 1 Hidden Layer: 0.67471811
  - 2 Hidden Layers: 0.67471608
- Testing Accuracy w/ Optimal Amount of Hidden Layers (1):
  - 0.8829799695443364

What did you discover? - Direct Comparison

	Logistic Regression	k-Nearest Neighbors	Decision Tree	MLP Classifier
Accuracy	0.8810	0.8743	0.8830	0.8830

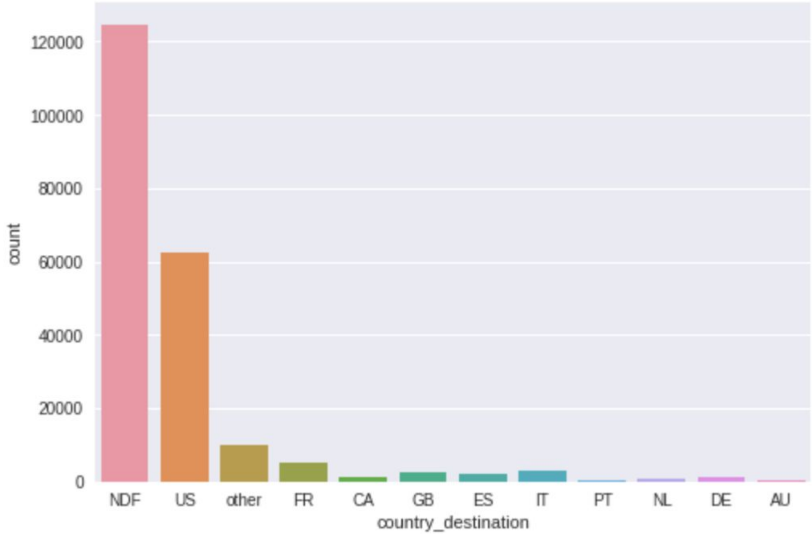
Discussion

How hard was it?

- Multiclass Problem
- Data Preprocessing (Notably Missing Data)
- Data Highly Skewed (NDF vs US)
- Long Runtimes for kNN and MLP Classifiers

Why did it not work?

- Data is skewed:



- Possibly more ideal ways of handling the missing values
- Interpreting which features to use and which not to use

Possible Improvements

- One improvement we could make is filling out our missing values in a better way (EM).
  - We used Mean Imputation for Age
    - Does not preserve the relationship between variables/predictors
  - Filling in difference in timestamps in more predictive way
    - Entered “-1” when user never booked
- Using tensorflow with a GPU cluster instead of sklearn for the MLP classifier to decrease runtime.
  - Decreasing the runtime to fit the model, allows for hidden layer parameters to be tested.

Next Step

- Since our data, when a user booked, is heavily skewed towards the US, we could add another predictor which takes into account which US cities they booked to.
- Expanding on this topic, we could do all global major regions.
  - Specialized prediction
- Incorporate the predictors “Date account created” and” timestamp of first activity” = peak interest
  - We can use this for seasonal predictions
  - This way, our model can predict which city a user might travel to depending on our existing features (age, month, etc..)

What did we learn?

- Reason why data is skewed?
  - Airbnb is more supportive in the United States
    - Most popular destinations are located in the US
    - More widely used among population in the US
    - Dataset 2010-2015 early stages
      - Mostly local support (booked within US)
- Data is linear separable between (NDF vs US) hence our accuracy ~88%

References

Kaggle Competition:
https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings

Literature 1:
https://www.mdpi.com/2071-1050/9/9/1635/pdf

Literature 2:
http://blog.kaggle.com/2016/03/17/airbnb-new-user-bookings-winners-interview-2nd-place-keiichi-kuroyanagi-keiku/

Code: https://github.com/angela278/airbnb_predictions