Predicting churn for CodeCombat players

By Jenny Temple

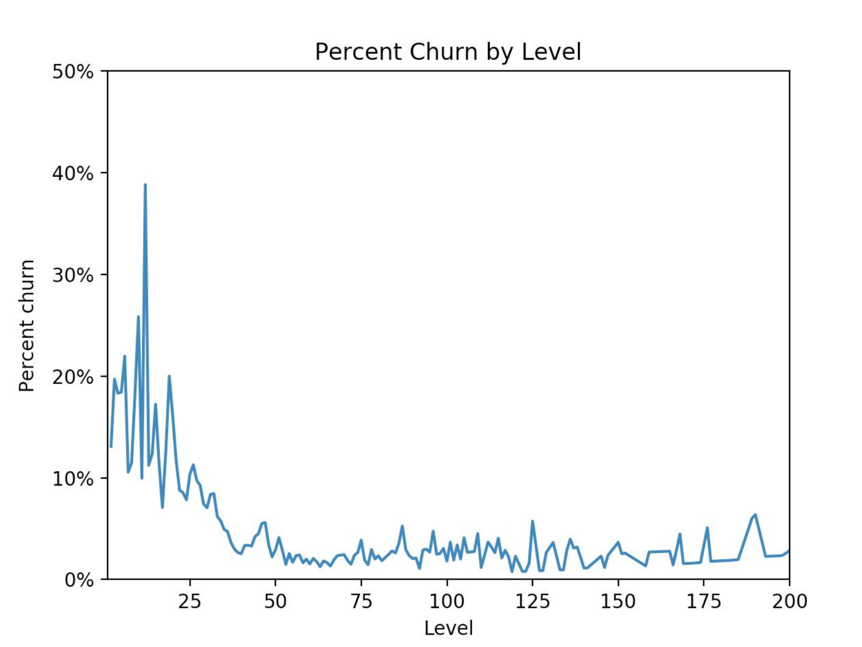
CodeCombat Background

CodeCombat gamifies coding to teach young people how to write code

Project Goal

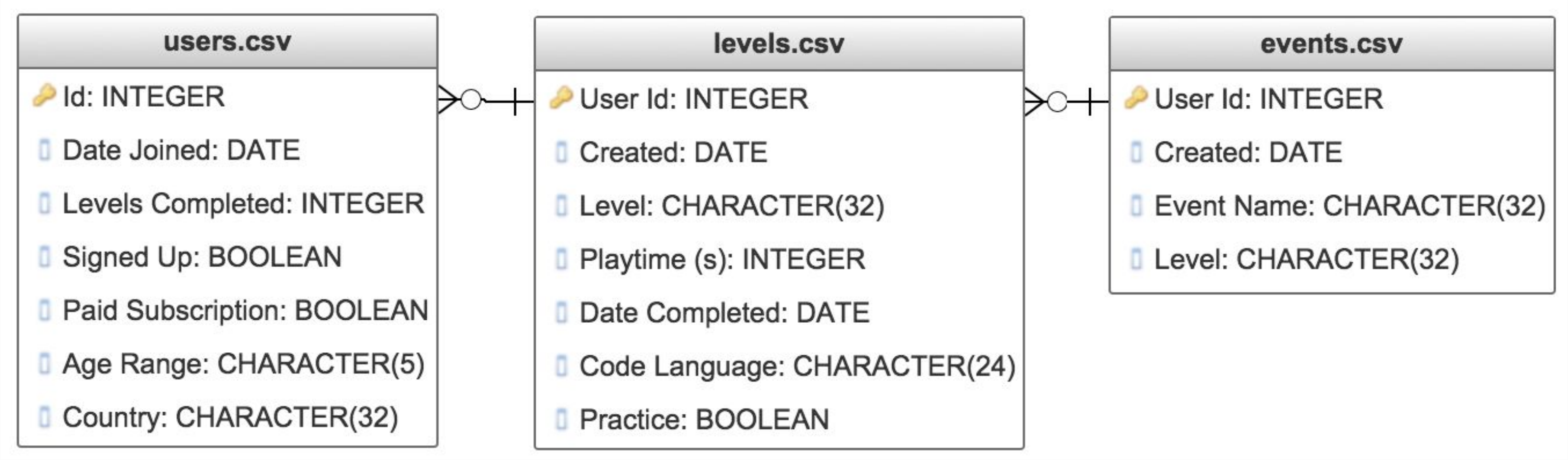
- Identify users who will churn
  → intervene to retain players

- Uncover drivers of churn
  → develop strategies to extend user play



Data

- Data for users joining in March 2017
- User data (160k), level data(1.3M), event data (5.2M)



users.csv

Id: INTEGER
Date Joined: DATE
Levels Completed: INTEGER
Signed Up: BOOLEAN
Paid Subscription: BOOLEAN
Age Range: CHARACTER(5)
Country: CHARACTER(32)

levels.csv

User Id: INTEGER
Created: DATE
Level: CHARACTER(32)
Playtime (s): INTEGER
Date Completed: DATE
Code Language: CHARACTER(24)
Practice: BOOLEAN

events.csv

User Id: INTEGER
Created: DATE
Event Name: CHARACTER(32)
Level: CHARACTER(32)

