Capstone 1: Recommender System - Data Storytelling

The goal of this project is to build a recommendation engine to recommend practice problems to students. The first milestone is to predict the number of attempts a student will take to solve a given problem. The data provided by Analytics Vidhya comes in three separate tables. The details of each dataset is described below.

1. **train_submissions.csv** - This contains 1,55,295 submissions which are selected randomly from 2,21,850 submissions. Contains 3 columns ('user_id', 'problem_id', 'attempts_range'). The variable '**attempts_range**' denoted the range no. in which attempts the user made to get the solution accepted lies.

   We have used following criteria to define the attempts_range :-

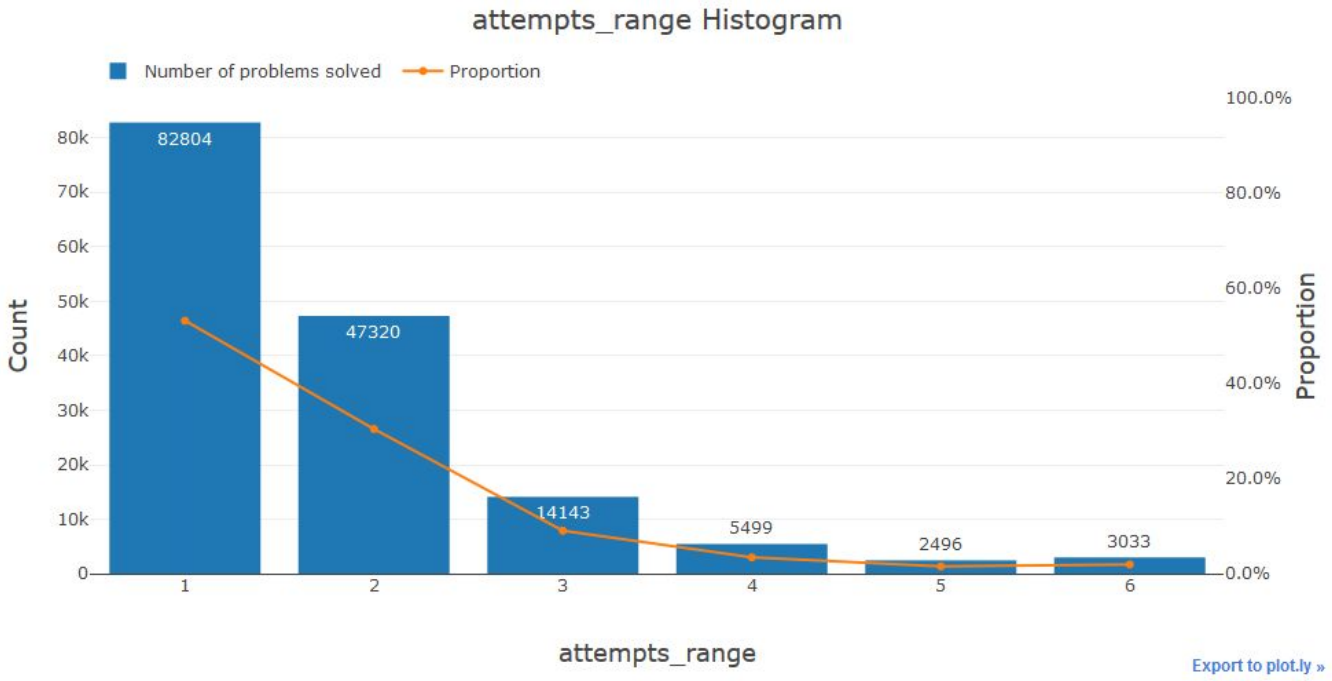
attempts_range	No. of attempts lies inside
1	1-1
2	2-3
3	4-5
4	6-7
5	8-9
6	>=10