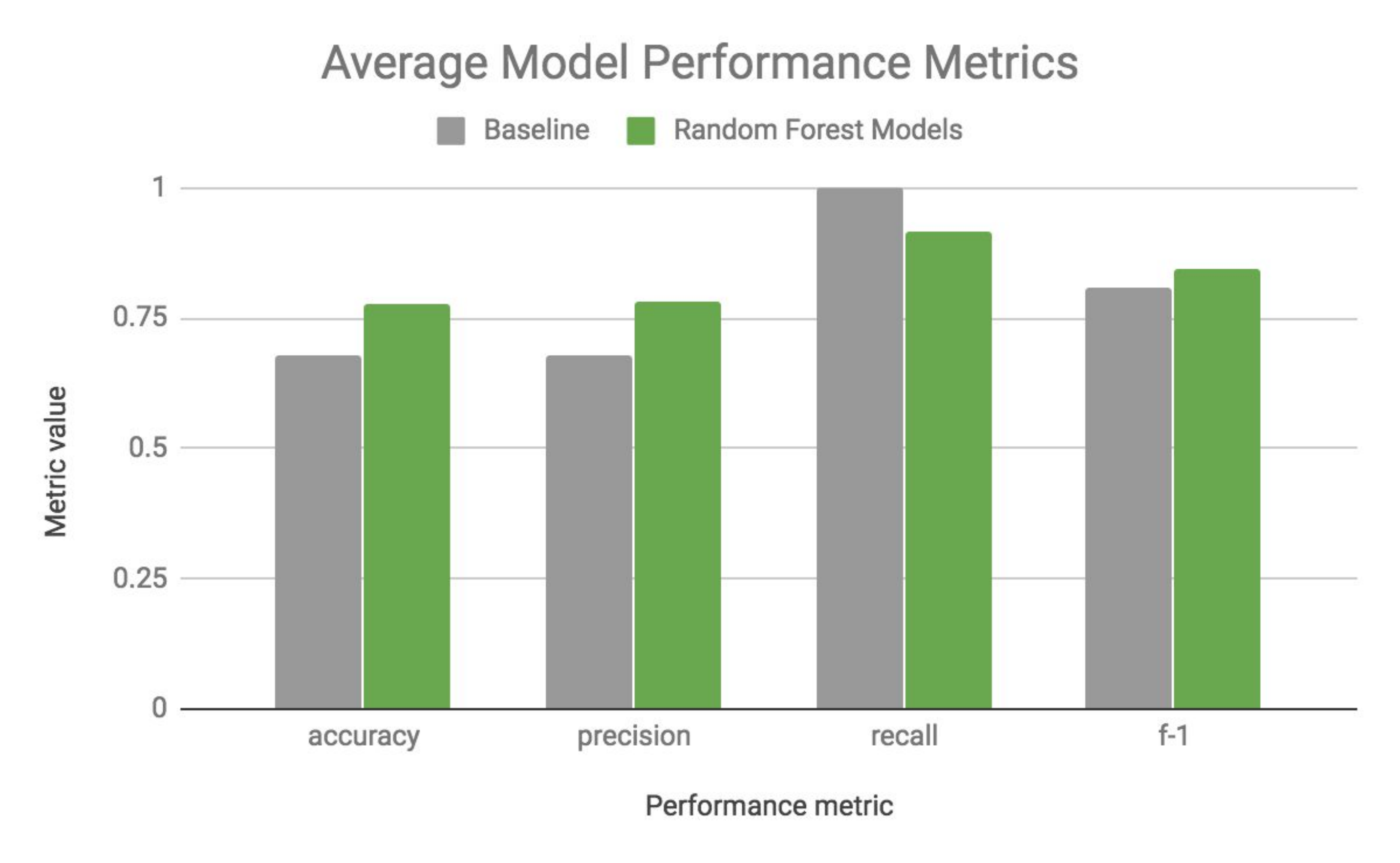
Methodology

Build a collection of models to predict churn at different game points

- Build features describing user experience at different game points
- To predict classification (churn/keep playing)
- Using five Random Forest models
- With tuned parameters

Models	Use data through	To predict churn at
Model 1	Level 5	Level 10
Model 2	Level 10	Level 15
Model 3	Level 15	Level 30
Model 4	Level 30	Level 60
Model 5	Level 60	Level 100

Results: Predictive Power



Results: Drivers of churn

What drives early churn?

- Registered account
- Time to complete levels
- Age
- Number of times started levels

What drives later churn?

- Number of times started levels
- Recent practice levels
- Rate of bugs in recent user code
- Time to complete mid-range levels

Results: Business recommendations

To retain players at higher game levels:

- Suggest practice levels
  - More frequently
  - More aggressively
- Provide additional support for users with bugs
  - Resources
  - Encouragement

Future Work

- Tune as hyperparameters:
    - Number of models
    - Granularity of data in features
- Design A/B tests based around drivers of churn

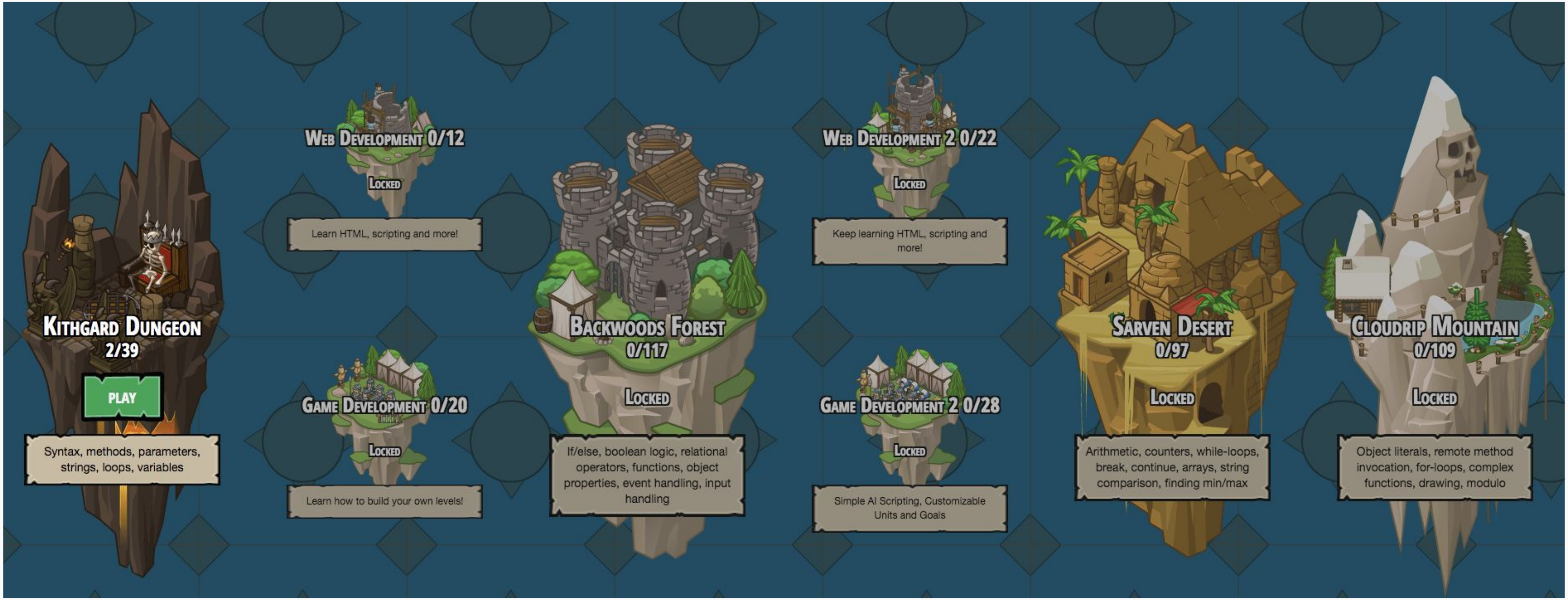
Contact

Jenny Temple

jennytemple@gmail.com
https://github.com/jennytemple
https://www.linkedin.com/in/jenny-temple/

Appendix

Appendix: Campaigns



Kithgard Dungeon 2/39

Syntax, methods, parameters, strings, loops, variables

Web Development 0/12

Learn HTML, scripting and more!

Game Development 0/20

Learn how to build your own levels!

Backwoods Forest 0/117

If/else, boolean logic, relational operators, functions, object properties, event handling, input handling

Web Development 2 0/22

Keep learning HTML, scripting and more!

Game Development 2 0/28

Simple AI Scripting, Customizable Units and Goals

Sarven Desert 0/97

Arithmetic, counters, while-loops, break, continue, arrays, string comparison, finding min/max

Cloudrip Mountain 0/109

Object literals, remote method invocation, for-loops, complex functions, drawing, modulo

Appendix: Predictive Power

Models	Use data through	To predict churn at	Baseline accuracy	Model accuracy	Baseline precision	Model precision
Model 1	Level 5	Level 10	0.60	0.72	0.60	0.73
Model 2	Level 10	Level 15	0.68	0.71	0.68	0.72
Model 3	Level 15	Level 30	0.84	0.87	0.84	0.87
Model 4	Level 30	Level 60	0.67	0.80	0.67	0.82
Model 5	Level 60	Level 100	0.60	0.78	0.60	0.77

Performance metric definitions

P = churn, N = no churn, T = correct prediction, F = incorrect prediction

Accuracy: (TP + TN)/n = % correct predictions

Precision: TP/(TP+FP) = % actually churn out of predicted to churn

Recall: TP/(TP+FN) = % correctly predicted to churn out of actually churn

F-1: 1/(.5/Precision + .5/Recall)