1. **user_data.csv** - This is the file containing data of users. It contains the following features :-
    1. user_id - unique ID assigned to each user
    2. submission_count - total number of user submissions
    3. problem_solved - total number of accepted user submissions
    4. contribution - user contribution to the judge
    5. country - location of user
    6. follower_count - amount of users who have this user in followers
    7. last_online_time_seconds - time when user was last seen online
    8. max_rating - maximum rating of user
    9. rating - rating of user
    10. rank - can be one of 'beginner' ,'intermediate' , 'advanced', 'expert'
    11. registration_time_seconds - time when user was registered

1. **problem_data.csv** - This is the file containing data of the problems. It contains the following features :-
    1. problem_id - unique ID assigned to each problem
    2. level_id - the difficulty level of the problem between 'A' to 'N'
    3. points - amount of points for the problem
    4. tags - problem tag(s) like greedy, graphs, DFS etc.

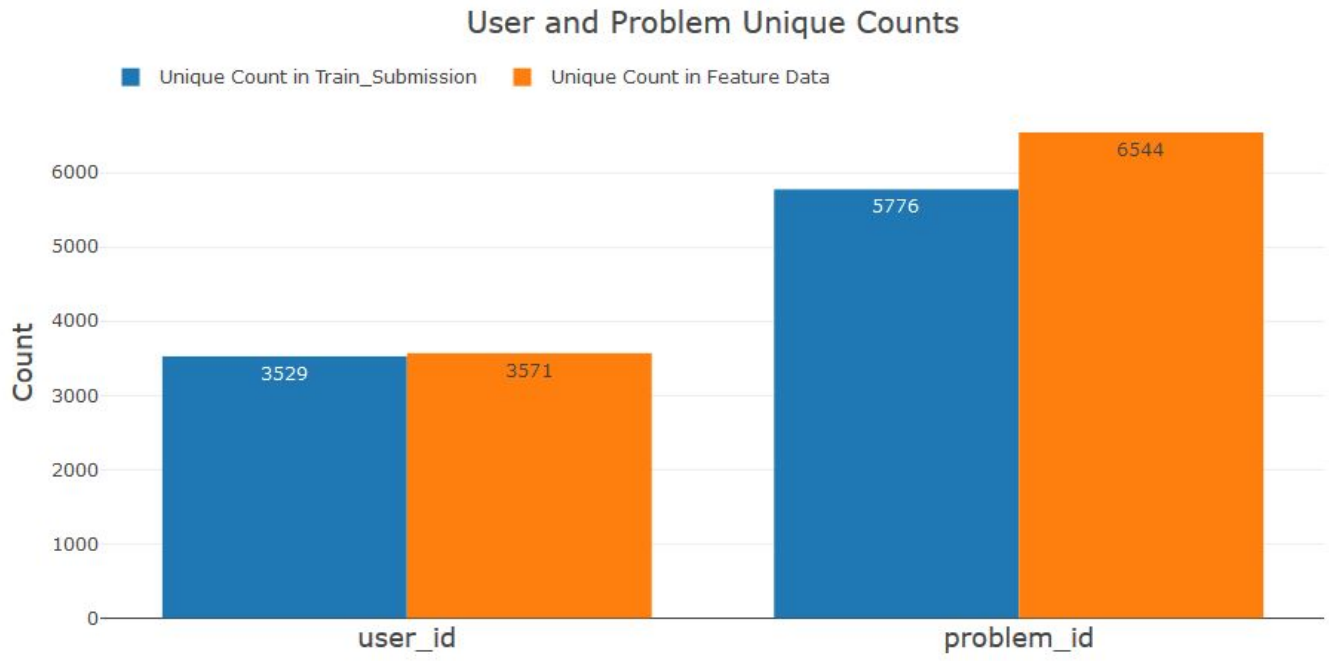
In this part of the project, I explore several interesting questions about these datasets. To view the full analysis, see the Data Storytelling jupyter notebook. The full project can be viewed on the github site here.

What attempts_range is the most common?



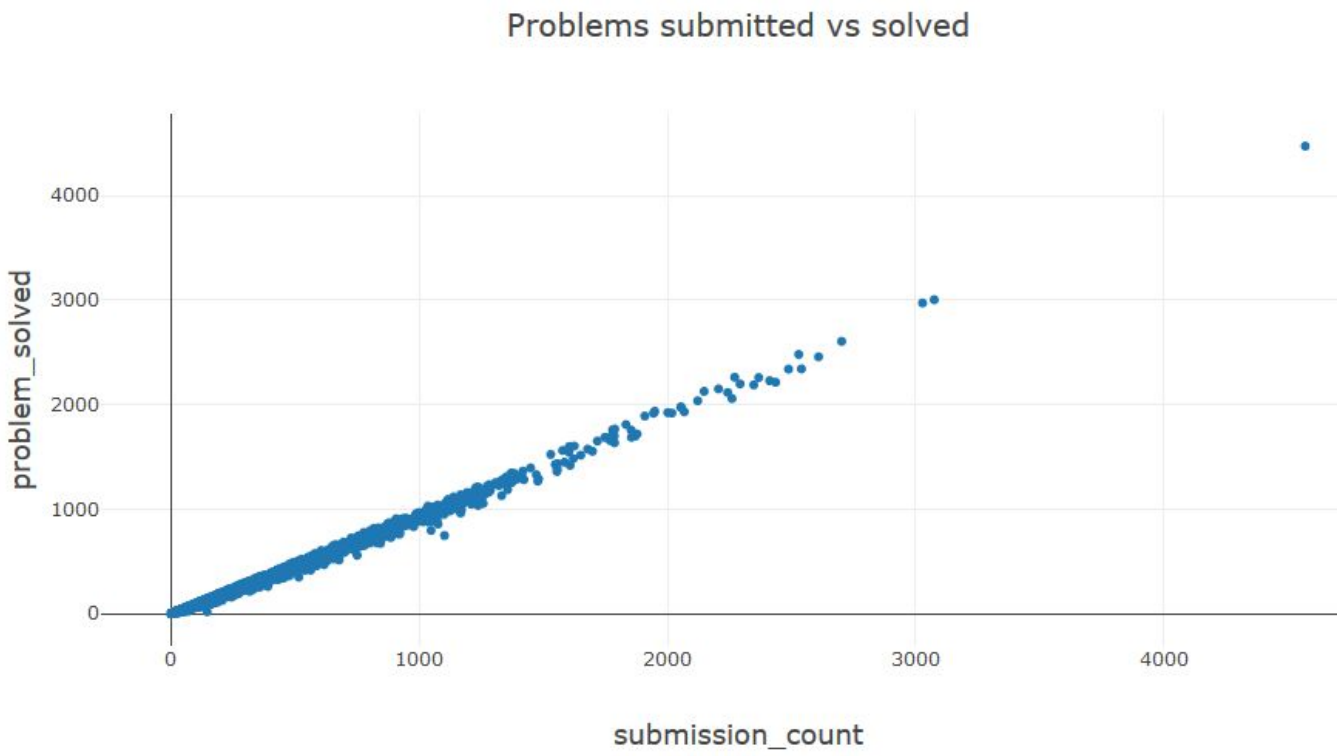
The histogram above shows both the count and proportion of completed problems by attempts_range. 53% of problems are solved in a single attempt. 30.5% of problems are solved between 2 to 3 attempts and this drops quickly to 9.1% of problems being solved in 4 to 5 attempts. Because the provided data was already binned, there is no way to know what proportion of problems were solved for a specific number of attempts other than 1 attempt.

How many unique users and problems are there?



The bar plot above shows the unique counts for user_id and problem_id both in the train_submission dataset (blue) and the feature data (orange). The feature data are just the other two tables, one for users and one for problems. The feature data has a higher unique count in both cases. This is what I would expect since the user and problem metadata should be collected for every user and problem. But not every user will necessarily have solved at least one problem and not ever problem will necessarily have been solved at least once, these would thus not be included in the train_submission data.

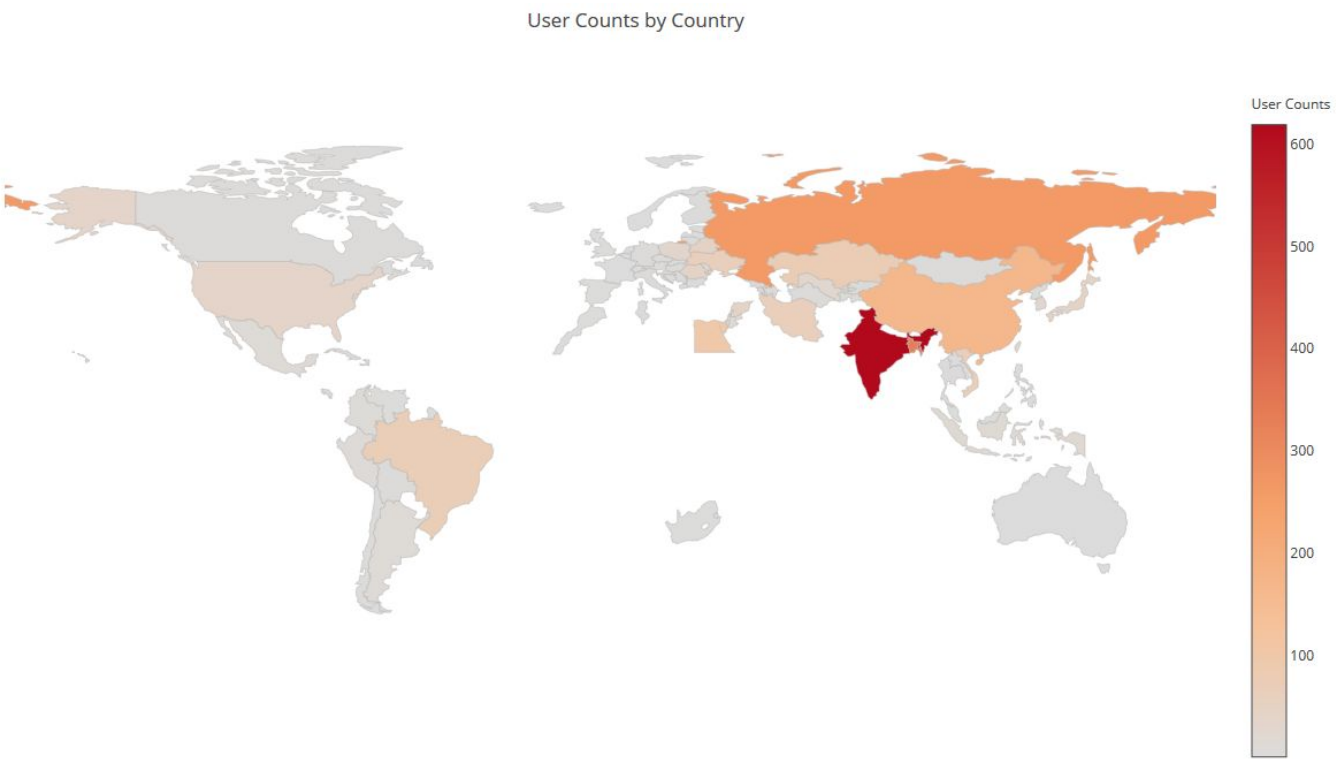
What's the relationship between problems solved vs problems submitted?



The scatter plot above shows that there is a high correlation between problem_solved and submission_count. This is something I'll have to be aware of when building the regression model as collinearity in the features can cause repeatability issues in making predictions.

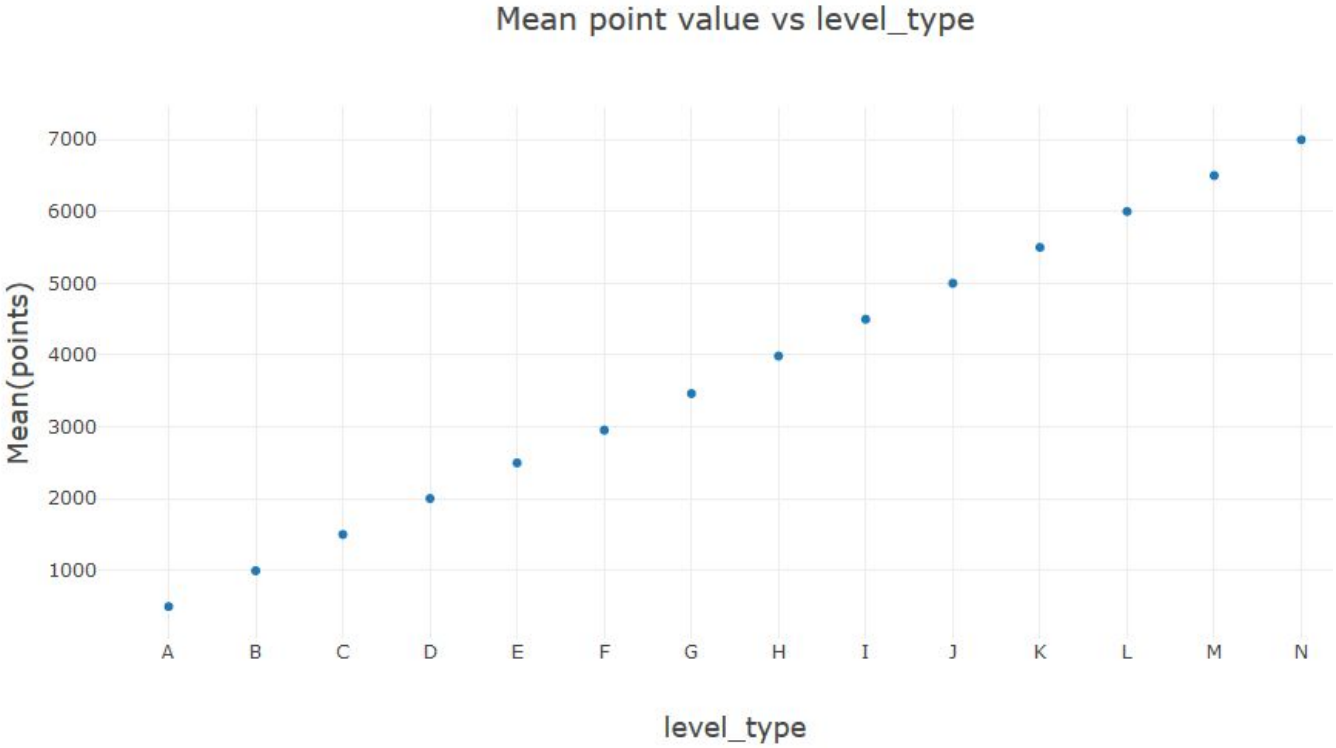
What countries are users from?

The majority of users do not actually have a recorded country of residence, but I can look at those that do. To visualize where users are predominantly from, I plotted user counts by country in a choropleth. The map below shows that the majority of users are in Asia, with the most (619 unique users) coming from India.



What is the correlation between points and level_type?

Recall that we inferred from the problem data that the point value associated with each level_type increased by 500 point increments with increasing level_type. I then used this assumption to fill in missing point or level_type values. This relationship is shown in the scatter plot below. Benchmarking the model will reveal whether or not this inference introduces too much bias in the model.


Mean point value vs level_type
Mean(points)
7000
6000
5000
4000
3000
2000
1000
A
B
C
D
E
F
G
H
I
J
K
L
M
N
level_type